Getting Started with nqBASIC

Preliminary

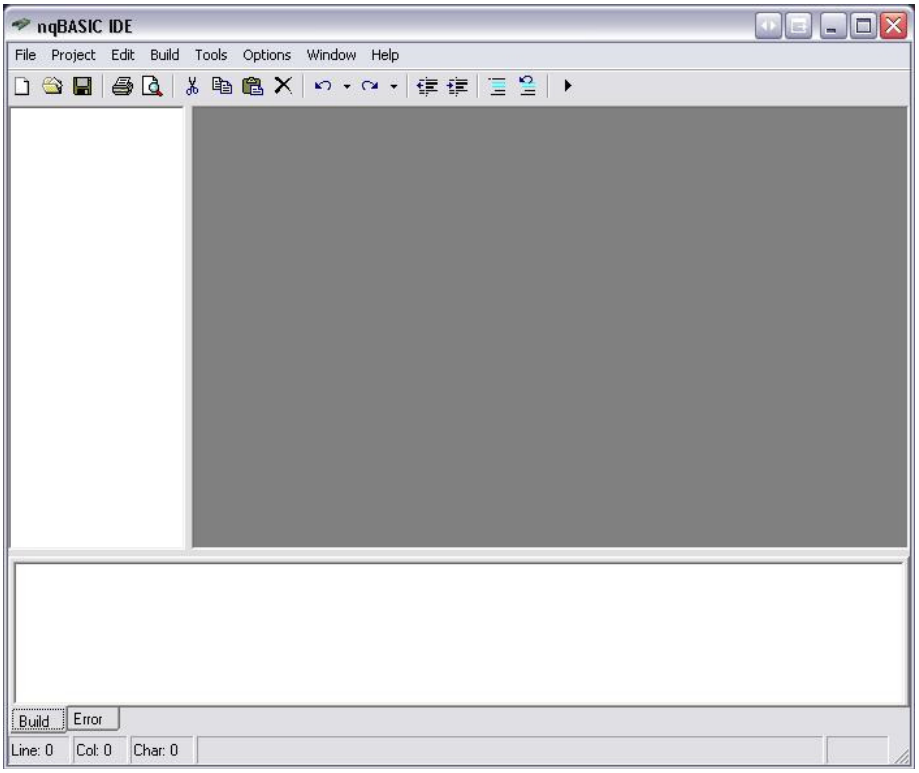
This document introduces the nqBASIC integrated development environment (IDE), a Windows-based graphical user interface (GUI) designed for easy application development with nqBASIC and the NanoCore12 family of microcontroller modules. Creating projects, and writing, editing, compiling, and loading programs can all be accomplished quickly and easily with this IDE.

Getting Started:

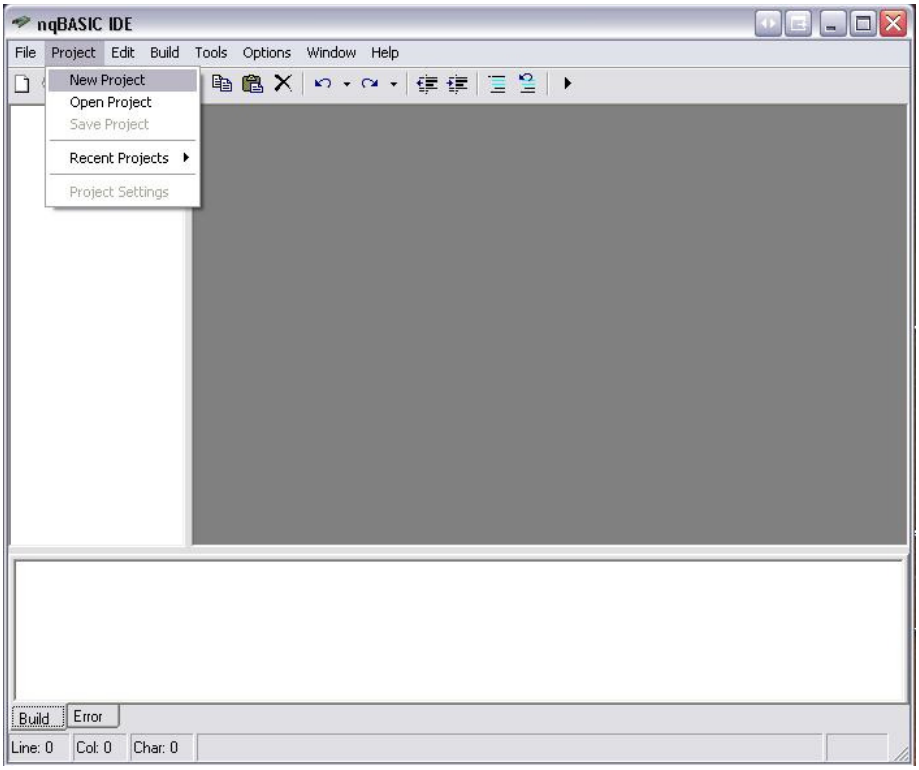
This document assumes that you have already completed installation and registration of nqBASIC. If not, then please take a moment to do so. Check the Resources section of this site for a video demonstrating how to do this.

When you are creating a project, it is important to select which module (i.e. "the target") you are using: NC12C32 (24 pin), NC12C32DX (32 pin), or NC12MAXC32/128 (40 pin). This makes the IDE aware of which port pins are available for your application.

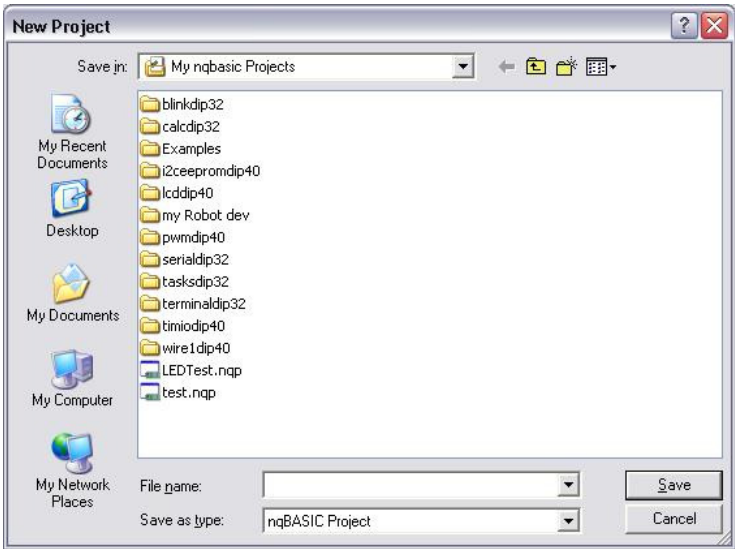
To start, double click the nqBASIC icon to launch the IDE. Below is the startup screen that you'll see.



