A novel reduced instruction set computer-communication processor design using field programmable gate array

Joseph Anthony Prathap[1], Sai Ramesh[2]
[1]Department of Electronics and Communications, Presidency University, Bengaluru, India
[2]Puredata Solution LLP, Hyderabad, India

Article Info

Article history:

Received Sep 7, 2022
Revised Dec 26, 2022
Accepted Jan 14, 2023

Keywords:

Communication
Field programmable gate array
Pipelining
Real time applications
RISC processor

ABSTRACT

In this paper, a novel reduced instruction set computer (RISC)-communication processor (RCP) has been designed with 32-bit operations which access 64-bit instruction format and implemented using field programmable gate array (FPGA). The design of the RISC processor is facilitated with communication operations like basic signals sine, cosine, and square, and modulation schemes like amplitude modulation, amplitude shift keying, analog, and digital quadrature amplitude modulation. Additionally, application-oriented operations like a traffic light, digital clock, and linear feedback shift register are included in the design. The pipeline mechanism is incorporated in the design to enhance the performance characteristics of the processor, hence allowing the execution of the instructions more effectively. Also, the design is implemented with Xilinx Virtex 7 family FPGA. The device utilization analysis of the proposed FPGA along with different FPGA families is evaluated and compared.

Corresponding Author:

Joseph Anthony Prathap
Department of Electronics and Communications, Presidency University
Rajanakunte, Yelahanka, Bengaluru, India
Email: joseph.anthony@presidencyuniversity.in

1. INTRODUCTION

A reduced instruction set computer (RISC) is a computer instruction set architecture (ISA) that allows a computer's microprocessor to have fewer cycles per instruction (CPI). RISC architectures are utilized in a wide range of applications ranging from cellular telephones, and tablet computers to some of the world's fastest supercomputers such as summit. RISC processors can perform arithmetic and logic, shifting and rotating, read and write by using instruction set architecture (ISA). At the ISA level, there are two categories of processors namely RISC and complex instruction set computers (CISC). The CISC deals with a complex instruction set and as a result, it has many addressing modes, execution time, and length of instructions varies compared to RISC which is quite opposite as it offers a fixed instruction format and better execution.

In addition to that, the concept of pipelining is implemented to enhance the performance of the processor. Pipelining is a process of executing instructions in a sequence through different phases, thereby increasing throughput. The pipe lining has five stages: i) instruction fetch (IF), ii) instruction decode (ID), iii) execute (EX), iv) read/write (RW), and v) memory access (MA) to the clock pulse.

The pipelining feature for low power and less memory consumption can be achieved for 32 operations. The parallel execution of instructions improves the effectiveness of the processor for the high speed of operation to achieve maximum throughput using fewer cycles for the instruction and to avoid glitches, and clock gating schemes, the RISC is implemented which has low power. The power consumption can be lowered in pipelining forwarding and stalling techniques and Tomasulo algorithms [1], [2]. The pipelining without interlocked stages is implemented in Harvard-type architecture to ensure no deadlocks occur between the stages of the pipelining and area, power, and delay are reduced [3], [4]. The QR decomposition core for the FPGA processor is developed for the 32-bit RISC processor. The design of a 16-bit RISC processor with a pipeline feature using von.

Neumann architecture supporting load/store instructions on FPGA Spartan 6, proved it has area utilization very less is kept forward in 2018. Digital modulation techniques [5] like binary amplitude-shift keying (BASK), binary frequency-shift keying (BFSK), binary phase-shift keying (BPSK), and quadrature phase shift keying (QPSK) are simulated successfully in the Xilinx environment with a 32-bit serial data transmission with self-adjustable carrier frequency and bit duration length.

The FPGA implementation of digital modulation techniques has a greater capacity to convey large amounts of information than analog modulation schemes. The amplitude modulation (AM) and quadrature amplitude modulation (QAM) are generated using coordinate rotation digital computer (CORDIC) [6] based design implemented in register-transfer-level (RTL), it is based on an analog carrier as well as baseband signal sampled at a particular rate using a quantizer then further applied to the digital modulation module to generate the modulated signal in the selected time frame for the specified data rate. CORDIC is used to solve the signal multiplication part in the specified data format of the signals. By the selection of merged clock gating techniques, the logic required also decreases [7]. The FPGA-implemented RISC processor utilizes the IEEE 754 standards fused with Vedic multiplication to minimize the delay by 55% than the conventional multiplication [8]. The micro-architecture is developed using the Verilog hardware description language (VHDL) code to design the 64-bit RISC V processor with optimized power and efficiency [9], [10].

