Implementation of Real-Time Applications on ARM Microcontroller

Fawad[1] and Dr. Imran Amin[2]
Imran.amin@szabist.edu.pk
SZABIST
Karachi, Pakistan

Abstract: *The ARM is a 32-bit reduced instruction set computer(RISC) architecture for microprocessor developed by ARM Limited[1]. I is also known as Advance RISC[2] Machine. AMR is suitable for the low power applications which made them leading in the mobile and embedded electronics industry relatively low cost and small microprocessor and microcontroller. The ARM architecture is licensable. companies that are currently ARM licenses include Alcatel, Atmel, Broadcom, Intel, LG, NVIDIA, Yamaha ST Microelectronics, Samsung and many more. ARM processor families include the ARM7, ARM9, ARM11 and Cortex.*

The purpose of this Research is to study the ARM architecture and uses of it on real-time Applications using ARM Cortex-A8 microprocessor, which is one of the microprocessor from ARM family. Beagleboard[5] is used for experimental setup and testing real-time application on ARM microprocessor. The Beagle Board is a low-power, low-cost Single-board computer developed by worldwide community. It is amazingly 3x3 inch width and height. As a open-source application the Beagle Board is the open-embedded hardware architecture which is available for develop and further enhancement.

Keywords: *Begal Board, Microprocessor, embedded systems , OMAP3, ARM, Cortex-A8*

1. INTRODUCTION

The Embedded systems is the special kind of computer system designed to perform one or multiple dedicated functions, often with reltime application constraints. To perform the specific task in embedded systems there is special kind of microprocessor is used namely ARM (Advance RISC Machines). The ARM microprocessor is low cost, high performance in low power consumption, which make this ARM architecture gient of electronics market.

1.1 ARM - Advanced RISC Machine

ARM is the RISC microprocessor based on 32bit architecture[1]. Everything it need to create an innovative product design based on industry-standard components that are 'next generation' compatible. ARM provides the multiple verity of processors on a its architecture, that gives high performance together with low power consumption and system cost. The ARM processor range provides solutions for:

- Open platforms running complex operating systems for wireless consumer and imaging applications.
- Embedded real-time systems for mass storage, automotive, industrial and networking applications.
- Secure applications including smart cards and SIMs.

Figure 1: A Conexant ARM processor used mainly in routers

ARM has a vast experience on developing embedded systems which delivers technology that achieves unsurpassed Power, Performance and Area. Currently ARM validates new core architectures by bringing up the latest version of Symbian OS[2], Microsoft Windows CE[3] and Linux to ensure the best compatibility and smooth development.

1.2 RISC Architecture

RISC, or Reduced Instruction Set Computer is a type of microprocessor architecture that utilizes a small, highly-optimized set of instructions, rather than a more specialized set of instructions often found in other types of architectures.[3]

The first RISC projects came from IBM, Stanford, and UC-Berkeley in the late 70s and early 80s. The IBM 801, Stanford MIPS, and Berkeley RISC 1 and 2 were all designed with a similar philosophy which has become known as RISC. Certain design features have been characteristic of most RISC processors:[4]

One cycle execution time: RISC processors have a CPI (clock per instruction) of one cycle. This is due to the optimization of each instruction on the CPU and a technique called ;

Pipelining: A technique that allows for simultaneous execution of parts, or stages, of instructions to more efficiently process instructions;

Large number of registers: The RISC design philosophy generally incorporates a larger number of registers to prevent in large amounts of interactions with memory.

1.3 ARM Architecture

The ARM is a 32-bit reduced instruction set computer (RISC) instruction set architecture (ISA) developed by ARM Limited. It was known as the Advanced RISC Machine, and before that as the Acorn RISC Machine. The ARM architecture is the most widely used 32-bit ISA in terms of numbers produced. They were originally conceived as a processor for desktop personal computers by Acorn Computers, a market now dominated by the x86 family used by IBM PC compatible computers. The relative simplicity of ARM processors made them suitable for low power applications. This has made them dominant in the mobile and embedded electronics market as relatively low cost and small microprocessors and microcontrollers.

As of 2009, ARM processors account for approximately 90% of all embedded 32-bit RISC processors. ARM processors are used extensively in consumer electronics, including PDAs, mobile phones, digital media and music players, hand-held game consoles, calculators and computer peripherals such as hard drives and routers.

