CS 331: Computer Organization and Architecture
Technology Research Project Assignment

Date: April 30, 2023

This will be based on selected topics in Computer Organization and Architecture. The essence of this assignment is to enable students to explore relevant contemporary computer organization and architectural issues on their own, which otherwise would not have been covered due to time and other constraints. It also inculcates the attitude of researching technology issues in students and prepares them to meet real-life situations which may require them to learn about totally new and emerging technologies and make presentations on them and use them. All presentation slides from the class (various teams) will be shared with the class to be used as a learning resource.

1. Energy efficiency and power consumption
2. Security and privacy
3. Cloud computing and virtualization
4. Distributed systems and parallel computing
5. Internet of Things (IoT) and embedded systems
6. Artificial intelligence and machine learning
7. Quantum computing
8. Wearable computing and augmented reality
9. Big data processing and analytics
10. Edge computing and fog computing
11. Blockchain and cryptocurrency
12. Neuromorphic computing
13. Cyber-physical systems
14. High-performance computing
15. High-speed communication and networking
16. Programmable logic devices
17. Fault-tolerant computing
18. Reconfigurable computing
19. Memory hierarchy and storage systems
20. Performance evaluation and benchmarking
21. Compilers and programming languages
22. Instruction set architectures.
23. Multi-core and many-core systems
24. Graphics processing units (GPUs)
25. Heterogeneous computing
26. Microarchitectural vulnerabilities (e.g., Spectre, Meltdown)
27. Microservices and containerization
28. Cloud-native architectures
29. Virtual memory management
30. Processor design and optimization
31. Cache coherence and consistency protocols
32. Input/output (I/O) subsystems and interfaces

33. Interconnect architectures
34. Instruction-level parallelism (ILP)
35. Branch prediction and control flow
36. Register renaming and out-of-order execution.
37. Memory consistency models
38. Transactional memory
39. Non-volatile memory technologies (e.g., flash memory, phase-change memory)
40. Hybrid memory systems
41. Network-on-Chip (NoC) architectures
42. Hardware acceleration for data-intensive applications (e.g., FPGAs)
43. Neural processing units (NPUs)
44. Energy-aware computing
45. Application-specific architectures

Some IEEE journal citations for each of the research themes provided above:

1. Energy-efficient computing:
   - IEEE Transactions on Sustainable Computing
   - IEEE Transactions on Very Large Scale Integration (VLSI) Systems
2. Memory hierarchy design:
   - IEEE Transactions on Computers
   - IEEE Transactions on Very Large Scale Integration (VLSI) Systems
3. Multi-core and many-core architectures:
   - IEEE Transactions on Parallel and Distributed Systems
   - IEEE Transactions on Computers
4. Parallel and distributed computing:
   - IEEE Transactions on Parallel and Distributed Systems
   - IEEE Transactions on Cloud Computing
5. Cache coherence protocols:
   - IEEE Transactions on Computers
   - IEEE Transactions on Parallel and Distributed Systems
6. Interconnect technologies:
   - IEEE Transactions on Very Large Scale Integration (VLSI) Systems
   - IEEE Transactions on Parallel and Distributed Systems
7. Virtualization and hypervisors:
   - IEEE Transactions on Computers
   - IEEE Transactions on Cloud Computing
8. Big data processing:
   - IEEE Transactions on Big Data
   - IEEE Transactions on Cloud Computing
9. Cloud computing:
   - IEEE Transactions on Cloud Computing
   - IEEE Cloud Computing Magazine
10. Quantum computing:
    - IEEE Transactions on Quantum Engineering
    - IEEE Journal of Quantum Electronics

11. Microprocessor design:
    - IEEE Transactions on Computers
    - IEEE Transactions on Very Large Scale Integration (VLSI) Systems
12. Microarchitecture design:
    - IEEE Transactions on Computers
    - IEEE Transactions on Very Large Scale Integration (VLSI) Systems
13. Instruction set architecture (ISA) design:
    - IEEE Transactions on Computers
    - IEEE Transactions on Very Large Scale Integration (VLSI) Systems
14. Operating system (OS) support for hardware features:
    - IEEE Transactions on Computers
    - IEEE Transactions on Operating Systems and Software Engineering
15. Performance optimization:
    - IEEE Transactions on Computers
    - IEEE Transactions on Parallel and Distributed Systems

Citations based on key sample thematic areas.

