

CSCE 611

Digital Systems Design I

2005/3/27

Week 9b Supplemental Notes

Comparing ASM and Logic Design Methods

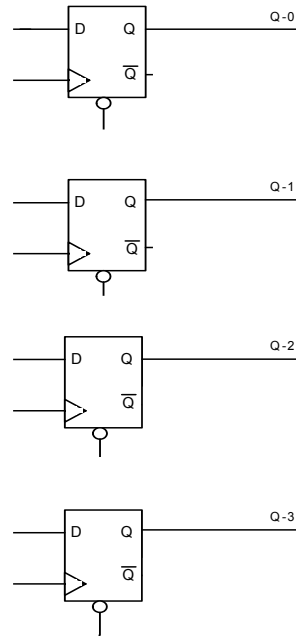
© 2004 Dr. James P. Davis
Figures: Lee © 2000 Prentice-Hall Publishers, Inc.

Week 9 - Outline

- Objective: To understand the various methods for realizing the logic of the controller block.
 - We have discussed how the FSM controls the data path.
 - We have explored how to map from the Register Transfer Notation (RTN) expressions, for both assignments from registers and assignments through macro-functions, to a data path architecture (block diagram).
 - We have explored how to take a data path diagram for some portion of a register-level architecture, and map its operations back to RTN expressions attached to states in the ASM model.
- Control logic techniques:
 - ✓ State register encoding – (1) binary, (2) Gray code, (3) one-hot encoding.
 - ✓ MUX-based control – we encode the selection of state outputs using Multiplexers.
 - ✓ PLD – using a programmable logic devices (e.g., PLA, EPROM).
 - ✓ Counters – encode the state sequencing using a counter.

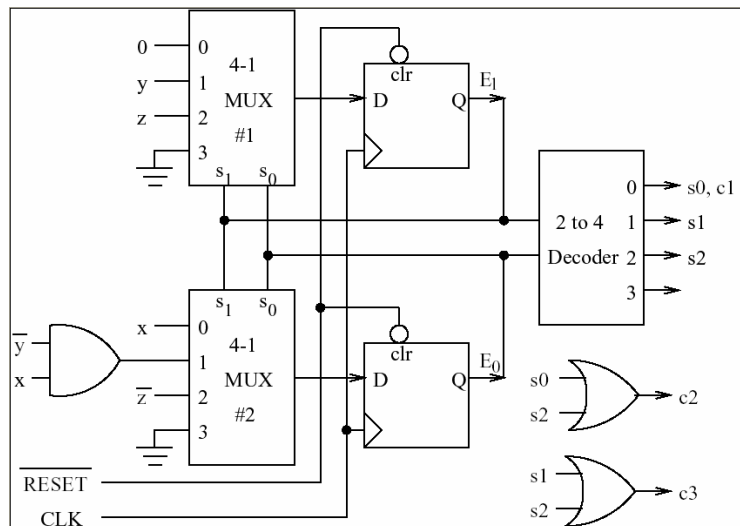
Control Path Technique – FSM Encoding

- Encoding Techniques – we are interested in encoding the output lines from state registers:
 - Binary – ascending count of binary numbers representing each state in the sequence. Control output signals require decoding (combinational logic) to generate a specific control signal.
 - Gray – sequencing of the states so that only one bit between adjacent values changes at a time. This minimizes “glitches”, since ascending Binary encoding involves transitioning through intermediate values when more than a single bit changes value.
 - One-hot – use one D-FF for each state output line, where only a single line is active for each state. No encoding of the lines, so output control signals can be derived directly off the state lines.



