

---

## A PC-Based Development Programmer for the PIC16C84

---

*Author: Robert Spur  
Analog Design Specialist, Inc.*

### INTRODUCTION

This application note describes the construction of a low cost serial programmer which uses a PC with a parallel (Centronix printer) port to control a PIC16C84. This programmer has the capability of programming a PIC16C84 microcontroller, and reading back internal data without removing the device from the target circuit.

This feature is very useful in applications where changes in program code or program constants are necessary to compensate for other system features. For example, an embedded control system may have to compensate for variances in a mechanical actuator's performance or loading. The basic program can be programmed and tested during design phase. The final program and control constants can be easily added later in the production phase without removing the microcontroller from the circuit.

Automatic software and performance upgrades can also be implemented in-system. Upon receiving new system software via disk or modem, a control processor with the included programming code could perform in-circuit reprogramming of other microcontrollers in the system.

This programmer can load program code, part configuration, and EEPROM data into the PIC16C84. In read back mode, it can verify all data entries.

### PROGRAMMING DESCRIPTION

The PIC16C84 microcontroller is placed into programming mode by forcing a low logic level on RB7 (pin 13) and RB6 (pin 12) while  $\overline{\text{MCLR}}$  (pin 4) is first brought low to reset the part, and then brought to the program/verify voltage of 12 to 14V. The  $\overline{\text{MCLR}}$  pin remains at the program/verify voltage for the remainder of the programming or verification.

After entering programming mode, RB7 is used to serially enter programming modes and data into the part. A high to low transition on RB6, the clock input, qualifies each bit of the data applied on RB7. Please refer to the PIC16C84 Programming Specification (DS30189) for details on the figures. The serial command-data format is specified in Figure 1.2.1.3 of the Microchip PIC16C84 Programming Specification (DS30189). The first 6 bits form the command field, and the last 16 bits form the data field. Notice that the data field is composed of one zero starting bit, 14 actual data bits, and one zero stop bit. The increment address command, shown in Figure 1.2.1.5 (PIC16C84 Programming Specification, DS30189), is comprised of only the command field. Table 1.2.1.1 (see DS30189) summarizes the available commands and command codes for serial programming mode.

Read mode is similar to programming mode with the exception that the data direction of RB7 is reversed after receiving the 6-bit command to allow the requested data to be returned to the programmer. Figure 1.2.1.4 (see DS30189) shows this sequence which starts by shifting the 6-bit command into the part. After the read command is issued, the programmer tri-states its buffer to allow the part to serially shift its internal data back to the programmer. The rising edge of RB6, (the clock input), controls the data flow by sequentially shifting previously programmed or data bits from the part. The programmer qualifies this data on the falling edge of RB6. Notice that 16 clock cycles are necessary to shift out 14 data bits.

Accidental in-circuit reprogramming is prevented during normal operation by the  $\overline{\text{MCLR}}$  voltage which should never exceed the maximum circuit supply voltage of 6 VDC and the logic levels of port bits RB7 and RB6.

After program/verification the  $\overline{\text{MCLR}}$  pin is brought low to reset the target microcontroller and then electrically released. The target circuit is then free to activate the  $\overline{\text{MCLR}}$  signal. In the event  $\overline{\text{MCLR}}$  is not forced by the target circuit, R4 (a 2 k $\Omega$  pull-up resistor in the programmer) provides a high logic level on the target microcontroller which enables execution of its program independent of the programmer connection. Provisions should be made to prevent the target circuit from resetting the target microcontroller with  $\overline{\text{MCLR}}$  or affecting the RB7 and RB6 pins during the programming process. In most cases this can be done without jumpers.

A logic high on PC parallel interface latch bit D4 turns on Q3 causing the MCLR pin to go low which places the target part in reset mode. The reset condition is then removed and the program/verify voltage is applied by placing a logic high on D3 and a logic low on D4 which turns off Q3 and turns on Q2 and Q1. Circuit protection of Q1 and Q3 is obtained from connecting the emitter of Q2 to latch bit D4 which prevents a simultaneous reset and program/verify voltage mode. Q2, a 2N3904, has a reverse emitter base breakdown voltage of 6V which will not be exceeded when 5V logic is used on the parallel interface.

