International Journal of Engineering Research and Development
e-ISSN: 2278-067X, p-ISSN: 2278-800X, www.ijerd.com
Volume 10, Issue 3 (March 2014), PP.29-39

Virtual Extended Memory Symmetric Multiprocessor (SMP) Organization Design Using LC-3 Processor: Case Study of Dual Processor

Rajendra S. Gad*[a], Udaysing V. Rane[a],
John Fernandes[b], Mahesh Salgaonkar[c].
a. Department of Electronics, Goa University, Taleigao Plateau, Goa 403206, India
b. NSTL RTC, DRDO, Varunapuri, Vasco – Da – Gama, Goa, India
c. MICD, National Institute of Oceanography, Dona pula Goa, India
**corresponding authors mail: rsgad@unigoa.ac.in*

Abstract:- *This paper presents the Symmetric Processor Organization design with the Little Computer (LC-3) processor. The physical memory space over 16 address bus lines of 64K is shares by the processors over various bank size of say 16K each to share the memory for parallelism. The paper discusses entire design architecture of the LC-3 processor and discusses the organization detail of Quad core LC-3 processor. The customized case of dual core LC-3 processor has been design sharing the memory over 32K back size. The instructions are design such that each one ends in exactly 5 clock cycles.*

Keywords:- SMP, LC-3, Memory Map, Memory extension.

I. INTRODUCTION

With the scaling of IC integration density, more processor cores could be accommodated on a single chip [1-4]. The design and integration of multicore architecture processors with transistor counts in the millions poses a challenge to designers given the complexity of the task and the time to market constraints. Due to the rapid advances in circuit integration technology, and to optimize performance while maintaining acceptable levels of energy efficiency and reliability, multicore technology or Chip-Multiprocessor is becoming the technology of choice for microprocessor designers. Multicore processors provide increased total computational capability on a single chip without requiring a complex micro-architecure. As a result, simple multicore processors have better performance per watt and area characteristics than complex single core processors [5]. One such example is a Godson-3 system which was designed from the ground up as a multiprocessing system capable of scaling to both small and large processor counts with little bandwidth, latency, or cost cliffs. The scalable directory-based cache coherence protocol is implemented to support cache coherence among a scalable multi-core/multi-chip system. Godson-3[6] system is a highly modular and hierarchical shared-memory multiprocessing design based on multi-core processors.

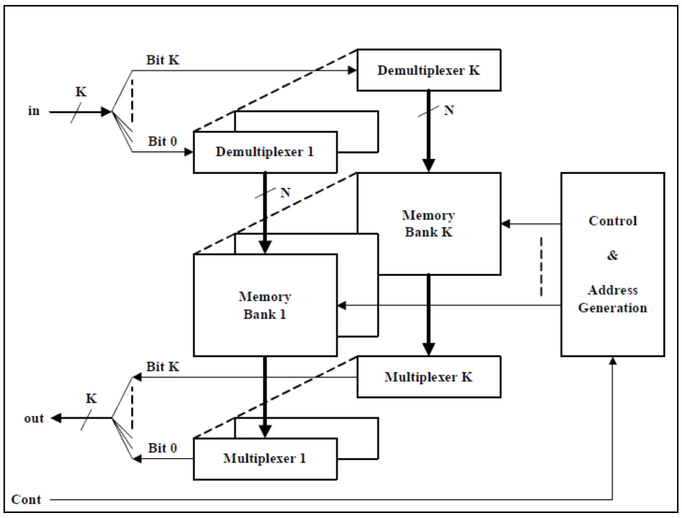
The latency resulting by the access of different levels of memory reduces the processing speeds causing more processor stalls while the data/instruction is being fetched from the main memory. Ways in which multiple cores send and receive data to the main memory greatly affect the access time and thus the processing speed. In multicore processors, two approaches to memory subsystem design have emerged in recent years, namely, the AMD Direct Connect architecture and the Intel Shared Bus architecture [7-10]. In the DirectConnect architecture, a processor is directly connected to a pool of memory using an integrated memory controller. A processor can access the other processors' memory pool via a dedicated processor-to-processor interconnect. On the other hand, in Intel's dual-core designs, a single shared pool of memory is at the heart of the memory subsystem. All processors access the pool via an external front-side bus and a memory controller hub.

Multi-core structures have not led to a decline in the rapid performance improvement of COTS (Commercial Off-The-Shelf) CPU recently. On the other hand, the performance of memory and I/O systems is insufficient to catch up with that of COTS CPU. The early virtual system prototyping and performance analysis of Multicore archiecture provides designers with critical information that can be used to evaluate them. Also the gap between CPU and memory speed has always been a critical concern that motivated researchers to study and analyze the performance of memory hierarchical architectures.

II. MEMORY LOGIC

a. Conventional Architecture

The conventional DRFM architecture [11], is based on standard components as demultiplexer, memory and multiplexer as shown in Fig. 1. High throughput is attained using relative slow memories. The demultiplexers and multiplexers reduce the data rate into and out of memory, thus increasing the time available for the memory READ/WRITE cycle. With sufficient demultiplexing (1:N) the throughput for this architecture becomes limited of the demultiplexer and multiplexer components instead of the memories. This architecture seams, at first glance, to be a reasonable approach to further increase throughput. Faster memory chips require less demultiplexing and multiplexing to match higher throughput demands.



