Microprocesseurs-Microcontrôleurs

ENSEA

ETIS / ENSEA
Mel : Bertrand.Granado@ensea.fr

Automne 2010

Sommaire

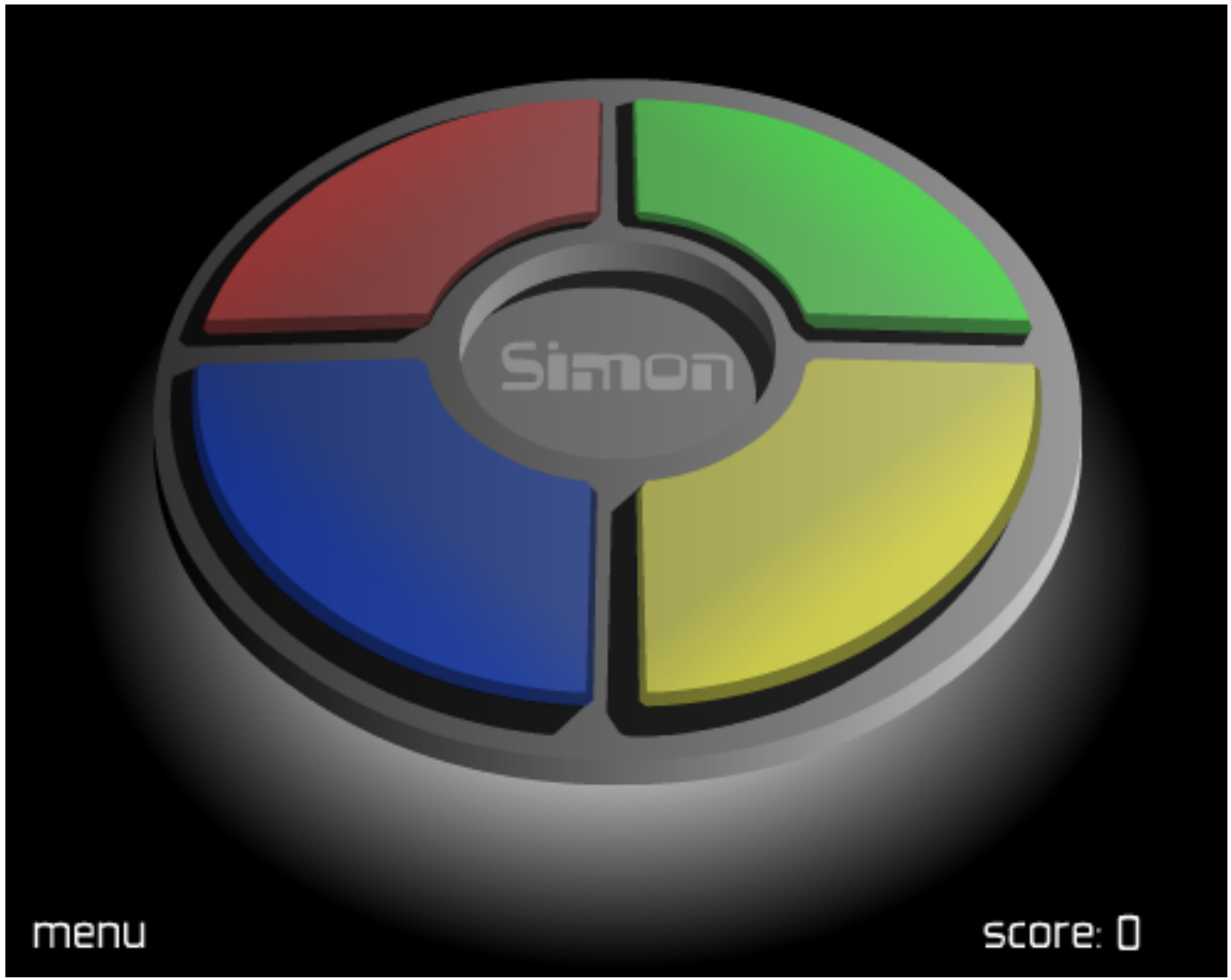
Les Chefs

- Samuel Garcia
- Mahmoud Karabernou
- Lounis Kessal
- Laurent Monchal
- Nicolas Simond
- Antoine Tauvel

La Sauce

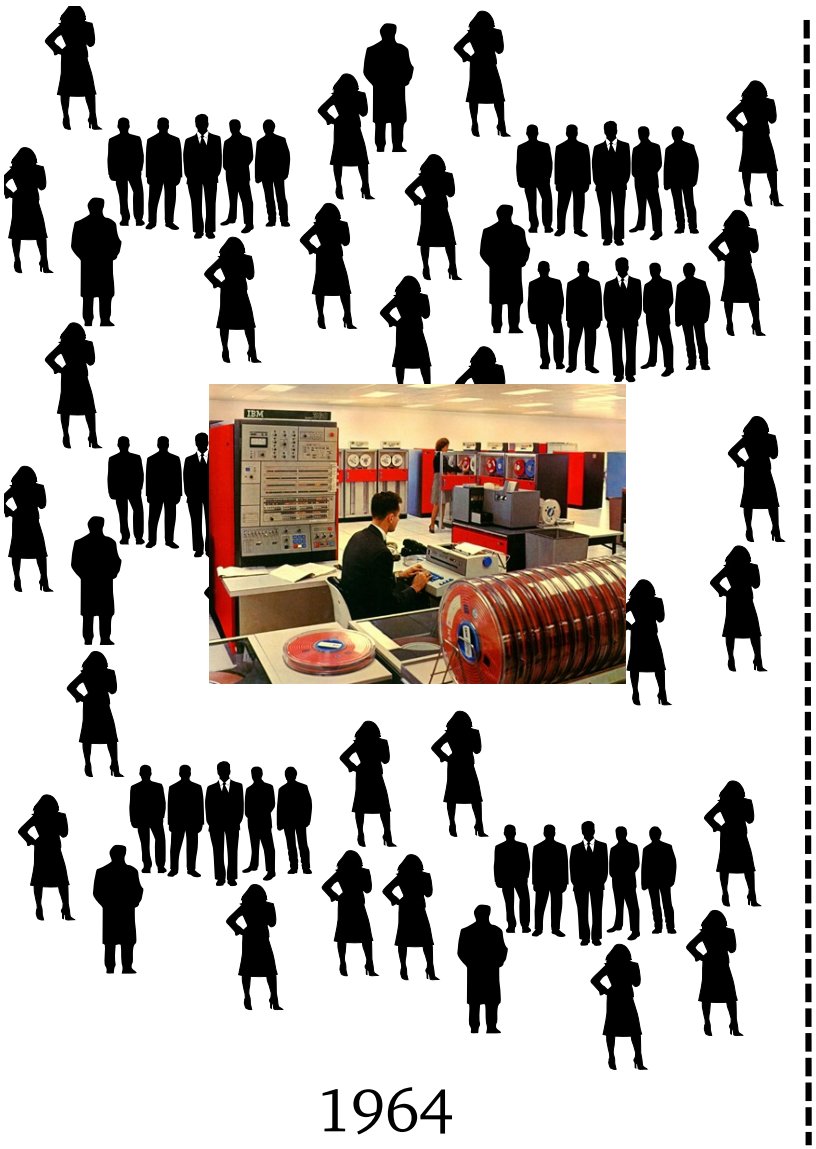
- 10 h de Cours
- 4 h de TD
- 16 h de Mini-Projet

Le Dessert

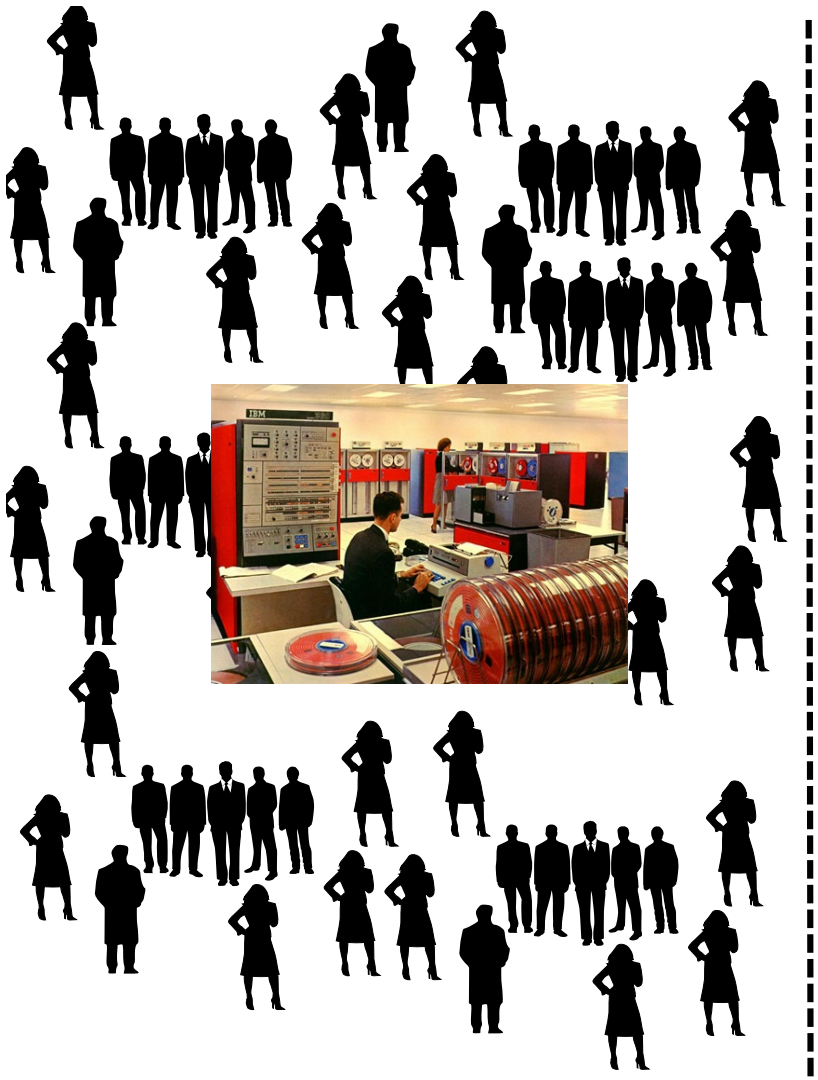


Sommaire

e-Life



e-Life



e-Life

2010

e-Life

2010

e-Life

e-Life

1964

2010

e-Life

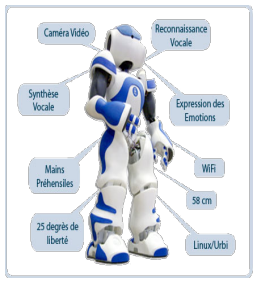
e-Life

1964

2010

e-Life

1964



2010

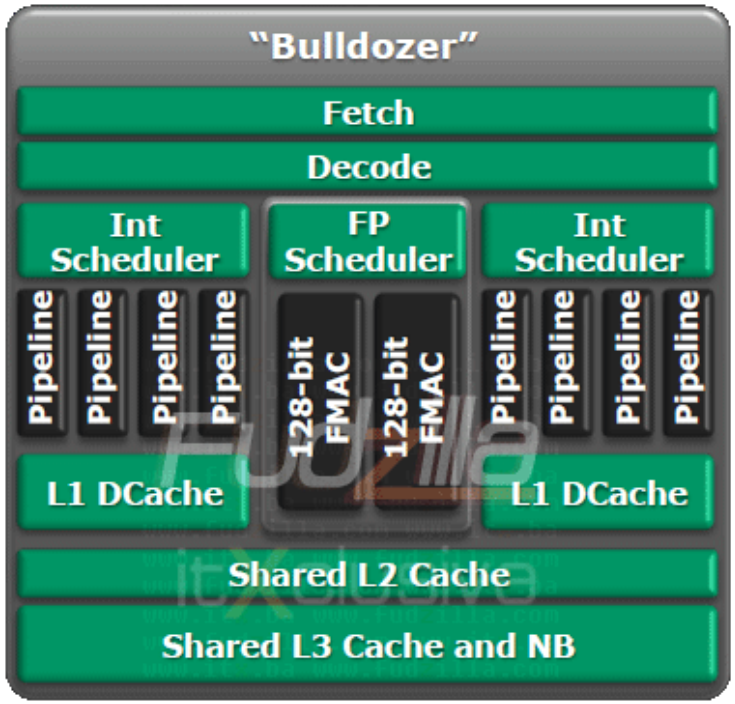
e-Life

des dizaines d'équivalents
ordinateur pour une personne

2010

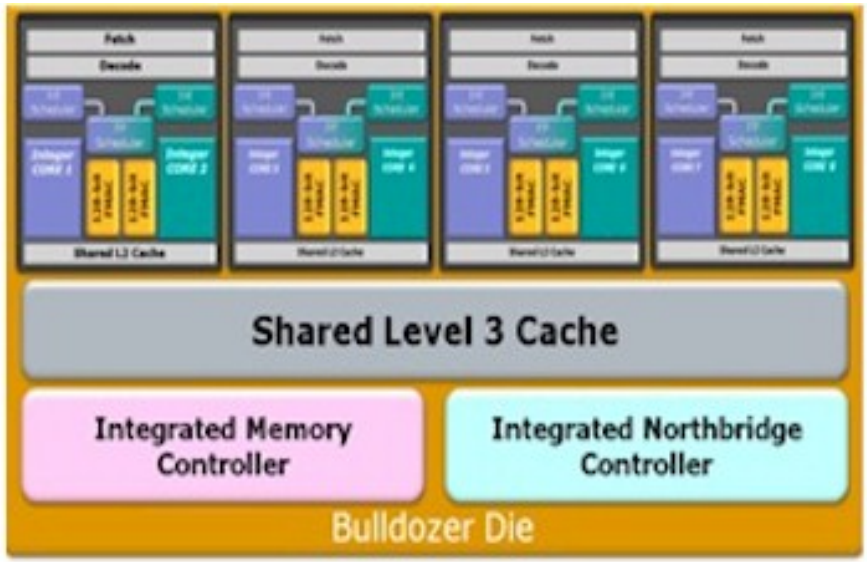
Support de la e-Life

- MicroProcesseurs



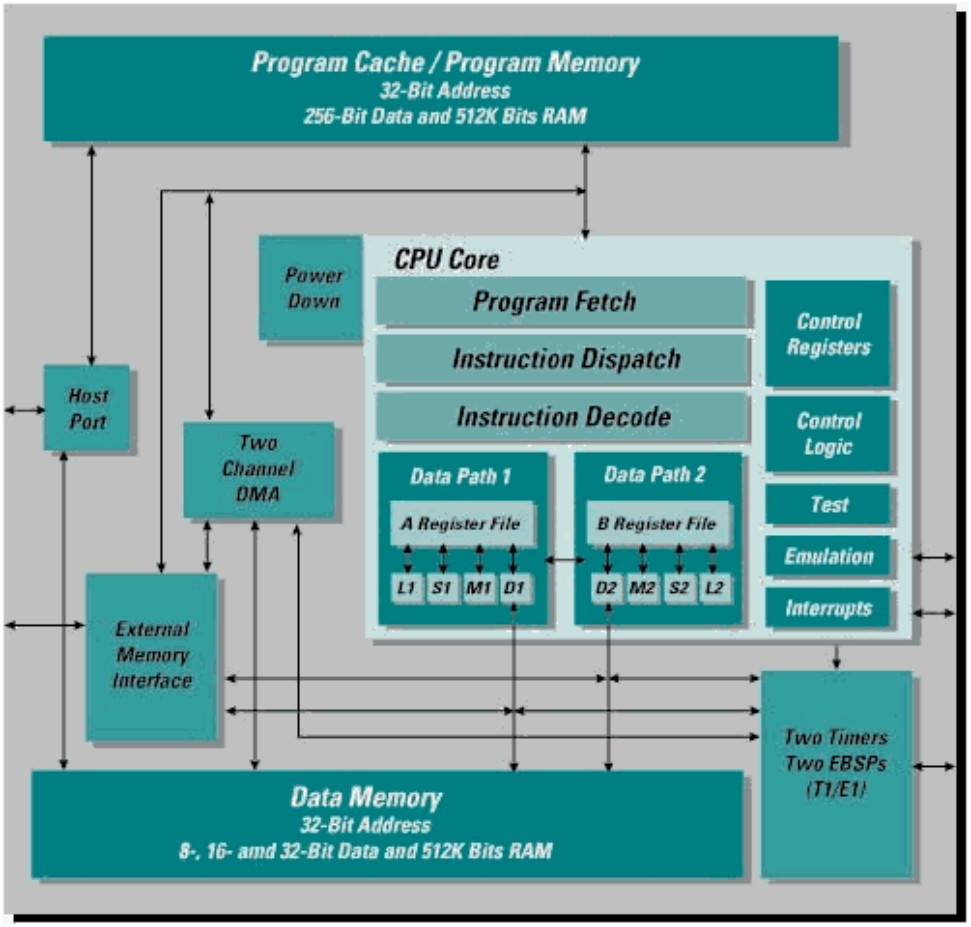
Support de la e-Life

- MicroProcesseurs



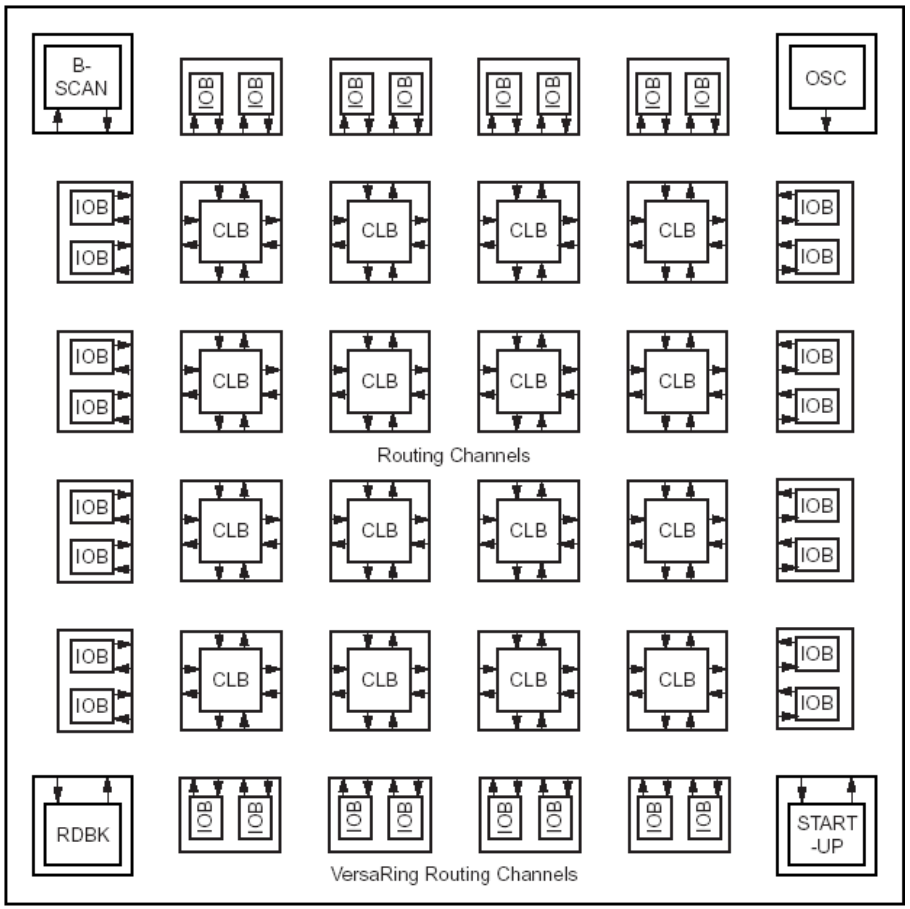
Support de la e-Life

- MicroProcesseurs
- Processeur Spécialisés : DSP, GPU



Support de la e-Life

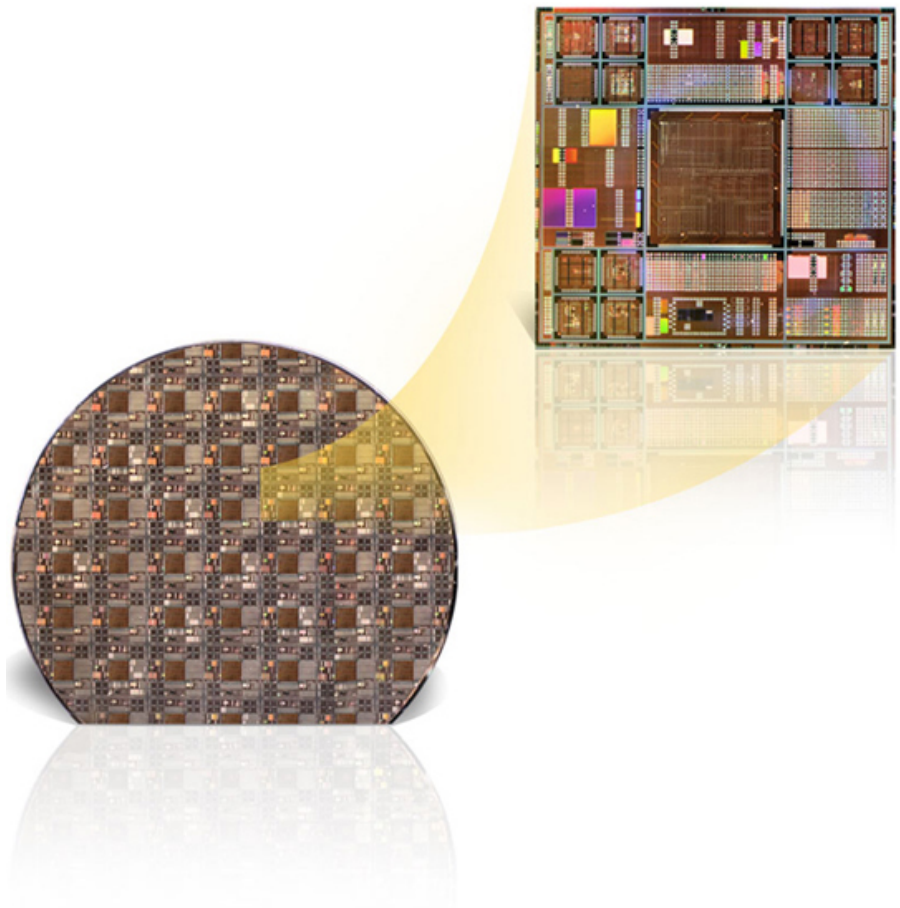
- MicroProcesseurs
- Processeur Spécialisés : DSP, GPU
- Circuits Reconfigurables : FPGA, PLD



Basic FPGA Block Diagram

Support de la e-Life

- MicroProcesseurs
- Processeur Spécialisés : DSP, GPU
- Circuits Reconfigurables : FPGA, PLD
- Circuits Spécifiques : ASIC, ASIP



Support de la e-Life

- MicroProcesseurs
- Processeur Spécialisés : DSP, GPU
- Circuits Reconfigurables : FPGA, PLD
- Circuits Spécifiques : ASIC, ASIP
- MicroContrôleurs

Les MicroContrôleurs

- Interaction forte avec l'environnement

Les MicroContrôleurs

- Interaction forte avec l'environnement
- Concevoir un composant qui associe 3 aspects

Les MicroContrôleurs

- Interaction forte avec l'environnement
- Concevoir un composant qui associe 3 aspects
  - Acquisition

Les MicroContrôleurs

- Interaction forte avec l'environnement
- Concevoir un composant qui associe 3 aspects
  - Acquisition
  - Traitement

Les MicroContrôleurs

- Interaction forte avec l'environnement
- Concevoir un composant qui associe 3 aspects
  - Acquisition
  - Traitement
  - Commande

Les MicroContrôleurs

- Interaction forte avec l'environnement
- Concevoir un composant qui associe 3 aspects
  - Acquisition
  - Traitement
  - Commande
- Premier SoC (System on Chip ou système sur puce)

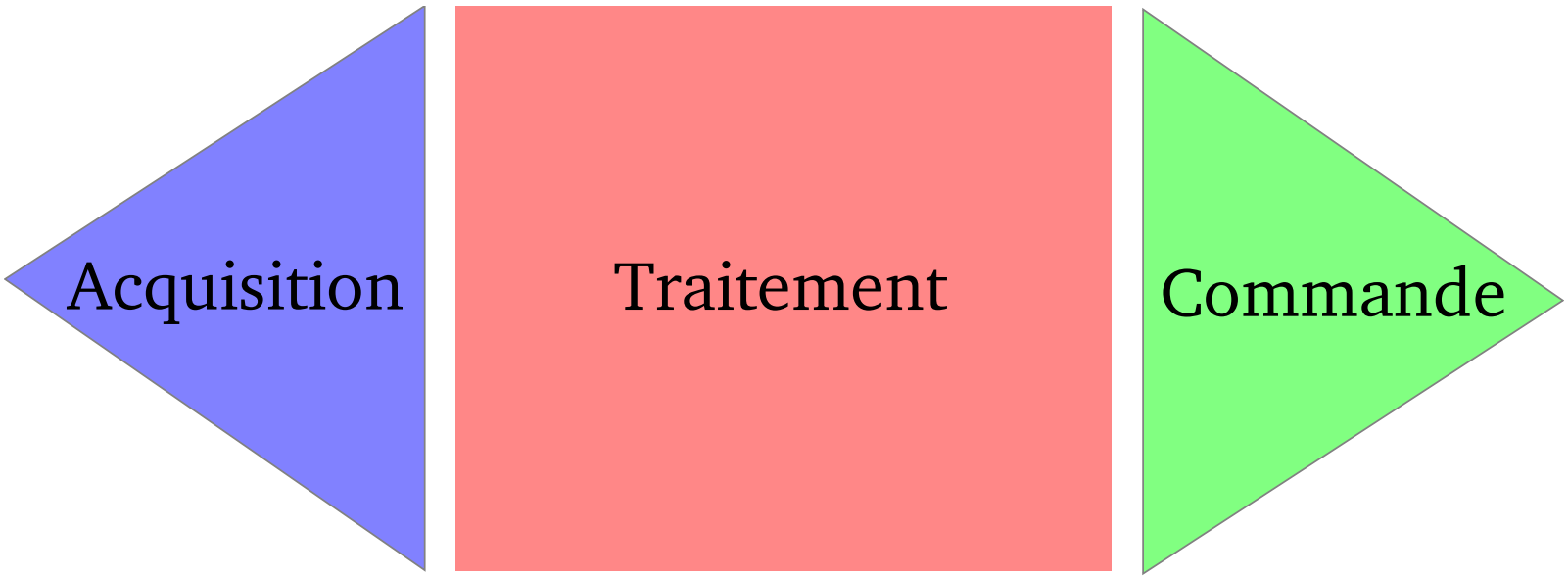
Les MicroContrôleurs

- Interaction forte avec l'environnement
- Concevoir un composant qui associe 3 aspects
  - Acquisition
  - Traitement
  - Commande
- Premier SoC (System on Chip ou système sur puce)
- Dans la pratique : faible capacité de traitement

Les MicroContrôleurs

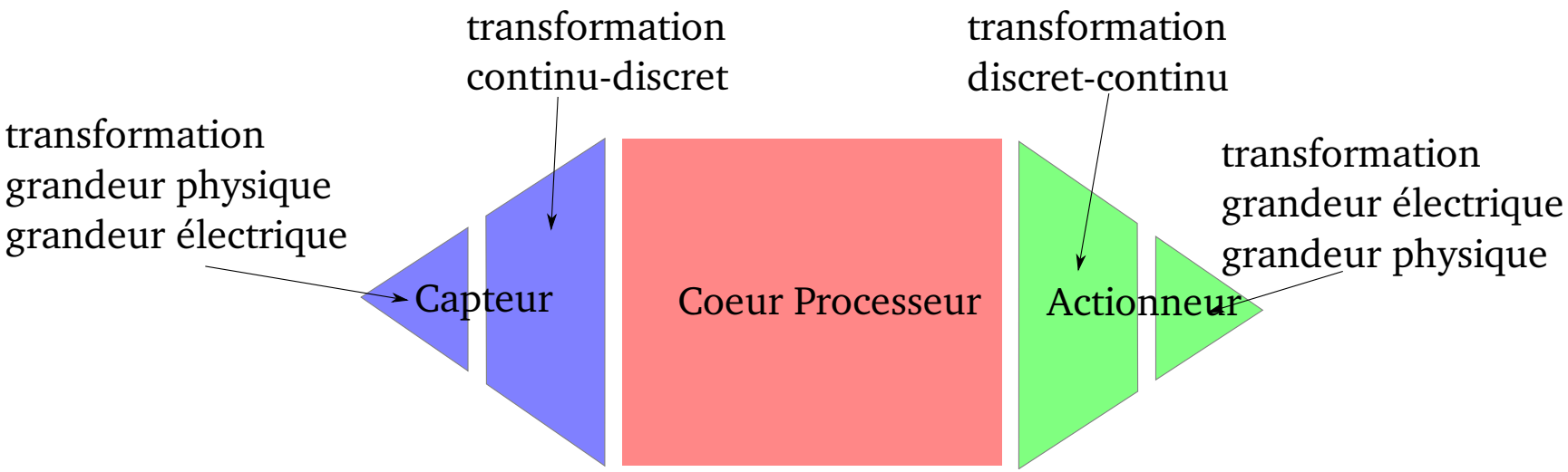
- Interaction forte avec l'environnement
- Concevoir un composant qui associe 3 aspects
  - Acquisition
  - Traitement
  - Commande
- Premier SoC (System on Chip ou système sur puce)
- Dans la pratique : faible capacité de traitement
- Dédié plutôt contrôle

Les MicroContrôleurs : Principes

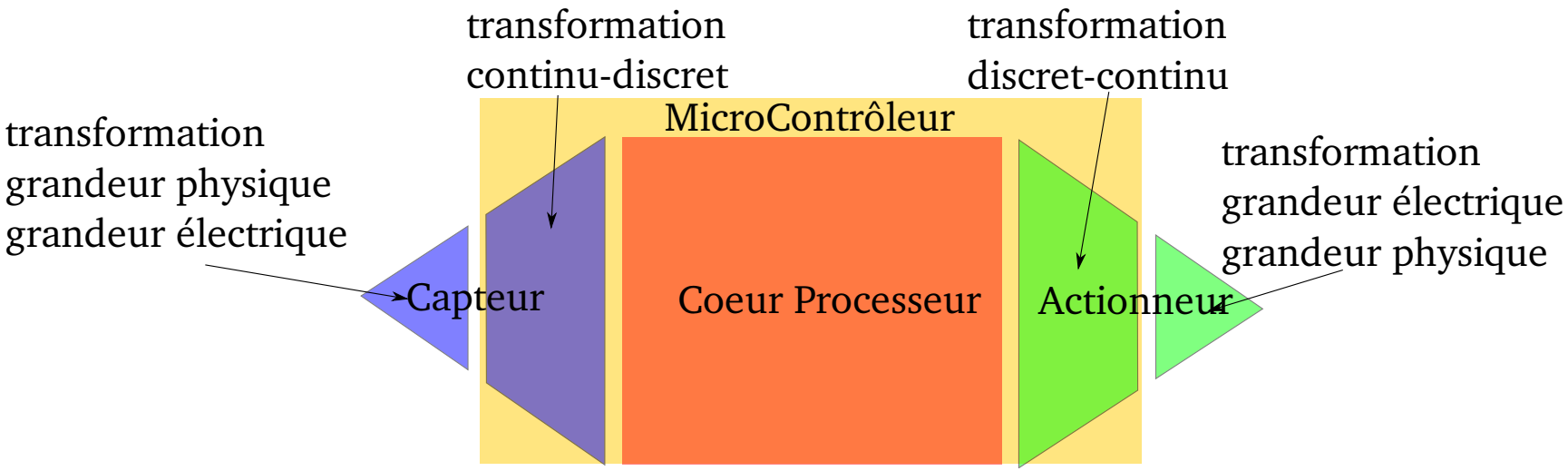


Les MicroContrôleurs : Principes

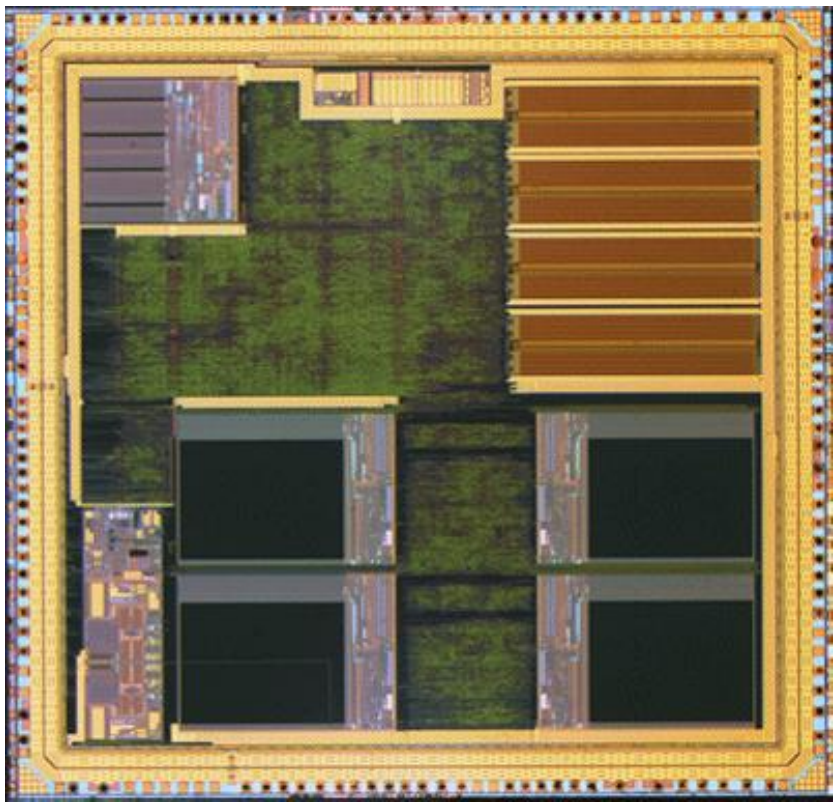
Les MicroContrôleurs : Principes



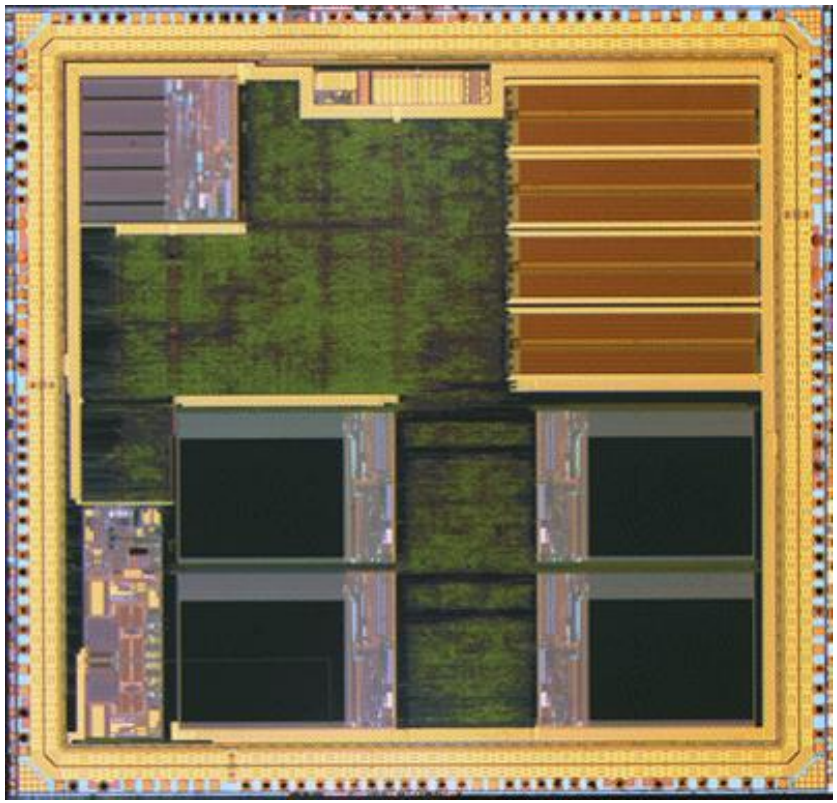
Les MicroContrôleurs : Principes

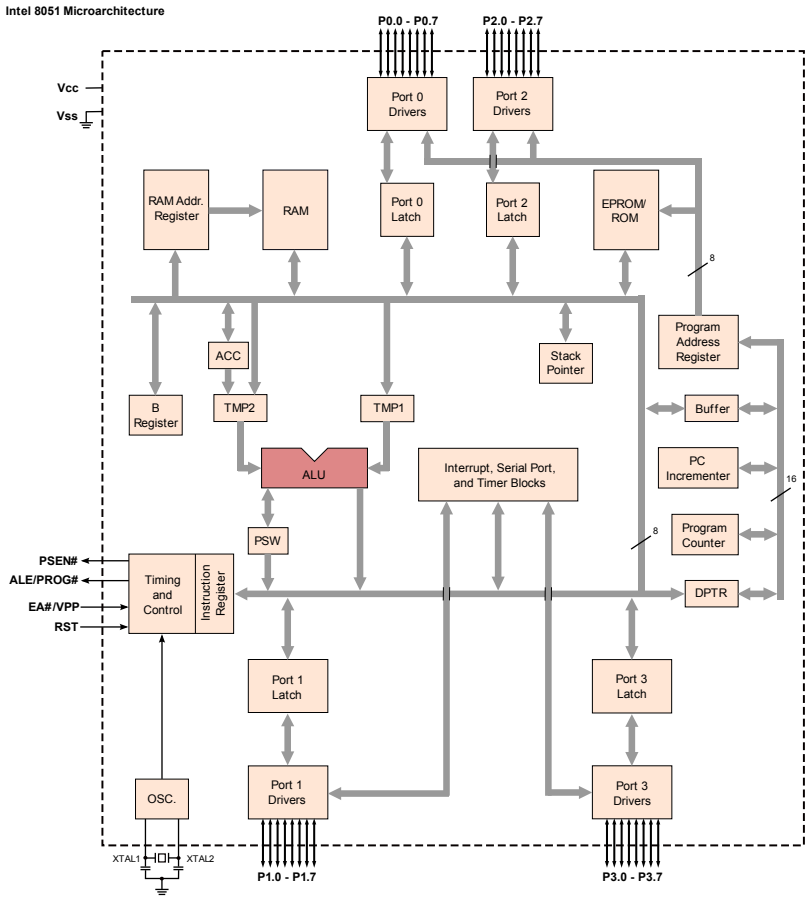


Les MicroContrôleurs : 8051

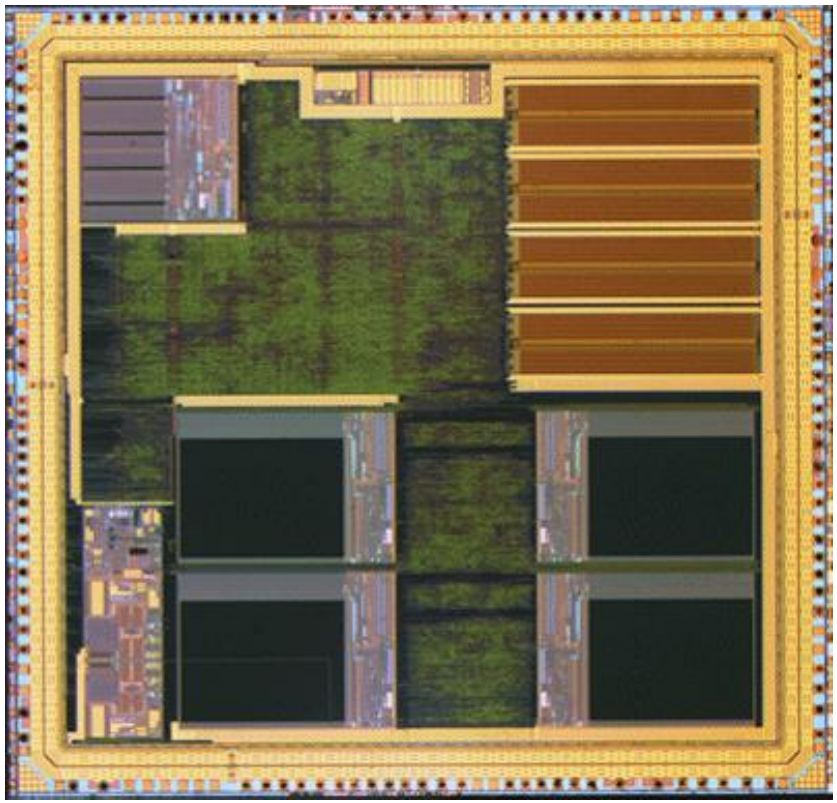


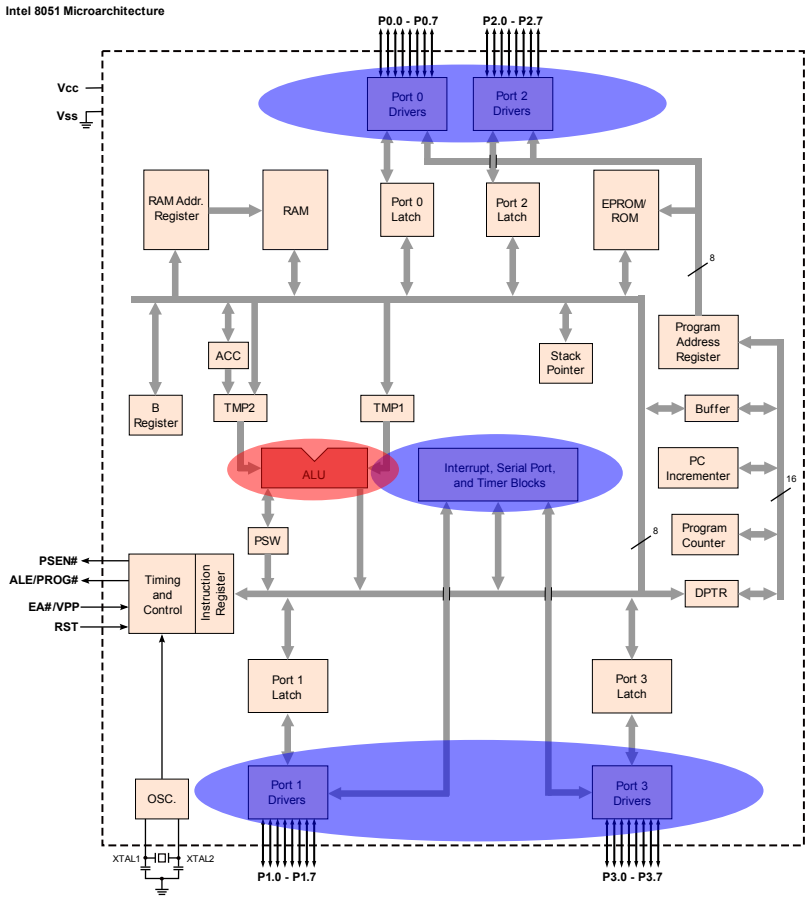
Les MicroContrôleurs : 8051



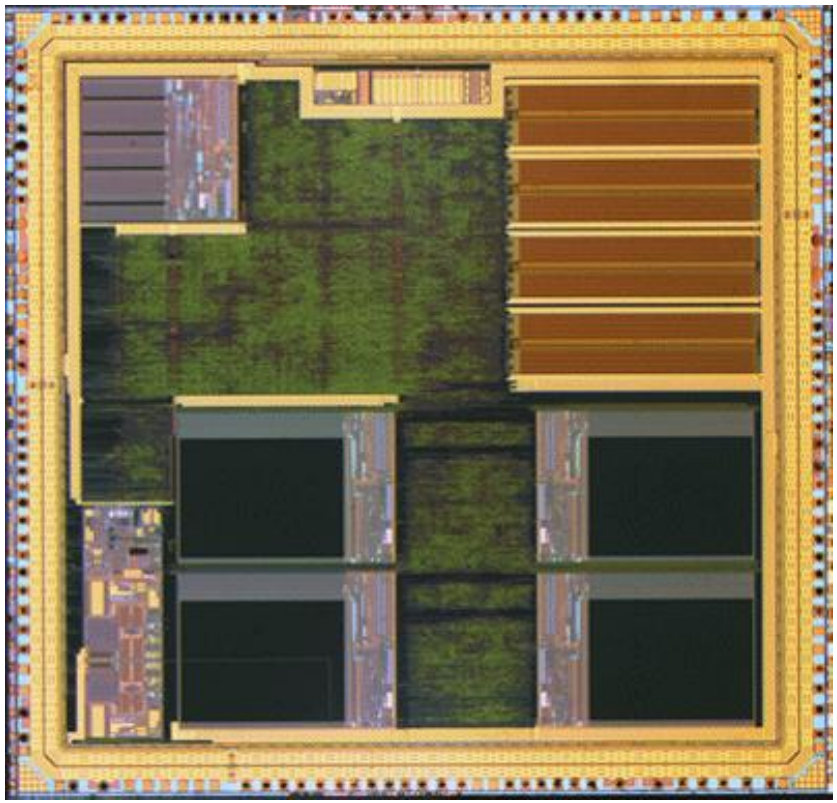


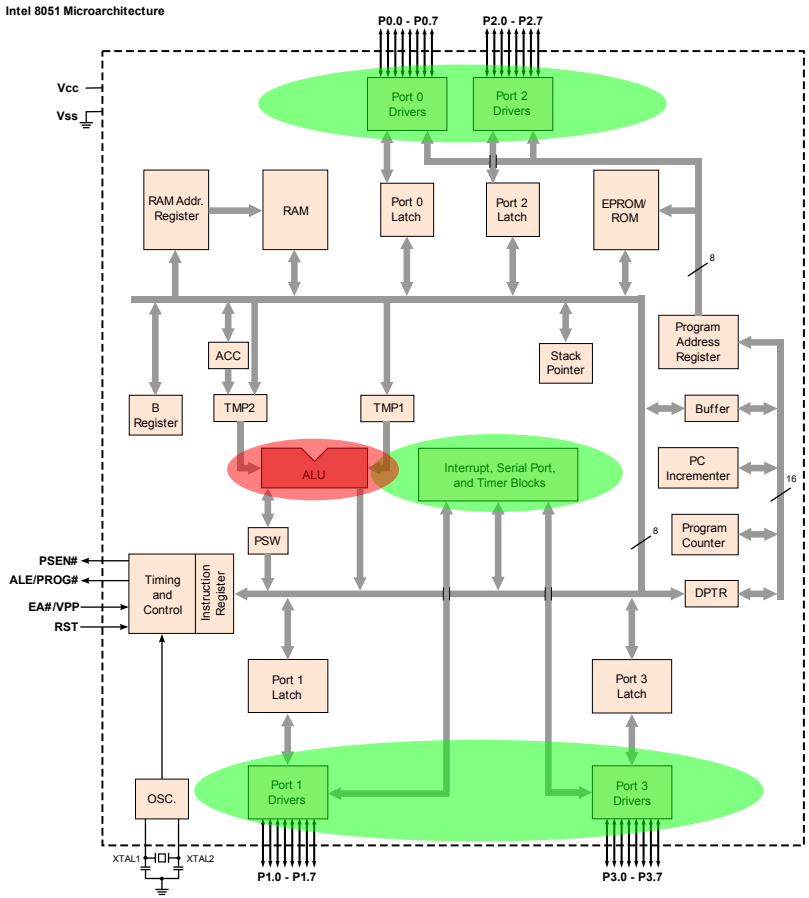
Les MicroContrôleurs : 8051





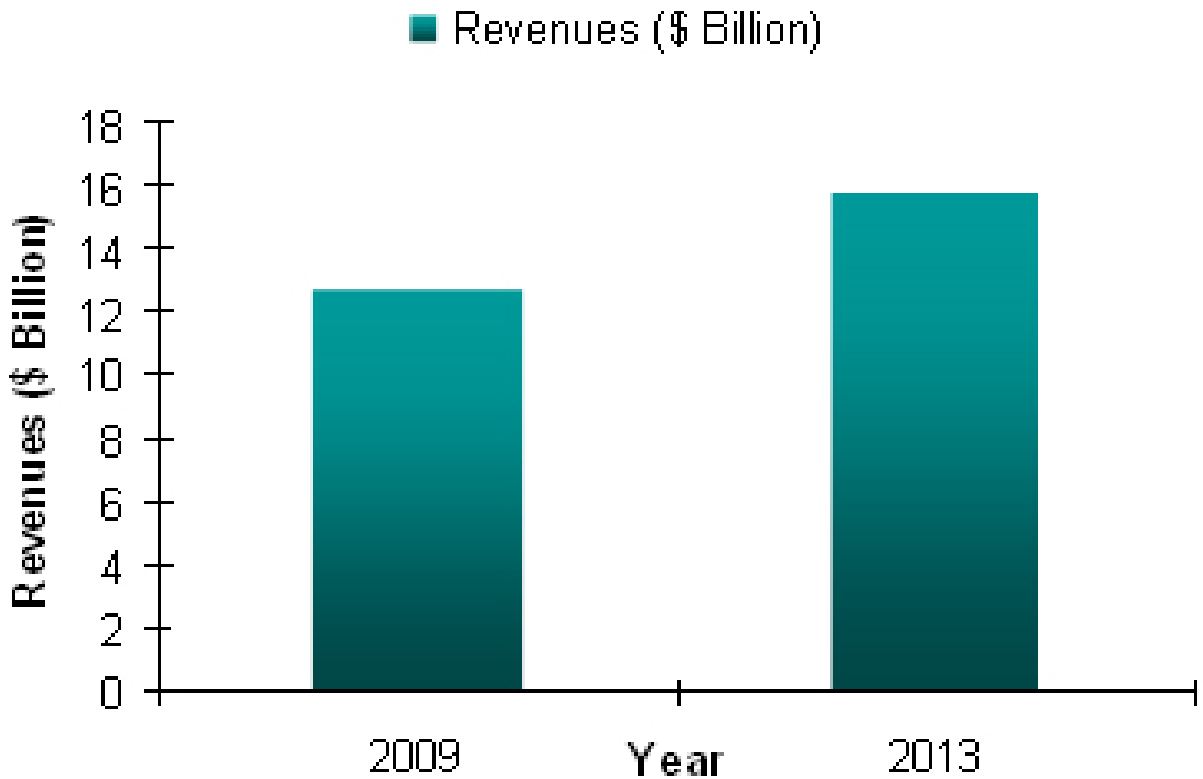
Les MicroContrôleurs : 8051





Le marché des microcontrôleurs

Chart 1.1
Microcontrollers Market: Revenue Forecasts (World), 2009 and 2013



Note: All figures are rounded; the base year is 2009. Source: Frost & Sullivan

FROST & SULLIVAN

Sommaire

Le STM32 : présentation

Le STM32 : présentation

- MicroContrôleur ST basé sur le processeur ARM Cortex M3

Le STM32 : présentation

- MicroContrôleur ST basé sur le processeur ARM Cortex M3
- Processeur RISC

Le STM32 : présentation

- MicroContrôleur ST basé sur le processeur ARM Cortex M3
- Processeur RISC
- Architecture Harvard : bus instructions et données séparés

Le STM32 : présentation

- MicroContrôleur ST basé sur le processeur ARM Cortex M3
- Processeur RISC
- Architecture Harvard : bus instructions et données séparés
- Mode de gestion de la consommation

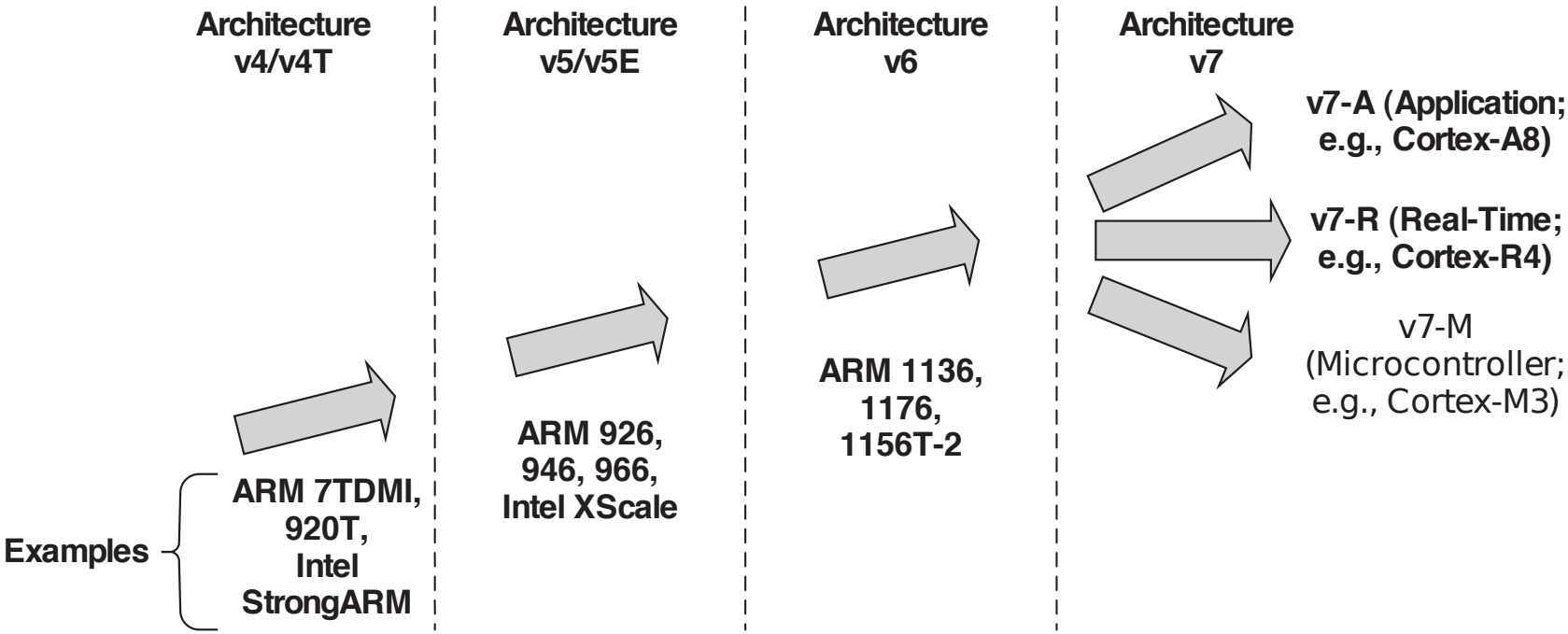
Le STM32 : présentation

- MicroContrôleur ST basé sur le processeur ARM Cortex M3
- Processeur RISC
- Architecture Harvard : bus instructions et données séparés
- Mode de gestion de la consommation
- Supporte uniquement le jeu d'instruction Thumb2

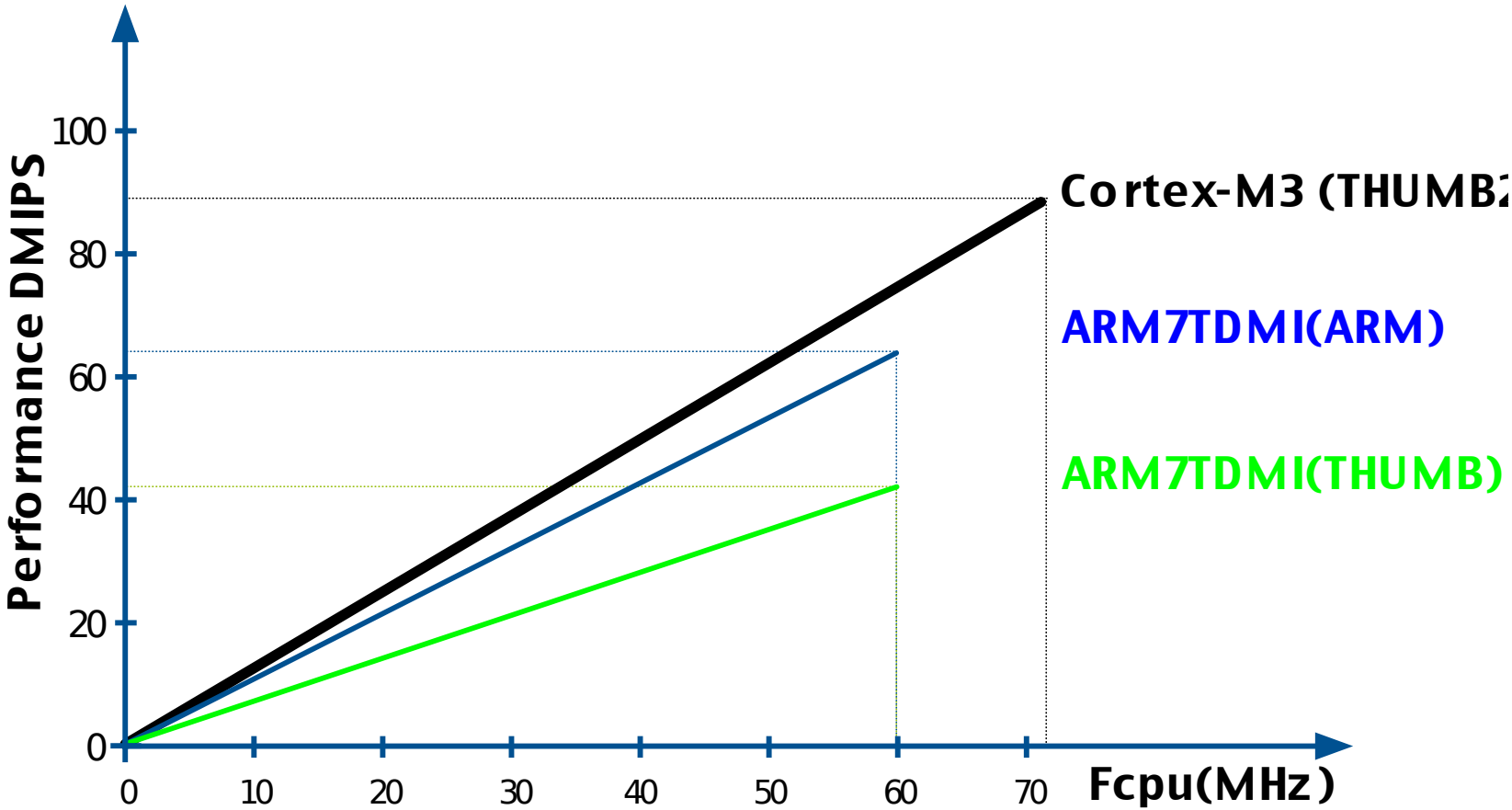
Le STM32 : présentation

- MicroContrôleur ST basé sur le processeur ARM Cortex M3
- Processeur RISC
- Architecture Harvard : bus instructions et données séparés
- Mode de gestion de la consommation
- Supporte uniquement le jeu d'instruction Thumb2
- Cours Mahmoud Karabernou - Première année (MOODLE)

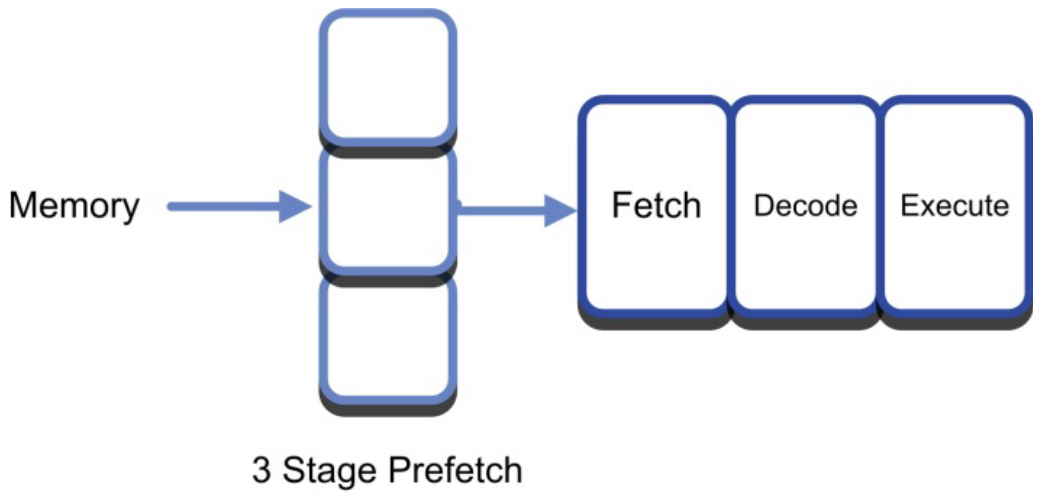
Le STM32 : evolution architecture ARM



Le STM32 : evolution architecture ARM

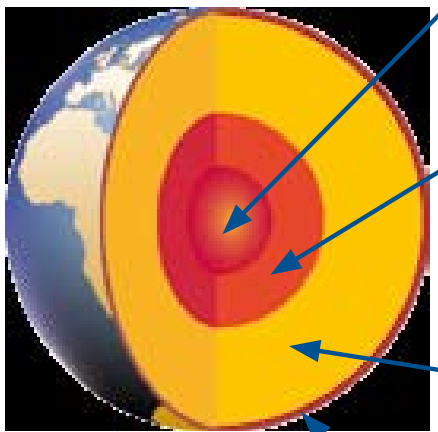


STM32 : Cortex M3



Le STM32 : environnement

Le monde du STM32



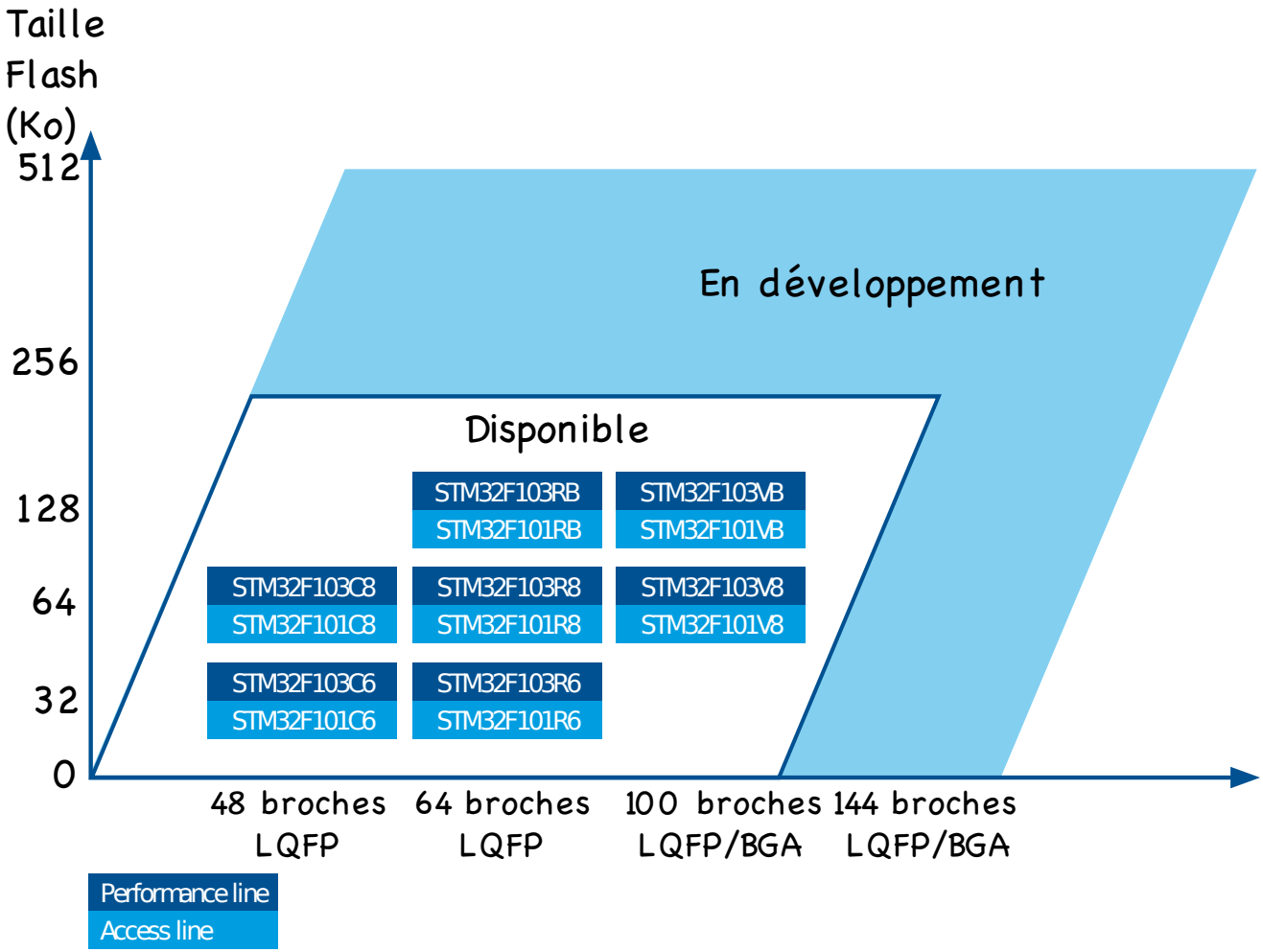
Coeur Cortex M3
- Communauté ARM
- Coeur Standard

Périphériques
- Haute Performances et Faible consommatic
- Très intégrés
- Innovants

Produits Catalogue
- Famille totalement compatible brochage, périphériques et logiciels.

