I.J. Engineering and Manufacturing, **2018, 5, 32-48**
Published Online September 2018 in MECS (http://www.mecs-press.net)
DOI: 10.5815/ijem.2018.05.04

Available online at http://www.mecs-press.net/ijem

Design and Implementation of a DDR2 SDRAM Controller for Audio Data on a Reconfigurable Platform

Arun Tigadi [a], Dr Hansraj Guhilot [b]

[a]*Assistant Professor, Department of E and C,KLE Dr. M.S.Sheshgiri College of Engineering and Technology, Belagavi,India*
[b]*Principal, K.C.College of Engineering And Management Studies And Research, Thane (E), India*

Received: 21 May 2018; Accepted: 18 June 2018; Published: 08 September 2018

Abstract

Multimedia applications play a very important role in the field of VLSI design and embedded systems. They need a large amount of memory storage with higher bandwidth and higher speed. To overcome this hazard, a memory controller is required. A memory controller is a device that stores the data and gives it back whenever required. Real-time recording of an audio data and finally storing it without losing the data is a difficult task. This paper describes the usage of Double Data Rate Synchronous Dynamic Random Access memory controller for storing the audio data. The design uses finite state machine (FSM) architecture that is developed for testing of this algorithm. Audio codec device is used for the conversion of analog data into digital and vice versa. The tool used to simulate this design is Xilinx ISE design suite. The hardware used to synthesize this design is FPGA Spartan-3 kit.

Index Terms: DDR2 SDRAM (Double Data Rate Synchronous Dynamic Random Access Memory), Audio Codec (coder-decoder), FPGA(Field Programmable Gate Array), UART(Universal Asynchronous Receiver and Transmitter).

1. Introduction

The memory controller is a very important component in the field of VLSI design. DDR2 SDRAM is the higher version of DDR SDRAM. When transferring an audio data speed, quality and power consumption are the three major factors. The audio decoder used is audio codec 97 which consist of audio jacks and Wolfson integrated chip for conversion of analog data into digital and vice versa.

* Corresponding author.
E-mail address: arun.tigadi@gmail.com

This project can be used in many of the applications such as mobile phone communication, audio/video conferencing, audio-based web search, weather forecasting, and many more applications.

The major difference between these is the prefetch length. The prefetch length of DDR SDRAM is 2n whereas that of DDR2 SDRAM is 4n meaning the internal bus width of DDR2 SDRAM is four times wider than the external bus width. [3]

2. Features of DDR2 SDRAM

Now let us compare DDR SDRAM with DDR2 SDRAM. At certain clock frequency, DDR2 SDRAM transfers twice the data per clock cycle than DDR SDRAM. The power consumption of DDR SDRAM is 2.5V, and that of DDR2 SDRAM is 1.5V. The chip density of DDR SDRAM is 1 GB, and that of DDR2 SDRAM is 4 GB. Thus, DDR2 SDRAM has higher speed, density, and low power consumption then DDR SDRAM. The clock frequency of DDR2 SDRAM is 133 MHz and the on board clock frequency is 50 MHz It has 8192 rows, and the data should be refreshed after every 64ms. It has a 13-bit of the row address and 10 bit of column address. It has a 16 bit of bidirectional data bus. It has four internal banks for the operation. A new feature added to DDR2 SDRAM is ODT (On Die Termination) which improves the signal quality and reduces the layout issues. It must be turned on before data transmission and turned off immediately after data transmission.

3. Proposed Memory Controller Design

The below figure1 represents the main block diagram of this proposed design. The source can be anything such as mike, which takes analog input and gives it to Matlab, which converts the analog data into a digital stream and gives to the FPGA on which our memory controller design is going to sit.

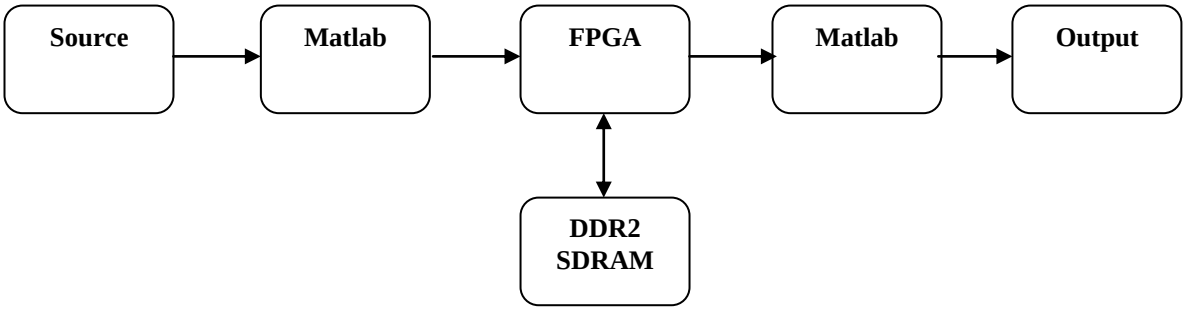

Fig.1. Block Diagram of the Proposed Design

The memory controller then stores the digital data on to the DDR2 SDRAM. Whenever we want to play back the audio data the memory controller on the FPGA takes the digital data from DDR2 SDRAM and gives again to Matlab. The Matlab then converts digital data into analog and gives to the output device.

The output device can be anything such as speakers or headphones.[4]

3.1. DDR2 SDRAM

DDR2 SDRAM is a double data rate synchronous dynamic random access memory. DDR2 SDRAM transfers twice the data on a single clock cycle. Figure2 represents the basic block diagram of DDR2 SDRAM. It has differential clock CK and CK#. All the input signals and address signals are samples at the crossing of the positive edge of CK and negative edge of CK#. CKE is the clock enable which should be kept high for all the operations. CS# is the chip select which should always be low to keep the chip active. RAS#, CAS#, and

WE# are the three command inputs, which performs all the operations such as read, write, refresh, bank active, precharge, etc of DDR2 SDRAM. A0 to A13 is the 14-bit of the row address, and it has a 16-bit of the bidirectional data bus.