Data and clock are connected to the part via tri-state buffer, U2. PC parallel port interface bit D0 is used for data and port bit D1 is used for clock. During programming mode both clock and data buffers are enabled by port bits D2 and D5. During read mode, the data buffer is tri-stated via D2 and the printer data acknowledge signal line is used to receive verification data from the part.

An optional 5V line was included in the 3-foot programming interconnect cable for convenience. Short interconnection leads and good grounding are always good construction practice.

To meet the programming/verification specification, the target part's supply voltage should first be set to the maximum specified supply voltage and a program/data read back should then be performed. This process is then repeated at the lowest specified supply voltage.

**PIC16C84 INTERFACE**

Resistors: 1/4 watt, 5%

## SOFTWARE DESCRIPTION

The listed code provides a hardware-software interface to a standard PC parallel (Centronix) interface port. The code can be adapted to a microprocessor parallel interface port by substituting an output command for the "biosprint" command.

Control software can transfer the PIC16C84 program, configuration bits, and EEPROM data from a standard PROM interface file into the target system by reading the file and calling the function in Example 1 using the appropriate command name in the definition table, and the data to be programmed. The command names are repeated here for reference.

LOAD_CONFIG	Sets PIC16C84 data pointer to configuration.
LOAD_DATA	Loads, but does not program, data.
READ_DATA	Reads data at current pointer location.
INC_ADDR	Increments PIC16C84 data pointer.

BEGIN_PROG	Programs data at current data pointer location.
PARALLEL_MODE	Puts PIC16C84 into parallel mode (not used).
LOAD_DATA_DM	Loads EEPROM data.
READ_DATA_DM	Reads EEPROM data.

Function "int ser\_pic16c84(<command>,<data [or 0]>)" is called to perform command. Function returns internal data after read commands.

Do not forget to initiate the programming mode before programming, increment the addresses after each byte is programmed, and put the programmer in run mode after programming.

Designed by: Analog Design Specialist, Inc.  
P.O. Box 26-0846  
Littleton, CO 80126

### EXAMPLE 1: PUT TARGET SYSTEM INTO PROGRAM MODE

```
.. program code..
ser_pic16c84(PROGRAM_MODE,0);
.. program code..
```

### EXAMPLE 2: READ DATA FROM THE TARGET SYSTEM

```
.. program code..
data = ser_pic16c84(READ_DATA,0); // read data
// transfers data from target part to variable "data"
.. more program code..
```

### EXAMPLE 3: PROGRAM DATA INTO THE TARGET SYSTEM

```
.. program code..
ser_pic16c84(LOAD_DATA,data); // load data into target
ser_pic16c84(BEGIN_PROG,0); // program loaded data
ser_pic16c84(INC_ADDR,0); // increment to next address
// transfers data from program variable "data" to target part
.. more program code..
```

