

DISASSEMBLED AND COMMENTED BY DARREN FREED. darrenfreed@shaw.ca
THIS DOCUMENT IS FREELY DISTRIBUTABLE, SO LONG AS THIS ACKNOWLEDGEMENT IS ATTACHED.
PLEASE NOTIFY ME WITH ADDITIONS AND CORRECTIONS.

A/D CHANNELS:

```
CH #0: KNOCK SENSOR
CH #1: CTS
CH #2: TPS
CH #3: NOT USED
CH #4: TRANS TEMP
CH #5: BATT VOLTS
CH #6: O2 SENSOR
CH #7: QDM A
CH #8: QDM B
CH #9: IAT
CH #A: A/C PRESSURE SENSOR
CH #B: TEST CHANNEL
```

```
0000 ;  
0000 ; M6811 Disassembler Generated Source Code  
0000 ;  
0000 ; For User Control File: BSBH.CTL  
0000 ; Program File: BSBH.BIN  
0000 ; Disassembly into File: BSBH.DIS  
0000 ;  
0000  
0000  
0000 L0000 = 0x0000  
0000 L0001 = 0x0001  
0001 L0002 = 0x0002  
0002 L0003 = 0x0003  
0003 L0006 = 0x0006  
0006 L0008 = 0x0008  
0008 L000C = 0x000C  
000C L000D = 0x000D  
000D L000E = 0x000E  
000E L0010 = 0x0010 ;  
0010 L0011 = 0x0011 ;MINOR LOOP COUNTER  
  
;B0 6.25 MSEC  
;B1 12.5 MSEC  
;B2 25 MSEC  
;B3 50 MSEC  
;B4 100 MSEC  
;B5 200 MSEC  
;B6 400 MSEC  
;B7 800 MSEC  
  
0011 L0012 = 0x0012 ;?NVMWD  
  
;B0 1 = O2 SENSOR READY  
;B1 1 = VATS PASSED - FUEL CUT WHEN CLEAR  
;B2  
;B3 SIMILAR USAGE AS $5B  
;B4 SIMILAR USAGE AS $5B  
;B5  
;B6 1 = IAC RESET ENABLED  
;B7  
  
0012 L0013 = 0x0013 ;"CURRENT" MALF STATUS FLAG 1  
  
;B0 1 = DTC P0123 - TPS VOLTAGE HIGH  
;B1 NOT ENABLED  
;B2 1 = DTC P0341 - CAM SIGNAL ERROR  
;B3 1 = DTC P0321 - 24x CRANK SIGNAL  
;B4 1 = DTC P1630 - LOW SYSTEM VOLTAGE  
;B5 1 = DTC P0117 - LOW CTS MALF
```

| | | | |
|------|---------|--------|---|
| | | | ;B6 1 = DTC P0118 - HIGH CTS MALF ;B7 1 = DTC P0134 - HO2S ERROR |
| 0013 | L0014 = | 0x0014 | ;"CURRENT" MALF STATUS FLAG 2 ALDL LIST ;B0 1 = NOT ENABLED ;B1 1 = NOT ENABLED ;B2 1 = NOT ENABLED ;B3 1 = DTC P1640 - QDM A MALF ;B4 1 = DTC P0113 - IAT HIGH TEMP ;B5 1 = DTC P0502 - NO VSS SIGNAL ;B6 1 = DTC P0112 - IAT LOW TEMP ;B7 1 = DTC P0122 - TPS LOW VOLTAGE |
| 0014 | L0015 = | 0x0015 | ;"CURRENT" MALF STATUS FLAG 3 ALDL LIST ;B0 1 = DTC P0703 - TCC BRAKE SW ERROR ;B1 1 = DTC P0501 - VSS INT SIGNAL ;B2 1 = DTC P0755 - SHIFT SOLN B ERROR ;B3 NOT ENABLED ;B4 1 = DTC P0101 - MAF, LOW GMS/SEC ;B5 NOT ENABLED ;B6 1 = DTC P0705 - TRANS RANGE SW ERROR ;B7 NOT ENABLED |
| 0015 | L0016 = | 0x0016 | ;"CURRENT" MALF STATUS FLAG 4 ALDL LIST ;B0 1 = DTC P0712 - LOW TRANS TEMP ;B1 1 = DTC P0713 - HIGH TRANS TEMP ;B2 1 = DTC P0132 - O2 SENSOR RIGH ;B3 1 = DTC P0131 - O2 SENSOR LEAN ;B4 1 = DTC P0325 - KNOCK SENS CKT ERROR ;B5 1 = DTC P1350 - IC BYPASS ERROR ;B6 1 = DTC P0342 - CAMSHAFT SENSOR ERROR ;B7 1 = DTC P0740 - TCC ERROR |
| 0016 | L0017 = | 0x0017 | ;"CURRENT" MALF STATUS FLAG 5 ALDL LIST ;B0 1 = DTC P1650 - QDM B MALF ;B1 1 = DTC P1405 - EGR SOLN MALF ;B2 1 = DTC P1404 - EGR SOLN MALF ;B3 1 = DTC P1403 - EGR SOLN MALF ;B4 1 = DTC P1361 - IGN CONTROL ERROR ;B5 1 = DTC P1623 - PROM ERROR ;B6 NOT ENABLED ;B7 NOT ENABLED |
| 0017 | L0018 = | 0x0018 | ;"CURRENT" MALF STATUS FLAG 6 ALDL LIST ;B0 ;B1 ;B2 ;B3 ;B4 ;B5 ;B6 1 = P1626 - PASSKEY II ERROR ;B7 |
| 0018 | L0019 = | 0x0019 | ;"CURRENT" MALF STATUS FLAG 7 ALDL LIST ;B0 ;B1 ;B2 ;B3 1 = A/C PRESSURE SENSOR MALF ;B4 ;B5 ;B6 ;B7 |
| 0019 | L001A = | 0x001A | ;"CURRENT" MALF STATUS FLAG 8 ALDL LIST ;B0 ;B1 1 = P1629 - PASSKEY II SIGNAL ERROR ;B2 ;B3 |
| 001A | L001B = | 0x001B | ;"MEMORY" MALF FLAG 1 ALDL LIST |
| 001B | L001C = | 0x001C | ;"MEMORY" MALF FLAG 2 ALDL LIST |
| 001C | L001D = | 0x001D | ;"MEMORY" MALF FLAG 3 ALDL LIST |

| | | | | |
|------|-------|---|--------|---|
| 001D | L001E | = | 0x001E | ;"MEMORY" MALF FLAG 4 ALDL LIST |
| 001E | L001F | = | 0x001F | ;"MEMORY" MALF FLAG 5 ALDL LIST |
| 001F | L0020 | = | 0x0020 | ;"MEMORY" MALF FLAG 6 ALDL LIST |
| | L0021 | = | 0x0021 | ;"MEMORY" MALF FLAG 7 ALDL LIST |
| | L0022 | = | 0x0022 | ;"MEMORY" MALF FLAG 8 ALDL LIST |
| 0020 | L0023 | = | 0x0023 | ;MALF TIMER - INCREMENTED Q 10 SECONDS |
| 0023 | L0024 | = | 0x0024 | ;SUM OF ALL MEMORY MALF FLAGS |
| 0024 | L0026 | = | 0x0026 | ;EGR MALF TIMER/COUNTER |
| | L0027 | = | 0x0027 | ;EGR MALF TIMER/COUNTER |
| 0026 | L0028 | = | 0x0028 | ;EGR MALF TIMER/COUNTER |
| 0028 | L0029 | = | 0x0029 | ;EGR MALF TIMER/COUNTER |
| 0029 | L002A | = | 0x002A | ;EGR STATUS FLAG |
| | | | | ;B0 |
| | | | | ;B1 |
| | | | | ;B2 |
| | | | | ;B3 |
| | | | | ;B4 1 = |
| | | | | ;B5 |
| | | | | ;B6 |
| | | | | ;B7 |
| 002A | L002B | = | 0x002B | ;PRNDL SW MONITOR |
| | | | | ;B1 1 = SW FUNCTIONING CORRECTLY |
| 002B | L002C | = | 0x002C | ;FATI - TIMEOUT FUEL AIR RATIO |
| 002C | L002E | = | 0x002E | ;STARTUP TIMER, INCREMENT Q 100 MSEC, UNTIL 255 |
| 002E | L002F | = | 0x002F | ;FATI DECAY TERM |
| 002F | L0030 | = | 0x0030 | ;? CRANKING TIMER |
| 0030 | L0031 | = | 0x0031 | ;TIMER |
| 0031 | L0032 | = | 0x0032 | ;RUN TIME MSB |
| 0032 | L0033 | = | 0x0033 | ;RUN TIME LSB |
| 0033 | L0034 | = | 0x0034 | ;BLMS |
| | | | | ; L0034 - L0044 |
| 0034 | L0035 | = | 0x0035 | ; |
| 0035 | L003A | = | 0x003A | ; |
| 003A | L0042 | = | 0x0042 | ; |
| | L0045 | = | 0x0045 | ;BLMS WITH INACTIVE CCP |
| | | | | ; L0045 - L0055 |
| | L0055 | = | 0x0055 | ;END OF BLMS |
| 0042 | L0056 | = | 0x0056 | ;FREE RUNNING IAC STEP COUNTER |
| 0056 | L0057 | = | 0x0057 | ;IAC POSITION - ALDL |
| 0057 | L0058 | = | 0x0058 | ;A TIMER |
| 0058 | L005A | = | 0x005A | ;FILTERED ENGINE TORQUE |
| 005A | L005C | = | 0x005C | ;FILTERED ENGINE TORQUE WHEN EGR ON |
| 005C | L005E | = | 0x005E | ; |
| 005E | L0060 | = | 0x0060 | ; |
| 0060 | L0062 | = | 0x0062 | ; |
| 0062 | L0064 | = | 0x0064 | ; |
| 0064 | L0065 | = | 0x0065 | ;AN IAC VARIABLE FILTERED COMMANDED AIRFLOW |
| 0065 | L0067 | = | 0x0067 | ;AN IAC VARIABLE FILTERED COMMANDED AIRFLOW |
| 0067 | L0069 | = | 0x0069 | ;F31 VARIABLE |
| 0069 | L006A | = | 0x006A | ;F31 VARIABLE |
| 006A | L006B | = | 0x006B | ;F31 VARIABLE |
| 006B | L006C | = | 0x006C | ;F31 VARIABLE |
| 006C | L006D | = | 0x006D | ;F31 VARIABLE |
| 006D | L006E | = | 0x006E | ;F31 VARIABLE |
| 006E | L006F | = | 0x006F | ;IATMAT |
| 006F | L0070 | = | 0x0070 | ; |
| 0070 | L0071 | = | 0x0071 | ; |
| 0071 | L0072 | = | 0x0072 | ; |
| 0072 | L0076 | = | 0x0076 | ;A STARTUP/CRANKING TIMER/COUNTER, WILL ALWAYS BE Z |
| 0076 | L0077 | = | 0x0077 | ;NVWMD2 |
| | | | | ;B0 1 = IGNITION ON - FOR LOW OIL LEVEL LOGIC |
| | | | | ;B1 1 = IGNITION OFF - FOR L02 OIL LEVEL LOGIC |
| | | | | ;B2 1 = STEPPER MOTOR CRUISE PRESENT ON VEHICLE |
| | | | | ;B3 1 = ENGINE COLD - FOR LOW OIL LEVEL LOGIC |

```

;B4      1 = OIL DRAINED BACK - FOR LOW OIL LEVEL LOGIC
;B5      1 = MALF 755A (F31 SOLENOID B FAILED OFF)
;B6      1 = MALF 755B (F31 SOLENOID B FAILED ON)
;B7      1 = SES LIGHT ON AT STALL

0077      L0078 =      0x0078      ;? ACCUMULATED FUEL COUNTER
007A      L007A =      0x007A
007D      L007D =      0x007D
007E      L007E =      0x007E
0080      L0080 =      0x0080
0081      L0081 =      0x0081
0081      L0082 =      0x0082      ;
0082      L0083 =      0x0083      ;BATTERY VOLTS
0083      L0084 =      0x0084      ; SDPDW1 SERIAL DATA PACKED DISCRETE WORD 1
;0        1 = ERROR FREE TRANSMISSION ON UART LINK
;1        1 = ALDL XMIT NEEDED (RESPONSE TO A RX'D MSG)
;2        1 = CLEAR MALF CODES
;3        1 = ALDL MODE 8 - DISABLE NORMAL COMMUNICATIONS
;4        1 = DO CHECKSUM ONLY
;5        1 = ALDL TESTER IN CONTROL OF LINK
;6        1 = CLEAR NON-VOLITAL RAM (REQUESTED BY HUD)
;7        1 = MODE 4

0084      L0085 =      0x0085      ;SCI STATUS FLAG
;B0        not used
;B1        not used
;B2        not used
;B3        not used
;B4        1 = MULTI SENSOR FAILURE
;B5        not used
;B6        1 = XMIT IN WORK
;B7        1 = L024B FULL (2ND BYTE WAITING)

0085      L0086 =      0x0086      ;MW1 - MODE WORD 1 (NEARLY IDENTICAL TO $5B)
;B0        1 = RELATED TO I/O
;B1        1 = HOT OPEN LOOP
;B2        1 = SERVICE INT EXECUTION EXCEEDED 6.25 MSEC
;B3        1 = TIMING ERROR
;B4        1 = TCC ROAD SPEED 1ST PULSE FLAG
;B5        1 = A/C CLUTCH OFF
;B6        1 = M42A PASSED
;B7        1 = ENGINE RUNNING

0086      L0087 =      0x0087      ;? MISFIRE LOGIC LOOP COUNTER
0087      L0088 =      0x0088      ;BIT STATUS FLAG
;B0        1 = ALL PWM OUTPUTS COMMANDED ON
;B1
;B2        1 = RPM/12.5 < DESIRED IDLE
;B5        1 = PROM ERROR
;B6        1 = PROM ERROR 1ST PASS

0088      L0089 =      0x0089      ;? SC1SDO - I/O STATUS
;B0        FOR SELECTING COOLANT P/U RESISTOR
;B1        GETS TOGGLED
;B2        1 = ENABLE EST - PUT 5V SIGNAL TO IGNITION MODULE
;B3        1 = ENABLE TCC SOLENOID
;B4
;B5        NOT USED
;B6        1 = ENABLE XIRQ - HOW???
;B7        RELATED TO I/O CONTROL

0089      L008A =      0x008A      ;FMDBYTE1
;B0        1 = PARITY HIGH
;B1        1 = INPUT C HIGH
;B2        1 = INPUT B HIGH
;B3        1 = INPUT A HIGH
;B4        1 = LOW OIL
;B5        CRUISE STATUS FOR SMC (0 = ENGAGED)
;B6        1 = TCC/BRAKE SW HIGH
;B7        1 = A/C REQUESTED (HIGH)

```

```

;
; PRNDL DECODE TABLE
; -----
; INP P      INP A      INP B      INP C      PRNDL POSITION
; -----
; 1          1          0          0          DRIVE 4
; 1          0          1          0          DRIVE 2
; 1          0          0          1          REVERSE
; 0          1          1          0          LOW
; 0          1          0          1          NEUTRAL
; 0          0          1          1          PARK
; 0          0          0          0          DRIVE 3

008A  L008B = 0x008B ;DISCRETE INPUT STATES
008B  L008C = 0x008C ;DISCRETE INPUT STATES
;B0    1 = 2ND GEAR START REQUESTED
;B6    1 = CAM POSN SENSOR LOW
008C  L008D = 0x008D ;DISCRETE INPUT STATES
;
008D  L008E = 0x008E ;FMD INPUT STATUS WORD
;B0    1 = P/N MODE
;B1    0 = IN 2ND GEAR
;B2    0 = IN 3RD GEAR
;B3    0 = IN 4TH GEAR
;B4    0 = IN 5TH GEAR (MANUAL TRANS)
;B5    1 = LOW OIL LEVEL
;B6    1 = TCC/BRAKE SW APPLIED (CKT OPEN)
;B7    0 = A/C REQUEST
008E  L008F = 0x008F ;BIT STATUS FLAG
;B0
;B1
;B2
;B3    1 = INVALID PRNDL SW INPUTS COMBO
;B4
;B5    1 = COND MET FOR LAUNCH MODE
;B6    1 = IDLE COND MET
;B7    1 = CCP DC > ZERO AT IDLE - FOR BLM LOGIC ONLY
008F  L0090 = 0x0090 ;BIT STATUS FLAG
;B7    1 = BLM < MIN
0090  L0091 = 0x0091 ;BIT STATUS FLAG
;B0    1 = SES LIGHT ON
;B1    1 = RPM >= 6375
;B2    1 = INT RESET
;B3
;B4    0 = MAF SENSOR OKAY, MIN NUM OF
;      PULSES MET
;B5
;B6    1 = FAN #2 WAITING TO TURN ON
;B7    1 =
0091  L0092 = 0x0092 ;BIT STATUS FLAG
;B0    1 = ? TCC LOCK NOT ALLOWED
;B1
;B2    INDUCES FUEL CUT WHEN SET
;B3
;B4    SOMETHING RELATED TO MANUAL TRANNY - INDUCES FUEL
;      CUT WHEN ENABLED
;B5
;B6
;B7
0092  L0093 = 0x0093 ;CAMFLAG
;B0    NOT USED
;B1    NOT USED
;B2    NOT USED
;B3    NOT USED

```

| | | | | | |
|------|-------|---|--------|------------------|---|
| | | | | ;B4 | NOT USED |
| | | | | ;B5 | 1 = CAM PULSE SEEN DURING CRANK |
| | | | | ;B6 | 1 = CAM PULSE SEEN |
| | | | | ;B7 | 1 = CAM PULSE SEEN - USED FOR CYLCNTR |
| 0093 | L0094 | = | 0x0094 | ;QDMMW | |
| | | | | ;B0 | NOT USED |
| | | | | ;B1 | 1 = QDM FAULT1 |
| | | | | ;B2 | 1 = QDM FAULT2 |
| | | | | ;B3 | USED IN THE XIRQ ROUTINE |
| | | | | ;B4 | 1 = A/C WAS ON THIS CRANK |
| | | | | ;B5 | 1 = LOW OIL |
| | | | | ;B6 | USED IN IAC ROUTINE |
| | | | | ;B7 | 1 = A/C CLUTCH ON |
| 0094 | L0095 | = | 0x0095 | ;BIT STATUS FLAG | |
| | | | | ;B0 | 1 = STACK OVERFLOWED |
| | | | | ;B1 | 1 = HUD |
| | | | | ;B2 | 1 = STACK OVERFLOWED |
| | | | | ;B3 | 1 = |
| | | | | ;B4 | 1 = BATT VOLTS < 4.0, ELSE |
| | | | | ;B5 | |
| | | | | ;B6 | SKIP COLD ENGINE REV LIMIT WHEN SET |
| | | | | ;B7 | 1 = HUD |
| 0095 | L0096 | = | 0x0096 | ;LCCPMW | |
| | | | | ;B0 | 1 = CCP PURGE ON |
| | | | | ;B1 | 1 = FAN #1 ON |
| | | | | ;B2 | 1 = RPM HIGH IN P/N |
| | | | | ;B3 | 1 = TCC LOCKED |
| | | | | ;B4 | 1 = FAN #2 ON |
| | | | | ;B5 | 1 = PSPS CRAMP |
| | | | | ;B6 | 1 = TABLE F7MAXTRQ FOR 3 RD GEAR TORQ MGMT |
| | | | | ;B7 | 1 = NORMAL A/C REQ HAS TURNED A/C ON |
| | | | | ; | AT STARTUP |
| 0096 | L0097 | = | 0x0097 | ;BIT STATUS FLAG | |
| | | | | ;B0 | 1 = SMC ENGAGED |
| | | | | ;B1 | GETS SET IN THE XIRQ ROUTINE |
| | | | | ;B2 | 1 = |
| | | | | ;B3 | 1 = GETS TOGGLED EVERY 100 MSEC |
| | | | | ;B4 | 1 = TABLE LOOKUP IN WORK (TRANNNY) |
| | | | | ;B5 | 1 = HIGH BATT VOLTAGE - 2 ND PASS (DISABLE PWMS) |
| | | | | ;B6 | 1 = HIGH BATT VOLTAGE - 1 ST PASS |
| | | | | ;B7 | 1 = RUN FUEL ALLOWED |
| | | | | ; | (IN THE ABSENCE OF A CAM SENSOR PULSE) |
| 0097 | L0098 | = | 0x0098 | ;BIT STATUS FLAG | |
| | | | | ;B0 | |
| | | | | ;B1 | RELATED TO TORQUE MGMT |
| | | | | ;B2 | |
| | | | | ;B3 | 1 = REF PULSE OCCURED |
| | | | | ;B4 | 1 = L02B9 (TIMER) = 255 (TIME MAXED OUT) |
| | | | | ;B5 | |
| | | | | ;B6 | USED IN THE XIRQ ROUTINE |
| | | | | ;B7 | 1 = A.I.R. ENABLED |
| 0098 | L0099 | = | 0x0099 | ;BIT STATUS FLAG | |
| | | | | ;B0 | 1 = |
| | | | | ;B1 | |
| | | | | ;B2 | 1 = CLEAR FLOOD MODE |
| | | | | ;B3 | 1 = ? OVER IDLE/UNDER IDLE BIT ? |
| | | | | ;B4 | 0 = CRUISE CONROL ENABLED |
| | | | | ;B5 | ONE SECOND BIT |
| | | | | ;B6 | |
| | | | | ;B7 | |
| 0099 | L009A | = | 0x009A | ;BIT STATUS FLAG | |
| | | | | ;B0 | 1 = NEGATIVE DELTA RPM > 50 RPM |
| | | | | ;B1 | 1 = AE ENABLED |
| | | | | ;B2 | 1 = DE ENABLED |

```

;B3      1 = ? SEFI MODE - NOT USED AS A QUALIFIER
;B4      1 = ? SPARK
;B5      1 = ?
;B6      1 = ? BLMS RESET TO 128
;B7      1 = ? OVER-RPM

009A  L009B  =      0x009B      ; MWFA      FUEL AIR MODE WORD
;0       1 = TCC SLIP IS OK FOR A/C ENGAGEMNT
;1       1 = DECEL FUEL CUTOFF ENABLED
;2       BL ADDRESS CHANGE FLAG (1 = CHANGE)
;3       DELAY BLM UPDATE (1 = BLM ADDR CHNG)
;4       1 = REVERSE->DRV CHANGE
;5       PE FLAG (1 = PE IS ACTIVE)
;6       HIGH LIMIT FUEL CUTOFF ENABLED
;7       1 = INCREMENT ODOMETER

009B  L009C  =      0x009C      ; MWFA1     FUEL AIR MODE WORD 1
;0       200 MSEC. OLD P/N BIT FROM FMDINST 1 = P/N
;1       LEARN CONTROL ENABLE FLAG (1=ENABLE,0=DISABLE)
;2       1 = FATI FILTER ACTIVE
;3       1 = O/L F/A WAS RICHER THAN KCLRATIO
;4       1 = FATC FILTER ACTIVE
;5       1 = FORCE LOW PULSE RESULT OPEN LOOP
;6       RICH-LEAN FLAG (1=RICH,0=LEAN)
;7       CLOSED LOOP FLAG (1=CLOSED LOOP, 0=OPEN LOOP)

009C  L009D  =      0x009D      ;BITS STATUS FLAG
;B0      1 =
;B1      1 =
;B2      1 =
;B3      1 =
;B4      1 =
;B5      1 = USED FOR GEARS
;B6      1 = USED FOR GEARS
;B7      1 = USED FOR GEARS

009D  L009E  =      0x009E      ;TORQUE MGMT BIT STATUS FLAG
;B0
;B1
;B2
;B3 - ?
;B4
;B5 -
;B6
;B7 - INDUCES FUEL CUT WHEN SET

009E  L009F  =      0x009F      ;
009F  L00A0  =      0x00A0      ;SES LIGHT ON/OFF TIMER
00A0  L00A1  =      0x00A1      ;PRNDL SW STATUS FLAG - ALDL LIST
;B0      1 = FAULT
;B1      1 = DRIVE 1
;B2      1 = DRIVE 2
;B3      1 = DRIVE 3
;B4      1 = DRIVE 4
;B5      1 = NEUTRAL
;B6      1 = REVERSE
;B7      1 = PARK

00A1  L00A2  =      0x00A2      ;BIT STATUS FLAG
;B0

00A2  L00A3  =      0x00A3      ; TCCPMMW
;B0      1 = APPLY MODE
;B1      1 = ON MODE
;B2      1 = RELEASE MODE
;B3      1 = OFF MODE
;B4      1 = POSITIVE DELTA TPS RELEASE OF TCC
;B5      1 = TCCRAMP IS NEGATIVE
;B6      1 = TCC SLIP REQUESTED FOR A/C ENGMT
;B7      1 = ABSOLUTE SLIP HAS EXCEEDED KLOCKH

      L00A4  =      0x00A4      ;EGR SOLN STATUS FLAG

```

| | | | |
|------|---------|--------|--|
| | | | ;B1 1 = SOLN A ACTIVE |
| | | | ;B2 1 = SOLN B ACTIVE |
| | | | ;B3 1 = SOLN C ACTIVE |
| 00A3 | L00A6 = | 0x00A6 | ;CTS2 N = (DEGC + 40)*(256/192) DEFAULTED |
| 00A6 | L00A7 = | 0x00A7 | ;FILTERED CTS N = (DEGC + 40)*(256/192) |
| 00A7 | L00A9 = | 0x00A9 | ;MINOR LOOP O2 A/D COUNTS mV = 4.44 * N |
| 00A9 | L00AA = | 0x00AA | ;TPS% N = TPS% * 2.56 |
| 00AA | L00AB = | 0x00AB | ;NLRPMX |
| 00AB | L00AC = | 0x00AC | ;RPM/25 |
| 00AC | L00AD = | 0x00AD | ;RPM/12.5 |
| 00AD | L00AE = | 0x00AE | ;BLM CELL - ALDL LIST |
| 00AE | L00AF = | 0x00AF | ;FILTERED BLM (0 - 2.0) |
| 00AF | L00B1 = | 0x00B1 | ;INT UPDATE DELAY/4 VS MAF - BLM UPDATE DELAY TIME |
| 00B1 | L00B2 = | 0x00B2 | ;INT UPDATE DELAY VS MAF - from table lookup |
| 00B2 | L00B3 = | 0x00B3 | ;INTEGRATOR - ALDL LIST |
| 00B3 | L00B4 = | 0x00B4 | ;FUEL INJ OFFSET VS BATT VOLTS LOOKUP |
| 00B4 | L00B5 = | 0x00B5 | ;FAR MSB |
| 00B5 | L00B6 = | 0x00B6 | ;FAR LSB |
| | | | ;AFR = 65536/[40*(MSB*256+LSB)] |
| 00B6 | L00B7 = | 0x00B7 | ;CYL COUNTER |
| | L00B8 = | 0x00B8 | ;? DELTA PW |
| 00B7 | L00B9 = | 0x00B9 | ; |
| 00B9 | L00BA = | 0x00BA | ;LV8 - ALDL LIST |
| 00BA | L00BB = | 0x00BB | ; |
| 00BB | L00BC = | 0x00BC | ; |
| 00BC | L00BD = | 0x00BD | ;FILTERED MPH |
| 00BD | L00BF = | 0x00BF | ;MPH*3 |
| 00BF | L00C0 = | 0x00C0 | ;MINOR LOOP REF PERIOD - ALDL LIST |
| | | | ;N = MSEC * 65.536 |
| | | | ;N = 1310720/RPM |
| 00C0 | L00C2 = | 0x00C2 | ;PREV REF PERIOD (3x - spark) |
| 00C2 | L00C4 = | 0x00C4 | ;A REF PERIOD - FROM 3FC4 (OR 3FEC) |
| 00C4 | L00C6 = | 0x00C6 | ;PREV REF PERIOD |
| 00C6 | L00C8 = | 0x00C8 | ;3/4 of 3x REF PERIOD |
| 00C8 | L00CA = | 0x00CA | ;FINDS ITS WAY INTO 0x3FE2 |
| 00CA | L00CB = | 0x00CB | ;STORED FROM L3FEC |
| 00CB | L00CD = | 0x00CD | ;MAF (GMS/SEC = N/256) |
| 00CD | L00CF = | 0x00CF | ;MAF (GMS/SEC) |
| 00CF | L00D1 = | 0x00D1 | ;CORRECTED MAF, FOR IAT AND RPM |
| | L00D3 = | 0x00D3 | ;MAF PULSE COUNTER |
| | L00D5 = | 0x00D5 | ;MAF |
| | L00D7 = | 0x00D7 | ;MAF PULSE ACCUMULATOR |
| 00D1 | L00D9 = | 0x00D9 | ;PIDMW2 - IDLE AIR CONTROL MODE WORD |
| | | | ;0 1 = A/C SLUGGING TEMPERATURES MET |
| | | | ;1 1 = MOTOR HAS BEEN AT 0 DURING RESET |
| | | | ;2 1 = MOTOR RESET IN PROGRESS |
| | | | ;3 1 = AIRFLOW > COMMAND AIR FLOW |
| | | | ;4 1 = RESET COMPLETE |
| | | | ;5 1 = DON'T ADD HOT STRT IDLE SPD |
| | | | ;6 1 = STEPPER MOTOR CRUISE ENGAGED |
| | | | ;7 1 = ASYNC A.E. DISABLED |
| 00D9 | L00DA = | 0x00DA | ;BIT STATUS FLAG, RELATED TO FUEL OUTPUT |
| | | | ;B1 |
| | | | ;B2 |
| | | | ;B3 - RELATED TO IDLE FUEL |
| | | | ;B4 - RELATED TO IDLE FUEL |
| | | | ;B7 |
| 00DA | L00DB = | 0x00DB | ;IAC BIT STATUS FLAG |
| | | | ;B0 |
| | | | ;B1 |
| | | | ;B2 |
| | | | ;B3 |
| | | | ;B4 |
| | | | ;B5 |

```

;B6
;B7 1 = A/C CLUTCH ON

00DB L00DC = 0x00DC ;
00DC L00DD = 0x00DD ;OVER OR UNDER IDLE RPM/12.5 (SIGNED)
00DD L00DE = 0x00DE ;DESIRED IDLE RPM - ALDL LIST
00DE L00DF = 0x00DF ;THROTTLE FOLLOWER IAC STEPS COUNTER
00DF L00E0 = 0x00E0
00E0 L00E1 = 0x00E1 ;COUNTER - FOR SCHEDULING 384 VS 4K CTS P/U
00E1 L00E2 = 0x00E2 ;BIT STATUS FLAG RELATED TO FINJ OUTPUTS

;B0 1 =
;B1 1 =
;B2 1 = 3F86 OUTPUT ZERO'D (INJ #4)
;B3 1 = 3F84 OUTPUT ZERO'D (INJ #3)
;B4 1 = 3F82 OUTPUT ZERO'D (INJ #2)
;B5 1 = 3F80 OUTPUT ZERO'D (INJ #1)
;B6 1 = WR 0000 TO I/O PORT (INJ #6)
;B7 1 = 3FD0 OUTPUT ZERO'D (INJ #5)

00E2 L00E3 = 0x00E3 ;A REF PERIOD VARIABLE
00E3 L00E4 = 0x00E4 ;FINAL BPW - ALDL LIST
      L00E6 = 0x00E6 ;MAF/2, CORRECTED FOR RPM (REF PERIOD)
      L00E7 = 0x00E7 ;
00E4 L00E9 = 0x00E9 ;F31 COMMANDED GEAR - ALDL LIST
00E9 L00EA = 0x00EA ;TCC PWM DUTY CYCLE - ALDL LIST
00EA L00EB = 0x00EB ;TRANNNY RPM (FRACTION OF 20/REF PER)
      ;N = RPM
00EB L00ED = 0x00ED ;1 SECOND TIMER MSB
00ED L00EE = 0x00EE ;1 SECOND TIMER LSB
00EE L00EF = 0x00EF ;1 SEC CNTR - INC IN 1 SECOND INTERVALS
00EF L00F0 = 0x00F0 ;IATMAT - DEG C
00F0 L00F1 = 0x00F1 ;TRANNNY MPH
00F1 L00F2 = 0x00F2 ;STARTUP COOLANT
00F2 L00F3 = 0x00F3 ;KNOCK RETARD DEGREES
00F3 L00F4 = 0x00F4 ;CCP DUTY CYCLE - ALDL LIST
00F4 L00F5 = 0x00F5 ;%TPS - SAME AS L00AA
00F5 L00F6 = 0x00F6 ;TPS A/D COUNTS (V = 5 * N / 255)
00F6 L00F7 = 0x00F7 ;COMMANDED AIRFLOW (GMS/SEC = N/10)
00F7 L00F8 = 0x00F8 ;TCC PWM VARIABLE
00F8 L00F9 = 0x00F9 ;A/C PRESSURE - PSI = 1.92n - 26.88
      L00FA = 0x00FA ;F31/TCC PWM VARIABLE (LOOKUP RESULT)
00F9 L0100 = 0x0100 ;ACCEL ENRICHMENT TIMER, GETS INCREMENTED IN XIRQ
0100 L0101 = 0x0101 ;DECEL ENLEANMENT TIMER, GETS INCREMENTED IN XIRQ
0101 L0102 = 0x0102 ;FUEL TERM, CORRECTED FOR BLM, INTEGRATOR AND FAR
0102 L0104 = 0x0104 ;BPW MSB
0104 L0105 = 0x0105 ;BPW LSB
0105 L0106 = 0x0106 ;ASYNCR PW
      L0108 = 0x0108 ;MAF (GM/SEC) MSB - ALDL
      L0109 = 0x0109 ;MAF (GM/SEC) LSB
      ; gm/sec = MSB + (LSB * 0.00396)
0106 L010A = 0x010A ;CLNT TEMP DEG C = .75N - 40
010A L010B = 0x010B ;PREV O2 A/D COUNTS
010B L010C = 0x010C ;FILTERED O2 A/D COUNTS
010C L010E = 0x010E ;ALSO FILTERED O2 A/D COUNTS
010E L0110 = 0x0110 ;%TPS > 1.95%
0110 L0111 = 0x0111 ;FILTERED TPS A/D COUNTS
0111 L0112 = 0x0112 ;
0112 L0113 = 0x0113 ;O2 SENSOR READY TIMER
0113 L0114 = 0x0114 ;INTEGRATOR UPDATE DELAY TIMER
0114 L0115 = 0x0115 ;INTEGRATOR TIMER
0115 L0116 = 0x0116 ;FATC - COOLANT FUEL AIR RATIO
0116 L0118 = 0x0118 ;POWER ENRICHMENT FAR
0118 L0119 = 0x0119 ;
0119 L011A = 0x011A ;
011A L011C = 0x011C ;
011C L011E = 0x011E ;DYNAMIC DWELL

```

```

011E L0120 = 0x0120 ;TOTAL DWELL
0120 L0122 = 0x0122 ;
0122 L0123 = 0x0123 ;SPARK FROM TABLE LOOKUP, WITH L0122
0123 L0124 = 0x0124 ;IAC VARIABLE
0124 L0126 = 0x0126 ;? OLD PA2 COUNTER
0126 L0127 = 0x0127
0127 L0128 = 0x0128
0128 L0129 = 0x0129 ;RESULT OF SCALED MAF TABLE LOOKUP
0129 L012A = 0x012A ;TRUNCATED CTS FOR TABLE LOOKUPS
012A L012B = 0x012B ;FATI DECAY DELAY TIME VS RPM, ADJUSTED TO CLNT TEMP
012B L012C = 0x012C ;PREV RPM/12.5
012C L012D = 0x012D ;O2 A/D MALF TIMER
012D L012E = 0x012E ;CTS MALF TIMER
012E L012F = 0x012F ;CTS MALF TIMER
012F L0130 = 0x0130 ;BATT/ALT MALF TIMER
0130 L0131 = 0x0131 ;24x CRANK SENSOR PULSE COUNTER
0131 L0132 = 0x0132 ;TIMER FOR MALF 341 (INT CAM SIGNAL)
0132 L0133 = 0x0133
0133 L0134 = 0x0134 ;TPS A/D MALF TIMER
0134 L0135 = 0x0135
0135 L0136 = 0x0136 ;DTC P0502 - VSS MALF TIMER
0136 L0137 = 0x0137
0137 L0138 = 0x0138 ;TRANSAXLE P/N -> IN GEAR TRANSITION TIMER
0138 L0139 = 0x0139
0139 L013A = 0x013A
013A L013B = 0x013B
013B L0140 = 0x0140
L0141 = 0x0141
0140 L0143 = 0x0143 ;KNOCK SENSOR MALF TIMER
0143 L0144 = 0x0144
0144 L0145 = 0x0145
0145 L0146 = 0x0146 ;TRANS TEMP MALF TIMER
0146 L0147 = 0x0147 ;TRANS A/D MALF TIMER
0147 L0148 = 0x0148
0148 L014C = 0x014C ;FILTERED CCP DC
014C L014E = 0x014E
014E L014F = 0x014F
014F L0150 = 0x0150
0150 L0151 = 0x0151 ;PE TIMER
0151 L0152 = 0x0152 ;CRANKING PW MULTIPLIER VS %TPS
0152 L0153 = 0x0153 ;FILTERED TPS%
0153 L0155 = 0x0155 ;FILTERED TPS%
0155 L0157 = 0x0157 ;FILTERED TPS%
0157 L0159 = 0x0159 ;ACCEL ENRICHMENT FUEL TERM
0159 L015A = 0x015A ;PREV D-TPS, FOR DE MULTIPLIER LOOKUP
015A L015B = 0x015B ;PREV D-TPS, FOR ASYNC ROUTINE
015B L015C = 0x015C ;AE MULTIPLIER VS CTS, FROM LOOKUP
015C L015D = 0x015D ;AE MULTIPLIER VS IAT, FROM LOOKUP
L015E = 0x015E ;O2 XCOUNTS
015D L015F = 0x015F ;OVER/UNDER IDLE RPM/12.5 (UNSIGNED)
015F L0160 = 0x0160 ;IDLE RPM ERROR (N = RPM)
0160 L0161 = 0x0161 ;IAC VARIABLE
0161 L0163 = 0x0163 ;LOW FLOW MAF ( = 10x MAF, GMS/SEC)
0163 L0164 = 0x0164 ;AN IAC VARIABLE
0164 L0165 = 0x0165 ;IAC VARIABLE MSB
0165 L0166 = 0x0166 ;IAC VARIABLE LSB
0166 L0167 = 0x0167 ;IAC VARIABLE
0167 L0168 = 0x0168 ;IAC VARIABLE
0168 L0169 = 0x0169 ;DIFF BETWEEN A/F AND COMMANDED A/F
0169 L016B = 0x016B ;MAIN SPARK ADVANCE LOOKUP RESULT
016B L016C = 0x016C ;SPARK CORRECTION VS CTS AND LV8 TERM
;IS ADDED TO SPK ADV
016C L016D = 0x016D ;SPARK CORRECTION FOR EGR
;IS ADDED TO SPK ADV

```

| | | | | |
|------|-------|---|--------|---|
| 016D | L016E | = | 0x016E | ;SPARK CORRECTION VS IAT AND LV8 TERM ;IS ADDED TO SPK ADV |
| 016E | L016F | = | 0x016F | ;SPARK VARIABLE VS LV8 AND RUN TIME LSB =64 ;IS ADDED TO SPK ADV |
| 016F | L0170 | = | 0x0170 | ;TCC LOCKED SPARK CORRECTION TERM |
| 0170 | L0171 | = | 0x0171 | ;LAUNCH MODE SPARK CORRECTION |
| 0171 | L0173 | = | 0x0173 | ; |
| 0173 | L0174 | = | 0x0174 | ;IS ADDED TO SPK ADV |
| 0174 | L0175 | = | 0x0175 | ;IAC STEPS, FROM TABLE LOOKUP |
| 0175 | L0176 | = | 0x0176 | ; |
| 0176 | L0178 | = | 0x0178 | ;POWER ENRICHMENT SPARK CORRECTION ;IS ADDED TO SPK ADV |
| 0178 | L0179 | = | 0x0179 | ;DEPENDENT ON FILTERED MPH, IS ADDED TO SPK RET |
| 0179 | L017A | = | 0x017A | ;ASYNC TIMER |
| 017A | L017B | = | 0x017B | ;L88FC LOOKUP RESULT - FATI DECAY DELAY VS CTS |
| 017B | L017C | = | 0x017C | ;UNFILTERED MAF FOR IAC (= 10x MAF, GMS/SEC) |
| 017C | L017D | = | 0x017D | ; |
| 017D | L017E | = | 0x017E | ;CUMULATIVE KNOCK RETARD DEGREES, IS ADDED TO SPK RET |
| 017E | L017F | = | 0x017F | |
| 017F | L0180 | = | 0x0180 | |
| | L0183 | = | 0x0183 | ;BAD CYL ID - ALDL LIST |
| 0180 | L0184 | = | 0x0184 | ;PREV BAD CYL ID |
| | L0185 | = | 0x0185 | ;8F36 TABLE LOOKUP RESULT - EGR |
| 0184 | L0186 | = | 0x0186 | ;8F3F TABLE LOOKUP RESULT - EGR |
| 0186 | L0187 | = | 0x0187 | ;IAT CORRECTION TERM TO BASE MAF |
| 0187 | L0188 | = | 0x0188 | ;IAT CORRECTION TERM TO BASE MAF |
| 0188 | L0189 | = | 0x0189 | |
| 0189 | L018A | = | 0x018A | ;RELATED TO AFR |
| 018A | L018B | = | 0x018B | ;RELATED TO AFR |
| 018B | L018C | = | 0x018C | ;RELATED TO AFR |
| 018C | L018D | = | 0x018D | ;IATMAT |
| 018D | L018E | = | 0x018E | ;FILTERED IATMAT |
| 018E | L0190 | = | 0x0190 | |
| 0190 | L0191 | = | 0x0191 | |
| 0191 | L0192 | = | 0x0192 | |
| 0192 | L0193 | = | 0x0193 | ; |
| 0193 | L0194 | = | 0x0194 | |
| 0194 | L0195 | = | 0x0195 | ;IAT A/D COUNTS |
| 0195 | L0198 | = | 0x0198 | ;TCC PWM VARIABLE |
| 0198 | L0199 | = | 0x0199 | ;TCC PWM VARIABLE |
| 0199 | L019C | = | 0x019C | ;PREV RUN TIME (FOR TCC PWM) |
| 019C | L019E | = | 0x019E | ;TCC PWM VARIABLE |
| 019E | L01A0 | = | 0x01A0 | ;CRANKING TIMER |
| 01A0 | L01A2 | = | 0x01A2 | |
| 01A2 | L01A6 | = | 0x01A6 | ;LOW REF PERIOD COUNTER |
| 01A6 | L01A7 | = | 0x01A7 | ;PW, UNCORRECTED FOR VOLTS OR INJ PW |
| | L01A9 | = | 0x01A9 | ;ALDLMW2 |
| | | | | ;B0 1 = VATS SHUT OFF FUEL |
| | | | | ;B1 1 = SMC ENGAGED |
| | | | | ;B2 1 = SMC INHIBITED |
| | | | | ;B3 1 = ENGINE RUNNING |
| | | | | ;B4 1 = LOW OIL LIGHT ON |
| | | | | ;B5 1 = A.I.R. ENABLED |
| | | | | ;B6 1 = RICH (0=LEAN) |
| | | | | ;B7 1 = CLOSED LOOP |
| 01A7 | L01AA | = | 0x01AA | ;? PREV BLM CELL |
| 01AA | L01AB | = | 0x01AB | ; |
| 01AB | L01AC | = | 0x01AC | ; |
| 01AC | L01AE | = | 0x01AE | ;PREV EGR DUTY CYCLE |
| 01AE | L01AF | = | 0x01AF | ;EGR DC/SOLN COMBINATION |
| 01AF | L01B0 | = | 0x01B0 | ;EGR DUTY CYCLE DELTA |
| | L01B1 | = | 0x01B1 | ;? EGR TIMER OF SOME KIND |
| | L01B2 | = | 0x01B2 | ;PREV NLRPMX |
| | L01B3 | = | 0x01B3 | ;TIMER - EGR |

```

        L01B4 = 0x01B4 ;TIMER - EGR
        L01B5 = 0x01B5 ;TIMER - EGR
01B0    L01B6 = 0x01B6 ;8F70 TABLE LOOKUP RESULT - EGR
01B6    L01B7 = 0x01B7 ;BASE EGR SPARK ADVANCE
01B7    L01B8 = 0x01B8 ;
01B8    L01B9 = 0x01B9 ;
01B9    L01BA = 0x01BA ;CRANKING COUNTER
01BA    L01BC = 0x01BC ;LOW REF PERIOD COUNTER
        L01BD = 0x01BD ;PREV TRANNY GEAR BYTE
01BC    L01BE = 0x01BE ;PREV %TPS (FOR TCC PWM)
        L01BF = 0x01BF ;F31 VARIABLE
        L01C0 = 0x01C0 ;F31 VARIABLE
01BE    L01C1 = 0x01C1 ;
01C1    L01C2 = 0x01C2 ;
01C2    L01C3 = 0x01C3 ;PRNDL STATUS CHECK LOOP COUNTER
01C3    L01C4 = 0x01C4
01C4    L01C5 = 0x01C5
01C5    L01C6 = 0x01C6 ;F31/TCC PWM VARIABLE
01C6    L01C7 = 0x01C7 ;FILTERED TCC SLIP RPM, RPM = 2 * N - 255
01C7    L01C8 = 0x01C8
01C8    L01C9 = 0x01C9 ;F31 VARIABLE
01C9    L01CA = 0x01CA ;F31/TCC PWM VARIABLE (TIMER)
01CA    L01CB = 0x01CB ;TCC PWM VARIABLE
01CB    L01CC = 0x01CC ;TCC PWM VARIABLE
01CC    L01CD = 0x01CD ;F31/TCC PWM VARIABLE
01CD    L01CE = 0x01CE ;TCC PWM VARIABLE
01CE    L01CF = 0x01CF ;TCC PWM VARIABLE
        L01D0 = 0x01D0 ;RATIO OF FILTERED ENGINE TORQUE (L005A/2)/L8698
01CF    L01D2 = 0x01D2 ;TCC PWM VARIABLE
01D2    L01D3 = 0x01D3 ;TCC PWM VARIABLE
01D3    L01D4 = 0x01D4 ;TRANS TEMP A/D COUNTS
01D4    L01D5 = 0x01D5 ;TRANS TEMP (IN CASE OF MALF, = CTS)
01D5    L01D6 = 0x01D6 ;TRANS TEMP - ALDL
01D6    L01D7 = 0x01D7 ;FILTERED TRANS TEMP
01D7    L01D8 = 0x01D8
01D8    L01DA = 0x01DA
01DA    L01DD = 0x01DD
01DD    L01DE = 0x01DE
01DE    L01DF = 0x01DF ;ADDED TO SPARK ADVANCE IF BIT 1 L0092 CLR
                                ;OTHERWISE IS ADDED TO SPK RET - WILL ALWAYS BE CLEAR

01DF    L01E0 = 0x01E0
01E0    L01E1 = 0x01E1
01E1    L01E2 = 0x01E2
        L01E4 = 0x01E4 ;F31 VARIABLE
        L01E5 = 0x01E5 ;TABLE LOOKUP RESULT
        L01E6 = 0x01E6 ;F31 VARIABLE (TABLE LOOKUP RESULT)
01E2    L01E7 = 0x01E7 ;OIL LEVEL WARNING LIGHT DELAY TIMER
01E7    L01EC = 0x01EC
01EC    L01F1 = 0x01F1
01F1    L01F3 = 0x01F3
01F3    L01F4 = 0x01F4
01F4    L01FE = 0x01FE
01FE    L0200 = 0x0200 ;SCI DATA BYTE COUNTER
0200    L0201 = 0x0201 ;RUNNING CKSUM ON SCI RX DATA
0201    L0202 = 0x0202 ;SCI DATA MSG TBL INDEX
0202    L0204 = 0x0204 ;SERIAL DATA MODE WD
0204    L0205 = 0x0205 ;
        L0206 = 0x0206 ;
        L0226 = 0x0226 ;MESSAGE BUFFER
        L0228 = 0x0228 ;
0205    L022B = 0x022B ;INPUT MESSAGE BUFFER
022B    L022C = 0x022C ;MODE 4 CONTROL BYTE (ICB + 1)
                                ;B0
                                ;B1

```

```

;B2    ELITE
;B3    A.I.R. SOLN
;B4    SSB
;B5    SSA
;B6    CC INHIBIT SIGNAL
;B7    ?LOW OIL LIGHT
022C    L022D  =    0x022D    ;MODE 4 CONTROL BYTE(ICB + 2)
;B0
;B1
;B2    ELITE
;B3    A.I.R. SOLN
;B4    SSB
;B5    SSA
;B6    CC INHIBIT SIGNAL
;B7    ?LOW OIL LIGHT
022D    L022E  =    0x022E    ;PWM OUTPUT CONTROL BYTE FOR MODE 4 (ICB + 3)
;B0    SES LIGHT
;B1    FAN #2 OUTPUT
;B2    FAN #1 OUTPUT
;B3    EGR PWM OUTPUT
;B4    EGR PWM OUTPUT
;B5    EGR PWM OUTPUT
;B6    TCC PWM OUTPUT
;B7    A/C PWM OUTPUT
022E    L022F  =    0x022F    ;CONTROL BYTE FOR MODE 4 (ICB + 4)
;B0    SES LIGHT
;B1    FAN #2 OUTPUT
;B2    FAN #1 OUTPUT
;B3    EGR PWM OUTPUT
;B4    EGR PWM OUTPUT
;B5    EGR PWM OUTPUT
;B6    TCC PWM OUTPUT
;B7    A/C PWM OUTPUT
022F    L0230  =    0x0230    ;CONTROL BYTE FOR MODE 4 (ICB + 5)
;B0    1 = FORCE O2 SENSOR READY
;B1    1 =
;B2    1 = ALL OUTPUTS COMMANDED ON
;B3    1 = DISABLE EST
;      ie DON'T PUT 5V TO BYPASS LINE
;B4    1 = RESET BLMS
;B5    1 = SET IAC TO MAX
;B6    1 = CLEAR MALF FLAGS
;B7    1 = RELATED TO VATS
0230    L0231  =    0x0231    ;STATUS FLAG (ICB + 6)
;B0    1 = FORCE CLOSED LOOP
;B5    1 = SKIP CTS MALF, USE DEFAULT A/D COUNTS
;B6    1 = SKIP IAT MALF, USE DEFAULT A/D COUNTS
;B7    1 = FORCE "MAF SIGNAL LOW"
0231    L0232  =    0x0232    ;CONTROL BYTE FOR MODE 4 (ICB + 7)
;B1    1 = COMMAND EGR
;B2    1 = COMMAND TCC PWM
;B3    1 = COMMAND CCP
0232    L0233  =    0x0233    ;MODE 4 COMMANDED DUTY CYCLE FOR CCP, TCC
; AND EGR (ICB + 8)
0233    L0234  =    0x0234    ;MODE 4 CONTROL BYTE (ICB + 9)
;B0    1 = IDLE SPEED COMMANDED
;B1    1 = COMMANDED IDLE SPEED IS IN L0235
;B2    1 = AFR COMMANDED
;B3    1 = SPARK COMMANDED
;B4    1 = SPARK ADVANCE
;B5    1 = SPARK RETARD
;B6
;B7
0234    L0235  =    0x0235    ;MODE 4 COMMANDED IDLE (ICB + 10)

```

| | | | | |
|------|-------|---|--------|--|
| 0235 | L0236 | = | 0x0236 | ;MODE 4 COMMANDED AFR (ICB + 11) |
| | | | | ;CAN BE ENTERED AS AFR*10 (147 FOR STOICH) |
| 0236 | L0237 | = | 0x0237 | ;MODE 4 COMMANDED SPARK ADVANCE (ICB + 12) |
| 0237 | L0239 | = | 0x0239 | ;MODE 4 COMMANDED ?? -> L00E2 (ICB + 13) |
| | | | | ;? INDIVIDUAL INJECTOR ON/OFF ??? |
| | | | | ;OR ASYNC ? |
| | | | | ;IF = 0xA8, ? ASYNC OUTPUT SKIPPED |
| | | | | ;B2 |
| | | | | ;B3 |
| | | | | ;B4 |
| | | | | ;B5 |
| | | | | ;B6 |
| | | | | ;B7 |
| | L023A | = | 0x023A | ;OUTPUT MESSAGE BUFFER |
| 0239 | L0248 | = | 0x0248 | ;? DEVICE ID CODE |
| 0248 | L0249 | = | 0x0249 | ;SERIAL INPUT DATA MSG LENGTH |
| 0249 | L024A | = | 0x024A | ;? SCI MSG CKSUM |
| 024A | L024B | = | 0x024B | ;SECOND BYTE TO TX IS STORED HERE |
| 024B | L024C | = | 0x024C | ;MODE 4 TIMER |
| 024C | L024E | = | 0x024E | ;MODE 4 TIMER |
| 024E | L0250 | = | 0x0250 | ;ACCUMULATED FUEL |
| 0250 | L0252 | = | 0x0252 | ;? ACCUMULATED DISTANCE ? |
| 0252 | L0254 | = | 0x0254 | ;IAT INVERSE A/D COUNTS $V = 5(256 - N)/256$ |
| 0254 | L0255 | = | 0x0255 | ;CTS A/D COUNTS |
| 0255 | L0256 | = | 0x0256 | ;DEFAULT MAF |
| 0256 | L0258 | = | 0x0258 | ;MAF FROM RPM AND IAT, USED FOR TORQUE CALC |
| | L025A | = | 0x025A | ;MAF, FOR TORQUE CALCULATION |
| 0258 | L025C | = | 0x025C | ;MAF CORRECTION TERM, RPM AND IAT |
| 025C | L025E | = | 0x025E | ; |
| 025E | L025F | = | 0x025F | ;PREV RPM (20/REF PER) - FOR A/C LOGIC |
| 025F | L0261 | = | 0x0261 | ;TURBINE RPM |
| 0261 | L0263 | = | 0x0263 | ;MPH - ALDL LIST |
| 0263 | L0265 | = | 0x0265 | ;RATIO OF TURBINE TO ENGINE RPM |
| 0265 | L0267 | = | 0x0267 | ; |
| 0267 | L0269 | = | 0x0269 | ;A VSS VARIABLE |
| 0269 | L026A | = | 0x026A | ; |
| 026A | L026B | = | 0x026B | |
| 026B | L026C | = | 0x026C | ; |
| 026C | L026D | = | 0x026D | ;REF PERIOD TIMER - COUNTS THE TIME BETWEEN REF PULSES |
| 026D | L026E | = | 0x026E | ;PASSKEY II MALF TIMER |
| 026E | L026F | = | 0x026F | |
| 026F | L0270 | = | 0x0270 | ; |
| 0270 | L0272 | = | 0x0272 | ;? ACCUMULATED FUEL |
| 0272 | L0273 | = | 0x0273 | ; |
| 0273 | L0275 | = | 0x0275 | |
| 0275 | L0276 | = | 0x0276 | ;ENGINE TORQUE (FT-LBS = 2N) |
| 0276 | L0278 | = | 0x0278 | ;TURBINE TORQUE (FT-LBS = 2N) |
| 0278 | L027A | = | 0x027A | ;? 2'S COMPLEMENT MAX SPARK |
| 027A | L027B | = | 0x027B | ;RATIO OF TURBINE TORQUE TO TORQUE THRESHOLD |
| | | | | ;AN AXIS FOR L85A8 TABLE (SPARK) |
| 027B | L027C | = | 0x027C | ;CRANKING PW |
| 027C | L027E | = | 0x027E | ; |
| 027E | L027F | = | 0x027F | ;CRANK TO RUN PW MULTIPLIER; STARTS AT FF, AND GETS |
| | | | | ; DECREMENTED BY L86CF AMOUNT |
| 027F | L0280 | = | 0x0280 | ;TIMER/COUNTER FOR TORQUE MGMT |
| 0280 | L0281 | = | 0x0281 | ;PREV 16 BIT RPM |
| 0281 | L0283 | = | 0x0283 | ;N/V RATIO MSB |
| | L0284 | = | 0x0284 | ;N/V RATIO LSB |
| 0283 | L0285 | = | 0x0285 | ;LOOKUP AXIS VARIABLE FOR L85A8 3D TABLE |
| 0285 | L0286 | = | 0x0286 | ;L85A8 LOOKUP RESULT |
| 0286 | L0287 | = | 0x0287 | ; |
| 0287 | L0288 | = | 0x0288 | ;2'S COMP MAX SPARK WITH L027A |
| 0288 | L0289 | = | 0x0289 | ;TORQUE MGMT SPK RET |
| 0289 | L028D | = | 0x028D | ;PREV RPM (20/REF PER) |

| | | | | |
|------|-------|---|--------|---|
| 028D | L028F | = | 0x028F | ;? TIMER/COUNTER |
| 028F | L0290 | = | 0x0290 | ;SA TOTAL UNLIMITED REL TO TDC MSB - ALDL |
| | L0291 | = | 0x0291 | ;SA TOTAL UNLIMITED REL TO TDC LSB - ALDL |
| | | | | ; DEG = 90 * N / 256 (N IS SIGNED) |
| 0290 | L0292 | = | 0x0292 | ;FAN #1 TURN ON DELAY TIMER |
| 0292 | L0294 | = | 0x0294 | ;A/C CLUTCH OFF TIMER |
| 0294 | L0296 | = | 0x0296 | ;LIMITED NLRPMX (FOR L85A8 TABLE LOOKUP ONLY) |
| 0296 | L0297 | = | 0x0297 | ;KNOCK |
| 0297 | L0298 | = | 0x0298 | ;PREV EST VOLTAGE |
| 0298 | L029A | = | 0x029A | ;PREV EST VOLTAGE |
| 029A | L029C | = | 0x029C | ;KNOCK COUNTS |
| 029C | L029E | = | 0x029E | ;O2 XCOUNTS |
| 029E | L029F | = | 0x029F | |
| 029F | L02A1 | = | 0x02A1 | ;DE TERM |
| 02A1 | L02A2 | = | 0x02A2 | ;DE TERM VS DE TIME LOOKUP RESULT |
| 02A2 | L02A3 | = | 0x02A3 | |
| 02A3 | L02A5 | = | 0x02A5 | ;BPW FINE CORRECTION |
| 02A5 | L02A6 | = | 0x02A6 | ;ALDL FLAGWORD |
| | | | | ;B0 1 = |
| | | | | ;B1 1 = P/N MODE |
| | | | | ;B2 1 = A/C CLUTCH ON |
| | | | | ;B3 NOT USED |
| | | | | ;B4 NOT USED |
| | | | | ;B5 1 = ALDL TESTER IN CONTROL OF LINK |
| | | | | ;B6 1 = SES LIGHT ON |
| | | | | ;B7 1 = IAC MOTOR RESET TO ZERO |
| 02A6 | L02A7 | = | 0x02A7 | ;ALDL FLAGWORD |
| | | | | ;B0 1 = PASSKEY II ERROR |
| | | | | ;B1 1 = VATS FAILED |
| | | | | ;B2 NOT USED |
| | | | | ;B3 1 = "NO VSS SIGNAL" MALF |
| | | | | ;B4 NOT USED |
| | | | | ;B5 1 = A/C REQUESTED |
| | | | | ;B6 1 = CTS MALFS OCCURED |
| | | | | ;B7 NOT USED |
| 02A7 | L02A8 | = | 0x02A8 | ;ALDL FLAGWORD |
| | | | | ;B0 1 = |
| | | | | ;B1 1 = P/N MODE |
| | | | | ;B2 1 = A/C CLUTCH ON |
| | | | | ;B3 1 = "NO VSS SIGNAL" MALF |
| | | | | ;B4 NOT USED |
| | | | | ;B5 1 = ALDL TESTER IN CONTROL OF LINK |
| | | | | ;B6 1 = SES LIGHT ON |
| | | | | ;B7 1 = BRAKE APPLIED |
| | L02A9 | = | 0x02A9 | ;ALDLMW1 |
| | | | | ;B0 1 = 2 ND GEAR STRT REQ LOW (SNOW SHFT) |
| | | | | ;B1 1 = DEGR SOLN A ON |
| | | | | ;B2 1 = DEGR SOLN B ON |
| | | | | ;B3 1 = DEGR SOLN C ON |
| | | | | ;B4 1 = SHIF LIGHT IS ON |
| | | | | ;B5 1 = PSPS CRAMP PRESENT |
| | | | | ;B6 NOT USED |
| | | | | ;B7 1 = A/C REQUEST ON (NOT USED) |
| 02A8 | L02AA | = | 0x02AA | |
| 02AA | L02AB | = | 0x02AB | |
| 02AB | L02AC | = | 0x02AC | ;ANTICIPATED A/C LOAD STEPS, FROM TABLE |
| 02AC | L02AD | = | 0x02AD | |
| 02AD | L02AE | = | 0x02AE | |
| 02AE | L02AF | = | 0x02AF | |
| 02AF | L02B0 | = | 0x02B0 | ;A/C CLUTCH ON TIMER |
| 02B0 | L02B1 | = | 0x02B1 | ;TIMER |
| 02B1 | L02B2 | = | 0x02B2 | ;TIMER |
| 02B2 | L02B3 | = | 0x02B3 | |
| 02B3 | L02B4 | = | 0x02B4 | |

| | | | | |
|------|-------|---|--------|---|
| 02B4 | L02B5 | = | 0x02B5 | |
| 02B5 | L02B6 | = | 0x02B6 | ;?FAN #1 IAC STEPS |
| 02B6 | L02B7 | = | 0x02B7 | ;?FAN #2 IAC STEPS |
| 02B7 | L02B8 | = | 0x02B8 | ; |
| 02B8 | L02B9 | = | 0x02B9 | ; |
| 02B9 | L02BA | = | 0x02BA | ;FUEL ENRICHMENT FOR SHIFT INTO GEAR |
| 02BA | L02BB | = | 0x02BB | ;TIMER |
| 02BB | L02BC | = | 0x02BC | ;BPW VARIABLE |
| 02BC | L02BD | = | 0x02BD | ;bpw |
| 02BD | L02BF | = | 0x02BF | ;TIMER |
| 02BF | L02C0 | = | 0x02C0 | ;TIMER |
| 02C0 | L02C1 | = | 0x02C1 | ;TIMER |
| 02C1 | L02C2 | = | 0x02C2 | ;TIMER |
| 02C2 | L02C3 | = | 0x02C3 | ;TIMER |
| 02C3 | L02C4 | = | 0x02C4 | ;TIMER |
| 02C4 | L02C5 | = | 0x02C5 | ;TIMER |
| 02C5 | L02C6 | = | 0x02C6 | ;PREV 16 BIT RPM |
| 02C6 | L02C8 | = | 0x02C8 | ;PREV 16 BIT RPM |
| 02C8 | L02CA | = | 0x02CA | ; |
| 02CA | L02CB | = | 0x02CB | ;IAC VARIABLE - TIMER/COUNTER |
| 02CB | L02CC | = | 0x02CC | ;? PREV ASYNC PW |
| 02CC | L02CE | = | 0x02CE | ;A DELTA REF PERIOD |
| 02CE | L02CF | = | 0x02CF | |
| 02CF | L02D0 | = | 0x02D0 | ;A DELTA REF PERIOD |
| 02D0 | L02D2 | = | 0x02D2 | ;A DELTA REF PERIOD |
| 02D2 | L02D4 | = | 0x02D4 | ;A DELTA REF PERIOD |
| | L02D6 | = | 0x02D6 | ;A DELTA REF PERIOD |
| 02D4 | L02D8 | = | 0x02D8 | ;A DELTA REF PERIOD |
| | L02DA | = | 0x02DA | ;A DELTA REF PERIOD |
| | L02DC | = | 0x02DC | ;A DELTA REF PERIOD |
| 02D8 | L02DE | = | 0x02DE | |
| 02DE | L02E0 | = | 0x02E0 | ;POSITIVE IDLE RPM ERROR SPARK TERM |
| | | | | ;IS ADDED TO SPK RET |
| 02E0 | L02E1 | = | 0x02E1 | |
| 02E1 | L02E3 | = | 0x02E3 | |
| 02E3 | L02E5 | = | 0x02E5 | |
| 02E5 | L02E6 | = | 0x02E6 | ;? TPS A/D COUNTS |
| 02E6 | L02E7 | = | 0x02E7 | ; |
| 02E7 | L02E9 | = | 0x02E9 | ; |
| 02E9 | L02EA | = | 0x02EA | ; |
| 02EA | L02EB | = | 0x02EB | ;AN IAC VARIABLE |
| 02EB | L02ED | = | 0x02ED | ; |
| 02ED | L02EE | = | 0x02EE | ;IDLE SPARK TERM |
| 02EE | L02EF | = | 0x02EF | ;PREV RPM (FOR PE MODE) |
| 02EF | L02F0 | = | 0x02F0 | ;TIMER |
| 02F0 | L02F1 | = | 0x02F1 | |
| 02F1 | L02F2 | = | 0x02F2 | ;DEFAULT %TPS VS MAF |
| 02F2 | L02F3 | = | 0x02F3 | ;? COUNTER, FOR TORQ MGMT |
| 02F3 | L02F5 | = | 0x02F5 | ; |
| 02F5 | L02F7 | = | 0x02F7 | |
| 02F7 | L02F8 | = | 0x02F8 | ;TCC PWM VARIABLE |
| 02F8 | L02F9 | = | 0x02F9 | |
| 02F9 | L02FA | = | 0x02FA | ;RPM INCREASING TIMER (FOR IDLE) |
| 02FA | L02FD | = | 0x02FD | |
| 02FD | L02FE | = | 0x02FE | ;16 BIT DELTA RPM/12.5 BETWEEN DESIRED AND ACTUAL |
| 02FE | L0300 | = | 0x0300 | ;IS ADDED TO SPK RET |
| 0300 | L0301 | = | 0x0301 | ;F31/TCC PWM VARIABLE |
| 0301 | L0303 | = | 0x0303 | |
| 0303 | L0304 | = | 0x0304 | |
| | L0305 | = | 0x0305 | ;DELTA 3X REF PERIOD |
| | L0306 | = | 0x0306 | |
| | L0307 | = | 0x0307 | |
| | L0308 | = | 0x0308 | |
| | L0309 | = | 0x0309 | |

| | | | | |
|------|-------|---|--------|---|
| 0304 | L030A | = | 0x030A | |
| 030A | L030B | = | 0x030B | ;2S COMP L0123 - TABLE LOOKUP RESULT |
| 030B | L030C | = | 0x030C | ;BPW (CRANKING) |
| 030C | L030E | = | 0x030E | ;A SPARK TERM - FOR A/C |
| 030E | L0310 | = | 0x0310 | ;FILTERED MPH |
| 0310 | L0312 | = | 0x0312 | |
| 0312 | L0314 | = | 0x0314 | ;F31 VARIABLE |
| | L0315 | = | 0x0315 | |
| | L0316 | = | 0x0316 | ;A/C PRESSURE A/D COUNTS |
| 0314 | L0317 | = | 0x0317 | ;A.I.R. TIMER |
| 0317 | L0319 | = | 0x0319 | ;A.I.R. TIMER |
| 0319 | L031B | = | 0x031B | ;FAN #1 ON TIMER |
| 031B | L031D | = | 0x031D | |
| 031D | L031F | = | 0x031F | ;? ASYNC FACTOR VS COOLANT TEMP LOOKUP |
| 031F | L0320 | = | 0x0320 | ;8D0B TABLE LOOKUP RESULT |
| 0320 | L0321 | = | 0x0321 | ;8D1C TABLE LOOKUP RESULT |
| 0321 | L0322 | = | 0x0322 | ;8D47 TABLE LOOKUP RESULT |
| 0322 | L0323 | = | 0x0323 | ;A/F OFFSET FOR MOVING VEH - 8EBD TABLE LOOKUP RESULT |
| 0323 | L0324 | = | 0x0324 | ;STARTUP A/F - 8E86 TABLE LOOKUP RESULT |
| | L0325 | = | 0x0325 | ;SHIFT LIGHT ON TIMER |
| | L0326 | = | 0x0326 | |
| | L0327 | = | 0x0327 | |
| 0324 | L0328 | = | 0x0328 | ;PREVIOUS FMD INPUT STATUS WORD |
| | L0329 | = | 0x0329 | |
| | L032A | = | 0x032A | |
| | L032B | = | 0x032B | |
| | L032C | = | 0x032C | ;F31 VARIABLE (LOOKUP RESULT) |
| | L032D | = | 0x032D | ;F31 VARIABLE |
| 0328 | L0333 | = | 0x0333 | ;RESERVED FOR CHECKING STACK OVERFLOW |
| 0333 | L033A | = | 0x033A | |
| 033A | L0380 | = | 0x0380 | |
| 0380 | L03C0 | = | 0x03C0 | |
| 03C0 | L03C3 | = | 0x03C3 | |
| 03C3 | L03C4 | = | 0x03C4 | |
| 03C4 | L03CC | = | 0x03CC | |
| 03CC | L03D6 | = | 0x03D6 | |
| 03D6 | L03D8 | = | 0x03D8 | |
| 03D8 | L03DA | = | 0x03DA | |
| 03DA | L03DD | = | 0x03DD | |
| 03DD | L03E4 | = | 0x03E4 | |
| 03E4 | L03EA | = | 0x03EA | |
| 03EA | L03EC | = | 0x03EC | |
| 03EC | L03FC | = | 0x03FC | |
| 03FC | L046A | = | 0x046A | ;AIR FUEL RATIO - NOT USED FOR ANY CALCS |
| 046A | L046B | = | 0x046B | ; |
| 046B | L046C | = | 0x046C | ;A/D READ OF CH #1 - KNOCK SENSOR |
| 046C | L046D | = | 0x046D | ;L864B TABLE LOOKUP RESULT |
| 046D | L046E | = | 0x046E | ; |
| 046E | L046F | = | 0x046F | ;OLD MINOR LOOP COUNTER |
| 046F | L0471 | = | 0x0471 | ;L867A LOOKUP RESULT |
| 0471 | L0480 | = | 0x0480 | ; |
| 0480 | L2707 | = | 0x2707 | |
| | L3DC0 | = | 0x3DC0 | ;INPUT |
| | L3DC2 | = | 0x3DC2 | ;NOT USED |
| | L3DC4 | = | 0x3DC4 | ;NOT USED |
| 2707 | L3DC6 | = | 0x3DC6 | ;PASSKEY II INPUT |
| | L3DC8 | = | 0x3DC8 | ;RELATED TO SMC ? ENGAGED INPUT |
| | L3DCA | = | 0x3DCA | ;PSPS INPUT |
| 3DC6 | L3DCC | = | 0x3DCC | ;OUTPUT - ? PWM CONTROL - |
| 3DCC | L3DCE | = | 0x3DCE | ;A.I.R. SOLN PWM OUTPUT |
| | L3DD0 | = | 0x3DD0 | ;OUTPUT |

| | | | |
|------|---------|--------|--|
| | L3DD2 = | 0x3DD2 | ;PWM OUTPUT - DEGR SOLN A (PIN A5) |
| 3DCE | L3DD4 = | 0x3DD4 | ;TCC PWM SOLENOID OUTPUT (PIN B9) (OR SHIFT LITE) |
| | L3DD6 = | 0x3DD6 | ;NOT USED |
| | L3DD8 = | 0x3DD8 | ;SHIFT SOLENOID A PWM CONTROL (PIN A22) |
| | L3DDA = | 0x3DDA | ;SHIFT SOLENOID B PWM CONTROL (PIN A21) |
| 3DD4 | L3DEA = | 0x3DEA | ;PWM OUTPUT - ???? |
| | L3DEC = | 0x3DEC | ;INPUT ONLY ? not related to rpm anyway |
| | L3DF0 = | 0x3DF0 | ;OUTPUT ONLY |
| 3DEA | L3DF2 = | 0x3DF2 | ;OUTPUT ONLY |
| | L3DFA = | 0x3DFA | ;?? |
| 3DF2 | L3DFC = | 0x3DFC | ;? 2 ND MCU CNTL REG |
| | | | ;b0 |
| | | | ;b1 |
| | | | ;b2 1 = |
| | | | ;b3 |
| | | | ;b4 |
| | | | ;b5 |
| | | | ;b6 |
| | | | ;b7 |
| | L3DFD = | 0x3DFD | ;? 2 ND MCU CNTL REG |
| | | | ;b0 |
| | | | ;b1 1 = ? TRANSMIT DATA |
| | | | ;b2 1 = ? TRANSMIT DATA |
| | | | ;b3 1 = |
| | | | ;b4 |
| | | | ;b5 |
| | | | ;b6 |
| | | | ;b7 |
| 3DFC | L3F00 = | 0x3F00 | ; |
| 3F00 | L3F80 = | 0x3F80 | ;FUEL INJ #1 OUTPUT |
| 3F80 | L3F82 = | 0x3F82 | ;FUEL INJ #2 OUTPUT |
| 3F82 | L3F84 = | 0x3F84 | ;FUEL INJ #3 OUTPUT |
| 3F84 | L3F86 = | 0x3F86 | ;FUEL INJ #4 OUTPUT |
| 3F86 | L3FAC = | 0x3FAC | ;? TIMER OUTPUT - FOR FUEL INJECTOR OUTPUTS |
| | | | ;HAS TO BE 0x7FFF, ELSE FUEL INJECTOR PW DOESN'T |
| | | | ; CHANGE AS IT SHOULD, and the first 3 injectors are |
| | | | ; 100% dc all the time |
| | L3FAE = | 0x3FAE | ; |
| | L3FB0 = | 0x3FB0 | ; |
| | L3FB2 = | 0x3FB2 | ; |
| | L3FB4 = | 0x3FB4 | ; |
| | L3FB6 = | 0x3FB6 | ; |
| | L3FBC = | 0x3FBC | ;? TRANNY CNTL REG |
| 3FAC | L3FC0 = | 0x3FC0 | ;3x REF PERIOD |
| 3FC0 | L3FC2 = | 0x3FC2 | ;INPUT |
| 3FC2 | L3FC4 = | 0x3FC4 | ;a ref period |
| 3FC4 | L3FC6 = | 0x3FC6 | ;PA2 COUNTER - for maf calc |
| 3FC6 | L3FC8 = | 0x3FC8 | ;EST VOLTAGE |
| 3FC8 | L3FCA = | 0x3FCA | ;KNOCK SENSOR INPUT |
| | L3FCC = | 0x3FCC | ;PWM OUTPUT - DEGR SOLN C (PIN A10) |
| 3FCA | L3FCE = | 0x3FCE | ;OUTPUT |
| 3FCE | L3FD0 = | 0x3FD0 | ;FUEL INJ #5 OUTPUT |
| 3FD0 | L3FD2 = | 0x3FD2 | ;PWM OUTPUT - A/C RELAY CONTROL (PIN A7) |
| 3FD2 | L3FD4 = | 0x3FD4 | ;PWM OUTPUT - CC INHIBIT SIGNAL (PIN A13) |
| | L3FD6 = | 0x3FD6 | ;OUTPUT - CCP DUTY CYCLE |
| | L3FD8 = | 0x3FD8 | ;PWM OUTPUT - OIL LEVEL LIGHT (PIN A4) |
| | L3FDA = | 0x3FDA | ;PWM OUTPUT - FAN (PIN A8) |
| 3FD4 | L3FDC = | 0x3FDC | ;SPARK DWELL (EST PW) |
| 3FDC | L3FE0 = | 0x3FE0 | ;? ROAD SPEED PULSE COUNTER |
| 3FE0 | L3FE2 = | 0x3FE2 | ;OUTPUT - IS A PRODUCT OF |
| | | | ; [(L00C8*L00CA)+L00E3+L00CB(OR C4)]-L81C9 |
| | | | ;ALSO MAY BE L3FC4 |
| | | | ;not the timer for sefi |

| | | | | |
|------|-------|---|--------|--|
| 3FE2 | L3FE4 | = | 0x3FE4 | ;B CNTR, START NEXT DWELL |
| 3FE4 | L3FE6 | = | 0x3FE6 | ;? LAST EST FALL spark |
| 3FE6 | L3FE8 | = | 0x3FE8 | ;? CURRENT EST FALL |
| | L3FEA | = | 0x3FEA | ;PWM OUTPUT - DEGR SOLN B (PIN A11) |
| 3FE8 | L3FEC | = | 0x3FEC | ;B CNTR, LAST DRP |
| 3FEC | L3FF2 | = | 0x3FF2 | ;? ASYNC PW |
| | L3FF4 | = | 0x3FF4 | ; |
| 3FF2 | L3FF6 | = | 0x3FF6 | ;OUTPUT |
| 3FF4 | L3FF8 | = | 0x3FF8 | ;MAF SENSOR INPUT |
| | L3FFA | = | 0x3FFA | ;CPU STATUS REG - ?NOT USED |
| 3FF6 | L3FFC | = | 0x3FFC | ;CPU CNTL REG MSB |
| | L3FFD | = | 0x3FFD | ;CPU CNTL REG LSB |
| | | | | ;B4 1 = EST ENABLED |
| | L4000 | = | 0x4000 | ;SPI DATA REGISTER |
| | | | | ;B7 AS AN OUTPUT, 1 = ALLOW SERIAL DATA TRANSMIT |
| | | | | ; 0 = PREVENT SERIAL DATA TRANSMIT |
| 3FFC | L4001 | = | 0x4001 | ;I/O CONTROL REGISTER, INCLUDING SPI CONTROL |
| | | | | ;b0 |
| | | | | ;b1 |
| | | | | ;b2 |
| | | | | ;b3 |
| | | | | ;b4 |
| | | | | ;b5 |
| | | | | ;b6 |
| | | | | ;b7 0 = ? START TRANSMISSION |
| 4001 | L4002 | = | 0x4002 | ;I/O PORT (? PROBABLY ALL OUTPUTS) |
| | | | | ;THINK THIS IS SIMILAR TO OTHER P4 ECMS |
| | | | | ;%10000000 chip select to output only SPI device |
| | | | | ;%01000000 chip select to in and out SPI device |
| | | | | ;%00001000 chip select to SPI interfaced ADC |
| | | | | ;%00000100 chip select to in and out SPI device |
| | | | | ;%00000010 IAC phase B output |
| | | | | ;%00000001 IAC phase A output |
| | L4003 | = | 0x4003 | ;DDR, PROBABLY FOR \$4004 |
| | L4004 | = | 0x4004 | ;I/O PORT |
| 4002 | L4005 | = | 0x4005 | ;REAL TIME COUNTER |
| | L4006 | = | 0x4006 | ;CAPTURE REG |
| 4006 | L4007 | = | 0x4007 | ;CPU TX/RX CNTL REG (8192 BAUD SCI) |
| | | | | ;B0 1 = CNTR INTERRUPT (BREAK ENABLED) |
| | | | | ;B1 1 = RX WAKE UP BIT ENABLED |
| | | | | ;B2 1 = RX ENABLED |
| | | | | ;B3 1 = TX ENABLED |
| | | | | ;B4 1 = IDLE INTERRUPT ENABLED |
| | | | | ;B5 1 = RDRF AND OR INTERRUPTS ENABLED |
| | | | | ;B6 1 = TC INTERRUPT ENABLED |
| | | | | ;B7 1 = TDRE INTERRUPT ENABLED |
| 4007 | L4008 | = | 0x4008 | ;CPU TX/RX STATUS REG (8192 BAUD SCI) |
| | | | | ;INTERRUPT SOURCE FLAGS |
| | | | | ;%10000000 TRANSMIT DATA REG EMPTY (TDRE) |
| | | | | ;%01000000 TRANSMIT COMPLETE (TC) |
| | | | | ;%00100000 RECEIVE DATA REG FULL (RDRF) |
| | | | | ;%00010000 ? IDLE |
| | | | | ;%00001000 OVERRUN |
| | | | | ;%00000100 NOISE FLAG |
| | | | | ;%00000010 FRAMING ERROR |
| | | | | ;%00000001 IRQ FLAG |
| 4008 | L4009 | = | 0x4009 | ;CPU RX REG (8192 BAUD SCI) |
| 4009 | L400A | = | 0x400A | ;CPU TX REG (8192 BAUD SCI) |
| 400A | L400B | = | 0x400B | ;COP ARM |
| | L400C | = | 0x400C | ;COP WATCHDOG |
| 400B | L5000 | = | 0x5000 | |
| 5000 | L5003 | = | 0x5003 | |

```

5003 L5009 = 0x5009
5009 L500C = 0x500C
500C L500F = 0x500F
500F L5868 = 0x5868
5868 L5BF2 = 0x5BF2
5BF2 L6000 = 0x6000
6000 L7E03 = 0x7E03

```

```

8000 .area CODE1 (ABS)
8000 .org 0x8000
8000
8000 0x25 ;PROM ID A
8001 0xC0 ;PROM ID B
8002 0xFF ;
8003 0xFF ;
8004 0xFF
8005 0xFF
8006 0x35,0x35 ;CHECKSUM
8008 0x2E ;MASK ID

```

THESE ARE OPTION FLAGS - SET TO 00 OR FF

```

-----
8009 0xFF ;OPTION FLAG - A/C PRESSURE SENSOR PRESENT
800A 0x00 ;OPTION FLAG - A.I.R. PUMP IN USE
800B 0x00 ;? NOT USED
800C 0xFF ;OPTION FLAG - RELATED TO PSPS INPUT & SMC
800D 0xFF ;OPTION FLAG - RELATED TO FAN #1 OUTPUT
;WHEN CLEAR, FAN ISN'T COMMANDED VIA PIN A2
800E 0xFF ;OPTION FLAG - RELATED TO FAN #2 OUTPUT
; WHEN CLEAR AND 800D CLEAR, FAN #1 IS
; COMMANDED VIA PIN A8, AND FAN #2 IS
; COMMANDED VIA ???
800F 0xFF ;OPTION FLAG - TYPE OF TCC IN USE (? PRNDL SW IN USE)
8010 0x00 ;OPTION FLAG - SET = MANUAL TRANNY OPTION
8011 0xFF ;OPTION WORD - SET = PASSKEY II - VATS
8012 0x00 ;FB8B VS FB8A FOR BAD CYL ID COMPARISON
; ??? FIRING ORDER ??
8013 0x00 ;NOT USED
8014 0xFF ;RELATED TO IAC CONTROL - STATUS FLAGS VS ? EVENTS
8015 0x00 ;RELATED TO CALCULATION OF L01DF VARIABLE
8016 0xFF ;OIL LEVEL SENSOR AND WARNING LIGHT IN USE
8017 0x00 ;OPTION TO USE 817F SPARK TABLE
8018 0xFF ;RELATED TO SPARK PERIOD 3FC8
;WHEN CLEAR, EST BYPASS CNTL ISN'T SW TO 5V
; WHEN ENGINE RUN ALLOWED
; ? 24X SENSOR IN USE ??
8019 0x00 ;RELATED TO BIT VATS INPUT AND
801A 0xFF ;OPTION TO USE F7MAXTRQ FOR 3RD GR TRQ MGMT
801B 0xFF ;SET = BLM CELL 16 IS IDLE CELL
801C 0x00 ;RELATED TO FUEL OUT - ASYNC
801D 0xFF ;SKIP _____ IF MAF MALF FLAG SET
801E 0xFF ;SKIP _____ IF MAF MALF FLAG SET
801F 0xFF ;SKIP _____ IF MAF MALF FLAG SET
8020 0x00 ;RELATED TO CALC OF DEFAULT MAF
8021 0xFF ;OPTION WORD - ? HUD RELATED
8022 0xFF ;RELATED TO L01B9 VARIABLE
8023 0x00 ;OPTION WORD - ? HUD RELATED
8024 0x00 ;NOT USED

```

MAIN SPARK ADVANCE TABLE VS NLRPMX AND LV8
N = DEG * 256/90

STARTUP SPARK CORRECTION VS RUNTIME LSB AND LV8
 - NOT USED, SINCE 817E = 00 AND 8017 IS CLEAR

```

-----
817F  0x00  0      MIN RUNTIME VALUE
      0x20  32      MIN LV8 VALUE
      0x07          COLUMNS
      0x00,0x00,0x00,0x00,0x00,0x00,0x00
      0x00,0x00,0x00,0x00,0x00,0x00,0x00
      0x00,0x00,0x00,0x00,0x00,0x00,0x00
      0x00,0x00,0x00,0x00,0x00,0x00,0x00
      0x00,0x00,0x00,0x00,0x00,0x00,0x00
      0x00,0x00,0x00,0x00,0x00,0x00,0x00
      0x00,0x00,0x00,0x00,0x00,0x00,0x00
      0x00,0x00,0x00,0x00,0x00,0x00,0x00
      0x00,0x00,0x00,0x00,0x00,0x00,0x00
      0x00,0x00,0x00,0x00,0x00,0x00,0x00
      0x00,0x00,0x00,0x00,0x00,0x00,0x00

81C1  0xAB
81C2  0x0C,0x0C      3084          425 RPM      ;SPARK REF ANGLE
81C4  0x03          3          3          ;RPM THRESHOLD TO ALLOW ENGINE RUN
81C5  0x0B,0x60      2912          450 RPM      ;TIME LIMIT TO ALLOW ENGINE RUN
81C7  0x03          3          ;RPM THRESHOLD FOR ? SEFI
81C8  0x10          16          ;CRANKING TIME LIMIT
81C9  0x00          0          ;TIME BETWEEN REF PULSES
81CA  0x09,0x00      ;OFFSET FOR L3FE2 OUTPUT
81CC  0xFF,0xE4      65508 -10 DEG      ;SPARK MAX ADVANCE REL TO REF (2'S COMPL)
81CE  0xFF,0x39      65337 -70 DEG      ;SPARK MAX RETARD REL TO REF (2'S COMPL)
81D0  0x05          5          ;
81D1  0xFF,0xFF      ;INITIAL SPARK ADVANCE
81D3  0x00,0x05      ;KNOCK CAL
81D5  0x50          ;IF RPM/12.5 < THIS, DISABLE ESC
81D6  0x72          ;MAX KNOCK RETARD
81D7  0x55          ;MAX KNOCK RETARD ?IN PE MODE
81D8  0x8C          ;IF CTS < THIS, DISABLE ESC
81D9  0xA0          160          80 DEG C      ;COOLANT THRESHOLD
81DA  0x48          72          1800 RPM      ;RPM THRESHOLD
81DB  0x0D          13          ;D-TPS
81DC  0x00          0          ;TIME LIMIT, FOR CUMMULATIVE KNOCK RETARD
81DD  0x03          3          ;TIME LIMIT, FOR CUMMULATIVE KNOCK RETARD
81DE  0x28          ;MAX CUMMULATIVE KNOCK RETARD
81DF  0x06          ;CUMMULATIVE KNOCK RET RECOVERY RATE
81E0  0x05          5          1.95% TPS      ;
81E1  0xFE          ;POS RPM ERROR THRESHOLD TO IMPLEMENT MULTIPLIERS
81E2  0x01          ;POS RPM ERROR MULTIPLIER
81E3  0x01          ;POS RPM ERROR MULTIPLIER
81E4  0x20
81E5  0x04          ;DELTA RPM THRESHOLD TO IMPLEMENT OVERRIDE MULT
81E6  0x44          ;UNDERIDLE SPARK MULTIPLIER WHEN IN P/N
81E7  0x44          ;UNDERIDLE SPARK MULTIPLIER WHEN IN GEAR
81E8  0x80
81E9  0xCD
81EA  0x50
81EB  0x00,0x47      ;MAX SPARK REL TO ?REF OR ?TDC
81ED  0x30          IS VS REFFER      ;? F1 EXTENSION HI RPM BREAKPOINT ??
81EE  0x12          ;? HI RPM ADVANCE SLOPE ??
81EF  0x03,0x33      819          ;

THESE 4 GET SUBTRACTED FROM L00CA
81F1  0x21          33
81F2  0x4C          76
81F3  0x77          119
81F4  0xA1          161          ;
81F5  0x02          2          ;FILTER COEFF FOR EGR SPARK ADVANCE

```

BASE EGR SA VS LV8 AND RPM/25
N = DEG * 256/90

```
-----
81F6  0x20          R MIN - 800 RPM
      0x40          Q MIN - 64 LOAD COUNTS LV8
      0x0D          COLUMNS
      0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
      0x00,0x0E,0x18,0x1B,0x1D,0x19,0x12,0x0E,0x08,0x05,0x03,0x00,0x00,0x00
      0x00,0x0E,0x12,0x16,0x1B,0x15,0x0F,0x0B,0x08,0x04,0x00,0x00,0x00,0x00
      0x00,0x0E,0x15,0x19,0x15,0x0E,0x09,0x07,0x07,0x03,0x00,0x00,0x00,0x00
      0x00,0x11,0x18,0x1B,0x15,0x11,0x0C,0x08,0x05,0x03,0x00,0x00,0x00,0x00
      0x00,0x0E,0x18,0x16,0x11,0x0E,0x0C,0x09,0x07,0x03,0x00,0x00,0x00,0x00
      0x00,0x16,0x19,0x16,0x0F,0x0C,0x09,0x07,0x03,0x01,0x00,0x00,0x00,0x00
```

EGR SA MULTIPLIER VS FILTERED ENGINE TORQUE
RESULT IS A MULTIPLIER OF BASE EGR SA -> L016D

```
-----
8254  0x20
      0x80
      0x80
      0x80
      0x80
      0x80
      0x68
      0x50
      0x50
      0x50
      0x50
```

SPARK CORRECTION VS IAT AND LV8
N = (DEG*256/90) + 100

```
-----
825F  0x08   8      MIN IAT VALUE
      0x20  32      MIN LV8 VALUE
      0x0F          COLUMNS
      0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64
      0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64
      0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64
      0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64
      0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64
      0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64
      0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64
```

TCC LOCKED SPARK CORRECTION VS LV8 AND RPM/25
N = (DEG*256/90) + 100

```
-----
82CB  0x60   96      2400  MIN RPM VALUE
      0x60   96      96      MIN LV8 VALUE
      0x0B          COLUMNS
      0x64,0x64,0x64,0x64,0x64,0x61,0x61,0x5E,0x5E,0x5E,0x5E
      0x64,0x64,0x64,0x64,0x64,0x61,0x5E,0x5E,0x5E,0x5E,0x5E,0x5E
      0x64,0x64,0x64,0x64,0x64,0x61,0x61,0x5E,0x5E,0x5E,0x5E,0x5E
      0x64,0x64,0x64,0x64,0x64,0x64,0x61,0x61,0x61,0x61,0x61,0x61
      0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64
```

TCC LOCKED SPARK CORRECTION MULTIPLIER VS IAT

```
-----
8305  0xFF
      0xFF
      0xFF
```

```

0xFF
0xFF
0xFF
0xFF
0xFF
0xFF

```

POWER ENRICHMENT SPARK CORRECTION VS PE TIME
 $N = (DEG * 256 / 90) + 100$

| | HEX | DEC | VALUE | PE TIME |
|-------|------|-----|---------|---------|
| ----- | | | | |
| 830E | 0x6C | 108 | 2.8 DEG | |
| | 0x6C | 108 | 2.8 DEG | |
| | 0x6C | 108 | 2.8 DEG | |
| | 0x6C | 108 | 2.8 DEG | |
| | 0x6C | 108 | 2.8 DEG | |
| | 0x6C | 108 | 2.8 DEG | |
| | 0x64 | 100 | 0 DEG | |
| | 0x64 | 100 | 0 DEG | |
| | 0x64 | 100 | 0 DEG | |

POWER ENRICHMENT SPARK MULTIPLIER VS RPM/25
 - FOR 830E LOOKUP RESULT >= 100 (SPARK ADVANCE)

| | HEX | DEC | VALUE | RPM |
|-------|------|-----|-------|------|
| ----- | | | | |
| 8317 | 0x80 | 128 | 0.5 | 800 |
| | 0xC0 | 192 | 0.75 | 1600 |
| | 0xFF | 255 | 1 | 2400 |
| | 0xFF | 255 | 1 | 3200 |
| | 0xFF | 255 | 1 | 4000 |
| | 0xFF | 255 | 1 | 4800 |
| | 0xFF | 255 | 1 | 5600 |
| | 0xFF | 255 | 1 | 6400 |

POWER ENRICHMENT SPARK MULTIPLIER VS RPM/25
 - FOR 830E LOOKUP RESULT < 100 (SPARK RETARD)

| | HEX | DEC | VALUE | RPM |
|-------|------|-----|-------|------|
| ----- | | | | |
| 831F | 0x80 | 128 | 0.5 | 800 |
| | 0xC0 | 192 | 0.75 | 1600 |
| | 0xFF | 255 | 1 | 2400 |
| | 0xFF | 255 | 1 | 3200 |
| | 0xFF | 255 | 1 | 4000 |
| | 0xFF | 255 | 1 | 4800 |
| | 0xFF | 255 | 1 | 5600 |
| | 0xFF | 255 | 1 | 6400 |

ESC RECOVERY RATE VS NLRPMX

| | | | | |
|-------|------|--|--|--|
| ----- | | | | |
| 8327 | 0x0B | | | |
| | 0x14 | | | |
| | 0x12 | | | |
| | 0x12 | | | |
| | 0x10 | | | |
| | 0x07 | | | |
| | 0x09 | | | |
| | 0x09 | | | |
| | 0x0A | | | |
| | 0x0B | | | |
| 8331 | 0x20 | | | |

THE INVERSE OF THIS LOOKUP RESULT GOES INTO L030B -> L00CA
 A SPARK VARIABLE, USED WHEN RPM <= 511

| | | |
|------|--|-------------|
| 8332 | 0x18 | MIN RPM |
| | 0x00 | MIN COOLANT |
| | 0x11 | 17 COLUMNS |
| | 0x6F, 0x6F, 0x72, 0x78, 0x7A, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D | |
| | 0x6F, 0x6F, 0x72, 0x78, 0x7A, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D | |
| | 0x6F, 0x6F, 0x72, 0x78, 0x7A, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D | |
| | 0x6F, 0x6F, 0x72, 0x78, 0x7A, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x84, 0x84, 0x89, 0x89, 0x8F, 0x8F, 0x8F | |
| | 0x6F, 0x6F, 0x72, 0x78, 0x7A, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x84, 0x84, 0x89, 0x89, 0x8F, 0x8F, 0x8F | |
| | 0x6F, 0x6F, 0x72, 0x78, 0x7A, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x7D, 0x84, 0x84, 0x89, 0x89, 0x8F, 0x8F, 0x8F | |

[illegible]

SPARK RETARD VS COOLANT TEMP - ?USED WHEN AT IDLE

[illegible]

| | | | | |
|------|-----------|-----|------------|---------------------------------------|
| 8419 | 0x00,0x00 | | | |
| 841B | 0x01 | 1 | 1 MPH | ;LAUNCH MODE VEH SPEED THRESHOLD |
| 841C | 0x50 | 80 | 80 COUNTS | ;LAUNCH MODE LV8 COUNTS THRESHOLD |
| 841D | 0x03 | 3 | 1.1 %TPS | ;LAUNCH MODE D-TPS THRESHOLD |
| 841E | 0xA0 | 160 | 80 DEG C | ;LAUNCH MODE TEMP THRESHOLD |
| 841F | 0x00,0x40 | 64 | 64 SECONDS | ;LAUNCH MODE RUN-TIME THRESHOLD |
| 8421 | 0x10 | 16 | | ;LAUNCH MODE TIME |
| 8422 | 0x05 | 5 | 1.9 %TPS | ;NEGATIVE D-TPS TO LEAVE LAUNCH MODE |
| 8423 | 0xFE | 254 | | ;D-LV8 TO LEAVE LAUNCH MODE |
| 8424 | 0xFE | 254 | | ;FILTER CONSTANT FOR LAUNCH MODE TIME |

[illegible]

| | | | | |
|------|----------------|---|---------|---------------------------------|
| 84AF | 0x00 | 0 | 0 DEG C | ; |
| 84B0 | 0x00 | 0 | 0 RPM | ; |
| 84B1 | 0x00 | 0 | 0 %TPS | ; |
| 84B2 | 0x00 | 0 | 0 %TPS | ; |
| 84B3 | 0x00 | | | ;D-TPS% |
| 84B4 | 0x00 | 0 | 0 %TPS | ;D-TPS% |
| 84B5 | 0x00 | | | |
| 84B6 | 0x00,0x00 | | | |
| 84B8 | 0x00 | | | |
| 84B9 | 0x00 | | | |
| | | | | |
| 84BA | 0x00,0x00,0x00 | | | |
| 84BD | 0x00,0x00,0x00 | | | |
| 84C0 | 0x00,0x00,0x00 | | | |
| 84C3 | 0x00,0x00,0x00 | | | |
| 84C6 | 0x00,0x00,0x00 | | | |
| 84C9 | 0x00,0x00,0x00 | | | |
| 84CC | 0x00,0x00,0x00 | | | |
| 84CF | 0x00,0x00,0x00 | | | |
| 84D2 | 0x00,0x00,0x00 | | | |
| | | | | |
| 84D5 | 0x06 | 6 | 6 | ; INITIAL VALUE FOR CYL COUNTER |
| 84D6 | 0x05 | 5 | 5 | ; |
| 84D7 | 0x02 | 2 | 2 | ; |

ENGINE EFFICIENCY VS FUEL AIR RATIO AND RPM/25
TIS A MULTIPLIER FOR ENGINE TORQUE CALC

```

-----
84D8  0x20  32      800    MIN RPM
      0x60                      MIN FAR
      0x06                      COLUMNS
      0x26,0x26,0x2A,0x2F,0x2A,0x27
      0x66,0x61,0x62,0x59,0x5B,0x56
      0x6E,0x85,0x8A,0x8A,0x87,0x83
      0x71,0x85,0x8D,0x8B,0x87,0x82
      0x6F,0x85,0x8D,0x8D,0x89,0x85
      0x74,0x88,0x8C,0x8C,0x86,0x83
      0x74,0x89,0x8D,0x8B,0x88,0x84
      0x72,0x88,0x8C,0x8B,0x88,0x85
      0x72,0x87,0x8B,0x8B,0x87,0x83
      0x73,0x87,0x8D,0x8A,0x86,0x83
      0x76,0x86,0x88,0x8B,0x84,0x81
      0x72,0x85,0x8C,0x8C,0x87,0x86
      0x72,0x85,0x8C,0x8C,0x85,0x85
      0x72,0x85,0x8C,0x8C,0x85,0x85

```

MASS AIR CORRECTION VS RPM AND CTS2
2S COMP OF RESULT/2 GOES INTO L0276 - ENGINE TORQUE

```

-----
852F  0x20  32      800    MIN RPM
      0x08      8          MIN COOLANT
      0x07                      COLUMNS
      0x62,0x34,0x2E,0x2C,0x29,0x29,0x29
      0x55,0x30,0x28,0x25,0x23,0x22,0x21
      0x4B,0x30,0x27,0x24,0x23,0x20,0x1F
      0x45,0x30,0x29,0x26,0x23,0x20,0x1E
      0x48,0x34,0x2C,0x27,0x23,0x20,0x1F
      0x4C,0x38,0x2D,0x2B,0x26,0x24,0x20
      0x4D,0x3D,0x31,0x2D,0x29,0x26,0x23
      0x4E,0x3E,0x37,0x31,0x2E,0x2B,0x28
      0x50,0x3F,0x39,0x34,0x31,0x2E,0x2C
      0x56,0x43,0x3D,0x36,0x34,0x30,0x2C
      0x59,0x47,0x41,0x3D,0x38,0x34,0x2D
      0x5B,0x48,0x43,0x3F,0x3A,0x39,0x32

```

0x5C,0x49,0x44,0x41,0x3C,0x3D,0x34
0x5F,0x4A,0x45,0x42,0x3F,0x3F,0x36

8594 0x40 64 ;MULTIPLIER WHEN L0265 IS FF
8595 0x00,0x14 20 ;TIME SINCE RUN ENABLE FOR USING L8597 TABLE

2D TABLE VS RATIO OF ENGINE TO TURBINE RPM
- MULTIPLIER FOR (MAF*REFPER*8644), TO MAKE TURBINE TORQUE

8597 0x8F
0x8A
0x84
0x7F
0x79
0x73
0x6D
0x67
0x61
0x5C
0x56
0x50
0x4A
0x44
0x41
0x41
0x40

F7MAXTRQ - MAX SPARK TORQ MGMT VS L027B AND LIMITED NLRPMX
-> LOOKUP RESULT GOES INTO L027A ?2'S COMP MAX SPARK

85A8 0x00
0x80
0x09
0x33,0x33,0x33,0x33,0x33,0x25,0x2E,0x17,0x00
0x3F,0x3F,0x3F,0x3F,0x3F,0x3F,0x44,0x06,0x00
0x44,0x44,0x44,0x44,0x44,0x3F,0x39,0x30,0x00
0x44,0x44,0x44,0x3F,0x36,0x2E,0x28,0x1F,0x00
0x44,0x44,0x44,0x3C,0x33,0x30,0x25,0x1A,0x00
0x44,0x44,0x41,0x3C,0x36,0x2E,0x25,0x1A,0x00
0x44,0x44,0x44,0x3C,0x33,0x2E,0x22,0x1C,0x00
0x44,0x44,0x44,0x3F,0x36,0x30,0x28,0x1A,0x00
0x44,0x44,0x44,0x3F,0x33,0x2B,0x22,0x00
0x44,0x44,0x44,0x41,0x39,0x33,0x2B,0x1F,0x00
0x44,0x44,0x3F,0x39,0x33,0x2E,0x28,0x1A,0x00
0x44,0x44,0x41,0x3C,0x33,0x2E,0x25,0x1A,0x00
0x44,0x44,0x41,0x3C,0x33,0x2E,0x25,0x1C,0x00
0x44,0x3F,0x3C,0x36,0x2E,0x2B,0x22,0x1A,0x00
0x3C,0x36,0x30,0x2E,0x28,0x22,0x1F,0x14,0x00
0x3C,0x36,0x30,0x2E,0x28,0x22,0x1F,0x14,0x00
0x3C,0x36,0x30,0x2E,0x28,0x22,0x1F,0x14,0x00

8644 0xBE 190 ;FUDGE FACTOR FOR TORQUE CALCULATION
8645 0x00,0x50 80 ;
8647 0x50 80 2000 RPM ;RPM THRESHOLD
8648 0x59 89 ;THRESHOLD TO IMPLEMENT TORQUE MGMT SPK RETARD
;
8649 0x34 52 ;THRESHOLD TO IMPLEMENT TORQUE MGMT SPK RETARD
; WHEN TCC LOCKED
864A 0x3C 60 ;THRESHOLD TO IMPLEMENT TORQUE MGMT SPK RETARD
; WHEN F7MAXTRQ TABLE IN USE
; ** IF THESE ARE MAXED, THEN STALL TORQUE MGMT SPARK CONTROL WON'T BE IMPLEMENTED**

2D TABLE

```

864B    0x20,0x20,0x20,0x20,0x20,0x20,0x20,0x40,0x60,0x80,0xA0,0xC0,0xE0,0xFF,0xFF,0xFF,0xFF
865C    0xFF,0xFF,0xFF,0x40,0x2F,0x2D,0x26,0x26,0x26,0x26
8666    0xFF,0xFF,0x40,0x38,0x30,0x2E,0x2C,0x24,0x23,0x23
8670    0xFF,0xFF,0xFF,0x30,0x2A,0x24,0x1F,0x1A,0x15,0x10

```

2D TABLE VS L0287 -> MULTIPLIER FOR L0286
-> L0288 ?2'S COMP MAX SPARK WITH L027A

```

-----
867A    0x00
        0x00
        0xFF
        0xFF
        0xFF
        0xFF
        0xFF
        0xFF
        0x00
        0x00
        0x00
        0x00
        0x00
        0x00
        0x00
        0x00

868B    0x73          115          45 %TPS          ;TPS THRESHOLD IN TORQUE MGMT ROUTINE
868C    0x00,0x10     16           16 RPM ;
868E    0x03          3           3
868F    0x00,0x4B     75           75 RPM ;
8691    0x00,0x28
8693    0x40                      ;FILTER COEFFICIENT
8694    0x73                      ;
8695    0x24          36           ;
8696    0x18
8697    0x40
8698    0xBE,0x00      ;INITIAL/DEFAULT VALUE FOR FILT ENGINE TORQUE
869A    0x96,0x00      ;INITIAL/DEFAULT VALUE FOR FILT ENGINE TORQUE
                        ;      (? FOR EGR)
869C    0x58          88           2200 RPM ;
869D    0x70          112          2800 RPM ;
869E    0x80          128          50% TPS ;
869F    0x97          151          58.9% TPS ;
86A0    0x05                      ;FILTER COEFF FOR ENGINE TORQUE, TO MAKE L005A
86A1    0x48
86A2    0x58
86A3    0x30
86A4    0x33
86A5    0x0A                      ;FILTER COEFF FOR ENGINE TORQUE, TO MAKE L005C
86A6    0xAA          170          ;
86A7    0x8C          140
86A8    0x00,0x19

        ;-----
        ; FAN CALS
        ;-----

86AA    0x32          50           50           ;MPH TO DISABLE FAN #2
86AB    0x32          50           50           ;MPH TO ENABLE FAN #2
86AC    0xCC          204          113 DEG C    ;TEMP TO TURN ON FAN #1 or #2
86AD    0xC7          199          109 DEG C    ;TEMP TO TURN OFF FAN #1 or #2
86AE    0x32          50           50 MPH ;MPH TO DISABLE FAN #1

```

| | | | |
|------|-----------|-----|--|
| 86AF | 0x32 | 50 | 50 MPH ;MPH TO ENABLE FAN #1 |
| 86B0 | 0xC3 | 195 | 106 DEG C ;TEMP TO TURN ON FAN #1 |
| 86B1 | 0xBD | 189 | 101 DEG C ;TEMP TO TURN OFF FAN #1 |
| 86B2 | 0x42 | 66 | ;INTAKE TEMP TO TURN ON FAN #1 |
| 86B3 | 0x3E | 62 | ;INTAKE TEMP TO TURN OFF FAN #1 |
| 86B4 | 0x00,0x00 | 00 | 00 ;FAN #1 TURN ON DELAY TIME |
| 86B6 | 0x1E | 30 | 3 SECONDS ;MIN TIME FOR FAN #1 ON BEFORE ALLOWING FAN #2 |
| 86B7 | 0xCF | 207 | 115 DEG C ;FAN #1 ON IF COOLANT TEMP > THIS, LOW BATT VOLTS |
| 86B8 | 0x60 | 96 | ;MAT TO TURN ON FAN #1, LOW BATT VOLTS |
| 86B9 | 0x02,0x58 | 600 | 60 SECONDS ;LOW BATT VOLTS FAN #1 ON LIMIT |
| 86BB | 0x1E | | ;A/C PRESSURE TO TURN FAN #1 ON |
| 86BC | 0x1C | | ;A/C PRESSURE TO TURN FAN #1 OFF |
| 86BD | 0x8B | | ;A/C PRESSURE TO TURN FAN #2 ON |
| 86BE | 0x71 | | ;A/C PRESSURE TO TURN FAN #2 OFF |
| 86BF | 0x01,0x2C | 300 | 30 SECONDS ;FAN #1 MIN ON TIME |
| 86C1 | 0x01,0x2C | 300 | 30 SECONDS ;ADDITIONAL FAN #1 MIN ON TIME WHEN FAN #2 ON AND |
| | | | ;FAN #2 MIN ON TIME |

| | | | | |
|------|-----------|-----------|-------------------------------|--|
| 86C3 | 0x59 | | | ;THROTTLE HIGH - THROTTLE LOW |
| 86C4 | 0x3E | | | ;INITIAL BASE TPS A/D COUNTS |
| 86C5 | 0x0F | 15 | | ;LOW THROTTLE FILTER COEFFICIENT |
| 86C6 | 0x66 | 106 | 453 mV ;INITIAL O2 A/D COUNTS | |
| 86C7 | 0x20 | | | ;FILTER COEFFICIENT FOR L00A6 -> L00A7 |
| 86C8 | 0xF0 | | | ;COEFFICIENT FOR LV8 CALCULATION |
| 86C9 | 0x15 | %00010101 | | ;FUEL OPTION FLAG |
| | | | * | ;B0 RELATED TO A/C |
| | | | | ;B1 1 = RESET INTEGRATOR ON ACCEL ENRICHMENT |
| | | | * | ;B2 1 = RESET INT ON DECEL ENLEANMENT |
| | | | | ;B3 1 = CHECK CTS BEFORE FUEL OUT |
| | | | * | ;B4 1 = |
| | | | | ;B5 |
| | | | | ;B6 |
| | | | | ;B7 |
| 86CA | 0x00 | 00 | 0 | ; |
| 86CB | 0x00 | 00 | 0 | ; |
| 86CC | 0x9A | 154 | 60 %TPS | ; |
| 86CD | 0x14 | 20 | 500 RPM | ; |
| 86CE | 0x01 | 1 | | ; |
| 86CF | 0x08 | 8 | | ;? REDUCTION FACTOR FOR AFR |
| 86D0 | 0x00 | 00 | -40 DEG C | ;IF COOLANT TEMP < THIS, ? DON'T DO SEFI |
| 86D1 | 0x30,0x00 | 12288 | 187.5 MSEC | ;CRANK TBL SCLR FOR MAX CRANK PW |

CRANK FUEL PW MULT VS COOLANT TEMP

| | HEX | DEC | PW | TEMP |
|--|-----|-----|----|------|
|--|-----|-----|----|------|

| | | | | |
|------|------|--------------------------|--|-----|
| 86D3 | 0x20 | TABLE LENGTH (33 VALUES) | | |
| | 0xB0 | | | -40 |
| | 0xAF | | | -34 |
| | 0xAE | | | -28 |
| | 0x9F | | | -22 |
| | 0x8E | | | -16 |
| | 0x74 | | | -10 |
| | 0x5B | | | -4 |
| | 0x4E | | | 2 |
| | 0x47 | | | 8 |
| | 0x40 | | | 14 |
| | 0x37 | | | 20 |
| | 0x31 | | | 26 |
| | 0x2B | | | 32 |
| | 0x19 | | | 38 |
| | 0x17 | | | 44 |
| | 0x15 | | | 50 |
| | 0x13 | | | 56 |

| | |
|------|-----|
| 0x11 | 62 |
| 0x10 | 68 |
| 0x0F | 74 |
| 0x0E | 80 |
| 0x0C | 86 |
| 0x0A | 92 |
| 0x0A | 98 |
| 0x0E | 104 |
| 0x12 | 110 |
| 0x12 | 116 |
| 0x15 | 122 |
| 0x15 | 128 |
| 0x15 | 134 |
| 0x15 | 140 |
| 0x15 | 146 |
| 0x15 | 152 |

CRANK FUEL PW MULT VS REF PULSES

```

-----
86F5  0xFF
      0xFF
      0xFF
      0xFC
      0xF8
      0xF3
      0xEE
      0xEA
      0xE6
      0xE2
      0xDD
      0xD8
      0xD1
      0xCA
      0xC7
      0xC3
      0xC0

```

CRANK FUEL PW MULTIPLIER VS %TPS

```

-----
8706  0x80  128      0      %TPS
      0x80  128      6.25
      0x84  132      12.5
      0x88  136      18.75
      0x88  136      25.00
      0x88  136      31.25
      0x88  136      37.50
      0x88  136      43.75
      0x88  136      50.00
      0x88  136      56.25
      0x88  136      62.5
      0x88  136      68.75
      0x88  136      75.00
      0x00  00      81.25
      0x00  00      87.50
      0x00  00      93.75
      0x00  00     100.00

```

```

8717  0x20      32                      ;INITIAL VALUE FOR L0030 WHEN CLEAR FLOOD MODE
8718  0x08      8                        ;INITIAL VALUE FOR L0030 (TIMER)
8719  0xD5     213      120 DEG C      ;

```

CRANK FUEL PW MULTIPLIER VS
FILTERED ENGINE TORQUE (L005C)

```

8747      0x00
8748      0x00,0x00      00
874A      0x00

```

[illegible]

[illegible]

FATI ADDITIVE OFFSET VS L0076 (NOT USED)

```

883C      0x04      TABLE LENGTH ( 5 VALUES)
          0x00
          0x00
          0x00
          0x00
          0x00

```

FATI VS STARTUP COOLANT FOR P/N

| | HEX | DEC | VALUE | TEMP |
|------|------|-----|--------------------------|------|
| 8842 | 0x20 | | TABLE LENGTH (33 VALUES) | |
| | 0xFE | | | -40 |
| | 0xFE | | | -34 |
| | 0xFE | | | -28 |
| | 0xEC | | | -22 |
| | 0x99 | | | -16 |
| | 0x66 | | | -10 |
| | 0x4F | | | -4 |
| | 0x3D | | | 2 |
| | 0x30 | | | 8 |
| | 0x2E | | | 14 |
| | 0x17 | | | 20 |
| | 0x14 | | | 26 |
| | 0x13 | | | 32 |
| | 0x11 | | | 38 |
| | 0x11 | | | 44 |
| | 0x0F | | | 50 |
| | 0x0D | | | 56 |
| | 0x0C | | | 62 |
| | 0x0C | | | 68 |
| | 0x09 | | | 74 |
| | 0x06 | | | 80 |
| | 0x04 | | | 86 |
| | 0x03 | | | 92 |
| | 0x03 | | | 98 |
| | 0x03 | | | 104 |
| | 0x10 | | | 110 |
| | 0x10 | | | 116 |
| | 0x10 | | | 122 |
| | 0x10 | | | 128 |
| | 0x10 | | | 134 |
| | 0x10 | | | 140 |
| | 0x10 | | | 146 |
| | 0x10 | | | 152 |

FATI VS STARTUP COOLANT, NOT P/N

| | HEX | DEC | VALUE | TEMP |
|------|------|-----|--------------------------|------|
| 8864 | 0x20 | | TABLE LENGTH (33 VALUES) | |
| | 0x94 | | | -40 |
| | 0x97 | | | -34 |
| | 0x9A | | | -28 |
| | 0x8E | | | -22 |
| | 0x7B | | | -16 |
| | 0x76 | | | -10 |
| | 0x64 | | | -4 |
| | 0x51 | | | 2 |
| | 0x41 | | | 8 |
| | 0x38 | | | 14 |
| | 0x22 | | | 20 |
| | 0x21 | | | 26 |
| | 0x20 | | | 32 |
| | 0x1E | | | 38 |
| | 0x1D | | | 44 |
| | 0x18 | | | 50 |
| | 0x14 | | | 56 |
| | 0x10 | | | 62 |
| | 0x0C | | | 68 |
| | 0x09 | | | 74 |
| | 0x06 | | | 80 |
| | 0x04 | | | 86 |
| | 0x03 | | | 92 |
| | 0x03 | | | 98 |
| | 0x03 | | | 104 |
| | 0x10 | | | 110 |
| | 0x10 | | | 116 |
| | 0x10 | | | 122 |
| | 0x10 | | | 128 |
| | 0x10 | | | 134 |
| | 0x10 | | | 140 |
| | 0x10 | | | 146 |
| | 0x10 | | | 152 |

FATC VS COOLANT TEMP FOR P/N (AFR = 1638.4/N)

| | HEX | DEC | VALUE | TEMP |
|------|------|-----|--------------------------|------|
| 8886 | 0x20 | | TABLE LENGTH (33 VALUES) | |
| | 0xBD | | | -40 |
| | 0xBD | | | -34 |
| | 0xBD | | | -28 |
| | 0xBD | | | -22 |
| | 0xBD | | | -16 |
| | 0xBA | | | -10 |
| | 0xB8 | | | -4 |
| | 0xB3 | | | 2 |
| | 0xAD | | | 8 |
| | 0xA9 | | | 14 |
| | 0xA5 | | | 20 |
| | 0x9E | | | 26 |
| | 0x97 | | | 32 |
| | 0x91 | | | 38 |
| | 0x8C | | | 44 |
| | 0x89 | | | 50 |
| | 0x87 | | | 56 |
| | 0x85 | | | 62 |
| | 0x83 | | | 68 |
| | 0x83 | | | 74 |
| | 0x83 | | | 80 |
| | 0x83 | | | 86 |
| | 0x83 | | | 92 |

| | |
|------|-----|
| 0x83 | 98 |
| 0x83 | 104 |
| 0x83 | 110 |
| 0x83 | 116 |
| 0x83 | 122 |
| 0x83 | 128 |
| 0x83 | 134 |
| 0x83 | 140 |
| 0x83 | 146 |
| 0x83 | 152 |

FATC VS COOLANT TEMP, NOT P/N

| | HEX | DEC | VALUE | TEMP |
|-------|------|-----|--------------------------|------|
| ----- | | | | |
| 88A8 | 0x20 | | TABLE LENGTH (33 VALUES) | |
| | 0x9D | | | -40 |
| | 0x9A | | | -34 |
| | 0x97 | | | -28 |
| | 0x95 | | | -22 |
| | 0x94 | | | -16 |
| | 0x91 | | | -10 |
| | 0x8F | | | -4 |
| | 0x8D | | | 2 |
| | 0x8C | | | 8 |
| | 0x89 | | | 14 |
| | 0x87 | | | 20 |
| | 0x83 | | | 26 |
| | 0x80 | | | 32 |
| | 0x80 | | | 38 |
| | 0x80 | | | 44 |
| | 0x80 | | | 50 |
| | 0x80 | | | 56 |
| | 0x81 | | | 62 |
| | 0x83 | | | 68 |
| | 0x83 | | | 74 |
| | 0x83 | | | 80 |
| | 0x83 | | | 86 |
| | 0x83 | | | 92 |
| | 0x83 | | | 98 |
| | 0x83 | | | 104 |
| | 0x83 | | | 110 |
| | 0x83 | | | 116 |
| | 0x83 | | | 122 |
| | 0x83 | | | 128 |
| | 0x83 | | | 134 |
| | 0x83 | | | 140 |
| | 0x83 | | | 146 |
| | 0x83 | | | 152 |

FATI DECAY DELAY VS NLRPMX, IN GEAR

| | HEX | DEC | VALUE | RPM |
|-------|------|-----|-------|------|
| ----- | | | | |
| 88CA | 0x60 | | | 1200 |
| | 0x60 | | | 1400 |
| | 0x58 | | | 1600 |
| | 0x58 | | | 1800 |
| | 0x58 | | | 2000 |
| | 0x58 | | | 2200 |
| | 0x58 | | | 2400 |
| | 0x50 | | | 2800 |
| | 0x50 | | | 3200 |
| | 0x50 | | | 3600 |
| | 0x40 | | | 4000 |

FATI DECAY DELAY VS NLRPMX, P/N

| | | |
|------|------|------|
| 88D5 | 0x70 | 1200 |
| | 0x70 | 1400 |
| | 0x68 | 1600 |
| | 0x68 | 1800 |
| | 0x60 | 2000 |
| | 0x60 | 2200 |
| | 0x58 | 2400 |
| | 0x50 | 2800 |
| | 0x50 | 3200 |
| | 0x50 | 3600 |
| | 0x40 | 4000 |

FATI DECAY DELAY MULTIPLIER VS COOLANT TEMP

| | | |
|------|------|-------------------------|
| 88E0 | 0x08 | TABLE LENGTH (9 VALUES) |
| | 0x68 | |
| | 0x5C | |
| | 0x2C | |
| | 0x1D | |
| | 0x05 | |
| | 0x05 | |
| | 0x2C | |
| | 0x2C | |
| | 0x2C | |

88EA 0x03 ;FATI/FATC FILTER COEFF, TBL LKUP -> L002C

FATI DECAY RATE VS STARTUP COOLANT

LARGER NUMBER = SLOWER DECAY

| | |
|------|------|
| 88EB | 0xF8 |
| | 0xF8 |
| | 0xF8 |
| | 0xF8 |
| | 0xF8 |
| | 0xF4 |
| | 0xF0 |
| | 0xF0 |
| | 0xF0 |
| | 0xF0 |
| | 0xEE |
| | 0xEE |
| | 0xEE |
| | 0xEE |
| | 0xEE |

TIME DELAY BEFORE DECAYING FATI VS CTS

| | |
|------|------|
| 88FC | 0x28 |
| | 0x23 |
| | 0x14 |
| | 0x0F |
| | 0x0F |
| | 0x01 |
| | 0x01 |

? NOT USED

8903 0x19,0x19,0x14,0x14,0x14,0x14,0x14

| | | | | |
|------|------|-----|--------------|--|
| 890A | 0xD1 | 209 | 116.75 DEG C | ;COOLANT TEMP |
| 890B | 0xD4 | 212 | 119 DEG C | ;COOLANT TEMP |
| 890C | 0x08 | 8 | 8 MPH | ;SPEED LIMIT |
| 890D | 0x0A | 10 | 10 MPH | ;SPEED LIMIT |
| 890E | 0x5A | 90 | 90 COUNTS | ;LV8 |
| 890F | 0x01 | 1 | | ;TIME LIMIT BEFORE SETTING HOT OPEN LOOP |
| 8910 | 0x00 | 00 | | ;? OPTION WORD |
| 8911 | 0xA0 | 160 | | ;FAR MULTILPIER |
| 8912 | 0x9C | 156 | | ;HOT OPEN LOOP FAR |
| 8913 | 0x93 | 147 | 70.25 DEG C | ;COOLANT TEMP |
| 8914 | 0x32 | 50 | | ;MAF |
| 8915 | 0x23 | 35 | | ;MAF |
| 8916 | 0x4B | 75 | | ; |
| 8917 | 0x64 | 100 | | ; |
| 8918 | 0x46 | 70 | | ;MAF |
| 8919 | 0x23 | 35 | | ;MAF |
| 891A | 0x4B | 75 | | ; |
| 891B | 0x95 | 149 | | ;FAR MULTIPLIER WHEN B0 L0099 SET |
| 891C | 0x08 | 8 | 3.1 %TPS | ;TPS HYSTERESIS (FROM 891D) TO DISABLE PE MODE |

PE %TPS THRESHOLD VS NLRPMX

| | HEX | DEC | VALUE | RPM |
|-------|------|-----|---------|------|
| ----- | | | | |
| 891D | 0x73 | 115 | 45 %TPS | 400 |
| | 0x73 | 115 | 45 %TPS | 600 |
| | 0x73 | 115 | 45 %TPS | 800 |
| | 0x73 | 115 | 45 %TPS | 1000 |
| | 0x73 | 115 | 45 %TPS | 1200 |
| | 0x73 | 115 | 45 %TPS | 1400 |
| | 0x73 | 115 | 45 %TPS | 1600 |
| | 0x73 | 115 | 45 %TPS | 1800 |
| | 0x73 | 115 | 45 %TPS | 2000 |
| | 0x73 | 115 | 45 %TPS | 2200 |
| | 0x73 | 115 | 45 %TPS | 2400 |
| | 0x73 | 115 | 45 %TPS | 2800 |
| | 0x73 | 115 | 45 %TPS | 3200 |
| | 0x73 | 115 | 45 %TPS | 3600 |
| | 0x73 | 115 | 45 %TPS | 4000 |
| | 0x73 | 115 | 45 %TPS | 4400 |
| 892D | 0x73 | 115 | 45 %TPS | 4800 |

PE AFR MULTIPLIER (0 - 2) VS COOLANT TEMP

| | HEX | DEC | VALUE | CLNT TEMP (DEG C) |
|-------|------|-------------------------|-------|-------------------|
| ----- | | | | |
| 892E | 0x08 | TABLE LENGTH (9 VALUES) | | |
| | 0xFF | 255 | 2.0 | -40 |
| | 0xC6 | 198 | 1.55 | -16 |
| | 0xB6 | 182 | 1.42 | 8 |
| | 0x94 | 148 | 1.16 | 32 |
| | 0x7E | 126 | 0.98 | 56 |
| | 0x7E | 126 | 0.98 | 80 |
| | 0x7E | 126 | 0.98 | 104 |
| | 0x7E | 126 | 0.98 | 128 |
| | 0x7E | 126 | 0.98 | 156 |

| | | | | |
|------|------|-----|-----------|-----------------------------------|
| 8938 | 0xFF | 255 | 99.8 %TPS | ;DON'T DECAY PE AFR IF TPS > THIS |
| 8939 | 0xA0 | 160 | | ;DON'T DECAY PE AFR IF RPM > THIS |

PE AFR VS TIME IN POWER ENRICHMENT

| | HEX | DEC | VALUE (AFR) | TIME (SEC) |
|-------|-----|-----|-------------|------------|
| ----- | | | | |

| | | | | |
|------|------|-----|-------|------|
| 893A | 0x80 | 128 | 12.8 | 0 |
| | 0x80 | 128 | 12.8 | 1.6 |
| | 0x80 | 128 | 12.8 | 3.2 |
| | 0x80 | 128 | 12.8 | 4.8 |
| | 0x80 | 128 | 12.8 | 6.4 |
| | 0x80 | 128 | 12.8 | 8.0 |
| | 0x80 | 128 | 12.8 | 9.6 |
| | 0x80 | 128 | 12.8 | 11.2 |
| | 0x80 | 128 | 12.8 | 12.8 |
| | 0x80 | 128 | 12.8 | 14.4 |
| | 0x80 | 128 | 12.8 | 16.0 |
| | 0x90 | 144 | 11.37 | 17.6 |
| | 0x90 | 144 | 11.37 | 19.2 |
| | 0x90 | 144 | 11.37 | 20.8 |
| | 0x90 | 144 | 11.37 | 22.4 |
| | 0x90 | 144 | 11.37 | 24.0 |
| | 0x90 | 144 | 11.37 | 25.6 |

| PE FAR MULTIPLIER (0 - 2) VS RPM/25 | | | |
|-------------------------------------|-----|-----|--|
| | HEX | DEC | |

| | | | MULTIPLIER | RPM |
|------|------|-----|------------|------|
| 894B | 0x85 | 133 | 1.04 | 400 |
| | 0x85 | 133 | 1.04 | 800 |
| | 0x83 | 131 | 1.02 | 1200 |
| | 0x83 | 131 | 1.02 | 1600 |
| | 0x85 | 133 | 1.04 | 2000 |
| | 0x87 | 135 | 1.05 | 2400 |
| | 0x86 | 134 | 1.046 | 2800 |
| | 0x86 | 134 | 1.046 | 3200 |
| | 0x87 | 135 | 1.05 | 3600 |
| | 0x86 | 134 | 1.046 | 4000 |
| | 0x89 | 137 | 1.07 | 4400 |
| | 0x8B | 139 | 1.085 | 4800 |
| | 0x86 | 134 | 1.046 | 5200 |
| | 0x86 | 134 | 1.046 | 5600 |
| | 0x86 | 134 | 1.046 | 6000 |
| | 0x86 | 134 | 1.046 | 6400 |

| | | | | | |
|------|-----------|-----|------------|--------|--|
| 895B | 0x00,0x00 | 00 | | ; | ? TIME DELAY BEFORE POWER ENRICHMENT ENABLED |
| 895D | 0x88 | | | ; | O2 SENSOR READY HIGH THRESHOLD |
| 895E | 0x44 | | | ; | O2 SENSOR READY LOW THRESHOLD |
| 895F | 0x85 | | | | |
| 8960 | 0x00,0x10 | 16 | | | |
| 8962 | 0x00,0x32 | 50 | | | |
| 8964 | 0x5A | 90 | 27.5 DEG C | ; | CLNT TEMP THRESHOLD FOR CLOSED LOOP |
| 8965 | 0x32 | 50 | 5 SECONDS | ; | O2 SENSOR READY TIME LIMIT |
| 8966 | 0x48 | 72 | 1800 RPM | ; | |
| 8967 | 0x05 | 5 | | ; | |
| 8968 | 0x90 | 144 | 68 DEG C | ; | |
| 8969 | 0x6F | 111 | | ; | KCLRATIO - STOICHIOMETRIC AFR |
| 896A | 0x10 | 16 | | ; | |
| 896B | 0x10 | 16 | | ; | |
| 896C | 0x66 | 102 | | ; | GAIN FACTOR FOR INTDLY |
| 896D | 0x50 | 80 | | ; | GAIN FACTOR FOR INTDLY ? AT IDLE |
| 896E | 0x00,0x00 | 00 | | ; | TIME SINCE RUN ENABLED |
| 8970 | 0x91 | 145 | 641 mV ; | | |
| 8971 | 0x3D | 61 | 270 mV ; | | |
| 8972 | 0x91 | 145 | 641 mV ; | | |
| 8973 | 0x42 | 66 | 291 mV ; | | |
| 8974 | 0x71 | 113 | 501 mV ; | ZEREFU | |

| | | | |
|------|------|-----|--|
| 8975 | 0x5C | 92 | 406 mV ;ZEREFL |
| 8976 | 0x80 | 128 | 566 mV ;ZEREFU |
| 8977 | 0x48 | 72 | 318 mV ;ZEREFL |
| 8978 | 0x03 | 3 | ;INTEGRATOR DECREMENT RATE |
| 8979 | 0x03 | 3 | ;INTEGRATOR INCREMENT RATE |
| 897A | 0x03 | 3 | ;INTEGRATOR DECREMENT RATE |
| 897B | 0x03 | 3 | ;INTEGRATOR INCREMENT RATE |
| 897C | 0x03 | 3 | ;INTEGRATOR DECREMENT RATE |
| 897D | 0x03 | 3 | ;INTEGRATOR INCREMENT RATE |
| 897E | 0x03 | 3 | ;INTEGRATOR DECREMENT RATE |
| 897F | 0x03 | 3 | ;INTEGRATOR INCREMENT RATE |
| 8980 | 0x64 | 100 | ;MIN INTEGRATOR |
| 8981 | 0xB6 | 182 | ;MAX INTEGRATOR |
| 8982 | 0x72 | 114 | ;IF INT > THIS, DON'T RESET INT ON BLM CELL CHANGE |

INT UPDATE DELAY VS MAF - RESULT GOES INTO L00B2;
 INT UPDATE DELAY/4 -> L00B1 (BLM UPDATE DELAY TIME)

| | HEX | DEC | VALUE | MAF |
|------|------|-----|-------|-----|
| 8983 | 0x40 | | | 0 |
| | 0x40 | | | 16 |
| | 0x28 | | | 32 |
| | 0x18 | | | 48 |
| | 0x14 | | | 64 |
| | 0x10 | | | 80 |
| | 0x0C | | | 96 |
| | 0x08 | | | 112 |
| | 0x08 | | | 128 |

FILTER COEFFICIENTS FOR O2 A/D COUNTS VS MAF

| | HEX | DEC | VALUE | MAF |
|------|-----------|-----|-------|---|
| 898C | 0x01 | | | |
| | 0x01 | | | |
| | 0x01 | | | |
| | 0x01 | | | |
| | 0x01 | | | |
| | 0x01 | | | |
| | 0x01 | | | |
| | 0x01 | | | |
| 8994 | 0x01 | | | ; |
| 8995 | 0xFF | | | ;FILTER COEFF FOR FILTERED O2 A/D -> L010E |
| 8996 | 0x74,0xBB | | | ;O2 FILTER NOM VALUE WHEN RESETING INTEGRATOR |
| 8998 | 0xFF | | | ;? OPTION FLAG |

O2 LEAN THRESHOLD VS LV8 OR BLM CELL

| | HEX | DEC | VALUE | LV8 | BLMCELL |
|------|------|-----|------------|-----|---------|
| 8999 | 0x44 | 68 | 300 mV 0 | 0 | |
| | 0x44 | 68 | 300 mV 16 | 1 | |
| | 0x44 | 68 | 300 mV 32 | 2 | |
| | 0x44 | 68 | 300 mV 48 | 3 | |
| | 0x44 | 68 | 300 mV 64 | 4 | |
| | 0x44 | 68 | 300 mV 80 | 5 | |
| | 0x44 | 68 | 300 mV 96 | 6 | |
| | 0x44 | 68 | 300 mV 112 | 7 | |
| | 0x44 | 68 | 300 mV 128 | 8 | |
| | 0x44 | 68 | 300 mV 144 | 9 | |
| | 0x44 | 68 | 300 mV 160 | 10 | |
| | 0x44 | 68 | 300 mV 176 | 11 | |
| | 0x44 | 68 | 300 mV 192 | 12 | |
| | 0x44 | 68 | 300 mV 208 | 13 | |

| | | | |
|------|----|------------|----|
| 0x44 | 68 | 300 mV 224 | 14 |
| 0x44 | 68 | 300 mV 240 | 15 |
| 0x44 | 68 | 300 mV 256 | 16 |

O2 VOLTS VS LV8 OR BLM CELL

| | HEX | DEC | VALUE | LV8 | BLMCELL |
|------|------|-----|------------|-----|---------|
| 89AA | 0x88 | 136 | 603 mV 0 | 0 | |
| | 0x88 | 136 | 603 mV 16 | 1 | |
| | 0x88 | 136 | 603 mV 32 | 2 | |
| | 0x88 | 136 | 603 mV 48 | 3 | |
| | 0x88 | 136 | 603 mV 64 | 4 | |
| | 0x88 | 136 | 603 mV 80 | 5 | |
| | 0x88 | 136 | 603 mV 96 | 6 | |
| | 0x88 | 136 | 603 mV 112 | 7 | |
| | 0x88 | 136 | 603 mV 128 | 8 | |
| | 0x88 | 136 | 603 mV 144 | 9 | |
| | 0x88 | 136 | 603 mV 160 | 10 | |
| | 0x88 | 136 | 603 mV 176 | 11 | |
| | 0x88 | 136 | 603 mV 192 | 12 | |
| | 0x88 | 136 | 603 mV 208 | 13 | |
| | 0x88 | 136 | 603 mV 224 | 14 | |
| | 0x88 | 136 | 603 mV 240 | 15 | |
| | 0x88 | 136 | 603 mV 256 | 16 | |

