Topic for today:

A first look at the control unit

Processor Control Unit

- Role of processor control unit
  - Keeps operations synchronized
  - Make sure that bits flow to the correct components at the correct time
- How can we build this control unit?
  - Hardwired control
  - Microprogrammed control
- The result is the same – control signals!

Decoding opcodes

Recall that a single operation in our ISA typically involves a sequence of several register transfers. A single register transfer can be executed by setting various control lines either high or low.

Early in the history of computers, each instruction involved just a single register transfer, and the opcode indicated the high/low settings.

Decoding opcodes

Questions:

How do you get the settings needed for a register transfer from a single opcode?

How can you get a sequence of transfers from a single opcode?

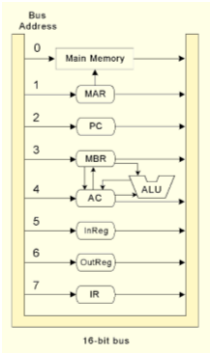
Processor Control Unit

- Remember the register transfer language description for each MARIE instruction?
  - Table 4.7
  - This is what the control unit manages.
- Each microoperation consists of a distinctive signal pattern that is interpreted by the control unit and results in the execution of an instruction
  - RTL for the ADD instruction

```
MAR ← X
MBR ← M[MAR]
AC ← AC + MBR
```

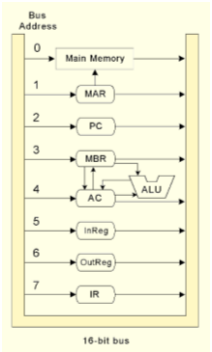
- Each of MARIE's registers and main memory have a unique address along the datapath.
- The addresses take the form of signals issued by the control unit.

How many signal lines does MARIE's control unit need?

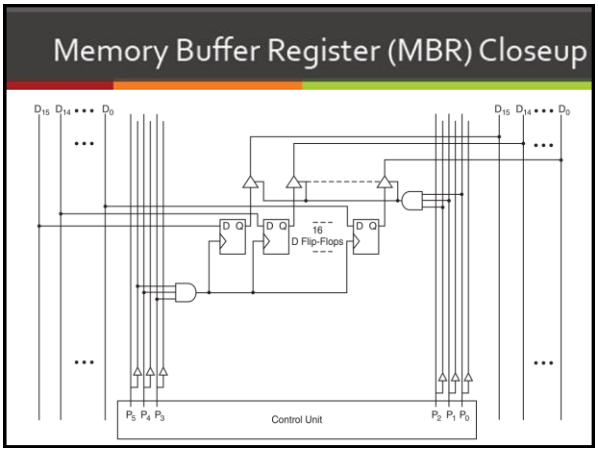


6

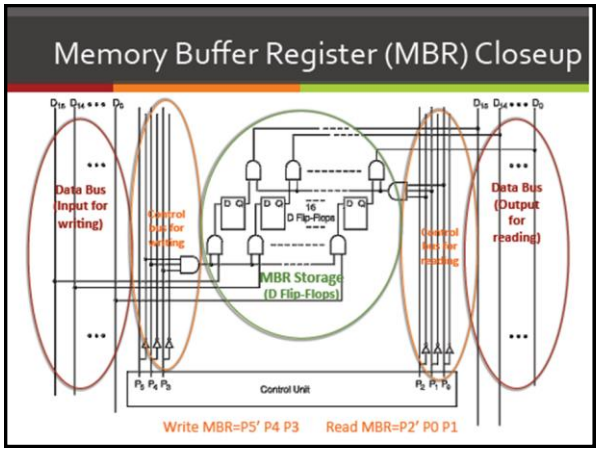
- Two sets of three signals each.
- {P_2, P_1, P_0}
  - controls **reading** from memory or a register
- {P_5, P_4, P_3}
  - controls **writing** to memory or a register.



7



Memory Buffer Register (MBR) Closeup



Processor Control Unit

- Control unit must manage more than just registers/ main memory
  - What about the ALU modes?
- ALU has only four operations
  - Add, subtract, clear, and "do nothing"
- **ALU controls: $A_0 - A_1$**

ALU Control Signals		ALU Response
A_1	A_0	
0	0	Do Nothing
0	1	AC ← AC + MBR
1	0	AC ← AC – MBR
1	1	AC ← 0 (Clear)

