METU NORTHERN CYPRUS CAMPUS

EEE 445 COMPUTER ARCHITECTURE I
CNG 331 COMPUTER ORGANISATION

TERM PROJECT: PART #3

FALL 2022

PART #3: Single Cycle Processor

This part will participate in %10 of the overall course grade (out of 30% which is the percentage of the overall term project). This percent will be divided as follows:

- **60% for functionality of the processor.**
- **40% for running the test code.**

1 OBJECTIVE

You will add four new instructions to the part 2 schematic in this part. First, you need to build your new control unit using table 2.7. Then you need to modify your ALU to support SLT instruction and generate the necessary flags to support the newly added instructions. A new mux will be needed to add before the PC register to finalize your design.

2 PROBLEM DESCRIPTION

Figure 2.1 shows the final schematic of the simple CPU you will build in this part. You need to modify your previous CPU to support the new four instructions:

1) **j**: jump
2) **beq**: branch if equal
3) **bne**: branch if not equal
4) **slt**: set less than

You will start by building your control (Note: new control unit will not be the same as Part 2). Next, you need to add two new control signals: PCmux, and Branch. Moreover, you need to add mux3 and connect the necessary components and wires, as shown in figure 2.1. Finally, you need to run a test assembly code on your designed CPU by compiling it to hex and storing it in the instruction memory. The Control Unit will generate the following signals according to the instruction:

1) ALUfunc: used to control the function of the ALU.
2) Mux1: responsible for choosing the value that will be stored to register file.
3) Mux2: responsible for selecting the second source ALU input.
4) Mux3: responsible for selecting the PC value.
5) Read: enable signal used when reading from memory.
6) Write: enable signal when storing a value in the memory.
7) WE: enable signal used when writing to register file.
8) Mux3 (PCmux AND Branch): Mux select signal to choose between branch target, jump target, and PC+1.
9) Branch: Signal set to one if a branch or jump Instruction is fetched.

Table 2.1: Mux1 control signal

Mux1	Destination register
00	ALU output
01	Memory output
10	Immediate value
11	Don’t care

Table 2.2: Mux2 control signal

Mux2	Second source operand
0	Second source register (Rt)
1	Immediate value

Table 2.3: ALU function

ALUfunc	ALU function
000	add
001	and
010	or
011	rotl
100	slt

Table 2.4: Branch control signal

Branch	Instruction
0	All instruction except Jump and branch
1	Branch and Jump

Table 2.5: PCmux function control signal

PCmux	PC value
00	PC+1
01	PC+ branch IMM
10	J IMM

Table 2.6: instructions specifications.

Opcode	Mnemonic	Instruction Operation	Assembly Format Code	Machine Code Format	Type
0000	**ADD**	**Rd <= Rt + Rs**	**ADD rd,rs,rt**	**0000 ssst ttdd d000 0000 0**	**R**
0100	**AND**	**Rd <= Rt AND Rs**	**AND rd,rs,rt**	**0100 ssst ttdd d000 0000 0**	**R**
1000	**OR**	**Rd <= Rt OR Rs**	**OR rd,rs,rt**	**1000 ssst ttdd d000 0000 0**	**R**
1100	**ROTL**	**Rd <= ROTL Rs {Rt}**	**ROTL rd,rs,rt**	**0100 ssst ttdd d000 0000 0**	**R**
0001	**LOAD**	**Load Rd [Rs+IMM]**	**LD rd,IMM(rs)**	**0001 ssst ttdd diii iiii i**	**I**
0010	**STORE**	**Store Rt [Rs+IMM]**	**ST rt,IMM(rs)**	**0010 ssst ttdd diii iiii i**	**I**
0011	**LOADI**	**Rd <= IMM**	**LI rd,IMM**	**0011 ssst ttdd diii iiii i**	**I**
0101	**BEQ**	**If (EQ=1) PC <= PC+IMM Else PC=PC+1**	**BEQ rt,rs,target**	**0101 ssst ttdd dbbb bbbb b**	**I**
0111	**BNE**	**If (EQ=0) PC <= PC+IMM Else PC=PC+1**	**BEQ rt,rs,target**	**0111 ssst ttdd dbbb bbbb b**	**I**
1011	**J**	**PC <= JMP IMM**	**J target**	**1011 0000 000j jjjj jjjj j**	**J**
1111	**SLT**	**IF(rs<rt) Rd<=1 Else Rd<=0**	**SLT rd,rs,rt**	**1111 ssst ttdd d000 0000 0**	**R**
				s <= Rs address t <= Rt address d <= Rd address i <= Immediate value j <= jump Immediate b <= branch Immediate	

Branch instructions will have an 8-bit immediate value. Meanwhile, jump instruction will have a 10-bit immediate value.

