Vorlesung 12

The topic of this lecture is the analog-to-digital converters (ADC). The topic is divided into:

Characterization of an A/D converter

INL, DNL, SNR, ENOB

Flash ADCs

Time-based ADCs

- Pseudodifferential comparator
- TDC
- Vernier TDC
- time interleaving

Algorithmic ADC (Optional)

Cyclic and Pipeline ADC (Optional)

Optional topics are not asked in the exam.

Analog-to-Digital Converter (ADC)

In modern circuits, most **signal processing is performed digitally**, while the input and output signals are typically **analog**.

An **analog-to-digital converter (ADC)** converts an analog signal into a **digital representation** by **discretizing both time and amplitude**.

- **Time discretization**: the analog signal is sampled at discrete time intervals.
- **Amplitude discretization**: each sample is assigned a **digital number D** corresponding to its amplitude.

One can plot the input voltage V_{in} of the ADC as a function of **D** and determine the mean values of V_{in} for each number D:

$$\mu\left(V_{in}(D)\right) \equiv D_{in} \qquad (1)$$

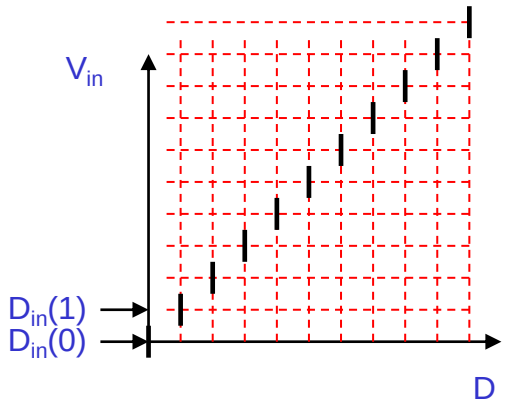


Figure 1: Determination of ADC characteristic curve $D_{in}(D)$

If D is the result of the AD conversion, $D_{in}(D)$ is a good approximation of the analog signal.

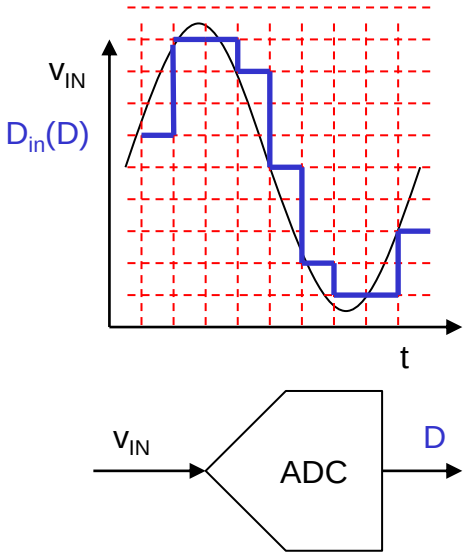


Figure 2: Function $D_{in}(D)$, where D is the result of A/D conversion

Many types of ADCs require only the following **analog components**:

- MOS switches
- Capacitors
- Comparators

ADCs are often implemented as **clocked switched-capacitor circuits**.

In addition to analog circuits, ADCs include **digital circuits**, which are critical for:

- Inverters and gated inverters
- Multiplexers
- Latches and flip-flops
- Counters

All of these digital components can be implemented on a **CMOS chip**. The design and operation of these circuits are covered in the **Digital Circuit Design (DCD)** course. In this lecture, we assume some prior knowledge of DCD. A **brief summary of the most important digital circuits** is provided in the appendix (Figures 32–35).

Circuits that combine **analog and digital components** are referred to as **mixed-signal circuits**.

Characterization of ADCs (Optional)

ADCs can be compared based on several key characteristics:

- **Resolution** – number of bits used to represent each sample
- **Sampling rate** – how fast the ADC can process input samples
- **Power consumption** – energy required per conversion (or time)
- **INL (Integral Nonlinearity)** – deviation of the ADC transfer function from a straight line
- **DNL (Differential Nonlinearity)** – variation in step size between consecutive digital codes
- **ENOB (Effective Number of Bits)** – effective resolution accounting for noise and nonlinearity

Accuracy depends on parameters such as **INL, DNL, and noise**.

Measurement procedure:

1. Apply a **precise input source**.
2. Sweep the input voltage V_{in} in **fine steps across the ADC's full input range**.
3. For each input voltage, read and store the ADC output **multiple times**.
4. For each digital output code **D**, collect the corresponding set of input voltages $V_{in}(D)$ that produced this code.
5. From these sets, calculate the **mean** and **standard deviation** of the input voltage for each code: $\mu(V_{in}(D))$ and $\sigma(V_{in}(D))$. These values describe the ADC's **linearity and noise performance** for each digital code.

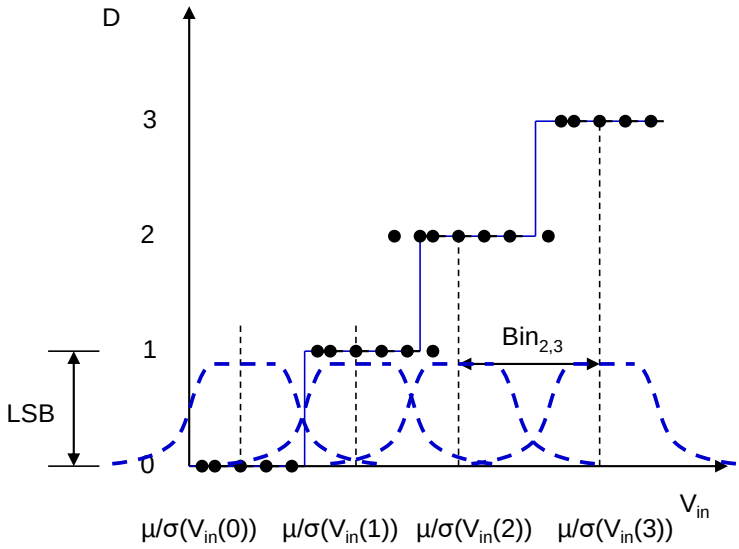


Figure 3: Representation of means and standard deviations of input voltages produced by four ADC digital outputs.

ADC Bin Width and Noise

The **bin width** of an ADC can be defined as the distance between the **average input voltages** corresponding to two consecutive digital output codes (Figure 3):

$$\mathrm{Bin}_{i,i+1} = \mu\left(V_{in}(i+1)\right) - \mu\left(V_{in}(i)\right) \qquad (2)$$

The **average bin width** over all codes is:

$$\mathrm{Bin} = \mu\left(\mathrm{Bin}_{i,i+1}\right) \qquad (3)$$

The **standard deviation** of $V_{in}(D)$ depends on **noise**.

- For an **ideal ADC** (Figure 4, top), the deviation of the input voltage from its mean, $V_{in}(D) - \mu(V_{in}(D))$, follows a **continuous uniform distribution**.
- In this case, the **width of the distribution** is equal to the **bin width**.

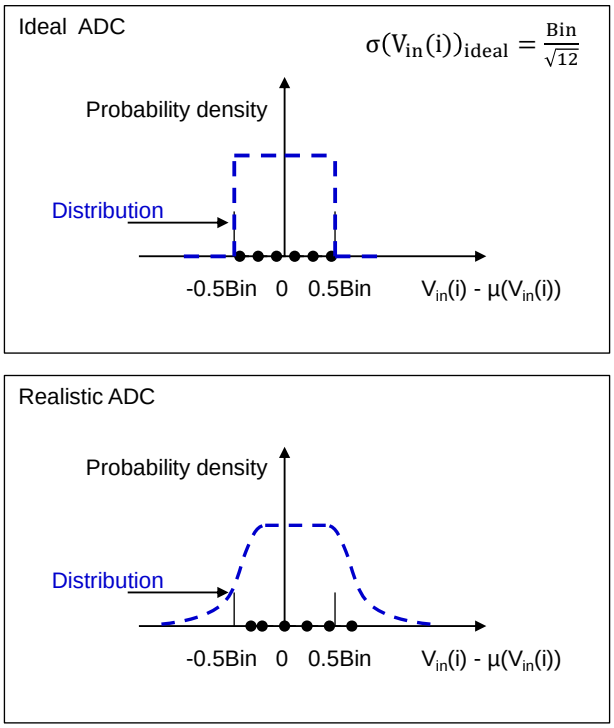


Figure 4: Distribution of measured values in the case of the ideal and realistic ADC

ADC Noise, SNR, and ENOB

For an **ideal ADC**, the **standard deviation** of the uniform distribution of input voltages corresponding to each output code is:

$$\sigma(V_{in}(D))_{ideal} = \frac{Bin}{\sqrt{12}} \qquad (4)$$

For a **realistic ADC** (Figure 4, bottom), the **mean standard deviation** is usually larger due to noise.

Signal-to-Noise Ratio (SNR) is calculated as:

$$SNR_{dB} = 20\log_{10}\left(\frac{V_{in,max}}{\sigma(V_{in})}\right) \qquad (5)$$

where:

$$V_{in,max} = \mu(V_{in}(D_{max})) - \mu(V_{in}(D_{min})) \qquad (6)$$

and
$\sigma(V_{in}) = \mu\sigma(V_{in}(i))$ is the **mean standard deviation** over all output codes.

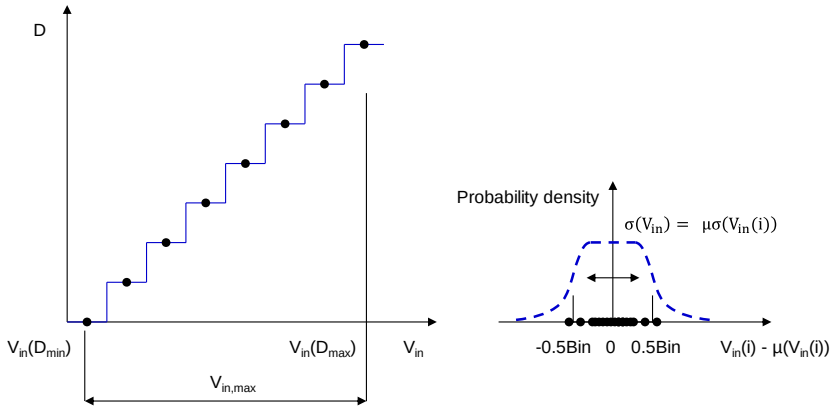


Figure 5: Maximum signal and noise

The effective number of bits is defined as follows:

$$ENOB = \frac{SNR_{dB} - 20\log_{10}\sqrt{12}}{20\log_{10} 2} = \frac{SNR_{dB} - 10.79}{6.021} \qquad (7)$$

Example: In the case of the ideal 8-bit ADC, the following applies:

$$\mu(V_{in}(D_{max})) - \mu(V_{in}(D_{min})) = 256 \times Bin$$

and

$$\sigma(V_{in}(D))_{ideal} = \frac{Bin}{\sqrt{12}}$$

It follows:

$$\mathrm{SNR_{dB}} = 20\log_{10}\left(256 \times \sqrt{12}\right) = 59.0\mathrm{dB}$$

The effective number of bits is:

$$\mathrm{ENOB} = \frac{59.0 - 20\log_{10}\sqrt{12}}{20\log_{10} 2} = 8$$

INL and DNL Measurement:

- For each output code D, determine the **midpoint input voltage**: $\mu(V_{in}(D))$.
- Example: For an 8-bit ADC, there are **256 points** ($\mu(V_{in}(D))$, D) with D=0, 1, …, 255.
- In real measurements, the points at D=0 and D=255 may be **less accurate**, because the input may exceed the ADC range.
- Fit a **line** through the points 0 < D < 255. The **reciprocal of the slope** gives the **average bin width** (Bin, equation 3).
- **INL (Integral Nonlinearity)** in LSBs is the **maximum vertical deviation** of each point ($\mu(V_{in}(i))$, i) from the fitted line (Figure 6).

