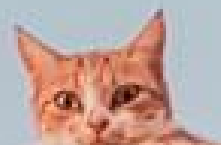

Computer Orgnization

CS202-2025s CPU Project

Overall Explanation 1 (Important!)

Development board requisition:
1) Each group has one development board. Please protect the board and compensate according to the price if it is lost or damaged.
2) Complete basic testing within one week of borrowing the development board. If there are any issues, please provide specific feedback and promptly contact the teacher for replacement.

Teaming rules:
1) It is necessary to ensure that all members are present during the defense time. If they are not present, the number of team members will be automatically reduced, and 100% of the contribution ratio will be allocated based on the actual number of members.
2) It is recommended that students in the same experimental class form teams, or form teams across classes (as everyone must be present during the defense, it is not recommended to form teams across classes).
3) Suggest a team of 3 people (in special circumstances, a team of 2 people is also possible, but not recommended).
4) The minimum and maximum contribution ratios shall not exceed 10%: that is, the maximum ratio for a two person group is 45:55, and the maximum ratio for a three person group is 30:30:40.

Disassembly and reorganization team rules:
1) If the group needs to be split due to special reasons, with the consent of all members of the group (it is recommended to communicate face-to-face or confirm in the group group), the group can be split and then reorganized. An email was written by a member explaining the reason for the split, which was sent to the laboratory teacher and copied to all members.
2) The deadline for splitting up is one week before the project defense. After this time, splitting up and restructuring will no longer be supported. Please be cautious.
3) The reorganization after dismantling shall be negotiated by the students themselves. The results of the reorganization shall be sent to the laboratory teacher and copied to all members of the group.

Integrity of project: **If network code or AI generated code is used in this project, please make sure to provide an explanation (such as which part of the code references network implementation, provide website address, which part uses AI, provide AI tool name and prompt words). If no explanation is provided, if high repetition is found during code plagiarism check, it will be directly judged as dishonest project, and all function scores (defense function acceptance score, bonus score) will be 0 points.**

Overall Explanation 2(Important!)

- ✓ If the score exceeds 100, the overflow will be proportionally included in the overall evaluation
- ✓ **Personal Score =Team Score * Team Size * Individual Contribution Percentage+Individual Defense Score (15)**
- ✓ **Team score=Function score (70) * Delay/Advance coefficient (0.6~1.05)+Code specification score (3)+Document (12)+Bonus (15)**
- ✓ **Individual Defense Score =mid-term defense score (10)+final defense score (5)**
  - ✓ NOTES: **If you do not participate in the mid-term defense, your mid-term defense will be divided into 0. If you do not participate in the final defense of the project, you will automatically withdraw from the group according to the Page2 team rules**.

Defense Explanation	code submission	document（vedio）submission	Delay/Advance coefficient (*Function score)
mid-term defense (lab class in week 13)	Not involved	Submit the mid-term defense statement document and PPT， Personal PPT presentation progress on site (within 3 minutes)	Not involved
final-defense in advance (lab class in week 15)	Before 12:00 noon on Monday of the 15th week	Before 12:00 noon on Monday of the 16th week	1.05
final-defence in normal (lab class in week 16)	Before 12:00 noon on Monday of the 16th week	Before 12:00 noon on Monday of the 17th week	1
final-defence in delay (All delayed groups will be arranged uniformly after submission)	Deduct 5% coefficient for one day of delay. No score will be given if it is later than 12:00 noon on Monday of week 17.	Before 12:00 noon on Monday of the 17th week No score will be given if it is later than 12:00 noon on Monday of 17th week	1) in week16， Tuesday 0.9, Wednesday 0.85, Thursday 0.8, Friday 0.75, Saturday 0.7, Sunday 0.65. 2） in week17, Monday 0.6. The deadline for code submission is 12:00 noon on Monday of week17

Coefficient explanation: If any submission of code, document (video) is delayed, the **coefficient** will be calculated based on the latest submission time to determine the "functional score".
For example, if Group A completes the defense within 15th week and all deliverables are submitted on time according to the requirements in the above table, the coefficient will be 1.05; If the code is submitted on time but the development document is submitted on Tuesday of the 16th week, the coefficient will be calculated based on the latest submitted development document, with a corresponding coefficient of 1. If Group A delays the defense but fails to submit the code and documents by 12:00 noon on Monday of 17th week, the defense function acceptance score will be 0 points.

