RUPRECHT-KARLS-
UNIVERSITÄT
HEIDELBERG

Graded Exercise VLSI Design 2019:

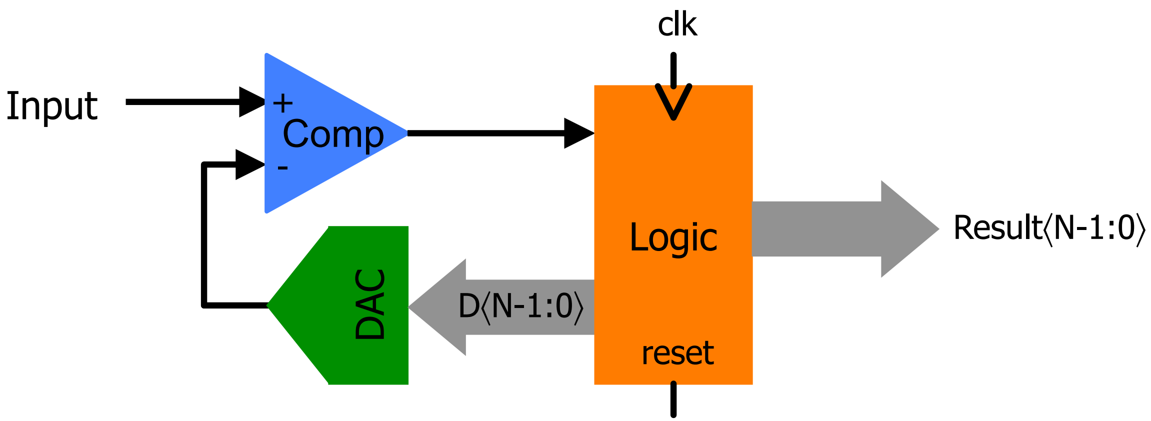
SAR ADC

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

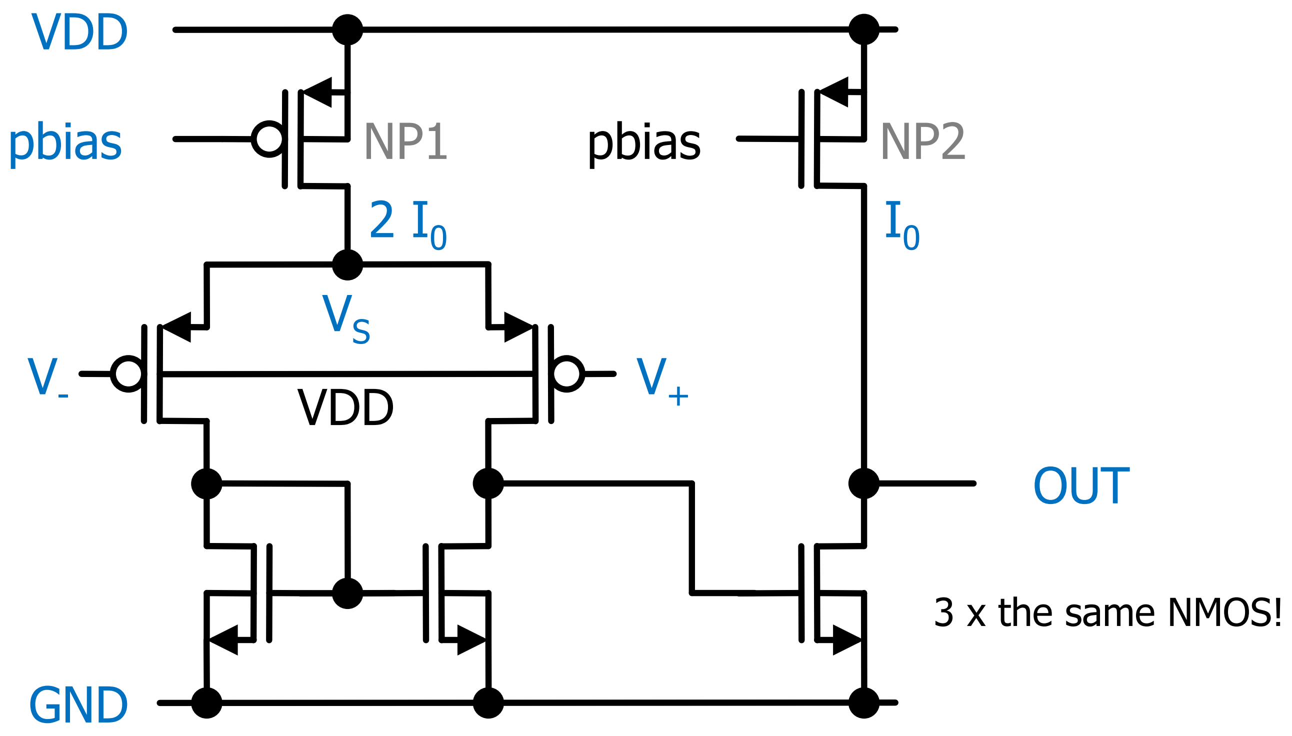


Operation

- The SAR ADC operates in conversion cycles of N clocks
  - We assume that the input voltage is constant during the cycle (in practice, a Sample-and-Hold 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 weather the input V_{in} is larger or smaller than this 'reference'
- At the next clock edge, the logic inspects the comparator output. This is the most significant bit (MSB) of the result.
  - If V_{in} < FSR/2, the logic sets DAC to ¼ FSR
  - If V_{in} ≥ FSR/2, the logic sets 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 output bit is generated

