How To Modify OPEN-ROBOT's PIC18F4520 Firmware

Introduction:

This document will explain the process of modifying OPEN-ROBOT's PIC18F4520 firmware. The PIC18F4520 firmware was created using the CCS C Compiler and also integrates a CCS bootloader. This bootloader allows for firmware upgrades without requiring the use of a PIC programmer unit. This document will focus on using the PCWH IDE version of the CCS C compiler.

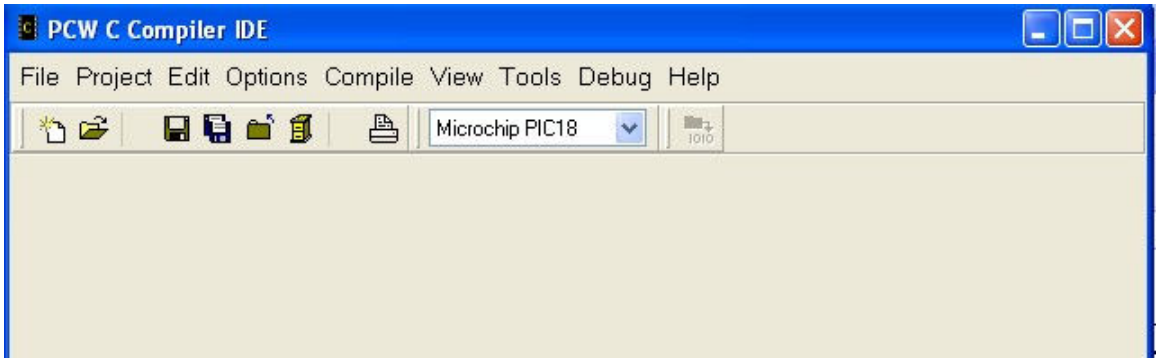
Requirements:

You will need to purchase or have access to a suitable CCS C compiler for the PIC18F4520. I recommend purchasing the PCHW IDE version of the compiler. However, you can use the PCH command line compiler and achieve the same results, but just without the ease of an IDE. You can check out the compilers by following the link below.

http://www.ccsinfo.com/content.php?page=ideoverview

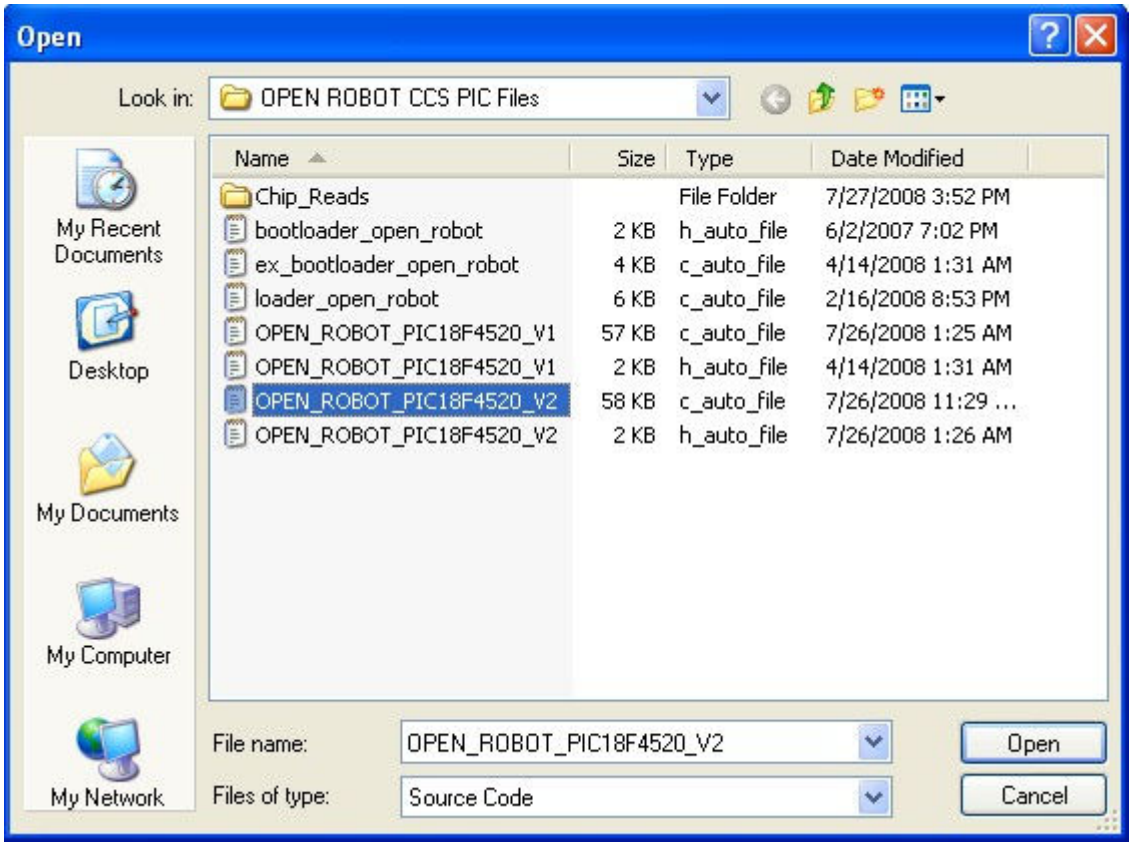
Download the latest version of the PIC18F4520 firmware from the OPEN-ROBOT website. After the download is complete, be sure to extract the zip files to a suitable location.

