R2.07
Informatique embarquée

B. Jacquot/F. Morain-Nicolier/D. Penant

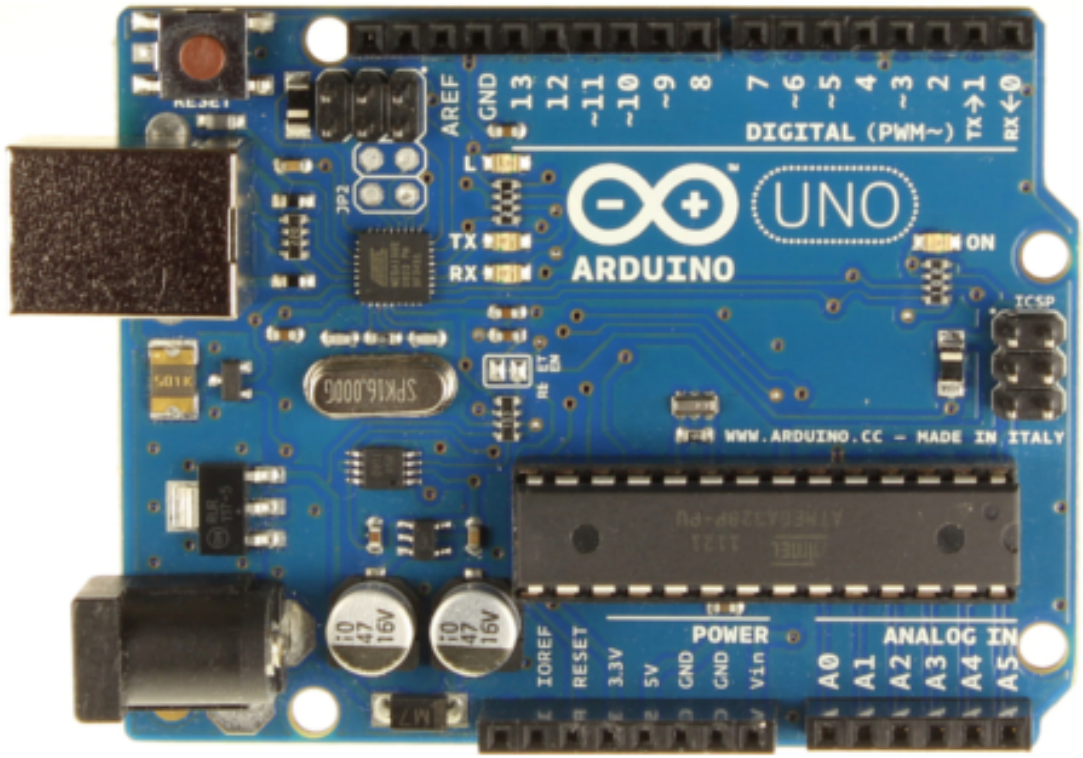
BUT1

Sommaire

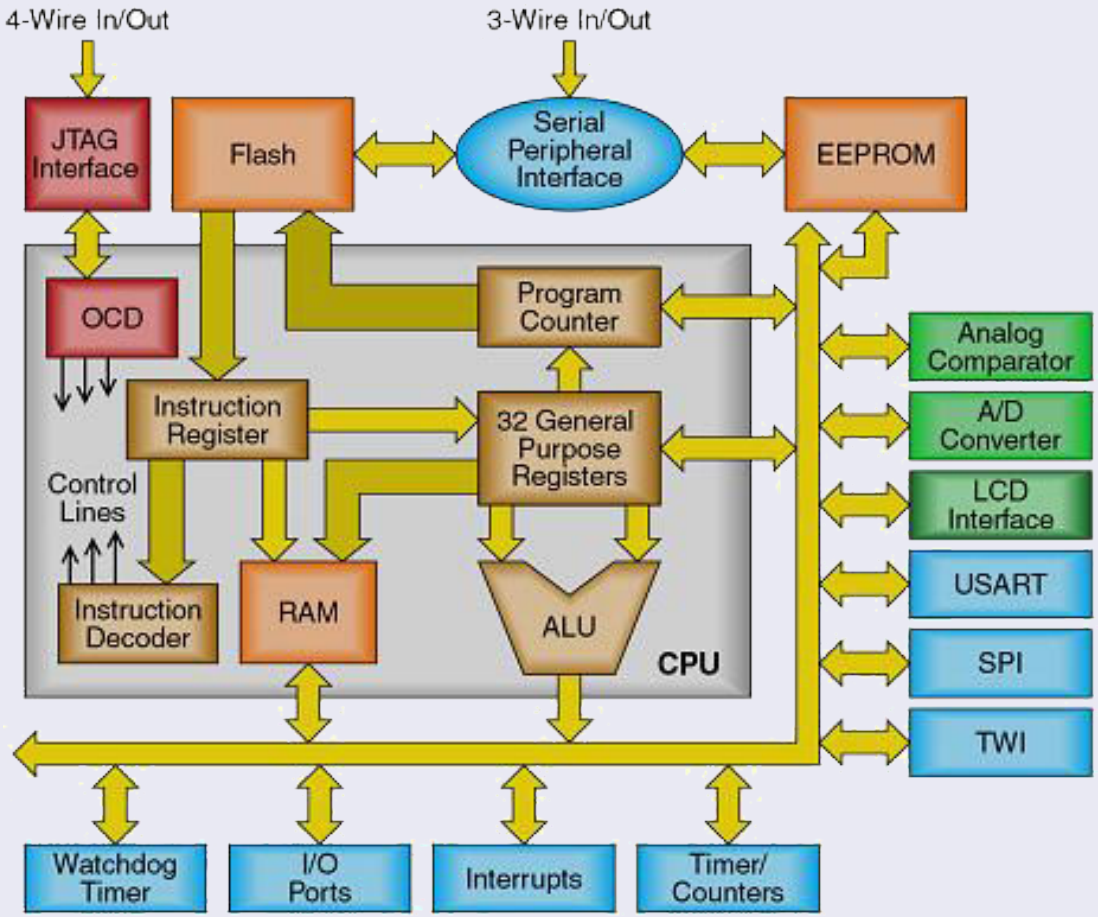
1. Introduction
2. Entrée, sortie ?
3. Modifier une sortie
4. Lire une entrée
5. Cas des bps
6. Gestion d'événement externe
7. Événement périodique
8. MLI/PWM
9. Les "modes" d'entrées
10. Transmission de données
11. Configuration du µcontrôleur
12. Choix microcontroleur

Atmel

ARDUINO



Architecture d'un µcontrôleur



- CPU
- bus de données (8 bits)
- Mémoires
- Périphériques

Sommaire

1 Introduction

2 Entrée, sortie ?

3 Modifier une sortie

4 Lire une entrée

5 Cas des bps

6 Gestion d'événement externe

7 Événement périodique

8 MLI/PWM

9 Les "modes" d'entrées

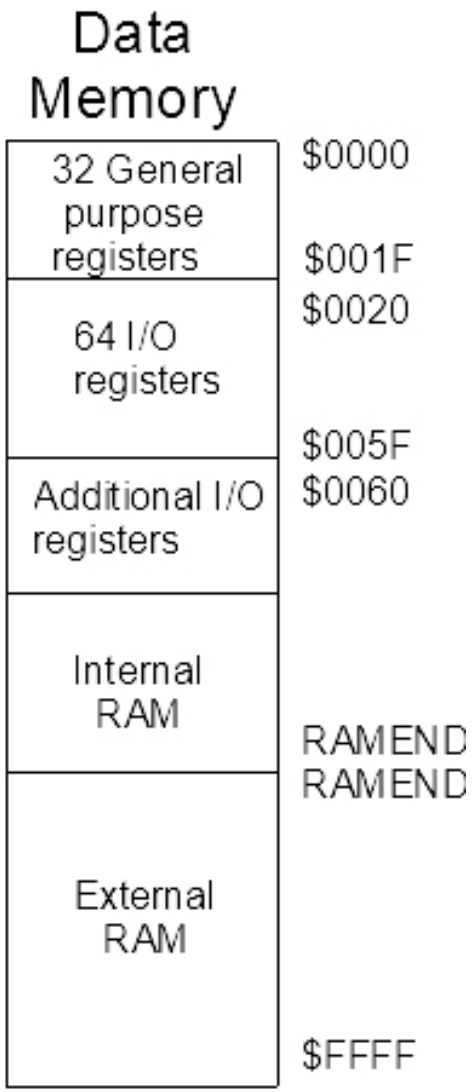
10 Transmission de données

11 Configuration du µcontrôleur

12 Choix microcontroleur

Sommaire

2 Entrée, sortie ?
- les registres
- registre DDRx



Un registre :

- est une "case" mémoire : c'est une @
- a une taille : 8 bits pour nous
- modifie/renseigne sur le fonctionnement
- ne peut être accessible bit/bit :
- Il faut modifier/lire le registre intégralement
- On utilisera les techniques étudiées ci-après

Exemple

Le registre suivant permet de configurer le périphérique USART0 (liaison série) d'un µcontroleur AVR. Ces informations proviennent de la datasheet du composant :

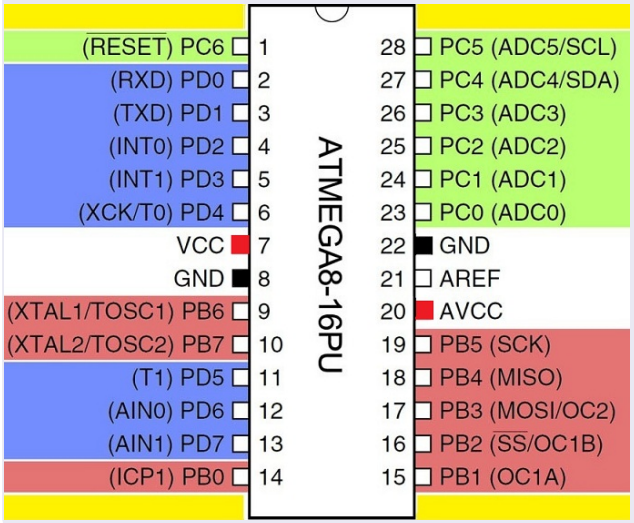
Bit	7	6	5	4	3	2	1	0
UCSR0A	RXC0	TXC0	UDRE0	FE0	DOR0	UOE0	U2X0	MCP0
Read/Write	R	R/W	R	R	R	R	R/W	R/W
Initial Value	0	0	1	0	0	0	0	0

La configuration de l'USART0 consistera principalement à modifier certains bits de ce registre.

Sommaire

2 Entrée, sortie ?
- les registres
- registre DDRx

IO port



- les pattes sont regroupées par port (8 bits)
- chaque port est défini par 3 registres
- le port n'est pas forcément sorti complètement
- chaque pin est configurable en entrée ou sortie

registre de direction DDRx

Pour chaque port, il existe un registre de direction DDRx

Chaque bit du port représente la configuration en e/s de la patte correspondante

On positionnera chaque bit du registre de la façon suivante :

0 entrée

1 sortie

exemple

Voici un exemple de configuration des e/s pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
e/s ?	:	e	e	s	s	e	s	e	e
DDRB	:								

registre de direction DDRx

Pour chaque port, il existe un registre de direction DDRx

Chaque bit du port représente la configuration en e/s de la patte correspondante

On positionnera chaque bit du registre de la façon suivante :

0 entrée

1 sortie

exemple

Voici un exemple de configuration des e/s pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
e/s ?	:	e	e	s	s	e	s	e	e
DDRB	:								0

registre de direction DDRx

Pour chaque port, il existe un registre de direction DDRx

Chaque bit du port représente la configuration en e/s de la patte correspondante

On positionnera chaque bit du registre de la façon suivante :

0 entrée

1 sortie

exemple

Voici un exemple de configuration des e/s pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
e/s ?	:	e	e	s	s	e	s	e	e
DDRB	:							0	0

registre de direction DDRx

Pour chaque port, il existe un registre de direction DDRx

Chaque bit du port représente la configuration en e/s de la patte correspondante

On positionnera chaque bit du registre de la façon suivante :

0 entrée

1 sortie

exemple

Voici un exemple de configuration des e/s pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
e/s ?	:	e	e	s	s	e	s	e	e
DDRB	:						1	0	0

registre de direction DDRx

Pour chaque port, il existe un registre de direction DDRx

Chaque bit du port représente la configuration en e/s de la patte correspondante

On positionnera chaque bit du registre de la façon suivante :

0 entrée

1 sortie

exemple

Voici un exemple de configuration des e/s pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
e/s ?	:	e	e	s	s	e	s	e	e
DDRB	:					0	1	0	0

registre de direction DDRx

Pour chaque port, il existe un registre de direction DDRx

Chaque bit du port représente la configuration en e/s de la patte correspondante

On positionnera chaque bit du registre de la façon suivante :

0 entrée

1 sortie

exemple

Voici un exemple de configuration des e/s pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
e/s ?	:	e	e	s	s	e	s	e	e
DDRB	:				1	0	1	0	0

registre de direction DDRx

Pour chaque port, il existe un registre de direction DDRx

Chaque bit du port représente la configuration en e/s de la patte correspondante

On positionnera chaque bit du registre de la façon suivante :

0 entrée

1 sortie

exemple

Voici un exemple de configuration des e/s pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
e/s ?	:	e	e	s	s	e	s	e	e
DDRB	:	0	0	1	1	0	1	0	0

registre de direction DDRx

Pour chaque port, il existe un registre de direction DDRx

Chaque bit du port représente la configuration en e/s de la patte correspondante

On positionnera chaque bit du registre de la façon suivante :

0 entrée

1 sortie

exemple

Voici un exemple de configuration des e/s pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
e/s ?	:	e	e	s	s	e	s	e	e
DDRB	:	0	0	1	1	0	1	0	0

Ce qui donnera le code suivant :

DDRB = 0b00110100 ;

registre de direction DDRx

Pour chaque port, il existe un registre de direction DDRx

Chaque bit du port représente la configuration en e/s de la patte correspondante

On positionnera chaque bit du registre de la façon suivante :

0 entrée

1 sortie

exemple

Voici un exemple de configuration des e/s pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
e/s ?	:	e	e	s	s	e	s	e	e
DDRB	:	0	0	1	1	0	1	0	0

Ce qui donnera le code suivant :

DDRB = 0b00110100 ; ou DDRB = 0x34 ;

registre de direction DDRx

Pour chaque port, il existe un registre de direction DDRx
Chaque bit du port représente la configuration en e/s de la patte correspondante
On positionnera chaque bit du registre de la façon suivante :

0 entrée
1 sortie

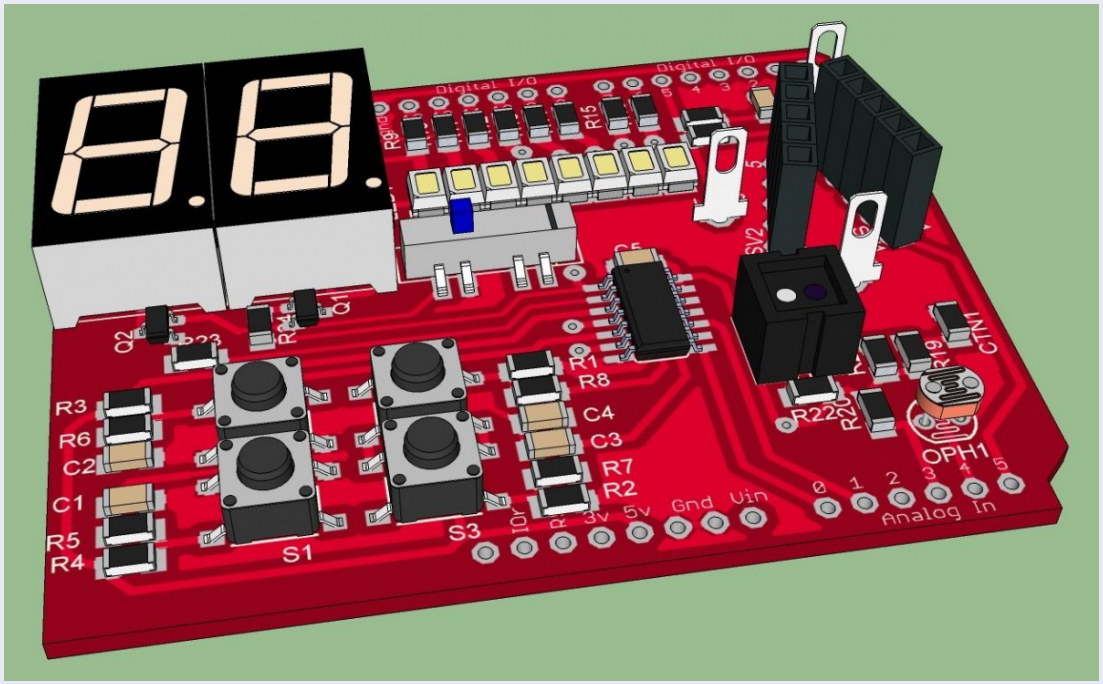
exemple

Voici un exemple de configuration des e/s pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
e/s ?	:	e	e	s	s	e	s	e	e
DDRB	:	0	0	1	1	0	1	0	0

Ce qui donnera le code suivant :
DDRB = 0b00110100 ; ou DDRB = 0x34 ; ou encore DDRB = 52 ;

Exemple



Couleur	r	o	v	r	o	v	v	r
Port	PB5	PB4	PB3	PB2	PB1	PB0	PD7	PD6

Sommaire

1 Introduction

2 Entrée, sortie ?

3 Modifier une sortie

4 Lire une entrée

5 Cas des bps

6 Gestion d'événement externe

7 Événement périodique

8 MLI/PWM

9 Les "modes" d'entrées

10 Transmission de données

11 Configuration du µcontrôleur

12 Choix microcontroleur

Sommaire

3 Modifier une sortie
- registre PORTx
- Décalage
- Inversion de bit
- Forçage à 1
- Complément à 1
- Forçage à 0

registre de données PORTx

Pour chaque port, il existe un registre de données PORTx
Chaque bit du port représente la valeur de la patte correspondante pour les broches en sortie.
(L'usage pour les pattes configurées en entrées sera expliqué plus tard.)
On positionnera chaque bit du registre de la façon suivante :

0	tension de sortie 0V
1	tension de sortie VCC

exemple

Voici un exemple pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
e/s	:	e	e	s	s	e	s	e	e
valeur désirée	:	x	x	1	1	x	0	x	x
PORTB	:								

registre de données PORTx

Pour chaque port, il existe un registre de données PORTx
Chaque bit du port représente la valeur de la patte correspondante pour les broches en sortie.
(L'usage pour les pattes configurées en entrées sera expliqué plus tard.)
On positionnera chaque bit du registre de la façon suivante :

0	tension de sortie 0V
1	tension de sortie VCC

exemple

Voici un exemple pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
e/s	:	e	e	s	s	e	s	e	e
valeur désirée	:	x	x	1	1	x	0	x	x
PORTB	:								0

registre de données PORTx

Pour chaque port, il existe un registre de données PORTx
Chaque bit du port représente la valeur de la patte correspondante pour les broches en sortie.
(L'usage pour les pattes configurées en entrées sera expliqué plus tard.)
On positionnera chaque bit du registre de la façon suivante :

0	tension de sortie 0V
1	tension de sortie VCC

exemple

Voici un exemple pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
e/s	:	e	e	s	s	e	s	e	e
valeur désirée	:	x	x	1	1	x	0	x	x
PORTB	:							0	0

registre de données PORTx

Pour chaque port, il existe un registre de données PORTx
Chaque bit du port représente la valeur de la patte correspondante pour les broches en sortie.
(L'usage pour les pattes configurées en entrées sera expliqué plus tard.)
On positionnera chaque bit du registre de la façon suivante :

0	tension de sortie 0V
1	tension de sortie VCC

exemple

Voici un exemple pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
e/s	:	e	e	s	s	e	s	e	e
valeur désirée	:	x	x	1	1	x	0	x	x
PORTB	:						0	0	0

registre de données PORTx

Pour chaque port, il existe un registre de données PORTx
Chaque bit du port représente la valeur de la patte correspondante pour les broches en sortie.
(L'usage pour les pattes configurées en entrées sera expliqué plus tard.)
On positionnera chaque bit du registre de la façon suivante :

0	tension de sortie 0V
1	tension de sortie VCC

exemple

Voici un exemple pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
e/s	:	e	e	s	s	e	s	e	e
valeur désirée	:	x	x	1	1	x	0	x	x
PORTB	:					0	0	0	0

registre de données PORTx

Pour chaque port, il existe un registre de données PORTx
Chaque bit du port représente la valeur de la patte correspondante pour les broches en sortie.
(L'usage pour les pattes configurées en entrées sera expliqué plus tard.)
On positionnera chaque bit du registre de la façon suivante :

0	tension de sortie 0V
1	tension de sortie VCC

exemple

Voici un exemple pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
e/s	:	e	e	s	s	e	s	e	e
valeur désirée	:	x	x	1	1	x	0	x	x
PORTB	:				1	0	0	0	0

registre de données PORTx

Pour chaque port, il existe un registre de données PORTx
Chaque bit du port représente la valeur de la patte correspondante pour les broches en sortie.
(L'usage pour les pattes configurées en entrées sera expliqué plus tard.)
On positionnera chaque bit du registre de la façon suivante :

0	tension de sortie 0V
1	tension de sortie VCC

exemple

Voici un exemple pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
e/s	:	e	e	s	s	e	s	e	e
valeur désirée	:	x	x	1	1	x	0	x	x
PORTB	:	0	0	1	1	0	0	0	0

registre de données PORTx

Pour chaque port, il existe un registre de données PORTx
Chaque bit du port représente la valeur de la patte correspondante pour les broches en sortie.
(L'usage pour les pattes configurées en entrées sera expliqué plus tard.)
On positionnera chaque bit du registre de la façon suivante :

0	tension de sortie 0V
1	tension de sortie VCC

exemple

Voici un exemple pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
e/s	:	e	e	s	s	e	s	e	e
valeur désirée	:	x	x	1	1	x	0	x	x
PORTB	:	0	0	1	1	0	0	0	0

Ce qui donnera le code suivant :

```
PORTB = 0b00110000;
```

registre de données PORTx

Pour chaque port, il existe un registre de données PORTx
Chaque bit du port représente la valeur de la patte correspondante pour les broches en sortie.
(L'usage pour les pattes configurées en entrées sera expliqué plus tard.)
On positionnera chaque bit du registre de la façon suivante :

0	tension de sortie 0V
1	tension de sortie VCC

exemple

Voici un exemple pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
e/s	:	e	e	s	s	e	s	e	e
valeur désirée	:	x	x	1	1	x	0	x	x
PORTB	:	0	0	1	1	0	0	0	0

Ce qui donnera le code suivant :
PORTB = 0b00110000 ; ou PORTB = 0x30 ;

registre de données PORTx

Pour chaque port, il existe un registre de données PORTx
Chaque bit du port représente la valeur de la patte correspondante pour les broches en sortie.
(L'usage pour les pattes configurées en entrées sera expliqué plus tard.)
On positionnera chaque bit du registre de la façon suivante :

0	tension de sortie 0V
1	tension de sortie VCC

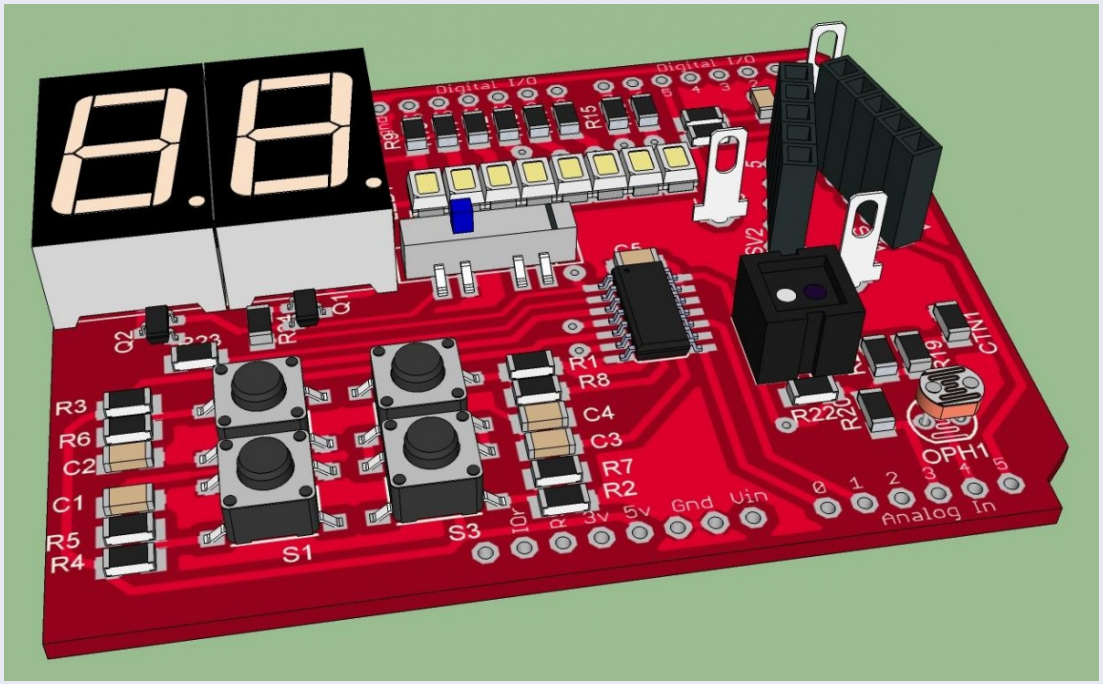
exemple

Voici un exemple pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
e/s	:	e	e	s	s	e	s	e	e
valeur désirée	:	x	x	1	1	x	0	x	x
PORTB	:	0	0	1	1	0	0	0	0

Ce qui donnera le code suivant :
PORTB = 0b00110000 ; ou PORTB = 0x30 ; ou encore DDRB = 48 ;

Exemple



Couleur	r	o	v	r	o	v	v	r
Port	PB5	PB4	PB3	PB2	PB1	PB0	PD7	PD6

Sommaire

3 Modifier une sortie
- registre PORTx
- Décalage
- Inversion de bit
- Forçage à 1
- Complément à 1
- Forçage à 0

Décalage : » ou « (chevron ou crochet oblique)

« / » : décalage à gauche/droite de tous les bits
les bits qui "sortent" sont perdus, les bits qui "rentrent" sont des '0'

X	:	b7	b6	b5	b4	b3	b2	b1	b0
X » 3	:	0	0	0	b7	b6	b5	b4	b3

Cas particulier : si X est un nombre signé, lors d'un décalage à droite les bits qui "rentrent" ont pour valeur le bit de poids fort avant décalage.

Ex :

X	:	1	1	1	0	1	1	0	1
X « 2	:								

Décalage : » ou « (chevron ou crochet oblique)

« / » : décalage à gauche/droite de tous les bits
les bits qui "sortent" sont perdus, les bits qui "rentrent" sont des '0'

X	:	b7	b6	b5	b4	b3	b2	b1	b0
X » 3	:	0	0	0	b7	b6	b5	b4	b3

Cas particulier : si X est un nombre signé, lors d'un décalage à droite les bits qui "rentrent" ont pour valeur le bit de poids fort avant décalage.

Ex :

X	:	1	1	1	0	1	1	0	1
X « 2	:						1		

Décalage : » ou « (chevron ou crochet oblique)

« / » : décalage à gauche/droite de tous les bits
les bits qui "sortent" sont perdus, les bits qui "rentrent" sont des '0'

X	:	b7	b6	b5	b4	b3	b2	b1	b0
X » 3	:	0	0	0	b7	b6	b5	b4	b3

Cas particulier : si X est un nombre signé, lors d'un décalage à droite les bits qui "rentrent" ont pour valeur le bit de poids fort avant décalage.

Ex :

X	:	1	1	1	0	1	1	0	1
X « 2	:					0	1		

Décalage : » ou « (chevron ou crochet oblique)

« / » : décalage à gauche/droite de tous les bits
les bits qui "sortent" sont perdus, les bits qui "rentrent" sont des '0'

X	:	b7	b6	b5	b4	b3	b2	b1	b0
X » 3	:	0	0	0	b7	b6	b5	b4	b3

Cas particulier : si X est un nombre signé, lors d'un décalage à droite les bits qui "rentrent" ont pour valeur le bit de poids fort avant décalage.

Ex :

X	:	1	1	1	0	1	1	0	1
X « 2	:				1	0	1		

Décalage : » ou « (chevron ou crochet oblique)

« / » : décalage à gauche/droite de tous les bits
les bits qui "sortent" sont perdus, les bits qui "rentrent" sont des '0'

X	:	b7	b6	b5	b4	b3	b2	b1	b0
X » 3	:	0	0	0	b7	b6	b5	b4	b3

Cas particulier : si X est un nombre signé, lors d'un décalage à droite les bits qui "rentrent" ont pour valeur le bit de poids fort avant décalage.

