

Interfacing the X84041 MPS EEPROM to the Motorola 68HC11 Microcontroller

by Applications Staff

This application note demonstrates how the Xicor X84041 MPS EEPROM can be interfaced to the 68HC11 microcontroller family when connected as shown in Fig. 1. The interface uses the time-multiplexed address/data bus and two control lines of the 68HC11 to interface to the MPS EEPROM. Although the X84041 requires minimal glue logic 3-NAND gates when connected to the 68HC11, the

advantage of the MPS EEPROM as a port-less serial memory device is still preserved. The 68HC11 assembly code listing for this application note can be obtained from the Xicor BBS, FaxBack system, or Xicor's web site at <http://www.xicor.com>

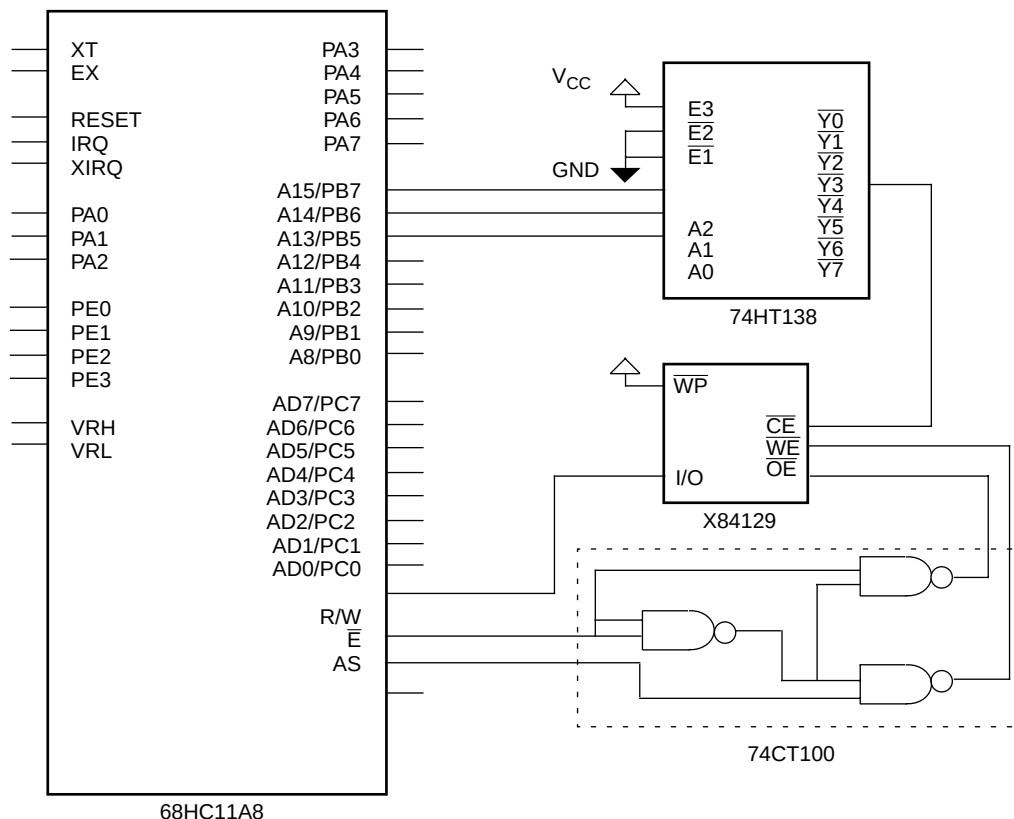


Figure 1. Typical hardware connection for interfacing an X84041 to the 68HC11 microcontroller.

```
*****
**
** DESCRIPTION:
**
** This file contains general utility routines written in 68HC11 assembly
** language used to interface the 68HC11 to the XICOR X84041 MPS EEPROM.
** The interface uses the 68HC11 parallel bus and two control lines to connect
** to the X84041. The microcontroller R/W and E control lines are connected
** through 3 NAND gates to match the X84041's /OE and /WE control lines.
** Address lines A15, A14, A13 are decoded for the chip select; mapping the
** X84041 to address space 6000: 7FFF.
** The following table lists all the subroutines in this file with a brief
** description:
**
**      ResetD: Resets the device before a read or write can take place
**      Page_Write: Writes a page of data to the device
**      Page_Read: Reads a page of data from the device into the uC's RAM
**      Byte_Read: Reads a byte of data from the device into the uC's RAM
**      Byte_Write: Writes a byte of data to the device
**      Inbyte: Called by read subroutines to shift data in
**      Outbyte: Called by write subroutines to shift data out
**      Sndaddr: Called by read/write subroutines to send address to device
**      SNVWrt: Provides start non-volatile write sequence required for all writes
**      CheckNVW: Checks to make sure the non-volatile write is completed
**
** The Main program writes a test string into the MPS EEPROM. After
** page is programmed, the first byte of the page is altered. The page is
** then read back and written to a different location in memory.
** The data read is temporarily stored in the internal RAM.
*****

*****
*
*      INTERNAL RAM
*****
RAMBASE      EQU      $0000      THE INTERNAL RAM BASE ADDRESS(Default)
RAMBuff      EQU      RAMBASE    RAM BUFFER ADDRESS
STACK        EQU      RAMBASE+$FF
*****
*
*      PROGRAM CONSTANTS
*****
Address      EQU      $6000
