International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 06 | June 2020 www.irjet.net p-ISSN: 2395-0072

Design of Low Power High Speed 8-Bit RISC Processor

Aishwarya H S[1], Sujatha Hiremath[2]

[1]Department of Electronics and Communication Engineering, RV College of Engineering, Bengaluru -59
[2]Assistant Professor, Department of Electronics and Communication Engineering, RV College of Engineering, Bengaluru – 59

Abstract - *The proposed paper develops and implements an 8-bit Reduced Instruction Set Computer (RISC). The processor can execute with simplified design for more number of instructions and obtained less critical path delay. The proposed processor will be implemented and synthesized using Xilinx. The proposed processor consists of the different blocks mainly IO Buffer, accumulator, clock generator, decoder, instruction register, ALU, memory, program counter and multiplexer. The functional verification takes place in Xilinx tool suit. RISC processors have a very large range of applications based on power consumption and speed.*

Key Words: **Reduced Instruction Set Computer, IO Buffer, ALU, Accumulator, Program Counter, Instruction Register**

1. INTRODUCTION

The computer uses less space and simple instruction sets, mainly to reduce the equipment used to implement the components. As the processor technology began to developed, it trend to have a large number of instructions capacity to the processor. The instruction register became very complex to design and the control unit occupies most of the chip area and the instructions that access memory results in decreasing the speed execution. Thus RISC processor is developed and a small set of instructions results in simplifies the design and improves the overall performance of the processor. These are cost-effective and compatible systems. The concept of RISC architecture reduces the execution time of the processor. The RISC processor will have comparatively less instructions, less addressing modes, the memory access is confined to load instructions and store instructions. All operations take place within processor register. The decoded instruction format shall be of fixed length and instruction shall be executed in one cycle. Memory access is only done during the execution stage, utilizing load-store instructions. All operations except store and load are stored in the processor to stored with.

2. The ARCHITECTURE OF 8-BIT RISC PROCESSOR

The 8-bit RISC processor architecture is shown in Fig-1. The building is made of the ALU, IO Buffer, accumulator, clock generator, decoder, instruction register, memory, program counter, and multiplexer.

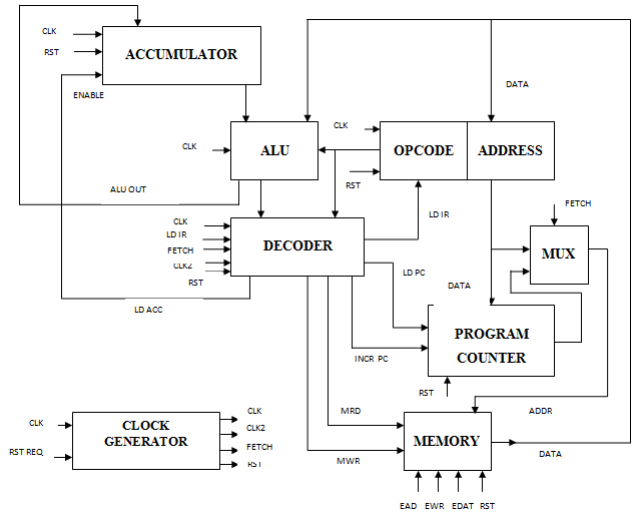

Fig-1: Architecture of 8-bit RISC processor

The RISC processor consists of a memory, accumulator, and ALU. The operands of ALU are drawn from the accumulator and memory. The ALU output is retained in the accumulator and/or memory. The result of the ALU is saved in the accumulator or the memory. The accumulator provides the address of the next instruction in the memory. Instructions are obtained from the memory and saved in the instruction register. The decoder provides the correct sequence of information, which includes the timing information and generates the control signal. The processor contains an eight bit bi-directional data bus and five bit single directional address bus to address 32 locations of the address. The processor instructions set is 8 bit by the data bus and 3 bit contains Opcode and 5-bit has the field of the address and specified instructions are shown in Table-1.

