```
0001   0000             ;Test programs for 8-bit TTL computer.
0002   0000                        .ORG    0000H          ;Get address from switches and jump to it
0003   0000 10 13                  LDI     13H            ;JMP instruction
0004   0002 12 00 08               STM     0800H          ;Start of RAM
0005   0005 17 00                  IN      00H            ;Low byte of jump target
0006   0007 12 01 08               STM     0801H
0007   000A 17 01                  IN      01H            ;High byte of jump target
0008   000C 12 02 08               STM     0802H          ;Full jump instruction in place now
0009   000F 13 00 08               JMP     0800H          ;Jump to the jump instruction
0010   0012             ;Simple port reflector
0011   0012 17 00       LOOP:      IN      00H
0012   0014 18 00                  OUT     00H
0013   0016 17 01                  IN      01H
0014   0018 18 01                  OUT     01H
0015   001A 13 12 00               JMP     LOOP
0016   001D             ;Simple counter -- run with slow clock
0017   001D 10 00                  LDI     00H
0018   001F 18 00       LOOPA:     OUT     00H
0019   0021 19                     INC
0020   0022 13 1F 00               JMP     LOOPA
0021   0025             ;Two-byte up counter -- run with fast clock
0022   0025 10 00                  LDI     00H
0023   0027 12 00 08               STM     0800H          ;Hi byte
0024   002A 18 01                  OUT     01H            ;Clear port 1 LEDs
0025   002C 18 00       LOOPB:     OUT     00H            ;Output low byte
0026   002E 19                     INC
0027   002F 14 35 00               JPZ     NEXTB          ;If zero, jump to increment high byte
0028   0032 13 2C 00               JMP     LOOPB          ;Low-byte increment loop
0029   0035 11 00 08    NEXTB:     LDM     0800H          ;Get high byte from memory
0030   0038 19                     INC
0031   0039 18 01                  OUT     01H            ;Output high byte to port 1 LEDs
0032   003B 12 00 08               STM     0800H          ;Store high byte
0033   003E 10 00                  LDI     00H
0034   0040 13 2C 00               JMP     LOOPB          ;Go back to low-byte increment loop
0035   0043             ;8-bit highest factor routine
0036   0043             ;Factor test
0037   0043 17 00       FACSTRT:   IN      00H            ;Number to factor
0038   0045 12 09 08               STM     ORIG
0039   0048 12 0A 08               STM     TESTF
0040   004B 11 0A 08    LOOP15:    LDM     TESTF
0041   004E 1A                     DEC
0042   004F 12 0A 08               STM     TESTF
0043   0052 11 09 08               LDM     ORIG
0044   0055 02 0A 08    LOOP16:    SUB     TESTF
0045   0058 14 61 00               JPZ     DONE           ;Factor found
0046   005B 16 55 00               JPC     LOOP16  ;For A - B, carry set if A >= B
0047   005E 13 4B 00               JMP     LOOP15  ;No carry, means A < B, and not a factor
0048   0061 11 0A 08    DONE:      LDM     TESTF
0049   0064 18 00                  OUT     00H
0050   0066 13 43 00               JMP     FACSTRT
0051   0069             ;Test for serial interface with 8-bit code
0052   0069             ;Set port to 9600 baud, 8-bit, no parity, 1 stop bit
0053   0069 10 4E                  LDI     4EH            ;1 stop bit, no parity, 8-bit char, 16x baud
0054   006B 18 03                  OUT     03H            ;write to UART control port
```

```
0055   006D 10 37                  LDI    37H           ;enable receive and transmit
0056   006F 18 03                  OUT    03H           ;write to control port
0057   0071 17 03       LOOP1:     IN     03H           ;get status
0058   0073 0C 02                  ANDIM  02H           ;check RxRDY bit
0059   0075 14 71 00               JPZ    LOOP1         ;not ready, loop
0060   0078 17 02                  IN     02H           ;get char from data port
0061   007A 18 00                  OUT    00H           ;put on LEDs
0062   007C 12 06 08               STM    TEMP          ;store the character
0063   007F 17 03       LOOP2:     IN     03H           ;get status
0064   0081 0C 01                  ANDIM  01H           ;check TxRDY bit
0065   0083 14 7F 00               JPZ    LOOP2         ;loop if not ready
0066   0086 11 06 08               LDM    TEMP          ;get char back
0067   0089 18 02                  OUT    02H           ;send to UART for output
0068   008B 13 71 00               JMP    LOOP1         ;start over
