FPGA Based Implementation of 16 Bit RISC Controller

Patel Nilam S. , Prof. J.H.Patil
E and TC Department,
D. N. Patel College of Engineering, Shahada, Nandurbar, Maharashtra, India.

Abstract - This paper describes the design and implementation of some of the internal hardware components of a microcontroller. The subsystems designed are the fundamental hardware components necessary to create the 8-bit timer's input capture and output compare modes, the index register, and other systems. The subsystems designed were the clock controller, the timer, the register controller, the register, and the comparator. The project engineers implemented their system using Xilinx to test their logic and VLSI to construct the gates. VHDL code was written in order to implement the project onto an FPGA board.

Keyword : 16 bit microcontroller, RISC, VHDL,FPGA,Xilinx.

I.INTRODUCTION

Figure 1 shows the block diagram of a basic computer system. A basic computer system must have the standard elements CPU, memory and I/O. All these elements communicate via the system bus, which is composed by the data, address and control buses.

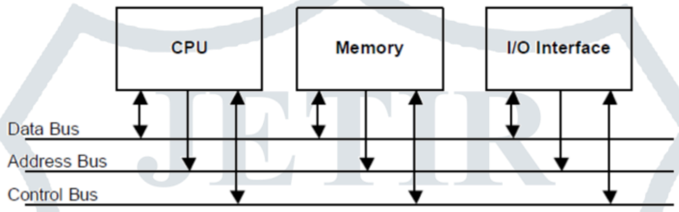


Figure 1 Basic Computer System

The CPU, as the 'brain' of the computer, administers all the activity in the system and performs all operations on data. The CPU has the ability to understand and execute instructions based on a set of binary codes, each representing a simple operation. These instructions are usually arithmetic, logic, data movement, or branch operations, and are represented by a set of binary codes called the instruction set. The memory, is used to store all the programs formed by the instruction set and all the require data. I/O interface provide an interconnection with the outside world, such as the keyboard as an input and the monitor as an output.

Minicomputers and mainframe computers, have CPUs consisting multiple ICs, ranging from several ICs (minicomputers) to several circuit boards of ICs (mainframes).This is necessary to achieve the high speeds and computational power of larger computers. On the other hand, the CPU of a microcomputer is contained in a single integrated circuit. They are known as a microprocessor.

It was pointed out above that microprocessors are single-chip CPUs used in microcomputer. A microcontroller contains, in a single IC, a CPU and much of the remaining circuitry of a basic computer system. A microcontroller has the CPU, memory (RAM, ROM) and the I/O interface (parallel, serial) all within the same IC. Of course, the amount of on-chip memory does not approach that of even a modest microcomputer system. Microprocessors are most commonly used as the CPU in microcomputer systems. Microcontrollers, on the other hand, are found in small, minimum-component designs performing control-oriented activities, such as the traffic lights. These designs were often implemented in the past using dozens or even hundreds of ICs. A microcontroller aids in reducing the overall component count. All that is requires is microcontroller, a small number of support components, and a control program in ROM.

A common misunderstanding of the phrase "Reduced Instruction Set Computer" is the mistaken idea that instructions are simply eliminated, resulting in a smaller set of instructions. In fact, over the years, RISC instruction sets have grown in size and today many of them have larger set of instructions than many CISC CPUs. The term "Reduced" in that phrase was intended to describe the fact that the amount of work any single instruction accomplishes is reduced at most a single data memory cycle compared to the "complex instructions" of CISCCPUs that may require number of data memory cycles in order to execute a single instruction[2]. Most microprocessors in today's market are based on either's or CISC architectures. Research has shown thatRISC architecture greatly boosts computer speed by using simplified machine instructions for frequently used functions.

II. ARCHITECTURE

The main objective of this project is to design a RISC microcontroller using VHDL. The microcontroller instruction set and features are based on Atmel AVR AT90S1200 RISC microcontroller.