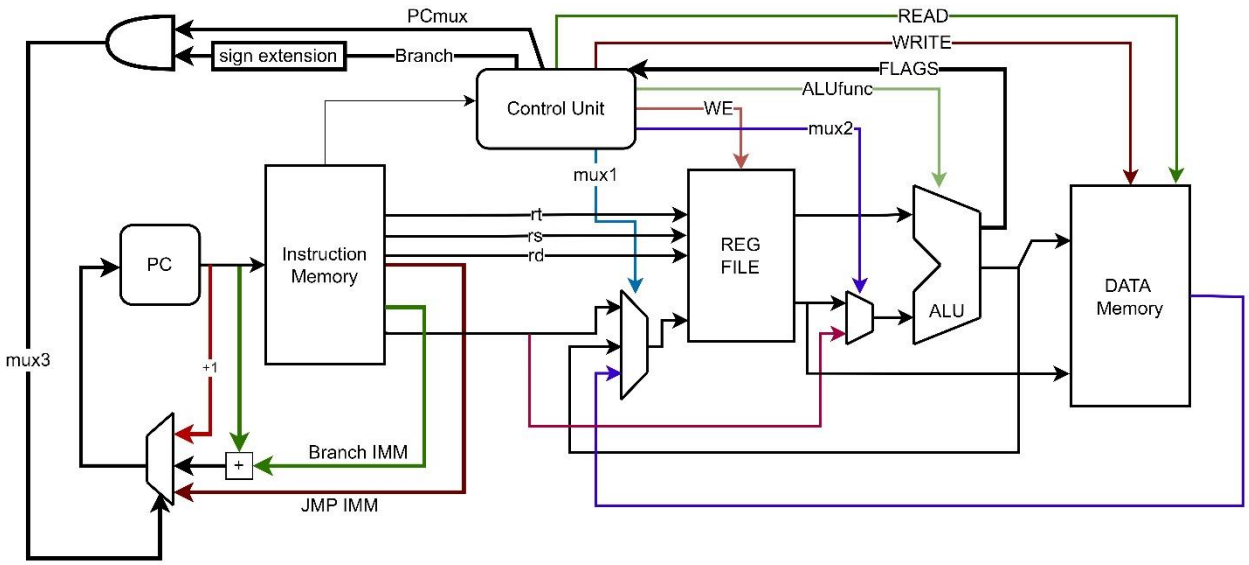

Figure 2.1: Simple CPU schematic.

Table 2.7 summarize the control unit output signals.

Table 2.7: Control unit table

Opcode [3:0]	EQ	ALUfunc [1:0]	Mux 1 [1:0]	Mux 2 [0]	WE [0]	Write [0]	Read [0]	PCmux	Branch	Instruction
0000	X	000	00	0	1	0	0	00	0	ADD
0100	X	001	00	0	1	0	0	00	0	AND
1000	X	010	00	0	1	0	0	00	0	OR
1100	X	011	00	0	1	0	0	00	0	ROTL
0001	X	000	01	1	1	0	1	00	0	LOAD
0010	X	000	XX	1	0	1	0	00	0	STORE
0011	X	000	10	X	1	0	0	00	0	LOADI
0101	0	XXX	XX	0	0	0	0	01	0	BEQ
0101	1	XXX	XX	0	0	0	0	01	1	BEQ
0111	0	XXX	XX	0	0	0	0	01	1	BNE
0111	1	XXX	XX	0	0	0	0	01	0	BNE
1011	X	XXX	XX	X	0	0	0	10	1	J
1111	X	100	00	0	1	0	0	00	0	SLT

Instruction format:

We will implement only two formats in this part: R-type and I-type. Instruction length will be 21-bit.

R-type

Opcode [3:0]	Rs [2:0]	rt[2:0]	rd[2:0]	00000000

I-type

Opcode [3:0]	Rs [2:0]	rt[2:0]	rd[2:0]	IMM[7:0]

J-type

Opcode [3:0]	0000000	J IMM[9:0]

You need to run the following assembly code on your CPU.

Test assembly code:

```
LI R3,6
LI R6,1
LI R4,12
L1:
ADD R7,R7,R6
SLT R5,R4,R7
BEQ R6,R5, L1
J END
ADD R2,R2,R6
END:
```

3 SUBMISSION INSTRUCTIONS

In this part, you need to build the schematic shown in figure 3.1. You need to submit one circuit file named as follows <Your Name>_<Your ID>.circ, e.g., ahmad_2455921.circ.

In addition to your circuit file, you need to submit a report containing a screenshot of your control unit, a Step-by-step how you design your control unit, a test file hex file, a screenshot of each component in your design, and screenshots of the register file after running the test file.

Please submit one zip file containing your circuit and your report as a PDF to the Term Project Part 3 assignment on ODTUClass.

Good Luck ☺