DESIGN OF FPGA BASED EPROM EMULATOR FOR 16 BIT MICROPROCESSOR DEVELOPMENT SYSTEM

Noohul Basheer Zain Ali
Dept. of Electrical & Electronic Engineering,
Universiti Teknologi PETRONAS
31750 Tronoh, Perak Darul Ridzuan, Malaysia,
email:noohul@alumni.rpi.edu

Lai Soon Chong
Dept. of Electrical & Electronic Engineering,
Universiti Teknologi PETRONAS
31750 Tronoh, Perak Darul Ridzuan, Malaysia,
email:laisoonchong@hotmail.com

Yap Voon Vooi
Middlesex University, Bounds Green
Road, London N11 2NQ
email: vooiyap@yahoo.com

ABSTRACT

The ultimate goal for this project is to design an EPROM Emulator for MOTOROLA 68000 Embedded System. Canonical method applied in this project is to design and test all the way through starting with discrete component design and later VHDL based design on the FPGA.

The main purpose of EPROM Emulator is to provide memory to a Microprocessor Embedded System (Circuit) such that no memory devices are required to be fitted into the circuitry. Heart of the EPROM Emulator is the Static RAM used to emulate, the target system or circuitry memory (EPROM). EPROM Emulator is used in assisting the System Software (Firmware) development for the targeted circuit. It allows the Static RAM being shared between the development platform and the target circuit exclusively. Allowing the memory device sharing, the EPROM Emulator can accelerate the System Software development process by eliminating time consuming process such as programming and erasing the EPROM. Implementing the design on FPGA, allows more flexible and portable design. This as such the circuitry of the EPROM Emulator can easily be modified to suit any type of microprocessor not only the Motorola 68000 family.

Design implementation proves that classical method of circuit design is rigid and more painstaking, the VHDL design proves to be more systematic and debugging of the design can be done at every stage of the design flow.

Keywords: FPGA, 68000, Microprocessor, EPROM Emulator, Integrated Circuit, System Software, Firmware.

1.0 INTRODUCTION

Emulate an EPROM of circuit is essentially provide an alternative memory for the targeted circuit. In another term, it is to design a circuit that if connected to the target circuit will be detected as EPROM for the circuit. This however can only be achieved by using device from the same family group – the Random Access Memory (RAM). The configuration of the EPROM Emulator can be simplified by the block diagram, as follow:

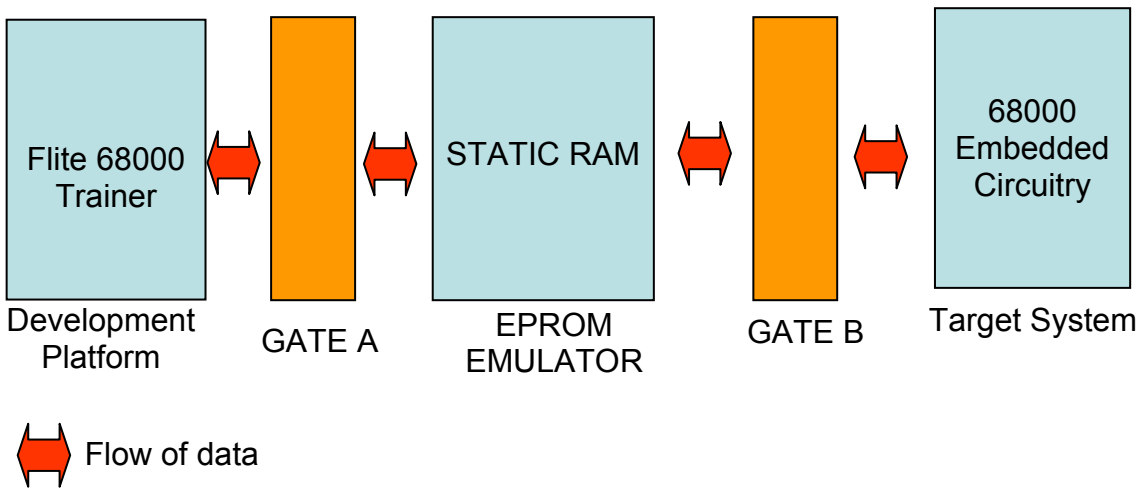


Figure 1.1 EPROM Emulator general block diagram

Since, the design is purely digital, implementation of the design is best done on a FPGA using VHDL. However, the discrete component circuit also can achieve similar outcome with extra effort. This is due to large amount wire connection between the devices, circuit assembly is painstaking.

