EE 3613: Computer Organization

Chapter 4: The Processor: Datapath & Control - 4

Avinash Karanth
Department of Electrical Engineering & Computer Science
Ohio University, Athens, Ohio 45701
E-mail: karanth@ohio.edu
Website: http://oucsace.cs.ohiou.edu/~avinashk/ee461a.htm
Acknowledgement: Srinivasan Ramasubramanian, UofA, Mary J. Irwin, PSU

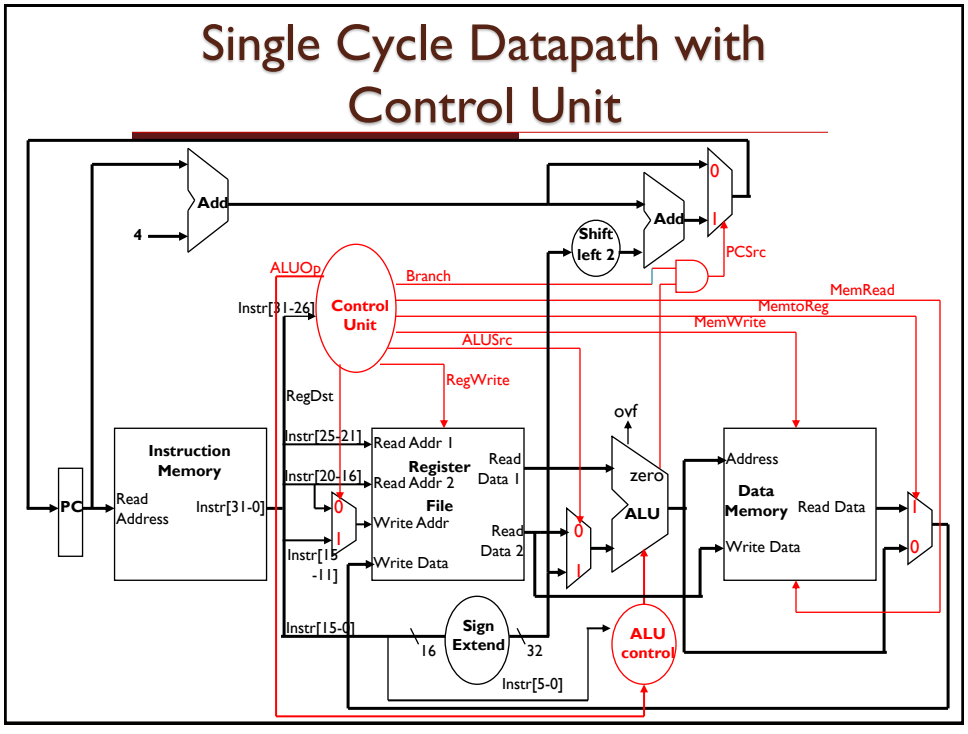
1

Course Administration

- Homework 3A is due today Fri Oct 16
- Homework 3B is due on Mon Oct 26

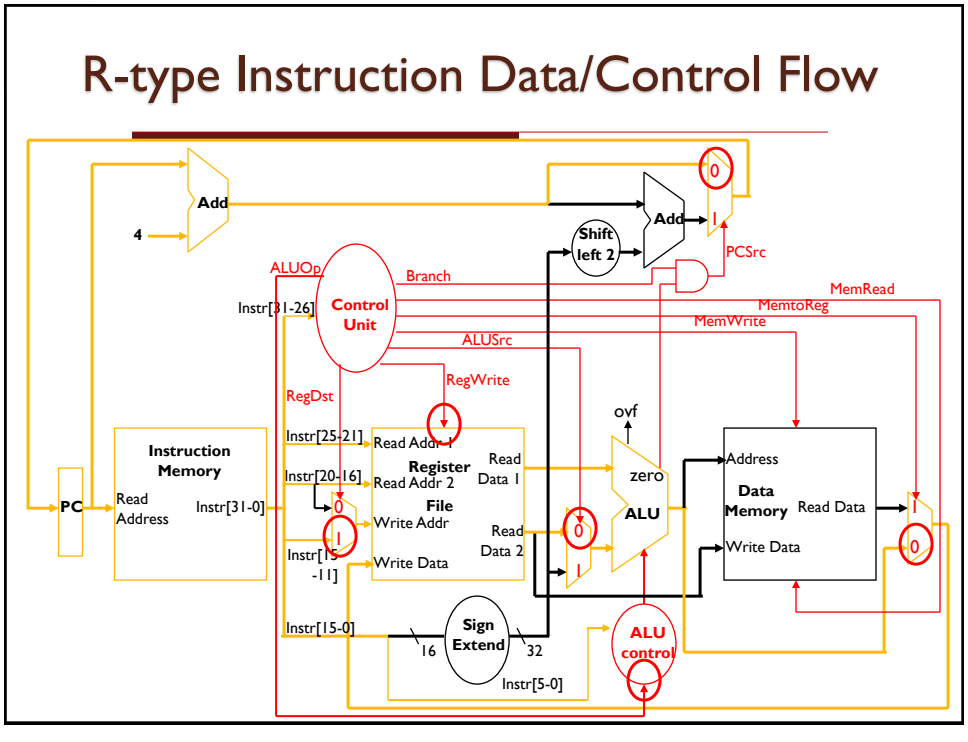
2

Single Cycle Datapath with Control Unit



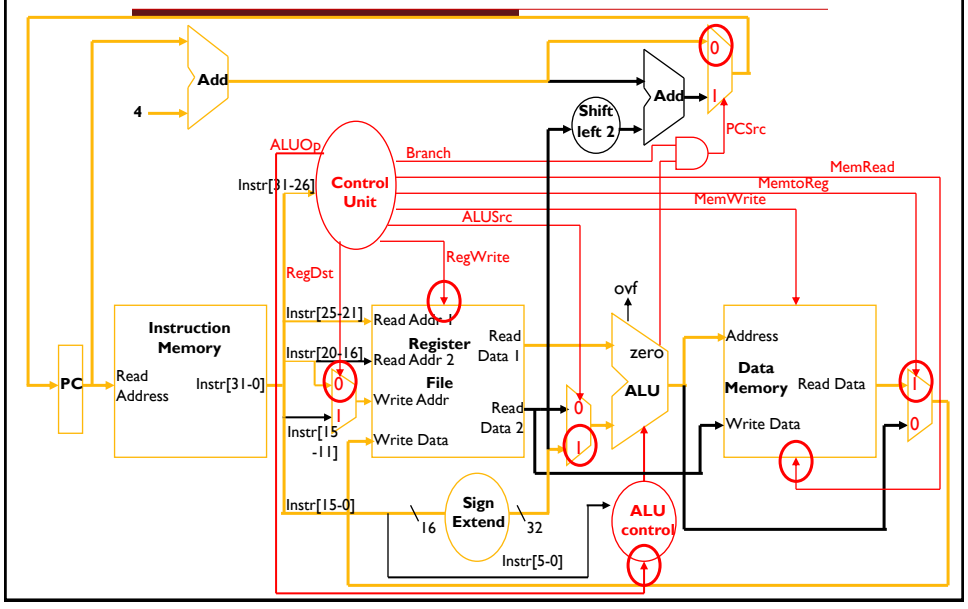
