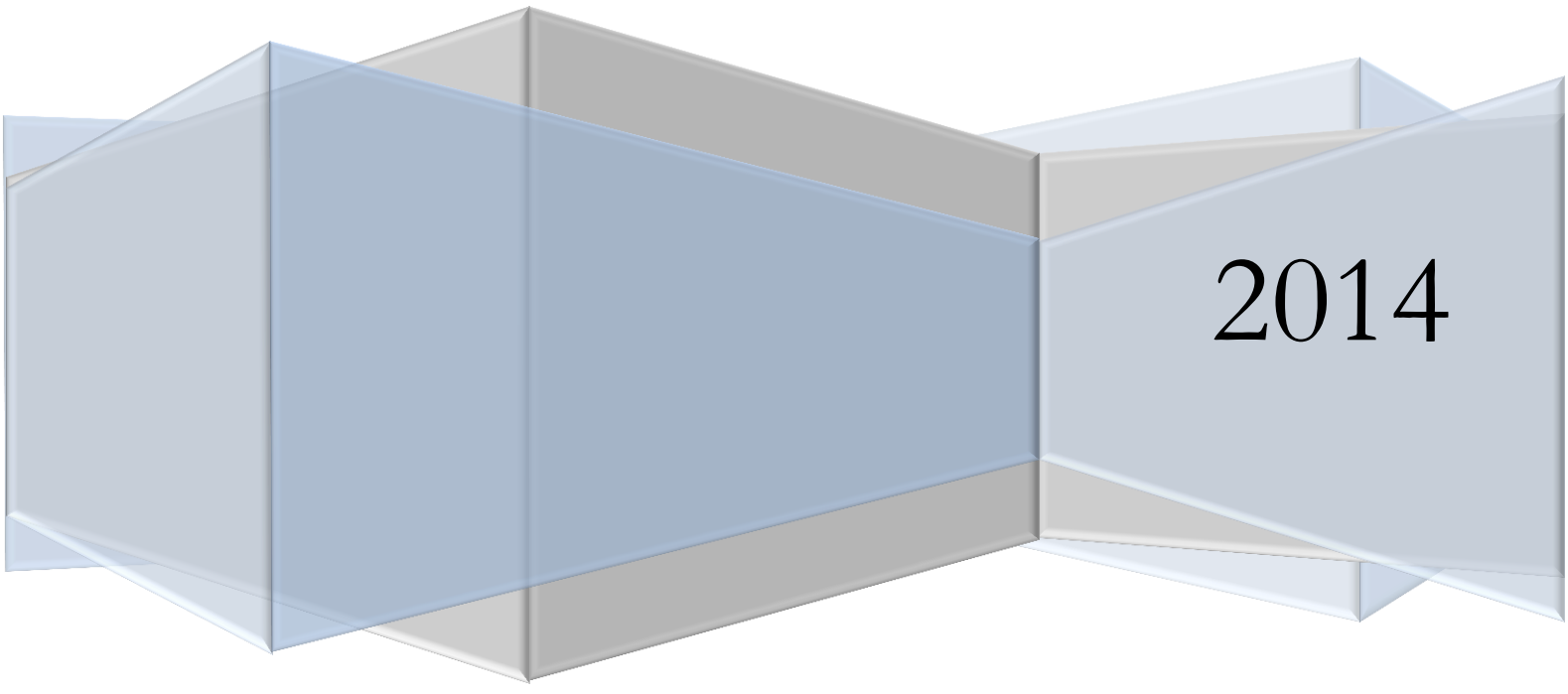
EIA-FR

Développement du nRF51 avec GCC et Eclipse sur Linux

Ceylan Ziran



Sommaire

Table des Figures

1)Outils Requis

Les outils suivant ont été utilisé dans ce tutorial. Le numéro de version de chaque outils est donnée en référence, mais comme les logiciels sont souvent mis à jour, vous ne devez pas forcement utiliser les même versions.

J-Link ARM version 4.84f

Eclipse plug-in for making nordic nrf51822 projects et le décompresser

Une fois décompresser il faudra copier le fichier "99-jlink.rules" donner dans le dossier /etc/udev/rules :

En utilisant cette commande sudo cp 99-jlink.rules /etc/udev/rules.d/

Télécharger et Installer depuis : http://www.segger.com/jlink-software.html .

Eclipse IDE for C/C++ Developers version Kepler Service Release 2

Télécharger et Installer depuis : http://www.eclipse.org/downloads/ .