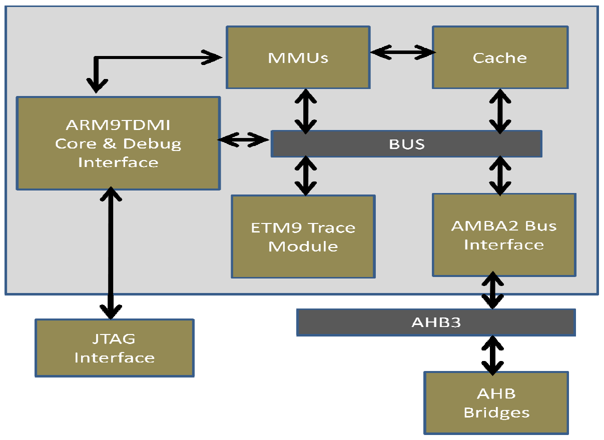

Figure 2: ARM922T Processor

The ARM architecture is licensable. Companies that are currently or formerly ARM licensees include Atmel, Broadcom, LG, Marvell Technology Group, NVIDIA, Samsung, Sharp, Texas Instruments and many more.

ARM processors are developed by ARM and by ARM licensees. Prominent examples of ARM Limited ARM processor families include the ARM7, ARM9, ARM11 and Cortex.

2. EMBEDED SYSTEMS

An embedded system is a special type of system designed to perform one or a few dedicated functions, with real-time computing constraints. It is embedded as part of a complete device often including hardware and mechanical parts. [4]

Embedded systems are controlled by a main processing core that is typically either a microcontroller or a digital signal processor (DSP).

Since the embedded system is dedicated to specific tasks, design engineers can optimize it reducing the size and cost of the product and increasing the reliability and performance. Some embedded systems are mass-produced, benefiting from economies of scale.

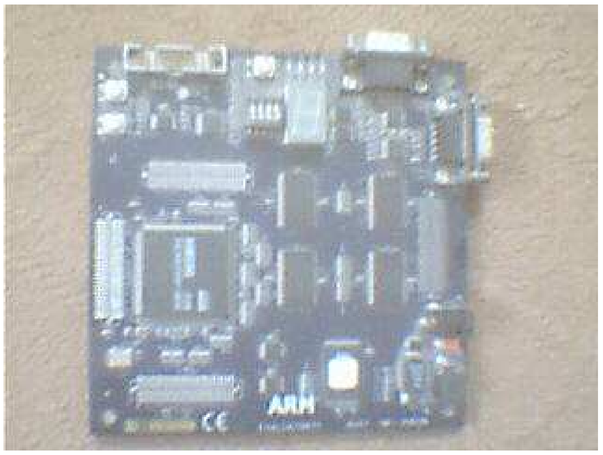

Figure 3: ARM7 Embedded Board

3. BEAGLE BOARD

The Beagle Board is a low-power, low-cost Single-board computer produced by Texas Instruments in association with Digi-Key, designed with open source development in mind, to demonstrate the Texas Instrument's OMAP3530 system-on-a-chip. The board was developed by a small team of TI engineers.[5]

The USB-powered Beagle Board is a low-cost, fan-less single board computer that unleashes laptop-like performance and expandability without the bulk, expense, or noise of typical desktop machines.

OMAP3530 processor highlights:

- Over 1,200 Dhrystone MIPS using the superscalar ARM Cortex-A8 with highly accurate branch prediction and 256KB L2 cache running at up to 600MHz[6]
- OpenGL© ES 2.0 capable 2D/3D graphics accelerator capable of rendering 10 million polygons per second
- HD video capable TMS320C64x+ DSP for versatile signal processing at up to 430MHz
- USB power via complete chip-set with minimal additional power-consuming logic