3

R-type Instruction Data/Control Flow



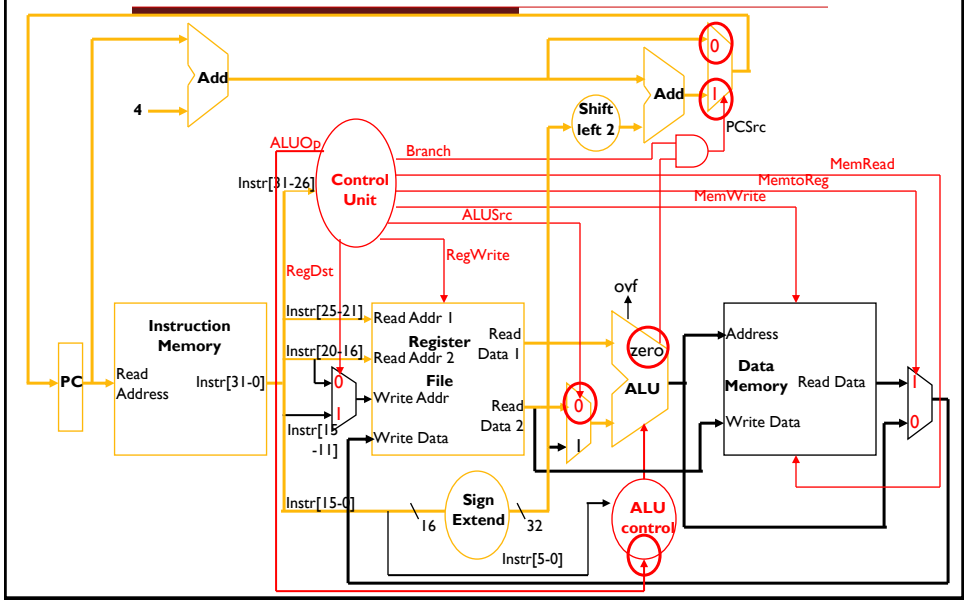
4

Load Word Instruction Data/Control Flow



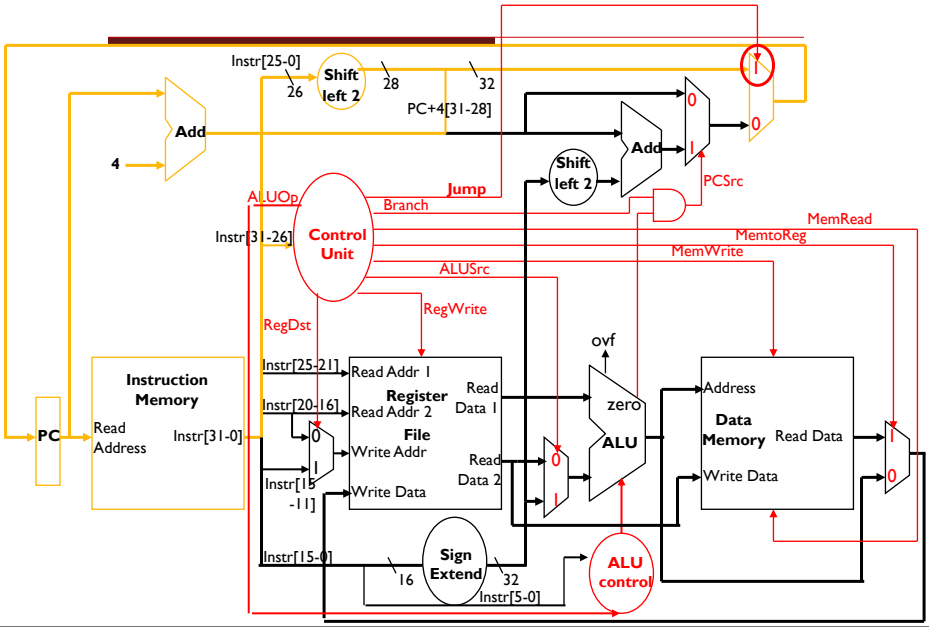
5

Branch Instruction Data/Control Flow



6

Adding the Jump Operation



7

Control Signal Values

Instruction	RegDst	ALUSrc	Memto Reg	Reg Write	Mem Read	Mem Write	Branch	ALUOp1	ALUOp2
R-Type	1	0	0	1	0	0	0	1	0
LW	0	1	1	1	1	0	0	0	0
SW	X	1	X	0	0	1	0	0	0
BEQ	X	0	X	0	0	0	1	0	1

8

ALU Control

- ALU's operation is based on instruction type and function code
  - Eg. What should the ALU do with any instruction?
- Example: `lw $t0, 32($s2)`

35	18	8	32
op	rs	rt	16-bit Offset

- ALU Control
  - 000 AND
  - 001 OR
  - 010 ADD
  - 110 SUB
  - 111 Set-if-less-than

9

Setting the ALU Control

- ALUOp
  - 00 = lw, sw; 01 = beq; 10 = arithmetic; 11 = Jump

Instruction	ALUOp	Instruction Operation	Function Field	Desired ALU function	ALU Control
LW	00	Load word	xxxxxx	ADD	010
SW	00	Store word	xxxxxx	ADD	010
BEQ	01	Branch if Equal	xxxxxx	SUB	110
R – Type	10	ADD	100000	ADD	010
R – Type	10	SUB	100010	SUB	110
R – Type	10	AND	100100	AND	000
R – Type	10	OR	100101	OR	001
R – Type	10	Set if less than	101010	Set if less than	111

