

CSE 141 – Computer Architecture Spring 2005

Lecture 4 ALU Part 1 Pramod V. Argade

CSE141: Introduction to Computer Architecture

Instructor: Pramod V. Argade (p2argade@cs.ucsd.edu)
Office Hour:
Mon. 5:00 - 6:00 PM (AP&M 5218)
Wed. 5:00 - 6:00 PM (AP&M 2444)

TAs:
Baris Arslan: barslan@cs.ucsd.edu
Chris Roedel: croedel@cs.ucsd.edu
Raid Ayoub: rayoub@cs.ucsd.edu
Leo Porter: leporter@cs.ucsd.edu

Textbook: Computer Organization & Design
The Hardware Software Interface, 3rd Edition.
Authors: Patterson and Hennessy

Web-page: <http://www.cse.ucsd.edu/classes/sp05/cse141>

Announcements

- **Discussions Sections for 141:**

Fridays, 10:00 - 10:50 am, Peterson Hall 104 (Chris)
Fridays, 2:00 - 2:50 pm, Peterson Hall 104 (Leo)

- **Reading Assignment**

– Chapter 2. Instructions: Language of the Computer
Sec. 2.1 - 2.9, 2-16
– Chapter 3. Arithmetic for Computers
Sections 3.1 - 3.3, B5

- **Homework 2: Due Mon., April 11 in class**

2.2, 2.4, 2.6, 2.29, 2.30, 2.32, 2.47
3.3, 3.5, 3.7, 3.9, 3.10, 3.12. From Edition 2: 4.23 (to be posted on the web)

- **Quiz**

When: Mon., April 11th, First 10 minutes of the class

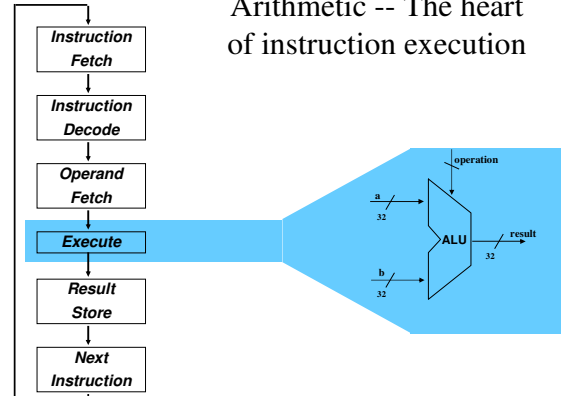
Topic: ISA, Chapter 2 **Need:** Paper, pen

Course Schedule

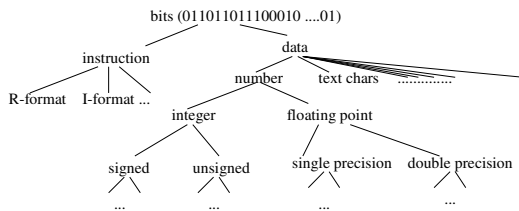
Lecture #	Date	Day	Lecture Topic	Quiz Topic	Homework Due
1	3/28	Monday	Introduction, Ch. 1	-	-
2	3/30	Wednesday	Performance, Ch. 4	-	-
3	4/4	Monday	ISA, Ch. 2	Performance	#1
4	4/6	Wednesday	Arithmetic, Ch. 3	-	-
5	4/11	Monday	Arithmetic, Ch. 3	ISA	#2
6	4/13	Wednesday	Single cycle CPU, Ch. 5	-	-
7	4/18	Monday	Single cycle CPU, Ch. 5	Arithmetic	#3
8	4/20	Wednesday	Multi-cycle CPU, Ch. 5	-	-
9	4/25	Monday	Multi-cycle CPU, Ch. 5	Single Cycle CPU	#4
10	4/27	Wednesday	Review for the Midterm	-	-
	5/2	Monday	Mid-term Exam	-	-
11	5/4	Wednesday	Exceptions, Ch. 5 and Pipelining, Ch. 6	-	-
12	5/9	Monday	Pipelining, Ch. 6	-	-
13	5/11	Wednesday	Data and control hazards, Ch. 6	-	-
14	5/16	Monday	Data and control hazards, Ch. 6	Pipeline Hazards	#5
15	5/18	Wednesday	Memory & cache design, Ch. 7	-	-
16	5/23	Monday	Memory & cache design, Ch. 7	Cache	#6
17	5/25	Wednesday	Virtual Memory & cache design, Ch. 7	-	-
No Class	5/30	Monday	Memorial Day Holiday	-	-
18	6/1	Wednesday	Course Review	-	-
	6/10	Friday	Final Exam	-	-

