Logic Powered Serial EEPROMs

Embedded applications increasingly want more integration and power, in less space for less cost. Using low power Serial EEPROMs (SEE) for application firmware, lookup tables, and microcode coupled with small footprints makes for permanent storage at respectable savings. One additional method of saving on the power budget is selectively powering off components when not needed, a basic for embedded power management. The low-power SEEs offered by Microchip Technology Inc., offer an additional benefit, powering the SEE from a microcontroller port. This allows the host controller to not only manipulate the Serial EEPROM Reads and Write, but also the periods when it is powered off or on. Satellite communications use this technique to save power and total dose accumulation. We call this technique POWER PORT™. The microcontroller port must have sufficient Ioh (source current) to sustain the voltage and current for all memory functions, READ, ERASE, and WRITE. Obviously, not all memory or peripheral devices could be powered thusly, but Microchip's SEE devices will function in this environment.

The microcontroller, using its internal software and hardware decision functions, determines when it needs to communicate with the memory device, then acts accordingly. Any standard wake-up sequence will accomplish this task. The wake-up code needs only power up the memory and wait for the power to become stable before doing a read or write by driving the POWER PORT high. Then all serial communication executes normally. The SEEs are powered off for additional power savings and the data or code is utilized from RAM. Obviously, the port output must be allowed to settle, but normal operation of the output structures would guarantee that this would be met. The I/O port Tpd for the Microchip PIC16C5X, is specified at 40ns maximum.

The 24LCXX and 93LCXX CMOS SEE series parts from Microchip were designed to achieve low current consumption across all ranges of operation.

The four primary ICC parameters for these products are:

Parameter	Conditions
ICC STANDBY	Not in an active operation while VCC is supplied.
ICC READ	The part is in a READ operation.
ICC PEAK WRITE	The BYTE / PAGE WRITE and ERASE operations have self timed cycles of 10 ms. A typical of 4 ms is the actual time of the operation. This is the amount of time when the ICC requires the most current (PEAK WRITE). The part is drawing STANDBY ICC during the remaining 6ms of the cycle.
ICC AVG WRITE	The avg of the PEAK WRITE ICC and STANDBY ICC during the self-timed 10ms write cycle.

The attached characteristic curves (Figures A and B) indicate that ICC PEAK WRITE current consumes the most current. The worst case condition is at 6.0V and –40°C. The 24LCXX series parts draw a typical 3.2 mA and the 93LCXX series parts draw a typical of 2.0 mA. These low ICC characteristics offer a unique current saving benefit for battery applications. Figure C and D illustrate the sink and source current capabilities of the PIC16C5X family of microcontrollers. It is clear from these characterization curves that the microcontroller can deliver sufficient current across all temperature ranges to power a SEE using the POWER PORT technique.

Figure E shows the connection scheme for the Microchip PIC16C54. It should be noted that not all versions of competitive microcontrollers are capable of powering a device in this manner and the specific data sheets for the microcontroller being considered must be consulted for maximum source current. The microcontroller port must be capable of sourcing sufficient current for the duration of the write cycle or 10ms, worse case. The peak write requirement for the 24LCXX product family is 3.2 mA at 5.5 Vdc (–40°C).

Listing A demonstrates the appropriate code sequences when using the PIC16C54 microcontroller. The sequences included are power control, start bit, stop bit, send and receive bit, Tx and Rx, and a general addressing routine.