Ex :

X	:	1	1	1	0	1	1	0	1
X « 2	:	1	0	1	1	0	1		

Décalage : » ou « (chevron ou crochet oblique)

« / » : décalage à gauche/droite de tous les bits
les bits qui "sortent" sont perdus, les bits qui "rentrent" sont des '0'

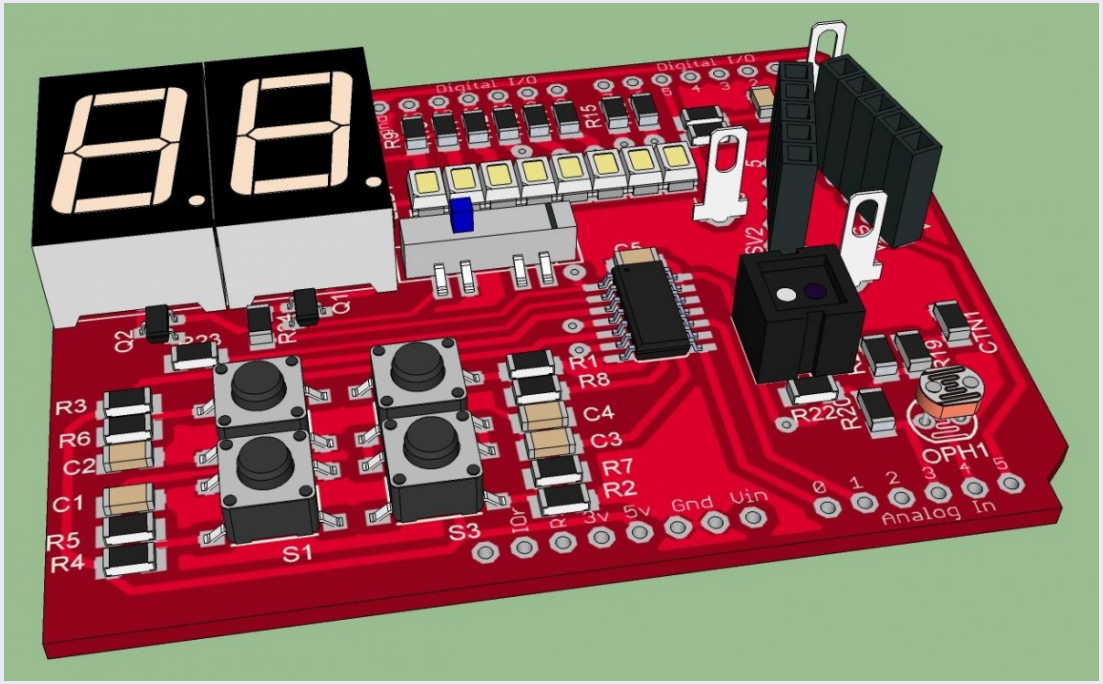
X	:	b7	b6	b5	b4	b3	b2	b1	b0
X » 3	:	0	0	0	b7	b6	b5	b4	b3

Cas particulier : si X est un nombre signé, lors d'un décalage à droite les bits qui "rentrent" ont pour valeur le bit de poids fort avant décalage.

Ex :

X	:	1	1	1	0	1	1	0	1
X « 2	:	1	0	1	1	0	1	0	0

Chenillard



Couleur	r	o	v	r	o	v	v	r
Port	PB5	PB4	PB3	PB2	PB1	PB0	PD7	PD6

Sommaire

3 Modifier une sortie
- registre PORTx
- Décalage
- Inversion de bit
- Forçage à 1
- Complément à 1
- Forçage à 0

OU exclusif bit à bit (bitwise xor) : ∧(caret, circonflexe)

Réalise un OU exclusif logique entre chaque paire de bits de même poids.

X	:	x7	x6	x5	x4	x3	x2	x1	x0
Y	:	y7	y6	y5	y4	y3	y2	y1	y0
X∧Y	:	x7⊕y7	x6⊕y6	x5⊕y5	x4⊕y4	x3⊕y3	x2⊕y2	x1⊕y1	x0⊕y0

Ex :

X	:	1	1	0	0	1	1	0	1
Y	:	1	0	0	1	1	0	0	1
X∧Y	:								

OU exclusif bit à bit (bitwise xor) : ∧(caret, circonflexe)

Réalise un OU exclusif logique entre chaque paire de bits de même poids.

X	:	x7	x6	x5	x4	x3	x2	x1	x0
Y	:	y7	y6	y5	y4	y3	y2	y1	y0
X∧Y	:	x7⊕y7	x6⊕y6	x5⊕y5	x4⊕y4	x3⊕y3	x2⊕y2	x1⊕y1	x0⊕y0

Ex :

X	:	1	1	0	0	1	1	0	1
Y	:	1	0	0	1	1	0	0	1
X∧Y	:								0

OU exclusif bit à bit (bitwise xor) : ∧(caret, circonflexe)

Réalise un OU exclusif logique entre chaque paire de bits de même poids.

X	:	x7	x6	x5	x4	x3	x2	x1	x0
Y	:	y7	y6	y5	y4	y3	y2	y1	y0
X∧Y	:	x7⊕y7	x6⊕y6	x5⊕y5	x4⊕y4	x3⊕y3	x2⊕y2	x1⊕y1	x0⊕y0

Ex :

X	:	1	1	0	0	1	1	0	1
Y	:	1	0	0	1	1	0	0	1
X∧Y	:							0	0

OU exclusif bit à bit (bitwise xor) : ∧(caret, circonflexe)

Réalise un OU exclusif logique entre chaque paire de bits de même poids.

X	:	x7	x6	x5	x4	x3	x2	x1	x0
Y	:	y7	y6	y5	y4	y3	y2	y1	y0
X∧Y	:	x7⊕y7	x6⊕y6	x5⊕y5	x4⊕y4	x3⊕y3	x2⊕y2	x1⊕y1	x0⊕y0

Ex :

X	:	1	1	0	0	1	1	0	1
Y	:	1	0	0	1	1	0	0	1
X∧Y	:						1	0	0

OU exclusif bit à bit (bitwise xor) : ∧(caret, circonflexe)

Réalise un OU exclusif logique entre chaque paire de bits de même poids.

X	:	x7	x6	x5	x4	x3	x2	x1	x0
Y	:	y7	y6	y5	y4	y3	y2	y1	y0
X∧Y	:	x7⊕y7	x6⊕y6	x5⊕y5	x4⊕y4	x3⊕y3	x2⊕y2	x1⊕y1	x0⊕y0

Ex :

X	:	1	1	0	0	1	1	0	1
Y	:	1	0	0	1	1	0	0	1
X∧Y	:	0	1	0	1	0	1	0	0

Usage du $\wedge$: Inverser la valeur d'un bit

On utilise un "masque" dans lequel tous les bits sont à 0, sauf celui/ceux que l'on souhaite inverser.

X	:	x7	x6	x5	x4	x3	x2	x1	x0
m	:	0	0	1	0	0	0	0	0
X $\wedge$m	:	x7	x6	!x5	x4	x3	x2	x1	x0

On écrira : X = X $\wedge$m ; ou encore : X $\wedge=$ m ;

Ex :

X	:	1	1	1	0	1	1	0	1
m	:	0	1	0	0	0	0	0	0
X$\wedge$m	:								

Usage du $\wedge$: Inverser la valeur d'un bit

On utilise un "masque" dans lequel tous les bits sont à 0, sauf celui/ceux que l'on souhaite inverser.

X	:	x7	x6	x5	x4	x3	x2	x1	x0
m	:	0	0	1	0	0	0	0	0
X $\wedge$m	:	x7	x6	!x5	x4	x3	x2	x1	x0

On écrira : X = X $\wedge$m ; ou encore : X $\wedge=$ m ;

Ex :

X	:	1	1	1	0	1	1	0	1
m	:	0	1	0	0	0	0	0	0
X$\wedge$m	:								1

Usage du $\wedge$: Inverser la valeur d'un bit

On utilise un "masque" dans lequel tous les bits sont à 0, sauf celui/ceux que l'on souhaite inverser.

X	:	x7	x6	x5	x4	x3	x2	x1	x0
m	:	0	0	1	0	0	0	0	0
X $\wedge$m	:	x7	x6	!x5	x4	x3	x2	x1	x0

On écrira : X = X $\wedge$m ; ou encore : X $\wedge=$ m ;

Ex :

X	:	1	1	1	0	1	1	0	1
m	:	0	1	0	0	0	0	0	0
X$\wedge$m	:							0	1

Usage du $\wedge$: Inverser la valeur d'un bit

On utilise un "masque" dans lequel tous les bits sont à 0, sauf celui/ceux que l'on souhaite inverser.

X	:	x7	x6	x5	x4	x3	x2	x1	x0
m	:	0	0	1	0	0	0	0	0
X $\wedge$m	:	x7	x6	!x5	x4	x3	x2	x1	x0

On écrira : X = X $\wedge$m ; ou encore : X $\wedge=$ m ;

Ex :

X	:	1	1	1	0	1	1	0	1
m	:	0	1	0	0	0	0	0	0
X$\wedge$m	:			1	0	1	1	0	1

Usage du $\wedge$: Inverser la valeur d'un bit

On utilise un "masque" dans lequel tous les bits sont à 0, sauf celui/ceux que l'on souhaite inverser.

X	:	x7	x6	x5	x4	x3	x2	x1	x0
m	:	0	0	1	0	0	0	0	0
X $\wedge$m	:	x7	x6	!x5	x4	x3	x2	x1	x0

On écrira : X = X $\wedge$m ; ou encore : X $\wedge=$ m ;

Ex :

X	:	1	1	1	0	1	1	0	1
m	:	0	1	0	0	0	0	0	0
X$\wedge$m	:		0	1	0	1	1	0	1

Usage du $\wedge$: Inverser la valeur d'un bit

On utilise un "masque" dans lequel tous les bits sont à 0, sauf celui/ceux que l'on souhaite inverser.

X	:	x7	x6	x5	x4	x3	x2	x1	x0
m	:	0	0	1	0	0	0	0	0
X $\wedge$m	:	x7	x6	!x5	x4	x3	x2	x1	x0

On écrira : X = X $\wedge$m ; ou encore : X $\wedge=$ m ;

Ex :

X	:	1	1	1	0	1	1	0	1
m	:	0	1	0	0	0	0	0	0
X$\wedge$m	:	1	0	1	0	1	1	0	1

Inverser un bit

Pour inverser le bit 4 d'une variable, on choisira comme masque

m : 0 0 0 1 0 0 0 0

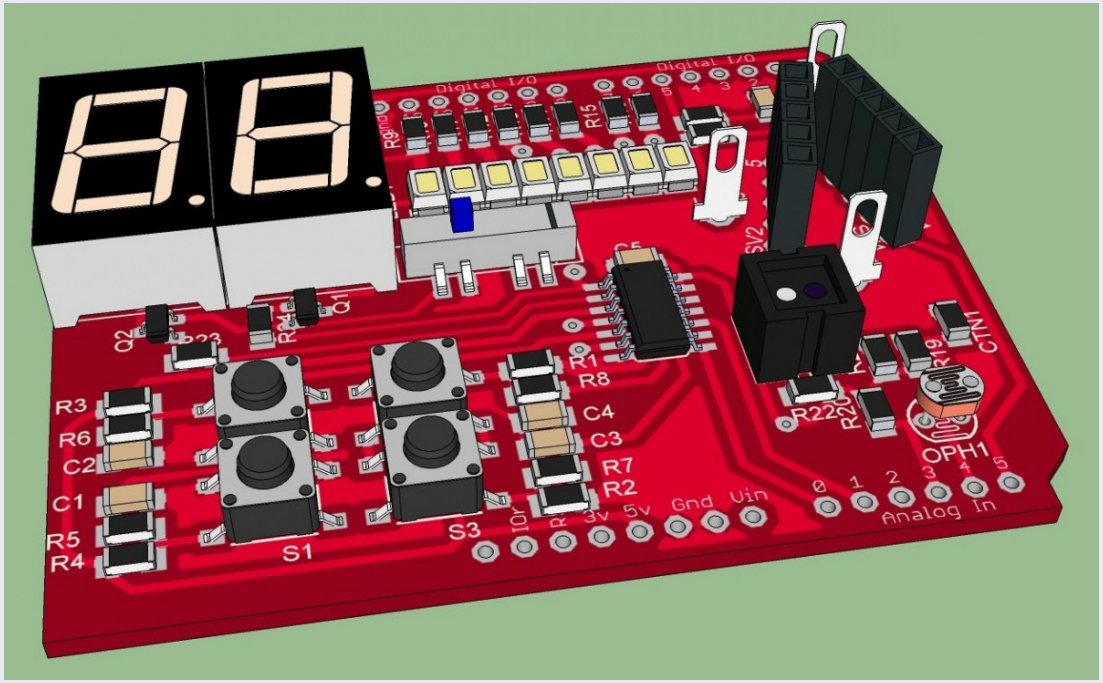
Remarquons que 0b00010000 = 1«4
Pour inverser le bit 4 du registre PORTB (changer l'état de PB4 !) :
PORTB = PORTB ∧(1 « 4) ; ou encore PORTB ∧= 1 « 4 ;

PORTB ∧= 1 « PB4 ;

Plusieurs bits

Pour modifier les sorties des broches PA3 et PA5 :
PORTA ∧= (1 « PA3) | (1 « PA5) ;

Clignotement de led



Couleur	r	o	v	r	o	v	v	r
Port	PB5	PB4	PB3	PB2	PB1	PB0	PD7	PD6

Sommaire

3 Modifier une sortie
- registre PORTx
- Décalage
- Inversion de bit
- Forçage à 1
- Complément à 1
- Forçage à 0

OU bit à bit (bitwise or) : | (barre verticale, tube ou pipe)

Réalise un OU logique entre chaque paire de bits de même poids.

X	:	x7	x6	x5	x4	x3	x2	x1	x0
Y	:	y7	y6	y5	y4	y3	y2	y1	y0
X|Y	:	x7+y7	x6+y6	x5+y5	x4+y4	x3+y3	x2+y2	x1+y1	x0+y0

Ex :

X	:	1	1	0	0	1	1	0	1
Y	:	1	0	0	1	1	0	0	1
X | Y	:								

OU bit à bit (bitwise or) : | (barre verticale, tube ou pipe)

Réalise un OU logique entre chaque paire de bits de même poids.

X	:	x7	x6	x5	x4	x3	x2	x1	x0
Y	:	y7	y6	y5	y4	y3	y2	y1	y0
X|Y	:	x7+y7	x6+y6	x5+y5	x4+y4	x3+y3	x2+y2	x1+y1	x0+y0

Ex :

X	:	1	1	0	0	1	1	0	1
Y	:	1	0	0	1	1	0	0	1
X | Y	:								1

OU bit à bit (bitwise or) : | (barre verticale, tube ou pipe)

Réalise un OU logique entre chaque paire de bits de même poids.

X	:	x7	x6	x5	x4	x3	x2	x1	x0
Y	:	y7	y6	y5	y4	y3	y2	y1	y0
X|Y	:	x7+y7	x6+y6	x5+y5	x4+y4	x3+y3	x2+y2	x1+y1	x0+y0

Ex :

X	:	1	1	0	0	1	1	0	1
Y	:	1	0	0	1	1	0	0	1
X | Y	:							0	1

OU bit à bit (bitwise or) : | (barre verticale, tube ou pipe)

Réalise un OU logique entre chaque paire de bits de même poids.

X	:	x7	x6	x5	x4	x3	x2	x1	x0
Y	:	y7	y6	y5	y4	y3	y2	y1	y0
X|Y	:	x7+y7	x6+y6	x5+y5	x4+y4	x3+y3	x2+y2	x1+y1	x0+y0

Ex :

X	:	1	1	0	0	1	1	0	1
Y	:	1	0	0	1	1	0	0	1
X | Y	:						1	0	1

OU bit à bit (bitwise or) : | (barre verticale, tube ou pipe)

Réalise un OU logique entre chaque paire de bits de même poids.

X	:	x7	x6	x5	x4	x3	x2	x1	x0
Y	:	y7	y6	y5	y4	y3	y2	y1	y0
X|Y	:	x7+y7	x6+y6	x5+y5	x4+y4	x3+y3	x2+y2	x1+y1	x0+y0

Ex :

X	:	1	1	0	0	1	1	0	1
Y	:	1	0	0	1	1	0	0	1
X | Y	:	1	1	0	1	1	1	0	1

Usage du | : mettre un bit à 1 -> Forcer un bit à 1

On utilise un "masque" dans lequel tous les bits sont à 0, sauf celui/ceux que l'on souhaite mettre à 1.

PORTB	:	1	1	1	1	0	0	0	0
m	:	0	0	0	0	1	0	0	0
PORTB | m	:	1	1	1	1	1	0	0	0

Pour mettre la sortie PB3 à 1 :

```
PORTB = PORTB | 0b00001000;
PORTB = PORTB | (1«3);
PORTB |= (1«PB3);
```

Plusieurs bits

Pour mettre à 1 les sorties des broches PA3 et PA5 :

```
PORTA |= (1 « PA3) | (1 « PA5);
```

Sommaire

3 Modifier une sortie
- registre PORTx
- Décalage
- Inversion de bit
- Forçage à 1
- Complément à 1
- Forçage à 0

Complément à un : ∼(tilde)

Le complément à un d'un nombre binaire est :
la valeur obtenue en inversant tous les bits de ce nombre.

X	:	b7	b6	b5	b4	b3	b2	b1	b0
∼X	:	!b7	!b6	!b5	!b4	!b3	!b2	!b1	!b0

Ex :

X	:	1	1	0	0	1	1	0	1
∼X	:								

Complément à un : ∼(tilde)

Le complément à un d'un nombre binaire est :
la valeur obtenue en inversant tous les bits de ce nombre.

X	:	b7	b6	b5	b4	b3	b2	b1	b0
∼X	:	!b7	!b6	!b5	!b4	!b3	!b2	!b1	!b0

Ex :

X	:	1	1	0	0	1	1	0	1
∼X	:								0

Complément à un : ∼(tilde)

Le complément à un d'un nombre binaire est :
la valeur obtenue en inversant tous les bits de ce nombre.

X	:	b7	b6	b5	b4	b3	b2	b1	b0
∼X	:	!b7	!b6	!b5	!b4	!b3	!b2	!b1	!b0

Ex :

X	:	1	1	0	0	1	1	0	1
∼X	:							1	0

Complément à un : ∼(tilde)

Le complément à un d'un nombre binaire est :
la valeur obtenue en inversant tous les bits de ce nombre.

X	:	b7	b6	b5	b4	b3	b2	b1	b0
∼X	:	!b7	!b6	!b5	!b4	!b3	!b2	!b1	!b0

Ex :

X	:	1	1	0	0	1	1	0	1
∼X	:	0	0	1	1	0	0	1	0

Sommaire

3 Modifier une sortie
- registre PORTx
- Décalage
- Inversion de bit
- Forçage à 1
- Complément à 1
- Forçage à 0

Usage du & : mettre un bit à 0 -> Forcer un bit à 0

On utilise un "masque" dans lequel tous les bits sont à 1, sauf celui/ceux que l'on souhaite mettre à 0.

X	:	x7	x6	x5	x4	x3	x2	x1	x0
m	:	1	1	1	0	1	1	1	1
X & m	:	x7	x6	x5	0	x3	x2	x1	x0

On écrira : X = X & m ; ou encore : X &= m ;

Ex :

X	:	1	1	1	0	1	1	0	1
m	:	1	0	1	1	1	1	1	1
X&m	:								

Usage du & : mettre un bit à 0 -> Forcer un bit à 0

On utilise un "masque" dans lequel tous les bits sont à 1, sauf celui/ceux que l'on souhaite mettre à 0.

X	:	x7	x6	x5	x4	x3	x2	x1	x0
m	:	1	1	1	0	1	1	1	1
X & m	:	x7	x6	x5	0	x3	x2	x1	x0

On écrira : X = X & m ; ou encore : X &= m ;

Ex :

X	:	1	1	1	0	1	1	0	1
m	:	1	0	1	1	1	1	1	1
X&m	:								1

Usage du & : mettre un bit à 0 -> Forcer un bit à 0

On utilise un "masque" dans lequel tous les bits sont à 1, sauf celui/ceux que l'on souhaite mettre à 0.

X	:	x7	x6	x5	x4	x3	x2	x1	x0
m	:	1	1	1	0	1	1	1	1
X & m	:	x7	x6	x5	0	x3	x2	x1	x0

On écrira : X = X & m ; ou encore : X &= m ;

Ex :

X	:	1	1	1	0	1	1	0	1
m	:	1	0	1	1	1	1	1	1
X&m	:							0	1

Usage du & : mettre un bit à 0 -> Forcer un bit à 0

On utilise un "masque" dans lequel tous les bits sont à 1, sauf celui/ceux que l'on souhaite mettre à 0.

X	:	x7	x6	x5	x4	x3	x2	x1	x0
m	:	1	1	1	0	1	1	1	1
X & m	:	x7	x6	x5	0	x3	x2	x1	x0

On écrira : X = X & m ; ou encore : X &= m ;

Ex :

X	:	1	1	1	0	1	1	0	1
m	:	1	0	1	1	1	1	1	1
X&m	:			1	0	1	1	0	1

Usage du & : mettre un bit à 0 -> Forcer un bit à 0

On utilise un "masque" dans lequel tous les bits sont à 1, sauf celui/ceux que l'on souhaite mettre à 0.

X	:	x7	x6	x5	x4	x3	x2	x1	x0
m	:	1	1	1	0	1	1	1	1
X & m	:	x7	x6	x5	0	x3	x2	x1	x0

On écrira : X = X & m ; ou encore : X &= m ;

Ex :

X	:	1	1	1	0	1	1	0	1
m	:	1	0	1	1	1	1	1	1
X&m	:		0	1	0	1	1	0	1

Usage du & : mettre un bit à 0 -> Forcer un bit à 0

On utilise un "masque" dans lequel tous les bits sont à 1, sauf celui/ceux que l'on souhaite mettre à 0.

X	:	x7	x6	x5	x4	x3	x2	x1	x0
m	:	1	1	1	0	1	1	1	1
X & m	:	x7	x6	x5	0	x3	x2	x1	x0

On écrira : X = X & m ; ou encore : X &= m ;

Ex :

X	:	1	1	1	0	1	1	0	1
m	:	1	0	1	1	1	1	1	1
X&m	:	1	0	1	0	1	1	0	1

Mettre un bit à 0

Pour mettre à 0 le bit 4 d'une variable, on choisira comme masque

m : 1 1 1 0 1 1 1 1

Remarquons que 0b11101111 = ∼(0b00010000) = ∼(1«4)
Pour mettre à 0 le bit 4 du registre PORTB (mettre à 0 la sortie PB4) :
PORTB = PORTB & ∼(1 « 4) ; ou encore PORTB &= ∼(1 « 4) ;

PORTB &= ∼(1 « PB4) ;

Plusieurs bits

Pour mettre à 0 les sorties des broches PA3 et PA5 :
PORTA &= ∼((1 « PA3) | (1 « PA5)) ;

Sommaire

1 Introduction

2 Entrée, sortie ?

3 Modifier une sortie

4 Lire une entrée

5 Cas des bps

6 Gestion d'événement externe

7 Événement périodique

8 MLI/PWM

9 Les "modes" d'entrées

10 Transmission de données

11 Configuration du µcontrôleur

12 Choix microcontroleur

Sommaire

4 Lire une entrée
- Observer un bit
- registre PINx

Expresion booléenne

On effectue souvent dans un programme des actions conditionnelles, liées la plupart du temps à l'état d'un seul bit dans une variable.

ex d'utilisation

si (expression) {....}
sinon {....}

langage c / c++

expression non booléenne	expression booléenne
char n=10 ;	char n=10 ;
if (n)	if (n !=0)
{	{
....	....
}	}

Utilisons un masque

Il faut faire en sorte de "masquer" les bits qui ne nous intéressent pas.
On souhaite par exemple observer uniquement le bit 5 d'une variable X :

	X	:	x7	x6	x5	x4	x3	x2	x1	x0
	m	:								
r = X	m	:								

Utilisons un masque

Il faut faire en sorte de "masquer" les bits qui ne nous intéressent pas.
On souhaite par exemple observer uniquement le bit 5 d'une variable X :

	X	:	x7	x6	x5	x4	x3	x2	x1	x0
	m	:								
r = X	m	:			x5					

Utilisons un masque

Il faut faire en sorte de "masquer" les bits qui ne nous intéressent pas.
On souhaite par exemple observer uniquement le bit 5 d'une variable X :

	X	:	x7	x6	x5	x4	x3	x2	x1	x0
	m	:								
r = X	m	:			x5					0

Utilisons un masque

Il faut faire en sorte de "masquer" les bits qui ne nous intéressent pas.
On souhaite par exemple observer uniquement le bit 5 d'une variable X :

	X	:	x7	x6	x5	x4	x3	x2	x1	x0
	m	:								
r = X	m	:	0	0	x5	0	0	0	0	0

Utilisons un masque

Il faut faire en sorte de "masquer" les bits qui ne nous intéressent pas.
On souhaite par exemple observer uniquement le bit 5 d'une variable X :

X	:	x7	x6	x5	x4	x3	x2	x1	x0
m	:								0
r = X & m	:	0	0	x5	0	0	0	0	0

Utilisons un masque

Il faut faire en sorte de "masquer" les bits qui ne nous intéressent pas.
On souhaite par exemple observer uniquement le bit 5 d'une variable X :

X	:	x7	x6	x5	x4	x3	x2	x1	x0
m	:	0	0		0	0	0	0	0
r = X & m	:	0	0	x5	0	0	0	0	0

Utilisons un masque

Il faut faire en sorte de "masquer" les bits qui ne nous intéressent pas.
On souhaite par exemple observer uniquement le bit 5 d'une variable X :

X	:	x7	x6	x5	x4	x3	x2	x1	x0
m	:	0	0	1	0	0	0	0	0
r = X & m	:	0	0	x5	0	0	0	0	0

Utilisons un masque

Il faut faire en sorte de "masquer" les bits qui ne nous intéressent pas.
On souhaite par exemple observer uniquement le bit 5 d'une variable X :

X	:	x7	x6	x5	x4	x3	x2	x1	x0
m	:	0	0	1	0	0	0	0	0
r = X & m	:	0	0	x5	0	0	0	0	0

Remarque : m = 1 « 5

r = X & m

On constate qu'après masquage, r peut prendre 2 valeurs différentes :

X	:	x7	x6	x5	x4	x3	x2	x1	x0
m	:	0	0	1	0	0	0	0	0
r = X & m	:	0	0	x5	0	0	0	0	0
si x5= , r	:	0	0		0	0	0	0	0
si x5= , r	:	0	0		0	0	0	0	0

r = X & m

On constate qu'après masquage, r peut prendre 2 valeurs différentes :

X	:	x7	x6	x5	x4	x3	x2	x1	x0
m	:	0	0	1	0	0	0	0	0
r = X & m	:	0	0	x5	0	0	0	0	0
si x5=0, r=0	:	0	0	0	0	0	0	0	0
si x5= , r	:	0	0		0	0	0	0	0

r = X & m

On constate qu'après masquage, r peut prendre 2 valeurs différentes :

X	:	x7	x6	x5	x4	x3	x2	x1	x0
m	:	0	0	1	0	0	0	0	0
r = X & m	:	0	0	x5	0	0	0	0	0
si x5=0, r=0	:	0	0	0	0	0	0	0	0
si x5=1, r≠0	:	0	0	1	0	0	0	0	0

Méthode

Les 2 tests suivants permettent de tester l'état du bit i d'une variable X :

si le bit xi est à 0 alors

```
if (                    )
{
....
}
```

si le bit xi est à 1 alors

```
if (                    )
{
....
}
```

Méthode

Les 2 tests suivants permettent de tester l'état du bit i d'une variable X :

si le bit xi est à 0 alors

```
if ( (X & ( 1 « i)) == 0 )
{
....
}
```

si le bit xi est à 1 alors

```
if (                    )
{
....
}
```

Méthode

Les 2 tests suivants permettent de tester l'état du bit i d'une variable X :

si le bit xi est à 0 alors

```
if ( (X & ( 1 « i)) == 0 )
{
....
}
```

si le bit xi est à 1 alors

```
if ( (X & ( 1 « i)) ≠0 )
{
....
}
```

Sommaire

4 Lire une entrée
- Observer un bit
- registre PINx

registre d'état PINx

Pour chaque port, il existe un registre d'état PINx

Chaque bit du port représente la valeur présente sur la broche correspondante.

Suivant la tension présente sur la broche, l'état lu sera :

tension de sortie ≈0V	0
tension de sortie ≈VCC	1

exemple

Voici un exemple pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
état des pins	:	0.1v	0.2v	4.8v	4.75v	0v	0v	5v	0v
PINB	:								

registre d'état PINx

Pour chaque port, il existe un registre d'état PINx

Chaque bit du port représente la valeur présente sur la broche correspondante.

Suivant la tension présente sur la broche, l'état lu sera :

tension de sortie ≈0V	0
tension de sortie ≈VCC	1

exemple

Voici un exemple pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
état des pins	:	0.1v	0.2v	4.8v	4.75v	0v	0v	5v	0v
PINB	:								0

registre d'état PINx

Pour chaque port, il existe un registre d'état PINx

Chaque bit du port représente la valeur présente sur la broche correspondante.

Suivant la tension présente sur la broche, l'état lu sera :

tension de sortie ≈0V	0
tension de sortie ≈VCC	1

exemple

Voici un exemple pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
état des pins	:	0.1v	0.2v	4.8v	4.75v	0v	0v	5v	0v
PINB	:	0	0	1	1	0	0	1	0

registre d'état PINx

Pour chaque port, il existe un registre d'état PINx

Chaque bit du port représente la valeur présente sur la broche correspondante.