Control Path Technique - MUX

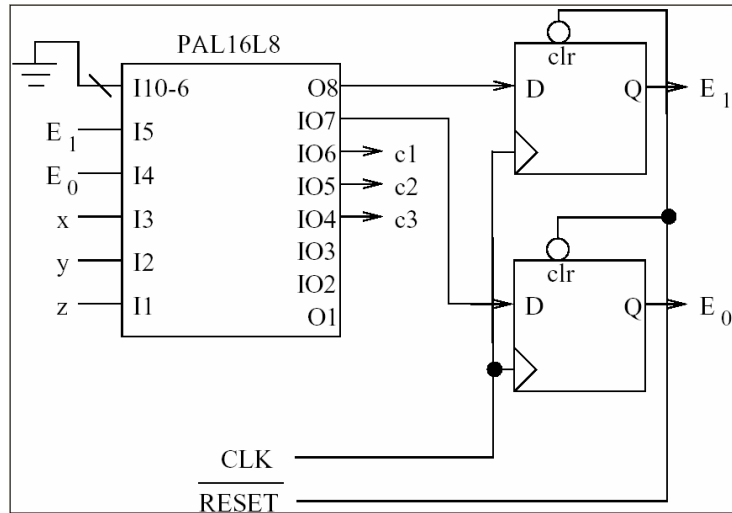
- MUX method:
 - N states are encoded using $\max(\log_2 N)$ D-FFs.
 - MUXes are used as universal combinational logic devices implementing the required decoding logic.
 - This is the basis for the LUT used in FPGAs.
 - $2^n : 1$ MUX is used to implement any combinational logic function of $n + 1$ variables.
 - If more than $n + 1$ variables are required, use other logic gates.



Control Path Technique - PLD

- PLD method:

- Start with logic equations for next state variables and control outputs.
- Use a PLD device to implement all the combinational logic.
- Can even use PLD to implement the register logic of the D-FF themselves.
- Example: PAL: programmable array logic (AND gates in horizontal plane, OR gates in vertical plane).



Source: Lee © 2000 Prentice-Hall Publishers, Inc.



© 2004 Dr. James P. Davis Page 5

Programmable Logic Devices - PLA

- Programmable Logic Array (detail):

- Generates outputs from inputs by applying Sum of Products (AND/OR map).
- Inputs enter through the AND plane; equation terms that are AND'ed are wired together (diode or fuse connection).
- Outputs generated off the OR plane; Sum terms from equations form outputs.
- Sometimes outputs from the PLA can be fed back into the array as inputs, such as the *present state* lines coming out of state registers to feed back into logic to generate *next state* equations.

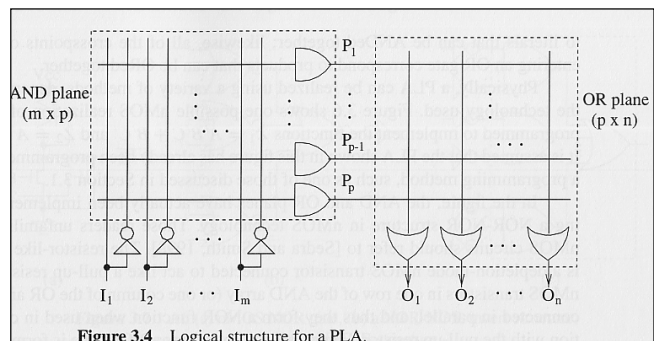
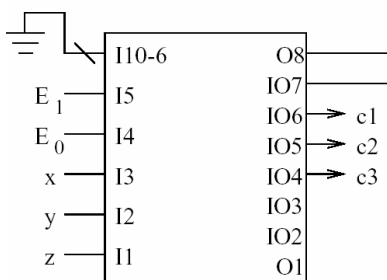


Figure 3.4 Logical structure for a PLA.

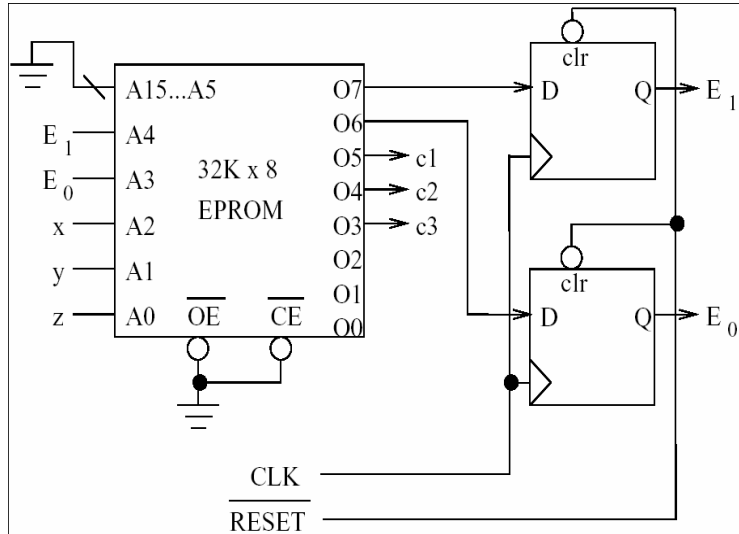
Source: Lee © 2000 Prentice-Hall Publishers, Inc.



