How to tune topo motin parameters
File is tune.top
MAE 9-7-83

All of the parameters for the PID are located in the fil motor4ab1. The parameters are 16 bit values that are interpreted as xx.yy , that is xx is the first byte and yy is the fisrt byte to the right of the decimal point. In other words 1 1/4 may be represented as 01 40 hex.

the names of the parameters are :

KS	S overall gain
KIS	s integral value
KDS	S differential value
KO	Theta overall gain
KIO	Theta integral vlaue
KDO	Theta diff. value

The values are assembles into the code space by the %CDS macro. Its inputs are the name , the first byte value and the second byte value.

In addition to the above constants, there are some conversion constants. They are:

KW	converts from wheel sods to s form
KT	converts from wheel speeds to Theta from
IKW	converts from s form to wheel speed
IKT	converts from Theta form to wheel speed
Kclick	converts from click to wheel speeds

The above constants are used for speed measure and duty cycle calculation. KW and KT are used by SPDM/es in motor4.ab1. IKW and IKT are used used to convert the output of the sequencer from s.thets to wheel speed valus

In additon to the above, some of the constants used by the sequencer for accel/decell ramping may be found in SEQ.AS1

Directory of files for topo on the MD5
File is topodir.doc

File	Contains
Mice.csd	Automatic ICE-51 loader loads Mike.mac and mike7
Mike.mac	Macros for debugging ice
Mike7	Current ice loadable file for motors soft.
Tasm.csd	Command file to assemble portions of Motor soft. Also can link (Mopo.csd) and load (Mice.csd)
Mopo.csd	Envokes r151 to link object modules
Motcpu.a51	Contains: all interupt servicew routines contains command decoder contains all hardware init stuff contains all software init except Initparameters which inits the sequencer
Motor.equ	include file for motcpu.a51
Topo.mac	has macros for external data movement plus math
Motor.a51	HAs the sped measurement and PID motor stuff
MAth51.a51	Has 16 and 24 bit math used by motor stuff
SEq.a51	Conatains: Initparamters — init s motion stuff seq — motion que and sequencer Check — verify motion command for legalness Joy — joy stick command parser Move- move commands parser SEt-velocity
Blend.p51	set up blended queue entries
Motion.p51	decide how to set motion queue entry
Exact.p51	set up exact mode queue entry

```
ISIS-II MCS-51 MACRO ASSEMBLER V2.1
OBJECT MODULE PLACED IN :F0:DECOD4.OBJ
ASSEMBLER INVOKED BY:  ASM51 DECOD4.A51 DEBUG


LOC  OBJ           LINE     SOURCE

                     1     $PL(59)
                     2     $NOSB
                     3     $EP(DECOD4.ERR)
                     4     $TT(TOPO DECODER        DECOD4.A51              MAE     8-04-83)
                     5             NAME    DECOD
                     6 +2                  ; IF TRUE THEN USING OLD HARDWARE
                     7 +2                  ; IF 0 THEN NOT TESTING THE IR CHANNEL
                     8                                             ; IF 1 THEN TESTING THE IRCHANNEL
                     9 +2                  ; IF 0 THEN NOT TESTING THE IR CHANNEL
                    10                                             ; IF 1 THEN TESTING THE IRCHANNEL
                    11 +2          ; IF 0 THEN NOT TESTING HIT SWITCHES
                    12 +1  $IC(TOPO.MAC)
               =1   13             EXTRN   CODE(ADD24,SUB24,ADD16,SUB16,MUL16,DIV16,SMUL16)
 0004          =1   14     Z3      EQU     4
               =1   15
 0005          =1   16     ZHI     EQU     5
 0005          =1   17     Z2      EQU     ZHI
               =1   18
 0006          =1   19     ZH      EQU     6
 0006          =1   20     Z       EQU     ZH
 0006          =1   21     Z1      EQU     ZH
               =1   22
 0007          =1   23     ZL      EQU     7
 0007          =1   24     Z0      EQU     ZL
               =1   25
 0008          =1   26     WH      EQU     8
 0008          =1   27     W       EQU     WH
 0009          =1   28     WL      EQU     9
               =1   29
 000A          =1   30     XH      EQU     0AH
 000A          =1   31     X       EQU     XH
 000B          =1   32     XL      EQU     0BH
               =1   33
 0008          =1   34     Y3      EQU     8
 0009          =1   35     Y2      EQU     9
 000A          =1   36     Y1      EQU     0AH
 000B          =1   37     Y0      EQU     0BH
               =1   38
 000C          =1   39     V1      EQU     0CH
 000D          =1   40     V0      EQU     0DH
 0008          =1   41     W24     EQU     8
 0008          =1   42     X24     EQU     0BH
 0005          =1   43     Z24     EQU     5
               =1   44
               =1   45     ;       MACROS
               =1   46     ;
               =1   47
               =1   48 +1  $EJECT
```

```
LOC  OBJ            LINE     SOURCE

                =1     49     ;
                =1     50     ; EXTERNAL DATA STORE MACROS
                =1     51     ;
                =1     52
                =1     53
                =1     54
                =1     55
                =1     56
                =1     57
                =1     58
                =1     59
                =1     60
                =1     61
                =1     62
                =1     63
                =1     64
                =1     65
                =1     66
                =1     67
                =1     68
                =1     69
                =1     70
                =1     71
                =1     72
                =1     73
                =1     74
                =1     75
                =1     76
                =1     77
                =1     78
                =1     79
                =1     80
                =1     81
                =1     82
                =1     83
                =1     84     ;R      EQU     021837H              ; 10.668*08000H
  29AC          =1     85     R       EQU     10668                ; 10.668*1000
  007D          =1     86     GR      EQU     125
  0010          =1     87     N       EQU     16
  0A68          =1     88     D       EQU     2667                 ; 26.67 * 100
                =1     89
                =1     90
                =1     91
                =1     92
                =1     93
                =1     94
                =1     95
                       96     ;
                       97     ; TOPO DEVICES
                       98     ;
  0020                 99     TOPDEV  EQU     00H+20H
  0030                100     COMDEV  EQU     00H+30H
  0040                101     MOTDEV  EQU     00H+40H
  0050                102     SNDEV   EQU     00H+50H
```

```
LOC  OBJ           LINE     SOURCE

 0060              103      HEADEV  EQU     00H+60H
                   104      ;
                   105      ; COMMANDS FOR TOPO
                   106      ;
 0020              107      STACMD  EQU     TOPDEV+0              ; GENERAL TOPO STATUS
 0024              108      MDLCMD  EQU     TOPDEV+4              ; SAY THE MODEL NUMBER
 0027              109      RESCMD  EQU     TOPDEV+7              ; GLOBAL HARDWARE RESET
 0028              110      SETPCMD EQU     TOPDEV+8              ; SET/CLR THE POWER FLAG
                   111
 0030              112      WHATCMD EQU     COMDEV+0              ; QUERY COMMAND
 0039              113      SETBCMD EQU     COMDEV+9              ; SET/CLR PUBLIC CHANNEL B
 003A              114      SETCCMD EQU     COMDEV+0AH            ; SET/CLR PUBLIC CHANNEL C
 003B              115      SETDCMD EQU     COMDEV+0BH            ; SET/CLR PUBLIC CHANNEL D
                   116
 0040              117      CURSCMD EQU     MOTDEV+0              ; GET CURRENT S
 0041              118      CUROCMD EQU     MOTDEV+1              ; GET CURRENT O
 0042              119      CRSOCMD EQU     MOTDEV+2              ; GET THE CURRENT S DOT, O DOT
 0044              120      MOVCMD  EQU     MOTDEV+4
 0045              121      ABTCMD  EQU     MOTDEV+5              ; ABORT MOTION COMMNAD IN QUEUE
 0046              122      WARMCMD EQU     MOTDEV+6              ; WARMTART THE MOTOR CP
 0047              123      MRESCMD EQU     MOTDEV+7              ; RESET THE MOTOR CP
 0048              124      SETSCMD EQU     MOTDEV+8
 0049              125      SETOCMD EQU     MOTDEV+9
 004A              126      SETVCMD EQU     MOTDEV+0AH            ; SET FORWARD VELOCITY
 004B              127      RAMPCMD EQU     MOTDEV+0BH            ; SET THE ACCEL RAMP CHARACTERISTIC
 004C              128      JOYCMD  EQU     MOTDEV+0CH            ; JOYSTICK COMMAND
 004D              129      DELYCMD EQU     MOTDEV+0DH            ; DELAY
                   130
 0057              131      SRESCMD EQU     SNDEV+7               ; RESET SPEECH DEVICE
 0058              132      HT1CMD  EQU     SNDEV+8               ; SET HEAD TONE 1
 0059              133      HT2CMD  EQU     SNDEV+9               ; SET HEAD TONE 2
 005A              134      HT3CMD  EQU     SNDEV+0AH             ; SET HEAD TONE 3
 005B              135      HT4CMD  EQU     SNDEV+0BH             ; SET HEAD TONE 4
 005C              136      SAYCMD  EQU     SNDEV+0CH
 005D              137      TONECMD EQU     SNDEV+0DH             ; SOUND OF MUSIC
                   138
 0064              139      LATCMD  EQU     HEADEV+4              ; LATH THE HEAD SWITCHES
 0068              140      AUTOCMD EQU     HEADEV+8              ; SET THE MODE OFF THE HEAD SWITCH DEV
                   141
                   142              PUBLIC  SEND?,RECV?,S512?,R512?,STABLE?,1N256?,1N32?,RECVBIT,SENDBIT?
                   143              PUBLIC  RSYNC?,PARITY?,SYNCERROR,PARITYERROR,SYNC?,CARRIER?
                   144              PUBLIC  L1NTVLO,L1NTVH1,RINTVLO,RINTVHI
                   145              PUBLIC  CURCHAR,SYNCOUNT,SYNCOUNT
                   146              PUBLIC  IBUFF,LCURINT,RCURINT
                   147              PUBLIC  RIGHTSPD,LEFTSPO,CURSTATE,STATCLK,LASTCHAR,LASTPTR
                   148              PUBLIC  LPOS,RPOS
                   149              PUBLIC  TIMINGERROR,TESTPIN
                   150              EXTRN   CODE(DAVE1SR,1NITPARAMETERS,1NITMOTOR)
                   151              EXTRN   CODE(JOY,MOVE_EXACT,MOVE_SMOOTH,SET_VELOCITY,STORZ16,MOV16Z)
                   152              EXTRN   XDATA(S,O,SWITCH,DOTSDSR,DOTSACT,DOTODSR,DOTOACT,TRACLOC)
                   153              EXTRN   BIT(HEAD?)
                   154              PUBLIC  START,XBUFF,NOMOTOR?,CLRSWTCH?
                   155      ; SEGMENT DEFINITIONS
                   156      ;
```

```
LOC  OBJ           LINE     SOURCE

                     157     PROG    SEGMENT         CODE INBLOCK
                     158     ISRDATA SEGMENT         DATA
                     159     ISR1DATA SEGMENT        DATA
                     160     ISR2DATA SEGMENT        DATA
                     161     ISR3DATA SEGMENT        DATA
                     162     MOTXDATA SEGMENT        XDATA
                     163     ISRBITS SEGMENT BIT
                     164     FRED    SEGMENT         CODE
                     165     STACK   SEGMENT         IDATA
----                 166             CSEG AT 2000H
2000                 167     NOCODE:         DS      0DFFFH          ; NO ROM AREA HERE
                     168 +1  $IC(EQU.A51)
                =1   169 +1  $IC(QUEUE.EQU)
                =2   170     ;
                =2   171     ; QUEUE STRUCTURE FOR MOTION STUFF
                =2   172     ;
  0040          =2   173     QLEN    EQU     64
                =2   174
  0000          =2   175     BITS    EQU     0               ;BYTE FOR BITS
                =2   176
  0001          =2   177     PARM1   EQU     1               ;WORD FOR S AT-SPEED DISTANCE, OR SLOW DELTA
  0003          =2   178     ODIS    EQU     3               ;WORD FOR O AT-SPEED DISTANCE
  0005          =2   179     DECEL   EQU     5               ;WORD FOR S OR O DECELERATION DISTANCE
  0006          =2   180     SVEL    EQU     6               ;BYTE FOR MAXIMUM S VELOCITY
  0007          =2   181     OVEL    EQU     7               ;BYTE FOR MAXIMUM O VELOCITY
                =2   182
  0008          =2   183     STRUCTSIZE  EQU (OVEL-BITS)+1           ;NUMBER OF BYTES IN EACH ENTRY
                =2   184
                =2   185     ;
                =2   186     ; STUFF FOR SEQ.A51 AND OTHERS TO DO WITH CURRENT MOTION PARAMETERS
                =2   187     ;
                =2   188
  0000          =2   189     DIS     EQU     0       ;WORD CONTAINING REQUESTED COMMAND DISTANCE
  0002          =2   190     VEL     EQU     2       ;WORD CONTAINING REQUESTED COMMAND VELOCITY
  0004          =2   191     SIGN    EQU     4       ;BYTE CONTAINING REQUESTED COMMAND SIGN
  0005          =2   192     ACCEL   EQU     5       ;BYTE CONTAINING ACCELERATION VALUE (CONSTANT)
  0006          =2   193     DISSOFAR   EQU  6       ;24-BIT CONTAINING DISTANCE REMAINING IN COMMAND, XX.X
  0009          =2   194     PHASE   EQU     9       ;BYTE INDICATING CURRENT PHASE OF COMMAND
  000A          =2   195     CLICKS  EQU     10      ;WORD WITH NUMBER OF CLICKS FROM ENCODER COUNTER
  000C          =2   196     DELTA   EQU     12      ;WORD CONTAINING ACCELERATION DELTA, 0.X (CONSTANT)
  000E          =2   197     DIS_CL  EQU     14      ;WORD CONTAINING DISTANCE PER CLICK 0.XX (CONSTANT)
  0010          =2   198     DOT     EQU     16      ;WORD WITH CURRENT VELOCITY FOR PWM DOER, X.X
  0012          =2   199     DIR     EQU     18      ;BYTE WITH SIGN OF CURRENT VELOCITY
  0013          =2   200     TARDOT  EQU     19      ;BYTE WITH TARGET VELOCITY (X.QT.QV OR X.VEL)
  0014          =2   201     TARDIR  EQU     20      ;BYTE WITH TARGET VEL SIGN (X.QT.QSIGN OR X.SIGN)
  0015          =2   202     CURDEL  EQU     21      ;WORD WITH CURRENTLY USED DELTA
  0017          =2   203     TARGETDIS EQU   23      ;WORD WITH CURRENT TARGET DISTANCE
                =2   204
  0019          =2   205     SLEN    EQU     (TARGETDIS+1-DIS)+1
  0019          =2   206     QLEN    EQU     SLEN
  0008          =2   207     H1LEN   EQU     (OVEL-BITS)+1
  0008          =2   208     H2LEN   EQU     H1LEN
                =1   209 +1
                =1   210 +1  ;
```

```
LOC  OBJ          LINE     SOURCE

                  =1   211 +1  ;
                  =1   212 +1  ;       NEW HARDWARE LOCATIONS
                  =1   213 +1  ;
  8000            =1   214 +1  PIA     EQU     8000H                ; THE PIA PORT FOR TOPO
  8000            =1   215 +1  PIACTL  EQU     PIA                  ; THE CONTROL REGISTER
  8001            =1   216 +1  PORTA   EQU     PIACTL+1             ;       ; PORT A
  8002            =1   217 +1  PORTB   EQU     PORTA+1              ; PORT B
  8003            =1   218 +1  PORTC   EQU     PORTB+1              ; PORT C
  0042            =1   219 +1  PIAINIT EQU     01000010B            ; STOP TIMER, PORT C-INPUT,
                  =1   220 +1                                       ; PORT B-OUTPUT, PORT A-INPUT
                  =1   221 +1  ;
                  =1   222 +1  ; PORT A ASSIGNMENTS ... INPUTS
                  =1   223 +1  ;
  8001            =1   224 +1  LEDPORT EQU     PORTA                ; LED IS TOP 4 BITS
  8001            =1   225 +1  CHANLOC EQU     PORTA                ; TOPO NUMBER SWITCHES PORT
  00F0            =1   226 +1  HITSW   EQU     0F0H                 ; SWITCH WRITE VALUE
  0080            =1   227 +1  HITSW3  EQU     80H                  ; HIT SWITCH 0 --PA7
  0040            =1   228 +1  HITSW2  EQU     40H                  ; HIT SWITCH 1 --PA6
  0020            =1   229 +1  HITSW1  EQU     20H                  ; HIT SWITCH 2 --PA5
  0010            =1   230 +1  HITSW0  EQU     10H                  ; HIT SWITCH 3 --PA4
                  =1   231 +1  ;SW3    EQU     8
                  =1   232 +1  ;SW2    EQU     4
                  =1   233 +1  ;SW1    EQU     2
                  =1   234 +1  ;SW0    EQU     1
  0000            =1   235 +1  PORTAINIT       EQU     000H         ; PORT A INIT VALUE
                  =1   236 +1  ;
                  =1   237 +1  ; PORT B ASSIGNMENTS ... OUTPUTS
                  =1   238 +1  ;
                  =1   239 +1  ; FOOT LEDS
  0080            =1   240 +1  LBCK    EQU     080H                 ; LEFT BACKWARD--PA3
