INSTRUCTIONS FOR SETTING UP ROACH SOFTWARE IN MPLAB-X

These instructions are provided to help you get started on the first lab to get up and running in MPLAB-X using our Uno32 boards with the roaches. Note that if you want to set up your own environment on your laptop, you will need to make some changes (contact the TAs or Instructor for further instructions).

1. First select the MPLAB X IDE and open it:

![MPLAB X IDE](image1)

2. Click on create new project (the icon with the green + sign in the toolbar):
3. Make a new Standalone Project:

4. Under the families, choose PIC32, under the device, select PIC32MX320F128H, click next.

5. For hardware, select PICKit3, next.

6. For compiler, Select C32 (v2.01), next.
7. Give your project a name (such as Lab0_RoachTest) and finish.

![Image showing the 'New Project' window with the project name set to Lab0_RoachTest]

8. You should see your empty project up on the left hand side:

![Image showing the project explorer with Lab0_RoachTest selected]

9. Right click on Source Files -> New -> Other...
10. Choose Microchip Embedded -> C32 Compiler -> mainp32.c

Name it something useful (ie: main) and click finish. The file should load into the editor, and appear in the project window. Do the same to add any headers if need be (for lab0 you won’t need to).

11. Next, we are going to add in the linker files that make the project work with our bootloader. Right click on the Linker Files, and choose Add Existing Item...
12. Navigate to the C:\CMPE118 Directory, select the linker folder, and choose both files. **Very important**, make sure that the Store path as radio button is **Absolute**. Click select.

At this point you can compile the project with no errors (use the small hammer up on toolbar to compile. We’ll get to this later.

13. Now we’ll add the source files for the CMPE118 Roach libraries so that you can easily play with them. Right click on source files and again, Add Existing Item... and choose from the src folder: AD, pwm, roach, serial, and timers.c, again make sure the **Store path as** radio button is **Absolute**:
14. Lastly, we need to set the path to the header files in the compiler so it knows where to find them: Under the Run menu, select Set Project Configuration -> Customize...

15. Choose the C32 -> pic32-gcc, scroll down in the General menu and get to the Include directories, click on the ... button:
16. Click on Browse..., navigate to the C:/CMPE118 directory, and choose the include folder, and open.

![Open File Dialog](image)

17. OK -> OK to get back to the main project. Again, you should be able to now compile without error. Lastly, we need to add a few #include's to our main file, and a call to Roach_Init() which will get everything started.

18. In the main.c file, above the main function, type in:

```c
#include <serial.h>  // note that you can type in the first few letters, and MPLAB-X will auto-complete it
#include <roach.h>
#include <timers.h>
```

And inside main, call Roach_Init(). Here you can use Control-space to autocomplete and show you documentation of the function. You main.c file should now look something like the following image (this shows the documentation feature for the Roach_Init() function, and is available for most functions.)
19. You are now ready to begin playing with the roach. Take a look at roach.h to see what functions are already define for you, same with timers. For the serial port, use printf and getchar for communication to the USB port. Note that you can control-click on the header file to open it.

20. Make main look like this:

```c
int main(void) {
    Roach_Init();
    printf("\nHello World!\n");
    while (1) {
        printf("\nLight level: %d Bumpers: 0x%x", LightLevel(), ReadBumpers());
        while (!IsTransmitEmpty()); // bad, this is blocking code
    }
    return 0;
}
```
21. And now let’s compile by clicking on the small hammer, and load it onto a roach ( 🐜):

![Image of IDE with compiled code]

You have now managed to create a new project, have it do something useful, and set up the IDE so that everything matches the processor and environment we are going to use in the class. Note that you could have also selected the Simulator, rather than the PICkit3, which would allow you to run the simulator (useful for debugging code). We’ll cover that at a later date in the class; this is just to get you up and running.

The ChipKit Uno32 boards from Digilent we use are nominally Arduino compatible boards. However, we don’t use Arduino around here, rather, we program in straight C. To do this, we’ve reflashed these boards with a custom bootloader which will load what we compile in the IDE over the USB port.

22. To do this, we’ll launch our ds30Loader, double click on the desktop icon for ds30 Loader GUI, and plug in your roach usb into the white usb extender cable. You should see a red LED light up on the roach when plugged in. There are a few settings we’ll need to change on the ds30 to match what we need:
23. First, select your hex file by choosing the ... next the the Hex-file and navigating to your project directory, choose \dist -> default -> production and select the .hex file. Next change the device to PIC32MX from the pulldown menu, and it should auto-select 320F128H. Change the baud rate to 115200, and select the USB Serial Port (it will only show the active ones), and check the Write Flash box:

![Image of the ds30 Loader GUI with the advanced mode settings and USB Serial Port selected]

24. In the menu, choose View -> Advanced Mode, and select the Reset tab, make sure the radio button for DTR is selected, and Reset time[ms] is set to 1:

![Image of the ds30 Loader GUI with DTR selected and Reset time set to 1]

25. Lastly, in the terminal tab, make sure the baud rate is at 115,200 and the Switch after write box is checked (this will turn on the serial port after flashing):

![Image of the ds30 Loader GUI with the terminal tab settings and Switch after write box checked]
26. Verify that your setting are correct; check for the boot loader by clicking on the **Check for bl** button. You should get a response back:

Resetting device...ok  
Searching for bl  
Found PIC32MX320F128H fw ver. 5.0.2

27. Now you can program your roach by clicking on the **Write** button, and it will immediately switch to the serial window as soon as it finished flashing the device.

28. Modify your code, reflash the device by closing the serial port and hitting write once again.

29. Good luck and happy programming.