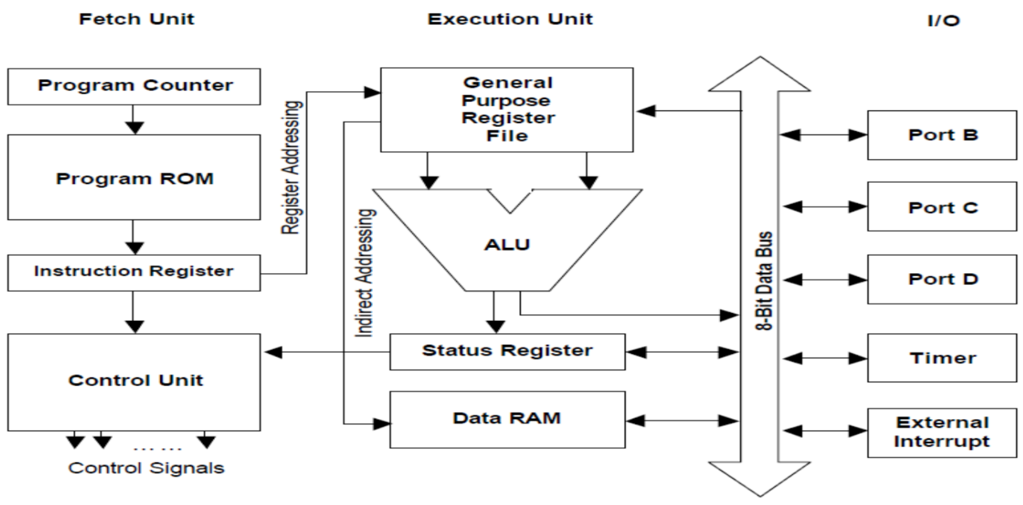


Fig. 2. Architecture Overview

Fig. 2 shows the top-level block diagram of the design, the bus structure has-been simplified, but every block represents a module to be designed. At first glance, there are 11 modules in the top-level, with the 3 ports sharing the same module. These 11modules are to be design separately using the top down design approach. Some modules like the instruction register and status register are easy to design, but modules like ALU and the control unit require a lot of understanding. The overall dataflow and bus structure between all the modules must be understand before designing the modules individually. There are basically two kinds of buses, direct bus and common bus. Direct bus connects two modules directly and is used specifically by the connected modules. There are many direct buses, such as the connection between program counter and program ROM, between program ROM and IR, between register file and ALU, etc. No control signals are required for direct buses. The data bus is the only common bus in this design. The data bus provides connection between the general-purpose register file, ALU, status register, SRAM and all the I/O features. The register file can only receive data from the data bus. All others modules can receive and send data to the data bus. Since there are so many possible data flows, control signals are required to control the correct flow direction. Only one source to the data bus is allowed at a time. If not, logic contentions will happen and the value of the data bus will be invalid. Tri-state bus is used to implement the common data bus. The impedance is so high that it can be seen as unconnected to the bus system. If the ALU is the data source, the data bus will be flooded with the result of the ALU and is available to all the connected modules. Control logic will generate an enable signal for the real destination to receive the data. The system can be divided into3 units, the fetch unit, execute unit and I/O unit. Fetch unit is in charge of fetching the next instruction and the execute unit is in charge of executing the current instruction. I/O unit provide a connection with the outside world. The fetch unit and execute unit form the CPU of the microcontroller. The first module of the fetch unit is the program counter (PC). The PC contains the address of the next instruction to be executed. It points to the program ROM to locate the instruction. The instruction from the ROM is then latched into the instruction register (IR). The control unit takes the content of the IR and decodes it. It then asserts the appropriate control signals to execute the instruction. All modules are connected with direct buses. The execute unit in charge of executing most instructions. Normally, to execute an instruction, 2 operands are output from the register file to the ALU. The ALU then perform the operation and send the result to the data bus. Contents of the data bus (the result) are then stored back to the register file. The ALU also evaluate the status register flags and send them directly to the status register (SR). The whole execution process is done in a single cycle. The ALU perform many operations - include passing the contents of a general register to the data bus. SR also has a direct bus connection to the control unit required for branch evaluation. The register file misaddressed directly by some bits in IR.

III. INSTRUCTION SET

The operation of the CPU is determined by the instruction it executes, referred toes machine instructions or computer instructions. The collection of different instructions that the CPU can execute is referred to as the CPU's instruction set. Since the instruction set defines the data path and everything else in a processor, it is necessary to study it first. Table 1 shows the instruction set summary of the designed microcontroller, while the instruction set summary of the original AT90S1200 is shown. There are 92 instructions grouped into 4 categories: arithmetic and logic instructions, branch instructions, data transfer instructions and the bit and bit-test instructions. As mentioned earlier, instruction set of the design is based on Atmel AVR AT90S1200instruction set. In this way, the design can use the same assembler and simulator provided by Atmel since the final design is actually an AT90S1200 compatible microcontroller.

