ECE 540 Getting Started with Software Development

MIPSfpga Assembly Language Programing

Based on material provided by Bruce Abelidinger

Modified for ECE 540 by:
Yiwei Li, Srivatsa Yogendra, Roy Kravitz,
Melih Erdogan, and Dan Hammerstrom

Note: Some of the window formats and Button positions have changed, but you should be able to make the appropriate translation.

MIPSfpga
by Imagination

MIPSfpga Programing using Codescape® for Eclipse®

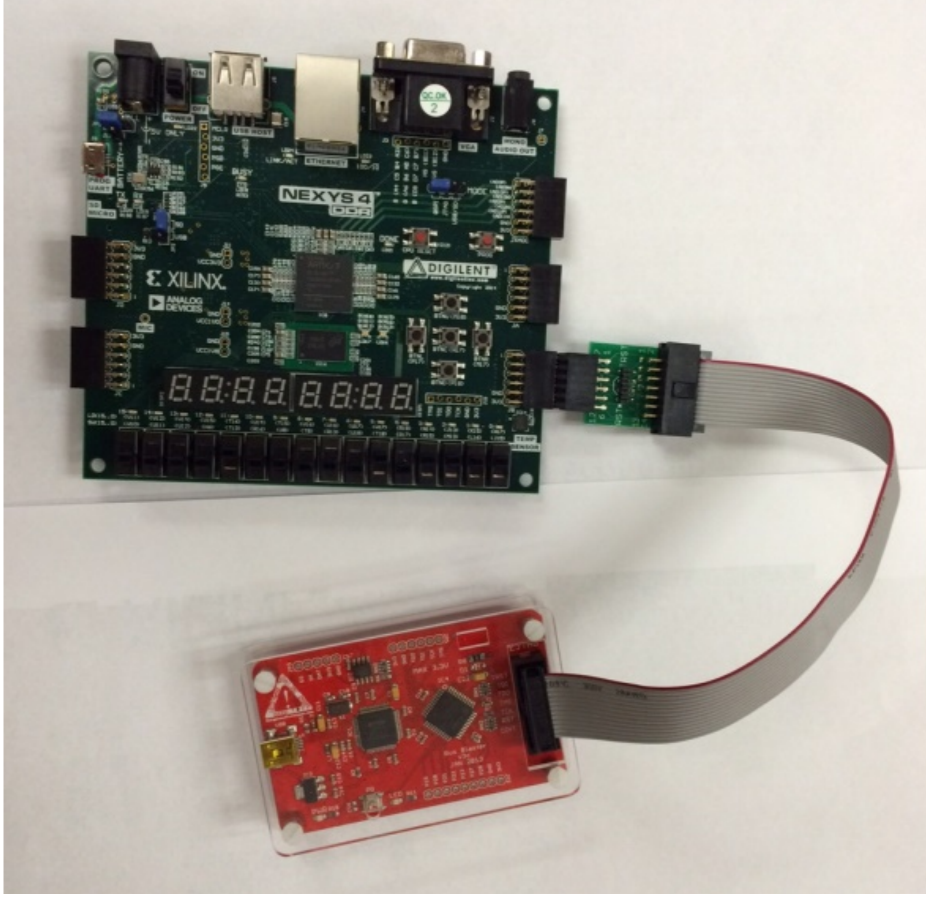
Introduction

In this part of the Getting Started project you will learn to execute and debug programs for the MIPSfpga processor in Assembly language using Codescape for Eclipse (C4E). You will complete a tutorial on compiling, downloading and running a very simple assembly language program on the Nexys4 DDR board and learn how to use C4E to modify and examine memory and registers, set breakpoints, and single-step the program.

Using C4E to Execute a Program

Step 1: Connect the Bus Blaster Probe

The best approach is to plug the BUSBLASTER into the Nexys4 DDR board before connecting (and consequently powering up) either to your PC's USB ports. Use the small extension board as shown in the figure. Plug the Nexys4 DDR USB into the PC after you have Vivado up and running. You do not need to plug the BUSBLASTER USB yet. Continue to Step 2.

