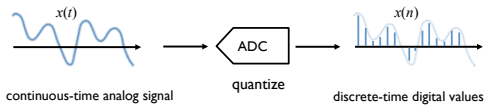
# Sec 6. Analog-to-Digital

## Converting Signals AD-DC Converters

1

## Analog-to-Digital Converter (ADC)

- ADC is important almost to all application fields
- Converts a continuous-time voltage signal within a given range to discrete-time digital values to quantify the voltage's amplitudes




2

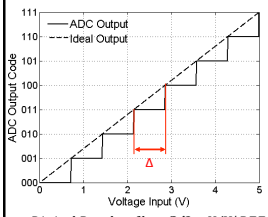
## Analog-to-Digital Converter (ADC)

- Three performance parameters:
  - sampling rate
  - resolution
  - power dissipation
- Many ADC implementations:
  - sigma-delta
  - successive-approximation
  - pipeline

3

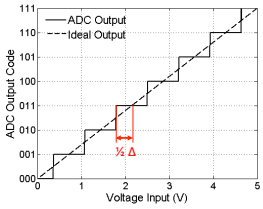
## Resolution

- Resolution is determined by number of bits (in binary) to represent an analog input.
- Example of two quantization methods (N = 3)




$Digital\ Result = floor(2^3 \times V/V_{REF})$

Max quantization error = $\Delta = V_{REF}/2^3$




$Digital\ Result = round(2^3 \times V/V_{REF})$

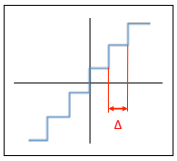
Max quantization error = $\pm\frac{1}{2}\Delta = \pm V_{REF}/2^4$

$round(x) = floor(x+0.5)$

4

## Quantization Error

- For N-bit ADC, it is limited to ±½Δ
- Δ = is the step size of the converter.



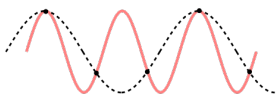

- Example: for 12-bit ADC and input voltage range [0, 3V]

$$Max\ Quantization\ Error = 1/2\,\Delta = 3V/2\times 2^{12} = 0.367mV$$

5

## Minimum Sampling Rate: Nyquist–Shannon Sampling Theorem

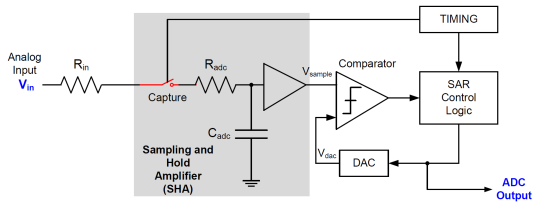
- In order to be able to reconstruct the analog input signal, the sampling rate should be at least twice the maximum frequency component contained in the input signal
- Example of two sine waves have the same sampling values. This is called aliasing.




- Antialiasing (beyond the scope of this course)
  - Pre-filtering: use analog hardware to filtering out high-frequency components and only sampling the low-frequency components. The high-frequency components are ignored.
  - Post-filtering: Oversample continuous signal, then use software to filter out high-frequency components

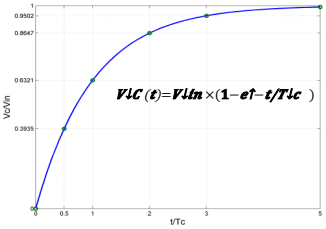
6

## Successive-approximation (SAR) ADC






## Determining Minimum Sampling Time




$$Vc(t) = Vin \times (1 - e^{-t/Tc})$$

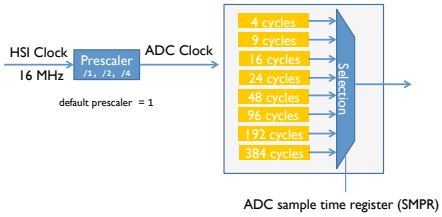
Sampling time is software programmable!

Larger sampling time → Smaller sampling error, Slower ADC speed

*Tradeoff*

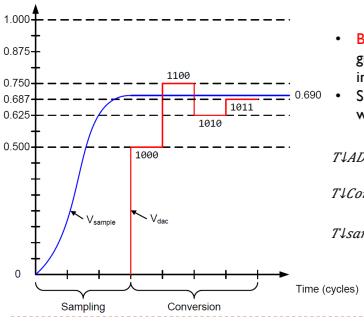


## Programming ADC Sampling Time






## Successive-approximation (SAR) ADC




- Binary search algorithm to gradually approaches the input voltage
- Settle into ±½ LSB bound within the time allowed

