# Tutorial on

# USB BOOT LOADER using LPC1768 BASED BOARDS

Designed by
CoiNel Technology Solutions LLP
No-816, 2nd Floor, 4th B Cross, 9th A Main,
RPC Layout, Vijaynagar,
Bangalore-560040
State: Karnataka
Country: India

www.coineltech.com



For any enquires, kindly contact info@coineltech.com

## CONTENTS:

The USB Boot loader for the LPC1768 HPLUS is a cool trick that CoiNel created for some of our ARM Cortex M3 based boards. Since the ARM architecture allows for large flash space, loading code onto the LPC1768 HPLUS ARM Cortex M3 over the serial port was very slow. The bootloader brings various techniques together to create an easy to use and very fast development system to load the program without any need for expensive hardware or software.

In this tutorial we will cover what a Bootloader is, why should we use it, and how to use it.

## 1. WHAT IS A BOOT LOADER?

A Boot loader is a small piece of code that runs before the operating system starts running. In our case the bootloader is the code that runs before the device firmware starts up. Typically a bootloader is used because the system memory is too small to contain the entire program, and so the bootloader uses a set of routines to call the program from a different part of memory.

The CoiNel LPC1768 USB Boot loader performs three steps:

1. First, the Boot loader checks to see if a USB cable has been plugged in. If the LPC1768 board is connected as a USB device then it initiates a USB Mass Storage system. This will cause the target board to appear on any computer platform as a removable flash drive. The user can then seamlessly transfer files to the flash drive.
2. The next thing the Boot loader does is look for a firmware file. This file contains the desired operating firmware (in a binary file format) for the LPC1768 board. If the Boot loader finds this file system then it programs the contents of this file to the flash memory of the LPC1768. In this way, the Boot loader acts as a “programmer” for the LPC1768 HPLUS; and we can upgrade the firmware on the LPC1768 board simply by loading a new file.
3. After performing for first two checks, the boot loader calls the main firmware.

## 2. WHY WOULD YOU USE THE LPC1768 USB BOOTLOADER?

The bootloader provides several distinct advantages. First, if you have a LPC1768 Board that has been preloaded with the bootloader then you don't even need a programmer to load code onto the board! Another reason that LPC1768 USB Boot loader is great, because it is fast at loading code. When using a serial programmer to load code onto an ARM, it can take several minutes to get your code up. This can be a major setback when you're writing very large programs, and you start debugging. Using the LPC1768 USB bootloader allows you to load code in seconds.

## 3. HOW DO YOU USE THE USB BOOT LOADER?

First you have to program the Boot loader code onto the LPC1768 board. The Boot loader hex file can be downloaded at the following link. You may either use FLASH MAGIC software for programming (i.e., serial programming), CoiNel ARM USB JTAG (USB Programming) or H-JTAG (parallel port programming) to load this code. Once the Boot loader program is on the LPC1768, using it is very simple!

Note: Make sure you erase the old program before loading the bootloader code.

Let us assume that you have written some neat code to use on LPC1768 Board. Plug in a USB cable, and when the USB Mass Storage Device opens on your desktop, just drag and drop the binary file onto the device then press reset; the bootloader will look for the binary file and if the file is found the bootloader will program the new code onto the controller.

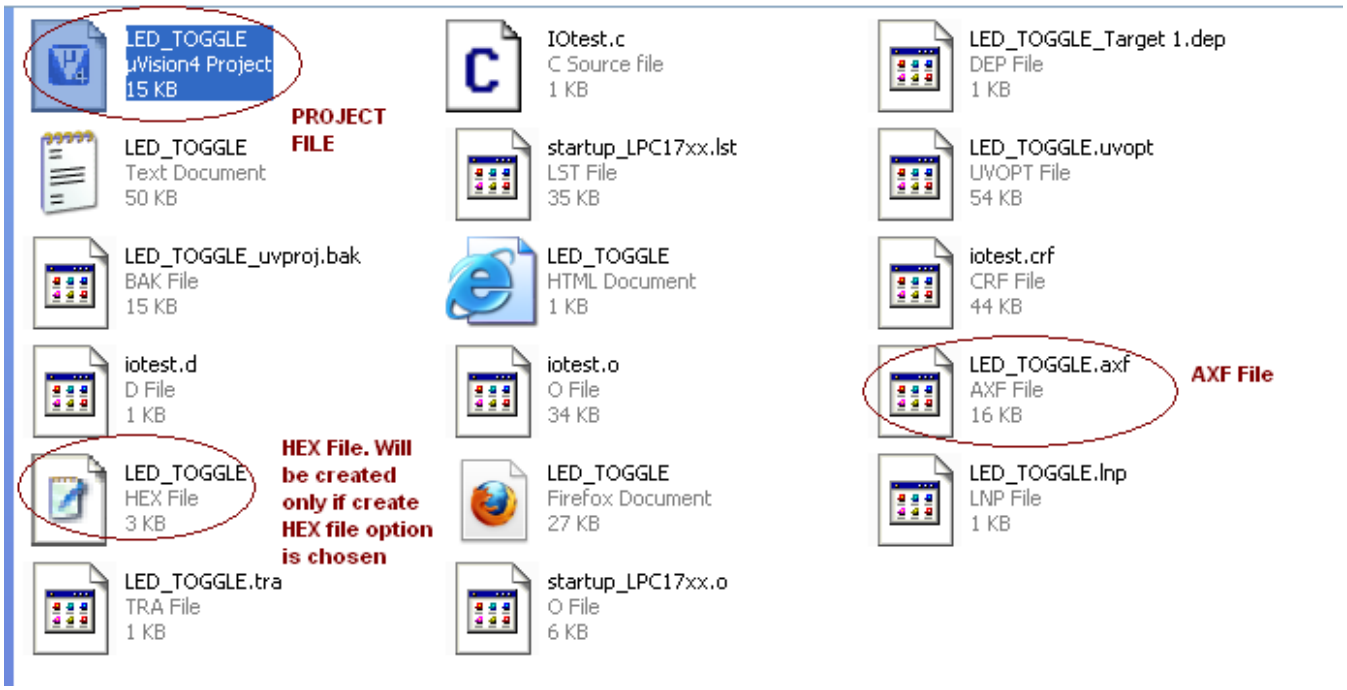
That seems easy enough, but how do I get this binary file? In order to get the binary file, first we have to compile the code. In order to compile the code you need have Kiel uVision4! Below steps would show you, how to create a binary file.