© 2004 Dr. James P. Davis Page 6

Control Path Technique - EPROM

- EPROM method:
 - Erasable programmable read-only memory.
 - Use an EPROM device to implement the combinational logic and the registers of the state machine.
 - Can even use PLD to implement the register logic of the D-FF themselves.
 - EPROM used for basic table lookup—this is known as *micro-programmed control* (much like how a CPU works).

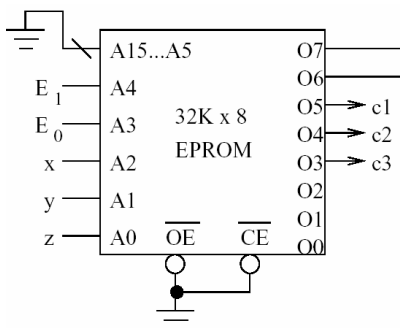


Source: Lee © 2000 Prentice-Hall Publishers, Inc.

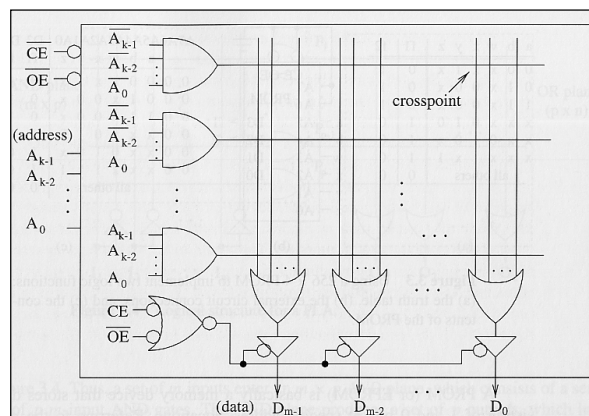


Programmable Read-Only Memory - PROM

- Programmable ROM (detail):
 - ✓ Memory device that stores data at specific locations, addressed through explicit address pins.
 - ✓ As AND/OR array: the address lines are inputs to AND array, and data lines are the outputs out of the OR array.
 - ✓ Selecting a specific address (SoP minterm), the data at that address (the output functions using that minterm) are “enabled” as ROM output.



• $N \times M$ ROM array has $2^P = N \Rightarrow P$ address bits, and M -bit memory word.
 OE: output enable (active low); CE: control enable (low)



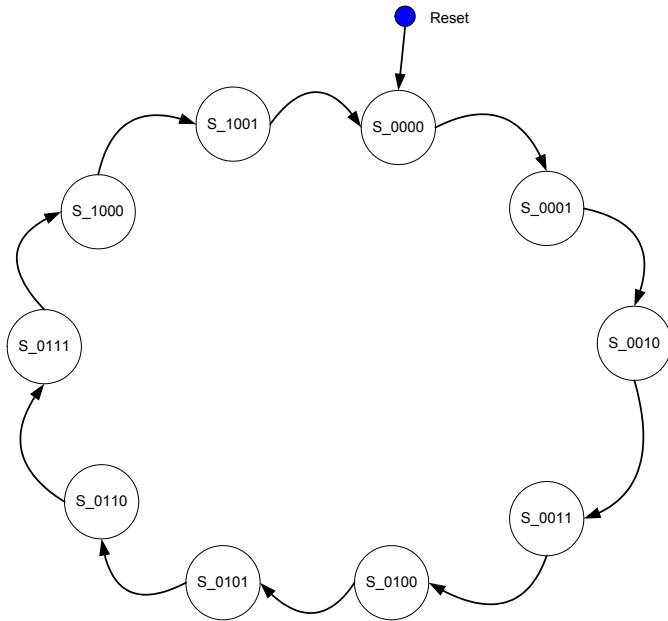
Source: Lee © 2000 Prentice-Hall Publishers, Inc.