Opcode	Operation
000	Halt
001	Skip zeros
010	ADD
011	AND
100	XOR
101	LOAD
110	STORE
111	JUMP

Table-1: Operation based on opcode

The decoder contains an Opcode which decodes the control signals. The decoded instruction based on the opcode performs the operations. Opcode includes word length instructions, and timing intervals are the same for all instructions. Even the fetch cycle is the same for all the instructions. For instruction to instruction execution cycle changes, but the execution time is the same. The process of retrieval for all instructions will be same. This is basic characteristic of the RISC processors.

The RISC processor design involves design of the each block and integrate all in the single module. The processor modules are ALU, memory, clock generator, multiplexer, instruction register, accumulator, program counter, decoder and IO buffer. The Constraints of RISC Processor is shown in Table-2.

CYCLE	INC_PC	LOAD_ACC	LOAD_PC	MEM_WR	MEM_RD	LOAD_IR
ADDRESS SETUP	0	0	0	0	0	0
INSTRUCTION FETCH	0	0	0	0	1	0
INSTRUCTION LOAD	0	0	0	0	1	1
IDLE	0	0	0	0	1	1
ADDRESS SETUP	1	0	0	0	0	0
OPERATION FETCH						
ADD,AND,XOR,LOAD	0	0	0	0	1	0
ALL OTHERS	0	0	0	0	0	0
ALU OPERATION						
SKIP AND ZERO	1	0	0	0	0	0
ADD,AND,XOR,LOAD	0	1	0	0	1	0
JUMP	0	0	1	0	0	0
ALL OTHERS	0	0	0	0	0	0
STORE RESULT						
JUMP	1	0	1	0	0	0
STORE	0	0	0	1	0	0
SKIP AND ZERO	1	0	0	0	0	0
ADD,AND,XOR,LOAD	0	1	0	0	1	0
ALL OTHERS	0	0	0	0	0	0

Table-2: Constraints of RISC Processor

3. FUNCTIONALITY OF THE MODELS IN PROCESSOR

The functionality of each model is explained below

3.1 Clock Generator

The clock generator contains clk, rstreq as the input, and generated signal as output. It generates clk1, clk2, and fetch signal. Clk2 is generated during the negative edge of the clk. The fetch signal is generated for every positive edge of the clk2. The generated signals are input to the decoder. These signals are the control signal to the processor. When fetch is zero, rst is high during the falling edge of clk2. The structure of Clock Generator is shown in Fig-2.

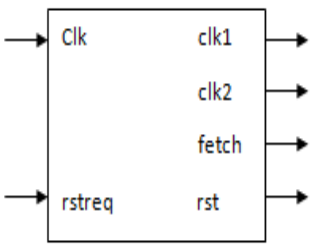

Fig-2: Structure of Clock Generator

3.2. Instruction Register

Instruction register is the part of the processor which hold the currently executed instruction. The IR contains mdat, clk, ldir, rst as the input and opcod and adir as the output. During positive edge of clock, IR performs the operation. The instruction register gets separated into upper 3-bit as an opcode and lower 5-bit as an address when the LDIR and reset both are high. The Illustration and flow chart of Instruction Register are shown in Fig-3 and Fig-4.

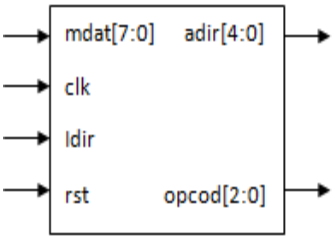

Fig-3: Illustration of Instruction Register

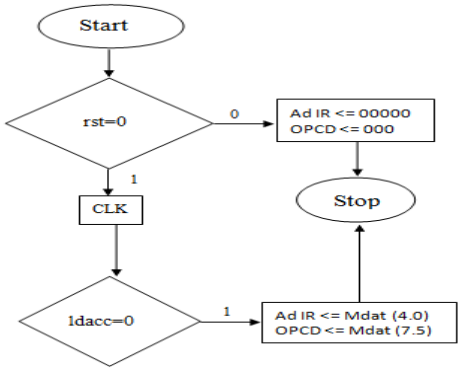

Fig-4: Flow chart of Instruction Register

3.3 Accumulator

The accumulator is a registry type. The output of the ALU is retained in the accumulator. It contains clk, idac, rst, alout as input to the accumulator, and acout as the output. During the positive edge of the clock, the accumulator gets activated. When both ldac and rst are high, data are stored in the acout in the accumulator. The Block diagram and flow chart of the accumulator are shown in Fig-5 and Fig-6.