O2 VOLTS VS LV8 OR BLM CELL DEPENDING ON 8998 OPTION FLAG

| | HEX | DEC | VALUE | LV8 | BLM CELL |
|------|------|-----|------------|-----|----------|
| 89BB | 0x65 | 101 | 448 mV 0 | 0 | |
| | 0x65 | 101 | 448 mV 16 | 1 | |
| | 0x65 | 101 | 448 mV 32 | 2 | |
| | 0x65 | 101 | 448 mV 48 | 3 | |
| | 0x65 | 101 | 448 mV 64 | 4 | |
| | 0x65 | 101 | 448 mV 80 | 5 | |
| | 0x65 | 101 | 448 mV 96 | 6 | |
| | 0x65 | 101 | 448 mV 112 | 7 | |
| | 0x63 | 99 | 439 mV 128 | 8 | |
| | 0x63 | 99 | 439 mV 144 | 9 | |
| | 0x83 | 131 | 582 mV 160 | 10 | |
| | 0x83 | 131 | 582 mV 176 | 11 | |
| | 0x83 | 131 | 582 mV 192 | 12 | |
| | 0x83 | 131 | 582 mV 208 | 13 | |
| | 0x83 | 131 | 582 mV 224 | 14 | |
| | 0x83 | 131 | 582 mV 240 | 15 | |
| | 0x83 | 131 | 582 mV 256 | 16 | |

O2 VOLTS VS LV8 OR BLM CELL DEPENDING ON 8998 OPTION FLAG

| | HEX | DEC | VALUE | LV8 | BLMCELL |
|------|------|-----|------------|-----|---------|
| 89CC | 0x71 | 113 | 501 mV 0 | 0 | |
| | 0x71 | 113 | 501 mV 16 | 1 | |
| | 0x71 | 113 | 501 mV 32 | 2 | |
| | 0x71 | 113 | 501 mV 48 | 3 | |
| | 0x71 | 113 | 501 mV 64 | 4 | |
| | 0x71 | 113 | 501 mV 80 | 5 | |
| | 0x71 | 113 | 501 mV 96 | 6 | |
| | 0x71 | 113 | 501 mV 112 | 7 | |
| | 0x71 | 113 | 501 mV 128 | 8 | |
| | 0x71 | 113 | 501 mV 144 | 9 | |
| | 0x83 | 131 | 582 mV 160 | 10 | |
| | 0x83 | 131 | 582 mV 176 | 11 | |
| | 0x83 | 131 | 582 mV 192 | 12 | |
| | 0x83 | 131 | 582 mV 208 | 13 | |
| | 0x83 | 131 | 582 mV 224 | 14 | |

| | | | |
|------|-----|------------|----|
| 0x83 | 131 | 582 mV 240 | 15 |
| 0x83 | 131 | 582 mV 256 | 16 |

| PROPORTIONAL | HEX | STEPS VS DEC | BLM CELL VALUE | BLMCELL |
|--------------|------|--------------|----------------|---------|
| ----- | | | | |
| 89DD | 0x05 | 5 | 5 CNTS 0 | |
| | 0x06 | 6 | 6 CNTS 1 | |
| | 0x06 | 6 | 6 CNTS 2 | |
| | 0x06 | 6 | 6 CNTS 3 | |
| | 0x06 | 6 | 6 CNTS 4 | |
| | 0x05 | 5 | 5 CNTS 5 | |
| | 0x05 | 5 | 5 CNTS 6 | |
| | 0x05 | 5 | 5 CNTS 7 | |
| | 0x05 | 5 | 5 CNTS 8 | |
| | 0x05 | 5 | 5 CNTS 9 | |
| | 0x05 | 5 | 5 CNTS 10 | |
| | 0x05 | 5 | 5 CNTS 11 | |
| | 0x05 | 5 | 5 CNTS 12 | |
| | 0x05 | 5 | 5 CNTS 13 | |
| | 0x05 | 5 | 5 CNTS 14 | |
| | 0x05 | 5 | 5 CNTS 15 | |
| | 0x05 | 5 | 5 CNTS 16 | |

89EE 0x03 ;PROPORTIONAL COUNTS (IDLE)

| PROPORTIONAL | HEX | STEPS VS DEC | BLM CELL WHEN TCC LOCKED VALUE | BLMCELL |
|--------------|------|--------------|--------------------------------|---------|
| ----- | | | | |
| 89EF | 0x05 | 5 | 5 CNTS 0 | |
| | 0x05 | 5 | 5 CNTS 1 | |
| | 0x08 | 8 | 8 CNTS 2 | |
| | 0x08 | 8 | 8 CNTS 3 | |
| | 0x08 | 8 | 8 CNTS 4 | |
| | 0x08 | 8 | 8 CNTS 5 | |
| | 0x05 | 5 | 5 CNTS 6 | |
| | 0x05 | 5 | 5 CNTS 7 | |
| | 0x05 | 5 | 5 CNTS 8 | |
| | 0x05 | 5 | 5 CNTS 9 | |
| | 0x05 | 5 | 5 CNTS 10 | |
| | 0x05 | 5 | 5 CNTS 11 | |
| | 0x05 | 5 | 5 CNTS 12 | |
| | 0x05 | 5 | 5 CNTS 13 | |
| | 0x05 | 5 | 5 CNTS 14 | |
| | 0x05 | 5 | 5 CNTS 15 | |
| | 0x05 | 5 | 5 CNTS 16 | |

8A00 0x71 113 45 DEG C ;

| PROPORTIONAL | HEX | STEPS VS DEC | BLM CELL WHEN COOLANT < 8A00 VALUE | BLMCELL |
|--------------|------|--------------|------------------------------------|---------|
| ----- | | | | |
| 8A01 | 0x05 | 5 | 5 CNTS 0 | |
| | 0x05 | 5 | 5 CNTS 1 | |
| | 0x09 | 9 | 9 CNTS 2 | |
| | 0x09 | 9 | 9 CNTS 3 | |
| | 0x09 | 9 | 9 CNTS 4 | |
| | 0x09 | 9 | 9 CNTS 5 | |
| | 0x09 | 9 | 9 CNTS 6 | |
| | 0x09 | 9 | 9 CNTS 7 | |
| | 0x09 | 9 | 9 CNTS 8 | |

| | | | |
|------|-----------|--------|----------|
| 0x09 | 9 | 9 CNTS | 9 |
| 0x09 | 9 | 9 CNTS | 10 |
| 0x09 | 9 | 9 CNTS | 11 |
| 0x09 | 9 | 9 CNTS | 12 |
| 0x09 | 9 | 9 CNTS | 13 |
| 0x09 | 9 | 9 CNTS | 14 |
| 0x09 | 9 | 9 CNTS | 15 |
| 0x03 | 3 | 3 CNTS | 16 |
| | | | |
| 8A12 | 0x30 | 48 | |
| 8A13 | 0x30 | 48 | |
| 8A14 | 0x4C | 76 | 1900 RPM |
| 8A15 | 0x50 | 80 | 2000 RPM |
| 8A16 | 0x01 | 1 | 1 MPH |
| 8A17 | 0x03 | 3 | 3 MPH |
| 8A18 | 0x03 | 3 | 1.2 %TPS |
| 8A19 | 0x04 | 4 | 1.5 %TPS |
| 8A1A | 0x18 | | |
| 8A1B | 0x19 | | |
| 8A1C | 0x50 | | |
| 8A1D | 0xDC | | |
| 8A1E | 0x19 | | |
| 8A1F | 0x0B | 11 | |
| 8A20 | 0x24 | 36 | 900 RPM |
| 8A21 | 0x34 | 52 | 1300 RPM |
| 8A22 | 0x5F | 95 | 2375 RPM |
| 8A23 | 0x0C,0x00 | 3072 | 12 GM/S |
| 8A25 | 0x13,0x00 | 4864 | 19 GM/S |
| 8A27 | 0xFE,0x00 | 65024 | 254 GM/S |

BLM CELLS:

16 = IDLE

| | | | | | |
|------|-------|-------|-------|-------|----------|
| | 12 | 13 | 14 | 15 | |
| 8A27 | ----- | ----- | ----- | ----- | 254 GM/S |
| | 8 | 9 | 10 | 11 | |
| 8A25 | ----- | ----- | ----- | ----- | 19 GM/S |
| | 4 | 5 | 6 | 7 | |
| 8A23 | ----- | ----- | ----- | ----- | 12 GM/S |
| | 0 | 1 | 2 | 3 | |
| | | | | | |
| | 8A20 | 8A21 | 8A22 | | |
| RPM | 900 | 1300 | 2375 | | |

| | | | | |
|------|-----------|-----|-----------|-----------------------------------|
| 8A29 | 0x05 | 5 | 125 RPM | ;RPM EDGE TO EDGE HYSTERESIS |
| 8A2A | 0x01,0x00 | 256 | 1 GM/SEC | ;AIRFLOW EDGE TO EDGE HYSTERESIS |
| 8A2C | 0x05 | 5 | | ;DELTA BLM AND 128 FOR BLM >= 128 |
| 8A2D | 0x05 | 5 | | ;DELTA BLM AND 128 FOR BLM < 128 |
| 8A2E | 0x01 | 1 | | ;BLM (HYSTERESIS) |
| 8A2F | 0x96 | 150 | | ;MAX BLM VALUE |
| 8A30 | 0xA0 | 160 | | ;MAX BLM VALUE IF MAF MALF SET |
| 8A31 | 0x64 | 100 | | ;MIN BLM VALUE |
| 8A32 | 0x5A | 90 | | ;MIN BLM VALUE IF MAF MALF SET |
| 8A33 | 0x64 | 100 | | ;MIN BLM VALUE IF CCP DC NOT ZERO |
| 8A34 | 0x4B | 75 | | ; |
| 8A35 | 0x05 | 5 | | ; |
| 8A36 | 0xC8 | 200 | 110 DEG C | ;STARTUP TEMP TO RESET BLMS |

2D TABLE VS COOLANT TEMP

| | | | | |
|------|------|--|--|-------------------------|
| 8A37 | 0x0A | | | ;BLM FILTER COEFFICIENT |
|------|------|--|--|-------------------------|

BLMS ARE SET TO THESE VALUES WHEN STARTUP TEMP LESS THAN 8A36

| | HEX | DEC/BLM | BLM CELL |
|-------|------|---------|----------|
| ----- | | | |
| 8A38 | 0x78 | 120 | 0 |
| | 0x78 | 120 | 1 |
| | 0x71 | 113 | 2 |
| | 0x71 | 113 | 3 |
| | 0x71 | 113 | 4 |
| | 0x78 | 120 | 5 |
| | 0x78 | 120 | 6 |
| | 0x71 | 113 | 7 |
| | 0x71 | 113 | 8 |
| | 0x71 | 113 | 9 |
| | 0x78 | 120 | 10 |
| | 0x78 | 120 | 11 |
| | 0x71 | 113 | 12 |
| | 0x71 | 113 | 13 |
| | 0x71 | 113 | 14 |
| | 0x71 | 113 | 15 |
| | 0x78 | 120 | 16 |

BLM CELL POINTERS VS BLM CELL ??

| | | | |
|-------|------|--|--|
| ----- | | | |
| 8A49 | 0x00 | | |
| | 0x01 | | |
| | 0x02 | | |
| | 0x03 | | |
| | 0x04 | | |
| | 0x05 | | |
| | 0x06 | | |
| | 0x07 | | |
| | 0x08 | | |
| | 0x0A | | |
| | 0x0A | | |
| | 0x0B | | |
| | 0x0C | | |
| | 0x0D | | |
| | 0x0E | | |
| | 0x0F | | |
| | 0x10 | | |

8A5A 0x10 16 16 MPH ;THRESHOLD TO DISABLE P/N -> IN GEAR ENRICHMENT

P/N -> GEAR FUEL ENRICHMENT
 ENABLE DELAY TIME VS CLNT TEMP

| | | | |
|-------|------|--|--|
| ----- | | | |
| 8A5B | 0x38 | | |
| | 0x30 | | |
| | 0x28 | | |
| | 0x14 | | |
| | 0x14 | | |
| | 0x14 | | |
| | 0x14 | | |
| | 0x14 | | |
| | 0x14 | | |

P/N -> IN GEAR FUEL ENRICHMENT VS CLNT TEMP
 USED WHEN MPH >= 8A5A

| | | | |
|-------|------|--|--|
| ----- | | | |
| 8A64 | 0x00 | | |
| | 0x0C | | |
| | 0x33 | | |
| | 0x33 | | |
| | 0x1C | | |
| | 0x1C | | |

STALL SAVER ASYNC FACTOR VS RUN TIME LSB

```

-----
8AA1  0x00
      0x00
      0x80
      0x80
      0x80
      0x80
      0x80
      0x80
      0x80

```

```

8AAA  0x02          2          ;MULTIPLIER FOR ASYNC FACTOR VS CLNT TEMP

```

STALL SAVER ASYNC FACTOR VS COOLANT TEMP | | HEX | DEC | FACTOR CLNT TEMP | |--|-----|-----|------------------| |--|-----|-----|------------------|

```

-----
8AAB  0x60
      0x60
      0x60
      0x60
      0x60
      0x40
      0x20
      0x20
      0x20
      0x20
      0x20
      0x3A
      0x3A
      0x20
      0x20
      0x20
      0x20

```

STALL SAVER ASYNC FACTOR VS DELTA TPS

```

-----
8ABC  0x40
      0xA0
      0xE0
      0xF0
      0xF0
      0xF0
      0xF0
      0xF0
      0xF0
      0xF0
      0xF0
      0xF0
      0xF0
      0xF0
      0xF0
      0xF0
      0xF0

```

; ACCELERATION ENRICHMENT PARAMETERS

| | | | | | |
|------|------|----|----------|--|---|
| 8ACD | 0x10 | | | | ;FILTER COEFF FOR FILTERED %TPS L0155 |
| 8ACE | 0x04 | 4 | 1.5 %TPS | | ;POSITIVE DELTA TPS TO ENABLE ACCEL ENRICH |
| 8ACF | 0x04 | 4 | 1.5 %TPS | | ;NEGATIVE DELTA TPS TO DISABLE ACCEL ENRICH |
| 8AD0 | 0x00 | 0 | 0 | | ;MIN AE TIME (RPM) VS XIRQ PULSES |
| 8AD1 | 0x50 | 80 | | | ;MAX AE TIME (RPM) VS XIRQ PULSES |
| 8AD2 | 0x00 | 0 | | | ;DON'T DO INT UPDATE IF AE TIME (XIRQ) < THIS |

ACCEL ENRICHMENT FACTOR VS POSITIVE DELTA-TPS

8AD3 0xFE
0xFE
0xFE
0xFE
0xFE
0xFE
0xFE
0xFE

ACCEL ENRICHMENT MULTIPLIER VS CTS2 - RESULT GOES INTO L015C
HEX DEC FACTOR CLNT TEMP (DEG C)

8ADC 0x08 TABLE LENGTH (9 VALUES)
0xC0 -40
0xC0 -16
0xA0 8
0x95 32
0x80 56
0x18 80
0x10 104
0x18 128
0x18 156

ACCEL ENRICHMENT MULTIPLIER VS CTS2 WHEN P/N - RESULT GOES INTO L015C
HEX DEC MULTIPLIER CLNT TEMP (DEG C)

8AE6 0x08 TABLE LENGTH (9 VALUES)
0xC0 -40
0xC0 -16
0xF0 8
0xC0 32
0x78 56
0x20 80
0x18 104
0x18 128
0x18 156

ACCEL ENRICHMENT MULTIPLIER VS IATMAT - RESULT GOES INTO L015D
HEX DEC MULTIPLIER IAT (DEG C)

8AF0 0x80
0x80
0x80
0x80
0x80
0x80
0x80
0x80
0x80

ACCEL ENRICHMENT MULTIPLIER VS FILTERED MPH
HEX DEC MULTIPLIER MPH

8AF9 0x80
0x6E
0x56
0x4A
0x38
0x2C
0x14
0x00

0x00
0x00
0x00
0x00
0x00
0x00
0x00
0x00
0x00

ACCEL ENRICHMENT MULTIPLIER VS A.E. TIME
HEX DEC MULTIPLIER TIME

8B0A 0xFF
0xFF
0xC0
0x80
0x70
0x5B
0x4E
0x42
0x37
0x2D
0x24
0x1C
0x15
0x0F
0x0A
0x06
0x03
0x01
0x00
0x00
0x00
0x00
0x00
0x00
0x00
0x00
0x00
0x00
0x00
0x00
0x00
0x00
0x00

A.E. MULTIPLIER VS RUN TIME
HEX DEC MULTIPLIER TIME

8B2B 0x80
0x80
0x80
0x80
0x80
0x80
0x80
0x80
0x80

8B34 0xFF,0xFF ;
8B36 0x10 ;FILTER COEFF FOR FILTERED %TPS L0157
;IF ZERO, L0157 = L00AA:L8B36
8B37 0x08 8 3.1 %TPS ;NEG D-TPS% TO ENABLE DE

| | | | | |
|------|------|---|----------|--|
| 8B38 | 0x03 | 3 | 1.2 %TPS | ;POS D-TPS% TO DISABLE DE |
| 8B39 | 0x06 | 6 | | ;MIN TIME CONDITIONS MET BEFORE DE TERM USED |
| 8B3A | 0xFE | | | ;MAXIMUM TIME IN DECEL ENLEANMENT |
| 8B3B | 0x5A | | | ;DON'T UPDATE INT IF DE TIME LESS THAN THIS |

DECEL ENLEANMENT TERM VS NEG D-TPS

| | HEX | DEC | FACTOR | D-TPS |
|------|------|-----|--------|-------|
| 8B3C | 0x18 | | | |
| | 0x18 | | | |
| | 0x20 | | | |
| | 0x28 | | | |
| | 0x28 | | | |
| | 0x28 | | | |
| | 0x28 | | | |
| | 0x28 | | | |
| | 0x28 | | | |

DE TERM MULTIPLIER VS DECEL ENLEANMENT TIME

| | HEX | DEC | MULTIPLIER | TIME |
|------|------|-----|------------|------|
| 8B45 | 0xFF | | | |
| | 0xC0 | | | |
| | 0x80 | | | |
| | 0x50 | | | |
| | 0x20 | | | |
| | 0x00 | | | |
| | 0x00 | | | |
| | 0x00 | | | |
| | 0x00 | | | |
| | 0x00 | | | |
| | 0x00 | | | |
| | 0x00 | | | |
| | 0x00 | | | |
| | 0x00 | | | |
| | 0x00 | | | |
| | 0x00 | | | |

| | | | | |
|------|-----------|-----|---------------|----------------------------------|
| 8B56 | 0x80 | 128 | 3200 RPM | ; |
| 8B57 | 0x28 | 40 | 1000 RPM | ; |
| 8B58 | 0x3C | 60 | 1500 RPM | ; |
| 8B59 | 0x2A | 42 | 1050 RPM | ; |
| 8B5A | 0x23 | 35 | 35 COUNTS | ;LV8 |
| 8B5B | 0x46 | 70 | 70 COUNTS | ;LV8 |
| 8B5C | 0x06 | 6 | 75 RPM ;DELTA | RPM/12.5 |
| 8B5D | 0x01,0x00 | 256 | 256 | ;TIME DELAY BEGORE ALLOWING DFCO |
| 8B5F | 0x06 | 6 | 2.35% TPS | ; |
| 8B60 | 0x8C | 140 | 65 DEG C | ;COOLANT TEMP |
| 8B61 | 0x19 | 25 | 25 MPH | ; |

THESE ARE RELATED TO OPTION WORD 8010 - RELATED TO REV LIMIT AND TRANNY

| | | | | |
|------|------|----|-----------|-------------|
| 8B62 | 0x00 | 00 | 00 %TPS | ; |
| 8B63 | 0x00 | 00 | 00 MPH | ; |
| 8B64 | 0x00 | 00 | 00 COUNTS | ;LV8 COUNTS |
| 8B65 | 0x00 | 00 | 00 RPM | ; |
| 8B66 | 0x00 | 00 | 00 COUNTS | ;LV8 COUNTS |
| 8B67 | 0x00 | 00 | 00 RPM | ; |

REV LIMIT CALIBRATIONS

| | | | | |
|------|-----------|-------|----------|-------------------|
| 8B68 | 0x19,0x64 | 6500d | 6500 RPM | ;FUEL CUT |
| 8B6A | 0x18,0x9C | 6300d | 6300 RPM | ;FUEL RESTORE |
| 8B6C | 0x19,0x64 | 6500d | 6500 RPM | ;FUEL CUT NEUTRAL |

| | | | | |
|------|-----------|-------|----------|-----------------------|
| 8B6E | 0x18,0x9C | 6300d | 6300 RPM | ;FUEL RESTORE NEUTRAL |
| 8B70 | 0x0F,0xA0 | 4000d | 4000 RPM | ;FUEL CUT PARK |
| 8B72 | 0x0F,0x9F | 3999d | 3999 RPM | ;FUEL RESTORE PARK |
| 8B74 | 0x0E,0x0D | 3597d | 3597 RPM | ;FUEL CUT REVERSE |
| 8B76 | 0x0E,0x0C | 3596d | 3596 RPM | ;FUEL RESTORE REVERSE |

OVERSPEED CALIBRATIONS

| | | | | |
|------|-----------|------|----------|--|
| 8B78 | 0x76 | | | ;FUEL CUT |
| 8B79 | 0x73 | | | ;FUEL RESTORE |
| 8B7A | 0x76 | | | ;FUEL CUT |
| 8B7B | 0x73 | | | ;FUEL RESTORE |
| 8B7C | 0x76 | | | ;FUEL CUT |
| 8B7D | 0x73 | | | ;FUEL RESTORE |
| 8B7E | 0x0B,0xB8 | 3000 | 3000 RPM | ;RPM MUST BE GREATER THAN THIS TO ACTIVATE ; OVERSPEED FUEL CUT |

? LEAN CRUISE FAR CALS ?

| | | | | |
|------|-----------|------|-----------------|--|
| 8B80 | 0x76 | 118 | 118 MPH | ;MPH THRESHOLD, 5 TH GEAR |
| 8B81 | 0x73 | 115 | 115 MPH | ;MPH THRESHOLD, 4 TH GEAR |
| 8B82 | 0x71 | 113 | 113 MPH | ;MPH THRESHOLD, 2 ND OR 3 RD GEAR |
| 8B83 | 0x02 | 2 | | ;MULTIPLIER FOR DECREASING AFR WHEN OVER THRESH ; DIFF BETWN THRESHOLD AND MPH * 8B83+DIFF IS ; MULTIPLIER FOR AFR REDUCTION |
| 8B84 | 0x0A | | | ;MULTIPLIER FOR SPARK RETARD WHEN ABOVE ; SPEED THERSHOLD |
| 8B85 | 0x1E,0xD8 | 7896 | | ;FUEL INJECTOR CONSTANT |
| 8B87 | 0x00,0x42 | 66 | 1 MSEC ;MIN BPW | |
| 8B89 | 0x00,0x62 | 98 | 1.5 MSEC | ;DEFAULT PW FOR LOW PW |

FUEL INJECTOR OFFSET VS BATTERY VOLTAGE

| | HEX | DEC | MSEC | BATT VOLTS |
|------|------|-----|-----------|-------------------------|
| 8B8B | 0x10 | | | TABLE LENGTH (17 VALUE) |
| | 0xFF | 255 | 7.78 MSEC | 0.0 VOLTS |
| | 0xFF | 255 | 7.78 | 1.6 |
| | 0xFF | 255 | 7.78 | 3.2 |
| | 0xFF | 255 | 7.78 | 4.8 |
| | 0xEE | 238 | 7.26 | 6.4 |
| | 0x80 | 128 | 3.90 | 8.0 |
| | 0x55 | 85 | 2.59 | 9.6 |
| | 0x3C | 60 | 1.83 | 11.2 |
| | 0x2B | 43 | 1.31 | 12.8 |
| | 0x26 | 38 | 1.16 | 14.4 |
| | 0x1F | 31 | 0.95 | 16.0 |
| | 0x16 | 22 | 0.67 | 17.6 |
| | 0x0F | 15 | 0.46 | 19.2 |
| | 0x08 | 8 | 0.24 | 20.8 |
| | 0x01 | 1 | 0.03 | 22.4 |
| | 0x00 | 0 | 0 | 24.0 |
| | 0x00 | 0 | 0 | 25.5 |

FUEL INJ OFFSET VS BPW

| | | |
|------|-----------|----|
| 8B9D | 0x00,0x0E | 14 |
| 8B9F | 0x00,0x0E | 14 |
| 8BA1 | 0x00,0x0E | 14 |
| 8BA3 | 0x00,0x09 | 9 |
| 8BA5 | 0x00,0x05 | 5 |
| 8BA7 | 0x00,0x02 | 2 |
| 8BA9 | 0x00,0x01 | 1 |

| | | | | | |
|------|-----------|-------|-----------|---|------------------------------------|
| 8BAB | 0x00,0x01 | 1 | | | |
| 8BAD | 0x00,0x01 | 1 | | | |
| 8BAF | 0x00,0x00 | 0 | | | |
| 8BB1 | 0x00,0x00 | 0 | | | |
| 8BB3 | 0x00,0x00 | 0 | | | |
| 8BB5 | 0x00,0x00 | 0 | | | |
| 8BB7 | 0x00,0x00 | 0 | | | |
| 8BB9 | 0x00,0x00 | 0 | | | |
| 8BBA | 0x00,0x00 | 0 | | | |
| | | | | | |
| 8BBD | 0x0A | 10 | 10 MPH | ; | |
| 8BBE | 0x0E,0xA6 | 3750 | 3750 RPM | | ; |
| 8BC0 | 0x66 | 102 | 39.8 %TPS | | ; |
| 8BC1 | 0x1E | 30 | 11.7% TPS | | ; |
| 8BC2 | 0x00,0xF0 | | | | ;INITIAL VALUE FOR L0273 |
| 8BC4 | 0xFF,0xFF | 65536 | 65536 RPM | | ; |
| 8BC6 | 0x04,0xB0 | 1200 | 1200 RPM | | ; |
| 8BC8 | 0x03,0xE8 | 1000 | 1000 RPM | | ; |
| 8BCA | 0x03,0xE8 | 1000 | 1000 RPM | | ; |
| 8BCC | 0x04,0xB0 | 1200 | 1200 RPM | | ; |
| 8BCE | 0x04,0xB0 | 1200 | 1200 RPM | | ; |
| 8BD0 | 0x00 | 00 | 00% TPS | | ; |
| 8BD1 | 0x00 | 00 | 00% TPS | | ; |
| 8BD2 | 0x00 | 00 | 00 MPH | ; | |
| 8BD3 | 0x00 | 00 | 00 MPH | ; | |
| 8BD4 | 0x00,0x00 | | | | ; |
| 8BD6 | 0x00,0x00 | | | | ; |
| | | | | | |
| 8BD8 | 0xFF | | | | ;BPW FILTER COEFFICIENT (NOT USED) |
| 8BD9 | 0x00,0x00 | | | | ;MIN VALUE FOR |
| 8BDB | 0x07,0xD0 | 2000 | | | ;MAF OFFSET (SUBTRACTED) |

MAF SENSOR SCALER TABLES

GM/SEC = ADDITIVE OFFSET + (TABLE VALUE * SCALER)

MAF #1

FOR PULSES 2000 - 3024

| | | | | ADDITIVE OFFSET |
|------|--------|-----|------------|-----------------|
| | | | | SCALER |
| 8BDD | 0x02F2 | 754 | 2.958 gm/s | |
| 8BDF | 0x07 | 7 | | |
| 8BE0 | 0x12 | 18 | | |
| | 0x25 | 37 | | |
| | 0x39 | 57 | | |
| | 0x4F | 79 | | |
| | 0x67 | 103 | | |
| | 0x80 | 128 | | |
| | 0x9C | 156 | | |
| | 0xB9 | 185 | | |
| | 0xD9 | 217 | | |

MAF #2

FOR PULSES 3025 - 4047

| | | | | ADDITIVE OFFSET |
|------|--------|------|------------|-----------------|
| | | | | SCALER |
| 8BE9 | 0x0863 | 2147 | 8.392 gm/s | |
| 8BEB | 0x0C | 12 | | |
| 8BEC | 0x0B | 11 | | |
| | 0x1F | 31 | | |
| | 0x34 | 52 | | |
| | 0x4B | 75 | | |
| | 0x63 | 99 | | |
| | 0x7E | 126 | | |
| | 0x9A | 154 | | |
| | 0xB9 | 185 | | |
| | 0xD9 | 217 | | |

MAF #3

FOR PULSES 4048 - 5071

```
-----
8BF5  0x120E      4622  18.055 gm/s  ADDITIVE OFFSET
8BF7  0x11        17                      SCALER
8BF8  0x08         8
      0x20        32
      0x39        57
      0x53        83
      0x6E       110
      0x8B       139
      0xAA       170
      0xCA       202
      0xEC       236
```

MAF #4

FOR PULSES 5072 - 6095

```
-----
8C01  0x2135 8501  33.2099 gm/s  ADDITIVE OFFSET
8C03  0x18         24                      SCALER
8C04  0x05         5
      0x1D        29
      0x38        56
      0x53        83
      0x70       112
      0x8E       142
      0xAC       172
      0xCE       206
      0xF1       241
```

MAF #5

FOR PULSES 6096 - 7119

```
-----
8C0D  0x374F      14159  55.313 gm/s  ADDITIVE OFFSET
8C0F  0x22        34                      SCALER
8C10  0x04         4
      0x1E        30
      0x39        57
      0x57        87
      0x74       116
      0x94       148
      0xB4       180
      0xD5       213
      0xF8       248
```

MAF #6

FOR PULSES 7120 - 8143

```
-----
8C19  0x57BF 22463  87.756 gm/s  ADDITIVE OFFSET
8C1B  0x2D        45                      SCALER
8C1C  0x03         3
      0x1E        30
      0x3A        58
      0x57        87
      0x75       117
      0x94       148
      0xB4       180
      0xD4       212
      0xF6       246
```

MAF #7

FOR PULSES 8144 - 9167

```

-----
8C25  0x827C      33404  130.491 gm/s ADDITIVE OFFSET
8C27  0x37        55                                SCALER
8C28  0x02         2
      0x1F         31
      0x3C         60
      0x5A         90
      0x78        120
      0x95        149
      0xB6        182
      0xD8        216
      0xFB        251

```

MAF #8
FOR PULSES 9168 - 10191

```

-----
8C31  0xB7DD      47069  183.875 gm/s ADDITIVE OFFSET
8C33  0x48        72                                SCALER
8C34  0x02         2
      0x1E         30
      0x3A         58
      0x58         88
      0x7A        118
      0x99        153
      0xB9        185
      0xD9        217
      0xFA        250

```

BASE MAF VS RPM/25 (FOR ENGINE TORQUE)

```

-----
8C3D  0x0E
      0x0E
      0x1B
      0x29
      0x37
      0x44
      0x52
      0x5F
      0x6D
      0x7B
      0x88
      0x96
      0xA4
      0xB1
      0xBF
      0xCC
      0xDA

```

BASE MAF MULTIPLIER VS IATMAT (FOR ENGINE TORQUE)

```

      HEX      DEC      MULTIPLIER      IAT (DEG C)
-----
8C4E  0x04      TABLE LENGTH (5 VALUES)
      0x9D
      0x80
      0x80
      0x6F
      0x64

```

MAX MAF (ADDED TO MEASURED MAF) VS NLRPMX

```

-----
8C54  0x06
      0x06
      0x06

```

2D TABLE VS FILTERED ENGINE TORQUE
FOR IAC (? A/C) STUFF

```

-----
8C8B  0x80
      0x80
      0x80
      0x80
      0x80
      0x80
      0x80
      0x80
      0x80
      0x80
      0x80

```

IAC STEPS VS L0076 (NOT USED)
 ADDED TO L8C8B TABLE LOOKUP RESULT

```

-----
8C96  0x04          TABLE LENGTH (5 VALUES)
      0x00
      0x00
      0x00
      0x00
      0x00

```

```

8C9C  0x08          8          100 RPM      ;ADDED TO DESIRED IDLE WHEN BIT 7 L0090 SET
8C9D  0x08          8          100 RPM      ;HOT START IDLE SPEED OFFSET
8C9E  0x80          128         56 DEG C     ;HOT START IDLE SPEED OFFSET CLNT THRESHOLD
8C9F  0x05          5           5 MPH        ;HOT START IDLE SPEED OFFSET SPEED THERSHOLD
8CA0  0x00,0x0A     10          10 SECONDS   ;HOT START IDLE SPEED OFFSET TIME THRESHOLD
8CA2  0x02          2           25 RPM ;ADDED TO DESIRED IDLE WHEN L02EB IS NOT ZERO
8CA3  0x00,0xF0     240         ;INITIAL VALUE FOR L02EB TIMER
8CA5  0x08          8           ;
8CA6  0x00          0           ;VS FILTERED ENGINE TORQUE
8CA7  0x00          0           ;ADDED TO DESIRED IDLE WHEN TRQ < L8CA6

```

| | HEX | DEC | RPM | COOLANT TEMP |
|------|------|-----|----------|--------------|
| 8CA8 | 0x60 | 96 | 1200 RPM | -40 |
| | 0x5C | 92 | 1150 | -28 |
| | 0x57 | 87 | 1087 | -16 |
| | 0x52 | 82 | 1025 | -4 |
| | 0x50 | 80 | 1000 | 8 |
| | 0x4E | 78 | 975 | 20 |
| | 0x49 | 73 | 913 | 32 |
| | 0x44 | 68 | 850 | 44 |
| | 0x40 | 64 | 800 | 56 |
| | 0x38 | 56 | 700 | 68 |
| | 0x38 | 56 | 700 | 80 |
| | 0x38 | 56 | 700 | 92 |
| | 0x38 | 56 | 700 | 104 |
| | 0x38 | 56 | 700 | 116 |
| | 0x38 | 56 | 700 | 128 |
| | 0x38 | 56 | 700 | 140 |
| | 0x38 | 56 | 700 | 152 |

| | HEX | DEC | RPM | COOLANT TEMP |
|------|------|-----|----------|--------------|
| 8CB9 | 0x70 | 112 | 1400 RPM | -40 |
| | 0x70 | 112 | 1400 | -28 |
| | 0x70 | 112 | 1400 | -16 |
| | 0x68 | 104 | 1300 | -4 |

| | | | |
|------|----|------|-----|
| 0x5F | 95 | 1187 | 8 |
| 0x5F | 95 | 1187 | 20 |
| 0x56 | 86 | 1075 | 32 |
| 0x4A | 74 | 925 | 44 |
| 0x40 | 64 | 800 | 56 |
| 0x3C | 60 | 750 | 68 |
| 0x3C | 60 | 750 | 80 |
| 0x3C | 60 | 750 | 92 |
| 0x3C | 60 | 750 | 104 |
| 0x3C | 60 | 750 | 116 |
| 0x3C | 60 | 750 | 128 |
| 0x3C | 60 | 750 | 140 |
| 0x3C | 60 | 750 | 152 |

| | | | | |
|------|------|-----|-------------|-----------------------------|
| 8CCA | 0x28 | | | ; |
| 8CCB | 0x04 | 4 | | ; |
| 8CCC | 0x1B | | | ;IAT OFFSET FOR L8CD1 TABLE |
| 8CCD | 0x93 | 147 | 70.25 DEG C | ;COOLANT TEMP |
| 8CCE | 0x89 | 137 | 62.75 DEG C | ;COOLANT TEMP |
| 8CCF | 0xA0 | 160 | | ;MAX VALUE FOR L0304 |
| 8CD0 | 0x02 | 2 | | ; |

2D TABLE VS IATMAT -> L0303

| | | | | |
|------|------|--|--|--|
| 8CD1 | 0x02 | | | |
| | 0x02 | | | |
| | 0x02 | | | |
| | 0x02 | | | |
| | 0x02 | | | |
| | 0x02 | | | |
| | 0x02 | | | |
| | 0x02 | | | |
| | 0x02 | | | |
| | 0x02 | | | |
| | 0x02 | | | |
| | 0x02 | | | |
| | 0x02 | | | |
| | 0x02 | | | |
| | 0x02 | | | |
| | 0x02 | | | |

| | | | | |
|------|-----------|-----|----------|--|
| 8CE2 | 0x03 | 3 | 1.2% TPS | ;TPS HYSTERESIS |
| 8CE3 | 0x0C | 12 | | ;DEFAULT MPH FOR DTC P0502 |
| 8CE4 | 0x0F | 15 | | ; |
| 8CE5 | 0x00,0x50 | 80 | | ; |
| 8CE7 | 0xFF | 255 | 6375 RPM | ; |
| 8CE8 | 0x00,0xFF | 255 | | ; |
| 8CEA | 0x0A | 10 | | ;TIME LIMIT FOR L0168 (P/N) |
| 8CEB | 0x0A | 10 | | ;TIME LIMIT FOR L0168 |
| 8CEC | 0x08 | 8 | 100 RPM | ;IDLE HYSTERESIS |
| 8CED | 0x1F | 31 | 31 RPM | ;IDLE RPM ERROR THRESHOLD |
| 8CEE | 0x34 | 52 | 52 RPM | ;IDLE RPM ERROR THRESHOLD |
| 8CEF | 0x08 | 8 | 100 RPM | ;IDLE RPM ERROR THRESHOLD |
| 8CF0 | 0x10 | 16 | 200 RPM | ;IDLE RPM ERROR THRESHOLD |
| 8CF1 | 0x00 | 0 | 0 RPM | ;UNDER IDLE RPM ERROR THRESHOLD |
| 8CF2 | 0x02 | 2 | | ; |
| 8CF3 | 0x2E | 46 | | ;MULTIPLIER FOR OVER/UNDER IDLE RPM (UNSIGNED) |
| 8CF4 | 0x08 | 8 | | ;MULTIPLIER FOR OVER/UNDER IDLE RPM (UNSIGNED) |
| 8CF5 | 0x2E | 46 | | ;MULTIPLIER FOR OVER/UNDER IDLE RPM (UNSIGNED) |
| 8CF6 | 0x06 | 6 | | ;MULTIPLIER FOR OVER/UNDER IDLE RPM (UNSIGNED) |
| 8CF7 | 0x10 | 16 | | ;MULTIPLIER FOR OVER/UNDER IDLE RPM (UNSIGNED) |

| | | | |
|------|------|---|--|
| 8CF8 | 0x06 | 6 | ;MULTIPLIER FOR OVER/UNDER IDLE RPM (UNSIGNED) |
| 8CF9 | 0x08 | 8 | |
| 8CFA | 0x02 | 2 | |

| OVER | IDLE | RPM | ERROR | THRESHOLD | VS | FILTERED | MPH |
|------|------|-------|-------|-----------|----|----------|-----|
| HEX | DEC | VALUE | | | | | MPH |
| 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 01 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 02 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 03 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| 04 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |
| 05 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| 06 | 6 | 6 | 6 | 6 | 6 | 6 | 6 |
| 07 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |
| 08 | 8 | 8 | 8 | 8 | 8 | 8 | 8 |
| 09 | 9 | 9 | 9 | 9 | 9 | 9 | 9 |
| 0A | 10 | 10 | 10 | 10 | 10 | 10 | 10 |
| 0B | 11 | 11 | 11 | 11 | 11 | 11 | 11 |
| 0C | 12 | 12 | 12 | 12 | 12 | 12 | 12 |
| 0D | 13 | 13 | 13 | 13 | 13 | 13 | 13 |
| 0E | 14 | 14 | 14 | 14 | 14 | 14 | 14 |
| 0F | 15 | 15 | 15 | 15 | 15 | 15 | 15 |
| 10 | 16 | 16 | 16 | 16 | 16 | 16 | 16 |
| 11 | 17 | 17 | 17 | 17 | 17 | 17 | 17 |
| 12 | 18 | 18 | 18 | 18 | 18 | 18 | 18 |
| 13 | 19 | 19 | 19 | 19 | 19 | 19 | 19 |
| 14 | 20 | 20 | 20 | 20 | 20 | 20 | 20 |
| 15 | 21 | 21 | 21 | 21 | 21 | 21 | 21 |
| 16 | 22 | 22 | 22 | 22 | 22 | 22 | 22 |
| 17 | 23 | 23 | 23 | 23 | 23 | 23 | 23 |
| 18 | 24 | 24 | 24 | 24 | 24 | 24 | 24 |
| 19 | 25 | 25 | 25 | 25 | 25 | 25 | 25 |
| 1A | 26 | 26 | 26 | 26 | 26 | 26 | 26 |
| 1B | 27 | 27 | 27 | 27 | 27 | 27 | 27 |
| 1C | 28 | 28 | 28 | 28 | 28 | 28 | 28 |
| 1D | 29 | 29 | 29 | 29 | 29 | 29 | 29 |
| 1E | 30 | 30 | 30 | 30 | 30 | 30 | 30 |
| 1F | 31 | 31 | 31 | 31 | 31 | 31 | 31 |
| 20 | 32 | 32 | 32 | 32 | 32 | 32 | 32 |
| 21 | 33 | 33 | 33 | 33 | 33 | 33 | 33 |
| 22 | 34 | 34 | 34 | 34 | 34 | 34 | 34 |
| 23 | 35 | 35 | 35 | 35 | 35 | 35 | 35 |
| 24 | 36 | 36 | 36 | 36 | 36 | 36 | 36 |
| 25 | 37 | 37 | 37 | 37 | 37 | 37 | 37 |
| 26 | 38 | 38 | 38 | 38 | 38 | 38 | 38 |
| 27 | 39 | 39 | 39 | 39 | 39 | 39 | 39 |
| 28 | 40 | 40 | 40 | 40 | 40 | 40 | 40 |
| 29 | 41 | 41 | 41 | 41 | 41 | 41 | 41 |
| 2A | 42 | 42 | 42 | 42 | 42 | 42 | 42 |
| 2B | 43 | 43 | 43 | 43 | 43 | 43 | 43 |
| 2C | 44 | 44 | 44 | 44 | 44 | 44 | 44 |
| 2D | 45 | 45 | 45 | 45 | 45 | 45 | 45 |
| 2E | 46 | 46 | 46 | 46 | 46 | 46 | 46 |
| 2F | 47 | 47 | 47 | 47 | 47 | 47 | 47 |
| 30 | 48 | 48 | 48 | 48 | 48 | 48 | 48 |
| 31 | 49 | 49 | 49 | 49 | 49 | 49 | 49 |
| 32 | 50 | 50 | 50 | 50 | 50 | 50 | 50 |
| 33 | 51 | 51 | 51 | 51 | 51 | 51 | 51 |
| 34 | 52 | 52 | 52 | 52 | 52 | 52 | 52 |
| 35 | 53 | 53 | 53 | 53 | 53 | 53 | 53 |
| 36 | 54 | 54 | 54 | 54 | 54 | 54 | 54 |
| 37 | 55 | 55 | 55 | 55 | 55 | 55 | 55 |
| 38 | 56 | 56 | 56 | 56 | 56 | | |

| | | | | | | |
|------|------|----|-----|-----|---|----|
| 8CFB | 0x04 | 4 | 50 | RPM | 0 | |
| | 0x04 | 4 | 50 | | | 1 |
| | 0x04 | 4 | 50 | | | 2 |
| | 0x04 | 4 | 50 | | | 3 |
| | 0x04 | 4 | 50 | | | 4 |
| | 0x06 | 6 | 75 | | | 5 |
| | 0x08 | 8 | 100 | | | 6 |
| | 0x0A | 10 | 125 | | | 7 |
| | 0x0C | 12 | 150 | | | 8 |
| | 0x0E | 14 | 175 | | | 9 |
| | 0x10 | 16 | 200 | | | 10 |
| | 0x12 | 18 | 225 | | | 11 |
| | 0x14 | 20 | 250 | | | 12 |
| | 0x16 | 22 | 275 | | | 13 |
| | 0x18 | 24 | 300 | | | 14 |
| | 0x1A | 26 | 325 | | | 15 |

THROTTLE FOLLOWER TPS MULTIPLIER VS FILTERED MPH, GOES INTO L0320
RESULT IS THROTTLE FOLLOWER IAC STEPS

| HEX | DEC | VALUE | MPH |
|-----|-----|-------|-----|
|-----|-----|-------|-----|

| | | | | |
|------|------|----|------|-----|
| 8D0B | 0x40 | 64 | 1.0 | 0 |
| | 0x40 | 64 | 1.0 | 8 |
| | 0x40 | 64 | 1.0 | 16 |
| | 0x40 | 64 | 1.0 | 24 |
| | 0x40 | 64 | 1.0 | 32 |
| | 0x30 | 48 | 0.75 | 40 |
| | 0x20 | 32 | 0.5 | 48 |
| | 0x20 | 32 | 0.5 | 56 |
| | 0x20 | 32 | 0.5 | 64 |
| | 0x20 | 32 | 0.5 | 72 |
| | 0x20 | 32 | 0.5 | 80 |
| | 0x20 | 32 | 0.5 | 88 |
| | 0x20 | 32 | 0.5 | 96 |
| | 0x20 | 32 | 0.5 | 104 |
| | 0x20 | 32 | 0.5 | 112 |
| | 0x20 | 32 | 0.5 | 120 |
| | 0x20 | 32 | 0.5 | 128 |

MAX THROTTLE FOLLOWER IAC STEPS VS FILTERED MPH, GOES INTO L0321

| HEX | DEC | VALUE | MPH |
|-----|-----|-------|-----|
|-----|-----|-------|-----|

| | | | | |
|------|------|----|----------|----|
| 8D1C | 0x23 | 35 | 35 STEPS | 0 |
| | 0x23 | 35 | 35 STEPS | 8 |
| | 0x23 | 35 | 35 STEPS | 16 |
| | 0x23 | 35 | 35 STEPS | 24 |
| | 0x23 | 35 | 35 STEPS | 32 |
| | 0x23 | 35 | 35 STEPS | 40 |
| | 0x23 | 35 | 35 STEPS | 48 |
| | 0x2A | 42 | 42 STEPS | 56 |
| | 0x32 | 50 | 50 STEPS | 64 |

| | | | | | |
|------|-----------|-----|-----------|---|----------------------|
| 8D25 | 0x05 | 5 | 5 | ; | ? AIRFLOW HYSTERESIS |
| 8D26 | 0x04 | 4 | 50 RPM | ; | RPM HYSTERESIS |
| 8D27 | 0x00,0x00 | 0 | 0 SECONDS | ; | |
| 8D29 | 0xFF | 255 | 151 DEG C | ; | |
| 8D2A | 0xFF | 255 | 6375 RPM | ; | |

| | | | | |
|------|-----------|------|------------|--|
| 8D2B | 0x50 | 80 | | ;TIME LIMIT |
| 8D2C | 0x00,0x00 | 0 | 0 SECONDS | ; |
| 8D2E | 0xFF | 255 | 3187.5 RPM | ; |
| 8D2F | 0xFF | 255 | 3187.5 RPM | ; |
| 8D30 | 0xFF | 255 | 3187.5 RPM | ; |
| 8D31 | 0x00,0x03 | 3 | 3 SECONDS | ; |
| 8D33 | 0xBC | 188 | 4700 RPM | ;TURN OFF A/C CLUTCH IF RPM > THIS |
| 8D34 | 0xA0 | 160 | 4000 RPM | ; |
| 8D35 | 0x00,0xF0 | 240 | | ;MIN A/C CLUTCH OFF TIME |
| 8D37 | 0x00,0x3C | 60 | 60 SECONDS | ; |
| 8D39 | 0x0C | 12 | 150 RPM | ;DELTA RPM |
| 8D3A | 0xF6 | 246 | 96 %TPS | ; |
| 8D3B | 0xCC | 204 | 79.7 %TPS | ; |
| 8D3C | 0x06,0x40 | 1600 | | ; |
| 8D3E | 0xDB | 219 | 124 DEG C | ; |
| 8D3F | 0xDB | 219 | 124 DEG C | ; |
| 8D40 | 0x14 | 20 | | ;TIME DELAY BEFORE TURNING ON A/C CLUTCH |
| 8D41 | 0x00 | | | ; |
| 8D42 | 0x00 | | | ; |
| 8D43 | 0x00 | 0 | 0 | ; |
| 8D44 | 0x0A | 10 | 10 MPH ; | |
| 8D45 | 0x08 | 8 | 3.1 %TPS | ; |
| 8D46 | 0x80 | 128 | | ;A/C LOAD STEPS MULTIPLIER |

2D TABLE VS IATMAT, RESULT GOES INTO L0322
IS A MULTIPLIER FOR CALCULATING IAC STEPS FOR A/C

| | |
|------|------|
| 8D47 | 0x00 |
| | 0x00 |
| | 0x00 |
| | 0x00 |
| | 0x00 |
| | 0x00 |
| | 0x00 |
| | 0x00 |
| | 0x00 |

| | | | | |
|------|-----------|-----|----------|---|
| 8D50 | 0x23 | 35 | | ;MAX CALCULATED ANTICIPATED IAC STEPS FOR A/C |
| 8D51 | 0x0A | 10 | | ; |
| 8D52 | 0x0C | 12 | 12 MPH ; | |
| 8D53 | 0x08 | 8 | 8 MPH | ; |
| 8D54 | 0x00 | 0 | | ;IATMAT |
| 8D55 | 0x00,0xA0 | 160 | | ; |
| 8D57 | 0x10 | 16 | | ; |
| 8D58 | 0x0A | 10 | | ;IAC STEPS FOR A/C MULTIPLIER |
| 8D59 | 0x0A | 10 | | ; |

DELTA RPM VS DESIRED IDLE RPM

| | |
|------|------|
| 8D5A | 0x20 |
| | 0x20 |
| | 0x20 |
| | 0x20 |
| | 0x20 |
| | 0x20 |
| | 0x0A |
| | 0x20 |
| | 0x20 |
| | 0x20 |
| | 0x20 |
| | 0x20 |

0x20
0x20
0x20
0x20

| | | | | |
|------|------------|-----|----------|-----------------------------------|
| 8D6B | 0x3C | 60 | | ; IATMAT |
| 8D6C | 0x50 | 80 | | ; DEFAULT IATMAT |
| 8D6D | 0x6B | 107 | 40 DEG C | ; COOLANT TEMP |
| 8D6E | 0x01, 0x2C | 300 | 300 RPM | ; |
| 8D70 | 0x0A | 10 | | ; INITIAL VALUE FOR L02B2 (TIMER) |
| 8D71 | 0x00 | 0 | 0 | ; IAC A/C STEPS FOR SLUGGING |
| 8D72 | 0x79 | 121 | | ; A/C PRESSURE |
| 8D73 | 0x1E | 30 | | ; A/C PRESSURE |
| 8D74 | 0x7C | | | ; |
| 8D75 | 0xF1 | | | ; |
| 8D76 | 0x1E | | | ; A/C PRESSURE |
| 8D77 | 0x1C | | | ; A/C PRESSURE |

ANTICIPATED A/C LOAD STEPS VS A/C PRESSURE

8D78 0x0E
0x0E
0x0E
0x0E
0x0E
0x0E
0x10
0x12
0x14
0x1C
0x1E
0x1E
0x1E
0x1E
0x1E
0x1E
0x1E

ANTICIPATED A/C LOAD STEPS VS A/C PRESSURE, IN GEAR

8D89 0x0E
0x0E
0x0E
0x0E
0x0E
0x0E
0x10
0x12
0x14
0x1C
0x1E
0x21
0x21
0x21
0x21
0x21
0x21

| | | | |
|------|------------|----|--------|
| 8D9A | 0x00, 0x32 | 50 | 50 RPM |
| 8D9C | 0x00, 0x32 | 50 | 50 RPM |
| 8D9E | 0x0A | | |
| 8D9F | 0x0F | | |

```

8DA0  0x08
8DA1  0x14
8DA2  0xFF

```

```

MIN VALUE FOR L02CB VS FILTERED MPH
      HEX      DEC      VALUE      MPH

```

```

-----
8DA3  0x0A    10              0
      0x0A    10              1
      0x0A    10              2
      0x0A    10              3
      0x0A    10              4
      0x0A    10              5
      0x0A    10              6
      0x0A    10              7
      0x0A    10              8
      0x0A    10              9
      0x0A    10             10
      0x0A    10             11
      0x0A    10             12
      0x0A    10             13
      0x0A    10             14
      0x0A    10             15

```

```

8DB3  0x01      1          12.5 RPM    ;
8DB4  0x01      1           1           ;
8DB5  0x00,0x20

```

```

TIME DELAY BEFORE MOVING IAC MOTOR VS FILTERED ENGINE TORQUE
WHEN BIT 6 L0094 CLEAR

```

```

-----
8DB7  0x0C
      0x0C
      0x0C
      0x0C
      0x0C
      0x0C
      0x0C
      0x0C
      0x0C
      0x0C
      0x0C

```

```

TIME DELAY BEFORE MOVING IAC MOTOR VS FILTERED ENGINE TORQUE
WHEN BIT 6 L0094 SET

```

```

-----
8DC2  0x0C
      0x0C
      0x0C
      0x0C
      0x0C
      0x0C
      0x0C
      0x0C
      0x0C
      0x0C
      0x0C

```

```

8DCD  0x02
8DCE  0x14      20              ;
8DCF  0x55              ;
8DD0  0x08      8              8    ;INIT VALUE FOR L02B6 IAC TIMER/COUNTER
8DD1  0x08      8              8    ;INIT VALUE FOR L02B7 IAC TIMER/COUNTER

```

| | | | | |
|------|-----------|-----|------------|---|
| 8DD2 | 0x04 | 4 | 4 | ;COMMANDED AIRFLOW FOR FAN #1 |
| 8DD3 | 0x04 | 4 | 4 | ;COMMANDED AIRFLOW FOR FAN #2 |
| 8DD4 | 0x85 | 133 | 60 DEG C | ;CLNT TEMP THRESHOLD |
| 8DD5 | 0x4B | | | ;INITIAL/MIN VALUE FOR L0065 |
| 8DD6 | 0x41 | | | ;INITIAL/MIN VALUE FOR L0067 |
| 8DD7 | 0x14 | 20 | | ; |
| 8DD8 | 0x14 | 20 | | ; |
| 8DD9 | 0x20 | 32 | | ;FILTER COEFF FOR L0065 AND L0067 |
| 8DDA | 0x48,0x00 | | | ;MAX VALUE FOR L0065/L0067 |
| 8DDC | 0x37,0x00 | | | ;MIN VALUE FOR L0065/L0067 |
| 8DDE | 0x05 | 5 | 62.5 RPM | ;RPM HYSTERESIS |
| 8DDF | 0x01 | 1 | 12.5 RPM | ;RPM HYSTERESIS |
| 8DE0 | 0x00,0x0A | 10 | 10 SECONDS | ; |
| 8DE2 | 0x03 | 3 | | ;IS ADDED TO FILTERED COMMANDED AIRFLOW |

UNDER-IDLE THRESHOLD VS DESIRED IDLE

| | HEX | DEC | RPM | DESIRED IDLE RPM |
|------|------|-----|------|------------------|
| 8DE3 | 0x00 | 0 | 0 | 400 RPM |
| | 0x00 | 0 | 0 | 500 |
| | 0x01 | 1 | 12.5 | 600 |
| | 0x02 | 1 | 12.5 | 700 |
| | 0x04 | 4 | 50 | 800 |
| | 0x06 | 6 | 75 | 900 |
| | 0x07 | 7 | 87.5 | 1000 |
| | 0x08 | 8 | 100 | 1100 |
| | 0x0A | 10 | 125 | 1200 |
| | 0x0C | 12 | 150 | 1300 |
| | 0x0E | 14 | 175 | 1400 |
| | 0x10 | 16 | 200 | 1500 |
| | 0x12 | 18 | 225 | 1600 |
| | 0x14 | 20 | 250 | 1700 |
| | 0x14 | 20 | 250 | 1800 |
| | 0x14 | 20 | 250 | 1900 |
| | 0x14 | 20 | 250 | 2000 |

| | | | | |
|------|------|-----|---------|--|
| 8DF4 | 0x08 | 8 | 100 RPM | ; |
| 8DF5 | 0x03 | | | ; |
| 8DF6 | 0x10 | | | ;SUBTRACTIVE OFFSET / MINIMUM D-A/F (L0169) |
| 8DF7 | 0x00 | | | ; |
| 8DF8 | 0x00 | | | ;DIFF BETWEEN COMMANDED AND ACTUAL AIRFLOW |
| | | | | ; FOR A/F < COMMANDED |
| 8DF9 | 0x00 | | | ;DIFF BETWEEN COMMANDED AND ACTUAL AIRFLOW |
| | | | | ; FOR A/F > COMMANDED |
| 8DFA | 0x66 | 102 | | ;D-A/F MULTIPLIER FOR A/F > COMMANDED |
| 8DFB | 0x1C | 28 | | ;D-A/F MULTIPLIER FOR A/F < COMMANDED |
| 8DFC | 0x66 | 102 | | ;D-A/F MULTIPLIER FOR A/F > COMMANDED IN P/N |
| 8DFD | 0x20 | 32 | | ;D-A/F MULTIPLIER FOR A/F < COMMANDED IN P/N |

BASE COMMANDED AIRFLOW*10 VS RPM/12.5

FOR P/N OR NOT 2ND GEAR AND DRV 1 OR 2 SELECTED

8DFE 0x26,0x24,0x21,0x20,0x1E,0x1E,0x1E,0x1E,0x1E,0x1E,0x1E,0x1E,0x1E,0x1E,0x1E,0x1E

FOR 3RD GEAR AND DRV 1 OR 2 SELECTED

8E0F 0x26,0x24,0x21,0x20,0x1F,0x1E,0x1E,0x1E,0x1E,0x1E,0x1E,0x1E,0x1E,0x1E,0x1E,0x1E

FOR PRNDL -> REVERSE

8E20 0x36,0x33,0x31,0x2F,0x2E,0x2E,0x2E,0x2E,0x2E,0x2E,0x2E,0x2F,0x20,0x21,0x22,0x23,0x24,0x25

FOR 3RD GEAR

8E31 0x36,0x33,0x31,0x2F,0x2F,0x20,0x21,0x22,0x23,0x24,0x25,0x26,0x27,0x28,0x29,0x2A,0x2B

```
8E42  0x36,0x33,0x31,0x2F,0x2F,0x2F,0x2F,0x2F,0x2F,0x2F,0x2F,0x2F,0x2F,0x2F,0x2F,0x2F,0x2F
```

```
8E53  0x36,0x33,0x31,0x2F,0x2E,0x2E,0x2E,0x2E,0x2E,0x2E,0x2E,0x2E,0x2E,0x2E,0x2E,0x2E,0x2E
```

```
8E64    0x36,0x33,0x31,0x2F,0x2E,0x2E,0x2E,0x2E,0x2E,0x2E,0x2F,0x20,0x21,0x22,0x23,0x24,0x25,
```

```
8E75  0x3C,0x38,0x34,0x30,0x2A,0x28,0x23,0x23,0x23,0x23,0x23,0x28,0x28,0x28,0x28,0x0A
```

```

8E86      0x00
           0x00
           0x05
8E89      0x5A,0x5A,0x5A,0x5A,0x5A
           0x5A,0x5A,0x5A,0x50,0x46
           0x46,0x46,0x3C,0x32,0x28
           0x14,0x14,0x14,0x14,0x14
           0x13,0x13,0x13,0x13,0x13
           0x13,0x11,0x06,0x03,0x00
           0x13,0x11,0x0D,0x06,0x00
           0x0C,0x0A,0x07,0x04,0x01
           0x09,0x07,0x05,0x03,0x01

```

RESULT GOES INTO COMMANDED AIRFLOW WHEN ERUNTIME < 64 SECONDS

```

8EB6      0x04      TABLE LENGTH (5 VALUES)
          0x00
          0x00
          0x00
          0x00
          0x00

```

[illegible]

| | | |
|------|------|-------|
| 8EBD | 0x00 | 0 MPH |
| | 0x00 | 8 |
| | 0x00 | 16 |
| | 0x02 | 24 |
| | 0x04 | 32 |
| | 0x06 | 40 |
| | 0x09 | 48 |
| | 0x0F | 56 |
| | 0x12 | 64 |

```
; ESSENTIALLY, L0076 IS NOT USED - WILL ALWAYS BE ZERO
```

| | | | | |
|------|-----------|---|-----------|---|
| 8EC6 | 0x00 | | | ;AIRFLOW, ADDED TO COM A/F WHEN 02C1 TIMER ACTIVE |
| 8EC7 | 0x00,0x00 | 0 | 0 SECONDS | ; |
| 8EC9 | 0x00 | 0 | 0 | ;COOLANT THRESHOLD FOR DOING L0076 THRESHOLD |
| 8ECA | 0x00 | | | ;SUBTRACTIVE OFFSET FOR L0076 |

2D TABLE VS L02FE/4 (CUMULATIVE DELTA RPM/12.5 BETWEEN DESIRED AND ACTUAL)

RESULT IS ADDED TO L0076 (NOT USED)

```
-----
8ECB  0x00
      0x00
      0x00
      0x00
      0x00
      0x00
      0x00
      0x00
      0x00
      0x00
      0x00
      0x00
      0x00
      0x00
      0x00
      0x00

8EDB  0x00      0      ;ADDED TO L0076 WHEN CRANKING

; A.I.R CALS
8EDC  0x00      ;
8EDD  0x00
8EDE  0x00
8EDF  0x00
8EE0  0x00
8EE1  0x00,0x00
8EE3  0x00,0x00
8EE5  0x00,0x00

8EE7  0x02,0x58

8EE9  0x02      ;CCP OPTION FLAG
      ;B7 = LV8 VS MAF FOR CCP DC TABLE LOOKUPS

8EEA  0x50      80      31.25 %DC      ;CCP DUTY CYCLE
8EEB  0xFF      255      255 MPH      ;
8EEC  0x7A
8EED  0xF0      240      240 COUNTS      ;LV8 COUNTS
8EEE  0x01      1
8EEF  0x07      7
8EF0  0x78      120      120      ;INTEGRATOR
8EF1  0x78      120      120      ;INTEGRATOR
8EF2  0x71      113      113      ;INTEGRATOR
8EF3  0x6E      110      110      ;INTEGRATOR
8EF4  0x01      1
8EF5  0x05      5
8EF6  0x01      1      ;ADDED TO CCP DC
8EF7  0x0A      10      ;ADDED TO CCP DC
8EF8  0x00,0x75  117
8EFA  0x26      38      14.8 %DC      ;CCP DC
8EFB  0x26      38      14.8 %DC      ;CCP DC
8EFC  0x01      1      0.1 SECOND      ;
8EFD  0x0A      10
8EFE  0x00,0x2D  45      45 SECONDS      ;
8F00  0x00,0x19  25      25 SECONDS      ;
8F02  0xA7      167      85.25 DEG C      ;STARTUP COOLANT TEMP THRESHOLD
8F03  0x92      146      69.5 DEG C      ;COOLANT TEMP THRESHOLD
8F04  0x0A      ;IATMAT FILTER COEFFICIENT
8F05  0x00      ;IATMAT
8F06  0x26      38      14.8 %DC      ;CCP DC
8F07  0x30      ;FILTER COEFFICIENT FOR CCP DUTY CYCLE
```

| | | | | | |
|------|------|---|---|--|-----------------------|
| 8F08 | 0xF8 | | | | ;TPS% |
| 8F09 | 0xFE | | | | ;TPS% WHEN CCP ACTIVE |
| 8F0A | 0x00 | 0 | 0 | | |

BASE CCP DC VS MAF OR LV8 - NOT USED, SINCE BIT 7 L8EE9 IS CLEAR
 DEPENDING ON B7 CCP OPTION WORD

| | HEX | DEC | DUTY CYCLE | AIRFLOW |
|------|------|-----|------------|---------|
| 8F0B | 0x26 | 38 | 14.8 % | |
| | 0x26 | 38 | 14.8 % | |
| | 0x26 | 38 | 14.8 % | |
| | 0x26 | 38 | 14.8 % | |
| | 0x26 | 38 | 14.8 % | |
| | 0x32 | 50 | | |
| | 0x55 | | | |
| | 0x78 | | | |
| | 0x9B | | | |
| | 0xBE | | | |
| | 0xE1 | | | |
| | 0xFF | 255 | 99.6 % | |
| | 0xFF | 255 | 99.6 % | |
| | 0xFF | 255 | 99.6 % | |
| | 0xFF | 255 | 99.6 % | |
| | 0xFF | 255 | 99.6 % | |
| | 0xFF | 255 | 99.6 % | |

MAX CCP DC VS MAF OR LV8
 DEPENDING ON B7 CCP OPTION WORD

| | HEX | DEC | DUTY CYCLE | AIRFLOW |
|------|------|-----|------------|---------|
| 8F1C | 0x26 | 38 | 14.8 % | |
| | 0x26 | 38 | 14.8 % | |
| | 0x26 | 38 | 14.8 % | |
| | 0x69 | 105 | 41.0 % | |
| | 0xB2 | 178 | 69.5 % | |
| | 0xFF | 255 | 99.6 % | |
| | 0xFF | 255 | 99.6 % | |
| | 0xFF | 255 | 99.6 % | |
| | 0xFF | 255 | 99.6 % | |
| | 0xFF | 255 | 99.6 % | |
| | 0xFF | 255 | 99.6 % | |
| | 0xFF | 255 | 99.6 % | |
| | 0xFF | 255 | 99.6 % | |
| | 0xFF | 255 | 99.6 % | |
| | 0xFF | 255 | 99.6 % | |
| | 0xFF | 255 | 99.6 % | |

| | | | | |
|------|------|-----|-----------|---|
| 8F2D | 0x26 | 38 | 14.8 %DC | ;MINIMUM CCP DC |
| 8F2E | 0x26 | 38 | 14.8 %DC | ;MINIMUM CCP DC |
| 8F2F | 0xFF | | | ;OPTION WORD FOR EGR - MUST BE CL LOOP FOR EGR |
| 8F30 | 0x04 | 4 | 1.6 %TPS | ;DISABLE EGR WHEN TPS < THIS |
| 8F31 | 0x03 | 3 | 1.2 %TPS | ;DISABLE EGR, WHEN ACTIVE, WHEN TPS < THIS |
| 8F32 | 0x3C | 60 | 5 DEG C | ;EGR ALLOWED IF STARTUP COOLANT > THIS |
| 8F33 | 0x99 | 153 | 74 DEG C | ;EGR DISABLED IF CLNT < THIS AND SU CLNT < 8F32 |
| 8F34 | 0xEC | 236 | 92.5 %TPS | ;DISABLE EGR WHEN TPS> THIS |
| 8F35 | 0xE6 | 230 | 89.9 %TPS | ;DISABLE EGR, IF ACTIVE, WHEN TPS > THIS |

TIME DELAY FROM STARTUP TO ALLOW EGR VS STARTUP COOLANT
 HEX DEC TIME TEMP

| | | | |
|------|------|--|--|
| 8F36 | 0xFF | | |
| | 0xFF | | |

0x26
0x26
0x26
0x26
0x26
0x20
0x20

EGR DUTY CYCLE MULTIPLIER VS FILTERED CTS

8F3F 0x00
0x00
0x00
0x20
0x47
0x70
0x80
0x80
0x80

EGR DUTY CYCLE MULTIPLIER VS LV8 WHEN IN 4TH GEAR

8F48 0x66
0x66
0x66
0x66
0x66
0x40
0x40
0x40
0x79
0x80
0x80
0x80
0x80

EGR DUTY CYCLE MULTIPLIER VS LV8 WHEN IN 3RD GEAR

8F55 0x66
0x66
0x66
0x6D
0x73
0x40
0x40
0x40
0x80
0x80
0x80
0x80
0x80

EGR DUTY CYCLE MULTIPLIER VS LV8 WHEN IN 2ND GEAR

8F62 0x80
0x80
0x80
0x80
0x80
0x80
0x80
0x80
0x80
0x80

0x80
0x80
0x80

8F6F 0x03 3 3 ;MIN EGR DC DELTA

EGR DUTY CYCLE MULTIPLIER
VS FILTERED ENGINE TORQUE -> GOES INTO L01B6

8F70 0x20
0x80
0x80
0x80
0x80
0x80
0x68
0x50
0x50
0x50
0x50
0x00
0x00
0x00
0x00

8F7F .byte 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
8F87 .byte 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
8F8F .byte 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
8F97 .byte 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
8F9F .byte 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
8FA7 .byte 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
8FAF .byte 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
8FB7 .byte 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
8FBF .byte 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
8FC7 .byte 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
8FCF .byte 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
8FD7 .byte 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
8FDF .byte 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
8FE7 .byte 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
8FEF .byte 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
8FF7 .byte 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x02
8FFF .byte 0xEF

BASE EGR SOLN COMBO VS LV8 AND NLRPMX

9000 0x10
0x30
0x0E
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
0x00,0x00,0x2E,0x2E,0x4C,0x6E,0x8C,0x8C,0x8C,0x8C,0x8C,0x4C,0x00,0x00,0x00,0x00
0x00,0x00,0x2E,0x4C,0x6E,0xAB,0xCC,0xCC,0xCC,0xCC,0x8C,0x4C,0x00,0x00,0x00,0x00
0x00,0x10,0x2E,0x4C,0x6E,0xCC,0xCC,0xEE,0xEE,0xCC,0x8C,0x00,0x00,0x00,0x00,0x00
0x00,0x00,0x2E,0x4C,0x8C,0xCC,0xEE,0xEE,0xEE,0xCC,0x8C,0x00,0x00,0x00,0x00,0x00
0x00,0x00,0x2E,0x4C,0xAB,0xEE,0xEE,0xEE,0xEE,0xEE,0x8C,0x00,0x00,0x00,0x00,0x00
0x00,0x20,0x2E,0x6E,0xAB,0xEE,0xEE,0xEE,0xEE,0xEE,0x8C,0x00,0x00,0x00,0x00,0x00
0x00,0x20,0x2E,0x6E,0xEE,0xEE,0xEE,0xEE,0xEE,0x8C,0x00,0x00,0x00,0x00,0x00
0x00,0x20,0x2E,0x8C,0xEE,0xEE,0xEE,0xEE,0xEE,0x8C,0x00,0x00,0x00,0x00,0x00
0x00,0x20,0x4C,0x8C,0xEE,0xEE,0xEE,0xEE,0xEE,0xEE,0xBC,0x00,0x00,0x00,0x00,0x00

BASE EGR SOLN COMBO VS LV8 FOR RPM > 4800 RPM

909D 0x00,0x00,0x4C,0x8C,0xEE,0xEE,0xEE,0xEE,0xEE,0xEE,0x8C,0x00,0x00,0x00,0x00

MALFUNCTION FLAGS

90AB 0xFD

```
-----
*      B0      1 = P0123 - TPS VOLTAGE HIGH
      B1
*      B2      1 = P0341 - CAM SIGNAL ERROR
*      B3      1 = P0321 - 24X CRANK SIGNAL
*      B4      1 = P1630 - SYSTEM VOLTAGE ERROR
*      B5      1 = P0117 - CTS LOW
*      B6      1 = P0118 - CTS HIGH
*      B7      1 = P0134 - O2 SENSOR ERROR
```

90AC 0xF8

```
-----
      B0
      B1
      B2
*      B3      1 = P1640 - QDM A MALF
*      B4      1 = P0113 - IAT HIGH MALF
*      B5      1 = P0502 - NO VSS SIGNAL
*      B6      1 = P0112 - IAT LOW MALF
*      B7      1 = P0122 - TPS LOW VOLTAGE
```

90AD 0x57

```
-----
*      B0      1 = P0703 - TCC BRAKE SW ERROR
*      B1      1 = P0501 - VSS INTERMITTENT SIGNAL
*      B2      1 = P0755 - SHIFT SOLENOID B ERROR
      B3
*      B4      1 = P0101 - MAF - LOW GMS/SEC INDICATED
      B5
*      B6      1 = P0705 - TRANS RANGE SWITCH ERROR
      B7
```

90AE 0xFF

```
-----
*      B0      1 = P0712 - LOW TRANS TEMP
*      B1      1 = P0713 - HIGH TRANS TEMP
*      B2      1 = P0132 - O2 SENSOR RICH
*      B3      1 = P0131 - O2 SENSOR LEAN
*      B4      1 = P0325 - KNOCK SENSOR CIRCUIT ERROR
*      B5      1 = P1350 - IGNITION CONTROL BYPASS ERROR
*      B6      1 = P0342 - CAMSHAFT SENSOR ERROR
*      B7      1 = P0740 - TCC ERROR
```

90AF 0x3F

```
-----
*      B0      1 = P1650 - QDM B MALF
*      B1      1 = P1405 - EGR SOLN MALF
*      B2      1 = P1404 - EGR SOLN MALF
*      B3      1 = P1403 - EGR SOLN MALF
*      B4      1 = P1361 - IGNITION CONTROL ERROR
*      B5      1 = P1623 - PROM ERROR
      B6
      B7
```

90B0 0x40

```
-----
      B0
      B1
      B2
      B3
```

```

      B4
      B5
*      B6      1 = P1626 - PASSKEY II ERROR - INVALID FREQUENCY AFTER START-UP
      B7

90B1      0x08
-----
      B0
      B1
      B2
*      B3      1 = P1530 - A/C PRESSURE SENSOR ERROR
      B4
      B5
      B6
      B7

90B2      0x06
-----
      B0
*      B1      1 = P1629 - PASSKEY II SIGNAL ERROR
*      B2      1 = NOT USED
      B3
      B4
      B5
      B6
      B7

90B3      0xF5
90B4      0xA8

90B5      0x12
*      B1
      B2
      B3
*      B4
      B5
      B6
      B7

90B6      0x7C
90B7      0x3F      %00111111      ;BIT STATUS MASK
90B8      0x00
90B9      0x00
90BA      0x00
90BB      0x32      50      500 SECONDS      ;MALF TIMER
90BC      0x28      40      ;MALF QUAL
90BD      0x01,0x4D      ;IS ADDED TO DEFAULT MAF
90BF      0x0E,0x00      ;IS ADDED TO DEFAULT MAF
90C1      0x05      5      ;IAC AIRFLOW MULTIPLIER
90C2      0x00,0x03      3      3 PULSES      ;MIN MAF PULSES
90C4      0x14      20      20      ;MIN IAC STEPS WITH MAF MALF
90C5      0x64      100      100      ;MIN IAC STEPS WITH MAF MALF

90C6      0x07      ;IAT LOW LIMIT
90C7      0x85      ;IAT HIGH LIMIT
90C8      0x0C      ;
90C9      0x19
90CA      0x32      ;IAT MALF TIME LIMIT
90CB      0x60      ;DEFAULT IAT A/D COUNTS
90CC      0xF0      ;IAT A/D HIGH LIMIT
90CD      0x23
90CE      0xAA      170      ;
90CF      0x32      ;IAT A/D MALF TIME LIMIT

```

| | | | | |
|------|-----------|--|--|-------------------------|
| 90D0 | 0x00,0x0F | | | ;TIME SINCE RUN ENABLED |
| 90D2 | 0xF7 | | | ;CTS A/D HIGH LIMIT |
| 90D3 | 0x04 | | | ;HIGH CTS TIME LIMIT |
| 90D4 | 0x00,0x03 | | | ;TIME SINCE RUN ENABLED |
| 90D6 | 0x25 | | | ;CTS A/D LOW LIMIT |
| 90D7 | 0x04 | | | ;LOW CTS TIME LIMIT |

DEFAULT CLNT TEMP VS RUNTIME (LSB)

| | | | | |
|------|-----------|---------------------------|--|--|
| 90D8 | 0x1B | | | |
| | 0x1E | | | |
| | 0x27 | | | |
| | 0x30 | | | |
| | 0x38 | | | |
| | 0x40 | | | |
| | 0x46 | | | |
| | 0x4C | | | |
| | 0x53 | | | |
| | 0x5C | | | |
| | 0x65 | | | |
| | 0x6A | | | |
| | 0x70 | | | |
| | 0x77 | | | |
| | 0x7E | | | |
| | 0xAB | | | |
| | 0xD5 | | | |
| 90E9 | 0xAD | | | ;MAX DEFAULT CTS A/D COUNTS |
| 90EA | 0x08 | 8 | | ;TPS A/D COUNTS LOW LIMIT |
| 90EB | 0x28 | 4.0 SECONDS | | ;TIME LIMIT |
| 90EC | 0x64 | 1.96 VOLTS TPS A/D COUNTS | | |
| 90ED | 0x18 | 600 RPM | | ;RPM - FOR BATT (ALTERNATOR) MALF |
| 90EE | 0x32 | 3.2 SEC | | ;TIME LIMIT FOR TPS MALF |
| 90EF | 0x11 | 17 | | ;?? DEFAULT MAF COUNTS ?? |
| 90F0 | 0xF0 | 4.7 VOLTS TPS A/D COUNTS | | |
| 90F1 | 0x03,0x1A | | | ;MAF - MIN AIRFLOW FOR DEFAULT %TPS CALC |
| 90F3 | 0x17 | | | ;IAC POSN A/F MULTIPLIER |

DEFAULT %TPS VS MAF (GM/SEC) FOR TPS MALF

| | | | | |
|------|--|------------|--------------|--|
| 90F4 | 0x00,0x07,0x16,0x1A,0x24,0x2E,0x42,0x5B,0x62,0x6A,0x72,0x78,0x80,0x88,0x90,0x94,0x96 | | | |
| 9105 | 0x38 | 56 | 248 mV | ;O2 SENSOR LOW LIMIT |
| 9106 | 0xFE | 254 | 25.4 SECONDS | ;O2 SENSOR TIME LIMIT |
| 9107 | 0xAA | 170 | 754 mV | ;O2 SENSOR HIGH LIMIT |
| 9108 | 0x78 | 120 | 12.0 SECONDS | ;O2 SENSOR TIME LIMIT |
| 9109 | 0x66 | 102 | 40 %TPS | ;%TPS HIGH LIMIT |
| 910A | 0x0F | 15 | 5.85 %TPS | ;%TPS LOW LIMIT |
| 910B | 0x00,0x28 | 40 | 4 SECONDS | ;TIME SINCE RUN ENABLED |
| 910D | 0x4F | 79 | 350 mV | ;O2 SENSOR LOW LIMIT |
| 910E | 0x7E | 126 | 560 mV | ;O2 SENSOR HIGH LIMIT |
| 910F | 0x08 | 8 | | ;TPS% LIMIT |
| 9110 | 0x1E | 30 | 3 SECONDS | ;TIME LIMIT |
| 9111 | 0x6E | 110 | 42.5 DEG C | ;COOLANT LIMIT |
| 9112 | 0x96 | 150 | | ;HIGH LIMIT - L0131 - FOR 24x CRANK SENSOR |
| 9113 | 0xC8 | | | ;KNOCK SENSOR MALF TIME LIMIT |
| 9114 | 0x55 | 15 DEGREES | | ;DEFAULT KNOCK RETARD |
| 9115 | 0x4C | | | ;KNOCK SENSOR A/D LOW LIMIT |
| 9116 | 0xB2 | | | ;KNOCK SENSOR A/D HIGH LIMIT |
| 9117 | 0x0A | | | ;M341 TIMER HIGH LIMIT |
| 9118 | 0x32 | 50 | 5 SECONDS | ;TIME LIMIT |
| 9119 | 0x04 | 4 | 4 | ;TIME LIMIT |

| | | | | |
|------|-----------|-----|---------|---------------------|
| 911A | 0x01,0x00 | 256 | 5 VOLTS | ;EST VOLTAGE |
| 911C | 0x10 | 16 | 400 RPM | ;ENGINE SPEED LIMIT |

EGR MALF QUALIFIERS

| | | | | |
|------|------|-----|------------|----------------------------------|
| 911D | 0xA5 | 165 | 83.7 DEG C | ; |
| 911E | 0x80 | 128 | 566 mV | ;O2 A/D COUNTS |
| 911F | 0x45 | 69 | 69 MPH | ; |
| 9120 | 0x57 | 87 | 2175 RPM | ;UPPER THRESHOLD |
| 9121 | 0x3F | 63 | 1575 RPM | ;UPPER THRESHOLD |
| 9122 | 0x3F | 63 | 1575 RPM | ;UPPER THRESHOLD |
| 9123 | 0x30 | 48 | 1200 RPM | ;LOWER THRESHOLD |
| 9124 | 0x2E | 46 | 1150 RPM | ;LOWER THRESHOLD |
| 9125 | 0x2A | 42 | 1050 RPM | ;LOWER THRESHOLD |
| 9126 | 0x06 | 6 | 6 | ;TIME THRESHOLD |
| 9127 | 0x08 | 8 | 8 | ;TIME THRESHOLD |
| 9128 | 0x08 | 8 | 8 | ;TIME THRESHOLD |
| 9129 | 0x05 | 5 | 125 RPM | ;DELTA RPM |
| 912A | 0x04 | 4 | 100 RPM | ;DELTA RPM |
| 912B | 0x04 | 4 | 100 RPM | ;DELTA RPM |
| 912C | 0x30 | | | |
| 912D | 0x48 | | | |
| 912E | 0x60 | | | |
| 912F | 0x03 | 3 | | |
| 9130 | 0x06 | 6 | | |
| 9131 | 0x09 | 9 | | |
| 9132 | 0x05 | 5 | | |
| 9133 | 0x08 | 8 | | |
| 9134 | 0x0C | 12 | 12 | |
| 9135 | 0x2D | 35 | 199 mV | ;O2 VOLTS THRESHOLD |
| 9136 | 0x04 | | | |
| 9137 | 0x12 | 18 | 18 MPH | ;CHANGE IN MPH FROM THIS TO ZERO |
| 9138 | 0xA0 | 160 | | ;TIME LIMIT FOR D-MPH |

DTC P0502 QUALIFICATIONS

| | | | | |
|------|------|-----|-----------|--------------------------------------|
| 9139 | 0x03 | 3 | 3 MPH | ;VSS LOW LIMIT |
| 913A | 0x78 | 120 | 3000 RPM | ;VSS RPM LIMIT |
| 913B | 0x28 | 40 | 4 SECONDS | ;TIME LIMIT FOR TRANSAXLE OUT OF P/N |
| 913C | 0x14 | 20 | 2 SECONDS | ;VSS TIME LIMIT |

DTC P1530 QUALIFICATIONS

| | | | | |
|------|-----------|-----|-----------|---|
| 913D | 0xFC | 252 | 4.9 VOLTS | ;A/C PRESSURE TRANSDUCER HIGH LIMIT |
| 913E | 0x05 | 5 | 0.3 VOLTS | ;A/C PRESSURE TRANSDUCER LOW LIMIT |
| 913F | 0xFE | 254 | 5 SECONDS | ;A/C PRESS TRANS MALF TIME LIMIT |
| 9140 | 0x23,0x28 | | | |
| 9142 | 0x0A | | | |
| 9143 | 0x00,0x0F | | | |
| 9145 | 0x03 | 3 | 3 | |
| 9146 | 0x00,0x0A | 10 | 10 | |
| 9148 | 0x07 | 7 | 7 MPH | ;SPEED LIMIT |
| 9149 | 0x04 | 4 | 4 PULSES | ;DELTA VATS SIGNAL AFTER STARTUP LOW LIMIT |
| 914A | 0x06 | 6 | 6 PULSES | ;DELTA VATS SIGNAL AFTER STARTUP HIGH LIMIT |
| 914B | 0x14 | | | ;DELTA VATS SIGNAL TIME LIMIT |
| 914C | 0xAD | | | ;BATT VOLTS - 17.3 VOLTS |
| 914D | 0x5A | | | ;BATT VOLTS - 9.0 VOLTS |
| 914E | 0x64 | | | ;BATT (ALTERNATOR) MALF TIME LIMIT |

? QDM FAULT QUALIFICATIONS

| | | | | |
|------|-----------|----|-----------|-------------------------|
| 914F | 0x00,0x05 | 5 | 5 SECONDS | ;TIME SINCE RUN ENABLED |
| 9151 | 0x02 | | | |
| 9152 | 0x32 | 50 | 5 SECONDS | ; |

;OPTION FLAG - RELATED TO PWM OUTPUTS

| | | | | | |
|------|-----------|-----|-------------|-------------------------|--|
| 9153 | 0xFF | | | | |
| | B0 | | | | |
| | B1 | | | | |
| | B2 | | | | |
| | B3 | | | | |
| | B4 | | | | |
| | B5 | | | | |
| | B6 | | | | |
| | B7 | | | | |
| 9154 | 0x04 | | | | |
| 9155 | 0x28 | | | | |
| 9156 | 0x04 | | | | |
| 9157 | 0x23 | 35 | 35 MPH | ;SPEED LIMIT | |
| 9158 | 0x04 | 4 | 4 | ; | |
| 9159 | 0x64 | 100 | 10 SECONDS | ; | |
| 915A | 0xFF | | | ;TRANS A/D HIGH LIMIT | |
| 915B | 0x00,0xBA | 186 | 186 SECONDS | ;TIME SINCE RUN ENABLED | |
| 915D | 0x10 | 16 | 1.6 SECONDS | ;TRANS A/D TIME LIMIT | |
| 915E | 0xFB | | | ;TRANS TEMP HIGH LIMIT | |
| 915F | 0x00,0x0A | 10 | 10 SECONDS | ;TIME SINCE RUN ENABLED | |
| 9161 | 0x0A | 10 | 1 SECOND | ;TRANS TEMP TIME LIMIT | |
| 9162 | 0x00,0x30 | 48 | | ;N/V RATIO HIGH LIMIT | |
| 9164 | 0x00,0x2D | 45 | | ;N/V RATIO LOW LIMIT | |
| 9166 | 0x00,0x23 | 35 | | ;N/V RATIO HIGH LIMIT | |
| 9168 | 0x00,0x1F | 31 | | ;N/V RATIO LOW LIMIT | |
| 916A | 0x64 | 100 | 10 SECONDS | ;TIME LIMIT | |
| 916B | 0x0A | 10 | | | |
| 916C | 0x64 | 100 | 10 VOLTS | ;BATT VOLTS | |
| 916D | 0x02 | 2 | | ;L0064 LOW LIMIT | |
| 916E | 0x64 | 100 | 10 SECONDS | ;L0064 TIME LIMIT | |
| 916F | 0x23 | 35 | | ;TIME LIMIT | |
| 9170 | 0x00,0x76 | 118 | 118 | ; | |
| 9172 | 0x05 | 5 | 5 MPH | ;SPEED LIMIT | |
| 9173 | 0x08 | 8 | 8 | ; | |
| 9174 | 0x14 | 20 | 2 SECONDS | ;TIME LIMIT | |
| 9175 | 0x00,0x32 | 50 | | ; | |
| 9177 | 0x50 | 80 | 80 COUNTS | ;LV8 COUNTS LIMIT | |
| 9178 | 0x1A | 26 | 10 %TPS | ;TPS LIMIT | |
| 9179 | 0x1A | 26 | | | |

DEFAULT MAF - VS %TPS AND RPM/25

| | |
|------|--|
| 917A | 0x00 |
| | 0x00 |
| | 0x09 |
| | 0x05,0x0D,0x16,0x1C,0x1F,0x20,0x21,0x22,0x23 |
| | 0x05,0x0A,0x10,0x13,0x17,0x18,0x19,0x1B,0x1C |
| | 0x05,0x0B,0x12,0x16,0x19,0x1B,0x1D,0x1F,0x20 |
| | 0x05,0x0C,0x13,0x19,0x1D,0x1F,0x20,0x21,0x22 |
| | 0x04,0x0C,0x14,0x1C,0x21,0x22,0x23,0x24,0x25 |
| | 0x05,0x0D,0x16,0x1C,0x23,0x25,0x27,0x29,0x2A |
| | 0x06,0x0E,0x17,0x1F,0x27,0x2B,0x2E,0x30,0x31 |
| | 0x06,0x0F,0x19,0x21,0x2A,0x2E,0x32,0x35,0x37 |
| | 0x06,0x10,0x1A,0x23,0x2C,0x32,0x37,0x3A,0x3D |
| | 0x06,0x11,0x1C,0x26,0x31,0x39,0x41,0x45,0x49 |
| | 0x06,0x12,0x1D,0x29,0x36,0x41,0x4B,0x51,0x56 |
| | 0x07,0x15,0x21,0x2C,0x37,0x43,0x4E,0x58,0x61 |

DEFAULT MAF VS %TPS FOR RPM OVER 4800

| | |
|------|--|
| 91E9 | 0x09,0x17,0x25,0x2F,0x38,0x46,0x52,0x5F,0x6D |
|------|--|

```

91F2    0x00,0x7B

NOT ? ALDL LIST
91F4    0x00,0x00    ;
91F6    0x01,0x55    ;
91F8    0x01,0x41    ;A TIMER/COUNTER
91FA    0x02,0x76    ;
91FC    0x00,0x75    ;LSB OF A TIMER/COUNTER
91FE    0x00,0x76
9200    0x00,0x86    ;BIT STATUS FLAG
9202    0x00,0x00
9204    0x00,0x00
9206    0x00,0x00
9208    0x02,0x80    ;A TIMER/COUNTER
920A    0x00,0x71    ;A TIMER/COUNTER
920C    0x01,0xF6
920E    0x00,0x70    ;A TIMER/COUNTER
9210    0x00,0x7A    ;0x977A IS STORED HERE
9212    0x02,0xBF    ;A TIMER/COUNTER
9214    0x01,0xF5
9216    0x02,0xF3    ;A TIMER/COUNTER
9218    0x00,0x00
921A    0x00,0x65    ;0x8DD5 IS STORED HERE
921C    0x00,0x00
921E    0x02,0xC1    ;A TIMER/COUNTER
9220    0x00,0x00
9222    0x02,0xC8    ;L00EB (20/REF PER) IS STORED HERE
9224    0x02,0xC9    ;LSB OF ABOVE
9226    0x00,0x84    ;BIT STATUS FLAG
9228    0x00,0x83    ;BATT VOLTS
922A    0x01,0x45    ;A TIMER/COUNTER
922C    0x01,0x5E    ;L029E (TIMER) IS STORED HERE
922E    0x01,0x56
9230    0x02,0xE4    ;LSB OF L02E3 (TIMER/COUNTER)
9232    0x02,0x8A
9234    0x01,0x95    ;L0254 IS STORED HERE (INVERSE OF CH#9 A/D)
9236    0x01,0x94    ;
9238    0x01,0x93    ;A TIMER/COUNTER
923A    0x01,0x69    ;
923C    0x02,0x6F    ;PREVIOUS PASSKEY II SIGNAL
923E    0x00,0x00
9240    0x00,0x00
9242    0x00,0x00
9244    0x00,0x00
9246    0x00,0x00
9248    0x00,0x00
924A    0x00,0x00
924C    0x02,0x13    ;
924E    0x02,0x72    ;
9250    0x00,0x00
9252    0x01,0x12    ;LSB OF L0111
9254    0x00,0x00
9256    0x00,0x00
9258    0x02,0x12
925A    0x00,0x00
925C    0x00,0x00
925E    0x02,0x70    ;
9260    0x02,0x71
9262    0x02,0x9A    ;PREV EST VOLTAGE
9264    0x02,0x75    ;A TIMER/COUNTER
9266    0x02,0x26

```

```

9268 0x00,0x00
926A 0x00,0x00
926C 0x00,0x00
926E 0x00,0x00
9270 0x00,0x00

9272 0x4A,0x4C      19020      ;? EMPERICALLY DERIVED MULTIPLIER FOR
                                ; ACCUMULATED FUEL CALCULATION

9274 37              55          ;
9275 0x01,0x34      ;

```

```

;-----
; ? SERIAL DATA MESSAGE RECEIVED, OR IS IT TRANSMITTED IN BROADCAST REMOTE MODE ?
;-----

```

```

9277 F4              ;ECM DEVICE ID

9278 9282            ;NEXT MESSAGE ENTRY ADDRESS
927A 01              ;DEVICE ID
927B 00              ;FLAG = USE NO MEMORY
927C 00              ;NUMBER OF OUTPUT DATA BYTES
927D 0226            ;
927F 0226            ;
9281 01

9282 928C            ;NEXT MESSAGE ENTRY ADDRESS
9284 02              ;DEVICE ID
9285 00              ;FLAG = USE NO MEMORY
9286 00              ;NUMBER OF OUTPUT DATA BYTES
9287 0226            ;
9289 0226            ;
928B 02

928C 92F0            ;NEXT MESSAGE ENTRY ADDRESS
928E 03              ;DEVICE ID
928F 00              ;FLAG = USE NO MEMORY
9290 18      24      ;NUMBER OF OUTPUT DATA BYTES
9291 0204            ;SERIAL DATA MODE WD
9293 0226            ;
9295 03

```

```

*****
* F9MSGT TABLE      *
* MESSAGE SCHEDULE TABLE *
* TABLE VALUE = ADDRESS *
*****

```

```

9296 92B6      0
9298 0000      1
929A 0000      2
929C 0000      3
929E 92B6      4
92A0 0000      5
92A2 92C6      6
92A4 0000      7
92A6 92B6      8
92A8 0000      9
92AA 0000      A
92AC 0000      B
92AE 92B6      C
92B0 0000      D
92B2 92E4      E
92B4 0000      F

```

* ALDL TRANSMIT TABLES FOR MODES ZERO (NORMAL MODE) AND SEVEN *
*

| | | | |
|------|------|--|--|
| 92B6 | 0000 | | ;NEXT MESSAGE ENTRY ADDRESS |
| 92B8 | 0A | | ;MESSAGE ID |
| 92B9 | 80 | | ;FLAG = USE ROM TABLE |
| 92BA | 03 | | ;NUMBER OF OUTPUT DATA BYTES |
| 92BB | 023A | | ;ADDRESS OF OUTPUT MESSAGE BUFFER |
| 92BD | 0226 | | ;ADDRESS OF INPUT MESSAGE BUFFER |
| 92BF | F4 | | ;DEVICE ID |
| 92C0 | 02A8 | | ;ALDL FLAGWORD |
| | | | ;B0 1 = |
| | | | ;B1 1 = P/N MODE |
| | | | ;B2 1 = A/C CLUTCH ON |
| | | | ;B3 1 = "NO VSS SIGNAL" MALF |
| | | | ;B4 NOT USED |
| | | | ;B5 1 = ALDL TESTER IN CONTROL OF LINK |
| | | | ;B6 1 = SES LIGHT ON |
| | | | ;B7 1 = BRAKE APPLIED |
| 92C2 | 00EB | | ;TRANNNY RPM MSB |
| 92C4 | 00EC | | ;TRANNNY RPM LSB |

;-----

| | | | |
|------|------|----|--|
| 92C6 | 0000 | | ;NEXT MESSAGE ENTRY ADDRESS |
| 92C8 | 05 | | ;MESSAGE ID |
| 92C9 | 80 | | ;FLAG = USE ROM TABLE |
| 92CA | 0A | 10 | ;NUMBER OF OUTPUT DATA BYTES |
| 92CB | 023A | | ;ADDRESS OF OUTPUT MESSAGE BUFFER |
| 92CD | 0226 | | ;ADDRESS OF INPUT MESSAGE BUFFER |
| 92CF | F4 | | ;DEVICE ID |
| 92D0 | 02A6 | | ;ALDL FLAGWORD |
| | | | ;B0 1 = |
| | | | ;B1 1 = P/N MODE |
| | | | ;B2 1 = A/C CLUTCH ON |
| | | | ;B3 NOT USED |
| | | | ;B4 NOT USED |
| | | | ;B5 1 = ALDL TESTER IN CONTROL OF LINK |
| | | | ;B6 1 = SES LIGHT ON |
| | | | ;B7 1 = IAC MOTOR RESET TO ZERO |
| 92D2 | 02A7 | | ;ALDL FLAGWORD |
| | | | ;B0 1 = PASSKEY II ERROR |
| | | | ;B1 1 = VATS FAILED |
| | | | ;B2 NOT USED |
| | | | ;B3 1 = "NO VSS SIGNAL" MALF |
| | | | ;B4 NOT USED |
| | | | ;B5 1 = A/C REQUESTED |
| | | | ;B6 1 = CTS MALFS OCCURED |
| | | | ;B7 NOT USED |
| 92D4 | 00BD | | ;MPH |
| 92D6 | 9274 | | ;0x37 |
| 92D8 | 0083 | | ;BATT VOLTS |
| 92DA | 0272 | | ;? ACCUMULATED FUEL |
| 92DC | 0333 | | ;RESERVED FOR TESTING STACK OVERFLOW |
| 92DE | 0333 | | ;RESERVED FOR TESTING STACK OVERFLOW |
| 92E0 | 0333 | | ;RESERVED FOR TESTING STACK OVERFLOW |
| 92E2 | 00A6 | | ;COOLANT TEMP |

#####

* F9MSG3 #
* MESSAGE TO BE TRANSMITTED IN BROADCAST REMOTE MODE #
* TABLE VALUE = ADDRESS #

#####

| | | | |
|------|------|--|-----------------------------|
| 92E4 | 0000 | | ;NEXT MESSAGE ENTRY ADDRESS |
| 92E6 | F0 | | ;MESSAGE ID |

```

92E7 80 ;FLAG = USE ROM TABLE
92E8 01 1 ;NUMBER OF OUTPUT DATA BYTES
92E9 023A ;ADDRESS OF OUTPUT MESSAGE BUFFER
92EB 0226 ;ADDRESS OF INPUT MESSAGE BUFFER
92EC F4 ;DEVICE ID
92EE 9277 ;MODE ? ADDRESS
92F0 930D ;NEXT MESSAGE ADDRESS

```

```

*****
* SDRF4TBL *
* MESSAGE RECEIVED IN RESPONSE TO ALDL POLLING MESSAGE *
* TABLE VALUE = ADDRESS *
*****

```

```

92F2 F4 ;MESSAGE ID
92F3 00 ;FLAG = USE NO BUFFER
92F4 80 ;? NUMBER OF OUTPUT DATA BYTES
92F5 023A ;ADDRESS OF OUTPUT MESSAGE BUFFER
92F7 0226 ;ADDRESS OF INPUT MESSAGE BUFFER
92F9 9277 ;SERIAL DATA RECEIVE MSG, MODE ?

```

```

92FB 930D ;SERIAL DATA RECEIVE MSG, MODE 0
92FD 931A ;SERIAL DATA RECEIVE MSG, MODE 1
92FF 93C8 ;SERIAL DATA RECEIVE MSG, MODE 2
9301 93D2 ;SERIAL DATA RECEIVE MSG, MODE 3
9303 93DC ;SERIAL DATA RECEIVE MSG, MODE 4
9305 93E6 ;SERIAL DATA RECEIVE MSG, MODE 5
9307 93F0 ;SERIAL DATA RECEIVE MSG, MODE 6
9309 93FA ;SERIAL DATA RECEIVE MSG, MODE 7
930B 9404 ;SERIAL DATA RECEIVE MSG, MODE 8

```

```

;-----
; ? TRANSMITTED IN BROADCAST REMOTE MODE
;-----

```

```

930D 0000 ;NEXT MESSAGE ENTRY ADDRESS
930F F4 ;MESSAGE ID
9310 80 ;FLAG = USE PROM TABLE
9311 01 ;NUMBER OF OUTPUT DATA BYTES
9312 023A ;ADDRESS OF OUTPUT MESSAGE BUFFER
9314 022B ;ADDRESS OF INPUT MESSAGE BUFFER
9316 F4 ;DEVICE ID
9317 9319 ;
9319 00 ;

```

```

;-----
; MODE 1 - TRANSMIT FIXED DATA STREAM, MESSAGE 0
; ALDL DEV MUST REQUEST MODE 1 BY XMITING THE FOLLOWING MSG TO THE ECM
;   DEVICE ID      =    $F4
;   MSG LENGTH     =    $57    (1+85)
;   MODE           =    $01
;   MESSAGE NUMB= $00
;   CKSUM
;
; THE ECM WILL RESPOND WITH
;   DEVICE ID      =    $F4
;   MSG LENGTH     =    $99    (68+85)
;   MODE           =    $01
;   DATA BYTES....
;   CKSUM
;-----

```

```

931A 0000 ;NEXT MESSAGE ENTRY ADDRESS
931C F4 ;MESSAGE ID
931D 80 ;FLAG - USE PROM TABLE
931E 44 ;68 - NUMBER OF OUTPUT BYTES
931F 023A ;ADDRESS OF OUTPUT MESSAGE BUFFER

```

```

9321    022B                ;ADDRESS OF INPUT MESSAGE BUFFER
9323    F4                  ;DEVICE ID

```

| ALDL LIST | POSITION | |
|-----------|----------|--|
| 9324 | 8000 | 1 ;PROM ID A |
| 9326 | 8001 | 2 ;PROM ID B |
| 9328 | 010A | 3 ;COOLANT TEMP (NOT DEFAULTED) DEG C = .75N - 40 |
| 932A | 0254 | 4 ;IAT INVERSE A/D COUNTS V = 5(256 - N)/256 |
| 932C | 00F6 | 5 ;TPS A/D COUNTS V = 5N/255 |
| 932E | 00AA | 6 ;% TPS % = N/2.55 |
| 9330 | 40C0 | 7 ;NEWREFPER MSB |
| 9332 | 40C0 | 8 ;NEWREFPER LSB |
| 9334 | 0057 | 9 ;IAC POSITION N = COUNTS |
| 9336 | 00DE | 10 ;DESIRED IDLE (RPM/12.5) |
| 9338 | 4108 | 11 ;RAWDSPFL MSB (MAF) GRAMS/SEC. = N/256 |
| 933A | 4108 | 12 ;RAWDSPFL LSB |
| 933C | 00BA | 13 ;LV8 |
| 933E | 4290 | 14 ;SAP SPK ADV TOTAL UNLIMITED SA REL. TO TDC MSB |
| 9340 | 4290 | 15 ;SAP SPK ADV TOTAL UNLIMITED SA REL. TO TDC LSB |
| 9342 | 029C | 16 ; DEG = 90N/256 (N IS SIGNED) |
| | | 17 ; KNOCK COUNTS (ECU PA3 COUNTER VALUE FROM |
| | | 18 ; LAST MINOR LOOP) |
| 9344 | 00F3 | 17 ;KNOCK RETARD DEGREES |
| 9346 | 00A9 | 18 ;MINOR LOOP O2 MV (A/D READ) mV = 4.44N |
| 9348 | 015E | 19 ;O2 XCOUNTS IN LAST SECOND |
| 934A | 00AF | 20 ;FILTERED BLM (0 - 2.0) |
| 934C | 00B3 | 21 ;INTEGRATOR |
| 934E | 00AE | 22 ;BLM CELL |
| 9350 | 0077 | 23 ;NVMWD2 |
| | | 24 ;B0 1 = IGNITION ON - FOR LOW OIL LEVEL LOGIC |
| | | 25 ;B1 1 = IGNITION OFF - FOR LOW OIL LEVEL LOGIC |
| | | 26 ;B2 1 = STEPPER MOTOR CRUISE PRESENT ON VEHICLE |
| | | 27 ;B3 1 = ENGINE COLD - FOR LOW OIL LEVEL LOGIC |
| | | 28 ;B4 1 = OIL DRAINED BACK - FOR LOW OIL LEVEL LOGIC |
| | | 29 ;B5 1 = MALF 755A (F31 SOLENOID B FAILED OFF) |
| | | 30 ;B6 1 = MALF 755B (F31 SOLENOID B FAILED ON) |
| | | 31 ;B7 1 = SES LIGHT ON AT STALL |
| 9352 | 0083 | 24 ;BATT VOLTS V = N/10 |
| 9354 | 00F4 | 25 ;CCP DUTY CYCLE %DC = N/2.56 |
| 9356 | 0132 | 26 ;TIMER FOR MALF 341 (INTERMITTENT CAM SIGNAL) |
| 9358 | 0183 | 27 ;BAD CYLINDER ID |
| 935A | 0093 | 28 ;CAMFLAG |
| | | 29 ;B0 NOT USED |
| | | 30 ;B1 NOT USED |
| | | 31 ;B2 NOT USED |
| | | 32 ;B3 NOT USED |
| | | 33 ;B4 NOT USED |
| | | 34 ;B5 1 = CAM PULSE SEEN DURING CRANK |
| | | 35 ;B6 1 = CAM PULSE SEEN |
| | | 36 ;B7 1 = CAM PULSE SEEN - USED FOR CYLCNTR |
| 935C | 4032 | 29 ;RUN TIME MSB |
| 935E | 4032 | 30 ;RUN TIME LSB |
| 9360 | 01A9 | 31 ;ALDLMW2 |
| | | 32 ;B0 1 = VATS SHUT OFF FUEL |
| | | 33 ;B1 1 = SMC ENGAGED |
| | | 34 ;B2 1 = SMC INHIBITED |
| | | 35 ;B3 1 = ENGINE RUNNING |
| | | 36 ;B4 1 = LOW OIL LIGHT ON |
| | | 37 ;B5 1 = A.I.R. ENABLED |
| | | 38 ;B6 1 = RICH (0=LEAN) |
| | | 39 ;B7 1 = CLOSED LOOP |
| 9362 | 02A9 | 32 ;ALDLMW1 |
| | | 33 ;B0 1 = 2 ND GEAR START REQ LOW (SNOW SHFT MODE) |

| | | | |
|------|------|----|--|
| | | | ;B1 1 = DEGR SOLN A ON ;B2 1 = DEGR SOLN B ON ;B3 1 = DEGR SOLN C ON ;B4 1 = SHIF LIGHT IS ON ;B5 1 = PSPS CRAMP PRESENT ;B6 NOT USED ;B7 1 = A/C REQUEST ON (NOT USED) |
| 9364 | 008A | 33 | ;FMDBYTE1 ;B0 1 = PARITY HIGH ;B1 1 = INPUT C HIGH ;B2 1 = INPUT B HIGH ;B3 1 = INPUT A HIGH ;B4 1 = LOW OIL ;B5 1 = (A CAR - CRUISE STATUS FOR SMC (0 = ENGAGED) ;B6 1 = TCC/BRAKE SW HIGH ;B7 1 = A/C REQUESTED (HIGH) |
| 9366 | 008E | 34 | ;FMD INPUT STATUS WORD ;B0 1 = P/N MODE ;B1 0 = IN 2 ND GEAR ;B2 0 = IN 3 RD GEAR ;B3 0 = IN 4 TH GEAR ;B4 0 = IN 5 TH GEAR (MANUAL TRANS) ;B5 1 = LOW OIL LEVEL ;B6 1 = TCC/BRAKE SW APPLIED (CKT OPEN) ;B7 0 = A/C REQUEST |
| 9368 | 0094 | 35 | ;QDMMW ;B0 NOT USED ;B1 1 = QDM FAULT1 ;B2 1 = QDM FAULT2 ;B3 USED IN THE XIRQ ROUTINE ;B4 1 = A/C WAS ON THIS CRANK (DUE TO SLUGGING) ;B5 1 = LOW OIL (WILL TURN ON LIGHT) ;B6 USED IN IAC ROUTINE ;B7 1 = A/C CLUTCH ON |
| 936A | 0096 | 36 | ;LCCPMW ;B0 1 = CCP PURGE ON ;B1 1 = FAN #1 ON ;B2 1 = RPM HIGH IN P/N ;B3 1 = TCC LOCKED ;B4 1 = FAN #2 ON ;B5 1 = PSPS CRAMP ;B6 1 = USE TABLE F7MAXTRQ FOR 3 RD GEAR TORQ MGMT ;B7 1 = NORMAL A/C REQ HAS TURNED A/C ON AT STARTUP |
| 936C | 0263 | 37 | ;MPH = N |
| 936E | 00A1 | 38 | ;PRNDL SW STATUS FLAG ;B0 1 = INVALID ;B1 1 = DRIVE 1 ;B2 1 = DRIVE 2 ;B3 1 = DRIVE 3 ;B4 1 = DRIVE 4 ;B5 1 = NEUTRAL ;B6 1 = REVERSE ;B7 1 = PARK |
| 9370 | 00E9 | 39 | ;TRANNNY GEAR BYTE - COMMANDED GEAR |
| 9372 | 01C7 | 40 | ;TCC SLIP RPM RPM = 2N - 255 |
| 9374 | 00EA | 41 | ;TCC PWM DUTY CYCLE %DC = N/2.55 |
| 9376 | 00A3 | 42 | ;TCCPMMW ;B0 1 = APPLY MODE ;B1 1 = ON MODE ;B2 1 = RELEASE MODE ;B3 1 = OFF MODE ;B4 1 = POSITIVE DELTA TPS RELEASE OF TCC ;B5 1 = TCCRAMPS IS NEGATIVE ;B6 1 = TCC SLIP REQUESTED FOR A/C ENGAGEMENT |

| | | | |
|------|------|----|--|
| | | | ;B7 1 = ABSOLUTE SLIP HAS EXCEEDED KLOCKH |
| 9378 | 0333 | 43 | ;RESERVED FOR TESTING STACK OVERWRITE |
| 937A | 0333 | 44 | ;RESERVED FOR TESTING STACK OVERWRITE |
| 937C | 00F9 | 45 | ;A/C PRESSURE - PSI = 1.92N - 26.88 |
| 937E | 01D6 | 46 | ;TRANS TEMP (NON DEF) DEG C |
| 9380 | 0284 | 47 | ;N/V RATIO N = RPM/MPH |
| 9382 | 0333 | 48 | ;RESERVED FOR TESTING STACK OVERWRITE |
| 9384 | 001B | 49 | ; MALFFLG1 HISTORY MALF FLAG WORD 1 |
| | | | ;0 P0123 - TPS VOLTAGE HIGH |
| | | | ;1 NOT USED |
| | | | ;2 P0341 - CAM SIGNAL ERROR |
| | | | ;3 P0321 - IC 24X SIGNAL ERROR |
| | | | ;4 P1630 - SYSTEM VOLTAGE ERROR |
| | | | ;5 P0117 - ENGINE COOLANT TEMP. LOW |
| | | | ;6 P0118 - ENGINE COOLANT TEMP. HIGH |
| | | | ;7 P0134 - HO2S ERROR |
| 9386 | 001C | 50 | ; MALFFLG2 HISTORY MALF FLAG WORD 2 |
| | | | ;0 NOT USED |
| | | | ;1 NOT USED |
| | | | ;2 NOT USED |
| | | | ;3 P1640 - QDM A ERROR |
| | | | ;4 P0113 - IAT SENSOR HIGH |
| | | | ;5 P0502 - VSS CKT. SIGNAL ERROR |
| | | | ;6 P0112 - IAT SENSOR LOW |
| | | | ;7 P0122 - TPS VOLTAGE LOW |
| 9388 | 001D | 51 | ; MALFFLG3 HISTORY MALF FLAG WORD 3 |
| | | | ;0 P0703 - TCC BRAKE SW. ERROR |
| | | | ;1 P0501 - VSS CKT. NO SIGNAL |
| | | | ;2 P0755 - SSB ERROR |
| | | | ;3 NOT USED |
| | | | ;4 P0101 - MAF SENSOR ERROR |
| | | | ;5 NOT USED |
| | | | ;6 P0705 - TRANS. RANGE SW. ERROR |
| | | | ;7 NOT USED |
| 938A | 001E | 52 | ; MALFFLG4 HISTORY MALF FLAG WORD 4 |
| | | | ;0 P0712 - TFT SENSOR CKT HIGH (LOW TEMP.) |
| | | | ;1 P0713 - TFT SENSOR CKT LOW (HIGH TEMP.) |
| | | | ;2 P0132 - HO2S SYSTEM RICH |
| | | | ;3 P0131 - HO2S SYSTEM LEAN |
| | | | ;4 P0325 - KS CIRCUIT ERROR |
| | | | ;5 P1350 - IC EST BYPASS ERROR |
| | | | ;6 P0342 - CAM SENSOR ERROR |
| | | | ;7 P0740 - TCC ERROR |
| 938C | 001F | 53 | ; MALFFLG5 HISTORY MALF FLAG WORD 5 |
| | | | ;0 P1650 - QDM B ERROR |
| | | | ;1 P1405 - EGR3 SOLENOID ERROR |
| | | | ;2 P1404 - EGR2 SOLENOID ERROR |
| | | | ;3 P1403 - EGR1 SOLENOID ERROR |
| | | | ;4 P1361 - IC EST ERROR |
| | | | ;5 P1623 - PROM ERROR |
| | | | ;6 NOT USED |
| | | | ;7 NOT USED |
| 938E | 0020 | 54 | ; MALFFLG6 HISTORY MALF FLAG WORD 6 |
| | | | ;0 NOT USED |
| | | | ;1 NOT USED |
| | | | ;2 NOT USED |
| | | | ;3 NOT USED |
| | | | ;4 NOT USED |
| | | | ;5 NOT USED |
| | | | ;6 P1626 - PASSKEY II ERROR |
| | | | ;7 NOT USED |
| 9390 | 0021 | 55 | ; MALFFLG7 HISTORY MALF FLAG WORD 7 |
| | | | ;0 NOT USED |
| | | | ;1 NOT USED |

| | | | | |
|------|------|----|-------------------------------|---|
| | | | ;2 | NOT USED |
| | | | ;3 | P1530 - A/C PRESS. SENSOR ERROR |
| | | | ;4 | NOT USED |
| | | | ;5 | NOT USED |
| | | | ;6 | NOT USED |
| | | | ;7 | NOT USED |
| 9392 | 0022 | 56 | ; MALFFLG8 | HISTORY MALF FLAG WORD 8 |
| | | | ;0 | NOT USED |
| | | | ;1 | P1629 - PASSKEY II SIGNAL ERROR |
| | | | ;2 | P1550 - SMC ENGAGEMENT ERROR |
| | | | ;3 | NOT USED |
| | | | ;4 | NOT USED |
| | | | ;5 | NOT USED |
| | | | ;6 | NOT USED |
| | | | ;7 | NOT USED |
| 9394 | 0013 | 57 | ; CURMALF1 | CURRENT MALF FLAG WORD 1 |
| | | | ;0 | P0123 - TPS VOLTAGE HIGH |
| | | | ;1 | NOT USED |
| | | | ;2 | P0341 - CAM SIGNAL ERROR |
| | | | ;3 | P0321 - IC 24X SIGNAL ERROR |
| | | | ;4 | P1630 - SYSTEM VOLTAGE ERROR |
| | | | ;5 | P0117 - ENGINE COOLANT TEMP. LOW |
| | | | ;6 | P0118 - ENGINE COOLANT TEMP. HIGH |
| | | | ;7 | P0134 - HO2S ERROR |
| 9396 | 0014 | 58 | ;"CURRENT" MALF STATUS FLAG 2 | |
| | | | ;B0 | NOT ENABLED |
| | | | ;B1 | NOT ENABLED |
| | | | ;B2 | NOT ENABLED |
| | | | ;B3 | 1 = DTC P1640 - QDM A MALF |
| | | | ;B4 | 1 = DTC P0113 - IAT HIGH TEMP |
| | | | ;B5 | 1 = DTC P0502 - NO VSS SIGNAL |
| | | | ;B6 | 1 = DTC P0112 - IAT LOW TEMP |
| | | | ;B7 | 1 = DTC P0122 - TPS LOW VOLTAGE |
| 9398 | 0015 | 59 | ;"CURRENT" MALF FLAG 3 | |
| | | | ;B0 | 1 = DTC P0703 - TCC BRAKE SW ERROR |
| | | | ;B1 | 1 = DTC P0501 - VSS INTERMITTENT SIGNAL |
| | | | ;B2 | 1 = DTC P0755 - SSB ERROR |
| | | | ;B3 | NOT ENABLED |
| | | | ;B4 | 1 = DTC P0101 - MAF, LOW GMS/SEC |
| | | | ;B5 | NOT ENABLED |
| | | | ;B6 | 1 = P0705 - TRANS RANGE SWITCH ERROR |
| | | | ;B7 | NOT ENABLED |
| 939A | 0016 | 60 | ;"CURRENT" MALF FLAG 4 | |
| | | | ;B0 | 1 = DTC P0712 - LOW TRANS TEMP |
| | | | ;B1 | 1 = DTC P0713 - HIGH TRANS TEMP |
| | | | ;B2 | 1 = DTC P0132 - O2 SENSOR RICH |
| | | | ;B3 | 1 = DTC P0131 - O2 SENSOR LEAN |
| | | | ;B4 | 1 = DTC P0325 - KNOCK SENSOR CKT ERROR |
| | | | ;B5 | 1 = DTC P1350 - IGNITION CONTROL BYPASS ERROR |
| | | | ;B6 | 1 = DTC P0342 - CAMSHAFT SENSOR ERROR |
| | | | ;B7 | 1 = DTC P0740 - TCC ERROR |
| 939C | 0017 | 61 | ;"CURRENT" MALF FLAG 5 | |
| | | | ;B0 | 1 = DTC P1650 - QDM B MALF |
| | | | ;B1 | 1 = DTC P1405 - EGR SOLN MALF |
| | | | ;B2 | 1 = DTC P1404 - EGR SOLN MALF |
| | | | ;B3 | 1 = DTC P1403 - EGR SOLN MALF |
| | | | ;B4 | 1 = DTC P1361 - IGN CONTROL ERROR |
| | | | ;B5 | 1 = DTC P1623 - PROM ERROR |
| | | | ;B6 | NOT ENABLED |
| | | | ;B7 | NOT ENABLED |
| 939E | 0018 | 62 | ;"CURRENT" MALF FLAG 6 | |
| | | | ;B0 | |
| | | | ;B1 | |
| | | | ;B2 | |

```

;B3
;B4
;B5
;B6      1 = P1626 - PASSKEY II ERROR
;B7
93A0      0019              63      ;"CURRENT" MALF FLAG 7
;B0
;B1
;B2
;B3      1 = A/C PRESSURE SENSOR MALF
;B4
;B5
;B6
;B7
93A2      001A              64      ;"CURRENT" MALF FLAG 8
;B0
;B1      1 = P1629 - PASSKEY II SIGNAL ERROR
;B2
;B3
;B4
;B5
;B6
;B7
93A4      0333              65      ;RESERVED FOR TESTING STACK OVERWRITE
93A6      40E4              66      ;BPW MSB
93A8      40E4              67      ;BPW LSB
; MSEC = N/65.536

```

```

;-----

```

```

; MODE 1 (TRANSMIT FIXED DATA STREAM MESSAGE 1)

```

```

;   ALDL REQUEST:
;   - MESSAGE ID           = $F4
;   - MESSAGE LENGTH       = $57
;   - MODE NUMBER          = $01
;   - MESSAGE NUMBER       = $01
;   - CHECKSUM
;

```

```

;   THE ECM WILL RESPOND WITH THE FOLLOWING MESSAGE:

```

```

;   - MESSAGE ID           = $F4
;   - MESSAGE LENGTH       = $60
;   - MODE NUMBER          = $01
;   - DATA BYTE 1
;   .
;   .
;
;   - DATA BYTE 10
;   - CHECKSUM
;-----

```

```

93AA      0000              ;INITIAL VALUE FOR ALDL MESSAGE SCHEDULER
93AC      F4                ;MESSAGE ID
93AD      80                ;FLAG = USE ROM
93AE      0B                ;11 = NUMBER OF OUTPUT DATA BYTES
93AF      023A              ;ADDRESS OF OUTPUT MESSAGE BUFFER
93B1      022B              ;ADDRESS OF INPUT MESSAGE BUFFER
93B3      F4                ;DEVICE ID
93B4      00AA              1  ;TPS %
93B6      00A6              2  ;COOLANT TEMP
93B8      00AC              3  ;NTRPMX - RPM = 25N
93BA      0094              4  ;QDMMW - QDM FAULT AND OTHER STATUS BITS
;0      NOT USED
;1      0 = QDM FAULT1*
;2      0 = QDM FAULT2*
;3      USED IN THE XIRQ ROUTINE
;4      1 = A/C WAS ON THIS CRANK

```

```

;5      1 = LOW OIL LIGHT ON
;6      NOT USED
;7      A/C CLUTCH STATUS 1 = ON
93BC    0333      5      ;RESERVED FOR TESTING STACK OVERWRITE
93BE    00D9      6      ;PIDMW2 - IDLE AIR CONTROL MODE WORD
;0      1 = A/C SLUGGING TEMPERATURES MET
;1      1 = MOTOR HAS BEEN AT 0 DURING RESET
;2      1 = MOTOR RESET IN PROGRESS
;3      1 = AIRFLOW > COMMAND AIR FLOW
;4      1 = RESET COMPLETE
;5      1 = DON'T ADD HOT STRT IDLE SPD OFFSET
;6      1 = STEPPER MOTOR CRUISE ENGAGED
;7      1 = ASYNC A.E. ENABLED
93C0    009B      7      ; MWFA      FUEL AIR MODE WORD
;0      1 = TCC SLIP IS OK FOR A/C ENGAGEMNT
;1      1 = DECEL FUEL CUTOFF ENABLED
;2      BL ADDRESS CHANGE FLAG (1 = CHANGE)
;3      DELAY BLM UPDATE (1 = BLM ADDR CHNG)
;4      1 = REVRSE->DRV CHANGE (OR DRV->REVRSE CHANGE)
;5      PE FLAG (1 = PE IS ACTIVE)
;6      HIGH LIMIT FUEL CUTOFF ENABLED
;7      1 = INCREMENT ODOMETER
93C2    009C      8      ; MWFA1     FUEL AR MODE WORD 1
;0      200 MSEC. OLD P/N BIT FROM FMDINST 1 = P/N
;1      LEARN CONTROL ENABLE FLAG (1=ENABLE,0=DISABLE)
;2      1 = FATI FILTER ACTIVE (TIMEOUT AFR)
;3      1 = O/L F/A WAS RICHER THAN KCLRATIO
;4      1 = FATC FILTER ACTIVE (CLNT TEMP AFR)
;5      1 = FORCE LOW PULSE RESULT OPEN LOOP
;6      RICH-LEAN FLAG (1=RICH,0=LEAN)
;7      CLOSED LOOP FLAG (1=CLOSED LOOP, 0=OPEN LOOP)
93C4    0096      9      ; LCCPMW     LCC & CCP MODE WORD
;0      1 = CCP PURGE ON
;1      1 = FAN1 ON
;2      1 = RPM HIGH IN P/N
;3      1 = TCC LOCKED
;4      1 = FAN2 ON
;5      1 = POWER STEERING PRESURE SWITCH CRAMP
;6      1 = USE TABLE F7MAXTRQ FOR 3RD GEAR TORQ MGMT
;7      1 = NORMAL A/C REQ HAS TURNED A/C ON AT STARTUP
93C6    0084      10     ; SDPDW1     SERIAL DATA PACKED DISCRETE WORD 1
;0      1 = ERROR FREE TRANSMISSION ON UART LINK
;1      1 = ALDL XMIT NEEDED (RESPONSE TO A RX'D MSG)
;2      1 = CLEAR MALF CODES
;3      1 = ALDL MODE 8 - DISABLE NORMAL COMMUNICATIONS
;4      1 = DO CHECKSUM ONLY
;5      1 = ALDL TESTER IN CONTROL OF LINK
;6      1 = CLEAR NON-VOLITAL RAM (REQUESTED BY HUD)
;7      1 = MODE 4

```

```

;-----

```

```

; MODE 2 SELECTABLE MEMORY DUMP - FROM ANHT HAC

```

```

; ALDL DEV MUST REQUEST MODE 2 BY TRANSMITTING THE FOLLOWING MSG TO THE ECM

```

```

;      DEVICE ID      =      $F4
;      MSG LENGTH     =      $58      (3 + 85)
;      MODE           =      $02
;      START ADD MSB  =      $aa
;      START ADD LSB  =      $aa
;      CKSUM          =      $nn
;

```

```

; THE ECM WILL RESPOND WITH

```

```

;      DEVICE ID      =      $F4
;      MSG LEGTH     =      $96      (65 + 85)
;      MODE          =      $02

```

```

;      ADD CONTENTS= $dd
;      "      "      TILL END OF OUTPUT BYTES
;      CKSUM      =      $29
;-----
93C8  0000      ;NEXT MESSAGE ENTRY ADDRESS
93CA  F4      ;MESSAGE ID
93CB  40      ;FLAG = USE RAM BUFFER
93CC  41      65      ;NUMBER OF OUTPUT DATA BYTES
93CD  0204      ;ADDRESS OF OUTPUT MESSAGE BUFFER
          ;IS SERIAL DATA MODE WD
93CF  022B      ;ADDRESS OF INPUT MESSAGE BUFFER
93D1  F4      ;DEVICE ID

MODE 3
-----
93D2  0000      ;NEXT MESSAGE ENTRY ADDRESS
93D4  F4      ;MESSAGE ID
93D5  40      ;FLAG = USE RAM BUFFER
93D6  5F      95      ;NUMBER OF OUTPUT DATA BYTES
93D7  0202      ;ADDRESS OF OUTPUT MESSAGE BUFFER
          ;L0202 IS ADDRESS OF ALDL MESSAGE SCHEDULER
93D9  022B      ;ADDRESS OF INPUT MESSAGE BUFFER
93DB  F4      ;DEVICE ID

MODE 4
-----
93DC  0000      ;NEXT MESSAGE ENTRY ADDRESS
93DD  F4      ;MESSAGE ID
93DE  40      ;FLAG = USE RAM BUFFER
93DF  01      ;NUMBER OF OUTPUT DATA BYTES
93E0  0226      ;ADDRESS OF OUTPUT MESSAGE BUFFER
93E2  022B      ;ADDRESS OF INPUT MESSAGE BUFFER
93E3  F4

93E4  0000      ;NEXT MESSAGE ENTRY ADDRESS

MODE 5
-----
93E6  F4      ;MESSAGE ID
93E7  80      ;FLAG = USE ROM
93E8  01      ;NUMBER OF OUTPUT DATA BYTES
93E9  023A      ;ADDRESS OF OUTPUT MESSAGE BUFFER
93EB  022B      ;ADDRESS OF INPUT MESSAGE BUFFER
93EC  F4
93ED  0000      ;? MODE 5 RESPONSE BYTE
93EF  F4      ;MESSAGE ID

MODE 6
-----
93F0  80      ;FLAG = USE ROM
93F4  01      ;NUMBER OF OUTPUT DATA BYTES
93F5  023A      ;ADDRESS OF OUTPUT MESSAGE BUFFER
93F7  022B      ;ADDRESS OF INPUT MESSAGE BUFFER
93F9  F4

MODE 7
-----
93FA  0000      ;NEXT MESSAGE ENTRY ADDRESS
93FC  F4      ;MESSAGE ID
93FD  80      ;FLAG = USE ROM
93FE  01      ;NUMBER OF OUTPUT DATA BYTES
93FF  023A      ;ADDRESS OF OUTPUT MESSAGE BUFFER
9401  022B      ;ADDRESS OF INPUT MESSAGE BUFFER
9403  F4

```

MODE 8 - DISABLE NORMAL COMMUNICATIONS

```

-----
9404 0000 ;NEXT MESSAGE ENTRY ADDRESS
9406 F4 ;MESSAGE ID
9407 80 ;FLAG = USE ROM
9408 01 ;NUMBER OF OUTPUT DATA BYTES
9409 023A ;ADDRESS OF OUTPUT MESSAGE BUFFER
940B 022B ;ADDRESS OF INPUT MESSAGE BUFFER
940D F4

940E FFFF ;ALLOW MODE 4 FOR THIS LONG
9410 0001 ;PREVENT MODE 4 FOR THIS LONG

9412 0x00,0x0A 10 10 SECONDS ;TIME LIMIT BEFORE ALLOWING CRUISE CONTROL
9414 0xFF 255 6375 RPM ;RPM THRESHOLD TO DISABLE CRUISE CONTROL
9415 0x14 20 500 RPM ;RPM THRESHOLD TO DESABLE CRUISE CONTROL
9416 0x16 22 22 MPH ;MPH THRESHOLD TO ALLOW CRUISE CONTROL
9417 0x11 17 17 MPH ;MPH THRESHOLD TO ALLOW CRUISE CONTROL
9418 0x69 105 105 MPH ;MPH THRESHOLD TO DISABLE CRUISE CONTROL
9419 0x00
941A 0x20
941B 0x20 ;TRANS TEMP FILTER COEFFICIENT
941C 0xE3
941D 0xD5

```

START OF F31 TRANNY PARAMETERS

```

941E 0x0A
941F 0x07
9420 0xB3 179 70 %TPS ;
9421 0x78 120 3000 RPM ;
9422 0x80 128 50 %TPS ;TPS THRESHOLD TO FORCE 3RD GEAR SHIFT
9423 0xC8 200 5000 RPM ;RPM THRESHOLD TO FORCE 3RD GEAR SHIFT
9424 0x4C 76 950 RPM ;
9425 0x06 6 75 RPM ;OFFSET

```

| RPM VS | %TPS | | | |
|--------|------|-----|------|--|
| HEX | DEC | RPM | %TPS | |
| 9426 | 0x48 | 72 | 1800 | |
| | 0x58 | 88 | 2200 | |
| | 0x70 | 112 | 2800 | |
| | 0x90 | 144 | 3600 | |
| | 0xB0 | 176 | 4400 | |
| | 0xC0 | 192 | 4800 | |
| | 0xC8 | 200 | 5000 | |

```

942D 0xC8 ;NOT USED
942E 0xC8 ;NOT USED
942F 0x00 ;
9430 0x00 ;
9431 0x00 ;
9432 0x00 ;
9433 0x00 ;
9434 0x19 ;
9435 0x26 ;
9436 0x23 ;
9437 0x14 ;
9438 0x23 ;
9439 0x17,0x40 5952 5952 RPM ;
943B 0x07

```

943C 0x06
943D 0x06

THESE ARE VS RPM

| | | | |
|------|-----------|------|----------|
| 943E | 0x17,0x40 | 5952 | 5952 RPM |
| 9440 | 0x17,0x70 | 6000 | 6000 RPM |
| 9442 | 0x14,0x1E | 5150 | 5150 RPM |
| 9444 | 0x17,0x40 | 5952 | 5952 RPM |
| 9446 | 0x17,0x70 | 6000 | 6000 RPM |
| 9448 | 0x14,0x1E | 5150 | 5150 RPM |

944A 0xFE,0x00
944C 0xFF,0x34
944E 0xFF,0xFF

| | | |
|------|------|------|
| 9450 | 0xFB | %TPS |
| 9451 | 0xF7 | |
| 9452 | 0x07 | |
| 9453 | 0x06 | |
| 9454 | 0x06 | |

| | | | |
|------|-----------|------|----------|
| 9455 | 0x17,0x40 | 5952 | 5952 RPM |
| 9457 | 0x17,0x70 | 6000 | 6000 RPM |
| 9459 | 0x14,0x1E | 5150 | 5150 RPM |
| 945B | 0x17,0x40 | 5952 | 5952 RPM |
| 945D | 0x17,0x70 | 6000 | 6000 RPM |
| 945F | 0x14,0x1E | 5150 | 5150 RPM |

9461 0xFE,0x00
9463 0xFF,0x34
9465 0xFF,0xFF

9467 0xFB
9468 0xF7

9469 0x0B,0x0B,0x0D,0x10,0x13,0x16,0x1A,0x1D,0x20,0x23,0x26,0x28,0x29,0x29,0x29,0x29,0x29
947A 0x16,0x16,0x17,0x1D,0x23,0x27,0x2B,0x2F,0x34,0x38,0x3C,0x41,0x45,0x49,0x4D,0x50,0x50
948B 0x30,0x30,0x30,0x32,0x35,0x43,0x50,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68

949C 0x09,0x09,0x09,0x09,0x09,0x09,0x0B,0x0E,0x10,0x13,0x15,0x17,0x1A,0x1C,0x1E,0x21,0x23
94AD 0x13,0x13,0x13,0x13,0x13,0x17,0x1C,0x21,0x25,0x2A,0x2F,0x34,0x39,0x3D,0x42,0x47,0x4C
94BE 0x26,0x26,0x26,0x26,0x26,0x26,0x26,0x2D,0x32,0x37,0x5E,0x5E,0x5E,0x5E,0x5E,0x5E

94CF 0x23,0x23,0x28,0x2D,0x37,0x3A,0x3E,0x4B,0x50,0x50,0x50,0x50,0x50,0x50,0x50,0x50
94E0 0x1E,0x1E,0x1E,0x1E,0x1F,0x21,0x24,0x27,0x2A,0x32,0x37,0x3C,0x4C,0x4C,0x4C,0x4C

94F1 0x00,0x00,0x00,0x00,0x13,0x16,0x1A,0x1D,0x20,0x23,0x26,0x28,0x29,0x29,0x29,0x29,0x29
9502 0x16,0x16,0x17,0x1D,0x23,0x27,0x2B,0x2F,0x34,0x38,0x3C,0x41,0x45,0x49,0x4D,0x50,0x50
9513 0x30,0x30,0x30,0x32,0x35,0x43,0x50,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68

9524 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x1A,0x1C,0x1E,0x21,0x23
9535 0x13,0x13,0x13,0x13,0x13,0x17,0x1C,0x21,0x25,0x2A,0x2F,0x34,0x39,0x3D,0x42,0x47,0x4C
9546 0x26,0x26,0x26,0x26,0x26,0x26,0x26,0x2D,0x32,0x37,0x5E,0x5E,0x5E,0x5E,0x5E,0x5E

