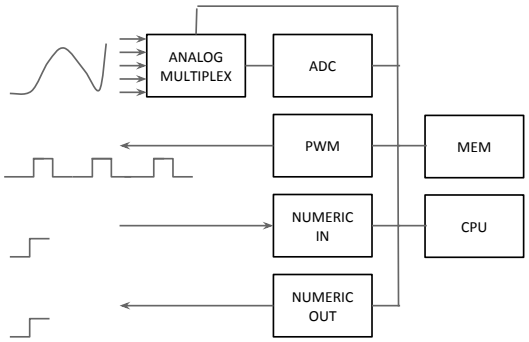
Arduino
signaux analogiques

Place de l'analogique

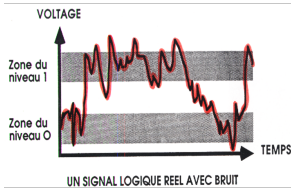


Signal numérique

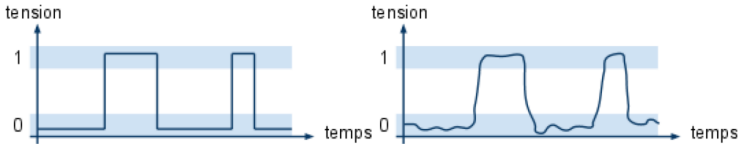
Un signal numérique se caractérise par deux valeurs de tension, soit en logique positive:

- 1 (5V ou 3.3V ou 1.8V)
- 0 (0V)

Dans la réalité le signal est bruité:

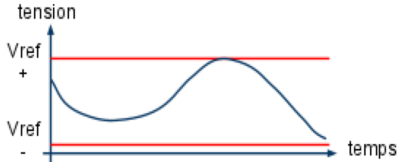


UN SIGNAL LOGIQUE REEL AVEC BRUIT



Signal analogique

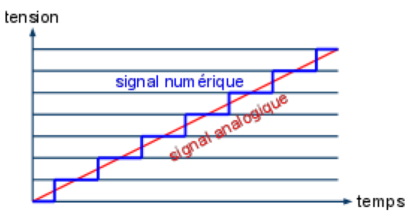
- Un signal analogique peut prendre toutes les valeurs de tension entre deux bornes:
  - Vref+ : tension maximale
  - Vref- : tension minimale
- Dans notre contexte Vref sera 5V et Vref- sera 0V

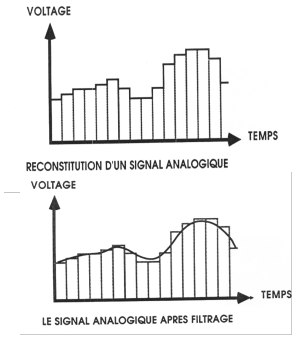


Conversion Numérique Analogique

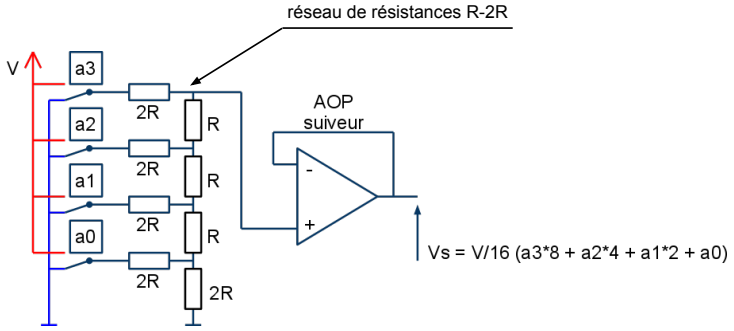
Signal Numérique sur N bits → Conversion Numérique Analogique → Signal Analogique sur 1 fil

La quantification se traduit par un effet escalier que l'on peut atténuer par filtrage.





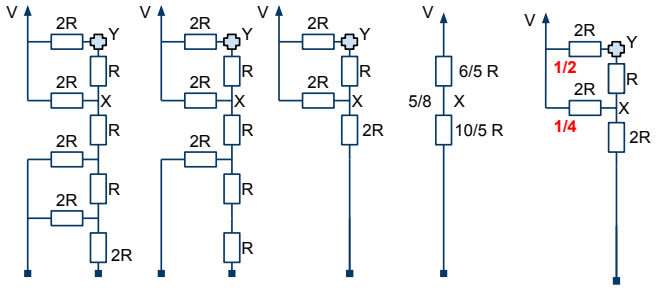
Convertisseur Numérique Analogique

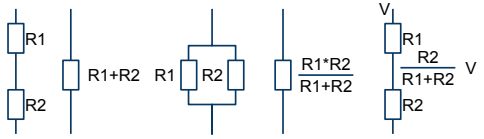


Exemple :

- Si tous les bits sont à 0 alors l'entrée + est à 0
- Si a3 est à V et a2, a1, a0 sont à 0, l'entrée + est à V/2

Un peu d'électronique....