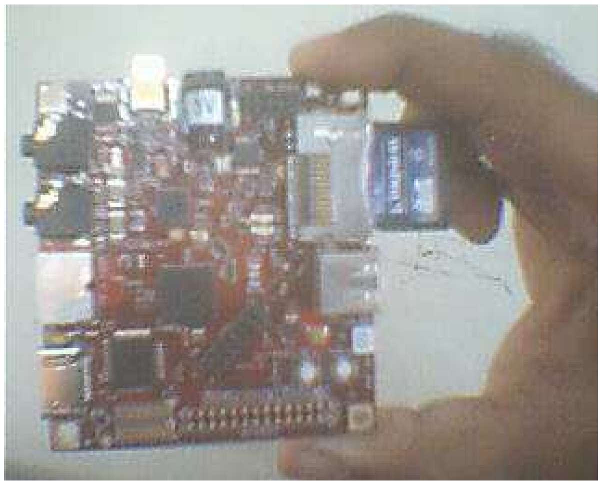

Figure 4: Top view of Beagle Board

Following is the Expansion capability and power options which provided by beagle board:

- DVI-D for connecting digital computer monitors
- MMC+/SD interface for memory
- S-Video out for connecting NTSC or PAL
- Stereo audio in and out
- Power via typical USB chargers for cell phones

4. ARM CORTEX-A8

The ARM CortexTM-A8 processor is the first applications processor based on the ARMv7 architecture and is the highest performance, most power-efficient processor available from ARM. With the ability to scale in speed from 600MHz to greater than 1GHz, the Cortex-A8 processor can meet the requirements for power-optimized mobile devices needing operation in less than 300mW; and performance-optimized consumer applications requiring 2000 Dhrystone MIPS. [7]

Architectural Features:

The ARM Cortex-A8 processor's sophisticated pipeline architecture is based on dual, symmetric, in-order issue, 13-stage pipelines with advanced dynamic branch prediction achieving 2.0 DMIPS/MHz.

In-order, dual-issue, superscalar microprocessor core

- 13-stage main integer pipeline
- 10-stage NEON media pipeline
- Dedicated L2 cache with programmable wait states
- Global history based branch prediction

Works in conjunction with a power optimized load store pipeline to deliver 2.0 DMIPS/MHz for power sensitive applications

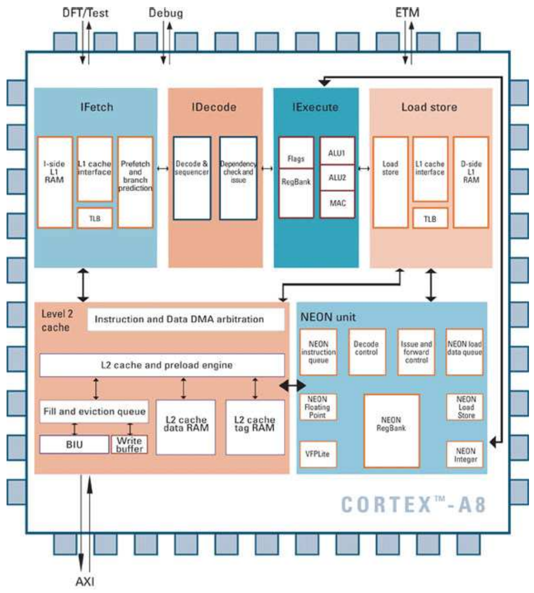

Figure 5: Arm Cortex-A8 microprocessor Architecture (used in Beagle Board)

5. LINUX OF EMBEDED AND REAL TIME APPLICATION

In our daily life the embedded system play a vital role. In which real time application is used in the form of Operating system or any package. Linux is one of the biggest supporter of embedded system because of its open source capability.

Linux is a generic term referring to Unix-like computer operating systems based on the Linux kernel. Their development is one of the most prominent examples of free and open source software collaboration; typically all the underlying source code can be used, freely modified, and redistributed, both commercially and non-commercially, by anyone under the terms of the GNU GPL[8].

What is Real-time

In computing, real-time refers to a time frame that is very brief, appearing to be immediate. When a computer processes data in real time, it reads and handles data as it is received, producing results without delay. For example, a website that is updated in real-time will allow its viewers to see changes as soon as they occur, rather than waiting for updates to be visible at some later date. [9]

A non-real-time computer process does not have a deadline. Such a process can be considered non-real-time, even if fast results are preferred. A real-time system, on the other hand, is expected to respond not just quickly, but also within a predictable period of time. A good example of a real-time computer system is a car's anti-lock break system. An anti-lock brake system is expected to release a vehicle's brakes, preventing dangerous wheel locking, in a predictably short time frame.

