Introduction to Embedded Systems

EE 186 – Lecture 7

Analog Signals, ADC, DAC

Zerina Kapetanovic

Today we are learning about…

- Overview of Analog Signals in Embedded Systems
  - Sampling Signals
  - Data Conversion

- Analog to Digital Converters (ADC)
- Digital to Analog Converters (DAC)

Reading

- **STM RM0432 Reference Manual**
  - Section 21: Analog to Digital Converter (ADC)
  - Section 22: Digital to Analog Converted (DAC)

- **Embedded Systems with ARM Cortex-M Microcontrollers in Assembly Language and C (Third Edition)**
  - Chapter 20: Analog to Digital Converter (ADC)
  - Chapter 21: Digital to Analog Conversion (DAC)

The physical world is … analog

- **Digital systems need to…**
  - Measure analog quantities (sensor inputs, sound waveforms, etc.)
  - Control analog systems (actuators, etc.)

- **Analog-to-digital (ADC)**
  - Audio recording

- **Digital-to-analog (DAC)**
  - Audio playback

Analog Signals

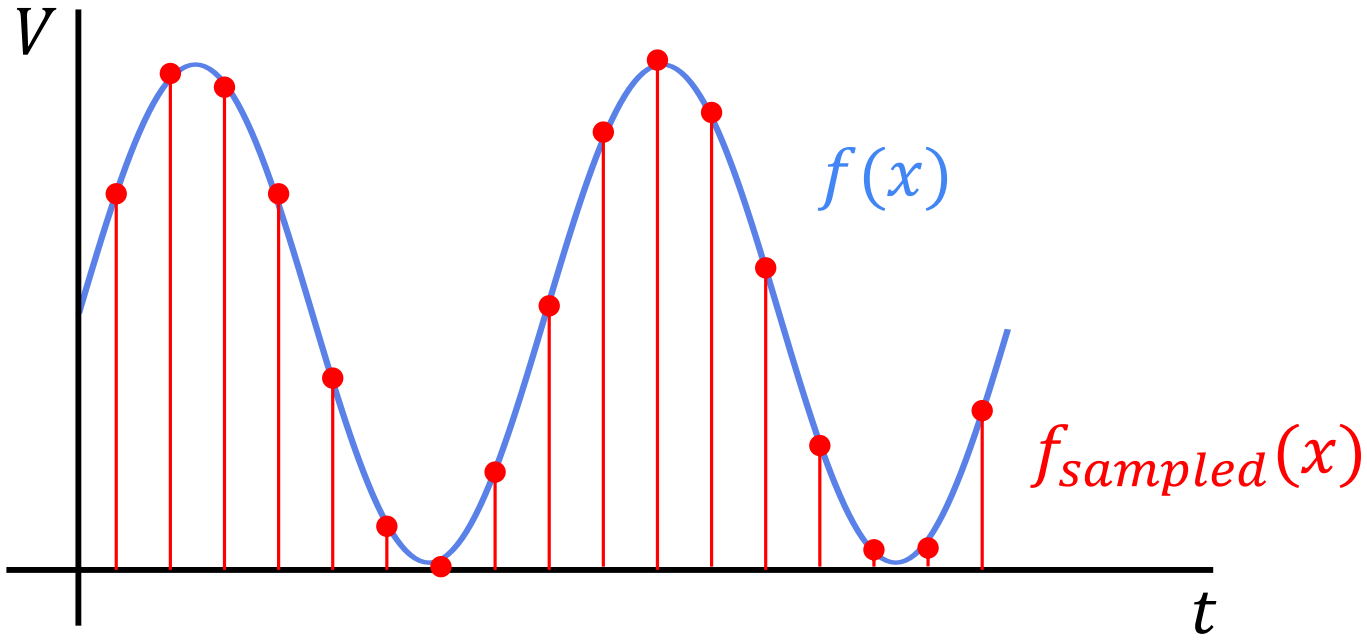
- An **analog signal** is one that is continuous in both amplitude and time
- To represent a signal in the digital domain **we must approximate** it in two ways:
  - Amplitude quantizing
  - Time quantizing

converting continuous or large sets of values into a discrete set of values.

- The **range** of the system is the maximum minus the minimum value
- The **precision** of the system defines the number of values from which the amplitude of the digital signal is selected
- The **resolution** is the smallest change in value that is significant

Analog Signal in the Digital Domain

How do we represent an analog signal?



Sampling

The sampling rate is the rate at which amplitude values are digitized from the original waveform

Can you observe the sampling rate?

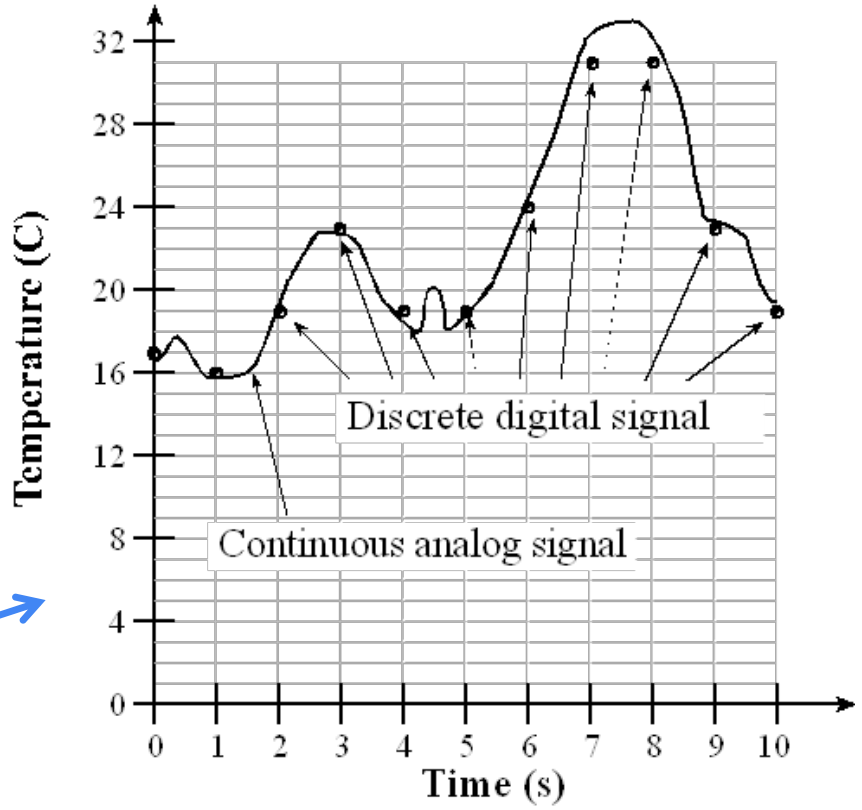


Photo Credit: Jonathan Volvano (UT Austin)

Shannon-Nyquist Theorem

If a continuous-time signal $f(x)$ contains **no frequencies higher than f_{max}**, it can be completely determined by discrete samples taken at a rate:

$$f_{samples} > 2f_{max}$$

Example

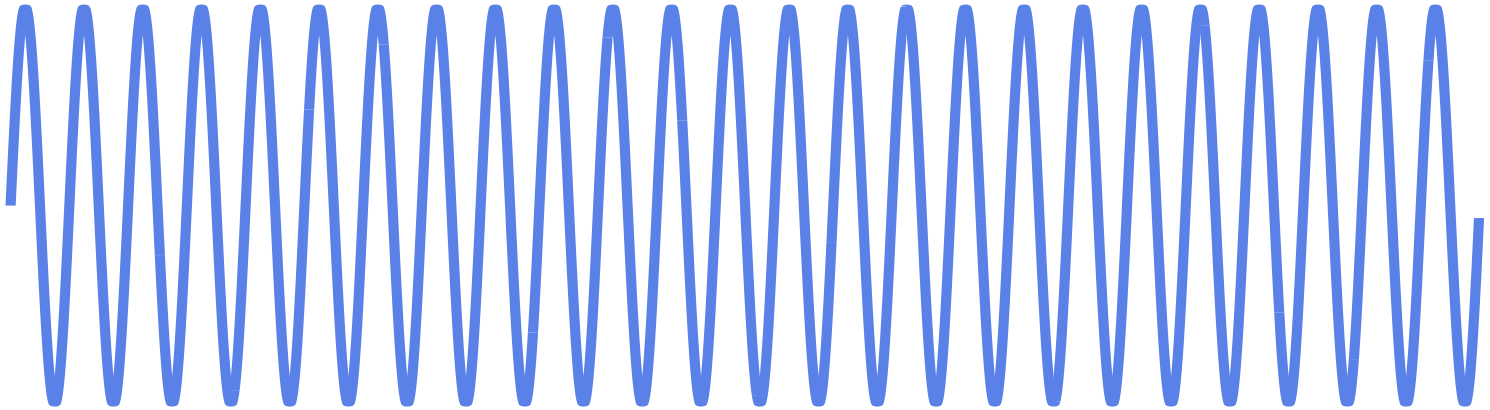
Humans can process audio signals
20Hz – 20kHz

Audio CDs: Sampled at 44.1 kHz



Example: Sampling a signal

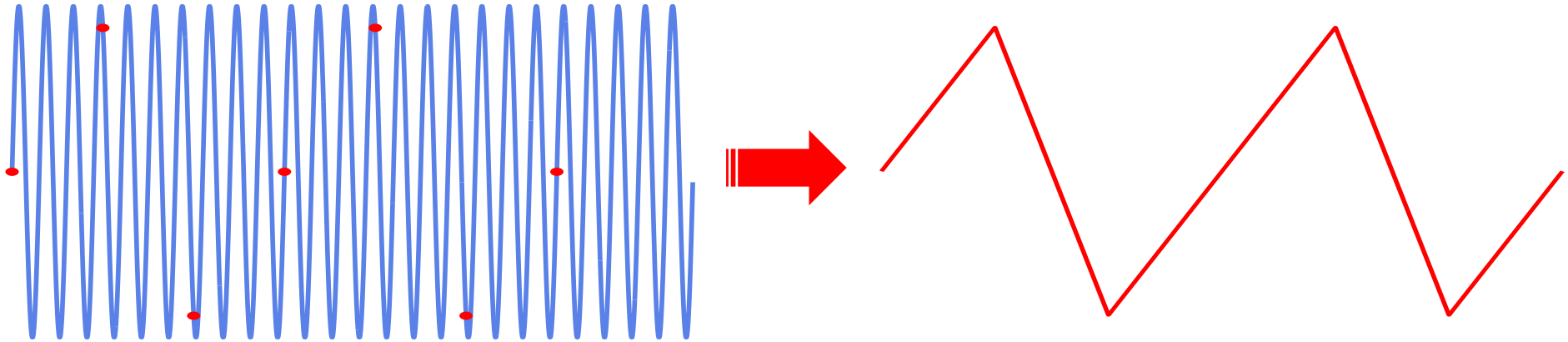
$f(t) = \sin(2\pi f_c t)\,, f_c = 5Hz$



Example: Sampling a signal

$$f(t) = \sin(2\pi f_c t), \quad f_c = 5Hz$$

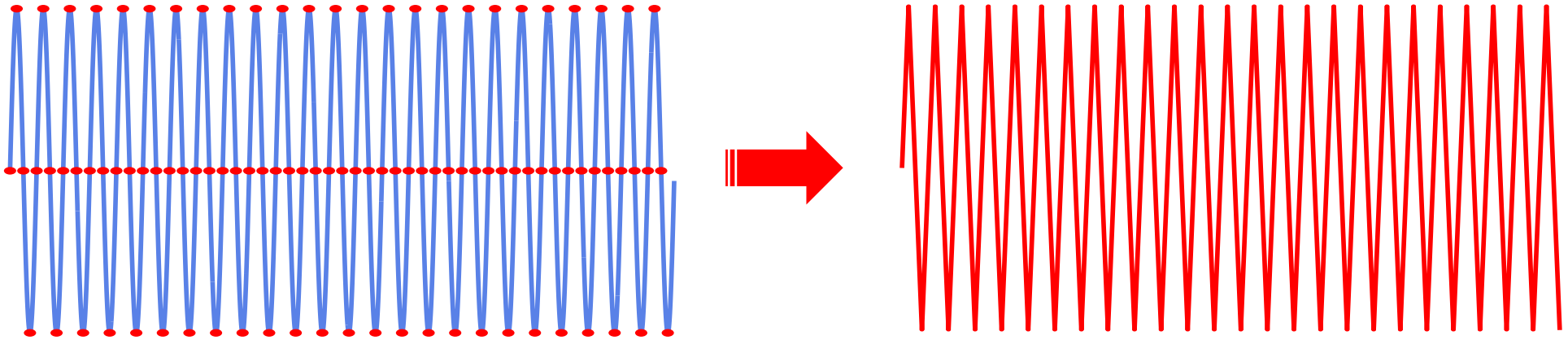
What signal do I expect to see if $f_s = 1.5Hz$?



Example: Sampling a signal

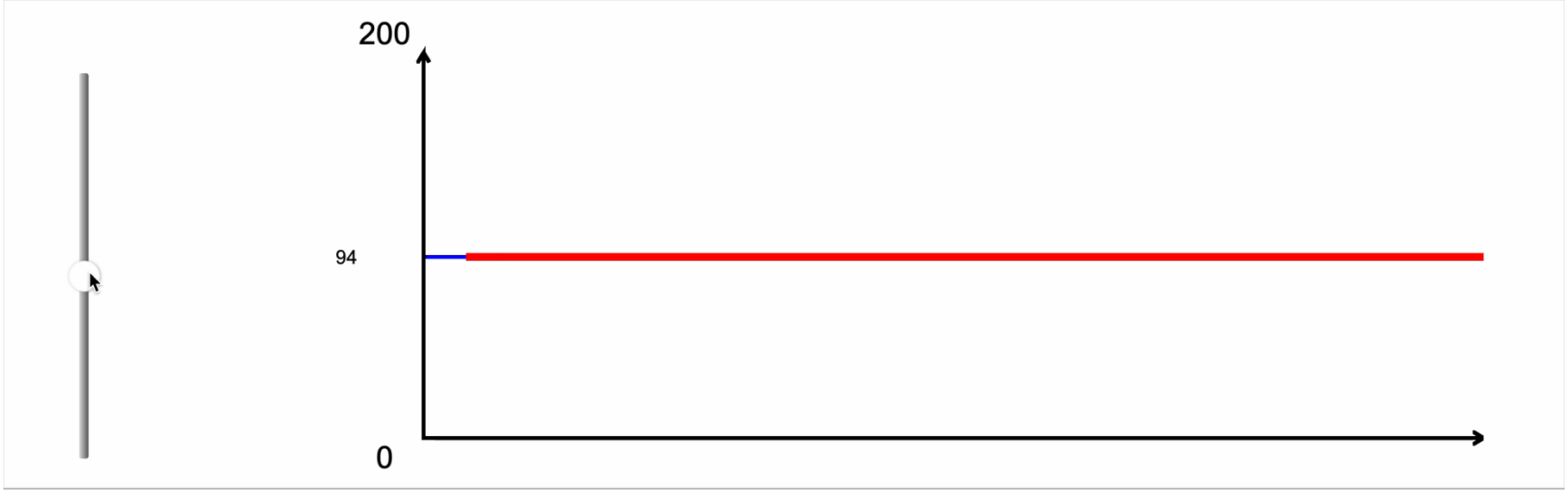
$f(t) = \sin(2\pi f_c t)\,, \quad f_c = 5Hz$

What signal do I expect to see if $f_s = 20Hz$?



