Name : Nguyen Quoc Thang

ID: 104193360 – sws00216

Lab4 : Submission

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

ROM(Read-only memory), that is a type of computer memory that stores data permanently. Data stored in ROM is not lost when the computer loses power, and the data stored in ROM cannot be modified - it can only be read. ROM allows the computer to boot or initialize every time it is powered on or restarted, with the stored data remaining intact.

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

RAM (random access memory) that is used to store data and programs. The data stored in RAM can only be retained for a limited period and is susceptible to loss when the computer abruptly stops functioning. Data stored in RAM can be quickly accessed, processed, and modified by the processor.

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

Static RAM :

1. The access speed is faster than DRAM
2. larger area of silicon per byte, modest power requirement
3. Use in cache memory in computer and mobile device
4. Static RAM is more expensive than DRAM

Dynamic RAM:

1. The access speed is lower than SRAM
2. Smaller area of silicon per byte, low power requirement
3. Use in main memory.
4. Dynamic RAM is cheaper than SRAM.

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

Flash memory(EEPROM) is typically used in USB thumb drives. We should not rely on this for storing critical data because Flash Memory used in USB drives is susceptible to data loss or theft. USB drives lack backup capabilities, so if our data is lost, there may not be a backup copy of the important data.

2. 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 ?

1MB = 2^30 Byte

1byte = 8bit ➔ 2^30 * 8 = 2^33 bit.

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 : Von Neumann is the most common architecture used in modern computers. It consists of four main components: the Central Processing Unit (CPU), memory, input/output

devices, and a bus that connects these components. It includes a shared memory space for both control bits and data bits. The Von Neumann architecture is implemented sequentially, meaning that instructions are executed one by one in a sequential manner.

Harvard computing architectures :

Harvard is a computer architecture in which the memory space for instructions and data is separate. This makes its processing speed faster and more secure because it allows direct access to instructions and data. Additionally, it allows the CPU to simultaneously fetch a new instruction from the instruction memory.

Fundamental differences between Von Neumann and Harvard computing architectures

Von Neumann:

1. In the Von Neumann computing architecture, the memory space is shared for both instruction bits and data bits. However, in the Harvard computing architecture, the instruction bits and data bits use separate memory spaces.
2. In the Von Neumann computer architecture, the CPU fetches new instructions from the instruction memory in a sequential order. On the other hand, in the Harvard computer architecture, simultaneous access to both instructions and data is allowed.

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

Cache is a small memory component located near the CPU and main memory, with the ability to read and store data very quickly. The main role of cache memory is to store and update data to improve system performance. Cache memory is placed near the CPU so that when the CPU needs to access data in the main memory and that data is not in the cache memory, it will go to the main memory to retrieve the data and store it in the cache, which helps to speed up CPU data retrieval since it has already been stored in the cache.

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

Polling is a computer protocol used to check the status of a device. In Polling, the CPU continuously participates in checking the device's status at regular intervals until the device responds. This is done to determine if any errors are occurring in the system. However, polling is not commonly used because it can waste CPU resources and prevent the CPU from performing other tasks.

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

A stack is a linear data structure that follows the Last-In-First-Out principle, which only allows performing additions and deletions at the end, meaning the first item added will be the first one to be deleted. A stack allows us to add data to the stack using push() and remove the last element of the stack using pop().