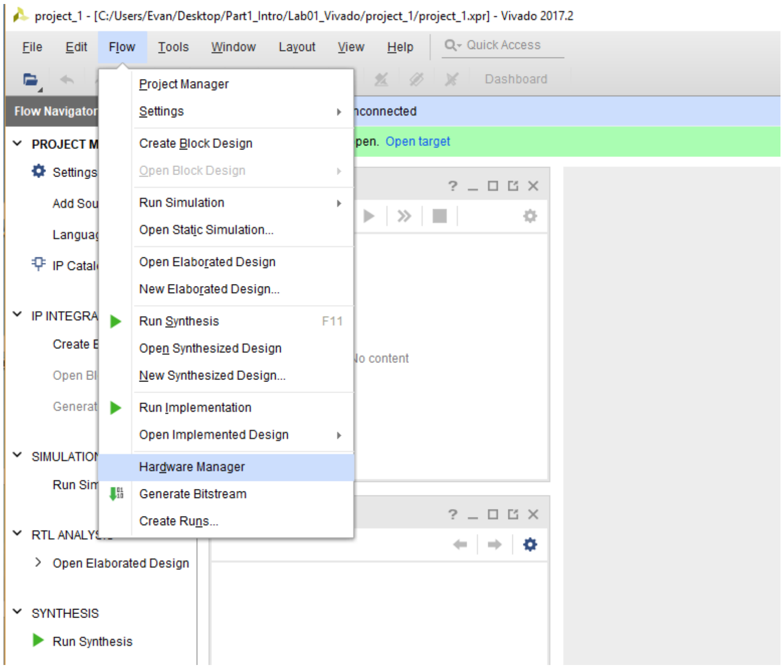


Step 2: Download the MIPSfpga system to the Nexys4 DDR board

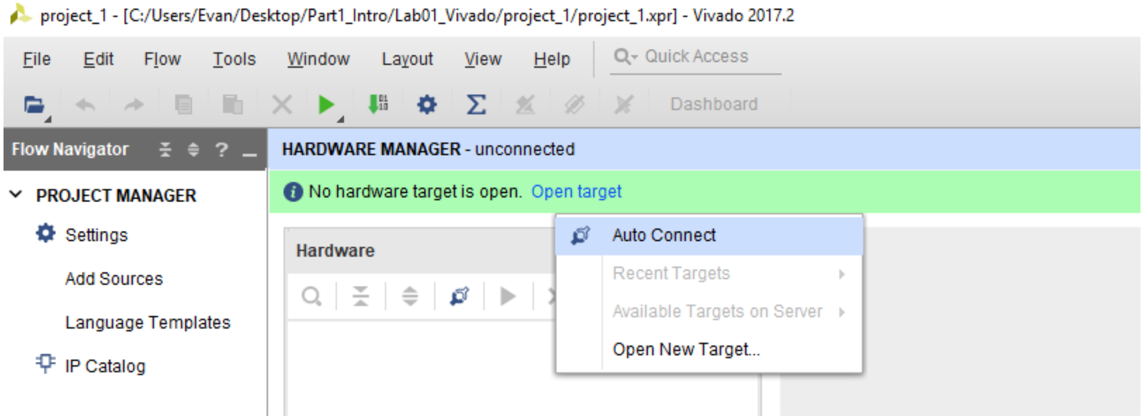
You should configure the FPGA with target system before you start C4E. Connect the board to your computer, turn the board on, and open the Vivado project you created earlier. After the project opens choose

Flow → Open Hardware Manager

from Vivado's top menu (these windows are more or less the same in Vivado 2017.3):



Next click on **Open Target → Auto Connect**


project_1 - [C:/Users/Evan/Desktop/Part1_Intro/Lab01_Vivado/project_1/project_1.xpr] - Vivado 2017.2
File
Edit
Flow
Tools
Window
Layout
View
Help
Quick Access
Dashboard
Flow Navigator
PROJECT MANAGER
Settings
Add Sources
Language Templates
IP Catalog
HARDWARE MANAGER - unconnected
No hardware target is open. Open target
Hardware
Auto Connect
Recent Targets
Available Targets on Server
Open New Target...

Now click on **Program device**

In the Program Device window, select the bitfile (.bit) you created earlier. It may already be selected. Then click **Program**. Leave the Debug probes file blank.

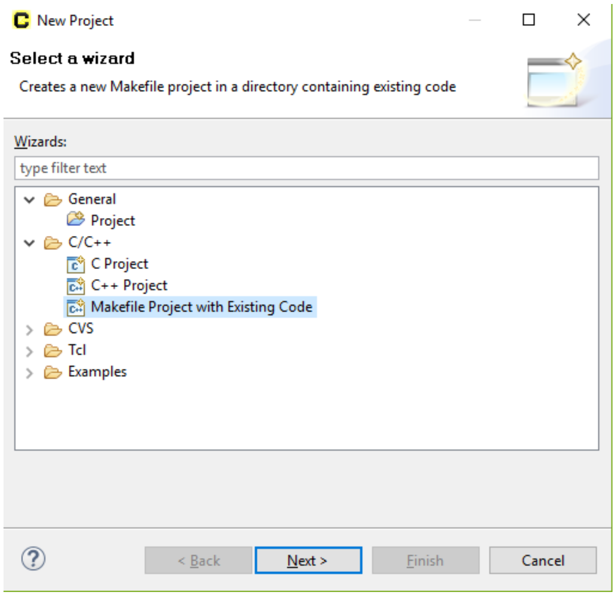
Press the red CPU Reset button on the Nexys4 DDR board to reset the MIPSfpga core and begin running the pre-loaded program that displays incremented values on the LEDs.

Step 3: Create and Load an Assembly Language Project

Start Codescape for Eclipse (C4E) then follow these steps to create a project. Set "<Your ProjGS Directory>/software" as the default directory.

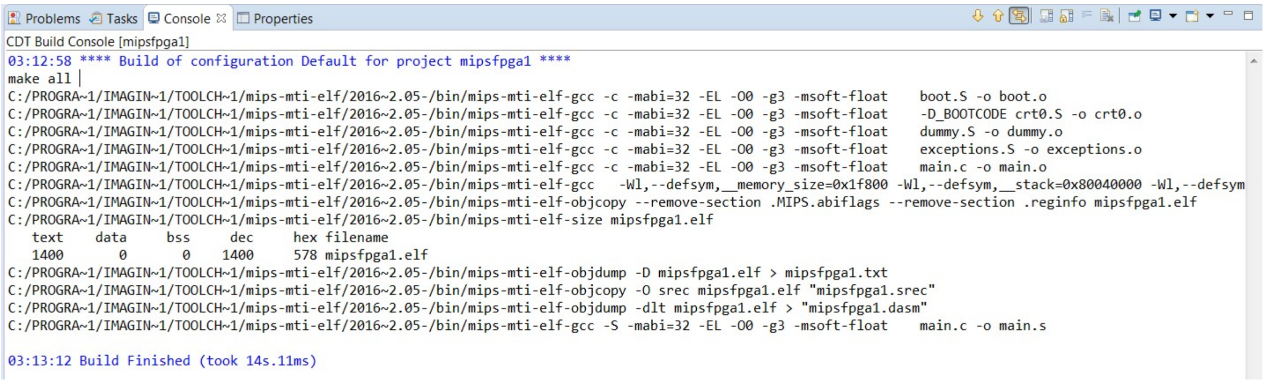
File → New → Project

Choose "Makefile Project with Existing Code" in the C/C++ folder, and click Next.