**Note: You should be familiar with Keil environment & programming. For more details on Keil programming, check tutorials on how to use Keil.**

### 3.1. COMPILING AND LOADING CODE ONTO AN USB BOOT LOADER ENABLED BOARD:

The axf file would be created once you compile the code. The axf file will be created in the folder where you have saved the project.

Note: You can change the location of .axf file storage if required whose details are not explained in this document.

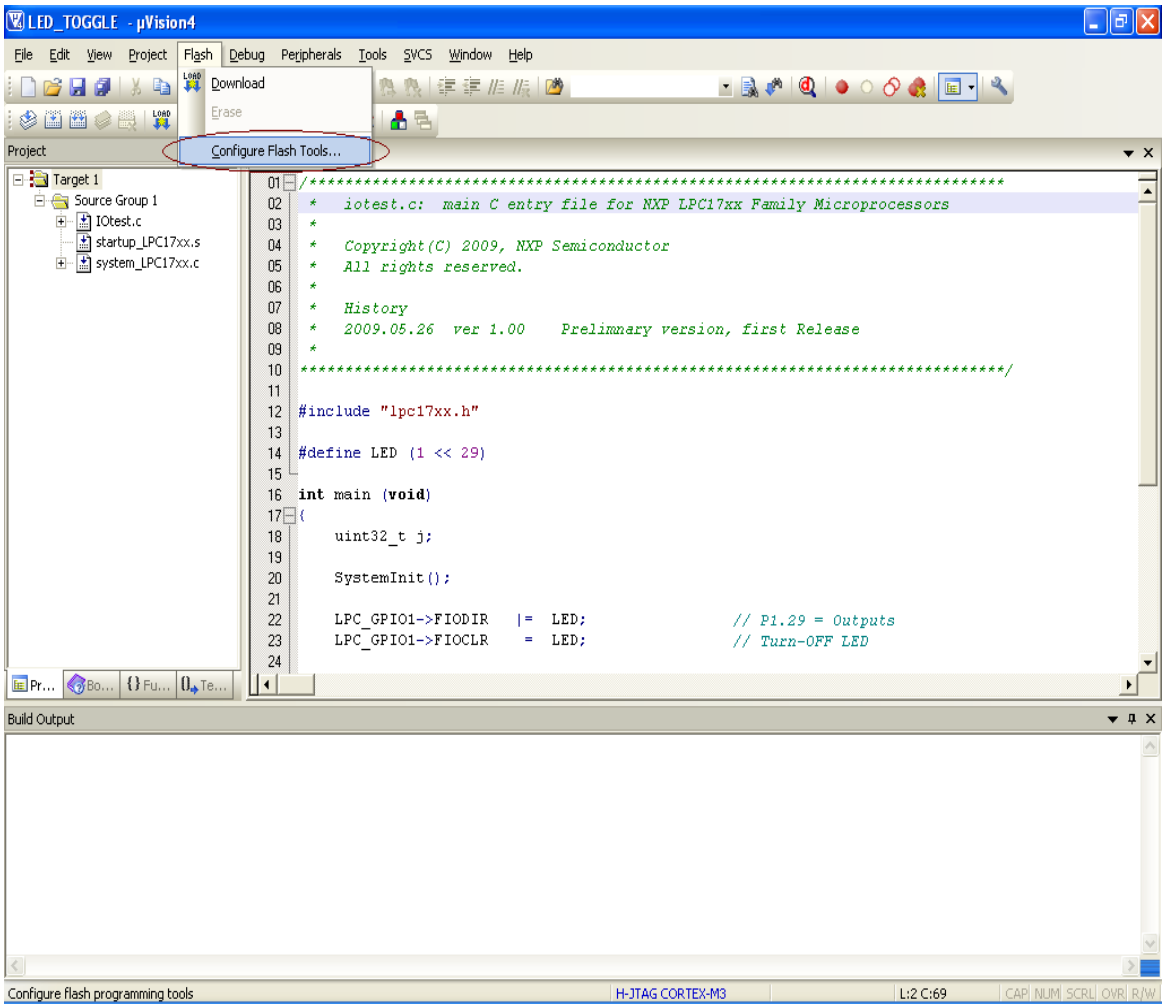



Now that you have created .axf file, how can it be converted into a binary file so that it can be loaded onto USB bootloader enabled LPC1768 board. The following documentation gives you the detail.