  0040            =1   241 +1  LFWD    EQU     040H                 ; LEFT FORWARD --PA2
  0020            =1   242 +1  RFWD    EQU     020H                 ; RIGHT FORWARD--PA1
  0010            =1   243 +1  RBCK    EQU     010H                 ; RIGHT BACKWARD-PA0
                  =1   244 +1
  0008            =1   245 +1  ECHONOTRESET    EQU     8            ; RESET NOT PIN TO ECHO
  0004            =1   246 +1  AGCRESET        EQU     4            ; AGC RESET TO IR SENDER
  0002            =1   247 +1  SPCHDISAB       EQU     2            ; ENABLE NOT TO RS232
                  =1   248 +1  ;N.C.           EQU     1            ; NOT CONNECTED
  000A            =1   249 +1  PORTBINIT       EQU     SPCHDISAB+ECHONOTRESET  ; PORT B INIT VALUE
                  =1   250 +1  ;
                  =1   251 +1  ; PORT C ASSIGNMENTS ... INPUTS
                  =1   252 +1  ;
  0010            =1   253 +1  RTS     EQU     10H                  ; ECHO SPEECH GARBAGE
  0008            =1   254 +1  DTR     EQU     8                    ; ECHO DATA TERMINAL READY HANDSHAKE
  0000            =1   255 +1  PORTCINIT       EQU     00H          ; PORT C INIT VALUE
                  =1   256 +1
  8000            =1   257 +1  XRAM    EQU     8000H                ; START OF EXTERNAL RAM
  80FF            =1   258 +1  MAXXRAM EQU     80FFH                ; END OF RAM
  8100            =1   259 +1  MAXRAM  EQU     8100H
  0085            =1   260 +1  SENDBITLOC      BIT     T1           ; FOR NOW IT ACCESSIBLE
                  =1   261 +1
                  =1   262 +1
                  =1   263
                  =1   264 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                =1   265
                =1   266
                =1   267     ; equates and storage locations
                =1   268     ;
  0070          =1   269     ?DALLY  EQU     070H    ; DELAY START STATE
  0000          =1   270     OFF     EQU     0                       ; OFF FOR FLAGS
  0001          =1   271     ON      EQU     1                       ; ON FOR FLAGS
  007F          =1   272     STRTSTATE       EQU     7FH             ; START OF MOTOR COUNTDOWN STATE TIMER
  00FF          =1   273     STATSTRT        EQU     0FFH            ; START OF MOTOR STATE CLOCK
  0020          =1   274     ?ERRDELAY       EQU     020H            ; WAIT FOR IR CHANNEL AFTER ERROR
  0006          =1   275     BUFSIZ  EQU     06H                     ; SIZE OF IR SEND AND RECEIVE BUFFER BUT NO CHECKSU
                             M
  0020          =1   276     SPECBITS EQU    20H                     ; SPECIAL BITS OF BYTES
                =1   277
  0002          =1   278     ISRBANK EQU     2                       ; FOR USING STATEMENT
  0000          =1   279     PROGBANK EQU    0                       ; MAINLINE BANK
  0001          =1   280     MATHBANK EQU    1                       ; MATH REGISTERS
                =1   281
  0000          =1   282     BANK0   EQU     00000000B               ; SELECT THE NORMAL REGISTER BANK 0
  0008          =1   283     BANK1   EQU     00001000B               ; SELECT THE NORMAL  REGISTER BANK 1
  0010          =1   284     BANK2   EQU     00010000B               ; SELECT THE NORMAL REGISTER BANK 2
  0018          =1   285     BANK3   EQU     00011000B               ; SELECT THE NORMAL REGISTER BANK 3
  E000          =1   286     SDK     CODE    0E000H                  ; START OF SDK MONITOR
  0020          =1   287     CHARSTART EQU   020H             ; START AT CRET
  007E          =1   288     CHAREND EQU     07EH             ; END OF CHARS
  0000          =1   289     MATHBANK0       EQU     BANK0           ; USED FOR MATH STUFF
  0008          =1   290     MATHBANK1       EQU     BANK1
  0010          =1   291     ENCODBANK       EQU     BANK2           ; FIRST ENCODER ISR BANK
  0000          =1   292     MAINBANK        EQU     BANK0           ; THE MAINLINE REGISTER BANK
  0087          =1   293     PCON            EQU     087H            ; THE SERIAL TIMES 2 BIT
                =1   294     ;
  000D          =1   295     CR      EQU     0DH                     ; CAARRIAGE RETURN
  000A          =1   296     LF      EQU     0AH                     ; LINEFEED
  0020          =1   297     BLANK   EQU     020H                    ; BLANK
  0005          =1   298     NUMBEROFSYNCS EQU 5                     ; NUMBER OF SYNCS TO SEND
  007F          =1   299     STARTCHAR EQU   07FH                    ; SET UP TO RECIEVE A CHAR
  0003          =1   300     MIDSYNC EQU     03H                     ; THIS IS WHEN SYNC BITS SHOULD BE STABLE
  0005          =1   301     MINSYNC EQU     05H                     ; MIN NUMBER OF SYNCS BEFORE START BIT
  0050          =1   302     TOPOMSG EQU     'P'                     ; STRART OF LONG PACKET RESPONSE
  0051          =1   303     ACK0    EQU     'Q'                     ; 1 TYPE OF ACK
  0052          =1   304     ACK1    EQU     'R'
  0010          =1   305     NAK     EQU     10H
  0000          =1   306     NULL    EQU     00H                     ; THE LAST CHAR NOT TO SEND
  0002          =1   307     SOM     EQU     02                      ; START OF MESSAGE
  000A          =1   308     EOM     EQU     LF                      ; ENDO OF MESSAGE
  0003          =1   309     EOT     EQU     03H                     ; END OF TRANSMISSION
                =1   310     ;
  0052          =1   311     SERIALMODE      EQU     01010010B       ; MODE 1 8BIT VARIABLE BAUD RATE
  00F3          =1   312     BAUDRATE        EQU     0F3H            ; 2400 BAUD SERIAL PORT
  0020          =1   313     T1INIT  EQU     20H                     ; TIMER 1 INIT : MODE 2;NO GATE;TIMER
  0003          =1   314     T0INIT  EQU     03H                     ; TIMER 0 INIT : MODE 0;NO GATE;TIMER
  00FF          =1   315     RESETP1 EQU     0FFH                    ; PROGRAM PORT 1 FOR ALL INPUT
  0001          =1   316     TIMESTART EQU   01H                     ; THE START OF THE HIGH BYTE TIMER
                =1   317
```

```
LOC  OBJ          LINE     SOURCE

 008C          =1   318     ENCODTIME EQU   TH0              ; ENCODER TIMER IS TH0
               =1   319     ;UNUSED EQU     TH1              ; UNUSED FOR NOW
               =1   320     ;
 0090          =1   321     RIGHTLATCH BIT  P1.0             ; LATCH FOR THE RIGHT ENCODER
 0091          =1   322     RLATCHRESET BIT P1.1             ; RESET FOR THE RIGHT LATCH
 0092          =1   323     LEFTLATCH BIT   P1.2             ; LATCH FOR LEFT ENCODER
 0093          =1   324     LLATCHRESET BIT P1.3             ; RESET PIN FOR THE LEFT ENCODER LATCH
 0095          =1   325     RDIROUT BIT     P1.5             ; RIGHT MOTOR DIRECTION CONTROL 1=FWD
 0094          =1   326     RENABLE BIT     P1.4             ; RIGHT MOTOR ENABLE
 0097          =1   327     LDIROUT BIT     P1.7             ; LEFT MOTOR DIRECTION CONTROL 0=FWD
 0096          =1   328     LENABLE BIT     P1.6             ; LEFT MOTOR ENABLE
 00B5          =1   329     TESTPIN BIT     T1               ; GENERAL PURPOSE TEST PIN
               =1   330 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                =1   331     ?CHANUM EQU     0                ; DEVICE NUMBER
  0001          =1   332     ?DEVNUM EQU     1                ; OFFSET IN IRCVBUFF OF DEVICE NUMBER
  0001          =1   333     ?CMDNUM EQU     1                ; " COMMAND
  0002          =1   334     ?PARAM1 EQU     2                ; OFFSET OF FIRST HEX NUMBER
  0004          =1   335     ?PARAM2 EQU     4                ; SAME AS ABOVE FOR #2
  0006          =1   336     ?CHCKSUM EQU    6                ; CHECK SUM OF MESSAGE
  0030          =1   337     FIRSTCHAN       EQU     '0'      ; FIRST PRIVATE CHAN NUMBER
  0041          =1   338     PUBCHAN1        EQU     'A'      ; FIRST PUBLIC CHANNEL
  0042          =1   339     PUBCHAN2        EQU     'B'      ; FIRST PUBLIC CHANNEL
  0001          =1   340     PUBB    EQU     1
  0002          =1   341     PUBC    EQU     2
  0004          =1   342     PUBD    EQU     4
  0046          =1   343     FALSE   EQU     'F'
  0054          =1   344     TRUE    EQU     'T'
  0080          =1   345     TALKING EQU     080H             ; STATUS...ONIF YAKKING
  0040          =1   346     MOVING  EQU     040H             ; STATUS...ON IF MOVING
  0020          =1   347     TONING  EQU     020H             ; EMITTING TONE
  0010          =1   348     POWERON EQU     010H             ; POWER ON FLAG
  00E4          =1   349     GOTSWT  BIT     ACC.4            ; BIT LOC FOR SWTCH STAT
                =1   350
  0002          =1   351     ?CHAR1  EQU     ?DEVNUM+1        ; FIRST STAT CHAR
  0003          =1   352     ?CHAR2  EQU     ?CHAR1+1         ; SECOND
  0004          =1   353     ?CHAR3  EQU     ?CHAR2+1
  0005          =1   354     ?CHAR4  EQU     ?CHAR3+1
                     355 +1
----                 356 +1           XSEG    AT 0
0000                 357 +1  NORAM:   DS      8000H            ; XRAM NOT AT 0-7777H
                     358 +1
                     359
                     360
----                 361             RSEG    STACK
0000                 362             DS      1
                     363     ;
                     364 +1  $EJECT
```

```
LOC  OBJ           LINE     SOURCE

                    365
----                366             RSEG    ISRBITS
                    367
0000                368     LOVR?:  DBIT    1               ; LEFT TIMER OVERFLOW FLAG
0001                369     ROVR?:  DBIT    1               ; RIGHT "
0002                370     RECVBIT:        DBIT    1               ; THE IR RECIEVED BIT
0003                371     SENDBIT?:       DBIT    1               ; THE SEND A BIT FLAG FOR IR
0004                372     RECV?:  DBIT    1               ; RECEIVE FALG FOR GETTING A PACKET
0005                373     SEND?:  DBIT    1               ; FLAG FOR SENDING A PACKET
0006                374     S512?:  DBIT    1               ; FALG FOR 512 MICRSECOND IR SEND BIT
  0006              375     R512?:  BIT     S512?   ; FALG FOR 512 MICROSECOND IR RECV BIT
0007                376     STABLE?:        DBIT    1               ; FLAG TRUE SAYS THE SYNC PULSES SHOULD BE STABLE
0008                377     SYNC?:  DBIT    1               ; FLAG TRUE SAYS THE WE ARE SENDING A SYNC
  0008              378     RSYNC?: BIT     SYNC?   ; FLAG TRUE SAYS WE ARE RECIEVING A SYNC
0009                379     SPAR?:  DBIT    1               ; STORAGE FOR THE PARITY BIT TO SEND
                    380
000A                381     PARITY?:        DBIT    1               ; FLAG TRUE SAYS NEXT BIT IS PARITY
000B                382     PARITYERROR:    DBIT    1               ; FLAG TRUE SAYS WE HAD A PARITY ERROR IN RECIEVE
000C                383     SYNCERROR:      DBIT    1               ; FLAG TRUE IF BAD SYNC
000D                384     SPARBIT:        DBIT    1               ; SEND THE PARITY BIT OF THE LAST CHAR
000E                385     READBIT:        DBIT    1               ; THIS IS THE BIT SET BY THE RECEVIE ISR
000F                386     ACK1?:  DBIT    1               ; FLAG TRUE SAYS SEND ACK0 ELSE SEND ACK
                    387
0010                388     IN256?: DBIT    1               ; FLAG TRUE SAYS WE ARE IN THE 256 US ISR
0011                389     IN32?:  DBIT    1               ; FLAG TRUE SAYS WE ARE IN THE 32 MS ISR
0012                390     NOACK?: DBIT    1               ; TRUE IF NOT TO ACKNOWLEGDE IR
0013                391     NOMOTOR?:       DBIT    1               ; TRUE ENABLES MOTOR TO MOVE
0014                392     CLRSWTCH?: DBIT         1       ; TRUE IF ISR SHOULD CLEAR SWITCHES
0015                393     POWER?:         DBIT    1       ; CLR ON POWER UP
0016                394     SHORT?: DBIT    1               ; TRUE IF JUST ACK IS RESPONSE ELSE LONG RESP.
0017                395     QUERY?: DBIT    1               ; TRUE IF RESPONSE TO QUERY
0018                396     CARRIER?:       DBIT    1       ; TRUE IF TOPO NOT RECEVIING CARRIER
                    397 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                    398             RSEG    ISRDATA
0000                399     SYNCOUNT:       DS      1               ; THE COUNT FOR THE NUMBER OF SYNCS TO SEND
0001                400     1BUFF:  DS      BUFSIZ+1                ; BUFFER TO GET AND SEND CHARS FROM 1R
  0007              401     LASTCHAR        EQU     $-1             ; THE LAST CHAR TO SEND +1
0008                402     LASTPTR:        DS      1               ; THE LOCATION OF THE LAST CHAR TO SEND
0009                403     CURCHAR:        DS      1               ; NEXT CHAR TO SEND
                    404             USING   ISRBANK
  REG               405     R1PTR           SET     R1              ; CURRENT LOCATION OF CHAR TO SEND
  0011              406     AR1PTR          EQU     AR1
----                407             RSEG    ISR2DATA
0000                408     LOLD:   DS      1                       ; START COUNT OF LEFT ENCODER
  REG               409     R2LNEW  SET     R2                      ; LOW " THE INT NEW TIME
                    410     ;LNEW:  DS      1                       ; THE LAST GOTTEN NEW TIME
                    411     ;LSTIME:        DS      1               ; LEFT 8 BIT SOFTWARE TIMER
  REG               412     LSTIME  SET     R4                      ; LEFT 8 BIT SOFTWARE TIMER
                    413
0001                414     ROLD:   DS      1                       ; START COUNT OF RIGHT ENCODER
  REG               415     R3RNEW  SET     R3                      ; LOW "
                    416     ;RNEW:  DS      1                       ; THE LAST GOTTEN ONE
                    417     ;RSTIME:        DS      1                       ; LEFT 8 BIT SOFTWARE TIMER
  REG               418     RSTIME  SET     R5                      ; LEFT 8 BIT SOFTWARE TIMER
                    419     ;
----                420             RSEG    ISR3DATA
0000                421     RIGHTSPD:       DS      1               ; NUMBER OF STATES OF 7F STATES TO OUTPUT TO ENABLE
                             MOTOR DRIVERS
0001                422     LEFTSPD:        DS      1               ; SAME FOR LEFT MOTORS
0002                423     CURSTATE:       DS      1               ; CURRENT MOTOR STATE
0003                424     STATCLK:        DS      1               ; MOTOR STATE CLOCK
0004                425     LPOS:           DS      1                       ; CURRENT POS FOR LEFT
0005                426     RPOS:           DS      1                       ; SAME FOR RIGHT MOTOR
0006                427     RSPTR:  DS      1               ; POINTER TO RS232 BUFF
                    428
----                429             RSEG    ISR1DATA
0000                430     LINTVHI:        DS      1
0001                431     LINTVLO:        DS      1
0002                432     RINTVHI:        DS      1
0003                433     RINTVLO:        DS      1
                    434     ;
0004                435     LCURINT:        DS      2               ; THE CURRENT LEFT INTERVAL
0006                436     RCURINT:        DS      2               ; SAME FOR LEFT
                    437
                    438     ;
----                439             RSEG    MOTXDATA
0000                440     PORTBMASK:      DS      1               ; PORT B MASK
0001                441     SWLOCLAT: DS    1                       ; LOC FOR HEAD SWITCH LATCH
0002                442     PUBLOC: DS      1                       ; LOC FOR PUBLIC CHANNEL ON BITS
0003                443     XBUFF:  DS      BUFSIZ
  0009              444     XBUFFEND        EQU     $
                    445 +1
0009                446 +2  SCUR:   DS      1
000A                447 +1          DS      1
                    448
                    449 +1  $EJECT
```

```
LOC  OBJ            LINE     SOURCE

                     450        RSEG PROG
                     451        USING   PROGBANK
                     452     ;
                     453     ; SUBROUTINES FOR THE REST OF THE CODE
                     454     ;
                     455     ; DECODE
                     456     ;       THIS ROUTINES JMPS TO AN ADRRESS BASED ON THE CONTENTS OF ACC AND
                     457     ;       THE DPTR.