Processor Control Unit

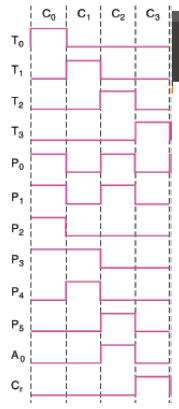
- How does the control unit perform operations in **sequence?**
- Longest instruction is JNS (look at RTL in Table 4.7)
  - 7 steps
  - Need a 3-bit counter wired to a 3-8 decoder
  - Counter reset for shorter instructions
- **Output of decoder is "timing" signals: $T_0 - T_7$**
- **The entire set of MARIE's control signals consists of:**
  - Register controls
    - P_0 through P_5
  - ALU controls
    - A_0 through A_1
  - Timing
    - T_0 through T_7
  - Counter reset C_T

ADD Instruction Control

- ADD instruction RTL
  - `MAR ← X`
  - `MBR ← M[MAR]`
  - `AC ← AC + MBR`
- After the add instruction is fetched, the address (X) is in the rightmost 12 bits of the IR
  - IR datapath address is 7
  - Raise signals P2, P1, and P0 to **read from IR**
- X is copied to the MAR
  - MAR datapath address is 1
  - Raise signal P3 to **write to MAR**

ADD Instruction Control

- Complete signal sequence for ADD instruction
  - P3 P2 P1 P0 T0: MAR ← X
  - P4 P3 T1: MBR ← M[MAR]
  - A0 P5 P1 P0 T2: AC ← AC + MBR
  - Cr T3: [Reset counter]
- These signals are ANDed with combinational logic to bring about the desired machine behavior



ADD Instruction Control

- Add instruction timing diagram
- Notice the concurrent signal states during each machine cycle: C0 through C3.

```
P3 P2 P1 P0 T0: MAR ← X
      P4 P3 T1: MBR ← M[MAR]
A0 P5 P1 P0 T2: AC ← AC + MBR
         Cr T3: [Reset counter]
```

14

Processor Control Unit

- This signal pattern needs to be produced regardless of whether the processor uses *hardwired* or *microprogrammed* control
- **Hardwired control unit**
  - Control unit is pure digital logic
- **Microprogrammed control unit**
  - A tiny program (called "microcode") saved in ROM
    - Even more rudimentary than assembly language!
  - Microinstructions are fetched, decoded, and executed in the same manner as regular instructions
  - Control unit works like a processor-in-miniature

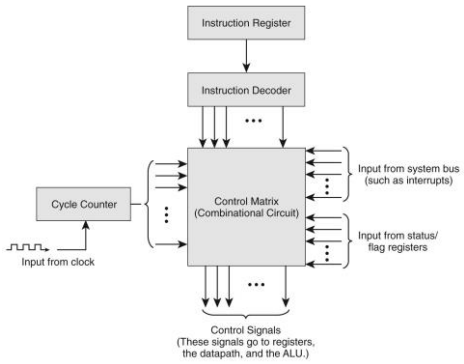
15

Hardwired control

In a hardwired approach, the opcode is decoded by a large set of digital circuits (decoders, multiplexers, etc.) to send the correct signals to the registers, ALU, and internal CPU bus controller. (This is all just Boolean algebra!)

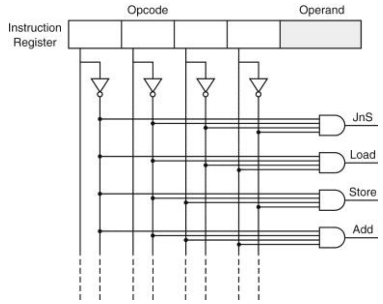
Also, the clock signal is connected to a counting circuit to ensure that actions happen in the correct sequence.

Hardwired Control Unit

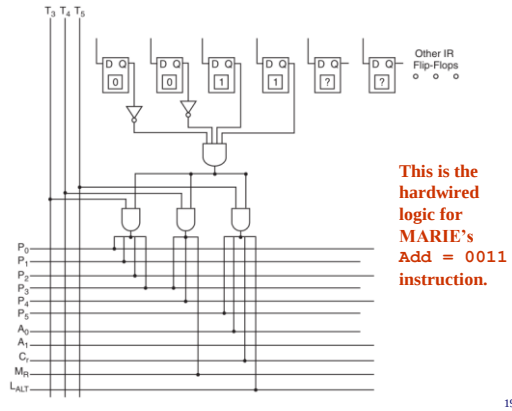


17

MARIE's instruction decoder (partial)



18



This is the hardwired logic for MARIE's `Add = 0011` instruction.

19

Note

In a hardwired approach, any change in the ISA requires a major redesign of the CPU circuitry.

Consider the following Discuss

Suppose you are designing a hardwired control unit for a very small computerized device. This system is so revolutionary that the system designers have devised an entirely new ISA for it. Because everything is so new, you are contemplating including one or two extra flip-flops and signal outputs in the cycle counter. Why would you want to do this? Why would you not want to do this? Discuss the tradeoffs.