forward together · saam vorentoe · masiye phambili

Computer Systems / Rekenaarstelsels 245

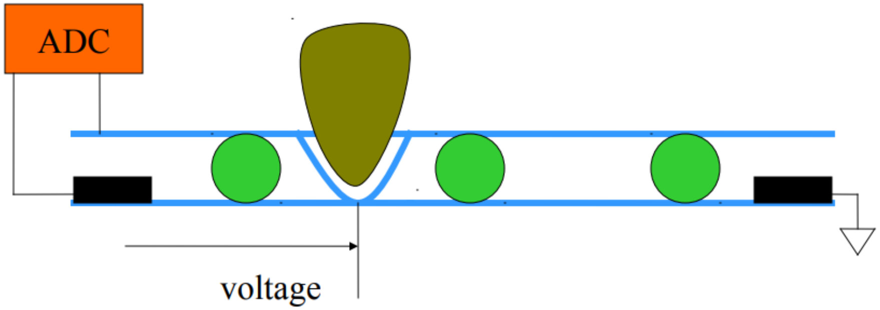
Lecture 23

Analog-to-Digital Converter (ADC)/ Analoog-na-Digitaal Omsetter

Dr Rensu Theart & Dr Lourens Visagie

Analog Signals
Analoog seine

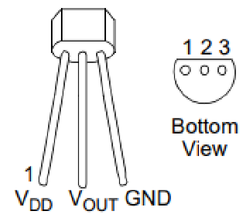
- Quite often the microcontroller has to “read” an analog voltage signal (and not interpret it as ‘0’ or ‘1’)
- Almost all sensing units produce an analog voltage as output
  - Temperature from a thermocouple
  - Magnetic field sensor (electronic compass)
  - MEMS accelerometer
  - MEMS rate sensor
  - Power/current sensor
  - Touchscreen position sensing
- Sensors that produce a digital output (I2C, or SPI) already performs the conversion to digital value inside the device.



Analog Signals
Analoog seine

- **Example**: MCP9700 temperature sensor
- 3 pins: Supply (V_{DD}), GND and V_{OUT}
- V_{OUT} voltage varies proportional to ambient temperature, T_A
- If the temperature is 20°C, V_{OUT} is expected to be 0.7V (20°C x 0.01V/°C + 0.5V)
- Possible microcontroller applications:
  - Display temperature on LCD screen
  - Turn on a heating element if it gets too cold
  - Turn on a fan if it gets too hot
  - Log temperature measurements to SD card
- If the microcontroller can sense the analog voltage, it can calculate the temperature
- How does the microcontroller sense the analog voltage?

3-Pin TO-92
MCP9700/9700A
MCP9701/9701A



MICROCHIP

MCP9700/9700A
MCP9701/9701A

Low-Power Linear Active Thermistor ICs

Sensor Output						
Output Voltage, T_A = 0°C	$V_{0°C}$	—	500	—	mV	**MCP9700/9700A**
Output Voltage, T_A = 0°C	$V_{0°C}$	—	400	—	mV	**MCP9701/9701A**
Temperature Coefficient	T_C	—	10.0	—	mV/°C	**MCP9700/9700A**
	T_C	—	19.5	—	mV/°C	**MCP9701/9701A**

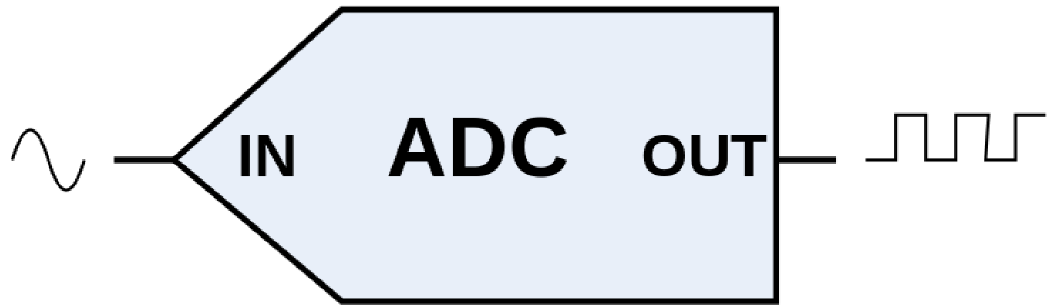
EQUATION 4-1: SENSOR TRANSFER FUNCTION

$$V_{OUT} = T_C \times T_A + V_{0°C}$$

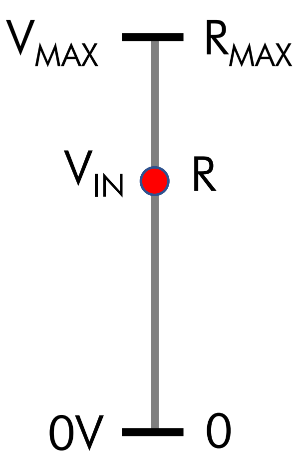
Where:

T_A = Ambient Temperature
V_{OUT} = Sensor Output Voltage
$V_{0°C}$ = Sensor Output Voltage at 0°C (see **DC Electrical Characteristics** table)
T_C = Temperature Coefficient (see **DC Electrical Characteristics** table)

Analog-to-Digital Convertor (ADC)
Analoog-na-Digitaal Omsetter

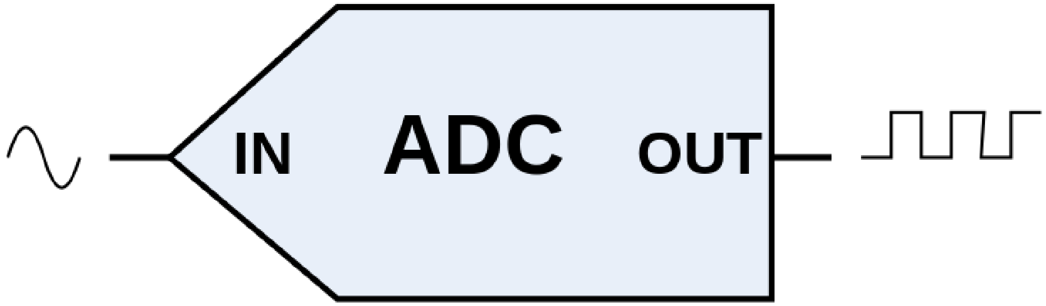


- Analog-to-Digital Converter (ADC) converts analog voltage on input pin to a number that can be represented by computer (digital value)
- The ADC has a linear relationship between converted number and input voltage

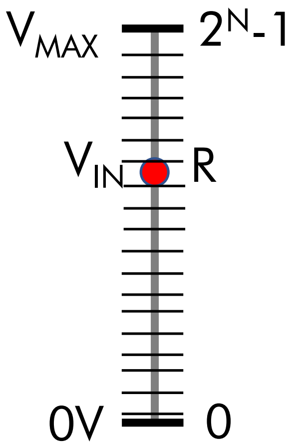


$$R = R_{MAX}\frac{V_{IN}}{V_{MAX}}$$

Analog-to-Digital Convertor (ADC)
Analoog-na-Digitaal Omsetter

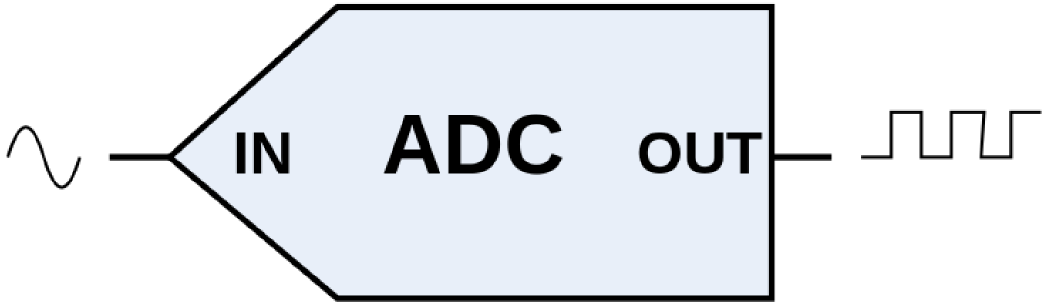


- The sampled ADC value (R) is an integer number, stored in N bits
- The analog voltage equivalent digital value is truncated to integer value – this is called **quantization**

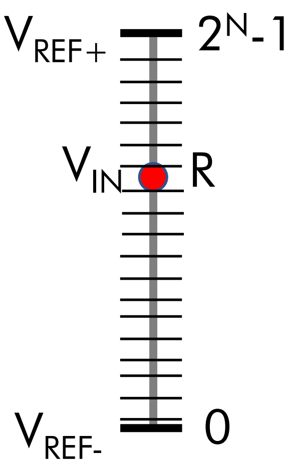


$$R = (2^N - 1)\frac{V_{IN}}{V_{MAX}}$$

Analog-to-Digital Convertor (ADC)
Analoog-na-Digitaal Omsetter



- In the general scenario, the ADC uses a lower and upper reference voltage, V_{REF-} and V_{REF+} (and not 0V and V_{MAX})
- Input ADC voltage (signal to convert) has to be between V_{REF-} and V_{REF+}