1. Energy-efficient computing

- *IEEE Transactions on Computers, "Energy-efficient computing for real-time systems"*
- *IEEE Transactions on Sustainable Computing, "Energy-efficient computing for sustainable society"*

2. Memory hierarchy design

- *IEEE Transactions on Computers, "Improving performance and energy efficiency of memory hierarchies with intelligent caching."*
- *IEEE Transactions on Parallel and Distributed Systems, "Dynamic memory hierarchy management for parallel systems"*

3. Multi-core and many-core architectures

- *IEEE Transactions on Computers, "Dynamic multicore resource allocation for power efficiency and performance improvement"*
- *IEEE Transactions on Parallel and Distributed Systems, "Efficient many-core architectures for big data processing"*

4. Parallel and distributed computing

- *IEEE Transactions on Parallel and Distributed Systems, "Distributed computing for big data processing"*
- *IEEE Transactions on Cloud Computing, "Parallel and distributed computing in cloud environments"*

5. Cache coherence protocols

- *IEEE Transactions on Computers, "Efficient cache coherence protocols for shared-memory multiprocessors"*
- *IEEE Transactions on Parallel and Distributed Systems, "Cache coherence protocols for heterogeneous many-core architectures"*

Note that these are just examples, and there are many more IEEE journals that publish research on these topics. When searching for research papers, you can use IEEE Xplore Digital Library (https://ieeexplore.ieee.org/) to find papers published in IEEE journals and conferences.

Go to the IEEE Xplore Digital Library and ACM Digital Library and search for articles using relevant keywords and filters related to the thematic area of interest.

- Look for IEEE and ACM conferences related to the thematic area and search for conference proceedings on the IEEE Xplore Digital Library and ACM Digital Library.
- Use Google Scholar to search for IEEE and ACM journal articles related to the thematic area of interest. Google Scholar allows you to filter your search results by publication date, citation count, and other parameters.
- Consult with a librarian or information specialist who can help you conduct a more comprehensive literature search.

Here are the key thematic areas of research related to the 45 topics provided:

1. Hardware Architecture: This includes topics related to the design, implementation, and optimization of hardware components of computing systems, such as microprocessors, memory systems, and storage systems.
2. Parallel and Distributed Computing: This includes topics related to the design, implementation, and optimization of computing systems that can execute multiple tasks simultaneously or across multiple machines, such as multi-core and many-core architectures, interconnect technologies, and network-on-chip design.
3. Energy Efficiency and Power Management: This includes topics related to the development of computing systems that consume less power, generate less heat, and are more energy-efficient, such as power management and thermal management, approximate computing, and low-power memory technologies.
4. Security and Trustworthiness: This includes topics related to the design and implementation of secure and trustworthy computing systems, such as hardware security modules, cryptography hardware, and cloud security and privacy.
5. Emerging Technologies and Applications: This includes topics related to the development of new technologies and applications that push the boundaries of computing, such as quantum computing, neuromorphic computing, machine learning hardware, and biologically-inspired computing.
6. Software and System Design: This includes topics related to the design and implementation of software and operating systems that run on hardware architectures, such as OS support for hardware features, serverless computing, and containerization.
7. Human-Computer Interaction and Ethics: This includes topics related to the interaction between humans and computing systems, such as wearable computing architectures, sensor networks and processing, and HCI. It also includes topics related to the ethical considerations surrounding computer organization and architecture, such as ethics in computer architecture and design and open-source hardware.
8. Performance and energy efficiency: Energy-efficient computing, performance optimization, power management, thermal management.
9. Architectures and systems: Memory hierarchy design, multi-core and many-core architectures, parallel and distributed computing, cache coherence protocols, interconnect technologies, network-on-chip design, 3D chip stacking and packaging, serverless computing, containerization, reconfigurable computing, high-performance computing architectures.
10. Memory and storage: Non-volatile memory technologies, storage systems design, database systems design.
11. Security and trustworthiness: Security and trustworthiness, hardware security modules, cryptography hardware, cloud security and privacy.
12. Emerging technologies: Quantum computing, neuromorphic computing, approximate computing, machine learning hardware, biologically-inspired computing.
13. Application-specific architectures: Mobile computing architectures, wearable computing architectures, real-time and embedded systems design, autonomous systems design, sensor networks and processing, human-computer interaction.
14. Ethics and open-source hardware: Ethics in computer architecture and design, open-source hardware.

These thematic areas reflect some of the major trends and challenges in computer organization and architecture research today. Of course, some of the research topics could fit into multiple thematic areas, and there are other ways to group the topics as well.

Grouping the 45 research topics based on these key thematic areas, we have:

1. Hardware Architecture:

- Memory hierarchy design
- Microprocessor design
- Microarchitecture design
- Instruction set architecture (ISA) design
- Hardware accelerators (e.g., GPUs, FPGAs)
- 3D chip stacking and packaging
- Non-volatile memory technologies (

2. Performance and energy efficiency:

- Energy-efficient computing
- Performance optimization
- Power management
- Thermal management

3. Architectures and systems:

- Memory hierarchy design
- Multi-core and many-core architectures
- Parallel and distributed computing
- Cache coherence protocols
- Interconnect technologies
- Network-on-Chip design
- 3D chip stacking and packaging
- Serverless computing
- Containerization
- Reconfigurable computing
- High-performance computing architectures

4. Memory and storage:

- Non-volatile memory technologies
- Storage systems design
- Database systems design

5. Security and trustworthiness:

- Security and trustworthiness
- Hardware security modules
- Cryptography hardware
- Cloud security and privacy

6. Emerging technologies:

- Quantum computing
- Neuromorphic computing
- Approximate computing
- Machine learning hardware
- Biologically-inspired computing