                     458     ;       INPUTS: DPTR POINTS TO A TABLE CONTAINING THE ASCII CODE AND OFFSET
                     459     ;                       OF A GIVEN COMMAND
                     460     ;               ACC HAS THE CHARACTER TO DECODE
                     461     ;
                     462     ;       OUPUTS: IF ENTRY EXISTS THEN ROUTINE JMPS TOP APPRORIATE ROUTINE
                     463     ;               ELSE ROUTINE RETURNS.
                     464     ;
                     465     ;       ALTERS: R0,ACC,DPTR
                     466     ;
                     467     ;       NOTE:   THIS ROUTINE SHOULD BE JUMPED TO NOT CALLED
                     468     ;               ALSO, A 0 ENTRY AS A COMMAND IN A TABLE IS THE END OF TABLE
                     469     DECODE:
0000 F8              470             MOV     R0,A                    ; SAVE CODE
                     471     CLOOP:
0001 E4              472             CLR     A                       ; SET UP FOR FETCH FROM CODE
0002 93              473             MOVC    A,@A+DPTR               ; FETCH NEXT COMMAND CHAR
0003 B50004          474             CJNE    A,AR0,CLOOP1            ; IF NOT MATH THEN GO
0006 A3              475             INC     DPTR                    ; ELSE GOT CMD: FETCH OFFSET
0007 E4              476             CLR     A
0008 93              477             MOVC    A,@A+DPTR               ;       FETCH OFFSET
0009 73              478             JMP     @A+DPTR                 ;       AND AWAY WE GO
                     479     CLOOP1:
000A A3              480             INC     DPTR                    ; POINT TO NEXT COMMAND
000B A3              481             INC     DPTR
000C B400F2          482             CJNE    A,#0,CLOOP              ; IF NOT END OF LOOP THEN CONT.
000F 22              483             RET                             ; ELSE RETURN TO CALLER
                     484 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                     485     ;
                     486     MOVTOXBUFF:
0010 E500      F     487             MOV     A,1BUFF+7DEVNUM ; IF COMMAND 15 SAYWHAT THEN DON'T MOVE
0012 B43001          488             CJNE    A,#WHATCMD,MOVIT
0015 22              489             RET
                     490     MOVIT:
0016 900000    F     491             MOV     DPTR,#XBUFF
0019 7906            492             MOV     R1,#BUFSIZ
001B 7800      F     493             MOV     R0,#IBUFF
                     494     MOV1:
001D E6              495             MOV     A,@R0
001E C2E7            496             CLR     ACC.7
0020 08              497             INC     R0
0021 F0              498             MOVX    @DPTR,A
0022 A3              499             INC     DPTR
0023 D9F8            500             DJNZ    R1,MOV1
0025 22              501             RET
                     502     ;
                     503     MOVFROMX:
                     504
0026 900000    F     505             MOV     DPTR,#XBUFF
0029 7906            506             MOV     R1,#BUFSIZ
002B 7800      F     507             MOV     R0,#IBUFF
                     508     MOV2:
002D E0              509             MOVX    A,@DPTR
002E F6              510             MOV     @R0,A
002F 08              511             INC     R0
0030 A3              512             INC     DPTR
0031 D9FA            513             DJNZ    R1,MOV2
0033 22              514             RET
                     515 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                     516     ;*************************************************
                     517     ;       MAIN ENTRY POINT FOR DECODER LEVEL      :
                     518     ;*************************************************
                     519     START:
0034 1100      F     520             ACALL   SPCHOFF                 ; NO TALKING YET
0036 900000    F     521             MOV     DPTR,#?CLEAR
0039 1100      F     522             ACALL   SHOLOOP
003B 1100      F     523             ACALL   SPCHON
                     524
003D 7830            525             MOV     R0,#030H
003F 1100      F     526             ACALL   LDELAY
                     527 +1
0041 900000    F     528             MOV     DPTR,#?MESS
0044 1100      F     529             ACALL   SHOLOOP
                     530 +1
0046 1100      F     531             ACALL   SPCHOFF
0048 750000    F     532             MOV             RSPTR,#LOW XBUFFEND
                     533     DECOD:
004B D200      F     534             SETB    SHORT?                   ASSUME SHORT RESPONSE UNLESS TOLD
004D 1100      F     535             ACALL   WAITMESS
004F 40FA            536             JC      DECOD                   ; IF ERROR THEN WAIT FOR MORE
                     537
0051 C0E0            538             PUSH    ACC                     ; SAVE COMMAND CHAR
0053 1100      F     539             ACALL   SPCHOFF                 ; DISABLE SPEECH
0055 D0E0            540             POP     ACC                     ; RESTORE COMMAND CHAR
0057 C2E7            541             CLR     ACC.7                   ; MASK HI BIT
                     542
0059 750000    F     543             MOV     RSPTR,#LOW XBUFF        ; POINT TO START OF MESSAGE FOR OUTPUT
                     544     DOCMD:
005C 900000    F     545             MOV     DPTR,#DEVTAB            ; POINT AT CMD TABLE WITH A+DPTR
005F 1100      F     546             ACALL   DECODE                  ; DECODE COMMAND
0061 200002    F     547             JB      NOACK?,DECODEX
0064 1100      F     548             ACALL   SENDIT
                     549     DECODEX:
0066 80E3            550             SJMP    DECOD                   ; GO DO NEXT
                     551     ;
                     552     ; SENDIT
                     553     ;
                     554     SENDIT:
0068 300002    F     555             JNB     QUERY?,NODOQ            ; IF TO SEND QUERY THEN DOIT
006B 1100      F     556             ACALL   MOVFROMX                ; ELSE MOV QUERY ANSWER FORM XBUFF
                     557     NODOQ:
                     558     ;       GET THE NEXT ACK TO SEND INTO A AND TOGGLE FLAG
                     559     ;
006D 7452            560             MOV     A,#ACK1
006F 100004    F     561             JBC     ACK1?,GTA1              ; IF TRUE THEN SEND ACK0
0072 D200      F     562             SETB    ACK1?                   ; ELSE SEND ACK1
0074 7451            563             MOV     A,#ACK0
                     564     GTA1:
0076 200007    F     565             JB      SHORT?,SENDSHT          ; IF NOT SHORT THEN OKAY TO SEND
0079 F500      F     566             MOV     IBUFF+1,A               ; GET THE CORRECT ACK
007B 750050    F     567             MOV     IBUFF,#TOPOMSG
007E 8004            568             SJMP    SENDIT1
                     569     ;*************************************************
```

```
LOC  OBJ            LINE     SOURCE

                     570     ;
                     571     ;       ACKNOWLEDGE THAT WE GOT THE LAST MESS FROM IR ...SHORT FORM IR
                     572     ;
                     573     ;***********************************************
                     574     SENDSHT:
0080 1100      F     575             ACALL   CLEARBUFF               ; CLEAR THE IRSEND BUFF
0082 F500      F     576             MOV     IBUFF,A                 ; PUT INT THE BYTE TO SEND
                     577     SENDIT1:
0084 1100      F     578             ACALL   MOVTOXBUFF              ; SAVE FOR POSSIBLE QUERY LATER
0086 1100      F     579             ACALL   SENDSUM                 ; CALULATE THE CHECKSUM
0088 1100      F     580             ACALL   ENABLESEND              ; OFF IT GOES
008A 2000FD    F     581             JB      SEND?,$                 ; WAIT TILL ITS GONE
008D 22              582             RET
                     583 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                    584     ;***********************************************
                    585     ;
                    586     ; SENDSUM
                    587     ;       CALC THE CHECKSUM OF THE PACKET TO SEND AND PUT IN IBUFF
                    588     ;
                    589     ;***********************************************
                    590     SENDSUM:
008E 7900      F    591             MOV     R1,#IBUFF
0090 7A00           592             MOV     R2,#0                   ; CHECKSUM FOR IR XMIT
                    593     SS1:
0092 E7             594             MOV     A,@R1                   ; FETCH NEXT CHAR
0093 2A             595             ADD     A,R2
0094 FA             596             MOV R2,A
0095 09             597             INC     R1                      ; BUMP BUFF PTR
0096 B900F9    F    598             CJNE    R1,#LASTCHAR,SS1        ; IF NOT AT END OF BUFF THEN GO
0099 E4             599             CLR     A
009A C3             600             CLR     C
009B 9A             601             UBB     A,R2                    ; FIND 2'S
009C F7             602             MOV     @R1,A                   ;
009D 22             603             RET
                    604 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                     605     ;************************************************
                     606     ;
                     607     ; SUBROUTINES FOR DEVICE HANDLERS
                     608     ;
                     609     ; STP
                     610     ;       CONVERT CHAR POINT TO BY DPTR TO TWO ACSII CHARS POINTED BY R0
                     611     ;       INPUT:  @DPTR,R0
                     612     ;       OUTPUT: @R0
                     613     ;       ALTERS: A,DPTR=DPTR+1,R0=R0+1
                     614
                     615     ;************************************************
                     616     STP:
009E C200      F     617             CLR     SHORT?                  ; IF STORING PARAMS THEN IT'S A LONG MESSAGE
00A0 E0              618             MOVX    A,@DPTR
00A1 C4              619             SWAP    A                       ; DO HIGH BYTE FIRST
00A2 120000    F     620             CALL    CONVHEX                 ; CONVERT TO ASCII
00A5 F6              621             MOV     @R0,A                   ; STORE T
00A6 08              622             INC     R0                      ; POINT TO NEXT STORAGE
                     623
00A7 E0              624             MOVX    A,@DPTR                 ; GET LOW PART
00A8 120000    F     625             CALL    CONVHEX                 ; CONVERT TO ASCII
00AB F6              626             MOV     @R0,A
00AC 08              627             INC     R0                      ; POINT TO NEXT
00AD A3              628             INC     DPTR                    ; POINT OT NEXT TO CONVERT
00AE 22              629             RET
                     630     ;************************************************
                     631     ;
                     632     ; STORP:
                     633     ;       CONVERTS TWO HEX CHARS @DPTR TO 4 ASCII @R0
                     634     ;       INPUTS: @DPTR
                     635     ;       OUTPUTS:@R0
                     636     ;       ALTERS: A,DPTR,R0
                     637     ;************************************************
                     638     STORP:
00AF 7800      F     639             MOV     R0,#IBUFF+?PARAM1       ; POINT TO THE PARAM TO GET
00B1 1100      F     640             CALL    STP                     ; GET TWO HEX DIGITS
00B3 1100      F     641             CALL    STP                     ; GET TWO MORE
00B5 22              642             RET
                     643     ;
                     644     ; BASE DEVICE TABLE
                     645     ;
                     646     DEVTAB:
                     647     ; GENERAL TOPO COMMANDS
00B6 20              648 +2                  DB      STACMD,DOSTAT-$
00B7 2E
00B8 24              649 +2                  DB      MDLCMD,MODEL-$
00B9 2F
00BA 27              650 +2                  DB      RESCMD,DORESET-$
00BB 36
00BC 28              651 +2                  DB      SETPCMD,DOSETP-$
00BD 3D
                     652     ; COMMUNICATION CMMANDS
00BE 30              653 +2                  DB      WHATCMD,DOQUERY-$
00BF 49
```

```
LOC  OBJ           LINE     SOURCE

00C0 39             654 +2                  DB      SETBCMD,DOSETB-$
00C1 49
00C2 3A             655 +2                  DB      SETCCMD,DOSETC-$
00C3 5B
00C4 3B             656 +2                  DB      SETDCMD,DOSETD-$
00C5 6D
                    657     ; MOTION COMMANDS
                    658
00C6 40             659 +2                  DB      CURSCMD,DOCURS-$
00C7 7F
00C8 41             660 +2                  DB      CUROCMD,DOCURO-$
00C9 82
00CA 42             661 +2                  DB      CRSOCMD,DOCURSO-$
00CB 85
00CC 44             662 +2                  DB      MOVCMD,DOMOVE-$
00CD 95
00CE 47             663 +2                  DB      MRESCMD,DOMRESET-$
00CF 90
00D0 48             664 +2                  DB      SETSCMD,DOSETS-$
00D1 94
00D2 49             665 +2                  DB      SETOCMD,DOSETO-$
00D3 A0
00D4 4A             666 +2                  DB      SETVCMD,DOSETV-$
00D5 AC
00D6 4B             667 +2                  DB      RAMPCMD,DORAMP-$
00D7 BA
00D8 4C             668 +2                  DB     JOYCMD,DOJOY-$
00D9 BA
                    669     ; SPEECH COMMANDS
00DA 57             670 +2                  DB      SRESCMD,DOSRESET-$
00DB D6
00DC 5C             671 +2                  DB      SAYCMD,DOSPCH-$
00DD DF
                    672     ; HEAD COMMANDS
00DE 64             673 +2                 DB      LATCMD,DOLATCH-$
00DF E2
00E0 68             674 +2                  DB      AUTOCMD,DOAUTO-$
00E1 E8
00E2 00             675 +2                  DB      0,DEVTABEND-$
00E3 01
                    676 +2  DEVTABEND:
00E4 22             677 +1          RET
                    678 1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                    679     ;*************************************************
                    680     ;
                    681     ;       GENERAL TOPO TYPE COMMANDS
                    682     ;*************************************************
                    683     ;
                   684     ; DOSTAT
                    685     ;
                    686     DOSTAT:
00E5 020000    F    687             JMP     STATL
                    688
                    689     ;*************************************************
                    690     MOEL:
00E8 1100      F    691             ACALL   SPCHON
00EA 900000    F    692             MOV     DPTR,#?MODEL
00ED 1100      F    693             ACALL   SHOLOOP
00EF 0100      F    694             AJMP    SPCHOF?
                    695
                    696     ;*************************************************
                    697     ;
                    698     ; DORESET
                    699     ;
                    700     ;*************************************************
                    701     DORESET:
00F1 D200      F    702             SETB    NOMOTOR?                        ; DISABLE MOTORS
00F3 7830           703             MOV     R0,#030H                ; DELAY A WHILE
00F5 1100      F    704             ACALL   LDELAY
00F7 020000         705             JMP     RESET                   ; RESET THE SYSTEM
                    706
                    707
                    708     ;*************************************************
                    709     ;
                    710     ; DOSETP
                    711     ;
                    712     ;*************************************************
                    713     DOSETP:
00FA E500      F    714             MOV     A,IBUFF+?PARAM1 ; GET FLAG
00FC B45403         715             CJNE    A,#TRUE,DP1
00FF D200      F    716             SETB    POWER?
