CS2100 - L10 - Processors : Control   Week 6

10.1 - Control Unit

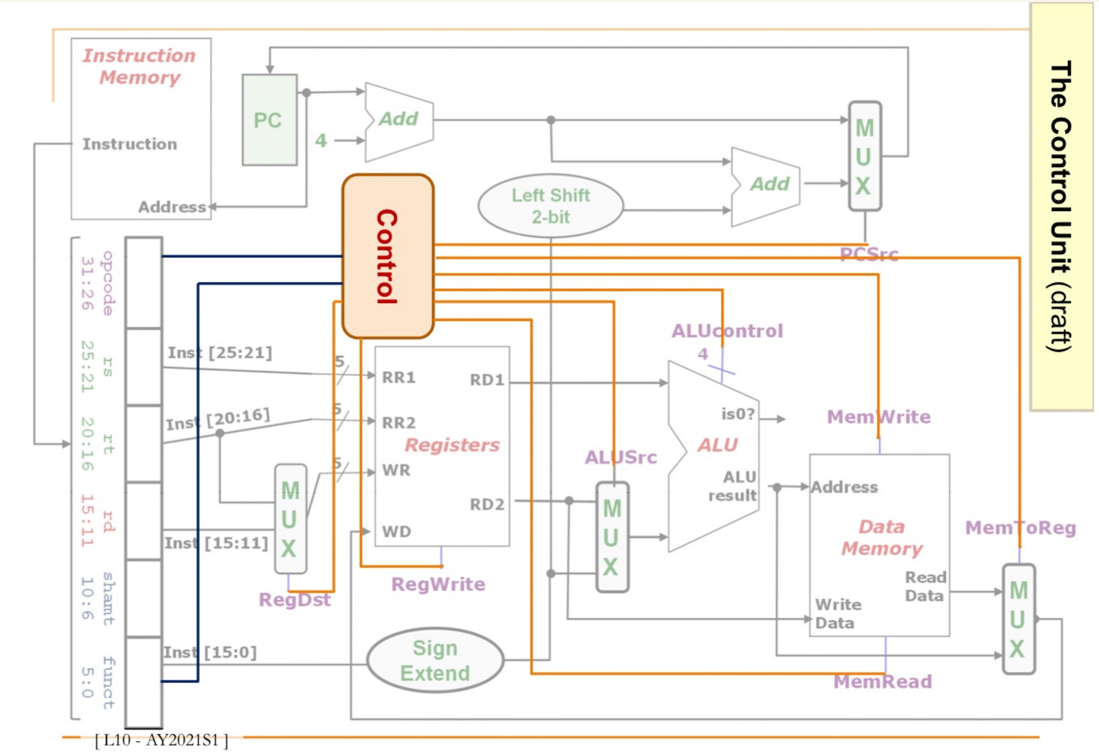
10.2 - Control Signals

- RegDst
- RegWrite
- ALUSrc
- Memread
- MemWrite
- MemToReg
- PCSrc
- ALU Control

