ECSESS RoboElectronics

Fundamentals of C programming and microcontrollers

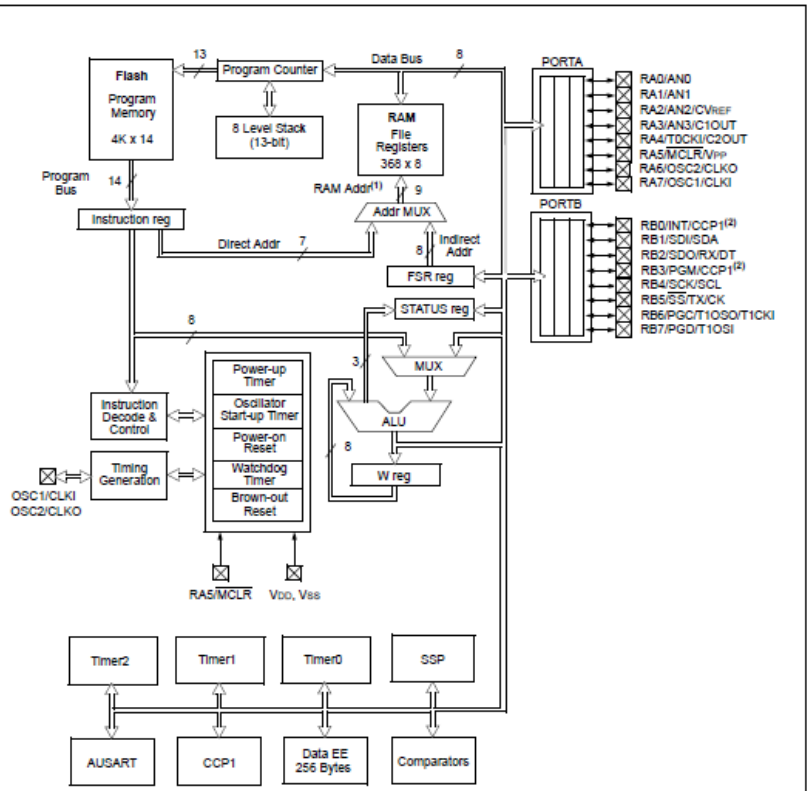
Topics Covered

- Microcontrollers and their uses
- Programmers and In Circuit Serial Programming (ICSP)
- Fundamentals of C programming

What is a Micro Controller

- A microcontroller is a small affordable controller consisting of a processor, memory, I/O ports, and special peripherals
- Often found in embedded systems, these controllers are highly flexible and come in a variety of shapes and sizes
- Can be low power devices
- Unlike PC's, they often don't have an operating system and have special peripherals to interface with the physical world

FIGURE 1-1: PIC16F87 DEVICE BLOCK DIAGRAM

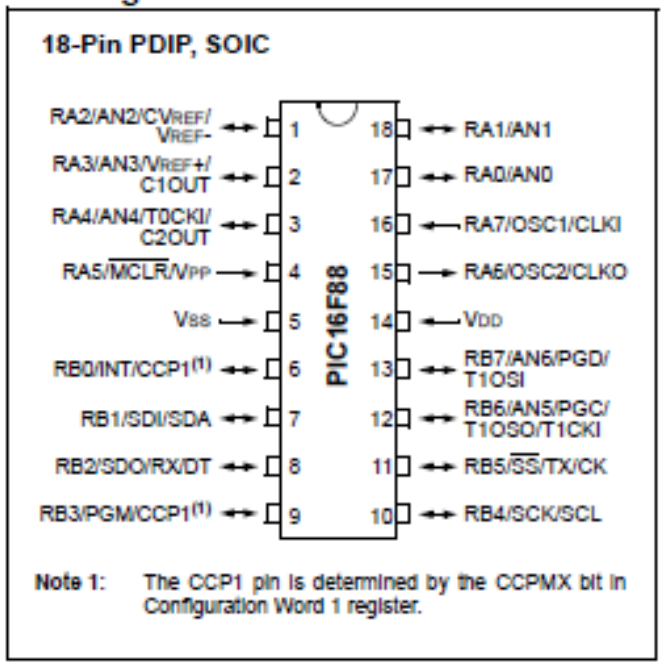


Note 1: Higher order bits are from the STATUS register.
2: The CCP1 pin is determined by the CCPMX bit in Configuration Word 1 register.

