Technical Workshop

Academic High Altitude Conference

Ethan Harstad Matthew Nelson

Stratospheric Ballooning Association

June 23-24, 2014

Outline

1. Getting Started
   - What is mbed?
   - mbed.org
   - Nucleo Development Board
2. The Bare Minimum
3. Your First Program
4. Taking Control
5. Talking to mbed
6. Writing Modular Code

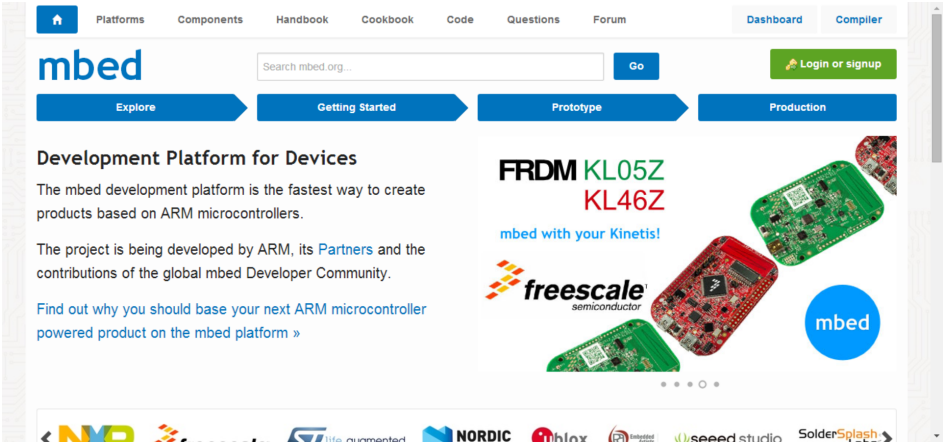
What is mbed?

The mbed platform is a collection of open source hardware and software to allow rapid ARM based prototyping

- Professional online compiler lets you work from any computer
- Integrated version control system lets you easily find and use libraries
- CMSIS based APIs let you work high level or bare metal
- Hardware abstraction layer insulates your application from hardware changes

Essentially a high performance Arduino with highly integrated tools to save you time!

Register on mbed



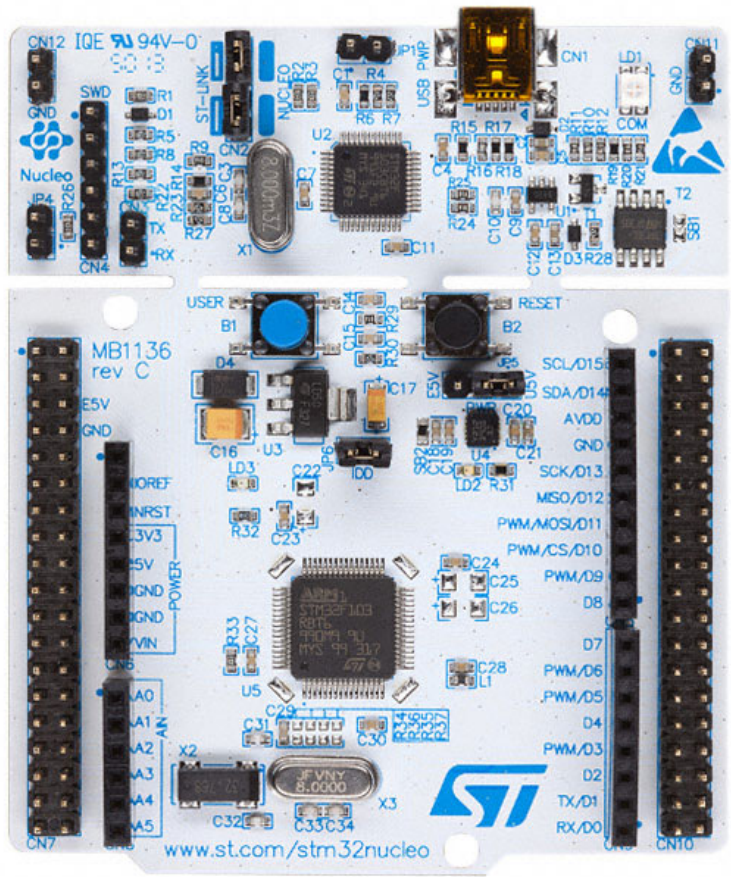
1. Navigate to `http://www.mbed.org`
2. Click the green "Login or signup" button
3. Click the "Signup" button
4. Follow the prompts
5. Confirm your e-mail address

Everyone should have an mbed account. You can create a team to share code between members of your organization.

