International Journal of Electronics, Communication & Soft Computing Science and Engineering (IJECSCSE)
Volume 1, Issue 1

# A High Throughput SRAM Implementation in Xilinx

**Akshay P. Dhande**

***Abstract- Memory is most important sub-system of digital circuit which store data/information. In this paper review of different type of existing memories with their type technologies and characteristics presented and the result for 8Mb low power SRAM implementation on FPGA Spartan II pro kit shown. Because today the complexity of binary digital hardware system is steadily increasing and no single type had all the characteristics of the ideal memory hence new type of memory and their way of implementation is investigated in this paper. The objective of this paper is to use Verilog Hardware Description Language to produce Verilog code for proper operation of SRAM. The synthesis and Simulation tool ISE9.1i is used to map the design in to targeted device. Validation of the result and timing simulation are done using XST tool of XILINX9.1i.***

***Keyword - SRAM, FPGA, Xilinx9.1i, Verilog.***

## I. INTRODUCTION

With the rapid growth of digital circuits in recent years, the need for high-speed data transmission has been increased. The VLSI industry faces the problem of providing the technology that can be able to support a high data capacity and to with low power. [2] Many memories are available but SRAM has gained much attention for different reasons. SRAM has been recognized as an outstanding memory for high-speed data access. Standby power of SRAM memories is very low in spite of high density of transistors. SRAM cells have high noise immunity due to larger noise margins, and have ability to operate at lower power. [5]

This work deals with the design and analysis of 8Mb Static Random Access Memory (SRAM), focusing on optimizing power and delay to achieve high throughput for RF applications. The SRAM access path is split into two portions: from address input to word line rise (the row decoder) and from word line rise to data output (the read data path). Techniques to optimize both of these paths are investigated and implemented. In this work the existing SRAM architectures are analyzed and innovative SRAM structure with a throughput of up to 12.6 Gbit/s at a clock frequency of 168 MHz is tried to achieve.

The proposed SRAM with data, address and control lines is planned to design and code in VeriLog, simulate and synthesize using Xilinx ISE 9.1i and finally implement using FPGA Spartan II Pro kit. The characterization is done on single bit SRAM cell to determine the cell characteristics in static mode. The simulated results obtained after performing the characterization of a single bit SRAM cell is tested using FPGA Spartan II pro kit.

## II. ARCHITECTURES OF SRAM

### *A. Synchronous Sram and Its Working*

As computer system clocks increased, the demand for very fast SRAMs necessitated variations on the standard asynchronous fast SRAM. The result was the synchronous SRAM (SSRAM).Synchronous SRAMs have their read or write cycles synchronized with the microprocessor clock and therefore can be used in very high-speed applications. An important application for synchronous SRAMs is cache SRAM used in Pentium- or Power-based PCs and workstations. SSRAMs typically have a 32 bit output configuration while standard SRAMs have typically a 8 bit output configuration.

### *B. Asynchronous Sram And Its Working*

Figure 1 shows a typical functional block diagram.The memory is managed by three control signals. One signal is the chip select (CS) or chip enable (CE) that selects or de-selects the chip. When the chip is de-selected, the part is in stand-by mode (minimum current consumption) and the outputs are in a high impedance state. Another signal is the output enable (OE) that controls the outputs (valid data or high impedance). Third is the write enable (WE) thatselects read or write cycles.[3]

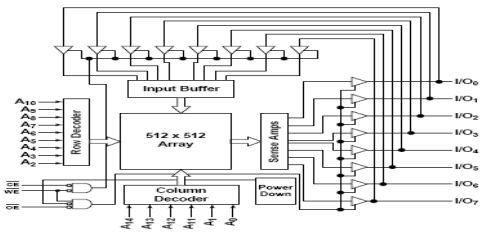


**Figure 1 Typical SRAM**

By considaring aboue two architectures we have selected asyncronous architectue for our coding.

## III. DESIGN CONCEPTS

### *A Circuit Partitioning*

A conventional memory has a single core with input/output block, control and the decoding circuit builds around it. This arrangement works well for a small memory, but as the memory size increases, the load (capacitive and the resistive) on the word lines and bit lines also increases which in turn reduces the speed of the memory. If by any means we can reduce the resistive and the capacitive load, we can control the delay. The popular approach is to divide the core into number

of pages and banks, this approach is good for both power and speed as the load on bitlines and wordlines is reduced, but it is not an area efficient solution as we need to have local wordlines and decoders for each page which increases the area. A very simple approach is to just split the core in two parts (see fig. 2) and use the same wordline driver for it This way there will be two paths for the driver and each of the paths will see half of the capacitive and resistive loads as compared to the case in which there is only a single core. [1,3]

