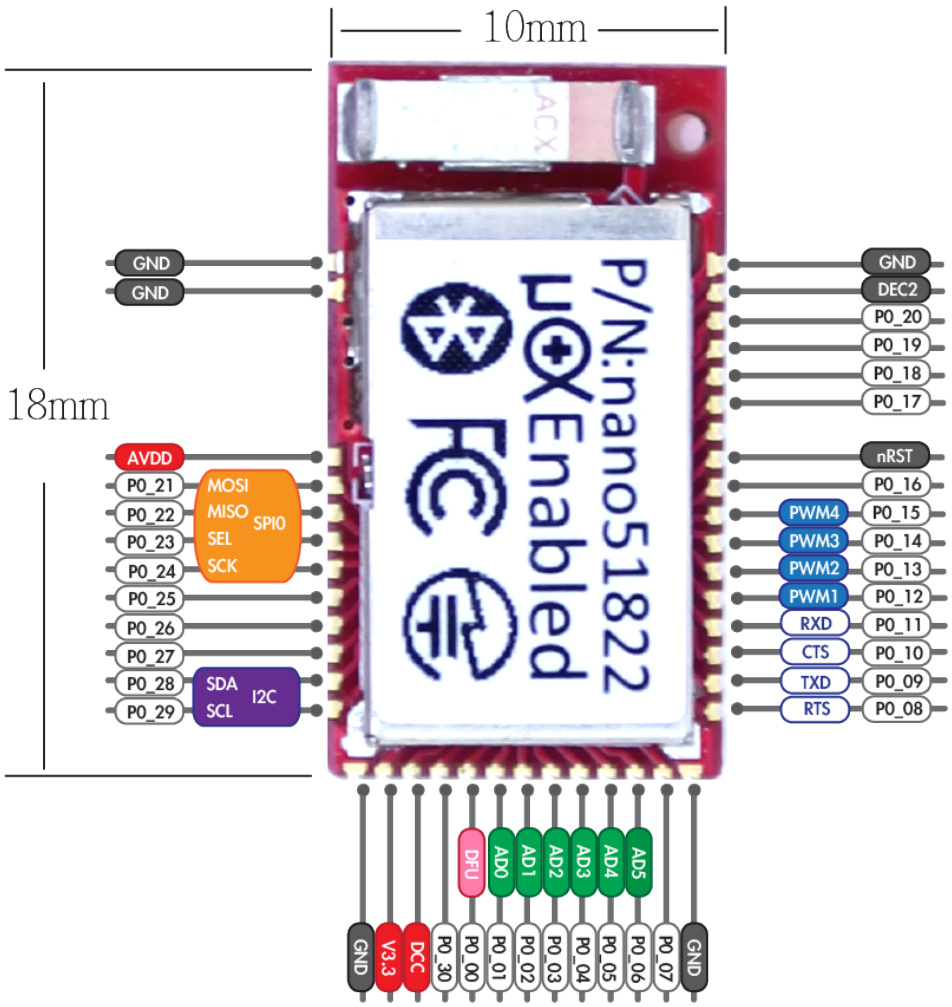
Getting Started with uCXpresso.NRF



2014/10/21

Beta-1b

History:

2014/10/20 First Edition

Overview

First of all, an espresso is necessary before the getting started.

The uCXpresso framework have to work on the LPCXpresso IDE, you need to install the LPCXpresso IDE in your system first. The LPCXpresso IDE are available for Windows, Linux and Mac OS/X, also you can download the LPCXpresso IDE from www.lpcware.com with a free license after register. About LPCXpresso IDE, please refer to http://www.lpcware.com/lpcxpresso/download

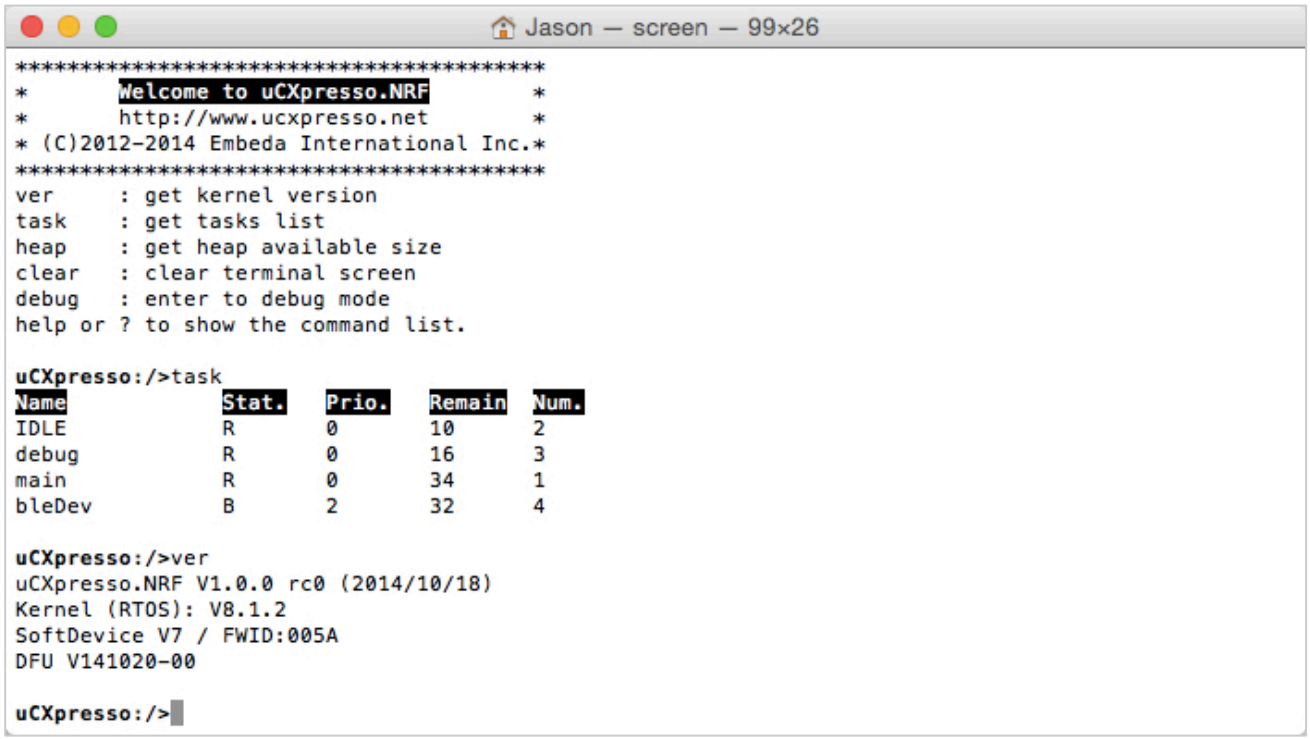
The uCXpresso.NRF is a RTOS C/C++ framework that port for the nano51822 SoC BLE module (Bluetooth Low Energy), and provides many features such as Peripherals (ADC, PWM, I2C, SPI, PIO, Interrupt, Timer….), BLE Services (Serial, Proximity, Heart Rate Meter…) and Multi-Thread (RTOS Thread, Semaphore, Mutex, MailBox…).

Multi-Thread

The uCXpresso.NRF framework is based on an advanced Real-Time Operating System (RTOS), By using the RTOS and the multi-tasking feature, you will simplify your concept, coding, and maintenance.

How uCXpresso.NRF RTOS works

When you type the 'task' on the debug console of serial-terminal, you can see the all tasks, which runs on the nano51822.



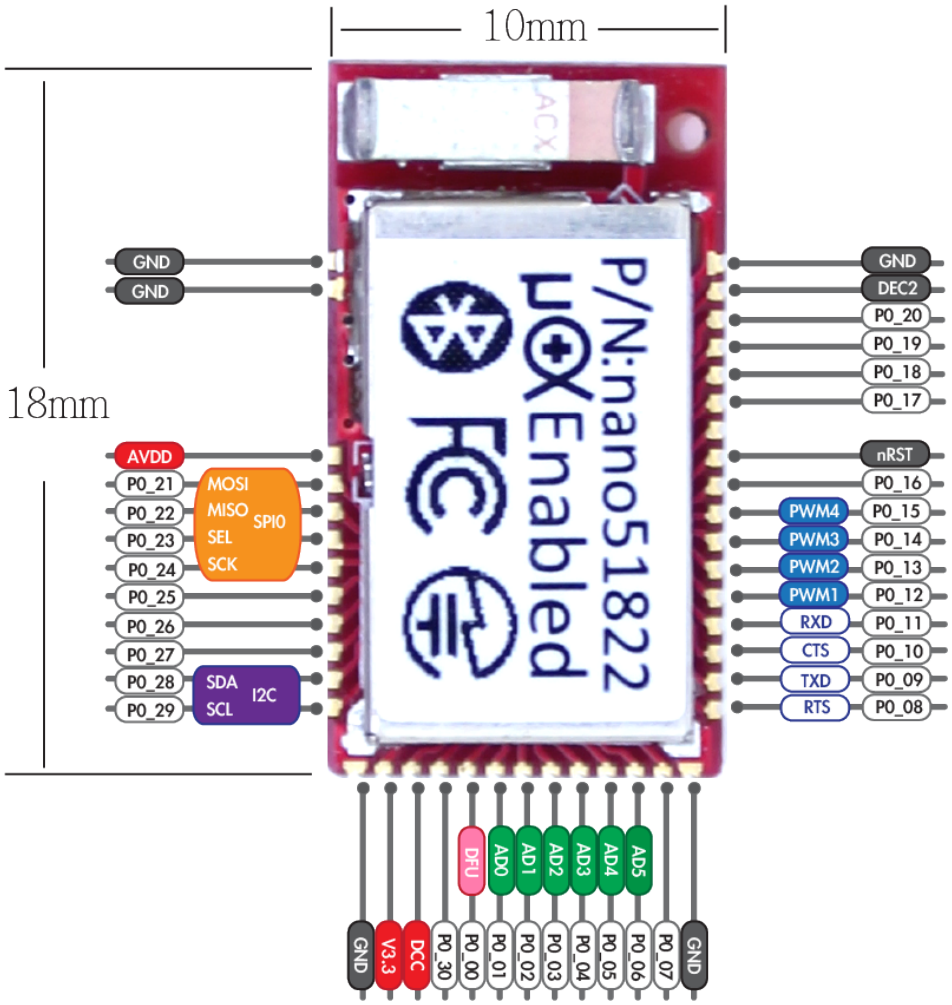
- The "bleDev" task is the "SoftDevice" driver and run in the background to receive and dispatch the ble events.
- The "main" task is the "main-routine" for application entry and main loop.
- The "debug" task is handler the DBG and debug console task.

Easy to use with C++ framework:

Ex. A LED on the pin 19 of nano11U37

```
CPin led(18);        // connect led object to P0.18 pin
led.output();        // set led as an output pin
led = HIGH;          // set led to HIGH (turn on LED)
```

Default Pin Definition:



DFU pin on P0.0 and active low (internal pull-up):

The nano51822 will be entered into the DFU mode when DFU pin is low.

If your application is crash, you also can force to enter into the DFU mode, and just put the DFU pin to low before power-on (or reset) the nano51822.

Ex. Set P0.20 as an input pin

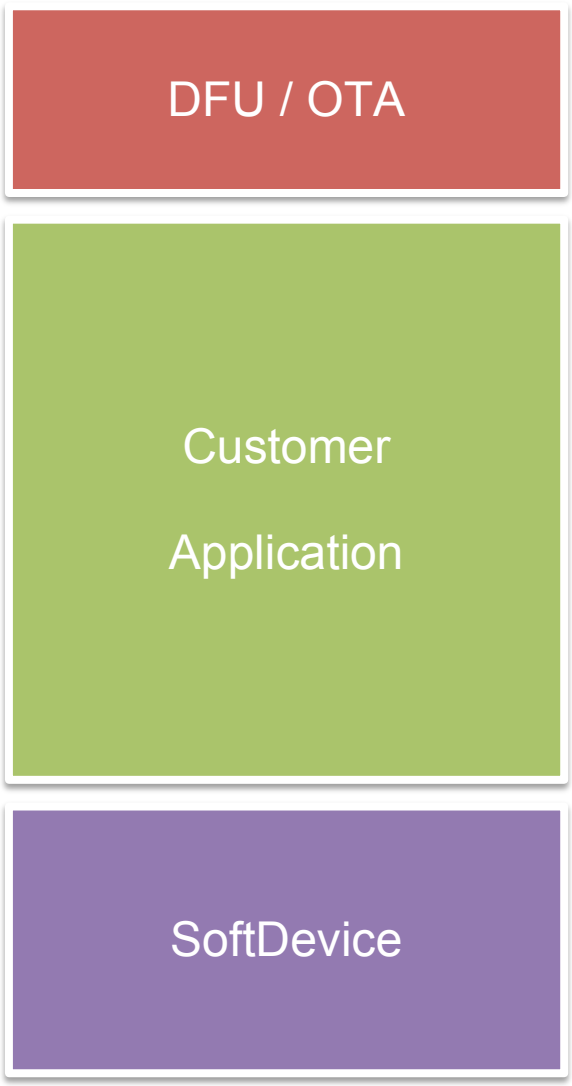
```
CPin key(20);          // connect key object to P0.20
key.input();           // set key as an input pin with internal Pull-Up.
if ( key==LOW ) {      // check key PIN LEVEL
     ...
}
```

Ex. Set P0.20 as a PWM output

```
swPwm servo;
servo.period(0.02);       // set globel PWM period time to 20ms
```

```
servo.add(20);              // add P0.20 to servo (pwm) module
servo.enable();             // enable the servo PWM to output
servo.dutyCycle(0, 0.8); // set servo duty-cycle to 80%
```

The blocks of nano51822



Final,

Let uCXpresso RTOS C/C++ framework decreases your development time and increases your joy and creativity!

1. Upgrade Your LPCXpresso IDE

1.1. Increase VM memory for Windows System

a. Close LPCXpesso if executing.

