



I/O PROCESSOR DESIGN OF MICRO RISC ISA PROCESSOR FOR IOT APPLICATION

GUIDED BY:

Er.SAJ KAPOOR
Senior Director of Engineering,
Analog Devices,Bangalore

PROF RESHNA S
Associate Professor
Dept.of ECE
TKM COLLEGE OF ENGINEERING
Kollam

PRESENTED BY:

ANAMIKA CHITHRAN(B18ECA17)
CHETHAN KRISHNA(B18ECA27)
S SANKARA SUBRAMANIAN(B18ECA60)
SUSMITH DAS K K(B18ECA62)

02-06-22

CONTENTS

1. Introduction
2. Literature Survey
3. Objective
4. Methodology
5. Results
6. Future scope
7. Conclusion
8. References

02-06-22



INTRODUCTION

- To design a DMA controller and memory controller for 16 bit fixed and floating point processor core derived from existing SHARC based ADSP-2106x processor.
- ADSP-2106x is a RISC based microprocessor developed by Analog Devices mainly used for DSP applications.
- DMA provides a mechanism for transferring an entire block of data between memories and devices with least inputs from CPU .
- Memory controller for interfacing external memory.

02-06-22



LITERATURE SURVEY

- 1) S. Min, M. Alian, W. -M. Hwu and N. S. Kim, "Semi-Coherent DMA: An Alternative I/O Coherency Management for Embedded Systems," in IEEE Computer Architecture Letters, vol. 17, no. 2, pp. 221-224, 1 July-Dec. 2018, doi: 10.1109/LCA.2018.2866568.

- ☐ Modern embedded CPUs adopt NC-DMA over C-DMA because of simplicity
- ☐ NC-DMA design - device driver flush a wide range of cache space
- ☐ SC-DMA can solve excessive invalidation problem
- ☐ Small DMA cache record recent DMA buffer regions
- ☐ DMA mapping - SC-DMA marks and invalidates the DMA region
- ☐ Marks are merged as a 64-bit mask and stored in the DMA cache

02-06-22

- ☐ Proposed model improved the Rx bandwidth of a 10GbE NIC by up to 54%



2) N. Vujic, F. Cabarcas, M. Gonzalez Tallada, A. Ramirez, X. Martorell and E. Ayguade, "DMA++: On the Fly Data Realignment for On-Chip Memories," in IEEE Transactions on Computers, vol. 61, no. 2, pp. 237-250, Feb. 2012, doi: 10.1109/TC.2010.255.

- Single-Instruction Multiple-Data (SIMD) units found in a variety of processors.
- Their efficiency are usually bounded by alignment of memory architecture.
- DMA++ engine is enriched with a Realignment Unit.
- The DMA++ engine consists of a DMAC, BIU ,RU.
- Realignment Unit that performs parallel to DMA engine.
- Reorder Buffer (ROB), Two-entry buffer, Block Decrementers, Realignment Networks, and Merge Networks.
- Realignment Network consists of a shift count register, Barrel shifter.



3) H. Jung, S. Jung and Y. H. Song, "Architecture exploration of flash memory storage controller through a cycle accurate profiling," in IEEE Transactions on Consumer Electronics, vol. 57, no. 4, pp. 1756-1764, November 2011, doi: 10.1109/TCE.2011.6131151.

- NAND flash memory a storage medium in devices such as mobile phones, MP3 players, and digital cameras
- SSDs have an inherent weakness stemming from NAND flash memory and its complex architecture
- If the data in a page are updated, even by one bit, the entire block that the page belongs to must be erased
- Limited write/erase cycle for SSD'S. Beyond this it can't be used for storage
- Architecture of SSD Simulator-CPU, SRAM/DRAM, Data Buffer, Host interface, Channel/way controller.

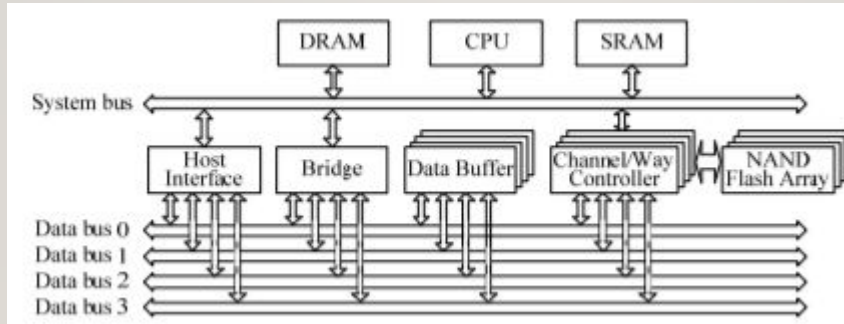


Fig.1 Architecture of SSD simulator

- When a read request arrives, an interrupt is sent from the host interface to the CPU as a notification of the event
- The CPU then starts to run the IRQ handler . This handler reads request information from the host interface and inserts it into the request queue
- The FTL then loads the new request from the queue.
- The command packet is sent to the channel/way controller and generate number of pages in the request
- Once the way buffer is filled with data, the channel/way controller performs a backend DMA operation to move data to the data buffer .
- When data are stored in the data buffer, the FTL checks whether or not the request has been completed successfully

02-06-22 If successful, the host interface trashes data from the data buffer and the request information is removed from the request queue.



4) N. Surana and J. Mekie, "Energy Efficient Single-Ended 6-T SRAM for Multimedia Applications," in IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 66, no. 6, pp. 1023-1027, June 2019, doi: 10.1109/TCSII.2018.2869945.

- A single ended 6-T (SE6T) static random access memory (SRAM) cell is proposed.
- 50% less dynamic power compared to conventional 6-T SRAM cell.
- Uses only single bit-line for write and single bit-line for read

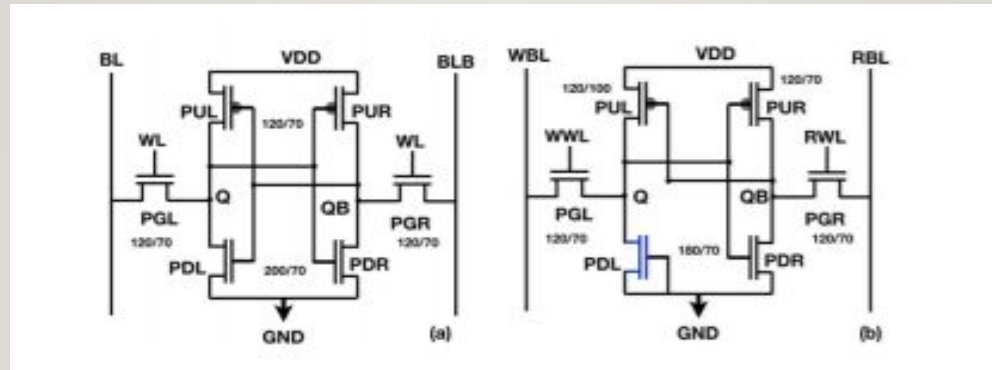


Fig.2 conventional 6-T SRAM (a) proposed SE6T SRAM cell.

- For a total BER of 0.05% at operating voltage of 0.55-0.65V, the 6T SRAM cell takes $2.34 \mu\text{m}^2$, whereas SE6T SRAM cell takes $1.2 \mu\text{m}^2$ as only one transistor (PGL) needs to be sized
- Power Consumption-
For PNSR of 15 dB, the proposed SE6T memory consumes only $0.48\times$ dynamic power as compared to conventional 6-T SRAM cell with barely 2% additional area.
- For PSNR of 27dB, proposed SE6T based memory is highly energy efficient and consumes $0.45\times$ dynamic power, $0.83\times$ leakage power and takes $0.6\times$ area when compared to conventional 6-T SRAM cell based embedded memory



Fig.3 Dynamic power, leakage power and area required per pixel

- Except for low PSNR value, SE6T embedded memory needs lesser area, and has lower leakage and dynamic power compared to conventional 6-T SRAM cell based memory.