7. Application-specific architectures:

- Mobile computing architectures
- Wearable computing architectures
- Real-time and embedded systems design
- Autonomous systems design
- Sensor networks and processing
- Human-computer interaction

8. Ethics and open-source hardware:

- Ethics in computer architecture and design
- Open-source hardware

DELIVERABLES

- **Abstract [*4*]**
- **Introduction [*5*]**
- **The main content and organization: This may be organized into appropriate subtopics. [*21*]**

Performing a literature review of a journal article involves critically analyzing and summarizing the existing research on a specific topic. Here are some steps to follow:

1. Read the article thoroughly: Start by reading the journal article carefully, and making notes on the key points, arguments, and findings presented.
2. Identify the research question or topic: Determine the main research question or topic that the article addresses. This will help you focus your literature review and identify relevant studies.
3. Conduct a search for related literature: Use databases such as Google Scholar, JSTOR, or PubMed to search for related literature on the topic. Look for studies that support or contradict the claims made in the article you are reviewing.
4. Evaluate the sources: Read through the studies you have found and evaluate their quality and relevance. Check that they are recent, peer-reviewed, and published in reputable journals.
5. Summarize the literature: Organize the literature you have found into categories or themes, summarizing the main findings and arguments of each study. Be sure to include both supporting and opposing viewpoints.
6. Compare and contrast the studies: Analyze the similarities and differences between the studies you have reviewed, noting any gaps or inconsistencies in the research.
7. Identify areas for future research: Based on your analysis of the literature, identify areas for future research and suggest how the article you are reviewing can contribute to this research.
8. Write the literature review: Finally, write your literature review, organizing your summary and analysis of the literature into a clear and concise narrative that supports your argument and research question.

Remember to cite all sources properly and adhere to any formatting or citation requirements specified by your instructor or the journal you are submitting to.

- **Conclusion and Recommendations: [*5*]**
- **Overall writing showing *clarity, conciseness, concreteness, coherence, and context*, coupled with correct referencing, will be rewarded. [*5*]**
- **References**

- **You are required to work in groups of 2 (required).**

GRADING CRITERIA

HOLISTIC RUBRIC FOR TECHNICAL REPORT ASSESSMENT

Here's a grading rubric for a literature review report:

1. Organization (20%)

- The report has a clear structure, with an introduction, body, and conclusion.
- The report is well-organized, with ideas presented in a logical and coherent manner.
- The report has clear and concise headings and subheadings that help guide the reader.

2. Research (30%)

- The report demonstrates a thorough understanding of the research topic.
- The report includes a comprehensive and diverse range of sources, with a mix of primary and secondary sources.
- The report is up-to-date, with recent research included

3. Analysis (30%)

- The report provides a critical analysis of the literature, with key themes, similarities, and differences identified and discussed.
- The report demonstrates the ability to synthesize and integrate information from different sources.
- The report offers a clear and well-supported argument, with evidence drawn from the literature.

4. Writing and Presentation (20%)

- The report is well-written, with clear and concise sentences and appropriate grammar and spelling.
- The report is visually appealing, with appropriate headings, formatting, and referencing.
- The report adheres to any formatting or citation requirements specified by the instructor or the target journal.

The grading rubric can be customized to suit the specific requirements of the literature review assignment and can be adjusted to include additional criteria or to assign different weights to each category based on the instructor's preferences.

GROUP PROJECT PRESENTATION EVALUATION RUBRIC:

Criteria	Excellent (5)	Very Good (4)	Good (3)	Fair (2)	Unsatisfactory (1)
Content	The content is clear, thorough, and accurate, and demonstrates a deep understanding of the topic. All group members contribute equally to the presentation	The content is mostly clear and accurate, with some minor gaps or oversights. All group members contribute to the presentation, but some more than others	The content is generally clear and accurate but lacks depth or has significant gaps. Some group members contribute more than others	The content is confusing or inaccurate and shows a limited understanding of the topic. Some group members contribute very little or not at all	The content is unclear, inaccurate, or completely irrelevant to the topic. No group members contribute, or the group is incomplete
Organization	The presentation is well-organized and easy to follow, with a clear introduction, body, and conclusion. All group members participate in the organization and delivery of the presentation	The presentation is generally well-organized but may lack a clear structure or flow in some areas. All group members participate in the organization and delivery of the presentation, but some more than others	The presentation is somewhat disorganized, with unclear transitions or confusing structure. Some group members are not fully engaged in the organization and delivery of the presentation	The presentation is poorly organized, with little coherence or logical structure. Some group members are not engaged in the organization and delivery of the presentation	The presentation is completely disorganized or lacks any discernible structure. No group members are engaged in the organization and delivery of the presentation
Delivery	The group is confident, engaging, and speaks clearly and audibly, using appropriate body language and eye contact. All group members participate in the delivery of the presentation	The group is generally confident and engaging but may lack some clarity or use distracting body language. All group members participate in the delivery of the presentation, but some more than others	The group is hesitant or monotone at times, with some difficulty being heard or maintaining audience attention. Some group members are not fully engaged in the delivery of the presentation	The group is very nervous, difficult to hear or understand, or uses distracting or inappropriate body language. Some group members are not engaged in the delivery of the presentation	The group is completely unprepared, speaks inaudibly, or shows complete disinterest in the presentation. No group members are engaged in the delivery of the presentation
Visual Aids	The visual aids are well-designed, appropriate, and effectively support the	The visual aids are generally appropriate but may lack clarity or be somewhat	The visual aids are confusing or irrelevant or detract from the content of the	The visual aids are completely irrelevant, confusing, or detract significantly from the	No visual aids are used, or they are completely unusable or inappropriate. No group

