Theory (Memory, Architectures, Interrupts and Stacks)

1. Review the lecture slides on types of memory and provide a short answer to the following questions (using your own words):

1.1. What is ROM and what is its primary purpose?

ROM stands for Read-Only Memory, which is a storage used to store permanent data on the computer or other electronic devices.

1.2. What is RAM and how is it different from ROM?

RAM (Random access memory) that stores temporary data that can be read and changed in any order is used to store working data. It much faster as compared to ROM as it takes more time to access data on rom because of its static nature.

1.3. What is the difference between static RAM and dynamics RAM?

SRAM requires constant power supply, in turn consuming more power. However, dram offers reduced power consumption, as the information on DRAM is stored in capacitors. But SRAM has lower access time therefore faster than the dram.

1.4. What type of memory is typically used in USB thumb drives? Why shouldn't we rely on this for critical data storage?

A USB thumb drive or a flash drive, like the name suggests these drives use flash memory. It is a type of memory that allows data storage even when power is turned off. This type is slow for reading and writing, has limited capacity therefore should not be used for critical memory storage. It also has a limited lifetime as after a finite number of times of reading and writing it starts ti degrade leading to data loss.

Q2 Consider a computer with 1GB RAM (1024 MB). Given memory addressing is for each byte, how many bits are needed to address all bytes in the system's RAM?

1GB = 2^ (30) bytes

1byte = 8 = 2^ (3) bits

== 2^ (30) * 2^ (3) = 2^ (30+3)

==2^ (33) bits

3. Give a brief description of the Von Neumann and Harvard computing architectures. What are the fundamental differences between the two and for what is is each designed to achieve ?

Von Neumann Architecture- Von Neumann Architecture is a computer architecture that uses a single processor for both processing and data storage. It consists of a CPU, memory, and input/output devices, all connected by a single bus. This architecture is used in most modern computers.

Harvard computing architectures-Harvard architecture is a computer architecture that uses separate memory spaces for instructions and data, allowing simultaneous access to both. It has separate buses for instructions and data, which allows for faster processing of instructions. This architecture is commonly used in embedded systems and digital signal processors.

Von Neumann Architecture	**Harvard Architecture**
Control bits and data bits share a common memory space/hardware.	Instructions and data are kept safe
Processes can be interrupted to store data while other priority tasks are	Immune to buffer overflow attacks

executed and then restored and resumed	
Stack is required. Stacks are implemented in software as well as hardware	Used in digital signal processors where memory is scarce, and speed is important
Common bus for data and instruction transfer	Data Bus and controller busses can be different widths
secure	less secure
More expensive	Less expensive

4. What is cache memory and what is its primary role?

Cache memory is a temporary storage unit that stores frequently used data/instructions in high-speed memory, providing faster and more efficient data retrieval and processing.

5. Explain the concept of an interrupt, and list four common types.

An interrupt occurs when the CPU is processing a task when it is suddenly stopped by another task requested by the user.

The 4 types are

1. Clock interrupt
2. Hardware interrupt
3. Network interrupt
4. Exception interrupt

5.1. Polling is an alternative to interrupts? Briefly explain polling and why it is not commonly used.

Polling is a protocol where CPU checks the state or input of all devices to determine which one has problems. Polling is not widely used as it is time-consuming and wasteful of resources for CPU. If one device freezes, this can make the entire system unresponsive.

6. Explain the general concept of a stack - how do they work, and what is their primary purpose.

A stack is an abstract data type which stores and performs tasks following an oredered,linear sequence. Stack follows the principle of Last In –First Out, or LIFO , where the last element inserted is the first one out or to be performed. Stack acts as a simple yet effective way to store data or recall it

6.1. How are stacks useful for handling interrupts?

Tacks are a useful data structure for handling interrupts because they allow for the efficient storage and retrieval of important information about the state of a program.

One way to do this is to use a stack. The processor can push the current state of the program onto the stack, which effectively saves it for later. It can then handle the interrupt, which may involve executing a separate interrupt service routine (ISR).