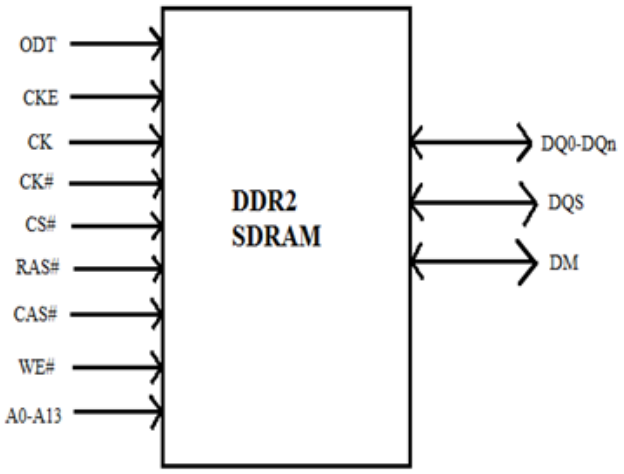

Fig.2. Block diagram of DDR2 SDRAM

DM is the data mask, which is the input for a write operation to occur. When DM is low, the input is masked, and when DM is sampled high, the write operation occurs. DQS is a bidirectional data strobe and ODT is on-die termination, which improves the signal quality. [7]

Table 1. The truth table of DDR2 SDRAM

Operations	**CKE**	**CS#**	**RAS#**	**CAS#**	**WE#**	**BA**	**An-A11**	**A10**	**A9-A0**
Refresh	H	L	L	L	H	X	X	X	X
Precharge	H	L	L	H	L	BA	X	L	X
Bank active	H	L	L	H	H	BA	Row_addr	Row_addr	Row_addr
Write	H	L	H	L	L	BA	Col_addr	L	Col_addr
Read	H	L	H	L	H	BA	Col_addr	L	Col_addr

The above table 1 shows truth table for DDR2 SDRAM, as per the table different signals needs to be either made high or low according to which all the operations occur. These are the signals which are going to control the read pr write operations on the memory device.

3.2. DDR2 SDRAM Operations

Refresh: Refresh means reading the data from a particular address and rewriting the same data on that particular address without changing it.

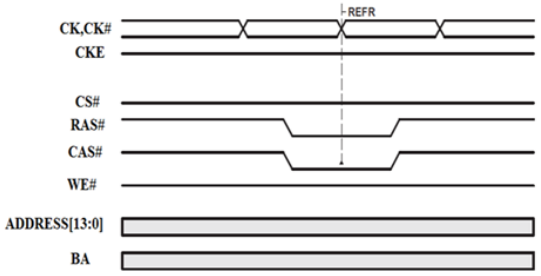

Fig.3. Waveform for Refresh

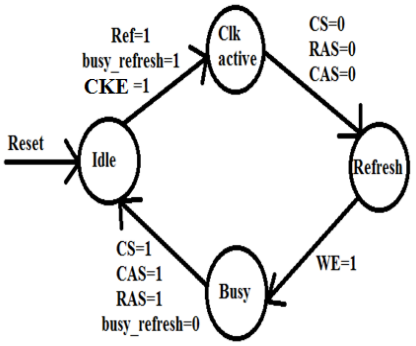

Fig.4. FSM for the Refresh Operation

Here the discussion is w.r.t the waveform for the refresh operation and state diagram for the refresh operation. When Ref is given high the busy_refresh flag is set to high, and the clock is enabled (CKE=1) it enters into a state called clk active, and when CS, RAS, CAS these signals are low, it enters into refresh state. WE is finally sampled high where it goes into a busy state where the refresh operation occurs. And when CS, CAS, RAS are sampled high, it enters into an idle state. The data must be refreshed after every 64ms so that the data is not lost. This is the refresh operation.

Precharge: Precharge is used to open a row in a selected bank or open a row in all the banks. Figure 6 and 7 shows the waveform and state diagram for precharge operation.

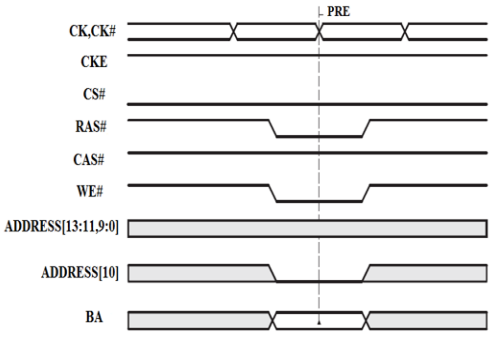

Fig.5. The Waveform for the Precharge Operation

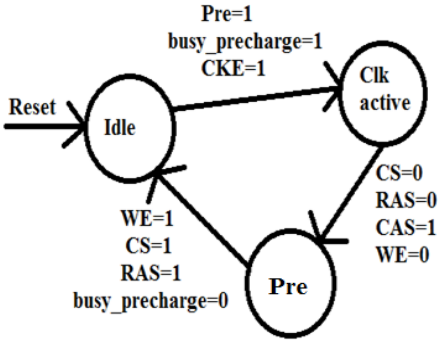

Fig.6. FSM for the Precharge Operation

The next command discussed is the pre-charge command. the waveform for the pre-charge operation and FSM for the pre-charge operation are as shown in the above figures. When Pre is given high the busy_precharge flag is set to high, and the clock is enabled (CKE=1) it enters into a state called clk active, and when CS, RAS, WE these signals are low and CAS is high, it enters into pre-state where the precharge operation occurs. The A [10] bit of row address should be low during pre-charge operation. When CS, CAS, RAS are sampled high, and the busy_precharge flag is set to low it enters into an idle state.

