1122 Digital System Design Final Project

Pipelined RISC-V Design

TA Information: 蔡文喬、羅翊誠

daniel@access.ee.ntu.edu.tw; alan@access.ee.ntu.edu.tw

1. Project Description

Table 1. Required Instruction Set

Name	Description
ADD	Addition, overflow detection for signed operand is not required*
ADDI	Addition immediate with sign-extension, without overflow detection*
SUB	Subtract, overflow detection for signed operand is not required*
AND	Boolean logic operation
ANDI	Boolean logic operation with 12bit of immediate
OR	Boolean logic operation
ORI	Boolean logic operation with 12bit of immediate
XOR	Boolean logic operation
XORI	Boolean logic operation with 12bit of immediate
SLLI	Shift left logical (zero padding)
SRAI	Shift right arithmetic (sign-digit padding)
SRLI	Shift right logical (zero padding)
SLT	Set less than, comparison instruction
SLTI	Set less than variable, comparison instruction
BEQ	Branch on equal, conditional branch instruction
BNE	Branch on not equal, conditional branch instruction
JAL	Unconditionally jump and link (Save next PC in $rd)
JALR	Jump and link register(Save next PC in $rd)
LW	Load word from data memory (assign word-aligned)
SW	Store word to data memory (assign word-aligned)
NOP	No operation(addi $r0 $r0 0)

* Different from the definition in [1], the exception handler for arithmetic overflow is not required.

In this final project, you are asked to design a **pipelined RISC-V processor (synchronous active low reset) with instruction cache and data cache**. This processor should at least support the instruction set defined in Table 1. The instruction set is referenced from Chapter 2 (RV32I base integer instruction set) of [1], and we encourage you to read it in detail.

The whole module hierarchy is shown in Figure 1. The processor architecture is shown in Figure 2. As you see, this is modified from the single-cycle architecture of our HW2. Your design should follow this **5-stage pipelined architecture**. You need to modify several parts to fit our specifications. For example, you need to add the path for **J-type instructions**.

Also, you should **solve the hazards** by adding some circuits. There are 3 hazard categories that should be properly handled in your pipelined processor:

1) Structure hazard
2) Data hazard
3) Branch hazard

Although all of these hazards can be solved by inserting NOP manually or automatically in your test program, we ask you to implement a **data forwarding unit** and **pipeline stall unit** to solve these hazards.