10.3 - Summary

- Truth Tables
  - Signals
  - ALU Control

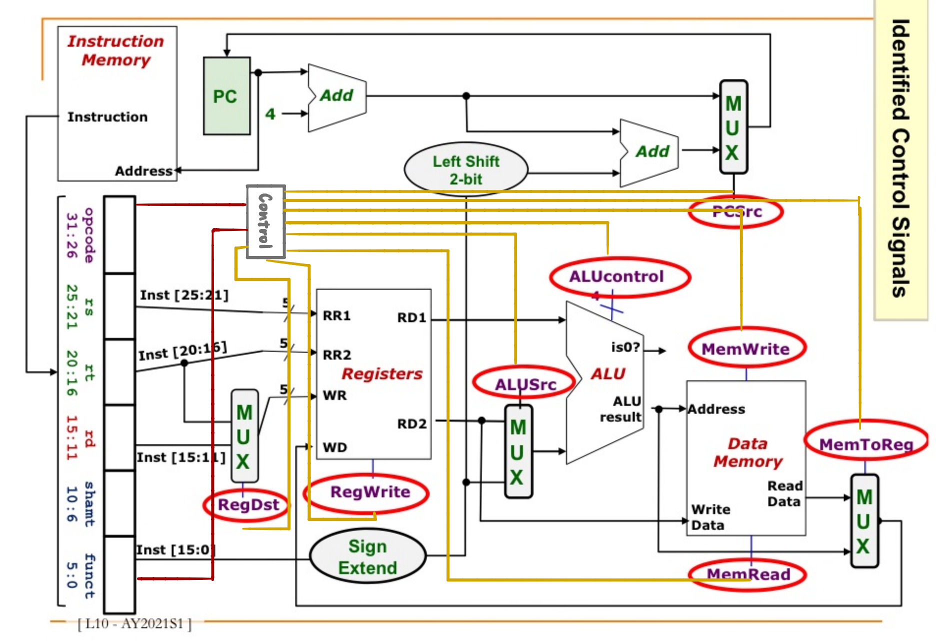
10.1 - Control Unit



The Control Unit (draft)

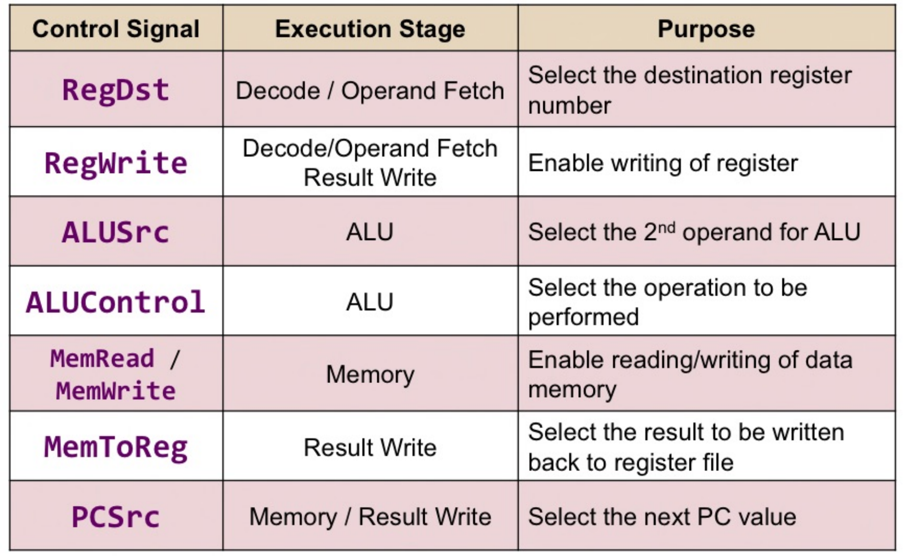
[L10 - AY2021S1]

10.2 - Control Signals



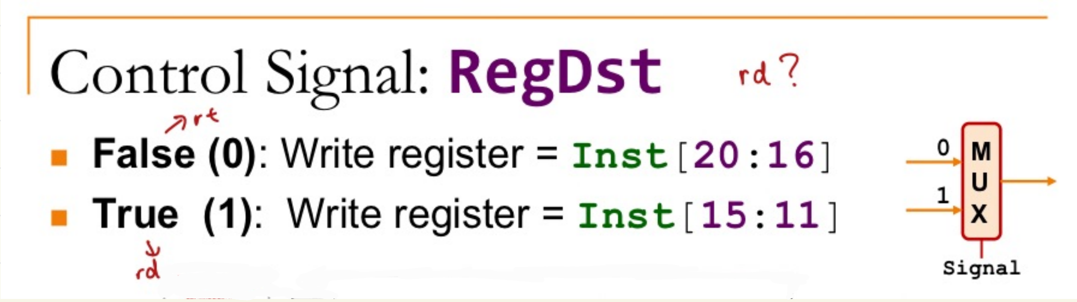
Identified Control Signals

[L10 - AY2021S1]



Control Signal	Execution Stage	Purpose
RegDst	Decode / Operand Fetch	Select the destination register number
RegWrite	Decode/Operand Fetch Result Write	Enable writing of register
ALUSrc	ALU	Select the 2nd operand for ALU
ALUControl	ALU	Select the operation to be performed
MemRead / MemWrite	Memory	Enable reading/writing of data memory
MemToReg	Result Write	Select the result to be written back to register file
PCSrc	Memory / Result Write	Select the next PC value