Bank Active:

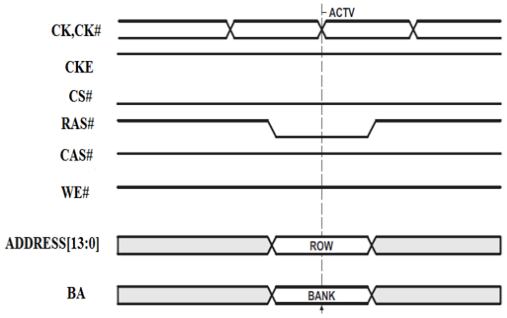

Fig.7. The Waveform for Bank Active Operation

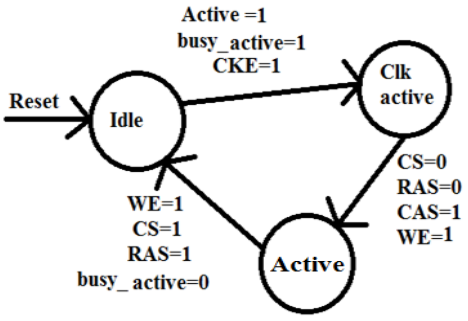

Fig.8. FSM for Bank Active Operation

When Active is given high the busy_active flag is set to high, and the clock is enabled (CKE=1) it enters into a state called clk active, and when CS, and RAS these signals are low whereas CAS and WE are high, it enters into an active state where the bank active operation occurs. And when CS, CAS, RAS are sampled high, and the busy_active flag is set to low it enters into an idle state.

Write:

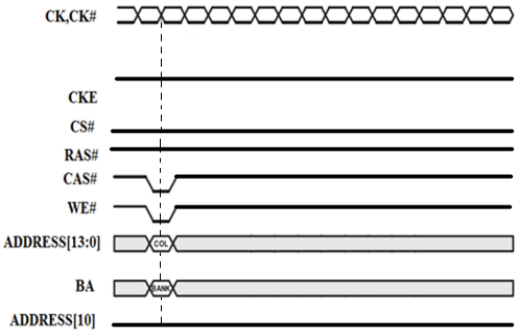

Fig.9. The Waveform for a Write Operation

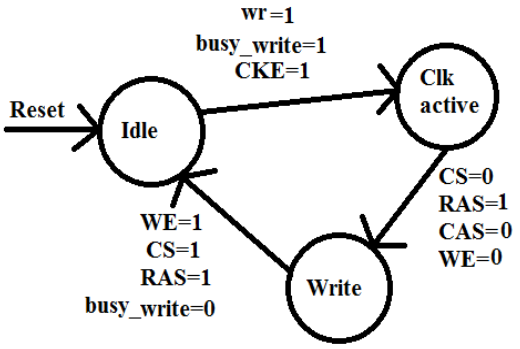

Fig.10. FSM for a Write Operation

When wr is given high the busy_wr flag is set to high, and the clock is enabled (CKE=1) it enters into a state called clk active and when CS, CAS and WE these signals are low whereas RAS is high it enters into write state where the data is written onto the particular bank and the write operation occurs. And when CS, CAS, RAS are sampled high, and busy_wr flag is set to low it enters into idle state.

Read:

When rd is given high the busy_rd flag is set to high and the clock is enabled (CKE=1) it enters into a state called clk active and when CS and CAS these signals are low where as RAS and WE are high it enters into reading state where the data is read from the particular bank, and the read operation occurs. And when CS, CAS, RAS are sampled high, and the busy_rd flag is set to low it enters into an idle state. While the read operation occurs the A [10] bit of row address should be low and the correct column address should be present for the data to be read.

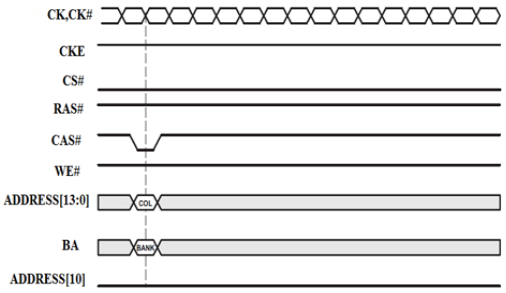

Fig.11. Waveform for Read Operation

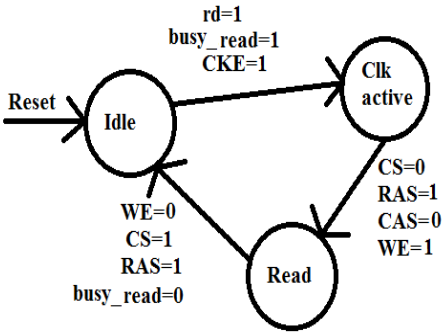

Fig.12. FSM for a Read Operation

3.3. The Architecture of DDR2 SDRAM

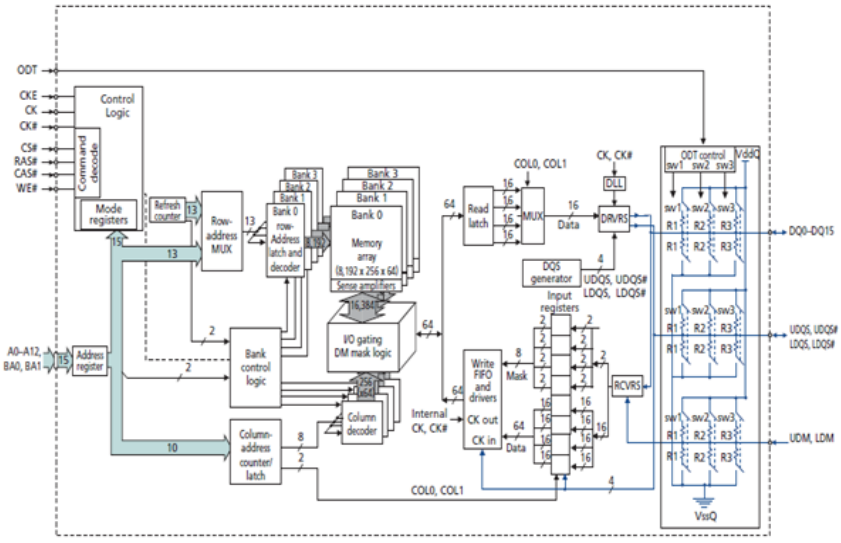

Fig.13. The internal architecture of DDR2 SDRAM

The figure13 shows the internal architecture of DDR2 SDRAM IP. The memory controller includes the control logic block that includes the command decode that indicated the command signals CS, RAS, CAS, WE. It has four banks from bank0 to bank3, which is controlled by bank control logic. It has other blocks such as column decoder, write FIFO and drivers, read latch, i/o gating DM mask logic, etc. The memory controller also provides a reordering option that reorders received requests to optimize data throughput and latency.

