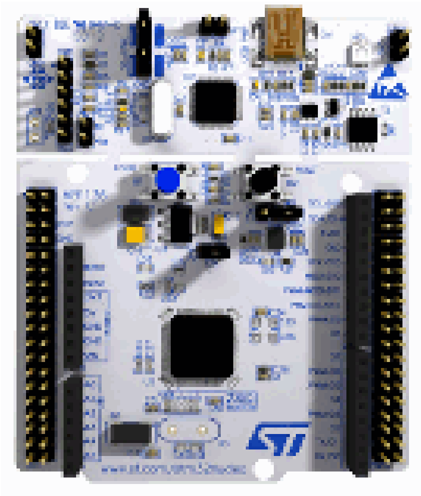
How to Implement STM32 into B4R

Revised 11th August 2020

If you want more FLASH memory space (say 128K flash), need more RAM (how about 20K), would like your program to run a little faster, or simply want to play around with a different micro-controller family, then have a go at this.

The STM32F103xx series of microcontrollers are manufactured by ST Microelectrinics. There are many boards/modules available employing one of the many STM32F103 microcontrollers available in the family. I chose the NUCLEO evaluation boards as the documentation indicated it uses compatible Arduino pin headers. It is a two in one board. The lower section of the PCB is the STM32 microcontroller evaluation board while the top section is a build-in ST-Link v2.1 programming interface via a single USB connection.



There are many NUCLEO evaluation boards for STM32 microcontrollers. I chose to implement the NUCLEO F103RB. This board can be purchased online for about $20 to $30.

http://www.ebay.com.au/sch/i.html?_odkw=ga227911+nucleo&_osacat=0&_from=R40&_trksid=p2045573.m570.l1313.TR0.TRC0.H0.Xnucleo+f103rbt6.TRS0&_nkw=nucleo+f103rbt6&_sacat=0

https://www.element14.com/community/docs/DOC-67145/l/stmicroelectronics-stm32-mcu-nucleo-boards

Your will need a usb to mini-usb cable to connect it to your PC. The Nucleo board also get power via this lead.

http://www.ebay.com.au/sch/i.html?_odkw=nucleo+f103rbt6&_osacat=0&_from=R40&_trksid=p2045573.m570.l1313.TR0.TRC0.H0.Xpremium+15ft+usb+2.0+A+male+mini+B+pin+male+plug+leads.TRS0&_nkw=premium+15ft+usb+2.0+A+male+mini+B+pin+male+plug+leads&_sacat=0

Now you have the hardware lets get going.

Arduino IDE

First you need to download and install Arduino IDE minimum version 1.6.9 It just won't work on earlier versions.

https://www.arduino.cc/en/Main/Software

Ok, now we need to run the Arduino IDE development suite. Select from the menu TOOLS/Board option then select the "Board Manager" from the top of the list.

Once it has updated the list of supported boards, click on the

"Arduino SAM Boards (32-bit ARM Cortex-M3)"

Download/install this board library.

Although this library is not for STM32 microcontrollers, we need it installed to get the ARM compiler into the Arduino IDE environment.

STM32 Libraries

The STM32 libraries have actually been written/edited/created and maintained by Roger Clarke and the guys at

http://www.stm32duino.com/

Without their devotion to this "Arduino for STM32" project, none of this would be possible.

As the STM32 libraries are not available from within Arduino's own Board Manager, we need to download them directly from

https://github.com/rogerclarkmelbourne/Arduino_STM32

Click on the button <CODE> then click DOWNLOAD ZIP.

Once the zip file is downloaded onto your PC, go to your MY DOCUMENTS folder. Go into the "Arduino" folder. You will see a folder "libraries" and as you probably don't have it, create another folder called "hardware". Now you have both "libraries" and "hardware" folders in the "Arduino" folder.

Now go into that "hardware" folder. Extract the contents of the downloaded zip and paste it into this "hardware" folder. You should see a single folder "Arduino_STM32-master". All the libraries are inside it. Now rename this folder to just "Arduino_STM32". Don't need the "-master" suffix.