Development Environment and Tools

The ARM Real View Development Suite 4.0 Professional is a complete, end-to-end solution for software development supporting all ARM processors and ARM CoreSigh debug technology. This full-featured product enables developers to begin software development, optimization and test ahead of silicon availability, significantly reducing application time-to-market and ensuring the highest degree of software quality.[10]

6. EXPERIMENTAL METHODOLOGY

It is little bit difficult to install OS linux on Beagleboard at first time. If there is no guideline then person will be lost. Once it are familiar with any embedded system then it can easily working on it. In this chapter there is define that how to install linux distribution on the Beagleboard.

Setup the Installation Enviroment

Before installation of Linux on beagleboard there is some accessories which is required for installation.

- HDMI to DVI connector : this will used for display on LCD
- OTG USB to Female USB-A connector for connecting Keyboard, and mouse.
- USB to Serial Cable: this cable is connecting to USB-A connector and in the other end if this cable two connector is available for keyboard and mouse.

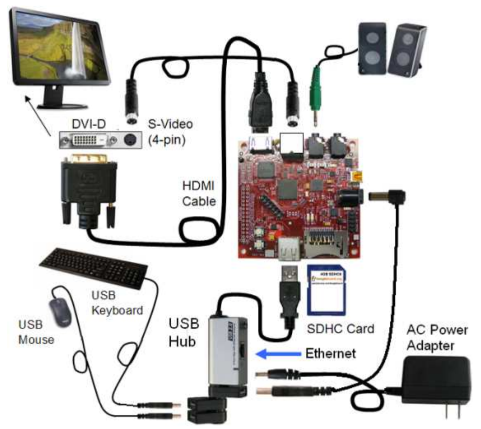

Figure 6: cables and connection diagram for Beagleboard

Installing Linux sakoman r9 GNOME distribution

To installing the Linux sakoman r9 GNOME[11] there is requirement of development machine which have linux install. In current installation Ubuntu 9.10 is used for development machine OS. The installation of OS on beagleboard is not like just put cd on cdrom and install on it. As a primary storage Beagleboard uses the SD/MMC card.[12]

Getting started

First insert the card into development machine's flash card slot. The newly inserted card shows up as /dev/sde and that is the device name that will be used through this example. It should substitute the proper device name for the machine.

Partitioning the card

Now launch fdisk and create an empty partition table. Note that the argument for fdisk is the entire device (/dev/sde) not just a single partition (i.e. /dev/sde1):

```
# sudo fdisk /dev/sde
```

Building a new DOS disklabel. Changes will remain in memory only, until it will decide to write them. After that, of course, the previous content won't be recoverable.

To go into "Expert" mode use the "x" command:

```
Command (m for help): x
```

Next set the geometry to 255 heads, 63 sectors and a calculated value for the number of cylinders required for the particular microSD card.

To calculate the number of cylinders, take the 2032664576 bytes reported above by fdisk divided by 255 heads, 63 sectors and 512 bytes per sector

Change the partition type to FAT32:

```
Command (m for help): t
Selected partition 1
Hex code (type L to list codes): c
Changed system type of partition 1 to c (W95 FAT32 (LBA))
```

And mark it bootable:

```
Command (m for help): a
Partition number (1-4): 1
```

Next create an ext3 partition for the rootfs:

```
Command (m for help): n
```

Up to this point no changes have been made to the card itself, so our final step is to write the new partition table to the card and then exit:

```
Command (m for help): w
The partition table has been altered!
Calling ioctl() to re-read partition table.
```

Formatting the new partitions

Format the first partition as a FAT file system (the -n parameter gives it a label of FAT, It can change or omit this)

```
# Sudo mkfs.vfat -F 32 /dev/sde1 -n FAT
```

Format the second partition as an ext3 file system:

```
$ sudo mkfs.ext3 /dev/sde2
```

This filesystem will be automatically checked every 36 mounts or 180 days, whichever comes first. Use tune2fs -c or -i to override.