Outils et Logiciels
- Très bon écosystème d'atelier de développement

Le STM32 : famille 2007



Le STM32 : famille 2007

Part number		Program memory type	Prog. (Bytes)	RAM (Bytes)	Timer functions		Serial interface	I/Os (High current)	Packages	Supply voltage
		Flash			12 or 16-bit (IC/OC/PWM)	Others				
STM32 (ARM Cortex-M3) - 32-bit microcontrollers										
48 pins	STM32F101C6	•	32 K	6 K	2x16-bit (8/8/8)	2xWDG, RTC, 24-bit down counter	1xSPI/1xI²C/2xUSART*	32(32)	LQFP48	2 to 3.6 V
	STM32F101C8	•	64 K	10 K	3x16-bit (12/12/12)		2xSPI/2xI²C/3xUSART*	32(32)	LQFP48	2 to 3.6 V
64 pins	STM32F101R6	•	32 K	6 K	2x16-bit (8/8/8)		1xSPI/1xI²C/2xUSART*	49(49)	LQFP64	2 to 3.6 V
	STM32F101R8	•	64 K	10 K	3x16-bit (12/12/12)		2xSPI/2xI²C/3xUSART*	49(49)	LQFP64	2 to 3.6 V
	STM32F101RB	•	128 K	16 K	3x16-bit (12/12/12)		2xSPI/2xI²C/3xUSART*	49(49)	LQFP64	2 to 3.6 V
100 pins	STM32F101V8	•	64 K	10 K	3x16-bit (12/12/12)		2xSPI/2xI²C/3xUSART*	80(80)	LQFP100	2 to 3.6 V
	STM32F101VB	•	128 K	16 K	3x16-bit (12/12/12)		2xSPI/2xI²C/3xUSART*	80(80)	LQFP100	2 to 3.6 V
48 pins	STM32F103C6	•	32 K	10 K	3x16-bit (12/12/14)		1xSPI/1xI²C/2xUSART*/USB/CAN	32(32)	LQFP48	2 to 3.6 V
	STM32F103C8	•	64 K	20 K	4x16-bit (16/16/18)		2xSPI/2xI²C/3xUSART*/USB/CAN	32(32)	LQFP48	2 to 3.6 V
64 pins	STM32F103R6	•	32 K	10 K	3x16-bit (12/12/14)		1xSPI/1xI²C/2xUSART*/USB/CAN	49(49)	LQFP64	2 to 3.6 V
	STM32F103R8	•	64 K	20 K	4x16-bit (16/16/18)		2xSPI/2xI²C/3xUSART*/USB/CAN	49(49)	LQFP64	2 to 3.6 V
	STM32F103RB	•	128 K	20 K	4x16-bit (16/16/18)		2xSPI/2xI²C/3xUSART*/USB/CAN	49(49)	LQFP64	2 to 3.6 V
100 pins	STM32F103V8	•	64 K	20 K	4x16-bit (16/16/18)		2xSPI/2xI²C/3xUSART*/USB/CAN	80(80)	LQFP100/BGA100	2 to 3.6 V
	STM32F103VB	•	128 K	20 K	4x16-bit (16/16/18)		2xSPI/2xI²C/3xUSART*/USB/CAN	80(80)	LQFP100/BGA100	2 to 3.6 V

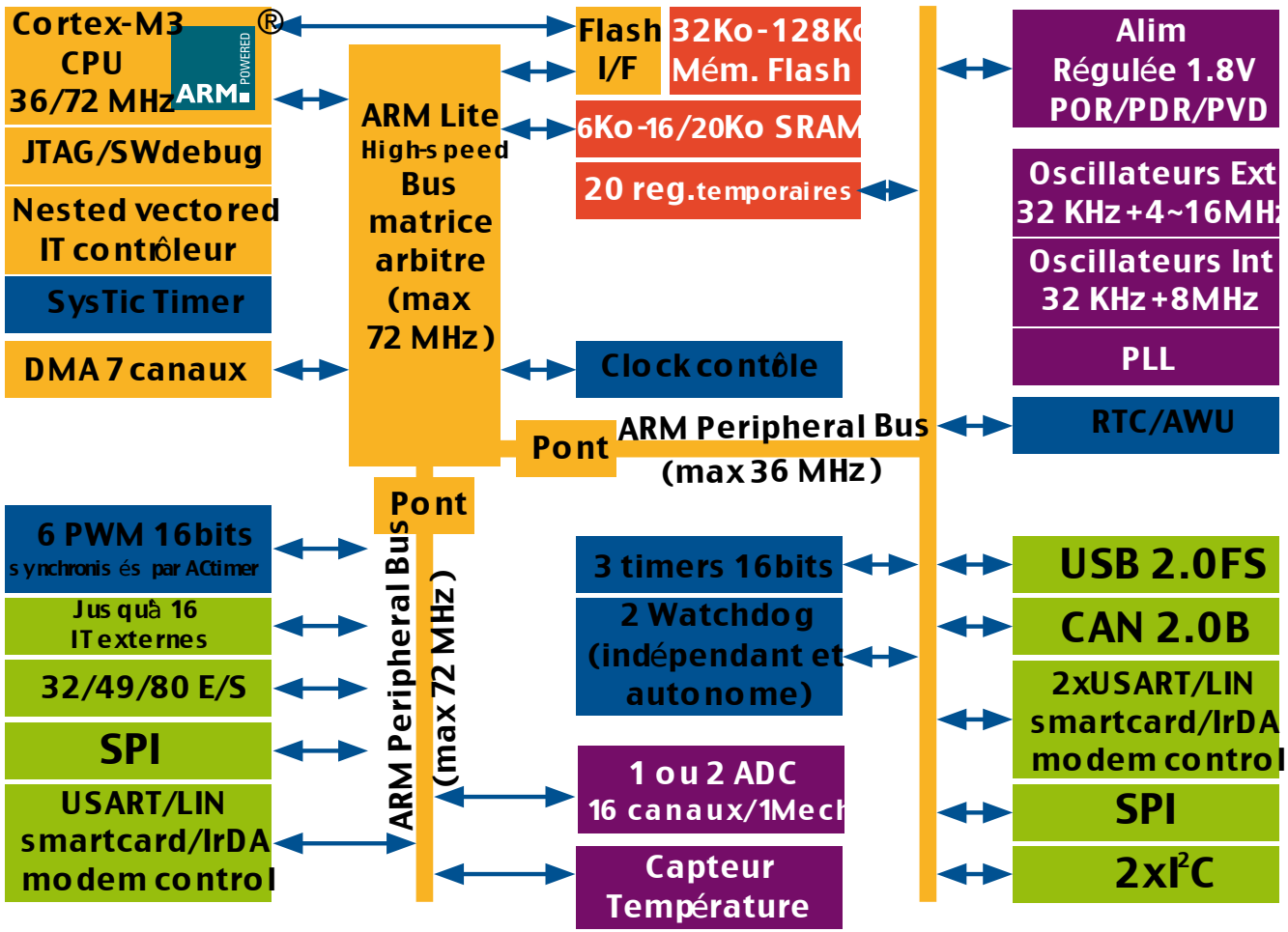
*(IrDA/ISO7816/LIN master/slave)

Le STM32 : famille 2010

Table 3. STM32F103xx family

Pinout	Low-density devices		Medium-density devices		High-density devices			XL-density devices	
	16 KB Flash	32 KB Flash(1)	64 KB Flash	128 KB Flash	256 KB Flash	384 KB Flash	512 KB Flash	768 KB Flash	1 MB Flash
	6 KB RAM	10 KB RAM	20 KB RAM	20 KB RAM	48 or 64 KB(2) RAM	64 KB RAM	64 KB RAM	96 KB RAM	96 KB RAM
144					**5 × USARTs 4 × 16-bit timers, 2 × basic timers 3 × SPIs, 2 × I²Ss, 2 × I2Cs USB, CAN, 2 × PWM timers 3 × ADCs, 2 × DACs, 1 × SDIO FSMC (100- and 144-pin packages(3))**			**5 × USARTs 10 × 16-bit timers, 2 × basic timers 3 × SPIs, 2 × I²Ss, 2 × I2Cs USB, CAN, 2 × PWM timers 3 × ADCs, 2 × DACs, 1 × SDIO, Cortex-M3 with MPU FSMC (100- and 144-pin packages(4)), dual bank Flash memory**	
100			**3 × USARTs 3 × 16-bit timers 2 × SPIs, 2 × I²Cs, USB, CAN, 1 × PWM timer 2 × ADCs**						
64	**2 × USARTs 2 × 16-bit timers 1 × SPI, 1 × I²C, USB, CAN, 1 × PWM timer 2 × ADCs**								
48									
36									

Le STM32 : synoptique



DMA : Direct Memory Access
RTC : Real Time Clock
AWU : Auto Wake Up capability with RTC alarm
POR : Power On Reset
PDR : Power Down Reset
PVD : Programmable Voltage Dete

Le STM32 : Outils de Développement

Fournisseur	Description
Hitex : www.hitex.com	Atelier, GNU C/C++, USB/JTAG
IAR : www.iar.com	Atelier, IAR C/C++, USB/JTAG
Keil : www.keil.com	Atelier, ARM C/C++, USB/JTAG
Raisonance : www.raisonance.com	Atelier, GNU C/C++, USB/JTAG
Rowley : www.rowley.co.uk	Atelier, GNU C/C++, JTAG

Le STM32 : Cartes de Développement

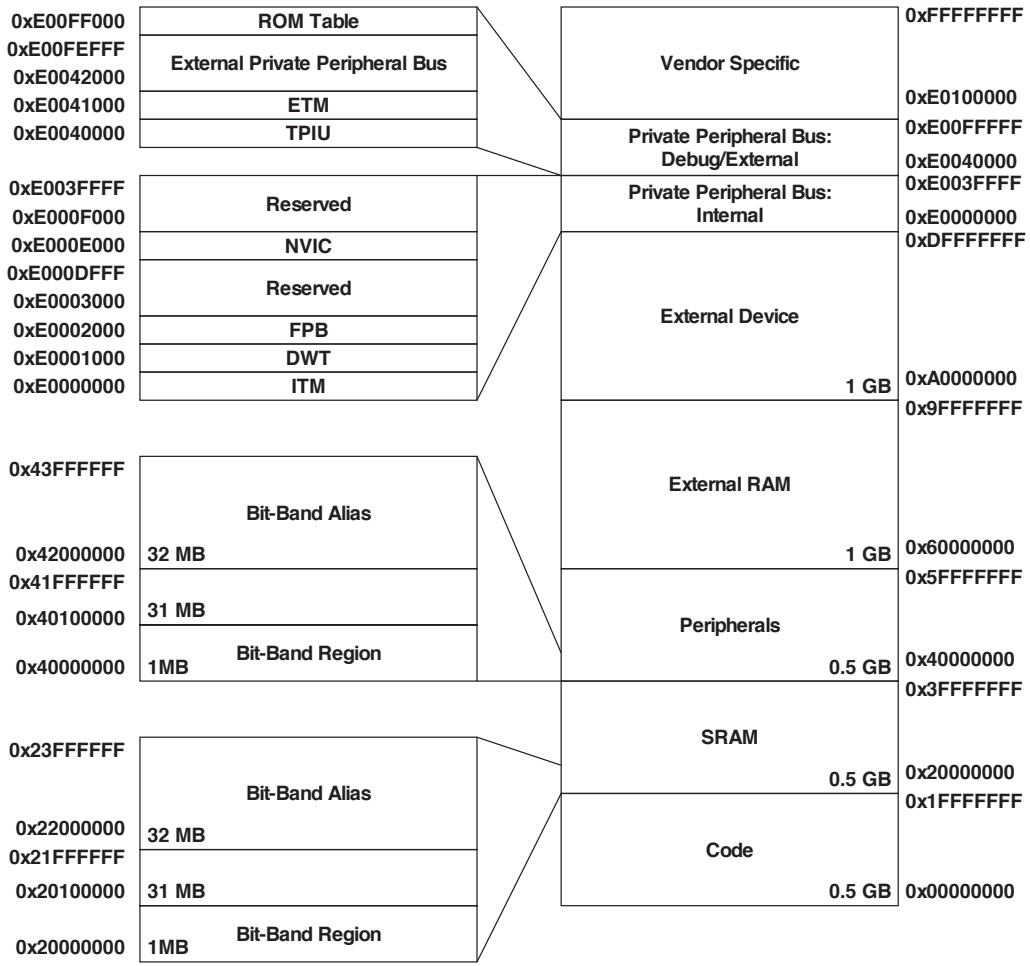
Fournisseur	Description
Hitex : www.hitex.com	STM3210B-SK/HIT
IAR : www.iar.com	STM3210B-SK/IAR
Keil : www.keil.com	STM3210B-SK/Keil
Raisonance : www.raisonance.com	STM3210B-PRIMER
Raisonance : www.raisonance.com	STM3210B-SK/RAIS
ST : www.st.com	STM3210B-MCKIT
...	...

Le STM32 : Cartes de Développement

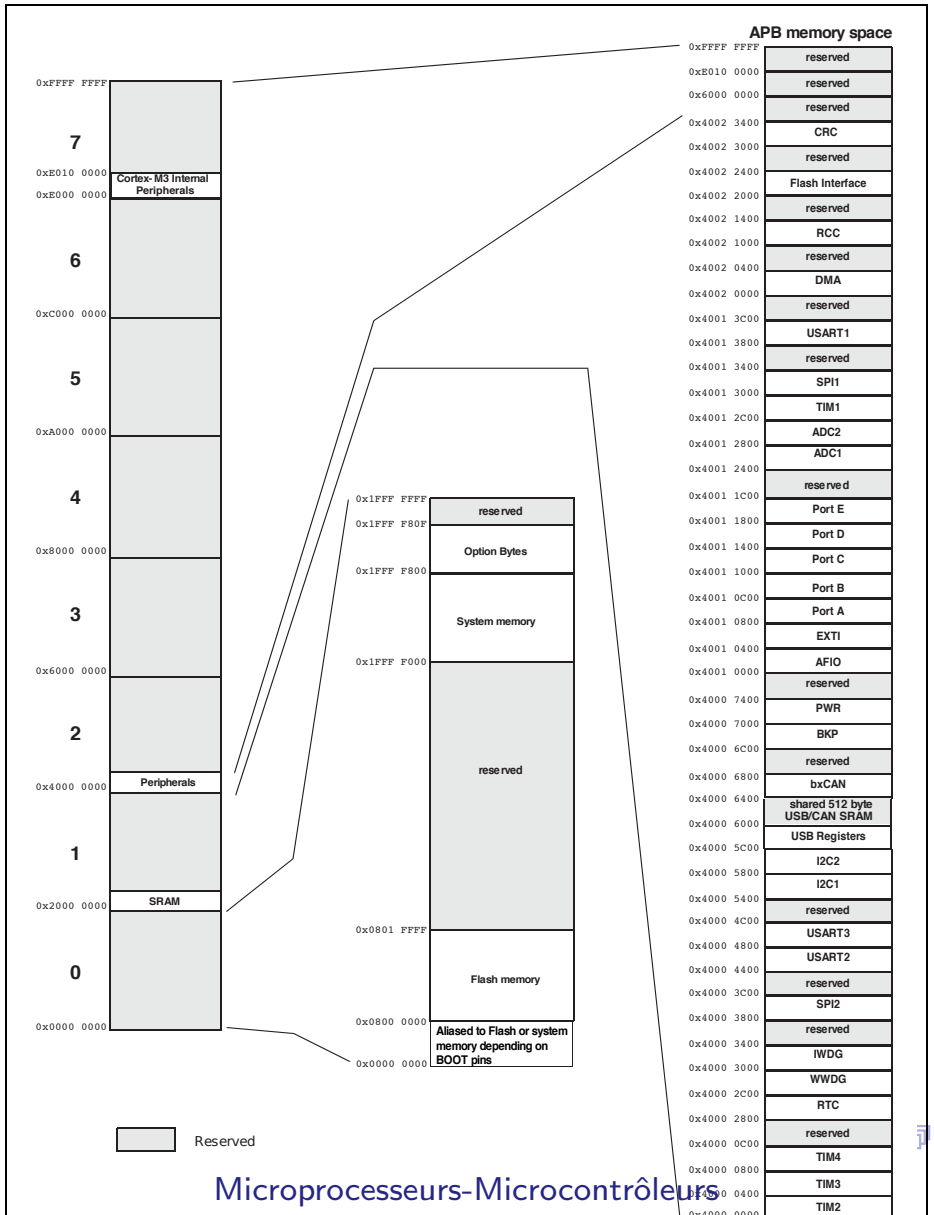
Le STM32 : Système d'Exploitation

Fournisseur	RTOS
Micrium : www.micrium.com	μC/OS-II
IAR : www.iar.com	PowerPac
Keil : www.keil.com	ARTX-ARM
www.FreeRTOS.org	FreeRTOS
Segger : www.segger.com	embOS
CMX Systems : www.cmx.com	CMX-RTX

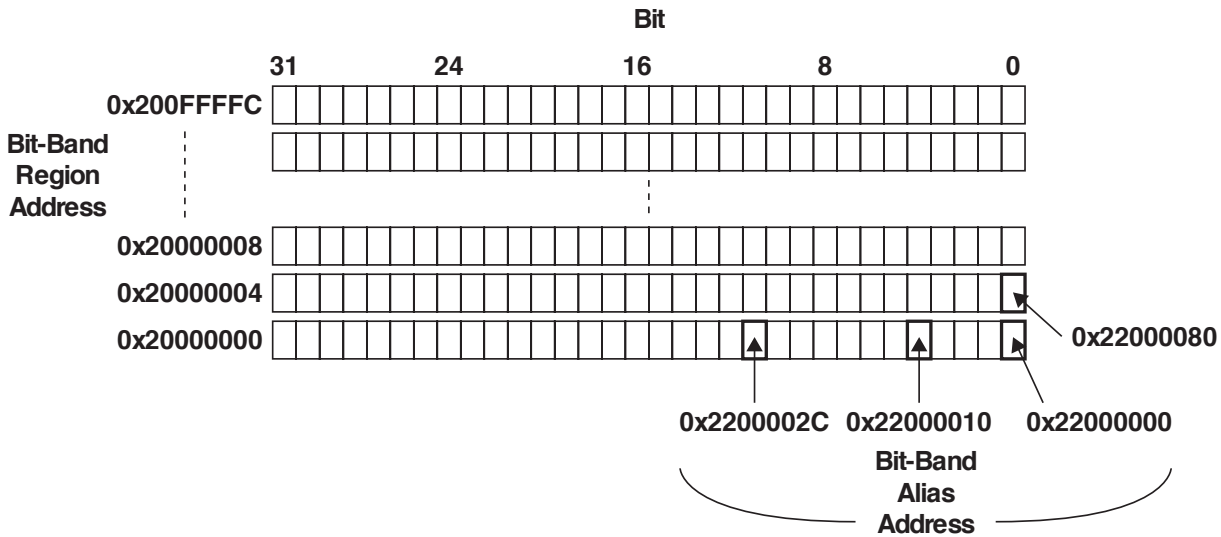
Le STM32 : mapping mémoire CortexM3



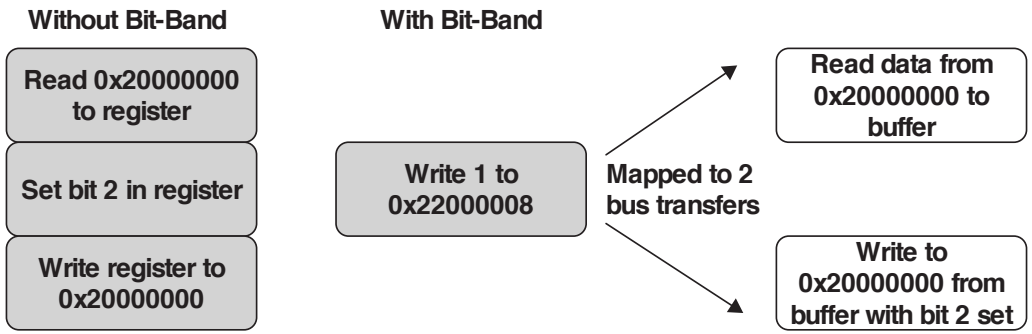
Le STM32 : mapping mémoire



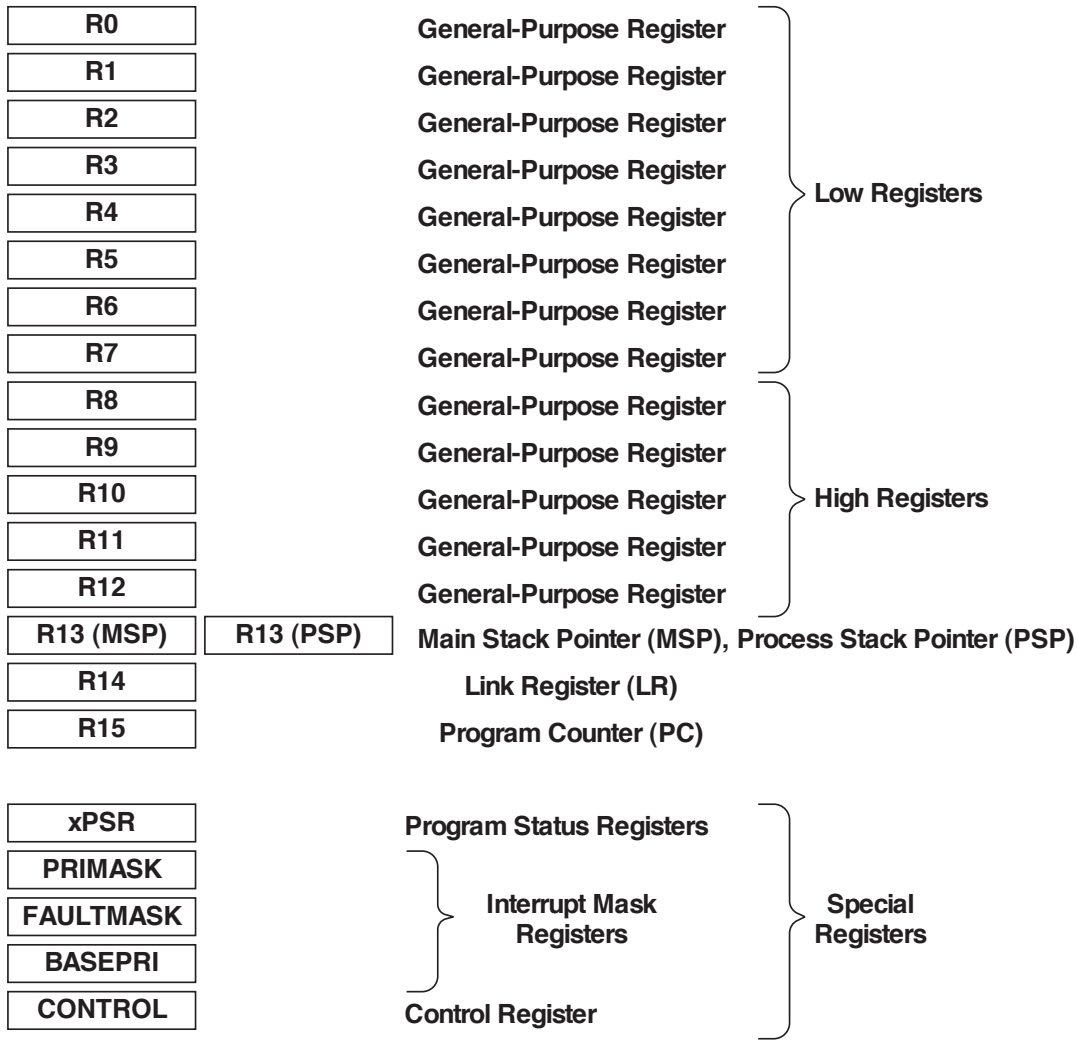
Le STM32 : Accès à des bits



Le STM32 : Accès à des bits



Le STM32 : les registres



Le STM32 : le registre xPSR

	31	30	29	28	27	26:25	24	23:20	19:16	15:10	9	8	7	6	5	4:0
APSR	N	Z	C	V	Q											
IPSR												Exception Number				
EPSR						ICI/IT	T			ICI/IT						

	31	30	29	28	27	26:25	24	23:20	19:16	15:10	9	8	7	6	5	4:0
xPSR	N	Z	C	V	Q	ICI/IT	T			ICI/IT		Exception Number				

Sommaire

Evénements externes

- Evénement externe est asynchrone

Evénements externes

- Evénement externe est asynchrone
- Impossibilité, en règle générale, de connaître le moment d'apparition de l'événement

Evénements externes

- Evénement externe est asynchrone
- Impossibilité, en règle générale, de connaître le moment d'apparition de l'événement
- Difficile d'inclure son traitement dans le flot synchrone d'un programme

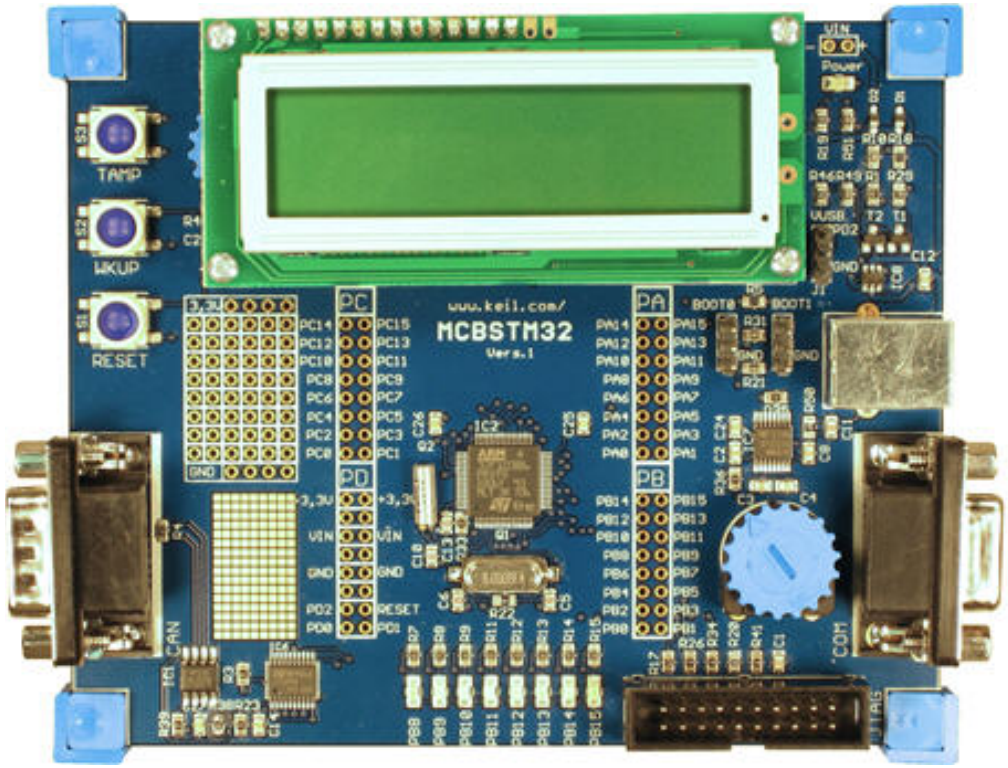
Evénements externes

- Evénement externe est asynchrone
- Impossibilité, en règle générale, de connaître le moment d'apparition de l'événement
- Difficile d'inclure son traitement dans le flot synchrone d'un programme
- Nécessité de méthodes pour capturer cet événement

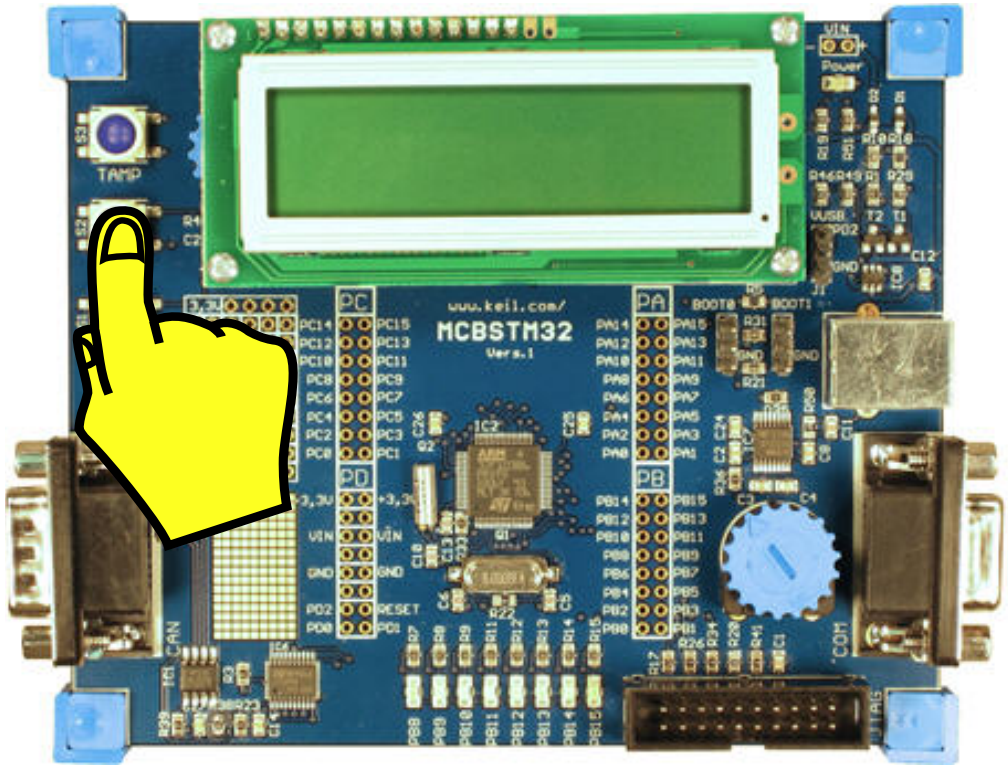
Evénements externes

- Evénement externe est asynchrone
- Impossibilité, en règle générale, de connaître le moment d'apparition de l'événement
- Difficile d'inclure son traitement dans le flot synchrone d'un programme
- Nécessité de méthodes pour capturer cet événement
- Fautes d'exécution tel que division par zéro peuvent être synchrone mais imprédictible

Attente Active

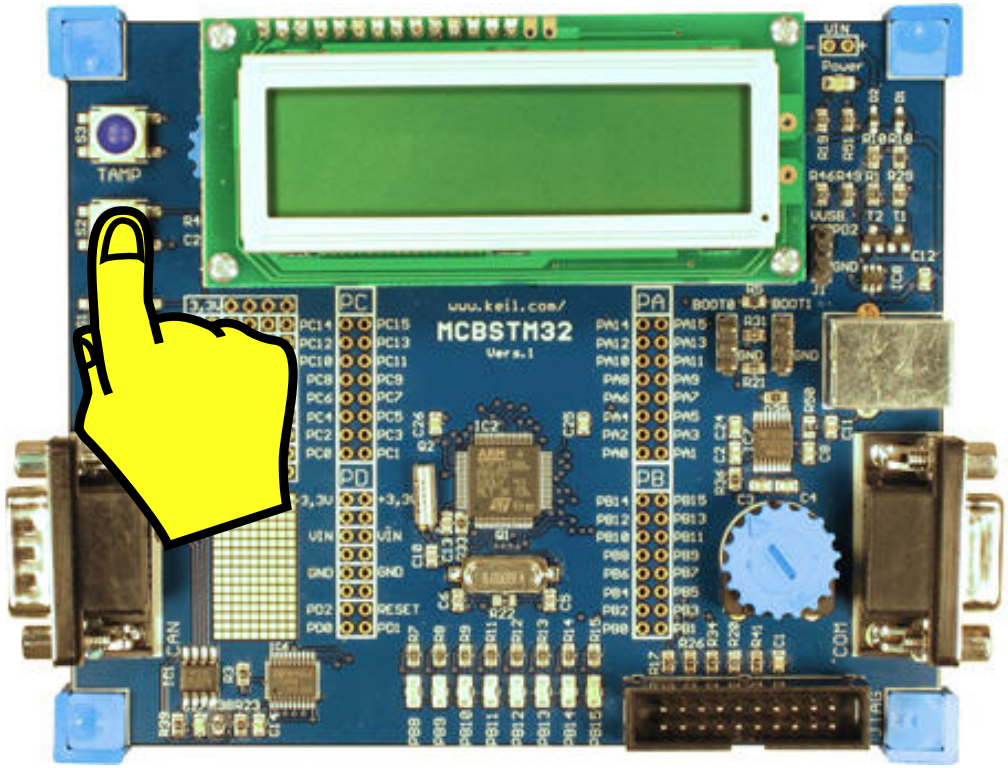


Attente Active



- Détection d'un appui sur un bouton

Attente Active



- Détection d'un appui sur un bouton
- Scrutation périodique

Attente Active

```
#include <stm32f10x_lib.h>
void S2appuye(void)
{
  if (S2presse) {
    if (!((GPIOA->IDR & 0x0001) == 0 )) {   // S2 n'est pas appuyé
        S2presse = 0;
        Tempo(500000);
    }
  }
  else {
    if (((GPIOA->IDR & 0x0001) == 0 ))  {   // S2 est appuyé
        S2presse = 1;
        Tempo(500000);
      }
    }
}
int main(void)
{
  while(1)
  {
   S2appuye();
   }
}
```

Attente Active

- Monopolise du temps processeur inutilement

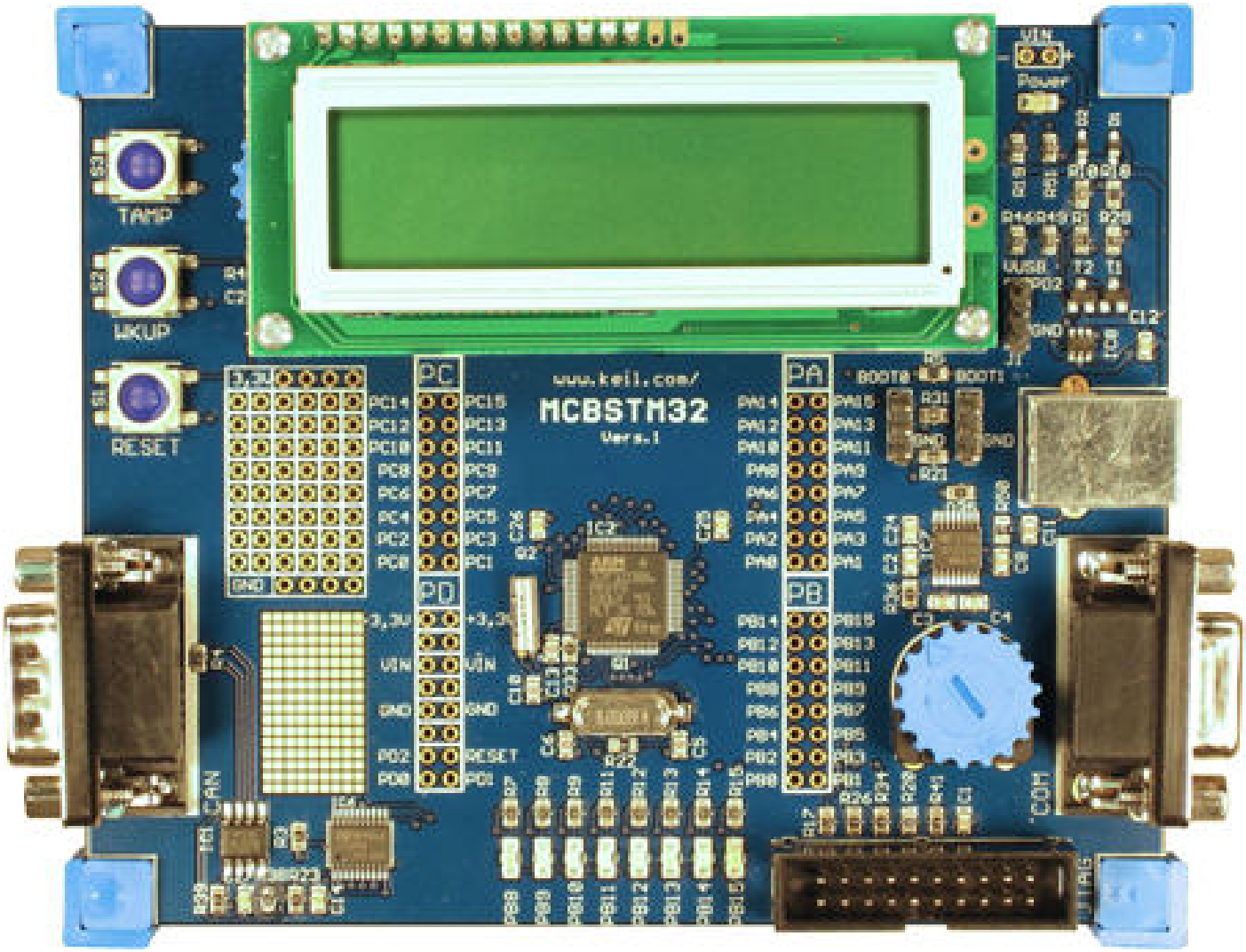
Attente Active

- Monopolise du temps processeur inutilement
- Temps de réaction à l'événement possiblement long

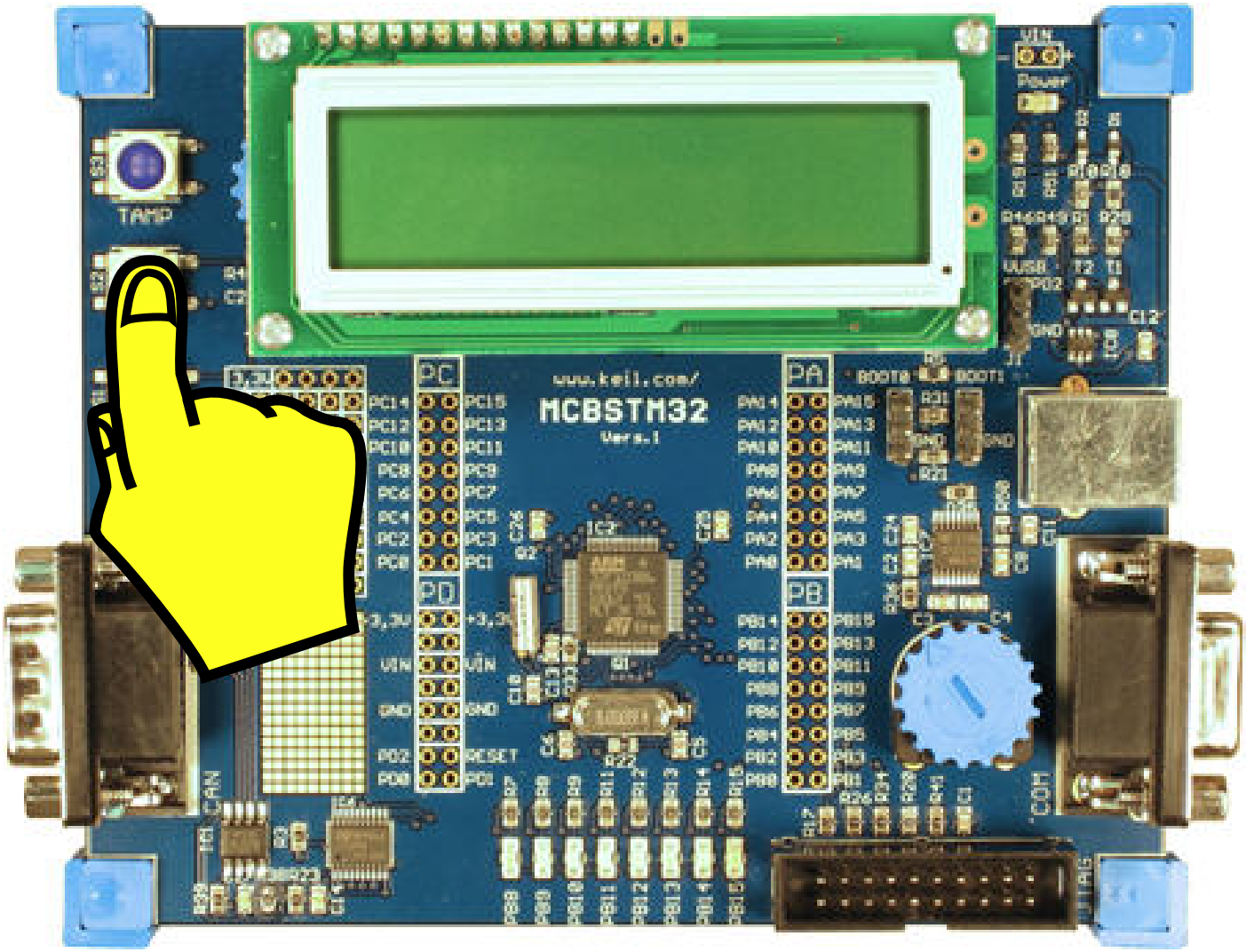
Attente Active

- Monopolise du temps processeur inutilement
- Temps de réaction à l'événement possiblement long
- Difficilement envisageable dans des systèmes temps réel

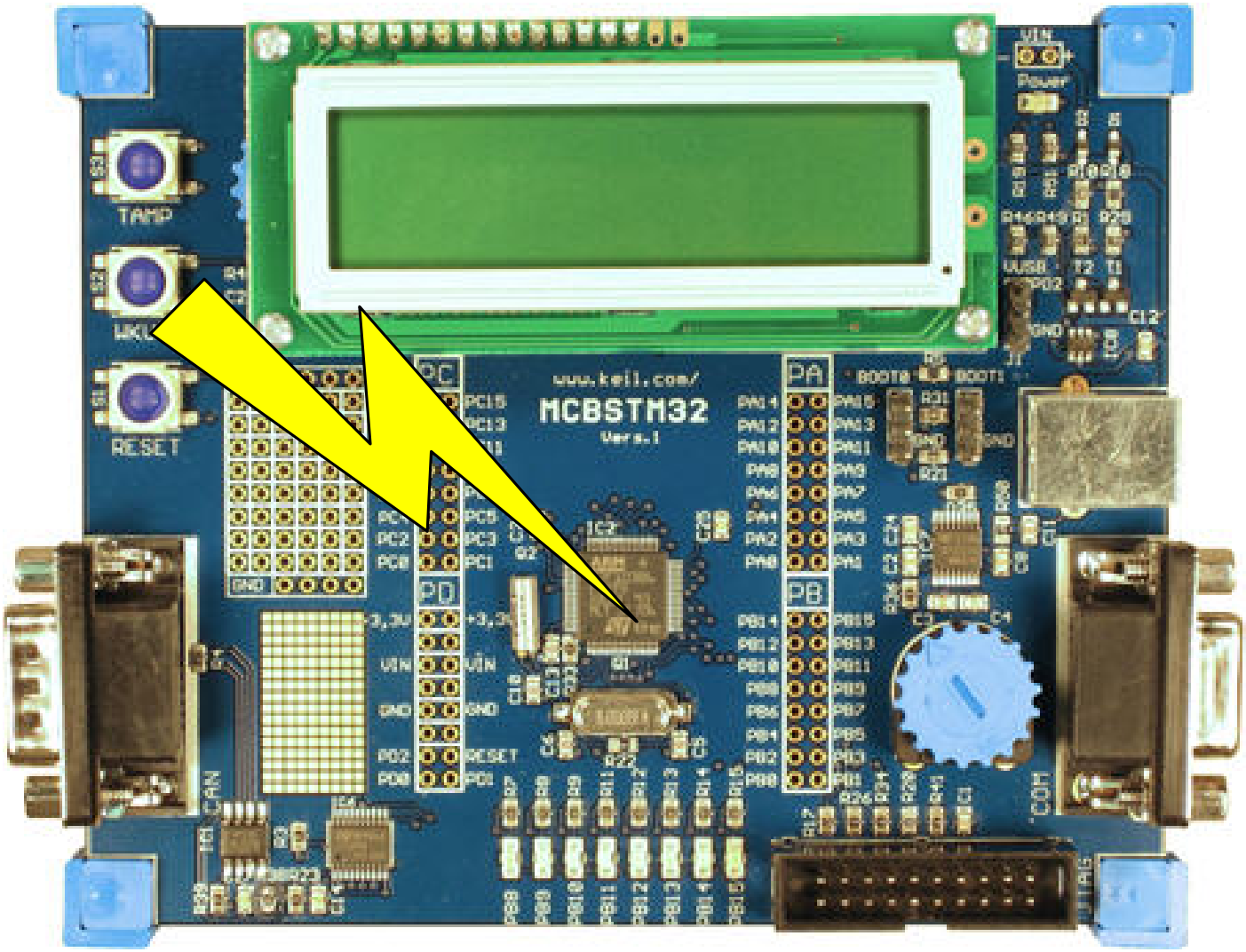
Une autre alternative



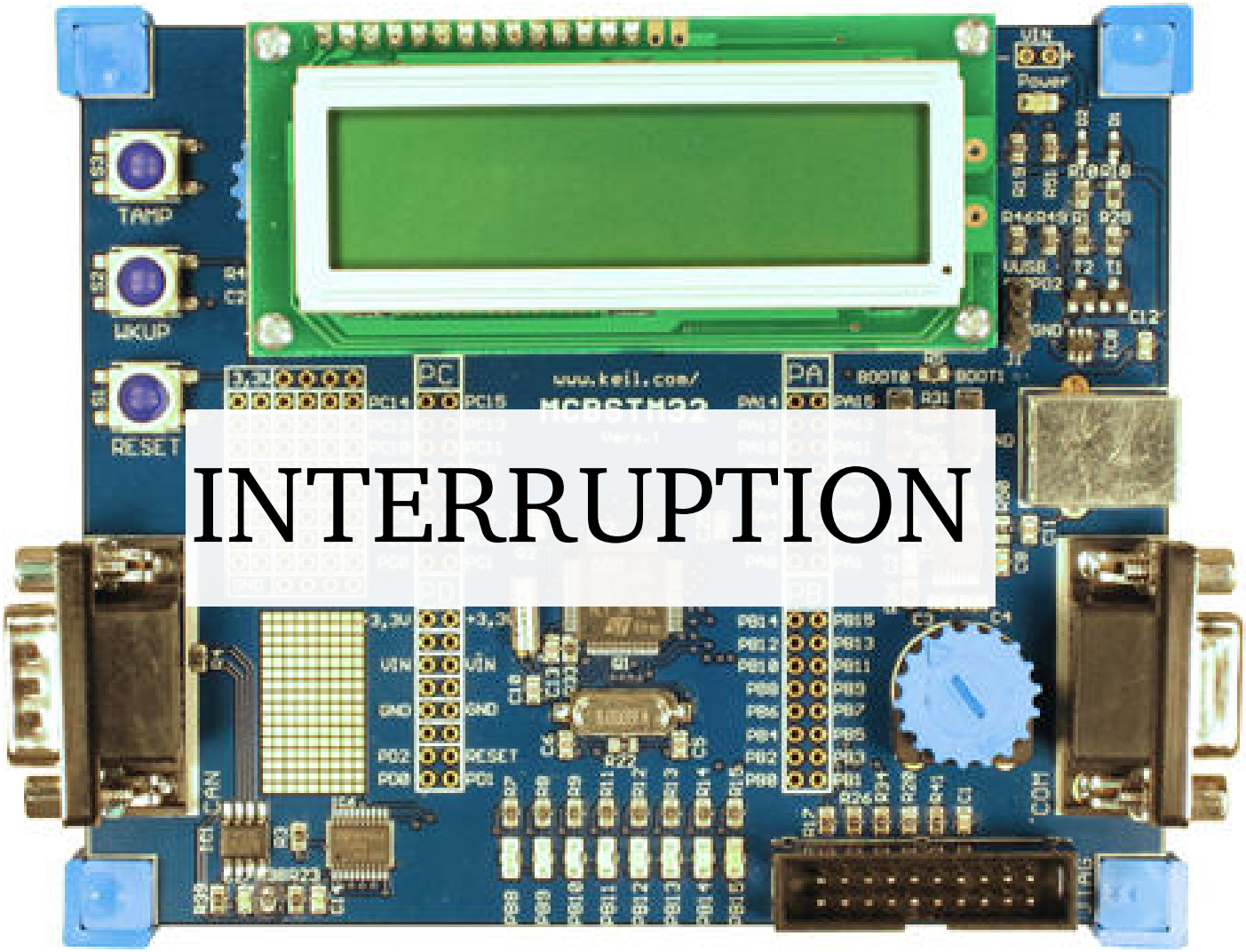
Une autre alternative



Une autre alternative



Une autre alternative



Interruptions : qu'est-ce que c'est ?

- C'est un événement qui provoque

Interruptions : qu'est-ce que c'est ?

- C'est un événement qui provoque
  - l'arrêt de l'exécution normale d'un programme : le programme est interrompu

Interruptions : qu'est-ce que c'est ?

- C'est un événement qui provoque
  - l'arrêt de l'exécution normale d'un programme : le programme est interrompu
  - la sauvegarde de l'état du programme en cours d'exécution : le contexte d'exécution

Interruptions : qu'est-ce que c'est ?

- C'est un événement qui provoque
  - l'arrêt de l'exécution normale d'un programme : le programme est interrompu
  - la sauvegarde de l'état du programme en cours d'exécution : le contexte d'exécution
  - l'exécution d'une routine spécifique liée à cet événement : la routine de service de l'interruption

Interruptions : qu'est-ce que c'est ?

- C'est un événement qui provoque
  - l'arrêt de l'exécution normale d'un programme : le programme est interrompu
  - la sauvegarde de l'état du programme en cours d'exécution : le contexte d'exécution
  - l'exécution d'une routine spécifique liée à cet événement : la routine de service de l'interruption
  - la restauration du contexte d'exécution

Interruptions : qu'est-ce que c'est ?

- C'est un événement qui provoque
  - l'arrêt de l'exécution normale d'un programme : le programme est interrompu
  - la sauvegarde de l'état du programme en cours d'exécution : le contexte d'exécution
  - l'exécution d'une routine spécifique liée à cet événement : la routine de service de l'interruption
  - la restauration du contexte d'exécution
  - le retour à l'exécution du programme interrompu

Interruptions : définitions

- **Interruptions :** surviennent de façon asynchrones. Elles sont commandées par le matériel (interruption externe) ou par le système (interruption système ou exception). Aucune relation avec les instructions en cours d'exécution.

Interruptions : définitions

- **Interruptions :** surviennent de façon asynchrones. Elles sont commandées par le matériel (interruption externe) ou par le système (interruption système ou exception). Aucune relation avec les instructions en cours d'exécution.
- **Exceptions :** déclenchées par des "accidents" dans l'exécution du programme : débordement arithmétique, erreur de bus, tentative d'utilisation d'instructions réservées, erreur d'adressage, défaut de cache, défaut de page...

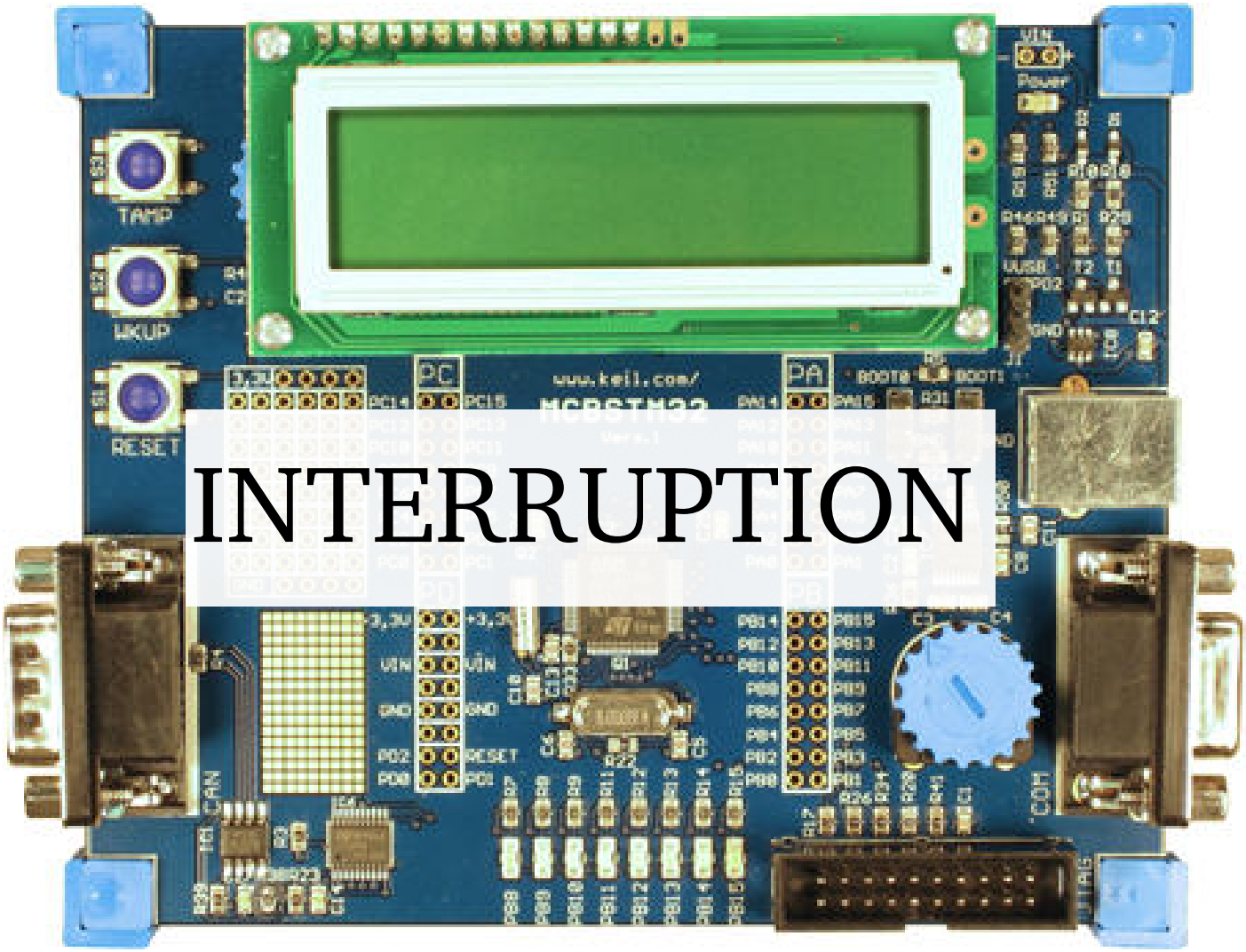
Interruptions : définitions

- **Interruptions :** surviennent de façon asynchrones. Elles sont commandées par le matériel (interruption externe) ou par le système (interruption système ou exception). Aucune relation avec les instructions en cours d'exécution.
- **Exceptions :** déclenchées par des "accidents" dans l'exécution du programme : débordement arithmétique, erreur de bus, tentative d'utilisation d'instructions réservées, erreur d'adressage, défaut de cache, défaut de page...
- **Gestion Précise :** l'état du processeur qui résulterait de l'exécution séquentielle de toutes les instructions antérieures à l'instruction provoquant l'interruption peut être reconstruit dans tous les cas.

