

Utilisation de Code Composer Studio v6

Auteur : Evangelina Lolivier-Exler

Dernière Mise à jour : 23.02.2017

Table des matières

Table des matières	1
Introduction.....	2
Versions des logiciels.....	2
Matériel nécessaire	2
Vue de la carte FPGA.....	3
Vue de la carte CPU	4
Connexion.....	5
Debug avec Code Composer Studio	5
Eclipse.....	5
Création de workspace.....	5
Création de projet	6
Compilation du projet	8
Ajout d'un fichier au projet	8
Connexion sur la carte cible	9
Test et debug d'un programme.....	11

Introduction

Ce document explique la marche à suivre pour l'utilisation de Code Composer Studio (CCSv6) pour déboguer des applications tournant sur le processeur Cortex-A8 de la carte REPTAR.

Code Composer Studio est un environnement de développement intégré (EDI ou IDE en anglais) pour le développement d'applications embarquées sur des processeurs de Texas Instruments. Ce logiciel est basé sur Eclipse.

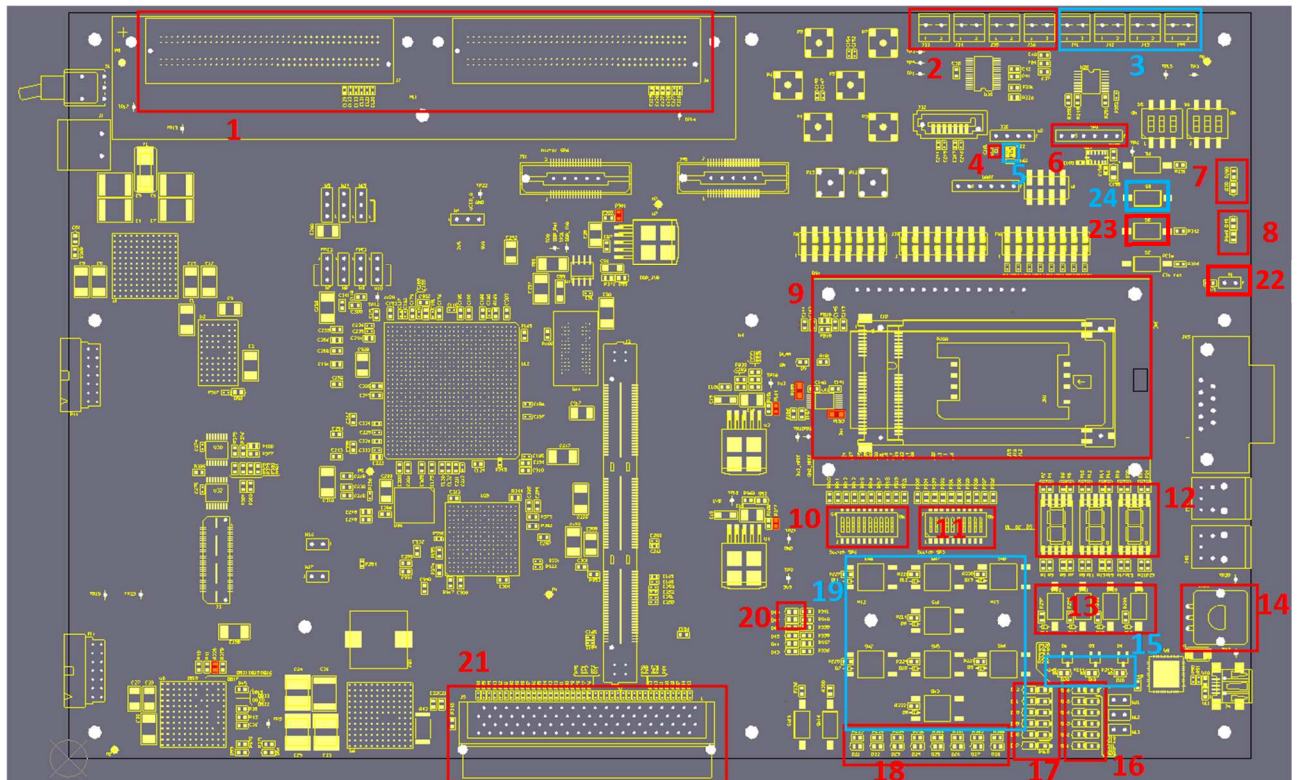
Versions des logiciels

- La version de Code Composer Studio utilisée est **la 6.0.1.00040**
 - Une mise à jour de CCSv6 pour supporter le microcontrôleur DM3730 a été téléchargée depuis le site http://processors.wiki.ti.com/index.php/Device_support_files
 - Les fichiers d'initialisation pour le DM3730 (GEL files) ont été téléchargés depuis le site http://processors.wiki.ti.com/index.php/OMAP_and_Sitara_CCS_support#DM3730_and_DM3725_2
 - NOTE : LE LOGICIEL AVEC LES MISES A JOUR SONT DISPONIBLES SUR LE DEPÔT GIT
- Tous les tests ont été réalisés sous Linux avec une distribution Ubuntu 12.04
- La toolchain utilisée est arm-none-eabi version 2011.03-42

Matériel nécessaire

- PC avec port USB 2.0
- 1x Câble mini-USB
- Emulateur JTAG sur USB : Blackhawk USB100v2-model D (équivalent TI XDS100v2)
- Carte REPTAR