RegDst



Control Signal: **RegDst**

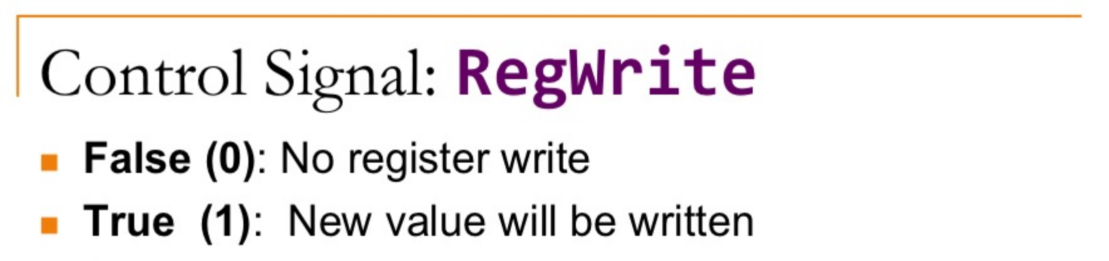
rd ?

rt

- **False (0)**: Write register = Inst[20:16]
- **True (1)**: Write register = Inst[15:11]

rd

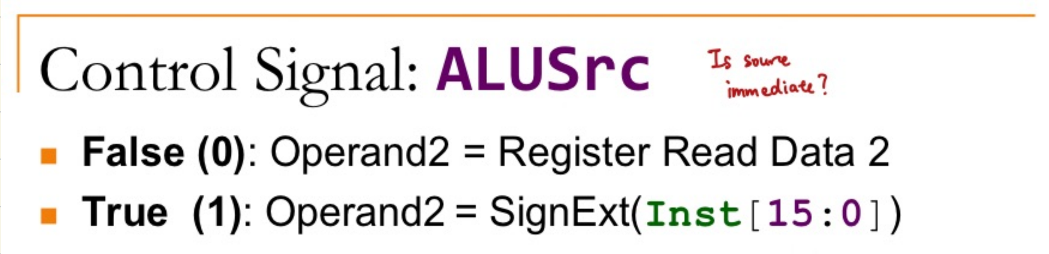
Reg Write



Control Signal: **RegWrite**

- **False (0)**: No register write
- **True (1)**: New value will be written

ALUSrc

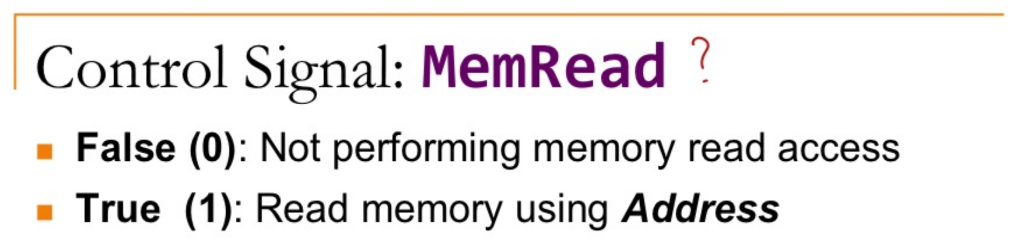


Control Signal: **ALUSrc**

Is source immediate?

- **False (0)**: Operand2 = Register Read Data 2
- **True (1)**: Operand2 = SignExt(Inst[15:0])

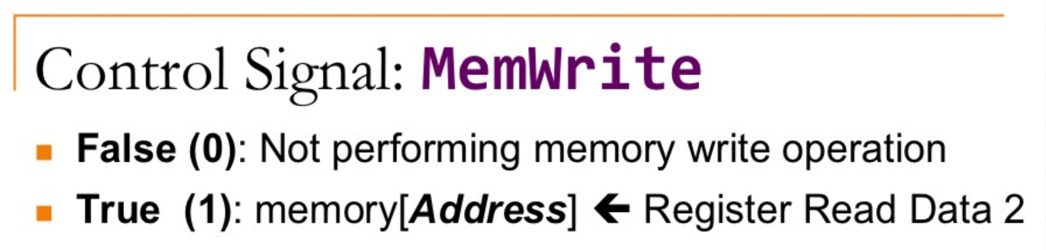
Memread



Control Signal: MemRead ?