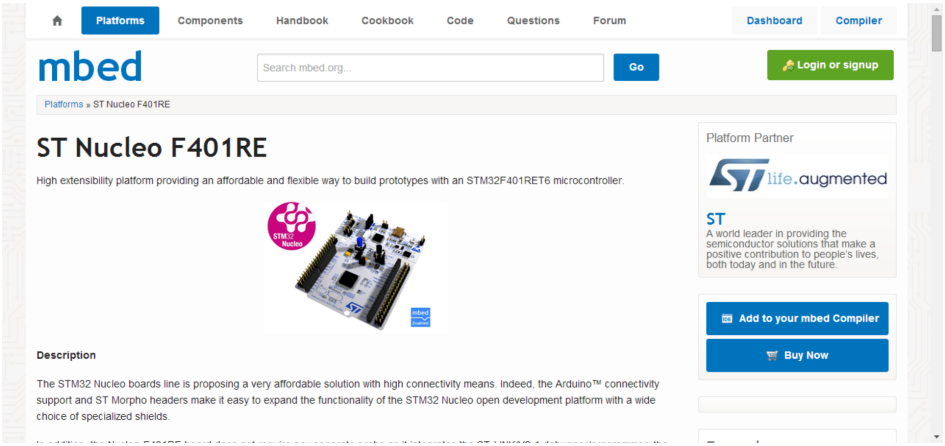
Nucleo Development Board

The Nucleo development board combines a USB programmer with a powerful STM32 processor and Arduino compatible headers

- ARM Cortex-M4 with FPU at 84 MHz
- 512 KBytes of flash memory
- 12 bit ADC at 2.4 Msps with up to 10 channels
- Up 3xUART, 3xI2C, 4xSPI interfaces



Add Nucleo to Your Account



1. Connect your Nucleo to your computer
2. Open the external drive the connects
3. Double click the mbed.htm file
4. Click "Add to your mbed Compiler"

Note

You only need to do this once per account!

