Graded Exercise VLSI Design WS 25/26:

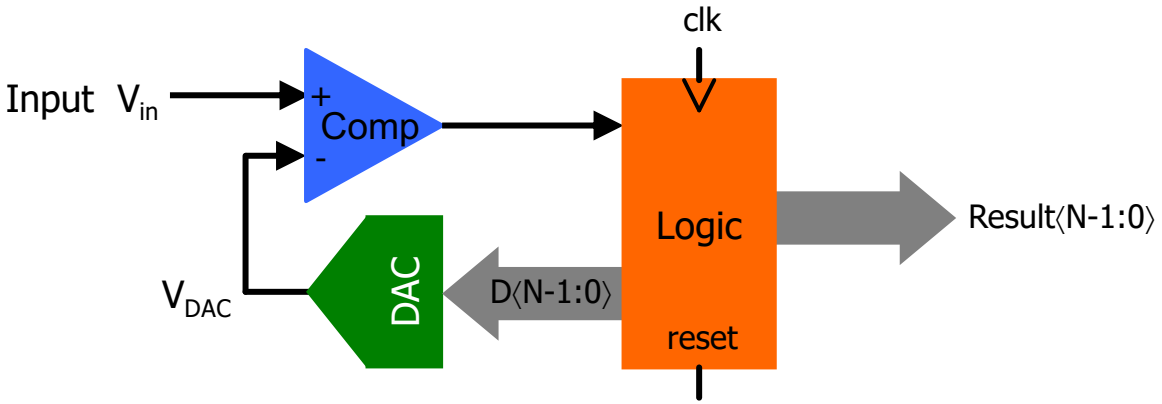
SAR ADC

Prof. Dr. P. Fischer

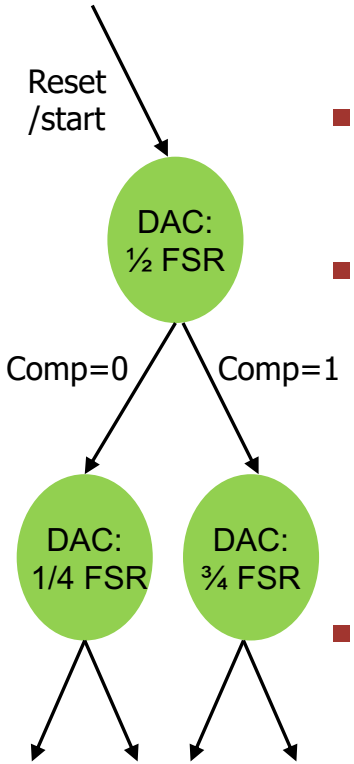
Lehrstuhl für Schaltungstechnik und Simulation
Uni Heidelberg

Successive Approximation ADC

- An Analogue – Digital – Converter (ADC) converts an analogue value (here a voltage) into a digital code (here binary).
- An N bit ADC provides an N bit result, distinguishing 2^N values.
- An N bit ADC using the 'successive approximation' (SAR) principle consists of
  - an N bit DAC (Digital – Analogue Converter) providing 2^N (voltage) values from (in our case) 0 V to some FSR (Full Scale Range)
  - an analogue comparator, comparing the input voltage V_{in} to the DAC output
  - a (clocked, digital) control logic which observes the comparator output and generates the DAC input word and the result