10

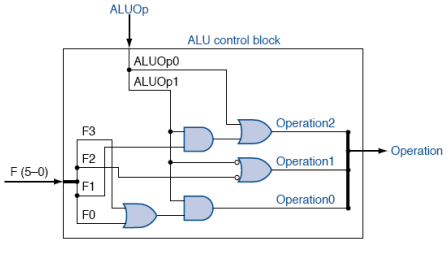
Truth Table for ALU Control

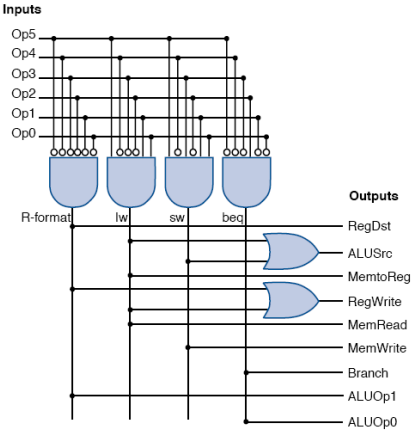
ALUOp		Function Field						ALU Control
ALUOp1	ALUOp2	F5	F4	F3	F2	F1	F0	
0	0	X	X	X	X	X	X	010
0	0	X	X	X	X	X	X	010
0	1	X	X	X	X	X	X	110
1	0	1	0	0	0	0	0	010
1	0	1	0	0	0	1	0	110
1	0	1	0	0	1	0	0	000
1	0	1	0	0	1	0	1	001
1	0	1	0	1	0	1	0	111

11

Implementation of Control

Simple combinational logic to realize the truth table





12

Functional Units used by an Instruction Class

Instruction	Functional Units used by the instruction class				
R-type	Instruction Fetch	Register Access	ALU	Register Access	
LW	Instruction Fetch	Register Access	ALU	Register Access	Register Access
SW	Instruction Fetch	Register Access	ALU	Register Access	
BEQ	Instruction Fetch	Register Access	ALU	0	0
Jump	Instruction Fetch				

13

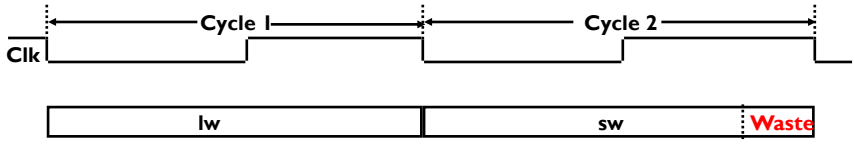
Timing for single cycle implementation (psec)

Instruction Class	Instruction Memory	Register Read	ALU Operation	Data Memory	Register Write
R-type	200	50	100		50
LW	200	50	100	200	50
SW	200	50	100	200	
BEQ	200	50	100		
Jump	200				

14

Single Cycle Advantages & Disadvantages

- Uses the clock cycle efficiently – the clock cycle must be timed to accommodate the slowest instruction



- Wasting area as some functional units (Eg. Adders) must be duplicated since they can not be shared during a clock cycle, however,
- Is simple and easy to understand

15

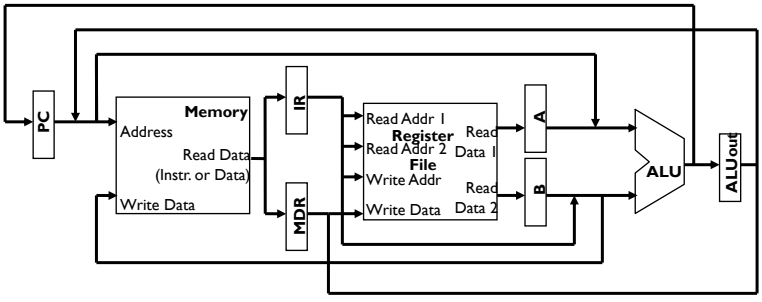
Multicycle Datapath Approach

- Let an instruction take more than one clock cycle to complete
  - Break up instructions into steps where each step takes a cycle while trying to
    - balance the amount of work to be done in each step
    - restrict each cycle to use only one major functional unit
  - Not every instruction takes the same number of clock cycles
- In addition to faster clock rates, multi-cycle allows functional units that can be used more than once per instruction as long as they are used on different clock cycles; as a result
  - need only one memory – but only one memory access per cycle
  - need only one ALU/adder – but only one ALU operation per cycle