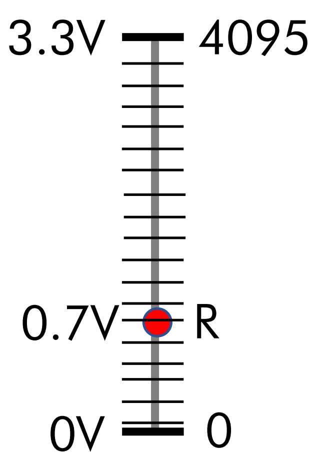


$$R = (2^N - 1)\frac{(V_{IN} - V_{REF-})}{(V_{REF+} - V_{REF-})}$$

ADC conversion examples
ADC omskakeling voorbeelde

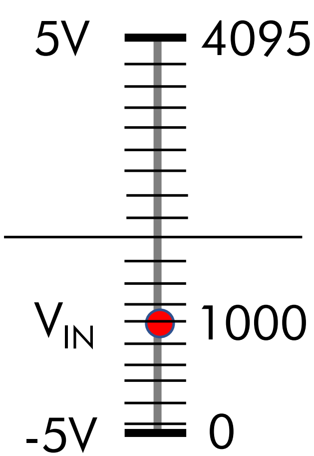
- V_{IN} = 0.7V
- V_{REF-} = 0V
- V_{REF+} = 3.3V
- 12-bit ADC (N = 12):

- R = 4095*0.7/3.3
  = 868



- ADC output is 1000
- V_{REF-} = -5V
- V_{REF+} = 5V
- 12-bit ADC (N = 12)
- What is the input voltage?

- V_{IN} = 1000 * (5V – (-5V) / 4095 -5V
  = -2.56V



ADC Errors
ADC Foute

- The number of bits used to represent the digital sampled value is called the ADC resolution (i.e. 12-bit resolution)
- Due to the **quantization error**, the smallest change in the ADC output (one LSB) corresponds to $(V_{REF+} - V_{REF-})/(2^N-1)$. The ADC cannot discern between finer input voltages
- Example: 12-bit ADC, Voltage range 0V to 5V, precision is 1.22mV/LSB. The ADC cannot measure "finer" than this.

- Another source of ADC error is **non-linearity**
- Ideal ADC is perfectly linear, but in practice response might deviate slightly
- Non-linearity (linearity error) is usually also specified in LSB units

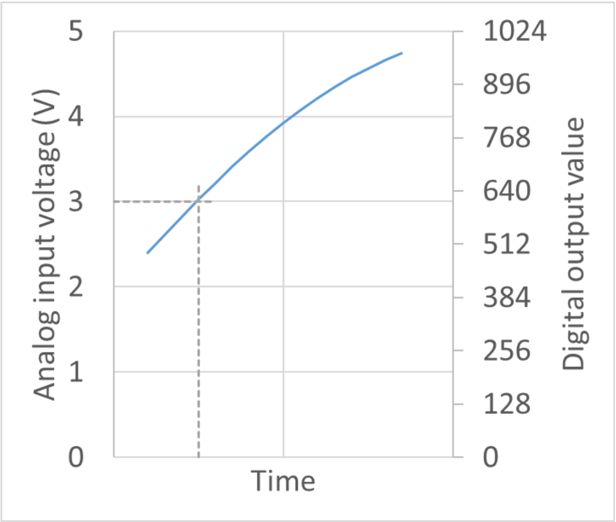
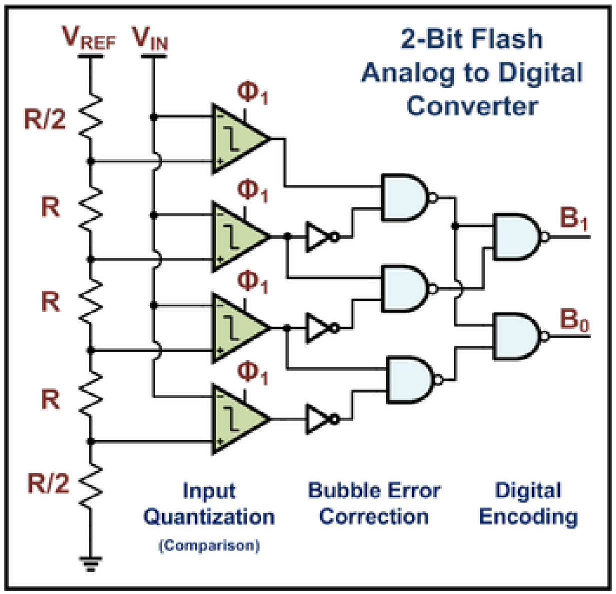