The PIC16F88 Microcontroller

- 8-bit microcontroller manufactured by **Microchip**
- A typical operating speed of 4 MHz
- Has EEProm, SRAM, and Enhanced Flash memory
- 16-I/O Ports and several other special hardware peripherals

Pin Diagram

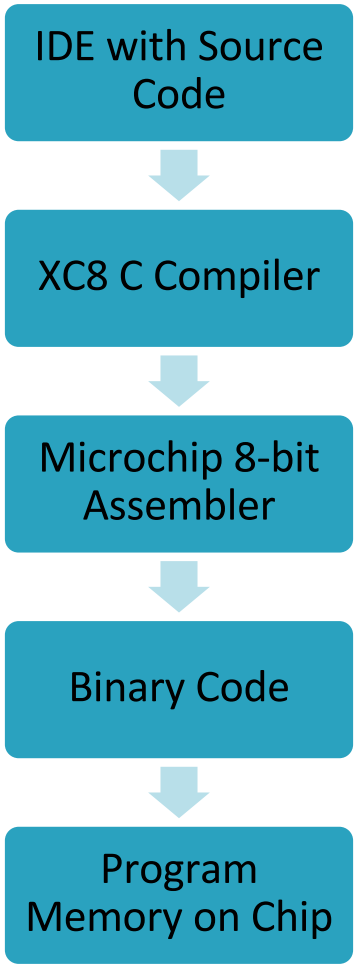


Note 1: The CCP1 pin is determined by the CCPMX bit in Configuration Word 1 register.

Device	Program Memory		Data Memory		I/O Pins	10-bit A/D (ch)	CCP (PWM)	AUSART	Comparators	SSP	Timers 8/16-bit
	Flash (bytes)	# Single-Word Instructions	SRAM (bytes)	EEPROM (bytes)							
PIC16F87	7168	4096	368	256	16	N/A	1	Y	2	Y	2/1
PIC16F88	7168	4096	368	256	16	1	1	Y	2	Y	2/1

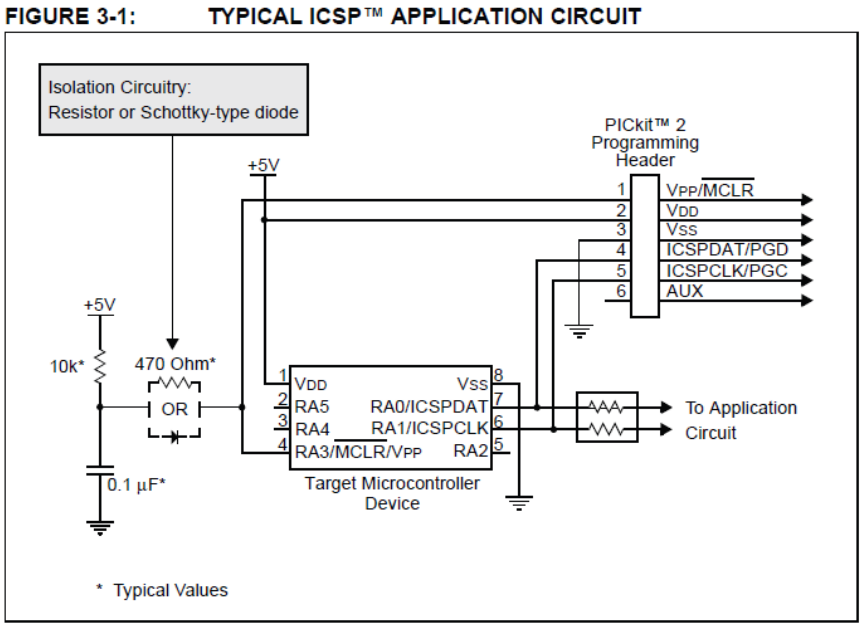
C tool chain for PIC16F88 controller

- The PIC16F88 will be programmed in C
- In order to write to the special function registers of the controller, a specialized C compiler is used: xc8
- A programmer, the PicKit3, is then used to load the assembled binary code onto the chip



Programming a Micro controller

- Once a program is assembled, it must be loaded into the controller
- A programmer is used to send the binary code using in circuit serial programming, ICSP
- ICSP uses three wires:
  - VPP, applies a programming voltage to the chip
  - PGD, the data line used to send the program
  - PGC, the program clock used to latch the data

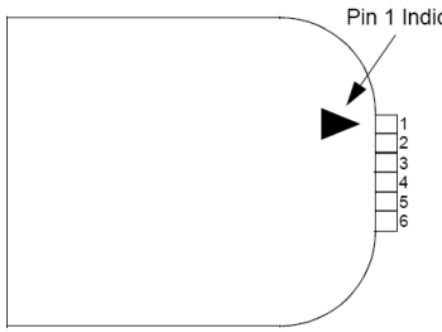


Connection diagram for ICSP with a generic controller

Pickit3 Programmer

- We use the Pickit3 programmer along with the MPLAB IDE to program the 16F88 chip
- The programmer is also used to power the circuit for debugging and testing purposes
- MPLAB is able to control the Pickit3 for us and automatically program the chip once the code has compiled

FIGURE 1-2: PICkit™ 2 PROGRAMMER CONNECTOR PINOUT



Note: The 6-pin header (0.100" spacing) accepts 0.025" square pins.