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

$$T_{Conversion} = N \times T_{ADC\_Clock}$$

$T_{sampling}$ software configurable



## ADC Conversion Time

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

Suppose ADCCLK = 16 MHz and Sampling time = 4 cycles

For 12-bit ADC

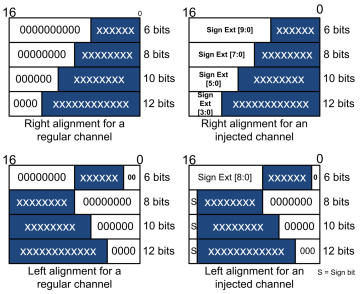
$$T_{ADC} = 4 + 12 = 16\ cycles = 1\mu s$$

For 6-bit ADC

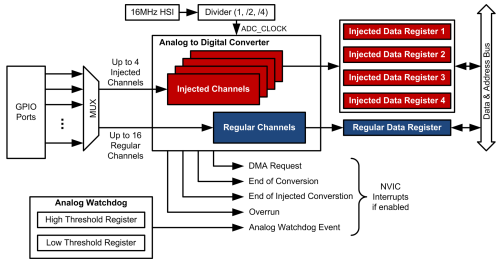
$$T_{ADC} = 4 + 6 = 10\ cycles = 625ns$$



## Data Alignment

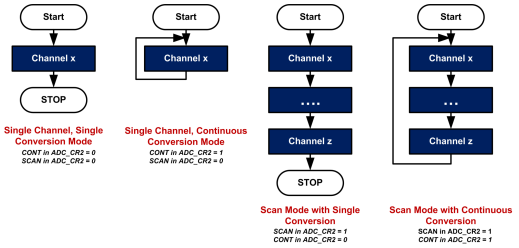


## ADC: Regular vs injected



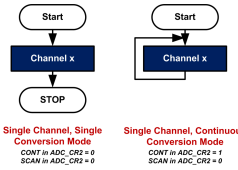

13

## ADC Mode




14

## ADC Mode




**Regular channel:**

1. Set SWSTART in ADC_CR2
2. The channel is selected by SQ1[4:0] in SQR5
3. Result is stored in ADC_DR
4. EOC is set after conversion
5. Interrupt is generated if EOCIE is set

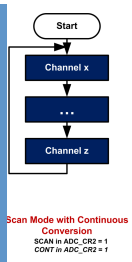
**Injected channel:**

1. Set JSWSTART in ADC_CR2
2. The channel is selected by JSQ1[4:0] in JSQR
3. Result is stored in ADC_JDR1
4. JEOC is set after conversion
5. Interrupt is generated if JEOCIE is set

15

## ADC Mode

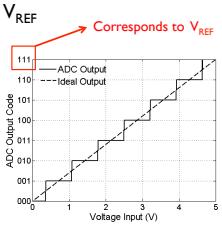

- Channels are selected by ADC_SQRx registers for regular channels, and by ADC_JSQR register for injected channel
- All channels in a regular group share the same result register ADC_DR. Make sure to read data between consecutive sampling.




16

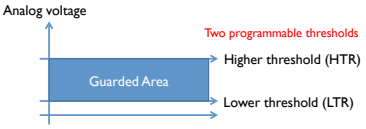
## $V_{REF}$

- Some chips does not expose $V_{REF}$ to a pin
  - STM32L LQFP64 does not have $V_{REF}$ pin
  - STM32L LQFP100 does
- Infer internal $V_{REF}$
  - How?




17

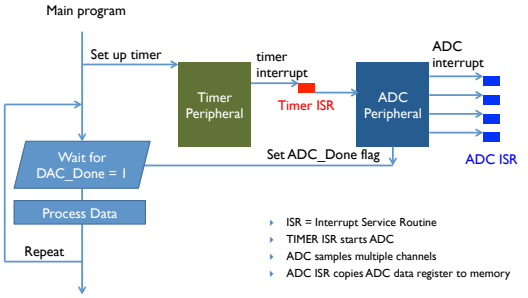
## Analog Watchdog




- If $V < V_{LTR}$ or $V > V_{HTR}$, the analog watchdog (AWD) flag (in ADC Status Register) is set, generating an interrupt to the processor
- The monitor is automatically performed by hardware, not software
- Convenient and efficient feature
- Help processor detect exceptions and recover from specific situations
  - For example, monitor sensor data and raise alarm on some level.