Suivant la tension présente sur la broche, l'état lu sera :

tension de sortie ≈0V	0
tension de sortie ≈VCC	1

exemple

Voici un exemple pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
état des pins	:	0.1v	0.2v	4.8v	4.75v	0v	0v	5v	0v
PINB	:	0	0	1	1	0	0	1	0

Si on souhaite exécuter un action seulement si PB6 est à l'état 1 :

registre d'état PINx

Pour chaque port, il existe un registre d'état PINx

Chaque bit du port représente la valeur présente sur la broche correspondante.

Suivant la tension présente sur la broche, l'état lu sera :

tension de sortie ≈0V	0
tension de sortie ≈VCC	1

exemple

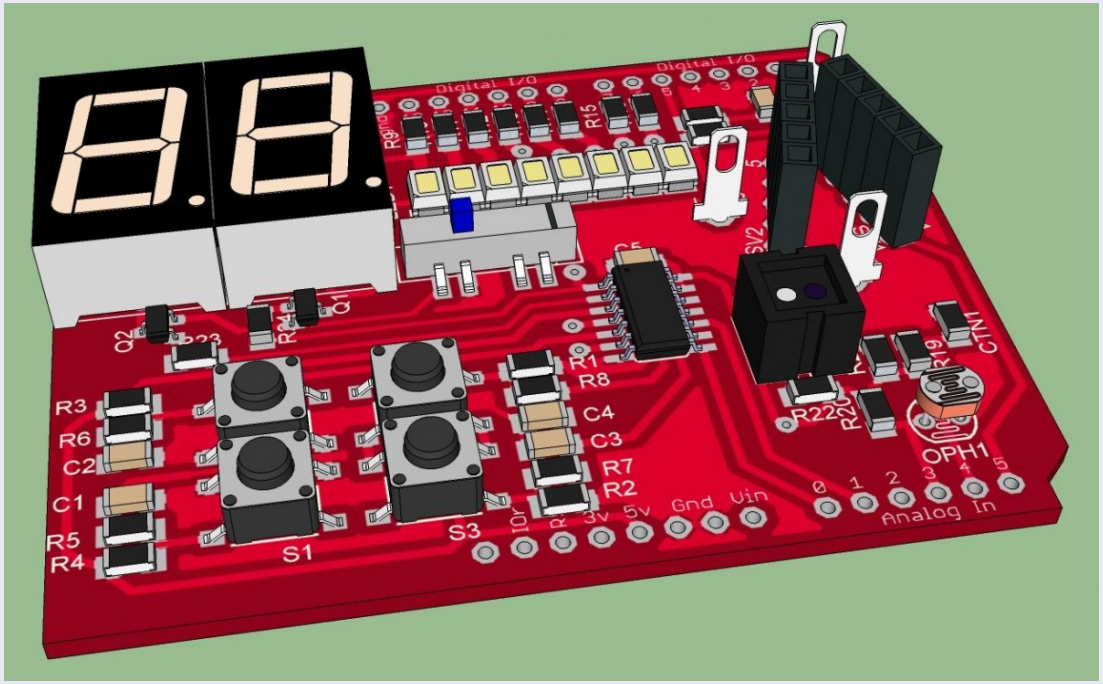
Voici un exemple pour le PORTB

pin	:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
état des pins	:	0.1v	0.2v	4.8v	4.75v	0v	0v	5v	0v
PINB	:	0	0	1	1	0	0	1	0

Si on souhaite exécuter un action seulement si PB6 est à l'état 1 :

```
if ( (PINB & (1«PB6) ) != 0) .....
```

"Lire" un bouton



Couleur	r	o	v	r	o	v	v	r
Port	PB5	PB4	PB3	PB2	PB1	PB0	PD7	PD6

Bouton	A	B	C	D
Port	PD2	PC0	PC1	PD3

avr/sfr_defs.h

bis_is_set(r,b) : Retourne vrai si le bit b du registre r est à 1.
bit_is_clear(r,b) : Retourne vrai si le bit b du registre r est à 0.

Utilisaton des macros AVR libc

```
#include <avr/sfr_defs.h>
int main()
{
        ...
        while(1)
        {
                if (bit_is_set(PINB,5))
                {
                        ...
                }
        }
}
```

avr/sfr_defs.h

loop_until_bit_is_set(r,b) ; : Attendre que le bit passe à l'état 1.
loop_until_bit_is_clear(r,b) ; : Attendre que le bit passe à l'état 0.

Utilisaton des macros AVR libc

```
#include <avr/sfr_defs.h>
int main()
{
...
while(1)
{
        ...
        loop_until_bit_is_clear(PINB,PB5);
        ...
}
}
```

Sommaire

1 Introduction

2 Entrée, sortie ?

3 Modifier une sortie

4 Lire une entrée

5 Cas des bps

6 Gestion d'événement externe

7 Événement périodique

8 MLI/PWM

9 Les "modes" d'entrées

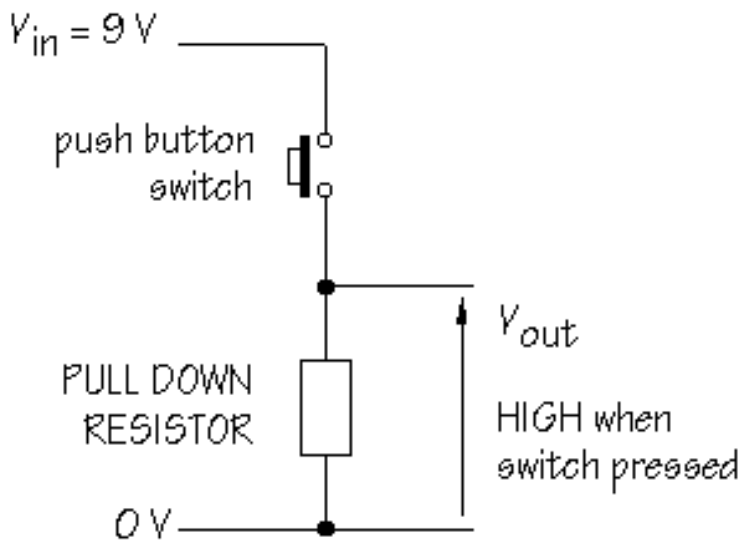
10 Transmission de données

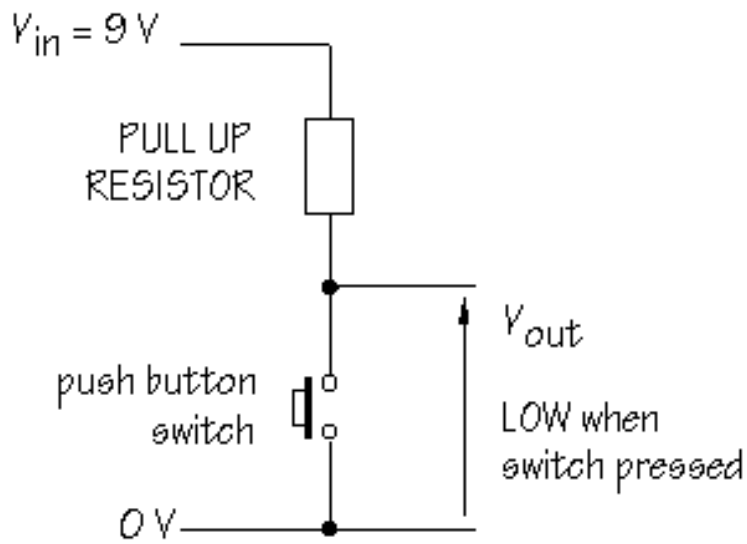
11 Configuration du µcontrôleur

12 Choix microcontroleur

Sommaire

5 Cas des bps
- Résistance de tirage
- Pull-Up


V_{in} = 9 V
push button switch
PULL DOWN RESISTOR
0 V
V_{out}
HIGH when switch pressed


V_{in} = 9 V
PULL UP RESISTOR
push button switch
0 V
V_{out}
LOW when switch pressed

Sommaire

5 Cas des bps
- Résistance de tirage
- Pull-Up

pull-up interne

Lorsqu'une broche est configurée en entrée, il est possible d'activer une résistance de pull-up interne.
Cette configuration se fait broche par broche. Ainsi sur un même port, il peut y avoir des entrées avec pull-up, et d'autres entrées sans.
Pour activer la résistance de pull-up, il faut mettre à 1 le bit du registre PORTx de la broche.

exemple

On connecte un bouton poussoir sur la broche PC6, et on utilise la résistance de pull-up interne :

```
int main()
{



}
```

pull-up interne

Lorsqu'une broche est configurée en entrée, il est possible d'activer une résistance de pull-up interne.
Cette configuration se fait broche par broche. Ainsi sur un même port, il peut y avoir des entrées avec pull-up, et d'autres entrées sans.
Pour activer la résistance de pull-up, il faut mettre à 1 le bit du registre PORTx de la broche.

exemple

On connecte un bouton poussoir sur la broche PC6, et on utilise la résistance de pull-up interne :

```
int main()
{
        DDRC &= ~( 1 « PC6 );

}
```

pull-up interne

Lorsqu'une broche est configurée en entrée, il est possible d'activer une résistance de pull-up interne.
Cette configuration se fait broche par broche. Ainsi sur un même port, il peut y avoir des entrées avec pull-up, et d'autres entrées sans.
Pour activer la résistance de pull-up, il faut mettre à 1 le bit du registre PORTx de la broche.

exemple

On connecte un bouton poussoir sur la broche PC6, et on utilise la résistance de pull-up interne :

```
int main()
{
        DDRC &= ~( 1 « PC6 );
        PORTC |= 1«PC6;
}
```

Sommaire

1 Introduction

2 Entrée, sortie ?

3 Modifier une sortie

4 Lire une entrée

5 Cas des bps

6 Gestion d'événement externe

7 Événement périodique

8 MLI/PWM

9 Les "modes" d'entrées

10 Transmission de données

11 Configuration du µcontrôleur

12 Choix microcontroleur

Sommaire

6 Gestion d'événement externe
- par scrutation
- par interruption dédiée
- par interruption par groupe de broches

Ex d'attente d'événement

L'exemple typique est l'attente de l'appui sur un bouton (relié sur PC6) :

```
int main()
{
        DDRC &= ~( 1 « PC6 );
        PORTC |= 1«PC6;
        while(1)
        {
                while( (PINC & (1«PC6)) ?? 0); // ne rien faire tant que ...
                ....
        }
}
```

inconvénient

- Le programme est bloqué sur l'attente.
- Le processeur ne peut rien faire d'autre en attendant !

Sommaire

6 Gestion d'événement externe
- par scrutation
- par interruption dédiée
- par interruption par groupe de broches

Principe

- Un événement déclenche une interruption
- L'interruption est associée à une fonction
- La boucle principale s'arrête
- Le programme d'interruption s'exécute
- La boucle principale reprend

INT0 et INT1

On considère ici le cas d'un atmega328p

- INT0 associée à PD2
- INT1 associée à PD3
- déclenchement sur front montant/descendant
- déclenchement sur niveau

fonction d'interruption

```
ISR(INTx_vect)
{
        ...
}

int main()
{
        // mode de declenchement
        // autorisation interruption
        ...
        while(1)
        {
                ...
        }
}
```

Flag (drapeau) et interruption

```
volatile uint8_t f_chgt=0;
ISR(INTx_vect)
{
        f_chgt = ....;
}

int main()
{
        ...
        while(1)
        {
                if (f_chgt == ...)
                ...
        }
}
```

mode de déclenchement

On considère ici un atmega328p.

Bit	7	6	5	4	3	2	1	0
EICRA	—	—	—	—	ISC11	ISC10	ISC01	ISC00
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

ISCx1	ISCx0	Interruption sur :
0	0	niveau bas de INTx
0	1	changement d'état de INTx
1	0	front descendant de INTx
1	1	front montant de INTx

autorisation d'interruption

On considère toujours un atmega328p.

Bit	7	6	5	4	3	2	1	0
EIMSK	—	—	—	—	—	—	INT1	INT0
Read/Write	R	R	R	R	R	R	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

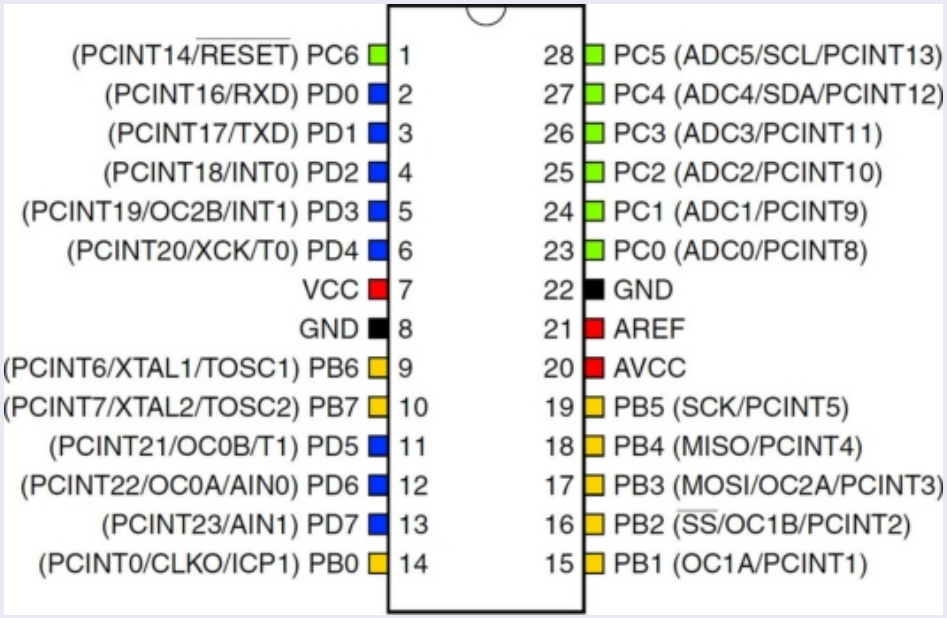
Bit	7	6	5	4	3	2	1	0
SREG	I	T	H	S	V	N	Z	C
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

```
...
cli();      // clear gloabl interrupt
...
sei();      // set global interrupt
...
```

Sommaire

6 Gestion d'événement externe
- par scrutation
- par interruption dédiée
- par interruption par groupe de broches

PCIx interrupt (cas de l'atmega328p)



- PCI0 : PORTB, PCINT[7:0]
- PCI1 : PORTC, PCINT[14:8]
- PCI2 : PORD, PCINT[23:16]
- uniquement sur changement
- choix des broches

Registres à configurer (atmega328p)

Registre Pin Change Interrupt Control Register.
PCIE : Pin Change Interrupt Enable.

Bit	7	6	5	4	3	2	1	0
PCICR	—	—	—	—	—	PCIE2	PCIE1	PCIE0
Read/Write	R	R	R	R	R	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Registre Pin Change Mask Register :

Bit	7	6	5	4	3	2	1	0
PCMSK0	PCINT7	PCINT6	PCINT5	PCINT4	PCINT3	PCINT2	PCINT1	PCINT0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

PCMSK1 : PCINT8 à PCINT14
PCMSK2 : PCINT16 à PCINT23

fonction d'interruption

```
ISR(PCINT0_vect)
{
        ...
}

int main()
{
        // mode de declenchement
        PCMSK0 = 0bxxxxxxxx;    // ou PCMSK0 |= 1<<PCINTx;
        // autorisation interruption
        PCICR = 0b00000xxx;     // ou PCICR  |= 1<<PCIEx;
        sei();
        while(1)
        {
                ...
        }
}
```

Sommaire

1 Introduction

2 Entrée, sortie ?

3 Modifier une sortie

4 Lire une entrée

5 Cas des bps

6 Gestion d'événement externe

7 Événement périodique

8 MLI/PWM

9 Les "modes" d'entrées

10 Transmission de données

11 Configuration du µcontrôleur

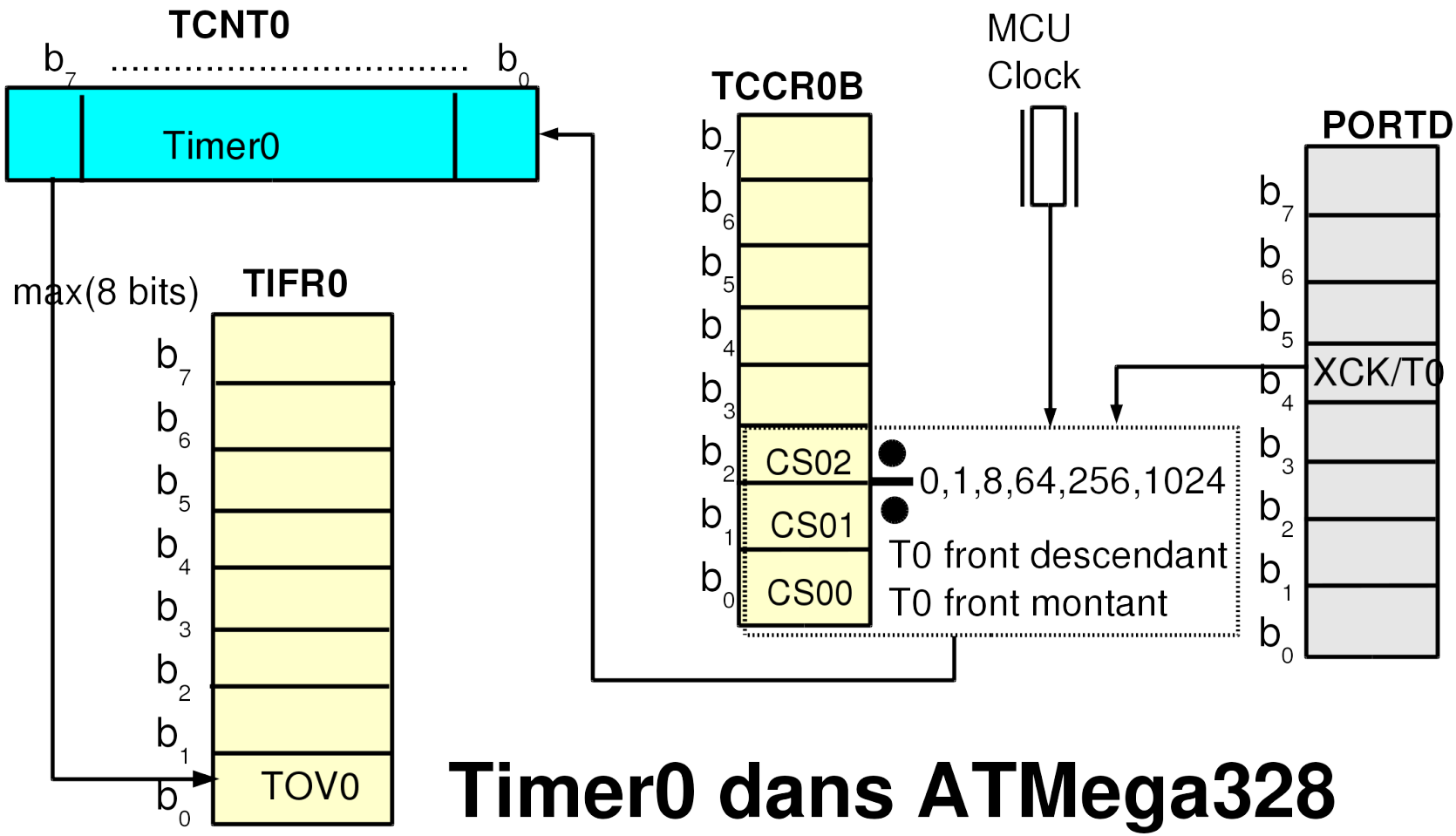
12 Choix microcontroleur

Sommaire

7 Événement périodique
- Le timer
- Mode normal
- Mode comparaison
- Compare Match Output Unit

Caractéristiques

- Compteur n bits
- Fréquence (source) de comptage



Timer0 dans ATMega328

Sommaire

7 Événement périodique
- Le timer
- Mode normal
- Mode comparaison
- Compare Match Output Unit

Interruption de débordement (Overflow)

En mode "normal" un timer déborde régulièrement. Il est possible de générer une interruption à ce moment précis.

Registre TIMSK0 atmega328p

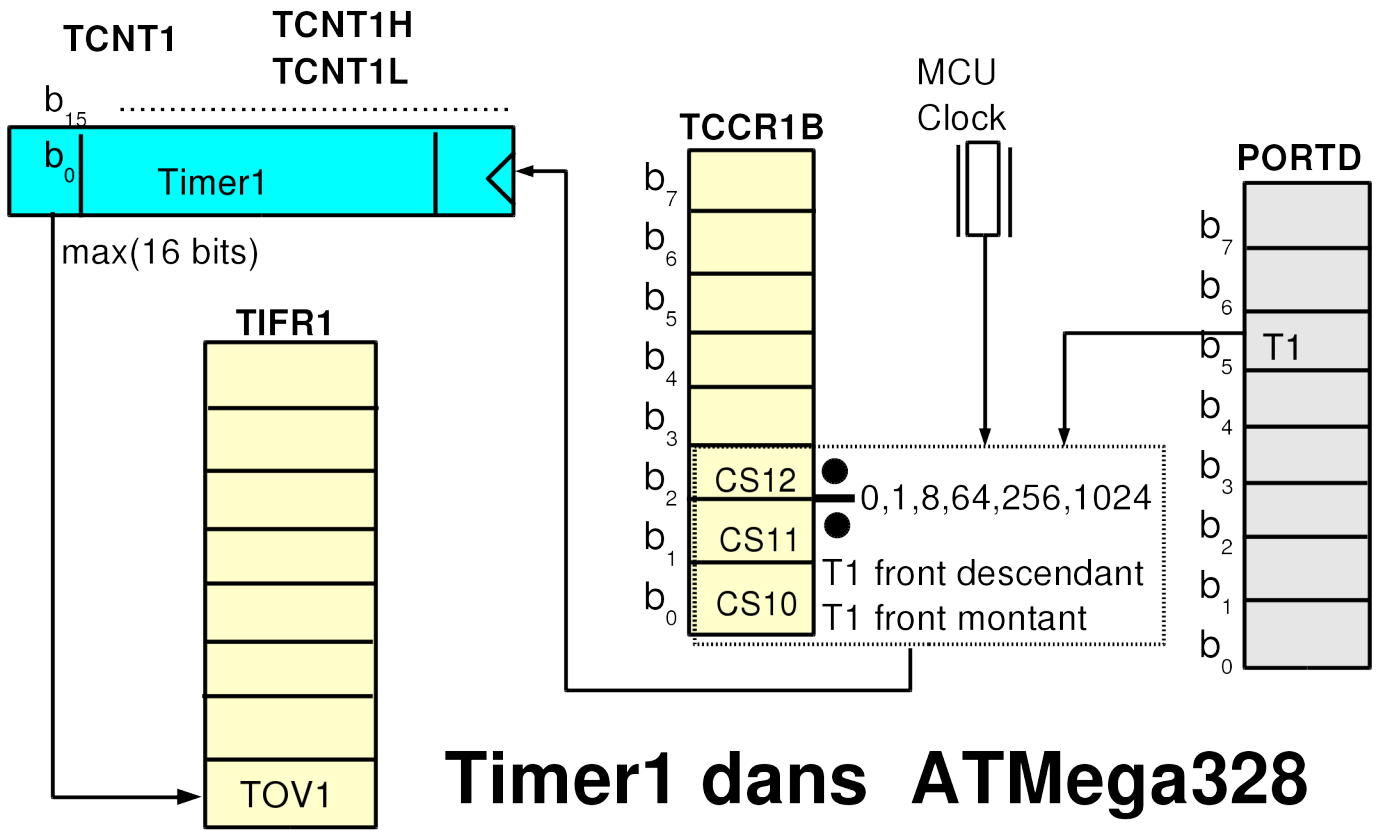
Bit	7	6	5	4	3	2	1	0
TIMSK0	—	—	—	—	—	OCIE0B	OCIE0A	TOIE0
Read/Write	R	R	R/W	R	R	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

TOIE0 : Timer0 Overflow Interrupt Enable

```
ISR(TIMER0_OVF_vect)
{
...
}
```

Ex avec atmega328p, 16MHz

Couleur	r	o	v	r	o	v	v	r
Port	PB5	PB4	PB3	PB2	PB1	PB0	PD7	PD6



Timer1 dans ATMega328

Registre TIMSK1 atmega328p

Timer1 Interrupt MaSK register :

Bit	7	6	5	4	3	2	1	0
TIMSK1	—	—	ICIE1	—	—	OCIE1B	OCIE1A	TOIE1
Read/Write	R	R	R/W	R	R	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

- TOIE1 : autorisation d'interruption de débordement Timer1 Overflow Interrupt Enable
- OCIE1A : autorisation d'interruption de comparaison A Output Compare A Interrupt Enable timer 1
- OCIE1B : autorisation d'interruption de comparaison B Output Compare B Interrupt Enable timer 1
- ICIE1 : autorisation d'interruption sur capture d'un événement

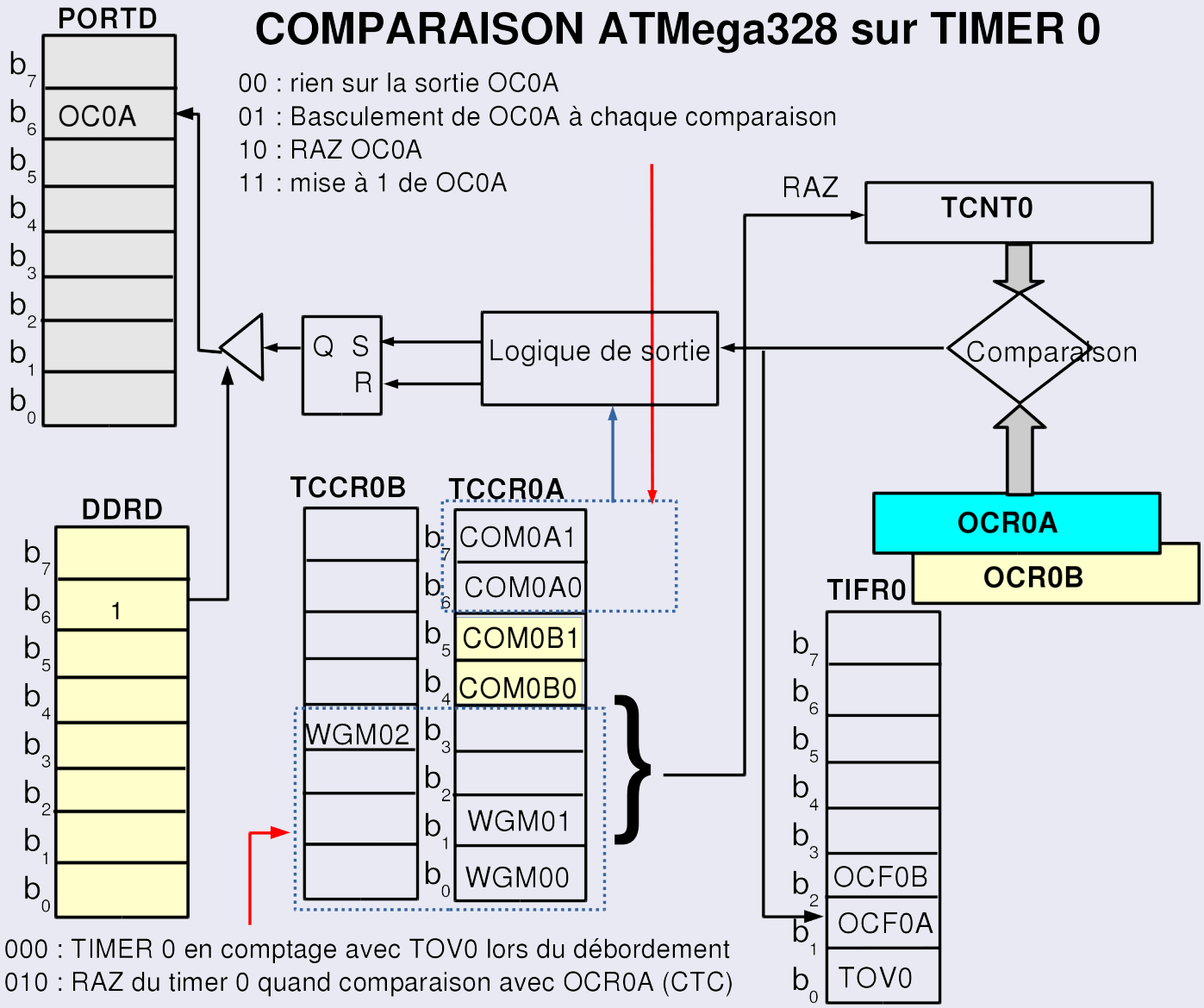
exemples

- fréquences de sortie possibles
- faire clignoter une led
- arrêter/activer le clignotement à l'appui sur un bouton

