Third Prize

Multi-Functional Digital Albums Based on the Nios II Processor

Institution: **Information Science Institute, Beijing Jiaotong University**

Participants: **Cheng Hong, Rui Deng, Yongxin Ye**

Instructor: **Xiaoming Ding**

Design Introduction

As digital cameras and high-pixel mobile phone cameras have become more popular, users save their photos in a variety of storage devices (e.g., PC hard disks, semiconductor storage devices, and CD-ROMs). Today, most printed photos are from digital cameras or mobile phone cameras. As users create more and more digital photos, they will spend a lot of money developing these photos, and traditional photo frames and albums do not store these photos well. Additionally, directly editing photos on a portable hardware platform independent of a PC would be very convenient when traveling.

Today, the main functions of digital albums on the market are photo storage and playing, which wastes valuable resources. It would be significant to integrate most functional modules into one system that fully utilizes the processor, implements more user operations, and meets more requirements.

Our design can be used together with a digital camera. It allows users to edit photos stored in the camera directly and then transmit the photos to the home or office through a network interface, relieving pressure on storage devices such as secure digital (SD) cards. A digital album that features digital photo storage, playing, viewing, editions, and network transmission will satisfy users' requirements in today's information society.

Our design provides users with a complete do-it-yourself (DIY) space that helps them create personalized photos. When the communication interface between the album and the mobile phone is developed in the future, users can send DIY photos directly to their friends' mobile phones.

Application Scope and Targeted Users

The concept of a digital photo is well received in society. Our design plays a role in any field in which digital photos exist. With the unique editions, receiving, and transmission functions of our design, travelers do not need to worry about mass storage space for their photos. The user can use this convienent, portable design at home or while traveling; our design can be widely applied to all groups.

Nios II Processor Design Advantages

Using the Nios® II processor provides the following advantages:

- *Configurability*—Compared with traditional processors, the Nios II processor features configurability to facilitate users' designs. We can customize the system according to our demands. Because Altera provides an abundant intellectual property (IP) core library, we can implement our design ideas quickly, shortening the development period. Additionally, the multi-functional technology cuts costs significantly and is more flexible.

- *Integrated development environment (IDE)*—The complete Altera®-provided IDE, from the Quartus® II software to SOPC Builder and to the Nios II Embedded Design Suite (EDS), creates all the hardware and software conditions for our design. Based on Altera's FPGAs, we can complete the system configuration and implementation with higher performance and stability.

- *Custom instructions and peripherals*—The Nios II soft-core processor can integrate 256 custom instructions and peripherals to accelerate system operation. The Nios II software development environment provides a standard program interface to facilitate program migration.

- *Unique C-to-Hardware Acceleration (C2H) Compiler functions*—The C2H Compiler has played a great role in the time since its launch. With this application, we do not need to worry that the complex algorithms in the Nios II EDS environment will influence the system performance. Instead, we can accelerate algorithm bottlenecks with the C2H Compiler to easily speed system operation.

Function Description

This design uses the Development and Education (DE1) multimedia development board to design a multi-functional digital product—a multi-functional digital album. The album supports all basic functions of its kind on the market while adding some new functions to meet the requirements of different user groups. This product contains the following functions:

- *Digital photo storage*—It is necessary for a digital album to have a certain amount of photo storage. Because the DE1 board has an SD card interface, we only need to load the SD card's data, demand, and address line to the Avalon® bus and the Nios II processor can control the SD card data reading and writing.

- *Digital photo viewing and playing*—It is a basic requirement for a digital album to allow users to view their photos at any time. With the µC/FS file system, the album can conveniently notify the Nios II CPU of photo files in the attached storage media, and users can choose any photo to view and play.

- *Special music effect*—Multimedia is popular in daily life. Most of Altera's FPGA development boards can implement multimedia playing. Considering the pressure on the storage media, we chose compressed G.729 code streams as the music format and embedded a laboratory decoding algorithm into our digital album so that users can listen to beautiful music while viewing photos, allowing it to be a "voice album."

- *Digital photo and audio file management*—After migrating the file system, we can easily group and archive files for convenient operation.

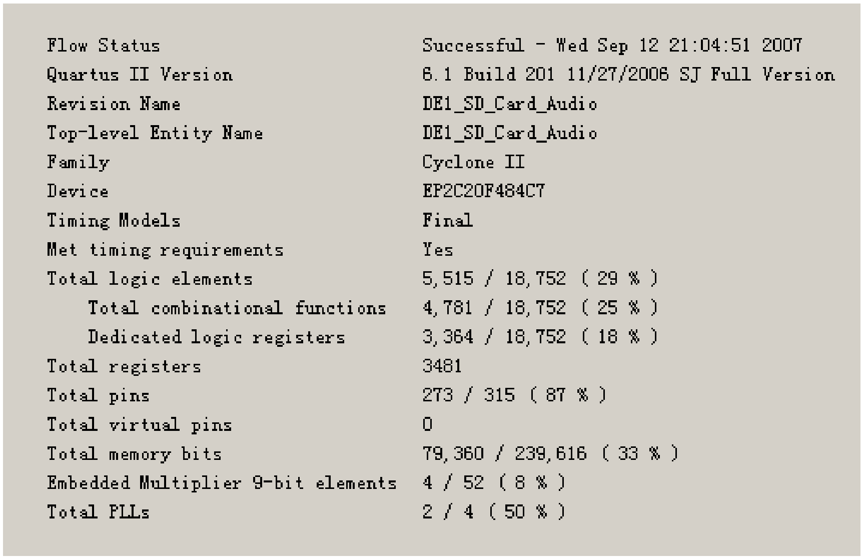
- *Digital photo editions and processing effects*—With an embedded platform we can provide a photo processing effect similar to that on a PC by using software like Adobe PhotoShop.
- *Photo format compression and decompression*—Because photos are usually stored in JPEG format, our multi-functional digital album must meet that requirement. Therefore, we embedded a JPEG decoder module into the Nios II processor, conducted all processing with decoded RGB data, and then compressed the photos into JPEG format for network transmission.
- *Digital photo perfection*—Traditional photos are usually stored in frames and can fade over time. In contrast, with a digital format we can add beautiful frames to photos for an amazing effect.
- *Digital photo network transmission*—The digital album design can receive JPEG code streams through the network and transfer them to the FPGA for processing. Then, the album compresses the photo data into JPEG code streams through the FPGA and transmits the code streams over the network, allowing the user to share photos with others. Implementing this function requires us to develop a network interface on the DE1 board, so we created a circuit board with a network function that connects with the FPGA through the general-purpose I/O (GPIO) pins. Therefore, our design can partner with digital cameras, allowing travelers to share photos with others anytime and anywhere.
- *User interface*—The DE1 board provides us with a PS/2 interface, so we can load a mouse and keyboard onto the Avalon bus. To respond simultaneously to the mouse and keyboard data, we modified the on-board circuit, allowing the Nios II processor to respond to the PS/2 peripheral any time and constructed the whole album.
- *Digital camera partner*—This design provides direct photo viewing and processing operations.
- *Digital watermark embedding and extraction*—When processing photos, users often hope the photos are not modified by others. The digital watermark embedding function of multi-functional digital album can be used as a photo copyright management tool without changing the effect of the photo.

