Journal on Electronic and Automation Engineering

Vol: 3(2), June 2024

REST Publisher; ISSN: ISSN: 2583-6951 (Online)

Website: https://restpublisher.com/journals/jeae/

DOI: https://doi.org/10.46632/jeae/3/2/3

Implementation of a Multiclocked Pipelined Processor Based on RISc-V using RV321

*** Kandula Dharani, B. Raghavaiah**
Ramachandra College of Engineering, Eluru, Andhra Pradesh, India.
*Corresponding author Email: dharani.kandulaa@gmail.com

Abstract: *This study's primary objective is to create a 32-bit pipelined processor based on the open-source RV32I Version 2.0 RISC-V ISA that operates across several clock domains. A processor known as a RISC (Reduced Instruction Set Computer) employs less hardware than a CISC (Complex Instruction Set Computer) in order to reduce the complexity of the instruction set and accelerate the execution time per instruction. In addition, we built this processor with five pipelining layers, which allows for concurrent processing of instructions. All of the procedures are thoroughly explained, supported with the required block diagrams. To guarantee that variable delays, such as clock skew and meta-stability, are avoided inside the stage pipeline registers, multiple clock domains using two clock sources are employed.*
Keywords: *RISC, Multiple Clock Domains, Pipeline, Verilog, Processor.*

1. INTRODUCTION

Generation of Microprocessors and Types of Microprocessors:

First-generation: The first generation of microprocessors, which included the INTEL 4004, Rockwell International PPS-4, and INTEL 8008, among others, debuted in 1971 and 1972.

Second generation: The second generation of 8-bit microprocessors was created between 1973 and 1978. They created processors such the Motorola 6800, INTEL 8085, and 6801.

Third generation: 16-bit CPUs, including the Motorola 68000, 68010, and INTEL 8086/80186/80286, were introduced in this generation. From 1979 to 1980, this generation used HMOS technology.

Fourth generation: The fourth generation was in existence from 1981 until 1995. 32-bit CPUs were developed using HMOS manufacturing. Among the well-known processors of this generation are the Motorola 68020 and Intel 80386.

Fifth Generation: Since 1995, we have been a part of the fifth generation. The PENTIUM, Celeron, twin, quad, and octa-core processors were among the 64-bit CPUs available.

RISC vs CISC:
Reduced Instruction Set computers and Complex Instruction Set computers are the two types of architectures that have provided the foundation for microprocessors and microcontrollers over the past few decades. The length of CISC instructions can vary, and they are encoded to perform more micro operations per instruction. Because of this, instructions take longer to execute on CISC processors due to their intricate design. Compared to a CISC processor, a RISC processor's instructions all have the same length, which makes decoding simpler. Because of its effective design, which can be used to low-power and high-speed processing applications, RISC is extensively employed. It only supports a small number of addressing modes; the only instructions required to access external memory are LOAD and STORE. Therefore For sophisticated task execution, CISC computers mostly rely on hardware, whereas RISC processors primarily rely on software.

Reduced Instruction Set Architecture (RISC)
The main objective is to use an instruction set with only a few essential loading, evaluating, and storing phases in order to simplify hardware. A store command saves the data, much like a load command does. The primary idea behind the Complex Instruction Set Architecture (CISC) is that all loading, evaluating, and storing operations will be handled by a single instruction, much as how a multiplication command will handle loading, evaluating, and saving data. This is why CISC is difficult.

Complex Instruction Set Architecture (CISC)
It's difficult because the core idea is that all loading, evaluating, and storing operations will be handled by a single instruction, much as how a multiplication command will handle loading, evaluating, and saving data.