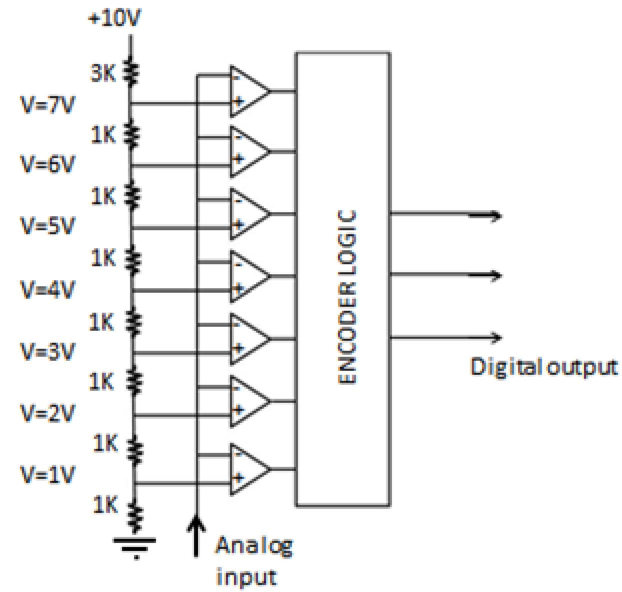
Table 66. ADC accuracy at f_{ADC} = 18 MHz(1)

Symbol	Parameter	Test conditions	Typ	Max(2)	Unit
ET	Total unadjusted error	f_{ADC} = 18 MHz V_{DDA} = 1.7 to 3.6 V V_{REF} = 1.7 to 3.6 V $V_{DDA} - V_{REF} < 1.2$ V	±3	±4	LSB
EO	Offset error		±2	±3	
EG	Gain error		±1	±3	
ED	Differential linearity error		±1	±2	
EL	Integral linearity error		±2	±3	

Types of ADC
Tipes A-na-D omsetters

<u>Direct-conversion ADC or Flash ADC</u>



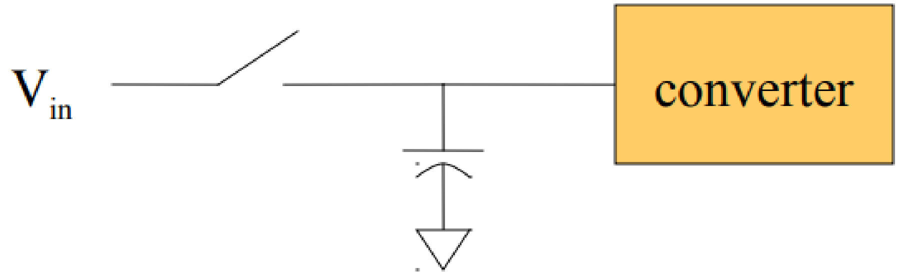


- Array of comparators (one for each digital output level) compares analog input signal to a reference voltage (through resistor divider network)
- Comparator array feeds encoder logic, which converts voltage range to binary “code” (the binary equivalent of the digitally sample value)
- Requires 2^n comparators for a n-bit binary output number
- Pros: Fast conversions. Conversion time = propagation delay through encoder logic.
- Cons: “Large” devices, with lots of logic elements. Also uses lots of power

Types of ADC
Tipes A-na-D omsetters

<u>Integrating conversion (or dual-slope conversion)</u>

- Use analog input voltage to charge a capacitor (run-up period)
  - Switch closes and capacitor charges up
- Allow capacitor to discharge through resistor (run-down period)
  - Switch opens, and capacitor discharges
- Count number of clock cycles until voltage reaches zero → this is the output of the ADC
- For fixed ADC clock, smaller load resistor results in slower discharge, but higher ADC resolution. Larger load resistor, quicker sample time, but lower resolution

