INTERNATIONAL JOURNAL OF RECENT TRENDS IN ENGINEERING & RESEARCH
Impact Factor 4.101 ISSN -2455-1457
RECENT ADVANCES IN ELECTRONICS & COMMUNICATION ENGINEERING (RAECE-2K17)

IMPLEMENTATION OF MULTI FORMAT CODEC USING ARM11 MP CORE PROCESSOR

Madhu Shiramshetty [1]
Assistant professor,
Department of ECE, NNRG institutions,
Telangana, India.

Abstract

The required processing performance of embedded processor core is getting higher and higher without increasing power consumption dramatically. In same time, large SoC design has more risk of re-spin and long design time due to the complexity and difficulty of verification. ARM offers multi core solution to overcome such a situation over various applications. Currently the extensive application of embedded multimedia processing calls for requirement of hardware acceleration and multi-function expansion, it firmly requires a scalable and efficient design of system architecture for embedded multimedia processing terminal. At first, system architecture with distinct structures and reasonable modules is designed, a simplified multimedia framework is designed for embedded application, which includes a scheduler and a multimedia processing engine based on FFMPEG. The design of engine interfaces focuses on compatibility and scalability to deal with a variety of applied requirements. Then, embedded multimedia processing software is implemented based on the system architecture design. A scheduling program which schedules modules is implemented. A media player engine with scalable software interfaces is developed. Eventually, on Real6410 platform, an embedded multimedia system is built, which makes use of LINUX Frame buffer device driver interface to display video and library to play audio for testing the system. The actual operation result proves that system architecture design simplifies the multimedia processing and supports multiple media formats as well as hardware acceleration. The scalability and efficiency make it suitable for embedded application.

Key Words: *ARM MPCore, S3c6410, Linux, MPEG*

----------***----------

1. INTRODUCTION

Embedded systems can be thought of as special purpose computers. Typically they offer limited functionality compared to their counterpart, the general purpose desktop computer. Additionally, their environment imposes several requirements for the hardware. Some of these requirements are cost effectiveness and robustness against extreme temperatures, humidity, vibration and sometimes even radiation. On the other hand, the device itself shall not interfere with its environment, so electromagnetic compatibility must be addressed as well. Moreover, the embedded device might be required to run on battery power only. The ordinary desktop computers utilize operating systems to abstract the underlying hardware under standard interfaces. On embedded systems, the application software can comprise of a simple assembly language program. However, as the size and features of the embedded software grows, an operating system is often used to reduce the programmer's workload. Many special purpose embedded operating systems exist.

They try to address the difficulties presented by the embedded devices, namely limited processor and memory resources as well as the need of robustness and real-time applications. Sometimes the hardware lacks vital resources in which the lack of Memory Management Unit (MMU) is perhaps the most important. The common desktop operating systems utilize MMU in order to implement virtual memory. The application development on these special purpose embedded operating systems is hindered by unfamiliar development conventions. These include application programming interfaces and development tool chain. On the other hand, general purpose operating systems lend themselves to easy application development due to well established development framework. Therefore, it is desirable to use as general purpose operating system as possible. This requires addressing of the problems introduced by the aforementioned special requirements for the embedded usage.

2. ARM ARCHITECTURE

ARM is a 32-bit reduced instruction set computer (RISC) instruction set architecture (ISA) developed by ARM Holdings. 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.[1][2] Originally conceived by Acorn Computers for use in its personal computers, the first

ARM-based products were the Acorn Archimedes range introduced in 1987.

The relative simplicity of ARM processors makes them suitable for low power applications. As a result, they have become dominant in the mobile and embedded electronics market, as relatively low-cost, small microprocessors and microcontrollers. In 2005, about 98% of the more than one billion mobile phones sold each year used at least one ARM processor.[3] As of 2009, ARM processors account for approximately 90% of all embedded 32-bit RISC processors[4] and 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.