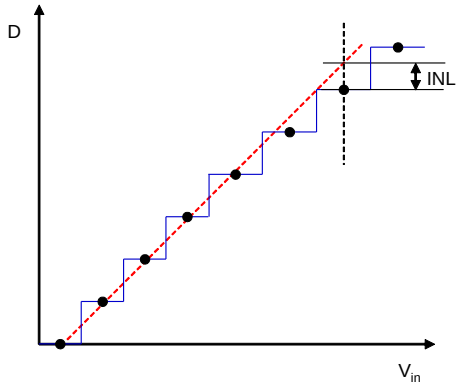


Figure 6: Integral nonlinearity

The **DNL** in LSBs is defined as the **maximum deviation of the step size** between consecutive ADC codes relative to the **average bin width**:

$$\mathrm{DNL} = \frac{\mu(V_{in}(i+1)) - \mu(V_{in}(i))}{\mathrm{Bin}} - 1 \qquad (8)$$

where Bin is the **average bin width** (equation 3).

- Graphically (Figure 7), DNL measures how uniform the **distance between consecutive input voltages** is.
- A perfectly linear ADC has **DNL = 0** for all codes.
- Large DNL values indicate **non-uniform steps**, which can cause missing codes or reduced linearity.

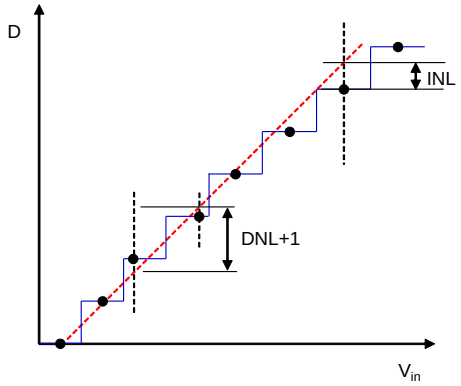


Figure 7: Differential nonlinearity

Relationship Between DNL and INL

- If the input voltage V_{in} for some digital codes D is **missing**, then the **DNL for those codes cannot be determined**.
- The **INL (Integral Nonlinearity)** for a given code can be expressed as the **sum of DNLs for all previous codes**:

This relationship shows that **small DNL errors accumulate**, producing the overall deviation of the ADC transfer function from the ideal straight line.

Flash ADC

Figure 8 shows the **block schematic of a 3-bit flash ADC**. The main components are:

1. **Comparators**: $2^n - 1$ comparators, where n is the number of bits.
2. **Encoder**: a combinational digital circuit with $2^n - 1$ digital inputs and n outputs.
    - The encoder produces the **binary number of the highest input that is equal to 1**.
    - If all comparator outputs are 0, the encoder output is also 0.
3. **Resistor ladder** – generates $2^n - 1$ **threshold voltages Th_i**.

Operation:

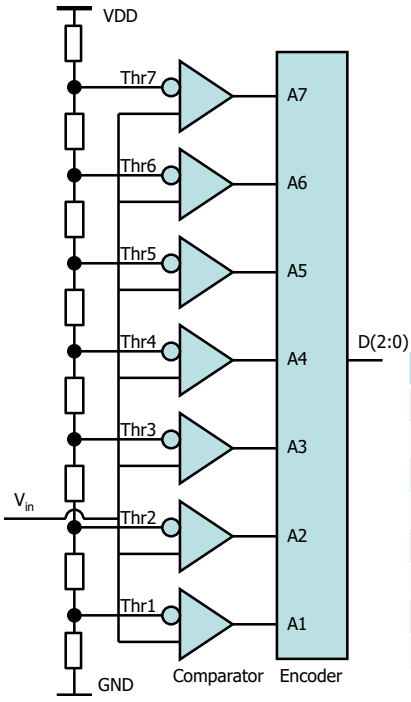
- When $Th_j > V_{in} > Th_{j+1}$, the comparators with indices $i < j$ produce **logic 1**, and those with $i > j$ produce **logic 0**.
- The comparator outputs are fed into the encoder, which generates the **n-bit binary output** $D(n-1 : 0)$.

Implementation:

- Comparators can be **clocked** or **static (continuous-time)** circuits.

Advantages and disadvantages:

- Flash ADCs typically have a **very high sampling rate**.
- The main disadvantage is the **large number of comparators**, which results in **high power consumption**.
- Typical resolutions for flash ADCs are **6–8 bits**.



A1	A2	A3	A4	A5	A6	A7	D2	D1	D0
0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	1
1	1	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	1	1
1	1	1	1	0	0	0	1	0	0
1	1	1	1	1	0	0	1	0	1
1	1	1	1	1	1	0	1	1	0
1	1	1	1	1	1	1	1	1	1

Figure 8: Block schematic of a 3-bit flash ADC

Time-Based ADCs

This type of ADC has traditionally featured low power consumption and very good resolution. In the past, however, it was considered relatively slow.

Recently, there has been significant and rapid development in these ADCs, as they particularly benefit from technology scaling. As a result, very high sampling rates can now be achieved.

Time-based ADCs operate according to the following principle (Figure 9):

A voltage-to-time converter (VTC) converts the input signal (voltage) into a time-domain signal. The resulting time signal is a pulse with an amplitude at the logic-1 level. The pulse duration (pulse width) is proportional to the input voltage:

$T_i = av_i + b$

In some implementations, two signals—start and stop—are generated, where the time difference between their rising edges is proportional to the input voltage.

The pulse duration is measured either using a counter or a time-to-digital converter (TDC).

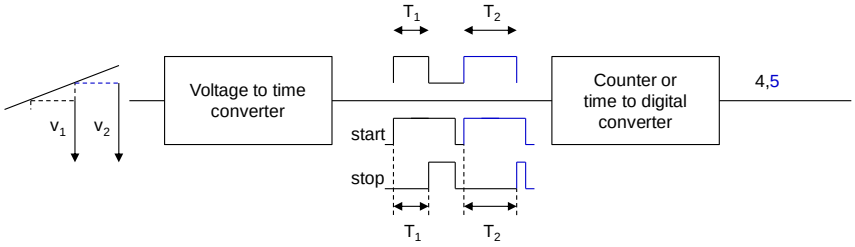


Figure 9 Time-based ADC

Let us first describe the VTC.

Voltage to Time Converter - VTC

The simplest circuit for voltage-to-time conversion (Figure 10) consists of a sampling circuit (switch **Sw** and capacitor **C**), a current source $\mathbf{I_{ramp}}$, and a comparator (**comp**).

The circuit operates either in the *track* state (**track = 1**) or in the *conversion* state (**track = 0**). A state machine generates the control signals for the circuit components.

When **track = 1**, the sampling capacitor is charged to the input voltage $\mathbf{V_{in}}$, so that $\mathbf{V_{cap} = V_{in}}$. The comparator threshold voltage $\mathbf{V_{thr}}$ is chosen to be relatively low, such that $\mathbf{V_{in} > V_{thr}}$.

When the *track* signal transitions to zero, the control signal **en** activates the NMOS current source $\mathbf{I_{ramp}}$. The capacitor **C** is then discharged, and the voltage $\mathbf{V_{cap}}$ decreases linearly. When $\mathbf{V_{cap}}$ falls below the threshold voltage $\mathbf{V_{thr}}$, the comparator output **comp** switches from 0 to 1.

The logical AND of the inverted comparator output and the signal **en** forms the output signal of the converter. The signal **en** can be interpreted as the *start* signal, while the comparator output serves as the *stop* signal.

The pulse width **T** of the output signal is given by:

$$I_{ramp} = \frac{C(Th - V_{in})}{T}; T = \frac{C(Th - V_{in})}{I_{ramp}} \qquad (9)$$

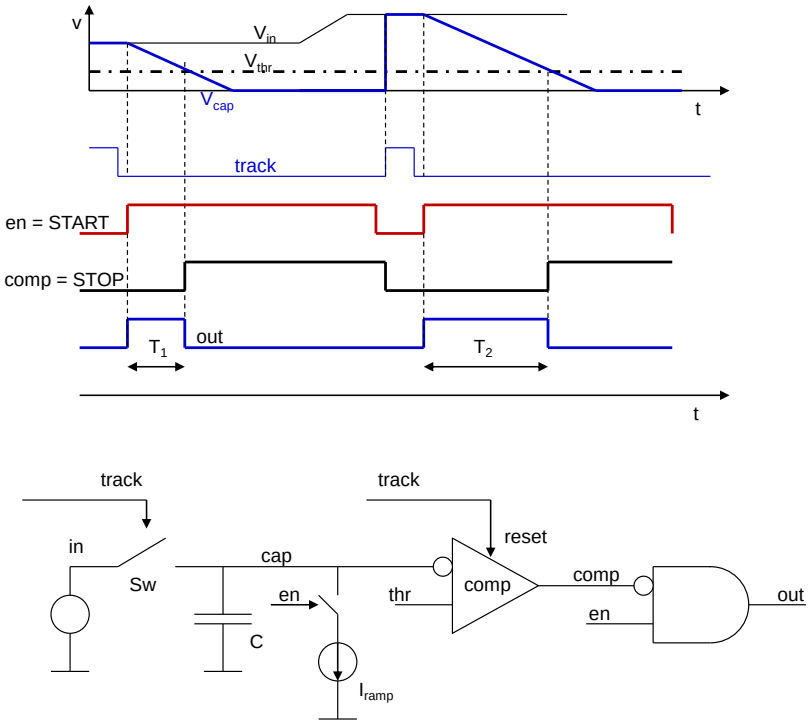


Figure 10: Voltage-time converter

Design Requirements

The input signal must remain above the threshold voltage $\mathbf{V_{thr}}$. To achieve a large input signal range, $\mathbf{V_{thr}}$ should be chosen as low as possible, while $\mathbf{V_{in}}$ can reach values up to **VDD**. The sampling switch **Sw** can be implemented using a PMOS transistor; alternatively, a bootstrapped NMOS switch may be used to improve linearity.

The comparator should be as fast as possible. A two-stage operational amplifier can be used as a comparator (Figure 11). However, this implementation has several disadvantages:

1. $\mathbf{V_{thr}}$ must not be too low; otherwise, the bias current source $\mathbf{I_{bias}}$ may leave saturation.
2. The current source $\mathbf{I_{bias2}}$ counteracts the output transition, which slows down the comparator response.
3. After returning to the *track* state, a relatively long time is required for the comparator output to reset to 0.

