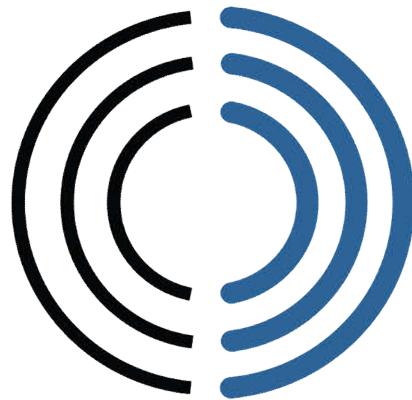


# EEL7030 - Microprocessadores

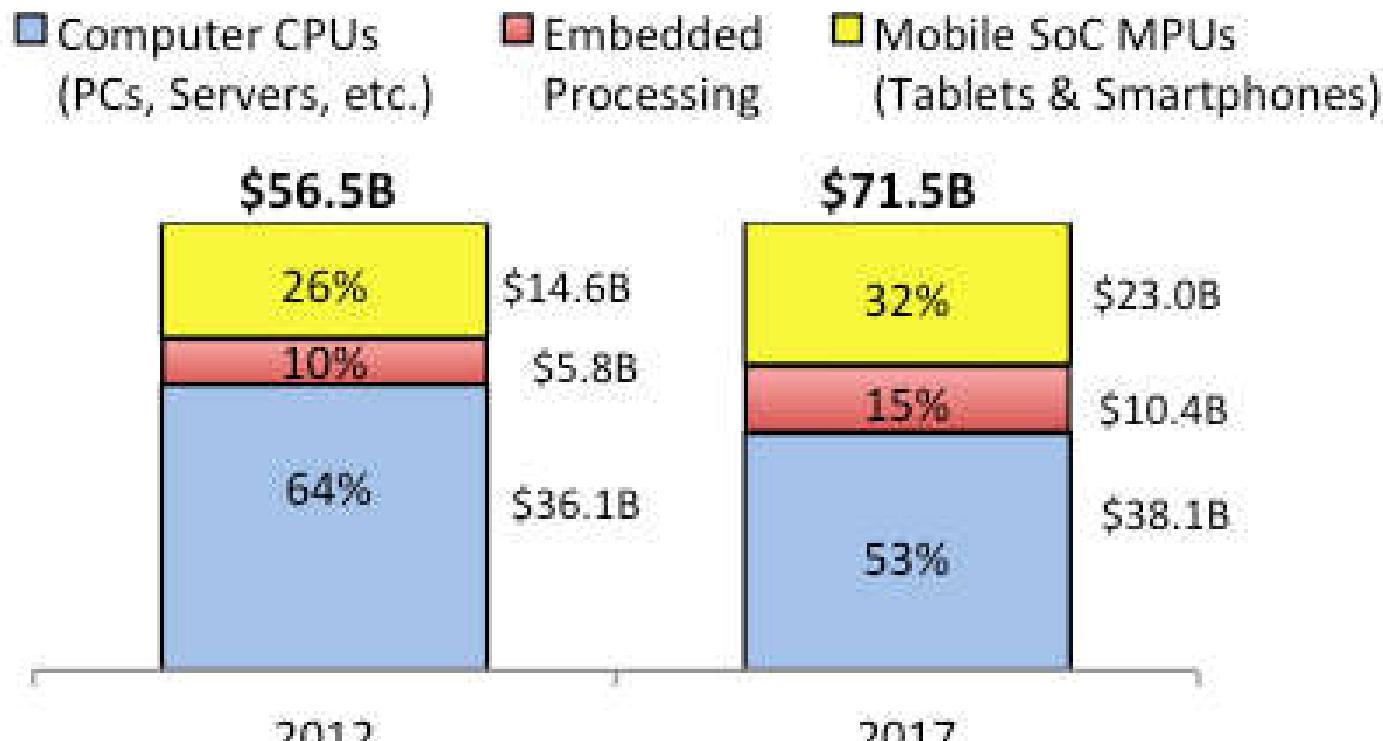


**LCS**

Laboratório de  
Comunicações  
e Sistemas  
Embarcados

**Prof. Raimes Moraes**  
**EEL - UFSC**

# Shifting Microprocessor Sales

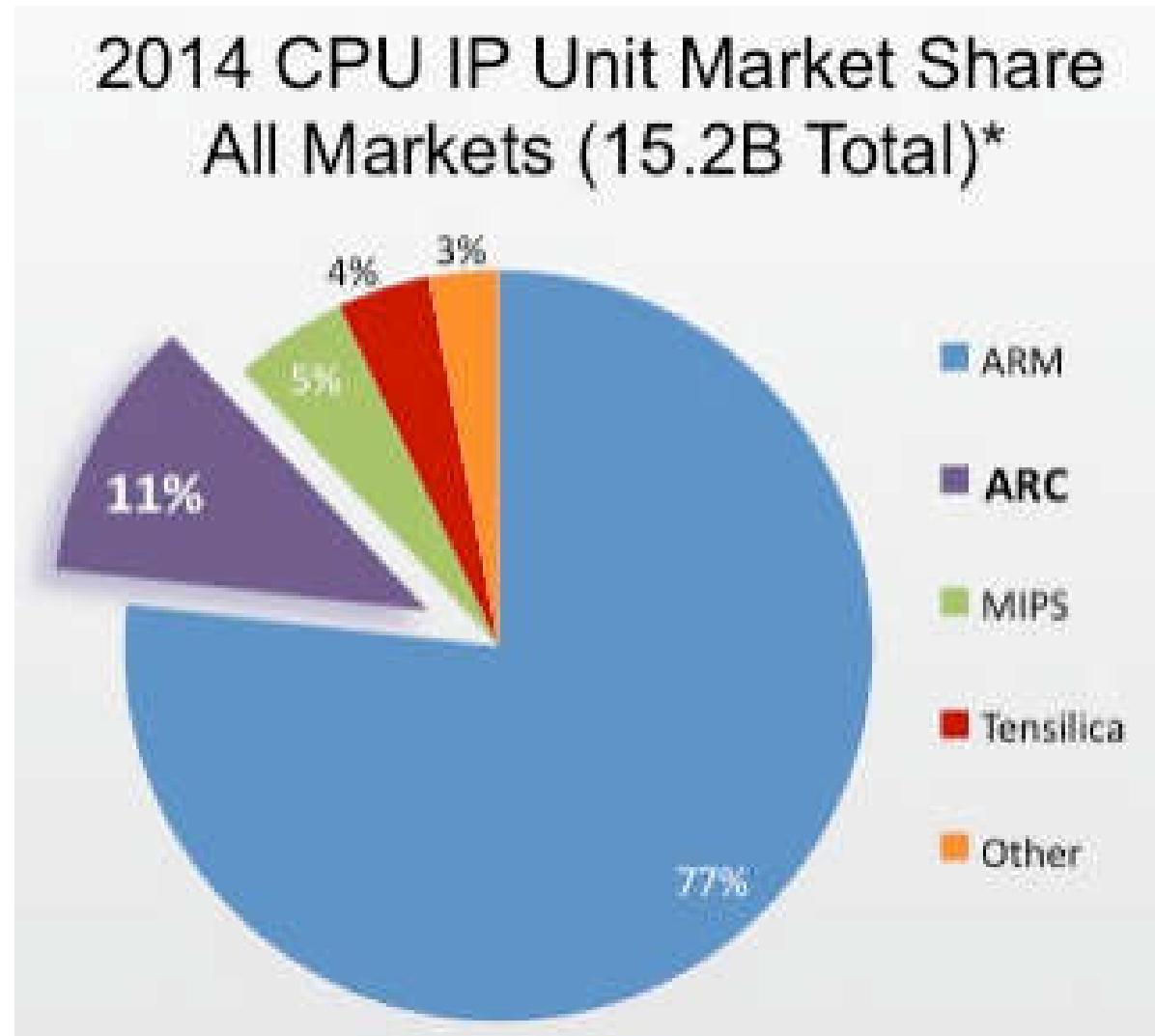


Source: IC Insights

- Participação da Intel no mercado de microprocessadores caiu abaixo de 60%