0101 22             717             RET
                    718     DP1:
0102 B44602         719             CJNE    A,#FALSE,DPEX
0105 C200      F    720             CLR     POWER?
                    721     DPEX:
0107 22             722             RET
                    723 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                     724     ;*************************************************
                     725     ;
                     726     ;       TOPO COMMUNICATION DEVICE COMMANDS
                     727     ;
                     728     ;*************************************************
                     729     ; DOQUERY
                     730     ;       RESPOND TO A QUERY
                     731     ;
                     732     ;*************************************************
                     733     DOQUERY:
0108 0100      F     734             AJMP    QUEERL
                     735
                     736
                     737     ;*************************************************
                     738     ;
                     739     ; DOSETB
                     740     ;       SET/CLR TOPO PUBLIC CHANNEL B
                     741     ;
                     742     ;*************************************************
                     743     DOSETB:
010A 900000    F     744             MOV     DPTR,#PUBLOC            ; GET THE PUBLIC BITS
010D E0              745             MOVX    A,@DPTR
010E A800      F     746             MOV     R0,IBUFF+?PARM1         ; FETCH 'T' OR 'F'
0110 B85404          747             CJNE    R0,#TRUE,DSB1
                     748
0113 4401            749             ORL     A,#PUBB
0115 F0              750             MOVX    @DPTR,A
0116 22              751             RET
                     752     DSB1:
0117 B84603          753             CJNE    R0,#FALSE,DSB2
011A 54FE            754             ANL     A,#NOT PUBB
011C F0              755             MOVX    @DPTR,A
                     756     DSB2:
011D 22              757             RET
                     758
                     759     ;*************************************************
                     760     ;
                     761     ; DOSETC
                     762     ;       SET/CLR TOPO PUBLIC CHANNEL C
                     763     ;
                     764     ;*************************************************
                     765     DOSETC:
011E 900000    F     766             MOV     DPTR,#PUBLOC            ; GET THE PUBLIC BITS
0121 E0              767             MOVX    A,@DPTR
0122 A800      F     768             MOV     R0,IBUFF+?PARM1         ; FETCH 'T' OR 'F'
0124 B85404          769             CJNE    R0,#TRUE,DSC1
                     770
0127 4402            771             ORL     A,#PUBC
0129 F0              772             MOVX    @DPTR,A
012A 22              773             RET
                     774     DSC1:
012B B84603          775             CJNE    R0,#FALSE,DSC2
012E 54FD            776             ANL     A,#NOT PUBC
0130 F0              777             MOVX    @DPTR,A
```

```
LOC  OBJ            LINE     SOURCE

                     778     DSC2:
0131 22              779             RET
                     780
                     781     ;*************************************************
                     782     ;
                     783     ; DOSETD
                     784     ;       SET/CLR TOPO PUBLIC CHANNEL D
                     785     ;
                     786     ;*************************************************
                     787     DOSETD:
0132 900000    F     788             MOV     DPTR,#PUBLOC            ; GET THE PUBLIC BITS
0135 E0              789             MOVX    A,@DPTR
0136 A800      F     790             MOV     R0,IBUFF+?PARM1         ; FETCH 'T' OR 'F'
0138 B85404          791             CJNE    R0,#TRUE,DSD1
                     792
013B 4404            793             ORL     A,#PUBD
013D F0              794             MOVX    @DPTR,A
013E 22              795             RET
                     796     DSD1:
013F B84603          797             CJNE    R0,#FALSE,DSD2
0142 54FB            798             ANL     A,#NOT PUBD
0144 F0              799             MOVX    @DPTR,A
                     800     DSD2:
0145 22              801             RET
                     802
                     803 +1  $EJ
```

```
LOC  OBJ            LINE     SOURCE

                     804
                     805     ;*************************************************
                     806     ;
                     807     ;       TOPO MOTOR CONTROL COMMANDS
                     808     ;
                     809     ;*************************************************
                     810     ; DOCUR5
                     811     ;       RETURN THE CURRENT 5 TO IR LINK
                     812     ;*************************************************
                     813     DOCUR5:
0146 900000    F     814             MOV     DPTR,#5+DISSOFAR
0149 0100      F     815             AJMP    STORP
                     816
                     817
                     818     ;*************************************************
                     819     ;
                     820     ; DOCUR0
                     821     ;       RETURN THE CURRENT 0 TO IR LINK
                     822     ;*************************************************
                     823     DOCUR0:
014B 900000    F     824             MOV     DPTR,#0+DISSOFAR
014E 0100      F     825             AJMP    STORP
                     826
                     827     ;*************************************************
                     828     ;
                     829     ; DOCUR50
                     830     ;       RETURN THE CURRENT 5 DOT AND 0 DOT
                     831     ;
                     832     ;*************************************************
                     833     DOCUR50:
0150 900000    F     834             MOV     DPTR,#DOT5D5R+1
0153 7800      F     835             MOV     R0,#IBUFF+?PARAM1
0155 1100      F     836             ACALL   STP
                     837
0157 900000    F     838             MOV     DPTR,#DOT0D5R+1
015A 7800      F     839             MOV     R0,#IBUFF+?PARAM2
015C 0100      F     840             AJMP    STP
                     841
                     842
                     843     ;*************************************************
                     844     ;
                     845     ; DOWARM
                     846     ;
                     847     ;*************************************************
                     848     DOWARM:
015E 22              849             RET
                     850
                     851
                     852     ;*************************************************
                     853     ;
                     854     ; DOMRESET
                     855     ;       RESET THE MOTORS AND THE PARAMTERS
                     856     ;
                     857     ;*************************************************
```

```
LOC  OBJ            LINE     SOURCE

                     858     DOMRESET:
015F 020000   F      859             LJMP    INITPARAMETERS
                     860
                     861 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                    862     ;*************************************************
                    863     ;
                    864     ; DOMOVE
                    865     ;
                    866     ;*************************************************
                    867     DOMOVE:
0162 020000   F     868             LJMP    MOVE EXACT                      ; CALL DAVE
                    869
                    870     ;*************************************************
                    871     ;
                    872     ; DOSET5
                    873     ;
                    874     ;*************************************************
                    875     DOSET5:
0165 900000   F     876             MOV     DPTR,#XBUFF+?PARAM1
0168 1100     F     877             ACALL   CONVPARAM                       ; CONVERT FIRST PARAM
016A 4014           878             JC      BADP
                    879 +1
                    880 +1
                    881 +2  ; STOR16 5+DIS
016C 900000   F     8 +2            MOV     DPTR,#5+DIS             ; THREE BYTE XFER
016F 120000   F     883 +1          CALL    STORZ16
0172 22             884             RET
                    885     ;*************************************************
                    886     ;
                    887     ; DOSET0
                    888     ;
                    889     ;*************************************************
                    890     DOSET0:
0173 900000   F     891             MOV     DPTR,#XBUFF+?PARAM1                     ; POINT TO SECOND
0176 1100     F     892             ACALL   CONVPARAM
0178 4006           893             JC      BADP                    ; IF ERROR THEN GO DO
                    894 +1
                    895 +1
                    896 +2  ; STOR16 0+DIS
017A 900000   F     897 +2          MOV     DPTR,#0+DIS             ; THREE BYTE XFER
017D 120000   F     898 +1          CALL    STORZ16
                    899     BADP:
0180 22             900             RET
                    901     ;*************************************************
                    902     ;
                    903     ; DOSETV
                    904     ;
                    905     ;*************************************************
                    906     DOSETV:
0181 900000   F     907             MOV     DPTR,#XBUFF+?PARAM1
0184 1100     F     908             ACALL   CONVPARAM                       ; CONVERT FIRST PARAM
0186 40F8           909             JC      BADP
                    910 +1
                    911 +1
                    912 +2  ; STOR16 5+VEL
0188 900000   F     913 +2          MOV     DPTR,#5+VEL             ; THREE BYTE XFER
018B 120000   F     914 +1          CALL    STORZ16
018E 020000   F     915             LJMP    SET VELOCITY
```

```
LOC  OBJ           LINE     SOURCE

                    916
                    917
                    918     ;***********************************************
                    919     ;
                    920     ; DORAMP
                    921     ;       TEST FOR THE JOYSTICK COMMANDS
                    922     ;
                    923     ;***********************************************
                    924     DORAMP:
0191 0100      F    925             AJMP    DORAMPL
                    926
                    927     ;***********************************************
                    928     ;
                    929     ; DOJOY
                    930     ;
                    931     ;***********************************************
                    932     DOJOY:
0193 900000    F    933             MOV     DPTR,#XBUFF+?PARAM1
0196 1100      F    934             ACALL   CONVP                   ; CONVERT FIRST PARAM
0198 4016           935             JC      BADPARAM
                    936 +1
                    937 +1
                    938 +2  ; STOR16 S+VEL
019A 900000    F    939 +2          MOV     DPTR,#S+VEL             ; THREE BYTE XFER
019D 120000    F    940 +1          CALL    STORZ16
                    941
01A0 900000    F    942             MOV     DPTR,#XBUFF+?PARAM2     ; POINT TO SECOND
01A3 1100      F    943             ACALL   CONVP
01A5 4009           944             JC      BADPARAM                ; IF ERROR THEN GO DO
                    945 +1
                    946 +1
                    947 +2  ; STOR16 O+VEL
01A7 900000    F    948 +2          MOV     DPTR,#O+VEL             ; THREE BYTE XFER
01AA 120000    F    949 +1          CALL    STORZ16
                    950
01AD 020000    F    951             LJMP    JOY                     ; DO A JOYSTICK
                    952
                    953     BADPARAM:
01B0 22             954             RET
                    955 +1  $EJ
```

```
LOC  OBJ            LINE     SOURCE

                     956     ;*************************************************
                     957     ;
                     958     ; DOSRESET
                     959     ;       RESET THE SPEECH MODULE
                     960     ;
                     961     ;*************************************************
                     962     DOSRESET:
                     963     ;
                    964 +1
01B1 78F7            965 +1          MOV     R0,#NOT ECHONOTRESET            ; ACTIVE LOW
01B3 120000   F      966 +1          CALL    CLRPORTB
                     967 +1
01B6 7808            968 +1          MOV     R0,#ECHONOTRESET
01B8 120000   F      969 +1          CALL    SETPORTB                                ; LEAVE IN NOT RESET
                     970 +1
                     971 +1
01BB 22              972             RET
                     973     ;*************************************************
                     974     ;
                     975     ; DOSPCH
                     976     ;
                     977     ;*************************************************
                     978     DOSPCH:
01BC 750000   F      979             MOV     RSPTR,#LOW( XBUFF+?CHAR1)       ; POINT TO START OF SPEECH
01BF 0100     F      980             AJMP    SPCHON
                     981
                     982
                     983 +1  $EJ
```

```
LOC  OBJ            LINE     SOURCE

                     984     ;*************************************************
                     985     ;
                     986     ;
                     987     ; HEAD COMMAND STUFF
                     988     ;
                     989     ;
                     990     ;*************************************************
                     991     ; DO LATCH
                     992     ;
                     993     ;*************************************************
                     994     DOLATCH:
01C1 1100      F     995             ACALL   READSWITCH
01C3 900000    F     996             MOV     DPTR,#SWLOCLAT                  ; POINT TO XRAM STORAGE
01C6 E8              997             MOV     A,R0
01C7 F0              998             MOVX    @DPTR,A
01C8 22              999             RET
                    1000     ;*************************************************
                    1001     ; DOAUTO
                    1002     ;
                    1003     ;*************************************************
                    1004     DOAUTO:
01C9 E500      F    1005             MOV     A,IBUFF+?PARAM1 ; GET FLAG
01CB B45403         1006             CJNE    A,$TRUE,DL1
01CE D200      F    1007             SETB    HEAD?
01D0 22             1008             RET
                    1009     DL1:
01D1 B44602         1010             CJNE    A,#FALSE,DLEX
01D4 C200      F    1011             CLR     HEAD?
                    1012     DLEX:
01D6 22             1013             RET
                    1014 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                    1015     ;
                    1016     ; DORAMPL
                    1017     ;
                    1018     DORAMPL:
01D7 900000   F     1019             MOV     DPTR,#5+DELTA
01DA 7419           1020             MOV     A,#HIGH 6552
01DC F0             1021             MOVX    @DPTR,A
01DD A3             1022             INC DPTR
01DE 7498           1023             MOV     A,#LOW 6552
01E0 F0             1024             MOVX    @DPTR,A
                    1025
01E1 900000   F     1026             MOV     DPTR,#0+DELTA
01E4 7437           1027             MOV     A,#HIGH 14088
01E6 F0             1028             MOVX    @DPTR,A
01E7 A3             1029             INC DPTR
01E8 7408           1030             MOV     A,#LOW 14088
01EA 22             1031             RET
                    1032     ;*************************************************
                    1033     ; STATL
                    1034     ;       SET UP A STATUS MESSAGE TO SEND TO HOST
                    1035     ;       MESSAGE IS :    DON'T CARE
                   1036     ;                       ACK
                    1037     ;                         QUEUE STAT
                    1038     ;                         HEAD STAT
                    1039     ;                         GENERAL STAT
                    1040     ;                         MODEL NUMBER
                    1041     ;
                    1042     ;*************************************************
                    1043     STATL:
01EB C200     F     1044             CLR     SHORT?
                    1045
                    1046     ; GET QUEUE STAT
01ED 7406           1047             MOV     A,#6
01EF 2430           1048             ADD     A,#30H
01F1 F500     F     1049             MOV     IBUFF+?CHAR1,A
                    1050     ;
                    1051     ; GET HEAD SWITCH STAT
                    1052     ;
01F3 900000   F     1053             MOV     DPTR,#SWLOCLAT
01F6 E0             1054             MOVX    A,@DPTR
01F7 D2E4           1055             SETB    GOTSWT                  ; SE IF ANY ACTIVE
01F9 F8             1056             MOV     R0,A                    ; SAVE IT
01FA 900000   F     1057             MOV     DPTR,#SWITCH
01FD E0             1058             MOVX    A,@DPTR
01FE 540F           1059             ANL     A,#0FH                  ; MASK UNUSED STUFF
0200 C8             1060             XCH     A,R0
0201 B80002         1061             CJNE    R0,#0,STT1
0204 C2E4           1062             CLR     GOTSWT
                    1063     STT1:
0206 F500     F     1064             MOV     IBUFF+?CHAR2,A  ; STOR THE SWITCH STAT
                    1065
                    1066     ;
                    1067     ; SET THE STAT BITS
                    1068     ;
```

```
LOC  OBJ           LINE     SOURCE

0208 78C0          1069             MOV     R0,#0C0H                ; CLEAR ALL BITS
                   1070     ;
                   1071     ;       SPCHTEST IN BIT 7
                   1072     ;
                   1073     ;       SEE IF MOVING
020A 900000   F    1074             MOV     DPTR,#DOTSACT+1         ; GET ACT SPEED
020D E0            1075             MOVX    A,@DPTR
020E B40009        1076             CJNE    A,#0,STA1               ; IF NOT 0 THEN MOVING
0211 900000   F    1077             MOV     DPTR,#DOTOACT+1         ; GET 0 SPEED
0214 E0            1078             MOVX    A,@DPTR
0215 B40002        1079             CJNE    A,#0,STA1
0218 8004          1080             SJMP    STA2                    ; NOT MOVING
                   1081     STA1:
021A 7440          1082             MOV     A,#MOVING               ; SAY WE ARE MOVING
021C 48            1083             ORL     A,R0
021D F8            1084             MOV     R0,A                    ; SAVE IT
                   1085     ;
                   1086     ;       TONE SOUNDING
                   1087     ;
                   1088     ;
                   1089     ;       POWER ON BIT
                   1090     ;
                   1091     STA2:
021E 7410          1092             MOV     A,#POWERON
0220 300002   F    1093             JNB     POWER?,STA3
0223 48            1094             ORL     A,R0
0224 F8            1095             MOV     R0,A                    ; STORE POWER BIT
                   1096     STA3:
0225 8800     F    1097             MOV     IBUFF+?CHAR3,R0 ; STORE TO SEND
                   1098     ;
                   1099     ; SEND THE TOPO NUM
                   1100     ;
0227 E4            1101             CLR     A                       ; GET THE MODEL NUMBER
0228 90000C        1102             MOV     DPTR,#MODELOC
022B 93            1103             MOVC    A,@A+DPTR
022C F500     F    1104             MOV     IBUFF+?CHAR4,A
022E 22            1105             RET
                   1106     ;***********************************************
                   1107     ;
                   1108     ; QUEERL
                   1109     ;
                   1110     ;***********************************************
                   1111     QUEERL:
022F 1100     F    1112             ACALL   MOVFROMX
0231 750050   F    1113             MOV     IBUFF,#TOPOMSG
0234 E500     F    1114             MOV     A,IBUFF+?DEVNUM
0236 B45202        1115             CJNE    A,#ACK1,DOQ1
0239 8008          1116             SJMP    DOQ3
                   1117     DOQ1:
023B B45102        1118             CJNE    A,#ACK0,DOQ2
023E 8003          1119             SJMP    DOQ3
                   1120     DOQ2:
0240 750052   F    1121             MOV     IBUFF+?DEVNUM,#ACK1
                   1122     DOQ3:
```

8
4
12

```
LOC  OBJ           LINE    SOURCE

0243 C200      F   1123            CLR     SHORT?
0245 22            1124            RET
                   1125    ;
                   1126    ; D01
                   1127    ;       RETURN HEX IN ZL,A
                   1128    ;
                   1129    D01:
0246 E0            1130            MOVX    A,@DPTR                 ; GET FIRST BYTE
0247 A3            1131            INC     DPTR                            ; POINT TO NEXT
0248 1100      F   1132            ACALL   ASCHEX                  ; CONVERT ASCII TO HEX
024A 22            1133            RET
                   1134    ;
                   1135    ; CONIT
                   1136    ;       RETURN HEX IN ZL,A
                   1137    ;       CLR C IS OKAY ; SETB C IS ERROR
                   1138    ;
                   1139    CONIT:
024B 1100      F   1140            ACALL   D01                     ; CON A CHAR
024D C4            1141            SWAP    A                       ; STORE IT
024E F507          1142            MOV     ZL,A
0250 4009          1143            JC      BADCONV                 ; JUMP IF NOT NUMBER
0252 1100      F   1144            ACALL   D01                     ; DO NEXT CHAR
0254 4507          1145            ORL     A,ZL                    ; STUFF TOGETHER
0256 F507          1146            MOV     ZL,A
0258 4001          1147            JC      BADCONV                 ; JUMP IF NOT NUMBER
025A 22            1148            RET
                   1149    BADCONV:
025B D3            1150            SETB    C
025C 22            1151            RET
                   1152    ;
                   1153    ; CONVP
                   1154    ;       RETURN HEX IN ZL,A
                   1155    ;       CLR C IS OKAY ; SETB C IS ERROR
                   1156    ;
                   1157    CONVP:
025D 1100      F   1158            ACALL   CONIT
025F 40FA          1159            JC      BADCONV                 ; IF ERROR THEN EXIT
                   1160    ;
                   1161    ; SIGN EXTEND
                   1162    ;
0261 750600        1163            MOV     ZH,#0
0264 30E703        1164            JNB     ACC.7,CONOK
0267 7506FF        1165            MOV     ZH,#0FFH
                   1166    CONOK:
026A 22            1167            RET
                   1168
                   1169    ;
                   1170    ; CONVPARAM
                   1171    ;       CONVERT A 4 ASCII DIGITS @R0 TO HEX AT ZL,ZH
                   1172    CONVPARAM:
026B 1100      F   1173            ACALL   CONIT                   ; CONVER FIRST TWO
026D 40EC          1174            JC      BADCONV                 ; IF BAD THEN EXIT
026F F506          1175            MOV     ZH,A                    ; SAVE TOP RESULT
0271 1100      F   1176            ACALL   CONIT                   ; CONVERT LOW BYTES
```

```
LOC  OBJ           LINE     SOURCE

0273 22            1177             RET
                   1178 +1  $EJ
```

```
LOC  OBJ            LINE     SOURCE

                    1179     ;
                    1180     ; test the encoder tecnique
                    1181     ; The scheme is to interupt on either encoder rising edge
                    1182     ;       The latched rising edges are or'd together and sent to int 0
                    1183     ;       The latched valuse are individually input to pins to detect which it was
                    1184     ;       The encoder ISR detects which it was and stores the current value of an 8 bit
                    1185     ;       timer into a location for that motor.