3. The ARM MPCore

ARM MPCore, based on the ARM11 micro architecture (8 pipeline stages), combines the efficiency of the ARMv6 architecture with the scalability of a multiprocessor core. It is available as a single synthesizable block containing a configurable number of CPU with individual cache size and VFP attached or not and an architected interrupt controller for fast inter-process communication. Whatever its configuration, it offers to the system designer the same interface as a single CPU with two 64-bit AMBA™ AXI™ ports, a debug port and interrupt lines inputs. The memory coherency has been localized to Level 1 data memory only, focusing on the situation where the Core is one of several processors in a system. That scheme allows performing all the coherency management at CPU frequency without having to deal with a new system bus protocol with potential lower speed that could have reduced the overall system performance. MPCore can then be seen as an 'island of coherency' in a non-coherent sea. That also eases the integration of a Level 2 cache between the core and the rest of the system as that second cache has no need to have any knowledge of the coherent traffic. Individual CPUs can be dynamically configured as operating in a symmetric (SMP mode) or asymmetric (AMP mode) manner, i.e. taking part in the L1 coherency or not. That allows existing non MP aware software to be ported easily to MPCore without any change. Software designers can then slowly start to split tasks between individual CPUs before applying dynamic task allocation techniques to their software.

4. S3c6410

The Mini6410 development board is an excellent ARM11 board offering a comprehensive solution integrating both hardware and software. It is designed, developed and distributed by Friendly ARM in Guangzhou, China. It applies the Samsung S3C6410 microprocessor and inherits all the features and benefits of our most popular Mini2440 products excelling in quality and easy to use with low cost. Compared to our previous products it has more reliable design and varied interfaces. These features make it easily and widely used in MID development, auto electronic devices, industrial applications, GPS systems and multimedia systems. It is good for educational training, embedded development and DIY as well.

The 6410 is a 16/32-bit RISC microprocessor, which is designed to provide a cost-effective, low-power capabilities, high performance Application Processor solution for mobile phones and general applications. To provide optimized H/W performance for the 2.5G & 3G communication services, the 6410 adopts 64/32-bit internal bus architecture. The 64/32-bit internal bus architecture is composed of AXI, AHB and APB buses. It also includes many powerful hardware accelerators for tasks such as motion video processing, audio processing, 2D graphics, display manipulation and scaling. An integrated Multi Format Codec (MFC) supports encoding and decoding of MPEG4/H.263/H.264 and decoding of VC1. This H/W Encoder/Decoder supports real-time video conferencing and TV out for both NTSC and PAL mode. Graphic 3D (hereinafter 3D Engine) is 3D Graphics Hardware Accelerator which can accelerate OpenGL ES 1.1 & 2.0 rendering. This 3D Engine includes two programmable shaders: one vertex shader and one pixel shader.

The 6410 has an optimized interface to external memory. This optimized interface to external memory is capable of sustaining the high memory bandwidths required in high-end communication services. The memory system has dual external memory ports, DRAM and Flash/ROM. The DRAM port can be configured to support mobile DDR, DDR, mobile SDRAM and SDRAM. The Flash/ROM port supports NOR-Flash, NAND-Flash, One NAND, CF and ROM type external memory. To reduce total system cost and enhance overall functionality, the 6410 includes many hardware peripherals such as a Camera Interface, TFT 24-bit true color LCD controller, System Manager (power management & etc.), 4- channel UART, 32-channel DMA, 5-channel 32bit Timers with 2PWM output, General Purpose I/O Ports, I2S-Bus interface, I2C-BUS interface, USB Host, USB OTG Device operating at high speed (480Mbps), 3-channel SD/MMC Host Controller and PLLs for clock generation. The ARM subsystem is based on the ARM1176JZF-S core. It includes separate 16KB Instruction and 16KB data caches, 16KB Instruction and 16KB Data TCM. It also includes a full MMU to handle virtual memory management. The ARM1176JZF-S is a single chip MCU, which includes support for JAVA acceleration. The ARM1176JZF-S includes a dedicated vector floating point coprocessor allowing efficient implementation of various encryption schemes as well as high quality 3D graphics applications. The S3c6410 adopts the de-facto standard AMBA bus architecture. These powerful, industry standard

features allow the S3c6410 to support many of the industry standard Operating Systems.