The RISC processor is innovative in design to accommodate several applications in real-time. The fusion of the RISC processor with the advanced encryption standard (AES) cryptography algorithm is implemented in the FPGA to achieve high performance and reduction in delay [11], [12]. The FPGA-implemented RISC V processor is utilized for the comprehension of the design flow, hardware incorporation, communication protocols for computer engineers and fault analysis in single event transients [13]–[15]. The RISC processor design is incorporated with the internet of things (IoT) to accomplish the optimized CPU microarchitecture and object detection [16], [17]. The VHDL-based RISC processor architecture is developed to minimize the execution time of the static code analysis [18]. The RISC architecture is used to embed the sensing system to reduce the size when implemented in the FPGA device [19]–[21]. The RISC processor is analyzed for single event transient fault for the possibility of multiple bit upsets by implementing in the FPGA [22]–[24].

This paper introduces a novel RISC-communication processor (RCP) design that uses signal generations namely sine, cosine, square, analog, and digital modulation schemes like amplitude modulation (AM), amplitude shift keying (ASK), analog quadrature amplitude modulation (AQAM), digital quadrature amplitude modulation (DQAM), and real-time applications like a traffic light, and digital clock. Communication along with RISC architecture offers many advantages in real-time applications. The proposed RCP design is presented in the next section.

2. PROPOSED RISC COMMUNICATION PROCESSOR

The RCP is a joint architecture consisting of a RISC processor along with the communication architecture containing operations. RCP uses the pipeline feature in its operations to utilize the processor efficiently. The RCP proposed in this paper has 32 operations consisting of arithmetic and logical unit, shifting and rotating units, comparator units, a communication unit, and an application unit. Figure 1 depicts the block diagram of the proposed RCP architecture.

2.1. Arithmetic and logical unit (ALU)

ALU consists of basic arithmetic operations like addition, subtraction, increment, decrement and logical operations like AND, NAND, OR, NOR, XOR, XNOR, NOT which are taken with a resolution of 2^{16} bits. The 16 bits data is manipulated according to the type of operations by the assigned opcode. Based on the opcode values, the 16 bits for the two operands are manipulated to obtain the output. The instruction format for the ALU of the RCP is shown in Table 1.

2.2. Comparator unit (CU)

The comparator uses two 16-bit data and compares the two numbers with 4 cases. The cases are equal to, greater than, less than, and not equal to. The instruction format for CU is given in Table 2.