Interruptions : définitions

- **Interruptions :** surviennent de façon asynchrones. Elles sont commandées par le matériel (interruption externe) ou par le système (interruption système ou exception). Aucune relation avec les instructions en cours d'exécution.
- **Exceptions :** déclenchées par des "accidents" dans l'exécution du programme : débordement arithmétique, erreur de bus, tentative d'utilisation d'instructions réservées, erreur d'adressage, défaut de cache, défaut de page...
- **Gestion Précise :** l'état du processeur qui résulterait de l'exécution séquentielle de toutes les instructions antérieures à l'instruction provoquant l'interruption peut être reconstruit dans tous les cas.
- **Gestion Imprécise :** Impossibilité de reconstruction de l'état du processeur.

Interruptions STM32



Interruptions STM32

- Les interruptions du STM32 sont dites vectorisées

Interruptions STM32

- Les interruptions du STM32 sont dites vectorisées
- A chaque source d'interruption correspond un vecteur d'interruption sur 32 bits

Interruptions STM32

- Les interruptions du STM32 sont dites vectorisées
- A chaque source d'interruption correspond un vecteur d'interruption sur 32 bits
- Le vecteur d'interruption contient l'adresse de la routine de service de l'interruption

Interruptions STM32

- Les interruptions du STM32 sont dites vectorisées
- A chaque source d'interruption correspond un vecteur d'interruption sur 32 bits
- Le vecteur d'interruption contient l'adresse de la routine de service de l'interruption
- Pour l'exception système reset, la valeur du vecteur est la valeur initiale du compteur de programme

Interruptions STM32

- Les interruptions du STM32 sont dites vectorisées
- A chaque source d'interruption correspond un vecteur d'interruption sur 32 bits
- Le vecteur d'interruption contient l'adresse de la routine de service de l'interruption
- Pour l'exception système reset, la valeur du vecteur est la valeur initiale du compteur de programme
- Les vecteurs d'interruption sont contenus dans une table qui est relogeable

Interruptions STM32

- Pour la gestion des interruption le microcontrôleur STM32 via le CortexM3 fournit

Interruptions STM32

- Pour la gestion des interruption le microcontrôleur STM32 via le CortexM3 fournit
  - Sauvegarde et restauration automatique du contexte d'exécution.

Interruptions STM32

- Pour la gestion des interruption le microcontrôleur STM32 via le CortexM3 fournit
  - Sauvegarde et restauration automatique du contexte d'exécution.
    - Le processeur empile les registres avant de servir l'interruption et les dépile après exécution de la routine de service de l'interruption. Pas de nécessité d'instruction spécifique, c'est automatique et transparent pour le programmeur.

Interruptions STM32

- Pour la gestion des interruption le microcontrôleur STM32 via le CortexM3 fournit
  - Sauvegarde et restauration automatique du contexte d'exécution.
    - Le processeur empile les registres avant de servir l'interruption et les dépile après exécution de la routine de service de l'interruption. Pas de nécessité d'instruction spécifique, c'est automatique et transparent pour le programmeur.
  - Lecture automatique d'une table dite de table de vecteur qui contient les adresses des routines de service des interruptions.

Interruptions STM32

- Pour la gestion des interruption le microcontrôleur STM32 via le CortexM3 fournit
  - Sauvegarde et restauration automatique du contexte d'exécution.
    - Le processeur empile les registres avant de servir l'interruption et les dépile après exécution de la routine de service de l'interruption. Pas de nécessité d'instruction spécifique, c'est automatique et transparent pour le programmeur.
  - Lecture automatique d'une table dite de table de vecteur qui contient les adresses des routines de service des interruptions.
    - Effectué en parallèle de la sauvegarde de contexte.

Interruption STM32

- Le microcontrôleur STM32 a deux types d'interruptions :

Interruption STM32

- Le microcontrôleur STM32 a deux types d'interruptions :
  - les exceptions système

Interruption STM32

- Le microcontrôleur STM32 a deux types d'interruptions :
  - les exceptions système
  - les interruptions externes numérotées de 0 à 59

Interruption STM32

- Le microcontrôleur STM32 a deux types d'interruptions :
  - les exceptions système
  - les interruptions externes numérotées de 0 à 59
- le numéro de l'interruption en cours de service peut-être connue par

Interruption STM32

- Le microcontrôleur STM32 a deux types d'interruptions :
  - les exceptions système
  - les interruptions externes numérotées de 0 à 59
- le numéro de l'interruption en cours de service peut-être connue par
  - le registre spécial IPSR (Interrupt Program Status Register)

Interruption STM32

- Le microcontrôleur STM32 a deux types d'interruptions :
  - les exceptions système
  - les interruptions externes numérotées de 0 à 59
- le numéro de l'interruption en cours de service peut-être connue par
  - le registre spécial IPSR (Interrupt Program Status Register)
  - le registre ICSR (Interrupt Control State Register) du NVIC (Nested Vectored Interrupt Controller) dans le champs VECTACTIVE

Interruption STM32

- Le microcontrôleur STM32 a deux types d'interruptions :
  - les exceptions système
  - les interruptions externes numérotées de 0 à 59
- le numéro de l'interruption en cours de service peut-être connue par
  - le registre spécial IPSR (Interrupt Program Status Register)
  - le registre ICSR (Interrupt Control State Register) du NVIC (Nested Vectored Interrupt Controller) dans le champs VECTACTIVE
- les interruptions ont des priorités permettant d'avoir un système préemptif

Exceptions système

Liste

Numéro Exception	Type	Priorité	Description
1	Reset	-3 (Plus élevée)	Reset
2	NMI	-2	Exception Non Masquable
3	Hard Fault	-1	Toutes les exceptions matérielles
4	MemManageFault	Programmable	Exception du à la gestion mémoire
5	BusFault	Programmable	Erreur matérielle du au bus
6	UsageFault	Programmable	Erreur du au programme
7-10	Reserved	Pas utilisée	—
11	SVC	Programmable	Appel système
12	DebugMonitor	Programmable	Moniteur de mise au point
13	Reserved	Pas utilisée	—
14	PendSV	Programmable	Requête de service d'exception
15	SYSTICK	Programmable	Timer interne cortex M3

Interruptions externes

Liste

Numéro Exception	Type	Priorité
0	Watchdog	Programmable
1	PVD	Programmable
2	TAMPER	Programmable
...	...	...
58	DMA2_Channel3	Programmable
59	DMA2_Channel4_5	Programmable

Table Vecteur Complète

Position	Priority	Type of priority	Acronym	Description	Address
	-	-	-	Reserved	0x0000_0000
	-3	fixed	Reset	Reset	0x0000_0004
	-2	fixed	NMI	Non maskable interrupt. The RCC Clock Security System (CSS) is linked to the NMI vector.	0x0000_0008
	-1	fixed	HardFault	All class of fault	0x0000_000C
	0	settable	MemManage	Memory management	0x0000_0010
	1	settable	BusFault	Pre-fetch fault, memory access fault	0x0000_0014
	2	settable	UsageFault	Undefined instruction or illegal state	0x0000_0018
	-	-	-	Reserved	0x0000_001C - 0x0000_002B
	3	settable	SVCall	System service call via SWI instruction	0x0000_002C
	4	settable	Debug Monitor	Debug Monitor	0x0000_0030
	-	-	-	Reserved	0x0000_0034
	5	settable	PendSV	Pendable request for system service	0x0000_0038
	6	settable	SysTick	System tick timer	0x0000_003C
0	7	settable	WWDG	Window watchdog interrupt	0x0000_0040
1	8	settable	PVD	PVD through EXTI Line detection interrupt	0x0000_0044
2	9	settable	TAMPER	Tamper interrupt	0x0000_0048

Table Vecteur Complète

Position	Priority	Type of priority	Acronym	Description	Address
3	10	settable	RTC	RTC global interrupt	0x0000_004C
4	11	settable	FLASH	Flash global interrupt	0x0000_0050
5	12	settable	RCC	RCC global interrupt	0x0000_0054
6	13	settable	EXTI0	EXTI Line0 interrupt	0x0000_0058
7	14	settable	EXTI1	EXTI Line1 interrupt	0x0000_005C
8	15	settable	EXTI2	EXTI Line2 interrupt	0x0000_0060
9	16	settable	EXTI3	EXTI Line3 interrupt	0x0000_0064
10	17	settable	EXTI4	EXTI Line4 interrupt	0x0000_0068
11	18	settable	DMA1_Channel1	DMA1 Channel1 global interrupt	0x0000_006C
12	19	settable	DMA1_Channel2	DMA1 Channel2 global interrupt	0x0000_0070
13	20	settable	DMA1_Channel3	DMA1 Channel3 global interrupt	0x0000_0074
14	21	settable	DMA1_Channel4	DMA1 Channel4 global interrupt	0x0000_0078
15	22	settable	DMA1_Channel5	DMA1 Channel5 global interrupt	0x0000_007C
16	23	settable	DMA1_Channel6	DMA1 Channel6 global interrupt	0x0000_0080
17	24	settable	DMA1_Channel7	DMA1 Channel7 global interrupt	0x0000_0084
18	25	settable	ADC1_2	ADC1 and ADC2 global interrupt	0x0000_0088
19	26	settable	USB_HP_CAN_TX	USB High Priority or CAN TX interrupts	0x0000_008C
20	27	settable	USB_LP_CAN_RX0	USB Low Priority or CAN RX0 interrupts	0x0000_0090
21	28	settable	CAN_RX1	CAN RX1 interrupt	0x0000_0094
22	29	settable	CAN_SCE	CAN SCE interrupt	0x0000_0098
23	30	settable	EXTI9_5	EXTI Line[9:5] interrupts	0x0000_009C
24	31	settable	TIM1_BRK	TIM1 Break interrupt	0x0000_00A0
25	32	settable	TIM1_UP	TIM1 Update interrupt	0x0000_00A4
26	33	settable	TIM1_TRG_COM	TIM1 Trigger and Commutation interrupts	0x0000_00A8
27	34	settable	TIM1_CC	TIM1 Capture Compare interrupt	0x0000_00AC
28	35	settable	TIM2	TIM2 global interrupt	0x0000_00B0
29	36	settable	TIM3	TIM3 global interrupt	0x0000_00B4
30	37	settable	TIM4	TIM4 global interrupt	0x0000_00B8
31	38	settable	I2C1_EV	I^2C1 event interrupt	0x0000_00BC
32	39	settable	I2C1_ER	I^2C1 error interrupt	0x0000_00C0
33	40	settable	I2C2_EV	I^2C2 event interrupt	0x0000_00C4
34	41	settable	I2C2_ER	I^2C2 error interrupt	0x0000_00C8

Table Vecteur Complète

Position	Priority	Type of priority	Acronym	Description	Address
35	42	settable	SPI1	SPI1 global interrupt	0x0000_00CC
36	43	settable	SPI2	SPI2 global interrupt	0x0000_00D0
37	44	settable	USART1	USART1 global interrupt	0x0000_00D4
38	45	settable	USART2	USART2 global interrupt	0x0000_00D8
39	46	settable	USART3	USART3 global interrupt	0x0000_00DC
40	47	settable	EXTI15_10	EXTI Line[15:10] interrupts	0x0000_00E0
41	48	settable	RTCAlarm	RTC alarm through EXTI line interrupt	0x0000_00E4
42	49	settable	USBWakeup	USB wakeup from suspend through EXTI line interrupt	0x0000_00E8
43	50	settable	TIM8_BRK	TIM8 Break interrupt	0x0000_00EC
44	51	settable	TIM8_UP	TIM8 Update interrupt	0x0000_00F0
45	52	settable	TIM8_TRG_COM	TIM8 Trigger and Commutation interrupts	0x0000_00F4
46	53	settable	TIM8_CC	TIM8 Capture Compare interrupt	0x0000_00F8
47	54	settable	ADC3	ADC3 global interrupt	0x0000_00FC
48	55	settable	FSMC	FSMC global interrupt	0x0000_0100
49	56	settable	SDIO	SDIO global interrupt	0x0000_0104
50	57	settable	TIM5	TIM5 global interrupt	0x0000_0108
51	58	settable	SPI3	SPI3 global interrupt	0x0000_010C
52	59	settable	UART4	UART4 global interrupt	0x0000_0110
53	60	settable	UART5	UART5 global interrupt	0x0000_0114
54	61	settable	TIM6	TIM6 global interrupt	0x0000_0118
55	62	settable	TIM7	TIM7 global interrupt	0x0000_011C
56	63	settable	DMA2_Channel1	DMA2 Channel1 global interrupt	0x0000_0120
57	64	settable	DMA2_Channel2	DMA2 Channel2 global interrupt	0x0000_0124
58	65	settable	DMA2_Channel3	DMA2 Channel3 global interrupt	0x0000_0128
59	66	settable	DMA2_Channel4_5	DMA2 Channel4 and DMA2 Channel5 global interrupts	0x0000_012C

le registre xPSR

	31	30	29	28	27	26:25	24	23:20	19:16	15:10	9	8	7	6	5	4:0
APSR	N	Z	C	V	Q											
IPSR												Exception Number				
EPSR						ICI/IT	T			ICI/IT						

	31	30	29	28	27	26:25	24	23:20	19:16	15:10	9	8	7	6	5	4:0
xPSR	N	Z	C	V	Q	ICI/IT	T			ICI/IT		Exception Number				

PRIMASK, FAULTMASK, BASEPRI

Nom	Valeur Initiale	Description
PRIMASK	0	Registre 1 bit. Autorise l'interruption NMI et les exceptions matérielles. Toutes les autres interruptions sont interdites
FAULTMASK	0	Registre 1 bit. Autorise seulement l'interruption NMI. Toutes les autres interruptions sont interdites
BASEPRI	0	Registre 8 bits. Défini le niveau de priorité. Permet d'interdire toute interruption de niveau même niveau ou de niveau plus faible. Les interruptions de niveau plus haut sont autorisées.

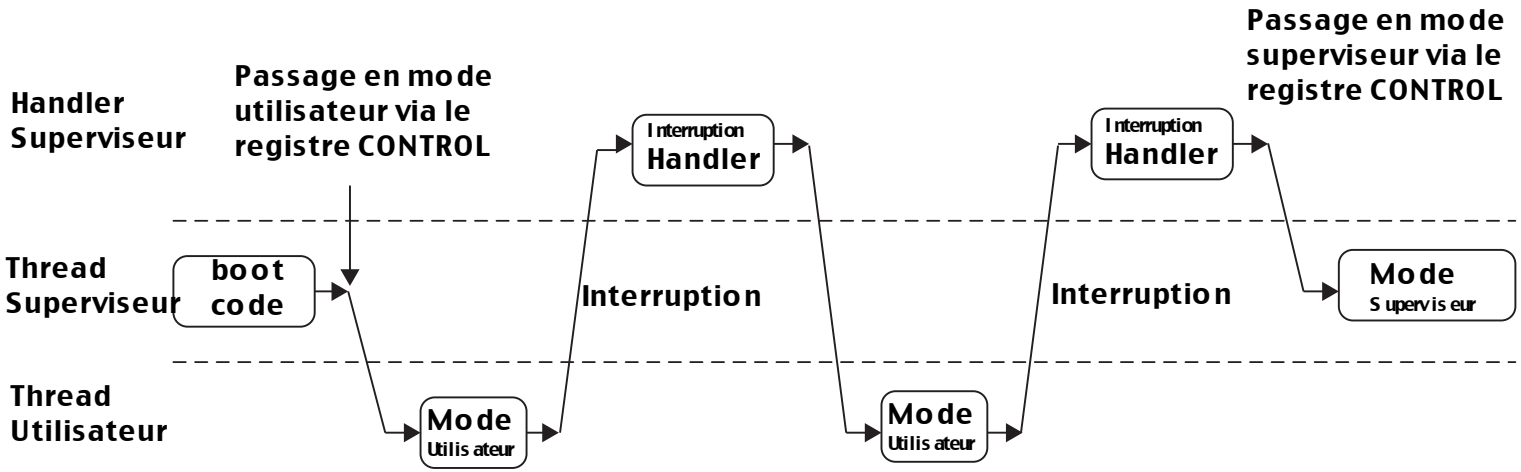
Registre CONTROL

bit	Description
1	Etat de la Pile
	= 0 Pile principale MSP
	= 1 Pile alternative
0	= 0 Mode superviseur
	= 1 Mode utilisateur

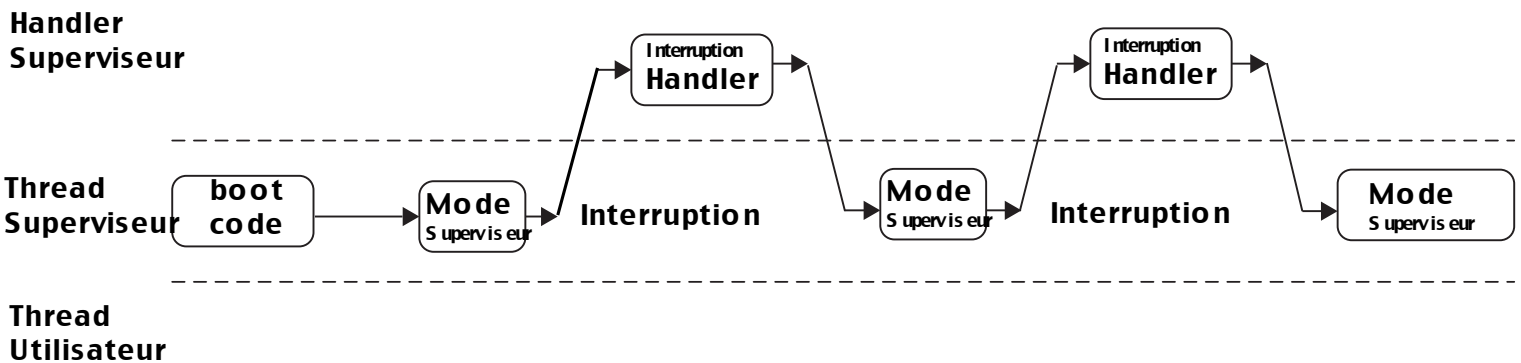
Registre CONTROL

	Superviseur	Utilisateur
Interruption	mode Handler CONTROL[1]=0	Non Autorisé
Programme Principal	mode Thread CONTROL[0]=0 CONTROL[1]=0(1)	mode Thread CONTROL[0]=1 CONTROL[1]=0(1)

Déroulement

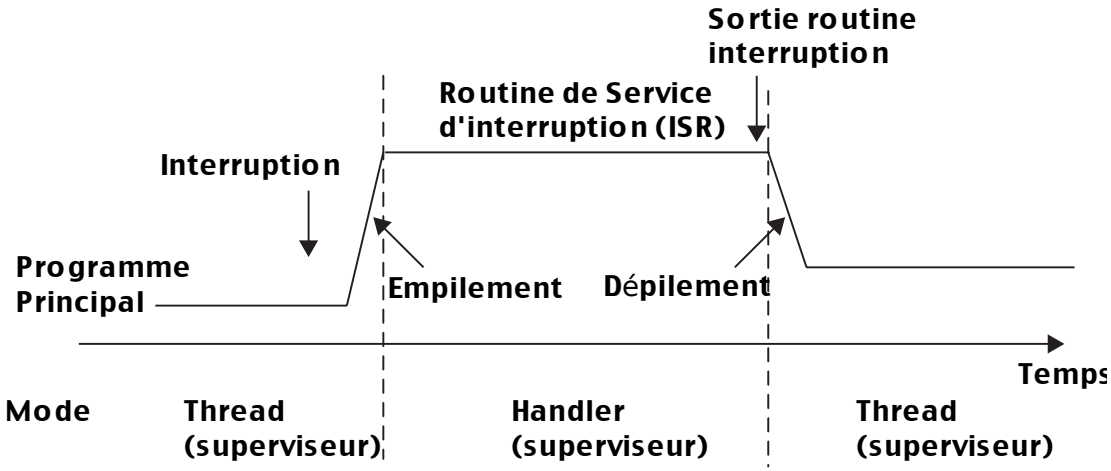


Mode Utilisateur

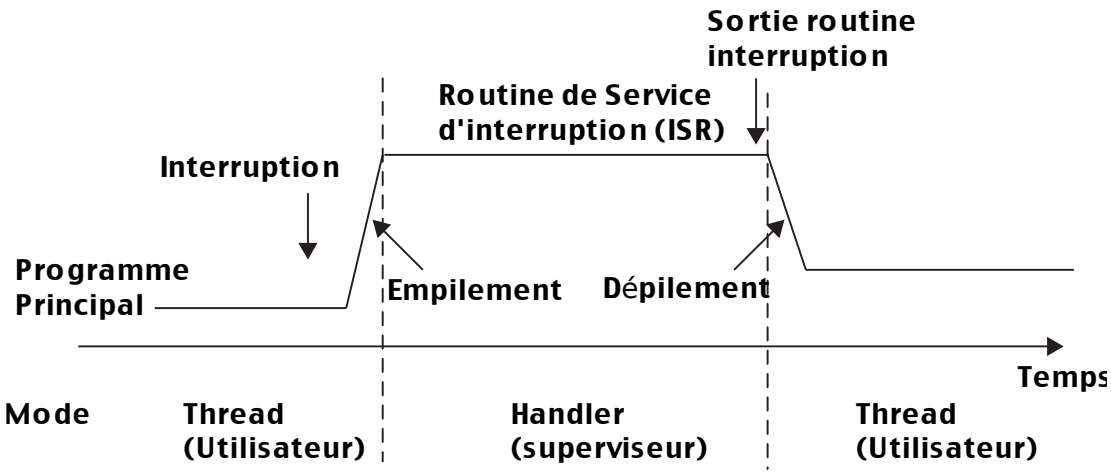


Mode Superviseur

Déroulement



Mode Superviseur



Mode Utilisateur

Le besoin de priorités

- Certains événements provoquant des interruptions sont plus critiques que d'autres

Le besoin de priorités

- Certains événements provoquant des interruptions sont plus critiques que d'autres

Le besoin de priorités

- Certains événements provoquant des interruptions sont plus critiques que d'autres

  - Premier Evénement : Demande d'affichage d'un message de contrôle sur l'écran du pilote d'un avion de ligne

Le besoin de priorités

- Certains événements provoquant des interruptions sont plus critiques que d'autres

  - Premier Evénement : Demande d'affichage d'un message de contrôle sur l'écran du pilote d'un avion de ligne
  - Seconde Evénement : Extinction d'un réacteur de cet avion

Le besoin de priorités

- Certains événements provoquant des interruptions sont plus critiques que d'autres

  - Premier Evénement : Demande d'affichage d'un message de contrôle sur l'écran du pilote d'un avion de ligne
  - Seconde Evénement : Extinction d'un réacteur de cet avion
- Il semble qu'il soit plus urgent de redémarrer le réacteur que d'afficher des informations

Le besoin de priorités

- Nécessité de définir une politique de gestion des priorités

Le besoin de priorités

- Nécessité de définir une politique de gestion des priorités
- Elle doit permettre qu'une interruption plus prioritaire interrompe une interruption moins prioritaire en cours d'exécution

Le besoin de priorités

- Nécessité de définir une politique de gestion des priorités
- Elle doit permettre qu'une interruption plus prioritaire interrompe une interruption moins prioritaire en cours d'exécution
- Elle doit interdire qu'une interruption moins prioritaire interrompe une interruption plus prioritaire en cours d'exécution

Le besoin de priorités

- Nécessité de définir une politique de gestion des priorités
- Elle doit permettre qu'une interruption plus prioritaire interrompe une interruption moins prioritaire en cours d'exécution
- Elle doit interdire qu'une interruption moins prioritaire interrompe une interruption plus prioritaire en cours d'exécution
- Elle peut fixer des priorités non modifiables

Le besoin de priorités

- Nécessité de définir une politique de gestion des priorités
- Elle doit permettre qu'une interruption plus prioritaire interrompe une interruption moins prioritaire en cours d'exécution
- Elle doit interdire qu'une interruption moins prioritaire interrompe une interruption plus prioritaire en cours d'exécution
- Elle peut fixer des priorités non modifiables
  - Cas du reset par exemple

Le besoin de priorités

- Nécessité de définir une politique de gestion des priorités
- Elle doit permettre qu'une interruption plus prioritaire interrompe une interruption moins prioritaire en cours d'exécution
- Elle doit interdire qu'une interruption moins prioritaire interrompe une interruption plus prioritaire en cours d'exécution
- Elle peut fixer des priorités non modifiables
  - Cas du reset par exemple
- Pour plus de souplesse de programmation elle doit offrir la possibilité de choisir le niveau de priorité de la plupart des sources d'interruption

Schéma des priorités du STM32

Priorités à 3 bits

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Actifs			Inactifs				

- 8 niveaux de priorités

Priorités à 4 bits

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Actifs				Inactifs			

- 16 niveaux de priorités

Registre AIRCR

Adresse : 0xE000ED0C

Bits	Nom	Type	Valeur Reset	Description
31:16	VECTKEY	R/W	–	0x05FA
15	ENDIANNESS	R	–	Indique l'arrangement des octets : 1 pour Big Endian, 0 pour Little Endian
10:8	PRIGROUP	R/W	0	Numéro du groupe de priorités
2	SYSRESETREQ	W	–	Requête de contrôle du circuit pour générer un reset
1	VECTCLRACTIVE	W	–	Efface toutes les informations sur l'état des exceptions
0	VECTRESET	W	–	Reset du coeur du processeur CortexM3

Schéma des priorités avec utilisation des groupes de priorités

liste

Groupe de Priorité	Priorité	Sous-Priorité	Nombre de Priorités	Nombre de Sous-Priorités
0	Bit[7:1]	Bit[0]	128	2
1	Bit[7:2]	Bit[1:0]	64	4
2	Bit[7:3]	Bit[2:0]	32	8
3	Bit[7:4]	Bit[3:0]	16	16
4	Bit[7:5]	Bit[4:0]	8	32
5	Bit[7:6]	Bit[5:0]	4	64
6	Bit[7]	Bit[6:0]	2	128
7	Aucune	Bit[7:0]	0	256

Schéma des priorités avec utilisation des groupes de priorités

Priorités à 3 bits avec Groupe de priorité égal à 5

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Priorité		Sous-Priorité	Inactifs				

- 4 niveaux de priorités et 2 niveaux de sous-priorités

Priorités à 3 bits avec Groupe égal à 1

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Priorité			Priorité toujours à 0			Sous Priorité toujours à 0	

- 8 niveaux de priorités

Schéma des priorités avec utilisation des groupes de priorités

Priorités à 3 bits avec Groupe égal à 1

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Priorité							Sous Priorité

- 8 niveaux de priorités

Vectorisation dans le STM32

- Quand une interruption intervient il est nécessaire d'identifier la bonne routine de service

Vectorisation dans le STM32

- Quand une interruption intervient il est nécessaire d'identifier la bonne routine de service
- Pour cela il faut déterminer l'adresse du programme correspondant à cette routine

Vectorisation dans le STM32

- Quand une interruption intervient il est nécessaire d'identifier la bonne routine de service
- Pour cela il faut déterminer l'adresse du programme correspondant à cette routine
- Cette information est stockée dans une table des vecteurs d'interruption

Vectorisation dans le STM32

- Quand une interruption intervient il est nécessaire d'identifier la bonne routine de service
- Pour cela il faut déterminer l'adresse du programme correspondant à cette routine
- Cette information est stockée dans une table des vecteurs d'interruption
- Par défaut cette table est stockée à l'adresse 0

Vectorisation dans le STM32

- Quand une interruption intervient il est nécessaire d'identifier la bonne routine de service
- Pour cela il faut déterminer l'adresse du programme correspondant à cette routine
- Cette information est stockée dans une table des vecteurs d'interruption
- Par défaut cette table est stockée à l'adresse 0
- Cette table est relogeable via le registre VTOR (Vector Table Offset Register)

Vectorisation dans le STM32

- Quand une interruption intervient il est nécessaire d'identifier la bonne routine de service
- Pour cela il faut déterminer l'adresse du programme correspondant à cette routine
- Cette information est stockée dans une table des vecteurs d'interruption
- Par défaut cette table est stockée à l'adresse 0
- Cette table est relogeable via le registre VTOR (Vector Table Offset Register)
- La table est organisée en mot de 4 octets (32 bits)

Vectorisation dans le STM32

- Quand une interruption intervient il est nécessaire d'identifier la bonne routine de service
- Pour cela il faut déterminer l'adresse du programme correspondant à cette routine
- Cette information est stockée dans une table des vecteurs d'interruption
- Par défaut cette table est stockée à l'adresse 0
- Cette table est relogeable via le registre VTOR (Vector Table Offset Register)
- La table est organisée en mot de 4 octets (32 bits)
- L'adresse du vecteur d'une interruption est calculée en multipliant par 4 son numéro.

Vecteur Interruption

Table des vecteurs d'exception après mise sous tension

Adresse	Numéro de l'exception	Valeur
0x00000000	–	Valeur initiale de MSP
0x00000004	1	Valeur initiale du Compteur de Programme (vecteur de Reset)
0x00000008	2	Adresse de la fonction de gestion de l'exception NMI
0x0000000C	3	Adresse de la fonction de gestion de l'exception HardFault
...	...	Autres adresse des fonctions de gestion des exceptions

Registre VTO

Adresse 0xE000ED08

Bits	Nom	Type	Valeur initiale	Description
29	TBLBASE	R/W	0	Base de la table : 0 pour ROM, 1 pour RAM
28:7	TBLOFF	R/W	0	Valeur du décalage de la table des vecteurs d'exceptions

Traitement d'une interruption

- Quand une interruption est servie il est réalisé :

Traitement d'une interruption

- Quand une interruption est servie il est réalisé :
  - L'empilement de 8 registres

Traitement d'une interruption

- Quand une interruption est servie il est réalisé :
  - L'empilement de 8 registres
  - Le chargement du vecteur de la routine d'interruption

Traitement d'une interruption

- Quand une interruption est servie il est réalisé :
  - L'empilement de 8 registres
  - Le chargement du vecteur de la routine d'interruption
  - La mise à jour du pointeur de pile, du registre de lien (LR) et du compteur de programme(PC)

Empilement des registres

- Empilement des registres R0 à R3,R12,LR, PC et PSR

Empilement des registres

- Empilement des registres R0 à R3,R12,LR, PC et PSR
- Utilisation de la Pile utilisateur PSP si le code du programme était exécuté en mode utilisateur

Empilement des registres

- Empilement des registres R0 à R3,R12,LR, PC et PSR
- Utilisation de la Pile utilisateur PSP si le code du programme était exécuté en mode utilisateur
- Utilisation de la Pile principale MSP si le code du programme était exécuté en mode superviseur

Empilement des registres

- Empilement des registres R0 à R3,R12,LR, PC et PSR
- Utilisation de la Pile utilisateur PSP si le code du programme était exécuté en mode utilisateur
- Utilisation de la Pile principale MSP si le code du programme était exécuté en mode superviseur
- Pendant le service de l'interruption dans tous les cas seule la pile principale MSP sera utilisée

Empilement des registres

- Empilement des registres R0 à R3,R12,LR, PC et PSR
- Utilisation de la Pile utilisateur PSP si le code du programme était exécuté en mode utilisateur
- Utilisation de la Pile principale MSP si le code du programme était exécuté en mode superviseur
- Pendant le service de l'interruption dans tous les cas seule la pile principale MSP sera utilisée
- Pour plus d'efficacité les valeurs de PC et PSR sont les premières à être empilées

Empilement des registres

Adresse	Données	Ordre d'empilement
Ancien SP(N)		
N-4	PSR	2
N-8	PC	1
N-12	LR	8
N-16	R12	7
N-20	R3	6
N-24	R2	5
N-28	R1	4
Nouveau SP(N-32)	R0	3

Chargement du vecteur d'interruption

- Le bus de données est occupé aux empilements de registres

Chargement du vecteur d'interruption

- Le bus de données est occupé aux empilements de registres
- Sur le bus d'instruction est chargé le vecteur d'interruption

Chargement du vecteur d'interruption

- Le bus de données est occupé aux empilements de registres
- Sur le bus d'instruction est chargé le vecteur d'interruption
- Puisque les deux opérations utilisent deux bus disctinct elles sont réalisées en parallèle

Mise à jour des registres

- Le pointeur de pile (MSP ou PSP) est actualisé après empilement des registres

Mise à jour des registres

- Le pointeur de pile (MSP ou PSP) est actualisé après empilement des registres
- La partie IPSR du registre d'état PSR est mise à jour avec le numéro de l'interruption

Mise à jour des registres

- Le pointeur de pile (MSP ou PSP) est actualisé après empilement des registres
- La partie IPSR du registre d'état PSR est mise à jour avec le numéro de l'interruption
- Le compteur de programme est mise à la valeur de l'adresse de la routine de service de l'interruption via le vecteur d'interruption

Mise à jour des registres

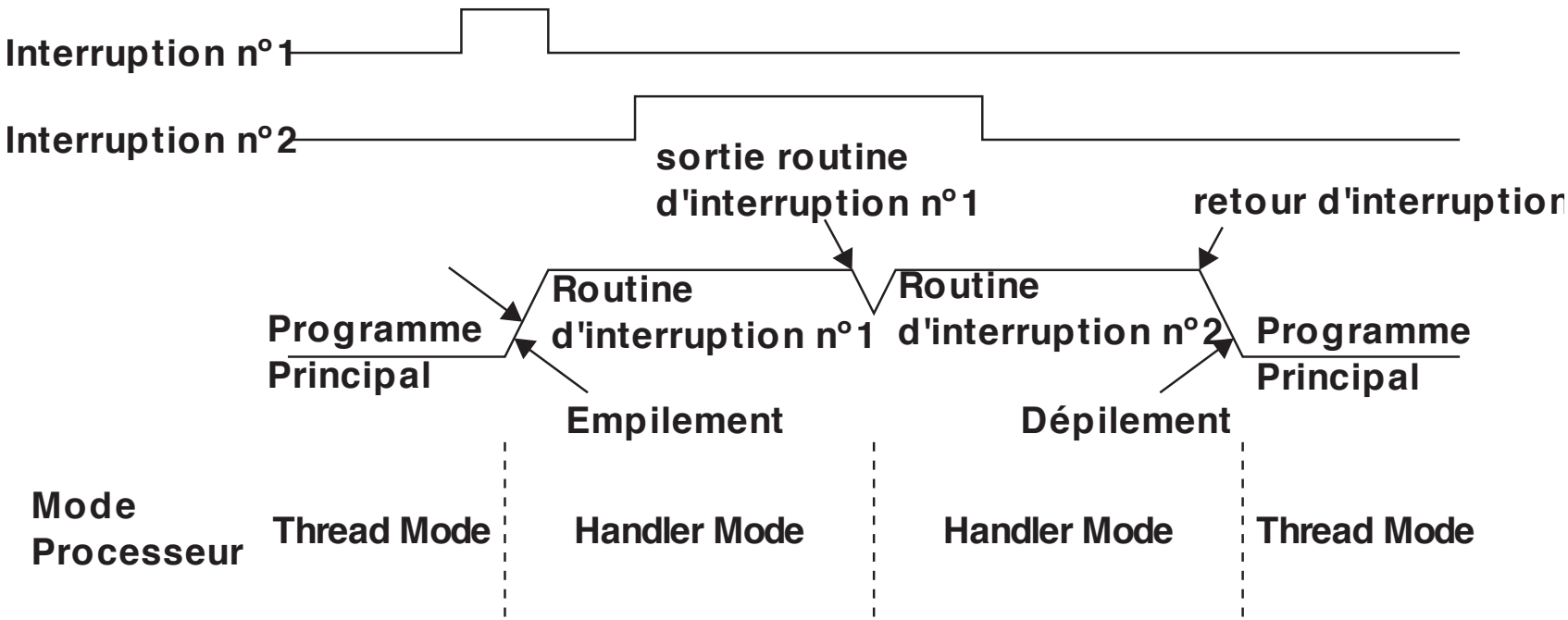
- Le pointeur de pile (MSP ou PSP) est actualisé après empilement des registres
- La partie IPSR du registre d'état PSR est mise à jour avec le numéro de l'interruption
- Le compteur de programme est mise à la valeur de l'adresse de la routine de service de l'interruption via le vecteur d'interruption
- Le registre de lien LR est mis à jour avec une valeur spéciale appelée EXC_RETURN permettant de gérer les retours d'interruption. Les 4 bits de poids faible de ce nouveau LR contiennent l'information de retour.

Valeur EXC_RETURN

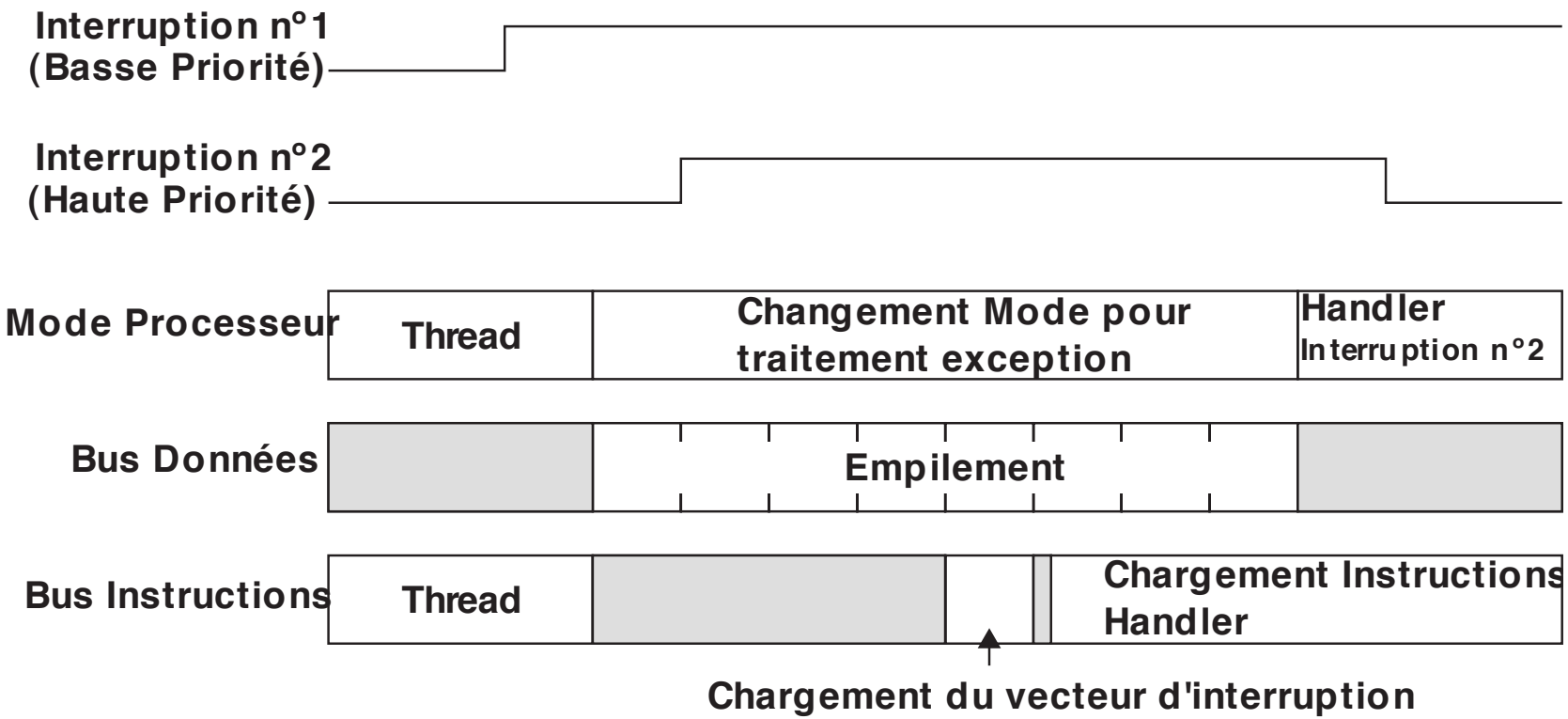
Bits	31 à 4	3	2	1	0
Description	0xFFFFFFF	Mode de retour (Thread/Handler)	Pile de retour (MSP/PSP)	0	Etat Processus (Thumb/ARM)

Valeur	Condition
0xFFFFFFF1	Retour à une routine de service d'interruption
0xFFFFFFF9	Retour à un programme principal en mode superviseur
0xFFFFFFFD	Retour à un programme principal en mode utilisateur

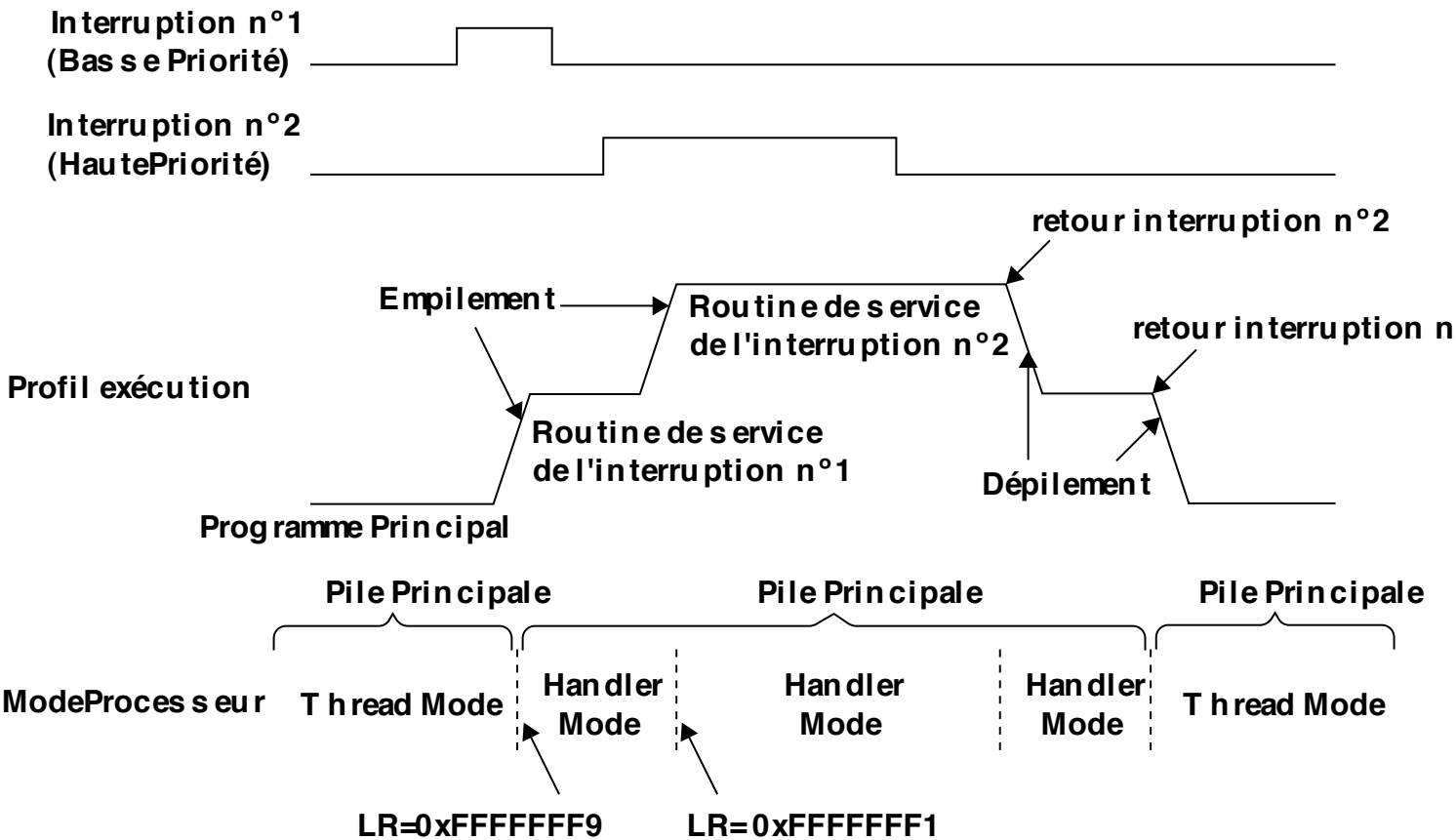
Augmenter la réactivité du service des interruptions



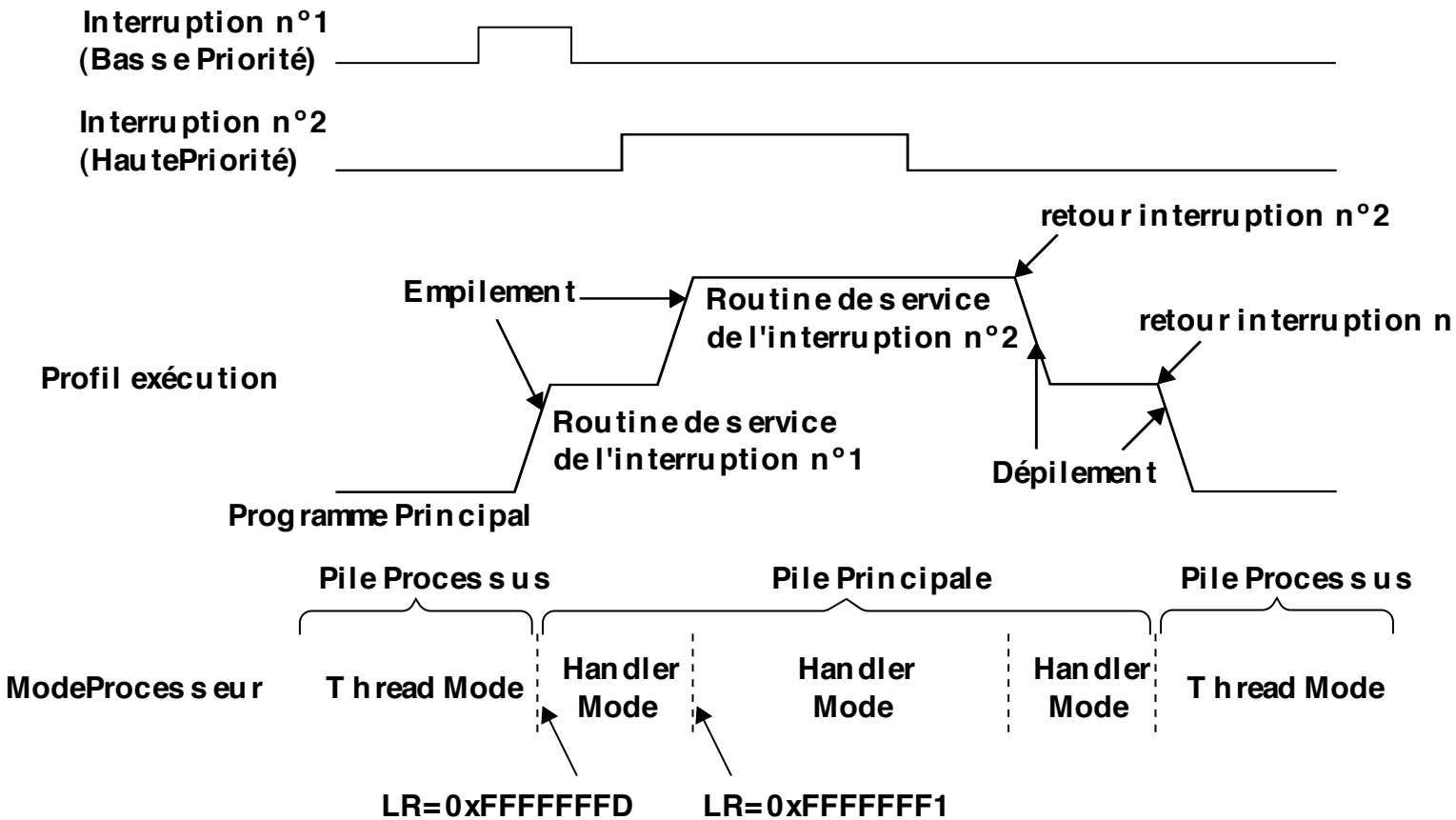
Augmenter la réactivité du service des interruptions



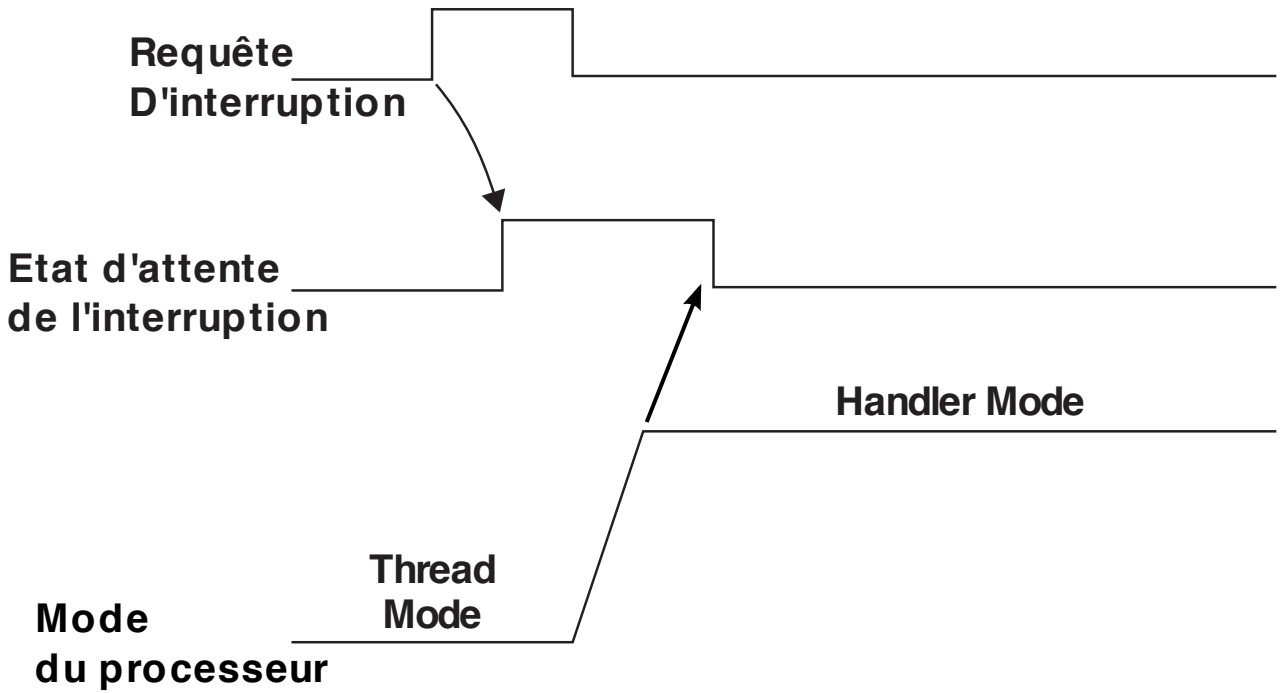
Déterminer les retours d'interruption en mode superviseur



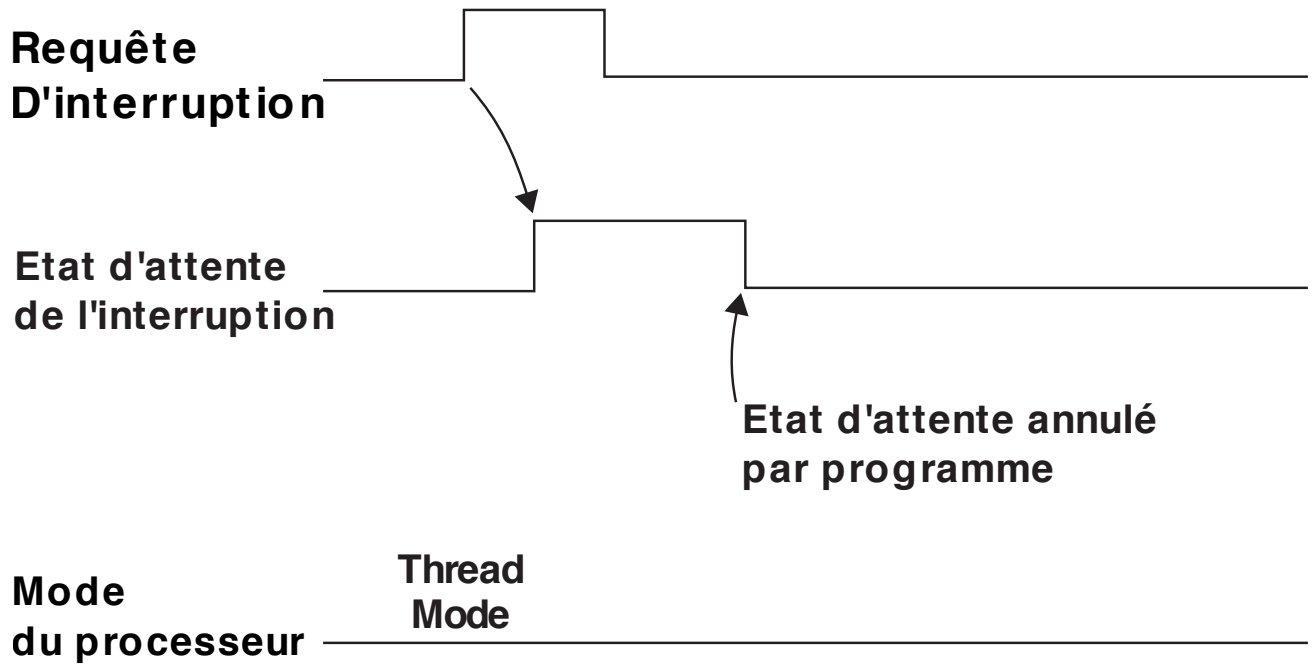
Déterminer les retours d'interruption en mode utilisateur



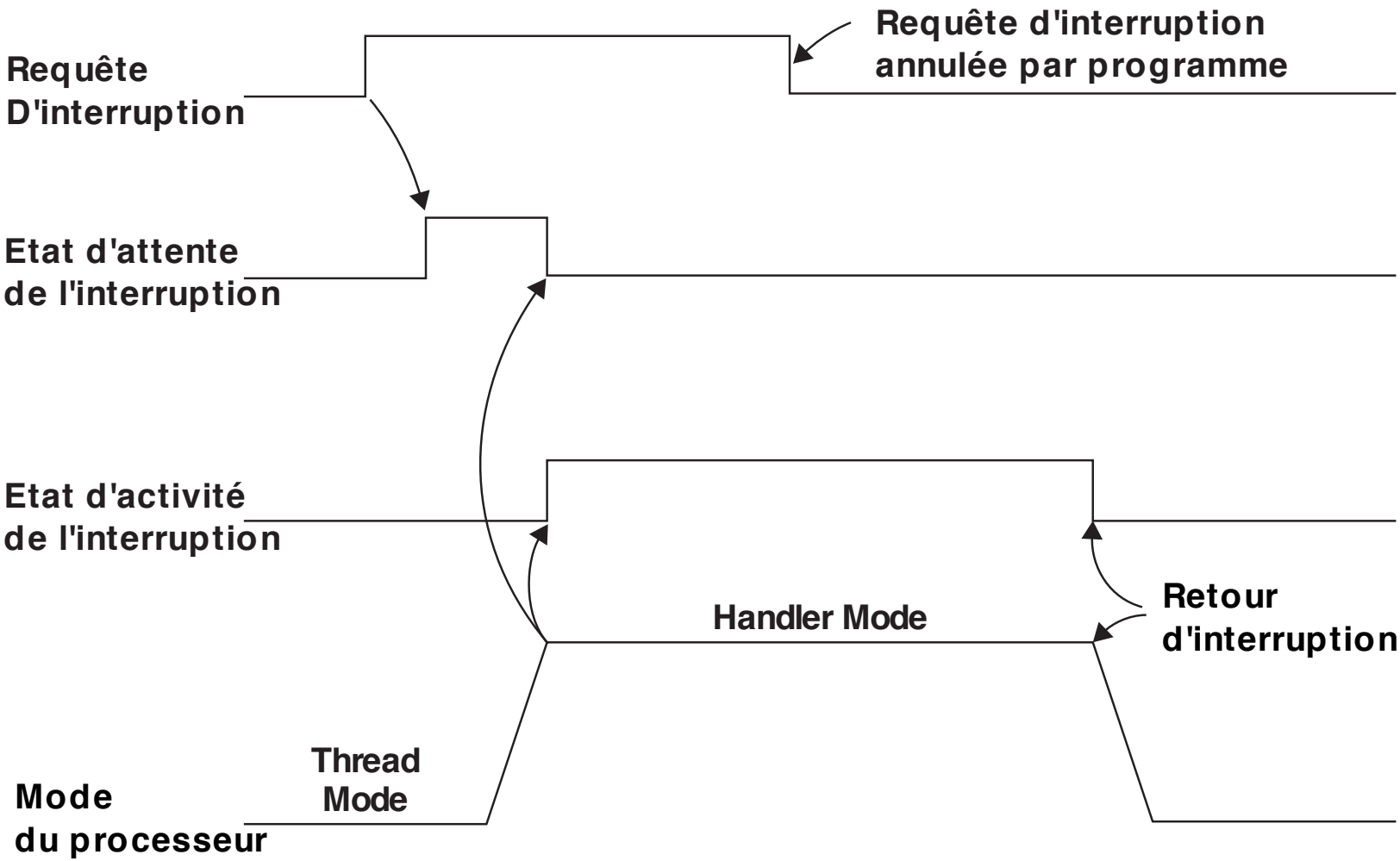
Mise en attente interruption



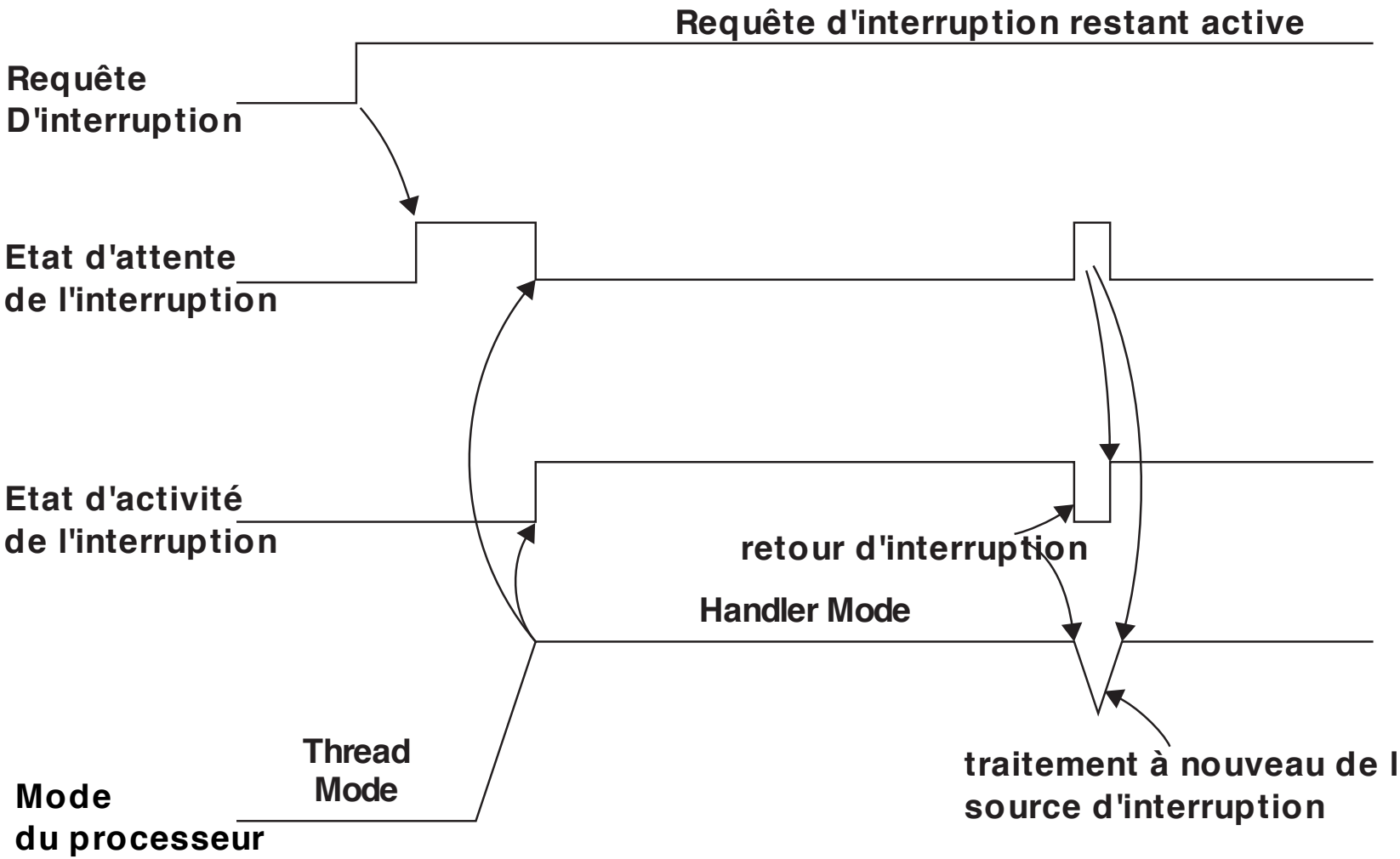
Mise en attente interruption avec raz avant service