### EXAMPLE 4: PUT TARGET SYSTEM INTO RUN MODE

```
.. program code..
ser_pic16c84(RUN,0);
.. program code..
```

```
//***** FIGURE #2 *****/
/**
/** SERIAL PROGRAMMING ROUTINE FOR THE PIC16C84 MICROCONTROLLER
/**
/**
/** Analog Design Specialists
/**
/**
//*****

//FUNCTION PROTOTYPE: int ser_pic16c84(int cmd, int data)

// cmd: LOAD_CONFIG -> part configuration bits
// LOAD_DATA -> program data, write
// READ_DATA -> program data, read
// INC_ADDR -> increment to the next address (routine does not auto increment)
// BEGIN_PROG -> program a previously loaded program code or data
// LOAD_DATA_DM -> load EEPROM data registers (BEGIN_PROG must follow)
// READ_DATA_DM -> read EEPROM data
//
// data: 1) 14 bits of program data or
// 2) 8 bits of EEPROM data (least significant 8 bits of int)

// Additional programmer commands (not part of PIC16C84 programming codes)
//
// cmd: RESET -> provides 1 ms reset pulse to target system
// PROGRAM_MODE -> initializes PIC16C84 for programming
// RUN -> disconnects programmer from target system
//
// function returns:1) 14 or 8 bits read back data for read commands
// 2) zero for write data commands
// 3) PIC_PROG_EROR = -1 for programming function errors
// 4) PROGMR_ERROR = -2 for programmer function errors

#include <bios.h>

#define LOAD_CONFIG 0
#define LOAD_DATA 2
#define READ_DATA 4
#define INC_ADDR 6
#define BEGIN_PROG 8
#define PARALLEL_MODE 10 // not used
#define LOAD_DATA_DM 3
#define READ_DATA_DM 5
#define MAX_PIC_CMD 63 // division between pic16c84 and programmer commands

#define RESET 64 // external reset command, not needed for programming
#define PROGRAM_MODE 65 // initialize program mode
#define RUN 66 // electrically disconnect programmer

#define PIC_PROG_EROR -1
#define PROGMR_ERROR -2

#define PTR 0 // use device #0

// parallel port bits
// d0: data output to part to be programmed
// d1: programming clock
// d2: data dirrection, 0= enable tri state buf -> send data to part
// d3: Vpp control 1= turn on Vpp
// d4: ~MCLR =0, 1 = reset device with MCLR line
// d5: clock line tri state control, 0 = enable clock line

int ser_pic16c84(int cmd, int data) // custom interface for pic16c84
{
    int i, s_cmd;
```

```

if(cmd <=MAX_PIC_CMD)                                // all programming modes
{
    biosprint(0,8,PTR);                                // set bits 001000, output mode, clock & data low
    s_cmd = cmd;                                        // retain command "cmd"
    for (i=0;i<6;i++)                                // output 6 bits of command
    {
        biosprint(0,(s_cmd&0x1) +2+8,PTR);            // set bits 001010, clock hi
        biosprint(0,(s_cmd&0x1)  +8,PTR);            // set bits 001000, clock low
        s_cmd >>=1;
    }

    if((cmd ==INC_ADDR)|| (cmd ==PARALLEL_MODE)        // command only, no data cycle
        return 0;

    else if(cmd ==BEGIN_PROG)                          // program command only, no data cycle
    {
        delay(10);                                    // 10 ms PIC programming time
        return 0;
    }

    else if((cmd ==LOAD_DATA)|| (cmd ==LOAD_DATA_DM)|| (cmd ==LOAD_CONFIG)) // output 14 bits
    {
        for (i=200;i;i--) ;                            // delay between command & data
        biosprint(0,2+8,PTR);                          // set bits 001010, clock hi; leading bit
        biosprint(0, 8,PTR);                          // set bits 001000, clock low

        for (i=0;i<14;i++)                            // 14 data bits, lsb first
        {
            biosprint(0,(data&0x1) +2+8,PTR);          // set bits 001010, clock hi
            biosprint(0,(data&0x1)  +8,PTR);          // set bits 001000, clock low
            data >>=1;
        }
        biosprint(0,2+8,PTR);                          // set bits 001010, clock hi; trailing bit

        // ***** Analog Design Specialists *****

        biosprint(0, 8,PTR);                          // set bits 001000, clock low

        return 0;
    }

    else if((cmd ==READ_DATA)|| (cmd ==READ_DATA_DM)) //read 14 bits from part, lsb first
    {
        biosprint(0, 4+8,PTR);                        // set bits 001100, clock low, tri state data buffer
        for (i=200;i;i--) ;                            // delay between command & data
        biosprint(0,2+4+8,PTR);                      // set bits 001110, clock hi, leading bit
        biosprint(0, 4+8,PTR);                      // set bits 001100, clock low

        data =0;
        for (i=0;i<14;i++)                            // input 14 bits of data, lsb first
        {
            data >>=1;                                // shift data for next input bit
            biosprint(0,2+4+8,PTR);                  // set bits 001110, clock hi
            biosprint(0, 4+8,PTR);                  // set bits 001100, clock low
            if(!(biosprint(2,0,0)&0x40)) data += 0x2000; //use printer acknowledge line for input,
                                                    //data lsb first
        }
        biosprint(0,2+4+8,PTR);                      // set bits 001110, clock hi, trailing bit
        biosprint(0, 4+8,PTR);                      // set bits 001100, clock low
        return data;
    }

    else return PIC_PROG_EROR;                        // programmer error

}

else if(cmd == RESET)                                // reset device

