ECE315 – Tutorial
Introduction to NBEclipse, Mttty and the MCF 5234-based NetBurner Board.

Lab Dates, Report Dates and Demo Due Dates:

Please login to Moodle at eclass.srv.ualberta.ca to access all relevant lab dates. The lab schedule section contains the important dates for this semester.

Objectives:

- To introduce the NetBurner version of the Eclipse Integrated Development Environment (IDE) called NBEclipse.
- To introduce the use of Mttty for viewing serial output from the NetBurner board.
- To introduce software design for an embedded real-time operating system.
- To introduce basic hardware interfacing to the on-board hardware.
- To introduce the use of an oscilloscope for basic hardware debugging.
- To review introductory concepts in memory management for the NetBurner board.

Equipment, Parts, and Provided Software:

Equipment

- DB9 cable connected between the computer's COM1 serial port and the MOD-100 DEV board serial port (connector J8)
- MOD-100 DEV board with power block
- MOD5234 - Microcontroller daughter card
- 50 - pin ribbon cable attached between the J2 header and the students' breadboard
- Crossover Ethernet cable connected to the MOD 5234 daughter-card and the expansion slot Ethernet port on the host computer
- Three channel Instek power supply
- Oscilloscope
- Digital Multi-meter (DMM)

Provided Software

- Windows 7 - PC host operating system
- NBEclipse - Integrated development environment on the host
- Mttty - Serial communication utility on the host
- Notepad++ - ASCII file editor on the host
- MicroC/OS operating system on the NetBurner board

Documentation:

Please login to the Moodle lab pages at eclass.srv.ualberta.ca to see a list of necessary documents about the Coldfire-based board used in ECE315. The documents are listed under the Lab Reference Materials heading.

The **Carrier Board Schematic** provides the layout of the J2 header on the carrier board. This schematic will help you find the location of the J2[41] and J2[44] pins.

The NetBurner Runtime Libraries Manual provides reference material for the NetBurner-provided libraries. For example, DHCP, HTTPD, the IO routines, the serial subroutines, and most of the networking subroutines are listed here.

The **ioboard.cpp** file is provided as a reference to the routines that control the DIPS, LEDS, and seven segment components on the board.

Introduction:

NBEclipse

The IDE used in ECE315 is called NBEclipse and is provided by NetBurner with each NetBurner board purchased. NBEclipse is a customized version of the Eclipse IDE specifically targeted at our Coldfire MCF 5234-based boards. It includes a rudimentary editor, a debugger, and many other tools that automate the process of developing source code; compiling, linking, downloading and writing object code to FLASH memory on the board, and running that code.

Mttty

The NetBurner boards come with a NetBurner-provided monitor in FLASH memory. This monitor provides features that allow object code to be downloaded and written to the board and then executed. It also has some limited debugging aids like displaying the contents of memory locations and CPU registers. A serial

communications application is used to send commands to the monitor. We'll be using Mttty for viewing the monitor commands and stdio output.

Provided Tutorial Code

The code provided for this tutorial displays a running pattern that lights up one LED in succession on the eight LEDs on the board. A fixed pattern on the seven segment is also displayed. The software running on the board has also been configured to generate a toggling signal at every tick (i.e. heartbeat) of the operating system timer as well as a toggling signal on task context switches. The heartbeat signal is available on pin J2[44] and the context switch signal on pin J2[41]. Hooking the oscilloscope up to these pins and verifying that your board and OS are happy and healthy should be a standard initial debugging technique when things go wrong.

Tutorial Coding Assignment

In this tutorial, students will add a small amount of code to read the DIP switches and turn on or off the corresponding LEDs. No external hardware is required to complete this tutorial. The DIP switches are numbered 1 through 8 and the LEDS are named LED1 to LED8.

To complete the I/O coding assignment, the existing LED task will need to be modified. Read the comments in the provided source code to see where you should put your modifications.

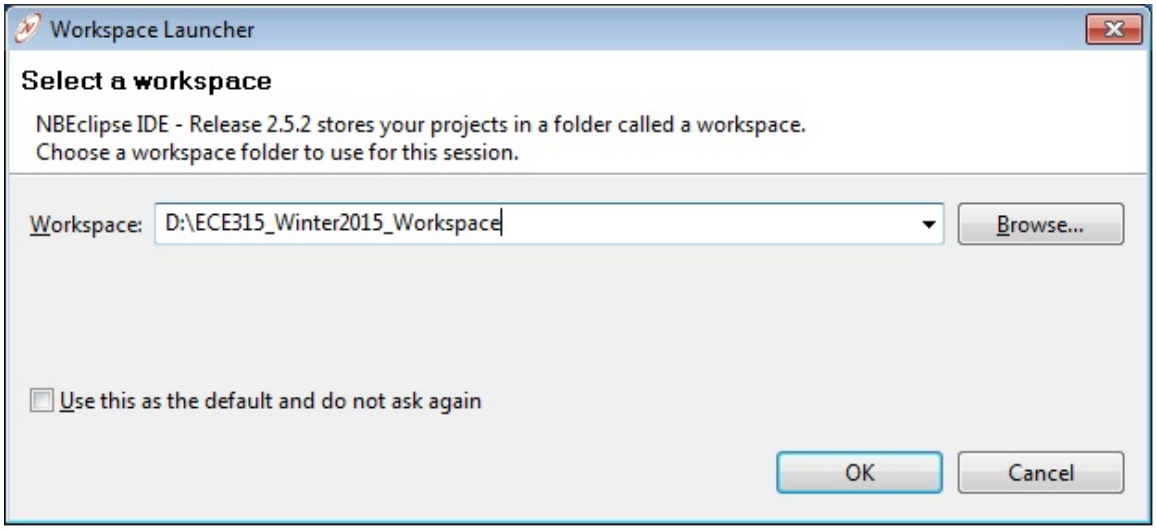
To complete the memory management coding exercise, you'll need to modify the Seven Segment task to display an incrementing pattern on the seven segment displays. The code given to you does not do the incrementing properly. It is your job to correct it.

Instructions:

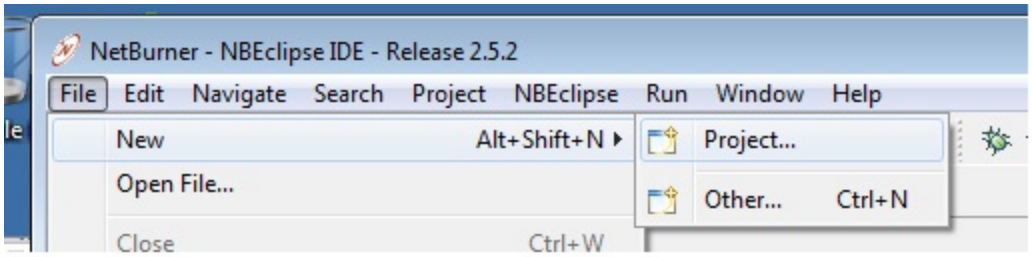
Create a New Default Project and Import the Provided Tutorial Code

- Start by creating a workspace for NBEclipse to use by clicking on Start->Computer. Create a new folder on the scratch partition D: and name it ECE315_Winter2015_Workspace.

