CprE 381: Computer Organization and Assembly Level Programming

MIPS Arithmetic

Henry Duwe
Electrical and Computer Engineering
Iowa State University

Administrative

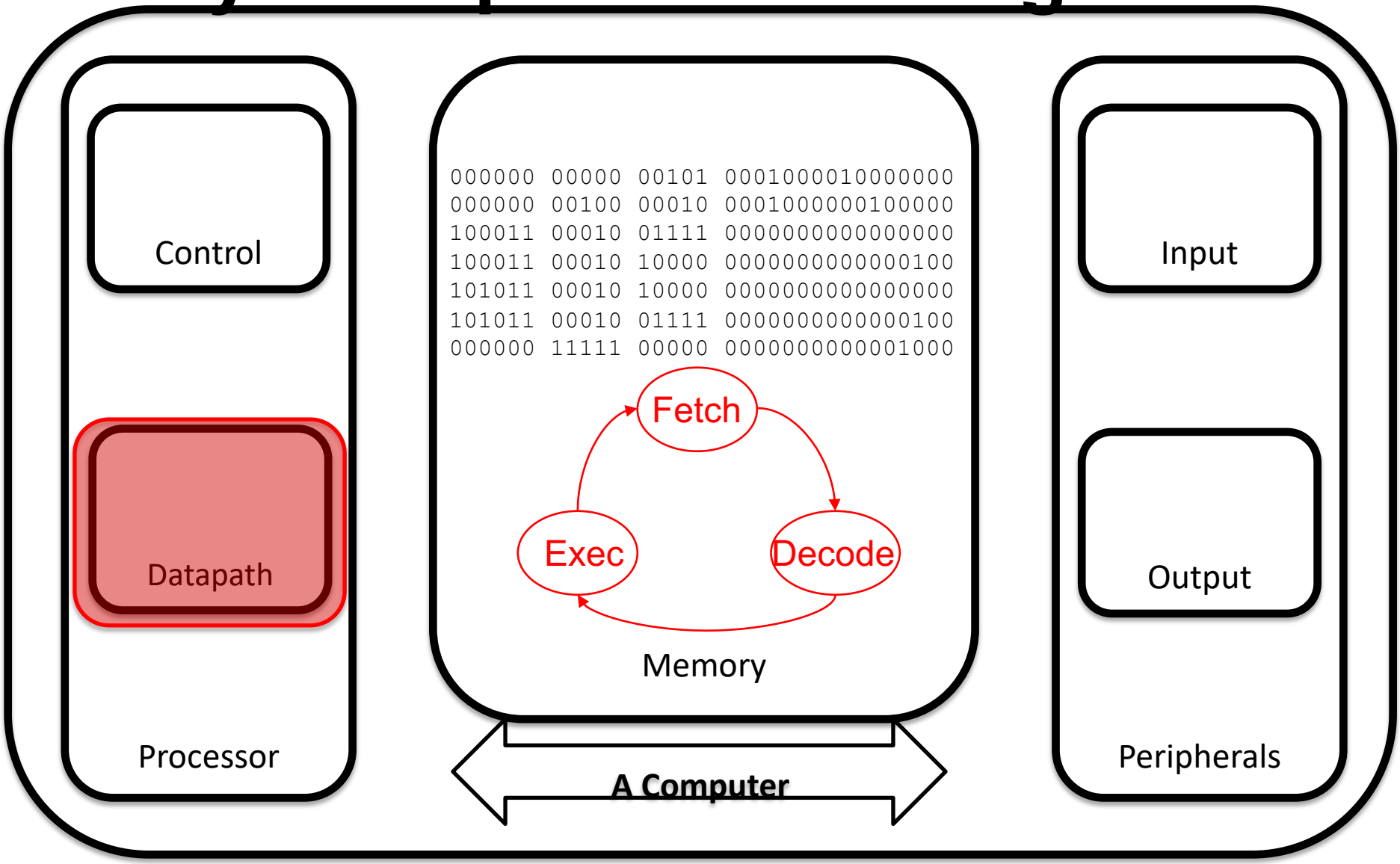
- Exam 1: T-5 days

- In lab: starting term projects!!!