```

```
{
    biosprint(0,32+16+4,PTR);
(reset
    delay(1);
    biosprint(0,32    +4,PTR);
    return 0;
}

else if(cmd ==PROGRAM_MODE)
{
    biosprint(0,32+16+4,PTR);

    delay(10);

    biosprint(0,8,PTR);

    delay(10);

    return 0;
}

else if(cmd ==RUN)
{
    biosprint(0,32+4,PTR);
    return 0;
}
else return PROGMR_ERROR;
}
```

```
// set bits 110100, MCLR = low
// PIC16C84), programmer not connected
// 1ms delay
// set bits 100100, MCLR = high

// enter program mode

// set bits 110100, Vpp off, MCLR =low
//(reset PIC16C84)
//10 ms, allow programming voltage to stabilize

// set bits 001000, Vpp on , MCLR = 13.5 volts,
// clock & data connected
// 10 ms, allow programming voltage to stabilize

// disconnects programmer from device

// set bits 100100

// command error
```

---

**Note the following details of the code protection feature on PICmicro® MCUs.**

- The PICmicro family meets the specifications contained in the Microchip Data Sheet.
- Microchip believes that its family of PICmicro microcontrollers is one of the most secure products of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the PICmicro microcontroller in a manner outside the operating specifications contained in the data sheet. The person doing so may be engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable”.
- Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our product.

If you have any further questions about this matter, please contact the local sales office nearest to you.

---

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip's products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights.

#### **Trademarks**


The Microchip name and logo, the Microchip logo, FilterLab, KEELOQ, microID, MPLAB, PIC, PICmicro, PICMASTER, PICSTART, PRO MATE, SEEVAL and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

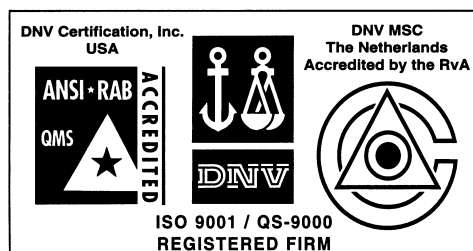
dsPIC, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, microPort, Migratable Memory, MPASM, MPLIB, MPLINK, MPSIM, MXDEV, PICC, PICDEM, PICDEM.net, rPIC, Select Mode and Total Endurance are trademarks of Microchip Technology Incorporated in the U.S.A.

Serialized Quick Turn Programming (SQTP) is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

© 2002, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

 Printed on recycled paper.



*Microchip received QS-9000 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona in July 1999. The Company's quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, KEELOQ® code hopping devices, Serial EEPROMs and microperipheral products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001 certified.*



---

## WORLDWIDE SALES AND SERVICE

---

### AMERICAS

#### Corporate Office

2355 West Chandler Blvd.  
Chandler, AZ 85224-6199  
Tel: 480-792-7200 Fax: 480-792-7277  
Technical Support: 480-792-7627  
Web Address: <http://www.microchip.com>

#### Rocky Mountain

2355 West Chandler Blvd.  
Chandler, AZ 85224-6199  
Tel: 480-792-7966 Fax: 480-792-7456

#### Atlanta

500 Sugar Mill Road, Suite 200B  
Atlanta, GA 30350  
Tel: 770-640-0034 Fax: 770-640-0307

#### Boston

2 Lan Drive, Suite 120  
Westford, MA 01886  
Tel: 978-692-3848 Fax: 978-692-3821

#### Chicago

333 Pierce Road, Suite 180  
Itasca, IL 60143  
Tel: 630-285-0071 Fax: 630-285-0075

#### Dallas

4570 Westgrove Drive, Suite 160  
Addison, TX 75001  
Tel: 972-818-7423 Fax: 972-818-2924

#### Detroit

Tri-Atria Office Building  
32255 Northwestern Highway, Suite 190  
Farmington Hills, MI 48334  
Tel: 248-538-2250 Fax: 248-538-2260

#### Kokomo

2767 S. Albright Road  
Kokomo, Indiana 46902  
Tel: 765-864-8360 Fax: 765-864-8387

#### Los Angeles

18201 Von Karman, Suite 1090  
Irvine, CA 92612  
Tel: 949-263-1888 Fax: 949-263-1338

#### New York

150 Motor Parkway, Suite 202  
Hauppauge, NY 11788  
Tel: 631-273-5305 Fax: 631-273-5335

#### San Jose

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

#### Toronto

6285 Northam Drive, Suite 108  
Mississauga, Ontario L4V 1X5, Canada  
Tel: 905-673-0699 Fax: 905-673-6509

### ASIA/PACIFIC

#### Australia

Microchip Technology Australia Pty Ltd  
Suite 22, 41 Rawson Street  
Epping 2121, NSW  
Australia  
Tel: 61-2-9868-6733 Fax: 61-2-9868-6755

