Complete 4-Bit ADC Design and Verification: Design a complete 4-bit 1 MSPS (mega-sample-per-second) successive approximation ADC given the following constraints and suggested design steps.

Design Constraints:

- Your final design should have the following I/O pins
  - Vin: input voltage, 0V to 1.0V
  - Dout: 4-bit parallel digital output, latched at 1MSPS
  - clk: one input clock for the ADC (any frequency you choose)
  - ibias: input current for biasing analog blocks
  - Vref: 1.0V reference voltage
  - Vdd and gnd: 3.3V and 0V: you can make these I/O pins or use global labels vdd! and gnd! internal
- Only the following external components can be used:
  - One ideal current source for ibias (should be tied between ibias and Vdd or gnd)
  - Three ideal voltage sources for Vref = 1.0V, Vdd = 3.3V, and Vin (variable, dc and transient, 0V to 1V)
  - One square wave voltage source for clk: operate at any frequency you choose, but the input should be sampled with corresponding digital outputs at 1 MSPS
  - Any other test sources or Verilog-A code desired for testing the ADC, but these components can not be required for ADC operation
- The complete design and layout should be verified through transient simulation and LVS and DRC.
  - For the digital logic, you can either manually place components into the schematic and layout from the CORELIB or use automated synthesis and place & route tools.

Suggested Design Procedure:

1. Design a 4-bit DAC using any of the techniques discussed in class or available in reference texts.
   - Start with ideal components, verify the concept using Verilog-A test code (see dac_tst_fix code from H1 solution on magellan, ams_5007_ref library or ahdlLib dac_dnl and dac_inl examples).
   - Realize analog components, verify through separate simulations then within the DAC.
   - Realize digital components as Verilog code or directly in hardware using CORELIB components.
   - Verify through simulation separately, then within the DAC. Verify complete DAC operation.
2. Design the 4-bit ADC
   - Design a successive approximation register (SAR) for your ADC using Verilog coding (or VHDL if you prefer). You can use the example code on magellan for H2 (H2_SAR). Verify your code using a digital simulator (Verilog-XL in Cadence on magellan or other tools if you prefer) and stimulus or test-bench code.
   - Design and verify the ADC architecture, with the SAR, ideal comparator, and your DAC (ideal or actual at this point). Verify using either an input ramp or Verilog-A code to sample at all 16 input levels at the ideal center of each input bin.
   - Realize the SAR through synthesis and the comparator using transistors. Verify each separately using Cadence analog simulations.
   - Verify the complete ADC as a pure analog simulation (with synthesized digital code).
3. Complete the full ADC layout
   - Layout the analog blocks and verify with DRC and LVS.
   - Use P&R and/or manual placement of digital components from the CORELIB library.
   - Combine the analog & digital blocks and wire together. Make the external I/O connections clear. Verify with DRC & LVS.

Turn in a design report with an overview of the ADC architecture, sub-component design, methods used for verification and description of results, full layout, and supporting waveforms attached as an appendix.

© 2005 R. Zane, University of Colorado at Boulder