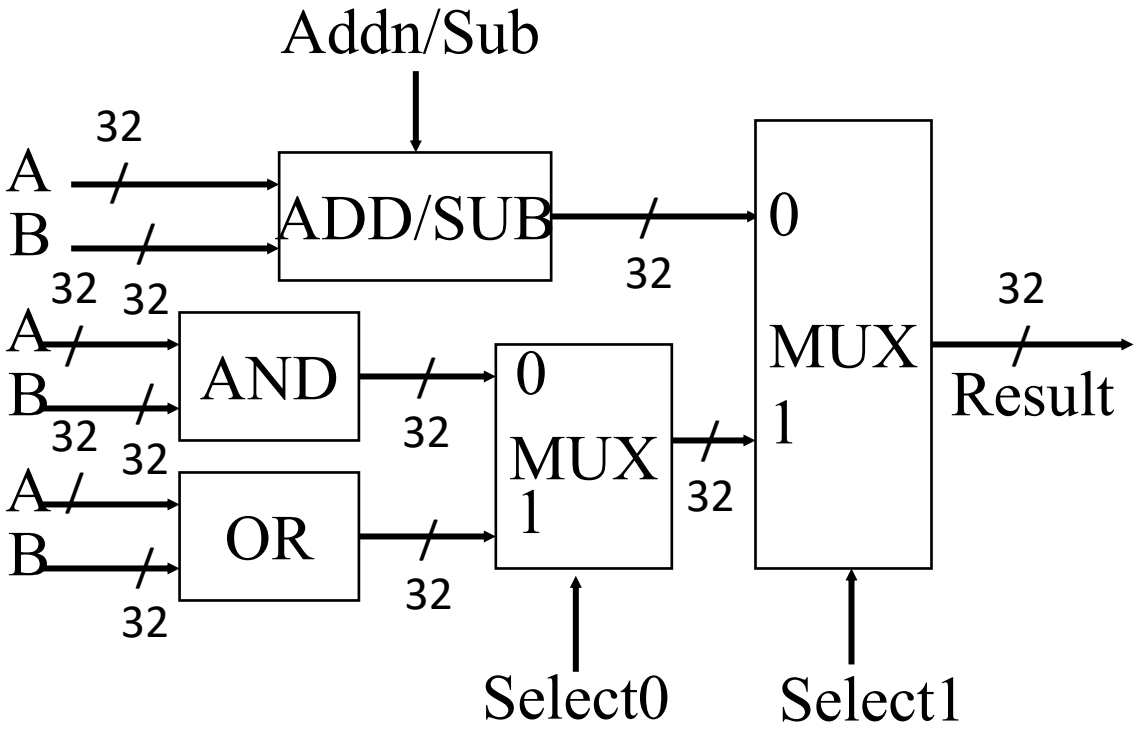
Administrative

- Exam 1
  - When: Mon, Feb 12 In-Class
  - Where: TBD (looking for a larger classroom)
  - What: Intro to computers + MIPS assembly
  - TODOs:
    - Send me conflicts/accommodations ASAP
    - Review in-class assessments, HWs, Lab Designs, P&H exercises, lecture slides, readings