Sommaire

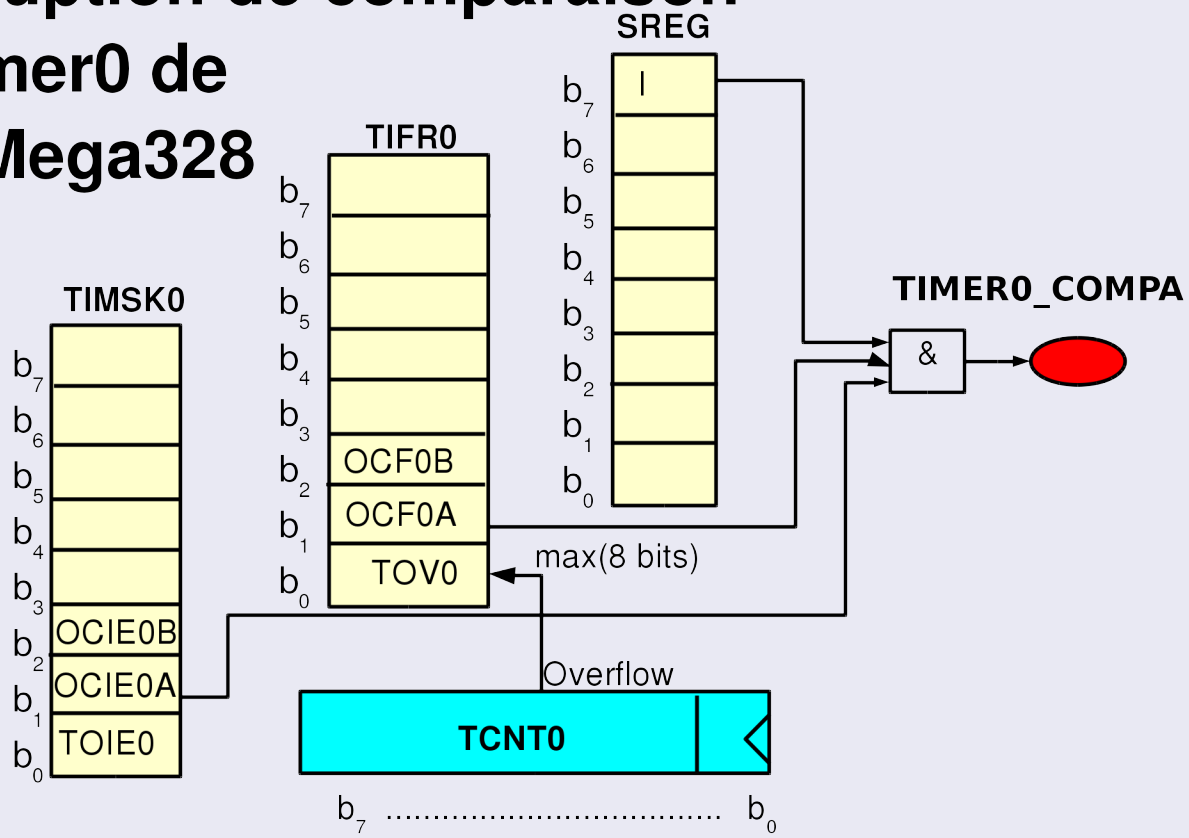
7 Événement périodique
- Le timer
- Mode normal
- Mode comparaison
- Compare Match Output Unit

Remise à zéro sur comparaison (CTC Clear Timer on Compare)



Interruption sur comparaison

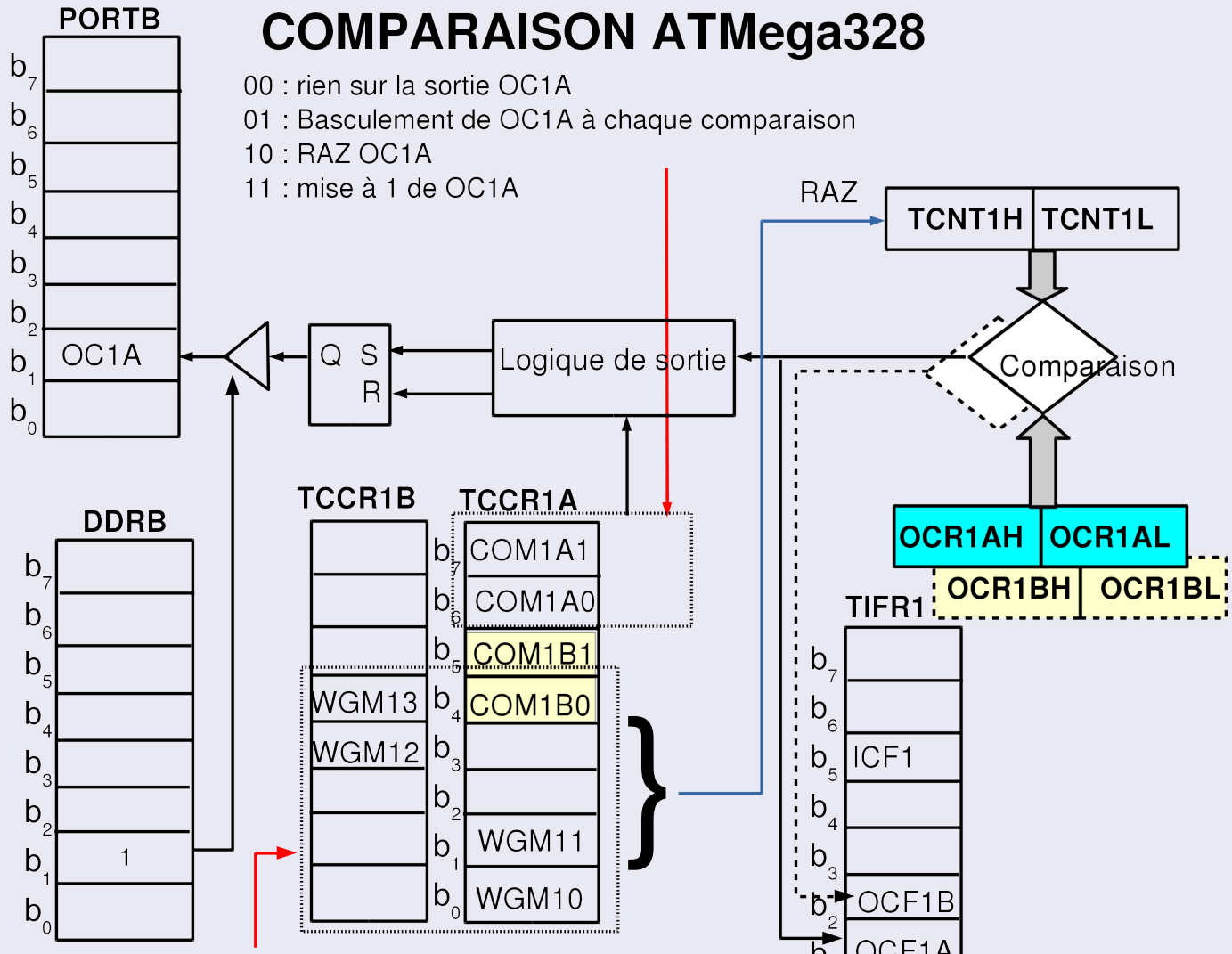
Interruption de comparaison du timer0 de l'ATMega328



exemples, atmega328p 16MHz

- faire clignoter une led à 1Hz

Ex atmega328p

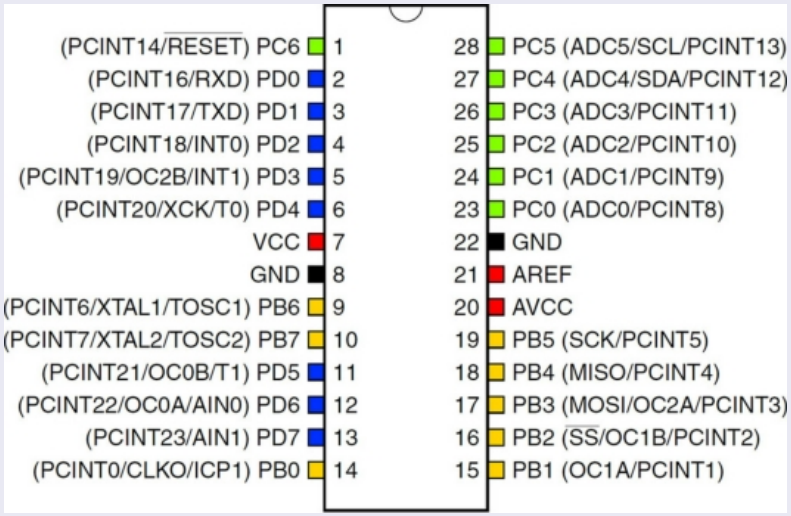


Sommaire

7 Événement périodique
- Le timer
- Mode normal
- Mode comparaison
- Compare Match Output Unit

Modification directe de la sortie

Bit	7	6	5	4	3	2	1	0
TCCR1A	COM1A1	COM1A0	COM1B1	COM1B0	—	—	WGM11	WGM10
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0



COM1A1	COM1A0	action sur OC1A
0	0	pas d'action
0	1	Basculement
1	0	mise à 0
1	1	mise à 1

Sommaire

1 Introduction

2 Entrée, sortie ?

3 Modifier une sortie

4 Lire une entrée

5 Cas des bps

6 Gestion d'événement externe

7 Événement périodique

8 MLI/PWM

9 Les "modes" d'entrées

10 Transmission de données

11 Configuration du µcontrôleur

12 Choix microcontroleur

Sommaire

8 MLI/PWM
- Quésaco
- Timer et PWM

De la sortie logique

- 2 tensions
- led, lampe, moteur ...
- transistor ?

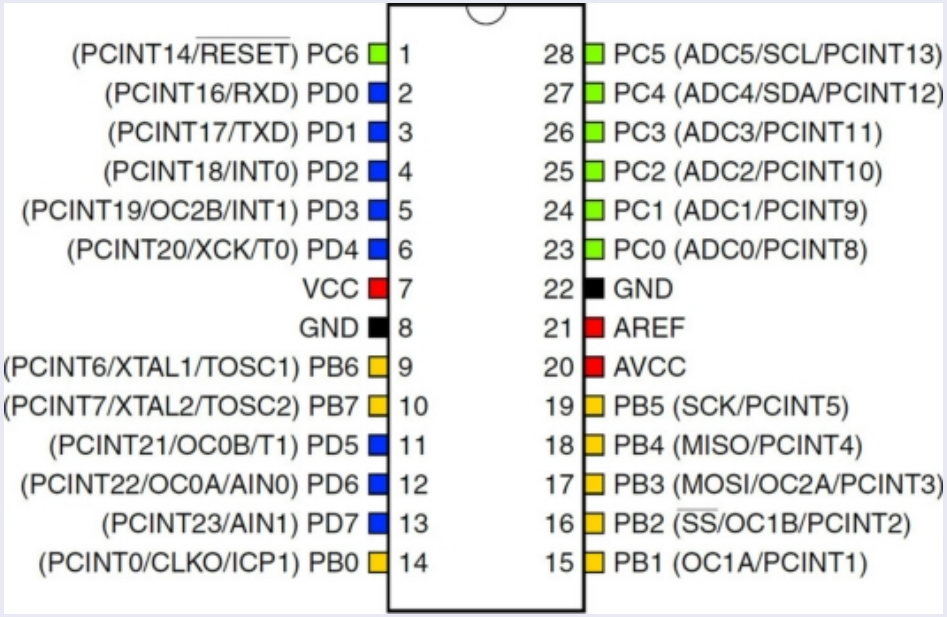
à la sortie MLI

- alterner rapidement les états logiques
- fréquence/période
- rapport cyclique (duty cycle)
- filtrage -> signal analogique : génération de signal

Sommaire

8 MLI/PWM
- Quésaco
- Timer et PWM

Principe



- choix de la fréquence
- valeur de comparaison
- gestion de la sortie

15.9 Register Description

15.9.1 TCCR0A – Timer/Counter Control Register A

Bit	7	6	5	4	3	2	1	0	
0x24 (0x44)	COM0A1	COM0A0	COM0B1	COM0B0	–	–	WGM01	WGM00	TCCR0A
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7:6 – COM0A1:0: Compare Match Output A Mode**

These bits control the Output Compare pin (OC0A) behavior. If one or both of the COM0A1:0 bits are set, the OC0A output overrides the normal port functionality of the I/O pin it is connected to. However, note that the Data Direction Register (DDR) bit corresponding to the OC0A pin must be set in order to enable the output driver.

When OC0A is connected to the pin, the function of the COM0A1:0 bits depends on the WGM02:0 bit setting. Table 15-2 shows the COM0A1:0 bit functionality when the WGM02:0 bits are set to a normal or CTC mode (non-PWM).

Table 15-2. Compare Output Mode, non-PWM Mode

COM0A1	COM0A0	Description
0	0	Normal port operation, OC0A disconnected.
0	1	Toggle OC0A on Compare Match
1	0	Clear OC0A on Compare Match
1	1	Set OC0A on Compare Match

Table 15-3 shows the COM0A1:0 bit functionality when the WGM01:0 bits are set to fast PWM mode.

Table 15-3. Compare Output Mode, Fast PWM Mode(1)

COM0A1	COM0A0	Description
0	0	Normal port operation, OC0A disconnected.
0	1	WGM02 = 0: Normal Port Operation, OC0A Disconnected. WGM02 = 1: Toggle OC0A on Compare Match.
1	0	Clear OC0A on Compare Match, set OC0A at BOTTOM, (non-inverting mode).
1	1	Set OC0A on Compare Match, clear OC0A at BOTTOM, (inverting mode).

Note: 1. A special case occurs when OCR0A equals TOP and COM0A1 is set. In this case, the Compare Match is ignored, but the set or clear is done at BOTTOM. See "Fast PWM Mode" on page 99 for more details.

Table 15-4 shows the COM0A1:0 bit functionality when the WGM02:0 bits are set to phase correct PWM mode.

Table 15-4. Compare Output Mode, Phase Correct PWM Mode(1)

COM0A1	COM0A0	Description
0	0	Normal port operation, OC0A disconnected.
0	1	WGM02 = 0: Normal Port Operation, OC0A Disconnected. WGM02 = 1: Toggle OC0A on Compare Match.
1	0	Clear OC0A on Compare Match when up-counting. Set OC0A on Compare Match when down-counting.
1	1	Set OC0A on Compare Match when up-counting. Clear OC0A on Compare Match when down-counting.

Note: 1. A special case occurs when OCR0A equals TOP and COM0A1 is set. In this case, the Compare Match is ignored, but the set or clear is done at TOP. See "Phase Correct PWM Mode" on page 125 for more details.

- **Bits 5:4 – COM0B1:0: Compare Match Output B Mode**

These bits control the Output Compare pin (OC0B) behavior. If one or both of the COM0B1:0 bits are set, the OC0B output overrides the normal port functionality of the I/O pin it is connected to. However, note that the Data Direction Register (DDR) bit corresponding to the OC0B pin must be set in order to enable the output driver.

When OC0B is connected to the pin, the function of the COM0B1:0 bits depends on the WGM02:0 bit setting. Table 15-5 shows the COM0B1:0 bit functionality when the WGM02:0 bits are set to a normal or CTC mode (non-PWM).

Table 15-5. Compare Output Mode, non-PWM Mode

COM0B1	COM0B0	Description
0	0	Normal port operation, OC0B disconnected.
0	1	Toggle OC0B on Compare Match
1	0	Clear OC0B on Compare Match
1	1	Set OC0B on Compare Match

Table 15-6 shows the COM0B1:0 bit functionality when the WGM02:0 bits are set to fast PWM mode.

Table 15-6. Compare Output Mode, Fast PWM Mode(1)

COM0B1	COM0B0	Description
0	0	Normal port operation, OC0B disconnected.
0	1	Reserved
1	0	Clear OC0B on Compare Match, set OC0B at BOTTOM, (non-inverting mode)
1	1	Set OC0B on Compare Match, clear OC0B at BOTTOM, (inverting mode).

Note: 1. A special case occurs when OCR0B equals TOP and COM0B1 is set. In this case, the Compare Match is ignored, but the set or clear is done at TOP. See "Fast PWM Mode" on page 99 for more details.

Table 15-7 shows the COM0B1:0 bit functionality when the WGM02:0 bits are set to phase correct PWM mode.

Table 15-7. Compare Output Mode, Phase Correct PWM Mode(1)

COM0B1	COM0B0	Description
0	0	Normal port operation, OC0B disconnected.
0	1	Reserved
1	0	Clear OC0B on Compare Match when up-counting. Set OC0B on Compare Match when down-counting.
1	1	Set OC0B on Compare Match when up-counting. Clear OC0B on Compare Match when down-counting.

Note: 1. A special case occurs when OCR0B equals TOP and COM0B1 is set. In this case, the Compare Match is ignored, but the set or clear is done at TOP. See "Phase Correct PWM Mode" on page 100 for more details.

- **Bits 3, 2 – Reserved**

These bits are reserved bits in the ATmega48A/PA/88A/PA/168A/PA/328/P and will always read as zero.

- **Bits 1:0 – WGM01:0: Waveform Generation Mode**

Combined with the WGM02 bit found in the TCCR0B Register, these bits control the counting sequence of the counter, the source for maximum (TOP) counter value, and what type of waveform generation to be used, see Table 15-8. Modes of operation supported by the Timer/Counter unit are: Normal mode (counter), Clear Timer on Compare Match (CTC) mode, and two types of Pulse Width Modulation (PWM) modes (see "Modes of Operation" on page 98).

Table 15-8. Waveform Generation Mode Bit Description

Mode	WGM02	WGM01	WGM00	Timer/Counter Mode of Operation	TOP	Update of OCRx at	TOV Flag Set on(1)(2)
0	0	0	0	Normal	0xFF	Immediate	MAX
1	0	0	1	PWM, Phase Correct	0xFF	TOP	BOTTOM
2	0	1	0	CTC	OCRA	Immediate	MAX
3	0	1	1	Fast PWM	0xFF	BOTTOM	MAX
4	1	0	0	Reserved	–	–	–
5	1	0	1	PWM, Phase Correct	OCRA	TOP	BOTTOM
6	1	1	0	Reserved	–	–	–
7	1	1	1	Fast PWM	OCRA	BOTTOM	TOP

Notes: 1. MAX = 0xFF
2. BOTTOM = 0x00

15.9.2 TCCR0B – Timer/Counter Control Register B

Bit	7	6	5	4	3	2	1	0	
0x25 (0x45)	FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00	TCCR0B
Read/Write	W	W	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – FOC0A: Force Output Compare A**

The FOC0A bit is only active when the WGM bits specify a non-PWM mode.

However, for ensuring compatibility with future devices, this bit must be set to zero when TCCR0B is written when operating in PWM mode. When writing a logical one to the FOC0A bit, an immediate Compare Match is forced on the Waveform Generation unit. The OC0A output is changed according to its COM0A1:0 bits setting. Note that the FOC0A bit is implemented as a strobe. Therefore it is the value present in the COM0A1:0 bits that determines the effect of the forced compare.

A FOC0A strobe will not generate any interrupt, nor will it clear the timer in CTC mode using OCR0A as TOP.

The FOC0A bit is always read as zero.

- **Bit 6 – FOC0B: Force Output Compare B**

The FOC0B bit is only active when the WGM bits specify a non-PWM mode.

However, for ensuring compatibility with future devices, this bit must be set to zero when TCCR0B is written when operating in PWM mode. When writing a logical one to the FOC0B bit, an immediate Compare Match is forced on the Waveform Generation unit. The OC0B output is changed according to its COM0B1:0 bits setting. Note that the FOC0B bit is implemented as a strobe. Therefore it is the value present in the COM0B1:0 bits that determines the effect of the forced compare.

A FOC0B strobe will not generate any interrupt, nor will it clear the timer in CTC mode using OCR0B as TOP.

The FOC0B bit is always read as zero.

- **Bits 5:4 – Reserved**

These bits are reserved bits in the ATmega48A/PA/88A/PA/168A/PA/328/P and will always read as zero.

- **Bit 3 – WGM02: Waveform Generation Mode**

See the description in the "TCCR0A – Timer/Counter Control Register A" on page 104.

- **Bits 2:0 – CS02:0: Clock Select**

The three Clock Select bits select the clock source to be used by the Timer/Counter.

Table 15-9. Clock Select Bit Description

CS02	CS01	CS00	Description
0	0	0	No clock source (Timer/Counter stopped)
0	0	1	$clk_{I/O}$/(No prescaling)
0	1	0	$clk_{I/O}$/8 (From prescaler)
0	1	1	$clk_{I/O}$/64 (From prescaler)
1	0	0	$clk_{I/O}$/256 (From prescaler)
1	0	1	$clk_{I/O}$/1024 (From prescaler)
1	1	0	External clock source on T0 pin. Clock on falling edge.
1	1	1	External clock source on T0 pin. Clock on rising edge.

If external pin modes are used for the Timer/Counter0, transitions on the T0 pin will clock the counter even if the pin is configured as an output. This feature allows software control of the counting.

15.9.3 TCNT0 – Timer/Counter Register

Bit	7	6	5	4	3	2	1	0	
0x26 (0x46)	TCNT0[7:0]								TCNT0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The Timer/Counter Register gives direct access, both for read and write operations, to the Timer/Counter unit 8-bit counter. Writing to the TCNT0 Register blocks (removes) the Compare Match on the following timer clock. Modifying the counter (TCNT0) while the counter is running, introduces a risk of missing a Compare Match between TCNT0 and the OCR0x Registers.

15.9.4 OCR0A – Output Compare Register A

Bit	7	6	5	4	3	2	1	0	
0x27 (0x47)	OCR0A[7:0]								OCR0A
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The Output Compare Register A contains an 8-bit value that is continuously compared with the counter value (TCNT0). A match can be used to generate an Output Compare interrupt, or to generate a waveform output on the OC0A pin.

15.9.5 OCR0B – Output Compare Register B

Bit	7	6	5	4	3	2	1	0	
0x28 (0x48)	OCR0B[7:0]								OCR0B
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The Output Compare Register B contains an 8-bit value that is continuously compared with the counter value (TCNT0). A match can be used to generate an Output Compare interrupt, or to generate a waveform output on the OC0B pin.

15.9.6 TIMSK0 – Timer/Counter Interrupt Mask Register

Bit	7	6	5	4	3	2	1	0	
(0x6E)	–	–	–	–	–	OCIE0B	OCIE0A	TOIE0	TIMSK0
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7:3 – Reserved**

These bits are reserved bits in the ATmega48A/PA/88A/PA/168A/PA/328/P and will always read as zero.

- **Bit 2 – OCIE0B: Timer/Counter Output Compare Match B Interrupt Enable**

When the OCIE0B bit is written to one, and the I-bit in the Status Register is set, the Timer/Counter Compare Match B interrupt is enabled. The corresponding interrupt is executed if a Compare Match in Timer/Counter occurs, i.e., when the OCF0B bit is set in the Timer/Counter Interrupt Flag Register – TIFR0.

- **Bit 1 – OCIE0A: Timer/Counter0 Output Compare Match A Interrupt Enable**

When the OCIE0A bit is written to one, and the I-bit in the Status Register is set, the Timer/Counter0 Compare Match A interrupt is enabled. The corresponding interrupt is executed if a Compare Match in Timer/Counter0 occurs, i.e., when the OCF0A bit is set in the Timer/Counter 0 Interrupt Flag Register – TIFR0.

- **Bit 0 – TOIE0: Timer/Counter0 Overflow Interrupt Enable**

When the TOIE0 bit is written to one, and the I-bit in the Status Register is set, the Timer/Counter0 Overflow interrupt is enabled. The corresponding interrupt is executed if an overflow in Timer/Counter0 occurs, i.e., when the TOV0 bit is set in the Timer/Counter 0 Interrupt Flag Register – TIFR0.

15.9.7 TIFR0 – Timer/Counter 0 Interrupt Flag Register

Bit	7	6	5	4	3	2	1	0	
0x15 (0x35)	–	–	–	–	–	OCF0B	OCF0A	TOV0	TIFR0
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7:3 – Reserved**

These bits are reserved bits in the ATmega48A/PA/88A/PA/168A/PA/328/P and will always read as zero.

- **Bit 2 – OCF0B: Timer/Counter 0 Output Compare B Match Flag**

The OCF0B bit is set when a Compare Match occurs between the Timer/Counter and the data in OCR0B – Output Compare Register0 B. OCF0B is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, OCF0B is cleared by writing a logic one to the flag. When the I-bit in SREG, OCIE0B (Timer/Counter Compare B Match Interrupt Enable), and OCF0B are set, the Timer/Counter Compare Match Interrupt is executed.

- **Bit 1 – OCF0A: Timer/Counter 0 Output Compare A Match Flag**

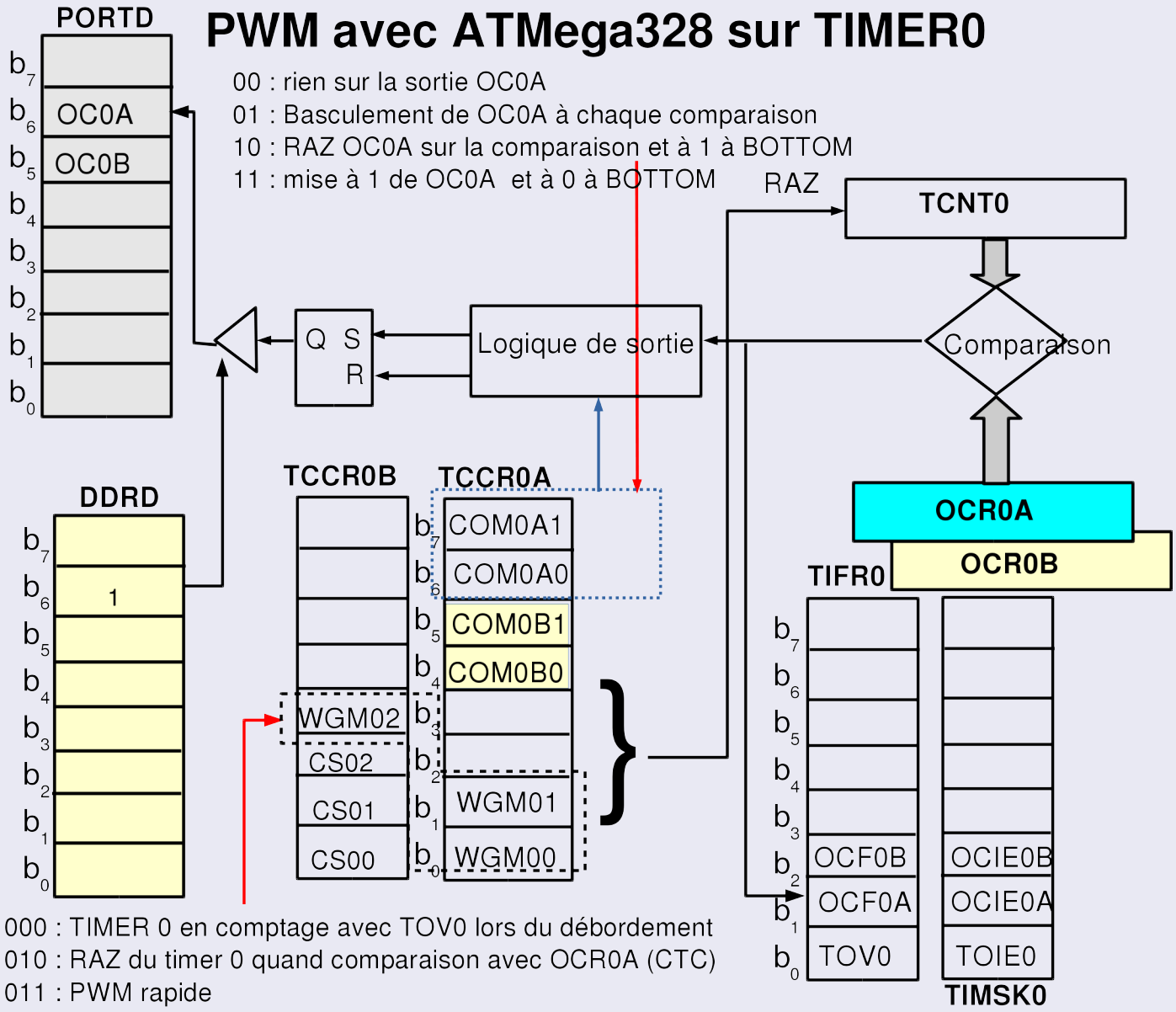
The OCF0A bit is set when a Compare Match occurs between the Timer/Counter0 and the data in OCR0A – Output Compare Register0. OCF0A is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, OCF0A is cleared by writing a logic one to the flag. When the I-bit in SREG, OCIE0A (Timer/Counter0 Compare Match Interrupt Enable), and OCF0A are set, the Timer/Counter0 Compare Match Interrupt is executed.

- **Bit 0 – TOV0: Timer/Counter0 Overflow Flag**

The bit TOV0 is set when an overflow occurs in Timer/Counter0. TOV0 is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, TOV0 is cleared by writing a logic one to the flag. When the SREG I-bit, TOIE0 (Timer/Counter0 Overflow Interrupt Enable), and TOV0 are set, the Timer/Counter0 Overflow interrupt is executed.

The setting of this flag is dependent of the WGM02:0 bit setting. Refer to Table 15-8, "Waveform Generation Mode Bit Description" on page 106.

