

# Testbench Writing

## 2010 CVSD Verilog Homework 4

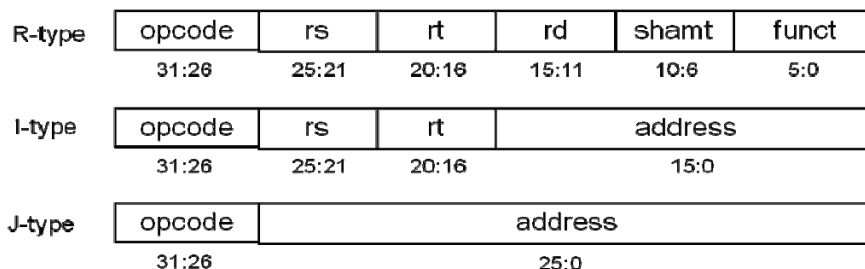
### I. Objective

Please write the testbench to debug a single cycle MIPS CPU. There are **some errors** in the design. You need to write the testbench to check all instruction sets, and then correct these errors. These errors include function errors and coding errors.

### II. Description of MIPS CPU

#### A. Supported instructions:

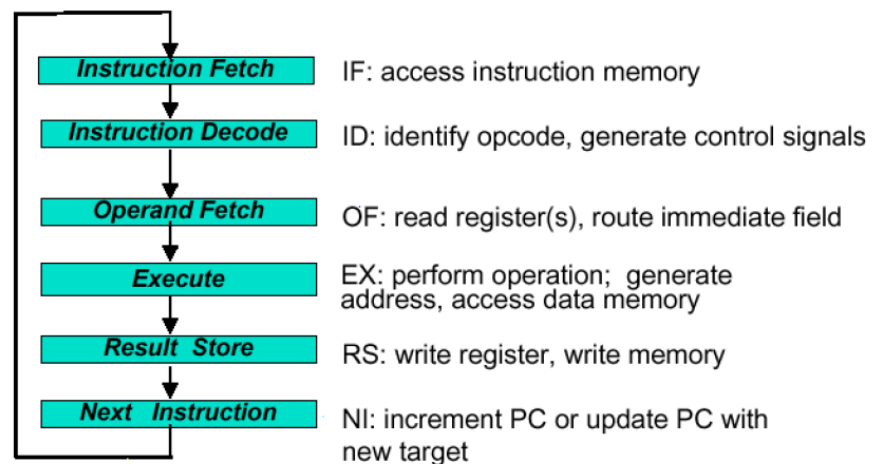
MIPS instruction	Name	Format (opcode)	
Add	add	R-type	00_0000
Subtract	sub	R-type	00_0000
And	and	R-type	00_0000
OR	or	R-type	00_0000
Set less than	slt	R-type	00_0000
Load word	lw	I-type	10_0011
Store word	sw	I-type	10_1011
Branch on equal	beq	I-type	00_0100
Jump	j	J-type	00_0010
Jump and link	jal	J-type	00_0011
Jump register	jr	R-type	00_0000



## B. ALU OP Code

Instruction opcode	ALUOp	Instruction operation	Funct field	Desired ALU action
LW	00	load word	xxxxxx	add
SW		store word	xxxxxx	add
Branch equal	01	branch equal	xxxxxx	subtract
R-type	10	add	100000	add
R-type		subtract	100010	subtract
R-type		AND	100100	and
R-type		OR	100101	or
R-type		set on less than	101010	set on less than

## C. Typical Instruction Execution



Note that each step does not necessarily correspond to a clock cycle. These only describe the basic flow of instruction execution. The details vary with instruction type.