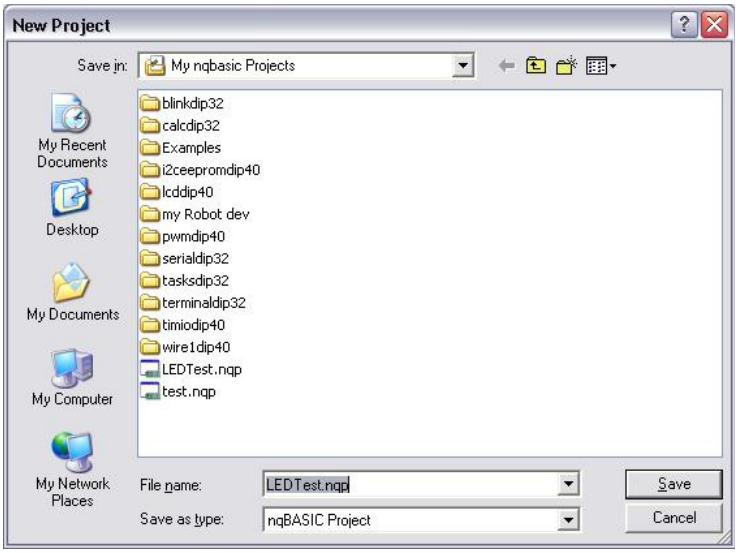
From the **Project** menu, select **New project**.



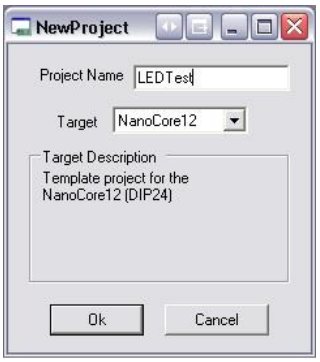
The **New Project** explorer window will pop up, as shown below.

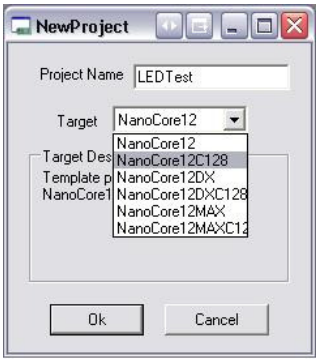


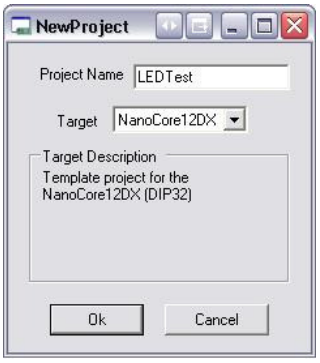
For our first example, we intend to work with LEDs, so we'll create a new project and call it `LEDtest`. Type **LEDtest** in the **File name** field and click **Save**.



A **NewProject** Window will pop up, showing **LEDtest** as the project name, and will show, by default, that the intended target is NanoCore12 (24 pin DIP). To select a different Target, click the dropdown arrow to reveal the available targets, and select the one you are using.