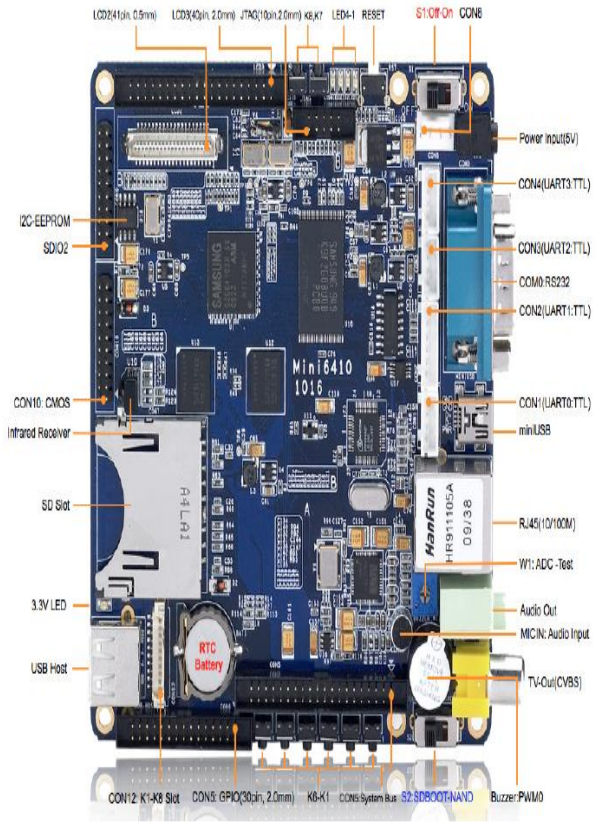

Fig.4.1 Hardware architecture for ARM11 processor

5. SOFTWARE DESIGN

We choose Linux as the operating system of the embedded system developing platform. Linux is a 32-bit, open and tailored embedded real-time operating system. It has the virtue of modularization, scalability, strong communication ability and good real-time, also can support a wide range of CPU. Linux is widely used in the development of various embedded intelligent devices, such as industrial control, information appliance, mobile communication, automotive electronics and Personal electronic consumer goods.

5.1. Linux Features

Kernel Version
- Linux 2.6.38(or latest) Boot Loader
- U-boot-1.6.1: open source, can be configured as Nand booting or SD booting
- Super boot: developed by Friendly ARM, not open sourced.

File Systems
- YAFFS2: recommended
- UBIFS: recommended for MLC NAND Flash.
- CRAMFS
- EXT2/3
- FAT32
- NFS

Drivers (all open sources)
- Driver for 4 serial ports
- DM9000 driver
- Audio driver (WM9714)
- RTC driver
- Drivers for 4 User LEDs
- USB host driver
- LCD driver (it supports 3.5-inch, 4.3-inch, 7-inch,
-LCD2VGA1024x768
-LCD2VGA800x600
- LCD2VGA640x480 and EZVGA800x600)
-4-wire touch screen driver and 1-wire precise touch driver
- USB camera driver
- Drivers for USB mouse, keyboard, flash drive and portable hard disk
- SD card driver, up to 32 GB
- I2C driver
- ADC driver
- LCD backlight driver
- Watchdog driver (watchdog reset is cold reset)
- Multimedia drivers (including JPEG, FIMC, MFC, 2D/3D Accelerator, TVENC and TVSCALER)
- CMOS camera driver
- Back light adjustor. This allows users to adjust the board's backlight up to 127 levels and experience a gradually dim effect when turning it down.
- SPI driver.

6. CREATING THE RAW KERNAL

The main work that utilizing Linux to carry on the development of embedded operating system is to customize the operating system. That is, we need to customize corresponding. The resources of embedded system are limited, so we should minimize the size of the kernel under the terms of meeting the function demand and guaranteeing its stability in order to save resources.