0069   008E             ;Program loader
0070   008E             ;Takes input from serial port, creates byte values from hex character pairs
0071   008E             ;Loads byte values sequentially into RAM starting at 0x0810
0072   008E             ;Jumps to location 0x0810 to start execution upon receiving return character
0073   008E             ;Quits without execution if invalid hex character input received
0074   008E             ;Setup routine for serial port
0075   008E 10 4E                  LDI    4EH           ;1 stop bit, no parity, 8-bit char, 16x baud
0076   0090 18 03                  OUT    03H           ;write to UART control port
0077   0092 10 37                  LDI    37H           ;enable receive and transmit
0078   0094 18 03                  OUT    03H           ;write to control port
0079   0096             ;Need to put instruction to store bytes in RAM so can increment the target address
0080   0096 10 12                  LDI    12H           ;STM instruction
0081   0098 12 00 08               STM    STORE_BYTE
0082   009B 10 10                  LDI    10H           ;Low byte of storage buffer start address
0083   009D 12 01 08               STM    STORE_BYTE+1
0084   00A0 10 08                  LDI    08H           ;Hi byte of storage buffer start address
0085   00A2 12 02 08               STM    STORE_BYTE+2
0086   00A5             ;Need to set return jump after STORE_BYTE
0087   00A5 10 13                  LDI    13H           ;JMP instruction for return
0088   00A7 12 03 08               STM    STORE_BYTE+3
0089   00AA 11 71 01               LDM    RETURN  ;Lo byte of return address
0090   00AD 12 04 08               STM    STORE_BYTE+4
0091   00B0 11 72 01               LDM    RETURN+1      ;Hi byte of return address
0092   00B3 12 05 08               STM    STORE_BYTE+5
0093   00B6 17 03       GET_HI:    IN     03H           ;Get hi-order nybble of pair
0094   00B8 0C 02                  ANDIM  02H           ;check RxRDY bit
0095   00BA 14 B6 00               JPZ    GET_HI  ;not ready, loop
0096   00BD 17 02                  IN     02H           ;get char from data port
0097   00BF 12 06 08               STM    TEMP          ;Store character
0098   00C2 0A 0D                  SUBIM  0DH           ;Carriage return?
0099   00C4 14 85 01               JPZ    RUN           ;Yes, run code
0100   00C7 17 03       LOOP3:     IN     03H           ;No, output character and validate
0101   00C9 0C 01                  ANDIM  01H           ;check TxRDY bit
0102   00CB 14 C7 00               JPZ    LOOP3         ;loop if not ready
0103   00CE 11 06 08               LDM    TEMP          ;get char back
0104   00D1 18 02                  OUT    02H           ;send to UART for output
0105   00D3             ;Code to validate hex character
0106   00D3 0F 30                  CMP    30H           ;Lower limit of hex characters
0107   00D5 16 DB 00               JPC    NEXT1         ;Char >= 30H, possibly valid
0108   00D8 13 F9 00               JMP    INVALID ;Char < 30H, invalid hex char
```

```
0109   00DB 0F 47       NEXT1:       CMP    47H            ;ASCII for "G"
0110   00DD 16 F9 00                 JPC    INVALID ;Char is G or greater, invalid
0111   00E0 0F 41                    CMP    41H            ;ASCII for "A"
0112   00E2 16 ED 00                 JPC    VALIDAF_HI     ;Char is valid A-F
0113   00E5 0F 3A                    CMP    3AH            ;ASCII for ":"
0114   00E7 16 F9 00                 JPC    INVALID ;Char is ":" or greater, but < "A", invalid
0115   00EA 13 F4 00                 JMP    VALID09_HI     ;Char is valid 0-9
0116   00ED 0C 0F       VALIDAF_HI:  ANDIM  0FH            ;Mask off high bits
0117   00EF 08 09                    ADDIM  9              ;Adjust ASCII to binary value
0118   00F1 13 FC 00                 JMP    SHIFT_HI
0119   00F4 0C 0F       VALID09_HI:  ANDIM  0FH            ;Mask off high bits
0120   00F6 13 FC 00                 JMP    SHIFT_HI
0121   00F9 13 88 01    INVALID:     JMP    ERROR          ;Invalid hex char, quit
0122   00FC 12 07 08    SHIFT_HI:    STM    BYTE           ;Will eventually contain the byte to load
0123   00FF 12 06 08                 STM    TEMP           ;Value to add
0124   0102 10 10                    LDI    10H            ;Multiply x 16 to shift into high-order nybble
0125   0104 12 08 08                 STM    COUNTER
0126   0107 11 08 08    MULTLOOP:    LDM    COUNTER
0127   010A 1A                       DEC