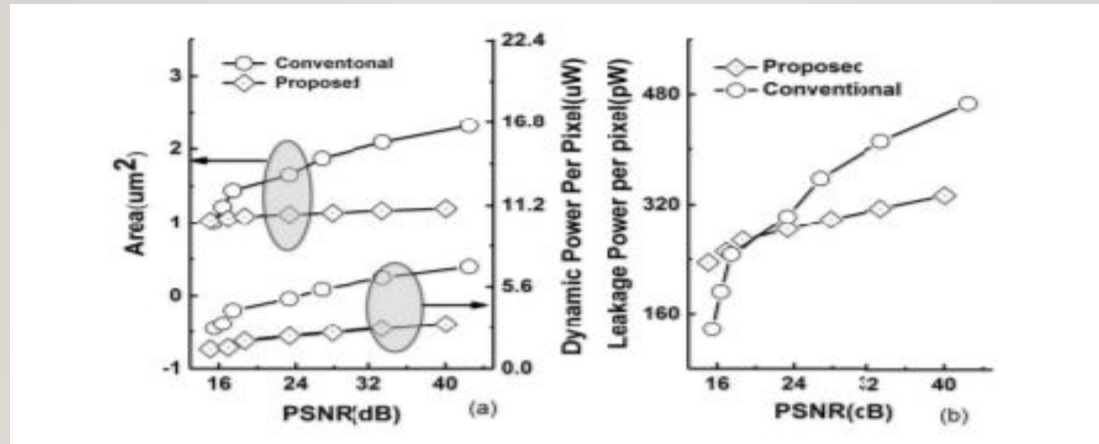
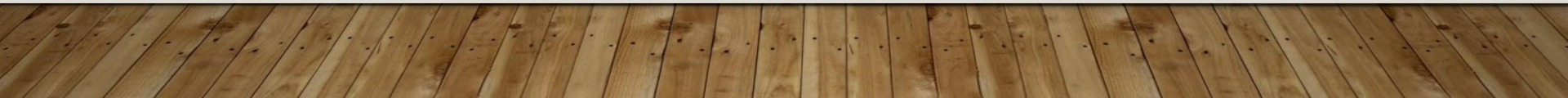


Fig.4 (a) Dynamic power and area consumed per pixel versus obtained PSNR (b) Leakage power consumed per pixel versus obtained PSNR.

5) H. Kavianipour, S. Muschter and C. Bohm, "High performance FPGA-based DMA interface for PCIe," 2012 18th IEEE-NPSS Real Time Conference, 2012, pp. 1-3, doi: 10.1109/RTC.2012.6418352.

- ☐ Data communication suit
- ☐ Developed for use in the Track Engine Trigger for IceCube Neutrino Observatory at South Pole.
- ☐ Applicable to bidirectional DMA transfer between FGPA logic and system memory on host PC via PCIe.
- ☐ Suit contains a DMA controller firmware, test benches, a Linux driver and a user application for DMA.



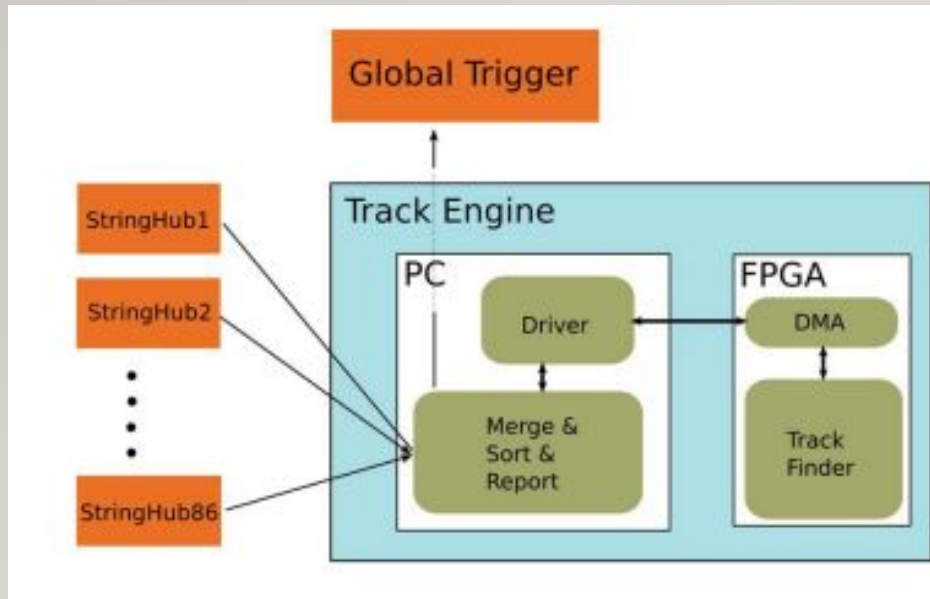


Fig. 5. Track Engine hierarchy.

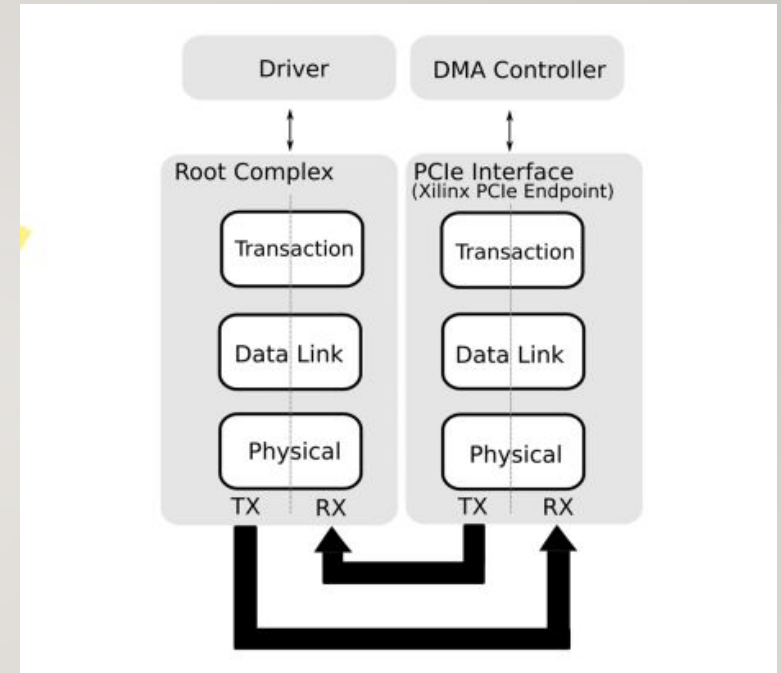


Fig. 6. Communication between the driver and the DMA Controller over the PCIe hierarchy.

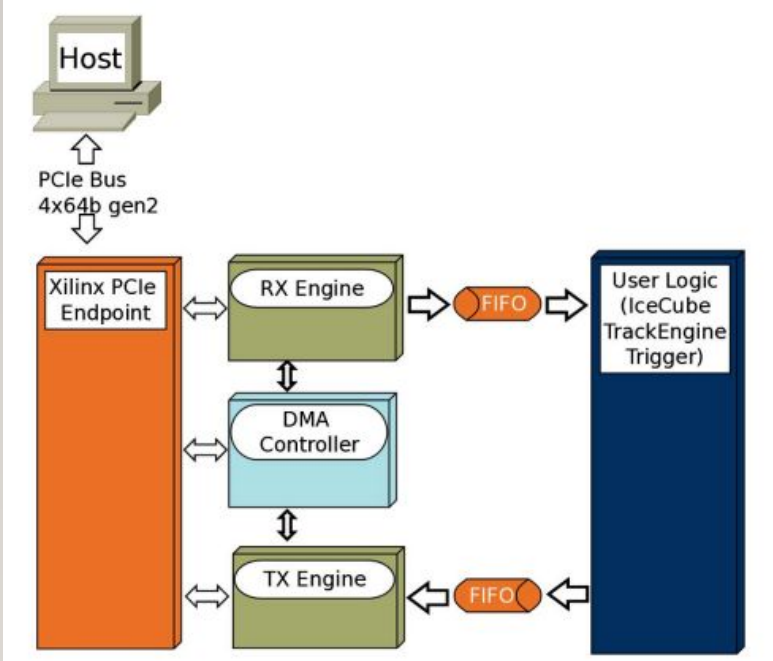
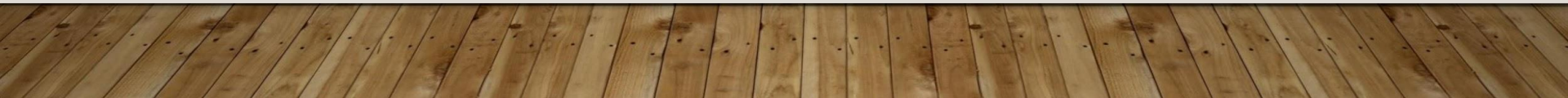


Fig. 7. DMA controller supervises the data transfer between the user logic and PCIe Endpoint.

6) L. Rota, M. Caselle, S. Chilingaryan, A. Kopmann and M. Weber, "A PCIe DMA Architecture for Multi-Gigabyte Per Second Data Transmission," in IEEE Transactions on Nuclear Science, vol. 62, no. 3, pp. 972-976, June 2015, doi: 10.1109/TNS.2015.2426877.