### 3.2. HOW TO CREATE BINARY FILE USING KEIL UVISION4:

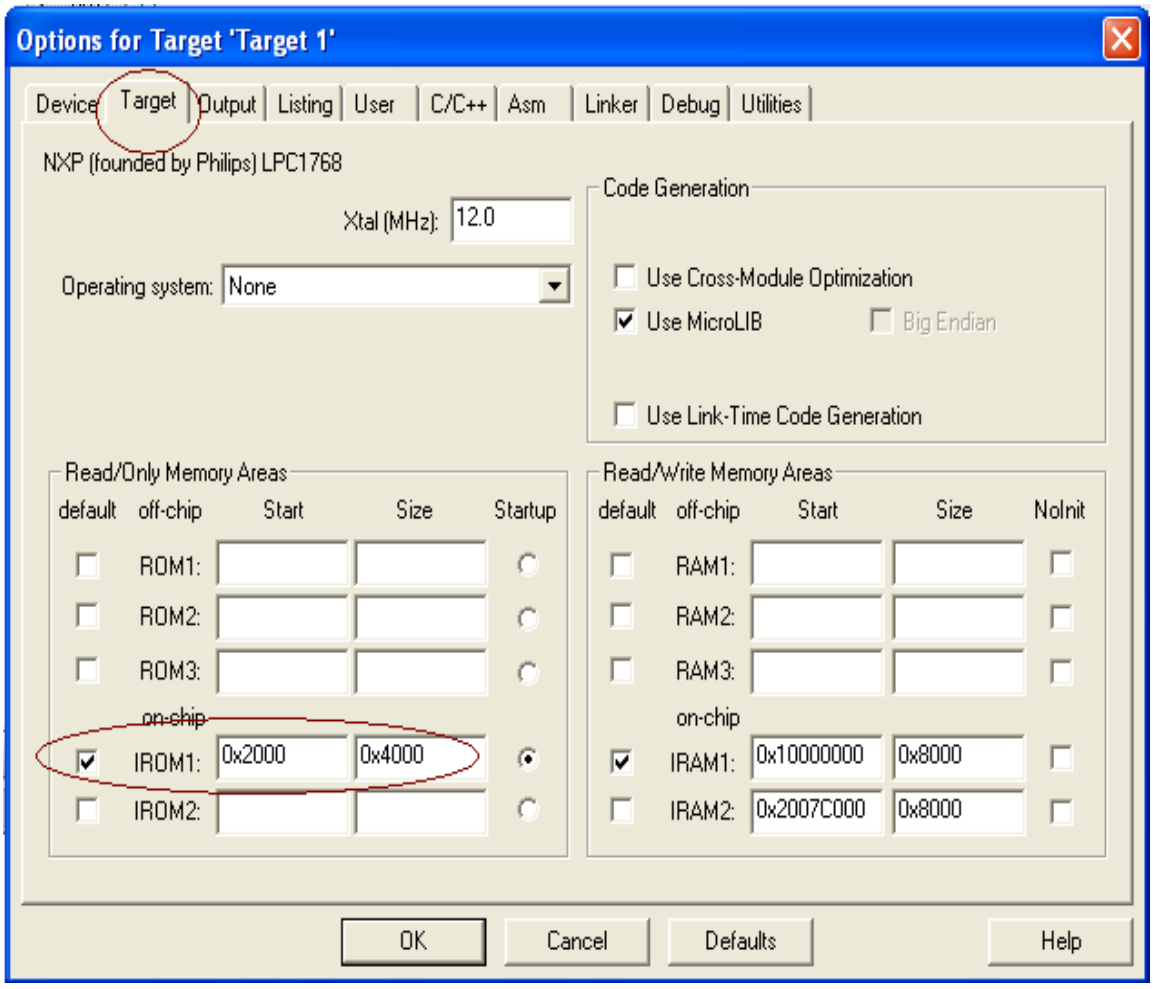
To create binary file follow the settings as shown in the images below.

