INTRODUCTION

There are many different microcontrollers on the market today that are being used in embedded control applications. Many of these embedded control systems need non-volatile memory. Because of their small footprint, byte level flexibility, low I/O pin requirement, low power consumption and low cost, Serial EEPROMs are a popular choice for non-volatile storage.

Microchip Technology Incorporated addresses this need by offering a full line of Serial EEPROMs covering industry standard serial communication protocols for 2-wire and 3-wire communication. The theory, operation and differences of these two protocols are discussed in detail in Microchip's application note AN536. The reader should refer to AN536 if unfamiliar with 2-wire and/or 3-wire communication protocols. Serial EEPROM devices are available in a variety of densities, operational voltage ranges and packaging options.

Microchip realizes that its customer base is very broad, and because of this, different microcontrollers are used to interface to Serial EEPROMs. One of the microcontrollers used in these applications is the Motorola® 68HC11. In order to simplify the design process, Microchip has written a 68HC11 assembly code to communicate with our 2-wire and 3-wire parts that is verified and tested to function properly.

There are 13 programs written for inclusion in this application note. A listing of one of the programs is included with this application note as an example. The following programs are available for downloading via Microchip's Bulletin Board (BBS). Users may consult the index of the Microchip Embedded Control Handbook for log-on instructions for the BBS. Once logged on to the Microchip BBS, select the FILE LIBRARY (download files) option from the main menu. Next, select a library. This application note is located in the MEM_APPS file library. Once in the MEM_APPS file library, the proper application note can be selected and downloaded by following the directions supplied by the BBS system. Only one file needs to be downloaded from the Microchip BBS to get all of the source code. All source code files are combined and compressed into one downloadable file called AN609.zip.

For Microchip 2-wire devices (24CXXA, 24LCXX, 24LCXXB, 24AAXX devices, excluding 24XX32 and 24XX65 devices), there are three programs to perform some of the basic communication functions.

- SRDMT2W.ASM - 2-Wire Sequential Read
- BW4MT2W.ASM - 2-Wire Sequential Write
- BW4PMT2W.ASM - 2-Wire Byte Write with Data Polling

Refer to the data sheets in the Microchip Data Book for the 2-wire devices listed above for explanations of sequential read, sequential write and data polling.

Microchip also offers a powerful and flexible family of products referred to as Smart Serial™. These devices use the same 2-wire interface as described above, but have added intelligence not found on Microchip's other 2-wire devices. These devices include the 24C32, 24LC32, 24AA32, 24C65, 24LC65, and 24AA65. Among Smart Serial features are: split erase/write cycle endurance (user selectable regions of the device with two different endurance ratings), a 64-byte write cache and the ability to permanently write protect part or all of the array.

- RRDMT65.ASM - Random Read from 64K Smart Serial
- SRDMT65.ASM - Sequential Read from 64K Smart Serial
- CACWMT65.ASM - Full Cache Write to 64K Smart Serial
- WR8MT65.ASM - Sequential Byte Write to 64K Smart Serial
- WR8PMT65.ASM - Sequential Byte Write with Data Polling to 64K Smart Serial
- HEMT65.ASM - Setting of High Endurance block for 64K Smart Serial
- SECMT65.ASM - Setting Security Features for 64K Smart Serial

Refer to individual data sheets of the parts listed for the definition and explanation of the functions.

For Microchip 3-wire devices (93CXX, 93LCXX 93LCXXB, and 93AAXX devices), there are three programs included to perform some of the basic communication functions.

- MT4BR3W.ASM - 3-Wire Multiple Word Read
- MT4BW3W.ASM - 3-Wire Multiple Word Write
- MT4BW3PW.ASM - 3-Wire Multiple Word Write with Data Polling
Refer to the individual data sheets of the 3-wire devices listed for explanations of multiple word read, multiple word write and data polling.

