EE434
ASIC & Digital Systems

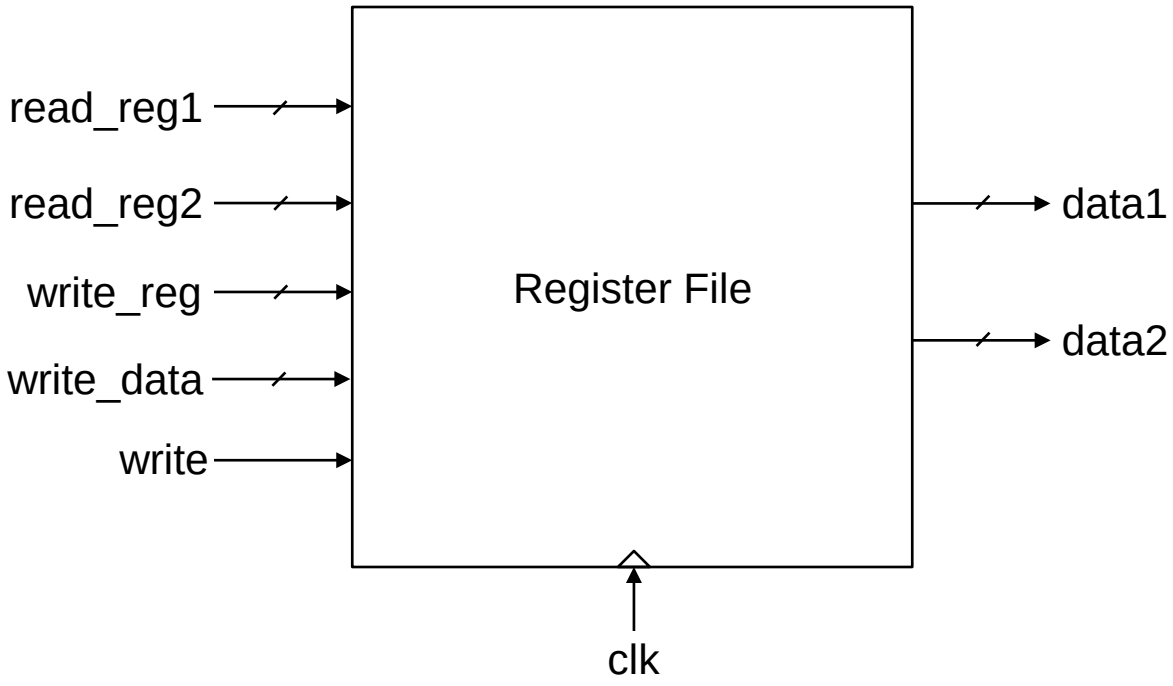
Lab1

Design of a Register File and an ALU for MIPS64

Dae Hyun Kim
daehyun@eecs.wsu.edu

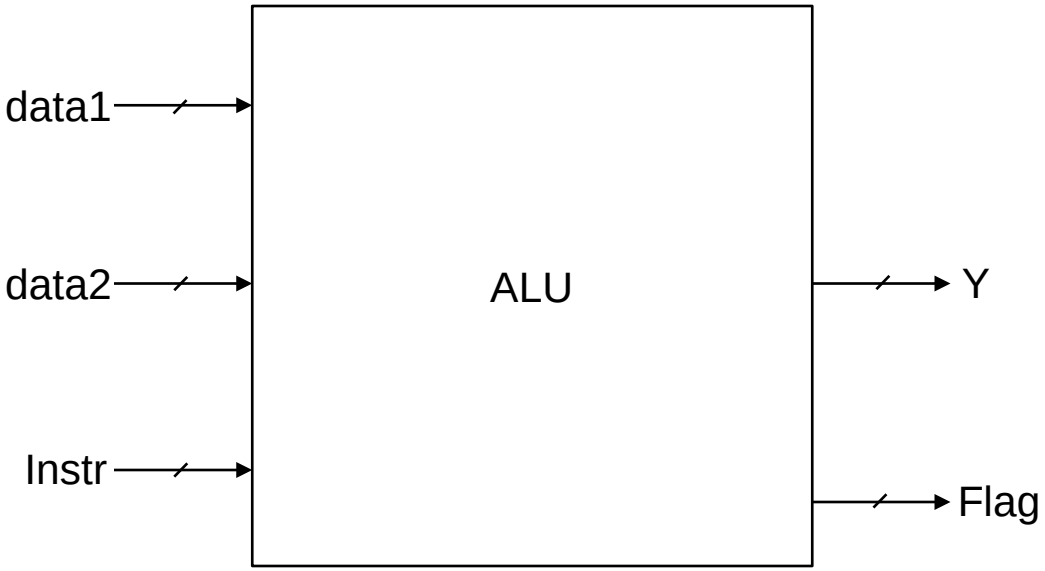
Overview

- Register File (RF)