Discrete component design circuit is complex and hard to design, however using VHDL to describe the behavior of the circuit, design can be implemented quite easily. Using VHDL for circuit description, major process used to be necessary in discrete component circuit design such as Boolean equation derivation can be eliminated. Hence this method will accelerate and help to produce more reliable device.

By having the design in VHDL codes, the circuit configuration can be modifies and amends much easier. This feature can allow the circuits initially design for 68000 family microprocessor can easily be modified other microprocessor system such as the INTEL X86 family microprocessors.

2.0 THEORY

Knowing the fact that software can accelerate and open design to unlimited possibilities, hence designing of software become more preferable compared to hardwired hardware. The development process by itself has to be fast an accurate. The preliminary idea to design an EPROM Emulator is from the fact that design of software need to be done fast and in the most non-time consuming way. Another fact is that to understand that what a microprocessor requires to operate is instruction (software). This regardless of medium the instruction is stored on. How the knowledge and application of this storage medium being applied in the design of the FPGA based EPROM Emulator will be further discussed in the following section.

2.1 Memory Devices (ROM-Read Only Memory and RAM-Random Access Memory)

Basically the memory devices can be comprised of two major types: i.e. the Read Only Memory and the Random Access Memory. Each type has it own characteristic that contribute significantly to the embedded system development. The non-volatile Read Only Memory is usually used to store system software but it has hassle and limitation in storing the information in it. Programming the ROM itself is time consuming and erasing the information in it is even more time consuming. RAM is fast read and writes access however it need to connected to supply all the time to maintain the information within it.

2.2 Gating Data Flow

Although the EPROM Emulator is transparent to both development system and the target circuit, they can only one side access allowed at a time. This requires the EPROM Emulator to be able to prohibit certain access while allowing the other access to the memory device.

2.3 Memory Address Decoding

Modern microprocessor is designed such that it can address much more storage capacity than it is usually supplied with. The 68000 16bit microprocessor for example can accommodate $2^{24} = 16$ Megabytes of address location. The EPROM Emulator also must be equipped with logic to let the microprocessor on both development platform and target circuitry to access the valid address range. Such circuit is known as the Memory Address Decoder

Memory address decoder also allows more then one piece of the memory device to be connected to the microprocessor hence can accommodate more storage space.