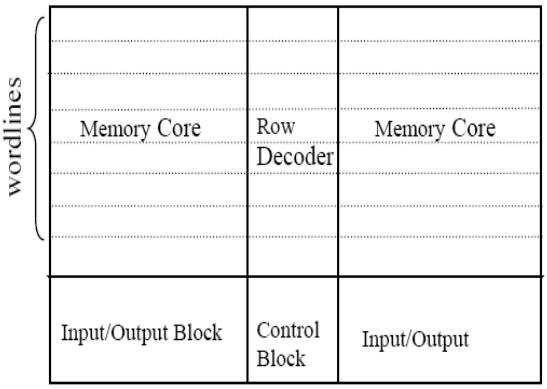


**Figure 2 Block diagram of a memory with a divided core and having the same wordline driver**

*B Dual Vth Usage*

In order to gain in speed, reducing the threshold voltage is a very effective technique. It is also advantageous at low voltage operations where we gain both in terms of speed and power. We have used this technique to improve the speed of our memory. Excluding the memory cells, the dummy row and the sense amplifier, rest all of the digital logic has been converted to low Vth transistors. The memory cells have been excluded since being very small transistors, the leakage current will increase whereas the gain in speed is almost negligible.

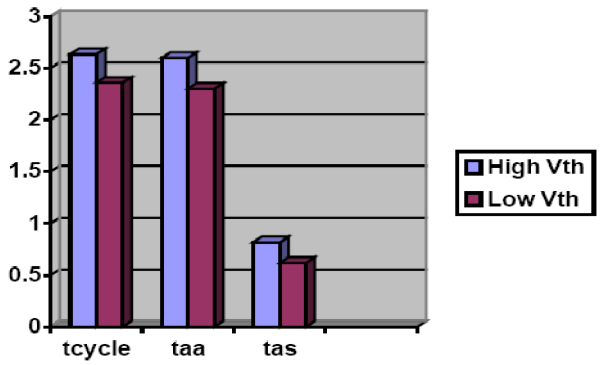


**Figure 3 Comparison between Low Vth and High Vth memory**

The control block, the decoders and I/O blocks are all in low Vth, whereas the memory cells, the dummy column and dummy row along with the sense amplifier are in high Vth. Having this kind of configuration has helped us in gaining in speed and also reducing the dynamic power consumption by a considerable amount. Figure 3 shows the comparison between a memory with high Vth transistors and low Vth transistors in the memory core periphery. All the timings shown in the said figure are the worst case timings for a 8192x16 memory size.[1]

*C DWL Architecture*

During an access to some row, the word line activates all the cells in that row and the desired sub word is accessed via the column multiplexers. This arrangement has two drawbacks for macros that have a very large number of columns: the word line RC delay grows as the square of the number of cells in the row, and bitline power grows linearly with the number of columns. Both these drawbacks can be overcome by further sub dividing the macros into smaller blocks of cells using the Divided Word Line (DWL) technique first proposed by Yoshimoto.

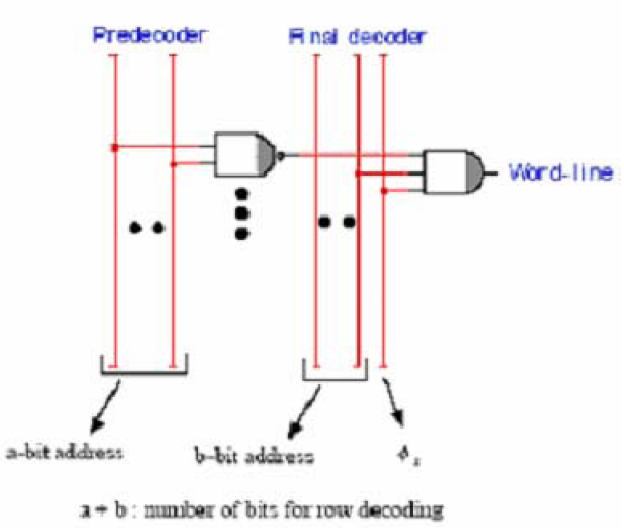


**Figure 4 Basic Row Decoder**

In the DWL technique the long word line of a conventional array is broken up into k sections, with each section activated independently thus reducing the word line length by k and hence reducing its RC delay by K2. The row selection is now done in two stages, first a global word line is activated which is then transmitted into the desired block by a block select signal to activate the desired local word line. Since the local word line is shorter, it has a lower RC delay. Before Divided Word Line (DWL) technique power is 20.423mw after Divided Word Line (DWL) technique power is reduced to 15.267mw.[3]