Personal Score Explanation:
If the contribution ratio of the three person group is 1:1:1 and the defense lasts for 15 weeks, the highest personal score can be obtained (70 * 1.05+3+12+15) * 3 * 0.33+10 (mid-term defense score)+5 (final defense score)=118.5

Submission

- **Each group only needs to submit one copy, divided into two submissions(code, document&vedio). Both submissions should be from the same member, and the information of the member responsible for submission needs to be registered in the shared document**:
  - 1st submission: source code (submit to BB site before 12:00 noon on Monday of the week of defense)
    - Submission content: ASM and COE files corresponding to the test scenario, CPU project directory (to avoid the compressed package size being too large, please delete the. runs subdirectories before packaging and uploading). During the defense, you need to download the project you submitted from BB and generate a bitstream file on site under the supervision of the inspector. Please test your submission content in advance to ensure that you can correctly generate a bitstream that passes the test during the defense。
    - **Attention: There is only one defense for the board test. Please ensure that the board test can be conducted before scheduling a defense time. Individual test cases that fail the test can be debugged on site, and the score for application function points will be multiplied by 0.5. If it is impossible to board during the defense, the test score will be based on the simulation case. Simulation test score=on-board test score * 0.3, simulation test bench needs to be provided by oneself.**
    - **Attention: The course team will conduct a plagiarism check on all Verilog and ASM codes after the defense.**
    - The name format of the compressed file is: C defense time _ list of group members' names

  e.g. c160156_A_B_C (Among them, c160156 represents the code submission for the defense during 56 classes on Monday, 16th week. A, B, and C are the names of three teammates)

  - **2nd submission: document and vedio** (submit to BB site before 12:00 noon on Monday of the week17)
    - Document (in PDF format), Document Name: D Defense Time _ List of Group Member Names. If there is no video, only the document can be submitted
    - vedio (Only record the bonus section, groups that have not implemented the bonus do not need to submit it) :
      - Please upload the PDF document and MP4 video separately: named as d defense time_ group member name list, v defense time_ group member name list
      - The recommended video size should not exceed 500M (if it exceeds this size, please handle it yourself before uploading), and the video must include the part of the bonus board demonstration.

Rating Explanation (1-1) Basic Score

- Basic score: Basic functions (70)+Code specifications (3)+Documents (12)+Personal defense score (15 points=10 points for mid-term defense+5 points for final defense)
  - mid-term defense (10) :
    - 1) Single cycle CPU basic component OJ score (2)
    - 2) Mid-term defense document (4): Please fill in the mid-term defense requirements in page 9 carefully. If one item is missing or filled in randomly, 0.5 points will be deducted, up to a maximum of 4 points.
    - 3) On site 3-minute defense score (4): Key part code display+submodule testing/integration testing result display, score based on defense performance.
    - ATTENTION!!!: Individual participation is required for the mid-term defense, and students who are not present will receive 0 points for the midterm defense.
  - Basic functions (70) :
    - The implemented single cycle CPU can be expanded in terms of functionality and instruction set, enabling on-board testing (70)
      - In the CPU code tested on the board, it is required to comply with the code specifications agreed upon in the mid-term defense document (structured design, unified naming conventions, annotation requirements, definition and use of symbolic constants, etc.)
      - Tested through Basic Scenario 1 and Basic Scenario 2
      - Use peripherals as required: buttons (function buttons such as data confirmation), dip switch (data input), LED (result display), and seven segment digital display tube (result display). If IO is not used according to the requirements of the question, 50% of the score for this case will be deducted. Please note that the IO address cannot use the address provided by the lab courseware and needs to be specified by oneself.
      - If the code cannot be tested on the board and passes the on-site modification test, the test case score will be multiplied by 0.5; If it cannot be tested on the board, simulation testing will be used, with a score of 0.3 for simulation test cases.
  - final defense (5) :
    - No need for PPT, conducted in the form of Q&A. The judges will use the mid-term defense materials to verify the implementation plan. If the plan does not match, sufficient reasons must be provided.
- Code Specification Description (3) :
  - The overall code specifications of the project (such as structured design, unified naming conventions, annotation requirements, definition and use of symbolic constants, etc.) should be determined at the early stage of project development. This agreement should be reflected in the mid-term defense statement document and strictly enforced during the development process.
  - If the code specifications formulated during the mid-term defense are not reflected in the final defense code, the code specifications during the defense will be deducted points, and the code specifications in the mid-term defense self description document will be divided into 0 points.