0128   010B 14 1D 01                 JPZ    GET_LO  ;Have added 16 times, done
0129   010E 12 08 08                 STM    COUNTER
0130   0111 11 06 08                 LDM    TEMP           ;Original nybble
0131   0114 00 07 08                 ADD    BYTE           ;Add to BYTE and store
0132   0117 12 07 08                 STM    BYTE
0133   011A 13 07 01                 JMP    MULTLOOP       ;Keep adding
0134   011D 17 03       GET_LO:      IN     03H            ;Get lo-order nybble of pair
0135   011F 0C 02                    ANDIM  02H            ;check RxRDY bit
0136   0121 14 1D 01                 JPZ    GET_LO  ;not ready, loop
0137   0124 17 02                    IN     02H            ;get char from data port
0138   0126 12 06 08                 STM    TEMP           ;Store character
0139   0129 17 03       LOOP4:       IN     03H            ;Output character
0140   012B 0C 01                    ANDIM  01H            ;check TxRDY bit
0141   012D 14 29 01                 JPZ    LOOP4
0142   0130 11 06 08                 LDM    TEMP           ;When ready, retrieve character and output
0143   0133 18 02                    OUT    02H
0144   0135 17 03       LOOP5:       IN     03H
0145   0137 0C 01                    ANDIM  01H
0146   0139 14 35 01                 JPZ    LOOP5
0147   013C 10 20                    LDI    20H            ;Space character
0148   013E 18 02                    OUT    02H            ;send to UART for output
0149   0140             ;Code to validate hex character
0150   0140 11 06 08                 LDM    TEMP           ;Retrieve character and validate
0151   0143 0F 30                    CMP    30H            ;Lower limit of hex characters
0152   0145 16 4B 01                 JPC    NEXT2          ;Char >= 30H, possibly valid
0153   0148 13 F9 00                 JMP    INVALID ;Char < 30H, invalid hex char
0154   014B 0F 47       NEXT2:       CMP    47H            ;ASCII for "G"
0155   014D 16 F9 00                 JPC    INVALID ;Char is G or greater, invalid
0156   0150 0F 41                    CMP    41H            ;ASCII for "A"
0157   0152 16 5D 01                 JPC    VALIDAF_LO     ;Char is valid A-F
0158   0155 0F 3A                    CMP    3AH            ;ASCII for ":"
0159   0157 16 F9 00                 JPC    INVALID ;Char is ":" or greater, but < "A", invalid
0160   015A 13 67 01                 JMP    VALID09_LO     ;Char is valid 0-9
0161   015D 0C 0F       VALIDAF_LO:  ANDIM  0FH            ;Mask off high bits
0162   015F 08 09                    ADDIM  9              ;Now lo nybble correct
```

```
0163   0161 00 07 08                ADD    BYTE           ;Combine with hi nybble stored in BYTE
0164   0164 13 6C 01                JMP    STORE          ;Store the byte in RAM
0165   0167 0C 0F       VALID09_LO: ANDIM  0FH            ;Mask off high bits
0166   0169 00 07 08                ADD    BYTE           ;Now full byte assembled
0167   016C 18 00       STORE:      OUT    00H            ;Display on LEDs
0168   016E 13 00 08                JMP    STORE_BYTE     ;Store the byte in RAM
0169   0171 73 01       RETURN:     .DW    $+2            ;Address to return from storage instruction
0170   0173 11 01 08                LDM    STORE_BYTE+1;Increment byte pointer, lo byte first
0171   0176 19                      INC
0172   0177 12 01 08                STM    STORE_BYTE+1
0173   017A 11 02 08                LDM    STORE_BYTE+2;Increment hi byte if a carry occurred when lo byte incremented
0174   017D 09 00                   ADCIM  00H
0175   017F 12 02 08                STM    STORE_BYTE+2
0176   0182 13 B6 00                JMP    GET_HI
0177   0185 13 10 08    RUN:        JMP    0810H          ;Run program
0178   0188 18 00       ERROR:      OUT    00H    ;Display erroneous character on LEDs
0179   018A 13 8A 01    HALT:       JMP    HALT           ;Halt
0180   0800                         .ORG   0800H
0181   0800 000000000000STORE_BYTE: .DB 0,0,0,0,0,0       ;Six spaces for storage instruction and return
0182   0806 00          TEMP:       .DB    00H            ;Temp storage for character, data
0183   0807 00          BYTE:       .DB    00H            ;For multiplication (shifting)
0184   0808 00          COUNTER:    .DB    00H            ;For multiplication (shifting)
0185   0809 00          ORIG:       .DB    00H
0186   080A 00          TESTF:      .DB    00H
0187   080B                         .END
tasm: Number of errors = 0
```