Synoptique TIMER0



Sommaire

1. Introduction
2. Entrée, sortie ?
3. Modifier une sortie
4. Lire une entrée
5. Cas des bps
6. Gestion d'événement externe
7. Événement périodique
8. MLI/PWM
9. Les "modes" d'entrées
10. Transmission de données
11. Configuration du µcontrôleur
12. Choix microcontroleur

Sommaire

9 Les "modes" d'entrées
- 3 types de signaux
- CAN
- mesure de temps

Grandeur à mesurer

- binaire (2 états)
- analogique (valeur continue)
- durée impulsion (valeur continue)

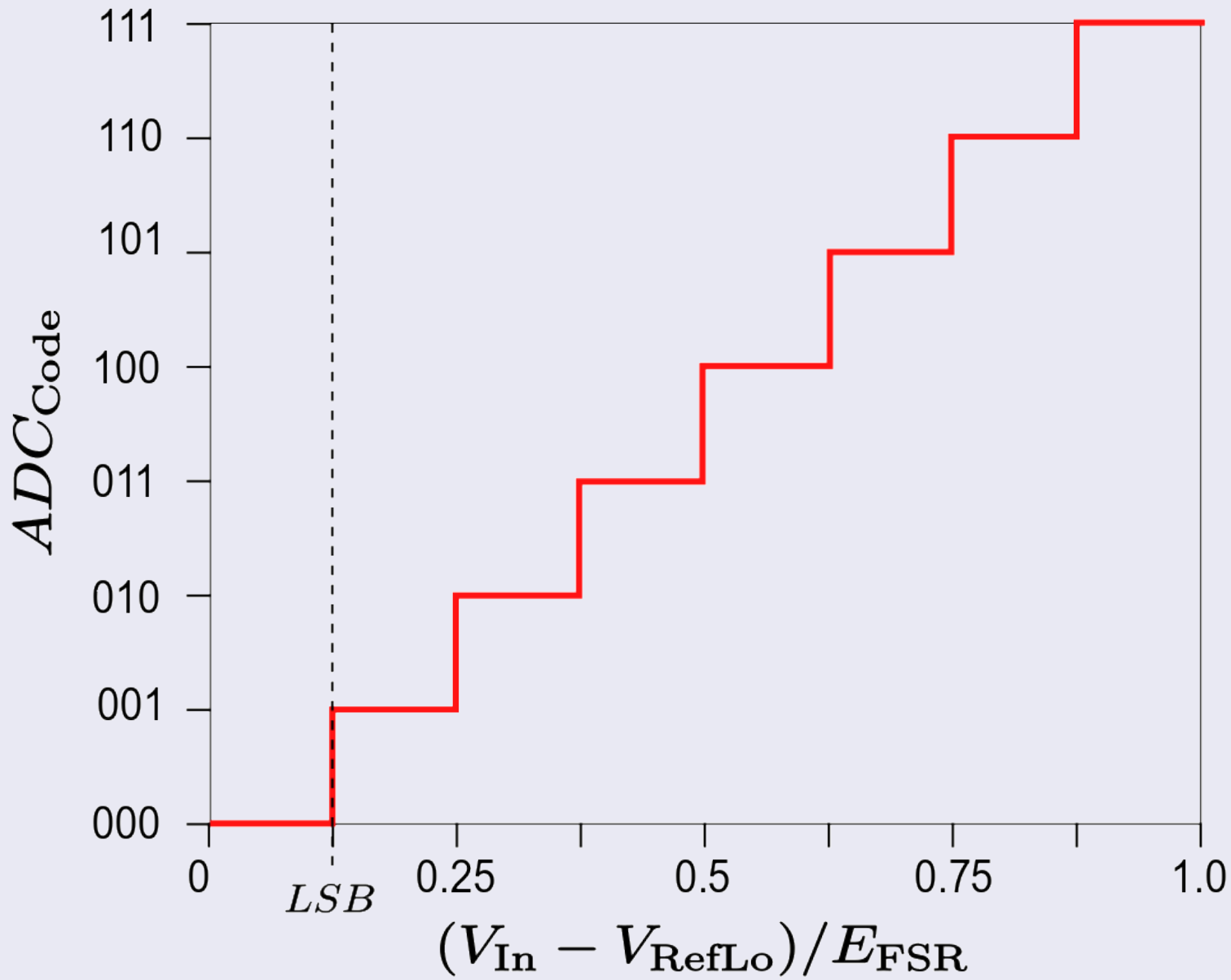
Sommaire

9 Les "modes" d'entrées
- 3 types de signaux
- CAN
- mesure de temps

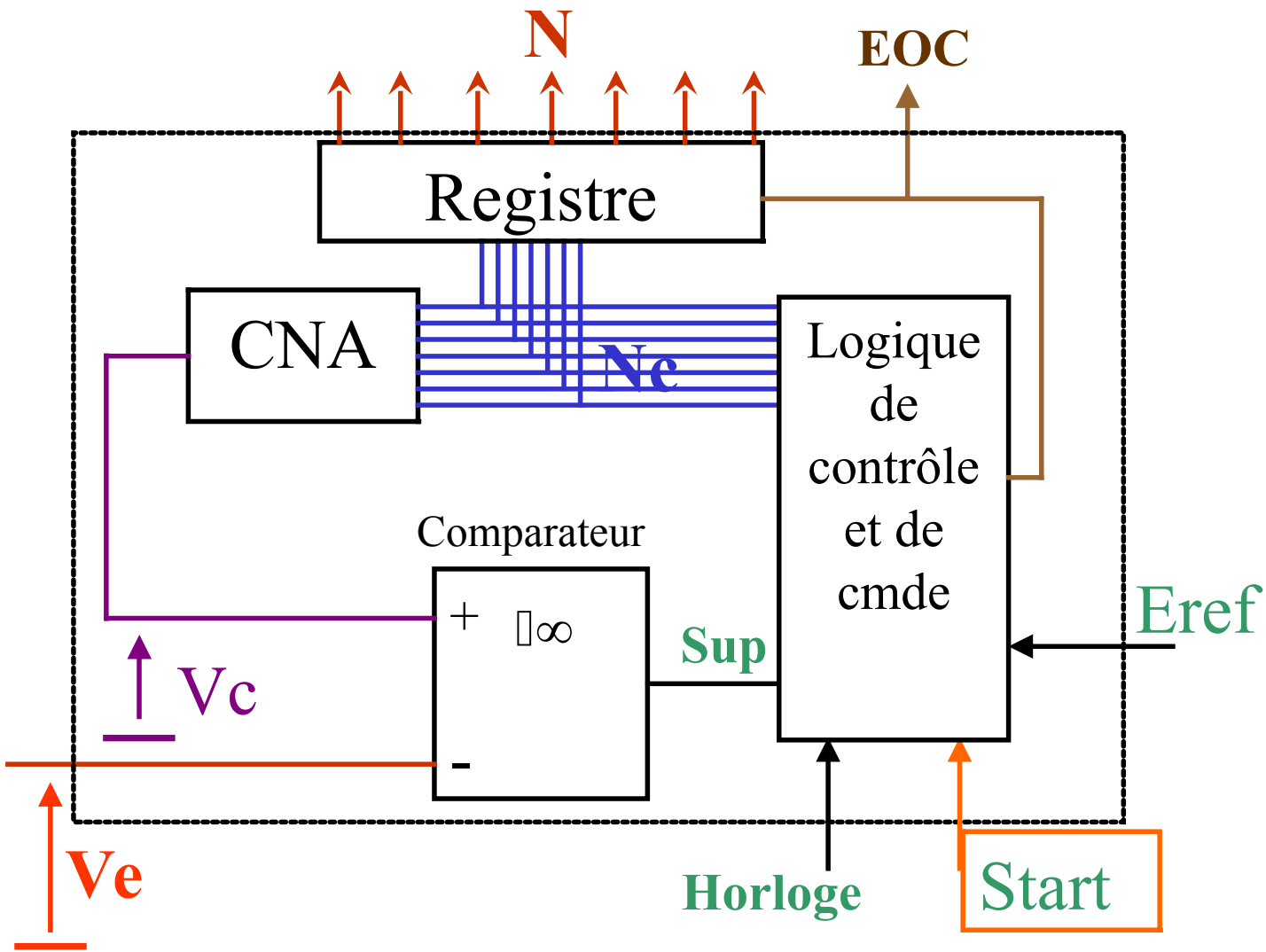
Caractérisation du convertisseur (Analog Digital Converter)

- Nombre de "bits"
- Tension de référence
- Rapidité

fontion de transfert CAN

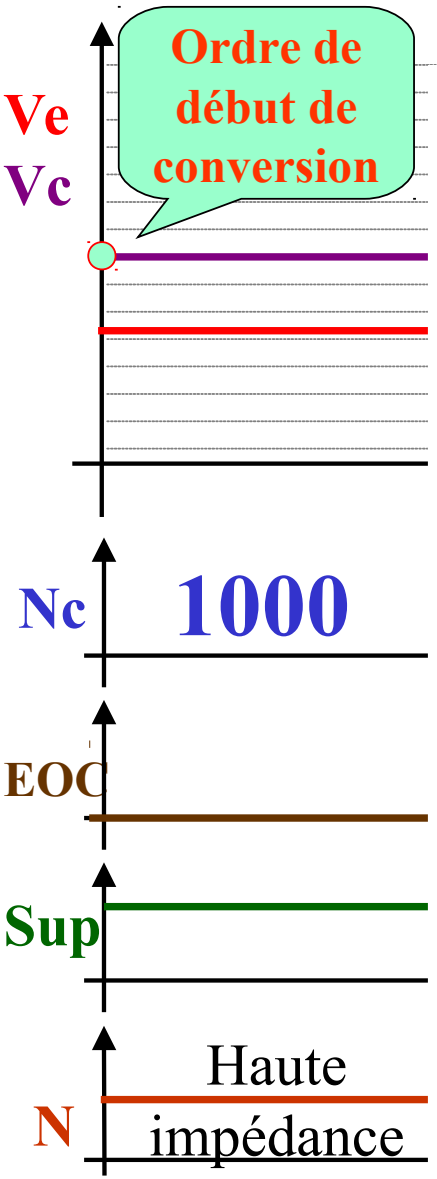


C.A.N. à approximations successives

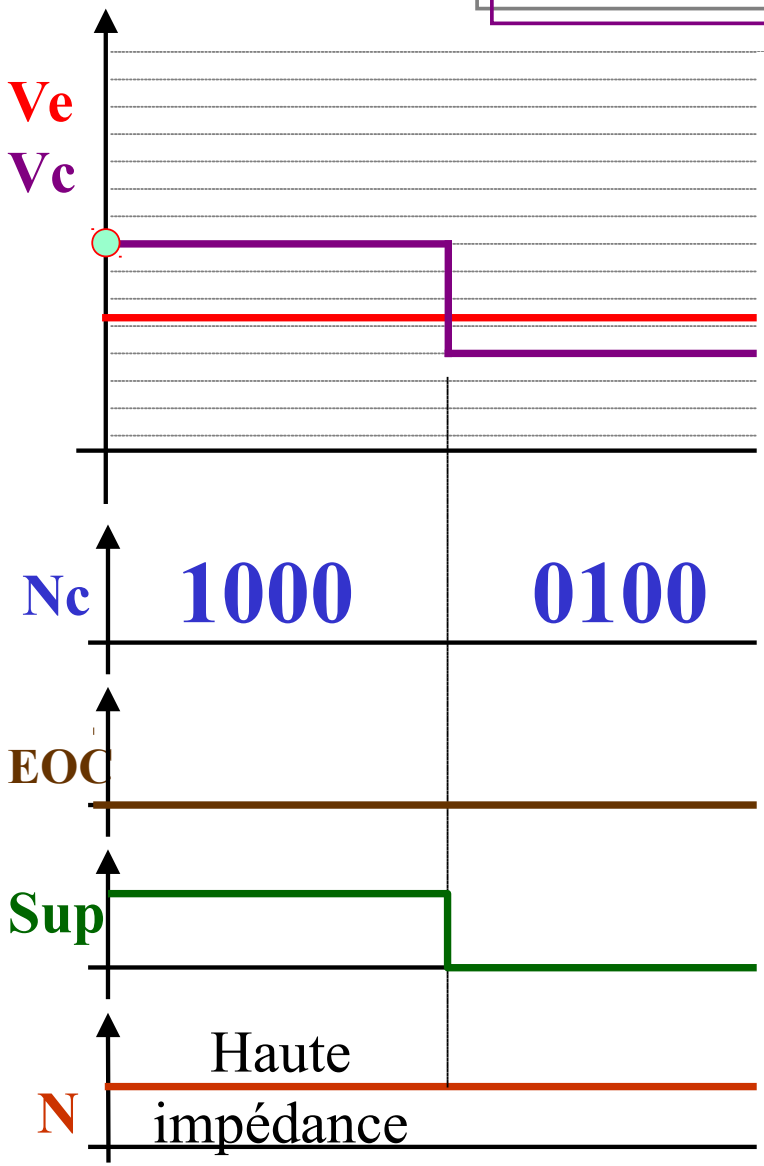


CAN

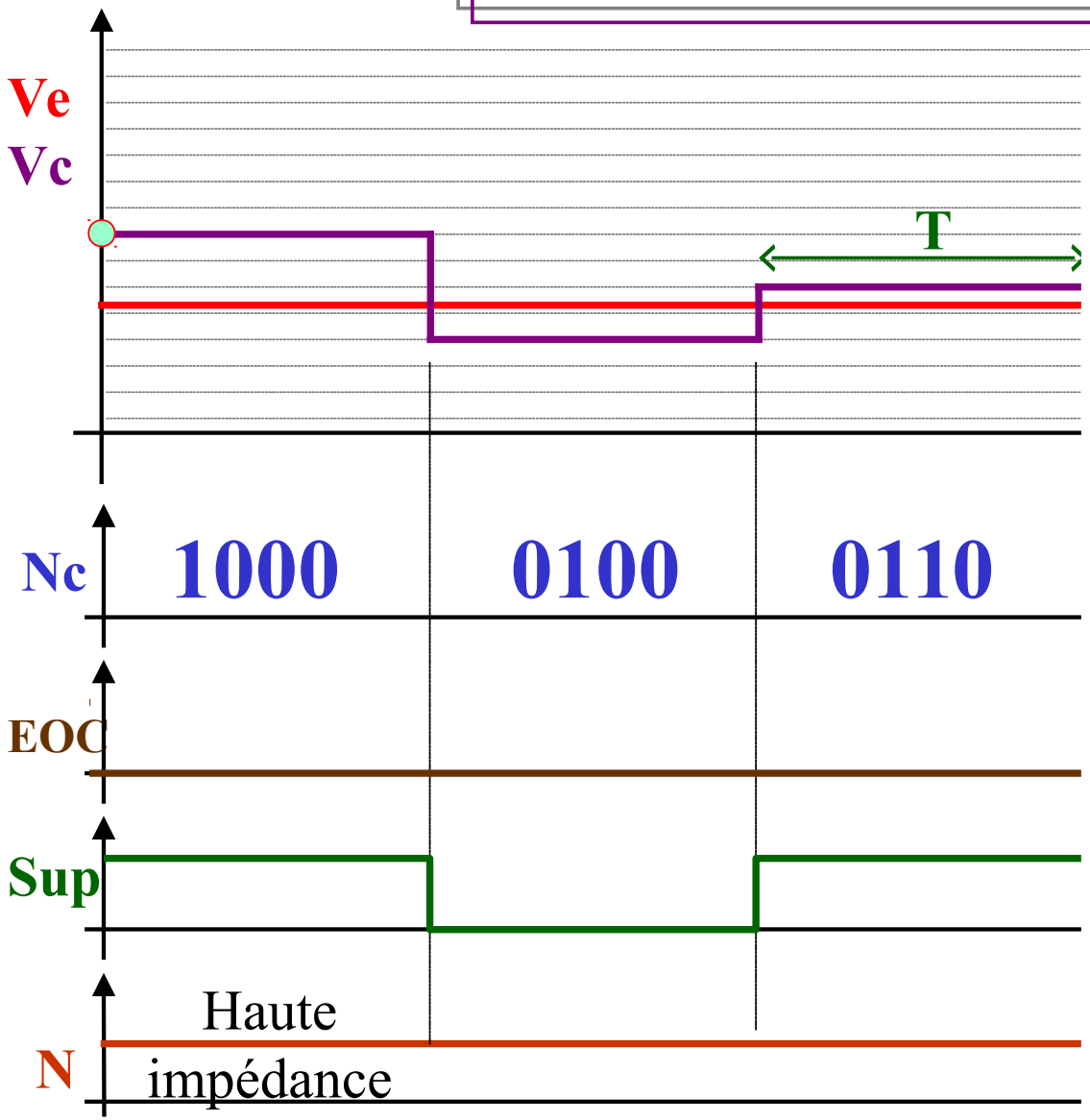
C.A.N. à approximations successives



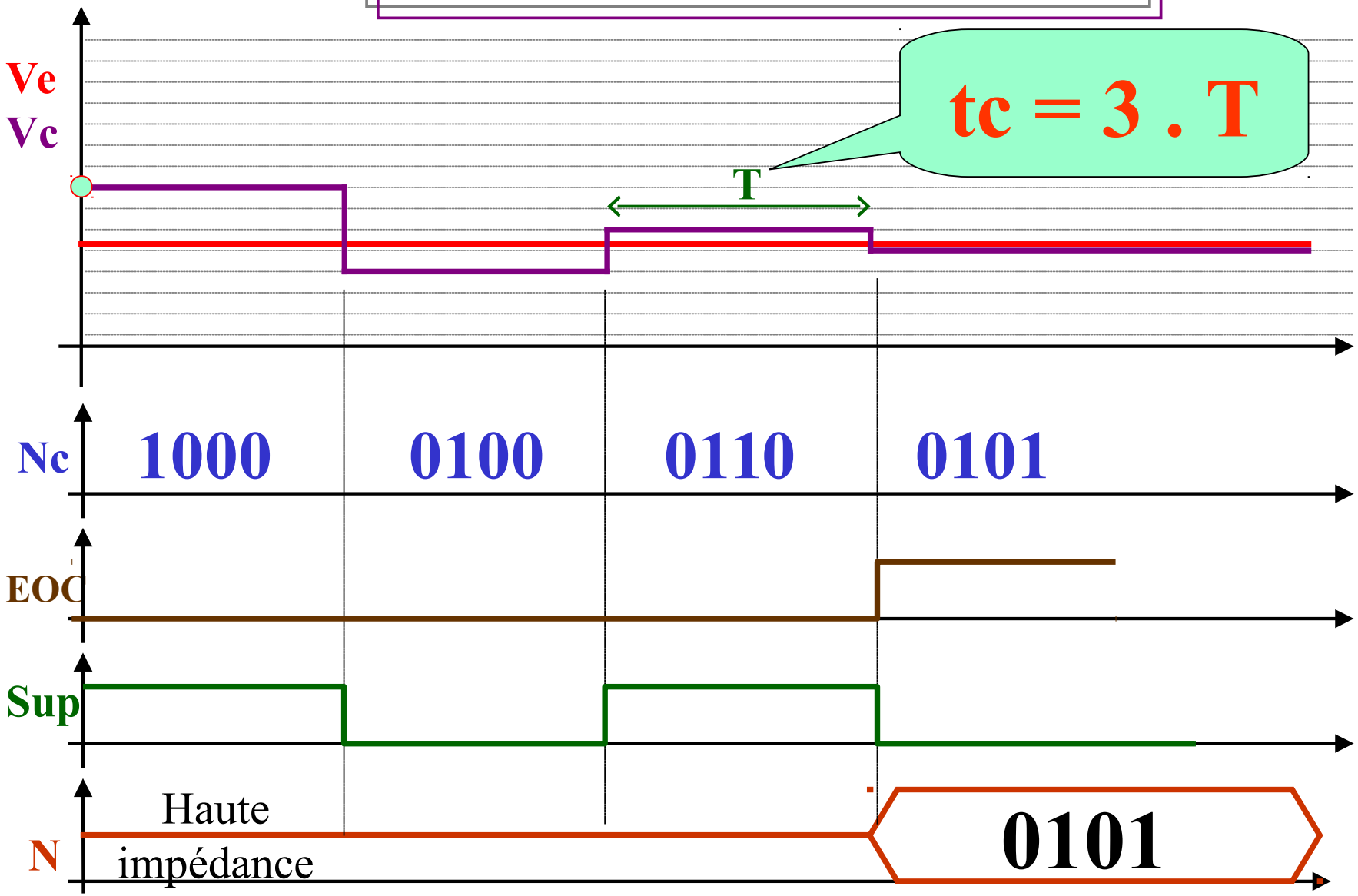
C.A.N. à approximations successives



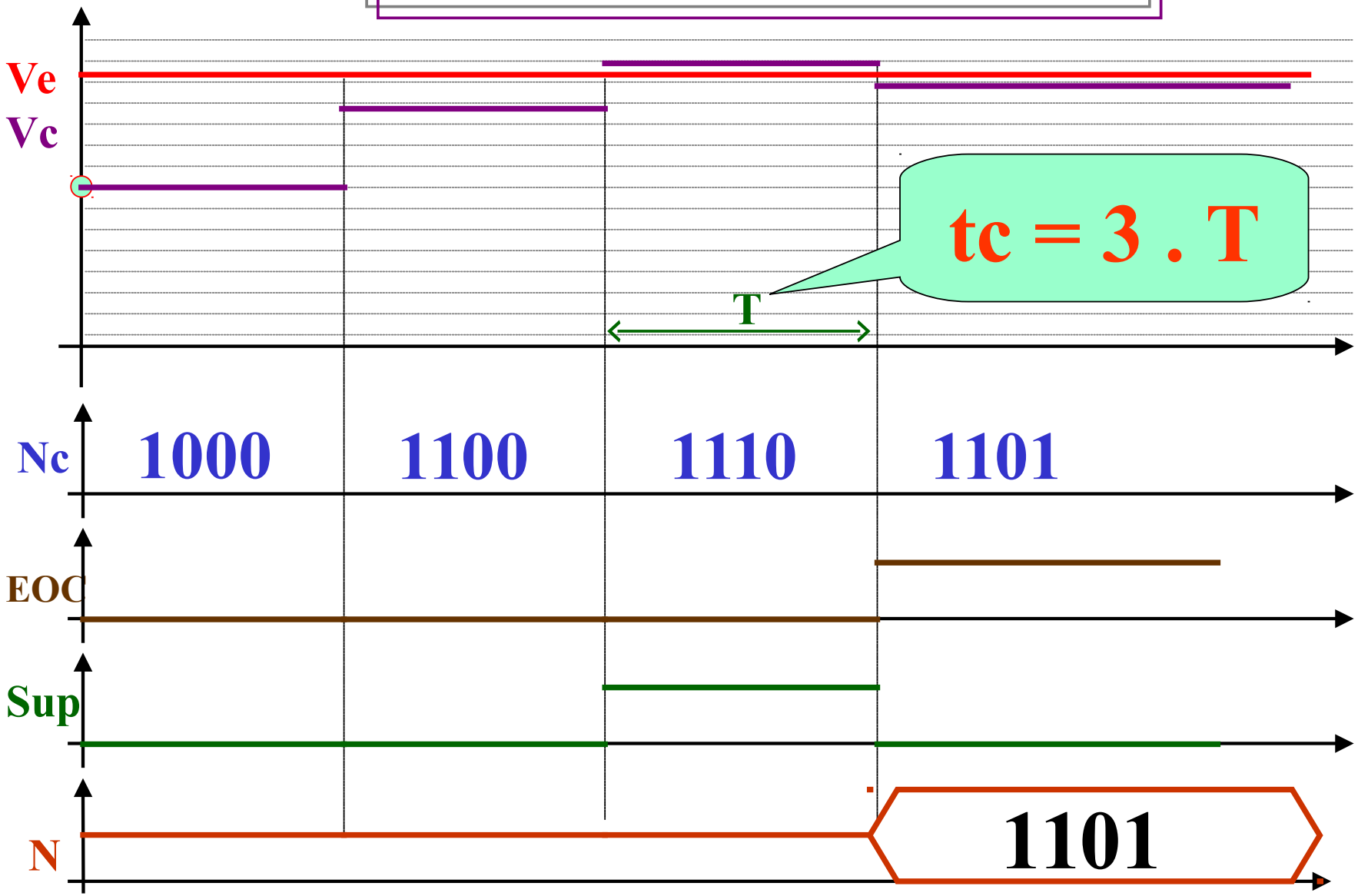
C.A.N. à approximations successives



C.A.N. à approximations successives



C.A.N. à approximations successives



Atmega 328p

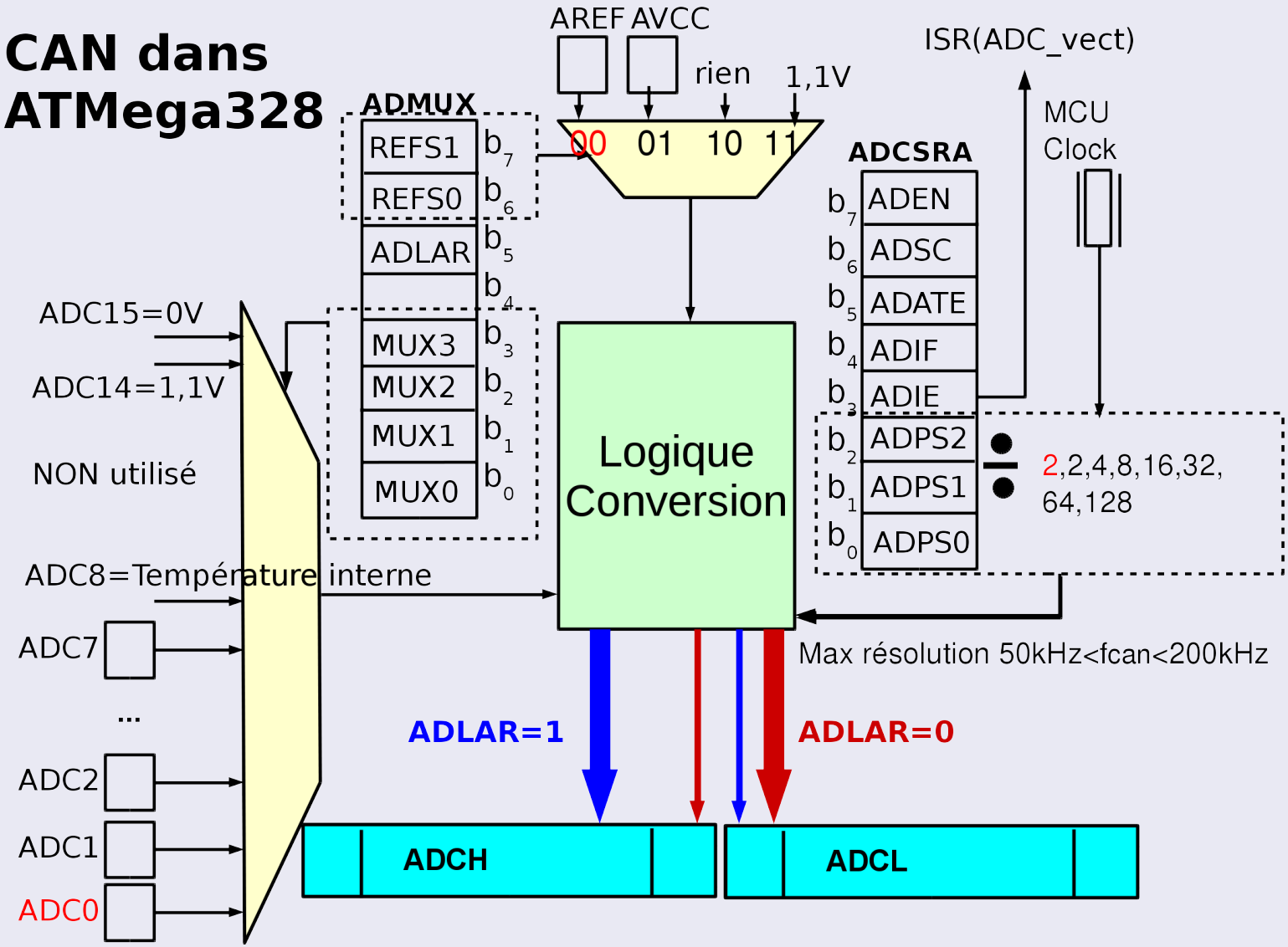

CAN dans ATMega328
AREF AVCC
rien
1,1V
ISR(ADC_vect)
MCU Clock
ADMUX
REFS1
REFS0
ADLAR
MUX3
MUX2
MUX1
MUX0
00 01 10 11
ADCSRA
ADEN
ADSC
ADATE
ADIF
ADIE
ADPS2
ADPS1
ADPS0
2,2,4,8,16,32, 64,128
Logique Conversion
ADC15=0V
ADC14=1,1V
NON utilisé
ADC8=Température interne
ADC7
...
ADC2
ADC1
ADC0
Max résolution 50kHz<fcan<200kHz
ADLAR=1
ADLAR=0
ADCH
ADCL

Table 24-4. Input Channel Selections

MUX3...0	Single Ended Input
0000	ADC0
0001	ADC1
0010	ADC2
0011	ADC3
0100	ADC4
0101	ADC5
0110	ADC6
0111	ADC7
1000	ADC8(1)
1001	(reserved)
1010	(reserved)
1011	(reserved)
1100	(reserved)
1101	(reserved)
1110	1.1V (V_{BG})
1111	0V (GND)

Note: 1. For Temperature Sensor.

24.9.2 ADCSRA – ADC Control and Status Register A

Bit	7	6	5	4	3	2	1	0	
(0x7A)	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – ADEN: ADC Enable**

Writing this bit to one enables the ADC. By writing it to zero, the ADC is turned off. Turning the ADC off while a conversion is in progress, will terminate this conversion.

