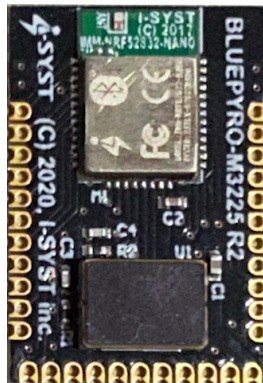


USER GUIDE

Bluepyro-M3225



Version 1.0

Revision history

Version	Date	Note	Contributor(s)	Approver
1.0	15 April 2021	Initial version	Ho Manh Tai	Nguyen Hoang Hoan

Copyright © 2019 I-SYST, all rights reserved.

3514, 1re Rue, Saint-Hubert, QC., Canada J3Y 8Y5

This document may not be reproduced in any form without, express written consent from I-SYST.

Contents

1. Introduction.....	5
1.1 Required components.....	5
2. Installation.....	5
2.1. Install Android Studio.....	5
2.2. Create new virtual device.....	8
2.3. Import Android Project.....	9
2.4. Run the Bluepyro-M3225 Android App Project with new Virtual device.....	11
2.5. Run on real Android device.....	12
2. Bluepyro-M3225 Firmware development with Eclipse IDE.....	13
2.1. Download Bluepyro-M3225 Firmware.....	13
2.2 Import firmware projects to Eclipse.....	13
2.3 Build Bluepyro-M3225 Firmware Projects.....	14
2.4 Connect Bluepyro-M3225 device.....	17
2.5 Debug Configurations.....	19
2.6 Flashing Firmware.....	22

1. Introduction

This document shows step-by-step to install Android Studio to develop Android apps with Bluepyro-M3225

1.1 Required components

The following are needed for a full development environment Bluepyro-M3225:

- Android Studio
- Eclipse
- Download source code Android app and firmware at [I-SYST/BluePyro: BluePyro \(github.com\)](https://github.com/I-SYST/BluePyro)

2. Installation

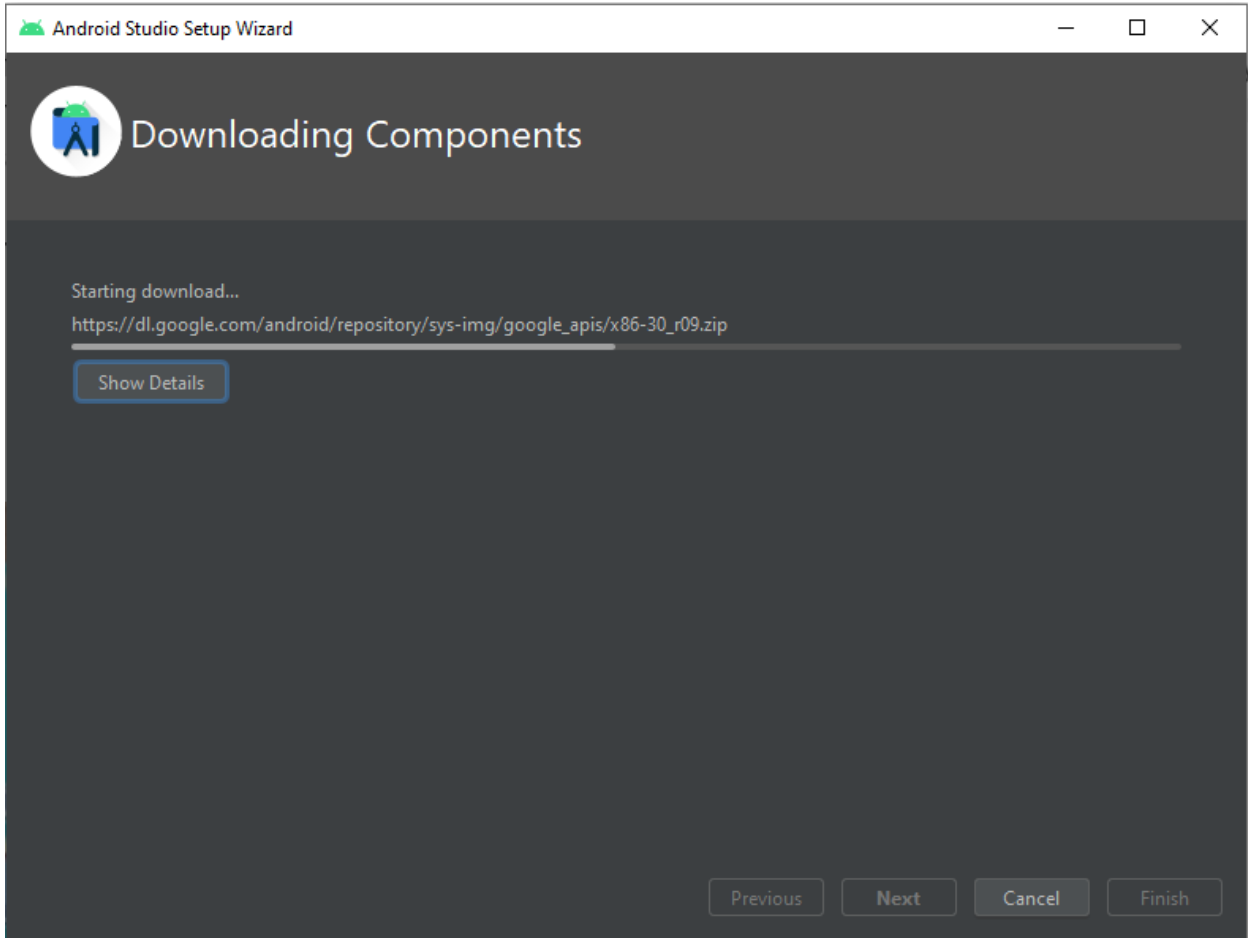
2.1. Install Android Studio

Download and install Android Studio to develop Android apps using Bluepyro-M3225 at the following link: <https://developer.android.com/studio/>

[Download Android Studio and SDK tools | Android Developers](#)

Follow the link below to install Android Studio: <https://developer.android.com/studio/install>

[Install Android Studio | Android Developers](#)