1. Click on Browse and go to the directory “<Your ProjGS Directory>/software/ReadSwitches”.
2. Select the folder and uncheck boxes for C and C++ and click finish.
3. After project created, expand ReadSwitches in the navigator and open main.s to view the assembly code of ReadSwitches.
4. Press Ctrl + B or select Project -> Build All from the menu bar to build the project.

Finish builds the program by running the Makefile. The Console output will look something like this:



```
03:12:58 **** Build of configuration Default for project mipsfpga1 ****
make all 
C:/PROGRA~1/IMAGIN~1/TOOLCH~1/mips-mti-elf/2016~2.05-/bin/mips-mti-elf-gcc -c -mabi=32 -EL -O0 -g3 -msoft-float    boot.S -o boot.o
C:/PROGRA~1/IMAGIN~1/TOOLCH~1/mips-mti-elf/2016~2.05-/bin/mips-mti-elf-gcc -c -mabi=32 -EL -O0 -g3 -msoft-float    -D_BOOTCODE crt0.S -o crt0.o
C:/PROGRA~1/IMAGIN~1/TOOLCH~1/mips-mti-elf/2016~2.05-/bin/mips-mti-elf-gcc -c -mabi=32 -EL -O0 -g3 -msoft-float    dummy.S -o dummy.o
C:/PROGRA~1/IMAGIN~1/TOOLCH~1/mips-mti-elf/2016~2.05-/bin/mips-mti-elf-gcc -c -mabi=32 -EL -O0 -g3 -msoft-float    exceptions.S -o exceptions.o
C:/PROGRA~1/IMAGIN~1/TOOLCH~1/mips-mti-elf/2016~2.05-/bin/mips-mti-elf-gcc -c -mabi=32 -EL -O0 -g3 -msoft-float    main.c -o main.o
C:/PROGRA~1/IMAGIN~1/TOOLCH~1/mips-mti-elf/2016~2.05-/bin/mips-mti-elf-gcc   -Wl,--defsym,__memory_size=0x1f800 -Wl,--defsym,__stack=0x80040000 -Wl,--defsym
C:/PROGRA~1/IMAGIN~1/TOOLCH~1/mips-mti-elf/2016~2.05-/bin/mips-mti-elf-objcopy --remove-section .MIPS.abiflags --remove-section .reginfo mipsfpga1.elf
C:/PROGRA~1/IMAGIN~1/TOOLCH~1/mips-mti-elf/2016~2.05-/bin/mips-mti-elf-size mipsfpga1.elf
   text    data     bss     dec     hex filename
   1400       0       0    1400     578 mipsfpga1.elf
C:/PROGRA~1/IMAGIN~1/TOOLCH~1/mips-mti-elf/2016~2.05-/bin/mips-mti-elf-objdump -D mipsfpga1.elf > mipsfpga1.txt
C:/PROGRA~1/IMAGIN~1/TOOLCH~1/mips-mti-elf/2016~2.05-/bin/mips-mti-elf-objcopy -O srec mipsfpga1.elf "mipsfpga1.srec"
C:/PROGRA~1/IMAGIN~1/TOOLCH~1/mips-mti-elf/2016~2.05-/bin/mips-mti-elf-objdump -dlt mipsfpga1.elf > "mipsfpga1.dasm"
C:/PROGRA~1/IMAGIN~1/TOOLCH~1/mips-mti-elf/2016~2.05-/bin/mips-mti-elf-gcc -S -mabi=32 -EL -O0 -g3 -msoft-float    main.c -o main.s

03:13:12 Build Finished (took 14s.11ms)
```

Useful Factoid: Changing Build Settings

You can change the build settings for a specific project by editing the settings in the Makefile. For example, if you wanted to change the level of compiler optimization, the line to change would be:

```
CFLAGS = -EL -g -march=m14kc -msoft-float -O1 -I$(LIBFDCPATH)
```

Changing “-O1” to –O0, for example is a useful thing to do if you are single-stepping through your program with the debugger. The MIPS assembler restructures you code to fill the branch and load delay slots resulting from the MIPS pipeline. Setting the Optimization level to 0 (O0) disables instruction reordering.

Step 4: Program and Debug the Nexys4 DDR board through the Bus Blaster

Now plug the BUSBLASTER USB cable into another USB port on your PC.