                    1186     ;       Meanwhile, the timer is counting up. When it overflows, it causes an interrrupt.
                    1187     ;       The TImer ISR increments the top byte of both the timers for the motors in
                    1188     ;       software. It then checks if either motor had a rising edge durring the previous
                    1189     ;       256 microseconds. If one did, then the ISR subtracts the captured timer time
                    1190     ;       from t last 8 bit time. This allows us to find the tiem interval. If
                    1191     ;       if either the right or left software motor timers overflow, then that motor
                    1192     ;       is considered to be stopped.
                    1193     ;
                    1194     ;
                    1195     ; MESSAGE STRINGS
                    1196     ;
0274 40454D4F       1197     ?COMPERR:       DB      'MEMORY FAILURE',CR,0
0278 52592046
027C 41494C55
0280 5245
0282 0D
0283 00
0284 05             1198     ?MESS:  DB      5,'33P MONDO BONZO 1.5 AUGUST FIFTEENTH 9 TEEN 8 T 3',CR,0
0285 33335020
0289 404F4E44
028D 4F20424F
0291 4E5A4F20
0295 312E3520
0299 41554755
029D 53542046
02A1 49465445
02A5 454E5448
02A9 20392054
02AD 45454E20
02B1 38205420
02B5 33
02B6 0D
02B7 00
02B8 0D             1199     ?CLEAR:         DB      CR,LF,0
02B9 0A
02BA 00
02BB 0D             1200     ?CARET:         DB      CR,00H
02BC 00
02BD 544F504F       1201     ?MODEL:         DB      'TOPO MAUD ALL 1 A B ',CR,0
02C1 204D4155
02C5 4420414C
02C9 4C203120
02CD 41204220
02D1 0D
02D2 00
                    1202 +1
                    1203
```

```
LOC  OBJ           LINE     SOURCE

                   1204     ;
                   1205     ; CONVERT THE HEX NIBBLE IN LOW FOUR BITS OF ACC TO ASCII
                   1206     ;       ON ENTRY: ACC.0-ACC.3 HAS NIBBLE
                   1207     ;       ON EXIT:  ACC HAS ASCII NUMBER
                   1208     CONVHEX:
02D3 540F          1209             ANL     A,#0FH                  ; MASK TOP NIBBLE
02D5 2490          1210             ADD     A,#90H                  ; 9A-9F AUX. C=1
02D7 D4            1211             DA      A
02D8 3440          1212             ADDC    A,#40H                  ; CONVERT TO ASCII NUMBER
02DA D4            1213             DA      A
02DB 22            1214             RET
                   1215 +1  $EJECT
```

```
LOC  OBJ           LINE     SOURCE

                   1216     ;
                   1217     ; MESSAGE DISPLAY ROUTINES
                   1218     ;
                   1219     SHOLOOP:
02DC E4            1220             CLR     A                       ; ZERO FOR @A+DPTR
02DD 93            1221             MOVC    A,@A+DPTR               ; FETCH NEXT MSG BYTE
02DE A3            1222             INC     DPTR                    ; INC THE DATA PTR
02DF B40001        1223             CJNE    A,#0,SHOIT              ; IF NOT ZERO THEN DISPLAY BYTE
02E2 22            1224             RET                             ; ELSE RETURN
                   1225     SHOIT:
02E3 1100      F   1226             ACALL   SPOUT                   ; OUT IT GOES
02E5 80F5          1227             JMP     SHOLOOP
                   1228     ;
                   1229     ; ASCHEX
                   1230     ;       CONVERT ASCII BYTE IN ACC TO HEX DIGIT IN LO NIBBLE OF ACC
                   1231     ;       ON ENTRY : ACC HAS BYTE
                   1232     ;       ON EXIT  : ACC(0-3) HAS DIGIT IF CAL
                   1233     ;                :                    ELSE ACC IS OLD CHAR
                   1234     ;                : CLC MEANS GOT A LEGAL DIGIT
                   1235     ;                : SEC MEANS NO GOT
                   1236     ;                : ALTERS ACC,R2
                   1237     ASCHEX:
02E7 1100      F   1238             ACALL   NMTEST                  ; CHECK FOR 0-9 AND CONVERT IF SO
02E9 5002          1239             JNC     GOTIT                   ; GO IF GOT IT
02EB 1100      F   1240             ACALL   HXTEST                  ; ELSE SEE IF A-F
                   1241     GOTIT:
02ED 22            1242             RET                             ; CARRY HAS APPRORIATE VALUES FROM ABOVE
                   1243     ;
                   1244     ; NMTEST
                   1245     ;       TEST FOR A DIGIT FROM 0-9 ASCII AND CONVERT IF IT IS
                   1246     ;       ON ENTRY : ACC HAS ASCII BYTE
                   1247     ;       ON EXIT  : IF '0'<=ACC<='9' THEN
                   1248     ;                       ACC=DIGIT
                   1249     ;                       CLC
                   1250     ;                  ELSE ACC=ASCII BYTE
                   1251     ;                       SEC
                   1252     ;                  END
                   1253     ;                : ALTERS ACC,R2
                   1254     NMTEST:
02EE FA            1255             MOV     R2,A                    ; SAVE CHAR IN CASE NOT DIGIT
02EF C3            1256             CLR     C
02F0 9430          1257             SUBB    A,#'0'                  ; CONVERT IT
02F2 B40902        1258             CJNE    A,#09,CHKNM             ; SEE IF NUMBER
                   1259     GOTNM:
02F5 C3            1260             CLR     C                       ; GOT THE DIGIT
02F6 22            1261             RET
                   1262     CHKNM:
02F7 40FC          1263             JC      GOTNM                   ; GO IF GOT IT
02F9 EA            1264             MOV     A,R2                    ; ELSE RESTORE CHAR
02FA D3            1265             SETB    C                       ; INDICATE NO DIGIT
02FB 22            1266             RET
                   1267
                   1268     ;
                   1269     ; HXTEST
```

```
LOC  OBJ           LINE     SOURCE

                   1270     ;       TEST FOR A DIGIT FROM 0-9 ASCII AND CONVERT IF IT IS
                   1271     ;       ON ENTRY : ACC HAS ASCII BYTE
                   1272     ;       ON EXIT  : IF 'A'<=ACC<='F' THEN
                   1273     ;                       ACC=DIGIT
                   1274     ;                       CLC
                   1275     ;                  ELSE ACC=ASCII BYTE
                   1276     ;                       SEC
                   1277     ;                  END
                   1278     ;                : ALTERS ACC,R2
                   1279     HXTEST:
02FC FA            1280             MOV     R2,A                    ; SAVE CHAR IN CASE NOT DIGIT
02FD C3            1281             CLR     C
02FE 9441          1282             SUBB    A,#'A'                  ; CONVERT IT
0300 B40504        1283             CJNE    A,#05,CHKHX             ; SEE IF NUMBER
                   1284     GOTHX:
0303 240A          1285             ADD     A,#0AH                  ; CONVERT TO HEX
0305 C3            1286             CLR     C                       ; GOT THE DIGIT
0306 22            1287             RET
                   1288     CHKHX:
0307 40FA          1289             JC      GOTHX                   ; GO IF GOT IT
0309 EA            1290             MOV     A,R2                    ; ELSE RESTORE CHAR
030A D3            1291             SETB    C                       ; INDICATE NO DIGIT
030B 22            1292             RET
                   1293 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                   1294     ;
                   1295     ; WAITMESS
                   1296     ;       ROUTINES WAITS FOR A MESSAGE
                   1297     ;       INPUTS: NONE
                   1298     ;       OUTPUT: IF NO ERROR THEN
                   1299     ;                       CLR C
                   1300     ;                       ACC IS THE COMMAND CHARACTER OF MESSAGE
                   1301     ;                       CLR THE OUT_OF_RANGE FLAG(CARRIER?)
                   1302     ;               ELSE SETB C
                   1303     ;
                   1304     ;       ALTERS: JUST ABOUT EVERY BODY
                   1305     ;
                   1306     WAITMESS:
030C 1100      F   1307             ACALL   RESETAGC                ; RESET THE AUTO GAIN CONTROL
030E 1100      F   1308             ACALL   WAITMESSAGE             ; WAIT TO RECEIVE NEXT IR BURST
0310 40FA          1309             JC      WAITMESS                ; IF ERROR THEN DO AGAIN
0312 1100      F   1310             ACALL   CHECKUS                 ; SEE IF MESSAGE TO US
0314 40F6          1311             JC      WAITMESS                ; IF NOT THEN DO AGAIN
0316 1100      F   1312             ACALL   CHKQUEER                ; SEE IF QUERY
0318 4002          1313             JB      QUERY,WAIT1             ; IF SO THEN DO AGAIN
031A 1100      F   1314             ACALL   MOVTOXBUFF              ; MOVE TO EXTERNAL RAM
                   1315     WAIT1:
031C C200      F   1316             CLR     CARRIER?                ; SAY WE GOT A MESS
031E E500      F   1317             MOV     A,IBUFF+?DEVNUM ; RETURN THE COMMAND CHAR OF MESSAGE
0320 C3            1318             CLR     C
                   1319 +1
0321 22            1320             RET
                   1321     ;
                   1322     ;RESETAGC
                   1323     ;       RESET THE AUTO GAIN CTL TO RECIEVE A MESSAGE
                   1324     ;
                   1325     RESETAGC:
                   1326 +1
0322 7804          1327 +1          MOV     R0,#AGCRESET                    ; RESET ACTIVE HIGH
0324 120000    F   1328 +1          CALL    SETPORTB                                ; RESET ECHO
                   1329 +1
0327 78FB          1330 +1          MOV     R0,#NOT AGCRESET                        ; LEAVE IT NOT RESETTED
0329 120000    F   1331 +1          CALL    CLRPORTB
                   1332 +1
                   1333 +1
032C 22            1334             RET
                   1335 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                    1336     ;
                    1337     ; WAITMESSAGE
                    1338     ;
                    1339     WAITMESSAGE:
032D 1100      F    1340             ACALL   ENABLERECV              ; SETUP TO GET IR
                    1341     WM1:
032F 300004    F    1342             JNB     RECV?,WM2              ; WAIT TIL GOT
0332 1100      F    1343             ACALL   CHECKRS232             ; SEE IF TO SEND
0334 80F9           1344             JMP     WM1
                    1345     WM2:
                    1346 +1
0336 200009    F    1347             JB      PARITYERROR,PARERROR   ; IF ERROR THENSET DPTR
0339 200006    F    1348             JB      SYNCERROR,SERR         ; IF SET THEN NOT ENOUGH SYNC BITS
033C 1100      F    1349             ACALL   CALCHECKSUM            ; GET CHECK SUM OF PCKET
033E 4002           1350             JC      SUMERROR               ; IF ERROR THEN SET DPTR
0340 C3             1351             CLR     C                      ; ELSE ALL OKAY
0341 22             1352             RET
                    1353     SERR:
                    1354     PARERROR:
                    1355     SUMERROR:
0342 D3             1356             SETB    C
0343 22             1357             RET
                    1358 +1  $EJ
```

```
LOC  OBJ           LINE    SOURCE

                   1359    ;
                   1360    ; CHECKRS232
                   1361    ;
                   1362    CHECKRS232:
0344 309905        1363            JNB     TI,CHROUT               ; ELSE IF TO EXIT THEN DO SO
0347 E500      F   1364            MOV     A,RSPTR                 ; GET PTR TO CURRENT CHAR IN BUFF
0349 B40001    F   1365            CJNE    A,#LOW XBUFFEND,CHR2    ; IF NOT LAST THEN DOIT
                   1366    CHROUT:
034C 22            1367            RET
                   1368    CHR2:
034D 0500      F   1369            INC     RSPTR
034F 758300    F   1370            MOV     DPH,#HIGH XBUFF         ; POINT TO CHAR
0352 F582          1371            MOV     DPL,A
0354 E0            1372            MOVX    A,@DPTR                 ; FETCH DATA TO SEND
0355 C2E7          1373            CLR     ACC.7                   ; NO PARITY
0357 C299          1374            CLR     TI                      ; INIT HARDWARE
0359 F899          1375            MOV     SBUF,A                  ; SEND IT
                   1377    ;
                   1378    ; FETCHNUM
                   1379    ;       GET THE TOPONUM AND RETURN IN A REG
                   1380    ;
                   1381    FETCHNUM:
035C 908001        1382            MOV     DPTR,#CHANLOC           ; FETCH SWITCH NUMBER
035F E0            1383            MOVX    A,@DPTR
0360 540F          1384            ANL     A,#0FH                  ; MASK TOPO BITS
0362 2430          1385            ADD     A,#FIRSTCHAN            ;
0364 22            1386            RET
                   1387    ;
                   1388    ; CHECKUS
                   1389    ;       SEE IF MESSAGE IS FOR OUR ATTENTION
                   1390    ;       OUTPUTS:NOACK? IS FLAG TO ACK OR NOT (PUBLIC OR PRIVATE CHANNEL)
                   1391    ;               CLR C IS MESSAGE IS FOR US
                   1392    ;       ALTERS: R0,A,DPTR
                   1393    CHECKUS:
0365 C200      F   1394            CLR     NOACK?                  ; SAY NO ACKNOWLEDGE
0367 A800      F   1395            MOV     R0,IBUFF+?CHANUM        ; GET CHANNEL NUMBER
                   1396
0369 1100      F   1397            ACALL   FETCHNUM
036B B50002        1398            CJNE    A,AR0,NOTOP             ; IF NOT OUR PRIVATE THEN GO
036E 8018          1399            SJMP    CUOK                    ; ELSE EXIT...ITS OUR PRIVATE CHAN
                   1400    NOTOP:
0370 D200      F   1401            SETB    NOACK?                  ; NOT OUR PRIVATE CHANNEL SO NO ACK
0372 E8            1402            MOV     A,R0                    ; GET CHAN NUM AGAIN
0373 B44102        1403            CJNE    A,#PUBCHAN1,NTP1
0376 8010          1404            SJMP    CUOK
                   1405    NTP1:
0378 9441          1406            SUBB    A,#PUBCHAN2-1           ; SET UP TO SEE IF PUBLIC
037A F8            1407            MOV     R0,A                    ; SAVE IT
037B D3            1408            SETB    C                       ;
                   1409    NTL:
037C 33            1410            RLC     A                       ; ROTATE BIT TO CORRECT SPOT
037D C3            1411            CLR     C
037E D8FC          1412            DJNZ    R0,NTL                  ; DO TIL DONE
```

```
LOC  OBJ            LINE     SOURCE

                    1413
0380 F8             1414             MOV     R0,A
                    1415
0381 900000   F     1416             MOV     DPTR,#PUBLOC          ; GET THE PUBLIC BITS
0384 E0             1417             MOVX    A,@DPTR
0385 58             1418             ANL     A,R0                  ; SEE IF ACTIVE
0386 6002           1419             JZ      NOTUS
                    1420     CUOK:
0388 C3             1421             CLR     C                     ; SAY WE GOT A MESSAGE
0389 22             1422             RET
                    1423     NOTUS:
038A D3             1424             SETB    C
038B 22             1425             RET
                    1426 +1  $EJ
```

MOV RSPTR, #LOW XBUFF

```
LOC  OBJ          LINE    SOURCE

                  1427    ;
                  1428    ; CHKQUEER
                  1429    ;       SEE IF QUEER(Y)
                  1430    ;
                  1431    CHKQUEER:
038C E500     F   1432            MOV     A,IBUFF+?DEVNUM         ; JOHNNY ARE YOU QUEER?