Flow chart of the operating system customization:

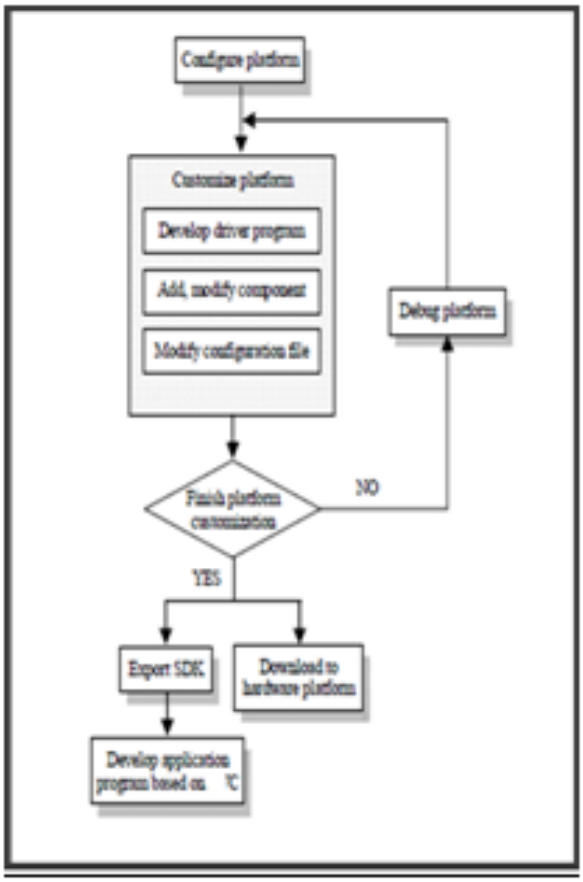

Fig.6.1- Flow chart of the operating system customization

7. EMBEDDING SOFTWARE INTO HARDWARE

Porting Android on ARM based target platforms can be divided in two stages:

Porting Linux

Following are the main steps for porting Linux:

- Download the patches of Linux Kernel for supporting ARM processor.
- Prepare the ARM based tool chain for compilation of Linux source code.
- Choose and Configure boot loader as per target board.
- Compile the boot loader.
- Configure Linux kernel as per target platform (this may require developing some additional device drivers or customization of the existing device drivers for the reference design board).
- Compile the Linux kernel source
- Burn compressed Linux image on the target platform
- Update Board specific components such as Codec, Camera, Audio, WIFI, Power Management, Bluetooth etc.
- Compile Android source code
- Burn system image on Target platform

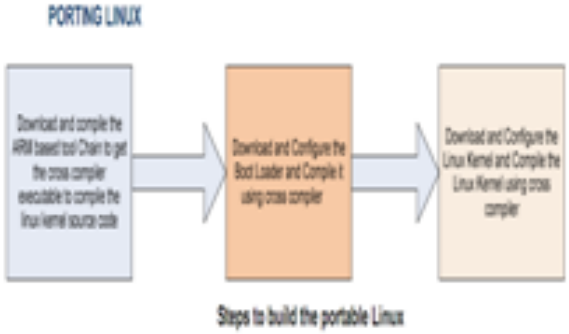

Fig.7.1- Steps to build the Portable Linux

8. APPLICATION

FIMV-MFC V1.0 is a multi-standard video codec IP that can handle the H.263P3, MPEG-4 Single Profile H.264 Baseline Profile and VC-1 Main Profile in single codec hardware. The FIMV-MFC V1.0 includes the following features.

• Multi-standard video codec – MPEG-4 simple profile encoding/decoding
– H.264/AVC baseline profile encoding/decoding
– H.263 P3 encoding/decoding
– VC-1(WMV9) main profile decoding
– Multi-party call is supported
Ex. Simultaneous 1 stream encoding and 3 streams decoding are possible.
– Multi-format is supported
Ex. The video IP encodes the MPEG-4 bit stream, and decodes H.264 bit stream in a time-division multiplexing manner.