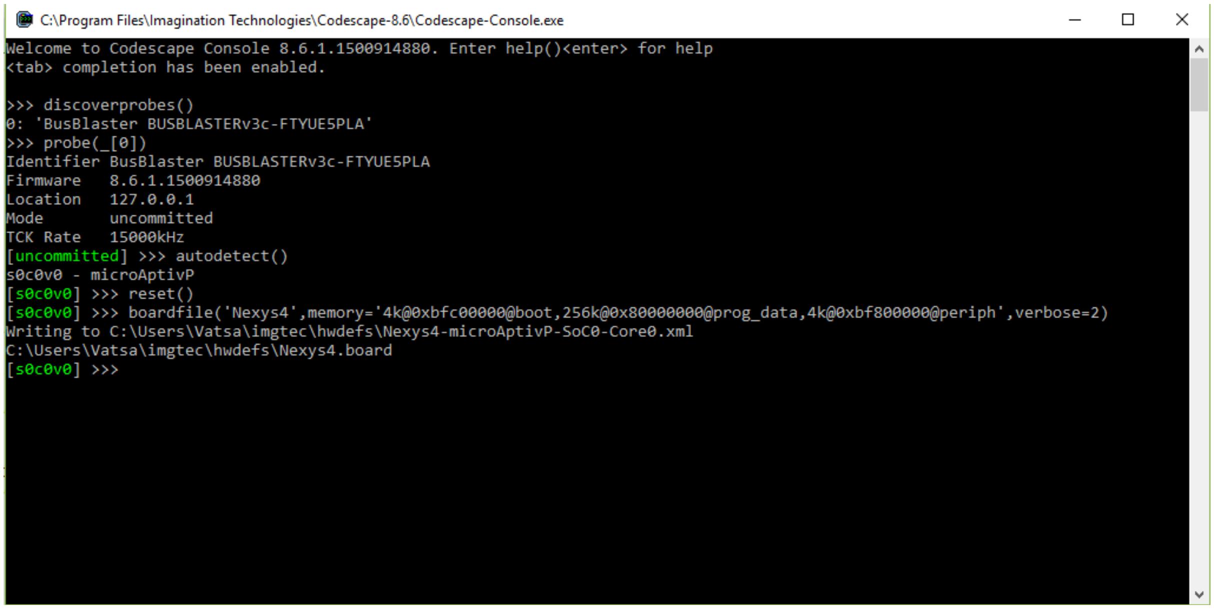
Note: You can go to the “Codescape-8.6” and “Codescape-For-Eclipse-8.6” directories (in “C:\Program Files\Imagination Technologies”) for the executables, you can then create shortcuts on your Desktop.

Create a Boardfile (you only need to do this the first time)

Start the Codescape-Console and enter the following commands:

- discoverprobes()
- probe(_[0])
- autodetect()
- reset()
- boardfile(‘Nexys4’,memory=’4K@0xbfc00000@boot,256K@0x80000000@prog_data, 4K@0xbf800000@periph’,verbose=2)

Note: You may not be able to copy and paste the single quote to the command prompt; please type the boardfile command (carefully) into the prompt.



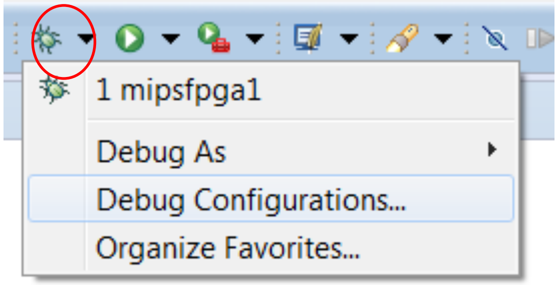
C:\Program Files\Imagination Technologies\Codescape-8.6\Codescape-Console.exe

```
Welcome to Codescape Console 8.6.1.1500914880. Enter help()<enter> for help
<tab> completion has been enabled.

>>> discoverprobes()
0: 'BusBlaster BUSBLASTERv3c-FTYUE5PLA'
>>> probe(_[0])
Identifier BusBlaster BUSBLASTERv3c-FTYUE5PLA
Firmware   8.6.1.1500914880
Location   127.0.0.1
Mode       uncommitted
TCK Rate   15000kHz
[uncommitted] >>> autodetect()
s0c0v0 - microAptivP
[s0c0v0] >>> reset()
[s0c0v0] >>> boardfile('Nexys4',memory='4k@0xbfc00000@boot,256k@0x80000000@prog_data,4k@0xbf800000@periph',verbose=2)
Writing to C:\Users\Vatsa\imgtec\hwdefs\Nexys4-microAptivP-SoC0-Core0.xml
C:\Users\Vatsa\imgtec\hwdefs\Nexys4.board
[s0c0v0] >>>
```

Set Up a Debug Configuration

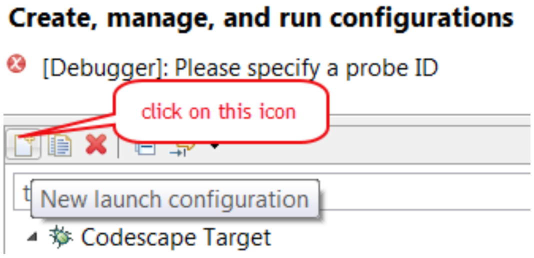
Moving back to C4E, the next step is to create a Debug Configuration. Make sure you have the project name selected under Project Explorer first.