- **Figure1: Conventional DRFM architecture using standard components.**

The maximum size of the linear address space in real-address mode for various flavors of Intel 32-bit architectures for examples are ‘1MB for 8086, 16MB for 286, 4GB for 386, 64GB for Pentium II and for 64-bit architectures like Intel Pentium, Dual–core and Core-2 is 64GB, Intel Xeon 1024GB and Intel Xeon processor 7560 with 16TB size[14]. Operating System use features of the processors memory management such as segmentation and paging to access memory which allow the same to be managed efficiently and reliably [12]. Programs running on an IA-32 processor can address up to 16,383 segments of different sizes and types, and each segment can be as large as 2^{32} bytes. Also advanced Transfer Cache Enhancement for two-three level Cache (Intel Xeon Processor 7560, L1-64KB, L2-256KB, L3-24MB) are very popular in Intel using NetBurst Microarchitecture[15]. Apart from these modes and enhancements there are some hardware convenience for latency time like Burst access mode as in synchronous DRAM with multiple bank architecture [13] makes it possible to access a number of data by changing only column addresses. However the numbers of cycles for row changes is considerable. Therefore accessing a large vector spanning several rows will not result in a continuous sequence of data.

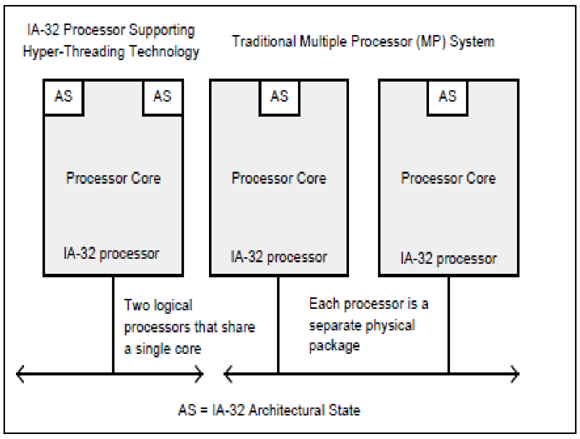
However, as technology improves, the speed of data, address and control lines becomes proportionally higher. Signal delays between chips become significant and a complex signal distribution network is required. There are also other drawbacks. Concurrent read and write operations using common bidirectional wide data bus is power consuming compared to two separate buses. Increasing the memory size (larger memory bank) will reduce throughput due to increased load on bit lines and expansion of the address space. Precise control of the time difference (delay) between read and write operations requires complex signal distribution due to that associated data is stored distributed in several memory banks. The access to data in memory for signal processing is unnecessarily complex due to the distributed architecture.

b. **Advance architecture.**

Memory scaling is in jeopardy as charge storage and sensing mechanisms become less reliable for prevalent memory technologies such as dynamic random access memory (DRAM) which are synchronized processor running at clock rate upto 3.0 GHz, [16]. Further the motherboard only mounts a fixed number of memory modules and cannot be extended. It is difficult to achieve memory capacity corresponding to a large computational capacity CPU with a multi-core structure. In large system for memory capacity intensive application such as search engine, database engine, and some of HPC applications, system cost and power consumption tend to be large, because of a large number of CPUs to keep the total memory capacity in the system.

Also one should note the mode of operation of OS. Usually address-Size Override Prefix is used in processor operating mode for compatibility issue. The default address size for instructions that access non-stack memory is determined by the current operating mode. Depending on the operating mode, this prefix allows mixing of 16-bit and 32-bit, or of 32-bit and 64-bit addresses, on an instruction-by-instruction basis[Refer Fig. 2]. The prefix changes the address size for memory operands. In 64-bit mode, the D (Default) bit is ignored, and all stack references have a 64-bit address size [29].

Proposed concepts like Memory module with vector scatter and gather access functions are well documented [25-26]. The basic concept of a memory module with vector load/store functions was proposed by Tanabe. On the other hand, when COTS are utilized for HPC, main memory capacity is insufficient, unlike the case of a vector supercomputer. The solution such DIMMnet-2 for problems of COTS comprises vector load/store functions and capacity extensibility on a memory module exist[23] or an attached accelerator on a CPU socket such as the DRC Reconfigurable Processor Unit (RPU110)[24].


IA-32 Processor Supporting Hyper-Threading Technology
Traditional Multiple Processor (MP) System
AS
AS
AS
AS
Processor Core
Processor Core
Processor Core
IA-32 processor
IA-32 processor
IA-32 processor
Two logical processors that share a single core
Each processor is a separate physical package
AS = IA-32 Architectural State

Also high density 3D memory architecture based on the resistive switching effect having several crossbar arrays stacked on top of each other, can circumvent the loss of lateral crossbar size. The fabrication of a second, third, fourth to nth crossbar layer, will only cause minor additional CMOS overhead in form of a layer decoder, which just needs to address n-layers (whereby n will certainly be a small number)[17]. Also architectures like Multibank memory banks architectures often customized for the specific applications [19], Multilevel phase change memory [20] and Chip Multiprocessors (CMPs) are now commodity hardware, but commoditization of parallel software remains elusive. In the near term, the current trend of increased core-per-socket count will continue, despite a lack of parallel software to exercise the hardware. Future CMPs must deliver thread-level parallelism when software provides threads to run, but must also continue to deliver performance gains for single threads by exploiting instruction-level parallelism and memory-level parallelism [21]. Hardware approach like Spin-transfer torque RAM and phase-change RAM are vying to become the next-generation embedded memory, offering high speed, high density, and non-volatility [22].

Recently, memory bandwidth and capacity requirements in distributed- memory multiprocessor systems are being accelerated by multicore configurations, but the interconnect networks between processors have not improved as fast, resulting in a widening gap between the performances of individual processors and multiprocessor computing systems. Attenuation, crosstalk and reflections in electrical signal lines become increasingly severe problems as bus and backplane speeds increase. Without innovations in the multiprocessor system architecture, it is becoming difficult to maintain bandwidth and capacity per core, as the number of cores per chip increases [27-28].

