Name: Stephen Campbell

Netid: sac170630

Computer Architecture
EE/CE 4304
Spring 2021 Project v1.0

VERSION Stephen Campbell

Instructions

This is your personalized project.

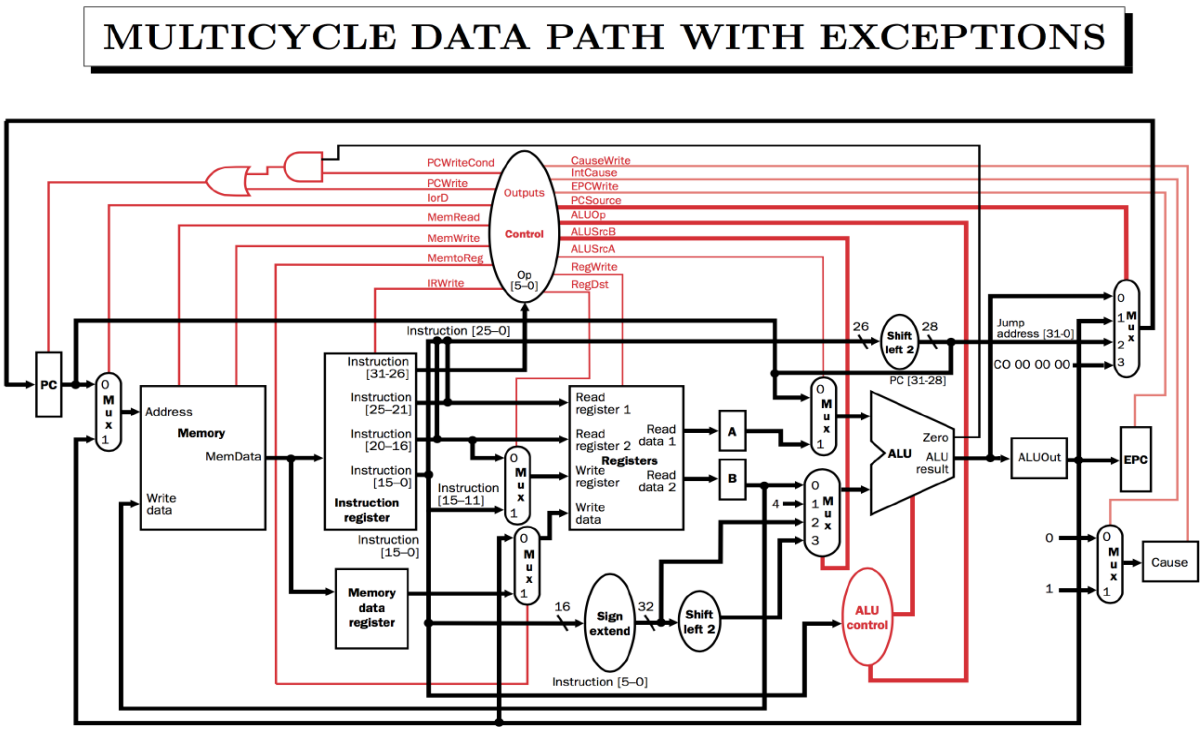
Consultation with any person (other than the TAs or Professor) at any location constitutes cheating. However, you are free to visit the Web and explore solutions to your design problem. In fact, ASIC World at http://asicworld.com and similar sites are encouraged.

If cheating is detected, the incident will be dealt with according to established UT-Dallas procedures. Possible disciplinary actions for academic dishonesty include a failing grade on this project.

Problem Statement—Version sac170630

In this project, you are to design your own Instruction Set Architecture (ISA) and a multicycle computer which implements it. You will be given specifications for your computer which make it unique from other students in this class. You will implement the architecture shown in Figure 1 below. In order to make the project less daunting, the project will have two phases: 1) ISA design and specification and 2) implementation and simulation using Verilog.

Figure 1: Multi Cycle Computer Architecture



After David A. Patterson and John L. Hennessy, *Computer Organization and Design*, 2nd Edition

Specifications for sac170630

Your computer specification follows:

Your address bus has a width of 11 bits which means that your computer must support 2048 addresses. This address space is shared between the instructions and your computer's program data space. The OS architect has already determined that the entry point of programs will be at the center of the address space or 0x400. All programs will be loaded at that address and any branches should be relative to that starting address.

Your computer will have a data width of 64 bits. Your register files will be addressed using 2 bits resulting in a register file which is 4 by 64.

You are to design your own instruction set architecture with an instruction that is 20 bits wide. Your instruction should accommodate an immediate field which has 8 bits. You should have 6 bits devoted to instruction types which means your computer can support 64 instructions.

Your computer will need to support two simple programs:

$C \longleftarrow A + B$

and

```
int sum = 0 ;
for( i = 0 ; i <= 10 ; i++ ){
  sum += i ;
}
```

Phase 1 - Instruction Set Design

You will have a week and a half to design and specify your instruction set. You will need to determine your command set. How will you define the fields for each instruction. You need to determine whether you want to use two or three operand codes. How will your registers be named and defined? You are free to borrow ideas from MIPS, x86, RiscV, PowerPC, and any computer Assembly Language. You will need to provide Verilog on how to decode your instructions. For an example, here is a slice of the MIPS architecture:

```
module instruction_fields(
    input [31:0] instruction,
    output [4:0] Register_Rs,
    output [4:0] Register_Rt,
    output [4:0] Register_Rd,
    output [15:0] immediate ) ;

    assign Register_Rs = instruction[25:21];
    assign Register_Rt = instruction[20:16];
    assign Register_Rd = instruction[15:11];
    assign immediate = instruction[15:0];
endmodule
```

Phase 2 - Implementation in Verilog and Simulation

In the second phase of the project, you will implement the various Verilog modules which will implement your design. You may use Xilinx ISE, EDA Playground, Synopsys, or Cadence tools to simulate your design. Your implementation will be tested on the two programs above.

Notes

This document will be updated with more details and information as needed.

Specifications for sac170630

Address Bus Width: 11 bits
Number of Addresses 2048
Program Load Address: 0x400
Data Bus Width: 64 bits
Number of Register File Address Bits: 2
Number Register File Registers: 4
Width of Register File Register : 64
Instruction Width: 20 bits
Size of Immediate Field: 8 bits
Maximum number of Instructions: 64