

Final Project Presentation RISC Architecture

Tyler McKean, Vulindsky Fanfan, Zachary Garnes

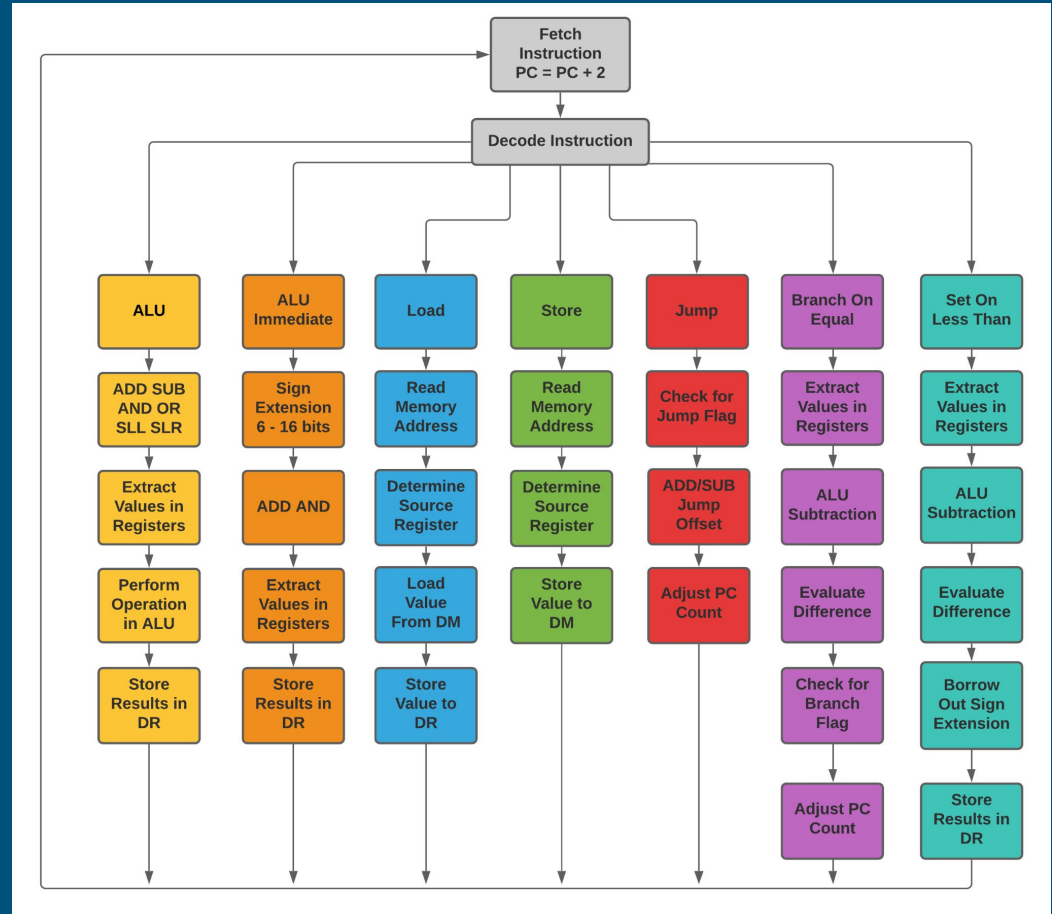
Instruction Set Architecture

- 16 bit Instructions
- 16 bit words and registers
- 8 bit memory Addresses
 - Instruction Memory
 - Data Memory
- Byte Addressable Memory
- Formats for Instructions
 - ALU
 - ALU Immediate
 - Memory
 - Control



Instruction Processing Flow Chart

- Covers all possible outcomes from Instruction
- Bits 15 -12 will be decoded from instructions to determine which function the architecture will perform



ALU Operations

0	0	0	0	0	0	0	0	0	1	0	1	0	x	x	x		
ADD				DR				SR1				SR2				Don't Care	
0	0	0	1	0	1	1	1	0	0	1	0	1	x	x	x		
SUB				DR				SR1				SR2				Don't Care	