9557 0x16,0x16,0x17,0x1D,0x23,0x26,0x2C,0x2F,0x33,0x36,0x3A,0x3E,0x43,0x47,0x4A,0x4E,0x50
9568 0x31,0x31,0x31,0x37,0x41,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68

9579 0x13,0x13,0x13,0x13,0x13,0x17,0x1C,0x21,0x25,0x2A,0x2F,0x34,0x39,0x3D,0x42,0x47,0x4B
958A 0x28,0x28,0x28,0x28,0x28,0x28,0x28,0x2A,0x2F,0x34,0x3E,0x5E,0x5E,0x5E,0x5E,0x5E,0x5E

END OF F31 TRANS PARAMETERS

START OF TCC PWM PARAMETERS

| | | |
|------|-----------|------------------|
| 959B | 0x24 | |
| 959C | 0x28 | |
| 959D | 0x0A | |
| 959E | 0x14 | ;STORED IN L0199 |
| 959F | 0x14 | |
| 95A0 | 0x00 | |
| 95A1 | 0x16 | |
| 95A2 | 0x00,0x0A | |
| 95A4 | 0x0A | |
| 95A5 | 0x14 | |
| 95A6 | 0x05 | |
| 95A7 | 0x05 | |
| 95A8 | 0x00 | |
| 95A9 | 0x04 | |
| 95AA | 0x10 | |
| 95AB | 0x04 | |
| 95AC | 0x10 | |
| 95AD | 0x04 | |
| 95AE | 0x10 | |
| 95AF | 0x04 | |
| 95B0 | 0x10 | |
| 95B1 | 0x04 | |
| 95B2 | 0x10 | |
| 95B3 | 0x04 | |
| 95B4 | 0x10 | |
| 95B5 | 0x7E | |
| 95B6 | 0x85 | |
| 95B7 | 0x00 | |
| 95B8 | 0x1C | |

THESE ARE VS %TPS (FOR TCC PWM)

| | | |
|------|--|-----------------------------------|
| 95B9 | 0x16,0x16,0x16,0x1A,0x1F,0x23,0x27,0x2C,0x30,0x34,0x38,0x3D,0x41,0x45,0x4A,0x4E,0x50 | |
| 95CA | 0x13,0x13,0x13,0x13,0x13,0x13,0x13,0x16,0x19,0x1C,0x1F,0x26,0x2D,0x35,0x3C,0x43,0x4A | |
| 95DB | 0x16,0x16,0x16,0x1A,0x1F,0x23,0x27,0x2C,0x30,0x34,0x38,0x3D,0x41,0x45,0x4A,0x4E,0x50 | |
| 95EC | 0x13,0x13,0x13,0x13,0x13,0x13,0x13,0x16,0x19,0x1C,0x1F,0x26,0x2D,0x35,0x3C,0x43,0x4A | |
| 95FD | 0x20,0x20,0x20,0x2C,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF | |
| 960E | 0x24,0x24,0x24,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF | |
| 961F | 0x30,0x30,0x30,0x35,0x3A,0x48,0x55,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A | |
| 9630 | 0x31,0x31,0x31,0x37,0x41,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68 | |
| 6941 | 0x1E,0x1E,0x1E,0x20,0x22,0x23,0x27,0x2B,0x30,0x35,0x64,0x64,0x64,0x64,0x64,0x64 | |
| 9652 | 0x22,0x22,0x22,0x22,0x22,0x23,0x27,0x2B,0x30,0x35,0x55,0xFF,0xFF,0xFF,0xFF,0xFF | |
| 9663 | 0x2A,0x2A,0x2A,0x2A,0x2A,0x2D,0x32,0x37,0x41,0x64,0x64,0x64,0x64,0x64,0x64,0x64 | |
| 9674 | 0x2A,0x2A,0x2A,0x2A,0x2A,0x2A,0x2E,0x34,0x3C,0x64,0x64,0x64,0x64,0x64,0x64,0x64 | |
| 9685 | 0x20,0x20,0x20,0x2C,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF | |
| 9696 | 0x30,0x30,0x30,0x35,0x3A,0x48,0x55,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A | |
| 96A7 | 0x1E,0x1E,0x1E,0x20,0x22,0x23,0x27,0x2B,0x30,0x35,0x64,0x64,0x64,0x64,0x64,0x64 | |
| 96B8 | 0x2A,0x2A,0x2A,0x2A,0x2A,0x2D,0x32,0x37,0x41,0x64,0x64,0x64,0x64,0x64,0x64,0x64 | |
| 96C9 | 0x25 | |
| 96CA | 0xFF | |
| 96CB | 0x14 | |
| 96CC | 0x8E | |
| 96CD | 0x00 | |
| 96CE | 0xFF | |
| 96CF | 0x05 | |
| 96D0 | 0xFF | |
| 96D1 | 0x00 | |
| 96D2 | 0x00 | 0 0 ;? INITIAL TCC PWM DUTY CYCLE |
| 96D3 | 0x83 | ; |

```

96D4    0x93          147                      ;vs FILTERED TCC SLIP RPM
96D5    0x8E
96D6    0x80                      ;TCC SLIP RPM FILTER COEFFICIENT
96D7    0x02
96D8    0x04
96D9    0x01
96DA    0x14
96DB    0x14
96DC    0xFE          254          99.2 % DC      ;? TCC PWM DUTY CYCLE

TCC PWM DUTY CYCLE DELTA VS %TPS WHEN IN 2ND OR 3RD GEAR
-----
96DD    0x07,0x08,0x0D,0x12,0x14,0x16,0x18,0x1A,0x1C,0x1C,0x1C,0x1C,0x1C,0x1C,0x1C,0x1C

TCC PWM DUTY CYCLE DELTA VS %TPS WHEN IN 3RD GEAR
-----
96EE    0x04,0x05,0x09,0x0D,0x11,0x13,0x15,0x1A,0x1C,0x1C,0x1C,0x1C,0x1C,0x1C,0x1C,0x1C

TCC PWM DUTY CYCLE DELTA VS %TPS WHEN NOT IN 3RD GEAR
-----
96FF    0x06,0x06,0x08,0x09,0x0A,0x0C,0x10,0x12,0x25,0x25,0x25,0x25,0x25,0x25,0x25,0x25

TCC PWM DUTY CYCLE DELTA VS %TPS
-----
9710    0x7F,0x6F,0x5F,0x4F,0x3F,0x2F,0x1F,0x0F,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00

9721    0x50          80          31.2 %DC      ;? TCC PWM DUTY CYCLE

2D TABLE VS %TPS
-----
9722    0x0D,0x0D,0x0D,0x0D,0x0F,0x0F,0x10,0x14,0x16,0x16,0x19,0x25,0x25,0x25,0x25,0x25

2D TABLE VS %TPS
-----
9733    0x64,0x6A,0x71,0x7D,0x83,0x89,0x90,0x96,0x9C,0x9C,0x9C,0x9C,0x9C,0x9C,0x9C,0x9C

2D TABLE VS %TPS
-----
9744    0x1E,0x1F,0x20,0x21,0x22,0x23,0x24,0x25,0x25,0x25,0x25,0x25,0x25,0x25,0x25,0x25

9755    0x00

2D TABLE VS LV8
-----
9756    0x08          TABLE LENGTH (9 VALUES)
          0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00

2D TABLE VS LV8
-----
9760    0x08          TABLE LENGTH (9 VALUES)
          0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00

? TCC PWM DUTY CYCLE VS LV8
-----
976A    0x08          TABLE LENGTH (9 VALUES)
          0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00

END OF TCC PWM PARAMETERS
#####

9774    0x02                      ;
9775    0x26,0x9D                  ;ROAD SPEED CONSTANT

```

```

9777 0x17 ;
9778 0x00,0x00
977A 0x00,0x00 00 0 ;INITIAL VALUE FOR L0072
977C 0x00 00 0 ;TIME LIMIT
977D 0x40 ;FILTER COEFFICIENT MPH -> L0310
977E 0x2A 42 ;
977F 0x4F 79
9780 0x68 104
9781 0x2A 42
9782 0x4F 79
9783 0x68
9784 0x14,0x0E,0x00,0x14,0x80,0x14,0x00
978B 0x48
978C 0x01 ;

987D 0x04,0xD6 ;MULTIPLIER FOR L0269 WHEN 4TH GEAR
978F 0x06,0xDC ;MULTIPLIER FOR L0269 WHEN 3RD GEAR
9791 0x0A,0xC1 ;MULTIPLIER FOR L0269 WHEN 2ND GEAR
9793 0x14,0x09 ;MULTIPLIER FOR L0269
; RESULT -> TURBINE RPM

9795 0xC0 ;MPH FILTER COEFFICIENT -> L00BD

;-----
; SHIFT LIGHT PARAMETERS
;-----
9796 0x00 ;FMPH - VEH SPD THRESH FOR P/N
9797 0x00,0x00 ;N/V RATIO INDICATING 5TH GEAR HIGH VALUE
9799 0x00,0x00 ;N/V RATIO INDICATING 5TH GEAR LOW VALUE
979B 0x00,0x00 ;N/V RATIO INDICATING 4TH GEAR HIGH VALUE
979D 0x00,0x00 ;N/V RATIO INDICATING 4TH GEAR LOW VALUE
979F 0x00,0x00 ;N/V RATIO INDICATING 3RD GEAR HIGH VALUE
97A1 0x00,0x00 ;N/V RATIO INDICATING 3RD GEAR LOW VALUE
97A3 0x00,0x00 ;N/V RATIO INDICATING 2ND GEAR HIGH VALUE
97A5 0x00,0x00 ;N/V RATIO INDICATING 2ND GEAR LOW VALUE

97A7 0x00 ;MAXIMUM SHIFT LITE ON TIME
97A8 0x00 ;TURN OFF E-LITE IF NEG D-TPS > THIS
97A9 0x00 ;COOLANT THRESHOLD FOR E-LITE ENABLE
97AA 0x00 ;HYST FOR MINIMUM TPS FOR SHFT LITE ON
97AB 0x00 ;RPM HYST FOR MIN RPM FOR SHIFT LITE ON
97AC 0x00 ;RPM ABOVE WHICH LIGHT IS ALWAYS ON
97AD 0x00 ;FMPH
97AE 0x00 ;%TPS

F47G1ST VS RPM/25
97AF 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00

97BA 0x00 ;MINIMUM RPM FOR LIGHT ON, 1ST GEAR
97BB 0x00 ;MINIMUM TPS FOR LIGHT ON, 1ST GEAR
97BC 0x00 ;LIGHT ON DELAY TIME, 1ST GEAR
97BD 0x00 ;TPS HYSTERESIS FOR LIGHT ON, 1ST GEAR
97BE 0x00 ;LIGHT ON DELAY TIME, 1ST GEAR

F47G2ND VS RPM/25
97BF 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00

97CA 0x00 ;MINIMUM RPM FOR LIGHT ON, 2ND GEAR
97CB 0x00 ;MINIMUM TPS FOR LIGHT ON, 2ND GEAR
97CC 0x00 ;LIGHT ON DELAY TIME, 2ND GEAR
97CD 0x00 ;TPS HYSTERESIS FOR LIGHT ON, 2ND GEAR
97CE 0x00 ;LIGHT ON DELAY TIME, 2ND GEAR

F47G3RD VS RPM/25

```

| | | |
|------|------|--|
| 97DA | 0x00 | ;MINIMUM RPM FOR LIGHT ON, 3RD GEAR |
| 97DB | 0x00 | ;MINIMUM TPS FOR LIGHT ON, 3RD GEAR |
| 97DC | 0x00 | ;LIGHT ON DELAY TIME, 3RD GEAR |
| 97DD | 0x00 | ;TPS HYSTERESIS FOR LIGHT ON, 3RD GEAR |
| 97DE | 0x00 | ;LIGHT ON DELAY TIME, 3 RD GEAR |

```
97DF 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
```

| | | |
|------|------|---|
| 97EA | 0x00 | ;MINIMUM RPM FOR LIGHT ON, 4 TH GEAR |
| 97EB | 0x00 | ;MINIMUM TPS FOR LIGHT ON, 4TH GEAR |
| 97EC | 0x00 | ;LIGHT ON DELAY TIME, 4TH GEAR |
| 97ED | 0x00 | ;TPS HYSTERESIS FOR LIGHT ON, 4TH GEAR |
| 97EE | 0x00 | ;LIGHT ON DELAY TIME, 4 TH GEAR |

[illegible]

| | |
|----------|-------|
| <i>i</i> | |
| <i>i</i> | ----- |
| 97FA | 0x00 |
| 97FB | 0x00 |
| 97FC | 0x00 |
| 97FD | 0x00 |
| 97FE | 0x00 |
| 97FF | 0x00 |
| 9800 | 0x00 |

| | HEX | DEC | TIME | TEMP |
|------|------|-----|------------|------|
| 9801 | 0x3C | 60 | 60 SECONDS | -40 |
| | 0x3C | 60 | 60 | -28 |
| | 0x2D | 45 | 45 | -16 |
| | 0x1E | 30 | 30 | -4 |
| | 0x14 | 20 | 20 | 8 |
| | 0x11 | 17 | 17 | 20 |
| | 0x0E | 15 | 15 | 32 |
| | 0x0C | 12 | 12 | 44 |
| | 0x0A | 10 | 10 | 56 |
| | 0x0A | 10 | 10 | 68 |
| | 0x0A | 10 | 10 | 80 |
| | 0x0A | 10 | 10 | 92 |
| | 0x0C | 12 | 12 | 104 |
| | 0x14 | 20 | 20 | 116 |
| | 0x14 | 20 | 20 | 128 |
| | 0x14 | 20 | 20 | 140 |
| | 0x14 | 20 | 20 | 152 |

[illegible]

| | | |
|------|-----|------|
| 0xC8 | 200 | 5000 |
| 0xC8 | 200 | 5000 |
| 0xC8 | 200 | 5000 |
| 0xC8 | 200 | 5000 |
| 0xC8 | 200 | 5000 |
| 0xC8 | 200 | 5000 |
| 0xC8 | 200 | 5000 |

MPH VS STARTUP COOLANT - RESULT GOES INTO L01E6

9823 0x00,0x00,0x00,0x00 2ND GEAR
0x00,0x00,0x00,0x00 3RD GEAR
0x00,0x00,0x00,0x00 4TH GEAR

COLD ENGINE RPM FUEL CUT VS SU COOLANT

| | HEX | DEC | RPM | TEMP | |
|------|------|-----|----------|------|-------|
| 982F | 0x9C | 156 | 3900 RPM | -40 | DEG C |
| | 0x9C | 156 | 3900 RPM | -28 | |
| | 0xB0 | 176 | 4400 RPM | -16 | |
| | 0xD8 | 216 | 5400 RPM | -4 | |
| | 0xD8 | 216 | 5400 RPM | 8 | |
| | 0xD8 | 216 | 5400 RPM | 20 | |
| | 0xD8 | 216 | 5400 RPM | 32 | |
| | 0xD8 | 216 | 5400 RPM | 44 | |
| | 0xD8 | 216 | 5400 RPM | 56 | |
| | 0xD8 | 216 | 5400 RPM | 68 | |
| | 0xD8 | 216 | 5400 RPM | 80 | |
| | 0xD8 | 216 | 5400 RPM | 92 | |
| | 0xD8 | 216 | 5400 RPM | 104 | |
| | 0xD8 | 216 | 5400 RPM | 116 | |
| | 0xD8 | 216 | 5400 RPM | 128 | |
| | 0xD8 | 216 | 5400 RPM | 140 | |
| | 0xD8 | 216 | 5400 RPM | 152 | |

COLD ENGINE RPM FUEL RESTORE VS SU COOLANT

| | HEX | DEC | RPM | TEMP | |
|------|------|-----|----------|------|-------|
| 9840 | 0x9B | 155 | 3875 RPM | -40 | DEG C |
| | 0x9B | 155 | 3875 | -28 | |
| | 0xAF | 175 | 4375 | -16 | |
| | 0xD7 | 215 | 5375 | -4 | |
| | 0xD7 | 215 | 5375 | 8 | |
| | 0xD7 | 215 | 5375 | 20 | |
| | 0xD7 | 215 | 5375 | 32 | |
| | 0xD7 | 215 | 5375 | 44 | |
| | 0xD7 | 215 | 5375 | 56 | |
| | 0xD7 | 215 | 5375 | 68 | |
| | 0xD7 | 215 | 5375 | 80 | |
| | 0xD7 | 215 | 5375 | 92 | |
| | 0xD7 | 215 | 5375 | 104 | |
| | 0xD7 | 215 | 5375 | 116 | |
| | 0xD7 | 215 | 5375 | 128 | |
| | 0xD7 | 215 | 5375 | 140 | |
| | 0xD7 | 215 | 5375 | 152 | |

9851 0x32 50 ;
9852 0x10 16
9853 0x49
9854 0xAD
9855 0x00,0x64
9857 0x1E 30 3 SECONDS ;OIL LIGHT TURN ON DELAY TIME

9858 0x01,0x80,0x80,0x1E,0x3C,0x02,0x00
985F .byte 0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00

```

9867      .byte  0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
986F      .byte  0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
9877      .byte  0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
987F      .byte  0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
9887      .byte  0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
988F      .byte  0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
9897      .byte  0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00

```

```

;-----
; 8 x 16 MULTIPLY
; A CONTAINS AN 8 BIT MULTIPLIER
; X POINTS TO THE 16 BIT MULTIPLICAND ON THE STACK
;-----

```

```

989D      L989D:psha                                ;SAVE 8 BIT MULTIPLIER
989E      ldab   0x01,x                              ;GET LSB OF MULTIPLICAND
98A0      mul                                ;A * B
98A1      stab   0x01,x                              ;STORE LSB OF PRODUCT
98A3      ldab   0x00,x                              ;LOAD MSB OF MULTIPLICAND
98A5      staa   0x00,x                              ;SAVE MSB OF PRODUCT
98A7      pula                                ;RESTORE MULTIPLIER
98A8      mul                                ;A * B
98A9      addb   0x00,x                              ;ROUND
98AB      adca   #0x00                              ;ROUND
98AD      rol    0x01,x                              ;MUL BY 2
98AF      rolb                                ;PRIOR RESULTS LOW BYTE
98B0      rola                                ;PRIOR RESULTS HIGH BYTE
98B1      bcc    L98B6                                ;BRANCH IF NO OVERFLOW, ELSE
98B3      ldd    #0xFFFF                            ;LOAD 65535
98B6      L98B6:std  0x00,x                          ;SAVE PRODUCT
98B8      rts                                ;RETURN

```

```

;-----
; 8 X 16 MULTIPLY ROUTINE - IDENTICAL TO $8F
;-----

```

```

98B9      L98B9:psha
98BA      ldab   0x01,x
98BC      mul
98BD      adca   #0x00
98BF      pulb
98C0      psha
98C1      ldaa   0x00,x
98C3      mul
98C4      pshx
98C5      tsx
98C6      addb   0x02,x
98C8      adca   #0x00
98CA      pulx
98CB      ins
98CC      rts

```

```

;-----
; 8 X 16 MULTIPLY - ONLY USED ONCE
; B = L0269: MULTIPLICAND
; X CONTAINS A 16 BIT MULTIPLIER
; LEAVE WITH RESULT IN ACCD
; X CONTAINS
;-----

```

```

98CD      L98CD:pshb                                ;SAVE L0269 ON STACK
98CE      ldaa   0x00,x                              ;MSB OF MULTIPLIER
98D0      mul                                ;A * B
98D1      pula                                ;GET 8 BIT MULTIPLICAND
98D2      pshb                                ;SAVE LSB OF PRODUCT ON STACK
98D3      ldab   0x01,x                              ;GET LSB OF MULTIPLIER

```

```

98D5          mul                    ;A * B
98D6          tsx                    ;X POINTS TO STACK
98D7          adda    0x00,x          ;ADD MSB OF
98D9          bcc     L98DE            ;BRANCH IF NO OVERFLOW, ELSE
98DB          ldd     #0xFFFF         ;LOAD 65535
98DE  L98DE:ins                    ;INCREMENT STACK POINTER
98DF          rts                    ;

;-----
; DIVIDE - ONLY USED ONCE
; X CONTAINS DENOMINATOR
; D CONTAINS NUMERATOR
; LEAVE WITH RESULT IN X REG
;-----
98E0  L98E0:pshx                    ;
98E1          idiv                    ;
98E2          pshx                    ;SAVE QUOTIENT ON STACK
98E3          puly                    ;GET QUOTIENT INTO Y REG
98E5          pulx                    ;
98E6          fdiv                    ;
98E7          pshx                    ;
98E8          pshy                    ;
98EA          pulb                    ;
98EB          pulx                    ;
98EC          pula                    ;
98ED          tstb                    ;
98EE          beq     L98F5            ;
98F0          ldx     #0xFFFF         ;
98F3          bra     L98F9            ;

98F5  L98F5:tsta                    ;
98F6          bpl     L98F9            ;
98F8          inx                    ;
98F9  L98F9:rts                    ;

;-----
; 16 X 16 MULTIPLY, LIMITED TO 65535
;-----
98FA  L98FA:pshb                    ;
98FB          pshb                    ;
98FC          pshx                    ;SAVE
98FD          psha                    ;
98FE          tsx                    ;
98FF          ldaa    0x02,x          ;
9901          mul                    ;
9902          adca    #0x00            ;ROUND
9904          staa    0x04,x          ;STORE
9906          ldaa    0x01,x          ;
9908          ldab    0x03,x          ;
990A          mul                    ;
990B          addb    0x04,x          ;
990D          adca    #0x00            ;
990F          std     0x03,x          ;
9911          ldaa    0x02,x          ;
9913          ldab    0x00,x          ;
9915          mul                    ;
9916          addd    0x03,x          ;ADD TO FINAL RESULT
9918          std     0x03,x          ;
991A          pula                    ;
991B          pulb                    ;
991C          bcs     L9929            ;BRANCH IF OVERFLOWED, ELSE
991E          mul                    ;MUL
991F          addb    0x03,x          ;ADD LSB TO MSB
9921          adca    #0x00            ;ROUND

```

```

9923         bne      L9929                ;BRANCH IF NOT = ZERO, ELSE
9925         tba                      ;TRANSFER B TO A REG
9926         pulx                      ;
9927         pulb                      ;
9928         rts

9929  L9929:ldd      #0xFFFF
992C         pulx
992D         ins
992E         rts

;-----
; 16 X 16 MULTIPLY - IDENTICAL TO $8F
; X = MULTIPLICAND
; ACCD = MULTIPLIER
;-----
992F  L992F:pshx                ;SAVE
9930         pshb                ;SAVE LSB OF MULTIPLIER
9931         pshx                ;SAVE
9932         psha                ;SAVE MSB OF MULTIPLIER
9933         tsx                  ;X POINTS TO STACK
9934         ldaa      0x02,x      ;GET LSB OF
9936         mul
9937         std       0x04,x
9939         ldaa      0x01,x
993B         ldab      0x03,x
993D         mul
993E         addb      0x04,x
9940         adca      #0x00
9942         std       0x03,x
9944         ldaa      0x00,x
9946         ldab      0x02,x
9948         mul
9949         addd      0x03,x
994B         std       0x03,x
994D         rola
994E         anda      #0x01
9950         staa      0x02,x
9952         pula
9953         pulb
9954         mul
9955         addd      0x02,x
9957         pulx
9958         pulx
9959         rts

;-----
; 3D LOOKUP ROUTINE - IDENTICAL TO $8F
;-----
995A  L995A:pshy
995C         pshb
995D         pshx
995E         suba      0x00,x      ;SUBTRACT MINIMUM
9960         bcc      L9963        ;BRANCH IF NO UNDERFLOW, ELSE
9962         clra                ;CLEAR
9963  L9963:subb      0x01,x      ;SUBTRACT MINIMUM
9965         bcc      L9968        ;BRANCH IF NO UNDERFLOW, ELSE
9967         clrb                ;CLEAR
9968  L9968:pshx                ;TRANSFER TO ACCY
9969         puly                ;
996B         psha                ;
996C         ldaa      #0x10
996E         mul
996F         pshb

```

| | | |
|------|------------|--------|
| 9970 | tab | |
| 9971 | abx | |
| 9972 | pula | |
| 9973 | pulb | |
| 9974 | psha | |
| 9975 | ldaa | #0x10 |
| 9977 | mul | |
| 9978 | pshb | |
| 9979 | ldab | 0x02,y |
| 997C | mul | |
| 997D | abx | |
| 997E | pshx | |
| 997F | ldab | 0x02,y |
| 9982 | abx | |
| 9983 | tsy | |
| 9985 | ldd | 0x03,x |
| 9987 | sba | |
| 9988 | ldab | 0x03,y |
| 998B | bcc | L9993 |
| 998D | nega | |
| 998E | mul | |
| 998F | adca | 0x03,x |
| 9991 | bra | L9999 |
| 9993 | L9993:mul | |
| 9994 | adca | #0x00 |
| 9996 | nega | |
| 9997 | adda | 0x03,x |
| 9999 | L9999:pulx | |
| 999A | psha | |
| 999B | ldd | 0x03,x |
| 999D | sba | |
| 999E | ldab | 0x03,y |
| 99A1 | bcc | L99A9 |
| 99A3 | nega | |
| 99A4 | mul | |
| 99A5 | adca | 0x03,x |
| 99A7 | bra | L99AF |
| 99A9 | L99A9:mul | |
| 99AA | adca | #0x00 |
| 99AC | nega | |
| 99AD | adda | 0x03,x |
| 99AF | L99AF:pulb | |
| 99B0 | psha | |
| 99B1 | sba | |
| 99B2 | ldab | 0x02,y |
| 99B5 | bcc | L99BE |
| 99B7 | nega | |
| 99B8 | mul | |
| 99B9 | adca | 0x01,y |
| 99BC | bra | L99C5 |
| 99BE | L99BE:mul | |
| 99BF | adca | #0x00 |
| 99C1 | nega | |
| 99C2 | adda | 0x01,y |
| 99C5 | L99C5:ins | |
| 99C6 | pulx | |
| 99C7 | pulx | |
| 99C8 | pulb | |
| 99C9 | puly | |
| 99CB | rts | |

```

;-----
; 2D LOOKUP ROUTINE
;-----
99CC L99CC:pshx ;
99CD pshb ;
99CE ldab 0x00,x ;GET 256/SPACING VALUE
99D0 inx ;START OF DATA
99D1 bra L99DB ;

;-----
; 2D LOOKUP WITH OFFSET (B REG = OFFSET)
;-----
99D3 L99D3:sba ;SUBTRACT OFFSET
99D4 bcc L99D7 ;BRANCH IF NO UNDERFLOW, ELSE
99D6 clra ;CLEAR A

;-----
; 2D LOOKUP, NO OFFSET
;-----
99D7 L99D7:pshx ;
99D8 pshb ;
99D9 ldab #0x10 ;
99DB L99DB:mul ;A * SPACING VALUE
99DC pshb ;SAVE LSB
99DD tab ;COPY MSB TO B REG
99DE abx ;ADD TO INDEX
99DF ldd 0x00,x ;GET BOTH VALUES
99E1 sba ;SUBTRACT
99E2 pulb ;GET LSB
99E3 bcc L99EB ;BRANCH IF NO UNDERFLOW, ELSE
99E5 nega ;INVERT
99E6 mul ;A * B
99E7 adca 0x00,x ;ROUND
99E9 bra L99F1 ;

99EB L99EB:mul ;A * LSB
99EC adca #0x00 ;ROUND
99EE nega ;INVERT
99EF adda 0x00,x ;ADD
99F1 L99F1:pulb ;RESTORE
99F2 pulx ;RESTORE
99F3 rts ;

;-----
; LAG FILTER ROUTINE, 8 OR 16 BIT
; A CONTAINS NEW VARIABLE TO BE FILTERED - 8 OR 16 BIT
; X REG CONTAINS OLD FILTERED VALUE - 16 BIT
; Y REG CONTAINS FILTER COEFFICIENT ADDRESS - 8 BIT FRACTION
; NEW FILTERED VALUE IS STORED AT X ADDRESS
;-----
99F4 L99F4:clrb ;
99F5 L99F5:subd 0x00,x ;ACCD - RAM CONTENTS
99F7 pshb ;SAVE LSB ON STACK
99F8 bcc L9A01 ;BRANCH IF ACCD >= RAM ADDRESS, ELSE
99FA ldab 0x00,x ;LOAD RAM ADDRESS
99FC subb 0x00,y ;SUBTRACT TABLE VARIABLE
99FF stab 0x00,x ;STORE
9A01 L9A01:ldab 0x00,y ;LOAD TABLE VARIABLE
9A04 mul ;A * B
9A05 addd 0x00,x ;ADD
9A07 std 0x00,x ;STORE
9A09 pula ;GET LSB INTO A REG
9A0A ldab 0x00,y ;LOAD MSB OF TABLE ADDRESS
9A0D mul ;A * B

```

```

9A0E      adca    0x01,x          ;ADD LSB AND ROUND
9A10      tab                    ;TRANSFER TO B REG
9A11      ldaa    0x00,x          ;LOAD RAM LOCATION
9A13      adca    #0x00           ;ROUND
9A15      std     0x00,x          ;STORE NEW FILTERED VALUE
9A17      rts

;-----
; A/D READ - SIMILAR TO $8D
; CALL WITH CHANNEL NUMBER IN A REG
; RETURN WITH DATA IN A REG
;-----

9A18      L9A18:sei                    ;HOLD INTERRUPTS
9A19      ldx     #0x4000          ;I/O DATA REGISTER
9A1C      bclr    0x02,x,#0x08     ;I/O PORT
9A1F      staa    0x00,x          ;WR TO DATA REG
9A21      bclr    0x01,x,#0x80     ;CLEAR BIT 7 I/O CNTL REG
9A24      pshb                    ;DELAY
9A25      mul
9A26      mul
9A27      mul
9A28      pulb
9A29      ldaa    0x00,x          ;GET DATA FROM REG
9A2B      staa    L046B           ;STORE DATA
9A2E      idiv                    ;DELAY
9A2F      mul                    ;
9A30      ldx     #0x4000          ;INDEX I/O DATA REG
9A33      ldaa    #0xB0           ;TEST CHANNEL
9A35      staa    0x00,x          ;WR TO DATA REG
9A37      bclr    0x01,x,#0x80     ;CLEAR BIT 7 I/O CNTL REG
9A3A      des                    ;DELAY
9A3B      mul
9A3C      mul
9A3D      mul
9A3E      pulb
9A3F      ldaa    0x00,x          ;GET DATA FROM REG
9A41      bset    0x02,x,#0x08     ;SET BIT 3 I/O PORT
9A44      brset   *L0086,#0x01,L9A49 ;
9A48      cli                    ;CLEAR AND ALLOW INTERRUPTS
9A49      L9A49:rts

;-----
; X = I/O REG (SPI DATA REGISTER)
; IDENTICAL TO $5B
;-----

9A4A      L9A4A:ldaa    *L0089          ;
9A4C      eora    #0x02           ;TOGGLE BIT 1
9A4E      staa    *L0089          ;STORE
9A50      ldx     #0x4000          ;INDEX I/O DATA REG
9A53      L9A53:bclr    *L0089,#0x80     ;CLEAR BIT 7
9A56      bra     L9A5B

9A58      L9A58:bset    *L0089,#0x80     ;SET BIT 7
9A5B      L9A5B:bset    0x02,x,#0x04     ;4002 - ? CHIP SELECT TO IN AND OUT SPI DEV
9A5E      ldaa    *L0089          ;LOAD DATA
9A60      staa    0x00,x          ;WR TO DATA REG
9A62      bclr    0x01,x,#0x80     ;CLEAR BIT 7 4001 I/O CNTL REG
9A65      pshb                    ;DELAY
9A66      mul
9A67      mul
9A68      mul
9A69      pulb
9A6A      ldaa    0x00,x          ;LOAD DATA
9A6C      bclr    0x02,x,#0x04     ;CLEAR BIT 2 4002 CPU DATA LATCH

```

```

9A6F      rts

;-----
; I/O ROUTINE - TX/RX ON I/O PORT
; X = 4000 (I/O PORT)
; A AND B CONTAIN THE DATA TO BE TRANSMITTED
;-----
9A70  L9A70:bset    0x02,x,#0x10      ;BIT 4 4002
9A73      staa     0x00,x            ;WR DATA TO I/O PORT
9A75      bclr     0x01,x,#0x80      ;CLEAR BIT 7 4001 - START TRANSMISSION
9A78      pshb
9A79      mul
9A7A      mul
9A7B      mul
9A7C      pulb
9A7D      ldaa     0x00,x            ;LOAD DATA
9A7F      stab     0x00,x            ;WR DATA TO I/O PORT
9A81      bclr     0x01,x,#0x80      ;CLEAR BIT 7 4001
9A84      staa     *L008D            ;STORE DATA
9A86      pshb
9A87      mul
9A88      mul
9A89      pulb
9A8A      pshb
9A8B      pulb
9A8C      ldaa     0x00,x            ;LOAD DATA
9A8E      staa     *L008C            ;STORE DATA
9A90      bita     #0x40              ;TEST BIT 6
9A92      beq      L9A97            ;BRANCH IF NO CAM REF PULSE, ELSE
9A94      bset     *L0093,#0xC0      ;SET CAM REF PULSE OCCURED
9A97  L9A97:bclr    0x02,x,#0x10      ;CLEAR BIT 4 4002
9A9A      rts                      ;RETURN

;-----
; THIS ONLY CALLED ONCE
; X = L4000 - I/O REG (SPI DATA REGISTER)
; A = 0x04
;-----
9A9B  L9A9B:staa    0x00,x            ;0x04 GOES INTO L4000
9A9D      bclr     0x01,x,#0x80      ;CLEAR BIT 7 SPI CNTL REG
9AA0      pshb
9AA1      mul
9AA2      mul
9AA3      mul
9AA4      pulb
9AA5      ldaa     0x00,x            ;LOAD DATA FROM I/O REG
9AA7      rts

;-----
; DATA RECEIVE - 8192 BAUD SERIAL DATA CODE - JUMP HERE FROM IRQ
;-----
9AA8  L9AA8:ldab    L4008            ;CPU TX/RX CNTL REG
9AAB      ldaa     L4009            ;CPU RX CNTL REG
9AAE      ldx      L0202            ;SCI DATA MSB TBL INDEX
9AB1      bitb     #0x0E            ;%00001110 - CHECK FOR ERRORS
9AB3      beq      L9AB8            ;BRANCH IF NO ERROR, ELSE
9AB5      jmp      L9C18            ;JUMP OVER RECEIVE CODE

9AB8  L9AB8:tab
9AB9      addb     L0201            ;RUNNING CK SUM ON SPI RX DATA
9ABC      stab     L0201            ;SAVE NEW SCI DATA CKSUM
9ABF      ldab     L0200            ;LOAD SCI DATA BYTE COUNTER
9AC2      bne      L9AE7            ;BRANCH IF NOT Z, ELSE
9AC4      ldx      L928C            ;INDEX

```

```

9AC7          brclr  *L0085,#0x10,L9ACE          ;BRANCH IF NOT MULTI SENSOR FAILURE, ELSE
9ACB          ldx    #0x9278                      ;INDEX
9ACE  L9ACE: cmpa   0x02,x                        ;DEVICE CODE 928E (0x03) OR
                                           ;DEVICE CODE 927A (0x01)
9AD0          beq    L9ADD                        ;BRANCH IF CORRECT DEVICE ID, ELSE
9AD2          ldx    0x00,x                        ;NEXT MESSAGE ADDRESS 928C OR
                                           ;NEXT MESSAGE ADDRESS 9278
9AD4          bne    L9ACE                        ;BRANCH IF MESSAGE ADDRESS NOT Z, ELSE
9AD6          cmpa   #0xE0                        ;224
9AD8          bcs    L9B3F                        ;BRANCH IF < 224 (CLR EVERYTHING - BAD MSG)
9ADA          bset   *L0084,#0x10                ;SET DO CKSUM ONLY
9ADD  L9ADD: stx    L0202                        ;SCI DATA MSG TBL INDEX

;-----
; ENABLE SCI RECEIVE INTERRUPTS
; B0  1 = CNTR INTERRUPT (BREAK ENABLED)
; B2  1 = RX ENABLED
; B5  1 = RDRF AND OR INTERRUPTS ENABLED
;-----

9AE0          ldaa   #0x25                        ;%00100101
9AE2          staa   L4007                        ;CPU TX/RX CNTL REG
9AE5          bra    L9B2D                        ;INCREMENT SCI BYTE COUNTER AND RETURN

9AE7  L9AE7: decb                               ;DEC BYTE COUNTER
9AE8          bne    L9AF9                        ;BRANCH IF NOT = ZERO, ELSE
9AEA          suba   #0x55                        ;SUB OFF MSG LENGTH BIAS
9AEC          bcs    L9B3F                        ;BRANCH IF < OFFSET (-> ERROR), ELSE
9AEE          cmpa   #0x22                        ;34 BYTES?
9AF0          bls    L9AF4                        ;BRANCH IF <= 34 BYTES, ELSE
9AF2          ldaa   #0x22                        ;LOAD 34
9AF4  L9AF4: staa   L0249                        ;SERIAL INPUT DATA MSG LENGTH
9AF7          bra    L9B2D                        ;INCREMENT SCI BYTE COUNTER AND RETURN

9AF9  L9AF9: decb                               ;DEC BYTE COUNTER
9AFA          cmpb   L0249                        ;SERIAL DATA MSG LENGTH
9AFD          bcc    L9B33                        ;BR IF LENGTH NZ, ELSE
9AFF          brset  *L0084,#0x10,L9B2D          ;BRANCH IF DO CKSUM ONLY, ELSE
9B03          tstb                               ;TEST B
9B04          bne    L9B27                        ;BRANCH IF NOT Z, ELSE
9B06          brset  *L0085,#0x10,L9B27          ;BRANCH IF MULTI SENSOR FAILURE, ELSE
9B0A          ldx    #0x92FB                      ;INDEX DATA RECEIVE TABLE ADDRESS
9B0D          cmpa   #0x0A                        ;MODE 10 ?
9B0F          bhi    L9B2D                        ;BRANCH IF > MODE 10, ERROR
9B11          cmpa   #0x07                        ;MODE 7 ?
9B13          bcc    L9B22                        ;BRANCH IF >= MODE 7, ELSE
9B15          cmpa   #0x04                        ;MODE 4 ?
9B17          bhi    L9B2D                        ;BRANCH IF MODE 5,6, ELSE
9B19          pshb                               ;SAVE BYTE COUNT ON STACK
9B1A          tab                               ;MODE NUM TO B REG
9B1B          lslb                               ;
9B1C          abx                               ;ADD MODE NUM TO ADDR
9B1D          pulb                               ;GET BYTE COUNT FROM STACK
9B1E          ldx    0x00,x                        ;GET A VALUE FROM TABLE 92FB
9B20          bra    L9B24                        ;GO SAVE ADDRESS

9B22  L9B22: ldx    0x0A,x                        ;9305 -> 0x93E6, ADDRESS FOR MODE 5 ?
9B24  L9B24: stx    L0202                        ;STORE SCI DATA MSG TBL INDEX
9B27  L9B27: ldx    #0x0204                      ;INDEX SERIAL DATA MODE WORD
9B2A          abx                               ;ADD BYTE COUNTER TO INDEX
9B2B          staa   0x00,x                        ;STORE DATA BYTE
9B2D  L9B2D: inc    L0200                        ;INC SCI DATA BYTE COUNTER
9B30          jmp    L9C28                        ;RETURN

9B33  L9B33: ldab   L0201                        ;SCI DATA CKSUM

```

```

9B36      bne      L9B3F      ;BRANCH IF NOT Z, ELSE
9B38      brclr   *L0084,#0x10,L9B42 ;BRANCH IF NOT DO CKSUM ONLY, ELSE
9B3C      stab    L024A      ;STORE CKSUM
9B3F      L9B3F: jmp     L9C18 ;CLEAR EVERYTHING - BAD MESSAGE

9B42      L9B42: ldaa    0x02,x ;GET DEVICE ID CODE (L0204)
9B44      staa    L0248      ;SERIAL DATA DEVICE CODE
9B47      cmpa    L9277      ;ECM MESSAGE ID
9B4A      bne     L9B4F      ;BRANCH IF NOT $F4, ELSE
9B4C      stab    L024A      ;STORE CKSUM
9B4F      L9B4F: bset    *L0084,#0x01 ;SET ERROR FREE XMISSION ON UART LINK
9B52      ldab    L0249      ;SERIAL DATA MSG LENGTH
9B55      beq     L9BAA      ;BRANCH IF Z, ELSE
9B57      brset   *L0085,#0x10,L9B96 ;BRANCH IF MULTI SENSOR FAILURE, ELSE
9B5B      bset    *L0084,#0x20 ;SET ALDL TESTER IN CONTROL OF LINK
9B5E      ldaa    L0204      ;SERIAL DATA MODE WD
9B61      cmpa    #0x0A      ;MODE 10?
9B63      beq     L9B80      ;BRANCH IF MODE 10, ELSE
9B65      cmpa    #0x09      ;MODE 9? (ENABLE NORMAL COMMUNICATIONS)
9B67      beq     L9B86      ;BRANCH IF MODE 9, ELSE
9B69      cmpa    #0x08      ;MODE 8? (DISABLE NORMAL COMMUNICATIONS)
9B6B      beq     L9B8C      ;BRANCH IF MODE 8, ELSE
9B6D      cmpa    #0x01      ;MODE 1?
9B6F      bne     L9B92      ;BRANCH IF NOT MODE 1, ELSE
9B71      ldaa    L0205      ;
9B74      cmpa    #0x01      ;MESSAGE 1 REQUESTED ?
9B76      bne     L9BAA      ;BRANCH IF NOT MSG 1, ELSE
9B78      ldx     #0x93AA    ;INDEX MODE 1 MSG 1 TABLE
9B7B      stx     L0202      ;SCI DATA MSB TBL INDEX
9B7E      bra     L9BAA      ;

9B80      L9B80: bset    *L0084,#0x04 ;SET CLEAR MALF CODES
9B83      jmp     L9BAA      ;

9B86      L9B86: bclr   *L0084,#0x08 ;CLEAR ALDL MODE 8
9B89      jmp     L9BAA

9B8C      L9B8C: bset    *L0084,#0x08 ;SET ALDL MODE 8
9B8F      jmp     L9BAA

;-----
; TRANSFER INPUT MESSAGE TO MODE 4 BUFFER, IF MODE 4 REQUESTED
;-----
9B92      L9B92: cmpa    #0x04      ;MODE 4?
9B94      bne     L9BAA      ;BRANCH IF NOT MODE 4, ELSE
9B96      L9B96: ldy     0x07,x    ;
9B99      ldx     #0x0204      ;SERIAL DATA MODE WD
9B9C      L9B9C: ldaa    0x00,x    ;LOAD DATA
9B9E      staa    0x00,y    ;STORE STARTING AT
9BA1      inx
9BA2      iny
9BA4      decb
9BA5      bne     L9B9C      ;LOOP TILL DONE
9BA7      ldx     L0202      ;INDEX SCI DATA MSB TBL
9BAA      L9BAA: brset   *L0085,#0x10,L9BC0 ;BRANCH IF MULTI SENSOR FAILURE, ELSE
9BAE      brclr   *L0084,#0x20,L9BC0 ;BRANCH IF ALDL TESTER NOT IN CONTROL OF
; LINK, ELSE
9BB2      ldaa    L0204      ;SERIAL DATA MODE WD
9BB5      bne     L9BBD      ;BRANCH IF NOT ZERO, ELSE
9BB7      bclr    *L0084,#0x28 ;CLEAR BITS 3 AND 5, ALDL MODE 8 AND
;ALDL TESTER IN CONTROL OF LINK
9BBA      bclr    *L0084,#0x80 ;CLEAR MODE 4
9BBD      L9BBD: staa    022B      ;STORE INPUT MESSAGE BUFFER
9BC0      L9BC0: brset   0x04,x,#0x80,L9C18 ;BIT 7 L0206

```

```

;X MAY ALSO POINT TO 93AA, IN WHICH CASE
; 0x04,x,0x80 IS THE FLAG TO SELECT ROM
; VS RAM FOR TX DATA
9BC4      brclr  *L0084,#0x20,L9BF2      ;BRANCH IF ALDL TESTER NOT IN CONTROL OF
; LINK, ELSE
9BC8      brset  *L0085,#0x10,L9BF2      ;BRANCH IF MULTI SENSOR FAILURE, ELSE
9BCC      ldaa   L0204                    ;SERIAL DATA MODE WD
9BCF      cmpa   #0x07                    ;MODE 7?
9BD1      bne    L9BEF                    ;BRANCH IF NOT MODE 7, ELSE
9BD3      ldz    #0x92B6                  ;INDEX
9BD6      ldab   L0205                    ;WHICH TBL TO TX - MESSAGE ID
9BD9      cmpb   0x02,x                  ;92B8 - MESSAGE ID
9BDB      beq    L9BE6                    ;BRANCH IF CORRECT MESSAGE ID, ELSE
9BDD      ldz    #0x92C6                  ;INDEX
9BE0      cmpb   0x02,x                  ;92C8 - MESSAGE ID
9BE2      beq    L9BE6                    ;BRANCH IF CORRECT MESSAGE ID, ELSE
9BE4      bra    L9C18                    ;CLEAR EVERYTHING - BAD MESSAGE

9BE6      L9BE6:jsr  Lf744                  ;
9BE9      brclr  *L0085,#0x40,L9C18      ;BRANCH IF XMIT NOT IN WORK, ELSE
9BED      bra    L9C28                    ;RETURN

9BEF      L9BEF:bset  *L0084,#0x02        ;SET ALDL XMIT NEEDED
9BF2      L9BF2:ldaa  0x09,x              ;L020B
9BF4      bset   *L0085,#0x40            ;SET XMIT IN WORK
9BF7      staa   L400A                    ;WR TO CPU TX REG
9BFA      staa   L0201                    ;MESSAGE CKSUM
9BFD      ldab   #0x01                    ;LOAD 1
9BFF      stab   L0200                    ;STORE SCI DATA BYTE COUNTER
9C02      ldd    L3DFC                    ;
9C05      orab   #0x04                    ;SET BIT 2
9C07      pshx                      ;DELAY
9C08      pulx                      ;
9C09      std    L3DFC                    ;STORE ? 2ND CPU CNTL REG

;-----
; FORCE SCI IDLE BYTE TX - FROM ANHT HACK
;-----
9C0C      ldaa   #0x81                    ;%10000001
9C0E      staa   L4007                    ;CPU TX/RX CNTL REG
9C11      ldaa   #0x89                    ;%10001001
9C13      staa   L4007                    ;CPU TX/RX CNTL REG
9C16      bra    L9C28                    ;RETURN

;-----
; BAD MESSAGE
;-----
9C18      L9C18:clra                      ;
9C19      clrb                      ;
9C1A      std    L0202                    ;SCI DATA MSG TBL INDEX
9C1D      std    L0200                    ;SCI DATA BYTE COUNTER
9C20      bclr   *L0084,#0x10            ;CLEAR DO CKSUM ONLY

;-----
; ALLOW 8192 SCI RX INTERRUPTS/SET SLEEP MODE
; B0 1 = CNTR INTERRUPT (BREAK ENABLED)
; B1 1 = RX WAKE UP BIT ENABLED
; B2 1 = RX ENABLED
; B5 1 = RDRF AND OR INTERRUPTS ENABLED
;-----
9C23      ldaa   #0x27                    ;%00100111
9C25      staa   L4007                    ;CPU TX/RX CNTL REG
9C28      L9C28:rti

```

```

;-----
; TX ROUTINE - 8192 BAUD SCI CODE - JUMP HERE FROM INTERNAL IRQ ROUTINE
;-----
9C29 L9C29:ldx      L0202                ;INDEX SCI DATA MSG TBL INDEX
9C2C      ldab     L0200                ;SCI DATA BYTE COUNTER
9C2F      decb     ;DEC COUNTER - 2ND BYTE TO BE SENT?
9C30      bne      L9C54                ;BRANCH IF NOT, ELSE
9C32      ldaa     0x04,x              ;LOAD L0206
9C34      brclr   *L0084,#0x02,L9C4D   ;BRANCH IF ALDL XMIT NOT NEEDED, ELSE
9C38      ldab     L0204                ;SERIAL DATA MODE WD
9C3B      subb     #0x02                ;MODE 2
9C3D      bls      L9C4D                ;BRANCH IF <= MODE 2, ELSE
9C3F      cmpb     #0x02                ;COMPARE 2
9C41      beq      L9C4D                ;BRANCH IF MODE 4, ELSE
9C43      ldaa     L0249                ;LOAD SERIAL DATA MSG LENGTH
9C46      cmpb     #0x05                ;MODE 7 ?
9C48      beq      L9C4D                ;BRANCH IF MODE 7, ELSE
9C4A      decb     ;DECREMENT MSG LENGTH
9C4B      lsra     ;DIV BY 2
9C4C      inca     ;INC MSG LENGTH
9C4D L9C4D:staa     L0249                ;SERIAL DATA MSG LENGTH
9C50      adda     #0x55                ;SERIAL BYTE CNT BIAS
9C52      bra      L9CC6                ;GO WR TO SCI TX REG

9C54 L9C54:decb     ;DEC SCI DATA BYTE COUNTER
9C55      bne      L9C5E                ;BRANCH IF NOT Z, ELSE
9C57      ldaa     L0204                ;SERIAL DATA MODE WD
9C5A      brset   *L0084,#0x02,L9CC6   ;BRANCH IF ALDL XMIT NEEDED, ELSE
9C5E L9C5E:cmpb     L0249                ;SERIAL DATA MSG LENGTH
9C61      bcc      L9CC0                ;BR IF CNT > SERIAL DATA MSG LENGTH, ELSE
9C63      brset   *L0085,#0x80,L9CB4   ;BRANCH IF 2ND BYTE XMIT WAITING
9C67      brset   0x03,x,#0x80,L9C76   ;L0205 - BIT 7 - OUTPUT IS ROM TABLE
9C6B      brset   0x03,x,#0x40,L9C81   ;L0205 - BIT 6 - OUTPUT IS RAM TABLE
9C6F      ld      0x05,x              ;GET ADDRESS
9C71      abx      ;ADD MSG COUNTER TO ADDRESS
9C72      ldaa     0x00,x              ;LOAD NEXT BYTE TO TRANSMIT
9C74      bra      L9CC6                ;WR TO SCI TX REG

9C76 L9C76:brclr   *L0084,#0x02,L9C7B   ;BRANCH IF ALDL XMIT NOT NEEDED, ELSE
9C7A      decb     ;DECREMENT BYTE COUNTER
9C7B L9C7B:lslb     ;MUL BY 2
9C7C      abx      ;ADD TO ADDRESS
9C7D      ld      0x0A,x              ;LOAD ADDRESS + 10
9C7F      bra      L9C9A                ;CK FOR DBL BYTE LD/ILLEGAL ADDRESS

9C81 L9C81:ldx      0x05,x              ;L0207
9C83      brclr   *L0084,#0x20,L9C8E   ;BRANCH IF ALDL TESTER NOT IN CONTROL, ELSE
9C87      ldaa     L0204                ;SERIAL DATA MODE WD
9C8A      cmpa     #0x02                ;MODE 2?
9C8C      beq      L9C95                ;BRANCH IF MODE 2, ELSE
9C8E L9C8E:lslb     ;MUL BY 2
9C8F      incb     ;INCREMENT
9C90      abx      ;ADD TO ADDRESS
9C91      ld      0x00,x              ;LOAD ADDRESS
9C93      bra      L9C9A

9C95 L9C95:inx      ;L0203
9C96      decb     ;DECREMENT BYTE COUNTER
9C97      ld      0x00,x              ;GET ADDRESS
9C99      abx      ;ADD BYTE COUNTER TO ADDRESS
9C9A L9C9A:cpx      #0x3000            ;CK ADDR WINDOW
9C9D      bcs      L9CBC                ;BRANCH IF ADDR < 0x3000, ELSE
9C9F      cpx      #0x6FFF            ;CK ADDR WINDOW
9CA2      bhi      L9CBC                ;BRANCH IF ADDR > 0x6FFF, ELSE

```

```

9CA4          pshx                      ;SAVE ADDRESS
9CA5          pula                      ;GET MSB
9CA6          anda    #0x03            ;
9CA8          psha                      ;PUT BACK MSB
9CA9          pulx                      ;GET BACK INTO X REG
9CAA          ldd    0x00,x            ;GET NEXT TWO BYTES
9CAC          stab    L024B            ;SAVE 2ND BYTE
9CAF          bset    *L0085,#0x80     ;SET 2ND BYTE WAITING
9CB2          bra     L9CC6            ;GO WR TO SCI TX REG

9CB4  L9CB4:bclr    *L0085,#0x80       ;CLEAR 2ND BYTE WAITING
9CB7          ldaa    L024B            ;LOAD SECOND BYTE
9CBA          bra     L9CC6            ;GO WR TO SCI TX REG

9CBC  L9CBC:ldaa    0x00,x            ;LOAD DATA AT THAT ADDRESS
9CBE          bra     L9CC6            ;GO WR TO SCI TX REG

9CC0  L9CC0:bne     L9CD4              ;
9CC2          ldaa    L0201            ;MESSAGE CKSUM
9CC5          nega                      ;INVERT
9CC6  L9CC6:staa    L400A              ;WR TO SCI TX REG
9CC9          adda    L0201            ;ADD RUNNING CKSUM
9CCC          staa    L0201            ;STORE NEW CKSUM
9CCF          inc     L0200            ;INC SCI DATA BYTE COUNTER
9CD2          bra     L9CED            ;RETURN

;-----
; LAST BYTE TX, CLEAN UP
;-----
9CD4  L9CD4:clra
9CD5          clrb
9CD6          bclr    *L0085,#0x80     ;CLEAR 2ND BYTE WAITING
9CD9          bclr    *L0084,#0x12     ;CLEAR BITS 1 AND 4
                                           ;ERROR FREE XMISSION ON UART LINK AND
                                           ;DO CKSUM ONLY
9CDC          std     L0200            ;CLEAR SCI DATA BYTE COUNTER
9CDF          std     L0202            ;SCI DATA MSB TBL INDEX
9CE2          ldaa    L4008            ;CPU TX/RX STATUS REG - CLEAR
9CE5          ldaa    L4009            ;CPU RX REG - CLEAR

;-----
; B0  1 = CNTR INTERRUPT (BREAK ENABLED)
; B6  1 = TRANSMIT COMPLETE INTERRUPT ENABLED
;-----
9CE8          ldaa    #0x41            ;%01000001
9CEA          staa    L4007            ;CPU TX/RX CNTL REG
9CED  L9CED:rti                      ;RETURN

;-----
; CLEAR RAM - CALLED FROM INIT ROUTINE AND RESET ECM
;-----
9CEE  L9CEE:ldaa    #0x8C              ;
9CF0          ldab    *L0056            ;FREE RUNNING IAC STEP COUNTER
9CF2          ldx     #0xFFD4            ;INDEX
9CF5          andb    #0x03            ;CLEAR BITS
9CF7          abx                      ;ADD TO ADDRESS
9CF8          ldab    #0x08            ;LOAD MASK
9CFA          orab    0x00,x            ;SET BITS
9CFC          ldx     #0x4000            ;I/O DATA REG
9CFF          std     0x01,x            ;I/O CONTROL REG
9D01          ldaa    #0x9F            ;
9D03          ldab    L9774            ;%00000010
9D06          andb    #0x0E            ;
9D08          orab    #0x90            ;SET BITS 4 AND 7

```

```

9D0A      std      0x03,x          ;STORE 4003
9D0C      bset     0x02,x,#0x04    ;SET BIT 2 I/O CNTL REG
9D0F      ldaa     #0x04           ;
9D11      jsr      L9A9B           ;WR TO SPI DATA REG & RX NEW SPI DATA
9D14      bclr     0x02,x,#0x04    ;CLEAR BIT 2 I/O PORT
9D17      staa     *L008A          ;STORE FMDBYTE1
9D19      bset     0x02,x,#0x80    ;SET BIT 7 I/O PORT
9D1C      ldd      #0x000A         ;CHECK ENG LIGHT ON
9D1F      ldz      #0x3F80         ;INDEX
9D22      std      0x7C,x          ;3FFC CPU CNTL REG
9D24      clrb                     ;

;-----
; CLEAR CPU REG'S, 3F80 - 3FBA AND 3FC0 - 3FFA
;-----
9D25      L9D25:std    0x00,x          ;3F80
9D27      std      0x40,x          ;3FC0
9D29      inx                     ;3F81
9D2A      inx                     ;3F82
9D2B      cpx      #0x3FBA         ;DONE ?
9D2E      bne      L9D25           ;LOOP TILL DONE

;-----
; CLEAR I/O REGS 3DC0 - 3DFA, AND RAM 0333 - 0082
;-----
9D30      ldz      #0x3DC0          ;LOAD 3DC0
9D33      L9D33:std    0x00,x          ;STORE IN 3DC0
9D35      inx                     ;
9D36      inx                     ;
9D37      cpx      #0x3DFA         ;
9D3A      bne      L9D33           ;LOOP TILL DONE, ELSE
9D3C      ldz      #0x7FFF         ;LOAD
9D3F      stx      L3FAC           ;STORE (SAME AS $5B)
9D42      ldz      #0x1B02         ;%00011011
                                   ;%00000010 - ENABLE SCI TRANSMIT
9D45      stx      L3DFC           ;STORE ? 2ND CPU CNTL REG
9D48      ldz      #0x02B1         ;
9D4B      L9D4B:staa   0x82,x          ;STORE IN 0x0333
9D4D      dex                     ;DECREMENT
9D4E      bne      L9D4B           ;LOOP TILL DONE
9D50      bset     *L0089,#0x04    ;ENABLE EST
9D53      rts                     ;

;-----
; THIS NOT CALLED
; (EXTERNAL IRQ POINTS HERE IN $5B)
;-----
9D54      bset     *L0010,#0x04    ;SET BIT 2
9D57      rti                     ;

;-----
; SWI VECTOR POINTS HERE
;-----
9D58      bset     *L0010,#0x01    ;SET BIT 0
9D5B      ldaa     L6000           ;? DELAY

;-----
; ILLEGAL OPCODE RESET VECTOR POINTS HERE
;-----
9D5E      ldaa     #0x08           ;
9D60      bra      L9D70           ;

;-----
; BUS RESET VECTOR POINTS HERE

```

```

;-----
9D62      ldaa    #0x10
9D64      bra     L9D70

;-----
; COP WATCHDOG RESET VECTOR POINTS HERE
;-----

9D66      ldaa    #0x20
9D68      bra     L9D70

;-----
; CLOCK MONITOR RESET VECTOR POINTS HERE
;-----

9D6A      ldaa    #0x40
9D6C      bra     L9D70

;-----
; RESET VECTOR POINTS HERE
; INITIALIZATION ROUTINE
;-----

9D6E      ldaa    #0x80                      ;
9D70      L9D70: oraa    *L0010                ;SET BITS ACCORDINGLY
9D72      staa    *L0010                ;STORE
9D74      lds     #0x03FF                    ;STACK ADDRESS
9D77      jsr     LA99A                    ;RTS - DELAY
9D7A      jsr     L9CEE                    ;CLEAR RAM
9D7D      L9D7D: ldx    *L0024                ;MALF WORD
9D7F      jsr     LF7EC                    ;ADD UP MALF FLAGS
9D82      cpx     *L0024                ;WAS THERE A CHANGE?
9D84      beq     L9D89                    ;BRANCH IF NOT, ELSE
9D86      jsr     LF775                    ;GO CLEAR RAM AND GET DEFAULT VALUES
9D89      L9D89: ldx    #0x3F94                ;INDEX
9D8C      ldd     #0x0001                    ;
9D8F      std     0x3A,x                    ;STORE IN 3FCE
9D91      mul                      ;DELAY
9D92      ldd     0x22,x                    ;LOAD 3FB6
9D94      addd    #0x7FFF                    ;ADD 32767
9D97      jsr     LF812                    ;GO STORE ACCD IN 3F94 - 3F9C
9D9A      ldd     #0xFB1A                    ;
9D9D      std     0x60,x                    ;STORE IN 3FF4
9D9F      ldd     #0x7B8A                    ;
9DA2      std     0x20,x                    ;STORE IN 3FB4
9DA4      ldx     L5000                    ;HUD
9DA7      cpx     #0x7E50                    ;
9DAA      bne     L9DAF                    ;BRANCH IF NOT Z, ELSE
9DAC      jsr     L500C                    ;UPDATE HUD

9DAF      L9DAF: ldx    #0x4000                ;I/O DATA REG ADDRESS
9DB2      ldd     #0x0008                    ;
9DB5      jsr     L9A70                    ;TRANSMIT ON I/O PORT
9DB8      ldd     #0x8001                    ;
9DBB      jsr     L9A70                    ;TRANSMIT ON I/O PORT
9DBE      bset    *L0086,#0x01                ;SET BIT 0
9DC1      ldd     #0x4000                    ;
9DC4      std     L3FF6                    ;STORE
9DC7      ldaa    #0xB0                    ;SEL CH #11
9DC9      jsr     L9A18                    ;A/D READ
9DCC      jsr     LE580                    ;IAT LOOKUP AND MALF CHECK
9DCF      ldaa    #0x03                    ;
9DD1      staa    *L00E1                    ;
9DD3      jsr     LE2E3                    ;COOLANT TEMP LOOKUP AND MALF CHECK
9DD6      ldaa    *L00A6                    ;LOAD CLNT TEMP (DEFAULTED)
9DD8      bne     L9DDF                    ;BRANCH IF NOT = ZERO, ELSE
9DDA      jsr     LE380                    ;DO COOLANT MALF CHECK

```

```

9DDD          ldaa    *L00A6
9DDF    L9DDF: staa    *L00A7
9DE1          staa    *L00F2
9DE3          ldaa    L86C6
9DE6          staa    *L00A9
9DE8          staa    L010C
9DEB          ldaa    #0x20
9DED          jsr     L9A18
9DF0          staa    *L00F6
9DF2          ldaa    L86C4
9DF5          staa    L0111
9DF8          staa    L02E6
9DFB          jsr     LF4BB
9DFE          staa    L030A
9E01          staa    L0110
9E04          ldaa    #0x50
9E06          jsr     L9A18
9E09          staa    *L0083
9E0B          cmpa    #0x5A
9E0D          bls     L9E19
9E0F          cmpa    #0xAB
9E11          bcc     L9E19
9E13          ldx     #0xFFFF
9E16          stx     L3DCC
9E19    L9E19: ldaa    *L00A6
9E1B          ldx     #0x8CB9
9E1E          jsr     L99D7
9E21          staa    *L00DE
9E23          jsr     LF402
9E26          brset   *L0012,#0x08,L9E2D
9E2A          bset    *L002F,#0xFF
9E2D    L9E2D: brset   *L008A,#0x01,L9E34
9E31          bset    *L009C,#0x01
9E34    L9E34: jsr     LF451
9E37          ldd     L3FC6
9E3A          std     L0126
9E3D          bset    *L00E1,#0x03
9E40          bclr    *L0017,#0x20
9E43          bclr    *L0017,#0x10
9E46          bclr    *L0014,#0x20
9E49          bclr    *L0015,#0x02
9E4C          bclr    *L0016,#0x23
9E4F          bset    *L00DA,#0x01
9E52          ldab    L84D5
9E55          stab    *L00B7
9E57          ldab    L8969
9E5A          stab    *L00B6
9E5C          ldaa    #0xFF
9E5E          staa    L0174
9E61          ldaa    #0x64
9E63          staa    L0171
9E66          ldaa    #0x0E
9E68          staa    *L0011

9E6A          bset    *L0086,#0x20
9E6D          ldaa    #0x80
9E6F          staa    *L00AF
9E71          staa    *L00B3
9E73          ldaa    #0x50
9E75          staa    L0119
9E78          ldd     #0xFFFF
9E7B          staa    L02B5
9E7E          staa    L027F
9E81          std     *L00C0

;LOAD CLNT TEMP (DEFAULTED)
;STORE FILTERED CTS
;STORE STARTUP COOLANT TEMP
;INITIAL O2 A/D COUNTS
;MINOR LOOP O2 A/D COUNTS (mV)
;FILTERED O2 A/D COUNTS (mV)
;SEL CH #2 - TPS SENSOR
;A/D READ
;STORE TPS A/D COUNTS
;INITIAL BASE TPS A/D COUNTS
;STORE
;STORE
;DO TPS LOAD ROUTINE
;STORE %TPS
;STORE %TPS
;SEL CH #5 - BATT
;A/D READ
;STORE BATT VOLTS
;9.0 VOLTS
;BRANCH IF <= 9.0 VOLTS, ELSE
;17.1 VOLTS
;BRANCH IF >= 17.1 VOLTS, ELSE
;
;
;LOAD CLNT TEMP (DEFAULTED)
;DESIRED IDLE VS TEMP IN PARK/NEUTRAL
;2D LOOKUP
;DESIRED IDLE RPM
;LOOKUP & STORE FUEL INJ OFFSET VS BATT
;BRANCH IF BIT 3, ELSE
;SET TO FATI DECAY TERM TO MAX
;BRANCH PARITY HIGH, ELSE
;SET P/N
;LOOKUP AND STORE L002C AND L0116
;PA2 COUNTER
;STORE OLD PA2 COUNTER
;SET BITS 0 AND 1
;CLEAR PROM ERROR
;CLEAR IGNITION CONTROL BYPASS MALF
;CLEAR VSS NO SIGNAL MALF
;CLEAR VSS INTERMITTENT MALF
;CLEAR TRANS TEMP AND IC BYPASS MALFS
;SET BIT 0
;0x05
;CYL COUNTER
;111 - STOICH AFR
;STORE AFR LSB
;
;
;100
;LAUNCH MODE SPARK CORRECTION TERM
;%00001110
;STORE MINOR LOOP COUNTER
;DO MAJOR LOOP 15 FIRST
;TURN A/C CLUTCH OFF
;LOAD 128
;STORE FILTERED BLM
;STORE INTEGRATOR
;80
;
;
;CRANK TO RUN PW MULTIPLIER
;SET REF PER TO MAX

```

```

9E83      bset      *L00A3,#0x08      ;SET OFF MODE
9E86      bclr      *L002A,#0x14      ;%00010100 EGR BIT STATUS FLAG
9E89      ldaa      *L0029            ;EGR MALF TIMER/COUNTER
9E8B      cmpa      L9134              ;12d
9E8E      bcs       L9E98              ;BRANCH IF < CAL, ELSE
9E90      clra      ;
9E91      clrb      ;
9E92      staa      *L002A            ;EGR BIT STATUS FLAG
9E94      std       *L0026            ;EGR MALF TIMER
9E96      std       *L0028            ;EGR MALF TIMER
9E98      L9E98: ldaa      L800F        ;OPTION WORD - TIS SET
9E9B      beq       L9EA3              ;BRANCH IF Z, ELSE
9E9D      bclr      *L0016,#0x80      ;CLEAR TCC ERROR
9EA0      bclr      *L0015,#0x04      ;CLEAR SHIFT SOLN B ERROR
9EA3      L9EA3: inc      L00E9        ;TRANNNY GEAR BYTE - SET TO 1ST GEAR
9EA6      ldaa      *L0071            ;COUNTER
9EA8      cmpa      L9145              ;3 ?
9EAB      bcc       L9EB0              ;BRANCH IF >= 3, ELSE
9EAD      bclr      *L0018,#0x20      ;CLEAR MALF - NOT ENABLED
9EB0      L9EB0: brclr    *L0095,#0x82,L9EB7 ;BRANCH IF HUD DEVICE NOT CONNECTED, ELSE
9EB4      jsr       L5868              ;UPDATE HUD

;-----
; CHECK IF MULTI SENSOR FAILURE
;-----

9EB7      L9EB7: jsr      LFAE0          ;LOOKUP TRANS TEMP
9EBA      bcs       L9ED5              ;CARRY SET IF TRANS TEMP ADC < 40 OR > 100
9EBC      ldaa      L0255              ;LOAD CTS A/D COUNTS
9EBF      cmpa      #0x05              ;5 ?
9EC1      bhi       L9ED5              ;BRANCH IF > 5, ELSE
9EC3      ldaa      *L00F6              ;TPS A/D COUNTS
9EC5      cmpa      #0xFC              ;4.94 VOLTS
9EC7      bcs       L9ED5              ;BRANCH IF < 4.94 VOLTS, ELSE
9EC9      ldaa      L0254              ;IAT INVERSE A/D COUNTS
9ECC      cmpa      #0x23              ;35d
9ECE      bcc       L9ED5              ;BRANCH IF >= 35d, ELSE
9ED0      bset      *L0085,#0x10      ;SET MULTI SENSOR FAILURE
9ED3      bra       L9F0A              ;

9ED5      L9ED5: ldaa      *L00A6          ;LOAD CLNT TEMP (DEFAULTED)
9ED7      cmpa      L8A36              ;STARTUP TEMP TO RESET BLMS
9EDA      bcc       L9EF1              ;BRANCH IF >= 110 DEG C, ELSE
9EDC      ldab      #0x11              ;17 LOOPS
9EDE      L9EDE: ldx      #0x8A37        ;ADDRESS
9EE1      abx              ;ADD TO ADDRESS
9EE2      ldaa      0x00,x              ;GET A VALUE FROM THE TABLE
9EE4      ldx      #0x0033              ;INDEX BLMS (- 1)
9EE7      abx              ;ADD B TO ADDRESS
9EE8      cmpa      0x00,x              ;COMPARE TABLE VALUE AND BLM
9EEA      bls       L9EEE              ;BRANCH IF TABLE VALUE <= RAM LOC, ELSE
9EEC      staa      0x00,x              ;STORE TABLE VALUE IN BLM RAM LOC
9EEE      L9EEE: decb      ;DECREMENT COUNTER
9EEF      bne       L9EDE              ;LOOP TILL DONE
9EF1      L9EF1: ldx      L8C6B          ;ENABLE IAC RESET IF RUN TIME EXCEEDS THIS
9EF4      bne       L9EF9              ;BRANCH IF NOT = ZERO, ELSE
9EF6      bset      *L0012,#0x40      ;SET IAC RESET ENABLED
9EF9      L9EF9: ldaa      L8019        ;OPTION WORD - TIS CLEAR
9EFC      bne       L9F01              ;BRANCH IF NOT = ZERO, ELSE
9EFE      bclr      *L0012,#0x04      ;CLEAR BIT 2
9F01      L9F01: bset      *L0089,#0x10 ;SET BIT 4
9F04      ldx      #0xDFFF              ;LOAD MAX DC
9F07      stx       L3DEA              ;STORE PWM OUTPUT
9F0A      L9F0A: bclr      *L0086,#0x01 ;CLEAR BIT 0
9F0D      ldx      #0x4000              ;INDEX

```

```

9F10      ldaa    0x08,x                ;4008 CPU TX/RX STATUS REG - CLEAR
9F12      ldaa    0x05,x                ;4005 REAL TIME COUNTER
9F14      adda    #0x02                  ;ADD 2
9F16      staa    0x06,x                ;4006 CAPTURE REG

;-----
; ALLOW 8192 SCI RX INTERRUPTS/SET SLEEP MODE
; B0  1 = CNTR INTERRUPT (BREAK ENABLED)
; B1  1 = RX WAKE UP BIT ENABLED
; B2  1 = RX ENABLED
; B5  1 = RDRF AND OR INTERRUPTS ENABLED
;-----

9F18      bset    0x07,x,#0x27          ;4007 CPU TX/RX CONTROL REG
9F1B      ldx     #0xFF00                ;TOGGLE COP
9F1E      stx     L400B                  ;COP WATCHDOG
9F21      jmp     LCCCA

;-----
; INTERNAL IRQ VECTOR POINTS HERE
; -> DETERMINE WHAT CAUSED THE INTERRUPT
;-----

9F24      bset    *L0010,#0x02          ;SET BIT 1
9F27      ldx     #0x4000                ;INDEX
9F2A      brset   0x08,x,#0x01,L9F63     ;4008 - BRANCH IF TIMER INTERRUPT
                                           ; (6.25 MSEC)
9F2E      brclr   0x07,x,#0x20,L9F39     ;4007 - BRANCH IF RX INTS NOT ENABLED
9F32      brclr   0x08,x,#0x20,L9F62     ;4008 - BRA IF RDR NOT FULL (NO NEW DATA)
9F36      jmp     L9AA8                  ;RX ROUTINE

9F39      L9F39:brclr 0x07,x,#0x80,L9F44 ;4007 - BRANCH IF TDRE INTERRUPT DISABLED
9F3D      brclr   0x08,x,#0x80,L9F62     ;4008 - BRANCH IF TDR NOT EMPTY
9F41      jmp     L9C29                  ;TX ROUTINE

9F44      L9F44:brclr 0x07,x,#0x40,L9F62 ;4007 - BRANCH IF TC INTERRUPT DISABLED
9F48      brclr   0x08,x,#0x40,L9F62     ;4008 - BRANCH IF TRANSMITTER BUSY, ELSE

;-----
; ALLOW 8192 SCI RX INTERRUPTS/SET SLEEP MODE
; B0  1 = CNTR INTERRUPT (BREAK ENABLED)
; B1  1 = RX WAKE UP BIT ENABLED
; B2  1 = RX ENABLED
; B5  1 = RDRF AND OR INTERRUPTS ENABLED
;-----

9F4C      ldaa    #0x27                  ;%00100111
9F4E      staa    0x07,x                ;4007 - CPU TX/RX CNTL REG
9F50      bclr    *L0085,#0x40          ;CLEAR XMIT IN WORK
9F53      bclr    *L0084,#0x02          ;CLEAR ALDL XMIT NEEDED
9F56      sei     ;HOLD INTERRUPTS
9F57      ldd     L3DFC                  ;LOAD 2ND MCU CNTL REG
9F5A      andb    #0xFB                  ;CLEAR BIT 2
9F5C      pshx    ;DELAY
9F5D      pulx    ;
9F5E      std     L3DFC                  ;STORE 2ND MCU CNTL REG
9F61      cli     ;CLEAR AND ALLOW INTERRUPTS
9F62      L9F62:rti

;-----
; 6.25 MSEC INTERRUPT ROUTINE
;-----

9F63      L9F63:brclr *L0085,#0x10,L9F6A ;BRANCH IF NOT MULTI SENSOR FAILURE, ELSE
9F67      jmp     LFAEC                  ;JUMP

9F6A      L9F6A:clr    L0480              ;
9F6D      ldab    0x06,x                ;4006 - CPU CAPT REG

```

| | | | |
|------|-------------|--------------------|---|
| 9F6F | addb | #0xCD | ;ADD 205 = 6.25 MSEC |
| 9F71 | stab | 0x06,x | ;4006 - CPU CAPT REG |
| 9F73 | stab | L026A | ;STORE TIME |
| 9F76 | ldaa | *L0011 | ;MINOR LOOP COUNTER |
| 9F78 | inca | | ;INCREMENT |
| 9F79 | brclr | *L0086,#0x0C,L9F82 | ;BRANCH IF NO TIMING ERRORS, ELSE |
| 9F7D | bset | *L0086,#0x04 | ;SET SERVICE INT EXECUTION EXCEEDED 6.25MS |
| 9F80 | bra | L9F88 | |
| | | | |
| 9F82 | L9F82:staa | L046F | ;ID OF MINOR LOOP EXCEEDING 6.25 MSEC |
| 9F85 | bset | *L0086,#0x08 | ;SET TIMING ERROR |
| 9F88 | L9F88:tsx | | ;TRANSFER STACK POINTER TO X |
| 9F89 | lds | #0x03FF | ;STACK ADDRESS |
| 9F8C | cpx | #0x03F7 | ;COMPARE |
| 9F8F | beq | L9F94 | ;BRANCH IF STACK POINTER = 0x03F7, ELSE |
| 9F91 | bset | *L0095,#0x01 | ;SET STACK OVERFLOW |
| 9F94 | L9F94:cli | | ;CLEAR AND ALLOW INTERRUPTS |
| 9F95 | cmpa | #0xA0 | ;160 - 1 SECOND YET? |
| 9F97 | bne | L9F9A | ;BRANCH IF NOT, ELSE |
| 9F99 | clra | | ;RESET MINOR LOOP COUNTER |
| 9F9A | L9F9A:staa | *L0011 | ;STORE MINOR LOOP COUNTER |
| 9F9C | ldab | *L00EF | ;1 SECOND COUNTER |
| 9F9E | cmpb | #0x9F | ;159 |
| 9FA0 | bcc | L9FA7 | ;BRANCH IF >= 159, ELSE |
| 9FA2 | incb | | ;INCREMENT |
| 9FA3 | stab | *L00EF | ;STORE 1 SECOND COUNTER |
| 9FA5 | bra | L9FC6 | ;GO UPDATE HUD DEVICE |
| | | | |
| 9FA7 | L9FA7:clr | L00EF | ;CLEAR 1 SECOND COUNTER |
| 9FAA | ldab | *L0099 | ; |
| 9FAC | eorb | #0x20 | ;TOGGLE 1 SECOND BIT |
| 9FAE | stab | *L0099 | ;STORE |
| 9FB0 | brclr | *L0086,#0x80,L9FC6 | ;BRANCH IF ENGINE NOT RUNNING, ELSE |
| 9FB4 | ldx | *L0032 | ;LOAD RUN TIME |
| 9FB6 | inx | | ;INC RUN TIME |
| 9FB7 | stx | *L0032 | ;RUN TIME |
| 9FB9 | cpx | L8C6B | ;ENABLE IAC RESET IF RUN TIME EXCEEDS THIS |
| 9FBC | bcs | L9FC1 | ;BRANCH IF RUN TIME < CAL, ELSE |
| 9FBE | bset | *L0012,#0x40 | ;SET IAC RESET ENABLED |
| 9FC1 | L9FC1:ldx | *L00ED | ;RUN TIME |
| 9FC3 | inx | | ;INCREMENT TIMER |
| 9FC4 | stx | *L00ED | ;STORE RUN TIME |
| 9FC6 | L9FC6:brclr | *L0095,#0x02,L9FCD | ;BRANCH IF HUD DEVICE NOT CONNECTED, ELSE |
| 9FCA | jsr | L5BF2 | ;OUTSIDE LOADED SOURCE FILE |
| 9FCD | L9FCD:ldab | *L00DC | ; |
| 9FCF | beq | LA02A | ;BRANCH IF Z, ELSE |
| 9FD1 | brset | *L0095,#0x10,L9FE7 | ;BRANCH IF BATT VOLTS < 4.0, ELSE |
| 9FD5 | brset | *L0092,#0x20,L9FE7 | ; |
| 9FD9 | clr | L00DC | ; |
| 9FDC | bclr | *L0012,#0x40 | ;CLEAR IAC RESET ENABLED |
| 9FDF | bclr | *L00D9,#0x06 | ;CLEAR MOTOR HAS BEEN AT 0 DURING RESET AND |
| | | | ;MOTOR RESET IN PROGRESS |
| 9FE2 | bset | *L00D9,#0x10 | ;SET RESET COMPLETE |
| 9FE5 | bra | LA002 | |
| | | | |
| 9FE7 | L9FE7:ldaa | L8022 | ;OPTION WORD - TIS SET |
| 9FEA | beq | L9FFB | ;BRANCH IF Z, ELSE |
| 9FEC | ldaa | L01B9 | ; |
| 9FEF | bne | L9FF8 | ;BRANCH IF NZ, ELSE |
| 9FF1 | ldaa | #0x03 | ;LOAD 3 |
| 9FF3 | staa | L01B9 | ;SAVE |
| 9FF6 | bra | LA02A | ; |
| | | | |
| 9FF8 | L9FF8:dec | L01B9 | ; |

```

9FFB L9FFB:dec L00DC ;
9FFE brclr *L00D9,#0x02,LA00C ;BRANCH IF MOTOR HAS NOT NOT BEEN RESET TO
; 0 DURING RESET
A002 LA002:inc L0056 ;FREE RUNNING IAC COUNTER
A005 ldab *L0057 ;IAC POSITION
A007 incb ;INCREMENT IAC POSITION
A008 beq LA016 ;BRANCH IF Z, ELSE
A00A bra LA014 ;GO STORE IAC POSITION

A00C LA00C:dec L0056 ;FREE RUNNING IAC COUNTER
A00F ldab *L0057 ;IAC POSITION
A011 beq LA014 ;BRANCH IF = ZERO, ELSE
A013 decb ;DECREMENT
A014 LA014:stab *L0057 ;IAC POSITION

;-----
; MOVE IAC MOTOR ONE STEP
;-----
A016 LA016:ldab *L0056 ;FREE RUNNING IAC COUNTER
A018 ldx #0xFFD4 ;INDEX IAC BYTES
A01B andb #0x03 ;CLEAR BITS 0 AND 1
A01D abx ;ADD TO ADDRESS
A01E sei ;HOLD INTERRUPTS
A01F ldab L4002 ;LOAD CPU DATA LATCH
A022 andb #0xFC ;CLEAR BITS 0 AND 1
A024 orab 0x00,x ;SET BITS ACCORDING TO MASK
A026 stab L4002 ;STORE CPU DATA LATCH
A029 cli ;CLEAR AND ALLOW INTERRUPTS
A02A LA02A:brset *L0086,#0x80,LA03E ;BRANCH IF ENGINE RUNNING, ELSE
A02E ldx L01BA ;
A031 bne LA037 ;BRANCH IF NOT Z, ELSE
A033 brclr *L0097,#0x02,LA03E ;THIS GETS SET IN THE XIRQ ROUTINE
A037 LA037:ldx L01BA ;
A03A inx ;
A03B stx L01BA ;
A03E LA03E:sei ;HOLD INTERRUPTS
A03F brset *L0097,#0x02,LA089 ;THIS GETS SET IN THE XIRQ ROUTINE
A043 cli ;CLEAR AND ALLOW INTERRUPTS
A044 brset *L0086,#0x80,LA04C ;BRANCH IF ENGINE RUNNING, ELSE
A048 brclr *L0097,#0x80,LA072 ;BRANCH IF RUN-FUEL NOT ALLOWED, ELSE
A04C LA04C:ldaa L026D ;REF PERIOD TIMER
A04F cmpa L81C8 ;16d
A052 bls LA072 ;BRANCH IF <= 81C8, ELSE
A054 ldaa L8718 ;0x08
A057 staa *L0030 ;STORE TIMER
A059 brset *L0095,#0x10,LA06F ;BRANCH IF BATT VOLTS < 4.0, ELSE
A05D ldaa *L00A6 ;LOAD CLNT TEMP (DEFAULTED)
A05F cmpa L8EC9 ;0
A062 bcc LA06F ;BRANCH IF CTS2 > CAL, ELSE
A064 ldaa *L0076 ;THIS NOT USED
A066 adda L8EDB ;ADD 0
A069 bcc LA06D ;BRANCH IF NO OVERFLOW, ELSE
A06B ldaa #0xFF ;LOAD 255
A06D LA06D:staa *L0076 ;NOT USED
A06F LA06F:jmp LB9D6 ;GO CLEAR SCI INTERRUPTS - WHY????????

A072 LA072:inc L026D ;INCREMENT REF PERIOD TIMER
A075 ldaa L026D ;LOAD TIMER
A078 cmpa #0x90 ;144d
A07A bls LA0BE ;BRANCH IF <= 144, ELSE
A07C bclr *L0098,#0x08 ;CLEAR REF PULSE OCCURED
A07F ldd #0xFFFF ;
A082 std *L00C0 ;SET REF PER TO MAX
A084 jsr LF691 ;GO CLEAR FUEL OUTPUT CONTROL REGS

```

```

A087          bra      LA0BE          ;

;-----
; BRANCH HERE IF BIT 1 L0097 SET
;-----

A089  LA089:bclr      *L0097,#0x02          ;THIS GETS SET IN THE XIRQ ROUTINE
A08C          cli                      ;CLEAR AND ALLOW INTERRUPTS
A08D          brset    *L0095,#0x10,LA0A7    ;BRANCH IF BATT VOLTS < 4.0, ELSE
A091          bset     *L009A,#0x30          ;SET BITS 4 AND 5
                                           ;? REF PULSE OCCURED
A094          ldaa     L8717                ;INITIAL VALUE FOR L0030 WHEN CLEAR FLOOD
A097          ldab     L0152                ;CRANKING PW MULTIPLIER VS %TPS
A09A          beq      LA0A5                ;BRANCH IF Z, ELSE
A09C          brset    *L0099,#0x04,LA0A5    ;BRANCH IF CLEAR FLOOD MODE, ELSE
A0A0          ldaa     *L0030                ;TIMER
A0A2          inca     ;INCREMENT
A0A3          beq      LA0A7                ;BRANCH IF Z, ELSE
A0A5  LA0A5:staa      *L0030                ;STORE TIMER
A0A7  LA0A7:ldaa      *L0031                ;LOAD TIMER
A0A9          inca     ;INCREMENT
A0AA          beq      LA0AE                ;BRANCH IF Z, ELSE
A0AC          staa     *L0031                ;TIMER
A0AE  LA0AE:ldx       L3FC0                ;LOAD REF PERIOD
A0B1          stx      *L00C0                ;STORE REF PERIOD
A0B3          inc      L02AA                ;
A0B6          ldaa     L02B2                ;TIMER
A0B9          beq      LA0BE                ;BRANCH IF Z, ELSE
A0BB          dec      L02B2                ;DECREMENT TIMER
A0BE  LA0BE:ldd       *L00CF                ;
A0C0          std      *L00CD                ;STORE MAF (GM/SEC)

;-----
; PROM ERROR CHECK
;-----

A0C2          ldab     L8DCF                ;0x55
A0C5          cmpb     #0x55                ;COMPARE
A0C7          bne      LA0CF                ;BRANCH IF 8DCF NOT 0x55, ELSE
A0C9          comb     ;INVERT B
A0CA          cmpb     L90CE                ;0xAA
A0CD          beq      LA0E3                ;BRANCH IF INVERSE 8DCF = 0xAA
A0CF  LA0CF:bset      *L0088,#0x20          ;SET PROM ERROR
A0D2          bclr     *L0086,#0x80          ;CLEAR ENGINE RUNNING
A0D5          bclr     *L0097,#0x80          ;CLEAR RUN-FUEL ALLOWED
A0D8          ldaa     #0x03                ;
A0DA          staa     *L00E1                ;

;-----
; CHECK IF TIME FOR 12.5 MSEC CRANKING ROUTINE
; SAME AS $5B
;-----

A0DC          brset    *L0011,#0x01,LA112    ;BRANCH IF 12.5 MSEC (DO FUEL), ELSE
A0E0          jmp      LA99B                ;DO IAC ROUTINE

;-----
; ? CHECK IF TIME FOR 12.5 MSEC SPARK ROUTINE
;-----

A0E3  LA0E3:brclr     *L0011,#0x01,LA115    ;BRANCH IF 12.5 MSEC, ELSE
A0E7          bclr     *L0090,#0x40          ;CLEAR BIT 6
A0EA          brclr    *L0084,#0x80,LA105    ;BRANCH IF NOT MODE 4, ELSE
A0EE          ldaa     #0x80                ;BIT 7
A0F0          anda     L0230                ;MODE 4 CONTROL BYTE
A0F3          beq      LA105                ;BRANCH IF BIT 7, ELSE
A0F5          ldab     L8011                ;OPTION WORD - VATS

```

```

A0F8      beq      LA0FE
A0FA      brclr   *L0012,#0x02,LA105
A0FE      LA0FE:bset *L0090,#0x40
A101      ldab    #0x03
A103      bra     LA110

A105      LA105:ldaa *L0089
A107      eora    #0x02
A109      staa    *L0089
A10B      ldab    *L00E1
A10D      beq     LA110
A10F      decb
A110      LA110:stab *L00E1
A112      LA112:jmp LB763

A115      LA115:brset *L009A,#0x10,LA11C
A119      jmp     LA1E4

A11C      LA11C:ldx  *L00EB
A11E      stx     L0281
A121      ldd     #0x0014
A124      ldx     *L00C0
A126      fdiv
A127      stx     *L00EB
A129      ldx     *L00C0
A12B      brset   *L009A,#0x08,LA144
A12F      ldaa    L01BC
A132      beq     LA13E
A134      cpx     #0x0B61
A137      bcc     LA13E
A139      bset    *L009A,#0x08
A13C      bra     LA144

A13E      LA13E:inca
A13F      beq     LA144
A141      staa    L01BC
A144      LA144:clrb
A145      cpx     L81C5
A148      bhi     LA151
A14A      ldab    L01A6
A14D      incb
A14E      bne     LA151
A150      decb
A151      LA151:stab L01A6
A154      cmpb    L81C7
A157      bls     LA15C
A159      bset    *L0097,#0x80
A15C      LA15C:ldd  L011E
A15F      lsr     L011E
A160      lsr     L011E
A161      lsr     L011E
A162      coma
A163      comb
A164      add     L011E
A167      bpl     LA16B
A169      clra
A16A      clrb
A16B      LA16B:std  L011E
A16E      brset   *L0086,#0x80,LA1D9
A172      clra
A173      cpx     L81C2
A176      bcc     LA185
A178      brclr   *L0095,#0x08,LA1DE

;BRANCH IF Z, ELSE
;BRANCH IF VATS FAILED, ELSE
;
;
;? SC1SDO
;TOGGLE BIT 1
;STORE
;
;BRANCH IF Z, ELSE
;DECREMENT COUNTER
;
;-> MAIN FUEL CALC

;BRANCH IF ? RUN-SPARK ALLOWED, ELSE
;GO DO RPM, MPH AND SPARK CALC

;RPM
;PREV RPM
;LOAD 20
;LOAD REF PERIOD
;DIVIDE
;RPM (16 BIT RPM)
;LOAD REF PERIOD
;BRANCH IF SEFI MODE, ELSE
;LOAD LOW REF PER COUNTER
;BRANCH IF Z, ELSE
;450 RPM
;BRANCH IF < 450 RPM, ELSE
;SET SEFI MODE (NOT A QUALIFIER)
;

;INCREMENT COUNTER
;BRANCH IF Z, ELSE
;STORE LOW REF PER COUNTER
;CLEAR
;0x0B60 - 450 RPM
;BRANCH IF < 450 RPM, ELSE
;LOAD LOW REF PER COUNTER
;INCREMENT COUNTER
;BRANCH IF NOT Z, ELSE
;DECREMENT TIMER
;STORE LOW REF PER COUNTER
;COMPARE COUNTS 0x03
;BRANCH IF <= 0x03, ELSE
;SET RUN FUEL ALLOWED
;LOAD DYNAMIC DWELL
;DIV BY 8

;INVERT A
;INVERT B
;ADD PREV DYNAMIC DWELL
;BRANCH IF PLUS, ELSE
;CLEAR A
;CLEAR B
;STORE DYNAMIC DWELL
;BRANCH IF ENGINE RUNNING, ELSE
;
;3084 - 425 RPM
;BRANCH IF REF PER >= 3084, ELSE
;BRANCH IF NOT BIT 3, ELSE
;-> GO SET BIT 3 L0095 AND CLEAR RPM

```

```

; VARIABLES AND RUN TIME
A17C      ldaa    L01A0      ;LOAD CRANKING TIMER
A17F      cmpa    L81C4      ;COMPARE TIME LIMIT
A182      bcc     LA1AE      ;BRANCH IF TIMER >= CAL, ELSE
                                ;GO SET ENGINE RUNNING
A184      inca
A185      LA185:staa    L01A0      ;INCREMENT TIMER
                                ;STORE CRANKING TIMER
A188      bclr    *L0098,#0x08      ;CLEAR REF PULSE OCCURED
A18B      ldaa    L8018      ;OPTION WORD - TIS SET
A18E      beq     LA1A9      ;BRANCH IF Z, ELSE
A190      brset   *L0013,#0x08,LA1A9      ;BRANCH IF 24X CRANK SIGNAL MALF, ELSE
A194      ldaa    L026D      ;REF PERIOD TIMER
A197      cmpa    #0x4E      ;78d
A199      bhi     LA1A9      ;BRANCH IF > 78, ELSE
A19B      ldd     L3FFC      ;CPU CNTL REG
A19E      bset    *L0089,#0x04      ;ENABLED EST
A1A1      orab    #0x10      ;SET BIT 4
A1A3      std     L3FFC      ;CPU CNTL REG
A1A6      bset    *L0098,#0x08      ;SET REF PULSE OCCURED
A1A9      LA1A9:clr    L026D      ;CLEAR REF PERIOD TIMER
A1AC      bra     LA1E1      ;-> GO CLEAR RPM VARIABLES AND RUN TIME

A1AE      LA1AE:bset    *L0086,#0x80      ;SET ENGINE RUNNING BIT
A1B1      ldaa    L0123      ;
A1B4      adda    L81C1      ;SPARK REF ANGLE
A1B7      staa    L030E      ;STORE SPARK TERM
A1BA      bclr    *L0013,#0x0C      ;CLEAR CAM SENSOR AND 24X ERRORS
A1BD      brclr   *L0012,#0x10,LA1D3      ;BRANCH IF NOT BIT 4, ELSE
A1C1      ldaa    #0x80      ;MASK
A1C3      ldab    *L0010      ;
A1C5      bmi     LA1C9      ;BRANCH IF MINUS, ELSE
A1C7      ldaa    #0x40      ;MASK
A1C9      LA1C9:ldx    #0x90AB      ;INDEX
A1CC      ldab    #0x06      ;OFFSET 90B1 - B6 AND B7 ARE NOT SELECTED
A1CE      jsr     LF87D      ;GO UPDATE MALF FLAGS
A1D1      bra     LA1D6      ;

A1D3      LA1D3:bclr    *L0019,#0xC0      ;CLEAR MALF FLAGS - NOT ENABLED
A1D6      LA1D6:bset    *L0012,#0x18      ;SET BITS 3 AND 4
A1D9      LA1D9:clr    L026D      ;REF PERIOD TIMER
A1DC      bra     LA1E8      ;GO DO RPM CALCULATIONS

A1DE      LA1DE:bset    *L0095,#0x08      ;
A1E1      LA1E1:jmp     LA99B      ;GO DO IAC ROUTINE

;-----
; DO RPM CALCULATIONS
;-----
A1E4      LA1E4:brclr   *L0086,#0x80,LA1E1      ;BRANCH IF ENGINE NOT RUNNING, ELSE
                                ;-> GO CLEAR RPM VARIABLES AND RUN TIME
A1E8      LA1E8:ldaa    #0xFF      ;LOAD 255
A1EA      staa    L400B      ;STORE
A1ED      ldd     *L00C0      ;LOAD REF PERIOD
A1EF      lsr     ;DIV BY 2
A1F0      addd    *L00C0      ;ADD REF PERIOD
A1F2      pshb    ;SAVE LSB
A1F3      psha    ;SAVE MSB
A1F4      pulx    ;GET INTO X REG
A1F5      rora    ;DIV BY 2
A1F6      rorb    ;DIV BY 2
A1F7      std     *L00C8      ;3/4 3x REF PERIOD
A1F9      ldd     #0x0133      ;307
A1FC      fdiv    ;DIVIDE BY 307
A1FD      pshx    ;SAVE RESULT ON STACK

```

```

A1FE      tsx      ;
A1FF      ldd      0x00,x      ;
A201      cmpa     #0x60      ;96 (2400 RPM)
A203      bcs      LA20E      ;BRANCH IF < 2400 RPM, ELSE
A205      addd     #0x4080     ;ADD 16512
A208      bcc      LA215      ;BRANCH IF NO OVERFLOW, ELSE
A20A      ldaa     #0xFF      ;LOAD 255
A20C      bra      LA215      ;GO STORE NLRPMX

A20E      LA20E:lsld      ;MUL BY 2
A20F      subd     #0x1F80     ;SUB 8064
A212      bcc      LA215      ;BRANCH IF NO UNDERFLOW, ELSE
A214      clra      ;CLEAR A
A215      LA215:staa     *L00AB      ;STORE NLRPM INDEX
A217      bset     *L0091,#0x02      ;SET BIT 1
A21A      cmpa     #0xFF      ;6375 RPM
A21C      beq      LA221      ;BRANCH IF = 6375 RPM, ELSE
A21E      bclr     *L0091,#0x02      ;CLEAR BIT 1
A221      LA221:ldd     0x00,x      ;LOAD
A223      addd     #0x0080     ;ADD 128
A226      sbca     #0x00      ;ROUND DOWN
A228      staa     *L00AC      ;STORE RPM/25
A22A      cmpa     #0x60      ;COMPARE 2400 RPM
A22C      bcc      LA237      ;BRANCH IF >= 2400 RPM, ELSE
A22E      ldaa     *L00AB      ;LOAD NLRPMX
A230      suba     #0x30      ;SUB 48 (1200 RPM)
A232      bcc      LA23D      ;BRANCH IF > 1200 RPM, ELSE
A234      clra      ;CLEAR A
A235      bra      LA23D

A237      LA237:adda     #0x10      ;ADD 16
A239      bcc      LA23D      ;BRANCH IF NO OVERFLOW
A23B      ldaa     #0xFF      ;LOAD 255
A23D      LA23D:staa     L0296      ;STORE LIMITED NLRPMX
A240      ldd      0x00,x      ;LOAD
A242      lsld      ;MUL BY 2
A243      bcs      LA24A      ;BRANCH IF OVERFLOW, ELSE
A245      addd     #0x0080     ;ROUND
A248      bcc      LA24C      ;BRANCH IF NO OVERFLOW, ELSE
A24A      LA24A:ldaa     #0xFF      ;LOAD 255 (3187.5 RPM)
A24C      LA24C:staa     *L00AD      ;STORE RPM/12.5

;-----
; DO MPH CALCULATION
;-----
A24E      brset     *L0095,#0x10,LA27B      ;BRANCH IF BATT VOLTS < 4.0, ELSE
A252      ldy      L3FE0      ;LOAD ? ROAD SPD PULSE COUNTER
A256      mul      ;DELAY
A257      ldz      L3FC2      ;B CNTR
A25A      mul      ;DELAY
A25B      cpy      L3FE0      ;? ANY VSS PULSES ?
A25F      beq      LA267      ;BRANCH IF NOT, ELSE
A261      iny      ;INCREMENT Y
A263      nop      ;DELAY
A264      ldz      L3FC2      ;LOAD B CNTR
A267      LA267:cpx      L011C      ;
A26A      bne      LA28D      ;
A26C      ldab     L0119      ;
A26F      cmpb     L9777      ;0x17
A272      bhi      LA27B      ;BRANCH IF > 0x17, ELSE
A274      incb     ;
A275      stab     L0119      ;
A278      jmp      LA38B      ;

```

| | | | | |
|------|--------|-------|--------------------|--|
| A27B | LA27B: | bclr | *L0086,#0x10 | ;CLEAR TCC ROAD SPD 1 ST PULSE FLAG |
| A27E | | ldd | #0x0000 | ; |
| A281 | | std | L0261 | ;TURBINE RPM |
| A284 | | std | L0263 | ;MPH |
| A287 | | std | L0265 | ;RATIO OF ENGINE TO TURBINE RPM |
| A28A | | jmp | LA32A | ; |
| A28D | LA28D: | pshx | | |
| A28E | | brset | *L0086,#0x10,LA29D | ;BRANCH IF TCC ROAD SPD PULSE, ELSE |
| A292 | | bset | *L0086,#0x10 | ;SET TCC ROAD SPD 1 ST PULSE FLAG |
| A295 | | inx | | |
| A296 | | stx | L011C | |
| A299 | | inx | | |
| A29A | | stx | L011A | |
| A29D | LA29D: | clr | L0119 | |
| A2A0 | | ldx | L011C | |
| A2A3 | | stx | L011A | |
| A2A6 | | pulx | | |
| A2A7 | | stx | L011C | |
| A2AA | | ldd | L011C | |
| A2AD | | subd | L011A | |
| A2B0 | | pshb | | |
| A2B1 | | psha | | |
| A2B2 | | ldd | L0267 | |
| A2B5 | | nega | | |
| A2B6 | | negb | | |
| A2B7 | | sbca | #0x00 | |
| A2B9 | | sty | L0267 | |
| A2BD | | addd | L0267 | |
| A2C0 | | stab | L0269 | ; |
| A2C3 | | tba | | |
| A2C4 | | ldx | L0252 | ;; ACCUMULATED DISTANCE |
| A2C7 | | abx | | |
| A2C8 | | stx | L0252 | ;; ACCUMULATED DISTANCE |
| A2CB | | ldx | #0x9775 | ;; ROAD SPEED CONSTANT |
| A2CE | | jsr | L98B9 | ;8 X 16 MULTIPLY |
| A2D1 | | pulx | | ;GET PRODUCT IN TO X REG |
| A2D2 | | pshx | | ;SAVE ON STACK |
| A2D3 | | fdiv | | ;MAKE THEM A FRACTION |
| A2D4 | | pshx | | ;SAVE ON STACK |
| A2D5 | | tsx | | ;X POINTS TO STACK |
| A2D6 | | ldaa | 0x00,x | ;GET LSB FROM STACK -> A REG |
| A2D8 | | ldab | L978C | ;1 |
| A2DB | | suba | L0263 | ;MPH |
| A2DE | | bcc | LA2E2 | ;BRANCH IF MPH < 1, ELSE |
| A2E0 | | negb | | ; |
| A2E1 | | nega | | ; |
| A2E2 | LA2E2: | cmpa | L978C | ;1 |
| A2E5 | | ldaa | 0x00,x | ; |
| A2E7 | | bls | LA2EF | ;BRANCH IF <= 1, ELSE |
| A2E9 | | ldaa | L0263 | ;MPH |
| A2EC | | aba | | ;ADD B TO MPH |
| A2ED | | staa | 0x00,x | ;STORE |
| A2EF | LA2EF: | pulx | | ;GET INTO X REG |
| A2F0 | | stx | L0263 | ;STORE MPH |
| A2F3 | | pshx | | ;SAVE ON STACK |
| A2F4 | | pula | | ;GET MSB |
| A2F5 | | pulb | | ;GET LSB |
| A2F6 | | clrb | | ;CLEAR LSB |
| A2F7 | | psha | | ;SAVE ON STACK |
| A2F8 | | pshb | | ; |
| A2F9 | | pulx | | ;GET INTO X REG |
| A2FA | | ldd | *L00EB | ;RPM |
| A2FC | | idiv | | ;DIVIDE |

```

A2FD          stx      L0283                      ;STORE QUOTIENT (RPM/MPH)
A300          ldab     #0x02                      ;2
A302          ldx      #0x978D                    ;INDEX
A305          brclr    *L008E,#0x08,LA314          ;BRANCH IF 4TH GEAR, ELSE
A309          abx      ;978F
A30A          brclr    *L008E,#0x04,LA314          ;BRANCH IF 3RD GEAR, ELSE
A30E          abx      ;9791
A30F          brclr    *L008E,#0x02,LA314          ;BRANCH IF 2ND GEAR, ELSE
A313          abx      ;9793
A314          LA314:ldab    L0269
A317          jsr      L98CD                      ;B x X, 8 X 16 MULTIPLY
A31A          pulx     ;GET INTO X REG
A31B          jsr      L98E0                      ;DIVIDE
A31E          stx      L0261                      ;STORE TURBINE RPM
A321          ldd      L0261                      ;LOAD TURBINE RPM
A324          ldx      *L00EB                      ;16 BIT RPM
A326          fdiv     ;D/X
A327          stx      L0265                      ;STORE RATIO OF ENGINE TO TURBINE RPM
A32A          LA32A:ldx    #0x90AB                ;INDEX
A32D          brclr    *L0015,#0x02,LA33B          ;BRANCH IF NOT P0501 VSS MALF, ELSE
A331          brclr    0x0A,x,#0x02,LA369          ;BRANCH IF NOT BIT 1 90B5 - TIS SET
A335          bset     *L00A0,#0x30                ;SET TIMER TO 48d
A338          jmp      LA369                      ;JUMP

;-----
; DTC P0501 CHECK - VSS INTERMITENT SIGNAL
;-----
A33B          LA33B:ldaa    L0137                      ;MALF TIMER
A33E          ldab     L0263                      ;MPH
A341          bne      LA366                      ;BRANCH IF MPH NOT Z, ELSE
A343          brset    *L0015,#0x01,LA366          ;BRANCH IF TCC BRAKE SW ERROR, ELSE
A347          brset    *L008E,#0x40,LA366          ;BRANCH IF BRAKE APPLIED, ELSE
A34B          ldab     L0138                      ;P/N -> IN GEAR TRANSITION TIMER
A34E          cmpb     0x90,x                      ;913B - 4 SECONDS
A350          bls      LA366                      ;BRANCH IF <= 4 SECONDS, ELSE
A352          ldab     *L00F1                      ;TRANNNY MPH
A354          cmpb     0x8C,x                      ;9137 - 18 MPH
A356          bls      LA366                      ;BRANCH IF <= 18 MPH, ELSE
A358          inca     ;INCREMENT TIMER
A359          cmpa     0x8D,x                      ;9138 - 160 -? 1.6 SECONDS
A35B          bcs      LA397                      ;BRANCH IF < 9138, ELSE
A35D          ldaa     #0x02                      ;MASK
A35F          ldab     #0x02                      ;OFFSET - 90AD, L0015
A361          jsr      LF87D                      ;GO UPDATE MALF FLAGS
A364          bra      LA39A

A366          LA366:clr     L0137                      ;CLEAR MALF TIMER
A369          LA369:ldd     L0263                      ;NEW UNFILTERED MPH
A36C          ldx      #0x00BD                      ;OLD FILTERED MPH
A36F          ldy      #0x9795                      ;FILTER COEFFICIENT ADDRESS
A373          sei      ;HOLD INTERRUPTS
A374          jsr      L99F5                      ;LAG FILTER
A377          cli      ;CLEAR AND ALLOW INTERRUPTS
A378          staa     *L00F1                      ;STORE FILTERED MPH MSB
A37A          cmpa     #0x3B                      ;59 MPH
A37C          bls      LA381                      ;BRANCH IF <= 59 MPH, ELSE
A37E          ldd      #0x3C00                      ;
A381          LA381:lsld    ;MUL BY 4
A382          lsld     ;
A383          ldab     #0xCD                      ;205
A385          mul      ;A * 205
A386          addd     #0x0080                      ;ROUND
A389          staa     *L00BF                      ;STORE MPH*3
A38B          LA38B:brclr    *L0014,#0x20,LA39A          ;BRANCH IF NOT DTC P0502 - NO VSS SIG, ELSE

```

```

A38F      ldaa    L8CE3                      ;DEFAULT MPH
A392      inca                      ;INCREMENT
A393      staa    *L00BF                    ;STORE MPH*3
A395      bra     LA39A

A397      LA397:staa    L0137                ;STORE MALF TIMER
A39A      LA39A:ldd     L0263                ;NEW UNFILTERED MPH
A39D      ldx     #0x0310                    ;OLD FILTERED MPH
A3A0      ldy     #0x977D                    ;FILTER COEFFICIENT ADDRESS
A3A4      sei                      ;HOLD INTERRUPTS
A3A5      jsr     L99F5                      ;LAG FILTER
A3A8      cli                      ;CLEAR AND ALLOW INTERRUPTS
A3A9      tsx                      ;X POINTS TO STACK
A3AA      ldaa    #0xFF                      ;
A3AC      ldab    *L00BA                    ;LOAD LV8
A3AE      subb    *L00BC                    ;
A3B0      bcs     LA3B7                      ;BRANCH IF LV8 DECREASING, ELSE
A3B2      cmpb    L8331                    ;COMPARE
A3B5      bcc     LA3C2                      ;BRANCH IF D-LV8 > CAL, ELSE

;-----
; TEST IF DYNAMIC DWELL EXCEEDS REF PER/8
;   - LIMIT DD TO REF PER/8
;-----
A3B7      LA3B7:ldd     *L00C2                ;LOAD PREV REF PERIOD
A3B9      subd    *L00C0                    ;SUB CURRENT REF PERIOD
A3BB      lsld                      ;MULTIPLY BY 2
A3BC      cpd     L011E                    ;COMPARE DYNAMIC DWELL
A3C0      bmi     LA3C5                    ;BRANCH IF DELTA REF PERIOD < DD, ELSE
A3C2      LA3C2:std     L011E                ;STORE DD

A3C5      LA3C5:ldd     *L00C0                ;LOAD REF PERIOD
A3C7      lsr     ;DIV / 8
A3C8      lsr
A3C9      lsr
A3CA      cpd     L011E                    ;DD
A3CE      bcc     LA3D3                    ;BRANCH IF > DD, ELSE
A3D0      std     L011E                    ;STORE DD

;-----
; ?? DECAY DD TO ZERO (COMPLETED EVERY 12.5 MSEC INTERVAL)
;-----
A3D3      LA3D3:ldd     *L00C0                ;LOAD REF PERIOD
A3D5      std     *L00C2                    ;STORE FOR NEXT PASS
A3D7      lsr     ;DIV BY 2
A3D8      subd    #0x00E5                    ;3.49 MSEC
A3DB      bcc     LA3E2                    ;BRANCH IF > 3.5 MSEC, ELSE
A3DD      addd    #0x0134                    ;ADD 4.697 MSEC
A3E0      bra     LA3F2                    ;GO STORE

A3E2      LA3E2:lsr     ;DIV BY 2
A3E3      subd    #0x0127                    ;4.5 MSEC
A3E6      bcs     LA3ED                    ;BRANCH IF < 4.5 MSEC, ELSE
A3E8      addd    #0x017E                    ;ADD 5.825 MSEC
A3EB      bra     LA3F2                    ;GO STORE

A3ED      LA3ED:add     #0x05F7                ;ADD 23.2 MSEC
A3F0      lsr     ;DIV BY 4
A3F1      lsr
A3F2      LA3F2:std     0x00,x                ;STORE X POINTS TO STACK

;-----
; BATT COMPENSATION FOR DYNAMIC DWELL
;-----

```

```

A3F4      ldaa      #0x7C      ;12.4 VOLTS
A3F6      suba      *L0083      ;SUB BATT VOLTS
A3F8      bcc      LA3FB      ;BRANCH IF BATT VOLTS >= 12.4, ELSE
A3FA      clra      ;CLEAR A

;-----
; ADD DWELL ACCORDING TO 12.0V - BATT VOLT
;-----
A3FB      LA3FB:ldab      #0x04      ;
A3FD      mul      ;MULTIPLY
A3FE      addd      0x00,x      ;ADD TOTAL DWELL
A400      addd      L011E      ;ADD DYNAMIC DWELL
A403      std      L0120      ;STORE NEW TOTAL DWELL

;-----
; LIMIT DWELL TO MAX ON TIME OF REF PER - 600 uSEC
;-----
A406      ldd      *L00C0      ;LOAD REF PERIOD
A408      subd      #0x0027
A40B      cpd      L0120      ;COMPARE TOTAL DWELL
A40F      bcc      LA414      ;BRANCH IF > TOTAL DWELL, ELSE
A411      std      L0120      ;STORE TOTAL DWELL

A414      LA414:pulx      ;
A415      clra      ;
A416      brclr      *L00DB,#0x80,LA43F      ;BRANCH IF A/C CLUTCH OFF, ELSE
A41A      ldab      *L00AC      ;LOAD RPM/25
A41C      cmpb      L81E4      ;
A41F      bhi      LA43F      ;BRANCH IF RPM/25 > CAL, ELSE
A421      ldab      *L00DE      ;LOAD DESIRED IDLE RPM
A423      subb      *L00AD      ;SUBTRACT RPM/12.5
A425      bcs      LA43F      ;BRANCH IF OVERRIDE, ELSE
A427      cmpb      L81E5      ;COMPARE
A42A      bls      LA43F      ;BRANCH IF DELTA RPM <= 81E5, ELSE
A42C      ldaa      L81E6      ;LOAD 81E6
A42F      brclr      *L008E,#0x01,LA436      ;BRANCH IF NOT P/N MODE, ELSE
A433      ldaa      L81E7      ;LOAD 81E7
A436      LA436:mul      ;DELTA RPM * 81E6 OR 81E7
A437      lsld      ;MUL BY 2
A438      bcs      LA43D      ;BRANCH IF OVERFLOW, ELSE
A43A      lsld      ;MUL BY 2
A43B      bcc      LA43F      ;BRANCH IF NO OVERFLOW, ELSE
A43D      LA43D:ldaa      #0xFF      ;LOAD 255
A43F      LA43F:staa      L02EE      ;STORE IDLE SPARK TERM
A442      brclr      *L0091,#0x02,LA455      ;BRANCH IF RPM NOT 6375, ELSE
A446      ldaa      *L00BA      ;LOAD LV8
A448      suba      #0x20      ;SUB 32
A44A      bcc      LA44D      ;BRANCH IF LV8 >= 32, ELSE
A44C      clra      ;CLEAR A
A44D      LA44D:ldx      #0x8118      ;LOAD RPM VS LV8
A450      jsr      L99D7      ;2D LOOKUP
A453      bra      LA45F

A455      LA455:ldab      *L00BA      ;LOAD LV8
A457      ldaa      *L00AB      ;LOAD NLRPMX
A459      ldx      #0x8025      ;MAIN SPARK ADVANCE VS RPM AND LV8
A45C      jsr      L995A      ;3D LOOKUP
A45F      LA45F:psha      ;SAVE TABLE LOOKUP RESULT ON STACK
A460      ldd      *L00EB      ;LOAD 16 BIT RPM
A462      subd      #0x12C0      ;4800
A465      bls      LA483      ;BRANCH IF <= 4800 RPM, ELSE
A467      ldx      #0x0019      ;LOAD 0019
A46A      idiv      ;DIVIDE
A46B      pshx      ;SAVE QUOTIENT (L00EB/19) ON STACK

```

| | | | |
|------|------------|--------------------|--|
| A46C | pula | | ;GET MSB |
| A46D | pulb | | ;GET LSB |
| A46E | cmpb | L81ED | ;COMPARE ?F1 EXTENSION HI RPM BREAKPOINT |
| A471 | bls | LA476 | ;BRANCH IF LESS, ELSE |
| A473 | ldab | L81ED | ;LOAD HI RPM BREAKPOINT |
| A476 | LA476:lslb | | ;MUL BY 2 |
| A477 | ldaa | L81EE | ;LOAD HI RPM ADVANCE SLOPE |
| A47A | mul | | ;MULTIPLY |
| A47B | pulb | | ;GET TABLE LOOKUP RESULT |
| A47C | aba | | ;ADD HI RPM ADVANCE TO TABLE RESULT |
| A47D | bcc | LA481 | ;BRANCH IF NO OVERFLOW, ELSE |
| A47F | ldaa | #0xFF | ;LOAD 255 |
| A481 | LA481:bra | LA484 | ;BRANCH -> STORE SPK ADV |
| | | | |
| A483 | LA483:pula | | ;GET TABLE LOOKUP RESULT |
| A484 | LA484:staa | L016B | ;STORE BASE SPARK ADVANCE |
| A487 | brclr | *L00DB,#0x80,LA4AB | ;BRANCH IF A/C CLUTCH OFF, ELSE |
| A48B | ldab | *L00A7 | ;FILTERED CTS |
| A48D | cmpb | L81E8 | ; |
| A490 | bcs | LA4AB | ;BRANCH IF CLNT < 81E8, ELSE |
| A492 | cmpb | L81E9 | ; |
| A495 | bhi | LA4AB | ;BRANCH IF CLNT > 81E9, ELSE |
| A497 | ldaa | *L00DE | ;LOAD DESIRED IDLE RPM |
| A499 | cmpa | L81EA | ; |
| A49C | bcc | LA4AB | ;BRANCH IF DES IDLE >= 81EA, ELSE |
| A49E | ldaa | L016B | ;LOAD BASE SPARK ADVANCE |
| A4A1 | adda | L02EE | ;ADD UNDERIDLE SPARK TERM |
| A4A4 | bcc | LA4A8 | ;BRANCH IF NO OVERFLOW, ELSE |
| A4A6 | ldaa | #0xFF | ;LOAD 255 |
| A4A8 | LA4A8:staa | L016B | ;STORE BASE SPARK ADVANCE |
| A4AB | LA4AB:jsr | LF8B0 | ;GO CALCULATE ENGINE TORQUE |
| A4AE | ldaa | *L00A7 | ;FILTERED CTS |
| A4B0 | cmpa | #0xD0 | ;COMPARE 116 DEG C |
| A4B2 | bls | LA4B6 | ;BRANCH IF SU CLNT <= 116 DEG C, ELSE |
| A4B4 | ldaa | #0xD0 | ;LOAD 116 DEG C |
| A4B6 | LA4B6:ldx | #0x8127 | ; |
| A4B9 | ldab | *L00BA | ;LOAD LV8 |
| A4BB | lsrb | | ;RESCALE |
| A4BC | jsr | L995A | ;3D LOOKUP |
| A4BF | staa | L016C | ;COOLANT SPARK CORRECTION VS LV8 |
| A4C2 | ldaa | #0x64 | ;100 |
| A4C4 | ldab | L8017 | ;OPTION WORD - TIS CLEAR |
| A4C7 | beq | LA4E6 | ;BRANCH IF Z, ELSE |
| A4C9 | ldab | *L00F2 | ;LOAD STARTUP COOLANT |
| A4CB | cmpb | L817E | ;COMPARE 817E = 00 |
| A4CE | bcc | LA4E6 | ;BRANCH IF >= 817E, ELSE |
| A4D0 | ldab | *L0032 | ;RUN TIME MSB |
| A4D2 | bne | LA4E6 | ;BRANCH IF NOT Z, ELSE |
| A4D4 | ldab | *L0033 | ;RUN TIME LSB |
| A4D6 | cmpb | #0x20 | ;32 SECONDS |
| A4D8 | bhi | LA4E6 | ;BRANCH IF > 32 SECONDS, ELSE |
| A4DA | lslb | | ;MUL BY 2 |
| A4DB | lslb | | ;MUL BY 2 |
| A4DC | tba | | ;TRANSFER TO A |
| A4DD | ldx | #0x817F | ;STARTUP SPARK CORRECTION |
| A4E0 | ldab | *L00BA | ;LV8 |
| A4E2 | lsrb | | ;RESCALE |
| A4E3 | jsr | L995A | ;3D LOOKUP |
| A4E6 | LA4E6:staa | L016F | ;STORE LOOKUP RESULT |
| A4E9 | ldaa | L01AF | ;LOAD EGR DC |
| A4EC | bne | LA4F4 | ;BRANCH IF NOT Z, ELSE |
| A4EE | clrb | | ;CLEAR B |
| A4EF | std | L01B7 | ;STORE EGR SPARK ADVANCE |
| A4F2 | bra | LA52A | ; |

| | | | |
|------|-------------|--------------------|--|
| A4F4 | LA4F4:ldab | *L00BA ` | ;LOAD LV8 |
| A4F6 | ldx | #0x81F6 | ;TABLE ADDRESS |
| A4F9 | ldaa | *L00AC | ;LOAD RPM/25 |
| A4FB | cmpa | #0x80 | ;3200 RPM |
| A4FD | bls | LA501 | ;BRANCH IF <= 3200 RPM, ELSE |
| A4FF | ldaa | #0x80 | ;LOAD 3200 RPM |
| A501 | LA501:jsr | L995A | ;3D LOOKUP |
| A504 | ldx | #0x01B7 | ;OLD FILTERED EGR SA |
| A507 | ldy | #0x81F5 | ;FILTER CONSTANT |
| A50B | jsr | L99F4 | ;DO LAG FILTER |
| A50E | ldx | #0x8254 | ;EGR SA MULTIPLIER VS FILT ENGINE TORQUE |
| A511 | ldaa | *L005C | ;FILTERED ENGINE TORQUE |
| A513 | cmpa | #0xB8 | ;184 |
| A515 | bls | LA519 | ;BRANCH IF <= 184, ELSE |
| A517 | ldaa | #0xB8 | ;LOAD 184 |
| A519 | LA519:suba | #0x70 | ; |
| A51B | bcc | LA51E | ;BRANCH IF NO UNDERFLOW, ELSE |
| A51D | clra | | ;CLEAR A |
| A51E | LA51E:jsr | L99CC | ;2D LOOKUP |
| A521 | ldab | L01B7 | ;FILTERED BASE EGR SA LSB |
| A524 | mul | | ;EGR SA * MULTIPLIER |
| A525 | lsl | | ;MUL BY 2 |
| A526 | bcc | LA52A | ;BRANCH IF NO OVERFLOW, ELSE |
| A528 | ldaa | #0xFF | ;LOAD 255 |
| A52A | LA52A:staa | L016D | ;STORE EGR SA CORRECTION |
| A52D | ldaa | L0195 | ;IAT A/D COUNTS |
| A530 | cmpa | #0xD1 | ;209 |
| A532 | bls | LA536 | ;BRANCH IF <= 209, ELSE |
| A534 | ldaa | #0xD1 | ;LOAD 209 |
| A536 | LA536:lsra | | ;RESCALE |
| A537 | ldab | *L00BA | ;LOAD LV8 |
| A539 | ldx | #0x825F | ;TABLE ADDRESS |
| A53C | jsr | L995A | ;3D LOOKUP |
| A53F | staa | L016E | ;STORE LOOKUP RESULT |
| A542 | ldaa | #0x64 | ; |
| A544 | ldab | L800F | ;OPTION WORD - TIS SET |
| A547 | beq | LA54F | ;BRANCH IF Z, ELSE |
| A549 | brset | *L00A3,#0x08,LA57E | ;BRANCH IF OFF MODE, ELSE |
| A54D | bra | LA553 | ; |
| A54F | LA54F:brclr | *L0096,#0x08,LA57E | ;BRANCH IF TCC NOT LOCKED, ELSE |
| A553 | LA553:ldab | *L00BA | ;LOAD LV8 |
| A555 | ldx | #0x82CB | ;TCC LOCKED SPARK CORRECTION |
| A558 | ldaa | *L00AC | ;LOAD RPM/25 |
| A55A | cmpa | #0x50 | ;2000 RPM |
| A55C | bls | LA560 | ;BRANCH IF <= 2000 RPM, ELSE |
| A55E | ldaa | #0x50 | ;LOAD 2000 RPM |
| A560 | LA560:lsla | | ;RESCALE |
| A561 | jsr | L995A | ;3D LOOKUP |
| A564 | psha | | ;SAVE RESULT ON STACK |
| A565 | ldaa | L0195 | ;IAT A/D COUNTS |
| A568 | lsra | | ;RESCALE |
| A569 | ldab | #0x08 | ;OFFSET |
| A56B | ldx | #0x8305 | ;TCC LOCKED SPK MULT VS IAT |
| A56E | jsr | L99D3 | ;2D LOOKUP |
| A571 | pulb | | ;GET PREVIOUS LOOKUP RESULT |
| A572 | subb | #0x64 | ;SUBTRACT 100 |
| A574 | bcc | LA57B | ;BRANCH IF NO UNDERFLOW, ELSE |
| A576 | negb | | ;INVERT RESULT |
| A577 | mul | | ;MULTIPLY |
| A578 | nega | | ;INVERT MSB OF PRODUCT |
| A579 | bra | LA57C | ; |

```

A57B    LA57B:mul                ;MULTIPLY
A57C    LA57C:adda    #0x64      ;ADD BACK OFFSET
A57E    LA57E:staa    L0170      ;STORE TCC LOCKED SPK CORRECTION
A581            brclr    *L008F,#0x20,LA59A ;BRANCH IF NOT LAUNCH MODE, ELSE
A585            ldaa    *L00AB    ;LOAD NLRPMX
A587            cmpa    #0x80      ;3200 RPM
A589            bls     LA58D      ;BRANCH IF <= 3200 RPM, ELSE
A58B            ldaa    #0x80      ;LOAD 3200 RPM
A58D    LA58D:ldab    *L00BA    ;LOAD LV8
A58F            ldx     #0x8425    ;TABLE ADDRESS
A592            jsr     L995A      ;3D LOOKUP
A595            staa    L0171      ;STORE LAUNCH MODE SPARK CORRECTION
A598            bra     LA5AC

A59A    LA59A:ldaa    #0x64      ;100
A59C            ldab    L0171      ;LAUNCH MODE SPARK CORRECTION
A59F            cba     ;COMPARE 100
A5A0            beq     LA5AC      ;BRANCH IF CORRECTION = 100, ELSE
A5A2            ldx     #0x0171    ;OLD LAUNCH MODE SPARK TERM
A5A5            ldy     #0x8424    ;FILTER CONSTANT
A5A9            jsr     L99F4      ;LAG FILTER
A5AC    LA5AC:ldaa    L8015      ;OPTION WORD - TIS CLEAR
A5AF            beq     LA5C0      ;BRANCH IF Z, ELSE
A5B1            brset   *L009B,#0x20,LA5C0 ;BRANCH IF PE MODE, ELSE
A5B5            ldaa    *L00A6      ;LOAD CLNT TEMP (DEFAULTED)
A5B7            cmpa    L84AF      ;0
A5BA            bls     LA5C0      ;BRANCH IF <= CAL, ELSE
A5BC            brclr   *L008F,#0xC0,LA5C3 ;BRANCH IF IDLE CONDITIONS NOT MET, ELSE
A5C0    LA5C0:jmp     LA685      ;JUMP OVER

;-----
; THIS USED ONLY IF OPTION L8015 SET
;-----

A5C3    LA5C3:ldaa    *L00AC      ;LOAD RPM/25
A5C5            cmpa    L84B0      ;0 RPM
A5C8            bhi     LA5C0      ;BRANCH IF > 0, ELSE
A5CA            brset   *L008E,#0x01,LA5C0 ;BRANCH IF P/N MODE, ELSE
A5CE            brclr   *L008E,#0x08,LA5C0 ;BRANCH IF 4TH GEAR, ELSE
A5D2            brset   *L009B,#0x02,LA5C0 ;BRANCH IF DFCO ENABLED, ELSE
A5D6            ldaa    *L00AA      ;%TPS
A5D8            cmpa    L84B1      ;0 %TPS
A5DB            bcs     LA615
A5DD            cmpa    L84B2      ;0 %TPS
A5E0            bls     LA5C0
A5E2            brset   *L0091,#0x80,LA5F9 ;%TPS
A5E6            ldaa    *L00AA      ;PREV FILTERED TPS%
A5E8            suba    L0153
A5EB            bcc     LA5EE
A5ED            clra
A5EE    LA5EE:cmpa    L84B4      ;0 dTPS%
A5F1            bls     LA5C0
A5F3            ldaa    L84B5      ;0
A5F6            staa    L01E1      ;0
A5F9    LA5F9:ldaa    L01E1      ;
A5FC            beq     LA5C0
A5FE            deca
A5FF            staa    L01E1
A602            ldx     #0x84BA      ;INDEX
A605            brclr   *L008E,#0x04,LA613 ;BRANCH IF 3RD GEAR, ELSE
A609            ldx     #0x84BD      ;INDEX
A60C            brclr   *L008E,#0x02,LA613 ;BRANCH IF 2ND GEAR, ELSE
A610            ldx     #0x84C0      ;INDEX
A613    LA613:bra     LA64D

```

```

A615  LA615:brset  *L0091,#0x80,LA62C      ;
A619      ldaa    L0153                      ;PREV FILTERED TPS%
A61C      suba    *L00AA                     ;%TPS
A61E      bcc     LA621                      ;BRANCH IF TPS DECREASING, ELSE
A620      clra    ;CLEAR
A621  LA621:cmpa   L84B3                      ;00
A624      bcs     LA685                      ;BRANCH IF < CAL, ELSE
A626      ld      L84B6                      ;0000
A629      stx     L01E2                      ;SAVE
A62C  LA62C:ldx    L01E2                      ;LOAD
A62F      beq     LA685                      ;BRANCH IF Z, ELSE
A631      dex     ;DECREMENT
A632      stx     L01E2                      ;SAVE
A635      ld      #0x84C3                    ;INDEX
A638      brclr   *L008E,#0x04,LA646        ;BRANCH IF 3RD GEAR, ELSE
A63C      ld      #0x84C9                    ;INDEX
A63F      brclr   *L008E,#0x02,LA646        ;BRANCH IF 2ND GEAR, ELSE
A643      ld      #0x84CF                    ;INDEX
A646  LA646:brset  *L0086,#0x20,LA64D        ;BRANCH IF A/C OFF, ELSE
A64A      ldab    #0x03                      ;OFFSET
A64C      abx     ;
A64D  LA64D:ldaa   0x01,x                    ;
A64F      staa    L01DD                      ;
A652      ldaa    0x02,x                    ;
A654      staa    L01DE                      ;
A657      ldaa    0x00,x                    ;
A659      bclr    *L0092,#0x02              ;CLEAR BIT 1
A65C      ld      #0x01DD                    ;
A65F      ldab    *L00AC                    ;LOAD RPM/25
A661      subb    L01E0                      ;
A664      bcc     LA66D                      ;
A666      negb    ;INVERT
A667      bset    *L0092,#0x02              ;
A66A      ld      #0x01DE                    ;
A66D  LA66D:mul    ;
A66E      tsta    ;
A66F      bne     LA675                      ;
A671      cmpb    0x00,x                    ;0x01DD OR
                                           ;0x01DE

A673      bcs     LA677                      ;
A675  LA675:ldab   0x00,x                    ;0x01DD OR
                                           ;0x01DE

A677  LA677:tba    ;
A678      bclr    *L0091,#0x80              ;
A67B      cmpa    L84B9                      ;0
A67E      bcs     LA6A4                      ;
A680      bset    *L0091,#0x80              ;
A683      bra     LA6A6                      ;

;-----
; JUMP HERE IF POWER ENRICHMENT MODE
; JUMP HERE IF 8015 OPTN FLAG CLEAR
; OR IF IDLE CONDITIONS
;-----

A685  LA685:brclr  *L0091,#0x80,LA695      ;
A689      ldaa    L01DF                      ;THIS WILL ALWAYS BE CLEAR
A68C      ldab    L84B8                      ;0
A68F      mul     ;A * B
A690      cmpa    L84B9                      ;0
A693      bcc     LA6A6                      ;BRANCH IF >= 0, ELSE
A695  LA695:bclr   *L0091,#0x80            ;CLEAR BIT 7
A698      ld      L84B6                      ;0000
A69B      stx     L01E2                      ;
A69E      ldaa    L84B5                      ;0

```

```

A6A1      staa    L01E1      ;
A6A4      LA6A4:ldaa    #0x00      ;
A6A6      LA6A6:staa    L01DF      ;SPARK TERM - THIS WILL ALWAYS BE ZERO
A6A9      ldaa    *L00AC      ;LOAD RPM/25
A6AB      staa    L01E0      ;STORE PREV RPM
A6AE      ldaa    #0x64      ;LOAD 100
A6B0      brclr   *L009B,#0x20,LA6DE ;BRANCH IF NOT PE MODE, ELSE
A6B4      ldaa    L0151      ;PE TIMER
A6B7      lsra    ;RESCALE
A6B8      ld      #0x830E      ;TABLE ADDRESS
A6BB      jsr     L99D7      ;2D LOOKUP
A6BE      psha    ;SAVE RESULT ON STACK
A6BF      cmpa    #0x64      ;COMPARE 100
A6C1      ld      #0x8317      ;TABLE ADDRESS
A6C4      bcc     LA6C9      ;BRANCH IF >= 100, ELSE
A6C6      ld      #0x831F      ;TABLE ADDRESS
A6C9      LA6C9:ldaa    *L00AC      ;LOAD RPM/25
A6CB      lsra    ;RESCALE
A6CC      ldab    #0x10      ;OFFSET (MIN 400 RPM)
A6CE      jsr     L99D3      ;2D LOOKUP
A6D1      pulb    ;RESTORE PREV TABLE LOOKUP RESULT
A6D2      subb    #0x64      ;SUB 100
A6D4      bcc     LA6DB      ;BRANCH IF >= 100, ELSE
A6D6      negb    ;INVERT RESULT
A6D7      mul     ;MUL BY CURRENT TABLE LOOKUP RESULT
A6D8      nega    ;INVERT MSB OF PRODUCT
A6D9      bra     LA6DC      ;GO ADD BACK 100 AND STORE

A6DB      LA6DB:mul     ;MUL BY CURRENT TABLE LOOKUP RESULT
A6DC      LA6DC:adda    #0x64      ;ADD BACK 100
A6DE      LA6DE:staa    L0178      ;STORE PE SPARK VARIABLE
A6E1      brset   *L0084,#0x80,LA735 ;BRANCH IF MODE 4, ELSE
A6E5      ldaa    *L00A7      ;FILTERED CTS
A6E7      cmpa    L81D9      ;
A6EA      bcs     LA735      ;BRANCH IF < CAL, ELSE
A6EC      ldaa    *L00AC      ;LOAD RPM/25
A6EE      cmpa    L81DA      ;
A6F1      bhi     LA735      ;BRANCH IF > CAL, ELSE
A6F3      ldaa    *L00AA      ;%TPS
A6F5      beq     LA735      ;BRANCH IF Z, ELSE
A6F7      brset   *L00DA,#0x40,LA708 ;
A6FB      suba    L0110      ;? PREV %TPS
A6FE      bcs     LA735      ;BRANCH IF TPS DECREASING, ELSE
A700      cmpa    L81DB      ;COMPARE
A703      bcs     LA735      ;BRANCH IF < CAL, ELSE
A705      bset    *L00DA,#0x40      ;CLEAR BIT 6
A708      LA708:ldaa    L017F      ;COUNTER, GETS INC WHEN BIT 6 L00DA SET
A70B      cmpa    L81DC      ;
A70E      bls     LA732      ;BRANCH IF <= CAL, ELSE
A710      cmpa    L81DD      ;
A713      bcc     LA735      ;BRANCH IF >= CAL, ELSE
A715      ldaa    *L00F3      ;KNOCK RETARD DEGREES
A717      beq     LA72A      ;BRANCH IF = ZERO, ELSE
A719      ldab    L0180      ;KNOCK RETARD ACCUMULATOR
A71C      aba     ;KNOCK RET + CUMMULATIVE KNOCK RETARD
A71D      staa    L0180      ;STORE FOR NEXT TIME
A720      cmpb    L81DE      ;MAX VALUE
A723      bls     LA73F      ;BRANCH IF L0180 <= MAX, ELSE
A725      ldab    L81DE      ;LOAD MAX
A728      bra     LA73F      ;GO STORE CUMMULATIVE KNOCK RETARD DEGREES

A72A      LA72A:ldab    L017E      ;CUMMULATIVE KNOCK RET DEGREES
A72D      subb    L81DF      ;SUBTRACT RECOVERY RATE
A730      bcc     LA73F      ;BRANCH IF NO UNDERFLOW, ELSE