Today: Datapath/ALU Design

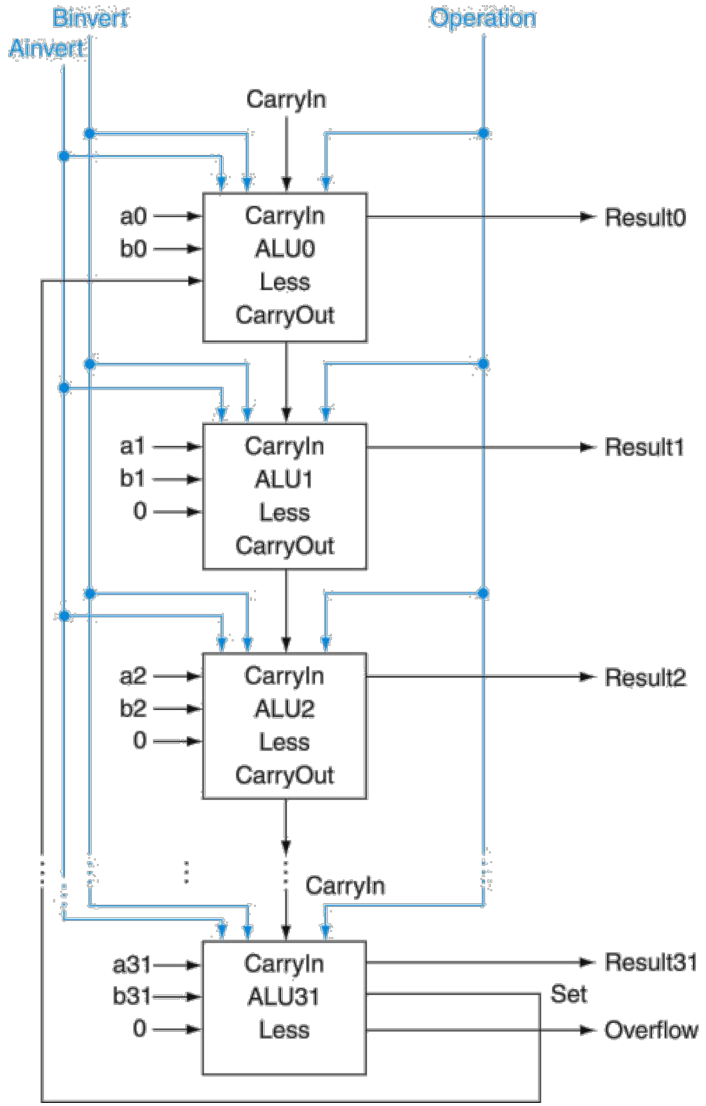


ALU Design Description



The Book's Version

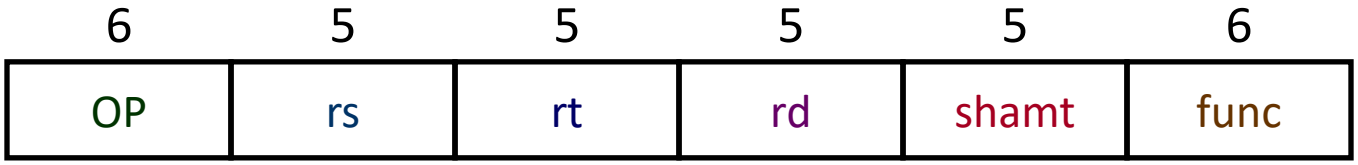
- A Ripple carry ALU
- Two bits decide operation
  - Add/Sub
  - AND
  - OR
  - LESS
- 1 bit decide add/sub operation
- A carry in bit
- Bit 31 generates overflow and set bit



Shift Operations

- Shifts move all the bits in a word left or right

```
sll  $t2, $s0, 8  # $t2 = $s0 << 8 bits
srl  $t2, $s0, 8  # $t2 = $s0 >> 8 bits
sra  $t2, $s0, 8  # $t2 = $s0 >> 8 bits
```



- Notice that a 5-bit shamt field is enough to shift a 32-bit value $2^5 - 1$ or 31 bit positions
- Logical shifts fill with zeros, arithmetic left shifts fill with the sign bit
- The shift operation is implemented by a barrel shifter

Shift Operations

- Shifts move all the bits in a word left or right

```
sll  $t2, $s0, 8   # $t2 = $s0 << 8 bits
srl  $t2, $s0, 8   # $t2 = $s0 >> 8 bits
sra  $t2, $s0, 8   # $t2 = $s0 >> 8 bits
```