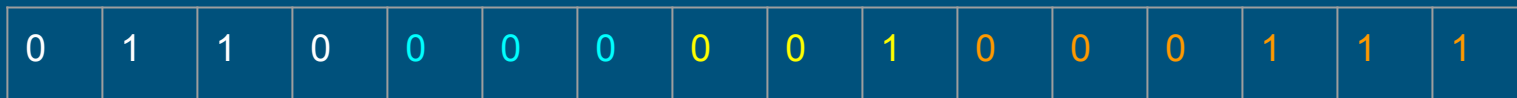
16 Bit Instruction Set

Functions

15 **Downto** 12 = Function Bit
11 **Downto** 9 = Destination Register
8 **Downto** 6 = Source Register 1
5 **Downto** 3 = Source Register 2
2 **Downto** 0 = Don't Care Bit

0000 -----> ADD
0001 -----> SUB
0010 -----> AND
0011 -----> OR
0100 -----> Shift Left
0101 -----> Shift Right

ALU Immediate Operations



ADDi

DR

SR

Immediate Value



ANDi

DR

SR

Immediate Value

16 Bit Instruction Set

Functions

- 15 **Downto** 12 = Function Bit
- 11 **Downto** 9 = Destination Register
- 8 **Downto** 6 = Source Register
- 5 **Downto** 0 = Immediate Value

- 0110 **----->** ADDi
- 0111 **----->** ANDi

Load/Store Operations

1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	x
Load				DR			Memory Address						Don't Care		
1	0	0	1	0	0	1	0	0	0	0	0	0	1	1	0
Store				SR			Memory Address						Address Bit		
1	0	0	1	0	0	1	0	0	0	x	x	x	x	x	1
Store				SR			Memory Reg				Don't Care				Address Bit

16 Bit Instruction Set

Functions

15 **Down to** 12 = Function Bit
 11 **Down to** 9 = Destination/Source Register
 8 **Down to** 1 = Memory Address
 Bit 0 = Don't Care Bit

1000 -----> Load
 1001 -----> Store

Control Operations

1	0	1	0	0	0	0	0	0	0	0	0	0	1	0	x	x
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Jump

Value to Be Added to PC

Jump Flag

F or B Don't Care

1	0	1	1	0	0	1	0	0	0	0	0	0	0	x	x	x
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Set on Less Than

DR

SR1

SR2

Don't Care

1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	1	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Branch