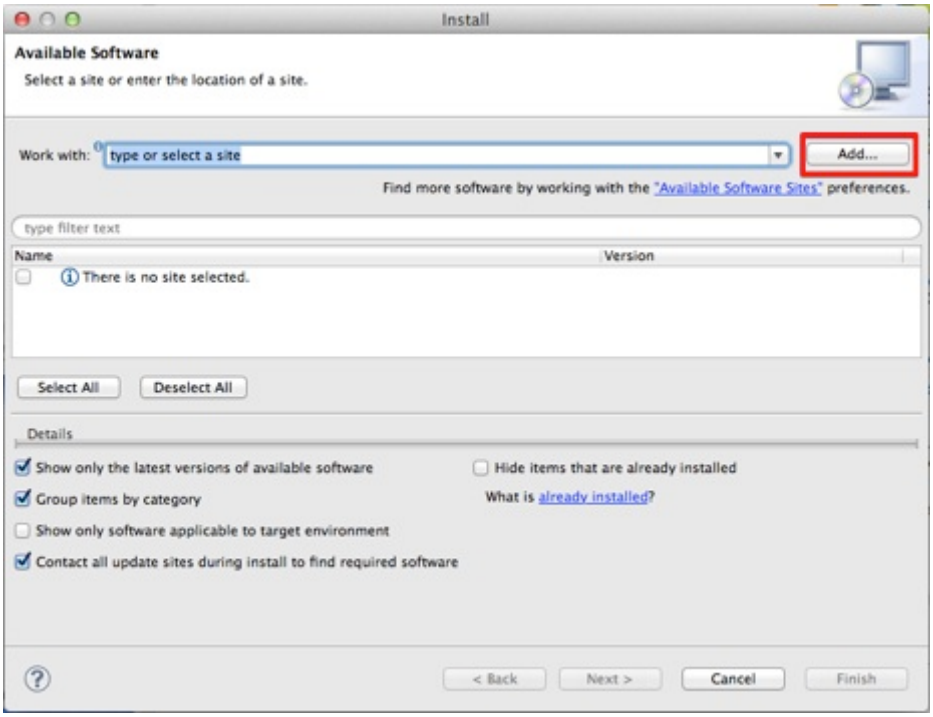
b. Open c:/nxp/LPCXpresso_x.x/lpcxpresso/lpcxpresso.ini

c. Modify MaxPermSize to 512

1.2 Install new software (Eclipse Plug-in)

LPCXpresso Main Menu > Help > Install New Software…

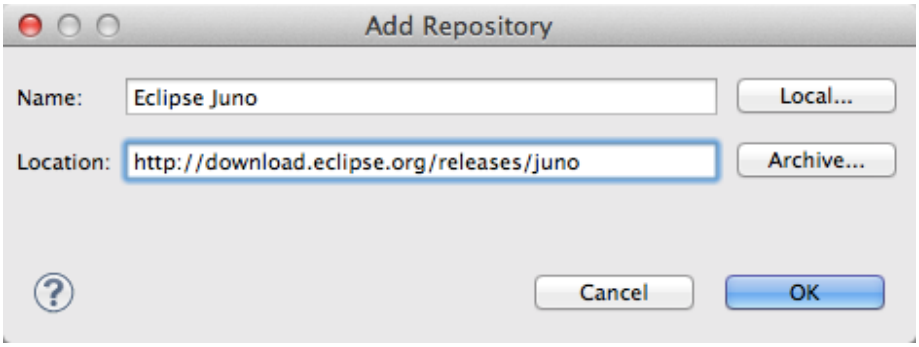
Click [Add]



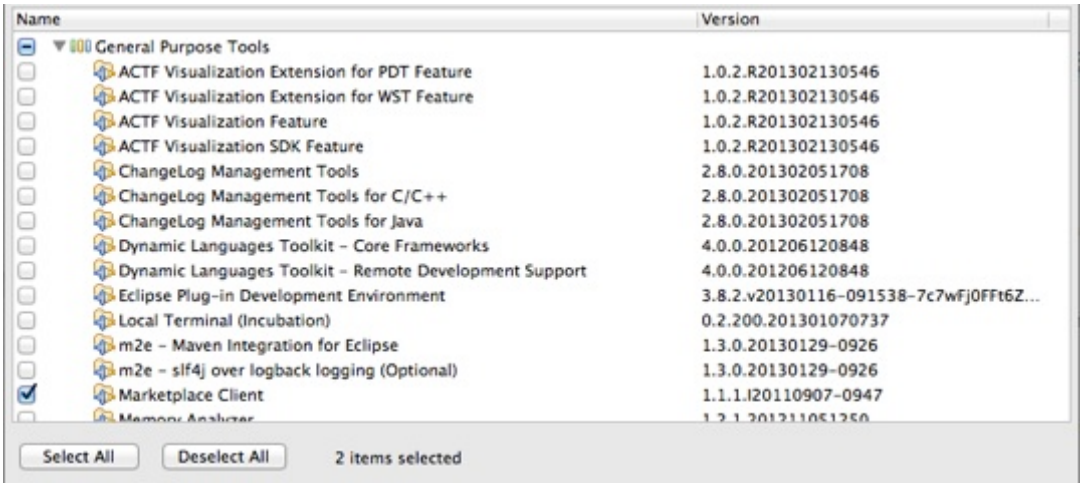
1.3 Install Eclipse Plug-In

Name: Eclipse Juno

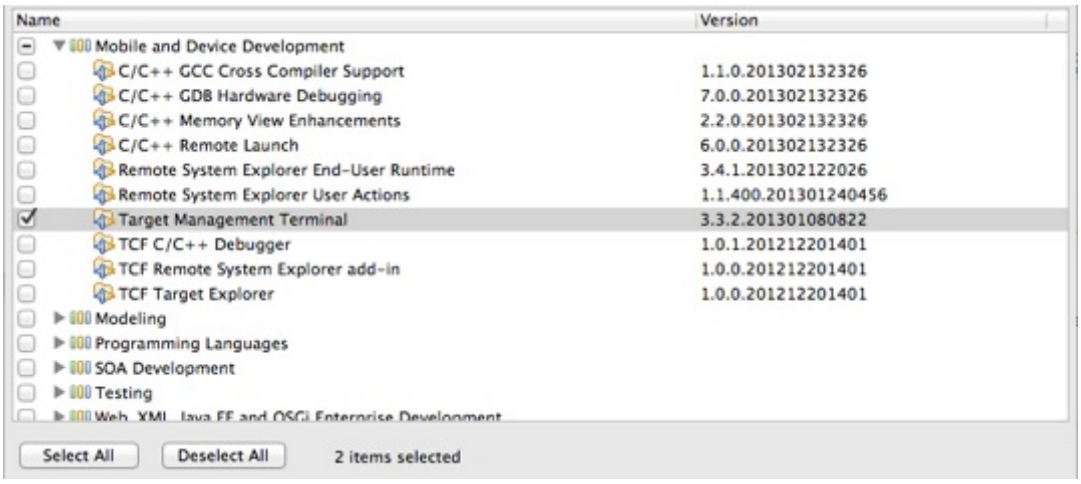
Location: http://download.eclipse.org/releases/juno



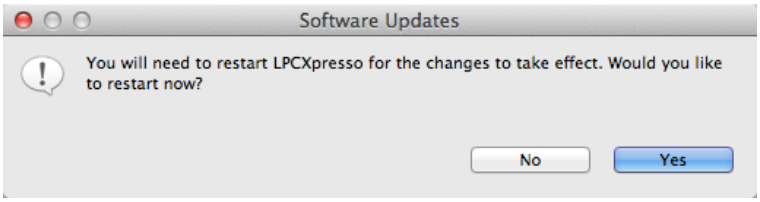
In General Purpose Tools section, select “Marketplace Client”



In Mobile and Device Development section, select “Target Management Terminal”



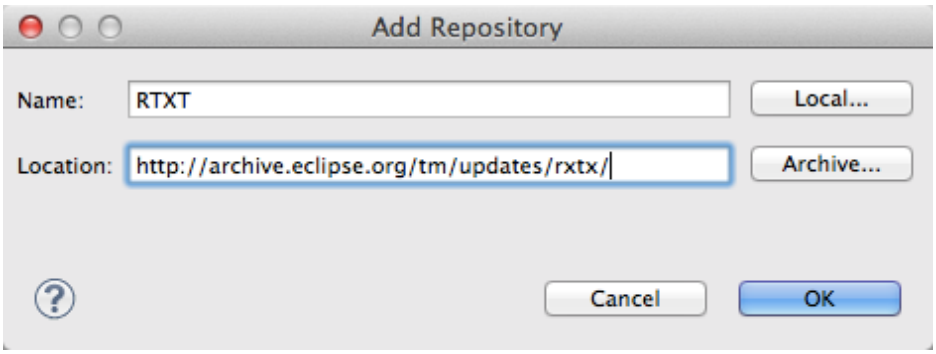
Click [Next] to start the installation. In progress, you need to [Confirm >] the Selected Features, and [accept] the licenses of software, and click [OK] to confirm any security-warning message, click [Yes] to restart and finish the installation if need.



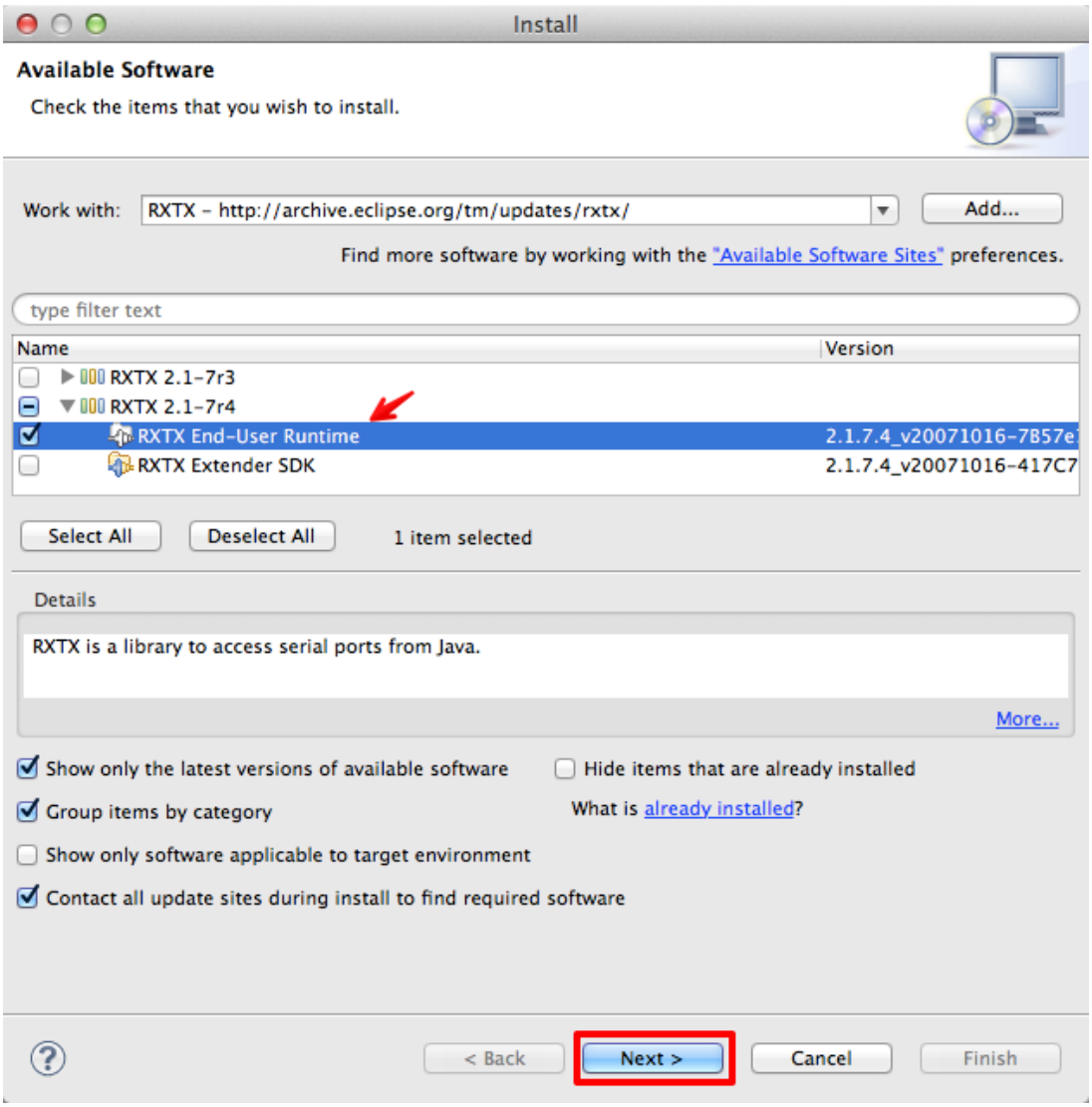
1.4. Install RXTX Plug-in

Name: RXTX

Location: http://archive.eclipse.org/tm/updates/rxtx/



Select latest version “RXTX End-User Runtime”



Click [Next] to start the installation.

Remark:

For Mac OS/X user, you have to follow below steps to enable the USB CDC virtual COM. Port:

1. Open Terminal App of OS/X
2. Type :

   sudo mkdir /var/lock

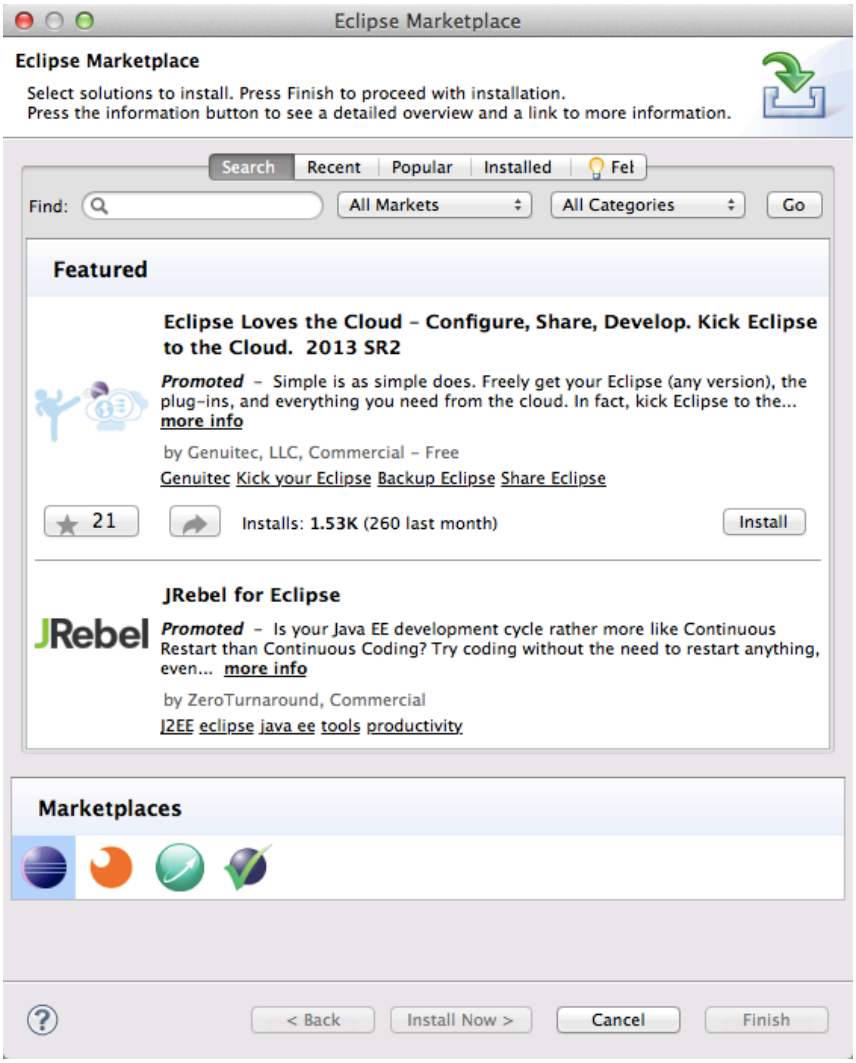
   sudo chmod a+rw /var/lock

In Windows System, The USB CDC driver can be download from below link:

for NANO11Uxx : http://www.embeda.com.tw/tw/wp-content/uploads/2014/01/nano11Uxx_usb_driver.zip (unzip and copy INF file to desktop, and USBCOM port driver direct to the INF file.)

