VLSI System Design (Graduate Level)

Fall 2020

HOMEWORK I

REPORT

Must do self-checking before submission:

Compress all files described in the problem into one tar

All SystemVerilog files can be compiled under SoC Lab environment

All port declarations comply with I/O port specifications

Organize files according to File Hierarchy Requirement

No any waveform files in deliverables

Student name: \_\_\_\_\_\_\_\_\_\_\_

Student ID: \_\_\_\_\_\_\_\_\_\_

|  |  |
| --- | --- |
| INTRODUCTION FOR DESIGN OF THE CPU | |
| SPEC | |
| PIPELINE STAGE | 5 |
| INSTRUCTION | 33 |
| FULL GRAPH | |
|  | |

|  |  |  |
| --- | --- | --- |
| STAGE1 INSTRUCTION FETCH | | |
|  | | |
| MODULE | FUNCTION | SPECIAL\_DESIGN |
| DIDER4 | Get the Instruction Address by dividing  the pc\_counter result by4 | Using shift right 2 bits instead using the divider which can reduce hardware area. |
| PAUSE INSTRUCTION | If there is any load use data hazard happen,this module will enable to pause the instruction.  The instruction amd pc value will be rewrited tio the register in next clock. | NONE |
| rst\_contoller | If there is any branch happen this module will enable flush the register. The module will be more detailed discuss later. | NONE |

|  |  |  |
| --- | --- | --- |
| STAGE2 INSTRUCTION DECODE & STAGE5 WRITE BACK | | |
|  | | |
| MODULE | FUNCTION | SPECIAL\_DESIGN |
| DECODER | Decoding 33 bits instruction into  different elements which is used for the following module | NONE |
| SIGNED UNSIGNED EXTENDED\_  SELECT\_IMM | Make the Immediate data referring to the type of the instruction | NONE |
| REG | 32×32-bit. registers | Posedge clk write back  Negedge clk read data |
| CONTROL | Make the Control signal | NONE |
| rst\_contoller | If there is any branch or load use data hazard happen  this module will enable flush the register.  The module will be more detailed discuss later. | NONE |

|  |  |  |
| --- | --- | --- |
| STAGE3 ALU EXECUTION | | |
|  | | |
| MODULE | FUNCTION | SPECIAL\_DESIGN |
| ALU\_IN | Data hazard signal will be the input module will decide the input of the alu. | NONE |
| ALU\_RD | calculate for the data store in $rd | NONE |
| ALU\_ADDR | calculate for the Data memory’s Address which will be stored data next clock. | NONE |
| PC\_ADDR | calculate the jumping address and make the signal to enable the system jump or not. | NONE |
| rst\_contoller | If there is any branch happen this module will enable flush the register. The module will be more detailed discuss later. | NONE |

|  |  |  |
| --- | --- | --- |
| STAGE4 WRITE OR READ MEMORY | | |
|  | | |
| MODULE | FUNCTION | SPECIAL\_DESIGN |
| DIVIDER4 |  |  |
| LOW\_BYTE\_CONTROL |  |  |
| WB\_CONTROL |  |  |

|  |  |
| --- | --- |
| OTHER CONTROL MODULE | |
| MODULE | LOAD\_DATA\_HAZARD |
|  | FUNCTION | SPECIAL\_DESIGN |
| FUNCTION | SPECIAL\_DESIGN |
|  |  |
| MODULE | FORWARDING UNIT |
|  |
| FUNCTION | SPECIAL\_DESIGN |
|  |  |

|  |  |
| --- | --- |
| MODULE | rst\_controller (for exe\_mem stage and if \_if stage) |
|  |
| FUNCTION | SPECIAL\_DESIGN |
|  |  |
| MODULE | PC\_CONTROL |
|  |
| FUNCTION | SPECIAL\_DESIGN |
|  |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |