International Journal of Computers, Communications & Control
Vol. I (2006), No. 4, pp. 85-91

Realization of Embedded Multimedia System Based On Dual-Core Processor OMAP5910

Peng Li, Yu Lu, Shen Li, Hongxing Wei

Abstract: This paper focuses on the realization of a complete embedded system using the dual-core processor OMAP5910. Detailed description of how to compose the hardware system is presented with a description of the software system on our platform. Tasks communication between the two cores is realized using the DSP driver. The system bootloader and the DSP bootloader are described in detail. The implementation of the MPEG-4 video decoder has been realized on the presented system. Higher speed can be achieved and less power is needed for MPEG-4 video processing on the dual-core platform. This dual-core system can be applied to 3G wireless communication, robot control and vision systems.

Keywords: Embedded System, MPEG-4 decoder, ARM, Omap5910, Multimedia Platform, Multimedia application, Operating system.

1 Introduction

With the growth of the 2.5/3G wireless markets, full-featured multimedia services are required by the wireless applications. More and more powerful processors are used to achieve the requirement. Dual-core architecture processors meet the rapid processing needs of next-generation embedded devices. Owing to the urgent need of the dual-core processor application, this paper presents a platform using the dual-core processor OMAP5910 and applications on this platform.

The OMAP5910 chip is a highly integrated hardware and software platform. It integrates a TMS320 C55xDSP core with a TI-enhanced ARM925 core on a single chip for the optimal combination of high performance with low power consumption. This unique architecture makes it ideal for audio and video processing in multimedia applications [1], [2].

The remaining part of this paper will be compiled as follows. In section 2, the composing of the hardware system is described and in section 3, the software system architecture is presented. In section 4, a detailed description of the system bootloader and DSP bootloader is given. Since video processing is one of the most important applications in multimedia systems, the implementation of the video decoding is described at the end of the paper.

2 Hardware System

OMAP5910 contains numerous interfaces for connecting to peripherals or external devices [3]. The complete Embedded Multimedia System we designed is illustrated as Figure 1.

As illustrated by Figure 1, 32M SDRAM and 16M NOR FLASH are used to compose the minimum system. The NOR FLASH is used to store the bootloader and the operating system. Another 256K SRAM is applied. A jumper is used to control the boot overlay mode, in which the SRAM is mapped to bank 0 and the NOR FLASH is mapped to bank 3.

Besides the minimum system, other devices of the platform are as follows: a camera for video capturing, a TFT color LCD for displaying, an audio codec for audio input and output, an Ethernet controller for operating system downloading and data transmitting, a PS/2 mouse and a keyboard as human device interfaces. The USB host/function interfaces and SD/MMC card interface are included in OMAP5910. They are used for the data storage in our system.

