



# ADC et DAC

Conversion analogique-numérique et  
conversion numérique-analogique



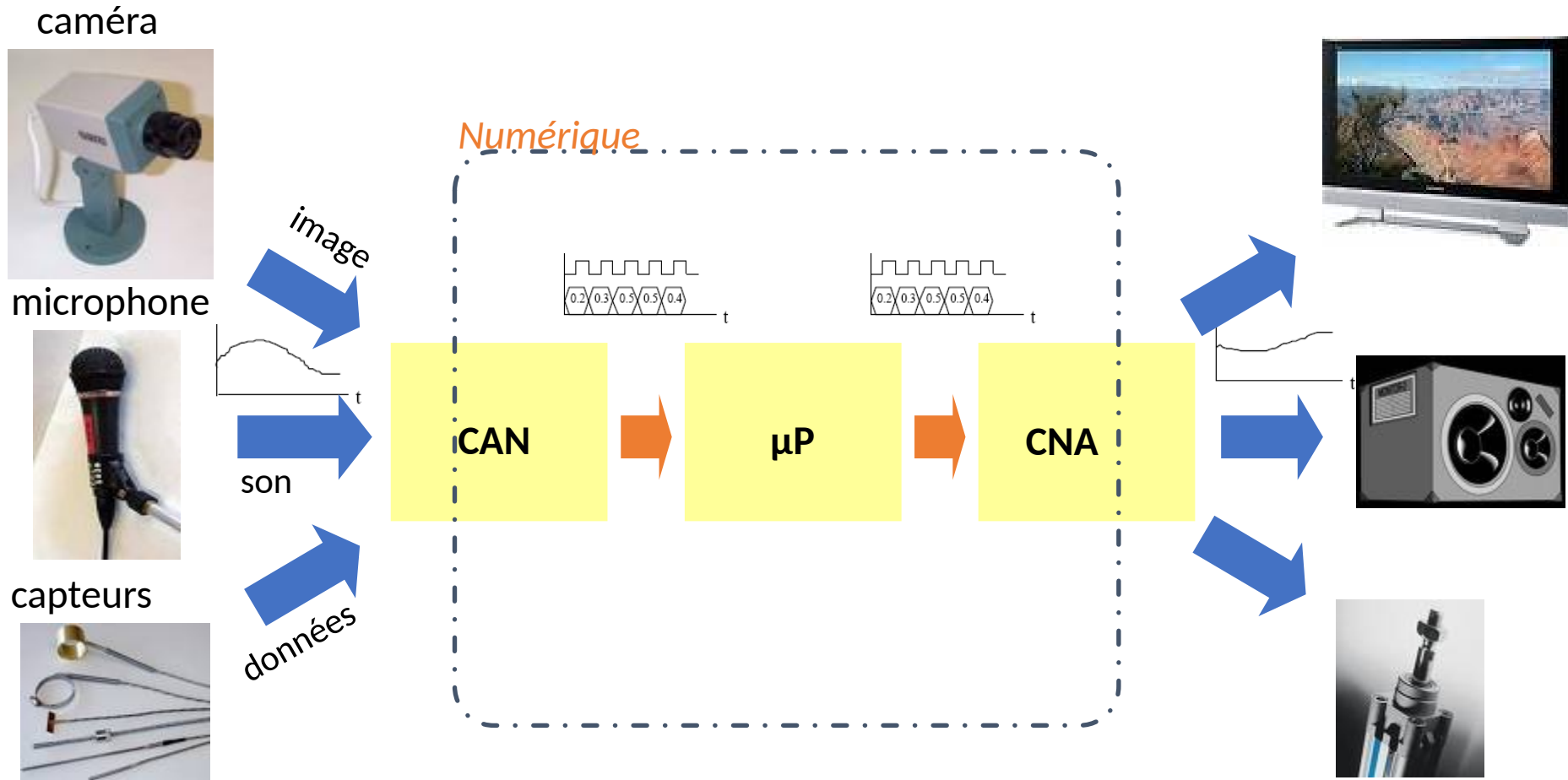
- ❑ **ADC/DAC – Conversions**
  - ❑ Fonctionnalité
  - ❑ Registres de configuration
  - ❑ Exemple de programmation

# Convertisseur Analogique Numérique



# Traitement du signal

## □ Synoptique d'une chaîne de traitement du signal

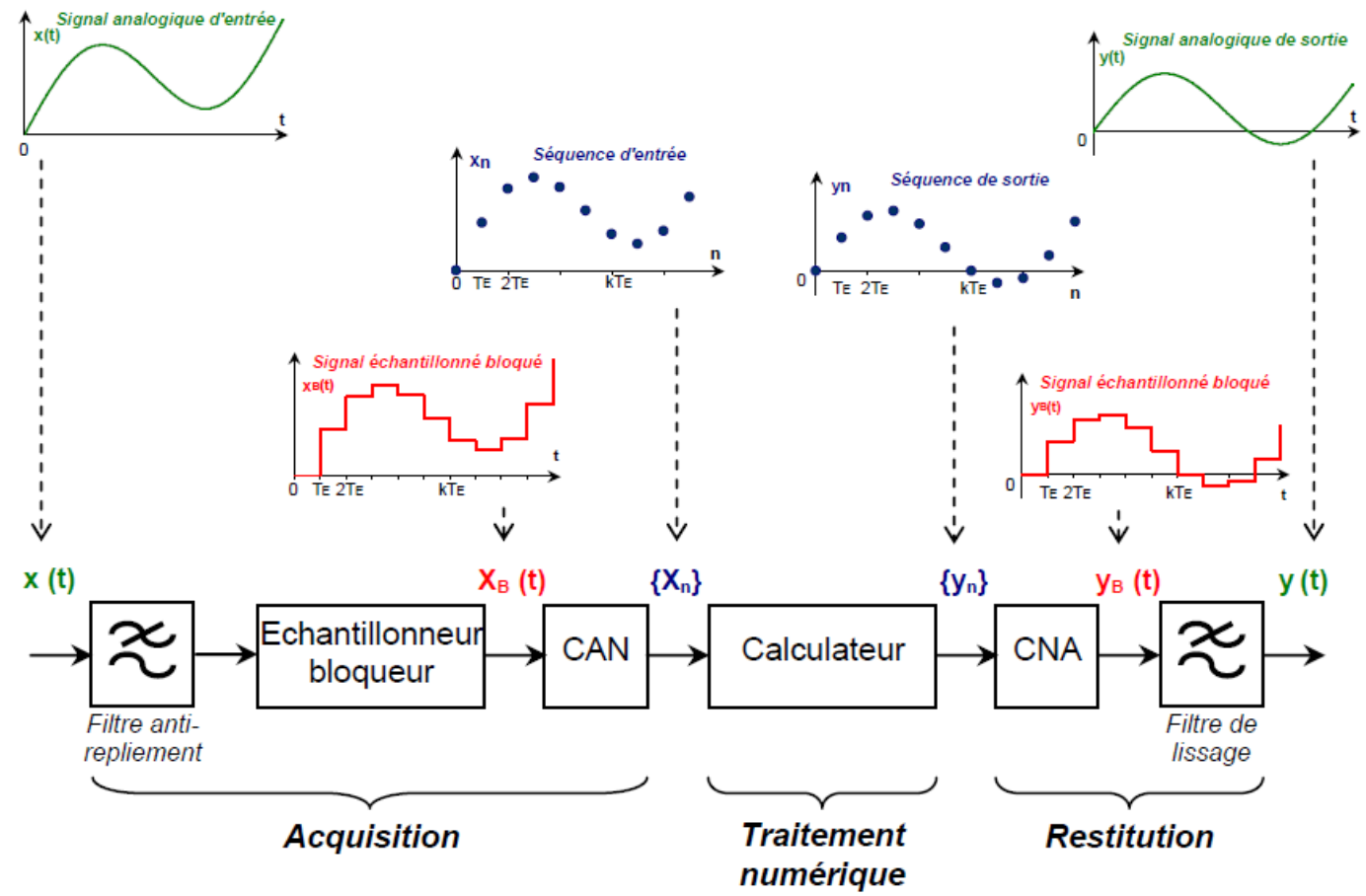


# Traitement du signal

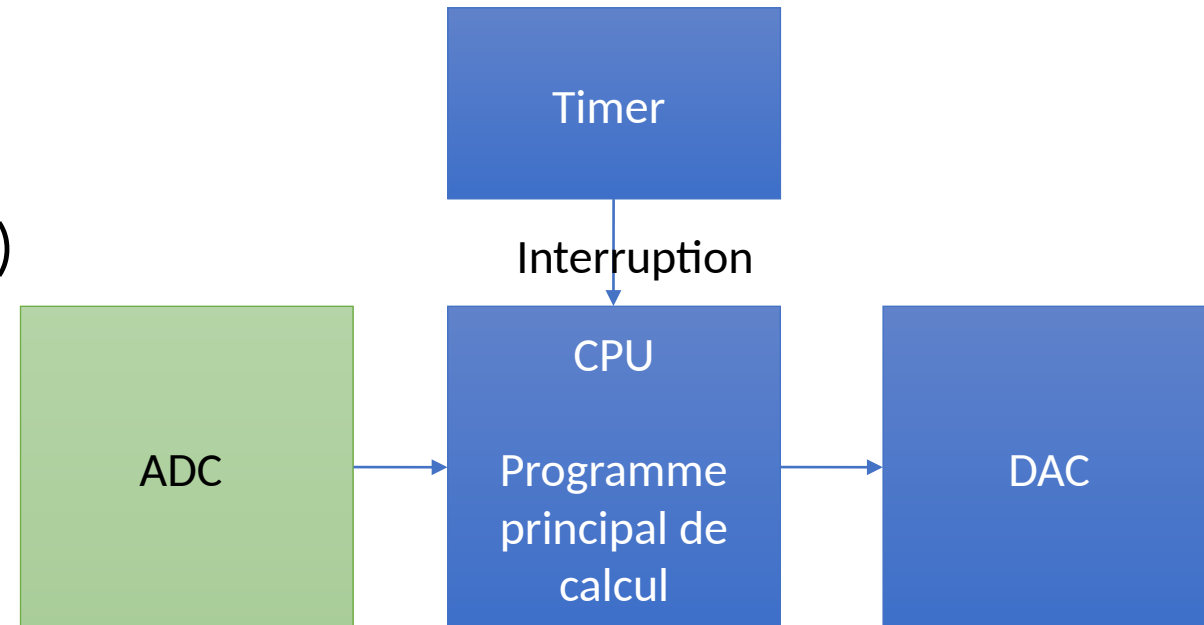
□ Traitement à intervalles temporels réguliers des données issues d'un ADC

□ Echantillonnage temporel des données

□ Nécessite une gestion précise temporelle de l'acquisition

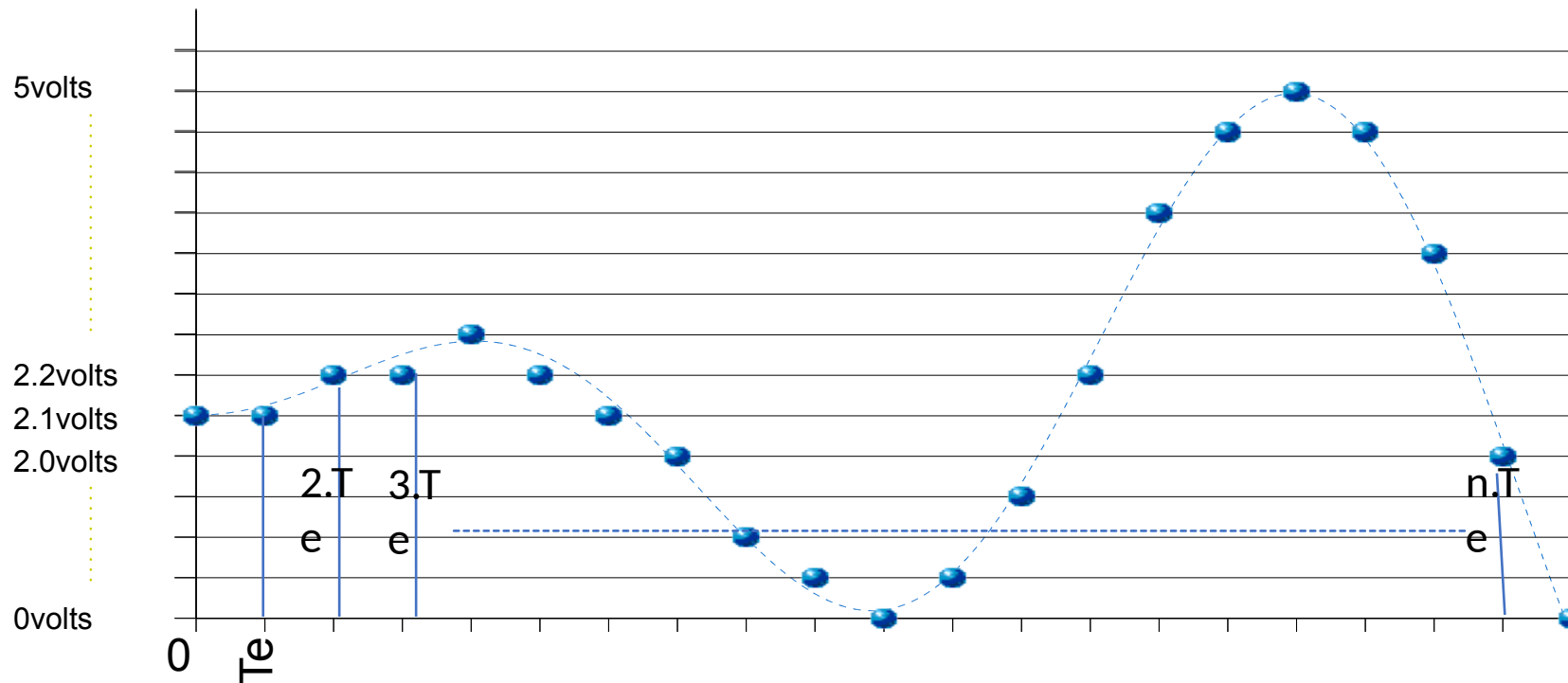


- ❑ Utilisation d'un périphérique dédié à l'acquisition du signal analogique :
  - ❑ ADC (Analog to Digital Converter)
  - ❑ CAN (Convertisseur Analogique Numérique)
- ❑ Remarque : Ne pas oublier d'utiliser un filtre anti-repliement avant acquisition



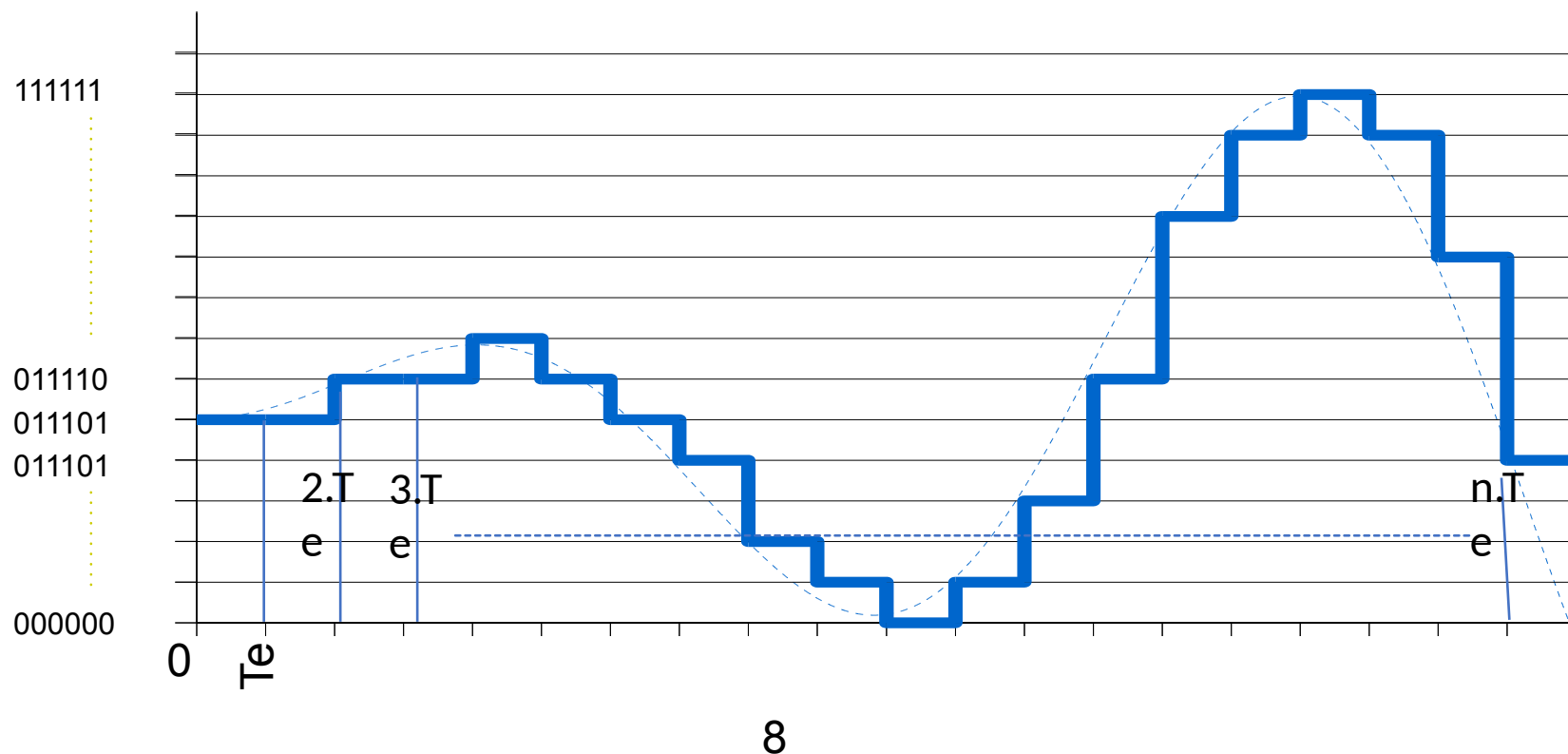
# Convertisseur Analogique Numérique

L'échantillonnage consiste à représenter un signal analogique continu  $s(t)$  par un ensemble de valeurs discrètes  $s(nT_e)$  avec  $n$  entier et  $T_e$  constant appelé **période d'échantillonnage**



