MICROCHIP

Section 28. In-Circuit Serial Programming™ (ICSP™)

HIGHLIGHTS

This section of the manual contains the following major topics:

28.1 Introduction

All midrange devices can be In-Circuit Serial Programmed (ICSP™) while in the end application circuit. This is simply done with two lines for clock and data, and three other lines for power, ground, and the programming voltage.

In-Circuit Serial Programming (ICSP™) is a great way to reduce your inventory overhead and time-to-market for your product. By assembling your product with a blank Microchip microcontroller (MCU), you can stock one design. When an order has been placed, these units can be programmed with the latest revision of firmware, tested, and shipped in a very short time. This method also reduces scrapped inventory due to old firmware revisions. This type of manufacturing system can also facilitate quick turnarounds on custom orders for your product.

Most people would think to use ICSP with PICmicro™ OTP MCUs only on an assembly line where the device is programmed once. However, there is a method by which an OTP device can be programmed several times depending on the size of the firmware. This method, explained later, provides a way to field upgrade your firmware in a way similar to EEPROM- or Flash-based devices.

28.2 Entering In-Circuit Serial Programming Mode

The device is placed into a program/verify mode by holding the RB6 and RB7 pins low while raising the $\overline{\text{MCLR}}$ (V_{PP}) pin from V_{IL} to V_{IHH} (see programming specification) and having V_{DD} at the programming voltage. RB6 becomes the programming clock and RB7 becomes the programming data. Both RB6 and RB7 are Schmitt Trigger inputs in this mode, and when RB7 is driving data it is a CMOS output driver.

After reset, to place the device into programming/verify mode, the program counter (PC) is at location 00h. A 6-bit command is then supplied to the device. Some commands then specify that 14-bits of program data are then supplied to or read from the device, depending on if the command was a load or a read. For complete details of serial programming, please refer to the device specific Programming Specifications.

During the In-Circuit Serial Programming Mode, the WDT circuitry is disabled from generating a device reset.

28.3 Application Circuit

The application circuit must be designed to allow all the programming signals to be directly connected to the PICmicro MCU. Figure 28-1 shows a typical circuit that is a starting point for when designing with ICSP. The application must compensate for the following issues:

- Isolation of the $\overline{\text{MCLR}}$/VPP pin from the rest of the circuit
- Loading of pins RB6 and RB7
- Capacitance on each of the VDD, $\overline{\text{MCLR}}$/VPP, RB6, and RB7 pins
- Minimum and maximum operating voltage for VDD
- PICmicro Oscillator
- Interface to the programmer

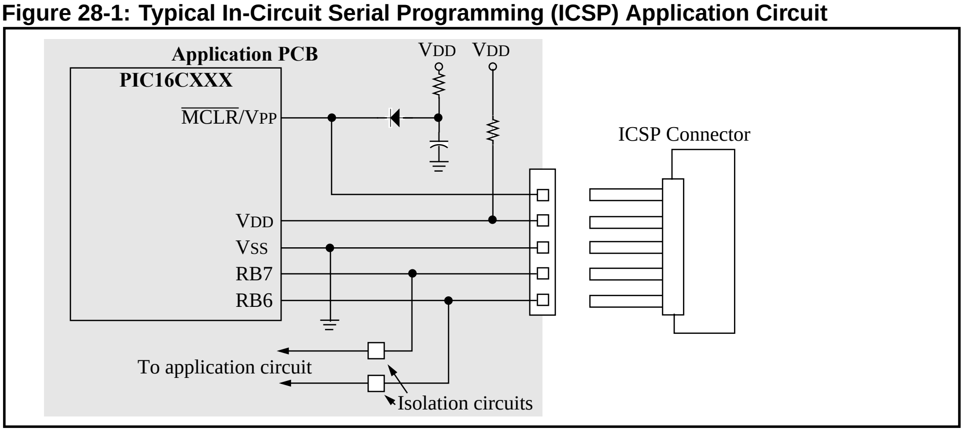
The $\overline{\text{MCLR}}$/VPP pin is normally connected to an RC circuit. The pull-up resistor is tied to VDD and a capacitor is tied to ground. This circuit can affect the operation of ICSP depending on the size of the capacitor since the VPP voltage must be isolated from the rest of the circuit (in most cases a resistor is not capable of isolating the circuit). It is, therefore, recommended that the circuit in Figure 28-1 be used when an RC is connected to $\overline{\text{MCLR}}$/VPP. The diode should be a Schottky-type device. Another issue with $\overline{\text{MCLR}}$/VPP is that when the PICmicro device is programmed, this pin is driven to approximately 13V and also to ground. Therefore, the application circuit must be isolated from this voltage provided by the programmer.