III. SYMMETRIC PROCESSOR USING LC3

a. Design of LC3 Processor

The LC-3 (Little computers with three instructions) specifies a word size of 16 bits for its registers and uses a 16-bit addressable memory with a 2^{16}-1 location address space. The register file contains eight registers, referred to by number as R0 through R7. All of the registers are general-purpose in that they may be freely used by any of the instructions that can write to the register file, but in some contexts (such as translating from C code to LC-3 assembly) some of the registers are used for special purposes [18].

Instructions are 16 bits wide and have 4-bit opcodes. The instruction set defines instructions for fifteen of the sixteen possible opcodes, though some instructions have more than one mode of operation. The architecture supports the use of a keyboard and monitor to regulate input and output; this support is provided through memory mapped I/O abstractions. In simulation, these registers can be accessed directly, and the architectural specification describes their contents. The entity diagram of the same is given in Fig. 3.

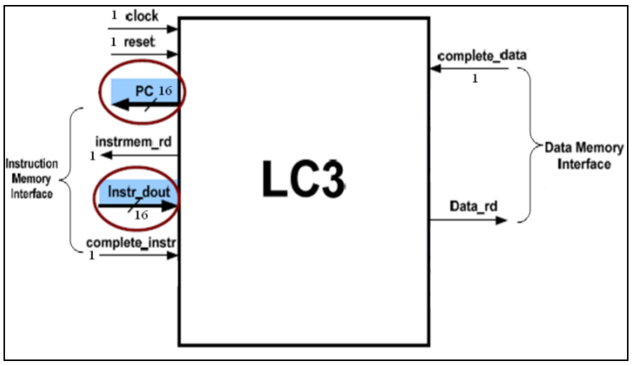


Figure 3: Entity diagram of the LC3 processor.

The LC-3 instruction set implements fifteen types of instructions, with a sixteenth opcode reserved for later use. The architecture is load-store architecture; values in memory must be brought into the register file before they can be operated upon. Arithmetic instructions available include addition, bitwise AND, and bitwise NOT. These operations are sufficient to implement a number of basic arithmetic operations, including subtraction (by negating values) and bitwise left shift. The LC-3 can also implement any bitwise logical function, because NOT and AND together are logically complete. The LC-3 provides both conditional and unconditional control flow instructions, however the LC-3 does not support the direct arithmetic comparison of two values; comparing two register values arithmetically requires subtracting one from the other and evaluating the result.

b. Entities Details:

The details of Fetch , Decode, Execute and WriteBack entities are given in Figure 4,5,6,7 respectively with the output and input port descriptions.

i. The FETCH BLOCK:

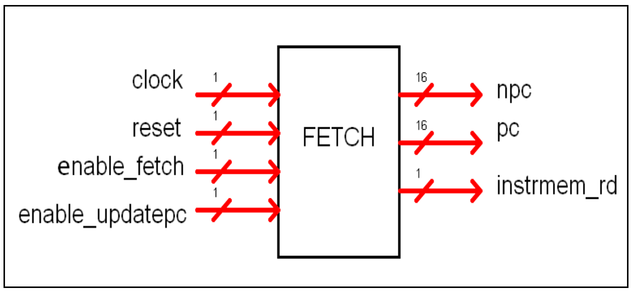


Figure 4: Entity diagram of the Fetch block.

Inputs:

• clock, reset[1bit]:

• enable_updatePC [1bit]: This signal enables the PC to change at the positive edge of the clock to PC+1.

• enable_fetch[1bit]: This signal allows for fetch to take place. If this is low, then reading of the Instruction Memory should not be allowed.

Outputs:

• instrmem_rd [1 bit]: This signal indicates to the Memory that a read is to be performed, rather than a write. This signal should be high when fetch is enabled and is asynchronous.

• pc[16 bits]: the current value of the program counter
• npc [16 bits] should always be pc+1 (asynchronous).

On reset, at the positive edge of the clock, pc = 16'h3000 and hence asynchronously npc = 16'h3001.

ii. The DECODE BlOCK:

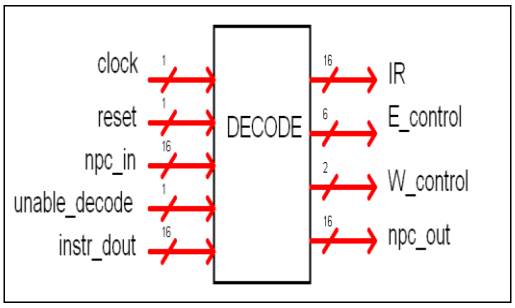


Figure 5: Entity diagram of the Decode block.

Inputs:
• clock, reset [1 bit]**:**
• Instr_dout [16 bits]**:** this signal comes from Instruction memory and contains the instruction to be used to do the relevant operations in "*Decode*" state.
• npc_in**[**16 bits**]:** This corresponds to the npc value from the Fetch stage which needs to be passed along the pipeline to the Execute block.
• enable_decode [1 bit]: When 1, this signal allows for the operation of the decode unit in normal mode where it creates the relevant control signals at the output based on the input from the Instruction Memory. If 0, the block stalls with no changes at the output.

Outputs:
• IR[16 bits] : This is equal to Instr_dout
• npc_out[16 bits]: This signal reflects the value of npc_in
• E_control [6 bits] : This signal controls the Execute unit. It allows for the choice of the right input values for the ALU within the Execute and also controls the type of operation that is going to be performed.
• W_control [2 bits]: This signal determines the right choice between the flowing for a write to the register file
1:the output from an ALU operation (for alu operations)
2:the output from a PC relative operation (for LEA) and
3:the output from memory (for loads).
All outputs are created at the positive edge of the clock when enable_decode = 1.