Android Studio Setup Wizard

— □ ×



Downloading Components

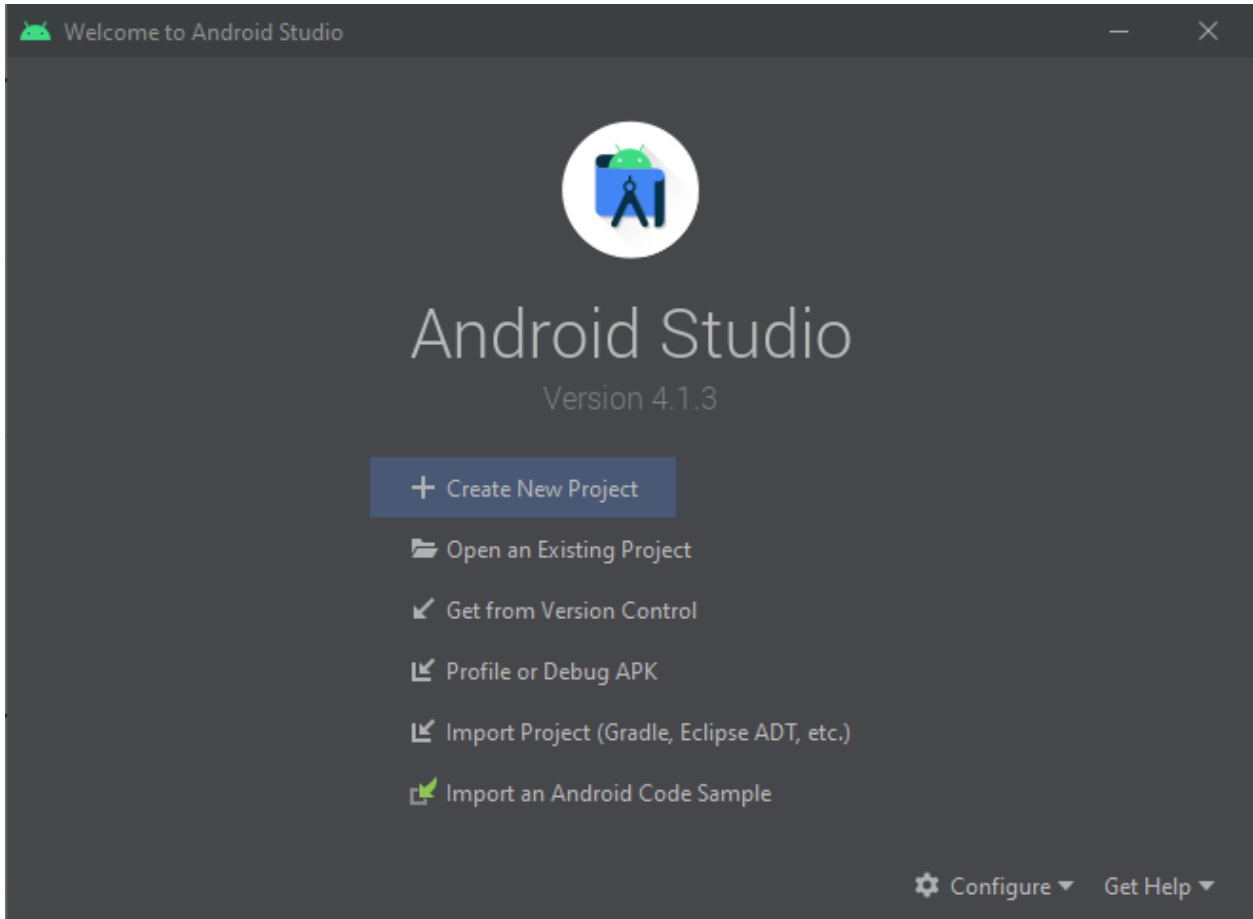
```
Downloading https://dl.google.com/android/repository/platform-30_r05.zip
"Install Android SDK Platform 30 (revision: 3)" ready.
Installing Android SDK Platform 30 in C:\Users\TAIHM\AppData\Local\Android\Sdk\platforms\android-30
"Install Android SDK Platform 30 (revision: 3)" complete.
"Install Android SDK Platform 30 (revision: 3)" finished.
Parsing C:\Users\TAIHM\AppData\Local\Android\Sdk\build-tools\30.0.3\package.xml
Parsing C:\Users\TAIHM\AppData\Local\Android\Sdk\emulator\package.xml
Parsing C:\Users\TAIHM\AppData\Local\Android\Sdk\extras\intel
  \Hardware_Accelerated_Execution_Manager\package.xml
Parsing C:\Users\TAIHM\AppData\Local\Android\Sdk\patcher\v4\package.xml
Parsing C:\Users\TAIHM\AppData\Local\Android\Sdk\platform-tools\package.xml
Parsing C:\Users\TAIHM\AppData\Local\Android\Sdk\platforms\android-30\package.xml
Parsing C:\Users\TAIHM\AppData\Local\Android\Sdk\sources\android-30\package.xml
Parsing C:\Users\TAIHM\AppData\Local\Android\Sdk\system-images\android-30\google_apis\x86
  \package.xml
```

Previous

Next

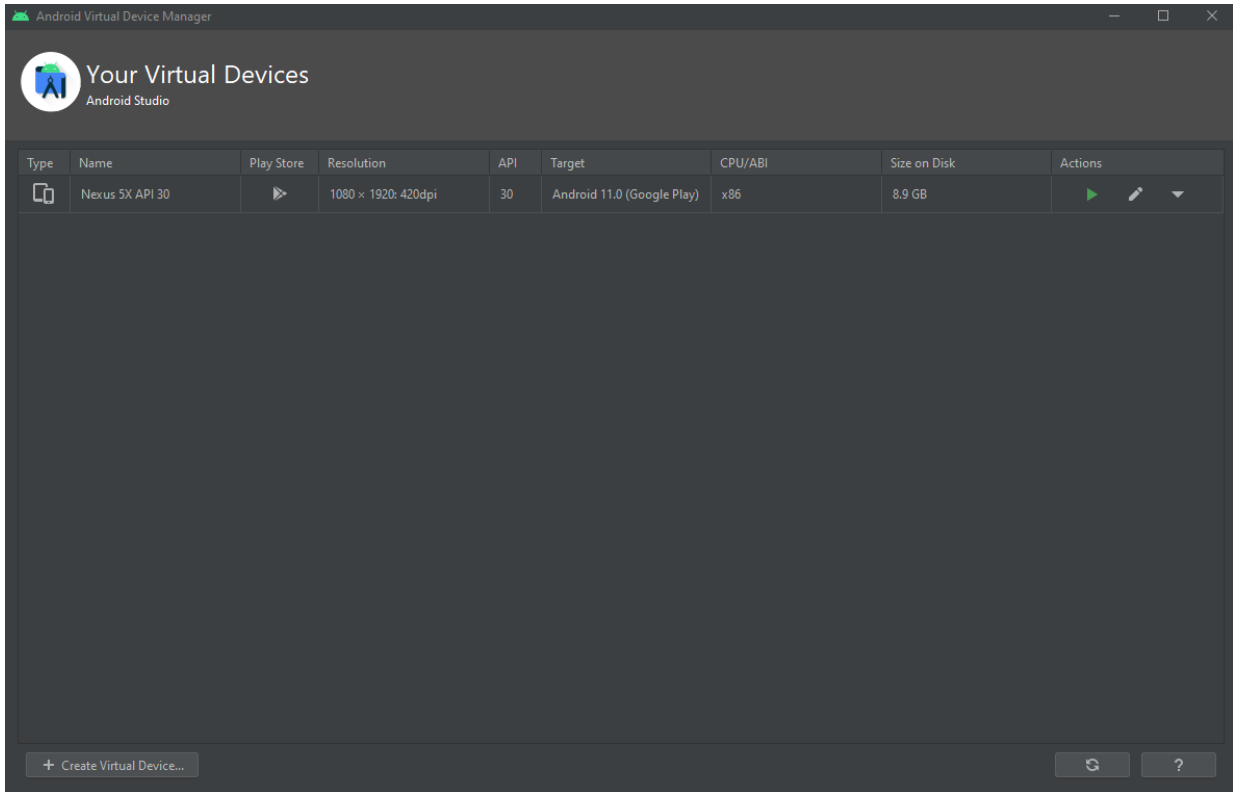
Cancel

Finish



2.2. Create new virtual device

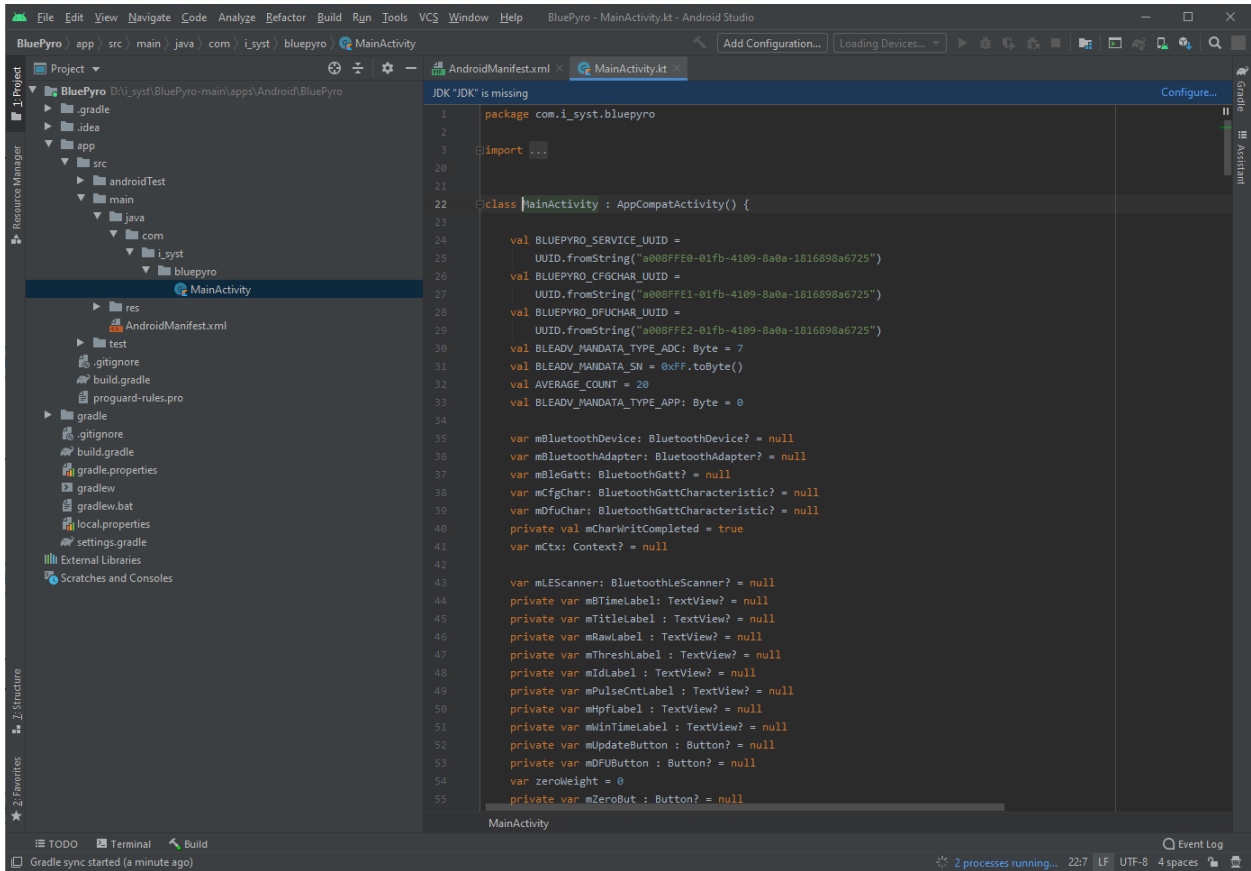
After finishing installation Android Studio, create a new Android virtual device in AVD Manager



