Design and Implementation of NIOS II System for Audio Application

S. Moslehpour, K. Jenab, and E. H. Siliveri

***Abstract*—This project deals with the NIOS II system, which reads wave files from an SD card in real time and produces sound output through speakers while providing user interaction such as pause music, fast forward and rewind. The Altera DE2 board supports communication through the SD card. It also features sound recording for 10 seconds by giving a microphone to the LINE-IN provided on the DE2 board and speakers to the LINE-OUT. These features are realized using Altera SOPC builder in the Altera Quartus 9.1 environment. The overall goal of this project is to explore various applications that are possible with Altera's DE2 Board.**

***Index Terms*— DE2 board, NIOS II processor, SOPC builder, SD card, LCD, Cyclone II 2C35 FPGA.**

I. Introduction

Audio devices these days come in many forms of embedded systems and they are widely used all over the world due to their compactness and ease of use. This popularity of such devices was the inspiring factor behind this project [6]-[8]. There are myriad applications of these devices such as mobile phones, portable video games etc.

This article discusses in detail, the implementation of the audio application using SOPC builder, NIOS II processor and Altera DE2 board. Given the toolset, it was a challenging and a rewarding experience to have implemented this project.

II. SOPC (System-on-A-Programmable-Chip) Builder

SOPC builder is a powerful system development tools for creating systems based on processors, peripherals, interfaces and memories. SOPC Builder is implemented for the purpose of generating a complete system-on-a-programmable-chip (SOPC) by consuming less time than the accustomed integration methods [4].

Altera has SOPC Builder functionality built in Quartus II, which accordingly connects the soft-hardware components to construct a complete computer system that can be controlled on any of the FPGA chips and is also capable of producing interconnect logic automatically. It is outfitted with a library of built-in components such as a Nios II soft processor, memory controllers, interfaces, standard peripherals and custom peripherals.

In SOPC Builder, the system components are routed in a GUI (Graphical User Interface). The GUI is exclusive in configuring the soft-hardware components. It is a general-purpose tool for creating systems that includes a soft processor apart from the Nios II processor. It also contributes to writing software and system simulation.

A. Core Functionalities of SOPC Builder

- Describes the hardware of the system.
- Performs the system generation.
- Performs memory mapping for initiating the software development
- Produces test bench to simulate the design.

B. Architecture and Design of SOPC Builder

Designs using SOPC Builder are generated to develop a top level HDL (hardware description language) file by connecting various modules together. These various modules are considered the building blocks for the SOPC Builder system [9]. For the connection of multiple components in the system, the modules use Avalon (Avalon switch interconnect) interfaces, such as memory-mapped, streaming and IRQ (interrupt request).

C. SOPC Components

The components in SOPC Builder are referred as hardware blocks of the system. They contain HDL descriptions of the hardware of the components and interfaces used for the hardware. They also contain the description of the parameters that determine the operation of the components (Fig. 1).

The SOPC components are connected to the system interconnect fabric using the Avalon Memory-Mapped interface (Avalon MM) or the Avalon streaming interface (Avalon-ST) [4].

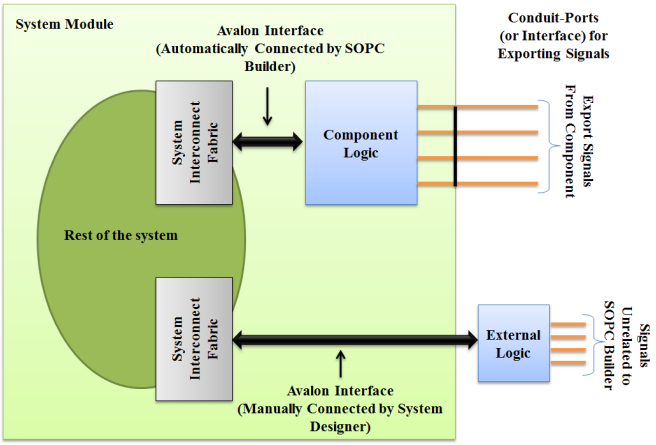


Fig. 1. SOPC Builder with system components inside and outside the system module

Manuscript received May 28, 2013, revised June 25, 2013.

S. Moslehpour is with the College of Engineering, Technology, and Architecture at Hartford University, Hartford, CT, USA (e-mail: moslehpou@hartford.edu).

K. Jenab is with the Society of Reliability Engineering-Ottawa Chapter, Ottawa, Ontario, Canada (e-mail: jenab@ieee.org).

E. H. Siliveri was with the College of Engineering, Technology, and Architecture at Hartford University, Hartford, CT, USA (e-mail: siliveri@hartford.edu).

D. Types of SOPC Components

SOPC Builder's inbuilt components are

a. Static HDL Components
b. Generated HDL Components
c. Composed HDL Components
d. Custom Components
e. Third-Party Components

(a) Static Components: These are the components that accept the VHDL parameters.