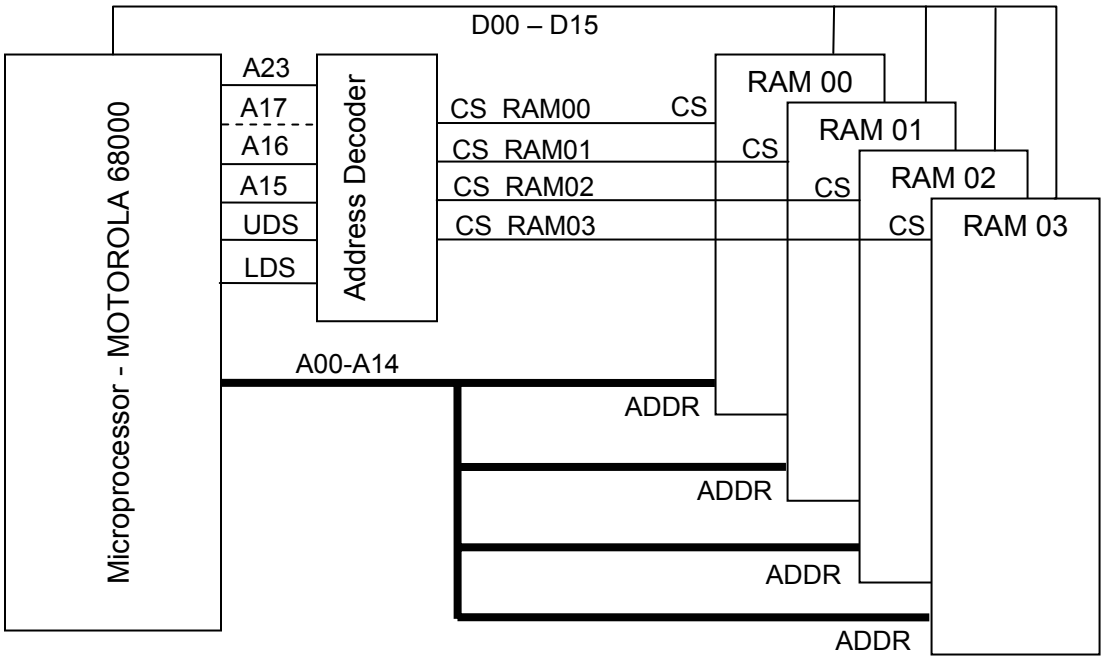


Figure 2.1 MOTOROLA 68000 Memory Interface Schematic

3. METHODOLOGY AND PROCEDURE

The EPROM Emulator was design in both discrete component and FPGA. By applying this design strategy, the performance and the level of difficulties of design can be compared among these two design method. The other advantage of using the discrete component design before implementing VHDL version of the design is to be able to visualize the design much clearer.

3.1 Discrete Hardware Design

The basic for the EPROM Emulator is to derive from the MOTOROLA 68000 microprocessor RAM interface circuit. The additional feature added is that the circuit must be able to restrict and allow access to the RAM depending on its operating mode. Data and signal are gated using the 8bit bidirectional transceiver. Since the MOTOROLA 68000 used is operating at 10MHz, all components have to be CMOS, to be able to cater with the circuit high speed and signal logic level.