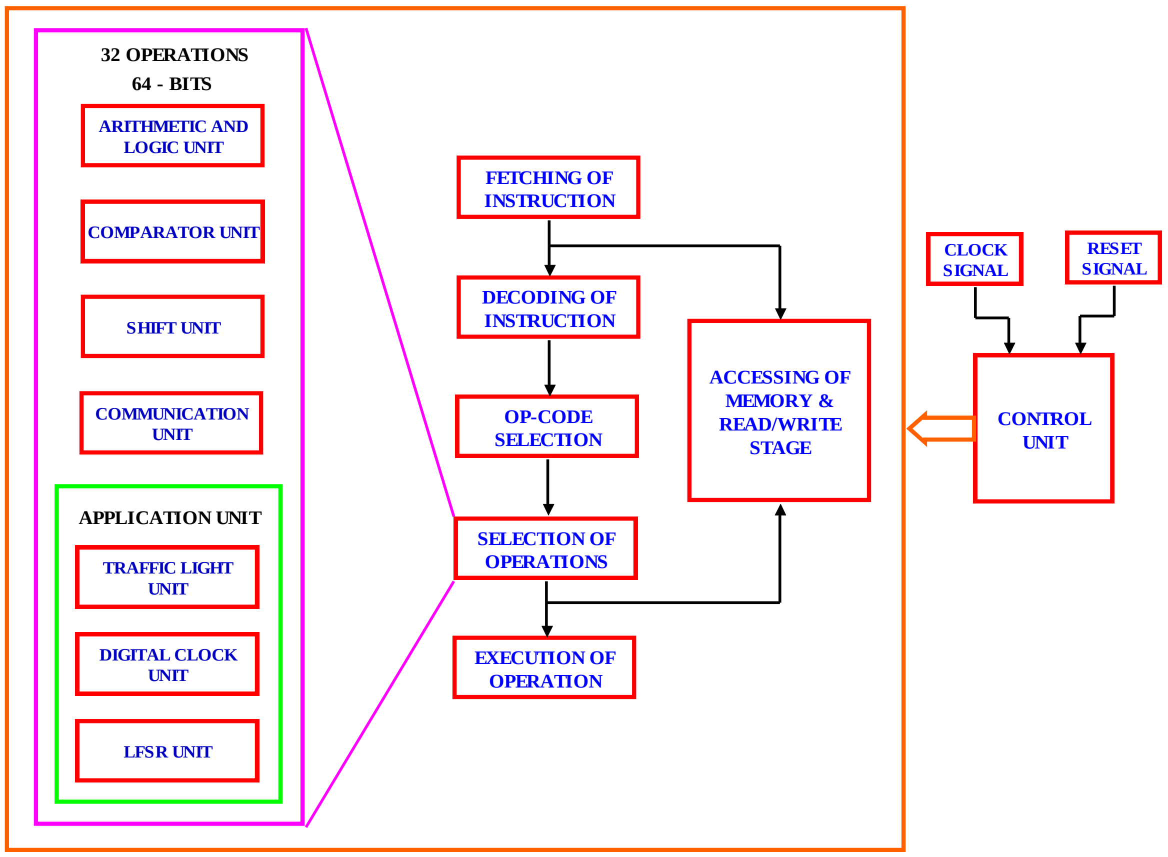

Figure 1. Block representation of proposed RISC-communication processor

Table 1. Instruction format for the arithmetic and logical unit of RCP design

Operations	RW	Bit format for inputs and outputs			Opcode
		Output	Second input	First input	
a+b	63	53-37 (17)	36-21 (16)	20-5 (16)	4-0 (00000)
a-b	63	53-37 (17)	36-21 (16)	20-5 (16)	4-0 (00001)
a-1	63	36-21 (16)	---	20-5 (16)	4-0 (00010)
a+1	63	36-21 (16)	---	20-5 (16)	4-0 (00011)
not a	63	36-21 (16)	---	20-5 (16)	4-0 (00100)
and	63	52-37 (16)	36-21 (16)	20-5 (16)	4-0 (00101)
nand	63	52-37 (16)	36-21 (16)	20-5 (16)	4-0 (00110)
or	63	52-37 (16)	36-21 (16)	20-5 (16)	4-0 (00111)
nor	63	52-37 (16)	36-21 (16)	20-5 (16)	4-0 (01000)
xor	63	52-37 (16)	36-21 (16)	20-5 (16)	4-0 (01001)
xnor	63	52-37 (16)	36-21 (16)	20-5 (16)	4-0 (01010)

Table 2. Instruction format for the comparator unit of RCP design

Operations	RW	Bit format for inputs and outputs			Opcode
		Output	Second input	First input	
a=b	63	37 (1)	36-21 (16)	20-5 (16)	4-0 (01011)
a<b	63	37 (1)	36-21 (16)	20-5 (16)	4-0 (01100)
a>b	63	37 (1)	36-21 (16)	20-5 (16)	4-0 (01101)
a! =b	63	37 (1)	36-21 (16)	20-5 (16)	4-0 (01110)

2.3. Shift unit (SU)

The SU unit is based on the number of bits shifted either to the left or right in the register. The shifting is based on the direction ‘d’ bit, if ‘d’ is 1 it indicates a right shift, and ‘d’ is 0 for a left shift. The shift left/shiftright (SL/SR) for 27-bit manipulation is implemented. The instruction format for this unit is given in Table 3.

Table 3. Instruction format for the shift unit of RCP design

Operations	RW	Bit format for inputs and outputs				Opcode
		Output	Bits change	Direction	Input	
1 bit-LS	63	38-23 (16)	22 (1)	21 (1)	20-5	4-0 (01111)
2 bit-LS	63	39-24 (16)	23-22 (2)	21 (1)	20-5	4-0 (10000)
3 bit-LS	63	40-25 (16)	24-22 (3)	21 (1)	20-5	4-0 (10001)
4 bit-LS	63	41-26 (16)	25-22 (4)	21 (1)	20-5	4-0 (10010)
5 bit-LS	63	42-27 (16)	26-22 (5)	21 (1)	20-5	4-0 (10011)
6 bit-LS	63	43-28 (16)	27-22 (6)	21 (1)	20-5	4-0 (10100)
7 bit-LS	63	44-29 (16)	28-22 (7)	21 (1)	20-5	4-0(10101)

2.4. Communication unit (CMU)

The communication unit is incorporated with the proposed RISC processor design in this work. Communication unit performs signal generation like sine, cosine, and square which are the main components in the various signal transmissions schemes like AM, ASK, AQAM, and DQAM. The instruction format regarding these operations is discussed here.

2.4.1. Sine wave generation

The sine wave is an important component in the signal modulation schemes for its transmission of information. It is generated using MATLAB Simulink with HDL-supported blocks. Then the Verilog code is generated using an HDL coder in the Simulink and the signal is tested in the integrated science instrument module (ISIM) environment. Table 4 shows the instruction format for the sine wave.

Table 4. Instruction format for the sine signal of communication unit of RCP design

Operation	RW	Bit format for inputs and outputs		Opcode
		Sine	Clock	
Sine	63	21-6 (16)	5 (1)	4-0 (10110)

2.4.2. Cosine wave generation

The cosine wave is generated in the same way as the sine with a 90^0 phase shift with the sine wave. It's a vital component in QAM analysis and carrier transmissions. The instruction format for the cosine wave is shown in Table 5.