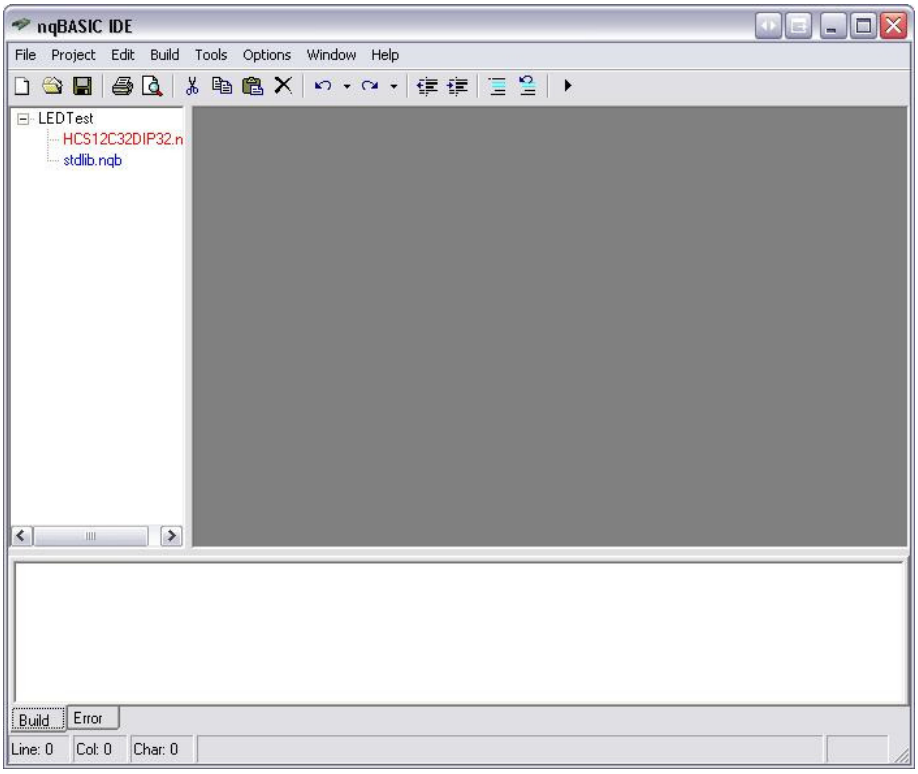






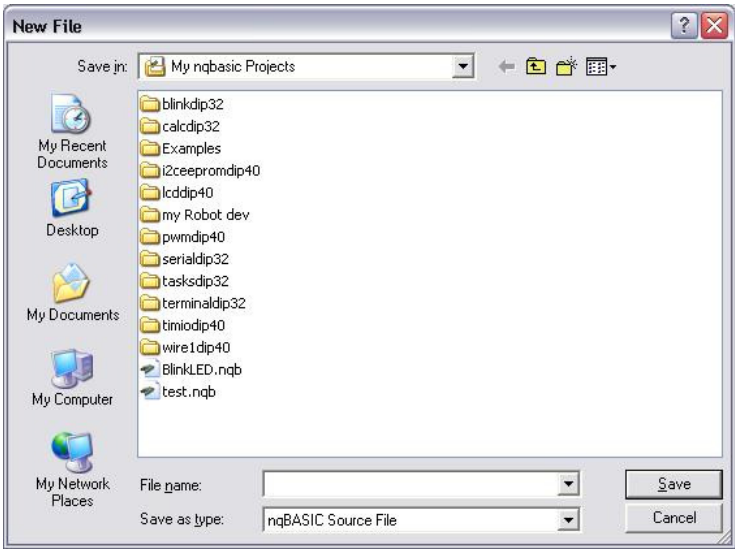
Once a Target is selected, click **OK**.

The leftmost pane of the IDE will change to show the addition of a couple of files. The file named **HC12C32DIP32.nqb** provides a list of port pin designations to the IDE so that it will be aware of which pins are available for your project. The **stdlib.nqb** file contains all of the standard nqBASIC functions that can be used in your program.

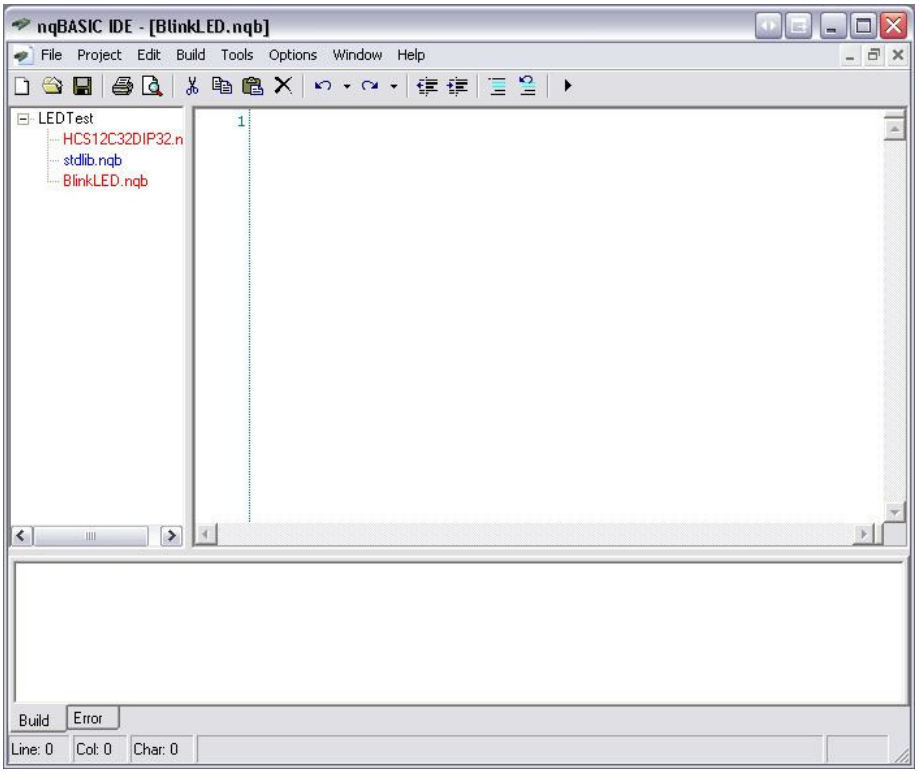


The grey window pane is where you will enter the source code for your program. To start, choose **File** – **New File** to add a file to the Project. Click the **Yes** button to agree.

You will be prompted to name the file. Enter **`BlinkLED`** and click **Save**.



Note that the new file has been added as **BlinkLED.nqb** to the file window pane, as shown.



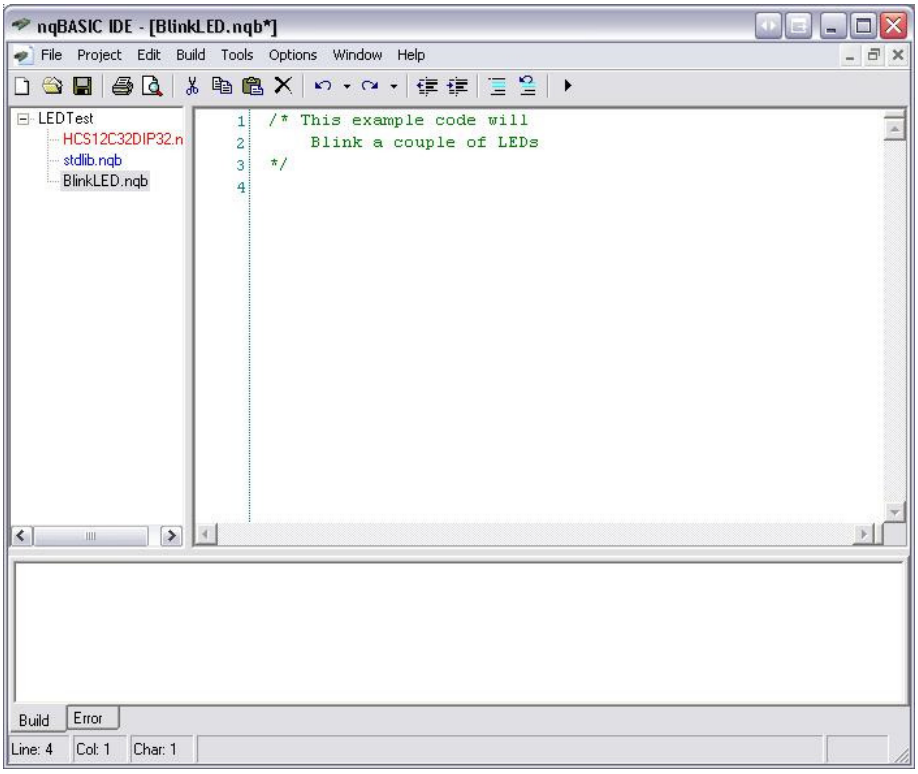
Notice that the center pane has now changed, and is ready to accept lines of code. We will add comments to describe the example. The syntax of comments is similar to C.