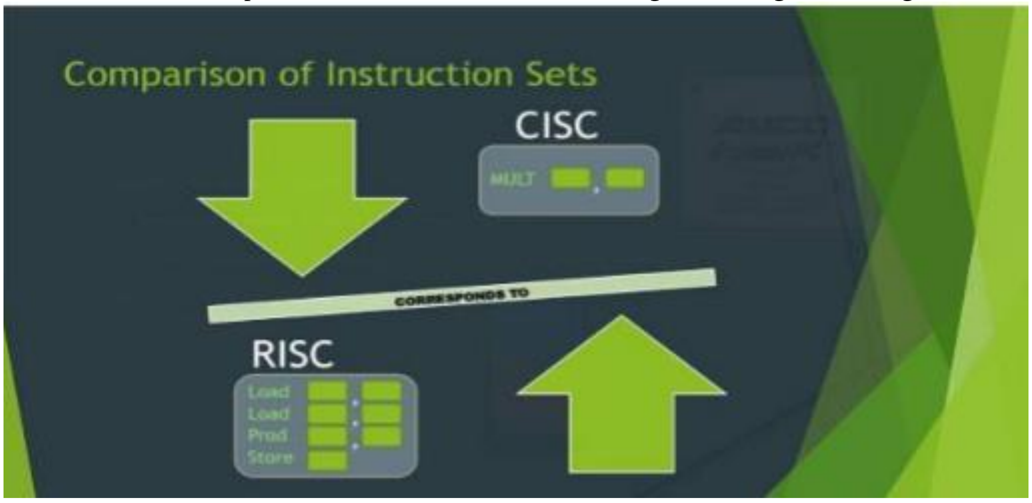


FIGURE 1. RISC vs CISC

2. LITERATURE SURVEY

Low power high speed multiplier using parallel multiplication by Madhav Thiyagarajan
This paper presents a low power and high-speed row bypassing multiplier. The primary power reductions are obtained by tuning off MOS components through multiplexers when the operands of multiplier are zero. Analysis of the conventional DSP applications shows that the average of zero input of operand in multiplier is 73.8 percent. Therefore, significant power consumption can be reduced by the proposed bypassing multiplier. The proposed multiplier adopts ripple-carry adder with fewer additional hardware components. In addition, the proposed bypassing architecture can enhance operating speed by the additional parallel architecture to shorten the delay time of the proposed multiplier. Both unsigned and signed operands of multiplier are developed. Post-layout simulations are performed with standard TSMC 0.18 μm CMOS technology and 1.8 V supply voltage by Cadence Spectre simulation tools. Simulation results show that the proposed design can reduce power consumption and operating speed compared to those of counterparts. For a 16×16 multiplier, the proposed design achieves 17 and 36 percent reduction in power consumption and delay, respectively, at the cost of 20 percent increase of chip area in comparison with those of conventional array multipliers. In addition, the proposed design achieves averages of 11 and 38 percent reduction in power consumption and delay with 46 percent less chip area in comparison with those counter parts for both unsigned and signed multipliers. The proposed design is suitable for low power and highspeed arithmetic applications. **FPGA**

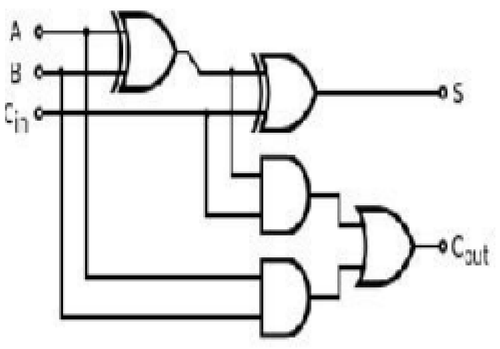
Implementation of Multiply Accumulate (MAC) Unit based on Block Enable Technique by Tasmia Shaikh, Manjunatha Beleri.
Power dissipation is one of the most important design objectives in integrated circuit, after speed. Digital signal processing (DSP) circuits whose main building block is a Multiply Accumulate (MAC) unit. High speed and low power MAC unit is desirable for any DSP processor. This is because speed and throughput rate are always the concerns of DSP system. This paper explores the design of low power MAC unit with block enable technique to reduce power dissipation. The whole MAC unit is implemented using 90-nm CMOS process technology. The whole MAC unit is operated at 314.268MHz with 1.5V supply voltage. The result analysis shows that the power consumption is reduced by using block enable technique.