Getting Started with the PCWH Compiler:



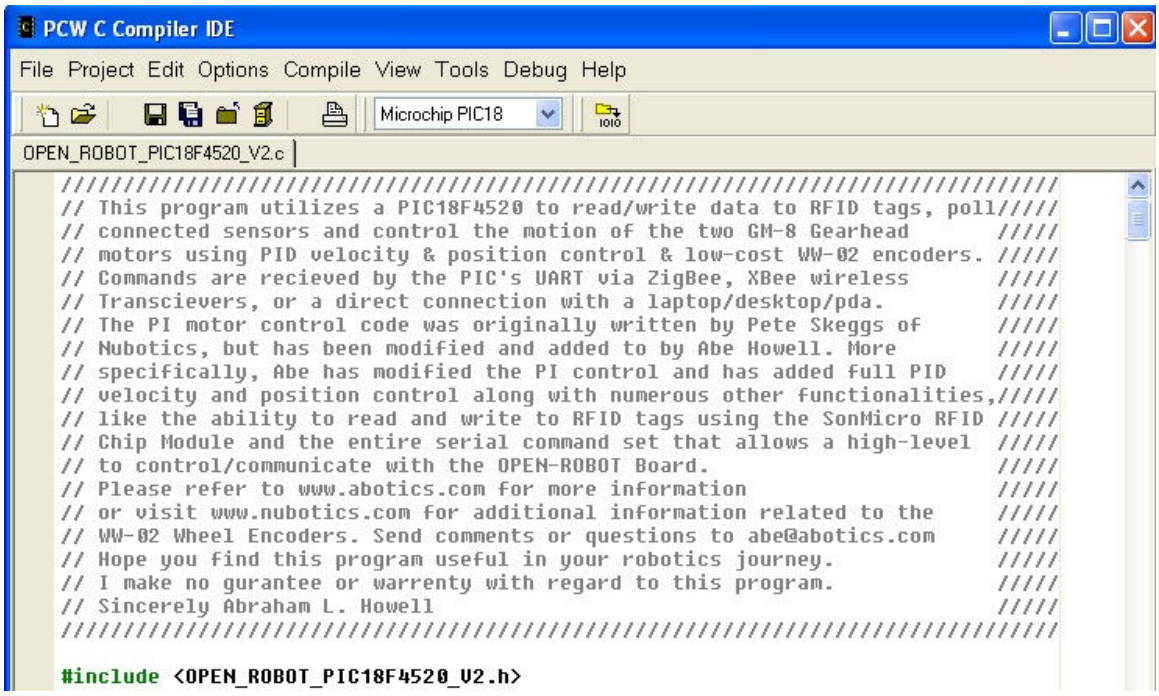
Figure#1. Run PCWH IDE.