Computer Arithmetic and ALU Design

Arithmetic -- The heart of instruction execution



Bits everywhere... What do they mean?



Questions About Numbers and ALU

- How do you represent
 - signed and unsigned numbers
 - negative numbers?
 - fractions?
 - really large numbers?
 - really small numbers?
- How do you
 - do arithmetic?
 - identify errors (e.g. overflow)?
- What is an ALU and what does it look like?
 - ALU=arithmetic logic unit

Introduction to Binary Numbers

Consider a 4-bit binary number

Decimal	Binary	Decimal	Binary
0	0000	4	0100
1	0001	5	0101
2	0010	6	0110
3	0011	7	0111

Examples of binary arithmetic:

$$\begin{array}{r}
 3 + 2 = 5 \\
 \begin{array}{r}
 0011 \\
 + 0010 \\
 \hline
 0101
 \end{array}
 \end{array}
 \qquad
 \begin{array}{r}
 3 + 3 = 6 \\
 \begin{array}{r}
 0011 \\
 + 0011 \\
 \hline
 0110
 \end{array}
 \end{array}$$

Negative Numbers?

- We would like a number system that provides
 - obvious representation of positive and negative integers
 - uses the same adder for addition and subtraction
 - single value of 0
 - equal coverage of positive and negative numbers
 - easy detection of sign
 - easy negation

Possible Representations

- Sign Magnitude:

Sign Magnitude	One's Complement	Two's Complement
000 = +0	000 = +0	000 = +0
001 = +1	001 = +1	001 = +1
010 = +2	010 = +2	010 = +2
011 = +3	011 = +3	011 = +3
100 = -0	100 = -3	100 = -4
101 = -1	101 = -2	101 = -3
110 = -2	110 = -1	110 = -2
111 = -3	111 = -0	111 = -1
- 2's complement representation of negative numbers
 - Take the bitwise inverse and add 1
- Issues: balance, number of zeros, ease for HW/SW
- Which one is best? Why?

Two's Complement Arithmetic

Decimal	2's Complement Binary	Decimal	2's Complement Binary
0	0000	-1	1111
1	0001	-2	1110
2	0010	-3	1101
3	0011	-4	1100
4	0100	-5	1011
5	0101	-6	1010
6	0110	-7	1001
7	0111	-8	1000

- Examples: $7 - 6 = 7 + (-6) = 1$ $3 - 5 = 3 + (-5) = -2$

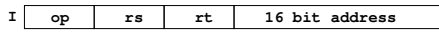
$$\begin{array}{r}
 7 - 6 = 7 + (-6) = 1 \\
 \begin{array}{r}
 0111 \\
 + 1010 \\
 \hline
 0001
 \end{array}
 \end{array}
 \qquad
 \begin{array}{r}
 3 - 5 = 3 + (-5) = -2 \\
 \begin{array}{r}
 0011 \\
 + 1011 \\
 \hline
 1110
 \end{array}
 \end{array}$$

Immediate Field in I-Format

- I-Format Instruction Example:

```
addi $s3, $t0, 4
```

6 bits 5 bits 5 bits



- For LW, SW, BEQ, BNE, ADDI

- 16-bit Immediate field is signed
 - > Copy sign bit to all upper 16 bits
- It is sign extended to 32 bits before use
 - > Sign extension:
 - ◊ 0010 in 4 bits is the same as 0000 0010 in 8 bits (+2)
 - ◊ 1110 in 4 bits is the same as 1111 1110 in 8 bits (-2)
- Why is there no SUBI instruction for MIPS?