- **Bit 6 – ADSC: ADC Start Conversion**

In Single Conversion mode, write this bit to one to start each conversion. In Free Running mode, write this bit to one to start the first conversion. The first conversion after ADSC has been written after the ADC has been enabled, or if ADSC is written at the same time as the ADC is enabled, will take 25 ADC clock cycles instead of the normal 13. This first conversion performs initialization of the ADC.

ADSC will read as one as long as a conversion is in progress. When the conversion is complete, it returns to zero. Writing zero to this bit has no effect.

- **Bit 5 – ADATE: ADC Auto Trigger Enable**

When this bit is written to one, Auto Triggering of the ADC is enabled. The ADC will start a conversion on a positive edge of the selected trigger signal. The trigger source is selected by setting the ADC Trigger Select bits, ADTS in ADCSRB.

When ADCL is read, the ADC Data Register is not updated until ADCH is read. Consequently, if the result is left adjusted and no more than 8-bit precision is required, it is sufficient to read ADCH. Otherwise, ADCL must be read first, then ADCH.

The ADLAR bit in ADMUX, and the MUXn bits in ADMUX affect the way the result is read from the registers. If ADLAR is set, the result is left adjusted. If ADLAR is cleared (default), the result is right adjusted.

- **ADC9:0: ADC Conversion Result**

These bits represent the result from the conversion, as detailed in "ADC Conversion Result" on page 247.

24.9.4 ADCSRB – ADC Control and Status Register B

Bit	7	6	5	4	3	2	1	0	
(0x7B)	–	ACME	–	–	–	ADTS2	ADTS1	ADTS0	ADCSRB
Read/Write	R	R/W	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7, 5:3 – Reserved**

These bits are reserved for future use. To ensure compatibility with future devices, these bits must be written to zero when ADCSRB is written.

- **Bit 2:0 – ADTS[2:0]: ADC Auto Trigger Source**

If ADATE in ADCSRA is written to one, the value of these bits selects which source will trigger an ADC conversion. If ADATE is cleared, the ADTS[2:0] settings will have no effect. A conversion will be triggered by the rising edge of the selected Interrupt Flag. Note that switching from a trigger source that is cleared to a trigger source that is set, will generate a positive edge on the trigger signal. If ADEN in ADCSRA is set, this will start a conversion. Switching to Free Running mode (ADTS[2:0]=0) will not cause a trigger event, even if the ADC Interrupt Flag is set.

Table 24-6. ADC Auto Trigger Source Selections

ADTS2	ADTS1	ADTS0	Trigger Source
0	0	0	Free Running mode
0	0	1	Analog Comparator
0	1	0	External Interrupt Request 0
0	1	1	Timer/Counter0 Compare Match A
1	0	0	Timer/Counter0 Overflow
1	0	1	Timer/Counter1 Compare Match B
1	1	0	Timer/Counter1 Overflow
1	1	1	Timer/Counter1 Capture Event

24.9.5 DIDR0 – Digital Input Disable Register 0

Bit	7	6	5	4	3	2	1	0	
(0x7E)	–	–	ADC5D	ADC4D	ADC3D	ADC2D	ADC1D	ADC0D	DIDR0
Read/Write	R	R	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7:6 – Reserved**

These bits are reserved for future use. To ensure compatibility with future devices, these bits must be written to zero when DIDR0 is written.

Sommaire

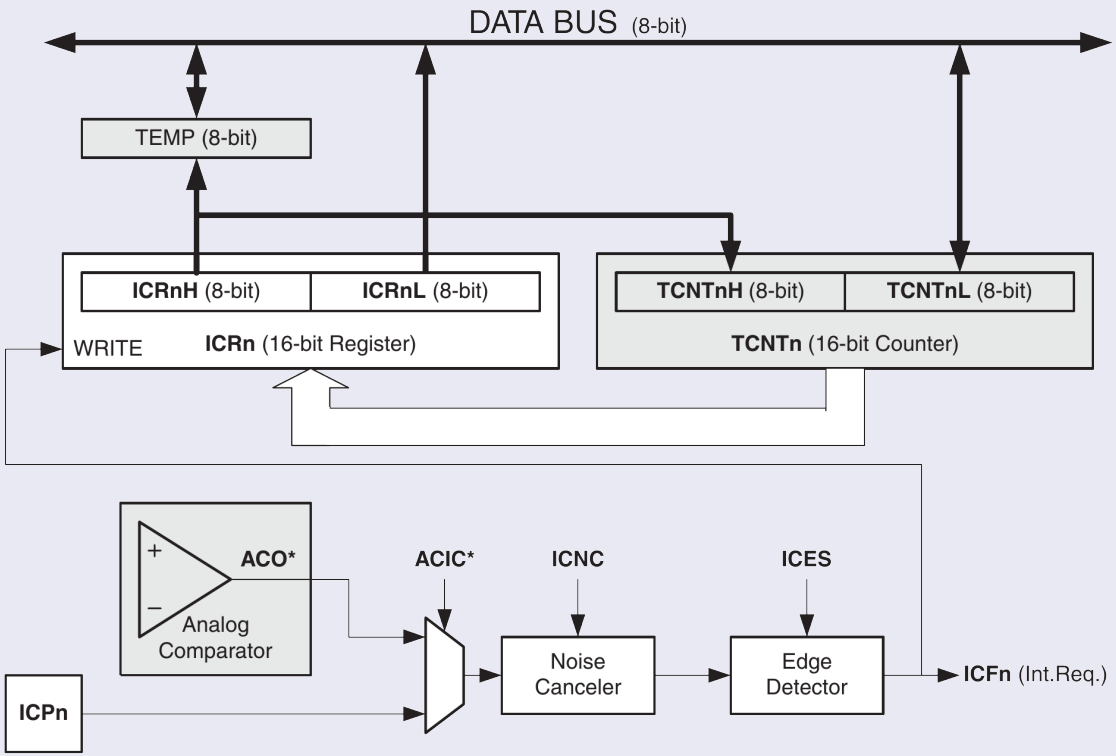
9 Les "modes" d'entrées
- 3 types de signaux
- CAN
- mesure de temps

temps donc ... TIMER !

- fréquence de comptage
- valeur maximum

Mode capture

Input Capture Unit Block Diagram



bit	reg	=1
ACIC	ACSR	source AC0
ICNC1	TCCR1B	noise remove
ICES1	TCCR1B	rising edge
ICIE1	TIMSK1	ISR TIMER1_CAPT

Sommaire

1 Introduction

2 Entrée, sortie ?

3 Modifier une sortie

4 Lire une entrée

5 Cas des bps

6 Gestion d'événement externe

7 Événement périodique

8 MLI/PWM

9 Les "modes" d'entrées

10 Transmission de données

11 Configuration du µcontrôleur

12 Choix microcontroleur

Sommaire

10 Transmission de données
- Caractérisation
- UART / USART
- I2C
- SPI
- ONEWIRE

classification des liaisons

- parallèle // série
- synchrone // asynchrone
- simplex // duplex (half full)

performance

- débit
- latence
- nombre de périphériques

Sommaire

10 Transmission de données
- Caractérisation
- UART / USART
- I2C
- SPI
- ONEWIRE

Description

Vidéo UART

Description

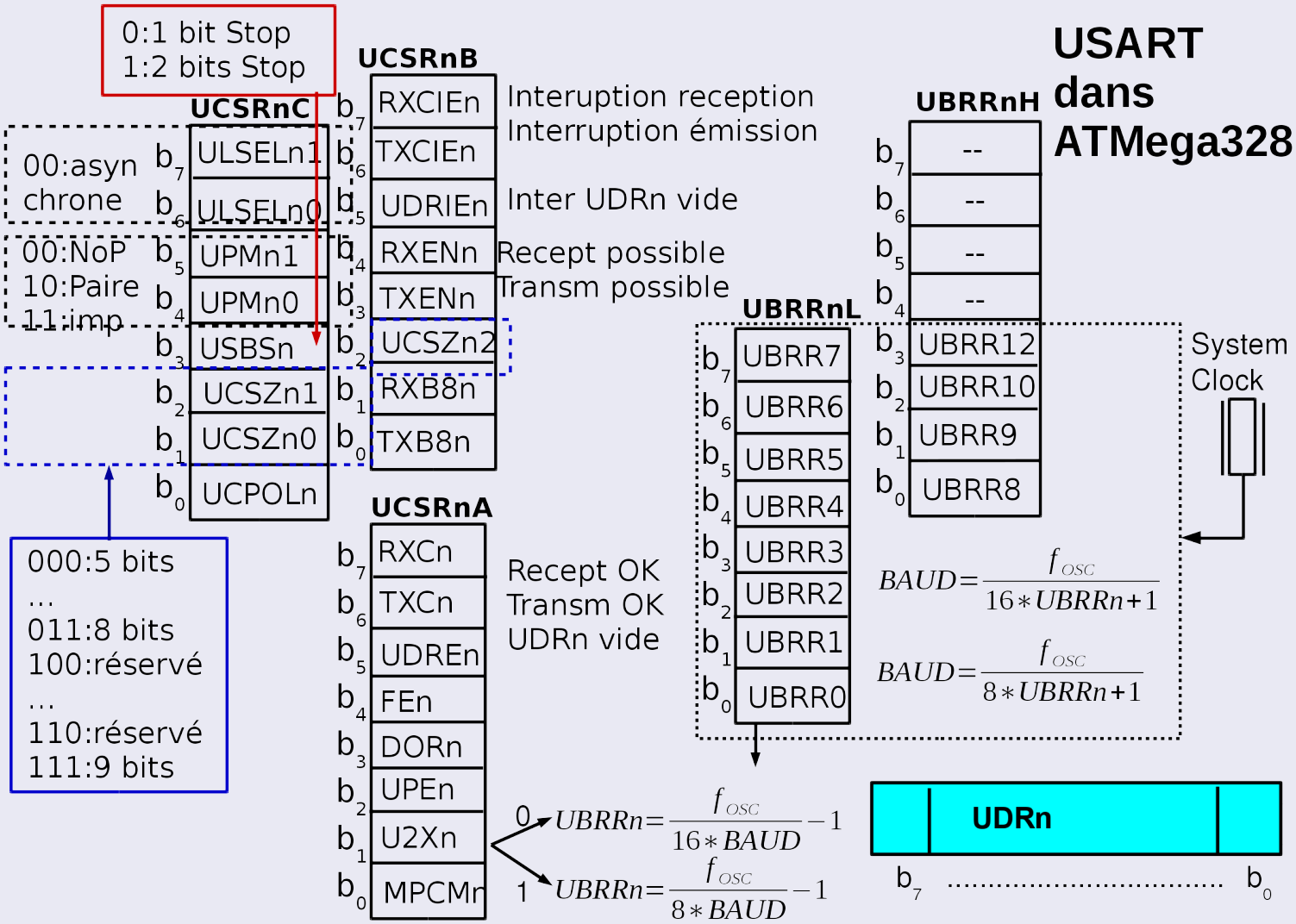
Vidéo UART

Caractéristiques

- débit
- Nbre de bits de données
- Parité
- Nbre de bits de Stop

Registres Atmega328p

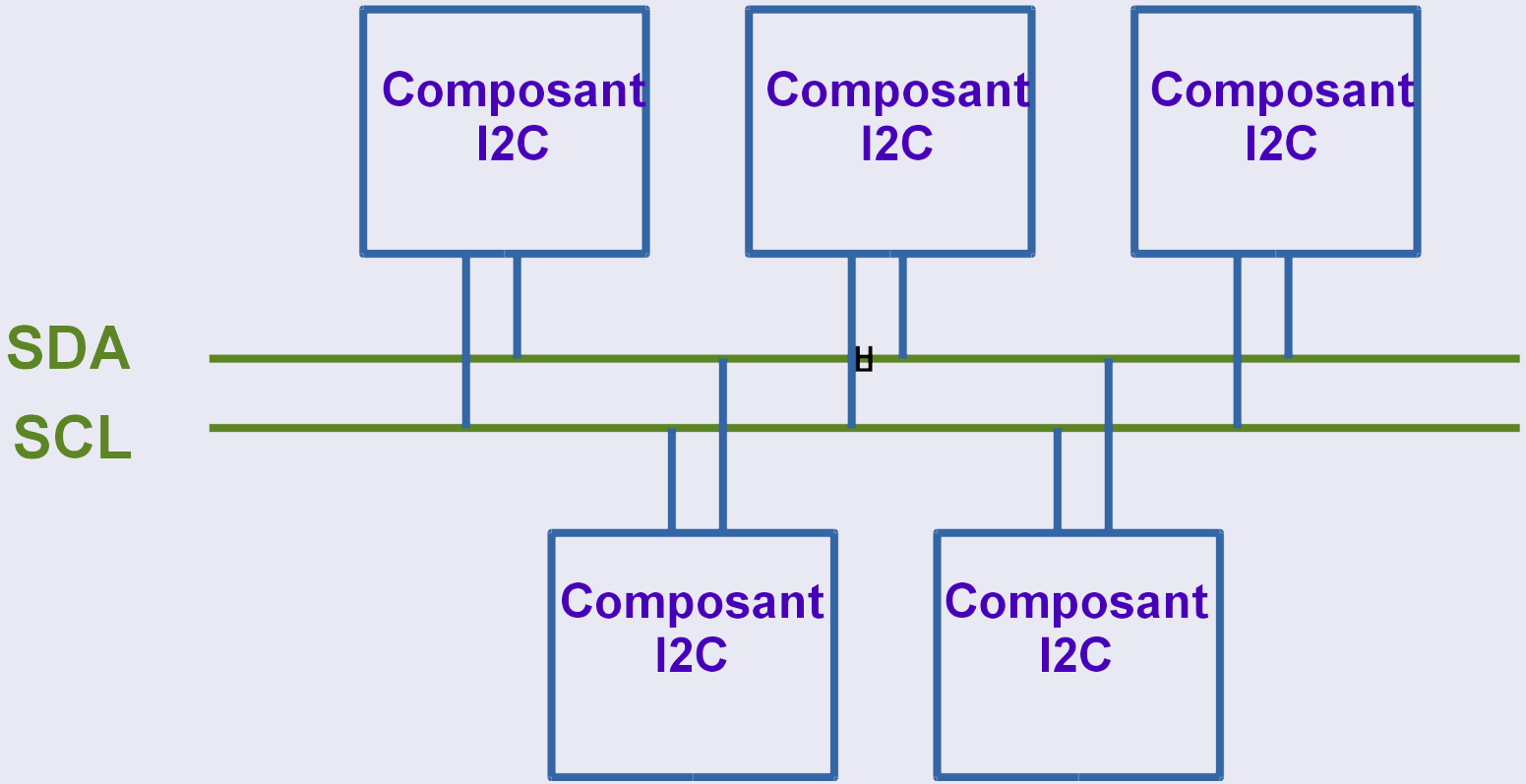
RS232 : USART dans ATMega328



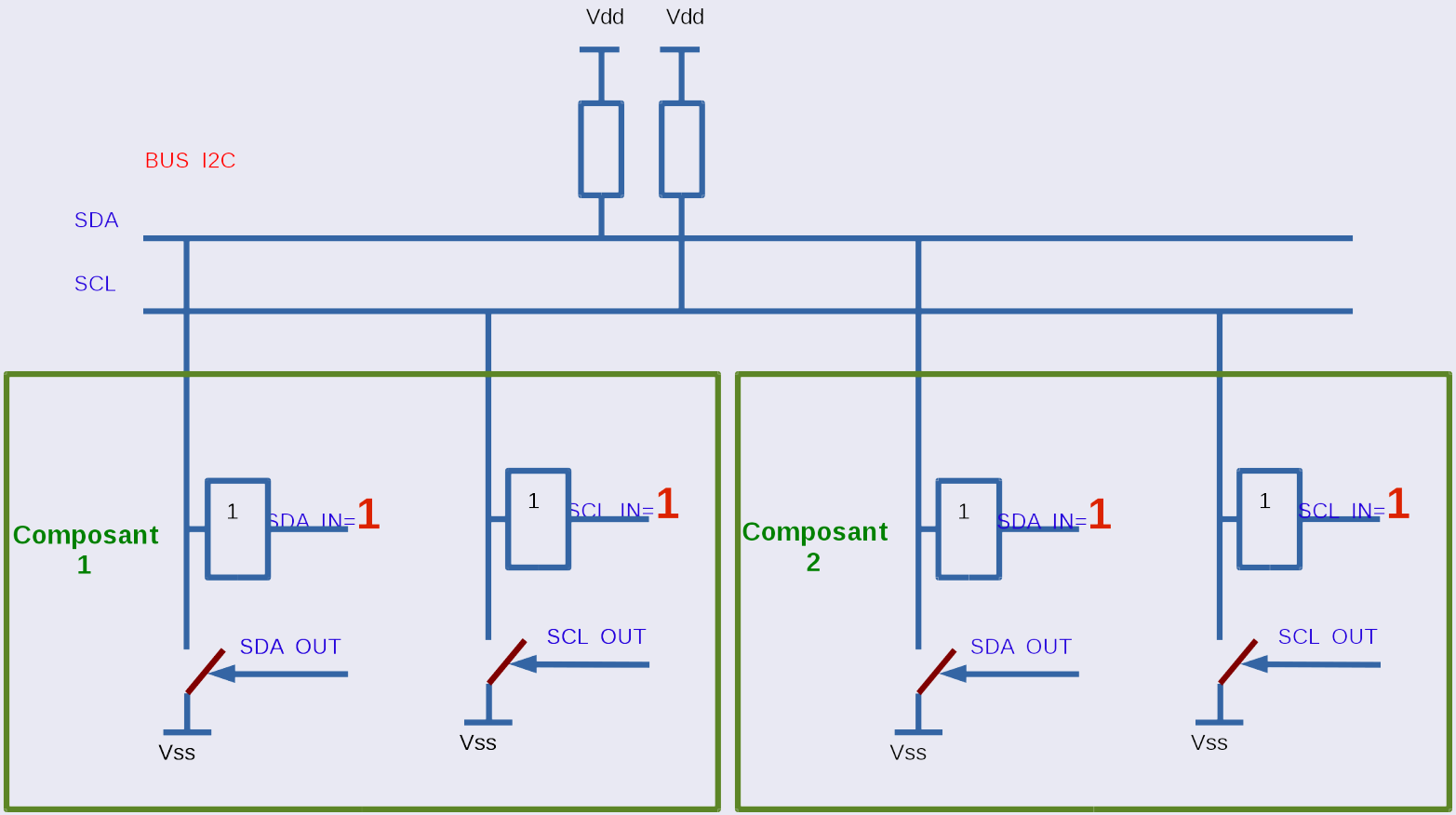
Sommaire

10 Transmission de données
- Caractérisation
- UART / USART
- I2C
- SPI
- ONEWIRE

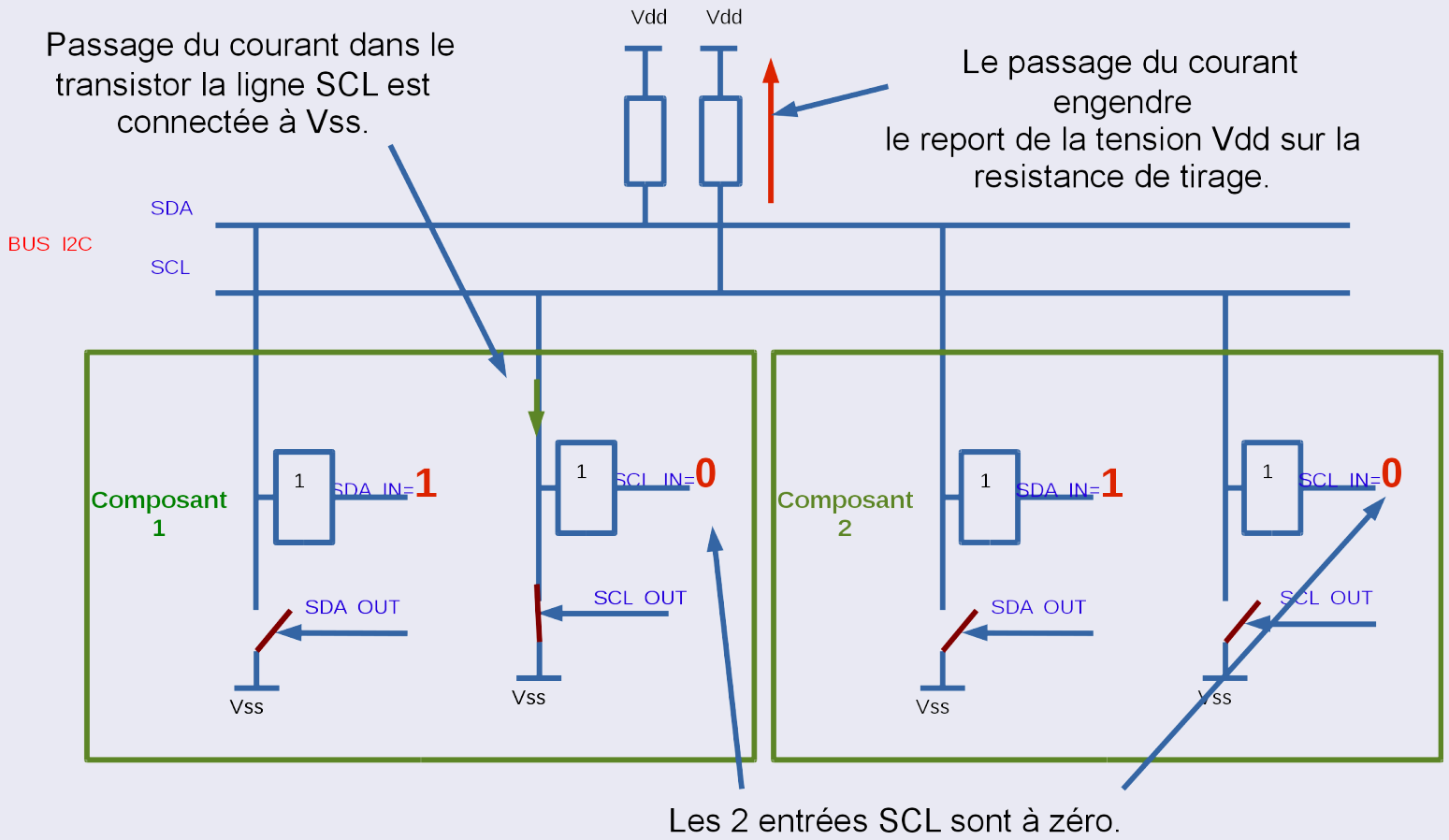
Bus I2c


Composant
I2C
Composant
I2C
Composant
I2C
SDA
SCL
Composant
I2C
Composant
I2C

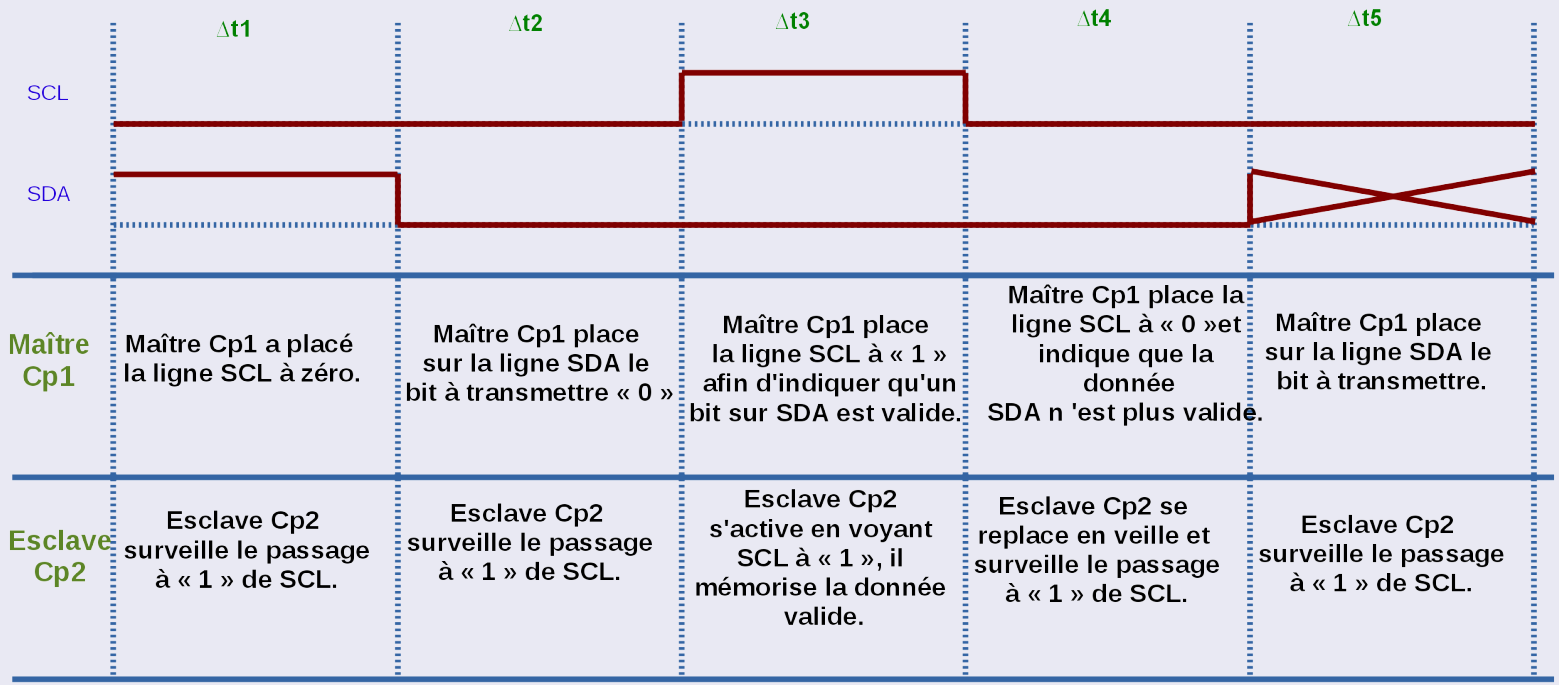
Comportement des composants I2C



Mise à 0 de la ligne

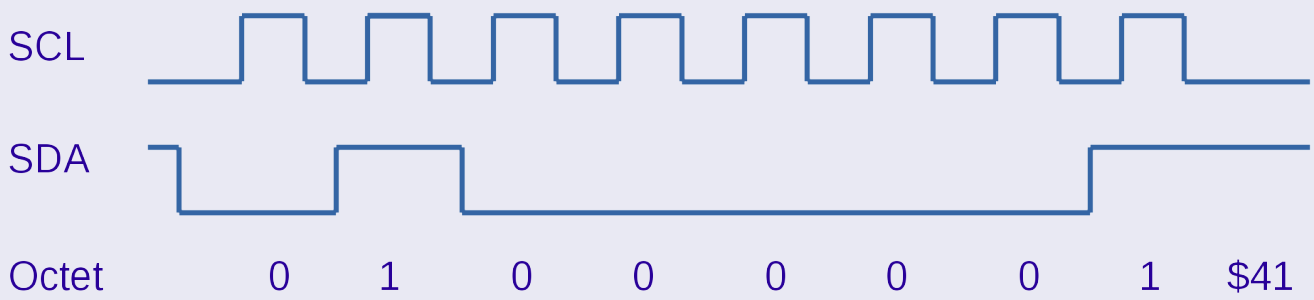


transmission d'un bit



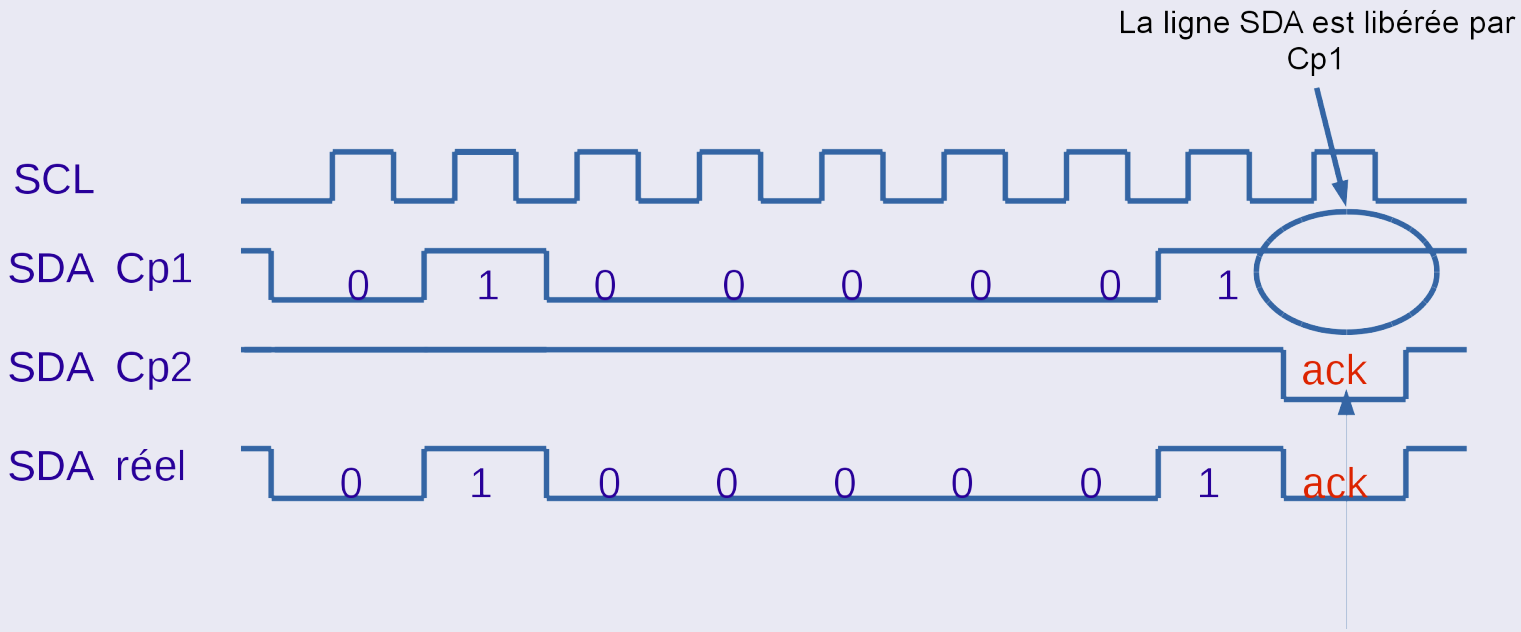
1 octet

- Le maître génère le signal d'horloge
- Les données sont transmises en série
- il y a un décalage entre SDA et SCL : on prépare la donnée avant le signal d'horloge