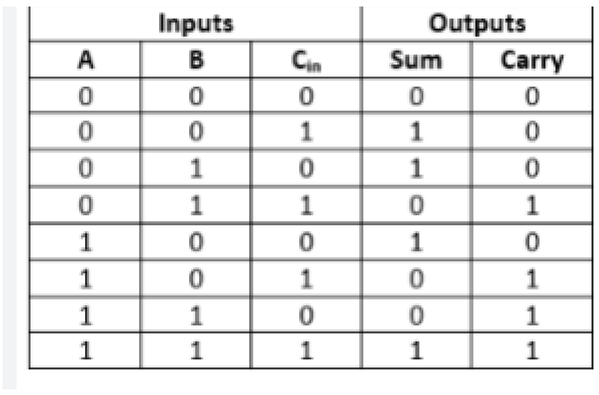
A Reduce Bit Multiplication.
A reduced-bit multiplication algorithm based on the ancient Vedic multiplication formulae is proposed in this paper. Both the Vedic multiplication formulae, [12] are first discussed in detail. being a general multiplication formula, is

equally applicable to all cases of multiplication. It is applied to the digital arithmetic and is shown to yield a multiplier architecture which is very similar to the popular array multiplier. Due to its structure, it leads to a high carry propagation delay in case of multiplication of large numbers. Nikhilam Sutra, on the other hand, is more efficient in the multiplication of large numbers as it reduces the multiplication of two large numbers to that of two smaller numbers. The framework of the proposed algorithm is taken from this Sutra and is further optimized by use of some general arithmetic operations such as expansion and bit- shifting to take advantage of bit-reduction in multiplication. We illustrate the proposed algorithm by reducing a general 4£4-bit multiplication to a single 2 £ 2-bit multiplication operation.

3. EXISTING SYSTEM

32 BIT Multi Clocked Pipelined Processor Based On RISCV:
The current approach, which uses binary multipliers to create a MAC unit, frequently results in high power consumption and high route latency while carrying out addition propagations, including final adds in multiplications and additions in accumulations. Two separate blocks make up a MAC unit: an accumulator, also known as an accumulate adder, and a multiplier. A (2N+α-1)-bit accumulator (adder) and an N-bit multiplier make up an N-bit MAC unit. α is the number of guard bits needed to prevent overflow, which is caused by lengthy multiply-accumulate operations. Numerous earlier studies focused on the optimization of the adder and multiplier, respectively. Usually, a multiplier consists of three stages. The partial product generation (PPG) process is the initial stage. For an unsigned multiplication, AND gates, for instance, can be utilized to create a partial product matrix (PPM). The partial product reduction (PPR) procedure is the second phase. The PPM can be divided into two rows by applying the Wallace tree technique or the Dadda tree approach. The last addition is made in the third stage. The total of the last two rows is calculated using an adder, referred to as the final adder. A (2N-1)-bit adder is needed for the final addition when using an N-bit multiplier.





Inputs			Outputs	
A	B	C_{in}	Sum	Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

FIGURE 2. Full Adder

Numerous applications in digital signal processing and other fields make use of multipliers. Owing to developments in modern technology, a large number of academics have focused mostly on design elements in order to improve performance. Several design objectives include reduced area, fast speed, precision, low power consumption, and regular arrangement. Multiplexers, adders, and MAC are just a few of the computational blocks found in DSP processors. In comparison to earlier iterations, these blocks operate and execute at a faster pace. Two aspects affect multiplier execution speed: multiplier architecture and semiconductor technology. The fundamental component of digital multiplexers are adders, which are used to make a sequence of repeated adds. Increasing the adder's operating speed is necessary to speed up the multiplier action.

4. PROPOSED SYSTEM

The processor's job is to efficiently carry out each and every instruction provided in accordance with the machine language. Arithmetic and Logical Unit, or ALU, is a combinational circuit. This unit is made to execute different integers with different sets of instructions. Operation code (opcode) plus a few operands make up the instruction (machine word) that an ALU input in a processor receives. The ALU is subsequently instructed by the opcode as to which and what operation to carry out, and these operands are employed in the procedure.
A little collection of data storage facilities is referred to as the Register Bank. The ALU verifies the bits and signals whether the operation was successful by storing the result of the operation in an accumulator, which is then stored in a storage register. Should the operation fail, a status message also referred to as a Z-Flag or status register will be shown.