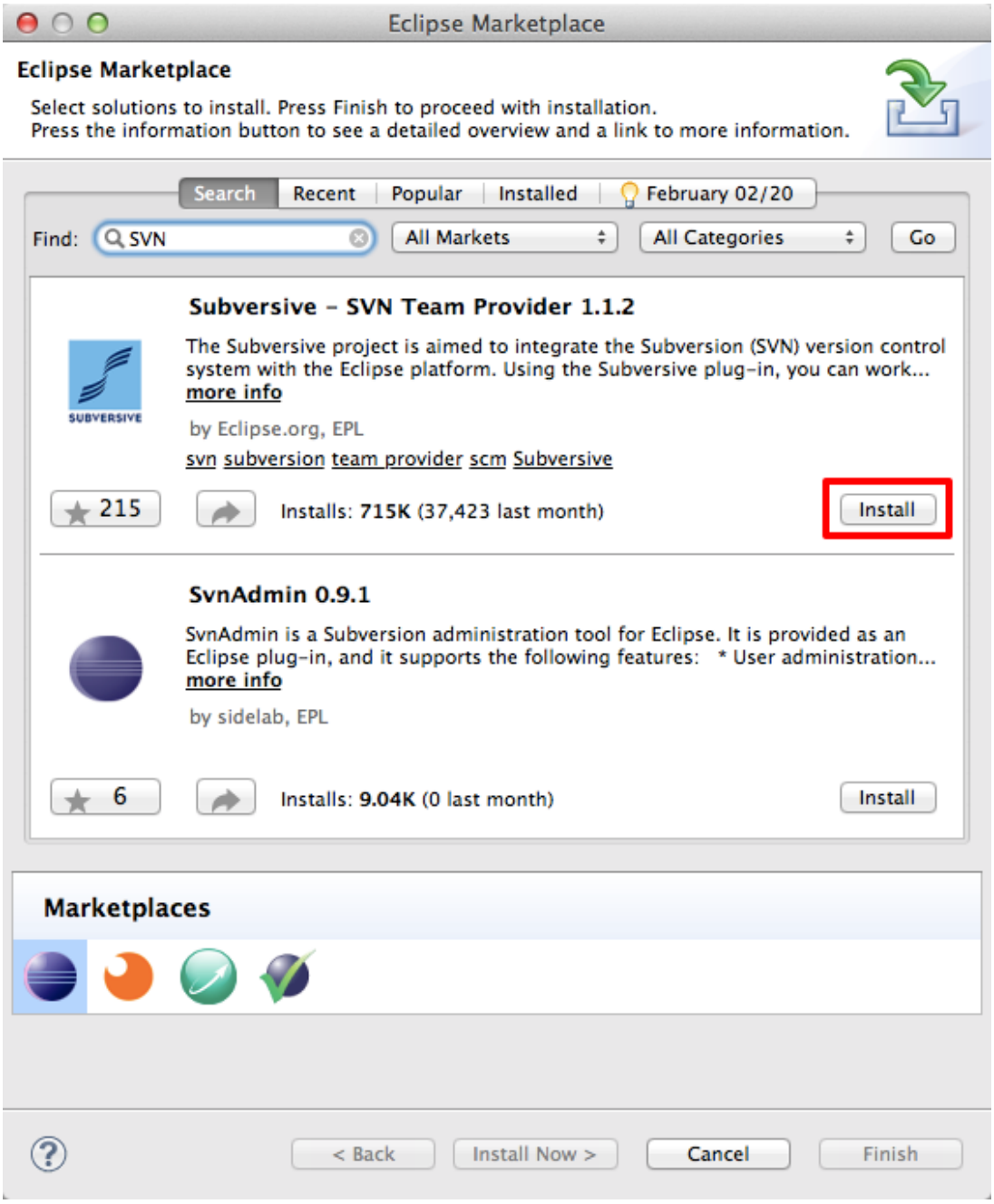
1.5. Install new software from Eclipse Marketplace

Click LPCXpresso Main Menu > Help > Marketplace…

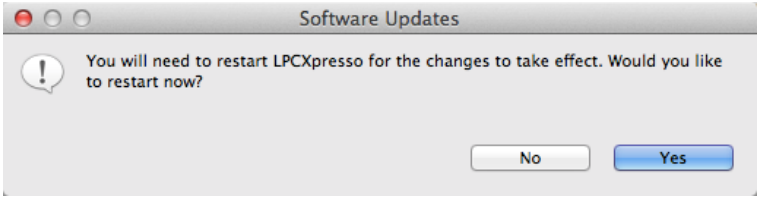


1.6 Install Subversion – SVN Team Provider (Plug-in)

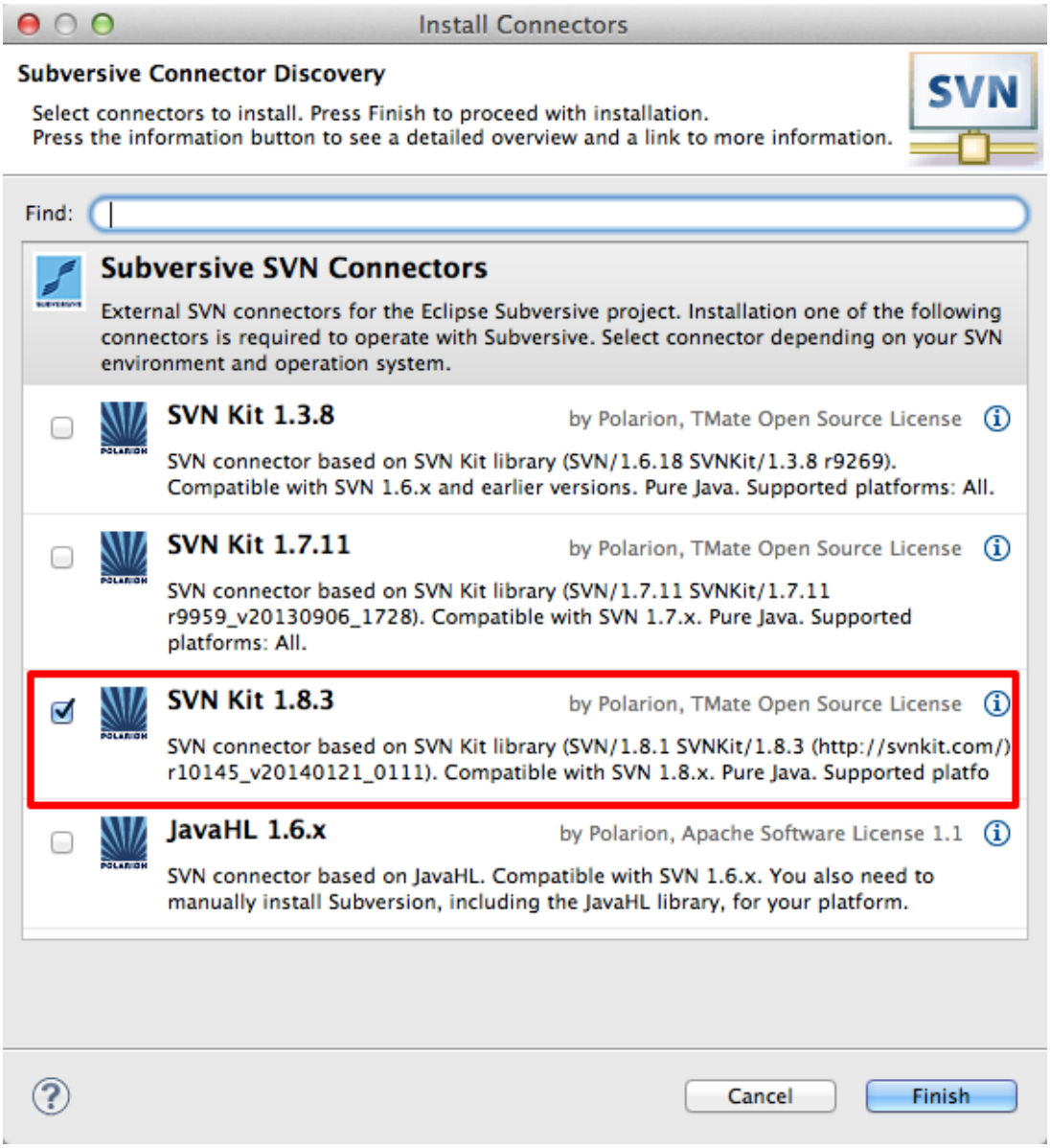
In Find, input “SVN” and click [Go]



Click [Install] to start the installation. In progress, you need to [Confirm >] the Selected Features, and [accept] the licenses of software, and click [OK] to confirm any security-warning message. Click [Yes] to restart and finish the installation if need.



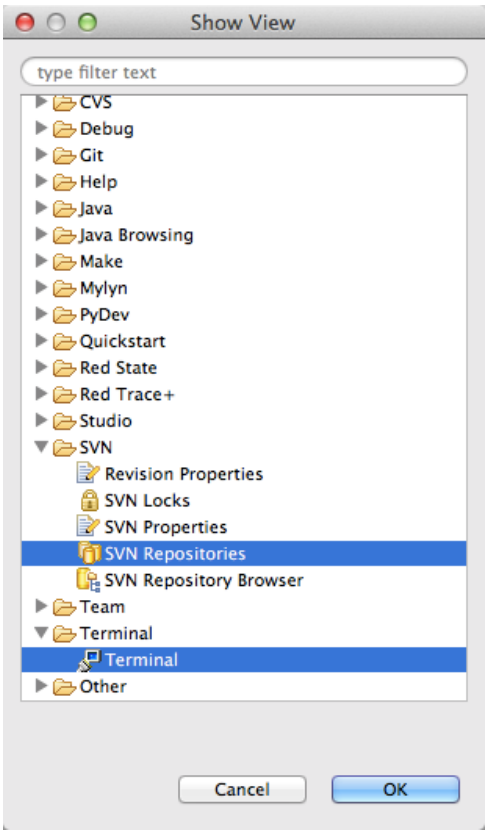
After restart, you may need to install the SVN connector kit:



Select the latest version “SVN Kit x.x.x”, and click [Finish] to start the installation. Confirm and click [Next], and [accept] the licenses of software, and click [OK] to confirm any security-warning message. Click [Yes] to restart and finish the installation if need.

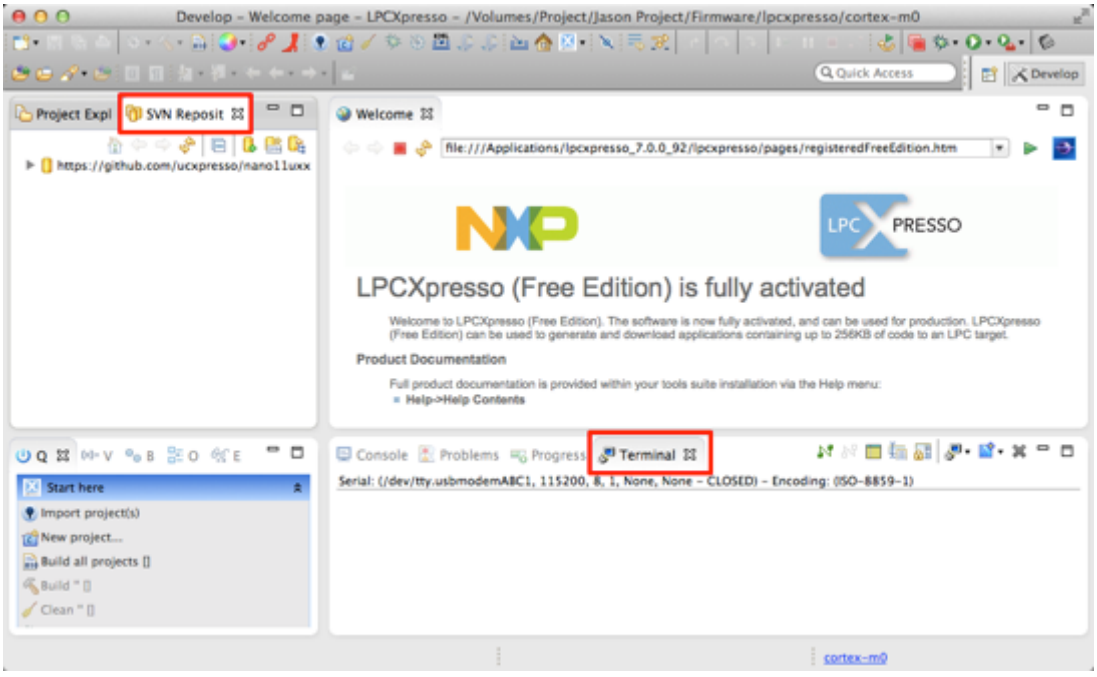
1-7. Show the SVN and Terminal Views

Click LPCXpresso Main Menu > Window > Show View > Other…



Select the “SVN Repositories” and “Terminal”, click [OK]

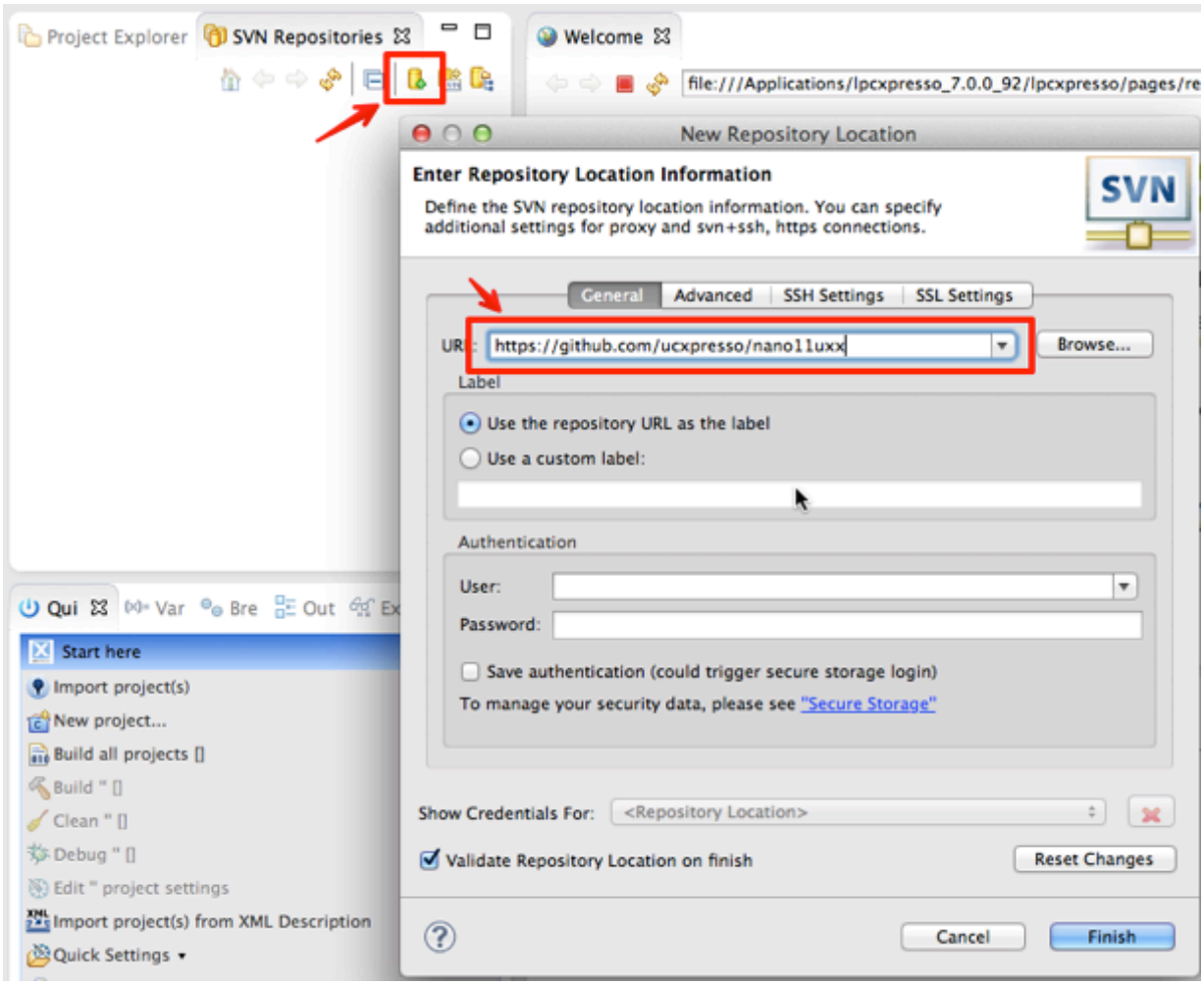
Move the “SVN Repositories” Tab to “Project Explorer” right side (optional)



