COS10004 – Computer System
Name: Phan Vũ – Student Id: 104222099
LAB 4

1.

1.1.

"ROM" means "read-only memory." It is a type of storage medium that allows computers and other electronic devices to keep data forever.

1.2.

“RAM” stands for random access memory. “RAM” stores data on a computer temporarily, while “ROM” stores data permanently.

1.3.

The most common type of memory used today is dynamic RAM (DRAM). In a dynamic RAM chip, each memory cell consists of a transistor and a capacitor and stores one bit of information. On the other hand, static RAM (SRAM) uses a flip-flop to hold each memory bit. A memory cell's flip-flop in SRAM requires four or six transistors and some wiring, but it does not need to be refreshed periodically.

1.4.

Flash memory is usually used to store data on USB thumb drives. Flash memory is a type of memory that doesn't lose its data when the power goes out. It is popular because it is easy to carry, lasts a long time, and can read and write quickly. But you shouldn't store important data on flash memory because it has a short lifespan, can be damaged by physical damage, and could pose security risks. To make sure that important data is safe and secure, it is best to use more backup methods and look into more reliable storage options.

2.

The number of bits required to address all bytes in a 1GB (1024MB) RAM system is 30 bits.

3.

The Von Neumann architecture and the Harvard architecture are both basic computer architectures, but they were made with different ideas in mind.

The Von Neumann architecture is based on the idea of a "stored program," in which both instructions and data are kept in the main memory or RAM. It runs instructions one after the other, and both instructions and data travel along the same path. Programming is made easier and more flexible with this architecture.
On the other hand, the Harvard architecture keeps instructions and data separate from how they are stored and processed. It keeps instructions and data in physically separate memories and uses different paths to get to instructions and data. This architecture aims for high performance and efficiency, especially in programs that need access to both instructions and data at the same time.
In short, the Von Neumann architecture puts instructions and data in the same memory to make things easier, while the Harvard architecture puts instructions and data in different memories to make things faster and more efficient.

4.
Cache memory is a small amount of fast memory that is near to the CPU. Its main job is to store data and instructions that are used often. This lets the CPU get them quickly without having to use the slower main memory. Cache memory improves system performance by making use of the concepts of temporal and spatial locality to cut down on the time it takes to access memory. When the CPU needs data, it first looks in the cache to see if the data is already there. If it is (a "cache hit"), the data is quickly retrieved. If the data is not in the cache (cache miss), it is fetched from main memory and stored in the cache for future use.

5.
An interrupt is a signal or event that is sent by a device or piece of software and stops the normal flow of a program for a short time. It lets the CPU stop doing what it is doing and focus on a task or event that is more important. When an interrupt happens, the CPU saves where it is in the process of running, runs the interrupt handler routine, and then goes back to normal. Four common types of interrupts are:

- Software interrupt
- Hardware interrupt
- Internal interrupt
- External interupt

5.1.
Polling serves as an alternative to interrupts, involving continuous checking of devices or flags by the CPU for events. However, polling is

inefficient due to high CPU overhead, increased latency, inefficient resource usage, and a lack of real-time responsiveness. Interrupts, in contrast, enable the CPU to focus on other tasks until an interrupt is triggered, leading to more efficient and responsive handling of events.

6.

A stack is a data structure that operates on the Last-In-First-Out (LIFO) principle. Elements can only be added or removed from the top of the stack. The primary purpose of a stack is to provide an organized way of managing data, where the last element added is the first one to be removed.

When an element is added to the stack, it is pushed onto the top. When an element is removed from the stack, the topmost element is popped off. This ensures that the most recently added elements are the first to be accessed.

Stacks are commonly used in programming and computer systems for various purposes. They are used to manage function calls and local variables, evaluate expressions, allocate memory, and facilitate backtracking or undo operations. The LIFO behaviour of stacks makes them efficient for certain data management tasks.

6.1.

