What is Register Cell Design?

- In Chapter 7: Register transfers introduced
- There may be multiple transfers into a register
- In Chapter 8: Multiplier ASM output development involved defining control signals for a set of register transfers:
  - Collect together all transfers into a given register
  - Define control signals that cause each distinct transfer
  - Find an equation for each control signal in terms of inputs and ASM state
- Given the above, we need to design the register
- This design is done if possible by:
  - Designing a representative cell for the register
  - Connecting copies of the cell together to form the register
  - Applying appropriate “boundary conditions” to end cells
- Register cell design is the first step of the above process
Foundations for Register Cell Design

- Assuming that the register has a “hold” function when no transfers occur, use a D flip-flop plus multiplexer register with a LOAD control.
- The LOAD for the cell is the “OR” of all of the control signals under which a transfer into the register occurs.
- The D-input for the cell can be designed using a K-map for a small number of transfers and a multiplexer + encoder or CAFÉ for a large number of transfers.

Example 1: Register Cell Design

- Register A has the following transfers into it:
  - CX: A <- B v A
  - CY :A <- B XOR A
- Design register cell A_i for A.
  - What is the LOAD_i input for A_i?
  - What is the D_i input for A_i?
Example 1: (Continued)

- K-Map method for minimum logic:

Example 1: (Continued)

- Multiplexer + decoder method
Example 2: Register Cell Design

- Multiplexer + decoder method for register B
  - OR: $B \leftarrow A \lor B$
  - ADD: $B \leftarrow A + B$
  - AND: $B \leftarrow A \land B$

- Design register cell $B_i$ for B.
  - What is the $LOAD_i$ input for $B_i$?
  - What is the $D_i$ input for $B_i$?

Example 2: (Continued)