2. Install uCXpresso.NRF Framework

2.1 Using GitHub Repositories

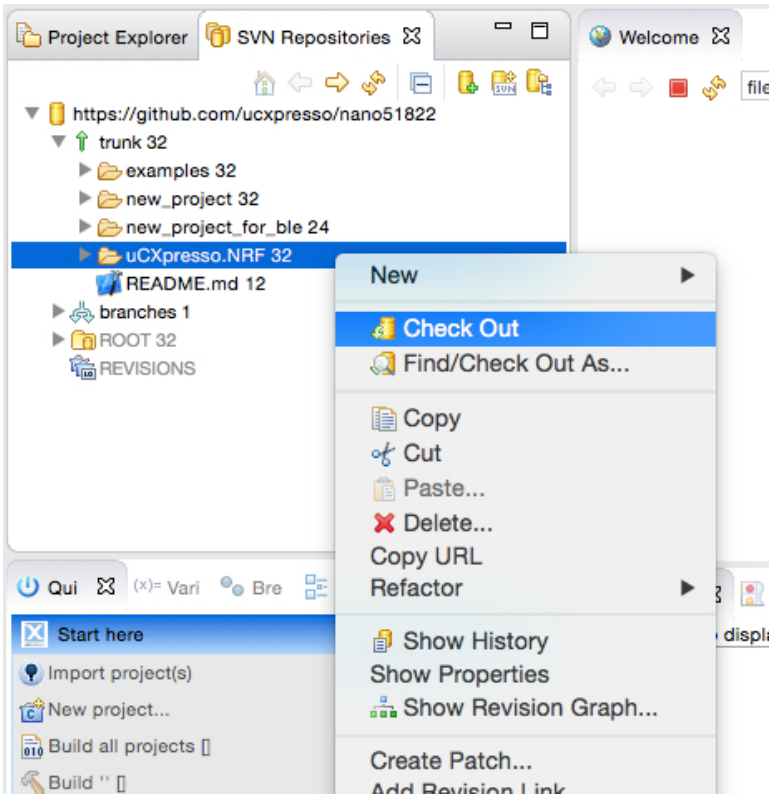
In “SVN Repositories” Tab, Click “New Repository Location”,



and URL Input: https://github.com/ucxpresso/nano51822

2.2 Check Out uCXpresso.NRF framework

Expand the uCXpresso.NRF SVN repository

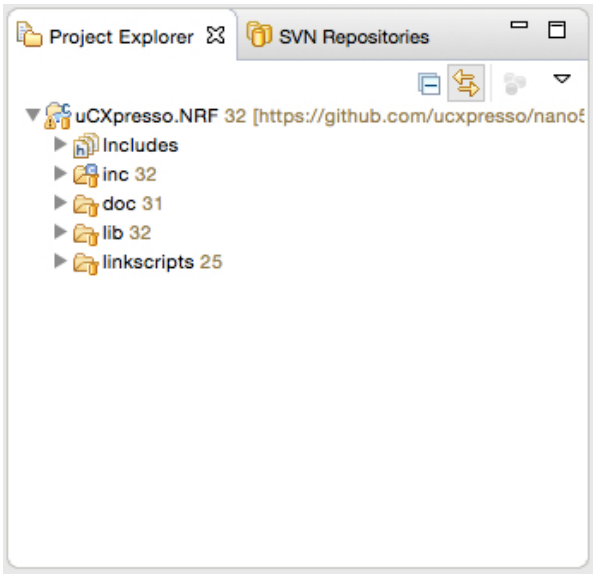


trunk: uCXpresso.NRF RC (Release Candidate) version

branches: All released version

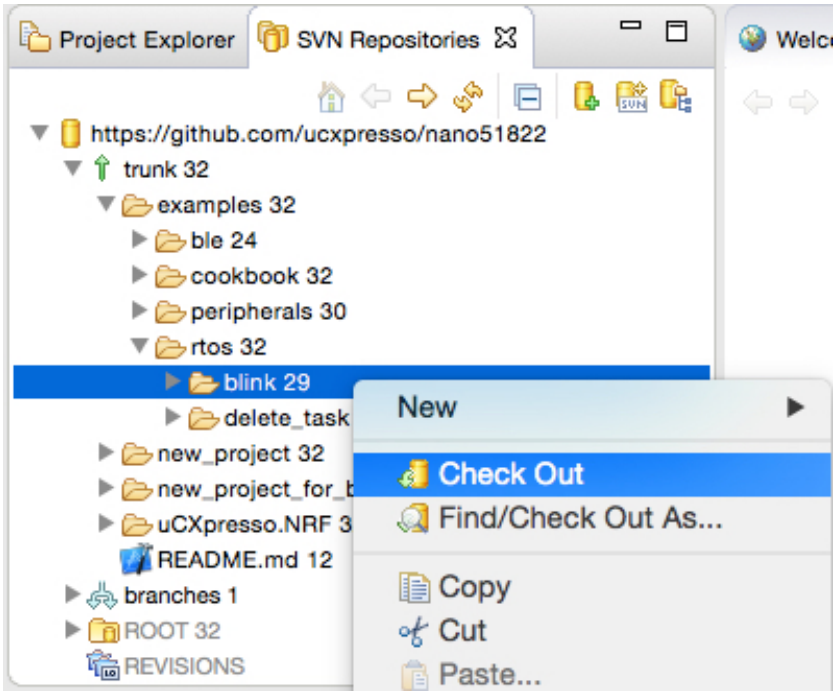
Select “uCXpresso.NRF” folder, and click right button of mouse, and click “Check Out” in drop down menu.

The uCXpresso.NRF framework show in your workspace after check out.



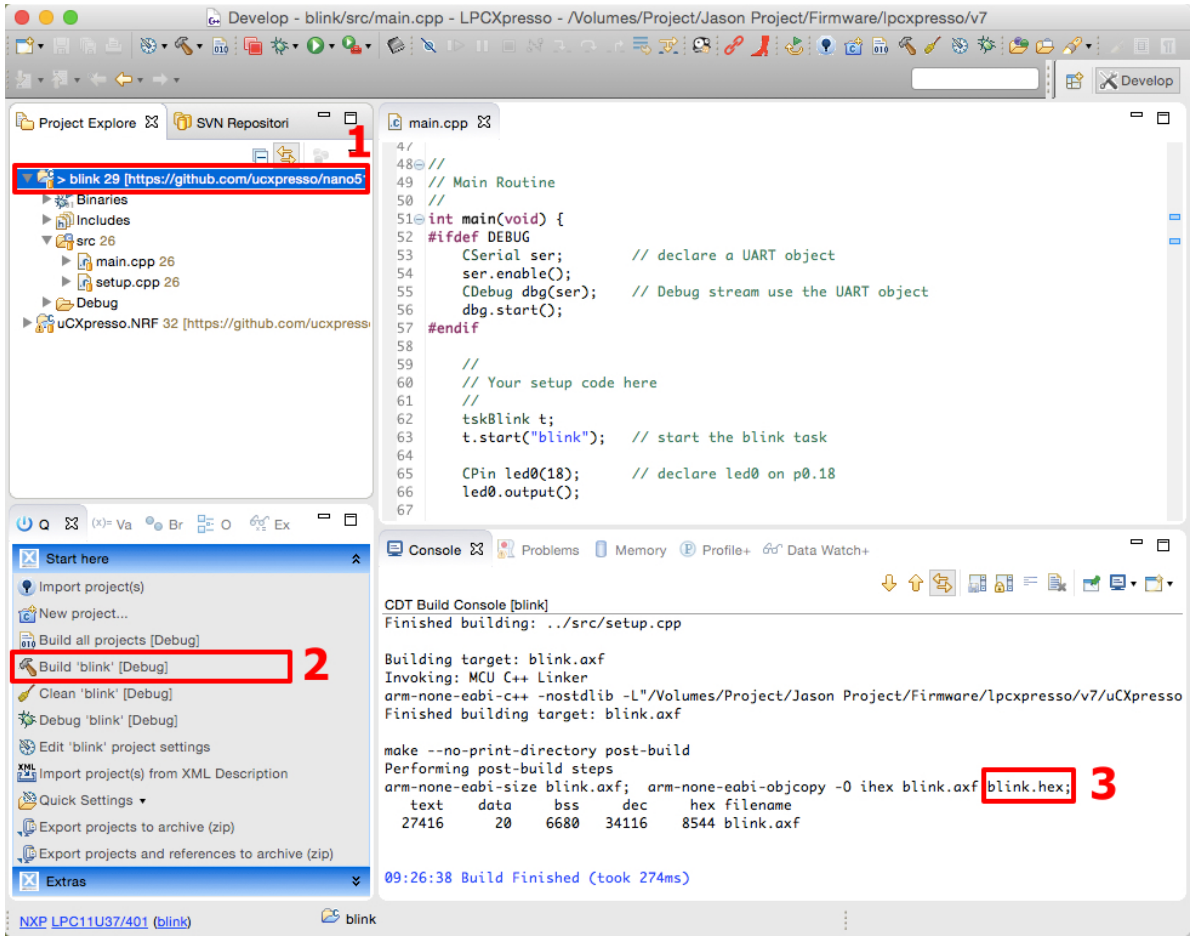
2.3 Check Out “blink” example

In SVN Repositories Tab, select the “blink” folder in “/examples/rtos”, then Check Out it.

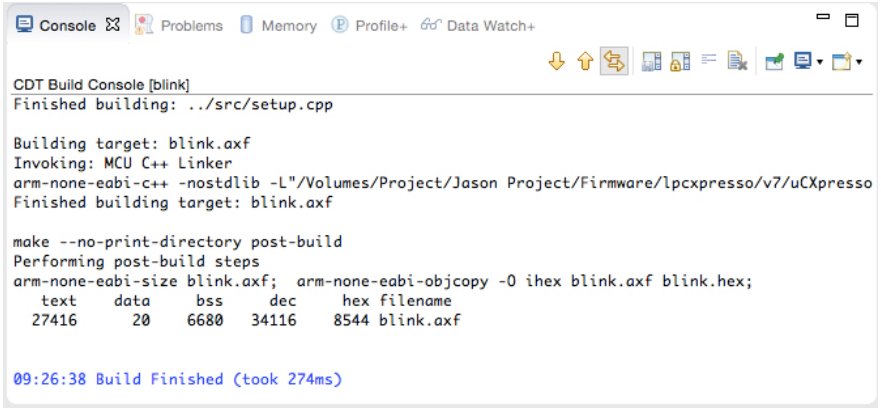


2.4 Build the “blink” example project

1. Click “blink” Project on Project Explorer View.
2. Click “Console” View
3. Click “Build ‘blink’ [Debug] on Quick Start View.



In Console View, the compiler will create a “blink.bin” image file.



Note:

code size = text + data = 27436 bytes

ram size = data + bss = 6700 bytes

3. Development Firmware Upgrade (DFU) and Over the Air (OTA)

Note:

The nano51822 provide the DFU code in module already.

3.1 Install the "Master Control Panel" (MCP) App for Android

Looking for "nRF Master Control Panel" on Google Play:

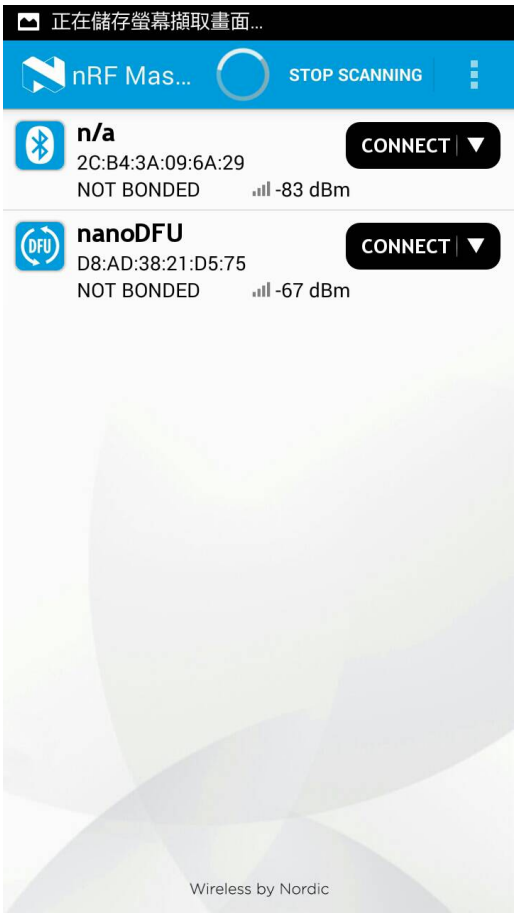
https://play.google.com/store/apps/details?id=no.nordicsemi.android.mcp

3.1 Prepare your Cloud Driver (Support Google Driver and Dropbox)

For example, copy the blink.hex to your dropbox that built at chapters 2.4.

3.2 DFU over the air

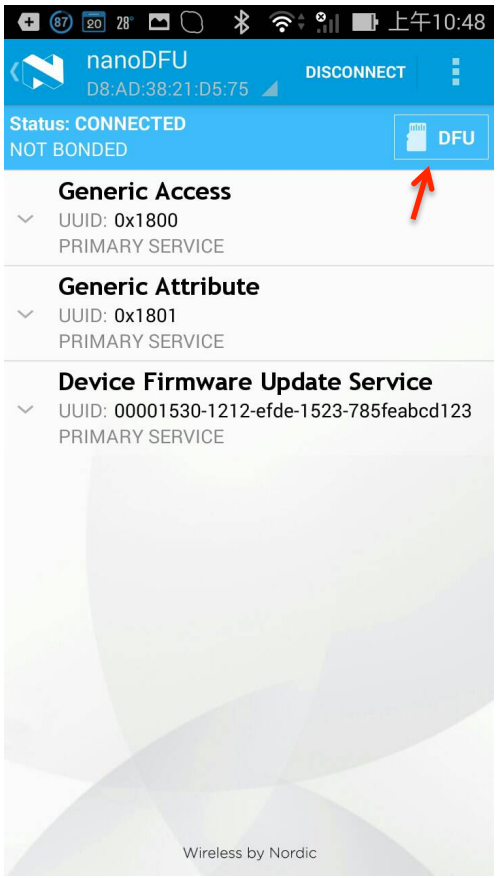
Scan the nano51822 DFU device (nanoDFU) in MCP.

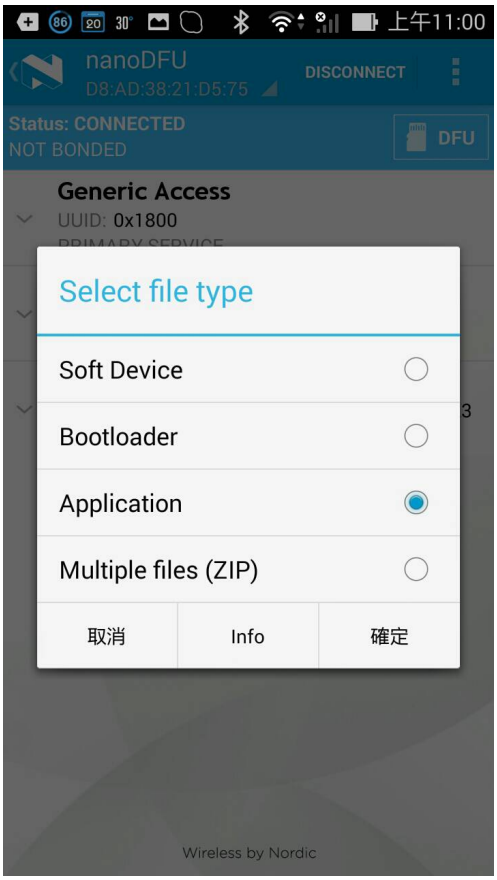


the nanoDFU will show in the scan list.

Click [CONNECT] on nanoDFU.

Connect to the nanoDFU device:



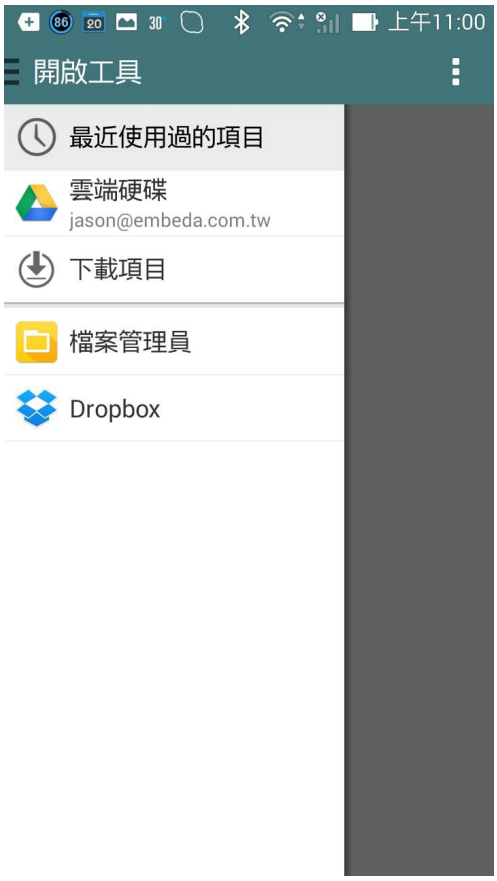


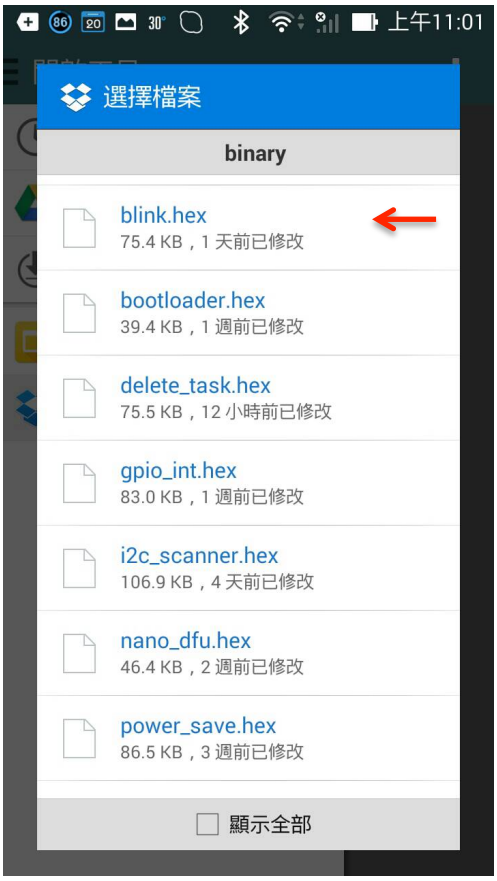
Click [DFU] and Select file type to “Application”

Warning:

You have to select the “Application” section for your application updated in nano51822.