Figure 1 describes the hardware schematic for the interface between Microchip’s 2-wire devices and the Motorola 68HC11E9. This schematic applies to the connection of the Smart Serial devices as well. Figure 2 describes the hardware schematic used to connect Microchip’s 3-wire devices to the Motorola microcontroller. The schematics show the connections necessary between the microcontroller and the serial EEPROM, and the software was written assuming these connections.

**FIGURE 1: CIRCUIT DIAGRAM FOR 68HC11 TO 2-WIRE SERIAL EEPROM INTERFACE**
FIGURE 2: CIRCUIT DIAGRAM FOR 68HC11 TO 3-WIRE SERIAL EEPROM INTERFACE

- **MC68HC11E9**
- **VDD**
- **PA0/IC3**
- **PA1/IC2**
- **PA2/IC0**
- **PA3/OC5/OC1**
- **PA4/OC4/OC1**
- **PA5/OC3/OC1**
- **PA6/OC2/OC1**
- **PA7/PAI/OC1**
- **PB0**
- **PB1**
- **PB2**
- **PB3**
- **PB4**
- **PB5**
- **PB6**
- **PB7**
- **PC0**
- **PC1**
- **PC2**
- **PC3**
- **PC4**
- **PC5**
- **PC6**
- **PC7**
- **PD0/RxD**
- **PD1/TxD**
- **PD2/MISO**
- **PD3/MOSI**
- **PD4/SCK**
- **PD5/SS**
- **PE0/AN0**
- **PE1/AN1**
- **PE2/AN2**
- **PE3/AN3**
- **PE4/AN4**
- **PE5/AN5**
- **PE6/AN6**
- **PE7/AN7**
- **VRH**
- **VRL**
- **EXTAL**
- **XTAL**
- **XIRQ**
- **IRQ**
- **MODA/LIR**
- **MODB/VSTBY**
- **RESET**
- **STRB**
- **STRA**
- **4.7k**
- **18pF**
APPENDIX A: SOURCE CODE

; *******************************************************************************
; 2-Wire Sequential Write Program (225 bytes)
;
; This program (bwr4mt2w.*) writes 4 bytes of $A5 to a Microchip 24LCxx
; beginning at location $10 using a Motorola 68HC11 microcontroller.
; This code has been verified that it writes properly to a Microchip
; 24LCxx serial EEPROM.
;
; This program was written using a 68CH11EVBU evaluation board.
; This board has a monitor program in firmware of the 68HC11 which
; allows single stepping, register viewing, modifying, etc. Since this
; is the case, the program code will be loaded into the on-chip EEPROM.
; This EEPROM begins at $B600. The control registers are left to their
; default location of $1000 and the RAM is left to its default location.
; RAM locations of $48-$ff are used by the monitor program and are not
; available for program use. Therefore, the stack pointer is set a $47
; and will be able to use all of the RAM to $00, and the RAM variables
; begin at location $100 and go up from there. I cannot program the
; reset vector since it is ROM space (at $FFFE), so the way I run this
; program is to use the monitor program that comes with the evaluation
; board to set the program counter to the starting address of my user
; program ($B600) and begin from there. For users who do have access
; to the reset vector, the label of the beginning program (in my case, it
; is called START) should be loaded at location $FFFE. This program was
; not assembled using a Motorola assembler, but was assembled using
; Universal Cross-Assemblers Cross-32 Meta-Assembler. It has the
; ability to assemble just about any microcontroller code. There are
; certain commands that are unique to the cross-assembler. These
; commands will be commented differently than other comments to
; be recognizable. They will look like this:

;++++++++++++++++++++++++++++++++
; Special cross-assembler command(s)
;++++++++++++++++++++++++++++++++

; I do not know the exact assembler commands required to accomplish
; assembly using a Motorola assembler.
;
; The crystal that comes with the evaluation board is 8 Mhz.
; With this used as the clock input, this code will output a serial
; data stream at approximately 32khz. Although this does not
; meet I2C spec of 100 khz, it communicate with the Microchip
; 24LCxx parts properly. If a different frequency crystal is used,
; the code may need to be modified to meet timing specifications.