4. Implementation

4.1. EDK and SDK Part of the Design

EDK refers to embedded development kit and SDK refers to software development kit. EDK is a Xilinx tool used to design complete embedded processor system for implementation on FPGA device. It is a soft processor, which includes Xilinx platform studio, SDK, and embedded processing intellectual property. SDK is a platform on which embedded development kit is produced. [11]

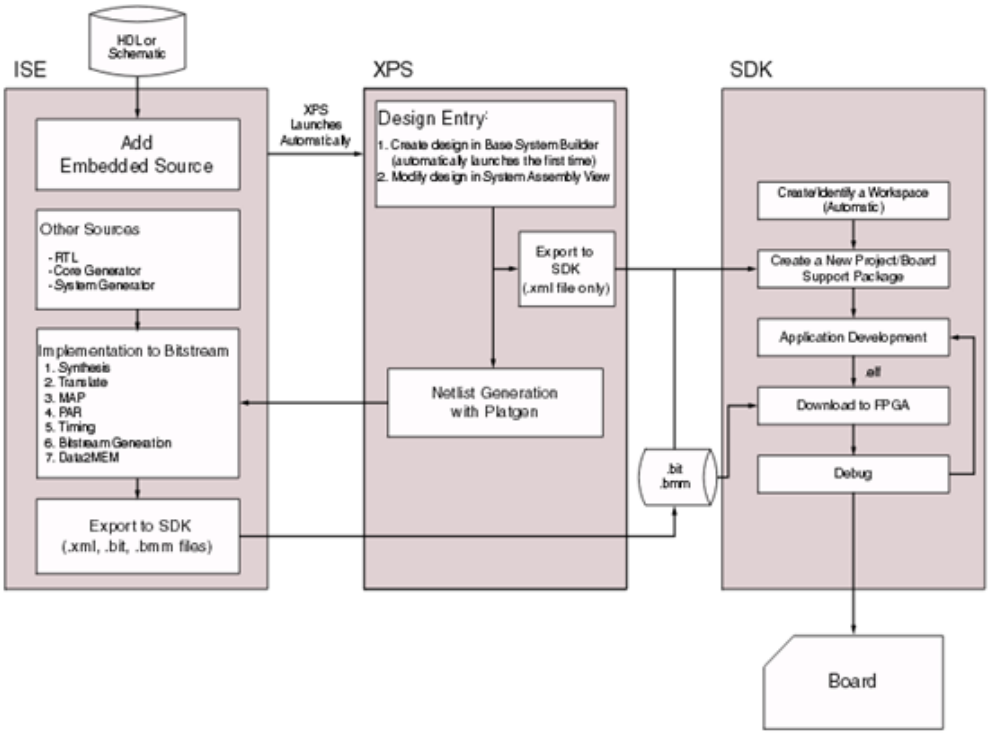

Fig.14. EDK and SDK

ISE is integrated synthesis environment, which consists of the program in HDL, which simulates it and then implements, the program and generates the bit files with extensions .xml, .bit, .bmm, etc.

The XPS launches automatically where in it exports the EDK to SDK. In EDK the base system builder is created and then modifies the design in system assembly view. The net list is generated and is exported to SDK. The SDK creates a new project and programs the FPGA to browse the .bit and .bmm files and finally runs in hardware.

4.2. The MATLAB Part

The above figure 15 represents the Matlab window. The input is given through mike after getting the command recording audio, please says something and is sent to the FPGA, which stores it on the memory with the help of the memory controller after sending the audio the message is displayed on Matlab window as a file sent. When we want to play back the audio, the FPGA takes the audio file stored on the memory, gives it to the Matlab, which outputs using speakers, and displays the message as playing audio and after finishing display the message as finished.