Table 5. Instruction format for the cosine signal of communication unit of RCP design

Operation	RW	Bit format for inputs and outputs		Opcode
		Cosine	Clock	
Cosine	63	21-6 (16)	5 (1)	4-0 (10111)

2.4.3. Square wave generation

The square wave is a basic signal used in digital modulation techniques with a duty cycle varying with the techniques. The square wave generated based on the frequency which predicts the pulse width is given by (duty cycle). The instruction format for the square wave generation is shown in Table 6.

$$\text{Frequency(Hz)} = \frac{1}{\text{ON TIME+OFF TIME}}$$

Table 6. Instruction format for the square signal of communication unit of RCP design

Operation	RW	Bit format for inputs and outputs		Opcode
		Square	Clock	
Square	63	15-6 (10)	5 (1)	4-0 (11000)

2.4.4. Amplitude modulation (AM)

AM is of 235-bit resolution. The message and carrier signals are taken in the full-wave lookup tables with 16 bits to generate the AM signal. It is mainly generated based on the clock, reset and clock enable signals specified in the design. It is used in radio broadcasting. The instruction format for the AM is given in Table 7.

Table 7. Instruction format for the AM of communication unit of RCP design

Operation	RW	Bit format for inputs and outputs		Opcode
		AM signal	Clock	
AM	63	40-6 (35)	5 (1)	4-0 (11001)

2.4.5. Amplitude shift keying (ASK)

ASK is a form of AM that represents digital data as the variations of the amplitude of a carrier wave. The message and carrier signals are taken in the full-wave lookup table. In this binary symbol '1' represents the transmission of the carrier signal and '0' represents no signal transmitted. The clock, reset and clock enable signals specified in the designs used for the ASK generation. The instruction format for the ASK is given in Table 8.

Table 8. Instruction format for the ASK of the communication unit of RCP design

Operation	RW	Bit format for inputs and outputs		Opcode
		ASK signal	Clock	
ASK	63	21-6 (16)	5 (1)	4-0 (11010)

2.4.6. Analog quadrature amplitude modulation (AQAM)

AQAM is an analog modulation technique that uses two analog message signals by changing the amplitudes of two carriers that are out of phase with each other and thus generate quadrature. The message and carrier signals are taken in the full-wave lookup table. The in-phase and quadrature-phase components are present in the AQAM signal. The instruction format for the AQAM is given in Table 9.

Table 9. Instruction format for the AQAM of communication unit of RCP design

Operation	RW	Bit format for inputs and outputs		Opcode
		AQAM signal	Clock	
AQAM	63	37-6 (32)	5 (1)	4-0 (11011)

2.4.7. Digital quadrature amplitude modulation (DQAM)

DQAM is a digital modulation technique that uses two digital message signals by changing the amplitudes of two carriers using the ASK scheme. The carriers are out of phase with each other, producing quadrature. The message and carrier signals are taken in the full-wave lookup table. The in-phase and quadrature-phase components are present in the DQAM signal. The instruction format for the DQAM is given in Table 10.

Table 10. Instruction format for the DQAM of communication unit of RCP design

Operation	RW	Bit format for inputs and outputs		Opcode
		DQAM signal	Clock	
DQAM	63	37-6 (32)	5 (1)	4-0 (11100)

2.5. Application unit (AU)

The application unit consists of 3 operations a traffic light unit for the control of traffic signals, a digital clock unit for the display of the 24-hour clock, and a linear feedback shift register (LFSR) unit for the generation of random bits which are used in cryptography, spread spectrum modulation. These 3 operations are listed in next section.

2.5.1. Traffic light unit (TLU)

The traffic light unit generates the three signals in the controllers in 4 directions. It uses multiplexers in its design. It uses 3 bits to represent the output of the traffic light in one direction and hence a total of 12 signals are generated as outputs from the controllers representing red, yellow, and green. The instruction format for the traffic light unit is shown in Table 11.

Table 11. Instruction format for the traffic light unit of RCP design

Operation	RW	Bit format for inputs and outputs					Opcode
		North	South	East	West	Clock	
Traffic light	63	17-15 (3)	14-12 (3)	11-9 (3)	8-6 (3)	5 (1)	4-0 (11101)

2.5.2. Digital clock unit (DCU)

The digital clock unit generates the time for 24 hours based on the hours, minutes, and seconds. It utilizes 2 variables for an hour, minutes, and seconds to store the value in it. After exceeding the value in variable 1 it increments the value stored in variable 2 by one value and the process is repeated for 24 hours. A seven-segment display is considered, and the VHDL code is written for the digital clock based on seven-segment displays. The instruction format based on the digital clock is shown in Table 12.