Criteria	Excellent (5)	Very Good (4)	Good (3)	Fair (2)	Unsatisfactory (1)
	content of the presentation. All group members participate in the creation and use of the visual aids	distracting. All group members participate in the creation and use of the visual aids, but some more than others	presentation. Some group members are not fully engaged in the creation and use of the visual aids	content of the presentation. Some group members are not engaged in the creation and use of the visual aids	members participate in the creation or use of visual aids
Collaboration	The group works together seamlessly, with each member contributing equally to the project and presentation. The group communicates well and resolves conflicts effectively	The group works well together, with each member contributing to the project and presentation. The group communicates effectively and resolves conflicts with minimal issues	The group works together, but some members may contribute less than others or have some communication or conflict-resolution issues	The group has some difficulty working together, with some members not contributing or communicating effectively, or experiencing conflict that is not resolved effectively	The group is completely unable to work together, with members not contributing or communicating, or experiencing significant conflict that is not resolved at all
Overall Impression	The presentation is outstanding in all aspects, and demonstrates a high level of preparation and professionalism	The presentation is very good overall, but may have some minor flaws or areas for improvement	The presentation is generally good, but may have significant areas for improvement or lack a polished finish	The presentation is below average, with significant flaws or weaknesses that detract from its effectiveness	The presentation is completely unsatisfactory in all aspects, and demonstrates a lack of preparation or professionalism

Guidelines on Giving Effective Presentations

A good technical presentation has the following properties:

- Good organization
  - At the most basic level, you need an introduction, the body, and a conclusion.
- Appropriate technical content, delivered at the appropriate depth and in a clear and concise manner.
  - When you are delivering a talk to a technical audience (like to our class) the audience wants and needs to know the technical details of your work to fully understand and assess your work. This is the "meat" of your presentation — don't leave it out! Make it absolutely clear what work is your own, and what you are leveraging from others (e.g., libraries you are using)
  - Obviously, you need to take into consideration the time available in deciding the appropriate depth.
  - The fact that you are providing technical details does not mean you should lose the audience in jargon. Think about *how* to present the work for the audience to understand. The next point can be helpful in this regard.
- Useful and high-quality visual aids
  - Your slides should well organized, attractive, and make appropriate use of images, figures, and text. Convey the most important information and avoid excessive text.
  - Use appropriate visual aids to help convey technical details: images, graphs, diagrams, charts, images, even animations. The use of videos is also sometimes helpful.
  - Avoid visual distractions: animated text and special effects are usually distracting and unnecessary.
  - Ensure consistency in your slides: titles, font sizes, etc. Ensure that any items in a bulleted list are grammatically consistent and use consistent punctuation. E.g., You can have a list of nouns, a list of adjectives, a list of verbs, a list of phrases, or a list of sentences. You should not have a list in which some items are nouns and other are sentences, or in which some items end with a period (full-stop) and others don't.
  - Number your slides to make it easy for a member of the audience to refer to a specific slide when asking a question.
- Well-rehearsed but natural delivery
  - Think ahead of time about who is presenting what, what you want to say on each slide and how you want to say it, and **practice**!
  - **Do not read from your phone!**
  - Although your presentation should be well-rehearsed, you should deliver it in as natural a manner as possible, well-paced, with appropriate pauses in speech, not memorized and recited.
  - Practice and ensure that you can give the presentation in the alloted time without rushing but still conveying all the information you need to convey.

Grading Scale:

Grade	Score	Grade Point	Description
A+	**85-100**	**4.00**	**Excellent**
A	**80-84**	**4.00**	**Excellent**
B+	**75-79**	**3.50**	**Very Good**
B	**70-74**	**3.00**	**Very Good**
C+	**65-69**	**2.50**	**Good**
C	**60-64**	**2.00**	**Satisfactory**
D+	**55-59**	**1.50**	**Pass**
D	**50-54**	**1.00**	**Pass**
E	**Below 50**	**0.00**	**Fail**
I	**---**	**---**	**Incomplete**

Explanation of Grading Criteria:

1. The thoroughness of research and analysis: The project should demonstrate a comprehensive understanding of the topic through thorough research and analysis of relevant data and sources.
2. A clear understanding of the topic and focus of the project: The project should have a clear and concise understanding of the topic and focus.
3. Quality and relevance of information presented: The project should present high-quality, relevant information that supports the topic and purpose of the project.
4. Originality and creativity: The project should demonstrate originality and creativity in the topic and information presentation approach.
5. Coherence and structure of the project: The project should have a logical and coherent structure that effectively communicates the purpose and content of the project.
6. Clarity and precision of writing: The project should be well-written with clear and precise language that effectively communicates the content of the project.
7. Effectiveness of visual aids (if any): The project should effectively utilize visual aids, such as charts, graphs, or diagrams, to support the content of the project.
8. Delivery and communication skills: The project should be effectively presented, with clear and articulate delivery, and demonstrate effective communication skills.