- For ANDI, ORI

- 16-bit Immediate field is zero-extended to 32 bits before use
 - > Copy zero to all upper 16 bits
- Why?

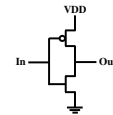
Foundation of ICs: Field Effect Transistor (FET)

- N-FET: Conducts when Gate is high
- P-FET: Conducts when Gate is low



- CMOS Inverter

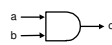
- Input high, output low
- Input low, output high
- Power consumed only during transition



- Current processing technology 0.09 μ, next 0.065 μ

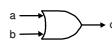
Some basics of digital logic

1. AND gate ($c = a \cdot b$)



a	b	$c = a \cdot b$
0	0	0
0	1	0
1	0	0
1	1	1

2. OR gate ($c = a + b$)



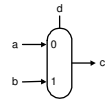
a	b	$c = a + b$
0	0	0
0	1	1
1	0	1
1	1	1

3. Inverter ($c = \bar{a}$)



a	$c = \bar{a}$
0	1
1	0

4. Multiplexor (if $d = 0$, $c = a$; else $c = b$)

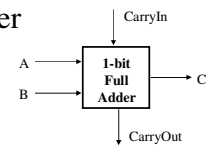


d	c
0	a
1	b

A One-bit Full Adder

- This is also called a (3, 2) adder

- Truth Table:



Inputs			Outputs		Comments
A	B	CarryIn	CarryOut	Sum	
0	0	0	0	0	$0 + 0 + 0 = 00$
0	0	1	0	1	$0 + 0 + 1 = 01$
0	1	0	0	1	$0 + 1 + 0 = 01$
0	1	1	1	0	$0 + 1 + 1 = 10$
1	0	0	0	1	$1 + 0 + 0 = 01$
1	0	1	1	0	$1 + 0 + 1 = 10$
1	1	0	1	0	$1 + 1 + 0 = 10$
1	1	1	1	1	$1 + 1 + 1 = 11$

Logic Equation for CarryOut

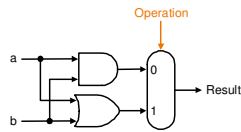
Inputs			Outputs		Comments
A	B	CarryIn	CarryOut	Sum	
0	0	0	0	0	0 + 0 + 0 = 00
0	0	1	0	1	0 + 0 + 1 = 01
0	1	0	0	1	0 + 1 + 0 = 01
0	1	1	1	0	0 + 1 + 1 = 10
1	0	0	0	1	1 + 0 + 0 = 01
1	0	1	1	0	1 + 0 + 1 = 10
1	1	0	1	0	1 + 1 + 0 = 10
1	1	1	1	1	1 + 1 + 1 = 11

Logic Equation for Sum

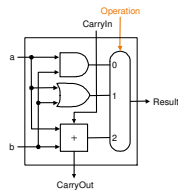
Inputs			Outputs		Comments
A	B	CarryIn	CarryOut	Sum	
0	0	0	0	0	0 + 0 + 0 = 00
0	0	1	0	1	0 + 0 + 1 = 01
0	1	0	0	1	0 + 1 + 0 = 01
0	1	1	1	0	0 + 1 + 1 = 10
1	0	0	0	1	1 + 0 + 0 = 01
1	0	1	1	0	1 + 0 + 1 = 10
1	1	0	1	0	1 + 1 + 0 = 10
1	1	1	1	1	1 + 1 + 1 = 11

1-bit ALU

- **ALU Control Lines (ALUop)** Function
 - 000 And
 - 001 Or

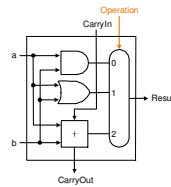


- **ALU Control Lines (ALUop)** Function
 - 000 And
 - 001 Or
 - 010 Add



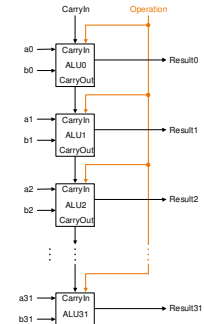
But how do we make the adder?