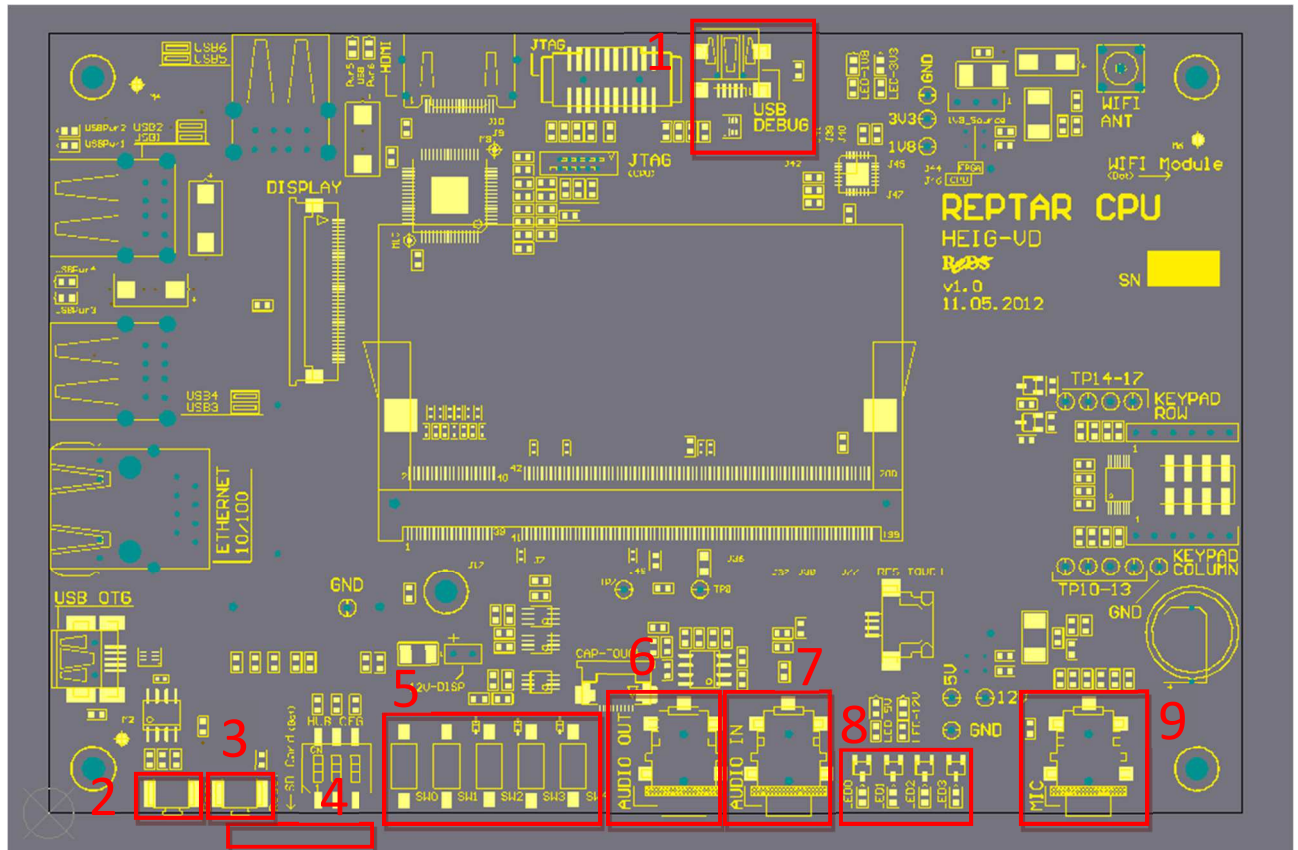
Vue de la carte FPGA



1. Connecteurs FMC
2. Connecteurs AD (Ch.0 à Ch.3)
3. Connecteurs DA (Ch.0 à Ch.3)
4. Capteur de température I2C
5. Capteur de lumière I2C
6. Connecteur externe SPI (W3)
7. Led SP6 Not Conf.
8. Led SP3 Not Conf.
9. Ecran mini-LCD 4x20
10. DIP Switch SP6
11. DIP Switch SP3
12. Afficheurs 7 segments
13. Switches uP
14. Encodeur
15. LEDS uP
16. LEDS SPARTAN 3 GREEN
17. LEDS SPARTAN 3 RED
18. LEDS SPARTAN 6
19. Boutons poussoir SPARTAN 6
20. LEDs Config Mode
21. REDS Connector (80 pôles)
22. Cavalier HSWAPEN
23. Bouton SP3 NProg
24. Bouton SP6 NProg

Figure 1. Vue de la carte FPGA

Vue de la carte CPU



1. Connecteur USB DEBUG (console série)
2. Bouton poussoir BOOT (SYS_BOOT5)
3. Bouton poussoir RESET
4. Emplacement carte SD (bottom)
5. Boutons poussoir sur GPIOs (SW0 à SW4)
6. Connecteur AUDIO OUT
7. Connecteur AUDIO IN
8. LEDS sur GPIOs (LED0 à LED3)
9. Connecteur audio MIC

Figure 2. Vue de la carte CPU

Connexion

1. Couper l'alimentation de la carte REPTAR
2. Connecter la sonde JTAG au PC à l'aide du câble mini-USB
3. Brancher l'autre bout de la sonde JTAG sur le connecteur J29 de la carte REPTAR CPU
4. Mettre la carte sous tension

Debug avec Code Composer Studio

Eclipse


L'environnement Eclipse-CDT permet l'édition et la cross-compilation de code C, le chargement dans la cible du code exécutable et le debug.

Création de workspace

Eclipse garde tous les paramètres de votre environnement de travail dans un répertoire appelé workspace. Ce répertoire doit contenir un sous-répertoire par projet.

- Depuis la console, placez-vous dans le dossier « /home/redsuser/etudiant »
- Clonez le dépôt Git pour ASP : **git clone <username>@eigit:/home2/reds/asp/asp_student**, mot de passe : correspond au <username>.
- Récupérez les fichiers avec la commande git pull. Dans le répertoire « asp_student /labos » récupéré vous trouverez un sous-répertoire par laboratoire
- Créez un nouveau dossier dans « /home/redsuser/etudiant » pour en faire votre workspace (ex : workspace_ccs) et placez dedans le répertoire du premier labo: labo_ASP_intro
- Lancez **Code Composer Studio**
- Dans le workspace launcher appuyez sur « Browse... » et cherchez votre répertoire workspace que vous venez de créer (ex : workspace_ccs)
- Appuyez sur « OK »