Pickit3 Pin	16F88 Pin
1, VPP	4, VPP
2, VDD	14, VDD
3, VSS	5, VSS
4, PGD	13, PGD
5, PGC	12, PGC

C Programming

- C is a low level language that can be found almost anywhere
- Unlike Java, CPP, and Python, C is not an object oriented (OO) language
  - **Pros:** Very powerful in terms of tweaking performance and knowing what code gets compiled to
  - **Cons:** Requires the programmer (you) to handle tedious things such as pointers, memory, garbage collection, etc
- Why we use C for micro controllers
  - Often our goal is to control specific hardware, such as I/O ports to light a LED, drive a motor, or read a value
  - C is compiled directly to an assembly language which is then turned into byte code and loaded on the micro controller
  - C allows code to be kept light and optimized for the small and simple controllers we are using

C Concepts: Conditional statements (If/else)

- If statements allow us to react to different conditions
- You can compare values using the following equalities
  - <, >, <=, >=, ==, !=
  - The example to the right checks if value3 is equal to 3, and changes value1 or value2 accordingly

```
//Declare vars
int value1 = 1;
int value2 = 2;
int value3 = value1 + value2;


if(value3 == 3)
{
    value1= 2;
}
else
{
    value2 = 1;
}
```

C Concepts: Loops (for/while)

- A loop allows us to repeat an action based on a value or condition
- The while statement will execute the code inside its curly braces provided the condition inside its brackets is true
  - The while loop on the right will execute 5 times before it exits
- The for loop is similar to the while, only you can specify a start value, condition, and operation in one line
  - The for loop sets the counter to 5, checks if it has reached zero, and increments the counter each time it loops around

```
//Declare vars
int i= 0; //Counting variable

while(i < 5)
{
    i = i + 1;
    //Do something
    .
    .
    .

}
for(i = 5; i > 0; i++)
{
    //Do something
}
```

C Concepts: Structuring code

```
#include <xc.h>

//Defines
#define INPUT 1
#define OUTPUT 0
#define ON 1
#define OFF 0

void main(void)
{
     init();  //Run an initialization function

     //Loop forever
     while(1)
     {
          PORTAbits.RA3 = ON; //Turn a LED On
          __delay_ms(500);

          PORTAbits.RA3 = OFF; //Turn a LE
          __delay_ms(500);
     }
}
```

Includes, brings in definitions of functions and variables from another file. <xc.h> has all the SFR defines