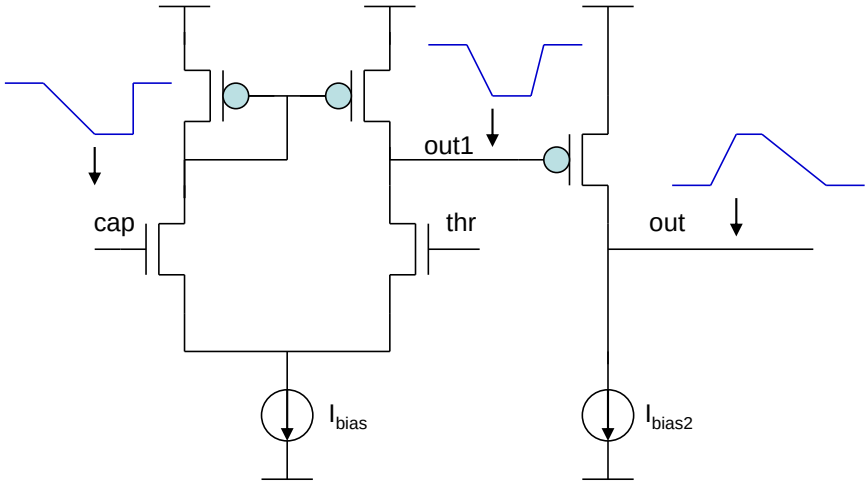


Figure 11: Standard comparator

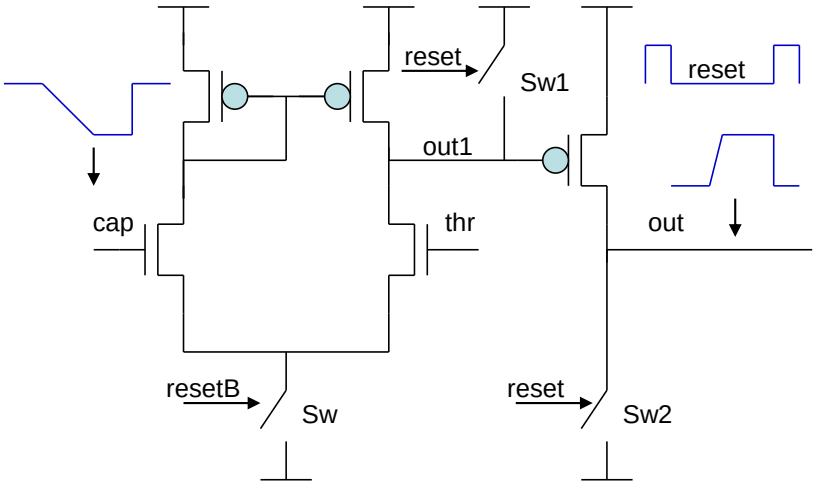


Figure 12: Pseudo-differential comparator

A comparator with a reset function (Figure 12) is often used. In this topology, the bias current sources $\mathbf{I_{bias}}$ and $\mathbf{I_{bias2}}$ are replaced by switches **Sw1** and **Sw2**. The switches **Sw1** and **Sw2** are closed during the *track* state, forcing the initial conditions $\mathbf{V_{out1}}$ **= VDD** and $\mathbf{V_{out}}$ **= 0**.

During the *conversion* phase, switch **Sw** is closed. Since the differential pair does not rely on a constant current source but instead uses a switch, the threshold voltage $\mathbf{V_{thr}}$ can be set to a lower value. Differential amplifiers whose input transistors have their sources connected to ground are referred to as *pseudodifferential amplifiers*.

The pseudodifferential comparator offers higher speed, as the effective bias current in the differential pair is larger and $\mathbf{I_{bias2} = 0}$. In addition, its output can be rapidly reset to 0 during the *track* phase.

A high-quality $\mathbf{I_{ramp}}$ current source (Figure 10) is critical to the performance of the ADC. Ideally, $\mathbf{I_{ramp}}$ should be as independent as possible of $\mathbf{V_{cap}}$, which implies that the output resistance of the current source must be large. A cascoded current source is therefore often a good choice.

The noise of the current source must also be minimized. The current noise power is proportional to the transconductance $\mathbf{g_m}$, so a current source with a small $\mathbf{g_m}$ is preferable from a noise perspective. However, a low transconductance implies a larger saturation voltage $\mathbf{V_{DS,sat,ramp}}$.

Consequently, the requirements for good linearity over a large signal range and for low transconductance are contradictory, leading to an inherent design trade-off.

Resolution and noise (optional)

The dominant error source is the noise in the ramp current $\mathbf{I_{ramp}}$. This noise is described by its power spectral density. For a MOSFET current source, the current noise spectral density is given by:

$$S_I = \frac{2}{3}4KTg_m = \frac{\frac{2}{3}4KT2I_{ramp}}{V_{dssat}} = \frac{\frac{16}{3}eU_TI_{ramp}}{V_{dssat}} \qquad (10)$$

It can be shown that if a capacitor **C** is charged with the noisy current $\mathbf{I_{ramp}}$ for a duration **T** (Figure 13, left), the capacitor voltage after time **T** has a variance given by:

$$\sigma^2(v_{cap}) = \frac{1}{C^2}\sigma^2(Q) = \frac{1}{C^2}\sigma^2(TI) = \frac{T^2}{C^2}\sigma^2(I) = \frac{T^2}{C^2}\frac{1}{2}\frac{S_I}{T} = \frac{1}{2}\frac{S_IT}{C^2} \qquad (11)$$

In a time-based ADC, the situation is different (Figure 13, right). The time **T** required to discharge the capacitor from $\mathbf{V_{in}}$ to $\mathbf{V_{thr}}$ is measured. The threshold voltage $\mathbf{V_{thr}}$ is assumed to be known and noise-free. The relevant question is therefore: *What is the uncertainty σ(V_{in}) given a measured time **T**?*

Since the capacitor voltage exhibits an uncertainty $\sigma(v_{cap})$ after charging for time **T** (Equation 11), the estimated input voltage $\mathbf{V_{in}}$ has the same uncertainty:

$$\sigma^2(v_{in}) = \frac{1}{2}\frac{S_IT}{C^2} \qquad (12)$$

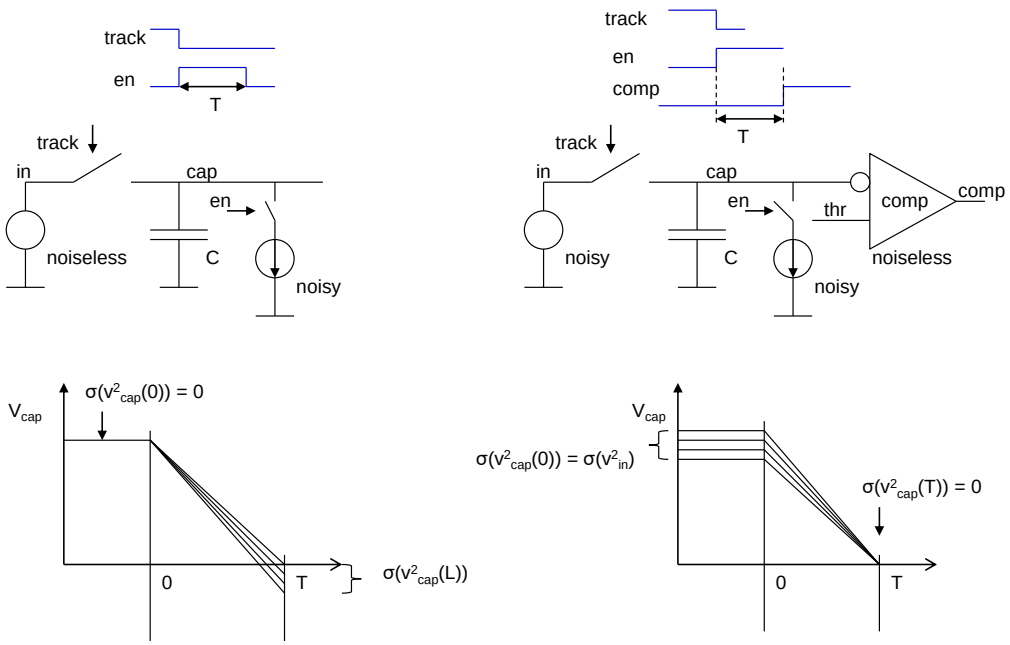


Figure 13: Left: Capacitance C is discharged for duration T with the noisy current I_{ramp}. Right: Capacitance C is charged to a voltage V_{in} and discharged with I_{ramp} until its voltage becomes equal to V_{thr}. The time T is measured. If a certain time is measured, what is the uncertainty in V_{in} measurement?

Signal-to-Noise Ratio

The following relationship holds:

$$\mathrm{SNR} = \frac{v_{in,max}}{\sigma(v_{in})}$$

The maximum signal can be calculated in the following way:

$$\frac{T_{max} I_{ramp}}{C} = \frac{Q_{max}}{C} = v_{IN,max} - V_{thr} = v_{in,max}$$

Substituting the expression for v_{in} uncertainty from Equation (12), we obtain:

$$\mathrm{SNR} = \frac{v_{in,max}}{\sigma(v_{in})} = \frac{\frac{Q_{max}}{C}}{\sqrt{\frac{1}{2}\frac{S_I T_{max}}{C^2}}} = I_{ramp}\sqrt{\frac{2T_{max}}{S_I}} = \sqrt{\frac{Q_{max} V_{dssat}}{\frac{8}{3} e U_T}} \quad (13)$$

Example: For the parameters C = 100 fF, $v_{in,max}$ = 0.8 V, V_{dssat} = 200 mV, we obtain using (13) SNR = 1200 and with (8) ENOB = 8.436.

Ramp ADCs can achieve high accuracy. Other advantageous features include low power consumption and a small layout area.

We now proceed to describe the circuit used for time-to-digital conversion.

Time Measurement with a Counter

The simplest circuit for time measurement is based on a counter.

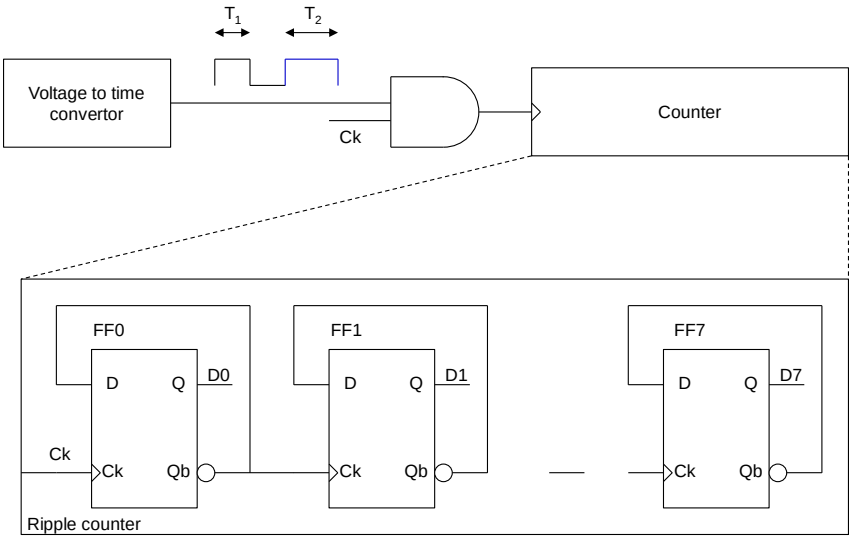


Figure 14: Time measurement with a ripple counter

The counter increments as long as the output of the voltage-to-time converter (VTC) remains at logic 1. The counter value **n** therefore corresponds to the measured time interval **T**:

$T = nT_0$

Where T_0 is the clock period.