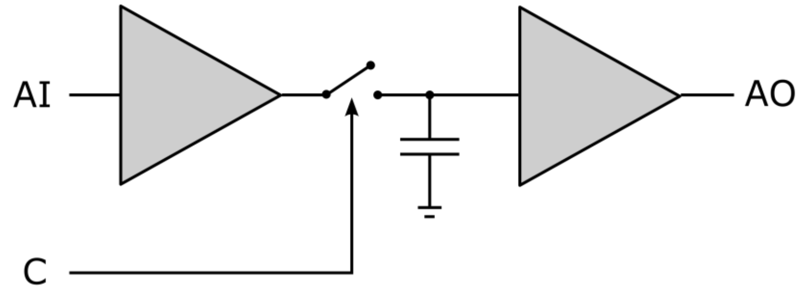


- Pros: Requires few components, with low power requirements. Small non-linearity error
- Cons: Takes longer to sample

Types of ADC
Tipes A-na-D omsetters

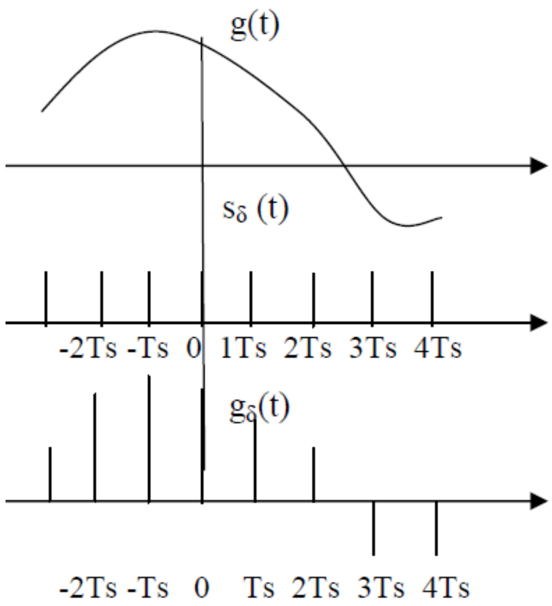
<u>Sample-and-hold</u>

- Since ADC conversion takes time, it is important that the analog signal remains constant for the duration of the conversion
- This is accomplished using a “sample-and-hold” circuit
- Typical implementation uses a switch (FET), capacitor and unity-gain op-amp (buffer)
- <u>Sample</u>: Switch closes to charge capacitor to input voltage (AI)
- <u>Hold</u>: Switch opens, and output buffer ensures AO is the same as voltage across the capacitor

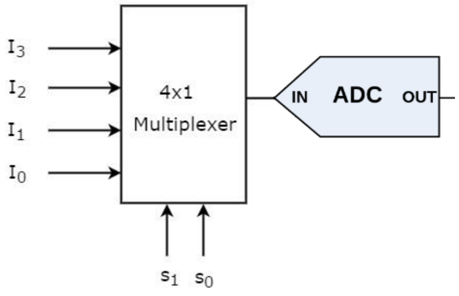


One-shot vs. Continuous ADC sampling
Enkel en Kontinue ADC monsterneming

- Sometimes only single conversions needed (get current temperature)
- Continuous conversion (audio, signal processing etc.)
  - Discrete time sampling of a continuous signal
  - At fixed sampling rate
  - (Nyquist and all that)
  - Timer module provides clock input for continuous conversion
  - Typically used with DMA, to store sampled data in memory array



- One ADC can usually sample multiple analog channels, by making use of a multiplexer
- Only one channel is sampled at a time, but multiplexer can switch between channels



ADC Specification
ADC Spesifikasies

- Typical specifications (on a datasheet):
  - The resolution or number of bits
  - Maximum conversion rate (Data rate, in samples per second or SPS)
  - Non-linearity (LSB units)

Texas Instruments

ADS1113, ADS1114, ADS1115

www.ti.com SBAS444D – MAY 2009 – REVISED JANUARY 2018

SYSTEM PERFORMANCE				
	Resolution (no missing codes)		16	Bits
DR	Data rate		8, 16, 32, 64, 128, 250, 475, 860	SPS
	Data rate variation	All data rates	–10% 10%	
	Output noise		See *Noise Performance* section	
INL	Integral nonlinearity	DR = 8 SPS, FSR = ±2.048 V(2)	1	LSB

- High-speed ADCs: sampling rates in excess of 10 million samples/s
- High-precision ADCs: >16-bit resolution

STM32F411 ADC

- The STM32F411VE ADC:
  - A single 12-bit ADC, with 19 multiplexed channels
  - Successive approximation implementation
  - Select between 12, 10, 8 or 6-bit resolution
  - Multiplexer channels: 16 external sources, internal temperature sensor and Vbat channel
  - Single or continuous operating mode
  - Ability to automatically “scan” a sequence of channels
  - Selection of trigger sources (timer events, external sources)
  - Can generate DMA requests after conversion completes

