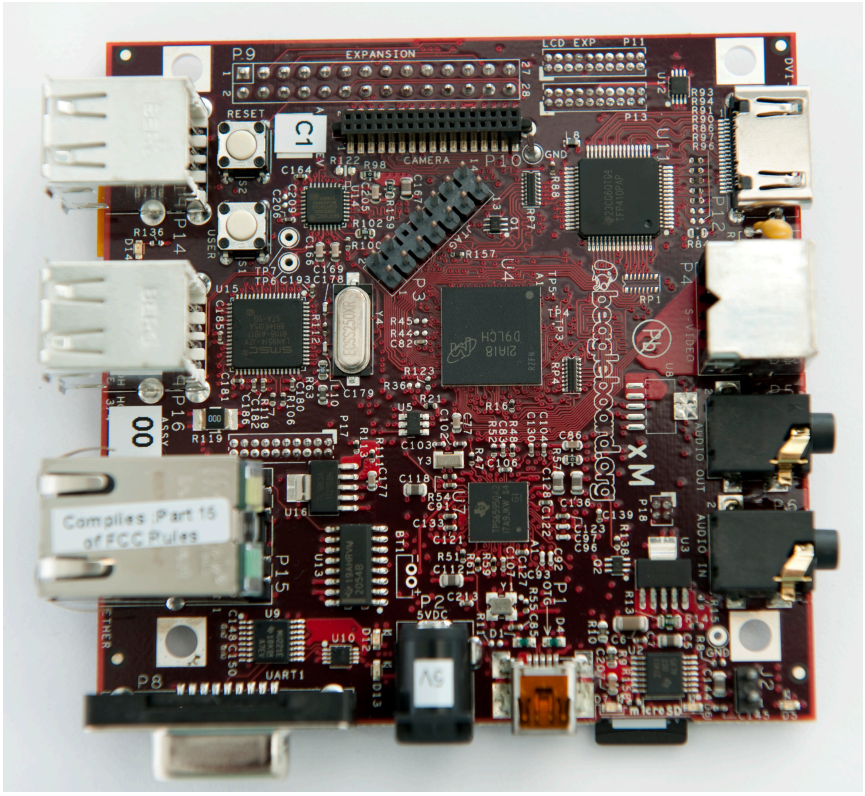
Georgia Institute of Technology

1 Introduction

In this lab you will familiarize yourself with the BeagleBoard-xM microcontroller and associated tools as well the lab practices expected of this course.

1.1 BeagleBoard-xM

BeagleBoard-xM is an open source board design for prototyping and experimentation. It features a compact design:



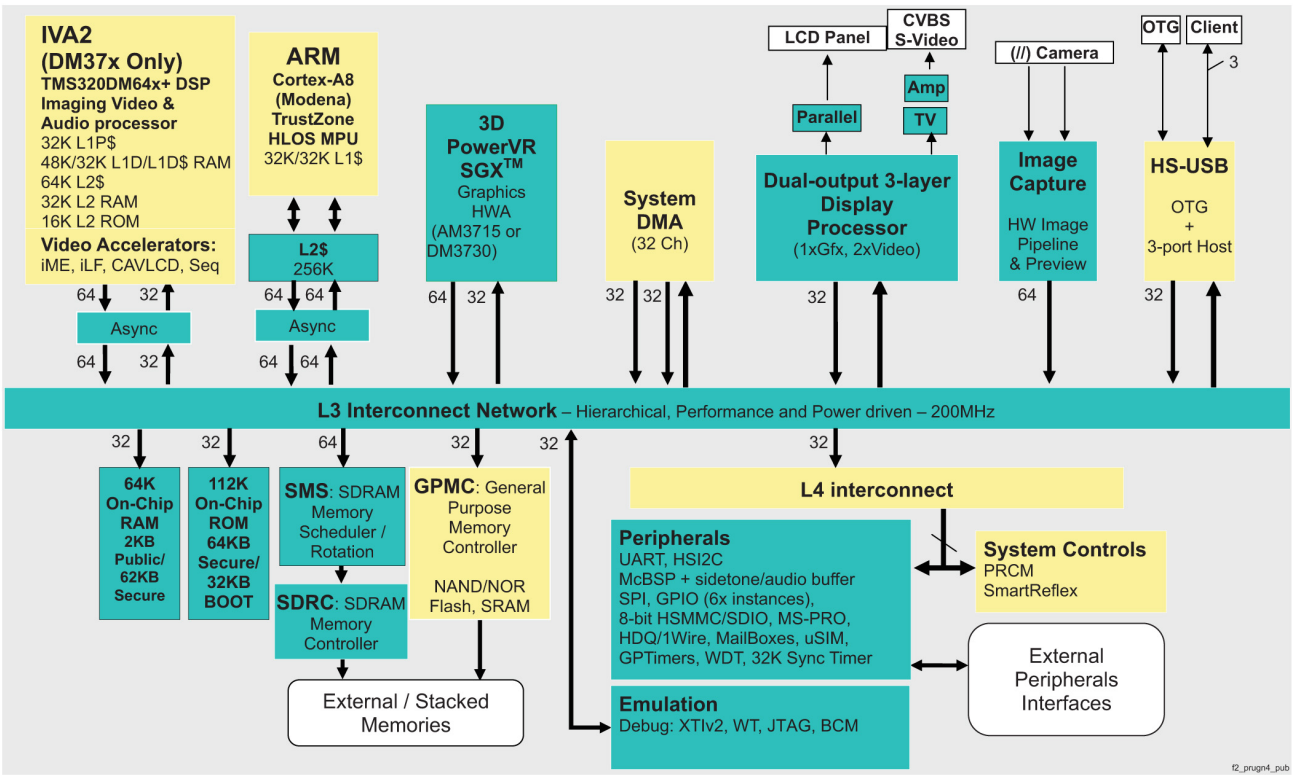
Here is a list of BeagleBoard-xM features from the manual:

Table 2. BeagleBoard-xM Features

	Feature	
Processor	Texas Instruments Cortex A8 1GHz processor	
POP Memory	Micron 4Gb MDDR SDRAM (512MB) 200MHz	
PMIC TPS65950	Power Regulators	
	Audio CODEC	
	Reset	
	USB OTG PHY	
Debug Support	14-pin JTAG	GPIO Pins
	UART	3 LEDs
PCB	3.1" x 3.0" (78.74 x 76.2mm)	6 layers
Indicators	Power, Power Error	2-User Controllable
	PMU	USB Power
HS USB 2.0 OTG Port	Mini AB USB connector	
	TPS65950 I/F	
USB Host Ports	SMSC LAN9514 Ethernet HUB	
	4 FS/LS/HS	Up to 500ma per Port if adequate power is supplied
Ethernet	10/100	From USB HUB
Audio Connectors	3.5mm	3.5mm
	L+R out	L+R Stereo In
SD/MMC Connector	MicroSD	
User Interface	1-User defined button	Reset Button
Video	DVI-D	S-Video
Camera	Connector	Supports Leopard Imaging Module
Power Connector	USB Power	DC Power
Overvoltage Protection	Shutdown @ Over voltage	
Main Expansion Connector	Power (5V & 1.8V)	UART
	McBSP	McSPI
	I2C	GPIO
	MMC2	PWM
2 LCD Connectors	Access to all of the LCD control signals plus I2C	3.3V, 5V, 1.8V
Auxiliary Audio	4 pin connector	McBSP2
Auxiliary Expansion	MMC3	GPIO,ADC,HDQ

The processor includes an ARM Cortex processor, memory, acclerators, and a variety of peripherals, as shown in Figure 1-2 of the AM/DM37x Technical Reference Manual:

Figure 1-2. Block Diagram



Here are some useful documents:

- BeagleBoard-xM System Reference Manual
  http://beagleboard.org/static/BBxMSRM_latest.pdf
- TI DM3730 manual
- TI DM3730 documents http://www.ti.com/product/dm3730#technicaldocuments
- The BeagleBoard cheat sheet available on the resources section of the class tsquare site.

Georgia Institute of Technology

1.2 BeagleBoard-xM software and tools

Several different tool sets can be used to program this board; we use an open source version of the Texas Instruments Code Composer Studio. The processor on the board runs Linux.

You can use the lab setups but you can also buy your own BeagleBoard-xM and run tools on your own machine. You may find it easier to install the tools under Ubuntu.

Here are some useful documents:

- Sitara tools and manuals http://www.ti.com/tool/linuxezsdk-sitara
- Sitara Linux Software Developer's Guide http://downloads.ti.com/dsps/dsps_public_sw/am_bu/sdk/BeagleBoardSDK/latest/exports/sitara-linuxsdk-sdg-05.05.00.00.pdf

Georgia Institute of Technology

1.3 Lab Practices

Only a basic level of C programming skill is required for this course. However, if you feel that you need additional references on C, C++, or Linux, consult the following tutorials:

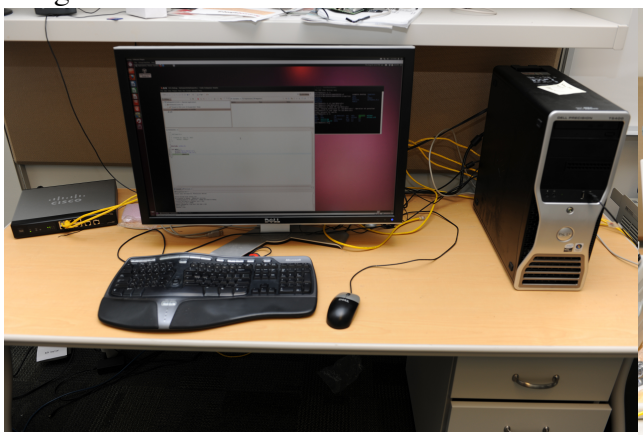
C Tutorial
http://www.cprogramming.com/tutorial.html#ctutorial

C++ Language Tutorial
http://www.cplusplus.com/doc/tutorial/

University of Surrey Linux tutorial:
http://www.ee.surrey.ac.uk/Teaching/Unix/

2 Lab Procedure

Labs will be performed in Klaus 2411. The development environment is fairly complex. The tools run on a Linux box, which talks to the Beagleboard via Ethernet and serial ports. The Beagleboard network is a separate, standalone network. The Beagleboard serial port is echoed back on the Linux development system. The Beagleboard also drives an HDTV display. You should not touch the Beagleboard or its cables for now.



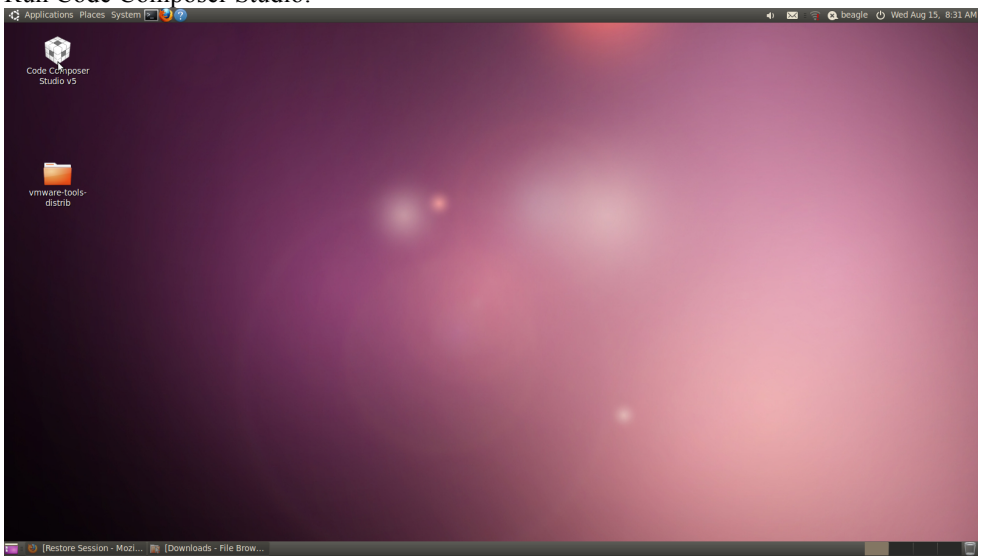
The tool environment itself is complex. The host system is Ubuntu. The tool set, Texas Instruments Code Composer Studio (CCS), runs in a VMware virtual machine on the host system. The tools talk to the Linux kernel on the Beagleboard; some of the operations in development are performed by typing commands into the Beagleboard's Linux shell prompt while others are run in the virtual machine host.

Starting CCS

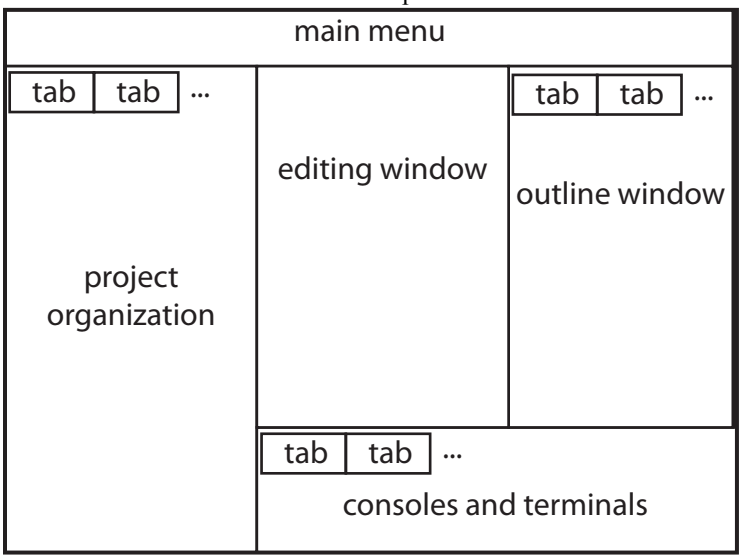
Starting the Code Composer Studio requires several steps:

- Use the automated login to log into the Ubuntu host.

