CHAPTER I

INTRODUCTION

1.1 Central Processing Unit

Figure 1.1 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.

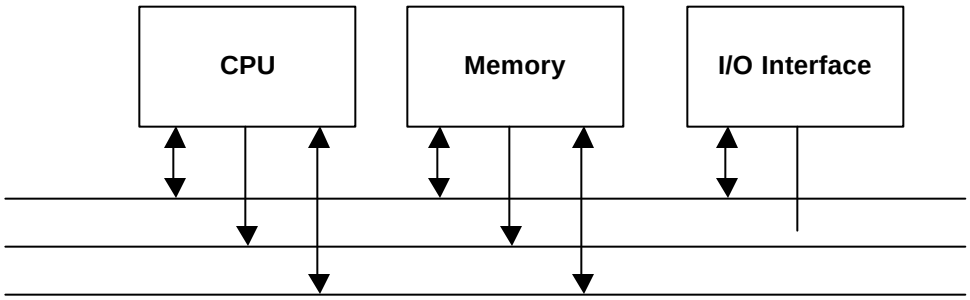

Figure 1.1 Basic Computer System

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,

1.3 Objectives

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

1.4 Atmel AVR AT90S1200

The AT90S1200 is a low-power CMOS 8-bit microcontroller based on the AVR RISC architecture. It has 89 powerful instructions and 32 general purpose registers.

7 6 13.5 Q 7 .33452 T709. Tf 0 /F5 Tj ET q75 6 130 -20.2 Tf 0 75 6 130 -21Tf 0 75 6 130 -20.

Most of the CISC systems are microprogrammed, because of the flexibility that microprogramming offers the designer. Different instructions usually have microroutines of d instruct 40.25qrogramc This means

CHAPTER III

DESIGN METHODOLOGY AND CAD TOOLS

3.1

Hardware design is done with the related CAD tools. The first step in the hardware design is to prepare the specification of the design (the microcontroller). The architecture and the instruction set must be understood thoroughly. The design ideas are then describe with VHDL in a text editor. Then, the VHDL code is synthesized with FPGA Express. If synthesized successfully, FPGA express will generate a netlist files (EDF file). This file is then send to Max+Plus II for compilation and simulation. Results are verified by simulation. The hardware design process is repeated until the microcontroller is complete without any errors.

Hardware implementation is performed by downloading the design into the targeted FPGA device, Altera EPF10K20RC240-4. The hardware implementation tests the design in real physical environment by some control applications. A microcontroller can perform thousands of control applications. For every application, different programs must be written and store into the program ROM of the microcontroller before it can do the job. So, before the mi

Windows Notepad Editor. Then all the files are loaded in a project in FPGA Express. It will check the VHDL file for syntax errors. If there are no errors, we can ask FPGA Express to create implementation for the project. Once the implementation is created, the EDF net list file of the implementation can be exported and used by MAX+plus II for compilation.

3.3 MAX+Plus II

MAX+Plus II is a free software provided by Altera. It has many sub components and the important components are the compil

CHAPTER IV

INSTRUCTION SET

instructions are single cycle except branch instructions, the LD/ST instructions and a few others.

Table 4.1 Instruction Set Summary

Mnemonic	Operation	Flags	# Clocks

4.2 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**

Although there are 7 addressing modes of all, direct register addressing (mode 1 and 2) are used most of the time. Others mode are used when accessing the I/O,

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				K	K	K	K	d	d	d	d	K	K	K	K

Table 4.4 Machine Codes

```
NOP   0000 0000 0000 0000
```

CHAPTER V

PIPELINE PROCESSING

5.1 Instruction Cycle

Figure 5.1 shows the the instruction cycle –

22 is fetched. Instruction 21 cannot be executed and is flushed from the pipeline. Only at T4, instruction 73 is being fetched. Instruction 22 must also be discarded. Finally at T5,

CHAPTER VI

MICROCONTROLLER ORGANIZATION

6.1 Pin Description

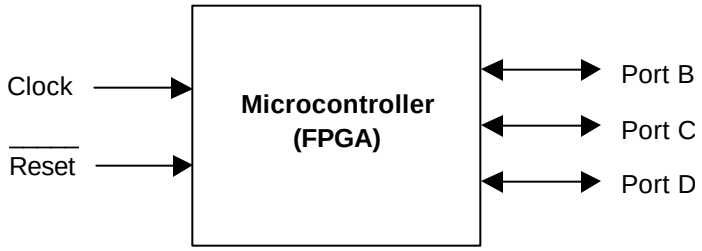

Figure 6.1 Microcontroller Pin Configuration

Figure 6.1 shows the pin configuration for the designed microcontroller. The microcontroller has 2 input pins and 3 bi-directional I/O ports. Each I/O port consists of 8 individual I/O pins. So 3 I/O ports contribute to a total of 24 I/O pins. The clock signal will drive the whole microcontroller directly. Reset is active low; when asserted it resets the microcontroller to the default state even if the clock is not running. Port B, Port C and Port D are all 8-bits port. Each bit can be configure to be input or output. All port pins are tri-stated when the microcontroller is reset. Pin D7 also serves as the external interrupt source and external timer clock source.

the appropriate control signals to execute the instruction. All modules are connected with direct buses.