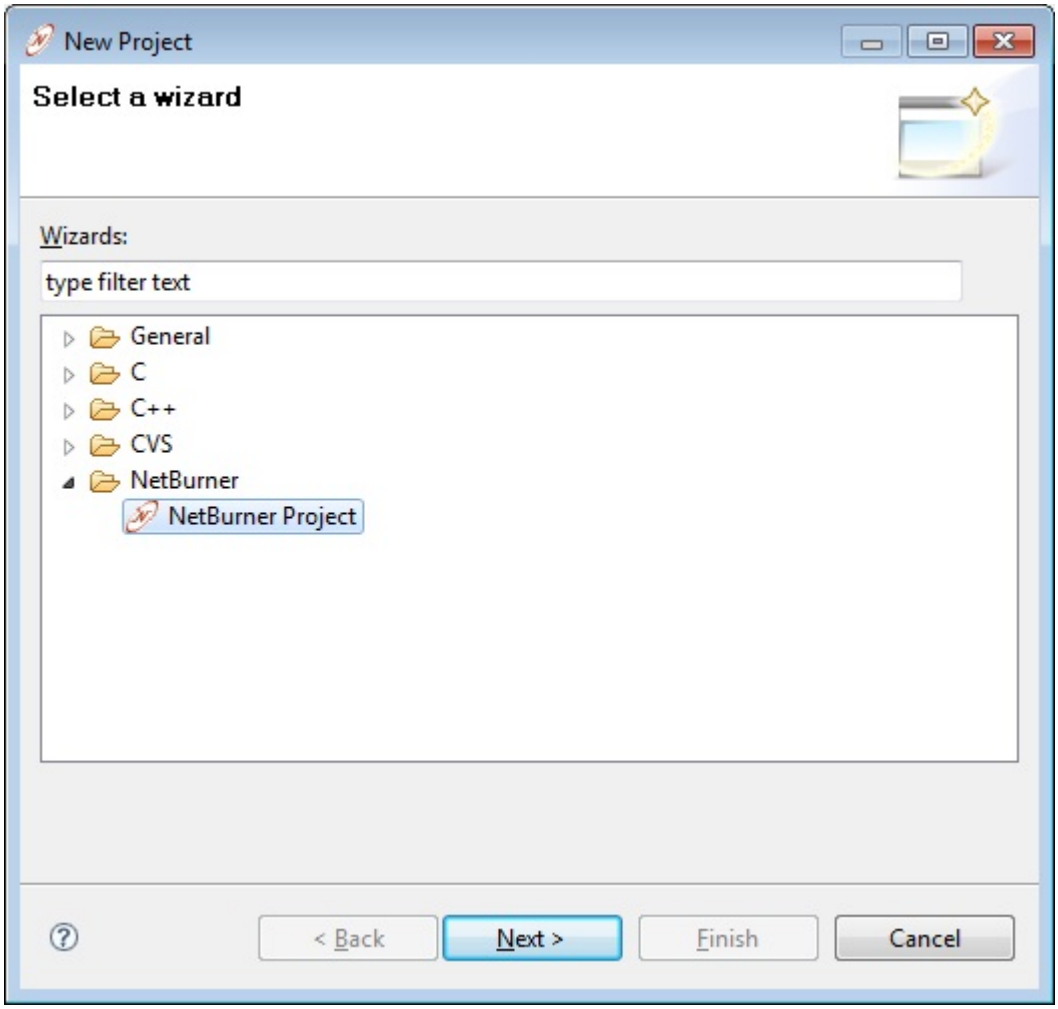
- Open NBEclipse by clicking Start->All Programs->NetBurner NNDK->NBEclipse->NBEclipse on the host PC. If NBEclipse prompts you to select a new workspace, browse to your newly created ECE315_Winter2015_Workspace folder and click on OK.



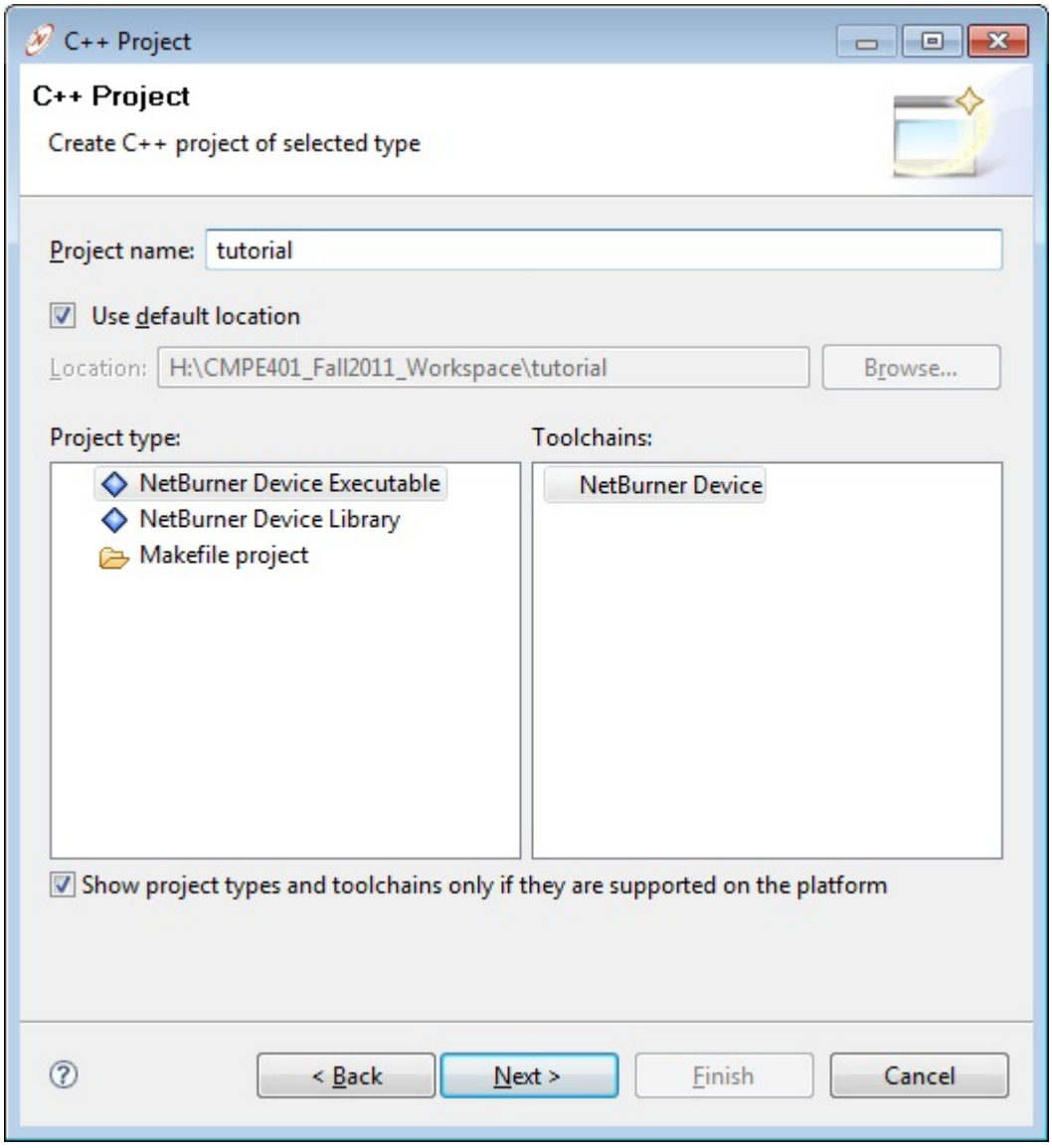
- Create a new project with default files.