When implementing the functional modules, we fully used the Nios II processor features to invoke the functional modules with the µC/OS-II embedded operation system.

Performance Parameters

Figure 1 shows the hardware resource utilization of the multi-functional digital album.

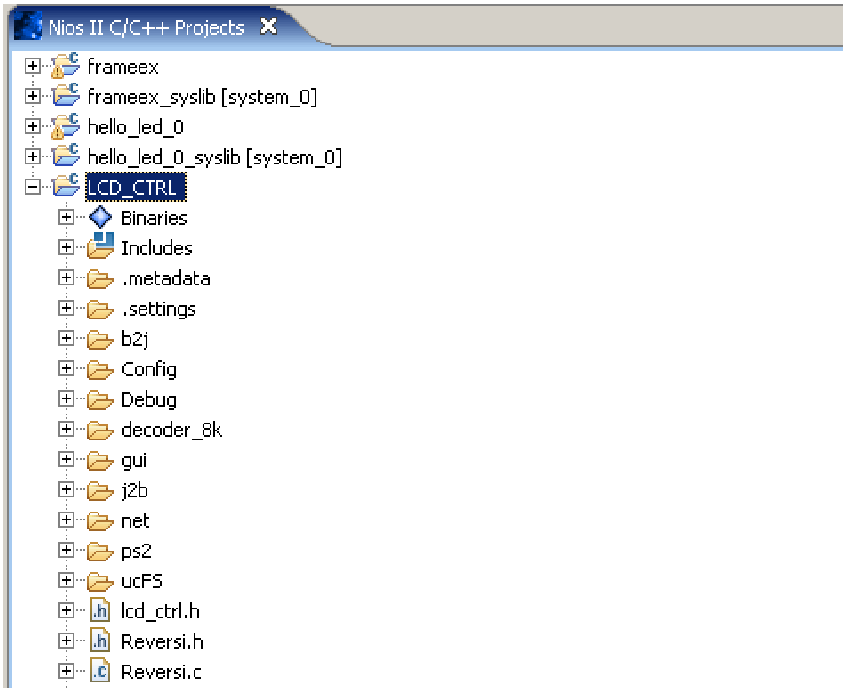
Figure 1. Hardware Resource Utilization



```
Flow Status                              Successful - Wed Sep 12 21:04:51 2007
Quartus II Version                       6.1 Build 201 11/27/2006 SJ Full Version
Revision Name                            DE1_SD_Card_Audio
Top-level Entity Name                    DE1_SD_Card_Audio
Family                                   Cyclone II
Device                                   EP2C20F484C7
Timing Models                            Final
Met timing requirements                  Yes
Total logic elements                     5,515 / 18,752 ( 29 % )
    Total combinational functions        4,781 / 18,752 ( 25 % )
    Dedicated logic registers            3,364 / 18,752 ( 18 % )
Total registers                          3481
Total pins                               273 / 315 ( 87 % )
Total virtual pins                       0
Total memory bits                        79,360 / 239,616 ( 33 % )
Embedded Multiplier 9-bit elements       4 / 52 ( 8 % )
Total PLLs                               2 / 4 ( 50 % )
```

Figure 2 shows the system software architecture.

Figure 2. Software Architecture



```
Nios II C/C++ Projects
+ frameex
+ frameex_syslib [system_0]
+ hello_led_0
+ hello_led_0_syslib [system_0]
- LCD_CTRL
    + Binaries
    + Includes
    + .metadata
    + .settings
    + b2j
    + Config
    + Debug
    + decoder_8k
    + gui
    + j2b
    + net
    + ps2
    + ucFS
    + lcd_ctrl.h
    + Reversi.h
    + Reversi.c
```

The system has the following specifications:

- System performance index parameters
  - Photo parameter: 320 x 240 JPEG format
  - LCD display parameters: 320 x 240 resolution, 5.7 inches
- System resource utilization (see Figure 1 on page 40 for details)
  - FPGA: Altera Cyclone® II EP2C20F484C7
  - Program operation space: 8-Mbyte SDRAM
  - Static storage: 512 Kbyte SRAM
  - User interface: key switch, LED, seven-segment nixie tube
  - VGA interface
  - Stereo audio codec
  - PS/2 mouse and keyboard interface
  - External network module control board for the GPIO interface and LCD display
  - Storage: Kingston SD card 2.0G/1.0G
- Expanded resources
  - SD card, four-line SD mode
  - Network controller circuit board
  - PS/2 one-for-two interface
  - LCD 65,000-color screen

Design Architecture

Figure 3 shows the hardware design and Figure 4 shows the system hardware and flow.