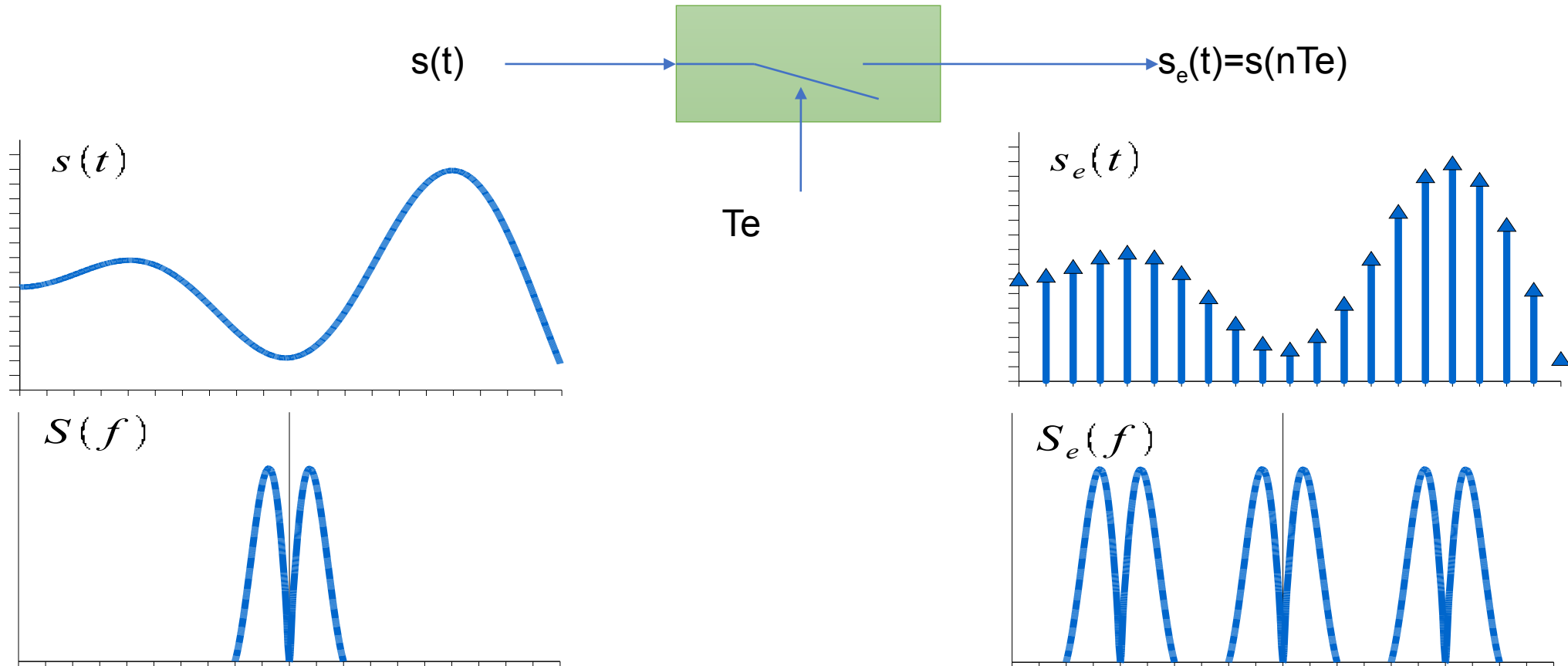
# Convertisseur Analogique Numérique

La numérisation d'un signal consiste en l'échantillonnage et la quantification des amplitudes du signal échantillonné. Le signal numérisé est discrétisé en temps et en amplitude. Il correspond à un signal numérique (binaire) qui évolue au rythme de  **$T_e$**





## Echantillonnage idéal

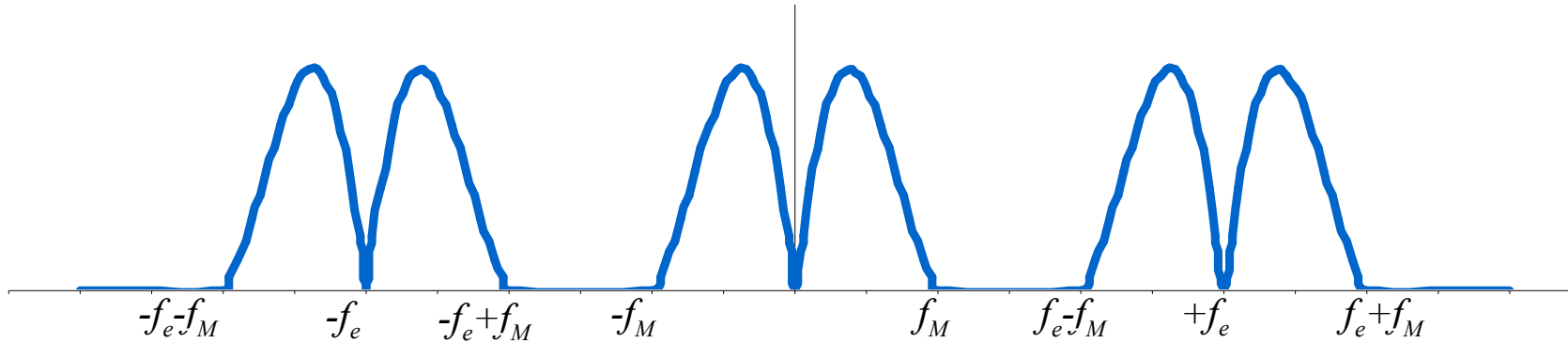


Périodisation du spectre

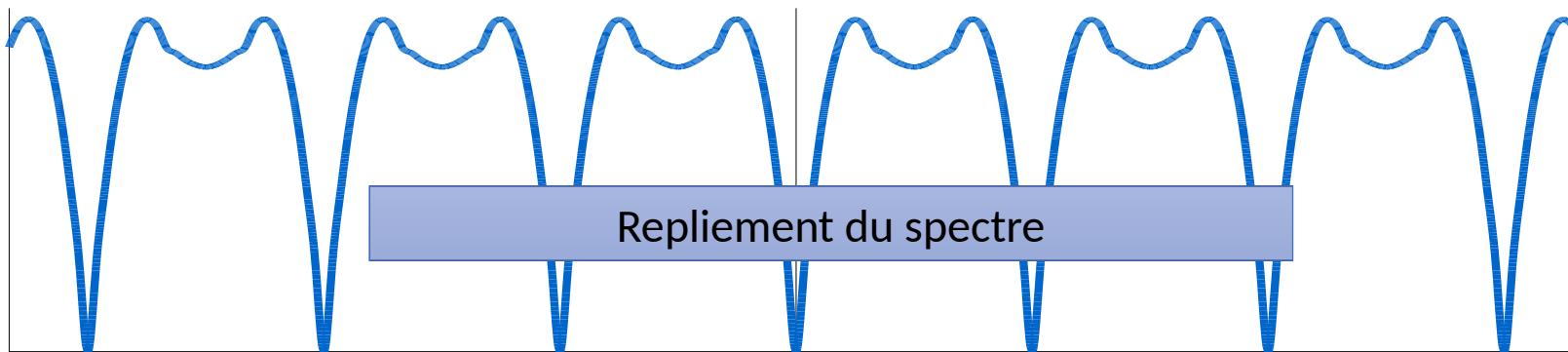


# Convertisseur Analogique Numérique

## Comment choisir $f_e$ ?



$$f_e > 2 \cdot f_M$$



$$f_e < 2 \cdot f_M$$

# Convertisseur Analogique Numérique

- ❑ Fréquence d'échantillonnage trop basse, le spectre du signal échantillonné est replié sur lui même

## ❑ Théorème de Shannon/Nyquist:

- ❑ La fréquence d'échantillonnage doit être 2 fois supérieure à la bande B occupée par le signal original

- ❑  $f_e = 2 \cdot B$  représente la fréquence critique d'échantillonnage ou Nyquist
- ❑ Si  $f_e > B \Rightarrow$  sur-échantillonnage
- ❑ Si  $f_e < B \Rightarrow$  sous-échantillonnage

$$f_e \geq 2 \cdot B$$

- ❑ Si les conditions sur la fréquence ne sont pas respectées, le spectre du signal échantillonné sera replié.



## Exemple du choix de $f_e$

$$s(t) = \sin(2\pi f_0 t)$$

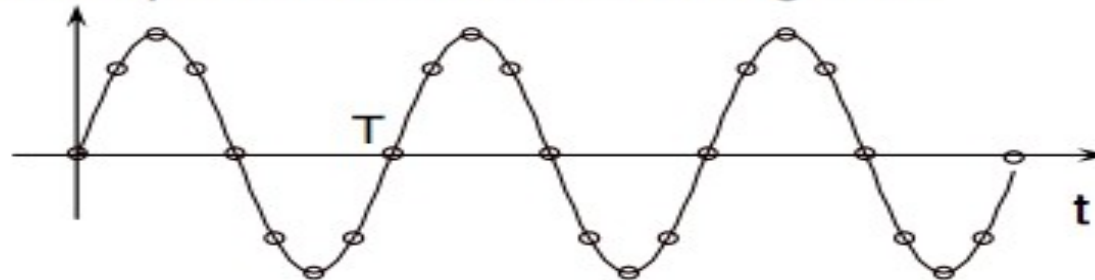
Considérons 3 choix de période d'échantillonnage  $T_e$

1er choix :

$$T = \frac{1}{f_0}$$

$$T_e = \frac{T}{8}$$

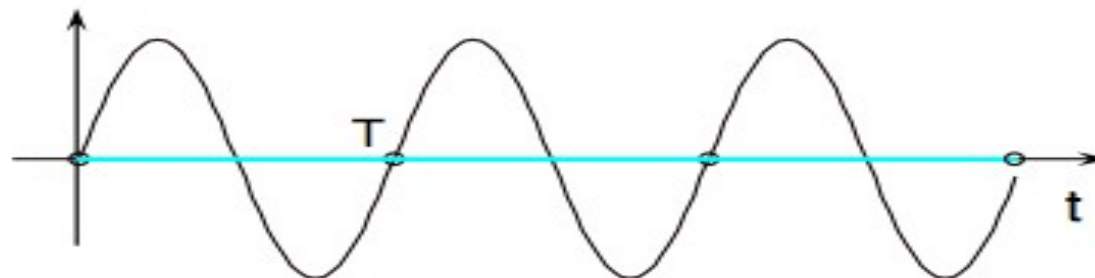
$$f_e = 8 f_0$$



2ème choix :

$$T_e = T$$

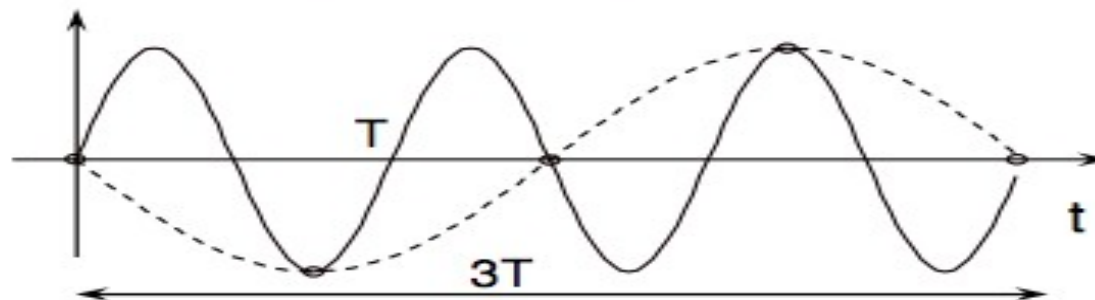
$$f_e = f_0$$



3ème choix :

$$T_e = \frac{3}{4} T$$

$$f_e = \frac{4}{3} f_0$$

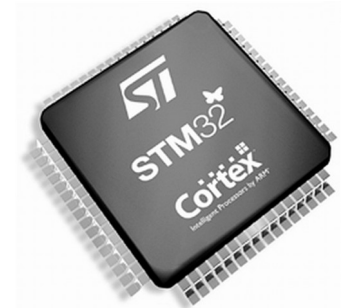
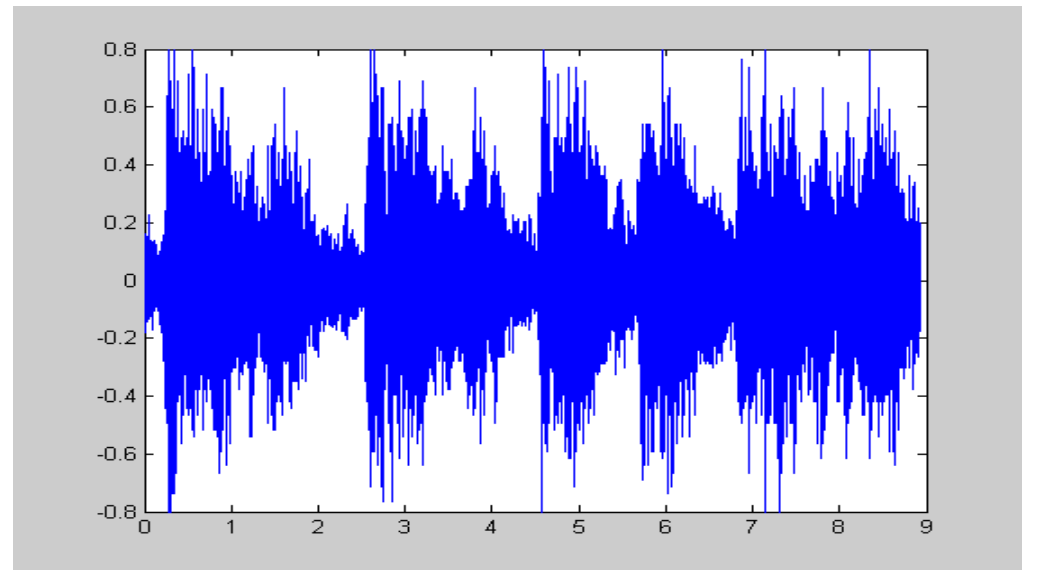
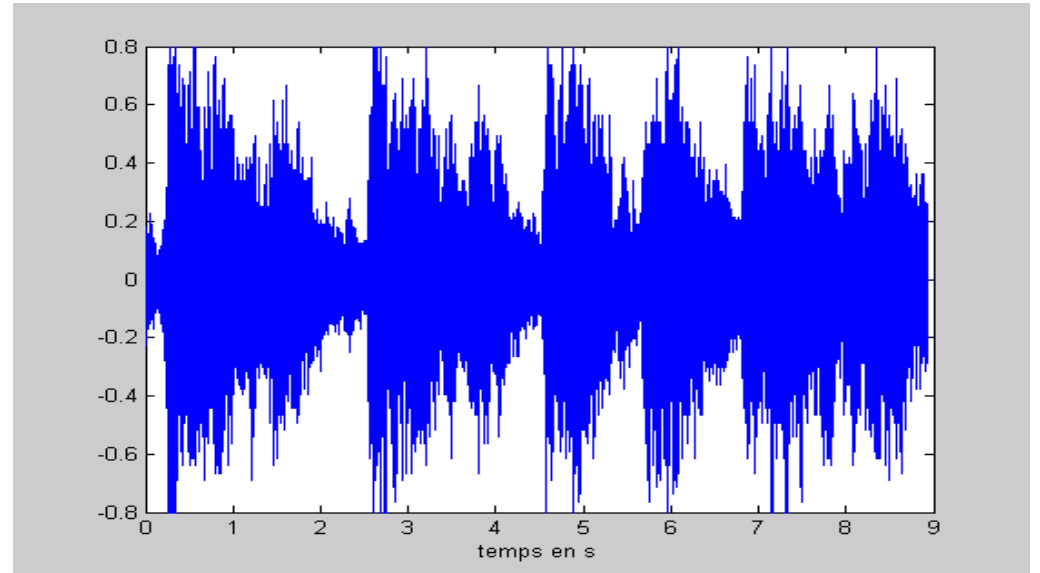


**Repliement spectral**

$$f = f_e - f_0 = \frac{1}{3} f_0$$

## Exemple

- Signal audio bande passante de 3kHz échantillonné à 8kHz
- Signal audio bande passante de 3kHz échantillonné à 1kHz



## Quantification

- $2^N$  valeurs sur la dynamique totale entre  $-A$  et  $+A$

$$A = 2^{n-1} \cdot q$$

- LSB représente la plus petite valeur acquise  $\approx 1$  quantum :  $q$

$$P_c = \frac{(2^{n-1} \cdot q)^2}{2}$$

$$\frac{P_c}{B} = \frac{3}{2} \cdot 2^{2N}$$

- En exprimant la puissance du signal:

$$SNR = \left( \frac{P_c}{B} \right)_{dB} = 10 \cdot \log \left( \frac{3}{2} \cdot 2^{2N} \right) = 6,2 \cdot N + 1,76$$



# Convertisseur Analogique Numérique

## ❑ Résolution

❑ N bits =>  $2^N$  états : exemple 12 bits => 4096 paliers d'acquisition

RESOLUTION N	$2^N$	VOLTAGE (10 V FS)	ppm FS	% FS	dB FS
2-bit	4	2.5 V	250,000	25	-12
4-bit	16	625 mV	62,500	6.25	-24
6-bit	64	156 mV	15,625	1.56	-36
8-bit	256	39.1 mV	3,906	0.39	-48
10-bit	1,024	9.77 mV (10 mV)	977	0.098	-60
12-bit	4,096	2.44 mV	244	0.024	-72
14-bit	16,384	610 $\mu$ V	61	0.0061	-84
16-bit	65,536	153 $\mu$ V	15	0.0015	-96
18-bit	262,144	38 $\mu$ V	4	0.0004	-108
20-bit	1,048,576	9.54 $\mu$ V (10 $\mu$ V)	1	0.0001	-120
22-bit	4,194,304	2.38 $\mu$ V	0.24	0.000024	-132
24-bit	16,777,216	596 nV*	0.06	0.000006	-144

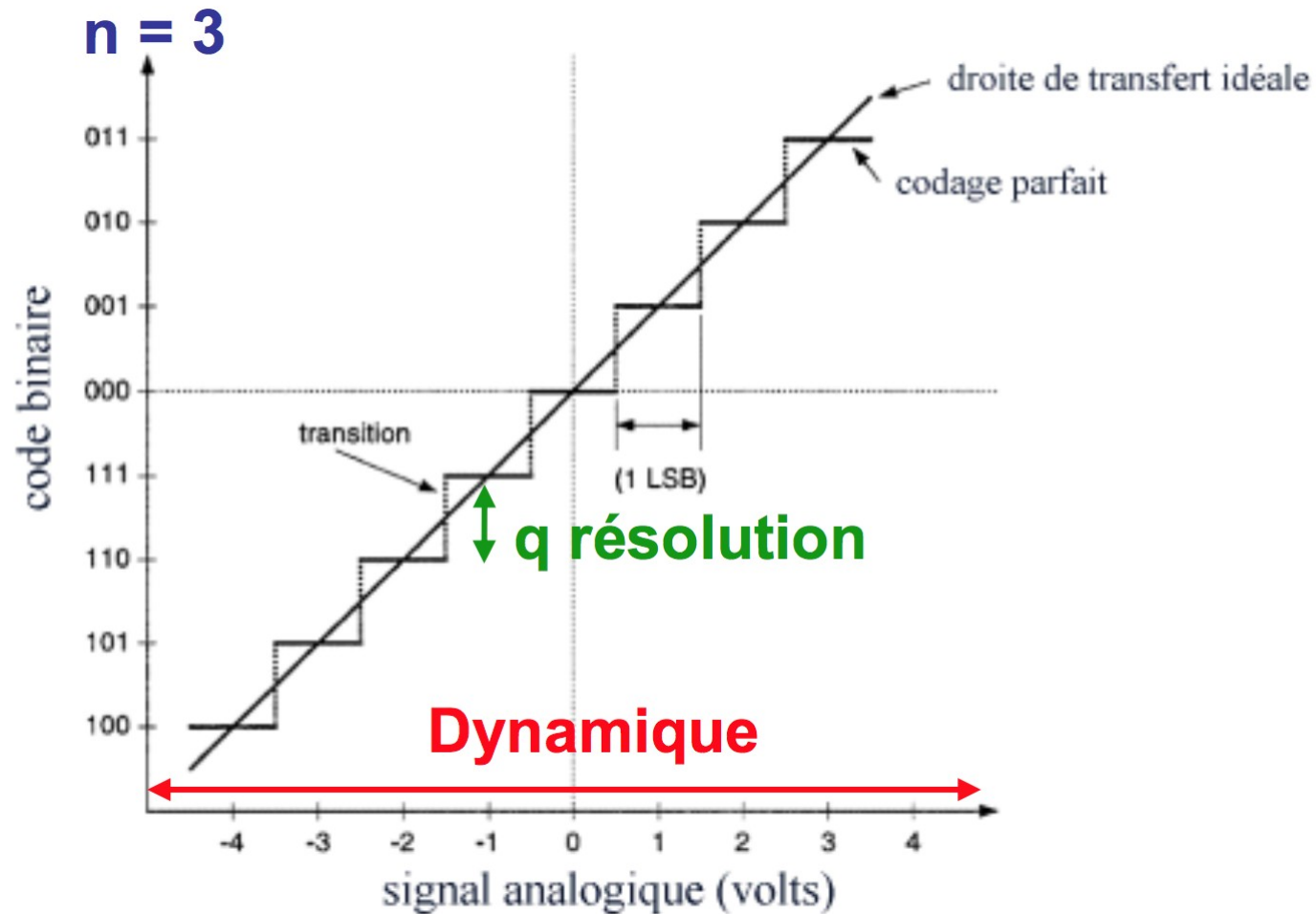
NOTES: \*600 nV is the Johnson Noise in a 10 kHz BW of a 2.2 k $\Omega$  Resistor @ 25°C

10 bits and 10 V FS yields an LSB of 10 mV, 1000 ppm, or 0.1%.  
All other values may be calculated by powers of 2.