Examples: Address widths, Data widths and FIFO depths.

(b) Generated Components: These are the components whose hardware description language file is generated based on the value of its specified parameters.

Example: A parameter that controls the number of interfaces.

(c) Composed Components: These are the components that are constructed from combinations of other components.

(d) Custom Components: The design flow used to merge the custom components into the SOPC Builder is as follows:

1. Determine the interfaces required by the custom components.
2. Write the logic for each custom component.
3. Develop the custom components with the hardware description language files by using the component editor.
4. Represent the custom component in the system by an instance.

(e) Third-Party Components: These components are built by third parties. Components external to the SOPC Builder: For the components that interfere to external logic or off-chip devices with Avalon- compatible signals outside the SOPC Builder system, the component files describe only the interface to the external logic. The connection of signals at the top-level of SOPC Builder to pins or logic defined outside the system is done manually.

E. SOPC Design Flow

The design flow of SOPC Builder is as follows (Fig. 2)

a. Add components by Component Editor.
b. Initiate Simulation of the system.
c. Develop the system design by adding components, IRQs and addresses.
d. Start the system generation.
e. Conduct system level simulation.
f. Compile the system design.
g. Download .sof file to an Altera FPGA.
h. Perform Testing.

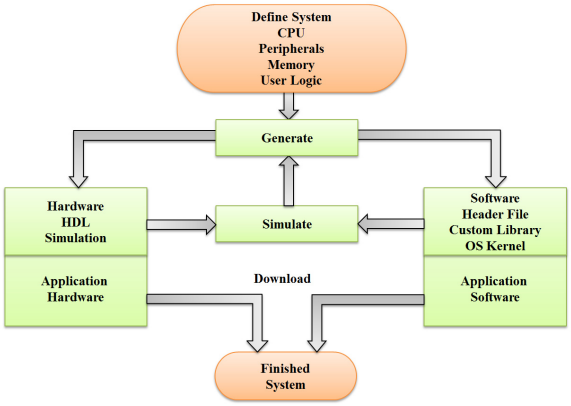


Fig. 2. SOPC Design Flow

F. Avalon Switch Interconnect

The Avalon Memory-Mapped (Avalon-MM) interface has a bandwidth of steep structure generally used for interfacing the components of the system. The Avalon switch interconnects uses less logic but supplies absolute flexibility. This interconnect is a cross-connect fabric used for the purpose of switching and multiplexing.

The Avalon switch interconnect fabric is a combination of interconnect and logic resources. It is used for stocking the Avalon memory mapped master and slaves on the components. It is referred to a device having information about connections of the entire system and its components. It assures that connections between the master and slaves are routed precisely. This meets the requirements of the components.

The interconnect fabric permits the connection of an unlimited count of master components and slave components. These master and slave components can have a one-to-one connection, a one-to-many connection, a many-to-one connection, or a many-to-many connection. The system interconnect fabric is used to support the interfaces for the on-chip components of the system and interfaces for the off-chip devices of the system. In this interconnect fabric, master components and slave components with altered data widths are supported.

The system interconnect fabric also serves as a platform for the master and slave components running with several clock domains. It also supports master and slave components with several memory mapped ports.

The system implementation fabric for the Avalon-MM interfaces acts as a partial crossbar interconnects structure. The partial crossbar interconnect structure is a matrix with multiple inputs and multiple outputs. This interconnect structure arranges simultaneous paths between the master and slave components. In the Cyclone II FPGA, routing resources and synchronous logic constitute the system interconnect fabric.

G. Functionalities of System Interconnect Fabric

a. Decoding Address
b. Multiplexing Datapath
c. Insertion of Wait State
d. Pipelined Read Transfers
e. Multi-master System Arbitration
f. Burst Adapters
g. Interrupts
h. Reset Distribution

H. Automated System Generation

SOPC Builder system integration tools automatically perform the process of configuration of the processor features, hence the hardware of the design is produced that is used to program an Altera device. The graphical user interfaces (GUI) help in structuring of Nios II systems with multiple peripherals and memory interfaces (Fig. 3).

After system generation, you can download the design onto a board, and debug the software executing on the board.