All the I/O modules contain many control registers. Data are sent to and received from it through the common data bus. Table 6.1 shows the complete list of the I/O control registers and their corresponding address. Reserved and unused locations are not shown in the table. The SR is also mapped into one of the I/O address. IN and OUT instructions are used to transfer data between these control registers and the general registers. The lower half of the control registers ($00 - $1F, shaded in gray) are directly bit-accessible using the SBI and CBI (Set/Clear Bit in I/O) instructions. Using SBIS and

ROM to the instruction register. The instruction register will receive the instruction in

Figure 6.4 Control Signal Timing

CHAPTER VII

DATAPATH DESIGN

7.1 Chapter Overview

The design of the microcontoller is discussed in 2 separate chapters – one for the datapath and one for the control unit. This chapter discusses the design of the datapath while the next chapter will discuss about the control unit. All modules in the top-level block diagram (Fig

7.2 Program Counter (PC)

Figure 7.1 Program Counter Architecture

Figure 7.1 shows the architecture of the PC module. In the most basic execution stream, the PC is incremented on every clock transition. But in some cases, the PC will be loaded with a new value instead of incrementing it. Hardware stack is used to keep the return address of a subroutine call or interrupt request. Program counter backup register (PCB) always loaded with the last PC value.

There are 3 circumstances that the PC will be loaded with a new value instead of incrementing it. The first is serving a branch instruction (conditional or unconditional); the second is serving an interrupt request; and the third is returning from a subroutine or interrupt service routine (ISR).

The description for branch instructions is PC ?

instruction to be fetched? Recall the pipelining discuss in chapter 5, when the CPU is executing the N^{th} instruction, the PC has already increased to N + 2 and the instruction in the IR is the N + 1 instruction. The PC in the description is actually the address of the branch instruction itself, not the real hardware PC. So PC + 1 points to the next

b

the pull operation. If there are more that 4 subsequent subroutine call or interrupt request, the first return address that is pushed into the stack will be lost.

7.5 General Purpose Register File

WR_REG signal is asserted if the result of the ALU will be written back to the destination register. It is a register file control signal, not the ALU. ORA and ORB columns shows what should be loaded into the operand register A and operand register B. If the cell is blank, it is default to Rd (destination register) for ORA and Rr (source register) for ORB. Zero is “0000 0000”; One is “0000 0001”; and Imm is the immediate value of the instructions.

should be loaded. SBI and CBI instruction require 2 cycles to complete. At the first cycle, the I/O register is fetched to ORA through the data bus. At the second cycle, a 0 or 1 is loaded to the bit location and result is written back to the I/O through the data quireio complete. At the

7.6.2 Execution Unit

The execution unit executes 7 groups of instructions that are discussed earlier - 5 groups from the basic instructions (ADD, SUBCP, LOGIC, RIGHT and DIR), the bit load group and the pass ORA group. As shown in Figure 7.11, the execution unit is divided into 5 subunits. Adder-subtracter executes instructions from both the ADD and SUBCP group. Logic unit executes instructions from the LOGIC group. Shifter for the RIGHT group; direct unit for the DIR group; and bit loader for the bit load group.

7.8 Data RAM

The actual AT90S1200 chip does not contain any SRAM. The AT90S2313 contains 128 Bytes of SRAM. Figure 7.13 show how the SRAM is organized in AT90S2313. The 32 general purpose registers and 64 I/O registers are mapped into the

Figure 7.14 shows the organization of the data RAM module. It contains two registers –

read the value of the physical pin to the data bus. The data flip-flop value does not change according to the phisical pin.

When the pin is configured as output (direction = 1), the tri-state buffer that connects to the data flip-flop is now enabled. The physical pin will be directly driven the the value of the data flip-flop.

-

The port are connected directly to the data bus. When writing to the data flip-flop and direction flip-flop, data is (onr6. 0 778data isIta bus. When ectia bu to e signwildata fl85.5 -79 TD -0.388

--on TD -0.2988 T37464131 T 746413lipl be diree valitsoed tenh c

-

f14P the7l_25 c0802T c3(3-)0Tj4628951.24975(v_op cb?v2d)0.2DT8TIIc 0) Tj 02895 (.2895) Tj -165

7.10 Timer

The timer is a simple 8-bit timer with overflow detection and interrupt request. There are 4 control registers in the timer – timer/counter interrupt mask register (TIMSK) at $39, timer/counter interrupt flag register (TIFR) at $38, timer/counter 0 control register (TCCR) at $33 and timer/counter 0 (TCNT0) at $32. Figure 7.18 shows the control bits in these registers.