Operation

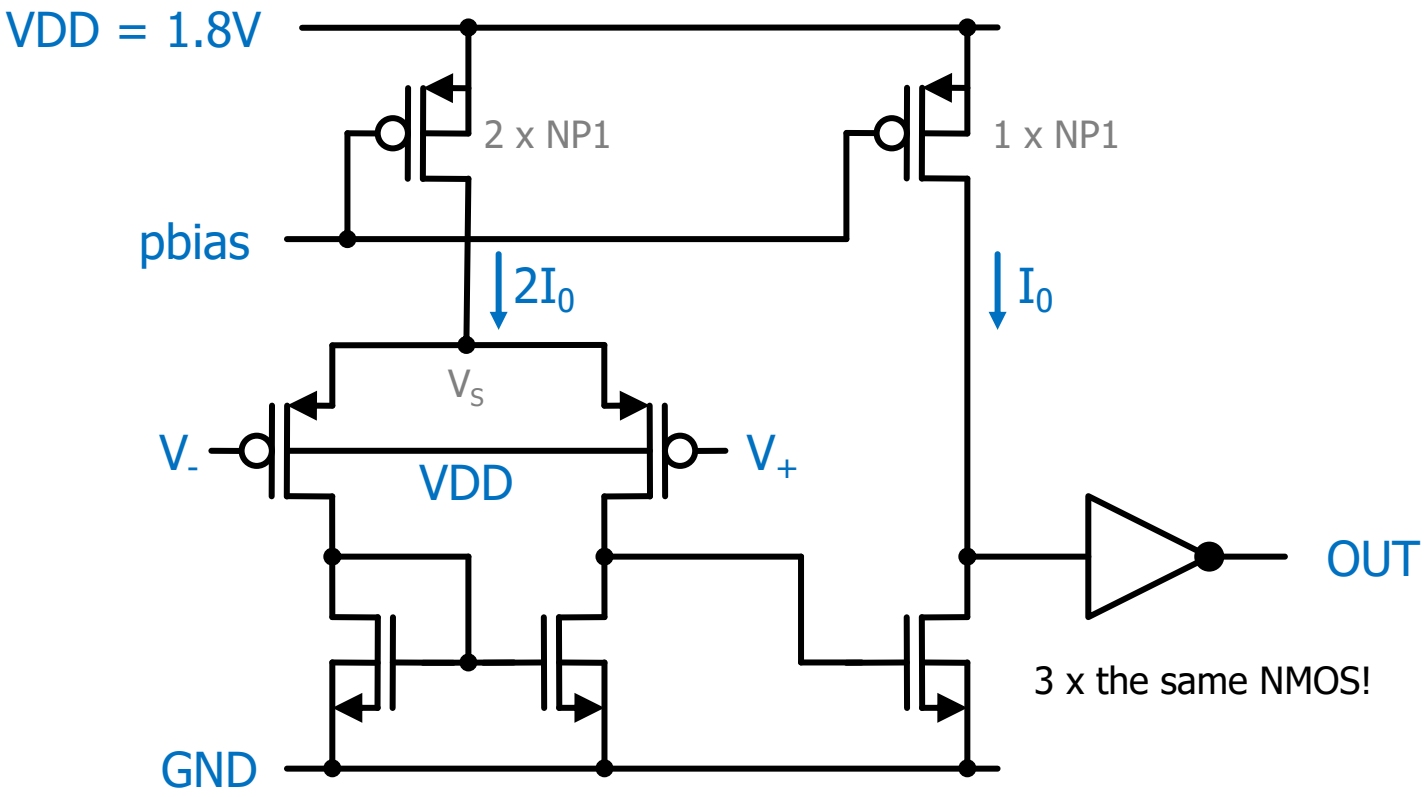


- One SAR conversion cycle requires N steps (clocks)
  - We assume that the input voltage V_{in} is constant during the cycle. In practice, a Sample-and-Hold (S&H) may be required at the input to 'freeze' a changing value.
- At the start of a cycle, the logic sets the DAC to ½ FSR
  - The comparator then finds out whether V_{in} is > or < than ½ FSR
- At the next clock edge, the logic inspects the comparator output (this is the most significant bit (MSB) of the result) and 'calculates' a new DAC value:
  - If V_{in} < FSR/2, the logic sets the DAC to ¼ FSR
  - If V_{in} ≥ FSR/2, the logic sets the DAC to ¾ FSR
- At the next clock edge, the next bit is extracted, and the DAC interval is again reduced by half, etc.
- With each clock, the difference between V_{in} and V_{DAC} is reduced to half and one more output bit is generated