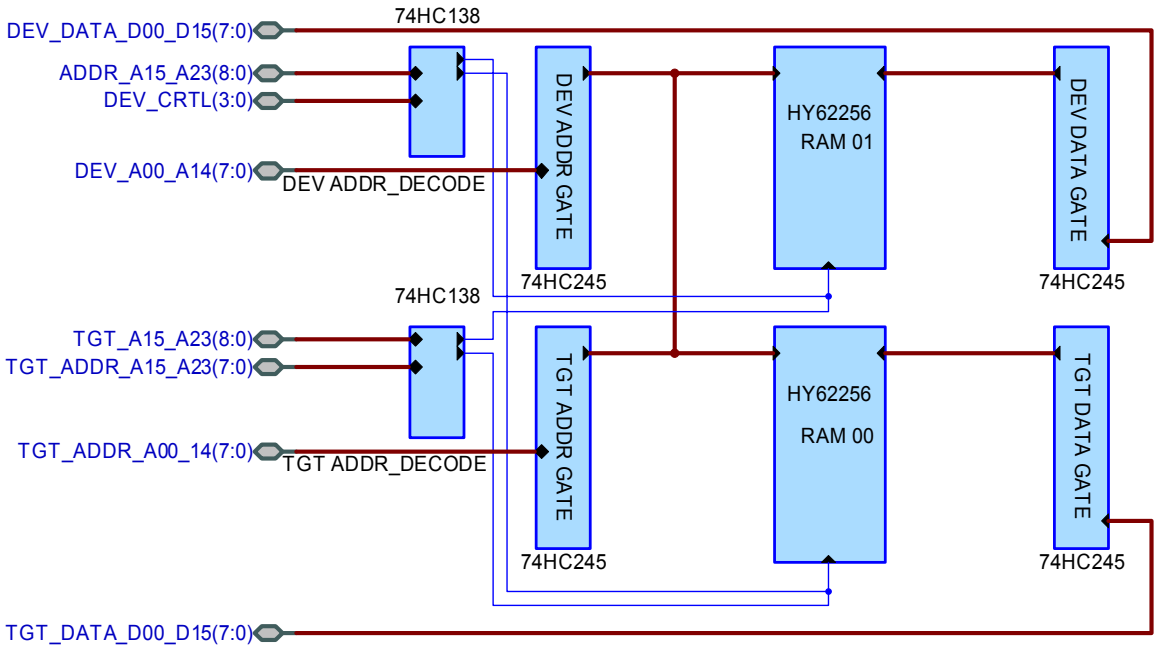


Figure 3.1 Discrete Component Hardware Implementation Block Diagram

3.2 VHDL Design Implementation

Since the design was derived in hardware discrete component design, the very same design can be transform to VHDL format by describing all the major component of the circuit into separate VHDL module which will later combined to produce a whole system. The VHDL design implement requires several stages before the design can be downloaded in FPGA device. These design stages are: *Register Transfer Level*, *Gate Level* and finally *Design Implementation*.

From the discrete component design, the major component need to be derived in VHDL is as tabulated follows:

Module Name	Entity	Function
`WriteControl.vhd`	`WriteControl`	To control the write enable and output enable of the memory chip connected to EPROM Emulator.
`Selector2InputWithControl.vhd`	`Selector2`	Select one of two (2) input depend on the operation mode of the EPROM Emulator and route it to the desired output.
`Selector4InputWithControl.vhd`	`Selector4`	Select one of four (4) input depend on the operation mode of the EPROM Emulator and route it to the desired output.
`Selector8InputWithControl.vhd`	`Selector8`	Select one of eight (8) inputs depend on the operation mode of the EPROM Emulator and route it to the desired output.

Table 3.1 The Basic Module of the VHDL design

The system is built by connecting and instances of these small modules. Combination and instances the small modules is known as the Top Level design. There are many advantages of using VHDL design compared to the discrete component design. Each and every occurrence of component will only requires instances of the component, hence the component can be used infinite number of times in the design without constrain of price and component count (subject to the FPGA device capacity). Using ALDEC VHDL or Xilinx Webpack ISE, the design can be graphically edited or coded. The final design view is as attached in the appendix.

3.2.1 VHDL design simulation and verification (Register Transfer Level)

Before the design is implemented, the VHDL description of the system can be simulated. Output in term of waveform or logic state can be monitor on every connection and component of the whole design or each module. This facility will enhance the design flow allowing designer to detect and eliminate the bug in the circuit at earlier stage.

3.2.2 Design Implementation on FPGA

After all modules are combined, verified and simulated, the whole system needs to be downloaded into the FPGA device. The target FPGA device used for this project is Xilinx SPARTAN2. This device can sustain the circuit requires system clock up to 100MHz way beyond the requirement of the system that is 10MHz only. The whole process of compiling, gate level synthesis, place and route are done using Xilinx Webpack ISE (version 3.2). Timing simulation is not possible using this software however; device timing result can be generated. This result is checked against the critical timing MOTOROLA 68000 memory read and write access cycle to ensure the final design is compatible with the existing MOTOROLA 68000 system. The complete design is finally downloaded into SPARTAN2 FPGA using third party download software.

4.0 RESULT AND DISCUSSION

This section will discuss the result and advantage of VHDL design using FPGA compared to the implementation of design using discrete hardware component.

4.1 VHDL Simulation Observation

The most important design criteria in this project to obtain and produce signal with correct timing and logic level. This is necessary because the prototype need to be interfaced with existing Flite 68000 Trainer operating at 10MHz. Any response involved critical timing slower than the existing system speed will cause design fault. There are two main operations involving the circuit. These are the microprocessor WRITE and READ cycle. Both operations are simulated using ALDEC Active HDL 4.2