#### 3.2.1. Click on Flash Menu → Go to Configure Flash Tools

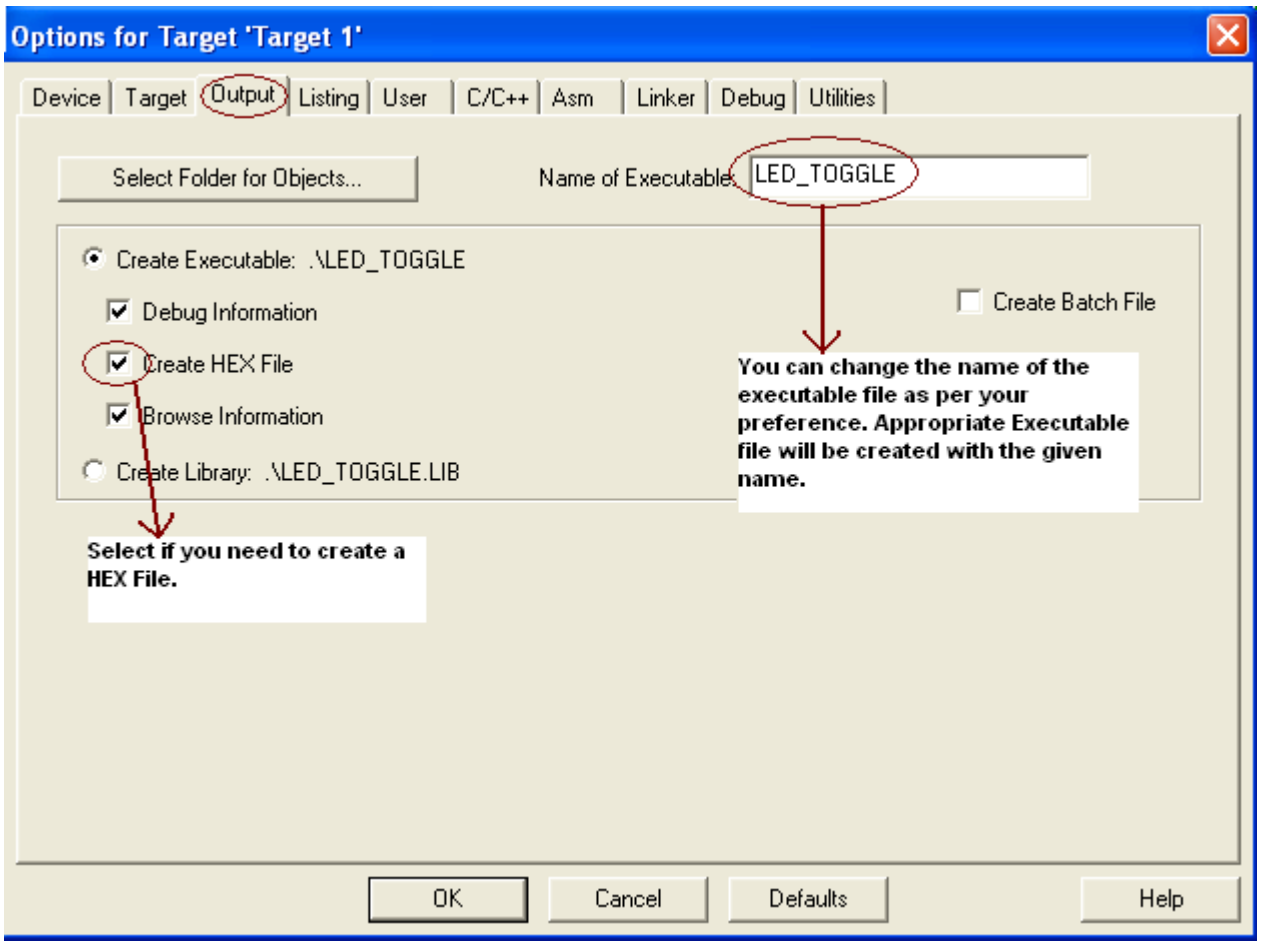


### 3.2.2. Click on Target Tab and type in IROM1 as shown in below image.

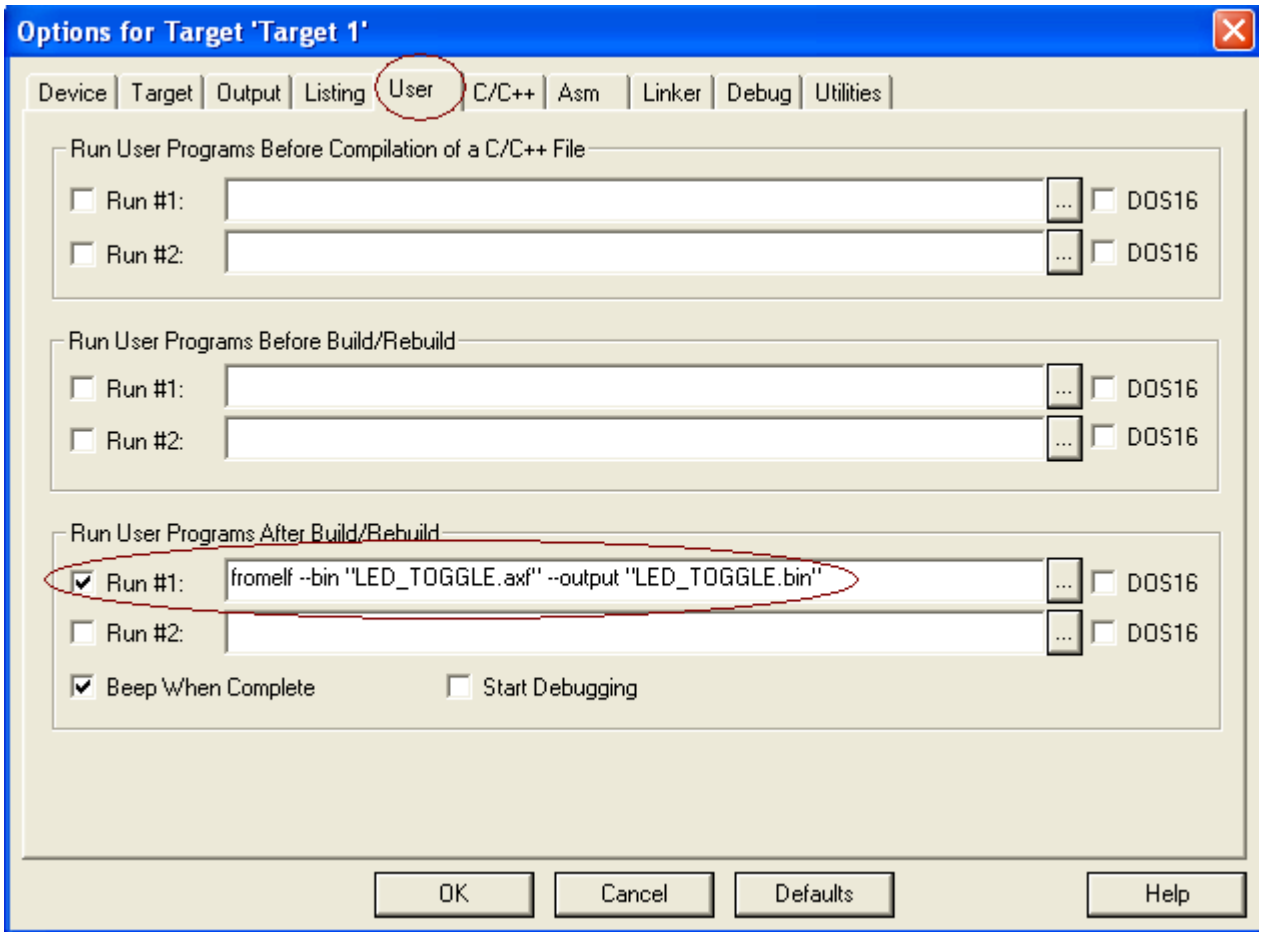
Change the start address of on-chip IROM1 memory to 0x2000 to 0x4000 as shown in fig. Below .This is the area where user program is placed and has to be run when the controller is reset.