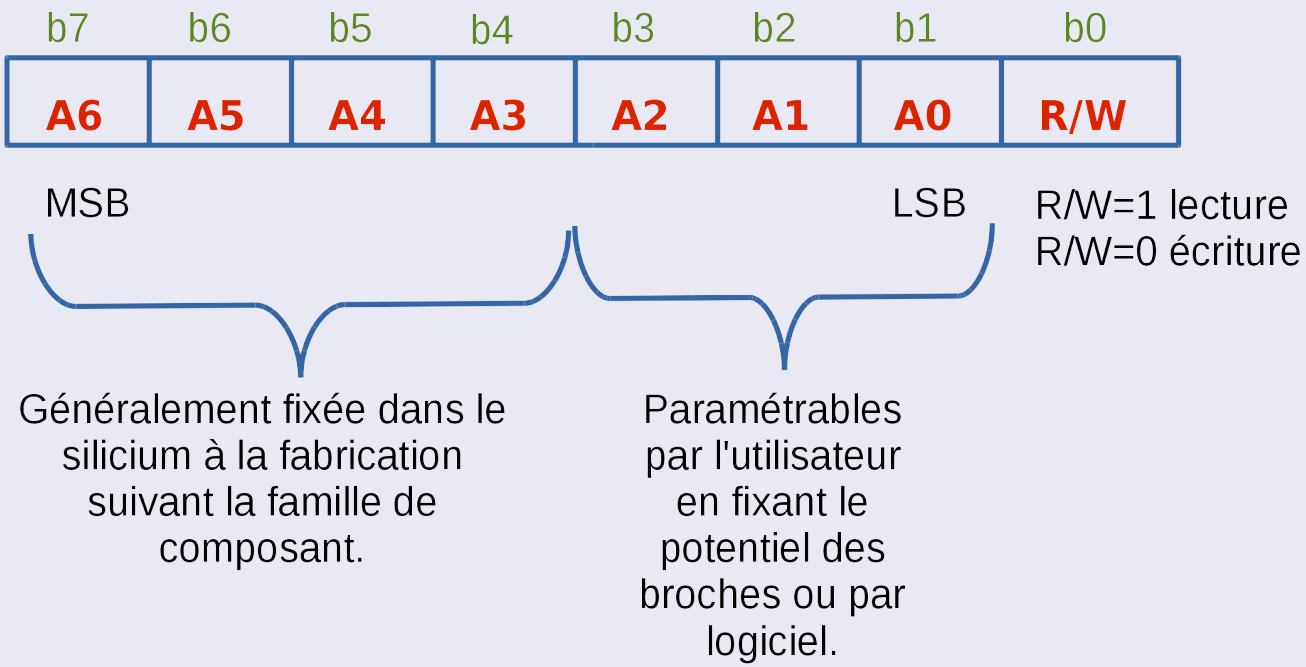
Acknowledge (acquittement)

- c'est un accusé de réception
- il est généré par le composant qui réceptionne les données
- sur le chronogramme suivant Cp1 et Cp2 sont des signaux virtuels



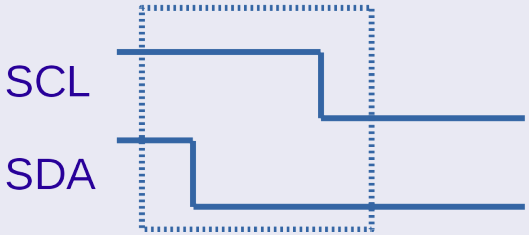
Plusieurs composants

- Choix du destinataire
- Mode d'accès
- L'esclave doit acquitter seulement si le message lui est destiné.

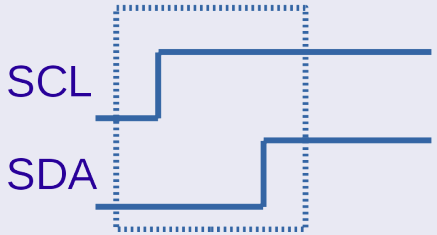


Début/Fin de trame

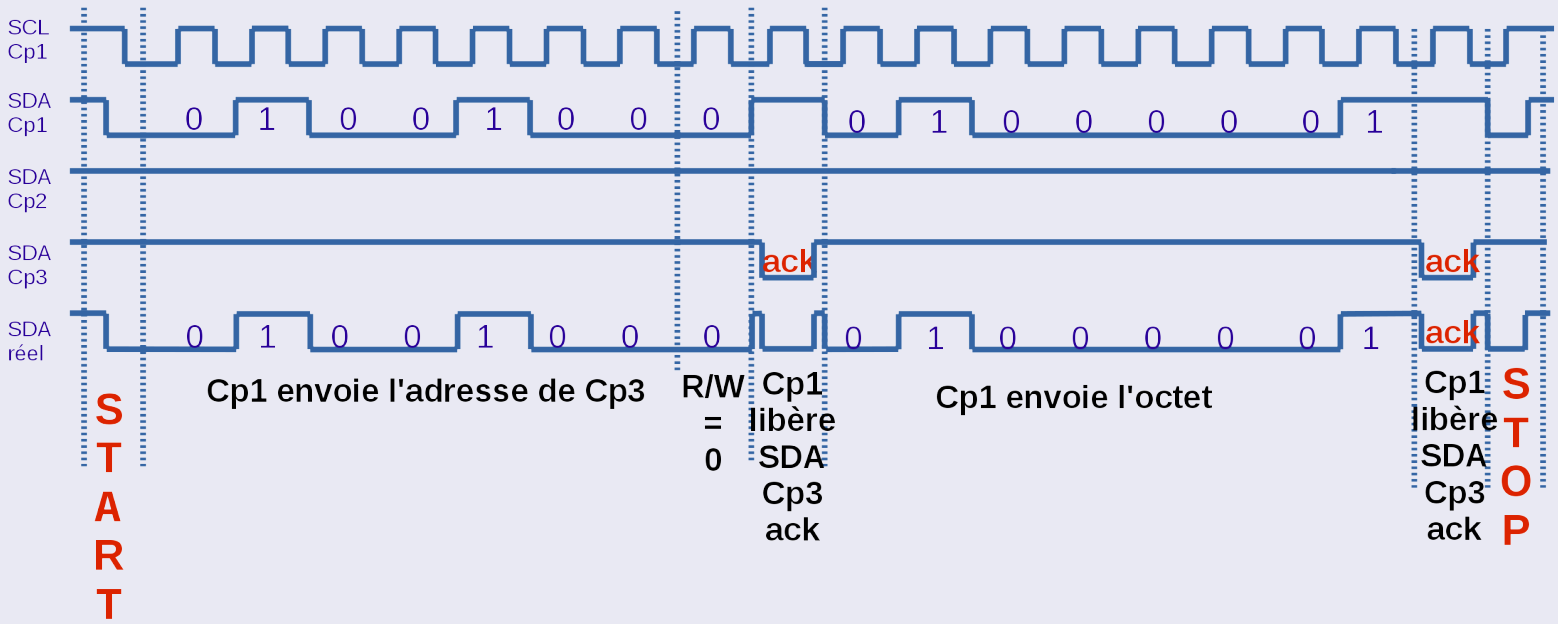
Est défini comme condition de départ (start condition) le passage de SDA de l'état haut à l'état bas lorsque SCL est au niveau haut.



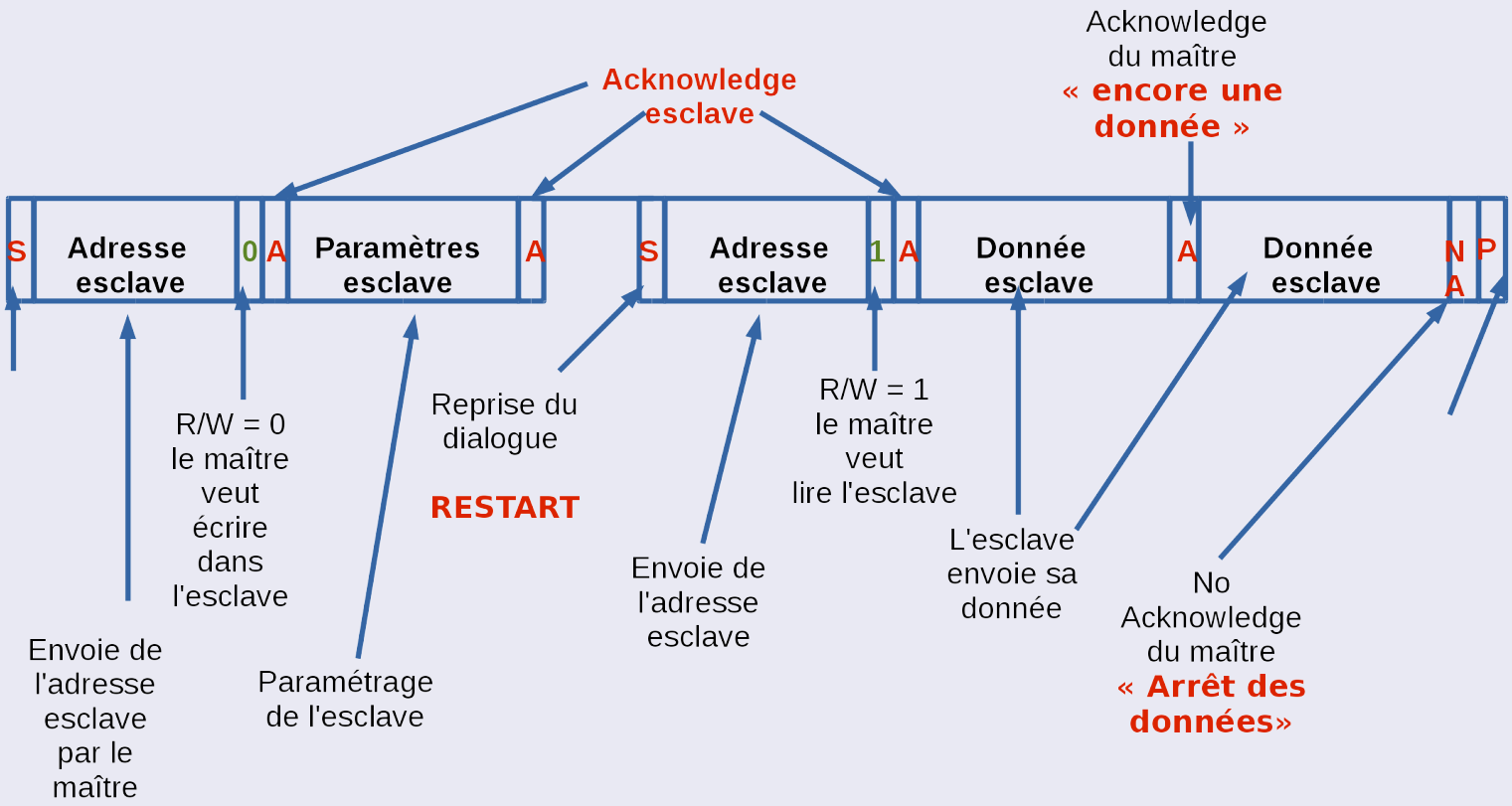
Est défini comme condition de stop (stop condition) le passage de SDA de l'état bas à l'état haut lorsque SCL se trouve au niveau haut.



Trame complète



Trame complète



Caractérisation d'une liaison I2C

- série / parallèle
- synchrone / asynchrone
- simplex / full duplex / half duplex
- liaison point à point / bus (nbre de périphériques maximum)
- débit

Sommaire

10 Transmission de données
- Caractérisation
- UART / USART
- I2C
- SPI
- ONEWIRE

liaison SPI

- série
- synchrone
- full duplex
- slave select

Sommaire

10 Transmission de données
- Caractérisation
- UART / USART
- I2C
- SPI
- ONEWIRE

Sommaire

1 Introduction

2 Entrée, sortie ?

3 Modifier une sortie

4 Lire une entrée

5 Cas des bps

6 Gestion d'événement externe

7 Événement périodique

8 MLI/PWM

9 Les "modes" d'entrées

10 Transmission de données

11 Configuration du µcontrôleur

12 Choix microcontroleur

FUSE byte : high low extended

- type/fréquence d'horloge
- durée de démarrage
- configuration du reset
- bootloader configuration
- reset sur variation de tension
- port de débogage
- watchdog

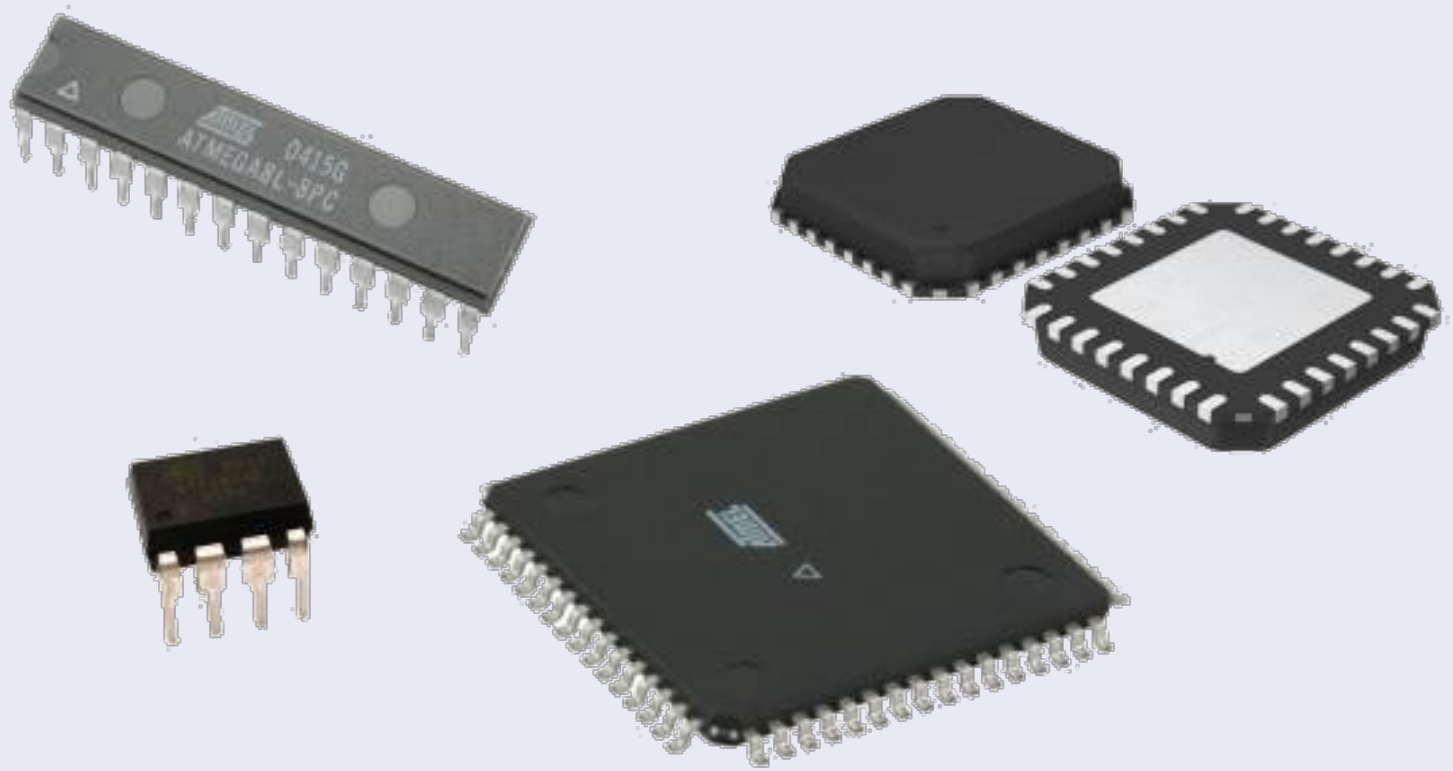
Lock byte

- protection de la mémoire programme
- protection de l'EEPROM
- protection de la fonction self-programming

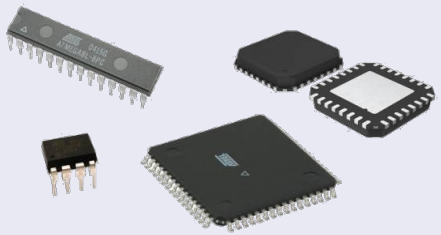
Sommaire

1. Introduction
2. Entrée, sortie ?
3. Modifier une sortie
4. Lire une entrée
5. Cas des bps
6. Gestion d'événement externe
7. Événement périodique
8. MLI/PWM
9. Les "modes" d'entrées
10. Transmission de données
11. Configuration du µcontrôleur
12. Choix microcontroleur

Famille de µcontrôleur



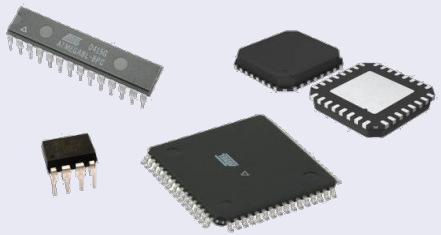
Famille de µcontrôleur



La taille dépend

- Du type de boîtier (CMS/Traversant)

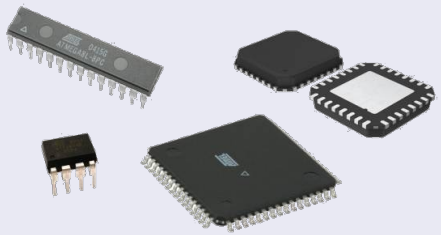
Famille de µcontrôleur



La taille dépend

- Du type de boîtier (CMS/Traversant)
- Nombre d'E/S

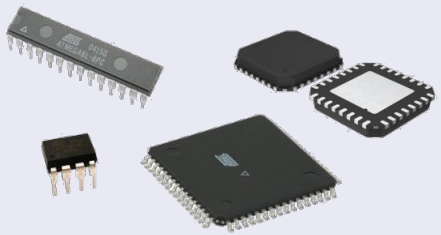
Famille de µcontrôleur



La taille dépend

- Du type de boîtier (CMS/Traversant)
- Nombre d'E/S
- Quantité de mémoire (FLASH SRAM EEPROM)

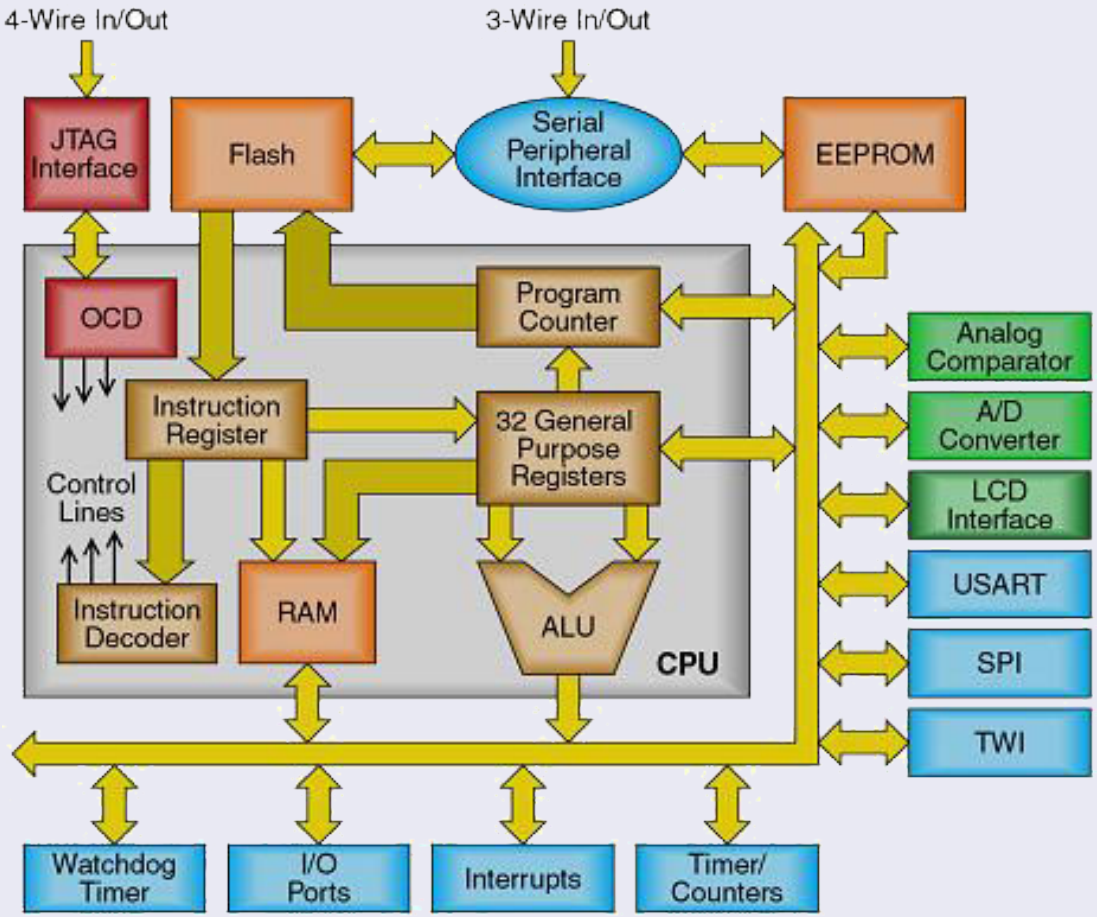
Famille de µcontrôleur



La taille dépend

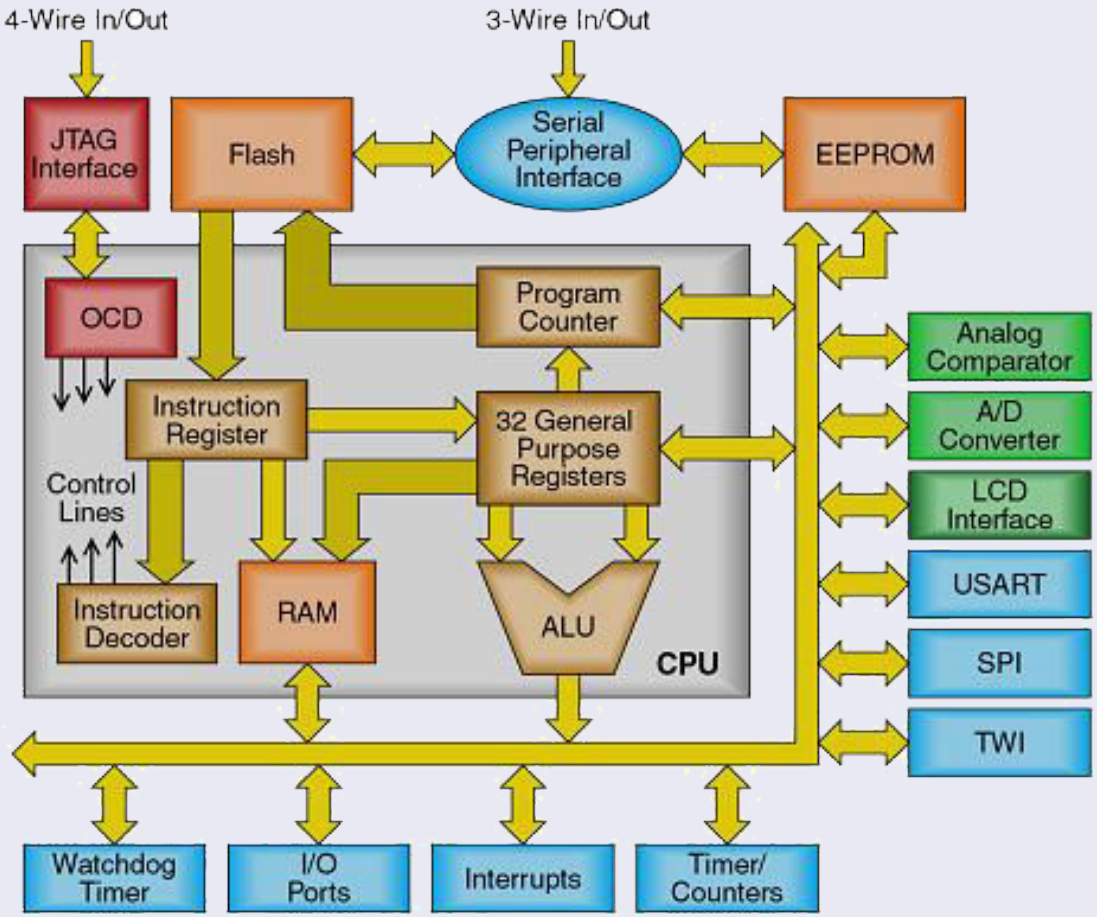
- Du type de boîtier (CMS/Traversant)
- Nombre d'E/S
- Quantité de mémoire (FLASH SRAM EEPROM)
- Périphériques (TIMER CAN ...)