Pins RB6 and RB7 are used by the PICmicro for serial programming. RB6 is the clock line and RB7 is the data line. RB6 is driven by the programmer. RB7 is a bi-directional pin that is driven by the programmer when programming, and driven by the PICmicro when verifying. These pins must be isolated from the rest of the application circuit so as not to affect the signals during programming. You must take into consideration the output impedance of the programmer when isolating RB6 and RB7 from the rest of the circuit. This isolation circuit must account for RB6 being an input on the PICmicro, and for RB7 being bi-directional (can be driven by both the PICmicro and the programmer). For instance, PRO MATE® II has an output impedance of 1kΩ. If the design permits, these pins should not be used by the application. This is not the case with most applications so it is recommended that the designer evaluate whether these signals need to be buffered. As a designer, you must consider what type of circuitry is connected to RB6 and RB7 and then make a decision on how to isolate these pins. Figure 28-1 does not show any circuitry to isolate RB6 and RB7 on the application circuit because this is very application dependent.

To simplify this interface the optimal usage of these I/O in the application are (in order):

1. Do not use RB6/RB7 so they are dedicated to ICSP.
2. Use these pins as outputs with minimal loading on signal line.
3. Isolation circuitry so that these signals can be driven to the ICSP specifications.

Figure 28-1: Typical In-Circuit Serial Programming (ICSP) Application Circuit



The total capacitance on the programming pins affects the rise rates of these signals as they are driven out of the programmer. Typical circuits use several hundred microfarads of capacitance on VDD which helps to dampen noise and ripple. However, this capacitance requires a fairly strong driver in the programmer to meet the rise rate timings for VDD. Most programmers are designed to simply program the PICmicro itself and don't have strong enough drivers to power the application circuit. One solution is to use a driver board between the programmer and the application circuit. The driver board requires a separate power supply that is capable of driving the VPP and VDD pins with the correct rise rates and should also provide enough current to power the application circuit. RB6 and RB7 are not buffered on this schematic but may require buffering depending upon the application. A sample driver board schematic is shown in Figure 28-2.

Note:	The driver board design MUST be tested in the user's application to determine the effects of the application circuit on the programming signals timing. Changes may be required if the application places a significant load on VDD, VPP, RB6 OR RB7.

The Microchip programming specification states that the device should be programmed at 5V. Special considerations must be made if your application circuit operates at 3V only. These considerations may include totally isolating the PICmicro during programming. The other issue is that the device must be verified at the minimum and maximum voltages at which the application circuit will be operating. For instance, a battery operated system may operate from three 1.5V cells giving an operating voltage range of 2.7V to 4.5V. The programmer must program the device at 5V and must verify the program memory contents at both 2.7V and 4.5V to ensure that proper programming margins have been achieved. This ensures the PICmicro operation over the voltage range of the system.

The final issue deals with the oscillator circuit on the application board. The voltage on $\overline{\text{MCLR}}$/VPP must rise to the specified program mode entry voltage before the device executes any code. The crystal modes available on the PICmicro are not affected by this issue because the Oscillator Start-up Timer waits for 1024 oscillations before any code is executed. However, RC oscillators do not require any start-up time and, therefore, the Oscillator Start-up Timer is not used. The programmer must drive $\overline{\text{MCLR}}$/VPP to the program mode entry voltage before the RC oscillator toggles four times. If the RC oscillator toggles four or more times, the program counter will be incremented to some value X. Now when the device enters programming mode, the program counter will not be zero and the programmer will start programming your code at an offset of X. There are several alternatives that can compensate for a slow rise rate on $\overline{\text{MCLR}}$/VPP. The first method would be to not populate the R, program the device, and then insert the R. The other method would be to have the programming interface drive the OSC1 pin of the PICmicro to ground while programming. This will prevent any oscillations from occurring during programming.

Now all that is left is how to connect the application circuit to the programmer. This depends a lot on the programming environment and will be discussed in that section.

28.4 Programmer

The second consideration is the programmer. PIC16CXXX MCUs only use serial programming and therefore all programmers supporting these devices will support ICSP. One issue with the programmer is the drive capability. As discussed before, it must be able to provide the specified rise rates on the ICSP signals and also provide enough current to power the application circuit. Figure 28-2 shows an example driver board. This driver schematic does not show any buffer circuitry for RB6 and RB7. It is recommended that an evaluation be performed to determine if buffering is required. Another issue with the programmer is what V_{DD} levels are used to verify the memory contents of the PICmicro. For instance, the PRO MATE II verifies program memory at the minimum and maximum V_{DD} levels for the specified device and is therefore considered a production quality programmer. On the other hand, the PICSTART® Plus only verifies at 5V and is for prototyping use only. The Microchip programming specifications state that the program memory contents should be verified at both the minimum and maximum V_{DD} levels that the application circuit will be operating. This implies that the application circuit must be able to handle the varying V_{DD} voltages.