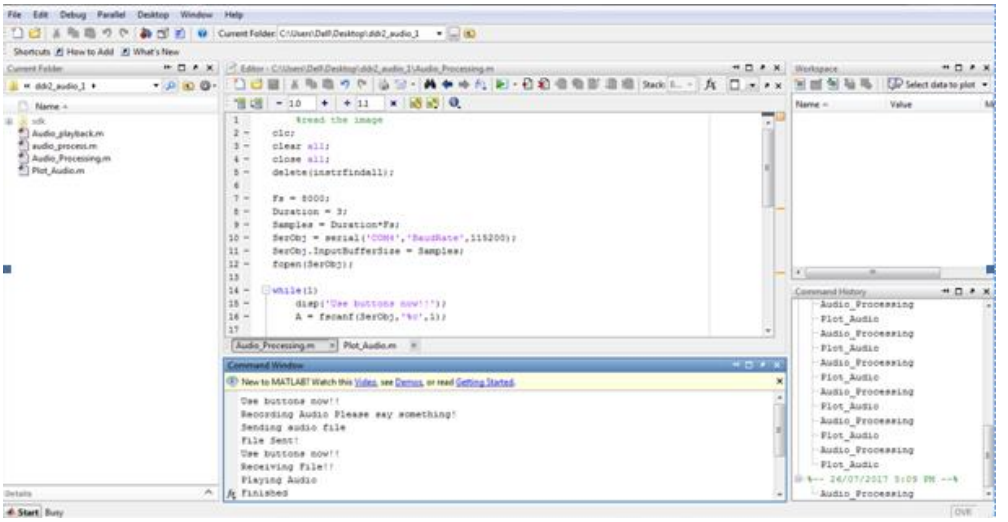

Fig.15. MATLAB Execution Part

5. Results and Observation

5.1. The DDR2 SDRAM

In this section, we describe the figures of merit obtained for our proposed architecture and design when implemented on FPGA. The functionally verified netlist is synthesized using Xilinx ISE and implemented on spartan board. Figure 16 represents the RTL schematic for the reconfigurable device and figure 17 represents the RTL schematic for the IP design. Figure 18 and 19 represent the read and write waveform for DDR2 SDRAM according to the command given the operations are performed. When CS, CAS and WE these signals are low whereas RAS is high the write operation occurs. When CS and CAS these signals are low whereas RAS and WE are high the read operation occurs. The read and write operation happens concerning the commands being generated by the command generated which are intern depended on the type of operation which user wants to perform. Once either read or write operation is initiated the appropriate bank, row address has to be activated and then it will be transferred to the row buffer from the device. Then the actual read will happen. In case of a write operation, the data to be written will be written first on to the row buffer which then will be transferred on to the device. The resource estimation, power analysis of the design and timing is done using Xilinx ISE tool. As the per the design specification the design is synthesised and implemented on Spartan board and the area utilization is mentioned in the table 2 shown below