2.3. Import Android Project

Download source code Android app at <https://github.com/I-SYST/BluePyro>

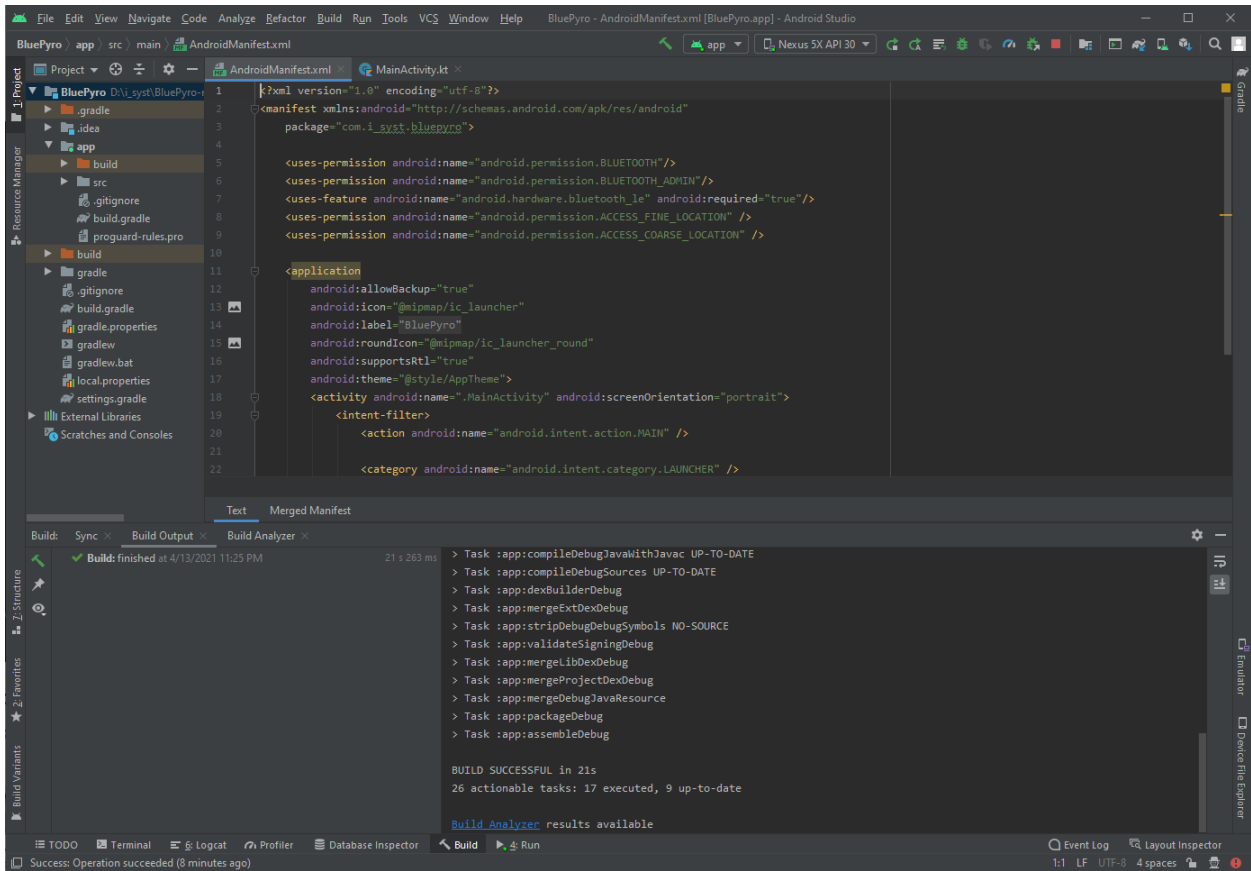
Import Android project BluePyro



The screenshot shows the Android Studio IDE with the MainActivity.kt file open. The code defines the package, imports, and class MainActivity. It includes several constants for UUIDs, BLEADV_MANDATA_TYPE, BLEADV_MANDATA_SN, AVERAGE_COUNT, and BLEADV_MANDATA_TYPE_APP. It also declares variables for BluetoothDevice, BluetoothAdapter, BluetoothGatt, BluetoothGattCharacteristic, Context, BluetoothScanner, and various TextView and Button instances.

```
1 package com.i_syst.bluepyro
2
3 import androidx.appcompat.app.AppCompatActivity
4
5 class MainActivity : AppCompatActivity() {
6
7     val BLUEPYRO_SERVICE_UUID =
8         UUID.fromString("a008FFE0-01fb-4109-8a0a-1816898a6725")
9     val BLUEPYRO_CFGCHAR_UUID =
10         UUID.fromString("a008FFE1-01fb-4109-8a0a-1816898a6725")
11     val BLUEPYRO_DFCHAR_UUID =
12         UUID.fromString("a008FFE2-01fb-4109-8a0a-1816898a6725")
13     val BLEADV_MANDATA_TYPE_ADC: Byte = 7
14     val BLEADV_MANDATA_SN = 0xFF.toByte()
15     val AVERAGE_COUNT = 20
16     val BLEADV_MANDATA_TYPE_APP: Byte = 0
17
18     var mBluetoothDevice: BluetoothDevice? = null
19     var mBluetoothAdapter: BluetoothAdapter? = null
20     var mBleGatt: BluetoothGatt? = null
21     var mCfgChar: BluetoothGattCharacteristic? = null
22     var mDfuChar: BluetoothGattCharacteristic? = null
23     private val mCharWriteCompleted = true
24     var mContext: Context? = null
25
26     var mLEScanner: BluetoothLEScanner? = null
27     private var mBTimeLabel: TextView? = null
28     private var mTitleLabel: TextView? = null
29     private var mRawLabel: TextView? = null
30     private var mThreshLabel: TextView? = null
31     private var mIdLabel: TextView? = null
32     private var mPulseCntLabel: TextView? = null
33     private var mHpfLabel: TextView? = null
34     private var mMinTimeLabel: TextView? = null
35     private var mUpdateButton: Button? = null
36     private var mDFUButton: Button? = null
37     var zeroWeight = 0
38     private var mZeroBut: Button? = null
39 }
```

2.4. Run the Bluepyro-M3225 Android App Project with new Virtual device



The screenshot displays the Android Studio interface. The main editor shows the `AndroidManifest.xml` file for the `BluePyro` app. The manifest includes permissions for Bluetooth, Bluetooth Admin, Fine Location, and Coarse Location. It also defines the application's icon, label, and theme, and declares the `MainActivity` as the main activity.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.i_syst.bluepyro">

    <uses-permission android:name="android.permission.BLUETOOTH"/>
    <uses-permission android:name="android.permission.BLUETOOTH_ADMIN"/>
    <uses-feature android:name="android.hardware.bluetooth_le" android:required="true"/>
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />