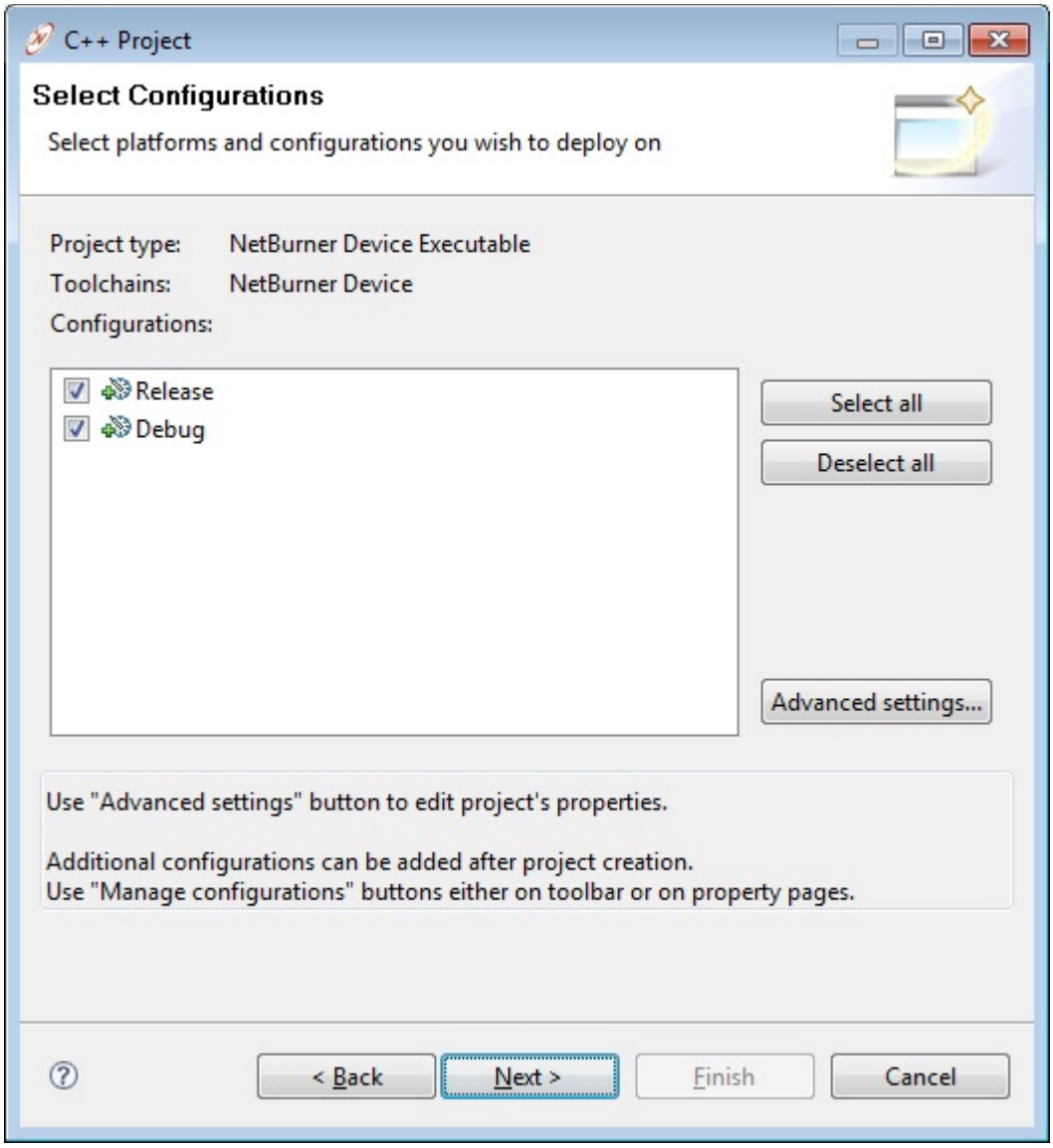
- Expand the NetBurner folder and click on NetBurner Project. Click on Next.



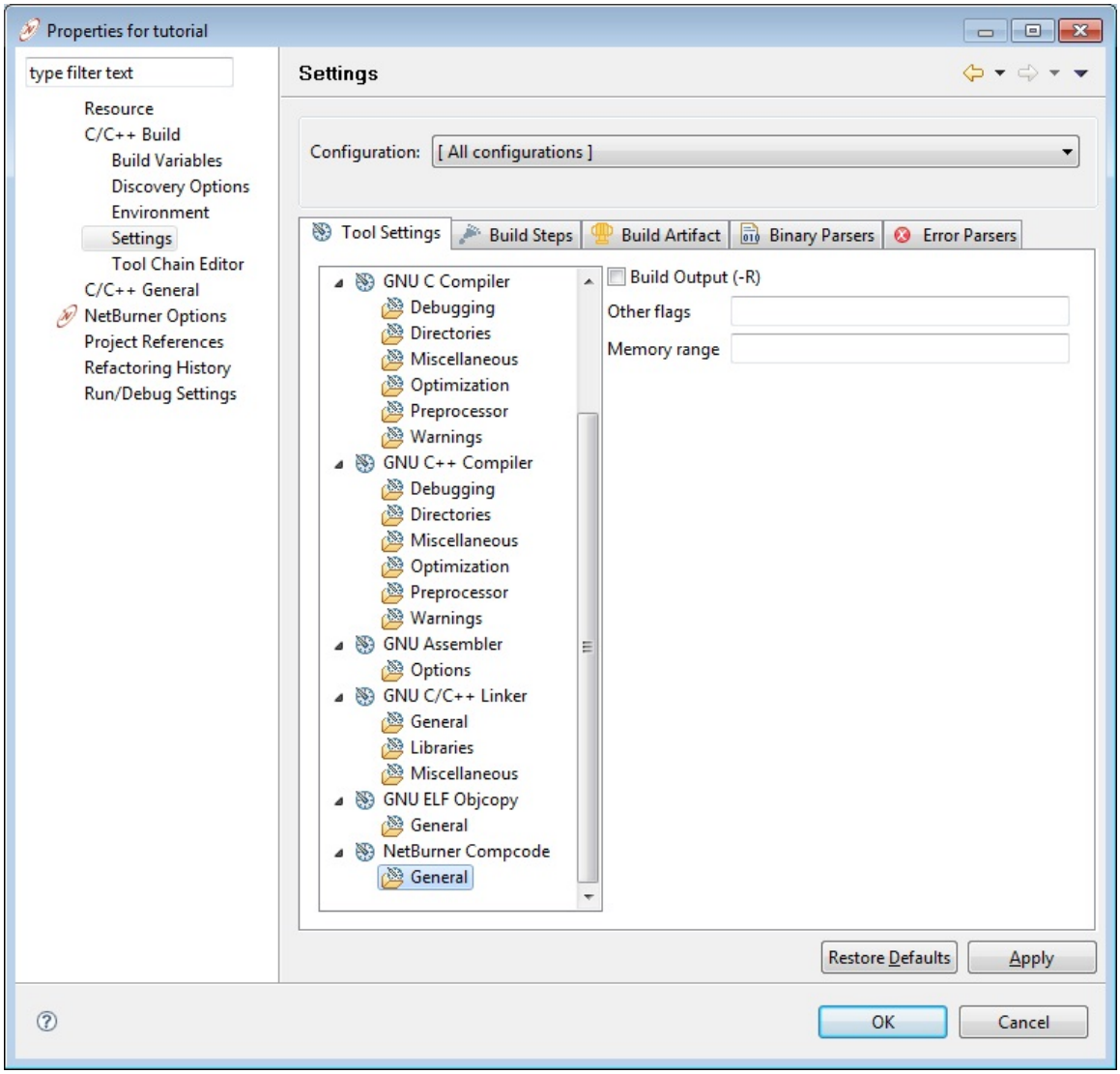
- Set the Project name, Location, Project type, and Toolchains. Click on Next.