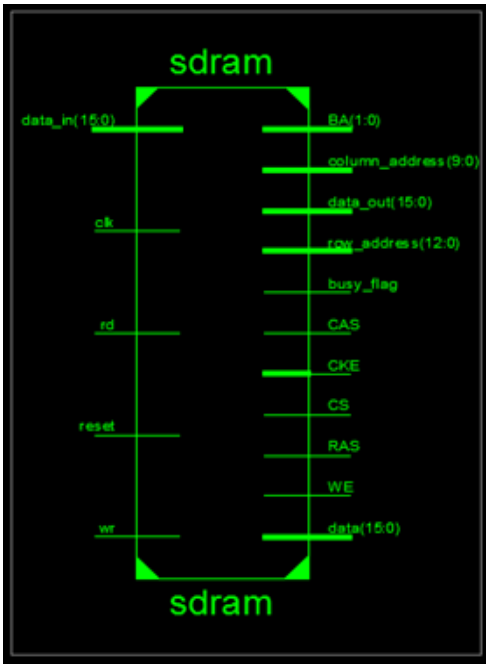

Fig.16. RTL Schematic of the Reconfigurable Device

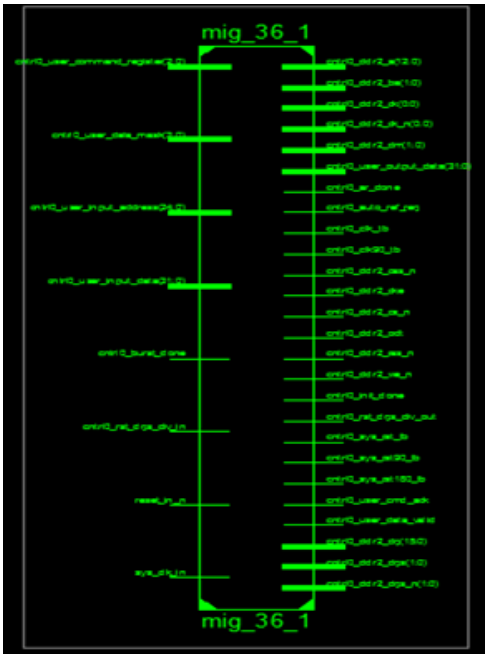

Fig.17. RTL Schematic with IP

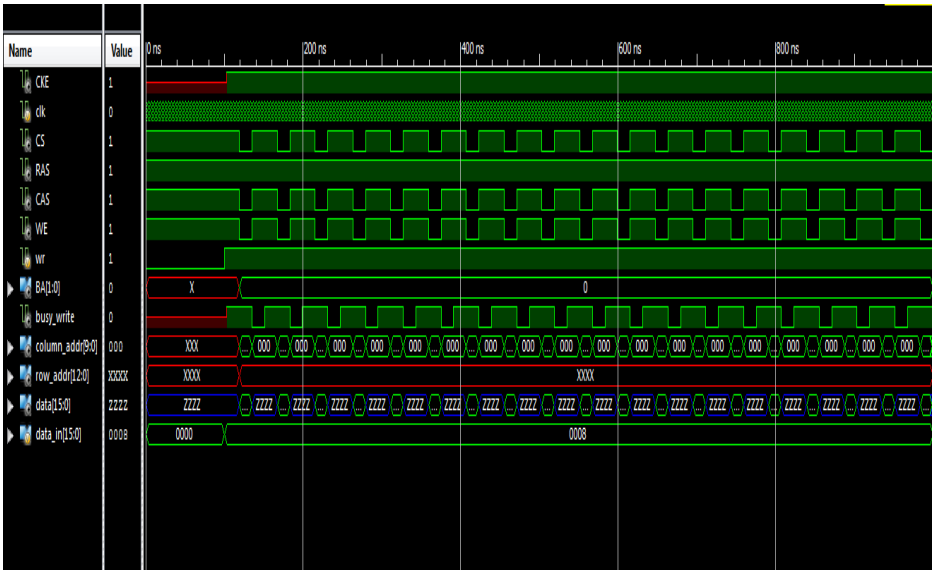

Fig.18. Simulation Result of the Write Signal

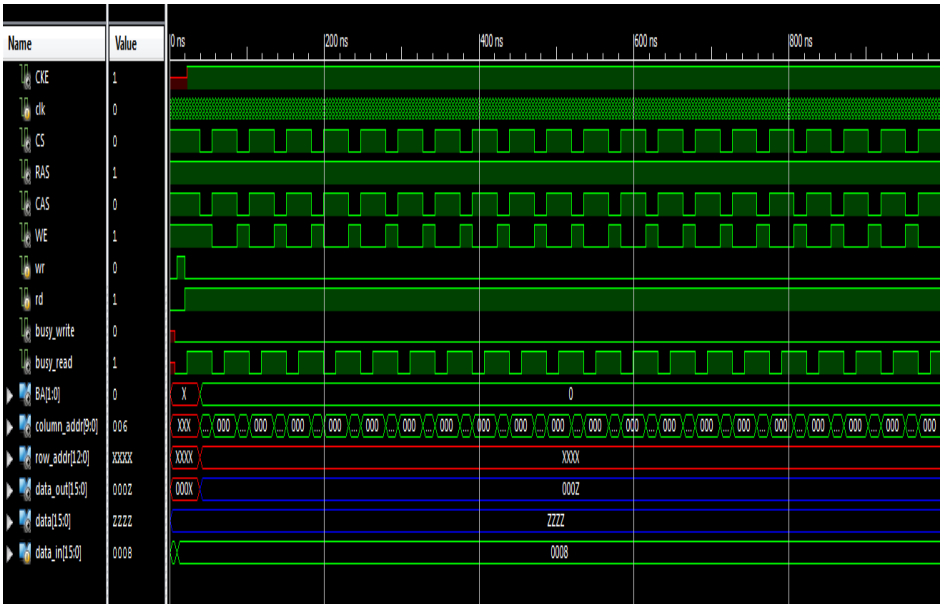

Fig.19. Simulation Result of the Read Signal

The above table 2 shows the power and timing report for DDR2 SDRAM using different devices. The IC used for this project is XC3S700A, which consumes 0.033W of power and 7.054ns of time for execution.

Table 2. Power and Timing Report with the Device

Device	Power(W)	Timing(ns)
XC3S700A	0.033	7.054
XC3S50A	0.012	7.054
XC3S50AN	0.014	7.054
XC3S200A	0.023	7.054
XC3S200AN	0.027	7.054
XC3S400A	0.027	7.054
XC3S400AN	0.027	7.054
XC3S700AN	0.038	7.054
XC3S1400A	0.056	7.054
XC3S1400AN	0.064	7.054

5.2. The Hardware Implementation

The Laptop is interfaced with the development Spartan board FPGA for programming. Once the interfacing is done the programming file of the design is generated. The device is then configured so that the generated programming file can be successfully implemented on spartan board. Figure 20 shows the final output of the project which includes Xilinx's kit with headphones and mike and a soft audio codec, which includes Matlab platform. As shown in the figure we have interfaced the Spartan board with the laptop. The memory controller design will be present on the FPGA board. The DDR2 SDRAM memory device on which we are storing the data is also present on the same board. Once after connecting the mike , the audio data(voice) from the mike will be converted into digital data with the help of the Matlab code then the data will be written onto the memory device with the help of the memory controller. We can assign different switches present on the FPGA boards to control the read and write operation. As shown in the below figure 21 we can see we are making use of the switches to read and write the data onto the memory.

Fig.20. Hardware Result

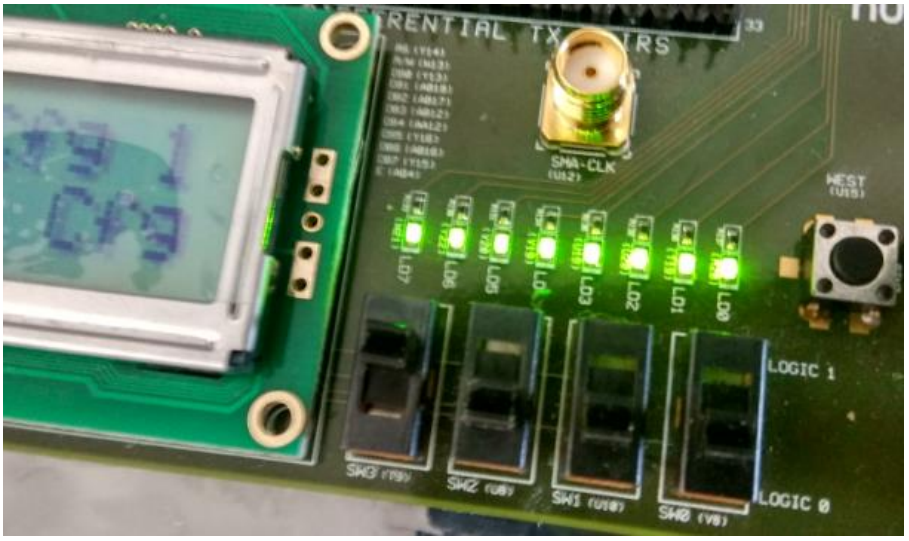

Fig.21. Input to FPGA

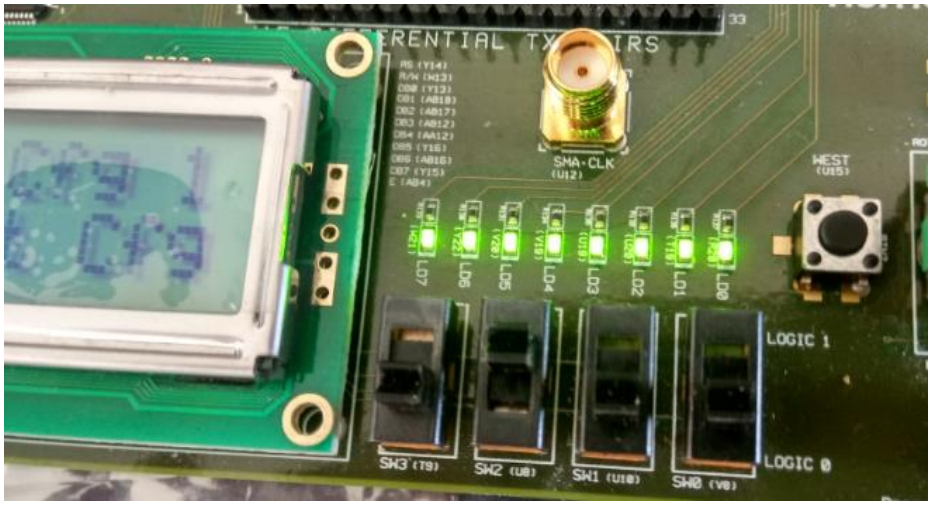

Fig.22. Output from FPGA

- Audio Plots

Sampling is a process in which we try to convert the continues time signal into a discrete time signal. In our design the continues time signal is the audio input which will be given as the input , then we have sampled the audio signal at different sampling rates such that we can compare the data being received at the receiver end at different sampling frequencies. As the sampling rate increases the quality also increase.