MPSaddress   EQU      $0000
MPSaddress2  EQU      $0100
Page_Size    EQU      8
*****
*
*      RESET VECTOR ENTRY POINT
*****

ORG      $FFFE      RESET VECTOR ADDRESS TO PROGRAM ENTRY
FDB      $E000      JUMP TO BEGINNING OF EXECUTABLE COD

* ASSEMBLER REQUIREMENT- CPU TYPE

P68H11
```

```

*****
*                                     *
*                               START OF USER CODE                               *
*                                     *
*****

        ORG $E000
MAIN:
        lds      #STACK                * LOAD STACK POINTER

* INITIALIZE THE BUFFER BEFORE PROGRAMMING THE CONTENT TO A SECTOR
        ldx      #TestString           * IX = Test String Address
        ldy      #MPSaddress           * Address within MPS to write data to
        jsr      Page_Write            * Write data to the first page
        ldy      #MPSaddress           * Address within MPS to write byte to
        jsr      Byte_Write            * Write byte to the first address location
        ldy      #MPSaddress           * Address within MPS to read from
        jsr      Page_Read             * Read data in the first page
        ldx      #RAMBuff              * Set IX data pointer to data just read
        ldy      #MPSaddress2          * Address within MPS to write data to
        jsr      Page_Write            * Write data to page 0100 hex
Done: jmp      Done
*****

*** Name: resetd
*** Description: Sends Reset sequence to the device.
*** Function: Performs a read, write "0", read
*** Calls:
*** Input:
*** Output:
*** Register Usage: A
*****
ResetD: ldaa    Address                * sends read command
        clra                                * clear accum
        staa    Address                * send write "0" command
        ldaa    Address                * sends read command
        rts

*****

*** Name: Page_Write
*** Description:
*** Function: Writes a page of data to the first address.
*** Calls: Sndaddr, OutByte
*** Input:
*** Output:
*** Register Usage: x, y
*****
Page_Write:
        jsr      Sndaddr                * Send Page address to device
        ldy      #Page_Size            * Y register contains number of bytes/page
PagePW: ldaa    0,x                    * Load the "test string" in the X register
        pshy
        jsr      OutByte                * Sends out the byte in the accum
        puly
        inx                            * Increments the X register
        dey                            * Decrements the page counter
        bne     PagePW                * Branches until all bytes are written
        jsr      SNVWrt                * Start Nonvolatile Write
        jsr      CheckNVW              * Checks completion of non-volatile write
        rts