;*******************************************************************************

;+++++++++++++++++++++++++++++++++++++++++++++++++++++++0000 CPU "C:\WINC32\68HC11.TBL" ; LOAD TABLE 0000 HOF "MOT8" ; Hex output is Motorola S-records 0000 PAGE 60 ; Sets # of lines in list file ; to 60 before pagebreak;+++++++++++++++++++++++++++++++++++++++++++++++++++++++ ; 68HC11 control register locations;******************************************************************************* 1000 = REGBS EQU 1000H ; BEGINNING OF REGISTERS
0100 = RAMBS EQU 100H ; BEGINNING OF RAM VARIABLES
1007 = DDRC EQU REGBS+07H ; DATA DIRECTION REG FOR PORT C
1003 = PORTC EQU REGBS+03H ; PORT C DATA REGISTER
0003 = PCOFF EQU 03H ; OFFSET FROM CONTROL REG BEG.

;*****************************************************************
;*****************************************************************
; User defined constants
;*****************************************************************
0003 = CHIDHI EQU 00000011B ; SET BOTH CLK AND DATA HI
0002 = CHIDLO EQU 00000010B ; SET CLK HI AND DATA LO
0001 = CLODHI EQU 00000001B ; SET CLK LO AND DATA HI
0000 = CLODLO EQU 00000000B ; SET CLK AND DATA BOTH LO
0001 = DIMASK EQU 00000001B ; BIT MASK FOR DATA IN BIT
0080 = DOMASK EQU 10000000B ; BIT MASK FOR DATA OUT BIT
0001 = SDAMASK EQU 00000001B ; BIT MASK FOR SERIAL DATA
0002 = SCKMASK EQU 00000010B ; BIT MASK FOR SERIAL CLOCK
0004 = LEDMASK EQU 00000100B ; BIT MASK FOR ACK FAILED LED

;*****************************************************************

0000 8E0047 LDS #0047H ; STACK POINTER BEGINS AT $47
0100 ORG 100H ; RAM variables begin at 100h

;***************************************************************
; Program code cannot be placed in ROM for eval board because
; eval board firmware is loaded in ROM. Program code will be
; loaded in EEPROM (which is 512 bytes and begins at B600h).

; Serial data (SDA) is located on port c, pin 0
; Serial clock (SCK) is located on port c, pin 1
; An acknowledge fail LED is connected to port c, pin 2. When
; the ack is not low like is should be, the ack failed LED is
; illuminated.

B600 ORG 0B600H

;***************************************************************
; This is the main portion of the code. It is where the reset
; vector should set program counter to.

B600 START
B600 8610 LDAA #00010000B ; ADDRESS OF MEMORY TO BEGIN
B602 B70102 STAA ADDR ; WRITING DATA
B605 B6D4 LDS #4 ; NUMBER OF BYTES TO WRITE OUT
B607 B70104 STAA BYTECNT ;
B60A CE1000 LDX #REGBS ; LOAD $1000 INTO X INDEX REG

B60D BDB639 JSR STRTBIT ; GOTO STRTBIT SUBROUTINE
B610 B6A0 LDS #10100000B ; LOAD CONTROL BYTE INTO
B612 B70100 STAA TXBUFF ; TXBUFF FOR OUTPUT TO SEEPROM
B615 BDB6A2 JSR TXBYTE ; OUTPUT 1 BYTE TO SEEPROM
B618 B60102 LDAA ADDR ; GET ADDRESS AND LOAD IN
B61B B70100 STAA TXBUFF ; TXBUFF FOR OUTPUT
B61E BDB6A2 JSR TXBYTE ; OUTPUT 1 BYTE TO SEEPROM
B621 F60104 LDAB BYTECNT
B624 86A5 NEXTWRLDAA #10100101B ; DATA BYTE TO OUTPUT IS A5H
B626 B70100 STAA TXBUFF ;
B629 37 PSHB ; PUSH DATA BYTE COUNTER TO STACK
B62A BDB6A2 JSR TXBYTE ; OUTPUT 1 BYTE TO SEEPROM
B62D 33 PULB ; PULL DATA BYTE COUNTER FROM STACK
B62E 5A DECB ; HAVE WE OUTPUT CORRECT # OF DATA BYTES?
B62F C100 CMPB #00H ;
B631 26F1 BNE NEXTWR ; NO, THEN SEND NEXT BIT
B633 BDB653 JSR STOPBIT ; SEND STOP BIT TO BEGIN INTERNAL WRITE CYCLE
B636 7EB600 JMP START ; START OVER AGAIN

