AN589

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 $\overline{MCLR}$ (pin 4) is first brought low to reset the part, and then brought to the program/verify voltage of 12 to 14V. The $\overline{MCLR}$ 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 $\overline{MCLR}$ 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 $\overline{MCLR}$ pin is brought low to reset the target microcontroller and then electrically released. The target circuit is then free to activate the $\overline{MCLR}$ signal. In the event $\overline{MCLR}$ is not forced by the target circuit, R4 (a 2 kΩ 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 $\overline{MCLR}$ or affecting the RB7 and RB6 pins during the programming process. In most cases this can be done without jumpers.

© 1997 Microchip Technology Inc.

DETAILED CIRCUIT DESCRIPTION

A logic high on PC parallel interface latch bit D4 turns on Q3 causing the $\overline{MCLR}$ 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.

Resistors R1, R2, R3, and diode D1 provide a logic level interface to the analog circuitry. R4 provides a $\overline{MCLR}$ (master clear) pull-up function during target circuit run mode. The programming voltage is supplied and adjusted by an external lab supply. This supply should have a current limit in the 100 mA range. 5V for U2 (LS244) is locally regulated from programming supply voltage by U1. R5 (750Ω resistor) is connected to the regulator output to insure proper 5V regulation when the 13.5V programming voltage is applied through the pull-up resistor R4.

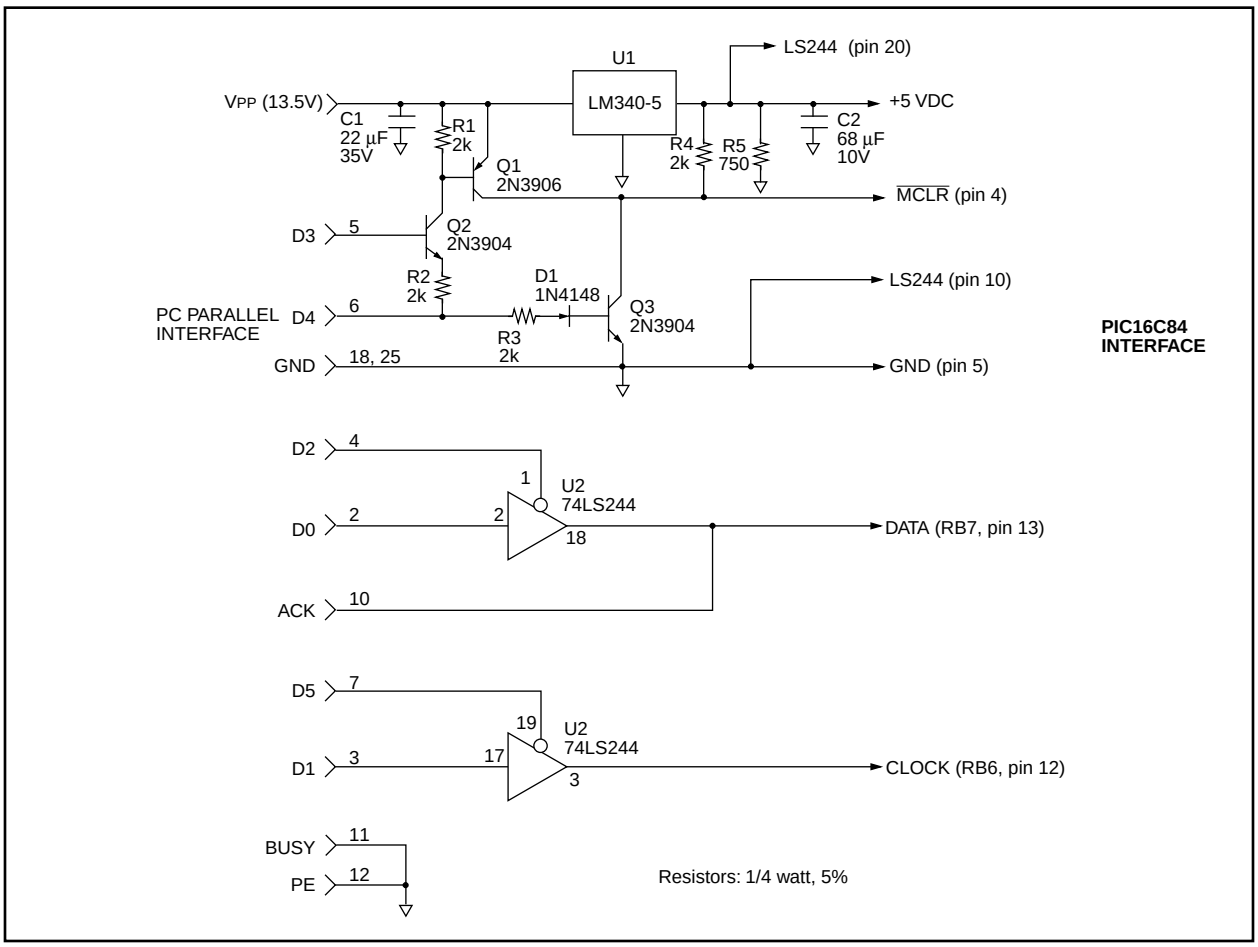
Data and clock are connected to the part via tri-state buffer, U2. PC parallel port interface bit D0 is used for data and port bit D1 is used for clock. During programming mode both clock and data buffers are enabled by port bits D2 and D5. During read mode, the data buffer is tri-stated via D2 and the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

After program/verification mode both the data and clock lines are tri-stated via D2 and D5, allowing these lines to be used by the target circuit. This allows the programmer to remain physically, but not electrically connected to the target system.