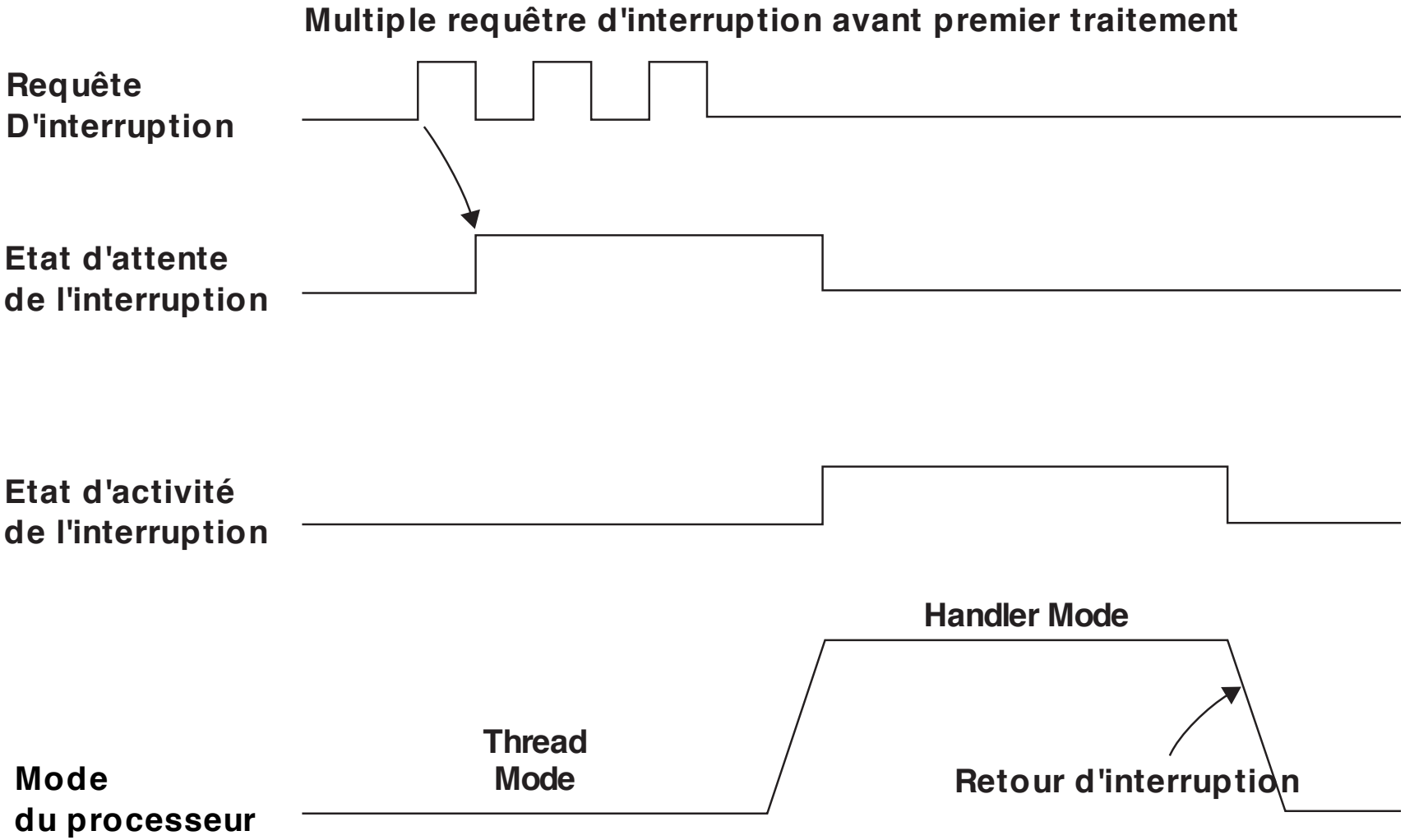
Requête maintenue lors du début de service de l'interruption



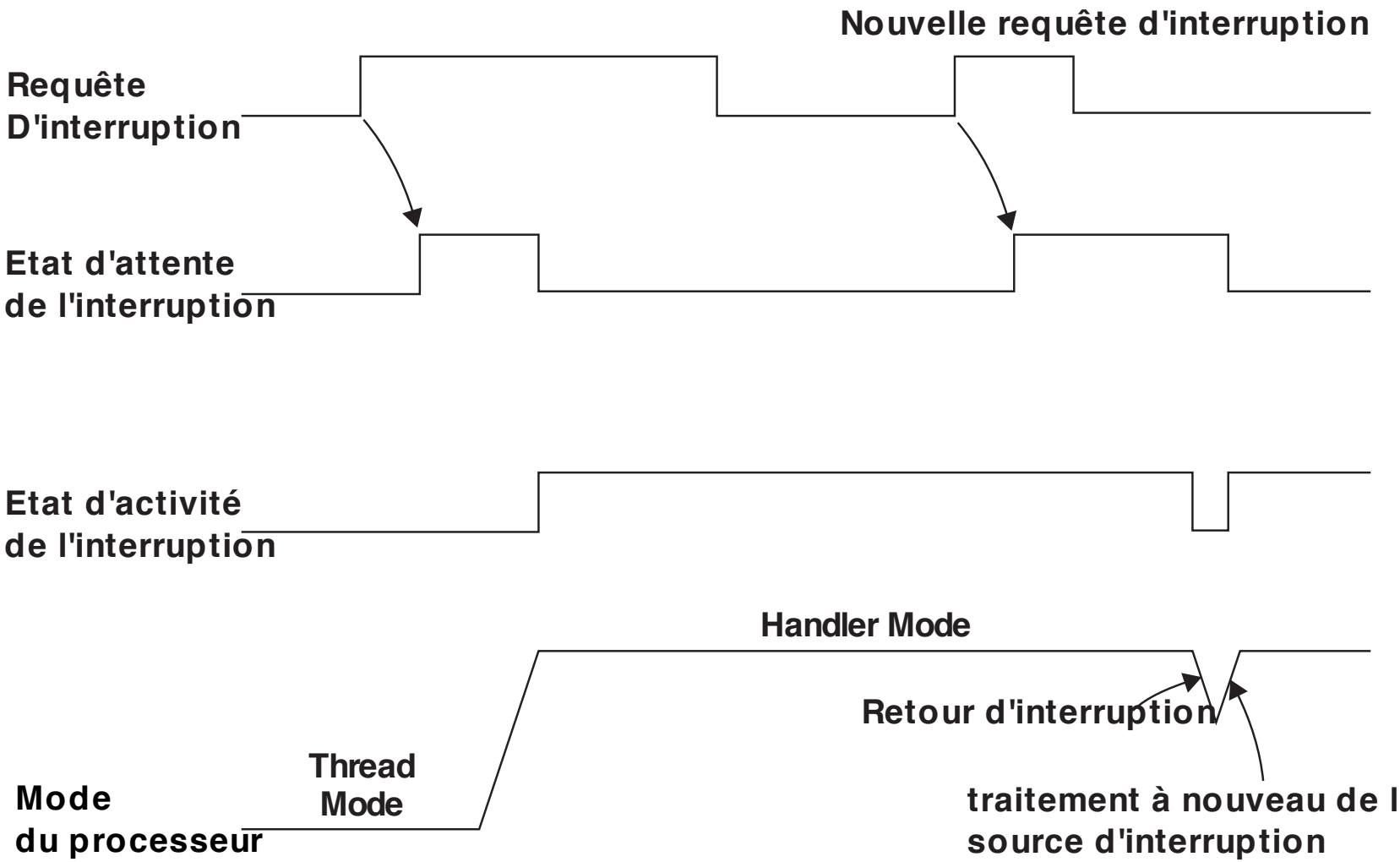
Requête maintenue après service de l'interruption



Requête hoquetante



Requête pendant service



Gestion des exceptions système

registre BFS

Adresse 0xE000ED29

Bits	Nom	Type	Valeur Initiale	Description
7	BFARVALID	–	0	Indique si le contenu du registre BFAR est valide
6:5	–	–	–	–
4	STKERR	R/Wc	0	Erreur d'empilement
3	UNSTKERR	R/Wc	0	Erreur de dépilement
2	IMPRECISERR	R/Wc	0	Violation d'accès aux données imprécis
1	PRECISERR	R/Wc	0	Violation d'accès aux données précis
0	IBUSERR	R/Wc	0	Violation d'accès aux instructions

registre MMS

Adresse 0xE000ED28

Bits	Nom	Type	Valeur Initiale	Description
7	MMARVALID	–	0	Indique si le contenu du registre MMAR est valide
6:5	–	–	–	–
4	MSTKERR	R/Wc	0	Erreur d'empilement
3	MUNSTKERR	R/Wc	0	Erreur de dépilement
2	–	–	–	–
1	DACCVIOL	R/Wc	0	Violation d'accès aux données
0	IACCVIOL	R/Wc	0	Violation d'accès aux instructions

registre UFS

Adresse 0xE000ED2A

Bits	Nom	Type	Valeur Initiale	Description
9	DIVBYZERO	–	0	Indique si qu'une division par zéro a eu lieu (nécessite que DIV_0_TRP soit positionné)
8	UNALIGNED	R/Wc	0	Indique qu'un accès aux donnée non-aligné a eu lieu
7:4	–	–	–	–
3	NOCP	R/Wc	0	Indique un essai d'exécution d'une instruction coprocesseur
2	INVCP	R/Wc	0	Indique un essai de retour d'exception avec une valeur erronée du PC
1	INVSTATE	R/Wc	0	Indique un essai de changement d'état invalide (état ARM)
0	UNDEFINSTR	R/Wc	0	Indique un essai d'exécution d'une instruction non définie

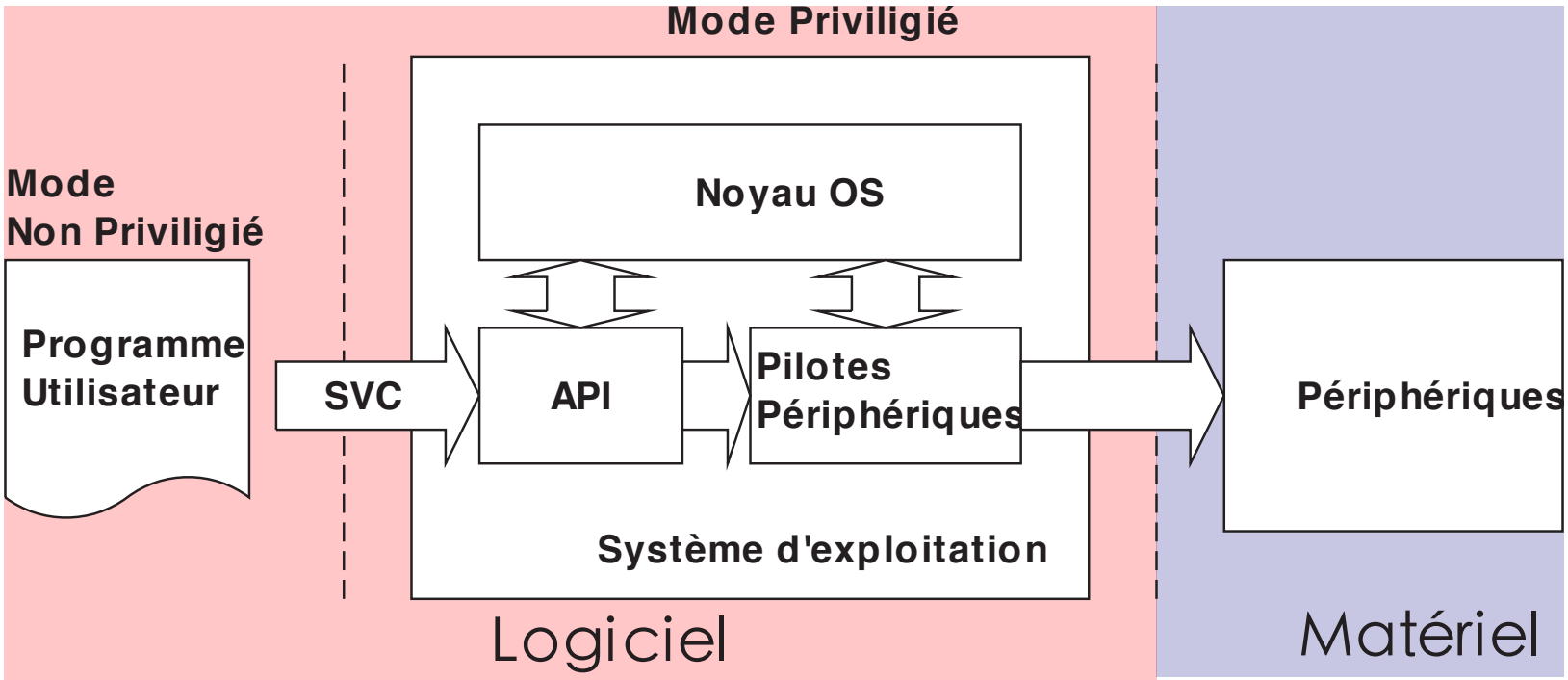
registre HFS

Adresse 0xE000ED2C

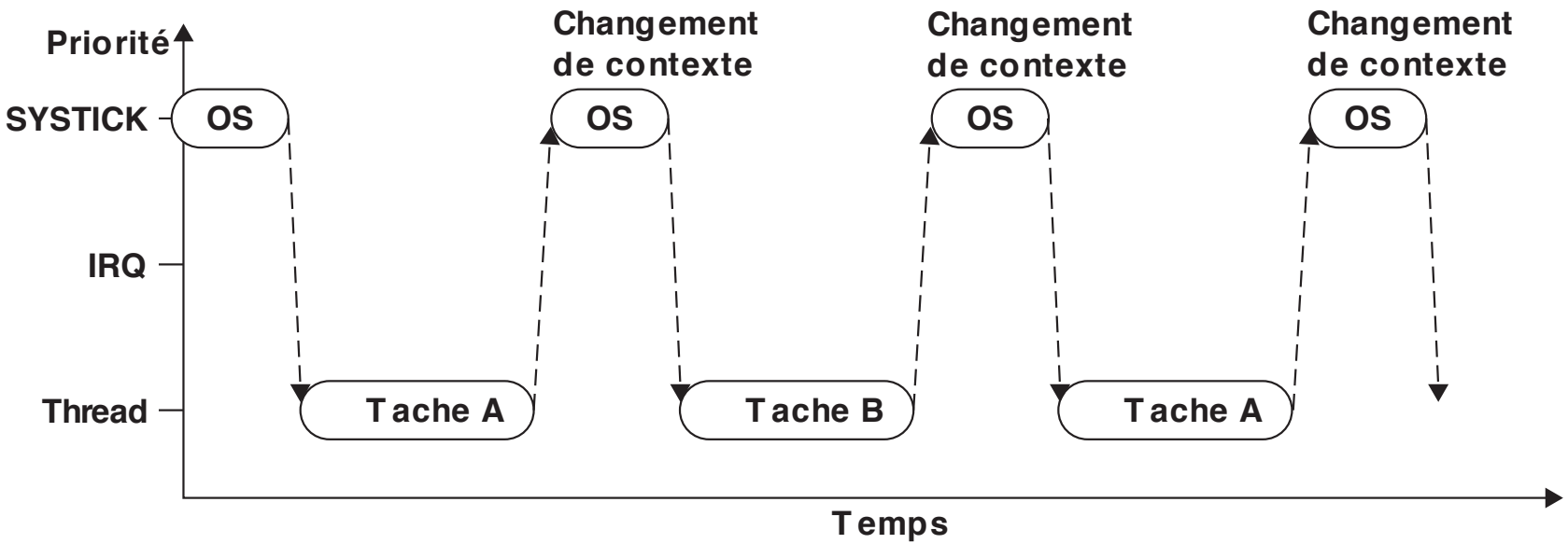
Bits	Nom	Type	Valeur Initiale	Description
31	DEBUGEVT	R/Wc	0	Indique qu'une faute matérielle est déclenché par un événement de debug
30	FORCED	R/Wc	0	Indique qu'une faute matérielle est déclenchée du à une faute mémoire, de bus ou d'usage
29:2	–	–	–	–
1	VECTL	R/Wc	0	Indique une faute matérielle causée par l'échec du chargement de la table des vecteurs d'exception
0	–	–	–	–

Système d'exploitation

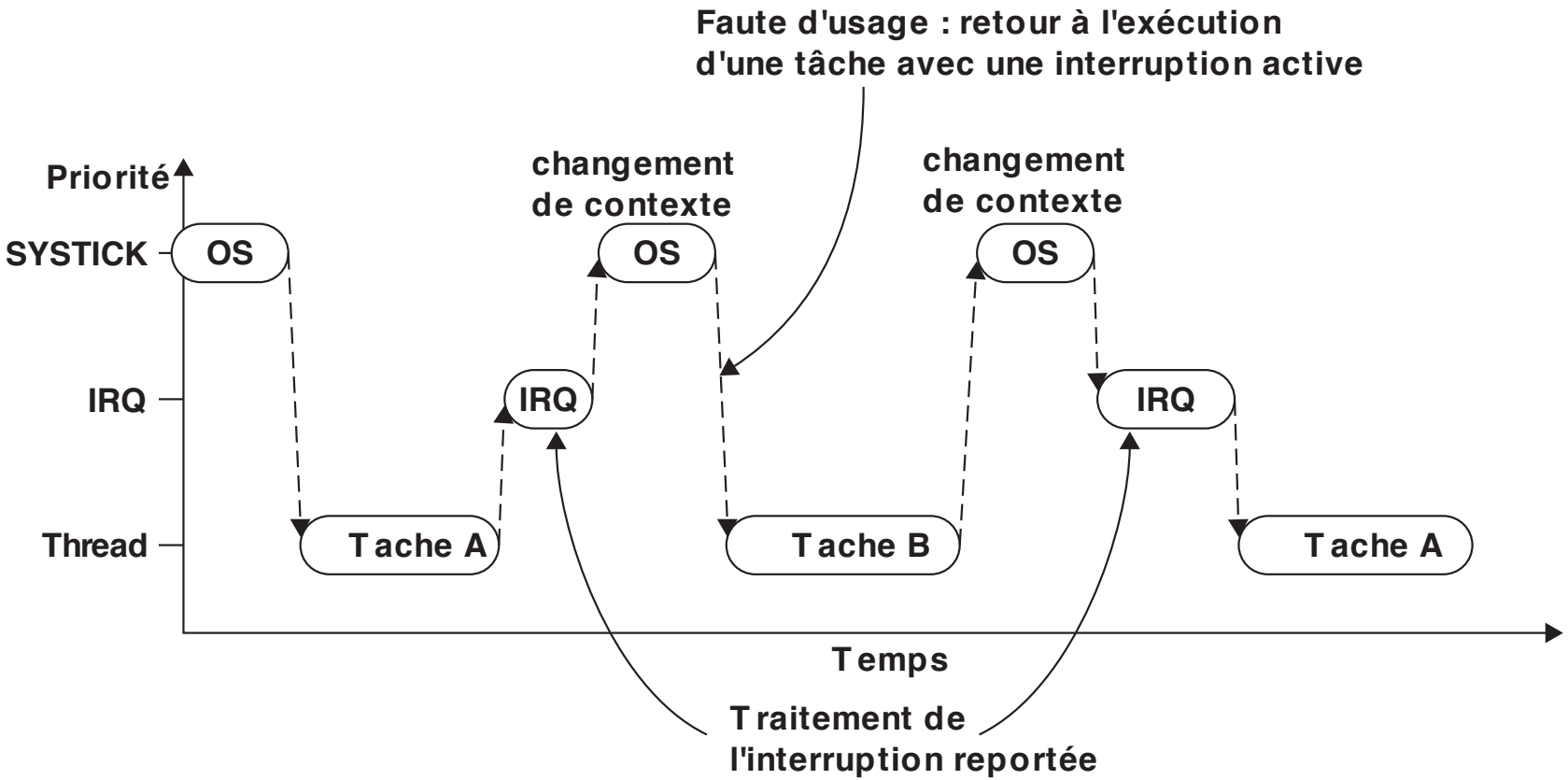
Utilisation de SVC



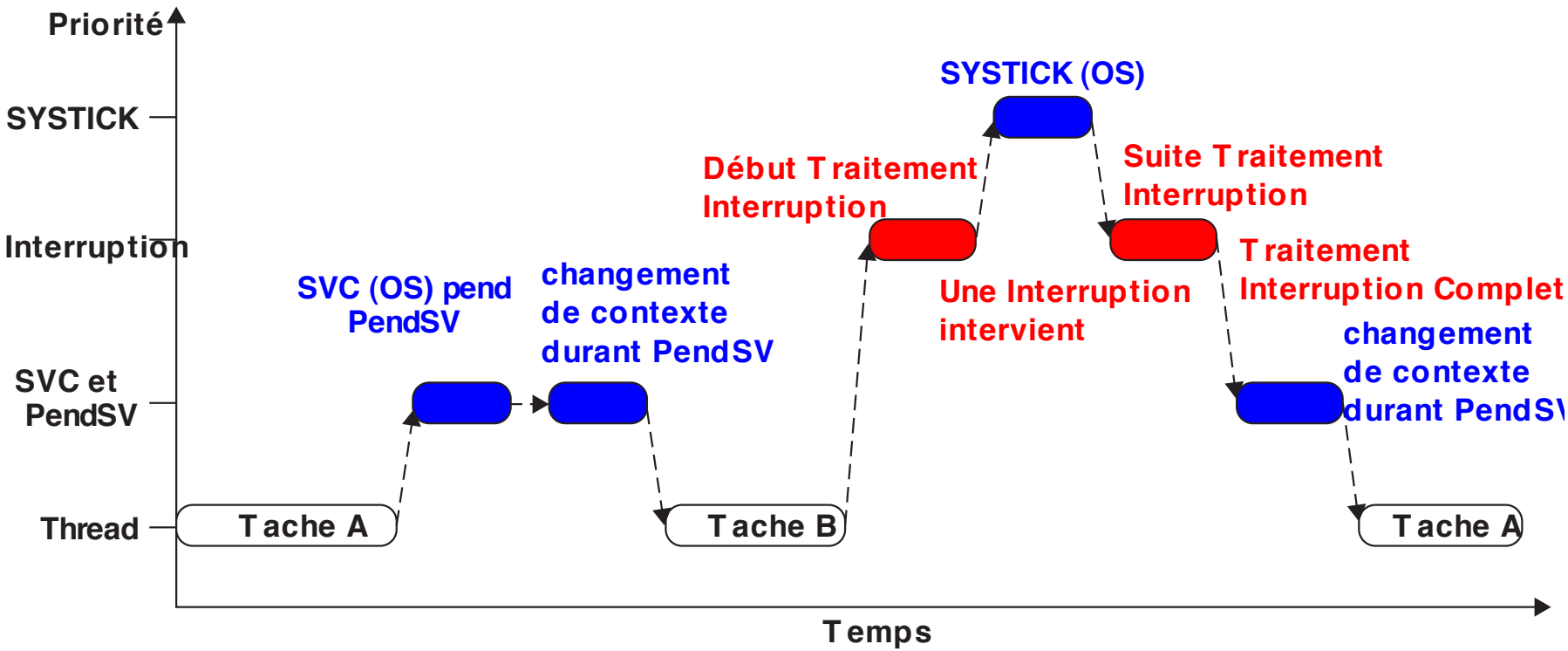
Simple Scénario avc SYSTICK



Problème lors d'une arrivé d'interruption



Utilisation de SVC et PendSV



Le contrôleur d'interruption du STM32

- Le STM32 hérite du contrôleur d'interruption du CortexM3

Le contrôleur d'interruption du STM32

- Le STM32 hérite du contrôleur d'interruption du CortexM3
- Le NVIC (Nested Vectored Interrupt Controller)

Le contrôleur d'interruption du STM32

- Le STM32 hérite du contrôleur d'interruption du CortexM3
- Le NVIC (Nested Vectored Interrupt Controller)
- Il contrôle les interruptions via des registres de configurations

Le contrôleur d'interruption du STM32

- Le STM32 hérite du contrôleur d'interruption du CortexM3
- Le NVIC (Nested Vectored Interrupt Controller)
- Il contrôle les interruptions via des registres de configurations
  - Autorisation ou Interdiction d'une source d'interruption

Le contrôleur d'interruption du STM32

- Le STM32 hérite du contrôleur d'interruption du CortexM3
- Le NVIC (Nested Vectored Interrupt Controller)
- Il contrôle les interruptions via des registres de configurations
  - Autorisation ou Interdiction d'une source d'interruption
  - Réglage du niveau de priorité des sources d'interruption

Le contrôleur d'interruption du STM32

- Le STM32 hérite du contrôleur d'interruption du CortexM3
- Le NVIC (Nested Vectored Interrupt Controller)
- Il contrôle les interruptions via des registres de configurations
  - Autorisation ou Interdiction d'une source d'interruption
  - Réglage du niveau de priorité des sources d'interruption
- Il contrôle aussi un timer système nommé SYSTICK

Fonctionnement de base du NVIC

- Registres de configuration pour autoriser et interdire les sources d'interruptions

Fonctionnement de base du NVIC

- Registres de configuration pour autoriser et interdire les sources d'interruptions
- Registres de configuration pour positionner ou annuler une demande d'interruption

Fonctionnement de base du NVIC

- Registres de configuration pour autoriser et interdire les sources d'interruptions
- Registres de configuration pour positionner ou annuler une demande d'interruption
- Registres de configuration des niveaux de priorité des sources d'interruptions

Fonctionnement de base du NVIC

- Registres de configuration pour autoriser et interdire les sources d'interruptions
- Registres de configuration pour positionner ou annuler une demande d'interruption
- Registres de configuration des niveaux de priorité des sources d'interruptions
- Registre d'état d'activité des routines de service des interruptions

Registres de configuration d'autorisation d'interruptions : NVIC_ISER0

Adresse	Nom	Type	Valeur Init.	Description
0xE000E100	SETENA	R/W	0	Autorisation source interruption numéro 0 à 31 bit[0] pour l'interruption numéro 0 (Watch-dog) bit[1] pour l'interruption numéro 1 (PVD) ... bit[31] pour l'interruption numéro 31 (I2C1_EV) Ecrire 1 pour autoriser - Ecrire 0 n'a aucun effet

Registres de configuration d'autorisation d'interruptions : NVIC_ISER1

0xE000E104	SETENA	R/W	0	Autorisation source interruption numéro 32 à 63 bit[0] pour l'interruption numéro 32 (I2C1_ER) bit[1] pour l'interruption numéro 33 (I2C2_EV) ... bit[27] pour l'interruption numéro 59 (DMA2_Channel4_5) Ecrire 1 pour autoriser - Ecrire 0 n'a aucun effet

Registres de configuration d'interdiction d'interruptions : NVIC_ICER0

Adresse	Nom	Type	Valeur Init.	Description
0xE000E180	CLRENA	R/W	0	Interdiction source interruption numéro 0 à 31 bit[0] pour l'interruption numéro 0 (Watch-dog) bit[1] pour l'interruption numéro 1 (PVD) ... bit[31] pour l'interruption numéro 31 (I2C1_EV) Ecrire 1 pour autoriser - Ecrire 0 n'a aucun effet

Registres de configuration d'interdiction d'interruptions : NVIC_ICER1

Adresse	Nom	Type	Valeur Init.	Description
0xE000E184	CLRENA	R/W	0	Interdiction source interruption numéro 32 à 63 bit[0] pour l'interruption numéro 32 (I2C1_ER) bit[1] pour l'interruption numéro 33 (I2C2_EV) ... bit[27] pour l'interruption numéro 59 (DMA2_Channel4_5) Ecrire 1 pour autoriser - Ecrire 0 n'a aucun effet

Registre de demande d'interruption : NVIC_ISPR0

Adresse	Nom	Type	Valeur Init.	Description
0xE000E200	SETPEND	R/W	0	Demande d'interruption des numéro 0 à 31 bit[0] pour l'interruption numéro 0 (Watch-dog) ... bit[31] pour l'interruption numéro 31 (I2C1_EV) Ecrire 1 pour autoriser - Ecrire 0 n'a aucun effet La lecture indique les demandes en cours

Registre de demande d'interruption : NVIC_ISPR1

Adresse	Nom	Type	Valeur Init.	Description
0xE000E204	SETPEND	R/W	0	Demande d'interruption des numéro 32 à 63 bit[0] pour l'interruption numéro 32 (I2C1_ER) ... bit[27] pour l'interruption numéro 59 (DMA2_Channel4_5) Ecrire 1 pour autoriser - Ecrire 0 n'a aucun effet La lecture indique les demandes en cours

Registre d'invalidation de demande d'interruption : NVIC_ICPR0

Adresse	Nom	Type	Valeur Init.	Description
0xE000E280	CLRPEND	R/W	0	Invalidation demande d'interruption des numéro 0 à 31 bit[0] pour l'interruption numéro 0 (Watch-dog) ... bit[31] pour l'interruption numéro 31 (I2C1_EV) Ecrire 1 pour autoriser - Ecrire 0 n'a aucun effet La lecture indique les demandes en cours

Registre d'invalidation de demande d'interruption : NVIC_ICPR1

Adresse	Nom	Type	Valeur Init.	Description
0xE000E284	CLRPEND	R/W	0	Invalidation demande d'interruption des numéro 32 à 63 bit[0] pour l'interruption numéro 32 (I2C1_ER) ... bit[27] pour l'interruption numéro 59 (DMA2_Channel4_5) Ecrire 1 pour autoriser - Ecrire 0 n'a aucun effet La lecture indique les demandes en cours

Registre de priorité des interruptions : NVIC_IPR0

Adresse	Nom	Type	Valeur Init.	Description
0xE000E400	IP[0]	R/W	0(8bits)	Niveau de priorité de l'interruption numéro 0 (Watchdog)
	IP[1]	R/W	0(8bits)	Niveau de priorité de l'interruption numéro 1 (PVD)
	IP[2]	R/W	0(8bits)	Niveau de priorité de l'interruption numéro 2 (TAMPER)
	IP[3]	R/W	0(8bits)	Niveau de priorité de l'interruption numéro 3 (RTC)

Registre de priorité des interruptions : NVIC_IPR0

Adresse	Nom	Type	Valeur Init.	Description
0xE000E400	IP[0]	R/W	0(8bits)	Niveau de priorité de l'interruption numéro 0 (Watchdog)
	IP[1]	R/W	0(8bits)	Niveau de priorité de l'interruption numéro 1 (PVD)
	IP[2]	R/W	0(8bits)	Niveau de priorité de l'interruption numéro 2 (TAMPER)
	IP[3]	R/W	0(8bits)	Niveau de priorité de l'interruption numéro 3 (RTC)

- NVIC_IPR0, NVIC_IPR1 à NVIC_IPR14

Registre de priorité des interruptions : NVIC_IPR8

Adresse	Nom	Type	Valeur Init.	Description
0xE000E420	IP[32]	R/W	0(8bits)	Niveau de priorité de l'interruption numéro 32 (I2C1_ER)
	IP[33]	R/W	0(8bits)	Niveau de priorité de l'interruption numéro 33 (I2C1_EV)
	IP[34]	R/W	0(8bits)	Niveau de priorité de l'interruption numéro 34 (I2C1_ER)
	IP[35]	R/W	0(8bits)	Niveau de priorité de l'interruption numéro 35 (SPI1)

Registre de priorité des interruptions : NVIC_IPR14

Adresse	Nom	Type	Valeur Init.	Description
0xE000E438	IP[56]	R/W	0(8bits)	Niveau de priorité de l'interruption numéro 56 (DMA2_Channel1)
	IP[57]	R/W	0(8bits)	Niveau de priorité de l'interruption numéro 57 (DMA2_Channel2)
	IP[58]	R/W	0(8bits)	Niveau de priorité de l'interruption numéro 58 (DMA2_Channel3)
	IP[59]	R/W	0(8bits)	Niveau de priorité de l'interruption numéro 59 (DMA2_Channel4_5)

Registre d'activité des interruptions : NVIC_IABR0

Adresse	Nom	Type	Valeur Init.	Description
0xE000E300	ACTIVE	R	0	Etat d'activité des interruptions 0 à 31 bit[0] pour l'interruption numéro 0 (Watch-dog) ... bit[31] pour l'interruption numéro 31 (I2C1_EV)

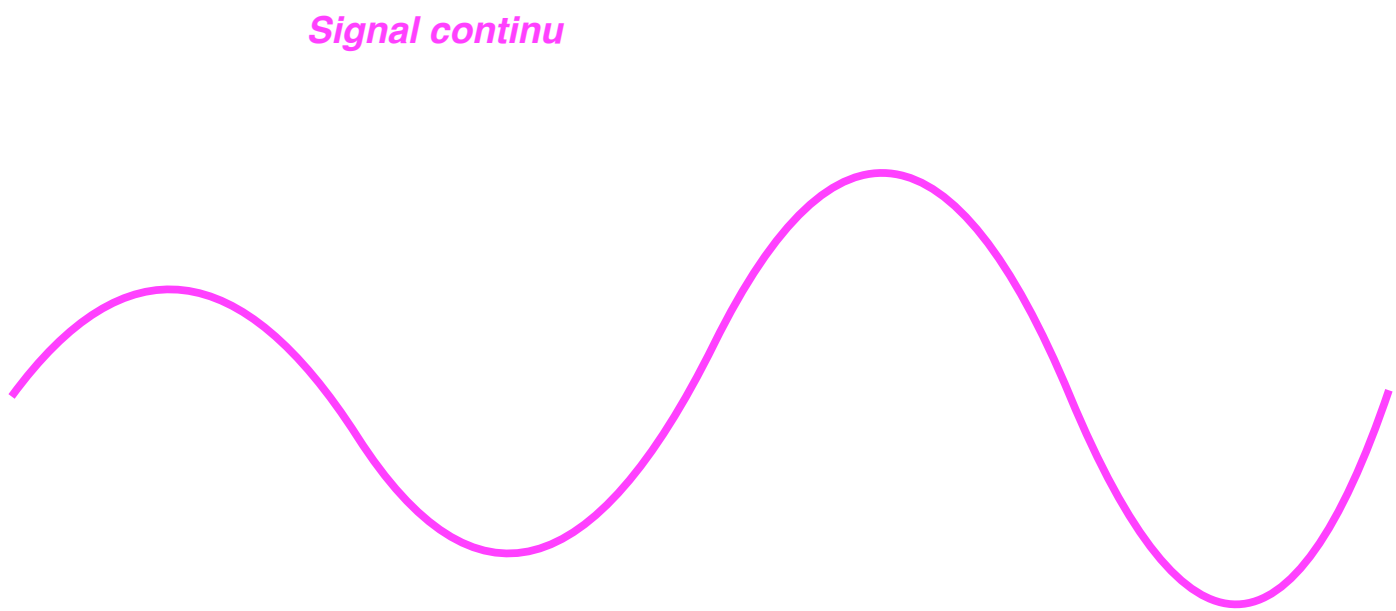
Registre d'activité des interruptions : NVIC_IABR1

Adresse	Nom	Type	Valeur Init.	Description
0xE000E304	ACTIVE	R	0	Etat d'activité des interruptions 32 à 59 bit[0] pour l'interruption numéro 32 (I2C1_ER) ... bit[27] pour l'interruption numéro 59 (DMA2_Channel4_5)

Sommaire

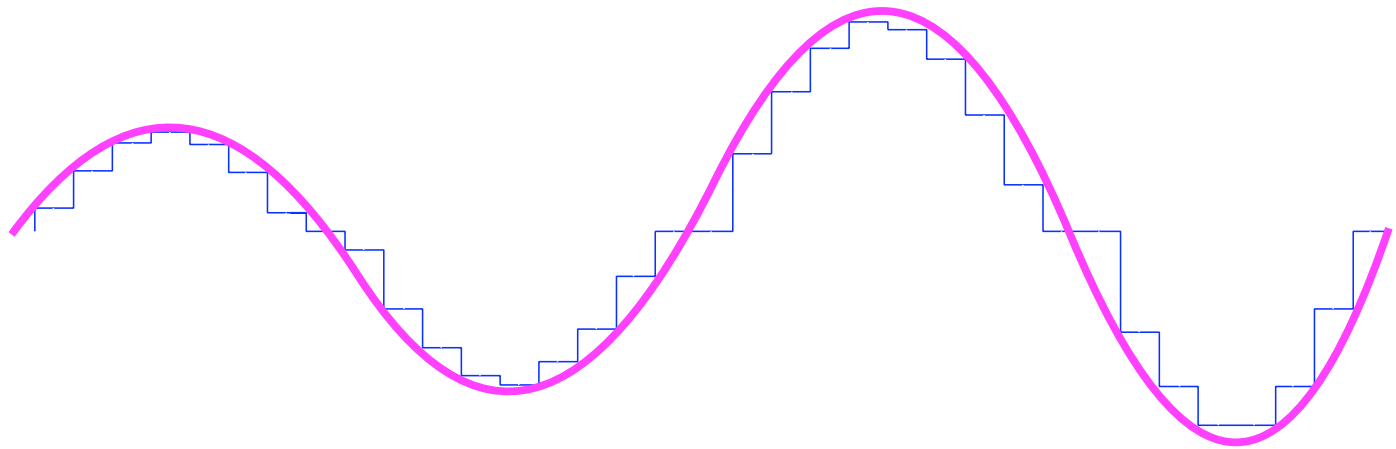
Convetisseur Analogique Numérique ou ADC (Analog Digital Conversion)

ADC : Définitions

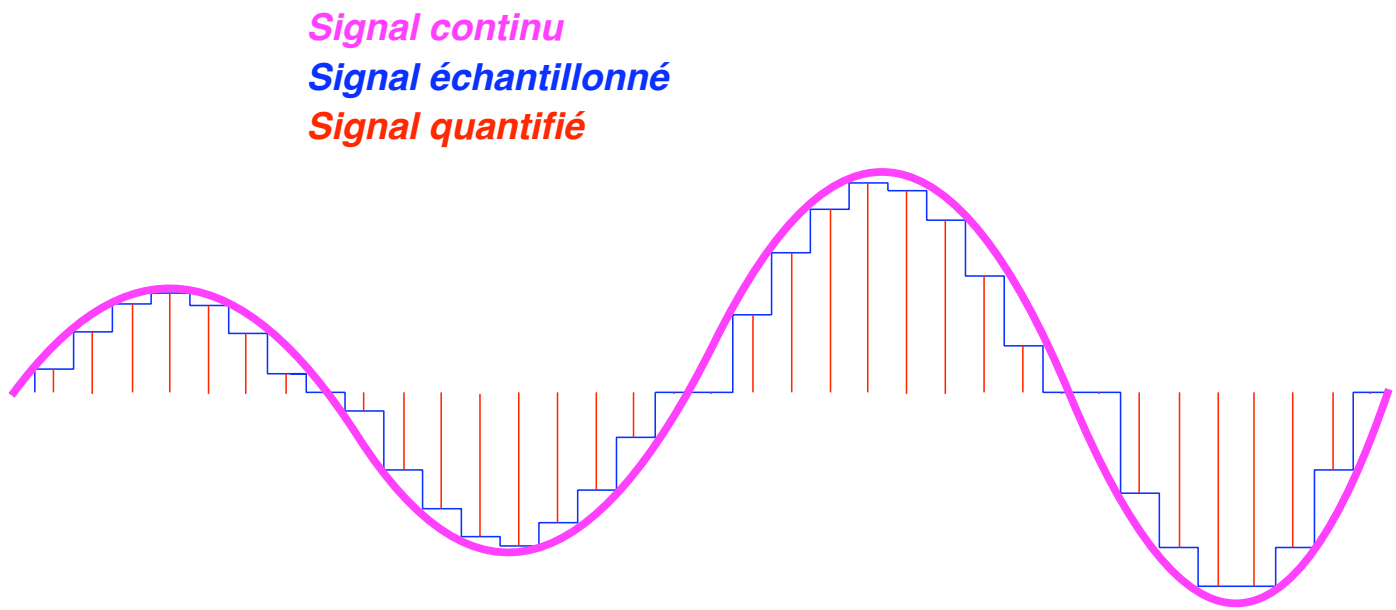


ADC : Définitions

Signal continu
Signal échantillonné

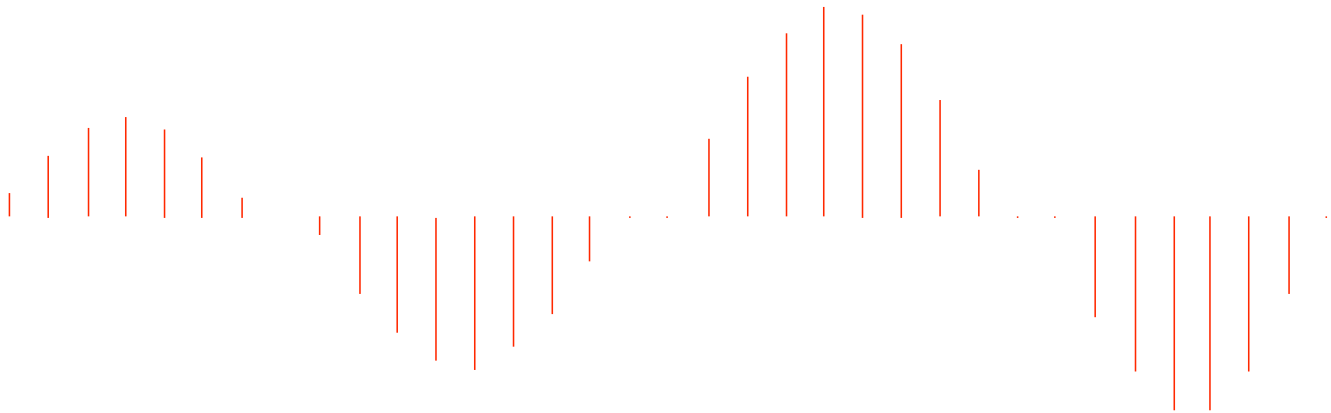


ADC : Définitions



ADC : Définitions

Signal quantifié



ADC : Caractéristiques

- Résolution : Amplitude de la plus petite variation. Correspond au LSB (Least Significant Bit)

ADC : Caractéristiques

- Résolution : Amplitude de la plus petite variation. Correspond au LSB (Least Significant Bit)
- Temps de conversion : Temps de stabilisation de la donnée en sortie

ADC : Caractéristiques

- Résolution : Amplitude de la plus petite variation. Correspond au LSB (Least Significant Bit)
- Temps de conversion : Temps de stabilisation de la donnée en sortie
- Erreur de Quantification : Incertitude du à la conversion

ADC : Caractéristiques

- Résolution : Amplitude de la plus petite variation. Correspond au LSB (Least Significant Bit)
- Temps de conversion : Temps de stabilisation de la donnée en sortie
- Erreur de Quantification : Incertitude du à la conversion
- Pleine Echelle : Etendue de la grandeur Analogique d'entrée

ADC : Types

- Il exite différents type de conversion

ADC : Types

- Il exite différents type de conversion
- La conversion à rampe

ADC : Types

- Il exite différents type de conversion
- La conversion à rampe
- La conversion à double rampe

ADC : Types

- Il exite différents type de conversion
- La conversion à rampe
- La conversion à double rampe
- La conversion à approximation successive

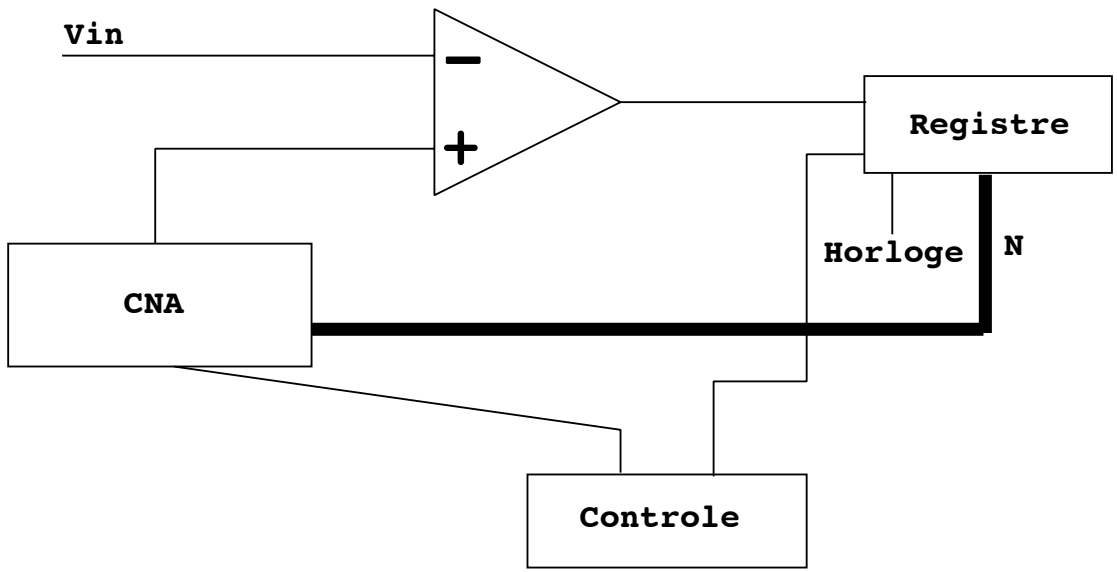
ADC : Types

- Il exite différents type de conversion
- La conversion à rampe
- La conversion à double rampe
- La conversion à approximation successive
- La conversion Flash

ADC : Types

- Il exite différents type de conversion
- La conversion à rampe
- La conversion à double rampe
- La conversion à approximation successive
- La conversion Flash
- La conversion Sigma-Delta

La conversion à approximations successives



La conversion à approximations successives

- Détermination des valeurs de bits de N les unes après les autres en commençant par le bit de poids fort

La conversion à approximations successives

- Détermination des valeurs de bits de N les unes après les autres en commençant par le bit de poids fort
- On fixe le bit de poids fort à 1 et les autres à 0. Conversion NA du registre et comparaison à V_{in}

La conversion à approximations successives

- Détermination des valeurs de bits de N les unes après les autres en commençant par le bit de poids fort
- On fixe le bit de poids fort à 1 et les autres à 0. Conversion NA du registre et comparaison à V_{in}
- Si V_{in} est plus grand alors le bit reste à 1 sinon il passe à 0.

La conversion à approximations successives

- Détermination des valeurs de bits de N les unes après les autres en commençant par le bit de poids fort
- On fixe le bit de poids fort à 1 et les autres à 0. Conversion NA du registre et comparaison à V_{in}
- Si V_{in} est plus grand alors le bit reste à 1 sinon il passe à 0.
- On garde la valeur du bit de poids fort et on passe au bit suivant

La conversion à approximations successives

- Détermination des valeurs de bits de N les unes après les autres en commençant par le bit de poids fort
- On fixe le bit de poids fort à 1 et les autres à 0. Conversion NA du registre et comparaison à V_{in}
- Si V_{in} est plus grand alors le bit reste à 1 sinon il passe à 0.
- On garde la valeur du bit de poids fort et on passe au bit suivant
- On refait le même traitement que précédemment pour ce bit et ainsi de suite jusqu'au bit de poids faible.

La conversion à approximations successives

- Exemple : Convertisseur 8 bits, V_{ref}=10 V

La conversion à approximations successives

- Exemple : Convertisseur 8 bits, V_{ref}=10 V
- Tension à convertir 6,92 V

La conversion à approximations successives

- Exemple : Convertisseur 8 bits, V_{ref}=10 V
- Tension à convertir 6,92 V
- $10000000 = 5V < 6,92 \rightarrow B_7 = 1$

La conversion à approximations successives

- Exemple : Convertisseur 8 bits, V_{ref}=10 V
- Tension à convertir 6,92 V
- $10000000 = 5V < 6,92 \rightarrow B_7 = 1$
- $11000000 = 7,5V > 6,92 \rightarrow B_6 = 0$

La conversion à approximations successives

- Exemple : Convertisseur 8 bits, V_{ref}=10 V
- Tension à convertir 6,92 V
- $10000000 = 5V < 6,92 \rightarrow B_7 = 1$
- $11000000 = 7,5V > 6,92 \rightarrow B_6 = 0$
- $10100000 = 6,25V < 6,92 \rightarrow B_5 = 1$

La conversion à approximations successives

- Exemple : Convertisseur 8 bits, V_{ref}=10 V
- Tension à convertir 6,92 V
- $10000000 = 5V < 6,92 \rightarrow B_7 = 1$
- $11000000 = 7,5V > 6,92 \rightarrow B_6 = 0$
- $10100000 = 6,25V < 6,92 \rightarrow B_5 = 1$
- $10110000 = 6,675V < 6,92 \rightarrow B_4 = 1$

La conversion à approximations successives

- Exemple : Convertisseur 8 bits, V_{ref}=10 V
- Tension à convertir 6,92 V
- $10000000 = 5V < 6,92 \rightarrow B_7 = 1$
- $11000000 = 7,5V > 6,92 \rightarrow B_6 = 0$
- $10100000 = 6,25V < 6,92 \rightarrow B_5 = 1$
- $10110000 = 6,675V < 6,92 \rightarrow B_4 = 1$
- $10111000 = 7,1875V > 6,92 \rightarrow B_3 = 0$

La conversion à approximations successives

- Exemple : Convertisseur 8 bits, V_{ref}=10 V
- Tension à convertir 6,92 V
- $10000000 = 5V < 6,92 \rightarrow B_7 = 1$
- $11000000 = 7,5V > 6,92 \rightarrow B_6 = 0$
- $10100000 = 6,25V < 6,92 \rightarrow B_5 = 1$
- $10110000 = 6,675V < 6,92 \rightarrow B_4 = 1$
- $10111000 = 7,1875V > 6,92 \rightarrow B_3 = 0$
- $10110100 = 7,03125V > 6,92 \rightarrow B_2 = 0$

La conversion à approximations successives

- Exemple : Convertisseur 8 bits, V_{ref}=10 V
- Tension à convertir 6,92 V
- $10000000 = 5V < 6,92 \rightarrow B_7 = 1$
- $11000000 = 7,5V > 6,92 \rightarrow B_6 = 0$
- $10100000 = 6,25V < 6,92 \rightarrow B_5 = 1$
- $10110000 = 6,675V < 6,92 \rightarrow B_4 = 1$
- $10111000 = 7,1875V > 6,92 \rightarrow B_3 = 0$
- $10110100 = 7,03125V > 6,92 \rightarrow B_2 = 0$
- $10110010 = 6,95312V > 6,92 \rightarrow B_1 = 0$

La conversion à approximations successives

- Exemple : Convertisseur 8 bits, V_{ref}=10 V
- Tension à convertir 6,92 V
- $10000000 = 5V < 6,92 \rightarrow B_7 = 1$
- $11000000 = 7,5V > 6,92 \rightarrow B_6 = 0$
- $10100000 = 6,25V < 6,92 \rightarrow B_5 = 1$
- $10110000 = 6,675V < 6,92 \rightarrow B_4 = 1$
- $10111000 = 7,1875V > 6,92 \rightarrow B_3 = 0$
- $10110100 = 7,03125V > 6,92 \rightarrow B_2 = 0$
- $10110010 = 6,95312V > 6,92 \rightarrow B_1 = 0$
- $10110001 = 6,91406V < 6,92 \rightarrow B_0 = 1$

La conversion à approximations successives

- Exemple : Convertisseur 8 bits, V_{ref}=10 V
- Tension à convertir 6,92 V
- $10000000 = 5V < 6,92 \rightarrow B_7 = 1$
- $11000000 = 7,5V > 6,92 \rightarrow B_6 = 0$
- $10100000 = 6,25V < 6,92 \rightarrow B_5 = 1$
- $10110000 = 6,675V < 6,92 \rightarrow B_4 = 1$
- $10111000 = 7,1875V > 6,92 \rightarrow B_3 = 0$
- $10110100 = 7,03125V > 6,92 \rightarrow B_2 = 0$
- $10110010 = 6,95312V > 6,92 \rightarrow B_1 = 0$
- $10110001 = 6,91406V < 6,92 \rightarrow B_0 = 1$
- Valeur Numérique :10110001

ADC STM32 : Caractéristiques Principales

- Résolution de 12 bits

ADC STM32 : Caractéristiques Principales

- Résolution de 12 bits
- Interruption de fin de conversion

ADC STM32 : Caractéristiques Principales

- Résolution de 12 bits
- Interruption de fin de conversion
- Mode simple conversion ou conversion continue

ADC STM32 : Caractéristiques Principales

- Résolution de 12 bits
- Interruption de fin de conversion
- Mode simple conversion ou conversion continue
- Mode de balayage multi-canal

ADC STM32 : Caractéristiques Principales

- Résolution de 12 bits
- Interruption de fin de conversion
- Mode simple conversion ou conversion continue
- Mode de balayage multi-canal
- Auto-Calibration

ADC STM32 : Caractéristiques Principales

- Résolution de 12 bits
- Interruption de fin de conversion
- Mode simple conversion ou conversion continue
- Mode de balayage multi-canal
- Auto-Calibration
- Alignement des données sur 16 bits (droite ou gauche)

ADC STM32 : Caractéristiques Principales

- Résolution de 12 bits
- Interruption de fin de conversion
- Mode simple conversion ou conversion continue
- Mode de balayage multi-canal
- Auto-Calibration
- Alignement des données sur 16 bits (droite ou gauche)
- Temps de conversion programmable canal par canal

ADC STM32 : Caractéristiques Principales

- Résolution de 12 bits
- Interruption de fin de conversion
- Mode simple conversion ou conversion continue
- Mode de balayage multi-canal
- Auto-Calibration
- Alignement des données sur 16 bits (droite ou gauche)
- Temps de conversion programmable canal par canal
- Déclenchement par signal externe

ADC STM32 : Caractéristiques Principales

- Résolution de 12 bits
- Interruption de fin de conversion
- Mode simple conversion ou conversion continue
- Mode de balayage multi-canal
- Auto-Calibration
- Alignement des données sur 16 bits (droite ou gauche)
- Temps de conversion programmable canal par canal
- Déclenchement par signal externe
- Mode de conversion discontinue lors de balayage

ADC STM32 : Caractéristiques Principales

- Résolution de 12 bits
- Interruption de fin de conversion
- Mode simple conversion ou conversion continue
- Mode de balayage multi-canal
- Auto-Calibration
- Alignement des données sur 16 bits (droite ou gauche)
- Temps de conversion programmable canal par canal
- Déclenchement par signal externe
- Mode de conversion discontinue lors de balayage
- Chainage de convertisseur (si le circuit en possède au moins 2)

ADC STM32 : Caractéristiques Principales

- Résolution de 12 bits
- Interruption de fin de conversion
- Mode simple conversion ou conversion continue
- Mode de balayage multi-canal
- Auto-Calibration
- Alignement des données sur 16 bits (droite ou gauche)
- Temps de conversion programmable canal par canal
- Déclenchement par signal externe
- Mode de conversion discontinue lors de balayage
- Chainage de convertisseur (si le circuit en possède au moins 2)
- Tension requise entre 2,4 V et 3,6V