- Participação de *ARM-based SOCs* aumentou

[https://www.eetimes.com/document.asp?doc\\_id=1332968](https://www.eetimes.com/document.asp?doc_id=1332968)

## ***SHARE OF LICENSED MICROPROCESSOR SHIPMENTS***



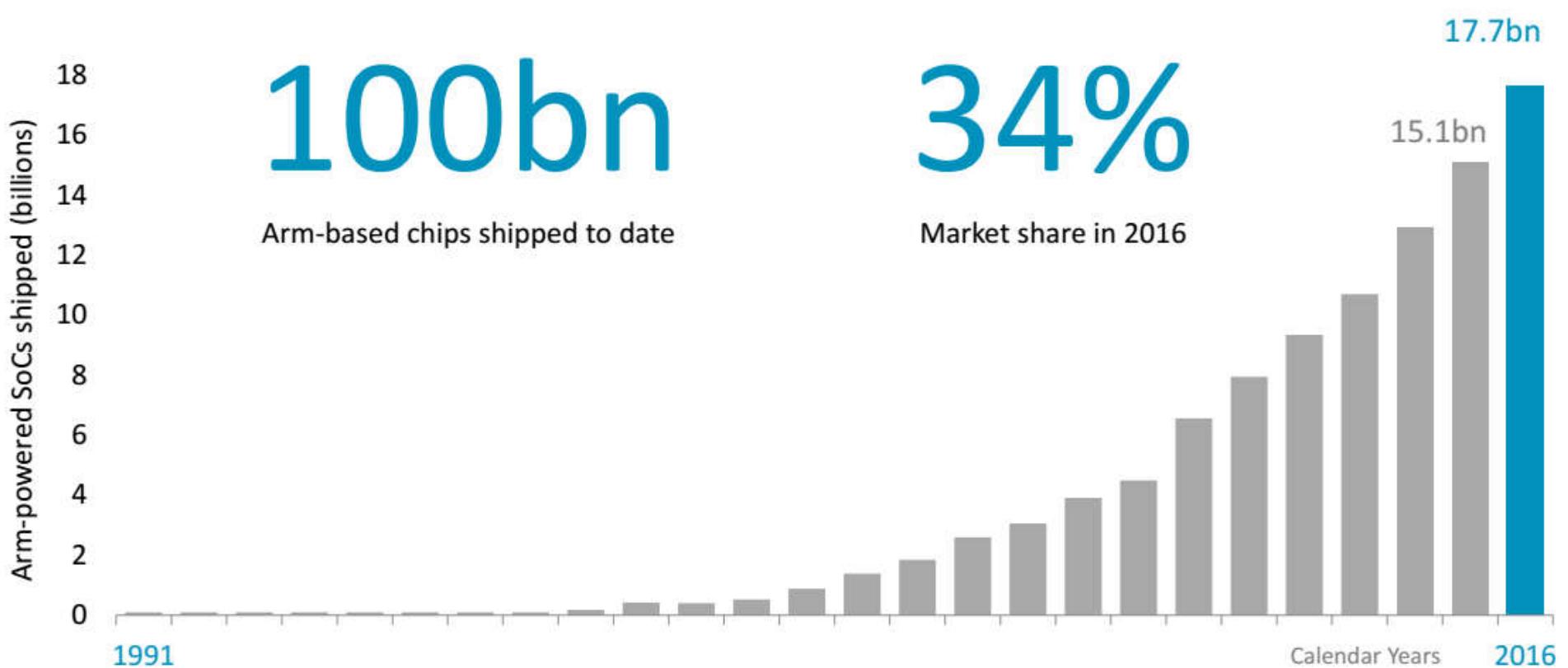
<https://www.semiwiki.com/forum/content/5008-were-number-two-we-try-harder.html>