#### China - Beijing

Microchip Technology Consulting (Shanghai)  
Co., Ltd., Beijing Liaison Office  
Unit 915  
Bei Hai Wan Tai Bldg.  
No. 6 Chaoyangmen Beidajie  
Beijing, 100027, No. China  
Tel: 86-10-85282100 Fax: 86-10-85282104

#### China - Chengdu

Microchip Technology Consulting (Shanghai)  
Co., Ltd., Chengdu Liaison Office  
Rm. 2401, 24th Floor,  
Ming Xing Financial Tower  
No. 88 TIDU Street  
Chengdu 610016, China  
Tel: 86-28-6766200 Fax: 86-28-6766599

#### China - Fuzhou

Microchip Technology Consulting (Shanghai)  
Co., Ltd., Fuzhou Liaison Office  
Unit 28F, World Trade Plaza  
No. 71 Wusi Road  
Fuzhou 350001, China  
Tel: 86-591-7503506 Fax: 86-591-7503521

#### China - Shanghai

Microchip Technology Consulting (Shanghai)  
Co., Ltd.  
Room 701, Bldg. B  
Far East International Plaza  
No. 317 Xian Xia Road  
Shanghai, 200051  
Tel: 86-21-6275-5700 Fax: 86-21-6275-5060

#### China - Shenzhen

Microchip Technology Consulting (Shanghai)  
Co., Ltd., Shenzhen Liaison Office  
Rm. 1315, 13/F, Shenzhen Kerry Centre,  
Renminnan Lu  
Shenzhen 518001, China  
Tel: 86-755-2350361 Fax: 86-755-2366086

#### Hong Kong

Microchip Technology Hongkong Ltd.  
Unit 901-6, Tower 2, Metroplaza  
223 Hing Fong Road  
Kwai Fong, N.T., Hong Kong  
Tel: 852-2401-1200 Fax: 852-2401-3431

#### India

Microchip Technology Inc.  
India Liaison Office  
Divyasree Chambers  
1 Floor, Wing A (A3/A4)  
No. 11, O'Shaugnessey Road  
Bangalore, 560 025, India  
Tel: 91-80-2290061 Fax: 91-80-2290062

### Japan

Microchip Technology Japan K.K.  
Benex S-1 6F  
3-18-20, Shinyokohama  
Kohoku-Ku, Yokohama-shi  
Kanagawa, 222-0033, Japan  
Tel: 81-45-471- 6166 Fax: 81-45-471-6122

### Korea

Microchip Technology Korea  
168-1, Youngbo Bldg. 3 Floor  
Samsung-Dong, Kangnam-Ku  
Seoul, Korea 135-882  
Tel: 82-2-554-7200 Fax: 82-2-558-5934

### Singapore

Microchip Technology Singapore Pte Ltd.  
200 Middle Road  
#07-02 Prime Centre  
Singapore, 188980  
Tel: 65-334-8870 Fax: 65-334-8850

### Taiwan

Microchip Technology Taiwan  
11F-3, No. 207  
Tung Hua North Road  
Taipei, 105, Taiwan  
Tel: 886-2-2717-7175 Fax: 886-2-2545-0139

### EUROPE

#### Denmark

Microchip Technology Nordic ApS  
Regus Business Centre  
Lautrup høj 1-3  
Ballerup DK-2750 Denmark  
Tel: 45 4420 9895 Fax: 45 4420 9910

#### France

Microchip Technology SARL  
Parc d'Activite du Moulin de Massy  
43 Rue du Saule Trapu  
Batiment A - 1er Etage  
91300 Massy, France  
Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79

#### Germany

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

#### Italy

Microchip Technology SRL  
Centro Direzionale Colleoni  
Palazzo Taurus 1 V. Le Colleoni 1  
20041 Agrate Brianza  
Milan, Italy  
Tel: 39-039-65791-1 Fax: 39-039-6899883

#### United Kingdom

Arizona Microchip Technology Ltd.  
505 Eskdale Road  
Winnersh Triangle  
Wokingham  
Berkshire, England RG41 5TU  
Tel: 44 118 921 5869 Fax: 44-118 921-5820

01/18/02