Faculty of Science, Engineering and Technology

stacks.



Computer Systems

Week 5

# Overview

In this laboratory session we start look at memory, encoders and

**Purpose:** To consolidate your knowledge of Memory and Stacks

**Task:**

**Time:** This lab is due by the start of your week 6 lab.

**Assessment:** This lab is worth 1% (up to a maximum of 5%) of your assessment for this unit, and only if demonstrated to your lab demonstrator in the week it is due.

**Resources:** ■ This week's lecture videos

***Submission Details***

You must submit the following files to Canvas:

* A document containing all required work as described below.



# Instructions

## Theory (Memory, Architectures, Interrupts and Stacks)

1. Review the lecture slides on types of memory and provide a short answers to the following questions (using your own words):
   1. What is ROM and what is its primary purpose ?

-> ROM (Read-Only Memory) is a type of non-volatile memory that stores data or instructions that are permanently written during manufacturing. Its primary purpose is to hold firmware or software instructions that are essential for the computer's operation.

* 1. What is RAM and how is it different from ROM ?

-> RAM (Random-Access Memory) is volatile memory used for temporary data storage during a computer's operation. It allows for fast read and write operations and is used to store data that can be quickly accessed and modified.

Difference: RAM's contents are not permanent and are erased when the computer is powered off.

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

-> Static RAM (SRAM) and Dynamic RAM (DRAM) differ in how they store data.

* SRAM uses flip-flops to store each bit, making it faster but more expensive and power-hungry.
* DRAM stores data if it’s refreshed frequently enough, making it less expensive and more power-efficient but slower.
  1. What type of memory is typically used in USB thumb drives ? Why shouldn’t we rely on this for critical data storage ?

-> USB thumb drives typically use EEPROM flash memory for storage.

Write bit by injecting electrons through a barrier layer (physically damaging it)

1. 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 ?

-> 32 bits needed

1. 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 and Harvard architecture are two different computer system designs.

+ In Von Neumann architecture, both data and instructions share a single memory space.

+ In Harvard architecture, data and instructions are stored in separate memory spaces.

- Von Neumann:

+ Data and instructions stored in the same location.

+ Stack central to handling multiple tasks/interrupts.

- Harvard:

+ Separates data and instructions.

+ Increased efficiency and security

+ Reduced generality and versatility.

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

-> Cache memory is a small, high-speed memory unit located between the CPU and main memory (RAM). Its primary role is to temporarily store frequently used data and instructions to reduce the CPU's access time to the main memory, thereby improving overall system performance.

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

-> An interrupt is a signal sent to the CPU to request its attention.  
- Hardware interrupts (I/O devices signaling CPU)

- Software interrupts (system calls)

- External interrupts (external hardware events)

- Internal interrupts (divide-by-zero error)

* 1. Polling is an alternative to interrupts ? Briefly explain polling and why it is not com- monly used.

-> Polling is an alternative to interrupts where the CPU periodically checks if a device or condition needs attention. Polling is not commonly used because it is less efficient, consumes CPU resources, and can introduce latency in responding to events.

1. Explain the general concept of a stack - how do they work, and what is their primary pur- pose.

-> A stack is a data structure that follows the Last-In-First-Out (LIFO) principle. It works by pushing data onto the top of the stack and popping data from the top. The primary purpose of a stack is to manage function calls and store local variables and return addresses in a program's execution.

* 1. How are stacks useful for handling interrupts ?

-> Stacks are useful for handling interrupts because they allow the CPU to save its current execution context (registers, program counter, etc.) when an interrupt occurs, so it can later resume the interrupted task.