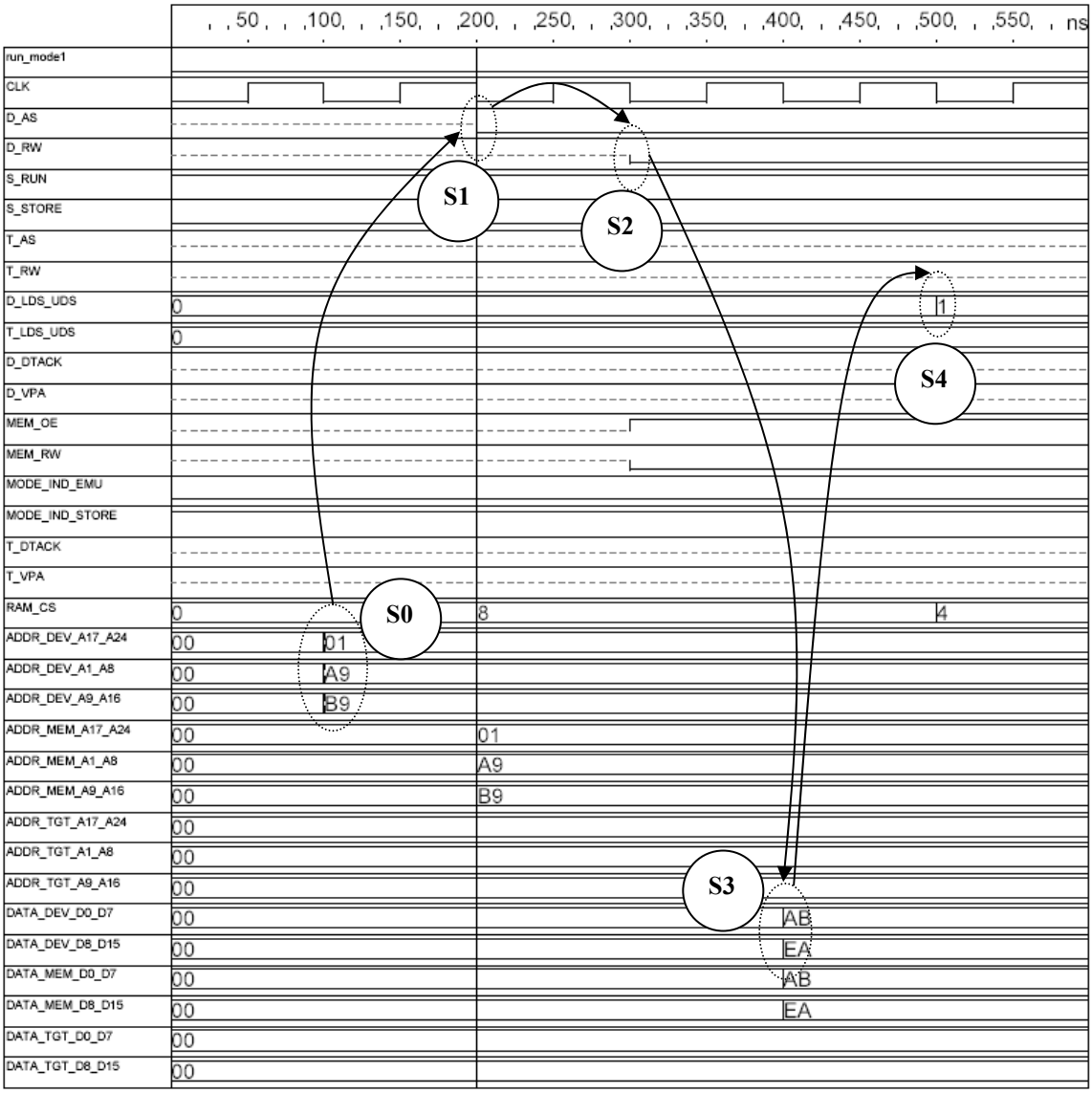


Figure 4.1 EPROM Emulator WRITE cycle simulation

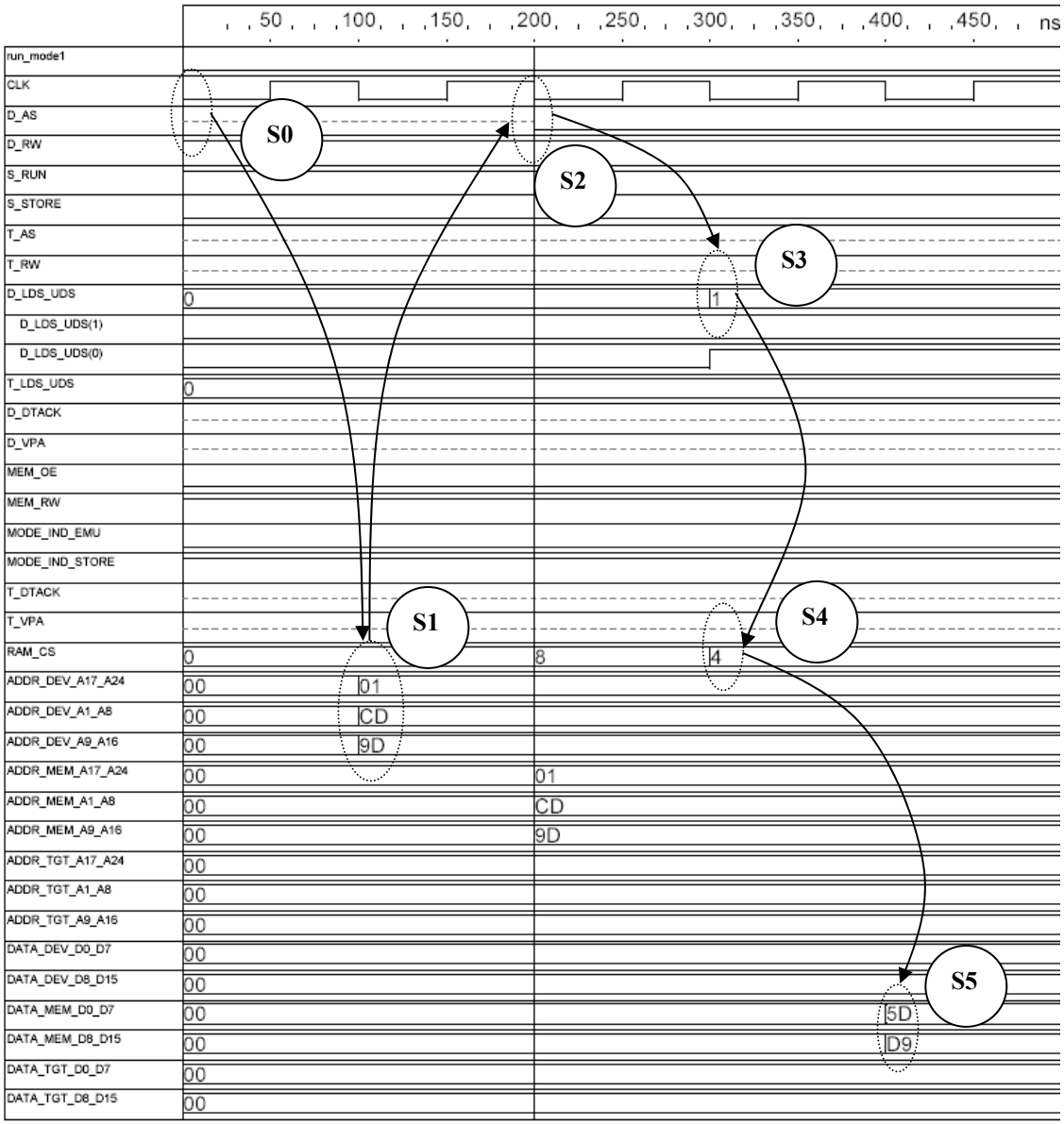


Figure 4.2 EPROM Emulator READ cycle simulation

The waveform obtained from the simulation is compared again the WRITE and READ cycles of the 68000 microprocessor. It is founded that both operation produces waveform that is compatible with the MOTOROLA 68000 WRITE and READ operation.

Another thing to observe when using FPGA is the capability of the target device to satisfy the number of required gate by the design. This information is available in the implementation place and route report

```
Device utilization summary:
Number of External GCLKIOBs 2 out of 4 50%
Number of External IOBs 140 out of 140 100%
Number of SLICEs 119 out of 2352 5%
Number of GCLKs 2 out of 4 50%
```

Figure 4.2 Xilinx Webpack ISE 3.2 Place and Route Report.