;******************************************************************************
;******************************************************************************
;     Start bit output subroutine
;******************************************************************************
B639 STRTBIT
B639 8607 LDAA #00000111B ;
B63B B71007 STAA DDRC ; PORT C ALL INPUTS EXCEPT BITS 0,1,2
B63E 8603 LDAA #CHIDHI ; SET SCLK AND SDATA HI
B640 B71003 STAA PORTC ;
B643 01 NOP ; OBEY PROPER START BIT SETUP ; TIME
B644 01 NOP ;
B645 01 NOP ;
B646 01 NOP ;
B647 01 NOP ;
B648 1D0301 BCLR PCOFF,X,SDAMASK ; SET DATA LOW FOR STOP BIT
B64B 01 NOP ; OBEY PROPER START BIT HOLD ; TIME
B64C 01 NOP ;
B64D 01 NOP ;
B64E 01 NOP ;
B64F 1D0302 BCLR PCOFF,X,SCMKAS ; SET CLK LO
B652 39 RTS ; END START BIT SUBROUTINE

;******************************************************************************
;******************************************************************************
;     Stop bit output subroutine
;******************************************************************************
B653 STOPBIT
B653 8607 LDAA #00000111B ;
B655 B71007 STAA DDRC ; PORT C ALL INPUTS EXCEPT FOR BITS 0,1
B658 1D0301 BCLR PCOFF,X,SDAMASK ; MAKE SURE DATA BIT IS LOW
B65B 1C0302 BSET PCOFF,X,SCMKAS ; CLK BIT HI
B65E 01  NOP ; OBEY PROPER STOP BIT SETUP
B65F 01  NOP ;
B660 01  NOP ;
B661 01  NOP ;
B662 1C0301 BSET PCOFF,X,SDAMASK ; DATA BIT HI CAUSES STOP BIT
B663 39  RTS ; END STOP BIT SUBROUTINE

;**********************************************************************************************
; Remember to wait internal write cycle time or acknowledge pole here until writing is complete before beginning next write to part.
;**********************************************************************************************

;**********************************************************************************************
; This routine reads in one bit from the data line
;**********************************************************************************************

B666 8606 LDAA #00000110B ; SET SDATA AS INPUT AND KEEP
B668 B71007 STAA DDRC ; SCLK AS OUTPUT
B66B 7F0101 CLR EE_IN ; GUESS INPUT IS A 0
B66E 1C0301 BSET PCOFF,X,SCKMASK ; SET CLK BIT HI
B671 01  NOP ; WAIT TO READ INPUT
B672 B61003 LDAA PORTC ; GET INPUT FROM SDATA
B675 1D0302 BCLR PCOFF,X,SCKMASK ; BRING CLK LO AFTER PORT READ
B678 8501 BITA #DIMASK ; SEE IF INPUT IS 1 OR 0
B67A 2705 BEQ DONEIN ; INPUT IS A ZERO
B67C 86FF LDAA #0FFH ; INPUT BIT IS ACTUALLY A 1
B67E B70101 STAA EE_IN ; STORE BACK IN EE_IN
B681 DONEIN
B681 39 RTS

;**********************************************************************************************

;**********************************************************************************************
; This routine writes out one bit to the sdata line
;**********************************************************************************************