4.1.3 *The EXECUTE BLOCK:*

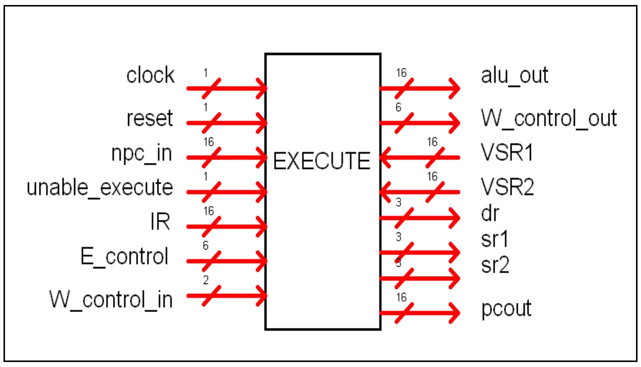


Figure 6: Entity diagram of the Execute block.

Inputs:
• clock, reset[1 bit]
• E_control: [6 bits]: This signal has already been explained in the Decoder section.
• IR: [16 bits]The instruction register that is used to create the offset values for PC based operations.
• npc_in: [16 bits]:The npc that was passed along the pipeline from the Decode stage corresponding to 1+ PC that gave the IR.

• VSR1, VSR2: [16 bits]: These values come from the Writeback unit based on the sr1 and sr2 outputs.
• enable_execute: [1 bit]This signal is the master control for the entire Execute block. All the outputs are created only when this signal is enabled. If zero, all the outputs should hold their previous value.

Outputs:
• aluout: [16 bits]This is the result of ALU operations
• pcout: [16 bits] Stores the result of program counter.
• dr: [3 bits] :The destination address for the instruction that came into the Execute which is of relevance for loads and ALU operations.
• sr1: [3 bits]This reflects the contents of IR[8:6] for all instruction types.
• sr2: [3 bits] This reflects the contents of IR[2:0] for ALU instructions, IR[11:9] for stores and is zero for all other types.

4.1.4 *The WRITE BACK BLOCK*:

The Writeback unit contains the register file which provides the relevant data values for register based operations. Hence it also controls the values that need to be written into the Register file.

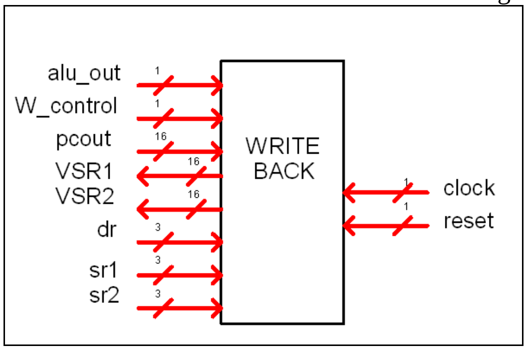


Figure 7: Entity diagram of the WriteBack block.

Inputs:
• clock, reset
• npc[16 bits]
• W_control_in [2 bits]
• aluout [16 bits]: value from Execute for ALU operations
• pcout [16 bits]: value from Execute corresponding to PC based operations for LEA
• enable_writeback[1 bit]: Enable signal to allow for operation of the Writeback block. If zero, the register file is not written to.
• sr1, sr2 [3 bits]: Source register addresses 1 and 2 for Execute operations to be performed.
• dr[3 bits]: Destination register address where the data chosen using W_Control value is written to in the Register File.

Outputs:
• clock, reset.

c. Integrated block of LC3

The block diagram (Figure 8) consists of four sub blocks known as entities namely Fetch, Decode, Execute and WriteBack are integrated together to form the final design of LC-3 processor.
The '***clock***' and '***reset***' signals are common for the entire system. The '***enable_update_pc***' enables updating of the program counter. The '***enable_fetch enables***' the fetch block.

a. Bus Structure:

The '***instructionmem_rd'*** is used to enable the instruction memory and the '16 bit PC' terminal is used to address the external instruction memory.

The 'npc' going from decode gives the next count of the PC, and '***instr_dout'*** from the instruction memory gives the instruction to the decode block.

The instruction is then used by the main control logic and the corresponding '***w_control_out'***, '***e_control_out'*** are generated. ('***mem_control_out'*** *not used in this paper*). The '***npc_out'*** and IR is same as '***npc***' and '***instr_dout'*** respectively.

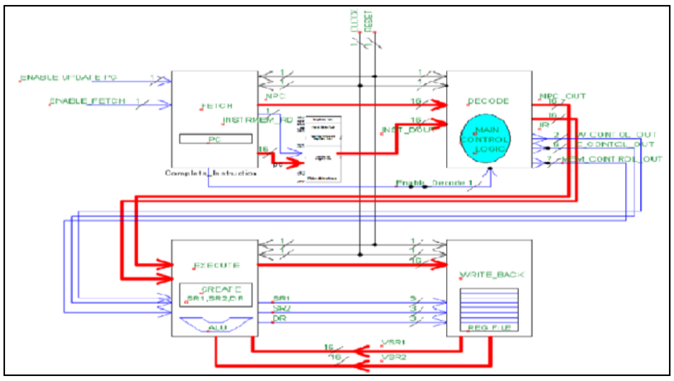


Figure 8: Detailed diagram of LC3 processor.