/* This is a comment format opener
and this is the closer */

// this is a single-line comment

In the pane below, comments have been added to explain the purpose of the exercise.

/* This example code will
blink a couple of LEDs
***/**



In this document, the syntax will not be explained in detail, except where required. The main objective here is to demonstrate how to write a program, compile it, and download it to your NanoCore12 module. Start by typing the code shown below into the editor window (or just copy/paste it).

How to use Const:

In the example we're creating, we'll need to declare a few constants.

Const FOREVER = 1
Const DEL_1MSEC = 1000
Const OFF = 0
Const ON = 1

Here the created constant FOREVER is given a value 1. This constant is checked in the Main program loop **while (FOREVER) loop** to decide whether to terminate. The usage of the rest of the constants is fairly apparent:

Const DEL_1MSEC = 1000 used in a delay loop
Const OFF = 0 used to turn OFF the LEDs
Const ON = 1 used to turn ON the LEDs

How to use Dim:

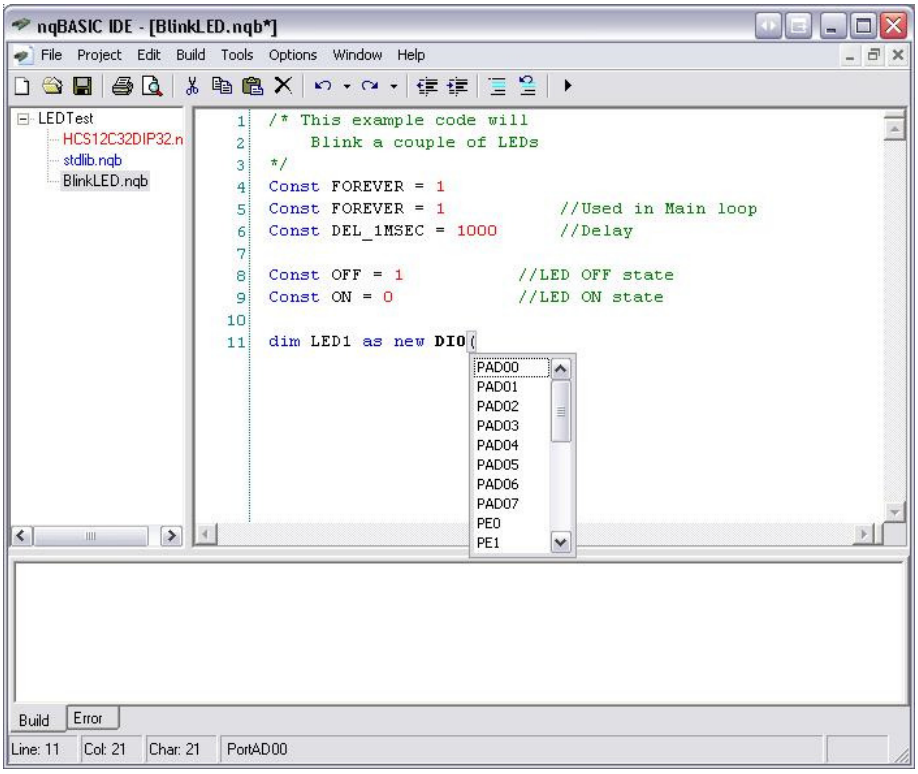
Now we need to declare which port pins to use. The `Dim` statement will allow us to declare which pins are to be assigned to the two LEDs. In the IDE, type the line shown below, and you'll discover how the syntax-aware IDE will help in the assigning of ports. As mentioned previously, it will only present the pins that are available on the target you have chosen. Not only that, but as you progress with port pin assignments, the list will be shortened to exclude pins which you have already assigned!

Dim LED1 as new DIO (

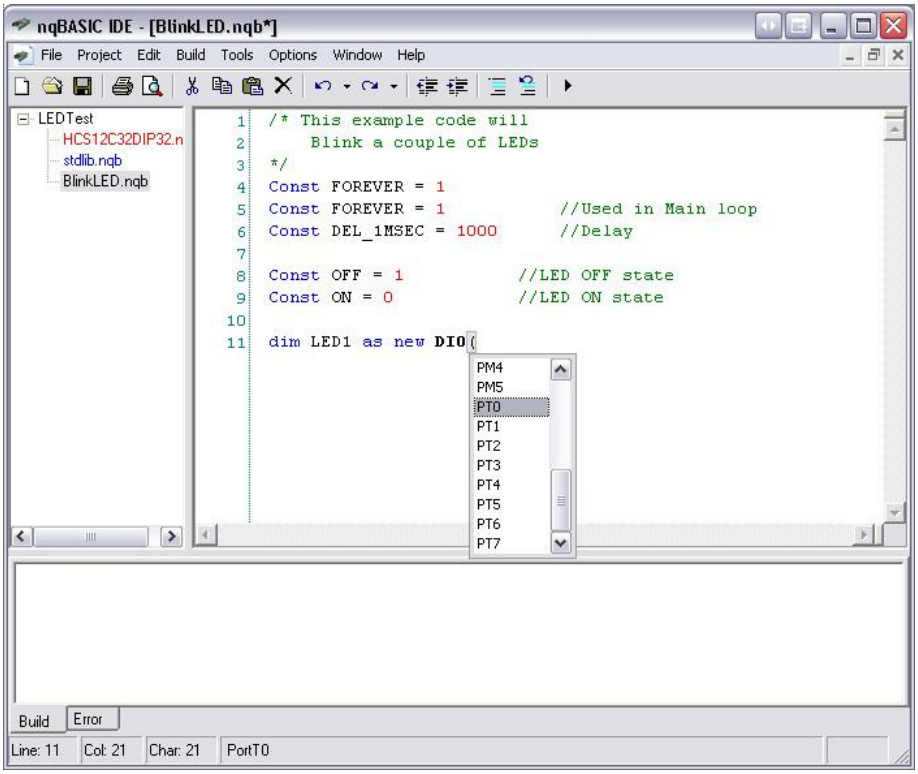
After the **(** open bracket is typed in, a dropdown list will appear beside it to help you select which port to use. Here, we will assign PT0 and PT1 to the LEDs.

To select a pin, use the mouse to scroll down and then make your selection by clicking on it. The keyboard up and down arrow can be used instead, if you prefer. Once the desired port is highlighted, press ENTER on the keyboard (or double-click with the mouse) to confirm the selection.

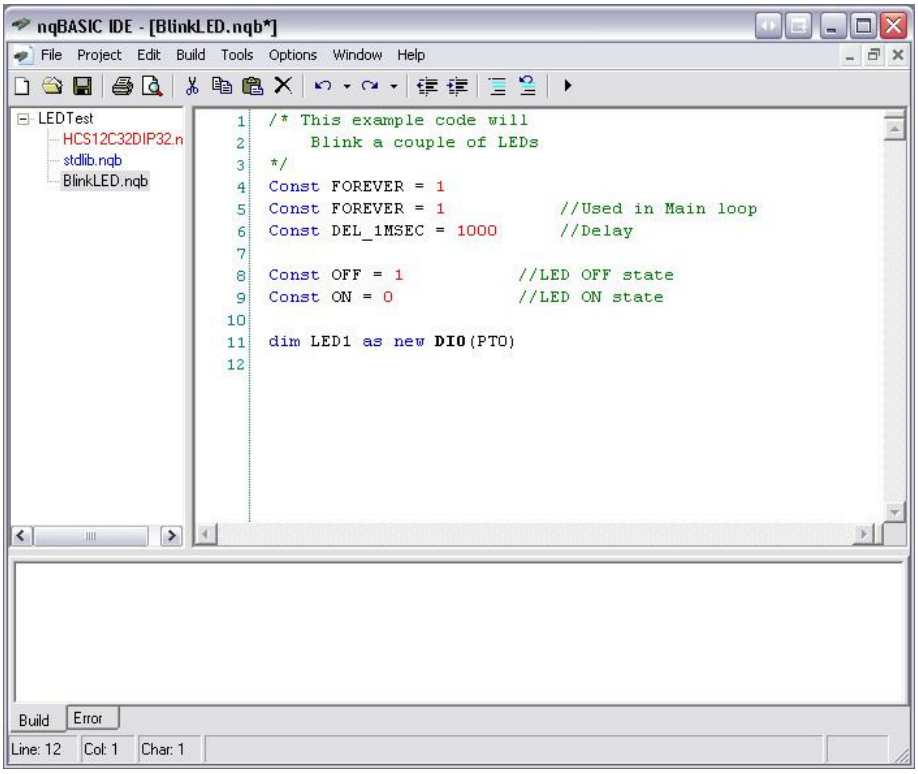
Below will show the sequence of steps followed after typing in the "(".



Scrolling down to PT0.



Once PT0 is selected, it will be automatically added to the line. The same sequence is followed with PT1.



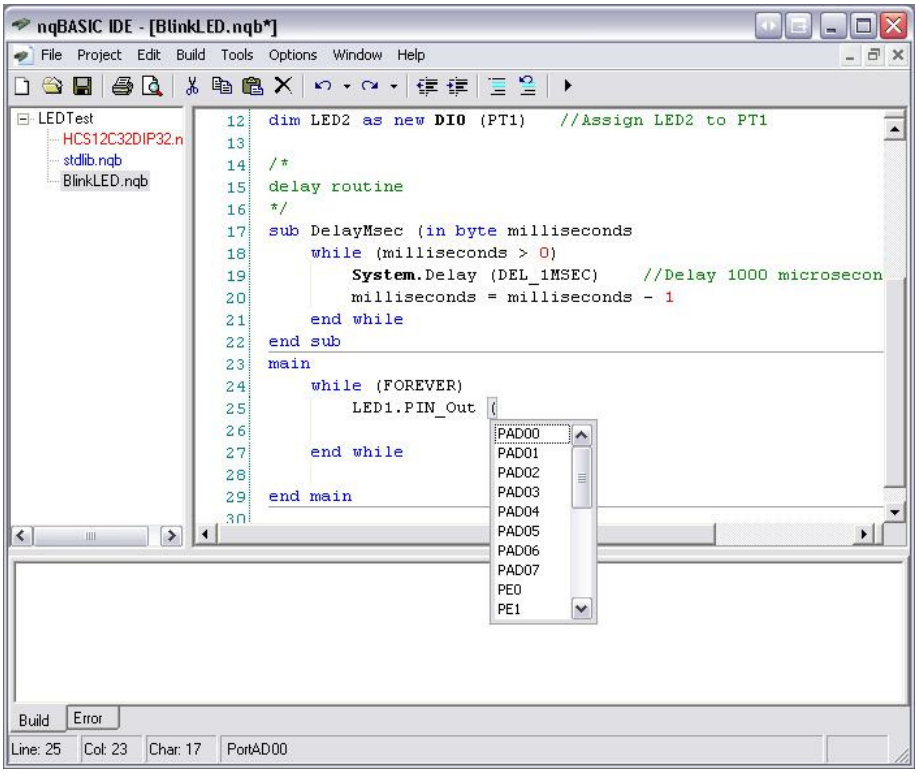
Now, let's carry on, typing in the rest of the program, starting with the delay subroutine. In the main loop there is another dropdown list to assist you.

```
/* This example code will
Blink a couple of LEDs
*/

Const FOREVER = 1        //Used in Main loop
Const DEL_1MSEC = 1000    //Delay
Const OFF = 1            //LED OFF state - Active high to turn it OFF
Const ON = 0             //LED ON state - Active low to turn ON the LED

dim LED1 as new DIO (PT0)        //Assign LED1 to PT0
dim LED2 as new DIO (PT1)        //Assign LED2 to PT1

/*
delay routine
*/


sub DelayMsec (in byte milliseconds>
    while (milliseconds > 0)
        System.Delay (DEL_1MSEC)      //Delay 1000 microsecond to milliseconds = milliseconds - 1
    end while
end sub
```