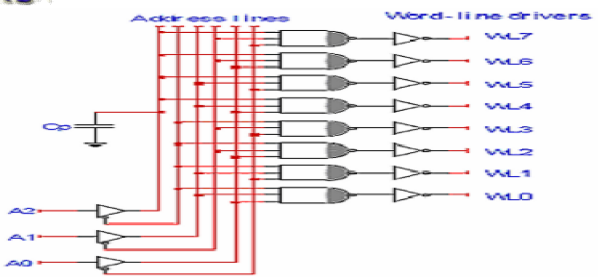


**Figure 5 Two Stage Decoder Architecture**

*D Low Power Decoders*

However, the normal decoder built using logic gates has the following drawbacks. The main problem is that the decoder will require a very large number of transistors. The capacitance associated with the long runs of wires and high gate input count will add to long delays. The address inputs will also have to be buffered to drive this huge capacitance load. The layout will become unnecessarily complex and so more time-consuming. Another problem is that the power consumption of such a decoder will be very high due to the large number of gates. SRAM chips are important components of embedded mobile systems, which generally run on batteries. To overcome these problems, we have used a dynamic NOR decoder. This structure reduces the number of transistors by half. It also increases the speed of the decoder and makes the layout simple and less time-consuming. Before low power decoders power is 15.267mw after using of low power decoder's power is reduced to 12mw [3]

## IV. Design Environment

We had used Verilog Hardware Description Language to produce Verilog code for proper operation and optimization of SRAM.[6] The synthesis & Simulation tool utilized is ISE9.1i to map the design to targeted device[8]. Validation of the result and timing simulation are also using XST tool of XILINX-9.1i. ASIC design approach needs sophisticated facilities and technical skill in designing digital circuit hence we have use FPGA design approach comparatively easy to implement and which produce standard result. We have follow Bottom-up design approach. In this method each design is performed at the gate level using the standard gates.

## V. Synthesis

Synthesis is the process of generating RTL view of the HDL design after synthesis the generated RTL View of the SRAM is as shown in fig.6

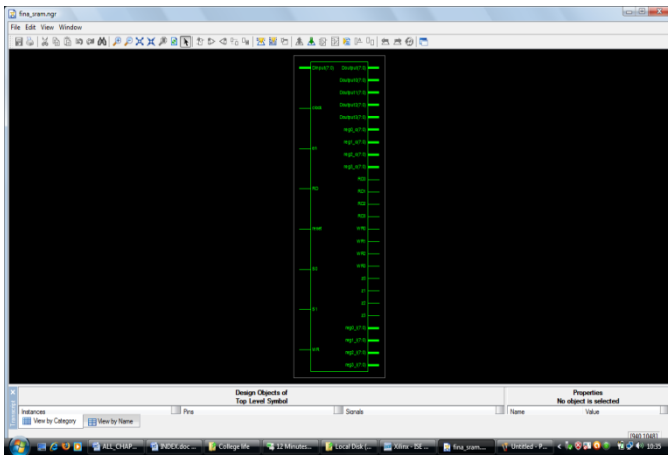

**Figure 6 Top Module Of SRAM**

After doing the synthesis we got the result as below

*A Advanced HDL Synthesis Report Summery*

**Table 1 Synthesis Report**

| | | |
|---|---|---|
| Top Level Output File Name : | final_sram | |
| Output Format | : | NGC |
| Optimization Goal | : | Speed |
| Keep Hierarchy | : | NO |
| Design Statistics | | |
| # IOs | : | 131 |
| Cell Usage: | | |
| # INV | : | 1 |
| # LUT3 | : | 12 |
| # LUT4 | : | 16 |
| # Flip-flops/Latches | : | 96 |
| # FDR_1 | : | 96 |
| # Clock Buffers | : | 1 |
| # BUFGP | : | 1 |
| # IO Buffers | : | 130 |
| # IBUF | : | 14 |

*B Device Utilization Summary:*

**Table 2 Device Utilization Summary**

| | |
|---|---|
| Selected Device: | 2s100pq208-5 |
| Number of Slices: | 19 out of 1200 1% |
| Number of 4 input LUTs: | 29 out of 2400 1% |
| Number of IOs: | 131 |
| Number of bonded IOBs: | 131outof 140 93% |
| IOB Flip Flops: | 96 |
| Number of GCLKs: | 1 out of 4 25% |