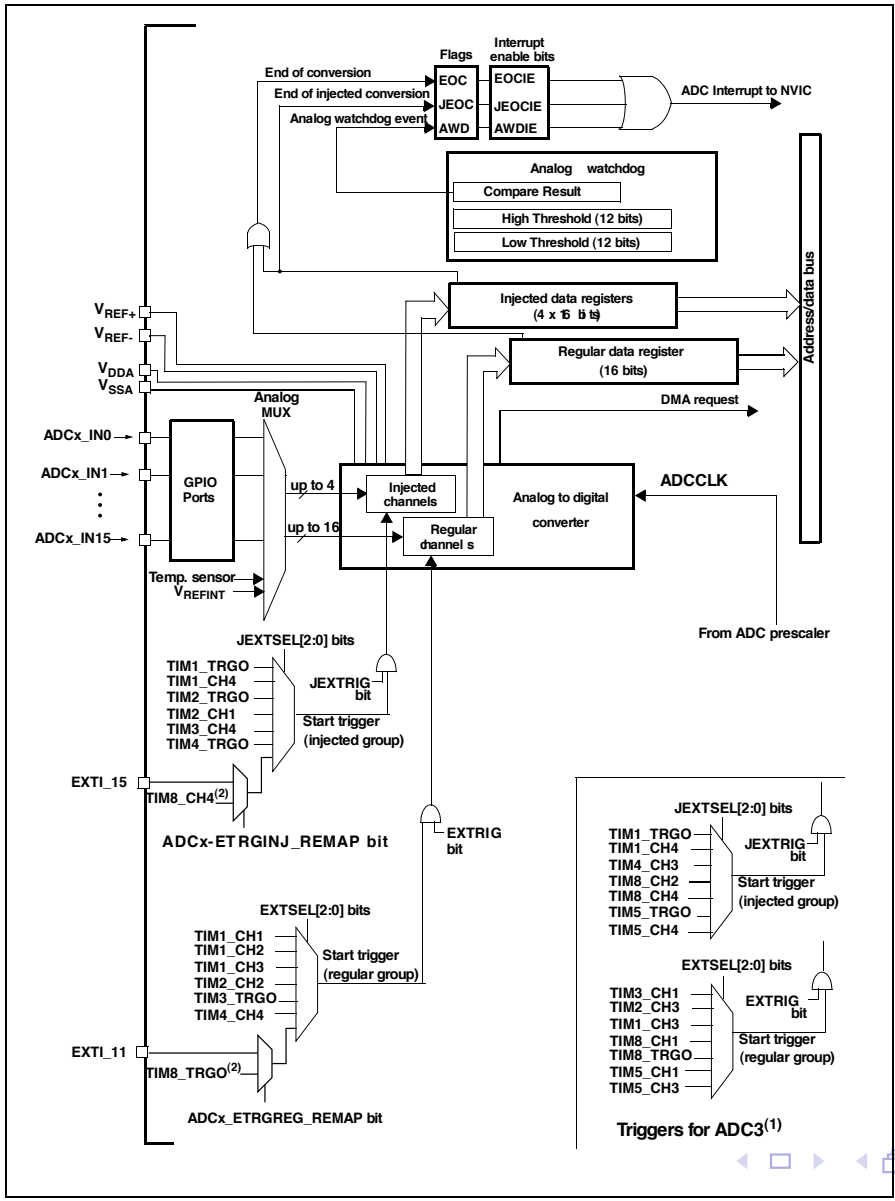
ADC STM32 : Caractéristiques Principales

- Résolution de 12 bits
- Interruption de fin de conversion
- Mode simple conversion ou conversion continue
- Mode de balayage multi-canal
- Auto-Calibration
- Alignement des données sur 16 bits (droite ou gauche)
- Temps de conversion programmable canal par canal
- Déclenchement par signal externe
- Mode de conversion discontinue lors de balayage
- Chainage de convertisseur (si le circuit en possède au moins 2)
- Tension requise entre 2,4 V et 3,6V
- Plage de conversion entre $V_{ref-} \leqslant V_{in} \leqslant V_{ref+}$

ADC STM32 : Caractéristiques Principales

- Résolution de 12 bits
- Interruption de fin de conversion
- Mode simple conversion ou conversion continue
- Mode de balayage multi-canal
- Auto-Calibration
- Alignement des données sur 16 bits (droite ou gauche)
- Temps de conversion programmable canal par canal
- Déclenchement par signal externe
- Mode de conversion discontinue lors de balayage
- Chainage de convertisseur (si le circuit en possède au moins 2)
- Tension requise entre 2,4 V et 3,6V
- Plage de conversion entre $V_{ref-} \leqslant V_{in} \leqslant V_{ref+}$
- Requête DMA pour la conversion régulière

ADC STM32 : Schéma



ADC STM32 : Description Fonctionnelle

Mise sous tension

ADC STM32 : Description Fonctionnelle

Mise sous tension

- Mise sous tension via la mise à 1 du bit ADON (registre CR2 de l'ADC)

ADC STM32 : Description Fonctionnelle

Mise sous tension

- Mise sous tension via la mise à 1 du bit ADON (registre CR2 de l'ADC)
- La première mise à 1 du bit ADON met sous tension l'ADC

ADC STM32 : Description Fonctionnelle

Mise sous tension

- Mise sous tension via la mise à 1 du bit ADON (registre CR2 de l'ADC)
- La première mise à 1 du bit ADON met sous tension l'ADC
- La seconde mise à 1 du bit ADON démarre la conversion

ADC STM32 : Description Fonctionnelle

Mise sous tension

- Mise sous tension via la mise à 1 du bit ADON (registre CR2 de l'ADC)
- La première mise à 1 du bit ADON met sous tension l'ADC
- La seconde mise à 1 du bit ADON démarre la conversion
- La mise hors tension et l'arrêt de la conversion s'effectue en mettant à 0 le bit ADON

ADC STM32 : Description Fonctionnelle

Mise sous tension

- Mise sous tension via la mise à 1 du bit ADON (registre CR2 de l'ADC)
- La première mise à 1 du bit ADON met sous tension l'ADC
- La seconde mise à 1 du bit ADON démarre la conversion
- La mise hors tension et l'arrêt de la conversion s'effectue en mettant à 0 le bit ADON
- En mode hors tension l'ADC ne consomme que quelques μA

ADC STM32 : Description Fonctionnelle

Mise sous tension

- Mise sous tension via la mise à 1 du bit ADON (registre CR2 de l'ADC)
- La première mise à 1 du bit ADON met sous tension l'ADC
- La seconde mise à 1 du bit ADON démarre la conversion
- La mise hors tension et l'arrêt de la conversion s'effectue en mettant à 0 le bit ADON
- En mode hors tension l'ADC ne consomme que quelques μA

Horloge

ADC STM32 : Description Fonctionnelle

Mise sous tension

- Mise sous tension via la mise à 1 du bit ADON (registre CR2 de l'ADC)
- La première mise à 1 du bit ADON met sous tension l'ADC
- La seconde mise à 1 du bit ADON démarre la conversion
- La mise hors tension et l'arrêt de la conversion s'effectue en mettant à 0 le bit ADON
- En mode hors tension l'ADC ne consomme que quelques μA

Horloge

- l'horloge ADCCLK du convertisseur est synchrone avec celle du bus APB2, l'horloge PCLK2

ADC STM32 : Description Fonctionnelle

Mise sous tension

- Mise sous tension via la mise à 1 du bit ADON (registre CR2 de l'ADC)
- La première mise à 1 du bit ADON met sous tension l'ADC
- La seconde mise à 1 du bit ADON démarre la conversion
- La mise hors tension et l'arrêt de la conversion s'effectue en mettant à 0 le bit ADON
- En mode hors tension l'ADC ne consomme que quelques μA

Horloge

- l'horloge ADCCLK du convertisseur est synchrone avec celle du bus APB2, l'horloge PCLK2
- Il est possible de programmer un prédiviseur, par rapport à PCLK2, pour l'horloge ADCCLK dans le contrôleur RCC

ADC STM32 : Description Fonctionnelle

Sélection des Canaux

ADC STM32 : Description Fonctionnelle

Sélection des Canaux

- Il y a 16 canaux de conversions multiplexés

ADC STM32 : Description Fonctionnelle

Sélection des Canaux

- Il y a 16 canaux de conversions multiplexés
- Organisation possible en deux groupes : réguliers et injectés

ADC STM32 : Description Fonctionnelle

Sélection des Canaux

- Il y a 16 canaux de conversions multiplexés
- Organisation possible en deux groupes : réguliers et injectés
- Un groupe est une séquence de conversions (16 au plus pour réguliers, 4 au plus pour injectés)

ADC STM32 : Description Fonctionnelle

Sélection des Canaux

- Il y a 16 canaux de conversions multiplexés
- Organisation possible en deux groupes : réguliers et injectés
- Un groupe est une séquence de conversions (16 au plus pour réguliers, 4 au plus pour injectés)
- Une séquence peut-être réalisée dans un ordre aléatoire

ADC STM32 : Description Fonctionnelle

Sélection des Canaux

- Il y a 16 canaux de conversions multiplexés
- Organisation possible en deux groupes : réguliers et injectés
- Un groupe est une séquence de conversions (16 au plus pour réguliers, 4 au plus pour injectés)
- Une séquence peut-être réalisée dans un ordre aléatoire
- Le nombre de conversions, l'ordre de conversion sont fixés via les registres SQRx et JSQR

ADC STM32 : Description Fonctionnelle

Sélection des Canaux

- Il y a 16 canaux de conversions multiplexés
- Organisation possible en deux groupes : réguliers et injectés
- Un groupe est une séquence de conversions (16 au plus pour réguliers, 4 au plus pour injectés)
- Une séquence peut-être réalisée dans un ordre aléatoire
- Le nombre de conversions, l'ordre de conversion sont fixés via les registres SQRx et JSQR

Capteur de Température/Tension

ADC STM32 : Description Fonctionnelle

Sélection des Canaux

- Il y a 16 canaux de conversions multiplexés
- Organisation possible en deux groupes : réguliers et injectés
- Un groupe est une séquence de conversions (16 au plus pour réguliers, 4 au plus pour injectés)
- Une séquence peut-être réalisée dans un ordre aléatoire
- Le nombre de conversions, l'ordre de conversion sont fixés via les registres SQRx et JSQR

Capteur de Température/Tension

- Un capteur de température interne est connecté au canal 16

ADC STM32 : Description Fonctionnelle

Sélection des Canaux

- Il y a 16 canaux de conversions multiplexés
- Organisation possible en deux groupes : réguliers et injectés
- Un groupe est une séquence de conversions (16 au plus pour réguliers, 4 au plus pour injectés)
- Une séquence peut-être réalisée dans un ordre aléatoire
- Le nombre de conversions, l'ordre de conversion sont fixés via les registres SQRx et JSQR

Capteur de Température/Tension

- Un capteur de température interne est connecté au canal 16
- Une tension V_{refint} interne est connectée au canal 17

ADC STM32 : Description Fonctionnelle

Sélection des Canaux

- Il y a 16 canaux de conversions multiplexés
- Organisation possible en deux groupes : réguliers et injectés
- Un groupe est une séquence de conversions (16 au plus pour réguliers, 4 au plus pour injectés)
- Une séquence peut-être réalisée dans un ordre aléatoire
- Le nombre de conversions, l'ordre de conversion sont fixés via les registres SQRx et JSQR

Capteur de Température/Tension

- Un capteur de température interne est connecté au canal 16
- Une tension V_{refint} interne est connectée au canal 17
- Accessible avec l'ADC1

ADC STM32 : Description Fonctionnelle

Mode Simple Conversion

ADC STM32 : Description Fonctionnelle

Mode Simple Conversion

- Bit CONT à 0 (registre CR2)

ADC STM32 : Description Fonctionnelle

Mode Simple Conversion

- Bit CONT à 0 (registre CR2)
- Démarrage par bit ADON uniquement pour groupe réguliers

ADC STM32 : Description Fonctionnelle

Mode Simple Conversion

- Bit CONT à 0 (registre CR2)
- Démarrage par bit ADON uniquement pour groupe réguliers
- Une fois le canal converti

ADC STM32 : Description Fonctionnelle

Mode Simple Conversion

- Bit CONT à 0 (registre CR2)
- Démarrage par bit ADON uniquement pour groupe réguliers
- Une fois le canal converti
  - La donnée convertie est stockée dans le registre 16bits DR

ADC STM32 : Description Fonctionnelle

Mode Simple Conversion

- Bit CONT à 0 (registre CR2)
- Démarrage par bit ADON uniquement pour groupe réguliers
- Une fois le canal converti
  - La donnée convertie est stockée dans le registre 16bits DR
  - Le drapeau EOC est positionnée

ADC STM32 : Description Fonctionnelle

Mode Simple Conversion

- Bit CONT à 0 (registre CR2)
- Démarrage par bit ADON uniquement pour groupe réguliers
- Une fois le canal converti
  - La donnée convertie est stockée dans le registre 16bits DR
  - Le drapeau EOC est positionnée
  - Une interruption est générée si le bit EOCIE est à 1

ADC STM32 : Description Fonctionnelle

Mode Simple Conversion

- Bit CONT à 0 (registre CR2)
- Démarrage par bit ADON uniquement pour groupe réguliers
- Une fois le canal converti
  - La donnée convertie est stockée dans le registre 16bits DR
  - Le drapeau EOC est positionnée
  - Une interruption est générée si le bit EOCIE est à 1

Mode Conversion Continue

ADC STM32 : Description Fonctionnelle

Mode Simple Conversion

- Bit CONT à 0 (registre CR2)
- Démarrage par bit ADON uniquement pour groupe réguliers
- Une fois le canal converti
  - La donnée convertie est stockée dans le registre 16bits DR
  - Le drapeau EOC est positionnée
  - Une interruption est générée si le bit EOCIE est à 1

Mode Conversion Continue

- Bit CONT à 1 (registre CR2)

ADC STM32 : Description Fonctionnelle

Mode Simple Conversion

- Bit CONT à 0 (registre CR2)
- Démarrage par bit ADON uniquement pour groupe réguliers
- Une fois le canal converti
  - La donnée convertie est stockée dans le registre 16bits DR
  - Le drapeau EOC est positionnée
  - Une interruption est générée si le bit EOCIE est à 1

Mode Conversion Continue

- Bit CONT à 1 (registre CR2)
- Démarrage par bit ADON uniquement pour groupe réguliers

ADC STM32 : Description Fonctionnelle

Mode Simple Conversion

- Bit CONT à 0 (registre CR2)
- Démarrage par bit ADON uniquement pour groupe réguliers
- Une fois le canal converti
  - La donnée convertie est stockée dans le registre 16bits DR
  - Le drapeau EOC est positionnée
  - Une interruption est générée si le bit EOCIE est à 1

Mode Conversion Continue

- Bit CONT à 1 (registre CR2)
- Démarrage par bit ADON uniquement pour groupe réguliers
- Une fois le canal converti

ADC STM32 : Description Fonctionnelle

Mode Simple Conversion

- Bit CONT à 0 (registre CR2)
- Démarrage par bit ADON uniquement pour groupe réguliers
- Une fois le canal converti
  - La donnée convertie est stockée dans le registre 16bits DR
  - Le drapeau EOC est positionnée
  - Une interruption est générée si le bit EOCIE est à 1

Mode Conversion Continue

- Bit CONT à 1 (registre CR2)
- Démarrage par bit ADON uniquement pour groupe réguliers
- Une fois le canal converti
  - La donnée convertie est stockée dans le registre 16bits DR

ADC STM32 : Description Fonctionnelle

Mode Simple Conversion

- Bit CONT à 0 (registre CR2)
- Démarrage par bit ADON uniquement pour groupe réguliers
- Une fois le canal converti
  - La donnée convertie est stockée dans le registre 16bits DR
  - Le drapeau EOC est positionnée
  - Une interruption est générée si le bit EOCIE est à 1

Mode Conversion Continue

- Bit CONT à 1 (registre CR2)
- Démarrage par bit ADON uniquement pour groupe réguliers
- Une fois le canal converti
  - La donnée convertie est stockée dans le registre 16bits DR
  - Le drapeau EOC est positionnée

ADC STM32 : Description Fonctionnelle

Mode Simple Conversion

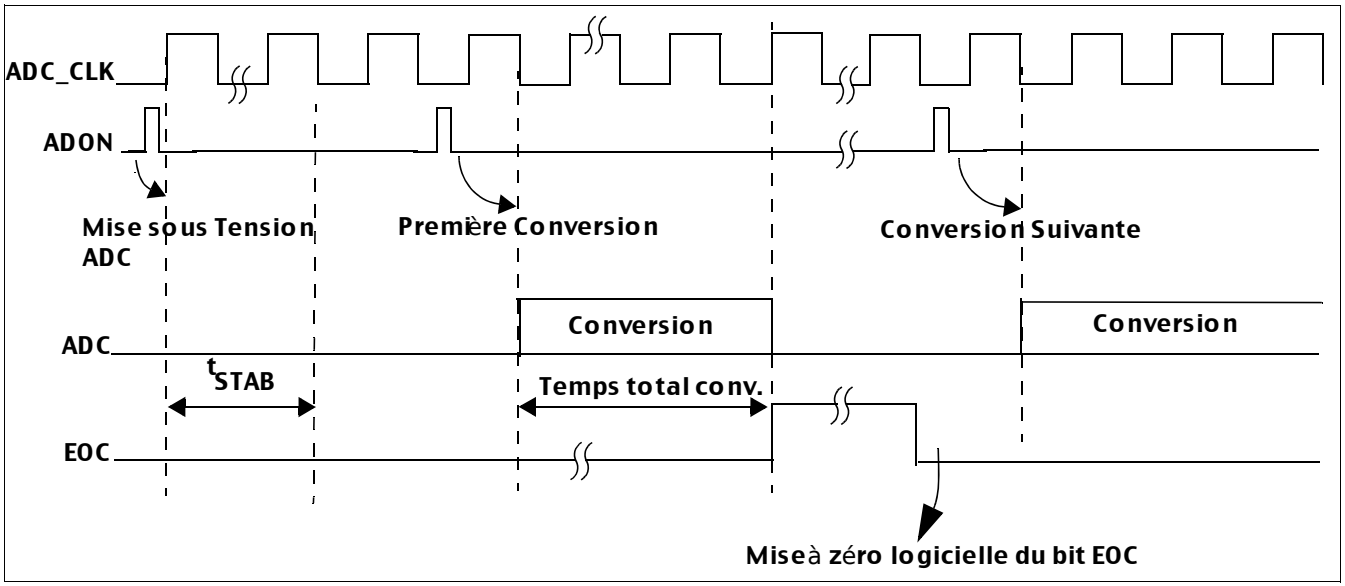
- Bit CONT à 0 (registre CR2)
- Démarrage par bit ADON uniquement pour groupe réguliers
- Une fois le canal converti
  - La donnée convertie est stockée dans le registre 16bits DR
  - Le drapeau EOC est positionnée
  - Une interruption est générée si le bit EOCIE est à 1

Mode Conversion Continue

- Bit CONT à 1 (registre CR2)
- Démarrage par bit ADON uniquement pour groupe réguliers
- Une fois le canal converti
  - La donnée convertie est stockée dans le registre 16bits DR
  - Le drapeau EOC est positionnée
  - Une interruption est générée si le bit EOCIE est à 1

ADC STM32 : Description Fonctionnelle

Diagramme Temporel Simple Conversion



ADC STM32 : Description Fonctionnelle

Balayage Multi-Canaux

ADC STM32 : Description Fonctionnelle

Balayage Multi-Canaux

- Bit SCAN à 1 (registre CR1)

ADC STM32 : Description Fonctionnelle

Balayage Multi-Canaux

- Bit SCAN à 1 (registre CR1)
- Conversion des canaux sélectionnés via les registres SQRx et JSQR

ADC STM32 : Description Fonctionnelle

Balayage Multi-Canaux

- Bit SCAN à 1 (registre CR1)
- Conversion des canaux sélectionnés via les registres SQRx et JSQR
- Une conversion est effectuée pour chaque canal du groupe

ADC STM32 : Description Fonctionnelle

Balayage Multi-Canaux

- Bit SCAN à 1 (registre CR1)
- Conversion des canaux sélectionnés via les registres SQRx et JSQR
- Une conversion est effectuée pour chaque canal du groupe
- Après la fin de conversion d'un canal, le canal suivant est directement traité

ADC STM32 : Description Fonctionnelle

Balayage Multi-Canaux

- Bit SCAN à 1 (registre CR1)
- Conversion des canaux sélectionnés via les registres SQRx et JSQR
- Une conversion est effectuée pour chaque canal du groupe
- Après la fin de conversion d'un canal, le canal suivant est directement traité
- Nécessité d'utiliser le DMA pour récupérer les données

ADC STM32 : Description Fonctionnelle

Balayage Multi-Canaux

- Bit SCAN à 1 (registre CR1)
- Conversion des canaux sélectionnés via les registres SQRx et JSQR
- Une conversion est effectuée pour chaque canal du groupe
- Après la fin de conversion d'un canal, le canal suivant est directement traité
- Nécessité d'utiliser le DMA pour récupérer les données
- Une interruption est générée, si EOCIE est à 1, après la dernière conversion

ADC STM32 : Description Fonctionnelle

Balayage Multi-Canaux

- Bit SCAN à 1 (registre CR1)
- Conversion des canaux sélectionnés via les registres SQRx et JSQR
- Une conversion est effectuée pour chaque canal du groupe
- Après la fin de conversion d'un canal, le canal suivant est directement traité
- Nécessité d'utiliser le DMA pour récupérer les données
- Une interruption est générée, si EOCIE est à 1, après la dernière conversion

Mode Discontinu

ADC STM32 : Description Fonctionnelle

Balayage Multi-Canaux

- Bit SCAN à 1 (registre CR1)
- Conversion des canaux sélectionnés via les registres SQRx et JSQR
- Une conversion est effectuée pour chaque canal du groupe
- Après la fin de conversion d'un canal, le canal suivant est directement traité
- Nécessité d'utiliser le DMA pour récupérer les données
- Une interruption est générée, si EOCIE est à 1, après la dernière conversion

Mode Discontinu

- Bit DISCEN à 1 (registre CR1)

ADC STM32 : Description Fonctionnelle

Balayage Multi-Canaux

- Bit SCAN à 1 (registre CR1)
- Conversion des canaux sélectionnés via les registres SQRx et JSQR
- Une conversion est effectuée pour chaque canal du groupe
- Après la fin de conversion d'un canal, le canal suivant est directement traité
- Nécessité d'utiliser le DMA pour récupérer les données
- Une interruption est générée, si EOCIE est à 1, après la dernière conversion

Mode Discontinu

- Bit DISCEN à 1 (registre CR1)
- Conversion d'un sous ensemble de canaux d'un groupe

ADC STM32 : Description Fonctionnelle

Balayage Multi-Canaux

- Bit SCAN à 1 (registre CR1)
- Conversion des canaux sélectionnés via les registres SQRx et JSQR
- Une conversion est effectuée pour chaque canal du groupe
- Après la fin de conversion d'un canal, le canal suivant est directement traité
- Nécessité d'utiliser le DMA pour récupérer les données
- Une interruption est générée, si EOCIE est à 1, après la dernière conversion

Mode Discontinu

- Bit DISCEN à 1 (registre CR1)
- Conversion d'un sous ensemble de canaux d'un groupe
- Taille du sous ensemble $n \leqslant 8$

ADC STM32 : Description Fonctionnelle

Balayage Multi-Canaux

- Bit SCAN à 1 (registre CR1)
- Conversion des canaux sélectionnés via les registres SQRx et JSQR
- Une conversion est effectuée pour chaque canal du groupe
- Après la fin de conversion d'un canal, le canal suivant est directement traité
- Nécessité d'utiliser le DMA pour récupérer les données
- Une interruption est générée, si EOCIE est à 1, après la dernière conversion

Mode Discontinu

- Bit DISCEN à 1 (registre CR1)
- Conversion d'un sous ensemble de canaux d'un groupe
- Taille du sous ensemble $n \leqslant 8$
- Taille spécifiée via les bits DISCNUM[2:0] (registre CR1)

ADC STM32 : Description Fonctionnelle

Balayage Multi-Canaux

- Bit SCAN à 1 (registre CR1)
- Conversion des canaux sélectionnés via les registres SQRx et JSQR
- Une conversion est effectuée pour chaque canal du groupe
- Après la fin de conversion d'un canal, le canal suivant est directement traité
- Nécessité d'utiliser le DMA pour récupérer les données
- Une interruption est générée, si EOCIE est à 1, après la dernière conversion

Mode Discontinu

- Bit DISCEN à 1 (registre CR1)
- Conversion d'un sous ensemble de canaux d'un groupe
- Taille du sous ensemble $n \leqslant 8$
- Taille spécifiée via les bits DISCNUM[2:0] (registre CR1)
- Piloté par un événement externe

ADC STM32 : Description Fonctionnelle

Balayage Multi-Canaux

- Bit SCAN à 1 (registre CR1)
- Conversion des canaux sélectionnés via les registres SQRx et JSQR
- Une conversion est effectuée pour chaque canal du groupe
- Après la fin de conversion d'un canal, le canal suivant est directement traité
- Nécessité d'utiliser le DMA pour récupérer les données
- Une interruption est générée, si EOCIE est à 1, après la dernière conversion

Mode Discontinu

- Bit DISCEN à 1 (registre CR1)
- Conversion d'un sous ensemble de canaux d'un groupe
- Taille du sous ensemble $n \leqslant 8$
- Taille spécifiée via les bits DISCNUM[2:0] (registre CR1)
- Piloté par un événement externe
- Lors de l'événement, conversion des n canaux parmi le groupe

ADC STM32 : Description Fonctionnelle

Balayage Multi-Canaux

- Bit SCAN à 1 (registre CR1)
- Conversion des canaux sélectionnés via les registres SQRx et JSQR
- Une conversion est effectuée pour chaque canal du groupe
- Après la fin de conversion d'un canal, le canal suivant est directement traité
- Nécessité d'utiliser le DMA pour récupérer les données
- Une interruption est générée, si EOCIE est à 1, après la dernière conversion

Mode Discontinu

- Bit DISCEN à 1 (registre CR1)
- Conversion d'un sous ensemble de canaux d'un groupe
- Taille du sous ensemble $n \leqslant 8$
- Taille spécifiée via les bits DISCNUM[2:0] (registre CR1)
- Piloté par un événement externe
- Lors de l'événement, conversion des n canaux parmi le groupe
- Exemple : $n = 3$, groupe canal=[0,1,2,3,6,7,9,10]

ADC STM32 : Description Fonctionnelle

Balayage Multi-Canaux

- Bit SCAN à 1 (registre CR1)
- Conversion des canaux sélectionnés via les registres SQRx et JSQR
- Une conversion est effectuée pour chaque canal du groupe
- Après la fin de conversion d'un canal, le canal suivant est directement traité
- Nécessité d'utiliser le DMA pour récupérer les données
- Une interruption est générée, si EOCIE est à 1, après la dernière conversion

Mode Discontinu

- Bit DISCEN à 1 (registre CR1)
- Conversion d'un sous ensemble de canaux d'un groupe
- Taille du sous ensemble $n \leqslant 8$
- Taille spécifiée via les bits DISCNUM[2:0] (registre CR1)
- Piloté par un événement externe
- Lors de l'événement, conversion des n canaux parmi le groupe
- Exemple : $n = 3$, groupe canal=[0,1,2,3,6,7,9,10]
  - 1^{er} événement, conv. des canaux [0,1,2]

ADC STM32 : Description Fonctionnelle

Balayage Multi-Canaux

- Bit SCAN à 1 (registre CR1)
- Conversion des canaux sélectionnés via les registres SQRx et JSQR
- Une conversion est effectuée pour chaque canal du groupe
- Après la fin de conversion d'un canal, le canal suivant est directement traité
- Nécessité d'utiliser le DMA pour récupérer les données
- Une interruption est générée, si EOCIE est à 1, après la dernière conversion

Mode Discontinu

- Bit DISCEN à 1 (registre CR1)
- Conversion d'un sous ensemble de canaux d'un groupe
- Taille du sous ensemble $n \leqslant 8$
- Taille spécifiée via les bits DISCNUM[2:0] (registre CR1)
- Piloté par un événement externe
- Lors de l'événement, conversion des n canaux parmi le groupe
- Exemple : $n = 3$, groupe canal=[0,1,2,3,6,7,9,10]
  - 1^{er} événement, conv. des canaux [0,1,2]
  - 2^{ieme} événement, conv. des canaux [3,6,7]

ADC STM32 : Description Fonctionnelle

Balayage Multi-Canaux

- Bit SCAN à 1 (registre CR1)
- Conversion des canaux sélectionnés via les registres SQRx et JSQR
- Une conversion est effectuée pour chaque canal du groupe
- Après la fin de conversion d'un canal, le canal suivant est directement traité
- Nécessité d'utiliser le DMA pour récupérer les données
- Une interruption est générée, si EOCIE est à 1, après la dernière conversion

Mode Discontinu

- Bit DISCEN à 1 (registre CR1)
- Conversion d'un sous ensemble de canaux d'un groupe
- Taille du sous ensemble $n \leqslant 8$
- Taille spécifiée via les bits DISCNUM[2:0] (registre CR1)
- Piloté par un événement externe
- Lors de l'événement, conversion des n canaux parmi le groupe
- Exemple : $n = 3$, groupe canal=[0,1,2,3,6,7,9,10]
  - 1^{er} événement, conv. des canaux [0,1,2]
  - 2^{ieme} événement, conv. des canaux [3,6,7]
  - 3^{ieme} événement, conv. des canaux [9,10] et EOC

ADC STM32 : Description Fonctionnelle

Balayage Multi-Canaux

- Bit SCAN à 1 (registre CR1)
- Conversion des canaux sélectionnés via les registres SQRx et JSQR
- Une conversion est effectuée pour chaque canal du groupe
- Après la fin de conversion d'un canal, le canal suivant est directement traité
- Nécessité d'utiliser le DMA pour récupérer les données
- Une interruption est générée, si EOCIE est à 1, après la dernière conversion

Mode Discontinu

- Bit DISCEN à 1 (registre CR1)
- Conversion d'un sous ensemble de canaux d'un groupe
- Taille du sous ensemble $n \leqslant 8$
- Taille spécifiée via les bits DISCNUM[2:0] (registre CR1)
- Piloté par un événement externe
- Lors de l'événement, conversion des n canaux parmi le groupe
- Exemple : $n = 3$, groupe canal=[0,1,2,3,6,7,9,10]
  - 1^{er} événement, conv. des canaux [0,1,2]
  - 2^{ieme} événement, conv. des canaux [3,6,7]
  - 3^{ieme} événement, conv. des canaux [9,10] et EOC
  - 4^{ieme} événement, conv. des canaux [0,1,2]

ADC STM32 : Description Fonctionnelle

Calibration

ADC STM32 : Description Fonctionnelle

Calibration

- Procédure d'auto-calibration

ADC STM32 : Description Fonctionnelle

Calibration

- Procédure d'auto-calibration
- Réduction des erreurs de précision du aux variations des capacités internes

ADC STM32 : Description Fonctionnelle

Calibration

- Procédure d'auto-calibration
- Réduction des erreurs de précision du aux variations des capacités internes
- Calcul de l'erreur pour chaque capacité

ADC STM32 : Description Fonctionnelle

Calibration

- Procédure d'auto-calibration
- Réduction des erreurs de précision du aux variations des capacités internes
- Calcul de l'erreur pour chaque capacité
- Démarrage par mise à 1 du bit CAL (registre CR2)

ADC STM32 : Description Fonctionnelle

Calibration

- Procédure d'auto-calibration
- Réduction des erreurs de précision du aux variations des capacités internes
- Calcul de l'erreur pour chaque capacité
- Démarrage par mise à 1 du bit CAL (registre CR2)
- Remise à zéro automatique du bit CAL après calibration

ADC STM32 : Description Fonctionnelle

Calibration

- Procédure d'auto-calibration
- Réduction des erreurs de précision du aux variations des capacités internes
- Calcul de l'erreur pour chaque capacité
- Démarrage par mise à 1 du bit CAL (registre CR2)
- Remise à zéro automatique du bit CAL après calibration
- Procédure recommandée à la mise sous tension de l'ADC

ADC STM32 : Description Fonctionnelle

Calibration

- Procédure d'auto-calibration
- Réduction des erreurs de précision du aux variations des capacités internes
- Calcul de l'erreur pour chaque capacité
- Démarrage par mise à 1 du bit CAL (registre CR2)
- Remise à zéro automatique du bit CAL après calibration
- Procédure recommandée à la mise sous tension de l'ADC

Alignement des Données

ADC STM32 : Description Fonctionnelle

Calibration

- Procédure d'auto-calibration
- Réduction des erreurs de précision du aux variations des capacités internes
- Calcul de l'erreur pour chaque capacité
- Démarrage par mise à 1 du bit CAL (registre CR2)
- Remise à zéro automatique du bit CAL après calibration
- Procédure recommandée à la mise sous tension de l'ADC

Alignement des Données

- Bit ALIGN (registre CR2)

ADC STM32 : Description Fonctionnelle

Calibration

- Procédure d'auto-calibration
- Réduction des erreurs de précision du aux variations des capacités internes
- Calcul de l'erreur pour chaque capacité
- Démarrage par mise à 1 du bit CAL (registre CR2)
- Remise à zéro automatique du bit CAL après calibration
- Procédure recommandée à la mise sous tension de l'ADC

Alignement des Données

- Bit ALIGN (registre CR2)
- Alignement à droite si ALIGN $= 0$

ADC STM32 : Description Fonctionnelle

Calibration

- Procédure d'auto-calibration
- Réduction des erreurs de précision du aux variations des capacités internes
- Calcul de l'erreur pour chaque capacité
- Démarrage par mise à 1 du bit CAL (registre CR2)
- Remise à zéro automatique du bit CAL après calibration
- Procédure recommandée à la mise sous tension de l'ADC

Alignement des Données

- Bit ALIGN (registre CR2)
- Alignement à droite si ALIGN $= 0$

ADC STM32 : Description Fonctionnelle

Calibration

- Procédure d'auto-calibration
- Réduction des erreurs de précision du aux variations des capacités internes
- Calcul de l'erreur pour chaque capacité
- Démarrage par mise à 1 du bit CAL (registre CR2)
- Remise à zéro automatique du bit CAL après calibration
- Procédure recommandée à la mise sous tension de l'ADC

Alignement des Données

- Bit ALIGN (registre CR2)
- Alignement à droite si ALIGN $= 0$
- Alignement à gauche si ALIGN $= 1$

ADC STM32 : Description Fonctionnelle

Temps de Conversion

ADC STM32 : Description Fonctionnelle

Temps de Conversion

- Echantillonage sur n ADCCLK cycles

ADC STM32 : Description Fonctionnelle

Temps de Conversion

- Echantillonage sur n ADCCLK cycles
- n modifiable

ADC STM32 : Description Fonctionnelle

Temps de Conversion

- Echantillonage sur n ADCCLK cycles
- n modifiable
- bits SMPx[2:0] pour le canal x (registres SMPR1 et SMPR2)

ADC STM32 : Description Fonctionnelle

Temps de Conversion

- Echantillonage sur n ADCCLK cycles
- n modifiable
- bits SMPx[2:0] pour le canal x (registres SMPR1 et SMPR2)
- Différenciation du temps par canal

ADC STM32 : Description Fonctionnelle

Temps de Conversion

- Echantillonage sur n ADCCLK cycles
- n modifiable
- bits SMPx[2:0] pour le canal x (registres SMPR1 et SMPR2)
- Différenciation du temps par canal
- Temps Total

ADC STM32 : Description Fonctionnelle

Temps de Conversion

- Echantillonage sur n ADCCLK cycles
- n modifiable
- bits SMPx[2:0] pour le canal x (registres SMPR1 et SMPR2)
- Différenciation du temps par canal
- Temps Total
  - $T_{conv} = Tps\ Echant. + 12,5\ cycles$

ADC STM32 : Description Fonctionnelle

Temps de Conversion

- Echantillonage sur n ADCCLK cycles
- n modifiable
- bits SMPx[2:0] pour le canal x (registres SMPR1 et SMPR2)
- Différenciation du temps par canal
- Temps Total
  - $T_{conv} = Tps\ Echant. + 12,5\ cycles$

Déclenchement externe

ADC STM32 : Description Fonctionnelle

Temps de Conversion

- Echantillonage sur n ADCCLK cycles
- n modifiable
- bits SMPx[2:0] pour le canal x (registres SMPR1 et SMPR2)
- Différenciation du temps par canal
- Temps Total
  - $T_{conv} = Tps\ Echant. + 12,5\ cycles$

Déclenchement externe

- Conversion pilotable par événement externe

ADC STM32 : Description Fonctionnelle

Temps de Conversion

- Echantillonage sur n ADCCLK cycles
- n modifiable
- bits SMPx[2:0] pour le canal x (registres SMPR1 et SMPR2)
- Différenciation du temps par canal
- Temps Total
  - $T_{conv} = Tps\ Echant. + 12,5\ cycles$

Déclenchement externe

- Conversion pilotable par événement externe
- Entrée EXTI

ADC STM32 : Description Fonctionnelle

Temps de Conversion

- Echantillonage sur n ADCCLK cycles
- n modifiable
- bits SMPx[2:0] pour le canal x (registres SMPR1 et SMPR2)
- Différenciation du temps par canal
- Temps Total
  - $T_{conv} = Tps\ Echant. + 12,5\ cycles$

Déclenchement externe

- Conversion pilotable par événement externe
- Entrée EXTI

ADC STM32 : Description Fonctionnelle

Temps de Conversion

- Echantillonage sur n ADCCLK cycles
- n modifiable
- bits SMPx[2:0] pour le canal x (registres SMPR1 et SMPR2)
- Différenciation du temps par canal
- Temps Total
  - $T_{conv} = Tps\ Echant. + 12,5\ cycles$

Déclenchement externe

- Conversion pilotable par événement externe
- Entrée EXTI
- bit EXTTRIG à 1 (registre CR2)

ADC STM32 : Description Fonctionnelle

Temps de Conversion

- Echantillonage sur n ADCCLK cycles
- n modifiable
- bits SMPx[2:0] pour le canal x (registres SMPR1 et SMPR2)
- Différenciation du temps par canal
- Temps Total
  - $T_{conv} = Tps\ Echant. + 12,5\ cycles$

Déclenchement externe

- Conversion pilotable par événement externe
- Entrée EXTI
- bit EXTTRIG à 1 (registre CR2)
- bits EXTSEL[2:0] pour sélectionner le type de l'événement

ADC STM32 : Description Fonctionnelle

Source de l'événement	Type	Valeur EXTSEL[2:0]
TIM1_CC1	Interne	000
TIM1_CC2	Interne	001
TIM1_CC3	Interne	010
TIM2_CC2	Interne	011
TIM3_TRGO	Interne	100
TIM4_CC4	Interne	101
EXTI line11/TIM8_TRGO	Externe/Interne	110
SWSTART	Logiciel	111

ADC STM32 : Description Fonctionnelle

Requête DMA

ADC STM32 : Description Fonctionnelle

Requête DMA

- Résultat de la conversion pour groupe régulier stocké dans un unique registre DR

ADC STM32 : Description Fonctionnelle

Requête DMA

- Résultat de la conversion pour groupe régulier stocké dans un unique registre DR
- Eviter la perte des données

ADC STM32 : Description Fonctionnelle

Requête DMA

- Résultat de la conversion pour groupe régulier stocké dans un unique registre DR
- Eviter la perte des données
- EOC d'un canal génère une requête DMA permettant le transfert du registre DR en mémoire

ADC STM32 : Description Fonctionnelle

Requête DMA

- Résultat de la conversion pour groupe régulier stocké dans un unique registre DR
- Eviter la perte des données
- EOC d'un canal génère une requête DMA permettant le transfert du registre DR en mémoire
- Emplacement mémoire spécifié par l'utilisateur

ADC STM32 : Description Fonctionnelle

Requête DMA

- Résultat de la conversion pour groupe régulier stocké dans un unique registre DR
- Eviter la perte des données
- EOC d'un canal génère une requête DMA permettant le transfert du registre DR en mémoire
- Emplacement mémoire spécifié par l'utilisateur

Interruptions de l'ADC

ADC STM32 : Description Fonctionnelle

Requête DMA

- Résultat de la conversion pour groupe régulier stocké dans un unique registre DR
- Eviter la perte des données
- EOC d'un canal génère une requête DMA permettant le transfert du registre DR en mémoire
- Emplacement mémoire spécifié par l'utilisateur

Interruptions de l'ADC

- Possibilité de générer une interruption à la fin de la conversion

ADC STM32 : Description Fonctionnelle

Requête DMA

- Résultat de la conversion pour groupe régulier stocké dans un unique registre DR
- Eviter la perte des données
- EOC d'un canal génère une requête DMA permettant le transfert du registre DR en mémoire
- Emplacement mémoire spécifié par l'utilisateur

Interruptions de l'ADC

- Possibilité de générer une interruption à la fin de la conversion
- Autorisation par bit EOCIE (groupe régulier) ou JEOCIE (groupe injecté)

ADC STM32 : Description Fonctionnelle

Requête DMA

- Résultat de la conversion pour groupe régulier stocké dans un unique registre DR
- Eviter la perte des données
- EOC d'un canal génère une requête DMA permettant le transfert du registre DR en mémoire
- Emplacement mémoire spécifié par l'utilisateur

Interruptions de l'ADC

- Possibilité de générer une interruption à la fin de la conversion
- Autorisation par bit EOCIE (groupe régulier) ou JEOCIE (groupe injecté)
- Les interruption de l'ADC1 et ADC2 sont mappées sur le même vecteur d'interruption

ADC STM32 : Registre Horloge RCC

CFGR, Adresse Base ADC1 = 0x4001 2400, Offset = 0x04, Valeur Initiale = 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					MCO[2:0]			Res.	USB PRE	PLLMUL[3:0]				PLL XTPRE	PLL SRC
					rw	rw	rw		rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADC PRE[1:0]		PPRE2[2:0]			PPRE1[2:0]			HPRE[3:0]				SWS[1:0]		SW[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	r	r	rw	rw

ADC STM32 : Registre Horloge RCC

APB2ENR, Adresse Base ADC1 = 0x4001 2400, Offset = 0x18, Valeur Initiale = 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADC3 EN	USART1EN	TIM8 EN	SPI1 EN	TIM1 EN	ADC2 EN	ADC1 EN	IOPG EN	IOPF EN	IOPE EN	IOPD EN	IOPC EN	IOPB EN	IOPA EN	Res.	AFIO EN
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw

ADC STM32 : Registre

SR, Adresse Base ADC1 = 0x4001 2400, Offset = 0x00, Valeur Initiale = 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											STRT	JSTRT	JEOC	EOC	AWD
Res.											rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

ADC STM32 : Registre

CR1, Adresse Base ADC1 = 0x4001 2400, Offset = 0x04, Valeur Initiale = 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								AWDEN	JAWDEN	Reserved		DUALMOD[3:0]			
Res.								rw	rw	Res.		rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DISCNUM[2:0]			JDISCEN	DISCEN	JAUTO	AWDSGL	SCAN	JEOC IE	AWDIE	EOCIE	AWDCH[4:0]				
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

CR2, Adresse Base ADC1 = 0x4001 2400, Offset = 0x08, Valeur Initiale = 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								TSVREFE	SWSTART	JSWSTART	EXTTRIG	EXTSEL[2:0]			Res.
Res.								rw	rw	rw	rw	rw	rw	rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
JEXTTRIG	JEXTSEL[2:0]			ALIGN	Reserved		DMA	Reserved				RSTCAL	CAL	CONT	ADON
rw	rw	rw	rw	rw	Res.		rw	Res.				rw	rw	rw	rw

ADC STM32 : Registre

SMPR1, Adresse Base ADC1 = 0x4001 2400, Offset = 0x0C, Valeur Initiale = 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								SMP17[2:0]			SMP16[2:0]			SMP15[2:1]	
Res.								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMP 15_0	SMP14[2:0]			SMP13[2:0]			SMP12[2:0]			SMP11[2:0]			SMP10[2:0]		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

SMPR2, Adresse Base ADC1 = 0x4001 2400, Offset = 0x10, Valeur Initiale = 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		SMP9[2:0]			SMP8[2:0]			SMP7[2:0]			SMP6[2:0]			SMP5[2:1]	
Res.		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMP 5_0	SMP4[2:0]			SMP3[2:0]			SMP2[2:0]			SMP1[2:0]			SMP0[2:0]		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

ADC STM32 : Registre

SQR1, Adresse Base ADC1 = 0x4001 2400, Offset = 0x2C, Valeur Initiale = 0x0000 0000

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

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

SQR2, Adresse Base ADC1 = 0x4001 2400, Offset = 0x30, Valeur Initiale = 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		SQ12[4:0]					SQ11[4:0]					SQ10[4:1]			
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SQ10_0	SQ9[4:0]					SQ8[4:0]					SQ7[4:0]				
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

SQR3, Adresse Base ADC1 = 0x4001 2400, Offset = 0x34, Valeur Initiale = 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		SQ6[4:0]					SQ5[4:0]					SQ4[4:1]			
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SQ4_0	SQ3[4:0]					SQ2[4:0]					SQ1[4:0]				
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

ADC STM32 : Registre

DR, Adresse Base ADC1 = 0x4001 2400, Offset = 0x4C, Valeur Initiale = 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADC2DATA[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

ADC STM32 : Registre

Autres Registres

- JOFR1, JOFR2, JOFR3, JOFR4
  - Offset pour groupe injecté. A soustraire de la valeur convertie
- HTR, LTR
  - Seuils Haut (HTR) et Bas (LTR) des tensions pour le watchdog
- JSQR
  - Nombre de conversion en groupe injecté
- JDR1, JDR2, JDR3, JDR4
  - Données lors de conversion groupe injecté

DMA STM32 : Caractéristiques Principales

- 12 canaux indépendants, 7 pour DMA1 et 5 pour DMA2

DMA STM32 : Caractéristiques Principales

- 12 canaux indépendants, 7 pour DMA1 et 5 pour DMA2
- Connectés sur les périphériques du STM32

DMA STM32 : Caractéristiques Principales

- 12 canaux indépendants, 7 pour DMA1 et 5 pour DMA2
- Connectés sur les périphériques du STM32
- Priorités programmables entre canaux

DMA STM32 : Caractéristiques Principales

- 12 canaux indépendants, 7 pour DMA1 et 5 pour DMA2
- Connectés sur les périphériques du STM32
- Priorités programmables entre canaux
- Adressage circulaire supporté

DMA STM32 : Caractéristiques Principales

- 12 canaux indépendants, 7 pour DMA1 et 5 pour DMA2
- Connectés sur les périphériques du STM32
- Priorités programmables entre canaux
- Adressage circulaire supporté
- 3 événements possible : demi-transfert DMA, transfert complet DMA et erreur de transfert

DMA STM32 : Caractéristiques Principales

- 12 canaux indépendants, 7 pour DMA1 et 5 pour DMA2
- Connectés sur les périphériques du STM32
- Priorités programmables entre canaux
- Adressage circulaire supporté
- 3 événements possible : demi-transfert DMA, transfert complet DMA et erreur de transfert
- Transferts mémoire-mémoire, périphérique-périphérique

DMA STM32 : Caractéristiques Principales

- 12 canaux indépendants, 7 pour DMA1 et 5 pour DMA2
- Connectés sur les périphériques du STM32
- Priorités programmables entre canaux
- Adressage circulaire supporté
- 3 événements possible : demi-transfert DMA, transfert complet DMA et erreur de transfert
- Transferts mémoire-mémoire, périphérique-périphérique
- Transferts périphérique-mémoire, mémoire-périphérique

DMA STM32 : Caractéristiques Principales

- 12 canaux indépendants, 7 pour DMA1 et 5 pour DMA2
- Connectés sur les périphériques du STM32
- Priorités programmables entre canaux
- Adressage circulaire supporté
- 3 événements possible : demi-transfert DMA, transfert complet DMA et erreur de transfert
- Transferts mémoire-mémoire, périphérique-périphérique
- Transferts périphérique-mémoire, mémoire-périphérique
- Nombre de données à transférer programmable jusqu'à 65536

DMA STM32 : Description fonctionnelle

- le DMA réalise des transferts directs en mémoire en partageant le bus Système avec le Cortex-M3

DMA STM32 : Description fonctionnelle

- le DMA réalise des transferts directs en mémoire en partageant le bus Système avec le Cortex-M3
- Après un événement (fin de conversion par exemple) le périphérique envoie une requête au DMA

DMA STM32 : Description fonctionnelle

- le DMA réalise des transferts directs en mémoire en partageant le bus Système avec le Cortex-M3
- Après un événement (fin de conversion par exemple) le périphérique envoie une requête au DMA
- Le DMA traite la requête en fonction de la priorité du canal lié

DMA STM32 : Description fonctionnelle

- le DMA réalise des transferts directs en mémoire en partageant le bus Système avec le Cortex-M3
- Après un événement (fin de conversion par exemple) le périphérique envoie une requête au DMA
- Le DMA traite la requête en fonction de la priorité du canal lié
- Le transfert s'effectue suivant 3 étapes :

DMA STM32 : Description fonctionnelle

- le DMA réalise des transferts directs en mémoire en partageant le bus Système avec le Cortex-M3
- Après un événement (fin de conversion par exemple) le périphérique envoie une requête au DMA
- Le DMA traite la requête en fonction de la priorité du canal lié
- Le transfert s'effectue suivant 3 étapes :
  - Chargement de la donnée à transférer. Adresse spécifiée par le registre DMA_CPARx (périphérique) ou DMA_CMARx (mémoire)

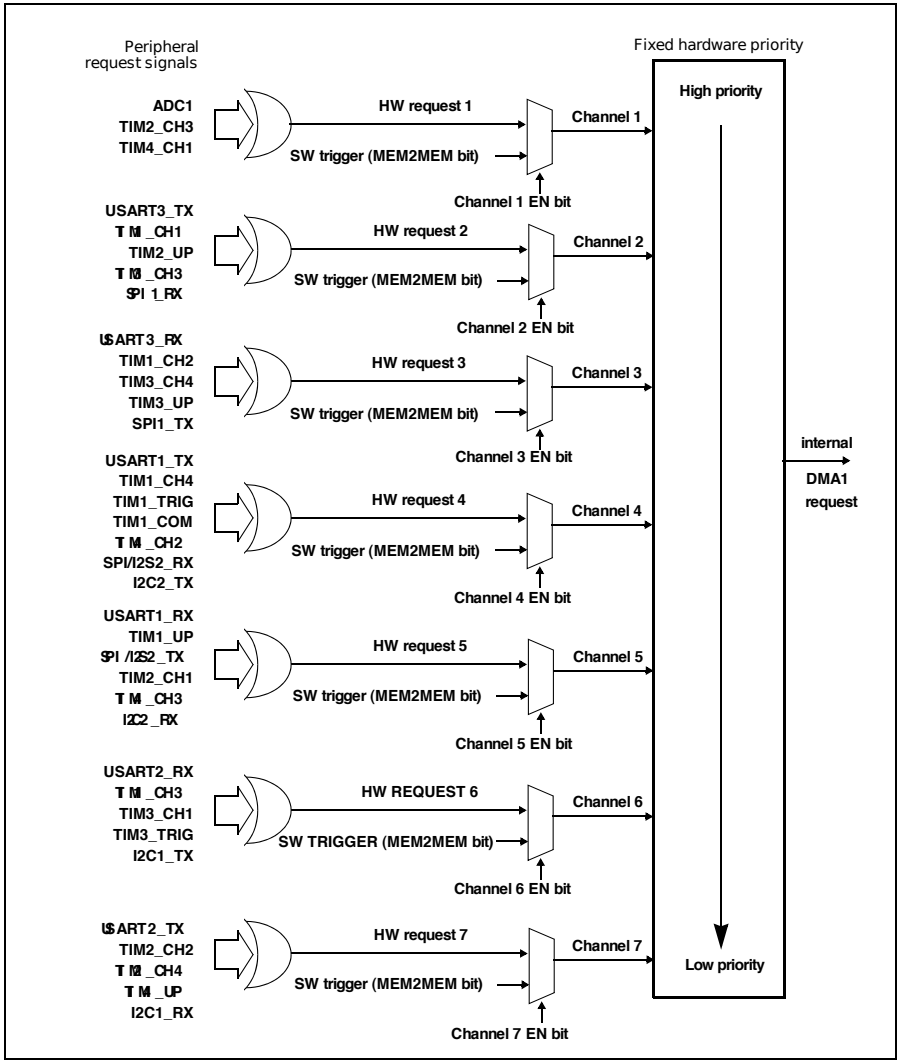
DMA STM32 : Description fonctionnelle

- le DMA réalise des transferts directs en mémoire en partageant le bus Système avec le Cortex-M3
- Après un événement (fin de conversion par exemple) le périphérique envoie une requête au DMA
- Le DMA traite la requête en fonction de la priorité du canal lié
- Le transfert s'effectue suivant 3 étapes :
  - Chargement de la donnée à transférer. Adresse spécifiée par le registre DMA_CPARx (périphérique) ou DMA_CMARx (mémoire)
  - Sauvegarde de la donnée transférée. Adresse spécifiée par le registre DMA_CPARx (périphérique) ou DMA_CMARx (mémoire)

DMA STM32 : Description fonctionnelle

- le DMA réalise des transferts directs en mémoire en partageant le bus Système avec le Cortex-M3
- Après un événement (fin de conversion par exemple) le périphérique envoie une requête au DMA
- Le DMA traite la requête en fonction de la priorité du canal lié
- Le transfert s'effectue suivant 3 étapes :
  - Chargement de la donnée à transférer. Adresse spécifiée par le registre DMA_CPARx (périphérique) ou DMA_CMARx (mémoire)
  - Sauvegarde de la donnée transférée. Adresse spécifiée par le registre DMA_CPARx (périphérique) ou DMA_CMARx (mémoire)
  - Décrémentation du registre contenant le nombre de transferts à réaliser DMA_CNDTRx

DMA STM32 : Contrôleur



DMA STM32 : Canaux DMA1

Peripherals	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6	Channel 7
ADC1	**ADC1**						
SPI/I^2S		**SPI1_RX**	**SPI1_TX**	**SPI/I2S2_RX**	**SPI/I2S2_TX**		
USART		**USART3_TX**	**USART3_RX**	**USART1_TX**	**USART1_RX**	**USART2_RX**	**USART2_TX**
I^2C				**I2C2_TX**	**I2C2_RX**	**I2C1_TX**	**I2C1_RX**
TIM1		**TIM1_CH1**	**TIM1_CH2**	**TIM1_CH4** **TIM1_TRIG** **TIM1_COM**	**TIM1_UP**	**TIM1_CH3**	
TIM2	**TIM2_CH3**	**TIM2_UP**			**TIM2_CH1**		**TIM2_CH2** **TIM2_CH4**
TIM3		**TIM3_CH3**	**TIM3_CH4** **TIM3_UP**			**TIM3_CH1** **TIM3_TRIG**	
TIM4	**TIM4_CH1**			**TIM4_CH2**	**TIM4_CH3**		**TIM4_UP**

DMA STM32 : Canaux DMA2

Peripherals	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5
ADC3(1)					**ADC3**
SPI/I2S3	**SPI/I2S3_RX**	**SPI/I2S3_TX**			
UART4			**UART4_RX**		**UART4_TX**
SDIO(1)				**SDIO**	
TIM5	**TIM5_CH4 TIM5_TRIG**	**TIM5_CH3 TIM5_UP**	**T**	**I M5 _CH2 T**	**I M5 _CH1**
TIM6/ DAC_Channel1			**TIM6_UP/ DAC_Channel1**		
TIM7/ DAC_Channel2				**TIM7_UP/ DAC_Channel2**	
TIM8(1)	**TIM8_CH3 TIM8_UP**	**TIM8_CH4 TIM8_TRIG TIM8_COM**	**TIM8_CH1**		**TIM8_CH2**

1. ADC3, SDIO and TIM8 DMA requests are available only in high-density devices.

DMA STM32 : Registre

ISR, Adresse Base DMA1 = 0x4002 0000, Offset = 0x00, Valeur Initiale = 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				TEIF7	HTIF7	TCIF7	GIF7	TEIF6	HTIF6	TCIF6	GIF6	TEIF5	HTIF5	TCIF5	GIF5
				r	r	r	r	r	r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TEIF4	HTIF4	TCIF4	GIF4	TEIF3	HTIF3	TCIF3	GIF3	TEIF2	HTIF2	TCIF2	GIF2	TEIF1	HTIF1	TCIF1	GIF1
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

DMA STM32 : Registre

CCRx, Adresse Base DMA1 = 0x4002 0000, Offset = 0x08 + 20d * Numéro Canal, Valeur Initiale = 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	MEM2 MEM	PL[1:0]		MSIZE[1:0]		PSIZE[1:0]		MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

DMA STM32 : Registre

CNDTRx, Adresse Base DMA1 = 0x4002 0000, Offset = 0x0C + 20d * Numéro Canal, Valeur Initiale = 0x0000 0000

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

DMA STM32 : Registre

CPARx, Adresse Base DMA1 = 0x4002 0000, Offset = 0x010 + 20d * Numéro Canal, Valeur Initiale = 0x0000 0000

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

DMA STM32 : Registre

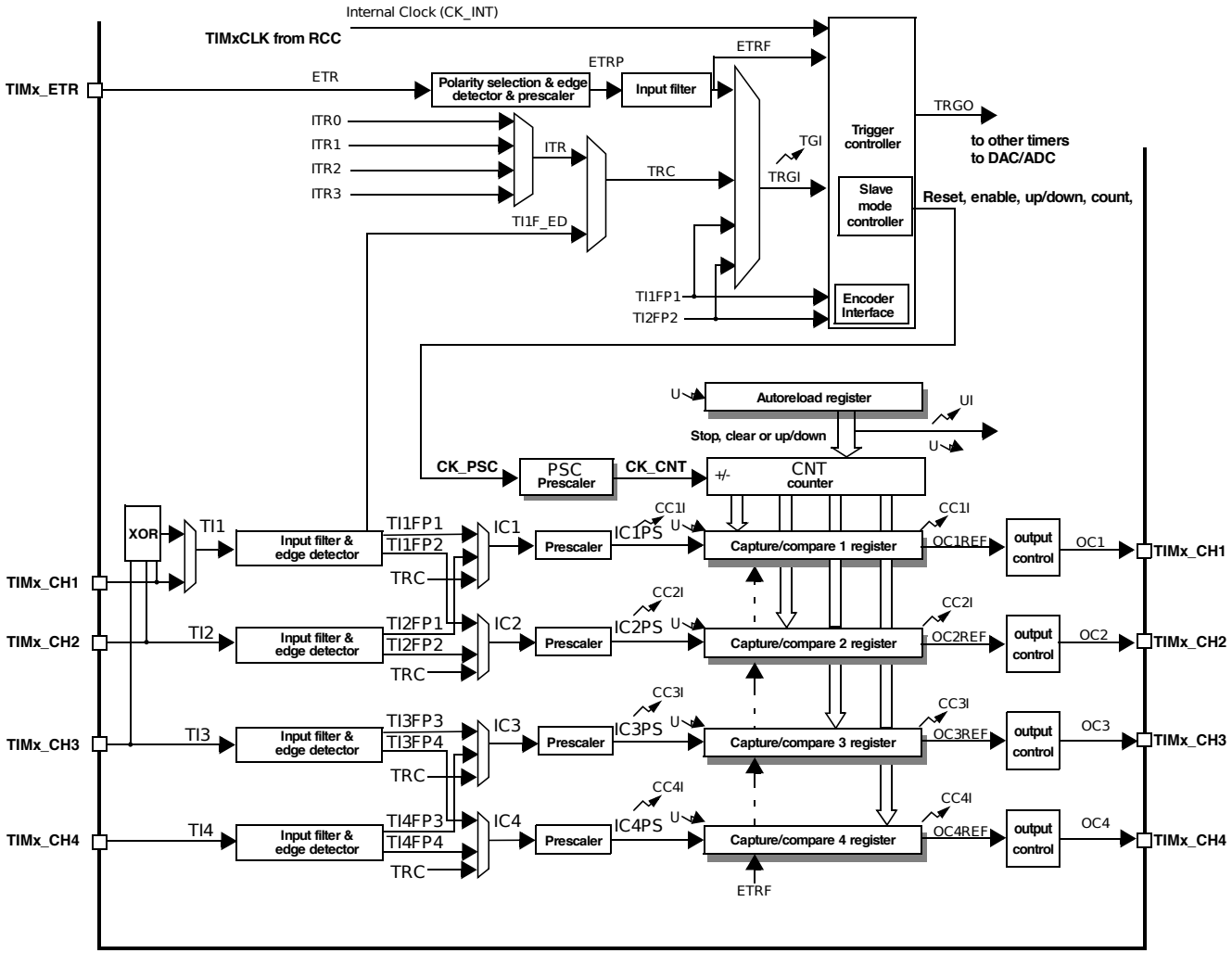
CMARx, Adresse Base DMA1 = 0x4002 0000, Offset = 0x010 + 20d * Numéro Canal, Valeur Initiale = 0x0000 0000

