Getting Started with PIC24F/PIC24H Programming and Interfacing in 'C'

This series of short articles covers the basics of programming a PIC24FJ32GA002/PIC24H 16-bit microcontroller, using Microchip's free C30 compiler. These articles assume a very basic understanding of Microchip microcontrollers and development tools, especially MPLAB IDE.

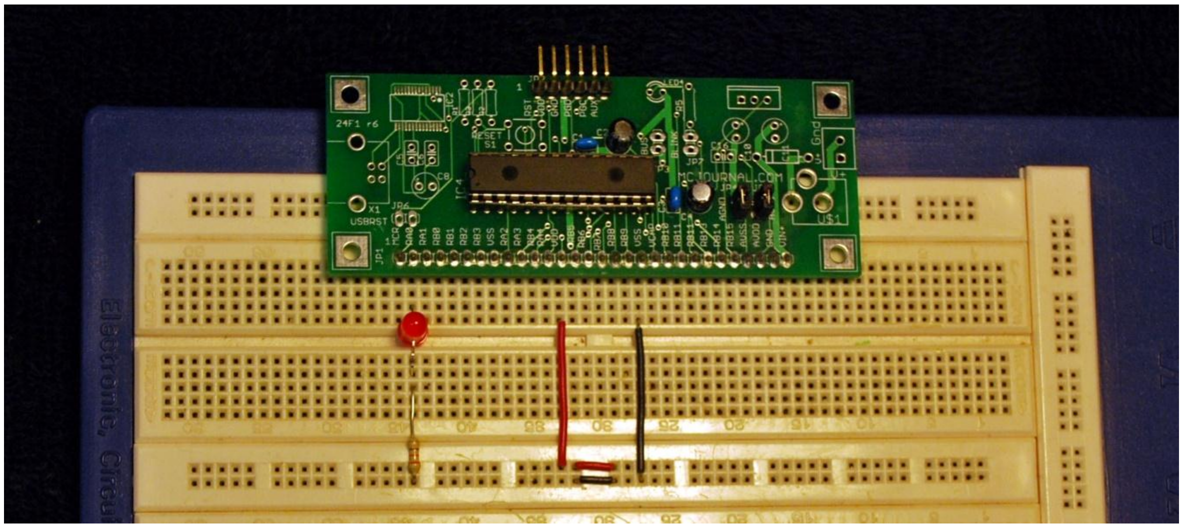
A number of projects will be presented, which are based upon using a low cost PIC24F/H board as the main platform, Microchip's free C30 C compiler, and a low cost PICKIT2 (or PICKIT3) used for programming and debugging function.

Project 1

Hello World – Turning On a LED

Introduction

This article covers the traditional “Hello World” program that turns on a LED connected to one of the I/O pins of a PIC24FJ32GA002 microcontroller. The article provides the interface diagram, as well as project files, containing ‘C30’ code.



The LED Interface

The LED is connected to RA0 pin (pin#2 on the PIC24F board). A 330 Ω resistor is used as a series dropping resistor to limit the current. Power to the circuit board is drawn from the USB interface, which supplies 3.3 V to PIC24F. The current drawn through the LED is given by the following expression

LED_Current x R + LED_Vdrop = 3.3 V

Assuming a LED drop of 1.5 V,

LED_Current = $\frac{3.3-1.5\ V}{330\ Ohm}$ = 5.5 ma OK

If a higher brightness is desired, a smaller resistor value can be used.