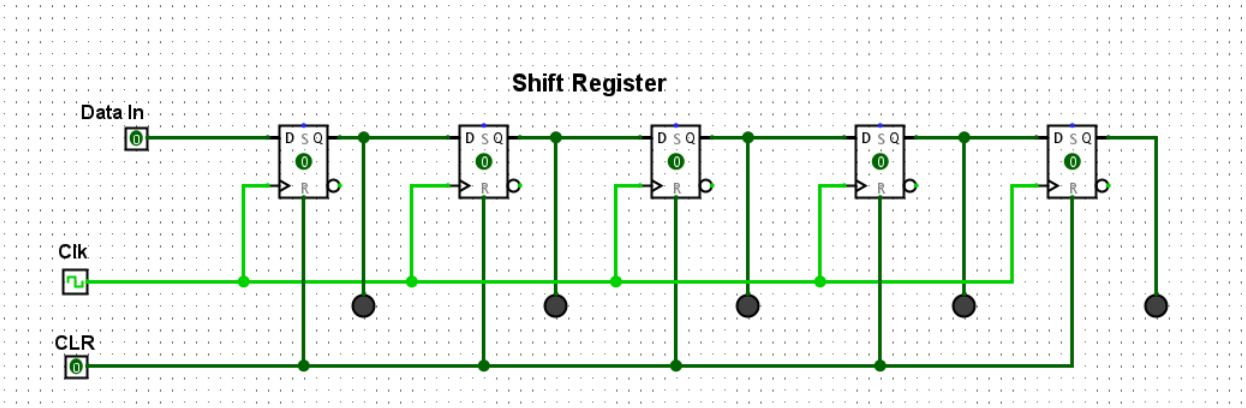
6.1. How are stacks useful for handling interrupts ?

Stacks play a role in maintaining the CPU state when an interrupt occurs. When interrupted, the data will be automatically stored in the stack, so when the interrupt ends, the CPU will return to the previously interrupted position and continue executing the task it was performing.

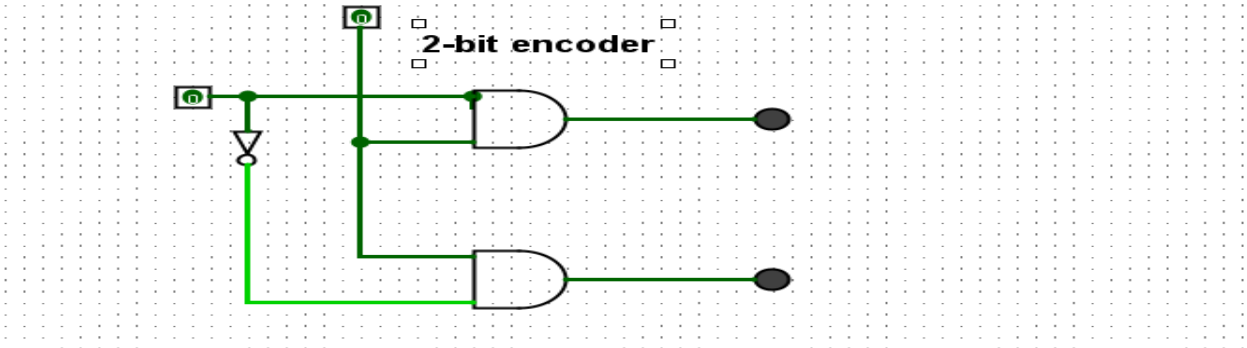
6.2. How are stacks useful in programming ?

In programming, stacks are commonly used to manage program memory or handle the process of function creation and invocation. They are also utilized for program execution. Stacks are used to check the validity of parentheses sequences, such as parentheses, braces, and brackets. For example, they can determine if the sequence of parentheses, including single parentheses, curly braces, or double parentheses, is valid or not. It also help us easy to implement.