Example: Sampling a Signal

Fixed sampling at 1Hz



Credit: Jonathan Valvano

Shannon-Nyquist Theorem

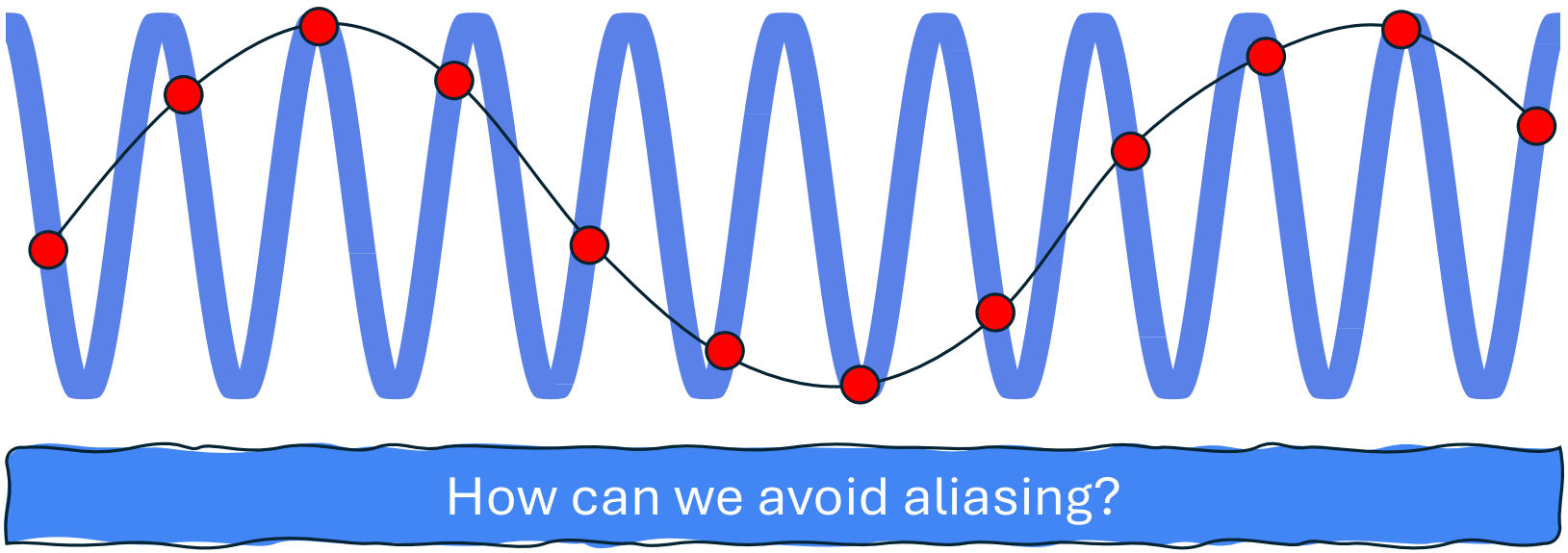
If a continuous-time signal $f(x)$ contains **no frequencies higher than f_{max}**, it can be completely determined by discrete samples taken at a rate:

$$f_{samples} > 2f_{max}$$

In a real system what are some issues you could run into?

Aliasing

Aliasing occurs **when a continuous signal is sampled at a rate insufficient to capture its changes accurately.** This results in the observance of false frequencies, or aliases, that were not present in the original signal[1].



[1] Keysight Technologies. (n.d.). What is aliasing? Keysight Knowledge. https://www.keysight.com/used/us/en/knowledge/glossary/oscilloscopes/what-is-aliasing

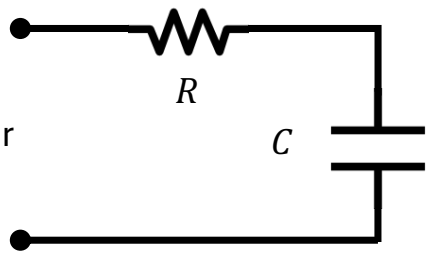
Anti-aliasing Filter

Simulate in Falstad!!

A low-pass filter **passes lower frequencies** and **attenuates higher frequencies**

$$|V_{out}| = |V_{in}| \frac{1}{\sqrt{1 + \omega^2 R^2 C^2}}$$

$$\omega = 2\pi f$$



RC Low Pass Filter

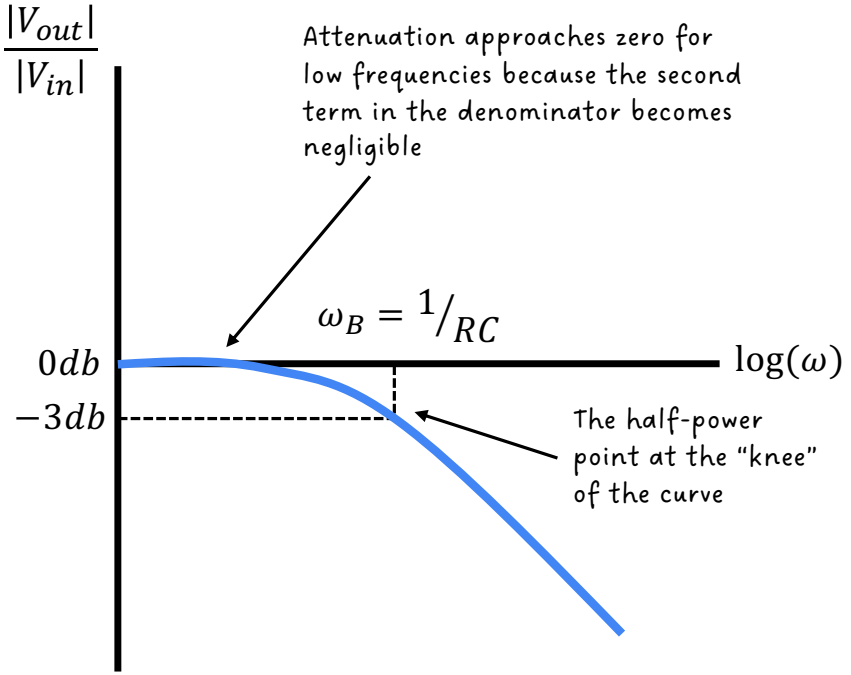


Figure adapted from Falstad

Say you have 0.1μF capacitor, what value should R be so that the half-power point occurs at 30Hz

$$|V_{out}| = |V_{in}| \frac{1}{\sqrt{1 + \omega^2 R^2 C^2}}$$

ω *is in radians,* $\omega = 2\pi f$

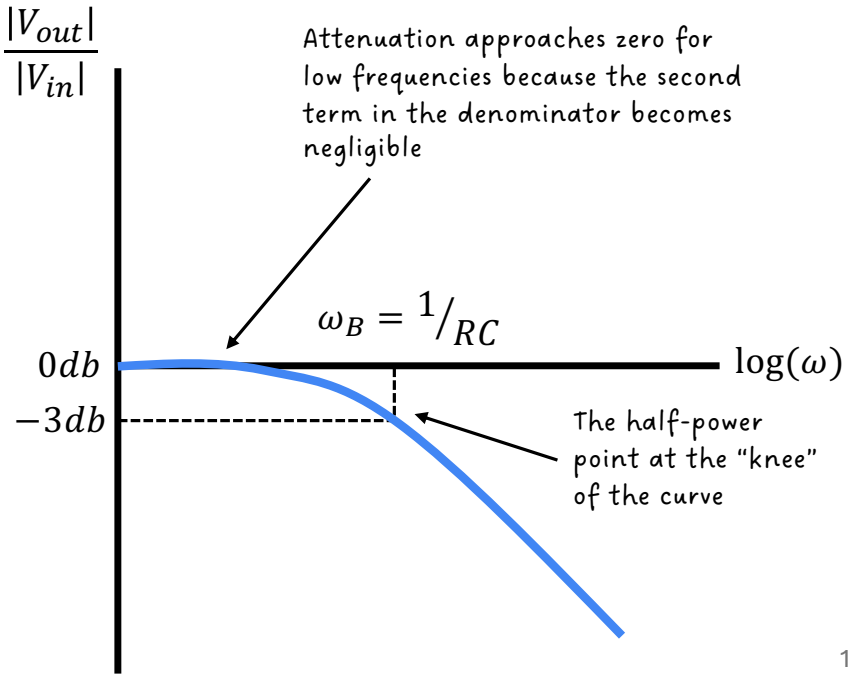


Figure adapted from Falstad

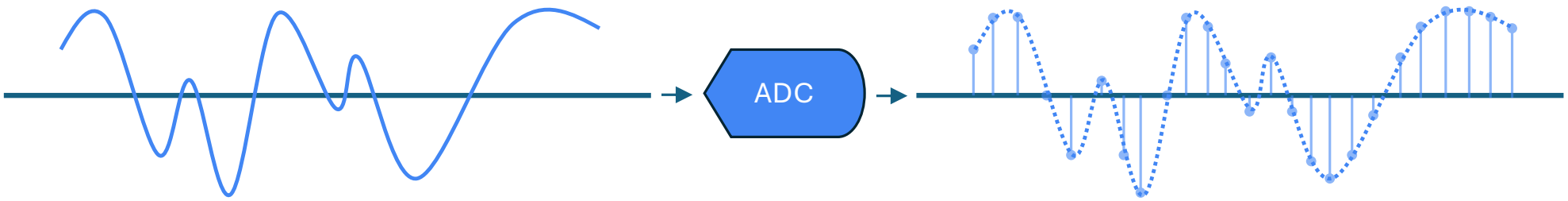
Analog to Digital Converter (ADC)

An analog-to-digital converter (ADC) converts an analog signal into digital form

We use ADCs in embedded systems to collect information about the external world

The input signal is usually analog voltage and the output is a binary number

Analog to Digital Conversion (ADC)



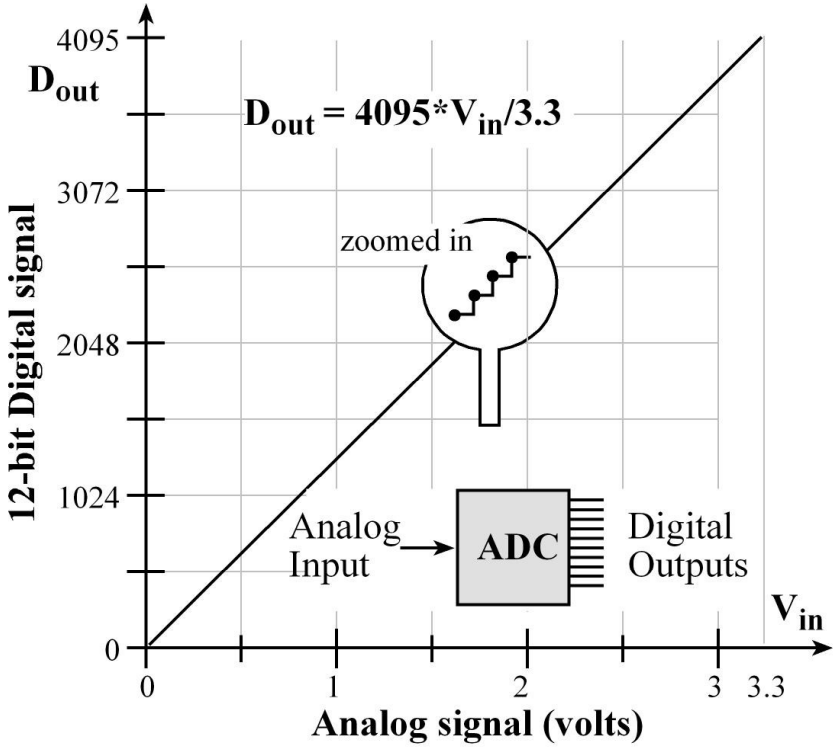
Continuous-time analog signal

Discrete-time digital values

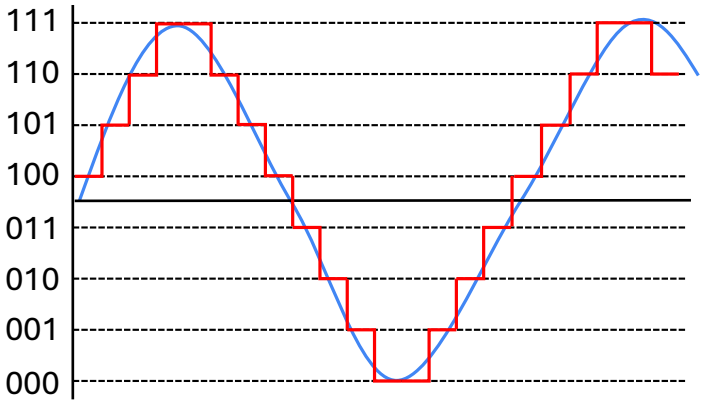
Analog to Digital Converter (ADC)

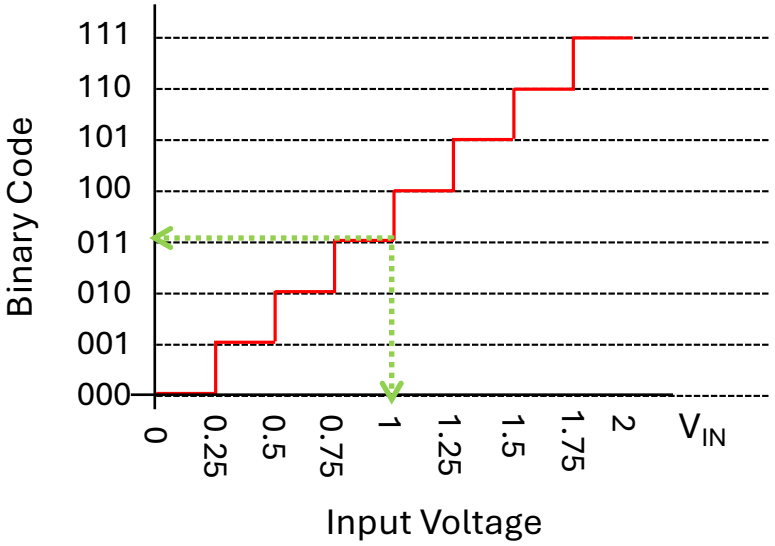
Range is the maximum and minimum ADC input (e.g., 0 to +3.3V)

Resolution is the smallest distinguishable change in the input (e.g., 3.3V/4095 … about 0.81mV)



Discretization





Resolution is determined by the number of bits (in binary) used to represent an analog input

Announcements

- No class next Tuesday (midterm study time)
- Midterm next Thursday
- HW 3 due today
- Final proposal due next Tuesday
  - Assignment will be posted on Gradescope
- Lab discussion sessions are tomorrow
  - You should all be signed up

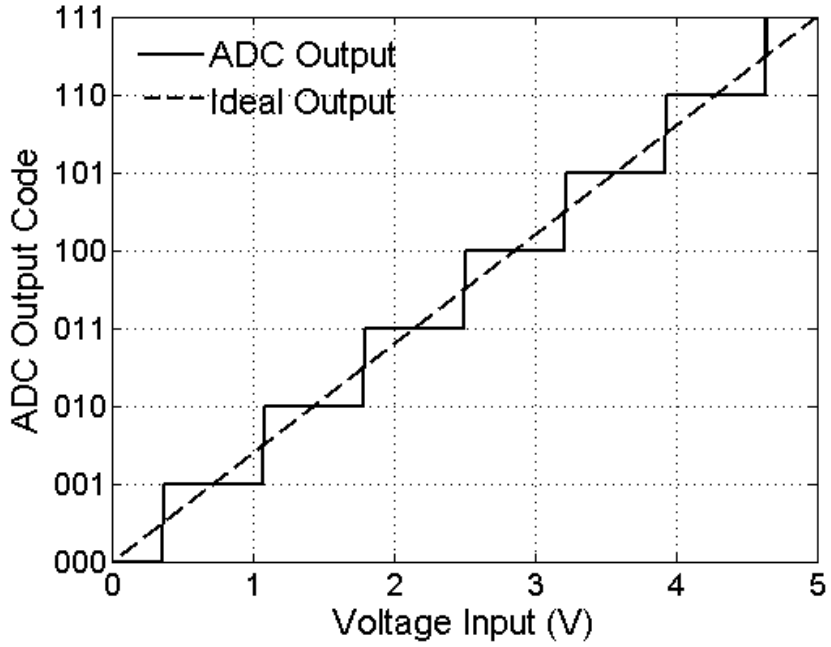
Quantization Method

If the ADC output has *n* bits, the ADC result of an input voltage *V* can be calculated as follows:

$$Digital\ Result = round\left((2^n - 1) \times \frac{V - V_{RL}}{V_{RH} - V_{RL}}\right)$$