Figure 3. Hardware Design Block Diagram

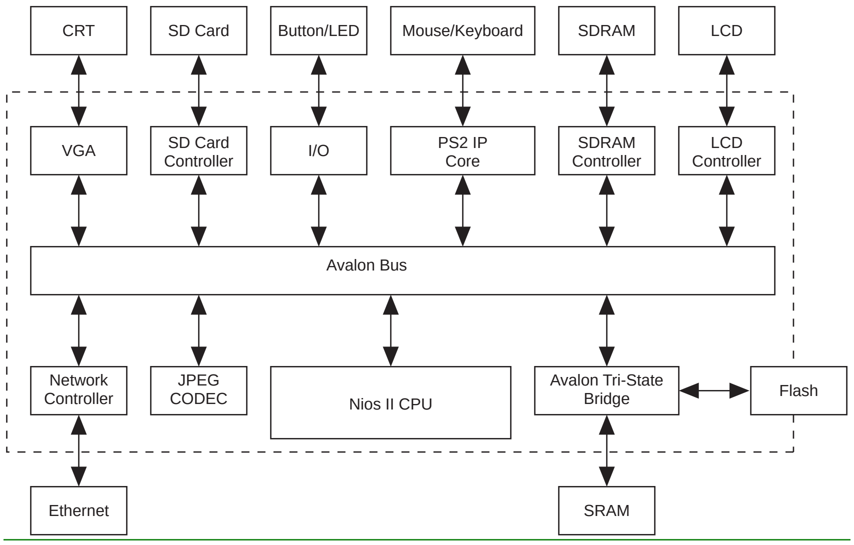
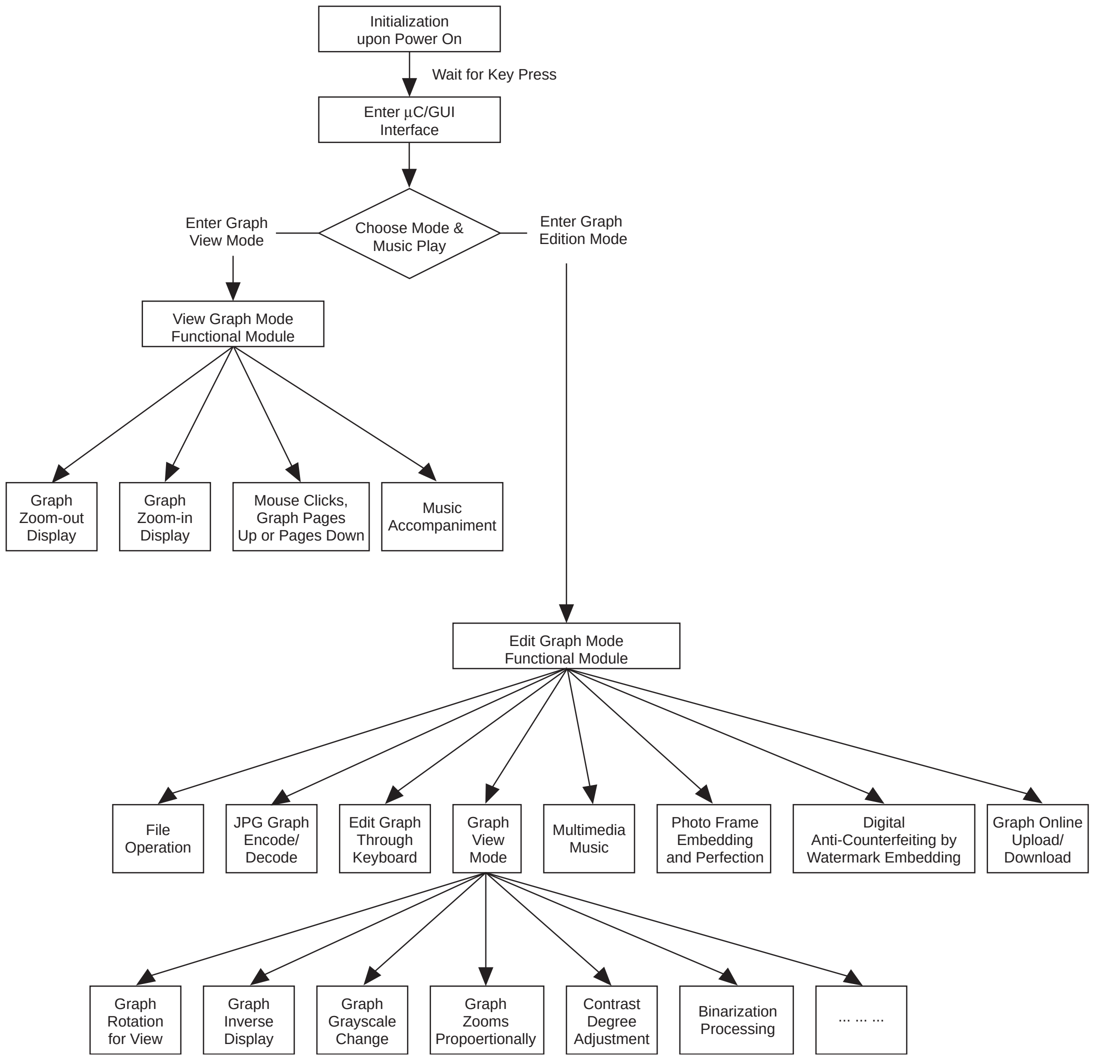


Figure 4. System Flow Chart



Design Methodology

This section describes the design methodology.

System Design

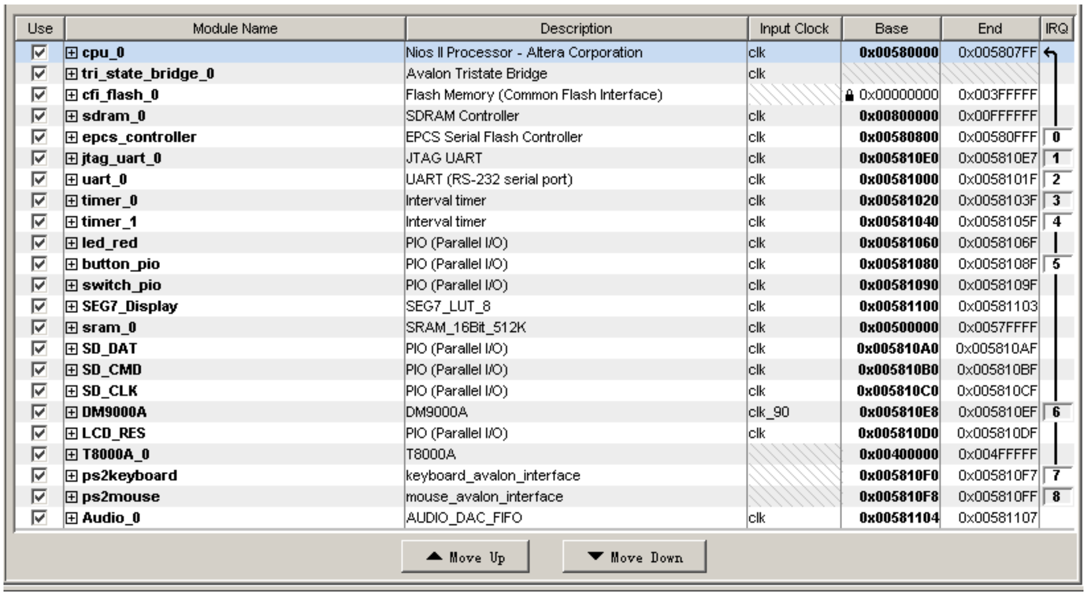
The system design contains these functional modules:

- SD card file system
- Keyboard input
- Audio digital-to-analog D/A converter (DAC)
- DM9000A network controller

- TCB8000A LCD controller
- VGA display
- JPEG codec
- Image processing
- Audio decoding

All functional modules are integrated into one system using SOPC Builder as shown in Figure 5.