General Project

CPU: ALU and Registers

- In this project, you will be using Logisim Evolution to implement a simple **16-bit processor (CPU)** using ALU and Registers connected by data and address buses, similar to your design in Lab 5. Good news - **You are allowed to use Logisim's built-in blocks for all parts of this project to enable faster implementation of the project!**

IMPORTANT INFO - PLEASE READ

- **Read through the whole project before you begin.**
- **PLEASE USE LOGISIM-EVOLUTION TO DESIGN YOUR CIRCUITS. We do not want to load different Logisim versions with different feature designs in another.**
- **Please start the project at your earliest convenience and see the FI if any of the instructions are not clear.**
- **You are allowed to use any of Logisim's built-in blocks for all parts of this project.**
- **Save often.** Logisim can be buggy and the last thing you want is to lose some of your hard work. There are students every semester who have had to start over large chunks of their projects due to this.
- Approach this project like you would any coding assignment: construct it piece by piece and test each component early and often!
- Because the files you are working on are not plain code and circuit schematics, they can't really be merged. **DO NOT WORK ON THE SAME FILE IN TWO PLACES AND TRY TO MERGE THEM. YOU WILL NOT BE ABLE TO MERGE THEM AND YOU WILL BE SAD.**

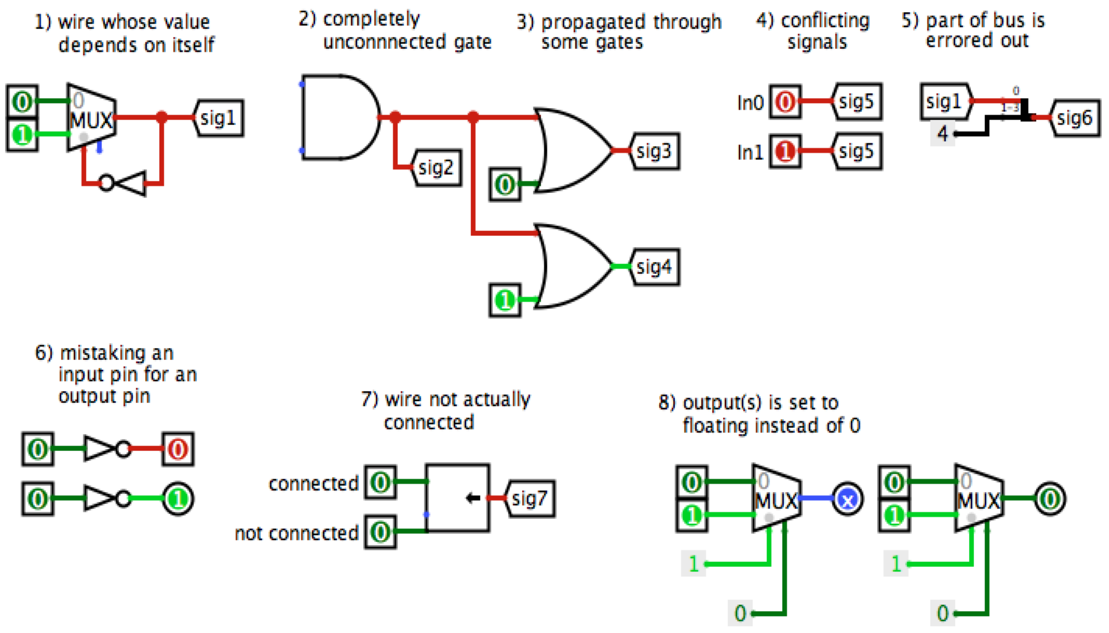
Logisim Notes

If you are having trouble with Logisim, ***RESTART IT and RELOAD your circuit!*** Don't waste your time chasing a bug that is not your fault. However, if restarting doesn't solve the problem, it is more likely that the bug is a flaw in your project. Please see the FI about any crazy bugs that you find, and he will help you investigate.

Things to Look Out For

- Use Library Reference! It is under Help tab, and it tells you the specifications of each built-in component.
- **BE CAREFUL with copying and pasting from different Logisim windows.** Logisim has been known to have trouble with this in the past.