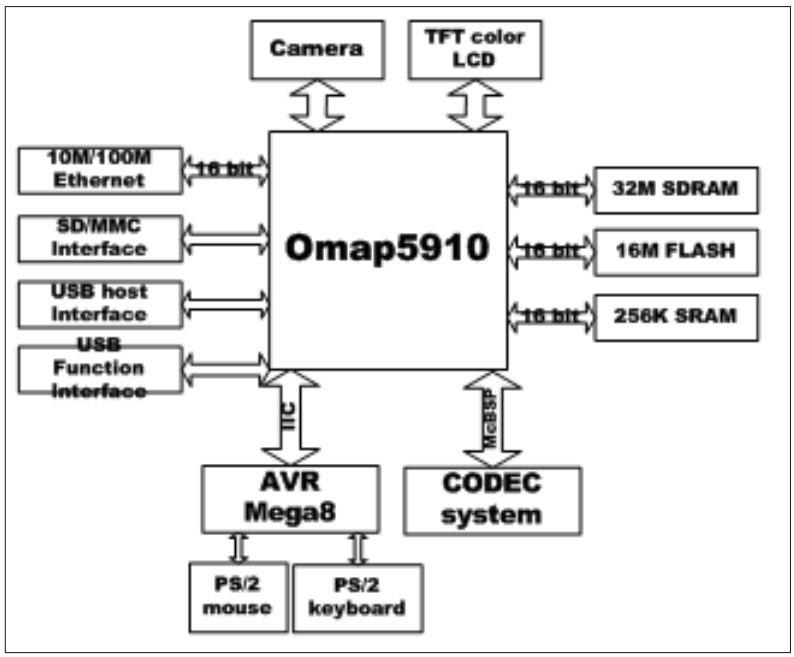

Figure 1: Hardware System

Due to the integration of the camera interface for CMOS sensors on OMAP5910, we choose the OV7648 for video capturing. The OV7648 is a sensor-on-board camera and lens module designed for mobile applications. The control interface of the sensor is the SCCB (OmniVision Serial Camera Control Bus), a 3-wire serial bus. Three MPU I/Os are used to realize the SCCB protocol. OV7648 output format is YCrCb 4:2:2 or RGB, and the YCrCb format is chosen in the system.

The color LCD controller on OMAP5910 supports a direct connection to the LCD panel and an 18bit TFT LCD is chosen for displaying. A high priority, dedicated DMA channel has been implemented in the system specifically for the LCD Controller. This dedicated DMA-LCD channel ensures minimum latency in real-time LCD operations.

Ethernet makes high-speed data exchanging, a very important function in multimedia systems, available between PC and the platform. It also provides an efficient way for system debugging. Here we adopt a fast Ethernet controller AX88796. The AX88796 is a high performance and highly integrated local CPU bus Ethernet Controller with embedded 10/100Mbps PHY/Transceiver and 8K*16 bit SRAM. It supports both 8 bit and 16 bit local CPU interfaces including MCS-51 series, 80186 series, MC68K series CPU and ISA bus. The 80186 16bit mode is chosen to meet the 16bits data bus of OMAP5910.

Because OMAP5910 does not have sufficient general I/Os, an AVR microcontroller is used to interface the PS/2 mouse and the keyboard. IIC bus is used to realize communicate between the AVR microcontroller and the OMAP5910. It is a good solution for the shortage of the GPIO in the system.

Since three McBSPs (Multi-channel Buffered Serial Ports) are integrated on the OMAP5910, an audio codec is connected to one of them directly. TLV320AIC23B, a high-performance stereo audio codec with highly integrated analog functionality, is selected to compose our codec system. The TLV320AIC23B supports glueless interface to the TI McBSP. We use the IIC bus to implement its control interface.

Various multimedia applications can be realized using this hardware platform on which we have implemented a MPEG4 video decoder.

3 Software System

We choose the DAS U-BOOT as the bootloader for the system, and the Linux 2.4.21 as the kernel. The software tools we used to develop programs are Code Composer Studio (CCS) and the GNU Tool Chain. CCS is used to develop DSP programs, while the GNU Tool Chain is used to compile the bootloader and the kernel. Programs on the MPU and the DSP are developed separately. In our system, the ARM RISC is used for handing control code, such as user interface, OS and OS application, while the DSP is used for the real-time signal-processing. Both the system bootloader and the kernel run on the MPU, and the DSP is registered as a character device in the kernel.

Applications that run on the MPU can not access or make use of the DSP directly, so they must use the DSP driver to control or communicate with DSP core. The DSP driver in the kernel provides user applications with the capability of controlling and the interfaces to communicate with DSP. To communicate or make use of DSP, applications have to send read, write or I/O control request to kernel, then kernel would deliver such request to DSP driver. Subsequently, DSP driver set the corresponding control registers and send the control code or application date to the DSP.