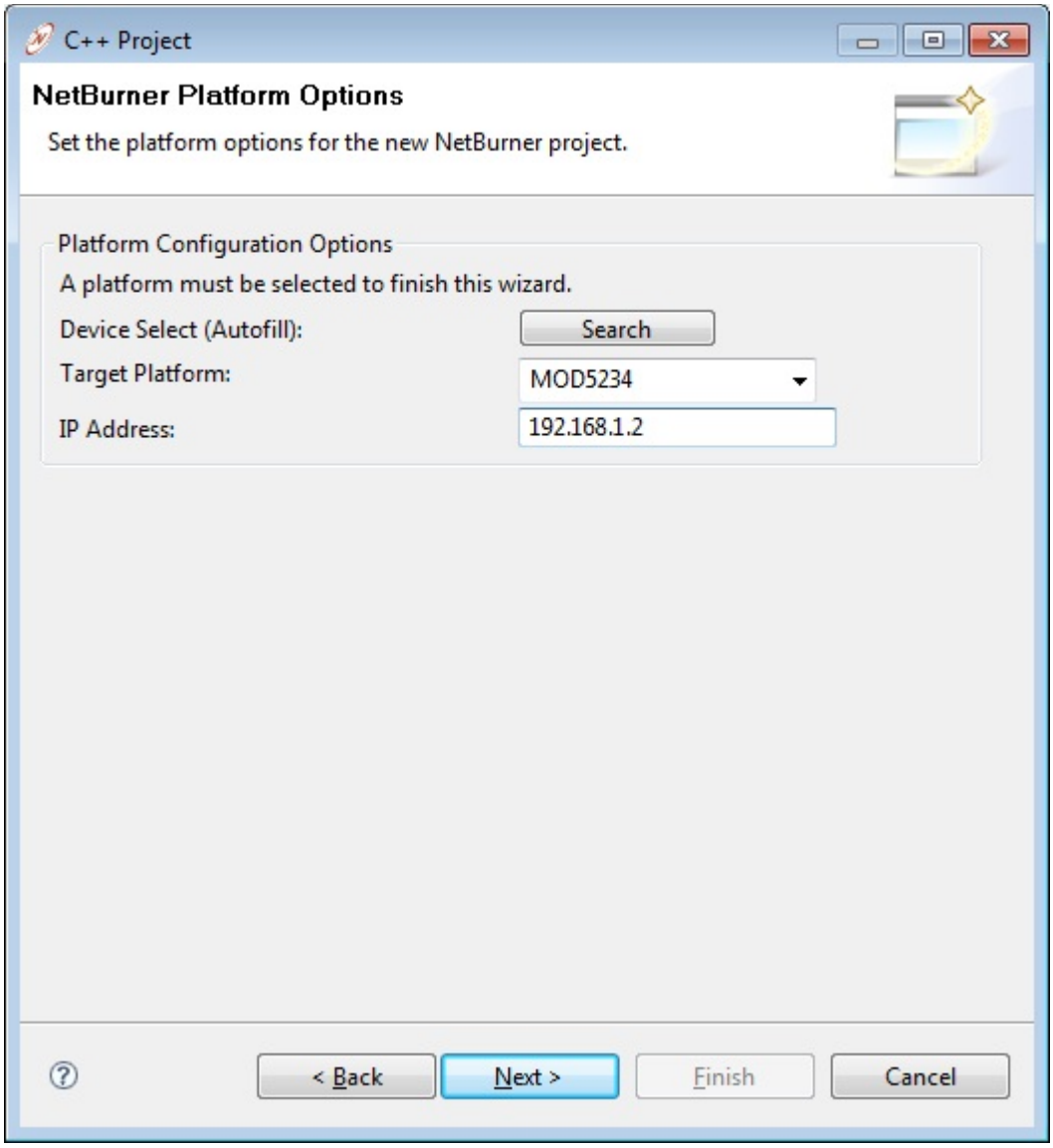
- Click on Advanced Settings.



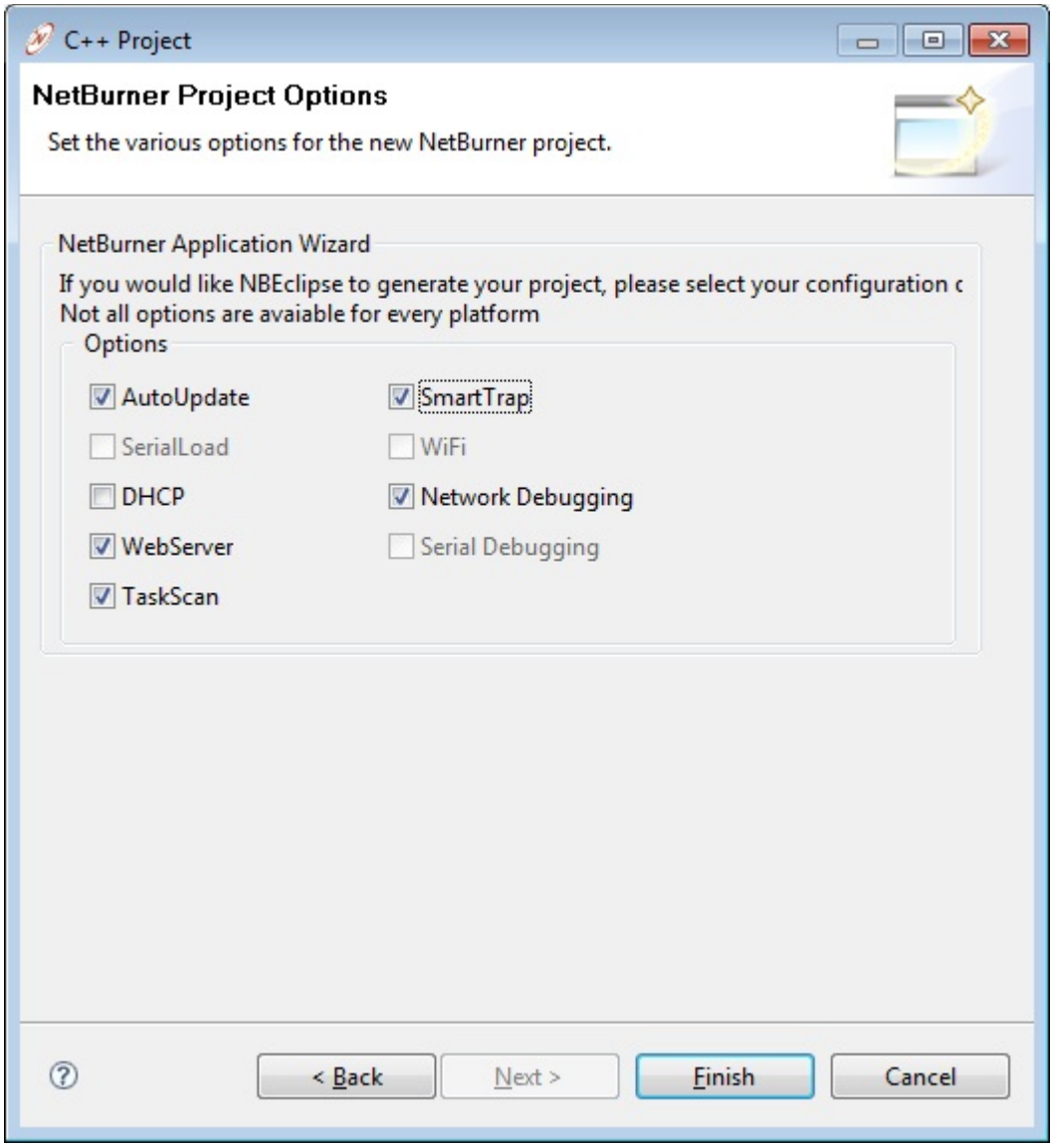
- Select the [All configurations] option from the dropdown menu and uncheck the Build Output (-R) checkbox for the NetBurner Compcode General setting. Click on Apply and OK. Click on Next.



- Set the NetBurner Platform Options. Click on Next.

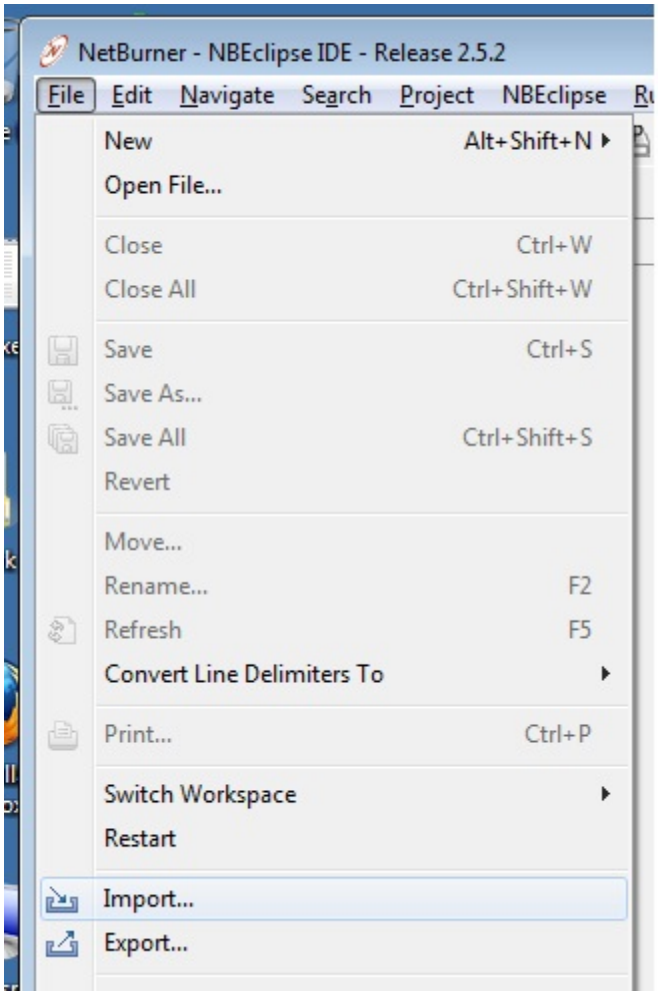


- Set the NetBurner Project Options. Click on Finish.