High reference voltage

Low reference voltage



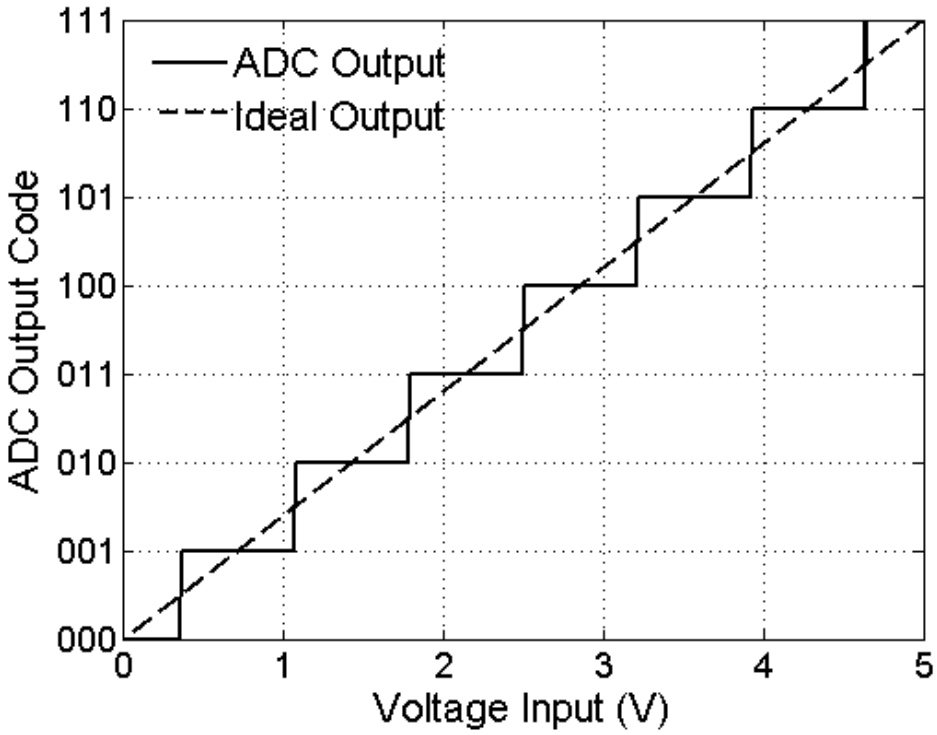
Quantization Error

Resolution is determined by the number of bits (in binary) used to represent an analog input

Larger step size → less information

Quantization error is how far off the discrete value is from actual

Larger step size → larger error



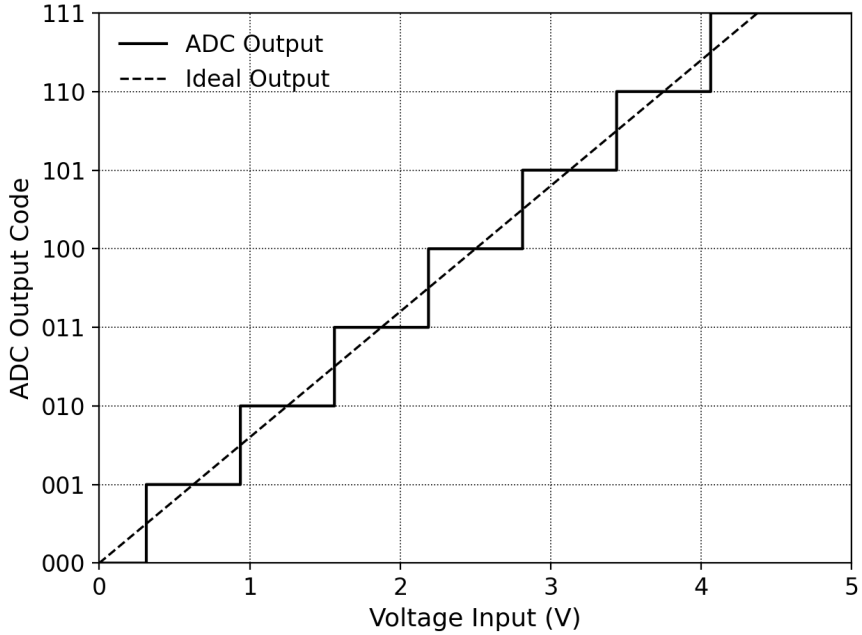
Quantization Methods

Single Ended Mode (V_{RL} = 0)

$$Digital\ Result\ = round\left((2^n - 1)\times \frac{V}{V_{RH}}\right)$$

$V_{REF} = 5V \qquad n = 3$

What is the ADC output if the voltage input is 2.2V?



Quantization Error

Step Size $(\Delta) = \frac{V_{REF}}{2^n - 1}$

$(\Delta) = \frac{5}{7} \approx 0.714\ V$

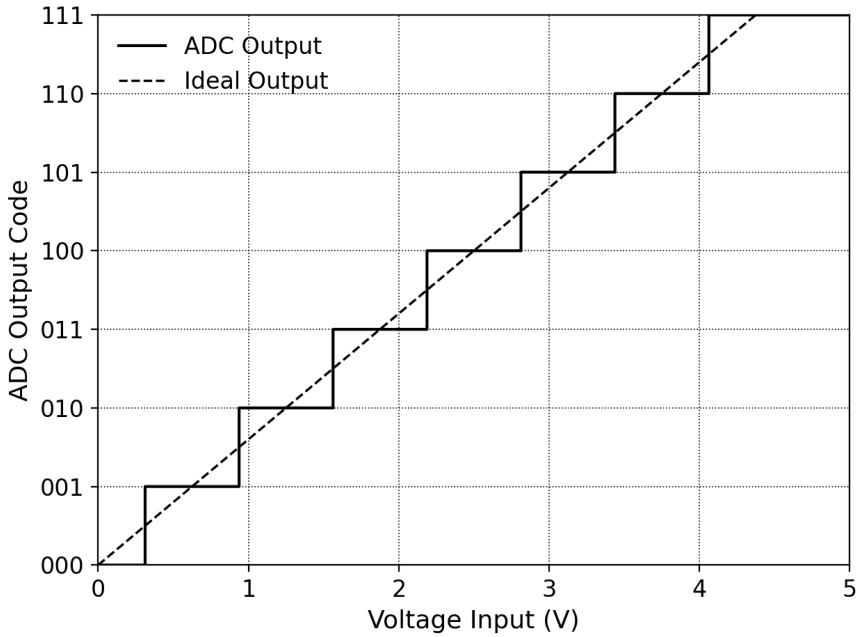
$Actual = 2.2V \qquad ADC\ Code = 3\ (011)$

$V_{Quantized} = ADC\ Code \times \Delta = 3 \times 0.714 \approx 2.14V$

$Quantization\ Error = 2.2\ - 2.14 \approx 0.06V$

$\frac{1}{2}\Delta$

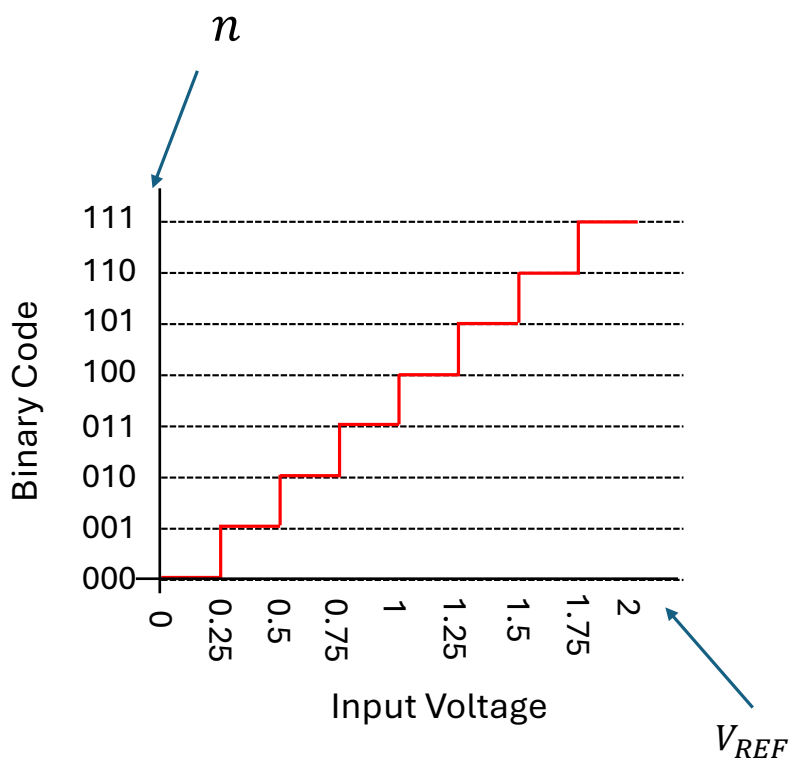
Why?



Transfer Function

Convert the ADC binary output code to a voltage value

$$V_{code} = ADC\ Output \times \frac{V_{REF}}{2^n - 1}$$



Data Conversion Example

Analog to Digital – How do we get there?

What we would like:



How do we actually do it:



Example: Analog Sensor

From the datasheet!

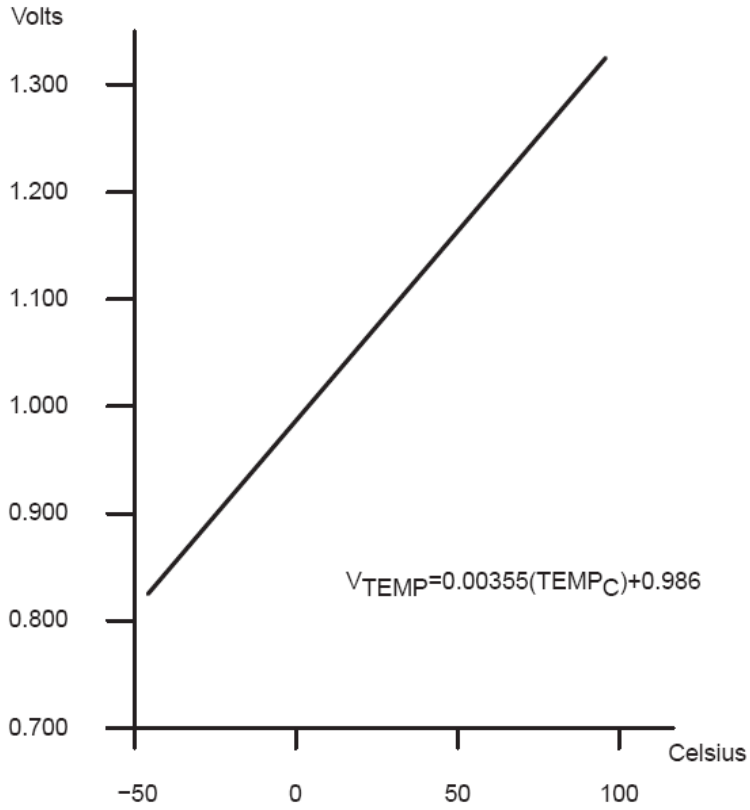
Analog Temperature Sensor



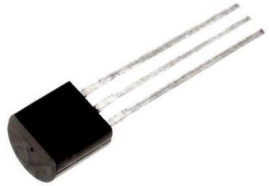
Analog Soil Moisture Sensor



Output curve and transfer function



Example



Three Pin Analog Temperature Sensor

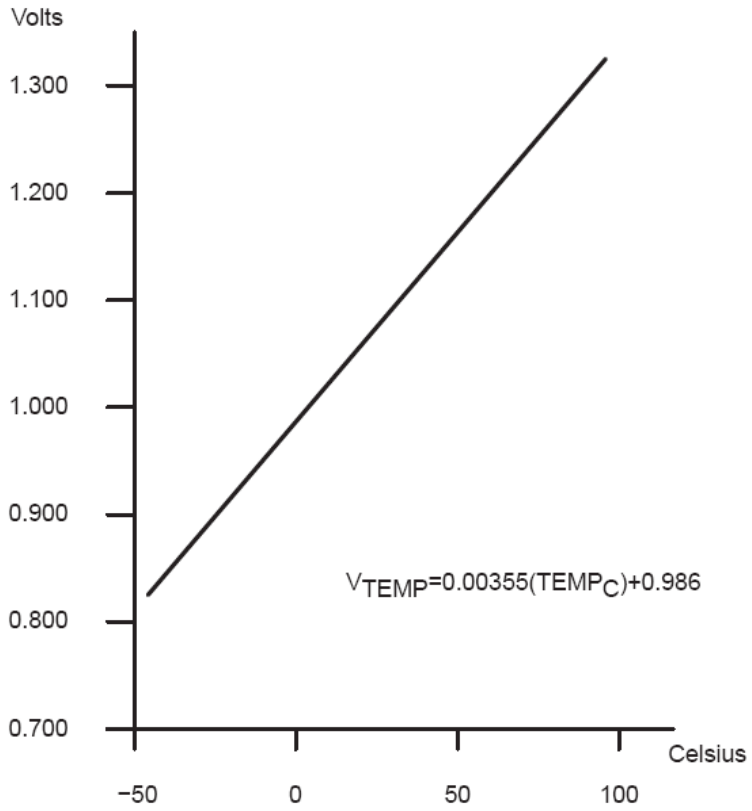
- Assume that the maximum valid output voltage for this sensor is 1.4 V
- We are using an MCU with a 10-bit ADC, referenced from 0 to 3.3V

What is the *minimum* step size of this ADC?

What is the transfer function you will need to write in your code? (ADC Value --> Temperature)

What is the ADC value at 50C

Output curve and transfer function

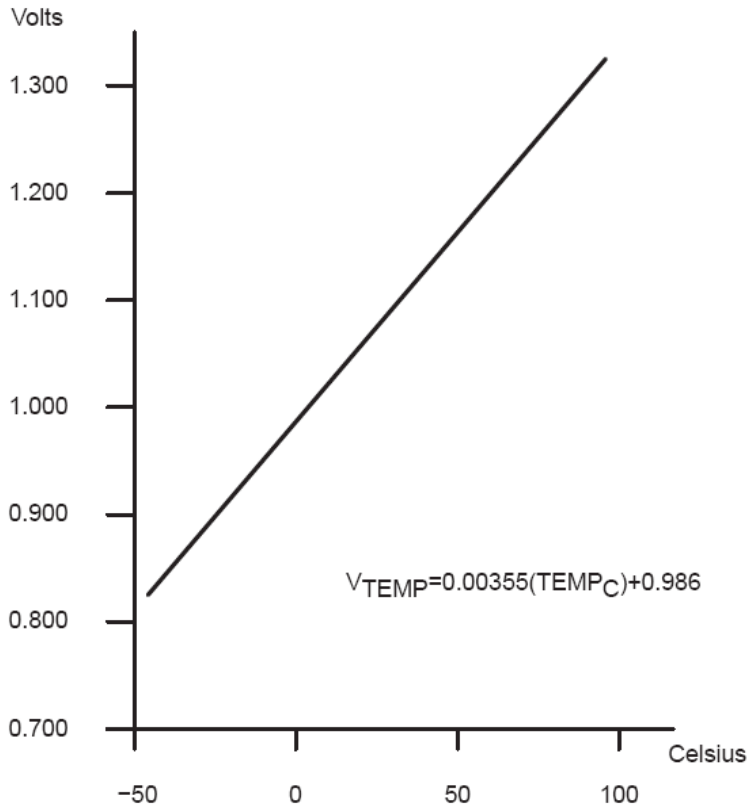


Example

What is the *minimum* step size of this ADC?

$$\Delta V = \frac{V_{REF}}{2^n - 1} = \frac{3.3}{2^{10} - 1} = 3.23mV$$

Output curve and transfer function



Example

What is the transfer function you will need to write in your code?