There are three ways we used to communicate between the MPU and the DSP. The first one is to use the MPUI port. It allows access to the full memory space (16M bytes) of the DSP, and it is the only way for the MPU and the system DMA to access the I/O spaces of the DSP. Since configuration and data registers for all the peripherals of the DSP reside in the DSP subsystem I/O space, we use this way to set the corresponding peripherals registers in the DSP. The second way is to use the MPU MMU and the DSP MMU, through which shared access to system memory can be realized between the two cores. The MPU MMU performs virtual-to-physical address translations and accesses permission checks for access to the system memory. The DSP MMU is controlled by the MPU. When the DSP MMU is on, the DSP MMU translates addresses from the DSP (virtual address) to addresses mapped by the traffic controller. The DSP MMU is used when the DSP software accesses external memory and we adopt the DSP MMU to map the external memory space of the DSP on the external SDRAM. We also utilize the MPU MMU to map an appropriate system addressing space on the external SDRAM. Then both the MPU and the DSP can access this shared memory regions. The third way is to use the mailbox-interrupt mechanism. This mechanism provides a very flexible software protocol between the two cores. We use it to create handshaking interrupts which will properly synchronize the MPU and DSP operations. The software architecture is shown in Figure 2.

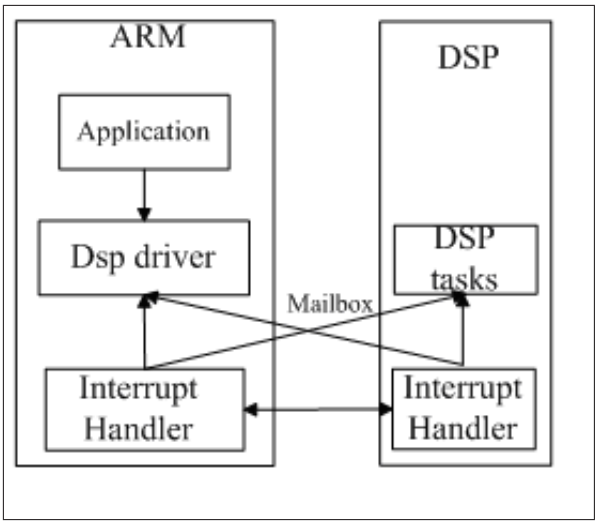

Figure 2: Software Architecture

When certain user application wants to utilize DSP to accomplish a specific calculating task, it must

send a request to the DSP driver. The DSP driver would write this request code and application data to the Mailbox. So the DSP core can get to know what the user application wants and where is the storage of the raw data. Then DSP would be interrupted, and both the control code and the application data address could be obtained during this Mailbox interrupt. With these data, DSP starts to do the corresponding calculation as the application asks. After finishing the application task, DSP returns a response to the MPU by writing to the Mailbox. DSP driver would process this Mailbox interrupt on the MPU side, and fetch the result of the application task.

4 System bootloader and DSP bootloader

Boot process is very important to both the MPU and the DSP. Each of the two cores needs their own bootloader. The system (MPU) bootloader is adopted to perform a simple initialization and to load the kernel to the system memory. U-BOOT is a bootloader for embedded system based on PowerPC and ARM processors. It is a full featured bootloader that is small and also easy to build and use. One of the most useful features of U-BOOT is the supporting of the TFTP downloading. The U-BOOT 1.1.1 supports the ARM925t architecture and the OMAP5910 processor. We need to set appropriate configurations and write drivers for various peripherals used during the booting process. The Ethernet controller driver enables the TFTP boot feature of U-BOOT and such TFTP boot function greatly facilitates the debugging process of the kernel. Much higher speed could be achieved when loading kernel to the system RAM, compared with the using of serial or parallel port. The loading process can be fully automatic by the U-BOOT when power on. The MPU is the master in the system and is responsible for setting up and bringing the DSP out of reset. Appropriate boot mode of the DSP must be set by MPU before releasing the DSP from reset.

When the DSP core is taken out of reset, DSP bootloader will be executed. The bootloader of the DSP resides in the DSP subsystem PDROM. It performs some initialization of the DSP resources before loading code. After the initialization, the bootloader branches to the starting point address specified by the selected boot mode. There are four boot modes: direct boot, external memory boot, DSP idle boot, internal memory boot. The internal memory boot mode is selected in our system, because it is more convenient compared with other modes.