ARC: produzido por Synopsys

MIPS: produzido por by Imagination Technologies

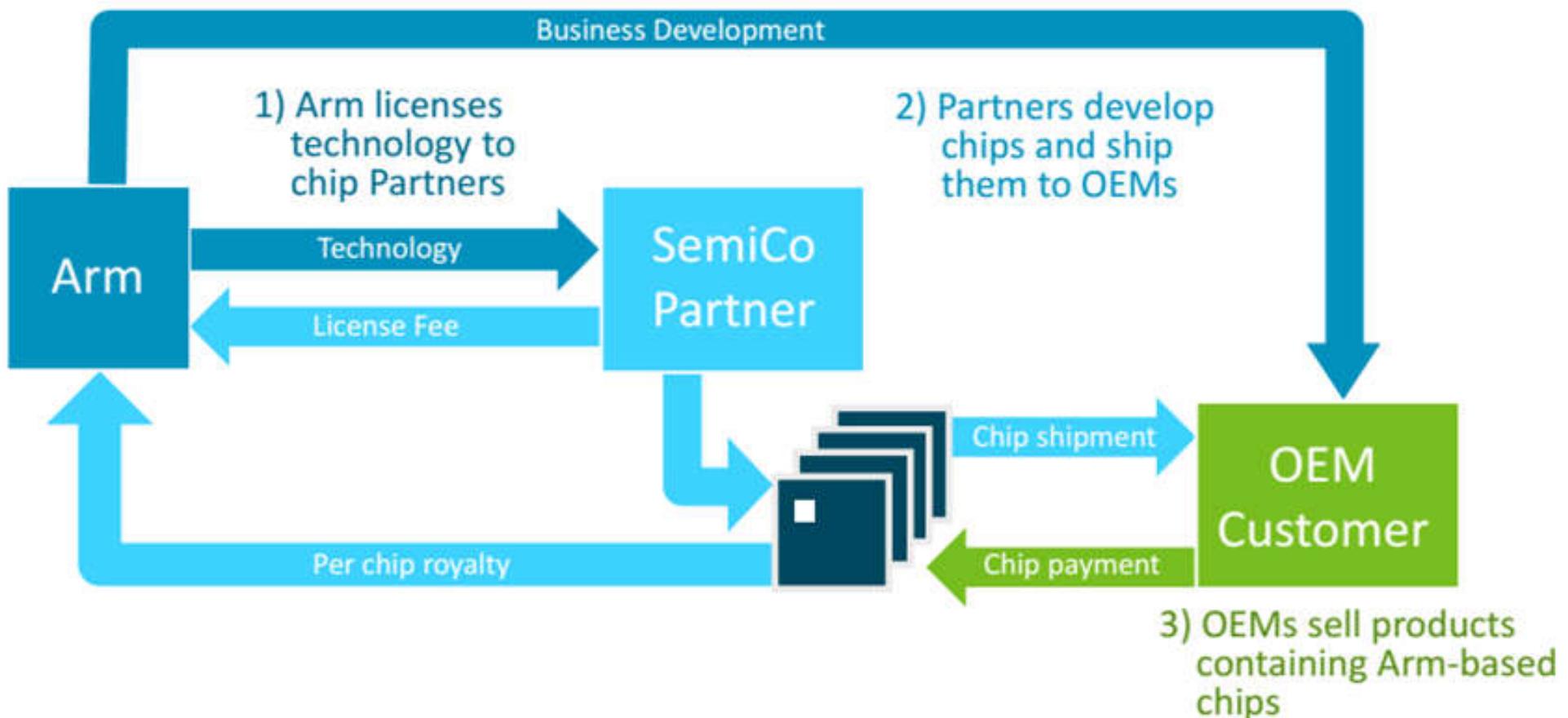
# ARM - Advanced RISC Machine

## Arm-based chip shipments



# Modelo de Negócio da ARM

- ARM desenvolve tecnologia que é licenciada para fabricantes de semicondutores
- ARM recebe pela venda da licença e *royalty* sobre cada IC vendido



**Original equipment manufacturers** (OEMs): empresas que adquirem componentes de outras para montar sistemas que são vendidos sob o seu nome (uma definição).

# **ARM - Advanced RISC Machine**

- Projetado para equipar computadores fabricados pela ACORN Computers (Cambridge – Inglaterra) para a BBC nos anos 80 (em paralelo com IBM PC)**
- Licenciado para fabricantes de semicondutores**
- Utilizado em tablets e outros dispositivos portáteis (iPod e iPhone)**

## **Referências:**

**STM32F429xx datasheet e RM0090 Reference Manual**

(<http://www.st.com/st-web-ui/static/active/en/resource/technical/document/datasheet/DM00071990.pdf>)

**The Insider's Guide To The STM32 ARM Based Microcontrollers**

(<http://www.hitex.com/fileadmin/pdf/insiders-guides/stm32/isg-stm32-v18d-scr.pdf>)

# ARM - Advanced RISC Machine

Ano	Família	Característica	Cache	MIPS@MHz Típicos
1985	ARM1	32-bit RISC	None	
1987	ARM2	Multiply and swap instructions; Integrated memory management unit, graphics and I/O processor	None	7@12
1989	ARM3	Processor cache	4 kB	12@25
1991	ARM6	32-bit addresses; floating-point	4 kB	28@33
1992	ARM7	Integrated SoC	8 kB	60@60
1996	ARM8	5-stage pipeline; static branch prediction	8 kB	84@72
1997	ARM9		16 kB	300@300
1999	ARM9E	Enhanced DSP instructions	16 kB	220@200
2001	ARM10E	6-stage pipeline	32 kB	
2002	ARM11	9-stage pipeline	Variable	740@665
2004	Cortex-M3	3-stage pipeline	Variable	2000@1G
2005	Cortex-A8	13-stage superscalar pipeline	16-32 kB	2,0 DMIPS