One limitation of this approach is the conversion speed. For an 8-bit ADC, the maximum conversion time is:

$$T_{max} = 2^8 T_0$$

It is difficult to achieve a clock period $\mathbf{T_0}$ shorter than about 250 ps. Consequently, the maximum conversion time for an 8-bit ADC is approximately:

$256 \times 250ps = 64ns$.

This raises the question: *Why is the clock speed limited?*

A counter is composed of memory elements (flip-flops) and combinational logic that computes the next digital state. The structure and operation of flip-flops and counters were discussed in the DCD course.

The minimum achievable clock period is approximately equal to the sum of the propagation delay of the flip-flops and the delay of the combinational logic.

The following chapter is optional.

Let us now determine these delays more precisely. The combinational logic of a binary 10-bit counter can be quite complex. The simplest implementation is a ripple counter (Figure 14). In this counter, the inverted output $\mathbf{Q_b}$ of the least significant bit flip-flop (**FF0**) is connected to its data input **D**.

The minimum clock period is approximately equal to the sum of all delays along the signal path from the clock input to the flip-flop output and through the feedback path back to the flip-flop input.

The internal structure of flip-flop **FF0** and the corresponding signal path (shown in blue) are illustrated in Figure 15. Along this path, there are eight inverters and gated inverters. Consequently, a total of eight internal nodes, with their associated capacitances, must be switched between **GND** and **VDD**.

The smallest delay element is an inverter driving another minimum-size inverter, which represents a capacitive load $\mathbf{C_{load}}$. We denote this delay by $\mathbf{T_0}$. The propagation delay of an inverter can be approximated as: $4\ C_{load}\ R_{on}$.

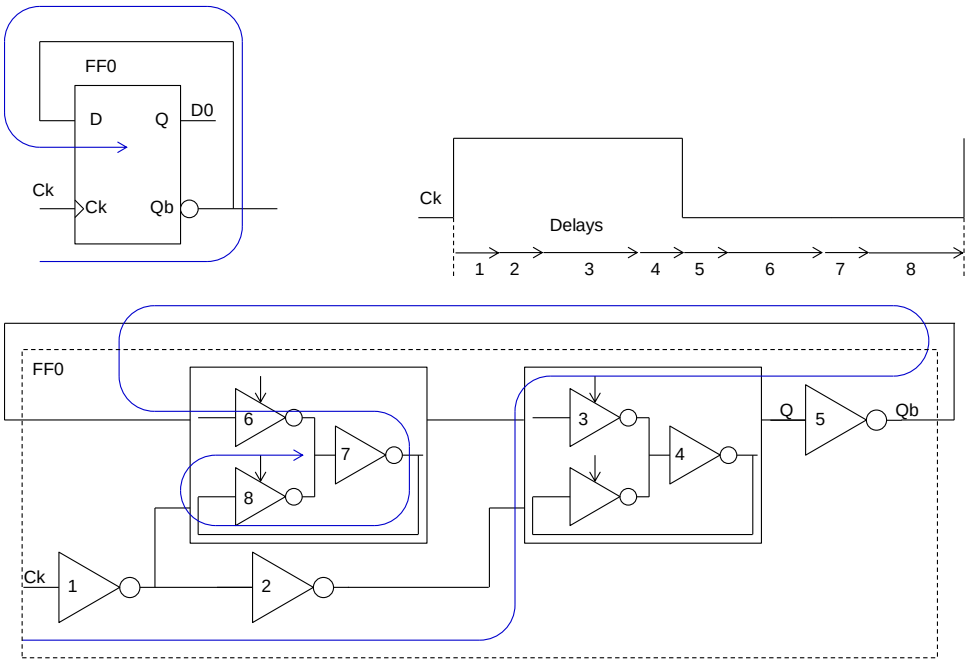


Figure 15: The structure of the FF0 flip-flop and the signal path (blue)

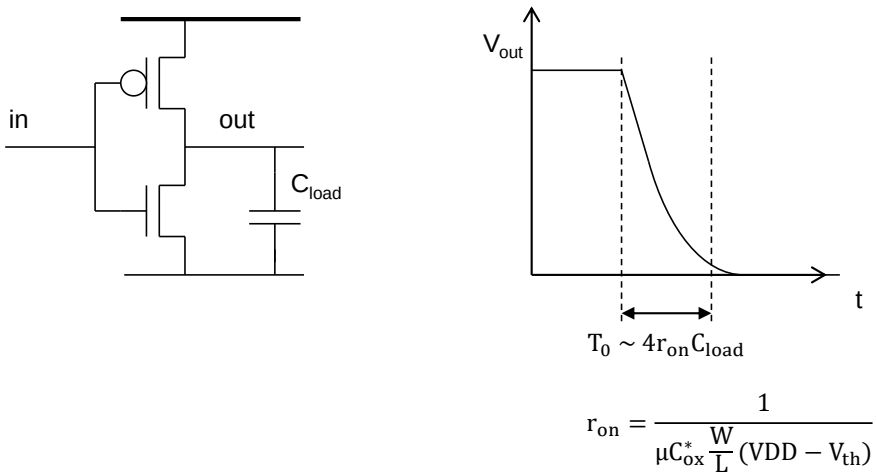


Figure 16: Discharge time at the output of an inverter

The effective on-resistance $\mathbf{R_{on}}$ depends on the circuit element:

- Inverter: $\mathbf{R_{on} = R_{MOSFET}}$
- Gated inverter: $\mathbf{R_{on} \approx 2R_{MOSFET}}$

The load capacitances $\mathbf{C_{load}}$ also differ between nodes. Taking these effects into account, the total delay along the data path of the ripple counter is approximately **10 $\mathbf{T_0}$**. In practice, this delay is typically greater than 250 ps.

In modern CMOS technologies (feature size **L < 65 nm**), $\mathbf{T_0}$ is of the order of 25 ps or less.

Time to Digital Converter - TDC

Conversion speed can be significantly improved by using a **Time-to-Digital Converter (TDC)**. TDCs can be implemented in several ways. One common approach is analog: a capacitor is charged during the pulse duration, and the resulting voltage is measured with an ADC.

Of greater interest here are **digital TDCs**. One approach is to use the fast propagation delay of an inverter or a pair of inverters in series to measure time. Figure 17 illustrates such a circuit. The core of the TDC is a **delay line** made of a chain of buffers.

The input of the first buffer (**B1**) is connected to the start signal **START**. When the rising edge of START occurs, the buffers propagate the signal sequentially, similar to dominoes falling. The rising edge of the **STOP** signal then latches the current states of all buffer outputs. Fast memory elements—**latches** (L1–L7)—are used for this purpose.

In this way, the delay between START and STOP can be measured with a **bin width of T_0**, achieving roughly **10× better resolution** than a simple counter.

To integrate this TDC into our ADC, the **voltage-to-time converter (VTC)** must generate both START and STOP signals.

The delay line must be sufficiently long. For an 8-bit resolution, we require $2^8-1=255$ buffers. The stored outputs form a **thermometer code**, similar to that used in flash ADCs.
An **encoder** converts this thermometer code into an 8-bit binary value.

Figure 17 shows a 3-bit example of a TDC, while Figure 18 illustrates the corresponding signal waveforms.

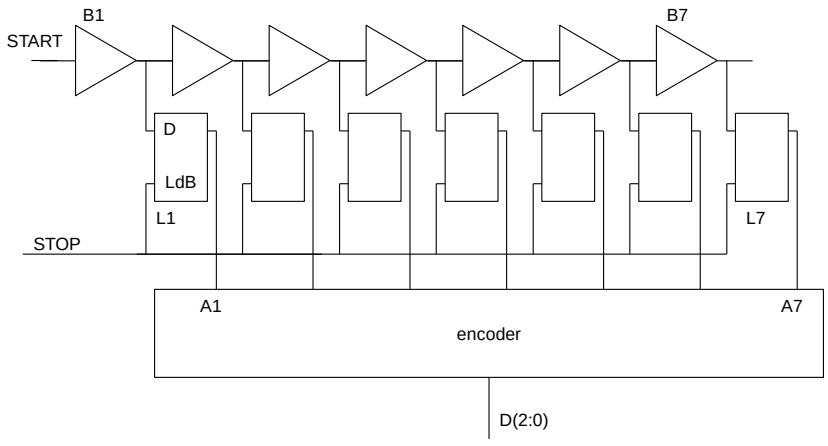


Figure 17: Time to digital converter

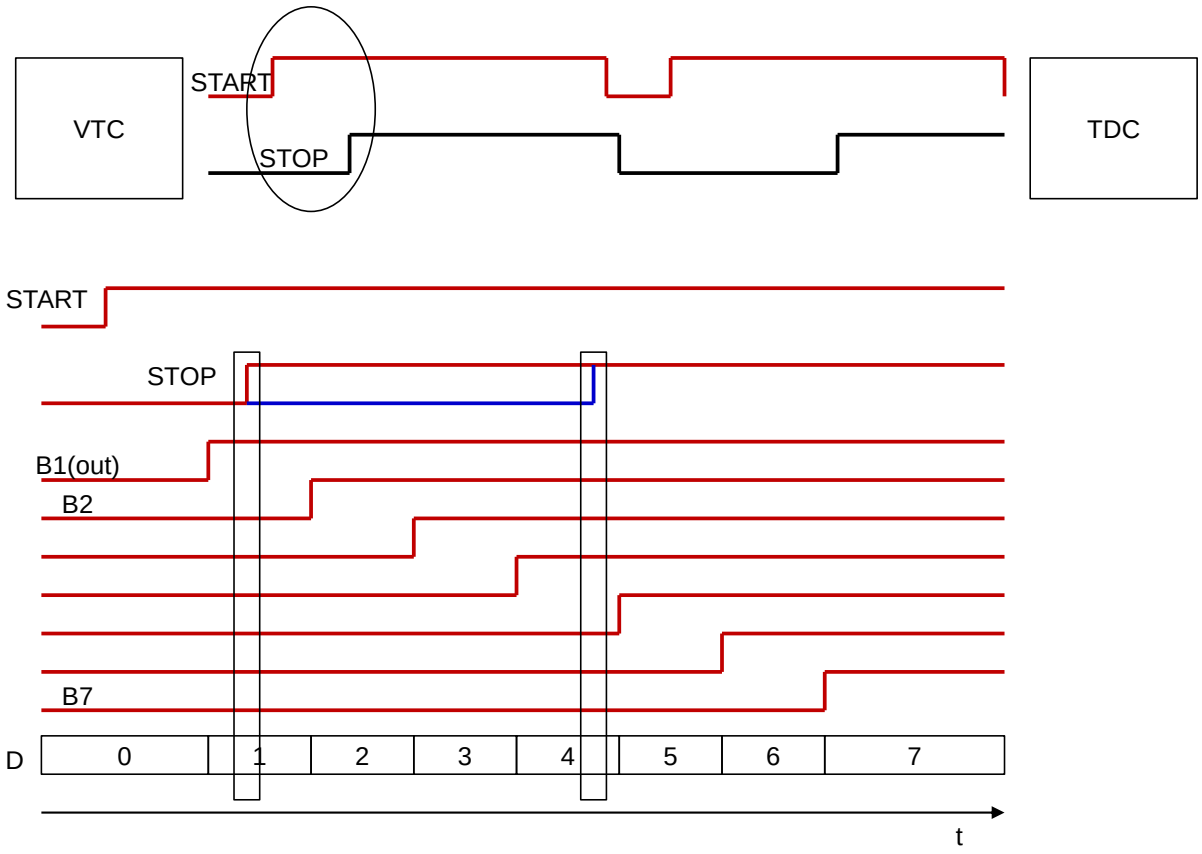


Figure 18: Signal waveform in a TDC. The red stop-signal stores code 1, the blue stop-signal stores code 4.