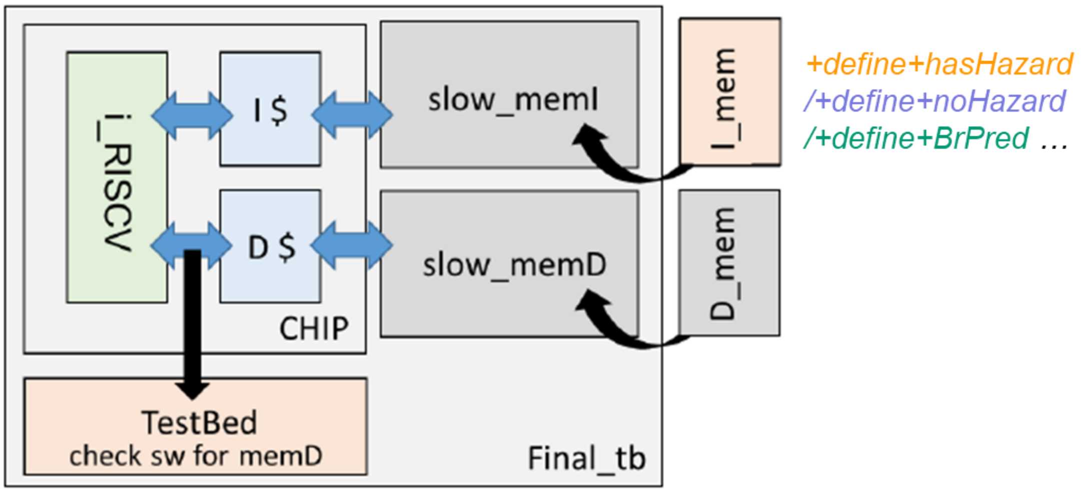

Figure 1. Module Hierarchy

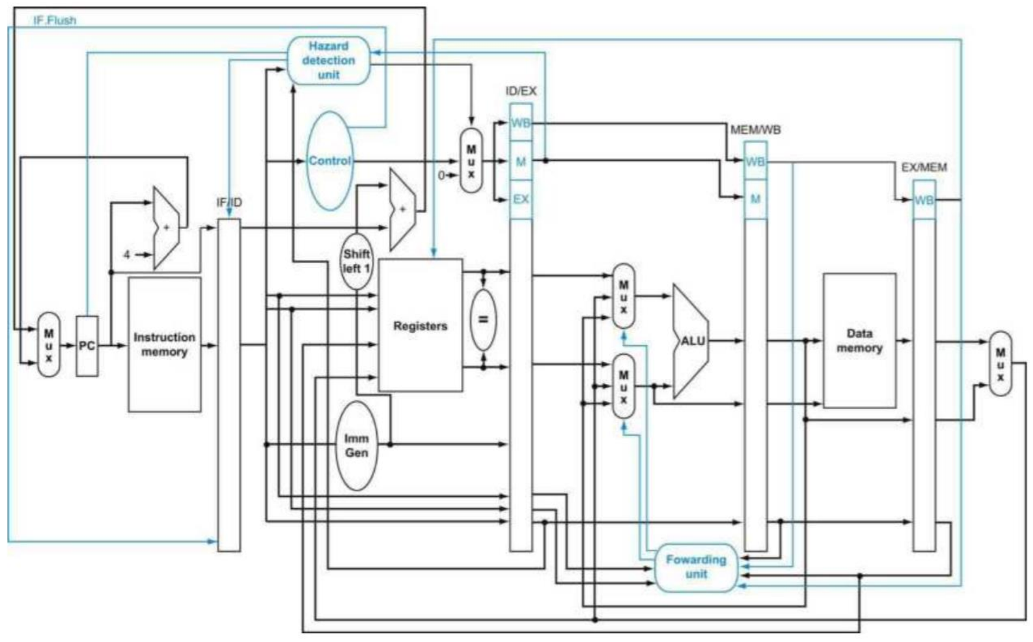

Figure 2. Simplified Pipeline Architecture of RISCV

2. Cache and Memory Interface

The instruction memory and data memory will not be contained in your design. The memory interface is left as module I/O. You have to use the provided slow memory model. **Do not synthesize the slow memory.**

The cache units are suggested to have the same block number (8) and block size (4) as in HW3. Besides, we do not restrict the replacement policy and writing policy of the cache design. You are encouraged to optimize the cache units to fit your RISC-V design.

3. Synthesis Notes

You should synthesize your design using TSMC 0.13 cell library, and the relevant files, e.g. *.synopsys_dc.setup*, can be copied from previous HW or use the attached file. The design constraints are specified in "*CHIP_syn.sdc*". Note that the pipelined RISC-V, instruction cache and data cache are included in the *CHIP.v*. They should be synthesized together.

The post-synthesis simulation is required, and all involved Verilog files should be modeled by gate level. Note that the maximum clock frequency must be verified by post-synthesis gate-level simulation. And you are recommended to buffer the input signal to avoid timing violations.

4. Grading Policy and Possible Extensions

All grades of this project consist of two aspects:

1) **Baseline checkpoint (40%)**
   - Checkpoint Presentation (5%)
   - noHazard Gate Level (15%)
   - hasHazard Gate Level (20%)
2) **Final presentation and submission (60%)**
   - Final Presentation (8%)
   - Branch Prediction Gate Level (9%)
   - Compressed Instr. Gate Level (9%)
   - Multiplication Instr. Gate Level (9%)
   - Q_sort & Conv AT Ranking (15%)
   - Report (10%)

Baseline Checkpoint (40%)

If your design meets all requirements of the description below and can be synthesized and simulated at the gate level, your design will be qualified to get baseline points. The solid requirements include:

1) Supporting all instructions above
2) With caches
3) Pass test assembly programs (noHazard, hasHazard)
4) Complete the circuit synthesis. Note that the slack cannot be negative.