Go ahead and run the PCWH IDE. Click Open from the File menu and browse to the location where you unzipped the PIC18F4520 firmware files. For this example the latest version of the firmware happens to be "OPEN_ROBOT_PIC18F4520_V2.c".



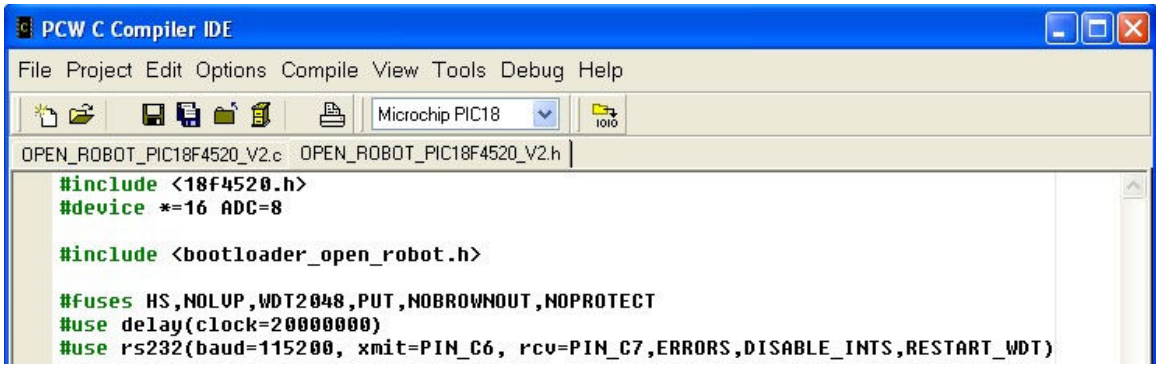
Figure#2. Open Firmware C-File.

Select to view the details of the listed files and select the c_auto_file.



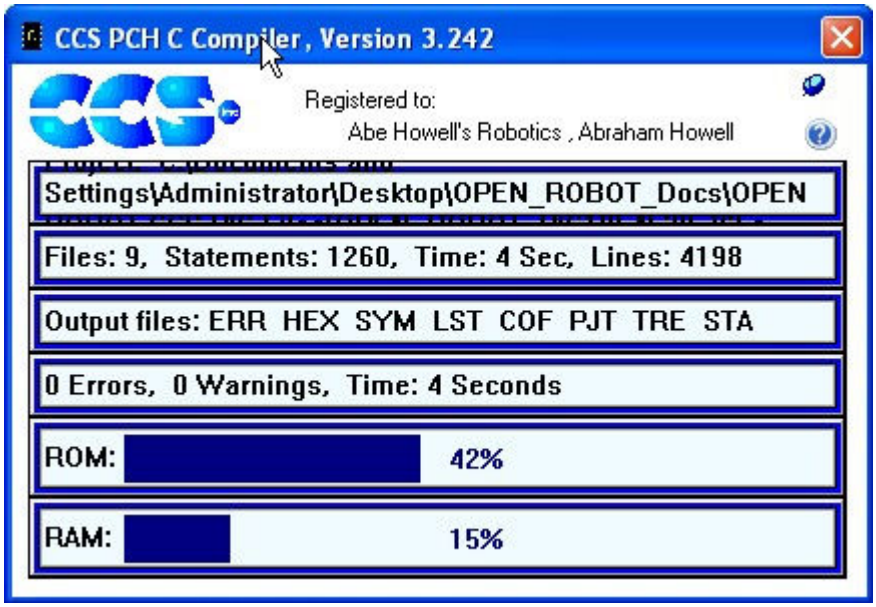
Figure#3. Source code displayed in IDE window.