6.2. How are stacks useful in programming?

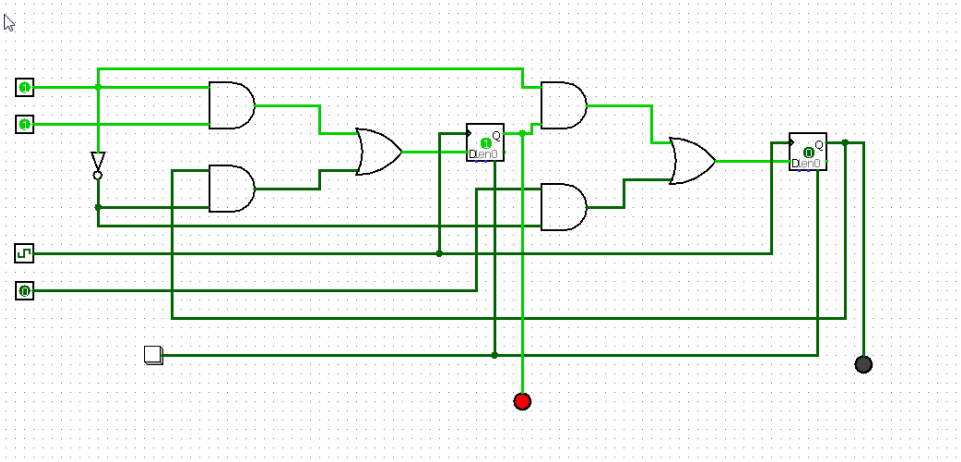
Stacks are useful in programming to store and manage data in an organized and efficient manner. They are a data structure that follows the Last-In-First-Out (LIFO) principle, which means that the last item pushed onto the stack is the first item to be popped off the stack.

7. Start Logisim and open a new canvas

8. Review the lecture slides on building a stack at the top of this lab sheet. We are going tobuild a 5-bit deep, 1-bit wide stack.

9. Start by building a simple shift register that moves bits from one flip flop to the next eachclock pulse. For this you will need a "Data In" pin which sets the next bit to be pushed tothe stack, and a clock to invoke the shifting.

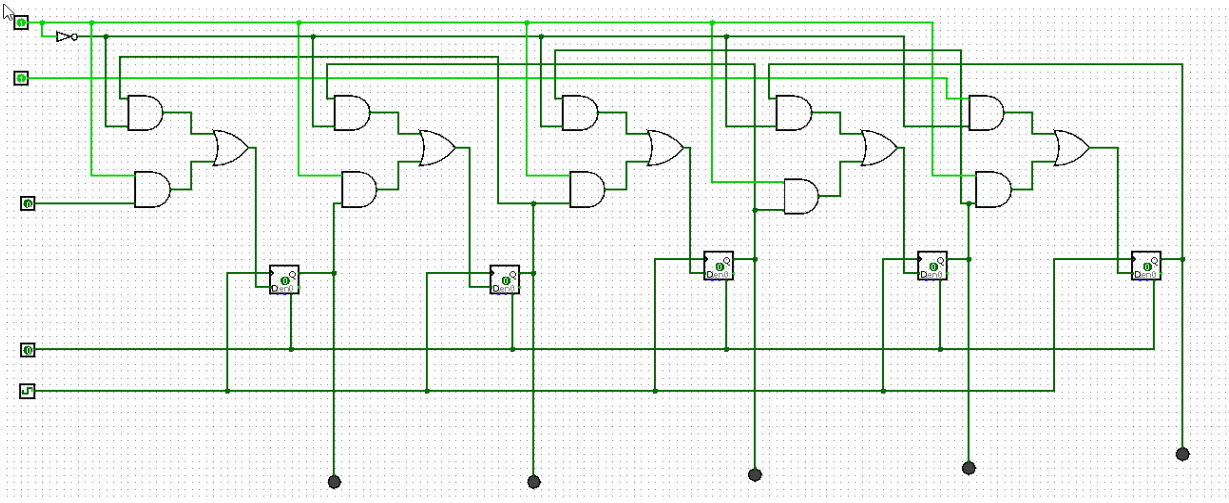
10. For your shift register to work as a stack, it needs to be bi-directional. This means the input to any Flip Flop could come from two places - the left or the right. In lectures we discussed a simple "encoder" circuit that selects which of two data inputs is allowed through,based on a third selection bit. Design the logic for this 2-bit encoder and demonstrate it to your lab demostrator.



11. Now incorporate your encoder above to allow bi-directional shifting of your stack. Yourstack should:

11.1. push and pop bits onto and off the stack, using clock pulses and a direction toggle switch

11.2. show the state of each Flip Flop using LEDs



12. Modify your stack so that it has the option to read out its contents in parallel to a separate register of D Flip Flops. This should only occur when a "stack dump" toggle switch(i.e., pin) is enabled. When the toggle is disabled, the register of D Flip Flops should retain the last state read in (and should have LEDs connected to each Flip Flop out showing its state).