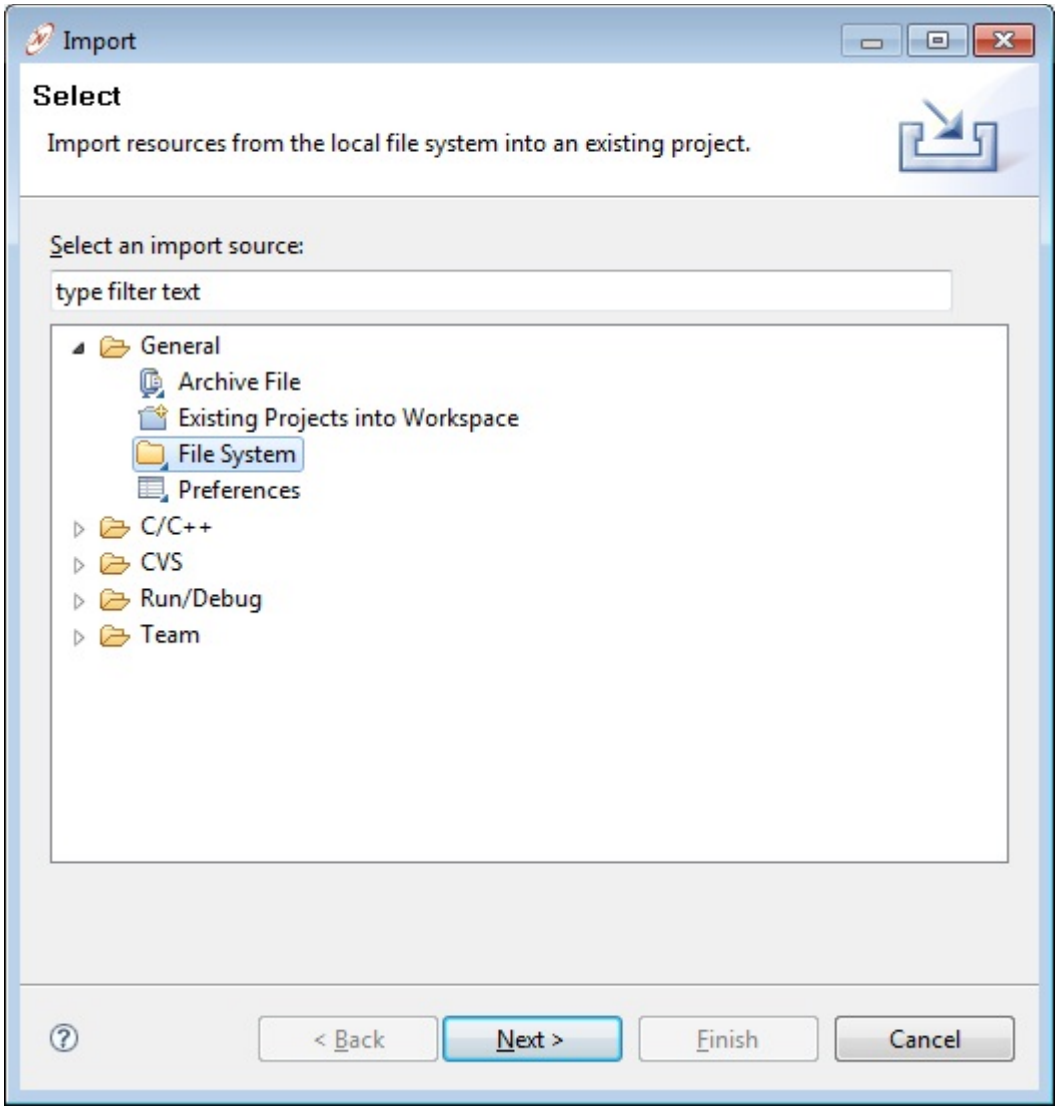


- Retrieve the provided tutorial code from the eClass site under the Tutorial section and store it into a suitably named folder.

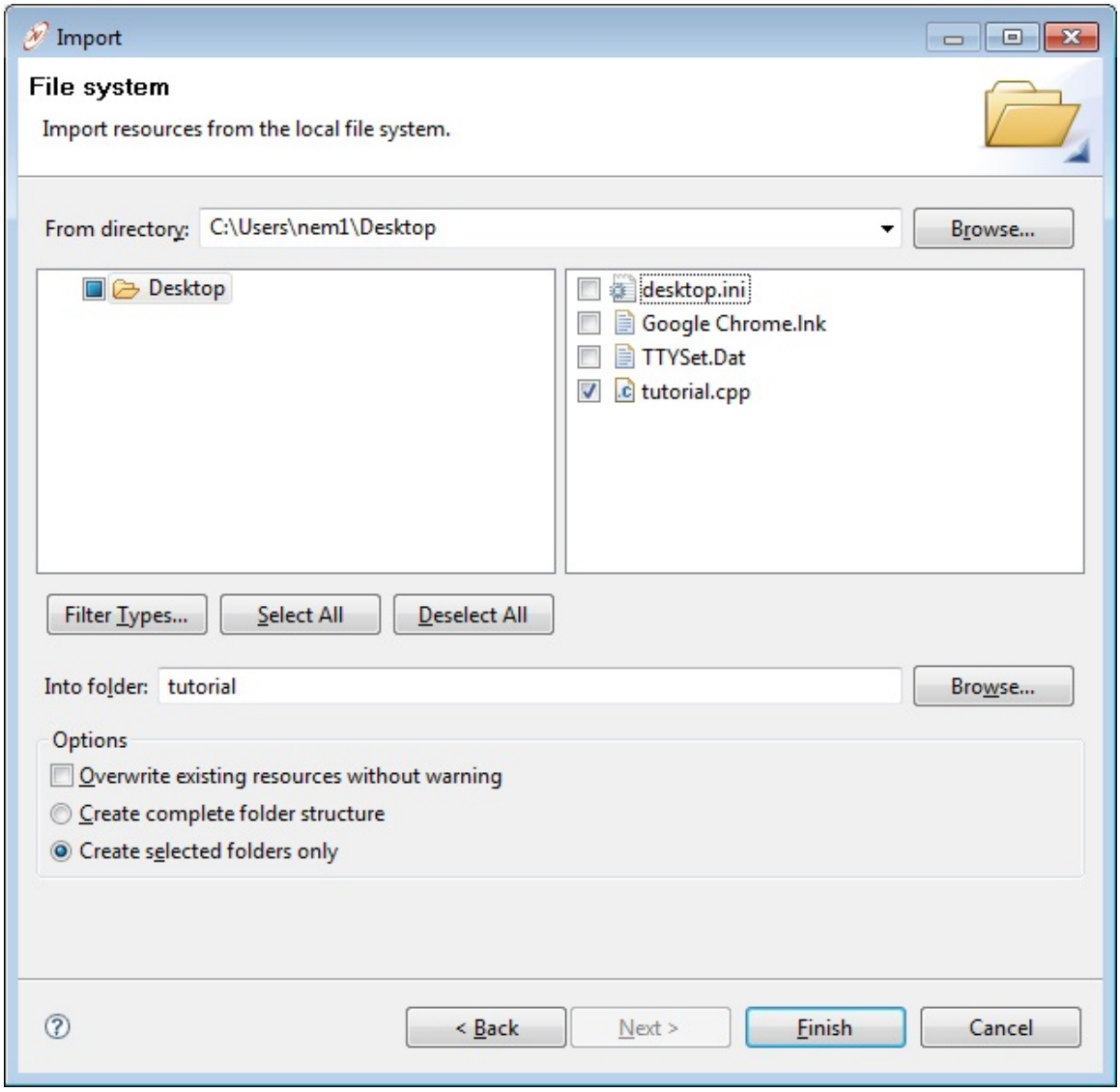
- Now we'll import the tutorial.cpp code into our project.