Table 12. Instruction format for the digital clock of RCP design

Operation	RW	Bit format for inputs and outputs							Opcode
		H1	H2	M1	M2	S1	S2	Clock	
DCU	63	47-41 (7)	40-34 (7)	33-22 (7)	26-20 (7)	19-13 (7)	12-6 (7)	5 (1)	4-0 (11110)

2.5.3. Linear-feedback shift register unit (LFSR)

LFSR is a shift register whose input bit is a linear function of its previous state. The linear function used is XOR. LFSR has a well-defined feedback function that can generate the sequence of bits for a long duration of cycles. This feature can be used in digital counters and pseudo-random numbers. The instruction format for LFSR is given in Table 13.

Table 13. Instruction format for the LFSR of RCP design

Operation	RW	Bit format for inputs and outputs		Opcode
		LFSR	Clock	
LFSR	63	36-20 (16)	20-5 (16)	4-0 (11111)

3. INSTRUCTION PIPELINING IN RCP

All these 32 RISC-communication-based operations are executed using the pipelining feature in VHDL. The instructions to access the operations are stored in 64 bits and the pipeline feature is initiated. The pipeline decreases the number of clock cycles needed to execute the operations. The pipeline is executed in the sequence of five operations.

3.1. Instruction fetch (IF)

This is the first stage in the pipeline process, it accesses the 64-bit instruction from the memory on a program counter (PC) register and it knows the other instructions by incrementing 1 to the present instruction to get the next instruction. The VHDL code is developed taking IF into the consideration for 32 operations of RCP.

3.2. Instruction decode (ID)

This stage involves bit splitting process for the 64-bit instruction fetched by the IF stage. The bit splitting is performed for all 32 operations of the RCP design. The main fields in the bit splitting are opcode, inputs, and outputs, read/write control. The opcode field is 4-0 bits for all the cases, R/W control is given to the Most Significant Bit (MSB) bit (64'bit). The value in R/W specifies the type of operation; if it is '0' it is used for the reading operation and '1' for the write operation. The remaining fields like inputs and outputs are operation based.

3.3. Execute (EX)

This stage deals with the output evaluations based on the inputs and type of opcode selected in the RCP design. This is the third stage of pipelining. When any operation is in this stage the preceding operation is in the next stage and the previous operation is in IF, ID stages.

3.4. Read/Write (RW)

This is a control signal in the instruction format when fetched; it specifies whether to read from memory or write the outputs to the memory. The RW uses only 1 bit in the opcode for its operation as indicated in all the instruction. The MSB is utilized to indicate the operation of read or write namely 63 bit is considered.

3.5. Memory access (MA)

It is the last stage in the pipeline process; it deals with the memory access of instructions based on the RW signal. The pipelining is adjusted to the RCP design by the control unit; it specifies when the operation needs to enter the stages and when to leave. The VHDL code is developed to adjust the pipelining feature.

4. RESULTS AND DISCUSSION

The instruction pipelining of RCP design is fed into the ISIM software which executes all the operations based on the pipeline feature. The pipelining is developed in VHDL, and it is shown for all the 32 operations described above in Figures 2-5. Figure 2 describes the simulation of the ALU and CU-based operations in the pipeline fashion and the result is stored in the register. Figure 3 shows the simulation of the SU, the left/right shifting is done based on the variable ‘d’ specified in the instruction format. Figure 4 shows the modulation signals of the communication unit. Figure 5 shows the outputs of the application unit of RCP design, it shows the simulation response of traffic light, digital clock, and LFSR units developed using VHDL code in the Xilinx ISE 14.5. Table 14 describes the device utilization of the RCP design using Xilinx Virtex 7. The proposed RCP is compared with [20] and proves that the Virtex 7 FPGA is advantageous in comparison with the other FPGA implementations. Table 15 shows the comparison of different FPGAs proposed for the RCP design.

