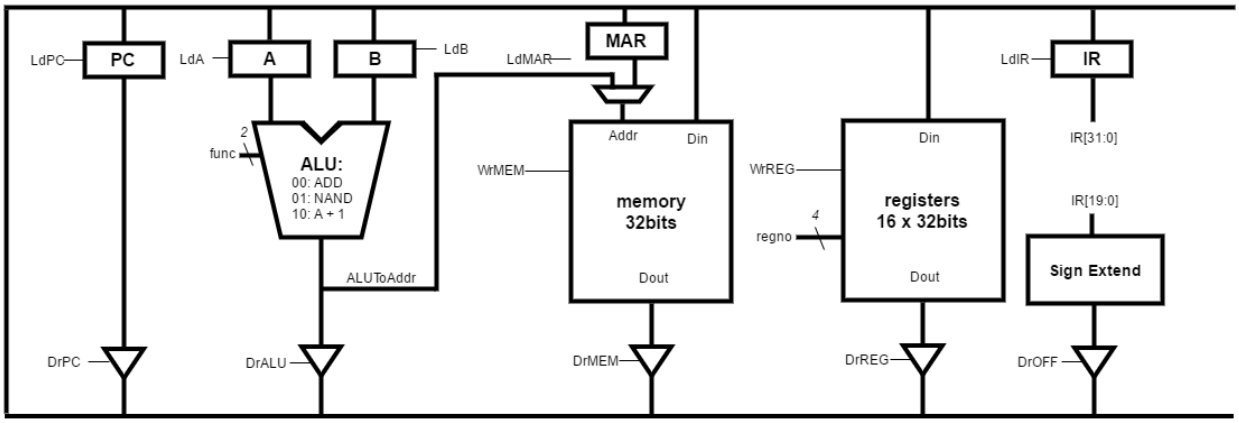
CS 2200 - Homework 2
Spring 2017

Important information:

1. Submit to T-Square as a PDF file.
2. This is an individual assignment. No collaboration is permitted.
3. Due date: February 13, 11:55pm

Problem 1: Datapath

Below is a modified datapath, the LC-2201, including an extra wire that feeds the output from the ALU directly to the address input for memory (ALUToAddr). Points will be taken off for inefficient answers.



1. Give the states to control the **UNMODIFIED** LC-2200 datapath in order to execute the SW instruction. Use the format of the given state below, and only list signals that are asserted.

   State0: DrREG, regno = RY, LdA

   State1: DrOff, LdB

   State2: DrALU, Func00, LdMAR

   State3: regno = Rx, DrReg, WrMEM memoryMux = MAR

2. Give the states to control the **MODIFIED** LC-2201 datapath in order to execute the SW instruction.
   - State0: DrReg, regno=RY, LdA
   - State1:DrOFF, LdB
   - State2: memoryMux = ALUToAddr, DrReg, regno = RX, WrMEM

3. What is the CPI for the SW instruction with and without the DPRF enhancement?

   4 cycles without
   3 cycles with

4. Assuming that the clock speed remains the same, what is the speedup for the SW instruction in the LC-2201 datapath above versus the LC-2200 datapath.

Before / After ➔ 4 / 3 ➔ 1.333333

Problem 2: Performance

1. In the year 2020, both Intel and AMD decide to abandon their work on the x86 and instead develop processors based on the LC-2200 architecture. In order to build a faster processor than their competition each manufacturer makes specific improvements to their implementation.

   - Intel's LC-2200 processor features additional hardware that offers an average speedup of 1.4x on memory accesses (LW and SW operations).
   - AMD's LC-2200 processor features better branch prediction, which gives its processor an average speedup of 1.9x on BEQ operations.

   (a) Using the chart below, calculate the speedup on Intel over AMD of GCC (GNU Compiler Collection) and SPICE (Simulation Program with Integrated Circuit Emphasis) on each manufacturer's processor.