$$V_{Temp} = 0.00355 \cdot T_c + 0.986$$

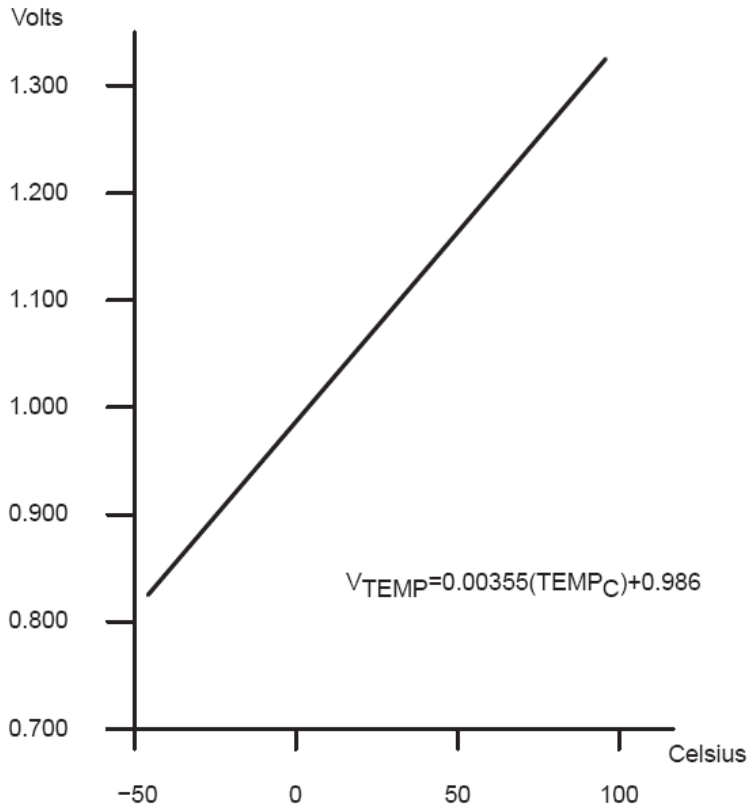
$$ADC = (2^n - 1)\frac{V_{Temp}}{V_{REF}} = (2^n - 1)\frac{0.00355 \cdot T_c + 0.986}{V_{REF}}$$

$$T_c = \frac{281.69 \cdot ADC \cdot V_{REF}}{2^n - 1} - 277.75$$

$$T_c = \frac{281.69 \cdot ADC \cdot 3.3}{2^n - 1} - 277.75$$

$$T_c = 0.9087 \cdot ADC - 277.75$$

Output curve and transfer function



Example

What is the ADC value at 50C?

$$ADC = (2^n - 1)\frac{0.00355 \cdot T_c + 0.986}{V_{REF}}$$

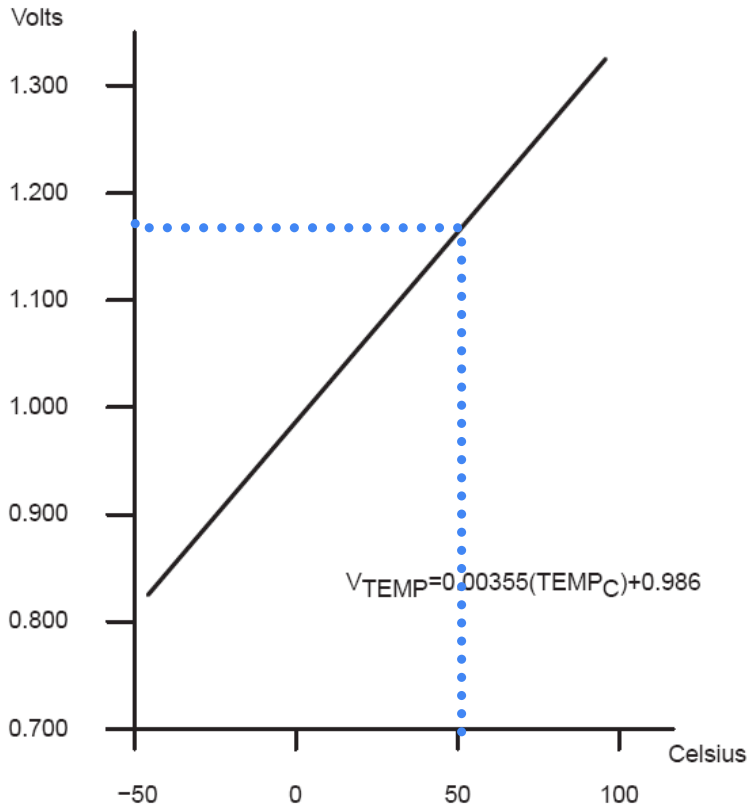
$$ADC = (2^{10} - 1)\frac{0.00355 \cdot 50 + 0.986}{3.3}$$

$$\boxed{ADC = 361}$$

Does the value make sense??

$$\frac{ADC}{2^n - 1} \cdot V_{REF} = \frac{361}{2^{10} - 1} \cdot 3.3 = 1.16V$$

Output curve and transfer function



Recap: Analog to Digital Converter (ADC)

- An ADC produces a finite-precision digital number to represent approximately the size of an analog voltage relative to a reference voltage
- Key performance parameters:
  - Sampling rate
  - Resolution
  - Power Dissipation

ADC Architecture

There are three popular ADC architectures

Sigma-Delta

- Low-speed applications
  - Voice/audio
- Low-sampling rates
- Typically < 100ksps

Successive-Approximation (SAR)

- Low-power data acquisition
- Moderate sampling rates
- Typically < 5Msps

Pipelined

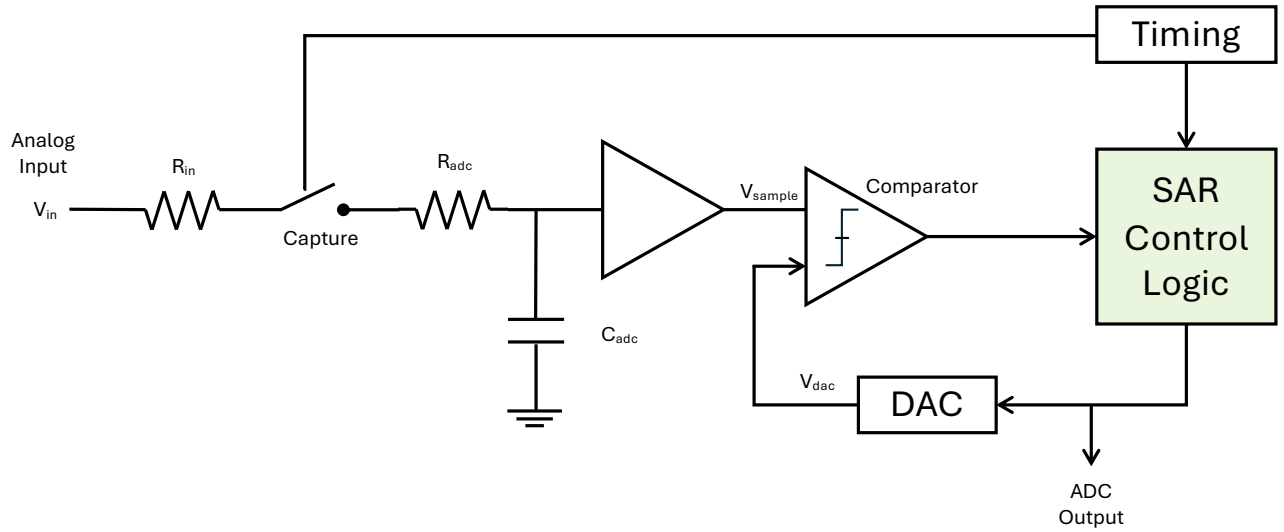
- High-speed applications
  - Digital oscilloscopes
  - Radar Communication
- Fast sampling
- Typically > 5Msps

The ADC on STM32 Microcontrollers is based on the *successive-approximation* (SAR) architecture

SAR Architecture

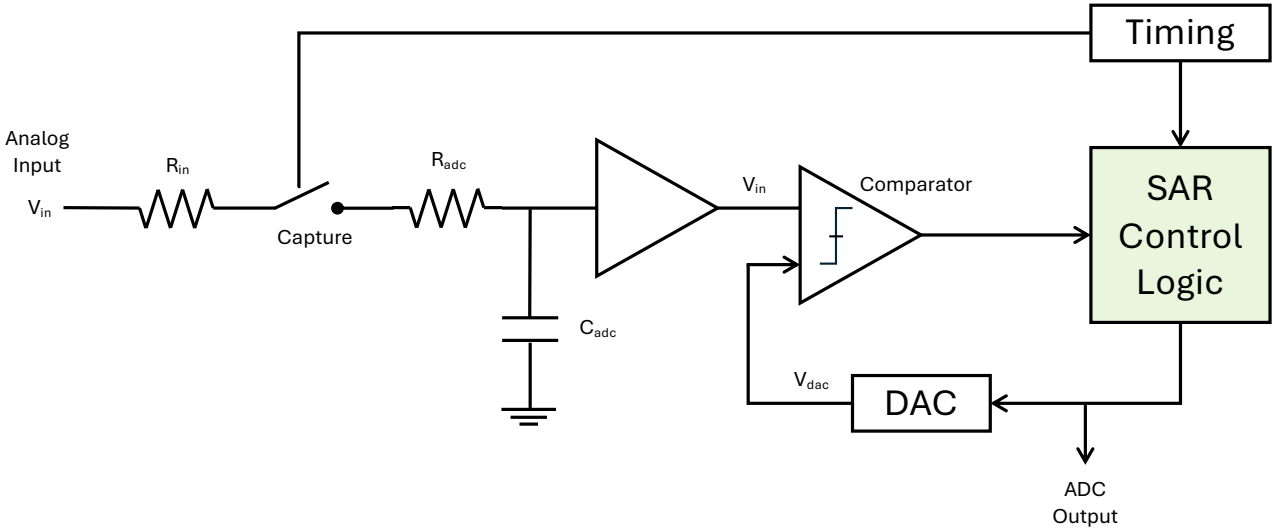
Two main components:

- SAR digital quantization
- Sample and hold amplifier



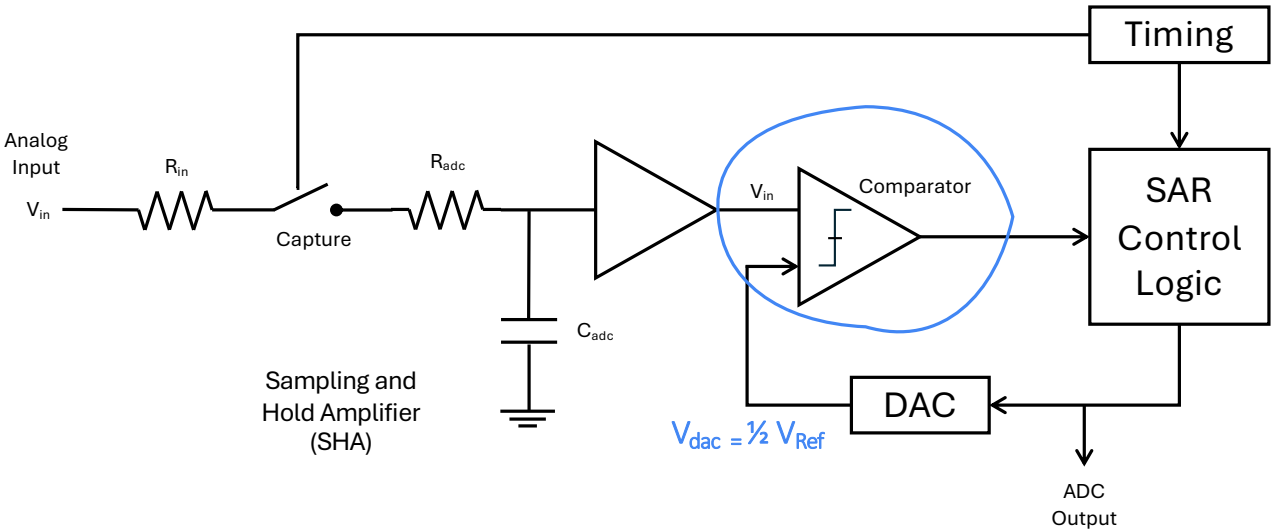
SAR Digital Quantization

SAR Digital Quantization



- Use the binary search algorithm to find the digital number (ADC output) that represents the analog input most closely
- Dynamically changes the ADC output so that the DAC output gradually approaches the DAC input voltage

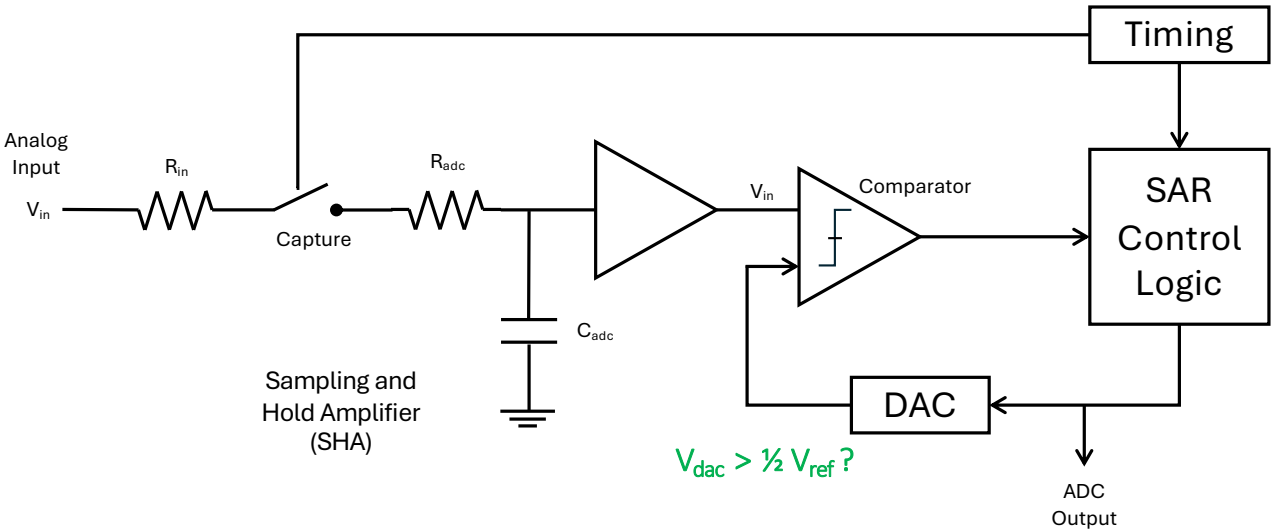
SAR Digital Quantization



1

Set the internal DAC output V_{dac} to ½ V_{Ref} and compare V_{dac} with the ADC input V_{in}

SAR Digital Quantization



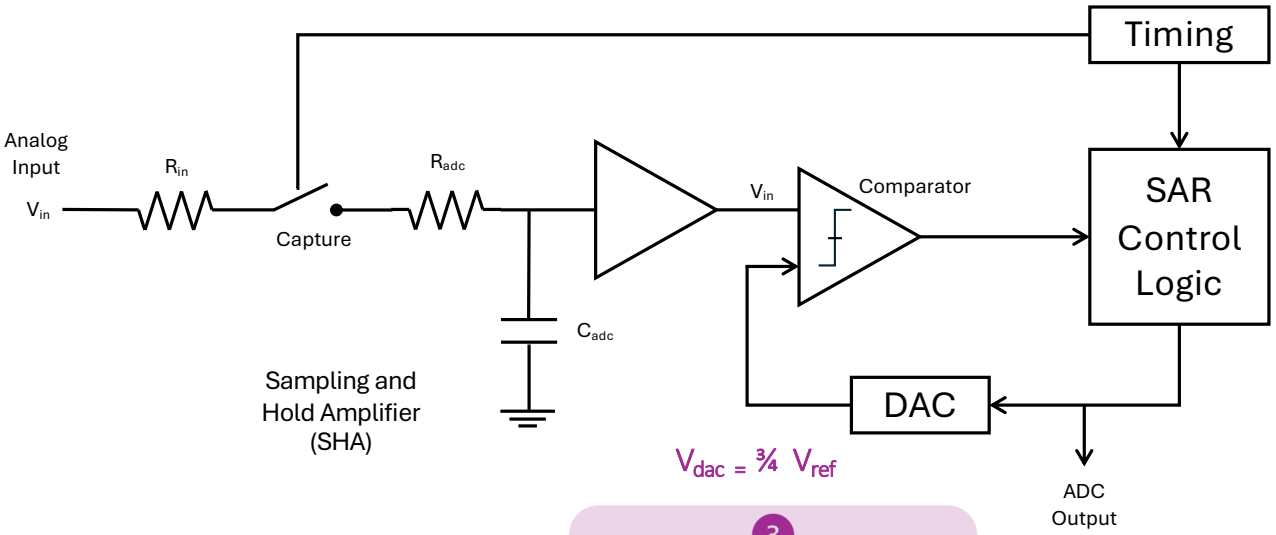
1

Set the internal DAC output V_{dac} to ½ V_{Ref} and compare V_{dac} with the ADC input V_{in}

2

If V_{in} is larger than ½ V_{ref} , set the most significant bit (MSB) of the ADC result. Otherwise, clear the MSB of the ADC

SAR Digital Quantization



1

Set the internal DAC output V_{dac} to ½ V_{Ref} and compare V_{dac} with the ADC input V_{in}