GNU Tools for ARM Embedded Processors

Télécharger et Installer : https://launchpad.net/gcc-arm-embedded/+download

Comme les executables sont des application 32-bit, si vous utilisez un système 64-bit vous devez installer les librairies 32-bit suivantes :

sudo apt-get install lib32z1

sudo apt-get install lib32ncurses5

sudo apt-get install lib32bz2-1.0

Téléchargez la dernière version de gcc-arm-none-eabi et décompressez la dans /usr/local/

Grace aux commandes suivantes :

cd /usr/local
sudo tar xjf <chemin d'accès>/gcc-arm-none-eabi-4_8-2014q2-20140609-linux.tar.bz2

Le resultat devrait etre un fichier du genre /usr/local/gcc-arm-none-eabi-4_8-2014q1

Testez si le Compiler est fonctionnel

$ /usr/local/gcc-arm-none-eabi-4_8-2014q2/bin/arm-none-eabi-gcc –version

Vous devriez avoir ce qui suit

arm-none-eabi-gcc (GNU Tools for ARM Embedded Processors) 4.8.4 20140526 (release) [ARM/embedded-4_8-branch revision 211358]

nRF518 SDK

Utilisez le SDK fournis ou Télécharger et Installer depuis :
http://www.nordicsemi.com/eng/Products/Bluetooth-R-low-energy/nRF51822-Evaluation-Kit

Outils additionnels utilisé dans ce document (Suivre les étapes - Réglages d'Eclipse):

- CDT Main Features
- GDB Hardware Debugging
- GNU ARM Eclipse Plug-Ins
- Nordic Eclipse Plug-In

a) Réglages d'Eclipse

1. Démarrer Eclipse.
2. Cliquez sur **HELP** et sélectionnez **Install New Software.**
3. Ajoutez http://download.eclipse.org/tools/cdt/releases/kepler/ à la liste des répertoires (remplacer « kepler » si vous avez installé une autre version d'Eclipse).
4. Installez **CDT Main Features** et **GDB Hardware Debugging**.
5. Répétez l'étape 2 et ajoutez http://gnuarmeclipse.sourceforge.net/updates à la liste des répertoires.
6. Installez **GNU ARM Eclipse Plug-ins**.
7. Répétez l'étape 2 et ajoutez http://nrf51osx.sourceforge.net/updates à la liste des répertoires.
8. Installez **GNU ARM Nordic Templates**
9. **Apres avoir tout installer un petit redemarage d'eclipse est le bienvenu**

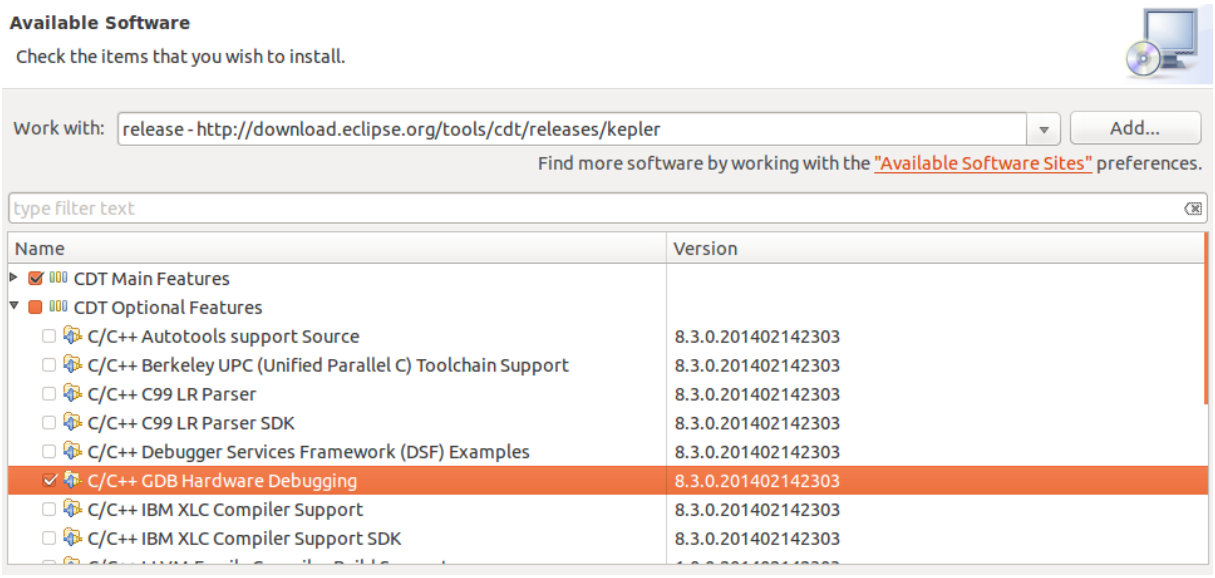


Figure 1: Fenêtre pour l'installation des plug-ins

2) Création d'un projet simple sous Eclipse

Créer un nouveau projet C

1. Dans le menu **File**, cliquez sur **New** puis sélectionnez **C Project**.
2. Donnez-lui un nom.
3. Dans la fenêtre Project type cliquez sur **Executable** et sélectionnez **Nordic NRF51822 C Project**.
4. Dans la fenêtre **Toolchains** sélectionnez **Cross ARM GCC.**