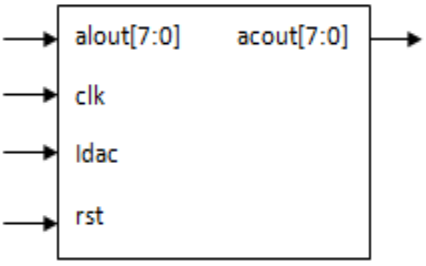

Fig-5: Block diagram of Accumulator

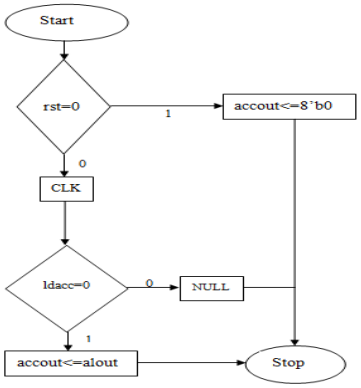

Fig-6: Flow chart of Accumulator

3.4 Arithmetic Logic Unit

ALU performs both logic and arithmetic functions based on input. During the negative edge of the clock, ALU gets synchronized with the input of the ALU. ALU contains acout, mdat, opcod as input and alout, and zr as output. For every negative edge, data is stored in both zr and alout. The Block diagram and flow chart of ALU is shown in Fig-7 and Fig-8 respectively.

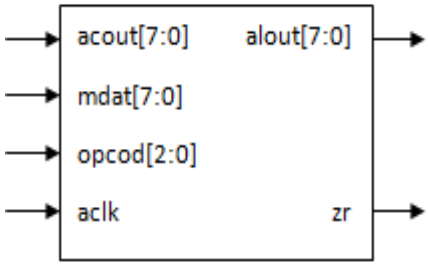

Fig-7: Block diagram of ALU

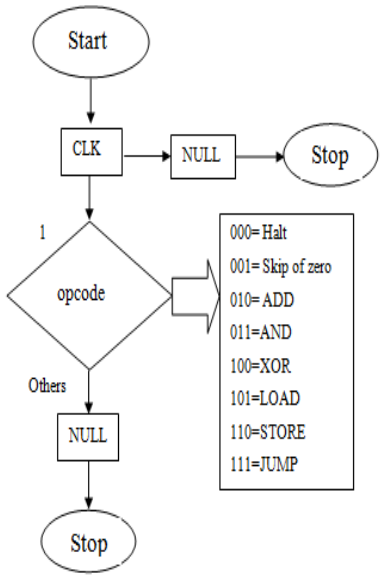

Fig-8: Flow chart of ALU

3.5 Memory

In the RISC processor, the length of the memory is 8-bit wide and 32-bit location. The instruction is received from the memory which contains opcode and address. The upper 3-bit sets the opcode and the lower 5-bit sets the address. This block contains memory read, memory write, and reset as the input and data as the output of the memory. Memory reads data from memory to the data register when the mrd is high. If mwr is high, the data will be written to the memory. The block diagram and flow chart of memory are shown in Fig-9 and Fig-10.