Création de projet

- Allez au menu File>New>Project...
- Sélectionnez « C/C++ » et « C Project », ensuite appuyez sur Next
- Introduisez le nom du projet: labo_ASP_intro. Le nom doit correspondre exactement avec le nom du répertoire existant
- Le chemin du répertoire apparaît automatiquement dans la case « Location »
- cliquez sur « Next », puis « Next » et « Finish ». Le projet apparaît à gauche de l'écran dans l'onglet « Project Explorer »
- Vérifiez que tous les fichiers du répertoire sont dans le projet en appuyant sur la petite flèche à gauche du nom du projet
- Dans l'onglet « Project Explorer », cliquez sur le nom du projet pour le sélectionner.
- Allez au menu Project>Properties
- Sur la liste qui se trouve à gauche sélectionnez « C/C++ Build ». A droite, sélectionnez la configuration « Debug », puis sous l'onglet « Builder Settings », décochez la case « Generate Makefiles automatically ».
- Dans la case « Build directory », vérifiez que le répertoire suivant est indiqué : \${workspace_loc:/labo_ASP_intro}. Si ce n'est pas le cas, cliquez sur « Workspace... », sélectionnez le nom du projet et cliquez sur OK.
- Cliquez sur OK pour enregistrer et quitter le menu.
- Changez de perspective (mode d'affichage) en cliquant sur  en haut et à droite de l'écran. Sélectionnez « C/C++ ».

