

Interfacing the X24C44/45 NOVRAMs to the Motorola 6805 Microcontroller

by Applications Staff, July 1992

The following code demonstrates how the Xicor X24C44/45 serial NOVRAMs could be interfaced to the 6805 microcontroller family when connected as shown in Figure 1. The interface uses port A, with the PA3 pin connected to the serial clock (SK), PA2 connected to chip enable (CE), and PA4 connected to both serial

data input (SI) and serial data output (SO) of the NOVRAM. Additional code can be found on the Xicor web site at <http://www.xicor.com> that will implement interfaces between Motorola microcontrollers and other Xicor serial devices.

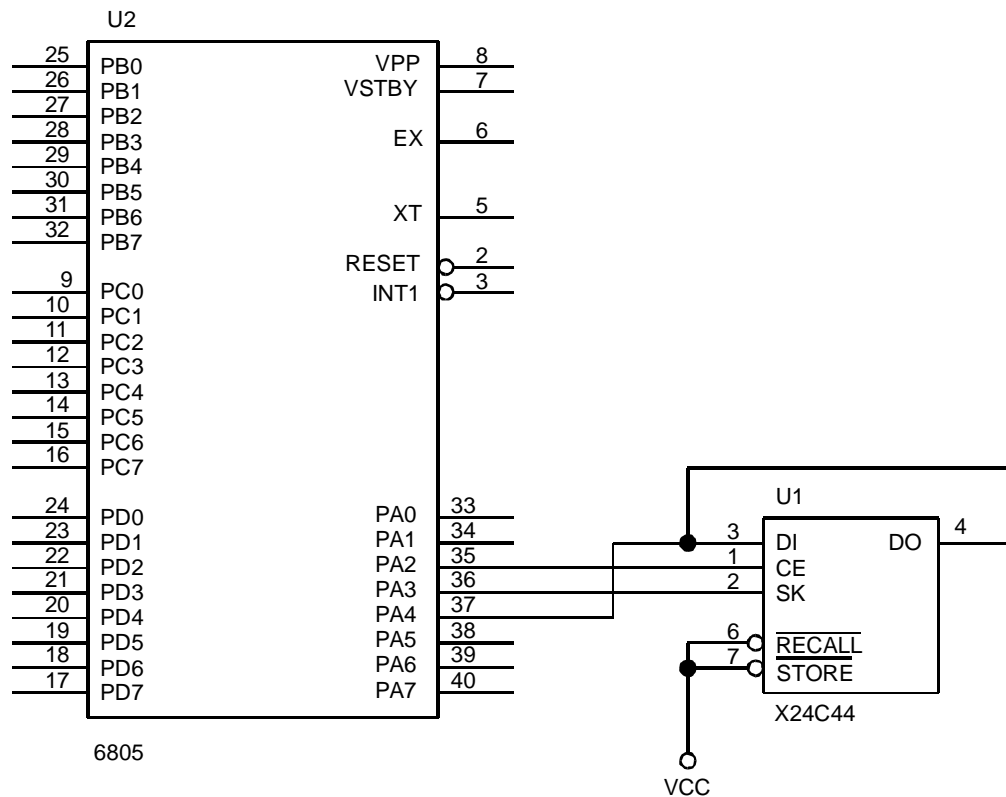


Figure 1. Typical hardware connection for interfacing an X24C44 to a 6805 microcontroller

```

*****
* THIS CODE WAS DESIGNED TO DEMONSTRATE HOW THE X24C44 COULD BE INTERFACED TO *
* THE 68HC05 MICROCONTROLLER. THE INTERFACE USES 3 LINES FROM PORT A (PA2, *
* PA3, AND PA4) TO COMMUNICATE. THE DI AND DO PINS ON THE X24C44 ARE TIED *
* TOGETHER WHICH ALLOWS 1 LESS PORT LINE TO BE USED. *
*
* THE CODE SHOWN DEMONSTRATES RCL, WREN, READ, WRITE, AND STORE *
* INSTRUCTIONS. THE REMAINING INSTRUCTIONS (WRDS AND ENAS) CAN BE ISSUED *
* USING THE SAME ROUTINE AS OTHER NON-DATA INSTRUCTIONS. *
*
* THE PROGRAM ISSUES A SEQUENCE OF INSTRUCTIONS TO READ THE CONTENTS OF *
* ADDRESS 5 AND STORES THE SAME VALUE IN ADDRESS 9. THE SEQUENCE OF *
* INSTRUCTIONS IS AS FOLLOWS : *
*
* 1. RCL          SETS THE PREVIOUS RECALL LATCH *
* 2. WREN        SETS THE WRITE ENABLE LATCH *
* 3. READ        DATA FROM ADDRESS 5 IS READ *
* 4. WRITE       THE DATA READ DURING STEP 3 IS WRITTEN TO ADDRESS 9 *
* 5. STO         THE RAM'S CONTENTS IS TRANSFERED TO THE EEPROM *
*
* DATA TRANSFER IS PERFORMED WITH THE MOST SIGNIFICANT BIT FIRST. *
*****
SKBIT      EQU    3          MASK INDICATING PORTD SK POSITION
CEBIT      EQU    2          MASK INDICATING PORTD CE POSITION
DIOBIT     EQU    4          MASK INDICATING PORTD DATA POSITION
DOUT       EQU    $1C       MASK TO MAKE DI/O AN OUTPUT
DIN        EQU    $0C       MASK TO MAKE DI/O AN INPUT
DMASK      EQU    $10       MASK TO LOOK FOR DATA FROM X24C44
WRDS       EQU    $80       RESET WRITE ENABLE LATCH
STO        EQU    $81       TRANSFERS FROM RAM TO EEPROM
SLEEP     EQU    $82       PLACES PART INTO POWER DOWN MODE
WRITE      EQU    $83       RAM WRITE
WREN       EQU    $84       SET WRITE ENABLE LATCH
RCL        EQU    $85       TRANSFERS FROM EEPROM TO RAM, RESETS
*                               WRITE ENABLE LATCH
READ       EQU    $86       RAM READ
DDRA       EQU    $04       DATA DIRECTION REGISTER FOR PORT A