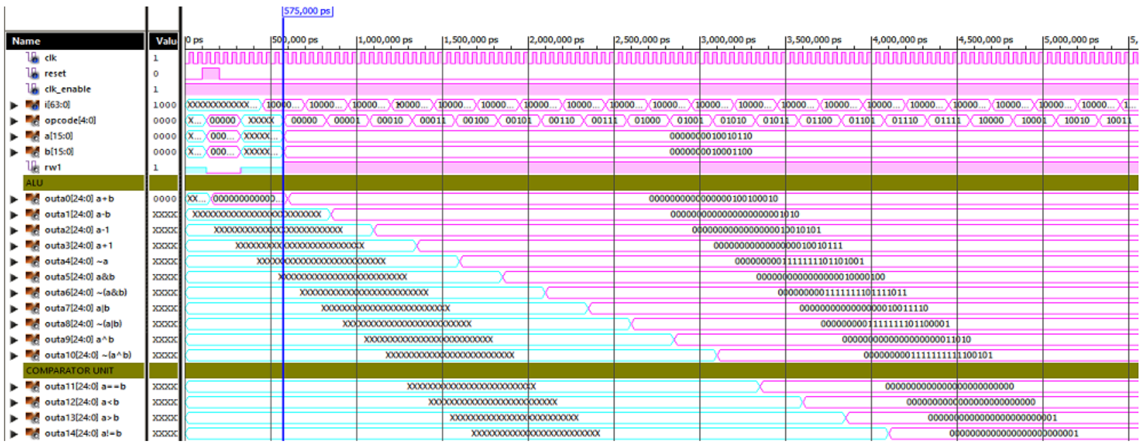

Figure 2. Simulation result for ALU and CU of RCP design using ISIM

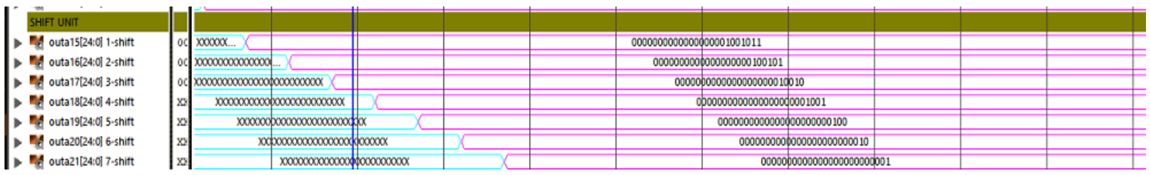

Figure 3. Simulation result for SU of RCP design using ISIM

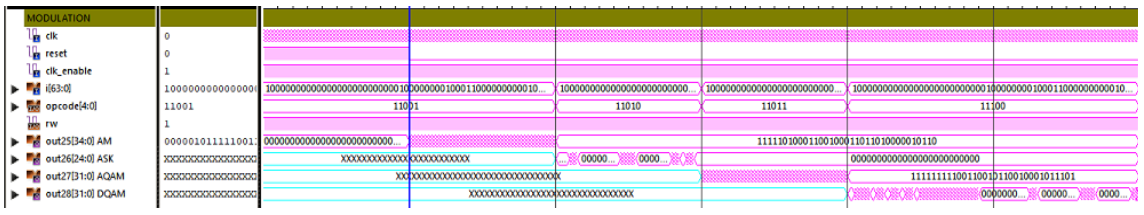

Figure 4. Simulation result for modulation of RCP design using ISIM

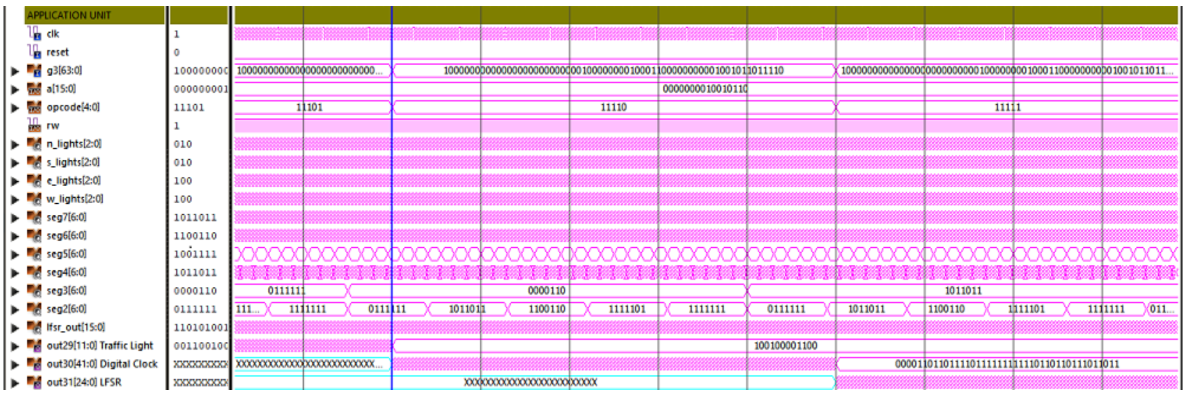

Figure 5. Simulation result for application unit of RCP design using ISIM

Table 14. Device utilization of Virtex 7 of the RCP design

Logic utilization	Used	Available	Utilization
Number of slice registers	264	2,443,200	0%
Number of slices LUTs	761	1,221,600	0%
Number of fully used LUT-FF pairs	167	858	19%
Number of bonded IOBs	908	1,200	75%
Number of BUFG/BUFGCTRLs	5	128	3%
Number of DSP48E1s	5	2,160	0%