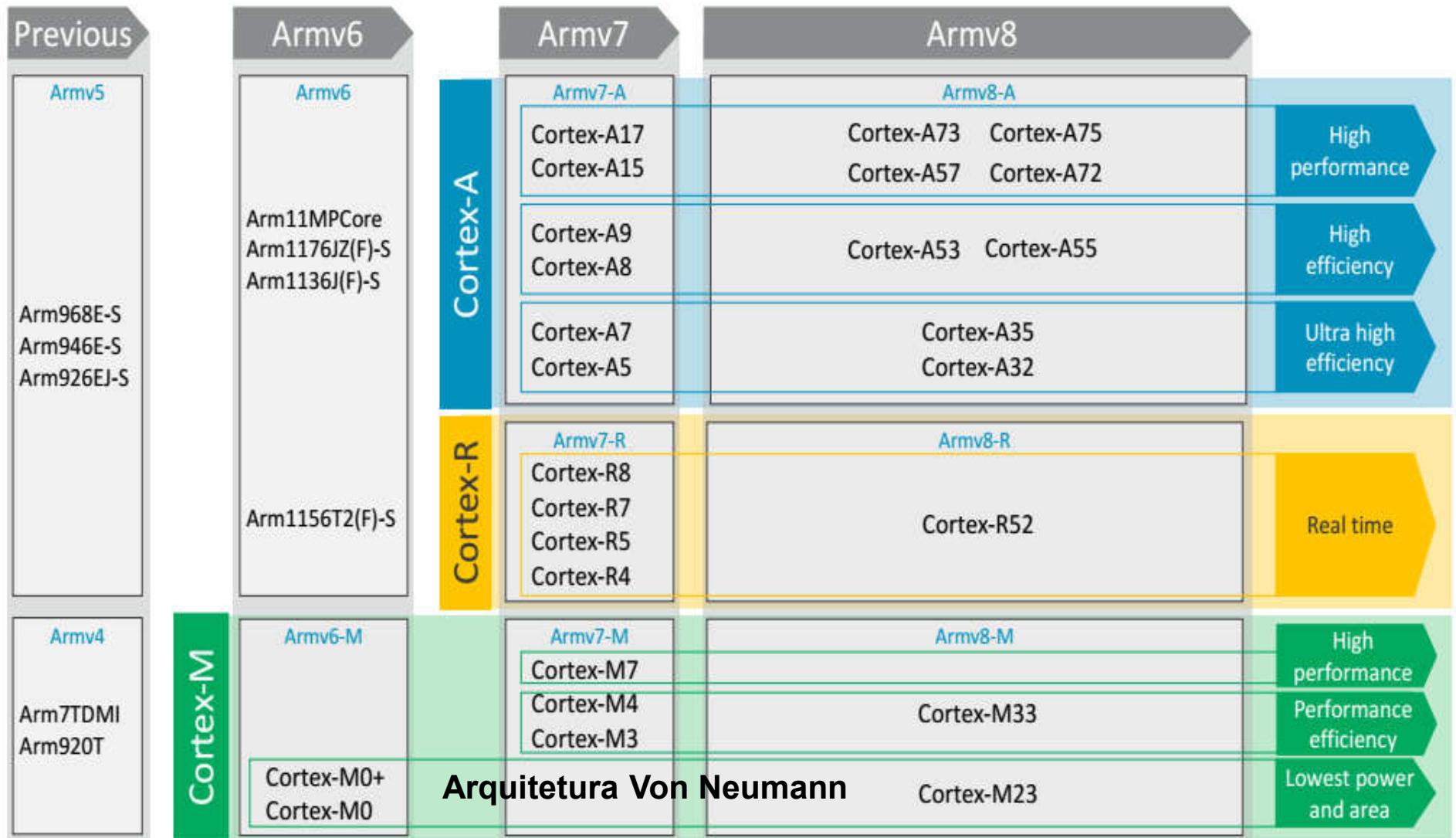
# ARM - Advanced RISC Machine

Ano	A (64 bits)	A (32 bits)	R	M
2007		Cortex-A9		Cortex-M1
2009		Cortex-A5		Cortex-M0
2010		Cortex-A15		Cortex-M4(F)
2011		Cortex-A7	Cortex-R4 Cortex-R5 Cortex-R7	
2012	Cortex-A53 Cortex-A57			Cortex-M0+
2013		Cortex-A12		
2014		Cortex-A17		Cortex-M7(F)
2015	Cortex-A35 Cortex-A72			
2016	Cortex-A73	Cortex-A32	Cortex-R8 Cortex-R52	Cortex-M23 Cortex-M33(F)
2017	Cortex-A55 Cortex-A75			