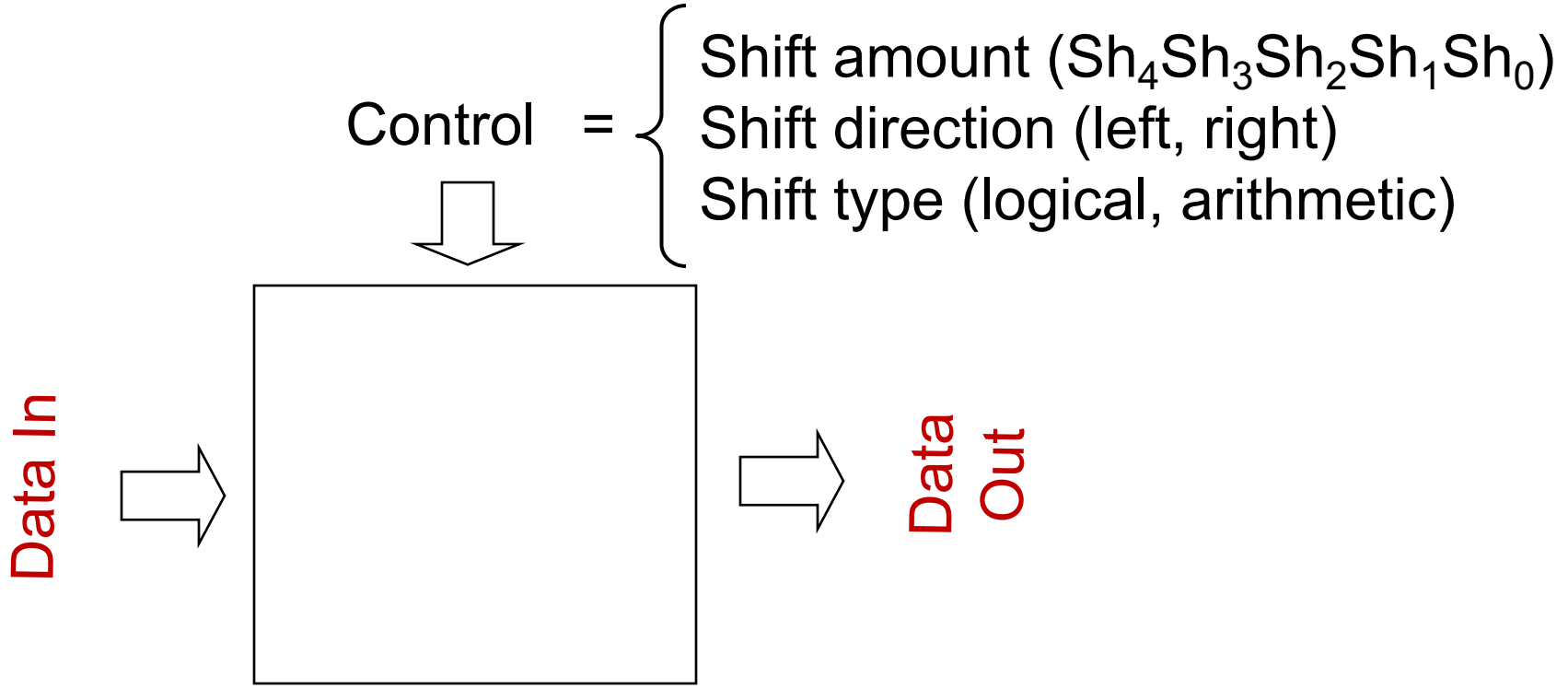
In-class Assessment!