- A direct memory access (DMA) engine compatible with the Xilinx PCI Express (PCIe) core to provide a high-performance.
- To maximize the PCIe throughput the DMA address list is stored inside the FPGA and not in the central memory of the host CPU.
- The FPGA design package is complemented with simple register access to control the DMA engine by a Linux driver.
- A data throughput of 3461 MBytes/s has been achieved with a single PCIe Gen2 x8 lanes endpoint.



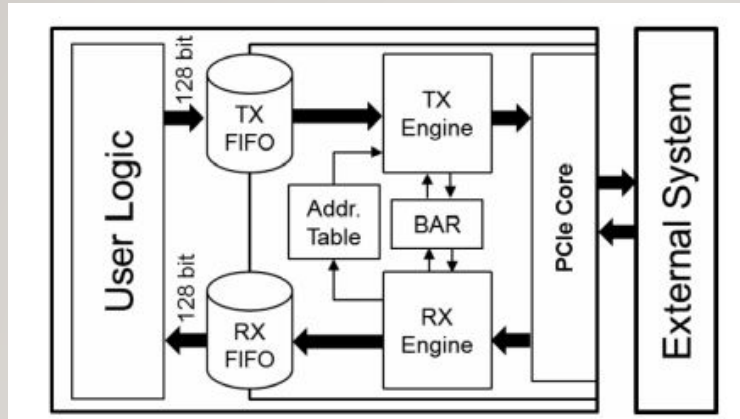


Fig. 8. Architecture of the DMA engine

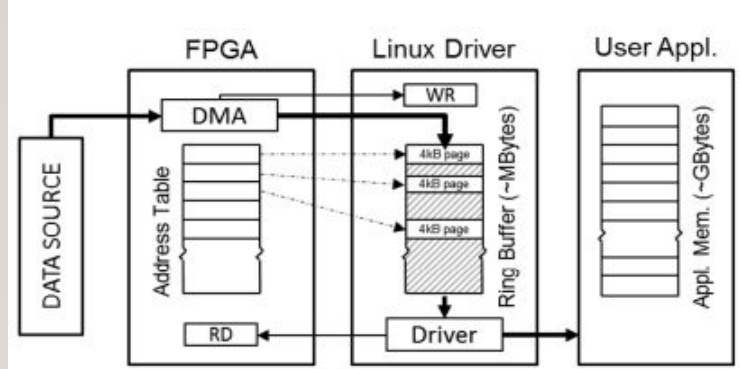


Fig. 9. Memory organization

7) Mrinal J Sarmah, Bokka Abhiram , Anil kumar A “Method for Booting an All programmable System on-Chip over PCI Express Link”,2017 July IEEE

- Increased complexity of processors and peripherals
- Peripheral Component Interconnect Express
- Problem with flash devices
- PCI as a candidate for booting



8) Design of Processing-“Inside”-Memory Optimized for DRAM Behaviors WON JUN LEE 1, CHANG HYUN KIM1, YOONAH PAIK1, June 2019

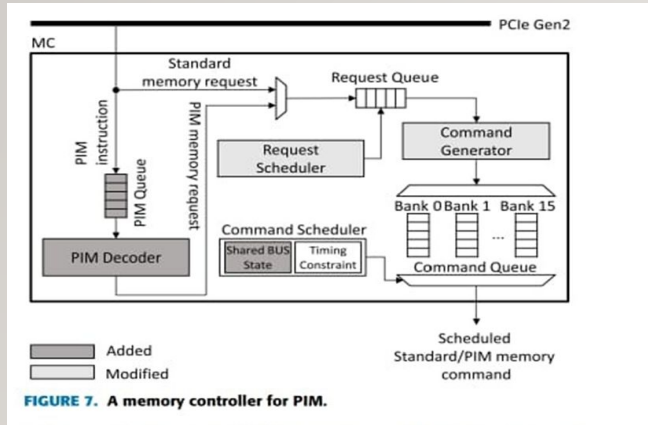


Fig.10 A memory controller for PIM

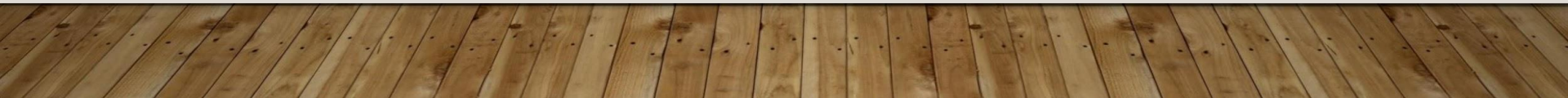
- To design and operate PIM Units inside DRAM
- Full computing performance and minimising the implementation cost

INFERENCES

- Design of DMA controller and its architecture.
- How to improve the dma transfer using different hardware elements.
- Memory organization and architecture of SRAM and Flash memory .

OBJECTIVES

- To design a DMA controller that can be configured to work in Chaining mode, Scatter-gather mode as well as in normal mode.
- 3-stage pipelining for improved throughput.
- To design a memory controller that can be used for external memory access.
- The core designed can work along with other subsystems to work as a processor for iot application.



METHODOLOGY

- Design Specification.
- Architecture- ADSP-2106x
- RTL design –Verilog
- RTL verification – ModelSim and Python
- Synthesis - Xilinx ISE design suite.

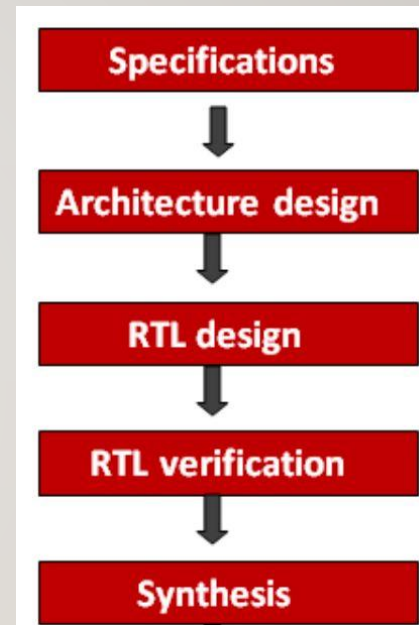
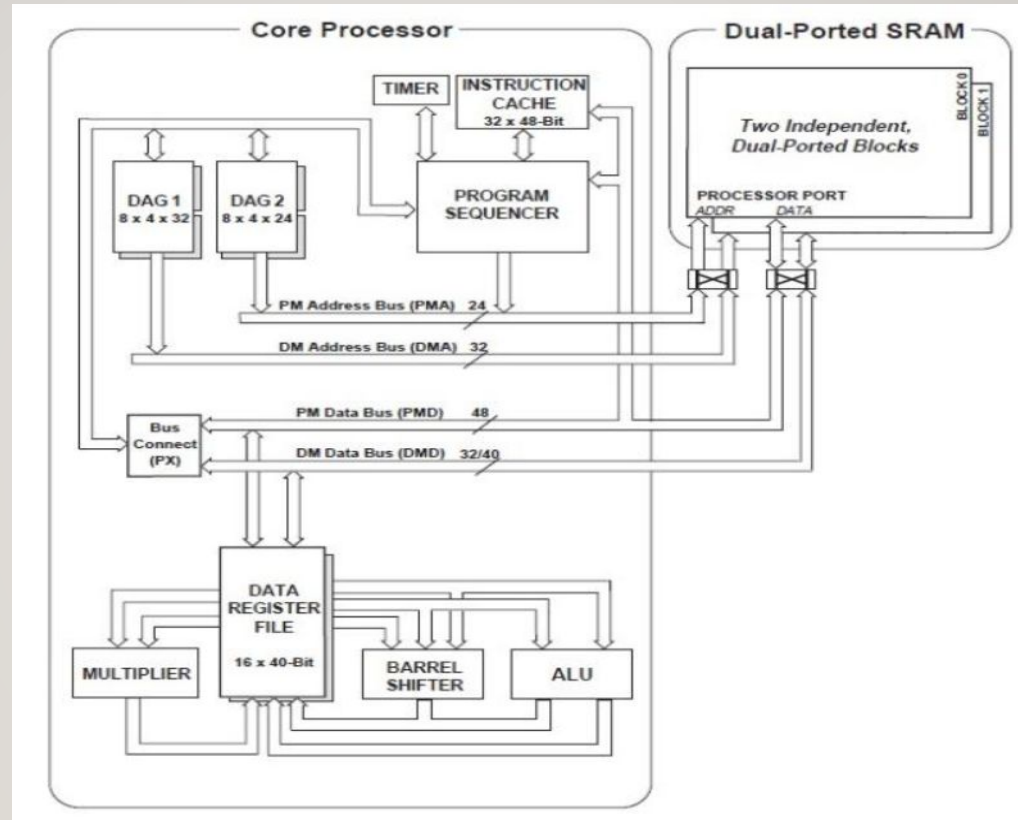


Fig 11: Block diagram of Design flow

Block Diagram



02-06-22

Fig.12 Sharc ADSP-2106 core architecture

DMA controller

- 8 Depth FIFO for handling stalling
- 3 stage pipelined DMA Controller
- Separate Internal and External Address Generators
- Compatible to Interface with External memory of N+1 and N+2 read latency
- Normal Mode, Scatter Gather mode , Chaining Mode.

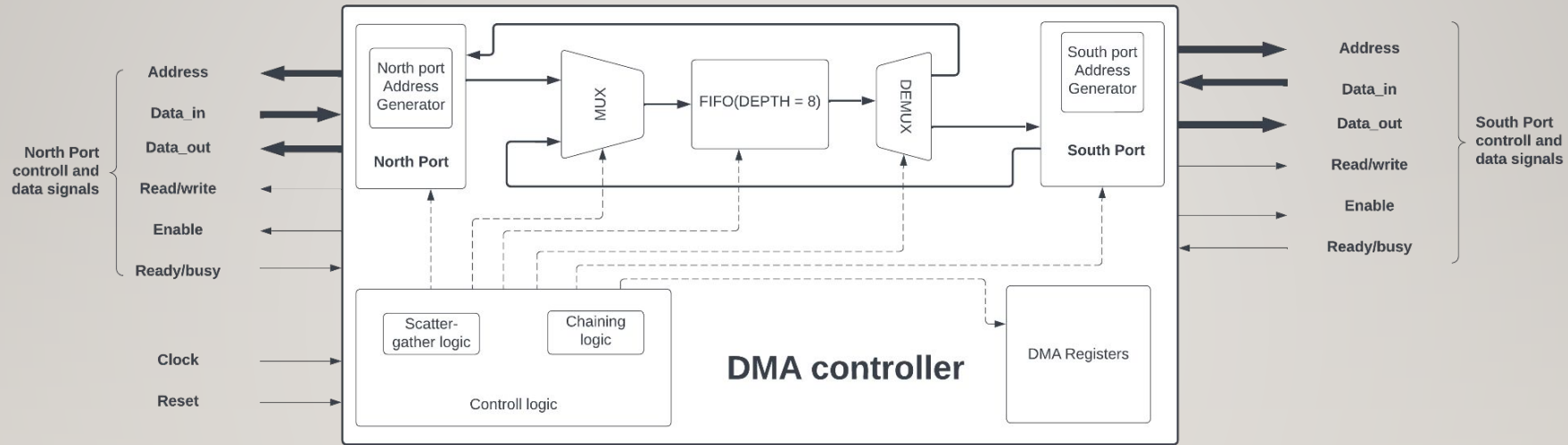


Fig 13: DMA Data & Control path

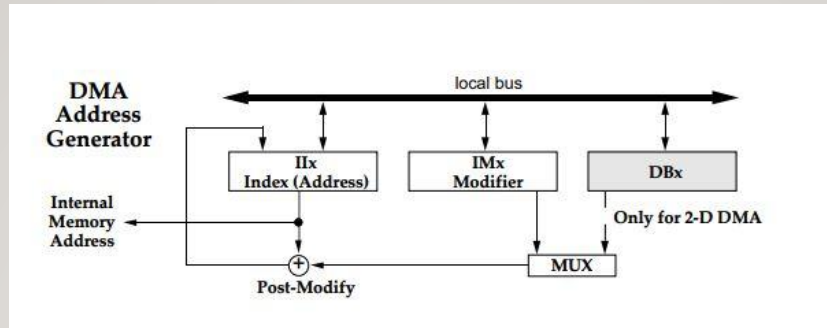


Fig 14: DMA Address Generator

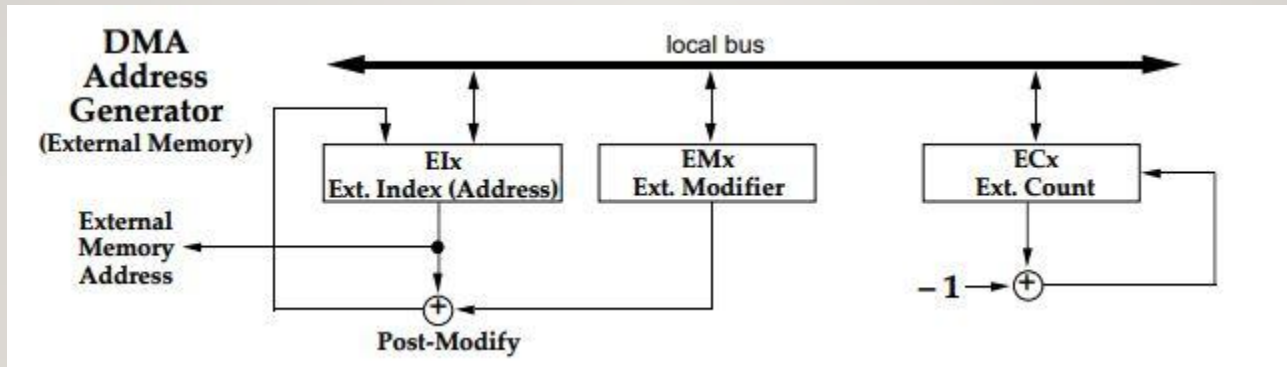


Fig 15: DMA Address Generator (External Memory)

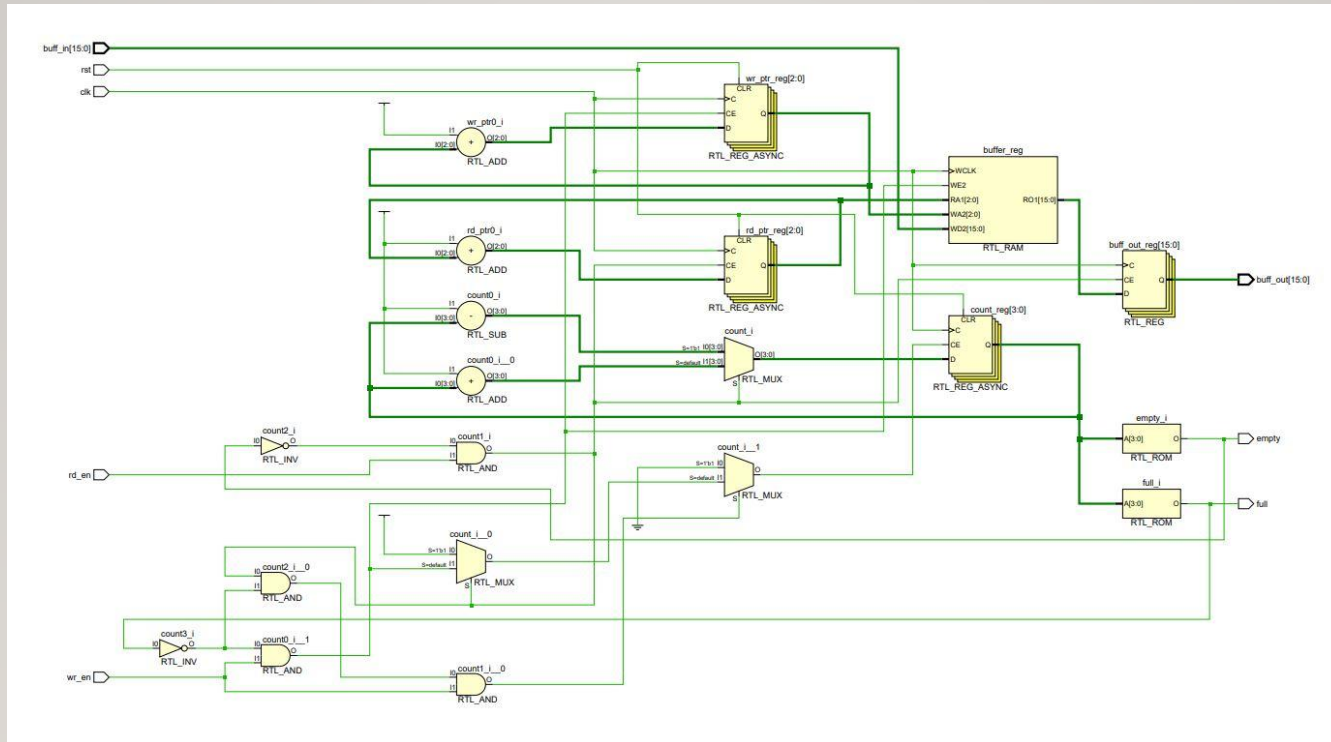
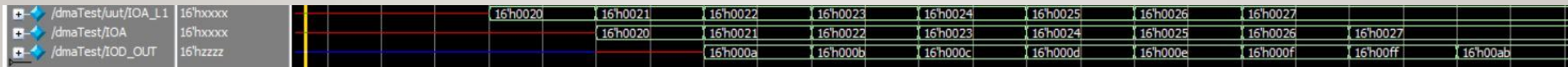
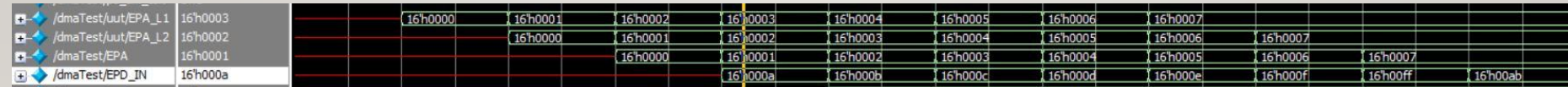