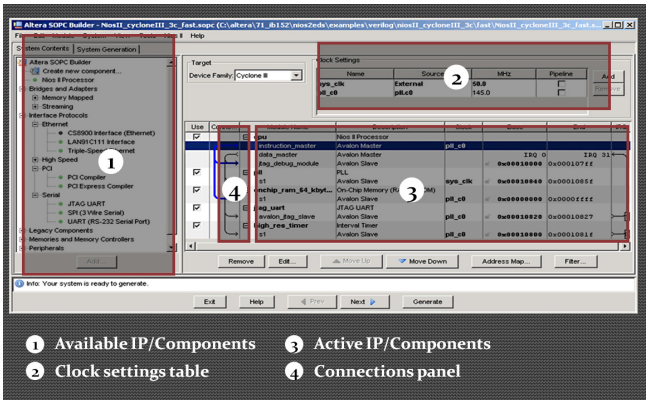


Fig. 3. SOPC Builder graphical user interfaces (GUI) of the system used in the project

III. NIOS II System on Altera's DE2 Board

The Nios II processor and many other components such as standard peripherals and custom peripherals are used for the formation of a total system that can be integrated into a Nios II system on Altera's DE2 board [3]. The process of interfacing the Nios II processor and peripherals to the DE2 board chips is enabled on the Cyclone II FPGA chip. The interconnection network connecting these components in the FPGA chip is called the Avalon Switch Fabric (Fig. 4).

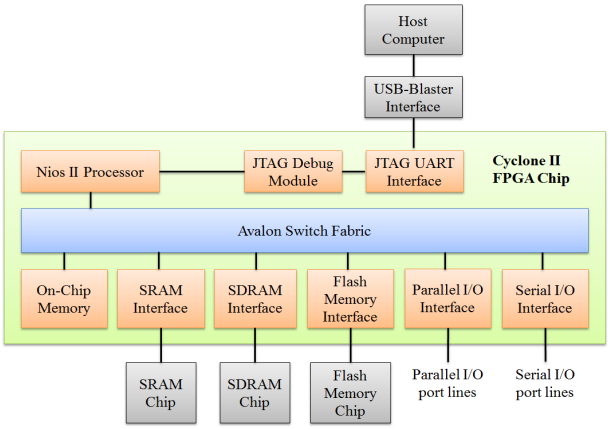


Fig. 4. Nios II system implemented on the DE2 board

The memory blocks available in the Cyclone II FPGA chip serve as the on-chip memory for the Nios II processor. These memory blocks can be connected to the Nios II processor directly with the help of the Avalon network. The Input / Output interfaces are used for connecting I/O devices. A JTAG UART interface is used for the purpose of providing a Universal Serial Bus link between the Altera's DE2 board and the host computer to which the board is connected. This Universal Serial Bus link is called the USB-Blaster. The JTAG Debug module is used by the host computer to control the Nios II processor, downloading programs into the memory, starting and stopping execution. The memory chips such as SRAM and SDRAM can be connected by applicable interfaces. A hardware description language is used to define all the Nios II system components on the Cyclone II FPGA chip.

A. Nios II Processor

The Nios II processor is a configurable RISC processor [10]. Hardware structure includes:

a. Functional units of the Nios II architecture
b. Fundamentals of the hardware implementation

B. Fundamentals of the hardware implementation

The hardware implementation of Nios II architecture explains an instruction set. Any functional unit whose hardware is implemented can be programmed in software. Every hardware implementation has different objectives, for example, that the size of the core should be small and must yield high performance (Fig. 5). These features help the Nios II architecture adjust to many different applications.

Implementation of the processor core specifically requires any of three trade-offs: more performance or less performance of a feature; inclusion of a feature in the core or exclusion of a feature; and implementation of the hardware and software programming of the features included in the core.

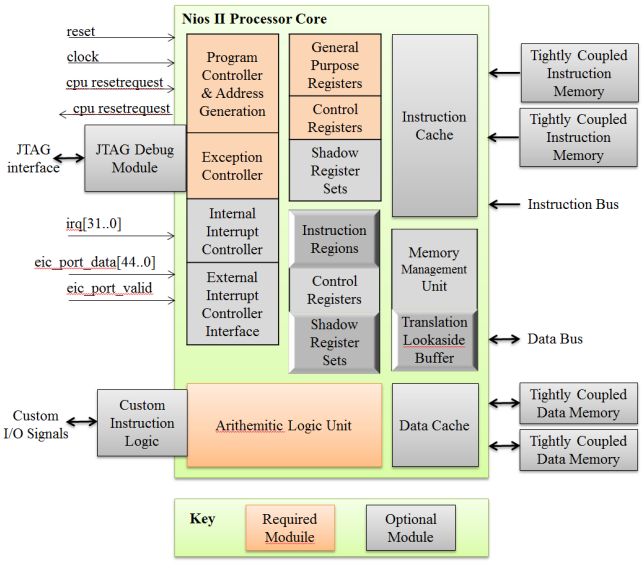


Fig. 5. Nios II Processor Core Block Diagram

C. Nios II Processor Core Architecture

The architecture of Nios II processor core consists of [3]

a. Register file
b. Arithmetic logic unit (ALU)
c. Interface to custom instruction logic
d. Exception controller
e. Interrupt controller
f. Instruction bus
g. Data bus
h. Memory management unit (MMU)
i. Memory protection unit (MPU)
j. Cache memories
k. Tightly-coupled memory interfaces for instructions and data
l. JTAG debug module

D. Customizing Nios II Processor

Altera FPGAs offer to add new features in order to increase the performance of the Nios II processor. The advantage of customization is elimination of unnecessary processor features and peripherals to suite the hardware design in a smaller and lower-cost device.