038E B43004       1433            CJNE    A,#WHATCMD,NOTQUEER     ; I GUESS NOT
0391 D200     F   1434            SETB    QUERY?
0393 D3           1435            SETB    C
0394 22           1436            RET
                  1437    NOTQUEER:
0395 C3           1438            CLR     C
0396 22           1439            RET
                  1440
                  1441
                  1442    ;
                  1443    ;ASSUME THAT R0 HAS THE NUMBER OF MOTOR CLOCKS TO DELAY
                  1444    ;
                  1445    LDELAY:
                  1446
0397 E500     F   1447            MOV     A,CURSTATE
0399 B401FB       1448            CJNE    A,#1,LDELAY                     ; WAIT 32 MS
                  1449    LD1:
039C E500     F   1450            MOV     A,CURSTATE
039E B40102       1451            CJNE    A,#1,LD2                        ; WAIT FOR CHANGE
03A1 80F9         1452            SJMP    LD1
                  1453    LD2:
03A3 D8F2         1454            DJNZ    R0,LDELAY                       ; WAIT X TIMES 32 MS
03A5 22           1455            RET
                  1456    ;
                  1457    ; CALCHECKSUM
                  1458    ;
                  1459    CALCHECKSUM:
03A6 E500     F   1460            MOV     A,IBUFF+?CHANUM ; GET THE ACK BIT
03A8 A2E7         1461            MOV     C,ACC.7
03AA 9200     F   1462            MOV     ACK1?,C
                  1463
03AC 7900         1464            MOV     R1,#0           ; CLEAR THE SUM
03AE 7800     F   1465            MOV     R0,#IBUFF               ; OUTPUT WHAT WE GOT
                  1466    CC1:
03B0 E6           1467            MOV     A,@R0                   ; GET THE NEXT CHAR
03B1 C0E0         1468            PUSH    ACC                     ; SAVE CHAR
03B3 29           1469            ADD     A,R1                    ; SUMM THE MESSAGE
03B4 F9           1470            MOV     R1,A                    ; SAVE FOR NEXT LOOP
03B5 D0E0         1471            POP     ACC                     ; GET CHAR TO CLEAR TOPOP BIT
03B7 C2E7         1472            CLR     ACC.7
03B9 F6           1473            MOV     @R0,A                   ; STORE IN BUFF
03BA 08           1474            INC     R0
03BB B800F2   F   1475            CJNE    R0,#LASTCHAR+1,CC1      ; SEND THE NEXT
03BE B90002       1476            CJNE    R1,#0,BADCHECK          ; IF NOT ZERO THEN SETC
03C1 C3           1477            CLR     C
03C2 22           1478            RET
                  1479    BADCHECK:
03C3 D3           1480            SETB    C                       ; BAD CHECKSUM
```

```
LOC  OBJ           LINE     SOURCE

03C4 22            1481           RET
                   1482     ;
                   1483     ; CLEARBUFF
                   1484     ;
                   1485     CLEARBUFF:
03C5 C0E0          1486           PUSH    ACC                     ; PRESERVE
03C7 7800      F   1487           MOV     R0,#IBUFF               ; OUTPUT WHAT WE GOT
03C9 7400          1488           MOV     A,#NULL                 ; CLEAR TO NULLS
                   1489     CB1:
03CB F6            1490           MOV     @R0,A                   ; GET THE NEXT CHAR
03CC 08            1491           INC     R0
03CD B800FB    F   1492           CJNE    R0,#LASTCHAR+1,CB1      ; SEND THE NEXT
03D0 D0E0          1493           POP     ACC                     ; RESTORE
03D2 22            1494           RET
                   1495 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                   1496     ;
                   1497     ; SPCHON
                   1498     ;
                   1499     SPCHON:
                   1500 +1
03D3 78F0          1501 +1          MOV     R0,#NOT SPCHDISAB
                   1502 +1  CLRPORTB:
03D5 900000   F    1503 +1          MOV     DPTR,#PORTBMASK
03D8 E0            1504 +1          MOVX    A,@DPTR
03D9 58            1505 +1          ANL     A,R0
03DA 908002        1506 +1          MOV     DPTR,#PORTB
03DD F0            1507 +1          MOVX    @DPTR,A
03DE 800E          1508 +1          SJMP    SAVMSKB
                   1509 +1
                   1510     ;
                   1511     ; DISABLE SPEECH
                   1512     ;
                   1513     SPCHOFF:
03E0 3099FD        1514             JNB     TI,$                    ; WAIT HERE TIL LAST CHAR IS XMITTED
                   1515 +1
03E3 7802          1516 +1          MOV     R0,#SPCHDISAB
                   1517 +1  SETPORTB:
03E5 900000   F    1518 +1          MOV     DPTR,#PORTBMASK
08 E0              1519 +1          MOVX    A,@DPTR
03E9 48            1520 +1          ORL     A,R0
03EA 908002        1521 +1          MOV     DPTR,#PORTB
03ED F0            1522 +1          MOVX    @DPTR,A
                   1523 +1  SAVMSKB:
03EE 900000   F    1524 +1          MOV     DPTR,#PORTBMASK
03F1 F0            1525 +1          MOVX    @DPTR,A
03F2 22            1526 +1          RET
                   1527 +1
                   1528     ;
                   1529     ;       READSWITCH
                   1530     ;
                   1531     READSWITCH:
03F3 900000   F    1532             MOV     DPTR,#SWITCH            ; READ THE SWITCHES RAM LOC
03F6 E0            1533             MOVX    A,@DPTR                 ; FETCH SWITCH
03F7 F8            1534             MOV     R0,A                    ; SAVE IT
03F8 D200     F    1535             SETB    CLRSWTCH?               ; TELL ISR TO CLEAR SWITCH
03FA 22            1536             RET
                   1537 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                   1538 +1
                   1539 +1
                   1540 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                   1541     ;
                   1542     ; IRCVINT
                   1543     ;       INIT THE IR RECV HARDWARE ANS SOFTWARE TO RECIEVE A BIT STREAM
                   1544     ;
                   1545     IRCVINIT:
                   1546     ENABLERECV:
03FB C200      F   1547             CLR     SEND?           ; DISABLE IR SENDING
03FD D200      F   1548             SETB    RSYNC?          ; LOOK FOR SYNCS
03FF C200      F   1549             CLR     PARITY?         ; NOT LOOKING FOR PARITY YET
0401 C200      F   1550             CLR     PARITYERROR             ; NO PARITY ERROR YET
0403 C200      F   1551             CLR     SYNCERROR               ; NO SYNC YET
0405 75007F    F   1552             MOV     CURCHAR,#STARTCHAR      ; SET UP FOR FIRST CHAR
0408 751100    F   1553             MOV     AR1PTR,#IBUFF   ; POIN TO GET BUFFER
040B 750000    F   1554             MOV     SYNCOUNT,#0             ; NUMBER OF SYNCS GOTTEN =0
040E C200      F   1555             CLR     RECVBIT                 ; INIT TO NO GOT
0410 C200          1556             CLR     STABLE?         ; SYNCS NOT STABLE YET
                   1557
0412 D200      F   1558             SETB    RECV?           ; SET TO INDIACTE WE SHOULD LOOK FOR BITS
0414 22            1559             RET
                   1560     ;
                   1561     ; IRSENDINT
                   1562     ;       INIT THE IR RECV HARDWARE AND SOFTWARE TO RECEIVE A BIT STREAM
                   1563     ;
                   1564     IRSENDINIT:
                   1565     ENABLESEND:
0415 750000    F   1566             MOV     LASTPTR,#IBUFF+1        ; ASSUME TO SEND SHORT PACKET
0418 200003    F   1567             JB      SHORT?,EN3              ; IF SHORT THEN GO
041B 750000    F   1568             MOV     LASTPTR,#LASTCHAR+1     ; ELSE LONG PACKET...SET LENGTH TO END
                   1569     EN3:
041E C200      F   1570             CLR     RECV?                   ; DISABLE IR SEND ISR FLAG
0420 D200      F   1571             SETB    SYNC?                   ; LOOK FOR SYNCS
0422 750000    F   1572             MOV     CURCHAR,#0              ; SAY DON'T SEND THIS CHAR SO WE START
0425 751100    F   1573             MOV     AR1PTR,#IBUFF   ; POINT TO START
0428 C200      F   1574             CLR     SENDBIT?                ; DON'T SEND BIT YET
042A D200      F   1575             SETB    SYNC?                   ; SEND SYNC FIRST
042C 750000    F   1576             MOV     SYNCOUNT,#0             ; NUMBER OF SYNCS GOTTEN =0
042F C200      F   1577             CLR     S512?                   ; INIT TO NOT 512 MICRO SECS
0431 D200      F   1578             SETB    SEND?                   ; SET TO INDICATE WE SHOULD LOOK FOR BITS
0433 22            1579             RET
                   1580 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                   1581     ;
                   1582     ; SPOUT ADD ODD PARITY TO ACC AND XMIT
                   1583     ;       WHEN SERIAL OUT READY
                   1584     ;       ON ENTRY : ACC = CHAR OUT
                   1585     ;       ON EXIT : ACC IS UNCHANGED
                   1586     ;                 C IS ALTERED
                   1587     SPOUT:
0434 C2E7          1588             CLR     ACC.7                   ; SET CHARS PARITY TO NO PARITY
0436 3099FD        1589             JNB     TI,$                    ; WAIT HERE TIL OKAY TO XMIT
0439 C299          1590             CLR     TI                      ; CLEAR BIT AFTER SERVICE
043B F599          1591             MOV     SBUF,A                  ; OUTPUT THE CHARACTER
043D 22            1592             RET
                   1593     ;
                   1594     ; SPIN INPUT THE NEXT CHRCTER FROM THE SERIAL PORT
                   1595     ;       SET CARRY IF ODD PARITY ERROR
                   1596     ;       ON ENTRY : NADA
                   1597     ;       N EXIT: ACC HAS CHAR
                   1598     ;                 C IS SET IF PARITY ERROR
                   1599     ;
                   1600     SPIN:
043E 3098FD        1601             JNB     RI,$                    ; WAIT HERE TIL CHARAACTER SHOWS UP
0441 C298          1602             CLR     RI                      ; CLEAR BIT AFTER SERVICE
0443 E599          1603             MOV     A,SBUF                  ; GET THE CHAR
0445 A2D0          1604             MOV     C,P                     ; GET PARITY BIT
0447 B3            1605             CPL     C                       ; COMPLEMENT FOR ODD-PARITY
0448 547F          1606             ANL     A,#7FH                  ; MASK TOP BIT
044A 22            1607             RET
                   1608 +1
                   1609 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                   1610             USING   ISRBANK                 ; USING THE ISR REGESTER BANK
                   1611     ;R1PTR          SET     R1              ; CURRENT LOCATION OF CHAR TO SEND
                   1612     ;R2LNEW SET     R2                      ; LOW " THE INT NEW TIME
                   1613     ;R3RNEW SET     R3                      ; LOW "
                   1614     ;LSTIME SET     R4                      ; LEFT 8 BIT SOFTWARE TIMER
                   1615     ;RSTIME SET     R5                      ; LEFT 8 BIT SOFTWARE TIMER
----               1616             CSEG AT 0
                   1617     ;       INTERRUPT VECTORS
                   1618     ;
                   1619     ; POWER UP RESET VECTOR
0000               1620             ORG     RESET
0000 020000    F   1621             JMP     RESTART
                   1622     ;
0003               1623             ORG     EXTI0
                   1624     ; EXTERNAL 0 INTERRUPT VECTOR
                   1625     ;       MOV     TEMPTIME,ENCODTIME      ; GET THE CURRENT TIME
0003 0100      F   1626             AJMP    ENCODISR
0005 32            1627             RETI
                   1628
000B               1629             ORG     TIMER0
                   1630     ; TIMER/COUNTER 0 INT VECTOR
000B 32            1631             RETI
000C 31            1632     MODELOC:        DB      '1'
000D 08            1633     DATELOC:        DB      08H,05H,083H
000E 05
000F 83
                   1634     ;
0013               1635             ORG     EXTI1
                   1636     ; EXTERNAL 1 INT VECTOR
                   1637     ;
                   1638     ; RECEIVEISR
                   1639     ;       IAR FOR THE IR RECEIVE DIODE FROM THE AGC
                   1640     ;
0013 D200      F   1641             SETB    READBIT
0015 32            1642             RETI
                   1643     ;
001B               1644             ORG     TIMER1
                   1645     ; TIMER/COUNTER 1 IN VECTOR
                   1646     ;       CLR     EA                      ; DISABLE INTERRUPTS TEMPORARILY
001B 0100      F   1647             AJMP    TMR1ISR
                   1648     ;
0023               1649             ORG     SINT
                   1650     ; SERIAL PORT INT VECTOR
0023 32            1651             RETI
                   1652 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                   1653             RSEG    FRED
                   1654     ;
                   1655     ; POWER ON ENTRY POINT
                   1656     ;
                   1657     ;
                   1658     RESTART:
0000 75D000        1659             MOV     PSW,#MAINBANK
0003 758100   F    1660             MOV     SP,#STACK-1            ; SET UP STACK
                   1661     ;
                   1662     ; CLEAR INTERNAL DATA
                   1663     ;
0006 E4            1664             CLR     A                      ;
0007 787F          1665             MOV     R0,#7FH                ; CLEAR ALL ONBOARD RAM
                   1666     CLRAM:
0009 F6            1667             MOV     @R0,A
000A D8FD          1668             DJNZ    R0,CLRAM               ; CLEAR TO 0
                   1669
000C 1100     F    1670             ACALL   SETUP
000E 020000   F    1671             JMP     START                  ; TEST FOR CE5 SHOW 6-5-83
                   1672     ;
                   1673     ;
                  1674 +1   $EJECT
```

```
LOC  OBJ           LINE     SOURCE

                   1675     ;
                   1676     ; INTERRUPT SERVICE ROUTINE FOR THE ENCODER RISING EDGE
                   1677     ;
                   1678     ENCODISR:
0011 C0D0          1679             PUSH    PSW
0013 75D010        1680             MOV     PSW,#ENCODBANK          ; SELECT THE ENCODER REGISTER BANK 1
                   1681     ; TEST FOR THE LEFT OR RIGHT INT
                   1682     ;
                   1683     EX1IN:
0016 209006        1684             JB      RIGHTLATCH,RIGHTENC     ; GO IF RIGHT ENCODER
0019 209211        1685             JB      LEFTLATCH,LEFTENC       ; GO IF LEFT RISING EDGE DETECTED
                   1686     ;
                   1687     ; ELSE SPURIOUS INT; JUST RETURN
                   1688     ;
001C D0D0          1689             POP     PSW
001E 32            1690             RETI                            ; RETURN FROM ISR
                   1691     ;
                   1692     RIGHTENC:
                   1693     RE1:
001F A88C          1694            MOV     R0RNEW,ENCODTIME                ; SINCE IT IS A RIGHT INT, PUT TIME
0021 D291          1695             SETB    RLATCHRESET             ; RESET THE RIGHT ENCODER LATCH
0023 C291          1696             CLR     RLATCHRESET             ; ACTIVE HIGH
0025 0500     F    1697             INC     RPOS                    ; BUMP THE MOTOR COUNTER
0027 209203        1698             JB      LEFTLATCH,LE1   ; MAKE SURE WE ARE IN LEVEL MODE
                   1699                                             ; GO DO LEFT IF NECESSARY
002A D0D0          1700             POP     PSW
002C 32            1701             RETI
                   1702     ;
                   1703     LEFTENC:
                   1704     LE1:
002D AA8C          1705             MOV     R2LNEW,ENCODTIME                ; SINCE IT IS A LEFT INT, PUT TIME
002F D293          1706             SETB    LLATCHRESET             ; RESET THE LEFT ENCODER LATCH
0031 C293          1707             CLR     LLATCHRESET             ; ACTIVE HIGH SO STROBE IT
0033 0500     F    1708             INC     LPOS                    ; BUMP THE LEFT MOTOR COUNTER
0035 2090E7        1709             JB      RIGHTLATCH,RIGHTENC     ; MAKE SURE WE ARE IN LEVEL MODE
                   1710                                             ; GO DO LEFT IF NECESSARY
0038 D0D0          1711             POP     PSW
003A 32            1712             RETI
                   1713 +1  $EJECT
```

```
LOC  OBJ            LINE     SOURCE

                    1714     ;
                    1715     ; ENCODER TIMER OVERFLOW ISR
                    1716     ;       FOR TIMER 0 INTERRUPT IT USES THE TIMER1 IN MODE3 SO ITS TF1)
                    1717     ;
                    1718     TIM1NGERROR:
0038 80FE           1719             SJMP    TIMINGERROR
                    1720
                    1721     TMR1ISR:
003D 2000FB   F     1722             JB      IN256?,TIM1NGERROR      ; DETECT ATTEMPTED REENTRANCY
0040 D200     F     1723             SETB    IN256?                  ; INDICATE ENTRANCY
0042 C0D0           1724             PUSH    PSW
0044 75D010         1725             MOV     PSW,#ENCODBANK          ; SELECT THE ENCODER REGISTER BANK 1
0047 300002   F     1726             JNB     SENDBIT?,NOSENDBIT
004A C2B5           1727             CLR     SENDBITLOC              ; SET THE BIT ACTIVE LOW
                    1728     ;
                    1729     ; THE FOLLOWING SECTION OF CODE MUST TAKE ONLY 4 CYCLES
                    1730     ; IT ACTS AS THE BIT ON TIME FOR THE IR DIODES
                    1731     ;
                    1732     NOSENDBIT:
004C 100002   F     1733             JBC     READBIT,DOBIT           ; IF BIT GOT THEN CLR
004F 8003           1734             SJMP    NODOBIT                 ; ELSE JUMP AROUND
                    1735     DOBIT:
0051 D200     F     1736             SETB    RECVBIT                 ; SAY GOT BIT
0053 00             1737             NOP                             ; THIS SECTION OF CODE MUST GO INT 4 CYCLES
                    1738     NODOBIT:
0054 D2B5           1739             SETB    SENDBITLOC              ; THEN OFF
                    1740
0056 C0E0           1741             PUSH    ACC
                    1742 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                    1743     ;