Its job is to run programs and provide effective operation for the data kept in memory. A processor is nothing more than a collection of commands that tell a computer how to do a certain task. The command to be carried out is stored in the control unit. The address register, data register, and instruction register are among the registers found in a CPU. The CPU's performance is to retrieve, decode, and carry out memory operations in accordance with the registers. Decoding the op-code, identifying the instruction, figuring out which operands are in memory, getting the operands from memory, and giving a processor an order to carry out the instruction are all part of the IR task. This is accomplished with the aid of a control unit, which produces the timing signals needed to regulate the several processing components involved in carrying out the command. MAR, also known as the address buffer, stores the current instruction in the memory location after the PC increments to the next address. The address in the program counter is applied to memory. Binary words, which indicate a word's position in RAM, fill the MAR entirely. The instruction is kept at this area.

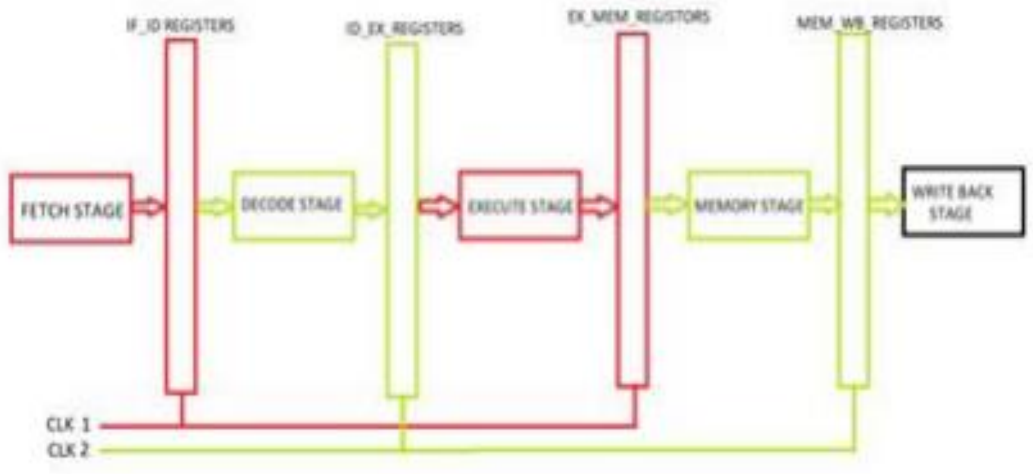


FIGURE 3. Block Diagram of Processor

The next instruction to be performed is indicated by the program counter. During the whole instruction cycle, the processor retrieves the instruction from the memory address indicated by the program counter and loads it into the instruction register. Because it creates the timing and control signals for the processes carried out by the CPU, the control unit is a crucial component of all computers and systems. Since the ALU regulates the signal transfer between the processor, memory, and other buses, communication in this case is between the ALU and main memory.

Pipelining concept:

Pipelining is a method of breaking down a sequential process into smaller processes, each of which is carried out in a separate segment that runs concurrently with all other segments. The ability to have many calculations running simultaneously in separate parts is a pipeline technique's most crucial feature. Every pipeline segment has a register assigned to it, which allows for computation to overlap. Each segment can function independently of the others at the same time thanks to the isolation that the registers offer between them. All that is needed to depict the structure of a pipeline organization is to include an input register after a combinational circuit for each segment. To have a better grasp of the pipeline architecture, let us look at an example of a combined multiplication and addition operation. A series of integers is used for the combined multiplication and addition operation, such as
Ai* Bi + Ci for i = 1, 2, 3, ..., 7

The operation that needs to be done on the numbers is broken down into smaller operations, each of which needs to be carried out in a pipeline segment. The following are the sub-operations carried out in each pipeline segment:
R1 ← Ai, R2 ← Bi Input Ai, and Bi
R3 ← R1 * R2, R4 ← Ci Multiply, and input Ci
R5 ← R3 + R4 Add Ci to product

The combined operations and individual sub-operations carried out in each pipeline section are shown in the block diagram that follows.