Installing the boot files

Make changes as appropriate if there are using a different machine or have downloaded the files to a different location. It will also need to adjust the revision number in the downloaded filenames.
There are three files required on the first (FAT) partition to boot the OMAP3 processor:

- MLO: the boot-loader loader - this small program is loaded into the OMAP3 processor's static RAM.
- u-boot.bin: the boot loader
- uImage: the linux kernel

```
sudo mount /dev/sde1 /media/card
$ sudo cp ~/Desktop/MLO-overo /media/card/MLO
$ sudo cp ~/Desktop/u-boot-overo.bin /media/card/u-boot.bin
$ sudo cp ~/Desktop/uImage-overo-2.6.X-rX.bin /media/card/uImage
```

The final step is to untar the rootfs onto the ext3 partition that it created above.

```
$ sudo mount /dev/sde2 /media/card
```

Now untar thr rootfs:

```
$ cd /media/card
$ sudo tar xvjf ~/Desktop/gnome-rX.tar.bz2
```

It is now safe to remove the SD/microSD card from the development machines card slot.

Insert the card in OMAP3 machine and press the reset button (on Beagle be sure to hold down the User button when pressing and releasing the reset button in order to ensure that the Beagle boots from the SD card)
After a short while it will see a login screen. Login as user root with password root. If it desire, it can then use the System/Administration/Users and Groups menu item to create a normal user.

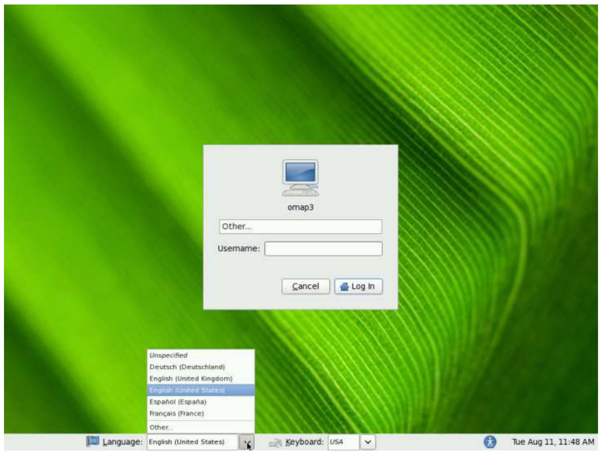

Figure 7: Login Screen of r9 GNOME OMAP3 OS on Beagle Board

Screen Shorts:

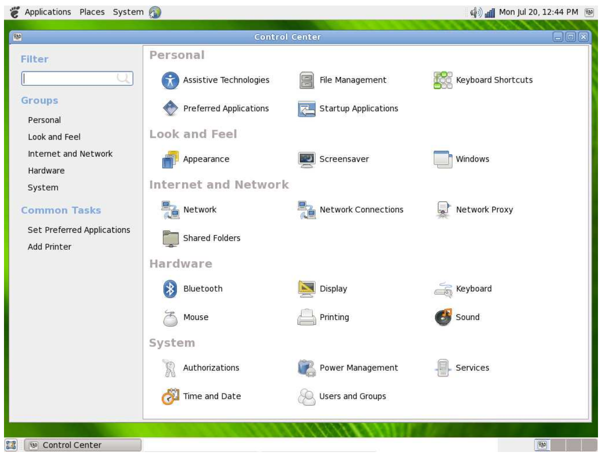

Figure 8: Control Center of OS and desktop preview

7. DISCUSSION

- The first issue which is come between installation is that keyboard and mouse is not connecting. This will resolved by short the 4th and 5th cable of USB-OTG.
- The second issue which is come that the OTG to USB-A cable is not easily available in market. So after purchasing the two cables OTG to USB male and USB male to USB female. The new cable is wired.
- There is only one usb connector is available in board so if there is requirement of more devices to connect then it must be required USB hub with at least 4 port.

8. REFERENCES

[1] Advance RISC Machine - http://www.arm.com/products/CPUs/

[2] Symbian - http://www.symbian.org

[3] Standfort University - http://cse.stanford.edu

[4] Embedded Systems Design Second Edition by Steve Heath, Page 1, EDN,2008

[5] Beagle Board - http://www.beagleboard.org

[6] Beagle Board Product Detail - http://beagleboard.org/hardware

[7] ARM product Details - http://www.arm.com/products/CPUs/ARM_Cortex-A8.html

[8] Linux - http://www.linux.org/

[9] "Linux for Embedded and REal-time Applications" by Dog Abbott, page 2, Newnes , 2009

[10] RealView Development Suite 4.0 - http://www.arm.com/products/DevTools/

[11] SAKOMAN - http://www.sakoman.com/

[12] Preparing a bootable SD card - http://www.sakoman.com/OMAP3/