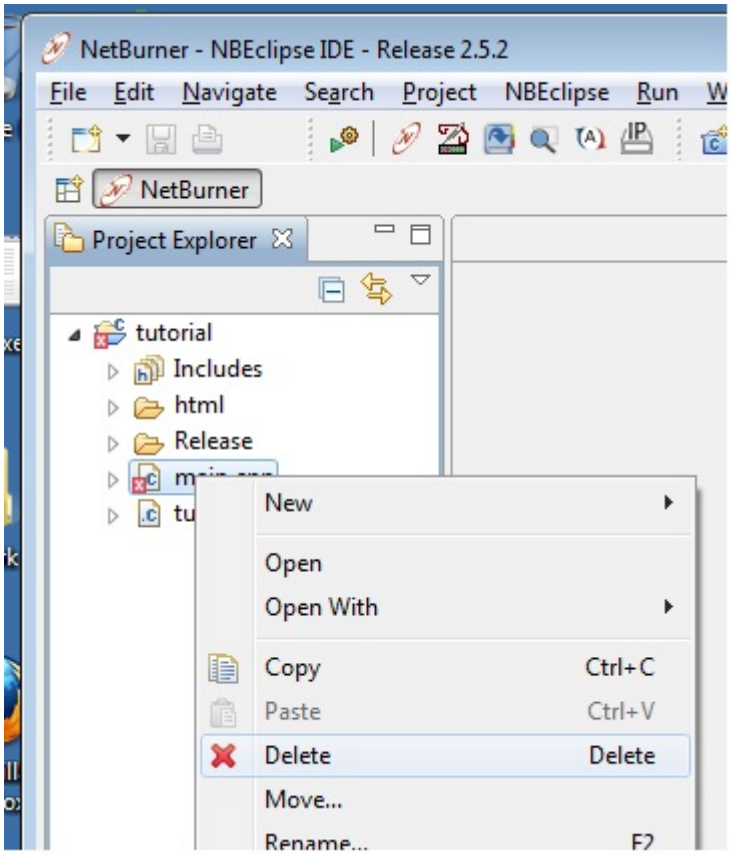
- Select the File System to import your code from. Click on Next.



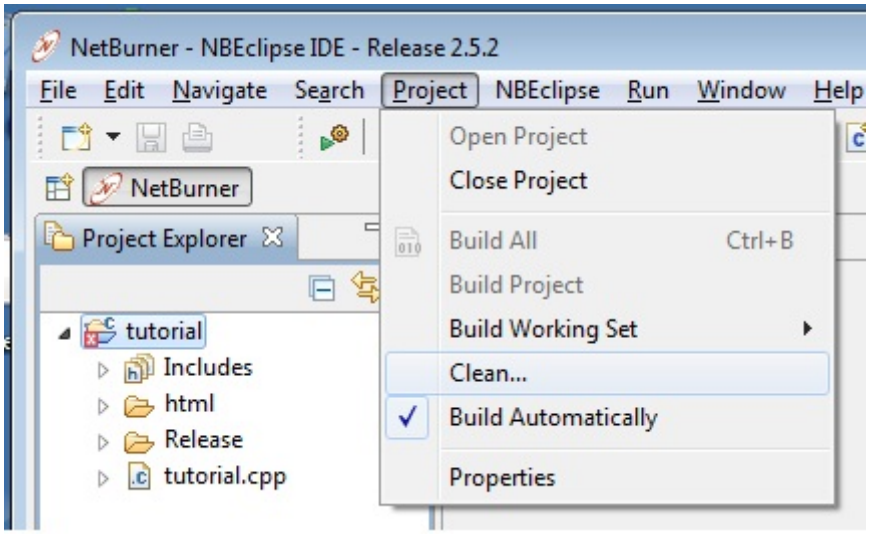
- Browse to where you stored the tutorial.cpp file. The example location in the screen shot below is probably different from where you stored yours. Click on Finish.



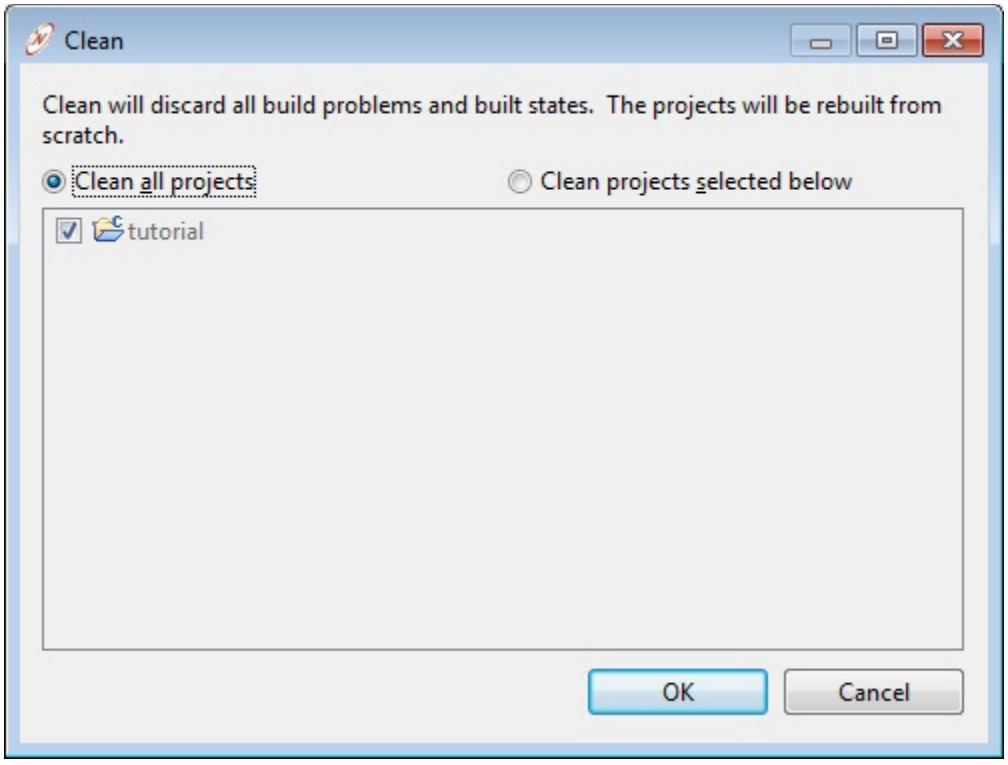
- The import will generate a build error because of the duplicate symbols in the existing main.cpp file. We need to delete the main.cpp file, and then clean and rebuild the project. Cleaning the project removes all the existing compiled object files and defined symbols. Once the project is cleaned the project should compile and link correctly. Right-click on the main.cpp file and select Delete.



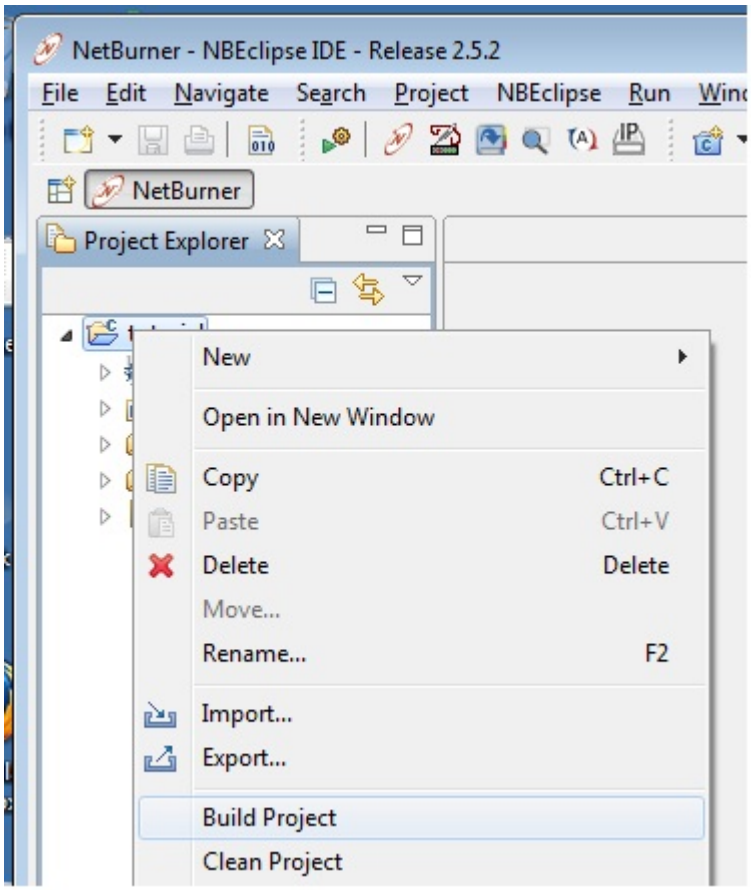
- After you have deleted the main.cpp file, you'll need to clean your project.