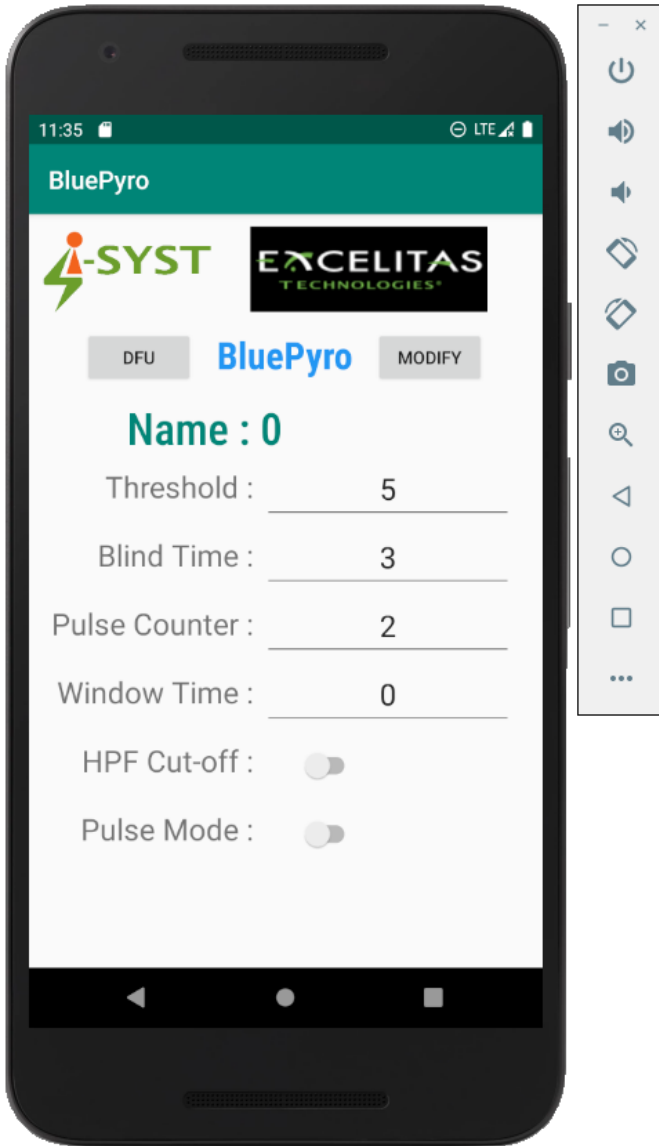
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity" android:screenOrientation="portrait">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

The Build Output console at the bottom shows the build process for the debug version of the app. The build is successful, taking 21 seconds. The console output includes the following tasks:

- Task :app:compileDebugJavaWithJavac UP-TO-DATE
- Task :app:compileDebugSources UP-TO-DATE
- Task :app:dexBuilderDebug
- Task :app:mergeExtDexDebug
- Task :app:stripDebugDebugSymbols NO-SOURCE
- Task :app:validateSigningDebug
- Task :app:mergeLibDexDebug
- Task :app:mergeProjectDexDebug
- Task :app:mergeDebugJavaResource
- Task :app:packageDebug
- Task :app:assembleDebug

The console also reports: `BUILD SUCCESSFUL in 21s` and `26 actionable tasks: 17 executed, 9 up-to-date`. A link to `Build Analyzer` results is provided.



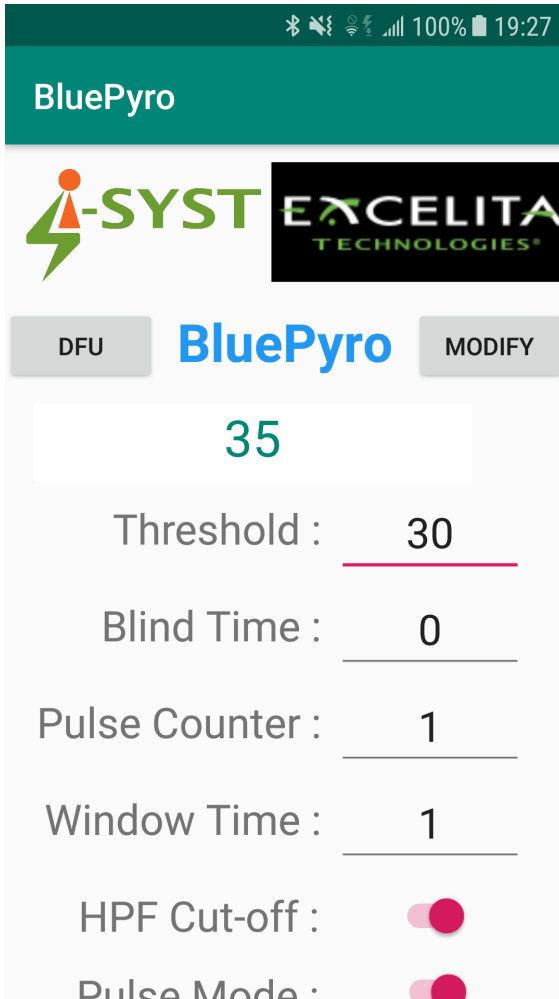
2.5. Run on real Android device

Now it's ready to run on your Android device to test with Bluepyro-M3225.

Connect your Android device and run the project on your device.

Test the Bluepyro-M3225 device with Bluepyro app on your device.

Now you are freely to develop any android apps with Bluepyro-M3225.



2. Bluepyro-M3225 Firmware development with Eclipse IDE

Please read the installation guide “Eclipse IDE in firmware development with Iosonata” for Eclipse installation at [Eclipse IDE in firmware development with IOsonata | I-SYST's Site](#).

In this section, we will show how to manually develop Bluepyro-M3225 Firmware by Eclipse.

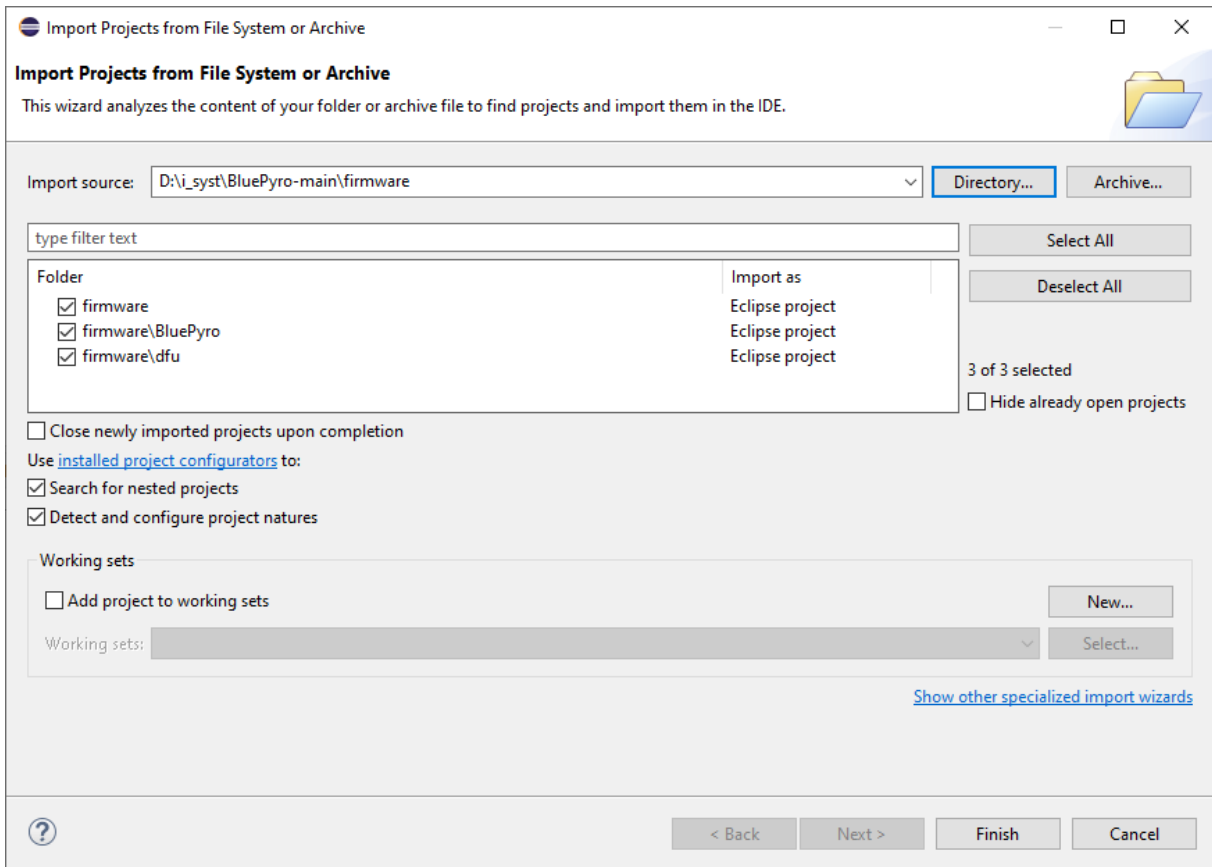
2.1. Download Bluepyro-M3225 Firmware

Download source code Bluepyro-M3225 firmware at:

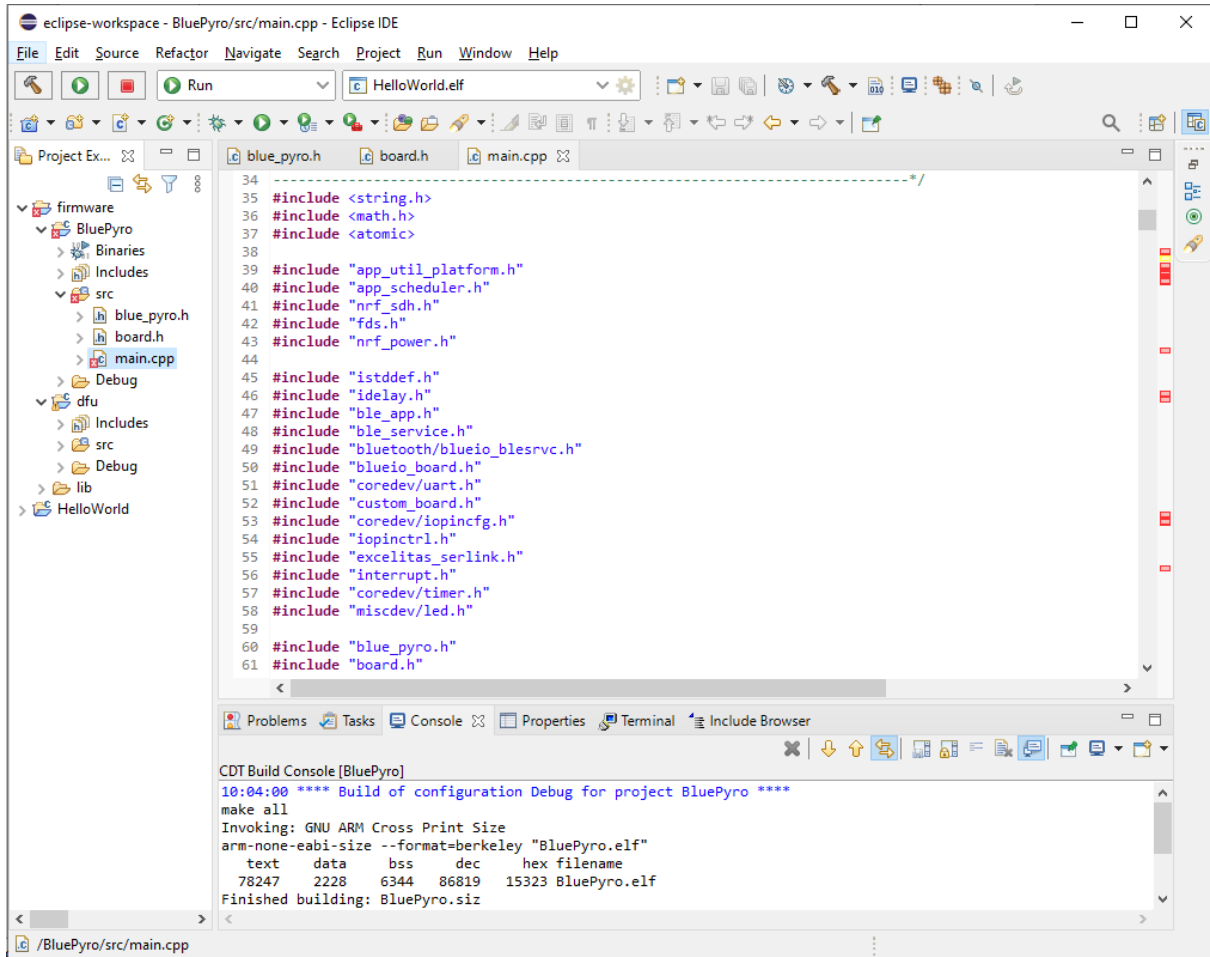
<https://github.com/I-SYST/BluePyro>

2.2 Import firmware projects to Eclipse

Select Open Projects from File System in File menu
Located the Bluepyro-M3225 Firmware directory.



2.3 Build Bluepyro-M3225 Firmware Projects



The screenshot shows the Eclipse IDE interface. The main editor displays the source code of `main.cpp` with the following content:

```
34 -----/
35 #include <string.h>
36 #include <math.h>
37 #include <atomic>
38
39 #include "app_util_platform.h"
40 #include "app_scheduler.h"
41 #include "nrf_sdh.h"
42 #include "fds.h"
43 #include "nrf_power.h"
44
45 #include "istddef.h"
46 #include "idelay.h"
47 #include "ble_app.h"
48 #include "ble_service.h"
49 #include "bluetooth/blueio_blesrvc.h"
50 #include "blueio_board.h"
51 #include "coredev/uart.h"
52 #include "custom_board.h"
53 #include "coredev/iopincfg.h"
54 #include "iopinctrl.h"
55 #include "excelitas_serlink.h"
56 #include "interrupt.h"
57 #include "coredev/timer.h"
58 #include "miscdev/led.h"
59
60 #include "blue_pyro.h"
61 #include "board.h"
```

The CDT Build Console at the bottom shows the following output:

```
CDT Build Console [BluePyro]
10:04:00 **** Build of configuration Debug for project BluePyro ****
make all
Invoking: GNU ARM Cross Print Size
arm-none-eabi-size --format=berkeley "BluePyro.elf"
  text  data   bss   dec   hex filename
78247  2228   6344  86819  15323 BluePyro.elf
Finished building: BluePyro.siz
```

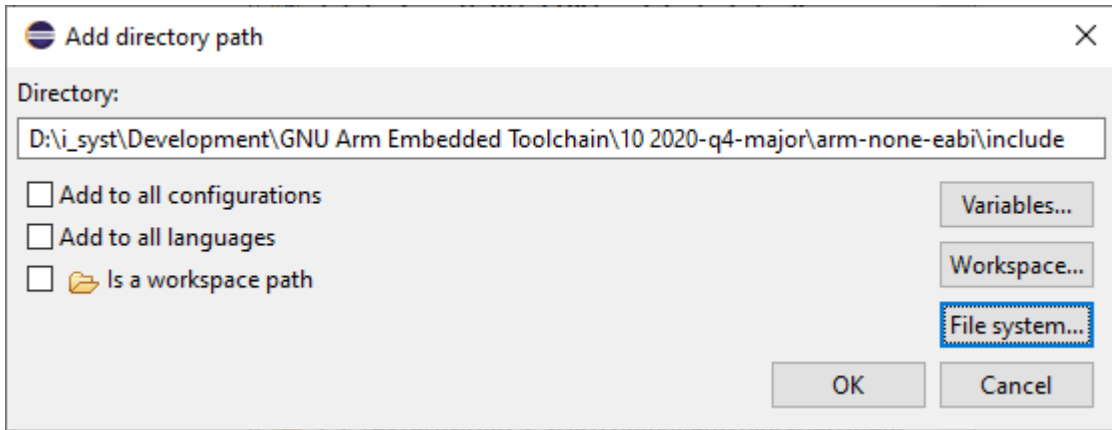
You may need to add some more libraries

Right click on Bluepyro Project, select Project Properties→ C/C++ General→ Paths and Symbols→ Includes tab. Click Add button: Add directory path, select File system

The screenshot shows the 'Properties for BluePyro' dialog box, specifically the 'Paths and Symbols' tab. The configuration is set to 'Debug [Active]'. The 'Include directories' list contains the following entries:

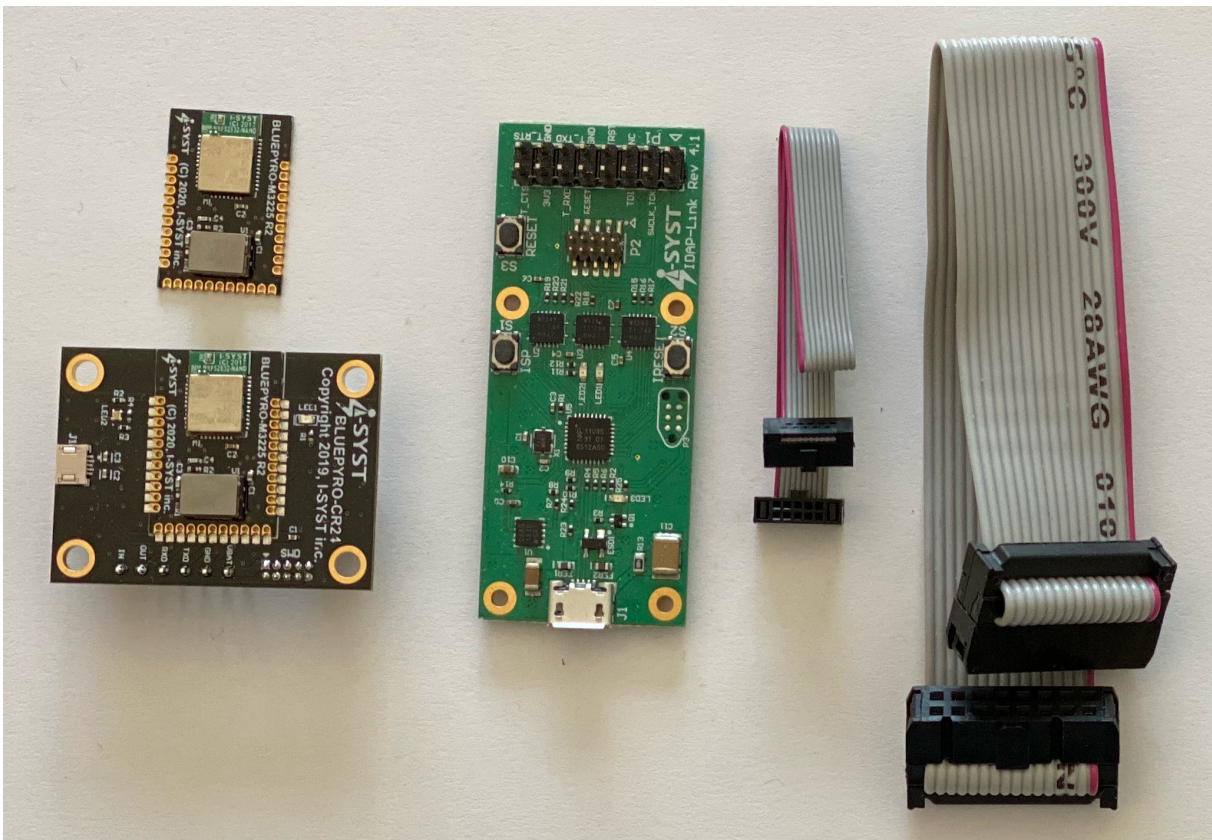
- ./src
- ../../../../IOsonata/ARM/Nordic/nRF52/nRF52832/lib/incl...
- ../../../../IOsonata/ARM/Nordic/nRF52/include
- ../../../../IOsonata/ARM/Nordic/include
- ../../../../IOsonata/ARM/include
- ../../../../IOsonata/ARM/CMSIS/Core/Include
- ../../../../IOsonata/include
- ../../../../external/nRF5_SDK/integration/nrfx
- ../../../../external/nRF5_SDK/modules/nrfx
- ../../../../external/nRF5_SDK/modules/nrfx/mdk
- ../../../../external/nRF5_SDK/modules/nrfx/hal
- ../../../../external/nRF5_SDK/components/ble/common
- ../../../../external/nRF5_SDK/components/libraries/atomic
- ../../../../external/nRF5_SDK/components/libraries/balloc
- ../../../../external/nRF5_SDK/components/libraries/fds
- ../../../../external/nRF5_SDK/components/libraries/mutex
- ../../../../external/nRF5_SDK/components/libraries/sched...
- ../../../../external/nRF5_SDK/components/libraries/exper...
- ../../../../external/nRF5_SDK/components/libraries/log
- ../../../../external/nRF5_SDK/components/libraries/log/src
- ../../../../external/nRF5_SDK/components/libraries/mem...
- ../../../../external/nRF5_SDK/components/libraries/pwr_...
- ../../../../external/nRF5_SDK/components/libraries/strerr...

At the bottom of the dialog, the 'Apply and Close' button is highlighted with a blue border.



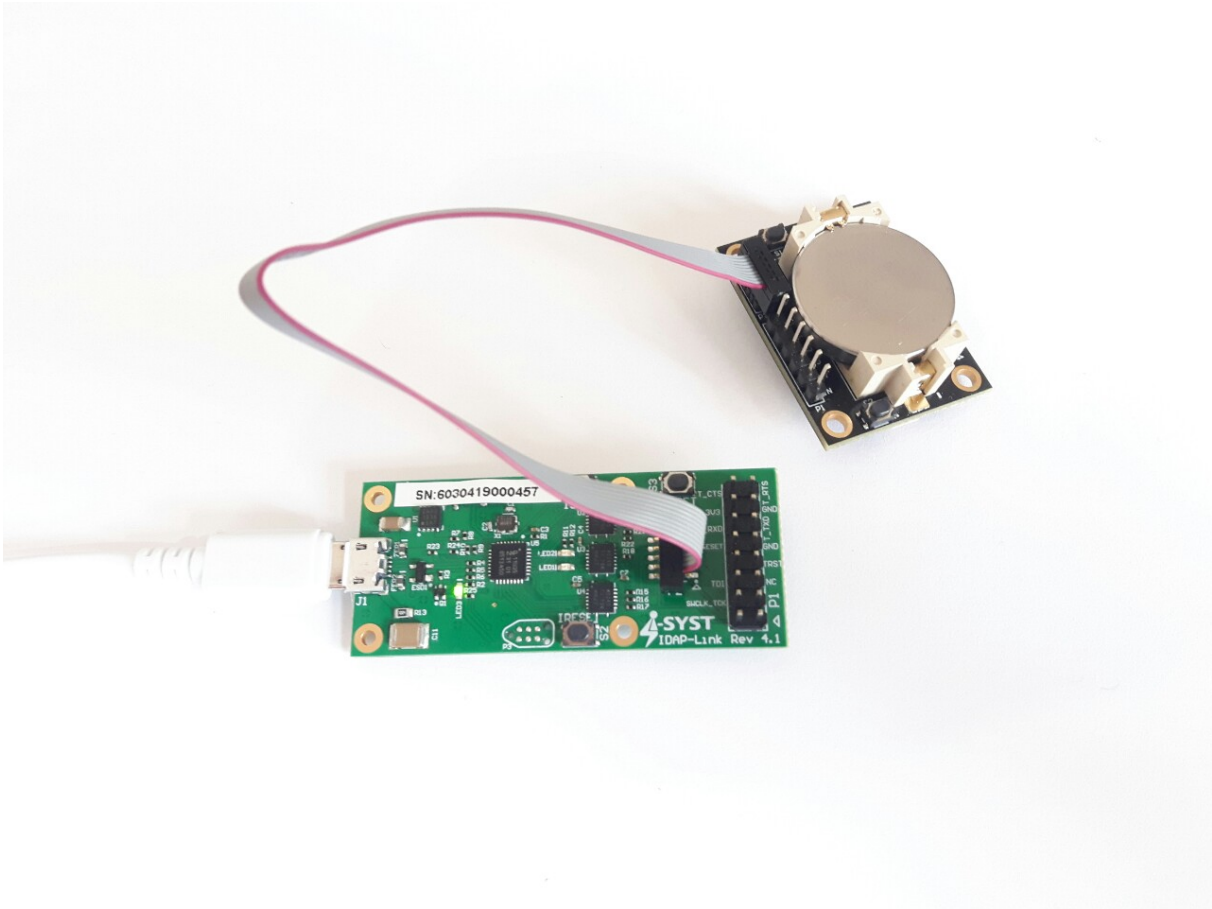
Right click on Bluepyro-M3225 Firmware project, select Build Project . Now you can build and manually develop Bluepyro-M3225 Firmware by yourself.