   GCC (88.15789474/94) ➔ 0.937849944 ➔ 1.134328358 / 1.063829787 = 1.07
   AMD ➔ 52 + 21 + 25/1.9 + 2 = 88.15789474 ➔ 100/88.15789474 = 1.134328358
   Intel ➔ 52 + 21/1.4 + 25 + 2 = 94 ➔ 100 / 94 = 1.063829787
   SPICE (91.94736842/91.42857143) ➔ 0.0108757871 / 0.0109375 = 0.99
   AMD ➔ 50 + 30 + 17/1.9 + 3 = 91.94736842 ➔ 1 / 91.94736842 = 0.0108757871
   Intel ➔ 50 + 30/1.4 + 17 + 3 = 91.42857143 ➔ 1/ 91.42857143 = 0.0109375

   (b) Which processor will execute GCC faster?
   AMD

   (c) Which processor will execute SPICE faster?
   INTEL

Instruction Type	LC-2200 Instructions	Percent of Instructions in GCC	Percent of Instructions in SPICE
Arithmetic	ADD, ADDI, NAND	52%	50%
Data Transfer	LW, SW	21%	30%
Branching	BEQ	25%	17%
Jumping	JALR	2%	3%

2. You are trying to decide which of 2 computers to buy, the AMD-2500 **(2.1 GHz)** and Intel-5000 **(1.8 GHz)**. Both architectures only include 3 instructions, and you only intend to run 1 program on these architectures. The number of cycles for each instruction and their frequency in the program are listed below.

	Instruction 1	Instruction 2	Instruction 3
Cycles for AMD	10	5	3
Cycles for Intel	6	3	5
Frequency	30%	50%	20%

   (a) Compute the speedup of AMD over Intel (round to nearest hundredth). Based on this, which processor would you choose to buy?

   AMD ➔ 3 + 2.5 + 0.6 = 6.1 ➔ 6.1 / (2.1 *10^9) ➔ 2.9047 * 10^-9

   INTEL ➔ 1.8 + 1.5 + 1 = 4.3 ➔ 4.3/ (1.8 * 10^9) ➔ 2.3889 * 10^-9
   Speedup of AMD over Intel ➔ Intel / AMD ➔ 0.82
Therefore speed up is 0.82 and therefore INTEL is faster so I would buy INTEL

Problem 3: Interrupts

1. What does the **RETI** instruction do? Explain why it is necessary to make **RETI** atomic.

Return from interrupt, does 1. Load Pc from $k0, 2. Enable interrupt
It is necessary to make RETI atomic so that we can execute the instruction completely before any new interrupt could occur. If it is not atomic, then JALR will pull $k0 off the stack in two transactions, but RETI makes sure original $k0 is restored.

2. What does the Interrupt Vector Table (IVT) hold? What is the Exception/Trap register used for?

Fixed size table of handler addresses., exception or trap It holds vectors which are a unique number from each discontinuity and it outputs specific handler address.
Internally generated vector in case of traps and exceptions are stored in ETR as a unique number to represent that exception or trap.

3. Below is an interrupt handler written in LC-2200 assembly. However, it is missing some crucial lines of assembly code to correctly handle the interrupt. Your task is to fill in the missing pieces. (Refer to your class notes and chapter 4 in the book to learn exactly what sequence of tasks the interrupt handler performs).

Some helpful instructions relevant to interrupts are:

- **EI** (Enable Interrupts)
- **DI** (Disable Interrupts)
- **RETI** (Return from Interrupt)

Counter inthandler:

Sw $k0 0($sp)

Enable Interrupts

sw $s0 -2($sp)

sw $s1 -1($sp)

la $s0 counter

lw $s0 0($s0)

lw $s1 0($s0)

addi $s1, $s1, 1

sw $s1, 0($s0)

lw $s0 -2($sp)

lw $s1 -1($sp)

disable interrupts

lw $k0 0($sp)

reti

counter: 0xFFFFFC