Fig 16: Synchronous FIFO

Pipelining



Internal memory access



External memory access

Fig.17- 3 stage Pipelining

Internal to external memory data transfer

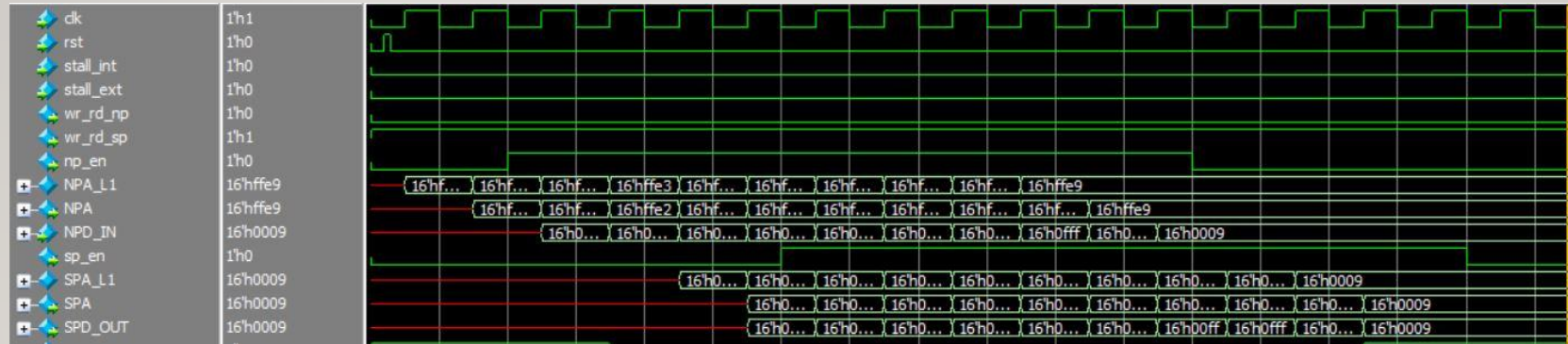


Fig.18 Internal to External Memory Data Transfer

Internal mem read: n^{th} cycle

External mem write: n^{th} cycle

02-06-22

External to internal memory data transfer (n+1)



Fig.19 External Memory to Internal Data Transfer

External read: $n+1$ cycle

Internal write: $n+1$ cycle

02-06-22

External to internal memory data transfer (n+2)



Fig.20 External to Internal Memory Data Transfer

External read: $n+2$ cycle

Internal write: $n+1$ cycle

02-06-22

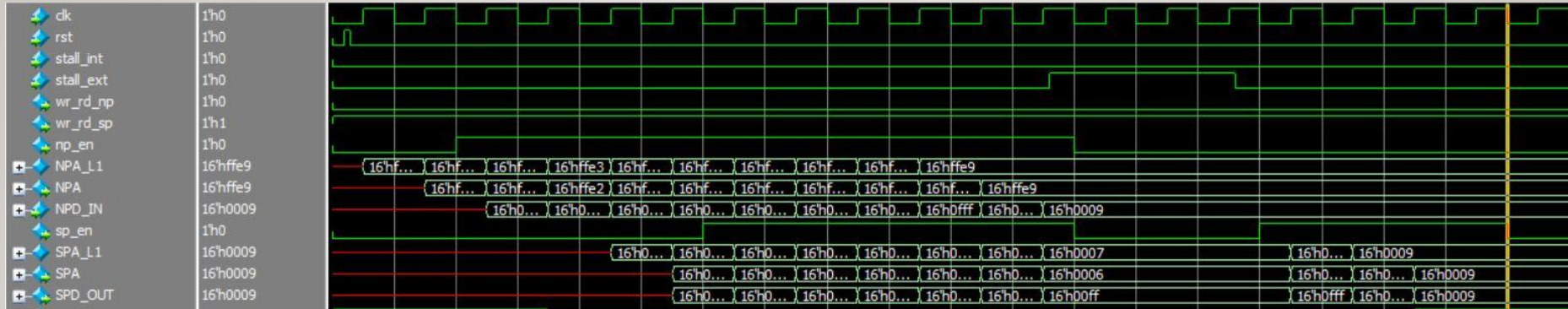
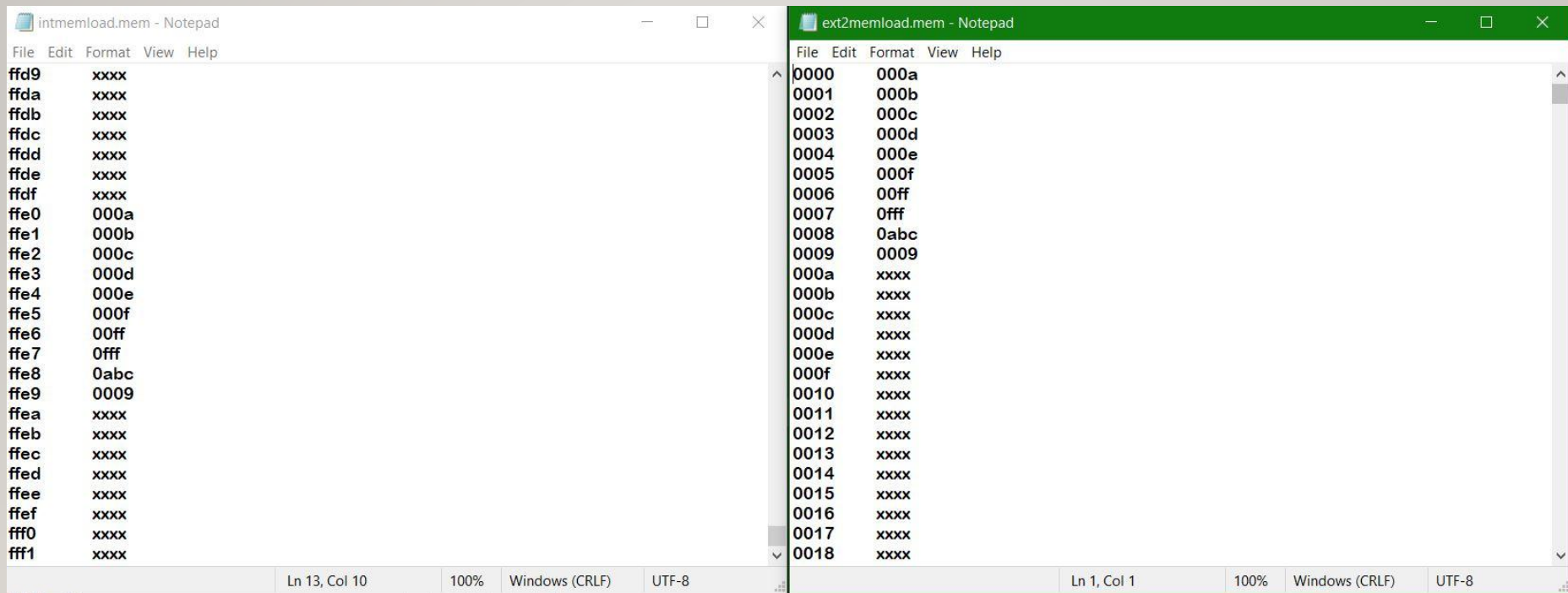