```

```

*****
*** Name: Page_Read
*** Description:
*** Function: Reads a page of data from the first address.
*** Calls: Sndaddr, InByte
*** Input:
*** Output:
*** Register Usage: x, y
*****

```

```

Page_Read:
    jsr    Sndaddr        * Send Page address to device
    ldy    #Page_Size    * Y register contains number of bytes/page
    ldx    #RAMBuff       * Sets the index register x to 0
PagePR: pshy
    jsr    InByte         * Receives the byte of data
    puly
    staa   0,x            * Stores the byte to RAM
    inx                      * Increments the X register
    dey                      * Decrements the page counter
    bne    PagePR         * Branches until all bytes are read
    rts

```

```

*****
*** Name: Byte_Read
*** Description:
*** Function: Reads a byte of data from the first address.
*** Calls: Sndaddr, InByte
*** Input:
*** Output:
*** Register Usage: x
*****

```

```

Byte_Read:
    jsr    Sndaddr        * Send Byte address to device
    ldx    #RAMBuff       * Sets the index register x to 0
PageBR: jsr    InByte     * Receives the byte of data
    staa   0,x            * Stores the byte to RAM
    rts

```

```

*****
*** Name: InByte
*** Description: Reads in 8 bits
*** Function:
*** Calls:
*** Input:
*** Output:
*** Register Usage:  y

```

```

*****
InByte: ldy    #$8        * Sets y to 8
        clra          * Clears accum
out2:  ldab    Address    * Load bit from device to accum b
        andb    #00000001b * Mask-out unwanted bits accum b
        rora          * Rotate accum 1 bit to the left
        aba          * Mask accum b into accum a
        dey
        bne    out2      * Branch until accum a contains complete byte
        rts

```

```

*****
*** Name: Byte_Write
*** Description:
*** Function: Writes a byte of data to the first address.
*** Calls: Sndaddr, OutByte
*** Input:
*** Output:
*** Register Usage:
*****

```

```

Byte_Write:
    jsr    Sndaddr          * Send Byte address to device
    ldaa   #$58             * Load accum with "X"
    jsr    OutByte          * Send
    jsr    SNVWrt           * Start Nonvolatile Write
    jsr    CheckNVW         * Checks completion of non-volatile write
    rts

```

```

*****
*** Name: Sndaddr
*** Description: Send address to the device
*** Function: Writes the 16 bit address to the device.
*** Calls: ResetD, Outbyte
*** Input:
*** Output:
*** Register Usage: y
*****

```

```

Sndaddr:
    jsr    ResetD           * Send the reset signal
    xgdy   xgdy             * Load the address in Y to double accum
    jsr    OutByte          * send MSB of address
    tba    tba              * transfer LSB to accum A
    jsr    OutByte          * send LSM of address
    rts

```

```

*****
*** Name: OutByte
*** Description:
*** Function: Sends out 8 bits to Address.
*** Calls:
*** Input:
*** Output:
*** Register Usage: y
*****

```

```

OutByte: ldy    #$8
        rola
out1:   rola
        staa   Address
        dey
        bne    out1
        rts

```

```
*****
*** Name: SNVWrt
*** Description:
*** Function: Sends out 8 bits to Address.
*** Calls:
*** Input:
*** Output:
*** Register Usage:
*****
SNVWrt: ldaa  Address      * sends read command
        ldaa  #$1         * set accum to "1"
        staa  Address     * send write "1" command
        ldaa  Address     * sends read command
        rts

*****
*** Name: CheckNVW
*** Description:
*** Function:
*** Calls:
*** Input:
*** Output:
*** Register Usage:
*****
CheckNVW: ldaa  Address    * sends read command
          rora           * rotate D0 to the carry bit
          bcc   CheckNVW  * loop if nonvolatile write is occuring
          rts

TestString: FCC  'xICORMPS'
*****
*** END OF  X84041  MPS INTERTERFACE SOURCE CODE
*****
END
```