Rating Explanation (1-2) Basic Score-document

- Developer Description: Each member's student ID, name, responsibilities, and contribution percentage.
- Development plan schedule and implementation status, version modification records (optional)
- CPU architecture design
  - CPU features:
    - ISA (Include all instructions (instruction names, corresponding codes, usage), reference ISA, and updates or optimizations made based on the reference ISA for this project; Information such as register width and number) ; exception handling。
    - CPU clock, CPI, a single cycle or multi cycle CPU or pipeline (such as support, what level of pipeline is it, and what method is used to solve pipeline conflicts)。
    - Addressing Space Design: Van Neumann structure or Harvard structure; Addressing unit, size of instruction space and data space, base address of stack space.
    - Support for external IO: whether to use separate instructions (and corresponding instructions) to access peripherals or MMIO (and corresponding addresses of related peripherals), and whether to use polling or interrupt to access IO.
  - CPU interface: clock、reset、uart interface、other I/O interface。
- Solution analysis explanation: Analyze the hardware and software solutions for the same function implementation separately, compare the functionality, performance, and other aspects through experiments, provide the comparison results, and explain the final selected solution. For example, the difference between hardware and software in implementing floating-point operations (5)
- Instructions for using the system board: Instructions for input and output operations related to system operations on the development board. (such as the input device used for resetting, how to achieve resetting; the buttons for switching CPU working modes and how to select modes; Observation area of output signal, corresponding relationship with output data, etc (1)
- Self testing instructions: List the testing methods (simulation, on-board), testing types (unit, integration), test case descriptions (excluding this document and OJ), and test results (pass/fail) in a table format; And the final test conclusion. (1)
- The inspiration and assistance of open source and AI for this project: If network code resources were used during the development process or relevant code was generated with the help of AI, please explain what help and inspiration these codes provided, what problems were encountered, and corresponding solutions. (2), if fully self-developed (without referencing any network resources or AI generated code), please provide relevant explanations based on the experimental courseware plan.
- Problem and Summary: Problems encountered, thoughts, and summaries during the development process. (3)
- Attention: Please complete the document according to the requirements. The document does not need to be lengthy.

Rating Explanation (2-1) bonus score

【max: 15】 Including but not limited to:

- 1) **Implement support for complex peripheral interfaces (such as VGA interface, keyboard interface, etc.)【max: 2】**
  - NOTEs: In this course, only access to complex peripheral interfaces implemented through software and hardware collaboration is supported (i.e., accessing related complex peripherals through corresponding instructions or address information in instructions, rather than just using hardware control to implement the use of complex peripherals). This course's bonus focuses more on CPU architecture optimization or application scenario implementation, therefore VGA, keyboard, or other peripheral implementation effects are not considered in the bonus score.
- 2) **Implement one-time burning of FPGA chip, which can switch between multiple testing scenarios through UART interface【max: 2】**
- 3) **Implemented ISA instruction extensions, such as auipc (1 point), ecall (1 point for each ecall function implemented, up to a maximum of 2 points), etc. Test cases need to be provided by oneself, and the modifications made to the CPU structure compared to the implementation of the extension instruction in class should be explained. If there is no such explanation, no points will be awarded.【max: 5】**
- 4) **A single cycle CPU based on the RISC-V32I introduced in the experimental class, implementing new design ideas (such as pipeline)【max: 10】**:
  - If implementing a pipeline, it is also necessary to implement a single cycle CPU and demonstrate CPU performance improvement based on the same test case (such as a loop).【1~4】
  - Able to pause and control clock cycle progression through debug mode, in order to observe register values and hazard resolution in real-time【2】
  - Correctly run code snippets containing control hazards, one or more data hazards, provide assembly code for testing, and pass the test.【1~4】
- 5) **Implementation or application of CPU based software hardware collaboration【max: 5】**
  - Self created small tools in the CPU software hardware collaboration toolchain, such as extension assembly tools that support custom extension instructions to be implemented on the current CPU (1 score), A tool for creating COE files with adjustable ROM/RAM size that can be matched and generated (1 score), Hardware Implementation with Adjustable UART Rate (1 score) or Communication tools (1 score) etc.
  - Self created software applications that can communicate and cooperate with the CPU, such as graphics processing, sound processing, upgraded game controllers, etc. (1-4 points depending on complexity)。
- NOTES:

  1) If implementing a CPU based on LoongArch or ARM instructions, there is no need to implement an RISC-V CPU again (the function score and bonus score points are shifted to LoongArch instructions/ARM instructions), and the bonus score is multiplied by a coefficient of 1.05-1.1 based on the completion status (but the bonus score does not exceed 15 points). Please note that the relevant documents must compare the differences between the implementation of this architecture and the RISC-V implementation introduced in class, otherwise the relevant bonus will be 0 points。

Rating Explanation (2-2) bonus score : document and vedio

- NOTES:
  - 2) The implementation of bonus should include corresponding code, demonstrations, documentation, and videos;
  - 3) The document should provide an explanation of the implementation of bonus related functions, including the implementation mechanism, test cases, and test results. The video should also record a demonstration of bonus functions
  - 4) If there is a missing document or video corresponding to the bonus, the bonus score will be discounted by 60% For example, if the bonus function scores 10 points and there is a lack of qualified videos or documents, then the total bonus score=10 * 0.6=6 points.。
- document with bonus
  - Please place the explanation related to bonus in the latter half of the basic function document.
  - Design description of corresponding function points for bonus
    - Design concept and its relationship with surrounding modules
    - Core code and necessary explanations
  - Test description: Test scenario description, test cases, test results and explanation
  - Inspiration and assistance from open source and AI for this major project: If network code resources were used during the development process or relevant code was generated with the help of AI, please explain what help and inspiration these codes provided, what problems were encountered, and corresponding solutions.
  - Problem and Summary: Problems, Reflections, and Conclusions Encountered during the Development of Bond Function Points.
- vedio with bonus:
  - A complete introduction to this major assignment is required in the video, including group members, overall functions, and especially bonus related feature points
  - The main content is: Introduction to the design concept of bonus, functional demonstration and explanation

Mid-term Defense requirements

- Prepare in Advance
  - ✓ Prepare the "Mid term Group Defense Self Statement Document" in advance (printed, A4 double-sided), which should include:
    - ① Group information (member name, student ID, lab class time)
    - ② Code specification (should cover whether structured design is carried out, naming conventions, annotation requirements, definition and use of symbolic constants, etc.)
    - ③ CPU features (single cycle/pipeline) , clk, ISA
    - ④ CPU architecture design (modules and wiring relationships/interface design specifications)
    - ⑤ Relationship between CPU instruction and control signals
    - ⑥ design code and testcase which have been done
    - ⑦ tools plan to use/ develop
    - ⑧ Project progress, current difficulties or issues, expected final defense time, follow-up plans
  - ✓ Preoare the ”personal defence **pp**t“ (Electronic version+A4 paper double-sided printing)
    - ① name, SID and description of personal work in the project
    - ② Screenshot of key content of code under personal responsibility
    - ③ Personal responsibility for testing work
      - ① Testcase form (including case description - such as relevant instructions, test results (passed or failed)
      - ② Screenshots of testbench and test waveforms used for testing, along with necessary explanations.