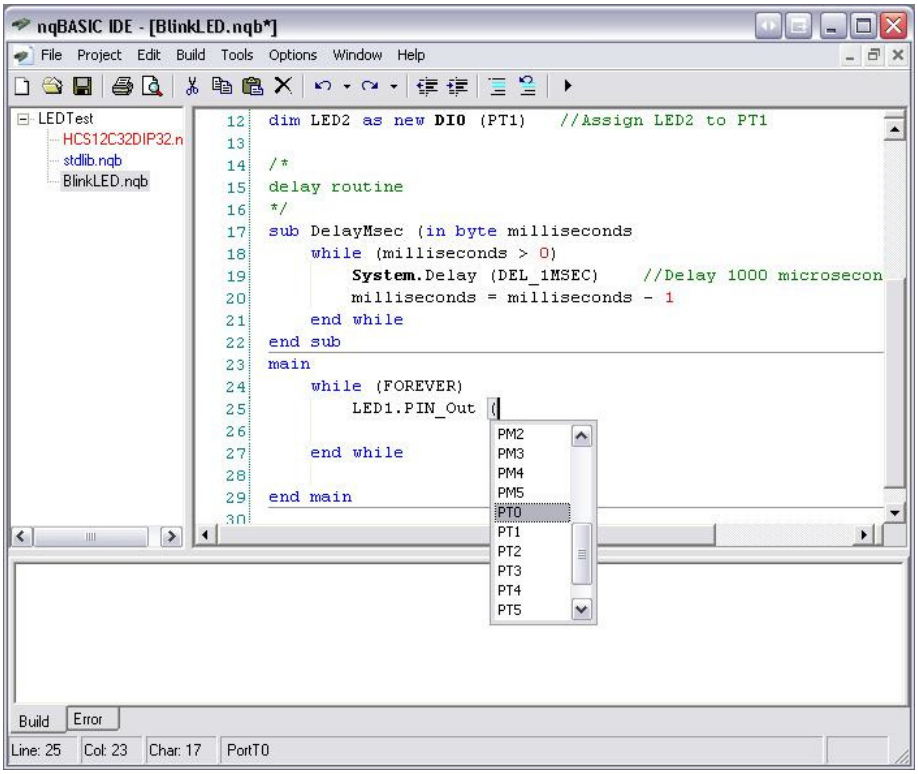
Now, to the Main loop. Here the IDE will display another dropdown menu to help you select which port pin to use. Type in the lines below in the IDE

```
Main

While (FOREVER)
    LED1.PIN_Out (
```

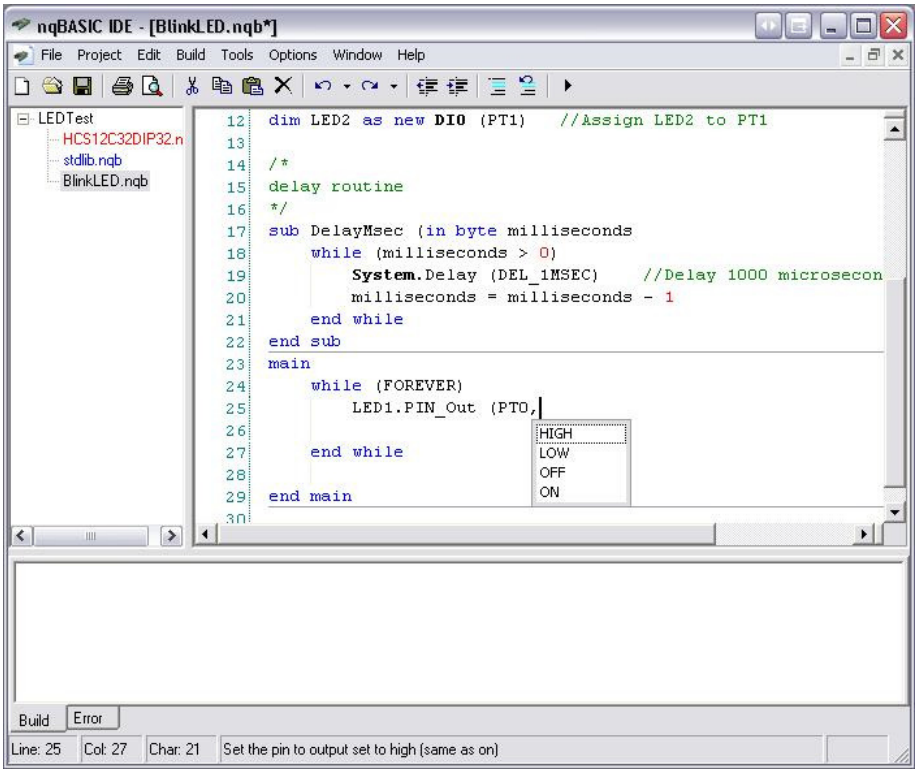
After typing in the opening bracket "(", a dropdown list will once again be displayed to help you select the Port pin.

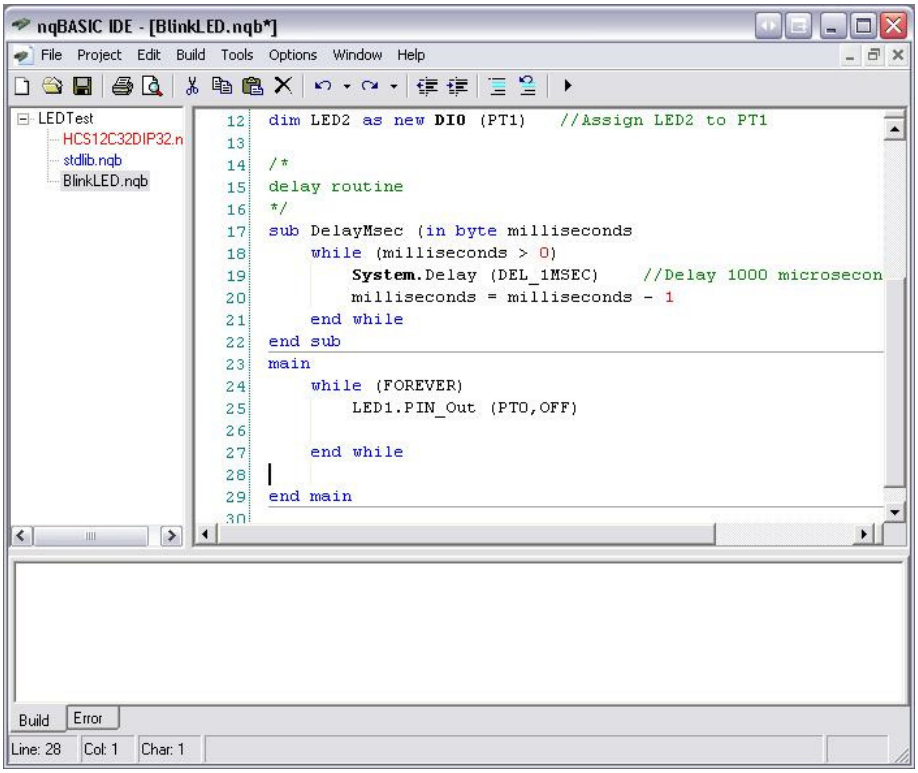


Scroll down to select PT0 or PT1.