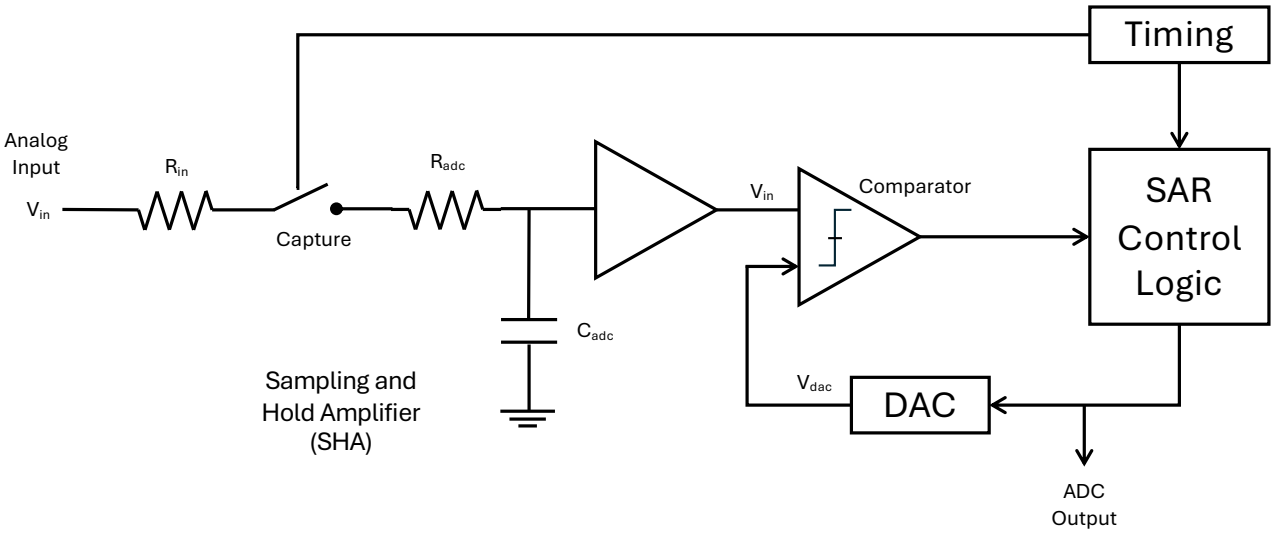
2

If V_{in} is larger than ½ V_{ref} , set the most significant bit (MSB) of the ADC result. Otherwise, clear the MSB of the ADC

3

If V_{dac} is set to either ¾ V_{ref} or ¼ V_{ref} , depending on comparison result

SAR Digital Quantization

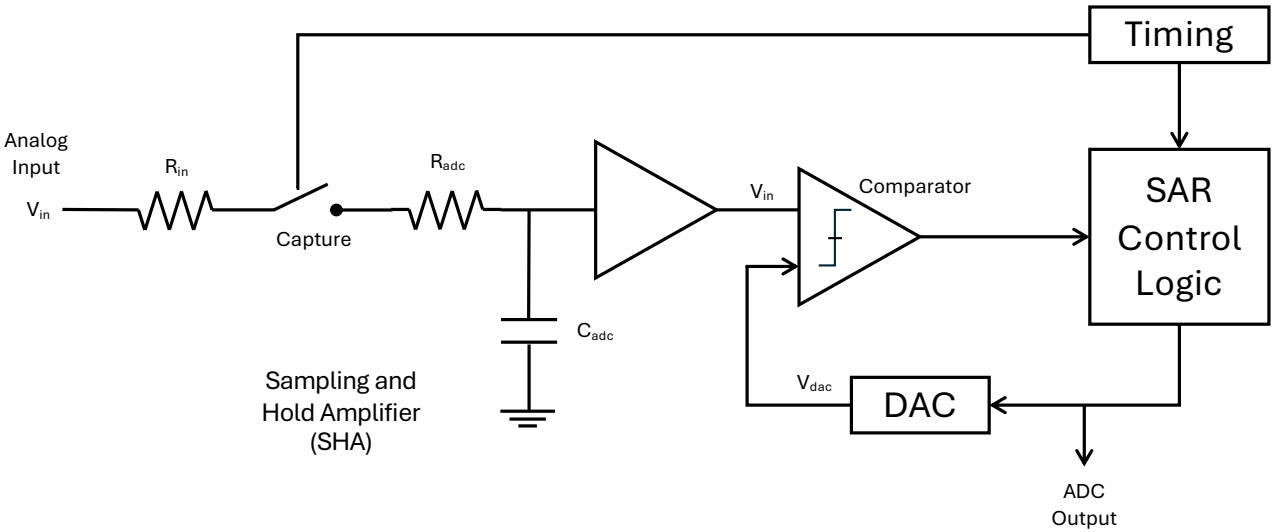


1. Set the internal DAC output V_{dac} to ½ V_{Ref} and compare V_{dac} with the ADC input V_{in}

2. If V_{in} is larger than ½ V_{ref}, set the most significant bit (MSB) of the ADC result. Otherwise, clear the MSB of the ADC

3. If V_{dac} is set to either ¾ V_{ref} or ¼ V_{ref}, depending on comparison result

SAR Digital Quantization



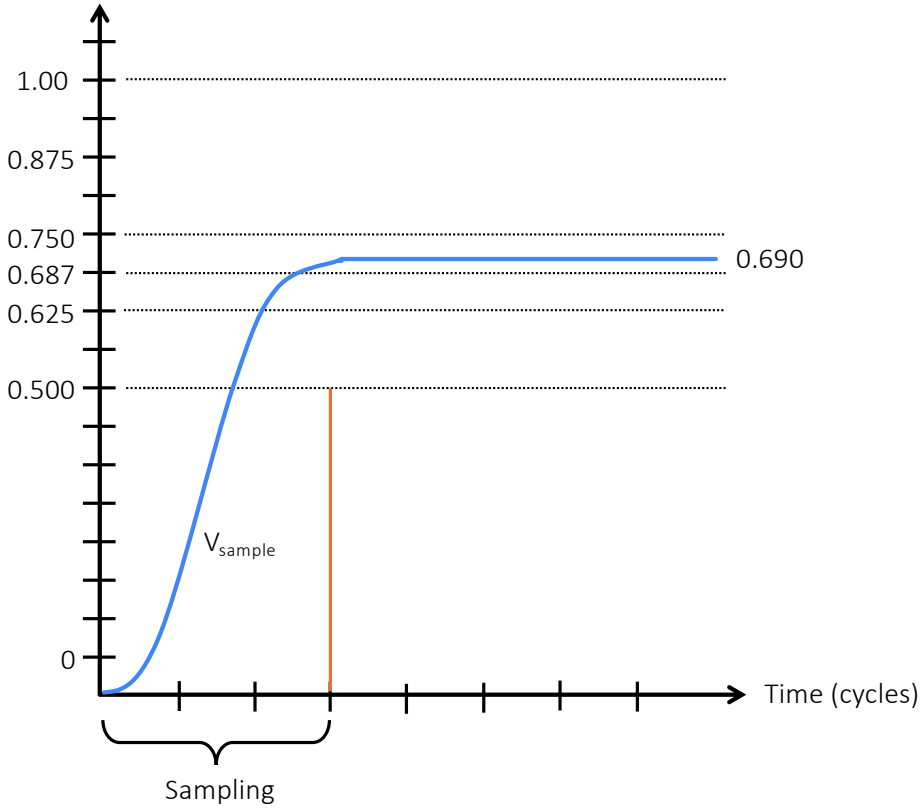
If the ADC has a resolution of *n* bits, the successive-approximation conversion takes *n* steps to complete

1. Set the internal DAC output V_{dac} to ½ V_{Ref} and compare V_{dac} with the ADC input V_{in}
2. If V_{in} is larger than ½ V_{ref}, set the most significant bit (MSB) of the ADC result. Otherwise, clear the MSB of the ADC
3. If V_{dac} is set to either ¾ V_{ref} or ¼ V_{ref}, depending on comparison result
4. Repeat until all bits of the ADC output have been determined

Example: SAR Digital Quantization

Input Voltage Range: 0 to 1V

After the input voltage is sampled,
$V_{dac} = ½ V_{ref} = 0.5V$

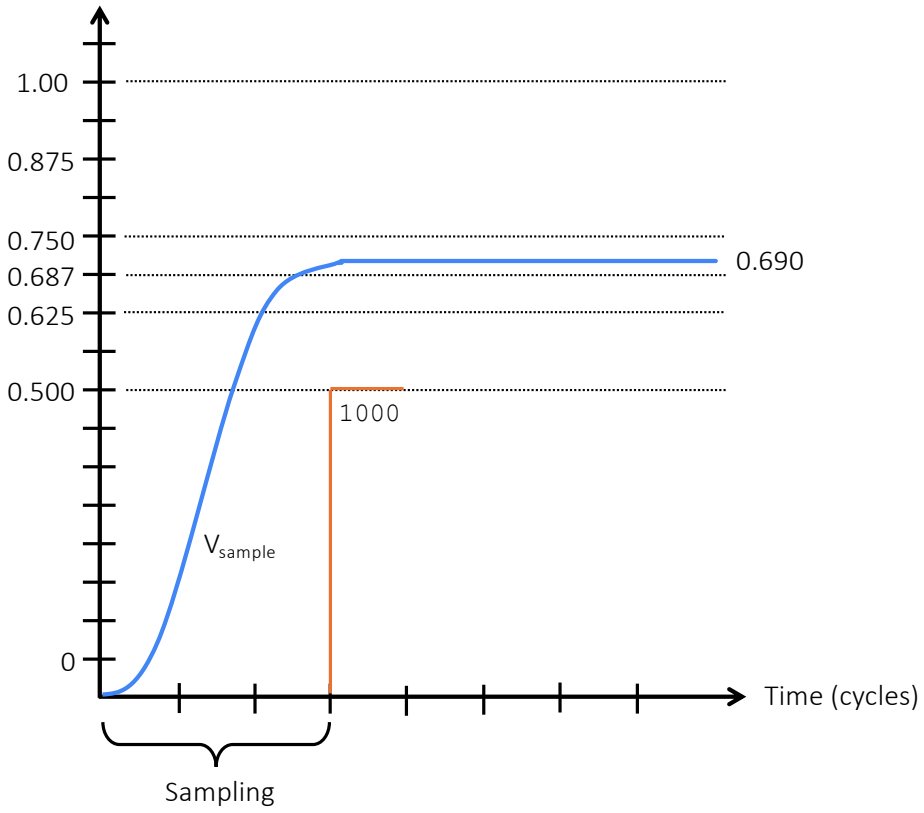


Example: SAR Digital Quantization

Input Voltage Range: 0 to 1V

After the input voltage is sampled,
$V_{dac} = ½ V_{ref} = 0.5V$

V_{sample} is **larger than** 0.5 so the MSB is set to 1



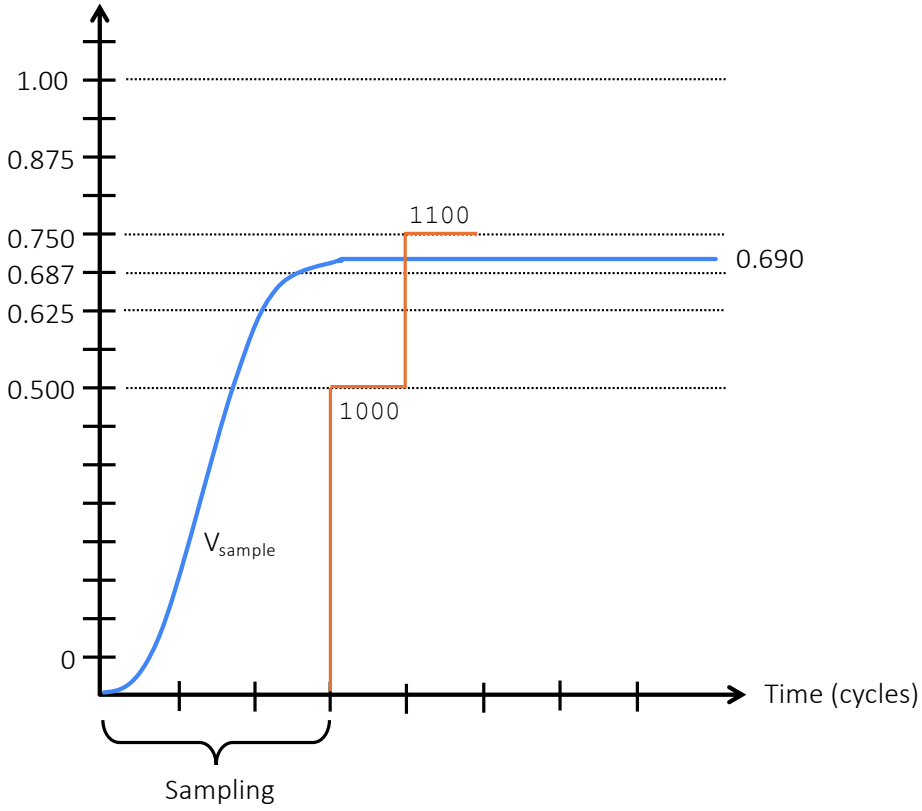
Example: SAR Digital Quantization

Input Voltage Range: 0 to 1V

After the input voltage is sampled,
$V_{dac} = ½ V_{ref} = 0.5V$

V_{sample} is **larger than** 0.5 so the MSB is set to 1

V_{dac} is set to 0.750V (¾ V_{ref}), which sets the ADC ouput to `0b110`



Example: SAR Digital Quantization

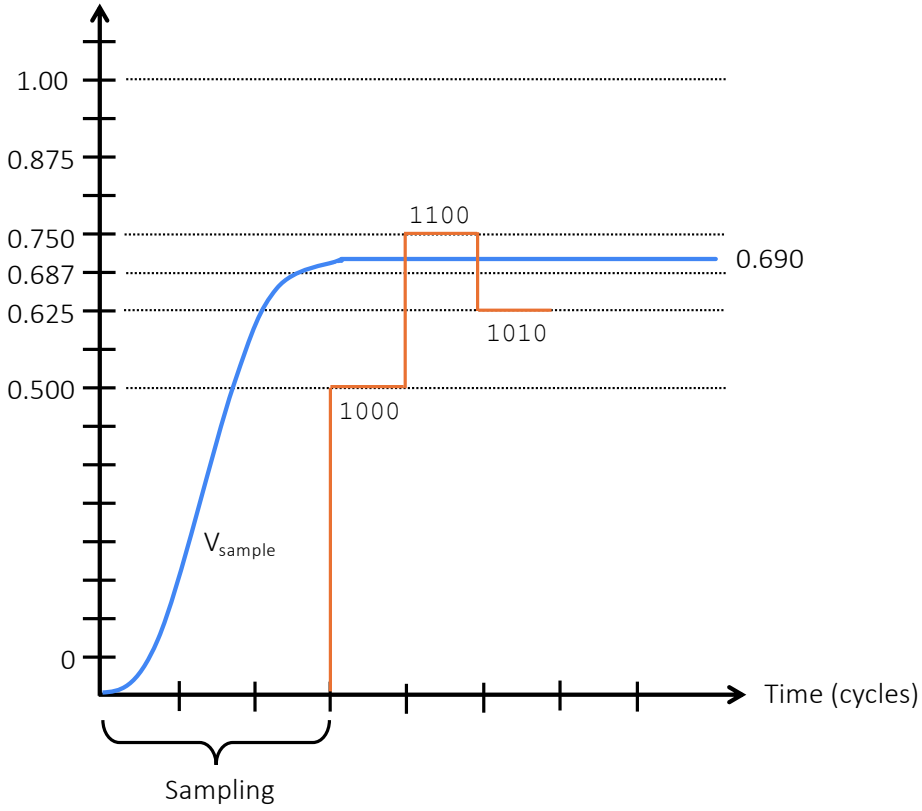
Input Voltage Range: 0 to 1V

After the input voltage is sampled,
$V_{dac} = ½ V_{ref} = 0.5V$

V_{sample} is **larger than** 0.5 so the MSB is set to 1

V_{dac} is set to 0.750V (¾ V_{ref}), which sets the ADC ouput to `0b110`

V_{sample} is **less than** 0.750V, so the the control logic sets **V_{dac} to 0.625V**