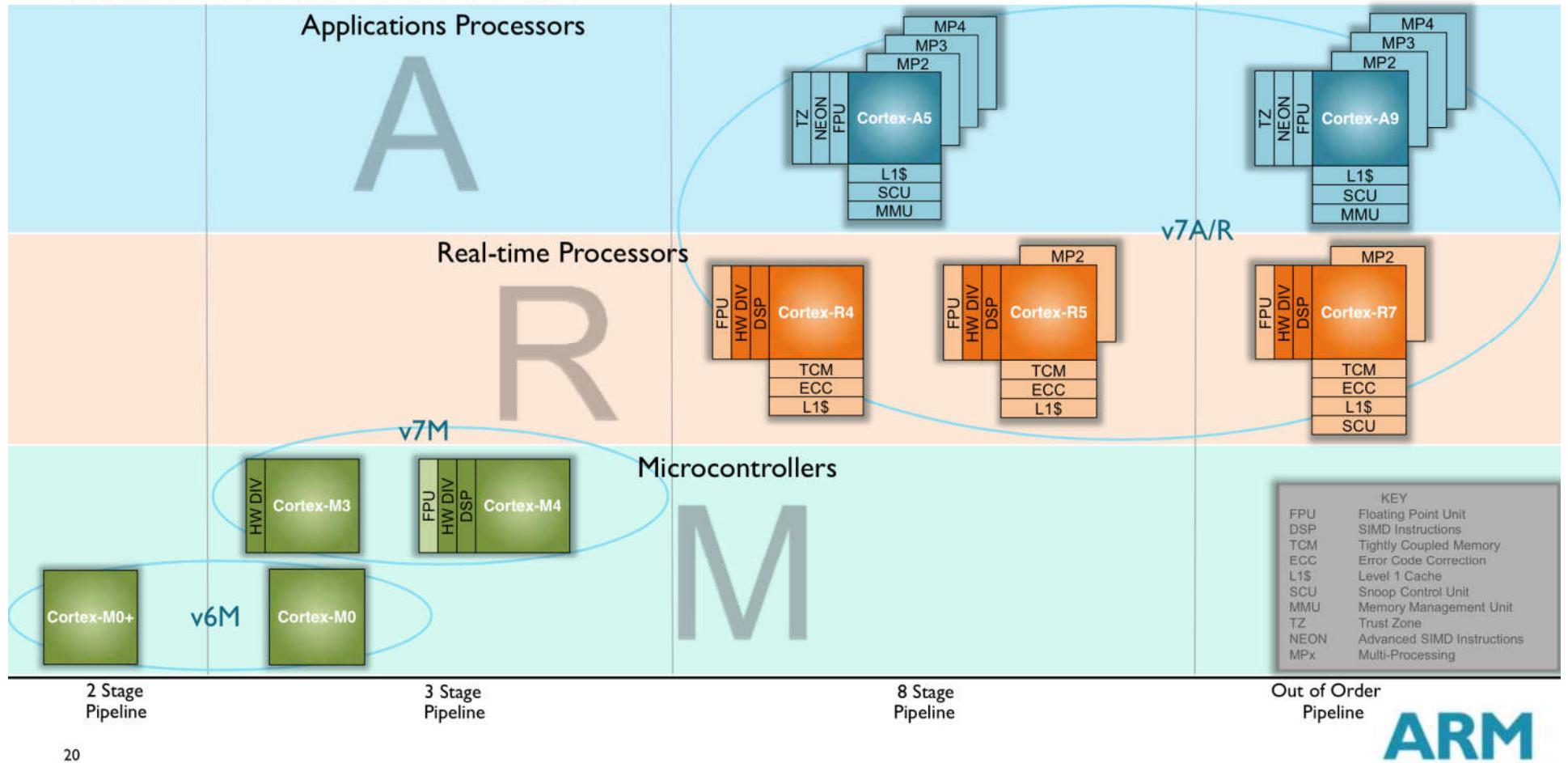
2016 => ARM foi adquirida pelo SoftBank (Japão) por U\$31 bilhões