SR1

SR2

PC Address

16 Bit Instruction Set

Jump

15 Down to 12 = Function Bit

11 Down to 4 = Value

Bit 3 = Jump Flag

Bit 2 = Forward or Backward

1 Down to 0 = Don't Care

Functions

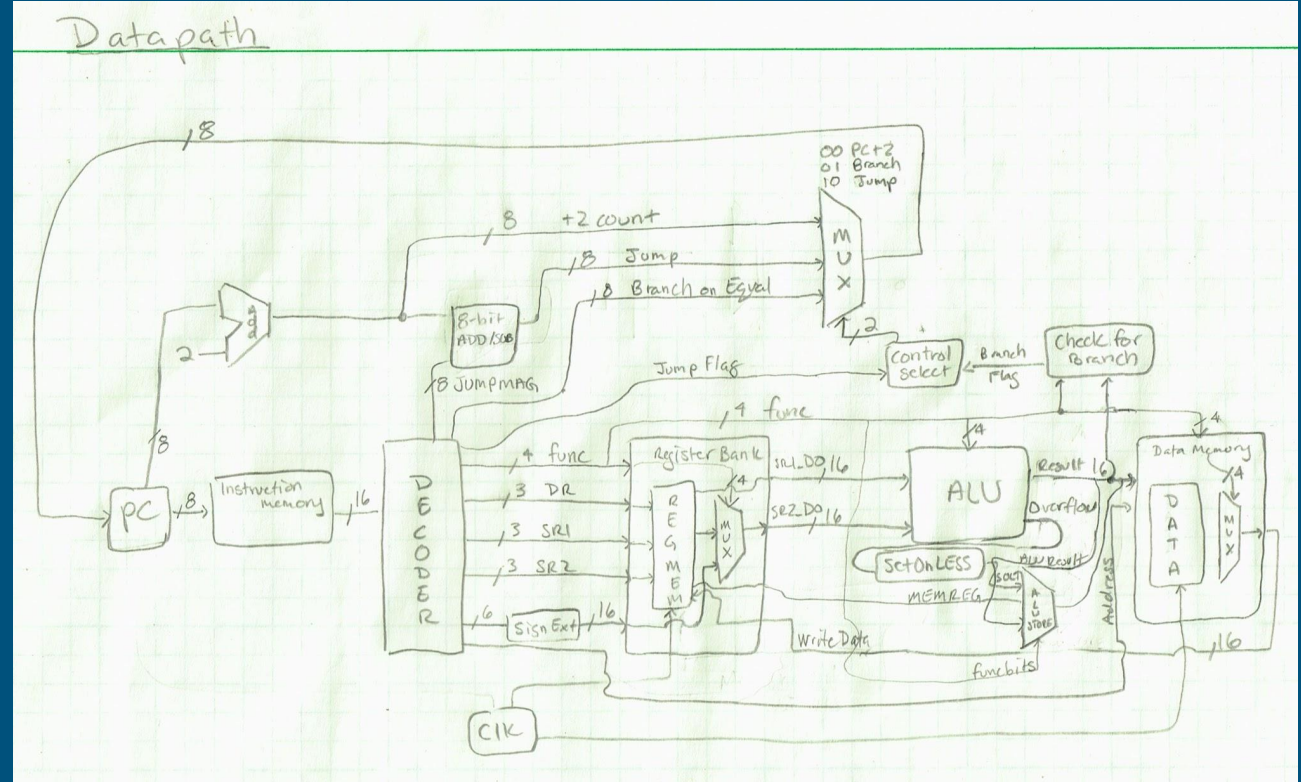
1010 -----> Jump

1011 -----> Set on Less Than

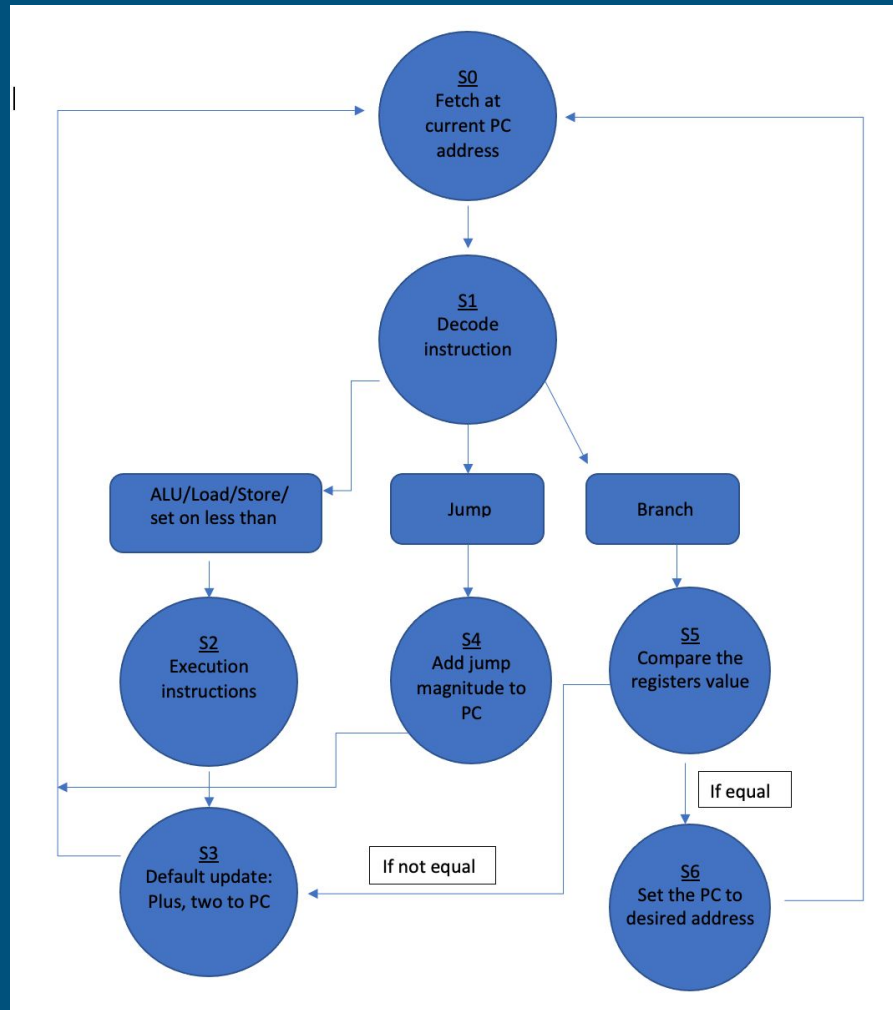
11 -----> Branch

Datapath

- Instruction Fetch Unit
- Decode Unit
- Execution Unit
- Registers
- Data Memory
- ALU