The Comparator

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



The Comparator

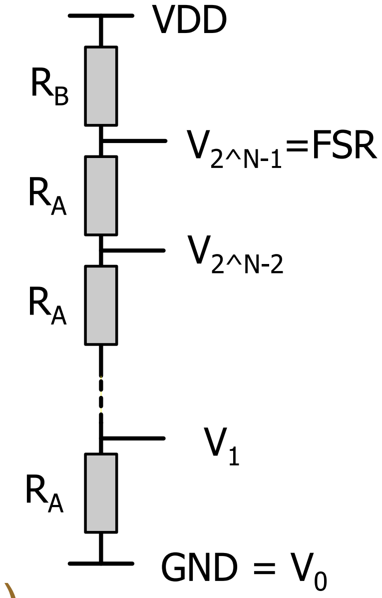
- The comparator is biased with a 'not too short' PMOS current source NP1 (L~1µm).
  - Generate the bias voltage pbias by a current mirror!
  - Use a current of – for instance – 10µA.
  - For the sake of simplicity, generate the input current to the mirror with a resistor to ground.

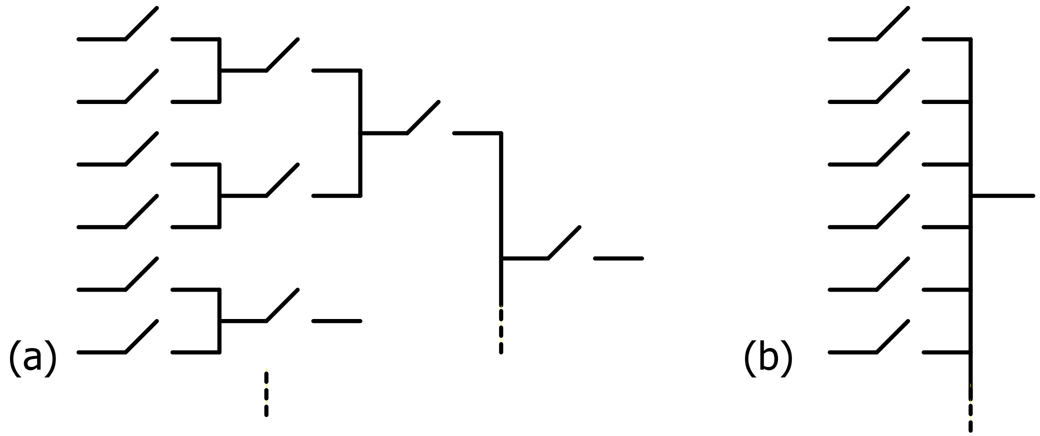
- The second stage must be biased with *half the curre*nt of the first stage so that its threshold is set to the condition $V_+ = V_-$. Try to understand why!

- This comparator can handle voltages at the inputs from (just) ground up to some positive voltage, but not up to VDD.
- We will therefore restrict the voltage range FSR of our ADC (and thus of the DAC) to this maximal value

The DAC

- We use a voltage divider made out of many equal resistors R_A with 2^N taps to generate all possible voltages.
  - Because FSR < VDD, we have 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 one switch and a decoder per resistor (b)





RUPRECHT-KARLS-
UNIVERSITÄT
HEIDELBERG

The Exercise

Goals 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 'nice', DRC and LVS error free layouts of Comparator and DAC
- These points should be documented in a short write-up with
  - a text describing your design decisions
  - text and screenshots of the simulation results, answering at least the question raised in the following
  - screenshots and explanations of your schematics layouts
- Use N = 5 or 6.

Exercise 1: Understanding the SAR principle

- Write down the DAC output codes for the conversion sequences of some input voltages.
- Define an algorithm 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 you algorithm works

Exercise 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)
  - Try to extend the range as much as you can be sizing the transistors.
  - Make sure V_1 = GND is ok!
  - This fixes the FSR
- 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!!!

Exercise 2 cont.: Simulating the Comparator

- Input offset and common mode effects:
  - Does the comparator really switch *exactly* at $V_- = V_+$?
  - Understand what happens!
  - Does this 'input offset' depend on the absolute input voltage level (The 'common mode input voltage') ?

- For our application, the input offset should be smaller than the LSB of our DAC…
  - What is the general effect of input offset on the transfer characteristic of the ADC ?

Exercise 3. Simulating the DAC

- Chose the resistors such that the current in the DAC is not too high (compared to what?)
- Find arguments for the type of decoder to use and implement the schematic
- Simulate (best with a mixed mode simulation) that the decoder works 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 ?

Exercise 4: The full ADC

- Write a Verilog module for the SAR algorithm and simulate the full ADC
  - Make sure 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
- 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.

Exercise 5: Layout

- Make the layout of the DAC such that
  - The resistance can be changed easily
  - The number of bits can be changed easily
- The layout of the DAC should be as compact as possible