Click on the down-arrow next to the "bug" icon. It will look something like this:



Click on Debug Configurations…

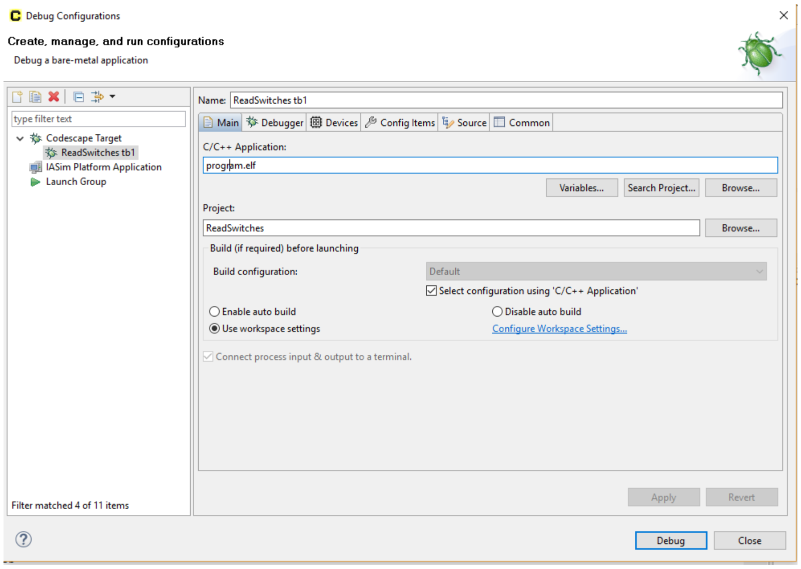
The first time you are building a debug setup for a project, you first click on "New launch configuration" icon.


Create, manage, and run configurations

[Debugger]: Please specify a probe ID

You can edit the name of the debug configuration, or use the default.

In C/C++ Application: type “program.elf”



Debug Configurations

Create, manage, and run configurations

Debug a bare-metal application

type filter text

- Codescape Target
  - ReadSwitches tb1
- IASim Platform Application
- Launch Group

Filter matched 4 of 11 items

Name: ReadSwitches tb1

C/C++ Application: program.elf

Project: ReadSwitches

Build (if required) before launching

Build configuration: Default

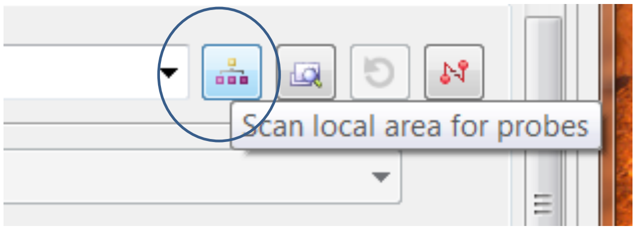
☑ Select configuration using 'C/C++ Application'

○ Enable auto build
○ Disable auto build
● Use workspace settings

☑ Connect process input & output to a terminal.

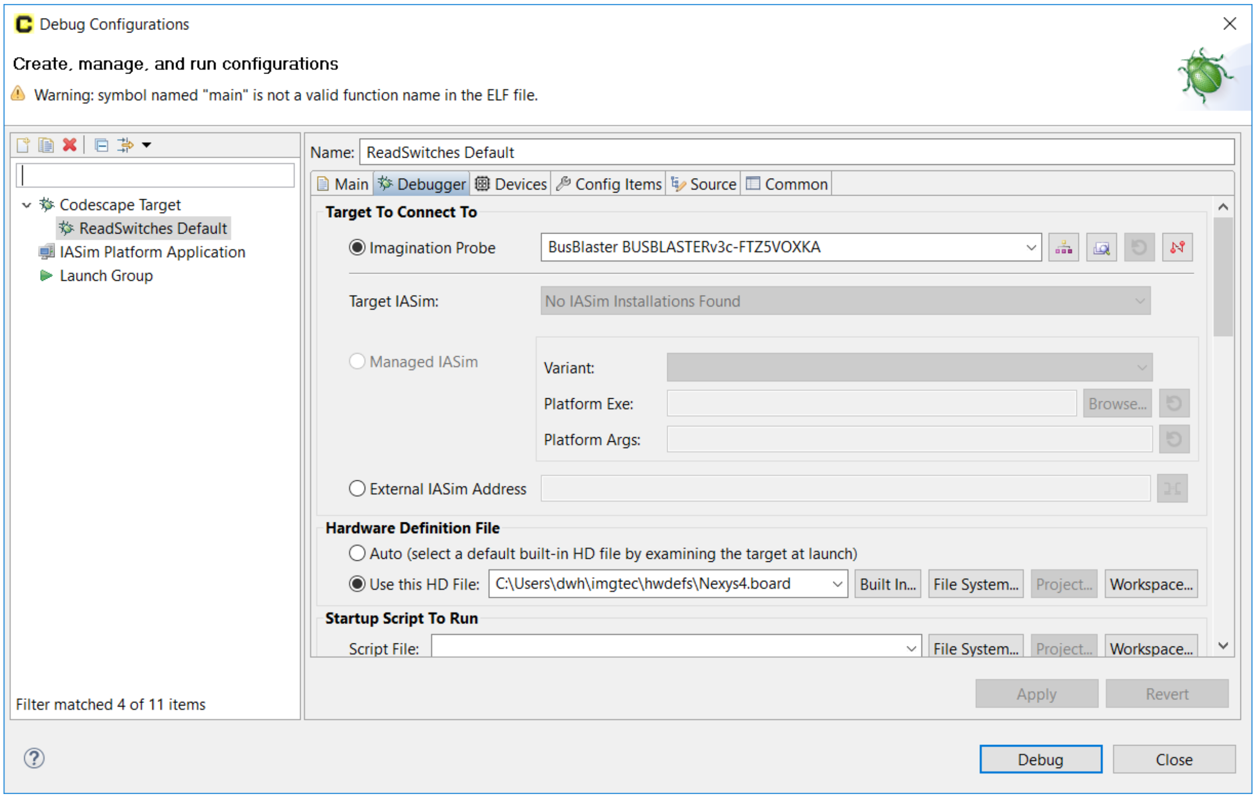
Next click on the Debugger tab

Click on the “network” icon to find the attached Bus Blaster. Make sure the Bus Blaster is plugged into your PC and the probe end is plugged into the Nexys4 DDR JB PMOD connector on the bottom right of the board.