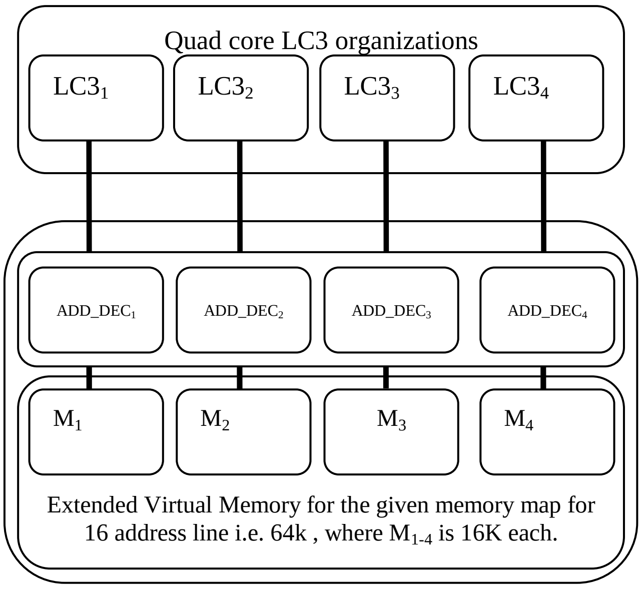
The ‘***w_control_out***’ and ‘***e_control_out***’ is then given to the execute block along with the ‘***npc***’ and ‘***IR***’, being used to generate the addresses of ‘sr1’, ‘sr2’ and ‘dr’.
These addresses are then given to the write back which consists of the ‘reg’ file which has the data stored. This is the fetched and given to the ALU through the signals ‘vr1’ and ‘vsr2’. The ALU then operates on them and generates the result.

- memory locations 3000h to *FDFF*h are used for storing instructions.
- locations *2FFF*h to *0200*h are used for memory operations (not used in this paper).

N.B: The remaining memory locations are not used in this paper.

d. SMP organization using LC3.

The scalable Quad-core architecture as shown in Fig. 9 is shown with address line details. The address lines of the every LC3 processor is decoded by the independent Address_decoder circuits for proposed decoder logic as shown in the Table 1. The decoder logic decodes the word lines of the four memory bank M1, M2, M3, M4 on the physical address space of 64K but having independent address decoder logic to fetch out words on the rows. These four bytes can be placed on the independent buses of the Quad processor organization with the help of the MUX and DEMUX for the data bus. One may note that in Table 1 no two rows over memory bank clashes for row contention during read/write operation of memory.


Quad core LC3 organizations
$LC3_1$
$LC3_2$
$LC3_3$
$LC3_4$
ADD_DEC_1
ADD_DEC_2
ADD_DEC_3
ADD_DEC_4
M_1
M_2
M_3
M_4
Extended Virtual Memory for the given memory map for 16 address line i.e. 64k , where $M_{1\text{-}4}$ is 16K each.
Figure 9: Memory map of LC3 processor.

This organization has been customized for dual core LC3 processor as shown in Fig. 10. It is clear from the FSM diagram that only the ‘Fetch’ operation there is possibility of having bus contention for memory module. Other block operation i.e. Decode, Execute and WriteBack are concurrent. The bus contention in the Fetch block have been taken in the design by positive and negative edge sensitive Fetch block for the two cores respectively shown in Fig. 10.

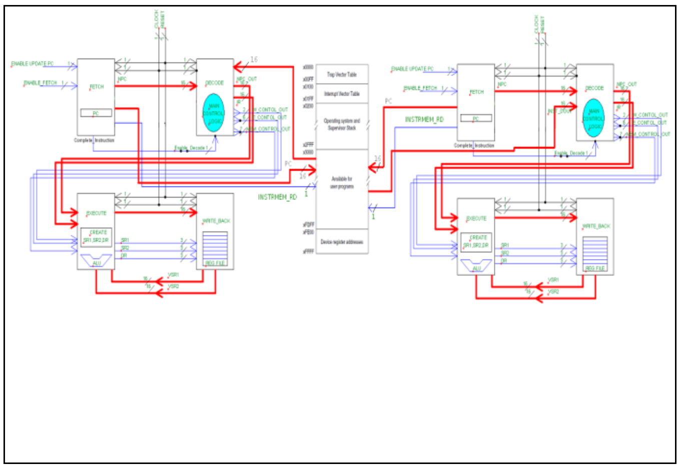


Figure 10: Dual-core SMP organization using LC3.

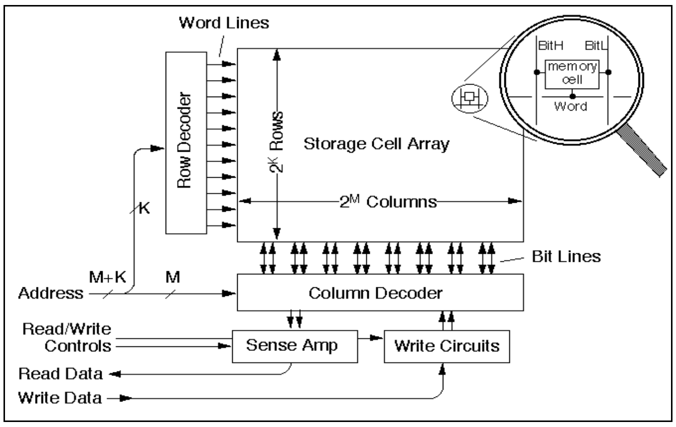


Figure 11: Typical memory architecture.

One may run the clock at slow rate to optimize the propagation delay in the row and column decoder , charging cycle of ‘Bit_line’ and ‘equalizer’ circuits, discharging cycle with the sense amplifier and other bidirectional buffers (as shown in Fig. 12) in the memory module during this positive and negative edge trigger mechanism.

Table 1: Four proposed addressing decoding logic for four memory logic for three address lines. The values of the ‘Wn’ shows the respective ‘word line’ of the memory chip having four memories logic with independent address decoders ‘ADD_DECn’.