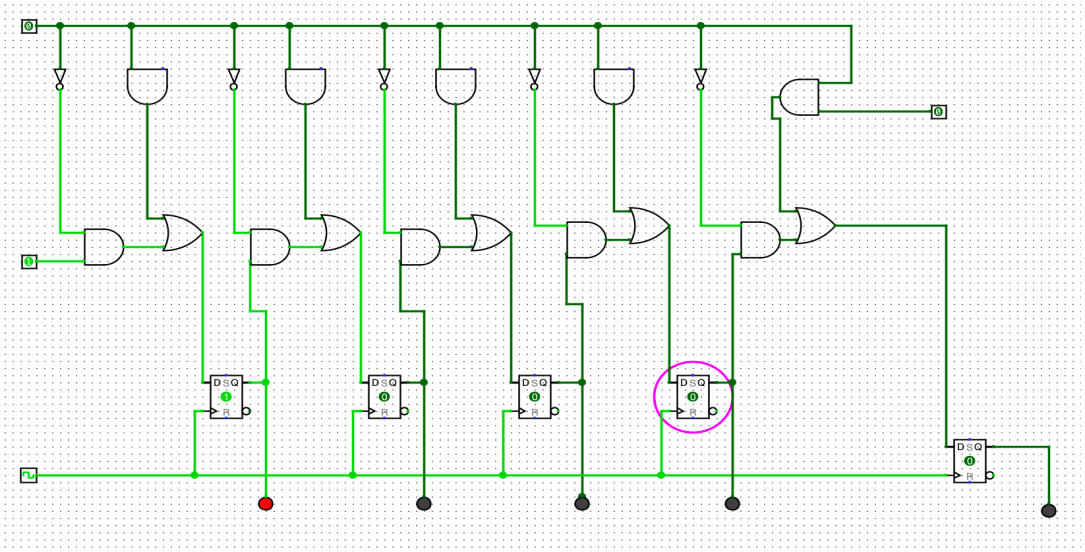
Stacks are useful for handling interrupts in computer systems. They allow for efficient saving and restoring of the CPU's state during interrupt handling. Stacks enable the storage of the interrupted task's context, support nested interrupts, facilitate context switching in multitasking systems, and ensure seamless resumption of interrupted tasks. Stacks are crucial in managing interrupt handling and maintaining the integrity of the system's execution flow.

6.2.

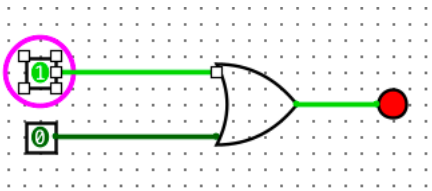
Stacks are useful in programming for their Last-In-First-Out (LIFO) behaviour:

- Peek: Allows inspection of the topmost element on the stack without removing it.
- Swap: Exchanges the positions of the top two elements on the stack.
- Duplicate: Pops the topmost element from the stack and pushes it back onto the stack twice, creating a duplicate.
- Rotate: Specifies the number of elements in the stack to be rotated. The topmost element moves to a specified position, and the remaining elements shift accordingly.

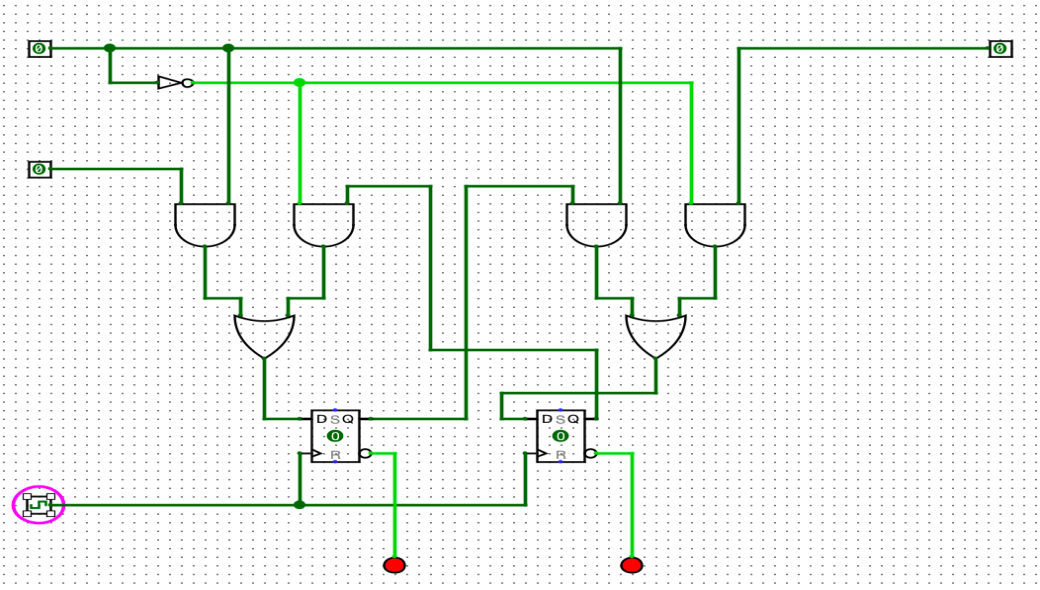
7.



8.



9.



10.