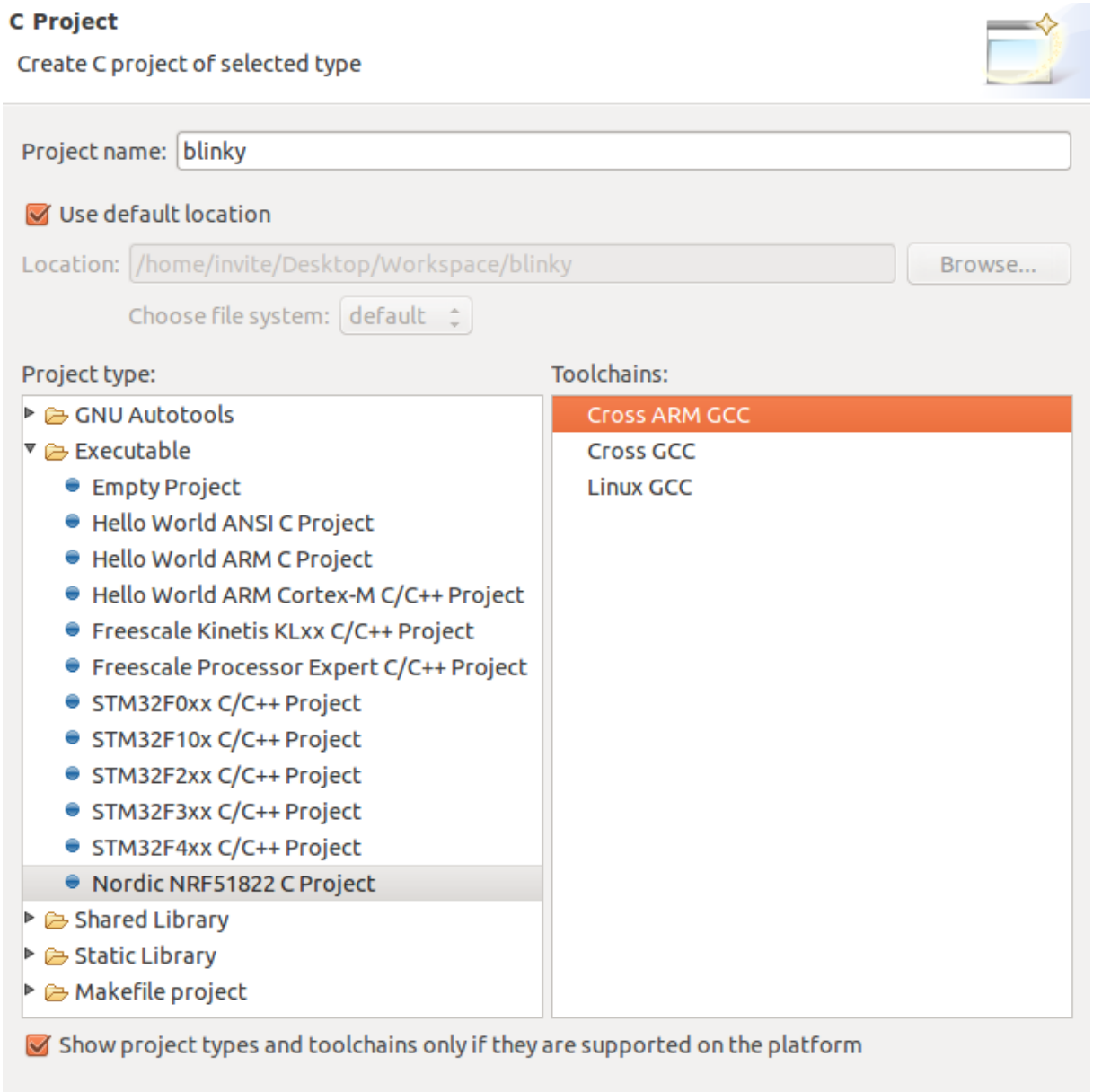


Figure 2: Fenêtre de création d'un nouveau projet C

La prochaine fenêtre vous permet de définir votre device ainsi que le Softdevice utilisé

5. Sélectionnez vos réglages ainsi que le chemin du dossier nordic sdk.
6. Puis faites next deux fois.

Sur la prochaine fenêtre vous devrez choisir le chemin de GNU Tools ARM

7. Le chemin devrait être de la forme « /usr/local/gcc-arm-none-eabi-4_8-2014q2 »
8. Le projet est créé avec un exemple qui fait clignoter les leds