- Select the project to clean and click on OK.



- If you have not changed the default “Build Automatically” setting, the project will build itself. But if you have turned off the “Build Automatically” setting you’ll need to manually build the project. Right-click on the tutorial project folder to build the project manually.



Verifying the Board Connections

The lab instructor has completed many of the items listed below at each lab station. You should verify each of these items each time you sit down to work.

Before you connect any cables it is important that you put on your anti-static strap. Taking precautions against damage caused by the discharge of static electricity from your body is critical to the health of your vulnerable electrical components. Modern electronic devices can be damaged or destroyed by discharges of only hundreds of volts, well below the strength of discharges that a human would notice. Make putting on your strap a habit when working in any lab.

- Wear your anti-static strap!

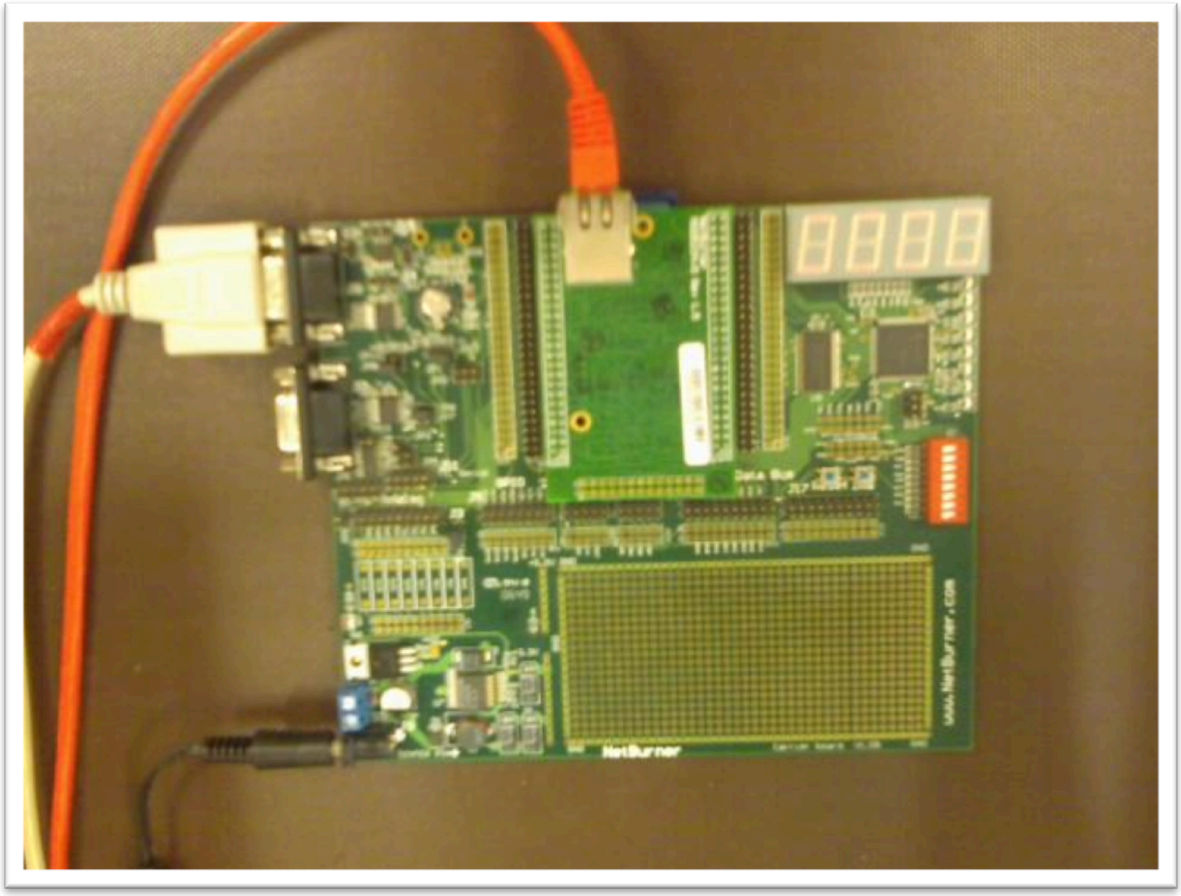
- Verify that the power cable is connected.

- Verify that the red network cable is connected into the second NIC on the back of the computer, and into the Ethernet connector on the NetBurner board.

- Verify that the serial cable is connected to the serial DB9 connector that is furthest away from the power cable. There will be a port conflict on lab 1 if the other DB9 connector is used.

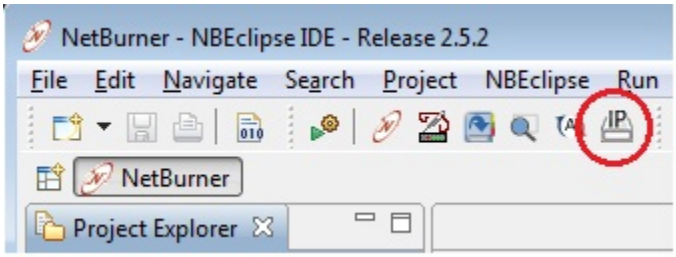
- Verify that the board is powered on by checking that the power LED is on. The power LED is identified as led9 next to the power-plug.

- Once your cables are plugged in, your board will look like this.