```

| | | | |
|------|-------------|--------------------|--|
| A732 | LA732:clr | b | ;CLEAR B |
| A733 | bra | LA73F | ;GO STORE CUMMULATIVE KNOCK RETARD DEGREES |
| A735 | LA735:bclr | *L00DA,#0x40 | ;CLEAR BIT 6 |
| A738 | clrb | | ; |
| A739 | stab | L017F | ; |
| A73C | stab | L0180 | ;STORE KNOCK RETARD ACCUMULATOR |
| A73F | LA73F:stab | L017E | ;STORE CUMMULATIVE KNOCK RET DEGREES |
| A742 | clra | | ;CLEAR |
| A743 | brset | *L009A,#0x01,LA753 | ;BRANCH IF NEG D-RPM > 50 RPM,ELSE |
| A747 | brclr | *L008F,#0x40,LA753 | ;BRANCH IF IDLE COND NOT MET, ELSE |
| A74B | ldaa | *L00A6 | ;LOAD CLNT TEMP (DEFAULTED) |
| A74D | ldx | #0x840A | ;SPARK RET VS COOLANT TEMP |
| A750 | jsr | L99D7 | ;2D LOOKUP |
| A753 | LA753:staa | L0300 | ;SPARK RETARD VS COOLANT TEMP TERM |
| A756 | jsr | LF531 | ;ESC ENABLE ROUTINE |
| A759 | clra | | ;CLEAR A |
| A75A | brclr | *L00DB,#0x80,LA77C | ;BRANCH IF A/C CLUTCH OFF, ELSE |
| A75E | ldab | *L00AD | ;LOAD RPM/12.5 |
| A760 | subb | *L00DE | ;SUBTRACT DESIRED IDLE |
| A762 | bcs | LA77C | ;BRANCH IF RPM < DESIRED, ELSE |
| A764 | cmpb | L81E1 | ;THRESHOLD |
| A767 | bis | LA77C | ;BRANCH IF ERROR <= THRESHOLD, ELSE |
| A769 | ldaa | L81E2 | ;MULTIPLIER |
| A76C | brclr | *L008E,#0x01,LA773 | ;BRANCH IF NOT P/N MODE, ELSE |
| A770 | ldaa | L81E3 | ;MULTIPLIER |
| A773 | LA773:mul | | ;A * B |
| A774 | lsld | | ;MUL BY 2 |
| A775 | bcs | LA77A | ;BRANCH IF OVERFLOW, ELSE |
| A777 | lsld | | ;MUL BY 2 |
| A778 | bcc | LA77C | ;BRANCH IF NO OVERFLOW, ELSE |
| A77A | LA77A:ldaa | #0xFF | ;LOAD 255 |
| A77C | LA77C:staa | L02E0 | ;STORE POS RPM ERROR SPARK TERM |
| A77F | clrb | | ; |
| A780 | ldaa | *L00BD | ;FILTERED MPH |
| A782 | brset | *L008E,#0x10,LA78B | ;BRANCH IF NOT 5 TH GEAR, ELSE |
| A786 | suba | L8B80 | ;118 |
| A789 | bra | LA797 | |
| A78B | LA78B:brset | *L008E,#0x08,LA794 | ;BRANCH IF NOT 4 TH GEAR, ELSE |
| A78F | suba | L8B81 | ; |
| A792 | bra | LA797 | |
| A794 | LA794:suba | L8B82 | ; |
| A797 | LA797:bcs | LA79D | ;BRANCH IF FMPH < CAL, ELSE |
| A799 | ldab | L8B84 | ;HI SPEED SPARK RETARD MULTIPLIER |
| A79C | mul | | ;A * B |
| A79D | LA79D:stab | L0179 | ;SAVE SPARK RETARD |
| A7A0 | ldx | #0x0000 | ;START WITH 0000 |
| A7A3 | brset | *L009A,#0x01,LA7B1 | ;BRANCH IF NEG D-RPM > 50 RPM, ELSE |
| A7A7 | ldaa | *L00BD | ;FILTERED MPH |
| A7A9 | cmpa | L81D0 | ;MIN VALUE |
| A7AC | bhi | LA7B1 | ;BRANCH IF HIGHER, ELSE |
| A7AE | ldx | L81D1 | ;INITIAL VALUE |
| A7B1 | LA7B1:ldaa | L0174 | ; |
| A7B4 | ldab | L016B | ;LOAD BASE SPARK ADVANCE |
| A7B7 | cba | | ;COMPARE |
| A7B8 | bcc | LA7BF | ;BRANCH IF L0174 >= L016B, ELSE |
| A7BA | inca | | ;INCREMENT L0174 |
| A7BB | staa | L0174 | ;STORE |
| A7BE | tab | | ;TRANSFER TO B REG |
| A7BF | LA7BF:abx | | ;ADD TO X |
| A7C0 | brclr | *L0092,#0x02,LA7C8 | ;WILL ALWAYS BE CLEAR |
| A7C4 | ldab | L01DF | ;THIS WILL ALWAYS BE ZERO |

| | | | |
|------|------------|--------------------|---|
| A7C7 | abx | | ;ADD TO X |
| A7C8 | LA7C8:ldab | L016C | ;SPARK COOLANT CORRECTION TERM |
| A7CB | abx | | ;ADD TO X |
| A7CC | ldab | L016F | ;SPARK START-UP CORRECTION TERM |
| A7CF | abx | | ;ADD TO X |
| A7D0 | ldab | L016E | ;SPARK IAT CORRECTION TERM |
| A7D3 | abx | | ;ADD TO X |
| A7D4 | ldab | L016D | ;EGR SPARK CORRECTION |
| A7D7 | abx | | ;ADD TO X |
| A7D8 | ldab | L0170 | ;TCC LOCKED SPK CORRECTION TERM |
| A7DB | abx | | ;ADD TO X |
| A7DC | ldab | L0178 | ;PE SPARK ADVANCE |
| A7DF | abx | | ;ADD TO X |
| A7E0 | ldab | L0171 | ;LAUNCH MODE SPARK TERM |
| A7E3 | abx | | ;ADD TO X |
| A7E4 | pshx | | ;SAVE X ON STACK |
| A7E5 | tsx | | ;X POINTS TO STACK |
| A7E6 | ldd | 0x00,x | ;GET INTO D REG |
| A7E8 | subd | #0x0258 | ;SUBTRACT |
| A7EB | std | 0x00,x | ;STORE TOTAL SPARK ADVANCE |
| A7ED | ldx | #0x0000 | ;START WITH 0000 |
| A7F0 | brset | *L0092,#0x02,LA7F8 | ;WILL ALWAYS BE CLEAR |
| A7F4 | ldab | L01DF | ;THIS WILL ALWAYS BE ZERO |
| A7F7 | abx | | ;ADD TO X |
| A7F8 | LA7F8:ldab | L0179 | ;HIGH SPEED SPARK RETARD |
| A7FB | abx | | ;ADD TO X |
| A7FC | ldab | L02E0 | ;POSITIVE IDLE RPM ERROR SPARK TERM |
| A7FF | abx | | ;ADD TO X |
| A800 | ldab | L0289 | ;TORQUE MGMT SPARK RETARD |
| A803 | abx | | ;ADD TO X |
| A804 | ldab | L0300 | ;SPARK RET VS COOLANT TEMP TERM |
| A807 | abx | | ;ADD TO X |
| A808 | ldab | L017E | ;CUMULATIVE KNOCK RETARD DEGREES |
| A80B | lsrb | | ;DIV BY 2 |
| A80C | abx | | ;ADD TO X |
| A80D | ldab | *L00F3 | ;LOAD CURRENT KNOCK RETARD DEGREES |
| A80F | lsrb | | ;DIV BY 2 |
| A810 | abx | | ;ADD TO X |
| A811 | pshx | | ;SAVE ON STACK |
| A812 | tsx | | ;X POINTS TO STACK |
| A813 | ldd | 0x02,x | ;LOAD TOTAL SPARK ADVANCE |
| A815 | subd | 0x00,x | ;SUBTRACT TOTAL RETARD |
| A817 | pulx | | ;RESTORE X |
| A818 | tsx | | ;X POINTS TO STACK |
| A819 | std | 0x00,x | ;STORE |
| A81B | tst | L02EE | ;IDLE SPARK TERM |
| A81E | beq | LA82B | ;BRANCH IF IDLE SPARK TERM = ZERO, ELSE |
| A820 | cpd | L81EB | ;COMPARE MAX VALUE |
| A824 | bis | LA82B | ;BRANCH IF <= MAX VALUE, ELSE |
| A826 | ldd | L81EB | ;LOAD MAX |
| A829 | std | 0x00,x | ;STORE TOTAL SPARK |
| A82B | LA82B:clra | | ; |
| A82C | ldab | L027A | ;LOAD TABLE LOOKUP RESULT |
| A82F | addb | L0288 | ;ADD |
| A832 | beq | LA844 | ;BRANCH IF Z, ELSE |
| A834 | negb | | ;INVERT A |
| A835 | negb | | ;INVERT B |
| A836 | sbca | #0x00 | ;ROUND |
| A838 | addb | L016B | ;ADD BASE SPARK ADVANCE |
| A83B | adca | #0x00 | ;ROUND |
| A83D | cpd | 0x00,x | ;COMPARE TOTAL SPARK |
| A840 | bgt | LA844 | ;BRANCH IF > SPARK ON STACK, ELSE |
| A842 | std | 0x00,x | ;STORE |
| A844 | LA844:ldab | L030E | ;A SPARK TERM, FROM L0123+SPK REF ANGLE |

```

A847      beq      LA85C      ;BRANCH IF Z, ELSE
A849      clra      ;CLEAR
A84A      cpd      0x00,x    ;TOTAL SPARK ON STACK
A84D      bpl      LA851      ;BRANCH IF > SPARK ON STACK
A84F      bmi      LA856      ;BRANCH IF < SPARK ON STACK
A851      LA851:clr    L030E    ;CLEAR SPARK TERM
A854      bra      LA85C

A856      LA856:incb      ;INCREMENT
A857      stab      L030E    ;STORE SPARK TERM
A85A      std      0x00,x    ;STORE TOTAL SPARK ON STACK

;-----
; CHECK IF MODE 4
;-----
A85C      LA85C:brclr   *L0084,#0x80,LA886    ;BRANCH IF NOT MODE 4, ELSE
A860      ldaa      L0234      ;MODE 4 CONTROL BYTE
A863      bita      #0x08      ;BIT 3?
A865      beq      LA886      ;BRANCH IF BIT 3 CLEAR, ELSE
A867      ldab      L0237      ;GET COMMANDED SPARK ADVANCE
A86A      bita      #0x10      ;BIT 4?
A86C      bne      LA879      ;BRANCH IF BIT 4 SET, ELSE
A86E      anda      #0x20      ;BIT 5?
A870      beq      LA884      ;BRANCH IF BIT 5 CLEAR, ELSE
A872      clra      ;CLEAR A
A873      nega      ;INVERT A
A874      negb      ;INVERT B
A875      sbca      #0x00      ;ROUND
A877      bra      LA884      ;STORE SPARK ADVANCE

A879      LA879:anda    #0x20      ;BIT 5?
A87B      beq      LA882      ;BRANCH IF BIT 5 CLEAR, ELSE
A87D      clra      ;CLEAR A
A87E      nega      ;INVERT
A87F      negb      ;INVERT
A880      sbca      #0x00      ;ROUND
A882      LA882:addd    0x00,x    ;ADD TOTAL SPARK ON STACK
A884      LA884:std     0x00,x    ;STORE TOTAL SPARK ON STACK
A886      LA886:ldd     0x00,x    ;GET SPK ADV REL TO TDC FROM STK
A888      pulx      ;RESTORE X
A889      std      L0290      ;SPARK ADVANCE REL TO TDC
A88C      brclr     *L0095,#0x82,LA896    ;BRANCH IF HUD DEVICE NOT CONNECTED, ELSE
A890      jsr      L5009      ;UPDATE HUD
A893      std      L0290      ;SPARK ADVANCE REL TO TDC, UNLIMITED
A896      LA896:ldd     L0290      ;SPARK ADVANCE REL TO TDC, UNLIMITED
A899      subb      L81C1      ;SUBTRACT SPARK REF ANGLE
A89C      sbca      #0x00      ;ROUND
A89E      cpd      L81CC      ;COMPARE MAX SA
A8A2      blt      LA8A9      ;BRANCH IF < MAX, ELSE
A8A4      ldd      L81CC      ;LOAD MAX SA
A8A7      bra      LA8B2      ;

A8A9      LA8A9:cpd     L81CE      ;MAX RETARD
A8AD      bgt      LA8B2      ;BRANCH IF > 81CE, ELSE
A8AF      ldd      L81CE      ;LOAD MAX RETARD
A8B2      LA8B2:brclr   *L009E,#0xA0,LA8C1    ;%10100000
A8B6      clr      L0174      ;
A8B9      clra      ;
A8BA      ldab      L81C1      ;LOAD SPARK REF ANGLE
A8BD      nega      ;A REG = FF
A8BE      negb      ;INVERT SPARK REF ANGLE
A8BF      sbca      #0x00      ;ROUND
A8C1      LA8C1:std     L0122      ;STORE SPARK REL TO REF PULSE
A8C4      ldx      *L00C8      ;3/4 3x REF PERIOD

```

```

A8C6      cpx      L81EF      ;0x0333
A8C9      bhi      LA8D1      ;BRANCH IF REF PERIOD > 0333, ELSE
A8CB      ldaa     L0123      ;LOAD SPARK REL TO REF PULSE LSB
A8CE      nega     ;INVERT
A8CF      bra      LA8EB      ;GO STORE L030B

A8D1      LA8D1:ldd      L81CA      ;0x0900
A8D4      ld      *L00C8      ;3/4 3x REF PERIOD
A8D6      idiv     ;0x0900/L00C8
A8D7      lsld     ;MUL REMAINDER BY 2
A8D8      bcs      LA8DF      ;BRANCH IF OVERFLOW, ELSE
A8DA      cpd      *L00C8      ;3/4 3x A REF PERIOD
A8DD      bcs      LA8E0      ;BRANCH IF REMAINDER < REF PERIOD, ELSE
A8DF      LA8DF:inx     ;ROUND QUOTIENT
A8E0      LA8E0:pshx     ;TRANSFER QUOTIENT TO ACCD
A8E1      pula     ;
A8E2      pulb     ;
A8E3      ldaa     L0123      ;SPARK REL TO REF PULSE LSB
A8E6      nega     ;2S COMP
A8E7      sba      ;SUBTRACT LSB OF QUOTIENT
A8E8      bcc      LA8EB      ;BRANCH IF 2S COMP L0123 > LSB, ELSE
A8EA      clra     ;CLEAR A
A8EB      LA8EB:staa     L030B      ;2S COMP L0123 - LSB OF L81CA/L00C8
A8EE      ldd      #0x0000      ;LOAD 0000 - RESET COUNTERS
A8F1      std      L3FE8      ;EST FALL
A8F4      jsr      LA99A      ;SHORT DELAY
A8F7      std      L3FE6      ;LAST EST FALL
A8FA      jsr      LA99A      ;SHORT DELAY
A8FD      ldd      L0120      ;LOAD TOTAL DWELL
A900      std      L3FDC      ;STORE SPARK PW
A903      jsr      LA99A      ;SHORT DELAY
A906      ldd      #0x4000      ;
A909      std      L3FF6      ;? EST FALL COUNTER
A90C      jsr      LA912      ;DO IGN CNTL ERROR CHECK
A90F      jmp      LAA19      ;

;-----
; DTC P1350 - IGNITION CONTROL ERROR CHECK
;-----
A912      LA912:brset   *L0017,#0x10,LA94E      ;BRANCH IF IGN CNTL BYPASS ERROR, ELSE
A916      brset   *L0086,#0x40,LA97E      ;BRANCH IF M42A PASSED, ELSE
A91A      brset   *L0016,#0x20,LA972      ;BRANCH IF IGN CNTL BYPASS ERROR, ELSE
A91E      ldaa     L8018      ;OPTION WORD - TIS SET
A921      beq      LA93B      ;BRANCH IF = ZERO, ELSE
A923      ld      L3FC8      ;LOAD EST VOLTAGE
A926      stx      L0298      ;STORE PREV EST VOLTAGE
A929      ldaa     *L0031      ;TIMER
A92B      cmpa     #0x02      ;2 ?
A92D      bhi      LA964      ;BRANCH IF > 2, ELSE
A92F      beq      LA934      ;BRANCH IF = 2, ELSE
A931      jmp      LA999      ;RETURN

A934      LA934:cpx      L911A      ;5 VOLTS
A937      bcs      LA964      ;BRANCH IF < 5 VOLTS, ELSE
A939      bra      LA972

;-THIS NOT USED UNLESS OPTION L8018 IS CLEAR-
A93B      LA93B:brclr   *L009A,#0x10,LA996      ;BRANCH IF ? RUN SPARK NOT ALLOWED, ELSE
A93F      ld      L3FC8      ;LOAD EST VOLTAGE
A942      cpx      L0298      ;COMPARE PREV EST VOLTAGE
A945      bne      LA950      ;BRANCH IF THEY'RE NOT EQUAL, ELSE
A947      brset   *L0095,#0x20,LA964      ;BRANCH IF BIT 5
A94B      bset    *L0095,#0x20      ;SET BIT 5
A94E      LA94E:bra     LA996      ;CLEAR EST ENABLED

```

```

A950    LA950:ldaa    L0128                ;MALF TIMER
A953          stx     L0298                ;STORE PREV EST VOLTAGE
A956          cmpa    L9119                ;0x04 TIME LIMIT
A959          bhi     LA972                ;BRANCH IF > 0x04, ELSE
A95B          inca    ;INCREMENT TIMER
A95C          staa    L0128                ;STORE MALF TIMER
A95F          bclr    *L0095,#0x20        ;CLEAR BIT 5
A962          bra     LA996                ;CLEAR EST ENABLED

A964    LA964:ldx     L3FEC                ;LAST REF
A967          bset    *L0086,#0x40        ;SET M42A PASSED
A96A          bclr    *L0016,#0x20        ;CLEAR IGN CNTL BYPASS ERROR FLAG
A96D          stx     L3FE4                ;B CNTR, START OF NEXT DWELL
A970          bra     LA97E                ;GO ENABLE EST

A972    LA972:ldx     #0x90AB              ;INDEX
A975          ldaa    #0x20                ;MASK
A977          ldab    #0x03                ;OFFSET - 90AE, L0016
A979          jsr     LF87D                ;GO UPDATE MALF FLAGS
A97C          bra     LA996                ;CLEAR EST ENABLED

A97E    LA97E:brclr    *L0084,#0x80,LA989 ;BRANCH IF NOT MODE 4, ELSE
A982          ldaa    L0230                ;? MODE 4 CONTROL WORD
A985          anda    #0x08                ;BIT 3 ?
A987          bne     LA996                ;BRANCH IF BIT 3 SET, ELSE
A989    LA989:ldd     L3FFC                ;CPU CNTL REG
A98C          bset    *L0089,#0x04        ;ENABLED EST
A98F          orab    #0x10                ;SET BIT 4
A991          std     L3FFC                ;CPU CNTL REG
A994          bra     LA999                ;RETURN

A996    LA996:bclr    *L0089,#0x04        ;DISABLE EST
A999    LA999:rts

;-----
A99A    LA99A:rts                ;ONLY USED AS A DELAY

;-----
; JUMP HERE IF 12.5 MSEC AND 8DCF NOT 0x55
; SAME AS $5B
;-----

A99B    LA99B:ldaa    #0xFF                ;
A99D          staa    L400B                ;COP ARM
A9A0          ldd     #0x0000              ;CLEAR THE FOLLOWING:
A9A3          std     *L00AB                ;NLRPMX
A9A5          staa    *L00AD                ;RPM/12.5
A9A7          staa    L012C                ;PREV RPM/12.5
A9AA          std     *L00ED                ;1 SECOND TIMER
A9AC          staa    *L00EF                ;
A9AE          std     L0106                ;ASYNC PW
A9B1          std     L0104                ;BPW
A9B4          brset    *L0012,#0x08,LA9BA  ;BRANCH IF BIT 3, ELSE
A9B8          std     *L0032                ;CLEAR RUN TIME
A9BA    LA9BA:jsr     LA912                ;DO IGNITION CONTROL ERROR CHECK
A9BD          brset    *L0016,#0x20,LA9C5  ;BRANCH IF IGN CNTL BYPASS ERROR, ELSE
A9C1          brset    *L0098,#0x08,LA9CA  ;BRANCH IF ? REF PULSE OCCURED, ELSE
A9C5    LA9C5:bclr    *L0089,#0x04        ;DISABLE EST
A9C8          bra     LAA19                ;GO UPDATE ALDL FLAGWORDS

A9CA    LA9CA:ldd     #0x0010              ;16
A9CD          std     L0120                ;STORE TOTAL DWELL
A9D0          ldx     #0x8332              ;TABLE ADDRESS
A9D3          ldd     *L00EB                ;LOAD 16 BIT RPM

```

```

A9D5          cpd      #0x01FF          ;511 RPM
A9D9          bls      LA9E1            ;BRANCH IF <= 511 RPM, ELSE
A9DB          ldx      #0x839B          ;TABLE ADDRESS
A9DE          subd     #0x0200          ;512 RPM
A9E1  LA9E1:lsrd      ;RESCALE
A9E2          tsta     ;TEST MSB
A9E3          beq      LA9E7            ;BRANCH IF Z, ELSE
A9E5          ldab     #0xFF            ;LOAD 255
A9E7  LA9E7:ldaa      L012A            ;TRUNCATED CTS FOR TABLE LOOKUPS
A9EA          lsra     ;RESCALE
A9EB          jsr      L995A            ;3D LOOKUP
A9EE          tab      ;TRANSFER LOOKUP RESULT TO B REG
A9EF          ldaa     #0xFF            ;LOAD 255
A9F1          std      L0122            ;STORE L0122:L0123
A9F4          ldaa     L0123            ;LOAD LSB
A9F7          nega     ;2'S COMP
A9F8          staa     L030B            ;STORE
A9FB          ldd      #0x0000          ;LOAD 0000
A9FE          std      L3FE8            ;? CURRENT EST FALL
AA01          jsr      LA99A            ;SHORT DELAY
AA04          std      L3FE6            ;SPARK
AA07          jsr      LA99A            ;SHORT DELAY
AA0A          ldd      L0120            ;TOTAL DWELL
AA0D          std      L3FDC            ;SPARK DWELL (EST PW)
AA10          jsr      LA99A            ;SHORT DELAY
AA13          ldd      #0x7FFF          ;
AA16          std      L3FF6            ;

;-----
; UPDATE ALDL FLAGWORDS
;-----

AA19  LAA19:clra          ;
AA1A          brclr    *L0018,#0x20,LAA20 ;BRANCH IF BIT 5 CLEAR, ELSE
AA1E          oraa     #0x01            ;SET BIT 0
AA20  LAA20:brclr    *L008E,#0x01,LAA26 ;BRANCH IF NOT P/N MODE, ELSE
AA24          oraa     #0x02            ;SET BIT 1
AA26  LAA26:brset    *L0086,#0x20,LAA2C ;BRANCH IF A/C OFF CLUTCH, ELSE
AA2A          oraa     #0x04            ;SET BIT 2
AA2C  LAA2C:brclr    *L0084,#0x20,LAA32 ;BRANCH IF ALDL TESTER NOT IN CONTROL, ELSE
AA30          oraa     #0x20            ;SET BIT 5
AA32  LAA32:brclr    *L0091,#0x01,LAA38 ;BRANCH IF SES LIGHT OFF, ELSE
AA36          oraa     #0x40            ;SET BIT 6
AA38  LAA38:brclr    *L00D9,#0x40,LAA3E ;BRANCH IF MOTOR NOT RESET TO 0, ELSE
AA3C          oraa     #0x80            ;SET BIT 7
AA3E  LAA3E:staa     L02A6            ;STORE ALDL FLAGWORD
AA41          clra     ;
AA42          brclr    *L0018,#0x40,LAA48 ;BRANCH IF NO PASSKEY II ERROR, ELSE
AA46          oraa     #0x01            ;SET BIT 0
AA48  LAA48:brclr    *L0092,#0x04,LAA4E ;BRANCH IF VATS OKAY, ELSE
AA4C          oraa     #0x02            ;SET BIT 1
AA4E  LAA4E:brclr    *L0014,#0x20,LAA54 ;BRANCH IF NOT "NO VSS SIGNAL" MALF, ELSE
AA52          oraa     #0x08            ;SET BIT 3
AA54  LAA54:brset    *L008E,#0x80,LAA5A ;BRANCH IF A/C NOT REQUESTED, ELSE
AA58          oraa     #0x20            ;SET BIT 5
AA5A  LAA5A:brclr    *L0013,#0x60,LAA60 ;BRANCH IF NO CTS MALFS, ELSE
AA5E          oraa     #0x40            ;SET BIT 6
AA60  LAA60:staa     L02A7            ;STORE ALDL FLAGWORD
AA63          clra     ;
AA64          ldab     L0136            ;
AA67          beq      LAA6B            ;
AA69          oraa     #0x01            ;SET BIT 0
AA6B  LAA6B:brclr    *L008E,#0x01,LAA71 ;BRANCH IF NOT P/N MODE, ELSE
AA6F          oraa     #0x02            ;SET BIT 1
AA71  LAA71:brset    *L0086,#0x20,LAA77 ;BRANCH IF A/C CLUTCH OFF, ELSE

```

```

AA75          oraa      #0x04                      ;SET BIT 2
AA77  LAA77:brclr    *L0014,#0x20,LAA7D          ;BRANCH IF NOT "NO VSS SIGNAL" MALF, ELSE
AA7B          oraa      #0x08                      ;SET BIT 3
AA7D  LAA7D:brclr    *L0084,#0x20,LAA83          ;BRANCH IF ALDL TESTER NOT IN CONTROL, ELSE
AA81          oraa      #0x20                      ;SET BIT 5
AA83  LAA83:brclr    *L0091,#0x01,LAA89          ;BRANCH IF SES LIGHT OFF, ELSE
AA87          oraa      #0x40                      ;SET BIT 6
AA89  LAA89:brclr    *L008E,#0x40,LAA8F          ;BRANCH IF BRAKE NOT APPLIED, ELSE
AA8D          oraa      #0x80                      ;SET BIT 7
AA8F  LAA8F:staa     L02A8                        ;STORE ALDL FLAGWORD

;-----
; MAJOR IAC ROUTINE
; EXECUTED EVERY 12.5 MSEC
;-----
AA92          brset    *L0086,#0x80,LAABD          ;BRANCH IF ENGINE RUNNING, ELSE
AA96          bclr     *L00D9,#0x01              ;CLEAR A/C SLUGGING TEMP MET
AA99          ldaa      *L00F0                    ;IATMAT
AA9B          cmpa     L8D6B                        ;60
AA9E          bls      LAABD                        ;BRANCH IF <= CAL, ELSE
AAA0          ldaa      *L006F                    ;IATMAT
AAA2          cmpa     L8D6C                        ;80
AAA5          bcs      LAABD                        ;BRANCH IF < CAL, ELSE
AAA7          ldaa      *L00A6                    ;LOAD CLNT TEMP (DEFAULTED)
AAA9          cmpa     L8D6D                        ;40 DEG C
AACAC          bcc      LAABD                        ;BRANCH IF >= CAL, ELSE
AAAE          ldaa      *L00F9                    ;A/C PRESSURE
AAB0          cmpa     L8D72                        ;121
AAB3          bhi      LAABD                        ;BRANCH IF > CAL, ELSE
AAB5          cmpa     L8D73                        ;30
AAB8          bcs      LAABD                        ;BRANCH IF < CAL, ELSE
AABA          bset     *L00D9,#0x01              ;SET A/C SLUGGING TEMP MET
AABD  LAABD:brclr    *L00D9,#0x04,LAAC4          ;BRANCH IF IAC RESET NOT IN PROGRESS, ELSE
AAC1          jmp      LAB44                        ;

AAC4  LAAC4:brclr    *L0095,#0x82,LAADA          ;BRANCH IF HUD DEVICE NOT CONNECTED, ELSE
AAC8          ldaa      L8021                        ;OPTION WORD - TIS SET
AACB          beq      LAADA                        ;BRANCH IF Z, ELSE
AACD          brclr    *L009F,#0x40,LAADA          ;
AAD1          jsr      L5003                        ;UPDATE HUD
AAD4          bclr     *L00DB,#0x80              ;SET A/C CLUTCH OFF REPEAT FLAG
AAD7          jmp      LB744                        ;GO EXERCISE A/C RELAY CONTROL OUTPUT

AADA  LAADA:brclr    *L0095,#0x10,LAEE0          ;BRANCH IF BATT VOLTS NOT < 4.0, ELSE
AADE          bra      LAB20                        ;

AAE0  LAEE0:brclr    *L0084,#0x80,LAB20          ;BRANCH IF NOT MODE 4, ELSE
AAE4          brclr    *L0088,#0x01,LAAF1          ;BRANCH IF NOT "ALL PWM OUTPUTS COMMAND ON"
AAE8          ld      #0xDFFF                      ;MAX DUTY CYCLE
AAEB          stx      L3FD2                        ;STORE A/C RELAY CONTROL OUTPUT
AAEE          jmp      LB6F1                        ;GO INCREMENT IAC POSITION

AAF1  LAAF1:ldaa     #0x20                          ;BIT 5
AAF3          anda     L0230                        ;? MODE 4 CONTROL BYTE
AAF6          beq      LAB0B                        ;BRANCH IF BIT 5 NOT SET, ELSE
AAF8          brset    *L0092,#0x20,LAB20          ;BRANCH IF BIT 5, ELSE
AAFC          bset     *L0092,#0x20              ;SET BIT 5
AAFF          bset     *L00D9,#0x04              ;SET IAC MOTOR RESET IN PROGRESS
AB02          bset     *L0057,#0xFF              ;SET IAC TO MAX
AB05          bset     *L00DC,#0xFF              ;
AB08          jmp      LB73E                        ;SKIP IAC ROUTINE, GO CLEAR BIT 4 L009A

AB0B  LAB0B:bclr     *L0092,#0x20              ;CLEAR BIT 5
AB0E          ldaa     L0234                        ;MODE 4 CONTROL BYTE

```

| | | | |
|------|-------------|--------------------|--|
| AB11 | rora | | ;SHIFT RIGHT |
| AB12 | bcc | LAB20 | ;BRANCH IF NO OVERFLOW (B0 CLEAR), ELSE |
| AB14 | rora | | ;SHIFT RIGHT AGAIN |
| AB15 | bcs | LAB20 | ;BRANCH IF OVERFLOW (B1 SET), ELSE |
| AB17 | bclr | *L00DB,#0x80 | ;SET A/C CLUTCH OFF REPEAT FLAG |
| AB1A | ldaa | L0235 | ;LOAD COMMANDED DC |
| AB1D | jmp | LB744 | ;GO TURN A/C RELAY OFF |
| AB20 | LAB20:brclr | *L0086,#0x80,LAB27 | ;BRANCH IF ENGINE NOT RUNNING, ELSE |
| AB24 | jmp | LABD1 | ; |
| AB27 | LAB27:brclr | *L0095,#0x10,LAB73 | ;BRANCH IF BATT VOLTS NOT < 4.0, ELSE |
| AB2B | brset | *L00D9,#0x10,LAB6D | ;BRANCH IF RESET COMPLETE, ELSE |
| AB2F | ldaa | *L00A6 | ;LOAD CLNT TEMP (DEFAULTED) |
| AB31 | cmpa | L8C6D | |
| AB34 | bcs | LAB3A | |
| AB36 | brclr | *L0030,#0xFE,LAB3E | ;BRANCH IF CRANKING TIMER NOT MAXED, ELSE |
| AB3A | LAB3A:brclr | *L0012,#0x40,LAB6D | ;BRANCH IF IAC RESET NOT ENABLED, ELSE |
| AB3E | LAB3E:bset | *L00D9,#0x04 | ;SET IAC MOTOR RESET IN PROGRESS |
| AB41 | bset | *L00DC,#0xFF | ;SET ? IAC MIN POSITION TO MAX |
| AB44 | LAB44:ldaa | *L00DC | ; |
| AB46 | brset | *L00D9,#0x02,LAB5B | ;BRANCH IF MOTOR HAS BEEN RESET TO 0, ELSE |
| AB4A | cmpa | #0x06 | ;6 ? |
| AB4C | bcc | LAB6A | ;BRANCH IF >= 6, ELSE |
| AB4E | clr | L0057 | ;MAKE IAC POSITION 00 |
| AB51 | ldab | L0175 | ;IAC STEPS |
| AB54 | stab | *L00DC | ; |
| AB56 | bset | *L00D9,#0x02 | ;SET MOTOR RESET TO 0 DURING RESET |
| AB59 | bra | LAB6A | |
| AB5B | LAB5B:bne | LAB6A | ;BRANCH IF L00DC NZ, ELSE |
| AB5D | bclr | *L0012,#0x40 | ;CLEAR IAC RESET ENABLED |
| AB60 | bclr | *L00D9,#0x06 | ;CLEAR MOTOR HAS BEEN RESET TO 0 DURING |
| | | | ; RESET AND MOTOR RESET IN PROGRESS |
| AB63 | bset | *L00D9,#0x10 | ;SET IAC RESET COMPLETE |
| AB66 | brclr | *L0095,#0x10,LABD1 | ;BRANCH IF BATT VOLTS NOT < 4.0, ELSE |
| AB6A | LAB6A:jmp | LB73E | ;SKIP THE REST, GO CLEAR BIT 4 L009A |
| AB6D | LAB6D:ldaa | L0175 | ;IAC STEPS |
| AB70 | jmp | LB744 | ;GO EXERCISE A/C RELAY CONTROL OUTPUT |
| AB73 | LAB73:ldx | *L00EB | ;TRANNNY RPM |
| AB75 | cpx | L8C6F | ;500 |
| AB78 | bls | LAB8B | ;BRANCH IF <= 500 RPM, ELSE |
| AB7A | ldaa | *L0030 | ;; CRANKING TIMER |
| AB7C | cmpa | L8C71 | ;0xFF |
| AB7F | bls | LAB8B | ;BRANCH IF <= CAL, ELSE |
| AB81 | ldx | *L00CD | ;LOAD MAF (GM/SEC) |
| AB83 | cpx | L8C72 | ;0x0000 |
| AB86 | bcc | LAB8B | ;BRANCH IF >= 0000, ELSE |
| AB88 | jmp | LB6F1 | ;GO INCREMENT IAC POSITION |
| AB8B | LAB8B:ldaa | *L0083 | ;BATT VOLTS |
| AB8D | cmpa | #0x5F | ;9.5 VOLTS |
| AB8F | bls | LABD1 | ;BRANCH IF <= 9.5 VOLTS, ELSE |
| AB91 | ldaa | L86C9 | ;OPTION WORD |
| AB94 | bita | #0x01 | ;BIT 1 SET? (TIS SET) |
| AB96 | beq | LABA1 | ;BRANCH IF BIT 1 CLEAR, ELSE |
| AB98 | brclr | *L00D9,#0x01,LABA1 | ;BRANCH IF A/C SLUGGING TEMP NOT MET, ELSE |
| AB9C | ldaa | L0175 | ;LOAD IAC STEPS |
| AB9F | bra | LABBB | |
| ABA1 | LABA1:ldaa | *L00A6 | ;LOAD CLNT TEMP (DEFAULTED) |
| ABA3 | ldx | #0x8C7A | ; |

```

ABA6      jsr      L99D7                      ;2D LOOKUP
ABA9      psha                      ;SAVE RESULT ON STACK
ABAA      ldx      #0x8C8B
ABAD      ldaa     *L005A                ;FILTERED ENGINE TORQUE
ABAF      ldab     #0x60                ;OFFSET
ABB1      jsr      L99D3                ;2D LOOKUP
ABB4      pulb                      ;GET PREV LOOKUP RESULT
ABB5      mul                      ;MULTIPLY TABLE LOOKUP RESULTS
ABB6      lsld                      ;MUL BY 2
ABB7      bcc      LABBB                ;BRANCH IF NO OVERFLOW, ELSE
ABB9      ldaa     #0xFF                ;LOAD 255
ABBB      LABBB: psha                    ;SAVE MSB (OR L0175) ON STACK
ABBC      ldaa     *L0076                ;THIS NOT USED
ABBE      ldx      #0x8C96
ABC1      jsr      L99CC                ;2D LOOKUP
ABC4      pulb                      ;GET MSB OF PRODUCT (OR L0175)
ABC5      aba                      ;ADD CURRENT LOOKUP RESULT
ABC6      bcc      LABCA                ;BRANCH IF NO OVERFLOW, ELSE
ABC8      ldaa     #0xFF                ;LOAD 255
ABCA      LABCA: cmpa *L0057                ;IAC POSITION
ABCC      beq      LABD1                ;BRANCH IF THEY'RE EQUAL, ELSE
ABCE      jmp      LB744                ;GO EXERCISE A/C RELAY CONTROL OUTPUT

ABD1      LABD1: ldy      #0x0065                      ;INDEX
ABD5      brset   *L0099,#0x08,LABE2
ABD9      brclr   *L0086,#0x20,LABE6
ABDD      ldaa     L02AF
ABE0      bne     LABE6
ABE2      LABE2: iny
ABE4      iny
ABE6      LABE6: brset *L0086,#0x80,LABED
ABEA      jmp      LAFE6

ABED      LABED: brset *L008F,#0x02,LAC02
ABF1      brclr   *L0015,#0x10,LAC02
ABF5      bset    *L008F,#0x02
ABF8      ldaa     L8DD5
ABFB      staa     *L0065
ABFD      ldaa     L8DD6
AC00      staa     *L0067
AC02      LAC02: ldd    *L00CD
AC04      cpd      #0x197E
AC08      bls      LAC0E
AC0A      ldaa     #0xFF
AC0C      bra      LAC13

AC0E      LAC0E: lsld                      ;MUL BY 4
AC0F      lsld                      ;
AC10      addd     *L00CD                ;ADD MAF (GM/SEC)
AC12      lsld                      ;MUL BY 2
AC13      LAC13: staa   L0163
AC16      ldab     L8010
AC19      bmi      LAC22
AC1B      brclr   *L008E,#0x01,LAC22
AC1F      jmp      LACB2

;-----
; CONDITIONS MET FOR FILTERED
; COMMANDED AIRFLOW ?
;-----

AC22      LAC22: ldab    *L00BF                ;MPH*3
AC24      bne      LACA4                    ;BRANCH IF NOT Z, ELSE
AC26      ldab     *L00AA
AC28      cmpb     L8CE2                    ;%TPS
                                           ;1.2 %TPS

```

| | | | | |
|-------------------------------|-------------|--------------------|--|--|
| AC2B | bhi | LACA4 | | ;BRANCH IF > CAL, ELSE |
| AC2D | ldab | *L00A6 | | ;LOAD CLNT TEMP (DEFAULTED) |
| AC2F | cmpb | L8DD4 | | ;60 DEG C |
| AC32 | bls | LACA4 | | ;BRANCH IF <= CAL, ELSE |
| AC34 | ldab | *L00DE | | ;LOAD DESIRED IDLE |
| AC36 | addb | L8DDE | | ;62.5 RPM |
| AC39 | cmpb | *L00AD | | ;COMPARE RPM/12.5 |
| AC3B | bls | LACA4 | | ;BRANCH IF <= RPM/12.5, ELSE |
| AC3D | ldab | *L00DE | | ;DESIRED IDLE RPM |
| AC3F | subb | L8DDF | | ;12.5 RPM |
| AC42 | cmpb | *L00AD | | ;COMPARE RPM/12.5 |
| AC44 | bhi | LACA4 | | ;BRANCH IF > RPM/12.5, ELSE |
| AC46 | ldaa | *L00E0 | | ;LOAD TIMER |
| AC48 | cmpa | L8DD7 | | ;20 |
| AC4B | bcc | LAC52 | | ;BRANCH IF >= CAL, ELSE |
| AC4D | inc | L00E0 | | ;INCREMENT TIMER |
| AC50 | bra | LACB2 | | ;GO CALCULATE COMMANDED AIRFLOW |
| AC52 | LAC52:ldaa | L017D | | ;TIMER |
| AC55 | bne | LAC5D | | ;BRANCH IF NOT Z, ELSE |
| AC57 | ldaa | L0163 | | ;LOW FLOW MAF |
| AC5A | staa | L017C | | ;UNFILTERED MAF/10 |
| AC5D | LAC5D:inc | L017D | | ;INCREMENT TIMER |
| AC60 | ldaa | L017D | | ;TIMER |
| AC63 | cmpa | L8DD8 | | ;20 |
| AC66 | bcs | LACB2 | | ;BRANCH IF < CAL, ELSE |
| AC68 | clr | L00E0 | | ;CLEAR TIMER |
| AC6B | clr | L017D | | ;CLEAR TIMER |
| AC6E | ldaa | L017C | | ;UNFILTERED MAF/10 |
| AC71 | brclr | *L0096,#0x02,LAC7B | | ;BRANCH IF FAN #1 OFF, ELSE |
| AC75 | suba | L8DD2 | | ;SUBTRACT COMMANDED AIRFLOW FOR FAN #1 |
| AC78 | bcc | LAC7B | | ;BRANCH IF NO UNDERFLOW, ELSE |
| AC7A | clra | | | ;CLEAR |
| AC7B | LAC7B:brclr | *L0096,#0x10,LAC85 | | ;BRANCH IF FAN #2 OFF, ELSE |
| AC7F | suba | L8DD3 | | ;SUBTRACT COMMANDED AIRFLOW FOR FAN #2 |
| AC82 | bcc | LAC85 | | ;BRANCH IF NO UNDERFLOW, ELSE |
| AC84 | clra | | | ;CLEAR |
| AC85 | LAC85:pshy | | | ;SAVE INDEX |
| AC87 | pulx | | | ;GET INTO X REG |
| AC88 | pshy | | | ;SAVE INDEX AGAIN |
| AC8A | ldy | #0x8DD9 | | ;FILTER CONSTANT |
| AC8E | jsr | L99F4 | | ;LAG FILTER |
| AC91 | puly | | | ;RESTORE INDEX |
| AC93 | cpd | L8DDA | | ;0x4800 - MAX |
| AC97 | bhi | LACAC | | ;BRANCH IF > MAX, ELSE |
| AC99 | cpd | L8DDC | | ;0x3700 - MIN |
| AC9D | bcc | LACAF | | ;BRANCH IF >= MIN, ELSE |
| AC9F | ldd | L8DDC | | ;LOAD MIN FILTERED AIRFLOW |
| ACA2 | bra | LACAF | | ;GO STORE FILTERED LOW AIRFLOW |
| ACA4 | LACA4:clr | L00E0 | | ;CLEAR TIMER |
| ACA7 | clr | L017D | | ;CLEAR TIMER |
| ACAA | bra | LACB2 | | ;GO CALCULATE COMMANDED AIRFLOW |
| ACAC | LACAC:ldd | L8DDA | | ;LOAD MAX FILTERED AIRFLOW |
| ACAF | LACAF:std | 0x00,y | | ;STORE FILTERED LOW AIRFLOW |
| ;----- | | | | |
| ; DETERMINE COMMANDED AIRFLOW | | | | |
| ;----- | | | | |
| ACB2 | LACB2:ldaa | *L00AD | | ;LOAD RPM/12.5 |
| ACB4 | suba | #0x20 | | ;400 RPM |
| ACB6 | bcc | LACB9 | | ;BRANCH IF >= 400 RPM, ELSE |
| ACB8 | clra | | | ;CLEAR A |

| | | | | |
|------|-------------|-------|--------------------|--|
| ACB9 | LACB9:ls | lsl | | ;RESCALE |
| ACBA | | bcc | LACBE | ;BRANCH IF NO OVERFLOW, ELSE |
| ACBC | | ldaa | #0xFF | ;LOAD MAX |
| ACBE | LACBE:brset | | *L00A1,#0x02,LACC8 | ;BRANCH IF PRNDL -> DRIVE 1, ELSE |
| ACC2 | | brset | *L00A1,#0x04,LACC8 | ;BRANCH IF PRNDL -> DRIVE 2, ELSE |
| ACC6 | | bra | LACD7 | ; |
| ACC8 | LACC8:ldx | | #0x8DFE | ;TABLE ADDRESS |
| ACCB | | ldab | *L008E | ;STATUS FLAG |
| ACCD | | lsrb | | ;DIV BY 4 |
| ACCE | | lsrb | | ; |
| ACCF | | bcs | LACFD | ;BRANCH IF P/N OR NOT 2 ND GEAR (DO LOOKUP) |
| ACD1 | | ldx | #0x8E0F | ;TABLE ADDRESS |
| ACD4 | | lsrb | | ;DIV BY 2 |
| ACD5 | | bcs | LACFD | ;BRANCH IF 3 RD GEAR (DO LOOKUP), ELSE |
| ACD7 | LACD7:ldx | | #0x8E75 | ;TABLE ADDRESS |
| ACDA | | brset | *L008E,#0x01,LACFD | ;BRANCH IF P/N MODE (DO LOOKUP), ELSE |
| ACDE | | ldx | #0x8E64 | ;TABLE ADDRESS |
| ACE1 | | brset | *L00A1,#0x40,LACFD | ;BRANCH IF PRNDL -> REVERSE (DO LKUP), ELSE |
| ACE5 | | ldx | #0x8E20 | ;TABLE ADDRESS |
| ACE8 | | ldab | *L008E | ;STATUS FLAG |
| ACEA | | lsrb | | ;DIV BY 4 |
| ACEB | | lsrb | | ; |
| ACEC | | bcs | LACFD | ;BRANCH IF P/N OR NOT 2 ND GEAR (DO LOOKUP) |
| ACEE | | ldx | #0x8E31 | ;TABLE ADDRESS |
| ACF1 | | lsrb | | ;DIV BY 2 |
| ACF2 | | bcs | LACFD | ;BRANCH IF 3 RD GEAR (DO LOOKUP), ELSE |
| ACF4 | | ldx | #0x8E42 | ;TABLE ADDRESS |
| ACF7 | | lsrb | | ;DIV BY 2 |
| ACF8 | | bcs | LACFD | ;BRANCH IF 4 TH GEAR (DO LOOKUP), ELSE |
| ACFA | | ldx | #0x8E53 | ;TABLE ADDRESS |
| ACFD | LACFD:jsr | | L99D7 | ;2D LOOKUP |
| AD00 | | ldab | 0x00,y | ;GET FILTERED AIRFLOW |
| AD03 | | aba | | ;ADD TABLE RESULT |
| AD04 | | bcc | LAD08 | ;BRANCH IF NO OVERFLOW, ELSE |
| AD06 | | ldaa | #0xFF | ;LOAD 255 |
| AD08 | LAD08:suba | | #0x1E | ;SUBTRACT 30 |
| | | | | ;BUG !!! THERE SHOULD BE A "BCC, ELSE CLRA" |
| AD0A | | brset | *L0015,#0x01,LAD40 | ;BRANCH IF TCC BRAKE SW ERROR, ELSE |
| AD0E | | brclr | *L008E,#0x40,LAD17 | ;BRANCH IF BRAKE NOT APPLIED, ELSE |
| AD12 | | clr | L0190 | ;PREV %TPS |
| AD15 | | bra | LAD40 | ; |
| AD17 | LAD17:ldab | | *L00AA | ;%TPS |
| AD19 | | cmpb | L0190 | ;PREV %TPS |
| AD1C | | bis | LAD21 | ;BRANCH IF TPS DECREASING, ELSE |
| AD1E | | stab | L0190 | ;SAVE %TPS |
| AD21 | LAD21:ldab | | L0190 | ;LOAD PREV %TPS |
| AD24 | | cmpb | L8EBC | ;TPS THRESHOLD FOR THROTTLE FOLLOWER |
| AD27 | | bis | LAD40 | ;BRANCH IF <= CAL, ELSE |
| AD29 | | ldab | *L00BF | ;MPH*3 |
| AD2B | | cmpb | L8CE3 | ;0x03 - ? DEFAULT MPH |
| AD2E | | bis | LAD40 | ;BRANCH IF <= CAL, ELSE |
| AD30 | | ldab | L8010 | ;OPTION WORD - TRANS TYPE - TIS CLEAR |
| AD33 | | bmi | LAD39 | ;BRANCH IF MANUAL TRANS, ELSE |
| AD35 | | brset | *L008E,#0x01,LAD40 | ;BRANCH IF P/N MODE, ELSE |
| AD39 | LAD39:adda | | L0323 | ;ADD THROTTLE FOLLOWER AIRFLOW VS FMPH |
| AD3C | | bcc | LAD40 | ;BRANCH IF NO OVERFLOW, ELSE |
| AD3E | | ldaa | #0xFF | ;LOAD 255 |
| AD40 | LAD40:adda | | L0324 | ;STARTUP AIRFLOW VS TIME AND CTS |
| AD43 | | bcc | LAD47 | ;BRANCH IF NO OVERFLOW, ELSE |
| AD45 | | ldaa | #0xFF | ;LOAD 255 |
| AD47 | LAD47:ldab | | L02C1 | ;TIMER |
| AD4A | | beq | LAD53 | ;BRANCH IF Z, ELSE |

| | | | |
|------|-------------|--------------------|---------------------------------------|
| AD4C | adda | L8EC6 | ;00 |
| AD4F | bcc | LAD53 | ;BRANCH IF NO OVERFLOW, ELSE |
| AD51 | ldaa | #0xFF | ;LOAD 255 |
| AD53 | LAD53:brclr | *L0096,#0x02,LAD5E | ;BRANCH IF FAN #1 OFF, ELSE |
| AD57 | adda | L8DD2 | ;ADD COMMANDED AIRFLOW FOR FAN #1 |
| AD5A | bcc | LAD5E | ;BRANCH IF NO OVERFLOW, ELSE |
| AD5C | ldaa | #0xFF | ;LOAD 255 |
| AD5E | LAD5E:brclr | *L0096,#0x10,LAD69 | ;BRANCH IF FAN #2 OFF, ELSE |
| AD62 | adda | L8DD3 | ;ADD COMMANDED AIRFLOW FOR FAN #2 |
| AD65 | bcc | LAD69 | ;BRANCH IF NO OVERFLOW, ELSE |
| AD67 | ldaa | #0xFF | ;LOAD 255 |
| AD69 | LAD69:ldx | *L00ED | ;1 SECOND TIMER |
| AD6B | cpx | #0x0040 | ;64 SECONDS? |
| AD6E | bcc | LAD7B | ;BRANCH IF MORE THAN 64 SECONDS, ELSE |
| AD70 | psha | | ;SAVE AIRFLOW ON STACK |
| AD71 | ldaa | *L0076 | ;THIS NOT USED |
| AD73 | ldx | #0x8EB6 | ;(ALL 00s) |
| AD76 | jsr | L99CC | ;2D LOOKUP |
| AD79 | pulb | | ;GET AIRFLOW INTO B REG |
| AD7A | aba | | ;ADD AIRFLOW TO TABLE LOOKUP |
| AD7B | LAD7B:staa | *L00F7 | ;STORE COMMANDED AIRFLOW |
| AD7D | brset | *L008E,#0x01,LADC2 | ;BRANCH IF P/N MODE, ELSE |
| AD81 | clr | L02C4 | ; |
| AD84 | ldd | #0xFFFF | ; |
| AD87 | std | L02C8 | ; |
| AD8A | ldaa | L02C3 | ;TIMER |
| AD8D | inca | | ;INCREMENT |
| AD8E | bne | LAD95 | ;BRANCH IF NZ, ELSE |
| AD90 | bset | *L009A,#0x80 | ;SET |
| AD93 | bra | LADB0 | ; |
| AD95 | LAD95:staa | L02C3 | ;SAVE TIMER |
| AD98 | brset | *L009A,#0x01,LADB5 | ;BRANCH IF NEG D-RPM > 50 RPM, ELSE |
| AD9C | ldd | *L00EB | ;RPM |
| AD9E | cpd | L02C6 | ;PREV RPM |
| ADA2 | bls | LADA7 | ;BRANCH IF <= PREV RPM, ELSE |
| ADA4 | std | L02C6 | ;SAVE FOR NEXT PASS |
| ADA7 | LADA7:addd | L8D9A | ;50 RPM |
| ADAA | cpd | L02C6 | ;COMPARE PREV RPM |
| ADAE | bcc | LADB3 | ;BRANCH IF >= PREV RPM, ELSE |
| ADB0 | LADB0:bset | *L009A,#0x01 | ;SET NEG DELTA-RPM > 50 RPM |
| ADB3 | LADB3:bra | LADED | ; |
| ADB5 | LADB5:ldd | *L00EB | ;RPM |
| ADB7 | cpd | L0281 | ;PREV RPM |
| ADBB | bls | LADED | ;BRANCH IF RPM = OR DECREASING, ELSE |
| ADBD | bset | *L009A,#0x80 | ;SET RPM INCREASING |
| ADC0 | bra | LADED | ; |
| ADC2 | LADC2:clra | | ; |
| ADC3 | clrb | | ; |
| ADC4 | staa | L02C3 | ; |
| ADC7 | std | L02C6 | ; |
| ADCA | ldaa | L02C4 | ;TIMER |
| ADCD | inca | | ;INCREMENT |
| ADCE | beq | LADE7 | ;BRANCH IF Z, ELSE |
| ADD0 | staa | L02C4 | ;STORE TIMER |
| ADD3 | ldd | *L00EB | ;16 BIT RPM |
| ADD5 | cpd | L02C8 | ;PREV RPM |
| ADD9 | bcc | LADDE | ;BRANCH IF >= PREV RPM, ELSE |
| ADDB | std | L02C8 | ;SAVE FOR NEXT PASS |
| ADDE | LADDE:subd | L8D9C | ;50 RPM |
| ADE1 | cpd | L02C8 | ;COMPARE PREV RPM |
| ADE5 | bls | LADED | ;BRANCH IF <= RPM, ELSE |

| | | | |
|------|-------------|--------------------|---|
| ADE7 | LAE7:bcclr | *L009A,#0x01 | ;CLEAR NEG DELTA RPM > 50 RPM |
| ADEA | bcclr | *L009A,#0x80 | ;CLEAR RPM INCREASING |
| ADED | LADED:ldaa | L8014 | ;OPTION WORD - TIS SET |
| ADF0 | beq | LADF8 | ;BRANCH IF Z, ELSE |
| ADF2 | brset | *L0096,#0x20,LAE56 | ;BRANCH IF PSPS INPUT HIGH (CRAMP) |
| ADF6 | bra | LAE5E | ; |
| | | | |
| ADF8 | LADF8:clra | | |
| ADF9 | brclr | *L0096,#0x02,LADFF | ;BRANCH IF FAN #1 OFF, ELSE |
| ADFD | oraa | #0x01 | ;SET BIT 0 |
| ADFF | LADFF:brclr | *L0096,#0x10,LAE05 | ;BRANCH IF FAN #2 OFF, ELSE |
| AE03 | oraa | #0x02 | ;SET BIT 1 |
| AE05 | LAE05:brclr | *L008E,#0x01,LAE0B | ;BRANCH IF NOT P/N MODE, ELSE |
| AE09 | oraa | #0x04 | ;SET BIT 2 |
| AE0B | LAE0B:brclr | *L008E,#0x80,LAE11 | ;BRANCH IF A/C REQUESTED, ELSE |
| AE0F | oraa | #0x08 | ;SET BIT 3 |
| AE11 | LAE11:brclr | *L0086,#0x20,LAE17 | ;BRANCH IF A/C CLUTCH ON, ELSE |
| AE15 | oraa | #0x10 | ;SET BIT 4 |
| AE17 | LAE17:brclr | *L00DB,#0x80,LAE1D | ;BRANCH IF A/C CLUTCH OFF, ELSE |
| AE1B | oraa | #0x20 | ;SET BIT 5 |
| AE1D | LAE1D:brclr | *L0090,#0x80,LAE23 | ;BRANCH IF BLM >= MIN WITH ACTIVE CCP, ELSE |
| AE21 | oraa | #0x80 | ;SET BIT 7 |
| AE23 | LAE23:cmpa | L02E9 | ;OLD STATUS WORD |
| AE26 | beq | LAE2B | ;BRANCH IF NO CHANGE, ELSE |
| AE28 | clr | L02EA | ;CLEAR |
| AE2B | LAE2B:staa | L02E9 | ;STORE NEW IAC STATUS WORD |
| AE2E | ldaa | L02EA | ;LOAD TIMER |
| AE31 | cmpa | L8DB4 | ;1 |
| AE34 | bcc | LAE3C | ;BRANCH IF > 1, ELSE |
| AE36 | inca | | ;INCREMENT TIMER |
| AE37 | staa | L02EA | ;STORE TIMER |
| AE3A | bra | LAE4B | |
| | | | |
| AE3C | LAE3C:brclr | *L00DB,#0x80,LAE4B | ;BRANCH IF A/C CLUTCH OFF, ELSE |
| AE40 | ldaa | *L00AD | ;LOAD RPM/12.5 |
| AE42 | adda | L8DB3 | ;12.5 RPM |
| AE45 | bcs | LAE4B | ;BRANCH IF OVERFLOWED, ELSE |
| AE47 | cmpa | *L00DE | ;COMPARE DESIRED IDLE |
| AE49 | bcs | LAE4E | ;BRANCH IF < DES IDLE, ELSE |
| AE4B | LAE4B:clra | | ;CLEAR A |
| AE4C | bra | LAE5B | ;GO STORE L02EB |
| | | | |
| AE4E | LAE4E:cmpa | L02ED | ;OLD D-RPM |
| AE51 | bcc | LAE5B | ;BRANCH IF > OLD D-RPM, ELSE |
| AE53 | staa | L02ED | ;STORE NEW D-RPM |
| | | | |
| AE56 | LAE56:ldd | L8CA3 | ;240 |
| AE59 | bra | LAE66 | ;GO STORE L02EB - TIMER |
| | | | |
| AE5B | LAE5B:staa | L02ED | ;STORE NEW D-RPM |
| AE5E | LAE5E:ldd | L02EB | ;TIMER |
| AE61 | beq | LAE66 | ;BRANCH IF Z, ELSE |
| AE63 | subd | #0x0001 | ;DECREMENT BY 1 |
| AE66 | LAE66:std | L02EB | ;STORE TIMER |
| AE69 | ldaa | *L00A7 | ;FILTERED CTS |
| AE6B | ldx | #0x8CA8 | ; |
| AE6E | brset | *L009A,#0x01,LAE75 | ;BRANCH IF NEG DELTA RPM > 50 RPM, ELSE |
| AE72 | ldx | #0x8CB9 | ; |
| AE75 | LAE75:jsr | L99D7 | ;2D LOOKUP |
| AE78 | ldab | L02AF | ; |
| AE7B | bne | LAE88 | ;BRANCH IF NZ, ELSE |
| AE7D | brset | *L0086,#0x20,LAE8E | ;BRANCH IF A/C CLUTCH OFF, ELSE |
| AE81 | brclr | *L0099,#0x08,LAE88 | ; |
| AE85 | jmp | LAF12 | ; |

| | | | |
|------|------------|--------------------|------------------------------------|
| AE88 | LAE88:psha | | ;SAVE TABLE LOOKUP RESULT ON STACK |
| AE89 | ldaa | L02B5 | ;PREV TIMER |
| AE8C | cmpa | L8CCA | ; |
| AE8F | bhi | LAE96 | ;BRANCH IF > CAL, ELSE |
| AE91 | ldaa | L8CCB | ;LOAD |
| AE94 | bra | LAEC3 | |
| | | | |
| AE96 | LAE96:ldaa | *L00F0 | ;IATMAT |
| AE98 | ldab | L8CCD | ;70.25 DEG C |
| AE9B | brset | *L0098,#0x01,LAEA2 | ; |
| AE9F | ldab | L8CCE | ;62.75 DEG C |
| AEA2 | LAEA2:cmpb | *L00A6 | ;LOAD CLNT TEMP (DEFAULTED) |
| AEA4 | bhi | LAEAB | ;BRANCH IF < CAL, ELSE |
| AEA6 | bclr | *L0098,#0x01 | ;CLEAR BIT 0 |
| AEA9 | bra | LAEB3 | ; |
| | | | |
| AEAB | LAEAB:bset | *L0098,#0x01 | ; |
| AEAE | adda | L8CCC | ;ADD |
| AEB1 | bcs | LAEBB | ;BRANCH IF OVERFLOW, ELSE |
| AEB3 | LAEB3:suba | #0x50 | ;SUBTRACT 80d |
| AEB5 | bcc | LAEB8 | ;BRANCH IF NO UNDERFLOW, ELSE |
| AEB7 | clra | | ;CLEAR A |
| AEB8 | LAEB8:lsla | | ;RESCALE |
| AEB9 | bcc | LAEBD | ;BRANCH IF NO OVERFLOW, ELSE |
| AEBB | LAEBB:ldaa | #0xFF | ;LOAD 255 |
| AEBD | LAEBD:ldx | #0x8CD1 | ; |
| AEC0 | jsr | L99D7 | ;2D LOOKUP |
| AEC3 | LAEC3:psha | | ;SAVE TABLE LOOKUP RESULT ON STACK |
| AEC4 | ldaa | L0303 | ; |
| AEC7 | ldab | L0304 | ; |
| AECA | cmpb | L8CCF | ;160d |
| AECD | beq | LAED5 | ;BRANCH IF = CAL, ELSE |
| AECF | incb | | ;INCREMENT |
| AED0 | stab | L0304 | ;SAVE |
| AED3 | bra | LAEDE | ; |
| | | | |
| AED5 | LAED5:clr | L0304 | ;CLEAR |
| AED8 | adda | L8CD0 | ;ADD 2 |
| AEDB | staa | L0303 | ;SAVE |
| AEDE | LAEDE:pula | | ;GET TABLE LOOKUP RESULT |
| AEDF | cmpa | L0303 | ;COMPARE PREV LOOKUP RESULT |
| AEE2 | bis | LAEE9 | ;BRANCH IF <= PREV, ELSE |
| AEE4 | ldaa | L0303 | ;LOAD |
| AEE7 | bra | LAF0E | ; |
| | | | |
| AEE9 | LAEE9:staa | L0303 | ;SAVE TABLE LOOKUP RESULT |
| AEEC | bra | LAF0E | |
| | | | |
| AEEE | LAEEE:psha | | ;SAVE TABLE LOOKUP RESULT |
| AEEF | ldaa | L0303 | ; |
| AEF2 | beq | LAF0E | ;BRANCH IF Z, ELSE |
| AEF4 | ldab | L0304 | ;LOAD |
| AEF7 | cmpb | L8CCF | ;160d |
| AEFA | beq | LAF02 | ;BRANCH IF = CAL, ELSE |
| AEFC | incb | | ;INCREMENT |
| AEFD | stab | L0304 | ;SAVE |
| AF00 | bra | LAF0E | ; |
| | | | |
| AF02 | LAF02:clr | L0304 | ;CLEAR |
| AF05 | suba | L8CD0 | ;SUBTRACT |
| AF08 | bpl | LAF0B | ;BRANCH IF PL, ELSE |
| AF0A | clra | | ;CLEAR A |
| AF0B | LAF0B:staa | L0303 | ;SAVE |

| | | | |
|------|-------------|--------------------|---|
| AF0E | LAF0E:pulb | | ;GET TABLE LOOKUP RESULT |
| AF0F | aba | | ;ADD TO L0303 |
| AF10 | bcs | LAF54 | ;BRANCH IF OVERFLOW, ELSE |
| AF12 | LAF12:brset | *L00D9,#0x20,LAF35 | ;BRANCH IF NOT TO ADD HOT START IDLE SPEED |
| | | | ; OFFSET, ELSE |
| AF16 | ldab | *L00A6 | ;LOAD CLNT TEMP (DEFAULTED) |
| AF18 | cmpb | L8C9E | ;56 DEG C |
| AF1B | bls | LAF2B | ;BRANCH IF <= CAL, ELSE |
| AF1D | ldab | *L00BD | ;FILTERED MPH |
| AF1F | cmpb | L8C9F | ;5 MPH |
| AF22 | bcc | LAF2B | ;BRANCH IF >= CAL, ELSE |
| AF24 | ldx | *L00ED | ;1 SECOND TIMER |
| AF26 | cpx | L8CA0 | ;10 SECONDS |
| AF29 | bcs | LAF30 | ;BRANCH IF < 10 SECONDS, ELSE |
| AF2B | LAF2B:bset | *L00D9,#0x20 | ;SET DON'T ADD HOT STRT IDLE SPD OFFSET |
| AF2E | bra | LAF35 | ; |
| AF30 | LAF30:adda | L8C9D | ;ADD HOT START IDLE SPEED OFFSET |
| AF33 | bcs | LAF54 | ;BRANCH IF OVERFLOW, ELSE |
| AF35 | LAF35:ldab | *L005A | ;FILTERED ENGINE TORQUE |
| AF37 | cmpb | L8CA6 | ;00 |
| AF3A | bhi | LAF41 | ;BRANCH IF > CAL, ELSE |
| AF3C | adda | L8CA7 | ;ADD 00 |
| AF3F | bcs | LAF54 | ;BRANCH IF OVERFLOW, ELSE |
| AF41 | LAF41:ldx | L02EB | ;LOAD TIMER |
| AF44 | beq | LAF4B | ;BRANCH IF Z, ELSE |
| AF46 | adda | L8CA2 | ;ADD 25 RPM |
| AF49 | bcs | LAF54 | ;BRANCH IF OVERFLOW, ELSE |
| AF4B | LAF4B:brclr | *L0090,#0x80,LAF56 | ;BRANCH IF BLM >= MIN WITH ACTIVE CCP, ELSE |
| AF4F | adda | L8C9C | ;ADD 100 RPM |
| AF52 | bcc | LAF56 | ;BRANCH IF NO OVERFLOW, ELSE |
| AF54 | LAF54:ldaa | #0xFF | ;LOAD 255 |
| AF56 | LAF56:psha | | ;SAVE DESIRED IDLE RPM ON STACK |
| AF57 | ldab | #0x01 | ;LOAD 1 |
| AF59 | suba | *L00DE | ;SUBTRACT DESIRED IDLE RPM |
| AF5B | bcc | LAF6F | ;BRANCH IF > PREV DESIRED IDLE, ELSE |
| AF5D | inc | L0189 | ;INCREMENT |
| AF60 | ldab | L0189 | ;LOAD |
| AF63 | cmpb | L8CA5 | ;8 |
| AF66 | bhi | LAF6C | ;BRANCH IF > CAL, ELSE |
| AF68 | clrb | | ;CLEAR B |
| AF69 | pula | | ;RESTORE DESIRED IDLE RPM |
| AF6A | bra | LAF77 | ;GO ADD B TO DESIRED IDLE |
| AF6C | LAF6C:ldab | #0xFF | ;LOAD 255 |
| AF6E | nega | | ;INVERT DELTA BETWEEN CURRENT AND PREV |
| AF6F | LAF6F:clr | L0189 | ;CLEAR L0189 |
| AF72 | cmpa | #0x01 | ;1 ? |
| AF74 | pula | | ;GET BACK DESIRED IDLE RPM |
| AF75 | bls | LAF7A | ;BRANCH IF DELTA RPM <= 1, ELSE |
| AF77 | LAF77:ldaa | *L00DE | ;DESIRED IDLE RPM |
| AF79 | aba | | ;ADD B TO DESIRED IDLE RPM |
| AF7A | LAF7A:brclr | *L0095,#0x82,LAF8A | ;BRANCH IF HUD DEVICE NOT CONNECTED, ELSE |
| AF7E | ldab | L8021 | ;OPTION WORD - TIS SET |
| AF81 | bne | LAF8A | ;BRANCH IF NOT Z, ELSE |
| AF83 | brclr | *L009F,#0x40,LAF8A | ; |
| AF87 | jsr | L5003 | ;UPDATE HUD |
| AF8A | LAF8A:brclr | *L0084,#0x80,LAF9A | ;BRANCH IF NOT MODE 4, ELSE |
| AF8E | ldab | L0234 | ;LOAD CONTROL BYTE |
| AF91 | rorb | | ;SHIFT RIGHT |
| AF92 | bcc | LAF9A | ;BRANCH IF NO OVERFLOW (B0 CLEAR), ELSE |
| AF94 | rorb | | ;SHIFT RIGHT AGAIN |
| AF95 | bcc | LAF9A | ;BRANCH IF NO OVERFLOW (B1 CLEAR), ELSE |
| AF97 | ldaa | L0235 | ;LOAD COMMANDED IDLE |

```

AF9A    LAF9A:staa    *L00DE                ;SAVE DESIRED IDLE
AF9C          brset  *L0090,#0x10,LAFE6      ;
AFA0          ldx    *L00ED                ;1 SECOND TIMER
AFA2          cpx    L8EC7                  ;0 SECONDS
AFA5          beq    LAFB6                  ;BRANCH IF 0 SECONDS, ELSE
AFA7          ldab   *L00DE                ;DESIRED IDLE
AFA9          subb   *L00AD                ;SUBTRACT RPM/12.5
AFAB          bcs    LAFE6                  ;BRANCH IF OVER IDLE, ELSE
AFAD          ldx    L02FE                  ;
AFB0          abx                    ;ADD DELTA RPM/12.5
AFB1          stx    L02FE                  ;SAVE
AFB4          bra    LAFE6                  ;

AFB6    LAFB6:ldd    L02FE                  ;
AFB9          idiv                    ;D/X
AFBA          stx    L02FE                  ;STORE QUOTIENT
AFBD          ldaa   *L00A6                ;LOAD CLNT TEMP (DEFAULTED)
AFBF          cmpa   L8EC9                ;0
AFC2          bcc    LAFE3                ;BRANCH IF > CAL, ELSE
AFC4          ldd    L02FE                ;LOAD
AFC7          lsr    ;DIV BY 4
AFC8          lsr    ;
AFC9          tsta                    ;TEST MSB
AFCA          beq    LAFCE                ;BRANCH IF Z, ELSE
AFCC          ldab   #0xFF                ;LOAD 255 (LSB)
AFCE    LAFCE:tba                    ;TRANSFER TO A REG
AFCF          ldx    #0x8ECB                ;(ALL 00s)
AFD2          jsr    L99D7                ;2D LOOKUP
AFD5          adda   *L0076                ;THIS NOT USED
AFD7          bcc    LAFDB                ;BRANCH IF NO OVERFLOW, ELSE
AFD9          ldaa   #0xFF                ;LOAD 255
AFDB    LAFDB:suba   L8ECA                ;SUBTRACT 00
AFDE          bcc    LAFE1                ;BRANCH IF NO UNDERFLOW, ELSE
AFE0          clra                    ;CLEAR
AFE1    LAFE1:staa   *L0076                ;THIS NOT USED
AFE3    LAFE3:bset   *L0090,#0x10          ;SET BIT 4
AFE6    LAFE6:brset  *L0018,#0x20,LAFF7    ;BRANCH IF MALF FLAG SET (NOT USED), ELSE

;-----
; START OF A/C CODE
;-----

AFEA          ldd    #0x8018                ;BIT 7 AND 0x18
AFED          jsr    LD933                ;GO CHECK MODE 4 CONTROL WORDS
AFF0          beq    LAFFA                ;BRANCH IF Z, ELSE
AFF2          bpl    LAFF7                ;BRANCH IF BIT 7 CLEAR, ELSE
AFF4          jmp    LB282                ;GO TURN ON A/C CLUTCH

AFF7    LAFF7:jmp    LB2C2                ;GO TURN OFF A/C CLUTCH

AFFA    LAFFA:ldaa   L8014                ;OPTION WORD - TIS SET
AFFD          beq    LB015                ;BRANCH IF Z, ELSE
AFFF          bpl    LB015                ;BRANCH IF BIT 7 CLEAR, ELSE
B001          brclr  *L0096,#0x20,LB012    ;BRANCH IF PSPS INPUT LOW (NOT CRAMP)
B005          brset  *L0094,#0x40,LAFF7    ;BRANCH IF ? (GO TURN OFF A/C CLUTCH), ELSE
B009          brset  *L0086,#0x20,LB012    ;BRANCH IF A/C CLUTCH OFF, ELSE
B00D          bset   *L0094,#0x40          ;SET BIT 6
B010          bra    LAFF7                ;GO TURN OFF A/C CLUTCH

B012    LB012:bclr   *L0094,#0x40          ;CLEAR BIT 6
B015    LB015:ldaa   L02B2                ;TIMER
B018          bne    LB037                ;BRANCH IF NZ, ELSE
B01A          brset  *L0086,#0x80,LB03D    ;BRANCH IF ENGINE RUNNING, ELSE
B01E          brset  *L0094,#0x10,LB03A    ;BRANCH IF A/C WAS ON THIS CRANK, ELSE
B022          brclr  *L00D9,#0x01,LB03A    ;BRANCH IF A/C SLUGGING TEMP NOT MET, ELSE

```

| | | | |
|------|-------------|--------------------|---|
| B026 | ldd | *L00EB | ;RPM |
| B028 | cpd | L8D6E | ;300 |
| B02C | bls | LB03A | ;BRANCH IF <= 300 RPM, ELSE |
| B02E | bset | *L0094,#0x10 | ;SET A/C WAS ON THIS CRANK |
| B031 | ldaa | L8D70 | ;10 |
| B034 | staa | L02B2 | ;TIMER |
| B037 | LB037: jmp | LB282 | ;GO TURN ON A/C CLUTCH |
| B03A | LB03A: jmp | LB29F | ; |
| B03D | LB03D: ldaa | *L00BD | ;FILTERED MPH |
| B03F | cmpa | L8D59 | ; |
| B042 | bcs | LB047 | ;BRANCH IF < CAL, ELSE |
| B044 | clr | L02B1 | ;CLEAR TIMER |
| B047 | LB047: ldaa | L0322 | ;TABLE LOOKUP RESULT (WILL ALWAYS BE Z) |
| B04A | ldab | *L0065 | ;FILTERED LOW AIRFLOW |
| B04C | subb | *L0067 | ;SUB FILTERED LOW AIRFLOW |
| B04E | bcc | LB051 | ;BRANCH IF NO UNDERFLOW, ELSE |
| B050 | clrb | | ;CLEAR B |
| B051 | LB051: mul | | ;A * B |
| B052 | lsl | | ;MUL BY 2 |
| B053 | cmpa | L8D51 | ;MIN IAC A/C STEPS |
| B056 | bcc | LB05B | ;BRANCH IF MSB > MIN, ELSE |
| B058 | ldaa | L8D51 | ;LOAD MIN STEPS |
| B05B | LB05B: ldab | L02B1 | ;TIMER |
| B05E | psha | | ;SAVE STEPS |
| B05F | ldaa | L8D58 | ;MULTIPLIER |
| B062 | mul | | ;A * B |
| B063 | pula | | ;RESTORE STEPS |
| B064 | aba | | ;ADD TO A |
| B065 | brclr | *L00D9,#0x01,LB070 | ;BRANCH IF A/C SLUGGING TEMP NOT MET, ELSE |
| B069 | brset | *L0096,#0x80,LB070 | ;BRANCH IF NORMAL A/C REQ HAS TURNED A/C ON |
| B06D | adda | L8D71 | ; AT STARTUP, ELSE |
| B070 | LB070: cmpa | L8D50 | ;ADD IAC STEPS FOR A/C SLUGGING |
| B073 | bls | LB078 | ;MAX IAC A/C STEPS |
| B075 | ldaa | L8D50 | ;BRANCH IF <= MAX, ELSE |
| B078 | LB078: ldab | L8009 | ;LOAD MAX STEPS |
| B07B | beq | LB08C | ;OPTION WORD - A/C PRESSURE SENSOR PRESENT |
| B07D | ldaa | *L00F9 | ;BRANCH IF Z, ELSE (TIS SET) |
| B07F | ldx | #0x8D78 | ;A/C PRESSURE |
| B082 | brclr | *L008E,#0x01,LB089 | ;ANTICIPATED IAC STEPS VS A/C PRESS IN GEAR |
| B086 | ldx | #0x8D89 | ;BRANCH IF NOT P/N MODE, ELSE |
| B089 | LB089: jsr | L99D7 | ;ANTICIPATED IAC STEPS VS A/C PRESS, P/N |
| B08C | LB08C: staa | L02AC | ;2D LOOKUP |
| B08F | ldab | *L00AA | ;STORE ANTICIPATED A/C LOAD STEPS |
| B091 | cmpb | L8CE2 | ;%TPS |
| B094 | bcc | LB0A1 | ;1.2 %TPS |
| B096 | brclr | *L0086,#0x20,LB0A1 | ;BRANCH IF >= CAL, ELSE |
| B09A | ldaa | *L00AD | ;BRANCH IF A/C CLUTCH ON, ELSE |
| B09C | cmpa | L8D2E | ;LOAD RPM/12.5 |
| B09F | bcc | LB0A6 | ;3187.5 RPM |
| B0A1 | LB0A1: clr | L030E | ;BRANCH IF > RPM THRESHOLD, ELSE |
| B0A4 | bra | LB0CC | ;CLEAR SPARK TERM |
| B0A6 | LB0A6: ldab | *L00ED | ;GO CLEAR BITS 3 AND 4 L00DA |
| B0A8 | bne | LB0CC | ;1 SECOND TIMER MSB |
| B0AA | ldx | *L00EE | ;BRANCH IF NOT Z, ELSE |
| B0AC | cpx | L8D2C | ;1 SECOND TIMER LSB |
| B0AF | bcc | LB0CC | ;0 SECONDS |
| B0B1 | suba | *L00DE | ;BRANCH IF >= CAL, ELSE |
| B0B3 | bcs | LB0CC | ;DESIRED IDLE RPM |
| B0B5 | cmpa | L8D2F | ;BRANCH IF UNDER-IDLE, ELSE |
| B0B8 | bls | LB0BF | ;3187.5 RPM |
| | | | ;BRANCH IF <= CAL, ELSE |

| | | | |
|------|------------|--------------------|---------------------------------------|
| B0BA | bset | *L00DA,#0x10 | ;SET BIT 4 |
| B0BD | bra | LB0C2 | ; |
| B0BF | LB0BF:bclr | *L00DA,#0x10 | ;CLEAR BIT 4 |
| B0C2 | LB0C2:cmpa | L8D30 | ;3187.5 RPM |
| B0C5 | bls | LB0CC | ;BRANCH IF <= CAL, ELSE |
| B0C7 | bset | *L00DA,#0x08 | ;SET BIT 3 |
| B0CA | bra | LB0CF | |
| B0CC | LB0CC:bclr | *L00DA,#0x18 | ;CLEAR BITS 3 AND 4 |
| B0CF | LB0CF:ldaa | *L00ED | ;1 SECOND TIMER MSB |
| B0D1 | bne | LB10B | ;BRANCH IF NOT Z, ELSE |
| B0D3 | ldx | *L00EE | ;1 SECOND TIMER LSB |
| B0D5 | cpx | L8D27 | ;0 SECONDS |
| B0D8 | bcc | LB10B | ;BRANCH IF >= 0 SECONDS, ELSE |
| B0DA | ldaa | *L00A6 | ;LOAD CLNT TEMP (DEFAULTED) |
| B0DC | cmpa | L8D29 | ;151 DEG C |
| B0DF | bls | LB10B | ;BRANCH IF <= COOLANT THRESHOLD, ELSE |
| B0E1 | ldaa | *L00AC | ;LOAD RPM/25 |
| B0E3 | cmpa | L8D2A | ;6375 RPM |
| B0E6 | bls | LB10B | ;BRANCH IF <= RPM THRESHOLD, ELSE |
| B0E8 | ldaa | L02FA | ;TIMER |
| B0EB | cmpa | L8D2B | ;TIME LIMIT (80d) |
| B0EE | bhi | LB10B | ;BRANCH IF > CAL, ELSE |
| B0F0 | ldx | *L00EB | ;RPM |
| B0F2 | cpx | L025F | ;PREV RPM |
| B0F5 | bls | LB0FC | ;BRANCH IF RPM INCREASING, ELSE |
| B0F7 | clr | L02FA | ;CLEAR TIMER |
| B0FA | bra | LB0FF | |
| B0FC | LB0FC:inc | L02FA | ;INCREMENT TIMER |
| B0FF | LB0FF:bset | *L0099,#0x08 | ;SET BIT 3 |
| B102 | ldaa | L8D40 | ;20 |
| B105 | staa | L02AD | ; |
| B108 | jmp | LB25B | ; |
| B10B | LB10B:bclr | *L0099,#0x08 | ;CLEAR BIT 3 |
| B10E | ldx | *L00ED | ;1 SECOND TIMER |
| B110 | cpx | L8D31 | ;3 SECONDS |
| B113 | bcc | LB118 | ;BRANCH IF >= 3 SECONDS, ELSE |
| B115 | jmp | LB29F | ;JUMP |
| B118 | LB118:ldaa | L8D3F | ;124 DEG C |
| B11B | brset | *L0086,#0x20,LB122 | ;BRANCH IF A/C CLUTCH OFF, ELSE |
| B11F | ldaa | L8D3E | ;124 DEG C |
| B122 | LB122:cmpa | *L00A6 | ;LOAD CLNT TEMP (DEFAULTED) |
| B124 | bcs | LB199 | ;BRANCH IF CTS > CAL, ELSE |
| B126 | ldx | #0x0000 | ;LOAD 0 |
| B129 | ldaa | *L00BD | ;FILTERED MPH |
| B12B | beq | LB130 | ;BRANCH IF Z, ELSE |
| B12D | ldx | L02F0 | ;TIMER |
| B130 | LB130:ldaa | L8D3A | ;96 %TPS |
| B133 | brclr | *L0086,#0x20,LB13A | ;BRANCH IF A/C CLUTCH ON, ELSE |
| B137 | ldaa | L8D3B | ;79.7 %TPS |
| B13A | LB13A:cmpa | *L00AA | ;%TPS |
| B13C | bcc | LB143 | ;BRANCH IF TPS <= CAL, ELSE |
| B13E | inx | | ;INCREMENT X |
| B13F | beq | LB151 | ;BRANCH IF Z, ELSE |
| B141 | bra | LB14E | |
| B143 | LB143:cpx | #0x0000 | ;Z ? |
| B146 | beq | LB161 | ;BRANCH IF Z, ELSE |
| B148 | dex | | ;DECREMENT |
| B149 | stx | L02F0 | ;SAVE TIMER |

| | | | |
|------|-------------|--------------------|--|
| B14C | bra | LB161 | |
| B14E | LB14E:stx | L02F0 | ;SAVE TIMER |
| B151 | LB151:ldx | L02F0 | ;LOAD TIMER |
| B154 | cpx | L8D3C | ;1600 |
| B157 | bhi | LB161 | ;BRANCH IF > CAL, ELSE |
| B159 | brset | *L0086,#0x02,LB161 | ;BRANCH IF HOT OPEN LOOP, ELSE |
| B15D | brclr | *L0099,#0x01,LB199 | ; |
| B161 | LB161:brset | *L008E,#0x80,LB184 | ;BRANCH IF A/C NOT REQUESTED, ELSE |
| B165 | ldaa | L02B3 | ; |
| B168 | inca | | ;INCREMENT |
| B169 | beq | LB16E | ;BRANCH IF Z, ELSE |
| B16B | staa | L02B3 | ;SAVE |
| B16E | LB16E:ldab | L8D52 | ;12 MPH |
| B171 | cmpb | *L00BD | ;FILTERED MPH |
| B173 | bhi | LB1BC | ;BRANCH IF < CAL, ELSE |
| B175 | cmpa | #0x00 | ; |
| B177 | bne | LB1BC | ;BRANCH IF NOT Z, ELSE |
| B179 | ldx | #0x0000 | ; |
| B17C | stx | L01AC | ; |
| B17F | bclr | *L00DA,#0x01 | ; |
| B182 | bra | LB1BC | |
| B184 | LB184:clr | L02B3 | ; |
| B187 | brset | *L00DA,#0x01,LB199 | ; |
| B18B | brset | *L008E,#0x01,LB196 | ;BRANCH IF P/N MODE, ELSE |
| B18F | ldab | *L00F0 | ;IATMAT |
| B191 | cmpb | L8D54 | ;00 |
| B194 | bis | LB19C | ;BRANCH IF <= 0, ELSE |
| B196 | LB196:bset | *L00DA,#0x01 | |
| B199 | LB199:jmp | LB29F | |
| B19C | LB19C:ldaa | L8D52 | ;12 MPH |
| B19F | ldx | L01AC | ; |
| B1A2 | bne | LB1A7 | ;BRANCH IF NOT Z, ELSE |
| B1A4 | ldaa | L8D53 | ;8 MPH |
| B1A7 | LB1A7:cmpa | *L00BD | ;FILTERED MPH |
| B1A9 | bis | LB199 | ;BRANCH IF MPH >= CAL, ELSE |
| B1AB | ldx | L0292 | ;FAN #1 TURN ON DELAY TIMER |
| B1AE | beq | LB199 | ;BRANCH IF Z, ELSE |
| B1B0 | ldx | L01AC | |
| B1B3 | cpx | L8D55 | ;160 |
| B1B6 | bhi | LB199 | ;BRANCH IF > 160, ELSE |
| B1B8 | inx | | |
| B1B9 | stx | L01AC | |
| B1BC | LB1BC:ldaa | L8009 | ;OPTION WORD - A/C PRESSURE SENSOR PRESENT |
| B1BF | beq | LB1D3 | ;BRANCH IF Z, ELSE (TIS SET) |
| B1C1 | ldaa | *L00F9 | ;A/C PRESSURE |
| B1C3 | ldx | #0x8D74 | ;INDEX |
| B1C6 | brset | *L0086,#0x20,LB1CB | ;BRANCH IF A/C CLUTCH OFF, ELSE |
| B1CA | inx | | ;8D75 |
| B1CB | LB1CB:cmpa | 0x00,x | ;8D74 OR |
| B1CD | bhi | LB199 | ;8D75 |
| B1CF | cmpa | 0x02,x | ;BRANCH IF > CAL, ELSE |
| B1D1 | bcs | LB199 | ;8D76 OR |
| B1D3 | LB1D3:brset | *L008E,#0x80,LB1DD | ;8D77 |
| B1D7 | ldd | L86B4 | ;BRANCH IF < CAL, ELSE |
| B1DA | std | L0292 | ;BRANCH IF A/C NOT REQUESTED, ELSE |
| B1DD | LB1DD:ldaa | *L00DE | ;00 |
| B1DF | suba | *L00AD | ;FAN #1 DELAY TIMER FOR A/C |
| B1E1 | bcs | LB20F | ;DESIRED IDLE RPM |
| B1E3 | psha | | ;SUBTRACT RPM/12.5 |
| | | | ;BRANCH IF RPM > DESIRED, ELSE |
| | | | ;SAVE DELTA RPM |

| | | | |
|------|-------------|--------------------|---|
| B1E4 | ldaa | *L00DE | ;DESIRED IDLE RPM |
| B1E6 | lsla | | ;RESCALE |
| B1E7 | bcc | LB1EB | ;BRANCH IF NO OVERFLOW, ELSE |
| B1E9 | ldaa | #0xFF | ;LOAD 255 |
| B1EB | LB1EB:ldx | #0x8D5A | ; |
| B1EE | jsr | L99D7 | ;2D LOOKUP |
| B1F1 | pulb | | ;GET BACK DELTA RPM |
| B1F2 | cba | | ;COMPARE |
| B1F3 | bhi | LB20F | ;BRANCH IF DELTA RPM > LOOKUP RESULT, ELSE |
| B1F5 | clr | L02AB | ;CLEAR |
| B1F8 | ldaa | L8D57 | ;16 |
| B1FB | staa | L02AF | ;STORE |
| B1FE | brset | *L0086,#0x20,LB20C | ;BRANCH IF A/C CLUTCH OFF, ELSE |
| B202 | ldaa | L02B0 | ;A/C CLUTCH ON TIMER |
| B205 | cmpa | #0x50 | ;80 |
| B207 | bcc | LB20C | ;BRANCH IF >= 80, ELSE |
| B209 | inc | L02B1 | ;INCREMENT TIMER |
| B20C | LB20C: jmp | LB2C2 | ;GO TURN OFF A/C CLUTCH |
| | | | |
| B20F | LB20F:brset | *L0086,#0x20,LB216 | ;BRANCH IF A/C CLUTCH OFF, ELSE |
| B213 | bset | *L0096,#0x80 | ;SET NORMAL A/C REQ HAS TURNED A/C ON AT SU |
| B216 | LB216:ldaa | *L00AC | ;LOAD RPM/25 |
| B218 | cmpa | L8D33 | ;4700 RPM |
| B21B | bhi | LB20C | ;BRANCH IF > CAL, ELSE |
| B21D | cmpa | L8D34 | ;4000 RPM |
| B220 | bis | LB22E | ;BRANCH IF <= CAL, ELSE |
| B222 | brclr | *L0086,#0x20,LB22E | ;BRANCH IF A/C CLUTCH ON, ELSE |
| B226 | ldx | L0294 | ;A/C CLUTCH OFF TIMER |
| B229 | cpx | L8D35 | ;MIN A/C CLUTCH OFF TIME |
| B22C | bcs | LB20C | ;BRANCH IF < CAL, ELSE |
| B22E | LB22E:ldx | *L0032 | ;RUN TIME |
| B230 | cpx | L8D37 | ;60 SECONDS |
| B233 | bhi | LB24A | ;BRANCH IF > 60 SECONDS, ELSE |
| B235 | brset | *L00D9,#0x01,LB24A | ;BRANCH IF A/C SLUGGING TEMP MET, ELSE |
| B239 | ldaa | *L00AD | ;LOAD RPM/12.5 |
| B23B | suba | *L00DE | ;DESIRED IDLE RPM |
| B23D | bcs | LB24A | ;BRANCH IF RPM < DESIRED, ELSE |
| B23F | cmpa | L8D39 | ;150 RPM |
| B242 | bis | LB24A | ;BRANCH IF DELTA RPM <= 150 RPM |
| B244 | ldaa | L8D40 | ;TIME DELAY BEFORE TURNING ON A/C CLUTCH |
| B247 | staa | L02AD | ;STORE TIMER |
| B24A | LB24A:brclr | *L0086,#0x20,LB25B | ;BRANCH IF A/C CLUTCH ON, ELSE |
| B24E | brclr | *L0096,#0x08,LB25B | ;BRANCH IF TCC NOT LOCKED, ELSE |
| B252 | brset | *L009B,#0x01,LB25B | ;BRANCH IF DFCO ENABLED, ELSE |
| B256 | bset | *L00A3,#0x40 | ;SET TCC SLIP REQUESTED FOR A/C ENGAGE |
| B259 | bra | LB263 | ; |
| | | | |
| B25B | LB25B:ldaa | L02AF | ;TIMER |
| B25E | beq | LB266 | ;BRANCH IF Z, ELSE |
| B260 | dec | L02AF | ;DECREMENT TIMER |
| B263 | LB263: jmp | LB2FF | ; |
| | | | |
| B266 | LB266:ldaa | L02AD | ;TIMER |
| B269 | cmpa | L8D40 | ;20 |
| B26C | bcc | LB282 | ;BRANCH IF >=CAL (TURN ON A/C CLUTCH), ELSE |
| B26E | ldab | L02AB | ; |
| B271 | cmpb | L02AC | ;ANTICIPATED A/C LOAD STEPS |
| B274 | bcc | LB27C | ;BRANCH IF >= A/C LOAD STEPS, ELSE |
| B276 | inc | L02AB | ;INCREMENT |
| B279 | jmp | LB6F1 | ;GO INCREMENT IAC POSITION |
| | | | |
| B27C | LB27C:inc | L02AD | ;INCREMENT TIMER |
| B27F | jmp | LB73E | ;SKIP THE REST, GO CLEAR BIT 4 L009A |

```

;-----
; TURN A/C CLUTCH ON
;-----
B282 LB282:bclr *L0086,#0x20 ;TURN A/C CLUTCH ON
B285 ldaa L02B0 ;LOAD A/C CLUTCH ON TIMER
B288 inca ;INCREMENT
B289 beq LB28E ;BRANCH IF Z, ELSE
B28B staa L02B0 ;STORE ON TIMER
B28E LB28E:ldd #0x0000 ;
B291 std L0294 ;RESET A/C CLUTCH OFF TIMER
B294 ldd #0xFFFF ;MAX DUTY CYCLE
B297 std L3FD2 ;STORE A/C RELAY CONTROL OUTPUT
B29A clr L02AE ;
B29D bra LB2FF

B29F LB29F:ldaa L02AB ;LOAD
B2A2 bne LB2DC ;BRANCH IF NZ, ELSE
B2A4 ldaa L8D41 ;
B2A7 brset *L008E,#0x01,LB2B8 ;BRANCH IF P/N MODE, ELSE
B2AB ldaa L8D42 ;
B2AE ldab *L00BD ;FILTERED MPH
B2B0 cmpb L8D44 ;10 MPH
B2B3 bcs LB2B8 ;BRANCH IF < 10 MPH, ELSE
B2B5 ldaa L8D43 ;
B2B8 LB2B8:cmpa L02AE ;
B2BB bls LB2C2 ;BRANCH IF > CAL, ELSE
B2BD inc L02AE ;INCREMENT
B2C0 bra LB2FF

;-----
; TURN A/C CLUTCH OFF
;-----
B2C2 LB2C2:bset *L0086,#0x20 ;TURN A/C CLUTCH OFF
B2C5 ldx L0294 ;A/C CLUTCH OFF TIMER
B2C8 inx ;INCREMENT
B2C9 beq LB2CE ;BRANCH IF Z, ELSE
B2CB stx L0294 ;A/C CLUTCH OFF TIMER
B2CE LB2CE:clr L02B0 ;CLEAR A/C CLUTCH ON TIMER
B2D1 ldx #0xF000 ;0 DUTY CYCLE
B2D4 stx L3FD2 ;STORE A/C RELAY CONTROL OUTPUT
B2D7 clr L02AD ;
B2DA bra LB2FF

B2DC LB2DC:ldaa L8CE2 ;1.2 %TPS
B2DF adda L8D45 ;3.1 %TPS
B2E2 cmpa *L00AA ;%TPS
B2E4 bls LB2F6 ;BRANCH IF <= 4.3 %TPS, ELSE
B2E6 brset *L00DB,#0x08,LB2F6 ;
B2EA ldaa L02AC ;A/C LOAD STEPS
B2ED ldab L8D46 ;A/C LOAD STEPS MULTIPLIER
B2F0 mul ;A * B
B2F1 staa L02AB ;STORE
B2F4 beq LB2FF ;BRANCH IF Z, ELSE
B2F6 LB2F6:dec L02AB ;DECREMENT STEPS
B2F9 bset *L00DB,#0x08 ;SET BIT 3
B2FC jmp LB708 ;GO DECREMENT IAC POSITION

B2FF LB2FF:ldx *L00EB ;RPM
B301 stx L025F ;SAVE FOR NEXT PASS
B304 bclr *L00DB,#0x08 ;CLEAR BIT 3
B307 brset *L0086,#0x80,LB30E ;BRANCH IF ENGINE RUNNING, ELSE
B30B jmp LB73E ;SKIP COMMANDED A/F CALCULATION

```

| | | | |
|------|-------------|--------------------|---|
| B30E | LB30E:brclr | *L0096,#0x02,LB32C | ;BRANCH IF FAN #1 OFF, ELSE |
| B312 | brclr | *L00DB,#0x80,LB324 | ;BRANCH IF A/C CLUTCH OFF, ELSE |
| B316 | ldaa | L02B6 | ;?FAN #1 |
| B319 | cmpa | L8DD0 | ;8 |
| B31C | bcc | LB32F | ;BRANCH IF >= 8, ELSE |
| B31E | inc | L02B6 | ;INCREMENT STEPS |
| B321 | jmp | LB6F1 | ;GO INCREMENT IAC POSITION |
| B324 | LB324:ldaa | L8DD0 | ;8 |
| B327 | staa | L02B6 | ;STORE |
| B32A | bra | LB32F | ; |
| B32C | LB32C:clr | L02B6 | ;CLEAR |
| B32F | LB32F:brclr | *L0096,#0x10,LB34D | ;BRANCH IF FAN #2 OFF, ELSE |
| B333 | brclr | *L00DB,#0x80,LB345 | ;BRANCH IF A/C CLUTCH OFF, ELSE |
| B337 | ldaa | L02B7 | ;? FAN #2 |
| B33A | cmpa | L8DD1 | ;8 |
| B33D | bcc | LB350 | ;BRANCH IF >= 8, ELSE |
| B33F | inc | L02B7 | ;INCREMENT |
| B342 | jmp | LB6F1 | ;GO INCREMENT IAC POSITION |
| B345 | LB345:ldaa | L8DD1 | ;8 |
| B348 | staa | L02B7 | ;STORE |
| B34B | bra | LB350 | ; |
| B34D | LB34D:clr | L02B7 | ;CLEAR |
| B350 | LB350:ldd | L02EB | ;LOAD TIMER |
| B353 | beq | LB371 | ;BRANCH IF Z, ELSE |
| B355 | ldx | #0x8DC2 | ; |
| B358 | brset | *L0094,#0x40,LB35F | ;BRANCH IF BIT 6, ELSE |
| B35C | ldx | #0x8DB7 | ; |
| B35F | LB35F:ldaa | *L005A | ;FILTERED ENGINE TORQUE |
| B361 | ldab | #0x60 | ;OFFSET |
| B363 | jsr | L99D3 | ;2D LOOKUP |
| B366 | cmpa | L02F7 | ;TIME DELAY BEFORE MOVING IAC MOTOR |
| B369 | bls | LB374 | ;BRANCH IF <= L02F7, ELSE |
| B36B | inc | L02F7 | ;INCREMENT |
| B36E | jmp | LB6F1 | ;GO INCREMENT IAC POSITION |
| B371 | LB371:clr | L02F7 | ;CLEAR TIME DELAY |
| B374 | LB374:brclr | *L008E,#0x01,LB382 | ;BRANCH IF NOT P/N MODE, ELSE |
| B378 | ldaa | L02C4 | ;TIMER |
| B37B | cmpa | L8DA1 | ;20 |
| B37E | bcc | LB39F | ;BRANCH IF >= CAL, ELSE |
| B380 | bra | LB3A2 | ; |
| B382 | LB382:ldaa | L02C3 | ;TIMER |
| B385 | cmpa | L8DA0 | ;8 |
| B388 | bls | LB3A2 | ;BRANCH IF <= CAL, ELSE |
| B38A | ldab | L8D9F | ;15 |
| B38D | brclr | *L0086,#0x20,LB394 | ;BRANCH IF A/C CLUTCH OFF, ELSE |
| B391 | ldab | L8D9E | ;10 |
| B394 | LB394:cmpb | L02C5 | ; |
| B397 | bls | LB3A2 | ;BRANCH IF >= CAL, ELSE |
| B399 | inc | L02C5 | ;INCREMENT |
| B39C | jmp | LB6F1 | ;GO INCREMENT IAC POSITION |
| B39F | LB39F:clr | L02C5 | ;CLEAR TIMER |
| B3A2 | LB3A2:ldaa | L800F | ;OPTION WORD - TIS SET |
| B3A5 | beq | LB3EC | ;BRANCH IF Z, ELSE |
| B3A7 | brset | *L008F,#0x04,LB3EC | ;BRANCH IF INVALID PRNDL SW INPUT, ELSE |
| B3AB | ldaa | L02CA | ;COUNTER |
| B3AE | brclr | *L00A1,#0x40,LB3BA | ;BRANCH IF PRNDL -> REVERSE, ELSE |
| B3B2 | inca | | ;INCREMENT COUNTER |

| | | | |
|------|-------------|--------------------|--------------------------------------|
| B3B3 | beq | LB3EC | ;BRANCH IF Z, ELSE |
| B3B5 | staa | L02CA | ;SAVE COUNTER |
| B3B8 | bra | LB3EC | ; |
| B3BA | LB3BA:tsta | | ;TEST COUNTER |
| B3BB | beq | LB3C1 | ;BRANCH IF Z, ELSE |
| B3BD | deca | | ;DECREMENT COUNTER |
| B3BE | staa | L02CA | ;SAVE COUNTER |
| B3C1 | LB3C1:brclr | *L00A1,#0x1E,LB3EC | ;BRANCH IF PRNDL NOT DRV 1-4, ELSE |
| B3C5 | ldaa | L02CA | ;LOAD COUNTER |
| B3C8 | cmpa | L8DA2 | ;255 |
| B3CB | bis | LB3E6 | ;BRANCH IF <= CAL, ELSE |
| B3CD | ldx | #0x8DA3 | ;TABLE ADDRESS |
| B3D0 | ldab | *L00BD | ;FILTERED MPH |
| B3D2 | cmpb | #0x0F | ;15 MPH ? |
| B3D4 | bis | LB3D8 | ;BRANCH IF < 15, ELSE |
| B3D6 | ldab | #0x0F | ;LOAD 15 |
| B3D8 | LB3D8:abx | | ;ADD TO ADDRESS |
| B3D9 | ldaa | 0x00,x | ;GET A VALUE FROM THE TABLE |
| B3DB | cmpa | L02CB | ;TIMER |
| B3DE | bis | LB3E6 | ;BRANCH IF <= L02CB, ELSE |
| B3E0 | inc | L02CB | ;INCREMENT TIMER |
| B3E3 | jmp | LB6F1 | ;GO INCREMENT IAC POSITION |
| B3E6 | LB3E6:clr | L02CA | ;CLEAR |
| B3E9 | clr | L02CB | ;CLEAR |
| B3EC | LB3EC:brclr | *L0091,#0x10,LB402 | ;BRANCH IF MAF SIGNAL OKAY, ELSE |
| B3F0 | ldab | *L0057 | ;IAC POSITION |
| B3F2 | cmpb | L90C4 | ;MIN IAC STEPS WITH MAF MALF |
| B3F5 | bcc | LB3FA | ;BRANCH IF >= CAL,ELSE |
| B3F7 | jmp | LB6F1 | ;GO INCREMENT IAC POSITION |
| B3FA | LB3FA:cmpb | L90C5 | ;MIN IAC STEPS WITH MAF MALF |
| B3FD | bis | LB402 | ;BRANCH IF <= CAL, ELSE |
| B3FF | jmp | LB708 | ;GO DECREMENT IAC POSITION |
| B402 | LB402:bclr | *L00DB,#0x20 | ; |
| B405 | ldaa | *L00F7 | ;COMMANDED AIRFLOW |
| B407 | adda | L8D25 | ;5d |
| B40A | bcs | LB414 | ;BRANCH IF OVERFLOW, ELSE |
| B40C | cmpa | L0163 | ;LOW FLOW MAF |
| B40F | bcs | LB414 | ;BRANCH IF < MAF/10, ELSE |
| B411 | clr | L00DF | ;CLEAR THROTTLE FOLLOWER IAC STPES |
| B414 | LB414:ldaa | *L00AA | ;%TPS |
| B416 | suba | L8CE2 | ;1.2 %TPS HYSTERESIS |
| B419 | bcc | LB41E | ;BRANCH IF >= 1.2 %TPS, ELSE |
| B41B | jmp | LB4B2 | ;JUMP |
| B41E | LB41E:psha | | ;%TPS |
| B41F | clr | L0167 | ;CLEAR UNDER IDLE A/F ADJUST |
| B422 | ldx | #0x0000 | ;RESET TIMER |
| B425 | stx | L0165 | ; |
| B428 | bset | *L00DB,#0x20 | ; |
| B42B | bclr | *L00DB,#0xC0 | ; |
| B42E | ldaa | *L00AD | ;LOAD RPM/12.5 |
| B430 | ldab | *L00DE | ;DESIRED IDLE RPM |
| B432 | subb | L8D26 | ;SUBTRACT HYSTERESIS |
| B435 | cba | | ; |
| B436 | bcc | LB43C | ;BRANCH IF RPM > DESIRED, ELSE |
| B438 | pula | | ;GET BACK %TPS |
| B439 | jmp | LB6F1 | ;GO INCREMENT IAC POSITION |
| B43C | LB43C:brset | *L0014,#0x20,LB44B | ;BRANCH IF NO VSS SIGNAL, ELSE |
| B440 | brset | *L0015,#0x02,LB44B | ;BRANCH IF VSS INT SIGNAL MALF, ELSE |

| | | | |
|------|-------------|--------------------|--|
| B444 | ldaa | *L00BF | ;MPH*3 |
| B446 | cmpa | L8CE3 | ;0x03 - ? DEFAULT MPH |
| B449 | bis | LB451 | ;BRANCH IF <= 0x03, ELSE |
| B44B | LB44B:bclr | *L00DB,#0x01 | ;CLEAR BIT 0 |
| B44E | bclr | *L00DA,#0x02 | ;CLEAR BIT 1 |
| B451 | LB451:brclr | *L008E,#0x01,LB45F | ;BRANCH IF NOT P/N MODE, ELSE |
| B455 | ldaa | *L00AC | ;LOAD RPM/25 |
| B457 | cmpa | L8CE7 | ;6375 RPM |
| B45A | bis | LB45F | ;BRANCH IF <= CAL, ELSE |
| B45C | bset | *L0096,#0x04 | ;SET RPM HIGH IN P/N |
| B45F | LB45F:ldaa | L0320 | ;TABLE LOOKUP RESULT |
| B462 | pulb | | ;GET %TPS INTO B REG |
| B463 | mul | | ;LOOKUP RESULT * %TPS |
| B464 | lsl | | ;MUL BY 2 |
| B465 | bcs | LB46A | ;BRANCH IF OVERFLOW, ELSE |
| B467 | lsl | | ;MUL BY 2 |
| B468 | bcc | LB46C | ;BRANCH IF NO OVERFLOW, ELSE |
| B46A | LB46A:ldaa | #0xFF | ;LOAD 255 |
| B46C | LB46C:ldab | *L00AA | ; %TPS |
| B46E | cmpb | #0x19 | ;10% TPS |
| B470 | bcs | LB48B | ;BRANCH IF < 10% TPS, ELSE |
| B472 | brclr | *L00DB,#0x04,LB48B | ; |
| B476 | bclr | *L00DB,#0x04 | ; |
| B479 | ldab | L0111 | ;FILTERED TPS A/D COUNTS |
| B47C | subb | L02E6 | ;SUBTRACT LOW THROTTLE A/D COUNTS |
| B47F | cmpb | #0x02 | ;2 |
| B481 | bhi | LB48B | ;BRANCH IF > 2, ELSE |
| B483 | ldab | L0111 | ;LOAD FILTERED TPS A/D COUNTS |
| B486 | addb | #0x02 | ;ADD 2 |
| B488 | stab | L0111 | ;SAVE FILTERED TPS A/D COUNTS |
| B48B | LB48B:cmpa | L0321 | ;COMPARE MAX VALUE VS MPH |
| B48E | bcs | LB493 | ;BRANCH IF < MAX, ELSE |
| B490 | ldaa | L0321 | ;LOAD MAX VALUE |
| B493 | LB493:staa | L0164 | ;STORE ? PREV TABLE LOOKUP RESULT |
| B496 | brset | *L001A,#0x04,LB49F | ;BRANCH IF MALF (NOT ENABLED), ELSE |
| B49A | brclr | *L00D9,#0x40,LB49F | ;BRANCH IF SMC NOT ENGAGED, ELSE |
| B49E | clra | | ;CLEAR |
| B49F | LB49F:cmpa | *L00DF | ;THROTTLE FOLLOWER IAC STEPS |
| B4A1 | bis | LB4A9 | ;BRANCH IF <= L00DF, ELSE |
| B4A3 | inc | L00DF | ;INCREMENT THROTTLE FOLLOWER IAC STEPS |
| B4A6 | jmp | LB6F1 | ;GO INCREMENT IAC POSITION |
| B4A9 | LB4A9:adda | #0x06 | ;ADD 6 |
| B4AB | cmpa | *L00DF | ;COMPARE THROTTLE FOLLOWER IAC STEPS |
| B4AD | bcc | LB515 | ;BRANCH IF >= L00DF (JUMP OUT), ELSE |
| B4AF | jmp | LB708 | ;GO DECREMENT IAC POSITION |
| B4B2 | LB4B2:bset | *L00DB,#0x04 | ; |
| B4B5 | brclr | *L00DB,#0x80,LB4BC | ; |
| B4B9 | jmp | LB5B2 | ; |
| B4BC | LB4BC:ldx | #0x0000 | ; |
| B4BF | ldaa | *L00BF | ;MPH*3 |
| B4C1 | cmpa | L8CE3 | ;0x03 ? DEFAULT MPH |
| B4C4 | bis | LB4C9 | ;BRANCH IF <= DEFAULT MPH, ELSE |
| B4C6 | jmp | LB53A | ;JUMP |
| B4C9 | LB4C9:ldab | *L00DE | ;DESIRED IDLE RPM |
| B4CB | subb | *L00AD | ;SUBTRACT RPM/12.5 |
| B4CD | bcs | LB518 | ;BRANCH IF OVERSPEED, ELSE |
| B4CF | brset | *L00DA,#0x02,LB518 | ; |
| B4D3 | pshx | | ;SAVE X |
| B4D4 | ldaa | *L00DE | ;DESIRED IDLE RPM |
| B4D6 | suba | #0x20 | ;SUB 400 RPM |

| | | | | | |
|------|--------|-------|--------------------|--|--|
| B4D8 | | bcc | LB4DB | | ;BRANCH IF NO UNDERFLOW, ELSE |
| B4DA | | clra | | | ;CLEAR A |
| B4DB | LB4DB: | lsla | | | ;RESCALE |
| B4DC | | bcc | LB4E0 | | ;BRANCH IF NO OVERFLOW, ELSE |
| B4DE | | ldaa | #0xFF | | ;LOAD 255 |
| B4E0 | LB4E0: | ldx | #0x8DE3 | | ; |
| B4E3 | | jsr | L99D7 | | ;2D LOOKUP |
| B4E6 | | pulx | | | ; |
| B4E7 | | cba | | | ;COMPARE B AND DESIRED IDLE |
| B4E8 | | bhi | LB518 | | ;BRANCH IF LOOKUP RESULT > D-RPM, ELSE |
| B4EA | | brset | *L008E,#0x01,LB518 | | ;BRANCH IF P/N MODE, ELSE |
| B4EE | | ldd | *L0032 | | ;RUN TIME |
| B4F0 | | cpd | L8DE0 | | ;10 SECONDS |
| B4F4 | | bls | LB518 | | ;BRANCH IF <= CAL, ELSE |
| B4F6 | | ldaa | L02C3 | | ;LOAD |
| B4F9 | | inca | | | ;INCREMENT |
| B4FA | | bne | LB518 | | ;BRANCH IF NZ, ELSE |
| B4FC | | bset | *L00DA,#0x02 | | ;SET BIT 1 |
| B4FF | | ldaa | 0x00,y | | ;LOAD FILTERED COMMANDED AIRFLOW |
| B502 | | adda | L8DE2 | | ;ADD 3 |
| B505 | | bcs | LB50C | | ;BRANCH IF OVERFLOW, ELSE |
| B507 | | cmpa | L8DDA | | ;MAX |
| B50A | | bls | LB50F | | ;BRANCH IF <= CAL, ELSE |
| B50C | LB50C: | ldaa | L8DDA | | ;LOAD MAX FILTERED COMMANDED AIRFLOW |
| B50F | LB50F: | staa | 0x00,y | | ;SAVE FILTERED COMMANDED AIRFLOW |
| B512 | | jmp | LB5C5 | | ; |
| B515 | LB515: | jmp | LB73E | | ;SKIP THE REST |
| B518 | LB518: | ldaa | *L00F7 | | ;COMMANDED AIRFLOW |
| B51A | | adda | L8CE4 | | ;15 |
| B51D | | bcc | LB521 | | ;BRANCH IF NO OVERFLOW, ELSE |
| B51F | | ldaa | #0xFF | | ;LOAD 255 |
| B521 | LB521: | cmpa | L0163 | | ;LOW FLOW MAF |
| B524 | | bls | LB53A | | ;BRANCH IF <= MAF/10, ELSE |
| B526 | | ldx | L0165 | | ;LOAD TIMER |
| B529 | | inx | | | ;INCREMENT |
| B52A | | brset | *L0096,#0x04,LB535 | | ;BRANCH IF RPM HIGH IN P/N |
| B52E | | cpx | L8CE5 | | ;80 |
| B531 | | bhi | LB540 | | ;BRANCH IF > TIME LIMIT, ELSE |
| B533 | | bra | LB53A | | ; |
| B535 | LB535: | cpx | L8CE8 | | ;255 |
| B538 | | bhi | LB540 | | ;BRANCH IF > CAL, ELSE |
| B53A | LB53A: | stx | L0165 | | ;STORE TIMER |
| B53D | | jmp | LB6A5 | | ; |
| B540 | LB540: | ldaa | L02C3 | | ;TIMER |
| B543 | | cmpa | L8DCE | | ;20 |
| B546 | | bhi | LB550 | | ;BRANCH IF > CAL, ELSE |
| B548 | | ldaa | L02C4 | | ;LOAD |
| B54B | | cmpa | L8DCD | | ;2 |
| B54E | | bls | LB57A | | ;BRANCH IF <= CAL, ELSE |
| B550 | LB550: | brset | *L00DB,#0x01,LB57A | | ; |
| B554 | | ldaa | *L00AD | | ;LOAD RPM/12.5 |
| B556 | | suba | *L00DE | | ;DESIRED IDLE RPM |
| B558 | | bcs | LB57A | | ;BRANCH IF UNDERIDLE, ELSE |
| B55A | | cmpa | L8DF4 | | ;100 RPM |
| B55D | | bls | LB57A | | ;BRANCH IF <= CAL, ELSE |
| B55F | | brset | *L008E,#0x01,LB57A | | ;BRANCH IF P/N MODE, ELSE |
| B563 | | bset | *L00DB,#0x01 | | ; |
| B566 | | ldaa | 0x00,y | | ;FILTERED COMMANDED AIRFLOW |
| B569 | | suba | L8DF5 | | ;3 |
| B56C | | bcc | LB56F | | ;BRANCH IF NO UNDERFLOW, ELSE |

```

B56E          clra                      ;CLEAR
B56F LB56F:cmpa    L8DDC                ;0x3700 - MAX
B572          bcc    LB577              ;BRANCH IF > CAL, ELSE
B574          ldaa    L8DDC              ;LOAD MAX
B577 LB577:staa    0x00,Y                ;SAVE FILTERED COMMANDED AIRFLOW
B57A LB57A:ldaa    *L00AD                ;LOAD RPM/12.5
B57C          ldab    *L00DE              ;DESIRED IDLE RPM
B57E          addb    L8CEC                ;100 RPM
B581          cba                      ;COMPARE
B582          bcs    LB5AF                ;BRANCH IF UNDERIDLE, ELSE
B584          inc     L0168                ;INCREMENT TIMER
B587          ldaa    L0168                ;LOAD TIMER
B58A          ldab    L8CEA                ;10
B58D          brset   *L008E,#0x01,LB594 ;BRANCH IF P/N MODE, ELSE
B591          ldab    L8CEB                ;10
B594 LB594:cba                      ;COMPARE
B595          bls     LB5A2                ;BRANCH IF <= CAL, ELSE
B597          inc     L0167                ;INCREMENT OVERIDLE COUNTER
B59A          bne     LB59F                ;BRANCH IF NZ, ELSE
B59C          dec     L0167                ;DECREMENT OVERIDLE COUNTER
B59F LB59F:clr     L0168                ;CLEAR TIMER
B5A2 LB5A2:ldaa    *L00F7                ;COMMANDED AIRFLOW
B5A4          suba    L0167                ;SUBTRACT OVERIDLE A/F COUNTER
B5A7          bcc     LB5AA                ;BRANCH IF NO UNDERFLOW, ELSE, ELSE
B5A9          clra                      ;CLEAR
B5AA LB5AA:staa    *L00F7                ;COMMANDED AIRFLOW
B5AC          jmp     LB6A5                ;GO CHECK IF A/F > OR < COMMANDED

;-----
; BRANCH HERE IF UNDER IDLE
;-----

B5AF LB5AF:clr     L0167                ;CLEAR OVERIDLE COUNTER
B5B2 LB5B2:ldaa    L02C3                ;LOAD
B5B5          cmpa    L8DCE                ;20
B5B8          bhi     LB5C5                ;BRANCH IF > CAL, ELSE
B5BA          ldaa    L02C4                ;
B5BD          cmpa    L8DCD                ;2
B5C0          bhi     LB5C5                ;BRANCH IF > CAL, ELSE
B5C2          jmp     LB73E                ;SKIP THE REST

B5C5 LB5C5:bset    *L00DB,#0x80          ;SET ? A/C CLUTCH ON
B5C8          bclr    *L0096,#0x04        ;CLEAR RPM HIGH IN P/N
B5CB          bclr    *L00DB,#0x40        ;
B5CE          ldd     L0281                ;PREV 16 BIT RPM
B5D1          subd    *L00EB                ;RPM
B5D3          bpl     LB5DC                ;BRANCH IF RPM DECREASING, ELSE
B5D5          coma    LB5DC                ;INVERT MSB
B5D6          beq     LB5E1                ;BRANCH IF Z, ELSE
B5D8          ldab    #0xFF                ;LOAD 255
B5DA          bra     LB5E1                ;

B5DC LB5DC:tsta    LB5E1                ;TEST MSB
B5DD          beq     LB5E1                ;BRANCH IF Z, ELSE
B5DF          ldab    #0xFF                ;LOAD 255
B5E1 LB5E1:stab    L0160                ;IDLE RPM ERROR (N = RPM)
B5E4          ldab    *L00DE                ;DESIRED IDLE RPM
B5E6          subb    *L00AD                ;SUBTRACT RPM/12.5
B5E8          bcc     LB5F0                ;BRANCH IF RPM < DESIRED, ELSE
B5EA          bmi     LB5F4                ;BRANCH IF RPM > DESIRED, ELSE
B5EC          ldab    #0x81                ;LOAD 129
B5EE          bra     LB5F4                ;

B5F0 LB5F0:bpl     LB5F4                ;BRANCH IF OVER IDLE, ELSE
B5F2          ldab    #0x7F                ;LOAD 127

```

```

B5F4    LB5F4:stab    *L00DD
B5F6          bpl    LB5F9
B5F8          negb
B5F9    LB5F9:stab    L015F
B5FC          ld      #0x8C6B
B600          brclr   *L008E,#0x01,LB606
B604          iny
B606    LB606:ldab    L0160
B609          bpl    LB60C
B60B          negb
B60C    LB60C:cmpb    0x82,y

B60F          bls     LB61D
B611          ldab    L0160
B614          andb    #0x80
B616          addb    *L00DD
B618          bpl    LB61D
B61A          jmp     LB69D

B61D    LB61D:ldaa    L8CF1
B620          tst     L00DD
B623          bpl    LB638
B625          ldab    *L00BD
B627          brclr   *L008E,#0x01,LB62C
B62B          clrb
B62C    LB62C:cmpb    #0x0F
B62E          bls     LB632
B630          ldab    #0x0F
B632    LB632:ldx     #0x8CFB
B635          abx
B636          ldaa    0x00,x
B638    LB638:ldab    #0x02
B63A          cmpa    L015F
B63D          bhi     LB69D
B63F          ldaa    L015F
B642          cmpa    0x84,y

B645          bhi     LB649
B647          aby

B649    LB649:tst     L00DD
B64C          bpl    LB651
B64E          lslb
B64F          aby

B651    LB651:ldaa    L015F
B654          ldab    0x88,y

B657          mul
B658          lsld
B659          bcs     LB65D
B65B          bpl    LB660
B65D    LB65D:ldd      #0x7FFF
B660    LB660:brclr   *L00DD,#0x80,LB668
B664          nega
B665          negb
B666          sbca    #0x00
B668    LB668:add      L0124

```

```

;SAVE OVER/UNDER IDLE RPM (SIGNED)
;BRANCH IF OVER IDLE, ELSE
;INVERT
;SAVE OVER/UNDER IDLE (UNSIGNED)
;INDEX
;BRANCH IF NOT MODE, ELSE
;8C6C
;IDLE RPM ERROR (N = RPM)
;BRANCH IF PL, ELSE
;INVERT
;8CED OR
;8CEE
;BRANCH IF <= CAL, ELSE
;IDLE RPM ERROR (N = RPM)
;CLEAR B7
;ADD OVER/UNDER IDLE RPM (SIGNED)
;BRANCH IF UNDERIDLE, ELSE
;GO CLEAR L0124

;00
;TEST OVER/UNDER IDLE RPM (SIGNED)
;BRANCH IF UNDERIDLE, ELSE
;FILTERED MPH
;BRANCH IF NOT P/N MODE, ELSE
;CLEAR
;15
;BRANCH IF <= 15, ELSE
;LOAD 15
;TABLE ADDRESS
;ADD TO ADDRESS
;GET A VALUE FROM THE TABLE
;LOAD 2
;OVER/UNDER IDLE RPM (UNSIGNED)
;BRANCH IF < TABLE RESULT, ELSE
;LOAD OVER/UNDER IDLE RPM (UNSIGNED)
;8CEF OR
;8CF0
;BRANCH IF > CAL, ELSE
;Y = 8C6D OR
;Y = 8C6E
;OVER/UNDERIDLE RPM (SIGNED)
;BRANCH IF UNDERIDLE, ELSE
;MUL BY 2
;Y = 8C6D OR
;Y = 8C6E OR
;Y = 8C6F OR
;Y = 8C70
;OVER/UNDER IDLE RPM (UNSIGNED)
;8CF3 OR
;8CF4 OR
;8CF5 OR
;8CF6 OR
;8CF7 OR
;8CF8
;IDLE RPM ERROR * B
;MUL BY 2
;BRANCH IF OVERFLOW, ELSE
;BRANCH IF PLUS, ELSE
;LOAD
;BRANCH IF UNDERIDLE, ELSE
;INVERT
;
;ROUND
;ADD

```

| | | | |
|------|-------------|--------------------|-------------------------------------|
| B66B | bvc | LB676 | ;BRANCH IF NO OVERFLOW, ELSE |
| B66D | ldd | #0x7FFF | ;LOAD |
| B670 | bcc | LB676 | ;BRANCH IF NO OVERFLOW, ELSE |
| B672 | nega | | ;INVERT |
| B673 | negb | | ; |
| B674 | sbca | #0x00 | ;ROUND |
| B676 | LB676:std | L0124 | ;SAVE |
| B679 | bpl | LB67C | ;BRANCH IF PL, ELSE |
| B67B | nega | | ;INVERT MSB |
| B67C | LB67C:staa | L0161 | ;SAVE |
| B67F | cmpa | #0x10 | ;16 ? |
| B681 | bcs | LB6A2 | ;BRANCH IF < 16, ELSE |
| B683 | bset | *L00DA,#0x04 | ;SET BIT 2 |
| B686 | brclr | *L00DD,#0x80,LB694 | ;BRANCH IF UNDERIDLE, ELSE |
| B68A | nega | | ;INVERT |
| B68B | addd | #0x1000 | ;ADD 4096 |
| B68E | std | L0124 | ;SAVE |
| B691 | jmp | LB708 | ;GO DECREMENT IAC POSITION |
| | | | |
| B694 | LB694:subd | #0x1000 | ;SUBTRACT 4096 |
| B697 | std | L0124 | ;SAVE |
| B69A | jmp | LB6F1 | ;GO INCREMENT IAC POSITION |
| | | | |
| B69D | LB69D:clra | | ;CLEAR L0124 |
| B69E | clrb | | ; |
| B69F | std | L0124 | ; |
| B6A2 | LB6A2:jmp | LB73E | ;SKIP THE REST |
| | | | |
| B6A5 | LB6A5:bset | *L00DB,#0x40 | ; |
| B6A8 | bclr | *L00DB,#0x80 | ;SET ? A/C CLUTCH OFF |
| B6AB | ldx | #0x8DD5 | ;INDEX |
| B6AE | ldaa | *L00D9 | ;PIDMW2 |
| B6B0 | ldab | *L00F7 | ;COMMANDED AIRFLOW |
| B6B2 | subb | L0163 | ;LOW FLOW MAF |
| B6B5 | bcs | LB6BD | ;BRANCH IF A/F > COMMANDED, ELSE |
| B6B7 | dex | | ;8DD4 |
| B6B8 | bclr | *L00D9,#0x08 | ;CLEAR AIRFLOW > COMMANDED AIRFLOW |
| B6BB | bra | LB6C1 | ; |
| | | | |
| B6BD | LB6BD:bset | *L00D9,#0x08 | ;SET AIRFLOW > COMMANDED AIRFLOW |
| B6C0 | negb | | ;INVERT D-A/F |
| B6C1 | LB6C1:eora | *L00D9 | ;TOGGLE BITS |
| B6C3 | bne | LB6C9 | ;BRANCH IF THERE WAS A CHANGE, ELSE |
| B6C5 | cmpb | 0x24,x | ;8DF8 OR |
| | | | ;8DF9 |
| B6C7 | bhi | LB6D1 | ;BRANCH IF > CAL (00s), ELSE |
| B6C9 | LB6C9:ldd | #0x0000 | ;LOAD 0 |
| B6CC | std | L0169 | ;SAVE |
| B6CF | bra | LB73E | ;SKIP THE REST |
| | | | |
| B6D1 | LB6D1:brclr | *L008E,#0x01,LB6D7 | ;BRANCH IF NOT P/N MODE, ELSE |
| B6D5 | inx | | ;8DD5 OR 8DD6 |
| B6D6 | inx | | ;8DD6 OR 8DD7 |
| B6D7 | LB6D7:ldaa | 0x26,x | ;8DFA OR |
| | | | ;8DFB OR |
| | | | ;8DFC (P/N) OR |
| | | | ;8DFD (P/N) |
| B6D9 | mul | | ;CAL * D-A/F |
| B6DA | addd | L0169 | ;ADD PREVIOUS |
| B6DD | bcc | LB6E2 | ;BRANCH IF NO OVERFLOW, ELSE |
| B6DF | ldd | #0xFFFF | ;LOAD MAX |
| B6E2 | LB6E2:std | L0169 | ;SAVE |
| B6E5 | subd | L8DF6 | ;16d |
| B6E8 | bcs | LB73E | ;BRANCH IF UNDERFLOW, ELSE |

```

B6EA      std      L0169                      ;STORE
B6ED      brset    *L00D9,#0x08,LB708        ;BRANCH IF A/F > COMMANDED (DEC IAC), ELSE
B6F1      LB6F1:brclr *L0091,#0x10,LB6FE      ;BRANCH IF MAF SIGNAL OKAY, ELSE
B6F5      ldab     *L0057                    ;IAC POSITION
B6F7      cmpb     L90C5                     ;100 STEPS
B6FA      beq      LB73E                     ;BRANCH IF = CAL, ELSE
B6FC      bhi      LB708                     ;BRANCH IF > CAL, ELSE
B6FE      LB6FE:inc L0056                     ;IAC FREE RUNNING STEP COUNTER
B701      ldab     *L0057                    ;IAC POSITION
B703      incb     ;INCREMENT
B704      beq      LB72A                     ;BRANCH IF Z (GO MOVE IAC ONE STEP), ELSE
B706      bra      LB728                     ;GO SAVE IAC POSN AND MOVE IAC ONE STEP

B708      LB708:brclr *L0091,#0x10,LB715      ;BRANCH IF MAF SIGNAL OKAY, ELSE
B70C      ldab     *L0057                    ;IAC POSITION
B70E      cmpb     L90C4                     ;20 STEPS
B711      beq      LB73E                     ;BRANCH IF = 20 STEPS, ELSE
B713      bcs      LB6F1                     ;BRANCH IF < 20 STEPS, ELSE
B715      LB715:dec L0056                     ;FREE RUNNING IAC STEP COUNTER
B718      ldab     *L0057                    ;IAC POSITION
B71A      beq      LB71D                     ;BRANCH IF Z, ELSE
B71C      decb     ;DECREMENT IAC POSITION
B71D      LB71D:brset *L00DB,#0x08,LB728      ;RELATED TO A/C
B721      ldaa     *L00DF                    ;THROTTLE FOLLOWER IAC STEPS
B723      beq      LB728                     ;BRANCH IF Z, ELSE
B725      deca     ;DECREMENT
B726      staa     *L00DF                    ;STORE THROTTLE FOLLOWER IAC STEPS
B728      LB728:stab *L0057                    ;IAC POSITION

;-----
; MOVE IAC MOTOR ONE STEP
;-----
B72A      LB72A:ldab *L0056                    ;FREE RUNNING IAC STEP COUNTER
B72C      ldx      #0xFFD4                    ;INDEX
B72F      andb     #0x03                      ;MASK
B731      abx      ;ADD TO INDEX
B732      sei      ;HOLD INTERRUPTS
B733      ldaa     L4002                      ;LOAD CPU DATA LATCH
B736      anda     #0xFC                      ;CLEAR BITS 0 AND 1
B738      oraa     0x00,x                    ;SET BITS ACCORDING TO MASK
B73A      staa     L4002                      ;WRITE TO CPU DATA LATCH
B73D      cli      ;CLEAR AND ALLOW INTERRUPTS
B73E      LB73E:bclr *L009A,#0x10            ;CLEAR ? RUN-SPARK ALLOWED
B741      jmp      LCB50                      ;SKIP FUEL CALC

B744      LB744:bset *L0086,#0x20            ;TURN A/C CLUTCH OFF
B747      ldx      #0xF000                    ;0 DUTY CYCLE
B74A      brclr    *L0086,#0x80,LB758        ;BRANCH IF ENGINE NOT RUNNING, ELSE
B74E      brset    *L008E,#0x80,LB758        ;BRANCH IF A/C NOT REQUESTED, ELSE
B752      bclr     *L0086,#0x20            ;TURN A/C CLUTCH ON
B755      ldx      #0xFFFF                    ;MAX DUTY CYCLE
B758      LB758:stx L3FD2                      ;STORE A/C RELAY CONTROL OUTPUT
B75B      cmpa     *L0057                    ;IAC POSITION
B75D      beq      LB73E                     ;BRANCH IF Z, ELSE
B75F      bcs      LB708                     ;BRANCH IF < IAC POSITION, ELSE
B761      bra      LB6F1                     ;GO INCREMENT IAC POSITION

;-----
; CHECK PRNDL SWITCH
;-----
B763      LB763:jsr LFB60                      ;ENABLE RX/TX AND TRASMIT BROADCAST MSG
B766      ldx      #0x4000                    ;I/O DATA REG ADDRESS
B769      sei      ;HOLD INTERRUPTS
B76A      jsr      L9A53                      ;TX L0089, RX DATA INTO ACCA

