Introduction

David Robinson

Von Neumann Architecture

Designed to run a sequence of instructions and required weeks of rewiring to reconfigure the sequence.

1. Fetch - Load an instruction from memory into a CPU register
2. Decode - Parse the instruction, determining how this instruction needs to be executed
3. Execute - Run the instruction
4. Repeat

VM/0

1. Load the address of the next instruction into the memory address register (MAR)
2. Increment the program counter
3. Load the instruction into the memory data register (MDR)
4. Load the instruction into the instruction register
5. Decode the instruction
6. Execute the instruction
7. Repeat

Incorporating Error Handling

- An overflow flip-flop bit gets set when an overflow condition occurs.
- The error handling instruction is stored in the NEWPC register.
- Before the next instruction in the sequence, the CPU checks the overflow bit and if it is set, it executes the instruction stored in NEWPC.

Incorporating Memory Protection

- Memory bounds for a program are stored in a FENCE register.
- An out-of-bounds flip-flop bit gets set when a requested address exceeds the FENCE value.

Instructions are divided between privileged and unprivileged (opcodes above a certain value are considered privileged). The system can also be running in privileged or unprivileged mode. If the system is running in unprivileged mode but the instruction is privileged, the system will throw an error.

Interrupt Handler Vector

1. Address Overflow Handler
2. Address Memory Protection (MP) Handler
3. Address Program Interrupt (PI) Handler
4. Address Timer Handler
5. Address I/O Handler
6. Address Supervisor Call (SVC) Handler

Operating System Principles

1. **Virtualization**

   Processes only have access to virtual representations of resources, but they can operate under the assumption that they have the resource to themselves.

2. **Concurrency**

   Processes and threads can be executed simultaneously. The concurrency must not be exposed to independent processes, but the operating system needs to provide synchronization abstractions and prevent deadlocking.

3. **Persistence**

   Data on disk are abstracted to the files and directories that make up filesystems. Also, some data must be allowed to persist through reboots or system crashes.

Operating Systems

1. Atlas - Introduced the concept of system calls
2. UNIX - Promoted the idea of building small, powerful programs that could be connected together
3. DOS - Represented a leap backwards for OS development because it was designed for single-task and single-user, but there were earlier operating systems like UNIX and Multics that supported multitasking and multi-user.