Course:	Computer Organization - ENCM 369
Lab #:	7
Instructor:	N. Bartley
Group	
Submission for:	B02
Submitted by:	Aarsh shah
UCID:	30150079
Partner:	William Fraser
UCID:	30158991
Date Submitted:	10-Mar-2023

Exercise A: Datapath and control signals

Exercise 1

Part I: lw t6, 20(sp)

RegWrite = 1

ImmSrc = 00_2

ALUSrc = 1

ALUControl = 000_2

MemWrite = 0

ResultSrc = 1

PCSrc = 0

5 bit

A1 = 00010

A2 = 10100

A3 = 31_{10} = 11111_2

32 bit

SrcA = 0x7fff_fca0

SrcB = 0x0000_0014

ALUResult = 0x7fff_feb4

WD3 = Result = 0x0006_6666

PC-Next = 0x0040_0134

Part II : Sub s1, s1, t5

RegWrite = 1

ALUSrc = 0

ALUControl = 001

MemWrite = 0

ResultSrc = 0

PCSrc = 0

5 bit

A1 = 01001_2

A2 = 11110_2

A3 = 01001_2

32-bit

SrcA = 0x0000_0064

SrcB = 0x ffff_fff0

PC-Next = 0x0040_0138

WD3 = ALUResult = Result = 0x0000_0074

Exercise B: Immediate-mode instructions in the singlecycle machine

Control Signals:

RegWrite: 1
ImmSrc: 00
ALUSrc: 1
MemWrite: 0
ALUControl: 010
ResultSrc: 0
PCSrc: 0

Values of R-file Inputs:

A1: 00110
A2: 00000
A3: 00110

Values of 32-bit Signals.

SrcA: Value in register given to RD1: 0x89ab_cdef
SrcB: Sign extended Imm[11:0]: FFFF_FF00
ALUResult: Bitwise AND of SrcA AND SrcB

```
  1000_1001_1010_1011_1100_1101_1110_1111
AND 1111_1111_1111_1111_1111_1111_0000_0000
-------------------------------------------
  1000_1001_1010_1011_1100_1101_0000_0000
```

= 0x89ab_cd00

Result: 0x89ab_cd00
PCNext: 0x0040_0160

Value Entering WD3:

WD3: 0x89ab_cd00

Exercise C: Support for jalr in the single-cycle machine

Completed Table:

Instruction	Opcode	RegWrite	ImmSrc	ALUSrc	TargetSrc	MemWrite	ResultSrc	Branch	ALUOp	Jump
`lw`	0000011	1	00	1	X	0	01	0	00	0
`sw`	0100011	0	01	1	X	1	xx	0	00	0
R-type	0110011	1	xx	0	X	0	00	0	10	0
`beq`	1100011	0	10	0	1	0	xx	1	01	0
I-type ALU	0010011	1	00	1	X	0	00	0	10	0
`jal`	1101111	1	11	x	1	0	10	x	xx	1
`jalr`	1100111	1	00	1	0	0	10	X	00	1

Reasonings:

RegWrite: Must be 1 if the instruction writes to a register, jalr writes "4 plus the current PC" to the GPR found with rd, so for jalr RegWrite must be 1.

ImmSrc: The offset given by bits 31:20 of the jalr instruction must be sign extended in order to be added to the GPR found with rs1, since this offset is in bits 31:20 this is like the lw instruction, so in this case for correct extension the ImmSrc must be 00.

ALUSrc: Must be 1 because the ALU must find "..the sum of the GPR found with rs1 and the 32-bit sign extension of $offset_{11:0}$". This means we want SrcB to be the sign-extended offset, so ALUSrc must be 1 for that to be the mux output.

TargetSrc: Don't cares were selected for all instruction that will only ever set PCNext to PCPlus4 (non-branching and non-jump instructions). Will be 1 for beq and jal because they, respectively, might use, and will use, the PC + ImmExt as the PCNext value. Lastly, will be 0 for jalr because it uses the ALUResult as PCNext.

MemWrite: 0 because jalr writes only to a register and does not store anything in D-mem.

ResultSrc: 10 because that is the mux input to allow result to be PCPlus4. This is necessary as PCPlus4 must be put into the rd for a jalr instruction.

Branch: Because PCSrc needs to be 1 to use PCTarget as PCNext, the equation for PCSrc, jump OR (Branch AND zero), must be 1. To ensure this, jump can be set to 1 as the OR expression will always evaluate 1. Because of this fact, Branch is a don't care value.

ALUOp: Must be 00 so that the ALUControl will be 000 (add) without relying on $funct3_{2:0}$ and $funct7_5$, as the jalr instruction does not even have funct7 bits, but definitely needs the ALU to add.

Jump: Must be 1 to ensure PCSrc selects the correct PCNext, see **Branch** for more detail.