```

| | | | |
|------|------------|--------------|------------------------------------|
| B76D | cli | | ;CLEAR AND ALLOW INTERRUPTS |
| B76E | staa | *L008A | ;STORE FMDBYTE1 |
| B770 | ldab | L800F | ;OPTION WORD - TIS SET |
| B773 | beq | LB7E8 | ;BRANCH IF CLEAR, ELSE |
| B775 | psha | | ;SAVE I/O DATA ON STACK |
| B776 | anda | #0x0F | ;%00001111 - MASK FOR LOWER NIBBLE |
| B778 | tab | | ;COPY TO B REG |
| B779 | ldx | #0xFB91 | ;PRNDL SW TRUTH TABLE |
| B77C | abx | | ;ADD TO INDEX |
| B77D | ldab | 0x10,x | ;GET VALUE FROM THE TABLE |
| B77F | cmpb | L01C1 | ;PREV PRNDL DATA SET |
| B782 | bne | LB79A | ;BRANCH IF THEY'RE NOT EQUAL, ELSE |
| B784 | lsrb | | ; |
| B785 | bcs | LB799 | ;BRANCH IF OVERFLOW, ELSE |
| B787 | lslb | | ; |
| B788 | ldaa | L01C3 | ;LOOP COUNTER |
| B78B | cmpa | #0x02 | ;2 ? |
| B78D | bcs | LB794 | ;BRANCH IF < 2, ELSE |
| B78F | bclr | *L008F,#0x04 | ;CLEAR INVALID PRNDL COMBINATION |
| B792 | bra | LB7AA | ; |
| B794 | LB794:inc | L01C3 | ;INCREMENT LOOP COUNTER |
| B797 | bra | LB7E4 | ; |
| B799 | LB799:lslb | | ;SHIFT LEFT |
| B79A | LB79A:ldaa | L01C4 | ;TIMER |
| B79D | cmpa | L977C | ;00 |
| B7A0 | bcc | LB7A7 | ;BRANCH IF > CAL, ELSE |
| B7A2 | inc | L01C4 | ;INCREMENT TIMER |
| B7A5 | bra | LB7E1 | ; |
| B7A7 | LB7A7:bset | *L008F,#0x04 | ;SET INVALID PRNDL COMBINATION |
| B7AA | LB7AA:stab | *L00A1 | ;SAVE PRNDL DATA FOR ECM |
| B7AC | clr | L01C4 | ;CLEAR TIMER |
| B7AF | pshb | | ;SAVE B ON STACK |
| B7B0 | ldaa | 0x00,x | ;GET DATA FROM TRUTH TABLE |
| B7B2 | staa | L01C5 | ;SAVE |
| B7B5 | ldaa | #0x03 | ; |
| B7B7 | bitb | #0x08 | ; |
| B7B9 | bne | LB7BC | ;BRANCH IF SET, ELSE |
| B7BB | deca | | ; |
| B7BC | LB7BC:cmpa | *L00E9 | ;TRANNNY GEAR BYTE |
| B7BE | bcc | LB7DD | ; |
| B7C0 | ldaa | L01C2 | ; |
| B7C3 | lsra | | ; |
| B7C4 | lslb | | ; |
| B7C5 | lslb | | ; |
| B7C6 | lslb | | ; |
| B7C7 | aba | | ; |
| B7C8 | tab | | ; |
| B7C9 | andb | #0xC8 | ;%11001000 |
| B7CB | cmpb | #0x48 | ;%01001000 |
| B7CD | bne | LB7D4 | ;BRANCH IF NOT Z, ELSE |
| B7CF | clr | L01CA | ;F31/TCCPWM TIMER |
| B7D2 | bra | LB7DD | ; |
| B7D4 | LB7D4:anda | #0x64 | ;%01100100 |
| B7D6 | cmpa | #0x24 | ;%00100100 |
| B7D8 | bne | LB7DD | ;BRANCH IF NZ, ELSE |
| B7DA | clr | L01CA | ;CLEAR TIMER |
| B7DD | LB7DD:pulb | | ;GET INTO B REG |
| B7DE | stab | L01C2 | ; |
| B7E1 | LB7E1:clr | L01C3 | ;PRNDL STATUS CHECK LOOP COUNTER |
| B7E4 | LB7E4:stab | L01C1 | ; |

```

B7E7          pula                                ;
B7E8  LB7E8:coma                                ;INVERT FMDBYTE1
B7E9          ldx      #0x02F5                    ;INDEX
B7EC          tab                                ;COPY TO B REG
B7ED          eorb     L02F5                      ;TOGGLE BITS
B7F0          andb     #0x40                      ;BIT 6 ?
B7F2          beq      LB800                      ;BRANCH IF Z, ELSE
B7F4          staa     0x00,x                      ;SAVE IN L02F5
B7F6          anda     #0xBF                      ;%10111111
B7F8          brclr    0x01,x,#0x40,LB802          ;BRANCH IF BIT 6 L02F6 CLEAR, ELSE
B7FC          oraa     #0x40                      ;SET BIT 6
B7FE          bra      LB802                      ;

B800  LB800:staa     0x01,x                      ;L02F6

;-----
; FIGURE OUT WHAT GEAR WE'RE IN
;-----
B802  LB802:ldab     *L008E                      ;FMD INPUT STATUS WORD
B804          stab     L0328                      ;STORE PREV STATUS FLAG
B807          pshb                                ;SAVE INPUT STATUS WORD ON STACK
B808          ldab     L8010                      ;OPTION WORD - TRANS TYPE - TIS CLEAR
B80B          bmi      LB82A                      ;BRANCH IF MANUAL TRANS, ELSE
B80D          ldab     L800F                      ;OPTION WORD - TIS SET
B810          bne      LB816                      ;BRANCH IF SET, ELSE
B812          oraa     #0x08                      ;SET BIT 3
B814          bra      LB82A                      ;

B816  LB816:psha                                ;SAVE
B817          ldab     #0x07                      ;OFFSET
B819          ldaa     *L00E9                      ;TRANNNY GEAR BYTE
B81B          deca                                ;DECREMENT GEAR BYTE
B81C          mul                                ;OFFSET * GEAR BYTE
B81D          addb     L01C5                      ;ADD PREV TRUTH TABLE DATA
B820          ldx      #0xFBB0                    ;INDEX
B823          abx                                ;ADD TO INDEX
B824          ldaa     0x00,x                      ;GET DATA FROM THE TRUTH TABLE
B826          pulb                                ;GET INTO B REG
B827          andb     #0xF0                      ;%11110000
B829          aba                                ;ADD TOGETHER
B82A  LB82A:ldab     L800C                      ;OPTION WORD - TIS SET
B82D          bne      LB839                      ;BRANCH IF SET, ELSE
B82F          bclr     *L0097,#0x01                ;CLEAR SMC ENGAGED
B832          bita     #0x10                      ;BIT 4 SET?
B834          beq      LB839                      ;BRANCH IF B4 CLEAR, ELSE
B836          bset     *L0097,#0x01                ;SET SMC ENGAGED
B839  LB839:anda     #0xDF                      ;%11011111
B83B          ldab     L8016                      ;OPTION WORD - TIS SET
                                           ;LOW OIL LEVEL LIGHT IN USE ?
B83E          beq      LB846                      ;BRANCH IF Z, ELSE
B840          bita     #0x10                      ;BIT 4 SET?
B842          bne      LB846                      ;BRANCH IF B4 SET, ELSE
B844          oraa     #0x20                      ;SET BIT 5
B846  LB846:oraa     #0x10                      ;SET BIT 4
B848          brclr    *L008E,#0x80,LB84F          ;BRANCH IF A/C REQUESTED, ELSE
B84C          bset     *L00DA,#0x20                ;SET BIT 5
B84F  LB84F:ldab     L8010                      ;OPTION WORD - TRANS TYPE - TIS CLEAR
B852          bpl      LB8AA                      ;BRANCH IF AUTO TRANS (SKIP N/V RATIO)
B854          bclr     *L00A2,#0x01                ;CLEAR BIT 0
B857          bita     #0x08                      ;BIT 3 ? IN GEAR
B859          bne      LB860                      ;BRANCH IF BIT 3 SET, ELSE
B85B          bset     *L00A2,#0x01                ;SET BIT 0
B85E          bra      LB867                      ;SET P/N MODE AND GO SAVE FMD INPUT STATUS

```

| | | | |
|------|-------------|--------------------|--|
| B860 | LB860:ldab | *L00BD | ;FILTERED MPH |
| B862 | cmpb | L9796 | ;VEH SPEED THRESHOLD FOR P/N |
| B865 | bcc | LB86B | ;BRANCH IF > CAL, ELSE |
| B867 | LB867:ldab | #0x1F | ;%00011111 (SET P/N MODE) |
| B869 | bra | LB8A7 | ;GO SAVE FMD INPUT STATUS WORD |
| B86B | LB86B:clrb | | ;START WITH Z |
| B86C | ldx | L0283 | ;N/V RATIO |
| B86F | cpx | L9797 | ;N/V RATIO INDICATING 5 TH GEAR, HIGH VALUE |
| B872 | bhi | LB87B | ;BRANCH IF > CAL, ELSE |
| B874 | cpx | L9799 | ;N/V RATIO INDICATING 5 TH GEAR, LOW VALUE |
| B877 | bcs | LB87B | ;BRANCH IF < CAL, ELSE |
| B879 | bra | LB8A7 | ;GO SAVE FMD INPUT STATUS WORD |
| B87B | LB87B:orab | #0x10 | ;SET BIT 4 (CLEAR IN 5 TH GEAR) |
| B87D | cpx | L979B | ;N/V RATIO INDICATING 4 TH GEAR, HIGH VALUE |
| B880 | bhi | LB889 | ;BRANCH IF > CAL, ELSE |
| B882 | cpx | L979D | ;N/V RATIO INDICATING 4 TH GEAR, LOW VALUE |
| B885 | bcs | LB889 | ;BRANCH IF < CAL, ELSE |
| B887 | bra | LB8A7 | ;GO SAVE FMD INPUT STATUS WORD |
| B889 | LB889:orab | #0x08 | ;SET BIT 3 (CLEAR IN 4 TH GEAR) |
| B88B | cpx | L979F | ;N/V RATIO INDICATING 3 RD GEAR, HIGH VALUE |
| B88E | bhi | LB897 | ;BRANCH IF > CAL, ELSE |
| B890 | cpx | L97A1 | ;N/V RATIO INDICATING 3 RD GEAR, LOW VALUE |
| B893 | bcs | LB897 | ;BRANCH IF < CAL, ELSE |
| B895 | bra | LB8A7 | ;GO SAVE FMD INPUT STATUS WORD |
| B897 | LB897:orab | #0x04 | ;SET BIT 2 (CLEAR IN 3 RD GEAR) |
| B899 | cpx | L97A3 | ;N/V RATIO INDICATING 2 ND GEAR, HIGH VALUE |
| B89C | bhi | LB8A5 | ;BRANCH IF > CAL, ELSE |
| B89E | cpx | L97A5 | ;N/V RATIO INDICATING 2 ND GEAR, LOW VALUE |
| B8A1 | bcs | LB8A5 | ;BRANCH IF < CAL, ELSE |
| B8A3 | bra | LB8A7 | ;GO SAVE FMD INPUT STATUS WORD |
| B8A5 | LB8A5:orab | #0x02 | ;SET BIT 1 (CLEAR IN 2 ND GEAR) |
| B8A7 | LB8A7:anda | #0xE0 | ;%11100000 |
| B8A9 | aba | | ;ADD TOGETHER |
| B8AA | LB8AA:staa | *L008E | ;FMD INPUT STATUS WORD |
| B8AC | ldab | L02B4 | ;TIMER |
| B8AF | incb | | ;INCREMENT |
| B8B0 | beq | LB8B5 | ;BRANCH IF Z, ELSE |
| B8B2 | stab | L02B4 | ;STORE TIMER |
| B8B5 | LB8B5:brset | *L008E,#0x80,LB8C9 | ;BRANCH IF A/C NOT REQUESTED, ELSE |
| B8B9 | brclr | *L00DA,#0x20,LB8C9 | ; |
| B8BD | ldab | L02B4 | ;LOAD TIMER |
| B8C0 | stab | L02B5 | ;PREV TIMER |
| B8C3 | clr | L02B4 | ;CLEAR CURRENT TIMER |
| B8C6 | bclr | *L00DA,#0x20 | ; |
| B8C9 | LB8C9:pulb | | ;GET PREV FMD INPUT STATUS WORD INTO B REG |
| B8CA | lslb | | ;SHIFT TO UPPER NIBBLE |
| B8CB | lslb | | |
| B8CC | lslb | | |
| B8CD | lslb | | |
| B8CE | anda | #0x0F | ;%00001111 |
| B8D0 | aba | | ;ADD B AND PREV FMD INPUT STATUS WORD |
| B8D1 | brclr | *L009E,#0x4E,LB8D7 | ;%01001110 |
| B8D5 | bra | LB909 | ; |
| B8D7 | LB8D7:tab | | ;COPY TO B REG |
| B8D8 | andb | #0x22 | ;%00100010 |
| B8DA | cmpb | #0x20 | ;BIT 5 ? |
| B8DC | bne | LB8E3 | ;BRANCH IF BIT 1 SET, ELSE |
| B8DE | bset | *L009E,#0x12 | ;%00010010 |

```

B8E1          bra      LB8FF

B8E3  LB8E3:tab                                ;COPY TO B REG
B8E4          andb     #0x44                    ;%01000100
B8E6          cmpb     #0x40                    ;BIT 6 ?
B8E8          bne      LB8F2                    ;BRANCH IF BIT 2 SET, ELSE
B8EA          bset     *L009E,#0x14             ;%00010100
B8ED          clr      L01C9                    ;CLEAR TIMER
B8F0          bra      LB8FF                    ;

B8F2  LB8F2:tab                                ;COPY TO B REG
B8F3          andb     #0x88                    ;%10001000
B8F5          cmpb     #0x80                    ;BIT 7 ?
B8F7          bne      LB8FF                    ;BRANCH IF BIT 8 SET, ELSE
B8F9          bset     *L009E,#0x50             ;%01010000
B8FC          clr      L01C9                    ;CLEAR TIMER
B8FF  LB8FF:tab                                ;COPY TO B REG
B900          andb     #0x44                    ;%01000100
B902          cmpb     #0x04                    ;BIT 2 ?
B904          bne      LB909                    ;BRANCH IF BIT 2 SET, ELSE
B906          bset     *L009E,#0x08             ;%00001000
B909  LB909:anda     #0x11                    ;%00010001
B90B          cmpa     #0x10                    ;BIT 4 ?
B90D          beq      LB913                    ;BRANCH IF BIT 4 CLEAR, ELSE
B90F          cmpa     #0x01                    ;BIT 0 ?
B911          bne      LB916                    ;BRANCH IF BIT 0 SET, ELSE
B913  LB913:bset     *L009E,#0x01             ;%00000001
B916  LB916:bclr     *L009D,#0xC0             ;%11000000
B919          brclr    *L00A1,#0x40,LB922       ;BRANCH IF PRNDL -> NOT REVERSE, ELSE
B91D          bset     *L009D,#0x40             ;SET BIT 6
B920          bra      LB929

B922  LB922:brset    *L008E,#0x01,LB929        ;BRANCH IF P/N MODE, ELSE
B926          bset     *L009D,#0x80            ;
B929  LB929:bclr     *L009B,#0x10             ;CLEAR REVERSE->DRV CHANGE
B92C          ldaa     *L009D                  ;
B92E          anda     #0xF0                    ;MASK FOR UPPER NIBBLE
B930          cmpa     #0x90                    ;
B932          beq      LB938                    ;BRANCH IF %10010000, ELSE
B934          cmpa     #0x60                    ;
B936          bne      LB93B                    ;BRANCH IF NOT %01100000, ELSE
B938  LB938:bset     *L009B,#0x10             ;SET REVERSE->DRV CHANGE
B93B  LB93B:bclr     *L009D,#0x30             ;CLEAR BITS 4 AND 5
B93E          brclr    *L00A1,#0x40,LB947       ;BRANCH IF PRNDL -> NOT REVERSE, ELSE
B942          bset     *L009D,#0x10             ;SET BIT 4 ? REV REPEAT FLAG
B945          bra      LB94E                    ;

B947  LB947:brset    *L008E,#0x01,LB94E        ;BRANCH IF P/N MODE, ELSE
B94B          bset     *L009D,#0x20            ;
B94E  LB94E:ldaa     L8010                    ;OPTION WORD - TRANS TYPE - TIS CLEAR
B951          bmi      LB969                    ;BRANCH IF MANUAL TRANNY, ELSE
B953          brclr    *L008E,#0x40,LB969       ;BRANCH IF BRAKE RELEASED, ELSE
B957          ldd      #0xD000                 ;LOAD 0 DC
B95A          std      L3DD4                   ;STORE TCC PWM OUTPUT
B95D          bclr     *L0096,#0x08             ;CLEAR TCC LOCKED
B960          bset     *L00A3,#0x08             ;SET OFF MODE
B963          bclr     *L00A3,#0x07             ;CLEAR APPLY, ON AND RELEASE MODES
B966          bclr     *L0089,#0x08             ;DISABLE TCC SOLENOID
B969  LB969:brclr    *L0095,#0x82,LB975       ;BRANCH IF HUD DEVICE NOT CONNECTED, ELSE
B96D          ldaa     L8023                    ;OPTION WORD - TIS CLEAR
B970          bne      LB975                    ;BRANCH IF NOT Z, ELSE
B972          jsr      L5868                    ;? UPDATE HUD DEVICE

B975  LB975:ldx      *L00ED                    ;1 SECOND TIMER

```

```

B977      cpx      #0x0005      ;5 SECONDS
B97A      bhi      LB97F        ;BRANCH IF > 5 SECONDS, ELSE
B97C      jsr      LE698        ;GO CHECK BATT/SYSTEM VOLTS
B97F      LB97F:brclr *L0097,#0x80,LB9BB ;BRANCH IF RUN-FUEL DISALLOWED (RESET ECM)
B983      ldaa     *L0083      ;BATT VOLTS
B985      cmpa     L916C        ;10 VOLTS
B988      bcs      LB9BB        ;BRANCH IF < CAL (RESET ECM), ELSE
B98A      ldaa     *L008E      ;FMD INPUT STATUS WORD
B98C      ldx      *L0032      ;RUN TIME
B98E      bne      LB9B3        ;BRANCH IF RUN TIME NOT ZERO, ELSE
B990      ldab     *L00BD      ;FILTERED MPH
B992      bne      LB9B3        ;BRANCH IF MPH NOT ZERO, ELSE
B994      ldab     L0275        ;
B997      cmpb     L916B        ;10
B99A      bhi      LB9B3        ;BRANCH IF > 10, ELSE
B99C      bne      LB9AE        ;BRANCH IF NOT Z, ELSE
B99E      inc      L0064        ;INCREMENT
B9A1      bne      LB9A6        ;BRANCH IF NOT Z, ELSE
B9A3      dec      L0064        ;DECREMENT
B9A6      LB9A6:eora #0x0F      ;%00001111
B9A8      anda     #0x0F      ;%00001111
B9AA      oraa     *L002B      ;KEEP BITS SET PREVIOUSLY
B9AC      staa     *L002B      ;STORE PRNDL SW MONITOR
B9AE      LB9AE:inc  L0275      ;
B9B1      bra      LB9BB        ;GO RESET ECM

B9B3      LB9B3:anda #0x0F      ;%00001111
B9B5      eora     #0xFF      ;TOGGLE ALL BITS
B9B7      anda     *L002B      ;CLEAR BITS SET PREVIOUSLY
B9B9      staa     *L002B      ;STORE PRNDL SW MONITOR

;-----
; RESET ECM - SIMILAR TO $5B
; ? CALLED WHEN ENGINE STALLS
;-----
B9BB      LB9BB:ldx  #0x0000      ;
B9BE      clrb     ;
B9BF      brclr   *L0095,#0x10,LBA18 ;BRANCH IF BATT VOLTS NOT < 4.0, ELSE
B9C3      ldx      *L0058      ;TIMER
B9C5      cpx      #0x0008      ;
B9C8      bne      LBA02        ;BRANCH IF NOT 8, ELSE
B9CA      bclr    *L0012,#0x1B    ;%00011011
B9CD      clra     ;
B9CE      std      *L0032      ;RUN TIME
B9D0      std      *L00ED      ;STORE 1 SECOND TIMER
B9D2      brclr   *L0086,#0x80,LBA17 ;BRANCH IF ENGINE NOT RUNNING, ELSE
B9D6      LB9D6:clra ;CLEAR
B9D7      staa     L4007        ;DISABLE INTERRUPTS
B9DA      bclr    *L0012,#0x14    ;BITS 2 AND 4
B9DD      bclr    *L0077,#0x80    ;CLEAR SES LIGHT ON AT STALL
B9E0      brclr   *L0091,#0x01,LB9E7 ;BRANCH IF SES LIGHT OFF, ELSE
B9E4      bset     *L0077,#0x80    ;SET SES LIGHT ON AT STALL
B9E7      LB9E7:lds  #0x03FF      ;STACK ADDRESS
B9EA      ldx      #0xFF00      ;
B9ED      stx      L400B        ;RESET COP ARM AND WATCHDOG
B9F0      jsr      L9CEE        ;CLEAR RAM
B9F3      jsr      L9A4A        ;I/O ROUTINE
B9F6      jsr      L9A4A        ;I/O ROUTINE
B9F9      ldx      #0xFF00      ;
B9FC      stx      L400B        ;RESET COP ARM AND WATCHDOG
B9FF      jmp      L9D7D        ;GO BACK TO INIT ROUTINE

BA02      LBA02:clr  L026C        ;CLEAR COUNTER
BA05      clr      L026B        ;

```

```

BA08      ldaa    *L00A6                ;LOAD CLNT TEMP (DEFAULTED)
BA0A      cmpa    L86CA                  ;
BA0D      bcs     LBA14                  ;BRANCH IF < CAL, ELSE
BA0F      ldaa    L8019                  ;OPTION WORD - TIS CLEAR
BA12      bne     LBA17                  ;BRANCH IF NOT Z, ELSE
BA14      LBA14:bclr  *L0012,#0x04      ;CLEAR BIT 2
BA17      LBA17:inx                    ;
BA18      LBA18:stx    *L0058            ;STORE TIMER
BA1A      cpx     #0x0226                ;550d
BA1D      bcs     LBA2B                  ;BRANCH IF < CAL, ELSE
BA1F      stab    *L0030                ;CLEAR TIMER
BA21      stab    *L0031                ;CLEAR TIMER
BA23      bclr    *L0012,#0x02          ;CLEAR VATS OKAY - MAKE VATS FAILED
BA26      LBA26:ldaa   L6000              ;
BA29      bra     LBA26                  ;WAIT FOR NEXT IRQ

;-----
; BRANCH HERE IF L0058 < 0x0226
;-----

BA2B      LBA2B:ldd     #0x0040          ;
BA2E      std     L3FCE                  ;
BA31      ldx     #0x3FAE                ;
BA34      jsr     L812                   ;STORE ACCD IN 3FAE - 3FB4
BA37      oraa    #0x80                  ;SET BIT 7
BA39      ldx     #0x4000                ;I/O DATA REG
BA3C      sei     ;HOLD INTERRUPTS
BA3D      jsr     L9A70                  ;WR ACCD TO SPI DATA REG
BA40      cli     ;CLEAR AND ALLOW INTERRUPTS

;-----
; CALCULATE DEFAULT %TPS VS MAF
;-----

BA41      ldaa    *L0057                ;IAC POSITION
BA43      ldab    L90F3                  ;IAC POSN AIRFLOW MULTIPLIER
BA46      mul     ;A * B
BA47      pshb    ;SAVE ON STACK
BA48      psha    ;
BA49      tsx     ;X POINTS TO STACK
BA4A      ldd     *L00CD                ;MAF (GM/SEC)
BA4C      subd    L90F1                  ;SUBTRACT MIN AIRFLOW
BA4F      bcs     LBA55                  ;BRANCH IF < CAL, ELSE
BA51      subd    0x00,x                ;SUBTRACT IAC AIRFLOW
BA53      bcc     LBA58                  ;BRANCH IF NO UNDERFLOW, ELSE
BA55      LBA55:ldd     #0x0000          ;LOAD 0
BA58      LBA58:pulx   ;GET IAC AIRFLOW
BA59      lsld    ;MUL BY 2
BA5A      bcc     LBA5E                  ;BRANCH IF NO OVERFLOW, ELSE
BA5C      ldaa    #0xFF                  ;LOAD 255
BA5E      LBA5E:ldx    #0x90F4          ;DEFAULT %TPS VS MAF
BA61      jsr     L99D7                  ;2D LOOKUP
BA64      staa    L02F2                  ;STORE DEFAULT %TPS VS MAF
BA67      ldaa    #0x20                  ;SEL CH #2 - TPS SENSOR
BA69      jsr     L9A18                  ;A/D READ
BA6C      staa    *L00F6                ;STORE TPS A/D COUNTS
BA6E      ldaa    *L00AA                ;%TPS
BA70      cmpa    L8A94                  ;MIN %TPS TO FORM FILTERED TPS%
BA73      bcc     LBA79                  ;BRANCH IF >= 8A94, ELSE
BA75      clra    ;CLEAR
BA76      clrb    ;
BA77      bra     LBA88                  ;

BA79      LBA79:ldab   L8A92              ;TPS FILTER COEFFICIENT
BA7C      beq     LBA88                  ;BRANCH IF Z, ELSE
BA7E      ldx     #0x0153                ;OLD FILTERED TPS%

```

```

BA81          ldy      #0x8A92                ;FILTER COEFFICIENT ADDRESS
BA85          jsr      L99F4                  ;LAG FILTER
BA88  LBA88:std    L0153                      ;FILTERED TPS%
BA8B          ldaa     *L00AA                  ;%TPS
BA8D          ldz      #0x0155                ;OLD FILTERED TPS%
BA90          ldy      #0x8ACD                ;FILTER COEFF ADDRESS
BA94          jsr      L99F4                  ;LAG FILTER
BA97          ldaa     *L00AA                  ;%TPS
BA99          ldab     L8B36                  ;0x10 - FILTER COEFFICIENT
BA9C          beq      LBAA8                  ;BRANCH IF Z - DON'T FILTER, ELSE
BA9E          ldz      #0x0157                ;OLD FILTERED TPS%
BAA1          ldy      #0x8B36                ;FILTER COEFFICIENT ADDRESS
BAA5          jsr      L99F4                  ;LAG FILTER
BAA8  LBAA8:std    L0157                      ;FILTERED TPS%
BAAB          jsr      LF4BB

;-----
; MALF CHECK 90AB BIT 1 - TPS SENSOR MALF
;-----
BAAE          ldz      #0x90AB                ;INDEX
BAB1          brclr    *L0084,#0x80,LBABC      ;BRANCH IF NOT MODE 4, ELSE
BAB5          ldab     L0231                  ;LOAD CONTROL BYTE
BAB8          andb     #0x10                  ;BIT 4 ?
BABA          bne      LBAF1                  ;BRANCH IF BIT 4 SET, ELSE
BABC  LBABC:ldaa    L0133                      ;MALF TIMER
BABF          ldab     *L00F6                  ;TPS A/D COUNTS
BAC1          cmpb     0x45,x                  ;90F0
BAC3          bhi      LBAEA                  ;BRANCH IF > 4.7 VOLTS, ELSE
BAC5          brset    *L0015,#0x10,LBAF3      ;BRANCH IF MAF SENSOR MALF, ELSE
BAC9          cmpb     0x41,x                  ;90EC
BACB          bls      LBAF3                  ;
BACD          brset    *L0013,#0x01,LBAEA      ;BRANCH IF TPS VOLTAGE HIGH, ELSE
BAD1          brclr    *L0011,#0x0E,LBAD7      ;MINOR LOOP COUNTER - 100 MSEC
BAD5          bra      LBAFA

BAD7  LBAD7:ldab     *L00AC                    ;LOAD RPM/25
BAD9          cmpb     L90ED                  ;600 RPM
BADC          bls      LBAF3                  ;BRANCH IF RPM <= 600 RPM, ELSE
BADE          ldab     L0129                  ;SCALED MAF LOOKUP RESULT
BAE1          cmpb     0x44,x                  ;90EF 0x11
BAE3          bhi      LBAF3                  ;BRANCH IF > 0x11, ELSE
BAE5          inca     ;INCREMENT TIMER
BAE6          cmpa     0x43,x                  ;90EE
BAE8          bls      LBAF7                  ;BRANCH IF <=3.2 SEC, ELSE
BAEA  LBAEA:ldaa     #0x01                    ;MASK
BAEC          ldab     #0x00                  ;OFFSET
BAEE          jsr      LF87D                  ;GO UPDATE MALF FLAGS
BAF1  LBAF1:bra      LBB22                    ;

BAF3  LBAF3:clra     ;CLEAR TIMER
BAF4          bclr     *L0013,#0x01            ;CLEAR TPS VOLTAGE HIGH
BAF7  LBAF7:staa     L0133                    ;STORE MALF TIMER

;-----
; TPS SENSOR MALF CHECK
;-----
BAFA  LBABA:ldaa     L0134                    ;TPS A/D COUNTS MALF TIMER
BAFD          ldab     *L00F6                  ;TPS A/D COUNTS
BAFF          cmpb     0x3F,x                  ;90EA - LOW LIMIT
BB01          bhi      LBB2B                  ;BRANCH IF > LOW LIMIT, ELSE
BB03          brset    *L0014,#0x80,LBB12      ;BRANCH IF TPS VOLTS LOW MALF, ELSE
BB07          brclr    *L0011,#0x0E,LBB0D      ;MINOR LOOP COUNTER - 100 MSEC
BB0B          bra      LBB32                  ;BRANCH

```

| | | | |
|------|-------------|--------------------|--|
| BB0D | LBB0D:inca | | ; INCREMENT TIMER |
| BB0E | cmpa | 0x40,x | ; 90EB - 4 SECONDS |
| BB10 | bls | LBB2F | ; BRANCH IF <= 4 SECONDS, ELSE |
| BB12 | LBB12:ldaa | #0x80 | ; MASK |
| BB14 | ldab | #0x01 | ; OFFSET - 90AC, L0014 |
| BB16 | jsr | LF87D | ; GO UPDATE MALF FLAGS |
| BB19 | ldaa | L86C4 | ; DEFAULT |
| BB1C | staa | L0111 | ; |
| BB1F | staa | L02E6 | ; |
| BB22 | LBB22:ldaa | L02F2 | ; LOAD DEFAULT %TPS VS MAF |
| BB25 | staa | *L00AA | ; %TPS |
| BB27 | staa | *L00F5 | ; %TPS |
| BB29 | bra | LBB32 | |
| BB2B | LBB2B:clra | | ; CLEAR A |
| BB2C | bclr | *L0014,#0x80 | ; CLEAR TPS VOLTS LOW MALF |
| BB2F | LBB2F:staa | L0134 | ; STORE TIMER |
| BB32 | LBB32:brclr | *L0097,#0x80,LBB39 | ; BRANCH IF RUN-FUEL NOT ALLOWED, ELSE |
| BB36 | jmp | LBC3B | ; |
| BB39 | LBB39:jsr | LF58E | ; GO LOOKUP MAF VS RPM, CORR FOR IAT |
| BB3C | ldd | #0x0014 | ; LOAD 20 |
| BB3F | ldx | *L00C0 | ; LOAD REF PERIOD |
| BB41 | fddiv | | ; MAKE THEM A FRACTION |
| BB42 | cpx | #0x0014 | ; COMPARE 20 |
| BB45 | bhi | LBB4A | ; BRANCH IF QUOTIENT > 20, ELSE |
| BB47 | ldx | #0x0000 | ; LOAD 0000 |
| BB4A | LBB4A:stx | *L00EB | ; STORE RPM |
| BB4C | ldaa | *L00E1 | ; |
| BB4E | beq | LBB56 | ; BRANCH IF Z, ELSE |
| BB50 | jsr | LE2E3 | ; GO DO COOLANT TEMP LOOKUP AND MALF CHECK |
| BB53 | LBB53:jmp | LC8BF | ; SKIP CRANKING PW CALC |
| BB56 | LBB56:brset | *L0098,#0x04,LBB5F | ; BRANCH IF CRANKING, ELSE |
| BB5A | bset | *L0098,#0x04 | ; |
| BB5D | bra | LBB53 | ; |
| BB5F | LBB5F:jsr | LE698 | ; GO CHECK BATT VOLTS |
| BB62 | ldaa | *L0083 | ; BATT VOLTS |
| BB64 | ldx | #0x8B8B | ; FUEL INJ OFFSET VS BATT VOLTS |
| BB67 | jsr | L99CC | ; 2D LOOKUP |
| BB6A | staa | *L00B4 | ; STORE LOOKUP RESULT |
| BB6C | ldaa | *L00AA | ; %TPS |
| BB6E | ldx | #0x0000 | ; 0000 |
| BB71 | bset | *L0099,#0x04 | ; SET CLEAR FLOOD MODE |
| BB74 | cmpa | L86CC | ; 60 % TPS |
| BB77 | bhi | LBBEB | ; BRANCH IF > CAL - MAKE BPW 0000 |
| BB79 | bclr | *L0099,#0x04 | ; CLEAR "CLEAR FLOOD MODE" |
| BB7C | ldx | #0x8706 | ; CRANKING PW MULTIPLIER VS %TPS |
| BB7F | jsr | L99D7 | ; 2D LOOKUP |
| BB82 | staa | L0152 | ; STORE LOOKUP RESULT |
| BB85 | ldaa | L012A | ; TRUNCATED CTS FOR TABLE LOOKUPS |
| BB88 | ldx | #0x86D3 | ; CRANK FUEL PW VS COOLANT TEMPERATURE |
| BB8B | jsr | L99CC | ; 2D LOOKUP |
| BB8E | ldx | #0x86D1 | ; CRANK TBL SCLR FOR MAX CRANK PW |
| BB91 | jsr | L98B9 | ; 8 X 16 MULTIPLY |
| BB94 | pshb | | ; |
| BB95 | psha | | ; SAVE LOOKUP RESULT ON STACK |
| BB96 | ldab | *L00A6 | ; LOAD CLNT TEMP (DEFAULTED) |
| BB98 | cmpb | L8719 | ; 120 DEG C |
| BB9B | bls | LBBAA0 | ; BRANCH IF <= 120 DEG C, ELSE |
| BB9D | tsx | | ; X POINTS TO STACK |
| BB9E | bra | LBBC8 | ; |

```

BBA0  LBBA0:ldaa    *L0030          ;? CRANKING TIMER
BBA2      lslda          ;RESCALE
BBA3      bcc    LBBA7          ;BRANCH IF NO OVERFLOW, ELSE
BBA5      ldaa    #0xFF          ;LOAD 255
BBA7  LBBA7:ldx    #0x871A        ;CRANK FUEL MULTIPLIER VS L0030 (TIMER)
BBA8      jsr    L99CC          ;2D LOOKUP
BBAD      tsx            ;X POINTS TO STACK
BBAE      jsr    L98B9          ;8 X 16 MULTIPLY
BBB1      std    0x00,x          ;STORE
BBB3      ldx    *L00C0          ;LOAD REF PERIOD
BBB5      ldd    #0x0666        ;
BBB8      fdiv           ;DIVIDE
BBB9      pshx           ;SAVE QUOTIENT ON STACK
BBBA      pula           ;GET MSB
BBBB      pulb           ;GET LSB
BBBC      ldx    #0x86F5        ;CRANK FUEL PW MULTILIER VS REF PULSES
BBBF      jsr    L99D7          ;2D LOOKUP
BBC2      tsx            ;X POINTS TO STACK
BBC3      jsr    L98B9          ;8 X 16 MULTIPLY
BBC6      std    0x00,x          ;STORE
BBC8  LBBC8:ldaa    L0152        ;MULTIPLIER VS %TPS
BBCB      jsr    L989D          ;8 x 16 MULTIPLY, PRODUCT ON STACK
BBCE      lsld           ;MUL BY 2
BBCF      pulx           ;GET BACK INTO X
BBD0      pshb           ;SAVE LSB ON STACK
BBD1      psha           ;SAVE MSB ON STACK
BBD2      ldaa    *L005C        ;FILTERED ENGINE TORQUE
BBD4      ldx    #0x873C        ;CRANK PW MULT VS ENG TORQUE (AIRFLOW)
BBD7      ldab    #0x60         ;OFFSET
BBD9      jsr    L99D3          ;2D LOOKUP
BBDC      tsx            ;X POINTS TO STACK
BBDD      jsr    L98B9          ;8 X 16 MULTIPLY
BBE0      std    0x00,x          ;SAVE ON STACK
BBE2      pulx           ;GET INTO X
BBE3      stx    L027C          ;STORE CRANKING BPW
BBE6      bpl    LBEBE          ;BRANCH IF PLUS, ELSE
BBE8      ldx    #0x7FFF        ;LOAD MAX
BBEB  LBEBE:stx    L030C        ;STORE CRANKING BPW
BBEE      brset   *L0012,#0x04,LBBFA ;BRANCH IF BIT 2, ELSE
BBF2      ldx    L030C          ;LOAD CRANKING BPW
BBF5      ldab    *L00B4        ;FUEL INJECTOR OFFSET VS BATT VOLTS
BBF7      abx            ;ADD TO X
BBF8      stx    *L00E4        ;STORE FINAL BPW
BBFA  LBBFA:brset   *L0095,#0x10,LBC30 ;BRANCH IF BATT VOLTS < 4.0, ELSE
BBFE      brset   *L0012,#0x04,LBC25 ;BRANCH IF BIT 2, ELSE
BC02      ldaa    L8019          ;OPTION FLAG - TIS CLEAR
BC05      beq    LBC25          ;BRANCH IF = ZERO, ELSE
BC07      ldaa    *L00A6        ;LOAD CLNT TEMP (DEFAULTED)
BC09      cmpa    L86CA          ;
BC0C      bls    LBC25          ;BRANCH IF <= CAL, ELSE
BC0E      ldx    L030C          ;BPW
BC11      pshx           ;SAVE ON STACK
BC12      tsx            ;X POINTS TO STACK
BC13      ldaa    L86CB          ;0x00
BC16      jsr    L989D          ;8 x 16 MULTIPLY, PRODUCT ON STACK
BC19      pulx           ;GET PRODUCT FROM STACK
BC1A      ldab    *L00B4        ;LOAD FUEL INJECTOR OFFSET VS BATT VOLTS
BC1C      abx            ;ADD TO X
BC1D      stx    *L00E4        ;STORE FINAL BPW
BC1F      jsr    LF67B          ;STORE X AS 3FF2 AND CYCLE 3FFC BITS
BC22      bset    *L0012,#0x04    ;SET BIT 2
BC25  LBC25:bset    *L0089,#0x40    ;ENABLE FUEL INJECTOR OUTPUTS
BC28      ldx    #0x0FA0        ;
BC2B      sei            ;HOLD INTERRUPTS

```

```

BC2C      jsr      L9A53                      ;WR L0089 TO 0x0FA0 ???
BC2F      cli                      ;CLEAR AND ALLOW INTERRUPTS
BC30      LBC30: ldd      *L00EB                ;RPM
BC32      cpd      #0x00C8                    ;200
BC36      bhi      LBC3B                    ;BRANCH IF TRANNY RPM > 200 RPM, ELSE
BC38      jmp      LC8BF                    ;SKIP FUEL CALC

*****
; MAIN FUEL CALC
;-----
; CALCULATE LV8
;-----

BC3B      LBC3B: ldd      *L00BA                ;LOAD LV8
BC3D      std      *L00BB                    ;STORE PREV LV8
BC3F      ldx      *L00C0                    ;LOAD REF PERIOD
BC41      ldd      *L00CD                    ;MAF (GMS/SEC)
BC43      jsr      L98FA                      ;16 X 16 MULTIPLY
BC46      pshb                      ;SAVE LSB ON STACK
BC47      psha                      ;SAVE MSB ON STACK
BC48      tsx                      ;X POINTS TO STACK
BC49      ldaa     L86C8                    ;LV8 FUDGE FACTOR
BC4C      jsr      L989D                      ;8 x 16 MULTIPLY, PRODUCT ON STACK
BC4F      staa     *L00BA                    ;STORE LV8
BC51      pulx                      ;RESTORE
BC52      brset    *L0013,#0x01,LBC5E        ;BRANCH IF TPS VOLTAGE HIGH, ELSE
BC56      brset    *L0014,#0x80,LBC5E        ;BRANCH IF TPS VOLTS LOW, ELSE
BC5A      brclr    *L0015,#0x10,LBC6A        ;BRANCH IF NO MAF ERROR, ELSE
BC5E      LBC5E: ldd      L8698                ;0xBE00
BC61      std      *L005A                    ;FILTERED ENGINE TORQUE
BC63      ldd      L869A                      ;0x9600
BC66      std      *L005C                    ;FILTERED ENGINE TORQUE
BC68      bra      LBCCF

BC6A      LBC6A: brset    *L009E,#0x08,LBCCF    ;THIS SET AT B906 ? RELATED TO GEAR ?
BC6E      brset    *L008E,#0x01,LBCCF    ;BRANCH IF P/N MODE, ELSE
BC72      ldaa     *L00AA                    ;%TPS
BC74      cmpa     L869E                      ;50% TPS
BC77      bcs      LBC9C                      ;BRANCH IF TPS% < 50%, ELSE
BC79      cmpa     L869F                      ;58.9% TPS
BC7C      bcc      LBC9C                      ;BRANCH IF TPS% >= 58.9%, ELSE
BC7E      ldaa     *L00AC                    ;LOAD RPM/25
BC80      cmpa     L869C                      ;2200 RPM
BC83      bls      LBC9C                      ;BRANCH IF <= 2200 RPM, ELSE
BC85      cmpa     L869D                      ;2800 PRM
BC88      bcc      LBC9C                      ;BRANCH IF >= 2800 RPM, ELSE
BC8A      ldd      L0276                      ;ENGINE TORQUE
BC8D      lsld                      ;MUL BY 2
BC8E      bcc      LBC92                      ;BRANCH IF NO OVERFLOW, ELSE
BC90      ldaa     #0xFF                      ;LOAD 255
BC92      LBC92: ldx      #0x005A                ;OLD FILTERED ENGINE TORQUE
BC95      ldy      #0x86A0                ;FILTER COEFFICIENT ADDRESS
BC99      jsr      L99F4                      ;LAG FILTER
BC9C      LBC9C: ldaa     L01AE                ;EGR DUTY CYCLE
BC9F      beq      LBCCF                      ;BRANCH IF Z (SKIP FILTER), ELSE
BCA1      ldaa     *L00AA                    ;%TPS
BCA3      cmpa     L86A3                      ;
BCA6      bcs      LBCCF                      ;BRANCH IF < CAL, ELSE
BCA8      cmpa     L86A4                      ;
BCAB      bcc      LBCCF                      ;BRANCH IF > CAL, ELSE
BCAD      ldaa     *L00AC                    ;LOAD RPM/25
BCAF      cmpa     L86A1                      ;
BCB2      bls      LBCCF                      ;BRANCH IF <= CAL, ELSE
BCB4      cmpa     L86A2                      ;

```

| | | | |
|--------------------------|-------------|--------------------|---------------------------------------|
| BCB7 | bcc | LBCCF | ;BRANCH IF >= CAL, ELSE |
| BCB9 | ldd | L0276 | ;ENGINE TORQUE |
| BCBC | lslld | | ;MULTIPLY BY 2 |
| BCBD | bcs | LBCC3 | ;BRANCH IF OVERFLOW, ELSE |
| BCBF | adda | #0x32 | ;ADD 50 |
| BCC1 | bcc | LBCC5 | ;BRANCH IF NO OVERFLOW, ELSE |
| BCC3 | LBCC3:ldaa | #0xFF | ;LOAD |
| BCC5 | LBCC5:ldx | #0x005C | ;OLD FILTERED ENGINE TORQUE |
| BCC8 | ldy | #0x86A5 | ;FILTER COEFFICIENT ADDRESS |
| BCCC | jsr | L99F4 | ;LAG FILTER |
| ;----- | | | |
| ; LAUNCH MODE COND MET ? | | | |
| ;----- | | | |
| BCCF | LBCCF:brset | *L0015,#0x02,LBD2C | ;BRANCH IF VSS INT SIGNAL MALF, ELSE |
| BCD3 | brset | *L008F,#0x20,LBD08 | ;BRANCH IF LAUNCH MODE, ELSE |
| BCD7 | ldaa | *L00BD | ;FILTERED MPH |
| BCD9 | cmpa | L841B | ;1 MPH |
| BCDC | bcc | LBD2C | ;BRANCH IF > 1 MPH, ELSE |
| BCDE | ldaa | *L00BA | ;LV8 |
| BCE0 | cmpa | L841C | ;80 |
| BCE3 | bls | LBD2C | ;BRANCH IF <= 80, ELSE |
| BCE5 | ldaa | *L00AA | ;%TPS |
| BCE7 | suba | L0153 | ;PREV FILTERED TPS% |
| BCEA | bcs | LBD2C | ;BRANCH IF TPS DECREASING, ELSE |
| BCEC | cmpa | L841D | ;1.1 %TPS |
| BCEF | bcs | LBD2C | ;BRANCH IF DELTA TPS < 1.1 %TPS, ELSE |
| BCF1 | ldaa | *L00A6 | ;LOAD CLNT TEMP (DEFAULTED) |
| BCF3 | cmpa | L841E | ;80 DEG C |
| BCF6 | bcs | LBD2C | ;BRANCH IF < 80 DEG C, ELSE |
| BCF8 | ldd | *L0032 | ;RUN TIME |
| BCFA | cpd | L841F | ;40 SECONDS |
| BCFE | bcs | LBD2C | ;BRANCH IF < 40 SECONDS, ELSE |
| BD00 | bset | *L008F,#0x20 | ;SET LAUNCH MODE |
| BD03 | clr | L0173 | ;CLEAR TIMER |
| BD06 | bra | LBD2F | |
| | | | |
| BD08 | LBD08:ldaa | L0173 | ;LOAD TIMER |
| BD0B | inca | | ;INCREMENT TIMER |
| BD0C | cmpa | L8421 | ;16 |
| BD0F | bhi | LBD2C | ;BRANCH IF > 16, ELSE |
| BD11 | staa | L0173 | ;STORE TIMER |
| BD14 | ldaa | *L00AA | ;%TPS |
| BD16 | suba | L0153 | ;PREV FILTERED TPS% |
| BD19 | bcc | LBD2F | ;BRANCH IF TPS NOT DECREASING, ELSE |
| BD1B | nega | | ;INVERT RESULT |
| BD1C | cmpa | L8422 | ;1.9% TPS |
| BD1F | bhi | LBD2C | ;BRANCH IF HIGHER, ELSE |
| BD21 | ldaa | *L00BA | ;LOAD LV8 |
| BD23 | suba | *L00BB | ;PREVIOUS LV8 |
| BD25 | bcs | LBD2F | ;BRANCH IF LV8 DECREASING, ELSE |
| BD27 | cmpa | L8423 | ;254 COUNTS |
| BD2A | bls | LBD2F | ;BRANCH IF <= CAL, ELSE |
| BD2C | LBD2C:bclr | *L008F,#0x20 | ;CLEAR LAUNCH MODE FLAG |
| BD2F | LBD2F:ldaa | L8010 | ;OPTION WORD - TRANS TYPE - TIS CLEAR |
| BD32 | bpl | LBD6F | ;BRANCH IF AUTO TRANS, ELSE |
| BD34 | brset | *L001A,#0x01,LBD6F | ;BRANCH IF MALF (NOT ENABLED), ELSE |
| BD38 | brclr | *L00A2,#0x01,LBD6F | ; |
| BD3C | ldaa | *L00AA | ;%TPS |
| BD3E | cmpa | L8B62 | ;00 |
| BD41 | bhi | LBD6F | ;BRANCH IF > 0 %TPS, ELSE |
| BD43 | ldaa | *L00BD | ;FILTERED MPH |
| BD45 | cmpa | L8B63 | ; |
| BD48 | bls | LBD6F | ;BRANCH IF <= CAL, ELSE |

| | | | |
|------|-------------|--------------------|-------------------------------------|
| BD4A | ldaa | *L00BA | ;LOAD LV8 |
| BD4C | cmpa | L8B64 | ;00 |
| BD4F | bls | LBD58 | ;BRANCH IF <= 00, ELSE |
| BD51 | ldaa | *L00AC | ;LOAD RPM/25 |
| BD53 | cmpa | L8B65 | ;00 RPM |
| BD56 | bhi | LBD6A | ;BRANCH IF > 0, ELSE |
| BD58 | LBD58:brclr | *L0092,#0x10,LBD6F | ;BRANCH IF NOT BIT 4, ELSE |
| BD5C | ldaa | *L00AC | ;LOAD RPM/25 |
| BD5E | cmpa | L8B67 | ;00 RPM |
| BD61 | bcs | LBD6F | ;BRANCH IF < 0, ELSE |
| BD63 | ldaa | *L00BA | ;LOAD LV8 |
| BD65 | cmpa | L8B66 | ;00 LV8 |
| BD68 | bcs | LBD6F | ;BRANCH IF < 0, ELSE |
| BD6A | LBD6A:bset | *L0092,#0x10 | ;SET BIT 4 |
| BD6D | bra | LBD72 | |
| BD6F | LBD6F:bclr | *L0092,#0x10 | ;CLEAR BIT 4 |
| BD72 | LBD72:ldx | #0x8B56 | ;INDEX |
| BD75 | ldaa | *L00A6 | ;LOAD CLNT TEMP (DEFAULTED) |
| BD77 | cmpa | 0x0A,x | ;8B60 - 65 DEG C |
| BD79 | bcs | LBDDD | ;BRANCH IF < 65 DEG C, ELSE |
| BD7B | brset | *L008E,#0x01,LBDDD | ;BRANCH IF P/N MODE, ELSE |
| BD7F | brset | *L002A,#0x10,LBDDD | ;EGR BIT STATUS FLAG |
| BD83 | ldaa | L0263 | ;MPH |
| BD86 | cmpa | L8B61 | ;25 MPH |
| BD89 | bcs | LBDDD | ;BRANCH IF < 25 MPH, ELSE |
| BD8B | ldab | L02C1 | ; |
| BD8E | bne | LBDDD | ;BRANCH IF NOT Z, ELSE |
| BD90 | ldab | L02C2 | ; |
| BD93 | bne | LBDDD | ;BRANCH IF NOT Z, ELSE |
| BD95 | brset | *L0013,#0x01,LBDA3 | ;BRANCH IF TPS VOLTAGE HIGH, ELSE |
| BD99 | brset | *L0014,#0x80,LBDA3 | ;BRANCH IF TPS VOLTS LOW MALF, ELSE |
| BD9D | ldaa | *L00AA | ;%TPS |
| BD9F | cmpa | 0x09,x | ;2.4 %TPS |
| BDA1 | bhi | LBDDD | ;BRANCH IF > 2.4 %TPS, ELSE |
| BDA3 | LBDA3:ldaa | L012C | ;PREV RPM/12.5 |
| BDA6 | suba | *L00AD | ;SUBTRACT RPM/12.5 |
| BDA8 | bcs | LBDAE | ;BRANCH IF RPM INCREASING, ELSE |
| BDAA | cmpa | 0x06,x | ;75 RPM |
| BDAC | bcc | LBDDD | ;BRANCH IF DELTA > 75 RPM, ELSE |
| BDAE | LBDAE:brclr | *L009B,#0x02,LBDB3 | ;BRANCH IF DFCO NOT ENABLED, ELSE |
| BDB2 | inx | | ;8B57 |
| BDB3 | LBDB3:brset | *L0015,#0x10,LBDBD | ;BRANCH IF MAF SENSOR MALF, ELSE |
| BDB7 | ldaa | 0x04,x | ;8B5A OR |
| | | | ;8B5B |
| BDB9 | cmpa | *L00BA | ;COMPARE LV8 |
| BDBB | bcs | LBDDD | ;BRANCH IF > CAL, ELSE |
| BDBD | LBDBD:ldaa | 0x00,x | ;8B56 OR |
| | | | ;8B57 |
| BDBF | brset | *L00A1,#0x02,LBDC9 | ;BRANCH IF PRNDL -> DRIVE 1, ELSE |
| BDC3 | brset | *L00A1,#0x04,LBDC9 | ;BRANCH IF PRNDL -> DRIVE 2, ELSE |
| BDC7 | ldaa | 0x02,x | ;8B58 OR |
| | | | ;8B59 |
| BDC9 | LBDC9:cmpa | *L00AC | ;LOAD RPM/25 |
| BDCB | bcc | LBDDD | ;BRANCH IF <= CAL, ELSE |
| BDCD | ldd | L0193 | ;LOAD TIMER |
| BDD0 | bne | LBDD8 | ;BRANCH IF NOT Z, ELSE |
| BDD2 | bset | *L009B,#0x02 | ;SET DFCO ENABLED |
| BDD5 | jmp | LBE8F | ;CLEAR AE ENABLED |
| BDD8 | LBDD8:subd | #0x0001 | ;DECREMENT BY 1 |
| Bddb | bra | LBDE0 | ; |
| BDDD | LBDDD:ldd | L8B5D | ;256 |

```

BDE0    LBDE0:std      L0193                      ;STORE TIMER
BDE3          bclr     *L009B,#0x02              ;CLEAR DFCO ENABLED
BDE6          brset    *L0013,#0x01,LBE08        ;BRANCH IF TPS VOLTAGE HIGH, ELSE
BDEA          brset    *L0014,#0x80,LBE08        ;BRANCH IF TPS VOLTS LOW MALF, ELSE
BDEE          ldaa     L801E                      ;OPTION WORD - TIS SET
BDF1          beq      LBDF7                      ;BRANCH IF Z, ELSE
BDF3          brset    *L0015,#0x10,LBE08        ;BRANCH IF MAF SENSOR MALF, ELSE

;-----
; ACCEL ENRICHMENT ROUTINE
;-----

BDF7    LBDF7:ldaa     *L00AA                      ;%TPS
BDF9          suba     L0155                      ;PREV FILTERED TPS%
BDFC          bcc      LBE0B                      ;BRANCH IF TPS INCREASING, ELSE
BDFE          nega     ;MAKE NEG DELTA TPS
BDF7          cmpa     L8ACF                      ;NEG D-TPS THRESHOLD TO DISABLE A.E.
BE02          bhi      LBE08                      ;BRANCH IF NEG D-TPS > CAL, ELSE
BE04          brset    *L009A,#0x02,LBE17        ;BRANCH IF A.E. ENABLED, ELSE
BE08    LBE08:jmp      LBE8F                      ;JUMP - CLEAR A.E. ENABLED

BE0B    LBE0B:brset    *L009A,#0x02,LBE17        ;BRANCH IF AE ENABLED, ELSE
BE0F          cmpa     L8ACE                      ;POS D-TPS THRESHOLD TO ALLOW ACCEL ENRICH
BE12          bls      LBE8F                      ;BRANCH IF < CAL, ELSE
BE14          bset     *L009A,#0x02              ;SET A.E. ENABLED
BE17    LBE17:ldx      #0x8AD3                    ;A.E. FACTOR VS D-TPS
BE1A          cmpa     L015A                      ;COMPARE PREV D-TPS
BE1D          bls      LBE22                      ;BRANCH IF <= PREV D-TPS, ELSE
BE1F          staa     L015A                      ;STORE PREV D-TPS
BE22    LBE22:ldaa     L015A                      ;LOAD PREV D-TPS
BE25          lsra     ;RESCALE
BE26          jsr      L99D7                      ;2D LOOKUP
BE29          ldab     L015C                      ;LOAD A.E. MULTIPLIER VS COOLANT TEMP
BE2C          mul      ;A * B
BE2D          lsld     ;MUL BY 2
BE2E          bcc      LBE32                      ;BRANCH IF NO OVERFLOW, ELSE
BE30          ldaa     #0xFF                      ;LOAD 255
BE32    LBE32:ldab     L015D                      ;LOAD A.E. MULTIPLIER VS IAT
BE35          mul      ;A * B
BE36          lsld     ;MUL BY 2
BE37          bcc      LBE3B                      ;BRANCH IF NO OVERFLOW, ELSE
BE39          ldaa     #0xFF                      ;LOAD 255
BE3B    LBE3B:psha     ;SAVE A.E. FACTOR ON STACK
BE3C          ldd      *L0032                    ;RUN TIME
BE3E          cpd      #0x0040                    ;64 SECONDS
BE42          bhi      LBE47                      ;BRANCH IF > 64 SECONDS, ELSE
BE44          tba      ;COPY TO A REG
BE45          bra      LBE49

BE47    LBE47:ldaa     #0x40                      ;LOAD 64 SECONDS
BE49    LBE49:lsla     ;RESCALE
BE4A          ld      #0x8B2B                    ;A.E. MULTIPLIER VS RUN TIME
BE4D          jsr      L99D7                      ;2D LOOKUP
BE50          pulb     ;GET A.E. FACTOR FROM STACK
BE51          mul      ;A * B
BE52          lsld     ;MUL BY 2
BE53          bcc      LBE57                      ;BRANCH IF NO OVERFLOW, ELSE
BE55          ldaa     #0xFF                      ;LOAD 255
BE57    LBE57:psha     ;SAVE MSB OF A.E. FACTOR ON STACK
BE58          ld      #0x8AF9                    ;A.E. MULTIPLIER VS MPH
BE5B          ldaa     *L00BD                    ;FILTERED MPH
BE5D          cmpa     #0x40                      ;64 MPH
BE5F          bcs      LBE63                      ;BRANCH IF < 64 MPH, ELSE
BE61          ldaa     #0x3F                      ;LOAD 63 MPH
BE63    LBE63:lsla     ;RESCALE

```

```

BE64      lslda
BE65      jsr      L99D7
BE68      pulb
BE69      mul
BE6A      lslda
BE6B      bcc      LBE6F
BE6D      ldaa     #0xFF
BE6F      LBE6F:ldab L0100
BE72      cmpb     L8AD0
BE75      bcs      LBE99
BE77      cmpb     L8AD1
BE7A      bhi      LBE8F
BE7C      lsrbb
BE7D      ldx      #0x8B0A
BE80      cmpb     #0x20
BE82      bls      LBE86
BE84      ldab     #0x20
BE86      LBE86:abx
BE87      ldab     0x00,x
BE89      stab     L046E
BE8C      mul
BE8D      bra      LBE9A

BE8F      LBE8F:bclr *L009A,#0x02
BE92      clra
BE93      staa     L0100
BE96      staa     L015A
BE99      LBE99:clra
BE9A      LBE9A:staa L0159

;-----
; STALL SAVER ASYNC ROUTINE
;-----

BE9D      ldaa     L8A92
BEA0      beq      LBE8D
BEA2      brset    *L0013,#0x01,LBE8D
BEA6      brset    *L0014,#0x80,LBE8D
BEAA      ldaa     L801D
BEAD      beq      LBE8B
BEAF      brset    *L0015,#0x10,LBF31
BEB3      LBE8B:ldaa *L00AC
BEB5      cmpa     L8A95
BEB8      bhi      LBF31
BEBA      inc      L017A
BEBD      ldaa     L017A
BEC0      cmpa     L8A93
BEC3      bls      LBE8C
BEC5      bset     *L00D9,#0x80
BEC8      LBE8C:ldaa *L00AA
BECA      suba     L0153
BECD      bhi      LBEDA
BECF      coma
BED0      cmpa     L8A8F
BED3      bls      LBEE3
BED5      bclr     *L00D9,#0x80
BED8      LBED8:bra LBF31

BEDA      LBEDA:brset *L00D9,#0x80,LBF31
BEDE      cmpa     L8A8E
BEE1      bcc      LBEF3
BEE3      LBEE3:brset *L00D9,#0x80,LBF31
BEE7      ldaa     L8A96
BEEA      coma
BEEB      ldx      #0x0106

;
;2D LOOKUP
;GET BACK A.E. FACTOR FROM STACK
;A * B
;MUL BY 2
;BRANCH IF NO OVERFLOW, ELSE
;LOAD 255
;A.E. TIMER, FROM XIRQ ROUTINE
;00
;BRANCH IF < CAL, ELSE
;80
;BRANCH IF > CAL, ELSE
;RESCALE
;TABLE ADDRESS
;COMPARE 32
;BRANCH IF <= 32, ELSE
;LOAD 32
;ADD TO ADDRESS
;GET A VALUE FROM THE TABLE
;STORE LOOKUP RESULT
;A.E. FACTOR * LOOKUP RESULT
;GO STORE A.E. TERM

;CLEAR A.E. ENABLED
;
;A.E. TIMER
;STORE PREV D-TPS
;
;STORE A.E. TERM

;-----
; STALL SAVER ASYNC ROUTINE
;-----

;TPS FILTER COEFFICIENT
;BRANCH IF Z, ELSE
;BRANCH IF TPS VOLTAGE HIGH, ELSE
;BRANCH IF TPS VOLTS LOW MALF, ELSE
;OPTION WORD - TIS SET
;BRANCH IF Z, ELSE
;BRANCH IF MAF SENSOR MALF, ELSE
;LOAD RPM/25
;1200 RPM
;BRANCH IF > CAL (GO CLR ASYNC PW), ELSE
;INCREMENT ASYNC TIMER
;LOAD ASYNC TIMER
;ASYNC TIME LIMIT
;BRANCH IF <= CAL, ELSE
;SET STALL SAVER ASYNC ENABLED
;%TPS
;PREV FILTERED TPS%
;BRANCH IF TPS INCREASING, ELSE
;INVERT RESULT
;0.4 %TPS
;BRANCH IF NEG D-TPS <= 0.4 %TPS
;SET ASYNC A.E. DISABLED
;GO CLEAR ASYNC PW

;BRANCH IF ASYNC DISABLED, ELSE
;POS D-TPS TO STAY IN A.E. MODE
;BRANCH IF D-TPS >= CAL, ELSE
;BRANCH IF ASYNC DISABLED, ELSE
;ASYNC PW DECAY MULTIPLIER
;INVERT
;ASYNC PW

```

```

BEEE      jsr      L98B9                      ;8 X 16 MULTIPLY
BEF1      bra      LBF0A

BEF3      LBEF3:staa    L015B                      ;PREV D-TPS
BEF6      ldaa      *L00A6                      ;LOAD CLNT TEMP (DEFAULTED)
BEF8      ldx       #0x8A97                      ;ASYNC FACTOR VS COOLANT TEMP
BEFB      jsr       L99CC                      ;2D LOOKUP
BEFE      ldab      L015B                      ;D-TPS
BF01      mul       ;A * B
BF02      addd      L0106                      ;ADD ASYNC PW
BF05      bcc       LBF0A                      ;BRANCH IF NO OVERFLOW, ELSE
BF07      ldd       #0xFFFF                      ;LOAD MAX
BF0A      LBF0A:pshb                      ;SAVE ON STACK
BF0B      psha                      ;
BF0C      ldd       *L0032                      ;RUN TIME
BF0E      cpd       #0x0040                      ;64 SECONDS ?
BF12      bhi       LBF17                      ;BRANCH IF > 64 SECONDS, ELSE
BF14      tba                      ;TRANSFER LSB TO A REG
BF15      bra      LBF19                      ;

BF17      LBF17:ldaa    #0x40                      ;LOAD 64
BF19      LBF19:lsla                      ;RESCALE
BF1A      ldx       #0x8AA1                      ;ASYNC FACTOR VS RUN TIME
BF1D      jsr       L99D7                      ;2D LOOKUP
BF20      tsx                      ;X POINTS TO STACK
BF21      jsr       L98B9                      ;8 X 16 MULTIPLY
BF24      lsld                      ;MUL BY 2
BF25      bcc       LBF2A                      ;BRANCH IF NO OVERFLOW, ELSE
BF27      ldd       #0xFFFF                      ;LOAD MAX
BF2A      LBF2A:pulx                      ;GET ASYNC PW FROM STACK
BF2B      cpd       L8A90                      ;MIN ASYNC PW
BF2F      bhi       LBF36                      ;BRANCH IF > MIN, ELSE
BF31      LBF31:clra                      ;
BF32      clrb                      ;
BF33      staa      L017A                      ;ASYNC TIMER
BF36      LBF36:std     L0106                      ;STORE ASYNC PW

;-----
; DECEL ENLEANMENT ROUTINE
;-----

BF39      ldaa      L8B36                      ;TPS FILTER COEFF
BF3C      beq       LBF94                      ;BRANCH IF Z, ELSE
BF3E      brset     *L0013,#0x01,LBF94          ;BRANCH IF TPS VOLTAGE HIGH, ELSE
BF42      brset     *L0014,#0x80,LBF94          ;BRANCH IF TPS VOLTS LOW MALF, ELSE
BF46      ldab      L0101                      ;DE TIMER
BF49      cmpb      L8B3A                      ;0xFE
BF4C      bhi       LBF94                      ;BRANCH IF > CAL, ELSE
BF4E      ldaa      *L00AA                      ;%TPS
BF50      suba      L0157                      ;PREV FILTERED TPS%
BF53      bcs       LBF60                      ;BRANCH IF TPS DECREASING, ELSE
BF55      cmpa      L8B38                      ;1.2 %TPS
BF58      bhi       LBF94                      ;BRANCH IF > CAL, ELSE
BF5A      brset     *L009A,#0x04,LBF6D          ;BRANCH IF DE ENABLED, ELSE
BF5E      bra      LBF94                      ;GO DISABLE DECEL ENLEANMENT

BF60      LBF60:nega                      ;MAKE NEG D-TPS
BF61      brset     *L009A,#0x04,LBF6D          ;BRANCH IF DE ENABLED, ELSE
BF65      cmpa      L8B37                      ;3.1 %TPS
BF68      bls       LBF94                      ;BRANCH IF <= CAL ELSE
BF6A      bset      *L009A,#0x04              ;SET DE ENABLED
BF6D      LBF6D:ldx     #0x8B3C                      ;TABLE ADDRESS
BF70      cmpa      L029F                      ;PREV D-TPS
BF73      bls       LBF78                      ;BRANCH IF DTPS <= PREV D-TPS, ELSE
BF75      staa      L029F                      ;STORE DELTA TPS

```

```

BF78    LBF78:ldaa    L029F                ;LOAD DELTA TPS
BF7B          lsra                ;RESCALE
BF7C          jsr     L99D7            ;2D LOOKUP
BF7F          tab                ;TRANSFER LOOKUP RESULT TO B REG
BF80          ldaa    L0101            ;DECEL ENLEANMENT TIMER
BF83          cmpa    L8B39            ;0x06
BF86          bcs     LBF9E            ;BRANCH IF < CAL, ELSE
BF88          ldx     #0x8B45          ;DE MULTIPLIER VS DE TIME
BF8B          jsr     L99D7            ;2D LOOKUP
BF8E          staa    L02A2            ;STORE LOOKUP RESULT
BF91          mul                ;MUL THE LOOKUP RESULTS
BF92          bra     LBF9F            ;GO SAVE DE TERM

BF94    LBF94:bclr    *L009A,#0x04        ;CLEAR DE ENABLED
BF97          clra                ;CLEAR
BF98          staa    L0101            ;CLEAR DE TIMER
BF9B          staa    L029F            ;D-TPS FOR AE
BF9E    LBF9E:clra                ;CLEAR
BF9F    LBF9F:staa    L02A1            ;SAVE DE TERM

;-----
; P/N -> IN GEAR FUEL ENRICHMENT
;-----

BFA2          brset   *L001D,#0x40,LBFFE    ;BRANCH IF TRANS RANGE SW ERROR, ELSE
BFA6          brset   *L008E,#0x01,LBFF5    ;BRANCH IF P/N MODE, ELSE
BFAA          brset   *L0098,#0x10,LBFFE    ;BRANCH IF L02B9 (TIMER) = 255, ELSE
BFAE          ldx     #0x8A5B                ;P/N -> IN GEAR FUEL ENRICHMENT ENABLE
                                           ; DELAY TIME VS CTS

BFB1          ldaa    *L00A6                ;LOAD CLNT TEMP (DEFAULTED)
BFB3          lsra                ;RESCALE
BFB4          jsr     L99D7            ;2D LOOKUP
BFB7          cmpa    L02B8            ;COMPARE TIMER
BFBA          bls     LBFC1            ;BRANCH IF TIMER >= LOOKUP RESULT, ELSE
BFBC          inc     L02B8            ;INCREMENT PRETIMER
BFBF          bra     LBFFE            ;GO CLEAR ENRICHMENT TERM

BFCl    LBFC1:brclr    *L009A,#0x20,LBFC8    ;BRANCH IF ? RUN-FUEL/SPARK, ELSE
BFC5          inc     L02B9                ;INCREMENT TIMER
BFC8    LBFC8:ldab    *L00BF                ;MPH*3
BFCA          cmpb    L8A5A                ;0x10
BFCD          bcc     LBFFE            ;BRANCH IF >= 16 MPH, ELSE
BFCF          ldx     #0x8A64                ;
BFD2          ldaa    *L00A6                ;LOAD CLNT TEMP (DEFAULTED)
BFD4          lsra                ;RESCALE
BFD5          jsr     L99D7            ;2D LOOKUP
BFD8          ldab    L02B9                ;TIMER
BFDB          cmpb    #0xFF                ;255 ?
BFDD          bne     LBFE2            ;BRANCH IF NOT 255, ELSE
BFDF          bset    *L0098,#0x10        ;SET BIT 4
BFE2    LBFE2:lsrb                ;RESCALE TIMER
BFE3          ldx     #0x8A6D                ;
BFE6          cmpb    #0x20                ;COMPARE 32
BFE8          bls     LBFE2            ;BRANCH IF <= 32, ELSE
BFEA          ldab    #0x20                ;LOAD 32
BFEC    LBFE2:abx                ;ADD TO ADDRESS
BFED          ldab    0x00,x                ;GET A VALUE FROM THE TABLE
BFEF          mul                ;8A64 LOOKUP RESULT * 8A6D LOOKUP RESULT
BFF0          staa    L02BA            ;STORE MSB - ENRICHMENT FOR P/N -> IN GEAR
BFF3          bra     LC001                ;

BFF5    LBFF5:clr     L02B8                ;CLEAR PRETIMER
BFF8          bclr    *L0098,#0x10        ;CLEAR TIME MAXED OUT
BFFB          clr     L02B9                ;CLEAR TIMER
BFFE    LBFFE:clr     L02BA                ;CLEAR FUEL TERM

```

```

;-----
; DECAY FATI
;-----
C001 LC001:brclr    *L0097,#0x80,LC04B    ;BRANCH IF RUN-FUEL NOT ALLOWED, ELSE
C005          ldab    L017B              ;FATI DECAY DELAY VS CTS
C008          cmpb    *L002E              ;STARTUP TIMER
C00A          bcc     LC04B              ;BRANCH IF LOOKUP RESULT > TIMER, ELSE
C00C          ldab    L012B              ;LOAD FATI DECAY DELAY VS RPM
C00F          bne     LC047              ;BRANCH IF NOT Z, ELSE
C011          ldx     #0x88EB              ;FATI DECAY RATE VS STARTUP COOLANT
C014          ldaa    *L00F2              ;LOAD STARTUP COOLANT
C016          jsr     L99D7              ;2D LOOKUP
C019          ldab    *L002F              ;LOAD DECAY TERM
C01B          mul     ;LOOKUP RESULT * L002F
C01C          staa    *L002F              ;SAVE FATI DECAY TERM
C01E          ldaa    *L00AB              ;LOAD NLRPMX
C020          cmpa    #0xA0              ;4000 RPM
C022          bls     LC026              ;BRANCH IF <= 4000 RPM, ELSE
C024          ldaa    #0xA0              ;LOAD 4000 RPM
C026 LC026:ldx     #0x88CA              ;TIME DELAY BEFORE DECAYING FATI VS RPM
C029          brset   *L008E,#0x01,LC030 ;BRANCH IF P/N MODE, ELSE
C02D          ldx     #0x88D5              ;TIME DLY BEFORE DECAYING FATI VS RPM, P/N
C030 LC030:jsr     L99D7              ;2D LOOKUP
C033          tab     ;TRANSFER LOOKUP RESULT TO B REG
C034          ldx     #0x88E0              ;RPM TIME DELAY MULTIPLIER VS CTS
C037          ldaa    *L00A6              ;LOAD CLNT TEMP (DEFAULTED)
C039          jsr     L99CC              ;2D LOOKUP
C03C          mul     ;LOOKUP RESULT * PREV LOOKUP RESULT
C03D          lsld    ;MUL BY 2
C03E          bcs     LC043              ;BRANCH IF OVERFLOW, ELSE
C040          lsld    ;MUL BY 2
C041          bcc     LC045              ;BRANCH IF NO OVERFLOW, ELSE
C043 LC043:ldaa    #0xFF              ;LOAD 255
C045 LC045:tab     ;TRANSFER TO B
C046          incb    ;INCREMENT
C047 LC047:decb    ;DECREMENT TIME
C048          stab    L012B              ;STORE FATI DECAY DELAY TIME VS RPM

;-----
; CONDITIONS MET FOR POWER ENRICHMENT ?
;-----
C04B LC04B:ldx     #0x891D              ;PE MODE TPS THRESHOLD
C04E          brclr   *L0091,#0x02,LC056 ;BRANCH IF RPM NOT > 6375, ELSE
C052          ldaa    0x10,x              ;892D
C054          bra     LC05B              ;

C056 LC056:ldaa    *L00AB              ;LOAD NLRPMX
C058          jsr     L99D7              ;2D LOOKUP
C05B LC05B:brclr   *L009B,#0x20,LC065 ;BRANCH IF NOT IN PE MODE, ELSE
C05F          suba    L891C              ;TPS HYSTERESIS
C062          bcc     LC065              ;BRANCH IF NO UNDERFLOW, ELSE
C064          clra    ;
C065 LC065:cmpa    *L00AA              ;%TPS
C067          bhi     LC09F              ;BRANCH IF > %TPS, ELSE
C069          ldaa    L8910              ;00
C06C          beq     LC072              ;BRANCH IF Z, ELSE
C06E          brset   *L0086,#0x02,LC09F ;BRANCH HOT OPEN LOOP, ELSE
C072 LC072:ldd     L895B              ;0000
C075          cpd     L02A3              ;PE MODE DELAY TIMER
C079          bhi     LC07E              ;BRANCH IF > TIMER
C07B          std     L02A3              ;TIMER
C07E LC07E:ldx     L02A3              ;PE MODE DELAY TIMER
C081          beq     LC089              ;GO SET PE MODE

```

```

C083          dex                      ;DECREMENT DELAY TIMER
C084          stx      L02A3            ;STORE TIMER
C087          bra      LC0B4            ;GO CLEAR PE MODE

C089  LC089:bset    *L009B,#0x20        ;SET PE MODE
C08C          ldx      #0x894B          ;PE FAR MULTIPLIER VS RPM
C08F          ldaa     *L00AC            ;LOAD RPM/25
C091          ldab     #0x10             ;OFFSET (400 RPM)
C093          jsr      L99D3            ;2D LOOKUP
C096          ldab     L0118            ;LOAD POWER ENRICHMENT FAR
C099          ldx      #0x0000          ;START WITH ZERO
C09C          jmp      LC160            ;

C09F  LC09F:brclr   *L0011,#0x02,LC0AF  ;MINOR LOOP COUNTER 25 MSEC
C0A3          ldx      L02A3            ;POWER ENRICHMENT DELAY TIMER
C0A6          inx                      ;INCREMENT
C0A7          cpx      L895B            ;00
C0AA          bhi      LC0AF            ;BRANCH IF > CAL, ELSE
C0AC          stx      L02A3            ;STORE TIMER
C0AF  LC0AF:ldaa     *L00AC            ;LOAD RPM/25
C0B1          staa     L02EF            ;STORE PREV RPM
C0B4  LC0B4:bclr     *L009B,#0x20        ;CLEAR PE MODE
C0B7          clra                      ;CLEAR CLEAR PE TIMER
C0B8          staa     L0151            ;STORE TIMER
C0BB          ldab     L8969            ;111 - STOICH
C0BE          brclr    *L009C,#0x80,LC11E ;BRANCH IF NOT CLOSED LOOP, ELSE
C0C2          brclr    *L009C,#0x08,LC11B ;BRANCH IF O/L F/A WAS NOT RICHER THAN
                                           ; KCLRATIO, ELSE
C0C6          psha                     ;SAVE
C0C7          ldaa     *L00BD            ;FILTERED MPH
C0C9          brset    *L008E,#0x10,LC0D2 ;BRANCH IF NOT 5TH GEAR, ELSE
C0CD          cmpa     L8B80            ;118 MPH
C0D0          bra      LC0DE            ;

C0D2  LC0D2:brset    *L008E,#0x08,LC0DB  ;BRANCH IF NOT 4TH GEAR, ELSE
C0D6          cmpa     L8B81            ;115 MPH
C0D9          bra      LC0DE            ;

C0DB  LC0DB:cmpa     L8B82              ;113 MPH
C0DE  LC0DE:pula                     ;
C0DF          bcc      LC118            ;BRANCH IF > CAL, ELSE
C0E1          cpd      *L00B5            ;FUEL AIR RATIO
C0E4          beq      LC118            ;BRANCH IF AFR = 14.7, ELSE
C0E6          bcs      LC100            ;BRANCH IF AFR > 14.7, ELSE
C0E8          ldd      *L00B5            ;FUEL AIR RATIO
C0EA          psha                     ;SAVE MSB ON STACK
C0EB          inc      L018B            ;INCREMENT
C0EE          ldaa     L018B            ;LOAD
C0F1          cmpa     L896B            ;16
C0F4          pula                     ;RESTORE MSB
C0F5          bls      LC115            ;BRANCH IF L018B <= 16, ELSE
C0F7          clr      L018B            ;CLEAR
C0FA          addd     #0x0001          ;INCREASE FAR BY 1
C0FD          jmp      LC199            ;GO STORE FAR

C100  LC100:ldd      *L00B5            ;FUEL AIR RATIO
C102          psha                     ;SAVE MSB
C103          inc      L018C            ;INCREMENT
C106          ldaa     L018C            ;LOAD
C109          cmpa     L896A            ;16
C10C          pula                     ;RESTORE MSB
C10D          bls      LC115            ;BRANCH IF L018C <= 16, ELSE
C10F          clr      L018C            ;CLEAR
C112          subd     #0x0001          ;DECREASE FAR BY 1

```

| | | | |
|------|--------------|----------------------|---|
| C115 | LC115: jmp | LC199 | ;GO STORE FAR |
| C118 | LC118: bclr | *L009C, #0x08 | ;CLEAR O/L F/A WAS RICHER THAN KCLRATIO |
| C11B | LC11B: jmp | LC199 | ;GO STORE FAR |
| C11E | LC11E: ldaa | L8747 | ;00 |
| C121 | beq | LC127 | ;BRANCH IF Z, ELSE |
| C123 | brclr | *L009A, #0x01, LC131 | ;BRANCH IF NEG DELTA RPM NOT > 50 RPM, ELSE |
| C127 | LC127: ldaa | *L00AB | ;LOAD NLRPMX |
| C129 | ldab | *L00BA | ;LOAD LV8 |
| C12B | ldx | #0x874B | ;OPEN LOOP AFR MULTIPLIER VS RPM AND LV8 |
| C12E | jsr | L995A | ;3D LOOKUP |
| C131 | LC131: brclr | *L0086, #0x02, LC13D | ;BRANCH IF NOT HOT OPEN LOOP, ELSE |
| C135 | ldab | L8912 | ;156 |
| C138 | beq | LC146 | ;BRANCH IF Z, ELSE |
| C13A | clra | | ;CLEAR A |
| C13B | bra | LC167 | ; |
| C13D | LC13D: brclr | *L0099, #0x01, LC157 | ; |
| C141 | ldaa | L891B | ;149 |
| C144 | bra | LC157 | ; |
| C146 | LC146: cmpa | L8969 | ;STOICH AFR (14.76) |
| C149 | bcc | LC14E | ;BRANCH IF > CAL, ELSE |
| C14B | ldaa | L8969 | ;LOAD STOICH |
| C14E | LC14E: ldab | L8911 | ;160 |
| C151 | mul | | ;160 * TABLE LOOKUP RESULT |
| C152 | lsld | | ;MUL BY 2 |
| C153 | bcc | LC157 | ;BRANCH IF NO OVERFLOW, ELSE |
| C155 | ldaa | #0xFF | ;LOAD 255 |
| C157 | LC157: ldx | #0x0000 | ;START WITH ZERO |
| C15A | ldab | L0116 | ;FATC - CLNT TEMP AFR |
| C15D | abx | | ;ADD TO X |
| C15E | ldab | *L002C | ;FATI - TIMEOUT AFR |
| C160 | LC160: abx | | ;ADD TO X |
| C161 | pshx | | ;SAVE FAR |
| C162 | tsx | | ;X POINTS TO STACK |
| C163 | jsr | L989D | ;8 x 16 MULTIPLY, PRODUCT ON STACK |
| C166 | pulx | | ;RESTORE |
| C167 | LC167: bclr | *L009C, #0x08 | ;CLEAR O/L F/A WAS RICHER THAN KCLRATIO |
| C16A | brset | *L009B, #0x20, LC179 | ;BRANCH IF PE MODE, ELSE |
| C16E | tsta | | ;TEST MSB |
| C16F | bne | LC176 | ;BRANCH IF NOT = ZERO, ELSE |
| C171 | cmpb | L8969 | ;FAR AND STOICH FAR |
| C174 | beq | LC179 | ;BRANCH IF FATI = STOICH, ELSE |
| C176 | LC176: bset | *L009C, #0x08 | ;SET O/L F/A WAS RICHER THAN KCLRATIO |
| C179 | LC179: cpd | L8748 | ;00 |
| C17D | bcc | LC199 | ;BRANCH IF FAR > 0, ELSE |
| C17F | cpd | *L00B5 | ;FUEL AIR RATIO |
| C182 | bcc | LC199 | ;BRANCH IF > FAR, ELSE |
| C184 | ldd | *L00B5 | ;FUEL AIR RATIO |
| C186 | psha | | ;SAVE MSB ON STACK |
| C187 | inc | L018A | ;INCREMENT TIMER |
| C18A | ldaa | L018A | ;LOAD TIMER |
| C18D | cmpa | L874A | ;00 |
| C190 | pula | | ;GET BACK MSB |
| C191 | bls | LC199 | ;BRANCH IF TIMER <= CAL, ELSE |
| C193 | clr | L018A | ;CLEAR TIMER |
| C196 | subd | #0x0001 | ;DECREMENT BY 1 |
| C199 | LC199: std | *L00B5 | ;STORE FUEL AIR RATIO |
| C19B | ldab | *L00BD | ;FILTERED MPH |
| C19D | brset | *L008E, #0x10, LC1A6 | ;BRANCH IF NOT 5 TH GEAR, ELSE |
| C1A1 | subb | L8B80 | ; |
| C1A4 | bra | LC1B2 | ; |

```

C1A6    LC1A6:brset    *L008E,#0x08,LC1AF    ;BRANCH IF NOT 4TH GEAR, ELSE
C1AA            subb    L8B81                ;
C1AD            bra     LC1B2                ;

C1AF    LC1AF:subb     L8B82                ;
C1B2    LC1B2:bcsl     LC1C8                ;BRANCH IF MPH < CAL, ELSE
C1B4            ldaa    L8B83                ;HIGH SPEED AFR MULTIPLIER
C1B7            mul     ;A * DELTA MPH
C1B8            tsta    ;TEST MSB
C1B9            bne     LC1C0                ;BRANCH IF NOT Z, ELSE
C1BB            ldaa    #0x80                ;LOAD 128
C1BD            aba     ;ADD TO SPEED
C1BE            bcc     LC1C2                ;BRANCH IF NO OVERFLOW, ELSE
C1C0    LC1C0:ldaa     #0xFF                ;LOAD 255
C1C2    LC1C2:ldx      #0x00B5              ;LOAD FUEL AIR RATIO
C1C5            jsr     L989D                ;8 X 16 MULTIPLY, PRODUCT SAVED IN L00B5
C1C8    LC1C8:brclr    *L0095,#0x82,LC1CF    ;BRANCH IF HUD DEVICE NOT CONNECTED, ELSE
C1CC            jsr     L5000                ;UPDATE HUD WITH FAR
C1CF    LC1CF:brclr    *L0084,#0x80,LC1ED    ;BRANCH IF NOT MODE 4, ELSE
C1D3            ldaa    L0234                ;LOAD CONTROL BYTE
C1D6            anda    #0x04                ;BIT 2?
C1D8            beq     LC1ED                ;BRANCH IF BIT 2 CLEAR, ELSE
C1DA            ldab    L0236                ;LOAD COMMANDED AFR*10
C1DD            stab    L046A                ;STORE AFR
C1E0            ldx     #0x0000              ;START WITH ZERO
C1E3            abx     ;ADD 4 TIMES
C1E4            abx     ;
C1E5            abx     ;
C1E6            abx     ;
C1E7            ldd     #0xFFFF              ;
C1EA            idiv    ;FFFF/X
C1EB            stx     *L00B5                ;STORE QUOTIENT AS FAR
C1ED    LC1ED:brclr    *L002A,#0x10,LC1F4    ;EGR BIT STATUS FLAG
C1F1            jmp     LC342                ;SKIP BLM UPDATE

;-----
; CONDITIONS MET FOR IDLE ?
;-----

C1F4    LC1F4:ldab     *L00AE                ;BLM CELL
C1F6            cmpb    #0x10                ;CELL 16 ?
C1F8            beq     LC1FD                ;BRANCH IF CELL 16, ELSE
C1FA            stab    L01AA                ;SAVE ? PREV BLM CELL
C1FD    LC1FD:ldx      #0x8A14                ;INDEX
C200            brclr    *L008F,#0xC0,LC205    ;BRANCH IF COND NOT MET FOR IDLE FUEL
C204            inx     ;8A15
C205    LC205:ldaa     *L00AC                ;LOAD RPM/25
C207            cmpa    0x00,x                ;8A14 OR
                                           ;8A15
C209            bhi     LC236                ;BRANCH IF > CAL, ELSE
C20B            ldaa    *L00BD                ;FILTERED MPH
C20D            cmpa    0x02,x                ;8A16 OR
                                           ;8A17
C20F            bhi     LC236                ;BRANCH IF > CAL, ELSE
C211            ldaa    *L00AA                ;%TPS
C213            cmpa    0x04,x                ;8A18 OR
                                           ;8A19
C215            bhi     LC236                ;BRANCH IF > CAL, ELSE
C217            ldaa    *L00CD                ;MAF (GMS/SEC)
C219            cmpa    0x06,x                ;8A1A OR
                                           ;8A1B
C21B            bhi     LC236                ;BRANCH IF MAF > CAL, ELSE
C21D            bset    *L008F,#0xC0          ;SET IDLE FUEL

```

```

;-----
; BLM CORRECTION
;-----

C220          ldaa    L801B                ;OPTION WORD (TIS SET - CELL 16 IS IDLE)
C223          beq     LC239                ;BRANCH IF Z, ELSE
C225          bclr    *L009B,#0x04        ;CLEAR BIT 2 - BLM CELL CHANGE
C228          ldab     *L00AE              ;BLM CELL
C22A          cmpb    #0x10                ;CELL 16 ?
C22C          beq     LC233                ;BRANCH IF CELL 16, ELSE
C22E          bset     *L009B,#0x0C        ;SET BITS 2 AND 3
                                           ;BLM ADDRESS CHANGE
                                           ;DELAY BLM UPDATE
                                           ;16

C231          ldab     #0x10                ;16
C233  LC233: jmp     LC2D0                ;

C236  LC236: bclr    *L008F,#0xC0          ;SET COND NOT MET FOR IDLE FUEL
C239  LC239: ldab     *L00AE              ;BLM CELL
C23B          cmpb    #0x10                ;16
C23D          bne     LC244                ;BRANCH IF NOT CELL 16, ELSE
C23F          ldab     L01AA                ;? PREV BLM CELL
C242          stab     *L00AE              ;STORE BLM CELL
C244  LC244: bclr    *L009B,#0x04          ;CLEAR BLM CELL CHANGE
C247          ldx      #0x8A1F              ;INDEX
C24A          ldab     *L00AE              ;BLM CELL
C24C          andb     #0x03                ;RPM BITS
C24E          beq     LC260                ;IF RPM INDEX = 0, THERE IS NO LOWER
                                           ; BOUNDRY TO CHECK
                                           ;ADD TO ADDRESS

C250          abx                                           ;GET CURRENT RPM BOUND
C251          ldaa     0x00,x                ;125 RPM - HYSTERESIS
C253          suba     L8A29                ;BRANCH IF UNDERFLOW, ELSE
C256          bcs     LC25C                ;COMPARE RPM/25
C258          cmpa     *L00AC              ;BRANCH IF RPM < LIMIT, ELSE
C25A          bhi     LC2A1                ;
C25C  LC25C: cmpb     #0x03                ;
C25E          beq     LC26B                ;
C260  LC260: ldaa     0x01,x                ;8A20
C262          adda     L8A29                ;125 RPM - HYSTERESIS
C265          bcs     LC26B                ;BRANCH IF OVERFLOW, ELSE
C267          cmpa     *L00AC              ;LOAD RPM/25
C269          bcs     LC2A1                ;BRANCH IF > EDGE + HYSTERESIS, ELSE
C26B  LC26B: ldab     *L00AE              ;BLM CELL
C26D          ldx      #0x8A21              ;INDEX
C270          andb     #0x0C                ;AIR BITS
C272          lsr     b                ;DIV BY 2
C273          lsr     b                ;DIV BY 2
C274          beq     LC291                ;BRANCH IF = ZERO, ELSE
C276          abx                                           ;ADD TO ADDRESS
C277          abx                                           ;ADD TO ADDRESS
C278          pshb                                           ;SAVE B
C279          ldd      0x00,x                ;GET A VALUE FROM THE TABLE
C27B          subd     L8A2A                ;SUBTRACT HYSTERESIS
C27E          pulb                                           ;GET OFFSET FROM STACK
C27F          bcs     LC28D                ;
C281          pshb                                           ;SAVE OFFSET ON STACK
C282          ldd      0x00,x                ;GET A VALUE FROM THE TABLE
C284          subd     L8A2A                ;SUBTRACT MAF EDGE TO EDGE HYSTERESIS
C287          cpd      *L00CD              ;COMPARE MAF (GMS/SEC)
C28A          pulb                                           ;
C28B          bhi     LC2A1                ;BRANCH IF > MAF, ELSE
C28D  LC28D: cmpb     #0x03                ;
C28F          beq     LC29D                ;BRANCH IF 3, ELSE
C291  LC291: ldd      0x02,x                ;
C293          addd     L8A2A                ;ADD MAF HYSTERESIS
C296          bcs     LC29D                ;BRANCH IF OVERFLOWED, ELSE

```

| | | | |
|------|-------------|--------------------|---|
| C298 | cpd | *L00CD | ;COMPARE MAF (GMS/SEC) |
| C29B | bcs | LC2A1 | ;BRANCH IF MAF > LIMIT (CELL HAS CHANGED) |
| C29D | LC29D:ldab | *L00AE | ;BLM CELL |
| C29F | bra | LC2D0 | ; |
| C2A1 | LC2A1:clrb | | ;CLEAR CELL POINTER |
| C2A2 | bset | *L009B,#0x0C | ;B2 AND B3 - BLM CELL CHANGE |
| | | | ; & DELAY BLM UPDATE |
| C2A5 | ldaa | *L00AC | ;LOAD RPM/25 |
| C2A7 | cmpa | L8A20 | ;COMPARE LOWER BOUNDRY |
| C2AA | bcs | LC2B9 | ;BRANCH IF < LOWER BOUNDRY, ELSE |
| C2AC | incb | | ;INCREMENT B |
| C2AD | cmpa | L8A21 | ;COMPARE MID BOUNDRY |
| C2B0 | bcs | LC2B9 | ;BRANCH IF < MID BOUNDRY, ELSE |
| C2B2 | incb | | ;INCREMENT B |
| C2B3 | cmpa | L8A22 | ;COMPARE UPPER BOUNDRY |
| C2B6 | bcs | LC2B9 | ;BRANCH IF < UPPER BOUNDRY, ELSE |
| C2B8 | incb | | ;INCREMENT B |
| C2B9 | LC2B9:ldx | *L00CD | ;LOAD MAF (GM/SEC) |
| C2BB | cpx | L8A23 | ;LOWER MAF BOUNDRY |
| C2BE | bcs | LC2D0 | ;BRANCH IF < LOWER BOUNDRY, ELSE |
| C2C0 | addb | #0x04 | ;ADD 4 |
| C2C2 | cpx | L8A25 | ;MID BOUNDRY |
| C2C5 | bcs | LC2D0 | ;BRANCH IF < MID BOUNDRY, ELSE |
| C2C7 | addb | #0x04 | ;ADD 4 |
| C2C9 | cpx | L8A27 | ;UPPER BOUNDRY |
| C2CC | bcs | LC2D0 | ;BRANCH IF < UPPER BOUNDRY, ELSE |
| C2CE | addb | #0x04 | ;ADD 4 |
| C2D0 | LC2D0:ldx | #0x0034 | ;INDEX BLM CELL ADDRESSES |
| C2D3 | abx | | ;ADD CELL POINTER |
| C2D4 | brset | *L009C,#0x80,LC2E0 | ;BRANCH IF CLOSED LOOP, ELSE |
| C2D8 | ldaa | 0x00,x | ;GET CURRENT CELL VALUE |
| C2DA | cmpa | 0x11,x | ;COMPARE BLM WITH INACTIVE CCP |
| C2DC | bcc | LC2EC | ;BRANCH IF > BLM WITH INACTIVE CCP, ELSE |
| C2DE | bra | LC2E4 | ; |
| C2E0 | LC2E0:ldaa | *L00F4 | ;CCP DUTY CYCLE |
| C2E2 | bne | LC2EC | ;BRANCH IF CCP DC NOT Z, ELSE |
| C2E4 | LC2E4:bclr | *L008F,#0x80 | ;CLEAR CCP DC > ZERO |
| C2E7 | tba | | ;TRANSFER CELL POINTER TO A REG |
| C2E8 | ldab | #0x11 | ;OFFSET - BLMS WITH ACTIVE CCP |
| C2EA | abx | | ;ADD TO ADDRESS |
| C2EB | tab | | ;COPY TO B REG |
| C2EC | LC2EC:ldaa | 0x00,x | ;GET CURRENT BLM |
| C2EE | cmpa | L8A2F | ;150d |
| C2F1 | bls | LC2FC | ;BRANCH IF < MAX, ELSE |
| C2F3 | brclr | *L0015,#0x10,LC329 | ;BRANCH IF NO MAF SENSOR MALF, ELSE |
| C2F7 | cmpa | L8A30 | ;MAX BLM VALUE |
| C2FA | bhi | LC329 | ;BRANCH IF > MAX BLM, ELSE |
| C2FC | LC2FC:tstb | | ;TEST POINTER |
| C2FD | beq | LC303 | ;BRANCH IF Z, ELSE |
| C2FF | cmpb | #0x10 | ;16 |
| C301 | bne | LC31B | ;BRANCH IF NOT CELL 16, ELSE |
| C303 | LC303:brset | *L008F,#0x80,LC319 | ;BRANCH IF CCP DC > Z, ELSE |
| C307 | ldy | #0x8A31 | ;INDEX |
| C30B | brclr | *L0015,#0x10,LC311 | ;BRANCH IF NO MAF SENSOR MALF, ELSE |
| C30F | iny | | ;8A32 |
| C311 | LC311:cmpa | 0x00,y | ;8A31 OR |
| | | | ;8A32 |
| C314 | bcc | LC319 | ; |
| C316 | ldaa | 0x00,y | ;8A31 OR |
| | | | ;8A32 |
| C319 | LC319:bra | LC32F | ; |

```

C31B    LC31B:cmpa    L8A31                ;100
C31E          bcc     LC32F                ;BRANCH IF BLM > MIN, ELSE
C320          brclr   *L0015,#0x10,LC329   ;BRANCH IF NO MAF SENSOR MALF, ELSE
C324          cmpa    L8A32                ;90 - MIN BLM VALUE
C327          bcc     LC32F                ;BRANCH IF > MIN, ELSE
C329    LC329:bset    *L009A,#0x40         ;SET ? BLMS RESET
C32C          jsr     LF7E1                ;GO RESET BLMS
C32F    LC32F:stab    *L00AE               ;BLM CELL
C331          ldab    L8A37                ;FILTER COEFF FOR BLM
C334          beq     LC340                ;BRANCH IF Z, ELSE
C336          ldx     #0x00AF              ;OLD FILTERED BLM
C339          ldy     #0x8A37              ;BLM FILTER COEFFICIENT ADDRESS
C33D          jsr     L99F4                ;LAG FILTER
C340    LC340:std     *L00AF               ;STORE FILTERED BLM

;-----
;  PROP/INT LOGIC
;-----
C342    LC342:ldaa    #0x60                ;SEL CH #6 - O2 SENSOR
C344          jsr     L9A18                ;A/D READ
C347          staa    *L00A9               ;MINOR LOOP O2 A/D COUNTS (mV)
C349          ldd     L895D                ;O2 A/D COUNTS LIMITS
C34C          cmpa    *L00A9               ;MINOR LOOP O2 A/D COUNTS (mV)
C34E          bcs     LC354                ;BRANCH IF > HIGH THRESHOLD, ELSE
C350          cmpb    *L00A9               ;MINOR LOOP O2 A/D COUNTS (mV)
C352          bls     LC35A                ;BRANCH IF > LOW THRESHOLD, ELSE
C354    LC354:clr     L0113                ;CLEAR O2 SENSOR READY TIMER
C357          bset    *L0012,#0x01         ;SET O2 SENSOR READY
C35A    LC35A:ldab    *L009C               ;MWFA1
C35C          orab    #0x40                ;SET BIT RICH BIT
C35E          ldx     #0x8970              ;INDEX
C361          ldaa    *L00A6               ;LOAD CLNT TEMP (DEFAULTED)
C363          cmpa    L8A00                ;45 DEG C
C366          bcs     LC377                ;BRANCH IF < 45 DEG C, ELSE
C368          ldy     *L0032               ;RUN TIME
C36B          cpy     L896E                ;00
C36F          bcs     LC377                ;BRANCH IF < 0, ELSE
C371          brclr   *L008F,#0xC0,LC383   ;BRANCH IF IDLE COND NOT MET, ELSE
C375          inx                      ;8971
C376          inx                      ;8972
C377    LC377:ldaa    *L00A9               ;MINOR LOOP O2 A/D COUNTS (mV)
C379          cmpa    0x00,x              ;8970 OR
C37B          bhi     LC3CA                ;8972
C37D          cmpa    0x01,x              ;BRANCH IF O2 A/D COUNTS > CAL, ELSE
C37F          bcs     LC3C6                ;8971 OR
C381          bra     LC3C1                ;8973
C383    LC383:ldaa    L8998                ;BRANCH IF O2 A/D COUNTS < CAL, ELSE
C386          bne     LC395                ;
C388          pshb                      ;
C389          ldab    *L00AE               ;OPTION WORD
C38B          ldx     #0x89AA              ;BRANCH IF NOT Z, ELSE
C38E          abx                      ;SAVE MODE WORD ON STACK
C38F          ldab    0x00,x              ;BLM CELL
C391          tba                      ;TABLE ADDRESS
C392          pulb                      ;ADD TO ADDRESS
C393          bra     LC39D                ;GET A VALUE FROM THE TABLE
C395    LC395:ldaa    *L00BA               ;
C397          ldx     #0x89AA              ;GET BACK MODE WORD
C39A          jsr     L99D7                ;
C39D    LC39D:cmpa    *L00A9               ;
C395          ;LOAD LV8
C397          ;
C39A          ;2D LOOKUP
C39D          ;MINOR LOOP O2 A/D COUNTS (mV)

```

| | | | |
|------|-------------|--------------------|--|
| C39F | bc | LC3C8 | ;BRANCH IF O2 A/D COUNTS > LOOUKP, ELSE |
| C3A1 | ldaa | L8998 | ;OPTION WORD |
| C3A4 | bne | LC3B3 | ;BRANCH IF NOT Z, ELSE |
| C3A6 | pshb | | ;SAVE MODE WORD ON STACK |
| C3A7 | ldab | *L00AE | ;BLM CELL |
| C3A9 | ldx | #0x8999 | ;O2 LEAN THRESHOLD VS LV8 OR BLM CELL |
| C3AC | abx | | ;ADD TO INDEX |
| C3AD | ldab | 0x00,x | ;GET A VALUE FROM THE TABLE |
| C3AF | tba | | ;COPY TO A REG |
| C3B0 | pulb | | ;GET BACK STATUS FLAG |
| C3B1 | bra | LC3BB | ; |
| C3B3 | LC3B3:ldaa | *L00BA | ;LOAD LV8 |
| C3B5 | ldx | #0x8999 | ;O2 LEAN THRESHOLD VS LV8 OR BLM CELL |
| C3B8 | jsr | L99D7 | ;2D LOOKUP |
| C3BB | LC3BB:cmpa | *L00A9 | ;MINOR LOOP O2 A/D COUNTS (mV) |
| C3BD | bhi | LC3C6 | ;BRANCH IF LOOKUP RESULT > O2 A/D, ELSE |
| C3BF | ldaa | *L00A9 | ;MINOR LOOP O2 A/D COUNTS (mV) |
| C3C1 | LC3C1:cmpa | L010B | ;PREV O2 A/D COUNTS |
| C3C4 | bhi | LC3CA | ;BRANCH IF PREV > CURRENT, ELSE |
| C3C6 | LC3C6:andb | #0xBF | ;SET LEAN BIT (CLEAR) |
| C3C8 | LC3C8:ldaa | *L00A9 | ;MINOR LOOP O2 A/D COUNTS (mV) |
| C3CA | LC3CA:staa | L010B | ;PREV O2 A/D COUNTS |
| C3CD | cmpb | *L009C | ;WAS THERE A RICH/LEAN SWITCH? |
| C3CF | beq | LC3D7 | ;BRANCH IF NOT, ELSE |
| C3D1 | inc | L029E | ;O2 XCOUNTS |
| C3D4 | clr | L0114 | ;CLEAR PROPORTIONAL COUNTER |
| C3D7 | LC3D7:stab | *L009C | ;STORE MWFAL |
| C3D9 | bpl | LC40E | ;BRANCH IF LEAN, ELSE |
| C3DB | clra | | ; |
| C3DC | ldab | L8969 | ;STOICH |
| C3DF | cpd | *L00B5 | ;FUEL AIR RATIO |
| C3E2 | bne | LC40E | ;BRANCH IF FAR NOT STOICH, ELSE |
| C3E4 | brclr | *L009B,#0x22,LC3EA | ;BRANCH IF NOT DFCO AND PE MODE |
| C3E8 | bra | LC40E | ;RESET INTEGRATOR |
| C3EA | LC3EA:ldab | *L00B3 | ;LOAD INTEGRATOR |
| C3EC | cmpb | L8982 | ;114d |
| C3EF | bcc | LC3F5 | ;BRANCH IF >= CAL, ELSE |
| C3F1 | brset | *L009B,#0x04,LC40E | ;BRANCH IF BLM ADDRESS CHANGE |
| C3F5 | LC3F5:brset | *L009C,#0x20,LC40E | ;BRANCH IF FORCE OL FOR LOW PW SET, ELSE |
| C3F9 | ldab | L86C9 | ;OPTION WORD |
| C3FC | bitb | #0x02 | ;BIT 1 SET ? (TIS CLEAR) |
| C3FE | beq | LC405 | ;BRANCH IF BIT 1 CLEAR, ELSE |
| C400 | ldaa | L0159 | ;LOAD ACCEL ENRICHMENT TERM |
| C403 | bne | LC40E | ;BRANCH IF NOT Z, ELSE |
| C405 | LC405:bitb | #0x04 | ;BIT 2 SET ? (TIS SET) |
| C407 | beq | LC418 | ;BRANCH IF BIT 2 CLEAR, ELSE |
| C409 | ldaa | L02A1 | ;DE TERM |
| C40C | beq | LC418 | ;BRANCH IF Z, ELSE |
| C40E | LC40E:ldaa | #0x80 | ;LOAD 128 - RESET INTEGRATOR |
| C410 | staa | *L00B3 | ;STORE INTEGRATOR |
| C412 | bset | *L0091,#0x04 | ;SET INTEGRATOR RESET |
| C415 | jmp | LC5DE | ;GO SAVE C.L. FINE CORRECTION |
| C418 | LC418:ldd | *L00CD | ;MAF (GMS/SEC) |
| C41A | lsld | | ;MUL BY 2 |
| C41B | bcs | LC41F | ;BRANCH IF OVERFLOW, ELSE |
| C41D | bpl | LC421 | ;BRANCH IF < 127, ELSE |
| C41F | LC41F:ldaa | #0x7F | ;LOAD 127 |
| C421 | LC421:psha | | ;SAVE ON STACK |
| C422 | brclr | *L008F,#0xC0,LC432 | ;BRANCH IF NOT IDLE FUEL, ELSE |
| C426 | ldaa | L8A12 | ;INT UPDATE DELAY AT IDLE |
| C429 | brclr | *L008F,#0x80,LC438 | ;BRANCH IF CCP DC NOT > Z, ELSE |

| | | | |
|------|-------------|--------------------|--|
| C42D | ldaa | L8A13 | ;INT UPDATE DELAY AT IDLE WITH ACTIVE CCP |
| C430 | bra | LC438 | ;GO STORE INT UPDATE DELAY |
| C432 | LC432:ldx | #0x8983 | ;INT UPDATE DELAY VS MAF |
| C435 | jsr | L99D7 | ;2D LOOKUP |
| C438 | LC438:staa | *L00B2 | ;STORE INT UPDATE DELAY |
| C43A | lsra | | ;DIV BY 4 |
| C43B | lsra | | |
| C43C | staa | *L00B1 | ;STORE INT UPDATE DELAY |
| C43E | pula | | ;GET MAF FROM STACK |
| C43F | brclr | *L009A,#0x02,LC448 | ;BRANCH IF NOT DECEL ENLEANMENT, ELSE |
| C443 | ldd | L8996 | ;O2 FILTER NOM VALUE WHEN RESETING INT |
| C446 | bra | LC464 | ;GO STORE FILTERED O2 |
| C448 | LC448:ldx | #0x898C | ;O2 FILTER COEFF VS AIRFLOW |
| C44B | jsr | L99D7 | ;2D LOOKUP |
| C44E | psha | | ;SAVE TABLE LOOKUP RESULT |
| C44F | ldaa | *L00A9 | ;MINOR LOOP O2 A/D COUNTS (mV) |
| C451 | ldx | #0x010C | ;OLD FILTERED O2 A/D COUNTS |
| C454 | tsy | | ;TABLE LOOKUP RESULT IS FILTER COEFFICIENT |
| C456 | jsr | L99F4 | ;LAG FILTER |
| C459 | ins | | ;IGNORE TABLE LOOKUP RESULT |
| C45A | ldx | #0x010E | ;OLD FILTERED O2 |
| C45D | ldy | #0x8995 | ;FILTER COEFFICIENT |
| C461 | jsr | L99F5 | ;LAG FILTER |
| C464 | LC464:std | L010E | ;STORE FILTERED O2 A/D COUNTS |
| C467 | inc | L0114 | ;PROPORTIONAL COUNTER |
| C46A | bclr | *L0091,#0x04 | ;CLEAR BIT 2 |
| C46D | ldab | L0114 | ;LOAD PROPORTIONAL COUNTER |
| C470 | cmpb | *L00B2 | ;INT UPDATE DELAY VS MAF |
| C472 | bhi | LC477 | ;BRANCH IF PROP COUNTER > CAL, ELSE |
| C474 | jmp | LC4FA | |
| C477 | LC477:clr | L0114 | ;PROPORTIONAL COUNTER |
| C47A | ldab | *L00B3 | ;LOAD INTEGRATOR |
| C47C | ldx | #0x8978 | ;INDEX |
| C47F | brclr | *L008F,#0xC0,LC489 | ;BRANCH IF IDLE CONDITIONS NOT MET, ELSE |
| C483 | brclr | *L009C,#0x40,LC4D1 | ;BRANCH IF LEAN, ELSE |
| C487 | bra | LC4A6 | |
| C489 | LC489:inx | | |
| C48A | inx | | ;897A |
| C48B | brclr | *L009C,#0x40,LC4BA | ;BRANCH IF LEAN, ELSE |
| C48F | bclr | *L009D,#0x0C | ;CLEAR BITS 2 AND 3 |
| C492 | brclr | *L009D,#0x01,LC4A3 | ;BRANCH IF NOT BIT 0, ELSE |
| C496 | inx | | |
| C497 | inx | | ;897C |
| C498 | brclr | *L009D,#0x02,LC4A0 | ;BRANCH IF NOT BIT 1, ELSE |
| C49C | inx | | |
| C49D | inx | | ;897E |
| C49E | bra | LC4A6 | |
| C4A0 | LC4A0:bset | *L009D,#0x02 | ;SET BIT 1 |
| C4A3 | LC4A3:bset | *L009D,#0x01 | ;SET BIT 0 |
| C4A6 | LC4A6:subb | 0x00,x | ;8978 OR |
| | | | ;897A OR |
| | | | ;897C OR |
| | | | ;897E |
| C4A8 | bcc | LC4AB | ;BRANCH IF NO UNDERFLOW, ELSE |
| C4AA | clrb | | |
| C4AB | LC4AB:brclr | *L009A,#0x02,LC4E6 | ;BRANCH IF NOT DECEL ENLEANMENT, ELSE |
| C4AF | ldaa | L0100 | ;A.E. TIMER, FROM XIRQ ROUTINE |
| C4B2 | cmpa | L8AD2 | ;00 |
| C4B5 | bhi | LC4E6 | ;BRANCH IF > CAL, ELSE |

| | | | |
|------|-------------|--------------------|--|
| C4B7 | jmp | LC5E1 | ;SKIP PROP/INTEGRATOR UPDATE |
| C4BA | LC4BA:bclr | *L009D,#0x03 | ;CLEAR BITS 0 AND 1 |
| C4BD | brclr | *L009D,#0x04,LC4CE | ;BRANCH IF NOT BIT 2, ELSE |
| C4C1 | inx | | ;897A |
| C4C2 | inx | | ;897B |
| C4C3 | brclr | *L009D,#0x08,LC4CB | ;BRANCH IF NOT BIT 3, ELSE |
| C4C7 | inx | | ;897C |
| C4C8 | inx | | ;897D |
| C4C9 | bra | LC4D1 | ; |
| C4CB | LC4CB:bset | *L009D,#0x08 | ;SET BIT 3 |
| C4CE | LC4CE:bset | *L009D,#0x04 | ;SET BIT 2 |
| C4D1 | LC4D1:adbb | 0x01,x | ;8979 OR |
| | | | ;897B OR |
| | | | ;897D OR |
| | | | ;897F |
| C4D3 | bcc | LC4D7 | ;BRANCH IF NO OVERFLOW, ELSE |
| C4D5 | ldab | #0xFF | ;LOAD 255 |
| C4D7 | LC4D7:brclr | *L009A,#0x04,LC4E6 | ;BRANCH IF DE-AE BLEND NOT ENABLED, ELSE |
| C4DB | ldaa | L0101 | ;DE TIMER, GETS INCREMENTED IN XIRQ |
| C4DE | cmpa | L8B3B | ;0x5A |
| C4E1 | bhi | LC4E6 | ;BRANCH IF > CAL, ELSE |
| C4E3 | jmp | LC5E1 | ;SKIP PROP/INT UPDATE |
| C4E6 | LC4E6:cmpb | L8981 | ;MAX INTEGRATOR |
| C4E9 | bis | LC4F0 | ;BRANCH IF <= MAX, ELSE |
| C4EB | ldab | L8981 | ;LOAD MAX INTEGRATOR |
| C4EE | bra | LC4F8 | ;GO STORE INTEGRATOR |
| C4F0 | LC4F0:cmpb | L8980 | ;MIN INTEGRATOR 100d |
| C4F3 | bcc | LC4F8 | ;BRANCH IF >= 8980, ELSE |
| C4F5 | ldab | L8980 | ;LOAD MIN INTEGRATOR |
| C4F8 | LC4F8:stab | *L00B3 | ;STORE INTEGRATOR |
| C4FA | LC4FA:ldaa | *L00B2 | ;INT UPDATE DELAY VS MAF |
| C4FC | ldab | L896C | ;GAIN FACTOR FOR INTDLY |
| C4FF | brclr | *L008F,#0xC0,LC506 | ;BRANCH IF NOT IDLE FUEL, ELSE |
| C503 | ldab | L896D | ;GAIN FACTOR FOR INTDLY AT IDLE |
| C506 | LC506:mul | | ;A * GAIN FACTOR |
| C507 | lsld | | ;MUL BY 2 |
| C508 | bcc | LC50C | ;BRANCH IF NO OVERFLOW, ELSE |
| C50A | ldaa | #0xFF | ;LOAD 255 |
| C50C | LC50C:cmpa | L0115 | ;INTEGRATOR UPDATE TIMER |
| C50F | bis | LC518 | ;BRANCH IF TIMER >= CAL, ELSE |
| C511 | ldab | L0115 | ;INTEGRATOR UPDATE TIMER |
| C514 | incb | | ;INCREMENT TIMER |
| C515 | jmp | LC59D | ;GO STORE INTEGRATOR UPDATE TIMER |
| C518 | LC518:ldab | *L00B3 | ;LOAD INTEGRATOR |
| C51A | ldx | #0x8974 | ;INDEX |
| C51D | ldaa | *L00A6 | ;LOAD CLNT TEMP (DEFAULTED) |
| C51F | cmpa | L8A00 | ; |
| C522 | bcs | LC533 | ;BRANCH IF < CAL, ELSE |
| C524 | ldy | *L0032 | ;RUN TIME |
| C527 | cpy | L896E | ;0000 |
| C52B | bcs | LC533 | ;BRANCH IF < CAL, ELSE |
| C52D | brclr | *L008F,#0xC0,LC540 | ;BRANCH IF NOT IDLE FUEL, ELSE |
| C531 | inx | | ; |
| C532 | inx | | ;8976 |
| C533 | LC533:ldaa | L010E | ;FILTERED O2 VOLTS |
| C536 | cmpa | 0x00,x | ;8974 OR |
| | | | ;8976 |
| C538 | bhi | LC583 | ;BRANCH IF O2 VOLTS > CAL, ELSE |
| C53A | cmpa | 0x01,x | ;8975 OR |

| | | | | |
|------|--------|-------|--------------------|--|
| C53C | | bcs | LC57E | ;8977 |
| C53E | | bra | LC59C | ;BRANCH IF O2 VOLTS < CAL, ELSE |
| | | | | ;GO CLEAR AND STORE INT UPDATE TIMER |
| C540 | LC540: | ldaa | L8998 | ;? OPTION WORD - TIS SET |
| C543 | | bne | LC552 | ;BRANCH IF NOT Z, ELSE |
| C545 | | pshb | | ;SAVE INTEGRATOR ON STACK |
| C546 | | ldab | *L00AE | ;BLM CELL |
| C548 | | ldx | #0x89CC | ;TABLE ADDRESS |
| C54B | | abx | | ;ADD TO ADDRESS |
| C54C | | ldab | 0x00,x | ;GET A VALUE FROM THE TABLE |
| C54E | | tba | | ;TRANSFER TO A REG |
| C54F | | pulb | | ;GET BACK INTEGRATOR |
| C550 | | bra | LC55A | |
| C552 | LC552: | ldaa | *L00BA | ;LOAD LV8 |
| C554 | | ldx | #0x89CC | ; |
| C557 | | jsr | L99D7 | ;2D LOOKUP |
| C55A | LC55A: | cmpa | L010E | ;FILTERED O2 |
| C55D | | bcs | LC583 | ;BRANCH IF RESULT < FILTERED O2, ELSE |
| C55F | | ldaa | L8998 | ;? OPTION WORD - TIS SET |
| C562 | | bne | LC571 | ;BRANCH IF NOT Z, ELSE |
| C564 | | pshb | | ;SAVE INTEGRATOR ON STACK |
| C565 | | ldab | *L00AE | ;BLM CELL |
| C567 | | ldx | #0x89BB | ;TABLE ADDRESS |
| C56A | | abx | | ;ADD TO ADDRESS |
| C56B | | ldab | 0x00,x | ;GET A VALUE FROM THE ADDRESS |
| C56D | | tba | | ;COPY TO A REG |
| C56E | | pulb | | ;GET BACK INTEGRATOR |
| C56F | | bra | LC579 | ; |
| C571 | LC571: | ldaa | *L00BA | ;LOAD LV8 |
| C573 | | ldx | #0x89BB | ;TABLE ADDRESS |
| C576 | | jsr | L99D7 | ;2D LOOKUP |
| C579 | LC579: | cmpa | L010E | ;FILTERED O2 |
| C57C | | bls | LC59C | ;BRANCH IF FILTERED O2 <= LOOKUP, ELSE |
| C57E | LC57E: | incb | | ;INCREMENT INTEGRATOR |
| C57F | | bne | LC588 | ;BRANCH IF NOT Z, ELSE |
| C581 | | bra | LC59C | ;GO CLEAR AND STORE INT UPDATE TIMER |
| C583 | LC583: | subb | #0x01 | ;SUBTRACT 1 |
| C585 | | bcc | LC588 | ;BRANCH IF NO UNDERFLOW, ELSE |
| C587 | | incb | | ;INCREMENT INTEGRATOR |
| C588 | LC588: | cmpb | L8981 | ;MAX INTEGRATOR 182d |
| C58B | | bls | LC592 | ;BRANCH IF <= MAX, ELSE |
| C58D | | ldab | L8981 | ;LOAD MAX INTEGRATOR |
| C590 | | bra | LC59A | ;GO SAVE INTEGRATOR |
| C592 | LC592: | cmpb | L8980 | ;MIN INTEGRATOR 100d |
| C595 | | bcc | LC59A | ;BRANCH IF >= MIN INT, ELSE |
| C597 | | ldab | L8980 | ;LOAD MIN INTEGRATOR |
| C59A | LC59A: | stab | *L00B3 | ;STORE INTEGRATOR |
| C59C | LC59C: | clrb | | ;CLEAR |
| C59D | LC59D: | stab | L0115 | ;INTEGRATOR UPDATE TIMER |
| C5A0 | | ldab | *L00AE | ;BLM CELL |
| C5A2 | | ldaa | L801B | ;OPTION WORD (TIS SET) |
| C5A5 | | bne | LC5AD | ;BRANCH IF NZ, ELSE |
| C5A7 | | brclr | *L008F,#0xC0,LC5AD | ;BRANCH IF NOT IDLE FUEL, ELSE |
| C5AB | | ldab | #0x10 | ;LOAD OFFSET |
| C5AD | LC5AD: | ldx | #0x8A01 | ;INDEX |
| C5B0 | | ldaa | *L00A6 | ;LOAD CLNT TEMP (DEFAULTED) |
| C5B2 | | cmpa | L8A00 | ;45 DEG C |
| C5B5 | | bcs | LC5C6 | ;BRANCH IF CTS2 < 45 DEG C, ELSE |
| C5B7 | | ldx | #0x89DD | ;TABLE ADDRESS |

| | | | |
|------|-------------|--------------------|--|
| C5BA | brclr | *L0096,#0x08,LC5C1 | ;BRANCH IF TCC NOT LOCKED, ELSE |
| C5BE | ldx | #0x89EF | ;TABLE ADDRESS |
| C5C1 | LC5C1:brclr | *L008F,#0x80,LC5C6 | ;BRANCH IF CCP DC 0, ELSE |
| C5C5 | inx | | ;89DE OR |
| | | | ;89F0 |
| C5C6 | LC5C6:abx | | ;ADD TO BLM CELL OR 16 TO TABLE ADDRESS |
| C5C7 | brset | *L009C,#0x40,LC5D6 | ;BRANCH IF RICH FLAG SET |
| C5CB | ldab | 0x00,x | ;GET PROP TERM FROM TABLE |
| C5CD | ldaa | *L00B3 | ;LOAD INTEGRATOR |
| C5CF | aba | | ;ADD TO INTEGRATOR |
| C5D0 | bcc | LC5DE | ;BRANCH IF NO OVERFLOW, ELSE |
| C5D2 | ldaa | #0xFF | ;LOAD 255 |
| C5D4 | bra | LC5DE | ;GO STORE L02A5 BPW FINE CORRECTION |
| C5D6 | LC5D6:ldab | 0x00,x | ;GET PROPORTIONAL TERM FROM THE TABLE |
| C5D8 | ldaa | *L00B3 | ;LOAD INTEGRATOR |
| C5DA | sba | | ;SUBTRACT B FROM INTEGRATOR |
| C5DB | bcc | LC5DE | ;BRANCH IF NO UNDERFLOW, ELSE |
| C5DD | clra | | ;CLEAR A |
| C5DE | LC5DE:staa | L02A5 | ;BPW FINE CORRECTION |
| C5E1 | LC5E1:ldx | #0x8BBD | ;INDEX |
| C5E4 | brset | *L001B,#0x01,LC5F0 | ;BRANCH IF TPS VOLTS HIGH, ELSE |
| C5E8 | brset | *L001D,#0x40,LC5F0 | ;BRANCH IF TRANS RANGE SW ERROR, ELSE |
| C5EC | brclr | *L001C,#0xA0,LC5F4 | ;BRANCH IF NO VSS OR TPS LOW ERROR, ELSE |
| C5F0 | LC5F0:clra | | ;LOAD 0000 |
| C5F1 | clrb | | ; |
| C5F2 | bra | LC630 | ;GO STORE L0273 |
| C5F4 | LC5F4:ldd | L0273 | ;LOAD |
| C5F7 | bne | LC616 | ;BRANCH IF NZ, ELSE |
| C5F9 | ldaa | *L00BD | ;FILTERED MPH |
| C5FB | cmpa | 0x00,x | ;8BBD |
| C5FD | bhi | LC65F | ;BRANCH IF > CAL, ELSE |
| C5FF | brset | *L009B,#0x10,LC607 | ;BRANCH IF REVERSE->DRV TRANSITION, ELSE |
| C603 | brclr | *L009E,#0x01,LC65F | ; |
| C607 | LC607:ldab | *L00AA | ;%TPS |
| C609 | cmpb | 0x03,x | ;8BC0 |
| C60B | bhi | LC625 | ;BRANCH IF > 39.8 %TPS,ELSE |
| C60D | ldd | *L00EB | ;RPM |
| C60F | cpd | 0x01,x | ;8BBE |
| C612 | bhi | LC625 | ;BRANCH IF > CAL, ELSE |
| C614 | bra | LC65F | |
| C616 | LC616:ldab | *L00AA | ;%TPS |
| C618 | cmpb | L8BC1 | ;0x1E |
| C61B | bls | LC62A | ;BRANCH IF <= CAL, ELSE |
| C61D | brset | *L009B,#0x10,LC625 | ;BRANCH IF REVERSE->DRV TRANSITION, ELSE |
| C621 | brclr | *L009E,#0x01,LC62A | ; |
| C625 | LC625:ldd | L8BC2 | ;0x00F0 |
| C628 | bra | LC630 | ;GO STORE L0273 |
| C62A | LC62A:ldd | L0273 | ;LOAD TIMER |
| C62D | subd | #0x0001 | ;DECREMENT BY 1 |
| C630 | LC630:std | L0273 | ;TIMER |
| C633 | beq | LC64F | ;BRANCH IF Z, ELSE |
| C635 | brclr | *L008E,#0x01,LC63B | ;BRANCH IF NOT P/N MODE, ELSE |
| C639 | inx | | ;8BBE |
| C63A | inx | | ;8BBF |
| C63B | LC63B:ldd | *L00EB | ;16 BIT RPM |
| C63D | cpd | 0x07,x | ;8BC4 OR |
| | | | ;8BC6 |
| C640 | bls | LC64A | ;BRANCH IF <= CAL, ELSE |
| C642 | bclr | *L009E,#0x20 | ;CLEAR BIT 5 |
| C645 | bset | *L009E,#0x80 | ;SET OVERREV |

```

C648          bra      LC65F          ;
C64A  LC64A:cpd      0x0B,x          ;8BC8 OR
                                           ;8BCA
C64D          bcc      LC654          ;BRANCH IF >= CAL, ELSE
C64F  LC64F:bclr     *L009E,#0xA0    ;CLEAR BITS 5 AND 7
C652          bra      LC65F          ;
C654  LC654:cpd      0x0F,x          ;8BCC OR
                                           ;8BCE
C657          bls      LC65F          ;BRANCH IF <= ,ELSE
C659          bclr     *L009E,#0x80    ;CLEAR OVERREV
C65C          bset     *L009E,#0x20    ;SET
C65F  LC65F:bclr     *L009E,#0x01    ;CLEAR BIT 0
C662          brclr    *L0099,#0x04,LC67A ;BRANCH IF NOT CLEAR FLOOD MODE, ELSE
C666          ldaa     *L00AC          ;LOAD RPM/25
C668          cmpa     L86CD           ;500 RPM
C66B          bhi      LC677           ;BRANCH IF > 500 RPM, ELSE
C66D          ldaa     *L00AA          ;%TPS
C66F          cmpa     L86CC           ;60% TPS
C672          bls      LC677           ;BRANCH IF <= 60% TPS
C674          jmp      LC700           ;GO CUT FUEL

;-----
; FUEL CUT/RESTORE
;-----
C677  LC677:bclr     *L0099,#0x04    ;CLEAR "CLEAR FLOOD MODE"
C67A  LC67A:ldaa     L8011           ;OPTION WORD - PASSKEY II IN USE
C67D          beq      LC683           ;BRANCH IF Z, ELSE
C67F          brclr    *L0012,#0x02,LC700 ;BRANCH IF VATS NOT OKAY, ELSE
C683  LC683:brset     *L009E,#0x80,LC700 ;BRANCH IF OVERREV, ELSE
C687          brset     *L009B,#0x02,LC700 ;BRANCH IF DFCO ENABLED, ELSE
C68B          brset     *L0092,#0x10,LC700 ;BRANCH IF BIT 4, ELSE (WILL ALWAYS BE CLR)
C68F          brset     *L0095,#0x40,LC6A6 ;SKIP COLD ENGINE REV LIMIT IF SET, ELSE
C693          ld        #0x982F       ;RPM FUEL CUT VS STARTUP COOLANT TEMP
C696          brclr    *L009B,#0x40,LC69D ;BRANCH IF NOT FUEL CUT, ELSE
C69A          ld        #0x9840       ;RPM FUEL RESTORE VS STARTUP COOLANT TEMP
C69D  LC69D:ldaa     *L00F2           ;LOAD STARTUP COOLANT
C69F          jsr      L99D7           ;2D LOOKUP
C6A2          cmpa     *L00AC          ;COMPARE LOOKUP RESULT AND RPM
C6A4          bcs      LC700           ;BRANCH IF < CAL, ELSE
C6A6  LC6A6:ldx       #0x8B78         ;INDEX
C6A9          brclr    *L009B,#0x40,LC6AE ;BRANCH IF NOT FUEL CUT, ELSE
C6AD          inx              ;8B79
C6AE  LC6AE:ldaa     *L00BD           ;FILTERED MPH
C6B0          brclr    *L008E,#0x10,LC6BC ;BRANCH IF 5TH GEAR, ELSE
C6B4          inx              ;8B7A
C6B5          inx              ;8B7B
C6B6          brclr    *L008E,#0x08,LC6BC ;BRANCH IF 4TH GEAR, ELSE
C6BA          inx              ;8B7C
C6BB          inx              ;8B7D
C6BC  LC6BC:cmpa     0x00,x          ;8B78 OR
                                           ;8B79 OR
                                           ;8B7B OR
                                           ;8B7D
C6BE          bcs      LC6C8           ;BRANCH IF < CAL, ELSE
C6C0          ldd      *L00EB          ;TRANNNY RPM
C6C2          cpd      L8B7E           ;3000
C6C6          bcc      LC700           ;BRANCH IF >= 3000 RPM (CUT FUEL), ELSE
C6C8  LC6C8:ldx       #0x8B68         ;
C6CB          ldab     #0x04          ;
C6CD          ldaa     L8010           ;OPTION WORD - TRANS TYPE - TIS CLEAR
C6D0          bpl      LC6D9           ;BRANCH IF AUTO TRANS, ELSE
C6D2          brclr    *L008E,#0x01,LC6F3 ;BRANCH IF NOT P/N MODE, ELSE

```

```

C6D6      abx                      ;ADD 4 TO INDEX = 8B6C
C6D7      bra      LC6F3

C6D9      LC6D9:brset  *L0015,#0x40,LC6F3      ;BRANCH IF TRANS RANGE SW MALF, ELSE
C6DD      brclr  *L00A1,#0x20,LC6E4      ;BRANCH IF PRNDL -> NOT NEUTRAL, ELSE
C6E1      abx                      ;ADD 4 TO INDEX = 8B6C
C6E2      bra      LC6F3

C6E4      LC6E4:brclr  *L00A1,#0x80,LC6EC      ;BRANCH IF PRNDL -> NOT PARK, ELSE
C6E8      abx                      ;ADD 4 TO INDEX = 8B6C
C6E9      abx                      ;ADD 4 TO INDEX = 8B70
C6EA      bra      LC6F3      ;

C6EC      LC6EC:brclr  *L00A1,#0x40,LC6F3      ;BRANCH IF PRNDL -> NOT REVERSE, ELSE
C6F0      abx                      ;ADD 4 TO INDEX = 8B6C
C6F1      abx                      ;ADD 4 TO INDEX = 8B70
C6F2      abx                      ;ADD 4 TO INDEX = 8B74
C6F3      LC6F3:brclr  *L009B,#0x40,LC6F9      ;BRANCH IF NOT FUEL CUT, ELSE
C6F7      inx                      ;8B75
C6F8      inx                      ;8B76
C6F9      LC6F9:ldd      *L00EB      ;16 BIT RPM
C6FB      cpd      0x00,x      ;COMPARE
C6FE      bls      LC712      ;BRANCH IF <= CAL, ELSE

;-----
; FUEL CUT - CLEAR INJ PW
;-----

C700      LC700:bset   *L009B,#0x40      ;SET OVERREV CONDITION
C703      ldx      #0x0000      ;LOAD 0000
C706      stx      L0104      ;CLEAR BPW
C709      stx      L0102      ;CLEAR INJ PW
C70C      stx      L0106      ;ASYNCR PW
C70F      jmp      LC78C

C712      LC712:bclr   *L009B,#0x40      ;CLEAR OVERREV CONDITION
C715      clrb                      ;CLEAR B
C716      ldaa     L0159      ;ACCEL ENRICHMENT TERM
C719      adda     L02BA      ;ADD P/N -> IN GEAR ENRICHMENT TERM
C71C      rolb                      ;MUL BY 2
C71D      suba     L02A1      ;DE TERM
C720      sbcb     #0x00      ;ROUND
C722      stab     L02BC      ;STORE LSB
C725      ldx      L01A7      ;INJ PW
C728      stx      L02BD      ;STORE PW
C72B      ldx      #0x02BD      ;LOAD PW
C72E      jsr      L98B9      ;8 X 16 MULTIPLY
C731      tst      L02BC      ;TEST LSB
C734      beq      LC740      ;BRANCH IF Z, ELSE
C736      bpl      LC73D      ;BRANCH IF PLUS, ELSE
C738      subd     L02BD      ;
C73B      bra      LC740

C73D      LC73D:addd   L02BD      ;? PW
C740      LC740:std     L0104      ;BPW CORRECTION TERM FOR A.E AND D.E.
C743      ldx      L8B85      ;7896 - INJ BASE PULSE CONSTANT
C746      pshx                      ;SAVE ON STACK
C747      tsx                      ;X POINTS TO STACK
C748      ldaa     *L00AF      ;LOAD FILTERED BLM
C74A      jsr      L989D      ;8 X 16 MULTIPLY, PRODUCT ON STACK
C74D      ldaa     L02A5      ;BPW FINE CORRECTION
C750      jsr      L98B9      ;8 X 16 MULTIPLY
C753      ror      0x00,x      ;MSB DIV BY 2
C755      ror      0x01,x      ;LSB DIV BY 2
C757      adcb     0x01,x      ;ROUND LSB