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

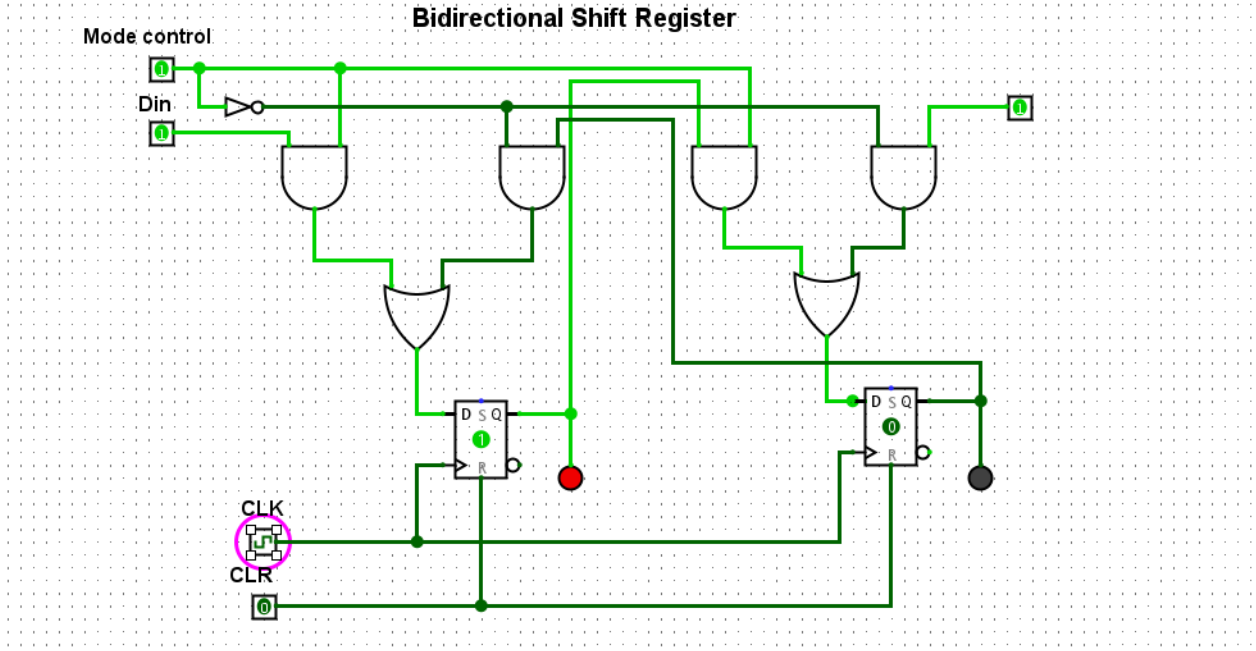


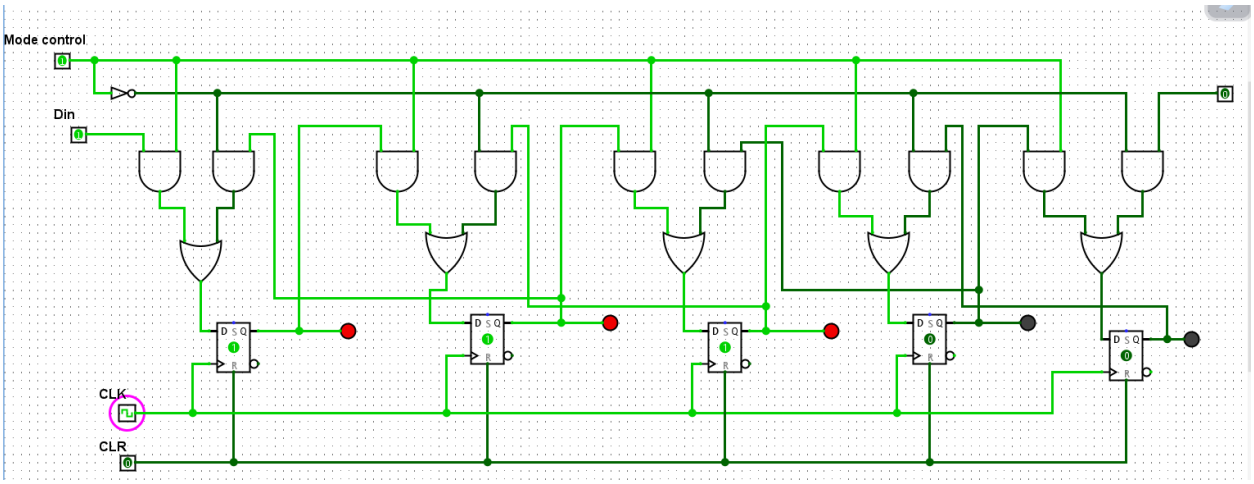
10. For your shift register to work as a stack, it needs to be bi-directional. This means the input to any Flip Flop could come from two places - the left or the right. In lectures we discussed a simple "encoder" circuit that selects which of two data inputs is allowed through, based on a third selection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



11. Now incorporate your encoder above to allow bi-directional shifting of your stack. Your stack 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.**


Bidirectional Shift Register
Mode control
Din
D S Q
R
CLK
CLR


Mode control
Din
D S Q
R
CLK
CLR

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).