To adopt the internal boot mode, we have to load code and data sections into DSP subsystem internal memory. The MPU core or system DMA can be used to accomplish this task. We utilize MPU core with the MPU interface (MPUI) to perform the loading. MPUI allows the MPU to communicate with the DSP and its peripherals. The endian mode should be noticed during the data loading process. The MPU (TI925T) operates in little endian mode while the DSP operates in big endian mode. Therefore, data have to be converted into big endian format when loaded into the DSP's memory. There are swapping buffers between the DSP and the MPUI, and the word or byte swapping can be programmed. After loading data into DSP internal memory, the MPU core can call the DSP subsystem out of reset, then the DSP bootloader begins to execute. The DSP bootloader transfers execution to the appropriate byte address, and the loaded application starts running. It should be noticed that the output format of CCS is DSP COFF format, and it cannot be used directly in ARM side. So we utilize the HEX55 utility to convert the COFF format file into HEX format file. The hex file could be parsed to get the DSP code and its running address.

5 System application

MPEG video processing is a very important application for multimedia systems. Due to the high-speed requirement of the computation, powerful processors are needed to process these data. Our dual-core system provides a better solution for video processing compared with normal RISC [6] or DSP

systems.

Handling of the control code and the real-time signal processing are implemented by the MPU and the DSP separately. The ARM RISC is well suited for handling control code, such as user interface, OS and OS applications. On the other hand, the DSP side is better suited for real-time data processing. Appropriate division of the work makes the dual-core system get a better performance.

The C55x DSP is especially suited for multimedia signal processing. Three hardware accelerators are included along with the C55x DSP core to enhance the multimedia processing capability [4]. They are discrete cosine transform (DCT) and its inverse IDCT accelerator, motion estimation accelerator and pixel interpolation accelerator. DCT/IDCT, motion estimation, and pixel interpolation are common tasks to all industry video-imaging standards. They are also the most time-consuming tasks in video processing. These C55x hardware accelerator modules assist the DSP core in implementing algorithms that are commonly used in video compression applications such as MPEG4 encoders/decoders. These accelerators enable implementation of such algorithms using fewer DSP instruction cycles and dissipating less power than the DSP core is operating alone.

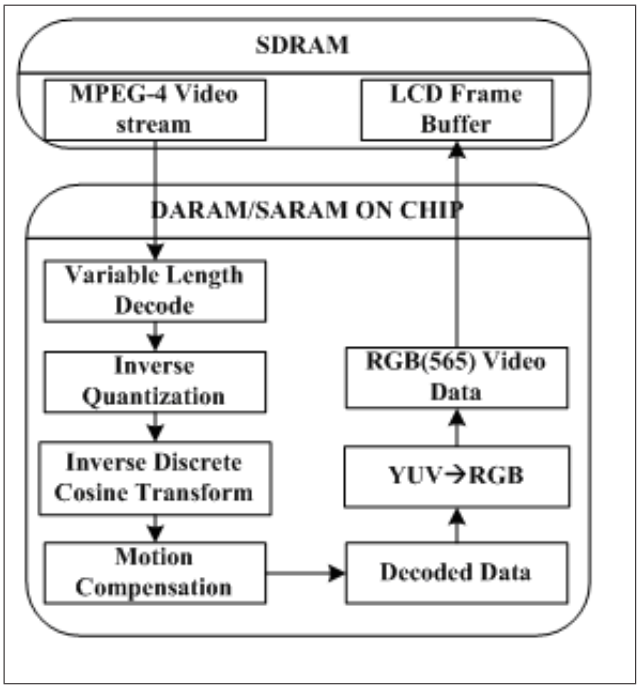

Figure 3: The decoding procedure on the DSP

The hardware accelerators are utilized via functions from the TMS320C55x Image/Video Processing Library available from Texas Instruments [5]. IMGLIB is an optimized image/video processing functions library for C programmers using TMS320C55x devices. It includes many C-callable, assembly-optimized, general-purpose image/video processing routines. These routines are typically used in computationally intensive real-time applications where optimal execution speed is critical. By using these routines, we achieve execution speeds considerably faster than equivalent code written in standard ANSI C language.