# Convertisseur Analogique Numérique

## □ Caractéristique de transfert



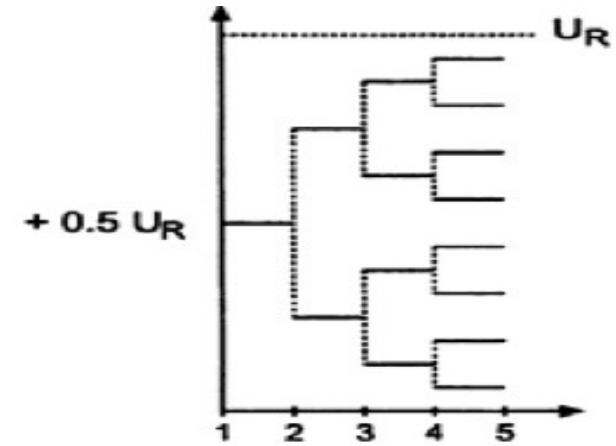
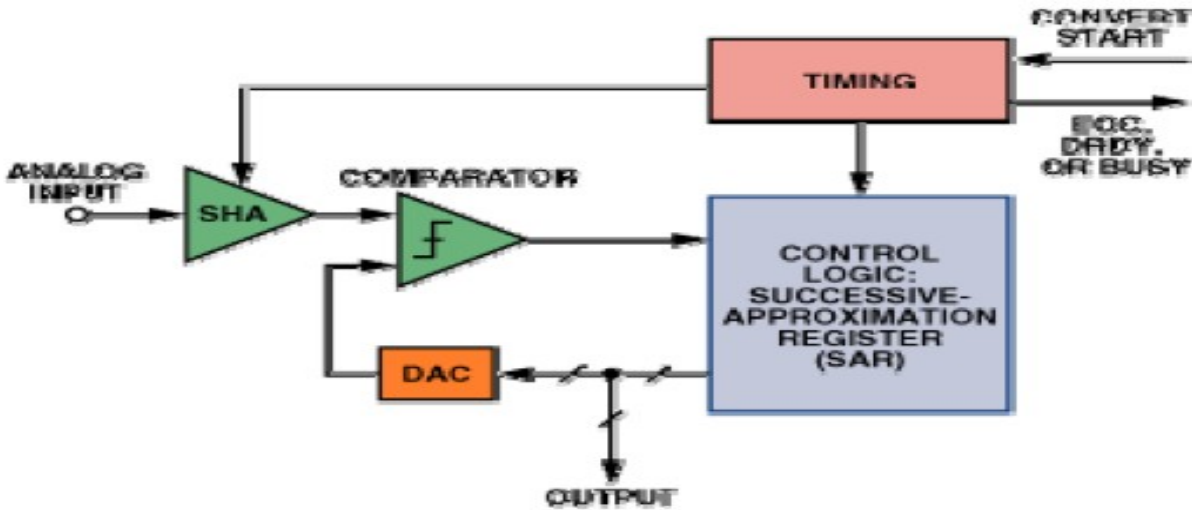




# Convertisseur Analogique Numérique

## ❑ CAN SAR

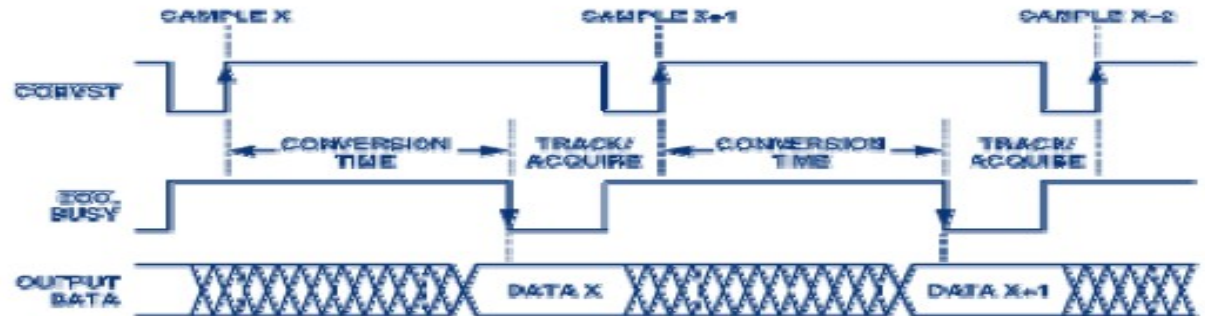
❑ Convertisseur à approximation successives



N cycles pour N bits

~ 16 bits @ ~ 3MHz

Bernard Gordon, at Epsco, introduced the first commercial vacuum-tube SAR ADC in 1954—an 11-bit, 50-kSPS ADC that dissipated 500 watts





# Convertisseur Analogique Numérique

## ☐ Résolution vs Fréquence acquisition

**Analog-to-Digital Converters**  
**Resolution/Throughput-Rate Selection Matrix**

<b>Resolution, Bits</b>	<b>17+</b>	●	●	●	●		
<b>14-16</b>	●	●	●	●	●	●	●
<b>12-13</b>		●	●	●	●	●	●
<b>10-11</b>		●	●	●	●	●	●
<b>8-9</b>			●	●	●	●	●
<b>&lt;8</b>					●		
		<b>&lt;10 kps</b>	<b>10 kps to 100 kps</b>	<b>100 kps to 1 MSPS</b>	<b>1 MSPS to 10 MSPS</b>	<b>10 MSPS to 100 MSPS</b>	<b>100 MSPS +</b>

❑ Exemple de programmation sous mbed

❑ Acquisition sur plusieurs voies

❑ Valeurs renvoyées entre 0 et 1  
❑  $1 \leq 4096$

## ADC

[mBed ADC manual](#)

Public Member Functions	
	<b>AnalogIn</b> (PinName pin) Create an <b>AnalogIn</b> , connected to the specified pin.
float	<b>read</b> () Read the input voltage, represented as a float in the range [0.0, 1.0].
unsigned short	<b>read_u16</b> () Read the input voltage, represented as an unsigned short in the range [0x0, 0xFFFF].
	<b>operator float</b> () An operator shorthand for <b>read()</b>

Define an **Analog Input (ADC)** connected to a specific Pin.

*Declaration:*

```
AnalogIn Name(Pin);
```

The **Pin** must be the name shown in the Blue Labels in the [fig.1](#)  
**Name** is as you want.

*Example*

```
AnalogIn Ana_In1(A0);
```

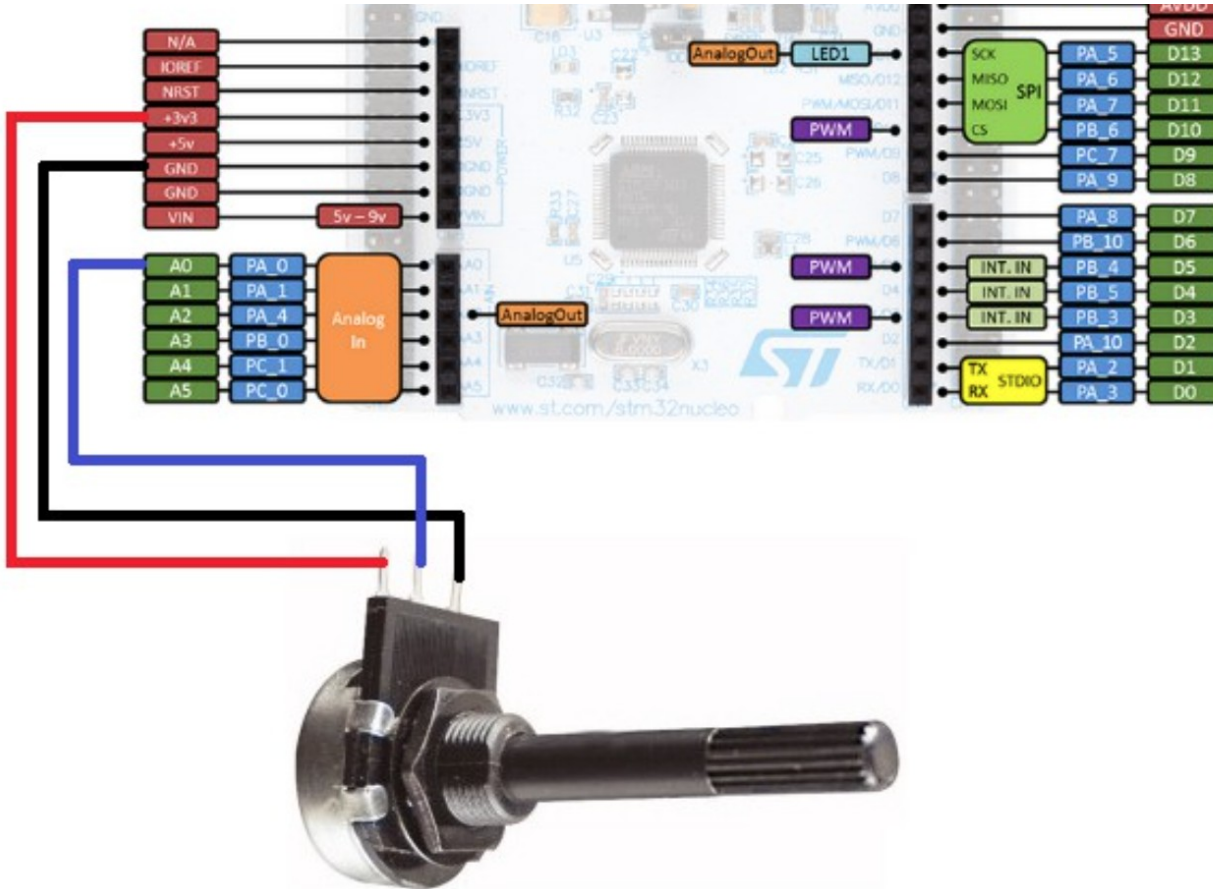
After the declaration of the analog pin you are able to read an analog value.

*Example*

```
unsigned long value = 0;  
value = Ana_In1.read_u16();
```



# Exemple ADC STM32F



```
#include "mbed.h"
```

```
AnalogIn adc_temp(ADC_TEMP);
AnalogIn adc_vref(ADC_VREF);
AnalogIn adc_vbat(ADC_VBAT); // Warning: Not
available on all devices
AnalogIn ain(A0);
```

```
DigitalOut led(LED1);
```

```
int main() {
    printf("\nSTM32 ADC internal channels reading
example\n");
    while(1) {
        printf("ADC Temp = %f\n", adc_temp.read());
        printf("ADC VRef = %f\n", adc_vref.read());
        printf("ADC VBat = %f\n", adc_vbat.read());
        printf("ADC A0 = %f\n", ain.read());

        printf("\033[3A");
        led = !led;
        wait(1.0);
    }
}
```

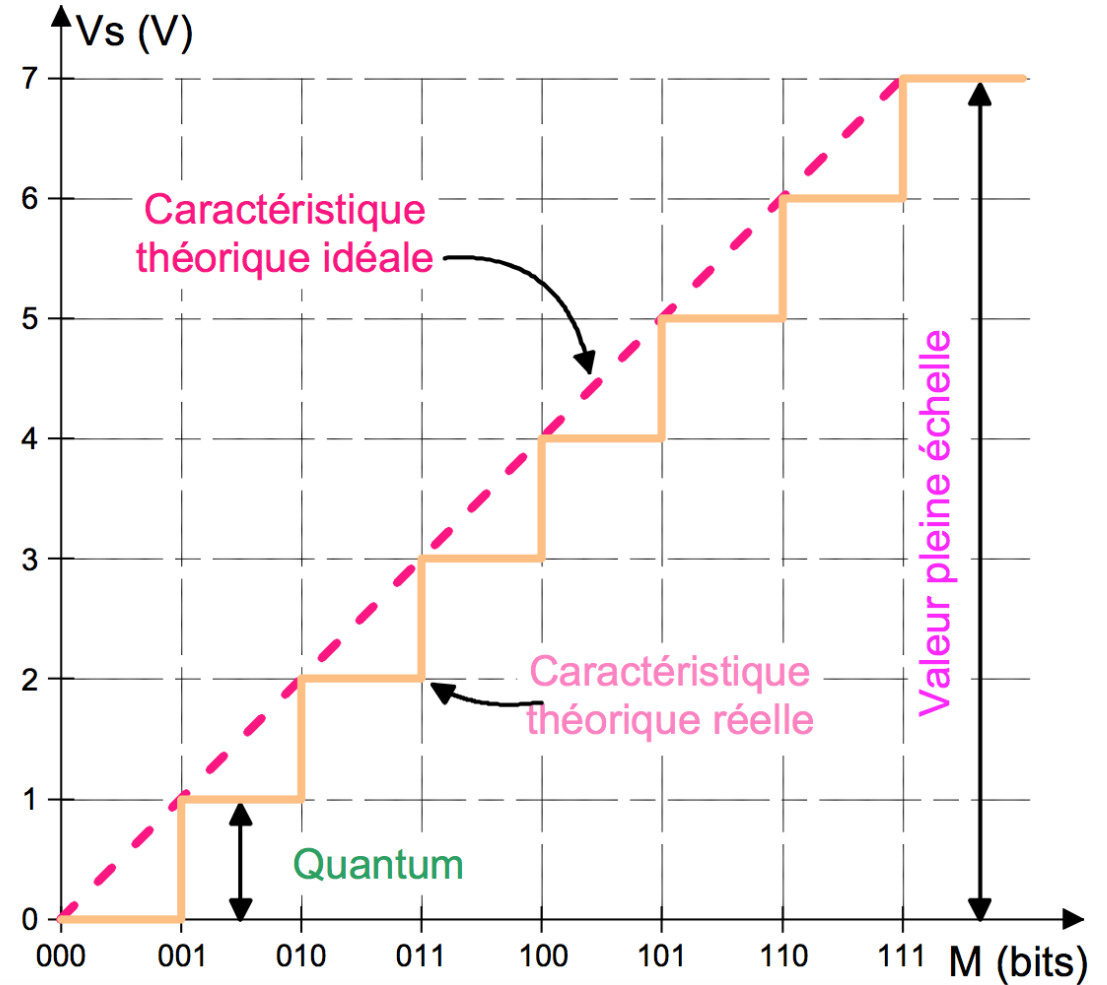
# Convertisseur Numérique Analogique



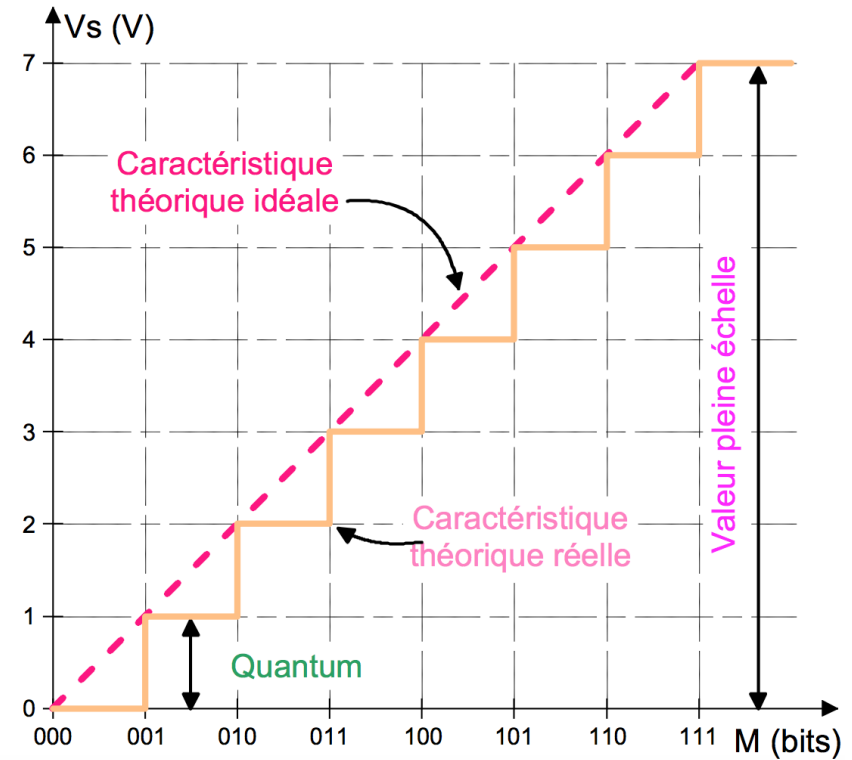
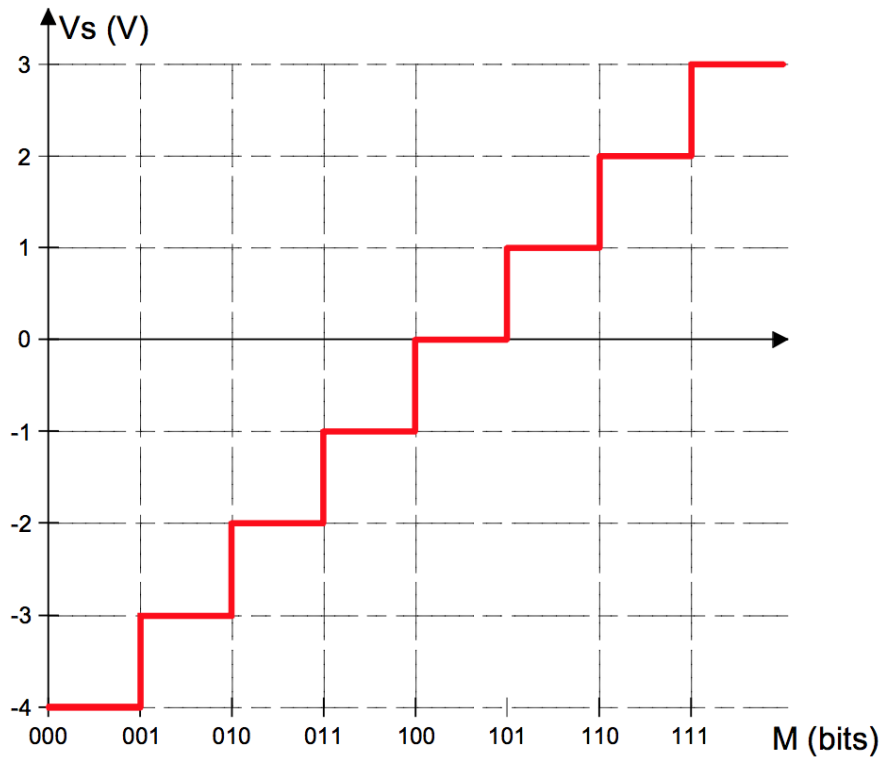
- ❑ Objectif :
  - ❑ Circuit dual du CAN
  - ❑ Convertir une donnée numérique en tension ou courant