There are also several third party programmers that are available. You should select a programmer based on the features it has and how it fits into your programming environment. The *Microchip Development Systems Ordering Guide* (DS30177) provides detailed information on all our development tools. The *Microchip Third Party Guide* (DS00104) provides information on all of our third party tool developers. Please consult these two references when selecting a programmer. Many options exist including serial or parallel PC host connection, stand-alone operation, and single or gang programmers. Some of the third party developers include Advanced Transdata Corporation, BP Microsystems, Data I/O, Emulation Technology and Logical Devices.

28.5 Programming Environment

The programming environment will affect the type of programmer used, the programmer cable length, and the application circuit interface. Some programmers are well suited for a manual assembly line while others are desirable for an automated assembly line. You may want to choose a gang programmer to program multiple systems at a time.

The physical distance between the programmer and the application circuit affects the load capacitance on each of the programming signals. This will directly affect the drive strength needed to provide the correct signal rise rates and current. This programming cable must also be as short as possible and properly terminated and shielded, or the programming signals may be corrupted by ringing or noise.

Finally, the application circuit interface to the programmer depends on the size constraints of the application circuit itself and the assembly line. A simple header can be used to interface the application circuit to the programmer. This might be more desirable for a manual assembly line where a technician plugs the programmer cable into the board. A different method is the use of spring loaded test pins (commonly referred to as pogo pins). The application circuit has pads on the board for each of the programming signals. Then there is a fixture that has pogo pins in the same configuration as the pads on the board. The application circuit or fixture is moved into position such that the pogo pins come into contact with the board. This method might be more suitable for an automated assembly line.

After taking into consideration the issues with the application circuit, the programmer, and the programming environment, anyone can build a high quality, reliable manufacturing line based on ICSP.

28.6 Other Benefits

ICSP provides other benefits, such as calibration and serialization. If program memory permits, it would be cheaper and more reliable to store calibration constants in program memory instead of using an external serial EEPROM. For example, if your system has a thermistor which can vary from one system to another, storing some calibration information in a table format allows the microcontroller to compensate (in software) for external component tolerances. System cost can be reduced without affecting the required performance of the system by using software calibration techniques. But how does this relate to ICSP? The PICmicro has already been programmed with firmware that performs a calibration cycle. The calibration data is transferred to a calibration fixture. When all calibration data has been transferred, the fixture places the PICmicro in programming mode and programs the PICmicro with the calibration data. Application note *AN656, In-Circuit Serial Programming of Calibration Parameters Using a PICmicro Microcontroller*, shows exactly how to implement this type of calibration data programming.

The other benefit of ICSP is serialization. Each individual system can be programmed with a unique or random serial number. One such application of a unique serial number would be for security systems. A typical system might use DIP switches to set the serial number. Instead, this number can be burned into program memory, thus reducing the overall system cost and lowering the risk of tampering.

28.7 Field Programming of PICmicro OTP MCUs

An OTP device is not normally capable of being reprogrammed, but the PICmicro architecture gives you this flexibility provided the size of your firmware is at least half that of the desired device and the device is not code protected. If your target device does not have enough program memory, Microchip provides a wide spectrum of devices from 0.5K to 8K program memory with the same set of peripheral features that will help meet the criteria.

The PIC16CXXX microcontrollers have two vectors, reset and interrupt, at locations 0x0000 and 0x0004. When the PICmicro encounters a reset or interrupt condition, the code located at one of these two locations in program memory is executed. The first listing of Example 28-2 shows the code that is first programmed into the PICmicro. The second listing of Example 28-2 shows the code that is programmed into the PICmicro for the second time.

Example 28-2 shows that to program the PICmicro a second time the memory location 0x0000, originally `goto Main` (0x2808), is reprogrammed to all 0's which happens to be a `NOP` instruction. This location cannot be reprogrammed to the new opcode (0x2860) because the bits that are 0's cannot be reprogrammed to 1's, only bits that are 1's can be reprogrammed to 0's. The next memory location 0x0001 was originally blank (all 1's) and now becomes a `goto Main` (0x2860). When a reset condition occurs, the PICmicro executes the instruction at location 0x0000 which is the `NOP`, a completely benign instruction, and then executes the `goto Main` to start the execution of code. The example also shows that all program memory locations after 0x005A are blank in the original program so that the second time the PICmicro is programmed, the revised code can be programmed at these locations. The same descriptions can be given for the interrupt vector at location 0x0004.