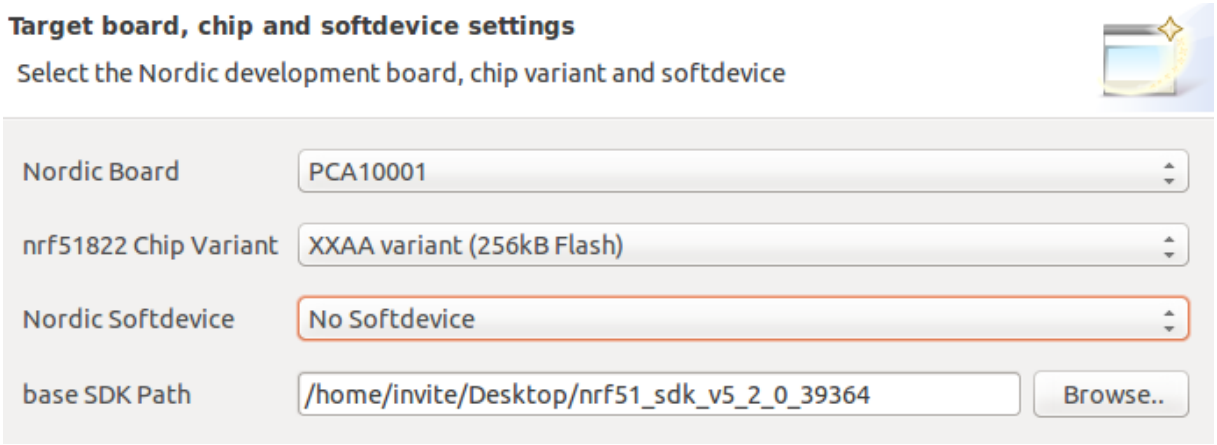


Figure 3: Choix des caractéristiques de notre carte

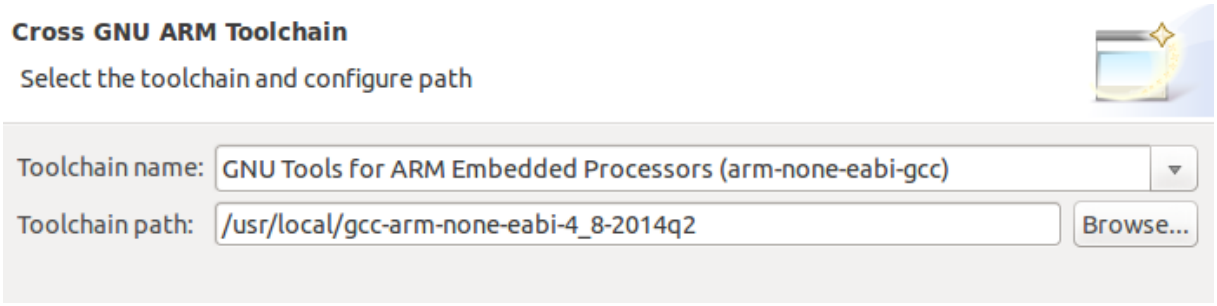


Figure 4 : choix du chemin pour GNU Tools ARM

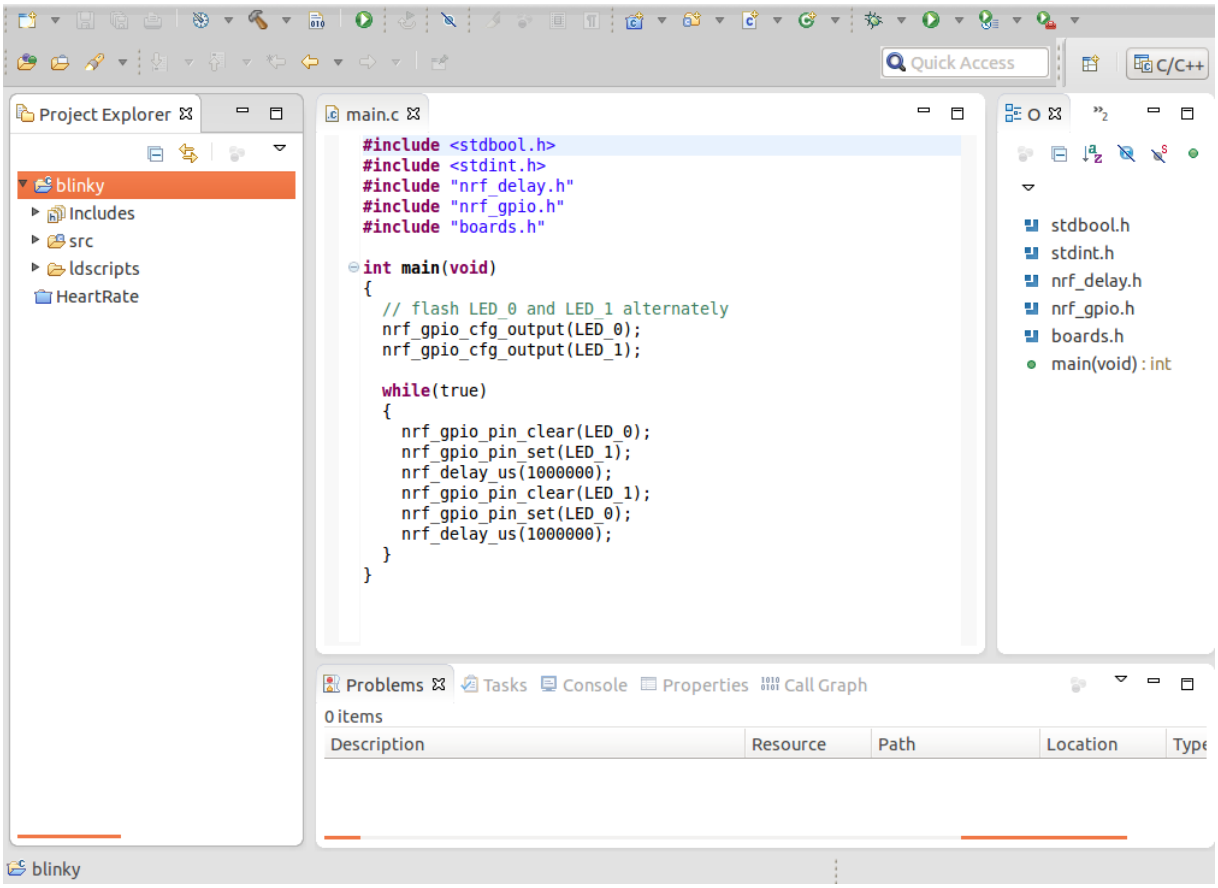


Figure 5: exemple de projet

Voici a quoi ressemble le projet. Dans le dossier src vous avez votre main ainsi que les deux fichiers qui servent au démarrage de la carte. Dans ldscripts vous avez les script linker qui définissent la mémoire et les sections pour le mapping de la mémoire.

3) Créer un projet BLE sous eclipse

1. Suivez les Etape pour créer un projet simple sous Eclipse mais choisissez le Softdevice que vous utiliserez
2. Lorsque le projet est créer vous pouvez modifier le main ainsi qu'ajouter des fichiers dans src.
3. Dans **Project Explorer** cliquez droit sur **votre projet** et sélectionnez **Properties**.
4. Dans le menu de gauche, déroulez **C/C++ General** et cliquez sur **Paths and Symbols**
5. Ajoutez les chemins d'accès des **Includes nécessaires** du **SDK nRF**