Figure 5. SOPC Builder System Architecture



Use	Module Name	Description	Input Clock	Base	End	IRQ
✓	⊞ **cpu_0**	Nios II Processor - Altera Corporation	clk	**0x00580000**	0x005807FF	
✓	⊞ **tri_state_bridge_0**	Avalon Tristate Bridge	clk	▨	▨	
✓	⊞ **cfi_flash_0**	Flash Memory (Common Flash Interface)	▨	🔒 0x00000000	0x003FFFFF	
✓	⊞ **sdram_0**	SDRAM Controller	clk	**0x00800000**	0x00FFFFFF	
✓	⊞ **epcs_controller**	EPCS Serial Flash Controller	clk	**0x00580800**	0x00580FFF	0
✓	⊞ **jtag_uart_0**	JTAG UART	clk	**0x005810E0**	0x005810E7	1
✓	⊞ **uart_0**	UART (RS-232 serial port)	clk	**0x00581000**	0x0058101F	2
✓	⊞ **timer_0**	Interval timer	clk	**0x00581020**	0x0058103F	3
✓	⊞ **timer_1**	Interval timer	clk	**0x00581040**	0x0058105F	4
✓	⊞ **led_red**	PIO (Parallel I/O)	clk	**0x00581060**	0x0058106F	
✓	⊞ **button_pio**	PIO (Parallel I/O)	clk	**0x00581080**	0x0058108F	5
✓	⊞ **switch_pio**	PIO (Parallel I/O)	clk	**0x00581090**	0x0058109F	
✓	⊞ **SEG7_Display**	SEG7_LUT_8	clk	**0x00581100**	0x00581103	
✓	⊞ **sram_0**	SRAM_16Bit_512K	clk	**0x00500000**	0x0057FFFF	
✓	⊞ **SD_DAT**	PIO (Parallel I/O)	clk	**0x005810A0**	0x005810AF	
✓	⊞ **SD_CMD**	PIO (Parallel I/O)	clk	**0x005810B0**	0x005810BF	
✓	⊞ **SD_CLK**	PIO (Parallel I/O)	clk	**0x005810C0**	0x005810CF	
✓	⊞ **DM9000A**	DM9000A	clk_90	**0x005810E8**	0x005810EF	6
✓	⊞ **LCD_RES**	PIO (Parallel I/O)	clk	**0x005810D0**	0x005810DF	
✓	⊞ **T8000A_0**	T8000A	▨	**0x00400000**	0x004FFFFF	
✓	⊞ **ps2keyboard**	keyboard_avalon_interface	▨	**0x005810F0**	0x005810F7	7
✓	⊞ **ps2mouse**	mouse_avalon_interface	▨	**0x005810F8**	0x005810FF	8
✓	⊞ **Audio_0**	AUDIO_DAC_FIFO	clk	**0x00581104**	0x00581107	

Implementation

The whole design adopts a top-down methodology to build the system hardware module in SOPC Builder. All required functional modules are integrated through the Avalon bus to improve system stability. Because a multi-functional digital album particularly needs a stable hardware system when performing multiple functions, we tried to use only SOPC Builder for the hardware.

SOPC Builder's powerful system integration feature enabled us to shorten the design cycle, build a stable system in a short time, complete the system software design on the basis of the hardware, and integrate the software and hardware into a complete system.

SD Card Storage Module

As a totally open standard, SD cards are used in MP3 players, digital video cameras, digital cameras, e-books, audio-visual (AV) appliances, and particularly in ultra-slim digital cameras. SD cards are the same shape as multimedia cards (MMC), but they are a little thicker than MMCs and have larger capacity. Additionally, SD cards are compatible with MMC interface specifications. The nine-pin SD card changes the transmission mode from serial to parallel, improving the transmission speed. It reads and writes faster than MMCs and is more secure.

To make the system more applicable and compatible, we decide to use an SD card as the primary storage media to store photos, music, etc. The one-line SD card read setting on the DE1 board is limited in speed. We modified the setting to a four-line mode to make it more compliant with the system demands.

Table 1 defines the SPI mode time sequence signals, Table 2 defines the four-line SD mode time sequence signals, and Figures 6 through 9 show the time sequences for the SPI and SD card modes.

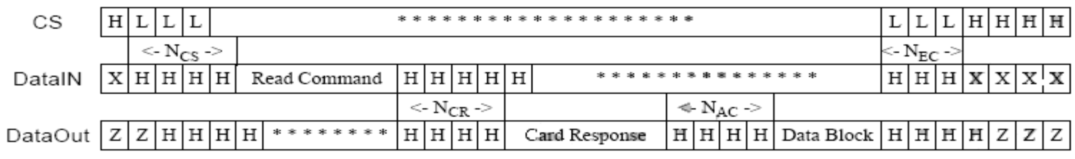
Table 1. SPI Mode Time Sequence

CS	Host-to-card chip selection signal.
CLK	Host-to-card clock signal.
DataIn	Host-to-card data signal.
DataOut	Card-to-host data signal.

Table 2. Four-Line SD Card Mode Time Sequence

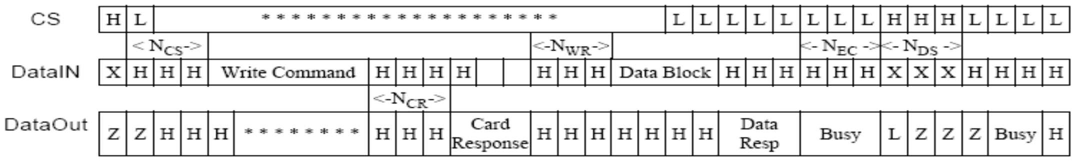
CLK	Host-to-card clock signal.
CMD	Control and answer signal.
DAT0-DAT3	4-bit data line.
VDD, VSS1 and VSS2	Power supply and grounding signal.

Figure 6. SPI Mode Read Time Sequence



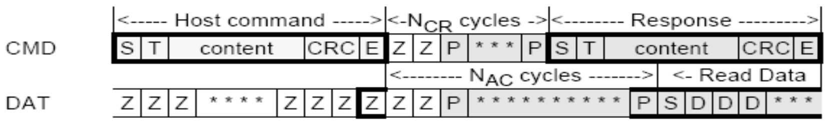
Read Single Block operations - bus timing

Figure 7. SPI Mode Write Time Sequence



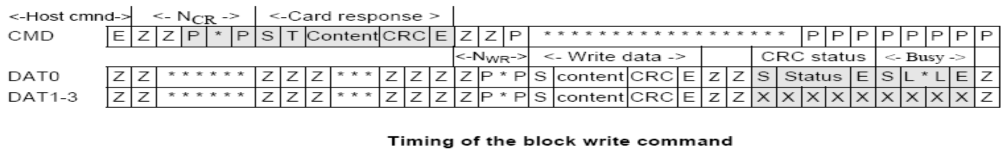
Write operation - bus timing

Figure 8. Four-Line SD Card Mode Read Time Sequence



Timing of single block read

Figure 9. Four-Line SD Card Mode Write Time Sequence



<-Host cmnd->
<- NCR ->
<-Card response >
CMD
E Z Z P * P S T Content CRC E Z Z P * * * * * * * * * * * * * * * * * * P P P P P P P P
<-NWR->
<- Write data ->
CRC status
<- Busy ->
DAT0
Z Z * * * * * * Z Z Z * * * Z Z Z Z P * P S content CRC E Z Z S Status E S L * L E Z
DAT1-3
Z Z * * * * * * Z Z Z * * * Z Z Z Z P * P S content CRC E Z Z X X X X X X X X X Z
Timing of the block write command

PS/2 Keyboard and Mouse IP Core Module

The PS/2 mouse and keyboard protocols are the same. The PS/2 mouse and keyboard perform a bidirectional synchronous serial protocol; each cycle, the data line transmits a bit of data and a clock line transmits a pulse that is read. The keyboard/mouse can transmit data to the host and the host can transmit data to the devices. Because the host has priority on the bus, it controls communication from the keyboard/mouse by pulling the clock low.