**3.2.3. Click on Output Tab and make settings as required.**



**3.2.4. Click on User Tab and type in box next to RUN #1 as shown in below image.**

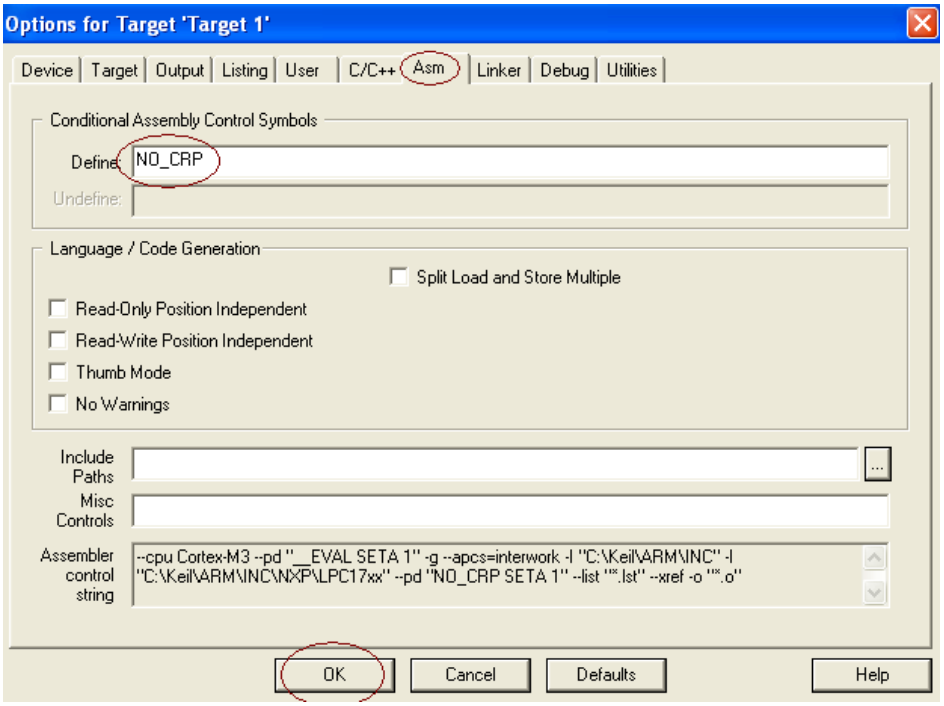



**You can also copy and paste the following**

**fromelf --bin "LED_TOGGLE.axf" --output "LED_TOGGLE.bin"**

Above image shows an example for LED_TOGGLE. If you are doing this for other code, replace LED_TOGGLE with name that you have given to your code/project.

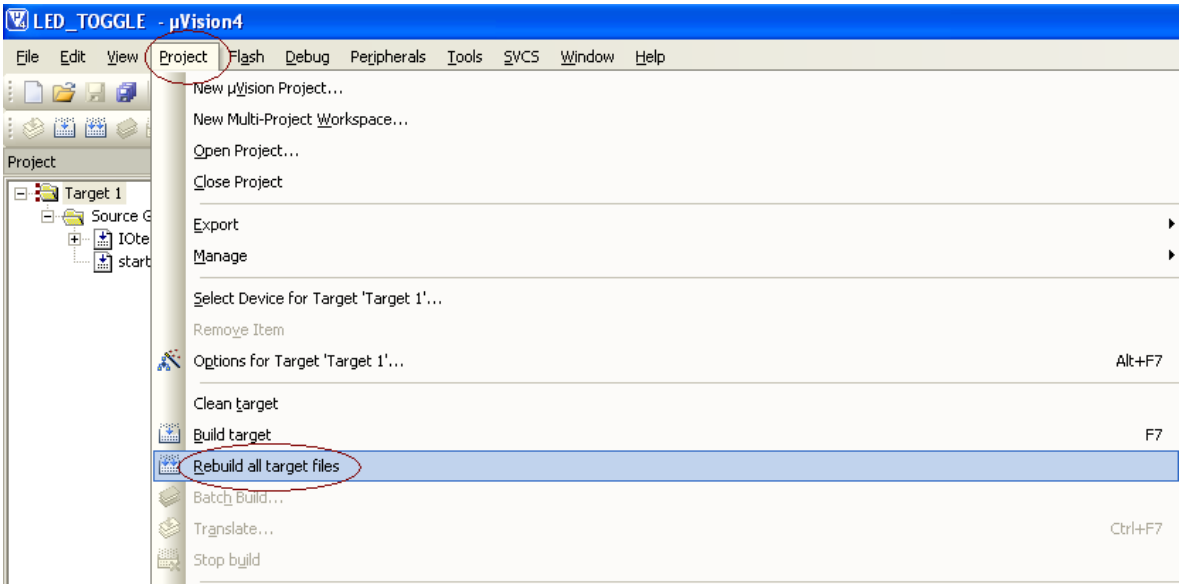
**3.2.5. Click on Asm Tab and make settings as below and then click OK button.**