Table 1. Instruction Set Summery

Mnemonic	Operation	Flags	# Clocks
ARITHMETIC AND LOGIC INSTRUCTIONS			
ADD	Add Two Registers	S,Z,C,N,V,H	1
ADC	Add with Carry Two Registers	S,Z,C,N,V,H	1
SUB	Subtract Two Registers	S,Z,C,N,V,H	1
SUBI	Subtract Constant from Register	S,Z,C,N,V,H	1
SBC	Subtract with Carry Two Registers	S,Z,C,N,V,H	1
SBCI	Subtract with Carry Constant from Register	S,Z,C,N,V,H	1
AND	Logical AND Registers	S,Z,N,V	1
ANDI	Logical AND Register and Constant	S,Z,N,V	1
OR	Logical OR Registers	S,Z,N,V	1
ORI	Logical OR Register and Constant	S,Z,N,V	1
EOR	Exclusive OR Registers	S,Z,N,V	1
COM	One's Complement Register	S,C,Z,N,V	1
NEG	Negate (2's Complement) Register	S,C,Z,N,V,H	1
SBR	Set Bit(s) in Register	S,Z,N,V	1
CBR	Clear Bit(s) in Register	S,Z,N,V	1
INC	Increment	S,Z,N,V	1
DEC	Decrement	S,Z,N,V	1
TST	Test for Zero or Minus	S,Z,N,V	1
CLR	Clear Register	S,Z,N,V	1
SER	Set Register	None	1

IV. Addressing Modes

There are 7 addressing modes in the microcontroller. Rd and Rr are devoted to the destination register and soure register.

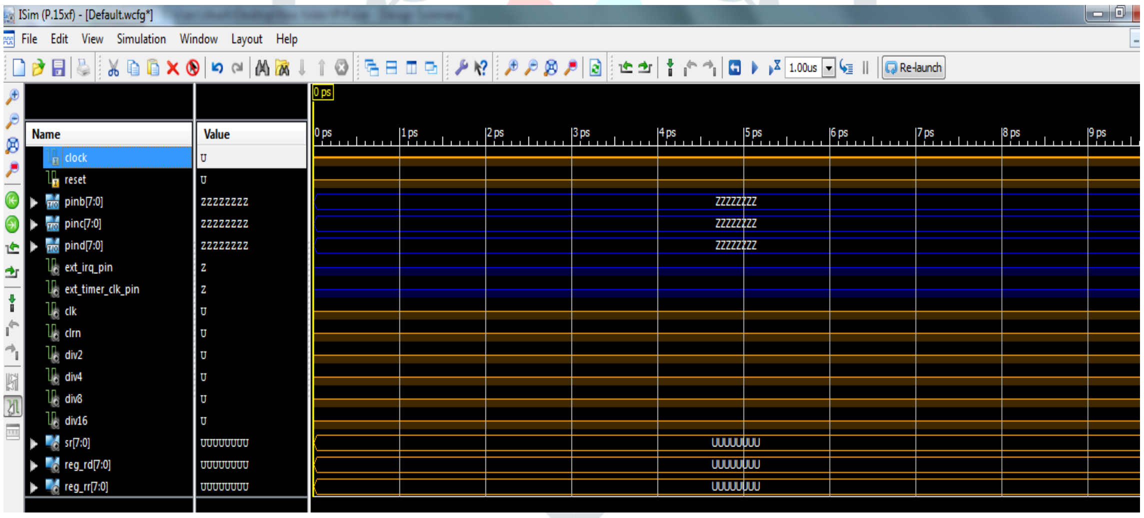
1. Direct Single Register Addressing The operand is in Rd.
2. Direct Double Register Addressing The operands are in Rd and Rr. Result is stored back to Rd.
3. I/O Direct Addressing First operand is one of the I/O registers. The address is contained in 6 bits of the instruction word. The second operand is either Rd or Rr. Used by IN and OUT instructions to read from or write to the I/O registers.
4. Data Indirect Addressing Operand address is the contents of the Z-register. Used when accessing the SRAM with LD and ST instructions.
5. Data Indirect Addressing with Pre-Decrement Z-pointer is decremented by 1 before the operation. Operand address is the decremented contents of the Z-register. Used when accessing the SRAM with LD and ST instructions.
6. Data Indirect Addressing with Post-Increment the Z-register is incremented by 1 after the operation. Operand address is the original content of the Z-register before increment. Used when accessing the SRAM with LD and ST instructions.
7. Relative Program Memory Addressing Program execution continues at address PC + offset. The offset is contains in the instruction word. Unconditional branch instructions (RJMP, RCALL) can reach the entire program memory from every location. However, conditional branch instructions can only reach –64 to 63 locations away from the current address.