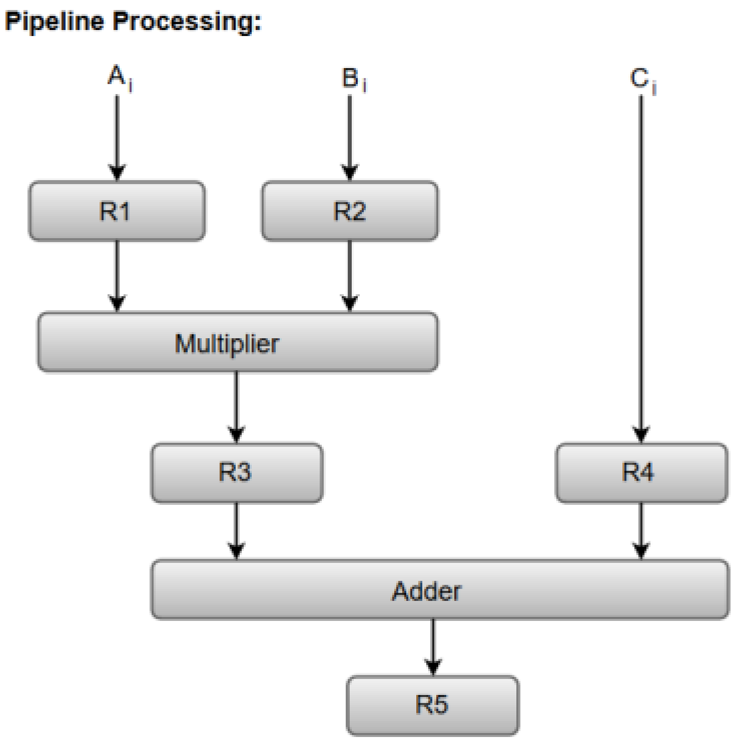
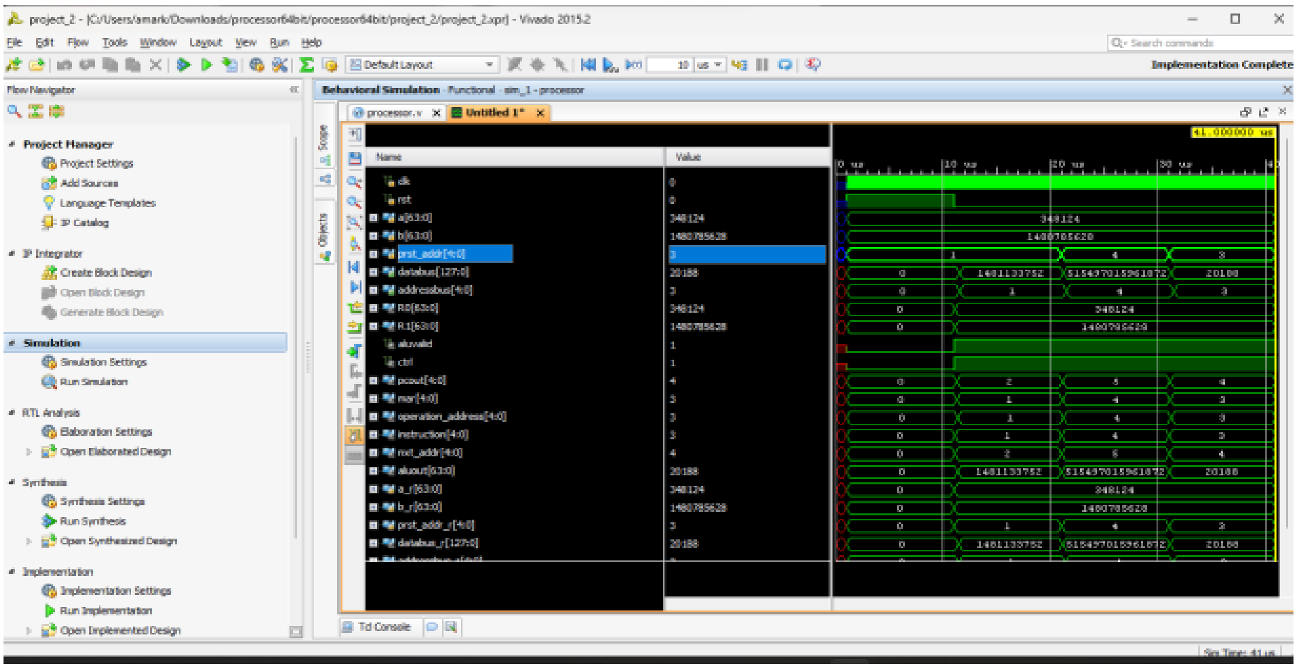