• Coding tools
– [-16,+16] 1/2 and 1/4-pel accuracy motion estimation
– All variable block sizes are supported.
* In case of encoding, 8x4, 4x8, and 4x4 block sizes are not supported.
– Unrestricted motion vector
– MPEG-4 AC/DC prediction
– H.264/AVC intra-prediction
– H.263 Annex I, J, K (RS=0), and T are supported
* In case of encoding, the Annex I and K (RS=1) are not supported.
– Error resilience tools
MPEG-4 resync Marker & data-partitioning with RVLC
– Bit-rate control (CBR: Constant Bit Rate & VBR: Variable Bit Rate)
– CIR (Cyclic Intra Refresh)/AIR (Adaptive Intra Refresh)

• Pre/post rotation/mirroring
– 8 rotation/mirroring modes for incoming image at encoder
– 8 rotation/mirroring modes for output image at decoder

• Programmability
– FIMV-MFC V1.0 embeds 16-bit DSP processor that is dedicated to processing bitstream and controlling the codec hardware.
– General purpose registers and interrupts for communication between a host processor and the video IP
• Performance

– Up to full-duplex VGA 30fps encoding/decoding
– Up to half-duplex 720x480 30fps (720x576 25fps) encoding/ decoding

Codec firmware

In addition to the boot code, a package of the firmware for driving the IP is required. Basically, the package for MPEG-4, H.263P3, H.264 and VC-1 codec is provided. You must write the firmware to a region of external memory and send information about the base address of the region by writing it to the CodeBufAddr register. At run-time, part of firmware is automatically loaded to the internal BIT memory.

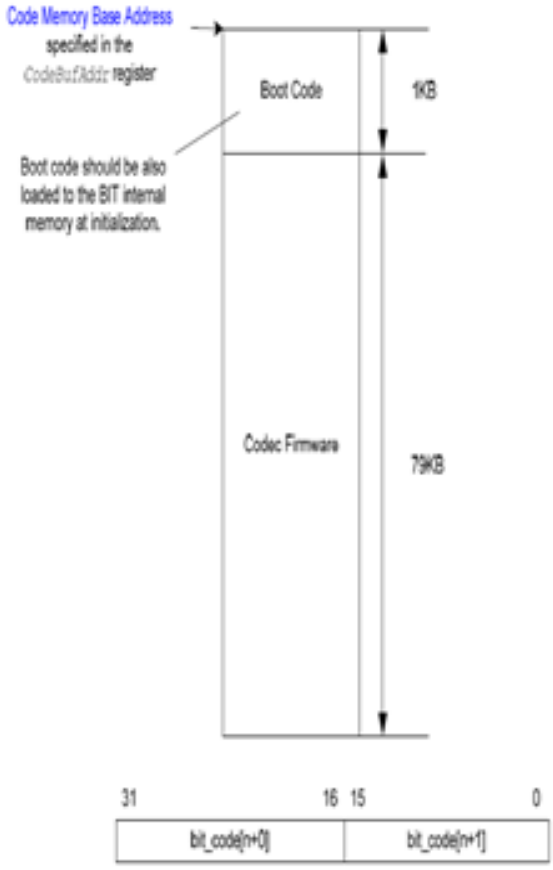

Fig.8.1-Codec Firmware

9. RUNNING THE CODEC

This section describes how the BIT processor controls the video codec and communicates with a host processor. The provided firmware can handle 8 processes simultaneously. Each process can have difference format –MPEG-4, H.263P3, H.264 or VC-1 - and codec process-encoding or decoding. For example, it is possible to handle 1 MPEG-4 encoding process, 1 H.264 encoding process, 1 H.263P3 decoding process and 1 VC-1 decoding process simultaneously. You can encode image and/or decode bitstream as following.

• Create processes: You can create and configure processes.

• Running processes: At a proper time instance, you can run a specific process. The proper time instance means when the codec is in idle state and image to be encoded or bitstream to be decoded is ready in the external memory.