❑ Idéalement :

$$V_{\text{pleine échelle}} = V_{\text{max}} - V_{\text{min}} = (2^N - 1) \cdot q$$



- Tension de sortie dépend du format du convertisseur
  - Tension mono-polaire ou bipolaire / format numérique





- ❑ Paramètres d'un CNA : les mêmes que ceux d'un CAN
  - ❑ Dynamique : Variation possible de tension de sortie d'un CNA
  - ❑ Résolution : Plus petite tension manipulable :  $q \Rightarrow$  quantum
  - ❑ Linéarité : Différence entre la sortie théorique et effective [% , mv ou lsb]
  - ❑ Cadence : Vitesse de restitution «Sample Per Seconde» (SPS)
  - ❑ Format : Représentation des données numériques
  - ❑ Temps d'établissement : Temps de conversion
  - ❑ Type de sortie : en tension ou en courant



## Architectures des CNA

### 2 classes de CNA

#### Architecture direct

- Généralement une somme pondérée de courant ou de tension

- Architecture R-2R

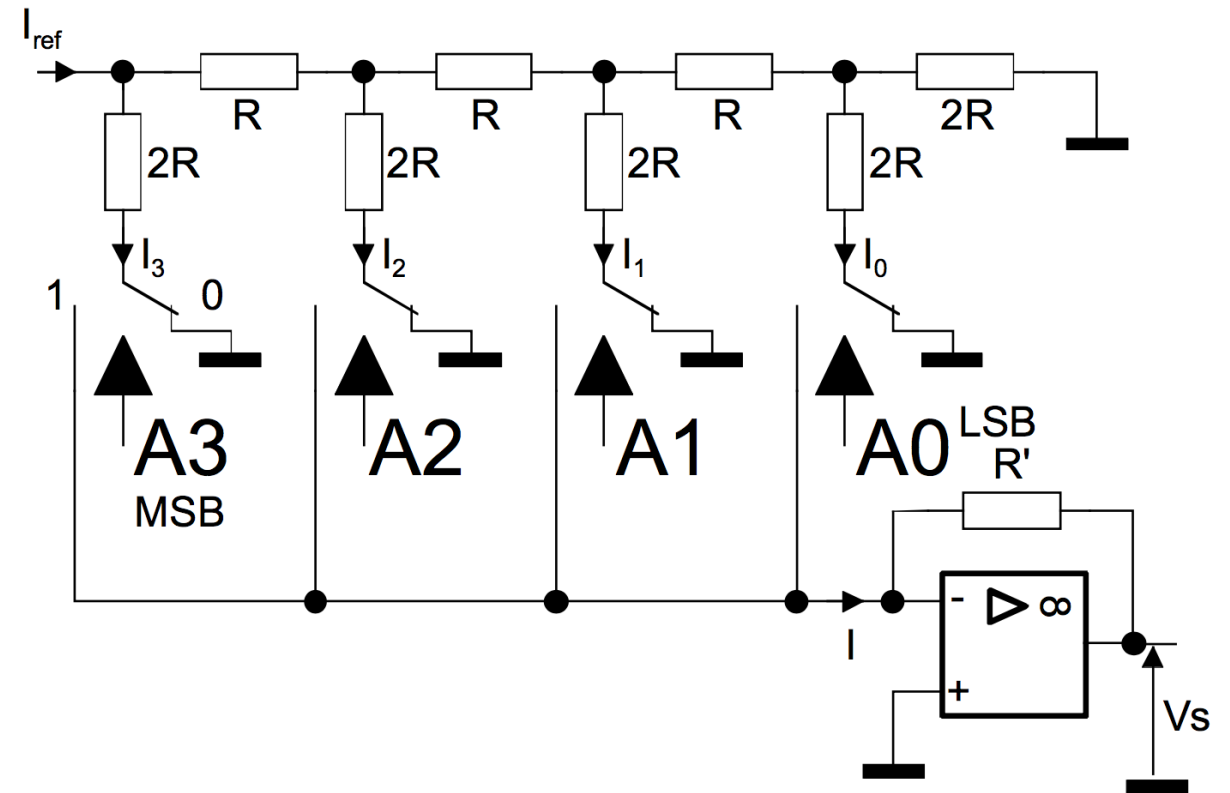
- Architecture potentiométrique

- Architecture à source de courant

#### Architecture indirecte

- MLI + lissage de tension

## Architecture simplifiée R-2R

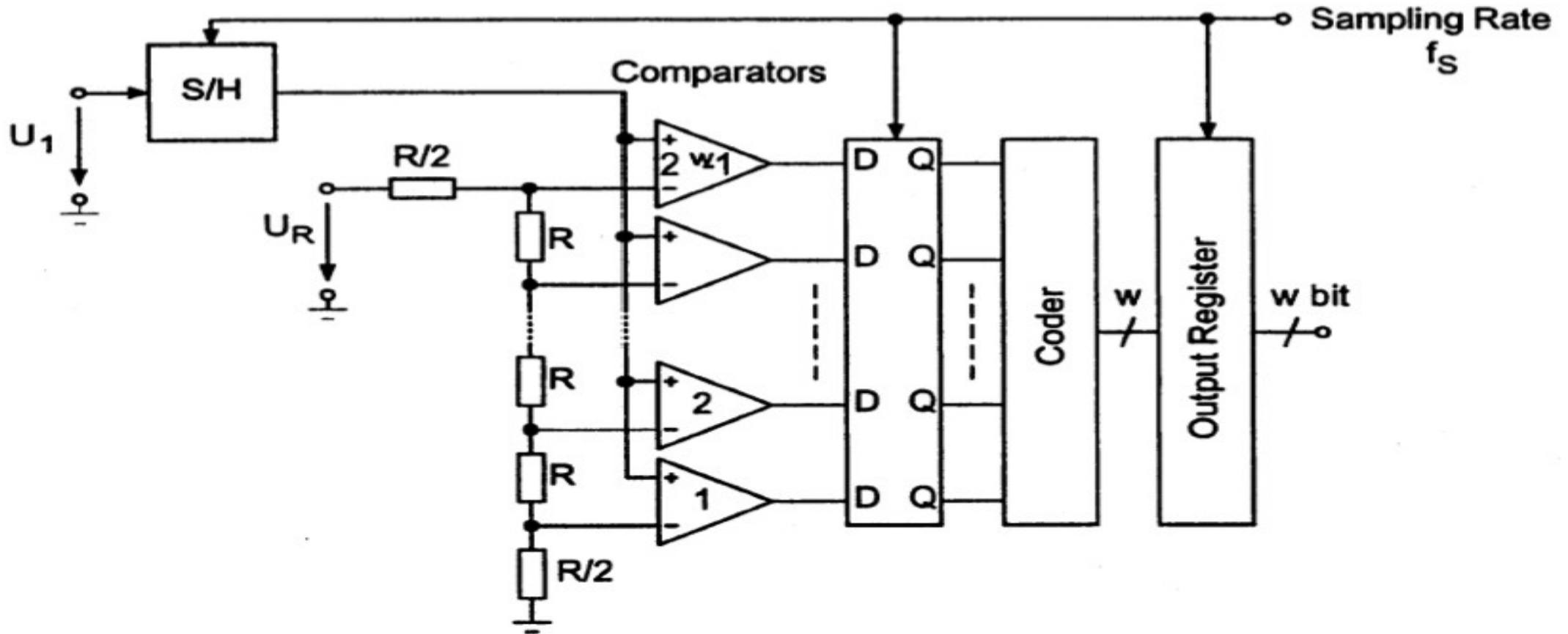


CNA:  
Pour aller plus loin...