Figure 19 shows the structure of the latch circuit.

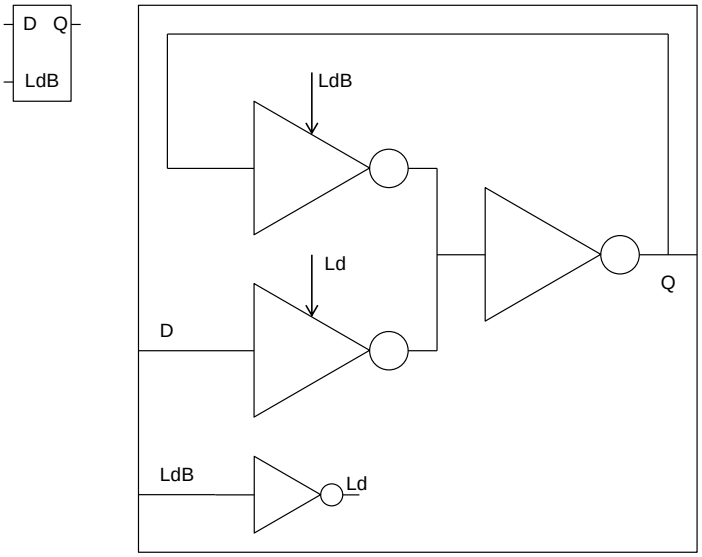


Figure 19: Structure of the latch circuit

High-Resolution TDCs

How can we further improve the bin width of a TDC? One interesting solution is the **Vernier TDC**.

(Historical note: Pierre Vernier, 19 August 1580 – 14 September 1637, was a French mathematician, director general of economy in the County of Burgundy, captain and castellan of the castle at Ornans, and counsellor to the King of Spain. The vernier principle is named after him.)

The Vernier TDC uses **two buffer chains**. The chain driven by **START_V** (SB1–SB7) contains slightly slower buffers, each with a delay of $T_0 + \Delta T$. The chain driven by **STOP_V** contains faster buffers, each with a delay of T_0.

Each memory cell Li stores a logic 1 only if the output of the slower chain (**SBi**) transitions to logic 1 before the corresponding output of the faster chain (**Si**).

The principle is analogous to a **vernier scale**, where the relative offset between two scales allows finer measurement (Figure 21). Using this technique, the effective bin width can be made much smaller than the delay of a single buffer, enabling higher TDC resolution.

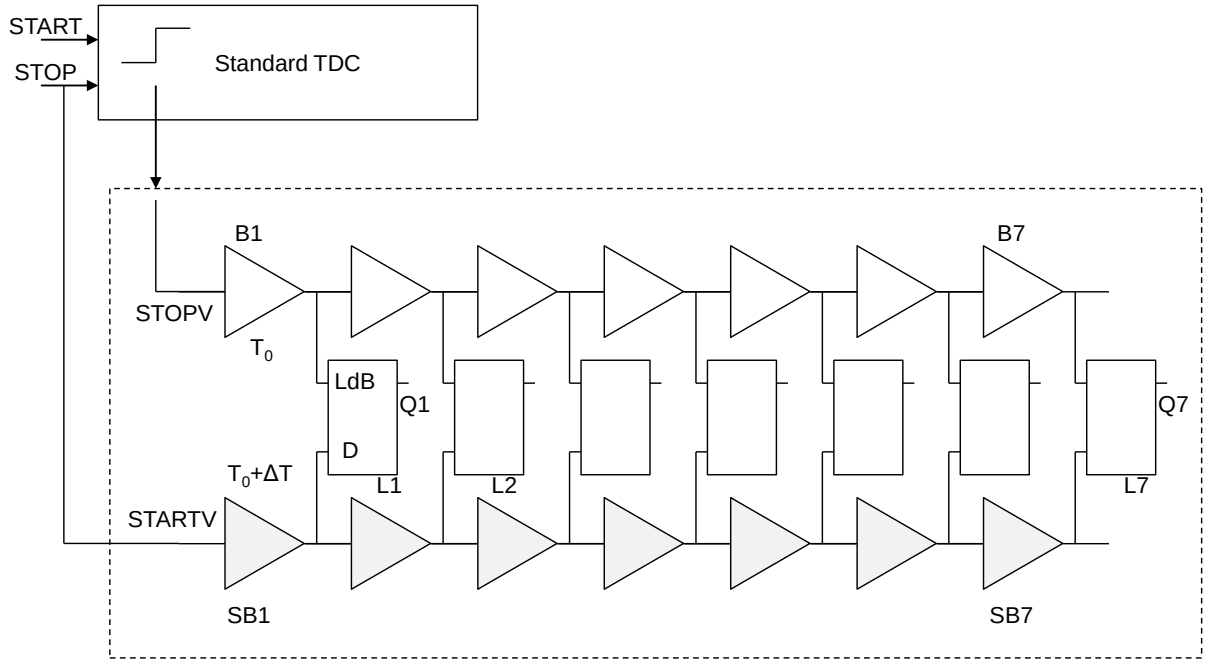


Figure 1: Vernier-TDC

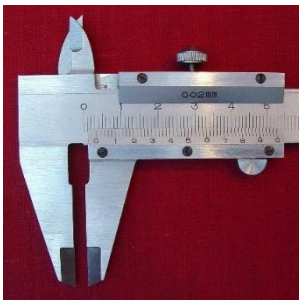


Figure 21: Vernier scale

The **Vernier TDC** is always used in combination with a **standard TDC** to achieve both coarse and fine time measurements. Figure 22 illustrates the signals.

The **start signal** for the Vernier TDC (**START_V**) is the **stop signal** of the standard TDC:

START_V = STOP

The Vernier TDC begins its measurement **after the standard TDC has stopped**.

The **stop signal** of the Vernier TDC (**STOP_V**) corresponds to the first signal edge in the standard TDC's buffer chain after its stop signal (see Figures 22 and 23).

In this combined setup:

- The **standard TDC** performs the **coarse measurement**.
- The **Vernier TDC** performs the **fine measurement**.

In the example shown in Figure 22:

- The standard TDC's bin width is $T0=8 \times \Delta T_0$.
- Both the standard and Vernier buffer chains contain 8 buffers.

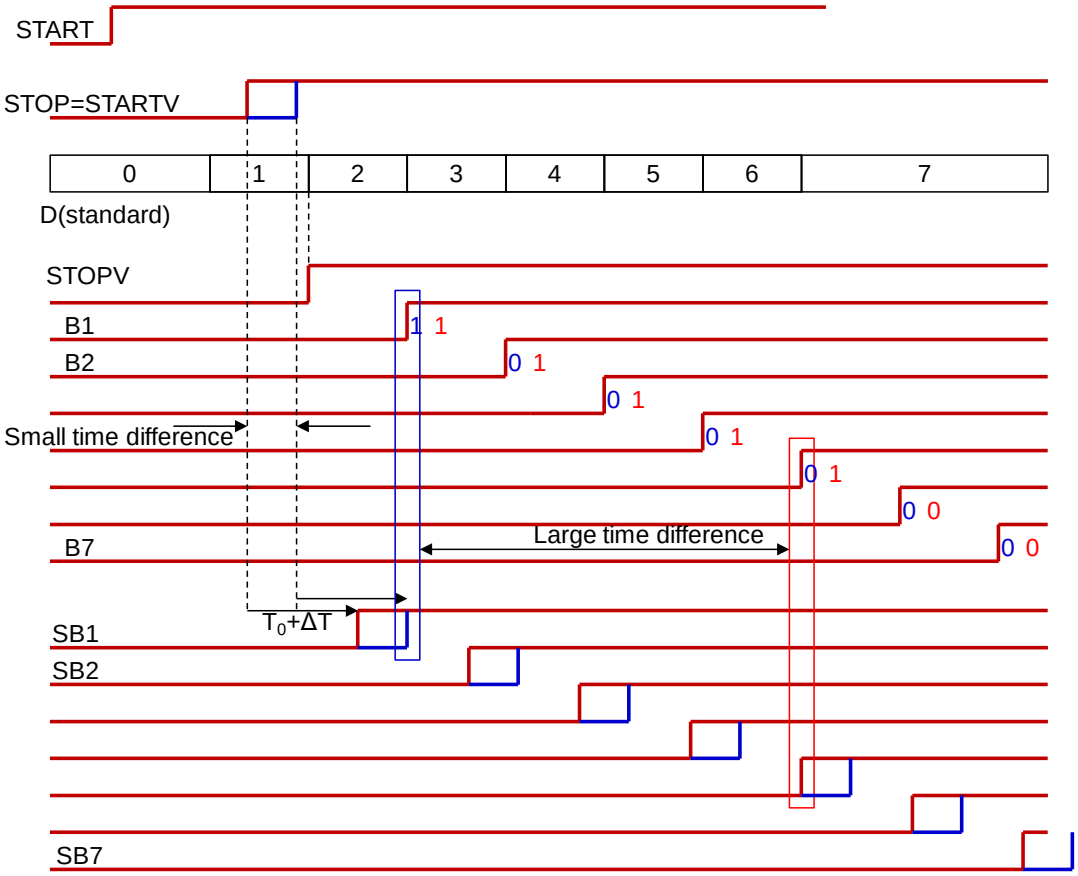


Figure 22: Signals in the Vernier TDC

Example of Vernier TDC Measurement

Figure 22 shows two possible **STOP signals**. The time difference between them is less than T_0, so **both signals produce the same result in the standard TDC** (coarse measurement = 1).