# ARM Family

64 bits



# ARM Cortex-M Profile

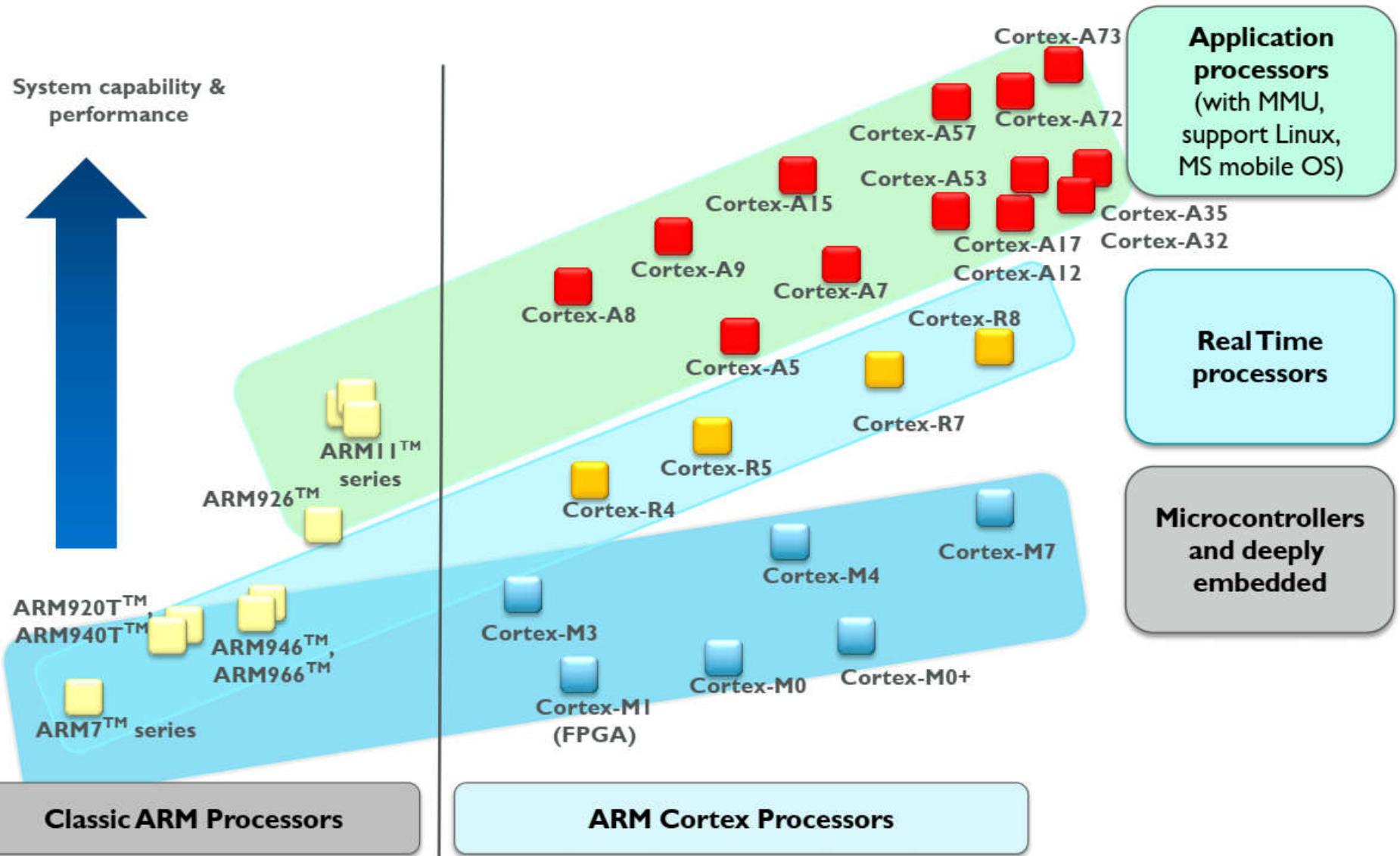


Cortex - A (>1GHz) => Alto desempenho para execução de sistemas operacionais - MMU (smartphone)

Cortex - R (200MHz-1GHz) => Ótimo desempenho para aplicações em tempo real (controle de rede)

Cortex - M (<200MHz) => Soluções de mais baixo custo e baixo consumo (microcontroladores)

# ARM Family



# ARM - Cortex

- A maioria dos processadores Cortex possuem arquitetura Harvard que permite a execução de atividades em paralelo, melhorando seu desempenho.
- Supre core padrão do microcontrolador e periféricos:
  - Sistema de interrupção (NVIC - *Nested Vectored Interrupt Controller* );
  - Timer do processador ( *SysTick timer* );
  - Mapeamento de memória ( MPU - *Memory Protection Unity* );
  - Sistema de depuração ( *Debug Unit* ).

# ARM - Advanced RISC Machine

- No clássico, fabricantes de CIs acrescentam os seus próprios módulos;
- Cortex X Classic:
  - Menor latência e maior previsibilidade no atendimento de interrupções;
  - Melhor suporte para depuração;
  - Menor consumo;
  - Maior desempenho para mesmo clock.
- Cortex facilita a portabilidade de implementações de uma pastilha para outra quando comparado a linha Classic.

- ARM Cortex-M processors: licenciado para mais de 175 fabricantes de semicondutores, dentre os quais:



<http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.dai0237a/index.html>

## AT89C51RD2

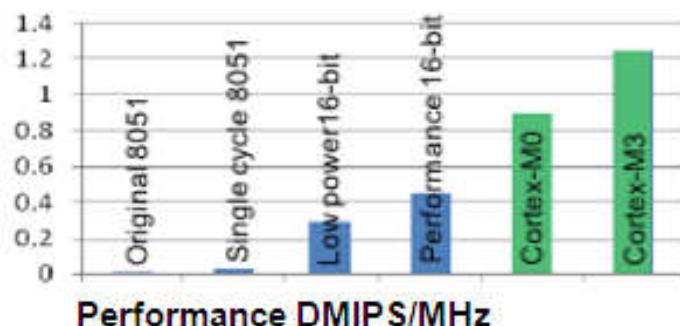
( 1 unidade: US\$ 6,13 )

X

## SAM3S1AB M3 core

( 1 unidade: US\$ 2,85 )

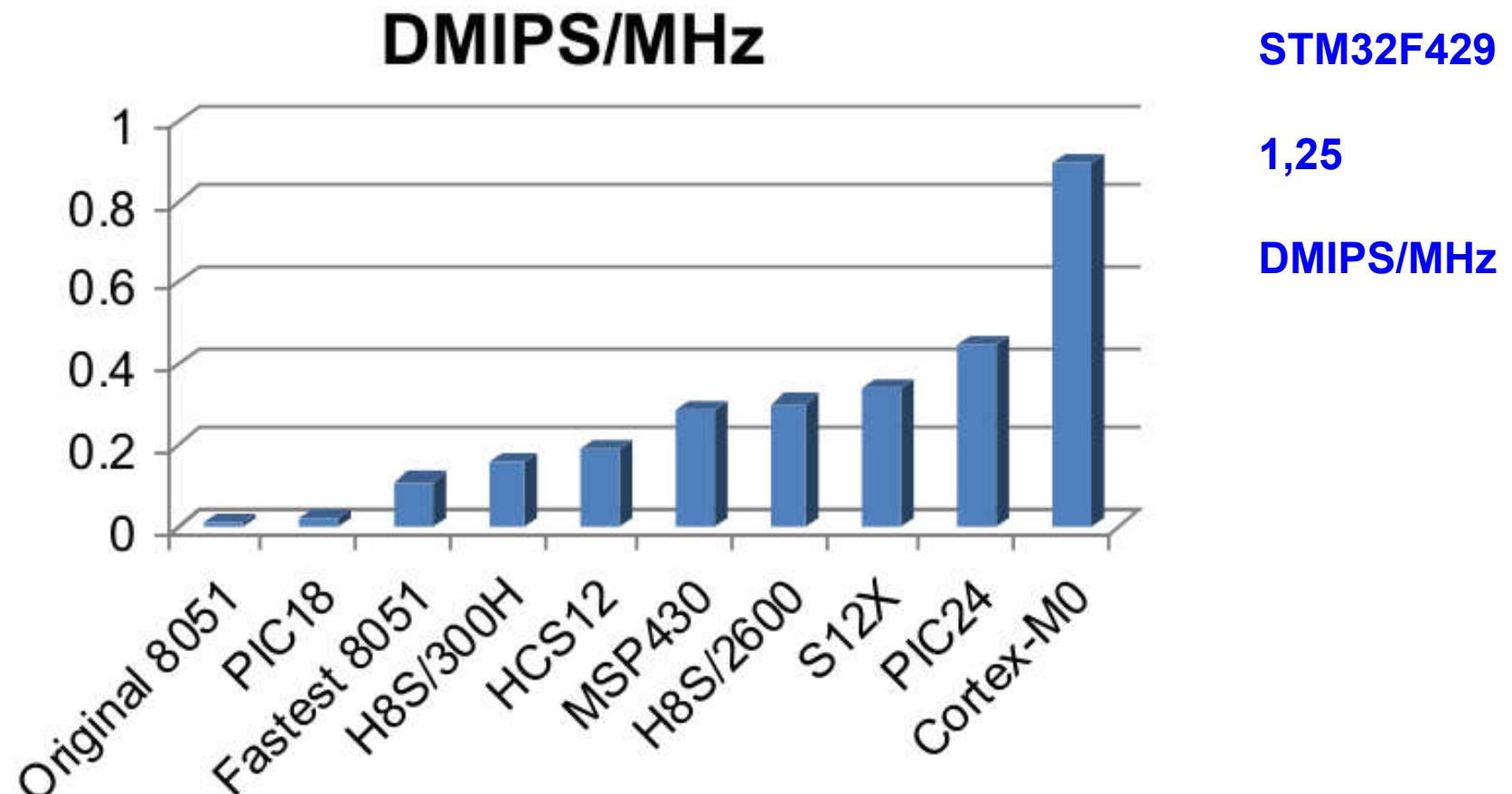
Digikey (05/2018)



