A Versatile High-speed Image Processing System Based on DSP and CPLD*

LUO Yan-xing, XIE Mei
School of Electronics Engineering, University of Electronic Science and Technology of China  Chengdu  610054  China

Abstract  In this paper, we present an optimized design method for high-speed embedded image processing system using 32 bit floating-point Digital Signal Processor (DSP) and Complex Programmable Logic Device (CPLD). The DSP acts as the main processor of the system: executes digital image processing algorithms and operates other devices such as image sensor and CPLD. The CPLD is used to acquire images and achieve complex logic control of the whole system. Some key technologies are introduced to enhance the performance of our system. In particular, the use of DSP/BIOS tool to develop DSP applications makes our program run much more efficiently. As a result, this system can provide an excellent computing platform not only for executing complex image processing algorithms, but also for other digital signal processing or multi-channel data collection by choosing different sensors or Analog-to-Digital (A/D) converters.

Key words  image processing; digital signal processor; complex programmable logic device; DSP/BIOS

Along with the development of digital signal processing technology, there is an increasing use of image and video processing in embedded multimedia applications which pose very high requirements in terms of computational power and memory bandwidth on processors. There are several technologies that allow the implementation of high-speed digital image processing. For example, Field Programmable Gates Array (FPGA) can offer complete hardware customization while implementing various digital signal processing applications since it can be reconfigured in hardware. However, design with FPGA for applications Digital Signal Processor (DSP) is difficult mainly due to lack of a C-based design flow[1]. In contrast, DSP seems to be a good solution for digital image processing applications which use C-based design flow frequently. This technology would be more mature and economical.

This paper introduces a small portable image processing system which works independently without computer or other external devices. By choosing high performance DSP, Complex Programmable Logic Device (CPLD) and CMOS image sensor, the system is advanced for its precision, high levels of integration and low power consumption. Especially, the choice of high speed CPLD can largely simplify the design and makes the hardware easy to interface with different sensors and Analog-to-Digital (A/D) converters.

This paper is organized as follows. Section 1 presents the hardware structure of the system and particularly introduces the use of CPLD to acquire images. Section 2 briefly discusses the applications in iris recognition system and the design method of DSP/BIOS applications. Some conclusions are given in section 3.

1 Hardware Structure Design

Fig.1 shows the hardware structure of our system. This system can be divided into 3 basic modules: image acquisition, image processing and input/output interface. The image acquisition part consists of a CMOS image sensor (OV7141) and a 512 KB Static RAM (SRAM). The OV7141 is used to capture 640×480 pixels images. Its output data is automatically stored in SRAM by using CPLD to assign addresses. The image processing part is the hardcore of the system. We choose both high-speed DSP TMS320C6713b and large external memories to fulfill this work: an 8 MB Synchronous Dynamic RAM (SDRAM) acts as the temporary code/data memory and a 2 MB FLASH memory (FLASH) is used to store program code and initialization data. Because the system is expected to be an integrated system, the I/O interface part involves a keyboard which is managed

Received 2005-12-26
* Supported by the National Natural Science Foundation of China (No.60472046)
by a Keyboard Control chip (KBC) and a LCD module to display processing results. In addition, in order to transmit data with a computer, we design a USB2.0 port, which allows the computer access our system resources freely.

Before the image acquisition, the CPLD sends a HOLD signal to possess the External Memory InterFace (EMIF) bus. Then the DSP releases the EMIF bus (high Z) and feeds back a HOLDA signal. The CPLD will begin its data storing operation until it detects this signal. When the HERF signal is active, the CPLD counts the PCLK, samples the data at each two PCLK rising edge and makes the ADDRESS signal add 1 to store the Y vector of the YUV data. The control signal of the SRAM is generated by the CPLD too. The Chip Select (CE) signal is generated by the HERF. The Write Enable (WE) signal is generated by the PCLK. When a frame has been stored completely, the CPLD sets the HOLD signal and generates an interrupt signal INT1 to begin image processing work. With working at 24 MHz, our system only needs 30 ms to acquire a 640×480 image. The hardware interface of the image acquisition part is shown in Fig.2 and the timing simulation result is shown in Fig.3.

1.1 Image Acquisition Design Approach

To achieve high processing precision, we choose Omni Vision’s OV7141 to acquire images. OV7141 is a black and white CMOS image sensor which can incorporate a 640×480 image array with complete user control over image quality, formatting and output data transfer. All required image processing functions can be programmed through the Serial Camera Control Bus (SCCB). In addition, OV7141 uses advanced sensor technology to improve image quality by reducing common lighting/electrical sources of image contamination such as fixed pattern noise, smearing and blooming to produce a clean and fully stable image[4].

Because our system is expected to be versatile with the ability to suit for various sensors, which require different control signals or power supply, we choose Altera’s EPM7128s CPLD to store the output data of OV7141. EPM7128s has 128 macrocells and 2500 usable gates with 6 ns pin-to-pin logic delays[5]. It also provides the ability to interface with both 3.3 V and 5 V devices by using external pull-up resistors.

In the process of system initialization, the DSP writes the SCCB registers to set the work parameter of OV7141 including exposure control, gamma, white balance as well as work speed, image size and output data format. Each frame is output in 8-bit 4:2:2 YUV format. In Fig.3, the PCLK is a 24 MHz pixel clock, the HERF indicates the valid data, the DATA are 8-bit output data signals latched on the PCLK rising edge, and the VSYNC is the vertical sync signal, which indicates the start of a frame.
deliver up to 1350 million floating-point operations per second and 1800 million instructions per second [6].