7

FIGURE A - TYPICAL ICC FOR 24LCXX

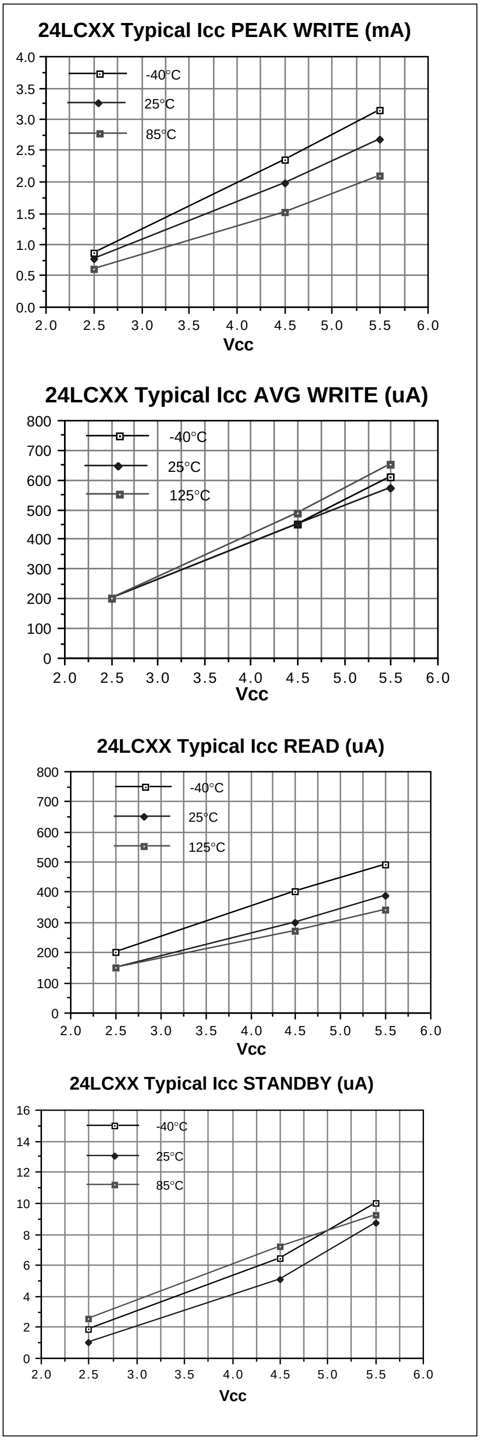


FIGURE B - TYPICAL ICC FOR 93CXX

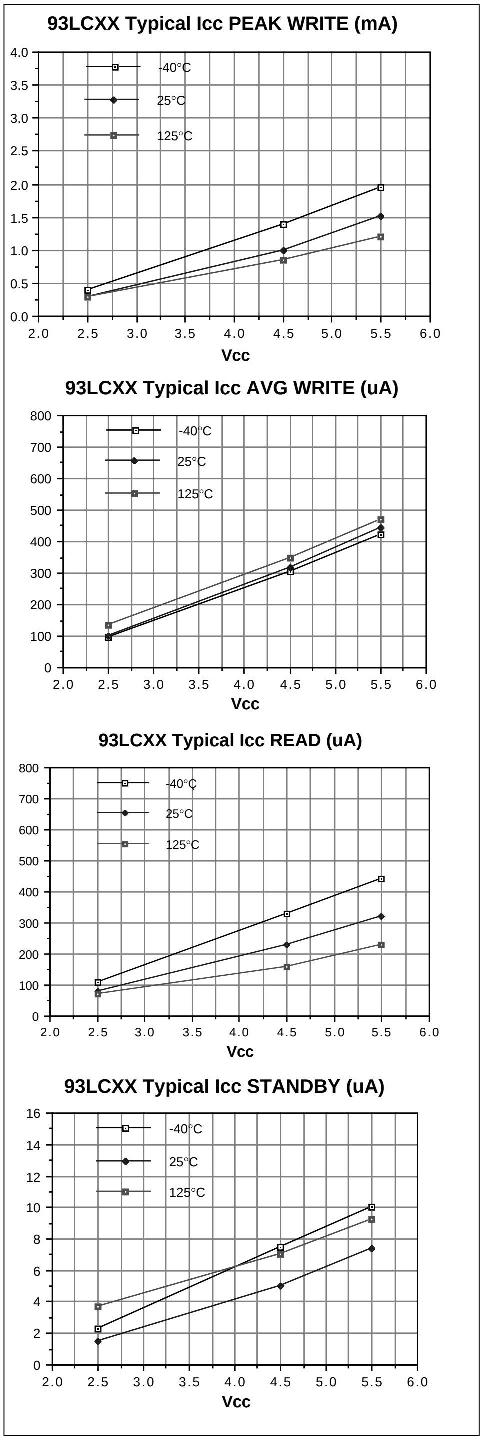