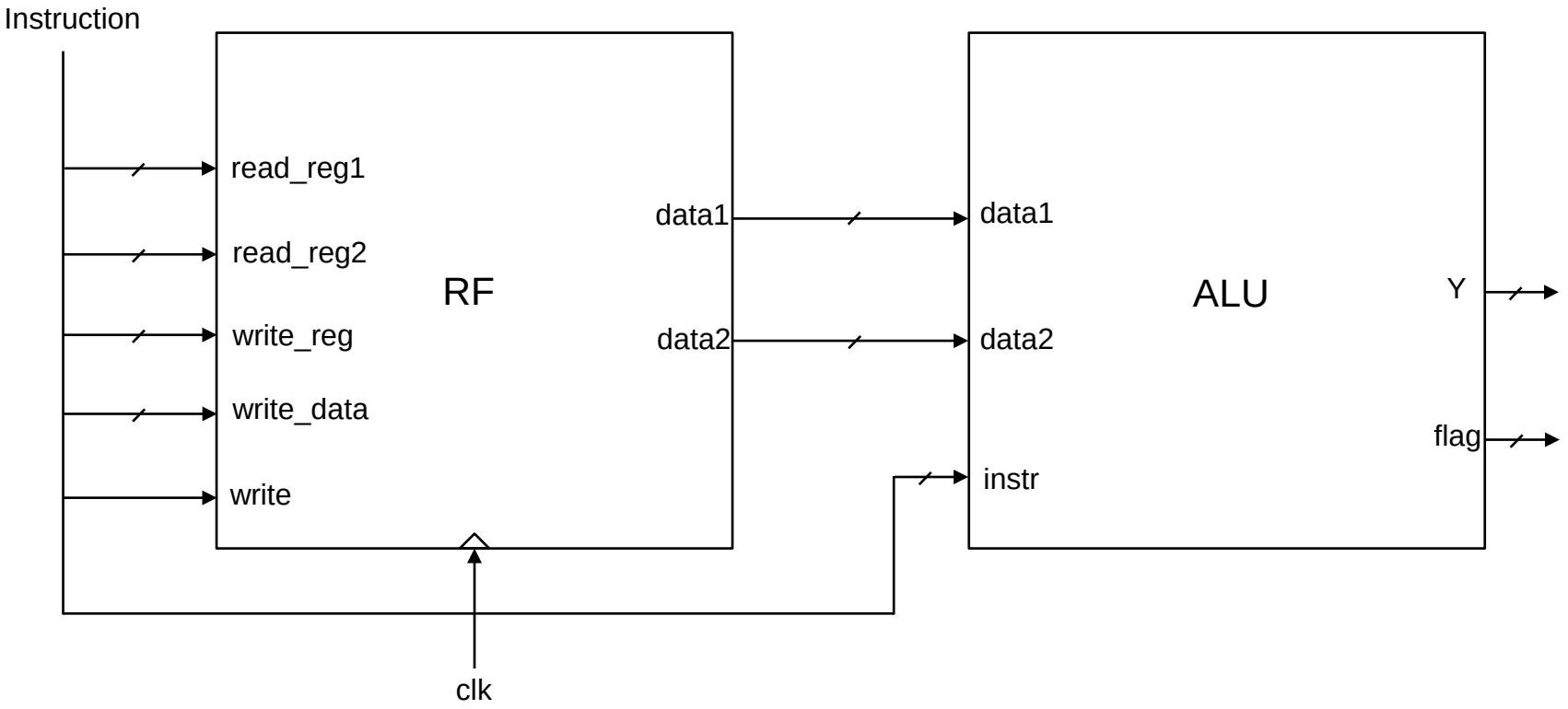


Overview

- Arithmetic Logic Unit (ALU)