The above figures represent the audio plots with a sampling rate of 8000 and duration of 3 seconds. The samples are between 120 to 130 for the original audio, and the received audio has samples in the range of 120 to 140. These we can notice that some amount of noise has been added to the received data.

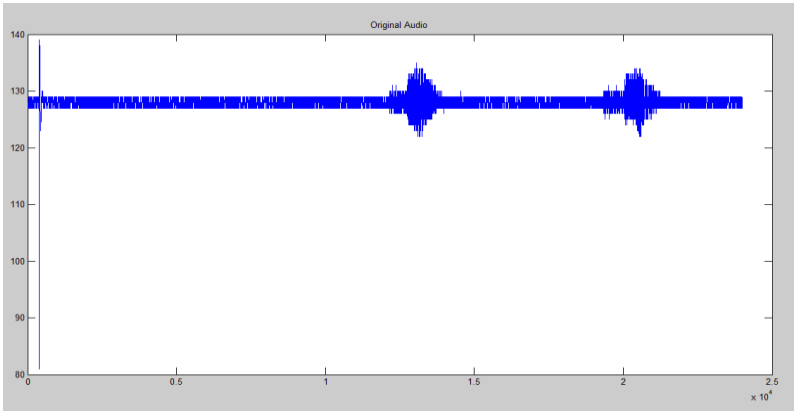

Fig.23. Original Audio with Sampling Rate 8000 & Duration 3s

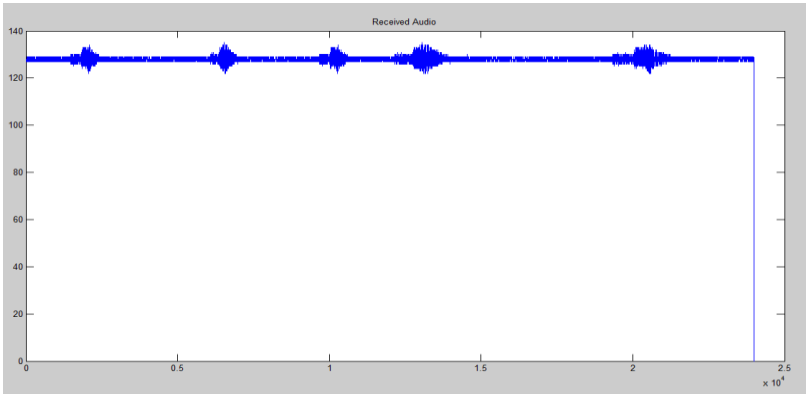

Fig.24. Received Audio with Sampling Rate 8000 & Duration 3s

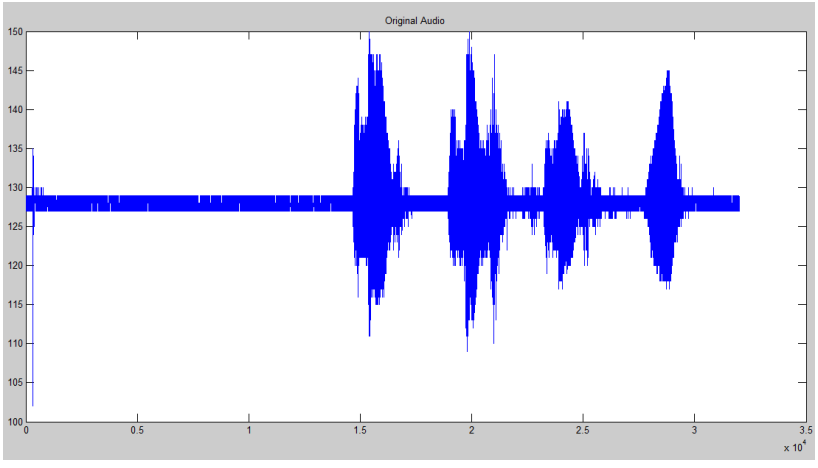

Fig.25. Original audio with sampling rate 8000 & duration 4s

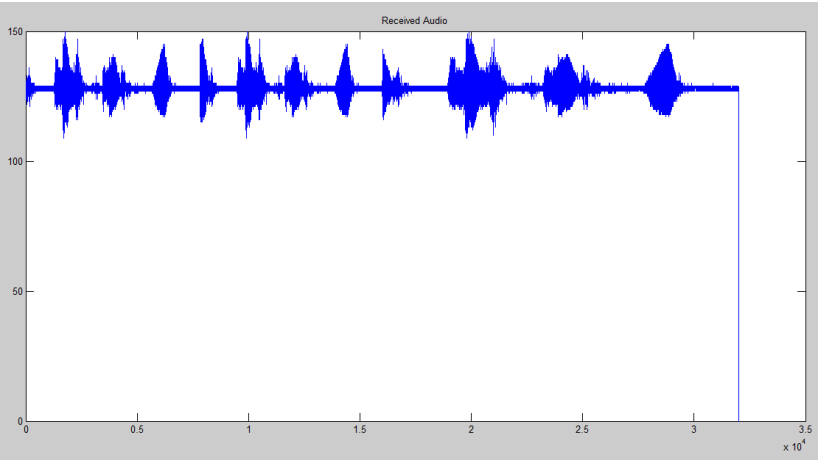

Fig.26. Received Audio with Sampling Rate 8000 & Duration 4s

The above figures represent the audio plots with a sampling rate of 8000 and duration of 4 seconds. The samples are between 125 to 130 for the original audio, and the received audio has samples in the range of 100 to 150. These we can notice that some amount of noise has been added to the received data.