```

| | | | |
|------|-------------|--------------------|---|
| C759 | adca | 0x00,x | ;ROUND MSB |
| C75B | std | 0x00,x | ;SAVE ON STACK |
| C75D | ldx | *L00B5 | ;FUEL AIR RATIO |
| C75F | jsr | L98FA | ;16 X 16 MULTIPLY |
| C762 | std | L0102 | ;STORE PW |
| C765 | pulx | | ;RESTORE X |
| C766 | brclr | *L0084,#0x80,LC76F | ;BRANCH IF NOT MODE 4, ELSE |
| C76A | ldaa | L0239 | ;LOAD COMMANDED STATUS FLAG |
| C76D | bra | LC786 | ;GO STORE BIT STATUS FLAG |
| C76F | LC76F:brset | *L009E,#0x20,LC784 | ;BRANCH IF BIT 5, ELSE |
| C773 | ldaa | #0x24 | ;BITS 5 AND 2 |
| C775 | brset | *L00DA,#0x10,LC786 | ;THIS BIT IS CLEARED |
| C779 | ldaa | #0x20 | ;BIT 5 |
| C77B | brset | *L00DA,#0x08,LC786 | ;THIS BIT IS CLEARED |
| C77F | ldaa | L02FD | ; |
| C782 | bra | LC786 | ;STORE BIT STATUS FLAG |
| C784 | LC784:ldaa | #0xA8 | ;10101000 |
| C786 | LC786:staa | *L00E2 | ;INJECTOR BIT STATUS FLAG |
| C788 | cmpa | #0xA8 | ;10101000 |
| C78A | beq | LC791 | ;BRANCH IF BIT STATUS FLAG = 0xA8, ELSE |
| C78C | LC78C:ldaa | L8A92 | ;TPS FILTER COEFFICIENT |
| C78F | bne | LC794 | ;BRANCH IF NOT Z, ELSE |
| C791 | LC791:jmp | LC864 | ;SKIP ASYNC OUTPUT |
| C794 | LC794:pshx | | ;SAVE ON STACK |
| C795 | tsy | | ;Y POINTS TO STACK (? BPW) |
| C797 | ldaa | L031F | ;ASYNC FACTOR VS COOLANT TEMP |
| C79A | ldab | L8AAA | ;ASYNC MULTIPLIER |
| C79D | mul | | ;A * B |
| C79E | std | 0x00,y | ;SAVE ON STACK |
| C7A1 | ldaa | *L00AA | ;TPS |
| C7A3 | suba | L0153 | ;SUB FILTERED TPS% |
| C7A6 | bcc | LC7A9 | ;BRANCH IF TPS INCREASING, ELSE |
| C7A8 | clra | | ;CLEAR A |
| C7A9 | LC7A9:ldx | #0x8ABC | ;ASYNC FACTOR VS DELTA TPS |
| C7AC | jsr | L99D7 | ;2D LOOKUP |
| C7AF | tsx | | ;X POINTS TO STACK |
| C7B0 | jsr | L989D | ;8 X 16 MULTIPLY - TABLE LOOKUP RESULT IS |
| | | | ; THE MULTIPLIER, PRODUCT ON STACK |
| C7B3 | ldx | 0x00,y | ;GET FROM STACK |
| C7B6 | cpx | L0106 | ;ASYNC PW |
| C7B9 | bis | LC7BE | ;BRANCH IF >= ASYNC PW, ELSE |
| C7BB | ldx | L0106 | ;LOAD ASYNC PW |
| C7BE | LC7BE:stx | 0x00,y | ;SAVE ON STACK |
| C7C1 | stx | L02CC | ;STORE ? PREV ASYNC PW |
| C7C4 | bne | LC7C9 | ;BRANCH IF NOT Z, ELSE |
| C7C6 | jmp | LC863 | ;SKIP ASYNC OUTPUT |
| C7C9 | LC7C9:ldaa | L801C | ;OPTION WORD - TIS CLEAR |
| C7CC | bne | LC811 | ;BRANCH IF NOT Z, ELSE |
| C7CE | ldd | #0x0008 | ;MASK |
| C7D1 | ldx | #0x4000 | ;I/O DATA REG ADDRESS |
| C7D4 | sei | | ;HOLD INTERRUPTS |
| C7D5 | jsr | L9A70 | ;WR ACCD TO SPI DATA REG |
| C7D8 | ldx | 0x00,y | ;GET ASYNC PW FROM STACK |
| C7DB | jsr | LF678 | ;GO DO ASYNC FUEL OUT ROUTINE |
| C7DE | cli | | ;CLEAR AND ALLOW INTERRUPTS |
| C7DF | ldd | *L00C0 | ;LOAD REF PERIOD |
| C7E1 | lsl | | ;MUL BY 2 |
| C7E2 | add | *L00C0 | ;ADD REF PERIOD |
| C7E4 | lsl | | ;MUL BY 2 |
| C7E5 | pshb | | ;SAVE LSB |

```

C7E6      psha                      ;SAVE MSB
C7E7      pulx                      ;GET INTO X
C7E8      cpx      *L00E4           ;COMPARE FINAL BPW
C7EA      bls      LC863            ;BRANCH IF <= FINAL BPW, ELSE
C7EC      ldd      *L00E4           ;LOAD FINAL BPW
C7EE      fdiv                      ;D/X
C7EF      tsta                      ;TEST MSB OF REMAINDER
C7F0      beq      LC7F3            ;BRANCH IF Z,ELSE
C7F2      inca                      ;ROUND
C7F3      LC7F3:nega                ;INVERT
C7F4      tsx                      ;X POINTS TO STACK
C7F5      jsr      L98B9            ;8 X 16 MULTIPLY
C7F8      lsld                      ;MUL BY 2
C7F9      std      0x00,x           ;SAVE ON STACK
C7FB      lsld                      ;MUL BY 2
C7FC      addd     0x00,x           ;ROUND
C7FE      addd     L0250            ;ACCUMULATED FUEL
C801      std      L0250            ;SAVE ACCUMULATED FUEL
C804      bcc      LC863            ;BRANCH IF NO OVERFLOW, ELSE
C806      ldd      L9272            ;ACCUMULATED FUEL MULTIPLIER
C809      addd     L0270            ;ADD PREVIOUS FUEL
C80C      std      L0270            ;STORE
C80F      bra      LC863            ;

;-----
; THIS NOT USED UNLESS L801C SET - TIS CLEAR
;-----
C811      LC811:clra                ;CLEAR A
C812      ldx      #0x4000          ;I/O DATA REG
C815      bset     0x02,x,#0x10     ;SET BIT 4 I/O PORT
C818      staa     0x00,x           ;CLEAR I/O DATA REG
C81A      bclr     0x01,x,#0x80     ;CLEAR BIT 7 I/O CNTL REG
C81D      nop                      ;DELAY
C81E      nop
C81F      ldab     *L008D           ;DATA
C821      andb     #0x07            ;CLEAR BITS %00000111
C823      ldx      #0xFB7C         ;TABLE ADDRESS
C826      abx                      ;ADD TO ADDRESS
C827      ldab     0x00,x           ;GET A VALUE FROM THE TABLE
C829      nop                      ;DELAY
C82A      ldx      #0x4000          ;I/O DATA REG
C82D      ldaa     0x00,x           ;GET DATA FROM DATA REG
C82F      stab     0x00,x           ;WR TABLE VALUE TO I/O DATA REG
C831      bclr     0x01,x,#0x80     ;CLEAR BIT 7 I/O CNTL REG
C834      staa     *L008D           ;STORE DATA
C836      nop                      ;DELAY
C837      ldx      0x00,y           ;GET PW FROM STACK
C83A      ldab     *L00B4           ;FUEL INJ OFFSET VS BATT VOLTS
C83C      abx                      ;ADD TO X
C83D      stx      L3FF2           ;ASYNC PW
C840      mul                      ;DELAY
C841      ldd      L3FFC           ;CPU CNTL REG
C844      oraa     #0x04            ;TRIGGER ASYNC PW
C846      psha                      ;SAVE ON STACK
C847      ldx      #0x4000          ;I/O DATA REG
C84A      ldaa     0x00,x           ;GET DATA FROM REG
C84C      staa     *L008C           ;STORE DATA
C84E      bita     #0x40            ;BIT 6 ?
C850      beq      LC855            ;BRANCH IF NO CAM REF PULSE, ELSE
C852      bset     *L0093,#0xC0     ;SET CAM REF PULSE OCCURED
C855      LC855:bclr 0x02,x,#0x10   ;SET BIT 4 I/O PORT
C858      pula                      ;
C859      std      L3FFC           ;CPU CNTL REG
C85C      anda     #0xFB           ;CLEAR ASYNC PW

```

```

C85E      pshx                      ;DELAY
C85F      pulx
C860      std      L3FFC            ;CPU CNTL REG

C863      LC863:pulx                ;GET PW FROM STACK
C864      LC864:ldd      *L00CD      ;LOAD MAF (GM/SEC)
C866      jsr      LF58E            ;GO LOOKUP MAF VS RPM, CORRECTED FOR IAT
C869      ldx      #0x917A          ;DEFAULT MAF VS TPS AND RPM/25
C86C      ldab     *L00AA            ;%TPS
C86E      cmpb     #0x80            ;50% TPS
C870      bls      LC874            ;BRANCH IF TPS <= %TPS, ELSE
C872      ldab     #0x80            ;LOAD 50% TPS
C874      LC874:ldaa     *L00AC      ;LOAD RPM/25
C876      cmpa     #0xC0            ;192: 4800 RPM
C878      bls      LC883            ;BRANCH IF <= 4800 RPM, ELSE
C87A      ldx      #0x91E9          ;LOAD DEFAULT MAF VS %TPS
C87D      tba
C87E      jsr      L99D7            ;TRANSFER TPS TO A REG
C881      bra      LC886            ;2D LOOKUP
;

C883      LC883:jsr      L995A      ;3D LOOKUP
C886      LC886:clrb                ;CLEAR B
C887      pshb                      ;SAVE ON STACK
C888      psha                      ;SAVE TABLE LOOKUP RESULT ON STACK
C889      ldaa     *L0057            ;IAC POSITION
C88B      ldab     L90C1            ;IAC AIRFLOW MULTIPLIER
C88E      mul                      ;L0057 * 90C1
C88F      cpd      L90BF            ;COMPARE
C893      bls      LC898            ;BRANCH IF <= 90BF, ELSE
C895      ldd      L90BF            ;LOAD 90BF
C898      LC898:tsx                ;X POINTS TO STACK
C899      addd     0x00,x            ;ADD TO DEFAULT MAF
C89B      addd     L90BD            ;
C89E      std      L0256            ;STORE DEFAULT MAF
C8A1      pulx                      ;GET TABLE RESULT INTO X REG
C8A2      brclr   *L009A,#0x20,LC8BF ;THIS SET WHEN ? RUN FUEL/SPARK ALLOWED
C8A6      ldaa     L027E            ;LOAD
C8A9      inca                      ;INCREMENT
C8AA      cmpa     L86CE            ;1
C8AD      bcs      LC8BC            ;BRANCH IF < CAL, ELSE
C8AF      ldab     L027F            ;CRANK TO RUN PW MULTIPLIER
C8B2      subb     L86CF            ;8
C8B5      bcc      LC8B8            ;BRANCH IF NO UNDERFLOW, ELSE
C8B7      clrb                      ;CLEAR
C8B8      LC8B8:stab     L027F      ;SAVE CRANK TO RUN PW MULTIPLIER
C8BB      clra
C8BC      LC8BC:staa     L027E

;-----
; JUMP HERE IF < 200 RPM
;-----
C8BF      LC8BF:bclr   *L009A,#0x20 ;CLEAR BIT 5
C8C2      ldaa     L9272            ;? EMPIRICALLY DERIVED MULTIPLIER FOR
; ACCUMULATED FUEL
C8C5      sei                      ;HOLD INTERRUPTS
C8C6      ldx      #0x0250          ;LOAD ACCUMULATED FUEL
C8C9      jsr      L98B9            ;8 x 16 MULTIPLY
C8CC      addd     L0270            ;ADD ACCUMULATED FUEL
C8CF      staa     L0272            ;STORE ACCUMULATED FUEL
C8D2      cli                      ;CLEAR AND ALLOW INTERRUPTS

;-----
; EGR ROUTINE - ?12.5 MSEC LOOP (634 BYTES)
;-----

```

| | | | |
|------|-------------|--------------------|---|
| C8D3 | brclr | *L0084,#0x80,LC8EA | ;BRANCH IF NOT MODE 4, ELSE |
| C8D7 | ldaa | #0xFF | ;LOAD 255 |
| C8D9 | brset | *L0088,#0x01,LC8E7 | ;BRANCH IF "ALL PWM OUTPUTS COMMANDED ON" |
| C8DD | ldaa | L0232 | ;MODE 4 CONTROL WORD |
| C8E0 | bita | #0x02 | ;EGR COMMANDED ? |
| C8E2 | beq | LC8EA | ;BRANCH IF NOT, ELSE |
| C8E4 | ldaa | L0233 | ;LOAD COMMANDED DUTY CYCLE |
| C8E7 | LC8E7: jmp | LC99A | ;GO STORE EGR DUTY CYCLE |
| | | | |
| C8EA | LC8EA:brset | *L0098,#0x20,LC8FC | ;BRANCH IF EGR ENABLED, ELSE |
| C8EE | ldd | L8F32 | ;COOLANT THRESHOLD |
| C8F1 | cmpa | *L00F2 | ;COMPARE STARTUP COOLANT |
| C8F3 | bcs | LC8F9 | ;BRANCH IF > CAL, ELSE |
| C8F5 | cmpb | *L00A7 | ;FILTERED CTS |
| C8F7 | bhi | LC938 | ;BRANCH IF < 74 DEG C (GO CLEAR EGR), ELSE |
| C8F9 | LC8F9:bset | *L0098,#0x20 | ;SET EGR ENABLED |
| C8FC | LC8FC:ldab | L0185 | ;TIME DELAY FROM STARTUP |
| C8FF | clra | | ; |
| C900 | lsld | | ; |
| C901 | subd | *L0032 | ;RUN TIME |
| C903 | bcc | LC938 | ;BRANCH IF >= RUN TIME (GO CLEAR EGR), ELSE |
| C905 | brset | *L008E,#0x01,LC938 | ;BRANCH IF P/N MODE, ELSE |
| C909 | brclr | *L0099,#0x01,LC90F | ; |
| C90D | bra | LC938 | ;GO CLEAR AND STORE EGR DC |
| | | | |
| C90F | LC90F:ldaa | L8F2F | ;OPTION WORD (TIS SET) |
| C912 | beq | LC918 | ;BRANCH IF Z, ELSE |
| C914 | brclr | *L009C,#0x80,LC938 | ;BRANCH IF NOT CLOSED LOOP, ELSE |
| C918 | LC918:brset | *L009B,#0x20,LC938 | ;BRANCH IF PE MODE (GO CLEAR EGR), ELSE |
| C91C | ldab | L8F34 | ;92.5 %TPS THRESHOLD TO DISABLE EGR |
| C91F | ldaa | L01AF | ;EGR DC |
| C922 | beq | LC927 | ;BRANCH IF Z, ELSE |
| C924 | ldab | L8F35 | ;89.9 %TPS THRESHOLD TO DISABLE EGR |
| C927 | LC927:cmpb | *L00AA | ; %TPS |
| C929 | bcs | LC938 | ;BRANCH IF %TPS > CAL (GO CLEAR EGR), ELSE |
| C92B | ldab | L8F30 | ;1.6 %TPS THRESHOLD TO DISABLE EGR |
| C92E | tsta | | ;TEST L01AF - ? EGR DC |
| C92F | beq | LC934 | ;BRANCH IF Z, ELSE |
| C931 | ldab | L8F31 | ;1.2 %TPS THRESHOLD TO DISABLE EGR |
| C934 | LC934:cmpb | *L00AA | ; %TPS |
| C936 | bis | LC93B | ;BRANCH IF CAL <= TPS%, ELSE |
| C938 | LC938:clra | | ;CLEAR A |
| C939 | bra | LC99A | ;GO STORE EGR |
| | | | |
| C93B | LC93B:ldaa | L8010 | ;OPTION WORD - TRANS TYPE - TIS CLEAR |
| C93E | bmi | LC946 | ;BRANCH IF MANUAL TRANS, ELSE |
| C940 | brclr | *L0096,#0x08,LC95B | ;BRANCH IF TCC NOT LOCKED, ELSE |
| C944 | bra | LC94D | ; |
| | | | |
| C946 | LC946:ldx | #0x8F62 | ;EGR DUTY CYCLE MULTIPLIER VS LV8 |
| C949 | brset | *L008E,#0x02,LC95F | ;BRANCH IF NOT 2 ND GEAR, ELSE |
| C94D | LC94D:ldx | #0x8F48 | ;EGR DUTY CYCLE MULTIPLIER VS LV8 |
| C950 | brclr | *L008E,#0x08,LC95F | ;BRANCH IF NOT 4 TH GEAR, ELSE |
| C954 | ldx | #0x8F55 | ;EGR DUTY CYCLE MULTIPLIER VS LV8 |
| C957 | brclr | *L008E,#0x04,LC95F | ;BRANCH IF NOT 3 RD GEAR, ELSE |
| C95B | LC95B:ldaa | #0x80 | ; |
| C95D | bra | LC966 | |
| | | | |
| C95F | LC95F:ldaa | *L00BA | ;LOAD LV8 |
| C961 | ldab | #0x40 | ;OFFSET |
| C963 | jsr | L99D3 | ;2D LOOKUP |
| C966 | LC966:ldab | L0186 | ;LOAD EGR DC MULTIPLIER VS COOLANT TEMP |
| C969 | mul | | ;L0186 * TABLE LOOKUP RESULT |
| C96A | lsld | | ;MUL BY 2 |

| | | | | | |
|------|--------|-------|--------------------|--|--|
| C96B | | bcc | LC96F | | ;BRANCH IF NO OVERFLOW, ELSE |
| C96D | | ldaa | #0xFF | | ;LOAD 255 |
| C96F | LC96F: | ldab | L01B6 | | ;LOAD 8F70 TABLE LOOKUP RESULT |
| C972 | | mul | | | ; |
| C973 | | lsl | | | ;MUL BY 2 |
| C974 | | bcc | LC978 | | ;BRANCH IF NO OVERFLOW, ELSE |
| C976 | | ldaa | #0xFF | | ;LOAD 255 |
| C978 | LC978: | psha | | | ;SAVE MULTIPLIER ON STACK |
| C979 | | ldx | #0x9000 | | ;BASE EGR SOLN COMBO VS LV8 AND RPM |
| C97C | | ldaa | *L00AB | | ;LOAD NLRPMX |
| C97E | | cmpa | #0xC0 | | ;4800 RPM |
| C980 | | bis | LC98E | | ;BRANCH IF <= 4800 RPM, ELSE |
| C982 | | ldaa | *L00BA | | ;LOAD LV8 |
| C984 | | ldab | #0x30 | | ;LOAD OFFSET |
| C986 | | ldx | #0x909D | | ;BASE EGR SOLN COMBO VS LV8 FOR RPM > 4800 |
| C989 | | jsr | L99D3 | | ;2D LOOKUP |
| C98C | | bra | LC993 | | |
| C98E | LC98E: | ldab | *L00BA | | ;LOAD LV8 |
| C990 | | jsr | L995A | | ;3D LOOKUP |
| C993 | LC993: | pulb | | | ;GET MULTIPLIER FROM STACK -> B REG |
| C994 | | mul | | | ;LOOKUP RESULT * B |
| C995 | | lsl | | | ;MUL BY 2 |
| C996 | | bcc | LC99A | | ;BRANCH IF NO OVERFLOW, ELSE |
| C998 | | ldaa | #0xFF | | ;LOAD 255 |
| C99A | LC99A: | brclr | *L0095,#0x82,LC9A1 | | ;BRANCH IF HUD DEVICE NOT CONNECTED, ELSE |
| C99E | | jsr | L500F | | ;UPDATE HUD |
| C9A1 | LC9A1: | staa | L01AF | | ;EGR DC |
| C9A4 | | ldaa | L01AF | | ;LOAD EGR DC |
| C9A7 | | tab | | | ;COPY TO B REG |
| C9A8 | | suba | L01AE | | ;SUBTRACT PREV EGR DUTY CYCLE |
| C9AB | | bcc | LC9AE | | ;BRANCH IF EGR DC INCREASING, ELSE |
| C9AD | | neg | | | ;INVERT RESULT |
| C9AE | LC9AE: | stab | L01B0 | | ;EGR DUTY CYCLE DELTA |
| C9B1 | | cmpa | L8F6F | | ;MIN EGR DC DELTA |
| C9B4 | | bhi | LC9B9 | | ;BRANCH IF > 3, ELSE |
| C9B6 | | ldab | L01AE | | ;LOAD PREV EGR DUTY CYCLE |
| C9B9 | LC9B9: | pshb | | | ;SAVE EGR DC DELTA ON STACK |
| C9BA | | bclr | *L0088,#0x08 | | ;CLEAR BIT 3 |
| C9BD | | ldaa | #0x20 | | ;BIT 5 |
| C9BF | | jsr | LD933 | | ;GO CHECK MODE 4 CONTROL WORDS |
| C9C2 | | beq | LC9D1 | | ;BRANCH IF Z, ELSE |
| C9C4 | | lsla | | | ;MUL BY 2 |
| C9C5 | | bset | *L0088,#0x08 | | ;SET BIT 3 |
| C9C8 | | tsx | | | ;X POINTS TO STACK |
| C9C9 | | bclr | 0x00,x,#0x20 | | ;CLEAR BIT 5 |
| C9CC | | bcc | LC9D1 | | ;BRANCH IF OVERFLOW, ELSE |
| C9CE | | bset | 0x00,x,#0x20 | | ;SET BIT 5 |
| C9D1 | LC9D1: | ldaa | #0x10 | | ;BIT 4 |
| C9D3 | | jsr | LD933 | | ;GO CHECK MODE 4 CONTROL WORDS |
| C9D6 | | beq | LC9E5 | | ;BRANCH IF Z, ELSE |
| C9D8 | | lsla | | | ;MUL BY 2 |
| C9D9 | | bset | *L0088,#0x08 | | ;SET BIT 3 |
| C9DC | | tsx | | | ;X POINTS TO STACK |
| C9DD | | bclr | 0x00,x,#0x40 | | ;CLEAR BIT 6 |
| C9E0 | | bcc | LC9E5 | | ;BRANCH IF OVERFLOW, ELSE |
| C9E2 | | bset | 0x00,x,#0x40 | | ;SET BIT 6 |
| C9E5 | LC9E5: | ldaa | #0x08 | | ;BIT 3 |
| C9E7 | | jsr | LD933 | | ;GO CHECK MODE 4 CONTROL WORDS |
| C9EA | | beq | LC9F9 | | ;BRANCH IF Z, ELSE |
| C9EC | | lsla | | | ;MUL BY 2 |
| C9ED | | bset | *L0088,#0x08 | | ;SET BIT 3 |
| C9F0 | | tsx | | | ;X POINTS TO STACK |
| C9F1 | | bclr | 0x00,x,#0x80 | | ;CLEAR BIT 7 |

| | | | |
|------|-------------|--------------------|--|
| C9F4 | bcc | LC9F9 | ;BRANCH IF OVERFLOW, ELSE |
| C9F6 | bset | 0x00,x,#0x80 | ;SET BIT 7 |
| C9F9 | LC9F9:brclr | *L0088,#0x08,LCA09 | ;BRANCH IF NOT BIT 3, ELSE |
| C9FD | tsx | | ;X POINTS TO STACK |
| C9FE | ldab | 0x00,x | ;GET EGR DUTY CYCLE |
| CA00 | stab | L01AE | ;EGR DUTY CYCLE |
| CA03 | stab | L01B0 | ;EGR DUTY CYCLE DELTA |
| CA06 | jmp | LCB1D | ;GO EXERCISE EGR SOLN OUTPUTS |
| CA09 | LCA09:brset | *L002A,#0xE0,LCA76 | ;EGR BIT STATUS FLAG |
| CA0D | ldab | #0x80 | ; |
| CA0F | ldx | #0x911D | ; |
| CA12 | ldy | #0x0026 | ;INDEX |
| CA16 | brclr | *L002A,#0x80,LCA26 | ;EGR BIT STATUS FLAG |
| CA1A | lsrb | | ; |
| CA1B | inx | | ;911E |
| CA1C | iny | | ;0027 |
| CA1E | brclr | *L002A,#0x40,LCA26 | ;EGR BIT STATUS FLAG |
| CA22 | lsrb | | ; |
| CA23 | inx | | ;911F |
| CA24 | iny | | ;0028 |
| CA26 | LCA26:ldaa | *L00BF | ;MPH*3 |
| CA28 | cmpa | L911F | ;69 MPH |
| CA2B | bis | LCA8E | ;BRANCH IF <= CAL, ELSE |
| CA2D | ldaa | *L00A6 | ;LOAD CLNT TEMP (DEFAULTED) |
| CA2F | cmpa | L911D | ;83.7 DEG C |
| CA32 | bis | LCA8E | ;BRANCH IF <= 83.7 DEG C, ELSE |
| CA34 | brclr | *L009C,#0x80,LCA8E | ;BRANCH IF NOT CLOSED LOOP, ELSE |
| CA38 | brset | *L008E,#0x01,LCA8E | ;BRANCH IF P/N MODE, ELSE |
| CA3C | brset | *L009B,#0x02,LCA8E | ;BRANCH IF DFCO ENABLED, ELSE |
| CA40 | brclr | *L0086,#0x20,LCA8E | ;BRANCH IF A/C CLUTCH ON, ELSE |
| CA44 | brset | *L008E,#0x0C,LCA8E | ;BRANCH IF NOT 3 RD OR 4 TH GEAR, ELSE |
| CA48 | brset | *L0013,#0x01,LCA8E | ;BRANCH IF TPS VOLTAGE HIGH, ELSE |
| CA4C | brset | *L0014,#0x80,LCA8E | ;BRANCH IF TPS VOLTS LOW, ELSE |
| CA50 | ldaa | *L00DF | ;THROTTLE FOLLOWER IAC STEPS |
| CA52 | bne | LCA8E | ;BRANCH IF NOT Z, ELSE |
| CA54 | ldaa | *L00AA | ;%TPS |
| CA56 | cmpa | L8CE2 | ;1.2 %TPS |
| CA59 | bhi | LCA8E | ;BRANCH IF > 1.2 %TPS, ELSE |
| CA5B | ldaa | L01B1 | ;TIMER |
| CA5E | cmpa | 0x09,x | ;9126 OR |
| | | | ;9127 OR |
| | | | ;9128 |
| CA60 | bcc | LCA78 | ;BRANCH IF TIMER > CAL, ELSE |
| CA62 | ldaa | *L00A9 | ;MINOR LOOP O2 A/D COUNTS (mV) |
| CA64 | cmpa | L911E | ;566 mV |
| CA67 | bis | LCA8E | ;BRANCH IF <= 566, ELSE |
| CA69 | ldaa | *L00AB | ;LOAD NLRPMX |
| CA6B | cmpa | 0x03,x | ;9120 OR |
| | | | ;9121 OR |
| | | | ;9122 |
| CA6D | bcc | LCA8E | ;BRANCH IF RPM > CAL, ELSE |
| CA6F | cmpa | 0x06,x | ;9123 OR |
| | | | ;9124 OR |
| | | | ;9125 |
| CA71 | bis | LCA8E | ;BRANCH IF RPM <= CAL, ELSE |
| CA73 | inc | L01B1 | ;INCREMENT TIMER |
| CA76 | LCA76:bra | LCADB | ;GO EXERCISE EGR SOLN OUTPUTS |
| CA78 | LCA78:ins | | ;INCREMENT STACK POINTER |
| CA79 | pshb | | ;SAVE EGR DUTY CYCLE ON STACK |
| CA7A | brset | *L002A,#0x10,LCA91 | ;EGR BIT STATUS FLAG |
| CA7E | bset | *L002A,#0x10 | ;EGR BIT STATUS FLAG |
| CA81 | bclr | *L002A,#0x04 | ;EGR BIT STATUS FLAG |

| | | | |
|------|--------------|----------------------|--|
| CA84 | ldaa | *L00AB | ;LOAD NLRPMX |
| CA86 | staa | L01B2 | ;STORE PREV RPM |
| CA89 | clr | L01B3 | ;CLEAR TIMER |
| CA8C | bra | LCADB | ; |
| CA8E | LCA8E: jmp | LCB14 | ;CLEAR TIMERS AND EXERCISE EGR OUTPUTS |
| CA91 | LCA91: ldaa | *L00A9 | ;MINOR LOOP O2 A/D COUNTS (mV) |
| CA93 | cmpa | L9135 | ;199 mV |
| CA96 | bcc | LCAA1 | ;BRANCH IF > 199 mV, ELSE |
| CA98 | ldaa | L01B4 | ;LOAD TIMER |
| CA9B | inca | | ;INCREMENT |
| CA9C | beq | LCAA1 | ;BRANCH IF Z, ELSE |
| CA9E | staa | L01B4 | ;STORE TIMER |
| CAA1 | LCAA1: brset | *L002A, #0x04, LCADD | ;EGR BIT STATUS FLAG |
| CAA5 | ldaa | L01B2 | ;LOAD PREV RPM |
| CAA8 | cmpa | *L00AB | ;COMPARE NLRPMX |
| CAA | bcc | LCAB1 | ;BRANCH IF RPM DECREASING, ELSE |
| CAAC | ldaa | *L00AB | ;LOAD NLRPMX |
| CAAE | staa | L01B2 | ;PREV RPM |
| CAB1 | LCAB1: suba | *L00AB | ;SUB NLRPMX |
| CAB3 | cpx | #0x911D | ; |
| CAB6 | bne | LCABB | ;BRANCH IF NOT |
| CAB8 | staa | L0329 | ;STORE DELTA RPM |
| CABB | LCABB: cpx | #0x911E | ; |
| CABE | bne | LCAC3 | ;BRANCH IF NOT |
| CAC0 | staa | L032A | ;STORE DELTA RPM |
| CAC3 | LCAC3: cpx | #0x911F | ; |
| CAC6 | bne | LCACB | ;BRANCH IF THEY'RE NOT EQUAL, ELSE |
| CAC8 | staa | L032B | ;STORE DELTA RPM |
| CACB | LCACB: cmpa | 0x0C, x | ;9129 OR |
| CACD | bls | LCAF1 | ;912A OR |
| CACF | ldaa | L01B3 | ;912B |
| CAD2 | staa | L01B5 | ;BRANCH IF <= CAL, ELSE |
| CAD5 | clr | L01B3 | ;LOAD |
| CAD8 | bset | *L002A, #0x04 | ;STORE PREV TIME |
| CADB | LCADB: bra | LCB1D | ;CLEAR TIMER |
| CADD | LCADD: ins | | ;EGR BIT STATUS FLAG |
| CADE | clra | | ;GO EXERCISE EGR SOLN OUTPUTS |
| CADF | psha | | ; |
| CAE0 | ldaa | *L00AB | ;SAVE EGR DUTY CYCLE ON STACK |
| CAE2 | cmpa | L01B2 | ;LOAD NLRPMX |
| CAE5 | bcc | LCAEA | ;PREV NLRPMX |
| CAE7 | staa | L01B2 | ;BRANCH IF RPM INCREASING, ELSE |
| CAEA | LCAEA: suba | L01B2 | ;SAVE PREV NLRPMX |
| CAED | cmpa | 0x0C, x | ; |
| CAEF | bhi | LCB10 | ;9129 OR |
| CAF1 | LCAF1: ldaa | L01B3 | ;912A OR |
| CAF4 | inca | | ;912B |
| CAF5 | staa | L01B3 | ; |
| CAF8 | cmpa | 0x0F, x | ;LOAD TIMER |
| CAFA | bcs | LCB1D | ;INCREMENT |
| CAFC | brclr | *L002A, #0x04, LCB0D | ;STORE |
| CB00 | ldaa | L01B5 | ;912C OR |
| CB03 | suba | L01B4 | ;912D OR |
| CB06 | bcs | LCB10 | ;912E |
| CB08 | cmpa | L9136 | ;BRANCH IF < CAL, ELSE |
| | | | ;EGR BIT STATUS FLAG |
| | | | ; |
| | | | ; |
| | | | ; |
| | | | ; |

| | | | |
|------|-------------|--------------------|--|
| CB0B | bc | LCB10 | ; |
| CB0D | LCB0D:inc | 0x00,y | ;INCREMENT EGR MALF TIMER/COUNTER |
| CB10 | LCB10:orab | *L002A | ;EGR BIT STATUS FLAG |
| CB12 | stab | *L002A | ;EGR BIT STATUS FLAG |
| CB14 | LCB14:bclr | *L002A,#0x14 | ;EGR BIT STATUS FLAG |
| CB17 | clr | L01B1 | ;CLEAR TIMER |
| CB1A | clr | L01B4 | ; |
| CB1D | LCB1D:pulb | | ;GET EGR DC FROM STACK |
| CB1E | stab | L01AE | ;EGR DUTY CYCLE |
| CB21 | clra | | ;CLEAR A REG |
| CB22 | ldx | #0xD000 | ;0 DUTY CYCLE |
| CB25 | pshx | | ;SAVE ON STACK |
| CB26 | lslb | | ;SHIFT BITS |
| CB27 | bcc | LCB2E | ;BRANCH IF NO OVERFLOW, ELSE |
| CB29 | oraa | #0x08 | ;SET BIT 3 |
| CB2B | ldx | #0xDFFF | ;MAX DUTY CYCLE |
| CB2E | LCB2E:stx | L3FCC | ;STORE DEGR SOLN C PWM OUTPUT |
| CB31 | lslb | | ;SHIFT BITS |
| CB32 | pulx | | ;GET D000 |
| CB33 | bcc | LCB3A | ;BRANCH IF |
| CB35 | oraa | #0x04 | ;SET BIT 2 |
| CB37 | ldx | #0xDFFF | ;MAX DUTY CYCLE |
| CB3A | LCB3A:stx | L3FEA | ;STORE DEGR SOLN B PWM OUTPUT |
| CB3D | ldx | #0xD000 | ;0 DUTY CYCLE |
| CB40 | lslb | | ;SHIFT BITS |
| CB41 | bcc | LCB48 | ;BRANCH IF NO OVERFLOW, ELSE |
| CB43 | oraa | #0x02 | ;SET BIT 1 |
| CB45 | ldx | #0xDFFF | ;MAX DUTY CYCLE |
| CB48 | LCB48:stx | L3DD2 | ;STORE DEGR SOLN A PWM OUTPUT |
| CB4B | staa | *L00A4 | ;STORE EGR SOLN STATUS FLAG |
| CB4D | jmp | LCB9D | |
| CB50 | LCB50:ldaa | L800C | ;OPTION WORD - TIS SET |
| CB53 | beq | LCB71 | ;BRANCH IF = ZERO, ELSE |
| CB55 | bclr | *L0097,#0x01 | ;CLEAR SMC ENGAGED |
| CB58 | ldd | L3DC8 | ; |
| CB5B | subd | L01D9 | ; |
| CB5E | cpd | L9419 | ; |
| CB62 | bls | LCB67 | ;BRANCH IF <= CAL, ELSE |
| CB64 | bset | *L0097,#0x01 | ;SET SMC ENGAGED |
| CB67 | LCB67:ldd | L3DC8 | ; |
| CB6A | pshy | | ;DELAY |
| CB6C | puly | | ; |
| CB6E | std | L01D9 | ;SAVE FOR NEXT PASS |
| CB71 | LCB71:bclr | *L0096,#0x20 | ;CLEAR PSPS CRAMP |
| CB74 | ldd | L3DCA | ;PSPS INPUT |
| CB77 | subd | L01DB | ;SUBTRACT PREV INPUT |
| CB7A | cpd | L8DB5 | ;32 |
| CB7E | bhi | LCB83 | ;BRANCH IF > 32, ELSE |
| CB80 | bset | *L0096,#0x20 | ;SET PSPS CRAMP |
| CB83 | LCB83:ldd | L3DCA | ;LOAD PSPS INPUT |
| CB86 | pshy | | ;DELAY |
| CB88 | puly | | ; |
| CB8A | std | L01DB | ;STORE PREV INPUT |
| CB8D | tst | L8009 | ;OPTION WORD - A/C PRESSURE SENSOR PRESENT |
| CB90 | bne | LCB9A | ;BRANCH IF NOT Z, ELSE (TIS SET) |
| CB92 | ldaa | #0xA0 | ;SEL CH #10 - A/C PRESSURE SENSOR |
| CB94 | jsr | L9A18 | ;A/D READ |
| CB97 | staa | L0316 | ;STORE A/D READ |
| CB9A | LCB9A: jmp | LCCA6 | ; |
| CB9D | LCB9D:brclr | *L0011,#0x06,LCBA4 | ;MINOR LOOP COUNTER 50 MSEC |
| CBA1 | jmp | LCCA2 | ; |

| Address | Instruction | Comment | Operation |
|---------|--------------------------------|---------|--|
| CBA4 | ldab *L00AE | | ;BLM CELL POINTER |
| CBA6 | ldx #0x8A49 | | ;TABLE ADDRESS |
| CBA9 | abx | | ;ADD TO ADDRESS |
| CBAA | ldab 0x00,x | | ;GET A VALUE FROM THE TABLE |
| CBAC | ldx #0x0034 | | ;INDEX FOR BLM CELLS |
| CBAF | ldaa *L00F4 | | ;CCP DUTY CYCLE |
| CBB1 | bne LCBB6 | | ;BRANCH IF NOT Z, ELSE |
| CBB3 | ldx #0x0045 | | ;INDEX FOR BLM CELLS WITH INACTIVE CCP |
| CBB6 | LCBB6:pshx | | ;SAVE CELL ADDRESS ON STACK |
| CBB7 | puly | | ;GET CELL ADDRESS INTO Y REG |
| CBB9 | abx | | ;ADD TABLE LOOKUP TO ADDRESS |
| CBBA | ldab *L00AE | | ;BLM CELL |
| CBBC | aby | | ;ADD TO ADDRESS |
| CBBE | ldaa 0x00,y | | ;GET VALUE |
| CBC1 | staa 0x00,x | | ;STORE NEW BLM |
| CBC3 | brset *L008F,#0xC0,LCBCA | | ;BRANCH IF IDLE CONDITIONS, ELSE |
| CBC7 | bclr *L0090,#0x80 | | ;CLEAR BLM < MIN WITH ACTIVE CCP |
| CBCA | LCBCA:brclr *L009C,#0x02,LCC19 | | ;BRANCH IF LEARN DISABLED, ELSE |
| CBCE | brset *L002A,#0x10,LCC19 | | ;EGR BIT STATUS FLAG |
| CBD2 | brset *L009B,#0x08,LCC19 | | ;BRANCH IF DELAY BLM UPDATE SET, ELSE |
| CBD6 | ldaa *L00B3 | | ;LOAD INTEGRATOR |
| CBD8 | cmpa #0x80 | | ;128 |
| CBDA | beq LCC19 | | ;BRANCH IF 128, ELSE |
| CBDC | ldab L0197 | | ;BLM UPDATE TIMER |
| CBDF | incb | | ;INCREMENT |
| CBE0 | bne LCBE3 | | ;BRANCH IF NOT Z, ELSE |
| CBE2 | dec b | | ;DECREMENT |
| CBE3 | LCBE3:stab L0197 | | ;STORE TIMER |
| CBE6 | cmpb *L00B1 | | ;COMPARE BLM UPDATE DELAY TIME |
| CBE8 | bcc LCC1C | | ;BRANCH IF >= L00B1, ELSE |
| CBEA | brset *L008F,#0x80,LCC16 | | ;BRANCH IF CCP DC > 0, ELSE |
| CBEE | ldy #0x8A31 | | ;INDEX |
| CBF2 | brclr *L0015,#0x10,LCBF8 | | ;BRANCH IF NO MAF SENSOR MALF, ELSE |
| CBF6 | iny | | ;8A32 |
| CBF8 | LCBF8:ldx #0x0034 | | ;INDEX FOR BLM CELLS |
| CBFB | ldab 0x00,x | | ;CELL 0 BLM |
| CBFD | cmpb 0x00,y | | ;8A31 MIN BLM OR |
| | | | ;8A32 MIN BLM |
| CC00 | bcc LCC07 | | ;BRANCH IF >= MIN BLM, ELSE |
| CC02 | ldab 0x00,y | | ;8A31 LOAD MIN BLM OR |
| | | | ;8A32 LOAD MIN BLM |
| CC05 | stab 0x00,x | | ;STORE CELL 0 BLM |
| CC07 | LCC07:ldab #0x10 | | ;LOAD 16 |
| CC09 | abx | | ;ADD TO X |
| CC0A | ldab 0x00,x | | ;? CELL 0 SAM |
| CC0C | cmpb 0x00,y | | ;8A31 |
| CC0F | bcc LCC16 | | ;BRANCH IF >= MIN |
| CC11 | ldab 0x00,y | | ;8A31 |
| CC14 | stab 0x00,x | | ;004A |
| CC16 | LCC16:jmp LCC9A | | ;SKIP THE REST OF THE BLM ROUTINE |
| CC19 | LCC19:jmp LCC93 | | ;SKIP THE REST OF THE BLM ROUTINE |
| CC1C | LCC1C:ldab L0196 | | ;BLM UPDATE TIMER |
| CC1F | incb | | ;INCREMENT TIMER |
| CC20 | bne LCC23 | | ;BRANCH IF NOT Z, ELSE |
| CC22 | dec b | | ;DECREMENT TIMER |
| CC23 | LCC23:stab L0196 | | ;STORE TIMER |
| CC26 | cmpb L8A1F | | ;COMPARE MIN TIME BETWEEN BLM UPDATES |
| CC29 | bcs LCC9A | | ;BRANCH IF < MIN TIME, ELSE DO UPDATE |
| CC2B | ldx #0x0034 | | ;INDEX FOR BLM CELLS |
| CC2E | ldab *L00F4 | | ;CCP DUTY CYCLE |
| CC30 | bne LCC35 | | ;BRANCH IF NOT Z, ELSE |
| CC32 | ldx #0x0045 | | ;INDEX FOR BLM CELLS WITH INACTIVE CC |

| | | | |
|------|-------------|--------------------|--|
| CC35 | LCC35:ldab | *L00AE | ;BLM CELL |
| CC37 | abx | | ;ADD TO X |
| CC38 | ldab | 0x00,x | ;GET BLM |
| CC3A | suba | #0x80 | ;SUBTRACT 128 |
| CC3C | bcs | LCC60 | ;BRANCH IF BLM < 128, ELSE |
| CC3E | cmpa | L8A2C | ;0x05 |
| CC41 | bls | LCC9A | ;BRANCH IF BLM-128 < 5, ELSE |
| CC43 | brset | *L009C,#0x40,LCC9A | ;BRANCH IF RICH FLAG SET, ELSE |
| CC47 | ldy | #0x8A2F | ;INDEX |
| CC4B | brclr | *L0015,#0x10,LCC51 | ;BRANCH IF NO MAF SENSOR MALF, ELSE |
| CC4F | iny | | ;8A30 |
| CC51 | LCC51:addb | L8A2E | ;0x01 |
| CC54 | bcs | LCC5B | ;BRANCH IF OVERFLOW, ELSE |
| CC56 | cmpb | 0x00,y | ;8A2F MAX BLM |
| | | | ;8A30 MAX BLM IF MAF MALF SET |
| CC59 | bls | LCC91 | ;BRANCH IF <= MAX BLM, ELSE |
| CC5B | LCC5B:ldab | 0x00,y | ;8A2F LOAD MAX BLM OR |
| | | | ;8A30 LOAD MAX BLM IF MAF MALF SET |
| CC5E | bra | LCC91 | ; |
| | | | ;INVERT BLM-128 |
| CC60 | LCC60:nega | | ;0x05 |
| CC61 | cmpa | L8A2D | ;BRANCH IF BLM-128 <= 0x05, ELSE |
| CC64 | bls | LCC9A | ;BRANCH IF LEAN FLAG SET, ELSE |
| CC66 | brclr | *L009C,#0x40,LCC9A | ;0x01 BLM HYSTERESIS |
| CC6A | subb | L8A2E | ;BRANCH IF UNDERFLOW, ELSE |
| CC6D | bcs | LCC8E | ;INDEX |
| CC6F | ldy | #0x8A31 | ;BRANCH IF NO MAF SENSOR MALF, ELSE |
| CC73 | brclr | *L0015,#0x10,LCC79 | ;8A32 |
| CC77 | iny | | ;BRANCH IF CCP DC Z, ELSE |
| CC79 | LCC79:brclr | *L008F,#0x80,LCC89 | ;8A31 MIN BLM OR |
| CC7D | cmpb | 0x00,y | ;8A32 MIN BLM |
| | | | ;BRANCH IF BLM >= MIN, ELSE |
| CC80 | bcc | LCC91 | ;SET BIT 7 - BLM < MIN |
| CC82 | bset | *L0090,#0x80 | ; |
| CC85 | ldy | #0x8A33 | ;8A31 OR |
| CC89 | LCC89:cmpb | 0x00,y | ;8A32 OR |
| | | | ;8A33 |
| | | | ;BRANCH IF > CAL, ELSE |
| CC8C | bcc | LCC91 | ;8A33 LOAD MIN BLM |
| CC8E | LCC8E:ldab | 0x00,y | ;STORE NEW BLM |
| CC91 | LCC91:stab | 0x00,x | ;RESET BLM UPDATE TIMER |
| CC93 | LCC93:clrb | | ; |
| CC94 | stab | L0197 | ;STORE TIMER |
| CC97 | stab | L0196 | ;CLEAR DELAY BLM UPDATE |
| CC9A | LCC9A:bclr | *L009B,#0x08 | ;LOAD RPM/12.5 |
| CC9D | ldaa | *L00AD | ;SAVE FOR NEXT PASS |
| CC9F | staa | L012C | ;CLEAR |
| CCA2 | LCCA2:clra | | ;RESET COP WATCHDOG |
| CCA3 | staa | L400C | ;MINOR LOOP COUNTER |
| CCA6 | LCCA6:ldab | *L0011 | ;MASK FOR LOWER NIBBLE |
| CCA8 | andb | #0x0F | ;SEGMENT TABLE ADDRESS |
| CCAA | ldx | #0xCCE1 | ; |
| CCAD | lslb | | ;ADD TO ADDRESS |
| CCAE | abx | | ;GET TABLE ADDRESS |
| CCAF | ldx | 0x00,x | ;DO THAT ROUTINE |
| CCB1 | jsr | 0x00,x | ;CLEAR TIMING ERROR FLAG |
| CCB3 | bclr | *L0086,#0x08 | ;LOAD TIME OF LAST 6.25 MSEC INTERRUPT |
| CCB6 | ldaa | L026A | ;SUBTRACT CURRENT TIME |
| CCB9 | suba | L4005 | ;MINOR LOOP COUNTER |
| CCBC | ldab | *L0011 | ;MASK FOR LOWER NIBBLE |
| CCBE | andb | #0x0F | ;RAM ADDRESS |
| CCC0 | ldx | #0x0000 | ;ADD TO ADDRESS |
| CCC3 | abx | | ;COMPARE |
| CCC4 | cmpa | 0x00,x | |

```

CCC6      bcc      LCCCA      ;BRANCH IF >
CCC8      staa     0x00,x      ;
CCCA      LCCCA:ldx    L0176      ;
CCCD      beq      LCCDC      ;BRANCH IF Z, ELSE
CCCF      LCCCF:dex      ;DECREMENT
CCD0      dex      ;
CCD1      LCCD1:sei      ;HOLD INTERRUPTS
CCD2      ldd      0x00,x      ;
CCD4      std      0x00,x      ;
CCD6      cli      ;CLEAR AND ALLOW INTERRUPTS
CCD7      stx      L0176      ;STORE
CCDA      bne      LCCCF      ;LOOP TILL DONE
CCDC      LCCDC:ldx    #0x0334      ;
CCDF      bra      LCCD1

;-----
; MAJOR SEGMENT TABLE
;-----
CCE1      0xD3,0xA9      ;O2 READY AND CLOSED LOOP
CCE3      0xD6,0x99      ;CCP SUBROUTINE AND EXERCISE PWM OUTPUTS
CCE5      0xD9,0x58      ;A.I.R. AND ??
CCE7      0xDA,0xA4      ;MISC STUFF AND UPDATE ALDL STATUS WORDS
CCE9      0xDC,0x18      ;UPDATE HUD DISPLAY
CCEB      0xDD,0x28      ;F31 OR SHIFT LIGHT
CCED      0xE2,0xE3      ;COOLANT TEMP LOOKUP AND MALF CHECK
CCEF      0xDC,0x29      ;UPDATE HUD DISPLAY AND TCC PWM
CCF1      0xE4,0xFF      ;ESC & MALF CHECK, A/C PRESS & MALF CHECK
CCF3      0xE5,0x80      ;IATMAT & MALF CHECK AND PKII
CCF5      0xE6,0x98      ;BATT VOLTS AND MALF CHECK
CCF7      0xE3,0xC5      ;TRANS TEMP LOOKUP AND MALF CHECK
CCF9      0xE7,0xE8      ;DIAGNOSTICS
CCFB      0xED,0x9F      ;DIAGNOSTICS
CCFD      0xE6,0xFD      ;CRUISE CONTROL AND VARIABLE LOOKUP
CCFF      0xF3,0x44      ;FATI, FATC AND PE FAR VARIABLE LOOKUP
CD01      rts

;-----
; EXTERNAL IRQ VECTOR POINTS HERE
; MAJOR FUEL OUT ROUTINE
;-----
CD02      brclr    *L0085,#0x10,LCD1B      ;BRANCH IF NOT MULTI SENSOR FAILURE, ELSE
CD06      inc      L0087      ;INCREMENT ? MISFIRE LOGIC LOOP COUNTER
CD09      ldx      #0x4002      ;CPU DATA LATCH
CD0C      bset     0x00,x,#0x80      ;SET BIT 7
CD0F      bclr     0x00,x,#0x80      ;CLEAR BIT 7
CD12      ldx      #0x4000      ;INDEX I/O DATA REG ADDRESS
CD15      jsr      L9A58      ;WR L0089 TO I/O PORT
CD18      staa     *L008B      ;STORE DISCRETE INPUT STATES
CD1A      rti      ;RETURN

;-----
; BRANCH HERE IF BIT 4 L0085 CLEAR UPON XIRQ
; BIT 4 = MULTI SENSOR FAILURE
;-----
CD1B      LCD1B:ldaa  L030B      ;2S COMP L0123 - TABLE LOOKUP RESULT
CD1E      brset    *L008F,#0x40,LCD2A      ;BRANCH IF IDLE CONDITIONS MET, ELSE
CD22      ldab     *L00B7      ;CYL COUNTER
CD24      ldx      #0x8403      ;ALL 00S
CD27      abx      ;
CD28      adda     0x00,x      ;ADD VALUE FROM THE TABLE
CD2A      LCD2A:staa  *L00CA      ;STORE
CD2C      nop      ;DELAY
CD2D      ldd      L3FC0      ;LOAD REF PERIOD
CD30      std      *L00C6      ;STORE PREV REF PERIOD

```

```

CD32      brset  *L0013,#0x08,LCD3E      ;BRANCH IF 24X CRANK SIGNAL ERROR, ELSE
CD36      ldaa  L0131                    ;24 X PULSE COUNTER
CD39      cmpa  L9112                    ;150
CD3C      bcs   LCD46                    ;BRANCH IF < 150, ELSE
CD3E      LCD3E:ldx  #0x4002              ;CPU DATA LATCH
CD41      bset  0x00,x,#0x80             ;SET BIT 7
CD44      bra   LCD54                    ;

CD46      LCD46:ldab  *L0087              ;? MISFIRE LOGIC LOOP COUNTER
CD48      bne   LCD51                    ;BRANCH IF NOT Z, ELSE
CD4A      ldx   *L00C8                    ;3/4 3x REF PERIOD
CD4C      cpx   L81EF                    ;819
CD4F      bls   LCD54                    ;BRANCH IF <= 819, ELSE
CD51      LCD51:jmp   LCD54              ;DO MINOR SEGMENT TABLE ROUTINE

;-----
;
;-----
CD54      LCD54:ldd   *L00C6              ;PREV REF PERIOD
CD56      lsr    ;DIV BY 2
CD57      addd   *L00C6                  ;ADD PREV REF PERIOD
CD59      ldx   L3FEC                    ;BCNTR, LAST REF
CD5C      stx   *L00CB                  ;STORE LAST REF
CD5E      lsr    ;DIV BY 2
CD5F      std   *L00C8                  ;3/4 3x A REF PERIOD
CD61      stx   L3FE4                    ;B CNTR, START NEXT DWELL
CD64      pshb   ;SAVE REF PER LSB
CD65      ldaa  *L00B7                  ;LOAD CYL COUNTER
CD67      inca   ;INCREMENT COUNTER
CD68      cmpa  #0x07                    ;7 ?
CD6A      bne   LCD6E                    ;BRANCH IF NOT 7, ELSE
CD6C      ldaa  #0x01                    ;LOAD 1
CD6E      LCD6E:staa *L00B7              ;STORE CYL COUNTER
CD70      bset  *L0097,#0x02             ;SET BIT 1
CD73      ldd   *L00CF                  ;MAF (GMS/SEC)
CD75      addd   L025C                    ;ADD RPM + IAT CORRECTION TERM
CD78      bcc   LCD7D                    ;BRANCH IF NO OVERFLOW, ELSE
CD7A      ldd   #0xFFFF                  ;LOAD MAX
CD7D      LCD7D:std  *L00D1              ;STORE CALCULATED MAF
CD7F      mul   ;DELAY
CD80      psha   ;
CD81      pula   ;
CD82      brclr  *L0097,#0x80,LCD95      ;BRANCH IF RUN-FUEL NOT ALLOWED, ELSE
CD86      brclr  *L009A,#0x02,LCD98      ;BRANCH IF NOT DE, ELSE
CD8A      ldaa  L0100                    ;ACCEL ENRICHMENT TIMER
CD8D      inca   ;INCREMENT TIMER
CD8E      beq   LCD9A                    ;BRANCH IF Z, ELSE
CD90      staa  L0100                    ;STORE A.E. TIMER
CD93      bra   LCD9C                    ;

CD95      LCD95:nop                      ;DELAY
CD96      nop
CD97      nop
CD98      LCD98:pshx
CD99      pulx
CD9A      LCD9A:psha
CD9B      pula
CD9C      LCD9C:mul
CD9D      mul
CD9E      nop
CD9F      pulb                          ;GET BACK REF PER LSB
CDA0      ldaa  *L00CA                  ;
CDA2      mul   ;A * B
CDA3      adca  #0x00                  ;ROUND

```

```

CDA5      staa    *L00E3                ;STORE MSB
CDA7      ldaa    *L00C8                ;3/4 3x A REF PERIOD
CDA9      ldab    *L00CA                ;
CDAB      mul     ;L00C8*L00CA
CDAC      addb    *L00E3                ;ADD L00E3
CDAE      adca    #0x00                ;ROUND
CDB0      addd    *L00CB                ;ADD L00CB
CDB2      subd    L81C9                ;0x00
CDB5      std     L3FE2                ;
CDB8      jmp     LCFB8                ;SKIP SEGMENT TABLE; DO MAF ROUTINE

;-----
;
; B CONTAINS L0087 - ? MISFIRE LOGIC LOOP COUNTER
;-----
CDBB      LCDBB:ldaa    L030B                ;2S COMP L0123 - TABLE LOOKUP RESULT
CDBE      brset   *L008F,#0x40,LCDD3        ;BRANCH IF IDLE CONDITIONS MET, ELSE
CDC2      pshb                    ;SAVE LOOP COUNTER
CDC3      ldab    *L00B7                ;LOAD CYL COUNTER
CDC5      cmpb    #0x01                ;1 ?
CDC7      bne     LCDCB                ;BRANCH IF NOT 1, ELSE
CDC9      ldab    #0x07                ;LOAD 7
CDCB      LCDCB:decb                    ;DECREMENT CYL COUNTER
CDCC      ldx     #0x8403                ;ADDRESS
CDCF      abx                    ;ADD TO ADDRESS
CDD0      pulb                    ;GET BACK LOOP COUNTER
CDD1      adda    0x00,x                ;ADD VALUE FROM TABLE (ALL 0s)
CDD3      LCDD3:staa    *L00CA                ;STORE
CDD5      ldx     #0xCDE1                ;SEGMENT TABLE ADDRESS
CDD8      inc     L0087                ;INCREMENT LOOP COUNTER
Cddb      lslb                    ;MUL LOOP COUNTER BY 2
CDDC      abx                    ;ADD TO ADDRESS
CDDD      ldx     0x00,x                ;LOAD ADDRESS
CDDF      jmp     0x00,x                ;JUMP TO THAT ADDRESS

;-----
; MINOR SEGMENT TABLE - SCHEDULING IS BASED ON L0087
;-----
CDE1      CF57                ;CALCULATE MAF
          CE35                ;
          CE62                ;
          CE89                ;
          CEB3                ;
          CED5                ;
          CEFE                ;
          CF12                ;

CDF1      LCDF1:psha                    ;SAVE A ON STACK
          ;(L00CA - L81F1, L81F2, L81F3, OR L81F4)
CDF2      bset    0x00,x,#0x80          ;X = 4002
CDF5      bclr    0x00,x,#0x80          ;
CDF8      ldd     L3FC4                ;A REF PERIOD ? 24 X
CDFB      subd    *L00C4                ;SUBTRACT PREV REF PERIOD
CDFD      brclr   *L0094,#0x08,LCE11    ;BRANCH IF NOT BIT 3, ELSE
CE01      ldd     L3FC0                ;LOAD REF PERIOD
CE04      std     *L00C6                ;STORE PREV REF PERIOD
CE06      lsr     ;DIV BY 2
CE07      addd    *L00C6                ;ADD PREV REF PERIOD
CE09      lsr     ;DIV BY 2
CE0A      std     *L00C8                ;3/4 3x A REF PERIOD
CE0C      bclr    *L0094,#0x08          ;CLEAR BIT 3
CE0F      bra     LCE19                ;
CE11      LCE11:std     *L00C8                ;3/4 3x REF PERIOD

```

```

CE13      lslld                      ;MUL BY 2
CE14      addd      *L00C8            ;ADD 3/4 3x REF PERIOD
CE16      lslld                      ;MUL BY 2
CE17      std      *L00C8            ;3/4 3x REF PERIOD
CE19      LCE19:ldx      L3FC4        ;A REF PERIOD
CE1C      stx      *L00C4            ;PREV REF PERIOD
CE1E      pula                      ;DELAY
CE1F      psha                      ;
CE20      mul                      ;A * L00C8 LSB
CE21      adca      #0x00            ;ROUND
CE23      staa      *L00E3            ;STORE
CE25      pulb                      ;GET INTO B REG
CE26      ldaa      *L00C8            ;3/4 3x REF PERIOD
CE28      mul                      ;L00C8 * B
CE29      addb      *L00E3            ;ADD L00E3
CE2B      adca      #0x00            ;ROUND
CE2D      addd      *L00C4            ;ADD PREV REF PERIOD
CE2F      std      L3FE2            ;STORE TIMER OUTPUT
CE32      jmp      LD333            ;RETURN

; -----
;  BRANCH HERE FROM MINOR SEGMENT TABLE
;
; -----

CE35      ldx      #0x4002            ;CPU DATA LATCH
CE38      ldd      L3FC4            ;A REF PERIOD
CE3B      subd      *L00C4            ;SUB PREV REF PERIOD
CE3D      std      L02CE            ;STORE DELTA REF PERIOD
CE40      ldd      L3FEC            ;? LAST REF
CE43      cpd      *L00C4            ;COMPARE PREV REF PERIOD
CE46      beq      LCE4B            ;BRANCH IF THEY'RE EQUAL, ELSE
CE48      jmp      LCF3A            ;

CE4B      LCE4B:ldaa      *L00CA            ;
CE4D      cmpa      L81F2            ;76
CE50      bls      LCE55            ;BRANCH IF <= CAL, ELSE
CE52      jmp      LCEF1            ;

CE55      LCE55:suba      L81F1            ;33
CE58      bcs      LCE5F            ;BRANCH IF UNDERFLOW, ELSE
CE5A      bset      *L0094,#0x08        ;SET BIT 3
CE5D      bra      LCDF1            ;

CE5F      LCE5F:jmp      LCEE6

; -----
;  BRANCH HERE FROM MINOR SEGMENT TABLE
;
; -----

CE62      ldx      #0x4002            ;CPU DATA LATCH
CE65      ldd      L3FC4            ;
CE68      subd      *L00C4            ;
CE6A      std      L02D0            ;STORE DELTA REF PERIOD
CE6D      bne      LCE72            ;
CE6F      jmp      LCF3A

CE72      LCE72:brset      *L0098,#0x40,LCEF1      ;
CE76      ldaa      *L00CA            ;
CE78      cmpa      L81F3            ;119
CE7B      bhi      LCEF1            ;
CE7D      suba      L81F2            ;76
CE80      bcs      LCEE6
CE82      brset      *L0086,#0x80,LCE86      ;BRANCH IF ENGINE RUNNING, ELSE
CE86      LCE86:jmp      LCDF1

```

```

;-----
; BRANCH HERE FROM MINOR SEGMENT TABLE
;
;-----
CE89      ldx      #0x4002                      ;CPU DATA LATCH
CE8C      ldd      L3FC4
CE8F      subd     *L00C4
CE91      std      L02D2                      ;STORE DELTA REF PERIOD
CE94      bne      LCE99
CE96      jmp      LCF3A

CE99      LCE99:brset *L0098,#0x40,LCEF1      ;
CE9D      ldaa     *L00CA
CE9F      cmpa     L81F4                      ;
CEA2      bhi      LCEF1
CEA4      suba     L81F3                      ;
CEA7      bcs      LCEE6                      ;
CEA9      brset    *L0086,#0x80,LCEB0          ;BRANCH IF ENGINE RUNNING, ELSE
CEAD      nop
CEAE      nop
CEAF      nop
CEB0      LCEB0: jmp      LCDF1

;-----
; BRANCH HERE FROM MINOR SEGMENT TABLE
;
;-----
CEB3      ldx      #0x4002                      ;CPU DATA LATCH
CEB6      ldd      L3FC4
CEB9      subd     *L00C4
CEBB      std      L02D4                      ;STORE DELTA REF PERIOD
CEBE      beq      LCF3A
CEC0      brset    *L0098,#0x40,LCEF1          ;
CEC4      ldaa     *L00CA
CEC6      suba     L81F4
CEC9      bcs      LCEE6
CECB      brset    *L0086,#0x80,LCED2          ;BRANCH IF ENGINE RUNNING, ELSE
CECF      nop
CED0      nop
CED1      nop
CED2      LCED2: jmp      LCDF1

;-----
; BRANCH HERE FROM MINOR SEGMENT TABLE
;
;-----
CED5      ldx      #0x4002                      ;CPU DATA LATCH
CED8      ldd      L3FC4
CEDB      subd     *L00C4
CEDD      std      L02D6                      ;STORE DELTA REF PERIOD
CEE0      beq      LCF3A
CEE2      brset    *L0098,#0x40,LCEF1          ;
CEE6      LCEE6: ldd      L3FC4
CEE9      nop
CEEA      nop
CEEB      bset     *L0098,#0x40                ;
EEEE      std      L3FE2                      ;STORE TIMER OUTPUT
CEF1      LCEF1: bset    0x00,x,#0x80
CEF4      bclr     0x00,x,#0x80
CEF7      ldd      L3FC4                      ;LOAD REF PERIOD
CEFA      std      *L00C4                    ;SAVE FOR NEXT PASS
CEFC      bra      LCF30

```

```

;-----
; BRANCH HERE FROM MINOR SEGMENT TABLE
;
;-----
CEFE      ldx      #0x4002                ;CPU DATA LATCH
CF01      ldd      L3FC4
CF04      subd     *L00C4                ;
CF06      std      L02D8                ;STORE DELTA REF PERIOD
CF09      beq      LCF3A                ;BRANCH IF Z, ELSE
CF0B      ldd      L3FC4                ;LOAD REF PERIOD
CF0E      std      *L00C4                ;SAVE FOR NEXT PASS
CF10      bra      LCEF1                ;

;-----
; BRANCH HERE FROM MINOR SEGMENT TABLE
;
;-----
CF12      ldx      #0x4002                ;CPU DATA LATCH
CF15      ldd      L3FC4                ;REF PERIOD
CF18      subd     *L00C4                ;SUB PREV REF PERIOD
CF1A      std      L02DA                ;STORE DELTA REF PERIOD
CF1D      beq      LCF3A                ;BRANCH IF Z, ELSE
CF1F      ldd      L3FC4                ;REF PERIOD
CF22      std      *L00C4                ;SAVE FOR NEXT PASS
CF24      bset     0x00,x,#0x80          ;SET BIT 7, I/O PORT
CF27      clr      L0087                ;CLEAR LOOP COUNTER
CF2A      bclr     *L0098,#0x40          ;CLEAR BIT 6
CF2D      nop
CF2E      nop
CF2F      nop
CF30      LCF30:nop
CF31      nop
CF32      nop
CF33      nop
CF34      nop
CF35      nop
CF36      mul
CF37      jmp      LD333                ;RETURN

CF3A      LCF3A:bset 0x00,x,#0x80          ;4002 SET BIT 7
CF3D      ldab     L0131                ;LOAD 24X PULSE COUNTER
CF40      cmpb     L9112                ;150
CF43      bcc      LCF4F                ;BRANCH IF >= 150
CF45      inc      L0131                ;INCREMENT PULSE COUNTER
CF48      bclr     0x00,x,#0x80          ;4002 CLEAR BIT 7
CF4B      ldaa     #0x01                ;LOAD 1
CF4D      staa     *L0087                ;STORE LOOP COUNTER
CF4F      LCF4F:ldd  L3FEC                ;? LAST REF
CF52      std      *L00C4                ;SAVE FOR NEXT PASS
CF54      jmp      LCD54                ;

;-----
; BRANCH HERE FROM MINOR SEGMENT TABLE
;
;-----
CF57      ldx      #0x4002                ;CPU DATA LATCH
CF5A      ldd      L3FEC                ;? LAST REF
CF5D      subd     *L00C4                ;SUBTRACT PREV REF
CF5F      std      L02DC                ;STORE DELTA REF
CF62      ldd      L3FEC                ;? LAST REF
CF65      std      *L00C4                ;STORE FOR NEXT PASS
CF67      bclr     0x00,x,#0x80          ;CLEAR BIT 7 I/O PORT
CF6A      std      L3FE4                ;STORE BCNTR, START NEXT DWELL
CF6D      ldaa     *L00CA                ;A REF PERIOD VARIABLE

```

```

CF6F      cmpa    L81F1      ;
CF72      bhi     LCF90      ;BRANCH IF > 81F1, ELSE
CF74      mul
CF75      mul
CF76      mul
CF77      mul
CF78      ldd     L3FC0      ;LOAD REF PERIOD
CF7B      std     *L00C6     ;STORE PREV REF PERIOD
CF7D      lsrd
CF7E      addd    *L00C6     ;ADD REF PERIOD
CF80      lsrd
CF81      std     *L00C8     ;DIV BY 2
CF83      ldx     #0x00C8    ;3/4 3x REF PERIOD
CF86      ldaa    *L00CA     ;GET INTO X REG
CF88      jsr     L98B9      ;
CF8B      addd    *L00C4     ;8 x 16 MULTIPLY
CF8D      std     L3FE2     ;ADD
CF90      LCF90:ldaa *L00B7   ;STORE TIMER OUTPUT
CF92      inca
CF93      cmpa    #0x07      ;LOAD CYL COUNTER
CF95      bne     LCF99      ;INCREMENT
CF97      ldaa    #0x01      ;COMPARE
CF99      LCF99:staa *L00B7   ;BRANCH IF NOT = 7, ELSE
CF9B      bset    *L0097,#0x02 ;LOAD 1
CF9E      ldd     *L00CF     ;STORE CYL COUNTER
CFA0      addd    L025C      ;
CFA3      bcc     LCFA8      ;LOAD MAF (GM/SEC)
CFA5      ldd     #0xFFFF    ;ADD RPM AND IAT CORRECTION TERM
CFA8      LCFAB:std *L00D1    ;BRANCH IF NO OVERFLOW, ELSE
CFAA      brclr   *L0097,#0x80,LCFBB ;LOAD MAX
CFAE      brclr   *L009A,#0x02,LCFBB ;SAVE MAF
CFB2      ldaa    L0100      ;BRANCH IF RUN-FUEL NOT ALLOWED, ELSE
CFB5      inca
CFB6      beq     LCFBB      ;BRANCH IF ACCEL ENRICH NOT ENABLED, ELSE
CFB8      staa    L0100      ;ACCEL ENRICHMENT TIMER
CFBB      LCFBB:ldx L3FC6     ;INCREMENT
CFBE      brclr   *L009A,#0x04,LCFCB ;BRANCH IF Z, ELSE
CFC2      ldaa    L0101      ;STORE TIMER
CFC5      inca
CFC6      beq     LCFCB      ;LOAD PULSE COUNTER
CFC8      staa    L0101      ;BRANCH IF DECEL ENLEAN NOT ENABLED, ELSE
CFCB      LCFCB:brclr *L00DA,#0x40,LCFD8 ;LOAD D.E. TIMER
CFCF      ldaa    L017F      ;INCREMENT
CFD2      inca
CFD3      beq     LCFD8      ;BRANCH IF Z, ELSE
CFD5      staa    L017F      ;SAVE D.E. TIMER
CFD8      LCFD8:ldy L3FF8     ;BRANCH IF NOT BIT 6, ELSE
CFDC      ldd     *L00D3     ;LOAD COUNTER
CFDE      nega
CFDF      negb
CFE0      sbca    #0x00      ;INCREMENT COUNTER
CFE2      cpx     L3FC6      ;BRANCH IF THEY'RE EQUAL, ELSE
CFE5      beq     LCFEC      ;INCREMENT X
CFE7      inx
CFE8      ldy     L3FF8      ;MAF INPUT
CFEC      LCFEC:sty *L00D3    ;MAF PULSE COUNTER
CFEF      addd    *L00D3     ;INVERT A
CFF1      std     *L00D7     ;INVERT B
CFF3      ldd     *L00D5     ;ROUND DOWN
CFF5      nega
CFD8      LCFD8:ldy L3FF8     ;COMPARE PULSE COUNTER
CFDC      ldd     *L00D3     ;BRANCH IF THEY'RE EQUAL, ELSE
CFDE      nega
CFDF      negb
CFE0      sbca    #0x00      ;INCREMENT X
CFE2      cpx     L3FC6      ;MAF INPUT
CFE5      beq     LCFEC      ;STORE MAF PULSE COUNTER
CFE7      inx
CFE8      ldy     L3FF8      ;ADD PREV PULSES
CFEC      LCFEC:sty *L00D3    ;STORE MAF PULSE ACCUMULATOR
CFEF      addd    *L00D3     ;LOAD
CFF1      std     *L00D7     ;INVERT
CFF3      ldd     *L00D5
CFF5      nega

```

```

CFF6          negb          ;INVERT
CFF7          sbca    #0x00    ;ROUND
CFF9          stx    *L00D5    ;STORE PULSE ACCUMULATOR
CFFB          addd    *L00D5    ;ADD PULSES
CFFD          bclr    *L0091,#0x10 ;CLEAR BIT 4 - MAF SIGNAL OKAY
D000          cpd    L90C2      ;3 - MIN MAF PULSES
D004          bcs    LD00F      ;BRANCH IF < 3, ELSE
D006          brclr   *L0084,#0x80,LD020 ;BRANCH IF NOT MODE 4, ELSE
D00A          tst    L0231      ;TEST MODE 4 CONTROL BYTE
D00D          bpl    LD020      ;BRANCH IF BIT 7 CLEAR, ELSE
D00F          LD00F:bset   *L0091,#0x10 ;SET BIT 4 - MAF SIGNAL LOW
D012          tst    L8020      ;OPTION WORD - TIS CLEAR
D015          bne    LD020      ;BRANCH IF NOT ZERO, ELSE
D017          ldd    #0x0000    ;LOAD 0000
D01A          std    L0108      ;MAF (GMS/SEC)
D01D          jmp    LD061      ;JUMP OVER MAF CALCULATION

D020          LD020:ldx    *L00D7    ;MAF PULSE ACCUMULATOR
D022          fdiv          ;D/X
D023          stx    L01A4      ;STORE QUOTIENT - ? DELTA MAF HZ ?
D026          ldd    L01A4      ;GET INTO ACCD
D029          subd    L8BDB      ;SUBTRACT 2000
D02C          bcc    LD031      ;BRANCH IF NO UNDERFLOW, ELSE
D02E          ldd    #0x0000    ;LOAD 0000
D031          LD031:cmpa   #0x1F    ;COMPARE
D033          bls    LD038      ;BRANCH IF <= 0x1F, ELSE
D035          ldd    #0x1FFF    ;LOAD MAX PULSE VALUE 8191d
D038          LD038:lsrd          ;DIV BY 4
D039          lsrd          ;
D03A          pshb          ;SAVE LSB ON STACK
D03B          ldab    #0x0C      ;12
D03D          mul          ;MSB * 12
D03E          ldx    #0x8BDF    ;MAF SCALER TABLE ADDRESS
D041          abx          ;ADD LSB OF PRODUCT TO ADDRESS
D042          ldab    0x00,x    ;GET THE SCALER VALUE
D044          inx          ;INCREMENT ADDRESS
D045          pula          ;GET LSB FROM STACK INTO A REG
D046          lsra          ;DIV BY 2
D047          jsr    L99D7      ;2D LOOKUP
D04A          mul          ;TABLE LOOKUP RESULT * B
D04B          dex          ;DECREMENT ADDRESS
D04C          dex          ;
D04D          dex          ;
D04E          addd    0x00,x    ;ADDITIVE OFFSET TO MAF LOOKUP
D050          staa    L0129      ;STORE LOOKUP RESULT MSB
D053          cpd    *L00D1      ;COMPARE MAX MAF
D056          bcs    LD05A      ;BRANCH IF MEASURED MAF < CORRECTED MAF
D058          ldd    *L00D1      ;LOAD MAX MAF
D05A          LD05A:std    L0108    ;STORE MAF (GM/SEC) - ALDL
D05D          brclr   *L0091,#0x10,LD064 ;BRANCH IF MAF SIGNAL OKAY, ELSE
D061          LD061:ldd    L0256    ;DEFAULT MAF
D064          LD064:std    *L00CF    ;STORE MAF (GMS/SEC)
D066          cpd    L0258      ;MAF FROM RPM AND IAT
D06A          bcs    LD06F      ;BRANCH IF < L0258, ELSE
D06C          ldd    L0258      ;LOAD MAF FROM RPM AND IAT
D06F          LD06F:std    L025A      ;STORE MAF FOR TORQUE CALCULATION

;-----
; DO 16 x 16 MULTIPLY, MAF x REF PERIOD
; RESULT IS MASS AIR, NO TIME COMPONENT
; SAVED IN L00E6
;-----
D072          ldd    *L00CF      ;MAF (GMS/SEC)
D074          lsrd          ;DIV BY 2

```

```

D075      std      *L00E6      ;STORE MAF/2
D077      ldaa     *L00C7      ;LOAD REF PERIOD LSB
D079      mul      ;L00C7 * L00E7
D07A      adca     #0x00      ;ROUND
D07C      ldab     *L00E7      ;LOAD MAF/2 LSB
D07E      staa     *L00E7      ;STORE MSB OF PRODUCT
D080      ldaa     *L00C6      ;REF PER MSB
D082      mul      ;REF PER MSB * MAF/2 LSB
D083      addb     *L00E7      ;ADD PREV MSB
D085      adca     #0x00      ;ROUND
D087      std      *L00E7      ;STORE
D089      ldaa     *L00E6      ;LOAD MAF/2
D08B      ldab     *L00C7      ;REF PER LSB
D08D      mul      ;MAF/2 * REF PER LSB
D08E      addd     *L00E7      ;ADD PREV
D090      bcs      LD0A0      ;BRANCH IF OVERFLOW, ELSE
D092      std      *L00E6      ;STORE MASS AIR
D094      ldaa     *L00CF      ;MAF (GMS/SEC) MSB
D096      lsra     ;DIV BY 2
D097      ldab     *L00C6      ;REF PER MSB
D099      mul      ;REF PER MSB * MAF
D09A      addb     *L00E6      ;ADD PREV
D09C      adca     #0x00      ;ROUND
D09E      beq      LD0A7      ;BRANCH IF Z, ELSE
D0A0      LD0A0: ldd     #0xFFFF ;LOAD MAX
D0A3      std      *L00E6      ;STORE MASS AIR THIS REF PERIOD
D0A5      bra      LD0AB      ;

D0A7      LD0A7: stab    *L00E6      ;STORE LSB
D0A9      ldab     *L00E7      ;GET PREV LSB

;-----
; DO 16 x 16 MULTIPLY, MASS AIR * FUEL TERM
; RESULT IS REQUIRED FUEL
;-----
D0AB      LD0AB: ldaa    L0103      ;FUEL TERM LSB
D0AE      mul      ;FUEL TERM LSB * MASS AIR MSB
D0AF      adca     #0x00      ;ROUND
D0B1      staa     *L00E8      ;STORE MSB OF PRODUCT
D0B3      ldaa     *L00E7      ;LOAD MASS AIR LSB
D0B5      ldab     L0102      ;FUEL TERM MSB
D0B8      mul      ;A * B
D0B9      addb     *L00E8      ;ADD LSB
D0BB      adca     #0x00      ;ROUND
D0BD      std      *L00E7      ;STORE
D0BF      ldaa     *L00E6      ;MASS AIR THIS REF LSB
D0C1      ldab     L0103      ;FUEL TERM LSB
D0C4      mul      ;A * B
D0C5      addd     *L00E7      ;ADD PREV
D0C7      std      *L00E7      ;STORE
D0C9      ldaa     #0x00      ;MAKE MSB 0
D0CB      rla      ;MUL BY 2
D0CC      ldab     *L00E6      ;LOAD LSB
D0CE      staa     *L00E6      ;STORE AS MSB
D0D0      ldaa     L0102      ;LOAD INJ PW MSB
D0D3      mul      ;A * B
D0D4      addd     *L00E6      ;ADD PREV PRODUCT
D0D6      pshb     ;SAVE LSB ON STACK
D0D7      psha     ;SAVE MSB ON STACK
D0D8      tsx      ;X POINTS TO STACK
D0D9      ldaa     L027F      ;CRANK TO RUN PW MULTIPLIER
D0DC      beq      LD0F4      ;BRANCH IF Z, ELSE
D0DE      ldd      L027C      ;CRANKING PW
D0E1      subd     0x00,x      ;SUBTRACT CONTENTS OF STACK

```

| | | | |
|------|-------------|--------------------|--|
| D0E3 | bc | LD0F4 | ;BRANCH IF UNDERFLOW, ELSE |
| D0E5 | pshb | | ;SAVE LSB ON STACK |
| D0E6 | psha | | ;SAVE MSB ON STACK |
| D0E7 | tsx | | ;X POINTS TO STACK |
| D0E8 | ldaa | L027F | ;CRANK TO RUN PW MULTIPLIER |
| D0EB | jsr | L98B9 | ;8 x 16 MULTIPLY |
| D0EE | pulx | | ;GET BACK INTO X REG |
| D0EF | tsx | | ;X POINTS TO STACK |
| D0F0 | add | 0x00,x | ;ADD TO CONTENTS OF STACK |
| D0F2 | bra | LD0F6 | |
| | | | |
| D0F4 | LD0F4:ldd | 0x00,x | ;GET CONTENTS OF STACK |
| D0F6 | LD0F6:std | L01A7 | ;STORE INJ PW |
| D0F9 | beq | LD10F | ;BRANCH IF Z, ELSE |
| D0FB | ldd | L0104 | ;BPW ADJUSTMENT TERM |
| D0FE | bmi | LD105 | ;BRANCH IF NEGATIVE (DECEL ENLEAN), ELSE |
| D100 | add | L01A7 | ;ADD INJ PW |
| D103 | bra | LD10F | ;GO SAVE ON STACK |
| | | | |
| D105 | LD105:add | L01A7 | ;ADD INJ PW |
| D108 | beq | LD10C | ;BRANCH IF Z, ELSE |
| D10A | bcs | LD10F | ;BRANCH IF OVERFLOW, ELSE |
| D10C | LD10C:ldd | #0x0001 | ;LOAD 0001 |
| D10F | LD10F:std | 0x00,x | ;STORE INJ PW ON STACK |
| D111 | add | L0250 | ;ADD ACCUMULATED FUEL |
| D114 | std | L0250 | ;STORE |
| D117 | bcc | LD12B | ;BRANCH IF NO OVERFLOW, ELSE |
| D119 | ldd | L9272 | ;EMPERICALLY DERIVED MULTIPLIER |
| D11C | add | L0270 | ;ADD ACCUMULATED FUEL |
| D11F | std | L0270 | ;STORE ACCUMULATED FUEL |
| D122 | bcc | LD12B | ;BRANCH IF NO OVERFLOW, ELSE |
| D124 | ldd | *L0078 | ;LOAD |
| D126 | add | #0x0001 | ;INCREMENT BY ONE |
| D129 | std | *L0078 | ;STORE |
| D12B | LD12B:ldd | 0x00,x | ;GET CONTENTS FROM STACK |
| D12D | pulx | | ;RESTORE X |
| D12E | beq | LD15D | ;BRANCH IF Z, ELSE |
| D130 | ldab | *L00B4 | ;LOAD FUEL INJ OFFSET VS BATT VOLTS |
| D132 | abx | | ;ADD TO X |
| D133 | pshx | | ;TRANSFER TO ACCD |
| D134 | pula | | ; |
| D135 | pulb | | ; |
| D136 | tsta | | ;TEST MSB |
| D137 | bne | LD147 | ;BRANCH IF NOT = ZERO, ELSE |
| D139 | tba | | ;COPY LSB TO A REG |
| D13A | ldx | #0x8B9D | ;FUEL INJ OFFSET VS BPW |
| D13D | lsrb | | ;RESCALE |
| D13E | lsrb | | ;RESCALE |
| D13F | lsrb | | ;RESCALE |
| D140 | andb | #0x1E | ;%00011110 CLEAR BITS |
| D142 | abx | | ;ADD TO ADDRESS |
| D143 | ldx | 0x00,x | ;GET A VALUE FROM THE TABLE |
| D145 | tab | | ;COPY BPW LSB BACK TO B REG |
| D146 | abx | | ;ADD TABLE VALUE TO BPW LSB |
| D147 | LD147:bclr | *L009C,#0x20 | ;CLEAR FORCE OL FOR LOW PW |
| D14A | cpx | L8B87 | ;COMPARE MIN BPW |
| D14D | bhi | LD155 | ;BRANCH IF HIGHER, ELSE |
| D14F | bset | *L009C,#0x20 | ;FORCE OPEN LOOP BECAUSE OF LOW PW |
| D152 | ldx | L8B89 | ;DEFAULT PW FOR LOW PW |
| D155 | LD155:cpx | #0x0000 | ;COMPARE |
| D158 | bpl | LD15D | ;BRANCH IF > 0000, ELSE |
| D15A | ldx | #0x7FFF | ;LOAD MAX |
| D15D | LD15D:brset | *L0097,#0x80,LD167 | ;BRANCH IF RUN-FUEL (SEFI) ALLOWED, ELSE |
| D161 | jsr | LF5C1 | ;DO BATCH FIRE FUEL OUT ROUTINE |

```

D164          jmp      LD287          ;SKIP SEFI OUTPUT

;-----
; MAJOR SEFI FUEL OUT ROUTINE
;-----

D167 LD167:pshx          ;SAVE BPW STACK
D168          tsx          ;X POINTS TO STACK
D169          ldd      0x00,x      ;GET BPW INTO ACCD REG
D16B          subd     *L00E4      ;SUBTRACT FINAL BPW
D16D          bcs      LD175      ;BRANCH IF < BPW, ELSE
D16F          cpd      L8B34      ;FFFF
D173          bcc      LD178      ;BRANCH IF >= FFFF, ELSE
D175 LD175:ldd      #0x0000      ;LOAD 0000
D178 LD178:std      *L00B8      ;STORE ? DELTA BPW
D17A          pulx          ;RESTORE X (BPW)
D17B          ldaa     *L00F4      ;CCP DUTY CYCLE
D17D          beq      LD187      ;BRANCH IF Z, ELSE
D17F          cpx      L8BD9      ;0x0000
D182          bhi      LD187      ;BRANCH IF > CAL, ELSE
D184          ldx      L8BD9      ;0x0000
D187 LD187:ldaa     *L00AA      ;%TPS
D189          cmpa     L8BD0      ;00
D18C          bcs      LD193      ;BRANCH IF TPS% < 00, ELSE
D18E          cmpa     L8BD1      ;00
D191          bhi      LD1A9      ;BRANCH IF TPS% > 00, ELSE
D193 LD193:ldaa     *L00BF      ;MPH*3
D195          cmpa     L8BD2      ;00
D198          bcs      LD19F      ;BRANCH IF < 00, ELSE
D19A          cmpa     L8BD3      ;00
D19D          bhi      LD1A9      ;BRANCH IF > 00, ELSE
D19F LD19F:cpx      L8BD4      ;0x0000
D1A2          bcs      LD1A9      ;BRANCH IF < CAL, ELSE
D1A4          cpx      L8BD6      ;0x0000
D1A7          bls      LD1AD      ;BRANCH IF <= CAL, ELSE
D1A9 LD1A9:stx      *L00E4      ;STORE FINAL BPW
D1AB          bra      LD1BA      ;

D1AD LD1AD:pshx          ;SAVE BPW ON STACK
D1AE          pula          ;TRANSFER TO ACCD
D1AF          pulb          ;
D1B0          ldx      #0x00E4      ;OLD FILTERED BPW
D1B3          ldy      #0x8BD8      ;FILTER COEFFICIENT ADDRESS (NOT USED)
D1B7          jsr      L99F5      ;LAG FILTER

D1BA LD1BA:ldx      *L00E4      ;FINAL BPW
D1BC          brclr    *L00E2,#0x80,LD1C3 ;BRANCH IF NOT BIT 7, ELSE
D1C0          ldx      #0x0000      ;
D1C3 LD1C3:stx      L3FD0      ;STORE FUEL INJ #5 OUTPUT
D1C6          ldd      *L00E4      ;FINAL BPW
D1C8          brclr    *L00E2,#0x40,LD1CF ;BRANCH IF NOT BIT 6, ELSE
D1CC          ldd      #0x0000      ;
D1CF LD1CF:oraa     #0x40      ;SET BIT 6
D1D1          ldx      #0x4000      ;I/O DATA REG
D1D4          bset     0x02,x,#0x10 ;SET BIT 4 I/O PORT
D1D7          staa     0x00,x      ;WR TO DATA REG
          ; - FUEL INJ #6 OUTPUT
          ;THIS ALSO CRITICAL FOR CORRECT TIMING AND
          ; SEQUENCE OF FUEL INJECTOR TIMING,
          ; ESPECIALLY IN THE ABSENCE OF A CAM PULSE

D1D9          brclr    0x01,x,#0x80 ;CLEAR BIT 7 I/O CNTL REG
D1DC          ldx      *L00E4      ;FINAL BPW
D1DE          brclr    *L00E2,#0x20,LD1E5 ;BRANCH IF NOT BIT 5, ELSE
D1E2          ldx      #0x0000      ;
D1E5 LD1E5:stx      L3F80      ;FUEL INJ #1 OUTPUT

```

| | | | |
|------|------------|--------------------|-------------------------------------|
| D1E8 | ldx | *L00E4 | ;FINAL BPW |
| D1EA | brclr | *L00E2,#0x10,LD1F1 | ;BRANCH IF NOT BIT 4, ELSE |
| D1EE | ldx | #0x0000 | ; |
| D1F1 | LD1F1:stx | L3F82 | ;FUEL INJ #2 OUTPUT |
| D1F4 | psha | | ;DELAY |
| D1F5 | pula | | ; |
| D1F6 | ldx | #0x4000 | ;I/O DATA REG ADDRESS |
| D1F9 | ldaa | 0x00,x | ;GET DATA FROM REG |
| D1FB | stab | 0x00,x | ;WR TO I/O DATA REG |
| D1FD | bclr | 0x01,x,#0x80 | ;CLEAR BIT 7 I/O CNTL REG |
| D200 | staa | *L008D | ;STORE DATA |
| D202 | ldd | *L00E4 | ;FINAL BPW |
| D204 | brclr | *L00E2,#0x08,LD20B | ;BRANCH IF NOT BIT 4, ELSE |
| D208 | ldd | #0x0000 | ; |
| D20B | LD20B:std | L3F84 | ;FUEL INJ #3 OUTPUT |
| D20E | ldd | *L00E4 | ;FINAL BPW |
| D210 | brclr | *L00E2,#0x04,LD217 | ;BRANCH IF NOT BIT 2, ELSE |
| D214 | ldd | #0x0000 | ; |
| D217 | LD217:std | L3F86 | ;FUEL INJ #4 OUTPUT |
| D21A | nop | | ;DELAY |
| D21B | nop | | |
| D21C | ldaa | 0x00,x | ;GET DATA FROM PORT |
| D21E | staa | *L008C | ;STORE DATA |
| D220 | bita | #0x40 | ;BIT 6 SET? |
| D222 | beq | LD227 | ;BRANCH IF NO CAM REF PULSE, ELSE |
| D224 | bset | *L0093,#0xC0 | ;SET CAM REF PULSE OCCURED |
| D227 | LD227:bclr | 0x02,x,#0x10 | ;CLEAR BIT 4 I/O PORT |
| D22A | ldd | L0106 | ;ASYNC PW |
| D22D | bne | LD287 | ;BRANCH IF NOT Z, ELSE |
| D22F | ldd | *L00B8 | ;; DELTA BPW |
| D231 | beq | LD287 | ;BRANCH IF Z, ELSE |
| D233 | clra | | ;CLEAR A |
| D234 | ldx | #0x4000 | ;I/O DATA REG ADDRESS |
| D237 | bset | 0x02,x,#0x10 | ;SET BIT 4 I/O PORT |
| D23A | staa | 0x00,x | ;WR TO I/O DATA REG |
| D23C | bclr | 0x01,x,#0x80 | ;CLEAR BIT 7 I/O CNTL REG |
| D23F | nop | | ;DELAY |
| D240 | nop | | |
| D241 | psha | | |
| D242 | pula | | |
| D243 | nop | | |
| D244 | ldab | *L008D | ;GET PREVIOUS DATA |
| D246 | andb | #0x07 | ;CLEAR BITS |
| D248 | ldx | #0xFB7C | ;INDEX |
| D24B | abx | | ;ADD TO INDEX |
| D24C | ldab | 0x00,x | ;GET DATA |
| D24E | nop | | ;DELAY |
| D24F | ldx | #0x4000 | ;I/O DATA REG ADDRESS |
| D252 | ldaa | 0x00,x | ;GET DATA FROM REG |
| D254 | stab | 0x00,x | ;WR TO I/O DATA REG |
| D256 | bclr | 0x01,x,#0x80 | ;CLEAR BIT 7 I/O CNTL REG |
| D259 | staa | *L008D | ;STORE NEW DATA |
| D25B | nop | | ;DELAY |
| D25C | ldx | *L00B8 | ;; DELTA BPW |
| D25E | ldab | *L00B4 | ;LOAD FUEL INJ OFFSET VS BATT VOLTS |
| D260 | abx | | ;ADD TO X |
| D261 | stx | L3FF2 | ;STORE ? ASYNC PW |
| D264 | mul | | ;DELAY |
| D265 | ldd | L3FFC | ;CPU CNTL REG |
| D268 | oraa | #0x04 | ;; TRIGGER ASYNC PW |
| D26A | psha | | ;SAVE ON STACK |
| D26B | ldx | #0x4000 | ;I/O DATA REG ADDRESS |
| D26E | ldaa | 0x00,x | ;GET DATA FROM REG |
| D270 | staa | *L008C | ;STORE NEW DATA |

| | | | |
|------|-------------|--------------------|---|
| D272 | bita | #0x40 | ;BIT 6 ? |
| D274 | beq | LD279 | ;BRANCH IF NO CAM REF PULSE, ELSE |
| D276 | bset | *L0093,#0xC0 | ;SET CAM REF PULSE OCCURED |
| D279 | LD279:bclr | 0x02,x,#0x10 | ;CLEAR BIT 4 I/O PORT |
| D27C | pula | | ; |
| D27D | std | L3FFC | ;CPU CNTL REG |
| D280 | anda | #0xFB | ;CLEAR BIT 2 |
| D282 | pshx | | ;DELAY |
| D283 | pulx | | |
| D284 | std | L3FFC | ;CPU CNTL REG |
| D287 | LD287:brclr | *L0093,#0x80,LD2AE | ;BRANCH IF CAM PULSE NOT SEEN, ELSE |
| D28B | brclr | *L0092,#0x80,LD2A3 | ;BRANCH IF NOT BIT 7, ELSE |
| D28F | ldaa | *L00B7 | ;CYL COUNTER |
| D291 | cmpa | L84D6 | ;0x05 |
| D294 | beq | LD2AB | ;BRANCH IF = 5, ELSE |
| D296 | brclr | *L0097,#0x80,LD2A3 | ;BRANCH IF RUN FUEL NOT ALLOWED, ELSE |
| D29A | ldab | L0132 | ;TIMER FOR MALF 341 (INT CAM SIGNAL) |
| D29D | incb | | ;INCREMENT |
| D29E | beq | LD2A3 | ;BRANCH IF Z, ELSE |
| D2A0 | stab | L0132 | ;STORE M341 TIMER |
| D2A3 | LD2A3:bset | *L0092,#0x80 | ; |
| D2A6 | ldaa | L84D6 | ;0x05 |
| D2A9 | staa | *L00B7 | ;CYL COUNTER |
| D2AB | LD2AB:bclr | *L0093,#0x80 | ;CLEAR CAM PULSE SEEN |
| D2AE | LD2AE:ldx | #0x4000 | ;I/O DATA REG |
| D2B1 | bset | *L0089,#0x80 | ;SET BIT 7 |
| D2B4 | bset | 0x02,x,#0x04 | ;SET BIT 2 I/O PORT |
| D2B7 | ldaa | *L0089 | ; |
| D2B9 | staa | 0x00,x | ;WR TO I/O DATA REG |
| D2BB | bclr | 0x01,x,#0x80 | ;CLEAR BIT 7 I/O CNTL REG |
| D2BE | pshb | | ;SAVE B - M341 TIMER |
| D2BF | ldd | L0181 | ;PREV REF PERIOD |
| D2C2 | subd | L3FC0 | ;SUBTRACT REF PERIOD |
| D2C5 | bcc | LD2DB | ;BRANCH IF REF PER INCREASING, ELSE |
| D2C7 | nega | | ;INVERT |
| D2C8 | negb | | ;INVERT |
| D2C9 | sbca | #0x00 | ;ROUND DOWN |
| D2CB | cpd | L0305 | ;COMPARE PREV DELTA REF PERIOD |
| D2CF | bls | LD2DB | ;BRANCH IF DELTA REF PER DECREASING, ELSE |
| D2D1 | std | L0305 | ;STORE DELTA REF PERIOD |
| D2D4 | psha | | ;SAVE MSB |
| D2D5 | ldaa | *L00B7 | ;LOAD CYL COUNTER |
| D2D7 | staa | L0307 | ;STORE COUNTER |
| D2DA | pula | | ;GET BACK MSB |
| D2DB | LD2DB:ldd | L3FC0 | ;LOAD REF PERIOD |
| D2DE | std | L0181 | ;SAVE FOR NEXT PASS |
| D2E1 | ldaa | *L00B7 | ;CYL COUNTER |
| D2E3 | cmpa | #0x01 | ;1 ? |
| D2E5 | bne | LD31F | ;BRANCH IF NOT = 1, ELSE |
| D2E7 | ldd | #0x0000 | ; |
| D2EA | std | L0305 | ; |
| D2ED | ldaa | L0307 | ;LOAD COUNTER |
| D2F0 | beq | LD313 | ;BRANCH IF Z, ELSE |
| D2F2 | cmpa | L0309 | ; |
| D2F5 | bne | LD313 | ;BRANCH IF THEY'RE NOT EQUAL, ELSE |
| D2F7 | ldab | L0308 | ;LOAD |
| D2FA | cmpb | L9179 | ;26 |
| D2FD | bhi | LD304 | ;BRANCH IF > 26, ELSE |
| D2FF | inc | L0308 | ;INCREMENT |
| D302 | bra | LD319 | ;GO STORE ACCA -> L0309 |
| D304 | LD304:tab | | ; |
| D305 | subb | L84D7 | ;2 |
| D308 | cmpb | #0x00 | ;0? |

```

D30A          bgt      LD30E          ;BRANCH IF > 0, ELSE
D30C          addb     #0x06          ;ADD 6
D30E  LD30E:stab  L0183          ;STORE BAD CYL ID
D311          bra      LD319          ;GO STORE ACCA -> L0309

D313  LD313:clr    L0308          ;
D316          clr      L0183          ;CLEAR BAD CYL ID
D319  LD319:staa   L0309          ;SAVE FOR NEXT PASS
D31C          clr      L0307
D31F  LD31F:pulb   ;GET M341 TIMER BACK INTO B REG
D320          ldaa     0x00,x        ;4000 - GET DATA FROM I/O REG
D322          bclr     0x02,x,#0x04 ;4002 - ? CHIP SELECT TO IN AND OUT SPI DEV
D325          staa     *L008B        ;STORE INPUT STATES
D327          brclr    *L0095,#0x82,LD333 ;BRANCH IF HUD DEVICE NOT CONNECTED, ELSE
D32B          ldaa     L8023          ;OPTION WORD - TIS CLEAR
D32E          beq      LD333          ;BRANCH IF Z, ELSE
D330          jsr      L59D2          ;? HUD
D333  LD333:rti          ;RETURN
; END OF XIRQ ROUTINE
;-----

;-----
; THIS NOT USED ?
;-----

D334          psha
D335          ldd      L02D0
D338          ldx      L02D2
D33B          fdiv          ;D/X
D33C          ldd      L02D4
D33F          jsr      L992F          ;16 x 16 MULTIPLY
D342          std      L02DE
D345          ldx      #0x02DE
D348          ldaa     #0x89
D34A          jsr      L989D          ;8 X 16 MULTIPLY, PRODUCT SAVED IN L02DE
D34D          bra      LD384

;-----
; THIS NOT USED ?
;-----

D34F          psha
D350          ldd      L02CE
D353          ldx      L02D0
D356          fdiv          ;D/X
D357          ldd      L02D2
D35A          jsr      L992F          ;16 x 16 MULTIPLY
D35D          std      L02DE
D360          ldx      #0x02DE
D363          ldaa     #0x89
D365          jsr      L989D          ;8 X 16 MULTIPLY, PRODUCT SAVED IN L02DE
D368          bra      LD384

;-----
; THIS NOT USED ?
;-----

D36A          psha
D36B          ldd      L02D0
D36E          lsr      L02D0
D36F          ldx      L02CE
D372          fdiv          ;D/X
D373          ldd      L02D0
D376          jsr      L992F          ;16 x 16 MULTIPLY
D379          std      L02DE
D37C          ldx      #0x02DE          ;MULTIPLICAND

```

```

D37F      ldaa    #0xF2                      ;MULTIPLIER
D381      jsr     L989D                      ;8 X 16 MULTIPLY, PRODUCT SAVED IN L02DE
D384      LD384:ldx    #0x4002                ;CPU DATA LATCH
D387      bset    0x00,x,#0x80                ;? CHIP SELECT TO OUTPUT ONLY SPI DEVICE
D38A      bclr    0x00,x,#0x80                ;
D38D      std     *L00C8                      ;3/4 3x A REF PERIOD
D38F      lsld    ;
D390      addd    *L00C8                      ;3/4 3x A REF PERIOD
D392      lsld    ;
D393      std     *L00C8                      ;3/4 3x A REF PERIOD
D395      ldd     L3FC4                      ;A REF PERIOD
D398      std     *L00C4                      ;STORE REF PERIOD
D39A      ldx     #0x00C8                    ;3/4 3x A REF PERIOD
D39D      pula    ;
D39E      jsr     L98B9                      ;8 x 16 MULTIPLY
D3A1      addd    *L00C4                      ;ADD REF PERIOD
D3A3      std     L3FE2                      ;STORE ? TIMER OUTPUT
D3A6      jmp     LD333                      ;RETURN

;-----
; BRANCH HERE FROM SEGMENT TABLE
;
;-----

D3A9      brset   *L0011,#0x10,LD3B0          ;MINOR LOOP COUNTER - 100 MSEC
D3AD      jmp     LD539                      ;ELSE JUMP OVER

D3B0      LD3B0:bclr    *L0097,#0x04          ;
D3B3      brclr   *L0086,#0x80,LD3D2          ;BRANCH IF ENGINE NOT RUNNING, ELSE
D3B7      ldx     #0x022B                    ;INDEX INPUT MESSAGE BUFFER
D3BA      brclr   *L0084,#0x80,LD3C9          ;BRANCH IF NOT MODE 4, ELSE
D3BE      brclr   0x05,x,#0x01,LD3C9          ;0230, BIT 0
D3C2      brclr   0x06,x,#0x01,LD3D2          ;0231, BIT 0
D3C6      jmp     LD4ED                      ;GO SET CLOSED LOOP

D3C9      LD3C9:brclr   *L0016,#0x0C,LD3D5      ;BRANCH IF NO O2 SENSOR MALFS, ELSE
D3CD      clr     L0113                      ;CLEAR O2 READY TIMER
D3D0      bra     LD440                      ;GO SET O2 SENSOR READY

D3D2      LD3D2: jmp     LD4F8                      ;GO CLEAR CLOSED LOOP

D3D5      LD3D5:ldaa    L0113                  ;LOAD O2 READY TIMER
D3D8      cmpa    L8965                      ;TIME LIMIT - 5 SECONDS
D3DB      bcc     LD3E3                      ;BRANCH IF > 5 SECONDS, ELSE
D3DD      inca    ;INCREMENT TIMER
D3DE      staa    L0113                      ;STORE TIMER
D3E1      bra     LD3E6                      ;

D3E3      LD3E3:bclr    *L0012,#0x01          ;CLEAR O2 SENSOR READY
D3E6      LD3E6:brset   *L0084,#0x80,LD41A      ;BRANCH IF MODE 4, ELSE
D3EA      clra    ;CLEAR A
D3EB      brset   *L009C,#0x80,LD414          ;BRANCH IF CLOSED LOOP, ELSE
D3EF      ldab    *L00A7                    ;FILTERED CTS
D3F1      cmpb    L8968                      ;68 DEG C
D3F4      bhi     LD41A                      ;BRANCH IF > 68 DEG C, ELSE
D3F6      brset   *L0099,#0x80,LD41A          ;BRANCH IF BIT 7, ELSE
D3FA      ldab    *L00AC                    ;LOAD RPM/25
D3FC      cmpb    L8966                      ;1800 RPM
D3FF      bls     LD40F                      ;BRANCH IF <= 1800 RPM, ELSE
D401      ldaa    L0192                      ;
D404      cmpa    L8967                      ;5
D407      bcs     LD40E                      ;BRANCH IF < 5, ELSE
D409      bset    *L0099,#0x80              ;SET BIT 7
D40C      bra     LD41A

```

| | | | |
|------|-------------|--------------------|---------------------------------------|
| D40E | LD40E:inca | | ; INCREMENT |
| D40F | LD40F:staa | L0192 | ; STORE |
| D412 | bra | LD440 | ; |
| | | | |
| D414 | LD414:staa | L0192 | ; |
| D417 | bclr | *L0099,#0x80 | ; CLEAR BIT 7 |
| D41A | LD41A:brclr | *L0012,#0x01,LD440 | ; BRANCH IF O2 SENSOR NOT READY, ELSE |
| D41E | brset | *L0084,#0x80,LD434 | ; BRANCH IF MODE 4, ELSE |
| D422 | ldx | #0x8960 | ; INDEX |
| D425 | ldaa | *L00F2 | ; LOAD STARTUP COOLANT |
| D427 | cmpa | L895F | |
| D42A | bhi | LD42E | |
| D42C | inx | | ; 8961 |
| D42D | inx | | ; 8962 |
| D42E | LD42E:ldd | *L0032 | ; RUN TIME |
| D430 | subd | 0x00,x | ; 8960 OR |
| | | | ; 8962 |
| D432 | bcs | LD440 | ; |
| D434 | LD434:brset | *L0091,#0x10,LD443 | ; BRANCH IF MAF SIGNAL LOW, ELSE |
| D438 | ldab | *L00A7 | ; FILTERED CTS |
| D43A | ldaa | L8964 | ; 27.5 DEG C |
| D43D | cba | | ; |
| D43E | bls | LD443 | ; |
| D440 | LD440:bset | *L0097,#0x04 | ; |
| D443 | LD443:ldaa | L801F | ; OPTION WORD - TIS SET |
| D446 | beq | LD44C | ; BRANCH IF Z, ELSE |
| D448 | brset | *L0015,#0x10,LD474 | ; BRANCH IF MAF SENSOR MALF, ELSE |
| D44C | LD44C:ldx | #0x890A | ; INDEX |
| D44F | ldaa | *L00BA | ; LOAD LV8 |
| D451 | cmpa | 0x04,x | ; 890E |
| D453 | bls | LD474 | ; BRANCH IF <= CAL, ELSE |
| D455 | brset | *L0086,#0x02,LD45D | ; BRANCH IF HOT OPEN LOOP, ELSE |
| D459 | bset | *L0086,#0x02 | ; SET HOT OPEN LOOP |
| D45C | inx | | ; 890B |
| D45D | LD45D:ldaa | *L00A6 | ; LOAD CLNT TEMP (DEFAULTED) |
| D45F | cmpa | 0x00,x | ; 890A OR |
| | | | ; 890B |
| D461 | bls | LD474 | ; BRANCH IF <= CAL, ELSE |
| D463 | ldaa | *L00BD | ; FILTERED MPH |
| D465 | cmpa | 0x02,x | ; 890C OR |
| | | | ; 890D |
| D467 | bcs | LD474 | ; BRANCH IF < CAL, ELSE |
| D469 | ldab | L025E | ; LOAD |
| D46C | cmpb | L890F | ; COMPARE 1 |
| D46F | bhi | LD47B | ; BRANCH IF > CAL, ELSE |
| D471 | incb | | ; INCREMENT |
| D472 | bra | LD475 | ; GO STORE L025E |
| | | | |
| D474 | LD474:clrb | | ; |
| D475 | LD475:stab | L025E | ; |
| D478 | bclr | *L0086,#0x02 | ; CLEAR HOT OPEN LOOP |
| D47B | LD47B:bset | *L0099,#0x01 | ; |
| D47E | brset | *L0091,#0x10,LD4D7 | ; BRANCH IF MAF SIGNAL LOW, ELSE |
| D482 | ldaa | *L00A6 | ; LOAD CLNT TEMP (DEFAULTED) |
| D484 | cmpa | L8913 | ; 70.25 DEG C |
| D487 | bcs | LD4D7 | ; BRANCH IF < 70.25 DEG C, ELSE |
| D489 | ldab | L02BF | ; |
| D48C | ldaa | *L00CD | ; LOAD MAF (GM/SEC) |
| D48E | cmpa | L8914 | ; 50 G/S |
| D491 | bhi | LD49E | ; BRANCH IF > CAL, ELSE |
| D493 | cmpa | L8915 | ; 35 G/S |
| D496 | bcc | LD49F | ; BRANCH IF >= CAL, ELSE |
| D498 | tstb | | ; TEST |
| D499 | beq | LD4B0 | ; BRANCH IF Z, ELSE |

| | | | |
|------|--------------|----------------------|--|
| D49B | dec b | | ; DECREMENT |
| D49C | bra | LD49F | ; |
| D49E | LD49E: inc b | | ; |
| D49F | LD49F: cmp b | L8916 | ; 75 |
| D4A2 | beq | LD4AB | ; BRANCH IF = CAL, ELSE |
| D4A4 | bhi | LD4DA | ; BRANCH IF > CAL, ELSE |
| D4A6 | stab | L02BF | ; SAVE |
| D4A9 | bra | LD4B0 | ; |
| D4AB | LD4AB: stab | L02BF | ; SAVE |
| D4AE | bra | LD4DA | ; |
| D4B0 | LD4B0: ldab | L02C0 | ; |
| D4B3 | ldaa | *L00CD | ; LOAD MAF (GM/SEC) |
| D4B5 | cmpa | L8918 | ; 70 G/S |
| D4B8 | bhi | LD4C5 | ; BRANCH IF > CAL, ELSE |
| D4BA | cmpa | L8919 | ; 35 G/S |
| D4BD | bcc | LD4C6 | ; BRANCH IF >= CAL, ELSE |
| D4BF | tstb | | ; TEST |
| D4C0 | beq | LD4D7 | ; BRANCH IF Z, ELSE |
| D4C2 | dec b | | ; DECREMENT |
| D4C3 | bra | LD4C6 | ; |
| D4C5 | LD4C5: inc b | | ; INCREMENT |
| D4C6 | LD4C6: cmp b | L891A | ; 75 |
| D4C9 | beq | LD4D2 | ; BRANCH IF = CAL, ELSE |
| D4CB | bhi | LD4DA | ; BRANCH IF > CAL, ELSE |
| D4CD | stab | L02C0 | ; SAVE |
| D4D0 | bra | LD4D7 | |
| D4D2 | LD4D2: stab | L02C0 | ; SAVE TIMER |
| D4D5 | bra | LD4DA | |
| D4D7 | LD4D7: bclr | *L0099, #0x01 | ; CLEAR BIT 0 |
| D4DA | LD4DA: brset | *L0099, #0x01, LD4E8 | ; |
| D4DE | brset | *L0086, #0x02, LD4E8 | ; BRANCH IF HOT OPEN LOOP, ELSE |
| D4E2 | brset | *L0097, #0x04, LD4F8 | ; |
| D4E6 | bra | LD4ED | ; GO SET CLOSED LOOP |
| D4E8 | LD4E8: bset | *L0099, #0x80 | ; |
| D4EB | bra | LD4F8 | ; GO CLEAR CLOSED LOOP |
| D4ED | LD4ED: bset | *L009C, #0x80 | ; SET CLOSED LOOP |
| D4F0 | brclr | *L0095, #0x82, LD4FB | ; BRANCH IF HUD DEVICE NOT CONNECTED, ELSE |
| D4F4 | brclr | *L009F, #0x30, LD4FB | ; |
| D4F8 | LD4F8: bclr | *L009C, #0x80 | ; CLEAR CLOSED LOOP |
| D4FB | LD4FB: brclr | *L0099, #0x01, LD507 | ; |
| D4FF | ldaa | L8917 | ; |
| D502 | staa | L02C1 | ; |
| D505 | bra | LD510 | |
| D507 | LD507: ldaa | L02C1 | ; TIMER |
| D50A | beq | LD510 | ; BRANCH IF Z, ELSE |
| D50C | deca | | ; DECREMENT |
| D50D | staa | L02C1 | ; SAVE |
| D510 | LD510: brclr | *L009C, #0x80, LD535 | ; BRANCH IF OPEN LOOP, ELSE |
| D514 | ldab | *L00A7 | ; FILTERED CTS |
| D516 | cmpb | L8A1C | |
| D519 | bls | LD535 | ; BRANCH IF <= CAL, ELSE |
| D51B | cmpb | L8A1D | ; |
| D51E | bhi | LD535 | ; BRANCH IF > CAL, ELSE |
| D520 | clra | | ; CLEAR |
| D521 | ldab | L8969 | ; STOICH FAR |

```

D524      cpd      *L00B5      ;FUEL AIR RATIO
D527      bne      LD535      ;BRANCH IF FAR NOT = STOICH, ELSE
D529      ldaa     *L00BA      ;LOAD LV8
D52B      cmpa     L8A1E      ;
D52E      bls      LD535      ;BRANCH IF < CAL, ELSE
D530      bset     *L009C,#0x02 ;SET LEARN ENABLED
D533      bra      LD538

D535      LD535:bclr  *L009C,#0x02 ;SET LEARN DISABLED
D538      LD538:rts

;-----
; FAN CONTROL
;-----

D539      LD539:brclr *L0095,#0x10,LD572 ;BRANCH IF BATT VOLTS NOT < 4.0, ELSE
D53D      ldx      *L0058      ;LOAD TIMER
D53F      cpx      #0x0050      ;80
D542      bls      LD56F      ;BRANCH IF <= 80, ELSE
D544      brset    *L0091,#0x08,LD56F ;BRANCH IF BIT 3, ELSE
D548      ldx      L02E7      ;
D54B      bne      LD55B      ;BRANCH IF NOT Z, ELSE
D54D      ldab     *L00A6      ;LOAD CLNT TEMP (DEFAULTED)
D54F      cmpb     L86B7      ;115 DEG C
D552      bls      LD56C      ;BRANCH IF <= CAL, ELSE
D554      ldab     *L00F0      ;IATMAT
D556      cmpb     L86B8      ;96d
D559      bls      LD56C      ;BRANCH IF <= CAL, ELSE
D55B      LD55B:cpx  L86B9      ;60 SECONDS - FAN ON LIMIT
D55E      bhi      LD56F      ;BRANCH IF > 60 SECONDS, ELSE
D560      inx      ;INCREMENT TIMER
D561      stx      L02E7      ;STORE
D564      ldx      #0x0051      ;81
D567      stx      *L0058      ;STORE TIMER
D569      jmp      LD5ED      ;GO TURN ON FAN #1

D56C      LD56C:bset  *L0091,#0x08      ;SET BIT 3
D56F      LD56F:jmp   LD5F9      ;GO TURN OFF FAN #1

D572      LD572:bclr  *L0091,#0x08      ;CLEAR BIT 3
D575      ldx      #0x0000      ;
D578      stx      L02E7      ;RESET TIMER
D57B      brset    *L0086,#0x80,LD584 ;BRANCH IF ENGINE RUNNING, ELSE
D57F      inc      L02F9      ;INCREMENT TIMER
D582      bne      LD589      ;BRANCH IF NOT Z, ELSE
D584      LD584:ldaa  #0xFF      ;LOAD 255
D586      staa     L02F9      ;STORE TIMER
D589      LD589:brclr *L0096,#0x02,LD596 ;BRANCH IF FAN #1 OFF, ELSE
D58D      ldx      L031B      ;FAN #1 ON TIMER
D590      inx      ;INCREMENT TIMER
D591      beq      LD596      ;BRANCH IF Z, ELSE
D593      stx      L031B      ;STORE TIMER
D596      LD596:brset *L0091,#0x40,LD5ED ;BRANCH IF FAN #2 WAITING TO TURN ON, ELSE
D59A      ldaa     #0x04      ;
D59C      jsr      LD933      ;GO CHECK MODE 4 CONTROL WORDS
D59F      bmi      LD5ED      ;BRANCH IF BIT 7
D5A1      bne      LD615      ;BRANCH IF NZ, ELSE
D5A3      ldab     L02F9      ;LOAD TIMER
D5A6      cmpb     #0x0F      ;15
D5A8      bcs      LD5F9      ;BRANCH IF < 15, ELSE
D5AA      brset    *L0013,#0x40,LD5ED ;BRANCH IF CTS HIGH MALF, ELSE
D5AE      brset    *L0013,#0x20,LD5ED ;BRANCH IF CTS LOW MALF, ELSE
D5B2      ldx      #0x86AC      ;INDEX
D5B5      brclr    *L0096,#0x02,LD5BA ;BRANCH IF FAN #1 OFF, ELSE
D5B9      inx      ;86AD

```

| | | | |
|------|---------------|--------------------|--|
| D5BA | LD5BA:ldaa | *L00A6 | ;LOAD CLNT TEMP (DEFAULTED) |
| D5BC | cmpa | 0x00,x | ;86AC - TEMP TO TURN ON FAN #1 OR #2 OR |
| | | | ;86AD - TEMP TO TURN OFF FAN #1 OR #2 |
| D5BE | bhi | LD5ED | ;BRANCH IF > CAL, ELSE |
| D5C0 | cmpa | 0x04,x | ;86B0 - TEMP TO TURN ON FAN #1 OR |
| | | | ;86B1 - TEMP TO TURN OFF FAN #1 |
| D5C2 | bhi | LD5ED | ;BRANCH IF > CAL, ELSE |
| D5C4 | ldab | *L00BD | ;FILTERED MPH |
| D5C6 | cmpb | 0x02,x | ;86AE OR |
| | | | ;86AF |
| D5C8 | bcc | LD5F9 | ;BRANCH IF >= CAL, ELSE |
| D5CA | ldaa | L8009 | ;OPTION WORD - A/C PRESSURE SENSOR PRESENT |
| D5CD | beq | LD5E2 | ;BRANCH IF Z, ELSE (TIS SET) |
| D5CF | brset | *L008E,#0x80,LD5E2 | ;BRANCH IF A/C NOT REQUESTED, ELSE |
| D5D3 | ldaa | *L00F9 | ;A/C PRESSURE |
| D5D5 | ldab | L86BC | ; |
| D5D8 | brset | *L0096,#0x02,LD5DF | ;BRANCH IF FAN #1 ON, ELSE |
| D5DC | ldab | L86BB | ; |
| D5DF | LD5DF: cba | | ;COMPARE CAL AND A/C PRESSURE |
| D5E0 | bhi | LD5ED | ;BRANCH A/C PRESSURE > CAL, ELSE |
| D5E2 | LD5E2:ldaa | *L00F0 | ;IATMAT (DEG C) |
| D5E4 | cmpa | 0x06,x | ;86B2 OR |
| | | | ;86B3 |
| D5E6 | bls | LD5F9 | ;BRANCH IF <= CAL, ELSE |
| D5E8 | ldx | L0292 | ;FAN #1 TURN ON DELAY TIMER FOR A/C |
| D5EB | beq | LD5F9 | ;BRANCH IF Z, ELSE |
| D5ED | LD5ED:bset | *L0096,#0x02 | ;TURN FAN #1 ON |
| D5F0 | ldaa | *L0070 | ;FAN #1 ON TIMER |
| D5F2 | inca | | ;INCREMENT TIMER |
| D5F3 | beq | LD620 | ;BRANCH IF Z, ELSE |
| D5F5 | staa | *L0070 | ;STORE TIMER |
| D5F7 | bra | LD620 | ; |
| | | | ;FAN #1 MIN ON TIMER |
| D5F9 | LD5F9:ldx | L031B | ;BRANCH IF Z, ELSE |
| D5FC | beq | LD615 | ;BRANCH IF FAN #2 ON, ELSE |
| D5FE | brset | *L0096,#0x10,LD609 | ;30 SECONDS |
| D602 | cpx | L86BF | ;BRANCH IF > CAL, ELSE |
| D605 | bhi | LD615 | ; |
| D607 | bra | LD620 | |
| | | | ;30 SECONDS |
| D609 | LD609:ldd | L86BF | ;30 SECONDS |
| D60C | addd | L86C1 | ;FAN #1 TIMER |
| D60F | cpd | L031B | ;BRANCH IF <= 60 SECONDS, ELSE |
| D613 | bcc | LD620 | ;TURN FAN #1 OFF |
| D615 | LD615:bclr | *L0096,#0x02 | ; |
| D618 | ldd | #0x0000 | ;RESET FAN #1 ON TIMER |
| D61B | staa | *L0070 | ;RESET FAN #1 TIMER |
| D61D | std | L031B | |
| | | | |
| | ;----- | | |
| | ; FAN #2 ALGO | | |
| | ;----- | | |
| D620 | LD620:ldaa | #0x02 | ;BIT 1 |
| D622 | jsr | LD933 | ;GO CHECK MODE 4 CONTROL WORDS |
| D625 | bmi | LD674 | ;BRANCH IF N, ELSE |
| D627 | bne | LD686 | ;BRANCH IF NOT Z, ELSE |
| D629 | brclr | *L0096,#0x10,LD636 | ;BRANCH IF FAN #2 OFF, ELSE |
| D62D | ldx | L031D | ;LOAD FAN #2 ON TIMER |
| D630 | inx | | ;INCREMENT TIMER |
| D631 | beq | LD636 | ;BRANCH IF Z, ELSE |
| D633 | stx | L031D | ;STORE FAN #2 ON TIMER |
| D636 | LD636:ldab | L02F9 | ; |
| D639 | cmpb | #0x0F | ;15 |
| D63B | bcs | LD679 | ;BRANCH IF < 15, ELSE |

| | | | | |
|------|--------|-------|--------------------|--|
| D63D | | brset | *L0013,#0x40,LD668 | ;BRANCH IF CTS HIGH MALF, ELSE |
| D641 | | brset | *L0013,#0x20,LD668 | ;BRANCH IF CTS LOW MALF, ELSE |
| D645 | | ldx | #0x86AA | ;INDEX |
| D648 | | brclr | *L0096,#0x10,LD64D | ;BRANCH IF FAN #2 OFF, ELSE |
| D64C | | inx | | ;86AB |
| D64D | LD64D: | ldaa | *L00A6 | ;LOAD CLNT TEMP (DEFAULTED) |
| D64F | | cmpa | 0x02,x | ;86AC OR |
| | | | | ;86AD |
| D651 | | bhi | LD668 | ;BRANCH IF > CAL, ELSE |
| D653 | | ldaa | L8009 | ;OPTION WORD - A/C PRESSURE SENSOR PRESENT |
| D656 | | beq | LD679 | ;BRANCH IF Z, ELSE (TIS SET) |
| D658 | | brset | *L008E,#0x80,LD679 | ;BRANCH IF A/C NOT REQUESTED, ELSE |
| D65C | | ldaa | *L00F9 | ;A/C PRESSURE |
| D65E | | cmpa | 0x13,x | ;86BD OR |
| | | | | ;86BE |
| D660 | | bls | LD679 | ;BRANCH IF <= CAL, ELSE |
| D662 | | ldaa | *L00BD | ;FILTERED MPH |
| D664 | | cmpa | 0x00,x | ;86AA OR |
| | | | | ;86AB |
| D666 | | bhi | LD679 | ;BRANCH IF > CAL, ELSE |
| D668 | LD668: | ldaa | *L0070 | ;FAN ON TIMER |
| D66A | | cmpa | L86B6 | ;FAN #2 TURN ON DELAY TIME |
| D66D | | bcc | LD674 | ;BRANCH IF >= CAL, ELSE |
| D66F | | bset | *L0091,#0x40 | ;SET BIT 6 - FAN #2 WAITING TO TURN ON |
| D672 | | bra | LD67C | |
| D674 | LD674: | bset | *L0096,#0x10 | ;TURN ON FAN #2 |
| D677 | | bra | LD68F | |
| D679 | LD679: | bclr | *L0091,#0x40 | ;CLEAR FAN #2 WAITING TO TURN ON |
| D67C | LD67C: | ldx | L031D | ;FAN #2 ON TIMER |
| D67F | | beq | LD686 | ;BRANCH IF Z, ELSE |
| D681 | | cpx | L86C1 | ;30 SECONDS |
| D684 | | bls | LD68F | ;BRANCH IF <= CAL, ELSE |
| D686 | LD686: | bclr | *L0096,#0x10 | ;TURN OFF FAN #2 |
| D689 | | ldd | #0x0000 | ; |
| D68C | | std | L031D | ;RESET FAN #2 ON TIMER |
| D68F | LD68F: | ldx | L0292 | ;LOAD FAN #1 TURN ON DELAY TIMER |
| D692 | | beq | LD695 | ;BRANCH IF Z, ELSE |
| D694 | | dex | | ;DECREMENT TIMER |
| D695 | LD695: | stx | L0292 | ;STORE TIMER |
| D698 | | rts | | ; |
| | | | | ; |
| | | | | ----- |
| | | | | ; BRANCH HERE FROM SEGMENT TABLE |
| | | | | ; CCP SUBROUTINE |
| | | | | ----- |
| D699 | | brclr | *L0097,#0x20,LD6A0 | ;BRANCH IF NOT HIGH BATT VOLTAGE, ELSE |
| D69D | | jmp | LD723 | ;JUMP OVER CCP CALCULATION |
| D6A0 | LD6A0: | brclr | *L0084,#0x80,LD6B9 | ;BRANCH IF NOT MODE 4, ELSE |
| D6A4 | | ldaa | #0xFF | ;LOAD 255 |
| D6A6 | | brset | *L0088,#0x01,LD6B6 | ;BRANCH IF "ALL PWM OUTPUTS COMMANDED ON" |
| D6AA | | ldaa | #0x08 | ;BIT 3? |
| D6AC | | anda | L0232 | ;MODE 4 CONTROL WORD |
| D6AF | | beq | LD6B9 | ;BRANCH IF BIT 3 CLEAR, ELSE |
| D6B1 | | ldaa | L0233 | ;LOAD COMMANDED DUTY CYCLE |
| D6B4 | | beq | LD723 | ;BRANCH IF Z, ELSE |
| D6B6 | LD6B6: | jmp | LD802 | ;GO STORE COMMANDED CCP DC |
| D6B9 | LD6B9: | ldx | #0x8EFE | ;INDEX |
| D6BC | | ldaa | L014E | ; |
| D6BF | | inca | | ;INCREMENT TIMER |
| D6C0 | | cmpa | L8EFC | ;0.1 SECOND |

| | | | | | |
|------|------------|-------|--------------------|--|---|
| D6C3 | | bcs | LD6D3 | | ;BRANCH IF < 100 MSEC, ELSE |
| D6C5 | | clra | | | ;CLEAR A |
| D6C6 | | ldab | L014F | | ; |
| D6C9 | | addb | L8EFD | | ;10 |
| D6CC | | bcc | LD6D0 | | ;BRANCH IF NO OVERFLOW, ELSE |
| D6CE | | ldab | #0xFF | | ;LOAD 255 |
| D6D0 | LD6D0:stab | | L014F | | ;STORE |
| D6D3 | LD6D3:staa | | L014E | | ;STORE TIMER |
| D6D6 | | ldaa | *L00F2 | | ;LOAD STARTUP COOLANT |
| D6D8 | | cmpa | 0x04,x | | ;8F02 - 85.25 DEG C |
| D6DA | | ldd | 0x00,x | | ;8EFE - 45 SECONDS |
| D6DC | | bcs | LD6E0 | | ;BRANCH IF < 85.25 DEG C, ELSE |
| D6DE | | ldd | 0x02,x | | ;25 SECONDS |
| D6E0 | LD6E0:subd | | *L00ED | | ;1 SECOND TIMER |
| D6E2 | | bhi | LD723 | | ;BRANCH IF TIMER > 25 OR 45 SECONDS, ELSE |
| D6E4 | | ldaa | *L00A7 | | ;FILTERED CTS |
| D6E6 | | cmpa | 0x05,x | | ;8F03 - 69.5 DEG C |
| D6E8 | | bcs | LD723 | | ;BRANCH IF < 69.5 DEG C, ELSE |
| D6EA | | brset | *L009B,#0x02,LD729 | | ;BRANCH IF DFCO ENABLED, ELSE |
| D6EE | | brclr | *L0096,#0x01,LD6F3 | | ;BRANCH IF CCP INACTIVE, ELSE |
| D6F2 | | inx | | | ;INCREMENT INDEX: 8EFF |
| D6F3 | LD6F3:ldaa | | 0x0A,x | | ;8F08 OR |
| | | | | | ;8F09 |
| D6F5 | | cmpa | *L00AA | | ;%TPS |
| D6F7 | | bcs | LD729 | | ;BRANCH IF > CAL, ELSE |
| D6F9 | | ldaa | *L00BA | | ;LOAD LV8 |
| D6FB | | ldab | L8F0A | | ;00 |
| D6FE | | bne | LD70B | | ;BRANCH IF NOT Z, ELSE |
| D700 | | ldd | *L00CD | | ;LOAD MAF (GM/SEC) |
| D702 | | cmpa | #0x40 | | ;64 GM/SEC |
| D704 | | bcs | LD709 | | ;BRANCH IF < 64, ELSE |
| D706 | | ldd | #0xFFFF | | ;LOAD MAX |
| D709 | LD709:lsld | | | | ;MUL BY 4 |
| D70A | | lsld | | | ; |
| D70B | LD70B:staa | | L0150 | | ;STORE MAF FOR CCP |
| D70E | | brset | *L009C,#0x80,LD734 | | ;BRANCH IF CLOSED LOOP, ELSE |
| D712 | | brset | *L0086,#0x02,LD73B | | ;BRANCH IF HOT OPEN LOOP, ELSE |
| D716 | | brclr | *L0099,#0x01,LD71C | | |
| D71A | | bra | LD73B | | |
| D71C | LD71C:ldab | | L8EE9 | | ;OPTION WORD |
| D71F | | bitb | #0x02 | | ;BIT 1? (TIS SET) |
| D721 | | bne | LD73B | | ;BRANCH IF NOT Z, ELSE |
| D723 | LD723:clr | | L014F | | ; |
| D726 | | clr | L014E | | ; |
| D729 | LD729:bclr | | *L0096,#0x01 | | ;CLEAR CCP ACTIVE |
| D72C | | clra | | | |
| D72D | | clrb | | | |
| D72E | | std | L014C | | ;CLEAR FILTERED CCP DUTY CYCLE |
| D731 | | jmp | LD7F3 | | ;GO STORE CCP DUTY CYCLE |
| D734 | LD734:ldab | | L8EE9 | | ;OPTION WORD |
| D737 | | bitb | #0x80 | | ;BIT 7 ? (TIS CLEAR) |
| D739 | | beq | LD747 | | ;BRANCH IF BIT 7 CLEAR, ELSE |
| D73B | LD73B:ldx | | #0x8F0B | | ;CCP DC VS AIRFLOW |
| D73E | | ldaa | L0150 | | ;AIRFLOW |
| D741 | | jsr | L99D7 | | ;2D LOOKUP |
| D744 | | jmp | LD7C0 | | ;GO FILTER AND STORE CCP DC |
| D747 | LD747:ldaa | | *L00F4 | | ;LOAD CCP DUTY CYCLE |
| D749 | | ldab | *L00BA | | ;LOAD LV8 |
| D74B | | cmpb | L8EED | | ;240 COUNTS |
| D74E | | bhi | LD785 | | ;BRANCH IF LV8 > 240, ELSE |
| D750 | | ldx | #0x8EEE | | ;INDEX |

| | | | |
|------|------------|--------------------|--|
| D753 | brset | *L008F,#0x40,LD758 | ;BRANCH IF IDLE COND MET, ELSE |
| D757 | inx | | ;INCREMENT INDEX |
| D758 | LD758:ldab | L0191 | ; |
| D75B | cmpb | 0x00,x | ;8EEE |
| | | | ;8EEF |
| D75D | bcc | LD765 | ;BRANCH IF > CAL, ELSE |
| D75F | incb | | ;INCREMENT TIMER |
| D760 | stab | L0191 | ;STORE |
| D763 | bra | LD789 | ; |
| D765 | LD765:ldy | *L00E4 | ;FINAL BPW |
| D768 | cpy | L8EF8 | ;COMPARE |
| D76C | bcs | LD778 | ;BRANCH IF < CAL, ELSE |
| D76E | ldab | *L00B3 | ;LOAD INTEGRATOR |
| D770 | cmpb | 0x02,x | ;8EF0 |
| | | | ;8EF1 |
| D772 | bcc | LD77F | ; |
| D774 | cmpb | 0x04,x | ;8EF2 |
| | | | ;8EF3 |
| D776 | bcc | LD785 | ; |
| D778 | LD778:suba | 0x06,x | ;8EF4 |
| | | | ;8EF5 |
| D77A | bcc | LD785 | ;BRANCH IF NO UNDERFLOW, ELSE |
| D77C | clra | | ;CLEAR |
| D77D | bra | LD785 | ; |
| D77F | LD77F:adda | 0x08,x | ;8EF6 |
| | | | ;8EF7 |
| D781 | bcc | LD785 | ;BRANCH OF NO OVERFLOW, ELSE |
| D783 | ldaa | #0xFF | ;LOAD 255 |
| D785 | LD785:clrb | | ; |
| D786 | stab | L0191 | ; |
| D789 | LD789:cmpa | L8EEA | ;31.25 % DC |
| D78C | bls | LD79C | ;BRANCH IF <= CAL, ELSE |
| D78E | ldab | *L00BF | ;MPH*3 |
| D790 | cmpb | L8EEB | ;255 MPH |
| D793 | bls | LD79C | ;BRANCH IF <= CAL, ELSE |
| D795 | ldab | *L00AF | ;LOAD FILTERED BLM |
| D797 | cmpb | L8EEC | ;0x00 |
| D79A | bhi | LD7C0 | ;BRANCH IF > 00, ELSE |
| D79C | LD79C:tab | | ;COPY CCP DC TO B REG |
| D79D | ldaa | L8EFA | ;14.8 %DC |
| D7A0 | brset | *L0090,#0x80,LD7B4 | ;BRANCH IF BLM < MIN WITH ACTIVE CCP, ELSE |
| D7A4 | ldaa | L8EFB | ;14.8 %DC |
| D7A7 | brset | *L008F,#0x40,LD7B4 | ;BRANCH IF IDLE COND MET, ELSE |
| D7AB | ldx | #0x8F1C | ;MAX CCP DC VS AIRFLOW |
| D7AE | ldaa | L0150 | ; |
| D7B1 | jsr | L99D7 | ;2D LOOKUP |
| D7B4 | LD7B4:cba | | ;COMPARE MAX DC AND CURRENT DC |
| D7B5 | bls | LD7C0 | ;BRANCH IF CURRENT > MAX, ELSE |
| D7B7 | tba | | ;COPY CURRENT CCP DC BACK TO A REG |
| D7B8 | cmpa | L8F2D | ;14.8 %DC |
| D7BB | bcc | LD7C0 | ;BRANCH IF >= 14.8 %, ELSE |
| D7BD | ldaa | L8F2D | ;LOAD |
| D7C0 | LD7C0:ldx | L014C | ;FILTERED CCP DC |
| D7C3 | cpx | #0xFA00 | ;64000d |
| D7C6 | bcc | LD7D7 | ;BRANCH IF > 64000, ELSE |
| D7C8 | psha | | ;SAVE CCP DC ON STACK |
| D7C9 | ldaa | #0xFF | ;NEW UNFILTERED VALUE |
| D7CB | ldx | #0x014C | ;OLD FILTERED VALUE |
| D7CE | ldy | #0x8F07 | ;FILTER COEFFICIENT ADDRESS |
| D7D2 | jsr | L99F4 | ;LAG FILTER |
| D7D5 | pulb | | ;GET BACK UNFILTERED CCP DC |
| D7D6 | mul | | ;A * B |

| | | | |
|------|--------------|-----------------------|---|
| D7D7 | LD7D7: cmpa | L8F2E | ;14.8 %DC |
| D7DA | bcc | LD7DF | ;BRANCH IF > 14.8 %DC, ELSE |
| D7DC | ldaa | L8F2E | ;LOAD |
| D7DF | LD7DF: brclr | *L008F, #0x40, LD7F3 | ;BRANCH IF IDLE COND NOT MET, ELSE |
| D7E3 | ldab | L018E | ;FILTERED IATMAT |
| D7E6 | cmpb | L8F05 | ; |
| D7E9 | bcc | LD7F3 | ;BRANCH IF > CAL, ELSE |
| D7EB | cmpa | L8F06 | ; |
| D7EE | bls | LD7F3 | ;BRANCH IF <= CAL, ELSE |
| D7F0 | ldaa | L8F06 | ; |
| D7F3 | LD7F3: cmpa | L014F | ; |
| D7F6 | bls | LD7FB | ;BRANCH IF <= L014F, ELSE |
| D7F8 | ldaa | L014F | ;LOAD |
| D7FB | LD7FB: brclr | *L0095, #0x82, LD802 | ;BRANCH IF HUD DEVICE NOT CONNECTED, ELSE |
| D7FF | jsr | L5012 | ;HUD |
| D802 | LD802: staa | *L00F4 | ;STORE NEW CCP DC |
| D804 | bne | LD80B | ;BRANCH IF NOT Z, ELSE |
| D806 | bclr | *L0096, #0x01 | ;CLEAR CCP ACTIVE |
| D809 | bra | LD80E | |
| D80B | LD80B: bset | *L0096, #0x01 | ;SET CCP ACTIVE |
| D80E | LD80E: ldx | #0xDFFF | ;MAX DUTY CYCLE |
| D811 | ldy | #0x9153 | ; |
| D815 | brset | 0x00, y, #0x80, LD81D | ;BIT 7 9153 |
| D81A | stx | L3DDA | ;SHIFT SOLN B PWM OUTPUT |
| D81D | LD81D: brset | 0x00, y, #0x40, LD825 | ;BIT 6 9153 |
| D822 | stx | L3DD8 | ;SHIFT SOLN A PWM OUTPUT |
| D825 | LD825: brset | 0x00, y, #0x20, LD82D | ;BIT 5 9153 |
| D82A | stx | L3DD4 | ;TCC PWM OUTPUT |
| D82D | LD82D: brset | 0x00, y, #0x08, LD835 | ;BIT 3 9153 |
| D832 | stx | L3FCC | ;STORE DEGR SOLN C PWM OUTPUT |
| D835 | LD835: brset | 0x00, y, #0x10, LD83D | ;BIT 4 9153 |
| D83A | stx | L3FD8 | ;STORE OIL LEVEL LIGHT PWM OUTPUT |
| D83D | LD83D: brset | 0x00, y, #0x04, LD845 | ;BIT 2 9153 |
| D842 | stx | L3FEA | ;STORE DEGR SOLN B PWM OUTPUT |
| D845 | LD845: brset | 0x00, y, #0x02, LD84D | ;BIT 1 9153 |
| D84A | stx | L3DD2 | ;STORE DEGR SOLN A PWM OUTPUT |
| D84D | LD84D: brset | 0x00, y, #0x01, LD855 | ;BIT 0 9153 |
| D852 | bset | *L0089, #0x08 | ;ENABLE TCC SOLENOID |
| D855 | LD855: ldaa | L8016 | ;OPTION WORD - TIS SET |
| D858 | beq | LD86E | ;BRANCH IF Z, ELSE |
| D85A | ldaa | L01E7 | ;DELAY TIMER |
| D85D | beq | LD864 | ;BRANCH IF Z, ELSE |
| D85F | dec | L01E7 | ;DECREMENT TIMER |
| D862 | bra | LD86B | ; |
| D864 | LD864: brset | *L0094, #0x20, LD86B | ;BRANCH IF LOW OIL, ELSE |
| D868 | ldx | #0xD000 | ;0 DC |
| D86B | LD86B: stx | L3FD8 | ;OIL LEVEL LIGHT PWM CONTROL |
| D86E | LD86E: clra | | ;CLEAR A |
| D86F | ldab | *L00F4 | ;CCP DUTY CYCLE |
| D871 | beq | LD878 | ;BRANCH IF Z, ELSE |
| D873 | lsl | | ;MUL BY 8 |
| D874 | lsl | | |
| D875 | lsl | | |
| D876 | orab | #0x07 | ;SET BITS 0,1,2 |
| D878 | LD878: oraa | #0xB0 | ; |
| D87A | std | L3FD6 | ;CCP PWM OUTPUT |
| D87D | brclr | *L0084, #0x80, LD897 | ;BRANCH IF NOT MODE 4, ELSE |
| D881 | brset | *L0088, #0x01, LD894 | ;BRANCH IF "ALL PWM OUTPUTS COMMANDED ON" |
| D885 | ldaa | #0x01 | ;BIT 0? |
| D887 | anda | L022E | ;CONTROL BYTE FOR MODE 4 |
| D88A | beq | LD897 | ;BRANCH IF NOT BIT 0, ELSE |
| D88C | bclr | *L0091, #0x01 | ;TURN OFF SES LIGHT |

```

D88F      anda    L022F      ;CONTROL BYTE FOR MODE 4
D892      beq     LD897      ;BRANCH IF NOT BIT 0, ELSE
D894      LD894:bset  *L0091,#0x01      ;TURN ON SES LIGHT
D897      LD897:ldd    L3FFC      ;CPU CNTL REG
D89A      andb    #0xF3      ;CLEAR BITS 2 AND 3
D89C      ld      #0xD000      ;0 DC
D89F      brclr   *L0096,#0x02,LD8A8    ;BRANCH IF FAN #1 OFF, ELSE
D8A3      orab    #0x04      ;SET BIT 2
D8A5      ld      #0xDFFF      ;MAX DUTY CYCLE
D8A8      LD8A8:tst   L800D      ;OPTION WORD - TIS SET
D8AB      bne     LD8B2      ;BRANCH IF NOT Z, ELSE
D8AD      andb    #0xFB      ;CLEAR BIT 2
D8AF      stx     L3FDA      ;FAN #1 PWM OUTPUT
D8B2      LD8B2:tst   L800E      ;OPTION WORD - TIS SET
D8B5      beq     LD8D1      ;BRANCH IF Z, ELSE
D8B7      ld      #0xD000      ;0 DUTY CYCLE
D8BA      brclr   *L0096,#0x10,LD8C1    ;BRANCH IF FAN #2 OFF, ELSE
D8BE      ld      #0xDFFF      ;MAX DUTY CYCLE
D8C1      LD8C1:tst   L800E      ;OPTION WORD - TIS SET
D8C4      bpl     LD8CB      ;BRANCH IF CLEAR, ELSE
D8C6      stx     L3FDA      ;STORE FAN PWM OUTPUT
D8C9      bra     LD8D1

D8CB      LD8CB:brclr *L0096,#0x10,LD8D1    ;BRANCH IF FAN #2 OFF, ELSE
D8CF      orab    #0x04      ;SET BIT 2 -> TURN FAN ON
D8D1      LD8D1:brset *L0095,#0x10,LD8D9    ;BRANCH IF BATT VOLTS < 4.0, ELSE
D8D5      brset   *L0091,#0x01,LD8DB      ;BRANCH IF SES LIGHT ON
D8D9      LD8D9:orab #0x08      ;SET BIT 3
D8DB      LD8DB:std   L3FFC      ;CPU CTRL REG
D8DE      ldaa    L8010      ;OPTION WORD - TRANS TYPE - TIS CLEAR
D8E1      bpl     LD8F0      ;BRANCH IF AUTO TRANS, ELSE
D8E3      ld      #0xD000      ;0 DUTY CYCLE
D8E6      brclr   *L00A2,#0x20,LD8ED      ;BRANCH IF SHIFT LIGHT OFF, ELSE
D8EA      ld      #0xDFFF      ;MAX DUTY CYCLE
D8ED      LD8ED:stx   L3DD4      ;STORE TCC PWM OUTPUT
D8F0      LD8F0:brset *L0095,#0x10,LD901    ;BRANCH IF BATT VOLTS < 4.0, ELSE
D8F4      ld      #0xF000      ;
D8F7      brset   *L0097,#0x20,LD904      ;BRANCH IF HIGH BATT VOLTS DETECTED, ELSE
D8FB      ldaa    *L0083      ;BATT VOLTS
D8FD      cmpa    #0x5A      ;9.0 VOLTS
D8FF      bls     LD904      ;BRANCH IF <= 9.0 VOLTS, ELSE
D901      LD901:ldx   #0xFFFF      ;
D904      LD904:stx   L3DCC      ;ENABLE PWM OUTPUTS
D907      ld      #0xDFFF      ;MAX DUTY CYCLE
D90A      stx     L3DEA      ;
D90D      rts

;-----
; FOR TESTING PWM OUTPUTS (MODE 4) A REG CONTAINS A BIT
;-----
D90E      LD90E:brset *L0097,#0x20,LD92C    ;BRANCH IF HIGH BATT VOLTS DETECTED, ELSE
D912      brset   *L0095,#0x10,LD92C      ;BRANCH IF BATT VOLTS < 4.0, ELSE
D916      brset   *L0088,#0x01,LD928      ;BRANCH IF "ALL PWM OUTPUTS COMMANDED ON"
D91A      brclr   *L0084,#0x80,LD930      ;BRANCH IF NOT MODE 4, ELSE
D91E      anda    L022C      ;MODE 4 CONTROL BYTE
D921      beq     LD930      ;BRANCH IF Z, ELSE
D923      anda    L022D      ;MODE 4 CONTROL BYTE
D926      beq     LD92C      ;BRANCH IF Z, ELSE
D928      LD928:ldaa #0xFF      ;255 %11111111
D92A      bra     LD931      ;TEST A

D92C      LD92C:ldaa #0x7F      ;127 %01111111
D92E      bra     LD931      ;TEST A