In the case of the **red STOP signal**:

- The slower chain starts earlier.
- Five logic 1s are stored in the Vernier TDC:

Q(1:7) = 1111100

- This corresponds to the fine measurement before the faster chain overtakes the slower one.

In the case of the **blue STOP signal**:

- The slower chain starts later.
- Only the first bit is stored as 1:

Q(1:7) = 1000000

- The faster chain overtakes the slower chain after the first buffer.

In both cases, the **bin width of the Vernier TDC** is ΔT_0

which is much smaller than the standard TDC bin width T_0, providing higher resolution for the fine measurement.

Complete Vernier TDC Circuit

Figure 23 shows the **complete TDC circuit**, including the circuit for generating **STOP_V**.

The STOP_V generation circuit uses flip-flops **DFF1–DFF7**:

- The **clock (CK) inputs** of these flip-flops are connected to the outputs of the standard TDC buffer chain.
- The **D-input** of each flip-flop is connected to the **STOP** signal.

The **STOP_V** signal is obtained as the **OR function of all flip-flop outputs**.

STOP_V is thus **synchronous with the first buffer output** that transitions from 0 to 1 after the STOP signal. This ensures that the fine measurement is correctly aligned with the standard TDC coarse measurement.

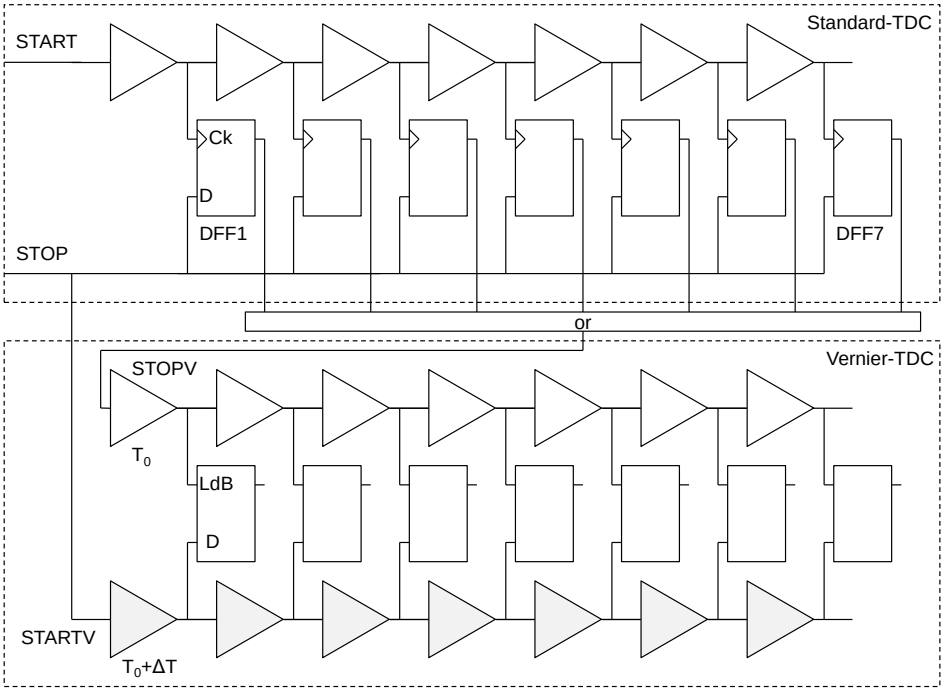


Figure 23: Complete circuit with the standard and Vernier TDC. Circuit for generating STOP_V is shown.

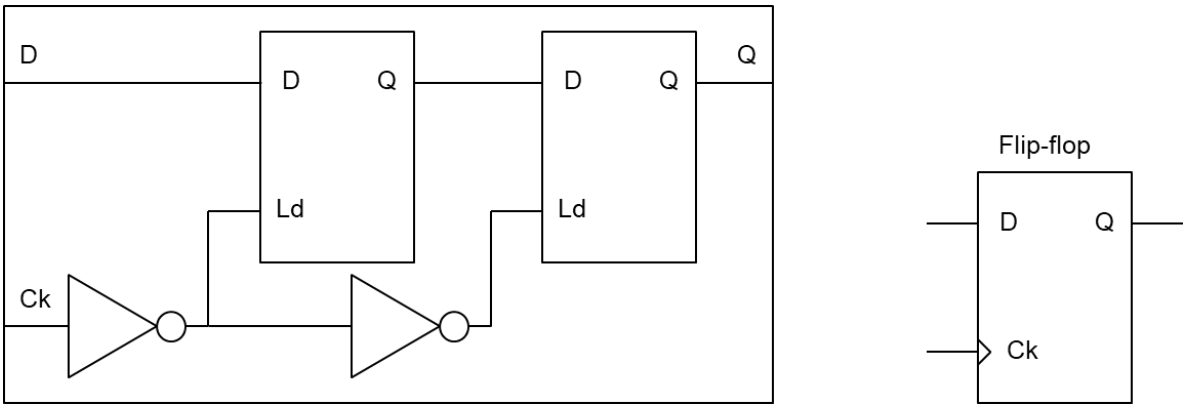


Figure 24: Structure of the flip-flop

Vernier TDC Flip-Flop and Resolution

Figure 24 shows the structure of the flip-flop used in the Vernier TDC. Each flip-flop consists of **two latch circuits**.

The **bin width** of the Vernier TDC can typically be **16 × smaller than the standard TDC bin width T_0**. This enables significantly higher conversion speeds.

Example:

- Both the Vernier and standard TDCs have **16 buffers** in their chains.
- Each TDC produces a **4-bit output**, and the combination yields an **8-bit result**.
- The **range width** of the standard TDC is T_0 = 25ps.
- The **bin width** of the Vernier TDC is
- 25ps/16 = 1.5625 ps
- The total **A/D conversion time** is approximately:

  16 × 25 ps (standard TDCs) + 16 × 25 ps (Vernier TDCs) = 800ps. (1.24 G samples /s).

This example demonstrates how combining **standard and Vernier TDCs** allows for **high-resolution and ultra-fast ADC operation**.

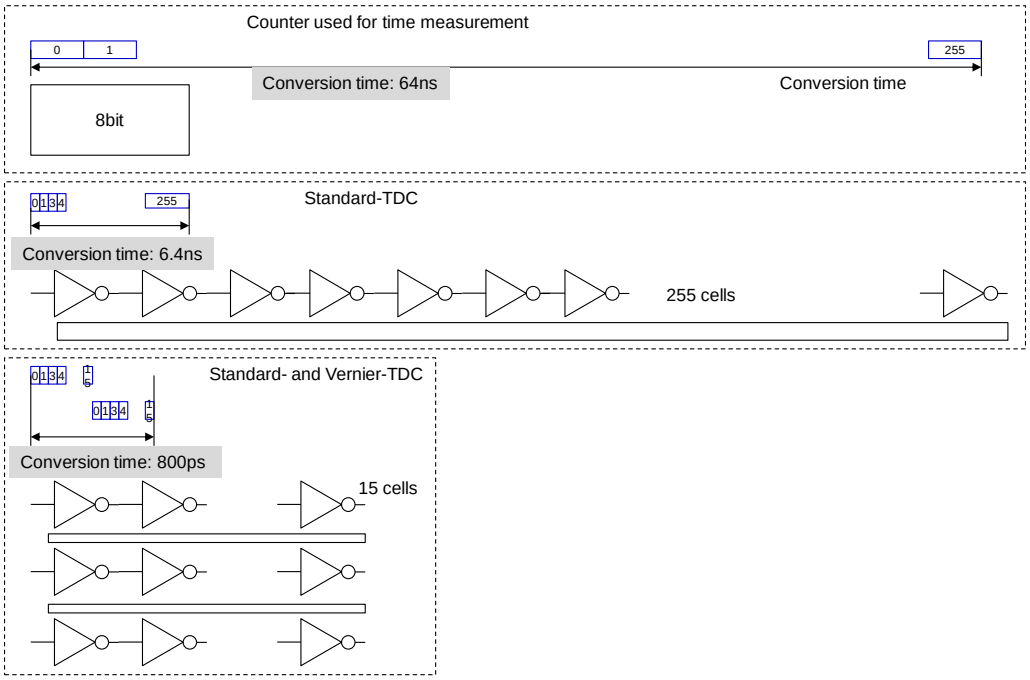


Figure 2: A/D conversion time for different circuits

The **sampling rate** of ADCs can be further increased by connecting multiple ADCs in parallel and **shifting their track signals in time** (Figure 26).

Time-based ADCs are particularly suitable for this approach because of their **compact layout** and **low power consumption**.

This technique is known as **time interleaving**. By employing time interleaving with time-based ADCs, **sampling rates of up to 5 Gsamples/s** at **8-bit resolution** are feasible.

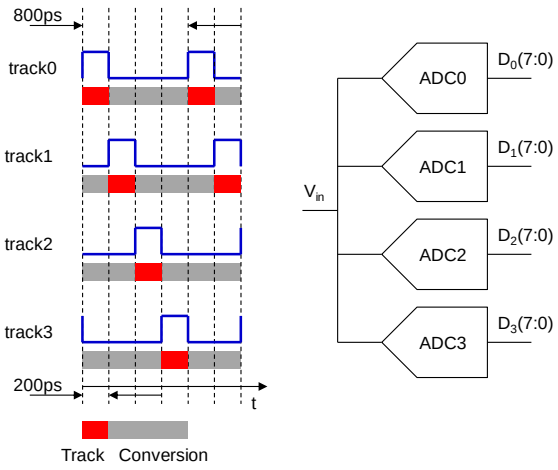


Figure 26: A high sampling rate can be achieved by time interleaving

Pipeline ADC (Informational Only – Not Exam Material)

The basic idea of a **pipeline ADC** is based on a staged conversion algorithm:

1. The input signal v_{in} lies in the range $\pm V_{max}$.
2. The input is first compared to 0 using a comparator:
   - If $v_{in} < 0$, the digital output a is set to 0, and a reference voltage $V_{ref}=V_{max}/2$ is **added** to the input.
   - If $v_{in} > 0$, the digital output a is set to 1, and a reference voltage

     $V_{ref} = V_{max}/2$ is **subtracted** from the input.
3. The resulting signal is **multiplied by 2**.

Repeating this process in successive stages generates the **characteristic curve** shown in Figure 27. Each stage resolves one bit, and the pipeline structure allows high-speed conversion with moderate comparator precision per stage.

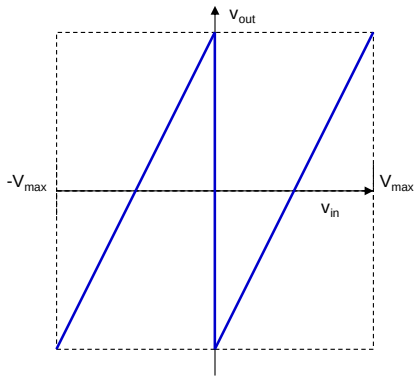


Figure 27: Characteristic curve after a conversion step of an algorithmic ADC

The characteristic curve of one stage has a **slope of 2**. By "folding" the characteristic curve, the output v_{out} is kept in the range $\pm V_{max}$.