- defense:
  - ✓ time: in lab class of week 13th
  - ✓ Registration of defense order: Register the defense time and order in the shared document
  - ✓ All members must be present. Members who are not present will receive 0 points in the mid-term defense.
  - ✓ Each student is required to provide a "Mid term Group Defense Self introduction Document" (fill in one copy, group members can make a copy) and a "Personal Defense PPT" (A4 double-sided printing, bound together with the document), and use an electronic version of the PPT to participate in the defense
  - ✓ Personal defense: PowerPoint presentation showcasing personal work in the project, including code, and tests (including test case tables, test results waveform screenshots, and necessary explanations), with strict time control within 3 minutes。
  - ✓ Answer teacher/SA's questions about projects and personal work
- NOTES:
  - The "Mid term Group Defense Self Statement Document" and personal defense PPT will be submitted to the defense reviewer after completing the mid-term defense, as a reference document for the final project defense.

Final Defense requirements

- Prepare in Advanc:
  - Equipment: Please prepare two computers with Vivado installed to participate in the defense (on-site modification of assembly code, burning of FPGA chips, answering questions by comparing the code, and two computers for convenient synchronous testing)。
  - Registration of defense order: Register the defense time and order in the shared document。
- Defense includes:
  - Both demonstration and Q&A sessions require all team members to be present and answer questions。
  - Request on-site modification of assembly source code according to demonstration requirements, complete the complete process of assembly, program distribution, and testing。
  - **Reminder: During the defense, it is necessary to download the project submitted by the group from BB and generate a bitstream file on site under the supervision of the inspector. Please test your submission content in advance to ensure that a testable bitstream can be generated correctly during the defense. Individual case tests that fail can be debugged on-site, and the score for application case functionality will be multiplied by 0.5. Attention: There is only one defense for the board test. Please ensure that the board test can be conducted before scheduling a defense time. If the defense cannot be presented on the board, the test score will be based on the simulation case. Simulation test score=on-board test score * 0.3.**
- During the demonstration, it is necessary to complete the on board (Minisys/EGO1 development board) testing of the CPU as required。
  - Basic testing scenarios for CPUs (see specific content on the following page)
  - Extended functions of CPU (refer to the "bonus function" section in page7)

Basic Test Scenario 1 (Implementation is relatively simple, accounting for 20% of the Function score)

Use the dip switches on the development board for input, with 3 dip switches (x2,.. X0) for inputting test case numbers and 8 dip switches (sw7,.. Sw0) for inputting test data (note that the high bits of dip switches, LEDs, and digital tubes are all on the left side, and the low bits are all on the right side)

NOTES 1: Please follow the instructions in the test case description for input and output. Test case scores that are not completed according to specified requirements=Test case score * 0.5.
NOTES 2: If other inputs (keyboard) or outputs (VGA) are implemented, when testing the bonus, select a test case from this scenario for separate display, but still need to implement the input and output of the basic peripherals required in the document.