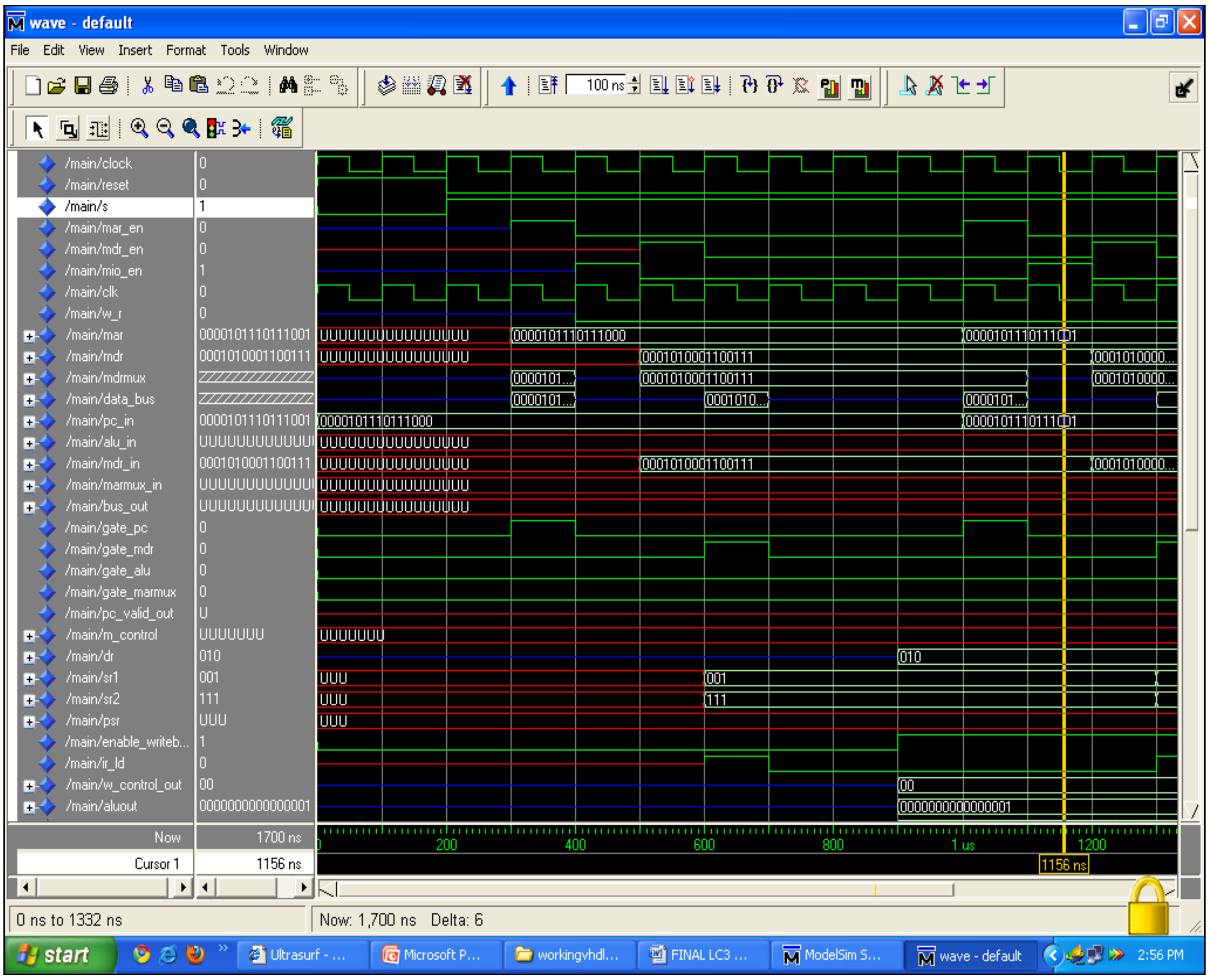
Three address lines			Address Decoder lines								Top down	Down Up addressing	Middle Up addressing	Middle Down addressing Up
A2	A1	A0	D7	D6	D5	D4	D3	D2	D1	D0	M1	M2	M3	M4
0	0	0	0	0	0	0	0	0	0	1	W0	W7	W3	W4
0	0	1	0	0	0	0	0	0	1	0	W1	W6	W2	W5
0	1	0	0	0	0	0	0	1	0	0	W2	W5	W1	W6
0	1	1	0	0	0	0	1	0	0	0	W3	W4	W0	W7
1	0	0	0	0	0	1	0	0	0	0	W4	W3	W7	W0
1	0	1	0	0	1	0	0	0	0	0	W5	W2	W6	W1
1	1	0	0	1	0	0	0	0	0	0	W6	W1	W5	W2
1	1	1	1	0	0	0	0	0	0	0	W7	W0	W4	W3

IV. CONCLUSION AND DISCUSSIONS

LC-3 design was confirmed for functional simulation for the following instructions.

- ADD, NOT, AND and LEA.

The program was placed between the memory map at *3000*h - *FDFF*h. The screen shots of the functional simulation are given in Figure 12 and13.



/main/clock	0
/main/reset	0
/main/s	1
/main/mar_en	0
/main/mdr_en	0
/main/mio_en	1
/main/clk	0
/main/w_r	0
/main/mar	0000101110111001
/main/mdr	0001010001100111
/main/mdrmux	▨
/main/data_bus	▨
/main/pc_in	0000101110111001
/main/alu_in	UUUUUUUUUUUUU
/main/mdr_in	0001010001100111
/main/marmux_in	UUUUUUUUUUUUU
/main/bus_out	UUUUUUUUUUUUU
/main/gate_pc	0
/main/gate_mdr	0
/main/gate_alu	0
/main/gate_marmux	0
/main/pc_valid_out	U
/main/m_control	UUUUUUU
/main/dr	010
/main/sr1	001
/main/sr2	111
/main/psr	UUU
/main/enable_writeb...	1
/main/ir_ld	0
/main/w_control_out	00
/main/aluout	0000000000000001
Now	1700 ns
Cursor 1	1156 ns

```
0 ns to 1332 ns          Now: 1,700 ns   Delta: 6
```

Figure 12: Detailed functional simulation diagram of Dual core LC3 processor executing AND.

