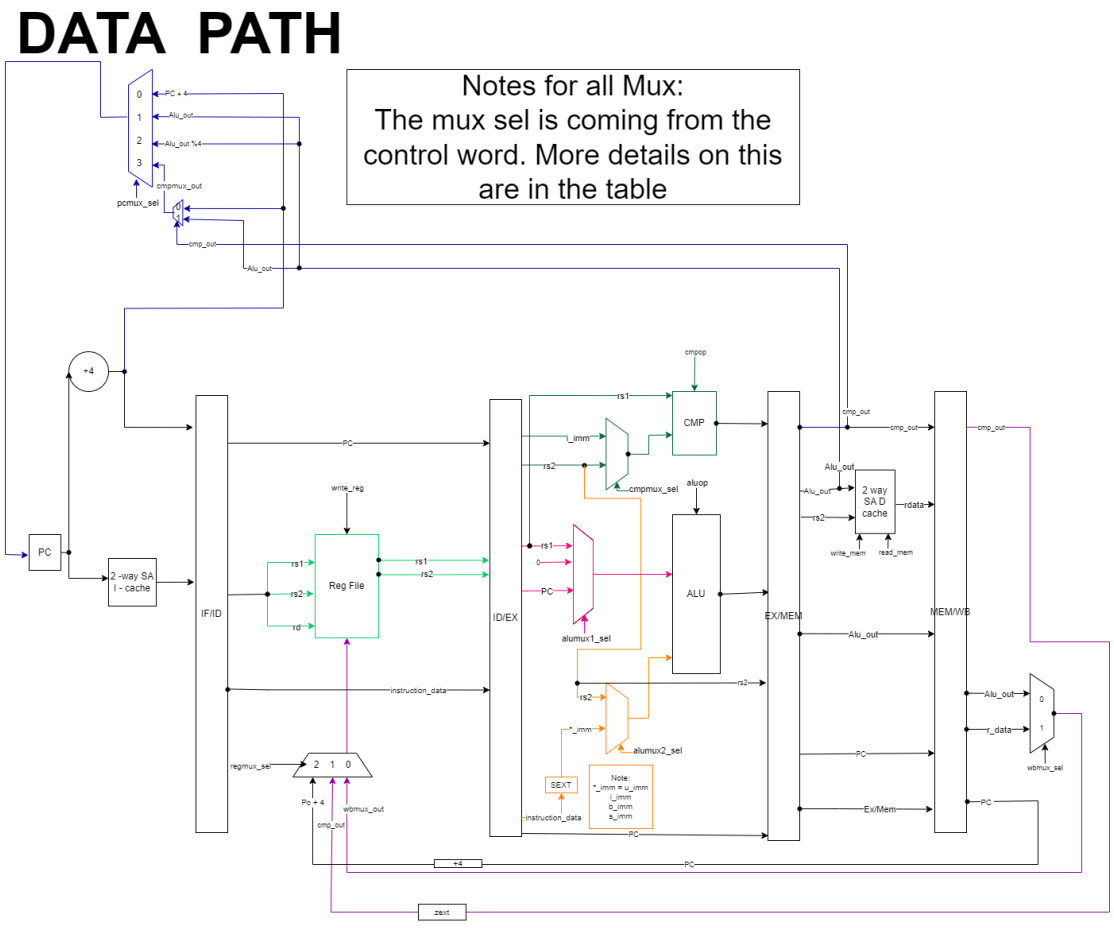
Edwin Louie, Ryan Metzger, Sohil Pokharna
"Three Don't Cares"