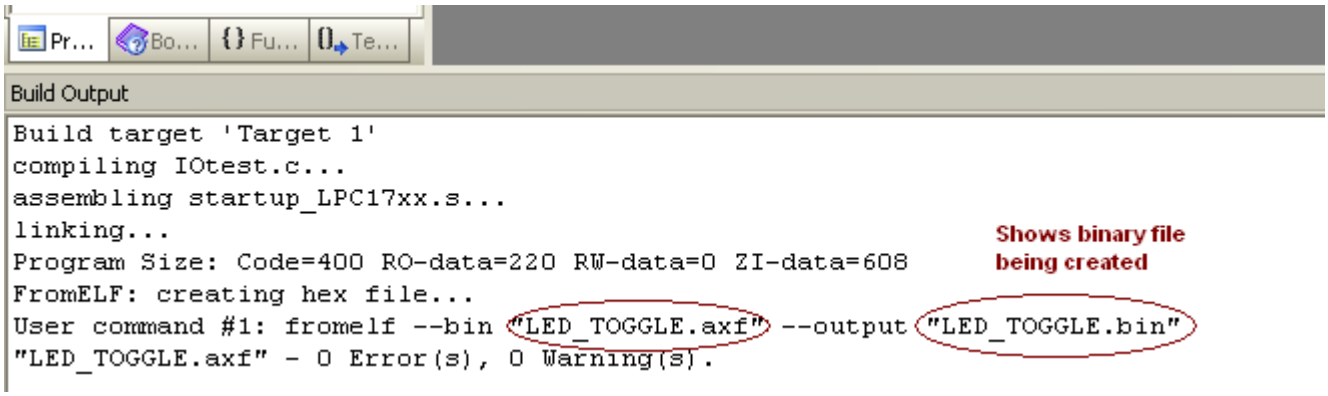

After all the settings made, press ok.

You need to rebuild the code for compilation. When the code is complied, binary files are created for your program that can be loaded on your LPC1768 Board.

Rebuilding the code



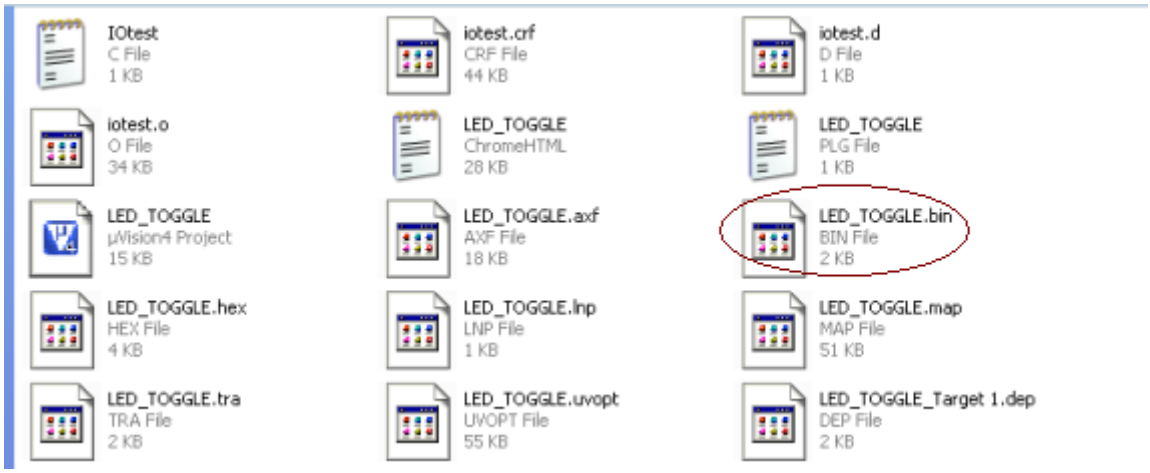

After compilation, you will see the following in build output window.




The bin File will be created in the same folder where project file is created.

Note: You can change the location of the bin file by changing the output location in Flash → Configure Flash Tools → Output → Select Folder for objects… The details of which are not explained in this document.

BIN File is created as shown




This bin file need to be loaded on the bootloader enabled board.

