MC68705U3 Bootstrap ROM

```
;This is a listing of the Bootstrap ROM which resides in Motorola's MC68705U3 single chip
;micros. Its sole purpose is to program its own EPROM by copying the data from an external
;EPROM (2732) which has been programmed with an exact duplicate of the required information
;(refer to the MC68705U3/R3 data sheets for more info).
;I obtained the listing by dumping the contents of the ROM and disassembling it.
;I believe the ROM in the MC68705R3 is identical but I haven't checked it however ROM size
;is 120 bytes in both devices.
;Peter Ihnat, Oct 2010.      pihnat@uow.edu.au


;In Motorola's suggested Programmer, PortA is used to read data from the external EPROM.
;The 4 lower bits of PortB are used as outputs with the following functions:
;     PB4 = reset the 4040 counter (1 = reset)
;     PB3 = clock the 4040 counter (clks on falling edge)
;     PB2 = 'verified' LED (0 = ON)
;     PB1 = 'programmed' LED (0 = ON)
;     PB0 = 0 applies +21V to the micro's Vpp input (pin 7). PB0 = 1 applies +5V.
;               +21V is used when programming and +5V is used when verifying.


;In this listing the code appears twice. That's because even though the Bootstrap ROM is the
;120 bytes starting from address 0F80H the first thing it does when it runs is to copy itself
;into RAM. This is so it can modify itself during execution.
;The first listing is the Bootstrap ROM exactly as it appears in the 68705U3 memory map.
;I've disassembled the first 14 bytes which is the part that does the block move.
;The second listing is the Bootstrap ROM located in its new position in RAM.
;This is fully disassembled.
```

```
;Basic operation is as follows:
;When powered up, 12V on the TIMER input (pin 8) forces the micro to fetch the vector at
;0FF6H & 0FF7H and to start executing the code from that address (0F80H).
;First it copies itself to RAM then continues executing from address 0019H.
;It sets up PortB & removes the reset to the 4040 counter. It pulses the counter 128 times
;to skip the first 128 bytes (remember, the internal EPROM starts at 0080H).
;It then starts the programming loop:
;    read the external EPROM, clock the 4040 counter,
;    increment the address pointer to the internal EPROM, store the data at that address,
;    apply the programming voltage for the correct length of time
;It loops until the whole external EPROM has been copied.
;Then it modifies some instructions so that the programming part of the code does a verify
;instead and then lights the 'programmed' LED.
;The same loop runs again until the whole EPROM has been verified.
;The 'verified' LED is lit if there were no errors.

;Notes:
;1. The Bootstrap program changes the address pointer to the internal EPROM by incrementing
;    the 2 address bytes in the 'sta 0F07FH' command.
;2. The internal EPROM address in the 'sta 0F07FH' command has an 'F' in the high nibble
;    of the high byte (F0). This is ignored by the micro but helps by making it easier to
;    skip non-EPROM locations and easier to stop the programming/verify loop.
;3. Any data bytes which are 00H are skipped (ie not programmed).
;4. In the programming and verification loops non-EPROM addresses are skipped ie 0000H to
     007FH inc and 0F3FH to 0FF7H inc.
;5. Just before the program reads the external EPROM it checks the INT input (pin 3). If
;    this pin is high, it skips the EPROM read. In Motorola's suggested Programmer the INT
;    pin is connected to 0V so it always reads the external EPROM.
;6. The command to turn on the 'verified' LED is 'bclr 2,PortB' and is executed
;    at the end of the whole verification procedure. If verification fails at any
;    point the Bootstrap program modifies this command to become 'bset 2,PortB'.
```

```
;    So at the end of the verification procedure when the command executes, the LED is
;    lit if the command is 'bclr' and not lit if it's 'bset'.
;7. The Bootstrap code does NOT check to see if the programming voltage (+21V) is correct
;     before it runs.
;8. The length of the programming pulse is calculated as follows:
;     "clr PCR" at address 0048H applies the programming voltage to the internal EPROM
;     "bsr Delay" takes 8 + 12810 cycles
;     "bra Loop"  takes 4 cycles
;     "ldx #0FEH" takes 2 cycles
;     "stx PCR" (removes the programming voltage from the internal EPROM)  takes 5 cycles
;   So the total delay is 12829 cycles.