- Changing attributes *before* placing a component changes the default settings for that component. So, if you are about to place many 16-bit pins, this might be desirable. If you only want to change that particular component, place it first before changing the attributes.
- When you change the inputs & outputs of a sub-circuit that you have already placed in `main`, Logisim will automatically add/remove the ports when you return to `main circuit`, and this sometimes shifts the block itself. If there were wires attached, Logisim will do its automatic moving of these as well, which can be extremely dumb in some cases. Before you change the inputs and outputs of a block, it can sometimes be easier to first disconnect all wires from it.
- Error signals (red wires) are obviously bad, but they tend to appear in complicated wiring jobs such as the one you will be implementing here. It's good to be aware of the common causes while debugging:


1) wire whose value depends on itself
MUX
sig1
2) completely unconnnected gate
sig2
3) propagated through some gates
sig3
sig4
4) conflicting signals
In0
In1
sig5
sig5
5) part of bus is errored out
sig1
sig6
4
6) mistaking an input pin for an output pin
7) wire not actually connected
connected
not connected
sig7
8) output(s) is set to floating instead of 0
MUX
MUX

Part 1: Arithmetic Logic Unit (ALU)

Your first task is to create a **16-bit ALU** that supports arithmetic and relational operations. Kindly find below the list of **arithmetic operations** that you need to implement (along with their associated **selector bit switch values**). **You are allowed and encouraged to use built-in Logisim blocks to implement all the arithmetic operations.**

Arithmetic Switch Value	Instruction
0	add: Result = A + B
1	Bitwise and: Result = A & B
2	Bitwise or: Result = A | B
3	xor: Result = A^B
4	div: Result = A / B

5	Rem (mod): Result = A % B
6	mult: Result = A* B
7	sub: Result = A - B
8	Shift logical left for A by 2: A << 2
9	Shift logical left for B by 2: B << 2
10	Shift logical right for A by 2: A >> 2
11	Shift logical right for B by 2: B >> 2
12	2’s complement of A: -A
13	2’s complement of B: -B

Additionally, find below the list of **arithmetic operations** that you need to implement (along with their associated **selector bit switch values**).

Relational Switch Value	**Instruction**
0	Greater than: A > B
1	Equal: A & B
2	Greater or Equal: A >= B
3	Less than: A < B
4	Less or Equal: A <= B
5	Not Equal: = A! = B

Below are the input names, their bit width and descriptions:

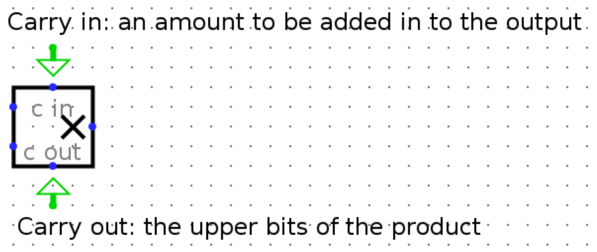
Input Names	**Bit Width**	**Description**
A	16	Data to use for Input A in the ALU operation.
B	16	Data to use for Input B in the ALU operation.
AUSel	4	Selects which arithmetic operation the ALU displays
RUSel	3	Selects which relational operation the ALU displays

Below are the output names, their bit width and descriptions:

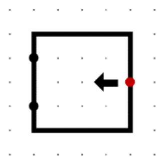
Output Name	**Bit Width**	**Description**
AUResult	16	Result of the ALU Arithmetic Operation.
AUDout	16	Copy of the AUResult as Dout.
RUResult	1	Result of the ALU Relational Operation.

Hints:

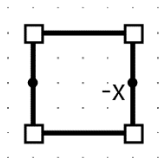
- **For instance,** **`Adder`** is already made for you, and feel free to use a similar structure when implementing your other blocks.
  - When implementing `mult`, notice that the multiply block has a "Carry Out" output (the adder block also has this, but you will not need this) located here:


Carry in: an amount to be added in to the output
c in
c out
Carry out: the upper bits of the product

- You can hover your cursor over an output/input on a block to get more detailed information about that input/output.
- The reminder bit input of the divider can be used to realize the mod (%) operation.
- The Shifter block (in the image below) is found in the Arithmetic folder and can be used to implement the Shift Logical Operations. Ensure that the property is set to "Logical Left" or "Logical Right" to realize your desired operation.



- The Negator block (in the image below) is also found in the Arithmetic folder and can be used to generate the Negation (2's complement) values of data bits.

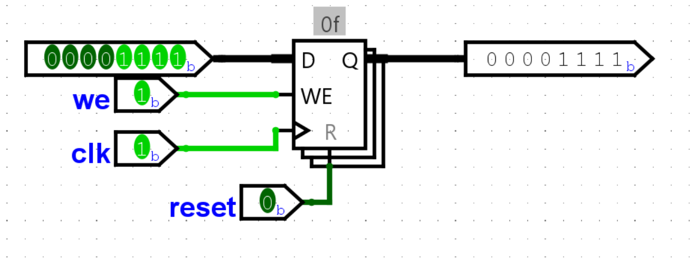


- A multiplexer (MUX) might be useful when deciding which block operation you want to output.
  - In other words, consider simply processing the input in all blocks, and then outputting the one of your choices.

- **NOTE:** Your ALU sub-circuit should be named ***project_alu.*** Your Arithmetic Operation Circuit should be named as ***project_ao*** while your Relational Operation Circuit should be named as ***project_ro.***

Part 2: Registers

Again, you are allowed to use any of Logisim's built-in blocks for all parts of this project. For instance, an implementation of the register for storing any number of bits can be found in the Memory folder in Logisim. For instance, the image below illustrates an 8-bit register storing data.