Par exemple pour l'exemple ble_app_hrs on inclut

- ~/nRF51 SDK_v5.2.0.39364/nrf51822/Include
- ~/nRF51 SDK_v5.2.0.39364/nrf51822/Include/gcc
- ~/nRF51 SDK_v5.2.0.39364/nrf51822/Include/app_common
- ~/nRF51 SDK_v5.2.0.39364/nrf51822/Include/ble
- ~/nRF51 SDK_v5.2.0.39364/nrf51822/Include/ble/ble_services
- ~/nRF51 SDK_v5.2.0.39364/nrf51822/Include/boards
- ~/nRF51 SDK_v5.2.0.39364/nrf51822/Include/s110
- ~/nRF51 SDK_v5.2.0.39364/nrf51822/Include/sd_common

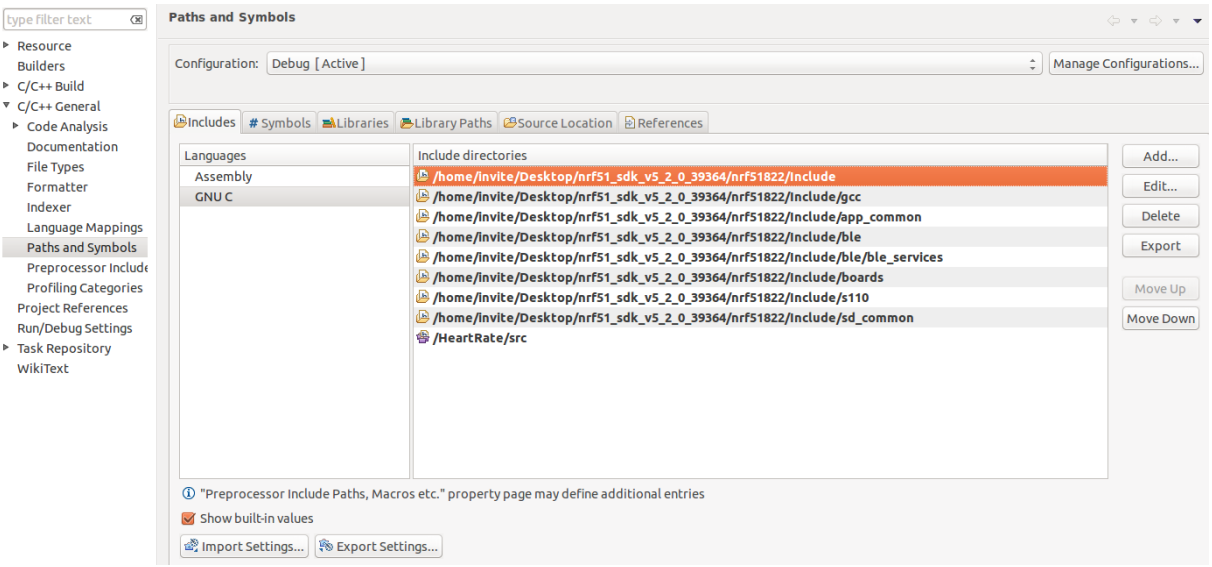


Figure 6: Fenêtre pour les Includes

6. Puis dans **Source Location** Il faut aussi "linker" les sources .c de ces .h
7. Cliquez sur **Link Folder** puis cochez **Link to folder in the file system**
8. Cliquer sur **browse** et ajoutez le chemin pour les sources du nrf51822

Par exemple : ~/nrf51_sdk_v5_2_0_39364/nrf51822/Source

9. Une fois le dossier Source lié on doit exclure du Debug et du release certain dossier et fichiers qui ne sont pas utile dans notre cas.Faire un click **droit>Ressource Configurations >Exclure from Build** sur le dossier a exclure.