At this stage C4E will check for a USB Bus Blaster probe plugged into your Windows PC.

When the probe is found, it will show up with a name similar to that in the window.



Debug Configurations

Create, manage, and run configurations

⚠ Warning: symbol named "main" is not a valid function name in the ELF file.

Name: ReadSwitches Default

Target To Connect To

● Imagination Probe: BusBlaster BUSBLASTERv3c-FTZ5VOXKA

Target IASim: No IASim Installations Found

○ Managed IASim — Variant: / Platform Exe: / Platform Args:

○ External IASim Address

Hardware Definition File

○ Auto (select a default built-in HD file by examining the target at launch)

● Use this HD File: C:\Users\dwh\imgtec\hwdefs\Nexys4.board

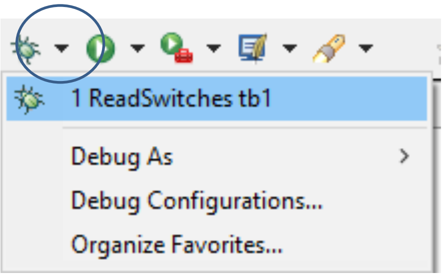
Startup Script To Run

Script File:

Filter matched 4 of 11 items

Go to the Hardware Definition File section, select “Use this HD File”, then use the File System browser to select Nexys4.board. Note the path used in the screenshot. Click Apply.

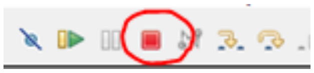
To start the debug session, you can either click on **Debug** in the Debug Configuration dialog or click the Debug button in the C4E toolbar:



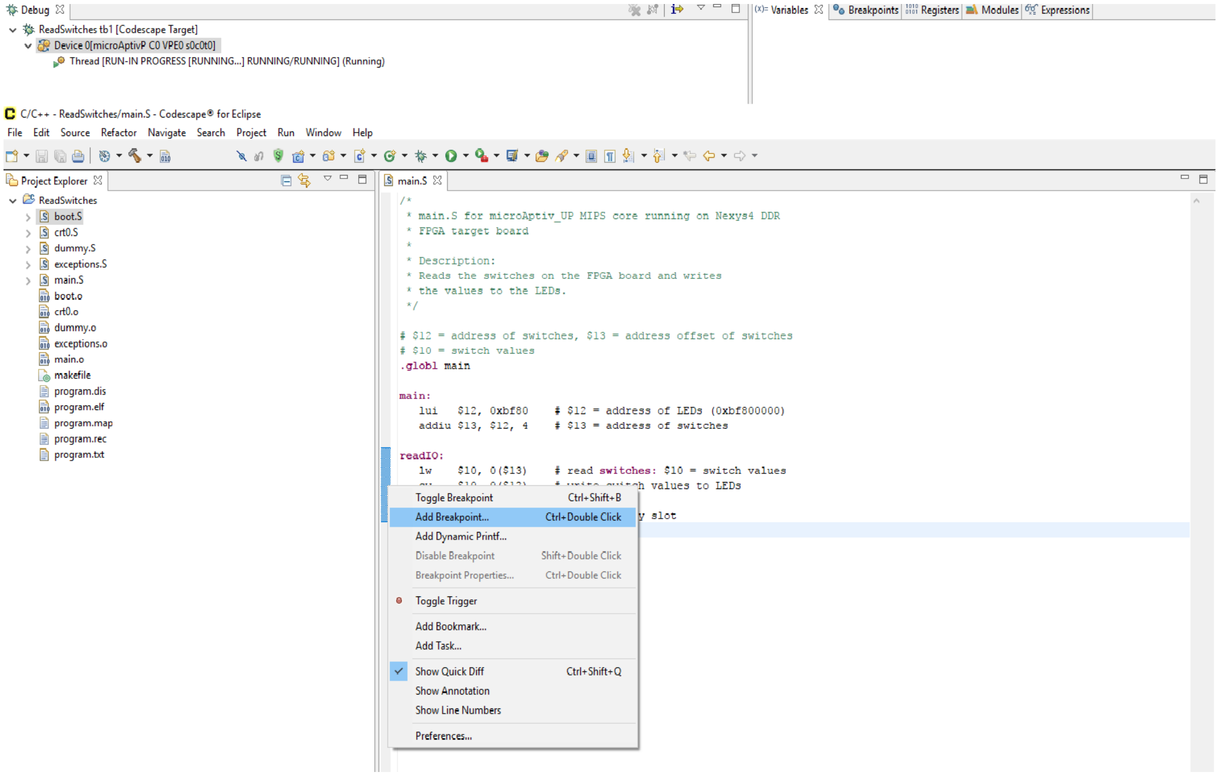
.

Now the ReadSwitches program should be programed to the board. You can turn on the LED by turn on the corresponding switch on the board.