Table 15. Comparison of device utilization of various FPGA of the RCP design

Device utilization of various FPGA standards				
Parameters	Methods			
	Virtex 7	Spartan 3A DSP	Spartan 3E	[25]
Number of slice registers	264	821	842	586
Number of slices LUTs	761	313	306	761
Number of LUT FF PAIRS	167	1,493	1,531	826
Number of bonded IOBs	908	908	908	223
Number of BUFG/BUFGCTRLs	5	5	4	1
Number of DSP48E1s	5	5	4	2

5. CONCLUSION

A RCP was implemented in real-time using Xilinx Virtex 7 FPGA. In comparison, the device utilization chart of the proposed method was found to be feasible. Also, the proposed RCP architecture can be used for the generation of many communication signals and applications by selecting the corresponding opcode as listed in the instruction format within the FPGA device. The timing analysis is performed very well according to the specifications and the area consumed is also low. Future work can be extended towards more complex operations with RISC/CISC architectures.

REFERENCES

[1] C. Venkatesan, M. T. Sulthana, G. M. Sumithra, and M. Suriya, "Design of a 16-Bit Harvard Structure RISC Processor in Cadence 45nm Technology," in *2019 5th International Conference on Advanced Computing and Communication Systems, ICACCS 2019*, Mar. 2019, pp. 173–178, doi: 10.1109/ICACCS.2019.8728479.

[2] G. Scotti and D. Zoni, "A fresh view on the microarchitectural design of FPGA-based RISC CPUs in the IoT Era," *Journal of Low Power Electronics and Applications*, vol. 9, no. 1, Feb. 2019, doi: 10.3390/jlpea9010009.

[3] S. S. Omran and A. K. A. Abbas, "Design of 32-bits RISC processor for hardware efficient QR decomposition," in *International Conference on Advances in Sustainable Engineering and Applications, ICASEA 2018 - Proceedings*, Mar. 2018, pp. 69–73, doi: 10.1109/ICASEA.2018.8370958.

[4] P. Jamieson *et al.*, "Computer engineering education experiences with RISC-V architectures-from computer architecture to microcontrollers," *Journal of Low Power Electronics and Applications*, vol. 12, no. 3, Aug. 2022, doi: 10.3390/jlpea12030045.

[5] S. M. Bhagat and S. U. Bhandari, "Design and analysis of 16-bit RISC processor," in *Proceedings - 2018 4th International Conference on Computing, Communication Control and Automation, ICCUBEA 2018*, Aug. 2018, pp. 1–4, doi: 10.1109/ICCUBEA.2018.8697859.

[6] S. Kaur, V. Varshitha, J. Siddharth, V. S. Tejus, and H. P. Kumar, "Adaptive traffic light controller using FPGA," in *2018 3rd IEEE International Conference on Recent Trends in Electronics, Information and Communication Technology, RTEICT 2018 - Proceedings*, May 2018, pp. 1612–1617, doi: 10.1109/RTEICT42901.2018.9012263.

[7] M. H. Vo, "The merged clock gating architecture for low power digital clock application on FPGA," in *International Conference on Advanced Technologies for Communications*, Oct. 2018, pp. 282–286, doi: 10.1109/ATC.2018.8587596.

[8] S. Sushma, S. K. Ravindran, P. R. Nadagoudar, and P. A. Sophy, "Implementation of a 32 – bit RISC processor with floating point unit in FPGA platform," *Journal of Physics: Conference Series*, vol. 1716, no. 1, Dec. 2021, doi: 10.1088/1742-6596/1716/1/012047.

[9] R. Deepika, G. S. M. Priyadharsini, M. M. Malar, and V. I. Anand, "Microarchitecture based RISC-V instruction set architecture for low power application," *Journal of Pharmaceutical Negative Results*, vol. 13, no. 6, pp. 362–371, 2022, doi: 10.47750/pnr.2022.13.S06.051.

[10] W. Wu, D. Su, B. Yuan, and Y. Li, "Intelligent security monitoring system based on risc-v soc," *Electronics (Switzerland)*, vol. 10, no. 11, Jun. 2021, doi: 10.3390/electronics10111366.

[11] T. Gomes, P. Sousa, M. Silva, M. Ekpanyapong, and S. Pinto, "FAC-V: An FPGA-based AES coprocessor for RISC-V," *Journal of Low Power Electronics and Applications*, vol. 12, no. 4, Sep. 2022, doi: 10.3390/jlpea12040050.

[12] B. Özkilbaç and T. Karacali, "Design of 32-bit RISC processor with IEEE754 standard floating-point unit in FPGA for digital signal processing applications," *Erzincan Üniversitesi Fen Bilimleri Enstitüsü Dergisi*, vol. 15, no. 3, pp. 699–714, Dec. 2022, doi: 10.18185/erzifbed.1077921.