The following are the possibilities to program customized pins and logic resources available on the Altera DE2 board [1]:

a. Rearrangement of the pins on the chip is the best way to reduce the design of the board. For example, address and data pins can be moved for external SDRAM memory to cut the traces of the board in short.

b. The use of extra pins and logic resources on the chip is independent of the processor. Extra resources supply a few more extra gates and registers for the purpose of designing the board (Fig. 6). Also, these extra resources affect the complete system. Using extra pins and logic resources on the chip, the additional peripherals for the Nios II processor can be implemented. We can easily access the additional peripherals from the library of SOPC Builder, which are used for connecting the Nios II processor [3].

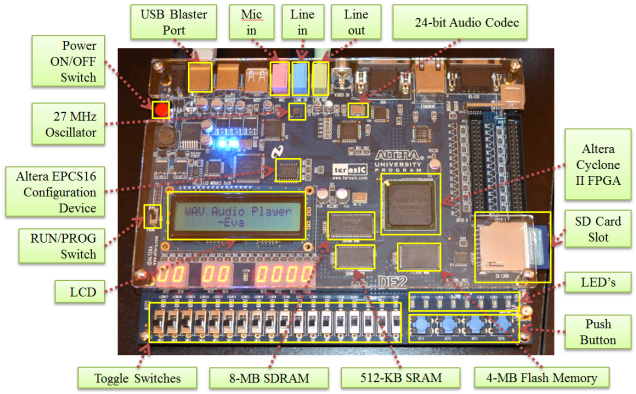


Fig. 6. Altera DE2 Board (Cyclone II 2C35 FPGA) Components and interfaces

E. Features of DE2 Board (Altera Cyclone II 2C35 FPGA)

Following is the list of features available on the DE2 Board [4]:

a. USB Blaster is inbuilt on the board mainly for programming purposes and for controlling API

b. JTAG Mode and AS Mode are supported

c. 8 MB ($1M \times 4 \times 16$) SDRAM

d. 1 MB Flash Memory

e. SD Card Port

f. 4 Push-buttons

g. 18 DPDT (Double Pole Double Throw) switches

h. 9 Green User LEDs

i. 18 Red User LEDs

j. 50 MHz Oscillator

k. 27 MHz Oscillator

l. 24-bit Audio CODEC including line-in/out, and mic-in jacks

m. VGA DAC

n. VGA out connector

o. TV Decoder

p. TV-in connector

q. Ethernet Controller

r. USB Controller as Host and Slave

s. RS-232 Transceiver and 9-pin connector

t. Mouse and keyboard connectors (PS/2)

u. IrDA transceiver

v. 2 x 40-pin Expansion Headers

IV. SOPC Builder IP Cores

A. Secure Data Card IP Core

An SD card (Secure Data Card) is a data storage device [2]. It is often used in digital appliances such as cameras, camcorders, MP3 players etc. It is very handy and it allows the data stored on it to be transferred across multiple devices. The Altera DE2 board contains an SD card port. We can connect an SD card to the FPGA-based board, and it allows access large amount of data.

The SD Card IP Core is a hardware circuit on the DE2 board. It facilitates the use of an SD card. When it is included in a design and connected to the SD card port, the core detects an SD card when plugged into the port. This lets the circuit easily access any data stored on it.

B. Functionality of SD card IP core

This IP core acts as an interface between the system and the SD card. In the SD card IP core the signals on the left side map to the Avalon interconnect. The Avalon interconnect receives read and write requests from the Avalon Interface Finite State Machine (FSM) and interprets them as command or data requests. The SD card is configured by the commands received, and the commands could reference various sections of the SD card as required. The raw data stored on the SD card is accessed by data requests. Once the finite state machine confirms the request issued, the SD card interface module is activated. This module communicates with the SD card to process the request by using serial communication protocol, and the result of the finite state machine is returned. Once the operation is complete, the result of the request is sent by the Avalon Interface FSM through its interconnect.

C. Instantiating the Core in SOPC Builder

To be able to add the SD Card IP Core in a SOPC design, it is necessary to create an instance of the core in the design. On including this IP core in the SOPC design, it is very important to connect the card's ports to their corresponding pins on DE2 board [2]. This can be done through the pin assignment tool in the SOPC builder.

TABLE I: Pin Assignments for SD Card Port on DE1, DE2, DE3 Boards

Pin Name	DE1	DE2	DE3
b_SD_cmd	PIN_Y20	PIN_Y21	PIN_R10
b_SD_dat	PIN_W20	PIN_AD24	PIN_P7
b_SD_dat3	PIN_U20	PIN_AC23	'Z'
o_SD_clock	PIN_V20	PIN_AD25	PIN_P8

On instantiating the SD card IP core in the design, the Avalon Interface, the reset signals are automatically connected to the synchronous and asynchronous reset signals. The clock input can either be set to 50 MHz or connected to an external clock source.