;   With a 1M clock the programming pulse length is 12829 X 4 / 1000000 = 51.3mS



;*****************************************************************************
;The Bootstrap ROM starts here (0F80H). The first thing it does is copy itself to RAM.
;Contents of address 0FF4H (ie 0F8CH + 68H) are copied to address 0076H (ie 000EH + 68H), etc.
;It finishes with contents of address 0F8DH copied to address 000FH.
;36H is left in the accumulator after the block move.
;*****************************************************************************

                        org     0F80H
0F80    9C              rsp
0F81    AE68            ldx     #68H
0F83    D60F8C  BM0:    lda     0F8CH,X
0F86    E70E            sta     0EH,X
0F88    5A              decx
0F89    26F8            bne     BM0
0F8B    BC19            jmp     19H         ;Continue running the program in RAM from address 0019H.
0F8D    36              db      36H         ;This byte is left in the accumulator when the jump occurs.
```

0F8E 1601
0F90 2FFE
0F92 B600
0F94 1701
0F96 81
0F97 B701
0F99 A61F
0F9B B705
0F9D 1901
0F9F ADED
0FA1 5C
0FA2 2AFB
0FA4 AEFE
0FA6 BF0B
0FA8 3C45
0FAA 2604
0FAC 3C44
0FAE 271C
0FB0 ADDC
0FB2 BE44
0FB4 5C
0FB5 260A
0FB7 BE45
0FB9 A33F
0FBB 2504
0FBD A3F8
0FBF 25E3
0FC1 C7F07F
0FC4 2704
0FC6 3F0B
0FC8 AD1E

```
0FCA  20D8
0FCC  03010D
0FCF  5F
0FD0  E662
0FD2  E743
0FD4  5C
0FD5  A308
0FD7  26F7
0FD9  5F
0FDA  20BB
0FDC  1501
0FDE  20FE
0FE0  C1F07F
0FE3  262B
0FE5  BC26
0FE7  15
0FE8  5F
0FE9  ADA9
0FEB  ADA7
0FED  5A
0FEE  26F9
0FF0  81
0FF1  115E
0FF3  BC26                                  ;Last instruction of Bootstrap ROM
0FF5  00                                    ;00H is not used
0FF6  0F80              dw      0F80H       ;Vector to the start of Bootstrap ROM. When powered up,
                                            ;the 12V applied to the TIMER input makes the micro
                                            ;fetch this vector.
```

```
                        ;*************************************************************************
                        ;The following code is the Bootstrap ROM after having been copied to RAM.
                        ;Execution continues at address 0019H.
                        ;*************************************************************************

                        ;The following routine reads a byte from the external EPROM & then increments the 4040 counter
0010  1601    GetByte:  bset  3,PortB   ;Bit 3 is connected to the 4040 CLK input
0012  2FFE              bih   ClrCLK    ;Don't read ext EPROM if INT input is HIGH
0014  B600              lda   PortA     ;Read ext EPROM value
0016  1701    ClrCLK:   bclr  3,PortB   ;4040 counter increments on falling edge of CLK
0018  81                rts



                        ;*************************
                        ;Main program starts here
                        ;*************************

0019  B701    START:    sta   PortB     ;36H was left in ACC after block move. Writing this to
                                        ;PortB results in the following:
                                        ;reset is true, CLK is low, LEDs are off & Vpp = 21V

                                        ;After the EPROM is programmed, 3 lines of code are
                                        ;overwritten to change the code into a verify procedure and
                                        ;then the whole program runs again from here (ie START).
                                        ;The value 1BH is left in ACC so when "sta PortB" executes
                                        ;the second time it sets PortB as follows:
                                        ;reset is true, CLK is low, Prog LED is ON, Ver LED is OFF
                                        ;and Vpp = 5V
```

```
001B  A61F                lda   #1FH
001D  B705                sta   DDRB        ;Set PortB bits 4 to 0 as outputs
001F  1901                bclr  4,PortB     ;Remove RESET to 4040 counter


                ;Skip the first 128 bytes since the MC68705U3 EPROM starts at 0080H
0021  ADED      L0:       bsr   GetByte
0023  5C                 incx
0024  2AFB                bpl   L0


                ;This is the programming/verify loop.