[13] F. Ferlini, F. Viel, L. O. Seman, H. Pettenghi, E. A. Bezerra, and V. R. Q. Leithardt, "A methodology for accelerating FPGA fault injection campaign using ICAP," *Electronics (Switzerland)*, vol. 12, no. 4, Feb. 2023, doi: 10.3390/electronics12040807.

[14] S. Kalapothas, M. Galetakis, G. Flamis, F. Plessas, and P. Kitsos, "A survey on RISC-V-based machine learning ecosystem," *Information*, vol. 14, no. 2, Jan. 2023, doi: 10.3390/info14020064.

[15] H. Zhang *et al.*, "A heterogeneous RISC-V processor for efficient DNN application in smart sensing system," *Sensors (Basel, Switzerland)*, vol. 21, no. 19, Sep. 2021, doi: 10.3390/s21196491.

[16] D. Wu, Y. Liu, and C. Tao, "A universal accelerated coprocessor for object detection based on RISC-V," *Electronics*, vol. 12, no. 3, Jan. 2023, doi: 10.3390/electronics12030475.

[17] J. Y. Lai, C. A. Chen, S. L. Chen, and C. Y. Su, "Implement 32-bit RISC-V architecture processor using verilog HDL," in *ISPACS 2021 - International Symposium on Intelligent Signal Processing and Communication Systems: 5G Dream to Reality, Proceeding*, Nov. 2021, pp. 1–2, doi: 10.1109/ISPACS51563.2021.9651130.

[18] I. G. D. Río *et al.*, "A risc-v processor design for transparent tracing," *Electronics (Switzerland)*, vol. 9, no. 11, pp. 1–23, Nov. 2020, doi: 10.3390/electronics9111873.

[19] G. Soulard, G. P. Lachance, E. Boisselier, M. Boukadoum, and A. Miled, "RISC-V based processor architecture for an embedded visible light spectrophotometer," in *Canadian Conference on Electrical and Computer Engineering*, Sep. 2022, pp. 360–363, doi: 10.1109/CCECE49351.2022.9918259.

[20] Y. Hu *et al.*, "Specially-designed out-of-order processor architecture for microcontrollers," *Electronics (Switzerland)*, vol. 11, no. 19, Sep. 2022, doi: 10.3390/electronics11192989.

[21] A. Coluccio, A. Ieva, F. Riente, M. R. Roch, M. Ottavi, and M. Vacca, "RISC-Vlim, a RISC-V framework for logic-in-memory architectures," *Electronics (Switzerland)*, vol. 11, no. 19, Sep. 2022, doi: 10.3390/electronics11192990.

[22] J. Sharma and N. Rao, "The characterization of errors in an FPGA-based RISC-V processor due to single event transients," *Microelectronics Journal*, vol. 123, May 2022, doi: 10.1016/j.mejo.2022.105392.

[23] J. Plusquellic, D. E. Owen, T. J. Mannos, and B. Dziki, "Information leakage analysis using a co-design-based fault injection technique on a RISC-V microprocessor," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 3, pp. 438–451, Mar. 2022, doi: 10.1109/TCAD.2021.3065915.

[24] J. Li, S. Zhang, and C. Bao, "Duckcore: A fault-tolerant processor core architecture based on the risc-v isa," *Electronics (Switzerland)*, vol. 11, no. 1, Dec. 2022, doi: 10.3390/electronics11010122.

[25] J. A. Prathap, T. S. Anandhi, N. Malladhi, and V. Roja, "Investigation of FPGA based 32-bit RISC-modulation processor," *International Journal of Computer Information Systems and Industrial Management Applications*, vol. 10, pp. 227–237, 2018.

BIOGRAPHIES OF AUTHORS

Joseph Anthony Prathap was born in 1981 in Puducherry. He obtained B.E [Electronics and Communication] and M. Tech [VLSI Design] degrees in 2003 and 2007 respectively, and a Ph.D. in FPGA-based Power Converters in 2017 from Annamalai University. He has put in 15 years of service in teaching and research. He is currently an Associate Professor in the Department of Electronics and Communication Engineering at Presidency University, Bengaluru, Karnataka, India. His research interest includes VLSI design, the development of digital switch patterns, FPGA control techniques for power converters, photovoltaic power electronics converters. He can be contacted at email: joseph.anthony@presidencyuniversity.in.

Sai Ramesh was born in 1996 in Telangana. He obtained B. Tech [Electronics and Communication] from Guru Nanak Institutions and M. Tech [DECS] from Vardhaman College of Engineering in 2018 and 2020 respectively. He has put in 2 years of service in research and development. He is currently an intern at Puredata Solution LLP, Hyderabad, Telangana, India. His research interest includes VLSI design, the development of Processor, FPGA based real-time applications. He can be contacted at email: sairameshati@gmail.com.