Table 3. Write and Read Timings

Sampling frequency	Duration (Sec)	Write time (Min)	Read time (Sec)
8000	3	3.43	7.9
8000	4	3.56	8
10000	3	3.55	6.8
10000	4	3.54	10.4
12000	3	8.30	17.5
12000	4	8.37	18.2

The above table represents the time taken to write and read the audio data at different sampling rates and different durations. Since the data is being written onto the memory with the help of the Mat lab coding it will take longer time as shown in the above table as duration. If we can directly interface the mike and speaker to the FPGA board then the delay observed will be much less.

6. Conclusion and Future Scope

6.1. Conclusion

In this project, DDR2 SDRAM has been designed on Xilinx platform using Spartan-3a FPGA. Matlab is used for audio data encoding, and decoding using a soft processor and the audio data is stored on to the memory device with the help of the designed memory controller for storing the data. There are many types of research going on DDR2 SDRAM controller in the VLSI industry by the scientists nowadays so as to improve the efficiency of the device used in terms of the bandwidth utilization , latency of operation and the power consumption etc.

6.2. Future Scope

DDR2 SDRAM memory controller can be used in many other applications such as image processing, video processing, etc. We can optimize the memory controller in the area, power, latency, etc. Using Matlab platform we can obtain different voice samples by changing the sampling frequency and duration. We can try directly connecting the mike and speaker to the FPGA board and try to control the read and write operations.

References

[1] Suyog V Pande, Prashant D Bhirange 'Simulink based Low Power Mpeg-4 AAC Audio Encoder and Decoder' *IEEE ICCSP 2015 conference*.

[2] Design of a TLM NAND Flash Controller Model for Audio Real-Time Applications M. Gianfelici; M. Conti; M. Caldari; F. Ripa *2015 12th International Workshop on Intelligent Solutions in Embedded Systems (WISES)*.

[3] Optimized FPGA-based DDR2 SDRAM controller Jian Qituo; Liu Liansheng; Peng Yu; Liu Datong *2013 IEEE 11th International Conference on Electronic Measurement & Instruments*.

[4] An innovative design of the DDR/DDR2 SDRAM compatible controller Darshan Makam; H. V Jayashree *International Conference on Nanoscience, Engineering and Technology (ICONSET 2011)* Year: 2011.

[5] ZHUO L, DU G M, ZHANG D L, et al. Design and Implementation of DDR2 Wrapper for Cluster-Based MPSoC[C]. 2010 *International Conference on Anti-Counterfeiting, Security, and Identification*, 2010: 60-62.

[6] Benny Akesson, Kees Goossens and Markus Ringhofer published a paper 'A Predictable SDRAM Memory Controller.'

[7] www.micron.com

[8] https://en.wikipedia.org/wiki/AC%2797

[9] https://en.wikipedia.org/wiki/DDR2_SDRAM

[10] www-inst.eecs.berkeley.edu

[11] https://www.xilinx.com/support/documentation/sw_manuals/xilinx11/edk_ctt.pdf

[12] https://www.micron.com/~/media/documents/products/datasheet/.../512mbddr2.df

[13] Prof. Arun.S.Tigadi1, Padmashree. G *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering* (An ISO 3297: 2007 Certified Organization)Vol. 5, Issue 8, August 2016 Design and Implementation of Memory Controller for Real-Time Video Acquisition using DDR3 SDRAM

[14] Mayankagr.in/images/matlab_tutorial.pdf

[15] Mani Shankar Anand, Barjeev Tyagi,"Design and Implementation of Fuzzy Controller on FPGA", *IJISA*, vol.4, no.10, pp.35-42, 2012.

[16] Sachin Raghav, Rinkesh Mittal,"Implementation of Fast and Efficient Mac Unit on FPGA", *International Journal of Mathematical Sciences and Computing(IJMSC)*, Vol.2, No.4, pp.24-33, 2016.DOI: 10.5815/ijmsc.2016.04.03

[17] Shiraz Afzal, Farrukh hafeez, Muhammad Ovais Akhter,"Single Chip Embedded System Solution: Efficient Resource Utilization by Interfacing LCD through Softcore Processor in Xilinx FPGA", *IJIEEB*, vol.7, no.6, pp.23-27, 2015. DOI: 10.5815/ijieeb.2015.06.04

Authors' Profiles

Arun S. Tigadi, Assistant professor Department of E and C, K.L.E DR. M.S. Sheshgiri College of Engineering and Technology. Have a working experience of 9 years in the department of E and C. Received my U.G Degree in E&C from S.D.M CET Dharwad in the year 2006 and P.G Degree in VLSI Design and Embedded systems from K.L.E CET Belagavi in the year 2008. Fields of interest are Low power VLSI design, FPGA Design, Memory controllers, arbiters, multiport memory design, Real-time system design and Operating systems. Published seven international journal papers and presented three papers in international conferences.

Dr. Hansraj Guhilot, held many academic and R&D positions over a career span of 30 years, currently working as the Principal, K. C. College of engineering and management studies and research. He worked as Dean (R&D) and Professor of EC at KLE Dr. M. S.Sheshgiri College of Engineering and Technology, Belgaum, Karnataka. He has teaching experience spanning 28 years with a Ph.D. in Electronics, having thesis titled "Design and Development of CMOS Mixed-Mode Integrated circuit for Chloroplast Measurement" and Research work published in IEEE Sensors Journal. He is an IEEE Technical Paper Reviewer at IEEE International Conference on Recent Trends in Information, Telecommunication, and Computing (ITC), 2010. He is a member of Entrepreneur Development Cell (EDC) in Visveswaraya Technological University (VTU), Belgaum. He is a subject expert in CMOS VLSI, Edusat Program, VTU, and Belgaum. He has published 36 papers, delivered ten invited technical talks and is awarded with one US Patent and nine international patents. Worked as Director (R&D), Paradigm Industries Inc. USA and Consultant for N&N Allied Energy Services Inc. USA.

How to cite this paper: Arun Tigadi, Hansraj Guhilot,"Design and Implementation of a DDR2 SDRAM Controller for Audio Data on a Reconfigurable Platform", International Journal of Engineering and Manufacturing(IJEM), Vol.8, No.5, pp.32-48, 2018.DOI: 10.5815/ijem.2018.05.04