FIGURE C - PIC16C5X IOL AT 5V

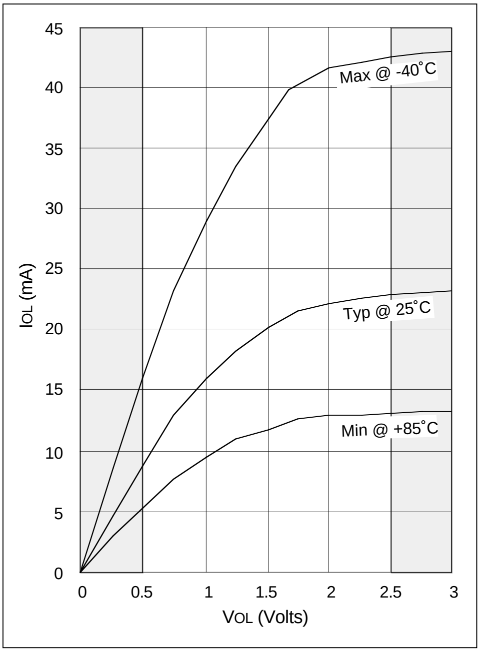


FIGURE D - PIC16C5X IOH AT 5V

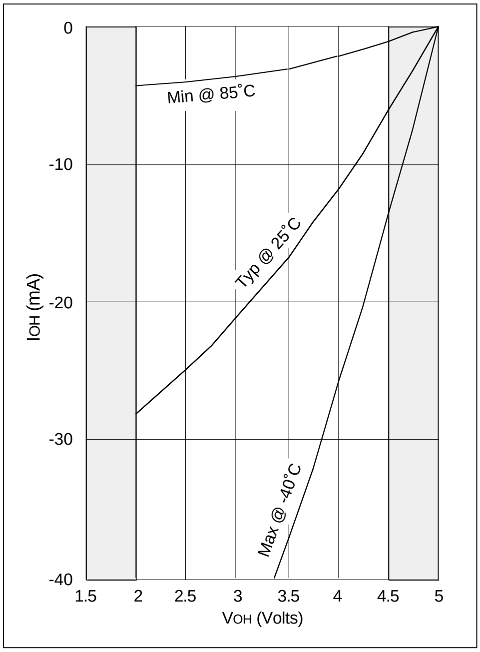
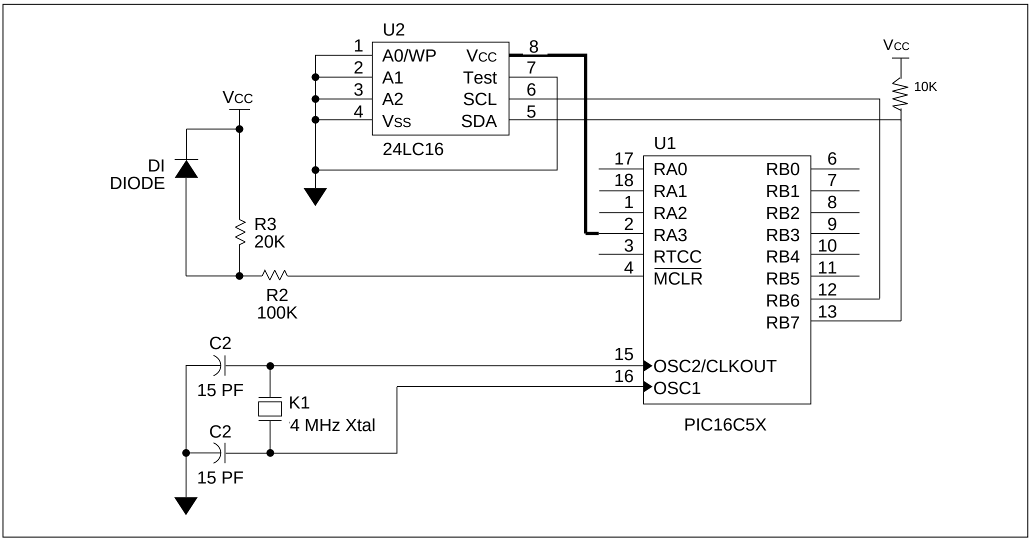