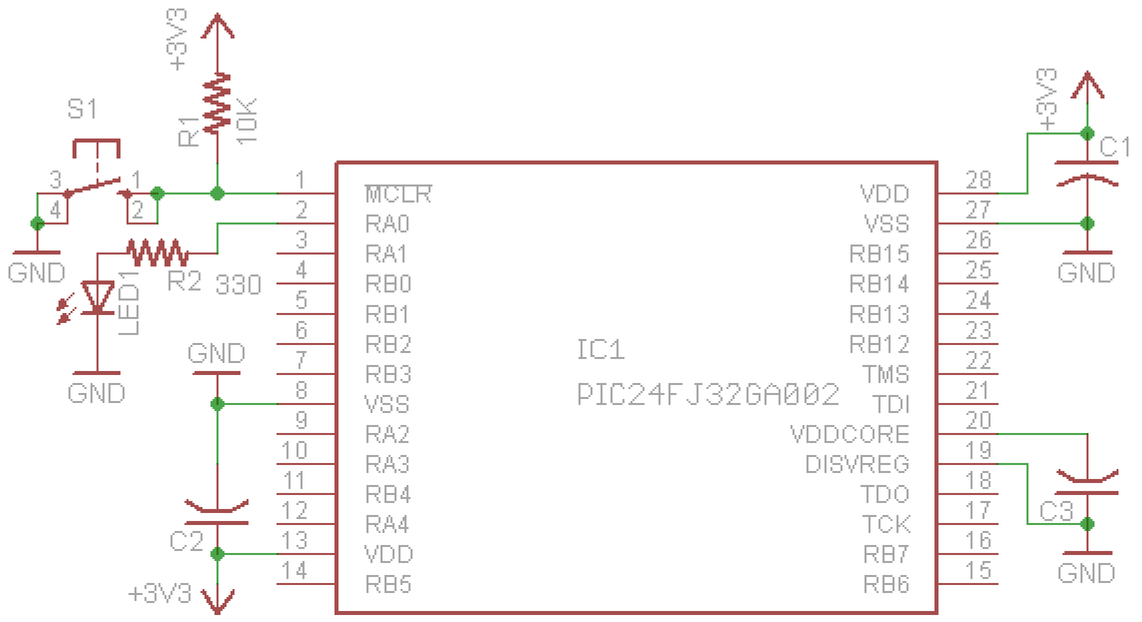


Figure 1 Schematic Diagram - Hello World

Operation

(See the code below). The code starts by first configuring the chip. There are two configuration registers CW1 and CW2, which are present in Flash program memory. These registers control the physical characteristics of the microcontroller unit (MCU). . Once programmed, these register values are retained till the next power-on-rest (POR) operation. These registers are outside the user program memory space and can only be accessed through Table read and write instructions. C30 along with MPLAB provides convenient _Config statements that allow the configuration to be set after a POR.

CW1

This register controls the JTAG interface, Watch Dog Timer (WDT), WDT prescaler (if WDT is enabled), and ICD interface.

FWDTEN: This bit enables (or disables) the watch dog timer and can only be programmed can only be modified after the next POR. The bit is set by default, which enables the WDT. For this development mode the WDT has been disabled through 'FWDTEN_OFF' statement in _CONFIG1.

DEBUG: This bit enables (or disables) Debug mode. If set, the device enters the debug mode after a POR and if cleared, the device enters operational mode without getting into Debug. The bit is also set by default, which enables the debug mode. For this development mode the WDT has been disabled through 'BKBUG_ON' statement in _CONFIG1.

ICS1:ICS0: These two bits control the pins, which are used for ICD interface (EMUC: EMUD and PGC:PGD). PIC24FJ32GA002 has 3 possibilities for ICD interface (see pin diagram). These are:

ICS1:ICS0 = 1:z Use PGC1: PGD1 – ICD interface available on RB1 and RB0
ICS1:ICS0 = 1:0 Use PGC2: PGD2 – ICD interface on RB10 and RB11
ICS1:ICS0 = 0:1 Use PGC3: PGD3 – ICD interface on RB6 and RB5

The PIC24F board is set to use PGC1:PGD1 interface is used. It is configured through 'ICS_PGx1' statement in _CONFIG1.