B682 8607 LDAA #00000111B ; BOTH CLK AND DATA ARE OUTPUTS
B684 B71007 STAA DDRC ; WHAT ARE WE TRYING TO OUTPUT,
B687 B60100 LDAA TXBUFF ; A 1 OR 0?
B68A 8580 BITA #DOMASK ; EE_OUT BIT IS 0
B68C 2705 BEQ LOWOUT
B68E 1C0301 BSET PCOFF,X,SDAMASK ; HIGH NEEDS TO BE SENT OUT
B691 2003 BRA CONTOUT
B693 1D030 BCLR PCOFF,X,SDAMASK ; SEND OUT A LOW
B695 1C0302 CONTOUT BSET PCOFF,X,SCKMASK ; SET CLOCK BIT
B699 01  NOP ; WAIT PROPER SCLK HI TIME
B69A 01   NOP
B69B 1D0302  BCLR PCOFF,X,SCKMASK ; CLR CLOCK BIT AND THEN
B69E 1D0301  BCLR PCOFF,X,SDAMASK ; SET DATA BIT TO 0 FOR NEXT TX
B6A1 39    RTS

;*******************************************************************************************

;*******************************************************************************************

; This routine outputs 1 byte of data out the sdata pin
;*******************************************************************************************

B6A2  TXBYTE
B6A2  C608   LDAB #8 ; SET BIT COUNTER
B6A4  BDB682  TXBIT JSR OUTBIT ; SEND 1 BIT
B6A7  790100  ROL TXBUFF ; GET NEXT BIT READY TO XMIT
B6AA  5A     DECB
B6AB  C100   CMPB #00H ; HAVE WE OUTPUT ALL 8 BITS
B6AD  26F5   BNE TXBIT ; NO, THEN SEND NEXT BIT

; GET ACK BIT AND TEST IF IT IS LOW. IF NOT, TURN ON ACK FAIL LED
B6AF  BDB666  JSR INBIT ; RECEIVE ACK BIT
B6B2  B60101  LDAA EE_IN ;
B6B5  8501    BITA #DIMASK ; TEST IF INPUT IS 0 OR 1
B6B7  2703    BEQ DONETX ; IF ACK IS LOW, GO TO END OF TXBYTE
B6B9  1C0304  BSET PCOFF,X,LEDMASK ; TURN ON ACK FAILED LED
B6BC  39    DONETX RTS

;*******************************************************************************************

;*******************************************************************************************

; This subroutine receives 1 byte from SEEPROM
;*******************************************************************************************

B6BD  RXBYTE
B6BD  C608   LDAB #8 ; LOAD BIT COUNTER IN ACCB
B6BF  790103  RXBIT ROL RXBUFF ; GET READY TO RECEIVE NEXT BIT
B6C2  B60103  LDAA RXBUFF ; GUESS THAT INPUT BIT IS 0
B6C5  84FE   ANDA #11111110B ;
B6C7  B70103  STAA RXBUFF ; WRITE ACCA BACK TO RXBUFF
B6CA  BDB666  JSR INBIT ; RECEIVES 1 BIT
B6CD  B60101  LDAA EE_IN ; IS IN BIT A 1 OR A 0?
B6D0  8501    BITA #DIMASK ;
B6D2  2708    BEQ CONTRX ;
B6D4  B60103  LDAA RXBUFF ; GET RXBUFF INPUT
B6D7  8A01    ORAA #00000001B ; SET INPUT BIT TO 1
B6D9  B70103  STAA RXBUFF ;
B6DC  5A     CONTX DECB
B6DD  C100   CMPB #00H ; HAVE WE OUTPUT ALL 8 BITS
B6DF  26DE   BNE RXBIT ; NO, THEN SEND NEXT BIT
B6E1  39    RTS

;*******************************************************************************************

;*******************************************************************************************

; This command tells the cross-assembler that code starts
; at the location of START

B600  END START
### AMERICAS
- **Corporate Office**
  Microchip Technology Inc.
  2355 West Chandler Blvd.
  Chandler, AZ 85224-6199
  Tel: 602 786-7200  Fax: 602 786-7277
  Technical Support: 602 786-7627
  Web: http://www.mchip.com/microchip