FIGURE E - 24LC16/PIC16C5X INTERFACE SCHEMATIC



7

The primary benefits of this application are:

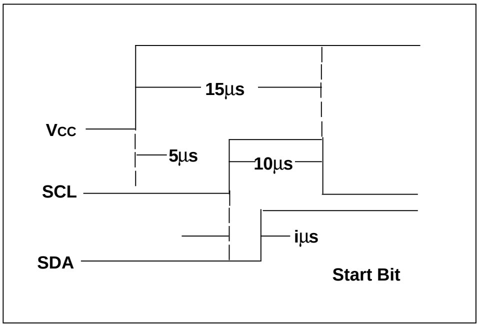
- The SEE is completely powered down to save power when the SEE is not executing an operation. This will directly effect the total system power consumption. This means that the SEE is in a total quiescent state and even the standby current savings are realized, greatly increasing usable battery life, and consequently allowing for a more sophisticated design on the same power budget.
- The very fast 5 µs power-up time minimizes power-up delay.
- Since the serial operation is gated by a stable microcontroller V_{OH}, risk of data being corrupted by a glitch is minimized. This, in effect, is a regulated V_{CC} supply and provides a reliable power source to ensure data integrity.

Several cautions need to be noted:

1. *Gang powering multiple devices must not exceed the I/O port I_{OH} or capacitive load specifications.*
2. *The total power requirements vs. power budget must be considered, including the extra drain on the microcontroller.*
3. *The microcontroller I_{CC} max must not be exceeded.*
4. *Normal decoupling methods must be employed.*
5. *The microcontroller I_{OH} for the port in use must not be exceeded.*

Figure F shows a typical power on to start bit sequence. Notice that the device is available to receive a clock at 5 µs after V_{CC} has become stable.

FIGURE F



Many applications, especially remote or handheld data acquisition applications, where power consumption is at a premium or battery life is critical can use the POWER PORT technique with the PIC16/17 microcontrollers and possibly other microcontrollers. Remote metering applications where the microcontroller must wake up and report previously stored data or periodically sample inputs, such as gas, electrical, or water monitoring systems are good examples where POWER PORT would be beneficial. Underground monitoring equipment for fuel storage and environmental monitoring systems are also suitable applications.