The report shows the design fit well inside the SPARTAN2 FPGA device perhaps it is too small for the device.

4.2 Advantage of VHDL design compared to discrete component design

There several factors that can be compared against the two method of design. These advantages and drawback are tabulated as follows

Feature	Discrete	VHDL
Development Life Cycle	Starts will derivation of design using schematics. Complex design is then transferred to breadboard or prototype board for testing. This step is highly susceptive to error. Final stage will be fitting of each component and wiring it.	Design stage starts with describing basic building block modules of the design. The next stage is to combine the design under Top Level design to compilation and synthesis. Downloading design to FPGA demo board will essentially produce desired circuit. Each of the design steps allows designer to modify design without painstaking effort.
Cost	The cost of prototyping is directly proportional with the component count of the design.	Any type of digital component can be modeled using VHDL, hence there will be only single fixed cost for the design.
Flexibility	Discrete component design is rigid although certain part can be program i.e. GAL and PAL device however overall design cannot be modified easily once there are in place	VHDL based design on FPGA not only allow designer to modify design in the design stage but also the complete on board hardware. This is due to fully programmable FPGA device which output of each I/O pin can be assigned to the required I/O signal.

Table 4.1 Comparison between discrete component and FPGA design

5.0 CONCLUSION AND RECOMMENDATION

Implementation of the EPROM Emulator using VHDL have yield result that can be verified right from the beginning of the design stage. Using FPGA, the VHDL design is not only limited to MOTOROLA 68000 microprocessor but it can easily modified to match other type of 16 bit microprocessor i.e INTEL 80286 and 80386.

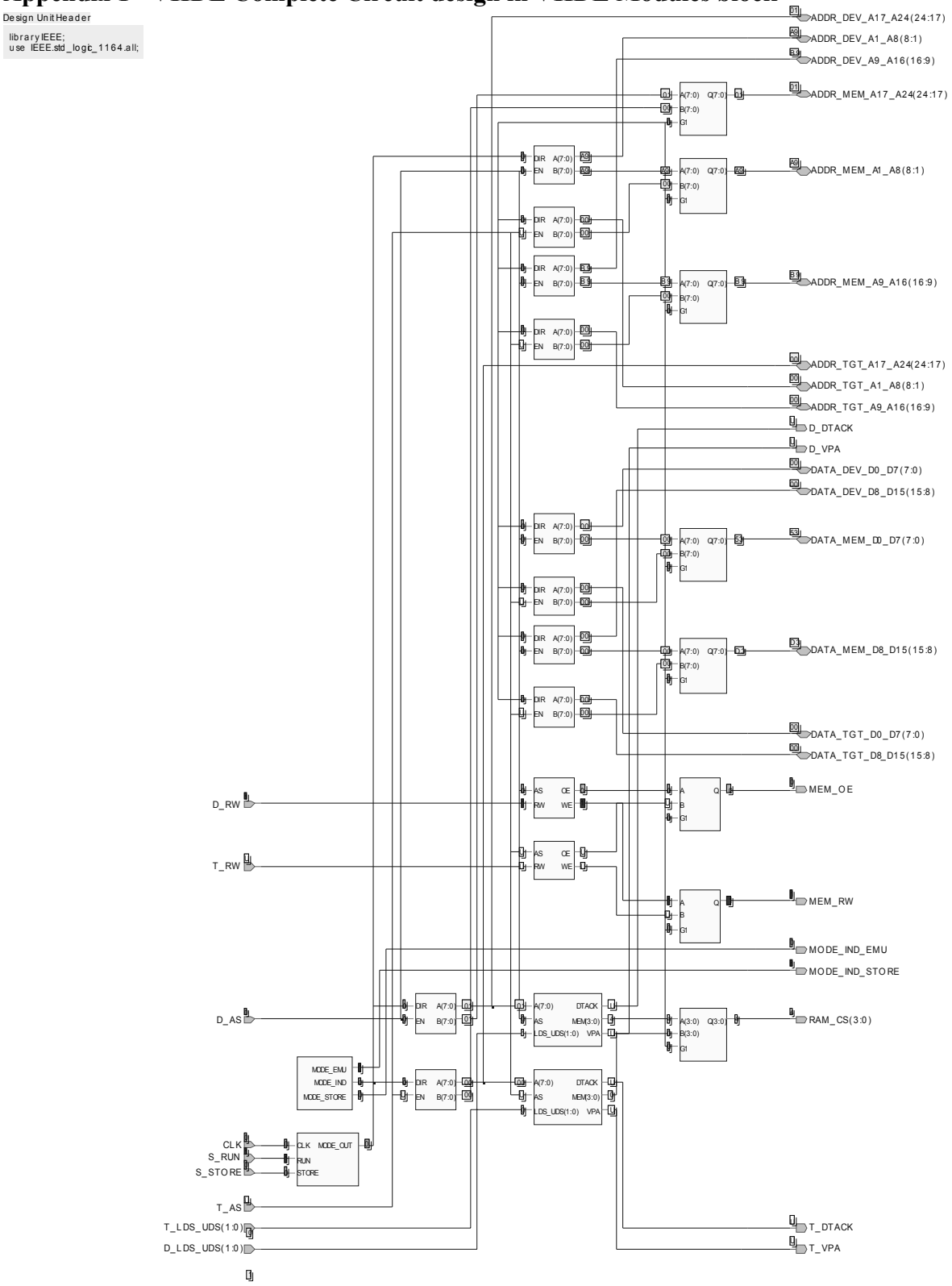
EPROM Emulator is a very handy useful tool that can assist Firmware or System Software development process, by having EPROM Emulator, the code can be tested in the most time efficient manner. Flash ROM although can equally competitive but it cannot be connected to the development platform directly. Adding extra functionality to the EPROM Emulator, it also can be used to emulate certain part of the circuit that is not yet available to the System Software developer.