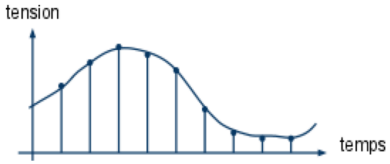


Y s'exprime en Volt
Y = V(X + 1/3 (1-X))
Y = V(5/8 + 1/3 (1 - 5/8))
Y = V(5/8 + 1/3 (3/8))
Y = 6/8 V
Y = 3/4 V
Y= (1/2 + 1/4)V

Conversion Analogique Numérique

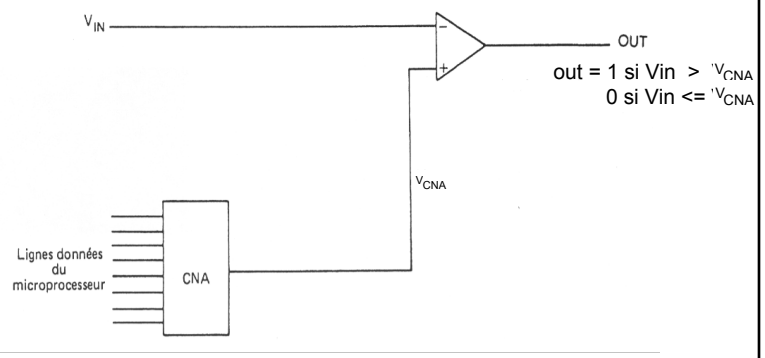
Signal Analogique sur 1 fil → Conversion Analogique Numérique → Signal Numérique sur N bits

Le conversion entraine une quantification, par exemple si le signal analogique est compris entre 0 et 5 volts, si le signal numérique est sur 8 bits. On ne représentera numériquement que 256 valeurs sur l'infinité des valeurs analogiques comprises entre 0 et 5V.



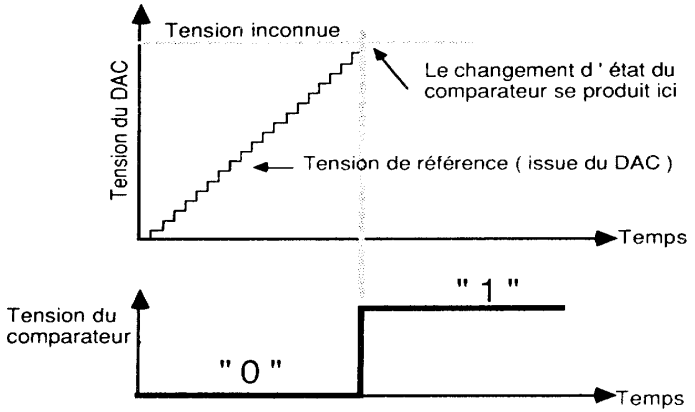
Conversion Analogique Numérique

On compare la tension analogique en entrée avec une valeur analogique produite par le processeur et recherche l'égalité



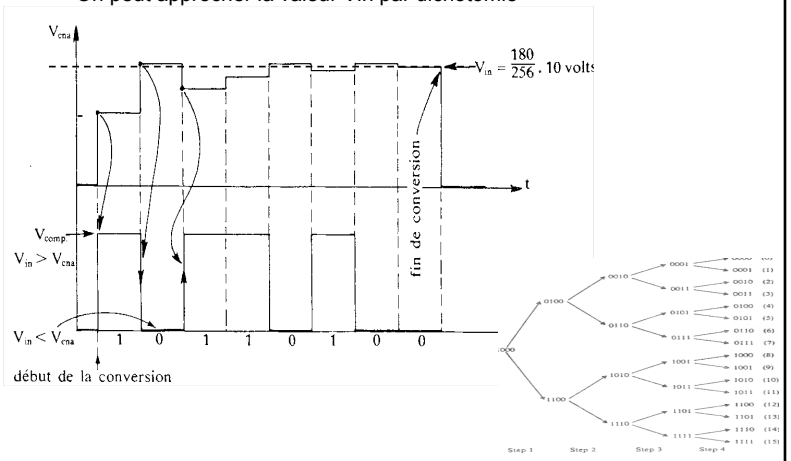
Principe de la conversion : rampe

Le processeur va calculer une valeur transformé par le CNA jusqu'à faire basculer le comparateur. On peut utiliser une simple rampe :