Then choose the state of the LED as defined by the Constant.





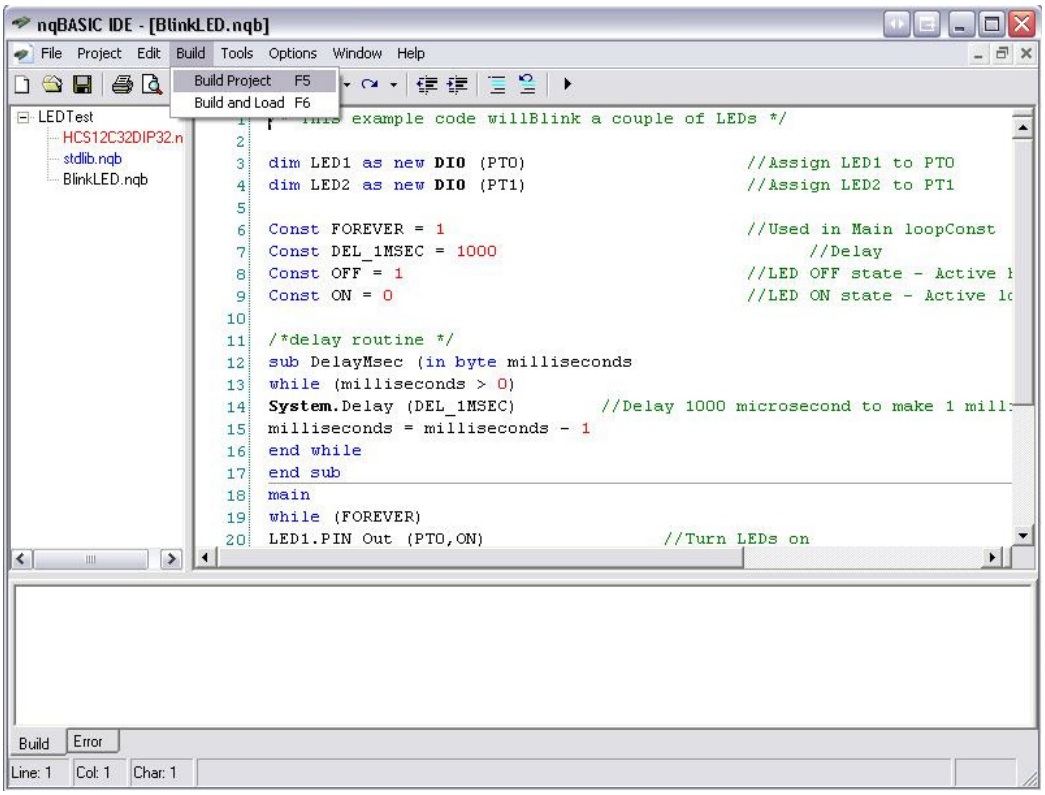
Here is the entire source code that will be compiled, from start to finish.

```
/* This example code willBlink a couple of LEDs
*/

Const FOREVER = 1                    //Used in Main loopConst
Const DEL_1MSEC = 1000                    //Delay
Const OFF = 1                    //LED OFF state - Active high to turn it OFF
Const ON = 0                    //LED ON state - Active low to turn ON the LED
dim LED1 as new DIO (PT0)                //Assign LED1 to PT0
dim LED2 as new DIO (PT1)                //Assign LED2 to PT1
/*delay routine
*/

sub DelayMsec (in byte milliseconds

    while (milliseconds > 0)
        System.Delay (DEL_1MSEC)        //Delay 1000 microsecond to make 1 millisecond
```

```
        milliseconds = milliseconds - 1
    end while
end sub
main
    while (FOREVER)
        LED1.PIN_Out (PT0,ON)            //Turn LEDs on
        LED2.PIN_Out (PT1,ON)
        DelayMsec (250)                  //Delay 0.25 seconds
        LED1.PIN_Out (PT0,OFF)           //Turn LEDs off
        LED2.PIN_Out (PT1,OFF)
        DelayMsec (250)                  //Delay 0.25 seconds
    end while
end main
```

Once the above lines of code have been typed (or copy/pasted) into the IDE we can now compile.

Select **Build** then **Build Project**.