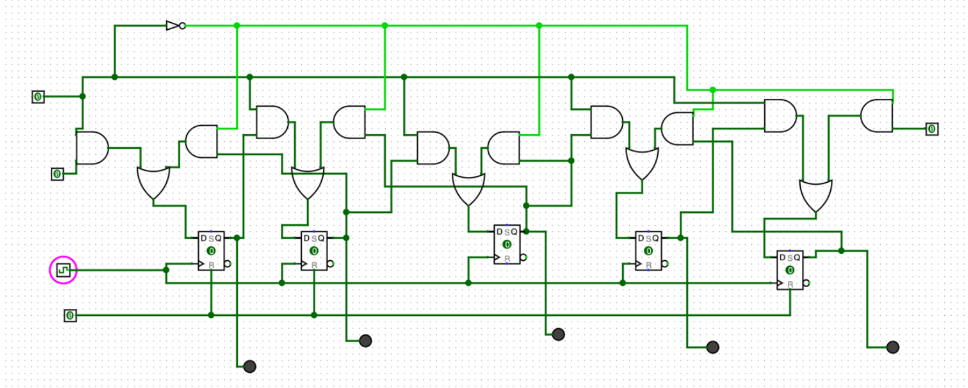
* 1. How are stacks useful in programming ?

-> Stacks are useful in programming for various tasks, including managing function calls and recursion, storing temporary data, and implementing data structures like the stack data structure itself. They help maintain program control flow and manage memory efficiently.

## Provide all the answers to the above questions in your submission document. Practical - Stacks of Stacks !

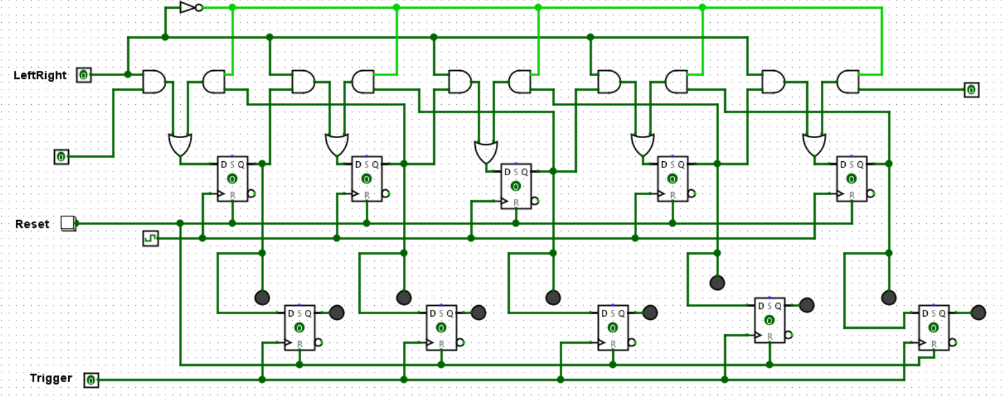
1. Start Logisim and open a new canvas
2. Review the lecture slides on building a stack at the top of this lab sheet. We are going to build a 5-bit deep, 1-bit wide stack.
3. Start by building a simple shift register that moves bits from one flip flop to the next each clock pulse. For this you will need a “Data In” pin which sets the next bit to be pushed to the stack, and a clock to invoke the shifting.
4. For your shift register to work as a stack, it needs to be bi-directional. This means the in- put to any Flip Flop could come from two places - the left or the right. In lectures we dis- cussed 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.
5. Now incorporate your encoder above to allow bi-directional shifting of your stack. Your stack should:
   1. push and pop bits onto and off the stack, using clock pulses and a direction toggle switch
   2. show the state of each Flip Flop using LEDs.

## Export your circuit as an image and include it in your submission document. Demon- strate your working stack to your lab demonstrator.



1. Modify your stack so that it has the option to read out its contents ***in parallel*** to a sepa- rate 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 re- tain the last state read in (and should have LEDs connected to each Flip Flop out showing its state).

## Export your circuit as an image and include it in your submission document. Demon- strate your working stack to your lab demonstrator.



**When complete:**

* Submit your answers (screen shots, etc) in a single document using **Canvas**
* Show your lab demonstrator your working circuits in class (you must do this to get the 1%). Your lab demonstrator may request you to resubmit if issues exist.