Authors: *Richard J. Fisher*
Memory Products Division

Bruce Negley
Memory Products Division

LISTING A

```
LIST P=16C54
;
;
;    Sample test program to power up serial EEPROM
;    using PIC16/17 port A, then write one byte and read same byte, then repeat forever.
;
;*******************************************************************************************
port_a  equ     5h     ; port 5 used for device
                       ; address select
port_b  equ     6h     ; port 6 used for data and
                       ; clock lines
eeprom  equu    0ah    ; bit buffer
addr    equ    0ch     ; address register
datai   equ    0dh     ; stored data input reg.
datao   equ    0eh     ; stored data output reg.
slave   equ    0fh     ; device address
                       ; (1010xxx0)
txbuf   equ    10h     ; tx buffer
count   equ    11h     ; bit counter
bcount  equ    12h     ; byte counter
rxbuf   equ    13h     ; receive buffer
loops   equ    15h     ; delay loop counter
loops2  equ    16h     ; delay loop counter 2
;
;       Bit Assignments
;
di      equ    7               ; eeprom input
do      equ    6               ; eeprom output
sdata   equ    7               ; data line (port_b)
sclk    equ    6               ; clock line (port_b)
vcc     equ    3               ; vcc for dut (port_a)
;
        org    01ffh
begin   goto   PWRUP
        org    000h
        goto   PWRUP
;
;*******************************************************************************************
;        DELAY ROUTINE
;        this routine takes the value in loops and loops that many times.  Every
;        increase in 'loops' yields approx 1 more millisecond.
;        i.e., if 'loops' is 10 then the wait period is approx 10 milliseconds.
;
;___________________________________________________________________________________________
WAIT
;

top2    movlw  .110
        movwf  loops2
top     nop            ; sit and wait
        nop
        nop
        nop
        nop
        nop
        decfsz loops2  ; inner loop done?
        goto   top     ; no, go again
        decfsz loops   ; outer loop done?
        goto   top2    ; no, go again
        retlw  0       ; yes, return from sub
;
;*******************************************************************************************
```

```
;       Start Bit Subroutine
;       this routine generates a start bit
;——————————————————————————————————————————————————————————————————————————————————————————
;
BSTART
        movlw   b'00111111'
        tris    port_b          ; port b for output
        bsf     port_b,sdata    ; set clock high
        nop
        nop
        bsf     port_b,sclk     ; set clock high
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        bcf     port_b,sdata    ; data line goes low during high clock for start bit
        nop
        nop
        nop
        nop
        nop
        nop
        bcf     port_b,sclk     ; start clock train
        nop
        nop
        nop
        retlw   0
         ;
         ;    End of Subroutine
;*******************************************************************************************
;
;
;       Stop Bit Subroutine
;       this routine generates a stop bit
;——————————————————————————————————————————————————————————————————————————————————————————
BSTOP
        movlw   b'00111111'     ;
        tris    port_b          ; set data/clock lines as outputs
        bcf     port_b,sdata    ; make sure data line is low
        nop
        nop
        nop
        nop
        nop
        nop
        bsf     port_b,sclk     ; set clock high
        nop
        nop
        nop
        nop
        nop
        nop
        bsf     port_b,sdata    ; data goes high while clock high
                                ; for stop bit
        nop
        nop
        nop
        nop
        nop
        bcf     port_b,sclk     ; set clock low again
        nop
        nop
        nop
        retlw   0
;
;           End of Subroutine
;*********************************************************************************
```

```
;       Serial data send 1 bit from PIC16/17 to dut
;------------------------------------------------------------------------------------------
BITOUT
        movlw   b'00111111'     ; set data,clock as outputs
        tris    port_b
        btfss   eeprom,do
        goto    BIT0
        bsf     port_b,sdata    ; output bit 0
        goto    CLK1            ; data line clocked low by device
;
BIT0
        bcf     port_b,sdata    ; output bit 0
CLK1
        nop
        nop
        bsf     port_b,sclk     ; set clock line high
BIT2
        nop
        nop
        nop
        nop
        bcf     port_b,sclk     ; return clock line low
        retlw    0
;
;       End of Subroutine
;
;*******************************************************************************************
;       Bit in routine
;       this routine gets a bit of data from the part
;        into the 'eeprom' register, bit 'di'
;------------------------------------------------------------------------------------------
BITIN
        movlw   b'10111111'     ; make sdata an input line
        tris    port_b
        bcf     eeprom,di       ; assume input bit low
        bsf     port_b,sclk     ; set clock line high
        nop                     ; just sit here a sec
        nop
        nop
        nop
        nop
        nop
        nop
        nop                     ;
        btfsc   port_b,sdata    ; read data line
        bsf     eeprom,di       ; set input bit if needed
        bcf     port_b,sclk     ; set clock line low
        retlw   0               ; hit the road
;
;*******************************************************************************************
;
;       Transmit Data Subroutine
;------------------------------------------------------------------------------------------
TX
        movlw   .8
        movwf   count           ; set the #bits to 8
        ;
TXLP
        bcf     eeprom,do
        btfsc   txbuf,7
        bsf     eeprom,do       ; otherwise data bit =1
        call    BITOUT          ; serial data out
        rlf     txbuf           ; rotate txbuf left
        decfsz  count           ; 8 bits done?
        goto    TXLP    ; no - go again
        call    BITIN   ; read ack bit
        ;
        retlw   0
;       End of Subroutine
;*******************************************************************************************
```