Instruction class	Functional units used by the instruction class				
R-type	Instruction fetch	Register access	ALU	Register access	
Load word	Instruction fetch	Register access	ALU	Memory access	Register access
Store word	Instruction fetch	Register access	ALU	Memory access	
Branch	Instruction fetch	Register access	ALU		
Jump	Instruction fetch				

### 1. Instruction fetching

- The CPU is always in an infinite loop, fetching instructions from memory and executing them.
- The program counter or PC register holds the address of the current instruction.
- MIPS instructions are each four bytes long, so the PC should be incremented by four to read the next instruction in sequence.

### 2. Instruction Decode

- The Instruction Decode (ID) step reads the source register from the register file.

### 3. Execute

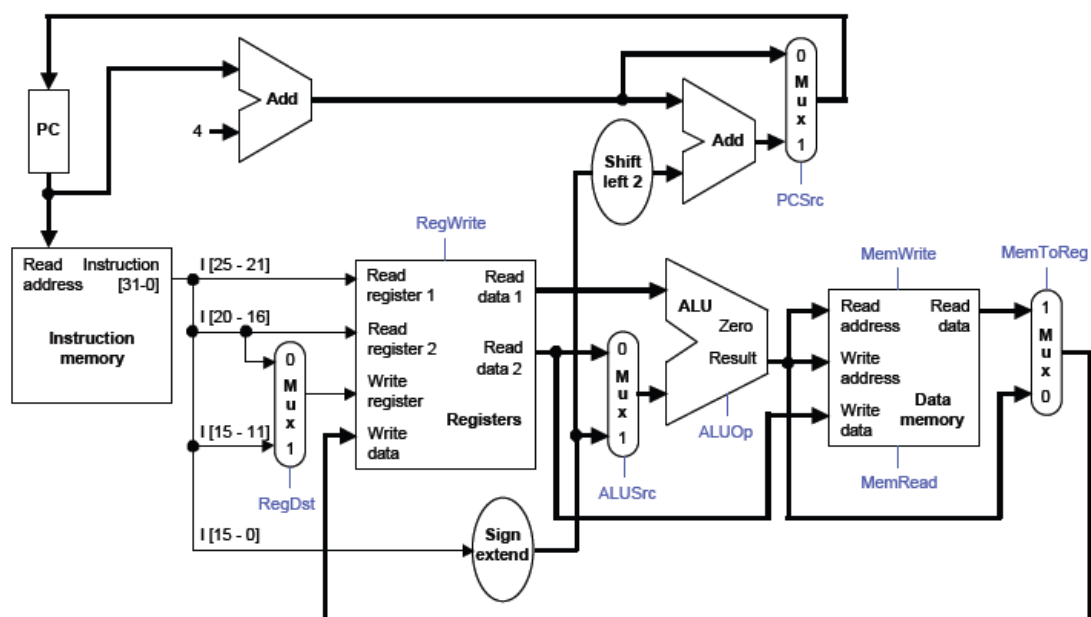
- The third step, Execute (EX), computes the effective memory address from the source register and the instruction's constant field.

### 4. Memory

- The Memory (MEM) step involves reading the data memory, from the address computed by the ALU.

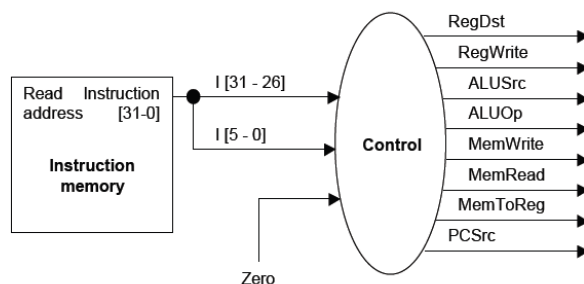
### 5. Write Back

- Finally, in the Writeback (WB) step, the memory value is stored into the destination register.



## 6. Control Unit

- The control unit needs 13 bits of inputs.
  - Six bits make up the instruction's opcode.
  - Six bits come from the instruction's func field.
  - It also needs the Zero output of the ALU.
- The control unit generates 10 bits of output, corresponding to the signals mentioned in the previous page.