Part 3: Designing 16-bit CPU.

The CPU design is an extended implementation of Lab 5; hence, you will be able to easily design the CPU if you comfortably designed the CPU in Lab 5. The CPU will have the following components, connected by an internal and external data and address buses:

- Instruction Register (IR) – **16-bits.**
- Arithmetic Logic Unit (ALU) – **16 bits.**
- Output Register – holding the output (result from ALU) after processing – **16 bits.**
- Register A – for storing the value for A – **16 bits.**
- Register B – for storing the value for B – **16 bits.**
- Memory Address Registers (MAR) - the CPU registers that store the memory address to which data will be sent and stored via system bus – **8 bits.**
- Memory Buffer Register (MBR) – comprises the data to be written in memory (write operation) or it obtains the data from memory (read operation) or stores the data being transferred to and from the immediate access store – **16 bits.**

Steps – As you did in Lab 5

- Get two registers and ALU. Label registers as register A and B. Connect the stored data output of each register to A and B input pins in the ALU.
- Connect clock input to each register clock pin. We will use Logisim's inbuilt clock feature (Wiring folder) and automate the switching.
- Connect WE pin to register A
- Connect another WE pin to register B.
- Connect the Din of each register to the internal data bus using a splitter.
- Connect OE pin, AU selector, LU selector inputs and output pins to the ALU.
- Connect output of the ALU to the internal data bus using the splitter

Get another register and label as **Output Register**.

- Connect clock, WE and OE to the register.
- Connect the Din of the output register to the internal data bus using the splitter.
- Connect the Dout of the output register to the internal data bus using the splitter.
- Connect output to the stored data pin of output register.

Get another register and label as **Instruction register (IR).**

- Connect clock, WE and OE to the register.
- Connect the Din of the IR to the internal data bus using the splitter.
- Connect the Dout of the IR to the internal address bus using the splitter and bit selector (to select only the **8 MSB** as the address) – See Hint Below
- Connect output to the stored data pin of IR.

Get **Memory Address Register (MAR)** onto the circuit board.

- Connect clock, WE and OE to the register.
- Connect Din of MAR to Internal Address bus using splitter
- Connect Dout of MAR to External Address bus using splitter
- Connect output to the stored data pin of MAR.

Get another register and label as **Memory Buffer Register (MBR).**

- Connect clock, WE and OE to the register.
- Connect Din of MBR to Internal Data using splitter. Connect to External Data bus also.
- Connect Dout of MBR to External Data bus using splitter
- Connect output to the stored data pin of MBR.
- Connect stored output of MBR to controlled buffer then to internal data bus. Connect internal output enable pin to controlled buffer.

Hint:

NB: The data buses should be labelled with LED signals.

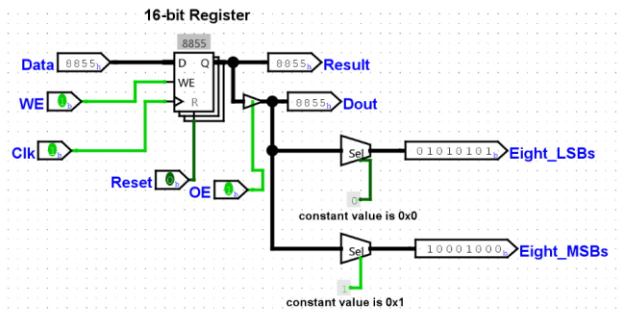
NB: This is similar to an operation in Lab 5 where you selected the 4 least significant bits from an 8-bit input as the memory address using the in-built bit selector and constant value 0.

Let us look at the behaviour of the bit-selector (Plexers folder) and the constant (wiring folder):

Given an input of several bits, this will divide it into several equal-sized groups (starting from the least significant bit) and output the group selected by the constant value.

For example, if we have a 16-bit input 1000100001010101, and we are to have an 8-bit output, then group 0 will be the least significant eight bits 01010101 and group 1 will be the next eight bits, 10001000. (Any bits beyond the top are filled in with 0.) The select input will be a 1-bit constant, whose value can be **0x0** or **0x1**, depending on the group of bits you want to select. Let us illustrate this example:

NB: In Logisim – Evolution, changing the data bits of an input to 16 converts the input's data to hexadecimal, and the toggle feature can be used to generate the hexadecimal data bits. Thus, we can convert our 16-bit binary input into hexadecimal as **8855**.



From the image above, we have stored our 16-bit data into a register. Now, we can select the 8 LSBs as output by using the selector and ensuring that our select input value is 0 (0x0) and select the 8 MSBs as output by ensuring that the select input value is 1 (0x1).

Testing your CPU – As you did in Lab 5

- We need to find a way to place data on the internal data bus to test. One way is to connect 16 bits input (test data) using a splitter and controlled buffer.
- Once you finish the above, go to simulate in the menu bar and make sure the tick frequency is assigned to 1hz and ticks enabled is checked. Switch values in your test data to ensure the test data floats on the data bus.