16

Multicycle Datapath Approach, con't

- At the end of a cycle
  - Store values needed in a later cycle by the current instruction in an internal register (not visible to the programmer). All (except IR) hold data only between a pair of adjacent clock cycles (no write control signal needed)



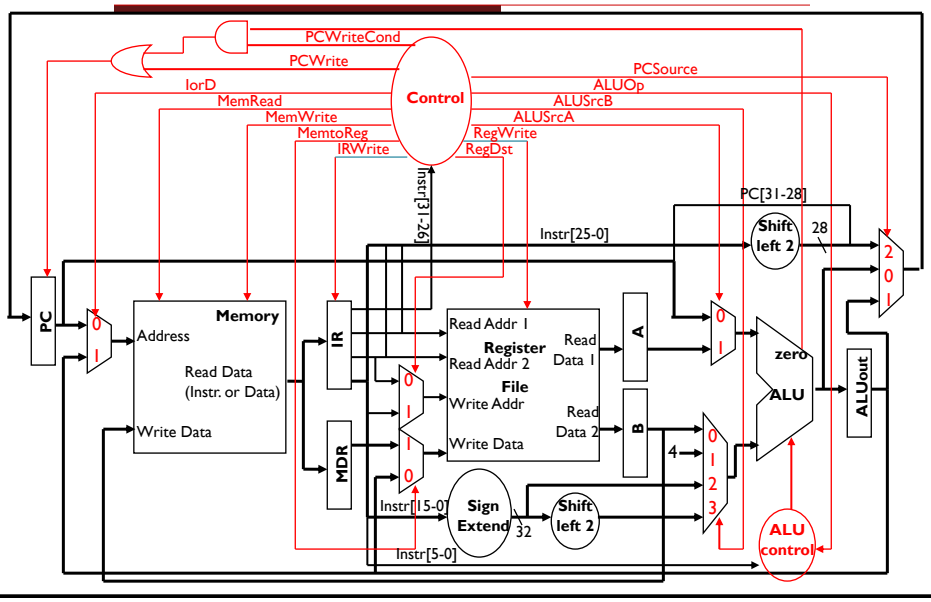
IR – Instruction Register **MDR** – Memory Data Register

A, B – regfile read data registers **ALUout** – ALU output register

- Data used by subsequent instructions are stored in programmer visible registers (i.e., register file, PC, or memory)

17

The Multicycle Datapath with Control Signals



18

Five Execution Steps

- Instruction Fetch (IF)
- Instruction Decode and Register Fetch (ID/IR)
- Execution, Memory Address Computation or Branch Computation (EX)
- Memory Access or R-type instruction completion (MEM)
- Write-back step (WB)

Instructions take 3 – 5 cycles !

19

Step 1: Instruction Fetch

- Use PC to get instruction and put it in the Instruction Register
- Increment PC by 4 and put the result back in the PC
- Can be described succinctly using the RTL
  - IR = Memory[PC];
  - PC = PC + 4;
- Can we figure out the values of the control signals?
- What is the advantage of updating the PC now?

20

Step 2: Instruction Decode and Register Fetch

- Read registers rs and rt in case we need them
- Compute the branch address in case the instruction is a branch
- RTL
  - A = Reg[IR[25-21];
  - B = Reg[IR[20-16];
  - ALUout = PC + (sign-extend(IR[15-0]) << 2);
- We aren't setting any control lines based on the instruction type
  - The instruction is still being decoded in the control logic

21

Step 3: Execution

- ALU is performing one of three functions, based on instruction type
  - **Memory Reference**: ALUout = A + sign-extend(IR[15-0]);
  - **R-type**: ALUout = A op B;
  - **Branch**: if (A == B) PC = ALUout;
  - **Jump**: PC = PC[31-28] | (IR[25:0] << 2);

22

Step 4: R-type of Memory Access

- Load and stores access memory
  - MDR = Memory[ALUout];
  - Memory[ALUout] = B;
- R-type instructions finish
  - Reg[IR[15-11]] = ALUout
- The write takes place at the end of the cycle on the edge

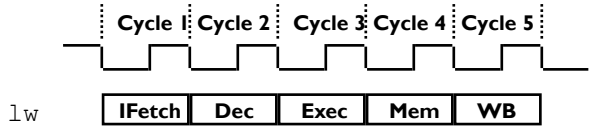
23

Step 5: Write-Back

- Reg[IR[20-16]] = MDR;
- What about other instructions?