test case ID	Description (In test case 3'b011-3'b111, both a and b are in binary complement form)
3‘b000(0)	Input the test number ‘a’, press confirm button to display the value of ‘a’ on the output device (8 LEDs), input the test number b, press confirm button to display the value of ‘b’ on the output device (8 LEDs)
3‘b001(1)	Input the test number ‘a’, load it in a register by ‘lb’ instruction, display the value of the 32-bit register in hexadecimal format on the output device (digital tube), and store the number to memory (in the 3'b011-3'b111 test case, the value of ‘a’ will be read from the memory unit through the ‘lw’ instruction for comparison)
3‘b010(2)	Input the test number ‘b’, load it in a register by ‘lbu’ instruction, display the value of the 32-bit register in hexadecimal format on the output device (digital tube), and store the number to memory (in the 3'b011-3'b111 test case, the value of ‘b’ will be read from the memory unit through the lw instruction for comparison)
3‘b011(3)	Compare test number ‘a’ and test number ‘b’ (from case 1 and case 2) using ‘beq’ instruction. If the relationship holds, light up 8 LEDs. If the relationship does not hold, turn off all 8 LEDs
3‘b100(4)	Compare test number ‘a’ and test number ‘b’ (from case 1 and case 2) using ‘blt’ instruction. If the relationship holds, light up 8 LEDs. If the relationship does not hold, turn off all 8 LEDs
3‘b101(5)	Compare test number ‘a’ and test number ‘b’ (from case 1 and case 2) using ‘bltu’ instruction. If the relationship holds, light up 8 LEDs. If the relationship does not hold, turn off all 8 LEDs
3‘b110(6)	Compare test number ‘a’ and test number ‘b’ (from case 1 and case 2) using ‘slt’ instruction, output the comparison result to the LED through the load command. If the relationship is valid, light up one LED. If the relationship is not valid, turn off the LED
3‘b111(7)	Compare test number ‘a’ and test number ‘b’ (from case 1 and case 2) using ‘sltu’ instruction, output the comparison result to the LED through the load command. If the relationship is valid, light up one LED. If the relationship is not valid, turn off the LED

Basic Test Scenario 2 (Implementation is relatively complex, accounting for 80% of the Function score)

Use the dip switches on the development board for input, with 3 dip switches (x2,.. X0) for inputting test case numbers and 8 dip switches (sw7,.. Sw0) for inputting test data (note that the high bits of dip switches, LEDs, and digital tubes are all on the left side, and the low bits are all on the right side).

NOTES 1: Please follow the instructions in the test case description for input and output. Test case scores that are not completed according to specified requirements=Test case score * 0.5.
NOTES 2: If other inputs (keyboard) or outputs (VGA) are implemented, when testing the bonus, select a test case from this scenario for separate display, but still need to implement the input and output of the basic peripherals required in the document.

Test case ID	Description (The input data for test case 3 'b010-3' b011 is a signed number)
3‘b000	Input an 8-bit number, reverse it, and output it. For example, input 8'b1010_1001. If not reversed, the output should be 8'b1010_1001. If reversed, the output should be 8'b1001_0101 (output the result on 8 LEDs)
3‘b001	Input an 8-bit number to determine if it is a binary palindrome (see Practice2 in Lab3). If it is a binary palindrome, turn on a led, else turn off the led.
3‘b010	Input two IEEE754 encoded floating-point numbers with a width of 12 bits (the highest bit is the sign bit, the exponent is 3 bits, and the number of bits is 4 bits), with the top 8 bits (the bottom 4 bits are default to 0), and store these two floating-point numbers in memory. After inputting the first floating-point number ‘a’, press confirm button to display the decimal form of the integer part of ‘a’ on the digital tube. After inputting the second floating-point number ‘b’, press confirm to display the decimal form of the integer part of ‘b’ on the digital tube.
3‘b011	Read out the two test data from test case ID 3'b010 in this scenario from the memory, add the two test data, and display the integer part of the sum in decimal format on the digital tub.
3‘b100	Input 4-bit raw data, generate its corresponding cyclic redundancy check code (CRC-4: X^4+X+1), and concatenate the input data with the check code (the input data is in the high position, and the check code is in the low position). For example, if the input data is 4'b1000, the check code generated based on CRC-4 is 4'b1010, and the final output is 10001010. Display the binary form of the results on 8 LEDs.
3‘b101	Input 8-bit raw data, perform (CRC-4: X^4+X+1) verification based on the input 8-bit data, and display whether the verification passes or fails through one LED light. For example, if you enter 8'b10001010 and the verification passes, one LED light will turn on. If you enter 8'b10001011 and the verification fails, the LED light will turn off.
3‘b110	Please design your own test case to test whether the “lui” instruction is effective on your CPU. Use a digital tube for output. (If this instruction is not designed in other ISAs implemented, consider implementing similar functionality in other ways and testing)
3‘b111	Please design your own test cases to test whether “jal” and “jalr” instructions are effective. (If this instruction is not designed in other ISAs implemented, consider implementing similar functionality in other ways and testing)