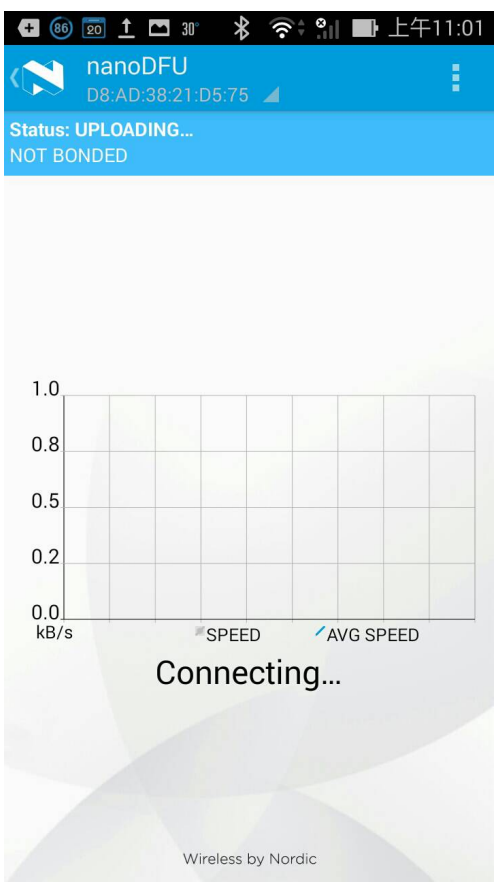
Select cloud driver (Google or Dropbox)

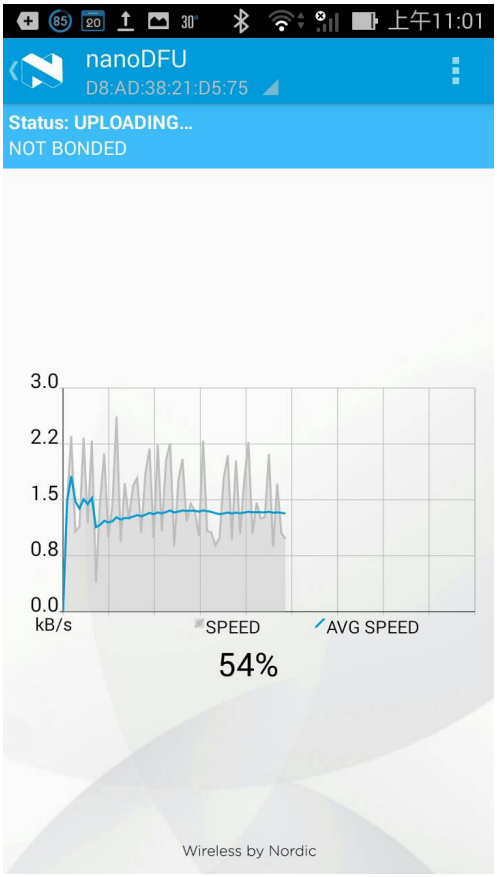




Select the “blink.hex” which you want to upload to DFU device.

Start to upload the “blink.hex” firmware to nanoDFU device.





After finished, the nanoDFU will run your application automatically.

4. Serial Terminal & Debug

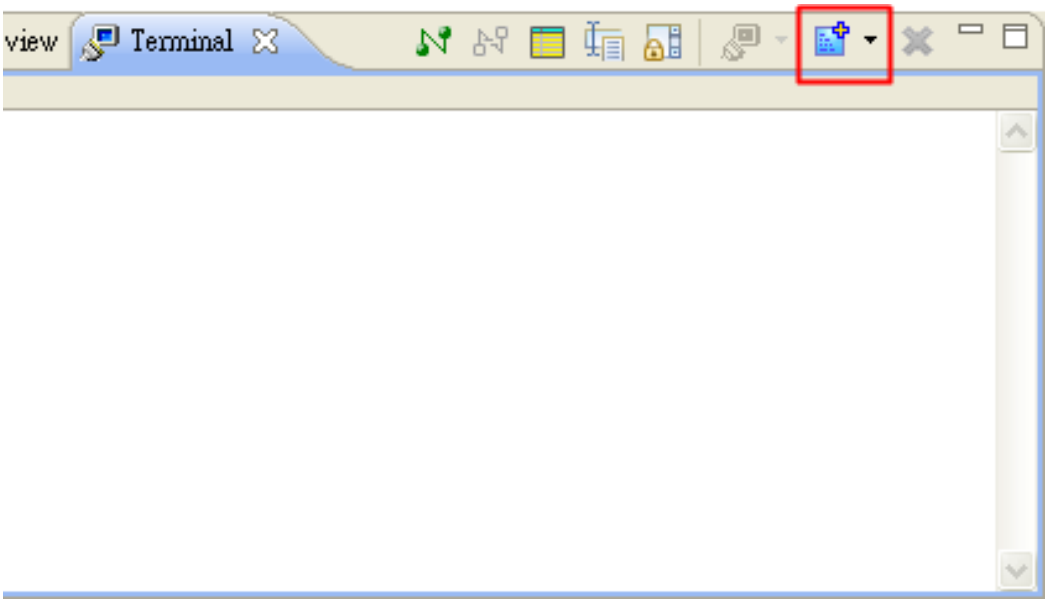
4.1 Serial Terminal (You need an USB to UART convertor)

In main.cpp of blink, the serial debug console will be enabled in DEBUG build.

main.cpp

```
47
48 //
49 // Main Routine
50 //
51 int main(void) {
52 #ifdef DEBUG
53     CSerial ser;          // declare a UART object
54     ser.enable();
55     CDebug dbg(ser);      // Debug stream use the UART object
56     dbg.start();
57 #endif
58
```

In Terminal View, click “New Terminal Connection” to add a new connection.

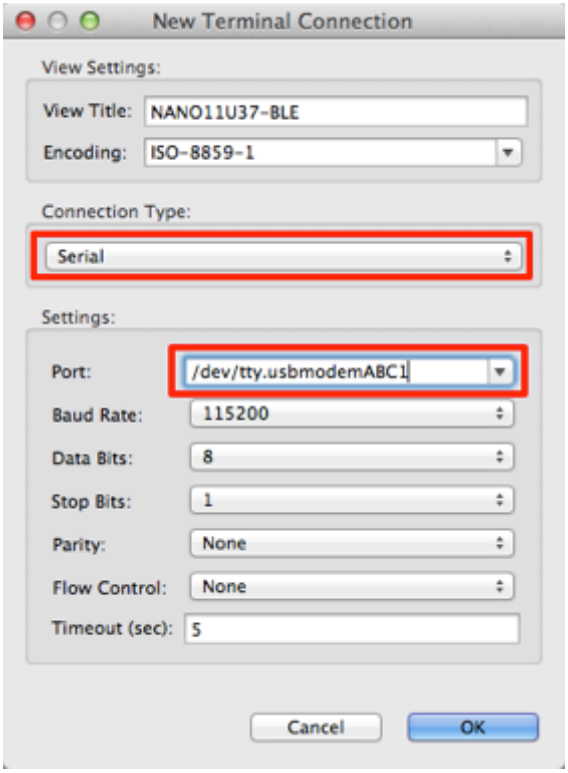


In Terminal Settings:

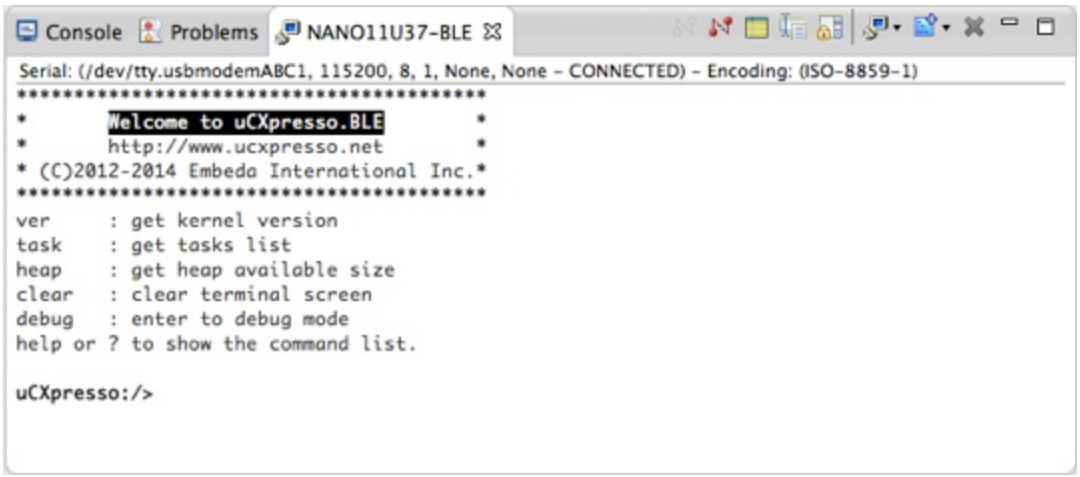
1. View Title: NANO11U37-BLE

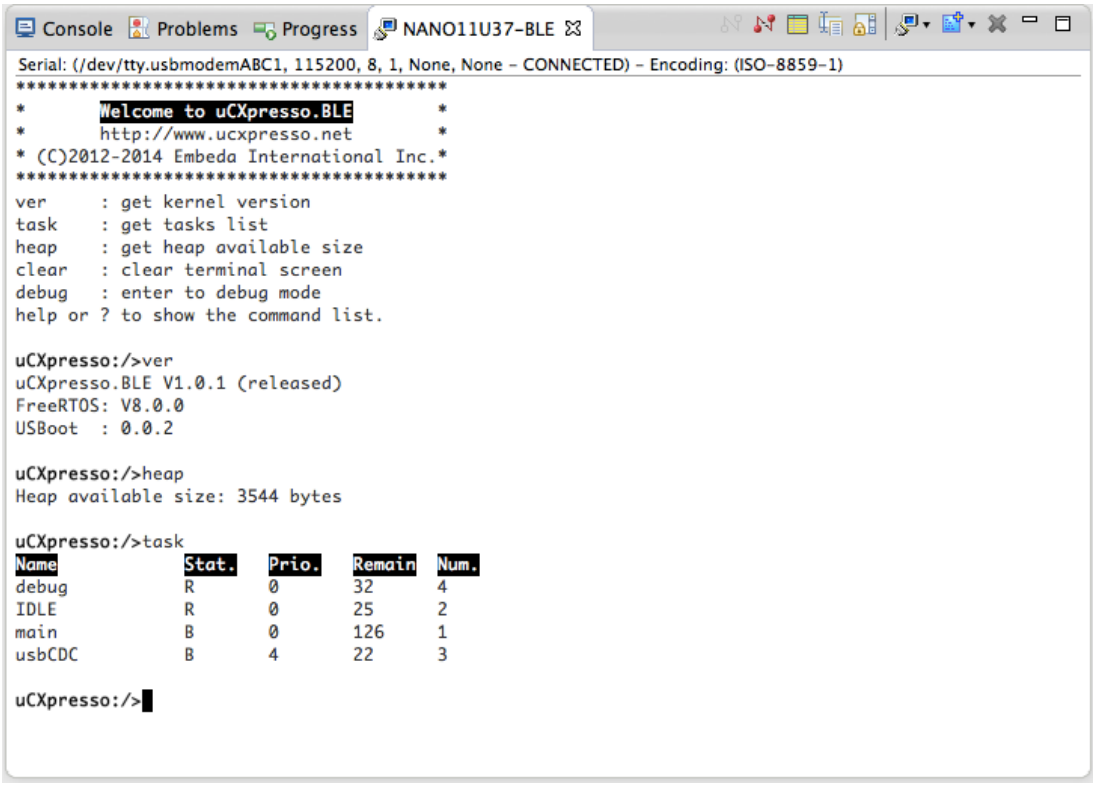
2. Connection Type: Serial

3. Port : /dev/tty.usbmodemABC1 (for Mac OS/X)



The blink serial shell will run in the Terminal View





Command:

ver, to show the kernel & module version.

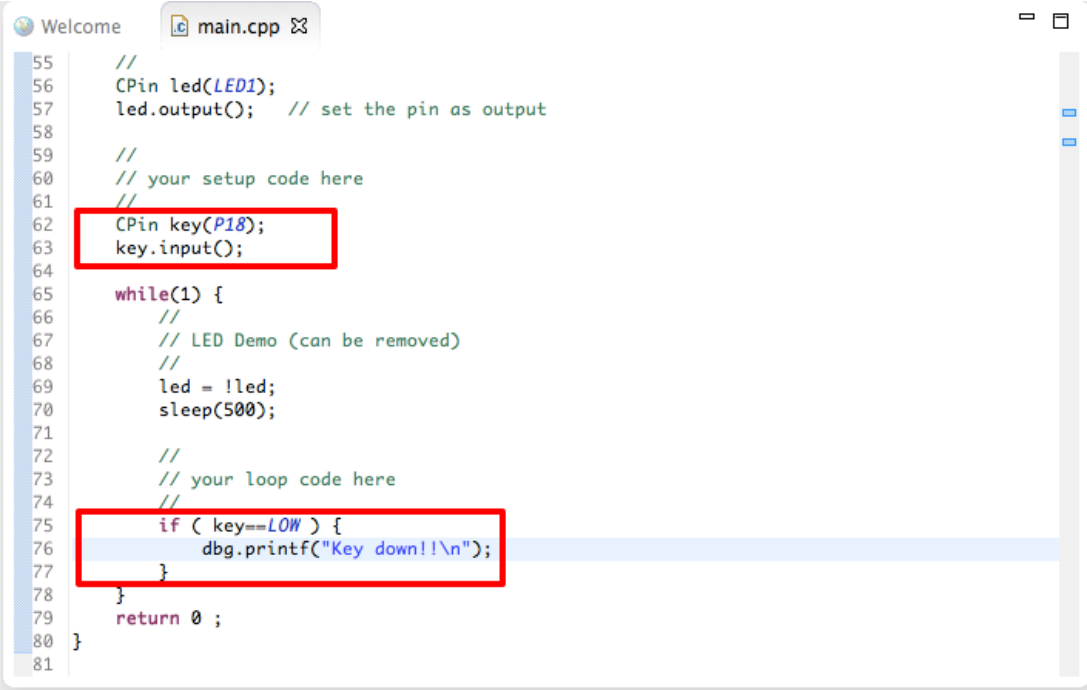
heap, to check the available size of heap memory.

task, to check all of tasks in the system.

debug, to enter to debug mode and view the DBG(…) message from user's program.

4.2 Debug

Add 'key' object in main.cpp of blink project.