Also, each team must prepare a **4-6 page slide** and a **5-minute presentation** confirming your current results and future plan.

Final Presentation and Submission (60%)

There are 3 topics of extension.

1) Branch prediction mechanism.
2) Supporting compressed instructions.
3) Supporting multiplication instructions.

Implement the topics of extension **as much and deep as you can**; the deeper the exploration, the higher the score (The content also affects the quality of your presentation and report).

Each team should prepare a full talk (within 10 minutes, about 10-15 slides) for your fantastic work on extension!

Then the AT performance is evaluated by Q_sort and Conv:

Area (um2) × Simulation time of Q_sort (ns) × Simulation time of Conv (ns).

5. Simulation Example

[License]

```
source /usr/cad/synopsys/CIC/vcs.cshrc
source /usr/spring_soft/CIC/verdi.cshrc
```

[Simulation]

```
// RTL
vcs Final_tb.v CHIP.v slow_memory.v [other RTL files] -full64 -R -
debug_access+all +v2k +define+noHazard

// Gate level
vcs Final_tb.v CHIP_syn.v slow_memory.v -v tsmc13.v -full64 -R
-debug_access+all +v2k +define+noHazard +define+SDF

// Simulate under the directory Src/
// For baseline, “noHazard” can be changed to “hasHazard”
// For extensions, change “noHazard” to “BrPred”, “compression”,
“compression_uncompressed”, or “Mul”
// For QSort and Conv, change “noHazard” to “QSort(_uncompressed)” or
“Conv(_uncompressed)”, see Src/Final_tb.v for more details
```

6. Submission Requirement

All the files need to be compressed as a single **ZIP file** and **uploaded to NTU COOL for each team.**

For Checkpoint Submission on 5/30:

Example of filename

```
    DSD_Final_Check_G#_v#.zip
    e.g. DSD_Final_Check_G1_v2.zip
```

Your submitted file should include the following files:

```
    DSD_Final_Check_G#/
        Src/
            rtl/
                CHIP.v
                [other RTL files]
            syn/
                CHIP_syn.v
                CHIP_syn.sdf
                CHIP_syn.ddc
        DSD_Final_Checkpoint_Scores.pdf
        Check_Presentation.pptx
```

For Final Submission on 6/17:

Example of filename

```
DSD_Final_G#_v#.zip
e.g. DSD_Final_G1_v2.zip
```

Your submitted file should include the following files:

```
DSD_Final_G#/
  Src/
    rtl/
        CHIP.v
        [other RTL files]
    syn/
        CHIP_syn.v
        CHIP_syn.sdf
        CHIP_syn.ddc

  DSD_Final_Project_Scores.pdf
  Final_Presentation.pptx
  Report.pdf
```

The homework will be graded **ONLY IF** the filename of your submission is correct!

7. Schedule and Necessary Submissions

Date	Submission/Event
5/16	Final project announcement
5/30	A. Baseline checkpoint. Each team should prepare a presentation (*4-6 pages, about 5 minutes*) to confirm your current results and future plan. You should ***upload the results and slide to COOL before 1 pm***. B. Briefly show your design for passing Baseline testbenches (noHazard, hasHazard). C. Extension topics plan should be included in the presentation D. Please attach the work assignment chart on last page
6/13	Final presentation. Each team should prepare a full talk (about 10-15 slides *within 10 minutes*) to demonstrate your fantastic work! Detail presentation plan will be announced on COOL.
6/17	Final submission, including a detailed report (*8-16 pages*), the presentation slides, and all the source codes (including all the RTL code and synthesis related files: *.v, *.sdf, *.ddc and a Readme.txt). You should upload the final submission to COOL by the day.

8. Reference

[1] https://riscv.org/specifications/