Computer Architecture
(CSC-3501)
Lecture 14
(04 Mar 2008)

Seung-Jong Park (Jay)
http://www.csc.lsu.edu/~sjpark

Announcement

4.13 A Discussion on Decoding

- A computer's control unit keeps things synchronized, making sure that bits flow to the correct components as the components are needed.
- There are two general ways in which a control unit can be implemented: *hardwired control* and *microprogrammed control*.
  - With microprogrammed control, a small program is placed into read-only memory in the microcontroller.
  - Hardwired controllers implement this program using digital logic components.

4.13 A Discussion on Decoding

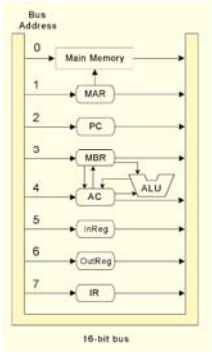
- Your text provides a complete list of the register transfer language for each of MARIE's instructions.
- The microoperations given by each RTL define the operation of MARIE's control unit.
- Each microoperation consists of a distinctive signal pattern that is interpreted by the control unit and results in the execution of an instruction.
  - Recall, the RTL for the Add instruction is:

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


4.13 A Discussion on Decoding

- 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.

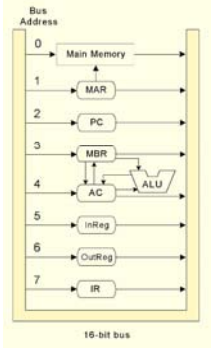
How many signal lines does MARIE's control unit need?



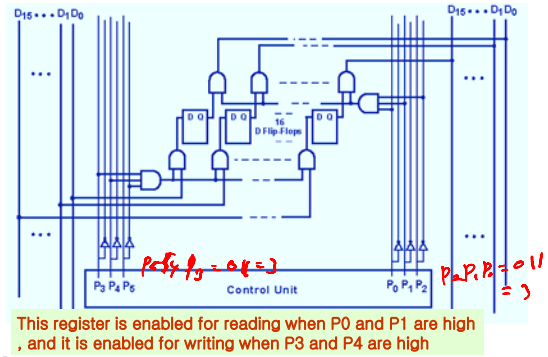
4.13 A Discussion on Decoding

- Let us define two sets of three signals.
- One set, P_0, P_1, P_2, controls reading from memory or a register, and the other set consisting of P_3, P_4, P_5, controls writing to memory or a register.

The next slide shows a close up view of MARIE's MBR.



4.13 A Discussion on Decoding



This register is enabled for reading when P0 and P1 are high , and it is enabled for writing when P3 and P4 are high

4.13 A Discussion on Decoding

- Careful inspection of MARIE's RTL reveals that th e ALU has only three operations: add, subtract, an d clear.
  - We will also define a fourth "do nothing" state.
- The entire set of MARIE's control signals consists of:
  - Register controls: P_0 through P_5.
  - ALU controls: A_0 through A_3
  - Timing: T_0 through T_7 and counter reset C_r

4.13 A Discussion on Decoding

- Consider MARIE's Add instruction. It's RTL is:

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

- After an Add instruction is fetched, the address, X, is i n the rightmost 12 bits of the IR, which has a datapat h address of 7.
- X is copied to the MAR, which has a datapath addres s of 1.
- Thus we need to raise signals P_2, P_1, and P_0 to read f rom the IR, and signal P_3 to write to the MAR.

4.13 A Discussion on Decoding

- Here is the complete signal sequence for MARIE's A dd instruction:

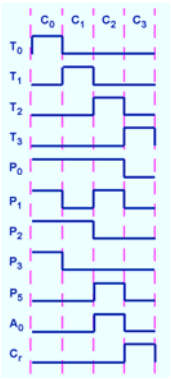
```
                P0 P1 P2  P3 T0: MAR ← X
                P3 P4     T1: MBR ← M[MAR]
A0  P0 P1 P2    P5        T2: AC ← AC + MBR
                Cr        T3: [Reset counter]
```

- These signals are ANDed with combinational logic to bring about the desired machine behavior.
- The next slide shows the timing diagram for this instr uction..

4.13 Decoding

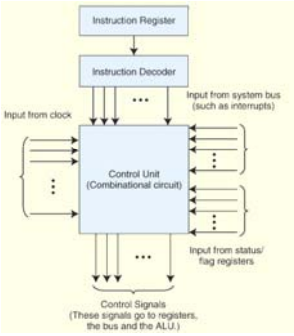
- Notice the concurrent sign al states during each mach ine cycle: C_0 through C_3.

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

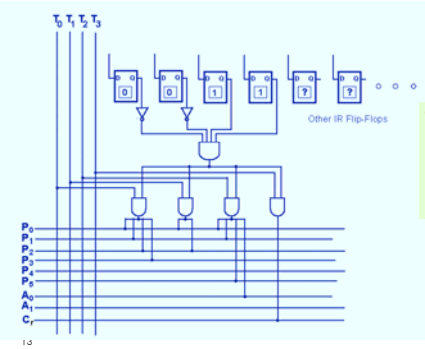


4.13 A Discussion on Decoding

- We note that the signal pat tern just described is the s ame whether our machine used hardwired or micropr ogrammed control.
- In microprogrammed contr ol, the bit pattern of an inst ruction feeds directly into t he combinational logic with in the control unit.