Example: SAR Digital Quantization

Input Voltage Range: 0 to 1V

After the input voltage is sampled,
$V_{dac} = ½ V_{ref} = 0.5V$

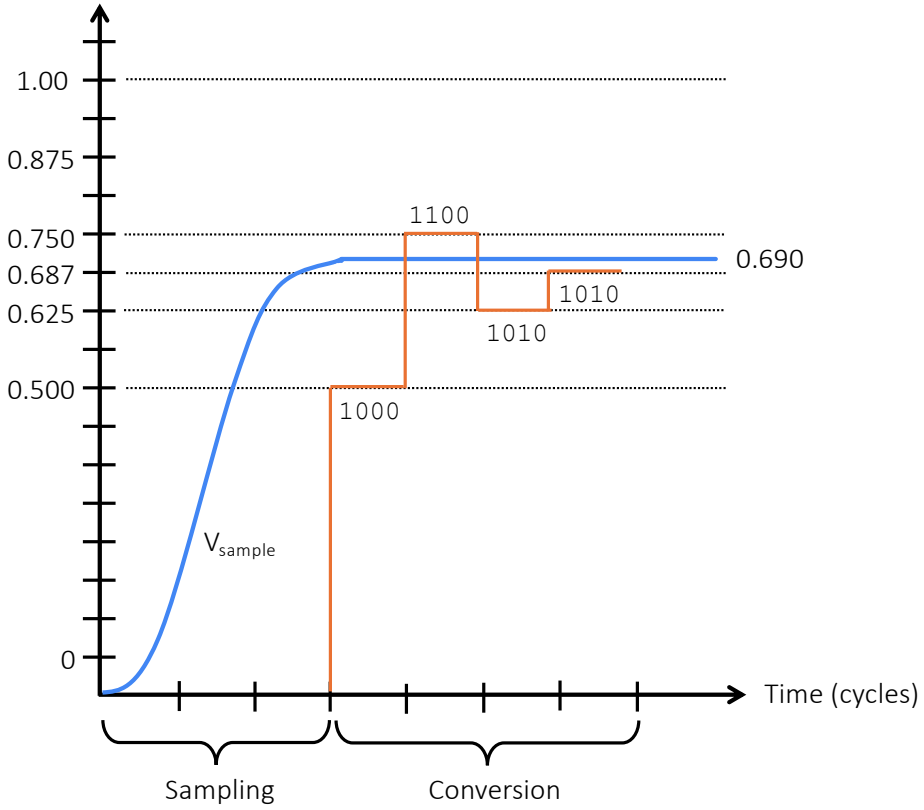
V_{sample} is **larger than** 0.5 so the MSB is set to 1

V_{dac} is set to 0.750V (¾ V_{ref}), which sets the ADC output to `0b110`

V_{sample} is **less than** 0.750V, so the the control logic sets **V_{dac} to 0.625V**

⋮

The **final ADC output is `0b1011`**, and the conversion process took four cycles

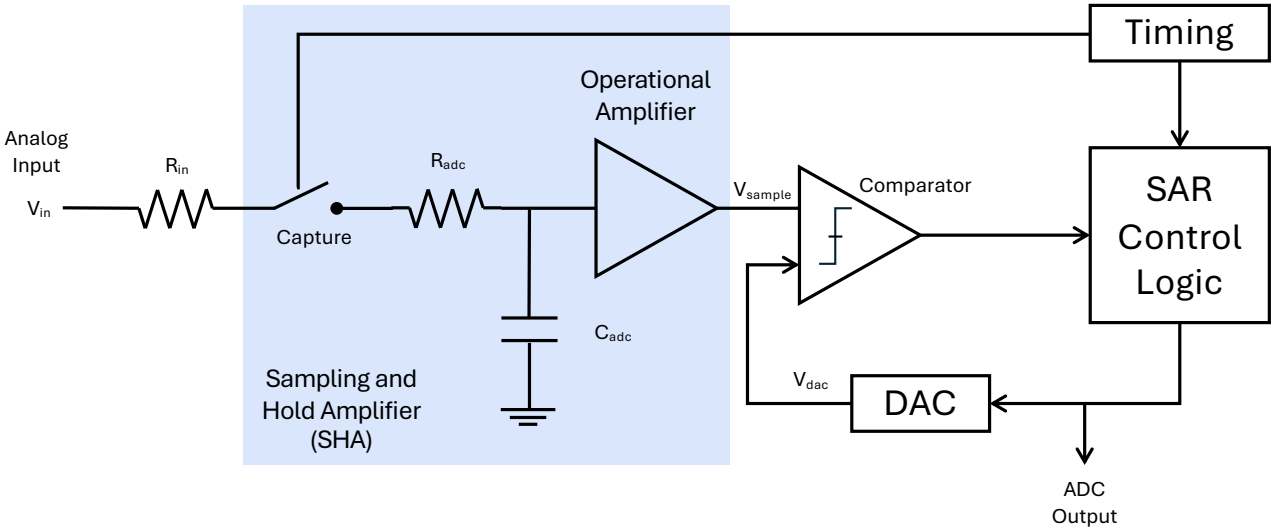


Sample and Hold Circuit

Sample and Hold

$$V_c(t) = V_{in} \times \left(1 - e^{-t/T_c}\right)$$

$$T_c = (R_{in} + R_{adc}) \times C_{adc}$$



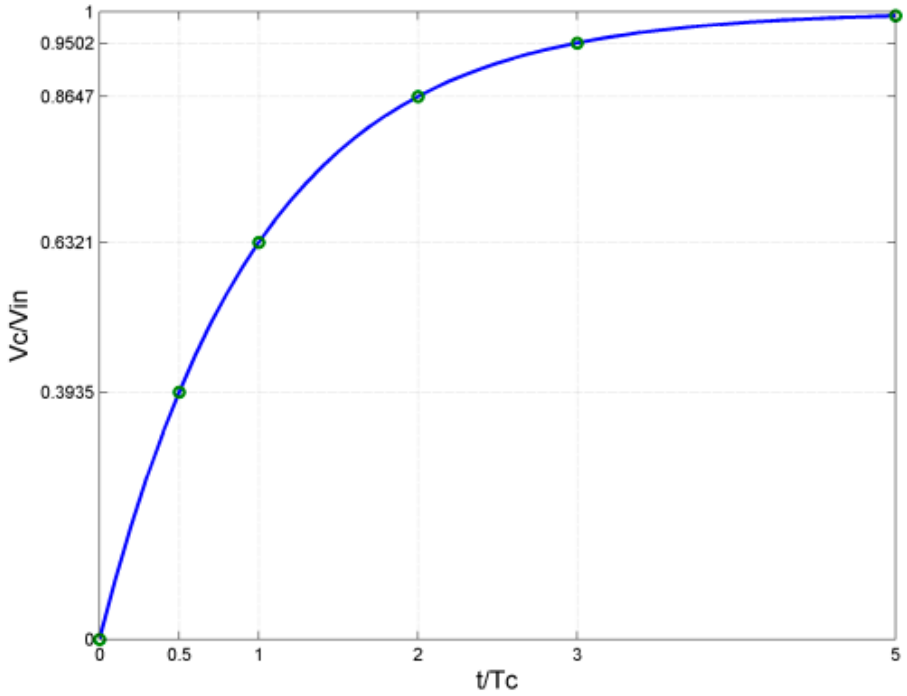
Sample and Hold

$$V_c(t) = V_{in} \times \left(1 - e^{t/T_c}\right)$$

$$T_c = (R_{in} + R_{adc}) \times C_{adc}$$

V_{in} cannot be sampled instantly

Example: if the switch is closed for a time period 3Tc, the voltage across the capacitor is only 95.02% of the input voltage



ADC Conversion Time

Recall...
If the ADC has a resolution of *n* bits, the successive-approximation conversion takes *n* steps to complete

$T_{ADC} = T_{sampling} + T_{conversion}$

$T_{sampling}$ is software configurable

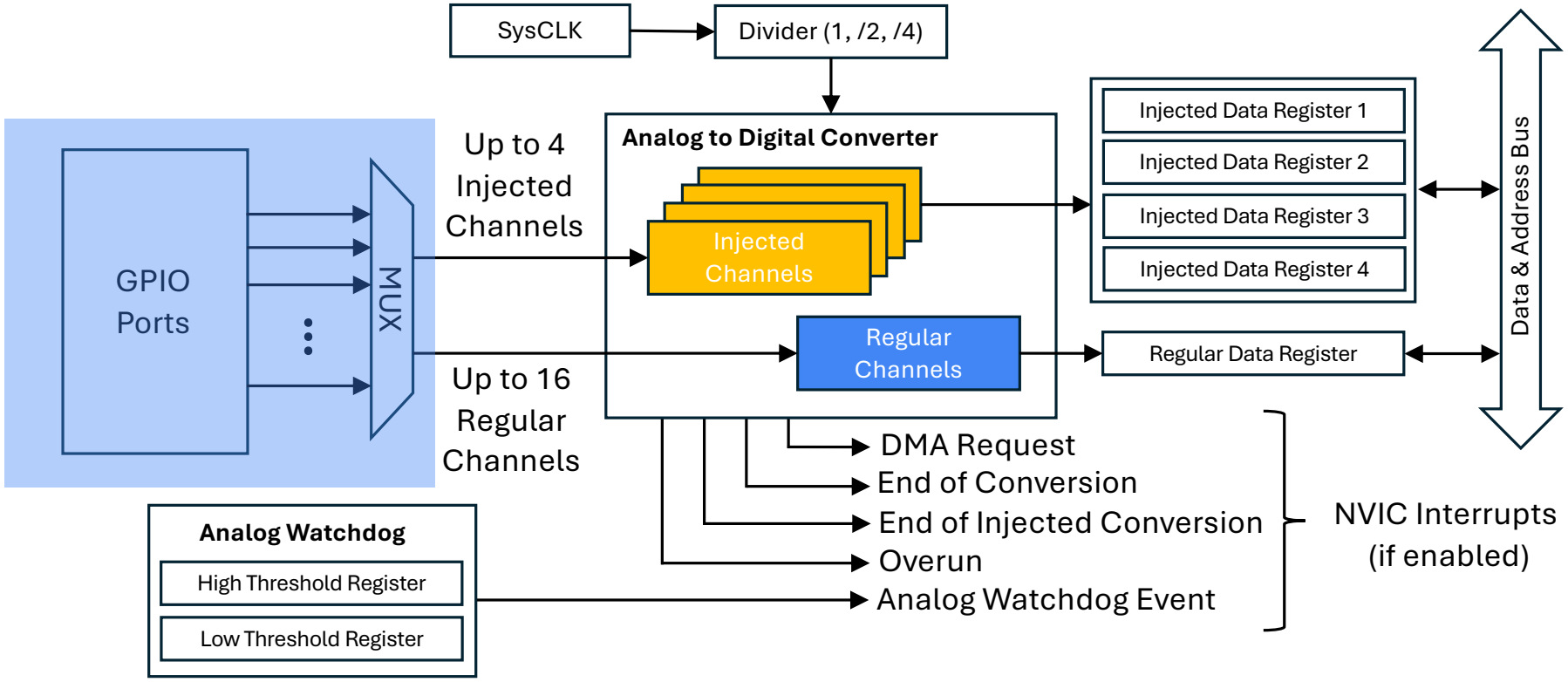
For a 12-bit ADC what is the conversion time of the **sampling time is set to 4 cycles** and the **ADC clock is set to 16MHz**?

$T_{ADC} = 4 + 12 = 16\ cycles = 1\mu s$

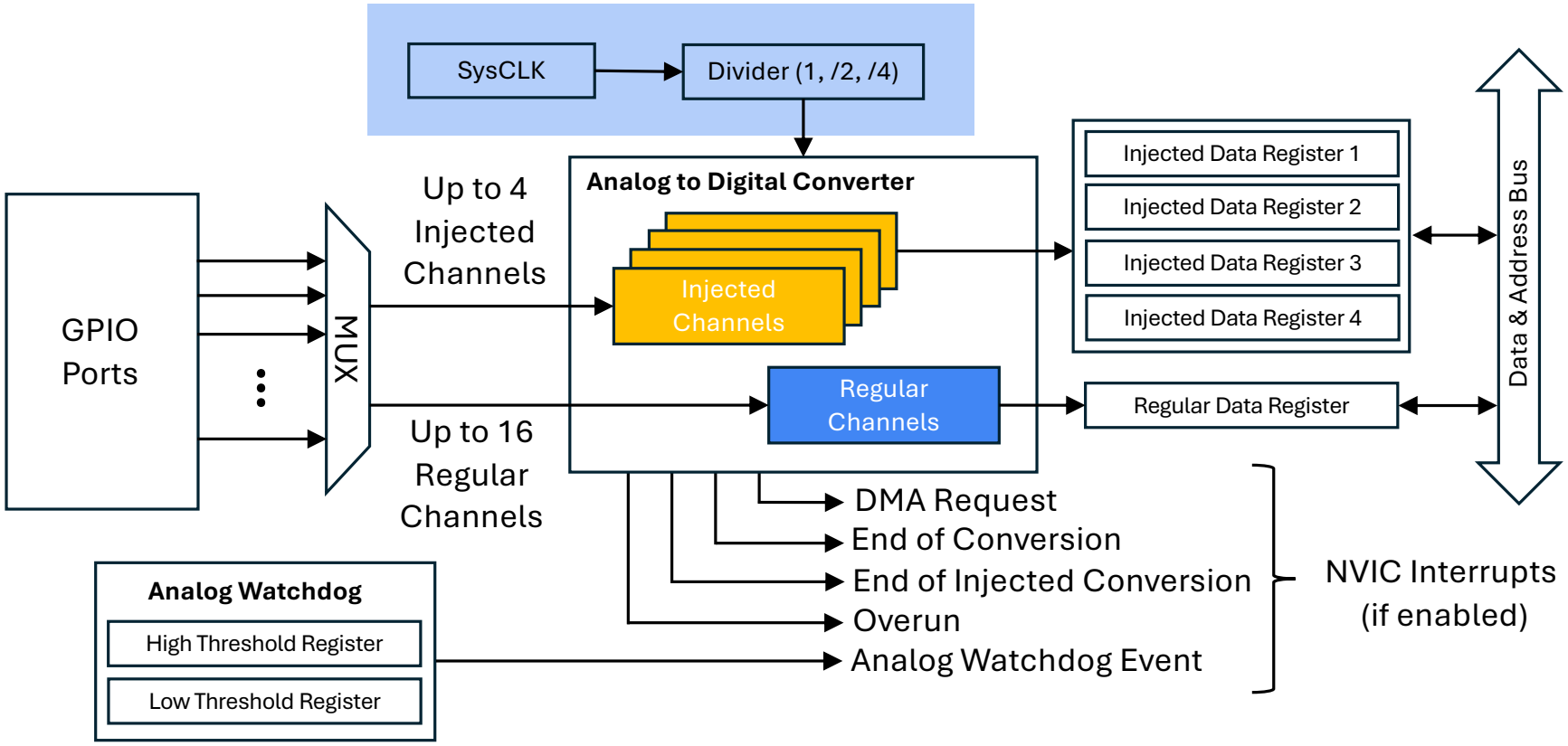
STM32L4 ADC

- Three ADC Modules (ADC1, ADC2, ADC3)
- Resolution: 12, 10, 8, or 6 bits
  - Depends on the application
- Driven by the same clock
  - Selected from three different clock sources