<http://www.emcu.it/STM32/WhyUsingCortexM3/WhyUsingCortexMx.html>

	AT89C51RD2	SAM3S1A
Architecture	8051 (8052)	ARMv7-M (Cortex-M3)
Program memory (flash)	64 Kbytes	64 Kbytes
Data memory (RAM)	2 Kbytes	16 Kbytes
EEPROM	2 Kbytes	N
Memory Protection Unit	N	Y
Max clock frequency	60 MHz	64 MHz
GPIO pins	48	34
ADC	N	8-channel x 12-bit
Timers	3 x 16-bit	3 x 16-bit + SysTick
Watchdog timer	Y	Y
SPI	1	1
I2C	N	1
USART	1 (UART)	2
PWM	Y	Y
RTC	Y (up to 5 x 8-bit)	4 channel x 16-bit
External interrupt sources	2 (+ 7 internal)	34 (+ 16 internal)
Interrupt prioritization	4 levels	16 levels
Vectored Interrupt Controller	Y (two separately-vectored external interrupts)	Y
Power-saving modes	Idle/Power-Down	Sleep/Wait/Backup
DMA	N	4-channel
Debug port	ONCE mode	Serial wire or JTAG debug
Voltage Detection	Y	Y

# Comparação de Desempenho



<http://www.embedded.com/print/4008863>

PIC24FJ128GA306 (1 unidade): US\$ 3,51

Digikey (05/2018)

# Glossário

**Dhrystone MIPS (DMIPS):** A indústria adotou a máquina VAX 11/780 como referência de desempenho como máquina 1 MIPS (*Million Instructions Per Second*). Portanto, "10 Dhrystone MIPS" significa velocidade 10 vezes maior que a VAX 11/780.

**Single instruction multiple data (SIMD):** Mesma operação simultânea em múltiplos dados

**Processador:** máquina de estados alocada no sistema a que se destina controlar, lendo, decodificando e executando instruções. **Microcontrolador** contém os periféricos necessários em um único chip para controlar um sistema.

**Embedded Processor** demanda baixo consumo de energia e destina-se a tarefas em tempo real. Tais processadores podem ser carregados com *Real-Time Operating System* (RTOS) para executar código desenvolvido pelo usuário, requerendo para tal, apenas uma **Memory Protection Unity** (MPU), em oposição à **Memory Management Unity** (MMU) disponível em **Application Processor**.

**MMU:** periférico para gerenciar memória, permitindo segmentação, proteção, paginação de memória virtual (decodifica endereços de memória virtual produzido pela CPU em endereços de memória física e gera exceção quando não há mapeamento válido para endereço acessado).

**MPU:** periférico para gerenciar memória que permite sua segmentação com o estabelecimento de diferentes atributos para cada partição.