### 3.3. LOADING THE BIN FILE ONTO THE BOOTLOADER ENABLED BOARD:

After following the steps above, we would be ready with the following

**LPC1768 Board** (with the USB Boot loader firmware)

**USB cable (A to B Type).**

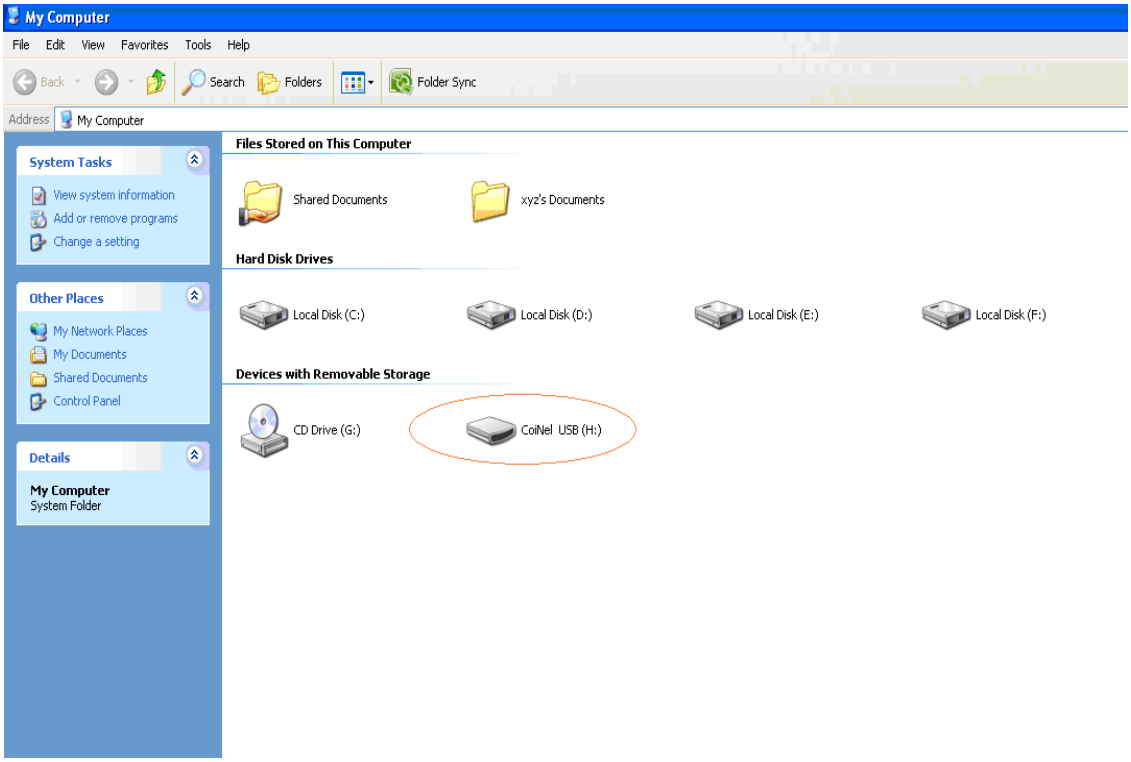
**Required .bin file for the project.**

Now we will see how we can load this bin file on the controller.

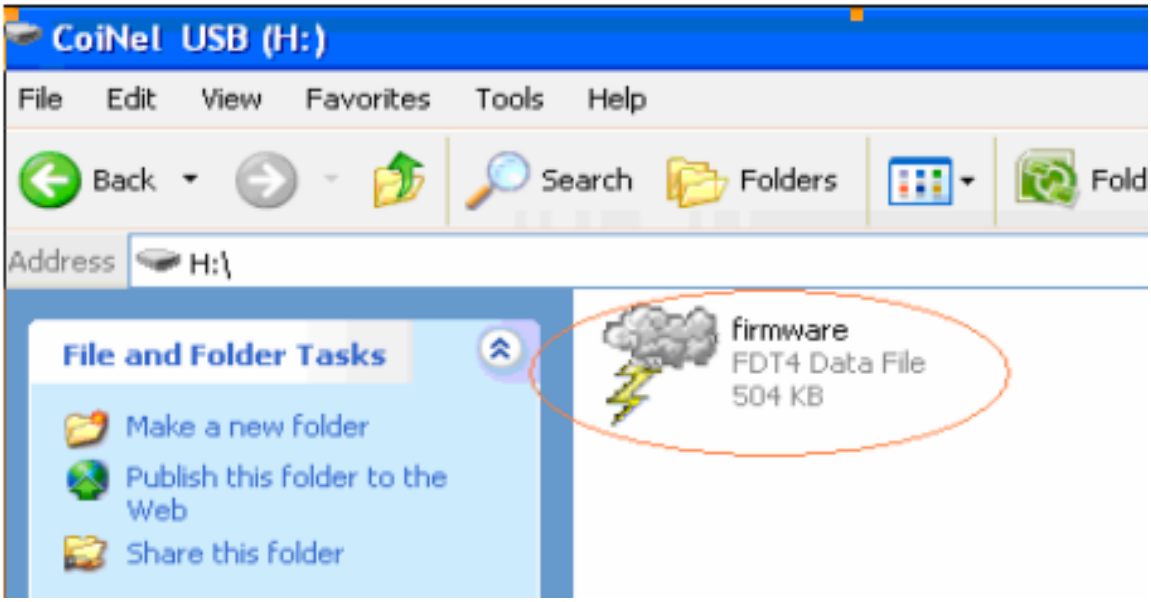
***Procedure*:**

- Plug in the USB cable to your computer and to the Board. Once the USB cable is plugged in, Board is powered up; a Removable Disk Drive will be available. This drive represents the memory space on the card.