Les dossiers et fichiers à exclure pour l'exemple ble_hrs_app sont :

- twi_master
- templates
- spi_slave
- spi_master
- simple_uart
- gzp
- ext_sensors
- device_manager
- console
- rpc
- hci_transport.c
- hci_slip.c
- hci_mem_pool.c
- app_uart.c
- app-uart_fifo.c

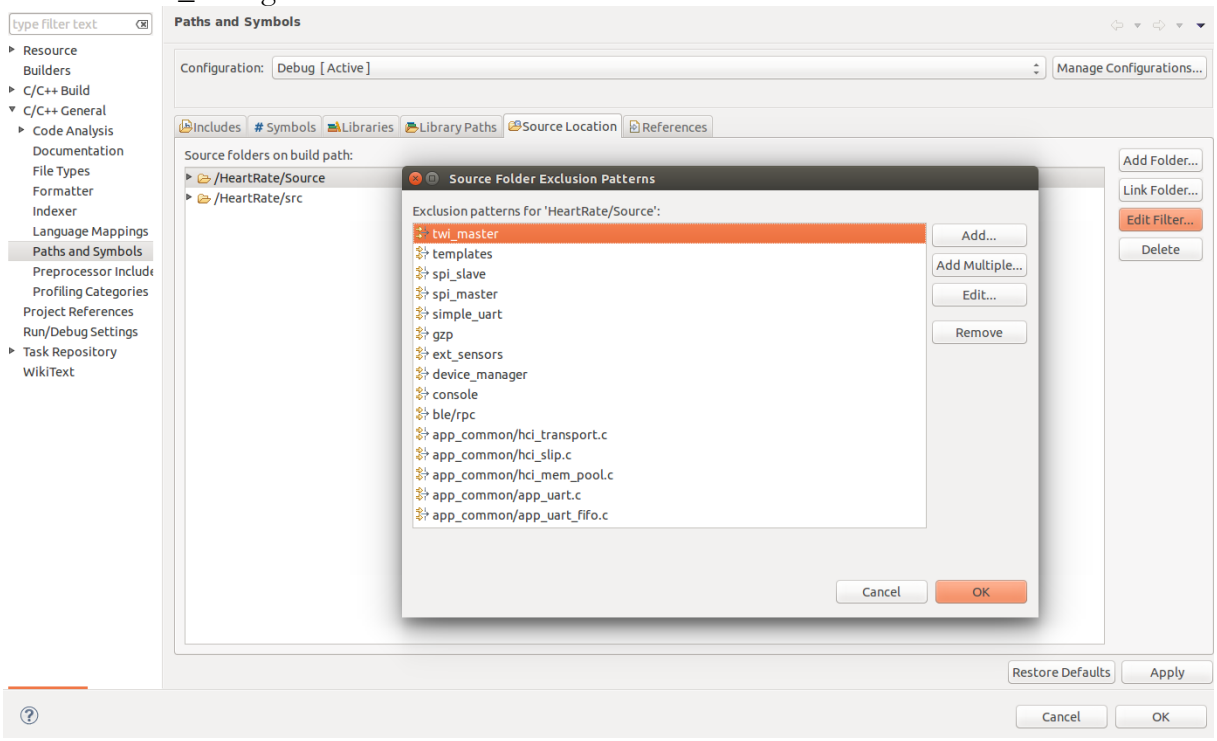


Figure 7: Fenêtre pour les Sources

10. dans l'onglet Symbols vérifiez bien que vous avez les symboles suivant :
    - **BLE_STACK_SUPPORT_REQD**(changez REQUIRED en REQD)
    - **BOARD_PCA10001**
    - **DEBUG**
    - **NRF51**
    - **NRF51822_QFAA_CA**
    - **S110_SUPPORT_REQUIRED**
11. Dans le menu de gauche, déroulez **C/C++ Build** et cliquez sur **Settings**
12. Dans l'onglet **Tool Settings** allez dans **Optimization** et entrez « --specs=nano.specs » dans **Other optimization** flags.Cela sert à optimiser le mapping de la mémoire.
13. Vous Devez modifier le **Heap_Size** de 2048 a 512 dans le fichier gcc_startup_nrf51.S.cela permet aussi une optimisation de la mémoire car on n'utilise pas la fonction malloc.

4) Réglages pour debugger un projet dans Eclipse

1. Dans **Project Explorer** sélectionnez le projet.
2. Ouvrez le menu **Run** et cliquez sur **Debug Configurations**.
3. Cliquez droit sur **GDB SEGGER J-LINK Debugging** et sélectionnez **New**.
4. Dans **C/C++ Application** cliquez sur **Browse**.
5. Sélectionnez le fichier **<nomduprojet>.elf** dans le dossier **Debug** du projet.