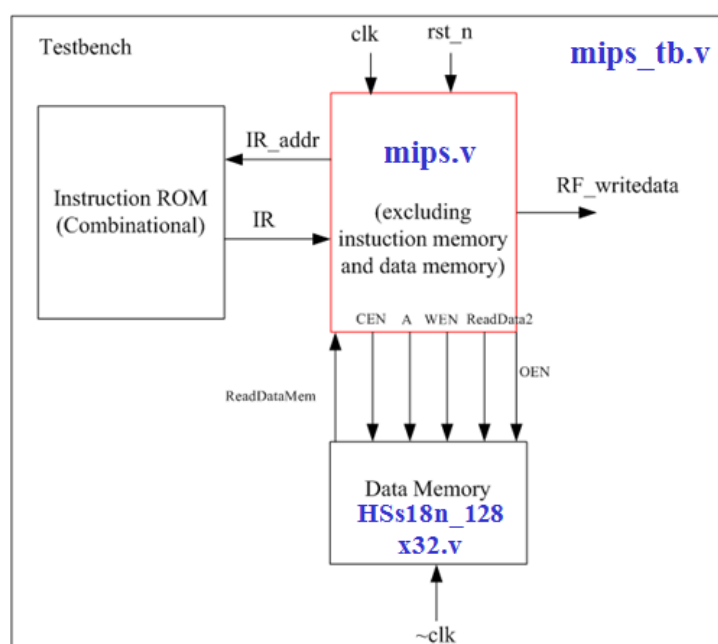
## III. File List & Block Diagram & I/O Definition

### A. File list of the homework package:

mips.v	A single cycle MIPS CPU with some errors.
mips_tb.v	Empty file for you to write the testbench.
HSs18n_128x32.v	A 128X32 high-speed single-port synchronous SRAM.

### B. Block Diagram

Testbench includes the mips, data memory and instruction ROM.



**D. File I/O definition:**

clk	<i>Posedge</i> clock signal.
rst_n	Active low asynchronous reset.
IR_addr	Instruction address
IR	Instruction
RF_writedata	The data to be written into the register file. Used for testing your circuit.
ReadDataMem	MIPS- Read data from memory. SRAM- Data outputs.
CEN	SRAM- Chip enable. 0 when you read/write data.
WEN	SRAM- Write enable. 0 when you write data into SRAM
A	SRAM- Address
ReadData2	MIPS- Write data to memory. SRAM-Data inputs.
OEN	SRAM- Read Enable. 0 when you read data from SRAM

**IV. Testbench Requirements**

- A. Initialize the instruction memory and the data memory.
- B. Reset your circuit.
- C. Execute the instructions, and read the values of RF\_writedata and IR\_addr to see if your circuit is correct.
- D. If your function is correct, please print the message is shown as below:

```

-----
    Congratulations!! Your design has passed all the test!!
-----

```

**V. Grading Policy**

- 1.mips.v & mips\_tb.v: correctness: 75%
- 2.report: Please **show the errors** of MIPS and describe your **debug strategy in detailed**. 25%

**(You can run the specific program to test the CPU. Ex: Fibonacci program, Bubble sorting program and so on)**

## VI. Submission Requirements

File submission: **1.mips.v (no errors), 2.mips\_tb.v, 3.report file**

In general, your report should be less than **2 pages** (1 page suggested) with at least 12pt fonts (but not including the parts of synthesis results.)

Please submit your design in one .zip file with the naming convention: **StudentID\_HW6.zip** (e.g., R99943001\_HW6.zip).

## VII. Submission Deadline

Please submit your **.zip** files online using ftp. (Deadline: 12/3 Friday)

## VIII. Reference Book

David A. Patterson and John L. Hennessy. *Computer Organization & Design: The Hardware/Software Interface*. 3<sup>rd</sup> edition.

Creator:

1<sup>st</sup> Edition: En-Jui Chang, 2010