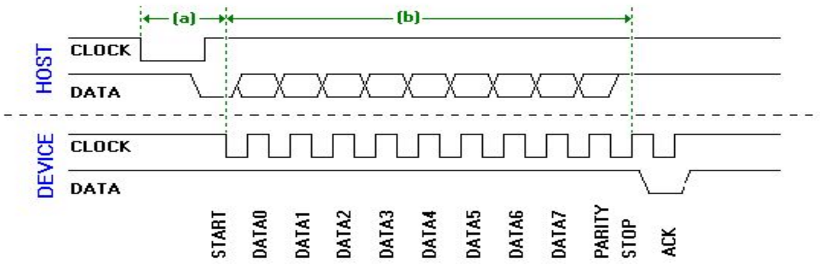
A standard PS/2 mouse supports the horizontal displacement, X, vertical displacement, Y, and mouse left, middle, and right clicks. It reads the input at a fixed frequency and then marks the reflected movement and click status. A standard mouse has two counters to track displacement. The X and Y displacement counters can store 9-bit binary complements, and each counter has a corresponding overflow flag. The overflow flag content and the three mouse-click statuses are transmitted together to the host in the form of a three-byte mobile data packet. The displacement counter represents the displacement from the last displacement data packet sent to the host. Table 3 shows the data format.

Table 3. Displacement Counter Data Format

Original Bit	8 Data Bits	Parity Bit	Stop Bit	Answer Bit
Logic 0	Low bit first	Determine the parity according to the number of 1s in the data bit.	Logic 1	Used in host/device communication.

Figure 10 shows the host/device communication.

Figure 10. Host/Device Communication



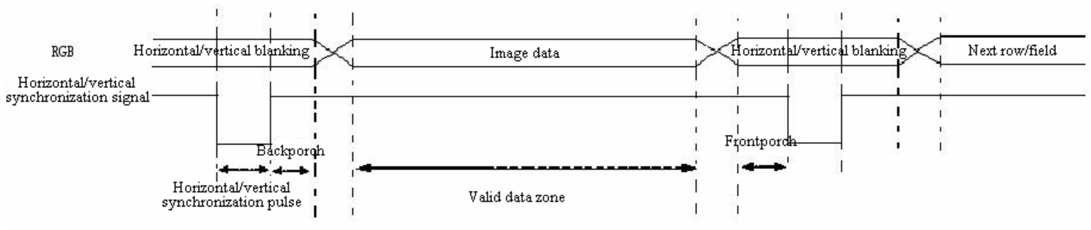
According to standard PS/2 protocol, we compiled the mouse and keyboard hardware modules (that are Avalon slaves in SOPC Builder) to complete the user interface.

VGA Image Display Module

Although this project uses an LCD display, we also designed a VGA interface. The VGA time sequence includes horizontal and vertical time sequences. Both time sequences contain the horizontal (vertical) synchronization pulse, the width from the end of horizontal (vertical) synchronization pulse to the beginning of valid data display zone (backporch), the width of the valid displaying zone, and the width from the end of valid data display zone to the beginning of horizontal (vertical) synchronization pulse (frontporch) parameters. The logic zone between the width of the horizontal valid display zone and the width of the vertical valid display zone is regarded as the video zone, and other zones are regarded as blank zones.

Figure 11 shows the time sequence of one row or one field.

Figure 11. Horizontal/Vertical Time Sequence


RGB
Horizontal/vertical blanking
Image data
Horizontal/vertical blanking
Next row/field
Horizontal/vertical synchronization signal
Backporch
Frontporch
Horizontal/vertical synchronization pulse
Valid data zone

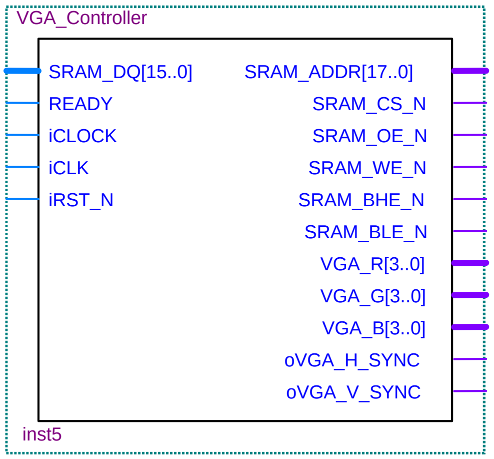
Based on the time sequence in Figure 11, Table 4 shows the time during each stage of the horizontal/ vertical signal.

Table 4. Time During Each Stage of Horizontal/Vertical Signal

Parameter	VGA(640×480)
Horizontal scanning frequency/vertical scanning frequency	31.469 KHz/59.94Hz
Scanline time	31.77 μs/16.68 ms
hsync/vsync	3.77 μs/0.06 ms
Horizontal/vertical frontporch	0.94 μs/0.35 ms
Horizontal/vertical backporch	1.89 μs/1.02 ms
Valid video zone	25.17 μs/15.25 ms

According to the time sequence requirements, we compiled the VGA controller to display images. We used SRAM as the image data memory to display the images after decompression and processing. Figure 12 shows the VGA controller.

Figure 12. VGA Controller



LCD Display and TCB8000A Controller Module

We used a 5.7-inch, 65,000-color thin film transistor (TFT) as the display and the TCB8000A device as the controller. During the design, we only needed to integrate the interfaces (including an 8-bit data line, write transmission signal, address signal, reset signal, and chip select signal) between the controller and the CPU in SOPC Builder, and load the module as a peripheral to the Avalon bus to implement the image data display.

System Software Design

The following sections describe the system software design.

µC/OS-II Operating System

As a free, embedded operation system with public source code, µC/OS-II has been applied widely in many fields worldwide, such as mobile phones, routers, hubs, aerocrafts, medical instruments, etc. µC/OS-II is suited to small control systems because it features high efficiency, small size, excellent real-time performance, good scalability, etc. Software development of the system is completed in the Nios II-integrated µc/OS-II embedded operating system (OS).

In our multi-functional digital album system, we created Task Main (main control tasks), Task_Gui (graphical user interface [GUI] user interface display task), and Task_Music (music playing control task). µC/OS-II schedules the three tasks using a message mailbox.

JPEG CODEC Module

The JPEG CODEC module converts the image format. Digital cameras also embed the CODEC for JPEG images, and this module is indispensable for our digital album. Figures 13 and 14 show the JPEG encoding and decoding procedure.