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

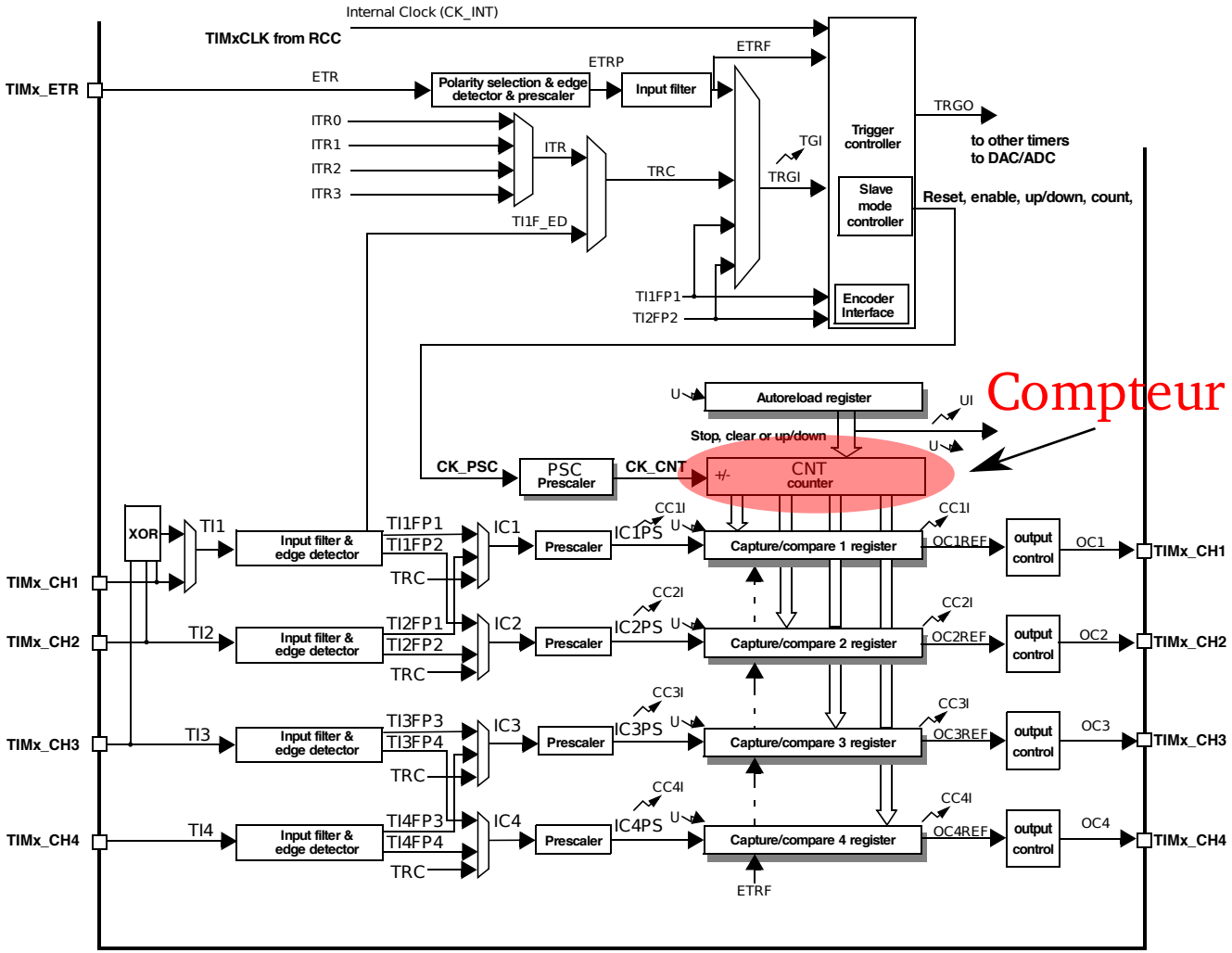
Timer STM32 : Caractéristiques Générales

- Compteur/Décompteur 16 bits rechargeable automatiquement.
- Prédiviseur 16 bits programmable
- Jusqu'à 4 canaux indépandant permettant :
  - La capture d'événements
  - La comparaison avec le compteur
  - Un mode PWM (Pulse Width Modulation)
  - Mode one-pulse
- Circuit de synchronisation externe et chainage des différents timers possible
- Génération d'Interruption ou de requête DMA sur différents événements :
  - Recyclage du compteur, initialisation du compteur
  - Déclenchemebt (Start, Stop, initialisation ou déclenchement interne/externe)
  - Capture d'événement
  - Comparaison

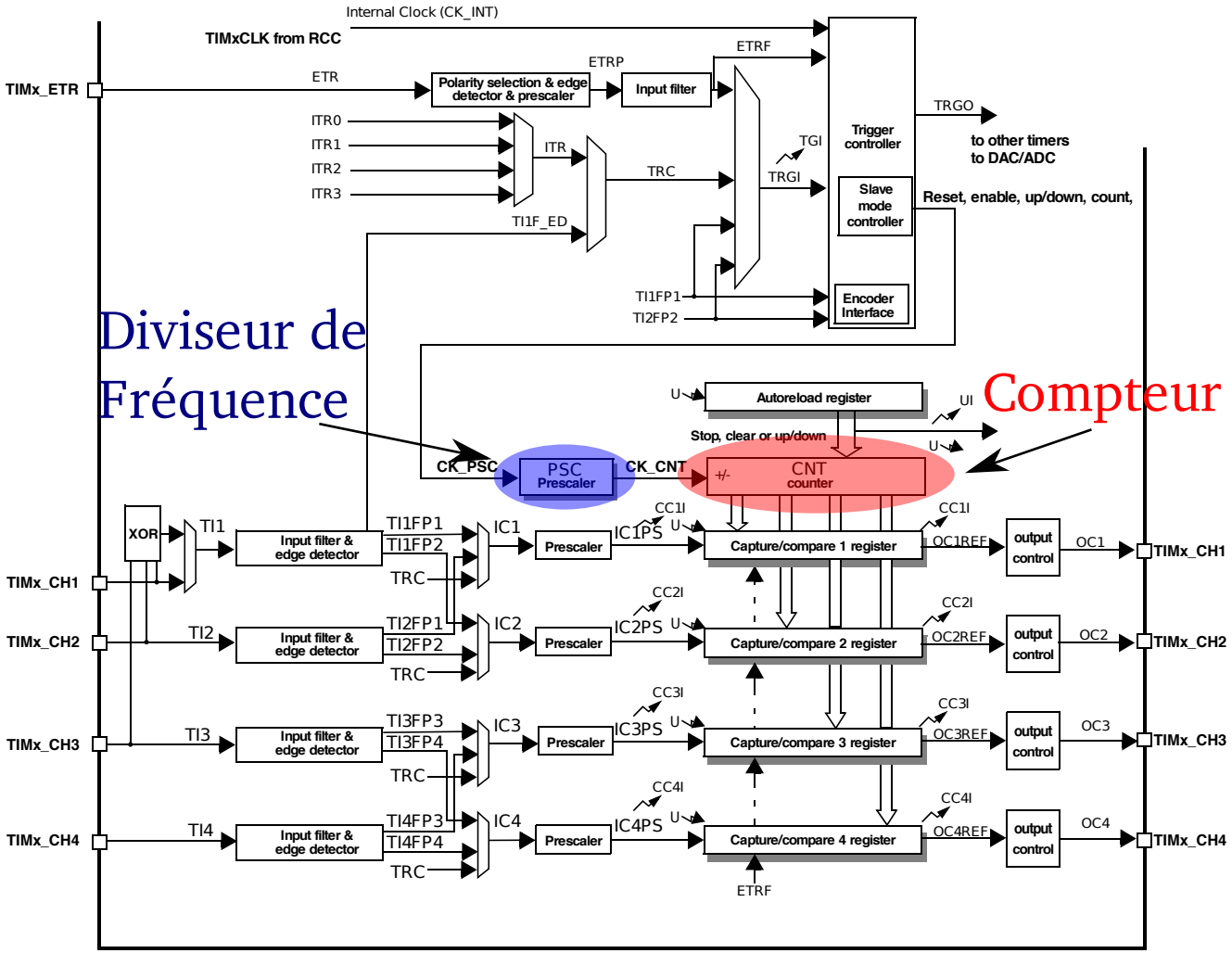
Timer STM32 : Schéma Fonctionnel



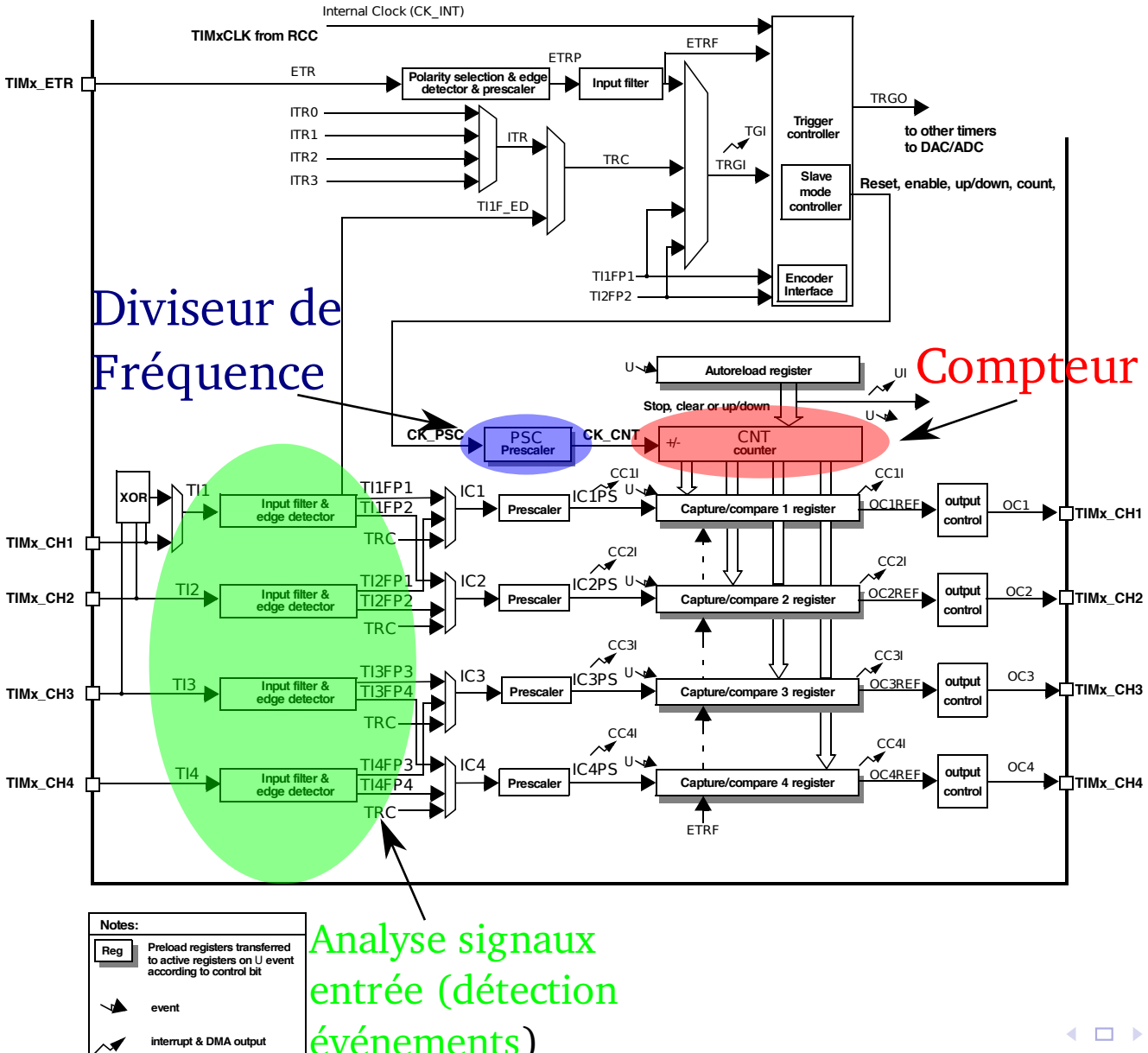
Timer STM32 : Schéma Fonctionnel



Timer STM32 : Schéma Fonctionnel

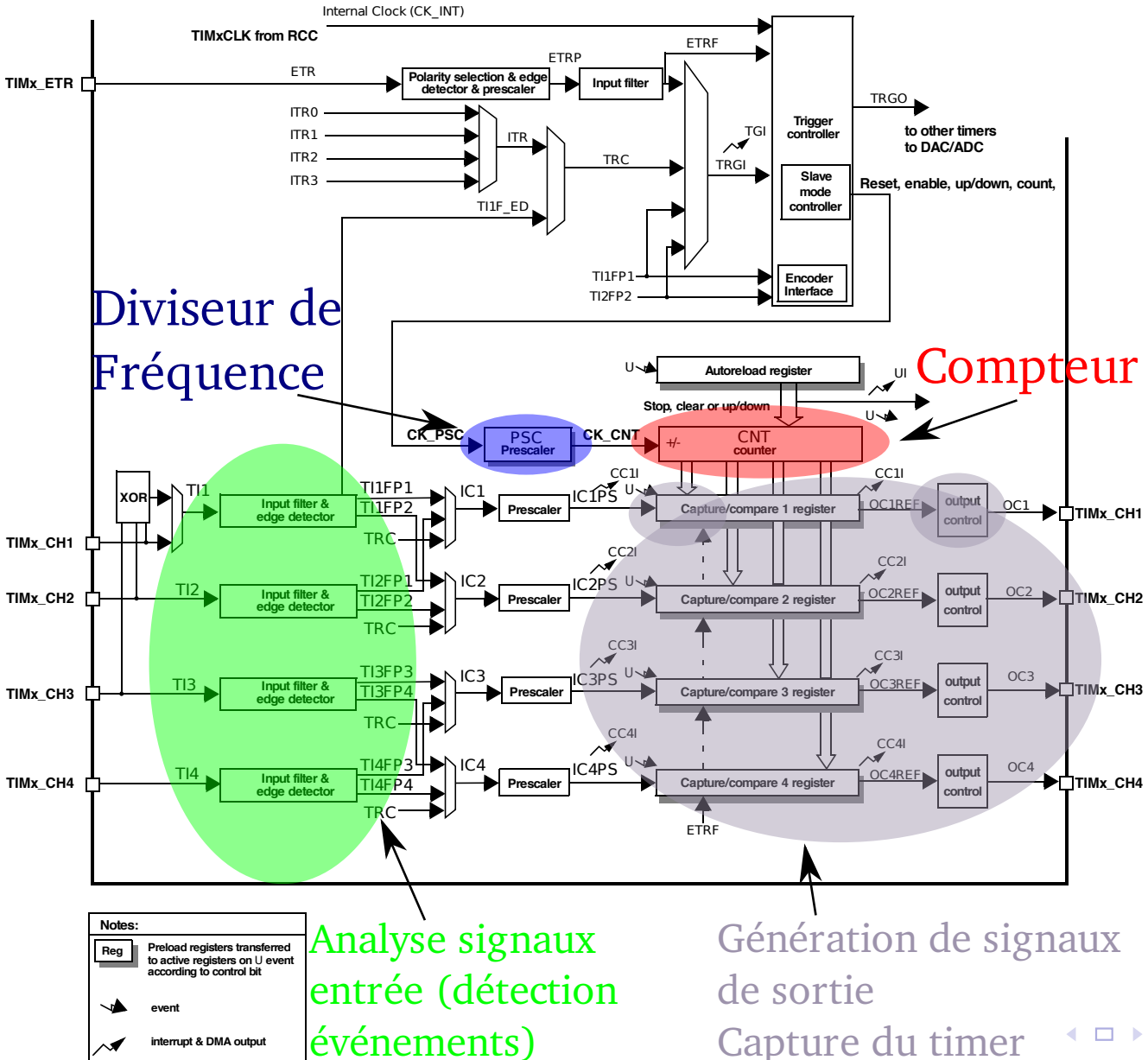


Timer STM32 : Schéma Fonctionnel



Analyse signaux entrée (détection événements)

Timer STM32 : Schéma Fonctionnel



Analyse signaux entrée (détection événements)

Génération de signaux de sortie
Capture du timer

TIMx STM32 : Description Fonctionnelle

Unité de base

TIMx STM32 : Description Fonctionnelle

Unité de base

- Le composant principal : un compteur 16 bits

TIMx STM32 : Description Fonctionnelle

Unité de base

- Le composant principal : un compteur 16 bits
- Mode décompteur/compteur - Auto chargement

TIMx STM32 : Description Fonctionnelle

Unité de base

- Le composant principal : un compteur 16 bits
- Mode décompteur/compteur - Auto chargement
- Prédiviseur de l'horloge du compteur

TIMx STM32 : Description Fonctionnelle

Unité de base

- Le composant principal : un compteur 16 bits
- Mode décompteur/compteur - Auto chargement
- Prédiviseur de l'horloge du compteur
- Les registres

TIMx STM32 : Description Fonctionnelle

Unité de base

- Le composant principal : un compteur 16 bits
- Mode décompteur/compteur - Auto chargement
- Prédiviseur de l'horloge du compteur
- Les registres
  - Registre du compteur : TIMx_CNT

TIMx STM32 : Description Fonctionnelle

Unité de base

- Le composant principal : un compteur 16 bits
- Mode décompteur/compteur - Auto chargement
- Prédiviseur de l'horloge du compteur
- Les registres
  - Registre du compteur : TIMx_CNT
  - Registre du prédiviseur : TIMx_PSC

TIMx STM32 : Description Fonctionnelle

Unité de base

- Le composant principal : un compteur 16 bits
- Mode décompteur/compteur - Auto chargement
- Prédiviseur de l'horloge du compteur
- Les registres
  - Registre du compteur : TIMx_CNT
  - Registre du prédiviseur : TIMx_PSC
  - Registre d'auto chargement : TIMx_ARR

TIMx STM32 : Description Fonctionnelle

Unité de base

- Le composant principal : un compteur 16 bits
- Mode décompteur/compteur - Auto chargement
- Prédiviseur de l'horloge du compteur
- Les registres
  - Registre du compteur : TIMx_CNT
  - Registre du prédiviseur : TIMx_PSC
  - Registre d'auto chargement : TIMx_ARR

Fonctionnement

TIMx STM32 : Description Fonctionnelle

Unité de base

- Le composant principal : un compteur 16 bits
- Mode décompteur/compteur - Auto chargement
- Prédiviseur de l'horloge du compteur
- Les registres
  - Registre du compteur : TIMx_CNT
  - Registre du prédiviseur : TIMx_PSC
  - Registre d'auto chargement : TIMx_ARR

Fonctionnement

- Le registre d'auto-chargement (ARR) est préchargé

TIMx STM32 : Description Fonctionnelle

Unité de base

- Le composant principal : un compteur 16 bits
- Mode décompteur/compteur - Auto chargement
- Prédiviseur de l'horloge du compteur
- Les registres
  - Registre du compteur : TIMx_CNT
  - Registre du prédiviseur : TIMx_PSC
  - Registre d'auto chargement : TIMx_ARR

Fonctionnement

- Le registre d'auto-chargement (ARR) est préchargé
- Lire ou écrire dans le registre d'auto-chargement revient à accéder au registre de préchargement

TIMx STM32 : Description Fonctionnelle

Unité de base

- Le composant principal : un compteur 16 bits
- Mode décompteur/compteur - Auto chargement
- Prédiviseur de l'horloge du compteur
- Les registres
  - Registre du compteur : TIMx_CNT
  - Registre du prédiviseur : TIMx_PSC
  - Registre d'auto chargement : TIMx_ARR

Fonctionnement

- Le registre d'auto-chargement (ARR) est préchargé
- Lire ou écrire dans le registre d'auto-chargement revient à accéder au registre de préchargement
- Le contenu du registre de préchargement est transférer dans le registre d'auto-chargement (Shadow) soit de façon permanente ou à chaque événement de mise à jour (bit UEV) en fonction du bit ARPE dans le registre CR1

TIMx STM32 : Description Fonctionnelle

Unité de base

- Le composant principal : un compteur 16 bits
- Mode décompteur/compteur - Auto chargement
- Prédiviseur de l'horloge du compteur
- Les registres
  - Registre du compteur : TIMx_CNT
  - Registre du prédiviseur : TIMx_PSC
  - Registre d'auto chargement : TIMx_ARR

Fonctionnement

- Le registre d'auto-chargement (ARR) est préchargé
- Lire ou écrire dans le registre d'auto-chargement revient à accéder au registre de préchargement
- Le contenu du registre de préchargement est transférer dans le registre d'auto-chargement (Shadow) soit de façon permanente ou à chaque événement de mise à jour (bit UEV) en fonction du bit ARPE dans le registre CR1
- L'événement de mise à jour est généré quand le compteur recycle (ou logiciellement) et que le bit UDIS du registre CR1 est à 0

TIMx STM32 : Description Fonctionnelle

Prédiviseur

TIMx STM32 : Description Fonctionnelle

Prédiviseur

- Le compteur est piloté par une horloge prédivisé CK_CNT

TIMx STM32 : Description Fonctionnelle

Prédiviseur

- Le compteur est piloté par une horloge prédivisé CK_CNT
- Cette horloge est activée uniquement si le bit CEN du registre CR1 est à 1

TIMx STM32 : Description Fonctionnelle

Prédiviseur

- Le compteur est piloté par une horloge prédivisé CK_CNT
- Cette horloge est activée uniquement si le bit CEN du registre CR1 est à 1
- Remarque : le signal CNT_EN relié à CK_CNT passe à 1 après 1 cycle d'horloge suite à la mise à 1 du bit CEN.

TIMx STM32 : Description Fonctionnelle

Prédiviseur

- Le compteur est piloté par une horloge prédivisé CK_CNT
- Cette horloge est activée uniquement si le bit CEN du registre CR1 est à 1
- Remarque : le signal CNT_EN relié à CK_CNT passe à 1 après 1 cycle d'horloge suite à la mise à 1 du bit CEN.
- Il peut diviser l'horloge d'un facteur compris entre 1 et 65536

TIMx STM32 : Description Fonctionnelle

Prédiviseur

- Le compteur est piloté par une horloge prédivisé CK_CNT
- Cette horloge est activée uniquement si le bit CEN du registre CR1 est à 1
- Remarque : le signal CNT_EN relié à CK_CNT passe à 1 après 1 cycle d'horloge suite à la mise à 1 du bit CEN.
- Il peut diviser l'horloge d'un facteur compris entre 1 et 65536
- C'est un compteur 16 bits contrôler par le registre 16 bits PSC

TIMx STM32 : Description Fonctionnelle

Prédiviseur

- Le compteur est piloté par une horloge prédivisé CK_CNT
- Cette horloge est activée uniquement si le bit CEN du registre CR1 est à 1
- Remarque : le signal CNT_EN relié à CK_CNT passe à 1 après 1 cycle d'horloge suite à la mise à 1 du bit CEN.
- Il peut diviser l'horloge d'un facteur compris entre 1 et 65536
- C'est un compteur 16 bits contrôler par le registre 16 bits PSC
- Sa valeur peut être changée au fil de l'eau elle sera alors prise en compte au prochain événement de mise à jour

TIMx STM32 : Description Fonctionnelle

Prédiviseur

- Le compteur est piloté par une horloge prédivisé CK_CNT
- Cette horloge est activée uniquement si le bit CEN du registre CR1 est à 1
- Remarque : le signal CNT_EN relié à CK_CNT passe à 1 après 1 cycle d'horloge suite à la mise à 1 du bit CEN.
- Il peut diviser l'horloge d'un facteur compris entre 1 et 65536
- C'est un compteur 16 bits contrôler par le registre 16 bits PSC
- Sa valeur peut être changée au fil de l'eau elle sera alors prise en compte au prochain événement de mise à jour

Mode compteur

TIMx STM32 : Description Fonctionnelle

Prédiviseur

- Le compteur est piloté par une horloge prédivisé CK_CNT
- Cette horloge est activée uniquement si le bit CEN du registre CR1 est à 1
- Remarque : le signal CNT_EN relié à CK_CNT passe à 1 après 1 cycle d'horloge suite à la mise à 1 du bit CEN.
- Il peut diviser l'horloge d'un facteur compris entre 1 et 65536
- C'est un compteur 16 bits contrôler par le registre 16 bits PSC
- Sa valeur peut être changée au fil de l'eau elle sera alors prise en compte au prochain événement de mise à jour

Mode compteur

- Bit CMS = 00 et bit DIR = 0 (registre CR1)

TIMx STM32 : Description Fonctionnelle

Prédiviseur

- Le compteur est piloté par une horloge prédivisé CK_CNT
- Cette horloge est activée uniquement si le bit CEN du registre CR1 est à 1
- Remarque : le signal CNT_EN relié à CK_CNT passe à 1 après 1 cycle d'horloge suite à la mise à 1 du bit CEN.
- Il peut diviser l'horloge d'un facteur compris entre 1 et 65536
- C'est un compteur 16 bits contrôler par le registre 16 bits PSC
- Sa valeur peut être changée au fil de l'eau elle sera alors prise en compte au prochain événement de mise à jour

Mode compteur

- Bit CMS = 00 et bit DIR = 0 (registre CR1)
- Dans ce mode le compteur compte de 0 à la valeur du registre d'auto-chargement (registre ARR)

TIMx STM32 : Description Fonctionnelle

Prédiviseur

- Le compteur est piloté par une horloge prédivisé CK_CNT
- Cette horloge est activée uniquement si le bit CEN du registre CR1 est à 1
- Remarque : le signal CNT_EN relié à CK_CNT passe à 1 après 1 cycle d'horloge suite à la mise à 1 du bit CEN.
- Il peut diviser l'horloge d'un facteur compris entre 1 et 65536
- C'est un compteur 16 bits contrôler par le registre 16 bits PSC
- Sa valeur peut être changée au fil de l'eau elle sera alors prise en compte au prochain événement de mise à jour

Mode compteur

- Bit CMS = 00 et bit DIR = 0 (registre CR1)
- Dans ce mode le compteur compte de 0 à la valeur du registre d'auto-chargement (registre ARR)
- Arrivée à la valeur du registre ARR il recycle en redémarrant un cycle de comptage de 0

TIMx STM32 : Description Fonctionnelle

Prédiviseur

- Le compteur est piloté par une horloge prédivisé CK_CNT
- Cette horloge est activée uniquement si le bit CEN du registre CR1 est à 1
- Remarque : le signal CNT_EN relié à CK_CNT passe à 1 après 1 cycle d'horloge suite à la mise à 1 du bit CEN.
- Il peut diviser l'horloge d'un facteur compris entre 1 et 65536
- C'est un compteur 16 bits contrôler par le registre 16 bits PSC
- Sa valeur peut être changée au fil de l'eau elle sera alors prise en compte au prochain événement de mise à jour

Mode compteur

- Bit CMS = 00 et bit DIR = 0 (registre CR1)
- Dans ce mode le compteur compte de 0 à la valeur du registre d'auto-chargement (registre ARR)
- Arrivée à la valeur du registre ARR il recycle en redémarrant un cycle de comptage de 0
- Au recyclage un événement de dépassement de capacité est généré (Overflow)

TIMx STM32 : Description Fonctionnelle

Prédiviseur

- Le compteur est piloté par une horloge prédivisé CK_CNT
- Cette horloge est activée uniquement si le bit CEN du registre CR1 est à 1
- Remarque : le signal CNT_EN relié à CK_CNT passe à 1 après 1 cycle d'horloge suite à la mise à 1 du bit CEN.
- Il peut diviser l'horloge d'un facteur compris entre 1 et 65536
- C'est un compteur 16 bits contrôler par le registre 16 bits PSC
- Sa valeur peut être changée au fil de l'eau elle sera alors prise en compte au prochain événement de mise à jour

Mode compteur

- Bit CMS = 00 et bit DIR = 0 (registre CR1)
- Dans ce mode le compteur compte de 0 à la valeur du registre d'auto-chargement (registre ARR)
- Arrivée à la valeur du registre ARR il recycle en redémarrant un cycle de comptage de 0
- Au recyclage un événement de dépassement de capacité est généré (Overflow)
- Un événement de mise à jour peut être généré à chaque recyclage ou en positionnant le bit UG du registre EGR

TIMx STM32 : Description Fonctionnelle

Mode Compteur

TIMx STM32 : Description Fonctionnelle

Mode Compteur

- L'événement de mise à jour peut être désactivé via le bit UDIS du registre CR1

TIMx STM32 : Description Fonctionnelle

Mode Compteur

- L'événement de mise à jour peut être désactivé via le bit UDIS du registre CR1
- Si le bit URS (registre CR1) est à 1, positionner le bit UG génére un événement de mise à jour sans générer d'interruption

TIMx STM32 : Description Fonctionnelle

Mode Compteur

- L'événement de mise à jour peut être désactivé via le bit UDIS du registre CR1
- Si le bit URS (registre CR1) est à 1, positionner le bit UG génére un événement de mise à jour sans générer d'interruption
- Quand un événement de mise à jour survient, tous les registres sont mise à jour et le bit d'interruption IUF du registre SR est mis à 1 en fonction de la valeur du bit URS.

TIMx STM32 : Description Fonctionnelle

Mode Compteur

- L'événement de mise à jour peut être désactivé via le bit UDIS du registre CR1
- Si le bit URS (registre CR1) est à 1, positionner le bit UG génére un événement de mise à jour sans générer d'interruption
- Quand un événement de mise à jour survient, tous les registres sont mise à jour et le bit d'interruption IUF du registre SR est mis à 1 en fonction de la valeur du bit URS.

Mode compteur - Mise à jour

TIMx STM32 : Description Fonctionnelle

Mode Compteur

- L'événement de mise à jour peut être désactivé via le bit UDIS du registre CR1
- Si le bit URS (registre CR1) est à 1, positionner le bit UG génére un événement de mise à jour sans générer d'interruption
- Quand un événement de mise à jour survient, tous les registres sont mise à jour et le bit d'interruption IUF du registre SR est mis à 1 en fonction de la valeur du bit URS.

Mode compteur - Mise à jour

- Le registre tampon du prédiviseur est rechargé avec la valeur du registre PSC

TIMx STM32 : Description Fonctionnelle

Mode Compteur

- L'événement de mise à jour peut être désactivé via le bit UDIS du registre CR1
- Si le bit URS (registre CR1) est à 1, positionner le bit UG génére un événement de mise à jour sans générer d'interruption
- Quand un événement de mise à jour survient, tous les registres sont mise à jour et le bit d'interruption IUF du registre SR est mis à 1 en fonction de la valeur du bit URS.

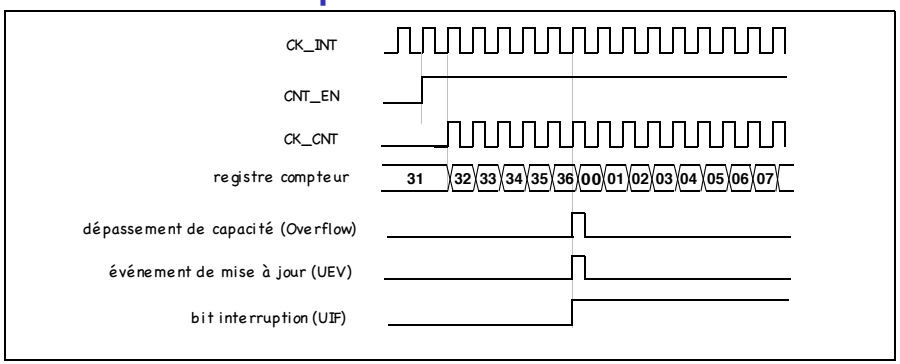
Mode compteur - Mise à jour

- Le registre tampon du prédiviseur est rechargé avec la valeur du registre PSC
- Le registre d'auto-chargement est rechargé avec la valeur du registre ARR

TIMx STM32 : Description Fonctionnelle

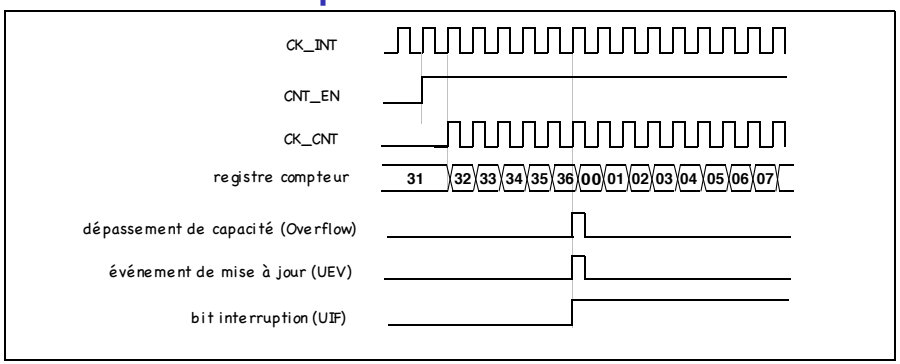
TIMx STM32 : Description Fonctionnelle

Mode Compteur
Prédivision par 1

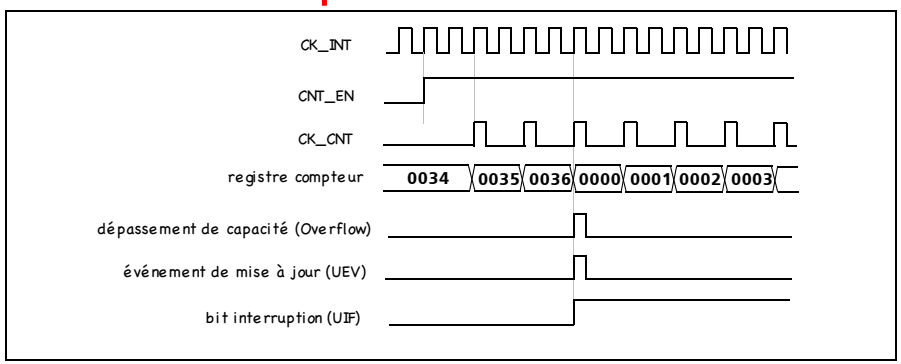


TIMx STM32 : Description Fonctionnelle

Mode Compteur
Prédivision par 1



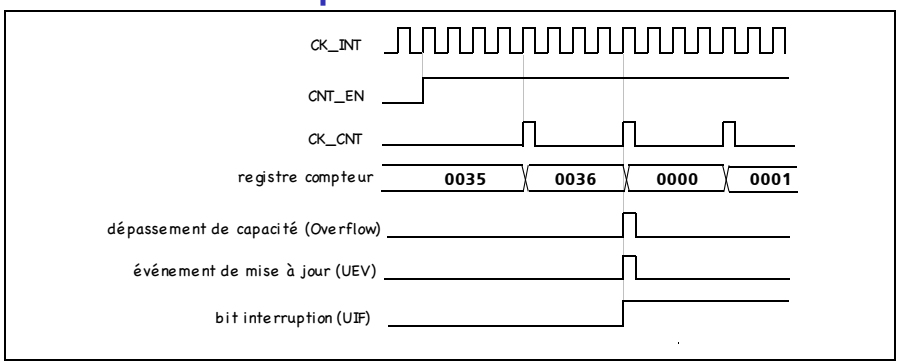
Mode compteur
Prédivision par 2



TIMx STM32 : Description Fonctionnelle

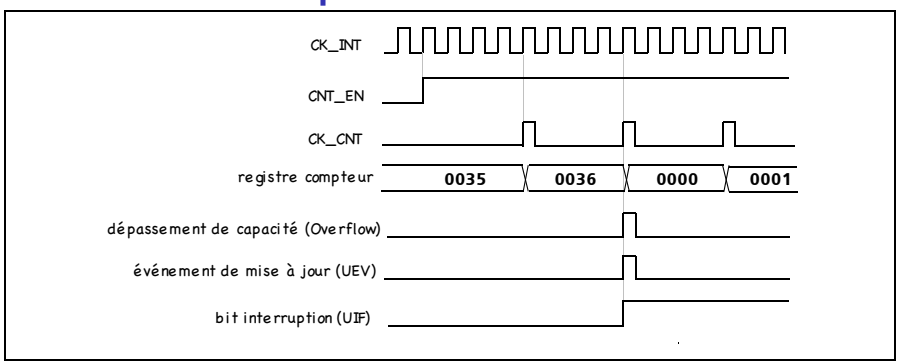
TIMx STM32 : Description Fonctionnelle

Mode Compteur
Prédivision par 4

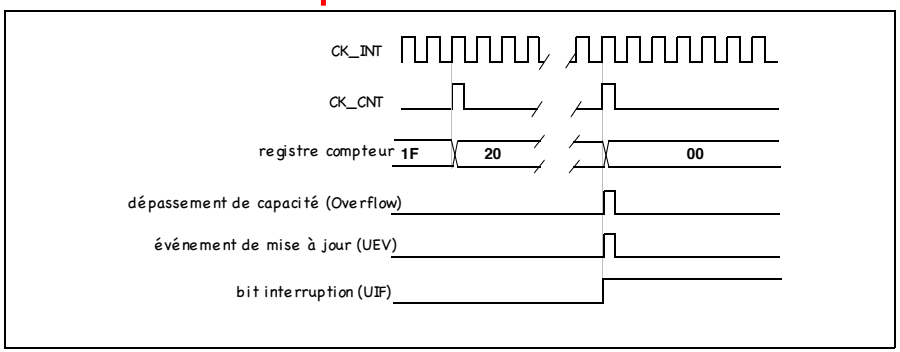


TIMx STM32 : Description Fonctionnelle

Mode Compteur
Prédivision par 4



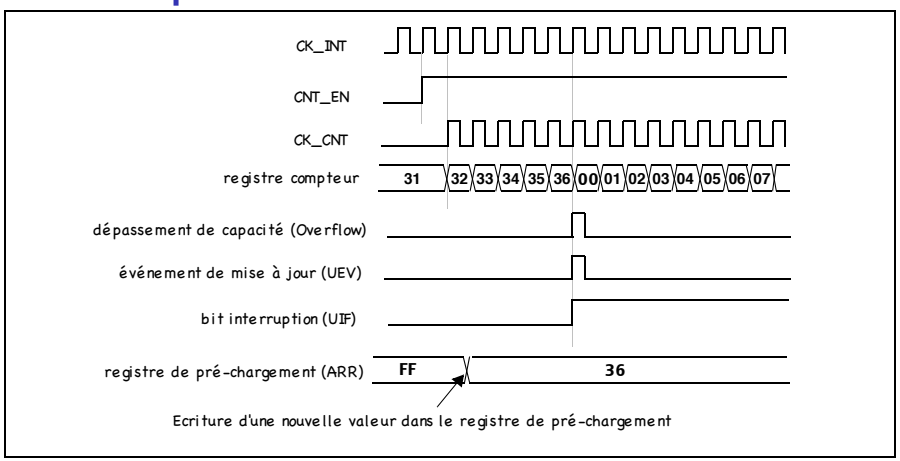
Mode compteur
Prédivision par N



TIMx STM32 : Description Fonctionnelle

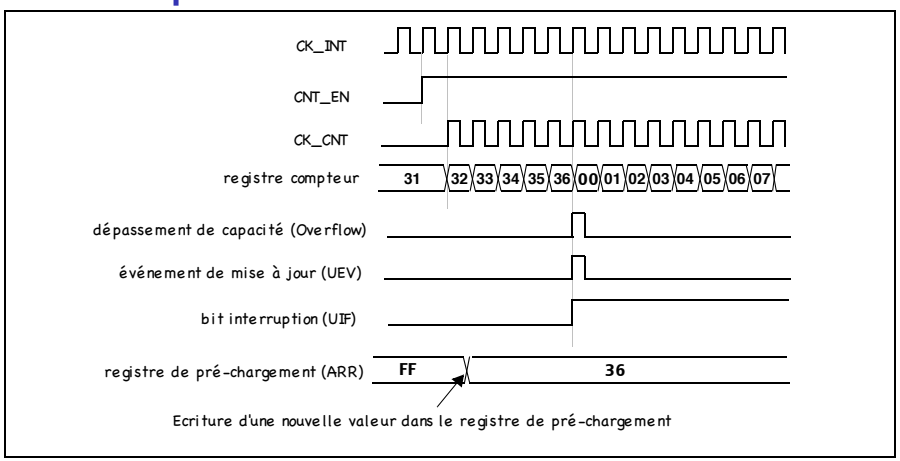
TIMx STM32 : Description Fonctionnelle

Mode Compteur
UEV quand ARPE=0

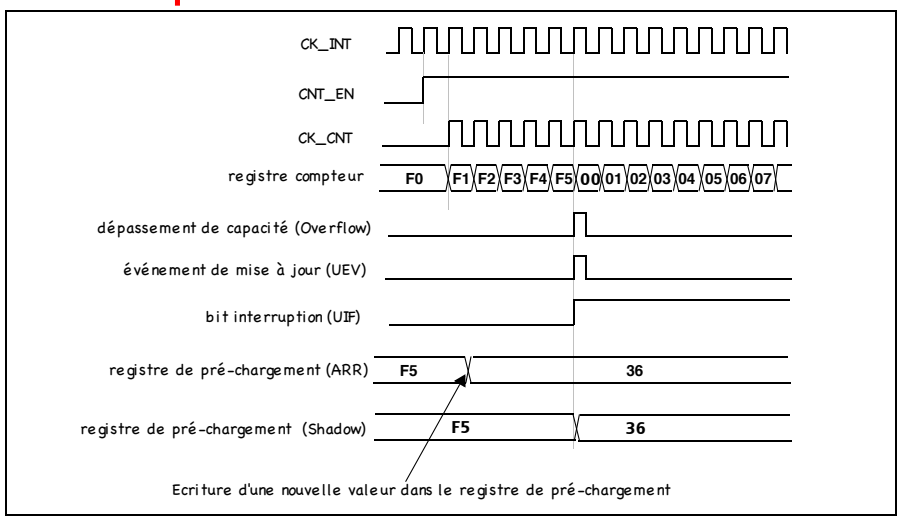


TIMx STM32 : Description Fonctionnelle

Mode Compteur
UEV quand ARPE=0



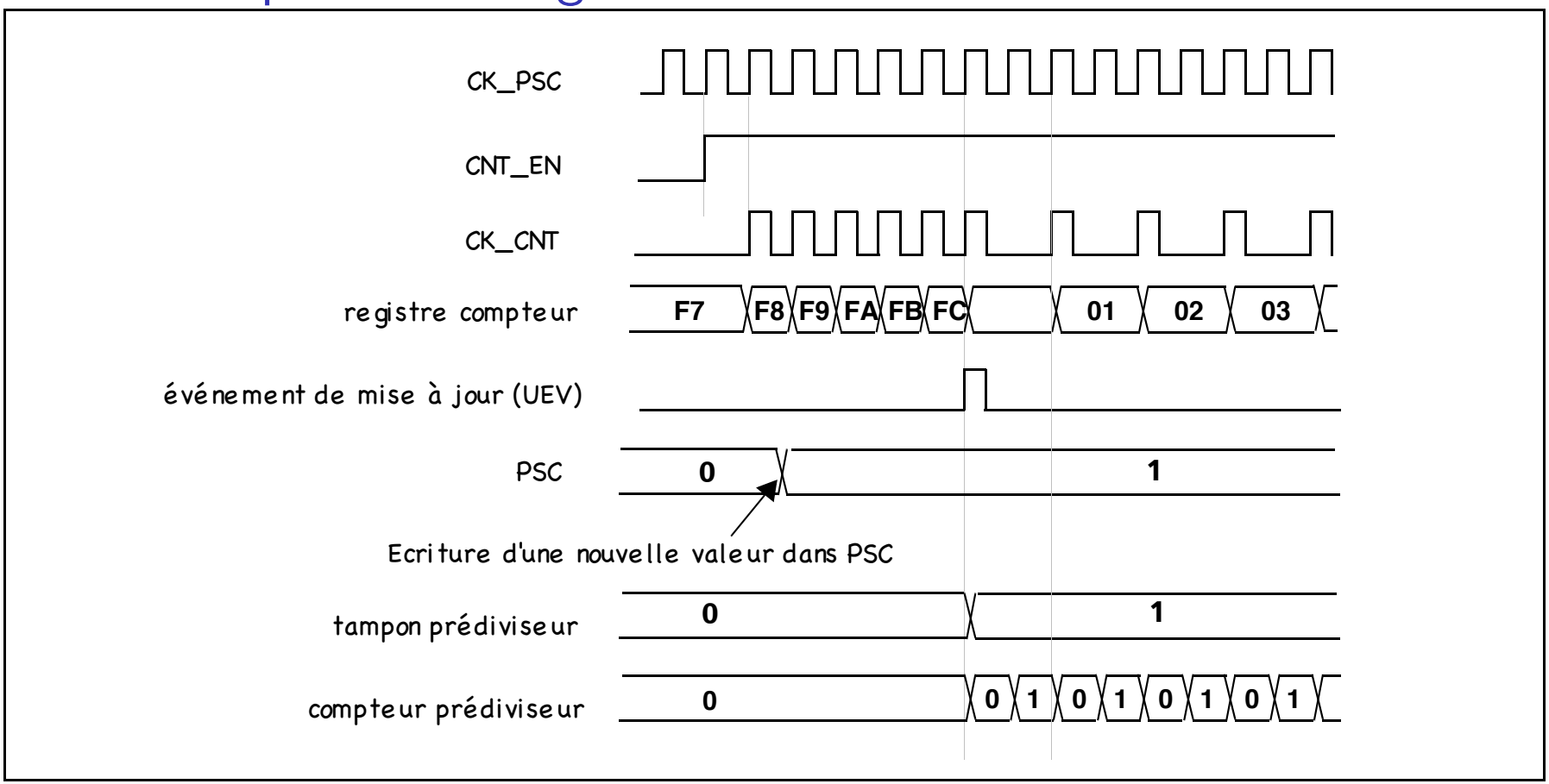
Mode compteur
UEV quand ARPE=1



TIMx STM32 : Description Fonctionnelle

TIMx STM32 : Description Fonctionnelle

Mode Compteur - Changement Prédivision de 1 à 2



TIMx STM32 : Description Fonctionnelle

Mode décompteur

TIMx STM32 : Description Fonctionnelle

Mode décompteur

- Bit CMS = 00 et bit DIR = 0 (registre CR1)

TIMx STM32 : Description Fonctionnelle

Mode décompteur

- Bit CMS = 00 et bit DIR = 0 (registre CR1)
- Idem mode compteur sauf que comptage de la valeur du registre de pré-chargement à 0

TIMx STM32 : Description Fonctionnelle

Mode décompteur

- Bit CMS = 00 et bit DIR = 0 (registre CR1)
- Idem mode compteur sauf que comptage de la valeur du registre de pré-chargement à 0

Mode Alterné

TIMx STM32 : Description Fonctionnelle

Mode décompteur

- Bit CMS = 00 et bit DIR = 0 (registre CR1)
- Idem mode compteur sauf que comptage de la valeur du registre de pré-chargement à 0

Mode Alterné

- Bit CMS = 01, 10 ou 11 (registre CR1)

TIMx STM32 : Description Fonctionnelle

Mode décompteur

- Bit CMS = 00 et bit DIR = 0 (registre CR1)
- Idem mode compteur sauf que comptage de la valeur du registre de pré-chargement à 0

Mode Alterné

- Bit CMS = 01, 10 ou 11 (registre CR1)
- Dans un premier temps mode compteur de 0 à valeur d'auto-chargement

TIMx STM32 : Description Fonctionnelle

Mode décompteur

- Bit CMS = 00 et bit DIR = 0 (registre CR1)
- Idem mode compteur sauf que comptage de la valeur du registre de pré-chargement à 0

Mode Alterné

- Bit CMS = 01, 10 ou 11 (registre CR1)
- Dans un premier temps mode compteur de 0 à valeur d'auto-chargement
- Puis recyclage

TIMx STM32 : Description Fonctionnelle

Mode décompteur

- Bit CMS = 00 et bit DIR = 0 (registre CR1)
- Idem mode compteur sauf que comptage de la valeur du registre de pré-chargement à 0

Mode Alterné

- Bit CMS = 01, 10 ou 11 (registre CR1)
- Dans un premier temps mode compteur de 0 à valeur d'auto-chargement
- Puis recyclage
- Dans un second temps mode décompteur de valeur d'auto-chargement à 0

TIMx STM32 : Description Fonctionnelle

Mode décompteur

- Bit CMS = 00 et bit DIR = 0 (registre CR1)
- Idem mode compteur sauf que comptage de la valeur du registre de pré-chargement à 0

Mode Alterné

- Bit CMS = 01, 10 ou 11 (registre CR1)
- Dans un premier temps mode compteur de 0 à valeur d'auto-chargement
- Puis recyclage
- Dans un second temps mode décompteur de valeur d'auto-chargement à 0
- Puis recyclage

TIMx STM32 : Description Fonctionnelle

Mode décompteur

- Bit CMS = 00 et bit DIR = 0 (registre CR1)
- Idem mode compteur sauf que comptage de la valeur du registre de pré-chargement à 0

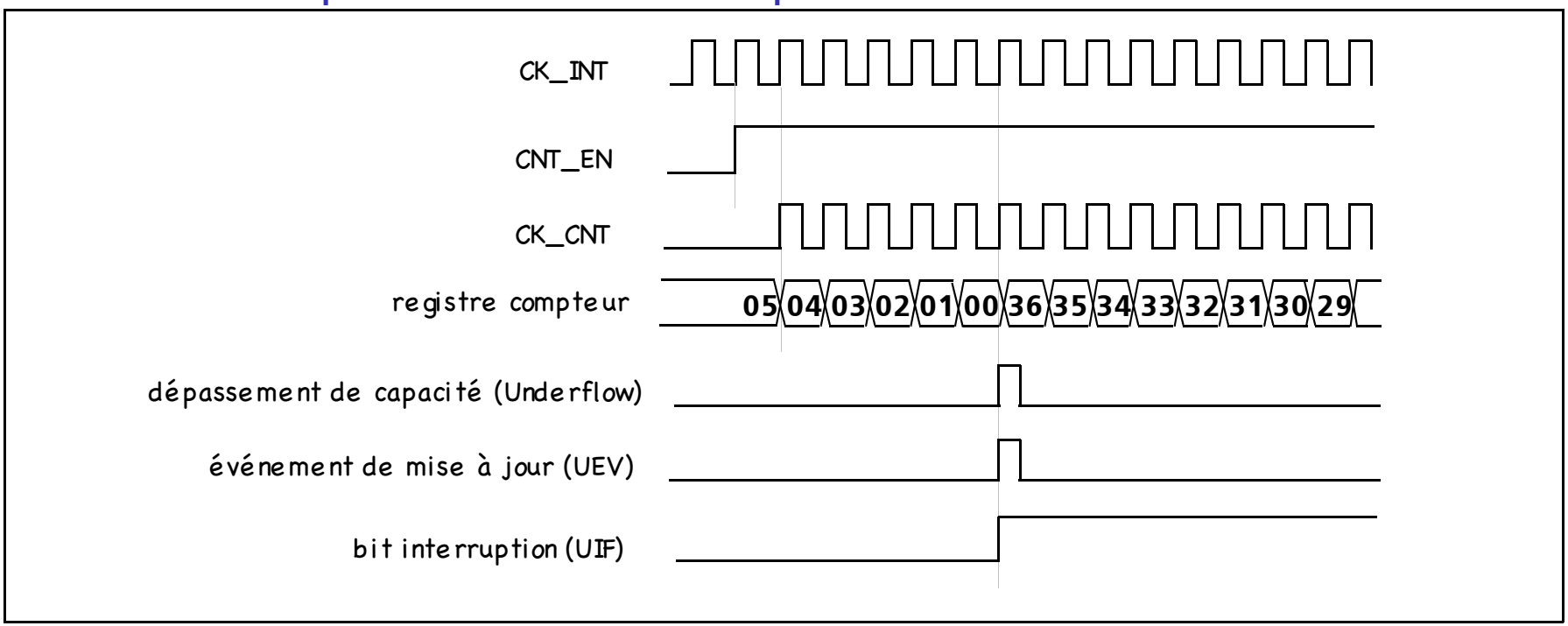
Mode Alterné

- Bit CMS = 01, 10 ou 11 (registre CR1)
- Dans un premier temps mode compteur de 0 à valeur d'auto-chargement
- Puis recyclage
- Dans un second temps mode décompteur de valeur d'auto-chargement à 0
- Puis recyclage
- et reprise en mode compteur et ainsi de suite

TIMx STM32 : Description Fonctionnelle

TIMx STM32 : Description Fonctionnelle

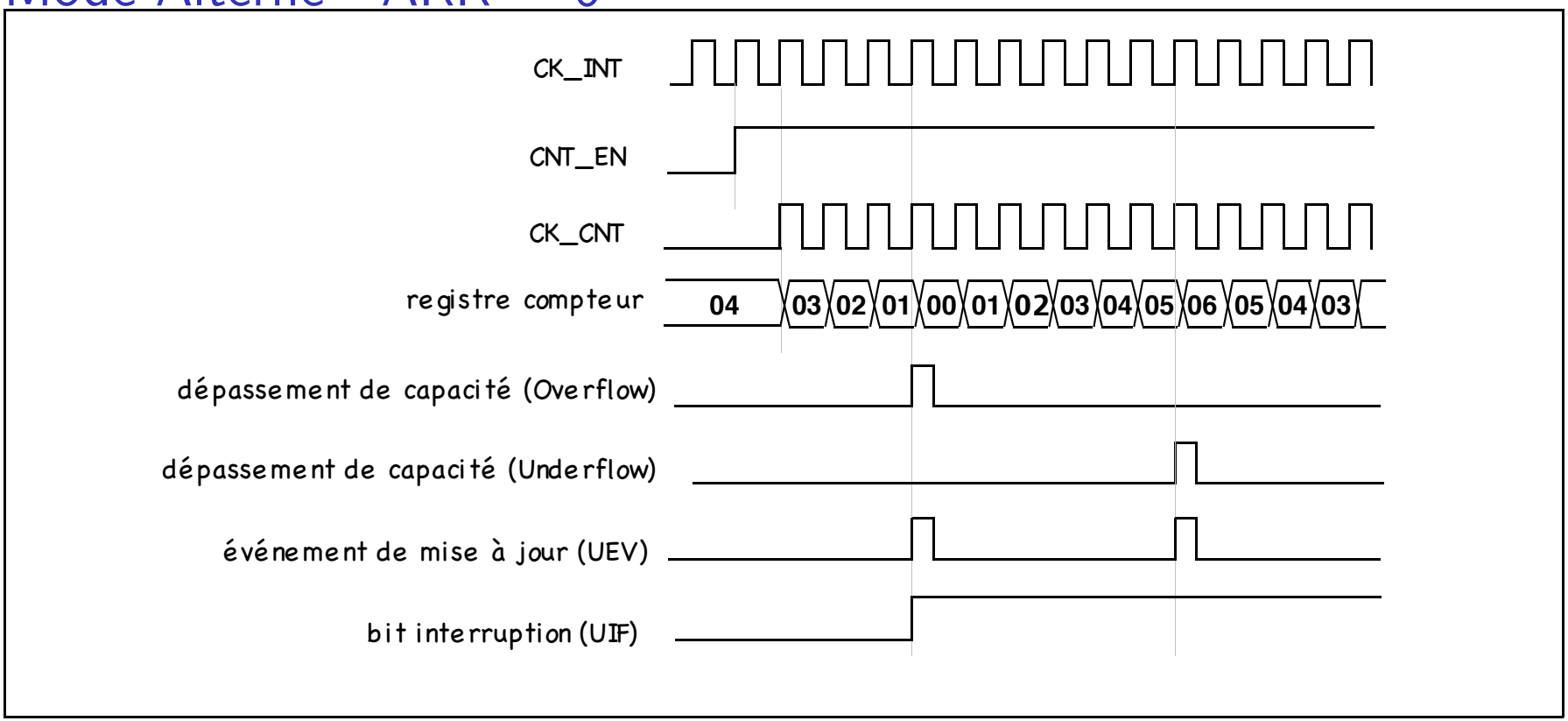
Mode Décompteur - Prédivision par 1



TIMx STM32 : Description Fonctionnelle

TIMx STM32 : Description Fonctionnelle

Mode Alterné - ARR = 6



TIMx STM32 : Description Fonctionnelle

Sélection d'horloge

TIMx STM32 : Description Fonctionnelle

Sélection d'horloge

- L'horloge du compteur peut provenir de

TIMx STM32 : Description Fonctionnelle

Sélection d'horloge

- L'horloge du compteur peut provenir de
  - L'horloge interne par défaut CK_INT qui est générée par :
    - HSI par défaut Oscillateur 8 MHz (bits SW = 00 registre CFGR unité CRR)
    - HSE oscillateur externe (bits SW = 01 registre CFGR unité CRR)
    - PLL (bits SW = 10 registre CFGR unité CRR)

TIMx STM32 : Description Fonctionnelle

Sélection d'horloge

- L'horloge du compteur peut provenir de
  - L'horloge interne par défaut CK_INT qui est générée par :
    - HSI par défaut Oscillateur 8 MHz (bits SW = 00 registre CFGR unité CRR)
    - HSE oscillateur externe (bits SW = 01 registre CFGR unité CRR)
    - PLL (bits SW = 10 registre CFGR unité CRR)
  - Une première horloge externe sur une broche TIx

TIMx STM32 : Description Fonctionnelle

Sélection d'horloge

- L'horloge du compteur peut provenir de
  - L'horloge interne par défaut CK_INT qui est générée par :
    - HSI par défaut Oscillateur 8 MHz (bits SW = 00 registre CFGR unité CRR)
    - HSE oscillateur externe (bits SW = 01 registre CFGR unité CRR)
    - PLL (bits SW = 10 registre CFGR unité CRR)
  - Une première horloge externe sur une broche TIx

Sélection d'horloge

TIMx STM32 : Description Fonctionnelle

Sélection d'horloge

- L'horloge du compteur peut provenir de
  - L'horloge interne par défaut CK_INT qui est générée par :
    - HSI par défaut Oscillateur 8 MHz (bits SW = 00 registre CFGR unité CRR)
    - HSE oscillateur externe (bits SW = 01 registre CFGR unité CRR)
    - PLL (bits SW = 10 registre CFGR unité CRR)
  - Une première horloge externe sur une broche TIx

Sélection d'horloge

- Une seconde horloge externe sur une entrée de déclenchement ETR

TIMx STM32 : Description Fonctionnelle

Sélection d'horloge

- L'horloge du compteur peut provenir de
  - L'horloge interne par défaut CK_INT qui est générée par :
    - HSI par défaut Oscillateur 8 MHz (bits SW = 00 registre CFGR unité CRR)
    - HSE oscillateur externe (bits SW = 01 registre CFGR unité CRR)
    - PLL (bits SW = 10 registre CFGR unité CRR)
  - Une première horloge externe sur une broche TIx

Sélection d'horloge

- Une seconde horloge externe sur une entrée de déclenchement ETR
- Un signal de déclenchement interne (ITRx) utilisant un timer comme prédiviseur pour un autre timer. Par exemple le Timer1 sert de prédiviseur au Timer2.