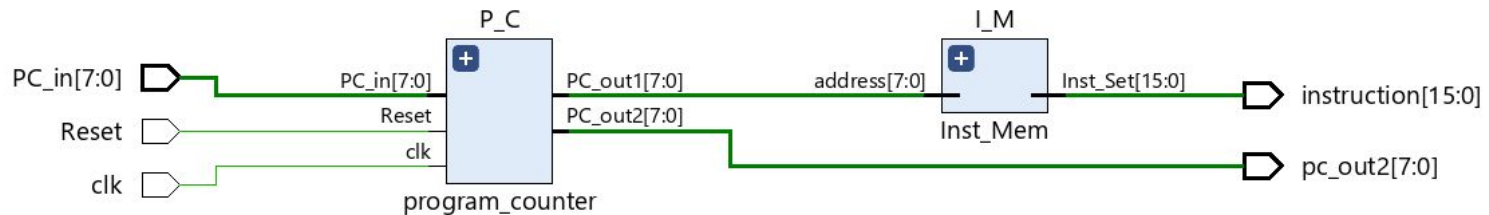


State Diagram



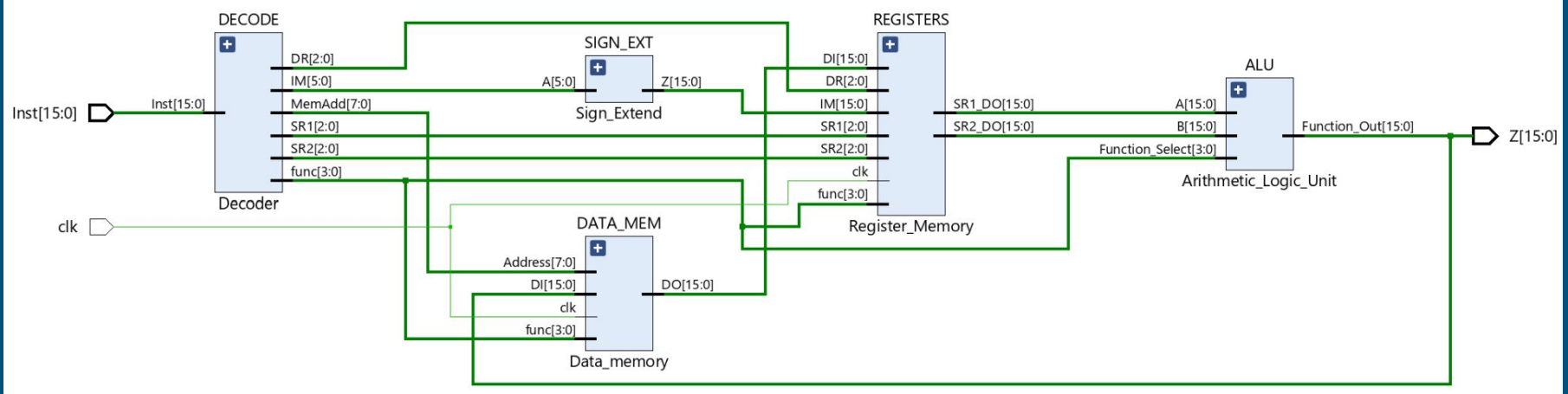
Datapath: Instruction Fetch

- Instruction Fetch Block:
 - Program Counter, Instruction Memory, Adder and Reset



Datapath: Instruction Execution Block

- Instruction Execution Block:
 - Decoder, Registers, ALU, and Data Memory



Plan to finish project

- Implement the 8-bit Adder/Subtractor for Jump Operation
- Create a Branch on Equal module that will check the ALU for result of 0x00 from Subtraction operation
- Create MUX with Control Select bits that will pass either
 - PC + 2
 - Jump Offset
 - Branch Address
- Tie all blocks into Structural module with State Machine process
- Create Testbench for verification
- Write report and submit on/before May 12th