D. Software Programming Model

We can build software programs to communicate with the SD card directly. This can be done with the help of memory-mapped registers and a memory-mapped buffer, or using HAL drivers.

E. Direct SD Card Communication using Memory-Mapped Registers

Using the memory-mapped registers, information can be exchanged between the SD card and the SOPC system. The SD card interface status can be read programmatically through these registers. Read/Write commands can also be sent using the same mechanism. This allows reading, writing and erasing blocks of data available on the SD card.

F. Hardware Abstraction Layer Device Driver

HAL device drivers facilitates convenient access to all the data store on the SD card. HAL drivers can act as a FAT (File Allocation Table) reader and FAT writer and thus make it possible to access SD card data. This requires the data on the SD card to be saved in FAT16 format. Other file systems such as FAT12, FAT32 and NTFS are not compatible with the DE2 board.

G. Formatting the SD Card in FAT16 format

Data is stored in several clusters. These clusters are identified by an ID that ranges from 2 through 65520. Files stored on the SD card, in the FAT16 format, occupy several clusters depending on the size of the file. The SD card IP core we use works only with FAT16 because it requires at least 4087 clusters and at most 65520 clusters so that they can be represented by 16-bit IDs.

V. Project Implementation

This project deals with a NIOS II system that reads the wave files from the SD card in real time and produces sound output through speakers while providing user interaction such as pause music, fast forward and rewind. For implementing this system, the following are required:

a. SD card Controller
b. SPI mode of communication
c. Audio codec
d. SOPC Builder
e. Nios II processor

A. SD card Controller

DE2 board comes with an SD card port which can be connected to an SD card (Fig. 7). An SD card is one of the memories used in this project.

Fig. 7. SD card

For the SD card to be detected, it should be formatted with FAT 16 file format. The FAT (File Allocation Table) is a table marking showing the position of files in the system (Fig. 8). This system partitions the whole memory into blocks of data, each block consisting of 512 bytes and the smallest possible data is 16 bit long [5].

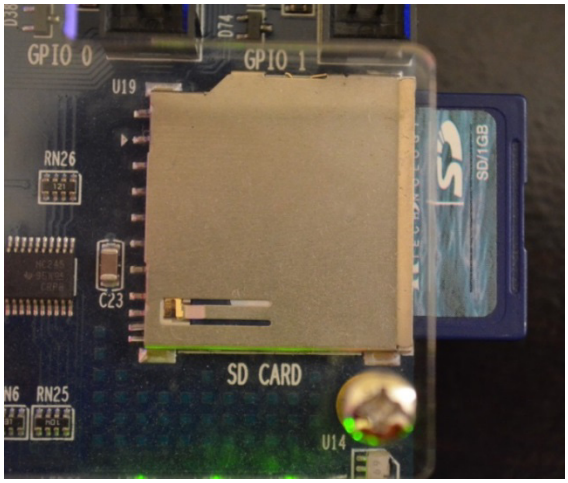


Fig. 8. SD card Controller on DE2 Board

B. Serial Peripheral Interface (SPI) mode of communication

SPI is a synchronous serial data link that functions with a single master and multiple slaves (at least one). In this design we use the SPI mode of communication to access the SD card (Fig. 9). Data transfers are carried out using one bit. This allows for an easy and simple interface design [1].

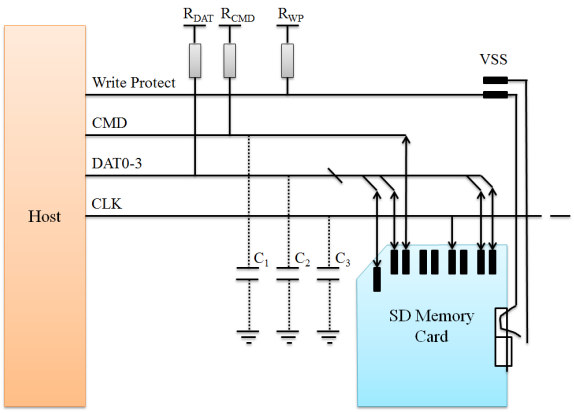


Fig. 9. SPI Interface

The pins used in SPI are

a. SD_CLK
b. SD_DATA
c. SD_CMD

SD_CLK: Used to control the SD card clock speed
SD_CMD: Used to send commands to the SD card. CMD constitutes standard SD card commands, and ACMD constitutes specific commands.
SD_DATA: Used for data transfer.