Figure 13. DCT-Based JPEG Encoding Procedures

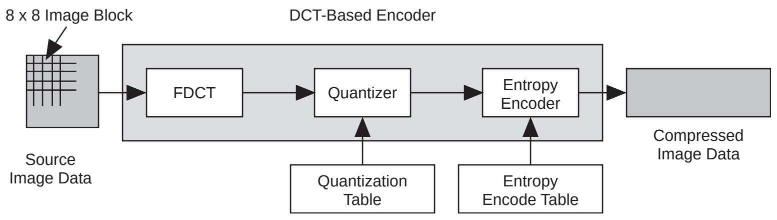
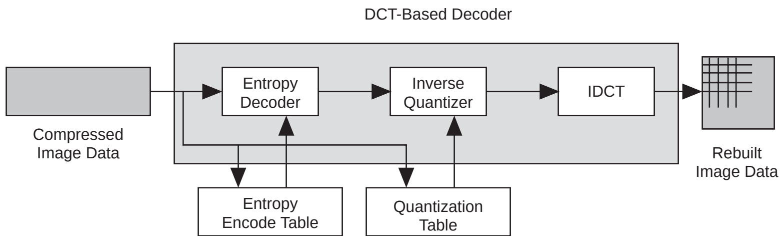


Figure 14. DCT-Based JPEG Decoding Procedures



In the procedures, the core algorithms are discrete cosine transform, quantization, inverse-quantization, and encoding/decoding of the quantization coefficient with a Huffman variable-length encoder. Additionally, the procedures involve technologies such as transforming the color model, a zigzag arrangement of quantization coefficients, the differential pulse-code modulation of the DC coefficients, and the run-length encoding of the AC coefficient. Figures 15 and 16 show the encoding and decoding flow charts, respectively.

Figure 15. Encoding Flow Chart

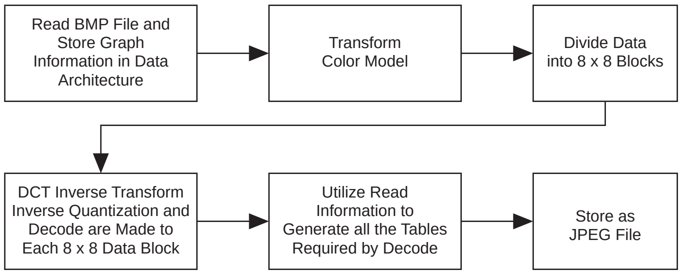
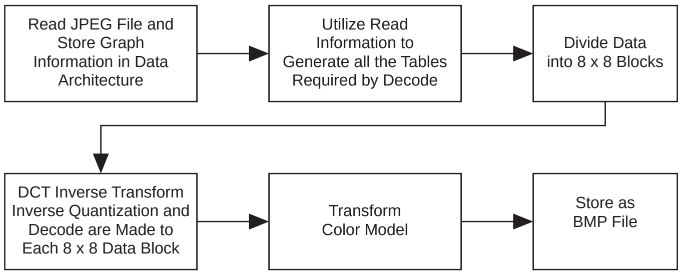

Read BMP File and Store Graph Information in Data Architecture
Transform Color Model
Divide Data into 8 x 8 Blocks
DCT Inverse Transform Inverse Quantization and Decode are Made to Each 8 x 8 Data Block
Utilize Read Information to Generate all the Tables Required by Decode
Store as JPEG File

Figure 16. Decoding Flow Chart

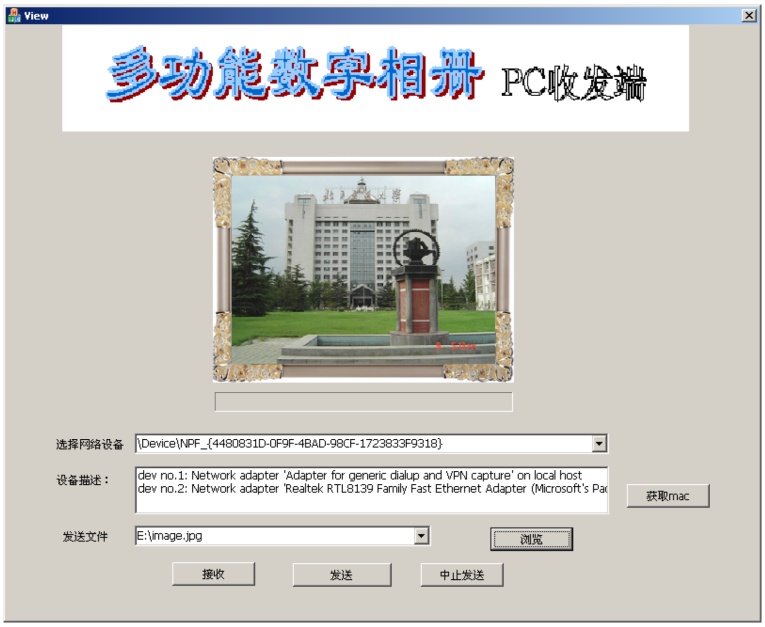

Read JPEG File and Store Graph Information in Data Architecture
Utilize Read Information to Generate all the Tables Required by Decode
Divide Data into 8 x 8 Blocks
DCT Inverse Transform Inverse Quantization and Decode are Made to Each 8 x 8 Data Block
Transform Color Model
Store as BMP File

With these procedures, users can view real-time photos from digital cameras, including JPEGs with different encodings, making the album a real partner to the digital camera and a good photo processing tool during traveling.

Network Transmission and Receiving Module

Migrating µC/OS-II enables us to make a network expansion board in the hardware infrastructure. Our network template creates its own data transmission method by invoking bottom receiving and transmission packet programs with the Nios II processor. The data is packaged at the Nios II transmission terminal and the data is unpacked at the network receiving terminal. Figure 17 shows the multi-functional digital album's transmitting/receiving interface on a PC terminal.

Figure 17. Transmitting and Receiving Terminal Interface



选择网络设备 \Device\NPF_{4480831D-0F9F-4BAD-98CF-1723833F9318}

设备描述：
dev no.1: Network adapter 'Adapter for generic dialup and VPN capture' on local host
dev no.2: Network adapter 'Realtek RTL8139 Family Fast Ethernet Adapter (Microsoft's Pac

发送文件 E:\image.jpg

Accessing the network functions means the system can be edited and viewed on-line, which significantly improves the system's application scope. On the network, the system can receive remote image data without depending on a local SD card data source. Additionally, users can transmit their photos to friends for sharing while traveling.

µC/FS File System

µC/OS-II is contained in the Nios II software development integration environment so that users can apply it in their own software engineering. To make the concurrent task processing and CPU sharing more reasonable, we used µC/OS-II with a file system. In the beginning, we selected the zlg file system, but its speed was unsatisfactory during testing. The reading and writing of 1-Mbyte data into the SD card required 37 seconds and 57 seconds, respectively. We analyzed why it was so time consuming, and found that one problem was the SD card's read/write limitation. On the DE1 development board, the SD card uses one-line reading and writing, i.e., it only has one data line. By adding data lines, we modified the mode to four-line reading and writing. Our tests showed that the speeds were improved to 17 seconds and 27 seconds, respectively. This result was still not quadruple the original speed as we expected.