PORTA      EQU    $00       ADDRESS FOR PORT A
ADDR       EQU    $80       LOCATION FOR X24C44 ADDRESS TO ACCESS
INST       EQU    $81       INSTRUCTION FOR PART
RWDAT      EQU    $82       LOCATION FOR X24C44 DATA TRANSFERED
COUNT     EQU    $84       COUNTER VARIABLE
TEMP1      EQU    $85

```

```

*****
* RESET VECTOR TO BEGINNING OF PROGRAM CODE *
*****

```

```

          ORG    $1FFE          RESET VECTOR TO PROGRAM ENTRY POINT
          FDB    $0100

```

```

*****
* START OF PROGRAM EXECUTION *
*****

```

```

          ORG    $0100          BEGINNING OF EXECUTABLE CODE

```

```

BEGIN:      LDA    #DOUT
            STA    DDRA          MAKE CE, SK, DI/O OUTPUTS
            LDA    #$00
            STA    PORTA        INITIALIZE CE, SK, DI/O TO ZEROS
            LDA    #RCL         PERFORM A RECALL TO SET
            STA    INST          THE RECALL LATCH
            JSR    CEHIGH
            JSR    OUTBYT
            JSR    CELOW
            LDA    #WREN        PERFORM A WRITE ENABLE TO SET
            STA    INST          THE WRITE ENABLE LATCH
            JSR    CEHIGH
            JSR    OUTBYT
            JSR    CELOW
            LDA    #$05         READ THE CONTENTS OF ADDRESS 5
            STA    ADDR         THE VALUE READ WILL BE IN STORED
            JSR    RDWRD        IN RWDATA
            LDA    #$09         WRITE THE DATA JUST READ INTO
            STA    ADDR         ADDRESS 9
            JSR    WRWRD
            LDA    #STO         PERFORM A STORE OPERATION
            STA    INST
            JSR    CEHIGH
            JSR    OUTBYT
            JSR    CELOW
            BRA    *           LOOP UNTIL RESET

*****
* WRITE THE WORD SPECIFIED IN RWDAT. THE ADDRESS TO *
* BE WRITTEN IS SPECIFIED IN ADDR.                 *
*****
WRWRD:     JSR    CEHIGH        WRITE VALUE IN RWDATA INTO LOCATION
            LDA    ADDR         SPECIFIED IN ADDR
            LSLA
            LSLA
            LSLA
            ORA    #WRITE       MASK IN WRITE INSTRUCTION
            STA    INST
            JSR    OUTBYT       SEND WRITE INSTRUCTION TO DUT
            LDA    RWDAT
            STA    INST
            JSR    OUTBYT       SEND IN UPPER BYTE OF DATA
            LDA    RWDAT+1
            STA    INST
            JSR    OUTBYT       SEND IN LOWER BYTE OF DATA
            JSR    CELOW
            RTS

*****
* READ THE WORD AT THE LOCATION SPECIFIED IN ADDR. THE *
* DATA READ WILL BE PLACED IN RWDAT.                *
*****
RDWRD:     JSR    CEHIGH        READ THE ADDRESS SPECIFIED IN ADDR
            LDA    ADDR
            LSLA
            LSLA
            LSLA