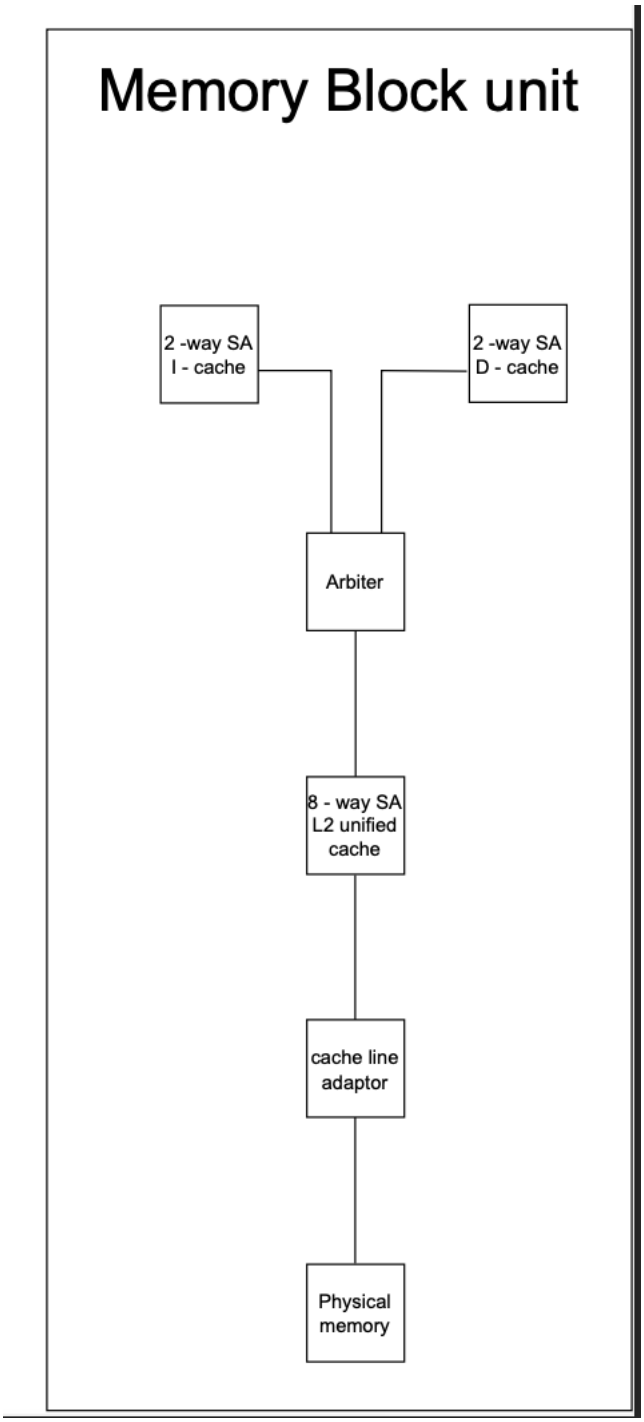
ECE 411 MP 4 Basic Datapath Design

Introduction:

For this MP, we are developing a pipelined processor using the knowledge we have gained from the course lectures. Aside from the basic datapath and handling the standard CPU instructions, we will implement forwarding and branch prediction to help improve performance. The design for the Hazard Detection Unit and Forwarding Unit has not been done. We hope to complete that in parallel with our checkpoint 1. We will continue to increase performance in future checkpoints by having L1 (2-way SA) and L2 (8-way SA). We have yet to decide other advanced features, but we hope to decide on that after we build our pipelined CPU.

Datapath:




Memory Block unit
2 -way SA
I - cache
2 -way SA
D - cache
Arbiter
8 - way SA
L2 unified
cache
cache line
adaptor
Physical
memory

Control Word Table:

Control Word [23:0]

bit 23

Control Word is generated by the opcode of the instruction and passed down the pipline.
Signals are grouped by the stage that they are used in, not necessarily the stage that contains their units

Note:

funct3 will be filled in with the apporiate value for each instruction type, it is noted as funct3 instead of constant value to shorten table.

mem_byteen will be filled wih appriate value for each instruction type, it is noted as a range 0000-

Bit 0

Instruction	IF	ID	EX					MEM			WB		
	pcmux_sel [1:0]		cmpop[2:0]	cmpmux_sel	aluop[2:0]	alumux1_sel[1:0]	alumux2_sel[2:0]	read_mem	write_mem	mem_byteen[3:0]	wbmux_sel	write_reg	regmux_sel[1:0]
ldr	pc+4		x	x	add	rs1_out	s_imm	1	0	xxxx	rdata	1	wbmux_out
str	pc+4		x	x	add	rs1_out	s_imm	0	1	0000-1111	x	0	x
imm rest (x8)	pc+4		x	x	funct3	rs1_out	i_mm	0	0	xxxx	alu_out	1	wbmux_out
slt	pc+4		blt	i_imm	funct3	rs1_out	i_imm	0	0	x	x	1	cmp_out
sltu	pc+4		bltu	i_imm	funct3	rs1_out	i_imm	0	0	x	x	1	cmp_out
sra	pc+4		x	x	sra	rs1_out	i_imm	0	0	x	alu_out	1	wbmux_out
sr	pc+4		x	x	srl	rs1_out	i_imm	0	0	x	alu_out	1	wbmux_out
reg rest	pc+4		x	x	funct3	rs1_out	rs2_out	0	0	x	alu_out	1	wbmux_out
add	pc+4		x	x	add	rs1_out	rs2_out	0	0	x	alu_out	1	wbmux_out
sub	pc+4		x	x	sub	rs1_out	rs2_out	0	0	x	alu_out	1	wbmux_out
slt	pc+4		blt	rs2_out	funct3	rs1_out	rs2_out	0	0	x	x	1	cmp_out
sltu	pc+4		bltu	rs2_out	funct3	rs1_out	rs2_out	0	0	x	x	1	cmp_out
sra	pc+4		x	x	sra	rs1_out	rs2_out	0	0	x	alu_out	1	wbmux_out
sr	pc+4		x	x	srl	rs1_out	rs2_out	0	0	x	alu_out		
jalr	alu_mod2		x	x	add	rs1_out	i_imm	0	0	x	x		
jal	alu_out		x	x	add	pc_out	j_imm	0	0	x	x		
br	cmpmux_out		fucnt3	funct3	add	pc_out	b_imm	0	0	xxxx	x		
lui	pc+4		x	x	add	0	u_imm	0	0	x	alu_out		
auipc	pc+4		x	x	add	pc_out	u_imm	0	0	x	alu_out		