

MPLAB IDE Introduction and Using External Oscillator

Tom Aprik

Team 9: Safety Enhancement Features for the 21st Century Automobile
ECE 480: Capstone Design
Michigan State University

November 14, 2008

Contents

1	Executive Summary	1
2	Objective	1
3	Development	1
3.1	Programming Environment	1
3.2	Using an External Oscillator	2
4	Issues	5
5	Conclusion	5

1 Executive Summary

Programmable Interface Controllers (PICs) are a very powerful and low-cost microcontrollers. They have many uses and can be programmed to do many useful things, from simple LED displaying to matched filtering algorithms. Alongside of the PIC, it is also useful to have an external oscillator to achieve higher frequencies with just using the internal PIC clock. There are two main programming languages that allow programming the microcontroller, assembly and C. This application note will demonstrate how to program a programming environment for the microcontroller and enable an external oscillator.

2 Objective

This application note will display MPLAB IDE and how it is used. It will also explain how to interface the PIC with an external oscillator using the PIC's phase locked loop (PLL).

3 Development

3.1 Programming Environment

PIC is a family of Harvard architecture microcontrollers solely made by Microchip Technology. Microchip also makes the programming environment, MPLAB IDE in which all the programming, compiling and linking are done. C is a powerful programming language that enables the use of certain libraries to be used to help program functions for a specific need. All the code that is to be written will be done inside of MPLAB IDE. Also, the MPLAB will compile this code - turn C code into something the PIC can understand - and program it to the PIC. The figure below shows the basic look to a project in MPLAB. On the left shows all the files that are included in the project. The center window is the code file, where code

would be written for compiling. On the right shows the output from the compiler, the linker, and eventual output from uploading the program to the PIC. Inside all of these windows is where the majority of the coding will occur.

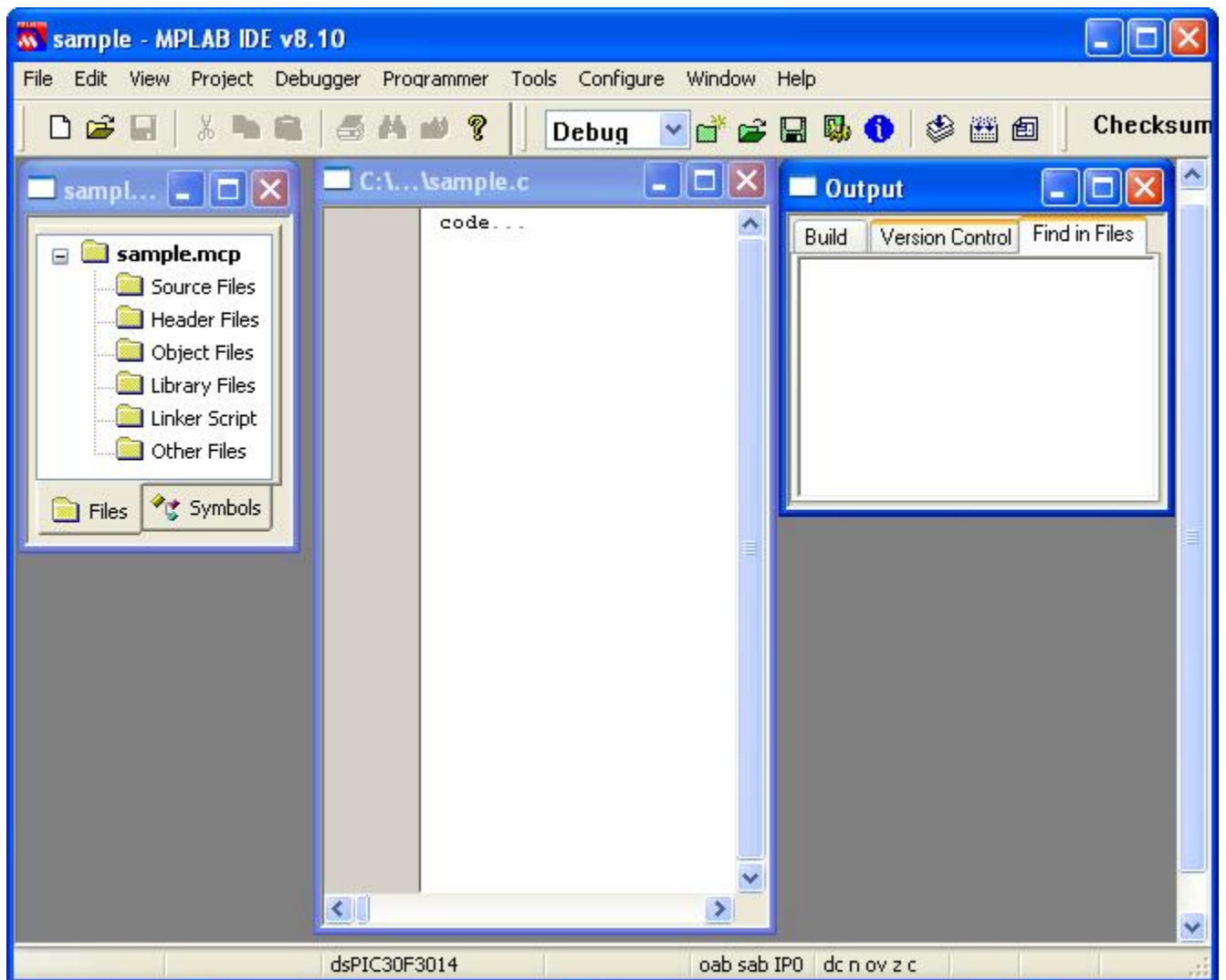


Figure 1: MPLAB IDE

3.2 Using an External Oscillator

The internal oscillators of PICs have a very large range. The less expensive ones have a lot lower frequency, but most are able to be clocked higher using an external oscillator. There