Fig.21 Stalling in pipelining



intmemload.mem - Notepad

Address	Value
ffd9	xxxx
ffda	xxxx
ffdb	xxxx
ffdc	xxxx
ffdd	xxxx
ffde	xxxx
ffdf	xxxx
ffe0	000a
ffe1	000b
ffe2	000c
ffe3	000d
ffe4	000e
ffe5	000f
ffe6	00ff
ffe7	0fff
ffe8	0abc
ffe9	0009
ffea	xxxx
ffeb	xxxx
ffec	xxxx
ffed	xxxx
ffee	xxxx
ffef	xxxx
fff0	xxxx
fff1	xxxx

Ln 13, Col 10 100% Windows (CRLF) UTF-8

ext2memload.mem - Notepad

Address	Value
0000	000a
0001	000b
0002	000c
0003	000d
0004	000e
0005	000f
0006	00ff
0007	0fff
0008	0abc
0009	0009
000a	xxxx
000b	xxxx
000c	xxxx
000d	xxxx
000e	xxxx
000f	xxxx
0010	xxxx
0011	xxxx
0012	xxxx
0013	xxxx
0014	xxxx
0015	xxxx
0016	xxxx
0017	xxxx
0018	xxxx

Ln 1, Col 1 100% Windows (CRLF) UTF-8

Fig.22 Memory contents

02-06-22

Scatter Gather DMA

- DMA transfer of contiguous to non contiguous memory and vice-versa.
- An FSM model is designed for supporting this complex data transfer.
- Uses a portion of internal memory to store the non-contiguous memory locations.
- Avoid Segmentation in memory.

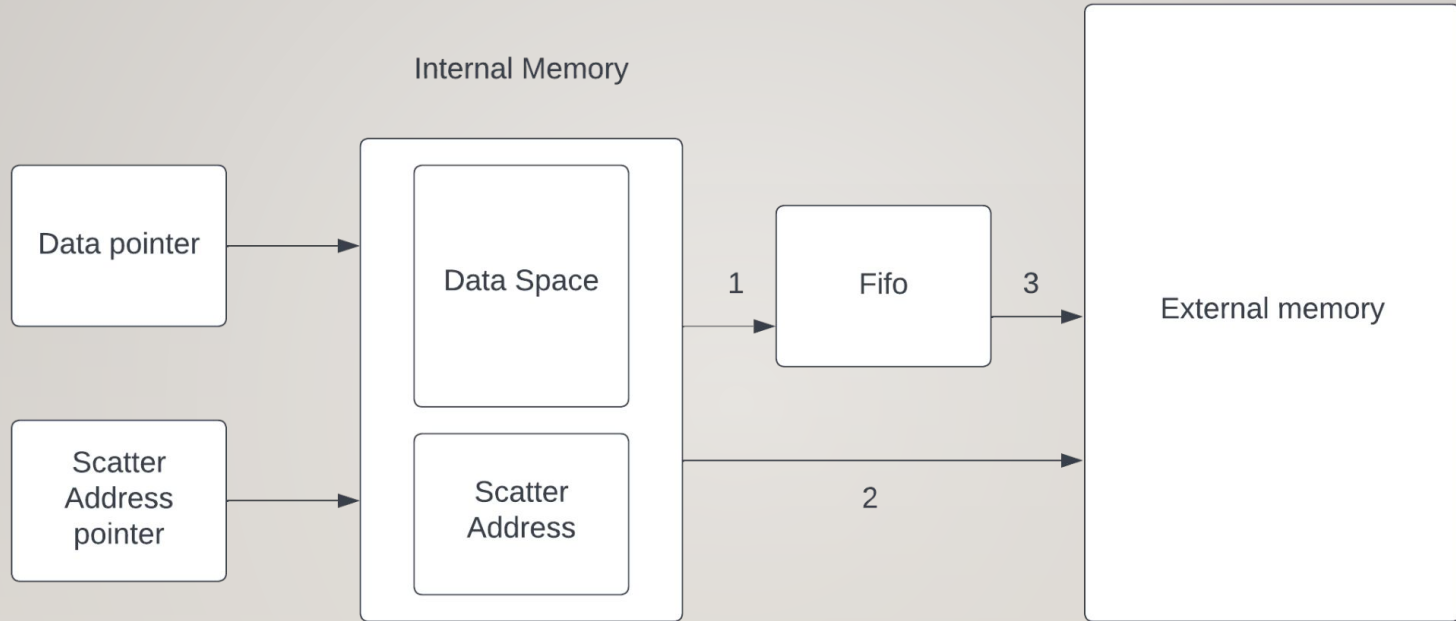


Fig.23 Scatter Gather Flow diagram

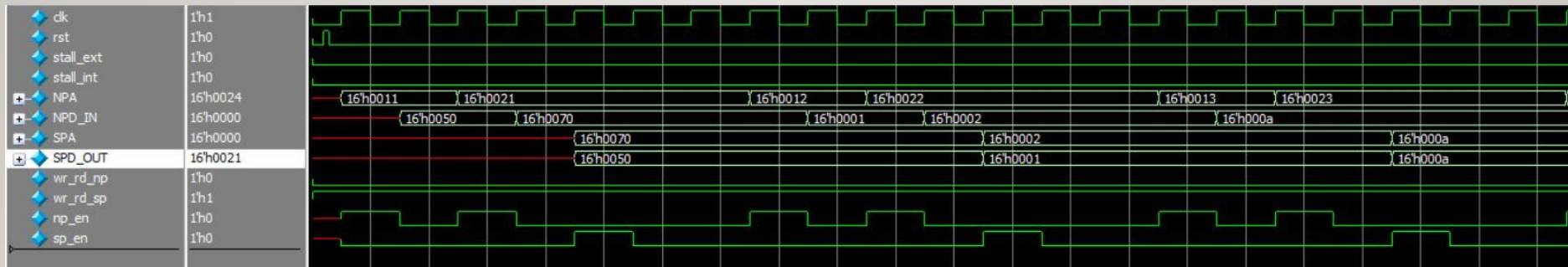


Fig.24 Scattering of data

Chaining Of DMA

- DMA controller to auto initialize itself between multiple DMA transfers.
- In chained DMA operation, automatically sets up another DMA transfer when the entire contents of the current buffer have been transmitted or received.
- It uses CP register

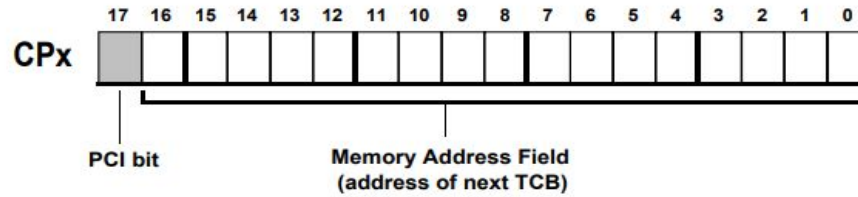


Fig 25: Chain Pointer Register & PCI Bit

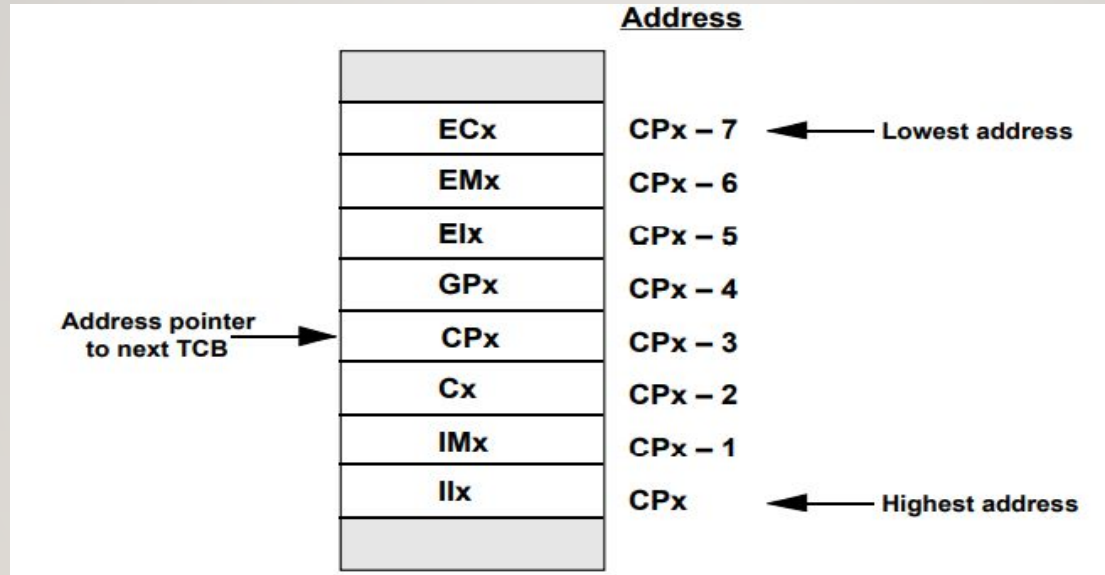


Fig 26: TCB Setup In Memory

Memory controller

- Input signals - Address (16 bit)
 - Data (16 bit)
 - Ready(1 bit)
 - Access Enable(1 bit)
 - Clock
 - Read Data from Memory(16 bit)