	7	6	5	4	3	2	1	0
TIMSK	-	-	-	-	-	-	**TOIE0**	-

It is important to note that the timer clock source does not drive the TCNT0 directly. Instead, TCNT0 is driven by the system clock. The timer clock source are sampled at the rising edge of the system clock. If a low to high transition is detected (a low is sampled followed by a high), the increment signal for TCNT0 is asserted to

interrupt request is served. The 4 RD signals read the timer control registers to the data bus while the 4 WR signals write the data bus value to the corresponding register.

Figure 7.22 shows the symbol of the external interrupt module. CLR_INTF is sent by the control unit to clear the external interrupt flag when the interrupt request is served. RD and WR signals provide reading and writing the control registers through the system data bus.

CHAPTER VIII

CONTROL UNIT DESIGN

8.1 Chapter Overview

The design of the datapath has been discussed in the last chapter. Only one module is left for the design – the control unit module, which will be discussed in this chapter. We have touched the instruction set, pipeline processing and many control signals, which controls the datapath. The control unit plays the role on decoding the instruction, implements the pipeline processing and asserts the control signals for the datapath at the correct timing. This chapter covers the decoding of the instruction and the design of the finite state machine (FSM).

8.2 Instruction Decoder

The inputs of the control unit are the instruction machine code from instruction register (IR), the flags value from status register (SR), skip request, timer interrupt request (timer IRQ) and external interrupt request (external IRQ). The machine code is decoded first before sending to the FSM, while the others inputs are connected directly to the FSM.

As discussed in chapter 5, the design process involves 51 machine codes. The instruction decoder takes the 16-bit machine code from the IR and generates 46 output signals to represents the 51 inst

The FSM contains only 8 states. Such a small number of states are results of using synchronous Mealy implementation. This is the second advantage. Since the state machine outputs are now gated to flip-flops, all single cycle instruction can share the same state. The state is unchanged but the input changed, so it can determine the next output.

8.4 Finite State Machine States

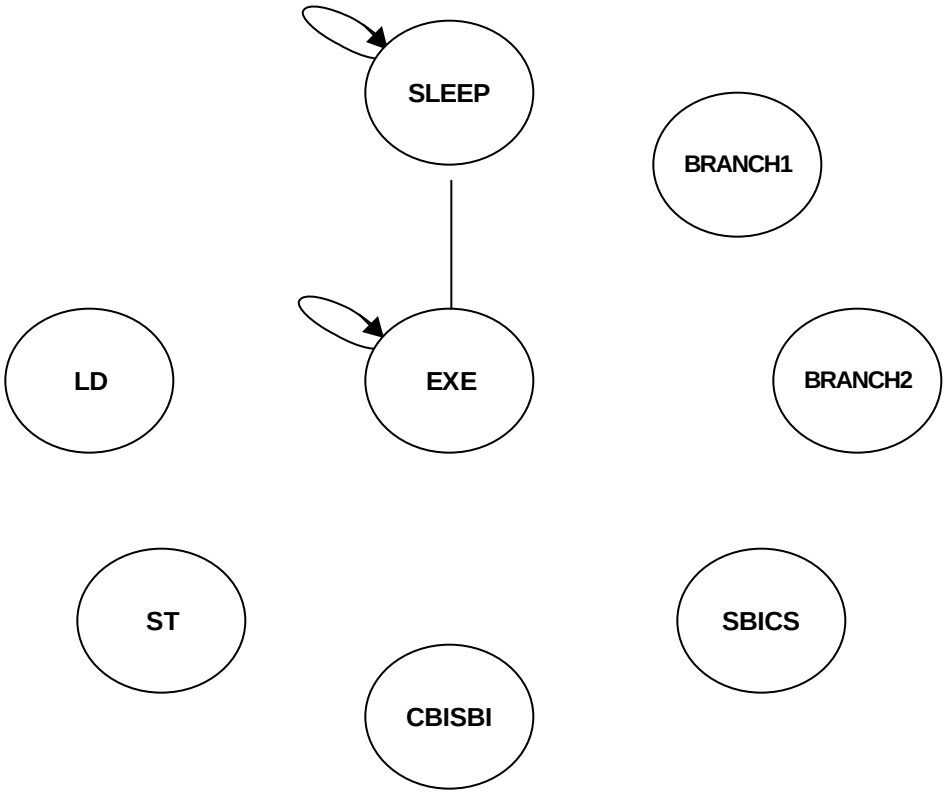

Figure 8.2 State Diagram

Figure 8.2 shows the state diagram of the finite state machine (FSM). The 8 states are EXE (execute), SLEEP, BRANCH1, BRANCH2, SBICS (skip if bit in I/O clear/set), CBISBI (clear/set bit in I/O), ST and LD.