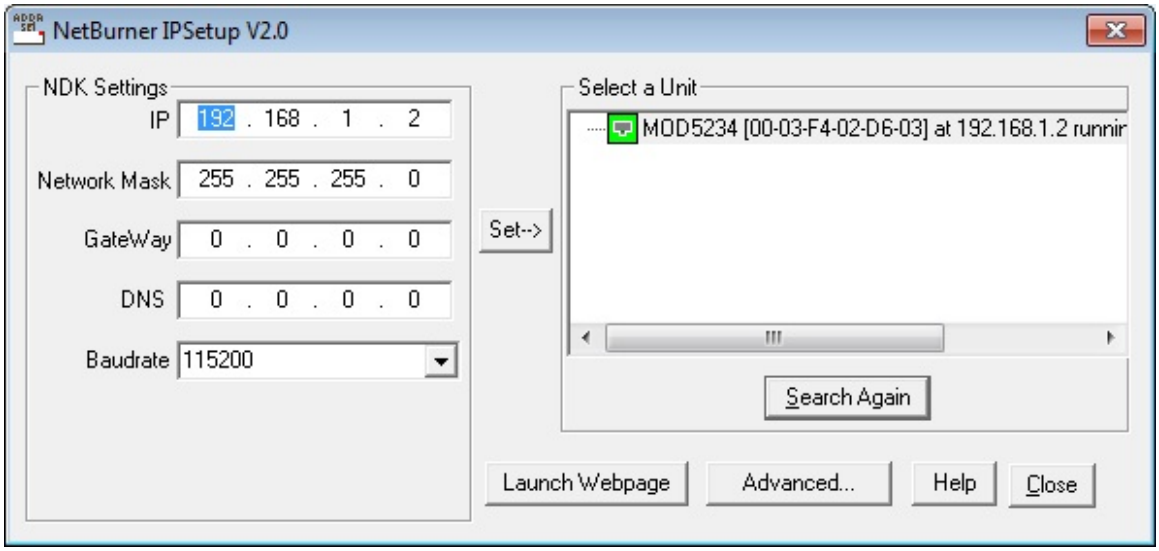


Verifying the Board Network Settings

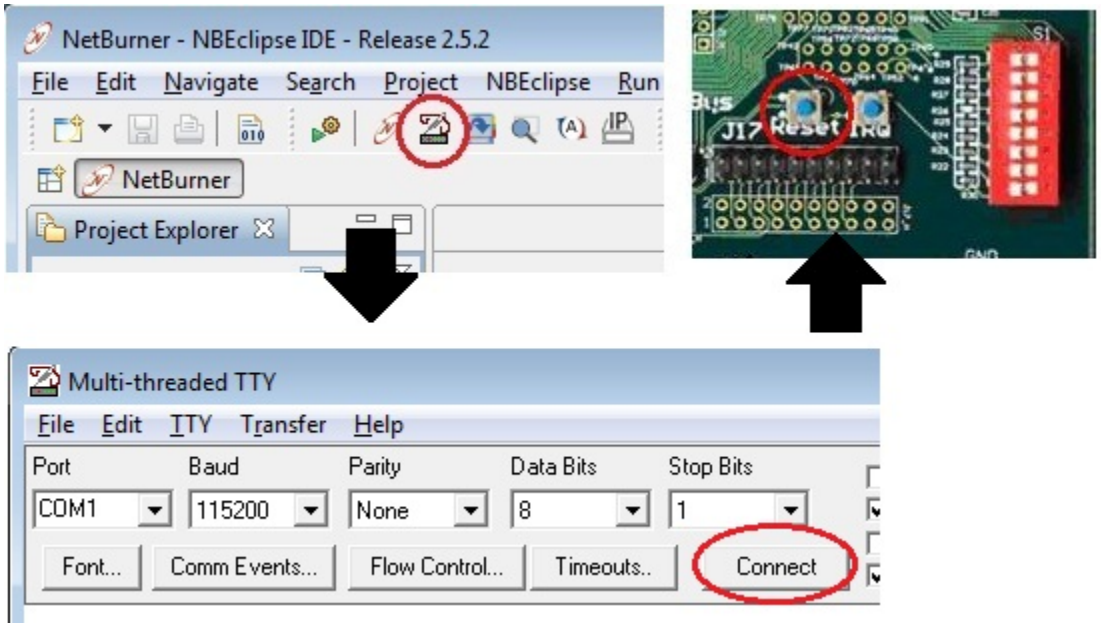
- Start up the program IPSetup by clicking on the IPSetup icon in the toolbar. The module must be plugged in and powered for it to be recognized.



- All of the boards in the lab have been pre-configured. Ensure that the settings match the screenshot below. To change incorrect settings, enter them into the configuration box on the left and click “Set”.



- Next, we’ll connect over the serial connection using Mttty. Click on Start->All Programs->NetBurner NNDK->Mttty Serial Terminal and click on “Connect”. Alternatively, click on the Mttty icon within the NBEclipse toolbar. Reset the board by pressing the Reset button on the board. Ensure that you can see some messages from the NB monitor before continuing. Once the timeout expires the main program will run. Note that whatever program happens to be loaded in flash memory will run. The running program may not be yours.

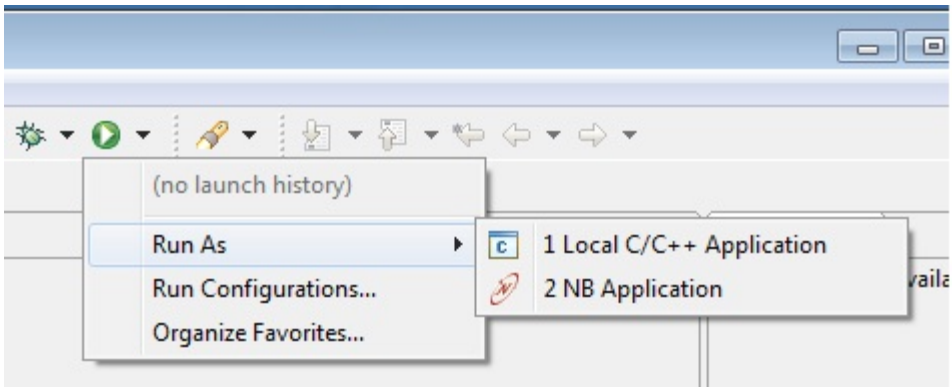


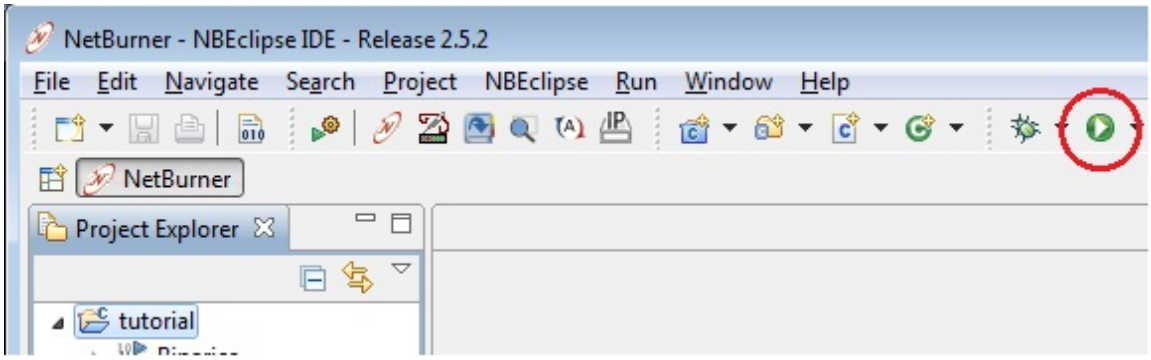
Flashing and Running the Project Code

We will now run the provided tutorial program that you've just built. In our system the code is not run directly from RAM. The code is first downloaded and written into flash memory using the AutoUpdate tool and the board is then rebooted to run the code.

- Hook up the oscilloscope leads to the heartbeat signal located at J2[44] and the context switch signals J2[41]. The heartbeat signal is produced by the operating system based on the ticks per second. Record the frequency of the heartbeat signal. The context-switching signal toggles on every context switch. We'll learn more about tasks and context switching as the semester progresses. Think about the differences between the two signals and what they mean. You won't see any signals until your tutorial code (or any code) is running.

- Right-click on the tutorial project and select Run As->NB Application. Normally, you only need to do this the first time the project is run. On subsequent runs you can click on the Run button on the NBEclipse toolbar instead.





The AutoUpdate program should be able to send the program now, and it will run after being written to flash memory. Watch in the Mttty window to see any messages generated by the running code. Open your web browser, and type in the IP address assigned to the board in the address bar. All of the boards in the lab have been assigned the same non-routable address: 192.168.1.2. The default web page will be served from the built-in httpd server. You should also see a series of trailing lights on the LEDs on the board.

IO Coding Exercise

In this exercise, your assignment is to read the value of the DIP switches and turn on and off the corresponding LEDs. DIP switch one should control LED one. DIP switch two should control LED two, etc. All of the code for the LEDs, seven-segment display, and the DIP switches is in the ioboard.cpp file. This file is already compiled and linked into the libraries so you do not have to add it to your project. Just call the subroutines as needed. The lab instructor's solution will be running on one of the workstations in the lab so you can always compare with your solution if you wish.

To identify your code, place the first names of each of your team members in the AppName constant char * array. You'll see it as Mttty displays the startup of the code. You'll be asked to do this in every lab to identify your code.

Memory Management Exercise

In this exercise, your assignment is to add in the incrementing seven-segment functionality. The seven-segment display should increment from 0x0000 to 0x9999 and rollover. The whole point of this exercise is to use pointers to complete the task, so stick to using the two variables available in the tutorial code. You have been given a pointer variable and an array. You may use either variable or both variables based on your preference.

Memory management and pointer operations are often poorly understood so we'll start here with a small exercise.

Report Requirements:

There is no report required for the tutorial.

Marking Scheme:

No marks are assigned for this tutorial. However, it is highly recommended that you complete it before the start of lab 1.