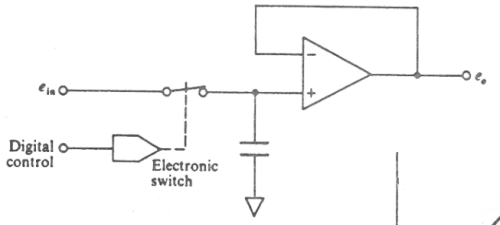


Principe de la conversion : dichotomie

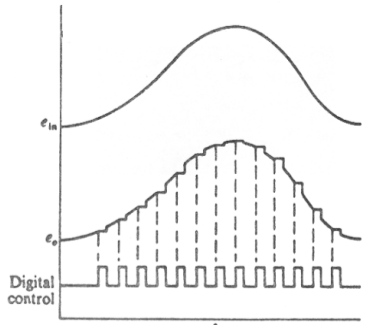
On peut approcher la valeur Vin par dichotomie



Acquisition de la valeur analogique

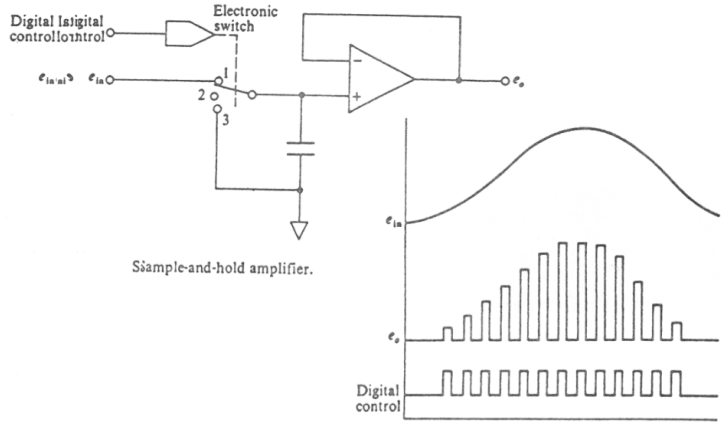


Track-and-hold amplifier.



Response curves and timing chart for track-and-hold amplifier.

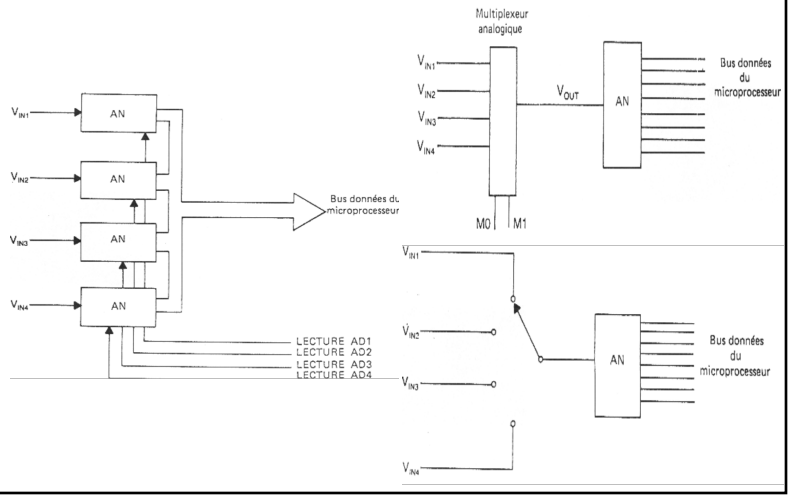
Acquisition de la valeur analogique



Sample-and-hold amplifier.

Response curves and timing chart for sample-and-hold amplifier.

Acquisition de la valeur analogique



caractéristiques ATmega ADC (extrait)

- 10-bit Resolution
- 1 LSB Integral Non-linearity
- ±2 LSB Absolute Accuracy
- 13µs - 260µs Conversion Time
- Up to 76.9kSPS (Up to 15kSPS at Maximum Resolution)
- 16 Multiplexed Single Ended Input Channels
- 14 Differential input channels (**comparaison de deux entrées)**
- Selectable 2.56V or 1.1V ADC Reference Voltage
- Free Running or Single Conversion Mode
- Interrupt on ADC Conversion Complete

API Arduino

anologReference(REF)

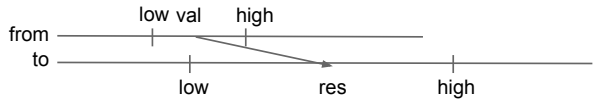
REF : DEFAULT, INTERNAL1V1, INTERNAL2V56, EXTERNAL

int analogREAD(int pin)