Testing Register A:

- Enter 16-bit data on the testing input and enable its output. Expected outcome is that the data will float on the data bus.
- Enable WE Control signal for register A. Expected outcome is that the data will be stored in register A. Remember that the data will be stored on the rising edge of the clock.
- Disable WE of register A and OE of testing input data. Expected outcome is that the data is still stored in register A.

Testing Register B:

- Enter 16-bit data on the testing input and enable its output. Expected outcome is that the data will float on the data bus.
- Enable WE Control signal for register A. Expected outcome is that the data will be stored in register A. Remember that the data will be stored on the rising edge of the clock.
- Disable WE of register B and OE of testing input data. Expected outcome is that the data is still stored in register B.

Testing ALU:

- ALU gets values from registers A and B as inputs A and B respectively by default (hardwired). Calculate and compare ALU Result.
- Enable OE signal for ALU. Expected outcome is that the ALU result will now float on the data bus through the ALU Dout.

Testing Output Register:

- Enable WE Control signal for the output register. Expected outcome is that the data (ALU Result) will be stored in output register. Remember that the data will be stored on the rising edge of the clock.
- Disable WE for output register. Then disable OE for ALU. Expected outcome is that the data is still stored in the output register.

<u>Testing Memory Buffer Register (MBR):</u>

- Enable OE for output register. Expected outcome is that the data will float on the data bus.
- Enable WE Control signal for the MBR. Expected outcome is that the data (ALU Result) will be stored in MBR. Remember that the data will be stored on the rising edge of the clock.
- Disable WE for MBR. Then disable OE for Output Register. Expected outcome is that the data is still stored in the MBR.

<u>Testing Instruction Register (IR):</u>

- Enter 16-bit data on the testing input and enable its output. Expected outcome is the data will float on the data bus.
- Enable WE Control signal for the IR. Expected outcome is that the data will be stored in IR. Remember that the data will be stored on the rising edge of the clock.
- Disable WE for IR. Then disable OE for Test input. Expected outcome is that the data is still stored in the IR.

<u>Testing Memory Address Register (MAR):</u>

- Enable OE for IR. Expected outcome is that the data in IR will float on the bus.
- Enable WE signal for MAR. Expected outcome is that the 8 most significant bits will be stored in the MAR.
- Disable WE for MAR. Then disable OE for IR. Expected outcome is that the data is still stored in the MAR.
- Enabling OE for MAR will transport the stored bits to the external address bus.

Recall the main stages of the CPU pipeline:

1. Instruction Fetch
2. Instruction Decode
3. Execute
4. Write Back

A simple illustration of these stages will be observed in our CPU design (can you think of any?).

Stage 1: Instruction Fetch

Here, the 16-bit test data is fetched by the Instruction Register via the test data input bits.

Stage 2: Instruction Decode

Now that we have our instruction coming from the `instruction` input, we have to break it down in the Instruction Decode step. In our CPU design, a simple illustration is the decoding of the bits in our instruction register such that the 8 MSBs refer to the MAR address to which the data should be sent.

Stage 3: Execute

The Execute stage, also known as the ALU stage, is where the computation of most instructions is performed.

Here, you should have already stored the values of A and B in Registers A and B. Pass the stored data from the registers into the ALU circuit to perform the various arithmetic and relational operators discussed above.

Stage 5: Write back.

The write-back stage is where the results of the operation is saved back to the registers. Although not all operations will write back to the register, our processor should write back the results from our arithmetic operations to the output register.

If you have done all the steps correctly, you should have a processor that simply illustrates the four main stages. Good luck!

<u>Resources:</u>

These videos might be helpful:

Designing CPU:

https://www.youtube.com/watch?v=yc_mA7Bjf6E&list=PL26O2JyrmxV7CWzV9hCCJWMyiOlVMIF8z&index=5

Testing the Connections:

https://www.youtube.com/watch?v=GvwjtiMZBTw&list=PL26O2JyrmxV7CWzV9hCCJWMyiOlVMIF8z&index=6

All parts of this project can be implemented using in-built Logisim blocks. Thus, the project can be easily implemented with minimum delay.

Submission and Grading

- This project will be graded in large thorough testing of the circuits. Therefore, ensure that all parts of your sub-circuits work well during testing and produces the desired outputs.
- Ensure that the file submitted contains all required circuits properly connected (save your work frequently). **There will be no excuses tolerated when certain sub-circuits are missing or were not saved before submission.**
- Double check after submission to ensure that the submitted file has all the required components. You are allowed to resubmit where necessary **before the deadline.**
- Again, it is very rare that one team's circuit looks exactly the same as another team's. PLEASE SUBMIT INDIVIDUAL TEAM CIRCUITS. **Violating this policy will result in a zero score for all team members.**
- Additionally, ensure that all team members contribute and understand the design and behavior of the circuits as any team member will be called to explain the behavior and justification of their team's circuit design.

Please pay attention to the submission instruction below:

Name circuit as:

Member 1's firstname_Member 2's firstname_Project.circ

NB: There will be no submissions accepted via email once the slot closes on 30th April, 23:59 GMT.