```

```

ORA    #READ      MASK IN READ INSTRUCTION
STA    INST
JSR    SEND7      SEND IN 7 BITS OF READ INSTRUCTION
LDA    #DIN       MAKE DATA LINE AN INPUT
STA    DDRA
JSR    CLOCK      SEND EIGHTH CLOCK PULSE FOR READ INSTRUCTION
LDA    #$10       PREPARE TO SHIFT IN 16 BITS
STA    COUNT
BITX:  CLC         ASSUME BIT IS GOING TO BE A ZERO (CLEAR CARRY)
LDA    PORTA      READ BIT VALUE
AND    #DMASK     MASK BIT OUT OF BYTE READ
BEQ    NO1        LEAVE CARRY FLAG ALONE IF BIT IS A 0
SEC                     SET CARRY IF BIT IS A 1
NO1:   ROL    RWDAT+1  ROLL CARRY FLAG INTO DATA WORD
ROL    RWDAT
JSR    CLOCK      SEND A CLOCK PULSE
DEC    COUNT      LOOP UNTIL 16 BITS ARE READ
BNE    BITX
LDA    #DOUT      MAKE DATA LINE AN OUTPUT
STA    DDRA
JSR    CELOW      BRING CE LOW
RTS

```

```

*****
* SEND DATA OUT TO THE PART. THE DATA TO BE SENT IS *
* LOCATED IN INST. *
*****

```

```

SEND7:  LDA    #$07      SHIFT OUT 7 BITS FOR READ INSTRUCTION
        STA    COUNT
        BRA    LOOPO
OUTBYT: LDA    #$08      PREPARE TO SHIFT OUT 8 BITS
        STA    COUNT
LOOPO:  ROL    INST
        BCC    IS0      JUMP IF DATA SHOULD BE 0
        BSET   #DIOBIT,PORTA  SEND 1 TO DI/O
        BRA    IS1
IS0:    BCLR   #DIOBIT,PORTA  SEND 0 TO DI/O
IS1:    JSR    CLOCK      SEND CLOCK SIGNAL
        DEC    COUNT
        BNE    LOOPO     LOOP UNTIL ALL 8 BITS HAVE BEEN SENT
        RTS

```

```

*****
* BRING CE HIGH *
*****

```

```

CEHIGH:  BSET   #CEBIT,PORTA  BRING CE HIGH
        RTS

```

```

*****
* BRING CE LOW *
*****

```

```

CELOW:  BCLR   #DIOBIT,PORTA  BRING DATA LINE LOW
        BCLR   #CEBIT,PORTA   BRING CE LOW
        RTS

```

```
*****  
* ISSUE A CLOCK PULSE. *  
*****
```

```
CLOCK:      BSET  #SKBIT,PORTA      BRING SK HIGH  
            BCLR  #SKBIT,PORTA      BRING SK LOW  
            RTS
```



LittleDiode supplies new, hard to find or obsolete electronic components and semiconductors all over the world.

With over two million different components listed you are sure to find the part you need.

Feel free to visit us today at our online store:

LittleDiode.com

Looking forward to providing you with the best possible service.