- **False (0)**: Not performing memory read access
- **True (1)**: Read memory using ***Address***

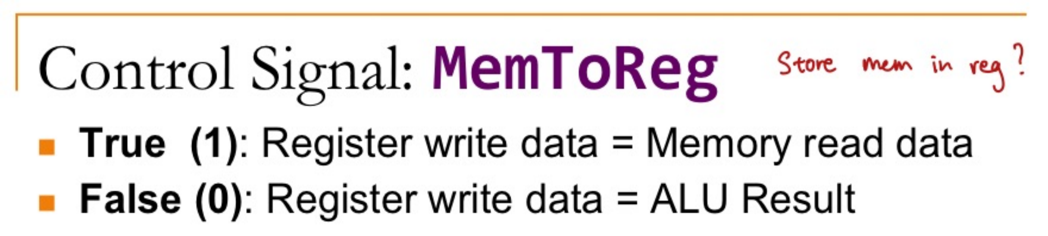
Mem Write



Control Signal: MemWrite

- **False (0)**: Not performing memory write operation
- **True (1)**: memory[***Address***] ← Register Read Data 2

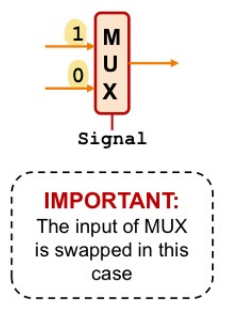
Mem To Reg



Control Signal: MemToReg

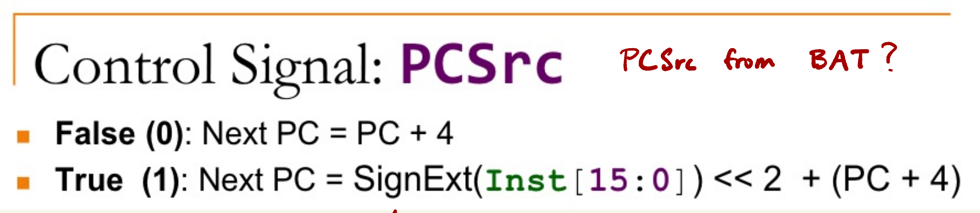
Store mem in reg?

- **True (1)**: Register write data = Memory read data
- **False (0)**: Register write data = ALU Result



IMPORTANT: The input of MUX is swapped in this case

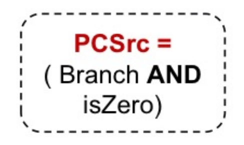
PCSrc



Control Signal: PCSrc

- **False (0)**: Next PC = PC + 4
- **True (1)**: Next PC = SignExt(Inst[15:0]) << 2 + (PC + 4)

PCSrc from BAT ?



PCSrc = (Branch **AND** isZero)

① isBranch ?

