AN556

Implementing a Table Read

Author: *Stan D'Souza*
Logic Products Division

INTRODUCTION

This application note shows how to implement a table look-up for all PIC16/17 products. The examples shown are for the PIC16CXX family. An explanation of differences for the PIC16C5X family and PIC17C42 is at the end of this application note.

To access data in Program Memory, a table read operation must be performed. The table consists of a series of `retlw K` instructions where, the 8-bit table constants are assigned to the literal K. The first instruction in the table computes the offset to the table by using `addwf pcl`, and consequently the program branches to the appropriate `retlw X` instruction (Example 1).

EXAMPLE 1:

```
    .
    .
    movlw   offset  ;load offset in w reg
    call    Table
    .
    .
    .
Table:
    addwf   pcl     ;add offset to pc to
                    ;generate a computed goto
    retlw   'A'     ;return the ASCII char A
    retlw   'B'     ;return the ASCII char B
    retlw   'C'     ;return the ASCII char C
    .
    .
    .
```

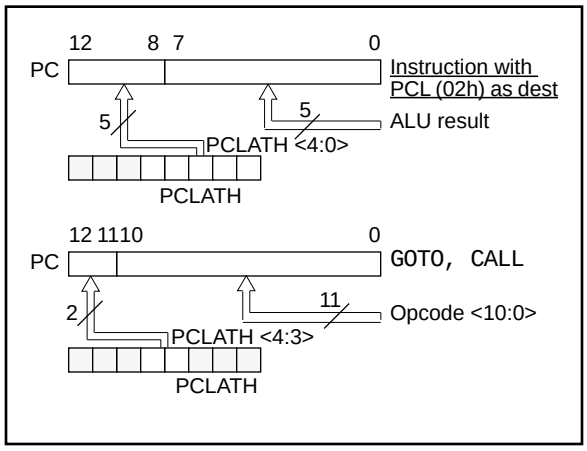
The method is straight forward, however certain precautions have to be exercised when doing a table read in the PIC16CXX.

IMPLEMENTATION

Program Counter Loading

The Program Counter (PC) in the PIC16CXX is 13-bits wide. The low 8-bits (PCL) are mapped in RAM at location 02h and are directly readable and writable. The high 5-bits are not accessible directly and can only by written through the PCLATH (Figure 1). The PCLATH is a R/W register with only five of its bits implemented <4:0>, all other bits read as '0'.

FIGURE 1: LOADING OF PC IN DIFFERENT SITUATIONS



SECTION 1

CALL and GOTO Instructions

When executing a CALL or GOTO, the low 11-bits are loaded directly from the instruction opcode. The high 2-bits are loaded from bits 3 and 4 of the PCLATH. It is a good practice to pre-load the PCLATH with the high byte of the location of the routine before executing the routine. This can be done as follows:

EXAMPLE 2:

```
    .
    .
    movlw   HIGH Table  ;load high 8 bit
                        ;address of Table
    movwf   PCLATH      ;into PCLATH
    call    Routine     ;execute Call
                        ;instruction
    .
    .
```

Note:	If the program memory size is less than 2K-bytes, then the above precaution is not necessary.

Computed GOTO Instruction

Any instruction with PCL as the destination, will load the PCH with the 5 low bits from the PCLATH (Figure 1). In Program 3, if the address where the CALL was made, was on page 0 and the address of the actual table was on page 3, then when executing the computed GOTO, the program will go to a location in page 0 instead of a location on page 3. To avoid the program from branching "erratically" when doing a table read, the PCLATH should be pre-loaded with the high byte of the "Table" address. Example 3 shows how this can be done.

EXAMPLE 3:

```
    .
    org     0x80    ;code location in page 0
    movlw   offset  ;load offset in w reg
    call    Table
    .
    .
    org     0x0320  ;Table located in page 3
Table
    addwf   pcl     ;add offset to pc to
                    ;generate a computed goto
    retlw   'A'     ;return the ASCII char A
    retlw   'B'     ;return the ASCII char B
    retlw   'C'     ;return the ASCII char C
    .
    .
    .
```

When doing a computed GOTO for a table read, care should taken about page boundaries. The ADDWF PCL instruction will not compute a value greater than 8-bits. In Example 4, the result of the computed GOTO will result in a branch to an unintended portion of the code. The user either has to be cautious as to where in a page the Table is resident or will have to monitor page roll-over and add it to the PCLATH ahead of the computed GOTO.