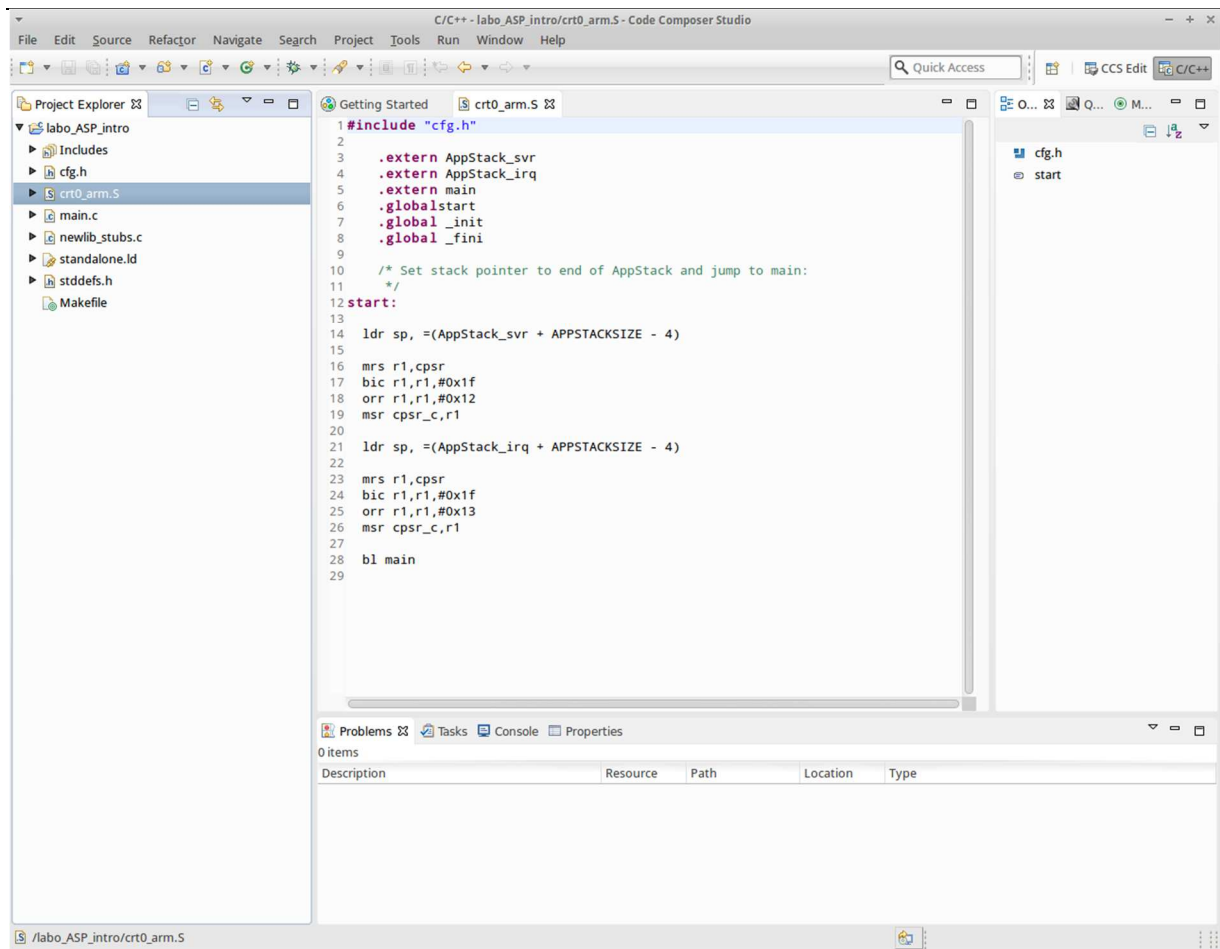


Figure 3. Perspective C/C++ de Code Composer Studio

Compilation du projet



- Dans l'onglet « Project Explorer », cliquez sur le nom du projet pour le sélectionner et ensuite allez au menu Project>Build Project
- Le résultat de la compilation s'affiche dans l'onglet Console en bas de l'écran
- Si vous avez des erreurs vous pouvez les voir sous l'onglet « Problems ». Cliquez sur les erreurs pour les localiser.
- Lorsque la compilation réussit, les fichiers résultants sont ajoutés au projet et vous pouvez les voir sous l'onglet Project Explorer en dessous des sources
- L'application exécutable a une extension **.elf**, telle que définie dans le Makefile.

Ajout d'un fichier au projet

- Pour créer un nouveau fichier source avec l'éditeur d'Eclipse, allez sur File>New>Source File. Le fichier est ajouté au projet et un nouvel onglet d'édition s'ouvre automatiquement. Si l'onglet d'édition ne s'ouvre pas automatiquement, double-cliquez sur le nom du fichier dans l'onglet Project Explorer
- Si vous disposez d'un fichier édité auparavant et vous voulez l'ajouter au projet, vous pouvez le copier dans le répertoire du projet depuis une console Linux ou l'explorateur de fichiers et il sera détecté par Eclipse. Si vous ne voyez pas le fichier s'ajouter automatiquement, cliquez sur le nom du projet ensuite tapez sur F5 pour rafraîchir l'affichage
- Afin que votre source C soit prise en compte à la compilation, vous devez créer un fichier d'en-tête .h (header), inclure le fichier dans votre fichier principal avec la directive #include, et ensuite modifier le Makefile (ajouter le nom du fichier .o dans la liste d'objets, variable OBJS).

Connexion sur la carte cible