∅ : next

1 : maybe BTA

② isCondition

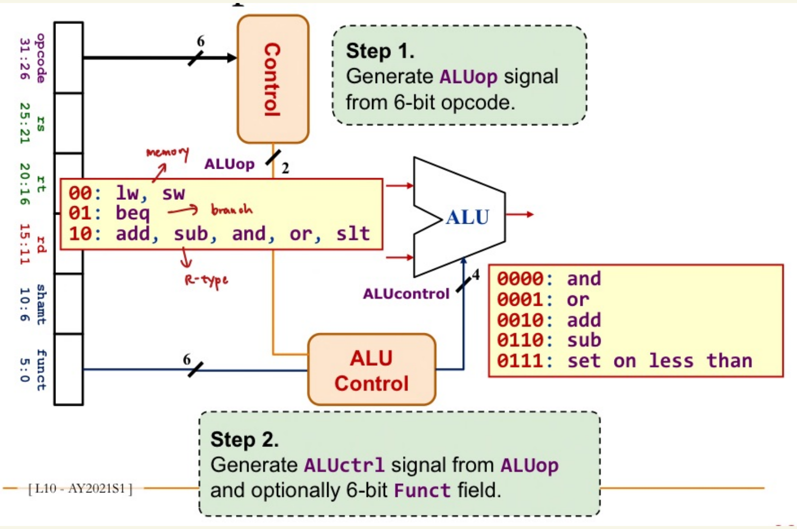
1 : taken

∅ : next

①	②	(AND) taken?
1	1	1
∅	1	∅
1	∅	∅
∅	∅	∅

ALU Control

Multilevel decoding approach : Use opcode (6b) and function code (6b) for R-type



ALUop	
00:	lw, sw
01:	beq
10:	add, sub, and, or, slt

ALUcontrol	
0000:	and
0001:	or
0010:	add
0110:	sub
0111:	set on less than

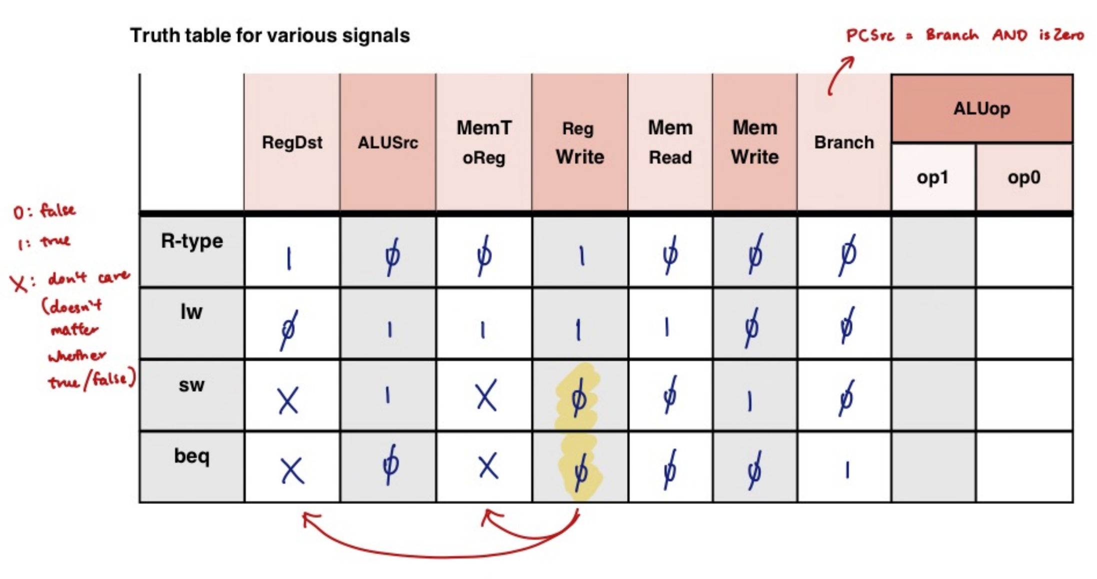
Step 1.
Generate **ALUop** signal from 6-bit opcode.

Step 2.
Generate **ALUctrl** signal from **ALUop** and optionally 6-bit **Funct** field.

[L10 - AY2021S1]