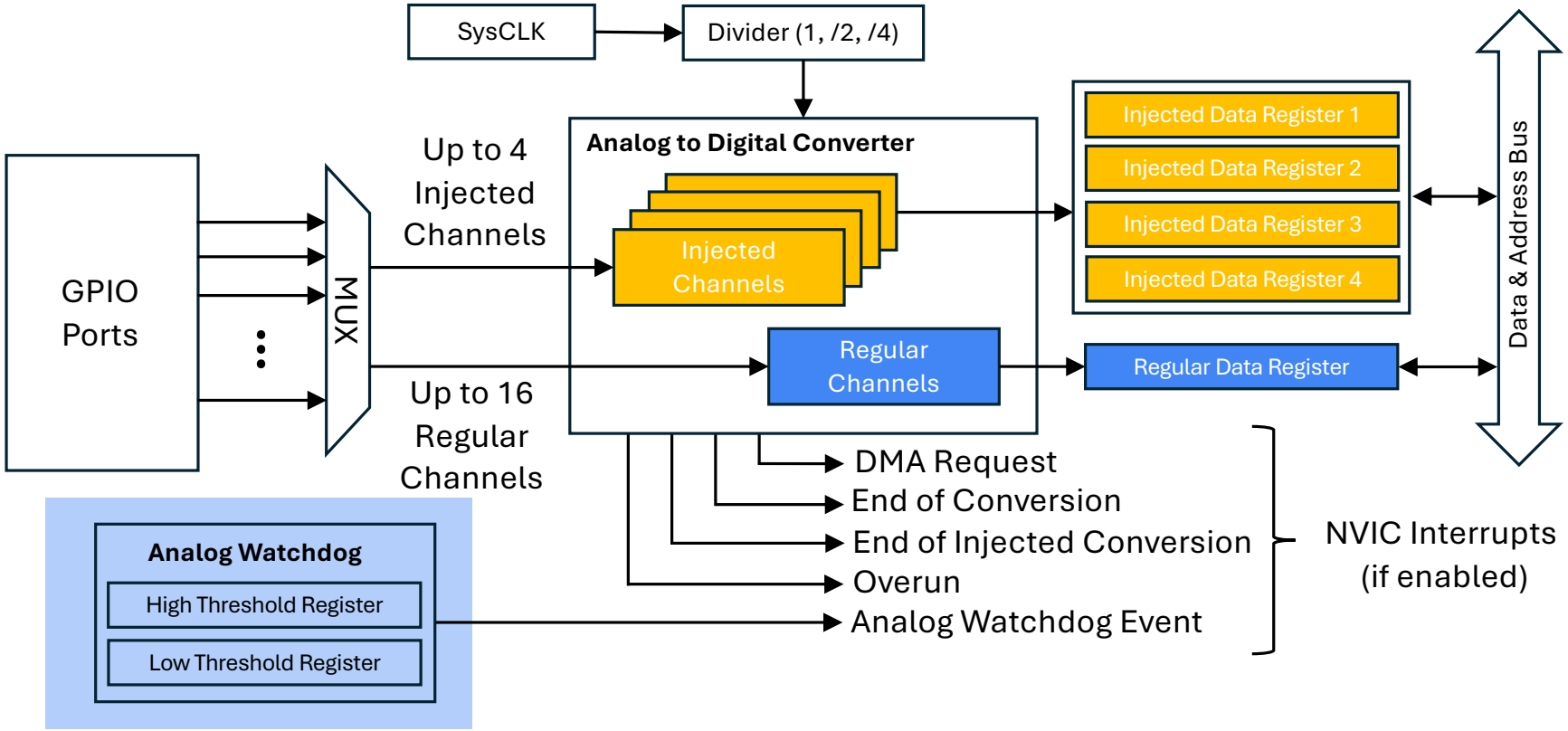
STM32L4 ADC


SysCLK
Divider (1, /2, /4)
Analog to Digital Converter
Injected Data Register 1
Injected Data Register 2
Injected Data Register 3
Injected Data Register 4
Data & Address Bus
GPIO Ports
MUX
Up to 4 Injected Channels
Injected Channels
Regular Channels
Regular Data Register
Up to 16 Regular Channels
DMA Request
End of Conversion
End of Injected Conversion
Overun
Analog Watchdog Event
NVIC Interrupts (if enabled)
Analog Watchdog
High Threshold Register
Low Threshold Register

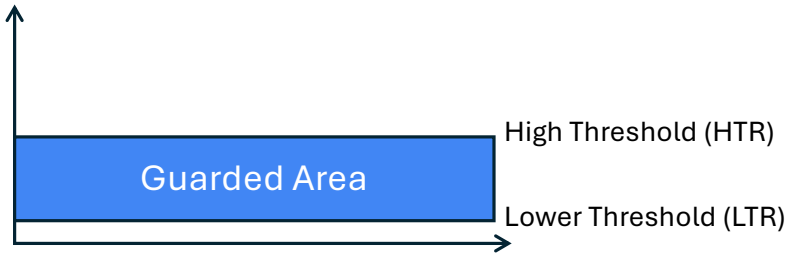
STM32L4 ADC


SysCLK
Divider (1, /2, /4)
Analog to Digital Converter
GPIO Ports
MUX
Up to 4 Injected Channels
Up to 16 Regular Channels
Injected Channels
Regular Channels
Injected Data Register 1
Injected Data Register 2
Injected Data Register 3
Injected Data Register 4
Regular Data Register
Data & Address Bus
DMA Request
End of Conversion
End of Injected Conversion
Overun
Analog Watchdog Event
NVIC Interrupts (if enabled)
Analog Watchdog
High Threshold Register
Low Threshold Register

STM32L4 ADC


SysCLK
Divider (1, /2, /4)
Analog to Digital Converter
Injected Data Register 1
Injected Data Register 2
Injected Data Register 3
Injected Data Register 4
Data & Address Bus
GPIO Ports
MUX
Up to 4 Injected Channels
Injected Channels
Regular Channels
Regular Data Register
Up to 16 Regular Channels
DMA Request
End of Conversion
End of Injected Conversion
Overun
Analog Watchdog Event
NVIC Interrupts (if enabled)
Analog Watchdog
High Threshold Register
Low Threshold Register

Analog Watchdog



If $V < V_{LTR}$ or $V > V_{HTR,}$ the analog watchdog (AWD) flag (in the ADC status register) is set, generating an interrupt to the processor

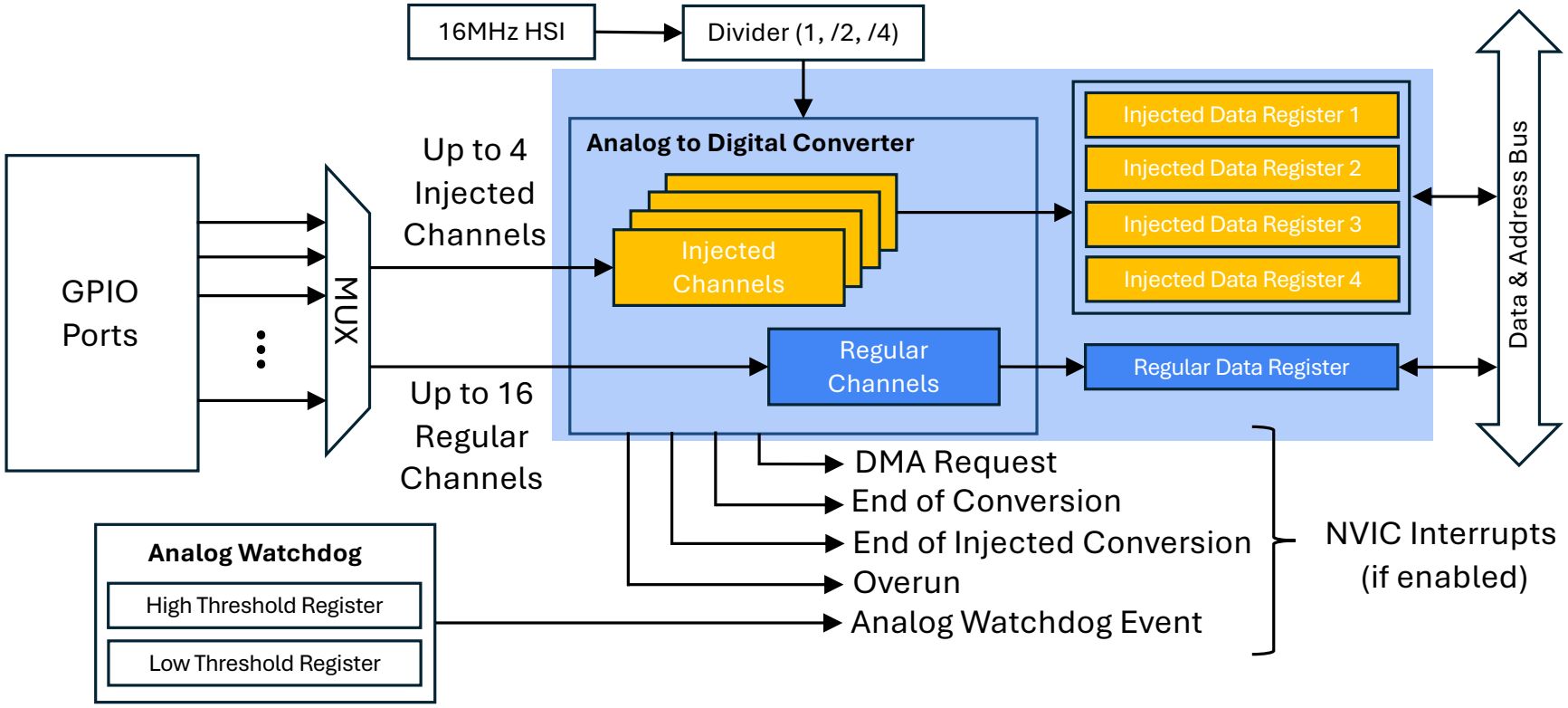
The monitoring is automatically performed by hardware, not software

Convenient and efficient feature

Helps the processor detect exceptions and recover from specific situations

- **Example:** monitor sensor data and raise an alarm on some level

STM32L4 ADC


16MHz HSI
Divider (1, /2, /4)
Analog to Digital Converter
Injected Data Register 1
Injected Data Register 2
Injected Data Register 3
Injected Data Register 4
Data & Address Bus
GPIO Ports
MUX
Up to 4 Injected Channels
Injected Channels
Regular Channels
Regular Data Register
Up to 16 Regular Channels
DMA Request
End of Conversion
End of Injected Conversion
Overun
Analog Watchdog Event
NVIC Interrupts (if enabled)
Analog Watchdog
High Threshold Register
Low Threshold Register

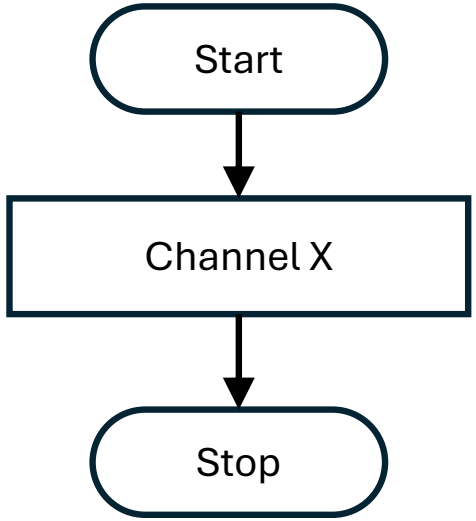
ADC Conversion Modes

The ADC module performs conversions on selected channels in either **single-conversion mode** or **continuous-conversion mode**.

ADC Conversion Mode

Single Conversion On One Input Channel (Regular Group)

1. The target channel is determined by the SQ1[4:0] bits in the SQR1 register
2. The 16-bit ADC data register (ADC_DR) holds the conversion result
3. After the conversion, the *end of regular conversion flag* (EOC) is set in the ISR register.
4. The ADC module generates an interrupt request if the EOCIE bit is set in the interrupt enable register (IER)

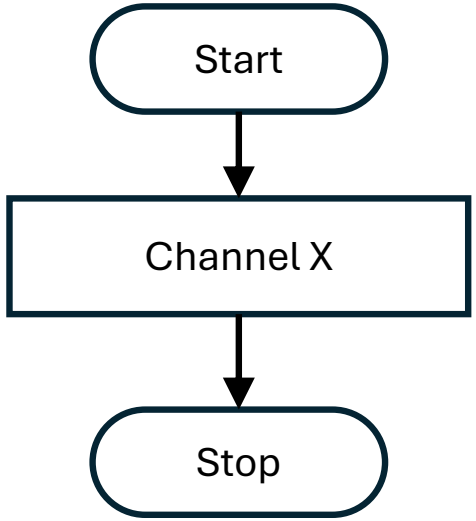


ADC Conversion Mode

Single Conversion On One Input Channel

(Injected Group)

1. The JSQ1[4:0] bits in the JSQR register selects conversion channel
2. The 16-bit register (ADC_JDR1) holds the conversion result
3. After the conversion, the *end of injected conversion flag* (EOC) is set in the ISR register.
4. The ADC module generates an interrupt request if the JEOCIE bit is set in the interrupt enable register (IER)

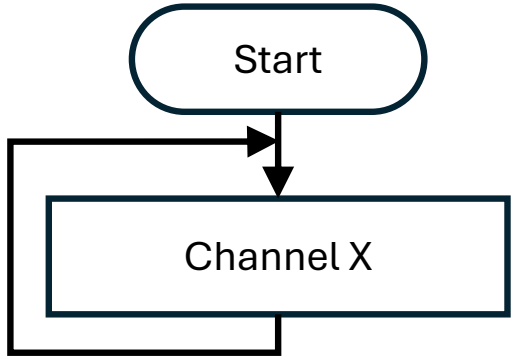


ADC Conversion Mode

Single Channel, Continuous Conversion

- ADC module automatically starts a new conversion immediately after it finishes one.

- The last conversion result is saved in the ADC_DR register for a regular channel

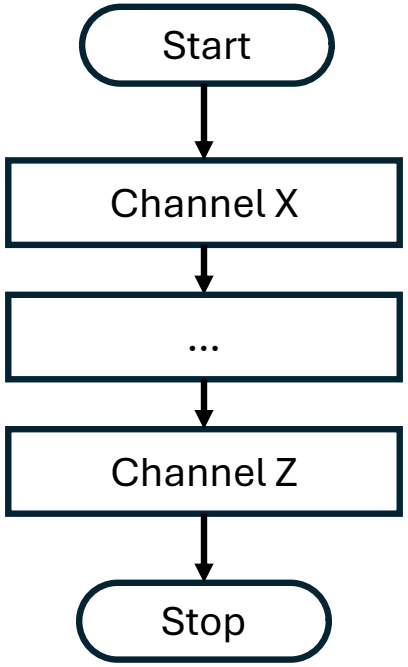


ADC Conversion Mode

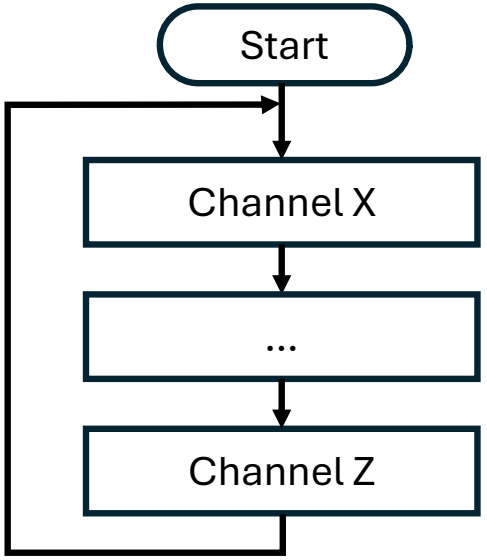
Scan Mode: The ADC module can perform sampling and conversion on a set of pre-defined input channels in a round-robin fashion.

Rugular:
Channels selected by ADC_SQRx

Injected:
Channels selected by ADC_JSQR



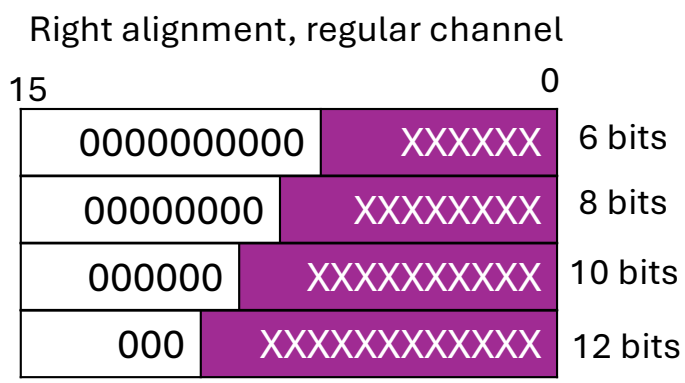
Scan with Single Conversion

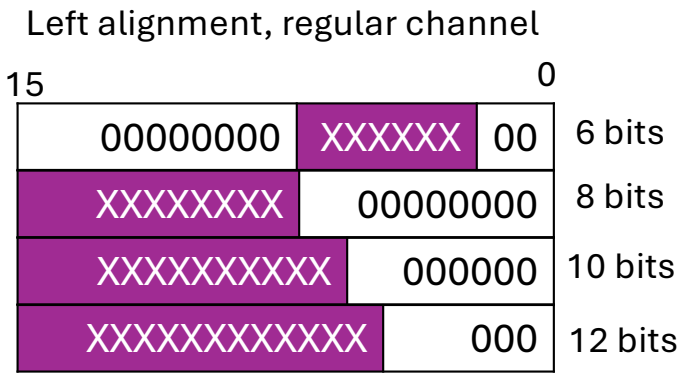


Scan with Continuous Conversion

ADC Data Alignment

- Software can change the ADC resolution (12, 10, 8, or 6 bits)
- Each ADC data register has 16 bits





Digital to Analog Conversion

What is a digital-to-analog converter (DAC)?