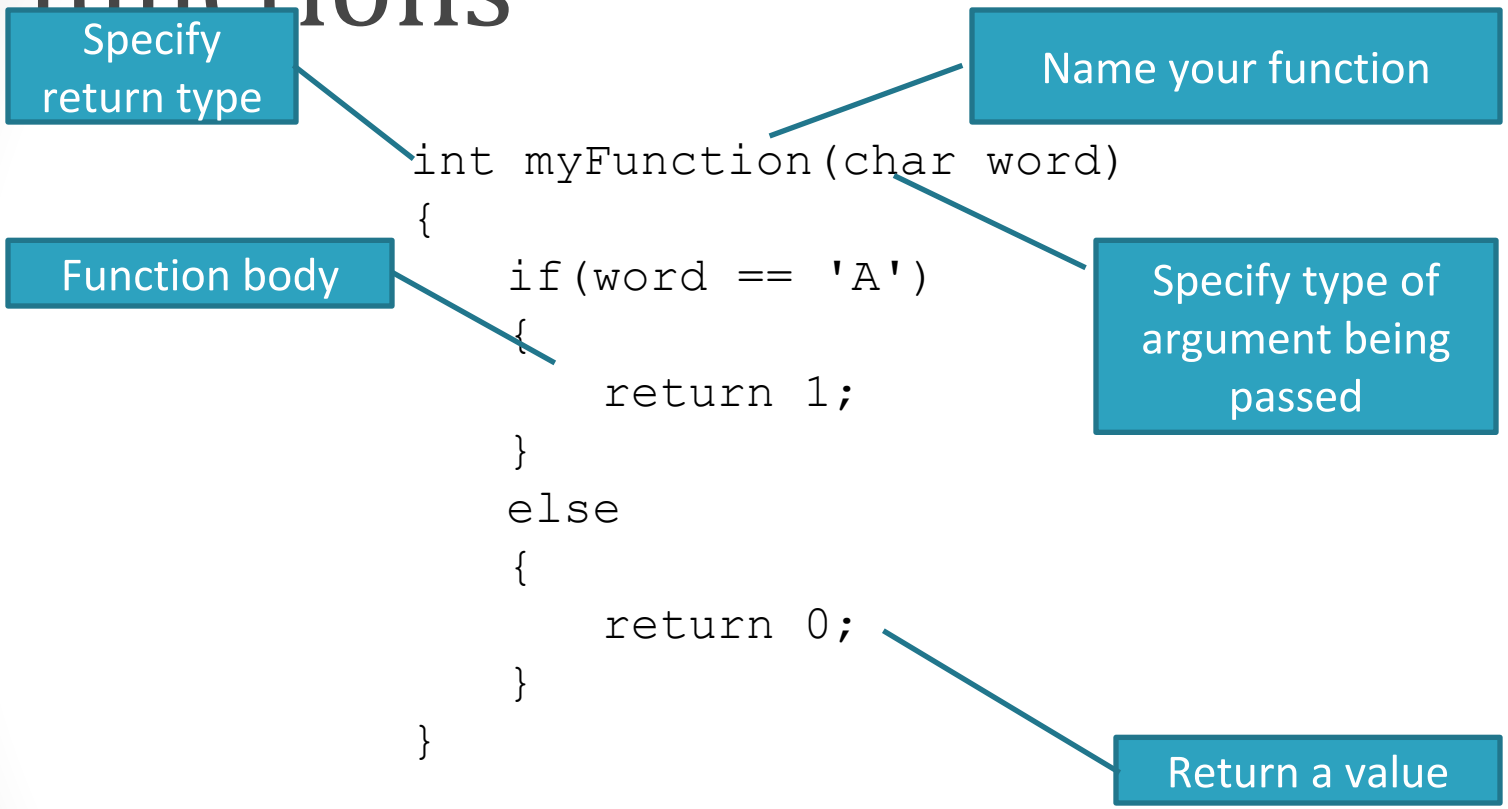
Defines, a simple text replace before compilation

The main function, when the controller is powered, execution begins here

Typically, our code should run forever, since the program running our robot should never end. This is done with an infinite loop

Setting a value to an I/O port defined by <xc.h>. This will put a logic high on that pin

C Concepts: Defining functions



Defining and Using Functions

```
int myFunction(char word);

void main(void)
{
    ar character = 'A';
    t value;

    value = myFunction(character);
}

int myFunction(char word)
{
    if(word == 'A')
    {
        return 1;
    }
    else
    {
        return 0;
    }
}
```

Functions must be **prototyped** in **C.** Line must end in a semi colon

Return value for your call

Pass in your value

Call your function

Your function can be filled out here

Header and Source files

- Often source files, those with a .c extension, have an associated header file, .h
- Header files are included at the top of your main file so that your program can call functions found in these files.
- Also, any defines or global variables should also be listed in the header file.
- Often, the source .c file is unavailable to us as programmers, and we can only see the .h

```
/*@file adc.h */

#define ADC_CHANNEL 0

int readADC(char channel);
```

```
/*@file main.c */
#include "adc.h"
.
.
.
void main(void)
{
    int value;

    value = readADC(ADC_CHANNEL);
}
```

Useful built in functions & defines

- There exist a number of useful built in functions and defines that you can use with the micro controller
- `__delay_ms(int value);` A function that delays for *value* milliseconds, note there are two under scores at the front
- `TRISXbits.TRISX#` : A define for an I/O ports tristate register to set a port as an input or output, where X is either A or B and # is a value ranging from 0 to 7
  - Eg. `TRISAbits.TRISA3 = 0; //sets pin RA3 as an OUTPUT.`
- `PORTXbits.RX#` : A define for an I/O port where X is either A or B and # is a value ranging from 0 to 7
  - Eg. `PORTAbits.RA3 = 1; //sets pin RA3 to HIGH.`

Resources

- Software
  - XC8 Compiler: http://www.microchip.com/mplabxc8windows
  - MPLAB IDE: http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&dDocName=en019469&part=SW007002
- Datasheets
  - PIC16F88: http://ww1.microchip.com/downloads/en/DeviceDoc/30487D.pdf
  - PIC16F88 XC8 header file (on your local machine): C:\Program Files (x86)\Microchip\xc8\v1.12\include\pic16f88.h
- Further Reading
  - Pickit3 Programming Guide: http://ww1.microchip.com/downloads/en/DeviceDoc/51553E.pdf
  - C tutorial: http://www.learn-c.org/