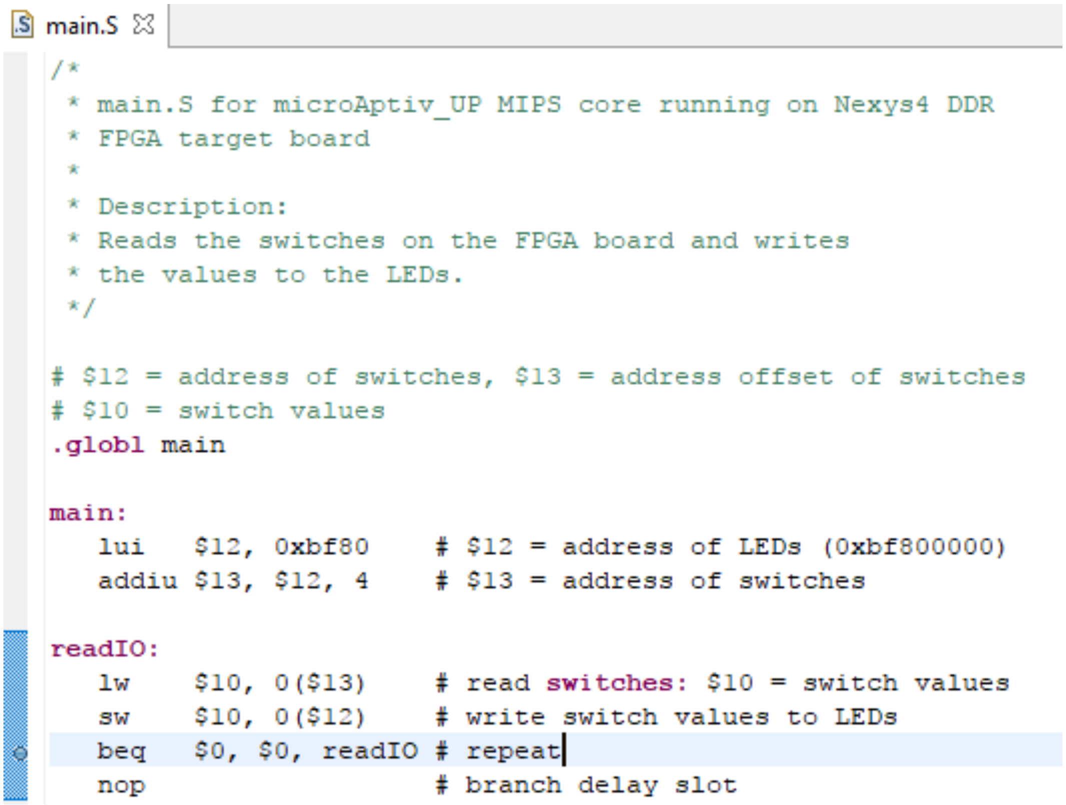
C4E Tips – Setting Breakpoints

1. Press the red button to stop the debugging session 

2. Add a breakpoint by right clicking on the line you want to add the breakpoint to and selecting **Add Breakpoint**

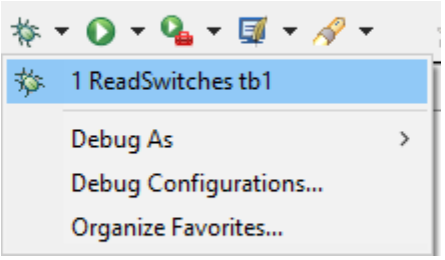

Debug
ReadSwitches tb1 [Codescape Target]
Device 0[microAptivP C0 VPE0 s0c0t0]
Thread [RUN-IN PROGRESS [RUNNING...] RUNNING/RUNNING] (Running)
Variables
Breakpoints
Registers
Modules
Expressions
C/C++ - ReadSwitches/main.S - Codescape® for Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
ReadSwitches
boot.S
crt0.S
dummy.S
exceptions.S
main.S
boot.o
crt0.o
dummy.o
exceptions.o
main.o
makefile
program.dis
program.elf
program.map
program.rec
program.txt
main.S
Toggle Breakpoint Ctrl+Shift+B
Add Breakpoint... Ctrl+Double Click
Add Dynamic Printf...
Disable Breakpoint Shift+Double Click
Breakpoint Properties... Ctrl+Double Click
Toggle Trigger
Add Bookmark...
Add Task...
Show Quick Diff Ctrl+Shift+Q
Show Annotation
Show Line Numbers
Preferences...

```
/*
 * main.S for microAptiv_UP MIPS core running on Nexys4 DDR
 * FPGA target board
 *
 * Description:
 * Reads the switches on the FPGA board and writes
 * the values to the LEDs.
 */

# $12 = address of switches, $13 = address offset of switches
# $10 = switch values
.globl main

main:
   lui   $12, 0xbf80    # $12 = address of LEDs (0xbf800000)
   addiu $13, $12, 4    # $13 = address of switches

readIO:
   lw    $10, 0($13)    # read switches: $10 = switch values
                                     h values to LEDs
                                     y slot
```