1-bit ALU



- **Implements functions:**
 - AND
 - OR
 - ADD
- **What about SUB and SLT?**

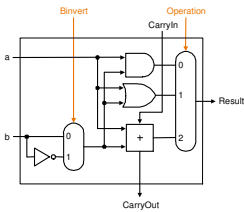
32-bit ALU



The 32-bit ALU

32-bit ALU: Subtraction

- Keep in mind the following:
 - $(A - B)$ is the same as: $A + (-B)$
- Bit-wise inverse of B is !B:
 - $A - B = A + (-B) = A + (!B + 1) = A + !B + 1$

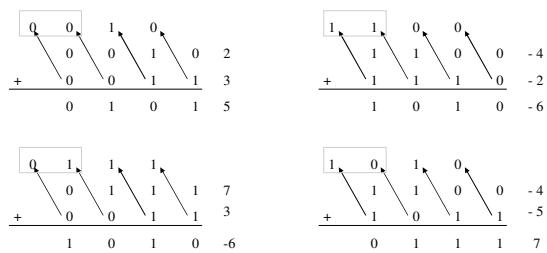


- Binvert provides the negation
- How about "+1"?
- For SUB
 - set Binvert = 1
 - CarryIn = 1 for LSB

Detecting Overflow

- No overflow when adding a positive and a negative number
- No overflow when signs are the same for subtraction
- Overflow occurs when the value affects the sign:
 - overflow when adding two positives yields a negative
 - or, adding two negatives gives a positive
 - or, subtract a negative from a positive and get a negative
 - or, subtract a positive from a negative and get a positive
- Consider the operations $A + B$, and $A - B$
 - Can overflow occur if B is 0 ?
 - Can overflow occur if A is 0 ?
- Response of MIPS to overflow will be covered later in the course

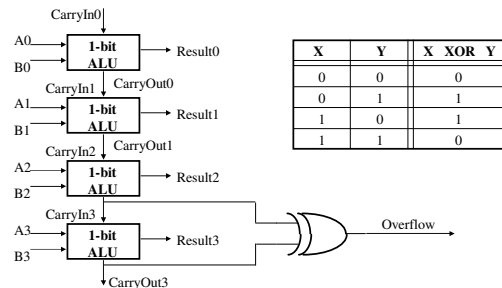
Overflow Detection



So how do we detect overflow?

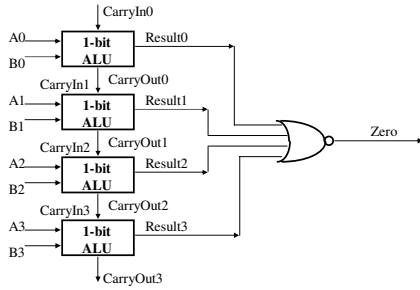
Overflow Detection Logic

- Carry into MSB \neq Carry out of MSB
 - For a N-bit ALU: $\text{Overflow} = \text{CarryIn}[N - 1] \text{ XOR } \text{CarryOut}[N - 1]$



BEQ/BNE: Zero Detection Logic

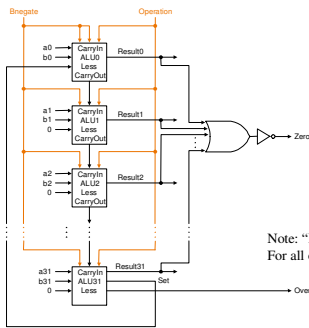
- Zero Detection Logic is just one BIG NOR gate
 - Any non-zero input to the NOR gate will cause its output to be zero