ST-Link Drivers

You need to download and install the ST-Link drivers. The drivers provide communication between the PC and the ST-Link programmer built onto the top section of the NUCLEO evaluation board. Click on this link.

https://developer.mbed.org/teams/ST/wiki/ST-Link-Driver

Follow the instructions, click the orange text “Download the latest ST-Link v2 driver”. This takes you to STSW-LINK009 web page. Go down the page to the section titled GET SOFTWARE. This section shows a table containing Part Number STSW-LINK009, move to the right column labelled “Order from ST” and click on the button “Get Software”. You will need to ACCEPT the terms and conditions then you progress to the GET SOFTWARE web page. You can chose to create a new Registration or fill in your name and email address to just download the driver. Clicking the button “DOWNLOAD” sends a request to ST. They will send you an email containing the actual download link for the driver archive file. This may take several minutes (yeh right) to receive the email from ST.

The downloaded archive file contains two installers.

dpinst_amd64.exe for 64bit windows and

dpinst_x86.exe for 32bit windows.

Install the right one for your PC.

Now when you connect the Nucleo evaluation board to the PC with the USB cable you should get two devices show up on your PC.

First, a new drive called “Nucleo_F103RB” will be mounted in your PC’s drive list, and

Secondly, a virtual serial Comm Port. (for me it came up as Com3)

In fact a third device is also created to facilitate the programming of the Nucleo board via the USB. For now we will consider it as just ”ST-Link” programmer.

The drive is not really useful for us. It is used by ST’s own “MBED” development suite for dragging and dropping the output BIN file onto. It is then automatically uploaded into the microcontroller and run.

The virtual serial com port is an extention of the second hardware serial port (USART:1) of the microcontroller. From the B4R environment we consider this as another PC com port.

ST-Link Utility

A software tool called “ST-Link Utility” is also available from ST. It can connect, communicate and program the Nucleo board. Although it is not required for using B4R with STM32 I did end up installing it.

I had trouble getting the ST-Link to be seen by the Arduino IDE. I used the “ST-Link/Update-Firmware” menu option of the “ST-Link Utility” to do just that. This seemed to get thing talking.

Click here to download it.

http://stm32-st-link-utility.software.informer.com/download/

ST-Link Utility is also a great tool for just debugging the connectivity of things. It’s useful to have.

Changes to STM32 Library

As the STM32 microcontrollers have different and extended features over the usual UNO (avr) microcontrollers, the STM32 libraries have been adapted to best suit the requirements of a coder. This means that it is 100% compatible with the Arduino IDE environment, but 95% compatible with existing project’s C source code. B4R generates a compatible INO project which it passes to the “Arduino Builder”. Arduino Builder then compiles it and the resultant BIN output file (not HEX) is then uploaded to the microcontroller board.

This 95% compatibility causes issues (compile errors) for B4R. As such, some minor changes need to be made to the STM32 library source files. (Note: This is now simplified due to the newer versions of STM32 library and B4R).

Remember, we installed these earlier in
  ..//My Documents/Arduino/hardware/

-------------------Change 1---

To get the Serial class and LOG display operational in B4R, I needed to swap the Hardware Serial ports used for the "SERIAL" and "SERIAL1" objects. The STM32F103RB has 3 hardware serial ports on the chip. The lines to be changed are defined right at the end this file.
  ..//Arduino_STM32/STM32/STM32F1/variants/nucleo_f103rb/board.cpp

It now looks like this.

```
DEFINE_HWSERIAL(Serial1, 2);   //note the order of 2,3 then 1
DEFINE_HWSERIAL(Serial2, 3);
DEFINE_HWSERIAL(Serial3, 1);
```

Changes to B4R

Some more changes are also required to not just the set up of the B4R environment but also to the underlying source files used within B4R itself. You need to find the install location of the B4R application and its libraries. Probably under
  ..//Program Files/Anywhere Software/B4R/

---------------------Change 2---

You also need to extend the process timeout in B4R so that the compilation results (from Arduino Builder) get displayed in the compilation dialog box.

Run up B4R. Make the following change to its settings.

Menu: Tools/IDE-Options/Configure-Process-Timeout

Set it to 45 seconds.

B4R - Select Board Type

Run up B4R. Select menu TOOLS/BOARD-SELECTOR.