4.13 A Discussion on Decoding

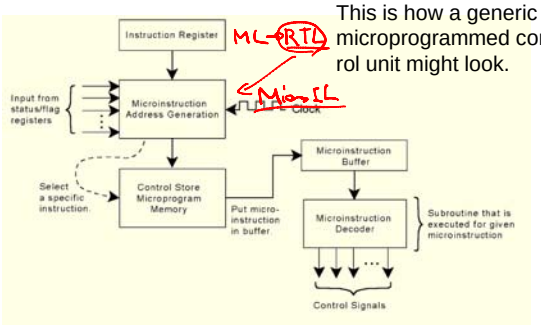


This is the hardwired logic for MARIE's Add = 0011 instruction.

4.13 A Discussion on Decoding

- In microprogrammed control, instruction microcode produces control signal changes.
- Machine instructions are the input for a microprogram that converts the 1s and 0s of an instruction into control signals.
- The microprogram is stored in firmware, which is also called the control store.
- A microcode instruction is retrieved during each clock cycle.

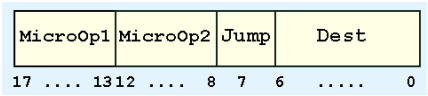
4.13 A Discussion on Decoding



This is how a generic microprogrammed control unit might look.

4.13 A Discussion on Decoding

- If MARIE were microprogrammed, the microinstruction format might look like this:



- **`MicroOp1`** and **`MicroOp2`** contain binary codes for each instruction. **`Jump`** is a single bit indicating that the value in the **`Dest`** field is a valid address and should be placed in the microsequencer.

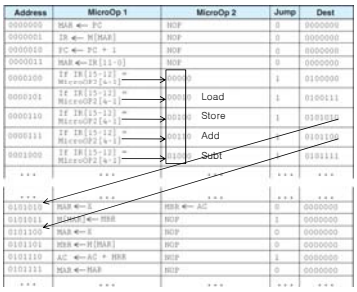
4.13 A Discussion on Decoding

- The table below contains MARIE's microoperation codes along with the corresponding RTL:

MicroOp Code	Microoperation	MicroOp Code	Microoperation
00000	NOP	01100	MBR ← M[MAR]
00001	AC ← 0	01101	outREG ← AC
00010	AC ← AC – MBR	01110	PC ← IR[11-0]
00011	AC ← AC + MBR	01111	PC ← MBR
00100	AC ← InREG	10000	PC ← PC + 1
00101	IR ← M[MAR]	10001	If AC = 00
00110	M[MAR] ← MBR	10010	If AC > 0
00111	MAR ← IR[11-0]	10011	If AC < 0
01000	MAR ← MBR	10100	If IR[11-10] = 00
01001	MAR ← PC	10101	If IR[11-10] = 01
01010	MAR ← X	10110	If IR[11-10] = 10
01011	MBR ← AC	10111	If IR[15-12] = MicroOp2[4-1]

4.13 A Discussion on Decoding

- The first nine lines of MARIE's microprogram are given below (using RTL for clarity):



4.13 A Discussion on Decoding

- The first four lines are the fetch-decode-execute cycle.
- The remaining lines are the beginning of a jump table.

Address	MicroOp 1	MicroOp 2	Jump	Dest
0000000	MAR ← PC	NOP	0	0000000
0000001	IR ← M[MAR]	NOP	0	0000000
0000010	PC ← PC + 1	NOP	0	0000000
0000011	MAR ← IR[11-0]	NOP	0	0000000
0000100	If IR[15-12] = MicroOP2[4-1]	00000	1	0100000
0000101	If IR[15-12] = MicroOP2[4-1]	00010	1	0100111
0000110	If IR[15-12] = MicroOP2[4-1]	00100	1	0101010
0000111	If IR[15-12] = MicroOP2[4-1]	00110	1	0101100
0001000	If IR[15-12] = MicroOP2[4-1]	01000	1	0101111
...	...	...	...	...
...	...	...	...	...
0101010	MAR ← X	MBR ← AC	0	0000000
0101011	M[MAR] ← MBR	NOP	1	0000000
0101100	MAR ← X	NOP	0	0000000
0101101	MBR ← M[MAR]	NOP	0	0000000
0101110	AC ← AC + MBR	NOP	1	0000000
0101111	MAR ← MAR	NOP	0	0000000
...	...	...	...	...

4.13 A Discussion on Decoding

- It's important to remember that a microprogrammed control unit works like a system-in-miniature.
- Microinstructions are fetched, decoded, and executed in the same manner as regular instructions.
- This extra level of instruction interpretation is what makes microprogrammed control slower than hardwired control.
- The advantages of microprogrammed control are that it can support very complcated instructions and only the microprogram needs to be changed if the instruction set changes (or an error is found).