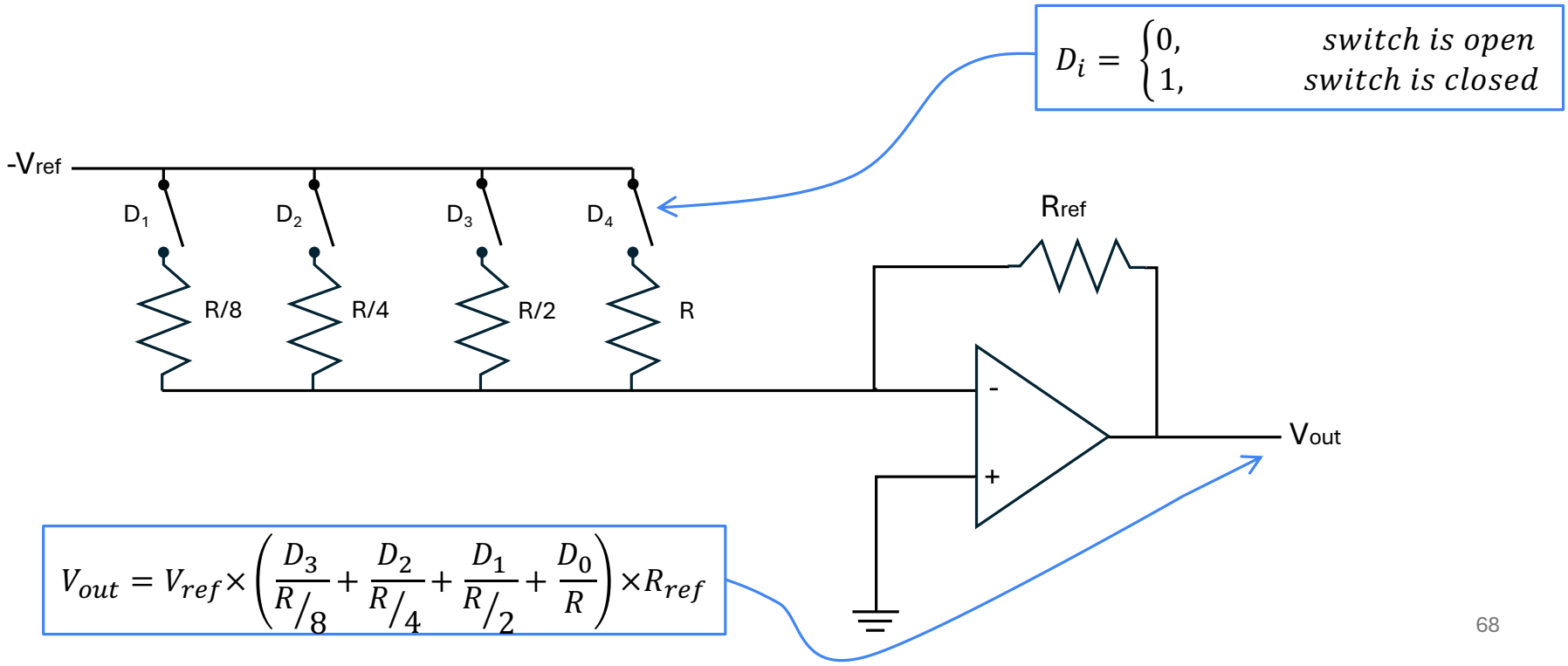
A digital-to-analog converter (DAC) transforms a finite-precision digital number to an analog voltage

Example →



DAC Architecture

A four-bit digital-to-analog converter (DAC)



DAC Characteristics

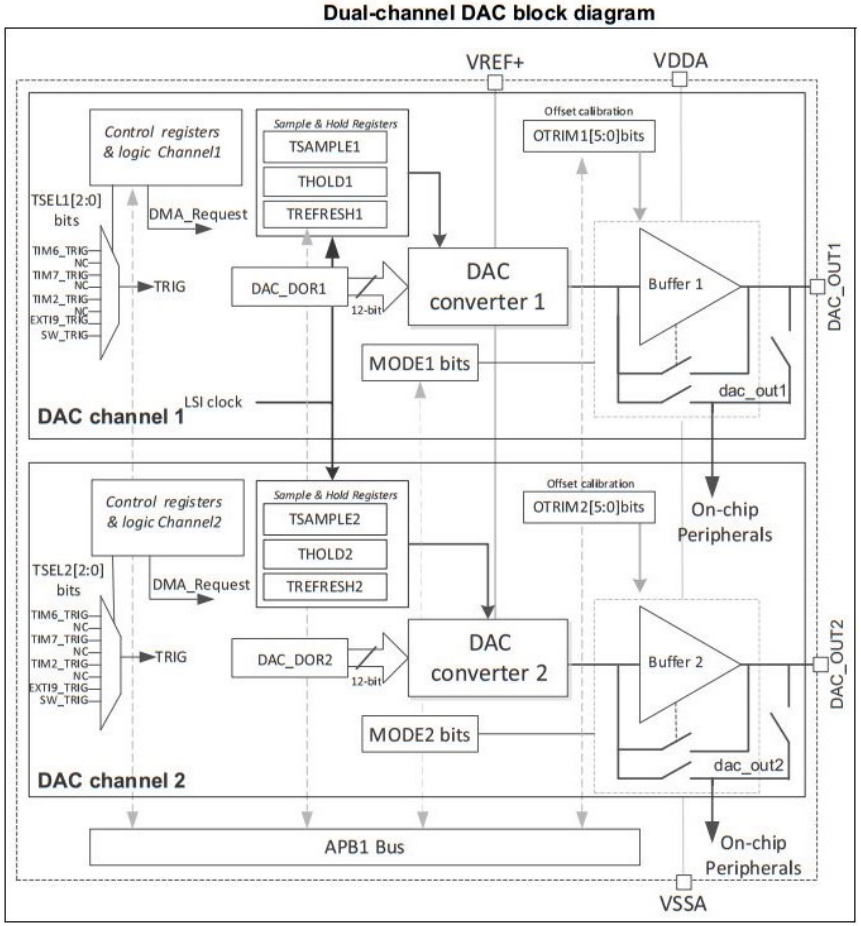
- **Precision:** the number of distinguishable DAC outputs (e.g., 1024 alternatives, 10bits)
- **Range:** the maximum and minimum DAC output
- **Setting Time:** the interval from an update of a DAC's digital input to the instant when the DAC output becomes stable within a specified percentage
- **Resolution:** the smallest change that occur in the analog output as the digital input varies (the change in output that occurs when the
- **Glitch:** the first peak transient that appears at the DAC output

DAC on the STM32L

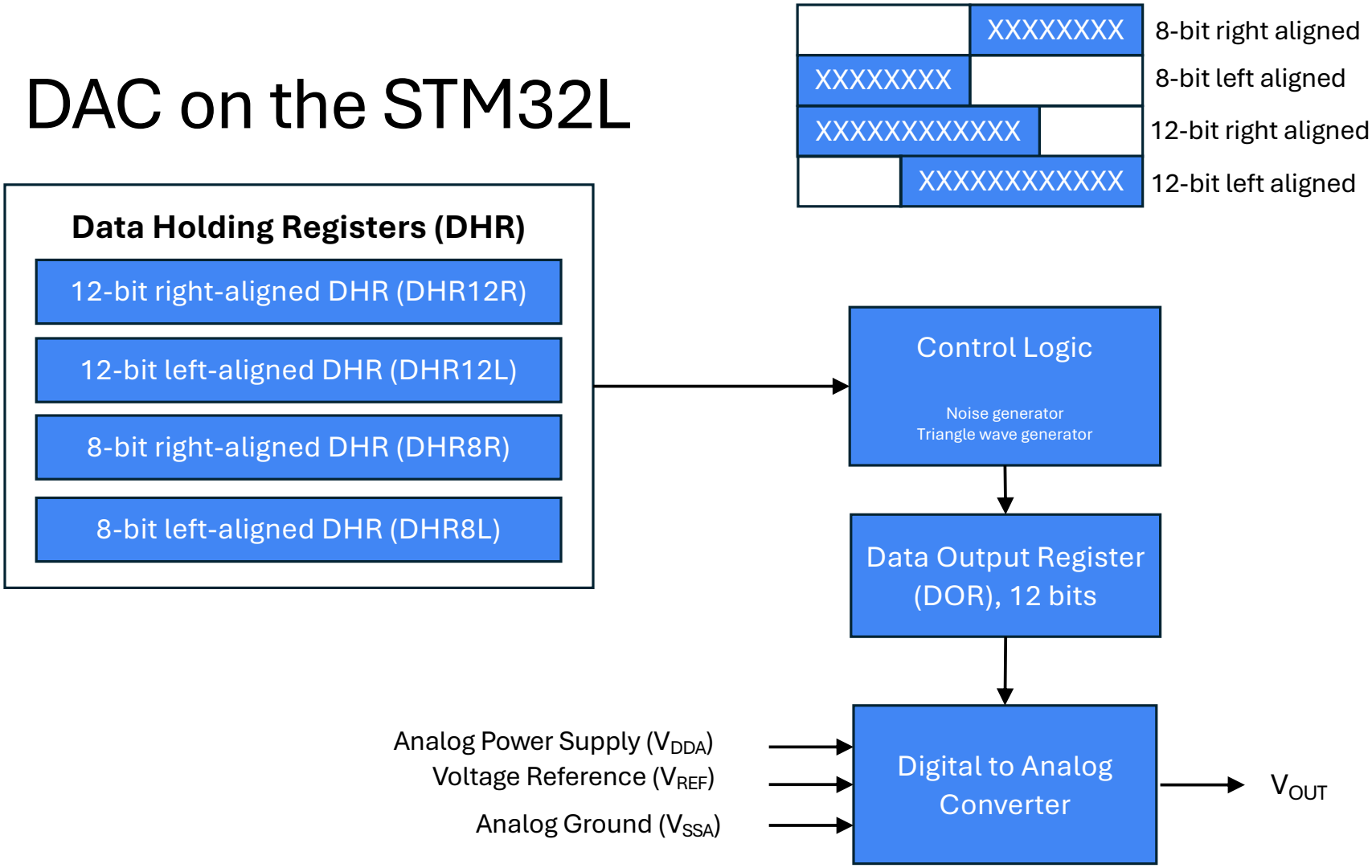
STM32L processor has two independent DAC converters, with one channel in each converter

Can be configured as a resolution of 8-bits or 12-bits

Two converters can update their output signals independently or synchronously



DAC on the STM32L

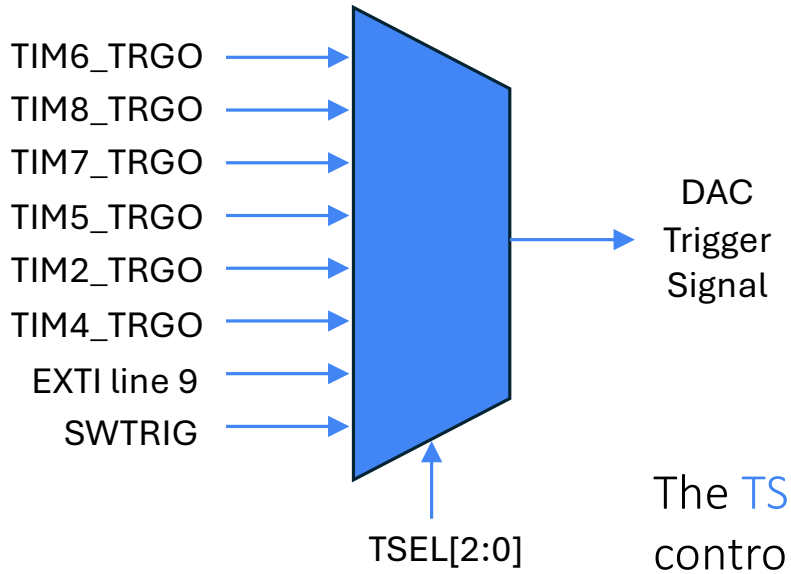


Conversion Trigger

Software triggers, external timers, and internal timers can trigger DAC conversions

Timer Trigger:
Conversion starts after detecting the rising edge of timer trigger output

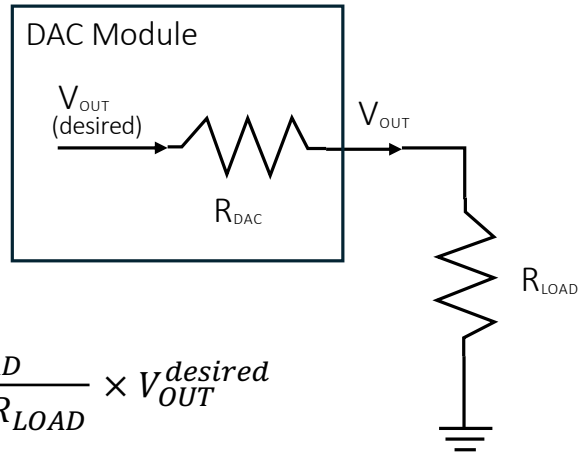
Software Trigger:
Conversions starts after the trigger bit in DAC_SWTRIGR is set by software



The TSEL bits in the control register (DAC_CR) select the DAC trigger

Loading Effects

When the DAC output is used to drive an external load (e.g., earphones) the voltage output may be lower than the desired value due to loading effects

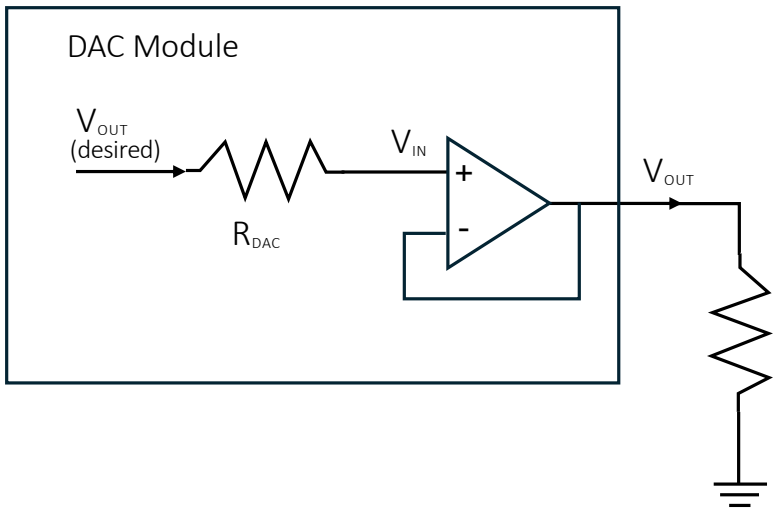


$$V_{OUT} = \frac{R_{LOAD}}{R_{DAC} + R_{LOAD}} \times V_{OUT}^{desired}$$

How do we fix this?

Buffered Output

Use an internal output buffer to avoid the load impedance problem!



Close to infinity

$$V_{IN} = \frac{R_{IN,Buffer}}{R_{DAC} + R_{IN,Buffer}} \times V_{OUT}^{desired} \approx \frac{\infty}{R_{DAC} + \infty} \times V_{OUT}^{desired} \approx V_{OUT}^{desired}$$

Close to zero

$$V_{OUT} = \frac{R_{OUT,Buffer}}{R_{LOAD} + R_{OUT,Buffer}} \times V_{IN} \approx \frac{R_{LOAD}}{R_{LOAD} + 0} \times V_{IN} \approx V_{OUT}^{desired}$$

The output buffer can be enabled or disabled by the DAC_CR_BOF bit in the control register DAC_CR

Acknowledgements

- These lecture slides were adapted from the University of Michigan's EECS 373 course (many thanks to Alanson Sample!)

- These lecture slides also leverage materials from the following books:
  - Embedded Systems with ARM Cortex-M Microcontrollers in Assembly Language and C
    - Yifeng Zhu
  - The Definitive Guide to ARM Cortex – M3 and Cortex – M4 Processors
    - Joseph Yiu
  - Introduction to ARM Cortex – M Microcontrollers, Embedded Systems
    - Jonathan W. Valvano