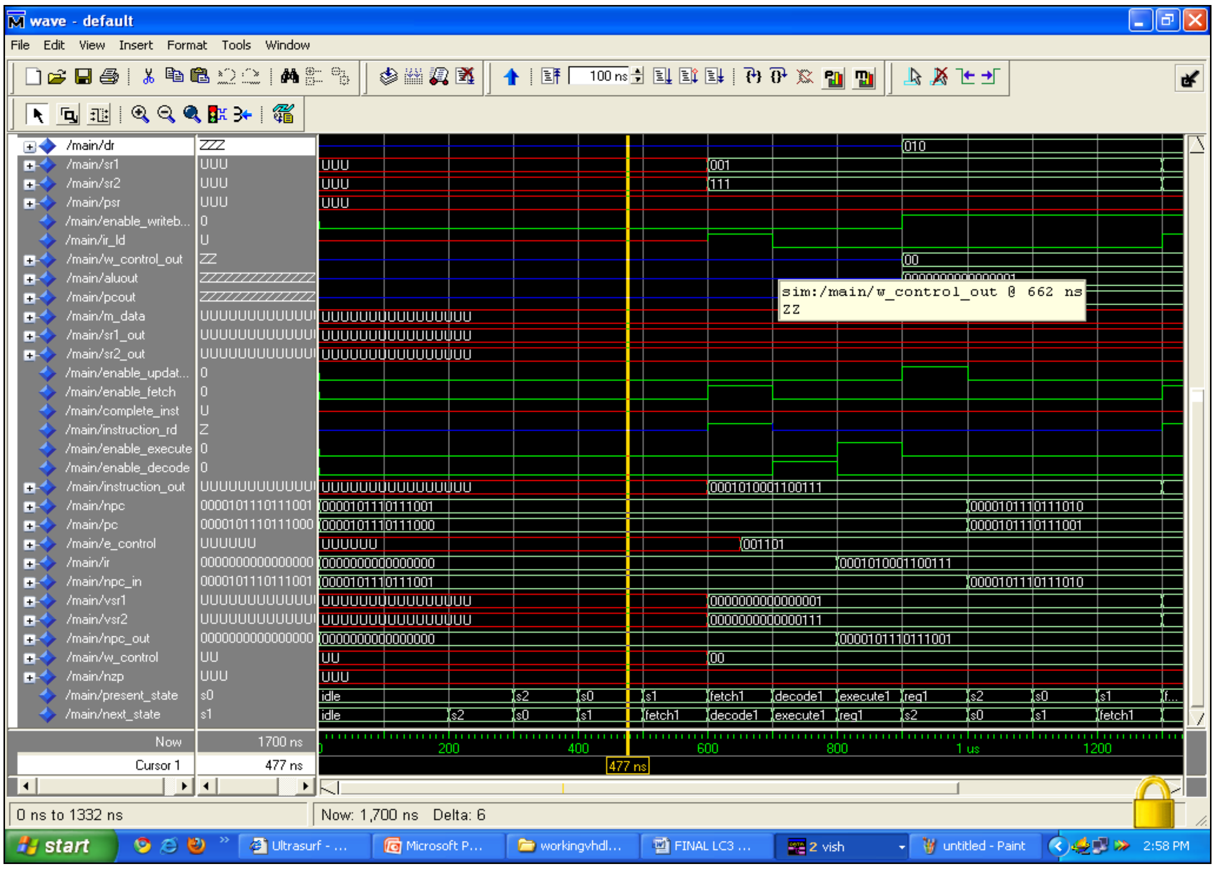


Figure 13: Detailed functional simulation diagram of Dual core LC3 processor executing AND.

The simulation Figure 12 shows on the last line from clock 900ns the result of AND operation. Line 26,27 at clock cycle 600ns onwards and in Figure 13 shows the data for AND operation.

DRAM scaling faces many significant technical challenges as scaling attacks weaknesses in both components of the one transistor and capacitor cell. Capacitor scaling is constrained by the DRAM storage mechanism, which requires maintaining charge on a capacitor. In future, process scaling is constrained by challenges in manufacturing small capacitors that store sufficient charge for reliable sensing despite large parasitic capacitances on the bitline. The scaling scenarios are also bleak for the access transistor. As this transistor scales down, increasing subthreshold leakage will make it increasingly difficult to ensure DRAM retention times. Not only is less charge stored in the capacitor, that charge is stored less reliably. These trends impact the reliability and energy efficiency of DRAM in future process technologies. According to ITRS, “manufacturable solutions are not known” for DRAM beyond 40nm.17 In contrast, ITRS projects PCM scaling mechanisms will extend to 32 nm, after which other scaling mechanisms might apply. Such PCM scaling has already been demonstrated with a novel device structure fabricated by Raoux[30]. Although both DRAM and PCM are expected to be viable at 40nm technologies, energy scaling trends strongly favor PCM with a 2.4× reduction in PCM energy from 80 to 40nm. In contrast, ITRS projects DRAM energy falls by only 1.5× at 40nm, which reflects the technical challenges of DRAM scaling [31].

REFERENCES

[1] Hammond L et al. A single-chip multiprocessor. IEEE Computer,Sept. 1997, 30(9): 79-85.

[2] Tendler J, Dodson S, Fields S, Le H, Sinharoy B. Power4 system microarchitecture. IBM Technical White Paper, October 2001.