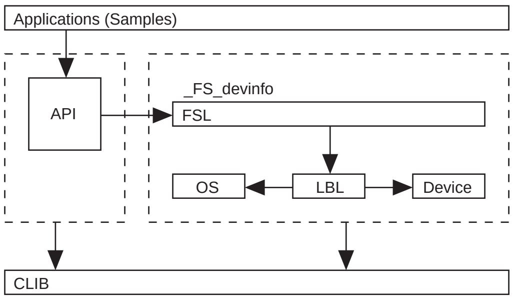
We determined that another reason for the speed problem was the file system. By searching the Internet, we found that zlg/fs had low performance and consumed a lot of processing time. So we decided to use Micrium's µC/FS, which has excellent compatibility with µC/OS-II and high performance. After several weeks effort, we successfully migrated µC/FS version 1.34 to the DE1 board to establish a file system for a four-line SD card. According to the comparison test, the reading and writing speeds were greatly increased, taking just 3.6 seconds and 11 seconds, respectively for 1 Mbyte of data. This speed basically satisfied our speed requirement for accessing data files. However, writing was still slow because when data is written into the SD card, each block needs to calculate a 16-bit cyclic redundancy code (CRC), occupying a lot of transmission time. Therefore, we accelerated this part with a custom instruction. Additionally, if the CPU operating frequency improves, the write speed will also increase.

We briefly introduce the µC/FS file architecture in the following sections. Table 5 shows the µC/FS software packet files. Figure 18 shows the µC/FS block diagram.

Table 5. µC/FS Software Packet Files

Application Program Interface (API)	Some external interfaces.
CLIB	Implements some basic standard C functions, including processing strings and memory.
CONFIG	The configuration items of each target, including all configuration types.
DEVICE	The packaging of the device layer, currently including hard disk, RAM, smc, windows (I/O interface) etc. It primarily implements the FS__device_type interface.
FSL	File system layer (supporting all file systems), currently including the FAT file system.
LBL	Some OS-level operations (number of signals) of the logic block layer (the packaging of the OS and device layer).

Figure 18. µC/FS Block Diagram



Some functions available in the file system are:

- File system control functions
  - *FS_Exit()*—Stop file system.
  - *FS_Init()*—Start file system.
- File access functions
  - *FS_FClose()*—Close a file.
  - *FS_FOpen()*—Open a file.
- Direct input/output functions
  - *FS_FRead()*—Read data from file.
  - *FS_FWrite()*—Write data to file.
- Directory functions
  - *FS_CloseDir()*—Close a directory.
  - *FS_MkDir()*—Create a directory.

- *FS_OpenDir()*—Open a directory.
- *FS_ReadDir()*—Read from a directory.
- *FS_RewindDir()*—Reset position in directory stream.
- *FS_RmDir()*—Remove a directory.

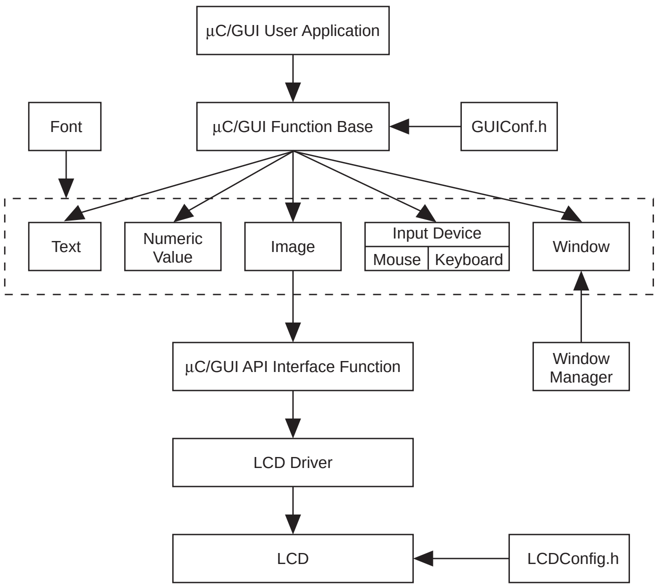
µC/GUI Interface

µC/GUI is an excellent graphic software for embedded systems, featuring open source code, portability, configurability, stability, and reliability. µC/GUI provides rich interface elements, such as buttons, edit boxs, sliders, etc. Additionally, it supports an efficient windows callback mechanism to provide interfaces between interfaces and application functions. We developed the user interface of our multi-functional digital album system with this tool.

µC/GUI Architecture

Figure 19 shows the µC/GUI system architecture.

Figure 19. µC/GUI System Architecture



The µC/GUI function library provides GUI interfaces to user programs. It contains functions such as text, numeric values, 2-dimensional (2-D) images, input devices, and various window objects. The input device can be a keyboard, mouse or touch screen. 2-D images contain pictures, beelines, polygons, circles, ellipses, arcs, etc. Window objects include buttons, edit boxs, progress bars, check boxs, etc.

The µC/GUI function library can be configured via the **GUIConf.h** file, including whether to use a memory device or window management device, whether to support an OS or touch screen, the size of the dynamic memory to be configured, etc. Various hardware-related attributes are defined in the **LCDConf.h** file, such as LCD size, color, and interface function. The LCD driver interprets the µC/GUI functions into the liquid crystal interface function defined in the **LCDConf.h** file regardless of the hardware connection. With the driver, the µC/GUI-LCD hardware interface converts the hardware interface function into an LCD read/write function as defined in the **LCDConf.h** file.

µC/GUI Migration

The first step in migration is modifying the **GUIConf.h** and **LCDConf.h** configuration files. According to the display module requirements of the digital album system, we configured the relevant parameters in the configuration files.

µC/GUI provides drivers for different LCD controllers. For example, the KS0713, SED1335, and T6963 controllers have corresponding LCD drivers. However, our system's display module is the TOPWAY TCB8000A LCD controller with a TFT 65,000 color LCD screen, and µC/GUI does not provide a driver for it. Furthermore, unlike the LCD controllers for which µC/GUI provides drivers, the TCB8000A controller has an independent screen control instruction system. These factors made it difficult to migrate the controller for µC/GUI.

During migration, we first used the TCB8000A instruction system to let µC/GUI provide the most API functions for the upper application function. Then, for API functions that the TCB8000A controller does not support in hardware, we repaired the drivers using software. Last, by testing a large quantity of controls, we adjusted the TCB8000A display instruction parameters used in the LCD drivers, optimizing the LCD driver performance and seamlessly migrating µC/GUI for the TCB8000A controller.

µC/GUI provides mouse and keyboard drivers. However, when we implemented the mouse and keyboard modules in the system, we needed to consider driver problems when combining the IP module with µC/GUI. Therefore, we compiled the keyboard data receive function in the Nios II processor so that it is invoked by the µC/GUI keyboard input driver and the obtained hardware keyboard value is transmitted to µC/GUI to complete a keyboard event. Then, µC/GUI processes the key during a message processing loop and transmits the key to the current focus form in µC/GUI. With this system module, we can transmit any input information to µC/GUI and implement text input for system editions.

The mouse driver is more complex. Fortunately, µC/GUI has good mouse driver support and its form management and message processing mechanism allows the mouse information to be updated and processed in real time.