FIGURE 4. Pipeline process

The data is stored in registers R1, R2, R3, and R4, and the combinational circuits function in a specific section. The combinational circuit's output in one segment is applied as the input register for the subsequent segment. For example, the combinational adder circuit uses register R3 as one of its input registers, as can be seen from the block diagram. Both the data stream and the instruction stream may undergo pipeline processing. To perform tasks like fetch, decode, and execute instructions, the majority of digital computers with complicated instructions need an instruction pipeline.

5. RESULTS



Name	Value
clk	0
rst	0
a[63:0]	348124
b[63:0]	1480785628
prst_addr[4:0]	3
databus[127:0]	20188
addressbus[4:0]	3
R0[63:0]	348124
R1[63:0]	1480785628
aluvalid	1
ctrl	1
pcout[4:0]	4
mar[4:0]	3
operation_address[4:0]	3
instruction[4:0]	3
nxt_addr[4:0]	4
aluout[63:0]	20188
a_r[63:0]	348124
b_r[63:0]	1480785628
prst_addr_r[4:0]	3
databus_r[127:0]	20188

FIGURE 5.

PROPOSED ELABORATED DESIGN

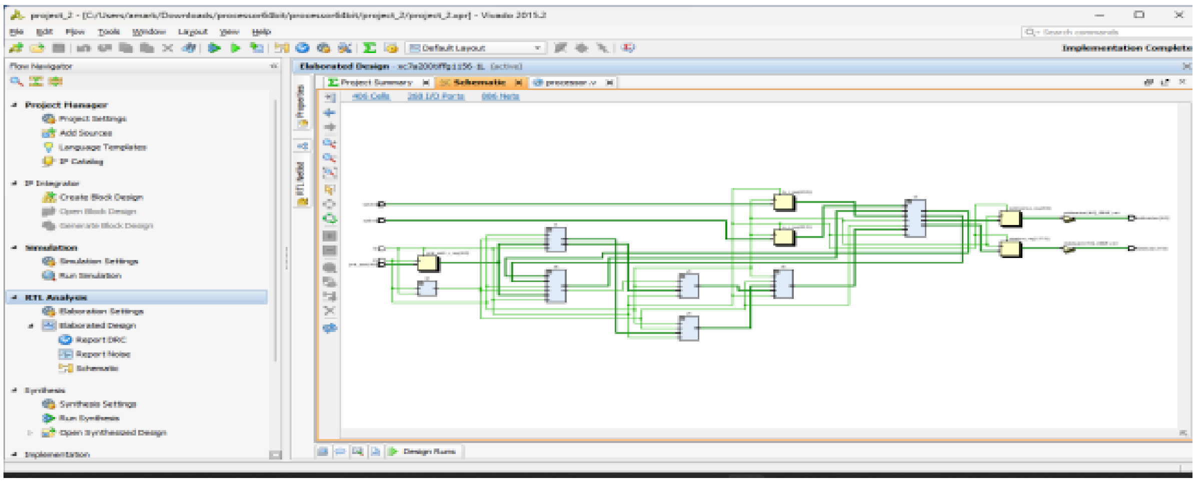
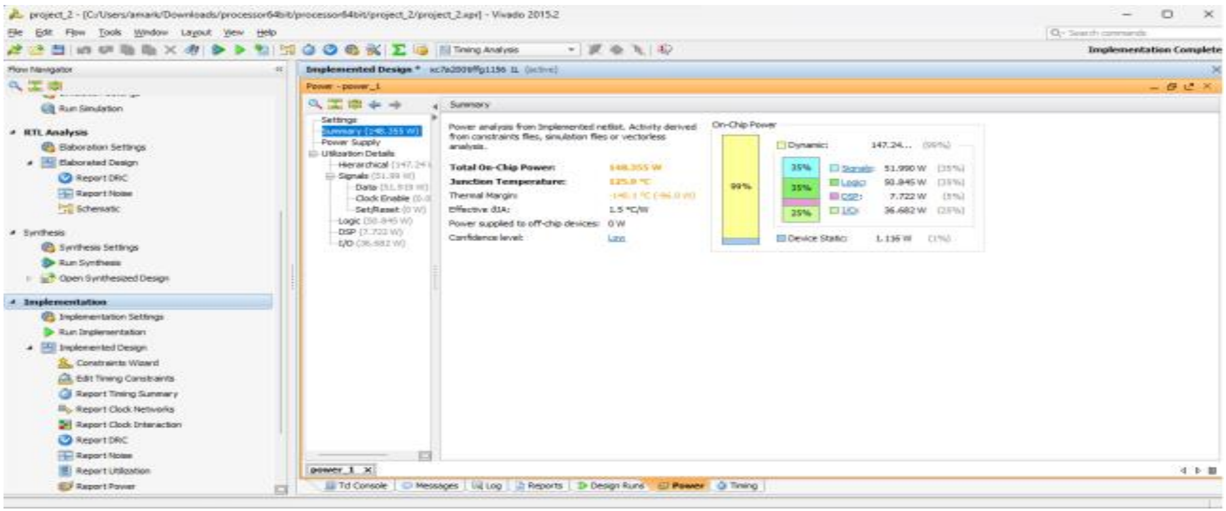