[3] Kalla R et al. IBM POWER5 chip: A dual-core multithreaded processor. IEEE Micro, March 2004, 24(2): 40-47.

[4] Kongetira P et al. Niagara: A 32-way multithreaded Sparc processor. IEEE Micro, March 2005, 25(2): 21-29.3-6]

[5] S. Balakrishnan, R. Rajwar, M. Upton, and K. Lai, “The impact of performance asymmetry in emerging multicore architectures,” in Proceedings 32nd International Symposium on Computer Architecture (ISCA ’05), pp. 506–517, Madison, Wis, USA, June 2005.

[6] Gao X, Chen YJ, Wang HD et al. System architecture of Godson-3 multi-core processors. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 25(2): 181–191 Mar. 2010.

[7] P. Conway and B. Hughes, "The AMD Opteron Northbridge architecture," IEEE Micro, vol. 27, no. 2, pp. 10–21, 2007.

[8] C. N. Keltcher, K. J.McGrath, A. Ahmed, and P. Conway, "The AMD Opteron processor for multiprocessor servers," IEEE Micro, vol. 23, no. 2, pp. 66–76, 2003.

[9] "Dual-Core Intel_ Xeon_ Processor 5000 Series Datasheet," May 2006.

[10] R. Varada, M. Sriram, K. Chou, and J. Guzzo, "Design and integration methods for a multi-threaded dual core 65nm XeonˆA_processor," in Proceedings of IEEE/ACMInternational Conference on Computer-Aided Design, Digest of Technical Papers (ICCAD '06), pp. 607–610, San Jose, Calif, USA, November 2006.

[11] G. Weber, J. Culp, and M. Robinson, "DRFM Requirements Demand Innovative Technology,"Microwave J., vol. 29, no. 2, pp. 91-104, Feb 1986.

[12] Protected-Mode Memory Management," in the Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 3A.

[13] H. Kim and I. –C. Park, "High-Performance and Low-Power Memory-Interface Architecture for Video Processing Applications," IEEE Transaction on circuits and systems for video technology, vol. 11, no. 11, pp. 1160-1170, Nov 2001.

[14] Intel® 64 and IA-32 Architectures Software Developer's Manual Volume 1: Basic Architecture, Order Number: 253665-037US, January 2011 Page 2-32.

[15] The Intel® 64 and IA-32 Architectures Software Developer's Manual consists of five volumes: Basic Architecture, 253665-037US, January 2011.

[16] Benjamin C. Lee, Engin Ipek, Onur Mutlu, and Doug Burger, 'Phase Change Memory Architecture and the Quest for Scalability', Communications of the ACM. July 2010, vol. 53 , No. 7 .

[17] C. Kügeler, M. Meier, R. Rosezin, S. Gilles, R. Waser, 'High density 3D memory architecture based on the resistive switching effect ,Solid-State Electronics 53 (2009) 1287–1292.

[18] www.scribd.com/doc/4596293/LC3-Instruction-Details.

[19] Yun-Nan Chang , 'A Multibank Memory-Based VLSI Architecture of DVB Symbol Deinterleaver', IEEE Transactions on Very Large Scale Integration (VLSI) Systems, May 2010, 18 Issue:5, 840 – 843.

[20] Moinuddin K. Qureshi; Michele M. Franceschini; Luis A. Lastras-montaño; John P. Karidis , 'Morphable Memory System: A Robust Architecture for Exploiting Multi-level Phase Change Memorie' ACM SIGARCH Computer Architecture News Vol.: 38, No.: 3, June 2010, Page 153-162.

[21] Dan Gibson, David A. Wood , 'Forwardflow: A Scalable Core for Power-constrained CMPs', ACM SIGARCH Computer Architecture News Vol.: 38, No.: 3, June 2010 [Page 14-25].

[22] Yuan Xie, Modeling, Architecture, and Applications for Emerging Memory Technologies, 'Design & Test of Computers, IEEE'28 Issue:1, P44 – 51.

[23] Tanabe N, Hakozaki H, Nakatake M, Dohi Y, Nakajo H, Amano H (2004) A new memory module for memory intensive applications. In: IEEE international conference on parallel computing in electrical engineering (ParElec2004), pp 123–128.

[24] DRC Computer Corp (2007) Datasheet of DRC reconfigurable processor unit RPU110. http://drccomputer.com/pdfs/DRC_RPU110_fall07.pdf.

[25] Tanabe N, Hakozaki H, Nakatake M, Dohi Y, Nakajo H, Amano H (2004) A new memory module for memory intensive applications. In: IEEE international conference on parallel computing in electrical engineering (ParElec2004), pp 123–128.

[26] Noboru Tanabe, Hirotaka Hakozaki, 'An enhancer of memory and network for applications with large-capacity data and non-continuous data accessing , J Supercomput (2010) 51: 279–309.

[27] John L. Hennessy, David A. Patterson, Computer Architecture: A Quantitative Approach, fourth ed., Elsevier, Amsterdam, 2007, pp. 200–201.

[28] Hyun-Shik Lee, Shin-Mo An, S.G. Lee, B.H. O, S.G. Park, El-Hnag Lee*, A 10 Gbps interchip optical interconnection module for optical circuit board application, OSA/NANO 2006 (JWB6.pdf).

[29] AMD64 Architecture Programmer's Manual, Volume 1:Application Programming, 24592 3.15 November 2009.

[30] Raoux, S. et al. Phase-change random access memory: A scalable technology. IBM J. Res. Dev. 52, 4/5 (Jul/Sept 2008).

[31] S emiconductor Industry Association. Process integration, devices & structures. International Technology Roadmap for Semiconductors, 2007.