Install Drivers

Outline

1 Getting Started

2 The Bare Minimum
- Creating a Program
- Importing a Library
- Program Structure

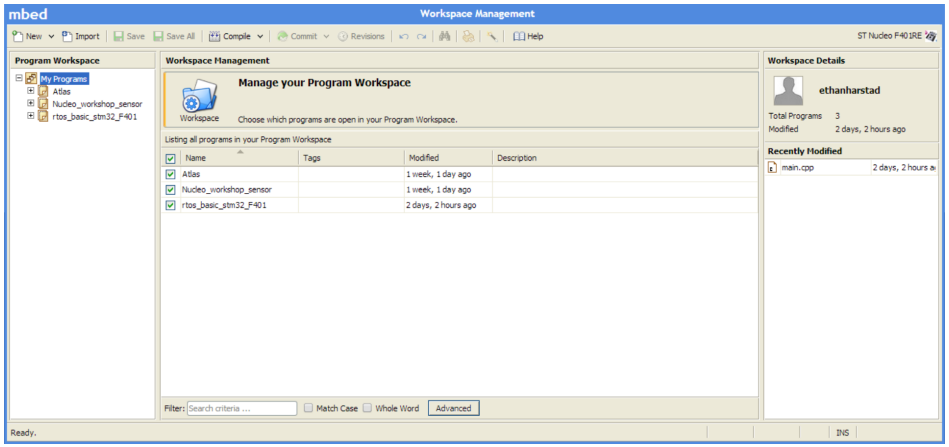
3 Your First Program

4 Taking Control

5 Talking to mbed

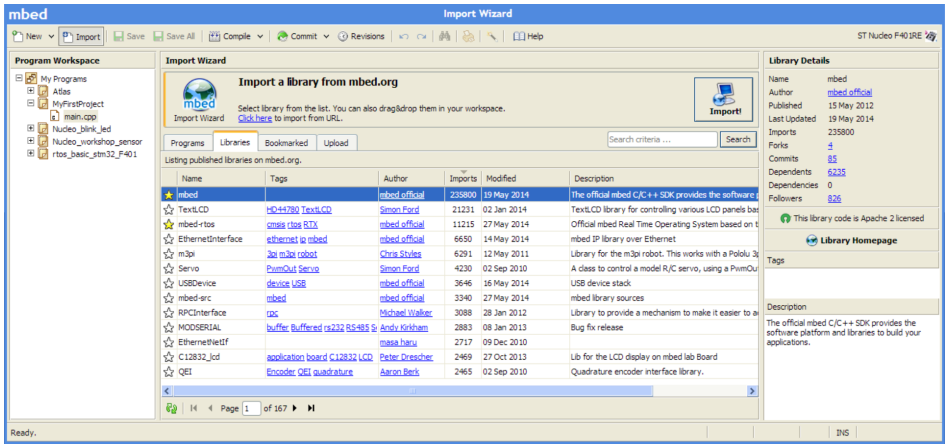
6 Writing Modular Code

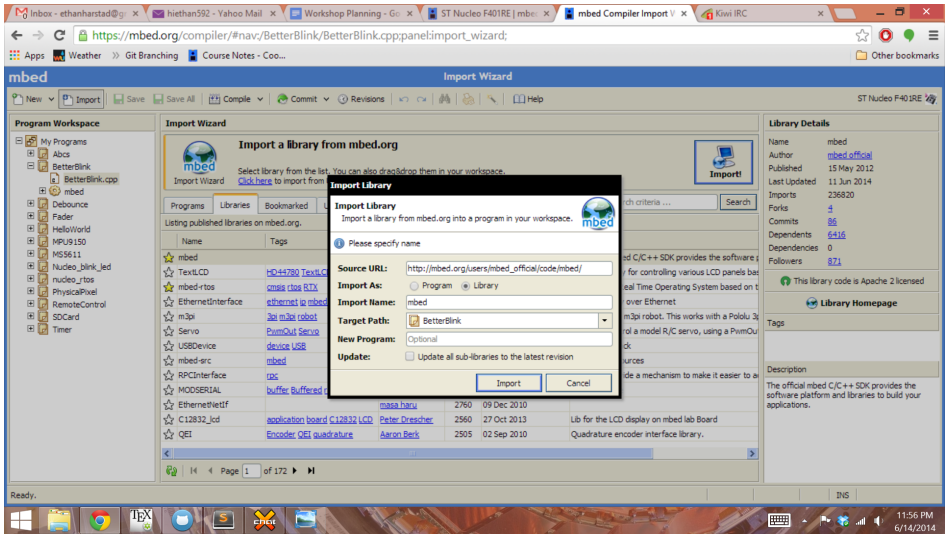
Creating a Program



1. Navigate to the mbed homepage and click the "Compiler" button
2. Click the "New" button and select "New Program"
3. Change the Template field to "Empty Program"
4. Give your program a name and click "OK"

Importing a Library





1. Click the "Import" button
2. Search for the "mbed" library
3. Select the library
4. Click the "Import!" button
5. Make sure the Target Path is your project root
6. Click "Import" one last time

Creating a New File

1. Click the root directory of your project to select it
2. Click the arrow next to "New" and select "New File"
3. Name the file "main.cpp" and click "OK"

Warning

Be sure to select the folder you want your file in before creating it!

You can name this file anything but it must have the .cpp extension. It is suggested that the main file be named "main" or the same as your project name (important later on).

Program Structure

```
1 /* Includes */
2 #include "mbed.h"
3
4 /* Global Variable
5    Declarations    */
6
7 /* Main Function */
8 int main() {
9   // Program code
10 }
```

Listing 1: main.cpp

- Line 2 includes the mbed library, **every** mbed program needs this.
- Line 8 is the main function, this is the entry point into your program. **Every** program needs a main function.
- Line 10 is a comment, everything after // is ignored.
- Line 4-5 is a multi line comment, everything between /* */ is also ignored.

Outline

1. Getting Started
2. The Bare Minimum
3. Your First Program
   - While Loops
   - Digital Output
   - Waiting
   - Compiling
4. Taking Control
5. Talking to mbed
6. Writing Modular Code

While Loops

```
1 while(conditional) {
2   // Code to execute when true
3 }
```

While loops executes the code contained within them while their conditional statement is true.

Example

A main function should (almost) never exit:

```
1 int main() {
2   while(true) {
3     // Main loop code, runs forever
4   }
5 }
```

Your Program

```
1  /* Includes */
2  #include "mbed.h"
3
4  /* Global Variable
5     Declarations     */
6
7  /* Main Function */
8  int main() {
9    while(true) {
10     // Program code
11   }
12 }
```

Listing 2: main.cpp

- Add an infinite while loop to your main function to prevent your program from ending.

Digital Output

DigitalOut constructor:

```
DigitalOut(PinName pin)
```

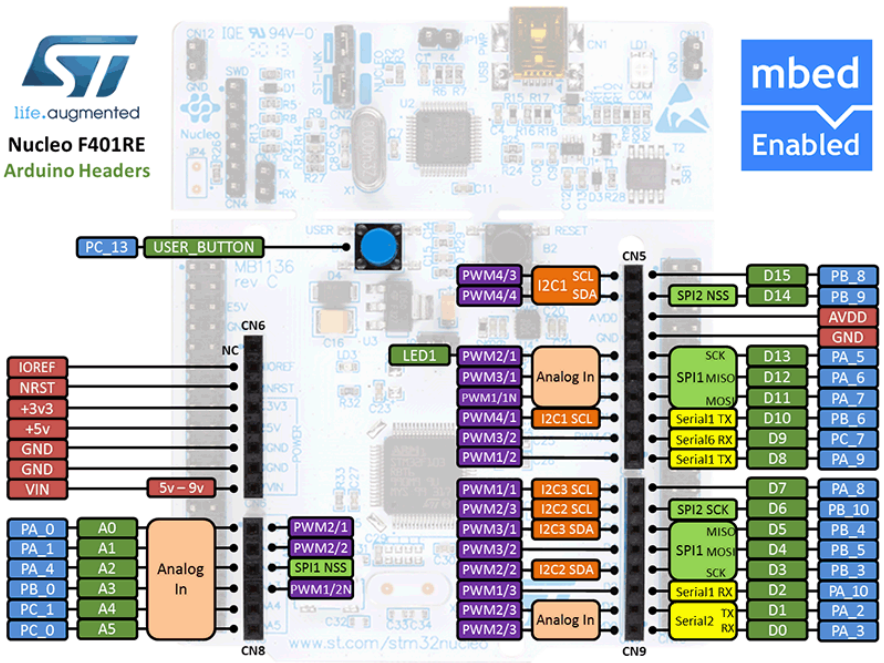
It creates an object attached to the given pin. Anytime you see PinName, use a name from the images on the next slide.
You can assign a value to the object using the equals sign. 1 turns the pin on while a 0 turns the pin off.

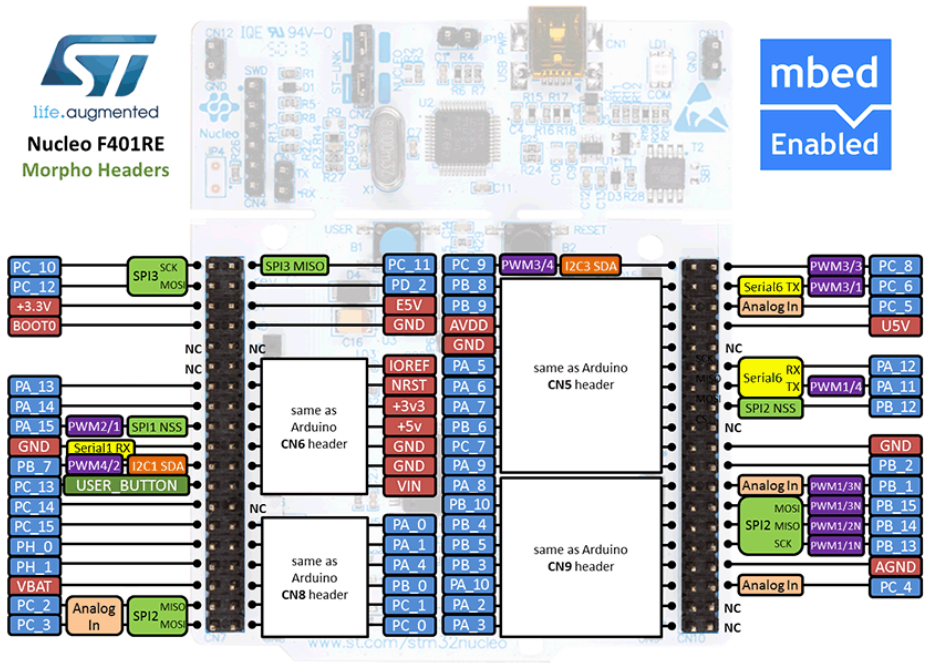
Example

Attach a DigitalOut to the LED1 pin on the Nucleo and turn it on:

```
DigitalOut led(LED1);
led = 1;
```

Nucleo Pin Names





Warning

You can only use the labels in blue and green!