SLT: Set-on-less-than Logic

- SLT \$1, \$2, \$3
 - if(\$2 < \$3)
 - \$1 = 1;
 - else \$1 = 0;
- To test A < B, do a subtraction (A - B)
 - (A < B) if (A - B) < 0, i.e. negative
- Use sign bit
 - Route the sign bit to bit 0 of result
 - Set bits 1 - 31 to zero
- There is a complication due to overflow
 - Work out solution in Homework problem 4.23

A Complete 32-bit ALU

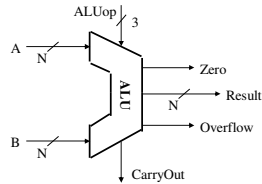


Functionality

- Arithmetic Operations:
 - ADD, SUB
- Logical Operations:
 - AND, OR
- Compare
 - SLT
- Support for branch
 - BEQ, BNE
- Exception detection
 - Overflow

Note: "Less" is connected to "Set" input for bit 0. For all other bits, less is connected to zero.

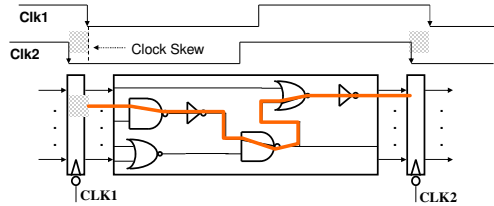
Designing an Arithmetic Logic Unit



- | ALU Control Lines (ALUop) | Function |
|---------------------------|------------------|
| 000 | And |
| 001 | Or |
| 010 | Add |
| 110 | Subtract |
| 111 | Set-on-less-than |

Our ALU has functionality but lacks speed...

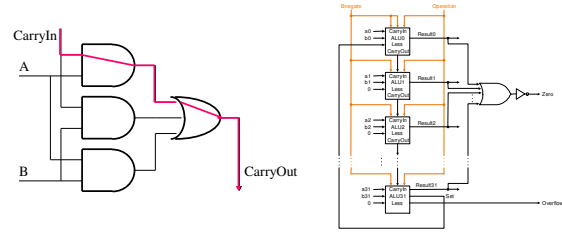
$$\text{CPU time} = \frac{\text{Seconds}}{\text{Program}} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Cycle}}$$



• Cycle Time = CLK-to-Q + Longest Gate Delay + Setup Time + Clock Skew

Adder in our ALU is in timing critical path

- The adder we just built is called a "Ripple Carry Adder"
 - The carry bit may have to propagate from LSB to MSB
 - Worst case delay for an N-bit RC adder: 2N-gate delay



Conclusion

- We can build an ALU to support the MIPS instruction set
 - Key idea: use multiplexor to select the output we want
 - Efficiently perform subtraction using two's complement
 - Replicate a 1-bit ALU to produce a 32-bit ALU
- Important points about hardware
 - All of the gates are always working
 - The speed of a gate is affected by the number of inputs to the gate
 - The speed of a circuit is affected by the number of gates in series (on the "critical path" or the "deepest level of logic")
 - For computer hardware, "Speed is it!"

Announcements

- **Discussions Sections for 141:**
 - Fridays, 10:00 - 10:50 am, Peterson Hall 104 (Chris)
 - Fridays, 2:00 - 2:50 pm, Peterson Hall 104 (Leo)
- **Reading Assignment**
 - Chapter 2. Instructions: Language of the Computer
 - Sec. 2.1 - 2.9, 2-16
 - Chapter 3. Arithmetic for Computers
 - Sections 3.1 - 3.3, B5, B6
- **Homework 2: Due Mon., April 11 in class**
 - 2.2, 2.4, 2.6, 2.29, 2.30, 2.32, 2.47
 - 3.3, 3.5, 3.7, 3.9, 3.10, 3.12. From Edition 2: 4.23 (to be posted on the web)
- **Quiz**
 - When:** Mon., April 11th, First 10 minutes of the class
 - Topic:** ISA, Chapter 2 **Need:** Paper, pen