The hardware accelerator and the IMGLIB are utilized to implement a MPGE-4 video decoder on our multimedia system. The decoding procedure on the DSP core is illustrated as Figure 3.

The MPEG-4 bit-streams are stored in shared system memory. Because the DSP accesses the internal DARAM and SARAM at much higher speed, the decoding process is completed in its own local memory. By the help of the TMS320C55x Image/Video Processing Library, we get the decoded data in the local memory. Because the decoded data are in YUV 4:2:0 formats, post-processing is needed to convert the

	ARM9E	C5510	Units
MPEG-4/H.263 decoding QCIF at 15 fps	16	9.6	Mcycles/s
MPEG-4/H.263 encoding QCIF at 15 fps	90	33	Mcycles/s

Figure 4: Results of comparative benchmarking

decoded data into RGB format. Having converted the data into16 bits RGB (565) format, we send it to the LCD Frame buffer for displaying [7].

Compared with a RISC processor ARM9, the performance of the decoding process is greatly improved. Results of comparative benchmarking by Pace Soft Silicon are shown in Figure4.

These benchmarks demonstrate that multimedia tasks require three times as many cycles and two times as much power to execute on a latest-generation RISC processor as they do on a C55x DSP [4].

6 Summary and Conclusions

This embedded multimedia system is successfully designed and implemented. Various multimedia applications can be realized on such platform. The software system presented herein provides a solution for communication between the dual-core. The successful implementation of the MPEG4 decoding procedure on this system proves that the higher speed can be achieved and the less power is needed.

With the growth of the 3G wireless markets, dual-core architecture systems will be widely used. The authors believe that the presented platform is suitable for many 3G wireless multimedia applications in the near future.

References

[1] James Song, Thomas Shepherd, Minh Chau, A low power open multimedia application platform for 3G wireless, *Proceedings of SOC Conference, 2003, IEEE International*, pp. 377-380, Sept.2003.

[2] JCheng-Nan Chiu, Chien-Tang Tseng, and Chun-Jen Tsai, Tightly-coupled MPEG-4 video encoder framework on asymmetric dual-core platforms, *ISCAS 2005, IEEE International*, pp. 2132-2135, May 2005.

[3] Texas Instruments, OMAP5910 Dual-Core Processor Technical Reference Manual, *TI Technical Document SPRU602B,* Jan.2003.

[4] Texas Instruments, TMS320C55x Hardware Extensions for Image/Video Applications Programmer's Reference, *TI Technical Document SPRU098*, Feb.2002.

[5] Texas Instruments, TMS320C55x Image/Video Processing Library Programmer's Reference Preliminary, *TI Technical Document SPRU037C,* Jan.2004.

[6] Hae-Yong Kang, Kyung-Ah Jeong, Jung-Yang Bae, Young-Su Lee, Seung-Ho Lee, MPEG4 AVC/H.264 DECODER WITH SCALABLE BUS ARCHITECTURE AND DUAL MEMORY CONTROLLER, *Circuits and Systems, 2004. ISCAS '04. Proceedings of the 2004 International Symposium on*, Vol. 2, pp.II - 145-8, May. 2004

[7] Thanh Tran, OMAP 5910 Video Encoding and Decoding, *TI Application Report SPRA985*, Dec. 2003.

Peng Li, Yu Lu
Beijing University of Aeronautics and Astronautics
School of Automation Science and Electrical Engineering

Shen Li
Beijing University of Aeronautics and Astronautics
School of Software

Hongxing Wei
Beijing University of Aeronautics and Astronautics
School of Mechanical Engineering and Automation
No. 37 Xue Yuan Road, Haidian district 100083 Beijing P.R. China
lastmars@gmail.com

Received: November 5, 2006