# Convertisseur Analogique Numérique

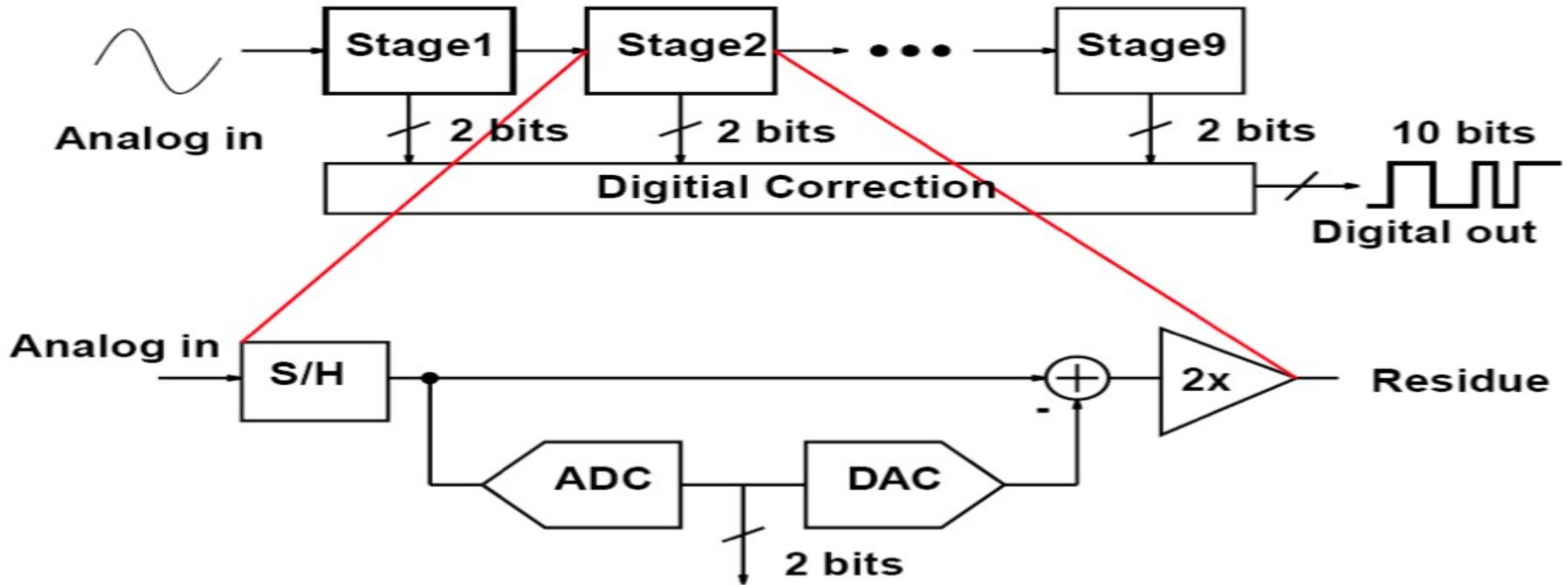
## □ CAN FLASH





# Convertisseur Analogique Numérique

## □ CAN PIPELINE

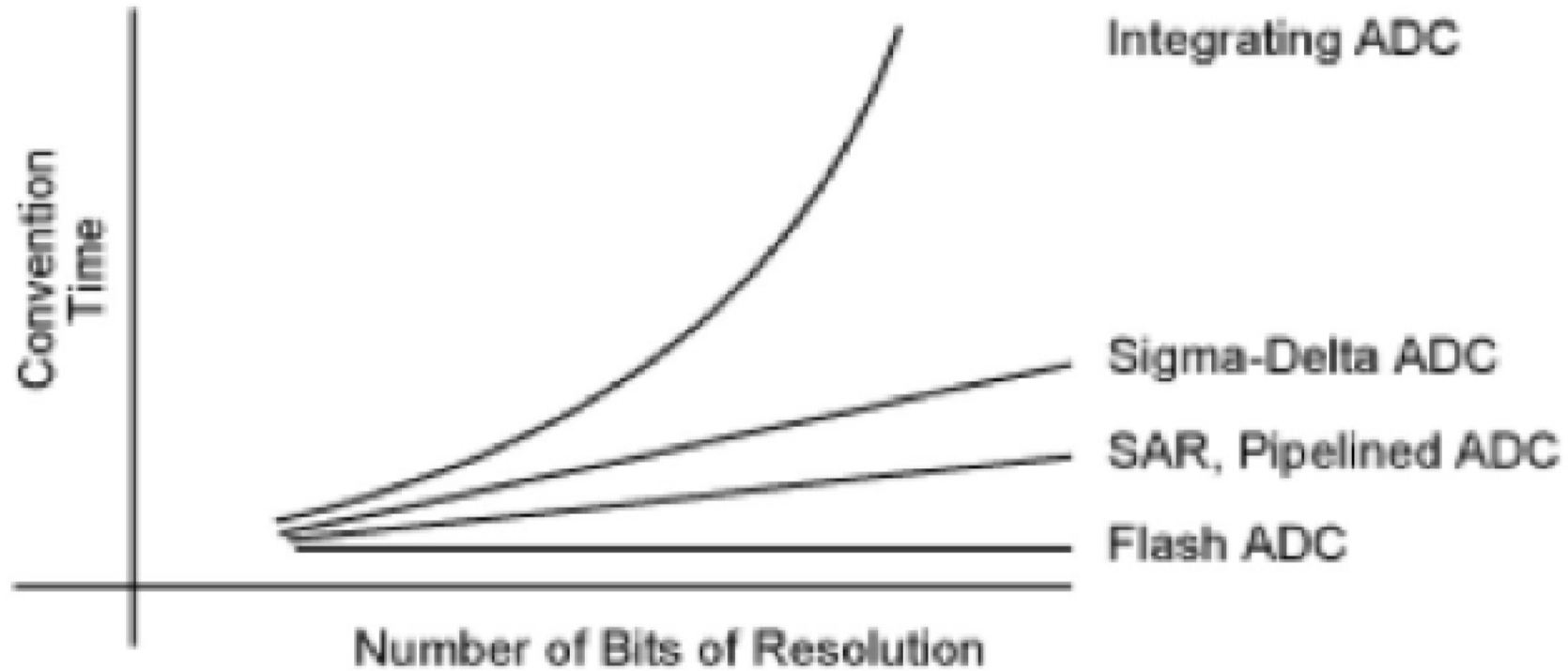


10 à 14 bits @ 150 MHz



# Convertisseur Analogique Numérique

## ☐ Comparaison





## Caractéristiques techniques

- SAR

- Résolutions programmables

  - 12, 10, 8, 6 et 4bits

- 19 canaux multiplexés

  - 16 sources externes

  - 2 sources internes

  - 1 canal dédié Vbat

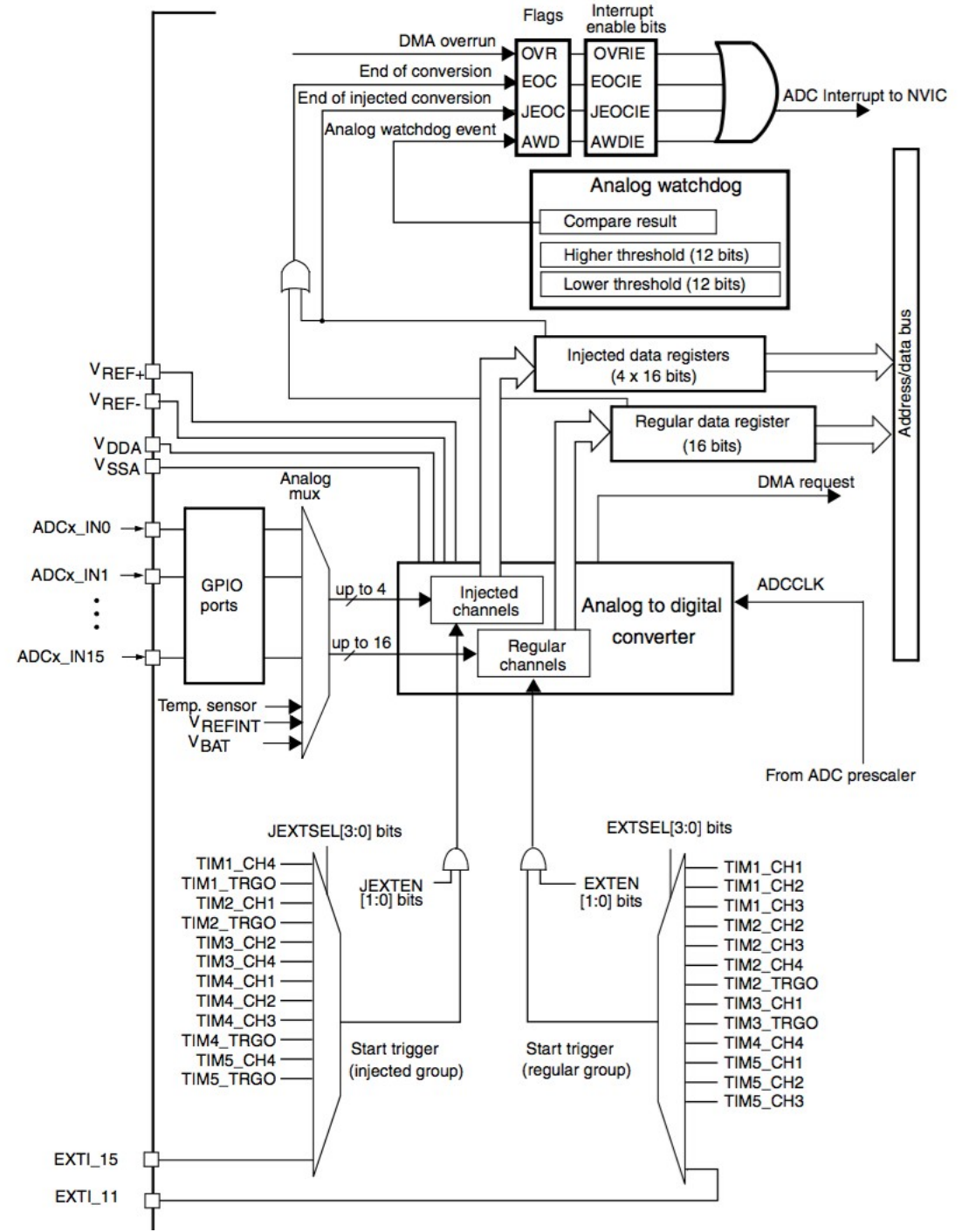
- Mode

  - Single

  - Continu/Discontinu

- Donnée stockée dans un registre 16 bits

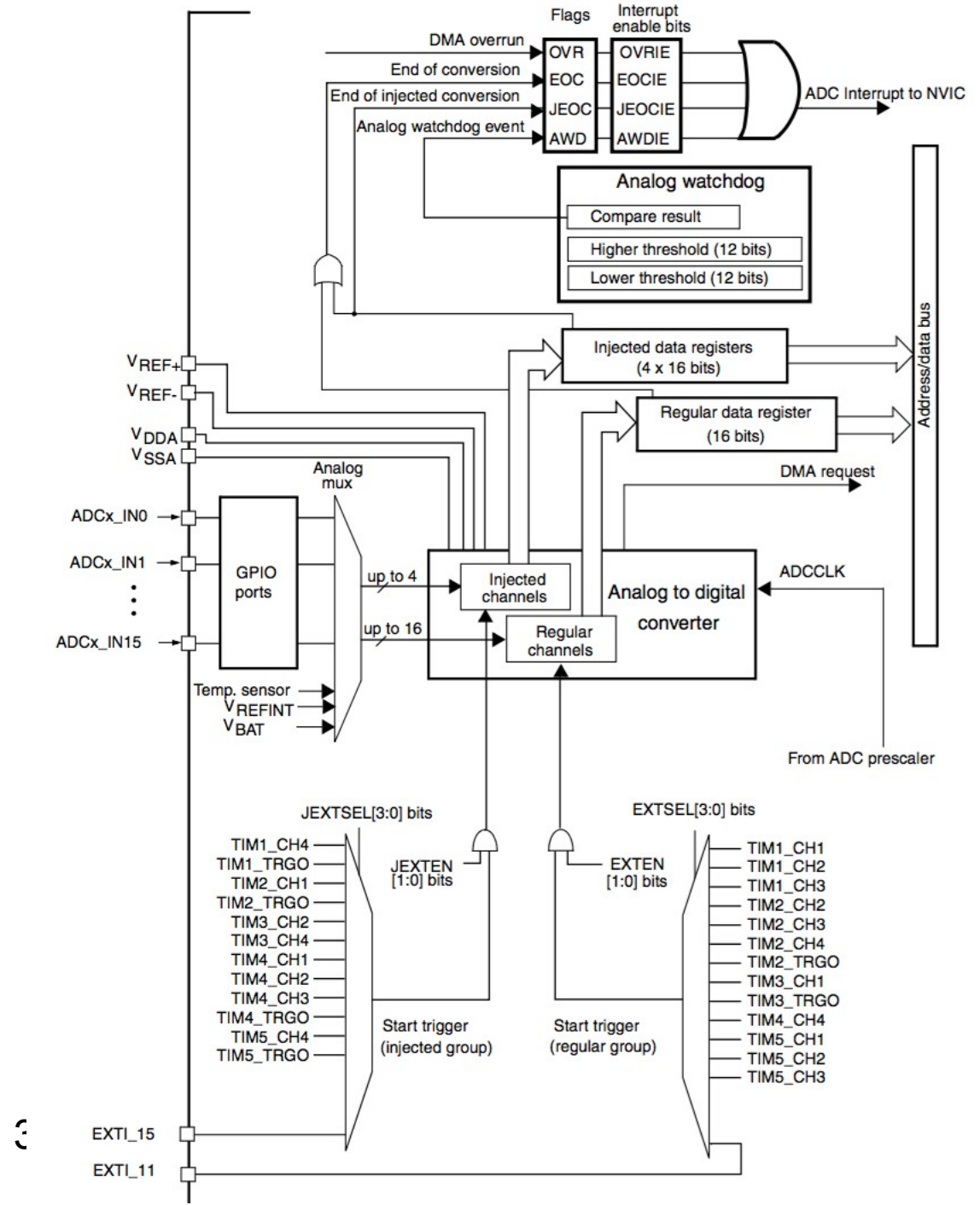
- Trigger externe



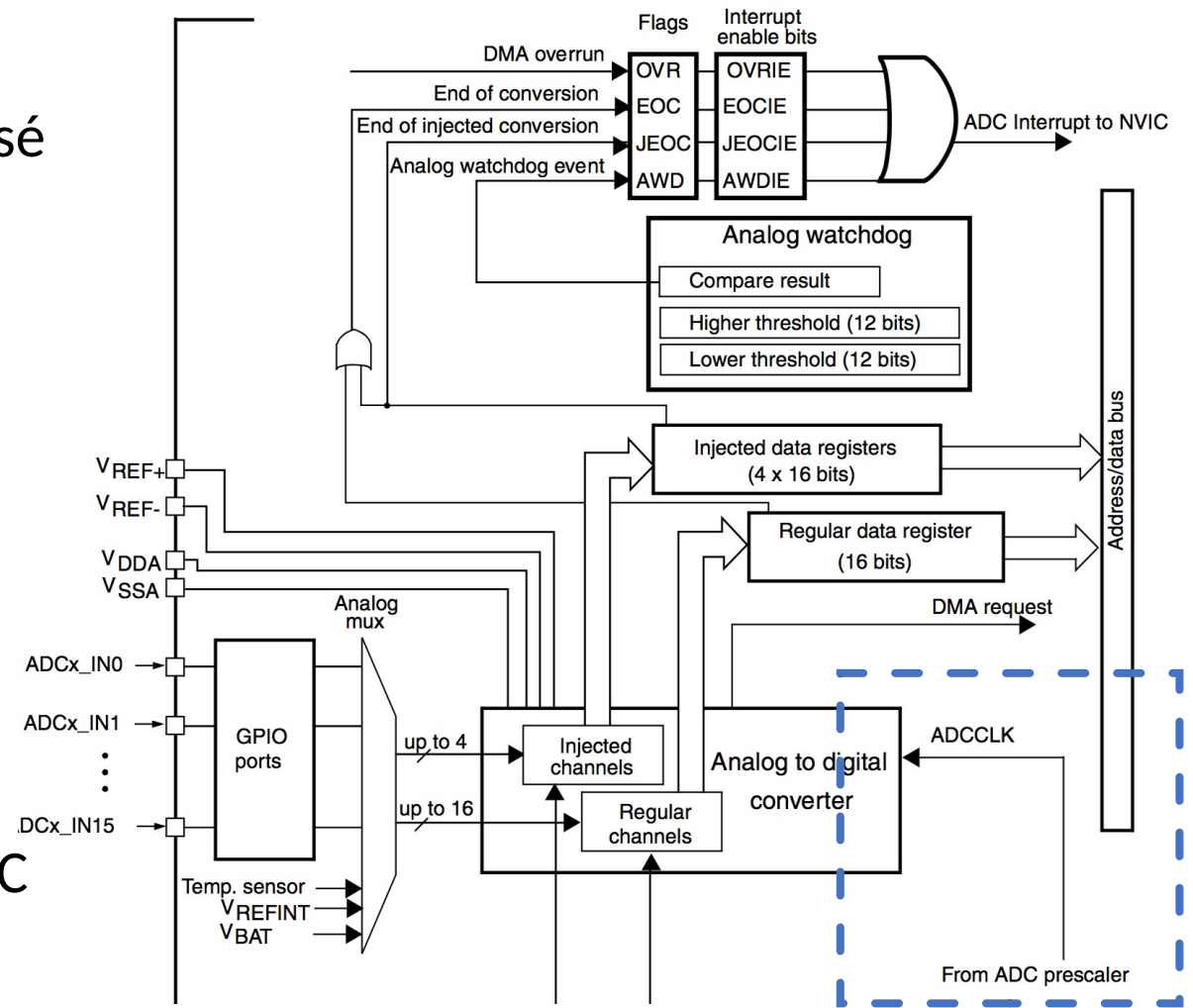


## Caractéristiques techniques

- Vref : 3.3V
- Amplitude max numérisable : Vref
- Amplitude min numérisable :  $q = 805\mu\text{V}$



- ❑ Deux domaines d'horloge
- ❑ ADCCLK : domaine analogique
  - ❑ ADCCLK généré à partir de l'horloge APB2 divisé par un prescaler programmable.
  - ❑ Permet à l'ADC de travailler à  $f_{PCLK2}/2$ ,  $/4$ ,  $/6$  or  $/8$ .
- ❑ Clock pour l'interface numérique (utilisé pour les registres en écriture et lecture)
  - ❑ Correspond à APB2 sous Cube.
  - ❑ Cette horloge peut être individuellement contrôlée (inhibée/autorisée) pour chaque ADC à partir du registre RCC\_APB2ENR.





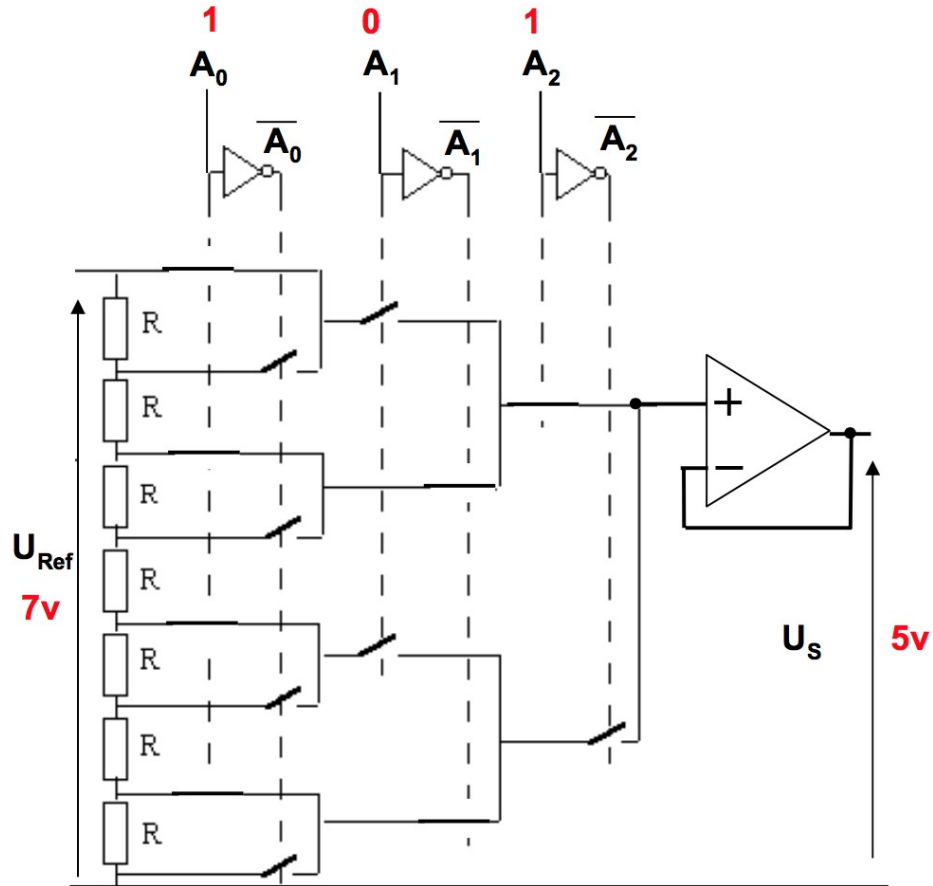
# Registre ADC STM32F

- Comment configurer son ADC ?
  - Choisir l'entrée
    - Activation et configuration de la broche d'entrée analogique
    - Choix des canaux réguliers
  - Réveil de l'ADC
    - Activation de l'ADC
  - Choix du mode de fonctionnement et du canal
  - Configuration de l'interruption
    - Permettre la génération d'une interruption en fin de conversion EOC
  - Déclenchement de la conversion
  - Récupération de la donnée numérisée

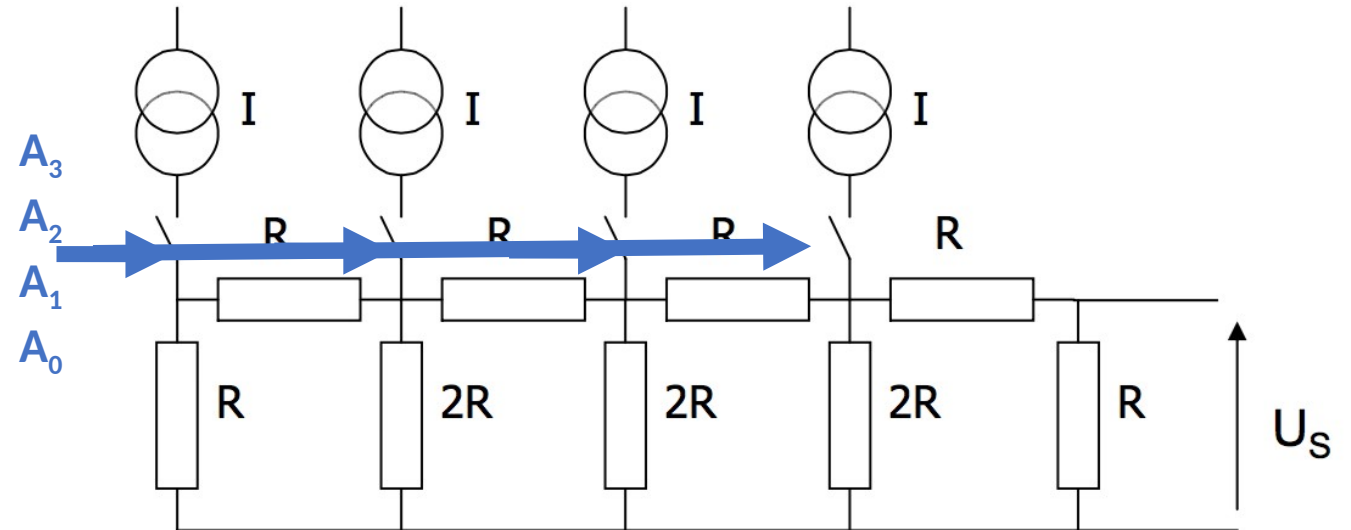
Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	ADC_SR	Reserved																								OVR	STRT	JSTRT	JEOC	EOC	AWD		
	Reset value																									0	0	0	0	0			
0x04	ADC_CR1	Reserved				OVRIE	RES[1:0]	AWDEN	JAWDEN	Reserved				DISC NUM [2:0]	JDISEN	DISCEN	JAUTO	AWD SGL	SCAN	JEOCIE	AWDIE	EOCIE	AWDCH[4:0]										
	Reset value					0	0	0	0					0	0	0	0	0	0	0	0	0	0	0									
0x08	ADC_CR2	Res r v e d	SWSTART	EXTEN[1:0]	EXTSEL [3:0]			Res r v e d	JSWSTART	JEXTEN[1:0]	JEXTSEL [3:0]			Reserved				ALIGN	EOCS	DDS	DMA	Reserved				CONT	ADON						
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	ADC_SMPR1	Sample time bits SMPx_x																															
	Reset value	0																															
0x10	ADC_SMPR2	Sample time bits SMPx_x																															
	Reset value	0																															

CAN:  
Pour aller plus loin...