After opening the source c-file you will see the source-code populated in the IDE window as shown above in figure#3.



Figure#4. Open header file.

Now open the corresponding h_auto_file or header file. The source code will be displayed on a second tab. You should notice that the only include file is as follows "bootloader_open_robot.h". This include is very important because it lets the compiler know that we are using the bootloader in the firmware.



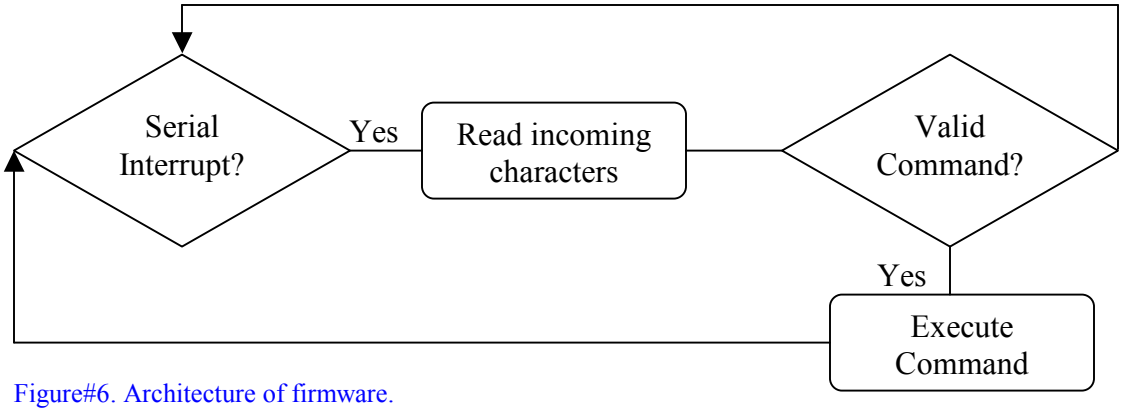
Figure#5. Compile.

Now we can go ahead and test to be sure that our program compiles successfully. Simply select "Compile" from the Compile menu or press the F9 key. Once the compile is complete you will be notified and the amount of ROM and RAM that has been allocated will be displayed in a pop-up window.

You can now begin modifying the firmware to suit your own needs or simply use this as a starting point for your own custom firmware creation.

Architecture of PIC18F4520 Firmware:

The basic architecture of the firmware is to wait for a serial interrupt to occur, read in the received characters, and once a "\r" char is received decipher the command and execute it.



Figure#6. Architecture of firmware.

The Timer2 interrupt service routine (ISR) is used to update the velocity and position control loops at a fixed frequency. Timer2 is also used to cycle through each of the analog-to-digital pins and retrieve the most current reading for the attached sensor and then store it in the appropriate variable. This ensures that the most up-to-date readings are transmitted when a user requests them.

The Timer0 ISR is used to capture the WW02 Wheel encoder clock signal transitions from the right encoder.

The external ISR for PIN_B0 is utilized to acquire the WW02 Wheel encoder clock signal transitions from the left encoder.

The above-listed hardware interrupts are the only four used in the firmware. Communication with the onboard RFID module is achieved by using a software UART. The remainder of the program is basically a collection of functions that support the Serial-Based firmware interaction with the outside world and provide an easy to use interface for controlling OPEN-ROBOT. For additional information, I recommend taking a look at the OPEN-ROBOT Control Board schematic and the PIC18F4520 I/O Allocation.

http://www.abotics.com/ExpressPCBs/OPEN_ROBOT_BOARD_SCHEMATIC.pdf

http://www.abotics.com/OPEN_ROBOT_DOCS/OPEN_ROBOT_PIC18F4520_IO.xls