The state diagram shows the state flow but does not clearly show the inputs. The inputs to the FSM are the 46 output lines of the instruction decoder, timer IRQ, external IRQ, skip request and branch request. Branch request is 1esr2 0 TD D.Icrion decoder, timer IRQ, external

When either BRBC or BRBS is fetched, the shared instruction decoder output line become active. Different from unconditional branch instructions, there will be no state chance on the next cycle. The FSM will assert the branch test signal on the next cycle to request the branch evaluation unit to perform a branch test. If the condition is not fulfilled, no branch request is generated. The pre-fetched instruction is not flushed from the pipeline and is executed. So it takes only one cycle for a conditional branch instruction if the branch if not taken.

If the condition is fulfilled, the branch evaluation unit will send back a branch request to the control unit immediately. At the same time, the control unit will also instruct the PC to loads the PC with the destination address. With the branch request, the FSM will transfer to BRANCH2 state on the next clock and the pre-fetched instruction is flushed. On the next clock, the second pre-fetched instruction is also flushed but the FSM now return to EXE state. The next instruction is the destination instruction and will be executed on next cycle. So it takes 3 execution cycles if the branch is taken for conditional branch instructions. Note that the control signal to load the PC is not asserted according to clock transition. It is asserted only after the branch evaluation unit has received the branch test signal and performs the test successfully. So there is delay for the PC to receive the signal 00.33f2p,ase.

When the FSM sees the SLEEP instruction, it will jump to the SLEEP state. When in the SLEEP state, the PC is stopped and no instruction is executed. Only when there is an IRQ (with the I-flag set), the FSM jumps to BRANCH1 state to serve the interrupt request. The process is exactly the same as serving an IRQ from the EXES.

For single cycle instruction, the instruction will not need to be remembered after the control signals is asserted because it f2p,ompleted in one cycle. When enter the execute cycle, the next instruction is fetched and the current instruction is lost. However, instructions that require 2 cycles to complete must have some way to remember the instruction in order to assert the correct control signals at the second cycle. So, the FSM

provides the second state to remember the instruction. Control signals are based on the state itself without considering the decoder's output line.

If the second cycle of the trol signal31( )D7mt the seameconsrol signals ,the the

8.5

As discussed in the ALU operand fetch unit section is Chapter 7, ASEL and BSEL control signals are used to select what should be loaded into the operand registers.

Figure 9.1 Altera UP1 Educational Board

The Altera UP1 (University Program) Educational Board as shown in Figure 9.1 is the only FPGA device available in the LAB. It has two FPGAs on it for developing complex programmable logic applications. The MAX7000 device on the left side of the board typically supports 2,500 gates for simple designs. The FLEX10K20 on the right

supports 20,000 gates, and includes connections to a DB25 VGA connector, as well as a PS/2 mouse port. The system is programmable via a PC pa

Figure 9.2 shows the pin arrangement of the FLEX10K device on the UP1 board. Before the design is programmed into the device, pin assignments must be made to map all the pins of the design to the physical pin on the UP1 board. Table 9.1 lists the pin assignments used.

Port B is configured as output and is used to control two 7

9.4 Fitting Report

```
***** Project compilation was successful


** DEVICE SUMMARY **

Chip/                          Input  Output  Bidir  Memory 0  Memory 0          LCs
POF       Device               Pins   Pins    Pins   Bits %Utilized  LCs  %Utilized

Top|comb.
          EPF10K20RC240-4      2      0       24     10240    83 %    1608    92 %

User Pins:                     2      0       24
```

9.5.1 Simple Calculator

The first application is a simple calculator that can only perform add and minus operations. The keypad is the input of the calculator and the two 7-segments digits are the output. The # key is he ay is he a# ko BTpress a+) ay iwhitor Tj * ay is he a# ko TD ( ) Tj 36

CHAPTER X

SUGGESTIONS AND CONCLUSION

CHAPTER X

10.2 Conclusion

As a conclusion, this project has been completed successfully fulfilling are the objectives and sc

Reference

[1] Daniel Tabak, *RISC Systems*, Research Studies Press Ltd.: Taunton, Somerset, England TA1 1HD, 1990

[2] M.Morris Mano, *Computer System Architecture*, Prentice Hall inc.: Englewood Cliffs, New Jersey 07632, 1993.

[3] *AVR 8-bit RISC Microcontrollers Data Book*, Atmel Corporation, San Jose, California 95131, August 1999.

[4] Randy H. Katz, *Contemporary Logic Design*, The Benjamin/Cummings Publishing Company, Inc.: Redwood City, California 94065, 1994.

[5] Douglas L. Perry, *VHDL*