Overview


Instruction
read_reg1
read_reg2
write_reg
write_data
write
RF
data1
data2
clk
data1
data2
instr
ALU
Y
flag

RF: Inputs and Outputs

- Input
  - Register id (0 ~ 31)
    - read_reg1: std_logic_vector [4:0]
    - read_reg2: std_logic_vector [4:0]
    - write_reg: std_logic_vector [4:0]
  - Data
    - write_data: std_logic_vector [63:0]
  - Control
    - write: std_logic
  - Clock
    - clk: std_logic

- Output
  - data1: std_logic_vector [63:0]
  - data2: std_logic_vector [63:0]

RF: Architecture

- It has 32 64-bit registers (r0 ~ r31).
- Each register is accessed by a 5-bit vector.

RF: Functionality

- When Clk ↑
  - If write = ‘0’
    - data1 <= mem[read_reg1]
    - data2 <= mem[read_reg2]
  - If write = ‘1’
    - mem[write_reg] <= write_data

ALU: Inputs and Outputs

- Input
  - Data
    - data1: std_logic_vector [63:0]
    - data2: std_logic_vector [63:0]
  - Control
    - instr: std_logic_vector [11:0]

- Output
  - Y: std_logic_vector [63:0]
  - flag: std_logic_vector [1:0]

ALU

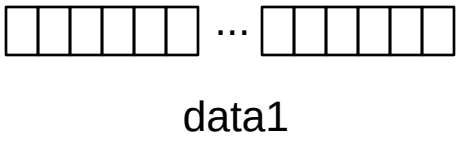
- flag
  - flag[0] (zero flag)
    - 1 if the result is zero
    - 0 otherwise
  - flag[1] (branch flag)
    - 1 if “branch” should executed
    - 0 otherwise

ALU: Functionality

instr [11:6]	instr [5:0]	Function	Y	flag [1:0]
000000	100000	ADD	data1 + data2	flag[0]
000000	100010	SUB	data1 – data2	flag[0]
000000	100100	AND	data1 AND data2	flag[0]
000000	100101	OR	data1 OR data2	flag[0]
000000	000000	SLL	shift left (data1 << data2)	
000000	000010	SRL	shift right (data1 >> data2)	
000100	------	BEQ	Branch on equal (data1 == data2)	flag[1]
000101	------	BNE	Branch on not equal (data1 != data2)	flag[1]
000001	------	BGEZ	Branch if data1 ≥ 0	flag[1]
000110	------	BLEZ	Branch if data1 ≤ 0	flag[1]

ALU: Functionality

- Shift left (SLL) or shift right (SRL)
  - data2 = k

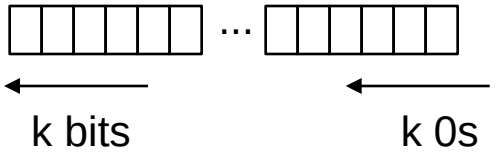


Example:

data2 = 000...0011 (3_{10})

data1 = 111...1111

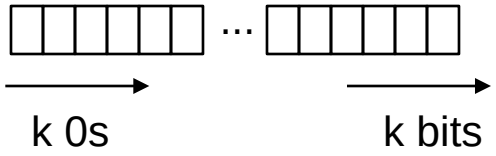
  - SLL



SLL: Y = 111...11000

SRL: Y = 000111...111

  - SRL



Implement SLL and SRL!

Do not use “sll” and “srl”!

ALU: Functionality

- Branch
  - Set flag[1] to ‘1’ if the branch condition is true.

Simulation

- Test input vectors will be given to you.
- Entity definition

```
ENTITY  myMIPS  IS
   PORT  (
      read_reg1, read_reg2, write_reg : IN std_logic_vector (4 DOWNTO 0);
      write_data : IN std_logic_vector (63 DOWNTO 0);
      instr : IN std_logic_vector (11 DOWNTO 0);
      write, clk : IN std_logic;
      Y : OUT std_logic_vector (63 DOWNTO 0);
      flag : OUT std_logic_vector (1 DOWNTO 0)
   );
END  myMIPS;
```

Submission

- Deadline
  - Feb. 28 (Sunday), 11:59:59pm.

- What to submit
  - Source code