## VI. Simulation Result

*A Write Operation :*

When ce=1 , rw=0
i0 – i7 Inputs to i/p Buffers
d0 – d7 Outputs of i/p Buffers

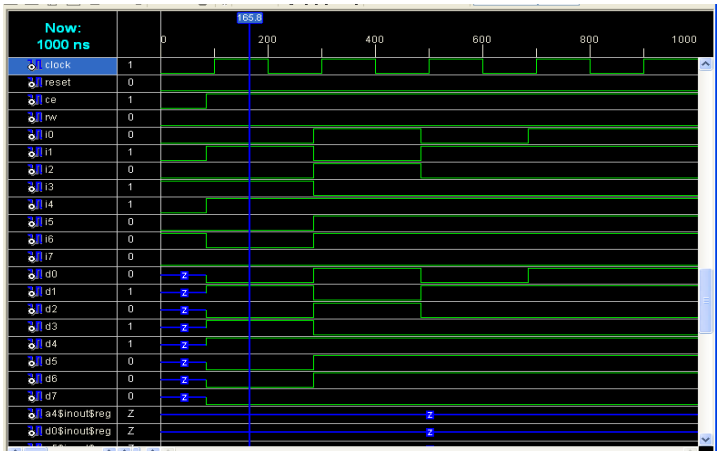


**Figure 7 Write Operation**

*B Read Operation:*

When ce=1, rw= 0
i0 – i7 Inputs to i/p Buffers
q0- q7 Outputs of Register

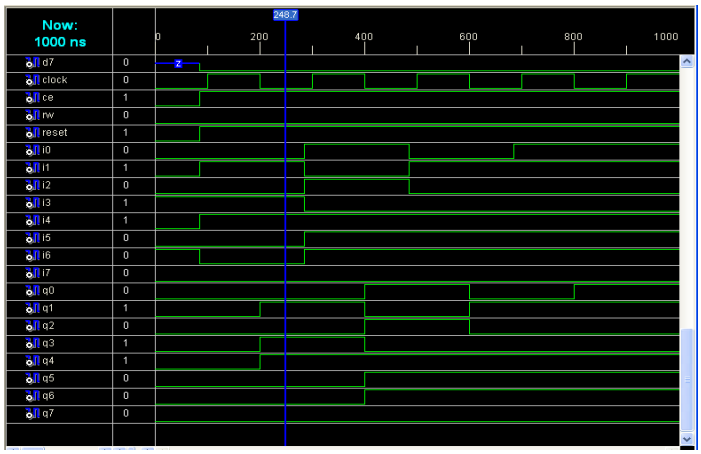


**Figure 8 Read Operation**

## VII. Conclusion

This paper work is dedicated to SRAM implementation and addressing of the critical issues in the designing of low power static RAM along with the design techniques used to overcome them. In this work techniques to optimize both of these paths are investigated and implemented. In this paper, all possible technique for SRAM implementation are compared and XILINXs is found to be golden suitable for SRAM implementation using verilog. With appropriate positioning of memory blocks and the use of proper verilog constructs, we achieve a high speed static RAM that will not dissipate too much power. We can say that our design is best suitable for advance application like networking etc.

## References

[1] Shobha Singh, Shamsi Azmi "Architecture and Design of a High Performance SRAM for SOC Design" Proceedings of the 15th International Conference on VLSI Design © 2002 IEEE
[2] Erik Jan Marinissen & Betty Prince "Challenges in Embedded Memory Design & Test" Proceedings of the 15th International Conference on VLSI Design © 2002 IEEE
[3] Sreerama Reddy G M & P Chandrasekhar Reddy "Design and Implementation of 8K-bits Low Power SRAM in 180nm Technology" Proceedings of the International Multi Conference of Engineers and Computer Scientists 2009 Vol II March 18 , 2009, Hong Kong
[4] Jon N. Tombs "Design and implementation of a ram on the xs40 board" IEEE journal of Solid State Circuits, vol. 33, pp 1208-1219, 1998.
[5] Sreerama Reddy G.M "Design and VLSI Implementation of 8 Mb Low Power SRAM in 90nm" European Journal of Scientific Research Vol.26 No.2 (2009), pp.305-314
[6] Shawki Areibi "A First Course in Digital Design Using VHDL and Programmable Logic" IEEE journal of Solid State Circuits, Vol. 35,No. 8, pp. 1200-1204, August 2000.
[7] Website:www.micron.com/mti/msp/html
[8] Xilinx University Program Virtex-II Pro Development System Hardware Reference Manual

## Author's Profile

**Akshay P. Dhande**
Department of Electronics And Telecommunication,
SSGM college of Engg, Shegaon,
Maharashtra, India
akshaydhande126@gmail.Com