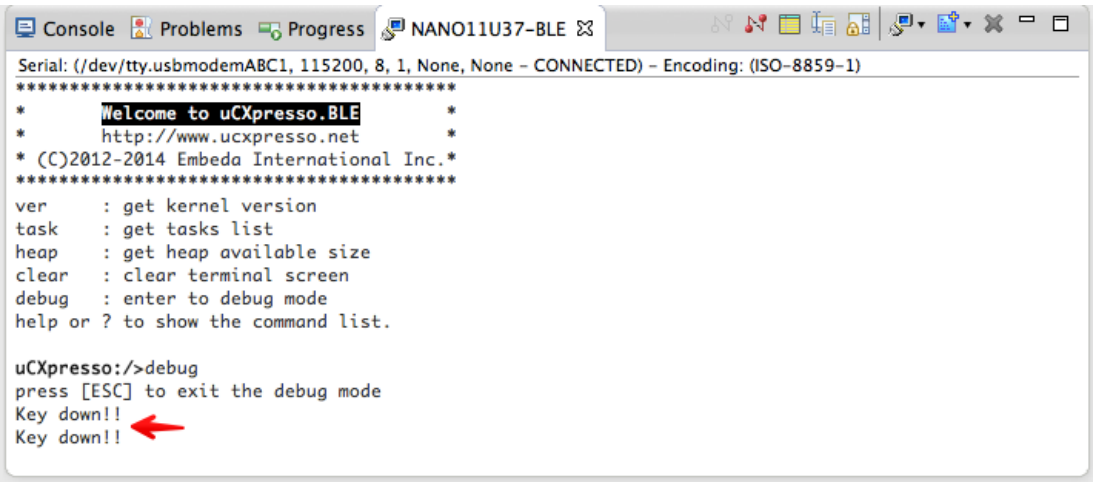


```
55      //
56      CPin led(LED1);
57      led.output();   // set the pin as output
58
59      //
60      // your setup code here
61      //
62      CPin key(P18);
63      key.input();
64
65      while(1) {
66          //
67          // LED Demo (can be removed)
68          //
69          led = !led;
70          sleep(500);
71
72          //
73          // your loop code here
74          //
75          if ( key==LOW ) {
76              dbg.printf("Key down!!\n");
77          }
78      }
79      return 0 ;
80  }
81
```

And add the key check in while loop.

Build Debug and download blink.bin to nanoFLASH, then execute it.

In Terminal View, input 'debug' command to enter the debug mode:



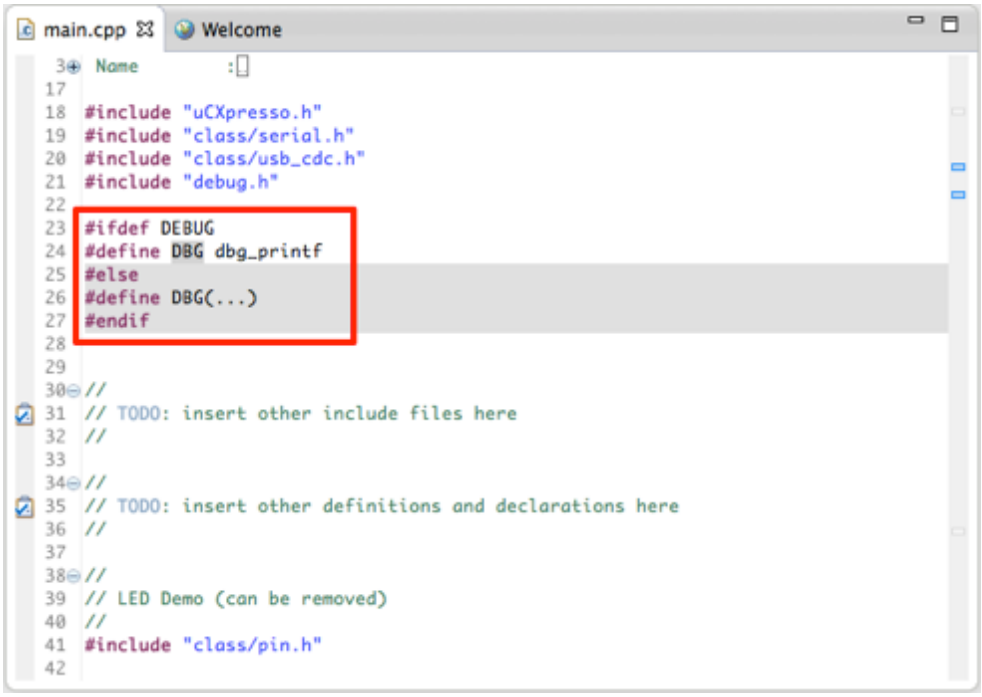
```
Serial: (/dev/tty.usbmodemABC1, 115200, 8, 1, None, None - CONNECTED) - Encoding: (ISO-8859-1)
*****************************************
*       Welcome to uCXpresso.BLE        *
*       http://www.ucxpresso.net        *
* (C)2012-2014 Embeda International Inc.*
*****************************************
ver     : get kernel version
task    : get tasks list
heap    : get heap available size
clear   : clear terminal screen
debug   : enter to debug mode
help or ? to show the command list.

uCXpresso:/>debug
press [ESC] to exit the debug mode
Key down!!
Key down!!
```

Try to push the key (pin 18) to LOW, and you can see the debug message to show in the terminal view.

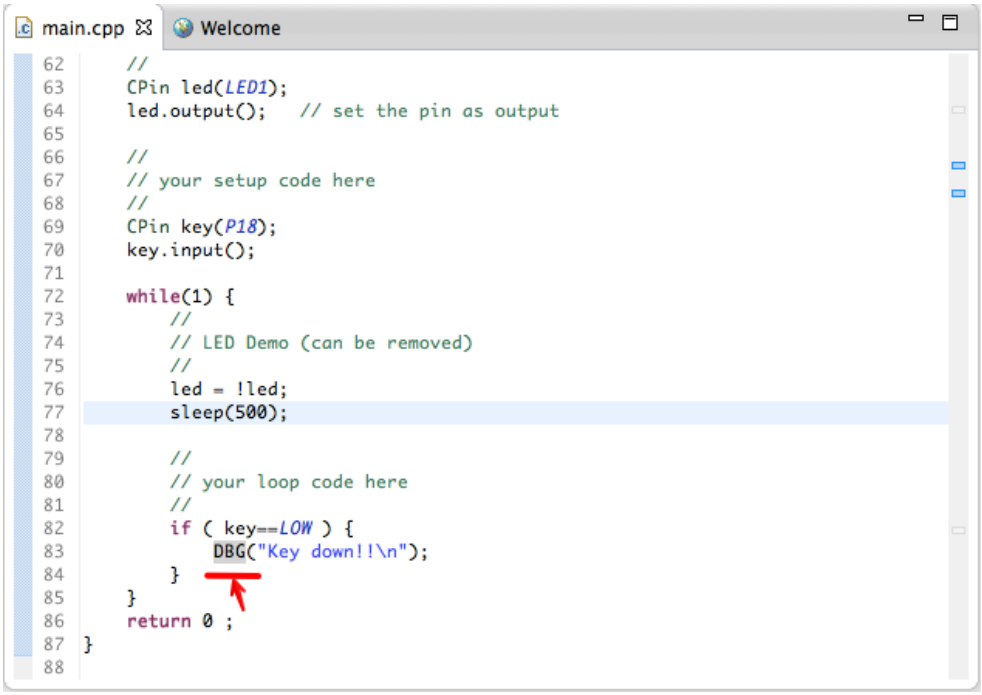
4.3 Made an invalidly debug message code.

Made an invalidly debug message code in [Release Build]



Using the #ifdef DEBUG to block and define the DBG to dbg_printf, and #else to block and define the DBG(...) to nothing.

Rename dbg.printf(...) to DBG(...) in your program

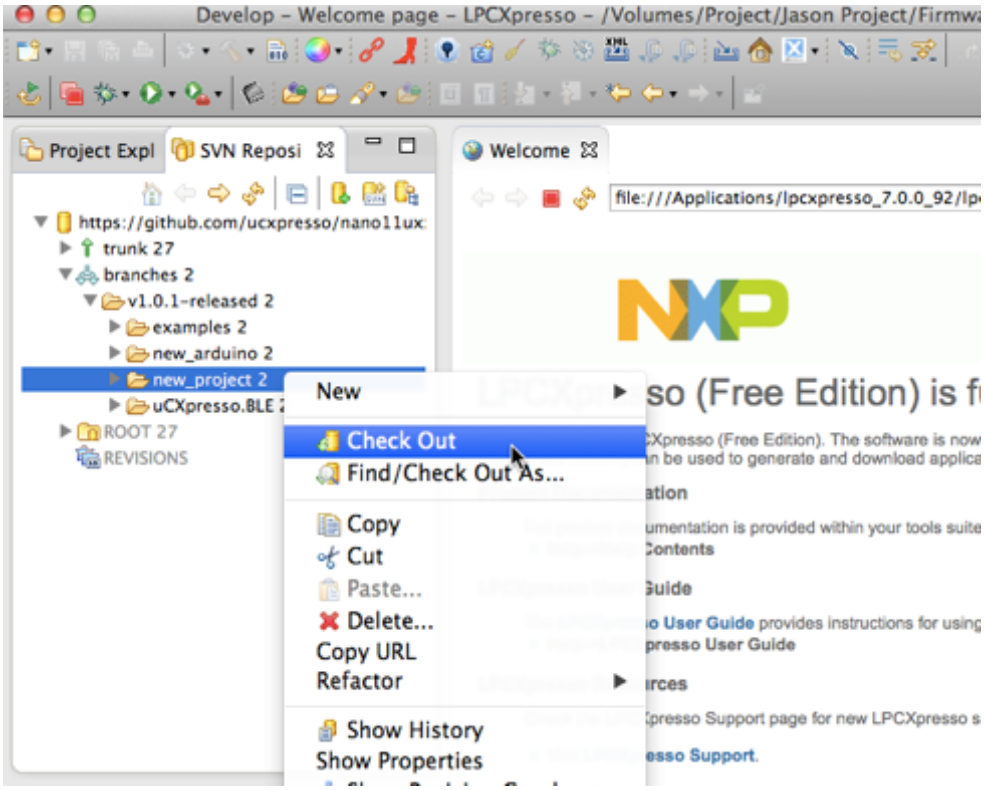


Now, The DBG(...) message code will be create in the [Debug Build] only, not in the [Release Build].

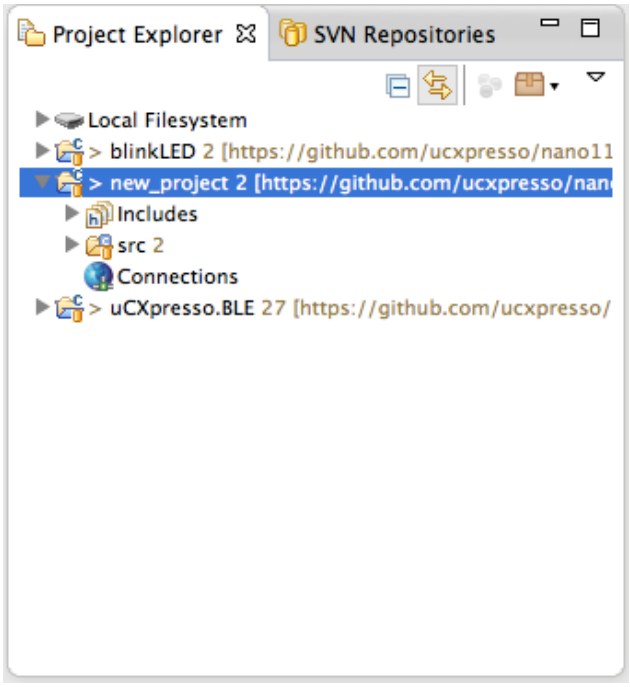
5. Create A New Project

5.1 Check Out the “new_project” template

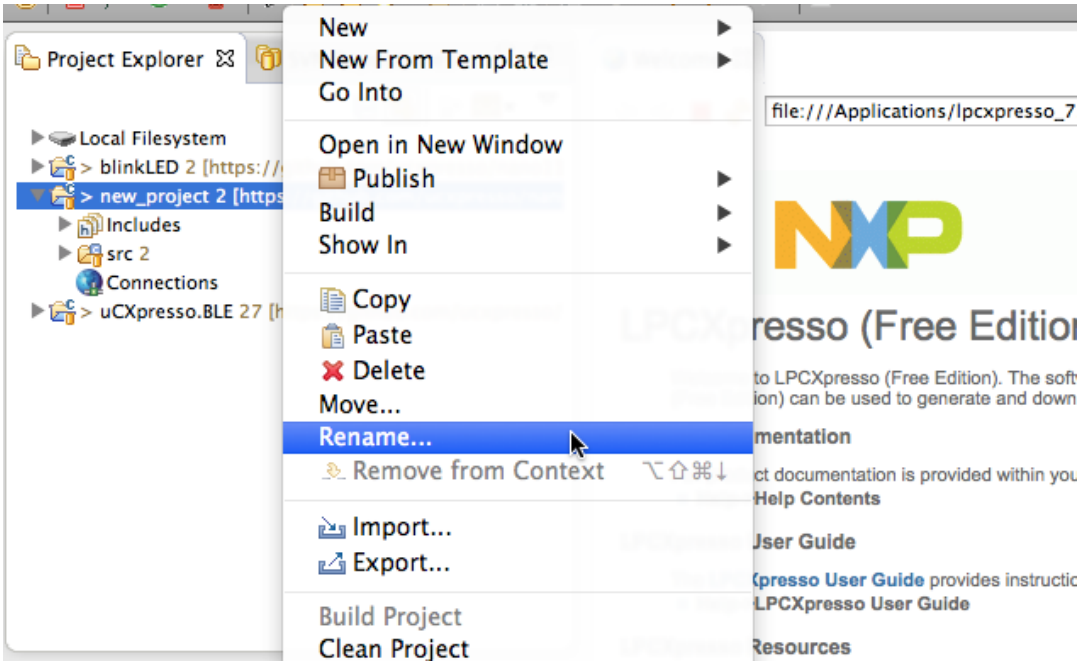
In SVN Repositories View, expand the three and “Check Out” the “new_project” to workspace.



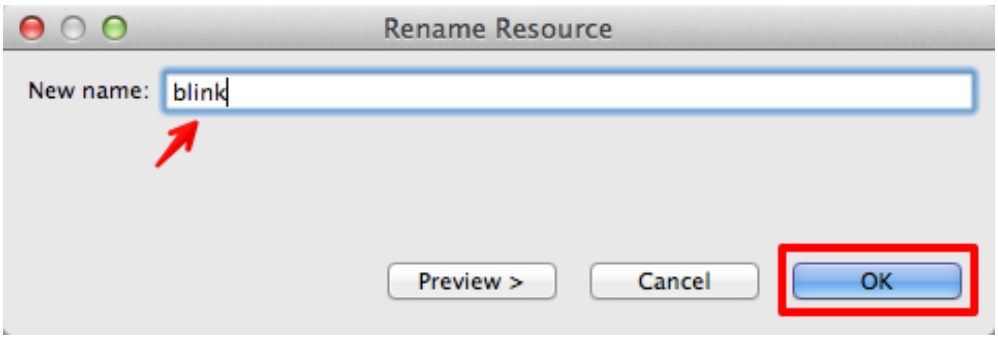
The “new_project” will show in workspace after Check Out.