- **Atlanta**
  Microchip Technology Inc.
  500 Sugar Mill Road, Suite 200B
  Atlanta, GA 30350
  Tel: 770 640-0034  Fax: 770 640-0307

- **Boston**
  Microchip Technology Inc.
  5 Mount Royal Avenue
  Marlborough, MA 01752
  Tel: 508 480-9990  Fax: 508 480-8575

- **Chicago**
  Microchip Technology Inc.
  333 Pierce Road, Suite 180
  Itasca, IL 60143
  Tel: 708 285-0071  Fax: 708 285-0075

- **Dallas**
  Microchip Technology Inc.
  14651 Dallas Parkway, Suite 816
  Dallas, TX 75240-8809
  Tel: 214 991-7177  Fax: 214 991-8588

- **Dayton**
  Microchip Technology Inc.
  35 Rockridge Road
  Englewood, OH 45322
  Tel: 513 832-2543  Fax: 513 832-2841

- **Los Angeles**
  Microchip Technology Inc.
  18201 Von Karman, Suite 455
  Irvine, CA 92715
  Tel: 714 263-1888  Fax: 714 263-1338

- **New York**
  Microchip Technology Inc.
  150 Motor Parkway, Suite 416
  Hauppauge, NY 11788
  Tel: 516 273-5305  Fax: 516 273-5335

- **San Jose**
  Microchip Technology Inc.
  2107 North First Street, Suite 590
  San Jose, CA 95131
  Tel: 408 436-7950  Fax: 408 436-7955

### ASIA/PACIFIC
- **Hong Kong**
  Microchip Technology
  Unit No. 3002-3004, Tower 1
  Metropiazza
  223 Hing Fong Road
  Kwai Fong, N.T. Hong Kong
  Tel: 852 2 401 1200  Fax: 852 2 401 3431

- **Korea**
  Microchip Technology
  168-1, Youngbo Bldg. 3 Floor
  Samsung-Dong, Kangnam-Ku,
  Seoul, Korea
  Tel: 82 2 554 7200  Fax: 82 2 558 5934

- **Singapore**
  Microchip Technology
  200 Middle Road
  #10-03 Prime Centre
  Singapore 188980
  Tel: 65 334 8870  Fax: 65 334 8850

- **Taiwan**
  Microchip Technology
  10F-1C 207 Tung Hua North Road
  Taipei, Taiwan, ROC
  Tel: 886 2 717 7175  Fax: 886 2 545 0139

### EUROPE
- **United Kingdom**
  Arizona Microchip Technology Ltd.
  Unit 6, The Courtyard
  Meadow Bank, Furlong Road
  Bourne End, Buckinghamshire SL8 5AJ
  Tel: 44 0 1628 851077 Fax: 44 0 1628 850259

- **France**
  Arizona Microchip Technology SARL
  2 Rue du Buisson aux Fraises
  91300 Massy - France
  Tel: 33 1 69 53 63 20  Fax: 33 1 69 30 90 79

- **Germany**
  Arizona Microchip Technology GmbH
  Gustav-Heinemann-Ring 125
  D-81739 Muenchen, Germany
  Tel: 49 89 627 144 0   Fax: 49 89 627 144 44

- **Italy**
  Arizona Microchip Technology SRL
  Centro Direzionale Colleoni
  Palazzo Pegaso Ingresso No. 2
  Via Paracelso 23, 20041 Agrate Brianza (MI) Italy
  Tel: 39 039 689 9939 Fax: 39 039 689 9883

### JAPAN
- **Microchip Technology Intl. Inc.**
  Benex S-1 6F
  3-18-20, Shin Yokohama
  Kohoku-Ku, Yokohama
  Kanagawa 222 Japan
  Tel: 81 45 471 6166  Fax: 81 45 471 6122

---

All rights reserved. © 1995, Microchip Technology Incorporated, USA.