                ;After programming finishes, addresses 0043H to 004AH (8 bytes) are overwritten with the values
                ;from addresses 0062H to 0069H. This changes the loop from a programming routine to a verify one.
                ;I've added the replacement code in the comment part of the 4 affected lines.
0026  AEFE      Loop:     ldx   #0FEH
0028  BF0B                stx   PCR         ;Remove program voltage from EPROM
002A  3C45                inc   45H         ;Increment address pointer to internal EPROM. It's the
                                            ;"0F07FH"  part of the "sta  0F07FH" instruction at
                                            ;address 0043H.
002C  2604                bne   L1
002E  3C44                inc   44H
0030  271C                beq   L4
0032  ADDC      L1:       bsr   GetByte
0034  BE44                ldx   44H         ;The following 8 lines skip non-EPROM addresses.
0036  5C                  incx              ;Skipped addresses are FF3FH to FFF7H inc. Note that
                                            ;the micro ignores the upper 4 bits of the address so the
                                            ;addresses actually skipped are 0F3FH to 0FF7H inc.

0037  260A                bne   L2
```

```
0039  BE45                ldx   45H
003B  A33F                cpx   #3FH
003D  2504                bcs   L2
003F  A3F8                cpx   #0F8H
0041  25E3                bcs   Loop
0043  C7F07F  L2:         sta   0F07FH      ;Write data to internal EPROM      cmp  0F07FH
0046  2704                beq   L3          ;Don't program if Byte = 0         bne  Change
0048  3F0B                clr   PCR         ;Apply program voltage to EPROM    jmp  Loop
004A  AD1E                bsr   Delay                                          db   15H
004C  20D8    L3:         bra   Loop

004E  03010D  L4:         brclr 1,PortB,FIN ;Check if this is the first or second time around.
                                            ;If first time (ie 'programmed' LED is off) then continue
                                            ;by changing the code which does the programming to one
                                            ;which does a verify.
                                            ;If second time then all done so go to FIN.


                          ;Change 8 bytes in the programming loop to make it verify instead
                          ;ie overwrite addresses 0043H to 004AH with the values from addresses 0062H to 0069H
0051  5F                  clrx
0052  E662    L5:         lda   62H,X
0054  E743                sta   43H,X
0056  5C                  incx
0057  A308                cpx   #08H
0059  26F7                bne   L5
005B  5F                  clrx
005C  20BB                bra   START       ;Repeat the whole program to perform verification of data
```

```
                        ;Come here at the end of everything. If data verified OK then "bclr" switches on the
                        ;'verified' LED. If verification failed, the code at address 0073H changed "bclr" to
                        ;"bset" so the 'verified' LED stays OFF.
005E  1501      FIN:    bclr    2,PortB     ;Turn on 'ver' LED if op is "bclr" or turn off if "bset".
0060  20FE              bra     *           ;FINISHED so stop here with infinite loop.

                        ;These 3 lines get copied to addresses 0043H - 004AH thereby changing the program
                        ;from a programmer to one that verifies.
0062  C1F07F            cmp     0F07FH
0065  262B              bne     +43
0067  BC26              jmp     0026H
0069  15                db     15H         ;This byte (15H) is used to set PortB differently


                        ;The following delay time is:
                        ;4 + 256 * (21 + 21 + 4 + 4) + 6 = 12810 cycles
                        ;With a 1M clock it's 12810 X 4 / 1000000 = 51.24mS
006A  5F        Delay:  clrx                ;4 cycles
006B  ADA9      del:    bsr     ClrCLK      ;8 + 7 + 6 = 21 cycles
006D  ADA7              bsr     ClrCLK      ;8 + 7 + 6 = 21 cycles
006F  5A                decx                ;4 cycles
0070  26F9              bne     del         ;4 cycles
0072  81                rts                 ;6 cycles


                        ;This changes the 'bclr' command at address 005EH to a 'bset' since the verify failed.
                        ;To change code for 'bclr 2' (15H) to 'bset 2' (14H) just clear bit 0.
0073  115E      Change: bclr    0,FIN
0075  BC26              jmp     Loop
```