C. Audio Codec (Wolfson Wm8731)

An audio codec is nothing but a device that performs encoding of analog audio signal to a digital signal. It also decodes a digital signal to an analog audio signal. It constitutes an ADC and DAC, which are clocked by the same

clock input. In order to generate sound using the board, the audio data is transferred from the SD card to the analog line out port of the DE2 board. In this project, we use a 24-bit sigma-delta audio codec called as Wolfson WM8731 [3].
Modules used in the Audio Codec system are:

a. FIFO module
b. I2C interface unit
c. PPL module

FIFO module is nothing but a buffer with a width of 256x16 bits. It connects the Nios and Digital to Analog Converter (DAC). The data flow of the codec is controlled by the I2C interface unit. PPL module produces an 18.4 MHz clock frequency. The .wav file is sampled at 48 KHz.

D. SOPC Builder

All the components used in this project are connected using the SOPC builder. The SPI interface pins are the PIO's used for communication with the DE2 board (Fig. 10).

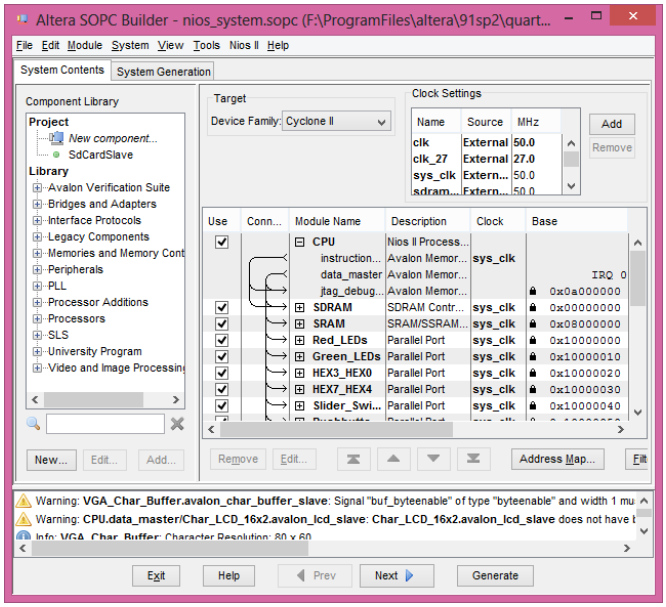


Fig. 10. SOPC Builder

E. Nios II Processor

The Nios II processor is used to implement all the functions of this project. The Nios II processor is based on a Reduced Instruction Set Computer (RISC) architecture. The general purpose register's operands are used to perform arithmetic and logic operations.

In the Nios II architecture, we have two separate buses for sending instructions and transferring data. This architecture is also popularly known as Harvard architecture. It has 32-bit registers and 32-bit word length as well.
There are three available configurations in which the Nios II processor can be implemented. They are as follows:

a. **Nios II / f**: This is a "fast" version optimized for high performance.
b. **Nios II / s**: This is a "standard" version that optimizes the resources available in the FPGA device. This trades off resource optimization with performance.
c. **Nios II / e**: This is an "economy" version that needs least possible FPGA resources. This limits the feature set to bare minimum functionality (Fig. 11).

	Nios II Family: Cyclone II	Performance at 50 MHz	Logic Usage	Memory Usage
Nios II/e	**RISC** **32-bit**	Up to 5 DMIPS	600-700 LEs	Two M4Ks (or equiv.)
Nios II/s	RISC 32-bit **Instruction Cache** **Branch Prediction** **Hardware Multiply** **Hardware Divide**	Up to 25 DMIPS	1200-1400 Les	Two M4Ks + cache
Nios II/f	RISC 32-bit Instruction Cache Branch Prediction Hardware Multiply Hardware Divide **Barrel Shifter** **Data Cache** **Dynamic Branch Prediction**	Up to 51 DMIPS	1400-1800 LEs	Three M4Ks + cache

Fig. 11. NIOS II core configurations

F. Nios II Code Flow

The Nios II processor looks for the SD card if it is inserted into its slot when the board is switched on. The SD card is initialized as soon as it is inserted and the files are detected and the file directories are listed using FAT 16 (Fig. 12). FAT 16 is used to search any media content available on the card which is inserted in the SD card slot. The media files are stored as file type: .WAV. The file names are then printed on the LCD character display of the DE2 board, and the user can select the desired song. Once the user selects a song, he can play the selected song from the beginning. Songs are played continuously when the SW[0] switch is turned on.