• Quit processes: You can quit a specific process.

The MPEG-4/H.263P3 transform/quantization module supports only the method1 quantization mode for MPEG-4 bitstream. It can process the AIC (Advanced Intra Coding) and the modified quantization mode for H.263P3 bitstream. To process one macro block in the MPEG-4/H.263P3, FIMV-MFC V1.0 transform/quantization requires about 500 cycles.

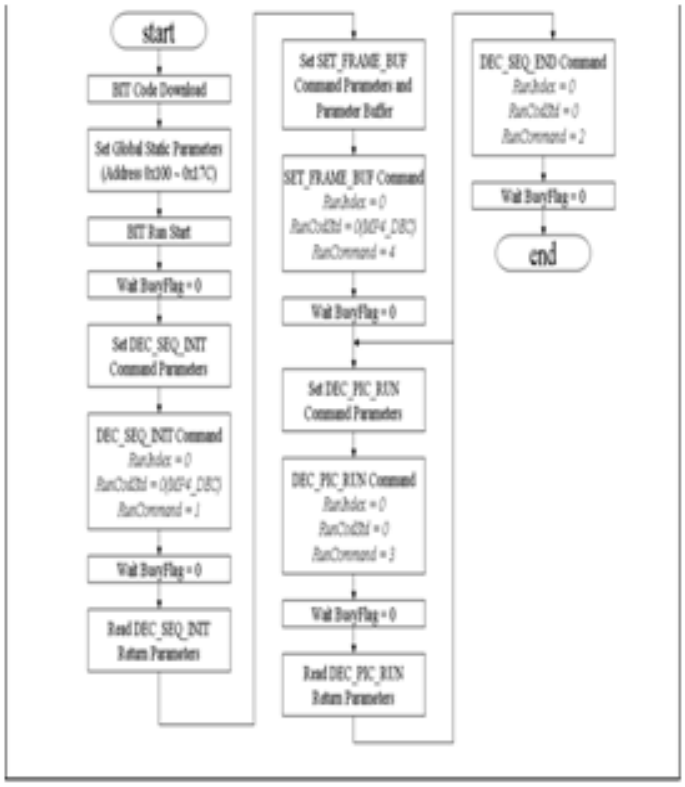

Fig.9.1-Block diagram of ARM MP core decoder

10. CONCLUSION

From the analysis and experimental results of MFC application, it can be seen that there is a vast variation in the performance of video streams for different configurations. The advantage of the usage of the shared memory is exemplified when we compare cases with each other. The system runs at a faster speed as well as the processor overhead is less in case 2. The analysis can be extended to other aspects where the use of shared buffer gives similar behavior. The advantage of use of video decoder on a hardware core with different video decoders using the scaled video device driver (Frame Buffer and sending the data in streamlined way to the all the frame buffers (fb0,fb1,fb2,fb3) can be exemplified by comparing the results on the LCD . The proposed design methodology can also be extended to other electronic devices like DVD players, recorders, camcorders, live camera automation for security applications.

11. REFERENCES

[1]. Gstreamer.freedesktop.org
[2]. Luo Rui , “Research and Implementation of Multimedia Player based on embedded Blackfin platform : [Master thesis] ,” Chengdu: University of Electronic Science and Technology, 2009.
[3].Samsung Electronics Co. Ltd, “S3C6410X USER'S MANUAL,” February 2008, pp.50-58.
[4]. "System Description Blu-ray Disc Read-Only Format part 3 Audio Visual Basic Specifications", Blu-ray Disk Association
[5]."DVD Specifications for Read-Only Disc / Part 3: Video Specifications", Version 1.9, DVD Forum, WG1/TG1-HD, 2004
[6].G. Ezer, "High-Performance Multicore Video Decoder Technology Preview", Fall Microprocessor Forum, 2005
[7]. M. Carchia, A. Wang, "Rapid Application Optimization Using Configurable Processor Extensions", Hot Chips 2001.