The album includes the following functions:

1. The µC/FS file system informs µC/GUI of the photo data stored in the attached SD card.

2. After JPEG decoding, the RGB data is output.

3. The system conducts image processing, output display, editing, and processing in various modes.

Figures 20 and 21 show the album in view and edit mode, respectively.

Figure 20. Album Schemes in View Mode

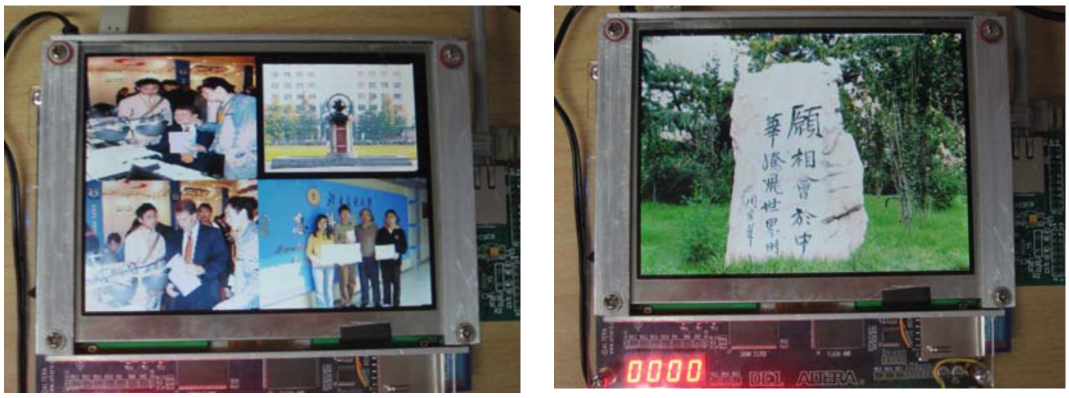
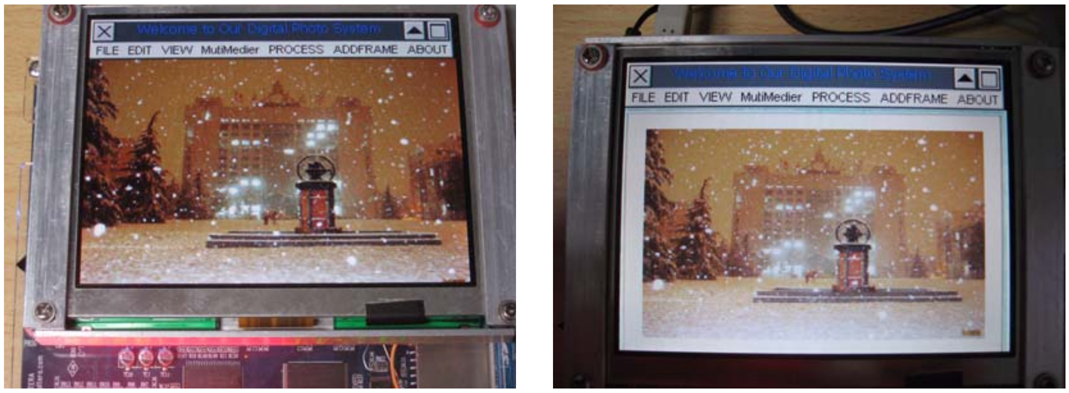


Figure 21. Album Menu Schemes in Edit Mode


Welcome to Our Digital Photo System
FILE EDIT VIEW MutiMedier PROCESS ADDFRAME ABOUT
Welcome to Our Digital Photo System
FILE EDIT VIEW MutiMedier PROCESS ADDFRAME ABOUT

Digital Watermark Embedding

This module is embedded as a menu management sub-block, so that we can add a digital watermark to our photos to protect them. Due to time and system resource limitations, we did not integrate this module into the system.

Applying SOPC Concepts

For this design, our hardware structure is completely integrated using system-on-a-programmable-chip (SOPC) concepts. SOPC design has the advantage that hardware can be rebuilt and reconfigured, which is a unique feature of FPGAs in hardware development. Additionally, the top-down SOPC design flow dramatically shortened our design cycle. At the very beginning of the design, we considered using SOPC design concepts for our system. In our opinion, the biggest advantage of embedded systems is not making a perfect design, but using the least number of resources to complete the design while optimizing power consumption and integrity. We also must consider future development of the system. If the current design is limited in future development and expansion, it will not be a successful design. In our design, we preserve many operations for future system expansion to allow users to bring their ideas into the system to perform more and better functions.

Design Features

Our system has the following features:

- The µC/OS real-time operating system (RTOS) is a key factor in the normal operation of all functional modules in the system. From a system perspective, this design integrates a digital album, image processing, compression and decompression, transmission, and receiving into one system for the first time. The Nios II CPU and Avalon bus arbitration played an important role in implementing the system. To achieve the integration and operation of functional modules in the multi-functional album, we introduced an OS embedded in the Nios II processor to schedule the functional tasks, providing stability, improving system performance, and simplifying our design.

- This design employs multiple user interfaces including SD card memory, LCD interface, Ethernet interface, PS/2 mouse and keyboard interfaces, etc. We loaded these peripherals onto the Avalon bus using SOPC Builder. In spite of the large number of interfaces, the modules operate correctly in the integrated SOPC environment.

- For the software interface, we successfully migrated µC/GUI into the the Nios II environment. The user interface makes the design friendly and photos are easy to manage.

- To use the mouse and keyboard interfaces simultaneously, we expanded the original PS/2 interface so that one PS/2 interface can allow the Nios II processor to respond to the mouse and keyboard interrupt signals simultaneously. To provide fast reading and writing to the system for the SD card interface, we modified the original one-line SPI mode into a four-line SD mode, improving the read/write speed by four times and satisfying the system requirements.

- Implementing µC/GUI, µC/FS, and µC/OS-II in the Nios II environment provided a good user interface for our album. The whole system must operate under a control environment, and combining the three elements meets the system requirements and provides good driver support to the hardware infrastructure.

- The expanded GPIO helped us load the network control module and LCD. The high-integrity SOPC design ensures stable communication in the whole system.

Conclusion

For this contest, we successfully implemented a multi-functional digital album based on the Nios II processor. We learned a lot from participating in the contest and now understand Nios II embedded systems more.

The system design targets an FPGA. We fully used Nios II features to control the whole system's operation. With OS scheduling, the tasks are managed according to the task priority. Where conflicts exist, the tasks are delivered to the OS for management. For example, in the beginning, the mouse and keyboard IP cores needed to be interrupted continuously and the Nios II processor needed to respond continuously, causing conflicts in some programs and causing the mouse not to operate in the GUI. After we introduced the OS, there were no system conflicts.

By participating in the contest, we learned about the high integrity of SOPC Builder, the stability of the Nios IDE, and the convenience of system debugging. The SOPC-based reconfigurable hardware and software design and top-down design method provided a flexible system implementation, higher system integrity, and a shorter design cycle.

Finally, thanks to Altera for giving us this rare opportunity to create the design.