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.

FIGURE 1: PROGRAMER SCHEMATIC



© 1997 Microchip Technology Inc.

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..
```

© 1997 Microchip Technology Inc.

```
//************************* 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 regesters (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;
```


© 1997 Microchip Technology Inc.

```
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);                 // set bits 110100, MCLR = low
(reset                                        // PIC16C84), programmer not connected
    delay(1);                                 // 1ms delay
    biosprint(0,32   +4,PTR);                 // set bits 100100, MCLR = high
    return 0;
    }

  else if(cmd ==PROGRAM_MODE)                 // enter program mode
    {
    biosprint(0,32+16+4,PTR);                 // set bits 110100, Vpp off, MCLR =low
                                              //(reset PIC16C84)
    delay(10);                                //10 ms, allow programming voltage to stabilize

    biosprint(0,8,PTR);                       // set bits 001000, Vpp on , MCLR = 13.5 volts,
                                              // clock & data connected
    delay(10);                                // 10 ms, allow programming voltage to stabilize

    return 0;
    }

  else if(cmd ==RUN)                          // disconnects programmer from device
    {
    biosprint(0,32+4,PTR);                    // set bits 100100
    return 0;
    }
  else return PROGMR_ERROR;                   // command error
  }
```


© 1997 Microchip Technology Inc.

WORLDWIDE SALES AND SERVICE

AMERICAS

Corporate Office
Microchip Technology Inc.
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-786-7200 Fax: 480-786-7277
Technical Support: 480-786-7627
Web Address: http://www.microchip.com

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: 630-285-0071 Fax: 630-285-0075

Dallas
Microchip Technology Inc.
4570 Westgrove Drive, Suite 160
Addison, TX 75248
Tel: 972-818-7423 Fax: 972-818-2924

Dayton
Microchip Technology Inc.
Two Prestige Place, Suite 150
Miamisburg, OH 45342
Tel: 937-291-1654 Fax: 937-291-9175

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

Los Angeles
Microchip Technology Inc.
18201 Von Karman, Suite 1090
Irvine, CA 92612
Tel: 949-263-1888 Fax: 949-263-1338

New York
Microchip Technology Inc.
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

AMERICAS (continued)

Toronto
Microchip Technology Inc.
5925 Airport Road, Suite 200
Mississauga, Ontario L4V 1W1, Canada
Tel: 905-405-6279 Fax: 905-405-6253

ASIA/PACIFIC

Hong Kong
Microchip Asia Pacific
Unit 2101, Tower 2
Metroplaza
223 Hing Fong Road
Kwai Fong, N.T., Hong Kong
Tel: 852-2-401-1200 Fax: 852-2-401-3431

Beijing
Microchip Technology, Beijing
Unit 915, 6 Chaoyangmen Bei Dajie
Dong Erhuan Road, Dongcheng District
New China Hong Kong Manhattan Building
Beijing 100027 PRC
Tel: 86-10-85282100 Fax: 86-10-85282104

India
Microchip Technology Inc.
India Liaison Office
No. 6, Legacy, Convent Road
Bangalore 560 025, India
Tel: 91-80-229-0061 Fax: 91-80-229-0062

Japan
Microchip Technology Intl. Inc.
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
Tel: 82-2-554-7200 Fax: 82-2-558-5934

Shanghai
Microchip Technology
RM 406 Shanghai Golden Bridge Bldg.
2077 Yan'an Road West, Hong Qiao District
Shanghai, PRC 200335
Tel: 86-21-6275-5700 Fax: 86 21-6275-5060

ASIA/PACIFIC (continued)

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

Taiwan, R.O.C
Microchip Technology Taiwan
10F-1C 207
Tung Hua North Road
Taipei, Taiwan, ROC
Tel: 886-2-2717-7175 Fax: 886-2-2545-0139

EUROPE

United Kingdom
Arizona Microchip Technology Ltd.
505 Eskdale Road
Winnersh Triangle
Wokingham
Berkshire, England RG41 5TU
Tel: 44 118 921 5858 Fax: 44-118 921-5835

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

France
Arizona Microchip Technology SARL
Parc d'Activite du Moulin de Massy
43 Rue du Saule Trapu
Batiment A - ler Etage
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 München, Germany
Tel: 49-89-627-144 0 Fax: 49-89-627-144-44

Italy
Arizona 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

11/15/99

DNV Certification, Inc. USA — ANSI-RAB QMS ACCREDITED — DNV — DNV MSC The Netherlands Accredited by the RvA
ISO 9001 / QS-9000 REGISTERED FIRM

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.

All rights reserved. © 1999 Microchip Technology Incorporated. Printed in the USA. 11/99 Printed on recycled paper.

Information contained in this publication regarding device applications and the like is intended for suggestion only and may be superseded by updates. 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. The Microchip logo and name are registered trademarks of Microchip Technology Inc. in the U.S.A. and other countries. All rights reserved. All other trademarks mentioned herein are the property of their respective companies.