                    1744     ;
                    1745     ; FIND THE INTERVALS FOR THE ENCODER EDGES IF FLAGS SET
                    1746     ;
                    1747     CALCINTV:
                    1748
0058 20001C     F   1749             JB      LOVR?,LOVER             ; IF OVERFLOW THEN GO
005B BA0008         1750             CJNE    R2LNEW,#0,CALCLEFT      ; IF GOT AN ENCODER THEN GO
                    1751     INCLEFT:
005E 0C             1752             INC     LSTIME                  ; INC THE SOFTWARE ENCOD HIGH BYTE
005F BC0025         1753             CJNE    LSTIME,#0,CHKRIGHT
0062 D200       F   1754             SETB    LOVR?                   ; ELSE SET THE LEFT OVERFLOW FLAG
                    1755     L1:
0064 8021           1756             SJMP    CHKRIGHT                        ;       ELSE IGNORE
                    1757     ;
                    1758     CALCLEFT:
0066 EA             1759             MOV     AR2LNEW                         ; GET THE LAST ENCODER TIME
0067 C3             1760             CLR     C                       ; CLEAR FOR SUBTRACT
0068 9500       F   1761             SUBB    A,LOLD                  ; FIND THE DIFF(NEW - OLD)
006A F500       F   1762             MOV     LINTVLO,A               ; STORE THE LOW ENCODER EDGE INTERVAL
006C EC             1763             MOV     A,LSTIME
006D 9400           1764             SUBB    A,#0                    ; CALCULATE THE HIGH BYTE OF TIMER
                    1765     L2:
006F F500       F   1766             MOV     LINTVHI,A               ; STORE THE UPPER IN RAM
0071 8A00       F   1767             MOV     LOLD,R2LNEW             ; MOV NEW TO OLD FOR NEXT TIME
0073 7C01           1768             MOV     LSTIME,#TIMESTART               ; CLEAR TOP OF TIMER
0075 8010           1769             SJMP    CHKRIGHT                ; EXIT NOW
                    1770     ;
                    1771     LOVER:
0077 BA0002         1772             CJNE    R2LNEW,#0,LRESET
007A 8006           1773             SJMP    LV1
                    1774     ;
                    1775     LRESET:
007C 8A00       F   1776             MOV     LOLD,R2LNEW             ; RESET OLD TO NOW
007E 7C01           1777             MOV     LSTIME,#TIMESTART       ; CLEAR THE SOFTWARE HIGH TIMER BYTE
0080 C200       F   1778             CLR     LOVR?                   ; SET TO READY CONDITION
                    1779     LV1:
0082 E4             1780             CLR     A                       ; SET THE CURRENT INTERVAL TO 0
0083 F500       F   1781             MOV     LINTVLO,A
0085 F500       F   1782             MOV     LINTVHI,A               ;
                    1783 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                   1784     CHKRIGHT:
                   1785
0087 20001C    F   1786             JB      ROVR?,ROVER             ; IF OVERFLOW THEN GO
008A BB0008        1787             CJNE    R3RNEW,#0,CALCRIGHT     ; ELSE IF FLG SET CALCULATE THE RIGHT INTERVAL
                   1788     INCRIGHT:                               ;       ELSE INCREMENT THE SOFTWARE TIMER
008D 0D            1789             INC     RSTIME                  ; INC THE SOFTWARE ENCOD HIGH BYTE
008E B00025        1790             CJNE    RSTIME,#0,ALLRIGHT
                   1791     RIGHT1:
0091 D200      F   1792             SETB    ROVR?                   ; ELSE SET THE RIGHT OVERFLOW FLAG
0093 8021          1793             SJMP    ALLRIGHT                ;       ELSE INCREMENT THE SOFTWARE TIMER
                   1794     ;
                   1795     CALCRIGHT:
0095 E8            1796             MOV     A,R3RNEW                        ; GET THE LAST ENCODER TIME
0096 C3            1797             CLR     C                       ; CLEAR FOR SUBTRACT
0097 9500      F   1798             SUBB    A,ROLD                  ; FIND THE DIFF(NEW - OLD)
0099 F500      F   1799             MOV     RINTVLO,A               ; STORE THE LOW ENCODER EDGE INTERVAL
009B ED            1800             MOV     A,RSTIME
009C 9400          1801             SUBB    A,#0                    ; CALCULATE THE HIGH BYTE OF TIMER
                   1802     RIGHT2:
009E F500      F   1803             MOV     RINTVHI,A               ; STORE THE UPPER IN RAM
00A0 8B00      F   1804             MOV     ROLD,R3RNEW             ; MOV NEW TO OLD FOR NEXT TIME
00A2 7D01          1805             MOV     RSTIME,#TIMESTART       ; CLEAR TOP OF TIMER
00A4 8010          1806             SJMP    ALLRIGHT                ; EXIT NOW
                   1807     ;
                   1808     ROVER:
00A6 BB0002        1809             CJNE    R3RNEW,#0,RRESET
00A9 8006          1810             SJMP    RV1
                   1811     ;
                   1812     RRESET:
00AB 8B00      F   1813             MOV     ROLD,R3RNEW             ; RESET OLD TO NEW
00AD 7D01          1814             MOV     RSTIME,#TIMESTART       ; CLEAR THE SOFTWARE HIGH TIMER BYTE
00AF C200      F   1815             CLR     ROVR?                   ; SET TO READY CONDITION
                   1816     RV1:
00B1 E4            1817             CLR     A                       ; SET THE CURRENT INTERVAL TO 0
00B2 F500      F   1818             MOV     RINTVLO,A
00B4 F500      F   1819             MOV     RINTVHI,A               ;
                   1820 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                   1821     ;
                   1822     ; CHECK IF EITHER SOFTWARE COUNTER HAS OVERFLOWED
                   1823     ;
                   1824     ALLRIGHT:
00B6 7A00          1825              MOV     R2LNEW,#0               ; CLEAR ENCODER MEASURES
00B8 7B00          1826              MOV     R3RNEW,#0               ;
00BA 20B320        1827              JB      INT1,IOISR              ; IF NOT INT PENDING THEN DO IO
00BD 209005        1828              JB      RIGHTLATCH,RENC         ; GO IF RIGHT ENCODER
00C0 20920F        1829              JB      LEFTLATCH,LENC          ; GO IF LEFT RISING EDGE DETECTED
                   1830     ;
                   1831     ; ELSE SPURIOUS INT; JUST RETURN
                   1832     ;
00C3 8018          1833              SJMP    IOISR                   ; ELSE SPURIOUS CONTINUE
                   1834     ;
                   1835     RENC:
00C5 A88C          1836              MOV     R3RNEW,ENCODTIME                ; SINCE IT IS A RIGHT INT, PUT TIME
00C7 D291          1837              SETB    RLATCHRESET             ; RESET THE RIGHT ENCODER LATCH
00C9 C291          1838              CLR     RLATCHRESET             ; ACTIVE HIGH
00CB 0500     F    1839              INC     RPOS                    ; BUMP THE MOTOR COUNTER
00CD 209202        1840              JB      LEFTLATCH,LENC          ; MAKE SURE WE ARE IN LEVEL MODE
                   1841                                              ; GO DO LEFT IF NECESSARY
00D0 800B          1842              SJMP    IOISR
                   1843     ;
                   1844     LENC:
00D2 AA8C          1845              MOV     R2LNEW,ENCODTIME                ; SINCE IT IS A LEFT INT, PUT TIME
00D4 D293          1846              SETB    LLATCHRESET             ; RESET THE LEFT ENCODER LATCH
00D6 C293          1847              CLR     LLATCHRESET             ; ACTIVE HIGH SO STROBE IT
00D8 0500     F    1848              INC     LPOS                    ; BUMP THE LEFT MOTOR COUNTER
00DA 2090E8        1849              JB      RIGHTLATCH,RENC         ; MAKE SURE WE ARE IN LEVEL MODE
                   1850                                              ; GO DO LEFT IF NECESSARY
                   1851 +1  $EJ
```

```
LOC  OBJ            LINE     SOURCE

                    1852     IOISR:
                    1853     ;
                    1854     ; CHECK IF EITHER SOFTWARE COUNTER HAS OVERFLOWED
                    1855     ;
                    1856     ;
                    1857     ; RECIEVEBIT.
                    1858     ;       THIS ROUTINE IS PART OF THE TIMER X OVERLOW ISR
                    1859     ;       IT GETS A BIT FROM THE IR LINK IF WE ARE RECEVING BITS
                    1860     ;       THE BIT IS SHIFTED INTO A CHAR. A PACKET OF CHARS ARE RECIEVED THEN A FLAG IS
                    1861     ;       SET AND NO MORE BITS ARE GOTTEN TIL THE FLAG IS CLEARED.
                    1862     RECEIVEBITS:
00DD 300068   F     1863             JNB     RECV?,RECVEXIT  ; IF NOT RECEIVING CHARS THEN EXIT
00E0 200015   F     1864             JB      RSYNC?,GETSYNC  ; IF STILL GETTING SYNC THEN DO THAT
                    1865     GET8BITS:
00E3 300063   F     1866             JNB     R512?,RECVDONE  ; DO IT EVER 512 MICROSECONDS
00E6 200049   F     1867             JB      PARITY?,GETPAR  ; IF PARITY BIT TIME THEN GO
00E9 E500     F     1868             MOV     A,CURCHAR               ; ELSE GET THE CURRENT CHAR
00EB A200     F     1869             MOV     C,RECVBIT               ; GET THE CURRENT INPUT BIT FROM INT1 ISR
00ED C200     F     1870             CLR     RECVBIT                 ; CLR FOR NEXT TIME
00EF 13             1871             RRC     A                       ; ROTATE THE CARRY INTO CHAR
00F0 F500     F     1872             MOV     CURCHAR,A               ; SAVE CURRENT CHAR
00F2 4002           1873             JC      RECV1                   ; IF CARRY THEN STILL GETTIN G BITS
                    1874     SETGETPAR:
00F4 D200     F     1875             SETB    PARITY?         ; INDICATE GOT CHAR, NOW GET THE PARITY BIT
                    1876     RECV1:
00F6 8051           1877             SJMP    RECVDONE                ; EXIT FROM THIS GO THRU
                    1878 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                   1879     GETSYNC:
00F8 200014   F    1880             JB      STABLE?,STABLESYNC      ; IF HAVE GOT SNOUGH SYNC BITS THEN GO
00FB 30004D   F    1881             JNB     RECVBIT,RECVEXIT        ; ELSE IF NO GOT THEN EXIT
00FE C200     F    1882             CLR     RECVBIT                 ; CLEAR FOR NEXT TIME
0100 E500     F    1883             MOV     A,SYNCOUNT              ; GET THE CURRENT NUMBE OF SYNC BITS
0102 B40306        1884             CJNE    A,#MIDSYNC,STILLSYNC    ; IF GOT ENOUGH THEN WE ARE STABLE
                   1885     SETSTABLE:
                   1886     ; NEED TO SET THE R512FLAG FLAG
                   1887     ;
0105 C200     F    1888             CLR     R512?           ; SET THE 512 MICROSECOND FLAG
0107 D200     F    1889             SETB    STABLE?         ; INDICATE AGC SHOULD BE STABEL
0109 8000          1890             SJMP    STILLSYNC               ; GO ...
                   1891     STILLSYNC:
010B 0500     F    1892             INC     SYNCOUNT                ; ELSE BUMP THE SYNC COUNTER
                  1893     ;                                ; COULD SEE IF TOO MANY HERE
010D 803C          1894             SJMP    RECVEXIT                ; GO ...
                   1895     STABLESYNC:
010F 300037   F    1896             JNB     R512?,RECVDONE  ; IF NOT TIME THEN GO
0112 100009   F    1897             JBC     RECVBIT,STILLSTABLE     ; ELSE IF BIT THEN STILL SYNCING
0115 7405          1898             MOV     A,#MINSYNC              ; CHECK IF HAVE A MIN NUMBER OF SYNCS
0117 B50008   F    1899             CJNE    A,SYNCOUNT,BADSYNC?
                   1900     OKAYSYNC:
011A C200     F    1901             CLR     RSYNC?          ; ELSE IT'S THE START BIT
011C 802B          1902             SJMP    RECVDONE                ;    EXIT ...
                   1903     STILLSTABLE:
011E 0500     F    1904             INC     SYNCOUNT
0120 8027          1905             SJMP    RECVDONE                ; EXIT
                   1906     BADSYNC?:
0122 40F6          1907             JC      OKAYSYNC                ; ITS OKAY
0124 C200     F    1908             CLR     RECV?
0126 D200     F    1909             SETB    SYNCERROR
0128 8021          1910             SJMP    RECVEXIT
                   1911     ;       MOV     SYNCOUNT,#0             ; RESTART
                   1912     ;       CLR     RECVBIT
                   1913     ;       CLR     stable?
012A 801F          1914             SJMP    RECVEXIT
                   1915     ;
                   1916 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                    1917     GETPAR:
012C E500      F    1918             MOV     A,CURCHAR              ; GET THE CURRENT CHAR
012E A2D0           1919             MOV     C,P                    ; GET PARITY OF CURRENT CHAR
0130 100006    F    1920             JBC     RECVBIT,ODD            ; IF BIT GOTTEN SAYS ODD GO CHECK C FOR ODD
                    1921     EVEN:
0133 4006           1922             JC      GOODPAR                ; ELSE CHECK IF EVEN ... NC SAYS EVEN
                    1923     BADPAR:
0135 D200      F    1924             SETB    PARITYERROR            ; SET THE PARITY ERROR BIT
0137 8002           1925             SJMP    EXITPAR                ; GO CLEAN UP
                    1926     ODD:
0139 40FA           1927             JC      BADPAR                 ; IF NC THEN EVEN PAR SO ERROR
                    1928     GOODPAR:
                    1929     EXITPAR:
013B F7             1930             MOV     @R1PTR,A                       ; STOR THE CHAR
013C B90004    F    1931             CJNE    R1PTR,#LASTCHAR,NOTFULL ; IF NO FULL THEN GO
013F C200      F    1932             CLR     RECV?          ; DON'T RECEIVE ANY MORE TIL NEXT PACKET
0141 8008           1933             SJMP    RECVEXIT               ; EXIT
                    1934     NOTFULL:
0143 09             1935             INC     R1PTR                  ; BUMP POINTER TO NEXT CHAR
0144 C200      F    1936             CLR     PARITY?        ; GET A NOTHER CHAR
0146 75007F    F    1937             MOV     CURCHAR,#STARTCHAR     ; SET CHAR TO INDICATE EMPTY
                    1938     ;
                    1939     ; EXIT THE RECEIVE ISR
                    1940     ;
                    1941     RECVDONE:
0149 B200      F    1942             CPL     R512?          ; CHANGE STATE OF CONTROLLER FOR NEXT ROUND
                    1943     RECVEXIT:
                    1944 +1  $EJ
```

```
LOC  OBJ            LINE     SOURCE

                    1945     ;
                    1946     ;SENDBIT
                    1947     ;       IF WE ARE SENDING, THHEN OUTPUT THE SYNC BURST THEN 16 CHARS
                    1948     ;
                    1949     SENDBIT:
014B C200       F   1950             CLR     SENDBIT?                ; ALWAYS CLEAR
014D 30003F     F   1951             JNB     SEND?,SENDEXIT          ; IF NOT SENDING THEN EXIT
0150 30003A     F   1952             JNB     S512?,SENDONE           ; IF NOT SENDING THIS 256 FRAME THEN GO
0153 20002C     F   1953             JB      SYNC?,SENDSYNC          ; IF SENDING SYNC THEN GO
0156 E500       F   1954             MOV     A,CURCHAR
0158 B40012         1955             CJNE    A,#0,SEND1              ; A 0 IS THE CHAR SENT CONDITION
015B E9             1956             MOV     A,R1PTR                 ; ELSE SEE OF AT END
015C B50004     F   1957             CJNE    A,LASTPTR,GETSCHAR      ; IF NOT AT END THEN CONT
                    1958     SENTLAST:
015F C200       F   1959             CLR     SEND?                   ; ELSEET TO NO SEND CUZ DONE
0161 802C           1960             SJMP    SENDEXIT
                    1961     GETSCHAR:
0163 E7             1962             MOV     A,@R1PTR                        ; GET NEXT CHAR
0164 A200           1963             MOV     C,P                     ; GET PARITY
0166 B3             1964             CPL     C                       ; ODD PARITY IT
0167 9200       F   1965             MOV     SPARBIT,C               ; SAVE PARITY FOR LATER
0169 09             1966             INC     R1PTR                   ; POINT TO NEXT CHAR
016A D3             1967             SETB    C                       ; SET FOR TERMINAL CONDITON
016B 8001           1968             SJMP    SEND2                   ; GO INTO MIDDLE
                    1969     SEND1:
016D C3             1970             CLR     C                       ; CLEAR TO SHIFT IN ZERO FOR DOEN CAHR CONDITION
                    1971     SEND2:
016E 13             1972             RRC     A                       ; GET NEXT BIT TO SEND
016F 600A           1973             JZ      SENDPAR                 ; IF CHAR IS ZERO THEN SEND PARITY
0171 F500       F   1974             MOV     CURCHAR,A               ; SAVE THE CURRENT CHAR
                    1975     SNDB:
0173 5002           1976             JNC     BITOFF                  ; IF BIT IS LOW THEN GO
                    1977     SNDBITS:
0175 D200       F   1978             SETB    SENDBIT?                ; SET TO SEND BIT
                    1979     BITOFF:
                    1980     SENDDONE:
                    1981     NOSENDNOW:
0177 B200       F   1982             CPL     S512?                   ; SET TO OPPOSITE FOR NEXT LOOP
0179 8014           1983             SJMP    SENDEXIT
                    1984 +1  $EJ
```

```
LOC  OBJ            LINE     SOURCE

                    1985     SENDPAR:
017B A200       F   1986             MOV     C,SPARBIT               ; GET THE PARITY BIT TO SEND
017D 750000     F   1987             MOV     CURCHAR.#0              ; MAKE SURE DONE
0180 80F1           1988             SJMP    SNDB                    ; SEND THE BIT
                    1989     SENDSYNC:
0182 E500       F   1990             MOV     A,SYNCOUNT              ; GET THE NUMBER OF SYNCS SENT
0184 0500       F   1991             INC     SYNCOUNT                ; BUMP TO NEXT
0186 B405EC         1992             CJNE    A,#NUMBEROFSYNCS,SNDBITS        ; IF NOT DONE THEN SEND
                    1993                                                     ; ELSE ASSUME CLR TO SEND START
                    1994                                             ; THIS EFFECTIVELY SENDS START BIT
0189 C200       F   1995             CLR     SYNC?
018B 80EA           1996             SJMP    SENDDONE
                    1997     ;
                    1998     ; EXIT THE SEND ISR
                    1999     ;
                   2000     SENDONE:
018D B200       F   2001             CPL     S512?