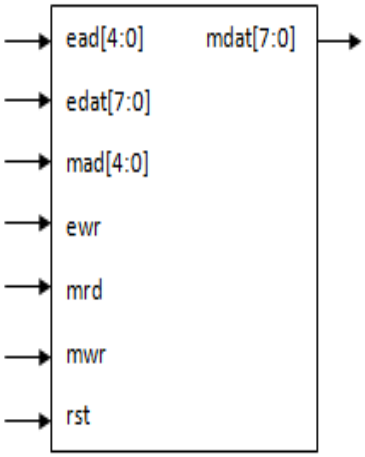

Fig-9: Block diagram of Memory

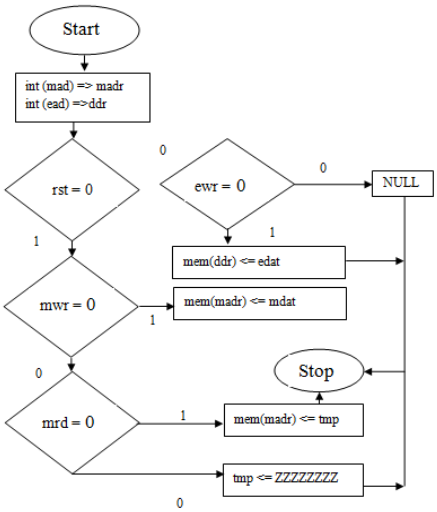

Fig-10: Flow chart of Memory

3.6 Multiplexer

The multiplexer selects the input based on the control signal. The location of the program counter is passed to the bus and instruction is retrieved when the signal is high. When the signal fetch is low, the IR location is transferred to the bus. The Block diagram and flow chart of multiplexer is shown in Fig-11 and Fig-12 respectively.

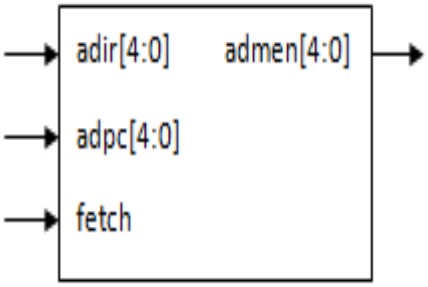

Fig-11: Block diagram of Multiplexer

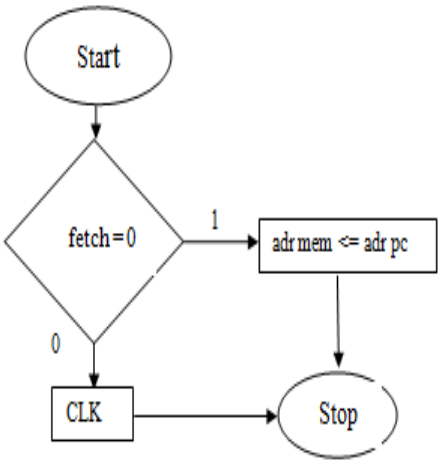

Fig-12: Flow chart of Multiplexer

3.7 Program Counter

Program counter includes the instruction address, which is executed. Program counter increases by 1, for each instruction, get it. PC is a general-purpose register of length 5. The program counter tries to use the next information of the memory to get the information. The program counter tends to the next instruction from memory to fetch the information. After fetch signal has been completed it refers to the next instruction. The Block diagram and flow chart of the program counter are shown in Fig-13 and Fig-14.