TIMx STM32 : Description Fonctionnelle

Sélection d'horloge

- L'horloge du compteur peut provenir de
  - L'horloge interne par défaut CK_INT qui est générée par :
    - HSI par défaut Oscillateur 8 MHz (bits SW = 00 registre CFGR unité CRR)
    - HSE oscillateur externe (bits SW = 01 registre CFGR unité CRR)
    - PLL (bits SW = 10 registre CFGR unité CRR)
  - Une première horloge externe sur une broche TIx

Sélection d'horloge

- Une seconde horloge externe sur une entrée de déclenchement ETR
- Un signal de déclenchement interne (ITRx) utilisant un timer comme prédiviseur pour un autre timer. Par exemple le Timer1 sert de prédiviseur au Timer2.
- Sélection de CK_INT quand le champs SMS=000 dans le registre SMCR

TIMx STM32 : Description Fonctionnelle

Sélection d'horloge

- L'horloge du compteur peut provenir de
  - L'horloge interne par défaut CK_INT qui est générée par :
    - HSI par défaut Oscillateur 8 MHz (bits SW = 00 registre CFGR unité CRR)
    - HSE oscillateur externe (bits SW = 01 registre CFGR unité CRR)
    - PLL (bits SW = 10 registre CFGR unité CRR)
  - Une première horloge externe sur une broche TIx

Sélection d'horloge

- Une seconde horloge externe sur une entrée de déclenchement ETR
- Un signal de déclenchement interne (ITRx) utilisant un timer comme prédiviseur pour un autre timer. Par exemple le Timer1 sert de prédiviseur au Timer2.
- Sélection de CK_INT quand le champs SMS=000 dans le registre SMCR
- Sélection première horloge externe quand SMS=111

TIMx STM32 : Description Fonctionnelle

Sélection d'horloge

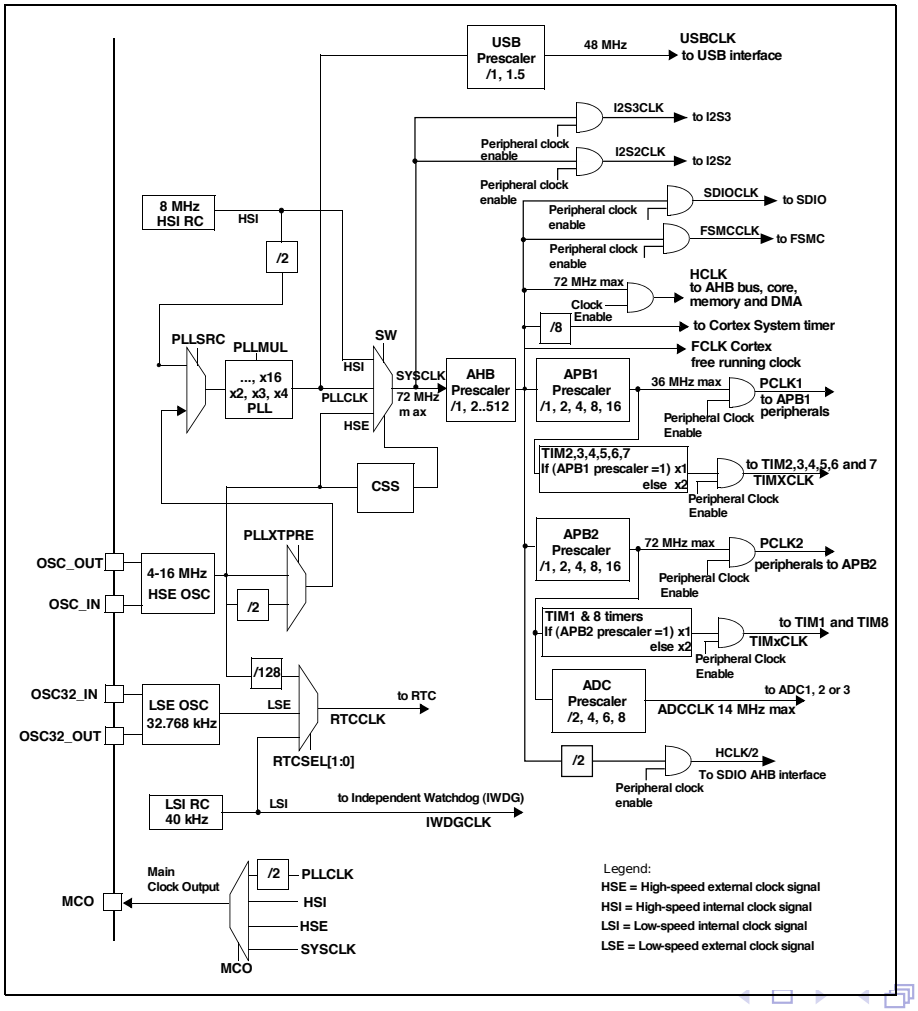
- L'horloge du compteur peut provenir de
  - L'horloge interne par défaut CK_INT qui est générée par :
    - HSI par défaut Oscillateur 8 MHz (bits SW = 00 registre CFGR unité CRR)
    - HSE oscillateur externe (bits SW = 01 registre CFGR unité CRR)
    - PLL (bits SW = 10 registre CFGR unité CRR)
  - Une première horloge externe sur une broche TIx

Sélection d'horloge

- Une seconde horloge externe sur une entrée de déclenchement ETR
- Un signal de déclenchement interne (ITRx) utilisant un timer comme prédiviseur pour un autre timer. Par exemple le Timer1 sert de prédiviseur au Timer2.
- Sélection de CK_INT quand le champs SMS=000 dans le registre SMCR
- Sélection première horloge externe quand SMS=111
- Sélection seconde horloge externe quand ECE=1 dans le registre SMCR

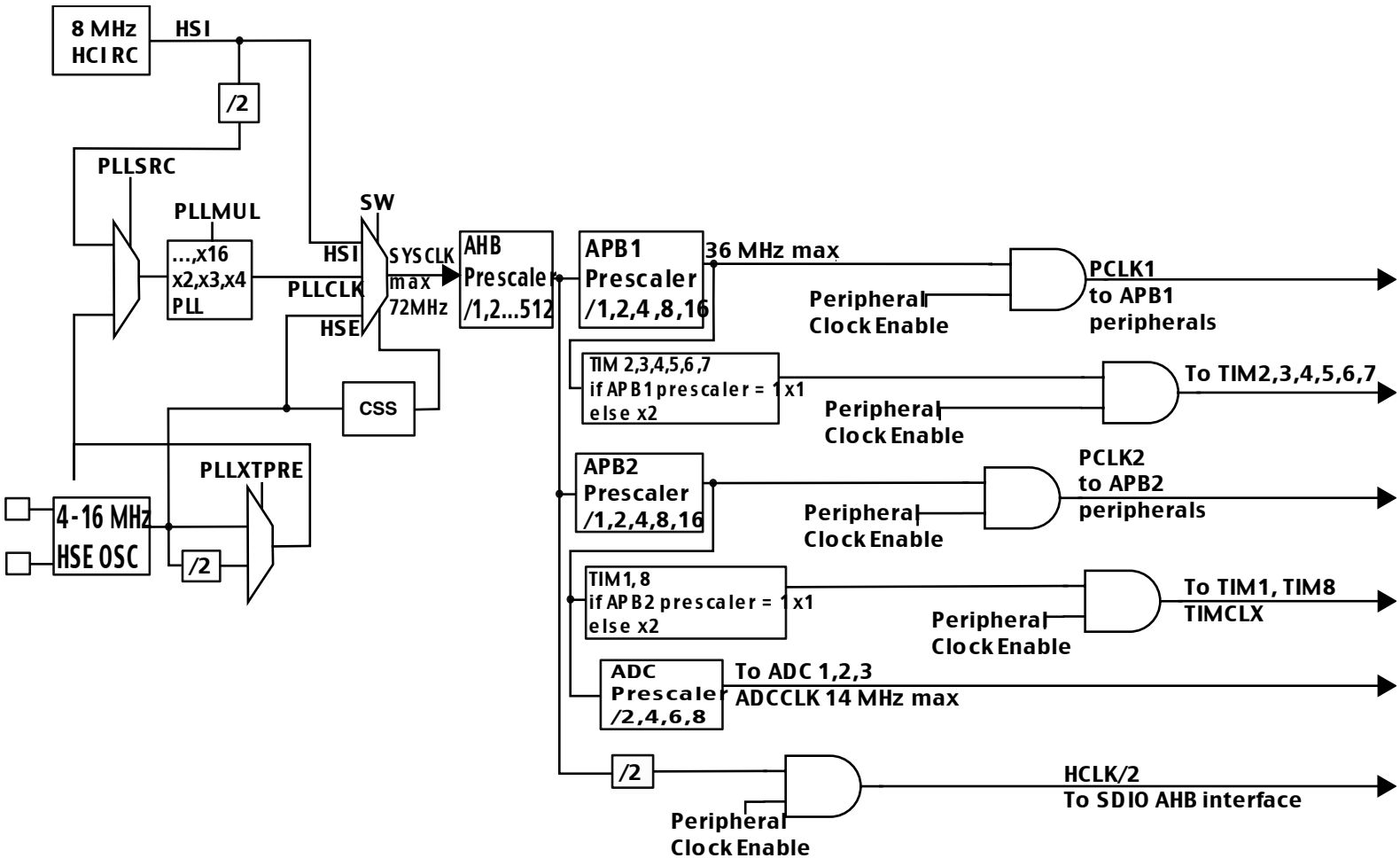
TIMx STM32 : Description Fonctionnelle

Arbre d'Horloge du STM32



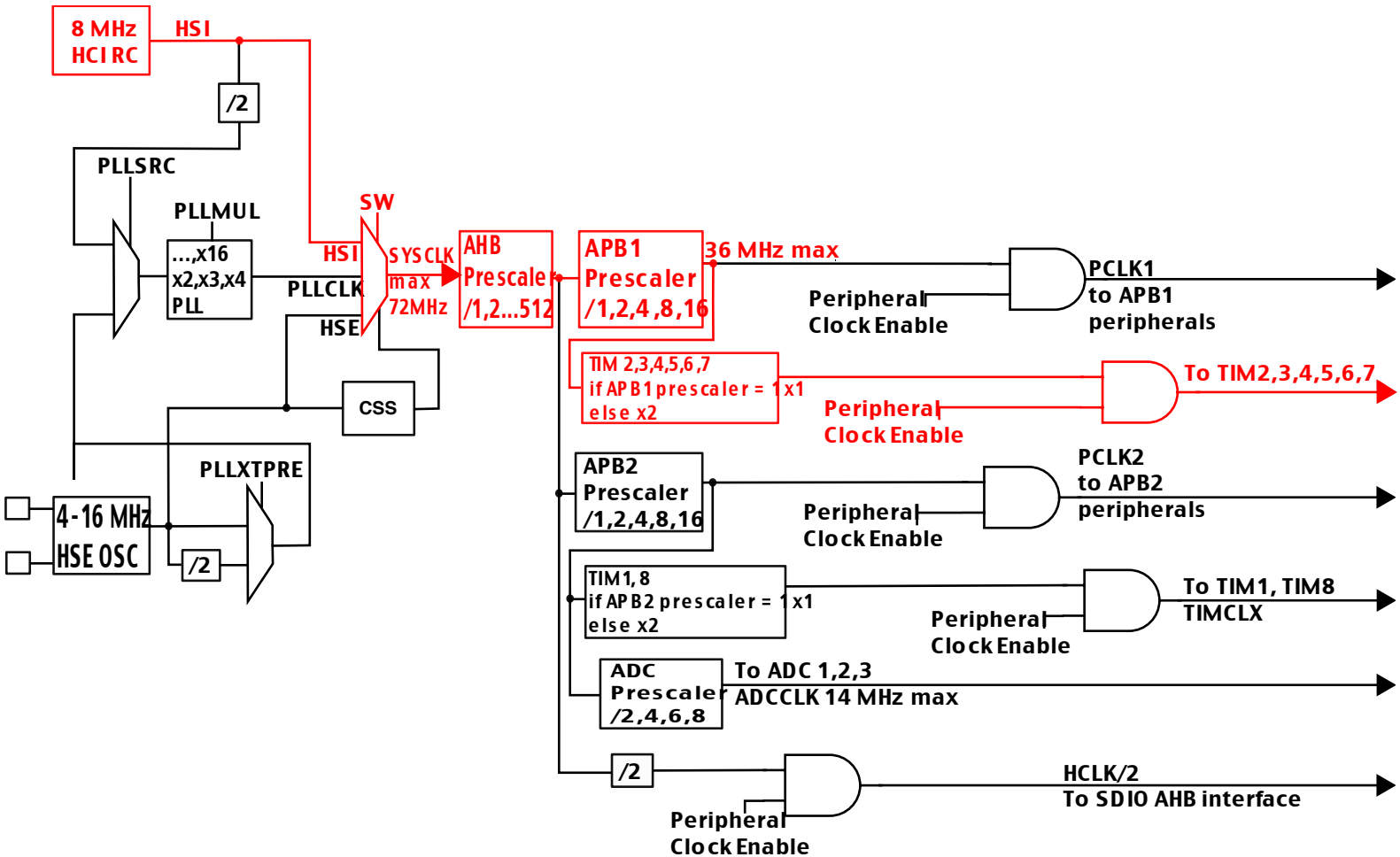
TIMx STM32 : Description Fonctionnelle

Arbre d'Horloge du bus APB1



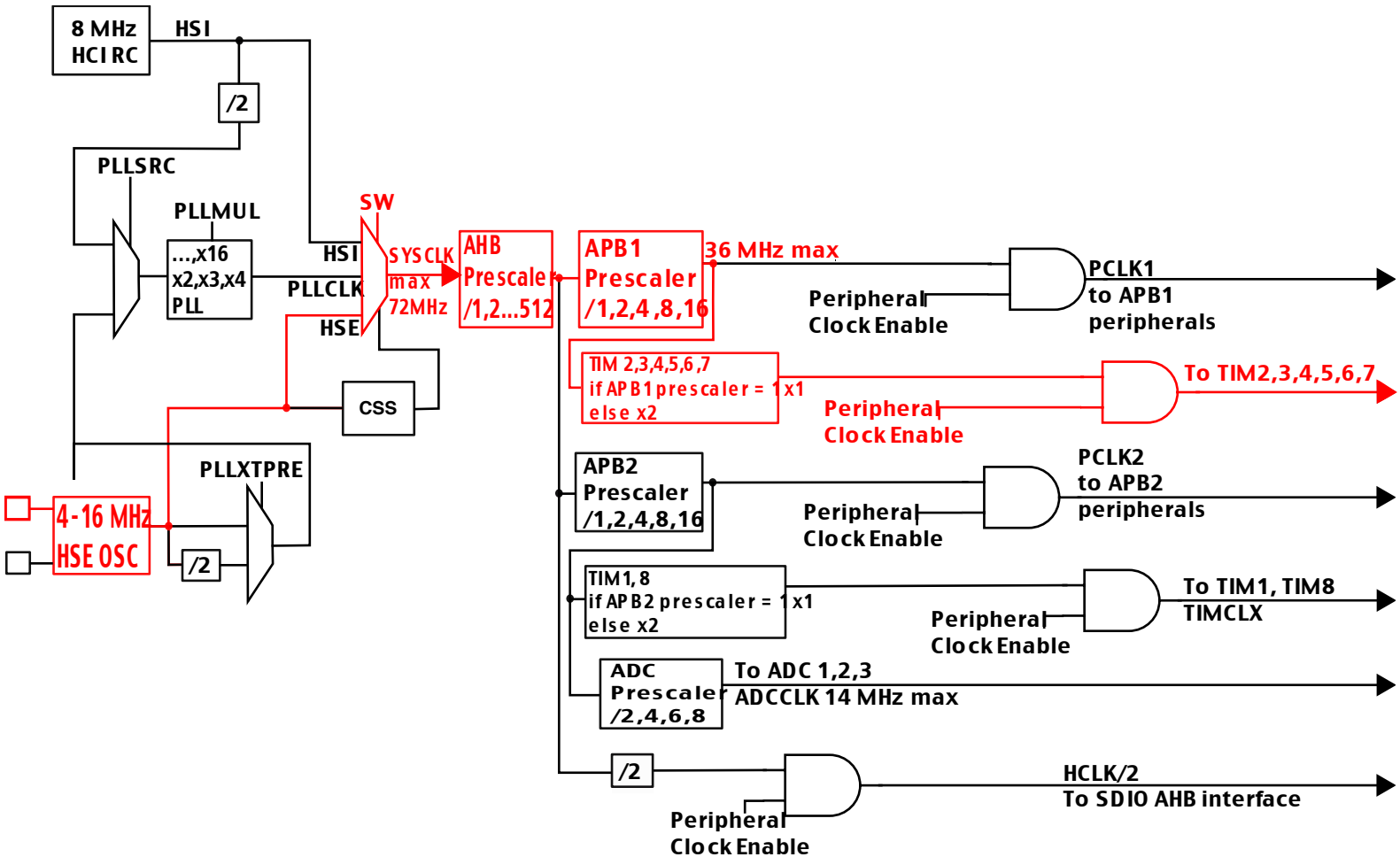
TIMx STM32 : Description Fonctionnelle

Arbre d'Horloge des Timerx - HSI - Horloge par Défaut



TIMx STM32 : Description Fonctionnelle

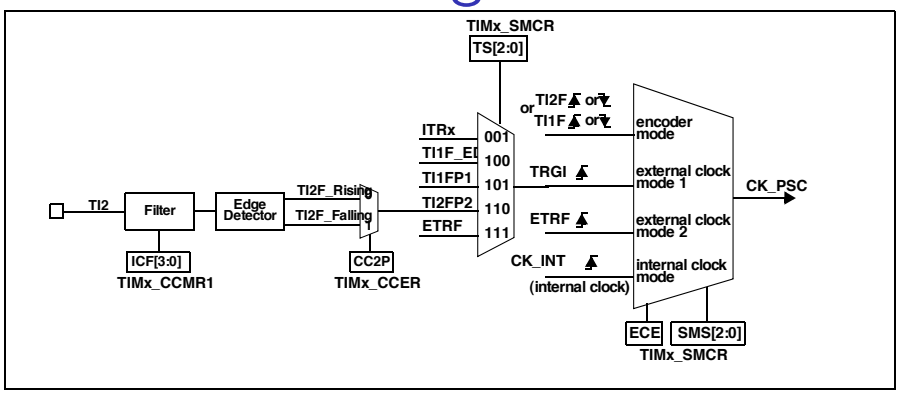
Arbre d'Horloge des Timerx - HSE



TIMx STM32 : Description Fonctionnelle

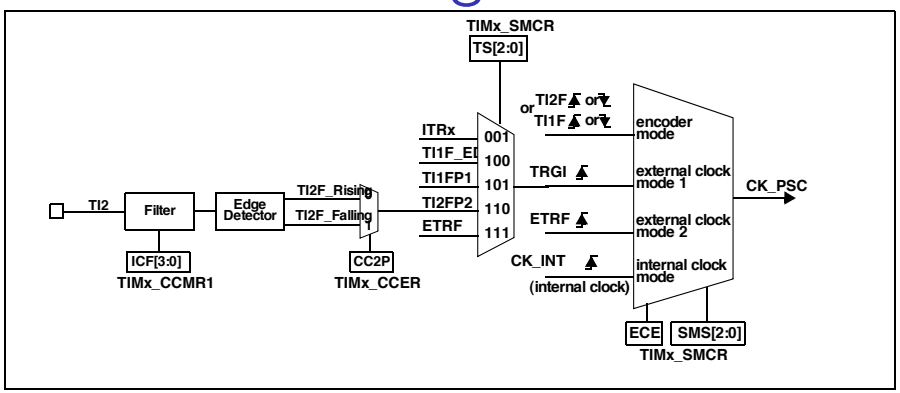
TIMx STM32 : Description Fonctionnelle

Sélection d'horloge

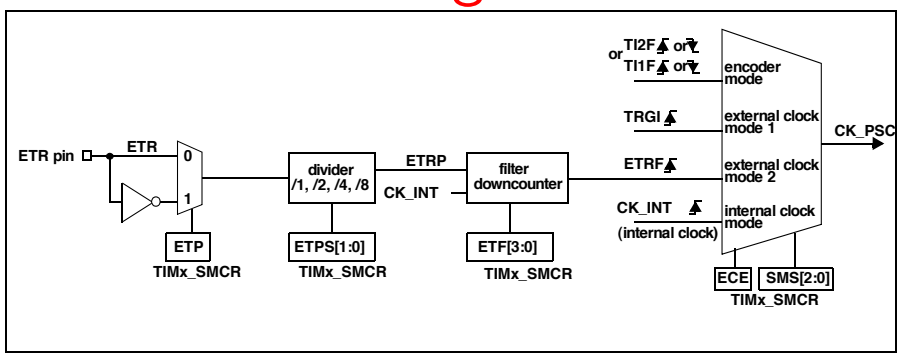


TIMx STM32 : Description Fonctionnelle

Sélection d'horloge



Sélection d'horloge



TIMx STM32 : Registre Horloge RCC

CFGR, Adresse Base RCC = 0x4002 1000, Offset = 0x04, Valeur Initiale = 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					MCO[2:0]			Res.	USB PRE	PLLMUL[3:0]				PLL XTPRE	PLL SRC
					rw	rw	rw		rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADC PRE[1:0]		PPRE2[2:0]			PPRE1[2:0]			HPRE[3:0]				SWS[1:0]		SW[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	r	r	rw	rw

TIMx STM32 : Registre Horloge RCC

APB1ENR, Adresse Base RCC = 0x4002 1000, Offset = 0x1C, Valeur Initiale = 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		DAC EN	PWR EN	BKP EN	Res.	CAN EN	Res.	USB EN	I2C2 EN	I2C1 EN	UART5E N	UART4 EN	USART 3EN	USART 2EN	Res.
Res.		rw	rw	rw	Res.	rw	Res.	rw	rw	rw	rw	rw	rw	rw	Res.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI3 EN	SPI2 EN	Reserved		WWD GEN	Reserved					TIM7 EN	TIM6 EN	TIM5 EN	TIM4 EN	TIM3 EN	TIM2 EN
rw	rw	Res.		rw	Res.					rw	rw	rw	rw	rw	rw

TIMx STM32 : Registre

CR1, Adresse Base TIM2 = 0x4000 0000, Offset = 0x00, Valeur Initiale = 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CKD[1:0]		ARPE	CMS		DIR	OPM	URS	UDIS	CEN
						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

TIMx STM32 : Registre

CR2, Adresse Base TIM2 = 0x4000 0000, Offset = 0x04, Valeur Initiale = 0x0000

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

TIMx STM32 : Registre

CR2, Adresse Base TIM2 = 0x4000 0000, Offset = 0x08, Valeur Initiale = 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]			Res.	SMS[2:0]		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw

TIMx STM32 : Registre

DIER, Adresse Base TIM2 = 0x4000 0000, Offset = 0x0C, Valeur Initiale = 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	TDE	Res	CC4 DE	CC3 DE	CC2 DE	CC1 DE	UDE	Res.	TIE	Res	CC4IE	CC3IE	CC2IE	CC1IE	UIE
	rw		rw	rw	rw	rw	rw		rw		rw	rw	rw	rw	rw

TIMx STM32 : Registre

SR, Adresse Base TIM2 = 0x4000 0000, Offset = 0x10, Valeur Initiale = 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			CC4OF	CC3OF	CC2OF	CC1OF	Reserved		TIF	Res	CC4IF	CC3IF	CC2IF	CC1IF	UIF
			rc_w0	rc_w0	rc_w0	rc_w0			rc_w0		rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

TIMx STM32 : Registre

EGR, Adresse Base TIM2 = 0x4000 0000, Offset = 0x14, Valeur Initiale = 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TG	Res.	CC4G	CC3G	CC2G	CC1G	UG
									w		w	w	w	w	w

TIMx STM32 : Registre

CNT, Adresse Base TIM2 = 0x4000 0000, Offset = 0x24, Valeur Initiale = 0x0000

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

TIMx STM32 : Registre

PSC, Adresse Base TIM2 = 0x4000 0000, Offset = 0x28, Valeur Initiale = 0x0000

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

TIMx STM32 : Registre

ARR, Adresse Base TIM2 = 0x4000 0000, Offset = 0x2C, Valeur Initiale = 0x0000

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

Capture Compare TIMx STM32 : Description Fonctionnelle

Après le minuteur

Capture Compare TIMx STM32 : Description Fonctionnelle

Après le minuteur

- Les unités TIMx du STM32 ont hormis le mode minuteur basique deux autres mode d'utilisation :

Capture Compare TIMx STM32 : Description Fonctionnelle

Après le minuteur

- Les unités TIMx du STM32 ont hormis le mode minuteur basique deux autres mode d'utilisation :
  - Le mode Capture-Compare. Dans ce mode il est possible d'une part de capturer en fonction d'un événement la valeur du compteur du Timer, d'autre part de comparer la valeur du compteur à une valeur préalablement charger dans le registre Capture-Compare.

Capture Compare TIMx STM32 : Description Fonctionnelle

Après le minuteur

- Les unités TIMx du STM32 ont hormis le mode minuteur basique deux autres mode d'utilisation :
  - Le mode Capture-Compare. Dans ce mode il est possible d'une part de capturer en fonction d'un événement la valeur du compteur du Timer, d'autre part de comparer la valeur du compteur à une valeur préalablement charger dans le registre Capture-Compare.
  - Le mode PWM : Pulse Width Modulation, qui permet de générer des signaux périodiques ayant un rapport cyclique variable.

Capture Compare TIMx STM32 : Description Fonctionnelle

Après le minuteur

- Les unités TIMx du STM32 ont hormis le mode minuteur basique deux autres mode d'utilisation :
  - Le mode Capture-Compare. Dans ce mode il est possible d'une part de capturer en fonction d'un événement la valeur du compteur du Timer, d'autre part de comparer la valeur du compteur à une valeur préalablement charger dans le registre Capture-Compare.
  - Le mode PWM : Pulse Width Modulation, qui permet de générer des signaux périodiques ayant un rapport cyclique variable.
- Ces fonctions sont mises en œuvre à l'aide de 4 canaux indépendants.

Capture Compare TIMx STM32

Bloc d'entrée

Capture Compare TIMx STM32

Bloc d'entrée

- Echantillonage de l'entrée TIx correspondant au canal (TI1 pour le canal1)

Capture Compare TIMx STM32

Bloc d'entrée

- Echantillonage de l'entrée TIx correspondant au canal (TI1 pour le canal1)
- Filtrage de cette entrée pour créer le signal TIxF

Capture Compare TIMx STM32

Bloc d'entrée

- Echantillonage de l'entrée TIx correspondant au canal (TI1 pour le canal1)
- Filtrage de cette entrée pour créer le signal TIxF
- Génération d'un signal TIxFPx après détection de front et sélection de polarité.

Capture Compare TIMx STM32

Bloc d'entrée

- Echantillonage de l'entrée TIx correspondant au canal (TI1 pour le canal1)
- Filtrage de cette entrée pour créer le signal TIxF
- Génération d'un signal TIxFPx après détection de front et sélection de polarité.
- Le signal TIxFPx peut être utilisé comme un déclencheur ou comme une commande de capture.

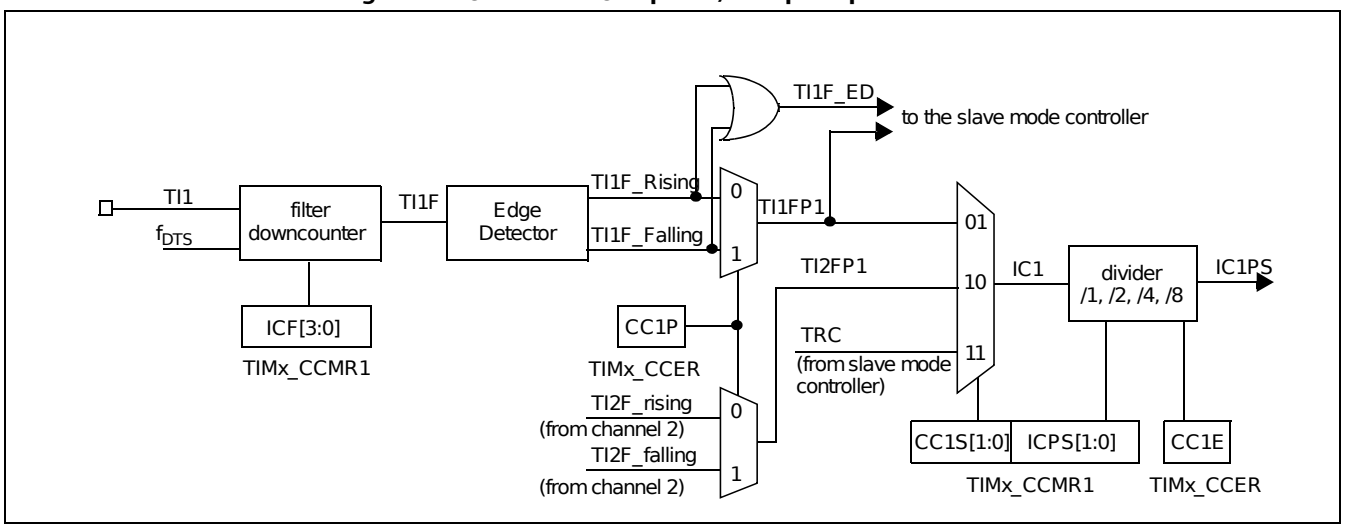
Capture Compare TIMx STM32

Bloc d'entrée

- Echantillonage de l'entrée TIx correspondant au canal (TI1 pour le canal1)
- Filtrage de cette entrée pour créer le signal TIxF
- Génération d'un signal TIxFPx après détection de front et sélection de polarité.
- Le signal TIxFPx peut être utilisé comme un déclencheur ou comme une commande de capture.
- Il est prédivisé avant d'être envoyé vers le registre de capture (ICxPS)

Capture Compare TIMx STM32

Bloc d'entrée



Capture Compare TIMx STM32

Bloc Principal

Capture Compare TIMx STM32

Bloc Principal

- Le bloc principal est composé :

Capture Compare TIMx STM32

Bloc Principal

- Le bloc principal est composé :
  - d'un registre capture/compare de préchargement

Capture Compare TIMx STM32

Bloc Principal

- Le bloc principal est composé :
  - d'un registre capture/compare de préchargement
  - d'un registre capture/compare fantôme

Capture Compare TIMx STM32

Bloc Principal

- Le bloc principal est composé :
  - d'un registre capture/compare de préchargement
  - d'un registre capture/compare fantôme
- Les accès en lecture et écriture s'effectue dans le registre capture/compare de préchargement

Capture Compare TIMx STM32

Bloc Principal

- Le bloc principal est composé :
  - d'un registre capture/compare de préchargement
  - d'un registre capture/compare fantôme
- Les accès en lecture et écriture s'effectue dans le registre capture/compare de préchargement
- En mode capture, les captures s'effectue dans le registre capture/compare fantôme, qui est ensuite copié dans le registre capture/compare de préchargement

Capture Compare TIMx STM32

Bloc Principal

- Le bloc principal est composé :
  - d'un registre capture/compare de préchargement
  - d'un registre capture/compare fantôme
- Les accès en lecture et écriture s'effectue dans le registre capture/compare de préchargement
- En mode capture, les captures s'effectue dans le registre capture/compare fantôme, qui est ensuite copié dans le registre capture/compare de préchargement
- En mode compare, le contenu du registre capture/compare de préchargement est copié dans le registre capture/compare fantôme qui est ensuite comparé au compteur

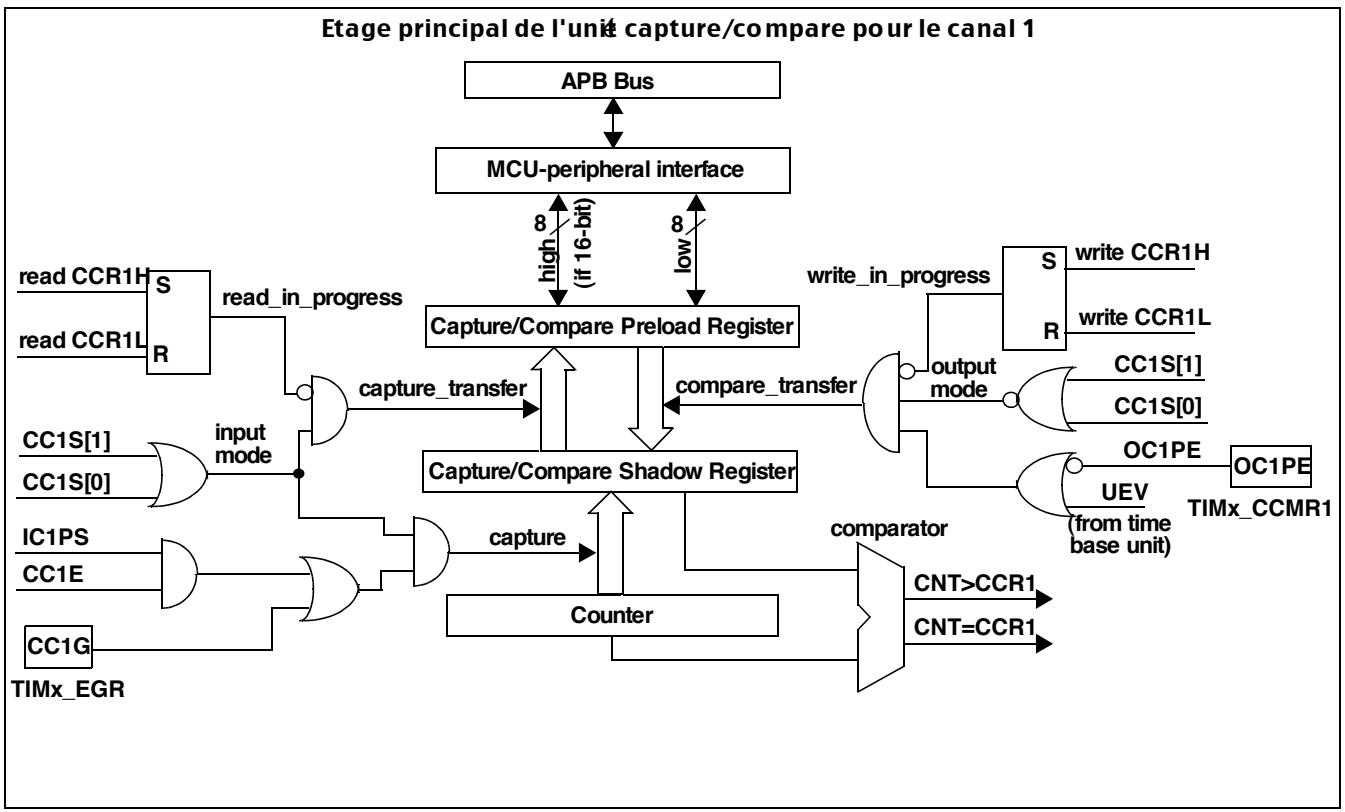
Capture Compare TIMx STM32

Bloc Principal

- Le bloc principal est composé :
  - d'un registre capture/compare de préchargement
  - d'un registre capture/compare fantôme
- Les accès en lecture et écriture s'effectue dans le registre capture/compare de préchargement
- En mode capture, les captures s'effectue dans le registre capture/compare fantôme, qui est ensuite copié dans le registre capture/compare de préchargement
- En mode compare, le contenu du registre capture/compare de préchargement est copié dans le registre capture/compare fantôme qui est ensuite comparé au compteur
- Le registre capture/compare est nommé TIM_CCRx

Capture Compare TIMx STM32

Bloc Principal



Capture Compare TIMx STM32

Bloc de sortie

Capture Compare TIMx STM32

Bloc de sortie

- Dans le bloc de sortie est généré un signal intermédiaire OCxRef

Capture Compare TIMx STM32

Bloc de sortie

- Dans le bloc de sortie est généré un signal intermédiaire OCxRef
- OCxRef est actif à l'état haut

Capture Compare TIMx STM32

Bloc de sortie

- Dans le bloc de sortie est généré un signal intermédiaire OCxRef
- OCxRef est actif à l'état haut
- Il est utilisé comme référence

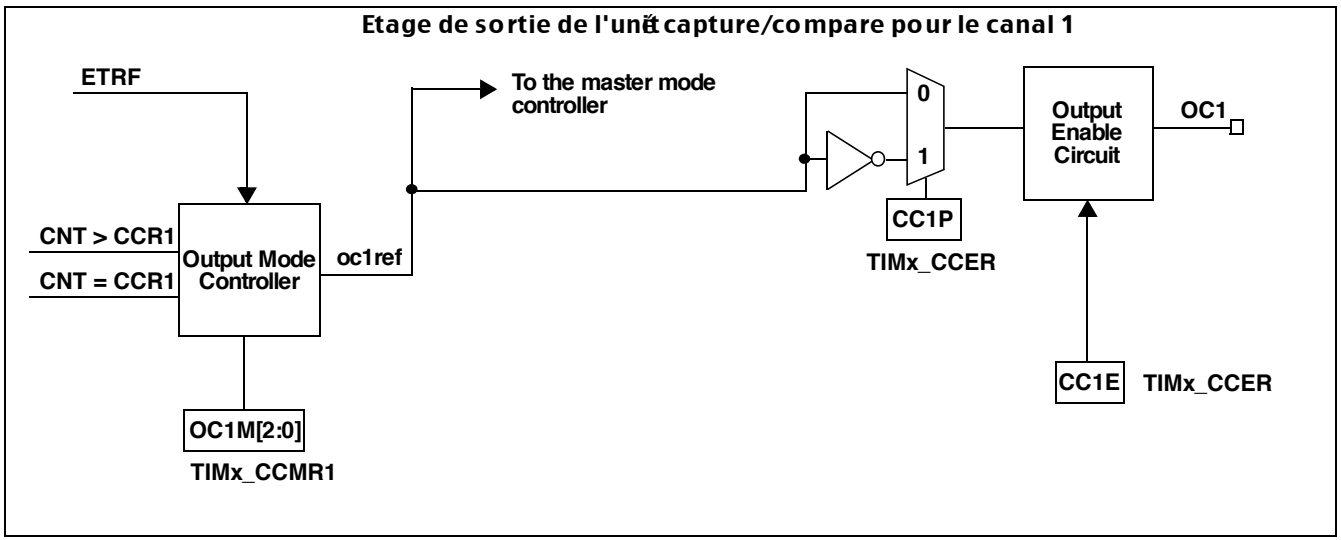
Capture Compare TIMx STM32

Bloc de sortie

- Dans le bloc de sortie est généré un signal intermédiaire OCxRef
- OCxRef est actif à l'état haut
- Il est utilisé comme référence
- La polarité intervient en bout de chaîne.

Capture Compare TIMx STM32

Bloc de sortie



Capture Compare TIMx STM32

Capture

Capture Compare TIMx STM32

Capture

- En mode capture, le registre TIMx_CCRx copie la valeur du compteur après la détection d'une transition sur le signal ICx

Capture Compare TIMx STM32

Capture

- En mode capture, le registre TIMx_CCRx copie la valeur du compteur après la détection d'une transition sur le signal ICx
- Quand une capture est réalisée, le drapeau correspondant CCxIF du registre TIMx_SR est mis à 1

Capture Compare TIMx STM32

Capture

- En mode capture, le registre TIMx_CCRx copie la valeur du compteur après la détection d'une transition sur le signal ICx
- Quand une capture est réalisée, le drapeau correspondant CCxIF du registre TIMx_SR est mis à 1
- Une interruption ou une requête DMA peut être prise en compte si elle est autorisée (bits CCxIE et CCxDE du registre TIMx_DIER).

Capture Compare TIMx STM32

Capture

- En mode capture, le registre TIMx_CCRx copie la valeur du compteur après la détection d'une transition sur le signal ICx
- Quand une capture est réalisée, le drapeau correspondant CCxIF du registre TIMx_SR est mis à 1
- Une interruption ou une requête DMA peut être prise en compte si elle est autorisée (bits CCxIE et CCxDE du registre TIMx_DIER).
- Si une capture intervient alors que le drapeau CCxIF est à 1, alors le drapeau CCxOF (over-capture) du registre TIM_SR est mis à 1.

Capture Compare TIMx STM32

Capture

- En mode capture, le registre TIMx_CCRx copie la valeur du compteur après la détection d'une transition sur le signal ICx
- Quand une capture est réalisée, le drapeau correspondant CCxIF du registre TIMx_SR est mis à 1
- Une interruption ou une requête DMA peut être prise en compte si elle est autorisée (bits CCxIE et CCxDE du registre TIMx_DIER).
- Si une capture intervient alors que le drapeau CCxIF est à 1, alors le drapeau CCxOF (over-capture) du registre TIM_SR est mis à 1.
- Le drapeau CCxIF est mis à 0 soit par logiciel, soit en lisant la valeur du registre TIMx_CCR.

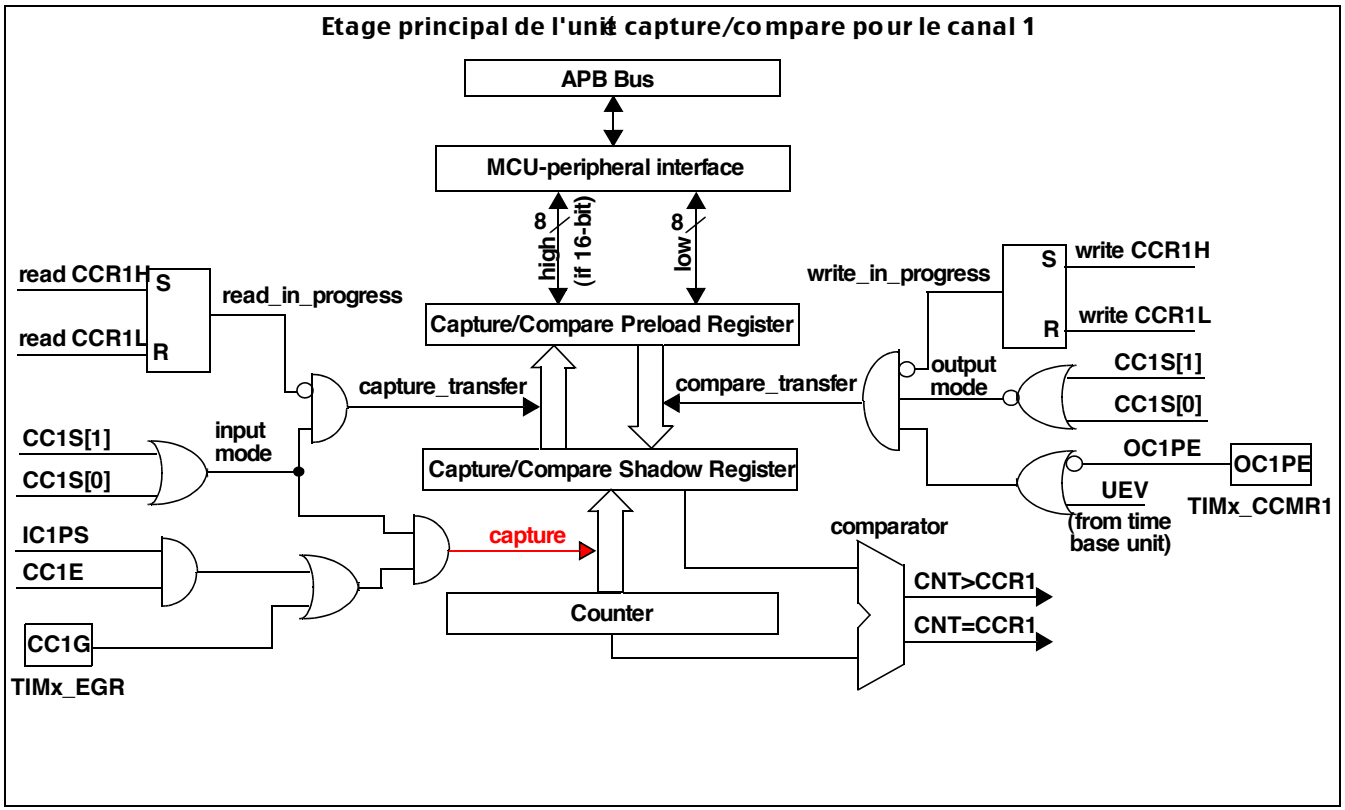
Capture Compare TIMx STM32

Capture

- En mode capture, le registre TIMx_CCRx copie la valeur du compteur après la détection d'une transition sur le signal ICx
- Quand une capture est réalisée, le drapeau correspondant CCxIF du registre TIMx_SR est mis à 1
- Une interruption ou une requête DMA peut être prise en compte si elle est autorisée (bits CCxIE et CCxDE du registre TIMx_DIER).
- Si une capture intervient alors que le drapeau CCxIF est à 1, alors le drapeau CCxOF (over-capture) du registre TIM_SR est mis à 1.
- Le drapeau CCxIF est mis à 0 soit par logiciel, soit en lisant la valeur du registre TIMx_CCR.
- Le drapeau CCxOF est mis 0 par logiciel.

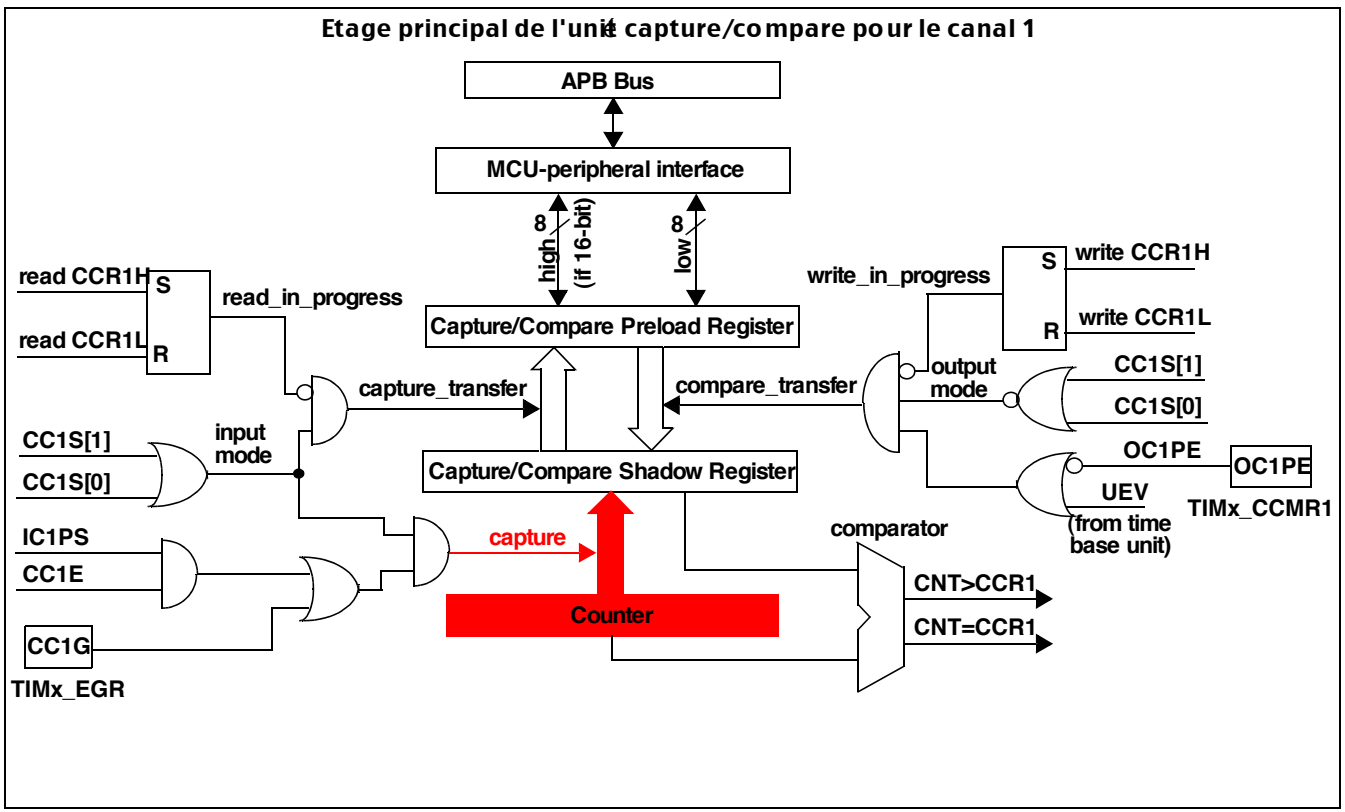
Capture Compare TIMx STM32

Capture



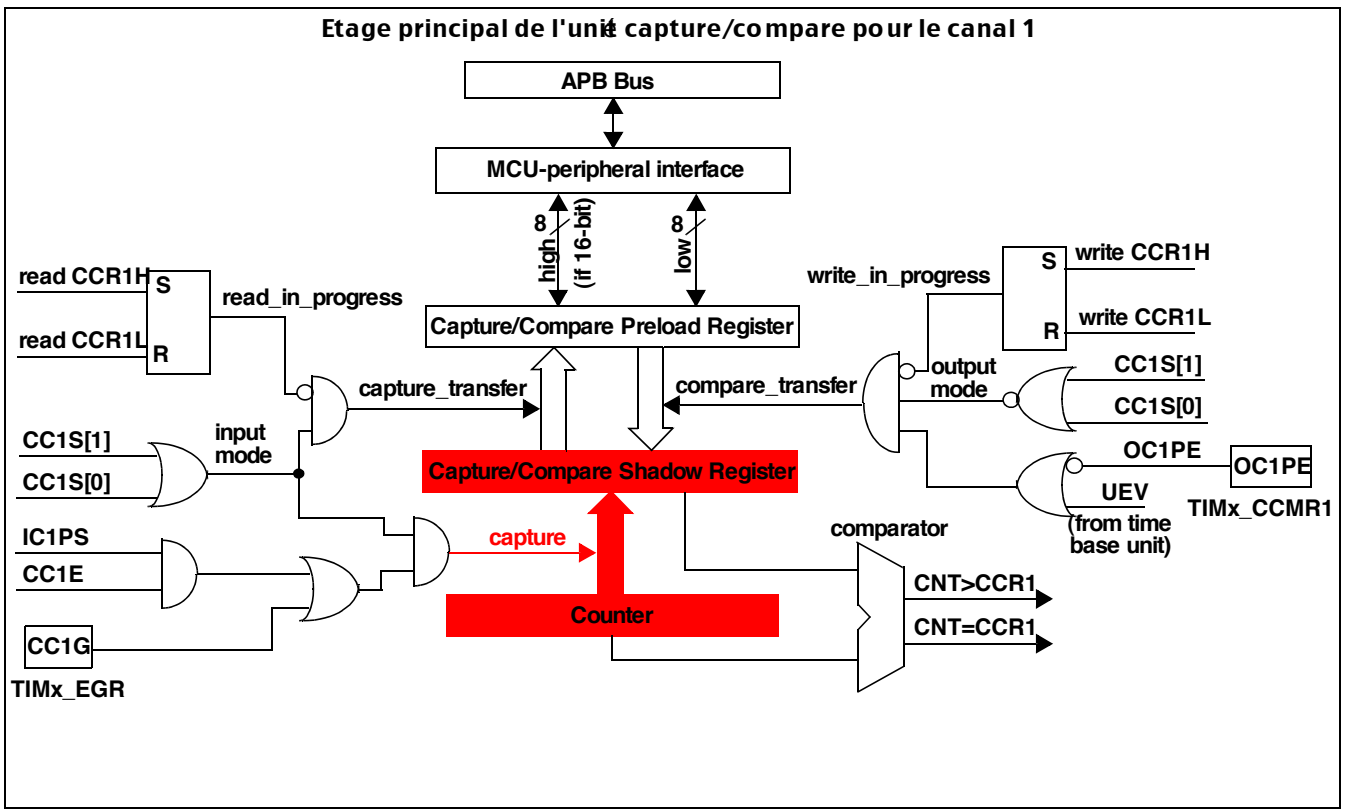
Capture Compare TIMx STM32

Capture



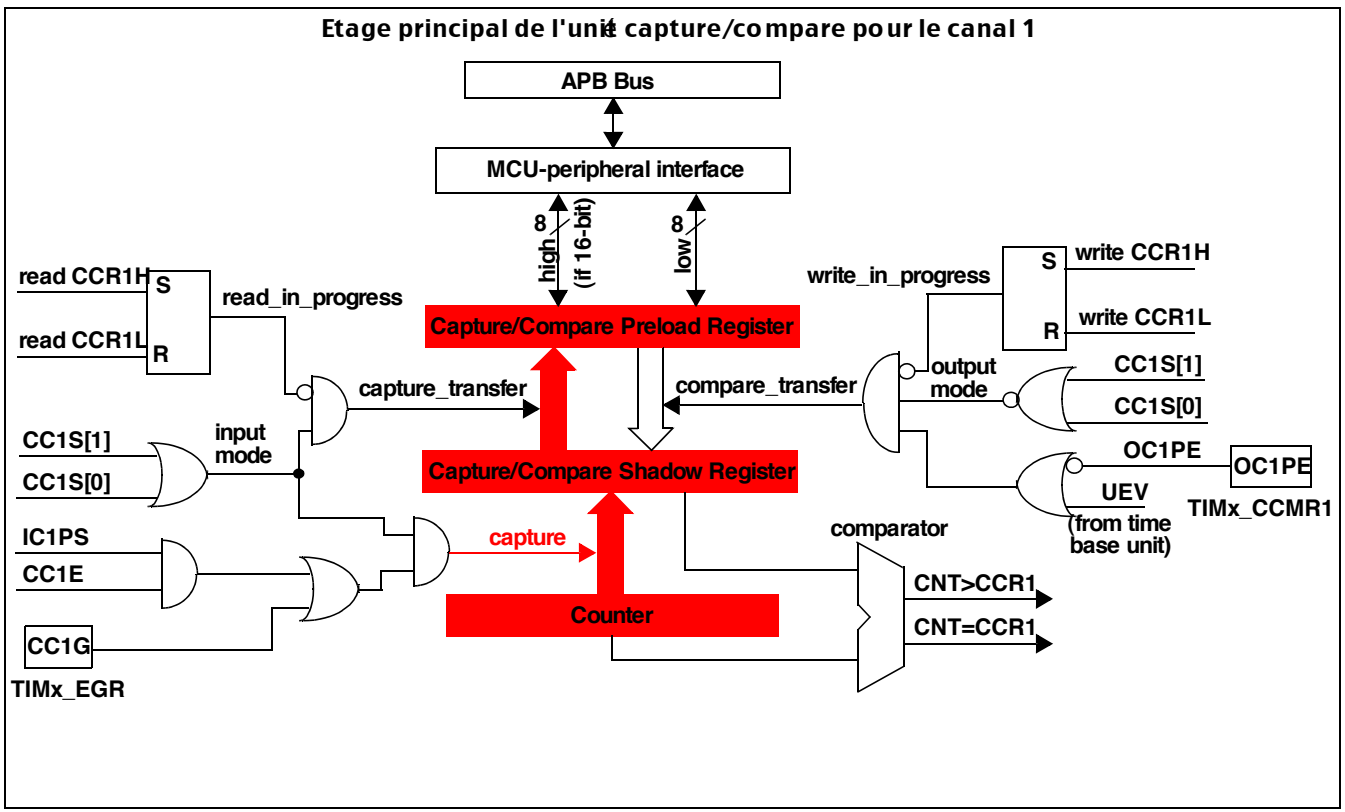
Capture Compare TIMx STM32

Capture



Capture Compare TIMx STM32

Capture



Capture Compare TIMx STM32

Compare

Capture Compare TIMx STM32

Compare

- Mode utilisé pour contrôler un signal de sortie ou indiqué une échéance temporelle.

Capture Compare TIMx STM32

Compare

- Mode utilisé pour contrôler un signal de sortie ou indiqué une échéance temporelle.
- Quand une correspondance est trouvée entre le registre de capture/compare et le compteur

Capture Compare TIMx STM32

Compare

- Mode utilisé pour contrôler un signal de sortie ou indiqué une échéance temporelle.
- Quand une correspondance est trouvée entre le registre de capture/compare et le compteur
  - La sortie correspondante (dépendant du canal) est mise à une valeur définie par le mode de comparaison (les bits OCxM du registre TIMx_CCMRx) et la polarité de sortie (le bit CCxP du registre TIMx_CCER)

Capture Compare TIMx STM32

Compare

- Mode utilisé pour contrôler un signal de sortie ou indiqué une échéance temporelle.
- Quand une correspondance est trouvée entre le registre de capture/compare et le compteur
  - La sortie correspondante (dépendant du canal) est mise à une valeur définie par le mode de comparaison (les bits OCxM du registre TIMx_CCMRx) et la polarité de sortie (le bit CCxP du registre TIMx_CCER)
  - Le bit d'interruption CCxIF du registre TIMx_SR est mis à 1

Capture Compare TIMx STM32

Compare

- Mode utilisé pour contrôler un signal de sortie ou indiqué une échéance temporelle.
- Quand une correspondance est trouvée entre le registre de capture/compare et le compteur
  - La sortie correspondante (dépendant du canal) est mise à une valeur définie par le mode de comparaison (les bits OCxM du registre TIMx_CCMRx) et la polarité de sortie (le bit CCxP du registre TIMx_CCER)
  - Le bit d'interruption CCxIF du registre TIMx_SR est mis à 1
  - L'interruption est prise en compte si le bit CCxIE du registre TIMx_DIER est à 1

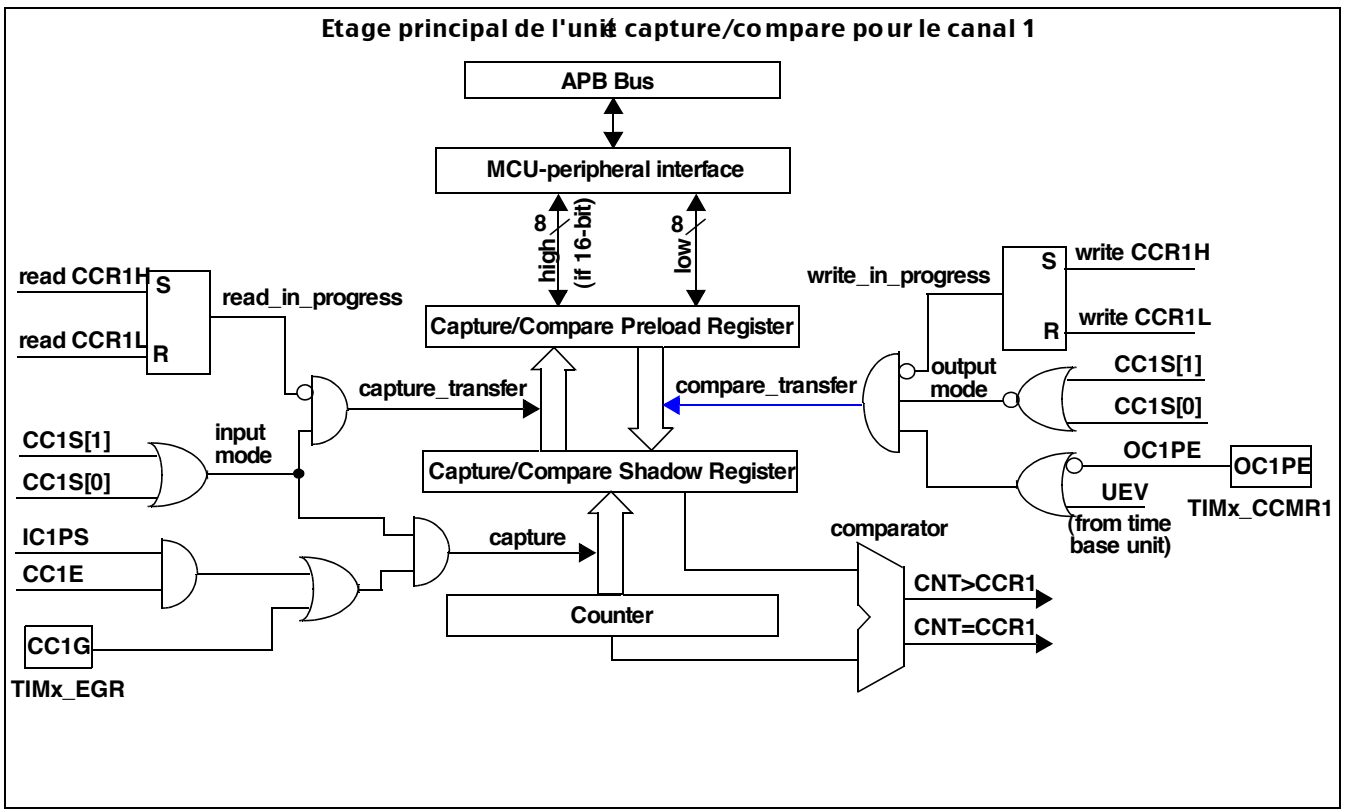
Capture Compare TIMx STM32

Compare

- Mode utilisé pour contrôler un signal de sortie ou indiqué une échéance temporelle.
- Quand une correspondance est trouvée entre le registre de capture/compare et le compteur
  - La sortie correspondante (dépendant du canal) est mise à une valeur définie par le mode de comparaison (les bits OCxM du registre TIMx_CCMRx) et la polarité de sortie (le bit CCxP du registre TIMx_CCER)
  - Le bit d'interruption CCxIF du registre TIMx_SR est mis à 1
  - L'interruption est prise en compte si le bit CCxIE du registre TIMx_DIER est à 1
  - Une requête DMA est générée sur le bit CCxDE du registre TIMx_DIER et le bit CCDS du registre TIMX_CR2 sont à 1.