THE BUILDING BLOCKS

The Comparator

- We use a very simple comparator (see CCS!) with
  - a (PMOS) differential pair
  - followed by a (NMOS) gain stage

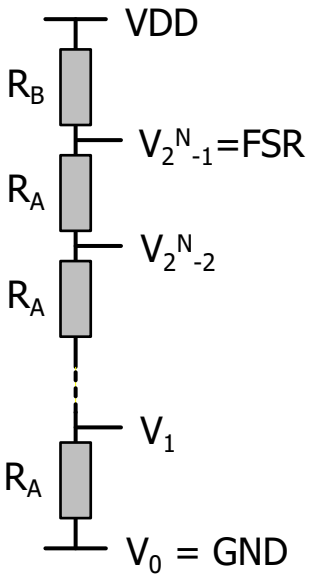


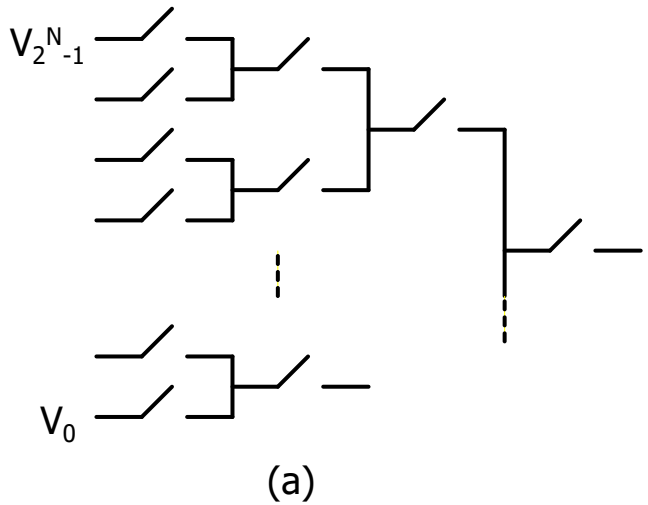
The Comparator

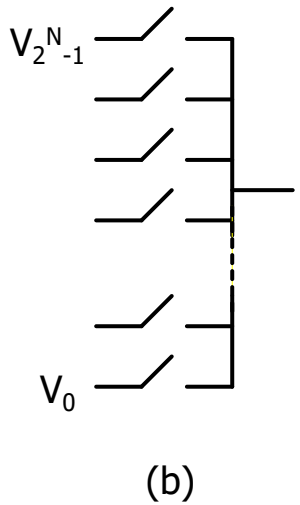
- The comparator is biased with a PMOS current source
  - Generate the bias voltage pbias by a current mirror!
  - Use a current of – for instance – 10µA.
  - For the sake of simplicity, generate the input current to the mirror with a resistor to ground.
- The second stage must be biased with *half the curre*nt of the first stage so that its threshold is reached for $V_+ = V_-$. Try to understand why!
  - This is achieved by using two PMOS for the differential pair, one for the second stage
- This comparator can handle voltages at its inputs from (just) ground up to *some* positive voltage, but not up to VDD.
  - Why does it help to connect the bulks of the pair to VDD?
- We therefore restrict the voltage range of our ADC (and thus of the DAC) to this maximal value 'Full Scale Range' FSR.

The DAC

- We use a voltage divider made out of 2^N-1 equal resistors R_A to generate all possible 2^N voltages.
  - We obtain FSR < VDD by an additional resistor R_B
- We then use analogue switches to implement a $2^N \rightarrow 1$ MUX which selects one of the voltages, depending on the DAC input code.
  - The MUX may be implemented as a binary tree (a), or using '1 hot encoded' switches to a bus (b)







The Graded Exercise

Task and Criteria for Grading

- In this exercise you should
  - Understand the SAR principle
  - Produce a well-structured *schematic hierarchy* of the design
  - Provide *analogue simulations* of comparator and DAC
  - Write a simple Verilog module for the control logic
  - Make a *mixed mode simulation* which shows correct conversion of a low, a medium and a high input voltage
  - Make DRC / LVS error-free *nice, compact layouts* of Comp. and DAC
- This should be documented in a write-up (~15-20 pages) with
  - An explanation and illustration of the SAR principle
  - A text justifying your design decisions
  - Text and screenshots of the simulation results, answering at least the question raised on the following slides
  - Screenshots and explanations of your schematics and layouts
- Use at least N = 5, better more
  - Ideally schematics and layout can be easily scaled to larger N….