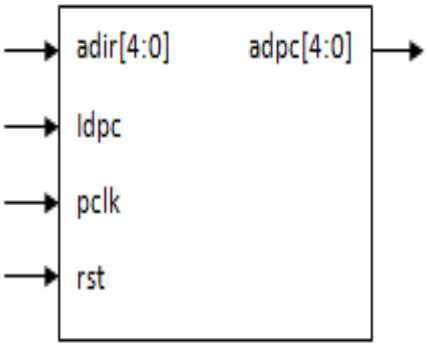

Fig-13: Block diagram of Program Counter

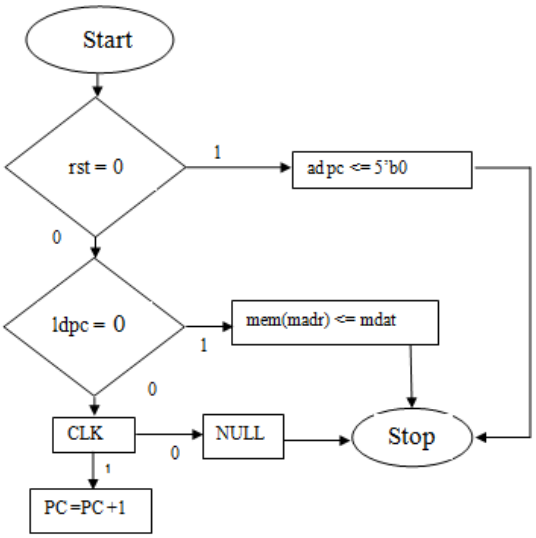

Fig-14: Flow chart of Program Counter

3.8 Decoder

A decoder has clk1, clk2, fetch, rst, and 3-bit opcode as input to the decoder. The output of the decoder are aclk, ldac, ldir, ldpc, mrd, mwr and pclk. The decoder gets the timing and control signal to decodes the signal. An opcode is received from the IR and decodes the information based on that opcode then it generates the control signal from the address bus. The contents of the address bus transfer to the program counter and then set the new address bus in the first stage. The fetch instruction gets activated and instructions are read from memory, then transfer the read signal to the data bus. When the instruction is loaded ldir becomes high and sent to the instruction register in the third stage. The Block diagram of the decoder is shown in Fig-15.

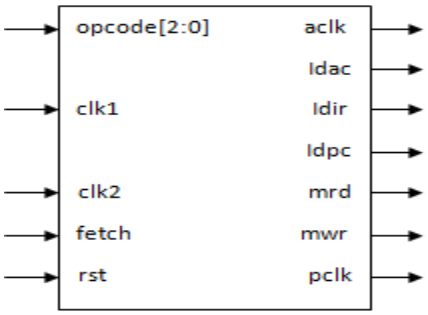

Fig-15: Block diagram of Decoder

During execution, based on control and timing signal opcode will be sent to the decoder. After receiving the signal, program counter starts incrementing the increment signal and sets the address in stage 5. The address field in the instruction register will be transferred to the address bus. In the sixth state, the opcode was retrieved, and mrd became high, and data was moved to the data bus, and then to the ALU. During state seven, ALU sends to the clock of ALU and is

synchronized with the negative edge of the clock, the particular operation being performed. The decoder sends load signal to save the output inside the accumulator.

3.9 IO Buffer

An IO buffer is a mechanism for temporarily holding data until it is ready to be used. The buffer serves as a queue for holding data until the processor is able to process the piece of next data for the input. The output buffer allows the processor to write the data rapidly and the system can read it. The flow chart of the IO Buffer is shown in Fig-16.

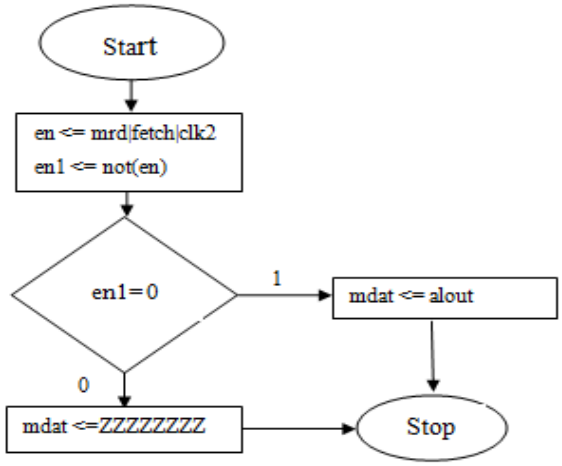

Fig-6: Block diagram of IO Buffer

4. RESULTS AND DISCUSSION

The RISC processor performs all the operations based on the opcode and the input. Based on the opcode it selects the module for different operations and performs. The implementation and synthesis are done using the Xilinx tools suit. The synthesis report is generated which includes the number of LUTs, flip-flops, the register used and the power report, timing constraints, and the timing report is generated. The final output of the processor is shown in Fig-17. The timing summary is generated in Xilinx and shown in Fig-18 and the timing summary is generated in Xilinx and shown in Fig-19. Table-3 shows the overall summary of the RISC processor.

Fig-17: Output of Processor

```
Timing Summary:
---------------
Speed Grade: -3

   Minimum period: 4.983ns (Maximum Frequency: 200.690MHz)
   Minimum input arrival time before clock: 3.119ns
   Maximum output required time after clock: 5.530ns
   Maximum combinational path delay: No path found

Timing Details:
---------------
All values displayed in nanoseconds (ns)

=========================================================================
Timing constraint: Default period analysis for Clock 'clkgen1/c2'
  Clock period: 4.263ns (frequency: 234.581MHz)
  Total number of paths / destination ports: 3 / 2
-------------------------------------------------------------------------
Delay:               2.131ns (Levels of Logic = 1)
  Source:            clkgen1/c3 (FF)
  Destination:       clkgen1/rst (FF)
  Source Clock:      clkgen1/c2 rising
  Destination Clock: clkgen1/c2 falling
```