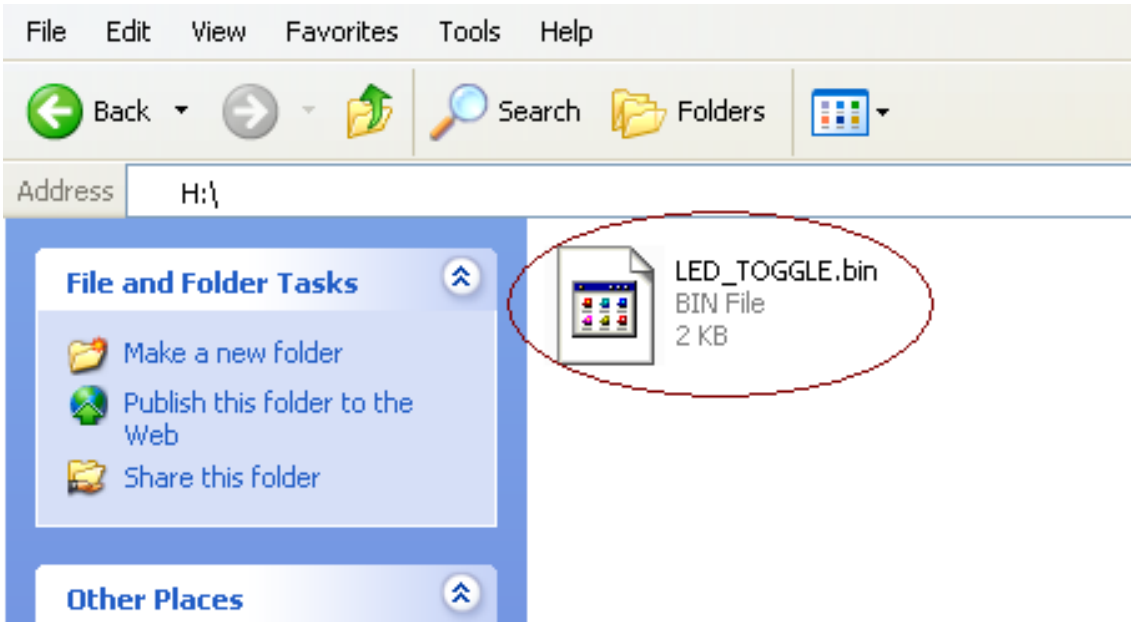
In the below image, you can notice, a drive named CoiNel USB.




- When you enter into this CoiNel USB drive, you could notice a default firmware file already present.



You need to delete this firmware file. *Copy and paste* the newly created project binary file into this drive. Below figure shows a newly created project binary file is pasted.




**Once the file has been copied to this new location, Press reset on your board, and your new code will be running.**

**Note:**

**In case, you want to dump a new program into LPC1768 Board, you need to connect port P1.25 to ground pin of the board and then press reset switch.**

**Doing this will again have the board detected as a removable disc (ie the board will exit from application mode and will switch back to bootloader mode). Follow the above procedure and copy/load the new bin file.**

**Note: Once programmed, make sure that the P1.25 pin is removed / not grounded.**

**For any technical discussion related to the product with our team and various users, visit and post your questions at**

**www.coineltech.com/forums**

## READER RESPONSE

It is our intention to provide you with the best documentation possible to ensure successful use of the product. If you wish to provide your comments on organization, clarity, subject matter, and ways in which our documentation can better serve you, please mail your comments to support@coineltech.com or call our Technical Publications Officer at (+91) 80-23154423.

Please list the following information, and use this outline to provide us with your comments about this Data Sheet.

1. How does this document meet your hardware and software development needs?
2. Do you find the organization of this data sheet easy to follow? If not, why?
3. What additions to the data sheet do you think would enhance the structure and subject?
4. What deletions from the data sheet could be made without affecting the overall usefulness?
5. Is there any incorrect or misleading information (what and where)?
6. How would you improve this document?
7. How would you improve our software, systems, and products?

## After-sale Service

We have special Technical Support Engineers to provide support and consultation in forms of telephone, E-mail and so on.

TEL: +91-80-23154423

Technical Support E-mail: support@coineltech.com

## Disclaimers

CoiNel Technology Solutions LLP. provides the enclosed document under the following conditions:

This document is intended for use for ENGINEERING DEVELOPMENT, DEMONSTRATION and EDUCATION OR EVALUATION PURPOSES ONLY. As such, the document being provided are not intended to be complete in terms of required design, marketing-, and/or related protective considerations,

The user assumes all responsibility and liability for proper usage of the document. Further, the user indemnifies CoiNel Technology Solutions LLP from all claims arising from the handling or use of the documents. EXCEPT TO THE EXTENT OF THE INDEMNITY SET FORTH ABOVE, NEITHER PARTY SHALL BE LIABLE TO THE OTHER FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES.

No license is granted under any patent right or other intellectual property right of CoiNel Technology Solutions LLP covering or relating to any machine, process, or combination in which such CoiNel Technology Solutions LLP products or services might be or are used.

Information in this document is believed to be reliable and accurate. However, CoiNel Technology Solutions LLP does not give any representations or warranties, expressed or implied, as to the completeness or accuracy of such information and shall have no liability for the consequences of use of such information.

CoiNel Technology Solutions LLP reserves the right to make changes to information published in this document, at any time and without notice, including without limitation specifications and product descriptions. This document replaces and supercedes all information supplied prior to the publication hereof.

### Trademarks

All referenced trademarks, product names, brands and service names are the property of their respective owners.