This method changes slightly for PICmicros with >2K words of program memory. Each of the `goto Main` and `goto ISR` instructions are replaced by the following code segment is Example 28-1 due to paging on devices with >2K words of program memory.

Example 28-1: Crossing Program Memory Pages

```
    movlw <page>              movlw <page>
    movwf PCLATH              movwf PCLATH
    goto  Main                goto  ISR
```

Now your one-time programmable PICmicro is exhibiting EEPROM- or Flash-like qualities.

Example 28-2: Programming Cycle Listing Files

```
         First Program Cycle                                                Second Program Cycle
__________________________________________________________________________________________
Prog     Opcode      Assembly                           |  Prog    Opcode  Assembly
Mem                  Instruction                        |  Mem             Instruction
------------------------------------------------------------------------------------------
0000     2808        goto  Main          ;Main loop     |  0000    0000    nop
0001     3FFF        <blank>             ; at 0x0008    |  0001    2860    goto  Main; Main now
0002     3FFF        <blank>                            |  0002    3FFF    <blank>   ;   at 0x0060
0003     3FFF        <blank>                            |  0003    3FFF    <blank>
0004     2848        goto  ISR           ; ISR at       |  0004    0000    nop
0005     3FFF        <blank>             ;   0x0048     |  0005    28A8    goto  ISR ; ISR now at
0006     3FFF        <blank>                            |  0006    3FFF    <blank>   ;   0x00A8
0007     3FFF        <blank>                            |  0007    3FFF    <blank>
0008     1683        bsf   STATUS,RP0                   |  0008    1683    bsf   STATUS,RP0
0009     3007        movlw 0x07                         |  0009    3007    movlw 0x07
000A     009F        movwf ADCON1                       |  000A    009F    movwf ADCON1
 .                                                      |   .
 .                                                      |   .
 .                                                      |   .
0048     1C0C        btfss PIR1,RBIF                    |  0048    1C0C    btfss PIR1,RBIF
0049     284E        goto  EndISR                       |  0049    284E    goto  EndISR
004A     1806        btfsc PORTB,0                      |  004A    1806    btfsc PORTB,0
 .                                                      |   .
 .                                                      |   .
 .                                                      |   .
0060     3FFF        <blank>                            |  0060    1683    bsf   STATUS,RP0
0061     3FFF        <blank>                            |  0061    3005    movlw 0x05
0062     3FFF        <blank>                            |  0062    009F    movwf ADCON1
 .                                                      |   .
 .                                                      |   .
 .                                                      |   .
00A8     3FFF        <blank>                            |  00A8    1C0C    btfss PIR1,RBIF
00A9     3FFF        <blank>                            |  00A9    28AE    goto  EndISR
00AA     3FFF        <blank>                            |  00AA    1806    btfsc PORTB,0
 .                                                      |   .
 .                                                      |   .
 .                                                      |   .
------------------------------------------------------------------------------------------
```