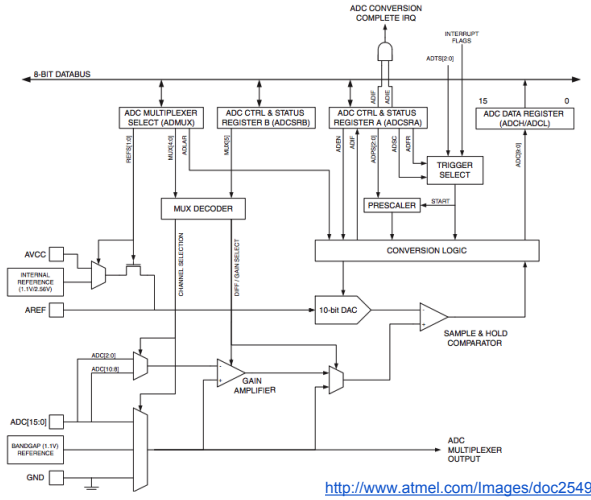
0 à 5 (uno) ou 0 à 15 (mega)

resultat sur 10bits 0 à 1023

int map(value, fromLow, fromHigh, toLow, toHigh)



ATmega ADC schematic



http://www.atmel.com/Images/doc2549.pdf p275

Fonctionnement

- Le composant ADC dispose de plusieurs modes de fonctionnement.
  - Pour la lecture un mode un-coup ou automatique.
  - Des comparateurs de tensions
  - Des activations d'interruptions
  - etc.
- Pour comprendre le fonctionnement détaillé, il faut lire la documentation du micro-contrôleur. Il est alors possible d'utiliser tous les modes disponibles.
- Pour commencer on peut regarder comment fonctionne analogRead() (http://garretlab.web.fc2.com/en/arduino/inside/arduino/wiring_analog.c/analogRead.html)

analogRead()

analogRead() utilise 4 registres de l'ADC:

- ADMUX
  - configure la référence de tension, la justification du résultat (gauche ou droite), la selection de l'entrée
- ADCSRA
  - ADC Control and Status Register A
  - commande la conversion et donne son statut
- ADCH - ADCL
  - High - Low
  - contiennent le résultat

ADMUX

ADMUX								
bit	7	6	5	4	3	2	1	0
name	REFS1	REFS0	ADLAR	-	MUX3	MUX2	MUX1	MUX0

REFS1	REFS0	Meanings	Argument of analogReference()
0	0	Voltage applied to AREF pin.	EXTERNAL(0)
0	1	Default referece voltage(5V in case of Arduino Uno).	DEFAULT(1)
1	0	Reserved.	-
1	1	Internal reference voltage(1.1V in case of Arduino Uno)	INTERNAL(3)

ADCSRA

ADCSRA								
bit	7	6	5	4	3	2	1	0
name	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0

ADEN Enable
← 1 allume le convertisseur (positionner au démarrage)

ADSC Start Conversion
← 1 démarre la conversion
repasse à 0 quand la conversion est terminée

ADATE Automatic Triger Enable
← 1 mode automatique (non utilisé par arduino)

ADIF Interrupt Flag (non utilisé par arduino)

ADIE Interrupt Enable (non utilisé par arduino)

ADPS Division d'horloge ...

ADPS

ADPS2	ADPS1	ADPS0	Division Factor
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

Division d'horloge du processeur pour réaliser la conversion.

Plus la fréquence est élevée moins bonne sera la précision (nb bits significatifs).
entre 50kHz et 200kHz, on a toute la précision.

Arduino : 111 ⇒ 16Mhz / 128 = 125kHz
la conversion prend 13 cycles ⇒ 13 * 1/125 kHz = 104µs
ou 125k/13 = 9600 échantillons par seconde

API readRead source

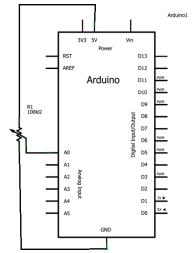
```
int analogRead(uint8_t pin)
{
        uint8_t low, high;

        if (pin >= 14) pin -= 14; // allow for channel or pin numbers

        // set the analog reference (high two bits of ADMUX) and select the
        // channel (low 4 bits).  this also sets ADLAR (left-adjust result)
        // to 0 (the default).
        ADMUX = (analog_reference << 6) | (pin & 0x07);

        // start the conversion
        sbi(ADCSRA, ADSC);

        // ADSC is cleared when the conversion finishes
        while (bit_is_set(ADCSRA, ADSC));

        // we have to read ADCL first; doing so locks both ADCL
        // and ADCH until ADCH is read.  reading ADCL second would
        // cause the results of each conversion to be discarded,
        // as ADCL and ADCH would be locked when it completed.
        low  = ADCL;
        high = ADCH;

        // combine the two bytes
        return (high << 8) | low;
}
```


TME

Serrure de coffre fort analogique

- lecture de valeurs successives d'un potentiomètre
- comparaison avec un code préprogrammé
- marque d'ouverture par affichage d'une led

On pourrait avoir un code maitre permettant de programmer un autre code.



http://eskimon.fr/