The main problem to limit the performance of our system is the speed of image processing. There are several approaches to enhance the data processing ability of the DSP. First, since the execution of program mainly in SDRAM would waste much time, we divide the DSP’s 256KB L2 memory into 64KB L2 cache and 192 KB additional L2 mapped RAM to provide a buffer between external memory and internal memory [7]. Second, we choose 32 bit external memory. Thus the access of 32/16/8 bit data can be done in one read/write period. And last, we use DSP/BIOS tool to develop DSP applications. The details will be introduced in section 2.

1.3 Input/Output Interface

In order to save the EMIF resource and improve the real-time performance, we choose an I2C KBC Zlg7290 to manage the 4×4 keyboard. When a key is put down, the KBC scans the keyboard and generates an interrupt signal (INT0). Then the DSP reads the key value from the I2C bus and determines which key to be put down.

To display the processing results, we use a 128×64 LCD module with T6963c controller [8]. Because the T6963c’s read/write period is much slower than the DSP, we must add many idle instructions in each read/write operation to match the speed of LCD module.

2 Application to Iris Recognition System

2.1 Workflow of Iris Recognition System

This hardware has been successfully applied to the iris recognition system. The work flow of the iris recognition system is shown in Fig.4. There are two work modes defined for this system: the register mode and the match mode. Both modes can be set by the keyboard. If we choose the register mode, we need to input user ID and burn both iris texture feature and ID to the FLASH. If we select the match mode, the system will compare the iris texture feature we just abstract with database in the FLASH to determine whether they belong to the same person or not. The processing result and corresponding ID are displayed on LCD.

After resetting the system, the DSP boots from the FLASH and initializes its internal modules and external devices (OV7141 and LCD). Then the system goes into idle mode. When one gets close to the camera lens, the DSP writes a predefined register (0xB0000001) of the CPLD to wake up the image sensor from the sleep mode. Then the CPLD starts to acquire iris images. When a frame is stored in the SRAM, the 2-Dimensional-to-2-Dimensional (2D-2D) Enhanced Direct-Memory-Access (EDMA) transfer is used to read the data from the SRAM, then the DSP begins the pre-processing work including iris localization, normalization and image enhancement [9], which can cope with the outer blurred boundary and spend less time on executing. Lastly, the quality of our picture is evaluated. If the image definition and iris integrity can meet the standard we predefined, the system will go on image processing work, including iris feature point extraction and signature encoding. Otherwise, the system requests the user to re-capture another iris image.

2.2 Using DSP/BIOS Tool to Design DSP Applications

To design a DSP application with complex constitution, the use of custom C/C++ language will be hard to trace and analyze. So TI develops a DSP/BIOS tool to solve this problem. The DSP/BIOS is a kernel including memory management, tasks management, an API for using real-time library services, easy-to-use tools for DSP configuration, and real-time program tracing and analysis.

There are four types of threads defined in DSP/BIOS: HardWare Interrupt (HWI), SoftWare Interrupt (SWI) and background thread (IDL). HWI is triggered in response to external asynchronous events and has the highest priority in a DSP/BIOS application. While HWI is triggered by a hardware interrupt, software interrupts are triggered by calling SWI functions from the program. SWI provides additional priority levels between HWI and TSK. It allows HWI to defer less critical processing to a lower-priority thread and to minimize the time the CPU spends inside an ISR, where other HWIs may be disabled. TSK is the basic cell of DSP/BIOS applications. It has higher priority than the IDL and
lower priority than SWI. Each TSK object is always in one of four possible states of execution: running, ready, blocked, and terminated. We can take control of these states by calling API functions or using synchronizations (SEM) or mailboxes (MBX). IDL is the background thread of DSP/BIOS, which runs continuously when no hardware interrupt service routines or software interrupts are running\footnote{10}.

A DSP/BIOS application always starts at the C environment entry point c_int00. Then DSP/BIOS calls BIOS_init to initialize the DSP/BIOS modules. After all DSP/BIOS modules have completed their initialization procedures, the main routine is called. All functions to initialize the devices are executed in main routine. Then DSP/BIOS calls BIOS_start to start DSP/BIOS. All HWIs, SWIs and TSKs are executed and blocked to wait for semaphores or mailboxes at this moment. By calling IDL_loop, the boot routine falls into the DSP/BIOS idle loop forever. At this point, hardware and software interrupts can occur and preempt idle execution.

Our DSP/BIOS application consists of two parts: the control part and the algorithm part. The control part is triggered by the external hardware interrupt INT0, which is generated by the KBC. Its ISR reads the key value form the I2C bus and calls SWI_post() to trigger a key verify SWI to verify which key is put down and then turn to the corresponding operation. The algorithm part is triggered by INT1 which indicates the accomplishment of the image acquisition. It includes 6 tasks: preprocess, evaluate, process, match, register, and display. In these tasks, the display TSK has the highest priority and is triggered by sending messages to a MBX. Other TSKs have the same priority and triggered by counting SEMs, as shown in Fig.5.

![Fig.5 DSP/BIOS block diagram](image)

### 3 Conclusions

This paper has shown a versatile platform for executing complex image processing algorithms using DSP and CPLD. In our laboratory, the hardware has been successfully applied to both iris recognition system and fingerprint recognition system. The iris recognition system is shown in Fig.6. It is hoped this system can be suitable for high-speed motive object capture in the future.

![Fig.6 The photo of iris recognition system](image)

### References