Access Code: RollOutThe

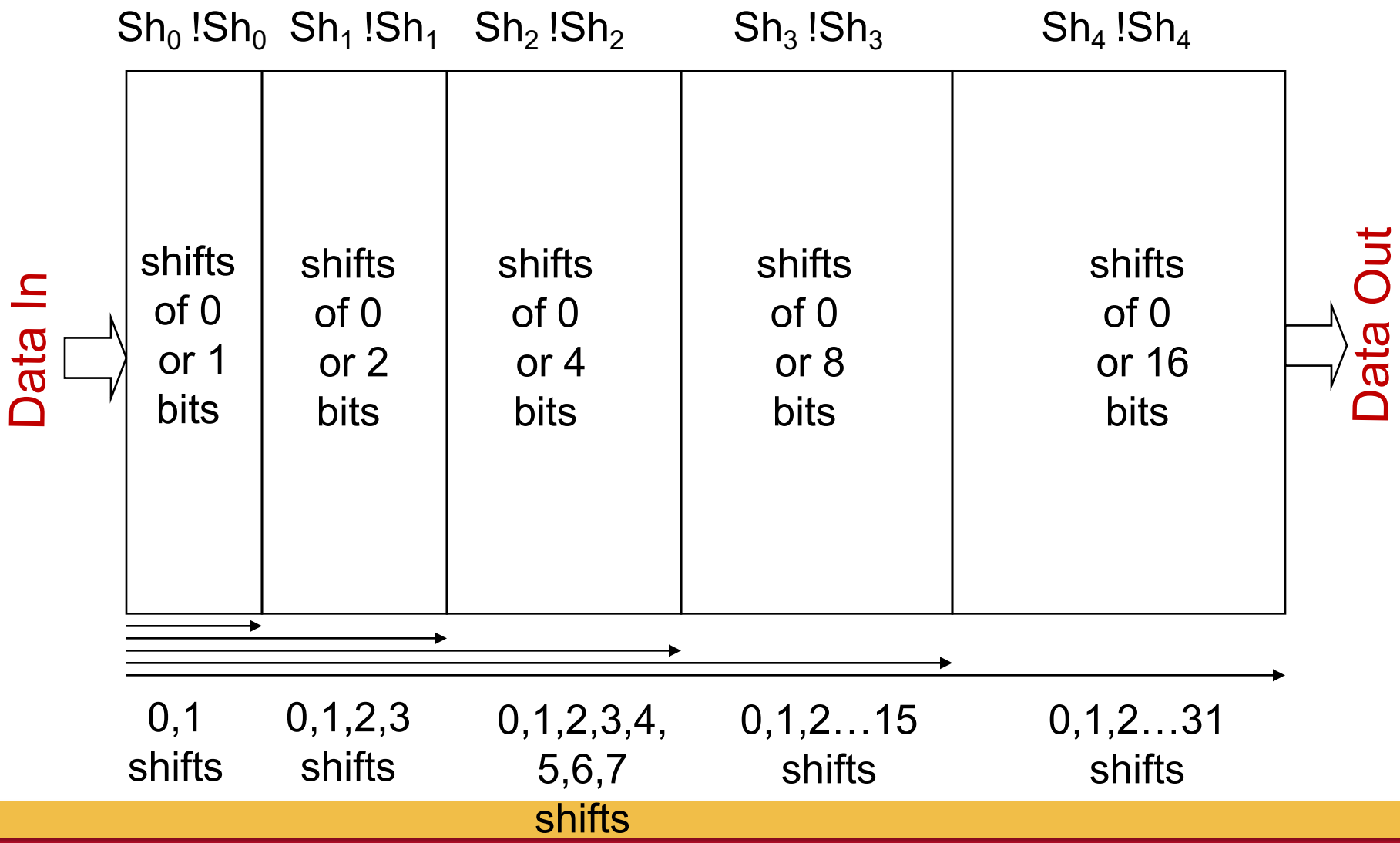
Note: sharing access code to those outside of classroom or using access while outside of classroom is considered cheating