General

- We use UMC180 nm
- Operate at 1.8V
- Start with MOS of 10u/0.5u for the comparator (quite large…). You can try to optimize later.
- Use small MOS for the decoder to limit capacitance, but estimate that their resistance is no issue.

- In the layout, try to match devices where necessary. You may want to place dummy devices where matching is relevant.

Part 1: Understanding the SAR principle

- Write down the DAC output codes for the conversion sequences of some input voltages.
- Find an algorithm / formula to generate the new DAC control word in the i-th step of the conversion.

- You may want to write a small program the check that your algorithm works. Try it on several cases!

Part 2: Simulating the Comparator

- Show that the comparator works:
  - Set one input to a fixed voltage V_1 and sweep (DC or transient) the other input from a bit below V_1 to a bit above V_1.
- Find out and understand for which voltage range of V_1 the comparator works (at decent speed)
  - Understand what limits the range (to low and high voltages!)
  - (Try to extend the range be sizing the transistors)
  - Make sure V_1 = GND is ok!
  - This fixes the FSR. Use a safe value.
- Determine the input offset voltage, i.e. at which input voltage difference the comparator switches.
  - Try different common mode voltages V_{CM} (as we will have them in the SAR application)
  - Verify that the input offset becomes worse if the 3 NMOS have not identical size or if the PMOS currents are not 2:1
  - Is the input offset small enough? Compare it to the LSB. What happens to the ADC if we have a fixed offset (independent of V_{CM})?

Part 2: Simulating the Comparator

- Check the speed:
  - The ‘precision’ of the comparator must ‘only’ be ~FSR / 2^N.
  - We can therefore define a figure of merit for ‘speed’ as the reaction time for a step input of this size around the threshold.
  - Does speed increase for higher bias current?
  - This speed sets your maximum clocking frequency!!!

Part 3: Simulating the DAC

- Chose the resistors such that the current in the DAC is not too high (compared to what?). Think at the layout size as well!
- Find arguments for the type of decoder to use and implement the schematic. Also think how a nice layout could look (also for large N)
- What type of MOS can be used for the switches? Or should you better use transmission gates?
- Use one switch and a decoder or multiple serial MOS to do the decoding?
- Simulate (best with a mixed mode simulation) that the decoder (and DAC) work as expected
- How long does it take to stabilize the output voltage for a large voltage step?
  - How does this depend on a load capacitance?
  - Which factors / components limit the speed? How?
  - Is the speed sufficient (compared to the comparator speed) when the capacitive load by the comparator is connected ?

Part 4: The full ADC

- Write a Verilog module for the SAR algorithm and simulate the full ADC
  - Make sure in your simulation that the input voltage is stable during one conversion

- Simulate several input voltages in one simulation
  - You may stimulate the input with a piecewise linear voltage source pwl, or superimpose voltages of several vpulse sources
  - You may also use ocean to script the simulation and then let it run for many voltages (smaller than the resolution).

- You may check that the ADC does *not* work any more if you run it too fast.
  - You could add an extra capacitance to the DAC output to artificially slow it down.

Part 5: Layout

- Make the layout of the DAC (with switches) such that
  - The resistance value can be changed easily without the need to re-do large parts
  - The number of bits can be increased relatively easily
- The layout of the DAC should
  - be reasonably compact
  - Not have an extreme aspect ratio, i.e. should be roughly square

- It may be clever to make a ‘unit cell’ which contains resistor and switch
- Your layout should be ‘regular’ for good matching of the resistors. You may even want to use dummy resistors at the ‘ends’.

- Add the layout of the compactor and make the LVS