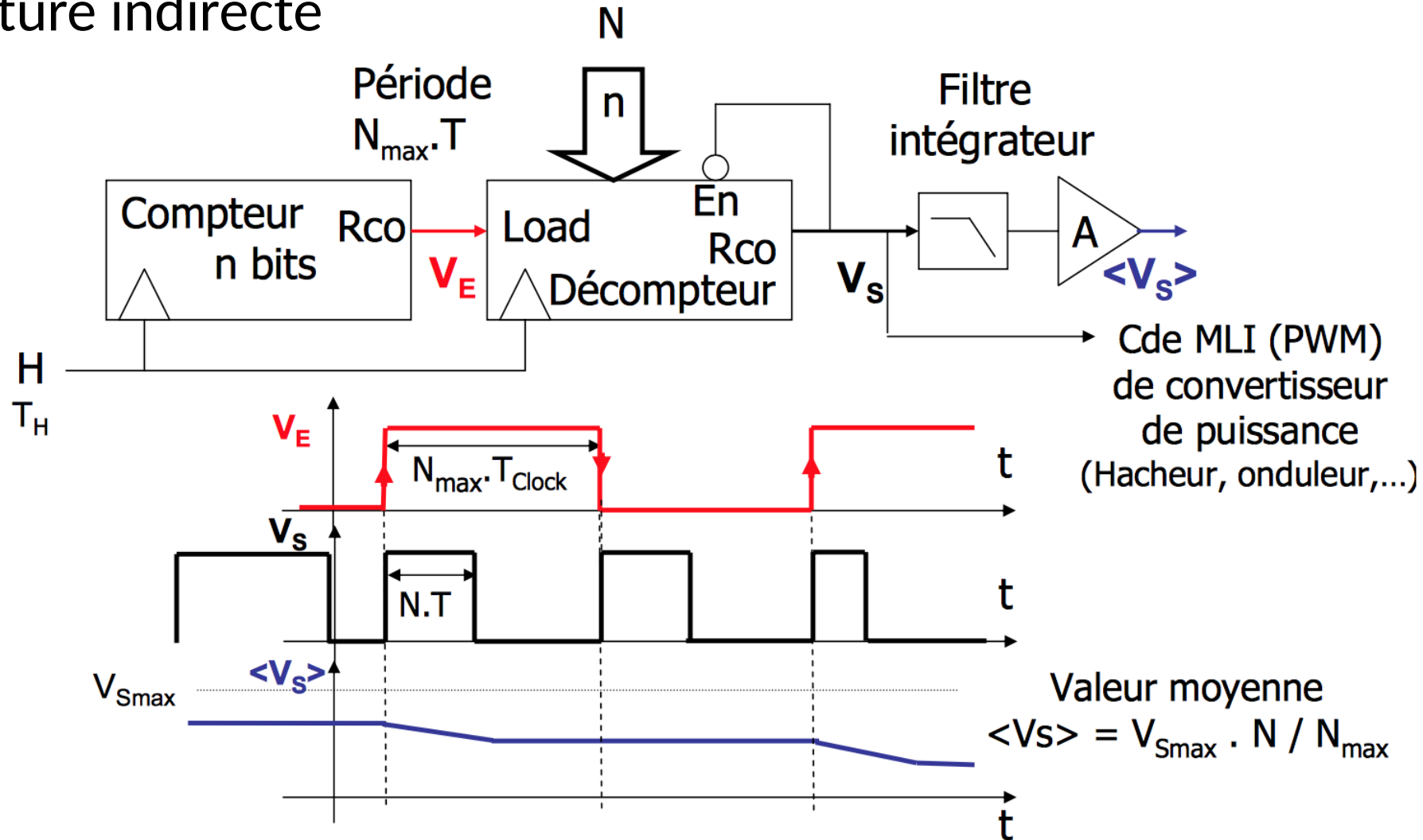
## Architectures potentiométrique



## Architecture à sources de courant



## Architecture indirecte



## Exemples de CNA

Part #	Resolution (bits)	Number of Channels	Programming Interface	Vout Range	Settling Time (typ) (s p-p)	Data Input Format	Update Rate (SPS)	Vnoise Spectral Density (typ) (dBm/rtHz)	Power Dissipation (typ) (W)	DAC Type	DAC Output Unipolar or
	12 Value...	11 Valu...	12 Values S...	40 V...	11n - 250u	15 Va...	12.5k - 12G	-166 - -129	200n - 2.65	6 Values Selected	4 Values S...
<a href="#">AD5761</a>	16	1	MicroWire, SPI	-2.5 to 7...	9μ	SPI	-	-	67.1m	Voltage Out	Bipolar, Unipolar
<a href="#">AD5761R</a>	16	1	MicroWire, SPI	±Vref	9μ	SPI	-	-	67.1m	Voltage Out	Bipolar, Unipolar
<a href="#">AD5721</a>	12	1	MicroWire, SPI	-2.5 to 7...	9μ	SPI	-	-	67.1m	Voltage Out	Bipolar, Unipolar
<a href="#">AD5675</a>	16	8	I <sup>2</sup> C, Serial	-	5μ	2 wire se...	-	-	300m	Voltage Out	Unipolar
<a href="#">AD9136</a>	16	2	SPI	-	20n	JESD20...	2.8G	-163	1.42	Current Out	-
<a href="#">AD9135</a>	11	2	SPI	-	20n	JESD20...	2.8G	-157	1.42	Current Out	-
<a href="#">AD5721R</a>	12	1	MicroWire, SPI	±Vref	9μ	SPI	-	-	67.1m	Voltage Out	Bipolar, Unipolar
<a href="#">AD5676R</a>	16	8	Serial, SPI	-	5μ	SPI	-	-	300m	Voltage Out	Unipolar
<a href="#">AD5676</a>	16	8	Serial, SPI	-	5μ	SPI	-	-	300m	Voltage Out	Unipolar
<a href="#">AD5675R</a>	12	8	I <sup>2</sup> C, Serial	-	5μ	2 wire se...	-	-	300m	Voltage Out	Unipolar
<a href="#">AD5672R</a>	12	8	Serial, SPI	-	5μ	SPI	-	-	300m	Voltage Out	Unipolar
<a href="#">AD5671R</a>	12	8	I <sup>2</sup> C, Serial	-	5μ	2 wire se...	-	-	300m	Voltage Out	Unipolar
<a href="#">AD9144</a>	16	4	SPI	-	20n	JESD20...	2.8G	-163	1.59	Current Out	-
<a href="#">AD5693R</a>	16	1	I <sup>2</sup> C	0 to Vref...	5μ	I <sup>2</sup> C	-	-	-	Voltage Out	Unipolar
<a href="#">AD5693</a>	16	1	I <sup>2</sup> C	0 to Vref...	5μ	I <sup>2</sup> C	-	-	-	Voltage Out	Unipolar

Les STM32 intègrent pour la plupart des DAC avec différentes configurations en fonction de la série STM32Fx :

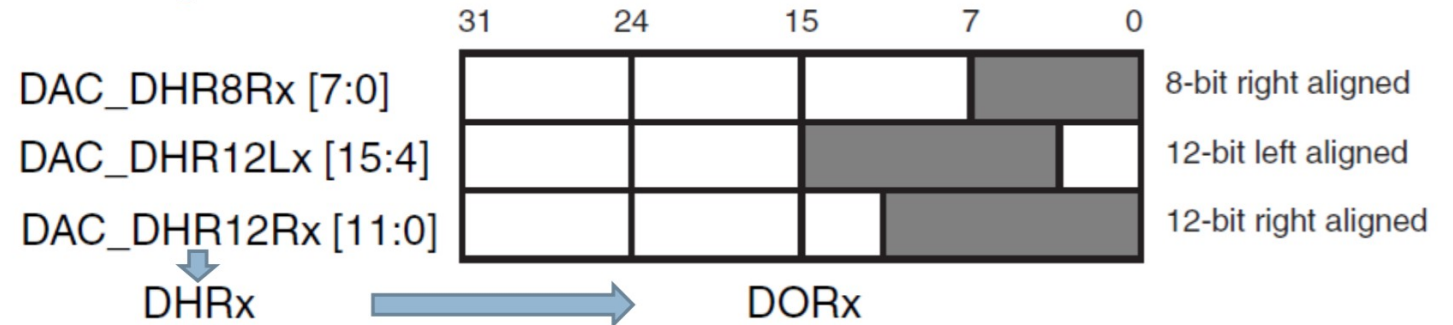
- 1 à 3 sorties
- Générateur de bruit blanc
- Générateur de signal triangulaire
- DMA
- Horloge analogique dédiée

Series	Product RPN	DAC outputs	White noise generator	Triangular wave generator	DMA capability	DMA underrun error
F0	STM32F030xx STM32F031xx STM32F038xx STM32F042xx STM32F048xx STM32F070xx	0	-	-	-	-
	STM32F051xx STM32F058xx	1	No	No	Yes	No
	STM32F071xx STM32F072xx STM32F078xx STM32F091xx STM32F098xx	2	Yes	Yes	Yes	Yes
F1	STM32F101x4/6/8B STM32F102xx STM32F103x4/6/8B	0	-	-	-	-
	STM32F100xx STM32F101xC/D/E/F/G STM32F103xC/D/E/F/G STM32F105xx STM32F107xx	2	Yes	Yes	Yes	Yes
F2	STM32F2xxxx	2	Yes	Yes	Yes	Yes

## 3 formats de donnée

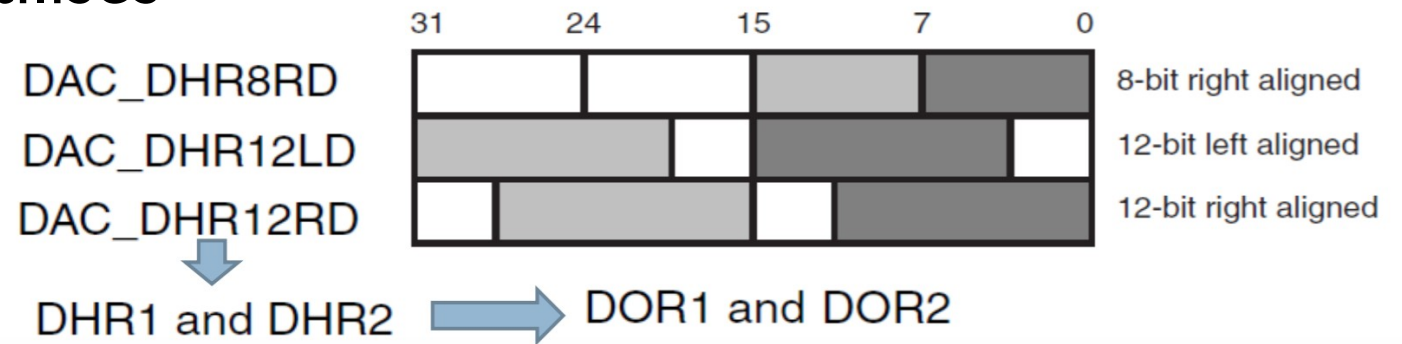
- 8 bits
- 12 bits alignés à droite
- 12 bits alignés à gauche

### 1 sortie : 1 canal



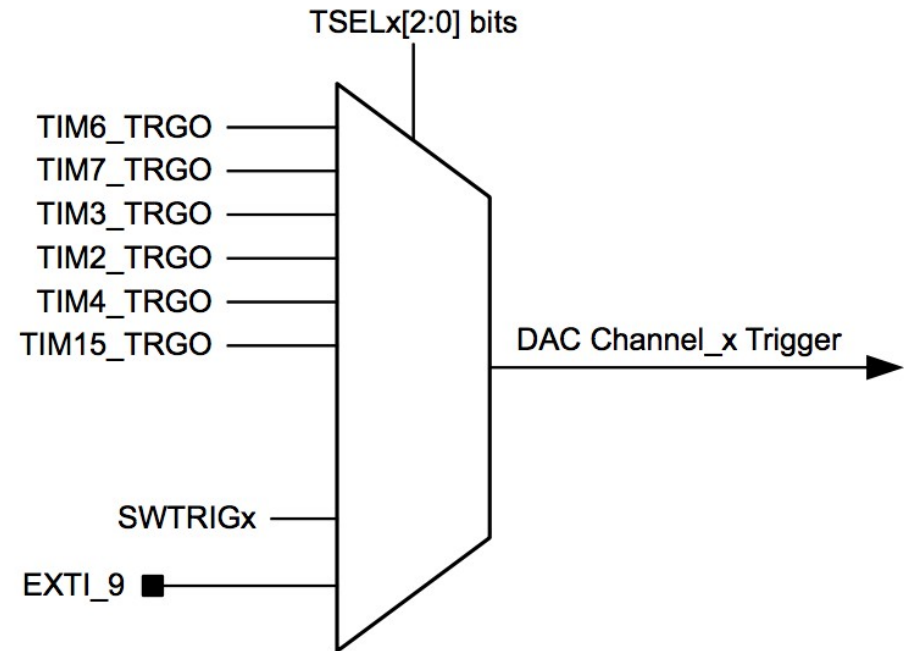
## Dépend du nombre de canaux utilisés

### 2 sorties : 2 canaux



## Triggers utilisable pour synchroniser le DAC

- Triggers logiciel SWTRIG<sub>x</sub>
- Trigger externe : EXTI\_<sub>x</sub>
- Trigger provenant de Timer :
  - TIM<sub>x</sub>\_TRGO : dépend du STM32 utilisé
  - Sur un front montant de TIM<sub>x</sub>\_TRGO, la dernière donnée enregistrée dans le registre DAC\_DHR<sub>x</sub> est transférée au registre DAC\_DOR<sub>x</sub>



Source	Type	TSEL[2:0]
Timer 6 TRGO event	Internal signal from on-chip timers	000
Timer 3 TRGO event		001
Timer 7 TRGO event		010
Timer 5 or Timer 15 TRGO event		011
Timer 2 TRGO event		100
Timer 4 TRGO event		101
EXTI line9	External pin	110
SWTRIG	Software control bit	111