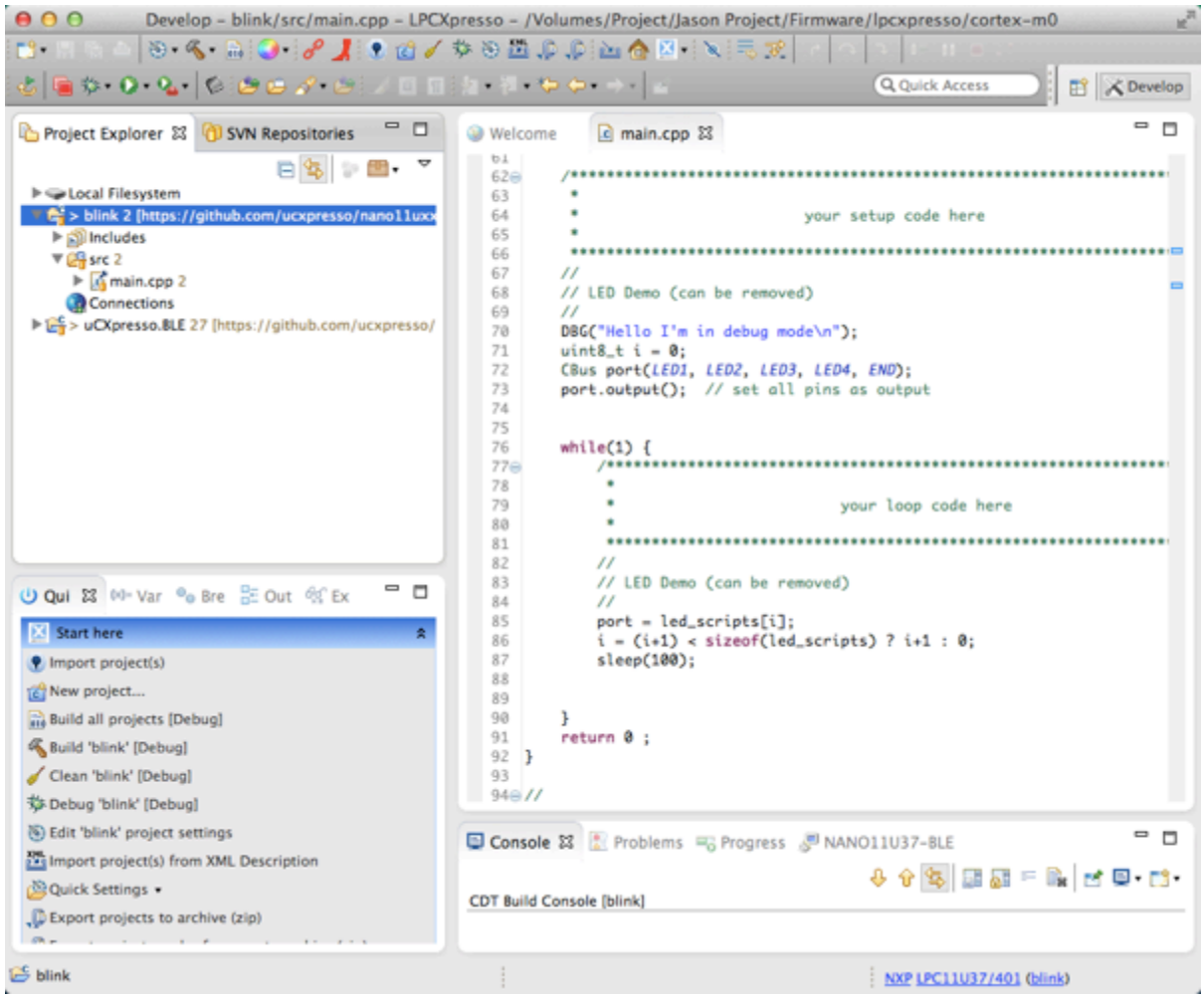
Rename the “new_project” to new project name what you want.



For example to rename to “blink” as below,



An empty project “blink” is in the workspace.



Expand the ‘src’ folder of new ‘blink’ project, and start to program your main.cpp.

6. RTOS Thread (need to modify for nano51822)

6.1 Overview

This is an example of how to make four threads to control the LED1, LED2, LED3 and LED4 respectively.

6.2 Make your thread class

Includes the CThread class to provide the thread features.

main.cpp

```
16  */
17
18  #include "uCXpresso.h"
19  #include "class/serial.h"
20  #include "class/usb_cdc.h"
21  #include "debug.h"
22
23  #ifdef DEBUG
24  #define DBG     dbg_printf
25  #else
26  #define DBG(...)
27  #endif
28
29  //
30  // TODO: insert other include files here
31  //
32  #include <class/thread.h>
33
34  //
35  // TODO: insert other definitions and declarations here
36  //
37
```

Creates a “tskLED1” thread and inherits from CThread class.

main.cpp

```
30  // TODO: insert other include files here
31  //
32  #include <class/thread.h>
33
34  //
35  // TODO: insert other definitions and declarations here
36  //
37
38  class tskLED1 : public CThread {
39  protected:
40
41      virtual void run() {
42          CPin led(LED1);     // use LED1
43          //
44          // thread run-loop
45          //
46          while(isAlive()) {
47              led = !led;     // blink led
48              sleep(100);
49          }
50      }
51  };
```

Implements the “virtual void run()” member function. and have to understand that “run()” member function is the life-cycle of thread. Put the isAlive() in the while-loop to check whether thread is actively or not.

In the same way, to make other tskLED2, tskLED3 and tskLED4 threads to control the different LEDs and different sleep-time.

main.cpp

```
class tskLED2 : public CThread {
protected:

    virtual void run() {
        CPin led(LED2);     // use LED2
        //
        // thread run-loop
        //
        while(isAlive()) {
            led = !led;     // blink led
            sleep(200);
        }
    }
};

class tskLED3 : public CThread {
protected:

    virtual void run() {
        CPin led(LED3);     // use LED3
        //
        // thread run-loop
        //
        while(isAlive()) {
            led = !led;     // blink led
            sleep(300);
        }
    }
};

class tskLED4 : public CThread {
protected:

    virtual void run() {
        CPin led(LED4);     // use LED4
        //
        // thread run-loop
        //
        while(isAlive()) {
            led = !led;     // blink led
            sleep(400);
        }
    }
};
```

Start the threads

main.cpp

```
    /*********************************************************************
     *
     *                        your setup code here
     *
     *********************************************************************/
    tskLED1 led1;
    led1.start("led1");     // task name = "led1"

    tskLED2 led2;
    led2.start("led2");     // task name = "led2"

    tskLED3 led3;
    led3.start("led3");     // task name = "led3"

    tskLED4 led4;
    led4.start("led4");     // task name = "led4"

    while(1) {
```

Remark:

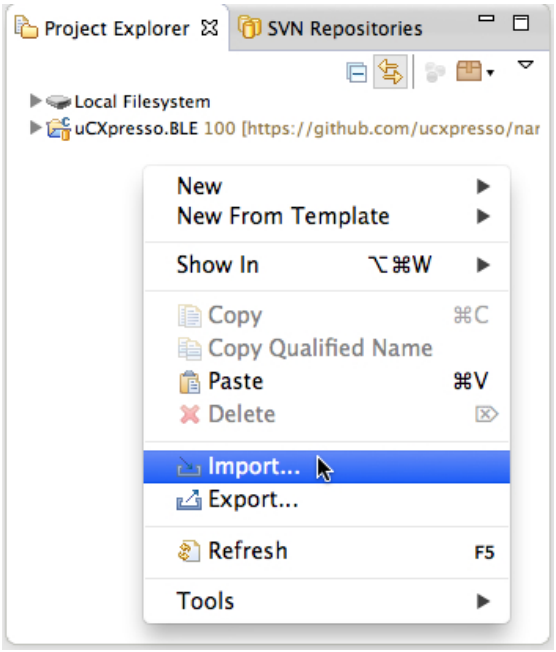
The example uses the LED1~LED4, so you need to remove the all of the original LED demo code from main.cpp as below:

To build the example and download the binary file to nano11U37. You will see that LED1, LED2, LED3 and LED4 are blinking standalone.

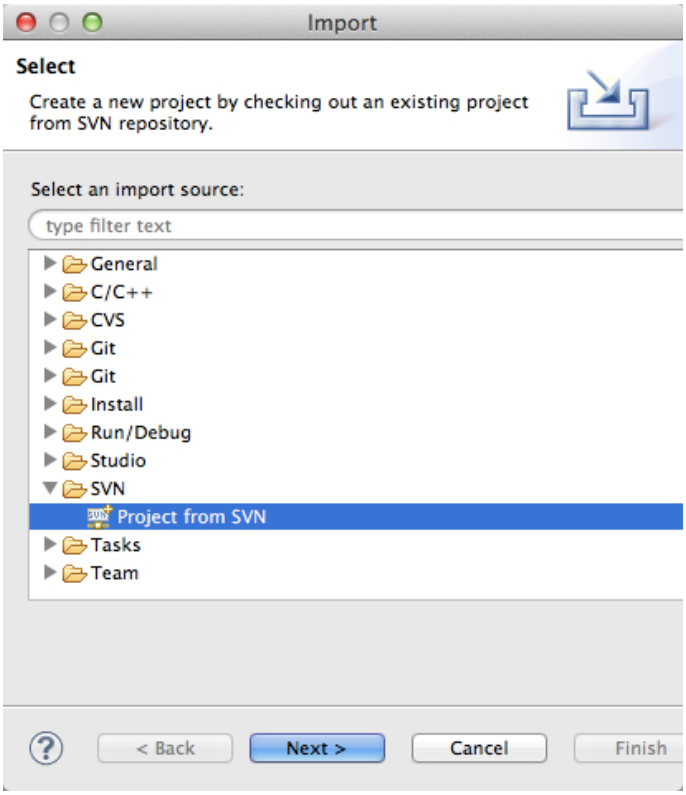
The final example “blink” project on the GitHub: /nano51822/examples/rtos :

Appendix A. Safe Way to Check Out the Example Projects

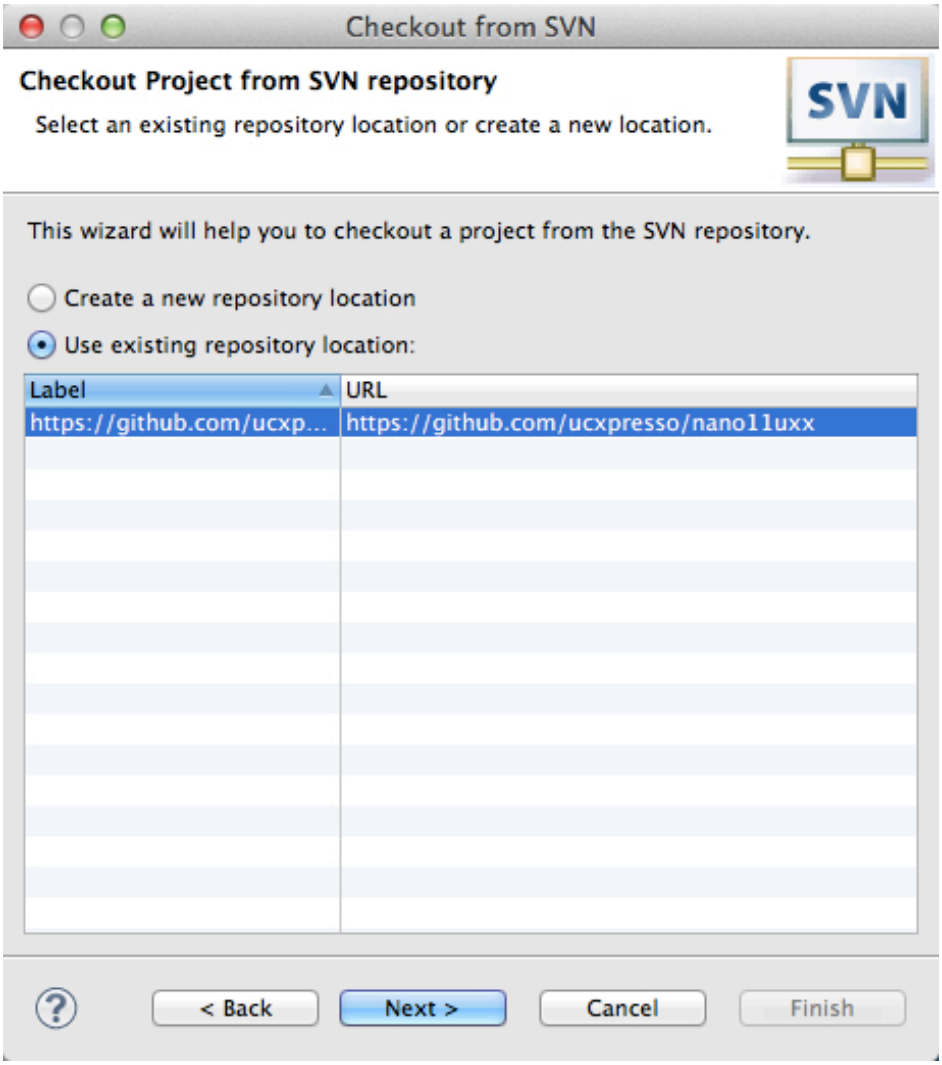
In “Project Explorer” view, click the mouse right button and show the drop-down menu, then select the “Import”.



Select “Project from SVN”.



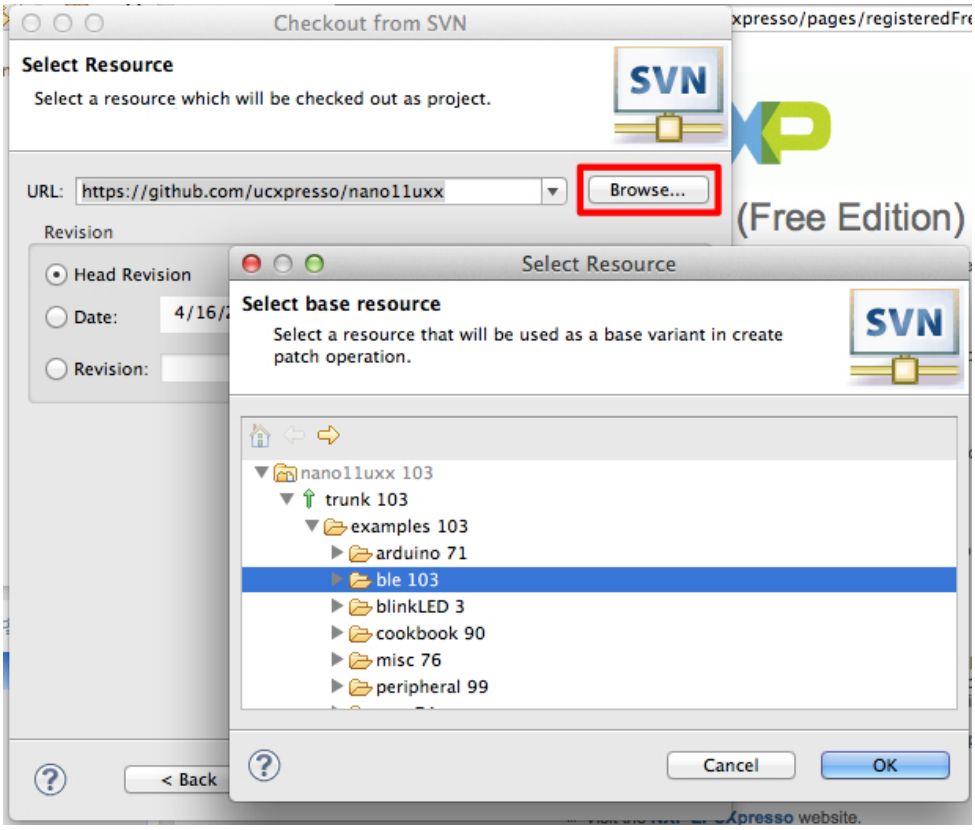
Use existing repository location:



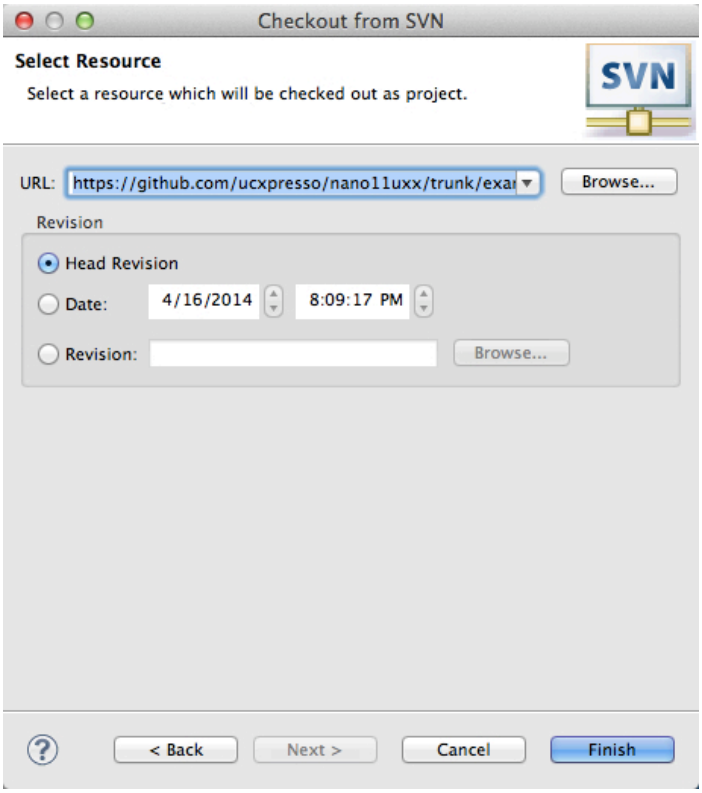
if you don't have our repository url, please select the "Create a new repository location", and input the https://github.com/ucxpresso/nano11uxx for new repository url.

Select SVN resource,

example, select the “ble” examples root path and click [OK].