Full size versions are available at

`https://mbed.org/platforms/ST-Nucleo-F401RE/`

aerodyne labs

Your Program

```
1  /* Includes */
2  #include "mbed.h"
3
4  /* Global Variable
5     Declarations     */
6  DigitalOut led(LED1);
7
8  /* Main Function */
9  int main() {
10   while(true) {
11     led = 1;   // Turn LED on
12     led = 0;   // Turn LED off
13   }
14 }
```

Listing 3: main.cpp

- Declare a global DigitalOut object
- Turn the output on and off in your main loop

Note

The LED won't seem to be flashing, but it actually is at about 42 MHz, much faster than your eye.

Waiting

There are three statements that can slow down execution:

```
void wait(float s);
void wait_ms(int ms);
void wait_us(int us);
```

All three will pause execution for the amount of time specified. Use these statements any time you need a controlled delay.

Notice

Wait and other block statements can have some unintended side effects. This will be demonstrated later.

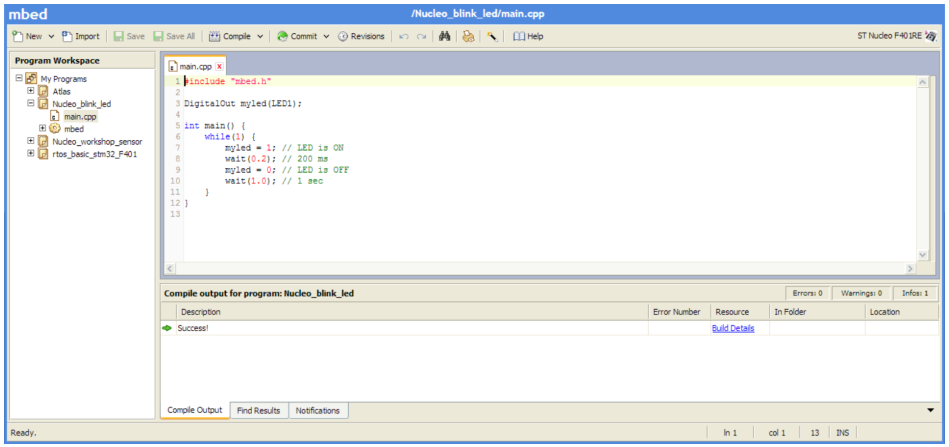
Your Program

```
1  /* Includes */
2  #include "mbed.h"
3
4  /* Global Variable
5     Declarations     */
6  DigitalOut led(LED1);
7
8  /* Main Function */
9  int main() {
10   while(true) {
11     led = 1;     // Turn LED on
12     wait(0.2);   // Wait a bit
13     led = 0;     // Turn LED off
14     wait(0.8);   // Wait longer
15   }
16 }
```

Listing 4: main.cpp

- Add a wait statement after each write to your output
- You should now be able to see your LED flashing
- Try making your own patterns!

Compiling



1. Click "Compile" (Ctrl D) to compile your program
2. A *.bin file will be downloaded
3. Move the downloaded file to the Nucleo drive
4. The Nucleo will flash red and green while programming
5. When the lights stop, your program has started successfully!

Tip

Set your browsers download location to the Nucleo to save time

You can also click "Build Only" (Ctrl B) to simply test your code

Outline

1. Getting Started
2. The Bare Minimum
3. Your First Program
4. Taking Control
   - Variables
   - Digital Input
   - Conditional Statements
5. Talking to mbed
6. Writing Modular Code

Outline

1 Getting Started

2 The Bare Minimum

3 Your First Program

4 Taking Control

5 Talking to mbed
- Serial Ports
- Switch/Case Statements
- Functions
- For Loops

6 Writing Modular Code

Outline

1 Getting Started
2 The Bare Minimum
3 Your First Program
4 Taking Control
5 Talking to mbed
6 Writing Modular Code
- Analog Input
- PWM Output
- Classes

Outline

1 mbed Platform

What is mbed?

Outline

1 Open Session

What is mbed?