Control Technique - Counter

- Counter method:

- Useful if we follow a cyclical state transition pattern.
- Cyclic counter, with period equal to the length of the counter length pattern, can be used in control logic.
- Output of the counter is the *sequence counter*.
- Counter can be coupled with random logic or PLD to generate appropriate control signals out of the controller block.
- Extra logic required for jumping if all states are not executed (e.g., Binary Up/Down, Modulo-N).

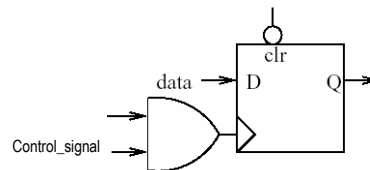


ASM Control – Exerted Across Threads

Source: Lee © 2000 Prentice-Hall Publishers, Inc.

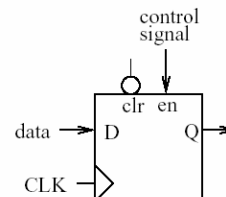
- Gated/User-defined clocking:

- We use some combination of signals to derive the clock signal for an ASM thread governing its state transitions and clocking of its data path registers.
- Nimbus only lets you specify a single user-defined signal on the clock.



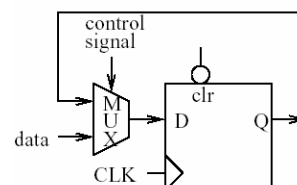
- Gated/User-defined resets:

- We use some control signal to reset the registers of a control block and its data path, either from the data path or from another controller.



- MUXed data path inputs:

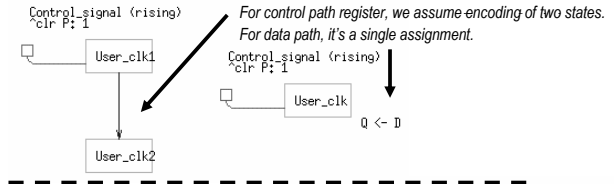
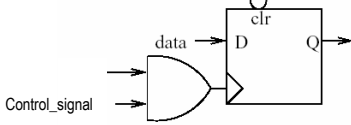
- We use control signals to “gate” the data into a register or data path stage by “implying” a MUX by the specification of an expression assignment to a particular bus in one or more states.



ASM Control – Exerted Across Threads

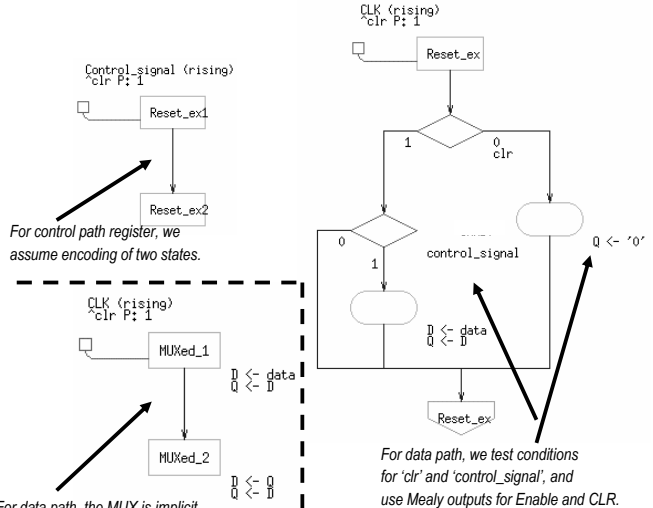
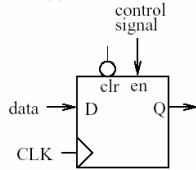
- User clocking:**

Figures: Lee © 2000 Prentice-Hall Publishers, Inc.

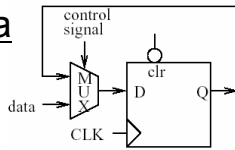


- User resets:**

- ✓ ASM models shown for both control and data path registers.



- MUX'ed data path inputs:**



For data path, the MUX is implicit, so you don't have to specify it. It will likely be synthesized into the circuit.