10.3 - Summary

Signals



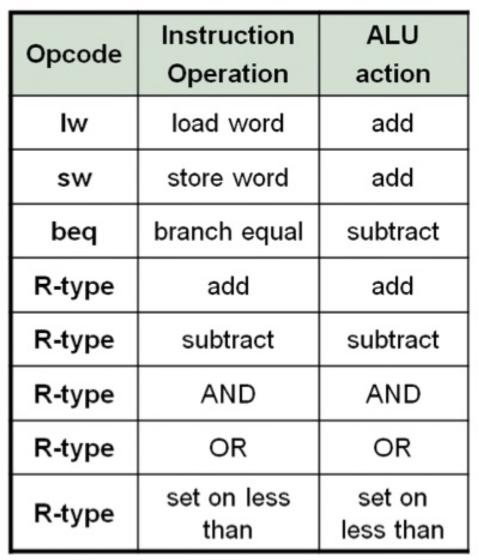
Truth table for various signals

PCSrc = Branch AND isZero

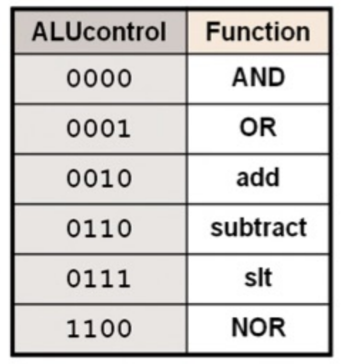
0: false
1: true
X: don't care (doesn't matter whether true/false)

	RegDst	ALUSrc	MemToReg	Reg Write	Mem Read	Mem Write	Branch	ALUop	
								op1	op0
R-type	1	ϕ	ϕ	1	ϕ	ϕ	ϕ		
lw	ϕ	1	1	1	1	ϕ	ϕ		
sw	X	1	X	ϕ	ϕ	1	ϕ		
beq	X	ϕ	X	ϕ	ϕ	ϕ	1		

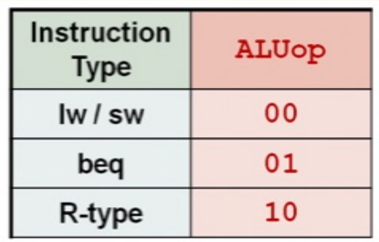
ALU Control



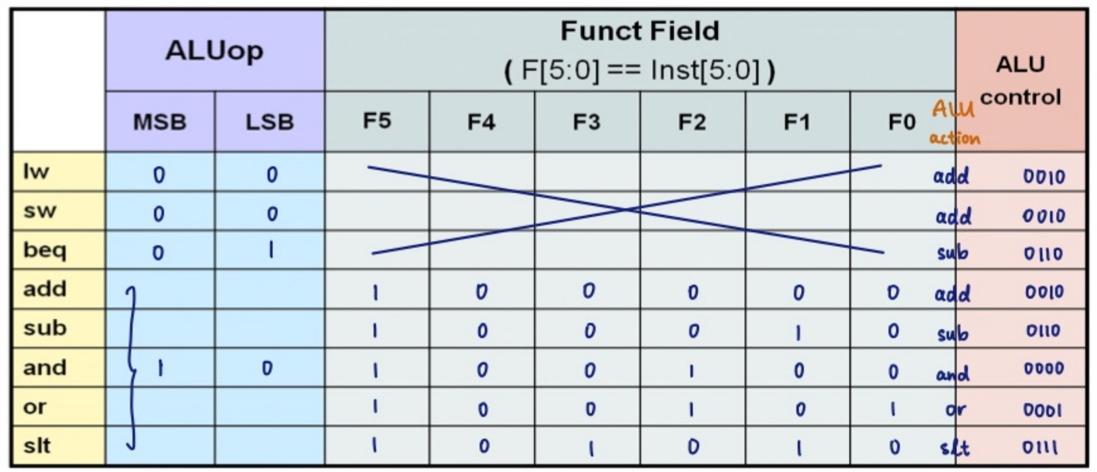
Opcode	Instruction Operation	ALU action
lw	load word	add
sw	store word	add
beq	branch equal	subtract
R-type	add	add
R-type	subtract	subtract
R-type	AND	AND
R-type	OR	OR
R-type	set on less than	set on less than



ALUcontrol	Function
0000	AND
0001	OR
0010	add
0110	subtract
0111	slt
1100	NOR



Instruction Type	ALUop
lw / sw	00
beq	01
R-type	10



	ALUop		Funct Field (F[5:0] == Inst[5:0])							ALU control
	MSB	LSB	F5	F4	F3	F2	F1	F0	ALU action	
lw	0	0							add	0010
sw	0	0							add	0010
beq	0	1							sub	0110
add	1	0	1	0	0	0	0	0	add	0010
sub			1	0	0	0	1	0	sub	0110
and			1	0	0	1	0	0	and	0000
or			1	0	0	1	0	1	or	0001
slt			1	0	1	0	1	0	slt	0111