- the sign bit
- The shift operation is implemented by a barrel shifter

Parallel Programmable Shifters



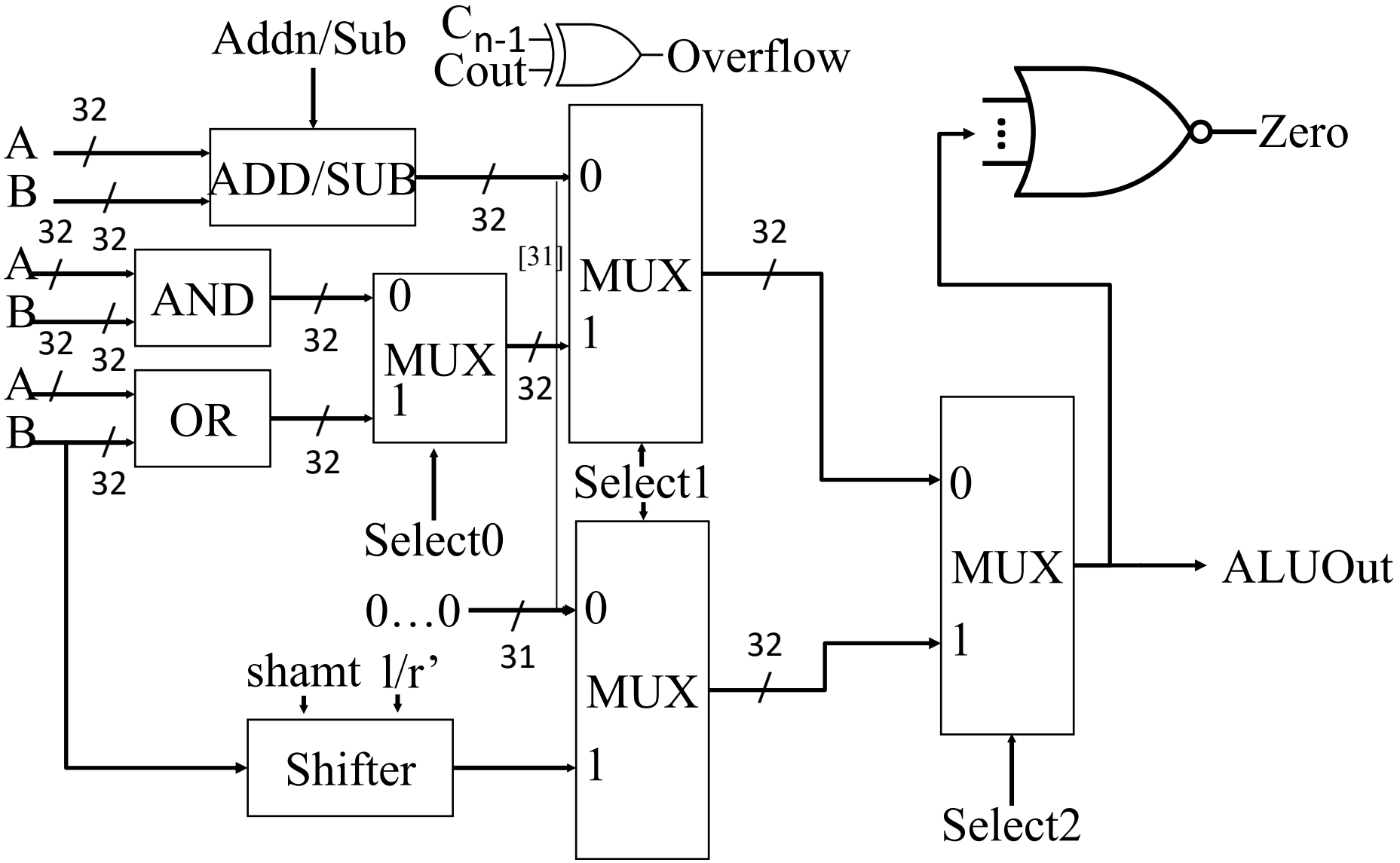
Logarithmic Shifter Structure



Tailoring the ALU to the MIPS

- Need to support the set-on-less-than instruction (`slt`)
  - Remember: `slt` is an arithmetic instruction
  - Produces a 1 if rs < rt and 0 otherwise
  - Use subtraction: (a-b) < 0 implies a < b
- Need to support test for equality (`beq $t5, $t6, $t7`)
  - Use subtraction: (a-b) = 0 implies a = b

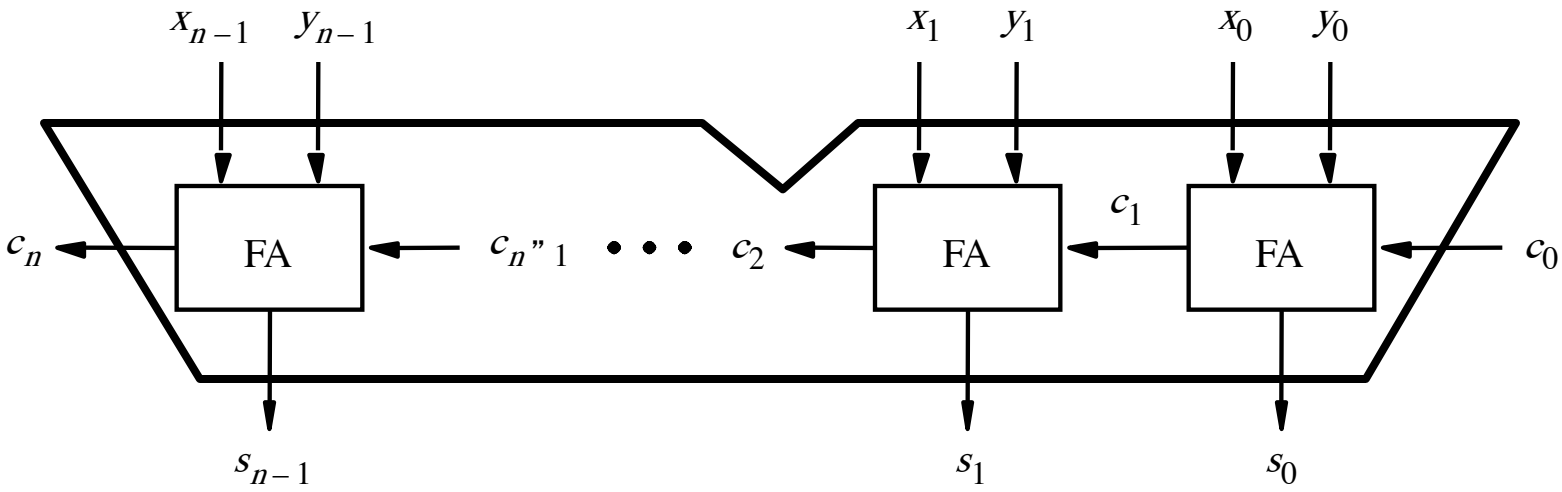
Putting it All Together: ALU Design Description