Fig-18: Timing Summary

```
                                On-Chip Power Summary
-----------------------------------------------------------------------------
On-Chip                | Power (mW) |  Used  | Available | Utiliz
-----------------------------------------------------------------------------
Clocks                 |       0.45 |     38 |       --- |
Logic                  |       0.00 |    185 |      5720 |
Signals                |       0.00 |    438 |       --- |
IOs                    |       0.00 |     17 |       102 |
Static Power           |      13.69 |        |           |
Total                  |      14.14 |        |           |
-----------------------------------------------------------------------------

2.  Thermal Summary
-------------------------------
     Thermal Summary       |
-------------------------------
Effective TJA (C/W)  | 38.4 |
Max Ambient (C)      | 84.5 |
Junction Temp (C)    | 25.5 |
-------------------------------

3.  Power Supply Summary
-----------------------------------------------------------
                Power Supply Summary                      |
-----------------------------------------------------------
                     | Total | Dynamic | Static Power |
-----------------------------------------------------------
Supply Power (mW)    | 14.14 | 0.45    | 13.69        |
```

Fig-19: Power Summary

Parameter	values
Number of Register	296
Number of LUT	185
Number of IOs	17
Delay	2.131ns
Total Power	14.14mW
Minimum Period	4.983ns

Table-3: Summary of the Processor

5. CONCLUSION

The RISC processor is implemented and synthesis in Xilinx tool suits and functional verification is verified. The total real-time to the last completion is 40secs and the total CPU time to the last completion in 39.92secs. The minimum period and maximum frequency of the processor are 4.983ns and 200.69MHz. The delay of the processor is 2.131ns. The static power consumption is 13.69mW, the dynamic power

consumption is 0.45mW thus the total power consumption is 14.14mW.

6. FUTURE WORK

The further processor can add some more functionality and modules. Similarly 16-bit, a 32-bit processor can be implemented in a new version of Xilinx. ALU can contain some other arithmetic and logical function for an efficient processor.

REFERENCES

[1] Nehru, K., Shanmugam, A. and Darmila Thenmozhi, G.(2012), "Design of Low Power ALU Using 8T FA and PTL Based MUX Circuits", IEEE-International Conference on Advances in Engineering, Science and Management, pp. 145-149

[2] Pravin S. Mane, Indra Gupta, M. K. Vasantha, Implementation of RISC Processor on FPGA, 1-4244-0726-5/06, 2006 IEEE.

[3] R. Uma."Design and Performance Analysis of 8-bit RISC Processor using Xilinx Tool,"International Journal of Engineering

[4] Research and Applications(IJERA) Vol2,Issue 2,Mar-Apr 2012.

[5] Xiaoping Huang,Xiaoya Fan, Shengbing Zhang, 2008,"Design and Performance Analysis of One 32-bit Dual Issue RISC Processor for Embedded Application".

[6] Neenu Joseph, Sabarinath.S, Sankarapandiammala K, "FPGA based Implementation of High Performance Architectural level Low Power 32- bit RISC Core" International Conference on Advances in Recent Technologies

[7] Seung PyoJung, Jingzhe Xu, Donghoon Lee, Ju Sung Park, 2008, "Design And Verification Of 16 Bit RISC Processor", International SOC Design Conference.

[8] Rohit Sharma, Vivek Kumar Sehgal, Nitin Nitin1, Pranav Bhasker, Ishita Verma, 2009, "Design And Implementation Of 64-Bit RISC Processor Using VHDL", UKSim : 11th International Conference on Computer Modeling And Simulation, pp. 568 – 573.

[9] Samiappa Sakthikumaran, S.Salivahanan and V.S.Kaanchana Bhaaskaran, June 2011, "16-Bit RISC Processor Design For Convolution Application",IEEE International Conference on Recent Trends In Information Technology, pp.394-397.

[10] Sivarama P.Dandamudi,"A Guide to RISC Processor For Programmers And Engineers", Springer.