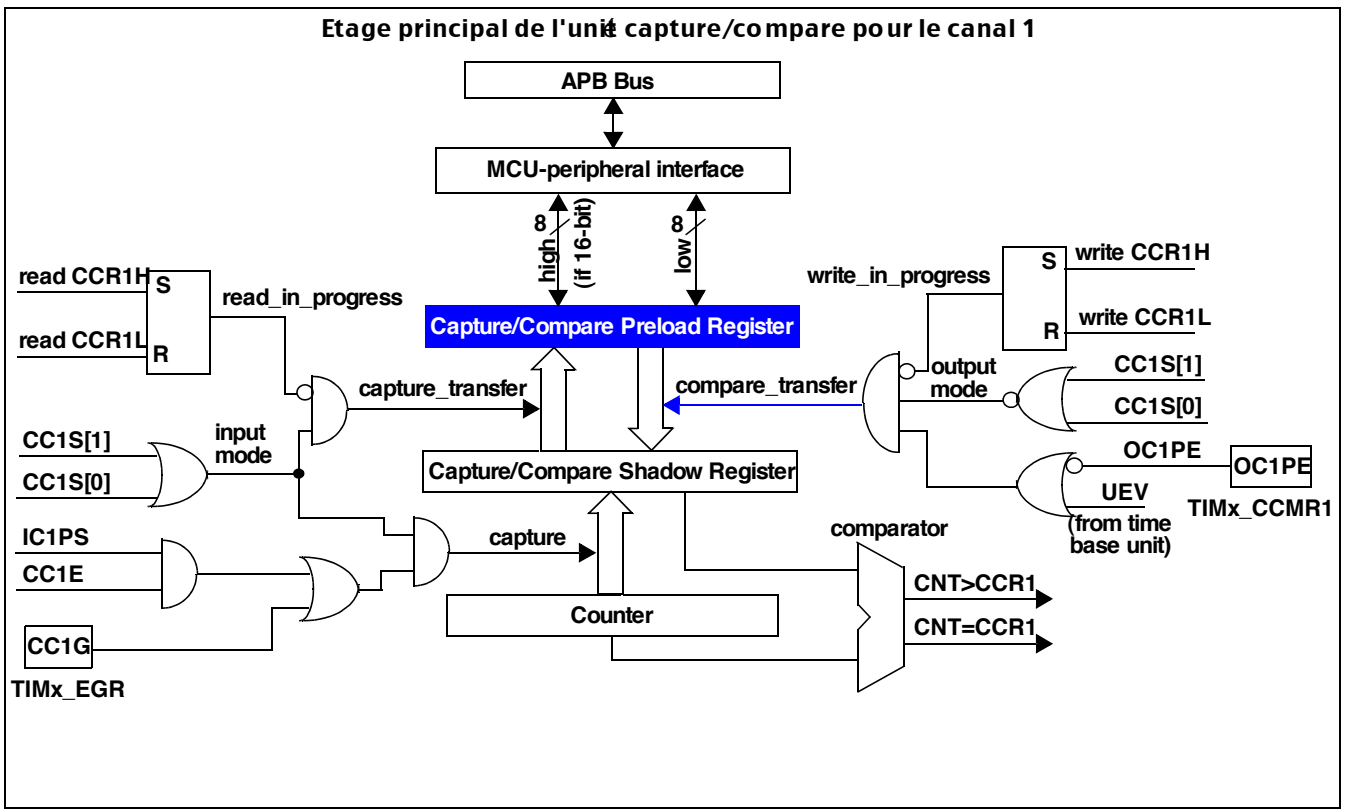
Capture Compare TIMx STM32

Compare



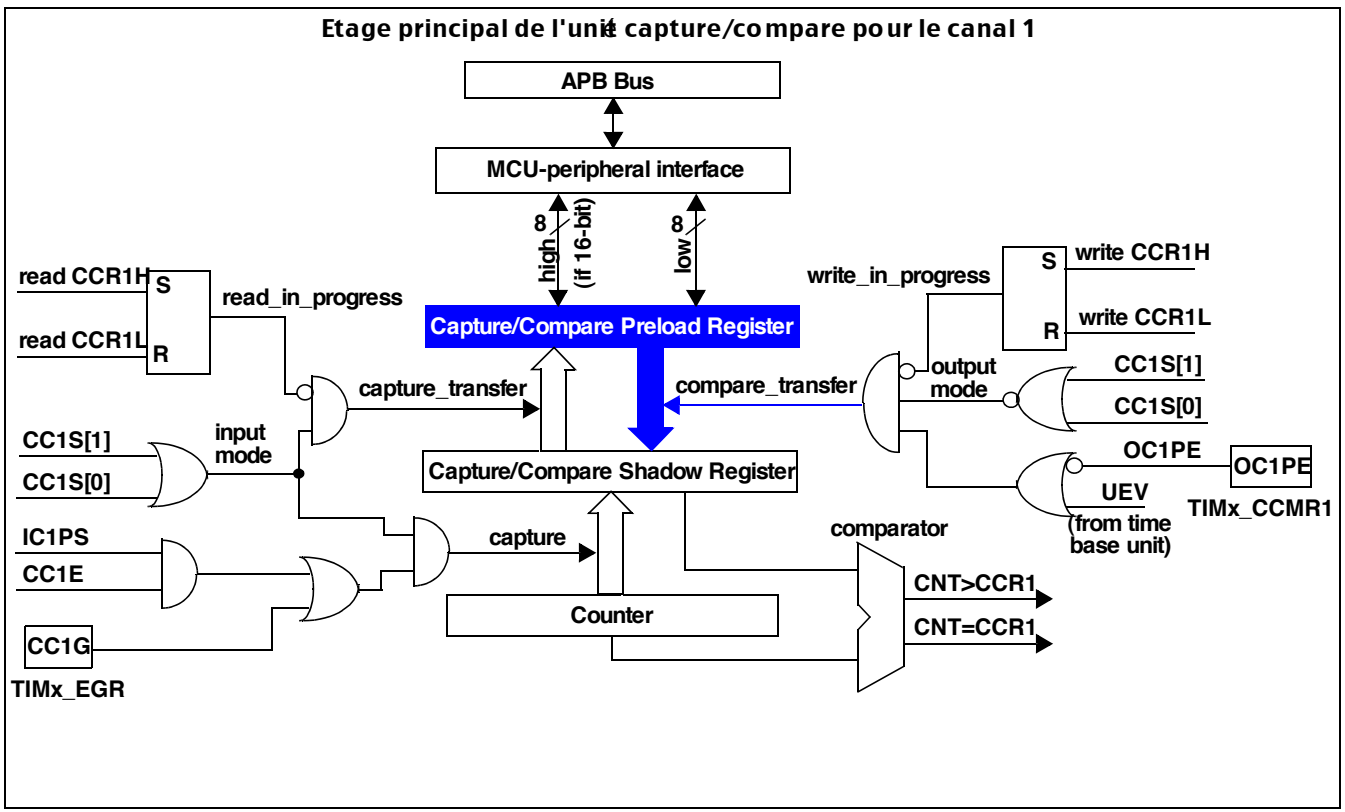
Capture Compare TIMx STM32

Compare



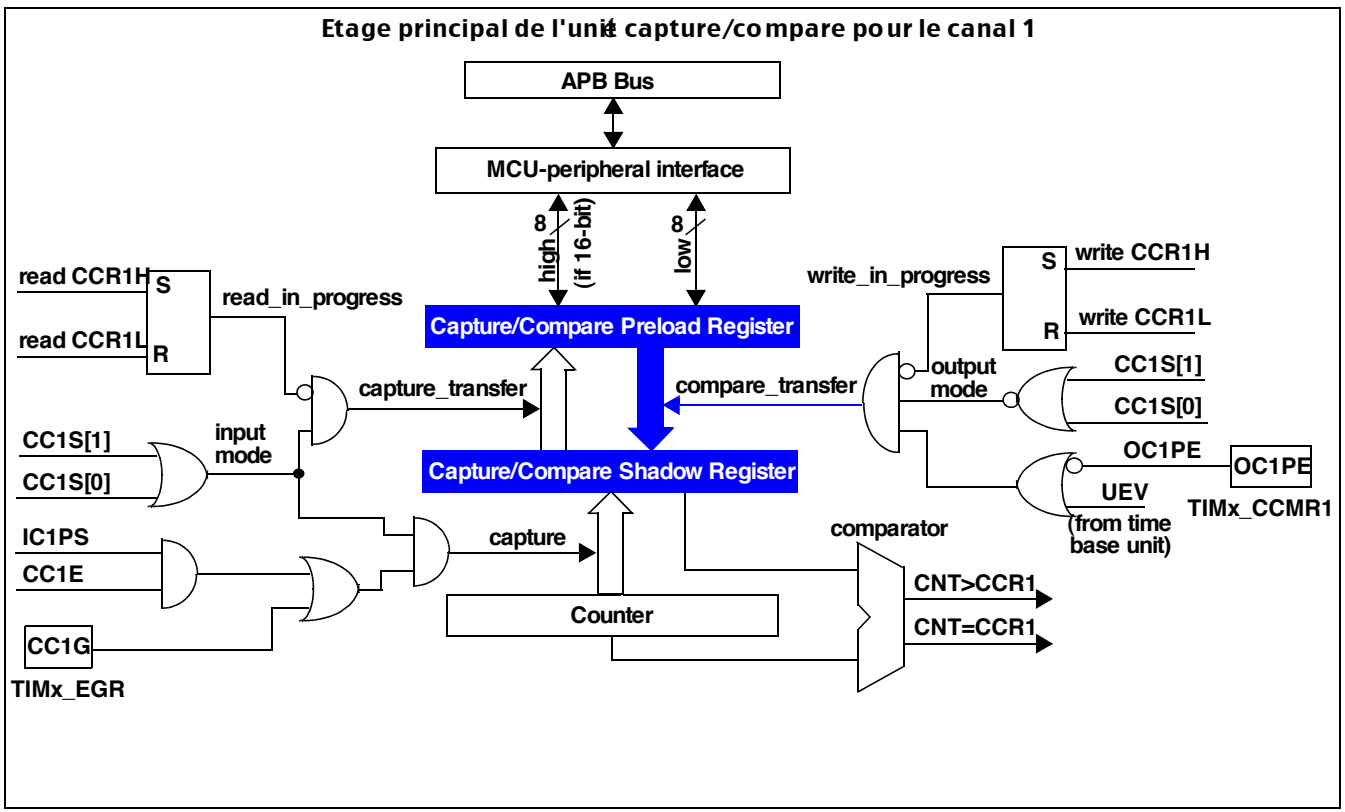
Capture Compare TIMx STM32

Compare



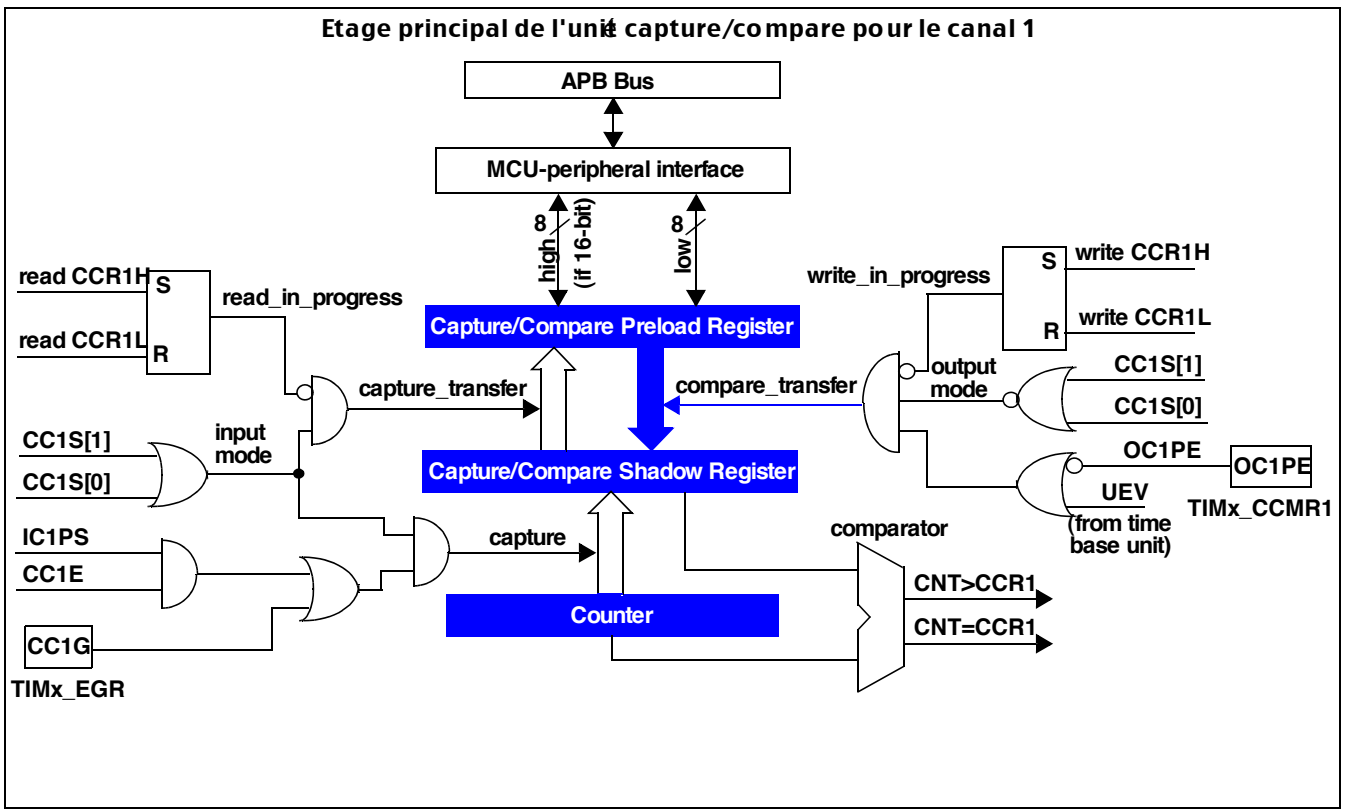
Capture Compare TIMx STM32

Compare



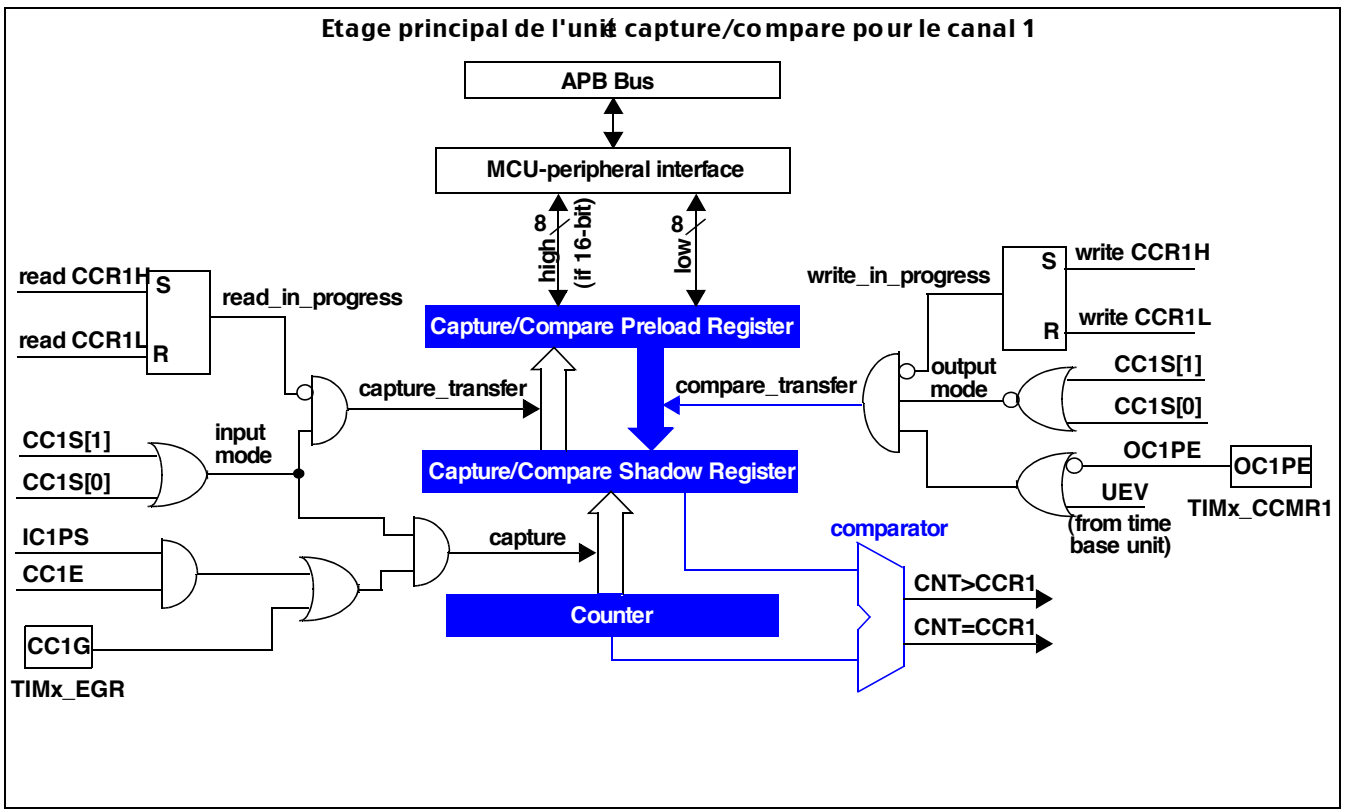
Capture Compare TIMx STM32

Compare



Capture Compare TIMx STM32

Compare



Capture Compare TIMx STM32

PWM

Capture Compare TIMx STM32

PWM

- Ce mode permet de générer un signal périodique de fréquence fixé par le registre TIMx_ARR et de rapport cyclique fixé par le registre TIMx_CCRx.

Capture Compare TIMx STM32

PWM

- Ce mode permet de générer un signal périodique de fréquence fixé par le registre TIMx_ARR et de rapport cyclique fixé par le registre TIMx_CCRx.
- Les 4 canaux peuvent être indépendamment initialisés dans ce mode en mettant les valeurs "110" ou "111" dans les bits OCxM du registre TIMx_CCMRx.

Capture Compare TIMx STM32

PWM

- Ce mode permet de générer un signal périodique de fréquence fixé par le registre TIMx_ARR et de rapport cyclique fixé par le registre TIMx_CCRx.
- Les 4 canaux peuvent être indépendamment initialisés dans ce mode en mettant les valeurs "110" ou "111" dans les bits OCxM du registre TIMx_CCMRx.
- Il faut autoriser la sortie via le bit OCxPE du registre TIMx_CCMRx

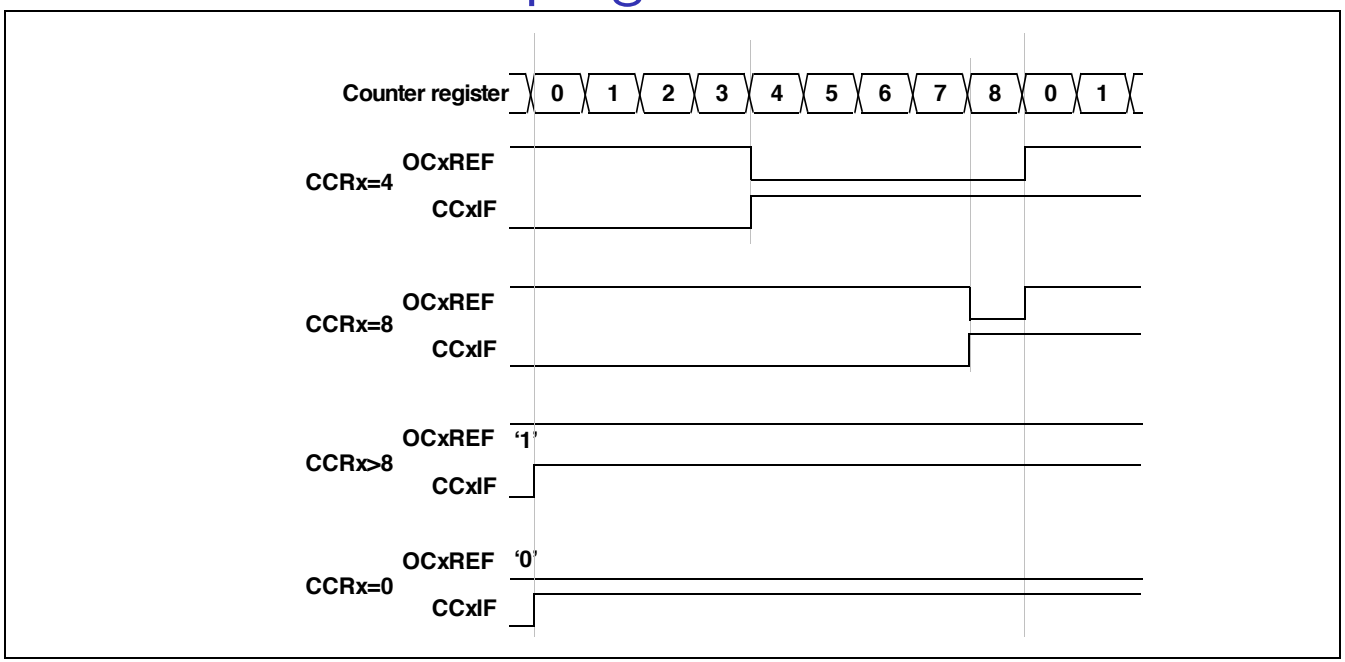
Capture Compare TIMx STM32

PWM

- Ce mode permet de générer un signal périodique de fréquence fixé par le registre TIMx_ARR et de rapport cyclique fixé par le registre TIMx_CCRx.
- Les 4 canaux peuvent être indépendamment initialisés dans ce mode en mettant les valeurs "110" ou "111" dans les bits OCxM du registre TIMx_CCMRx.
- Il faut autoriser la sortie via le bit OCxPE du registre TIMx_CCMRx
- Possibilité d'utiliser le mode Capture pour spécifier la valeur de la fréquence et du rapport cyclique

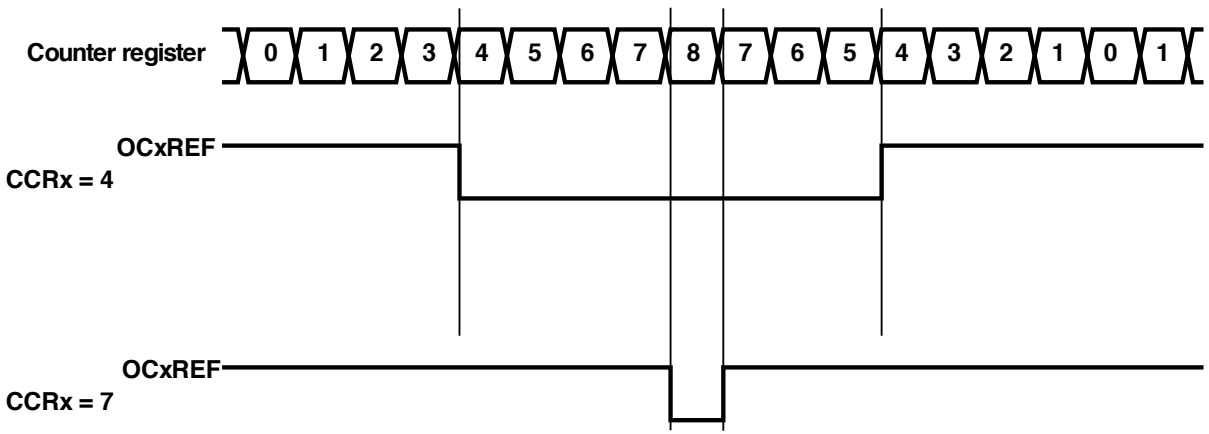
Capture Compare TIMx STM32

PWM - Mode Comptage



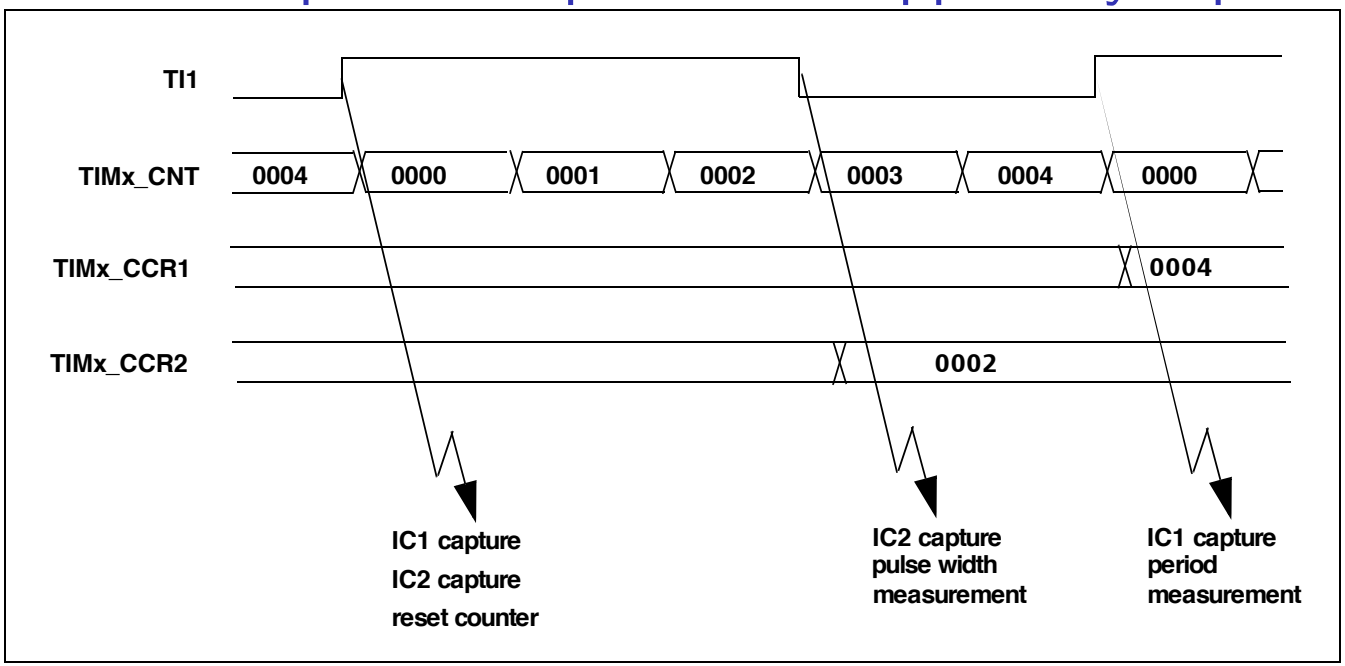
Capture Compare TIMx STM32

PWM - Mode Alterné



Capture Compare TIMx STM32

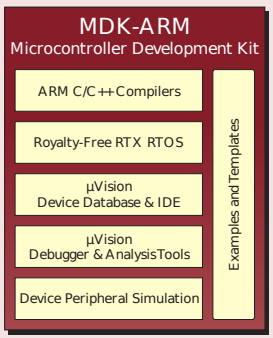
PWM - Capture Fréquence et Rapport Cyclique

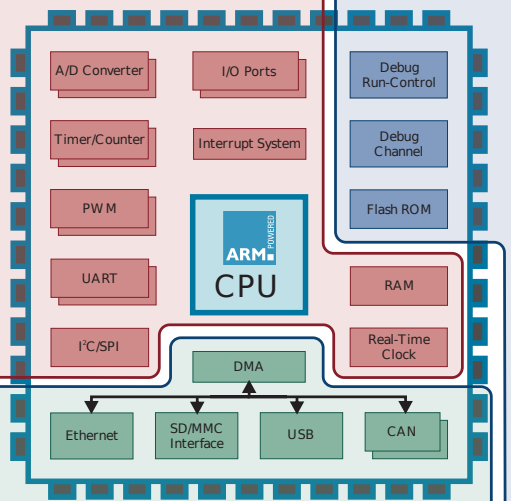


Chaine de Développement

Microcontroller Development Kit (MDK)

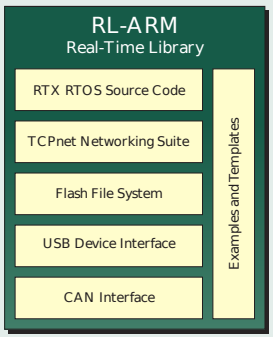
- Best-in-class ARM C/C++ Compilation Tools.
- Genuine Keil µVision®4 IDE/Debugger/Simulator.
- Royalty-free RTX Real-Time Operating System.
- Easy device configuration with Device Database support for more than 500 ARM-Powered devices.





ULINK® USB Adapters

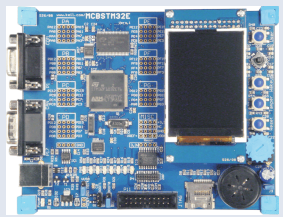
- JTAG & Serial Wire Interface.
- Flash Programming.
- On-the-fly Target Debugging.
- Real-Time Data Trace.
- ETM Instruction Trace (ULINKpro



RTOS and Middleware

- RTX Real-Time OS with Source Code.
- TCP/IP Suite with Server Applications.
- File System for ROM and Memory Cards.
- Direct suport for USB and CAN interfaces.

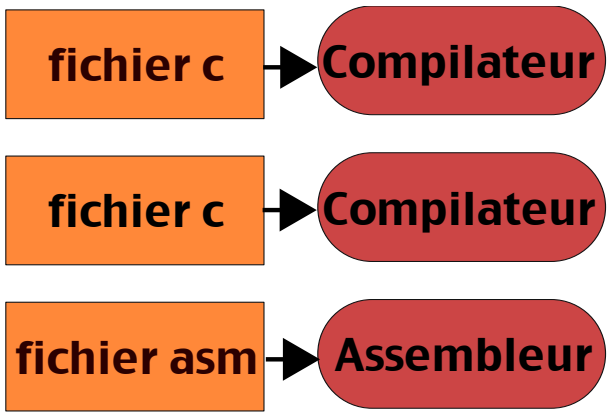
Evaluation Boards



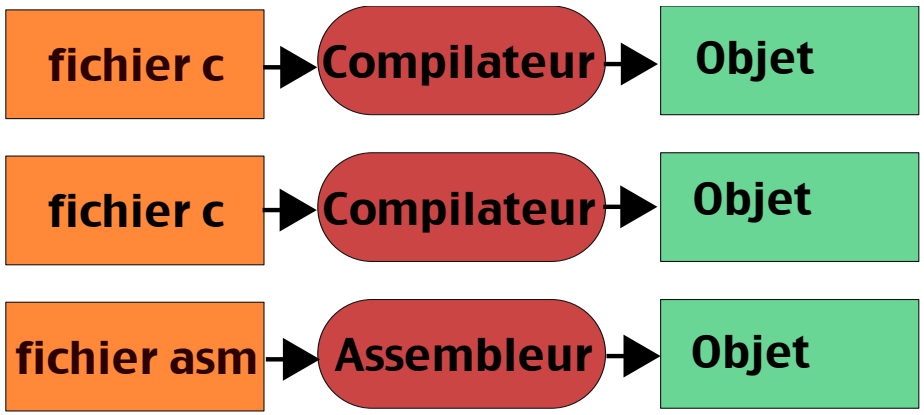
Keil provides a wide range of evaluation boards for 8, 16 and 32-bit devices

Chaine de Compilation

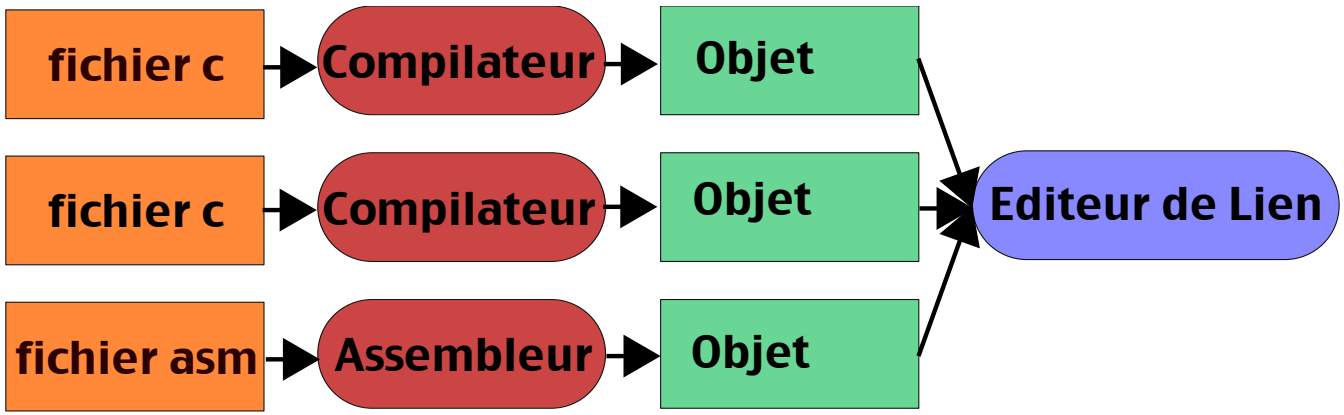
Chaine de Compilation



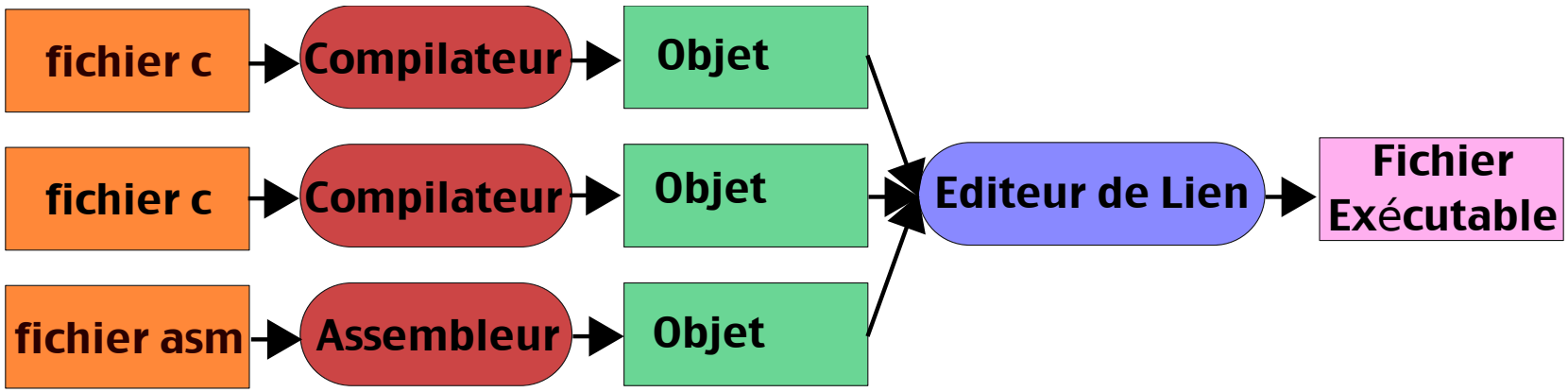
Chaine de Compilation



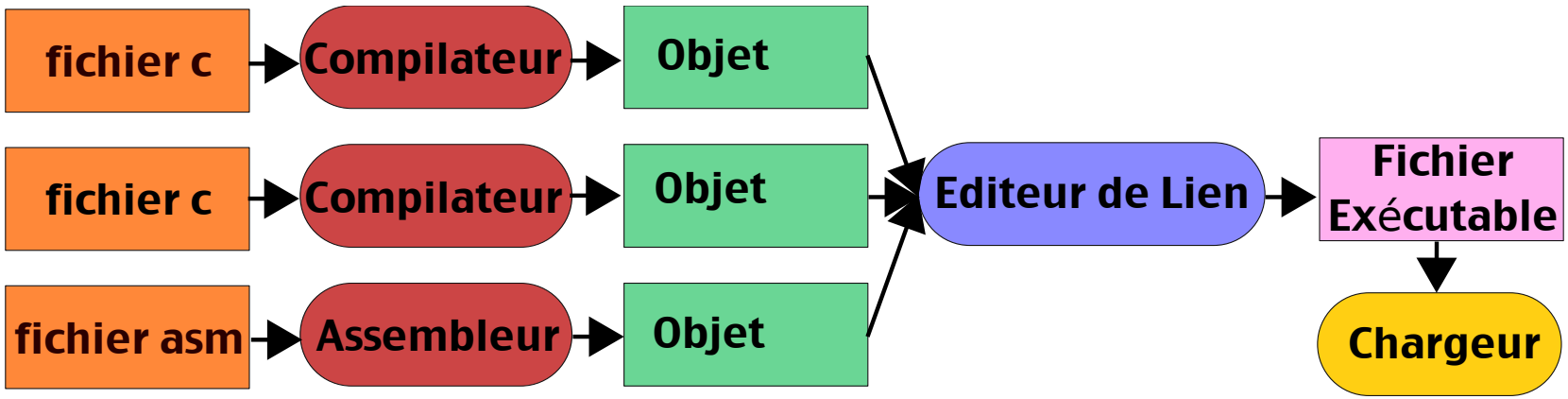
Chaine de Compilation



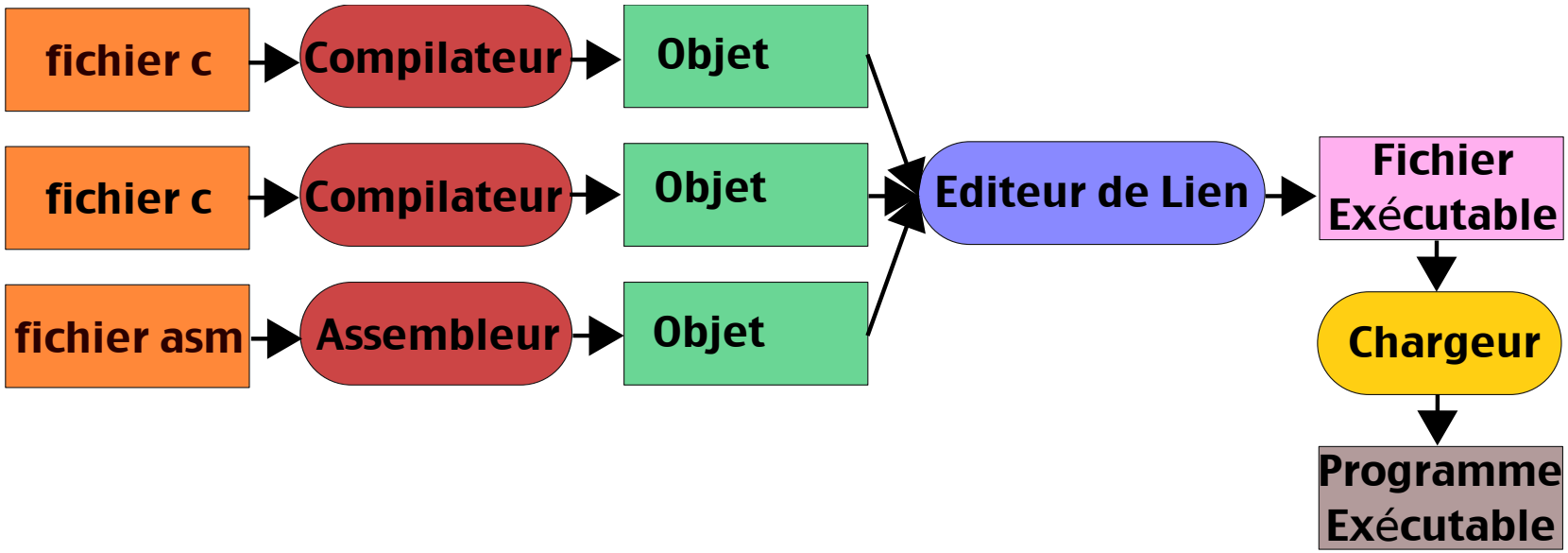
Chaine de Compilation



Chaine de Compilation



Chaine de Compilation



Editeur de lien

Editeur de lien

- Crée les liens entre les différents fichiers objets

Editeur de lien

- Crée les liens entre les différents fichiers objets
- Résoud les références

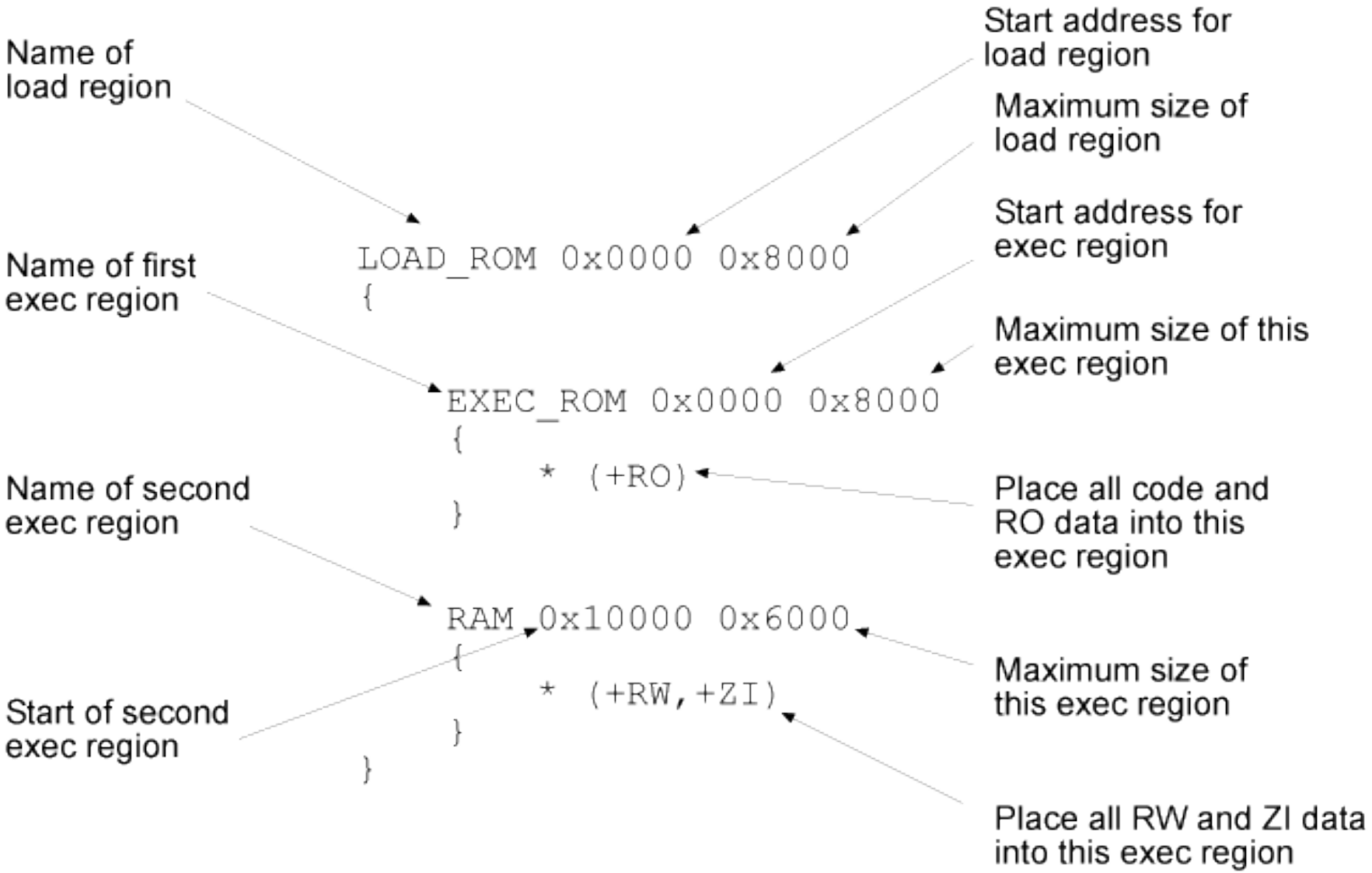
Editeur de lien

- Crée les liens entre les différents fichiers objets
- Résoud les références
- Génére les adresses finales de variables et fonction

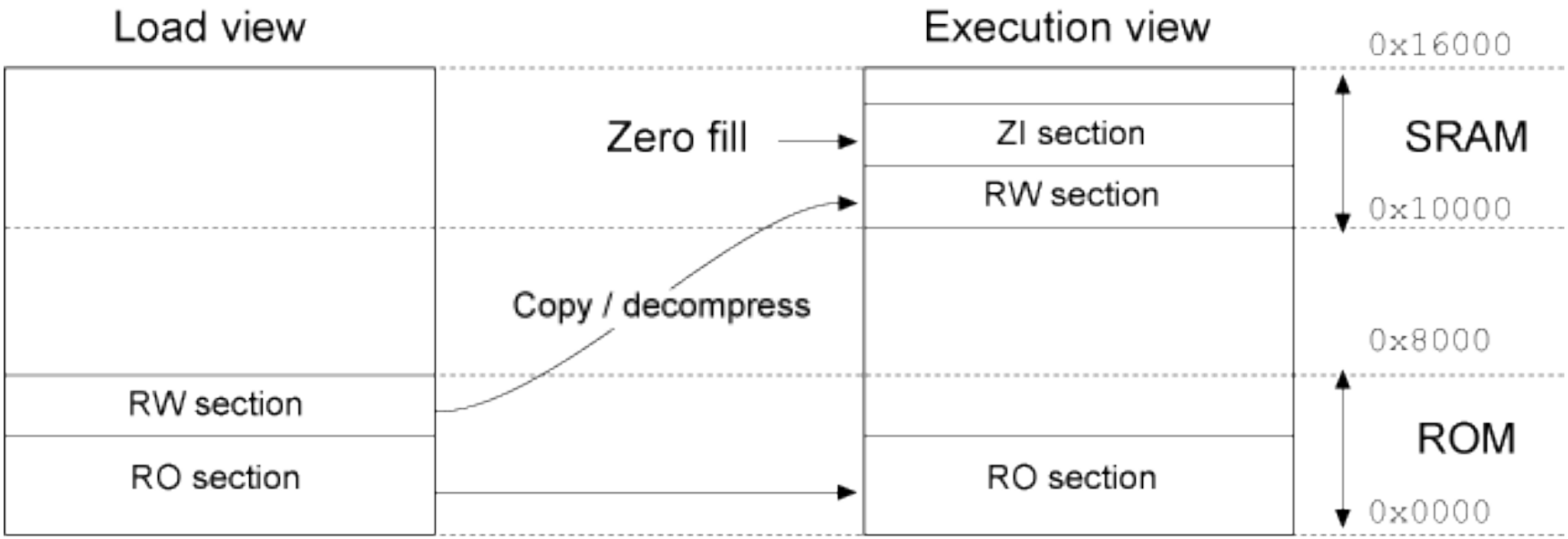
Editeur de lien

- Crée les liens entre les différents fichiers objets
- Résoud les références
- Génére les adresses finales de variables et fonction
- S'appuie sur un fichier de description du mappage entre la mémoire et les données (fonctions et variables)

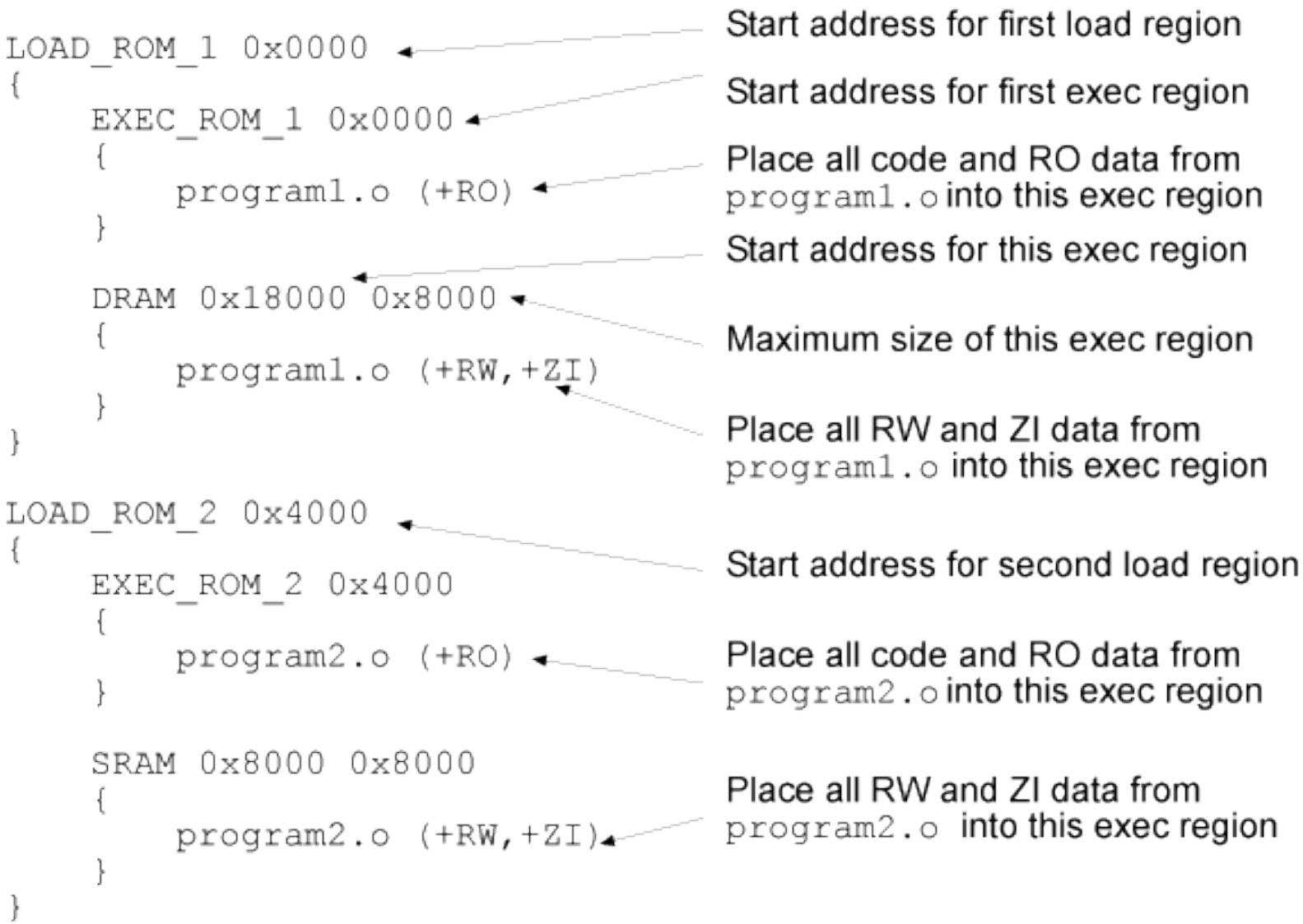
Editeur de Lien



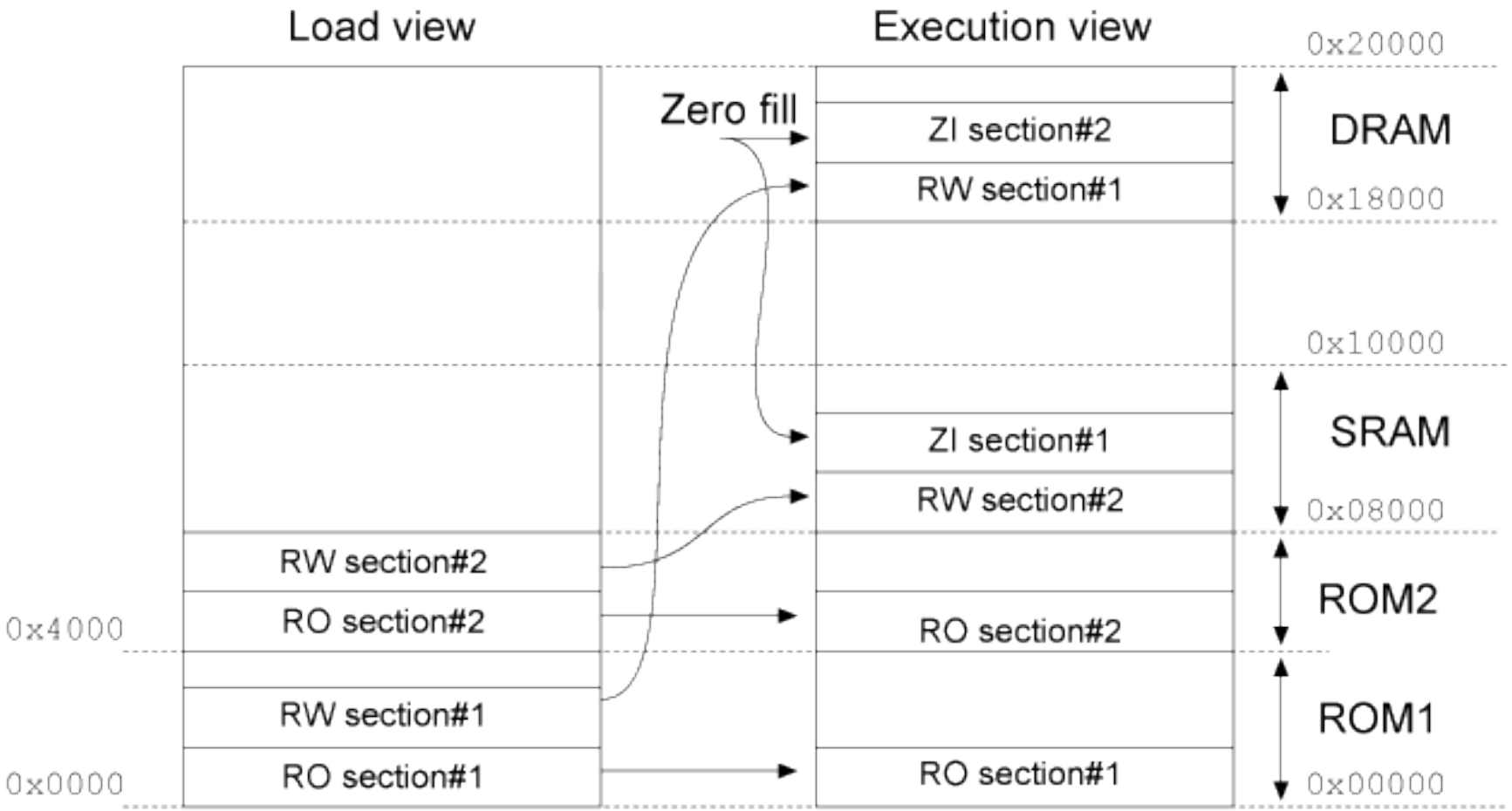
Editeur de Lien



Editeur de Lien



Editeur de Lien



Editeur de Lien : régions contigües

```
LR_1 0x040000  ; Definition d'une region de chargement nommee LR_1,
                           ; placement à l'adresse 0x040000.
{
    ER_RO +0   ; Region appelée ER_RO qui debute à la fin de la region precedente.
               ; Comme il n'y a pas de region avant l'adresse est 0x040000.
    {
        * (+RO) ; Toutes les sections RO vont dans cette région.
    }                ;  Elles sont placées consécutivement.
    ER_RW +0   ;  Region appelée ER_RW, qui debute à la fin de la region precedente
               ; L'adresse est  donc 0x040000 + la taille de la region ER_RO .
    {
        * (+RW) ; Toutes les sections RW vont dans cette région.
    }                  ;  Elles sont placées consécutivement.
    ER_ZI +0   ; Derniere region ER_ZI, qui debute à la fin de la region precedente
               ; L'adresse est donc 0x040000 + la taille de la region ER_RO +
               ; la taille de la region ER_RW.
    {
        * (+ZI) ; Toutes les sections ZI vont dans cette région.
    }               ;  Elles sont placées consécutivement.
}
```

Editeur de lien : régions non-contigües

```
LR_1 0x010000          ;  Definition region chargement LR_1
{
    ER_RO +0           ; Region ER_RO. Adresse=0x010000.
    {
        * (+RO)        ;  Toutes les sections RO
    }
    ER_RW 0x040000     ; Region ER_RW. Adresse fixée = 0x040000.
    {
        * (+RW)        ;  Toutes les sections RW
    }
    ER_ZI +0           ;  Region ER_ZI. Adresse = 0x40000 + Taille de la region ER_RW
    {
        * (+ZI)        ;  Toutes les sections ZI.
    }
}
```

Editeur de lien : régions non-contigües

```
LR_1 0x010000      ;  Definition premiere region chargement 0x010000.
{
    ER_RO +0       ;  Region ER_RO. Adresse=0x010000.
    {
        * (+RO)    ; Toutes les sections RO.
    }
}
LR_2 0x040000      ;  Definition seconde region chargement 0x040000.
{
    ER_RW +0       ;  Region ER_RW. Adresse=0x040000.
    {
        * (+RW)    ; Toutes les section RW.
    }
    ER_ZI +0       ;  Region ER_ZI. Adresse=0x040000 + Taille region ER_RW
    {
        * (+ZI)    ; Toutes les sections ZI.
    }
}
```

Editeur de lien : placer un fichier objet dans une région

```
ROM_LOAD 0x0000 0x4000 ; region de chargement ROM_LOAD
{
  ROM_EXEC 0x0000 0x4000       ; region ROM_EXEC à 0x0
  {
    vectors.o (Vect, +FIRST)   ; Table des vecteurs d'interruption
    * (InRoot$$Sections)       ; Toutes les sections qui doivent
                               ; etre dans une region maitre, par exemple __main.o,
                               ; __scatter*.o, __dc*.o, and * Region$$Table
  }
  RAM 0x10000 0x8000     ; region RAM à 0x10000
  {
    * (+RO, +RW, +ZI)          ; Toutes les autres sections
  }
}
```