If we repeat the algorithm **cyclically n times** (iterations 0, 1, …,n−1), the output of the last stage is:

$$v_{out} = 2\left(\dots 2\left(2\left(v_{in} - \left(a_{n-1} - \frac{1}{2}\right)V_{max}\right) - \left(a_{n-2} - \frac{1}{2}\right)V_{max}\right)\dots - \left(a_0 - \frac{1}{2}\right)V_{max}\right)$$

This can be rewritten as:

$$2^n v_{in} = v_{out} + \sum_{i=0}^{n-1} 2^{i+1}\left(a_i - \frac{1}{2}\right)V_{max}$$

$$2^n v_{in} = v_{out} + \sum_{i=0}^{n-1} 2^{i+1} a_i V_{max} - (2^n - 1)V_{max}$$

Approximating $2n-1 \approx 2n$, we can write:

$$2^{n-1}\left(\frac{v_{in}+V_{max}}{V_{max}}\right) \sim \frac{1}{2}\frac{v_{out}}{V_{max}} + \sum_{i=0}^{n-1} 2^i a_i \equiv \Delta + D_{out} \qquad (14)$$

The number:

$$D_{out} = \sum_{i=0}^{n-1} 2^i a_i \in (0, 2^n - 1)$$

is the digital representation of

$$2^{n-1}\left(\frac{v_{in}+V_{max}}{V_{max}}\right)$$

and thus of v_{in}.

The factor

$$\Delta = \frac{1}{2}\frac{v_{out}}{V_{max}}$$

is the conversion error.

Since v_{out} is in the range $\mp V_{max}$, the conversion error is in the range ∓ 0.5, corresponding to **one LSB**.

Implementation of Pipeline/Cyclic ADCs with Switched-Capacitor Circuits

The pipeline ADC algorithm can be implemented relatively easily using **switched-capacitor circuits**. Two main implementation approaches are illustrated in Figure 28:

1. **Cyclic ADC (Figure 28, top)**
   - The analog output of the cell is **fed back to its own input**, and the algorithm is repeated **n times**.
   - The total conversion time is therefore approximately **n times the processing time of a single cell**.
   - This approach is called a **cyclic ADC**.
2. **Pipeline ADC (Figure 28, bottom)**
   - **n identical cells** are connected in series to form a pipeline.
   - The **first cell** receives the input v_{in}, while each subsequent cell receives the **output of the previous cell** as its input.
   - During the **first clock cycle**, the first cell processes the first sample V0 and generates the **MSB bit $a_{n-1,0}$**.
   - In the **second clock cycle**, the second cell generates the next bit $a_{n-2,0}$, while the first cell processes the **second sample V_1**.
   - This process continues, and in the **nth clock cycle**, the nth cell generates the **LSB bit $a_{0,0}$**, while the first cell processes the n_{th} sample.

Figure 28 (right) shows the **timing diagram for the case n = 4**, illustrating how the pipeline cells process consecutive samples over successive clock cycles.

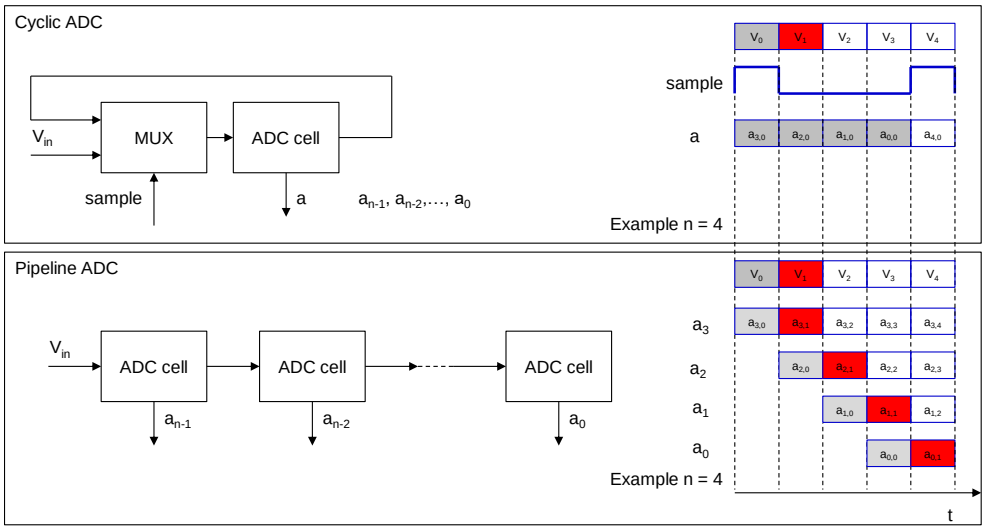


Figure 28: Principle of a cyclic and a pipeline ADC.

Pipeline ADC – Detailed Operation

Figure 29 shows a more detailed **block diagram of a pipeline ADC**. The ADC cells are controlled by a **clock signal Ck**.

Each ADC cell operates in one of two states:

- **Reset state** (reset=1)
- **Amplification state** (reset=0)

At the end of the amplification state, the **digital output a_i** of the cell is ready.
Importantly, **two subsequent cells operate in opposite phases**: when one cell is in the reset state, the next cell is in the amplification state.

Using **shift registers**, the results of all bits for a given sample I ($a_{n-1,i}$, …, $a_{0,i}$) are collected and presented at the **digital output (D_{n-1} – D_0) in parallel**. This technique is called **phase balancing**.

- There is a **pipeline delay of n clock cycles**.
- For example, if sample V_0 enters the first ADC cell at clock 1, its complete conversion result appears at the output at clock n.
- At clock n+1, the next sample V_1 is already available at the output.

Thus, although the **conversion time** for a single sample is n clock cycles, the **sampling period** is only one clock cycle. This is why **pipeline ADCs achieve high throughput**.

Moreover, **adding more pipeline stages** increases resolution **without reducing the sampling rate**. The n-clock delay is typically **not a problem** in most applications.

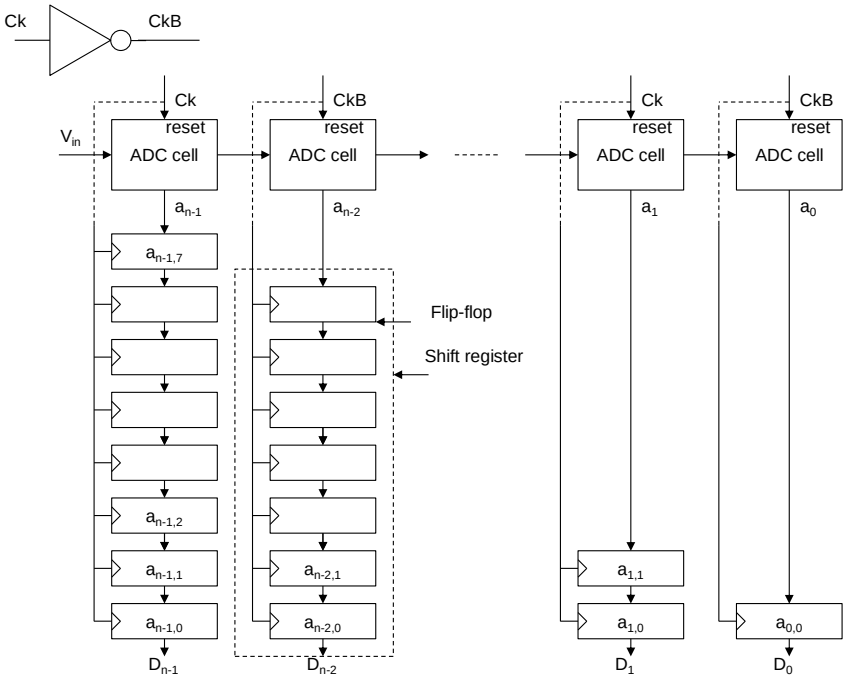


Figure 3: Pipeline ADC with shift registers for phase correction.

Pipeline ADC with Switched-Capacitor Amplifiers (Optional)

Let us now describe a possible **implementation of a pipeline ADC cell** using switched-capacitor techniques.

Inputs of the ADC cell:

- **Reset** – digital input controlling the cell's state
- **In** – analog input, either from the switched-capacitor amplifier output or from the previous ADC cell
- **Out** – analog output to the next ADC cell
- **TooHi (= a)** – digital output generated by the comparator

Components of the ADC cell:

- **Multiplier ×2** with capability to add or subtract a reference voltage
- **Comparator**

Operation of the ADC cell:

- The **×2 multiplier** is implemented using a **switched-capacitor amplifier**.
- Unlike the amplifier in Lecture 11, there is **no input multiplexer**, since the input voltage is already in the appropriate stage form.
- The **feedback capacitance C_{fb}** is **half of the input capacitance C_{in},** achieving an amplification factor of 2.

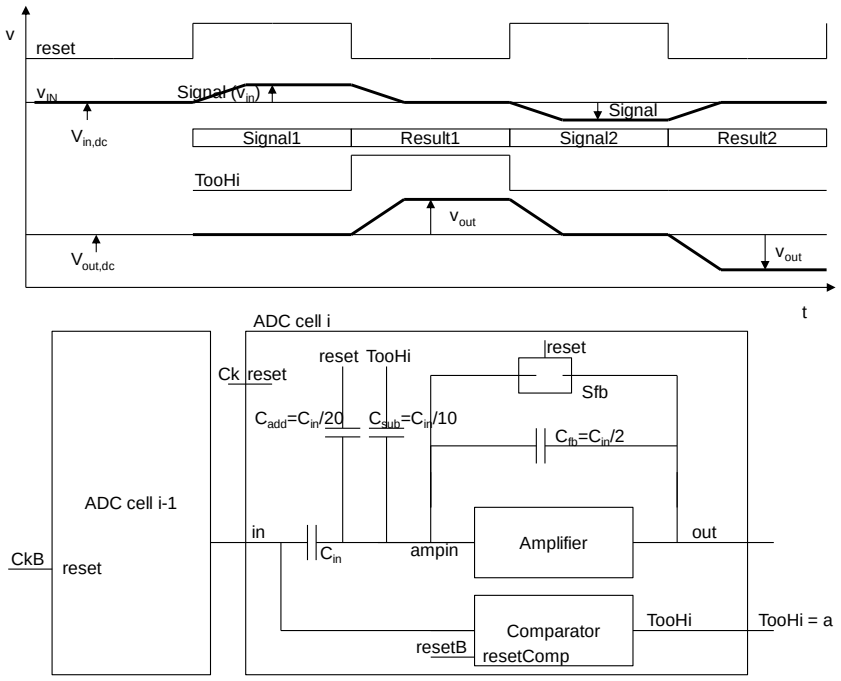


Figure 30: Realization of an ADC cell

Operation of a Pipeline ADC Cell (Figure 30)

Signal 1 period ("Input phase"):

- The input signal v_{in} is applied.
- The **comparator** amplifies the signal and prepares the result (not shown in Figure 30).
- The **multiplier** is in the **reset state**.
- During this phase: $v_{OUT} = V_{out,dc}$

Result 1 period ("Amplification & Conversion phase"):

- If $v_{in} > 0$ (as in Signal 1 in Figure 30), the comparator produces **TooHi = 1**
- The input voltage decreases from $v_{in} + V_{in,dc}$ to $V_{in,dc}$ (here, v_{in} is the small signal).
- C_{sub} subtracts charge
- The signal is amplified by the switched-capacitor amplifier.