The Code

The code is shown below. The configuration bits are used to set the operating mode of the microcontroller.

```
//          ---------------------------------- Start of Code -------------------------------------------------
// SG
// April 17, 2011
// PIC24FJ32GA002
// PIC24FJ32_LED_1.c

#include <p24FJ32GA002.h>
// *************************************************************************
//          Configuration Bits
 // *************************************************************************/
_CONFIG1 (JTAGEN_OFF & BKBUG_ON & ICS_PGx1 & FWDTEN_OFF)
            // JTAG disabled
            // Background debug on
            // Communication Channel: PGC1/EMUC1 and PGD1/EMUD1
            // Watchdog Timer disabled

_CONFIG2 (FNOSC_FRC)
            // Select FRC Oscillator

// ===========================================================
//                                      int main (void)
// ===========================================================
int main (void)
{
            OSCTUN = 0;                 // Tune FRC oscillator, if FRC is used
            RCONbits.SWDTEN = 0;        // Disable Watch Dog Timer

            // Make AN0 (RA0) as digital others as analog
            AD1PCFG = 0b0000000000001;
```

```
        TRISAbits.TRISA0 = 0;         // Set RA0 as an output for LED
        LATAbits.LATA0 = 1;           // Turn on the LED

        while(1);                     // Stay here for now
        return (0);
}
//      ---------------------------------- End  of Code -----------------------------------------------
```

Code Dissection

The various sections of the codes are now briefly explained.

```
//      ---------------------------------- Start of Code -----------------------------------------------
// SG
// April 17, 2011
// PIC24FJ32GA002
// PIC24FJ32_LED_1.c
```

The above are the comments used to provide basic information. '//' style commenting is used here that indicates that all text that follows is to be ignored by compiler.

```
#include <p24FJ32GA002.h>
```

This (pre-processor) statement indicates to the compiler that register names and bits that are used in the program have been declared in the indicated header file (essentially a text file that can be opened and viewed, but should not be modified).

```
_CONFIG1 (JTAGEN_OFF & BKBUG_ON & ICS_PGx1 & FWDTEN_OFF)
        // JTAG disabled
        // Background debug on
        // Communication Channel: PGC1/EMUC1 and PGD1/EMUD1
        // Watchdog Timer disabled

_CONFIG2 (FNOSC_FRC)
        // Select FRC Oscillator
```

These statements are required to configure the operation of microcontroller. The configuration bits are located in a reserved area of Flash memory. Some of the critical aspects of the operation that are configured here are:

- Select internal FRC oscillator
- Turn off the JTAG interface available on PIC24
- Enable Debugger interface
- Set the debugger interface on pins corresponding to PGC1 and PGD1
- Disable the Watchdog Timer

Refer to PIC24FJ32GA002 data sheet at www.microchip.com for additional details.

```
int main (void)
```

The main function declared as return an integer (common format).

```
OSCTUN = 0;                     // Tune FRC oscillator, if FRC is used
```

Use factory calibrated FRC oscillator setting

```
RCONbits.SWDTEN = 0;            // Disable Watch Dog Timer
```

The Watchdog timer can be enabled in software as well. This statement ensures that software Watchdog timer is not enabled

```
AD1PCFG = 0b0000000000000001;
```

By default, the microcontroller pins that can be used for analog input are configured as analog pins and digital Input/output is disabled. This statement makes RA0 to be used for digital Input/output and the remaining pins are left for analog inputs (default).

```
TRISAbits.TRISA0 = 0;           // Set RA0 as an output for LED
```

Once the RA0 pin is set for digital Input/output, its direction for actual usage needs to be set. In this case the pin is used for controlling the voltage to the LED, so it is configured as an output.

```
LATAbits.LATA0 = 1;             // Turn on the LED
```

This statement simply set the voltage level on RA0 pinto high, enabling the LED to be turned on.

```
while(1);
```

As the program runs in a linear fashion (from top to bottom), eventually it will reach the terminating statement in main (), which will terminate the user program (main() function). C compiler places an indefinite loop in the program so that on termination the main is called again, essentially causing the program to reset. This can be troublesome and it a 'while (1);' statement prevents the termination of the program and prevent unnecessary program reset..

```
return (0);
```

The terminating statement in main () function indicates that a '0' is being returned (common in computer C programs) and program terminated normally.

Reference:

PIC24FJ64GA004 Family Data Sheet www.microchip.com