V.VHDL

VHDL is acronym for very high speed integrated circuit Hardware Description Language. It is designed to fill a number of needs in the design process. First it allows description of the structure of system that is, how it is decomposed into subsystems and how they are interconnected. Second, it allows the specification of the function of a system using familiar programming language forms. Third, as a result, it allows the design of a system to be simulated before being manufactured, so that designers can quickly compare alternatives and test for correctness without the delay and expense of hardware prototyping. Fourth, it allows the detailed structure of a design to be synthesized from a more abstract specification, allowing designers to concentrate on more strategic design decisions and reducing time to market VHDL was established as the IEEE 1076 standard in 1987. In 1993, the IEEE 1076 standard was updated and an additional standard, IEEE 1164 was adopted. In 1996, IEEE 1076.3 became the VHDL synthesis standard.

VI. SIMULATION RESULTS

This project introduces a scheme of implementation of RISC controller using VHDL coding. VHDL code is developed module wise successfully run on Xilinx software. The fig. 3 shows the simulation results of microcontroller.


ISim (P.15xf) - [Default.wcfg*]
File Edit View Simulation Window Layout Help
Re-launch
1.00us
0 ps
0 ps 1 ps 2 ps 3 ps 4 ps 5 ps 6 ps 7 ps 8 ps 9 ps
ZZZZZZZZ
ZZZZZZZZ
ZZZZZZZZ
UUUUUUUU
UUUUUUUU
UUUUUUUU

Name	Value
clock	U
reset	U
pinb[7:0]	ZZZZZZZZ
pinc[7:0]	ZZZZZZZZ
pind[7:0]	ZZZZZZZZ
ext_irq_pin	Z
ext_timer_clk_pin	Z
clk	U
clrn	U
div2	U
div4	U
div8	U
div16	U
sr[7:0]	UUUUUUUU
reg_rd[7:0]	UUUUUUUU
reg_rr[7:0]	UUUUUUUU

Figure 3. Simulation Result

Design utilization summery is describe in table 2 and different between AT90S1200 Vs current design is shown in table3

Table 2. Design utilization summery

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	463	5888	7%
Number of Slice Flip Flops	376	11776	3%
Number of 4 input LUTs	791	11776	6%
Number of bonded IOBs	26	372	6%
Number of GCLKs	1	24	4%

Table 3. AT90S1200 Vs current design

Specification	AT90S1200	Current Design
Instructions	89	92
G.P Registers	32	16
Program ROM	512 words	512 words
SRAM	None	128 bytes
Hardware Stack	3 Level Deep	4 Level Deep
I/O Ports	2 (15 pins)	3 (24 pins)
Addressing Modes	5	7
Speed	4 MHz / 12 MHz	12 MHz
8-bit Timer	1	1
External Interrupt	1	1
Implementation	CMOS	FPGA
Others	Analog Comparator, Watch Dog Reset, EEPROM, Internal Pull Up Resistors	None

VII. CONCLUSION

As a conclusion, this project has been completed successfully fulfilling are the objectives and scopes specified. The author has used his extra time to optimize the speed of the design until 12 MHz the data RAM that is not specified in the scope of the project has also been included. Hardware stack is enlarged to 4-level instead of 3 and a total of 24 I/O lines are available.

REFERENCES

[1]. Mamun B, Shabiul I. and Sulaiman S,"A SingleClock Cycle MIPS RISC Processor Designusing VHDL"

[2]. Hamblen J." Synthesis, Simulation, andHardware Emulation to Prototype a PipelinedRISC Computer System"

[3]. Zainalabedin N,"VHDL for Modeling and Designof Processing Units"

[4]. Takanori M, Satoshi A and Masaaki I, "AMultithread Processor Architecture Based on theContinuation"

[5]. Kasuga-Koen, Kasuga, Fukuoka, " The InnovativeArchitecture for Future Generation HighPerformanceProcessors and Systems"

[6]. Virendra S. and Michiko I,"Instruction-BasedSelfTesting of Delay Faults in PipelinedProcessors", IEEE Transaction on VLSI systems,vol. 14, no.11,pp.1203-1215.

[7]. Patterson A. and Hennessy J,"ComputerOrganization & Design", Morgan KaufmannPublishers, 1999

[8]. 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.

[9]. "Implementation of RISC Processor on FPGA" Pravin S. Mane, Indra Gupta, M. K. Vasantha Computer Science & Engineering Department, Mody Institute Of Technology & Science, Lakshmangar ,2006 . "Electrical Engineering Department, Indian Institute of Technology Roorkee, Roorkee-247667

[10]. Tanaji M. Dudhane and Charudatta V Kulkarni "VLSI design of 8 bit microcontroller using reconfigurable hardware "MITCOE, Pune ,India . International Journal For Technical Research And Development ,Jan 2012 .