FIGURE 6.

Power Results



Summary

Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

Total On-Chip Power:	148.355 W
Junction Temperature:	125.0 °C
Thermal Margin:	-140.1 °C (-96.0 W)
Effective ϑJA:	1.5 °C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Dynamic:	147.24...	(99%)
Signals:	51.990 W	(35%)
Logic:	50.845 W	(35%)
DSP:	7.722 W	(5%)
I/O:	36.682 W	(25%)
Device Static:	1.136 W	(1%)

FIGURE 7.

Existing Model Results

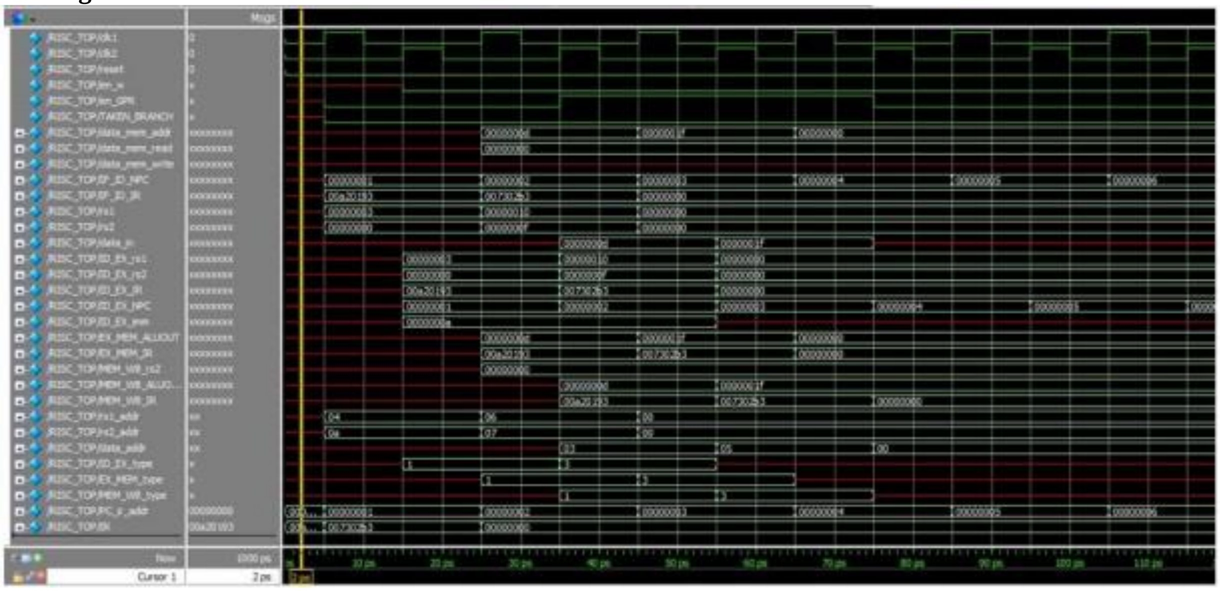


FIGURE 8.