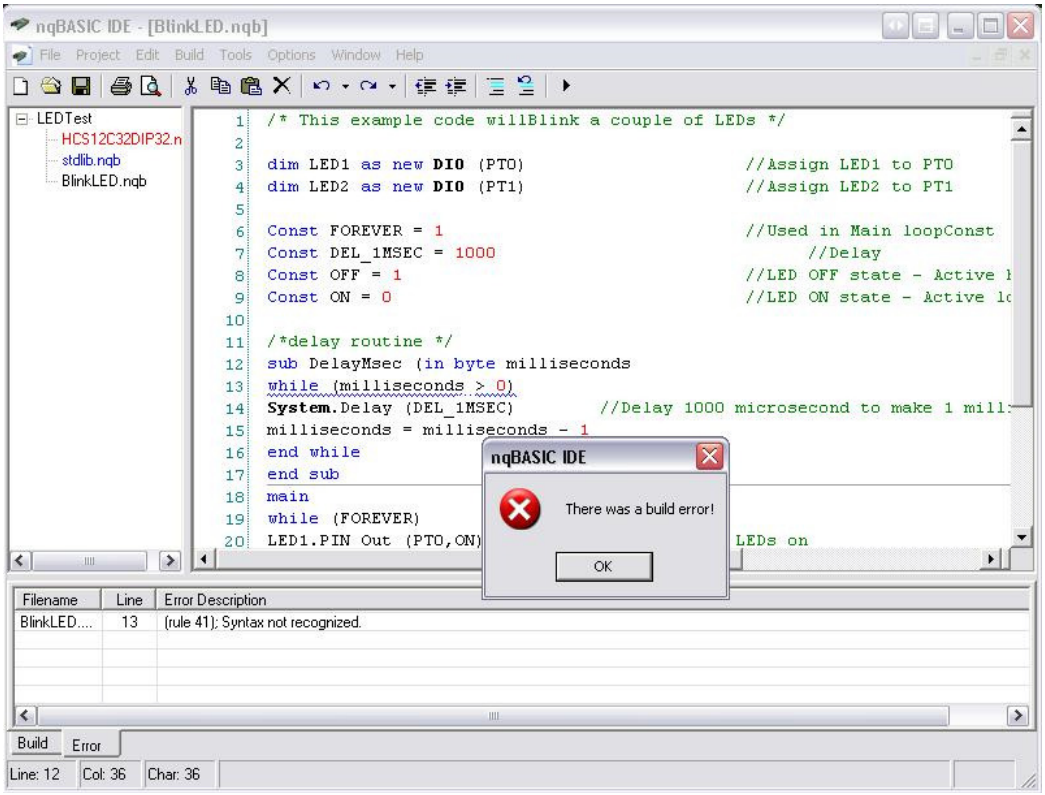


If the file has not been saved, you will be prompted to **Save** before continuing. Click **Yes** to continue.

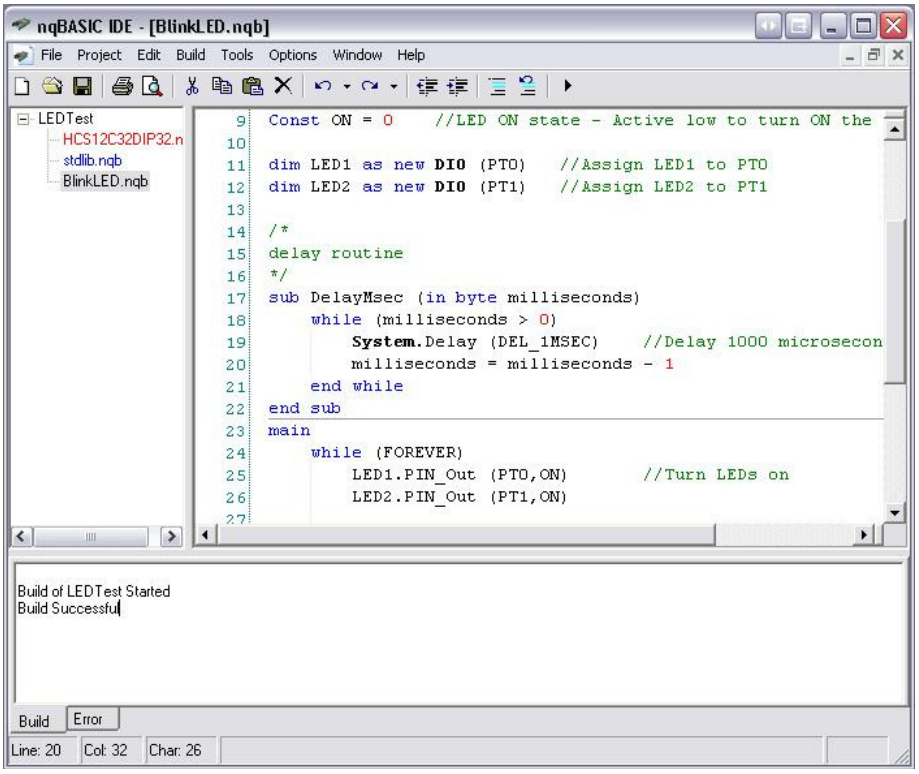
The IDE will save the file and immediately compile it. If there are errors, an error message will be displayed, and the line of code where the error occurred will be highlighted. Click **OK** to continue.

A ')' was deliberately missed, to force an error message to be displayed. This was done to demonstrate how the IDE pinpoints and displays errors. Fix the error to continue to the next level by adding the ')' to the line below then build the project once more.

sub DelayMsec (in byte milliseconds)



After compiling, the IDE will display a message showing that the build was successful. This does not necessarily mean that the code will work-- it just means that it was free of syntax errors.



It's now time now to load the program into the NanoCore12 module.

Loading the program into NanoCore12

We'll assume that a Docking Module, School Board, or Servo/Sensor?Motor Interface Module is being used and that the NanoCore12 module has been properly inserted into the socket, with pin 1 of the module aligned to pin 1 of the socket (shown with a white square dot on the circuit board). If you're using a solderless breadboard or your own circuit board, make sure power and serial connections have been made correctly.

Connect the serial cable between your module and an available serial port of the PC. In this example, we'll assume that it is COM1 (other com ports can be used by substituting the correct number in the related command-- even a virtual com port, if you are using a USB adapter). Connect the other end of the cable to the docking module DB9 connector.

From the IDE navigation bar, select **Options** and then **Editor**. Select the COM port you are using, and click **OK**.

Slide the **Run/Load** switch on NanoCore12 to the **Load** position and then apply power to the Docking module. Make sure that the PWR LED is ON. If not, re-check your power supply and connections, and the orientation of NanoCore12 in the socket.

From the IDE navigation bar, select **Build and Load**.

The IDE will compile your code again, and then establish communication with your NanoCore12 module. If all goes well, you will see the message **Loaded Okay**.

Do not proceed past this point if you don't see the message. You must first troubleshoot to find out why.

2 possible errors can occur:

Connection Error: Unable to open COM1 -> Another application is using the COM port

Connection Error: Read Error: Timeout error -> The MCU is not currently in LOAD mode, or the cable is disconnected from either the PC or your NanoCore12 setup, or the module does not have power.

Once you have resolved any problems and loaded the code successfully, press and hold the RESET button while sliding the **Run/Load** switch to RUN. Then release the RESET button. The two LEDs connected to PT0 and PT1 pins will immediately turn ON and OFF in rapid succession.

Congratulations! You now know how to create an nqBASIC program and load it into your NanoCore12 module! Using the Reference material and other tutorials, you will soon be creating many fun and useful applications.

Last Updated (Tuesday, 20 October 2009 09:04)