- Output signals - Write Data to Memory(16 bit)
 - Write Address to Memory (16 bit)
 - Read Data from Memory (16 bit)
 - Read Out (16 bit)

Block Diagram

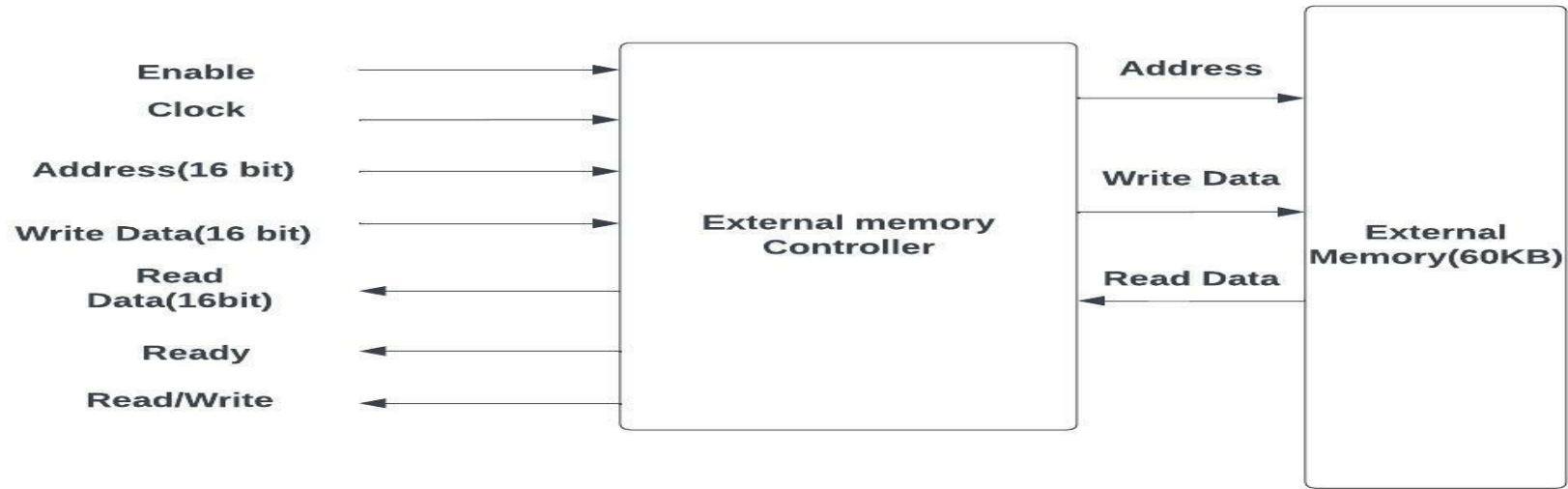
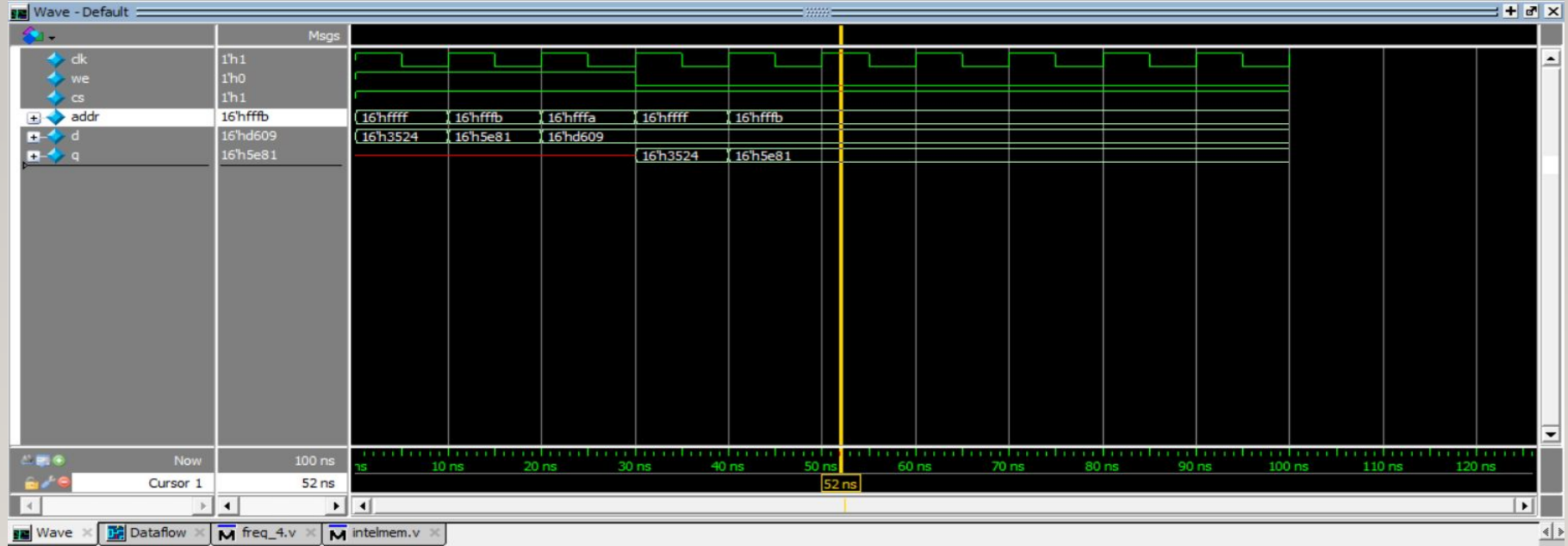