```

```

D930      LD930:clra                      ;0
D931      LD931:tsta                      ;TEST A
D932              rts                      ;RETURN

;-----
; RELATED TO MODE 4
;-----
D933      LD933:brset    *L0097,#0x20,LD951      ;BRANCH IF HIGH BATT VOLTS, ELSE
D937              brset    *L0095,#0x10,LD951      ;BRANCH IF BATT VOLTS < 4.0, ELSE
D93B              brset    *L0088,#0x01,LD94D      ;BRANCH IF "ALL PWM OUTPUTS COMMANDED ON"
D93F              brclr    *L0084,#0x80,LD955      ;BRANCH IF NOT MODE 4, ELSE
D943              anda     L022E                  ;MODE 4 CONTROL BYTE
D946              beq      LD955                  ;BRANCH IF Z, ELSE
D948              anda     L022F                  ;MODE 4 CONTROL BYTE
D94B              beq      LD951                  ;BRANCH IF Z, ELSE
D94D      LD94D:ldaa     #0xFF                    ;
D94F              bra      LD956

D951      LD951:ldaa     #0x7F
D953              bra      LD956

D955      LD955:clra
D956      LD956:tsta                      ;TEST
D957              rts                      ;RETURN

;-----
; BRANCH HERE FROM SEGMENT TABLE - 100 MSEC ROUTINE
; A.I.R. ROUTINE
;-----
D958              ldaa     L800A                  ;OPTION WORD - A.I.R. PUMP - TIS CLEAR
D95B              bne      LD960                  ;BRANCH IF NZ, ELSE
D95D              jmp      LD9C6                  ;JUMP OVER

D960      LD960:ldaa     #0x08                    ;BIT 3
D962              jsr      LD90E                  ;TEST MODE 4 CONTROL WORDS
D965              beq      LD96B                  ;BRANCH IF Z, ELSE
D967              bpl      LD9C6                  ;BRANCH IF PL, ELSE
D969              bra      LD9B2                    ;

D96B      LD96B:brset    *L0086,#0x80,LD99A      ;BRANCH IF ENGINE RUNNING, ELSE
D96F              bset     *L0098,#0x80          ;SET A.I.R. ENABLED
D972              ldaa     *L00F0                ;IATMAT
D974              cmpa     L8EDC                  ;
D977              bls      LD9C3
D979              cmpa     L8EDD
D97C              bcc      LD9C3
D97E              ldaa     *L00A6                ;LOAD CLNT TEMP (DEFAULTED)
D980              cmpa     L8EDE
D983              bcs      LD9C3
D985              cmpa     L8EDF
D988              bhi      LD9C3
D98A              ldx      L8EE3
D98D              cmpa     L8EE0
D990              bhi      LD995
D992              ldx      L8EE1
D995      LD995:stx      L0319
D998              bra      LD9C6                  ;BRANCH OUT

D99A      LD99A:brclr    *L0098,#0x80,LD9C6      ;BRANCH IF A.I.R. DISABLED, ELSE
D99E              ldx      *L0032                ;RUN TIME
D9A0              cpx      L8EE5                  ;
D9A3              bcs      LD9C6                  ;BRANCH IF < CAL, ELSE
D9A5              brset    *L009B,#0x20,LD9C6      ;BRANCH IF PE MODE, ELSE
D9A9              brset    *L009C,#0x80,LD9C3      ;BRANCH IF CLOSED LOOP, ELSE

```

```

D9AD      cpx      L0319
D9B0      bhi      LD9C3
D9B2      LD9B2:ldx  L0317                      ;TIMER
D9B5      cpx      L8EE7                      ;TIME LIMIT
D9B8      beq      LD9C3                      ;BRANCH IF = CAL, ELSE
D9BA      inx                      ;INCREMENT TIMER
D9BB      stx      L0317                      ;STORE
D9BE      ldd      #0x0000                    ;0 DC
D9C1      bra      LD9C9

D9C3      LD9C3:bclr  *L0098,#0x80              ;DISABLE A.I.R.

;-----
; JUMP HERE IF 800A OPTION FLAG CLEAR - TIS CLEAR
;-----
D9C6      LD9C6:ldd   #0x7FFF                    ;LOAD MAX DC
D9C9      LD9C9:std   L3DCE                      ;A.I.R. SOLN OUTPUT
D9CC      ldx      #0x97FA                      ;INDEX
D9CF      ldaa     *L00A6                      ;LOAD CLNT TEMP (DEFAULTED)
D9D1      cmpa     0x06,x                      ;9800
D9D3      bcs      LD9F0                      ;BRANCH IF < CAL, ELSE
D9D5      brclr    *L00DA,#0x80,LD9DA          ;
D9D9      inx                      ;97FB
D9DA      LD9DA:bclr  *L00DA,#0x80              ;
D9DD      ldaa     *L00AC                      ;LOAD RPM/25
D9DF      cmpa     0x00,x                      ;97FA OR
                                           ;97FB
D9E1      bls      LD9F0                      ;BRANCH IF <= CAL, ELSE
D9E3      cmpa     0x02,x                      ;97FC OR
                                           ;97FD
D9E5      bhi      LD9F0                      ;BRANCH IF > CAL, ELSE
D9E7      ldaa     *L00AA                      ;%TPS
D9E9      cmpa     0x04,x                      ;97FE OR
                                           ;97FF
D9EB      bls      LD9F0                      ;BRANCH IF <= CAL, ELSE
D9ED      bset     *L00DA,#0x80                ;
D9F0      LD9F0:ldx  #0x0000                    ;MIN
D9F3      brclr    *L00DA,#0x80,LD9FA          ;BRANCH IF NOT BIT 7, ELSE
D9F7      ldx      #0x7FFF                      ;MAX
D9FA      LD9FA:stx  L3DF2                      ;
D9FD      pshy                      ;DELAY
D9FF      puly                      ;
DA01      ldx      L3DEC                      ;
DA04      pshy                      ;DELAY
DA06      puly                      ;
DA08      stx      L3DF0                      ;TRANSFER CONTENTS OF L3DEC TO L3DF0
DA0B      mul                      ;DELAY
DA0C      mul
DA0D      mul
DA0E      nop
DA0F      nop
DA10      sei                      ;HOLD INTERRUPTS
DA11      ldd      L3DFC                      ;? 2ND MCU CNTL REG
DA14      pshy                      ;DELAY
DA16      puly
DA18      ldx      L3DCE
DA1B      pshx                      ;SAVE
DA1C      ldx      #0x0000                      ;DELAY
DA1F      ldx      #0x0000
DA22      stx      L3DD0                      ;CLEAR L3DD0
DA25      pshy                      ;DELAY
DA27      puly
DA29      stx      L3DCE                      ;CLEAR L3DCE
DA2C      orab     #0x08                      ;SET BIT 3

```

```

DA2E      oraa    #0x04                ;SET BIT 2
DA30      psha                    ;DELAY
DA31      pula
DA32      std     L3DFC                ;? 2ND MCU CNTL REG
DA35      andb    #0xF7                ;CLEAR BIT 3
DA37      anda    #0xFB                ;CLEAR BIT 2
DA39      psha                    ;DELAY
DA3A      pula
DA3B      std     L3DFC                ;STORE ? 2ND MCU CNTL REG
DA3E      psha                    ;DELAY
DA3F      pshb
DA40      idiv
DA41      mul
DA42      pulb
DA43      pula
DA44      nop
DA45      ldx     #0x7FFF                ;
DA48      stx     L3DD0                ;SET L3DD0 TO MAX
DA4B      pulx                    ;RESTORE X
DA4C      psha                    ;DELAY
DA4D      pula
DA4E      stx     L3DCE                ;SET L3DCE TO MAX
DA51      orab    #0x08                ;SET BIT 3
DA53      oraa    #0x04                ;SET BIT 2
DA55      psha                    ;DELAY
DA56      pula
DA57      std     L3DFC                ;STORE ? 2ND MCU CNTL REG
DA5A      andb    #0xF7                ;CLEAR BIT 3
DA5C      anda    #0xFB                ;CLEAR BIT 2
DA5E      psha                    ;DELAY
DA5F      pula
DA60      std     L3DFC                ;STORE ? 2ND MCU CNTL REG
DA63      cli                    ;CLEAR AND ALLOW INTERRUPTS
DA64      brset   *L0095,#0x40,LDA7A    ;BRANCH IF COLD ENG REV LIMIT ENABLED, ELSE
DA68      ldaa    *L00F2                ;LOAD STARTUP COOLANT
DA6A      ldx     #0x9801                ;TABLE ADDRESS
DA6D      jsr     L99D7                ;2D LOOKUP
DA70      tab                    ;TRANSFER LOOKUP RESULT TO B
DA71      clra                    ;CLEAR A
DA72      cpd     *L0032                ;RUN TIME
DA75      bhi     LDA7A                ;BRANCH IF > RUNTIME, ELSE
DA77      bset    *L0095,#0x40          ;SET COLD ENG REV LIMIT ENABLED
DA7A      LDA7A:ldaa *L00F2                ;LOAD STARTUP COOLANT
DA7C      ldx     #0x9812                ;TABLE ADDRESS/INDEX
DA7F      jsr     L99D7                ;2D LOOKUP
DA82      staa    L01E5                ;STORE LOOKUP RESULT
DA85      ldx     #0x9823                ;INDEX
DA88      ldab    *L00E9                ;TRANNNY GEAR BYTE
DA8A      subb    #0x02                ;
DA8C      bcc     LDA8F                ;BRANCH IF >= 2ND GEAR, ELSE
DA8E      clrb                    ;
DA8F      LDA8F:ldaa #0x04                ;4
DA91      mul                    ;4 * TRANNNY GEAR BYTE - 2
DA92      abx                    ;ADD TO ADDRESS
DA93      ldaa    *L00F2                ;LOAD STARTUP COOLANT
DA95      cmpa    #0x40                ;8 DEG C
DA97      bcs     LDA9B                ;BRANCH IF < 8 DEG C, ELSE
DA99      ldaa    #0x40                ;LOAD 0x40
DA9B      LDA9B:ldab #0x10                ;OFFSET
DA9D      jsr     L99D3                ;2D LOOKUP
DAA0      staa    L01E6                ;STORE - ALL 00s
DAA3      LDAA3:rts
;-----

```

```

; BRANCH HERE FROM SEGMENT TABLE - 100 MSEC ROUTINE
;
;-----
DAA4      ldaa    *L00F2                ;LOAD STARTUP COOLANT
DAA6      lsra                    ;RESCALE
DAA7      ldx     #0x8F36                ;EGR DISABLE TIME VS SU CLNT
DAAA      jsr     L99D7                ;2D LOOKUP
DAAD      staa    L0185                ;EGR DISABLED TIME
DAB0      ldaa    *L00A7                ;FILTERED CTS
DAB2      cmpa    #0xC0                ;104 DEG C
DAB4      bls     LDAB8                ;BRANCH IF <= 104 DEG C, ELSE
DAB6      ldaa    #0xC0                ;LOAD 104
DAB8      LDAB8:ldab    #0x40            ;OFFSET
DABA      ldx     #0x8F3F                ;EGR DC MULTIPLIER VS COOLANT TEMP
DABD      jsr     L99D3                ;2D LOOKUP
DAC0      staa    L0186                ;STORE EGR COOLANT MULTIPLIER
DAC3      ldaa    *L00F0                ;IATMAT
DAC5      ldx     #0x8C4E                ;
DAC8      jsr     L99CC                ;2D LOOKUP
DACB      staa    L0187                ;IAT CORRECTION TERM TO MAF
DACE      ldaa    *L00F0                ;IATMAT
DAD0      ldx     #0x8C65                ;
DAD3      jsr     L99CC                ;2D LOOKUP
DAD6      staa    L0188                ;IAT CORRECTION TERM TO MAF
DAD9      ldx     #0x8C74                ;
DADC      ldaa    *L005A                ;FILTERED ENGINE TORQUE
DADE      ldab    #0x70                ;OFFSET
DAE0      cmpa    #0xC0                ;
DAE2      bls     LDAE6                ;BRANCH IF <= UPPER LIMIT, ELSE
DAE4      ldaa    #0xC0                ;LOAD
DAE6      LDAE6:jsr    L99D3            ;2D LOOKUP
DAE9      psha                    ;SAVE LOOKUP RESULT
DAEA      ldaa    *L0076                ;THIS NOT USED
DAEC      ldx     #0x8C96                ;(ALL 00s)
DAEF      jsr     L99CC                ;2D LOOKUP
DAF2      pulb                    ;GET PREV LOOKUP RESULT
DAF3      aba                    ;ADD BOTH LOOKUP RESULTS
DAF4      bcc     LDAF8                ;BRANCH IF NO OVERFLOW, ELSE
DAF6      ldaa    #0xFF                ;LOAD 255
DAF8      LDAF8:staa    L0175            ;STORE IAC STEPS
DAFB      ldaa    *L00A6                ;LOAD CLNT TEMP (DEFAULTED)
DAFD      ldx     #0x8AAB                ;? ASYNC FACTOR VS COOLANT TEMP
DB00      jsr     L99D7                ;2D LOOKUP
DB03      staa    L031F                ;STORE ASYNC FACTOR
DB06      ldaa    L0183                ;BAD CYL ID
DB09      beq     LDB0E                ;BRANCH IF Z, ELSE
DB0B      staa    L0184                ;STORE PREV BAD CYL ID
DB0E      LDB0E:ldaa    *L0065            ;FILTERED LOW AIRFLOW
DB10      cmpa    L8DDA                ;COMPARE MAX
DB13      bhi     LDB1A                ;BRANCH IF > MAX, ELSE
DB15      cmpa    L8DDC                ;MIN
DB18      bcc     LDB1F                ;BRANCH IF > MIN, ELSE
DB1A      LDB1A:ldaa    L8DD5            ;LOAD MIN VALUE
DB1D      staa    *L0065                ;STORE FILTERED LOW AIRFLOW
DB1F      LDB1F:ldaa    *L0067            ;FILTERED LOW AIRFLOW
DB21      cmpa    L8DDA                ;COMPARE MAX
DB24      bhi     LDB2B                ;BRANCH IF > MAX, ELSE
DB26      cmpa    L8DDC                ;COMPARE MIN
DB29      bcc     LDB30                ;BRANCH IF > MIN, ELSE
DB2B      LDB2B:ldaa    L8DD6            ;MIN VALUE
DB2E      staa    *L0067                ;STORE FILTERED LOW AIRFLOW
DB30      LDB30:brset    *L0086,#0x80,LDB40 ;BRANCH IF ENGINE RUNNING, ELSE
DB34      ldaa    *L00A6                ;LOAD CLNT TEMP (DEFAULTED)
DB36      staa    *L00F2                ;STORE STARTUP COOLANT

```

| | | | |
|------|-------------|--------------------|---|
| DB38 | bclr | *L009E,#0xFF | ;CLEAR EVERYTHING |
| DB3B | ldaa | L86C6 | ;INITIAL O2 A/D COUNTS |
| DB3E | staa | *L00A9 | ;MINOR LOOP O2 A/D COUNTS (mV) |
| DB40 | LDB40:bclr | *L009B,#0x80 | ;CLEAR INCREMENT ODOMETER |
| DB43 | ldx | L0252 | ;? ACCUMULATED DISTANCE |
| DB46 | cpx | L9778 | ; |
| DB49 | bls | LDB62 | ;BRANCH IF <= CAL, ESE |
| DB4B | bset | *L009B,#0x80 | ;SET INCREMENT ODOMETER |
| DB4E | ldd | *L0072 | |
| DB50 | addd | #0x0001 | |
| DB53 | cmpa | #0x07 | |
| DB55 | bhi | LDB62 | |
| DB57 | std | *L0072 | |
| DB59 | ldd | L0252 | ;? ACCUMULATED DISTANCE |
| DB5C | subd | L9778 | |
| DB5F | std | L0252 | ;? ACCUMULATED DISTANCE |
| DB62 | LDB62:ldx | #0x8F70 | ; |
| DB65 | ldaa | *L005C | ;FILTERED ENGINE TORQUE |
| DB67 | cmpa | #0xB8 | ; |
| DB69 | bls | LDB6D | ;BRANCH IF <= 0xB8, ELSE |
| DB6B | ldaa | #0xB8 | ; |
| DB6D | LDB6D:suba | #0x70 | ;OFFSET |
| DB6F | bcc | LDB72 | ;BRANCH IF NO OVERFLOW, ELSE |
| DB71 | clra | | ;CLEAR |
| DB72 | LDB72:jsr | L99CC | ;2D LOOKUP |
| DB75 | staa | L01B6 | ;STORE 8F70 TABLE LOOKUP RESULT |
| DB78 | ldaa | *L005C | ;FILTERED ENGINE TORQUE |
| DB7A | cmpa | L86A6 | ;170 |
| DB7D | bls | LDB9C | ;BRANCH IF <= CAL, ELSE |
| DB7F | ldaa | *L005A | ;FILTERED ENGINE TORQUE |
| DB81 | cmpa | L86A7 | ;140 |
| DB84 | bcc | LDB9C | ;BRANCH IF >= CAL, ELSE |
| DB86 | brclr | *L009B,#0x80,LDBA1 | ;BRANCH IF INCREMENT ODOMETER CLEAR, ELSE |
| DB8A | ldd | *L0074 | |
| DB8C | addd | #0x0001 | |
| DB8F | std | *L0074 | |
| DB91 | cpd | L86A8 | |
| DB95 | bcs | LDBA1 | |
| DB97 | ldd | L8698 | |
| DB9A | std | *L005A | ;FILTERED FILTERED ENGINE TORQUE |
| DB9C | LDB9C:ldd | #0x0000 | |
| DB9F | std | *L0074 | ; |
| DBA1 | LDBA1:clra | | |
| DBA2 | brset | *L008C,#0x01,LDBA8 | ;BRANCH IF NOT 2 ND GEAR START (SNOW SHFT) |
| DBA6 | oraa | #0x01 | ;SET BIT 0 |
| DBA8 | LDBA8:brclr | *L00A4,#0x02,LDBAE | ;BRANCH IF DEGR SOLN A OFF, ELSE |
| DBAC | oraa | #0x02 | ;SET BIT 1 |
| DBAE | LDBAE:brclr | *L00A4,#0x04,LDBB4 | ;BRANCH IF DEGR SOLN B OFF, ELSE |
| DBB2 | oraa | #0x04 | ;SET BIT 2 |
| DBB4 | LDBB4:brclr | *L00A4,#0x08,LDBBA | ;BRANCH IF DEGR SOLN C OFF, ELSE |
| DBB8 | oraa | #0x08 | ;SET BIT 3 |
| DBBA | LDBBA:brclr | *L00A2,#0x20,LDBC0 | ;BRANCH IF SHIFT LIGHT OFF, ELSE |
| DBBE | oraa | #0x10 | ;SET BIT 4 |
| DBC0 | LDBC0:brclr | *L0096,#0x20,LDBC6 | ;BRANCH IF PSPS INPUT LOW (NOT CRAMP), ELSE |
| DBC4 | oraa | #0x20 | ;SET BIT 5 |
| DBC6 | LDBC6:staa | L02A9 | ;STORE ALDL STATUS WORD |
| DBC9 | clra | | ; |
| DBCA | brclr | *L0092,#0x04,LDBD0 | ;BRANCH IF VATS OKAY, ELSE |
| DBCE | oraa | #0x01 | ;SET VATS FUEL SHUT OFF |
| DBD0 | LDBD0:brclr | *L0097,#0x01,LDBD6 | ;BRANCH IF SMC NOT ENGAGED, ELSE |
| DBD4 | oraa | #0x02 | ;SET SMC ENGAGED REPEAT FLAG |
| DBD6 | LDBD6:brclr | *L0099,#0x10,LBDC6 | ;BRANCH IF CRUISE ALLOWED, ELSE |
| BD8A | oraa | #0x04 | ;SET SMC INHBITIED REPEAT FLAG |
| LBDC | LDBDC:brclr | *L0086,#0x80,LDBE2 | ;BRANCH IF ENGINE NOT RUNNING, ELSE |

```

DBE0      oraa      #0x08      ;SET ENGINE RUNNING
DBE2      LDBE2:brclr *L0094,#0x20,LDBE8      ;BRANCH IF NOT LOW OIL, ELSE
DBE6      oraa      #0x10      ;SET LOW OIL LIGHT ON REPEAT FLAG
DBE8      LDBE8:brclr *L0098,#0x80,LDBEE      ;BRANCH IF A.I.R. NOT ENABLED, ELSE
DBEC      oraa      #0x20      ;SET A.I.R. ENABLED
DBEE      LDBEE:brclr *L009C,#0x40,LDBF4      ;BRANCH IF LEAN, ELSE
DBF2      oraa      #0x40      ;SET RICH FLAG
DBF4      LDBF4:brclr *L009C,#0x80,LDBFA      ;BRANCH IF NOT CLOSED LOOP
DBF8      oraa      #0x80      ;SET CLOSED LOOP REPEAT FLAG
DBFA      LDBFA:staa  L01A9      ;STORE ALDL STATUS WORD
DBFD      bclr      *L0094,#0x80      ;CLEAR A/C CLUTCH ON
DC00      brclr     *L0086,#0x20,LDC07      ;BRANCH A/C CLUTCH ON, ELSE
DC04      bset      *L0094,#0x80      ;SET BIT 7
DC07      LDC07:ldaa  *L0011      ;MINOR LOOP COUNTER
DC09      cmpa      #0x03      ;? 18.75 MSEC
DC0B      bne       LDC17      ;BRANCH IF NOT, ELSE
DC0D      ldaa      L029E      ;O2 XCOUNTS
DC10      staa      L015E      ;O2 XCOUNTS IN LAST SECOND
DC13      clra      ;CLEAR
DC14      staa      L029E      ;O2 XCOUNTS
DC17      LDC17:rts

;-----
; BRANCH HERE FROM SEGMENT TABLE
;
;-----

DC18      brclr     *L0095,#0x02,LDC21      ;BRANCH IF HUD DEVICE NOT CONNECTED, ELSE
DC1C      jsr       L5A03
DC1F      bra       LDC28      ;RETURN

DC21      LDC21:brclr *L0095,#0x80,LDC28      ;BRANCH IF HUD DEVICE NOT CONNECTED, ELSE
DC25      jsr       L5006
DC28      LDC28:rts

;-----
; BRANCH HERE FROM SEGMENT TABLE
;
;-----

DC29      brset     *L0095,#0x02,LDC34      ;BRANCH IF HUD DEVICE NOT CONNECTED, ELSE
DC2D      brclr     *L0095,#0x80,LDC34      ;BRANCH IF HUD DEVICE NOT CONNECTED, ELSE
DC31      jsr       L5006      ;
DC34      LDC34:jmp  LEE23      ;TO TCC ROUTINE

;-----
; BRANCH HERE IF OPTN WORD 8010 SET (MANUAL TRANS)
; SHIFT LIGHT CODE
; 238 BYTES
;-----

DC37      LDC37:ldaa  #0x04      ;BIT 2
DC39      jsr       LD90E      ;GO CHECK MODE 4 CONTROL WORDS
DC3C      beq       LDC46      ;
DC3E      bpl       LDC43      ;
DC40      jmp       LDD1A      ;GO TURN ON SHIFT LIGHT

DC43      LDC43:jmp  LDD22      ;GO TURN OFF SHIFT LIGHT

DC46      LDC46:ldaa  *L00A6      ;LOAD CLNT TEMP (DEFAULTED)
DC48      cmpa      L97A9      ;COOLANT THRESHOLD FOR E-LITE ENABLE
DC4B      bcs       LDC8E      ;BRANCH IF < CAL, ELSE
DC4D      ldaa      *L00BD      ;FILTERED MPH
DC4F      cmpa      L97AD      ;
DC52      bcs       LDC90      ;BRANCH IF < CAL, ELSE
DC54      ldaa      *L00AA      ;%TPS
DC56      cmpa      L97AE

```

| | | | |
|------|------------|--------------------|---|
| DC59 | bc | LDC90 | ;BRANCH IF < CAL, ELSE |
| DC5B | ldaa | L0328 | ; |
| DC5E | anda | #0x1F | ; |
| DC60 | ldab | *L008E | ;FMD INPUT STATUS WORD |
| DC62 | andb | #0x1F | ; |
| DC64 | cba | | ;COMPARE |
| DC65 | bne | LDC90 | ;BRANCH IF THEY'RE NOT EQUAL, ELSE |
| DC67 | brset | *L008E,#0x01,LDC90 | ;BRANCH IF P/N MODE, ELSE |
| DC6B | brclr | *L008E,#0x10,LDC90 | ;BRANCH IF 5 TH GEAR, ELSE |
| DC6F | ldab | #0x10 | ;OFFSET |
| DC71 | ldx | #0x97AF | ;INDEX |
| DC74 | brclr | *L008E,#0x08,LDC83 | ;BRANCH IF 4 TH GEAR, ELSE |
| DC78 | abx | | ;97BF |
| DC79 | brclr | *L008E,#0x04,LDC83 | ;BRANCH IF 3 RD GEAR, ELSE |
| DC7D | abx | | ;97CF |
| DC7E | brclr | *L008E,#0x02,LDC83 | ;BRANCH IF 2 ND GEAR, ELSE |
| DC82 | abx | | ;97DF |
| DC83 | LDC83:ldaa | L0326 | ;SHIFT LITE TIMER |
| DC86 | cmpa | 0x0D,x | ;97BC OR |
| | | | ;97CC OR |
| | | | ;97DC OR |
| | | | ;97EC |
| DC88 | bcc | LDC92 | ;BRANCH IF >= CAL, ELSE |
| DC8A | inca | | ;INCREMENT TIMER |
| DC8B | staa | L0326 | ;STORE |
| DC8E | LDC8E:bra | LDD03 | |
| DC90 | LDC90:bra | LDD00 | |
| DC92 | LDC92:ldaa | *L00AC | ;LOAD RPM/25 |
| DC94 | cmpa | #0x28 | ; |
| DC96 | bcs | LDD03 | |
| DC98 | ldaa | *L00AA | ;%TPS |
| DC9A | suba | L0110 | ; |
| DC9D | bcc | LDCA5 | ;BRANCH IF TPS INCREASING, ELSE |
| DC9F | nega | | ;INVERT RESULT |
| DCA0 | cmpa | L97A8 | ;TURN OFF E-LITE IF NEG D-TPS > THIS |
| DCA3 | bhi | LDD03 | ;BRANCH IF > CAL, ELSE |
| DCA5 | LDCA5:ldaa | 0x0C,x | ;97BB OR |
| | | | ;97CB OR |
| | | | ;97DB OR |
| | | | ;97EB |
| DCA7 | brclr | *L00A2,#0x20,LDCB1 | ;BRANCH IF SHIFT LIGHT OFF, ELSE |
| DCAB | suba | L97AA | ;HYST FOR MINIMUM TPS FOR SHFT LITE ON |
| DCAE | bcc | LDCB1 | |
| DCB0 | clra | | |
| DCB1 | LDCB1:cmpa | *L00AA | ;%TPS |
| DCB3 | bhi | LDD03 | ;BRANCH IF > %TPS, ELSE |
| DCB5 | ldaa | 0x0B,x | ;97BA OR |
| | | | ;97CA OR |
| | | | ;97DA OR |
| | | | ;97EA |
| DCB7 | brclr | *L00A2,#0x20,LDCC1 | ;BRANCH IF SHIFT LIGHT OFF, ELSE |
| DCBB | suba | L97AB | ;RPM HYST FOR MIN RPM FOR SHIFT LITE ON |
| DCBE | bcc | LDCC1 | |
| DCC0 | clra | | |
| DCC1 | LDCC1:cmpa | *L00AC | ;COMPARE RPM/25 |
| DCC3 | bhi | LDD03 | ; |
| DCC5 | ldaa | *L00AC | ;LOAD RPM/25 |
| DCC7 | ldab | #0x28 | ;OFFSET |
| DCC9 | cmpa | #0xC8 | ;5000 RPM |
| DCCB | bls | LDCCF | ;BRANCH IF <= 5000 RPM, ELSE |
| DCCD | ldaa | #0xC8 | ;LOAD 5000 RPM |
| DCCF | LDCCF:jsr | L99D3 | ;2D LOOKUP |

```

DCD2      brclr  *L00A2,#0x20,LDCDC      ;BRANCH IF SHIFT LIGHT OFF, ELSE
DCD6      adda   0x0E,x                  ;97BD OR
                                           ;97CD OR
                                           ;97DD OR
                                           ;97ED
                                           ;BRANCH IF NO OVERFLOW, ELSE
DCD8      bcc    LDCDC                   ;LOAD 255
DCDA      ldaa   #0xFF                   ;SAVE TABLE LOOKUP RESULT
DCDC      LDCDC:psha                      ;SAVE TABLE ADDRESS
DCDD      pshx                                     ;
DCDE      ldz    #0x97EF                 ;
DCE1      ldaa   *L005A                  ;FILTERED ENGINE TORQUE
DCE3      ldab   #0x60                   ;OFFSET
DCE5      jsr    L99D3                   ;2D LOOKUP
DCE8      pulx                                     ;RESTORE
DCE9      pulb                                     ;GET BACK PREV TABLE LOOKUP RESULT
DCEA      mul                                     ;A * B
DCEB      lsld                                     ;MUL BY 2
DCEC      bcc    LDCF0                   ;BRANCH IF NO OVERFLOW, ELSE
DCEE      ldaa   #0xFF                   ;LOAD 255
DCF0      LDCF0:cmpa  *L00AA              ;%TPS
DCF2      bcs    LDD03                   ;BRANCH IF < %TPS, ELSE
DCF4      ldaa   L0327                   ;SHIFT LITE TIMER
DCF7      cmpa   0x0F,x                  ;97BE OR
                                           ;97CE OR
                                           ;97DE OR
                                           ;97EE
                                           ;BRANCH IF > CAL, ELSE
DCF9      bcc    LDD0F                   ;INCREMENT TIMER
DCFB      inc    L0327
DCFE      bra    LDD06                   ;
                                           ;
DD00      LDD00:clr  L0326                ;CLEAR TIMER
DD03      LDD03:clr  L0327                ;CLEAR TIMER
DD06      LDD06:ldaa  *L00AC              ;LOAD RPM/25
DD08      cmpa   L97AC                   ;RPM ABOVE WHICH LIGHT IS ALWAYS ON
DD0B      bls    LDD1F                   ;BRANCH IF <= CAL, ELSE
DD0D      bra    LDD1A                   ;
                                           ;
DD0F      LDD0F:ldaa  L0325               ;SHIFT LIGHT TIMER
DD12      cmpa   L97A7                   ;MAX ON TIME
DD15      bhi    LDD22                   ;BRANCH IF > CAL, ELSE
DD17      inc    L0325                   ;INCREMENT TIMER
DD1A      LDD1A:bset  *L00A2,#0x20        ;TURN SHIFT LIGHT ON
DD1D      bra    LDD25                   ;
                                           ;
DD1F      LDD1F:clr  L0325               ;CLEAR TIMER
DD22      LDD22:bclr  *L00A2,#0x20        ;TURN SHIFT LIGHT OFF
DD25      LDD25:jmp   LE28D               ;JUMP OVER F31 STUFF
                                           ;
;-----
; BRANCH HERE FROM SEGMENT TABLE - 100 MSEC ROUTINE
; TRANNY STUFF
; F31 (1370 BYTES) OR SHIFT LIGHT (238 BYTES), DEPENDING ON L8010
;-----
DD28      ldaa   L8010                   ;OPTION WORD - TRANS TYPE - TIS CLEAR
DD2B      bpl    LDD30                   ;BRANCH IF AUTO TRANS, ELSE
DD2D      jmp    LDC37                   ;JUMP
                                           ;
DD30      LDD30:brset  *L0013,#0x10,LDD3C  ;BRANCH IF LOW SYSTEM VOLTS MALF, ELSE
DD34      brset  *L0015,#0x02,LDD3C      ;BRANCH IF VSS INT SIGNAL MALF, ELSE
DD38      brclr  *L0014,#0x20,LDD3F      ;BRANCH IF NO VSS SIGNAL, ELSE
DD3C      LDD3C:jmp   LE1B5               ;
                                           ;
DD3F      LDD3F:ldaa  *L00E9              ;TRANNY GEAR BYTE
DD41      cmpa   #0x02                   ;2ND GEAR?

```

| | | | |
|------|-------------|--------------------|--|
| DD43 | bne | LDD53 | ;BRANCH IF NOT 2 ND GEAR, ELSE |
| DD45 | ldab | L01BF | ; |
| DD48 | incb | | ; |
| DD49 | beq | LDD4E | ;BRANCH IF ZERO, ELSE |
| DD4B | stab | L01BF | ;STORE |
| DD4E | LDD4E:clr | L01C0 | ; |
| DD51 | bra | LDD56 | ; |
| DD53 | LDD53:clr | L01BF | ; |
| DD56 | LDD56:cmpa | #0x03 | ;3 RD GEAR? |
| DD58 | bne | LDD68 | ;BRANCH IF NOT 3 RD GEAR, ELSE |
| DD5A | ldab | L01C0 | ; |
| DD5D | incb | | |
| DD5E | beq | LDD63 | |
| DD60 | stab | L01C0 | |
| DD63 | LDD63:clr | L01BF | |
| DD66 | bra | LDD6B | |
| DD68 | LDD68:clr | L01C0 | |
| DD6B | LDD6B:ldab | *L00E9 | ;TRANNNY GEAR BYTE |
| DD6D | ldaa | L942F | ; |
| DD70 | decb | | ;DECREMENT GEAR BYTE |
| DD71 | beq | LDD7C | ;BRANCH IF Z, ELSE |
| DD73 | ldaa | L9430 | ;0x00 |
| DD76 | decb | | ;DECREMENT GEAR BYTE |
| DD77 | beq | LDD7C | ;BRANCH IF Z, ELSE |
| DD79 | ldaa | L9431 | ;0x00 |
| DD7C | LDD7C:staa | L0301 | ;STORE |
| DD7F | ldaa | *L00E9 | ;TRANNNY GEAR BYTE |
| DD81 | cmpa | #0x04 | ;4 TH GEAR? |
| DD83 | bne | LDD9D | ;BRANCH IF NOT 4 TH GEAR, ELSE |
| DD85 | brset | *L0090,#0x20,LDDFB | ; |
| DD89 | brset | *L0013,#0x01,LDDFB | ;BRANCH IF TPS VOLTS HIGH, ELSE |
| DD8D | brset | *L0014,#0x80,LDDFB | ;BRANCH IF TPS VOLTS LOW, ELSE |
| DD91 | brset | *L0015,#0x04,LDDFB | ;BRANCH IF SSB ERROR, ELSE |
| DD95 | brset | *L0015,#0x01,LDDFB | ;BRANCH IF TCC BRAKE SW ERROR, ELSE |
| DD99 | brset | *L0016,#0x80,LDDFB | ;BRANCH IF TCC ERROR, ELSE |
| DD9D | LDD9D:ldd | *L005A | ;FILTERED ENGINE TORQUE |
| DD9F | lsrd | | ;DIV BY 2 |
| DDA0 | ldx | L8698 | ;0xBE00 |
| DDA3 | fdiv | | ;D/X |
| DDA4 | stx | L01D0 | ;SAVE RATIO |
| DDA7 | ldaa | L01C9 | ;TIMER |
| DDAA | inca | | ;INCREMENT |
| DDAB | beq | Lddb0 | ;BRANCH IF Z, ELSE |
| DDAD | staa | L01C9 | ;STORE TIMER |
| DDB0 | Lddb0:ldaa | L01CA | ;LOAD F31/TCCPWM TIMER |
| DDB3 | inca | | ;INCREMENT |
| DDB4 | beq | Lddb9 | ;BRANCH IF Z, ELSE |
| DDB6 | staa | L01CA | ;SAVE TIMER |
| DDB9 | Lddb9:ldaa | L01E4 | ;LOAD TIMER |
| DDBC | inca | | ;INCREMENT |
| DDBD | beq | LDDC2 | ;BRANCH IF Z, ELSE |
| DDBF | staa | L01E4 | ;SAVE TIMER |
| DDC2 | LDDC2:brclr | *L00A1,#0x02,LDE13 | ;BRANCH IF PRNDL -> NOT DRIVE 1, ELSE |
| DDC6 | bclr | *L00A2,#0x80 | ;CLEAR BIT 7 |
| DDC9 | ldaa | *L00E9 | ;TRANNNY GEAR BYTE |
| DDCB | cmpa | #0x02 | ;2 ND GEAR? |
| DDCD | bhi | LDE0D | ;BRANCH IF 3 RD OR 4 TH GEAR, ELSE |
| DDCF | ldab | *L00F1 | ;TRANNNY MPH |
| DDD1 | ldaa | L9434 | ;0x19 |
| DDD4 | brclr | *L00A2,#0x02,LDDDB | |
| DDD8 | ldaa | L9435 | ;0x26 |
| DDDB | LDDDB:cmpa | *L00F5 | ;%TPS |

| | | | |
|------|-------------|--------------------|---|
| DDDD | bc | LDEE | |
| DDDF | bset | *L00A2,#0x02 | ;SET BIT 1 |
| DDE2 | ldaa | *L00E9 | ;TRANNNY GEAR BYTE |
| DDE4 | deca | | |
| DDE5 | beq | LDE10 | |
| DDE7 | cmpb | L9436 | ;0x23 |
| DDEA | bls | LDE0D | ; |
| DDEC | bra | LDE10 | |
| | | | |
| DDEE | LDEE:bclr | *L00A2,#0x02 | ;CLEAR BIT 1 |
| DDF1 | ldaa | *L00E9 | ;TRANNNY GEAR BYTE |
| DDF3 | deca | | ;DECREMENT GEAR BYTE |
| DDF4 | beq | LDDFD | ;BRANCH IF Z, ELSE |
| DDF6 | cmpb | L9437 | ;0x14 |
| DDF9 | bhi | LDE10 | ;BRANCH IF > 0x14 |
| DDFB | LDDFB:bra | LDE0D | |
| | | | |
| DDFD | LDDFD:cmpb | L9438 | ;0x23 |
| DE00 | bls | LDE10 | ; |
| DE02 | ldd | *L00EB | ;TRANNNY RPM |
| DE04 | cpd | L9439 | ;5952 |
| DE08 | bls | LDE10 | ;BRANCH IF <= 5952 RPM, ELSE |
| DE0A | LDE0A: jmp | LE100 | |
| | | | |
| DE0D | LDE0D: jmp | LE170 | |
| | | | |
| DE10 | LDE10: jmp | LE1B5 | |
| | | | |
| DE13 | LDE13:brset | *L0095,#0x40,LDE42 | ;BRANCH IF BIT 6, ELSE |
| DE17 | ldaa | *L00AC | ;LOAD RPM/25 |
| DE19 | cmpa | L01E5 | ;TABLE LOOKUP RESULT |
| DE1C | bc | LDE2E | ;BRANCH IF < |
| DE1E | ldaa | *L00E9 | ;TRANNNY GEAR BYTE |
| DE20 | cmpa | #0x04 | ;4 TH GEAR? |
| DE22 | beq | LDE10 | ;BRANCH IF 4 TH GEAR, ELSE |
| DE24 | ldaa | L01E4 | ;LOAD |
| DE27 | cmpa | L9851 | ;50 |
| DE2A | bcc | LDE0A | ;BRANCH IF > 50, ELSE |
| DE2C | bra | LDE10 | |
| | | | |
| DE2E | LDE2E:ldaa | *L00F1 | ;TRANNNY MPH |
| DE30 | cmpa | L01E6 | ;TABLE LOOKUP RESULT |
| DE33 | bhi | LDE42 | ;BRANCH IF > LOOKUP, ELSE |
| DE35 | ldaa | L01E4 | ; |
| DE38 | cmpa | L9851 | |
| DE3B | bc | LDE10 | |
| DE3D | clr | L01E4 | |
| DE40 | bra | LDE0D | |
| | | | |
| DE42 | LDE42:brset | *L008C,#0x01,LDE49 | ;BRANCH IF NOT 2 ND GEAR START, ELSE |
| DE46 | jmp | LDF08 | |
| | | | |
| DE49 | LDE49:ldaa | L9451 | |
| DE4C | brset | *L00A2,#0x80,LDE53 | |
| DE50 | ldaa | L9450 | |
| DE53 | LDE53:cmpa | *L00AA | ; %TPS |
| DE55 | bc | LDE5A | |
| DE57 | jmp | LDFBA | |
| | | | |
| DE5A | LDE5A:bset | *L00A2,#0x80 | |
| DE5D | ldaa | *L00F1 | ;TRANNNY MPH |
| DE5F | ldab | *L00E9 | ;TRANNNY GEAR BYTE |
| DE61 | dec | | |
| DE62 | beq | LDE72 | |

| | | | | |
|------|--------------|--------------------|--|---|
| DE64 | ldx | #0x943A | | ;TABLE ADDRESS |
| DE67 | abx | | | ;ADD TO ADDRESS |
| DE68 | adda | 0x00,x | | ;ADD VALUE TO A |
| DE6A | ldx | #0x0068 | | ; |
| DE6D | abx | | | |
| DE6E | cmpa | 0x00,x | | |
| DE70 | bls | LDE0D | | |
| DE72 | LDE72: cmpb | #0x03 | | |
| DE74 | beq | LDE10 | | |
| DE76 | ldx | #0x0069 | | ;INDEX |
| DE79 | abx | | | ;ADD COMMANDED GEAR |
| DE7A | pshx | | | ;SAVE |
| DE7B | incb | | | ;INCREMENT COMMANDED GEAR |
| DE7C | ldx | #0x9469 | | ; |
| DE7F | decb | | | ;DECREMENT COMMANDED GEAR |
| DE80 | beq | LDE9D | | ;BRANCH IF Z (1 ST GEAR), ELSE |
| DE82 | ldx | #0x947A | | ; |
| DE85 | brset | *L001A,#0x04,LDE90 | | ;BRANCH IF MALF (NOT ENABLED), ELSE |
| DE89 | brclr | *L00D9,#0x40,LDE90 | | ;BRANCH IF SMC NOT ENGAGED, ELSE |
| DE8D | ldx | #0x9557 | | |
| DE90 | LDE90: brclr | *L0090,#0x20,LDE97 | | |
| DE94 | ldx | #0x94CF | | |
| DE97 | LDE97: decb | | | |
| DE98 | beq | LDE9D | | |
| DE9A | ldx | #0x948B | | |
| DE9D | LDE9D: pshx | | | ;SAVE INDEX |
| DE9E | bset | *L0097,#0x10 | | ;SET LOOKUP IN WORK |
| DEA1 | jsr | LF81F | | ;DO LOOKUP |
| DEA4 | bcclr | *L0097,#0x10 | | ;CLEAR LOOKUP IN WORK |
| DEA7 | ldab | *L00E9 | | ;TRANNNY GEAR BYTE |
| DEA9 | ldx | #0x977D | | ; |
| DEAC | abx | | | ;ADD COMMANDED GEAR |
| DEAD | ldab | 0x00,x | | ;GET TABLE VALUE |
| DEAF | pulx | | | ;RESTORE INDEX |
| DEB0 | cmpb | 0x10,x | | |
| DEB2 | bcc | LDEB6 | | |
| DEB4 | ldab | 0x10,x | | |
| DEB6 | LDEB6: mul | | | |
| DEB7 | lsld | | | |
| DEB8 | bcc | LDEBC | | |
| DEBA | ldaa | #0xFF | | |
| DEBC | LDEBC: ldab | *L00F1 | | ;TRANNNY MPH |
| DEBE | cba | | | |
| DEBF | bcc | LDF04 | | |
| DEC1 | ldab | *L00E9 | | ;TRANNNY GEAR BYTE |
| DEC3 | ldy | #0x9442 | | ;ADDRESS |
| DEC7 | ldx | #0x943C | | ;ADDRESS |
| DECA | lslb | | | ;RESCALE |
| DECB | aby | | | ;ADD TO ADDRESS |
| DECD | pshy | | | ;SAVE |
| DECF | abx | | | ;ADD TO ADDRESS |
| DED0 | ldy | L0314 | | ; |
| DED4 | cpy | 0x0C,x | | |
| DED7 | blt | LDEEB | | |
| DED9 | pshx | | | ;SAVE ADDRESS |
| DEDA | ldaa | *L00FA | | |
| DEDC | jsr | L98B9 | | ;8 x 16 MULTIPLY |
| DEDF | lsld | | | |
| DEE0 | bcc | LDEE5 | | |
| DEE2 | ldd | #0xFFFF | | |
| DEE5 | LDEE5: pulx | | | ;RESTORE ADDRESS |
| DEE6 | cpd | 0x00,x | | ;COMPARE |
| DEE9 | bls | LDEED | | ;BRANCH IF <= MAX VALUE, ELSE |
| DEEB | LDEEB: ldd | 0x00,x | | ;LOAD MAX VALUE |

| | | | |
|------|------------|--------------------|--|
| DEED | LDEED:puly | | |
| DEEF | cpd | 0x00,y | |
| DEF2 | bcc | LDEF7 | |
| DEF4 | ldd | 0x00,y | |
| DEF7 | LDEF7:cpd | *L00EB | ;RPM |
| DEFA | bcc | LDF04 | |
| DEFC | ldaa | *L00F1 | ;TRANNNY MPH |
| DEFE | pulx | | |
| DEFF | staa | 0x00,x | |
| DF01 | jmp | LE100 | |
| DF04 | LDF04:pulx | | |
| DF05 | LDF05:jmp | LE1B5 | |
| DF08 | LDF08:ldaa | L9468 | |
| DF0B | brset | *L00A2,#0x80,LDF12 | |
| DF0F | ldaa | L9467 | |
| DF12 | LDF12:cmpa | *L00AA | ; %TPS |
| DF14 | bcs | LDF19 | |
| DF16 | jmp | LDFBA | |
| DF19 | LDF19:bset | *L00A2,#0x80 | |
| DF1C | ldaa | *L00F1 | ;TRANNNY MPH |
| DF1E | ldab | *L00E9 | ;TRANNNY GEAR BYTE |
| DF20 | decb | | |
| DF21 | beq | LDF34 | |
| DF23 | ldx | #0x9451 | ; INDEX |
| DF26 | abx | | |
| DF27 | adda | 0x00,x | |
| DF29 | ldx | #0x006B | |
| DF2C | abx | | |
| DF2D | cmpa | 0x00,x | |
| DF2F | bhi | LDF34 | |
| DF31 | jmp | LE170 | |
| DF34 | LDF34:cmpb | #0x03 | ; 3 RD GEAR? |
| DF36 | bcc | LDF05 | ; BRANCH IF > 3 RD GEAR, ELSE |
| DF38 | ldx | #0x006C | ; |
| DF3B | abx | | |
| DF3C | pshx | | |
| DF3D | incb | | |
| DF3E | ldx | #0x94F1 | ; TABLE ADDRESS |
| DF41 | decb | | |
| DF42 | beq | LDF4D | |
| DF44 | ldx | #0x9502 | ; |
| DF47 | decb | | |
| DF48 | beq | LDF4D | |
| DF4A | ldx | #0x9513 | ; ADDRESS |
| DF4D | LDF4D:pshx | | ; SAVE |
| DF4E | bset | *L0097,#0x10 | ; SET LOOKUP IN WORK |
| DF51 | jsr | LF81F | ; DO LOOKUP |
| DF54 | bclr | *L0097,#0x10 | ; CLEAR LOOKUP IN WORK |
| DF57 | staa | *L00FA | ; SAVE LOOKUP RESULT |
| DF59 | ldab | *L00E9 | ; TRANNNY GEAR BYTE |
| DF5B | ldx | #0x9780 | ; INDEX |
| DF5E | abx | | ; ADD COMMANDED GEAR TO INDEX |
| DF5F | ldab | 0x00,x | ; GET A VALUE |
| DF61 | pulx | | ; ADDRESS = 9513 |
| DF62 | cmpb | 0x10,x | ; COMPARE |
| DF64 | bcc | LDF68 | |
| DF66 | ldab | 0x10,x | |
| DF68 | LDF68:mul | | |
| DF69 | lslld | | |
| DF6A | bcc | LDF6E | |

| | | | | |
|------|--------|-------|--------------------|-------------------------------------|
| DF6C | | ldaa | #0xFF | |
| DF6E | LDF6E: | ldab | *L00F1 | ;TRANNY MPH |
| DF70 | | cba | | |
| DF71 | | bcc | LDFB6 | |
| DF73 | | ldab | *L00E9 | ;TRANNY GEAR BYTE |
| DF75 | | ldy | #0x9459 | |
| DF79 | | ldx | #0x9453 | |
| DF7C | | ls1b | | |
| DF7D | | aby | | |
| DF7F | | pshy | | |
| DF81 | | abx | | |
| DF82 | | ldy | L0314 | |
| DF86 | | cpy | 0x0C,x | |
| DF89 | | blt | LDF9D | |
| DF8B | | pshx | | |
| DF8C | | ldaa | *L00FA | |
| DF8E | | jsr | L98B9 | ;8 x 16 MULTIPLY |
| DF91 | | lsld | | |
| DF92 | | bcc | LDF97 | |
| DF94 | | ldd | #0xFFFF | |
| DF97 | LDF97: | pulx | | |
| DF98 | | cpd | 0x00,x | |
| DF9B | | b1s | LDF9F | |
| DF9D | LDF9D: | ldd | 0x00,x | |
| DF9F | LDF9F: | puly | | |
| DFA1 | | cpd | 0x00,y | |
| DFA4 | | bcc | LDFA9 | |
| DFA6 | | ldd | 0x00,y | |
| DFA9 | LDFA9: | cpd | *L00EB | ;TRANNY RPM |
| DFAC | | bcc | LDFB6 | |
| DFAE | | ldaa | *L00F1 | ;TRANNY MPH |
| DFB0 | | pulx | | |
| DFB1 | | staa | 0x00,x | |
| DFB3 | | jmp | LE100 | |
| DFB6 | LDFB6: | pulx | | |
| DFB7 | | jmp | LE1B5 | |
| DFBA | LDFBA: | bclr | *L00A2,#0x80 | |
| DFBD | | brset | *L001A,#0x04,LDFC5 | ;BRANCH IF MALF (NOT ENABLED), ELSE |
| DFC1 | | brset | *L00D9,#0x40,LDFC8 | ;BRANCH IF SMC ENGAGED, ELSE |
| DFC5 | LDFC5: | jmp | LE057 | |
| DFC8 | LDFC8: | ldaa | *L00E9 | ;TRANNY GEAR BYTE |
| DFCA | | deca | | |
| DFCB | | bne | LDFD0 | |
| DFCD | | jmp | LE0A9 | |
| DFD0 | LDFD0: | ldab | L942F | |
| DFD3 | | stab | L0301 | |
| DFD6 | | clrb | | |
| DFD7 | | ldx | #0x949C | |
| DFDA | | deca | | |
| DFDB | | beq | LDFFE | |
| DFDD | | ldab | L9430 | |
| DFE0 | | stab | L0301 | |
| DFE3 | | clrb | | |
| DFE4 | | ldx | #0x9579 | |
| DFE7 | | brclr | *L0090,#0x20,LDFFE | |
| DFEB | | ldx | #0x94E0 | |
| DFEE | LDLEE: | deca | | |
| DFEF | | beq | LDFFE | |
| DFF1 | | ldab | L9431 | |
| DFF4 | | stab | L0301 | |

```

DFF7      clr      b
DFF8      ldx      #0x958A
DFFB      dec      a
DFFC      bne      LE054
DFFE      LDFFE:jsr    LF81F
E001      ldab     *L00FA
E003      mul
E004      lsld
E005      bcc      LE009
E007      ldaa     #0xFF
E009      LE009:cmpa  0x00,x
E00B      bcc      LE00F
E00D      ldaa     0x00,x
E00F      LE00F:cmpa  *L00F1
E011      bcs      LE016
E013      jmp      LE170

E016      LE016:ldab  *L00E9
E018      cmpb     #0x04
E01A      beq      LE054
E01C      ldx      #0x9557
E01F      brclr    *L0090,#0x20,LE026
E023      ldx      #0x94CF
E026      LE026:ldaa  L9430
E029      subb     #0x02
E02B      beq      LE033
E02D      ldx      #0x9568
E030      ldaa     L9431
E033      LE033:staa  L0301
E036      jsr      LF81F
E039      staa     L032C
E03C      ldab     *L00FA
E03E      mul
E03F      lsld
E040      bcc      LE044
E042      ldaa     #0xFF
E044      LE044:cmpa  0x00,x
E046      bcc      LE04A
E048      ldaa     0x00,x
E04A      LE04A:staa  L032D
E04D      cmpa     *L00F1
E04F      bcc      LE054
E051      jmp      LE100

E054      LE054:jmp   LE1B5

E057      LE057:ldab  *L00E9
E059      decb
E05A      beq      LE0A9
E05C      ldx      #0x9513
E05F      brclr    *L008C,#0x01,LE066
E063      ldx      #0x948B
E066      LE066:ldaa  #0x11
E068      mul
E069      abx
E06A      cpx      #0x94AD
E06D      beq      LE074
E06F      cpx      #0x9535
E072      bne      LE07B
E074      LE074:brclr *L0090,#0x20,LE07B
E078      ldx      #0x94E0
E07B      LE07B:ldaa  L942F
E07E      ldab     *L00E9
E080      decb

;DO LOOKUP
;ANOTHER LOOKUP RESULT
;A * B
;MUL BY 2
;BRANCH IF NO OVERFLOW, ELSE
;
;
;TRANNNY MPH

;TRANNNY GEAR BYTE

;BRANCH IF Z, ELSE
;
;
;DO LOOKUP
;SAVE LOOKUP RESULT
;

;TRANNNY MPH

;TRANNNY GEAR BYTE
;DECREMENT
;BRANCH IF ZERO, ELSE
;INDEX
;BRANCH IF 2ND GEAR START, ELSE
;INDEX
;OFFSET
;OFFSET x GEAR BYTE
;ADD TO INDEX
;
;BRANCH IF INDEX = 0x04AD, ELSE
;
;BRANCH IF INDEX NOT = 0x9535, ELSE

;INDEX
;
;TRANNNY GEAR BYTE
;DECREMENT

```

```

E081          decb                ;DECREMENT
E082          beq      LE08D       ;BRANCH IF ZERO, ELSE
E084          ldaa      L9430      ;LOAD
E087          decb                ;DECREMENT
E088          beq      LE08D       ;BRANCH IF ZERO, ELSE
E08A          ldaa      L9431      ;LOAD
E08D  LE08D: staa      L0301      ;STORE
E090          jsr      LF81F       ;DO LOOKUP
E093          ldab      *L00FA      ;
E095          incb                ;INCREMENT
E096          mul                ;LOOKUP RESULT * L00FA+1
E097          lsld                ;MUL BY 2
E098          bcc      LE09C       ;BRANCH IF NO OVERFLOW, ELSE
E09A          ldaa      #0x80      ;LOAD 128
E09C  LE09C: cmpa      0x00,x      ;
E09E          bcc      LE0A2
E0A0          ldaa      0x00,x
E0A2  LE0A2: cmpa      *L00F1      ;TRANNNY MPH
E0A4          bls      LE0A9
E0A6          jmp      LE170

E0A9  LE0A9: ldab      *L00E9      ;TRANNNY GEAR BYTE
E0AB          cmpb      #0x04      ;4TH GEAR?
E0AD          beq      LE054      ;BRANCH IF 4TH GEAR, ELSE
E0AF          ldx      #0x94E0      ;
E0B2          brclr    *L008C,#0x01,LE0B9 ;BRANCH IF 2ND GEAR START, ELSE
E0B6          ldx      #0x9458
E0B9  LE0B9: ldaa      #0x11      ;OFFSET
E0BB          mul                ;
E0BC          abx                ;ADD TO ADDRESS
E0BD          cpx      #0x947A      ;
E0C0          beq      LE0C7      ;
E0C2          cpx      #0x9502
E0C5          bne      LE0CE
E0C7  LE0C7: brclr    *L0090,#0x20,LE0CE
E0CB          ldx      #0x94CF
E0CE  LE0CE: ldaa      L942F
E0D1          ldab      *L00E9      ;TRANNNY GEAR BYTE
E0D3          decb
E0D4          beq      LE0DF
E0D6          ldaa      L9430
E0D9          decb
E0DA          beq      LE0DF
E0DC          ldaa      L9431
E0DF  LE0DF: staa      L0301
E0E2          jsr      LF81F       ;DO LOOKUP
E0E5          staa      L032C      ;SAVE LOOKUP RESULT
E0E8          ldab      *L00FA
E0EA          mul
E0EB          lsld
E0EC          bcc      LE0F0
E0EE          ldaa      #0x80
E0F0  LE0F0: cmpa      0x00,x
E0F2          bcc      LE0F6
E0F4          ldaa      0x00,x
E0F6  LE0F6: staa      L032D
E0F9          cmpa      *L00F1      ;TRANNNY MPH
E0FB          bls      LE100
E0FD          jmp      LE1B5

E100  LE100: clr      L01E4
E103          ldab      *L00E9      ;TRANNNY GEAR BYTE
E105          brset    *L00A2,#0x80,LE123
E109          cmpb      #0x02      ;2ND GEAR?

```

```

E10B      bne      LE117      ;BRANCH IF NOT 2ND GEAR, ELSE
E10D      ldaa     L01BF      ;LOAD
E110      cmpa     L941E
E113      bls      LE16E
E115      bra      LE123

E117      LE117:cmpb     #0x03      ;3RD GEAR?
E119      bne      LE123      ;BRANCH IF NOT 3RD GEAR, ELSE
E11B      ldaa     L01C0      ;LOAD
E11E      cmpa     L941F      ;
E121      bls      LE16E      ;BRANCH IF <= CAL, ELSE
E123      LE123:cmpb     #0x03      ;3RD GEAR?
E125      bcs      LE155      ;BRANCH IF < 3RD GEAR, ELSE
E127      brset    *L0090,#0x20,LE16E      ;
E12B      brset    *L0016,#0x80,LE16E      ;BRANCH IF TCC ERROR
E12F      brset    *L0013,#0x01,LE16E      ;BRANCH IF TPS VOLTS HIGH, ELSE
E133      brset    *L0014,#0x80,LE1B5      ;BRANCH IF TPS VOLTS LOW, ELSE
E137      brset    *L0015,#0x04,LE1B5      ;BRANCH IF SSB ERROR, ELSE
E13B      brset    *L0015,#0x01,LE1B5      ;BRANCH IF TCC BRAKE SW ERROR, ELSE
E13F      brset    *L00A1,#0x08,LE155      ;BRANCH IF PRNDL -> DRIVE 3, ELSE
E143      bset     *L00A2,#0x40
E146      ldaa     L01CD
E149      adda     L01C6
E14C      bcc      LE150
E14E      ldaa     #0xFF
E150      LE150:cmpa     L9432
E153      bcs      LE1B5
E155      LE155:cmpb     #0x02
E157      bne      LE16B
E159      bset     *L00A2,#0x10
E15C      ldaa     L01CD
E15F      adda     L01C6
E162      bcc      LE166
E164      ldaa     #0xFF
E166      LE166:cmpa     L9432
E169      bcs      LE1B5
E16B      LE16B:incb
E16C      bra      LE1AD

E16E      LE16E:bra      LE1B5

E170      LE170:ldab     #0x04      ;
E172      brclr    *L0090,#0x20,LE17A
E176      brclr    *L008E,#0x08,LE194      ;BRANCH IF 4TH GEAR, ELSE
E17A      LE17A:brclr    *L008E,#0x08,LE184      ;BRANCH IF 4TH GEAR, ELSE
E17E      decb
E17F      brclr    *L008E,#0x04,LE184      ;BRANCH IF 3RD GEAR, ELSE
E183      decb
E184      LE184:cmpb     *L00E9      ;TRANNNY GEAR BYTE
E186      bcs      LE197
E188      brclr    *L008E,#0x08,LE194      ;BRANCH IF 4TH GEAR, ELSE
E18C      ldab     L01CA      ;F31/TCCPWM TIMER
E18F      cmpb     L9433      ;
E192      bcs      LE1B5      ;BRANCH IF < CAL, ELSE
E194      LE194:clr      L01CA      ;CLEAR TIMER
E197      LE197:brclr    *L00A3,#0x08,LE1B5      ;BRANCH IF NOT OFF MODE, ELSE
E19B      ldab     *L00E9      ;TRANNNY GEAR BYTE
E19D      cmpb     #0x02      ;2ND GEAR?
E19F      bne      LE1A9      ;BRANCH IF NOT 2ND GEAR, ELSE
E1A1      brset    *L001A,#0x04,LE1A9      ;BRANCH IF MALF (NOT ENABLED), ELSE
E1A5      brset    *L00D9,#0x40,LE1B5      ;BRANCH IF SMC ENGAGED, ELSE
E1A9      LE1A9:decb
E1AA      bne      LE1AD      ;DECREMENT GEAR BYTE
E1AC      incb      ;BRANCH IF NOT = ZERO, ELSE
                        ;INCREMENT GEAR BYTE

```

| | | | |
|------|-------------|--------------------|---|
| E1AD | LE1AD:cmpb | #0x04 | ;4 TH GEAR? |
| E1AF | bls | LE1B3 | ;BRANCH IF <= 4, ELSE |
| E1B1 | ldab | #0x04 | ;LOAD 4 TH GEAR |
| E1B3 | LE1B3:stab | *L00E9 | ;TRANNNY GEAR BYTE |
| E1B5 | LE1B5:brset | *L0013,#0x10,LE1FB | ;BRANCH IF LOW SYSTEM VOLTS MALF, ELSE |
| E1B9 | brset | *L0015,#0x02,LE1C1 | ;BRANCH IF VSS INT SIGNAL MALF, ELSE |
| E1BD | brclr | *L0014,#0x20,LE200 | ;BRANCH IF NO VSS SIGNAL, ELSE |
| E1C1 | LE1C1:brclr | *L0091,#0x20,LE1FB | ; |
| E1C5 | ldaa | L01BD | ;TRANNNY GEAR BYTE |
| E1C8 | cmpa | #0x03 | ;3 RD GEAR ? |
| E1CA | bne | LE20F | ;BRANCH IF NOT 3 RD GEAR, ELSE |
| E1CC | ldaa | *L00AA | ;%TPS |
| E1CE | cmpa | L8CE2 | ;1.2% TPS |
| E1D1 | bcc | LE1EB | ;BRANCH IF >= 1.2% TPS, ELSE |
| E1D3 | ldaa | *L00AD | ;LOAD RPM/12.5 |
| E1D5 | cmpa | L9424 | ;950 RPM |
| E1D8 | bcc | LE23B | ;BRANCH IF >= 950 RPM, ELSE |
| E1DA | ldab | *L00DE | ;DESIRED IDLE RPM |
| E1DC | addb | L9425 | ;75 RPM |
| E1DF | bcc | LE1E3 | ;BRANCH IF NO OVERFLOW, ELSE |
| E1E1 | ldab | #0xFF | ;LOAD MAX |
| E1E3 | LE1E3:cba | | ;COMPARE RPM/12.5 AND DESIRED IDLE |
| E1E4 | bcc | LE23B | ;BRANCH IF RPM/12.5 >= DESIRED, ELSE |
| E1E6 | bset | *L0088,#0x04 | ;SET RPM < DESIRED |
| E1E9 | bra | LE1F7 | |
| E1EB | LE1EB:cmpa | L9420 | ;70% TPS |
| E1EE | bls | LE23B | ;BRANCH IF <= 70%, ELSE |
| E1F0 | ldaa | *L00AC | ;LOAD RPM/25 |
| E1F2 | cmpa | L9421 | ;3000 RPM |
| E1F5 | bcc | LE23B | ;BRANCH IF >= 3000 RPM, ELSE |
| E1F7 | LE1F7:ldaa | #0x02 | ;LOAD 2 |
| E1F9 | bra | LE233 | ;SHIFT INTO 2 ND GEAR |
| E1FB | LE1FB:bset | *L0091,#0x20 | ; |
| E1FE | bra | LE231 | |
| E200 | LE200:brclr | *L0091,#0x20,LE23B | ; |
| E204 | ldaa | *L00E9 | ;TRANNNY GEAR BYTE |
| E206 | cmpa | #0x03 | ;3 RD GEAR? |
| E208 | bne | LE236 | ;BRANCH IF NOT 3 RD GEAR, ELSE |
| E20A | bclr | *L0091,#0x20 | ;CLEAR BIT 5 |
| E20D | bra | LE23B | |
| E20F | LE20F:ldaa | *L00AA | ;%TPS |
| E211 | brclr | *L0088,#0x04,LE225 | ;BRANCH IF RPM < DESIRED, ELSE |
| E215 | ldx | #0x9426 | ;RPM VS %TPS |
| E218 | lsra | | ;RESCALE TPS% |
| E219 | jsr | L99D7 | ;2D LOOKUP |
| E21C | cmpa | *L00AC | ;LOAD RPM/25 |
| E21E | bhi | LE23B | ;BRANCH IF RPM > CAL, ELSE |
| E220 | bclr | *L0088,#0x04 | ;CLEAR RPM < DESIRED |
| E223 | bra | LE231 | ; |
| E225 | LE225:cmpa | L9422 | ;50 %TPS |
| E228 | bcs | LE231 | ;BRANCH IF < 50 %TPS, ELSE |
| E22A | ldaa | *L00AC | ;LOAD RPM/25 |
| E22C | cmpa | L9423 | ;5000 RPM |
| E22F | bls | LE23B | ;BRANCH IF <= 5000 RPM, ELSE |
| E231 | LE231:ldaa | #0x03 | ;FORCE 3 RD GEAR |
| E233 | LE233:staa | L01BD | ;2 ND TRANNNY GEAR BYTE |
| E236 | LE236:ldaa | L01BD | ;LOAD 2 ND TRANNNY GEAR BYTE |
| E239 | staa | *L00E9 | ;TRANNNY GEAR BYTE |
| E23B | LE23B:ldaa | #0x20 | ;BIT 5 - SSA |

```

E23D      jsr      LD90E      ;TEST MODE 4 CONTROL WORDS
E240      bmi      LE249      ;BRANCH IF BIT N, ELSE
E242      beq      LE251      ;BRANCH IF Z, ELSE
E244      ldz      #0xD000    ;0 DUTY CYCLE
E247      bra      LE24C      ;GO STORE SSA OUTPUT

E249      LE249:ldx      #0xDFFF      ;MAX DUTY CYCLE
E24C      LE24C:stx      L3DD8      ;STORE SHIFT SOLENOID A OUTPUT
E24F      bra      LE264      ;

E251      LE251:ldx      #0xFBCC      ;TRANNNY SOLENOID TRUTH TABLE
E254      ldab      *L00E9      ;TRANNNY GEAR BYTE
E256      abx              ;ADD TO ADDRESS
E257      ldd      #0xDFFF      ;MAX DUTY CYCLE
E25A      brset     0x00,x,#0x01,LE261 ;BRANCH IF BIT 0, ELSE
E25E      ldd      #0x0000      ;0 DUTY CYCLE
E261      LE261:std      L3DD8      ;STORE SHIFT SOLENOID A OUTPUT
E264      LE264:ldaa     #0x10      ;BIT 4 - SSB
E266      jsr      LD90E      ;TEST MODE 4 CONTROL WORDS
E269      bmi      LE272      ;BRANCH IF N, ELSE
E26B      beq      LE27A      ;BRANCH IF Z, ELSE
E26D      ldz      #0xD000    ;0 DUTY CYCLE
E270      bra      LE275      ;GO STORE SSB OUTPUT

E272      LE272:ldx      #0xDFFF      ;MAX DUTY CYCLE
E275      LE275:stx      L3DDA      ;STORE SHIFT SOLENOID B OUTPUT
E278      bra      LE28D

E27A      LE27A:ldx      #0xFBCC      ;TRANNNY GEAR SOLENOID WORD
E27D      ldab      *L00E9      ;TRANNNY GEAR BYTE
E27F      abx              ;ADD TO ADDRESS
E280      ldd      #0xDFFF      ;MAX DUTY CYCLE
E283      brset     0x00,x,#0x02,LE28A ;BRANCH IF BIT 1, ELSE
E287      ldd      #0x0000      ;0 DUTY CYCLE
E28A      LE28A:std      L3DDA      ;STORE SHIFT SOLENOID B OUTPUT

E28D      LE28D:ldaa     L0333      ;BYTE FOR TESTING STACK OVERFLOW
E290      beq      LE295      ;BRANCH IF Z, ELSE
E292      bset      *L0095,#0x04    ;SET STACK OVERFLOWED
E295      LE295:sei              ;HOLD INTERRUPTS
E296      ldd      L3FBC          ;
E299      oraa      #0x1F          ;
E29B      orab      #0x90          ;
E29D      psha              ;DELAY
E29E      pula              ;
E29F      std      L3FBC          ;STORE
E2A2      cli              ;CLEAR AND ALLOW INTERRUPTS
E2A3      brclr     *L0084,#0x04,LE2AD ;BRANCH IF NOT CLEAR MALF FLAGS, ELSE
E2A7      jsr      LF84C          ;GO CLEAR MALF FLAGS
E2AA      bclr      *L0084,#0x04    ;CLEAR "CLEAR MALF FLAGS"
E2AD      LE2AD:brclr     *L0084,#0x40,LE2B7 ;BRANCH IF NOT "CLEAR NVRAM"
E2B1      jsr      LF775          ;CLEAR NVRAM AND GET INITIAL VALUES
E2B4      bclr      *L0084,#0x40    ;CLEAR "CLEAR NVRAM"
E2B7      LE2B7:bclr      *L0088,#0x01 ;CLEAR "ALL PWM OUTPUTS COMMANDED ON"
E2BA      brclr     *L0084,#0x80,LE2E2 ;BRANCH IF NOT MODE 4, ELSE
E2BE      ldab      L0230          ;MODE 4 CONTROL BYTE
E2C1      bitb      #0x04          ;BIT 2 ?
E2C3      beq      LE2D4          ;BRANCH IF NOT BIT 2, ELSE
E2C5      brset     *L0095,#0x10,LE2D4 ;BRANCH IF BATT VOLTS < 4.0, ELSE
E2C9      ldaa      *L00AC          ;LOAD RPM/25
E2CB      bne      LE2D4          ;BRANCH IF NOT Z, ELSE
E2CD      brclr     *L008E,#0x01,LE2D4 ;BRANCH IF NOT P/N MODE, ELSE
E2D1      bset      *L0088,#0x01    ;SET "ALL PWM OUTPUTS COMMANDED ON"
E2D4      LE2D4:bitb      #0x10      ;BIT 4 ?

```

```

E2D6          beq      LE2DB          ;BRANCH IF NOT BIT 4, ELSE
E2D8          jsr      LF7E1          ;GO RESET BLMS
E2DB  LE2DB: bitb      #0x40          ;BIT 6 ?
E2DD          beq      LE2E2          ;BRANCH IF NOT BIT 6, ELSE
E2DF          jsr      LF84C          ;GO CLEAR MALF FLAGS
E2E2  LE2E2: rts

;-----
; COOLANT TEMP LOOKUP AND MALF CHECK
;-----

E2E3  LE2E3: ldaa      #0x10          ;SEL CH #1 - CTS
E2E5          jsr      L9A18          ;A/D READ
E2E8          staa     L0255          ;STORE A/D COUNTS
E2EB          ldx      #0xFB6B        ;384 OHM INV COOLANT TABLE
E2EE          ldab     *L00E1          ;
E2F0          bne      LE2F6          ;BRANCH IF NOT = ZERO, ELSE
E2F2          brset    *L0089,#0x01,LE2FF ;BRANCH IF 4K OHM P/U IN USE, ELSE
E2F6  LE2F6: ldx      #0xFB5A        ;4K INV COOLANT TABLE
E2F9          adda     #0x0A          ;ADD 10
E2FB          bcc      LE2FF          ;BRANCH IF NO OVERFLOW, ELSE
E2FD          ldaa     #0xFF          ;LOAD 255
E2FF  LE2FF: jsr      L99D7          ;2D LOOKUP
E302          staa     L010A          ;STORE COOLANT TEMP
E305          ldab     *L00E1          ;LOAD
E307          cmpb     #0x01          ;1 ?
E309          bhi      LE318          ;BRANCH IF > 1, ELSE
E30B          cmpa     #0x78          ;120 (50 DEG C)
E30D          bls      LE314          ;BRANCH IF CTS <= 50 DEG C, ELSE
E30F          bset     *L0089,#0x01   ;SET COOL P/U = 384 OHM
E312          bra      LE31B          ;

E314  LE314: cmpa     #0x6A          ;106 (40 DEG C)
E316          bhi      LE31B          ;BRANCH IF CTS > 40 DEG C, ELSE
E318  LE318: bclr      *L0089,#0x01   ;SET COOL P/U = 4K OHM
E31B  LE31B: ldx      #0x90AB        ;INDEX
E31E          brclr    *L0084,#0x80,LE329 ;BRANCH IF NOT MODE 4, ELSE
E322          ldab     L0231          ;LOAD CONTROL BYTE
E325          andb     #0x20          ;BIT 5 ?
E327          bne      LE350          ;BRANCH IF BIT 5 SET, ELSE
E329  LE329: ldaa     L012E          ;LOAD MALF TIMER
E32C          ldab     L0255          ;LOAD CTS A/D COUNTS
E32F          cmpb     0x2B,x         ;90D6 - LOW LIMIT
E331          bhi      LE352          ;BRANCH IF CTS A/D > LOW LIMIT, ELSE
E333          brset    *L0013,#0x40,LE349 ;BRANCH IF HIGH CTS MALF, ELSE
E337          ldy      *L0032          ;RUN TIME
E33A          cpy      0x29,x         ;90D4 - TIME SINCE RUN ENABLED
E33D          bls      LE377          ;BRANCH IF <= CAL, ELSE
E33F          inca     ;INCREMENT MALF TIMER
E340          cmpa     0x2C,x         ;90D7 - TIMER LIMIT
E342          bcc      LE349          ;BRANCH IF >= TIME LIMIT, ELSE
E344          staa     L012E          ;STORE TIMER
E347          bra      LE377          ;BRANCH

E349  LE349: ldaa     #0x40          ;MASK
E34B          ldab     #0x00          ;OFFSET - 90AB, L0013
E34D          jsr      LF87D          ;GO UPDATE MALF FLAGS
E350          bra      LE380          ;

E352  LE352: clra          ;CLEAR A
E353          bclr      *L0013,#0x40   ;CLEAR HIGH CTS MALF
E356          staa     L012E          ;STORE TIMER
E359          ldaa     L012F          ;LOAD CTS MALF TIMER
E35C          ldab     L0255          ;LOAD CTS A/D COUNTS
E35F          cmpb     0x27,x         ;90D2 - HIGH LIMIT

```

```

E361      bcs      LE3A3                      ;BRANCH IF < HIGH LIMIT, ELSE
E363      brset   *L0013,#0x20,LE379        ;BRANCH IF LOW CTS MALF, ELSE
E367      ld      *L0032                     ;RUN TIME
E36A      cpy     0x25,x                     ;90D0 - TIME SINCE RUN ENABLED
E36D      bls     LE377                      ;BRANCH IF <= CAL, ELSE
E36F      inca    LE377                      ;INCREMENT TIMER
E370      cmpa    0x28,x                     ;90D3 - TIME LIMIT
E372      bcc     LE379                      ;BRANCH IF >= TIME LIMIT, ELSE
E374      staa    L012F                      ;STORE TIMER
E377      LE377:bra LE3C4                     ;RETURN

E379      LE379:ldaa #0x20                    ;MASK
E37B      ldab    #0x00                      ;OFFSET - 90AB, L0013
E37D      jsr     L018D                      ;GO UPDATE MALF FLAGS
E380      LE380:ldd  *L0032                  ;RUN TIME
E382      lsr     LE382                      ;DIV BY 2
E383      cpd     #0x00FF                    ;COMPARE 255
E387      bls     LE38B                      ;BRANCH IF <= 255, ELSE
E389      ldab    #0xFF                      ;LOAD 255
E38B      LE38B:tba LE38B                    ;TRANSFER B TO A REG
E38C      ld      #0x90D8                    ;DEFAULT CLNT TEMP VS RUNTIME
E38F      jsr     L99D7                      ;2D LOOKUP
E392      adda    L018D                      ;ADD IATMAT
E395      bcc     LE399                      ;BRANCH IF NO OVERFLOW, ELSE
E397      ld      #0xFF                      ;LOAD 255
E399      LE399:cmpa L90E9                    ;COMPARE
E39C      bls     LE3AD                      ;BRANCH IF <= MAX, ELSE
E39E      ld      L90E9                      ;LOAD MAX DEFAULT CLNT TEMP
E3A1      bra     LE3AD                      ;BRANCH -> GO STORE CLNT TEMP

E3A3      LE3A3:clra LE3A3                   ;CLEAR A
E3A4      bclr    *L0013,#0x20               ;CLEAR LOW CTS MALF FLAG
E3A7      staa    L012F                      ;STORE TIMER
E3AA      ld      L010A                      ;CLNT TEMP
E3AD      LE3AD:staa *L00A6                  ;STORE CLNT TEMP (DEFAULTED)
E3AF      ld      #0x00A7                    ;OLD FILTERED VALUE
E3B2      ld      #0x86C7                    ;FILTER COEFFICIENT ADDRESS
E3B6      jsr     L99F4                      ;LAG FILTER
E3B9      ldab    *L00A6                     ;LOAD CTS2
E3BB      cmpb    #0xD0                      ;116 DEG C
E3BD      bls     LE3C1                      ;BRANCH IF <= 116, ELSE
E3BF      ldab    #0xD0                      ;LOAD 116 DEG C
E3C1      LE3C1:stab L012A                   ;STORE TRUNCATED CTS FOR TABLE LOOKUPS
E3C4      LE3C4:rts

;-----
; BRANCH HERE FROM SEGMENT TABLE
; TRANS TEMP LOOKUP AND MALF CHECK
;-----
E3C5      ld      #0x40                      ;SEL CH #4 - TRANS TEMP
E3C7      jsr     L9A18                      ;A/D READ
E3CA      staa    L01D4                      ;STORE A/D COUNTS
E3CD      ld      #0xFB6B                    ;384 OHM COOLANT TABLE
E3D0      jsr     L99D7                      ;2D LOOKUP
E3D3      staa    L01D5                      ;STORE TRANS TEMP
E3D6      staa    L01D6                      ;STORE TRANS TEMP
E3D9      ld      #0x90AB                    ;INDEX
E3DC      ld      L0146                      ;TRANS TEMP MALF TIMER
E3DF      brset   *L0016,#0x02,LE3FC        ;BRANCH IF HIGH TRANS TEMP MALF, ELSE
E3E3      ldab    L01D6                      ;LOAD TRANS TEMP
E3E6      cmpb    0xB3,x                     ;915E - TRANS TEMP HIGH LIMIT
E3E8      bcs     LE405                      ;BRANCH IF < HIGH LIMIT, ELSE
E3EA      ld      *L0032                     ;RUN TIME
E3ED      cpy     0xB4,x                     ;915F - 10 SECONDS

```

| | | | |
|------|------------|--------------------|-----------------------------------|
| E3F0 | bls | LE403 | ;BRANCH IF <= 10 SECONDS, ELSE |
| E3F2 | inca | | ;INCREMENT TIMER |
| E3F3 | cmpa | 0xB6,x | ;9161 - 1 SECOND |
| E3F5 | bcc | LE3FC | ;BRANCH IF > TIME LIMIT, ELSE |
| E3F7 | staa | L0146 | ;STORE TIMER |
| E3FA | bra | LE403 | |
| | | | |
| E3FC | LE3FC:ldaa | #0x02 | ;MASK |
| E3FE | ldab | #0x03 | ;OFFSET - 90AE, L0016 |
| E400 | jsr | LF87D | ;GO UPDATE MALF FLAGS |
| E403 | LE403:bra | LE433 | |
| | | | |
| E405 | LE405:clra | | ;CLEAR A |
| E406 | bclr | *L0016,#0x02 | ;CLEAR HIGH TRANS TEMP MALF |
| E409 | staa | L0146 | ;STORE TIMER |
| E40C | ldaa | L0147 | ;LOAD MALF TIMER |
| E40F | brset | *L0016,#0x01,LE42C | ;BRANCH IF LOW TRANS TEMP, ELSE |
| E413 | ldab | L01D4 | ;LOAD TRANS TEMP A/D COUNTS |
| E416 | cmpb | 0xAF,x | ;915A - TRANS A/D HIGH LIMIT |
| E418 | bcs | LE43A | ;BRANCH IF < 915A |
| E41A | ldy | *L0032 | ;RUN TIME |
| E41D | cpy | 0xB0,x | ;915B -186 SECONDS |
| E420 | bls | LE433 | ;BRANCH IF <= 915B, ELSE |
| E422 | inca | | ;INCREMENT TIMER |
| E423 | cmpa | 0xB2,x | ;915D - 1.6 SECONDS |
| E425 | bcc | LE42C | ;BRANCH IF >= TIME LIMIT, ELSE |
| E427 | staa | L0147 | ;STORE TIMER |
| E42A | bra | LE433 | ;BRANCH |
| | | | |
| E42C | LE42C:ldaa | #0x01 | ;MASK |
| E42E | ldab | #0x03 | ;OFFSET - 90AE, L0016 |
| E430 | jsr | LF87D | ;GO UPDATE MALF FLAGS |
| E433 | LE433:ldaa | *L00A6 | ;LOAD CLNT TEMP (DEFAULTED) |
| E435 | staa | L01D5 | ;TRANS TEMP |
| E438 | bra | LE441 | |
| | | | |
| E43A | LE43A:clra | | ;CLEAR A |
| E43B | bclr | *L0016,#0x01 | ;CLEAR MALF FLAG |
| E43E | staa | L0147 | ;STORE MALF TIMER |
| E441 | LE441:ldaa | L01D5 | ;NEW UNFILTERED TRANS TEMP |
| E444 | ldx | #0x01D7 | ;OLD FILTERED TRANS TEMP |
| E447 | ldy | #0x941B | ;FILTER COEFFICIENT ADDRESS |
| E44B | jsr | L99F4 | ;LAG FILTER |
| E44E | ldaa | L941D | |
| E451 | brset | *L0090,#0x20,LE458 | ; |
| E455 | ldaa | L941C | |
| E458 | LE458:bclr | *L0090,#0x20 | |
| E45B | cmpa | L01D7 | ;FILTERED TRANS TEMP |
| E45E | bcc | LE463 | |
| E460 | bset | *L0090,#0x20 | |
| E463 | LE463:ldaa | L8016 | ;OPTION WORD - TIS SET |
| E466 | beq | LE4A4 | ;BRANCH IF CLEAR, ELSE |
| E468 | brset | *L0095,#0x10,LE4B3 | ;BRANCH IF BATT VOLTS < 4.0, ELSE |
| E46C | ldaa | #0x80 | |
| E46E | jsr | LD90E | ;TEST MODE 4 CONTROL WORDS |
| E471 | beq | LE484 | |
| E473 | bmi | LE47B | |
| E475 | clr | L01E7 | ;CLEAR DELAY TIMER |
| E478 | jmp | LE4FB | |
| | | | |
| E47B | LE47B:ldaa | L9857 | ;OIL LIGHT TURN ON DELAY TIME |
| E47E | staa | L01E7 | ;DELAY TIMER |
| E481 | jmp | LE4FE | ;RETURN |

```

E484      LE484:bset      *L0077,#0x01      ;SET IGN ON - FOR LOW OIL LEVEL LOGIC
E487              brclr   *L0077,#0x02,LE4FE ;BRANCH IF IGN ON, ELSE
E48B              bclr    *L0077,#0x02      ;SET IGN OFF - FOR LOW OIL LEVEL LOGIC
E48E              ldaa     L9857              ;OIL LIGHT TURN ON DELAY TIME
E491              staa     L01E7              ;STORE DELAY TIMER
E494              brclr   *L0077,#0x08,LE4A6 ;BRANCH IF ENGINE WARM, ELSE
E498              ldaa     *L00F2            ;LOAD STARTUP COOLANT
E49A              suba     *L0080
E49C              bcc      LE4FE              ;RETURN
E49E              nega
E49F              cmpa     L9852
E4A2              bcc      LE4AA
E4A4      LE4A4:bra      LE4FE              ;RETURN

E4A6      LE4A6:brclr   *L0077,#0x10,LE4FE ;BRANCH IF OIL NOT DRAINED BACK, ELSE
E4AA      LE4AA:brclr   *L008E,#0x20,LE4FB ;BRANCH IF NOT LOW OIL LEVEL, ELSE
E4AE              bset     *L0094,#0x20      ;SET LOW OIL
E4B1              bra      LE4FE              ;RETURN

E4B3      LE4B3:bset     *L0077,#0x02      ;SET IGN OFF - FOR LOW OIL LEVEL LOGIC
E4B6              brclr   *L0077,#0x01,LE4DF ;BRANCH IF IGN OFF, ELSE
E4BA              bclr     *L0077,#0x19      ;%00011001
                                      ;CLEAR IGN ON - FOR LOW OIL LEVEL LOGIC
                                      ;CLEAR ENGINE COLD - FOR LOW OIL LEVEL
                                      ;CLEAR OIL DRAINED BACK
E4BD              brset    *L0013,#0x20,LE4FE ;BRANCH IF LOW CTS MALF SET, ELSE
E4C1              brset    *L0013,#0x40,LE4FE ;BRANCH IF HIGH CTS MALF SET, ELSE
E4C5              ldaa     *L00A6            ;LOAD CLNT TEMP (DEFAULTED)
E4C7              staa     *L0080            ;STORE
E4C9              cmpa     L9853              ;COMPARE
E4CC              bls      LE4FE              ;BRANCH IF <= CAL, ELSE
E4CE              cmpa     L9854              ;
E4D1              bcs      LE4DA              ;BRANCH IF < CAL, ELSE
E4D3              ldx      L9855              ;LOAD TIME FOR OIL TO DRAIN BACK
E4D6              stx      *L007E            ;SAVE TIMER
E4D8              bra      LE4FE              ;RETURN

E4DA      LE4DA:bset     *L0077,#0x08      ;SET ENGINE COLD
E4DD              bra      LE4FE              ;RETURN

E4DF      LE4DF:ldx      *L007E
E4E1              beq      LE4E6              ;BRANCH IF Z, ELSE
E4E3              dex
E4E4              stx      *L007E            ;SAVE TIMER
E4E6      LE4E6:bne      LE4ED              ;BRANCH IF NOT Z, ELSE
E4E8              bset     *L0077,#0x10      ;SET OIL DRAINED BACK - FOR LOW OIL LVL
E4EB              bra      LE4FE              ;RETURN

E4ED      LE4ED:ldx      *L0058              ;TIMER
E4EF              cpx      #0x0050            ;80 ?
E4F2              bls      LE4FE              ;BRANCH IF <= 80 (RETURN), ELSE
E4F4              ldx      #0x0051            ;LOAD 81
E4F7              stx      *L0058            ;STORE TIMER
E4F9              bra      LE4FE              ;RETURN

E4FB      LE4FB:bclr     *L0094,#0x20      ;CLEAR LOW OIL
E4FE      LE4FE:rts

;-----
; BRANCH HERE FROM SEGMENT TABLE
; ESC, INCUDING MALF CHECK
;-----
E4FF      ldaa     *L00EF
E501      cmpa     #0x0F

```

```

E503      bhi      LE518      ;
E505      ldaa     *L00AB      ;LOAD NLRPMX
E507      lsra     ;RESCALE
E508      lsra
E509      ldx      #0x832C      ;ESC RECOVERY RATE VS RPM
E50C      jsr      L99D7      ;2D LOOKUP
E50F      tab      ;TRANSFER TABLE LOOKUP RESULT TO B REG
E510      ldaa     *L00F3      ;KNOCK RETARD DEGREES
E512      sba      ;SUBTRACT B FROM A
E513      bcc      LE516      ;BRANCH IF NO UNDERFLOW, ELSE
E515      clra     ;CLEAR A
E516      LE516:staa *L00F3      ;STORE KNOCK RETARD DEGREES
E518      LE518:ldaa #0x00      ;SEL CH $0 - KNOCK SENSOR
E51A      jsr      L9A18      ;A/D READ
E51D      ldx      #0x90AB      ;INDEX
E520      staa     L046C      ;STORE ESC A/D READ
E523      cmpa     0x6A,x      ;9115
E525      bcs      LE52B      ;BRANCH IF A/D COUNTS < 9115, ELSE
E527      cmpa     0x6B,x      ;9116
E529      bcs      LE544      ;BRANCH IF A/D COUNTS < 9116, ELSE
E52B      LE52B:brset *L0016,#0x10,LE537 ;BRANCH IF KNOCK SENSOR MALF, ELSE
E52F      ldaa     L0143      ;LOAD MALF TIMER
E532      inca     ;INCREMENT TIMER
E533      cmpa     0x68,x      ;9113
E535      bcs      LE548
E537      LE537:ldaa #0x10      ;MASK
E539      ldab     #0x03      ;OFFSET 90AE, L0016
E53B      jsr      LF87D      ;GO UPDATE MALF FLAGS
E53E      ldab     0x69,x      ;9114 - DEFAULT KNOCK RETARD DEGREES
E540      stab     *L00F3      ;KNOCK RETARD DEGREES
E542      bra      LE54B

E544      LE544:clra     ;CLEAR A
E545      bclr     *L0016,#0x10 ;CLEAR KNOCK SENSOR MALF
E548      LE548:staa     L0143 ;STORE

;-----
; A/C PRESSURE SENSOR LOOKUP AND MALF CHECK
;-----
E54B      LE54B:ldaa     L8009      ;OPTION WORD - A/C PRESSURE SENSOR PRESENT
E54E      beq      LE57F      ;BRANCH IF CLEAR (TIS SET)
E550      ldaa     #0xA0      ;SEL CH #10 - A/C PRESSURE TRANSDUCER
E552      jsr      L9A18      ;A/D READ
E555      staa     *L00F9      ;STORE A/D READ - A/C PRESSURE
E557      tab      ;TRANSFER TO B REG
E558      ldx      #0x90AB      ;MALF INDEX
E55B      ldaa     L0148      ;MALF TIMER
E55E      cmpb     0x92,x      ;913D - A/C PRESS HIGH LIMIT
E560      bhi      LE566      ;BRANCH IF > LIMIT, ELSE
E562      cmpb     0x93,x      ;913E - A/C PRESS LOW LIMIT
E564      bcc      LE578      ;BRANCH IF >= LOW LIMIT, ELSE
E566      LE566:brset *L0019,#0x08,LE56F ;BRANCH IF A/C PRESSURE SENSOR MALF, ELSE
E56A      inca     ;INCREMENT TIMER
E56B      cmpa     0x94,x      ;913F
E56D      bls      LE57C      ;BRANCH IF <= TIME LIMIT, ELSE
E56F      LE56F:ldaa     #0x08      ;MASK
E571      ldab     #0x06      ;OFFSET 90B1, L0019
E573      jsr      LF87D      ;GO UPDATE MALF FLAGS
E576      bra      LE57F      ;BRANCH - RETURN

E578      LE578:clra     ;CLEAR A
E579      bclr     *L0019,#0x08 ;CLEAR A/C PRESSURE SENSOR MALF FLAG
E57C      LE57C:staa     L0148      ;STORE TIMER
E57F      LE57F:rts

```

```

;-----
; BRANCH HERE FROM SEGMENT TABLE
; IAT SENSOR LOOKUP AND MALF CHECK
;-----
E580 LE580:ldaa    #0x90                ;SEL CH #9 - IAT SENSOR
E582      jsr     L9A18                ;A/D READ
E585      coma    ;INVERT RESULT
E586      staa    L0254                ;IAT INVERSE A/D COUNTS
E589      ldx     #0x90AB              ;INDEX
E58C      brclr   *L0084,#0x80,LE597  ;BRANCH IF NOT MODE 4, ELSE
E590      ldab    L0231                ;LOAD CONTROL BYTE
E593      andb    #0x40                ;BIT 6 ?
E595      bne     LE5E9                ;BRANCH IF BIT 6 SET, ELSE
E597 LE597:cmpa    0x1B,x              ;90C6 - IAT LOW LIMIT
E599      bhi     LE5C2                ;BRANCH IF > LOW LIMIT, ELSE
E59B      brset   *L0014,#0x40,LE5B9  ;BRANCH IF LOW IAT MALF, ELSE
E59F      ldaa    *L00A6                ;LOAD CLNT TEMP (DEFAULTED)
E5A1      cmpa    0x1C,x              ;90C7
E5A3      bcs     LE5C2                ;BRANCH IF < HIGH LIMIT
E5A5      ldaa    *L00CD                ;LOAD MAF (GM/SEC)
E5A7      cmpa    0x1D,x              ;90C8
E5A9      bhi     LE5C2                ;BRANCH IF > 90C8, ELSE
E5AB      ldaa    *L00BD                ;FILTERED MPH
E5AD      cmpa    0x1E,x              ;90C9
E5AF      bhi     LE5C2                ;BRANCH IF > 90C9, ELSE
E5B1      ldaa    L0135                ;MALF TIMER
E5B4      inca    ;INCREMENT TIMER
E5B5      cmpa    0x1F,x              ;90CA
E5B7      bcs     LE5C6                ;BRANCH IF < 90CA, ELSE
E5B9 LE5B9:ldaa    #0x40                ;MASK
E5BB      ldab    #0x01                ;OFFSET - 90AC, L0014
E5BD      jsr     LF87D                ;GO UPDATE MALF FLAGS
E5C0      bra     LE5E9

E5C2 LE5C2:clra    ;CLEAR A
E5C3      bclr    *L0014,#0x40        ;CLEAR LOW IAT MALF
E5C6 LE5C6:staa    L0135                ;STORE TIMER
E5C9      ldaa    L0254                ;INVERSE IAT A/D COUNTS
E5CC      cmpa    0x21,x              ;90CC - IAT A/D HIGH LIMIT
E5CE      bls     LE5ED                ;BRANCH IF <= LIMIT, ELSE
E5D0      brset   *L0014,#0x10,LE5E2  ;BRANCH IF HIGH IAT MALF, ELSE
E5D4      ldaa    *L00BD                ;LOAD FILTERED MPH
E5D6      cmpa    0x22,x              ;90CD
E5D8      bcs     LE5ED                ;BRANCH IF < 90CD, ELSE
E5DA      ldaa    L0139                ;LOAD TIMER
E5DD      inca    ;INCREMENT TIMER
E5DE      cmpa    0x24,x              ;90CF - TIME LIMIT
E5E0      bls     LE5F1                ;BRANCH IF <= TIME LIMIT, ELSE
E5E2 LE5E2:ldaa    #0x10                ;MASK
E5E4      ldab    #0x01                ;OFFSET - 90AC, L0014
E5E6      jsr     LF87D                ;GO UPDATE MALF FLAGS
E5E9 LE5E9:ldaa    0x20,x              ;90CB
E5EB      bra     LE5F7                ;GO STORE IAT A/D COUNTS

E5ED LE5ED:clra    ;CLEAR A
E5EE      bclr    *L0014,#0x10        ;CLEAR HIGH IAT MALF
E5F1 LE5F1:staa    L0139                ;STORE TIMER
E5F4      ldaa    L0254                ;INVERSE IAT A/D COUNTS
E5F7 LE5F7:staa    L0195                ;STORE IAT A/D COUNTS
E5FA      ldx     #0xFB49              ;MAT LOOKUP TABLE
E5FD      jsr     L99D7                ;2D LOOKUP
E600      staa    *L00F0                ;IATMAT (DEG C)
E602      brclr   *L0095,#0x10,LE612  ;BRANCH IF BATT VOLTS NOT < 4.0, ELSE

```

```

E606          psha                                ;SAVE ON STACK
E607          ldd      *L0032                      ;LOAD RUN TIME
E609          cpd      #0x000A                      ;COMPARE 10
E60D          pula                                ;GET BACK IATMAT
E60E          bls      LE612                        ;BRANCH IF <= 10, ELSE
E610          staa     *L006F                      ;STORE IATMAT
E612  LE612:brclr  *L0086,#0x01,LE61C              ;BRANCH IF NOT BIT 0, ELSE
E616          staa     L018D                        ;STORE IATMAT
E619          clrb
E61A          bra      LE62B

E61C  LE61C:cmpa   L018E                            ;FILTERED IATMAT
E61F          bcc      LE62E
E621          ldx      #0x018E                      ;OLD UNFILTERED VALUE
E624          ldy      #0x8F04                      ;FILTER COEFFICIENT ADDRESS
E628          jsr      L99F4                        ;LAG FILTER
E62B  LE62B:std    L018E                            ;FILTERED IATMAT

;-----
; DTC P1626 - PASSKEY II SIGNAL CHECK - DID FREQUENCY CHANGE AFTER STARTUP ?
;-----
E62E  LE62E:ldx    #0x90AB                          ;INDEX
E631          ldaa     L8011                        ;PASSKEY II OPTION WORD
E634          beq      LE691                        ;BRANCH IF Z, ELSE (TIS SET)
E636          brset    *L0095,#0x10,LE691           ;BRANCH IF BATT VOLTS < 4.0, ELSE
E63A          ldd      L3DC6                        ;LOAD VATS INPUT
E63D          subb     L026F                        ;SUBTRACT PREV VALUE
E640          cmpb     0x9E,x                        ;9149 - 4
E642          bcs      LE65F                        ;BRANCH IF < LOW LIMIT, ELSE
E644          cmpb     0x9F,x                        ;914A - 6
E646          bhi      LE65F                        ;BRANCH IF > HIGH LIMIT, ELSE
E648          brset    *L0092,#0x08,LE651           ;BRANCH IF BIT 3, ELSE
E64C          bset     *L0092,#0x08                 ;SET BIT 3
E64F          bra      LE65A                        ;
E651  LE651:bclr   *L0018,#0x40                     ;CLEAR PASSKEY II ERROR
E654          bclr     *L0092,#0x04                 ;CLEAR VATS FUEL SHUTOFF
E657          bset     *L0012,#0x02                 ;SET VATS OKAY
E65A  LE65A:clr    L026E                            ;MALF TIMER
E65D          bra      LE691

E65F  LE65F:brclr  *L0018,#0x40,LE66B              ;BRANCH IF NO PASSKEY II ERROR, ELSE
E663          bset     *L0012,#0x02                 ;SET VATS OKAY/SIGNAL PRESENT
E666          clr      L026E                        ;CLEAR MALF TIMER
E669          bra      LE682                        ;GO SET MALF FLAG

E66B  LE66B:ldab   L026E                            ;MALF TIMER
E66E          incb
E66F          cmpb     L914B                        ;INCREMENT TIMER
E672          bhi      LE679                        ;COMPARE TIME LIMIT
E674          stab     L026E                        ;BRANCH IF > TIME LIMIT, ELSE
E677          bra      LE689                        ;STORE TIMER

E679  LE679:brset  *L0012,#0x02,LE682              ;BRANCH IF VATS OKAY, ELSE
E67D          bset     *L0092,#0x04                 ;SET VATS FUEL SHUTOFF
E680          bra      LE689

E682  LE682:ldaa   #0x40                            ;MASK
E684          ldab     #0x05                        ;OFFSET - 90B0, L0018
E686          jsr      LF87D                        ;GO UPDATE MALF FLAGS
E689  LE689:bclr   *L0092,#0x08                     ;CLEAR BIT 3
E68C          bra      LE691

E68E          bclr     *L0018,#0x40                 ;CLEAR PASSKEY II SIGNAL ERROR

```

```

E691 LE691:ldd      L3DC6                ;LOAD VATS INPUT
E694      stab     L026F                ;SAVE LSB FOR NEXT PASS
E697      rts

;-----
; BRANCH HERE FROM SEGMENT TABLE
; CHECK BATT VOLTS
;-----
E698 LE698:ldaa     #0x50                ;SEL CH #5 - BATT
E69A      jsr      L9A18                ;A/D READ
E69D      staa     *L0083                ;STORE BATT VOLTS
E69F      ldab     *L0095
E6A1      cmpa     #0x5A                ;9.0 VOLTS
E6A3      bcc      LE6AD                ;BRANCH IF >= 9.0 VOLTS, ELSE
E6A5      cmpa     #0x28                ;4.0 VOLTS
E6A7      bcc      LE6AF                ;BRANCH IF >= 4.0 VOLTS, ELSE
E6A9      orab     #0x10                ;SET BATT VOLTS < 4.0, ELSE
E6AB      bra      LE6AF

E6AD LE6AD:andb     #0xEF                ;CLEAR BATT VOLTS < 4.0, ELSE
E6AF LE6AF:stab     *L0095                ;STORE FLAG
E6B1      cmpa     #0xAB                ;17.1 VOLTS
E6B3      bcs      LE6C1                ;BRANCH IF < 17.1 VOLTS, ELSE
E6B5      brclr    *L0097,#0x40,LE6BC    ;BRANCH IF NOT HIGH BATT VOLTS, ELSE
E6B9      bset     *L0097,#0x20          ;SET HIGH BATT VOLTAGE FLAG
E6BC LE6BC:bset     *L0097,#0x40          ;SET HIGH BATT VOLTAGE FLAG
E6BF      bra      LE6C4

;-----
; SYSTEM VOLTS CHECK
;-----
E6C1 LE6C1:bclr     *L0097,#0x60          ;CLEAR BATT VOLTAGE FLAGS
E6C4 LE6C4:ldx      #0x90AB                ;INDEX
E6C7      ldaa     L0130                ;MALF TIMER
E6CA      ldab     *L0083                ;BATT VOLTS
E6CC      cmpb     0xA1,x                ;914C - HIGH LIMIT
E6CE      bhi      LE6E1                ;BRANCH IF BATT VOLTS > HIGH LIMIT, ELSE
E6D0      cmpb     0xA2,x                ;914D - LOW LIMIT
E6D2      bcc      LE6F3                ;BRANCH IF BATT VOLTS >= LOW LIMIT, ELSE
E6D4      brset    *L0013,#0x10,LE6EA    ;BRANCH IF LOW SYSTEM VOLTS, ELSE
E6D8      ldab     *L00AC                ;LOAD RPM/25
E6DA      cmpb     L90ED                ;RPM
E6DD      bhi      LE6E5                ;BRANCH IF RPM/25 > MIN VALUE, ELSE
E6DF      bra      LE6F3                ;BRANCH

E6E1 LE6E1:brset    *L0013,#0x10,LE6EA    ;BRANCH IF LOW SYSTEM VOLTS, ELSE
E6E5 LE6E5:inca                ;INCREMENT TIMER
E6E6      cmpa     0xA3,x                ;914E - TIME LIMIT
E6E8      bcs      LE6F7                ;BRANCH IF < TIME LIMIT, ELSE
E6EA LE6EA:ldaa     #0x10                ;MASK
E6EC      ldab     #0x00                ;OFFSET - 90AB, L0013
E6EE      jsr      LF87D                ;GO UPDATE MALF FLAGS
E6F1      bra      LE6FA

E6F3 LE6F3:clra                ;CLEAR A
E6F4      bclr     *L0013,#0x10          ;CLEAR LOW SYSTEM VOLTS MALF FLAG
E6F7 LE6F7:staa     L0130                ;STORE TIMER
E6FA LE6FA:ldab     *L0095                ;
E6FC      rts

;-----
; BRANCH HERE FROM SEGMENT TABLE
; CRUISE CONTROL
;-----

```

```

E6FD      ldaa    #0x40                ;CC INHIBIT SIGNAL
E6FF      jsr     LD90E                ;TEST MODE 4 CONTROL WORDS
E702      beq     LE709                ;BRANCH IF Z, ELSE
E704      bmi     LE77E                ;BRANCH IF MINUS, ELSE
E706      jmp     LE786

E709      LE709:ldaa    L800C                ;OPTION WORD - TIS SET
E70C      bne     LE711                ;BRANCH IF NOT Z, ELSE
E70E      bset    *L0077,#0x04          ;SET STEPPER MOTOR CRUISE PRESENT
E711      LE711:brset  *L0077,#0x04,LE72C    ;BRANCH IF SMC PRESENT, ELSE
E715      brclr   *L0099,#0x02,LE720      ;
E719      brset   *L0097,#0x01,LE720      ;BRANCH IF SMC ENGAGED, ELSE
E71D      bset    *L0077,#0x04          ;SET STEPPER MOTOR CRUISE PRESENT
E720      LE720:bclr   *L0099,#0x02      ;
E723      brclr   *L0097,#0x01,LE738      ;BRANCH IF SMC NOT ENGAGED, ELSE
E727      bset    *L0099,#0x02          ;
E72A      bra     LE738

E72C      LE72C:brset  *L0097,#0x01,LE735    ;BRANCH IF SMC ENGAGED, ELSE
E730      bclr    *L00D9,#0x40          ;CLEAR SMC ENGAGED
E733      bra     LE738

E735      LE735:bset   *L00D9,#0x40          ;SET SMC ENGAGED
E738      LE738:ldx    *L00ED            ;1 SECOND TIMER
E73A      cpx     L9412                ;TIME LIMIT TO ALLOW CRUISE CONTROL
E73D      bls     LE786                ;BRANCH IF <= 10 SECONDS, ELSE
E73F      brset   *L0015,#0x40,LE786      ;BRANCH IF TRANS RANGE SW MALF, ELSE
E743      brset   *L00A2,#0x01,LE786      ;
E747      ldaa    *L00AC                ;LOAD RPM/25
E749      cmpa    L9414                ;6375
E74C      bhi     LE786                ;BRANCH IF > CAL, ELSE
E74E      cmpa    L9415                ;
E751      bcs     LE786                ;
E753      ldaa    L9418                ;
E756      cmpa    L0310                ;FILTERED MPH
E759      bcs     LE786                ;BRANCH IF > CAL, ELSE
E75B      ldaa    L9416                ;
E75E      brclr   *L00D9,#0x40,LE765      ;BRANCH IF SMC NOT ENGAGED, ELSE
E762      ldaa    L9417                ;
E765      LE765:cmpa    L0310                ;FILTERED MPH
E768      bhi     LE786                ;BRANCH IF MPH < CAL, ELSE
E76A      brset   *L008E,#0x01,LE786      ;BRANCH IF P/N MODE, ELSE
E76E      brset   *L00A1,#0x02,LE786      ;BRANCH IF PRNDL -> DRIVE 1, ELSE
E772      brset   *L00A1,#0x40,LE786      ;BRANCH IF PRNDL -> REVERSE, ELSE
E776      brset   *L0013,#0x10,LE786      ;BRANCH IF LOW SYSTEM VOLTS MALF, ELSE
E77A      brset   *L008E,#0x02,LE786      ;BRANCH IF NOT 2ND GEAR, ELSE
E77E      LE77E:ldx    #0xDFFF          ;MAX DUTY CYCLE
E781      bclr    *L0099,#0x10          ;ALLOW CRUISE CONTROL
E784      bra     LE78C                ;STORE INHIBIT SIGNAL

E786      LE786:ldx    #0xD000                ;0 DUTY CYCLE
E789      bset    *L0099,#0x10          ;INHIBIT CRUISE CONTROL
E78C      LE78C:stx    L3FD4                ;CC INHIBIT SIGNAL OUTPUT

;-----
; LOOKUP VARIOUS THINGS
;-----
E78F      ldd     *L00BD                ;FILTERED MPH
E791      lsld                    ;MUL BY 2
E792      bcc     LE796                ;BRANCH IF OVERFLOW, ELSE
E794      ldaa    #0xFF                ;LOAD 255
E796      LE796:ldx    #0x8D0B          ;TABLE ADDRESS - TPS MULTIPLIER
E799      jsr     L99D7                ;2D LOOKUP
E79C      staa    L0320                ;SAVE LOOKUP RESULT

```

```

E79F      ldd      *L00BD      ;FILTERED MPH
E7A1      cmpa     #0x40      ;64
E7A3      bcs     LE7A8      ;BRANCH IF < 64 MPH, ELSE
E7A5      ldd      #0x3FFF      ;LOAD
E7A8      LE7A8:lsld      ;MUL BY 2
E7A9      ldx      #0x8D1C      ;
E7AC      jsr      L99D7      ;2D LOOKUP
E7AF      staa     L0321      ;STORE
E7B2      ldx      #0x8D47      ;
E7B5      ldaa     *L00F0      ;IATMAT
E7B7      lsra     ;RESCALE
E7B8      jsr      L99D7      ;2D LOOKUP
E7BB      staa     L0322      ;STORE
E7BE      ldd      *L00BD      ;FILTERED MPH
E7C0      cmpa     #0x40      ;64
E7C2      bls     LE7C7      ;BRANCH IF < 64 MPH, ELSE
E7C4      ldd      #0x4000      ;LOAD
E7C7      LE7C7:lsld      ;MUL BY 2
E7C8      ldx      #0x8EBD      ;THROTTLE FOLLOWER A/F VS FMPH
E7CB      jsr      L99D7      ;2D LOOKUP
E7CE      staa     L0323      ;STORE THROTTLE FOLLOWER A/F
E7D1      ldx      #0x8E86      ;STARTUP A/F OFFSET
E7D4      ldd      *L00ED      ;1 SECOND TIMER
E7D6      cpd      #0x0040      ;64 SECONDS?
E7DA      bcs     LE7DE      ;BRANCH IF < 64 SECONDS, ELSE
E7DC      ldab     #0x3F      ;
E7DE      LE7DE:ldaa     *L00A7      ;FILTERED CTS
E7E0      lsra     ;RESCALE
E7E1      jsr      L995A      ;3D LOOKUP
E7E4      staa     L0324      ;STORE STARTUP A/F
E7E7      rts      ;

;-----
; BRANCH HERE FROM SEGMENT TABLE
; DIAGNOSTICS
;-----

E7E8      brset    *L0086,#0x80,LE7EF      ;BRANCH IF ENGINE RUNNING, ELSE
E7EC      jmp      LEB19      ;SKIP QDM MALF CHECK

;-----
; QDM MALF CHECK
;-----

E7EF      LE7EF:bset    *L0094,#0x02      ;SET QDM FAULT1
E7F2      ldaa     #0x70      ;SEL CH #7
E7F4      jsr      L9A18      ;A/D READ
E7F7      cmpa     #0x28      ;40
E7F9      bcc     LE7FE      ;BRANCH IF >= 40, ELSE
E7FB      bclr     *L0094,#0x02      ;CLEAR QDM FAULT1
E7FE      LE7FE:bset    *L0094,#0x04      ;SET QDM FAULT2
E801      ldaa     #0x80      ;SEL CH #8
E803      jsr      L9A18      ;A/D READ
E806      cmpa     #0x28      ;40
E808      bcc     LE80D      ;BRANCH IF >= 40, ELSE
E80A      bclr     *L0094,#0x04      ;CLEAR QDM FAULT2
E80D      LE80D:ldx     #0x90AB      ;INDEX
E810      ldd      *L00ED      ;1 SECOND TIMER
E812      cpd      L914F      ;5 SECONDS?
E816      bcs     LE86B      ;BRANCH IF < 5 SECONDS, ELSE
E818      ldab     *L0094      ;QDMMW
E81A      brset    *L008E,#0x40,LE827      ;BRANCH IF BRAKE APPLIED, ELSE
E81E      ldaa     L800F      ;OPTION WORD - TIS SET
E821      bne      LE829      ;BRANCH IF NOT Z, ELSE
E823      brclr    *L008E,#0x02,LE829      ;BRANCH IF 2ND GEAR, ELSE
E827      LE827:orab     #0x02      ;SET QDM FAULT1

```

```

E829    LE829:ldaa    L013A                ;LOAD TIMER
E82C        andb    #0x02                ;CLEAR EVERYTHING EXCEPT BIT 1
E82E        eorb    0xA6,x                ;TOGGLE BITS 9151
E830        andb    0xA6,x                ;CLEAR BITS
E832        beq     LE842                ;BRANCH IF Z, ELSE
E834        inca    ;INCREMENT TIMER
E835        cmpa    0xA7,x                ;9152 - 5 SECONDS
E837        bls     LE846                ;BRANCH IF <= 5 SECONDS, ELSE
E839        ldaa    #0x08                ;MASK
E83B        ldab    #0x01                ;OFFSET - 90AC, L0014
E83D        jsr     LF87D                ;GO UPDATE MALF FLAGS
E840        bra     LE849

E842    LE842:clra    ;
E843        bclr    *L0014,#0x08        ;CLEAR QDM A MALF
E846    LE846:staa    L013A                ;STORE MALF TIMER

;-----
; DTC P1650 - QDM B MALF CHECK
;-----
E849    LE849:ldaa    L013B                ;
E84C        ldab    *L0094                ;QDMMW
E84E        andb    #0x04                ;SET BIT 2 - QDM FAULT2
E850        eorb    0xA9,x                ;9154
E852        andb    0xA9,x                ;9154
E854        beq     LE864                ;
E856        inca    ;INCREMENT TIMER
E857        cmpa    0xA7,x                ;9152
E859        bls     LE868                ;BRANCH IF <= TIME LIMIT, ELSE
E85B        ldaa    #0x01                ;MASK
E85D        ldab    #0x04                ;OFFSET - 90AF, L0017
E85F        jsr     LF87D                ;GO UPDATE MALF FLAGS
E862        bra     LE86B                ;

E864    LE864:clra    ;CLEAR TIMER
E865        bclr    *L0017,#0x01        ;CLEAR QDM B MALF
E868    LE868:staa    L013B                ;STORE TIMER

;-----
; DTC P0134 - O2 SENSOR MALF CHECK
;-----
E86B    LE86B:ldab    *L00A9                ;MINOR LOOP O2 A/D COUNTS (mV)
E86D        cmpb    0x63,x                ;910E
E86F        bhi     LE8A6                ;BRANCH IF > LOW LIMIT, ELSE
E871        cmpb    0x62,x                ;910D
E873        bls     LE8A6                ;BRANCH IF <= LOW LIMIT, ELSE
E875        brset   *L0013,#0x80,LE89D    ;BRANCH IF O2 A/D COUNTS MALF, ELSE
E879        ldd     *L0032                ;RUN TIME
E87B        subd    0x60,x                ;4 SECONDS
E87D        bls     LE8A6                ;BRANCH IF <= 4 SECONDS, ELSE
E87F        brclr   *L008F,#0x08,LE8AD    ;BRANCH IF NOT BIT 3, ELSE
E883        ldaa    L012D                ;LOAD TIMER
E886        ldab    *L00A6                ;LOAD CLNT TEMP (DEFAULTED)
E888        cmpb    0x66,x                ;9111
E88A        bls     LE8A6                ;BRANCH IF <= COOLANT LIMIT, ELSE
E88C        ldab    *L00AA                ;%TPS
E88E        cmpb    0x64,x                ;910F
E890        bhi     LE898                ;BRANCH IF > TPS% LIMIT, ELSE
E892        suba    #0x01                ;SUB 1
E894        bcs     LE8AD                ;BRANCH IF TIMER < 1, ELSE
E896        bra     LE8AA

E898    LE898:inca    ;INCREMENT TIMER
E899        cmpa    0x65,x                ;9110 - TIME LIMIT

```

```

E89B      bls      LE8AA      ;BRANCH IF <= TIME LIMIT, ELSE
E89D  LE89D:ldaa   #0x80      ;MASK
E89F      ldab     #0x00      ;OFFSET - 90AB, L0013
E8A1      jsr      LF87D      ;GO UPDATE MALF FLAGS
E8A4      bra      LE8AD

E8A6  LE8A6:clra      ;CLEAR A
E8A7      bclr     *L0013,#0x80 ;CLEAR O2 A/D COUNTS MALF
E8AA  LE8AA:staa     L012D      ;STORE TIMER

;-----
; DTC P0321 - 24x CRANK SIGNAL MALF CHECK
;-----
E8AD  LE8AD:brset   *L0013,#0x08,LE8B8 ;BRANCH IF 24X CRANK MALF, ELSE
E8B1      ldaa     L0131      ;LOAD 24X PULSE COUNTER
E8B4      cmpa     0x67,x      ;9112 - 150
E8B6      bcs      LE8BF      ;BRANCH IF < 150, ELSE
E8B8  LE8B8:ldaa   #0x08      ;MASK
E8BA      ldab     #0x00      ;OFFSET - 90AB, L0013
E8BC      jsr      LF87D      ;GO UPDATE MALF FLAGS

;-----
; DTC P0341 - INTERMITTENT CAM SIGNAL
;-----
E8BF  LE8BF:brset   *L0013,#0x04,LE8CA ;BRANCH IF CAM SIGNAL ERROR, ELSE
E8C3      ldaa     L0132      ;M341 TIMER - ALDL LIST
E8C6      cmpa     0x6C,x      ;9117
E8C8      bcs      LE8D1      ;BRANCH IF < HIGH LIMIT, ELSE
E8CA  LE8CA:ldaa   #0x04      ;MASK
E8CC      ldab     #0x00      ;OFFSET - 90AB, L0013
E8CE      jsr      LF87D      ;GO UPDATE MALF FLAGS

;-----
; DTC P0502 CHECK - NO VSS SIGNAL
;-----
E8D1  LE8D1:clra      ;CLEAR A
E8D2      brset   *L008E,#0x01,LE8DC ;BRANCH IF P/N MODE, ELSE
E8D6      ldaa     L0138      ;LOAD TRANSAXLE OUT OF P/N TIMER
E8D9      inca      ;INCREMENT TIMER
E8DA      beq      LE8DF      ;BRANCH IF = ZERO, ELSE
E8DC  LE8DC:staa     L0138      ;STORE TRANSAXLE OUT OF P/N TIMER
E8DF  LE8DF:ldaa   L0136      ;LOAD VSS MALF TIMER
E8E2      brset   *L0014,#0x20,LE906 ;BRANCH IF NO VSS SIGNAL, ELSE
E8E6      ldab     *L00BD      ;LOAD FILTERED MPH
E8E8      cmpb     0x8E,x      ;9139 - 3 MPH
E8EA      bhi      LE90F      ;BRANCH IF > LOW LIMIT, ELSE
E8EC      brset   *L0015,#0x40,LE90F ;BRANCH IF TRANS RANGE SW MALF, ELSE
E8F0      ldab     L0138      ;LOAD P/N -> IN GEAR TRANSITION TIMER
E8F3      cmpb     0x90,x      ;913B - 4 SECONDS
E8F5      bls      LE90F      ;BRANCH IF TIME OUT OF P/N <= 4 SEC, ELSE
E8F7      ldab     *L0064      ;LOAD L0064
E8F9      bne      LE90F      ;BRANCH IF NOT = ZERO, ELSE
E8FB      ldab     *L00AC      ;LOAD RPM/25
E8FD      cmpb     0x8F,x      ;913A - 3000 RPM
E8FF      bls      LE90F      ;BRANCH IF <= 3000 RPM, ELSE
E901      inca      ;INCREMENT TIMER
E902      cmpa     0x91,x      ;913C - 2 SECONDS
E904      bcs      LE910      ;BRANCH IF < 2 SECONDS, ELSE
E906  LE906:ldaa   #0x20      ;MASK
E908      ldab     #0x01      ;OFFSET - 90AC
E90A      jsr      LF87D      ;GO UPDATE MALF FLAGS
E90D      bra      LE913

E90F  LE90F:clra      ;CLEAR A

```

```

E910    LE910:staa    L0136                                ;STORE VSS MALF TIMER

;-----
; DTC P0705 - TRANS RANGE SWITCH
;-----

E913    LE913:ldaa    L013C                                ;LOAD TIMER
E916          brclr   *L002B,#0x01,LE923                  ;BRANCH IF NOT PARK OR NEUTRAL, ELSE
E91A          ldab    *L0064                                ;LOAD
E91C          cmpb    L916D                                ;COMPARE LOW LIMIT
E91F          bls     LE926                                  ;BRANCH IF <= LOW LIMIT
E921          bra     LE92F

E923    LE923:clr     L0064                                ;CLEAR COUNTER
E926    LE926:brclr   *L008F,#0x04,LE938                  ;BRANCH IF VALID PRNDL COMBINATION, ELSE
E92A          inca    ;INCREMENT TIMER
E92B          cmpa    0xC3,x                                ;916E - 10 SECONDS
E92D          bcs     LE93C                                  ;BRANCH IF < 10 SECONDS, ELSE
E92F    LE92F:ldaa    #0x40                                ;MASK
E931          ldab    #0x02                                ;OFFSET - 90AD, L0015
E933          jsr     LF87D                                  ;GO UPDATE MALF FLAGS
E936          bra     LE93F

E938    LE938:clra    ;CLEAR A
E939          bclr    *L0015,#0x40                          ;CLEAR TRANS RANGE SWITCH MALF
E93C    LE93C:staa    L013C                                ;STORE TIMER

;-----
; P0755 - SHIFT SOLN B ERROR
;-----

E93F    LE93F:ldd     L0283                                ;N/V RATIO
E942          cpd     L9170                                  ;
E946          bcc     LE982                                  ;BRANCH IF >= CAL, ELSE
E948          brset   *L0077,#0x20,LE988                    ;BRANCH IF M755A (F31 SOLN B FAILED OFF)
E94C          ldaa    *L00E9                                ;TRANNNY GEAR BYTE
E94E          cmpa    #0x01                                  ;1ST GEAR?
E950          bne     LE982                                  ;BRANCH IF NOT 1ST GEAR, ELSE
E952          brset   *L0013,#0x01,LE982                    ;BRANCH IF TPS VOLTS HIGH, ELSE
E956          brset   *L0014,#0x80,LE982                    ;BRANCH IF TPS VOLTS LOW, ELSE
E95A          brset   *L0014,#0x20,LE982                    ;BRANCH IF "NO VSS SIGNAL" MALF, ELSE
E95E          brset   *L0015,#0x02,LE982                    ;BRANCH IF VSS INT SIGNAL MALF, ELSE
E962          ldaa    *L00AA                                ;%TPS
E964          cmpa    L9173                                  ;
E967          bls     LE982                                  ;
E969          ldaa    *L00BD                                ;FILTERED MPH
E96B          cmpa    L9172                                  ;5 MPH
E96E          bls     LE982                                  ;BRANCH IF <= 5 MPH, ELSE
E970          ldaa    L013E                                  ;MALF TIMER
E973          cmpa    L916F                                  ;TIME LIMIT
E976          bhi     LE97D                                  ;BRANCH IF > TIME LIMIT, ELSE
E978          inc     L013E                                  ;INCREMENT TIMER
E97B          bra     LE988                                  ;

E97D    LE97D:bset    *L0077,#0x20                          ;SET M755A (F31 SOLENOID B FAILED OFF)
E980          bra     LE988

E982    LE982:bclr    *L0077,#0x20                          ;CLEAR M755A (F31 SOLENOID B FAILED OFF)
E985          clr     L013E                                  ;CLEAR MALF TIMER
E988    LE988:ldaa    *L00E9                                ;TRANNNY GEAR BYTE
E98A          cmpa    #0x04                                  ;4TH GEAR ?
E98C          bne     LE9BF                                  ;BRANCH IF NOT 4TH GEAR, ELSE
E98E          ldaa    *L00AA                                ;%TPS
E990          cmpa    L9178                                  ;10 %TPS
E993          bls     LE9BF                                  ;BRANCH IF <= 10 %TPS,ELSE
E995          brclr   *L00A1,#0x18,LE9BF                    ;BRANCH IF PRNDL -> NOT DRV 3 OR 4, ELSE