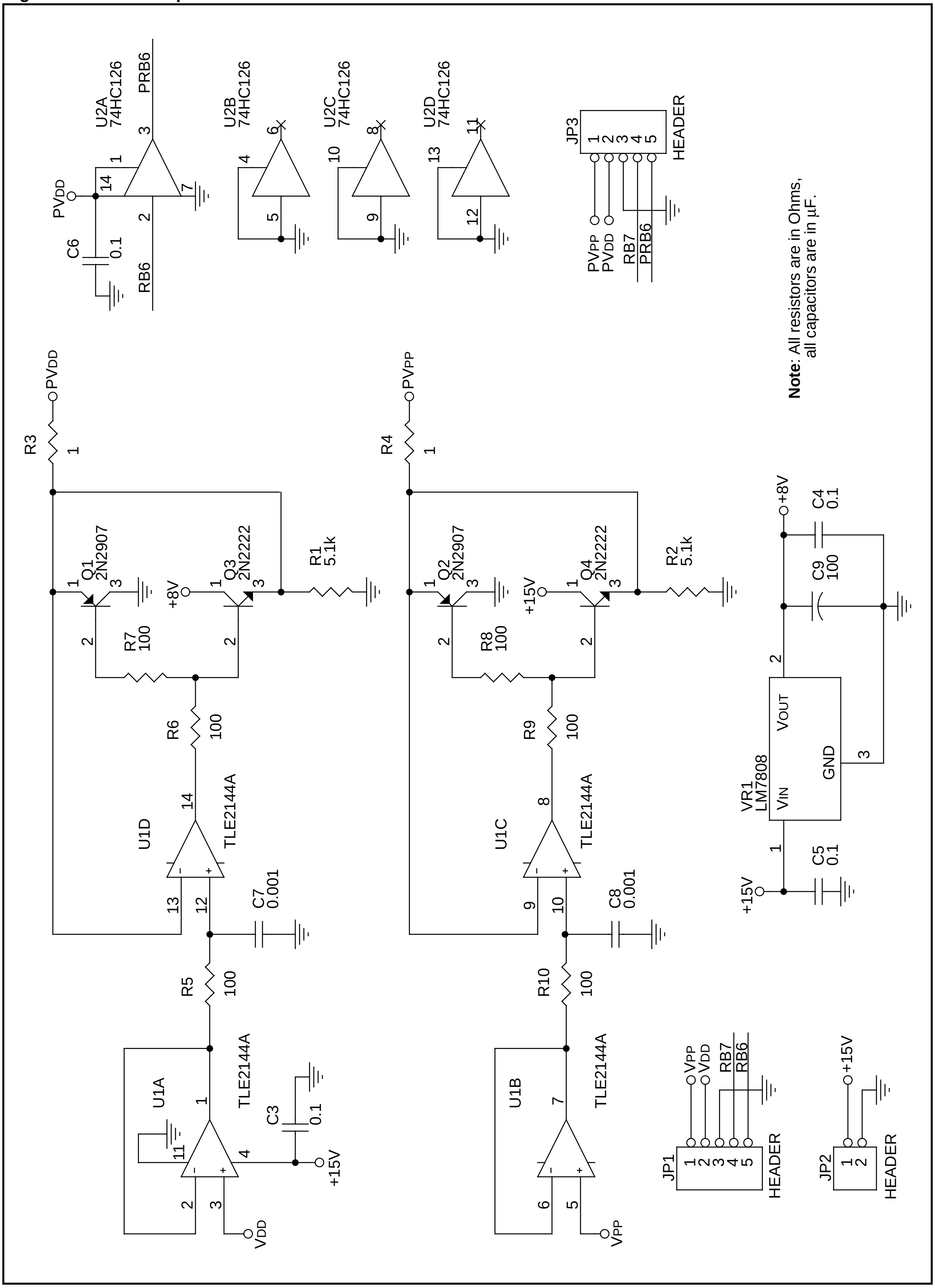
28

ICSP

28.8 Field Programming of FLASH PICmicros

With the ICSP interface circuitry already in place, FLASH-based PICmicros can be easily reprogrammed in the field. These FLASH devices allow you to reprogram them even if they are code protected. A portable ICSP programming station might consist of a laptop computer and programmer. The technician plugs the ICSP interface cable into the application circuit and downloads the new firmware into the PICmicro. The next thing you know the system is up and running without those annoying “bugs.” Another instance would be that you want to add an additional feature to your system. All of your current inventory can be converted to the new firmware and field upgrades can be performed to bring your installed base of systems up to the latest revision of firmware.

Figure 28-2: Example Driver Board Schematic



28.9 Design Tips

Question 1: ***When I try to do ICSP, the entire program is shifted (offset) in the device program memory.***

Answer 1:

If the $\overline{MCLR}$ pin does not rise fast enough, while the device's voltage is in the valid operating range, the internal Program Counter (PC) can increment. This means that the PC is no longer pointing to the address that you expected to be at. The exact location depends on the number of device clocks that occurred in the valid operating region of the device.

Question 2: ***I am using a PRO MATE II with a socket that I designed to bring the programming signal to my application board. Sometimes when I try to do ICSP, the program memory is programmed wrong.***

Answer 2:

The voltages / timings may be violated at the device. This could be due to the:

- Application board circuitry
- Cable length from programmer to target
- Large capacitance on V_{DD} which affects levels / timings

28.10 Related Application Notes

This section lists application notes that are related to this section of the manual. These application notes may not be written specifically for the Mid-Range MCU family (that is they may be written for the Base-Line, or High-End families), but the concepts are pertinent, and could be used (with modification and possible limitations). The current application notes related to In-Circuit Serial Programming are:

Title	**Application Note #**
In-Circuit Serial Programming of Calibration Parameters using a PICmicro	AN656
In-Circuit Serial Programming Guide	DS30277

28.11 Revision History

Revision A

This is the initial released revision of the In-Circuit Serial Programming description.