# STM32 CNA

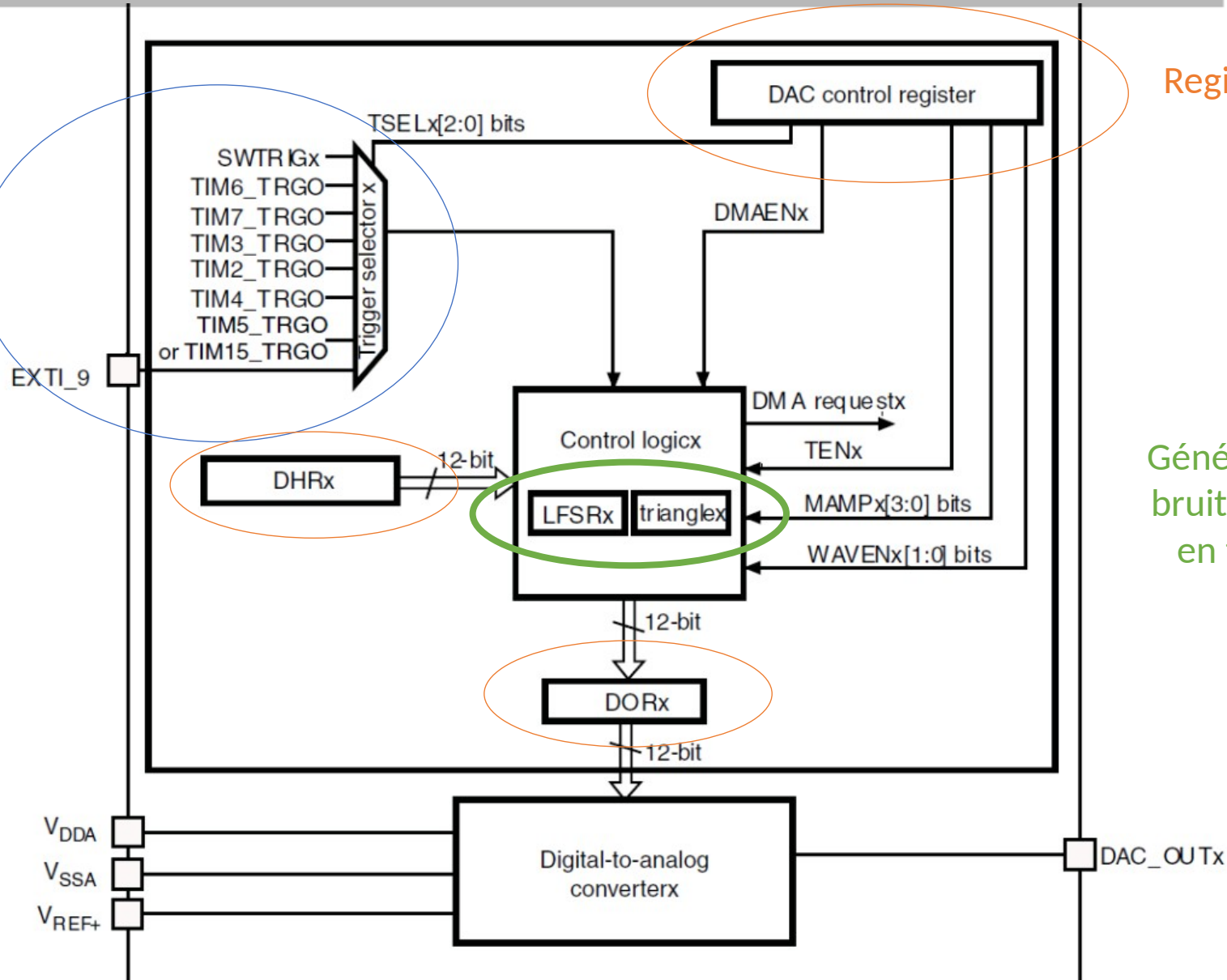
Configuration  
du trigger

Registres du  
DAC

Générateur de  
bruit (LFSR) et  
en triangle

Entrée externe  
de référence

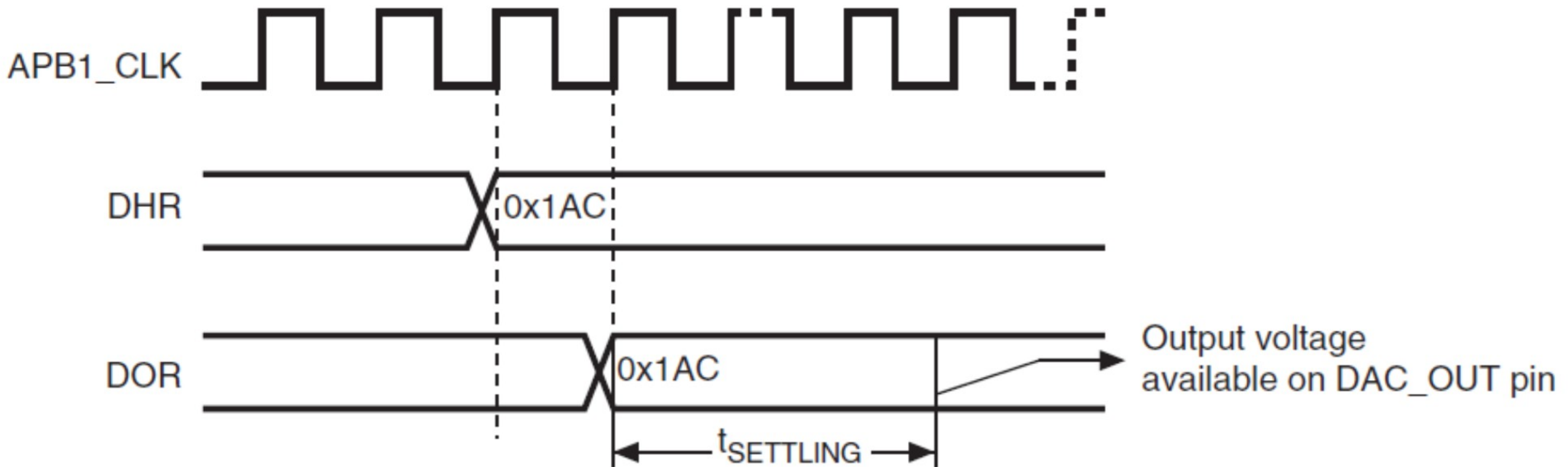
Sortie du DAC  
Tension  
analogique



$$DACoutput = V_{REF} \times \frac{DOR}{4095}$$

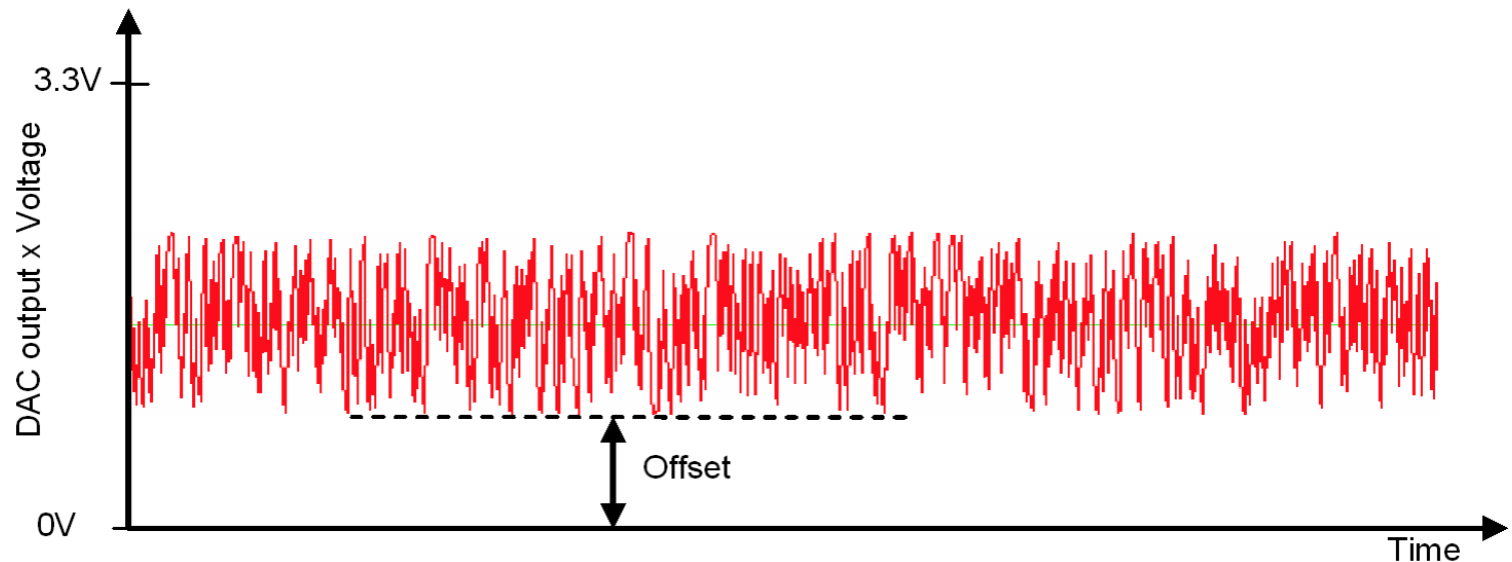
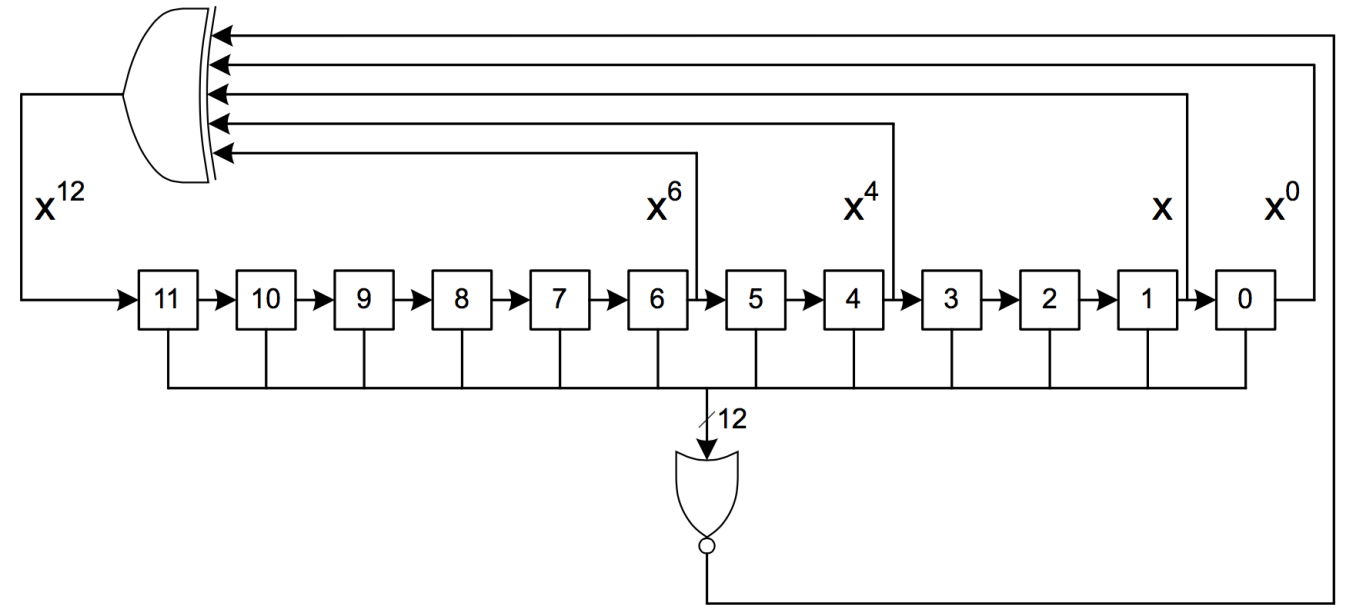
- ❑ Le registre `DAC_DORx` ne peut pas être écrit directement et les données sont transférées uniquement lorsque le registre `DAC_DHRx` est chargé d'une nouvelle valeur.
- ❑ Les données enregistrées dans le registre `DAC_DHRx` sont transférées au registre `DAC_DORx` après un cycle d'horloge, si aucun trigger est sélectionné (`TENx` bit dans le registre `DAC_CR` est reseté).
- ❑ Quand un trigger est sélectionné (`TENx` bit dans le registre `DAC_CR` est autorisé), et un signal arrive, le transfert est réalisé en 3 cycles d'horloge.
- ❑ Quand le registre `DAC_DORx` est chargé avec le contenu du registre `DAC_DHRx`, la sortie analogique prend sa nouvelle valeur après un temps d'établissement  $T_{\text{settling}}$ , qui dépend de la tension d'alimentation et de l'impédance de charge en sortie du DAC.

# STM32 CNA : fonctionnement



## Générateur de bruit :

- LFSR – Linear Feedback Shift Register
- 12 bascules D avec un rebouclage sur la 1ère
- Longueur de la séquence : pôleynome de rebouclage
- Génère un pseudo code aléatoire de longueur finie
- Répartition fréquentielle s'apparente à un bruit blanc (répartition constante de la puissance en fonction des fréquences)



# CAN/ADC : configuration des registres



# CAN STM32F

## Registre RCC\_APB1ENR

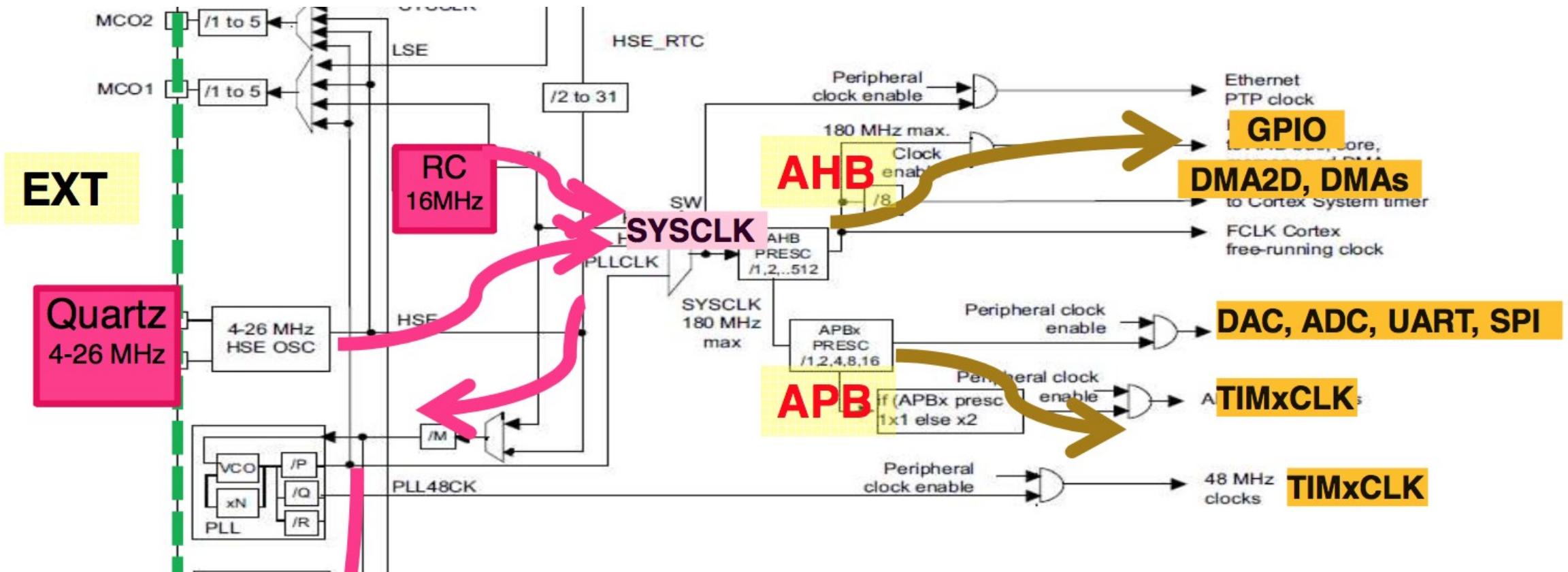
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UART8 EN	UART7 EN	DAC EN	PWR EN	Reserved	CAN2 EN	CAN1 EN	Reserved	I2C3 EN	I2C2 EN	I2C1 EN	UART5 EN	UART4 EN	USART3 EN	USART2 EN	Reserved
rw	rw	rw	rw		rw	rw		rw	rw	rw	rw	rw	rw	rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI3 EN	SPI2 EN	Reserved		WWDG EN	Reserved		TIM14 EN	TIM13 EN	TIM12 EN	TIM7 EN	TIM6 EN	TIM5 EN	TIM4 EN	TIM3 EN	TIM2 EN
rw	rw			rw			rw	rw	rw	rw	rw	rw	rw	rw	rw

## Registre RCC\_APB2ENR

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				LTDC EN	Reserved				SAI1EN	SPI6EN	SPI5EN	Res.	TIM11 EN	TIM10 EN	TIM9 EN
				rw					rw	rw	rw		rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	SYSCFG EN	SPI4EN	SPI1 EN	SDIO EN	ADC3 EN	ADC2 EN	ADC1 EN	Reserved	USART6 EN	USART1 EN	Reserved		TIM8 EN	TIM1 EN	
	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw			rw	rw



# CAN STM32F



# Registre ADC STM32F

## ❑ Contrôle ADC (on/off)

❑ L'ADC est mise en marche en positionnant le bit ADON à 1 dans le registre ADC\_CR2

❑ L'ADC passe du mode Power-Down au mode WAKE-UP

❑ Le bit ADON positionner à 0 permet aussi d'arrêter la conversion de l'ADC.

❑ La conversion commence après que soit le bit SWSTART ou que le bit

JSWSTART

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved	SWST ART	EXTEN			EXTSEL[3:0]				reserved	JSWST ART	JEXTEN		JEXTSEL[3:0]			
		rw	rw	rw	rw	rw	rw	rw			rw	rw	rw	rw	rw	rw
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved					ALIGN	EOCS	DDS	DMA	Reserved						CONT	ADON
					rw	rw	rw	rw							rw	rw

Bit 31 Reserved, must be kept at reset value.

Bit 30 **SWSTART**: Start conversion of regular channels

This bit is set by software to start conversion and cleared by hardware as soon as the conversion starts.

0: Reset state

1: Starts conversion of regular channels

*Note: This bit can be set only when ADON = 1 otherwise no conversion is launched.*



# Registre ADC STM32F

## ☐ Sélection des canaux

☐ 16 canaux sont multiplexés

☐ 2 types d'acquisition

### ☐ Regular Channel :

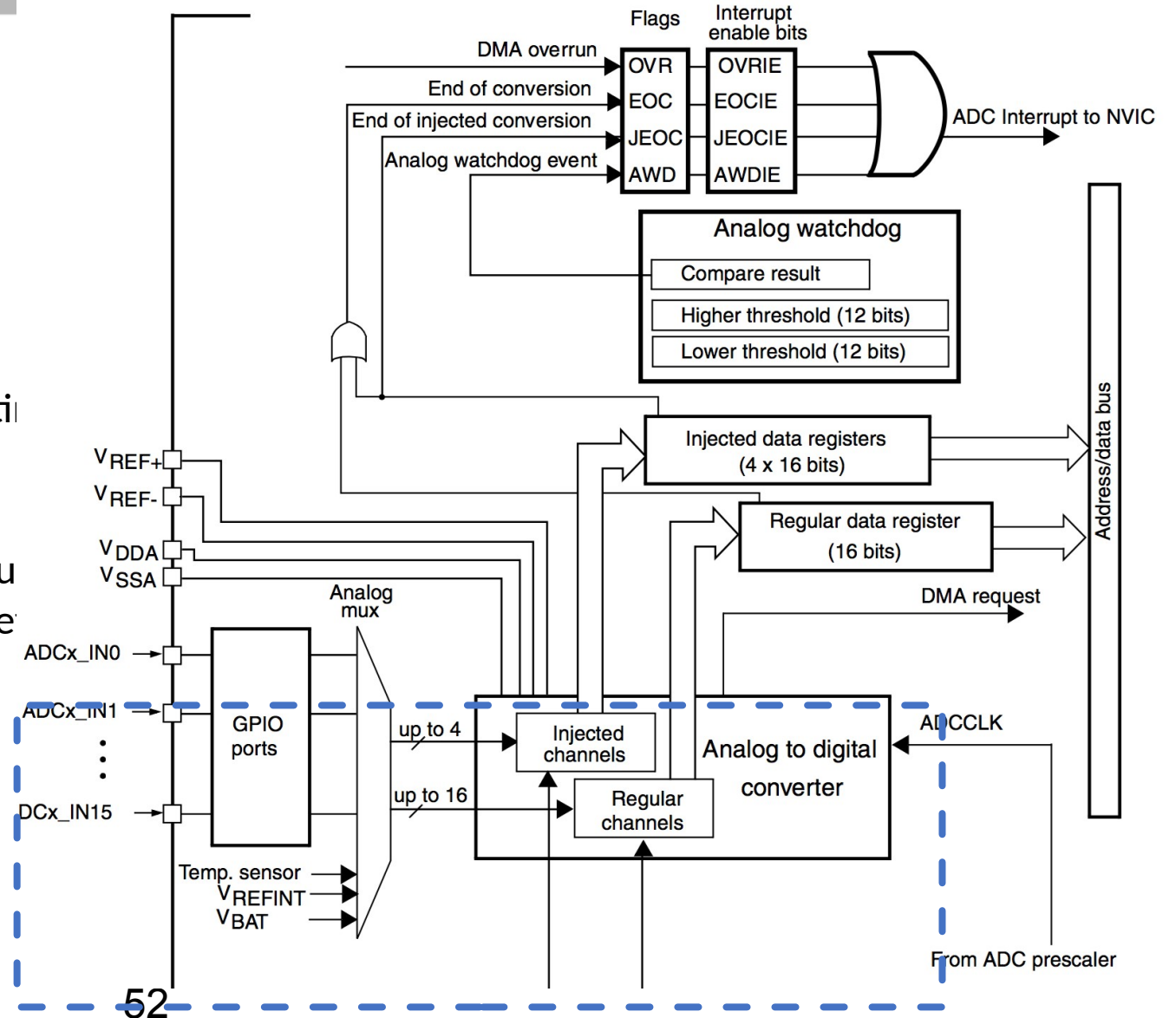
☐ Les canaux ainsi que l'ordre d'acquisition peut être définie à partir de la configuration du registre ADC\_SQRx ; à travers les bits L[3:0]

☐ Exemple : Acquisition de ADC\_IN3, puis de ADC\_IN2, puis de ADC\_IN15, ..., e ADC\_IN4

### ☐ Injected Channel :

☐ Composé jusqu'à 4 canaux

☐ Registre ADC\_JSQR



# Registre ADC STM32F

## ☐ Sélection du type de conversion

### ☐ Acquisition unique

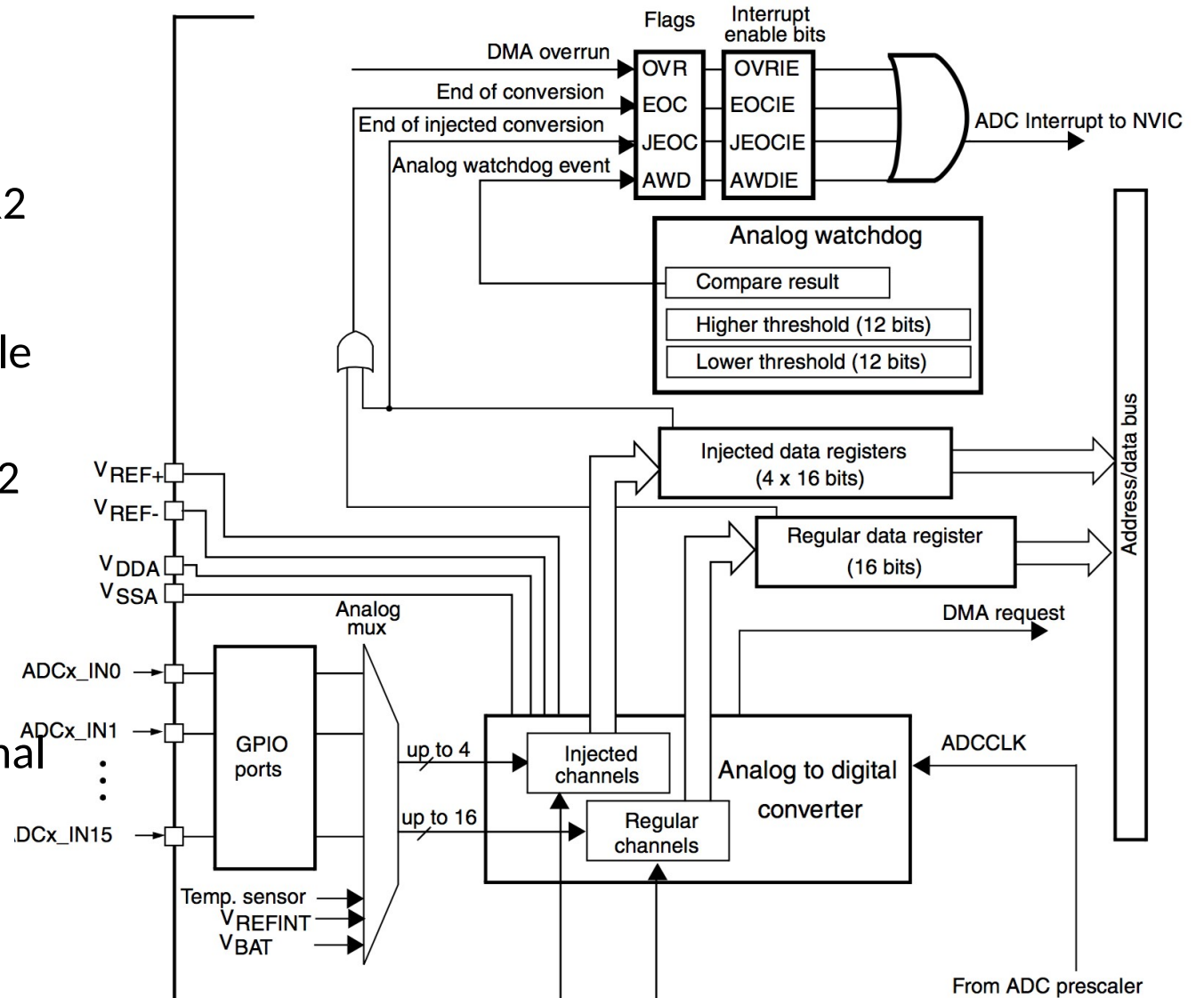
- ☐ Mode asynchrone
- ☐ Bit SWSTART dans le registre ADC\_CR2