Pour se connecter sur la carte à travers la sonde JTAG il faut d'abord créer une nouvelle configuration dans Code Composer Studio.

- Changez de perspective (mode d'affichage) en cliquant sur  en haut et à droite de l'écran. Sélectionnez « CCS Edit (default) »
- Allez au menu View>Target Configurations. Un nouvel onglet apparaît à droite de l'écran
- Cliquez sur User Defined avec le bouton droit de la souris et choisissez New Target Configuration
- Donnez un nom à votre configuration, par exemple « Reptar_conf_ccs »
- Décochez la case Use shared location, ensuite appuyez sur File System et allez chercher votre répertoire de workspace. Validez et appuyez sur Finish
- Dans la fenêtre qui s'ouvre automatiquement, dans la case Connection sélectionnez « Texas Instruments XDS100v2 USB Debug Probe »
- Dans la case Device sélectionnez DM3730
- Appuyez sur Save (dans le cadre Save Configuration, à droite)
- Assurez-vous que la carte est correctement branchée et allumée et ensuite appuyez sur Test Connection
- Si la connexion réussie vous obtenez le message « The JTAG DR Integrity scan-test has succeeded »
- Dans l'onglet Target Configurations à droite, cliquez sur la flèche à côté de User Defined et vérifiez que votre configuration a été ajoutée. Si elle n'est pas là vous pouvez l'ajouter en cliquant avec le bouton droit sur User Defined, ensuite cliquez sur Import Target Configuration et sélectionnez votre fichier « Reptar_conf_ccs.ccxml ». Sélectionnez aucune des 2 possibilités « copy files » et « Link to files », puis cliquez sur Ok.
- Pour vous connecter sur le processeur, cliquez sur le nom de la configuration avec le bouton droit et choisissez Launch Selected Configuration
- Cliquez sur la petite flèche de l'icône  et choisissez Debug configurations...