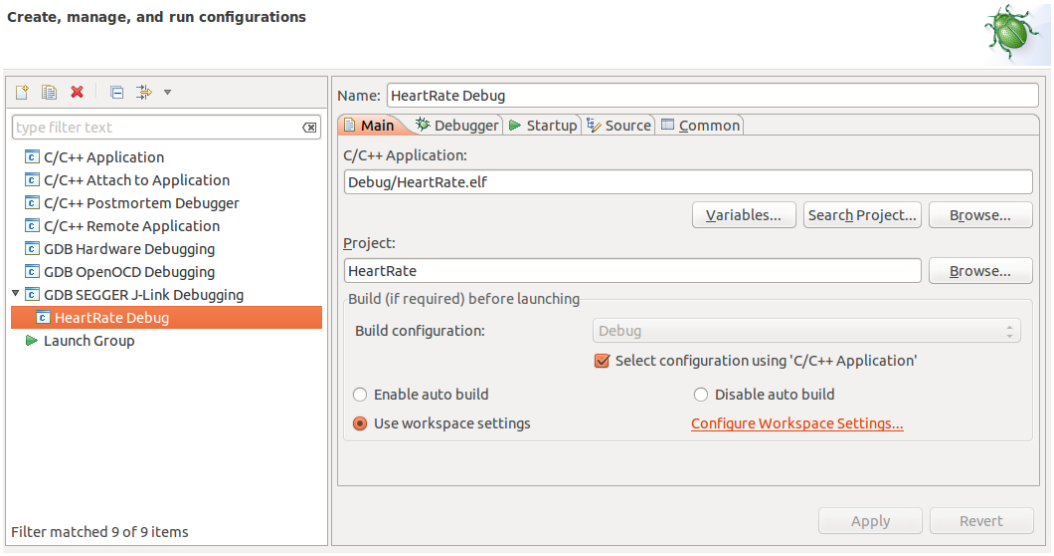


Figure 8: Fenêtre Main du debuguage

6. Sélectionnez l'onglet **Debugger** et changez le chemin de l'exécutable JlinkGDBServer en cliquant sur browse
7. Ajoutez le **device name** : nrf51822_xxaa

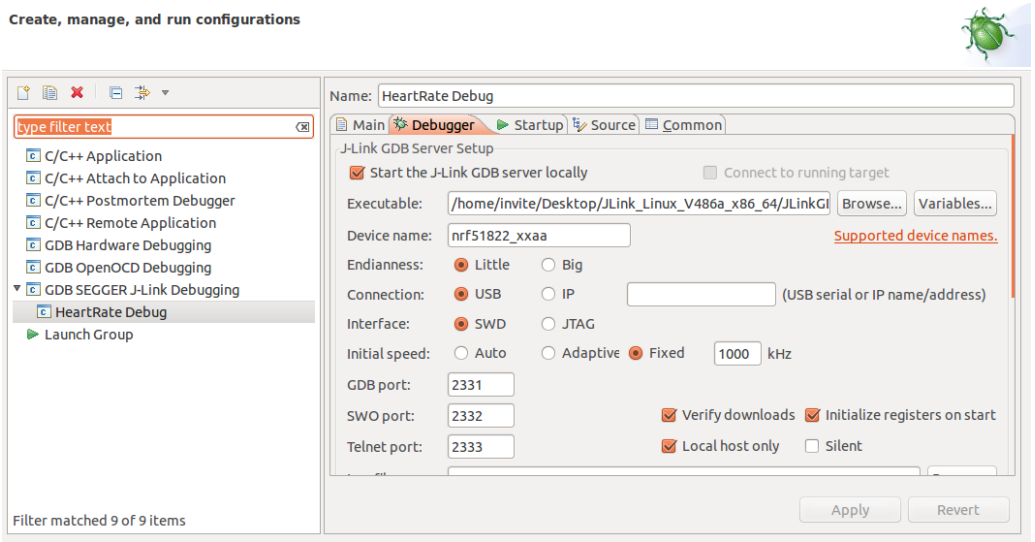


Figure 9 : Fenêtre Debugger du Debuguage

5) Téléchargement Flash

Le téléchargement Flash et le lancement d'un programme directement depuis Eclipse sont possible sur l'appareil sans debugger en utilisant le fichier **Flasher** fournis.

Vous devez d'abord mettre dans votre dossier du projet les fichiers suivant :

- Flasher et lui **donner les droits** (chmod 777 depuis le Terminal)
- s110_nrf51822_6.0.0_softdevice.hex

1. Dans le menu **Run**, cliquez **External Tools** puis **External Tools Configurations** et enfin **New**.
2. Dans le champ **Name** tapez **Flash <Nom du projet>** comme nom de configuration.
3. Dans le champ Location tapez le chemin d'accès entier vers Flasher (Par défaut : **${workspace_loc:/${project_name}/Flasher}**).
4. Dans le champ **Working Directory** sélectionnez le dossier de sortie pour **blinky (${workspace_loc:/${project_name}/Debug})**.
5. Dans le champ **Arguments** entrez :

   ${project_name}
   ${workspace_loc:/${project_name}}
   nrf51822_xxaa
   s110_nrf51822_6.0.0_softdevice.hex
   /home/invite/Documents/JLink_Linux_V484f_x86_64

   Les trois dernières lignes dépendent de votre device et de vos installations.