EXAMPLE 4:

```
    .
    org     0x80        ;code location in
                        ; page 0
    movlw   HIGH Table  ;load PCLATH with hi
                        ; address
    movwf   PCLATH      ;   /
    movlw   offset      ;load offset in w reg
    call    Table
    .
    .
    org     0x02ff      ;Table located end of
                        ; page 2
Table:
    addwf   pcl         ;value in pc will not
                        ; roll over to page 3
    retlw   'A'         ;return the ASCII
                        ; char A
    retlw   'B'         ;return the ASCII
                        ; char B
    retlw   'C'         ;return the ASCII
                        ; char C
    .
    .
    .
```

To take care of both table location and page boundary crossing, it is necessary to do a 13-bit computed GOTO operation as shown in Example 5.

The code in Example 5 will allow the user to place and access a table anywhere in program memory.

EXAMPLE 5:

```
    .
    movlw   LOW Table   ;get low 8 bits of
                        ; address
    addwf   offset      ;do a 8 bit add
                        ; operation
    movlw   HIGH Table  ;get high 5 bits of
                        ; address
    btfsc   status,c    ;page crossed?
    addlw   1           ;yes then inc high
                        ; address
    movwf   PCLATH      ;load high address in
                        ; latch
    movf    offset,w    ;load computed offset
                        ; in w reg
    call    Table
    .
    .
Table:
    movwf   pcl         ;load computed offset
                        ; in PCL
    retlw   'A'         ;return the ASCII
                        ; char A
    retlw   'B'         ;return the ASCII
                        ; char B
    retlw   'C'         ;return the ASCII
                        ; char C
    .
    .
    .
```

SECTION 2

Implementation for the PIC16C5X Family

The PIC16C5X has no PCH or PCLATH register, so the user has to take into consideration all the precautions mentioned in Section 1. In the PIC16C5X, the location of the Table has to be in the top-half of a 512 byte page. This restriction is not valid for the PIC16CXX family. To convert a table read operation from PIC16C5X code to the PIC16CXX code, the following should be done:

1. Remove any program memory bank select instructions (PIC16C56/57), if present. These are not necessary for the PIC16CXX.
2. Do a 13-bit computed GOTO operation (as shown in Example 5), when doing a table read operation.

Implementation for the PIC17C42

Unlike the PIC16CXX family, the PIC17C42 loads the PCLATH with the PCH value during a CALL or GOTO operation, however a computed GOTO, does not take care of page boundary crossing. A 16-bit computed jump address should be calculated before the table read is executed. Example 6 shows how this is done.

Example 6 allows the user to locate his Table at any program memory location, however, for large table or tables which cross page boundaries, it is recommended that the tablrd/tlrd instruction be used, these instructions are specific for table read operations and are very code efficient.

EXAMPLE 6:

```
    .
    movlw   LOW Table    ;get low 8 bits of
                         ; address
    addwf   offset       ;do a 8 bit add
                         ; operation
    movlw   HIGH Table   ;get high 8 bits of
                         ; address
    btfsc   statist,c    ;page crossed?
    addlw   1            ;yes then inc high
                         ; address
    movwf   PchBuffer,w  ;load in temp
                         ;location
    call    Table
    .
    .
Table:
    movf    PchBuffer,w  ;get high offset
    movwf   PCLATH       ;load in latch
    movf    offset,w     ;get low offset
    movwf   pcl          ;load computed
                         ; offset in PCL
    retlw   'A'          ;return the ASCII
                         ; char A
    retlw   'B'          ;return the ASCII
                         ; char B
    retlw   'C'          ;return the ASCII
                         ; char C
    .
    .
    .
```

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.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: 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.
Suite 150
Two Prestige Place
Miamisburg, OH 45342
Tel: 513 291-1654 Fax: 513 291-9175

Los Angeles
Microchip Technology Inc.
18201 Von Karman, Suite 1090
Irvine, CA 92715
Tel: 714 263-1888 Fax: 714 263-1338

AMERICAS *(continued)*

New York
Microchip Technology Inc.
150 Motor Parkway, Suite 416
Hauppauge, NY 11788
Tel: 516 273-5305 Fax: 516 273-5335

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

Toronto
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 Technology
Rm 3801B, Tower Two
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 1 628 850303 Fax: 44 1 628 850178

France
Arizona Microchip Technology SARL
Zone Industrielle de la Bonde
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 Taurus 1 V. Le Colleoni 1
20041, Agrate Brianza, Milan Italy
Tel: 39 39 689 9939 Fax: 39 39 689 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

5/10/96

MICROCHIP