In the “Board Type” drop down box (it’s a long way down the list) select

STM Nucleo F103RB (STLink) (device_variant=NucleoF103_HSI)

In the “Serial Port” drop down box select

COMx STMicroelectronics STLink Virtual Com Port

Baud Rate

115200.

Logs

IDE

Click the <OK> button to save these settings.

Ready to Run

If everything has gone to plan you should be ready to compile and run your first B4R project on the STM32F103RB Nucleo evaluation board. I started with the B4R Blink example. The green test LED (just below the black reset button) on the Nucleo board is connected to Arduino pin 13. The output to the LOG window will also display fine.

Good Luck and have fun.

Other STM32 F103 boards

I have also been implementing other STM32 F103 boards for programming from B4R with reasonable success. Both of the boards below are STM32F103C8T6 and sell for about $6 each. Similar specs to the NUCLEO F103RB board but they require either a separate ST-LINK programmer dongle or they can be programmed using a serial link such as a FTDI1232 USB to serial module.

I hope to publish additional information for these boards at a later time.

Known Issues

Although everything I have tested so far seems to work fine, other changes may still need to be made as further libraries, classes and hardware support are confirmed.

I have only made changes to support the STM32F1 series. The Arduino_STM32 libraries appear to support the other microcontrollers in the ST family. F3 and F4 devices are even more powerful should others want to play around getting them going.

My only outstanding issue at this time is that compiling a B4R project for STM32 creates a very large output BIN file. Around 70K. Even if it is a small (or nothing) project. This is not a problem for the Nucleo F103RB evaluation board as it supports 128K of Flash memory. Still plenty of room for play. In the case of the STM32F103C8T6 boards described above it is a bit tricky. The F103C8 is only a 64K flash device so the B4R output BIN file never fits (or does it). On the www.stm32duino.com forum they discuss that the F103C8 devices (although rated 64K) usually contain 128K flash anyway (all manufactured from the same silicon die). My testing together with others on the forum have confirmed this fact. So, the fix is to declare the device as a F103CB type being 128K flash, then the BIN file uploads and executes fine.

I broke down the INO project generated by B4R to determine the actual cause of this over size BIN file. If I make a simple blink project in C in the Arduino IDE it is only 5K in size. The difference is that the B4R inside its core C source code uses class instantiation to manage the PollerNode list as well as many other class structures. The NEW operator used to declare an object of a class, adds about 64K. Oddly, if I compile to Arduino UNO its output memory requirement is only 5K (class instantiation only adds about 600 bytes). The problem is not B4R. The guys at www.stm32duino.com indicated its caused by the inclusion of a great deal of support code once a classs instantiation is required (even when the class itself is empty). I really don't understand this. My knowledge of C is limited so I have chosen to moved on as it works anyway.

General References and Resources

Arduino for STM32 – Developers forum

http://www.stm32duino.com/

Roger Clarke's Gut hub (Arduino_STM32 libraries)

https://github.com/rogerclarkmelbourne/Arduino_STM32

STM32F103RB Nucleo – User Manual

http://www.st.com/content/ccc/resource/technical/document/user_manual/98/2e/fa/4b/e0/82/43/b7/DM00105823.pdf/files/DM00105823.pdf/jcr:content/translations/en.DM00105823.pdf

STM32F103RB microcontroller – Data Sheet

http://www.st.com/content/ccc/resource/technical/document/datasheet/33/d4/6f/1d/df/0b/4c/6d/CD00161566.pdf/files/CD00161566.pdf/jcr:content/translations/en.CD00161566.pdf

ST-Link Utility User Manual

http://www.st.com/content/ccc/resource/technical/document/user_manual/e6/10/d8/80/d6/1d/4a/f2/CD00262073.pdf/files/CD00262073.pdf/jcr:content/translations/en.CD00262073.pdf

YouTube - Setup Arduino IDE for STM32

https://youtu.be/-zwGnytGT8M

YouTube - First LED Blink STM32 via Arduino

https://youtu.be/Ze6q6NidS5w

WebPage - STM32F103C8T6 Programming with ST-Link v2

https://github.com/rogerclarkmelbourne/Arduino_STM32/wiki/Programming-an-STM32F103XXX-with-a-generic-%22ST-Link-V2%22-programmer-from-Linux

------------ end ------------