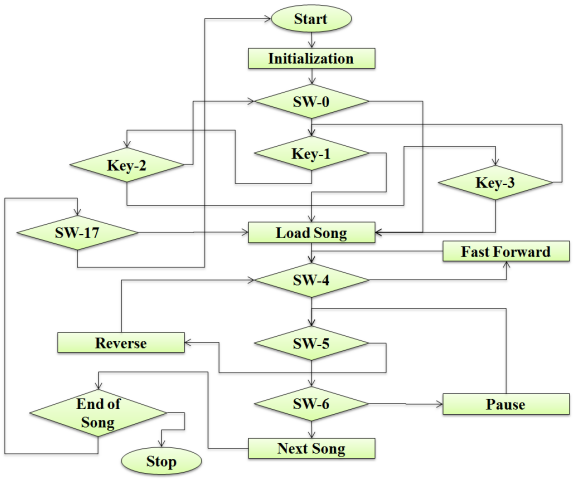
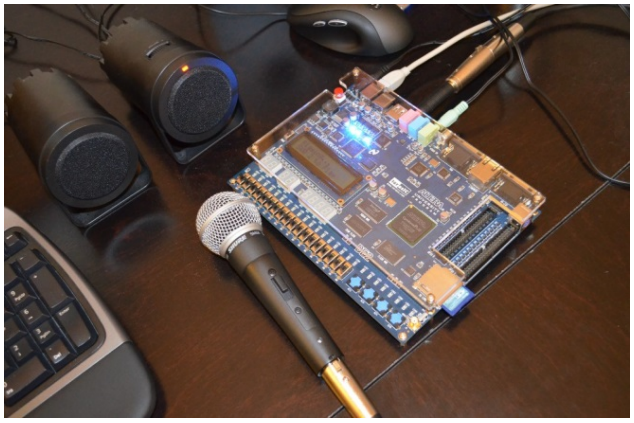


Fig 12 NIOS II Code Flow Chart

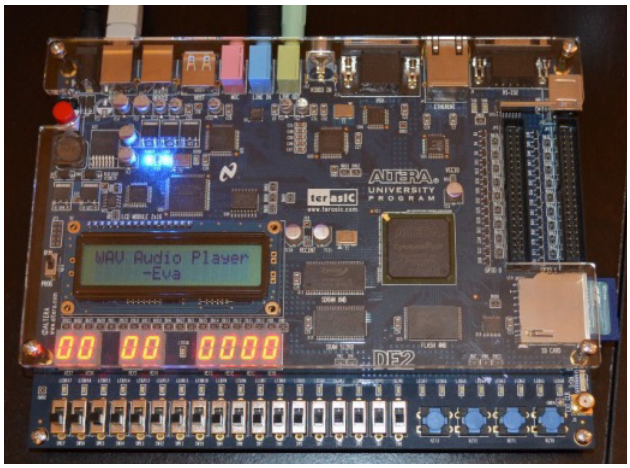
To browse through the available songs in the playlist, key-2 and key-3 of the board are used and a specific song can be selected. The name of the current song is displayed on the LCD screen for browsing and for selecting the song. Once browsing through the playlist is done by the user from the directory and a particular song is selected, Key-1 is pressed to select the song, which is indicated on the LCD screen. Once the song is selected, the digital data is read from the card and sent to the FIFO buffer. Switch SW-4 is used to fast-forward, reverse play is enabled using SW-5, and the song can be paused with SW-6. On completion of the song, the LCD displays the main menu on the board. SW-17 is used to reset the board.

VI. Results and Discussion

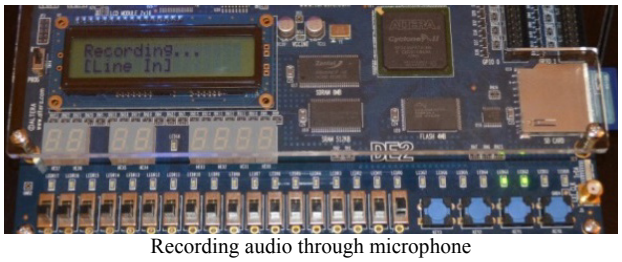
This project reads the wave files from the SD card and produces sound output through speakers while providing user interaction such as play, pause, and reverse play at various speeds as shown in Fig. 13. The SD card is primarily used as the source of music files in this project. It also features sound recording by giving the microphone to the LINE-IN provided on the DE2 board and speakers to the LINE-OUT. Once the sound is recorded, there are several playback options such as pause, fast forward and rewind. These features are achieved using SOPC Builder tool provided by Altera, and programming of the DE2 board is done by Nios II Eclipse.



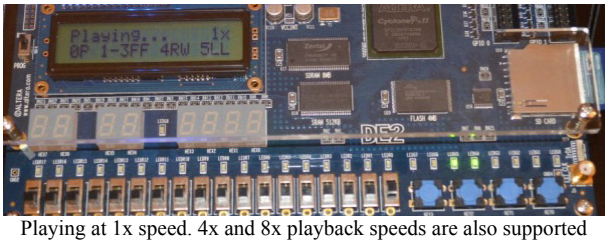
The project setup includes speakers, mic and the DE2 board connected to the computer



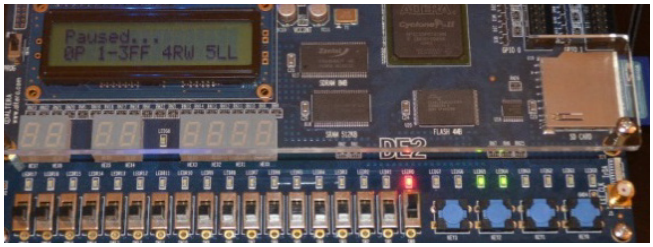
Start screen of the project on DE2 board