STM32F411 ADC

- Pins

Table 39. ADC pins

Name	Signal type	Remarks
V_{REF+}	Input, analog reference positive	The higher/positive reference voltage for the ADC, $1.8\ V \leq V_{REF+} \leq V_{DDA}$
V_{DDA}	Input, analog supply	Analog power supply equal to V_{DD} and $2.4\ V \leq V_{DDA} \leq V_{DD}$ (3.6 V) for full speed $1.8\ V \leq V_{DDA} \leq V_{DD}$ (3.6 V) for reduced speed
V_{REF-}	Input, analog reference negative	The lower/negative reference voltage for the ADC, $V_{REF-} = V_{SSA}$
V_{SSA}	Input, analog supply ground	Ground for analog power supply equal to V_{SS}
ADCx_IN[15:0]	Analog input signals	16 analog input channels

ADC channel	MCU pin	ADC channel	MCU pin
0	PA0	8	PB0
1	PA1	9	PB1
2	PA2	10	PC0
3	PA3	11	PC1
4	PA4	12	PC2
5	PA5	13	PC3
6	PA6	14	PC4
7	PA7	15	PC5

STM32F411 ADC

- Clock: ADC1 clock comes from APB2 clock. Can be divided by 2, 4, 6 or 8 – setting in the ADC registers
- Sample time is determined by ADC clock frequency. Selection bits in ADC registers determine variable number of clock cycles for conversion (+ fixed number of cycles, depending on selected resolution)
- Minimum sample times:
  - 12 bits: 3 + 12 = 15 ADCCLK cycles
  - 10 bits: 3 + 10 = 13 ADCCLK cycles
  - 8 bits: 3 + 8 = 11 ADCCLK cycles
  - 6 bits: 3 + 6 = 9 ADCCLK cycles

STM32F411 ADC

- <u>Channel selection</u>
- The STM32F4 ADC does not have a single channel selection, but a sequence of up to 16 channel selections

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								L[3:0]				SQ16[4:1]			
								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SQ16_0	SQ15[4:0]					SQ14[4:0]					SQ13[4:0]				
rw	rw	rw	rw	rw	rw	rw	rw				rw	rw	rw	rw	rw

Bits 31:24 Reserved, must be kept at reset value.

Bits 23:20 **L[3:0]:** Regular channel sequence length

These bits are written by software to define the total number of conversions in the regular channel conversion sequence.

0000: 1 conversion
0001: 2 conversions
...
1111: 16 conversions

Bits 19:15 **SQ16[4:0]:** 16th conversion in regular sequence

These bits are written by software with the channel number (0..18) assigned as the 16th in the conversion sequence.

- If SCAN setting (ADC control register) is 0, only the first selection in the sequence registers are used – single conversion. If SCAN = 1, the entire sequence is used.
- Only one data register (DR). So if you use scan mode, you will probably also use DMA to make sure sampled data is not overwritten

STM32F411 ADC

- Single/Continuous conversion
- If CONT setting in control register is set to 1, another conversion is launched immediately after the previous one finished
- (If SCAN=1, another sequence of conversions are started automatically)
- Continuous mode will convert all the time, with no delay – as fast as ADC can sample
- Useful if you don't want your program to wait for a result – just read the last value that was converted from the data register.
- (It is also possible to perform an "injected" conversion – while continuously converting, interrupt the normal sequence and have another group of channels that are sampled once.)
- When not using continuous mode, conversion can be started by software (writing bit in ADC control register), or triggered by timer event (Timer update/overflow event (TRGO), or channel compare event (CHx)
- Using timer is useful if you want to sample the signal at specific data rate. Also likely you will use DMA for this.

STM32F411 ADC

- <u>Modes and scenarios</u>

	Single channel infrequent	Multiple channels infrequent	Single channel continuous	Multiple channels continuous	Single/multiple channels timed
Example scenario	Read temperature sensor once a second	Read two temperature sensors once a second	Want the most recent conversion available immediately – no waiting	Can't think of an example ☹	Need to sample (a) signal(s) at X Hz
Use continuous mode	No	No	Yes	Yes	No
Use scan mode	No	No	No	Yes	Yes/No
Use DMA	No	No	Possibly?	Possibly?	Yes
Triggered by timer event	No	No	No	No	Yes