Architecture d'un µcontrôleur



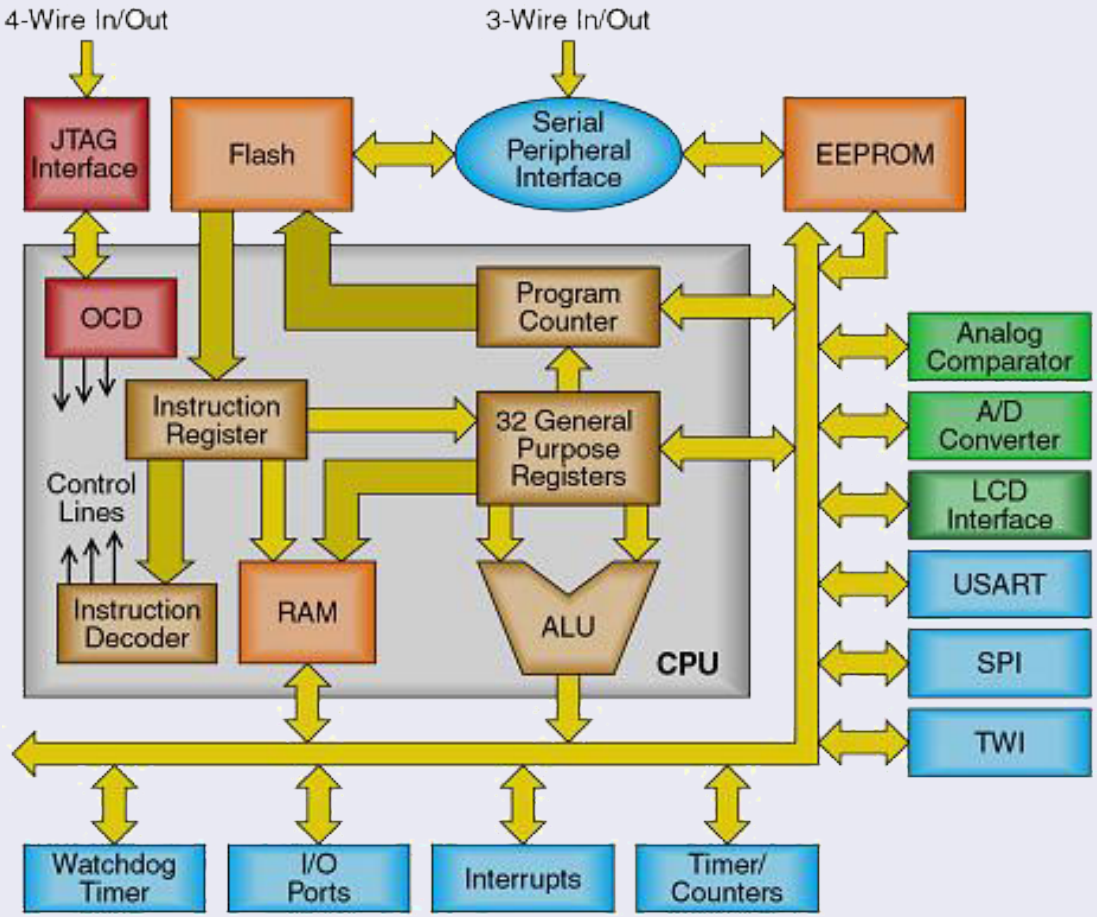
- CPU

Architecture d'un µcontrôleur



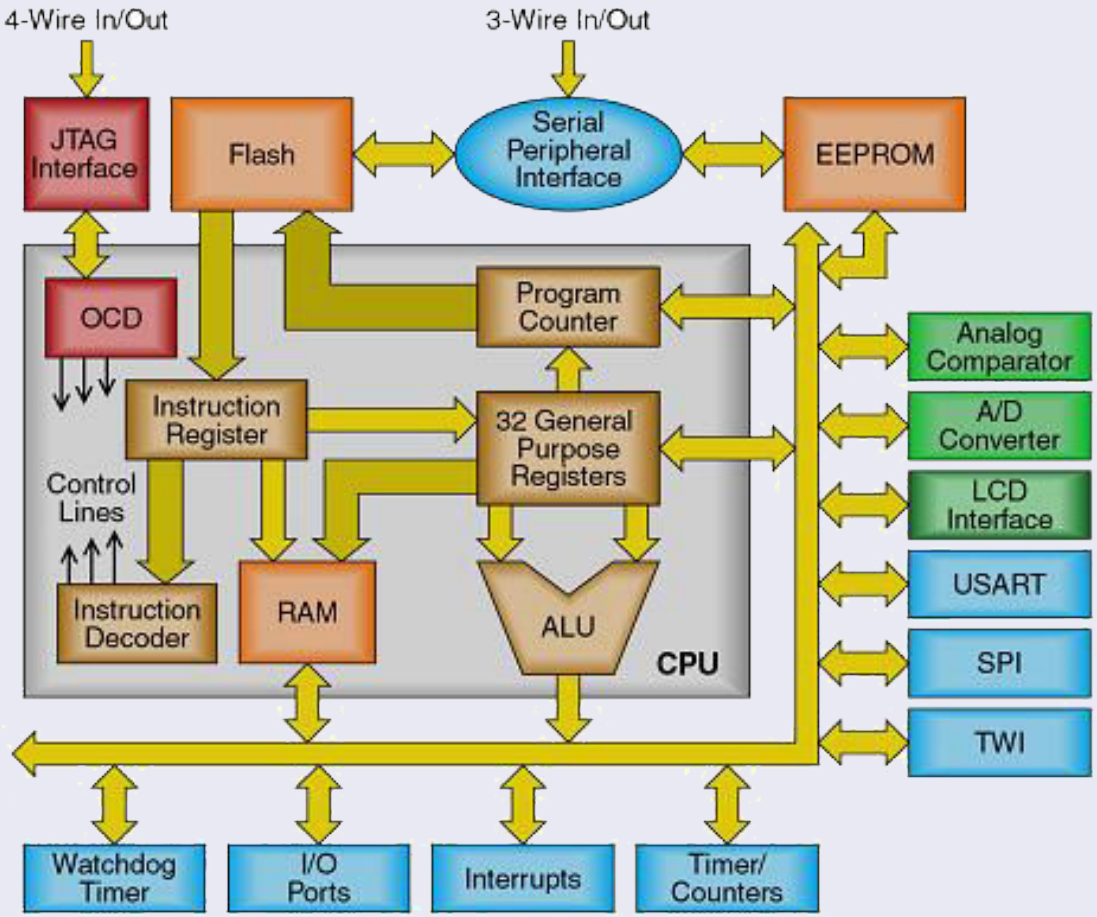
- CPU
- Horloge

Architecture d'un µcontrôleur



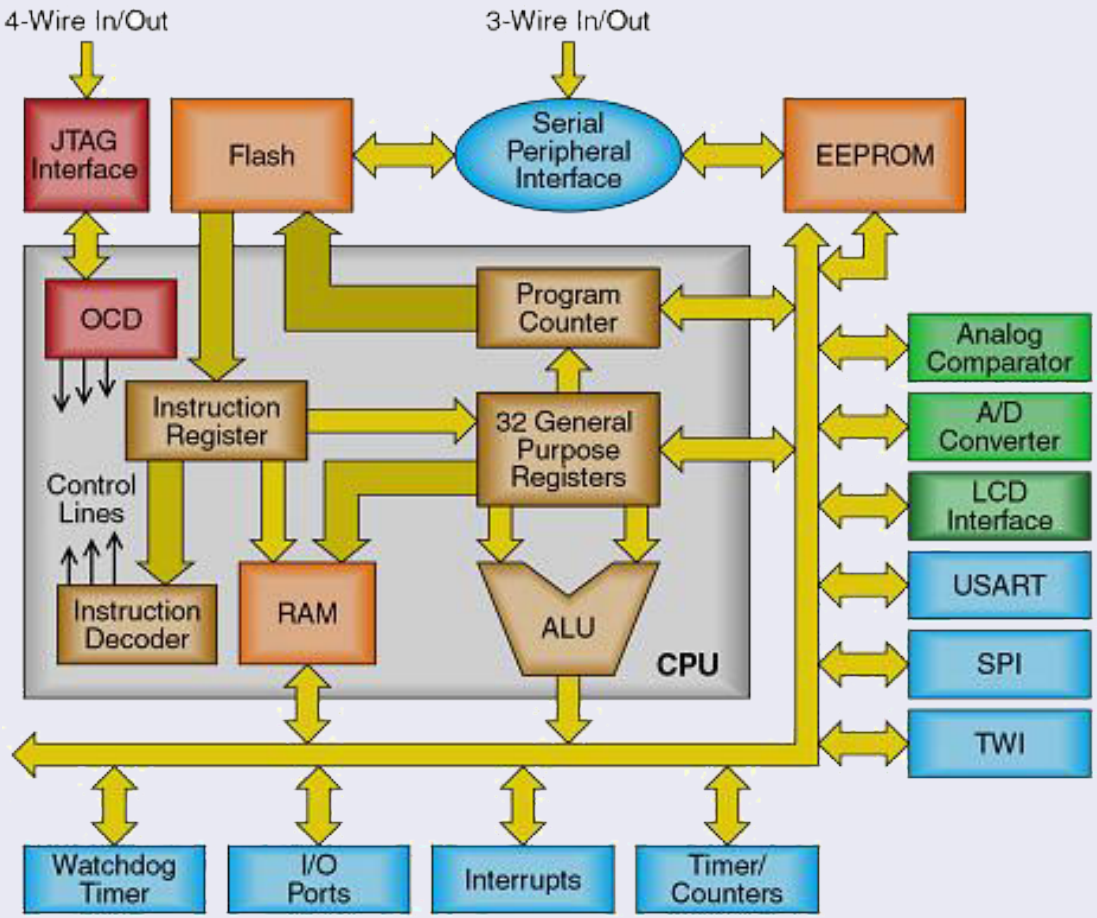
- CPU
- Horloge
- bus de données (8 bits)

Architecture d'un µcontrôleur



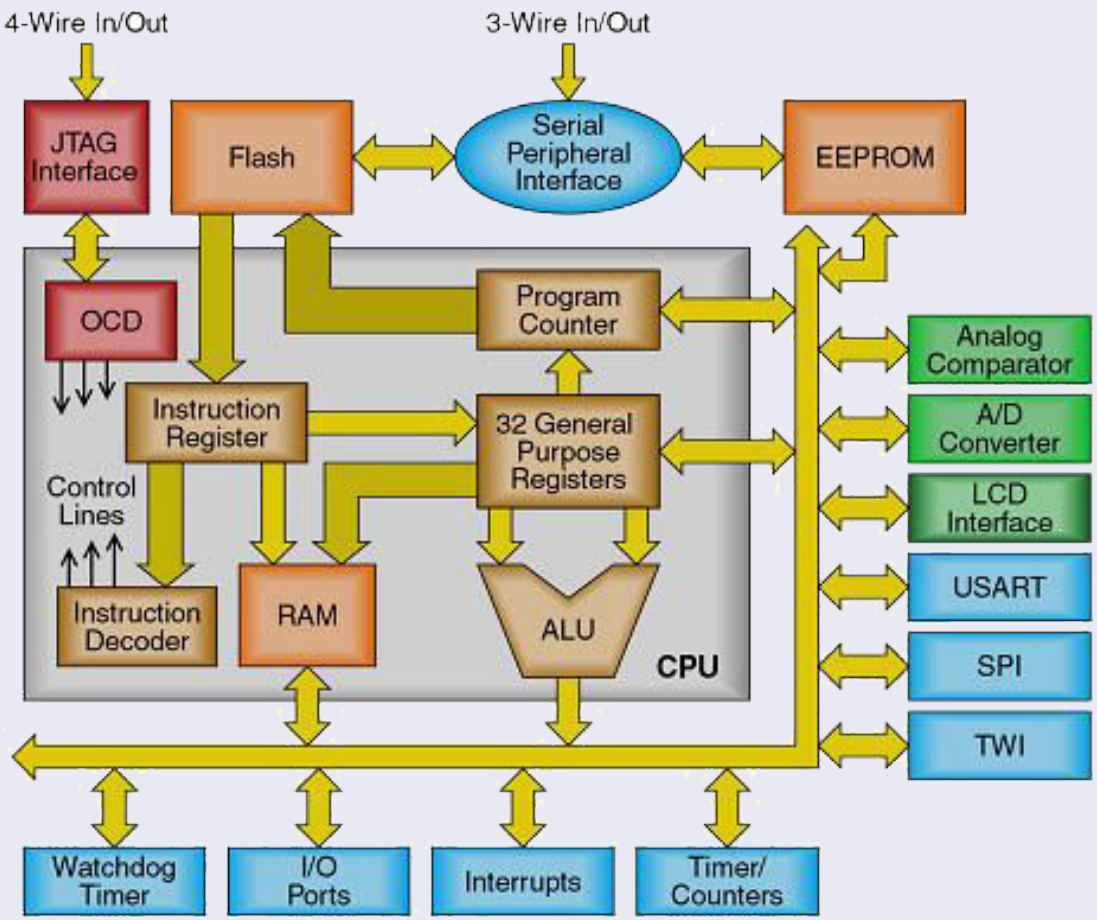
- CPU
- Horloge
- bus de données (8 bits)
- Mémoires

Architecture d'un µcontrôleur



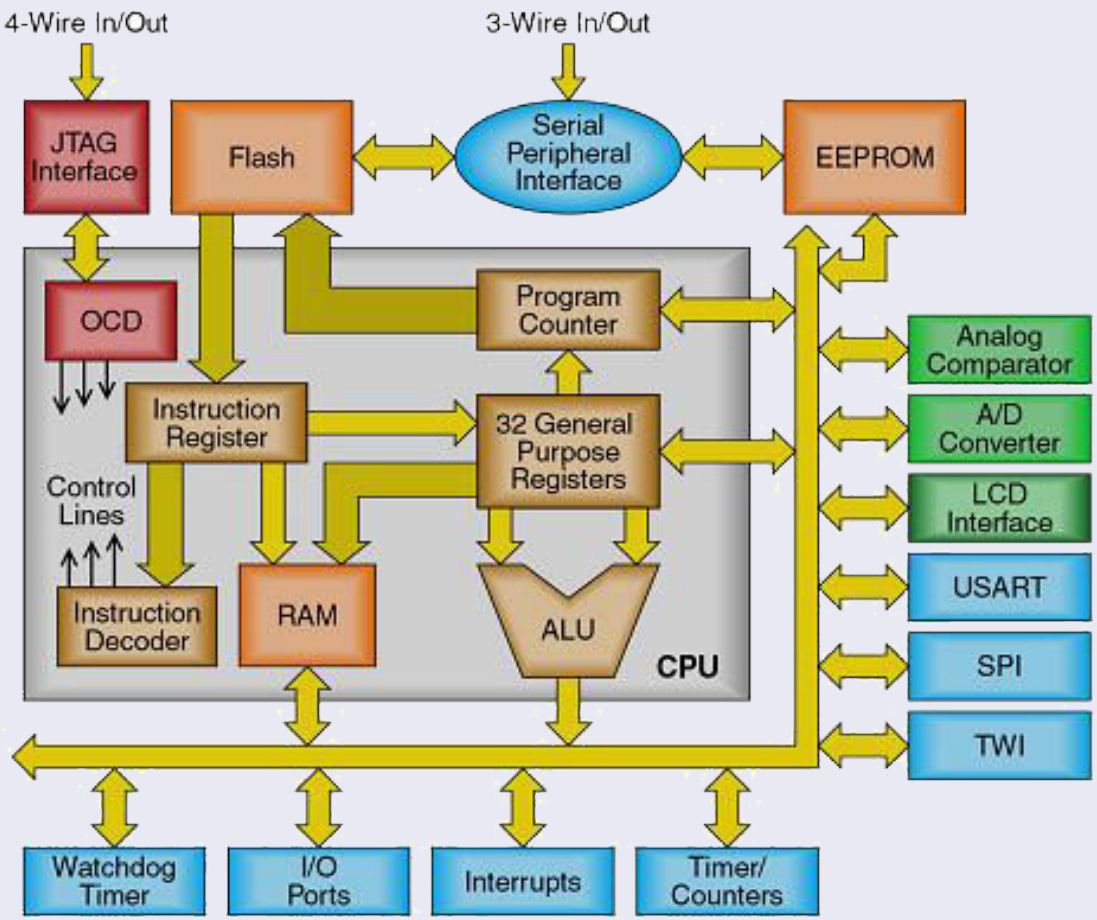
- CPU
- Horloge
- bus de données (8 bits)
- Mémoires
- I/O Ports

Architecture d'un µcontrôleur



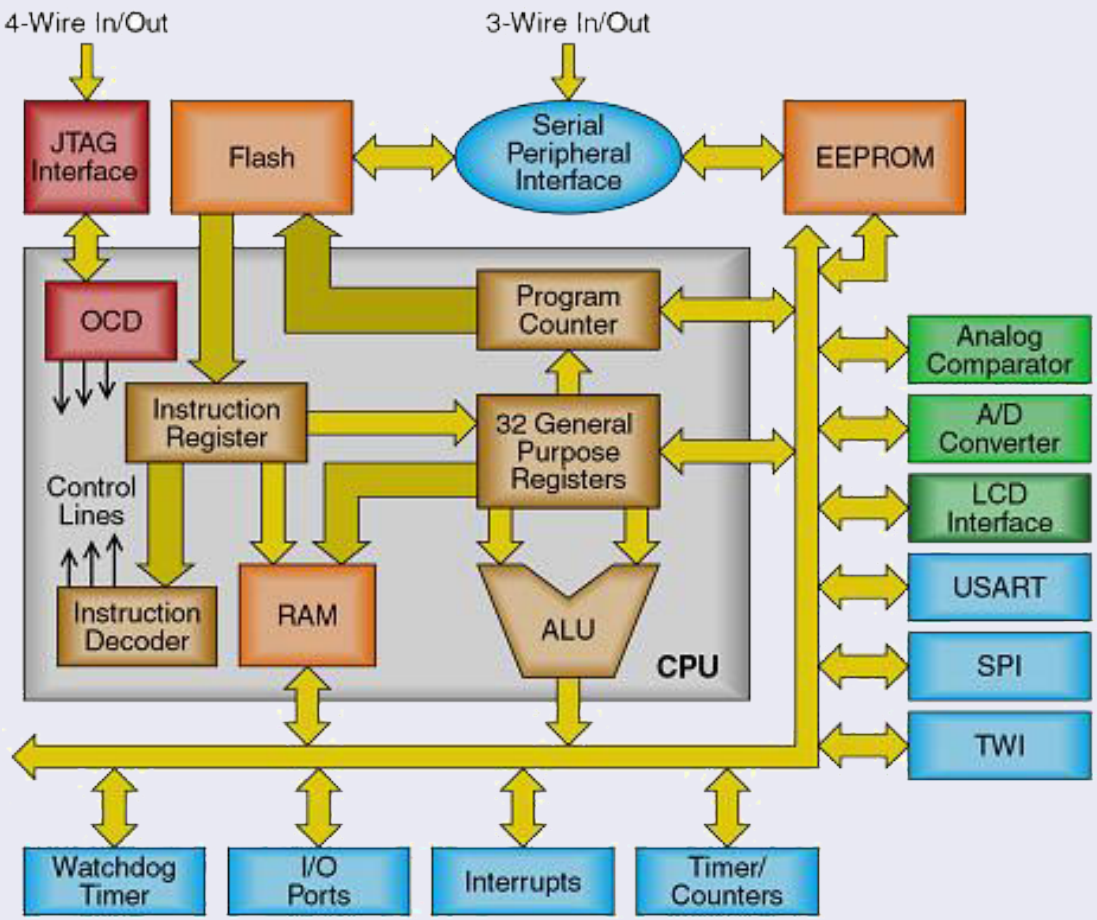
- CPU
- Horloge
- bus de données (8 bits)
- Mémoires
- I/O Ports
- Interruptions

Architecture d'un µcontrôleur



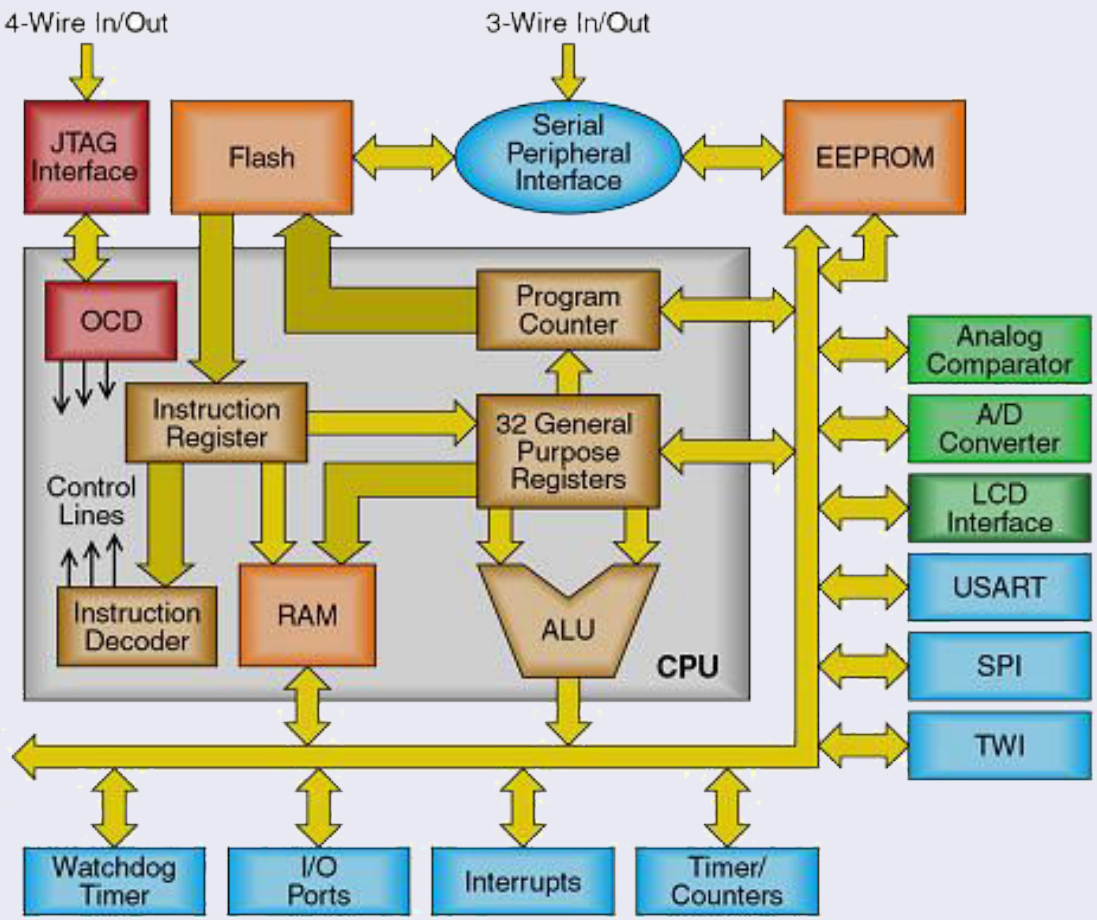
- CPU
- Horloge
- bus de données (8 bits)
- Mémoires
- I/O Ports
- Interruptions
- Circuit de reset

Architecture d'un µcontrôleur

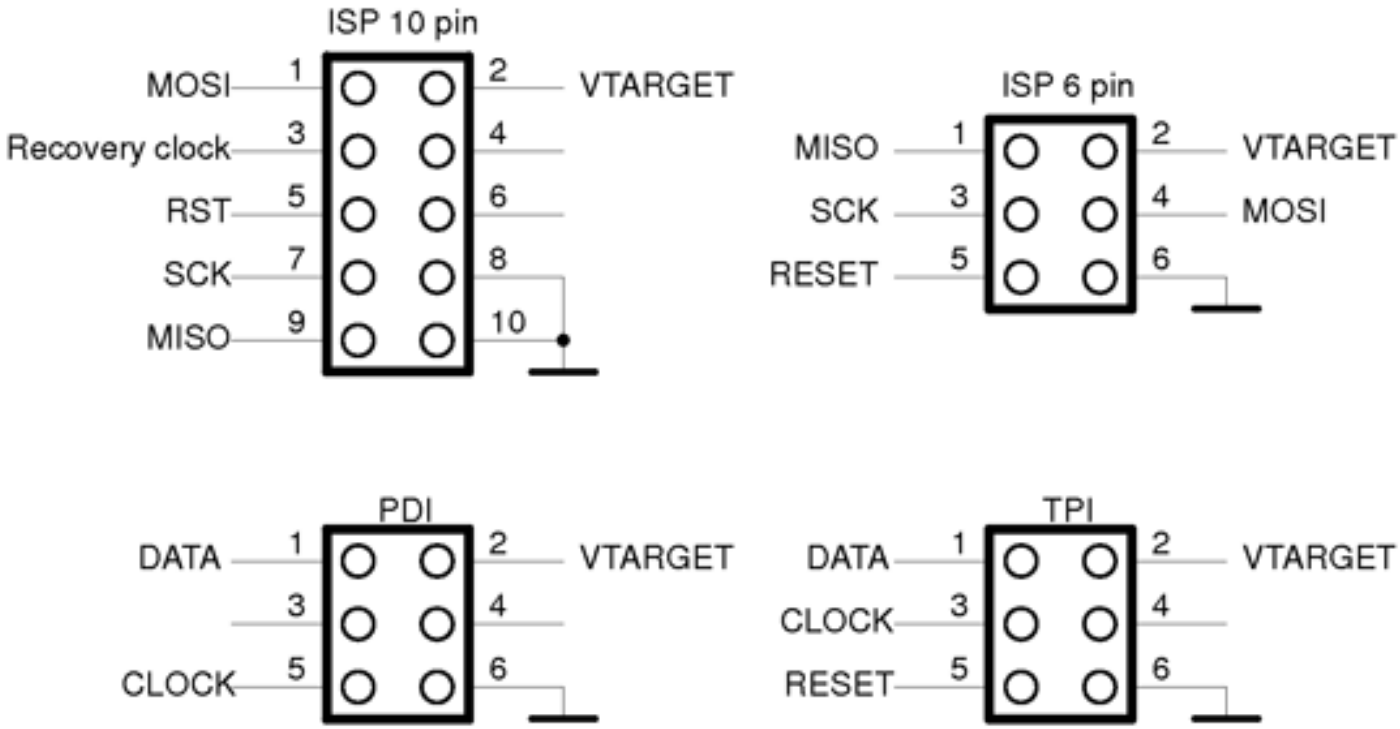


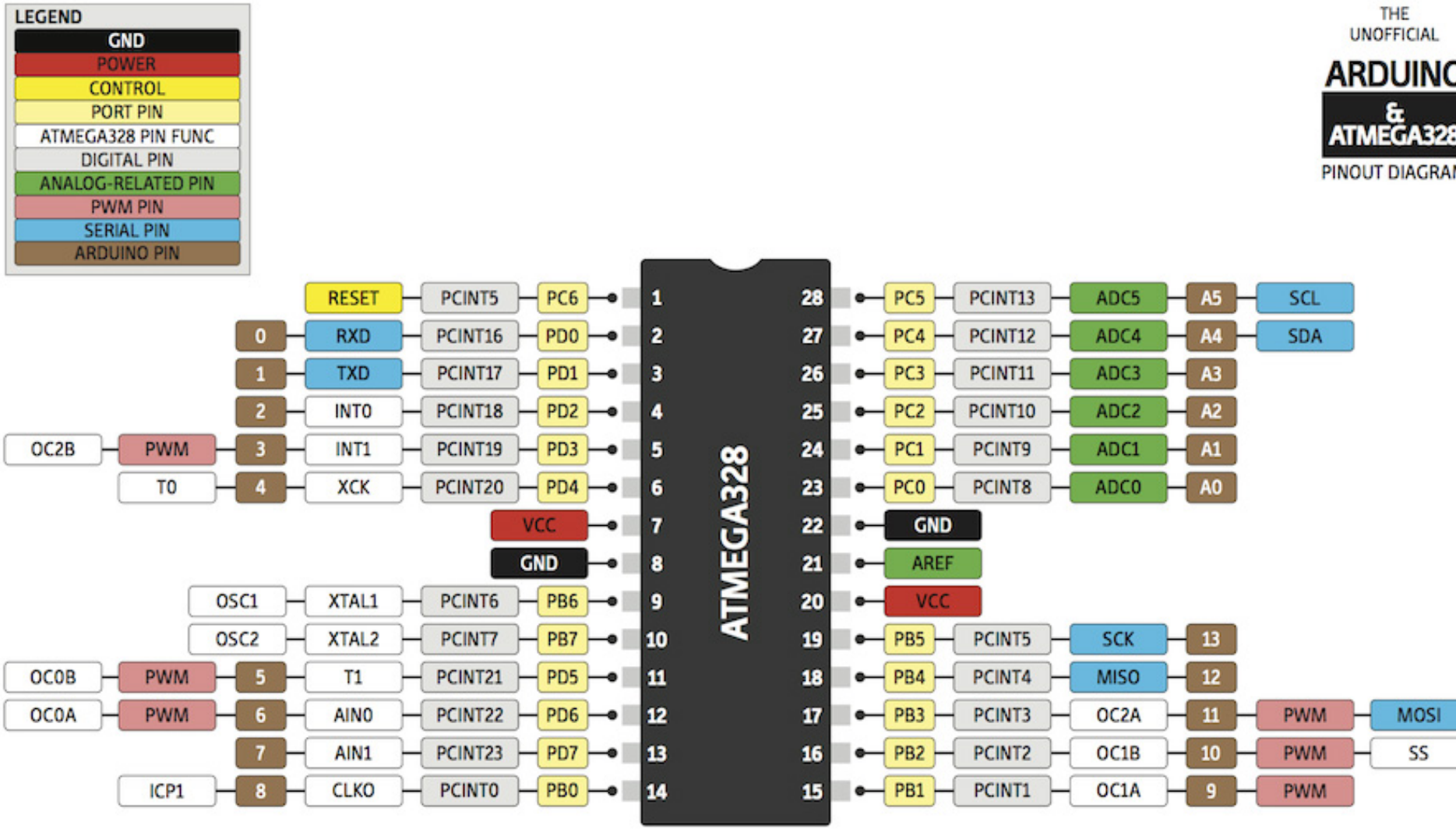
- CPU
- Horloge
- bus de données (8 bits)
- Mémoires
- I/O Ports
- Interruptions
- Circuit de reset
- Port de communication (SPI I2C USART ...)

Architecture d'un µcontrôleur



- CPU
- Horloge
- bus de données (8 bits)
- Mémoires
- I/O Ports
- Interruptions
- Circuit de reset
- Port de communication (SPI I2C USART ...)
- Port de programmation


ISP 10 pin
MOSI
1
2
VTARGET
Recovery clock
3
4
RST
5
6
SCK
7
8
MISO
9
10
ISP 6 pin
MISO
1
2
VTARGET
SCK
3
4
MOSI
RESET
5
6
PDI
DATA
1
2
VTARGET
3
4
CLOCK
5
6
TPI
DATA
1
2
VTARGET
CLOCK
3
4
RESET
5
6


LEGEND
GND
POWER
CONTROL
PORT PIN
ATMEGA328 PIN FUNC
DIGITAL PIN
ANALOG-RELATED PIN
PWM PIN
SERIAL PIN
ARDUINO PIN
THE UNOFFICIAL
ARDUINO
& ATMEGA328
PINOUT DIAGRAM
ATMEGA328
RESET PCINT5 PC6 1
0 RXD PCINT16 PD0 2
1 TXD PCINT17 PD1 3
2 INT0 PCINT18 PD2 4
OC2B PWM 3 INT1 PCINT19 PD3 5
T0 4 XCK PCINT20 PD4 6
VCC 7
GND 8
OSC1 XTAL1 PCINT6 PB6 9
OSC2 XTAL2 PCINT7 PB7 10
OC0B PWM 5 T1 PCINT21 PD5 11
OC0A PWM 6 AIN0 PCINT22 PD6 12
7 AIN1 PCINT23 PD7 13
ICP1 8 CLKO PCINT0 PB0 14
28 PC5 PCINT13 ADC5 A5 SCL
27 PC4 PCINT12 ADC4 A4 SDA
26 PC3 PCINT11 ADC3 A3
25 PC2 PCINT10 ADC2 A2
24 PC1 PCINT9 ADC1 A1
23 PC0 PCINT8 ADC0 A0
22 GND
21 AREF
20 VCC
19 PB5 PCINT5 SCK 13
18 PB4 PCINT4 MISO 12
17 PB3 PCINT3 OC2A 11 PWM MOSI
16 PB2 PCINT2 OC1B 10 PWM SS
15 PB1 PCINT1 OC1A 9 PWM