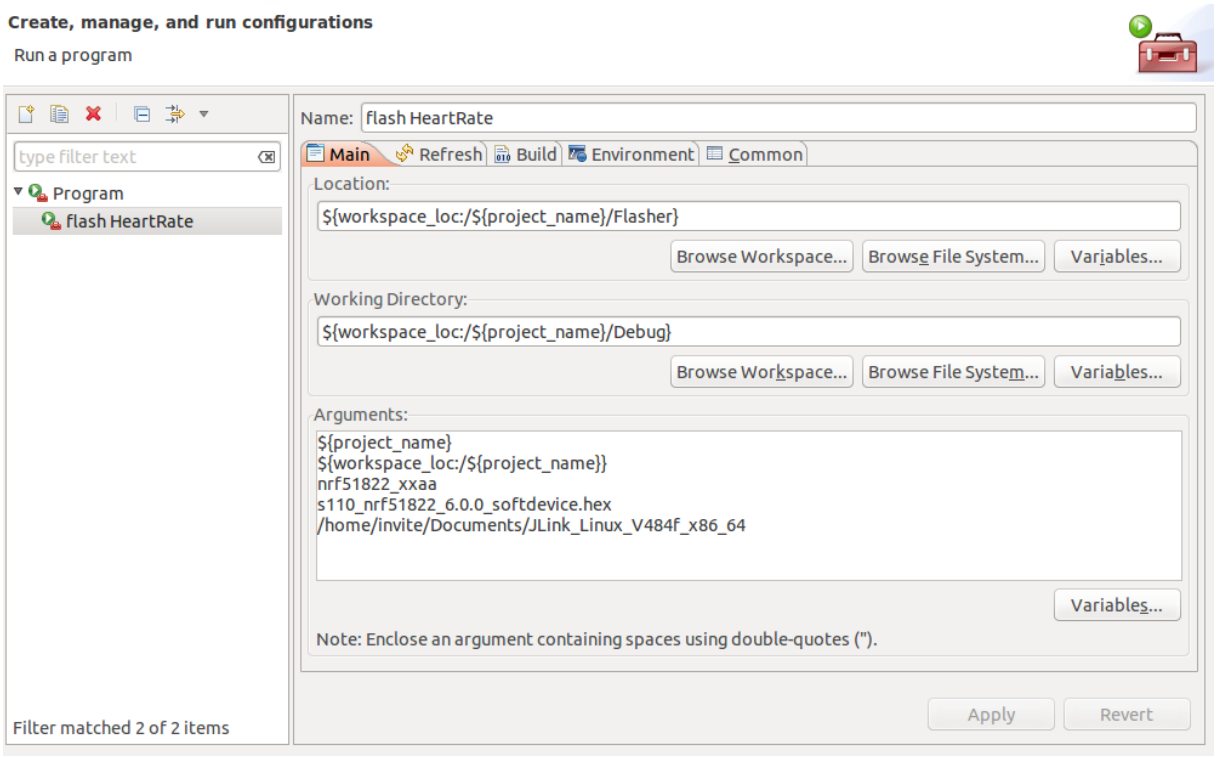


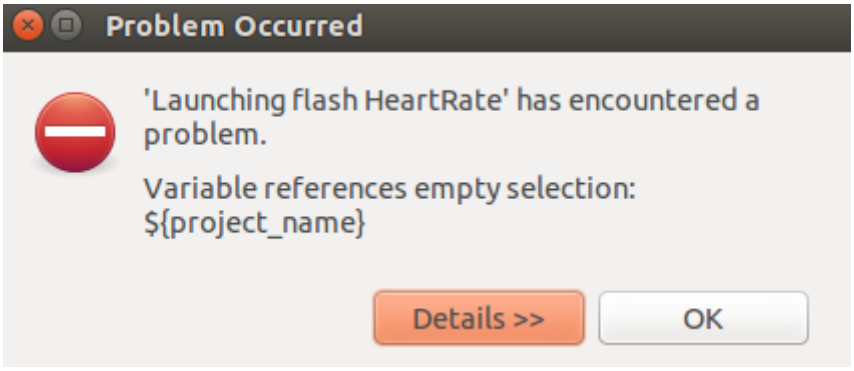
Figure 10: Fenêtre pour le Flash du programme

6) Erreur

1) Problème lors du Flash

Si lors du Flash vous rencontré le problème suivant : 'Lauching flash <nom du projet>' has encountered a problem.

Cliquez droit sur votre projet et puis sur Refresh



Références

Nordic Semi-Conductor, (2013). nRF51 Development with GCC and Eclipse.

Retrieved from https://www.nordicsemi.com/eng/nordic/download_resource/22748/7/20430912

Sourceforge. Nrf51osx- Eclipse plug-in for making Nordic nrf51822 projects.

Retrieved from http://sourceforge.net/projects/nrf51osx/