Fig.28 Memory Controller

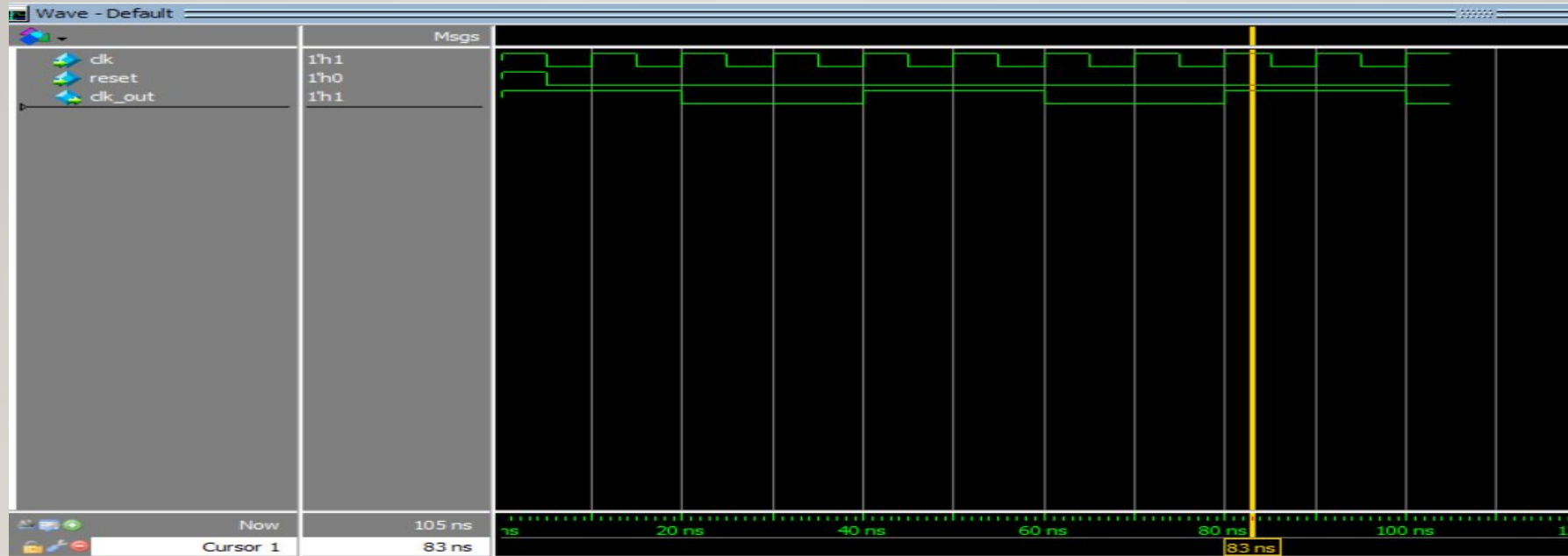
Write and Reading data to Memory



02-06-22

Fig.29 Testing of memory

Testing Frequency Divider



02-06-22

Fig.30 Testing of Frequency Divider

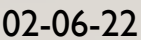
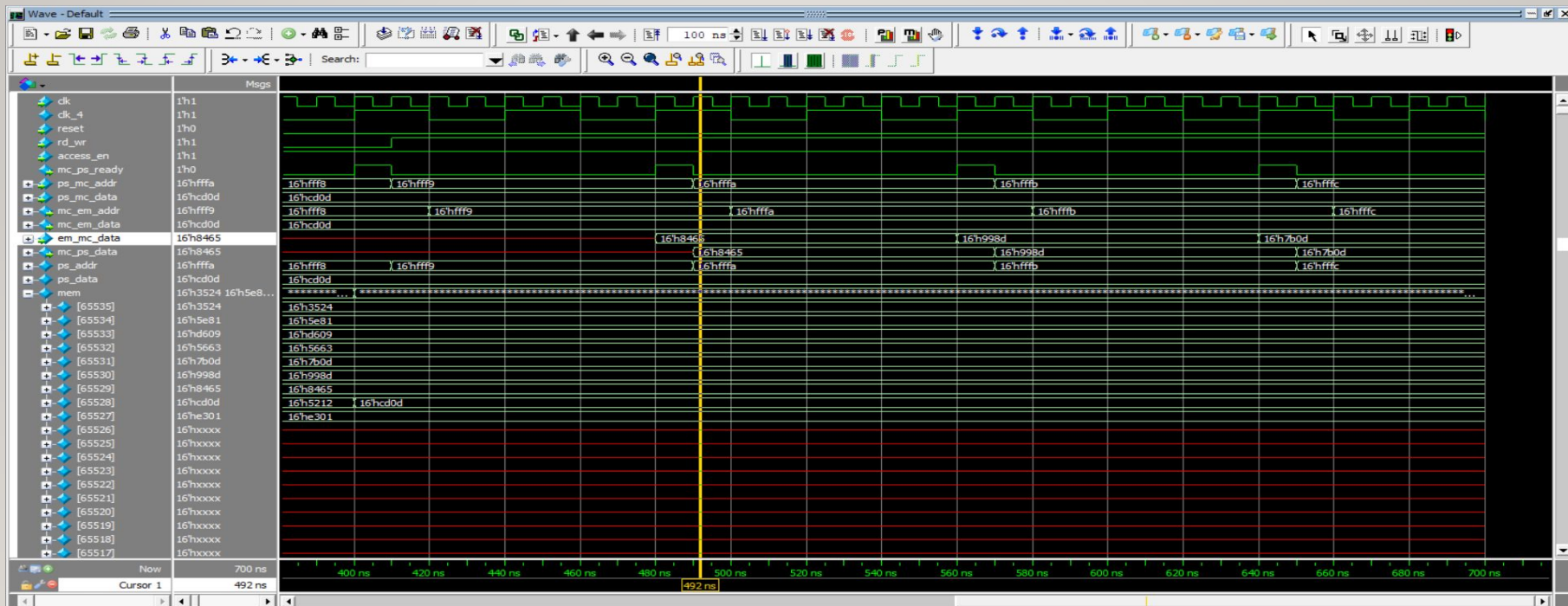


Fig.31 Memory Controller operation in write operation

Memory Controller(Read Operation)

43



02-06-22

Fig.32 Memory Controller operation in read operation

Future Scope

- Serial Port and Link port can be implemented and integrated to already implemented blocks.
- Bulk Data transfer in IoT applications.
- Program loading into program memory during booting.



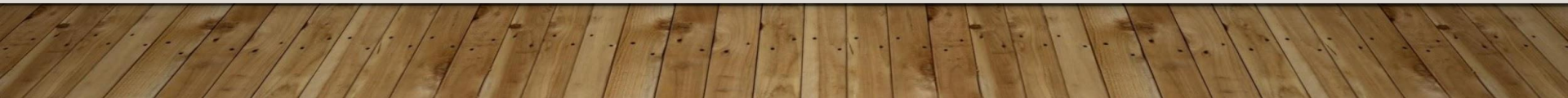
Results

- Implemented different mode of implementation of DMA Controller
 - Normal mode
 - Chaining mode
 - Scatter-Gather mode
- Implemented the Memory control interface to handle read/write request from DMA and core processor



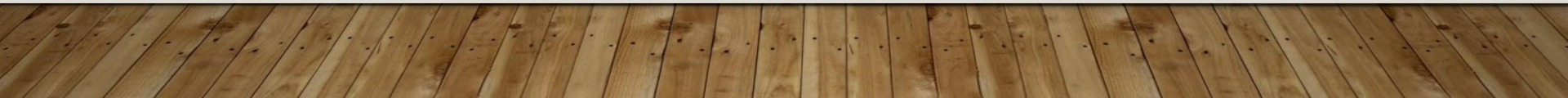
CONCLUSION

- Scatter-Gather DMA is introduced to the existing dma architecture.
- This DMAC can perform both contiguous and non – contiguous memory transfer.
- Memory controller for external memory interfacing.
- Core will support functioning of the Micro RISC ISA processor for iot application .

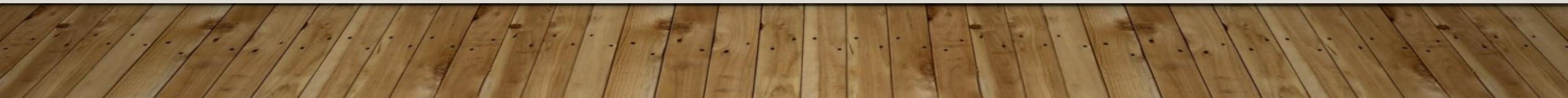


REFERENCES

- [1] ADSP-2106x SHARC Processor User's Manual, Revision 2.1.
- [2] H. Morales, C. Duran and E. Roa, "A Low-Area Direct Memory Access Controller Architecture for a RISC-V Based Low-Power Microcontroller," 2019 IEEE 10th Latin American Symposium on Circuits & Systems (LASCAS), 2019, pp. 97-100, doi: 10.1109/LASCAS.2019.8667579.
- [3] S. Min, M. Alian, W. -M. Hwu and N. S. Kim, "Semi-Coherent DMA: An Alternative I/O Coherency Management for Embedded Systems," in IEEE Computer Architecture Letters, vol. 17, no. 2, pp. 221-224, 1 July-Dec. 2018, doi: 10.1109/LCA.2018.2866568.
- [4] N. Vujic, F. Cabarcas, M. Gonzalez Tallada, A. Ramirez, X. Martorell and E. Ayguade, "DMA++: On the Fly Data Realignment for On-Chip Memories," in IEEE Transactions on Computers, vol. 61, no. 2, pp. 237-250, Feb. 2012, doi: 10.1109/TC.2010.255.
- [5] H. Jung, S. Jung and Y. H. Song, "Architecture exploration of flash memory storage controller through a cycle accurate profiling," in *IEEE Transactions on Consumer Electronics*, vol. 57, no. 4, pp. 1756-1764, November 2011, doi: 10.1109/TCE.2011.6131151.



- [6] N. Surana and J. Mekie, "Energy Efficient Single-Ended 6-T SRAM for Multimedia Applications," in *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 66, no. 6, pp. 1023-1027, June 2019, doi: 10.1109/TCSII.2018.2869945.
- [7] H. Kavianipour, S. Muschter and C. Bohm, "High performance FPGA-based DMA interface for PCIe," *2012 18th IEEE-NPSS Real Time Conference*, 2012, pp. 1-3, doi: 10.1109/RTC.2012.6418352.
- [8] L. Rota, M. Caselle, S. Chilingaryan, A. Kopmann and M. Weber, "A PCIe DMA Architecture for Multi-Gigabyte Per Second Data Transmission," in *IEEE Transactions on Nuclear Science*, vol. 62, no. 3, pp. 972-976, June 2015, doi: 10.1109/TNS.2015.2426877.
- [9] Mrinal J Sarmah, Bokka Abhiram , Anil kumar A "Method for Booting an All programmable System on-Chip over PCI Express Link", 2017 July IEEE.
- [10] Design of Processing-"Inside"-Memory Optimized for DRAM Behaviors WON JUN LEE 1, CHANG HYUN KIM1, YOONAH PAIK1, June 2019.



THANK YOU

02-06-22