Simple ALU Summary

- We can build an ALU to support the MIPS instruction set
  - Focus is on comprehension, not performance
  - Key idea: use multiplexor to select the output we want
  - We can efficiently perform subtraction using two's complement
- **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")**

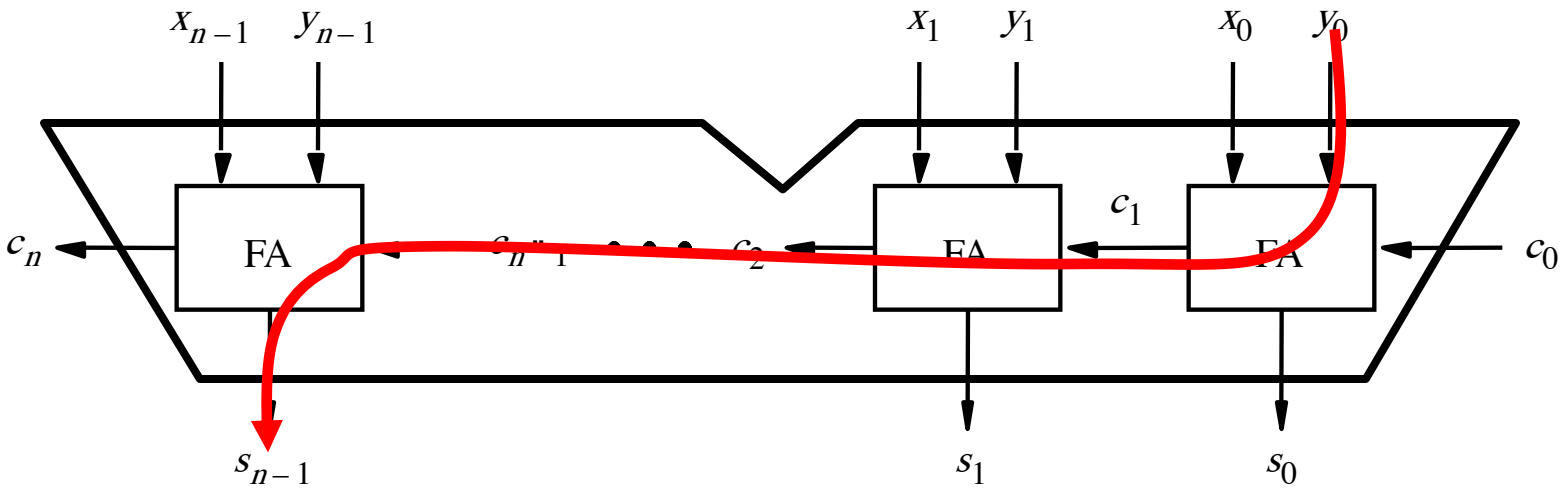
Propagation Delay



Data represented by voltages!

Voltages don't change instantly.

Propagation Delay



Data represented by voltages!

Voltages don't change instantly.

Acknowledgments

- These slides contain material developed and copyright by:
  - Joe Zambreno (Iowa State)
  - David Patterson (UC Berkeley)
  - Mary Jane Irwin (Penn State)
  - Christos Kozyrakis (Stanford)
  - Onur Mutlu (Carnegie Mellon)
  - Krste Asanović (UC Berkeley)