Computer Architecture
Milestone II

Lobna ElHawary 900160270
Marian Ramsis 900163444
Shahd El Ashmawy 900161393

I. Items removed from original 7 instruction datapath

- **Shift left 1 and 12:** no longer required due to its functionality being internally implemented in the ImmGen module provided in the support code on blackboard.
  The following 2 lines add an additional 0 bit to the end of the immediate, which is equivalent to shifting left by 1 that would have been executed by the shift left 1 module:

```
`OPCODE_Branch    :    Imm = { {20{IR[31]}}, IR[7], IR[30:25],
IR[11:8], 1'b0}
```

  Equivalently, the following are used for shifting by 12 :

```
`OPCODE_LUI       :     Imm = { IR[31], IR[30:20], IR[19:12], 12'b0 };
`OPCODE_AUIPC     :     Imm = { IR[31], IR[30:20], IR[19:12], 12'b0 };
```

- **ALU control:** removed since all control over the ALU operations will now be handled by the 4 bit ALUSelect line determined by the Control unit.
  We Justified our removal of the ALU control by noticing that the func3 of the Immediates and the R-Format instructions were essentially identical, hence the ALU control would also be the same. The only thing that would be different between them would be the second input to the ALU (whether from the ImmGen or from the register file's read data 2). Hence it seemed logical to remove the ALU control, and control the ALU operation from a 4-bit signal (ALUSelect) from the control unit.
- **ALU's Zero flag:** no longer required to determine whether operation is a branch instruction. All branching instructions (BEQ, BNE, BLT, BGT) in our design are handled by the **comparator** which will be connected to the AND gate similar to how the zero flag once was. If the branching operation satisfies conditions in the comparator, and the branch line extended from the control unit is also found to be true, only then will the AND gate return **1,** which will then determine the target to which the PC should be updated to.

II. Items Added/updated

- **Control Unit lines:**
  - **MemtoReg (2 bits):**

    Acts as a select line to the MUX that selects which output will be written back to the register file in rd.

    Its size was updated from 1 bit to 2 bits since there are now 3 inputs to the MUX (one from the **Data memory's read data,** another from the **ALU result,** and the third from the result of the **PC+4** adder.)

    Note: Refer to **Figure 4** for the truth table.

  - **ALUSelect (4 bits):**

    4 bits controlling the operations of the ALU. The updated ALU can now perform 11 operations (including the " nothing" operation, which essentially takes in only one input value (and the other is set as zero) and essentially assigns the ALU result the same output as the non-zero input.

ALUSelect	Operation
0000	AND
0001	OR
0010	ADD
0011	XOR
0100	SRL
0101	SRA
0110	SUB
0111	SLL
1000	SLT

1001	SLTU
1010	Nothing

Figure 1

- **Jalselect (1 bit):**

  Connected to the 2x1 MUX that takes in one input from the PC and the other from read data 1.

Input	Selection
0	rs1
1	PC

Figure 2

- **PCselect (1 bits):**

  Connected to the 4x1 MUX that determines the value to update the PC. Determines which value the PC is to be updated to based on the value of the PCselect as well as the result of the AND gate.

- **2x1 MUX:**
  - Receives one input from the current PC, and the other from read data 1.

    Note: Refer to **Figure 2** for truth table.
- **4x1 MUX:**
  - The MUX that determines the updated value of the PC and the MUX that determines what is to be written back to the register file were both updated to be 4x1 MUX from their previous 2x1 MUX states.

Branch	PCselect	Operation
0	0	PC+4
0	1	rs1+imm
1	0	Branch

Figure 3

MemtoReg (rd select line)	
00	R-Format, Immediates, AUIPC
01	Read data, LW
10	JAL, JALR

Figure 4

- **Comparator:**
    - Handles all branching instructions that require certain conditions to be satisfied in order to branch (BEQ, BNE, BLT, BGT)
    - The comparator receives 2 32-bit inputs (which it obtains from read data 1 and read data 2 from the register file), a 2-bit control line to determine the operation, and a 1 bit sign to determine whether to perform the signed or unsigned version of the operation (these 3 bits are obtained from instruction [14-12]/ func3, since these 3 bits are enough to differentiate between the 4 instructions. The second bit of the func3, instruction[13], determines whether the operation is signed or unsigned, while instruction[12] and instruction[14] determine the operation of the comparator).
    - The output is a 1 bit flag, which is then ANDed with the Branch signal from the Control unit to ensure that branching will only occur if the instruction is a branch instruction, otherwise the PC will not be updated with the target we want to branch to.

Control Line (2 bits)	Sign (1 bit)	Operations
00	X	BEQ
01	X	BNE

10	0	BLT
10	1	BLTU
11	0	BGT
11	1	BGTU

Figure 5

Below is the Table of Values for the Control unit for each Instruction type.

Control unit										
Instruction	Inst [6-2]	Branch	MemRead	MemtoReg	ALUOp	MemWrite	ALUSrc	RegWrite	PC_Select	Jalselect
R-Format	01100	0	0	00	10	0	0	1	0	0
LW	00000	0	1	01	00	0	1	1	0	0
SW	01000	0	0	xx	00	1	1	0	0	0
BEQ	11000	1	0	xx	01	0	0	0	0	0
JAL	11011	0	0	11	10	0	1	1	1	1
JALR	11001	0	0	11	11	0	1	1	1	1
AUIPC	00101	0	0	00	xx	0	1	1	0	0
LUI	01101	0	0	00	xx	0	1	1	0	0
Immediates	00100	0	0	00	11	0	1	1	0	0

All figures referenced in this report can also be found in the Excel sheet in the Zip file.