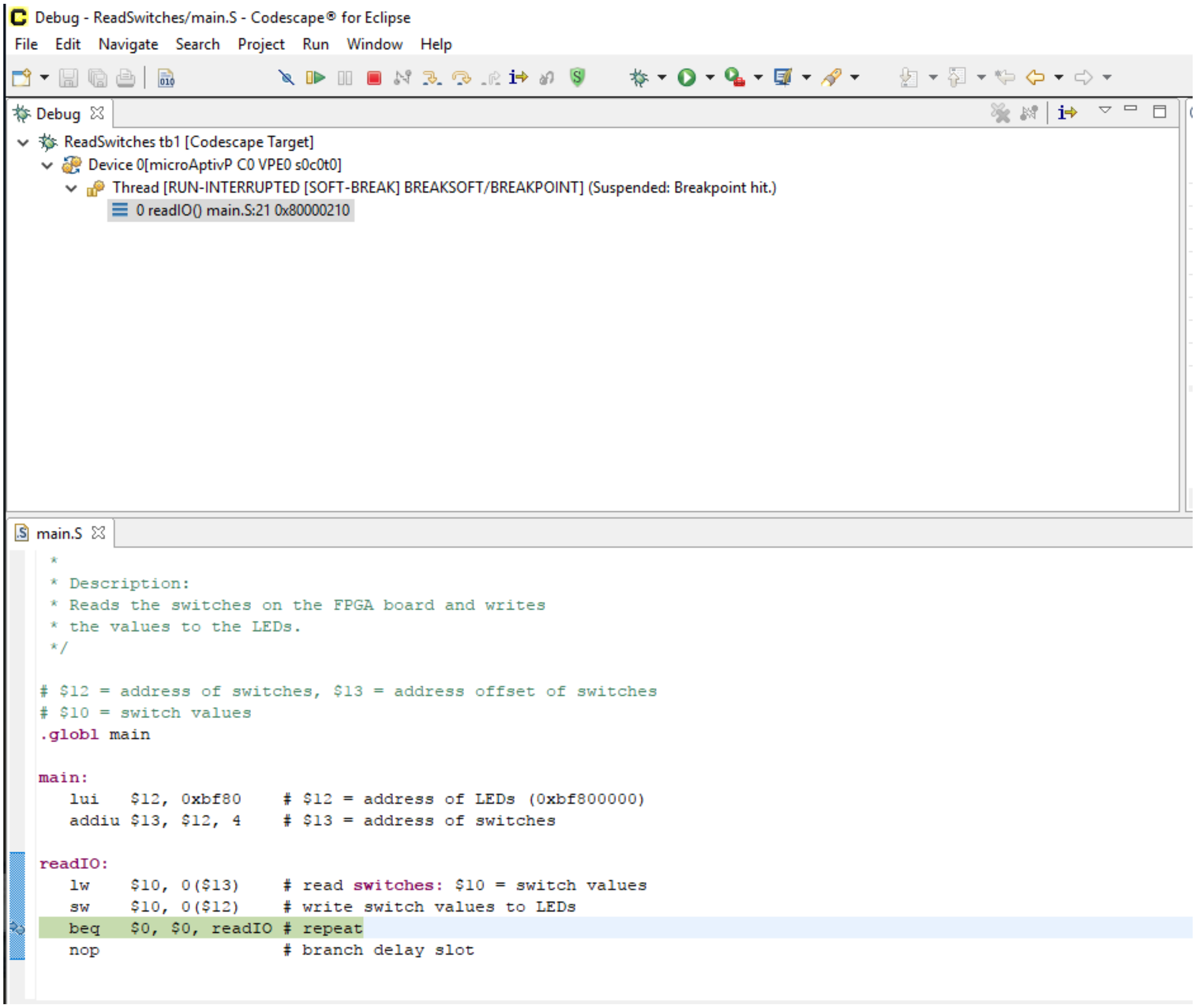


```
/*
 * main.S for microAptiv_UP MIPS core running on Nexys4 DDR
 * FPGA target board
 *
 * Description:
 * Reads the switches on the FPGA board and writes
 * the values to the LEDs.
 */

# $12 = address of switches, $13 = address offset of switches
# $10 = switch values
.globl main

main:
   lui   $12, 0xbf80    # $12 = address of LEDs (0xbf800000)
   addiu $13, $12, 4    # $13 = address of switches

readIO:
   lw    $10, 0($13)    # read switches: $10 = switch values
   sw    $10, 0($12)    # write switch values to LEDs
   beq   $0, $0, readIO # repeat
   nop                  # branch delay slot
```

3. Run the debugger again.



4. The program should stop at the break point.



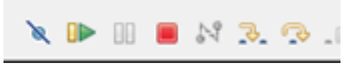
Debug - ReadSwitches/main.S - Codescape® for Eclipse
File Edit Navigate Search Project Run Window Help
Debug
main.S

ReadSwitches tb1 [Codescape Target]
Device 0[microAptivP C0 VPE0 s0c0t0]
Thread [RUN-INTERRUPTED [SOFT-BREAK] BREAKSOFT/BREAKPOINT] (Suspended: Breakpoint hit.)
0 readIO() main.S:21 0x80000210

```
 *
 * Description:
 * Reads the switches on the FPGA board and writes
 * the values to the LEDs.
 */

# $12 = address of switches, $13 = address offset of switches
# $10 = switch values
.globl main

main:
   lui   $12, 0xbf80     # $12 = address of LEDs (0xbf800000)
   addiu $13, $12, 4     # $13 = address of switches

readIO:
   lw    $10, 0($13)     # read switches: $10 = switch values
   sw    $10, 0($12)     # write switch values to LEDs
   beq   $0, $0, readIO  # repeat
   nop                   # branch delay slot
```

5. Turn on one switch on your board, then hit until you see the corresponding LED light up.

6. Now you can add other break points, and try out different option



such as step into, resume etc.