Charge analysis:

- **Reset** and **TooHi** signals are connected to capacitors C_{add} and C_{sub}.
- At the beginning of the Result 1 period: reset: 1.8→0, TooHi: 0→1.8

- The total charge flowing through C_{in}, C_{add}, C_{sub} is:

$Q_{in} = C_{in} \times v_{in} + C_{add}\ 1.8V - C_{sub}\ 1.8V = C_{in} \times (v_{in} + 0.05 \times 1.8V - 0.1 \times 1.8V)$

The output voltage is then:

$$v_{out} = \frac{Q_{in}}{C_{fb}} = 2(v_{in} - 0.09) \quad (15)$$

Signal 2 case ($v_{in} < 0$) – TooHi not generated:

$Q_{in} = C_{in} \times v_{in} - C_{add}\ 1.8V = C_{in} \times (v_{in} + 0.05 \times 1.8V)$

and

$$v_{out} = \frac{Q_{in}}{C_{fb}} = 2(v_{in} + 0.09) \quad (16)$$

Equations (17) and (18) correspond to the **pipeline ADC conversion algorithm** and generate the **characteristic curve** shown in Figure 27.

Input range requirement:

$V_{in,dc} \mp 2 \times 90$ mV ($\mp$ 180 mV) .

In this design, $V_{in,dc} = V_{out,dc}$.

Realization of the Comparator

Figure 31 shows the **circuit diagram of the comparator** used in the pipeline ADC.

- The **comparator input** is connected to v_{in} of the ADC cell.
- It is implemented as a **switched-capacitor amplifier**, but with a **much smaller feedback capacitor C_{fb}** and a **higher gain** than the main amplifier in the ADC cell.
- The **reset signal of the comparator** (resetComp) is connected to the **inverted reset of the amplifier** (resetB, Figure 31).

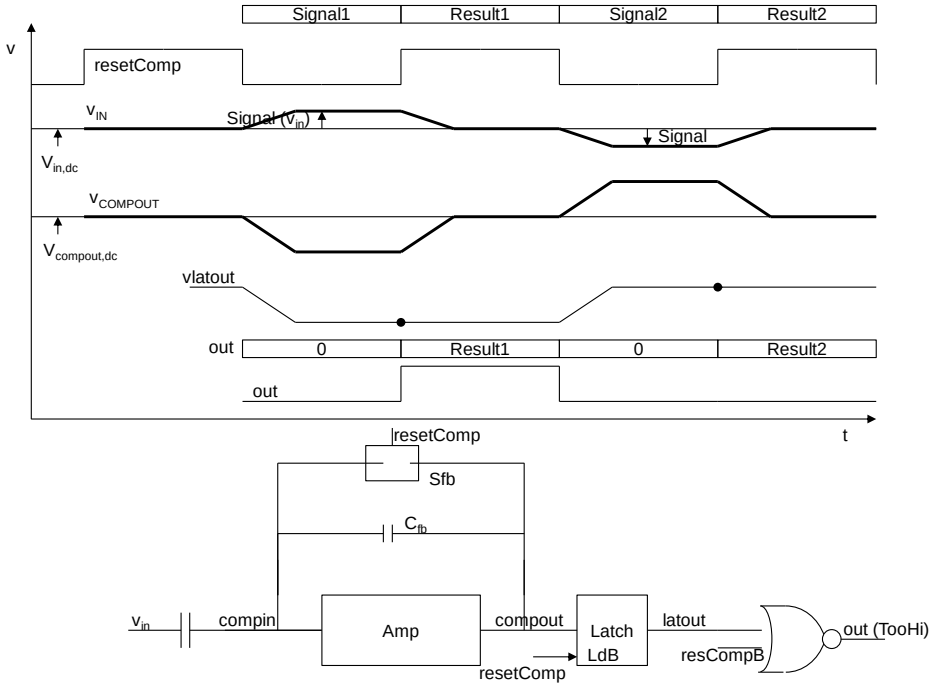


Figure 31: Implementation of the comparator

Latch operation:

- The **latch is transparent** (write state) when resetComp=0 V.
- The latch is in **memory state** when resetComp=1.8V.
- Therefore, the comparator result is available at the output during the **resetComp = 1.8 V** phase.

Comparator output behavior:

- If $v_{in} > 0$ (Signal 1 period), $v_{COMPOUT}$ becomes **low**.
- If $v_{in} < 0$ (Signal 2 period), $v_{COMPOUT}$ becomes **high**.
- Note that $v_{COMPOUT}$ is **negated**.
- The final comparator result is available at the **latch output latout** during the **result period** (resetComp=1.8V).

NOR circuit function:

- Ensures that the digital output is **0 during resetComp = 0 V** (signal period).

- Prevents subsequent digital circuits from receiving an **undefined logical level** at the start of the signal period.
- Performs a **logical negation**, necessary because $v_{COMPOUT}$ is inverted.

Appendix

Basic CMOS Digital Circuits

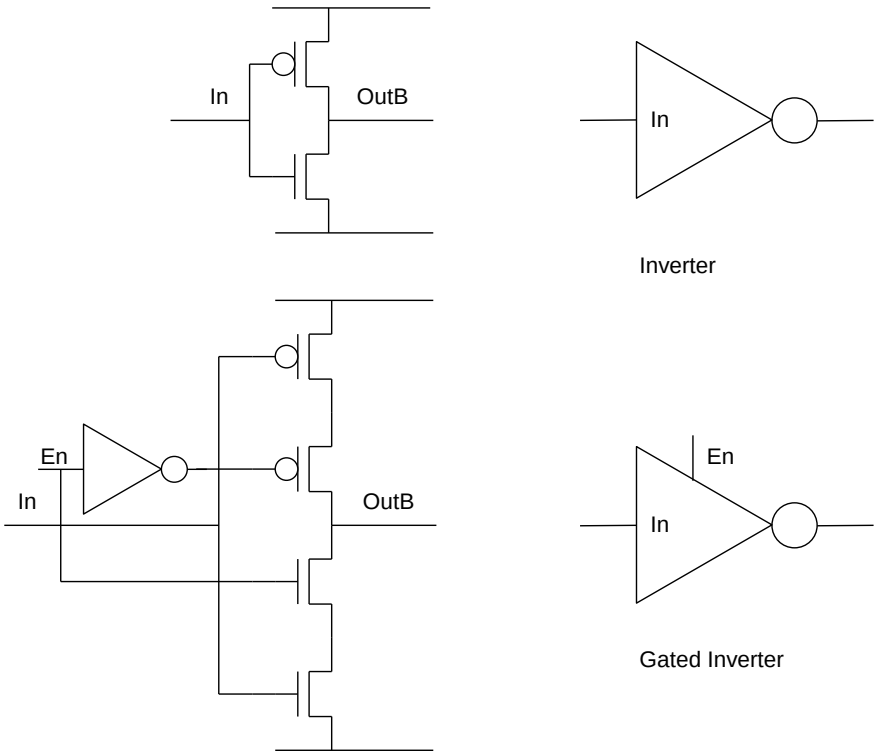


Figure 4: Inverter and Gated-Inverter

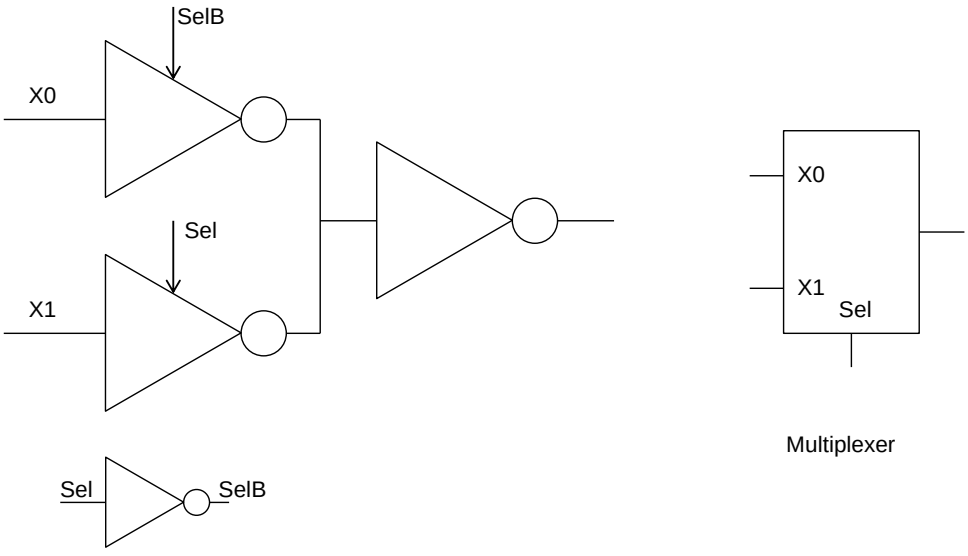


Figure 5: Multiplexer

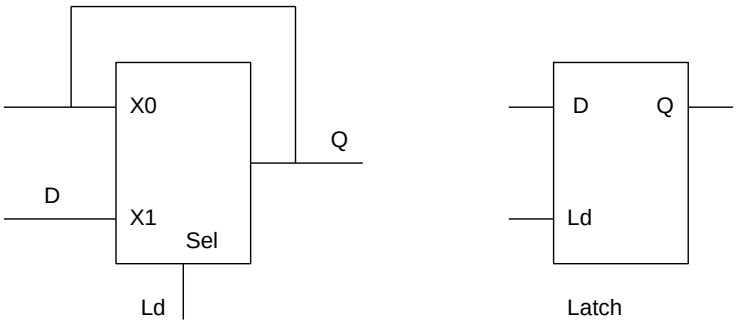


Figure 6: Latch

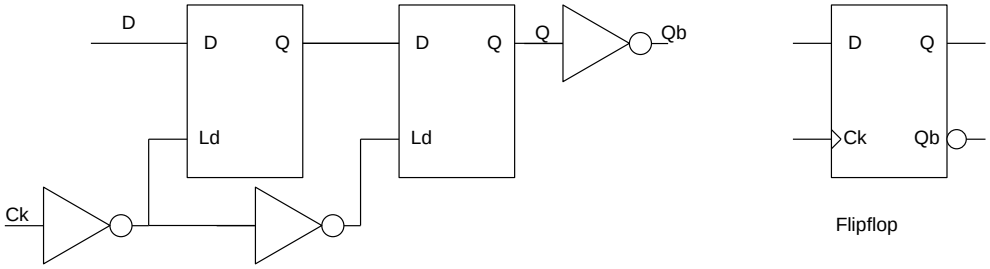


Figure 7: Flip-flop