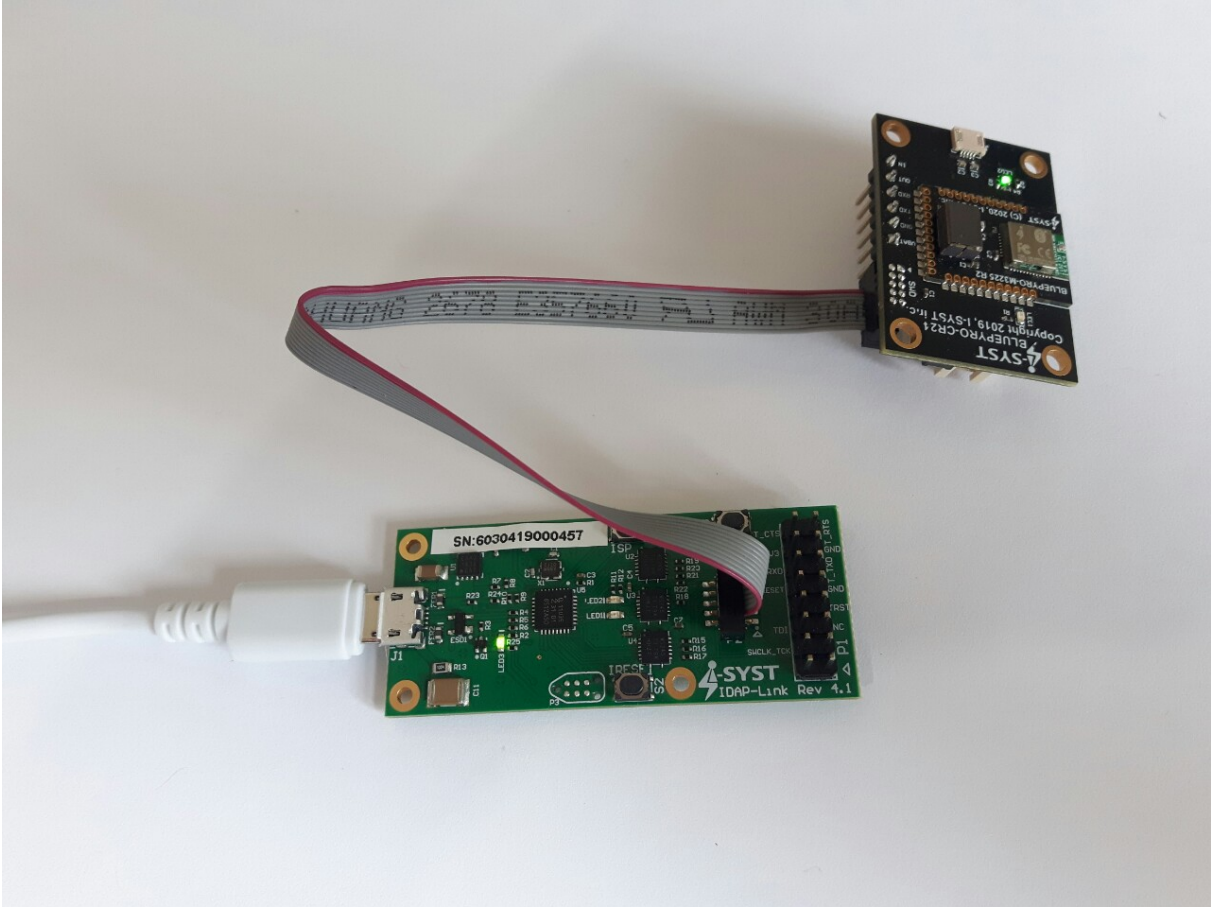
2.4 Connect Bluepyro-M3225 device



Prepare Bluepyro-M3225 development kit as the figure above include Bluepyro-M3225 board, IDAP-Link.

Connect Bluepyro-M3225 to computer.

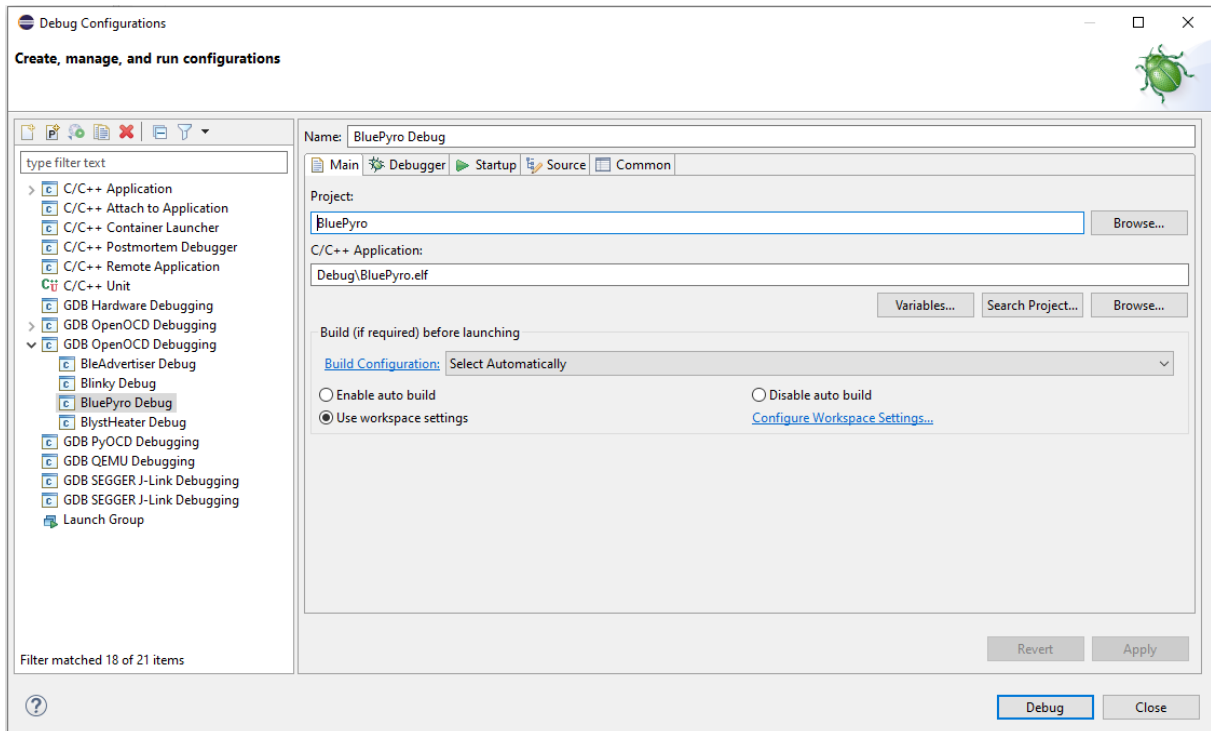
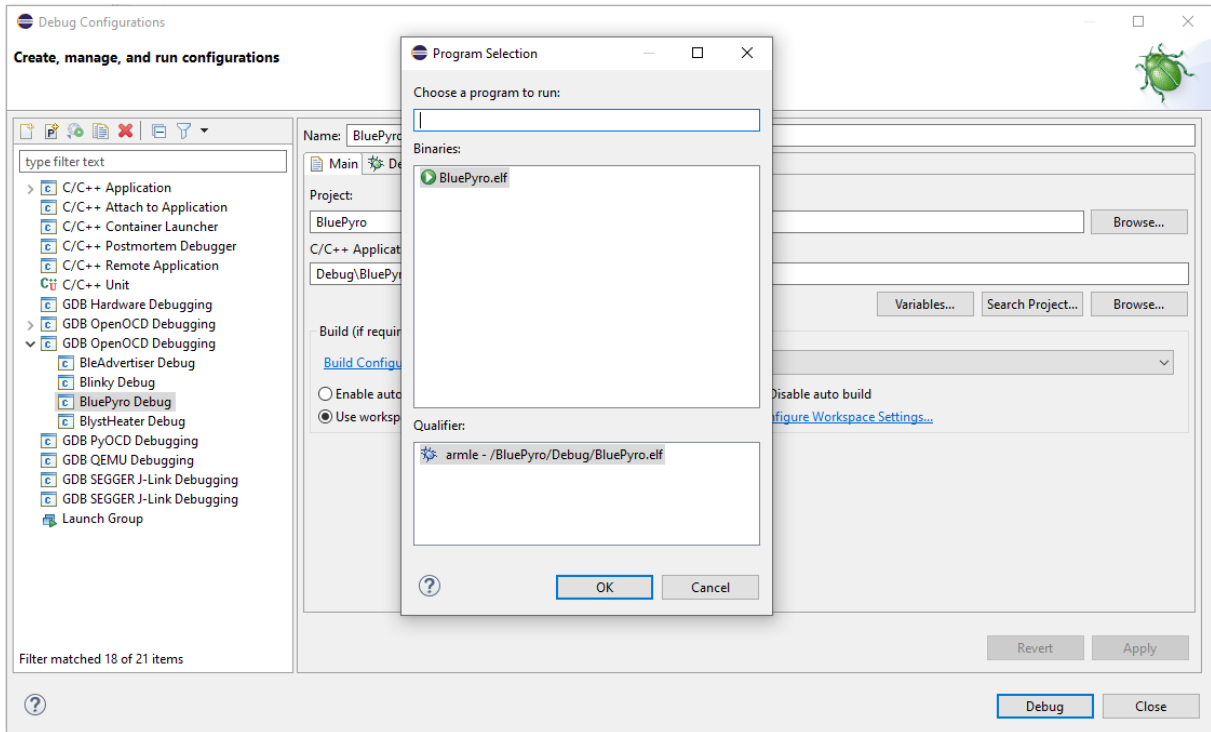