are many benefits to using an external oscillator rather than having a faster internal clock. It will reduce the cost of the PIC itself having a slower oscillator and also using the oscillator externally will let the PIC run less hot. PICs with an OSC1 input, the oscillator crystal input, are able to take advantage of using an external oscillator. The input to the PIC from the output of an external clock goes into the OSC1 input. The input is then routed to the PLL multiplier which will increase the frequency of the external clock. After the PLL multiplier, the signal is then routed to the clock switching and control block where all the timing is done.

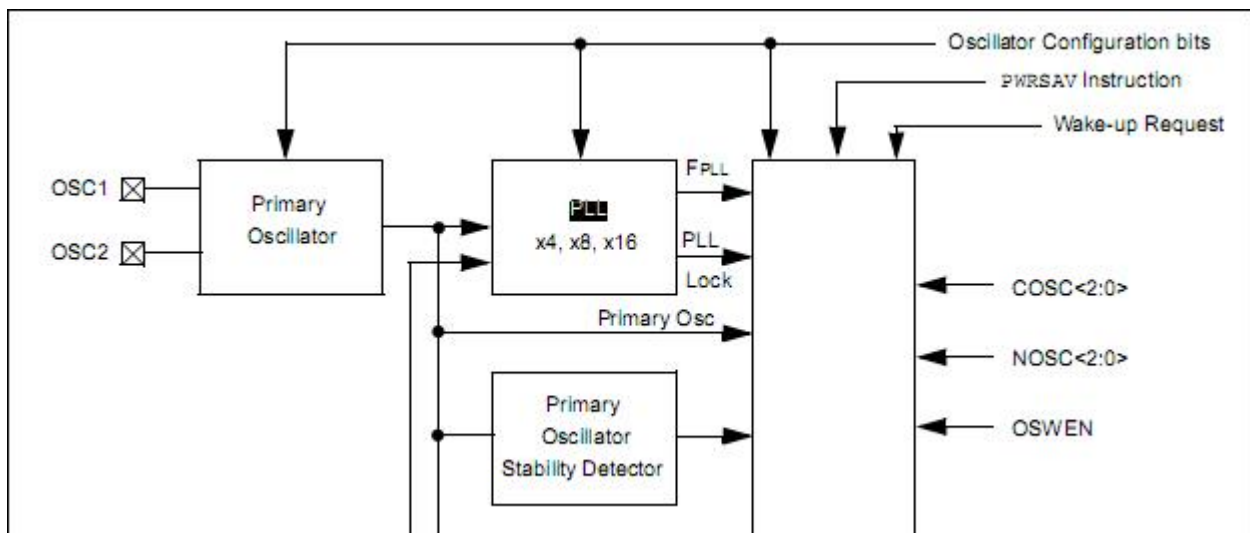


Figure 2: Oscillator System Block Diagram

The figure above shows the diagram in which this all takes place. However, there needs to be code for the PIC to know to use these features. The PIC will know to use these features through the FOSC register, the oscillator configuration register. This register is a 24-bit register, with bits 23 through 16 always being zeros.

REGISTER 20-3: FOSC: OSCILLATOR CONFIGURATION REGISTER

Upper Byte:							
U	U	U	U	U	U	U	U
—	—	—	—	—	—	—	—
bit 23				bit 16			

Middle Byte:							
R/P	R/P	U	U	U	R/P	R/P	R/P
FCKSM<1:0>		—	—	—	FOS<2:0>		
bit 15				bit 8			

Lower Byte:							
U	U	U	R/P	R/P	R/P	R/P	R/P
—	—	—	FPR<4:0>				
bit 7			bit 0				

Figure 3: FOSC: Oscillator Configuration Register

There are many ways to configure this register for using the 4x, 8x, and 16x multiplier that Microchip Technology has taken their time to create macros for the programmer to easily enable certain things in the register instead of having to write the binary, in return making the code less messy. Below is some code that will turn off the Watch-Dog Timer. The Watch-Dog Timer which is essentially a failsafe causing the microprocessor to reset if the main program hangs or has some other type of fault. We turn this off for debugging purposes. The next line is where we write to the FOSC register using the built-in macros. What the second line is doing is disabling the clock switching and failsafe clock monitoring, again for debugging purposes. Lastly, the important part is enabling the external clock using the PLL and setting it to a 16 multiplier.

```
{
  _FWDTC(WDT_OFF);           // Turn off the Watch-Dog Timer.
  _FOSC(CSW_FSCM_OFF & ECIO_PLL16); // Disables failsafes and enable PLL x16.
}
```

4 Issues

When using an external clock with a PIC, you always have to carefully calculate the clock speed that is desirable and match it against the PIC. If you are generating too high of a frequency, then you can potentially damage the PIC. You have to consider the multiplier you are using with the PLL as well as the frequency of the external oscillator itself. It is always best to test under lower frequencies until everything is stable.

5 Conclusion

This application note demonstrated the MPLAB IDE and how the basics of an external oscillator. External oscillators are an integral part of PICs, in that they allow for much higher frequencies of just the internal oscillator.