Microprogrammed Control

Chapter 17
From chapter 16…

- Implementation of the control unit:
  - Hardwired
    - Essentially a combinatorial circuit
  - Microprogrammed
    - An alternative to a hardwired implementation.
    - It consists of a sequence of instructions in a microprogramming language.
• The term *microprogram* was coined by Wilkes in early 1950s

• The proposal was a control unit design that was organized and systematic and avoided the complexities of a hardwired implementation

• It would require a fast, relatively inexpensive control memory => 1964 - IBM
Microinstructions (1)

- To implement a CU as an interconnection of basic logic elements is no easy task.
- The design must include logic for sequencing through micro-operations for: executing micro-operations, interpreting codes, and making decisions based on flags.
- Difficult: to design and test such a piece of hardware, and to change the design if one wishes to add a new machine instruction.
- An alternative: to implement a microprogrammed control unit.
Microinstructions (2)

- This is a language: *microprogramming language*.
- Each line describes a set of micro-operations occurring at one time: *microinstruction*.
- A sequence of instructions: *microprogram* or *firmware* (a midway between hardware and software).

<table>
<thead>
<tr>
<th>Micro-operations</th>
<th>Timing</th>
<th>Active Control Signals</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fetch:</strong></td>
<td>t1: MAR ← (PC)</td>
<td>C₂</td>
</tr>
<tr>
<td></td>
<td>t2: MBR ← Memory</td>
<td>C₅, C_R</td>
</tr>
<tr>
<td></td>
<td>PC ← (PC) + 1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>t3: IR ← (MBR)</td>
<td>C₄</td>
</tr>
<tr>
<td><strong>Indirect:</strong></td>
<td>t1: MAR ← (IR(Address))</td>
<td>C₈</td>
</tr>
<tr>
<td></td>
<td>t2: MBR ← Memory</td>
<td>C₅, C_R</td>
</tr>
<tr>
<td></td>
<td>t3: IR(Address) ← (MBR(Address))</td>
<td>C₄</td>
</tr>
<tr>
<td><strong>Interrupt:</strong></td>
<td>t1: MBR ← (PC)</td>
<td>C₁</td>
</tr>
<tr>
<td></td>
<td>t2: MAR ← Save-address</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PC ← Routine-address</td>
<td></td>
</tr>
<tr>
<td></td>
<td>t3: Memory ← (MBR)</td>
<td>C₁₂, C_W</td>
</tr>
</tbody>
</table>

Cᵣ = Read control signal to system bus.
Cₜ = Write control signal to system bus.
Microinstructions (3)

- For each micro-operation: generation of control signals.
- *Control word*: each bit represents one control line.
- *Horizontal microinstruction*: bits for the control signal, microinstruction address, and jump condition.

<table>
<thead>
<tr>
<th>Internal CPU control signals</th>
<th>System bus control signal</th>
<th>Microinstruction address</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jump condition:</td>
<td></td>
<td>Unconditional, Zero, Overflow, Indirect bit</td>
</tr>
</tbody>
</table>
Organization of Control Memory

- The microinstructions can be arranged in a control memory.
- This figure is a concise description of the complete operation of the control unit (it can be used like a way to implement the control unit).
Functioning of Microprogrammed Control Unit

Introduction to Computer Architecture and Organization
Lesson 21 – Slide 9/24
Control Unit functions (1) 
(during one clock pulse)

1. To execute an instruction, the sequencing logic unit issues a READ command to the control memory.

2. The word whose address is specified in the control address register is read into the control buffer register.

3. The content of the control buffer register generates control signals and next address information for the sequencing logic unit.
Control Unit functions (2)  
(during one clock pulse)

4. The sequencing logic unit loads a new address into the control address register based on the next-address information from the control buffer register and the ALU flags.
   – Get the next instruction
   – Jump to a new routine based on a jump microinstruction
   – Jump to a machine instruction routine
Advantages and Disadvantages

• **Advantages:**
  – It simplifies the design of the CU
  – Cheaper and less error-prone to implement
  – Decoders and sequencing logic are very simple pieces of logic
  – *Hardwired*: CU contains complex logic for sequencing micro-operations

• **Disadvantages:**
  – Slower than a hardwired unit of comparable technology.

• **CISC**: microprogrammed control
• **RISC**: hardwired control
Microinstruction Sequencing

- Two basic tasks performed by a microprogrammed control unit are as follows:
  - Microinstruction sequencing
  - Microinstruction execution
- In designing a control unit, these tasks must be considered together, because both affect the format of the microinstruction and the timing of the control unit.
Design considerations (1)

- **The size of the microinstruction**: about the minimizing the size of the control memory reduces the cost of that component.
- **The address-generation time**: about to execute microinstructions as fast as possible.
Design considerations (2)

- In executing a microprogram, the address of the next microinstruction to be executed in one of these categories:
  - Determined by instruction register
  - Next sequential address
  - Branch
Sequencing Techniques

- Based on the current microinstruction, condition flags, and the contents of the instruction register, a control memory address must be generated for the next microinstruction.

- Techniques:
  - Two address fields
  - Single address field
    - Address field
    - Instruction register code
    - Next sequential address
  - Variable format
### Address generation techniques

<table>
<thead>
<tr>
<th>Explicit</th>
<th>Implicit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Two-field</td>
<td>Mapping</td>
</tr>
<tr>
<td>Unconditional branch</td>
<td>Addition</td>
</tr>
<tr>
<td>Conditional branch</td>
<td>Residual control</td>
</tr>
</tbody>
</table>
Microinstruction Execution
A Taxonomy of Microinstruction

- Microinstructions can be classified in a variety of ways.
- Distinctions that are commonly made in the literature include:
  - Vertical / horizontal
  - Packed / Unpacked
  - Hard / soft microprogramming
  - Direct / indirect encoding
Vertical / horizontal

- Microinstruction address
- Jump condition
- Function codes
- Internal CPU control signals
- System bus control signal
- Jump condition: Unconditional, Zero, Overflow, Indirect bit
- Microinstruction address
### The Microinstruction Spectrum

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Terminology</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unencoded</td>
<td>Highly encoded</td>
</tr>
<tr>
<td>Many bits</td>
<td>Few bits</td>
</tr>
<tr>
<td>Detailed view of hardware</td>
<td>Aggregated view of hardware</td>
</tr>
<tr>
<td>Difficult to program</td>
<td>Easy to program</td>
</tr>
<tr>
<td>Concurrency fully exploited</td>
<td>Concurrency not fully exploited</td>
</tr>
<tr>
<td>Little or no control logic</td>
<td>Complex control logic</td>
</tr>
<tr>
<td>Fast execution</td>
<td>Slow execution</td>
</tr>
<tr>
<td>Optimize performance</td>
<td>Optimize programming</td>
</tr>
</tbody>
</table>

---

**Introduction to Computer Architecture and Organization**

*Lesson 21 – Slide 22/24*
Microinstruction encoding

(a) Direct encoding

(b) Indirect encoding
Applications of Microprogramming

- Realizations of computers
- Emulation
- Operating system support
- Realization of special-purpose devices
- High-level language support
- Microdiagnostics