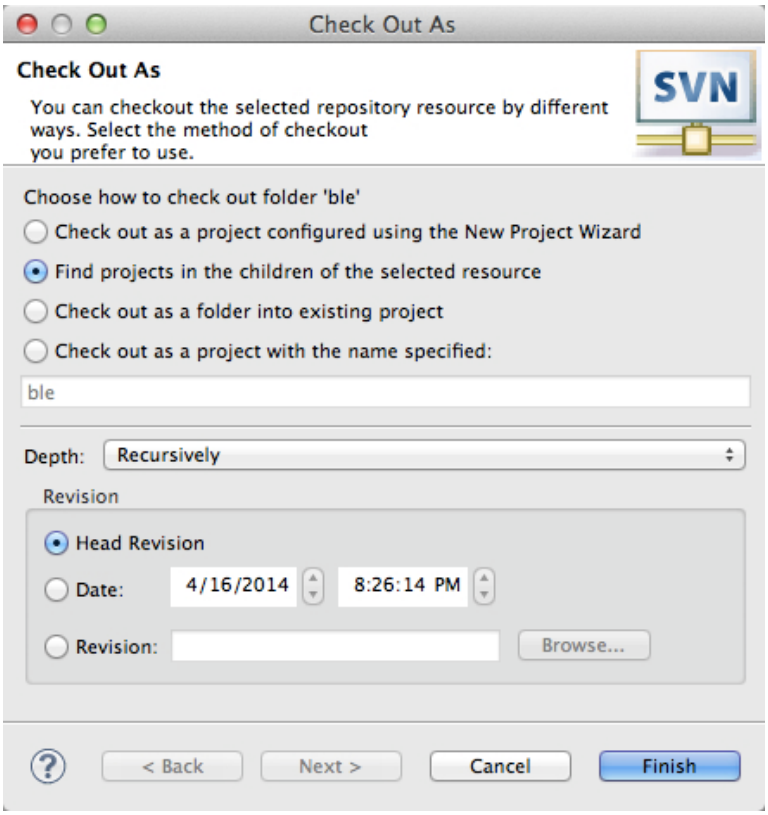


Click [Finish] to check out SVN project from the URL.

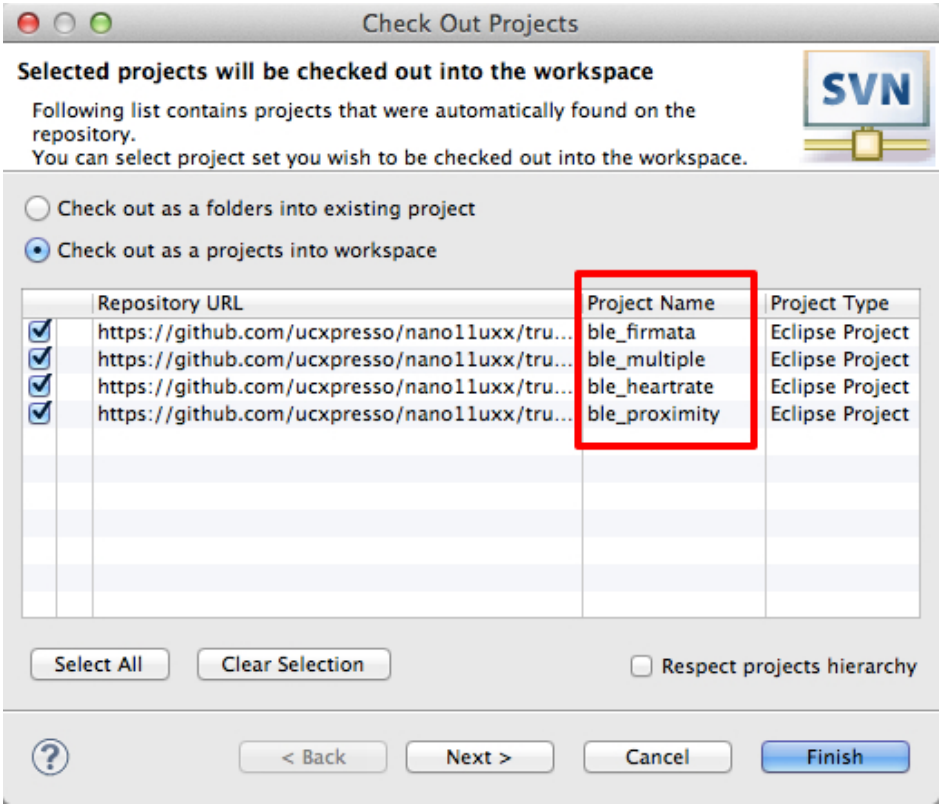


Check Out As,

select “Find projects in the children of the select resource”.



Select one (or more) projects which project do you want.



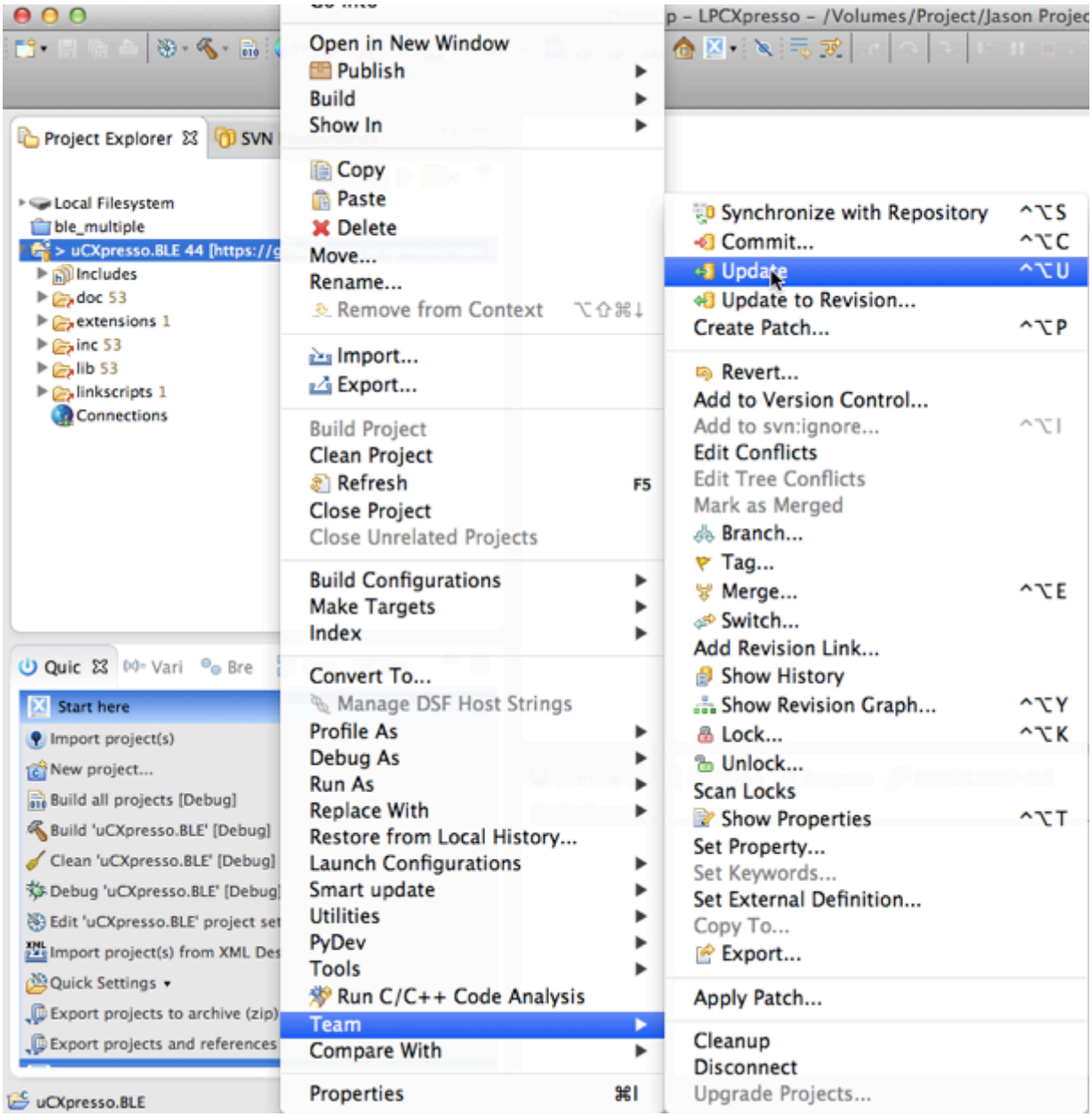
Click [Finish] to start to check out the selected projects into your workspace.

Appendix B. Update uCXpresso.NRF framework

When your uCXpresso.NRF is check-out from the trunk of SVN repository, you may use this appendix to update your uCXpresso.NRF framework.

Click right button of mouse on the “uCXpresso.NRF”,

in drop down menu , select Team > Update

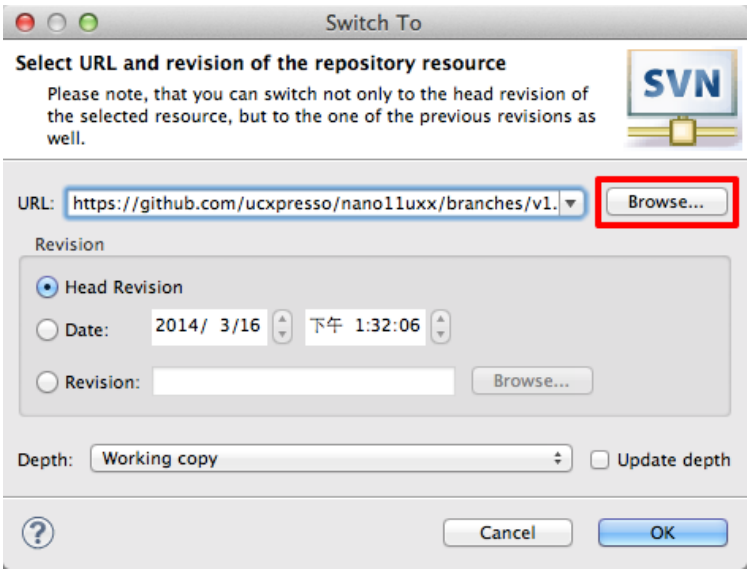


Appendix C. Switch uCXpresso.NRF version Branch

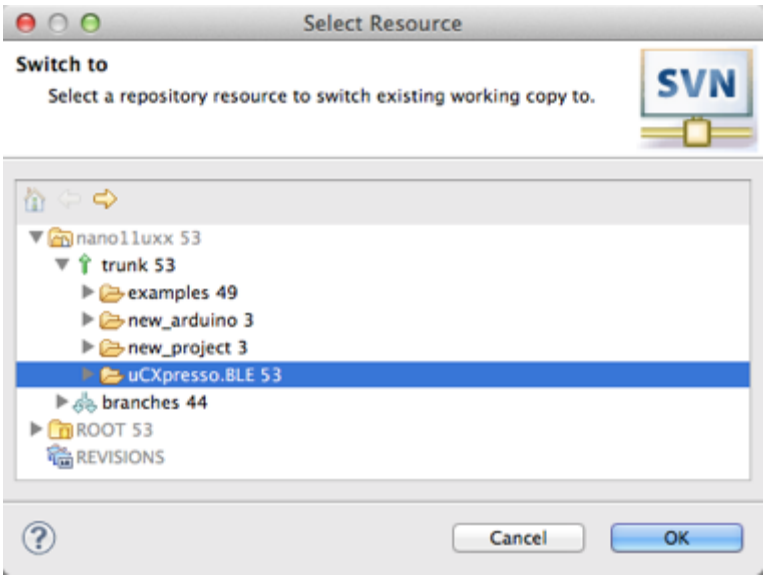
C.1 Switch SVN Branch

To switch the uCXpresso.NRF version branch, click the right button of mouse on the “uCXpresso.NRF” in Project Explorer View.

Select drop down menu “Team” > “Switch…” :



Click [Browse…] select resource (ex. select to trunk/uCXpresso.NRF)



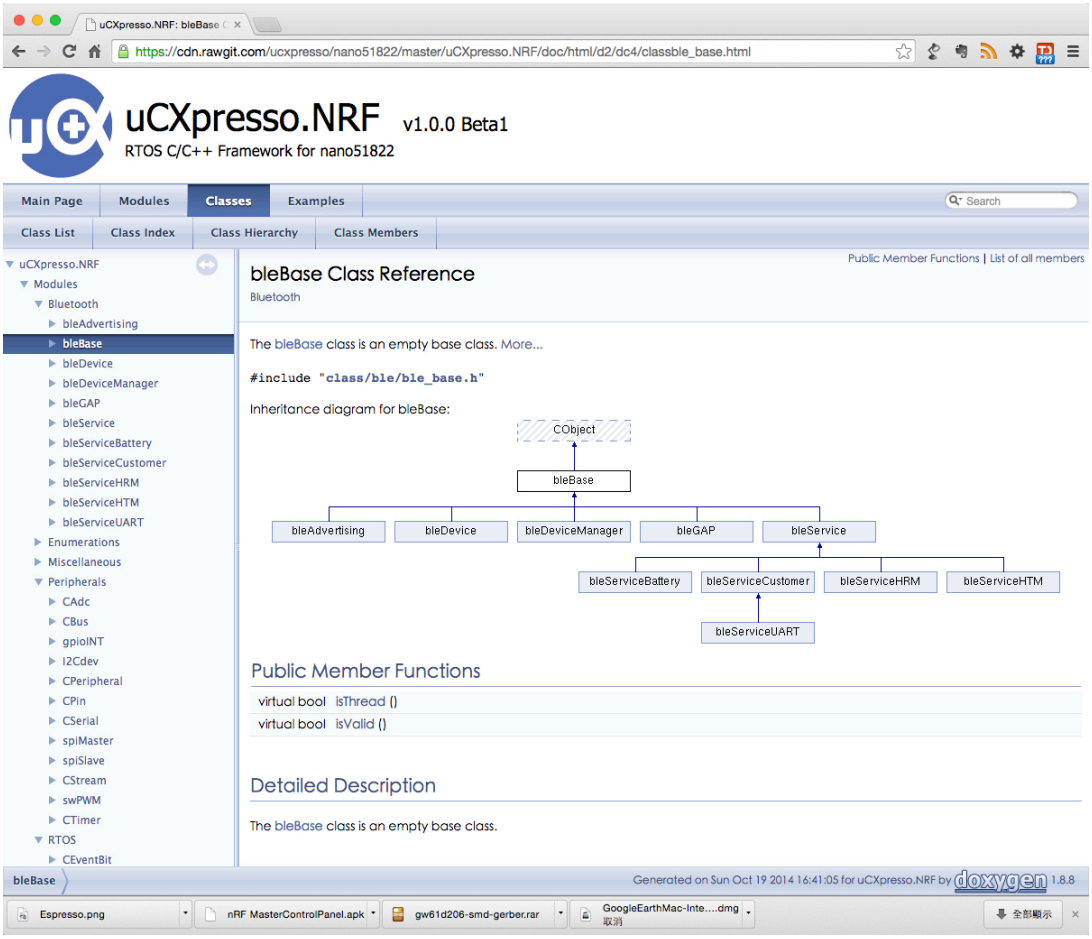
Click [OK] on “Select Resource” > Click [OK] on “Switch To”

Final, see the Appendix A to update the uCXpresso.NRF framework.

Appendix D. Class Manual

D.1 Online Class Manual

https://cdn.rawgit.com/ucxpresso/nano51822/master/documents/doxygen/html/index.html



Appendix E. The uCXpresso.NRF License

Hardware		nano51822	nRF51822
License type	Free	module	chip
Can use in commercial application	No	Yes	Yes
How number of the Task	1	Unlimited	Unlimited
How number of the BLE service	1	Unlimited	Unlimited
SoftDevice Version	Must be 7.1	7.1 or later	7.1 or later
App RAM size	6KB	Unlimited[1]	Unlimited[1]
App Code Size	Unlimited[1]	Unlimited[1]	Unlimited[1]
Peripherals framework (I/O)	Unlimited	Unlimited	Unlimited
Price	Free	Contact us	Contact us

Note:

1. Limit by chip condition.
2. The nano51822 datasheet.
3. Another license requirement, please contact us : service@embeda.com.tw