7

```
;       Receive data Routine
;       this routine gets a byte of data from the part into 'rxbuf'
;-----------------------------------------------------------------------------------------
RX
        movlw   .8              ; set # bits to 8
        movwf   count
        clrf    rxbuf           ; clear receive buffer
RXLP    rlf     rxbuf           ; rotate buffer left 1 bit
        bcf     rxbuf,0         ; assume bit is zero
        call    BITIN           ; read a bit
        btfsc   eeprom,di       ; input bit high?
         bsf    rxbuf,0         ; yes, set buffer bit high
        decfsz  count           ; 8 bits done?
        goto    RXLP            ; no, do another
        bcf     eeprom,do       ; set ack bit = 0
        call    BITOUT          ; to finish transmission
        retlw   0
;
;*******************************************************************************************
;       Power up routine
;       this routine blinks the lights
;-----------------------------------------------------------------------------------------
PWRUP
        movlw   b'00000001'
        tris    port_a          ; set RA0 as input, rest output
        bsf     port_a,vcc      ; turn on power to dut
        nop                     ; wait for dut to power up
        nop
        nop
        nop
        nop
;
;*******************************************************************************************
;       Byte Write Routine
;       this writes the data in "55h" to the first byte
;       in the serial EEPROM.
;-----------------------------------------------------------------------------------------
;
WRBYTE
                                ;
        movlw   b'10100000'     ; set slave address and write mode
        movwf   slave
        movlw   b'01010101'     ; set data to 55h
        movwf   datao
        ;
        clrf    addr            ; set address to 00h
        ;
        call    BSTART          ; generate start bit
        movf    slave,w         ; get slave address
        movwf   txbuf           ; into transmit buffer
        call    TX              ; and send it
        movf    addr,w          ; get word address
        movwf   txbuf           ; into transmit buffer
        call    TX              ; and send it
        movf    datao,w         ; move data
        movwf   txbuf           ; to tranmit buffer
        call    TX              ; and transmit it
        call    BSTOP           ; generate stop bit
                                ;
        movlw                   .10
        movwf   loops   ; set delay time to give
        call    WAIT        ; 10 ms wait after every byte
                        ;
                        ; now drop through and do the read
                        ;
;*******************************************************************************************
;       READ (read routine)
;       this routine reads the first address
;       of the dut
```

```
;————————————————————————————————————————————————————————————————————————————
READ
                              ;
        movlw   b'10100000'   ; set slave address and write mode
        movwf   slave
                              ;
        clrf    addr          ; set address to 00h
                              ;
        call    BSTART        ; generate start bit
        nop
        nop
        movf    slave,w       ; get slave address
        movwf   txbuf         ; into transmit buffer
        call    TX            ; and send it
        movf    addr,w        ; get word address
        movwf   txbuf         ; into transmit buffer
        call    TX            ; and send it
        nop
        nop
        call    BSTART        ; generate start bit
        nop
        nop
        movlw   b'10100001'   ; get slave address and read mode
        movwf   txbuf         ; into transmit buffer
        call    TX            ; and transmit it
        nop
        call    RX            ; get 8 bits of data
        bsf     eeprom,do
        call    BITOUT        ; send high ack bit and then a
        call    BSTOP         ; stop bit to end transmission from dut
        nop                   ;
        nop
        nop
        nop
        nop
        bcf     port_a,vcc    ; turn power to dut off
        movlw   .100
        movwf   loops
        call    WAIT          ; wait awhile
        goto    PWRUP         ; go do the whole thing over again
;
END
```

NOTES:

WORLDWIDE SALES & SERVICE

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/microhip

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

AMERICAS (continued)

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
Metroplaza
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

9/22/95