                    2002     SENDEXIT:
                    2003
                    2004 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                   2005     MOTORISR:
                   2006
                   2007     MOT1:
01BF E500      F   2008             MOV     A,CURSTATE              ; GET CURRENT STATE
0191 B50002    F   2009             CJNE    A,LEFTSPD,MOT2          ; SEE IF TURN ON
0194 C296          2010             CLR     LENABLE
                   2011     MOT2:
0196 B50002    F   2012             CJNE    A,RIGHTSPD,MOT3
0199 C294          2013             CLR     RENABLE
                   2014     MOT3:
019B D50012    F   2015             DJNZ    CURSTATE,EXITISR                ; DEC MOTOR STATE; IF NOT 0 THEN GO
019E 75007F    F   2016             MOV     CURSTATE,#STRTSTATE     ; ELSE GET NEW START
01A1 D296          2017             SETB    LENABLE                 ; DISABLE MOTOR
01A3 D294          2018             SETB    RENABLE                 ; SAME FOR RIGHT
                   2019     ;
                   2020     ;
01A5 7400      F   2021             MOV     A,#LOW DAVEISR
01A7 C0E0          2022             PUSH    ACC
01A9 7400      F   2023             MOV     A,#HIGH DAVEISR
01AB C0E0          2024             PUSH    ACC
01AD C200      F   2025             CLR     IN256?                  ;LEAVING THIS 256 US ISR
01AF 32            2026             RETI                                    ; DO A SOFTWARE INTERRUPT
                   2027 +1  $EJ
```

```
LOC  OBJ           LINE     SOURCE

                   2028     EXITISR:
01B0 D0E0          2029              POP     ACC
01B2 D0D0          2030              POP     PSW
01B4 C200      F   2031              CLR     IN256?
01B6 32            2032              RETI
                   2033 +1  $EJECT
```

```
LOC  OBJ           LINE     SOURCE

                   2034             USING   PROGBANK
                   2035     ;
                   2036     ; INITIALIZE ALL STUFF... ASSUME ALL INTERNAL RAM IS CLEAR
                   2037     ;
                   2038     ;
                   2039     ;
                   2040     SETUP:
                   2041     ;
                   2042     ; ENABLE OR DISABLE INTERRUPTS
                   2043     ;
                   2044     ;       CLR     EA                      ; DISABLE ALL INTERRUPTS AND SET UP
                   2045     ;       CLR     ES                      ; NO SERIAL INTERRUPTS(UNUSED)
                   2046     ;       CLR     ET0                     ; NO TIMER 1 INTERRUPTS(UNUSED)
                   2047     ;       SETB    EX1                     ; EXTERNAL 1 INTERRUPTS(IR RECV)
                   2048     ;       SETB    ET1                     ; YES WE WANT ENCODER TIMER
                   2049     ;       SETB    EX0                     ; YES ENCODER INTERUPTS
  000D             2050     INTDIS  EQU     000011B                 ; SEE ABOVE FOR VALUES
01B7 75A80D        2051             MOV     IE,#INTDIS
                   2052     ;
                   2053     ; SETUP MOTOR STUFF
                   2054     ;
01BA 7590F0        2055             MOV     P1,#0F0H                ; DISABLE MOTORS,CLR LATCHRESET
01BD 7590FF        2056             MOV     P1,#0FFH                ; DISABLE MOTORS,RESET LATCH
01C0 7590F5        2057             MOV     P1,#0F5H                ; DISABLE MOTORS, CLR RESET,LATCH INPUT
                   2058
                   2059     ;       8 BIT UART MODE (VARIABLE BAUD RATE)
                   2060     ;       & SET TRANSMIT READY BIT
                   2061     ;
01C3 759852        2062             MOV     SCON,#SERIALMODE
                   2063     ;
                   2064     ; TIINTI INITIALIZE TIMER 1 FOR AUTORELOAD AT 32*2400 HZ
                   2065     ;       ( FOR USE AS 8-BIT AUTO-RELOAD COUNTER FOR RS232 SERIAL)
01C6 7588D2        2066             MOV     TCON,#11010010B         ;CONFIGURE TIMER, AND OFF IT
01C9 758DF3        2067             MOV     TH1,#BAUDRATE           ;0F3H COUNT RELOAD
01CC 758923        2068             MOV     TMOD,#T1INIT+T0INIT     ; SETUP THE TIMERS RIGHT
01CF 758700        2069             MOV     87H,#00H                ; SELECT REGULAR  NOT TIMES 2 BAUD (PCON)
                   2070     ;
                   2071     ;
                   2072     ; MEMORY TEST
                   2073     ;
01D2 120000   F    2074             CALL    MEMTEST
                   2075
                   2076     ;
                   2077     ; INIT THE 8255A
                   2078     ;
                   2079 +1
                   2080 +1
                   2081 +2  ; STORX PIACTL,PIAINIT
01D5 908000        2082 +2                  MOV     DPTR,#PIACTL
01D8 7442          2083 +2                  MOV     A,#PIAINIT
01DA F0            2084 +1                  MOVX    @DPTR,A                 ; INIT THE PIA
                   2085 +1
                   2086 +1
                   2087 +2  ; STORX PORTA,PORTAINIT
```

```
LOC  OBJ           LINE     SOURCE

01DB 908001        2088 +2                MOV     DPTR,#PORTA
01DE 7400          2089 +2                MOV     A,#PORTAINIT
01E0 F0            2090 +1                MOVX    @DPTR,A                         ; INIT PORT A
                   2091 +1
                   2092 +1
                   2093 +2  ; STORX PORTB,PORTBINIT
01E1 908002        2094 +2                MOV     DPTR,#PORTB
01E4 740A          2095 +2                MOV     A,#PORTBINIT
01E6 F0            2096 +1                MOVX    @DPTR,A                         ; SAME FOR B
                   2097 +1
                   2098 +2
                   2099 +2
                   2100 +3  ; STORX PORTBMASK,PORTBINIT
01E7 900000    F   2101 +3                MOV     DPTR,#PORTBMASK
01EA 740A          2102 +3                MOV     A,#PORTBINIT
01EC F0            2103 +2                MOVX    @DPTR,A
                   2104 +1
                   2105 +1
                   2106 +1
                   2107 +2  ; STORX PORTC,PORTCINIT
01ED 908003        2108 +2                MOV     DPTR,#PORTC
01F0 7400          2109 +2                MOV     A,#PORTCINIT
01F2 F0            2110 +1                MOVX    @DPTR,A                         ; SAME FOR C
                   2111     ;
01F3 D200      F   2112             SETB    LOVR?                   ; SET SO NEED EDGE TO START
01F5 D200      F   2113             SETB    ROVR?
                   2114     ;
                   2115     ; SET THE EXTERNAL TRIGGER CONDITIONS IN TCON
                   2116     ;
01F7 D28E          2117             SETB    TR1                     ; SET TRANSMIT TIMER GOING
01F9 C288          2118             CLR     IT0                     ; THE ENCODER INRTERRUPT IS LEVEL TRIGGERED
01FB D28A          2119             SETB    IT1                     ; IR INT IS EDGEDI
                   2120
                   2121     ;       SETB    INT0                    ; PROGRSAM FOR INPUT
                   2122     ;       SETB    INT1                    ; SAME
                   2123     ;       SETB    RXD                     ; JUST IN CASE
                   2124     ;       SETB    SENDBITLOC              ; ACTIVE LOW -- OFF IT
01FD 7580FF        2125             MOV     P3,#11111111B           ; SET PORT PINS TO ABOVE VALUES
                   2126
                   2127     ;
                   2128     ; SET THE INTERRUPTS PRIORITIES
                   2129     ;
                   2130     ;       CLR     PS                      ; LOW PRIORITY SERAIL INTERRUPT (UNUSED)
                   2131     ;       CLR     PT0                     ; LOW PRIOR FOR TIMER 0
                   2132     ;       CLR     PX1                     ; LOW  PRIOR EXT 1 (IR RECV)
                   2133     ;       SETB    PT1                     ; HI  PRIOR TIMER 1 ENCODER TIMER
                   2134     ;       CLR     PX0                     ; LOW  PRIOR FOR EXTERNAL 0 ENCODER INT
  0008             2135     INTPRIOR        EQU     00001000B       ; SEE ABOVE FOR VALUES
0200 758808        2136             MOV     IP,#INTPRIOR            ; SET INT PROIRITY
                   2137
0203 758C00        2138             MOV     ENCODTIME,#0            ; CLEAR THE TIMER TO START AT 0
0206 7500FF    F   2139             MOV     STATCLK,#STATSTRT       ; INIT MOTOR STAT CLOCK
0209 75007F    F   2140             MOV     CURSTATE,#STRTSTATE
                   2141     ;
```

```
LOC  OBJ            LINE     SOURCE

020C 120000   F     2142           LCALL   INITPARAMETERS  ;SET UP DAVE'S STUFF
020F 120000   F     2143           LCALL   INITMOTOR              ; DO TO INIT IT
                    2144     ; START WITH A BANG ( OR IS IT A WHIMPER....)
                    2145
0212 D2AF           2146           SETB    EA                     ; ENABLE ALL INTERRUPTS AND AWAY WE GO
0214 22             2147           RET
                    2148 +1  $EJ
```

```
LOC  OBJ            LINE     SOURCE

                    2149     ;
                    2150     ; MEMORY TEST
                    2151     ;
                    2152     MEMTEST:
0215 7A13           2153             MOV     R2,#13H
0217 7B7A           2154             MOV     R3,#7AH
0219 7CD1           2155             MOV     R4,#0D1H
021B 7D37           2156             MOV     R5,#37H
021D 7EAD           2157             MOV     R6,#0ADH
                    2158
021F 120000   F     2159             CALL    FILLRAM
0222 120000   F     2160             CALL    CHECKRAM
0225 402A           2161             JC      RAMERROR
                    2162
0227 7AF2           2163             MOV     R2,#NOT 13
0229 7B85           2164             MOV     R3,#NOT 7AH
022B 7C2E           2165             MOV     R4,#NOT 0D1H
022D 7DC8           2166             MOV     R5,#NOT 37H
022F 7E52           2167             MOV     R6,#NOT 0ADH
                    2168
0231 120000   F     2169             CALL    FILLRAM
0234 120000   F     2170             CALL    CHECKRAM
0237 4018           2171             JC      RAMERROR
                    2172     ;
                    2173     ; CLEAR XDATA
                    2174     ;
0239 E4             2175             CLR     A
023A 908000         2176             MOV     DPTR,#XRAM              ; CLEAR ALL XDATA
                    2177     CLRX:
023D F0             2178             MOVX    @DPTR,A                 ; SET NEXT OT ZERO
023E A3             2179             INC     DPTR                    ; BUMP TO NEXT
023F B582FB         2180             CJNE    A,DPL,CLRX              ; IF NOT ZERO THEN CRUISE
0242 A883           2181             MOV     R0,DPH
0244 B881F6         2182             CJNE    R0,#HIGH(MAXRAM),CLRX   ; GO TIL DONE
                    2183
0247 7444           2184             MOV     A,#44H                  ; POINT TO LOGIC AREA
0249 900000   F     2185             MOV     DPTR,#TRACLOC
024C F0             2186             MOVX    @DPTR,A
                    2187
024D A3             2188             INC     DPTR
024E E4             2189             CLR     A
024F F0             2190             MOVX    @DPTR,A
                    2191
0250 22             2192             RET
                    2193     RAMERROR:
0251 900000   F     2194             MOV     DPTR,#?COMPERR
0254 120000   F     2195             CALL    SHOLOOP
0257 80FE           2196             SJMP    $
                    2197     ;
                    2198     ; FILLRAM
                    2199     ;
                    2200     FILLRAM:
0259 908000         2201             MOV     DPTR,#XRAM
                    2202     FL1:
```

```
LOC  OBJ            LINE     SOURCE

025C EA             2203             MOV     A,R2
025D F0             2204             MOVX    @DPTR,A
025E A3             2205             INC     DPTR
025F EB             2206             MOV     A,R3
0260 F0             2207             MOVX    @DPTR,A
0261 A3             2208             INC     DPTR
0262 EC             2209             MOV     A,R4
0263 F0             2210             MOVX    @DPTR,A
0264 A3             2211             INC     DPTR
0265 ED             2212             MOV     A,R5
0266 F0             2213             MOVX    @DPTR,A
0267 A3             2214             INC     DPTR
0268 EE             2215             MOV     A,R6
0269 F0             2216             MOVX    @DPTR,A
026A A3             2217             INC     DPTR
                    2218
026B E583           2219             MOV     A,DPH
026D B480EC         2220             CJNE    A,#HIGH MAXRAM-1,FL1
0270 E582           2221             MOV     A,DPL
0272 B4FB02         2222             CJNE    A,#LOW MAXRAM-5,FL2
0275 8002           2223             SJMP    FLEX
                    2224     FL2:
0277 40E3           2225             JC      FL1
                    2226     FLEX:
0279 22             2227             RET
                    2228     ;
                    2229     ; CHECKRAM
                    2230     ;
                    2231     CHECKRAM:
027A 908000         2232             MOV     DPTR,#XRAM
                    2233     CR1:
027D E0             2234             MOVX    A,@DPTR
027E B50225         2235             CJNE    A,AR2,BADRAM
0281 A3             2236             INC     DPTR
                    2237
0282 E0             2238             MOVX    A,@DPTR
0283 B50320         2239             CJNE    A,AR3,BADRAM
0286 A3             2240             INC     DPTR
                    2241
0287 E0             2242             MOVX    A,@DPTR
0288 B50418         2243             CJNE    A,AR4,BADRAM
028B A3             2244             INC     DPTR
                    2245
028C E0             2246             MOVX    A,@DPTR
028D B50516         2247             CJNE    A,AR5,BADRAM
0290 A3             2248             INC     DPTR
                    2249
0291 E0             2250             MOVX    A,@DPTR
0292 B50611         2251             CJNE    A,AR6,BADRAM
0295 A3             2252             INC     DPTR
                    2253
0296 E583           2254             MOV     A,DPH
0298 B480E2         2255             CJNE    A,#HIGH MAXRAM-1,CR1
029B E582           2256             MOV     A,DPL
```

```
LOC  OBJ           LINE     SOURCE

029D B4FB02        2257             CJNE    A,#LOW MAXRAM-5,CR2          ; SAFETY
02A0 8002          2258             SJMP    CREX
                   2259     CR2:
02A2 40B8          2260             JC      FL1
                   2261     CREX:
02A4 C3            2262             CLR     C
02A5 22            2263             RET
                   2264     BADRAM:
02A6 D3            2265             SETB    C
02A7 22            2266             RET
                   2267
                   2268             END

REGISTER BANK(S) USED: 0 2

ASSEMBLY COMPLETE, NO ERRORS FOUND
```