2.5 Debug Configurations

In order to debug the Bluepyro-M3225 firmware, we have to set the Debug Configurations as following:

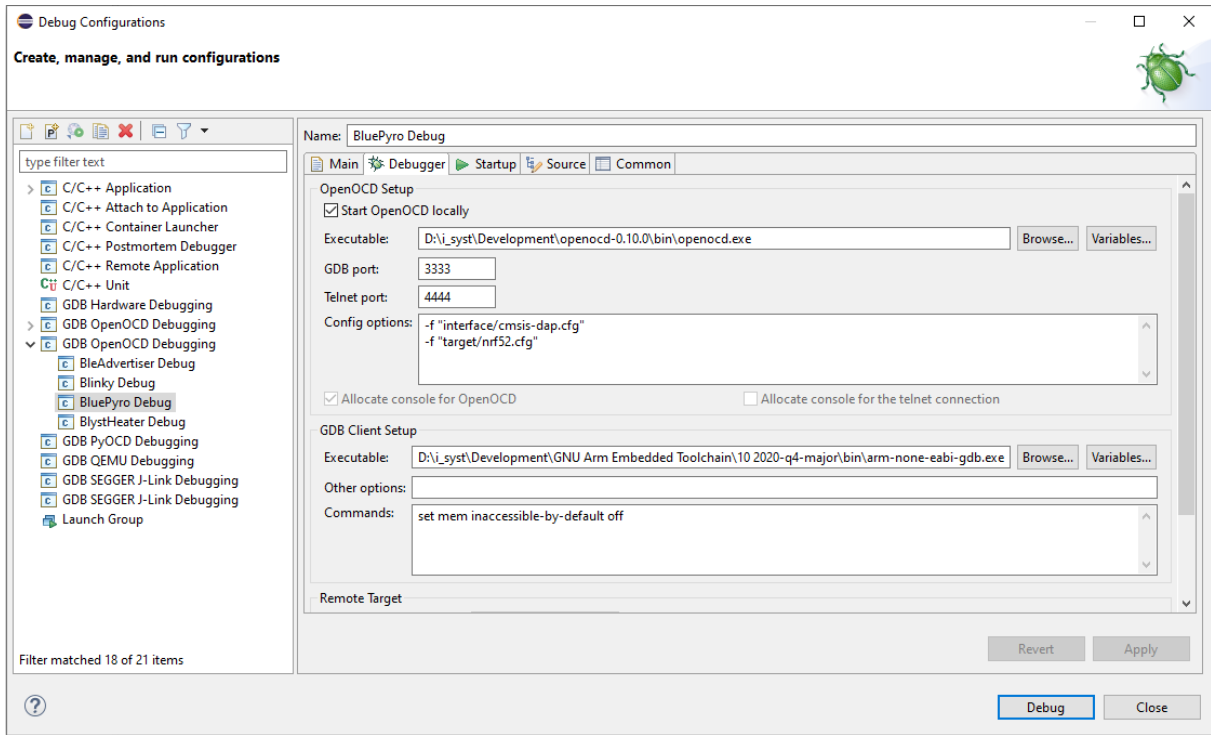
In the Main tab, select C/C++ Application by clicking Search Project. Select BluePyro.elf file.



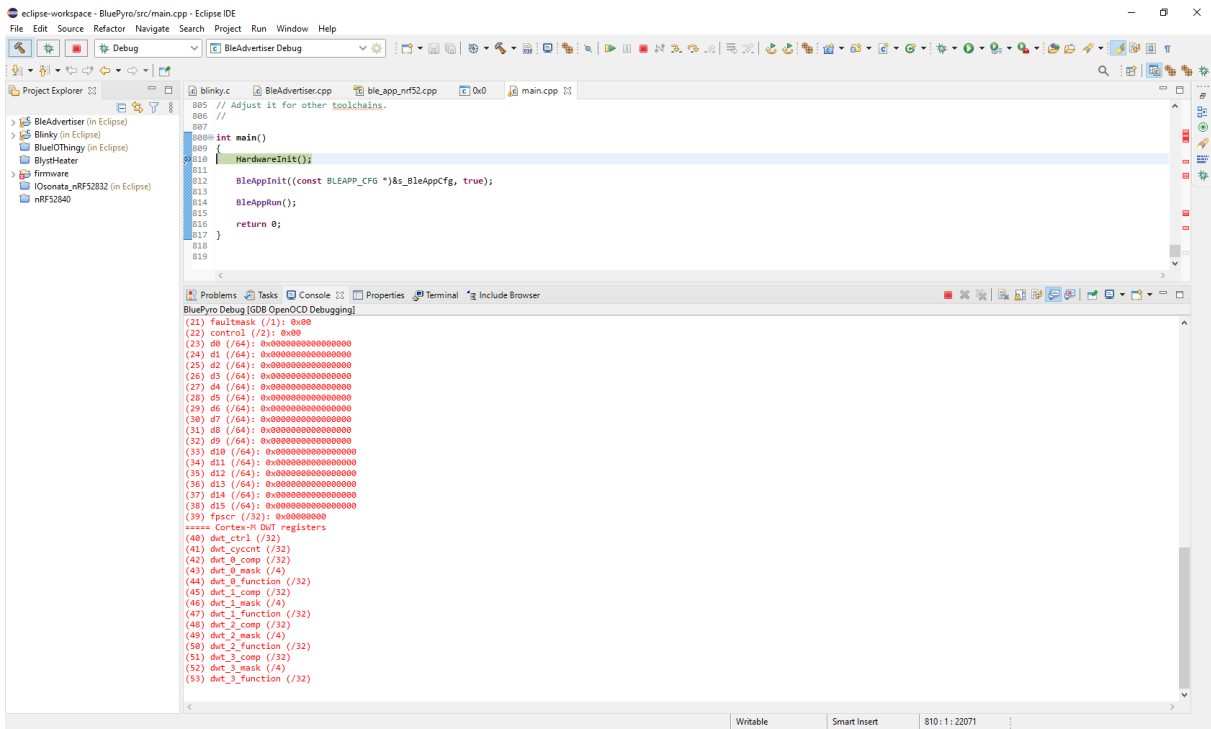
In the Debugger tab, set the Executable by browsing to the OpenOCD executable file.

Set the Config Option:

```
-f "interface/cmsis-dap.cfg"
-f "target/nrf52.cfg"
```



Then click Debug button.



After you start the debugger, it will stop at main(). Now you can debug the firmware by clicking the

step button (F5, F6) to trace your source code line by line.

2.6 Flashing Firmware

Click Run button to run the firmware on your device.

Note:

Make sure that the softdevice is flashed first. Using Use IDAPnRFProg to flash NRF softdevice using IDAP-Link. Download here [IDAP-Link/M - Browse /Windows at SourceForge.net](#)

Run IDAPnRFProg by following command line:

```
$. \IDAPnRFProg.exe .\external\nRF5_SDK\components\softdevice\s132\  
hex\s132_nrf52_7.2.0_softdevice.hex
```