- Run the VMware player by clicking on the VMware player logo:  This logo may be on the left-hand tool bar or it may be accessed by clicking on the top-left logo for recently used tools.
- Open the virtual machine archive File System/usr/vm1/setup_1/setup_1.vmx. Select play virtual machine.
- A separate copy of Ubuntu will run inside the player. Log into this system using the your OIT account name and password abcd.
- Run Code Composer Studio:



The CCS window is made of a set of panes some of which have tabs to control different views into the pane:



- The main menu pane gives global commands for CCS.
- The project organization pane lists projects, remote systems, etc.
- The editing window is used to edit a file.
- The outline window provides a view of the files in the project.
- The consoles and terminals pane provides output of build steps (console), the connection to the Beagleboard (terminal), etc.

CCS operates several different perspectives including:

- CCS Edit to edit source files and build projects.
- CCS Debug to run the debugger.
- Remote System Explorer to talk to the target system.

You can switch perspectives from the top-level menu or from the tab on the upper-right side of the screen below the menu bar.

Creating your project

A project is a set of source files, configuration data, etc. to build an executable. You will create your own project by copying an existing project. We keep copies of files you will use for your lab in File System/Lab_Materials. Your own project should be kept in your directory. Do not modify the original project.

- Select File->Import->General->Existing Projects into Workspace. Check "Copy Projects into Workspace."
- Browse for File System/Lab_Materials/lab1, then select OK and finish.

Building your project

The project contains source code for helloworld.c and other information that CCS needs to build an executable.

- Compile and link your project using Project->Build All or Project->Build Project

Running your project

Running your project requires several steps: possibly connecting to the Beagleboard, opening a terminal connection to the Beagleboard, moving the binary to the Beagleboard, and actually executing the binary.
First start the Remote System Explorer perspective:

- Run Remote System Explorer using Window->Open Perspective->Other; check show all, then select Remote System Explorer and hit OK. Once it is open you can select it from the upper-righthand corner.

You should see BeagleBoard-xM listed as one of the systems in the right-hand pane. If you don't, you need to start a connection:

- Right-click Local->New->Connection->Linux->Remote System Explorer->BeagleBoard-xM. Set the IP address for your board and set any port numbers to 10000.

To set up a terminal connection:

- Enter Remote System Explorer, then right click on Beagleboard, then right click to connect
- Login to the Beagleboard as root with no password
- Right click on Ssh terminals->Connect->Launch terminal, then Ssh terminals->Launch terminal.

To move your binary to the Beagleboard:

- Go to the CCS Edit perspective. Select C++ -> C/C++ Projects.
- Go to binaries->helloworld. Right click to copy.
- Go to the Remote System Explorer perspective. Select BeagleBoard-xM->Sftp file->My Home, then right click paste.
- To make the file executable, right click helloworld->properties->executable, or go to the terminal window and use the Linux chmod command (chmod ogu+x helloworld).

To execute your program:

- Select the Terminal tab at the bottom of the screen.
- % ./helloworld

Debugging your project

The gdb debugger is also a complex tool. To practice using it, add a second printf statement to your program so that the program has at least two lines. You can find a detailed walkthrough of gdb under CCS in the Sitara Linux Software Developer's Guide (find it using the link in Section 1.

You must first start gdb on the target system:

- Go to the Remote System Explorer perspective. Select the Terminal tab at the bottom of the screen. Start the gdb server %gdbserver :10000 helloworld. The gdb server must be restarted on every execution of the program.

You then set up and start gdb on the host system.

- Go to the CCS Edit perspective. Select the C/C++ Projects on Project Explorer tab.

- Right click helloworld-> Debug configurations->paste.
- Go to the CCS Edit perspective. Select helloworld-> Debug configuration->Other->Manual Remote Debugging, launcher
- Check this window to be sure the settings are correct: IP address 192.168.1.141, port 10000; Main->File System/home/beagle/debugger/ti-sdk…/linux-devkit/bin/ddbinit; Main->Shared libraries-> File System/home/beagle/debugger/ti-sdk…/gdbinit
- Select Apply->Debug in the Manual Remote Debugging pane to start the debugger.
- Hit the Step Into button to execute the first line of the program. The output will appear in the terminals tab.

3 Lab Report

You should turn in:

- An email to wolf@ece.gatech.edu stating that you compiled helloworld and ran the debugger.