18

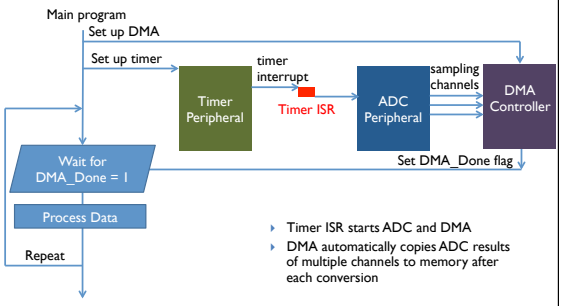
## Example: ADC with Timer Interrupts




- ISR = Interrupt Service Routine
- TIMER ISR starts ADC
- ADC samples multiple channels
- ADC ISR copies ADC data register to memory



## Example: ADC with Timer and DMA




- Timer ISR starts ADC and DMA
- DMA automatically copies ADC results of multiple channels to memory after each conversion



## Exercises for ADC #1

Suppose $V_{REF}$ = 1.5V, what is the minimum number of bits required to achieve a resolution of 1mV?

- **Solution:**
  - **Unique number of values = 1.5/1mV + 1 = 1501**
  - **Thus we need a minimum of 11 bits**
  - **$2^{11}$ = 2048, $2^{10}$ = 1024**



## Exercises for ADC #2

Successive-approximate (SAR) ADC is widely used. Suppose the ADC has a resolution of 14 bits, and the time for sampling and hold is set as 6 clock cycles. How many clock cycles are required to complete one analog-to-digital conversion?

- **Solution:**
  - **Sampling/Hold time + Conversion Time = 6 + 14 = 20 cycles**



## Digital-to-analog converter (DAC)

- Converts digital data into a voltage signal by a N-bit DAC

$$DAC\,output = Vref \times Digital\,Value/2\uparrow N - 1$$

- For 12-bit DAC

$$DAC\,output = Vref \times Digital\,Value/4095$$

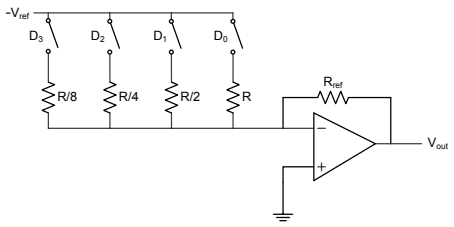
- Many applications:
  - digital audio
  - waveform generation
- Performance parameters
  - speed
  - resolution
  - power dissipation



## DAC Implementations

- Pulse-width modulator (PWM)
- Binary-weighted resistor (We will use this one as an example)
- R-2R ladder (A special case of binary-weighted resistor)

## Binary-weighted Resistor DAC

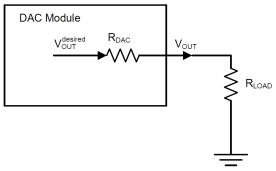



$$V1out = V1ref \times R1ref\ /R \times (D13 \times 2\uparrow 3 + D12 \times 2\uparrow 2 + D11 \times 2 + D10)$$



## Buffered Output

- Load Effect

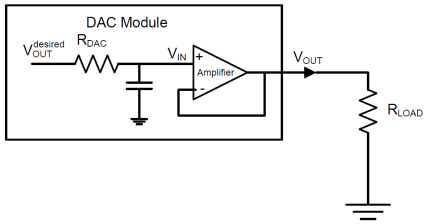



$$V1OUT = R1LOAD\ /R1DAC + R1LOAD \times V1OUT\uparrow desired$$



## Buffered Output

- Use buffer to remove load effect




$$V1OUT \approx V1OUT\uparrow desired$$



## Exercise for DAC

Assume an audio is recorded at a rate of 44,100 Hz, and the DAC is driven by the timer trigger output (TGRO). What is the time interval between two consecutive triggers? If the timer is driven by the HSI clock (16 MHz), how do you set the timer prescaler register (PSC) and the auto-reload register (ARR)? Show your calculations.

- **Solution: The solution is not unique.**
  - **HSI = 16 MHz**
  - **(1+ARR)(1+PSC) = 16MHz/44.1KHz = 362.8118**
  - **We can set PSC = 0, ARR = 363 – 1 = 362**
  - **We can also set PSC = 1, ARR = 362.8/2 – 1 = 180.4 = 180**
- ARR: CNT counts from 0 to ARR



## Timer Exercise #1

Suppose we want to generate a PWM signal with a duty cycle of 0.25. The processor clock is 32 MHz. We want the PWM signal to have a fixed frequency of 320 Hz. How would you design the prescaler (PSC), auto-reload register (ARR), and compare and capture (CCR)? Show your calculation. The timer output mode is set as follows: the PWM output is high if the counter is larger than or equal to the content of CCR.

- **Solution: The solution is not unique. Let's start with selecting prescaler (PSC)**
  - $PWM\ Signal\ Frequency * (ARR+1)(ARR+1) = 320Hz * (ARR+1)(PSC+1) = 32MHz$
  - $(PSC+1)(ARR+1) = 100000$
  - **We can set PSC = 99 and ARR = 999**
  - **In order to achieve a duty cycle of 0.25, we should set CCR to 750**