TABLE 1. Compression of Proposed model vs existing model

ALGORITHM	POWER CONSUMPTION	LUT UTILISATION	TIME DELAY
Existing method (Rijndael Algorithm)	148.388W	4722	48.956 ns
Proposed methodology (Area-Optimized AES Algorithm Using Pipeline Technology)	137.398W	2356	32.025ns

6. CONCLUSION

This article discusses the architecture of a multiple clock domain pipelined RISC processor. We built the architecture with more than thirty instructions based on RV32I ISA V 2.0. By using pipelining to reduce the number of clocks per instruction, we were able to enhance throughput. The design is modeled and synthesized using Xilinx Vivido. Each instruction was verified independently using a large number of simulations. By adopting non-overlapping two-phase clocks, meta-stability and unanticipated delays, such as clock skew, may be avoided. Thus, our goal of minimizing latency is partially achieved.

REFERENCES

[1]. M. N. Topiwala and N. Saraswathi, "Implementation of a 32-bit MIPS based RISC processor using Cadence," 2014 IEEE International Conference on Advanced Communications, Control and Computing Technologies, 2014, pp. 979-983, doi: 10.1109/ICACCCT.2014.7019240.

[2].] S. P. Ritpurkar, M. N. Thakare and G. D. Korde, "Synthesis and Simulation of a 32Bit MIPS RISC Processor using VHDL," 2014 International Conference on Advances in Engineering & Technology Research (ICAETR - 2014), 2014, pp. 1-6, doi: 10.1109/ICAETR.2014.7012843.

[3].] S. P. Ritpurkar, M. N. Thakare and G. D. Korde, "Design and simulation of 32-Bit RISC architecture based on MIPS using VHDL," 2015 International Conference on Advanced Computing and Communication Systems, 2015, pp. 1-6, doi: 10.1109/ICACCS.2015.7324067.

[4]. D. K. Dennis et al., "Single cycle RISC-V micro architecture processor and its FPGA prototype," 2017 7th International Symposium on Embedded Computing and System Design (ISED), 2017, pp. 1-5, doi: 10.1109/ISED.2017.8303926.

[5].] S. Palekar and N. Narkhede, "32-Bit RISC processor with floating point unit for DSP applications," 2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), 2016, pp. 2062-2066, doi: 10.1109/RTEICT.2016.7808202.

[6]. A. Raveendran, V. B. Patil, D. Selvakumar and V. Desalphine, "A RISC-V instruction set processor-micro-architecture design and analysis," 2016 International Conference on VLSI Systems, Architectures, Technology and Applications (VLSI-SATA), 2016, pp. 1- 7, doi: 10.1109/VLSI-SATA.2016.7593047.

[7]. J. V. Kumar, B. Nagaraju, C. Swapna and T. Ramanjappa, "Design and development of FPGA based low power pipelined 64-Bit RISC processor with double precision floating point unit," 2014 International Conference on Communication and Signal Processing, 2014, pp. 1054-1058, doi: 10.1109/ICCSP.2014.6950008.

[8].] R. J. L. Austria, A. L. Sambile, K. M. Villegas and J. N. T. Tabing, "Design of an 8- bit five stage pipelined RISC microprocessor for sensor platform application," TENCON 2017 - 2017 IEEE Region 10 Conference, 2017, pp. 2110-2115, doi: 10.1109/TENCON.2017.8228209.

[9]. R. Aneesh. and K. Jiju., "Design of FPGA based 8-bit RISC controller IP core using VHDL," 2012 Annual IEEE India Conference (INDICON), 2012, pp. 427-432, doi: 10.1109/INDCON.2012.6420656.