### ☐ Acquisition continue

- ☐ A la fin d'une acquisition, une nouvelle est faite automatiquement
- ☐ Bit JWSTART dans le registre ADC\_CR2

### ☐ Acquisition de type balayage

- ☐ Ordre de balayage est défini dans le registre ADC\_SQRx
- ☐ Conversion séquentielle canal par canal automatiquement

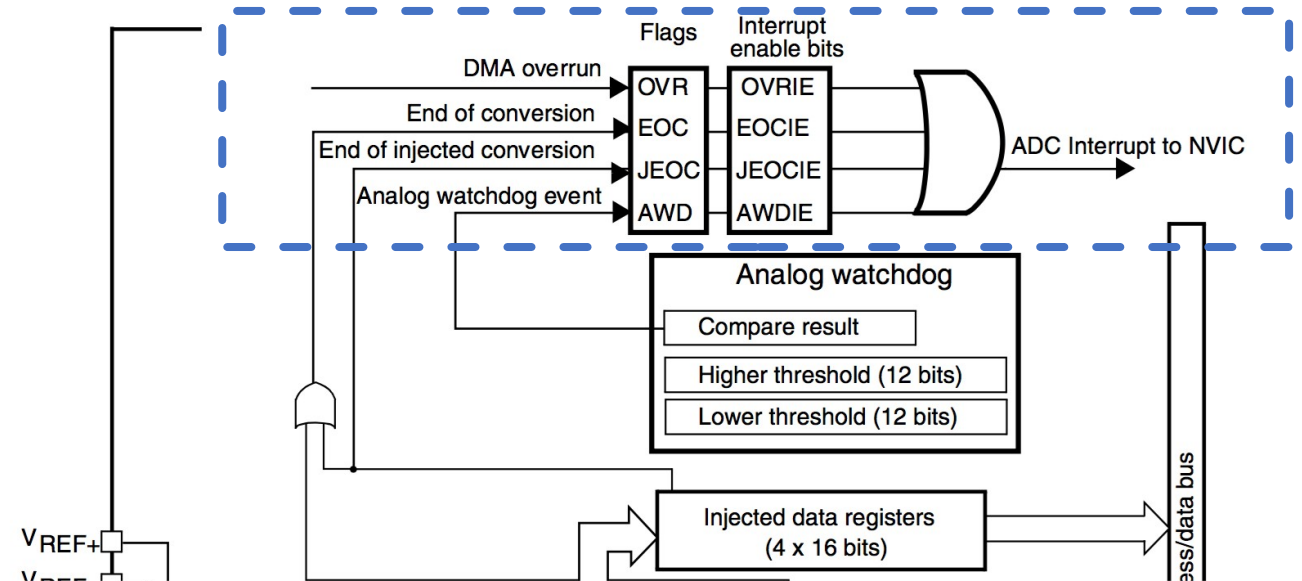


- Acquisition sur un trigger
  - Externe ou interne
  - Programmation du type de signal (front, niveau, etc.)
  - Configuration via EXTEN[1:0] dans les registre EXTI

Source	EXTEN[1:0] / JEXTEN[1:0]
Trigger detection disabled	00
Detection on the rising edge	01
Detection on the falling edge	10
Detection on both the rising and falling edges	11

# Registre ADC STM32F

- ❑ Interruption en fin de conversion
  - ❑ Générée pour tout type de conversion
  - ❑ Le type de d'interruption est programmable ADC\_CR1 et son état est disponible dans le registre ADC\_SR



Interrupt event	Event flag	Enable control bit
End of conversion of a regular group	EOC	EOCIE
End of conversion of an injected group	JEOC	JEOCIE
Analog watchdog status bit is set	AWD	AWDIE
Overrun	OVR	OVRIE

# Registre ADC STM32F

## Fréquence d'échantillonnage

### Pour le mode continue

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					SMP18[2:0]			SMP17[2:0]			SMP16[2:0]			SMP15[2:1]	
					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMP15_0	SMP14[2:0]			SMP13[2:0]			SMP12[2:0]			SMP11[2:0]			SMP10[2:0]		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 31: 27 Reserved, must be kept at reset value.

Bits 26:0 **SMPx[2:0]**: Channel x sampling time selection

These bits are written by software to select the sampling time individually for each channel.  
During sampling cycles, the channel selection bits must remain unchanged.

Note: 000: 3 cycles  
 001: 15 cycles  
 010: 28 cycles  
 011: 56 cycles  
 100: 84 cycles  
 101: 112 cycles  
 110: 144 cycles  
 111: 480 cycles

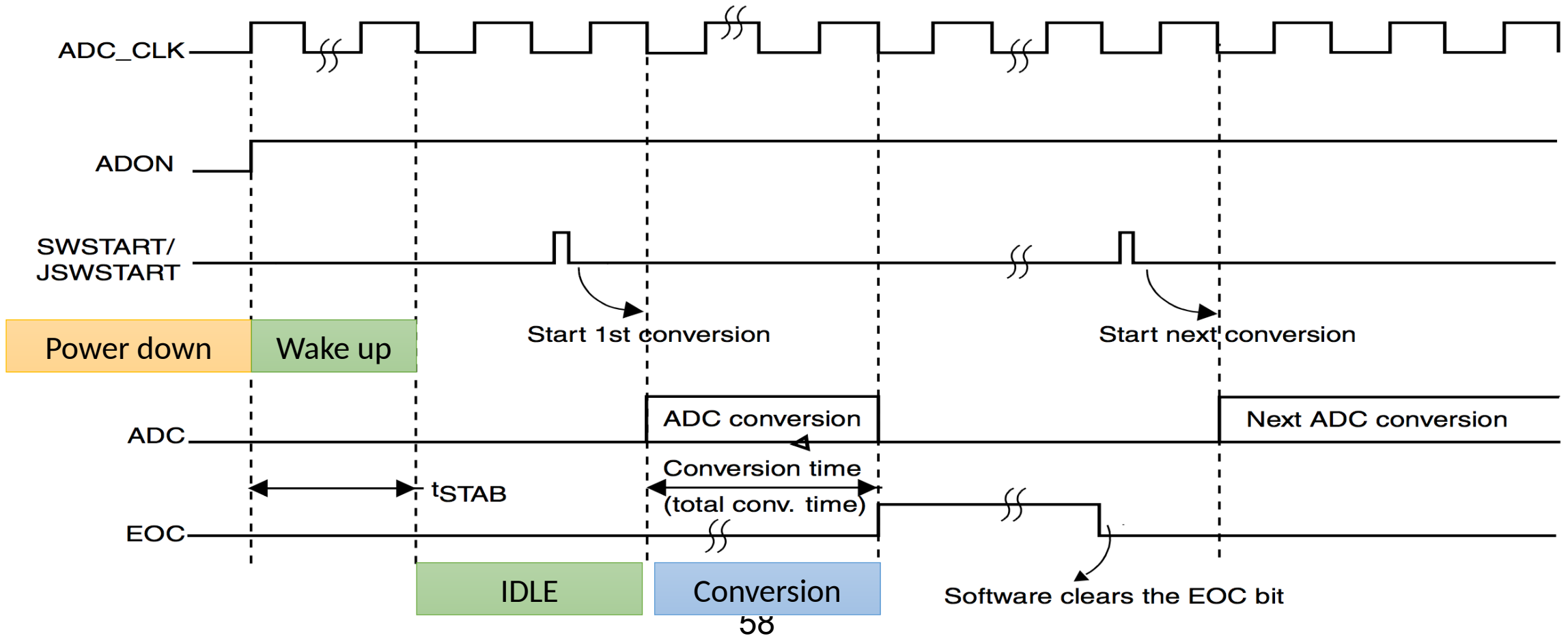
# Registre ADC STM32F

## Les principaux registres

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0x00	ADC_SR	Reserved																								OVR	STRT	JSTRT	JEOC	EOC	AWD			
	Reset value																									0	0	0	0	0	0			
0x04	ADC_CR1	Reserved						OVRIE	RES[1:0]	AWDEN	JAWDEN	Reserved						DISC NUM [2:0]	JDISCEN	DISCEN	JAUTO	AWD SGL	SCAN	JEOCIE	AWDIE	EOCIE	AWDCH[4:0]							
	Reset value							0	0	0	0							0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	ADC_CR2	Rese ded	SWSTART	EXTEN[1:0]	EXTSEL [3:0]				Rese ded	JSWSTART	JEXTEN[1:0]	JEXTSEL [3:0]				Reserved				ALIGN	EOCS	DDS	DMA	Reserved				CONT	ADON					
	Reset value		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0x0C	ADC_SMPR1	Sample time bits SMPx_x																																
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	ADC_SMPR2	Sample time bits SMPx_x																																
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

# Registre ADC STM32F

□ Timing : 15 cycles d'horloge pour le temps de conversion pour 12 bits





IUT

CERGY-PONTOISE

# CNA / DAC: configuration des registres



## Registre DAC\_CR

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		DMAU DRIE2	DMA EN2	MAMP2[3:0]				WAVE2[1:0]		TSEL2[2:0]			TEN2	BOFF2	EN2
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		DMAU DRIE1	DMA EN1	MAMP1[3:0]				WAVE1[1:0]		TSEL1[2:0]			TEN1	BOFF1	EN1
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 11:8 **MAMP1[3:0]**: DAC channel1 mask/amplitude selector

These bits are written by software to select mask in wave generation mode or amplitude in triangle generation mode.

- 0000: Unmask bit0 of LFSR/ triangle amplitude equal to 1
- 0001: Unmask bits[1:0] of LFSR/ triangle amplitude equal to 3
- 0010: Unmask bits[2:0] of LFSR/ triangle amplitude equal to 7
- 0011: Unmask bits[3:0] of LFSR/ triangle amplitude equal to 15
- 0100: Unmask bits[4:0] of LFSR/ triangle amplitude equal to 31
- 0101: Unmask bits[5:0] of LFSR/ triangle amplitude equal to 63
- 0110: Unmask bits[6:0] of LFSR/ triangle amplitude equal to 127
- 0111: Unmask bits[7:0] of LFSR/ triangle amplitude equal to 255
- 1000: Unmask bits[8:0] of LFSR/ triangle amplitude equal to 511
- 1001: Unmask bits[9:0] of LFSR/ triangle amplitude equal to 1023
- 1010: Unmask bits[10:0] of LFSR/ triangle amplitude equal to 2047
- ≥ 1011: Unmask bits[11:0] of LFSR/ triangle amplitude equal to 4095

Bits 7:6 **WAVE1[1:0]**: DAC channel1 noise/triangle wave generation enable

These bits are set and cleared by software.

- 00: wave generation disabled
- 01: Noise wave generation enabled
- 1x: Triangle wave generation enabled

Bits 5:3 **TSEL1[2:0]**: DAC channel1 trigger selection

Bit 2 **TEN1**: DAC channel1 trigger enable

Bit 1 **BOFF1**: DAC channel1 output buffer disable

Bit 0 **EN1**: DAC channel1 enable

## ☐ Registre DAC\_SWTRIGR

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														SWTRIG2	SWTRIG1
														w	w

### Bit 1 **SWTRIG2**: DAC channel2 software trigger

This bit is set and cleared by software to enable/disable the software trigger.

0: Software trigger disabled

1: Software trigger enabled

*Note: This bit is cleared by hardware (one APB1 clock cycle later) once the DAC\_DHR2 register value has been loaded into the DAC\_DOR2 register.*

### Bit 0 **SWTRIG1**: DAC channel1 software trigger

This bit is set and cleared by software to enable/disable the software trigger.

0: Software trigger disabled

1: Software trigger enabled

*Note: This bit is cleared by hardware (one APB1 clock cycle later) once the DAC\_DHR1 register value has been loaded into the DAC\_DOR1 register.*

## ☐ Registre TIMX\_CR2

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reserved								TI1S	MMS[2:0]			CCDS	Reserved			
								rw	rw	rw	rw	rw				

Bits 6:4 **MMS**: Master mode selection

These bits allow to select the information to be sent in master mode to slave timers for synchronization (TRGO). The combination is as follows:

000: **Reset** - the UG bit from the TIMx\_EGR register is used as trigger output (TRGO). If the reset is generated by the trigger input (slave mode controller configured in reset mode) then the signal on TRGO is delayed compared to the actual reset.

001: **Enable** - the Counter enable signal, CNT\_EN, is used as trigger output (TRGO). It is useful to start several timers at the same time or to control a window in which a slave timer is enabled. The Counter Enable signal is generated by a logic OR between CEN control bit and the trigger input when configured in gated mode.

When the Counter Enable signal is controlled by the trigger input, there is a delay on TRGO, except if the master/slave mode is selected (see the MSM bit description in TIMx\_SMCR register).

010: **Update** - The update event is selected as trigger output (TRGO). For instance a master timer can then be used as a prescaler for a slave timer.

011: **Compare Pulse** - The trigger output send a positive pulse when the CC1IF flag is to be set (even if it was already high), as soon as a capture or a compare match occurred. (TRGO)

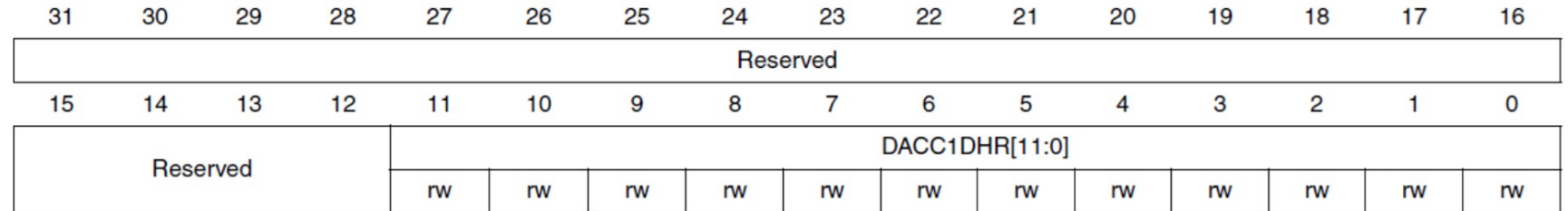
100: **Compare** - OC1REF signal is used as trigger output (TRGO)

101: **Compare** - OC2REF signal is used as trigger output (TRGO)

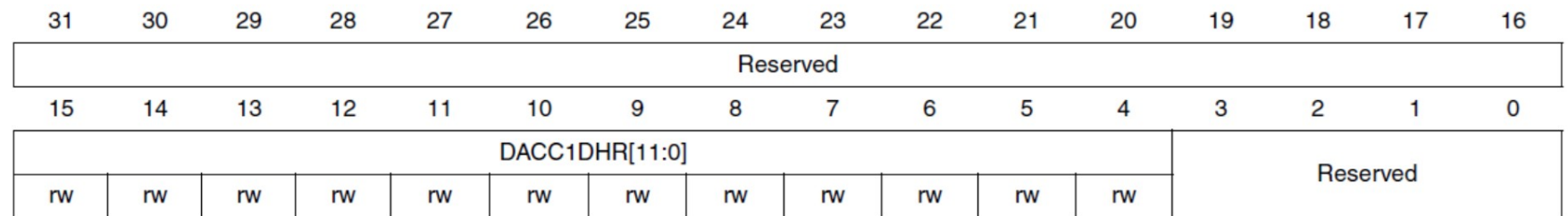
110: **Compare** - OC3REF signal is used as trigger output (TRGO)

111: **Compare** - OC4REF signal is used as trigger output (TRGO)

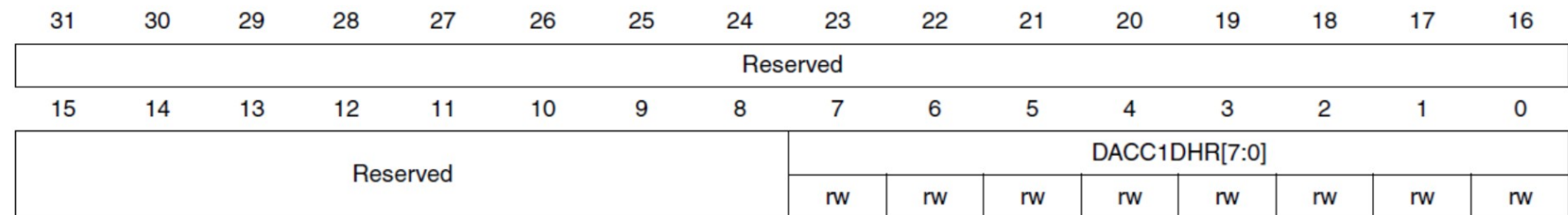
## ☐ Registre DAC\_DHR12R1



## ☐ Registre DAC\_DHR12L1

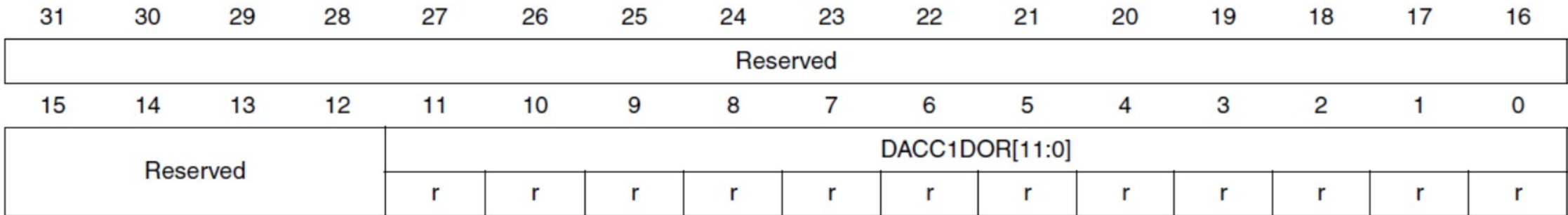


## ☐ Registre DAC\_DHR8R1



## ❑ Registre DAC\_DOR (DATA OUTPUT REGISTER)

Reset value: 0x0000 0000



Bits 31:12 Reserved.

Bit 11:0 **DACC1DOR[11:0]**: DAC channel1 data output

These bits are read-only, they contain data output for DAC channel1.