-
- Dans l'onglet Program, sélectionnez le Device « Texas Instruments XDS100v2 USB Debug Probe/Cortex_A8_0 ». Ensuite appuyez sur Workspace... et sélectionnez votre projet pour remplir la case Project
 - Afin de remplir la case Program cliquez sur Workspace... Allez dans le répertoire de votre projet (labo_ASP_intro), appuyez sur la flèche pour afficher les fichiers. Ensuite sélectionnez votre fichier d'extension **ELF** et validez.
 - Sous l'onglet Common, cochez Debug pour afficher votre configuration dans les favoris.
 - Cliquez sur Apply et ensuite sur Close
 - Votre Configuration est prête. Lorsque vous voulez lancer votre application cliquez sur l'icône



Test et debug d'un programme

- Lancez votre application en cliquant sur la flèche , puis votre configuration.
- L'application est chargée sur la carte et l'exécution s'arrête sur la première instruction du **main()**. Le code désassemblé apparaît dans la fenêtre **Disassembly** à droite de l'écran. En haut et à droite il y a les onglets **Registers**, **Breakpoints** et **Variables**. Si vous notez l'absence d'une des fenêtres allez au menu Window>Show View et sélectionnez la fenêtre qu'il manque.
- Au-dessus de l'onglet **Debug**, il y a une barre d'outils qui contient les boutons qui permettent d'avancer dans le programme ou de l'arrêter :



- Le bouton Step into permet d'avancer en pas à pas dans le code C. Le bouton Step over permet de ne pas rentrer dans le code des fonctions appelées. Le bouton Resume avance l'exécution jusqu'au prochain breakpoint
- A tout moment vous pouvez modifier votre programme et le recompiler sous la perspective C/C++.
- Pour ajouter un **breakpoint**, vous avez deux possibilités :
 - Double-cliquez sur la ligne de code dans la fenêtre **Disassembly**.
 - Dans le code C, cliquez avec le bouton droit de la souris sur la ligne de code et choisissez Breakpoint, puis **Hardware Breakpoint** (Attention : Le Breakpoint obtenue avec le double clic sur une ligne de code C ne fonctionne pas).
- Dans l'onglet **Registers** vous trouverez tous les registres internes du CPU (Core Registers : PC, SP, LR, R0..R14, etc.), ainsi que les registres de configuration de tous les modules du microcontrôleur DM3730
- Pour afficher les variables globales, allez dans l'onglet **Variables**, puis cliquez avec le bouton droit pour ajouter les variables. L'autre possibilité est d'ajouter le nom de la variable dans l'onglet **Expressions**. ATTENTION, l'affichage de tableaux ralentit énormément le système, pour voir le contenu d'un tableau il est conseillé d'insérer son adresse dans le Memory Browser
- Les variables locales s'affichent automatiquement dans l'onglet **Variables**, pas besoin de les ajouter
- Pour afficher un espace mémoire allez dans l'onglet **Memory Browser**. Rentrez l'adresse en hexadécimal en commençant par « 0x ». Cliquez sur Go

-
- Pour afficher le contenu d'un registre dans le **Memory Browser**, cliquez avec le bouton droit sur le nom du registre dans l'onglet Registers et choisissez View memory at value...
 - Pour terminer votre application, cliquez avec le bouton droit de la souris sur la ligne « main » dans l'onglet **Debug** et choisissez Terminate and Remove. Ceci déconnecte la carte cible.