```

```

E999      ldd      L0283                ;N/V RATIO
E99C      cpd      L9175                ;50
E9A0      bls      LE9BF                ;BRANCH IF <= 50, ELSE
E9A2      brset    *L0015,#0x40,LE9BF    ;BRANCH IF TRANS RANGE SW ERROR, ELSE
E9A6      ldab     *L00BA                ;LOAD LV8
E9A8      cmpb     L9177                ;80
E9AB      bhi      LE9BF                ;BRANCH IF LV8 > 80, ELSE
E9AD      ldaa     L013F                ;LOAD MALF TIMER
E9B0      cmpa     L9174                ;2 SECONDS
E9B3      bhi      LE9BA                ;BRANCH IF > 2 SECONDS, ELSE
E9B5      inc      L013F                ;INCREMENT MALF TIMER
E9B8      bra      LE9C5

E9BA      LE9BA:bset *L0077,#0x40        ;SET M755B (F31 SOLENOID B FAILED ON)
E9BD      bra      LE9C5

E9BF      LE9BF:bclr *L0077,#0x40        ;CLEAR M755B (F31 SOLENOID B FAILED ON)
E9C2      clr      L013F                ;CLEAR MALF TIMER
E9C5      LE9C5:brclr *L0077,#0x60,LE9D5 ;BRANCH IF NOT M755A OR M755B, ELSE
E9C9      ldaa     #0x04                ;MASK
E9CB      ldab     #0x02                ;OFFSET 90AD, L0015
E9CD      jsr      LF87D                ;GO UPDATE MALF FLAGS
E9D0      bra      LE9D5

E9D2      bclr     *L0015,#0x04        ;CLEAR SSB ERROR

;-----
; THIS MALF NOT ENABLED
;-----

E9D5      LE9D5:brclr 0x06,x,#0x01,LEA48 ;B1 90B1 - TIS CLEAR
E9D9      brset    *L0019,#0x01,LEA27    ;BRANCH IF MALF FLAG SET (NOT ENABLED)
E9DD      brset    *L0012,#0x20,LEA36    ;
E9E1      ldaa     *L00BD                ;FILTERED MPH
E9E3      cmpa     #0x3C                ;60 MPH
E9E5      bcs      LE9EA                ;BRANCH IF < 60 MPH, ELSE
E9E7      staa     L02FB                ;STORE MPH
E9EA      LE9EA:ldab L02FB                ;LOAD MPH
E9ED      beq      LEA36                ;BRANCH IF Z, ELSE
E9EF      cmpa     #0x0A                ;10 MPH
E9F1      bcs      LEA36                ;BRANCH IF < 10 MPH, ELSE
E9F3      ldaa     *L00AA                ;%TPS
E9F5      cmpa     L8CE2                ;
E9F8      bhi      LEA36                ;BRANCH IF > THRESHOLD, ELSE
E9FA      brset    *L009B,#0x02,LEA36    ;BRANCH IF DFCE ENABLED, ELSE
E9FE      brset    *L002A,#0x10,LEA36    ;EGR BIT STATUS FLAG
EA02      ldab     L02FC                ;COUNTER
EA05      cmpb     #0x06                ;6 ?
EA07      bcc      LEA33                ;BRANCH IF > 6, ELSE
EA09      ldaa     L0183                ;LOAD BAD CYL ID
EA0C      beq      LEA39                ;BRANCH IF Z, ELSE
EA0E      tst      L02FD                ;TEST
EA11      beq      LEA36                ;BRANCH IF L02FD = ZERO, ELSE
EA13      ldym     #0xFB8B                ;ADDRESS
EA17      tst      L8012                ;OPTION WORD - TIS CLEAR
EA1A      beq      LEA20                ;BRANCH IF Z, ELSE
EA1C      ldym     #0xFB8A                ;
EA20      LEA20:aby                ;ADD TO ADDRESS
EA22      cmpa     0x00,y                ;BAD CYL ID AND _____
EA25      beq      LEA2E                ;BRANCH IF THEY'RE EQUAL, ELSE
EA27      LEA27:ldaa #0x01                ;MASK
EA29      ldab     #0x06                ;OFFSET - 90B1 - NOT ENABLED
EA2B      jsr      LF87D                ;GO UPDATE MALF FLAGS
EA2E      LEA2E:inc  L02FC                ;COUNTER
EA31      bra      LEA36

```

```

EA33  LEA33:bset    *L0012,#0x20                ;SET BIT 5
EA36  LEA36:clra                    ;CLEAR A
EA37          bra    LEA45                    ;GO STORE L02FD

EA39  LEA39:ldab    L02FC                    ;LOAD COUNTER
EA3C          ld    #0xFB84                    ;TABLE ADDRESS
EA40          aby                    ;ADD COUNTER TO ADDRESS
EA42          ldaa   0x00,y                    ;GET CONTENTS OF THAT ADDRESS
EA45  LEA45:staa    L02FD                    ;STORE

;-----
; DTC P1629 - NO PASSKEY II SIGNAL ERROR
;-----

EA48  LEA48:ldaa    L8011                    ;OPTION WORD - TIS SET
EA4B          beq    LEA5D                    ;BRANCH IF Z, ELSE
EA4D          brset  *L0012,#0x02,LEA5A        ;BRANCH IF VATS OKAY, ELSE
EA51          ldaa   #0x02                    ;MASK
EA53          ldab   #0x07                    ;OFFSET - 90B2, L001A
EA55          jsr    LF87D                    ;GO UPDATE MALF FLAGS
EA58          bra    LEA5D

EA5A  LEA5A:bclr    *L001A,#0x02                ;CLEAR MALF FLAG

EA5D  LEA5D:ldaa    L800C                    ;OPTION WORD - TIS SET
EA60          bne    LEA77                    ;BRANCH IF NOT Z, ELSE
EA62          brclr  *L00D9,#0x40,LEAA2        ;BRANCH IF SMC NOT ENGAGED, ELSE
EA66          brset  *L0014,#0x20,LEAA2        ;BRANCH IF NO VSS SIGNAL, ELSE
EA6A          brset  *L0015,#0x02,LEAA2        ;BRANCH IF VSS INT SIGNAL MALF, ELSE
EA6E          ldaa   *L00BD                    ;FILTERED MPH
EA70          cmpa   L9148                    ;7 MPH
EA73          bcs    LEA83                    ;BRANCH IF < 7 MPH, ELSE
EA75          bra    LEAA2

EA77  LEA77:brclr   *L0077,#0x04,LEAAA        ;BRANCH IF SMC NOT PRESENT, ELSE
EA7B          brclr  *L00D9,#0x40,LEAA2        ;BRANCH IF SMC NOT ENGAGED, ELSE
EA7F          brclr  *L0099,#0x10,LEAA2        ;BRANCH IF CRUISE CONTROL ALLOWED, ELSE
EA83  LEA83:ldd     L0149                    ;LOAD TIMER
EA86          cpd    L9146                    ;10 - TIME LIMIT
EA8A          bls    LEA91                    ;BRANCH IF < 10, ELSE
EA8C          bclr   *L0077,#0x04                ;CLEAR SMC PRESENT ON VEHICLE
EA8F          bra    LEA99

EA91  LEA91:addd    #0x0001                    ;INCREMENT
EA94          std    L0149                    ;STORE TIMER
EA97          bra    LEAAA

EA99  LEA99:ldaa    #0x04                    ;MASK
EA9B          ldab   #0x07                    ;OFFSET 90B2, L001A
EA9D          jsr    LF87D                    ;GO UPDATE MALF FLAGS
EAA0          bra    LEAAA

EAA2  LEAA2:clra                    ;CLEAR TIMER
EAA3          clrb                    ;
EAA4          std    L0149                    ;TIMER
EAA7          bclr   *L001A,#0x04                ;CLEAR MALF FLAG
EAAA  LEAAA:brclr   0x07,x,#0x01,LEB19        ;90B2 - TIS CLEAR - RTS
EAAE          ldaa   L8010                    ;OPTION WORD - TRANS TYPE - TIS CLEAR
EAB1          bpl    LEB19                    ;BRANCH IF AUTO TRANS - RTS, ELSE
EAB3          brset  *L0015,#0x02,LEB19        ;BRANCH IF VSS INT SIGNAL MALF, ELSE
EAB7          brset  *L0014,#0x20,LEB19        ;BRANCH IF NO VSS SIGNAL, ELSE
EABB          brset  *L00A5,#0x02,LEB19        ;
EABF          brset  *L001A,#0x01,LEB03        ;BRANCH IF MALF (NOT ENABLED), ELSE
EAC3          brset  *L00A5,#0x10,LEADA

```

```

EAC7      bclr    *L00A5,#0x01
EACA      brclr   *L00A2,#0x01,LEAD1
EACE      bset    *L00A5,#0x01
EAD1      LEAD1:ldaa *L00BD                      ;FILTERED MPH
EAD3      bne     LEB19                          ;BRANCH IF NOT Z, ELSE
EAD5      bset    *L00A5,#0x10                  ;SET BIT 4
EAD8      bra     LEB19                          ;

EADA      LEADA:ldaa *L00A5                      ;
EADC      adda    *L00A2
EADE      rora
EADF      bcc     LEAE4                          ;DIV BY 2
EAE1      bset    *L00A5,#0x04                  ;BRANCH IF NO OVERFLOW, ELSE
EAE4      LEAE4:brset *L00A5,#0x08,LEAF3          ;SET BIT 2
EAE8      ldaa    *L00BD                          ;BRANCH IF BIT 3, ELSE
EAEA      cmpa    0xAA,x                        ;FILTERED MPH
EAE4      bcs     LEB19                          ;9155
EAE6      bset    *L00A5,#0x08
EAF1      bra     LEB19

EAF3      LEAF3:ldaa *L00BD                      ;FILTERED MPH
EAF5      bne     LEB19
EAF7      ldaa    L014B
EAF8      brset   *L00A5,#0x04,LEB0C
EAFE      inca
EAF9      cmpa    0xAB,x                        ;9156
EB01      bls     LEB13
EB03      LEB03:ldaa #0x01                      ;MASK
EB05      ldab    #0x07                          ;OFFSET 90B2 - NOT ENABLED
EB07      jsr     LF87D                          ;GO UPDATE MALF FLAGS
EB0A      bra     LEB19

EB0C      LEB0C:bset *L00A5,#0x02                ;
EB0F      bclr    *L001A,#0x01                  ;CLEAR MALF FLAG (NOT ENABLED)
EB12      clra
EB13      LEB13:staa L014B                      ;
EB16      bclr    *L00A5,#0x1C                  ;
EB19      LEB19:rts

;-----
; DTC P0101 - MAF SENSOR MALF CHECK
;-----
EB1A      LEB1A:ldx  #0x90AB                      ;INDEX
EB1D      ldaa    L013D                          ;TIMER
EB20      brclr   *L0091,#0x10,LEB32            ;BRANCH IF MAF SIGNAL OKAY, ELSE
EB24      inca
EB25      cmpa    0x11,x                        ;INCREMENT TIMER
EB27      bcs     LEB39                          ;90BC - 6.4 SECONDS
EB29      ldaa    #0x10                          ;BRANCH IF < 6.4 SECONDS, ELSE
EB2B      ldab    #0x02                          ;MASK
EB2D      jsr     LF87D                          ;OFFSET - 90AD, L0015
EB30      bra     LEB3C                          ;GO UPDATE MALF FLAGS

EB32      LEB32:clra                          ;CLEAR TIMER
EB33      bclr    *L0015,#0x10                  ;CLEAR MAF MALF FLAG
EB36      bclr    *L008F,#0x02                  ;CLEAR BIT 1
EB39      LEB39:staa L013D                      ;STORE TIMER

;-----
; DTC P0703 - TCC BRAKE SWITCH ERROR
;-----
EB3C      LEB3C:brset *L0014,#0x20,LEB8C          ;BRANCH IF NO VSS SIGNAL, ELSE
EB40      ldab    *L008E                          ;FMD INPUT STATUS WORD
EB42      andb    #0x40                          ;CLEAR BITS EXCEPT BRAKE APPLIED/RELEASED

```

```

EB44          tba                ;COPY TO A REG
EB45          addb    *L0092      ;ADD L0092
EB47          bclr    *L0092,#0x40 ;CLEAR BIT 6
EB4A          oraa     *L0092      ;SET BITS
EB4C          staa     *L0092      ;STORE
EB4E          bitb     #0x40        ;BIT 6 SET?
EB50          bne      LEB8C        ;BRANCH IF BIT 6 SET, ELSE
EB52          brset    *L0015,#0x01,LEB83 ;BRANCH IF TCC BRAKE SW MALF, ELSE
EB56          ldab     *L00BD        ;FILTERED MPH
EB58          brset    *L0092,#0x01,LEB73 ;BRANCH IF 2ND GEAR START REQUESTED, ELSE
EB5C          cmpb     0xAC,x        ;9157 - 35 MPH
EB5E          bhi      LEB63        ;BRANCH IF > 35 MPH, ELSE
EB60          clra                ;CLEAR A
EB61          bra      LEB67

EB63          LEB63:ldaa    L01AB        ;MALF TIMER
EB66          inca                ;INCREMENT TIMER
EB67          LEB67:staa    L01AB        ;STORE TIMER
EB6A          cmpa     0xAE,x        ;9159 - 10 SECONDS
EB6C          bls      LEB99        ;BRANCH IF <= 10 SECONDS, ELSE
EB6E          bset     *L0092,#0x01    ;SET 2ND GEAR START REQUESTED
EB71          bra      LEB99

EB73          LEB73:bne     LEB99        ;BRANCH IF NOT Z, ELSE
EB75          bclr    *L0092,#0x01    ;CLEAR BIT 0
EB78          clr      L01AB        ;CLEAR MALF TIMER
EB7B          ldaa     L0140
EB7E          inca
EB7F          cmpa     0xAD,x        ;9158
EB81          bls      LEB96        ;BRANCH IF <= 4, ELSE
EB83          LEB83:ldaa    #0x01        ;MASK
EB85          ldab     #0x02        ;OFFSET 90AD, L0015
EB87          jsr      LF87D        ;GO UPDATE MALF FLAGS
EB8A          bra      LEB99

EB8C          LEB8C:clra                ;CLEAR MALF TIMER
EB8D          bclr    *L0015,#0x01    ;CLEAR TCC BRAKE SW ERROR
EB90          bclr    *L0092,#0x01    ;CLEAR BIT 0
EB93          staa     L01AB        ;STORE MALF TIMER
EB96          LEB96:staa    L0140        ;STORE MALF TIMER

;-----
; DTC P0740 - TCC MALF CHECK
;-----
EB99          LEB99:ldaa    L800F        ;OPTION WORD - TIS SET
EB9C          beq      LEBEF        ;BRANCH IF Z - SKIP MALF CHECK, ELSE
EB9E          brclr    *L0096,#0x08,LEBA6 ;BRANCH IF TCC NOT LOCKED, ELSE
EBA2          brset    *L0090,#0x20,LEBE9 ;
EBA6          LEBA6:brset    *L0016,#0x80,LEBE0 ;BRANCH IF TCC ERROR, ELSE
EBAE          brclr    *L00A3,#0x02,LEBE9 ;BRANCH IF NOT ON MODE, ELSE
EBAE          brclr    *L008E,#0x08,LEBB8 ;BRANCH IF 4TH GEAR, ELSE
EBB2          brclr    *L008E,#0x04,LEBB8 ;BRANCH IF 3RD GEAR, ELSE
EBB6          bra      LEBE9

EBB8          LEBB8:ldd     L0283        ;N/V RATIO
EBBB          cpd      L9162        ;48
EBBF          bhi      LEBD3        ;BRANCH IF > 48, ELSE
EBC1          cpd      L9164        ;45
EBC5          bcc      LEBE9        ;BRANCH IF > 45, ELSE
EBC7          cpd      L9166        ;35
EBCB          bhi      LEBD3        ;BRANCH IF > 35, ELSE
EBCD          cpd      L9168        ;31
EBD1          bcc      LEBE9        ;BRANCH IF > 31, ELSE
EBD3          LEBD3:ldaa    L0141        ;LOAD MALF TIMER

```

```

EBD6      cmpa    L916A                ;10 SECONDS
EBD9      bhi     LEBE0                ;BRANCH IF > 10 SECONDS, ELSE
EBDB      inc     L0141                ;INCREMENT MALF TIMER
EBDE      bra     LEBEF                ;

EBE0      LEBE0:ldaa    #0x80                ;MASK
EBE2      ldab    #0x03                ;OFFSET 90AE, L0016
EBE4      jsr     LF87D                ;GO UPDATE MALF FLAGS
EBE7      bra     LEBEF

EBE9      LEBE9:bclr    *L0016,#0x80        ;CLEAR CAMSHAFT SIGNAL MALF
EBEC      clr     L0141                ;CLEAR MALF TIMER

;-----
; DTC P0342 - CAMSHAFT SENSOR MALF CHECK
;-----
EBEF      LEBEF:ldaa    L0142                ;LOAD MALF TIMER
EBF2      brset   *L0093,#0x40,LEC08        ;BRANCH IF CAMSHAFT PULSE OCCURED, ELSE
EBF6      brset   *L0016,#0x40,LEBFF        ;BRANCH IF CAMSHAFT SENSOR MALF, ELSE
EBFA      inca    ;INCREMENT MALF TIMER
EBFB      cmpa    0x6D,x                ;9118 - 5 SECONDS
EBFD      bcs     LEC0C                ;BRANCH IF < 5 SECONDS, ELSE
EBFF      LEBFF:ldaa    #0x40                ;MASK
EC01      ldab    #0x03                ;OFFSET 90AE, L0016
EC03      jsr     LF87D                ;GO UPDATE MALF FLAGS
EC06      bra     LEC0F

EC08      LEC08:clra                ;CLEAR MALF TIMER
EC09      bclr    *L0016,#0x40        ;CLEAR CAMSHAFT SENSOR MALF
EC0C      LEC0C:staa    L0142                ;STORE MALF TIMER
EC0F      LEC0F:bclr    *L0093,#0xE0        ;CLEAR CAM PULSE SEEN BITS

;-----
; DTC P1361 - IGNITION CONTROL BYPASS ERROR CHECK
;-----
EC12      brset   *L0011,#0x10,LEC50        ;MINOR LOOP COUNTER - 100 MSEC
EC16      brset   *L0090,#0x40,LEC50        ;
EC1A      brset   *L0098,#0x08,LEC22        ;BRANCH IF ? REF PULSE OCCURED, ELSE
EC1E      brclr   *L0086,#0x40,LEC50        ;BRANCH IF M42A FAILED, ELSE
EC22      LEC22:ldy     L3FC8                ;EST VOLTAGE
EC26      brset   *L0017,#0x10,LEC40        ;BRANCH IF IGN CNTL ERROR, ELSE
EC2A      ldaa    *L00AB                ;LOAD NLRPMX
EC2C      cmpa    L911C                ;400 RPM
EC2F      bls     LEC49                ;BRANCH IF < 400 RPM, ELSE
EC31      cpy     L029A                ;COMPARE PREV EST VOLTAGE
EC35      bne     LEC49                ;BRANCH IF THEY'RE NOT EQUAL, ELSE
EC37      brset   *L008F,#0x01,LEC40        ;BRANCH IF BIT 0, ELSE
EC3B      bset    *L008F,#0x01        ;SET BIT 0
EC3E      bra     LEC4C                ;

EC40      LEC40:ldaa    #0x10                ;MASK
EC42      ldab    #0x04                ;OFFSET 90AF, L0017
EC44      jsr     LF87D                ;GO UPDATE MALF FLAGS
EC47      bra     LEC4C

EC49      LEC49:bclr    *L008F,#0x01        ;CLEAR BIT 0
EC4C      LEC4C:sty     L029A                ;STORE FOR NEXT PASS

;-----
; O2 SENSOR MALF CHECK
;-----
EC50      LEC50:ldaa    L0144                ;MALF TIMER
EC53      ldab    *L00A9                ;MINOR LOOP O2 A/D COUNTS (mV)
EC55      cmpb    0x5A,x                ;9105

```

```

EC57      bcc      LEC77
EC59      brset    *L0016,#0x08,LEC6E
EC5D      brclr    *L009C,#0x80,LEC77
EC61      brset    *L0091,#0x04,LEC7E
EC65      brclr    *L008F,#0x08,LEC7E
EC69      inca
EC6A      cmpa     0x5B,x
EC6C      bls      LEC7B
EC6E      LEC6E:ldaa #0x08
EC70      ldab     #0x03
EC72      jsr      LF87D
EC75      bra      LEC7E

;BRANCH IF > 248 mV, ELSE
;BRANCH IF O2 SENSOR LEAN, ELSE
;BRANCH IF NOT CLOSED LOOP, ELSE
;BRANCH IF INT RESET, ELSE
;
;INCREMENT TIMER
;9106 - 25.4 SECONDS
;BRANCH IF <= 25.4 SECONDS, ELSE
;MASK
;OFFSET 90AE, L0016
;GO UPDATE MALF FLAGS

EC77      LEC77:clra
EC78      bclr     *L0016,#0x08
EC7B      LEC7B:staa L0144
EC7E      LEC7E:ldaa L0145
EC81      brset    *L0013,#0x01,LECB7
EC85      brset    *L0014,#0x80,LECB7
EC89      ldab     *L00A9
EC8B      cmpb     0x5C,x
EC8D      bls      LECB7
EC8F      brset    *L0016,#0x04,LECAE
EC93      ldab     *L00AA
EC95      cmpb     0x5E,x
EC97      bcc      LECB7
EC99      cmpb     0x5F,x
EC9B      bls      LECB7
EC9D      brclr    *L009C,#0x80,LECB7
ECA1      brset    *L0091,#0x04,LECBE
ECA5      brclr    *L008F,#0x08,LECBE
ECA9      inca
ECAA      cmpa     0x5D,x
ECAC      bls      LECBB
ECAE      LECAE:ldaa #0x04
ECB0      ldab     #0x03
ECB2      jsr      LF87D
ECB5      bra      LECBE

;CLEAR TIMER
;CLEAR O2 SENSOR LEAN MALF
;STORE MALF TIMER
;LOAD MALF TIMER
;BRANCH IF TPS VOLTS HIGH, ELSE
;BRANCH IF TPS VOLTS LOW, ELSE
;MINOR LOOP O2 A/D COUNTS (mV)
;9107 - 754 mV
;BRANCH IF < 754 mV, ELSE
;BRANCH IF O2 SENSOR RICH, ELSE
;%TPS
;9109
;BRANCH IF > 40 %TPS, ELSE
;910A
;BRANCH IF <= 5.85 %TPS, ELSE
;BRANCH IF NOT CLOSED LOOP, ELSE
;BRANCH IF INTEGRATOR RESET, ELSE
;BRANCH IF NOT BIT 3, ELSE
;INCREMENT TIMER
;9108
;BRANCH IF <= 12 SECONDS, ELSE
;MASK
;OFFSET 90AE, L0016
;GO UPDATE MALF FLAGS
;

ECB7      LECB7:clra
ECB8      bclr     *L0016,#0x04
ECBB      LECBB:staa L0145

;CLEAR MALF TIMER
;CLEAR O2 SENSOR RICH MALF
;STORE MALF TIMER

;-----
; THIS MALF NOT ENABLED
;-----

ECBE      LECBE:brset *L0018,#0x20,LED10
ECC2      ldd      L02E1
ECC5      cpd      L9140
ECC9      bls      LECD0
ECCB      clr      L0071
ECCE      bra      LED29

;BRANCH IF MALF FLAG SET (NOT ENABLED)
;
;CLEAR STARTUP COUNTER

ECD0      LECD0:brclr *L0090,#0x04,LECD4
ECD4      addd     #0x0001
ECD7      std      L02E1
ECDA      LECDA:brset *L008E,#0x80,LECF4
ECDE      brset    *L0090,#0x08,LECE8
ECE2      bset     *L0090,#0x04
ECE5      inc      L02E5
ECE8      LECE8:ldd  L02E3
ECED      cpd      L9143
ECEB      bcs      LECF4
ECE5      bcs      LECF4
ECF1      clra

```

```

ECF2          bra      LED32

ECF4  LECF4: addd      #0x0001
ECF7          std      L02E3
ECFA          bset     *L0090, #0x08
ECFD          bra      LED35

ECFF  LECFF: ldd       #0x0000
ED02          std      L02E3
ED05          bclr     *L0090, #0x08
ED08          ldaa     L02E5
ED0B          cmpa     L9142
ED0E          bcs      LED35
ED10  LED10: brset     *L0018, #0x20, LED1C      ;BRANCH IF MALF SET, ELSE
ED14          inc      L0071                      ;INCREMENT ERROR COUNTER
ED17          bne      LED1C                      ;BRANCH IF NOT Z, ELSE
ED19          dec      L0071                      ;DECREMENT COUNTER
ED1C  LED1C: ldaa      #0x20                      ;MASK
ED1E          ldab      #0x05                      ;OFFSET 90B0 L0018 - NOT ENALBLED
ED20          jsr      LF87D                      ;GO UPDATE MALF FLAGS
ED23          ldd       #0x0000
ED26          std      L02E1
ED29  LED29: bclr     *L0090, #0x0C
ED2C          ldd       #0x0000
ED2F          std      L02E3
ED32  LED32: staa     L02E5

;-----
; EGR MALFS
;-----

ED35  LED35: ldx       #0x90AB                      ;INDEX
ED38          ldab      *L002A                      ;EGR BIT STATUS WORD
ED3A          cmpb      #0xE0                      ;%11100000
ED3C          bcs      LED5A                      ;BRANCH IF < 224, ELSE
ED3E          andb      #0x1F                      ;%00011111
ED40          ldaa      *L0029                      ;COUNTER
ED42          inca      L0029                      ;INCREMENT COUNTER
ED43          cmpa     L9132                      ;5
ED46          bcs      LED56                      ;BRANCH IF < 5, ELSE
ED48          orab      #0x80                      ;SET BIT 7
ED4A          cmpa     L9133                      ;8
ED4D          bcs      LED56                      ;BRANCH IF < 8, ELSE
ED4F          orab      #0x40                      ;SET BIT 6
ED51          cmpa     L9134                      ;12
ED54          bcc      LED67                      ;BRANCH IF > 12, ELSE
ED56  LED56: staa      *L0029                      ;STORE COUNTER
ED58          stab      *L002A                      ;STORE STATUS FLAG
ED5A  LED5A: ldaa      *L0017                      ;MALF FLAG
ED5C          anda      #0x0E                      ;CLEAR ALL EXCEPT BITS 1,2,3
ED5E          anda      0x0C, x                      ;90B7 - %00111111
ED60          beq      LED9E                      ;BRANCH IF Z, ELSE
ED62          bset     *L00A0, #0x30                ;SET TIMER TO 48d
ED65          bra      LED9E                      ;

ED67  LED67: bne      LED6B                      ;BRANCH IF L0029 NOT = 12, ELSE
ED69          staa      *L0029                      ;STORE COUNTER
ED6B  LED6B: bclr     *L0017, #0x02                ;CLEAR MALF FLAG
ED6E          ldaa      *L0026                      ;
ED70          cmpa     L912F                      ;3
ED73          bls      LED7C                      ;BRANCH IF <= 3, ELSE
ED75          ldaa      #0x02                      ;MASK
ED77          ldab      #0x04                      ;OFFSET - 90AF, L0017
ED79          jsr      LF87D                      ;GO UPDATE MALF FLAGS
ED7C  LED7C: bclr     *L0017, #0x04                ;CLEAR MALF FLAG

```

```

ED7F      ldaa    *L0027      ;
ED81      cmpa    L9130      ;6
ED84      bls     LED8D      ;BRANCH IF <= 6, ELSE
ED86      ldaa    #0x04      ;MASK
ED88      ldab    #0x04      ;OFFSET - 90AF, L0017
ED8A      jsr     LF87D      ;GO UPDATE MALF FLAGS
ED8D      LED8D:bclr  *L0017,#0x08 ;CLEAR MALF
ED90      ldaa    *L0028      ;
ED92      cmpa    L9131      ;9
ED95      bls     LED9E      ;BRANCH IF <= 9, ELSE
ED97      ldaa    #0x08      ;MASK
ED99      ldab    #0x04      ;OFFSET - 90AF, L0017
ED9B      jsr     LF87D      ;GO UPDATE MALF FLAGS
ED9E      LED9E:rts

;-----
; DIAGNOSTICS
; BRANCH HERE FROM SEGMENT TABLE
;-----

ED9F      brclr   *L0011,#0xF0,LEDA8 ;MINOR LOOP COUNTER - 1 SECOND
EDA3      bclr    *L008F,#0x18      ;
EDA6      bra     LEDB2

EDA8      LEDA8:bset  *L008F,#0x08      ;
EDAB      brset   *L0033,#0x01,LEDB2  ;RUNTIME LSB - 2 SECONDS
EDAF      bset    *L008F,#0x10      ;
EDB2      LEDB2:brset *L0017,#0x20,LEDCB ;BRANCH IF PROM ERROR, ELSE
EDB6      brclr   *L0088,#0x20,LEDD2  ;BRANCH IF PROM ERROR NOT SET, ELSE
EDBA      brclr   *L0088,#0x40,LEDCD  ;BRANCH IF NOT PROM ERROR 1ST PASS, ELSE
EDBE      bset    *L0017,#0x20      ;SET PROM ERROR
EDC1      brset   *L001F,#0x20,LEDCB  ;BRANCH IF BIT 5, ELSE
EDC5      bset    *L001F,#0x20      ;SET BIT 5
EDC8      jsr     LF7EC      ;GO ADD UP MALF WORDS
EDCB      LEDCB:bra  LEE1F      ;GO TURN ON SES LIGHT

EDCD      LEDCD:bset  *L0088,#0x40      ;SET PROM ERROR 1ST PASS
EDD0      bra     LEDD5      ;

EDD2      LEDD2:bclr  *L0088,#0x40      ;CLEAR PROM ERROR 1ST PASS
EDD5      LEDD5:bclr  *L0088,#0x20      ;CLEAR PROM ERROR
EDD8      brset   *L0095,#0x10,LEE1F   ;BRANCH IF BATT VOLTS < 4.0, ELSE
EDDC      brclr   *L0086,#0x80,LEDE3   ;BRANCH IF ENGINE RUNNING, ELSE
EDE0      jsr     LEB1A      ;MALF CHECK
EDE3      LEDE3:brset *L0086,#0x80,LEDF1 ;BRANCH IF ENGINE RUNNING, ELSE
EDE7      brclr   *L0012,#0x08,LEE1F   ;BRANCH IF BIT 3 CLEAR, ELSE
EDEB      brset   *L0077,#0x80,LEE1F   ;BRANCH IF SES LIGHT ON AT STALL
EDEE      bra     LEE22      ;RETURN

EDF1      LEDF1:brclr  *L008F,#0x08,LEE13
EDF5      ldx     *L0032      ;RUN TIME
EDF7      cpx     #0x000A      ;10 SECONDS
EDFA      bne     LEE13      ;BRANCH IF NOT = 10 SECONDS, ELSE
EDFC      ldaa    *L0023      ;MALF TIMER
EDFE      cmpa    L90BB      ;50
EE01      beq     LEE0A      ;BRANCH IF 500 SECONDS, ELSE
EE03      bhi     LEE13      ;BRANCH IF > 500 SECONDS, ELSE
EE05      inca    ;INCREMENT TIMER
EE06      staa    *L0023      ;STORE MALF TIMER
EE08      bra     LEE13      ;

EE0A      LEE0A:jsr    LF84C      ;
EE0D      ldaa    L90BB      ;50 SECONDS
EE10      inca    ;INCREMENT TIMER
EE11      staa    *L0023      ;STORE MALF TIMER

```

```

EE13 LEE13:ldaa *L00A0 ;LOAD TIMER
EE15 bne LEE1C ;BRANCH IF NOT Z, ELSE
EE17 bclr *L0091,#0x01 ;TURN SES LIGHT OFF
EE1A bra LEE22 ;RETURN

EE1C LEE1C:deca ;DECREMENT TIMER
EE1D staa *L00A0 ;STORE
EE1F LEE1F:bset *L0091,#0x01 ;TURN SES LIGHT ON
EE22 LEE22:rts

;-----
; JUMP HERE INDIRECTLY FROM SEGMENT TABLE - 100 MSEC ROUTINE
; TCC PWM ROUTINE
; 1313 BYTES LONG
;-----

EE23 LEE23:ldaa L8010 ;OPTION FLAG - TRANS TYPE - TIS CLEAR
EE26 bpl LEE2B ;BRANCH IF AUTO TRANS, ELSE
EE28 jmp LF343 ;SKIP TCC PWM STUFF - RETURN

EE2B LEE2B:ldaa L0199
EE2E brset *L00A2,#0x08,LEE37
EE32 ldab L0198
EE35 bne LEE71
EE37 LEE37:ldab L019E
EE3A incb
EE3B cmpb L95A4
EE3E bls LEE46
EE40 clrb
EE41 suba L95A6
EE44 bcs LEE5B
EE46 LEE46:brset *L001A,#0x04,LEE4E ;BRANCH IF MALF (NOT ENABLED), ELSE
EE4A brset *L00D9,#0x40,LEE53 ;BRANCH IF SMC ENGAGED, ELSE
EE4E LEE4E:cmpa L959F
EE51 bra LEE59

EE53 LEE53:pshb
EE54 ldab L95A0
EE57 cba
EE58 pulb
EE59 LEE59:bcc LEE6B
EE5B LEE5B:brset *L001A,#0x04,LEE63 ;BRANCH IF MALF (NOT ENABLED), ELSE
EE5F brset *L00D9,#0x40,LEE68 ;BRANCH IF SMC ENGAGED, ELSE
EE63 LEE63:ldaa L959F
EE66 bra LEE6B

EE68 LEE68:ldaa L95A0
EE6B LEE6B:staa L0199
EE6E stab L019E
EE71 LEE71:cmpa L0198
EE74 bcc LEE79
EE76 staa L0198
EE79 LEE79:brclr *L0086,#0x80,LEE81 ;BRANCH IF ENGINE NOT RUNNING, ELSE
EE7D brset *L008E,#0x0C,LEE9E ;BRANCH IF NOT 3RD OR 4TH GEARS, ELSE
EE81 LEE81:ldaa #0x40
EE83 jsr LD933 ;TEST MODE 4 CONTROL WORDS
EE86 beq LEE9E
EE88 bpl LEE92
EE8A bclr *L00A3,#0x0D ;CLEAR APPLY, RELEASE AND OFF MODES
EE8D bset *L00A3,#0x02 ;SET ON MODE
EE90 bra LEE98

EE92 LEE92:bclr *L00A3,#0x07 ;CLEAR APPLY, ON AND RELEASE MODES
EE95 jmp LEFFC

```

```

EE98 LEE98:clr      L0198
EE9B      jmp      LF009

EE9E LEE9E:brclr   *L0013,#0x10,LEEA5      ;BRANCH IF LOW SYSTEM VOLTS NOT SET, ELSE
EEA2 LEEA2:jmp     LEF53

;-----
; TEST MALF FLAGS
;-----

EEA5 LEEA5:brset   *L0013,#0x01,LEEA2      ;TPS VOLTS HIGH
EEA9      brset   *L0014,#0x80,LEEA2      ;TPS VOLTS LOW
EEAD      brset   *L0014,#0x20,LEEA2      ;NO VSS SIGNAL MALF
EEB1      brset   *L0015,#0x02,LEEA2      ;VSS INT SIGNAL MALF, ELSE
EEB5      brset   *L0015,#0x04,LEEA2      ;SHIFT SOLN B ERROR
EEB9      brset   *L0015,#0x01,LEEA2      ;TCC BRAKE SW ERROR
EEBD      brset   *L0016,#0x80,LEEA2      ;TCC ERROR
EEC1      brset   *L008E,#0x40,LEEA2      ;BRANCH IF BRAKE APPLIED, ELSE
EEC5      brset   *L0091,#0x20,LEEA2      ;
EEC9      brset   *L008E,#0x0E,LEEA2      ;BRANCH IF NOT 2ND, 3RD OR 4TH GEARS, ELSE
EECD      brclr   *L0090,#0x20,LEED8      ;
EED1      brset   *L008E,#0x08,LEEF5      ;BRANCH IF NOT 4TH GEAR, ELSE
EED5      jmp     LF009

EED8 LEED8:brset   *L008E,#0x04,LEF53      ;BRANCH IF NOT 3RD GEAR, ELSE
EEDC      ldaa    L01CA                    ;LOAD F31/TCCPWM TIMER
EEDF      cmpa    L95A1                    ;
EEE2      bhi     LEEE7                    ;BRANCH IF > CAL, ELSE
EEE4      jmp     LEFFF                    ;

EEE7 LEEE7:ldaa    L95B6
EEEE      brclr   *L0096,#0x08,LEEF1      ;BRANCH IF TCC NOT LOCKED, ELSE
EEEE      ldaa    L95B5
EEF1 LEEF1:cmpa    *L00A7                  ;FILTERED CTS
EEF3      bhi     LEF53
EEF5 LEEF5:ldx     #0x95A9                  ;INDEX
EEF8      ldab    #0x06                    ;OFFSET
EEFA      brset   *L001A,#0x04,LEF17      ;BRANCH IF MALF (NOT ENABLED), ELSE
EEFE      brclr   *L00D9,#0x40,LEF17      ;BRANCH IF SMC NOT ENGAGED, ELSE
EF02      ldaa    *L00AA                  ;%TPS
EF04      cmpa    L95A8                    ;
EF07      bcc     LEF16
EF09      ldaa    *L00F5                  ;%TPS
EF0B      suba    L01BE                    ;PREV %TPS
EF0E      bcc     LEF16
EF10      nega
EF11      cmpa    L95A7
EF14      bhi     LEF53
EF16 LEF16:abx     ;95AF
EF17 LEF17:brclr   *L008E,#0x08,LEF23      ;BRANCH IF 4TH GEAR, ELSE
EF1B      inx
EF1C      inx
EF1D      brclr   *L008E,#0x04,LEF23      ;BRANCH IF 3RD GEAR, ELSE
EF21      inx
EF22      inx
EF23 LEF23:ldaa    0x00,x                  ;95A9 OR
                                           ;95AB OR
                                           ;95AD OR
                                           ;95AF OR
                                           ;95B1 OR
                                           ;95B3
EF25      brset   *L0096,#0x08,LEF2B      ;BRANCH IF TCC LOCKED, ELSE
EF29      ldaa    0x01,x                  ;95AA OR
                                           ;95AC OR
                                           ;95AE OR

```

```

;95B0 OR
;95B2 OR
;95B4

EF2B LEF2B: cmpa *L00F5 ;%TPS
EF2D bls LEF56 ;BRANCH IF <= CAL, ELSE
EF2F bset *L00A2,#0x08 ;SET BIT 3
EF32 brclr *L0096,#0x08,LEF4E ;BRANCH IF TCC NOT LOCKED, ELSE
EF36 ldd *L0032 ;RUN TIME
EF38 subd L019C ;
EF3B cpd L95A2 ;
EF3F bhi LEF4E
EF41 ldaa L95A5
EF44 adda L0199
EF47 bcc LEF4B
EF49 ldaa #0xFF
EF4B LEF4B: staa L0199
EF4E LEF4E: ldd *L0032 ;RUN TIME
EF50 std L019C ;SAVE FOR NEXT PASS
EF53 LEF53: jmp LEFFC

EF56 LEF56: bclr *L00A2,#0x08
EF59 ldaa L95B7
EF5C brclr *L00A3,#0x08,LEF63 ;BRANCH IF NOT OFF MODE, ELSE
EF60 ldaa L95B8
EF63 LEF63: cmpa L01C7 ;FILTERED TCC SLIP RPM
EF66 bhi LEF98
EF68 brclr *L0090,#0x20,LEF85
EF6C ldaa L942F
EF6F ldx #0x95B9 ;INDEX
EF72 ldab #0x11 ;OFFSET
EF74 brset *L008E,#0x04,LEF7E ;BRANCH IF NOT 3RD GEAR, ELSE
EF78 ldaa L9430
EF7B ldx #0x95DB ;INDEX
EF7E LEF7E: brclr *L0096,#0x08,LEFD3 ;BRANCH IF TCC NOT LOCKED, ELSE
EF82 abx
EF83 bra LEFD3

EF85 LEF85: ldaa *L00F5 ;%TPS
EF87 suba L01BE ;PREV %TPS
EF8A bcc LEF9B
EF8C nega
EF8D cmpa L959C ;
EF90 bls LEFA9 ;
EF92 ldab L959E ;
EF95 stab L0199
EF98 LEF98: jmp LEFFC

EF9B LEF9B: ldab L959D ;
EF9E cmpa L959B ;
EFA1 bls LEFA9 ;BRANCH IF <= CAL, ELSE
EFA3 bset *L00A3,#0x10 ;SET POSITIVE DELTA TPS RELEASE OF TCC
EFA6 jmp LF002

EFA9 LEFA9: ldx #0x9685 ;INDEX
EFAC ldab #0x11 ;OFFSET
EFAE brclr *L008C,#0x01,LEFB7 ;BRANCH IF 2ND GEAR START REQ, ELSE
EFB2 ldx #0x95FD ;INDEX
EFB5 ldab #0x22 ;OFFSET
EFB7 LEFB7: ldaa L9430 ;
EFBA brset *L008E,#0x08,LEFC2 ;BRANCH IF NOT 4TH GEAR, ELSE
EFBE ldaa L9431 ;
EFC1 abx ;
EFC2 LEFC2: brclr *L0096,#0x08,LEFC8 ;BRANCH IF TCC NOT LOCKED, ELSE
EFC6 lsib ;

```

| | | | |
|------|-------------|--------------------|---------------------------------------|
| EFC7 | abx | | |
| EFC8 | LEFC8:brset | *L001A,#0x04,LEFD3 | ;BRANCH IF MALF (NOT ENABLED), ELSE |
| EFCC | brclr | *L00D9,#0x40,LEFD3 | ;BRANCH IF SMC NOT ENGAGED, ELSE |
| EFD0 | ldab | #0x11 | ;OFFSET |
| EFD2 | abx | | |
| EFD3 | LEFD3:staa | L0301 | |
| EFD6 | bset | *L0097,#0x10 | |
| EFD9 | jsr | LF81F | ;DO LOOKUP |
| EFDC | bclr | *L0097,#0x10 | |
| EFDF | ldaa | *L00F5 | %TPS |
| EFE1 | jsr | L99D7 | ;2D LOOKUP |
| EFE4 | ldab | *L00FA | |
| EFE6 | incb | | |
| EFE7 | mul | | |
| EFE8 | lsl | | |
| EFE9 | bcc | LEFED | |
| EFEB | ldaa | #0x80 | |
| EFED | LEFED:cmpa | 0x00,x | |
| EFEF | bcc | LEFF3 | ;BRANCH IF > |
| EFF1 | ldaa | 0x00,x | |
| EFF3 | LEFF3:tsta | | |
| EFF4 | beq | LEFFF | |
| EFF6 | cmpa | *L00F1 | ;TRANNNY MPH |
| EFF8 | bcs | LF009 | ;GO SET TCC PWM ACTIVE |
| EFFA | bra | LEFFF | |
| EFFC | LEFFC:bset | *L00A3,#0x08 | ;SET OFF MODE |
| FFFF | LEFFF:ldab | L0199 | |
| F002 | LF002:cmpb | L0198 | |
| F005 | bls | LF019 | |
| F007 | bra | LF016 | |
| F009 | LF009:bset | *L0096,#0x08 | ;SET TCC LOCKED |
| F00C | brset | *L0084,#0x80,LF01C | ;BRANCH IF MODE 4, ELSE |
| F010 | ldab | L0198 | |
| F013 | beq | LF01C | ;BRANCH IF Z, ELSE |
| F015 | dec | | ;DECREMENT |
| F016 | LF016:stab | L0198 | |
| F019 | LF019:bclr | *L0096,#0x08 | ;CLEAR TCC LOCKED |
| F01C | LF01C:ldaa | *L00F5 | %TPS |
| F01E | LF01E:staa | L01BE | ;PREV %TPS |
| F021 | brclr | *L0096,#0x08,LF028 | ;BRANCH IF TCC NOT LOCKED, ELSE |
| F025 | bclr | *L00A3,#0x10 | ;CLEAR POS DELTA TPS RELEASE OF TCC |
| F028 | LF028:ldd | *L00EB | ;16 BIT RPM |
| F02A | subd | L0261 | ;TURBINE RPM |
| F02D | bcc | LF039 | ;BRANCH IF TURBINE < ENGINE RPM, ELSE |
| F02F | cpd | #0xFF00 | ;65280 |
| F033 | bhi | LF040 | ;BRANCH IF > 65280, ELSE |
| F035 | ldab | #0x80 | ;LOAD 128 |
| F037 | bra | LF041 | |
| F039 | LF039:tsta | | |
| F03A | beq | LF040 | |
| F03C | ldab | #0x7F | |
| F03E | bra | LF041 | |
| F040 | LF040:lsrd | | |
| F041 | LF041:addd | #0x80 | |
| F043 | tba | | ;A NOW CONTAINS NEW UNFILTERED VALUE |
| F044 | ldx | #0x01C7 | ;OLD FILTERED TCC SLIP RPM |
| F047 | ldy | #0x96D6 | ;FILTER COEFFICIENT ADDRESS |
| F04B | jsr | L99F4 | ;LAG FILTER |
| F04E | brclr | *L0084,#0x80,LF067 | ;BRANCH IF NOT MODE 4, ELSE |
| F052 | ldaa | #0xFF | |

| | | | |
|------|-------------|--------------------|---|
| F054 | brset | *L0088,#0x01,LF062 | ;BRANCH IF "ALL PWM OUTPUTS COMMANDED ON" |
| F058 | ldaa | #0x04 | ;BIT 2 |
| F05A | anda | L0232 | ;MODE 4 CONTROL BYTE |
| F05D | beq | LF067 | ;BRANCH IF BIT 2 CLEAR, ELSE |
| F05F | ldaa | L0233 | ;LOAD COMMANDED DUTY CYCLE |
| F062 | LF062:staa | *L00EA | ;TCC PWM DUTY CYCLE |
| F064 | jmp | LF330 | |
| F067 | LF067:brclr | *L00A3,#0x40,LF06E | ;BRANCH IF TCC SLIP NOT REQ FOR A/C, ELSE |
| F06B | jmp | LF11C | |
| F06E | LF06E:brset | *L00A2,#0x40,LF08A | |
| F072 | brclr | *L00A2,#0x10,LF0AC | |
| F076 | ldaa | L01CD | |
| F079 | adda | L01C6 | |
| F07C | bcc | LF080 | |
| F07E | ldaa | #0xFF | |
| F080 | LF080:cmpa | L96D9 | ; |
| F083 | bcs | LF09E | |
| F085 | bclr | *L00A2,#0x10 | |
| F088 | bra | LF0AC | |
| F08A | LF08A:ldaa | L01CD | |
| F08D | adda | L01C6 | |
| F090 | bcc | LF094 | |
| F092 | ldaa | #0xFF | |
| F094 | LF094:cmpa | L96D8 | ; |
| F097 | bcs | LF09E | |
| F099 | bclr | *L00A2,#0x40 | |
| F09C | bra | LF0AC | |
| F09E | LF09E:brclr | *L00A3,#0x08,LF0A4 | ;BRANCH IF NOT OFF MODE, ELSE |
| F0A2 | bra | LF106 | |
| F0A4 | LF0A4:bclr | *L00A3,#0x0B | ;CLEAR APPLY, ON AND OFF MODES |
| F0A7 | bset | *L00A3,#0x04 | ;SET RELEASE MODE |
| F0AA | bra | LF10D | |
| F0AC | LF0AC:ldaa | L01CA | ;F31/TCCPWM TIMER |
| F0AF | cmpa | L95A1 | ; |
| F0B2 | bcc | LF0D9 | ;BRANCH IF < CAL, ELSE |
| F0B4 | brset | *L00A3,#0x08,LF106 | ;BRANCH IF OFF MODE, ELSE |
| F0B8 | brclr | *L00A3,#0x04,LF0D1 | ;BRANCH IF NOT RELEASE MODE, ELSE |
| F0BC | ldaa | L96CD | ; |
| F0BF | brset | *L001A,#0x04,LF0CA | ;BRANCH IF MALF (NOT ENABLED), ELSE |
| F0C3 | brclr | *L00D9,#0x40,LF0CA | ;BRANCH IF SMC NOT ENGAGED, ELSE |
| F0C7 | ldaa | L96CE | ; |
| F0CA | LF0CA:cmpa | L01C7 | ;FILTERED TCC SLIP RPM |
| F0CD | bcs | LF106 | |
| F0CF | bra | LF10D | |
| F0D1 | LF0D1:bset | *L00A3,#0x04 | ;SET RELEASE MODE |
| F0D4 | bclr | *L00A3,#0x03 | ;CLEAR APPLY AND ON MODES |
| F0D7 | bra | LF10D | |
| F0D9 | LF0D9:brclr | *L0096,#0x08,LF0FF | ;BRANCH IF TCC NOT LOCKED, ELSE |
| F0DD | bclr | *L00A3,#0x0C | ;CLEAR RELEASE AND OFF MODES |
| F0E0 | brset | *L00A3,#0x01,LF139 | ;BRANCH IF APPLY MODE, ELSE |
| F0E4 | brclr | *L00A3,#0x02,LF0EB | ;BRANCH IF NOT ON MODE, ELSE |
| F0E8 | jmp | LF26A | |
| F0EB | LF0EB:bset | *L00A3,#0x01 | ;SET APPLY MODE |
| F0EE | bclr | *L00A3,#0x80 | ;CLEAR ABSOLUTE SLIP HAS EXCEEDED KLOCKH |
| F0F1 | clr | L01CB | ; |

| | | | |
|------|-------------|--------------------|---------------------------------------|
| F0F4 | ldaa | L96DC | |
| F0F7 | ldab | *L00EA | ;TCC PWM DUTY CYCLE |
| F0F9 | bne | LF119 | ;BRANCH IF NOT Z, ELSE |
| F0FB | staa | *L00EA | ;STORE TCC PWM DUTY CYCLE |
| F0FD | bra | LF119 | |
| | | | |
| F0FF | LF0FF:bclr | *L00A3,#0x03 | ;CLEAR APPLY AND ON MODES |
| F102 | brclr | *L00A3,#0x08,LF109 | ;BRANCH IF NOT OFF MODE, ELSE |
| F106 | LF106:jmp | LF316 | |
| | | | |
| F109 | LF109:brclr | *L00A3,#0x04,LF110 | ;BRANCH IF NOT RELEASE MODE, ELSE |
| F10D | LF10D:jmp | LF20A | |
| | | | |
| F110 | LF110:bset | *L00A3,#0x04 | ;SET RELEASE MODE |
| F113 | clr | L01CD | |
| F116 | clr | L01CC | |
| F119 | LF119:jmp | LF330 | ; |
| | | | |
| F11C | LF11C:brclr | *L0090,#0x20,LF125 | ;BRANCH IF NOT BIT 5, ELSE |
| F120 | bset | *L009B,#0x01 | ;SET TCC SLIP OK FOR A/C ENGAGEMENT |
| F123 | bra | LF133 | ; |
| | | | |
| F125 | LF125:brset | *L008E,#0x80,LF12D | ;BRANCH IF A/C NOT REQUESTED, ELSE |
| F129 | brset | *L0086,#0x20,LF13C | ;BRANCH IF A/C CLUTCH OFF, ELSE |
| F12D | LF12D:bclr | *L009B,#0x01 | ;CLEAR TCC SLIP OK FOR A/C ENGAGEMENT |
| F130 | bclr | *L00A3,#0x40 | ;CLEAR TCC SLIP REQ FOR A/C |
| F133 | LF133:bset | *L00A3,#0x81 | ;SET APPLY MODE AND ABS SLIP > KLOCKH |
| F136 | clr | L01CB | |
| F139 | LF139:jmp | LF18B | |
| | | | |
| F13C | LF13C:ldaa | *L00BA | ;LOAD LV8 |
| F13E | ldx | #0x9756 | |
| F141 | jsr | L99CC | ;2D LOOKUP |
| F144 | bset | *L009B,#0x01 | ;SET TCC SLIP OK FOR A/C ENGAGEMENT |
| F147 | cmpa | L01C7 | ;FILTERED TCC SLIP RPM |
| F14A | bls | LF188 | |
| F14C | bclr | *L009B,#0x01 | ;CLEAR TCC SLIP OK FOR A/C ENGAGEMENT |
| F14F | ldaa | *L00BA | ;LOAD LV8 |
| F151 | ldx | #0x9760 | |
| F154 | jsr | L99CC | ;2D LOOKUP |
| F157 | inc | L01D2 | |
| F15A | ldab | L01D2 | |
| F15D | cba | | |
| F15E | bcc | LF188 | |
| F160 | clr | L01D2 | |
| F163 | ldaa | *L00BA | ;LOAD LV8 |
| F165 | ldx | #0x976A | ; |
| F168 | jsr | L99CC | ;2D LOOKUP |
| F16B | adda | *L00EA | ;ADD TCC PWM DUTY CYCLE |
| F16D | bcc | LF171 | ;BRANCH IF NO OVERFLOW, ELSE |
| F16F | ldaa | #0xFF | ;LOAD 255 |
| F171 | LF171:staa | *L00EA | ;STORE TCC PWM DUTY CYCLE |
| F173 | cmpa | #0xFF | |
| F175 | bcs | LF188 | |
| F177 | ldaa | L01D3 | |
| F17A | inc | L01D3 | |
| F17D | cmpa | L9755 | |
| F180 | bcs | LF188 | ;BRANCH IF < CAL, ELSE |
| F182 | clr | L01D3 | ;CLEAR TIMER |
| F185 | bset | *L009B,#0x01 | ;SET TCC SLIP OK FOR A/C ENGAGEMENT |
| F188 | LF188:jmp | LF330 | |
| | | | |
| F18B | LF18B:clr | L01C6 | ; |
| F18E | clr | L01CD | |

```

F191      clr      L01CE
F194      ldaa     L96DC
F197      cmpa     *L00EA
F199      bhi      LF19D
F19B      staa     *L00EA
F19D      LF19D: ldaa *L00F5
F19F      ldx      #0x96DD
F1A2      brclr    *L008E,#0x08,LF1B0
F1A6      ldx      #0x96EE
F1A9      brclr    *L008E,#0x04,LF1B0
F1AD      ldx      #0x96FF
F1B0      LF1B0: jsr L99D7
F1B3      suba     *L00EA
F1B5      bcs      LF1B8
F1B7      clra
F1B8      LF1B8: nega
F1B9      staa     *L00EA
F1BB      cmpa     L96D2
F1BE      bls      LF1E6
F1C0      ldaa     L01C7
F1C3      cmpa     L96D4
F1C6      bls      LF1CB
F1C8      bset     *L00A3,#0x80
F1CB      LF1CB: cmpa L96D3
F1CE      bcc      LF1E1
F1D0      brset    *L00A3,#0x80,LF1E6
F1D4      ldaa     L01CB
F1D7      cmpa     L96D5
F1DA      bcc      LF1EB
F1DC      inc      L01CB
F1DF      bra      LF207

F1E1      LF1E1: clr L01CB
F1E4      bra      LF207

F1E6      LF1E6: ldaa L96D2
F1E9      staa     *L00EA
F1EB      LF1EB: bset *L00A3,#0x02
F1EE      bclr     *L00A3,#0x8D

F1F1      ldaa     *L00F5
F1F3      ldx      #0x9710
F1F6      jsr      L99D7
F1F9      tab
F1FA      bclr     *L00A3,#0x20
F1FD      ldaa     *L00EA
F1FF      sba
F200      bcc      LF205
F202      bset     *L00A3,#0x20
F205      LF205: staa *L00F8
F207      LF207: jmp LF330

F20A      LF20A: inc L01CD
F20D      clr      L01C6
F210      ldaa     L9721
F213      cmpa     *L00EA
F215      bls      LF219
F217      staa     *L00EA
F219      LF219: ldab L96C9
F21C      ldaa     *L00EA
F21E      brset    *L00A3,#0x10,LF239
F222      tab
F223      ldaa     *L00F5
F225      ldx      #0x9744

;254 (99.2 % DC)
;TCC PWM DUTY CYCLE
;BRANCH IF > 99.2 %, ELSE
;STORE TCC PWM DUTY CYCLE
;%TPS
;
;BRANCH IF 4TH GEAR, ELSE
;
;BRANCH IF 3RD GEAR, ELSE
;
;2D LOOKUP
;SUBTRACT TCC PWM DUTY CYCLE
;BRANCH IF UNDERFLOW, ELSE
;CLEAR A
;INVERT DELTA-DC
;STORE NEW DUTY CYCLE
;0
;BRANCH IF < 0, ELSE
;FILTERED TCC SLIP RPM
;KLOCKH
;BRANCH IF <= CAL, ELSE
;SET ABS SLIP > KLOCKH
;
;BRANCH IF ABS SLIP > KLOCKH, ELSE

;
;TCC PWM DUTY CYCLE
;SET ON MODE
;CLEAR APPLY, RELEASE AND OFF MODES AND ABS
;SLIP > KLOCKH
;%TPS
;
;2D LOOKUP
;TRANSFER LOOKUP RESULT TO B REG
;CLEAR TCCRAMP IS NEGATIVE
;LOAD OLD DUTY CYCLE
;SUBTRACT LOOKUP VALUE
;BRANCH IF NO UNDERFLOW, ELSE
;SET TCCRAMP IS NEGATIVE
;STORE NEW DUTY CYCLE
;
;
;31.2 %DC
;TCC PWM DUTY CYCLE
;BRANCH IF > 31.2 %DC, ELSE
;STORE NEW DUTY CYCLE
;
;TCC PWM DUTY CYCLE
;BRANCH IF POS D-TPS RELEASE OF TCC, ELSE
;%TPS

```

```

F228      brset  *L00A2,#0x40,LF236
F22C      ld      #0x9733
F22F      brset  *L00A2,#0x10,LF236
F233      ld      #0x9722
F236      LF236:jsr  L99D7          ;2D LOOKUP
F239      LF239:aba
F23A      bcc     LF23E
F23C      ldaa    #0xFF
F23E      LF23E:staa *L00EA          ;TCC PWM DUTY CYCLE
F240      ldab    L01C7          ;FILTERED TCC SLIP RPM
F243      cmpb    L96CA
F246      bhi     LF253
F248      ldab    L01CD
F24B      stab    L01CE
F24E      cmpb    L96DA
F251      bhi     LF267
F253      LF253:ldaa *L00EA          ;TCC PWM DUTY CYCLE
F255      inca
F256      beq     LF267
F258      ldaa    L96DB
F25B      ldab    L01CD
F25E      subb    L01CE
F261      bcc     LF264
F263      clrb
F264      LF264:sba
F265      bcc     LF207
F267      LF267:jmp  LF316

F26A      LF26A:clr  L01C6
F26D      clr     L01CD
F270      clr     L01CE
F273      ldaa    L01CC
F276      bne     LF2A8
F278      ldaa    L96CC
F27B      cmpa    L01C7          ;FILTERED TCC SLIP RPM
F27E      bcs     LF288          ;BRANCH IF < CAL, ELSE
F280      nega
F281      cmpa    L01C7          ;FILTERED TCC SLIP RPM
F284      bhi     LF288          ;BRANCH IF > INVERSE CAL, ELSE
F286      bra     LF2AB          ;

F288      LF288:ldaa *L00F8          ;
F28A      brset  *L00A3,#0x20,LF298 ;BRANCH IF TCCRAMP IS NEGATIVE, ELSE
F28E      suba    L96CB
F291      bcc     LF29E
F293      bset    *L00A3,#0x20      ;SET TCCRAMP IS NEGATIVE
F296      bra     LF29E

F298      LF298:suba L96CB
F29B      bcc     LF29E
F29D      clra
F29E      LF29E:staa *L00F8
F2A0      ldaa    L96CF
F2A3      staa    L01CC
F2A6      bra     LF2AB

F2A8      LF2A8:dec  L01CC
F2AB      LF2AB:ldaa L01CF
F2AE      inca
F2AF      cmpa    L96D0
F2B2      bcs     LF2D0
F2B4      ldaa    *L00F8
F2B6      brclr  *L00A3,#0x20,LF2C4 ;BRANCH IF TCCRAMP IS POSITIVE, ELSE
F2BA      adda    L96D1

```

```

F2BD      bcc      LF2C9
F2BF      bclr     *L00A3,#0x20      ;SET TCCRAMP IS POSITIVE
F2C2      bra      LF2C9

F2C4      LF2C4:adda    L96D1
F2C7      bcs      LF2CB
F2C9      LF2C9:staa    *L00F8
F2CB      LF2CB:clr     L01CF
F2CE      bra      LF2D3

F2D0      LF2D0:inc     L01CF
F2D3      LF2D3:ldaa    *L00F5      ;%TPS
F2D5      ldx      #0x9710
F2D8      jsr      L99D7      ;2D LOOKUP
F2DB      staa     L02F8
F2DE      brclr    *L00A3,#0x20,LF2F2 ;BRANCH IF TCCRAMP IS POSITIVE, ELSE
F2E2      adda     *L00F8
F2E4      bcs      LF2F8
F2E6      ldaa     L02F8
F2E9      nega
F2EA      staa     *L00F8
F2EC      bset     *L00A3,#0x20      ;SET TCCRAMP IS POSITIVE
F2EF      clra
F2F0      bra      LF2F8

F2F2      LF2F2:adda    *L00F8
F2F4      bcc      LF2F8
F2F6      ldaa     #0xFF
F2F8      LF2F8:staa    *L00EA      ;TCC PWM DUTY CYCLE
F2FA      cmpa     L96D2
F2FD      bhi      LF314
F2FF      ldaa     L96D2      ;MIN TCC PWM DUTY CYCLE
F302      staa     *L00EA      ;TCC PWM DUTY CYCLE
F304      ldab     L02F8
F307      bclr     *L00A3,#0x20      ;CLEAR TCCRAMP IS NEGATIVE
F30A      ldaa     *L00EA      ;TCC PWM DUTY CYCLE
F30C      sba
F30D      bcc      LF312
F30F      bset     *L00A3,#0x20      ;SET TCCRAMP IS NEGATIVE
F312      LF312:staa    *L00F8
F314      LF314:bra     LF330

F316      LF316:bset    *L00A3,#0x08 ;SET OFF MODE
F319      bclr     *L00A3,#0x07      ;CLEAR APPLY, ON AND RELEASE MODES
F31C      ldaa     L01C6
F31F      ldab     #0x00
F321      cmpa     L96D7
F324      bhi      LF328
F326      ldab     #0xFF
F328      LF328:stab    *L00EA      ;TCC PWM DUTY CYCLE
F32A      inca
F32B      beq      LF330
F32D      staa     L01C6
F330      LF330:ldaa    #0x34      ;
F332      ldab     *L00EA      ;TCC PWM DUTY CYCLE
F334      lsld
F335      lsld
F336      std      L3DD4      ;STORE TCC PWM OUTPUT
F339      bset     *L0089,#0x08      ;ENABLE TCC SOLENOID
F33C      brclr    *L00A3,#0x08,LF343 ;BRANCH IF NOT OFF MODE, ELSE
F340      bclr     *L0089,#0x08      ;DISABLE TCC SOLENOID
F343      LF343:rts

```

;-----

```

; LOOKUP POWER ENRICHMENT FAR AND VARIOUS OTHER THINGS
; BRANCH HERE FROM SEGMENT TABLE - 100 MSEC ROUTINE
;-----
F344      ldx      #0x892E      ;POWER ENRICHMENT FAR MULT VS CLNT TEMP
F347      ldaa     *L00A6      ;LOAD CLNT TEMP (DEFAULTED)
F349      jsr      L99CC      ;2D LOOKUP
F34C      psha     ;SAVE LOOKUP RESULT ON STACK
F34D      ldaa     L0151      ;POWER ENRICHMENT TIMER
F350      inca     ;INCREMENT
F351      bne      LF354      ;BRANCH IF NOT Z, ELSE
F353      deca     ;DECREMENT
F354      LF354:staa    L0151      ;POWER ENRICHMENT TIMER
F357      ldx      #0x893A      ;PE FAR COMPONENT VS TIME
F35A      jsr      L99D7      ;2D LOOKUP
F35D      ldab     L02EF      ;PREV RPM
F360      cmpb     L8939      ;4000 RPM
F363      bhi      LF370      ;BRANCH IF > 4000 RPM, ELSE
F365      ldab     *L00AA      ;%TPS
F367      cmpb     L8938      ;99.8 %
F36A      bhi      LF370      ;BRANCH IF > CAL, ELSE
F36C      brclr    *L0099,#0x01,LF376 ;BRANCH IF NOT BIT 0, ELSE
F370      LF370:cmpa    #0x80      ;TABLE LOOKUP RESULT AND 128
F372      bcc      LF376      ;BRANCH IF >= 128, ELSE
F374      ldaa     #0x80      ;LOAD 128
F376      LF376:tab     ;TRANSFER LOOKUP RESULT TO B REG
F377      pula     ;GET PREV LOOKUP RESULT
F378      ldx      #0x0000      ;START WITH 0000
F37B      abx      ;ADD LOOKUP RESULT TO X
F37C      ldab     *L002C      ;FATI - TIMEOUT AFR
F37E      abx      ;ADD TO X
F37F      pshx     ;SAVE ON STACK
F380      tsx      ;X POINTS TO STACK
F381      jsr      L989D      ;8 X 16 MULTIPLY, PRODUCT ON STACK
F384      pulx     ;GET BACK INTO X
F385      tsta     ;TEST MSB
F386      beq      LF38A      ;BRANCH IF Z, ELSE
F388      ldab     #0xFF      ;LOAD 255
F38A      LF38A:stab    L0118      ;STORE POWER ENRICHMENT FAR

;-----
; LOOKUP A.E. MULTIPLIERS
;-----
F38D      ldaa     *L00A6      ;LOAD CLNT TEMP (DEFAULTED)
F38F      ldx      #0x8ADC      ;AE MULTIPLIER VS CTS
F392      brclr    *L008E,#0x01,LF399 ;BRANCH IF NOT P/N, ELSE
F396      ldx      #0x8AE6      ;A.E. MULTIPLIER VS CTS WHEN IN P/N
F399      LF399:jsr      L99CC      ;2D LOOKUP
F39C      staa     L015C      ;ACCEL ENRICHMENT MULTIPLIER VS CTS
F39F      ldaa     *L00F0      ;IATMAT
F3A1      lsra     ;RESCALE
F3A2      ldx      #0x8AF0      ;A.E. MULTIPLIER VS IAT
F3A5      jsr      L99D7      ;2D LOOKUP
F3A8      staa     L015D      ;STORE LOOKUP RESULT
F3AB      ldaa     L02C2      ;TIMER
F3AE      ldab     *L00B5      ;FUEL AIR RATIO MSB
F3B0      bne      LF3BF      ;BRANCH IF NOT Z, ELSE
F3B2      ldab     *L00B6      ;FAR LSB
F3B4      cmpb     L8969      ;111 - STOICH
F3B7      bhi      LF3BF      ;BRANCH IF > STOICH, ELSE
F3B9      tsta     ;TEST L02C2
F3BA      beq      LF3C5      ;BRANCH IF Z, ELSE
F3BC      deca     ;DECREMENT
F3BD      bra      LF3C2      ;

```

```

F3BF      LF3BF:inca                ;INCREMENT
F3C0              beq      LF3C5      ;BRANCH IF Z, ELSE
F3C2      LF3C2:staa      L02C2      ;STORE
F3C5      LF3C5:brclr     *L0011,#0x10,LF3CC ;MINOR LOOP COUNTER - 100 MSEC
F3C9              jmp      LF450      ;RETURN

F3CC      LF3CC:brset     *L0086,#0x80,LF3E5 ;BRANCH IF ENGINE RUNNING, ELSE
F3D0              ld      #0x88FC      ;TIME DELAY BEFORE DECAYING FATI VS CTS
F3D3              ldaa     L012A      ;TRUNCATED CTS FOR TABLE LOOKUPS
F3D6              jsr      L99D7      ;2D LOOKUP
F3D9              staa     L017B      ;SAVE TIME DELAY
F3DC              brset    *L0012,#0x08,LF402 ;
F3E0              clr      L002E      ;STARTUP TIMER
F3E3              bra      LF402

F3E5      LF3E5:ldaa      *L002E      ;STARTUP TIMER
F3E7              inca                ;INCREMENT
F3E8              beq      LF3EC      ;BRANCH IF Z, ELSE
F3EA              staa     *L002E      ;STORE STARTUP TIMER
F3EC      LF3EC:ldaa      *L009C      ;MODE WORD FLAG
F3EE              adda     *L008E      ;FMD INPUT STATUS WORD
F3F0              rora                ;SHIFT RIGHT
F3F1              bcc      LF3F6      ;BRANCH IF NO OVERFLOW (NOT P/N MODE), ELSE
F3F3              bset     *L009C,#0x14 ;SET FATI FATC FILTERS ACTIVE
F3F6      LF3F6:bset      *L009C,#0x01 ;SET P/N BIT
F3F9              brset    *L008E,#0x01,LF400 ;BRANCH IF P/N MODE, ELSE
F3FD              bclr     *L009C,#0x01 ;CLEAR P/N BIT
F400      LF400:bsr      LF451      ;GO LOOKUP FATI AND FATC

F402      LF402:ldaa      *L0083      ;BATT VOLTS
F404              ld      #0x8B8B      ;FUEL INJECTOR OFFSET VS BATT VOLTS
F407              jsr      L99CC      ;2D LOOKUP
F40A              staa     *L00B4      ;STORE OFFSET
F40C              ldaa     *L00AE      ;BLM CELL
F40E              cmpa     L8A35      ;5 ?
F411              bne      LF434      ;BRANCH IF NOT CELL 5, ELSE
F413              brclr    *L0096,#0x01,LF434 ;BRANCH IF CCP INACTIVE, ELSE
F417              ldaa     L02BB      ;TIMER
F41A              cmpa     L8A34      ;75d
F41D              bhi      LF424      ;BRANCH IF < 75d, ELSE
F41F              inc      L02BB      ;INCREMENT TIMER
F422              bra      LF437

F424      LF424:ldaa      *L00AF      ;FILTERED BLM
F426              cmpa     *L0034      ;BLM IN CELL 1
F428              bls      LF42D      ;BRANCH IF < CELL 1 BLM, ELSE
F42A              inc      L0034      ;INCREMENT CELL 1 BLM
F42D      LF42D:cmpa      *L0035      ;CELL 2 BLM
F42F              bls      LF434      ;BRANCH IF < CELL 2 BLM, ELSE
F431              inc      L0035      ;INCREMENT CELL 2 BLM
F434      LF434:clr      L02BB      ;CLEAR TIMER
F437      LF437:ldaa      *L0097      ;
F439              eora     #0x08      ;TOGGLE BIT 3
F43B              staa     *L0097      ;
F43D              brclr    *L0097,#0x08,LF450 ;BRANCH IF BIT 3 CLEAR, ELSE
F441              ldd      L0312      ;PREV FILTERE MPH
F444              subd     L0310      ;FILTERED MPH
F447              std      L0314      ;STORE DELTA MPH
F44A              ldd      L0310      ;FILTERED MPH
F44D              std      L0312      ;PREV FILTERED MPH
F450      LF450:rts                ;

```

```

;-----
; LOOKUP AND FILTER FATI AND FATC

```

```

; FATI(L002C) = [(8842 OR 8864 LOOKUP RESULT)+(883C LOOKUP RESULT)]*L002F
;   WHICH IS THEN FILTERED WITH L88EA
; FATC(L0116) = 8886 OR 88A8 LOOKUP RESULT
;   WHICH IS THEN FILTERED WITH L88EA
;-----
F451 LF451:ldx      #0x8842                ;? STARTUP ENRICH VS SUCLNT WHEN IN P/N
F454      brset   *L008E,#0x01,LF45B      ;BRANCH IF P/N MODE, ELSE
F458      ldx     #0x8864                ;? STARTUP ENRICH VS SUCLNT
F45B LF45B:pshx                    ;SAVE TABLE ADDRESS
F45C      ldaa    *L00F2                ;LOAD STARTUP COOLANT
F45E      jsr     L99CC                ;2D LOOKUP
F461      psha                    ;SAVE LOOKUP RESULT
F462      ldaa    *L0076                ;THIS NOT USED
F464      ldx     #0x883C                ;(ALL 00s)
F467      jsr     L99CC                ;2D LOOKUP
F46A      pulb                    ;GET PREVIOUS LOOKUP RESULT
F46B      aba                    ;ADD TO CURRENT RESULT
F46C      bcc     LF470                ;BRANCH IF NO OVERFLOW, ELSE
F46E      ldaa    #0xFF                ;LOAD 255
F470 LF470:ldab    *L002F                ;LOAD DECAY TERM
F472      mul                    ;A * B
F473      adca    #0x00                ;ROUND
F475      clrb                    ;CLEAR B
F476      brclr   *L009C,#0x04,LF491      ;BRANCH IF FATI FILTER NOT ACTIVE, ELSE
F47A      psha                    ;SAVE MSB ON STACK
F47B      ldx     #0x002C                ;OLD FILTERED FATI
F47E      ldy     #0x88EA                ;FILTER COEFFICIENT ADDRESS
F482      jsr     L99F4                ;LAG FILTER
F485      lslb                    ;MUL BY 2
F486      adca    #0x00                ;ROUND
F488      pulb                    ;GET BACK MSB
F489      cba                    ;COMPARE
F48A      bne     LF493                ;BRANCH IF THEY'RE NOT EQUAL, ELSE
F48C      bclr    *L009C,#0x04          ;CLEAR FATI FILTER ACTIVE
F48F      bra     LF493                ;

F491 LF491:std     *L002C                ;STORE FATI - TIMEOUT AFR
F493 LF493:pulx                    ;GET BACK TABLE ADDRESS
F494      ldab    #0x44                ;LOAD OFFSET
F496      abx                    ;ADD TO ADDRESS (8886 OR 88A8)
F497      ldaa    *L00A6                ;LOAD CLNT TEMP (DEFAULTED)
F499      jsr     L99CC                ;2D LOOKUP
F49C      brclr   *L009C,#0x10,LF4B7      ;BRANCH IF FATC FILTER NOT ACTIVE, ELSE
F4A0      psha                    ;SAVE TABLE LOOKUP RESULT ON STACK
F4A1      ldx     #0x0116                ;OLD FILTERED FATC
F4A4      ldy     #0x88EA                ;FILTER COEFFICIENT ADDRESS
F4A8      jsr     L99F4                ;LAG FILTER
F4AB      lslb                    ;MUL BY 2
F4AC      adca    #0x00                ;ROUND
F4AE      pulb                    ;GET BACK UNFILTERED LOOKUP RESULT
F4AF      cba                    ;COMPARE
F4B0      bne     LF4BA                ;BRANCH IF THEY'RE NOT EQUAL, ELSE
F4B2      bclr    *L009C,#0x10          ;CLEAR FATC FILTER ACTIVE
F4B5      bra     LF4BA                ;RETURN

F4B7 LF4B7:std     L0116                ;STORE FATC - CLNT TEMP AFR
F4BA LF4BA:rts

;-----
; TPS LOAD / FILTER CALC
;-----
F4BB LF4BB:ldaa    *L00AA                ;%TPS
F4BD      cmpa    L81E0                ;1.9 %TPS
F4C0      bhi     LF4C3                ;BRANCH IF > 1.95 %TPS

```

```

F4C2          clra                                ;
F4C3    LF4C3:staa    L0110                        ;STORE TPS
F4C6          ldaa    *L00F6                        ;TPS A/D COUNTS
F4C8          cmpa    L0111                        ;FILTERED TPS A/D COUNTS
F4CB          bhi     LF4E4                        ;BRANCH IF > FILTERED TPS A/D COUNTS
F4CD          cmpa    L90EA                        ;TPS LOW LIMIT - 8
F4D0          bls     LF4E4                        ;BRANCH IF <= 8, ELSE
F4D2          ldx     #0x0111                      ;OLD FILTERED TPS
F4D5          ldy     #0x86C5                      ;LOW THROTTLE FILTER COEFFICIENT
F4D9          jsr     L99F4                        ;LAG FILTER
F4DC          cmpa    L02E6                        ;COMPARE PREVIOUS
F4DF          bcc     LF4E4                        ;BRANCH IF > PREVIOUS, ELSE
F4E1          staa    L02E6                        ;SAVE LOW THROTTLE A/D COUNTS
F4E4    LF4E4:ldaa    L0111                        ;FILTERED TPS
F4E7          ldab    *L00BD                        ;FILTERED MPH
F4E9          beq     LF4FA                        ;BRANCH IF Z, ELSE
F4EB          cmpb    #0x0A                        ;10 MPH
F4ED          bhi     LF4F5                        ;BRANCH IF > 10 MPH, ELSE
F4EF          cmpa    *L00F6                        ;TPS A/D COUNTS
F4F1          bcs     LF510
F4F3          bra     LF50D

F4F5    LF4F5:bset    *L00DB,#0x10                ;
F4F8          bra     LF510

F4FA    LF4FA:brclr   *L00DB,#0x10,LF510          ;
F4FE          suba    L02E6
F501          cmpa    #0x02                        ;2 ?
F503          bhi     LF50D                        ;BRANCH IF > 2, ELSE
F505          ldaa    L0111                        ;FILTERED TPS
F508          adda    #0x02                        ;ADD 2
F50A          staa    L0111                        ;STORE FILTERED TPS
F50D    LF50D:bclr    *L00DB,#0x10                ;
F510    LF510:ldd     L0111                        ;FILTERED TPS A/D
F513          addd    #0x0080                      ;ROUND UP
F516          tab     ;TRANSFER MSB TO B REG
F517          ldaa    *L00F6                        ;TPS A/D COUNTS
F519          sba     ;SUBTRACT ACC
F51A          bcc     LF51D                        ;BRANCH IF NO UNDERFLOW, ELSE
F51C          clra   ;CLEAR A
F51D    LF51D:ldab    L86C3                        ;THROTTLE HIGH - THROTTLE LOW
F520          mul     ;
F521          addd    #0x0020                      ;ROUND
F524          lsld    ;RESCALE
F525          bcs     LF52A                        ;BRANCH IF OVERFLOW, ELSE
F527          lsld    ;RESCALE
F528          bcc     LF52C                        ;BRANCH IF NO OVERFLOW, ELSE
F52A    LF52A:ldaa    #0xFF                        ;LOAD 255
F52C    LF52C:staa    *L00AA                        ;%TPS
F52E          staa    *L00F5                        ;%TPS
F530          rts

;-----
; ESC ENABLE
;-----

F531    LF531:ldd     L3FCA                        ;KNOCK SENSOR INPUT
F534          pshb   ;SAVE LSB
F535          psha   ;SAVE MSB
F536          pulx   ;PUT INTO X REG
F537          subd    L029C                        ;SUBTRACT KNOCK COUNTS
F53A          tsta   ;
F53B          beq     LF53F                        ;
F53D          ldab    #0xFF                        ;
F53F    LF53F:stab    L029F                        ;STORE KNOCK

```

```

F542      stx      L029C                      ;STORE KNOCK COUNTS
F545      clrb
F546      ldx      *L0032                    ;RUN TIME
F548      cpx      L81D3                      ;
F54B      bcs      LF58B                      ;BRANCH IF RUN TIME < 81D3, ELSE
F54D      ldaa     *L00A6                    ;LOAD CLNT TEMP (DEFAULTED)
F54F      cmpa     L81D8                      ;IF CTS < THIS, DISABLE ESC
F552      bcs      LF58B                      ;BRANCH IF CTS2 < 81D8, ELSE
F554      ldaa     *L00AD                    ;LOAD RPM/12.5
F556      cmpa     L81D5                      ;IF RPM < THIS, DISABLE ESC
F559      bcs      LF58D                      ;BRANCH IF RPM < 81D5, ELSE
F55B      ldab     L9114                      ;LOAD DEFAULT KNOCK DEGREES?
F55E      ldaa     *L0083                    ;BATTERY VOLTS
F560      cmpa     #0x5D                      ;9.3 VOLTS
F562      bcs      LF58B                      ;BRANCH IF BATT VOLTS < 9.3, ELSE
F564      brset    *L0016,#0x10,LF58B        ;BRANCH IF KNOCK SENSOR MALF, ELSE
F568      ldaa     *L00AB                    ;LOAD NLRPMX
F56A      lsra
F56B      lsra                                ;RESCALE FOR LOOKUP
F56C      ldx      #0x8327                    ;ESC ATTACK RATE VS RPM
F56F      jsr      L99D7                      ;2D LOOKUP
F572      ldab     L0297                      ;LOAD
F575      mul
F576      lsld
F577      adda     *L00F3                    ;ADD KNOCK RETARD DEGREES
F579      bcc      LF57D                      ;BRANCH IF NO OVERFLOW, ELSE
F57B      ldaa     #0xFF                      ;LOAD 255
F57D      LF57D:ldab L81D6                    ;MAX KNOCK RETARD
F580      brclr    *L009B,#0x20,LF587        ;BRANCH IF NOT IN PE MODE, ELSE
F584      ldab     L81D7                      ;MAX KNOCK RETARD IN PE
F587      LF587:cba                          ;COMPARE
F588      bcc      LF58B                      ;BRANCH IF KNOCK RETARD >= THAN MAX, ELSE
F58A      tab
F58B      LF58B:stab *L00F3                  ;TRANSFER RETARD TO B REG
F58D      LF58D:rts                          ;STORE KNOCK RETARD DEGREES
                                           ;RETURN

;-----
; MAF VARIABLE LOOKUP
;-----
F58E      LF58E:ldx  #0x8C3D                    ;MAF VS RPM
F591      ldaa     *L00AC                    ;LOAD RPM/25
F593      jsr      L99D7                      ;2D LOOKUP
F596      ldab     L0187                      ;MAF IAT CORRECTION TERM
F599      mul
F59A      lsld
F59B      bcc      LF5A0                      ;LOOKUP RESULT * L0187
F59D      ldd      #0xFFFF                    ;MUL BY 2
F5A0      LF5A0:std  L0258                      ;BRANCH IF NO OVERFLOW, ELSE
                                           ;LOAD MAX
                                           ;STORE IAT AND RPM CORRECTION TERM
                                           ;USED TO CORRECT MAF FOR TORQUE CALCULATION
F5A3      ldx      #0x8C54                      ;RPM MAF CORRECTION TERM
F5A6      brclr    *L0091,#0x02,LF5AE        ;BRANCH IF RPM NOT >= 6375, ELSE
F5AA      ldaa     0x10,x                      ;8C64
F5AC      bra      LF5B3                      ;

F5AE      LF5AE:ldaa *L00AB                    ;LOAD NLRPMX
F5B0      jsr      L99D7                      ;2D LOOKUP
F5B3      LF5B3:ldab L0188                      ;IAT CORRECTION TO MAF
F5B6      mul
F5B7      lsld
F5B8      bcc      LF5BD                      ;LOOKUP RESULT * L0188
F5BA      ldd      #0xFFFF                    ;MUL BY 2
F5BD      LF5BD:std  L025C                      ;BRANCH IF NO OVERFLOW, ELSE
F5C0      rts                                ;LOAD MAX
                                           ;STORE MAF IAT AND RPM CORRECTION TERM

```

```

;-----
; ? BATCH FIRE FUEL OUT ROUTINE
;-----
F5C1  LF5C1:brset  *L0095,#0x10,LF626      ;BRANCH IF BATT VOLTS < 4.0 (rts), ELSE
F5C5      ldaa    L86C9                    ;OPTION WORD
F5C8      bpl     LF5D4                    ;BRANCH IF BIT 3 CLEAR (TIS CLEAR), ELSE
F5CA      ldaa    *L00A6                    ;LOAD CLNT TEMP (DEFAULTED)
F5CC      cmpa    L86D0                    ;00
F5CF      bcc     LF5D4                    ;BRANCH IF > 00, ELSE
F5D1      jmp     LF65F                    ;DO BATCH FIRE MODE FUEL OUTPUT

F5D4  LF5D4:ldx    #0x4000                  ;I/O DATA REG
F5D7      ldd     #0x0008                  ;MASK
F5DA      jsr     L9A70                    ;WRITE TO I/O DATA REG
F5DD      ldab    L026B                    ;LOAD FUEL OUT LOOP COUNTER
F5E0      brclr   *L0093,#0x40,LF5EC      ;BRANCH IF NO CAM REF PULSE, ELSE
F5E4      bclr    *L0093,#0x40            ;CLEAR CAM REF PULSE OCCURRED
F5E7      bset    *L0093,#0x20            ;SET CAM REF PULSE SEEN DURING CRANKING
F5EA      ldab    #0xFF                    ;LOAD 255
F5EC  LF5EC:incb                    ;INCREMENT LOOP COUNTER
F5ED      cmpb    #0x06                    ;COMPARE 6
F5EF      bcs     LF5FC                    ;BRANCH IF < 6, ELSE
F5F1      clr     L026C                    ;CLEAR COUNTER
F5F4      bclr    *L0012,#0x04            ;CLEAR BIT 2
F5F7      jsr     LF691                    ;? CLEAR FUEL OUTPUT REGS
F5FA      ldab    #0x00                    ;RESET COUNTER
F5FC  LF5FC:stab    L026B                    ;STORE LOOP COUNTER
F5FF      brset   *L0012,#0x04,LF610      ;BRANCH IF BIT 2, ELSE
F603      bset    *L0012,#0x04            ;SET BIT 2
F606      ldx     L030C                    ;CRANKING BPW
F609      ldab    *L00B4                    ;LOAD FUEL INJ OFFSET VS BATT VOLTS
F60B      abx                     ;ADD TO X
F60C      stx     *L00E4                    ;STORE FINAL BPW
F60E      bra     LF67B                    ;DO ASYNC PW OUTPUT ROUTINE

F610  LF610:jsr     LF691                    ;? CLEAR FUEL OUTPUT REGS
F613      brset   *L0093,#0x20,LF61B      ;BRANCH IF CAM PULSE SEEN DURING CRNK, ELSE
F617      brset   *L0016,#0x40,LF63D      ;BRANCH IF CAMSHAFT SENSOR MALF - RTS, ELSE
F61B  LF61B:ldaa    L026C                    ;LOAD COUNTER
F61E      cmpa    #0x04                    ;4 ?
F620      bcc     LF628                    ;BRANCH IF >= 4, ELSE
F622      inca                    ;INCREMENT
F623      staa    L026C                    ;STORE COUNTER
F626  LF626:bra     LF63D                    ;RETURN

F628  LF628:ldx     L030C                    ;CRANKING BPW
F62B      ldab    *L00B4                    ;LOAD FUEL INJ OFFSET VS BATT VOLTS
F62D      abx                     ;ADD TO X
F62E      stx     *L00E4                    ;STORE FINAL BPW

;-----
; EXERCISE INJECTOR OUTPUTS - SEFI
;-----
F630      ldx     #0xF63E                    ;SEGMENT TABLE ADDRESS
F633      ldab    L026B                    ;LOOP COUNTER (0 - 6)
F636      lslb                    ;MUL BY 4
F637      lslb                    ;
F638      abx                     ;ADD TO ADDRESS
F639      ldd     *L00E4                    ;LOAD BPW
F63B      jsr     0x00,x                    ;JUMP TO THAT ADDRESS
F63D  LF63D:rts

F63E      std     L3FD0

```

```

F641          rts

F642          bra      LF656

F644          nop
F645          nop
F646          std      L3F80
F649          rts

F64A          std      L3F82
F64D          rts

F64E          std      3F84
F651          rts

F652          std      3F86
F655          rts

F656          LF656: oraa    #0x40
F658          ldx      #0x4000
F65B          jsr      L9A70                      ; I/O ROUTINE
F65E          rts

;-----
; BATCH FIRE MODE FUEL OUTPUT - for cranking only
;-----
F65F          LF65F: jsr      LF691                      ; CLEAR FUEL OUTPUT REGS
F662          ldd      L030C                      ; BPW
F665          ldx      #0x0006                      ; 6
F668          idiv                     ; BPW/6
F669          pshx                     ; SAVE QUOTIENT ON STACK
F66A          ldd      #0x0008                      ; MASK
F66D          ldx      #0x4000                      ; I/O DATA REG ADDRESS
F670          jsr      L9A70                      ; TRANSMIT ON I/O PORT
F673          pulx                     ; GET BACK QUOTIENT
F674          jsr      LF678                      ; DO ASYNC FUEL OUT ROUTINE
F677          rts

;-----
; ASYNC FUEL OUT ROUTINE
;-----
F678          LF678: ldab    *L00B4                      ; FUEL INJ OFFSET VS BATT VOLTS
F67A          abx                     ; ADD TO PW
F67B          LF67B: stx      L3FF2                      ; STORE ? ASYNC PW
F67E          mul                     ; DELAY
F67F          ldd      L3FFC                      ; LOAD CPU CNTL REG
F682          oraa    #0x04                      ; SET BIT 2 - TRIGGER ASYNC PULSE
F684          pshx                     ; DELAY
F685          pulx                     ;
F686          std      L3FFC                      ; STORE CPU CNTL REG
F689          anda    #0xFB                      ; CLEAR BIT 2 - CLEAR ASYNC BIT
F68B          pshx                     ; DELAY
F68C          pulx                     ;
F68D          std      L3FFC                      ; STORE CPU CNTL REG
F690          rts                      ; RETURN

;-----
; CLEAR ?? FUEL OUTPUT CONTROL REGS
;-----
F691          LF691: ldd      #0x0000
F694          std      L3F80                      ; CLEAR 3F80
F697          std      L3FD0                      ; CLEAR 3FD0
F69A          ldx      #0x4000                      ; I/O DATA REG

```

```

F69D      psha                      ;SAVE ON STACK
F69E      std      L3F82             ;CLEAR 3F82
F6A1      oraa     #0x40             ;SET BIT 6
F6A3      jsr      L9A70             ;TRANSMIT ON I/O PORT
F6A6      pula     ;RESTORE
F6A7      std      L3F84             ;CLEAR 3F84
F6AA      pshx     ;DELAY
F6AB      pulx
F6AC      std      L3F86             ;CLEAR 3F86
F6AF      rts

;-----
; PART OF ALDL ROUTINE
; ALLOW RX/TX INTERRUPTS AND TRANSMIT BROADCAST MESSAGE
;-----
F6B0      LF6B0:brclr *L0084,#0x20,LF6CF ;BRANCH IF NOT "CLEAR NVRAM," ELSE
F6B4      ldaa     L024A             ;MESSAGE CKSUM
F6B7      cmpa     #0x05             ;5 ?
F6B9      bcc      LF6DB             ;BRANCH IF >= 5, ELSE
F6BB      ldab     L022B             ;INPUT MESSAGE BUFFER
F6BE      cmpb     #0x04             ;MODE 4 ?
F6C0      bne      LF6CF             ;BRANCH IF NOT 4, ELSE
F6C2      ldx      L024E             ;MODE 4 TIMER
F6C5      beq      LF6CC             ;BRANCH IF Z, ELSE
F6C7      bclr     *L0084,#0x80      ;CLEAR MODE 4 ACTIVE
F6CA      bra      LF6CF

F6CC      LF6CC:bset *L0084,#0x80    ;SET MODE 4 ACTIVE
F6CF      LF6CF:ldab *L0011          ;MINOR LOOP COUNTER
F6D1      cmpb     #0x09             ;? 100 MSEC
F6D3      bne      LF730             ;BRANCH IF NOT, ELSE
F6D5      inca     ;INCREMENT
F6D6      staa     L024A             ;STORE MSG CKSUM
F6D9      bra      LF6FF             ;

F6DB      LF6DB:clr  L022B            ;CLEAR INPUT MESSAGE BUFFER
F6DE      bclr     *L0084,#0x2A      ;%00101010
;CLEAR ALDL XMIT NEEDED
;CLEAR ALDL MODE 8
;CLEAR ALDL TESTER IN CONTROL

F6E1      bclr     *L0085,#0x40      ;CLEAR XMIT IN WORK
F6E4      ldd      L3DFC             ;? 2ND MCU CNTL REG
F6E7      andb     #0xFB             ;CLEAR BIT 2
F6E9      pshx     ;DELAY
F6EA      pulx
F6EB      std      L3DFC             ;STORE

;-----
; ALLOW 8192 SCI RX INTERRUPTS/SET SLEEP MODE
; B0 1 = CNTR INTERRUPT (BREAK ENABLED)
; B1 1 = RX WAKE UP BIT ENABLED
; B2 1 = RX ENABLED
; B5 1 = RDRF AND OR INTERRUPTS ENABLED
;-----
F6EE      ldaa     #0x27             ;%00100111
F6F0      staa     L4007             ;
F6F3      clra
F6F4      clrb
F6F5      std      L0202             ;SCI DATA MSG TBL INDEX
F6F8      std      L0200             ;SCI DATA BYTE COUNTER
F6FB      brset    *L0084,#0x80,LF70F ;BRANCH IF MODE 4, ELSE
F6FF      LF6FF:brclr *L0084,#0x80,LF71A ;BRANCH IF NOT MODE 4, ELSE
F703      ldx      L024C             ;MODE 4 TIMER
F706      inx      ;INCREMENT TIMER

```

```

F707          stx      L024C                      ;MODE 4 TIMER
F70A          cpx      L940E                      ;ALLOW MODE 4 FOR THIS LONG
F70D          bcs      LF730                      ;BRANCH IF < TIME LIMIT, ELSE
F70F  LF70F:bclr  *L0084,#0x80                    ;CLEAR MODE 4
F712          ld      #0x0001                      ;
F715          stx      L024E                      ;MODE 4 INHIBIT TIMER
F718          bra      LF730                      ;

F71A  LF71A:ldx      L024E                      ;MODE 4 INHIBIT TIMER
F71D          beq      LF730                      ;BRANCH IF = Z, ELSE
F71F          inx                      ;INC MODE TIMER
F720          stx      L024E                      ;STORE MODE INHIBIT TIMER
F723          cpx      L9410                      ;PREVENT MODE 4 FOR THIS LONG
F726          bcs      LF730                      ;BRANCH IF < 9410, ELSE
F728          clra                      ;CLEAR A
F729          clrb                      ;CLEAR B
F72A          std      L024E                      ;STORE MODE 4 INHIBIT TIMER
F72D          std      L024C                      ;STORE MODE 4 TIMER
F730  LF730:brset  *L0085,#0x40,LF774              ;BRANCH IF XMIT IN WORK (RETURN), ELSE
F734          brset  *L0084,#0x08,LF774              ;BRANCH IF MODE 8, ELSE
F738          ld      #0x9296                      ;INDEX - MESSAGE SCHEDULE TABLE
F73B          ldab     *L0011                      ;MINOR LOOP COUNTER
F73D          andb     #0x1E                      ;CLEAR BITS %00011110
F73F          abx                      ;ADD TO ADDRESS
F740          ld      0x00,x                      ;GET VALUE
F742          beq      LF774                      ;BRANCH IF ADDRESS = ZERO (RETURN), ELSE
F744  LF744:ldaa     0x02,x                      ;LOAD NEXT ADDRESS
F746          beq      LF774                      ;BRANCH IF MSG ID = ZERO (RETURN), ELSE
F748          brset  *L0085,#0x40,LF774              ;BRANCH IF XMIT IN WORK (RETURN), ELSE
F74C          bset     *L0085,#0x40              ;SET XMIT IN WORK BIT
F74F          stx      L0202                      ;SCI DATA MSG TBL POINTER
F752          ldab     L4008                      ;CPU TX/RX STATUS REG - CLEAR
F755          staa     L400A                      ;WR TO CPU TX REG
F758          staa     L0201                      ;STORE MESSAGE CKSUM
F75B          ldab     0x04,x                      ;GET NUM OF OUTPUT BYTES
F75D          stab     L0249                      ;MSG LENGTH
F760          ldab     #0x01                      ;SET BYTE COUNT = 1
F762          stab     L0200                      ;STORE SCI DATA BYTE COUNTER
F765          ldd      L3DFC                      ;? 2ND MCU CNTL REG
F768          orab     #0x04                      ;SET BIT 2
F76A          pshx                      ;DELAY
F76B          pulx                      ;
F76C          std      L3DFC                      ;STORE

;-----
; ALLOW TX INTERRUPTS
; B0  1 = CNTR INTERRUPT (BREAK ENABLED)
; B3  1 = TX ENABLED
; B7  1 = TDRE INTERRUPT ENABLED
;-----

F76F          ldaa     #0x89                      ;%10001001
F771          staa     L4007                      ;CPU TX/RX CNTL REG
F774  LF774:rts

;-----
; CLEAR NVRAM 0082 - 0011, AND LOAD INIT VALUES
; CALLED FROM INIT WHEN THERE IS A NEW MALF
; OR WHEN REQUESTED FROM MODE 4
;-----

F775  LF775:ldx      #0x0071                      ;INDEX
F778          clra                      ;
F779  LF779:staa     0x11,x                      ;
F77B          dex                      ;
F77C          bne      LF779                      ;LOOP TILL DONE

```

```

;-----
; FILL FIRST 15 BYTES WITH 0xFF
;-----
F77E      ldx      #0x000F      ;15
F781      coma     ;INVERT
F782      staa     *L0000      ;STORE
F784      LF784:staa    0x00,x  ;
F786      dex      ;
F787      bne      LF784      ;LOOP TILL DONE
F789      ldaa     L8C6E      ;PARK STARTUP IAC POSITION
F78C      staa     *L0057      ;IAC POSITION
F78E      ldd      L8698      ;0xBE00
F791      std      *L005A      ;FILTERED ENGINE TORQUE
F793      ldd      L869A      ;0x9600
F796      std      *L005C      ;FILTERED ENGINE TORQUE
F798      ldd      L977A      ;00
F79B      std      *L0072      ;
F79D      std      *L007A      ;
F79F      clrb     ;
F7A0      ldaa     L8695      ;INITIAL VALUE
F7A3      std      *L005E
F7A5      std      *L0060
F7A7      std      *L0062
F7A9      ldaa     L8DD5      ;INITIAL VALUE
F7AC      staa     *L0065      ;FILTERED LOW AIRFLOW
F7AE      ldaa     L8DD6      ;INITIAL VALUE
F7B1      staa     *L0067      ;FILTERED LOW AIRFLOW
F7B3      ldaa     L977E      ;INITIAL VALUE
F7B6      staa     *L0069      ;F31 VARIABLE
F7B8      ldaa     L977F      ;INITIAL VALUE
F7BB      staa     *L006A      ;F31 VARIABLE
F7BD      ldaa     L9780      ;INITIAL VALUE
F7C0      staa     *L006B      ;F31 VARIABLE
F7C2      ldaa     L9781      ;INITIAL VALUE
F7C5      staa     *L006C      ;F31 VARIABLE
F7C7      ldaa     L9782      ;INITIAL VALUE
F7CA      staa     *L006D      ;F31 VARIABLE
F7CC      ldaa     L9783      ;INITIAL VALUE
F7CF      staa     *L006E      ;F31 VARIABLE
F7D1      ldaa     L8D6C      ;INITIAL VALUE
F7D4      staa     *L006F      ;IATMAT
F7D6      ldx      #0x0001      ;
F7D9      stx      *L0024      ;
F7DB      bset     *L009A,#0x40 ;
F7DE      bset     *L002F,#0xFF ;SET FATI DECAY TERM TO MAX

;-----
; RESET BLMS
;-----
F7E1      LF7E1:ldaa    #0x80      ;128 - DEFAULT BLM VALUE
F7E3      ldx      #0x0022      ;NUMBER OF LOOPS
F7E6      LF7E6:staa    0x33,x    ;
F7E8      dex      ;DEC LOOP COUNTER
F7E9      bne      LF7E6      ;REPEAT UNTIL DONE
F7EB      rts      ;

;-----
; SUM ALL MALF FLAGS
;-----
F7EC      LF7EC:ldd     #0x0001      ;LOAD 0001
F7EF      addb     *L001B      ;ADD L001B
F7F1      adca     #0x00      ;ROUND
F7F3      addb     *L001C      ;ADD L001C
F7F5      adca     #0x00      ;ROUND

```

```

F7F7      addb    *L001D      ;ADD L001D
F7F9      adca    #0x00      ;ROUND
F7FB      addb    *L001E      ;ADD L001E
F7FD      adca    #0x00      ;ROUND
F7FF      addb    *L001F      ;ADD L001F
F801      adca    #0x00      ;ROUND
F803      addb    *L0020      ;ADD L0020
F805      adca    #0x00      ;ROUND
F807      addb    *L0021      ;ADD L0021
F809      adca    #0x00      ;ROUND
F80B      addb    *L0022      ;ADD L0022
F80D      adca    #0x00      ;ROUND
F80F      std     *L0024      ;STORE
F811      rts          ;RETURN

```

```

;-----
; STORE CONTENTS OF ACCD IN 3FAE - 3FB4
; X = 0x3F94
; X = 0x3FAE
;
;-----

```

```

F812      LF812:ldy    #0x0004      ;NUMBER OF LOOPS
F816      LF816:std     0x00,x      ;STORE
F818              inx
F819              inx
F81A              dey
F81C              bne     LF816      ;LOOP TILL DONE
F81E      rts

```

```

;-----
; TABLE LOOKUP - FOR F31 TRANS
;-----

```

```

F81F      LF81F:pshx          ;SAVE TABLE ADDRESS ON STACK
F820              ldx     #0x01D0      ;GET RATIO OF FILTERED ENGINE TORQUE
F823              ldaa    L0301      ;
F826              bpl     LF833      ;
F828              nega          ;
F829              jsr     L98B9      ;8 x 16 MULTIPLY
F82C              lsld          ;MUL BY 2
F82D              nega          ;INVERT
F82E              negb
F82F              sbca    #0x00      ;ROUND
F831              bra     LF837

```

```

F833      LF833:jsr     L98B9      ;8 x 16 MULTIPLY
F836              lsld          ;
F837      LF837:pulx          ;
F838              lsll          ;
F839              adca    #0x00      ;ROUND
F83B              suba    L0301      ;
F83E              adda    #0x80      ;ROUND
F840              staa    *L00FA      ;SAVE
F842              brset   *L0097,#0x10,LF84B      ;
F846              ldaa    *L00F5      ;%TPS
F848              jsr     L99D7      ;2D LOOKUP
F84B      LF84B:rts

```

```

;-----
; CLEAR MALF FLAGS
;-----

```

```

F84C      LF84C:clra          ;
F84D              brclr   *L0013,#0x08,LF853      ;BRANCH IF NOT 24X CRANK SIGNAL MALF, ELSE
F851              staa    *L0087      ;LOOP COUNTER
F853      LF853:ldx     #0x0013      ;INDEX MALF FLAGS

```

```

F856  LF856:staa    0x12,x                ;0025
F858                dex                    ;
F859                bne    LF856            ;LOOP TILL DONE
F85B                ldx     #0x0001         ;
F85E                stx     *L0024          ;SUM OF ALL MALF WORDS
F860                staa   *L0026          ;EGR TIMER
F862                staa   *L0027          ;EGR TIMER
F864                staa   *L0028          ;EGR TIMER
F866                staa   *L0029          ;EGR TIMER
F868                staa   *L002A          ;EGR STATUS FLAG
F86A                staa   *L002B          ;CLEAR PRNDL SW MONITOR
F86C                staa   *L0064          ;CLEAR L0064
F86E                staa   *L0071
F870                staa   L0131            ;CLEAR 24X SIGNAL MALF TIMER
F873                staa   L02FB
F876                staa   L02FC
F879                bclr   *L0012,#0x20
F87C                rts

;-----
; TEST MALF OPTION WORDS AND UPDATE MALF FLAGS
; X REG IS THE INDEX ADDRESS = 90AB
; B REG CONTAINS THE OFFSET
; A REG CONTAINS THE MASK FOR MALF WORD
;-----
F87D  LF87D:pshx                ;SAVE ADDRESS ON STACK
F87E                abx                ;ADD OFFSET TO ADDRESS
F87F                bita    0x00,x        ;TEST BIT
F881                beq     LF8A4          ;BRANCH IF NOT SET, ELSE
F883                bita    0x08,x        ;TEST BIT
F885                beq     LF88A          ;BRANCH IF NOT SET, ELSE
F887                bset    *L00A0,#0x30  ;SET TIMER TO 48
F88A  LF88A:psha                ;SAVE
F88B                ldx     #0x0013        ;INDEX MALF FLAGS
F88E                abx                ;ADD B TO X
F88F                oraa    0x00,x        ;SET BITS
F891                staa    0x00,x        ;STORE
F893                clr     L0023          ;CLEAR MALF TIMER
F896                pula    bita          ;RESTORE A
F897                bita    0x08,x        ;TEST "MEMORY" MALF WORD
F899                bne     LF8AE          ;BRANCH IF "MEMORY" MALF SET, ELSE
F89B                oraa    0x08,x        ;SET BITS ACCORDING TO MASK
F89D                staa    0x08,x        ;STORE
F89F                jsr     LF7EC          ;GO ADD UP AND STORE L0024
F8A2                bra     LF8AE          ;
F8A4  LF8A4:ldx     #0x0013            ;INDEX MALF FLAGS
F8A7                abx                ;OFFSET
F8A8                eora    #0xFF          ;TOGGLE ALL BITS
F8AA                anda    0x00,x        ;CLEAR BITS ACCORDING TO MASK
F8AC                staa    0x00,x        ;STORE
F8AE  LF8AE:pulx                ;RESTORE ORIGINAL ADDRESS
F8AF                rts                ;RETURN

;-----
; CALLED FROM MAJOR SPARK ROUTINE, EXECUTED ONCE PER 12.5 MSEC
; CALCULATE ENGINE TORQUE
; CALCULATE ACTUAL MASS AIR THIS REF
; ?CALCULATE OVER/UNDER IDLE SPARK
; CALCULATE TORQ MGMT SPARK
; 560 BYTES OF CODE
;-----
F8B0  F8B0:  bclr   *L0098,#0x02          ;CLEAR BIT 1
F8B3                ldd     L025A          ;MASS AIR FLOW

```

| | | | |
|------|-------------|--------------------|--|
| F8B6 | lsrd | | ;DIV BY 2 |
| F8B7 | ldx | *L00C0 | ;LOAD REF PERIOD |
| F8B9 | jsr | L98FA | ;16 x 16 MULTIPLY |
| | | | ;NOW HAVE MASS OF AIR, NO TIME COMPONENT |
| F8BC | pshb | | ;SAVE LSB |
| F8BD | psha | | ;SAVE MSB |
| F8BE | tsx | | ;X POINTS TO STACK |
| F8BF | ldaa | L8644 | ;190 - ?FUDGE FACTOR |
| F8C2 | jsr | L989D | ;8 x 16 MULTIPLY, PRODUCT ON STACK |
| F8C5 | brset | *L009C,#0x80,LF8D8 | ;BRANCH IF CLOSED LOOP, ELSE |
| F8C9 | brset | *L0086,#0x02,LF8D8 | ;BRANCH IF HOT OPEN LOOP, ELSE |
| F8CD | brclr | *L0099,#0x01,LF8D3 | ; |
| F8D1 | bra | LF8D8 | |
| F8D3 | LF8D3:ldaa | L801A | ;OPTION WORD - TIS SET |
| F8D6 | beq | LF8EE | ;BRANCH IF Z, ELSE |
| F8D8 | LF8D8:brset | *L008E,#0x04,LF8EE | ;BRANCH IF NOT 3 RD GEAR, ELSE |
| F8DC | brclr | *L008E,#0x08,LF8EE | ;BRANCH IF 4 TH GEAR, ELSE |
| F8E0 | ldd | L02F3 | ; |
| F8E3 | cpd | L8645 | ;80 |
| F8E7 | bls | LF8F4 | ;BRANCH IF <= CAL, ELSE |
| F8E9 | bset | *L0096,#0x40 | ;SET USE F7MAXTRQ FOR 3 RD GEAR TORQ MGMT |
| F8EC | bra | LF904 | |
| F8EE | LF8EE:bclr | *L0096,#0x40 | ;CLEAR USE F7MAXTRQ FOR 3 RD GEAR TORQ MGMT |
| F8F1 | ldd | #0xFFFF | ;LOAD MAX |
| F8F4 | LF8F4:addd | #0x0001 | ;ADD ONE |
| F8F7 | std | L02F3 | ;STORE |
| F8FA | ldaa | *L00AC | ;LOAD RPM/25 |
| F8FC | cmpa | L8647 | ;2000 RPM |
| F8FF | bcc | LF904 | ;BRANCH IF > CAL, ELSE |
| F901 | bset | *L0098,#0x02 | ;SET BIT 1 |
| F904 | LF904:ldaa | *L00AC | ;LOAD RPM/25 |
| F906 | cmpa | #0xF0 | ;6000 RPM |
| F908 | bls | LF90C | ;BRANCH IF <= 6000 RPM, ELSE |
| F90A | ldaa | #0xF0 | ;LOAD 6000 RPM |
| F90C | LF90C:ldab | *L00B5 | ;FUEL AIR RATIO MSB |
| F90E | bne | LF916 | ;BRANCH IF NOT Z, ELSE |
| F910 | ldab | *L00B6 | ;FAR LSB |
| F912 | cmpb | #0xB0 | ;176 |
| F914 | bls | LF918 | ;BRANCH IF < 176, ELSE |
| F916 | LF916:ldab | #0xB0 | ;176 |
| F918 | LF918:ldx | #0x84D8 | ;ENGINE EFFICIENCY VS RPM AND FAR |
| F91B | jsr | L995A | ;3D LOOKUP |
| F91E | tsx | | ;X POINTS TO STACK |
| F91F | jsr | L989D | ;8 X 16 MULTIPLY, PRODUCT ON STACK |
| | | | ;EFFICIENCY * MAF THIS REF PERIOD |
| F922 | std | 0x00,x | ;SAVE ON STACK |
| F924 | ldaa | *L00AC | ;LOAD RPM/25 |
| F926 | cmpa | #0xF0 | ;6000 RPM |
| F928 | bls | LF92C | ;BRANCH IF <= 6000 RPM, ELSE |
| F92A | ldaa | #0xF0 | ;LOAD 6000 RPM |
| F92C | LF92C:ldab | L012A | ;TRUNCATED CTS FOR TABLE LOOKUPS |
| F92F | lsrb | | ;RESCALE |
| F930 | ldx | #0x852F | ; |
| F933 | jsr | L995A | ;3D LOOKUP |
| F936 | clrb | | ; |
| F937 | lsrd | | ;DIV BY 2 |
| F938 | nega | | ;INVERT MSB |
| F939 | negb | | ;INVERT LSB |
| F93A | sbca | #0x00 | ;ROUND |
| F93C | tsx | | ;X POINTS TO STACK |
| F93D | addd | 0x00,x | ;ADD TO CONTENTS OF STACK |
| F93F | bcs | LF943 | ;BRANCH IF OVERFLOW, ELSE |

| | | | |
|------|-------------|--------------------|---|
| F941 | clra | | ;CLEAR A |
| F942 | clrb | | ;CLEAR B |
| F943 | LF943:std | 0x00,x | ;SAVE ON STACK |
| F945 | std | L0276 | ;STORE ENGINE TORQUE |
| F948 | ldaa | L8594 | ;64 |
| F94B | ldab | L0265 | ;RATIO OF ENGINE TO TURBINE RPM |
| F94E | incb | | ;INCREMENT |
| F94F | beq | LF960 | ;BRANCH IF Z, ELSE |
| F951 | decb | | ;DECREMENT |
| F952 | ldx | *L00ED | ;1 SECOND TIMER |
| F954 | cpx | L8595 | ;20 SECONDS |
| F957 | bis | LF960 | ;BRANCH IF <= 20, ELSE |
| F959 | tba | | ;COPY L0265 TO A REG |
| F95A | ldx | #0x8597 | ;ENGINE TORQUE MULTIPLIER VS RATIO |
| F95D | jsr | L99D7 | ;2D LOOKUP |
| F960 | LF960:tsx | | ;X POINTS TO STACK |
| F961 | jsr | L989D | ;8 x 16 MULTIPLY, PRODUCT ON STACK |
| F964 | std | L0278 | ;STORE TURBINE TORQUE |
| F967 | clra | | ;CLEAR A |
| F968 | staa | L027A | ;SPARK VARIABLE - ? 2'S COMP MAX SPARK |
| F96B | pulx | | ;RESTORE X |
| F96C | brclr | *L009E,#0x46,LF972 | ;BRANCH IF %01000110 CLEAR, ELSE |
| F970 | bra | LF9BF | ; |
| F972 | LF972:brclr | *L009E,#0x08,LF979 | ;BRANCH IF NOT BIT 3, ELSE |
| F976 | jmp | LFAD9 | ;CLEAR SPK RET AND RETURN |
| F979 | LF979:staa | L0288 | ;SPARK TERM - ? 2'S COMP MAX SPARK |
| F97C | staa | L0289 | ;SPARK RETARD TERM |
| F97F | staa | L027B | ;RATIO OF ENGINE TORQUE TO TORQUE THRESHOLD |
| F982 | brset | *L0098,#0x02,LF9BC | ; |
| F986 | brset | *L008E,#0x01,LF9BC | ;BRANCH IF P/N MODE, ELSE |
| F98A | ldd | *L00EB | ;RPM |
| F98C | std | L028D | ;STORE PREV RPM |
| F98F | ldaa | L864A | ;THRESHOLD TO IMPLEMENT TORQUE MGMT SPK RET |
| F992 | brset | *L0096,#0x40,LF9A0 | ;BRANCH IF USE F7MAXTRQ TABLE, ELSE |
| F996 | ldaa | L8648 | ;THRESHOLD TO IMPLEMENT TORQUE MGMT SPK RET |
| F999 | brclr | *L0096,#0x08,LF9A0 | ;BRANCH IF TCC NOT LOCKED, ELSE |
| F99D | ldaa | L8649 | ;THRESHOLD TO IMPLEMENT TORQUE MGMT SPK RET |
| F9A0 | LF9A0:cmpa | L0278 | ;TURBINE TORQUE |
| F9A3 | bcc | LF9BC | ;BRANCH IF <= CAL (RETURN), ELSE |
| F9A5 | ldx | L0278 | ;LOAD TURBINE TORQUE |
| F9A8 | clrb | | ;CLEAR B |
| F9A9 | fdiv | | ;D/X |
| F9AA | pshx | | ;SAVE QUOTIENT |
| F9AB | pulb | | ;GET LSB |
| F9AC | pula | | ;GET MSB |
| F9AD | ldaa | L0296 | ;LIMITED NLRPMX |
| F9B0 | stab | L027B | ;STORE LSB (LOOKUP AXIS FOR L85A8 TABLE) |
| F9B3 | ldx | #0x85A8 | ;RATIO OF TURBINE TORQUE TO TORQUE THRESH |
| F9B6 | jsr | L995A | ;F7MAXTRQ TABLE |
| F9B9 | staa | L027A | ;3D LOOKUP |
| F9BC | LF9BC: jmp | LFADF | ;STORE LOOKUP RESULT - ?2'S COMP MAX SPARK |
| F9BF | LF9BF:ldx | #0x865C | ;RETURN |
| F9C2 | ldy | #0x005E | ;ROM INDEX |
| F9C6 | brset | *L009E,#0x02,LF9DC | ;RAM INDEX |
| F9CA | ldx | #0x8666 | ; |
| F9CD | ldy | #0x0060 | ; |
| F9D1 | brset | *L009E,#0x04,LF9DC | ; |
| F9D5 | ldx | #0x8670 | ; |
| F9D8 | ldy | #0x0062 | ; |
| F9DC | LF9DC:ldd | *L00EB | ;RPM |

| | | | | |
|------|--------------|----------------------|------------|----------------------------------|
| F9DE | cpd | L028D | | ;PREV RPM |
| F9E2 | bcs | LF9E7 | | ;BRANCH IF DECREASING, ELSE |
| F9E4 | std | L028D | | ;STORE PREV RPM |
| F9E7 | LF9E7: addd | L868C | | ;16 RPM |
| F9EA | subd | L0281 | | ;PREV 16 BIT RPM |
| F9ED | bcs | LF9F7 | | ;BRANCH IF RPM INCREASING, ELSE |
| F9EF | ldaa | L028F | | ;LOAD |
| F9F2 | inca | | | ;INCREMENT |
| F9F3 | beq | LF9FB | | ;BRANCH IF Z, ELSE |
| F9F5 | bra | LF9F8 | | ; |
| | | | | |
| F9F7 | LF9F7: clra | | | ;CLEAR |
| F9F8 | LF9F8: staa | L028F | | ;SAVE TIMER |
| F9FB | LF9FB: brclr | *L009E, #0x10, LFA08 | ; | |
| F9FF | ldd | L0276 | | ;ENGINE TORQUE (FT-LBS = 2N) |
| FA02 | cpd | L8691 | | ;0x0028 |
| FA06 | bls | LFA54 | | ;BRANCH IF <= CAL, ELSE |
| FA08 | LFA08: ldaa | *L00AA | ;%TPS | |
| FA0A | cmpa | L8694 | | ; |
| FA0D | bls | LFA54 | | ;BRANCH IF <= CAL, ELSE |
| FA0F | ldaa | L0280 | | ;TIMER/COUNTER |
| FA12 | beq | LFA23 | | ;BRANCH IF Z, ELSE |
| FA14 | cmpa | L8696 | | ; |
| FA17 | bcs | LFA33 | | ;BRANCH IF < CAL, ELSE |
| FA19 | ldab | L028F | | ;TIMER/COUNTER |
| FA1C | cmpb | L868E | | ; |
| FA1F | bls | LFA33 | | ;BRANCH IF <= CAL, ELSE |
| FA21 | bra | LFA38 | | ; |
| | | | | |
| FA23 | LFA23: ldd | L028D | | ;LOAD PREV RPM |
| FA26 | subd | *L00EB | ;RPM | |
| FA28 | cpd | L868F | | ;75 RPM |
| FA2C | bls | LFA5A | | ;BRANCH IF D-RPM <= 75 RPM, ELSE |
| FA2E | ldaa | L0280 | | ;TIMER/COUNTER |
| FA31 | bra | LFA4C | | ; |
| | | | | |
| FA33 | LFA33: cmpa | L8697 | | ; |
| FA36 | bcs | LFA4C | | ;BRANCH IF L0280 < CAL, ELSE |
| FA38 | LFA38: pshx | | | ;SAVE ROM INDEX |
| FA39 | pshy | | | ;SAVE RAM INDEX |
| FA3B | ldab | L8693 | | ;FILTER COEFFICIENT |
| FA3E | pulx | | | ;GET RAM INDEX INTO X REG |
| FA3F | pshx | | | ;SAVE AGAIN |
| FA40 | pshb | | | ;SAVE FILTER COEFFICIENT |
| FA41 | tsy | | | ;Y CONTAINS FILTER COEFFICIENT |
| FA43 | jsr | L99F4 | | ;LAG FILTER |
| FA46 | ins | | | ;IGNORE FILTER COEFFICIENT |
| FA47 | puly | | | ;RESTORE RAM INDEX |
| FA49 | pulx | | | ;RESTORE ROM INDEX |
| FA4A | bra | LFA54 | | ; |
| | | | | |
| FA4C | LFA4C: inca | | | ;INCREMENT |
| FA4D | beq | LFA5A | | ;BRANCH IF Z, ELSE |
| FA4F | staa | L0280 | | ;SAVE TIMER/COUNTER |
| FA52 | bra | LFA5A | | ; |
| | | | | |
| FA54 | LFA54: clr | L0280 | | ;TIMER/COUNTER |
| FA57 | bclr | *L009E, #0x46 | ;%01000110 | |
| FA5A | LFA5A: clra | | | ; |
| FA5B | brclr | *L009E, #0x10, LFAAB | | ;BRANCH IF BIT 4 CLEAR, ELSE |
| FA5F | bclr | *L009E, #0x10 | | ;CLEAR BIT 4 |
| FA62 | ldab | *L00AA | ;%TPS | |
| FA64 | cmpb | L868B | | ;45% TPS |
| FA67 | bls | LFAA8 | | ;BRANCH IF <= CAL, ELSE |

| Address | OpCode | OpCode Hex | OpCode Mnemonic | Comments |
|---------|------------|------------|-----------------|---|
| FA69 | pshx | | | ;SAVE RAM INDEX |
| FA6A | ldd | 0x00,y | | ;L005E OR |
| | | | | ;L0060 OR |
| | | | | ;L0062 |
| FA6D | cmpa | #0x3F | | ;MSB 0x3F ? |
| FA6F | bls | LFA74 | | ;BRANCH IF MSB <= 0x3F, ELSE |
| FA71 | ldd | #0x3FFF | | ;LOAD |
| FA74 | LFA74:lsld | | | ;MUL BY 4 |
| FA75 | lsld | | | |
| FA76 | ldx | #0x864B | | ;TABLE ADDRESS |
| FA79 | jsr | L99D7 | | ;2D LOOKUP |
| FA7C | staa | L046D | | ;SAVE LOOKUP RESULT |
| FA7F | ldab | L0278 | | ;TURBINE TORQUE |
| FA82 | mul | | | ;LOOKUP RESULT * TURBINE TORQUE |
| FA83 | lsld | | | ;MUL BY 2 |
| FA84 | bcc | LFA88 | | ;BRANCH IF NO OVERFLOW, ELSE |
| FA86 | ldaa | #0xFF | | ;LOAD 255 |
| FA88 | LFA88:pulx | | | ;GET RAM INDEX |
| FA89 | pshb | | | ;SAVE PRODUCT ON STACK |
| FA8A | psha | | | ; |
| FA8B | ldaa | *L00AC | | ;LOAD RPM/25 |
| FA8D | suba | #0x70 | | ;OFFSET |
| FA8F | bcc | LFA92 | | ;BRANCH IF NO UNDERFLOW, ELSE |
| FA91 | clra | | | ;CLEAR A |
| FA92 | LFA92:jsr | L99D7 | | ;2D LOOKUP |
| FA95 | clrb | | | ; |
| FA96 | pulx | | | ;GET PRODUCT FROM STACK |
| FA97 | fddiv | | | ;D/X |
| FA98 | pshx | | | ;TRANSFER QUOTIENT TO ACCD |
| FA99 | pula | | | ; |
| FA9A | pulb | | | ; |
| FA9B | staa | L0285 | | ;SAVE MSB |
| FA9E | tab | | | ;COPY TO B REG |
| FA9F | ldaa | L0296 | | ;LIMITED NLRPMX |
| FAA2 | ldx | #0x85A8 | | ;TABLE ADDRESS |
| FAA5 | jsr | L995A | | ;3D LOOKUP |
| FAA8 | LFAA8:staa | L0286 | | ;SAVE LOOKUP RESULT |
| FAAB | LFAAB:ldaa | #0xFF | | ;LOAD 255 |
| FAAD | ldab | L0280 | | ;TIMER/COUNTER |
| FAB0 | cmpb | 0x00,y | | ;L005E OR |
| | | | | ;L0060 OR |
| | | | | ;L0062 |
| FAB3 | bcc | LFAC0 | | ;BRANCH IF < L0280, ELSE |
| FAB5 | ldaa | 0x00,y | | ;L005E OR |
| | | | | ;L0060 OR |
| | | | | ;L0062 |
| FAB8 | psha | | | ;SAVE ON STACK |
| FAB9 | clra | | | ; |
| FABA | psha | | | ;SAVE |
| FABB | pulx | | | ;GET INTO X REG |
| FABC | fddiv | | | ;D/X |
| FABD | pshx | | | ;SAVE QUOTIENT |
| FABE | pula | | | ;GET MSB |
| FABF | ins | | | ;IGNORE LSB |
| FAC0 | LFAC0:staa | L0287 | | ;SAVE |
| FAC3 | ldx | #0x867A | | ; |
| FAC6 | jsr | L99D7 | | ;2D LOOKUP |
| FAC9 | staa | L0471 | | ;STORE LOOKUP RESULT |
| FACC | ldab | L0286 | | ;LOAD |
| FACF | mul | | | ;LOOKUP RESULT * L0286 |
| FAD0 | staa | L0288 | | ;STORE MSB OF RESULT - 2'S COMP MAX SPARK |
| FAD3 | clr | L0289 | | ;CLEAR SPARK RET TERM |
| FAD6 | jmp | LFADF | | ;RETURN |

```

FAD9  LFAD9:clr      L0289                ;CLEAR TORQUE MGMT SPARK RET TERM
FADC          bclr   *L009E,#0x08        ;CLEAR BIT 3
FADF  LFADF:rts

;-----
; PART OF SENSOR CHECK
;-----

FAE0  LFAE0:ldaa     #0x40                ;LOAD CH #4 - TRANS TEMP
FAE2          jsr     L9A18                ;A/D READ
FAE5          cmpa    #0x28                ;40
FAE7          bcs     LFAEB                ;BRANCH IF < 40, ELSE
FAE9          adda    #0x9C                ;ADD 156
FAEB  LFAEB:rts      ;RETURN

;-----
; JUMP HERE IF BIT 4 L0085 SET - MULTI SENSOR FAILURE
;-----

FAEC  LFAEC:lds      #0x03FF              ;STACK ADDRESS
FAEF          ldab     L4006                ;CPU CAPT REG
FAF2          addb     #0xCD                ;ADD 205 = 6.25 MSEC
FAF4          stab     L4006                ;STORE CPU CAPT REG
FAF7          jsr     LFAE0                ;? GO READ TRANS TEMP ?
FAFA          bcs     LFB47                ;GO WAIT FOR NEXT IRQ
                                           ;CARRY IS SET WHEN TRANS A/D < 40 OR > 100

FAFC          ldd      #0xFF00              ;
FAFF          std      L400B                ;COP ARM/WATCHDOG
FB02          ldab     L0248                ;
FB05          cmpb     #0x02                ;2 ?
FB07          beq     LFB11                ;BRANCH IF 2, ELSE
FB09          cmpb     #0x03                ;3 ?
FB0B          bne     LFB1B                ;BRANCH IF NOT 3, ELSE
FB0D          brclr   *L007F,#0x01,LFB3C    ;
FB11  LFB11:bset     *L007F,#0x01          ;
FB14          ldx      L0226                ;ADDRESS OF MESSAGE BUFFER
FB17          jsr      0x00,x                ; Undetermined Branch Address
FB19          bra     LFB47                ;GO WAIT FOR NEXT IRQ

FB1B  LFB1B:ldab     L0248                ;? DEVICE ID CODE
FB1E          cmpb     #0x01                ;1 ?
FB20          bne     LFB42                ;BRANCH IF NOT 1, ELSE
FB22          bclr    *L007F,#0x01          ;CLEAR BIT 0
FB25          ldx      L0226                ;INDEX MESSAGE BUFFER
FB28          ldy      #0x0228                ;INDEX MESSAGE BUFFER + 2
FB2C          ldab     #0x20                ;32 LOOPS
FB2E  LFB2E:ldaa     0x00,y                ;
FB31          staa     0x00,x                ;
FB33          iny      ;
FB35          inx      ;
FB36          decb     ;
FB37          bne     LFB2E                ;LOOP TILL DONE
FB39          com      L0248                ;INVERT
FB3C  LFB3C:sei      ;HOLD INTERRUPTS
FB3D          jsr      L9A4A                ;I/O ROUTINE
FB40          cli      ;CLEAR AND ALLOW INTERRUPTS
FB41          nop      ;
FB42  LFB42:sei      ;HOLD INTERRUPTS
FB43          jsr      L9A4A                ;I/O ROUTINE
FB46          cli      ;CLEAR AND ALLOW INTERRUPTS
FB47  LFB47:bra      LFB47                ;WAIT HERE FOR NEXT IRQ

;-----

```

MAT TABLE FOR 1K PULLUP - FROM ANHT HAC - VALUES DIFFER

| | HEX | DEC | TEMP |
|------|------|-----|------|
| FB49 | 0x00 | 00 | |
| | 0x15 | 21 | |
| | 0x26 | 38 | |
| | 0x32 | 50 | |
| | 0x3B | 59 | |
| | 0x44 | 68 | |
| | 0x4B | 75 | |
| | 0x53 | 83 | |
| | 0x5A | 90 | |
| | 0x62 | 98 | |
| | 0x6A | 106 | |
| | 0x73 | 115 | |
| | 0x7E | 126 | |
| | 0x8B | 139 | |
| | 0x9D | 157 | |
| | 0xBE | 190 | |
| | 0xFF | 255 | |

```

;-----
; TABLES USED FOR CALCULATING CTS, DEFAULT AND MALF
; TABLE VALUES ARE IDENTICAL BETWEEN $8F, $A1, $DF AND $8D;
; ($A1 AND $DF HAVE THE MAT TABLE AFTER THESE)
;-----
; COOLANT VARIABLE TABLE #1, 4K PULL UP
; COOL = ((DEG C)+40)*(256/192)
;-----

```

| | ;HEX | DEC | TEMP DEG C | A/D |
|------|------|------|------------|-----|
| FB5A | FF, | ;255 | VERY HOT | 0 |
| | D7, | ;215 | 121 | 6 |
| | 9B, | ;155 | 76 | 22 |
| | 84, | ;132 | 59 | 38 |
| | 75, | ;117 | 48 | 54 |
| | 6A, | ;106 | 39 | 70 |
| | 61, | ;97 | 32 | 86 |
| | 58, | ;88 | 26 | 102 |
| | 51, | ;81 | 21 | 118 |
| | 4A, | ;74 | 15 | 134 |
| | 43, | ;67 | 10 | 150 |
| | 3C, | ;60 | 5 | 166 |
| | 34, | ;52 | -1 | 182 |
| | 2C, | ;44 | -7 | 198 |
| | 22, | ;34 | -14 | 214 |
| | 16, | ;22 | -24 | 230 |
| | 00 | ;00 | -40 | 246 |

```

;-----
; COOLANT VARIABLE TABLE #2, 384 OHM TBL - ALSO USED FOR TRANS TEMP SENSOR CONVERSION
; COOL = ((DEG C)+ 40)*(256/192)
;-----

```

| | ;HEX | DEC | TEMP DEG C | A/D |
|------|------|------|------------|-----|
| FB6B | FF, | ;255 | VERY HOT | 0 |
| | FF, | ;255 | VERY HOT | 16 |
| | F9, | ;249 | 147 | 32 |
| | DF, | ;223 | 127 | 48 |
| | CD, | ;205 | 114 | 64 |
| | BF, | ;191 | 103 | 80 |
| | B2, | ;178 | 94 | 96 |
| | A7, | ;167 | 85 | 112 |

| | | | |
|-----|------|-----------|-----|
| 9D, | ;157 | 78 | 128 |
| 93, | ;147 | 70 | 144 |
| 89, | ;137 | 63 | 160 |
| 7F, | ;127 | 55 | 176 |
| 75, | ;117 | 47 | 192 |
| 68, | ;104 | 38 | 208 |
| 59, | ;89 | 27 | 224 |
| 43, | ;67 | 10 | 240 |
| 00 | ;00 | VERY COLD | 256 |

2D TABLE VS L008D - FOR FUEL INJECTORS ?

```
-----
FB7C  0x88
      0x98
      0xA8
      0xB8
      0xC8
      0xF8
      0xF8
      0xF8
```

2D TABLE VS L02FC - RELATED TO FUEL INJECTOR OUTPUTS MALF

```
FB84  0x80
      0x40
      0x20
      0x10
      0x08
      0x04
```

```
FB8A  0x04
FB8B  0x05
FB8C  0x06
FB8D  0x01
      0x02
      0x03
      0x04
```

PRNDL SWITCH TRUTH TABLE

```
FB91  0x05,0x04,0x04,0x02,0x04,0x06,0x01,0x04,0x04,0x04,0x03,0x04,0x07,0x04,0x04,0x04
```

```
FBA1  0x08,0x11,0x11,0x40,0x11,0x04,0x80,0x11,0x01,0x10,0x20,0x11,0x02,0x11,0x11,0x11
```

```
FBF1  0x0F,0x0E,0x0F,0x0E,0x0E,0x0E,0x0E,0x0F,0x0E,0x0F,0x0C,0x0C,0x0C,0x0C,0x0F,0x0E,0x0F
```

```
FBC2  0x08,0x08,0x0C,0x0C,0x0F,0x0E,0x0F,0x00,0x08,0x0C,0x0C
```

TRANNNY SOLENOID TRUTH TABLE

| | | | |
|------|------|-----------|-----------------------|
| FBCD | 0x03 | %00000011 | ;1 ST GEAR |
| FBCE | 0x02 | %00000010 | ;2 ND GEAR |
| FBCF | 0x00 | %00000000 | ;3 RD GEAR |
| FBD0 | 0x01 | %00000001 | ;4 TH GEAR |

```
FBD1  0x00 - NOT USED
```

```
FBD2      .byte  0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
FBDA      .byte  0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
FBE2      .byte  0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
FBEA      .byte  0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
FBF2      .byte  0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
FBFA      .byte  0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
FC02      .byte  0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
```

[illegible]

;-----

| | | |
|------|------|---------------------------------------|
| FFDA | 9A70 | ;I/O ROUTINE |
| FFDC | F7EC | ;ADD UP MALF FLAGS AND STORE IN L0024 |
| FFDE | 0087 | ;? MISFIRE LOGIC LOOP COUNTER |
| FFE0 | 0204 | ;INPUT MESSAGE BUFFER |
| FFE2 | 0089 | ;? SCISDO |
| FFE4 | 9A4A | ;I/O ROUTINE |
| FFE6 | 9A53 | ;I/O ROUTINE |
| FFE8 | 9A18 | ;A/D READ ROUTINE |
| FFEA | 9A9B | ;AN I/O ROUTINE |
| FFEC | 0000 | ; |
| FFEE | 9324 | ;ALDL LIST |
| FFF0 | 9D58 | ;SWI VECTOR |
| FFF2 | 9F24 | ;INTERNAL IRQ VECTOR |
| FFF4 | CD02 | ;EXTERNAL IRQ VECTOR |
| FFF6 | 9D5E | ;ILLEGAL OPCODE RESET VECTOR |
| FFF8 | 9D62 | ;BUS RESET VECTOR |
| FFFA | 9D66 | ;COP WATCHDOG RESET VECTOR |
| FFFC | 9D6A | ;CLOCK MONITOR RESET VECTOR |
| FFFE | 9D6E | ;RESET VECTOR |
| 0000 | ; | .end |