Activity 1 - Von Neumann VS Harvard architecture

1. List at least four main differences between Von Neumann and Harvard architecture.
Von Neumann architecture uses a single shared memory for instruction/data, using sequential access as part of its simple system complexity. This simplification permits instruction read/writes in the same way memory data read/writes are performed, on a single bus. Harvard takes an alternative approach, distinguishing instruction/data buses and memory, so the CPU can read an instruction while performing memory read/write operations. **[1]** This approach allows parallel access of instructions and data, compared to Von Neumann's sequential access.

2. Discuss advantages and disadvantages of Harvard architecture
Despite Harvard architecture's parallel access, this distinction limits flexibility and increases complexity/expense by facilitating dual-memory processing. Von Neumann architecture is cheaper and simpler to implement, mitigating cost and complexity. Von Neumann's shared memory permits write operations on instructions as if they were memory data, so things like self-modifying code are more convenient to implement. This can also be seen as a drawback considering the potential for memory corruption risks, and the shared memory bus introduces the Von Neumann bottleneck, since the CPU has to sequentially wait and fetch either instruction or data any given time. **[2]**
Harvard architecture optimises memory usage and improves the system security by distinguishing instruction/data memory, mitigating unintentional memory artefacts and increasing execution speed through parallel access. These enhancements also come with increased design complexity, higher cost, and less flexibility to modify instructions at runtime.

Activity 2 - Arduino board architecture

1. Based on the diagram, what architecture is used in the Arduino board? Justify your answer.
The control line directions and allocation of instruction/data spaces indicate that this model follows Harvard architecture. The shared 8-bit data bus is bidirectional, with bidirectional control between the bus and I/O, Data, Interrupt, and Registers components. Across the top axis (8-bit bus), the program counter -> flash -> inst. register -> inst. decoder follows a unidirectional control flow for instructions, in a distinguished instruction space (away from data memory). The bidirectional control line along the bus and distinguished instruction/data spaces, despite indirect addressing between SRAM <--> PC && Flash, clarifies that this system takes Harvard's approach.

2. Identify different types of memory in the Arduino architecture, discuss their usage, and sort them based on speed.

The architecture uses a combination of SRAM, Flash Memory, EEPROM and Registers: **[3]**
Flash - Stores the program, read-only at runtime and writable via programming (can be done at runtime with methods like a bootloader, etc.). Flash retains all memory content after power-off, only facilitates sequential fetch via program counter. Runtime code cannot directly modify flash memory.
SRAM - Static RAM for performing stack operations, heap memory allocation, global/static/local vars and no power-off persistence.
EEPROM - Stores things like user settings and calibration data, important data that needs to persist for the system to behave as intended. This type of memory is retained indefinitely without power for full persistence.
Registers - Fast volatile memory located in the CPU core, general-purpose memory. Handles operand storage for ALU, temp vars, and instruction-level data manipulation. Registers have single clock cycle access speed.

Rank by speed:

1) Registers (32x8-bit Volatile, read-write), instant, CPU-local
2) SRAM (2KB Volatile, read-write) fast, direct access by data bus
3) Flash (32KB Non-volatile, read) medium, read-only fetch
4) EEPROM (1KB Non-volatile, read-write) highest latency, lowest endurance

References

[1] "Harvard Architecture," GeeksforGeeks. [Online]. Available: https://www.geeksforgeeks.org/computer-organization-architecture/harvard-architecture/. [Accessed: 11-Jul-2025].

[2] "Von Neumann Architecture," GeeksforGeeks. [Online]. Available: https://www.geeksforgeeks.org/computer-organization-architecture/computer-organization-von-neumann-architecture/. [Accessed: 11-Jul-2025].

[3] Microchip Technology Inc., "ATmega328P Datasheet," Atmel-7810-Automotive-Microcontrollers, 2016. [Online]. Available: https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf. [Accessed: 11-Jul-2025].