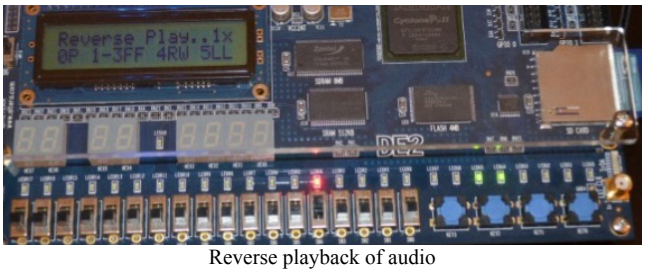
Recording audio through microphone



Playing at 1x speed. 4x and 8x playback speeds are also supported



Paused audio



Reverse playback of audio

Fig. 13. Demonstration of NIOS II System on DE2 Board for Audio Application

References

[1] Altera Audio/Video Configuration Core for DE2-Series Boards. (July 2010). [Online]. Available: ftp://ftp.altera.com/up/pub/Altera_Material/10.1/Universiy_Program_IP_Cores/Audio_Video/Audio_and_Video_Config.pdf

[2] SD card IP Core. Altera University Program Secure Data Card IP Core. (March 2009). [Online]. Available: ftp://ftp.altera.com/up/pub/University_Program_IP_Cores/90/SD_Card_Interface_for_SOPC_Builder.pdf (accessed September 19 -2012)

[3] Wolfson Electronics. (2004, April). *Portable Internet Audio* CODEC *with Headphone Driver and Programmable Sample Rates*. (WM8731 Rev3.4). [Online]. Available: https://instruct1.cit.cornell.edu/courses/ece576/DE2_Datasheets/Audio%20CODEC/WM8731_WM8731L.pdf (accessed October 22-2012)

[4] Altera SOPC Builder User Guide. (2010, December). [Online]. Available: http://www.altera.com/literature/ug/ug_SOPC_builder.pdf

[5] Altera Embedded Peripherals IP Guide. (2011, June). [Online]. Available: http://www.altera.com/literature/ug/ug_embedded_ip.pdf

[6] S. Moslehpour, K. Jenab and B.S. Pabla, "Implementing a soft core NIOS II processor for VGA application," *International Journal of Engineering Research and Innovation*. vol.4, no.2, pp. 12-26, 2012.

[7] S. Moslehpour, K. Jenab and S. Valiveti, "GPS time reception using Altera SOPC builder and Nios II: Application in train positioning," *International Journal of Industrial Engineering and Production Research*, vol.23, no.1, pp. 13-21, 2012.

[8] S. Moslehpour, K. Jenab and B. K. Matcha, *Design of the Nios II System for the Playing of Wave Files on an Altera DE2 Board*, 2012.

[9] J. O. Hamblen and T. S. Hall, "Using system on a programmable chip technology to design embedded systems," *IJCA*, vol.13, no.3, pp. 1-11, 2006.

[10] *My First NIOS II Software, Altera Corporation based on Altera Complete Design Suite Vrsion9.1*,. January 2010. [Online]. Available: http://www.altera.com/literature/tt/tt_my_first_nios_sw.pdf

Saeid Moslehpour is an associate professor and department chair in the Electrical and Computer Engineering Department in the College of Engineering, Technology, and Architecture at the University of Hartford. He holds Ph.D. (1993) from Iowa State University and Bachelor of Science (1989) and Master of Science (1990) degrees from University of Central Missouri. His research interests include logic design, CPLDs, FPGAs, Embedded electronic system testing and distance learning.

Kouroush Jenab is a senior member of IEEE who received the B.Sc. degree from the IE Department at Isfahan University of Technology (1989), the M.Sc. degree from the IE Department at Tehran Polytechnic (1992), and the Ph.D. degree from the Department of Mechanical Engineering at the University of Ottawa (2005). He served as a senior engineer/manager in auto, and high-tech industries for 18 years. He joined the National Research Council Canada as a research officer where he participated in several international research projects. In 2006, he joined the Department of Mechanical and Industrial Engineering at Ryerson University, Toronto, as an assistant professor. Currently, Dr. Jenab is education chair of the Society of Reliability Engineering (SRE)-Ottawa Chapter. He has published over 81 papers in international scientific journals and conferences, edited a special issue on Applied Computational Techniques in Engineering and Technology for the International Journal of Industrial Engineering Computations, and produced over 29 technical reports.

Evangeline Harica Siliveri received the bachelor degree of engineering in electronics and communications from JNTU College of Engineering, India, in May 2009. She received her Master's degree in Electrical Engineering at the University of Hartford in 2013.