The implementation of the EPROM Emulator using VHDL has been a great success in the short most time frames. It also allows the exploration of the borderless design and efficient design stage.

6.0 REFERENCES

1. J. Mirkowski, M. kapustka, Z.Skowronski, A. Biniszkiewicz (1988) Active-VHDL Series Book #2, Aldec, Inc: Henderson
2. Aldec Inc (2001) VHDL Reference Guide. Adlec Inc: Handerson
3. Trans-Dist Engineering (2001) Introduction to VHDL with Active-HDL, Trans-Dist Engineering Sdn Bhd: Putra Jaya
4. Trans-Dist Engineering (2001) FPGA Development System Workshop – Lab Exercise, Trans-Dist Engineering Sdn Bhd: Putra Jaya
5. Motorola (1985) MC68000 16/32 Bit Microprocessor, Motorola Inc: East Kilbride, Scotland
6. R.F. Coates (1997) The Flight 68000-MKII Training System, Flight Electronic International Limited: Hampshire, UK
7. Douglas L. Perry (1999) VHDL – Third Edition, McGraw-Hill, New York
8. Alan Clements (1997) Microprocessor System Design, 68000 Hardware, Software and Interfacing, PWS Publishing Company: Boston
9. James L. Antonakos (1999) The 68000 Microprocessor, Hardware and Software Principles and Applications, Prentice Hall: new Jersey
10. Andrew Rusthon (1998) VHDL For Logic Synthesis 2nd Ed., John Willey & Sons Ltd: England
11. Zainalahedin Nawawi (1998) The definitive guide to VHDL 2nd Ed, Analysis and Modeling of Digital System: McGraw Hill: Singapore
12. Sudhakar YalamChilli (1998) VHDL Starter Guide, Prentice Hall Publishing: New Jersey
13. BurCHED Electronic Australia (2001) B3-Spartan2+ Data Sheet, B3-SPARTAN2+ Quickstart Guide www.burched.com.au (last accessed 6/6/2002)
14. Spartan-II 2.5V FPGA Family (2000) www.xilinix.com (last accessed 6/6/2002)

Appendix 1 - VHDL Complete Circuit design in VHDL Modules block



Appendix 2 – Discrete Design Main Memory Interface Circuit