24

The Five Steps of the Load Instruction

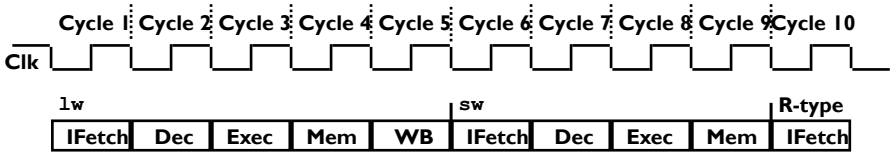


- IFetch: Instruction Fetch and Update PC
- Dec: Instruction Decode, Register Read, Sign Extend Offset
- Exec: Execute R-type; Calculate Memory Address; Branch Comparison; Branch and Jump Completion
- Mem: Memory Read; Memory Write Completion; R-type Completion (RegFile write)
- WB: Memory Read Completion (RegFile write)

25

Multicycle Advantages & Disadvantages

- Uses the clock cycle efficiently – the clock cycle is timed to accommodate the slowest instruction step



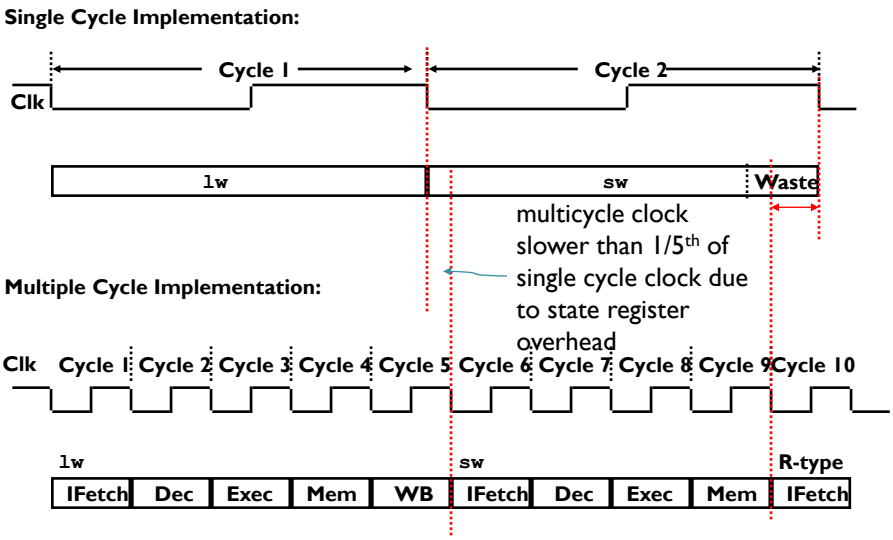
- Multicycle implementations allow functional units to be used more than once per instruction as long as they are used on different clock cycles

but

- Requires additional internal state registers, more muxes, and more complicated (FSM) control

26

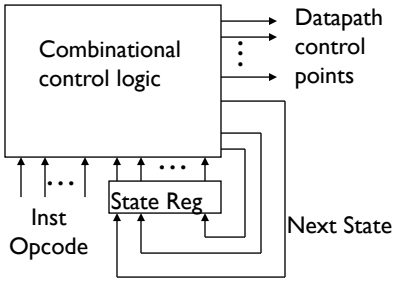
Single Cycle vs. Multiple Cycle Timing



27

Multicycle Control Unit

- Multicycle datapath control signals are not determined solely by the bits in the instruction
  - e.g., op code bits tell what operation the ALU should be doing, but *not* what instruction cycle is to be done next
- Must use a finite state machine (FSM) for control
  - a set of states (current state stored in State Register)
  - next state function (determined by current state and the input)
  - output function (determined by current state and the input)



28