## ARM e STM32F4

VABS	VADD	VCMP	VCMPE	VCVT	VCVTR	VCV	Floating Point		CVTA	CVTN
PKHBT	PKHTB	QADD	QADD16	QADD8	QASX	QDADD	DSP (SIMD, fast MAC)		VFMA	VCVTP
QSAX	QSUB	QSUB16	QSUB8	QADDC	QADDG	QASXG	QDSUB		VFMS	VCVTM
SHADD16	SHADD8	SHASX	SHSAX	SMLAD	SMLADX	SMLAWB	SEL		VFNMA	VMAXNM
SMLATB	SMLATT	SMLAD	SMLADX	SMLAWB	SMLAWT	SMLAWT	SMLABT		VFNMS	VMINNM
SMLALD	SMLALDX	SMLAWB	SMLAWT	SMLAWT	SMLAWT	SMLAWT	MLALTT		VLDL	VRINTA
Advanced data processing bit field manipulations							MLSLDX		VLDR	VRINTN
ADC	ADD	ADR	AND	ASR	B		SMMILA		VMLA	VRINTP
BFC	BFI	BIC	CBNZ	CBZ	CDP	CDP2	SMMILSR		VMLS	VRINTM
CLREX	CLZ	CMN	CMP	DBG	EOR		SMMUL		VMOV	VRINTX
LDC	LDC2	LDMIA	LDMDB	LDR	LDRB		SMUAD		VMRS	VRINTZ
LDRBT	LDRD				LDRH		SMULBB		VMUL	VSEL
LDRHT	LDRSB				LSL		SMULTB		VNEG	
LSR	MCR				MLA		SMULWT		VNMLA	
MLS	MOV				MRRC		SMUSD		VNMLS	
MRRC2	MUL				MVN	NOP	SSAT16	SSAX	VNMUL	
PLD	PLI				POP	PUSH	SSUB16	SSUB8	VPOP	
REV16	REVSH				RBX	RBIT	SXTAB	SXTAB16	VPUSH	
SBFX	SDIV				SMLAL	SMULL	SXTAH	UADD16	VSQRT	
General data processing I/O control tasks							UADD8		VSTM	
ADC	ADD	ADR	AND	ASR	B		UASX		VSTR	
BIC	BKPT	BL	BLX	BX			UHADD16		VSUB	
CMN	CMP	CPS	DMB	EOR			UMAAL			
DSB	ISB	LDMIA	LDR				UQADD16			
Cortex-M0+							UQADD8			
KEV1D	REVSH	ROR	RSB	SDL	SEV		UQASX			
STMIA	STR	STRB	STRH	SUB	SVC		UQSAX			
SXTB	SXTH	TST	UDF	UXTB	UXTH		UQSUB16			
WFE	WFI	YIELD					UQSUB8			
Cortex-M3							USAD8			
IT							USADA8			
Cortex-M4							USAT16			
Cortex-M7							USAX			
Cortex-M4 FPU							USUB16			
Cortex-M7 FPU							UXTAB			
Cortex-M7 FPU							UXTAH			
Cortex-M7 FPU							UXTB16			

Maioria das instruções são de 16 bits - Thumb

DSC: digital signal controllers

MAC: multiply-accumulate

SIMD: single instruction multiple data

FPU: floating point unit

Cortex-M0/M0+

8/16-bit applications

Cortex-M3

16/32-bit applications

Cortex-M4

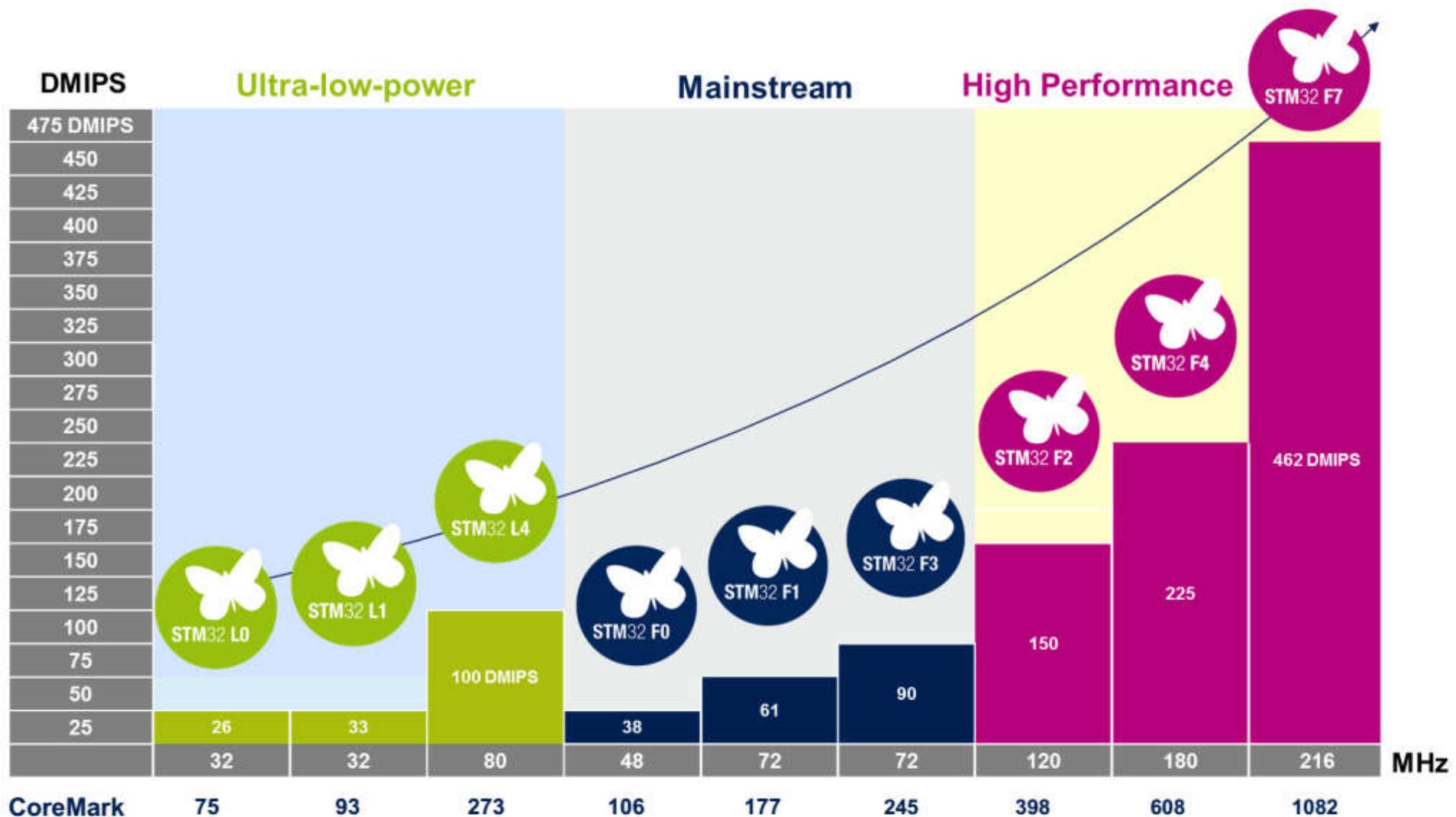
16/32-bit DSC applications

Cortex-M7

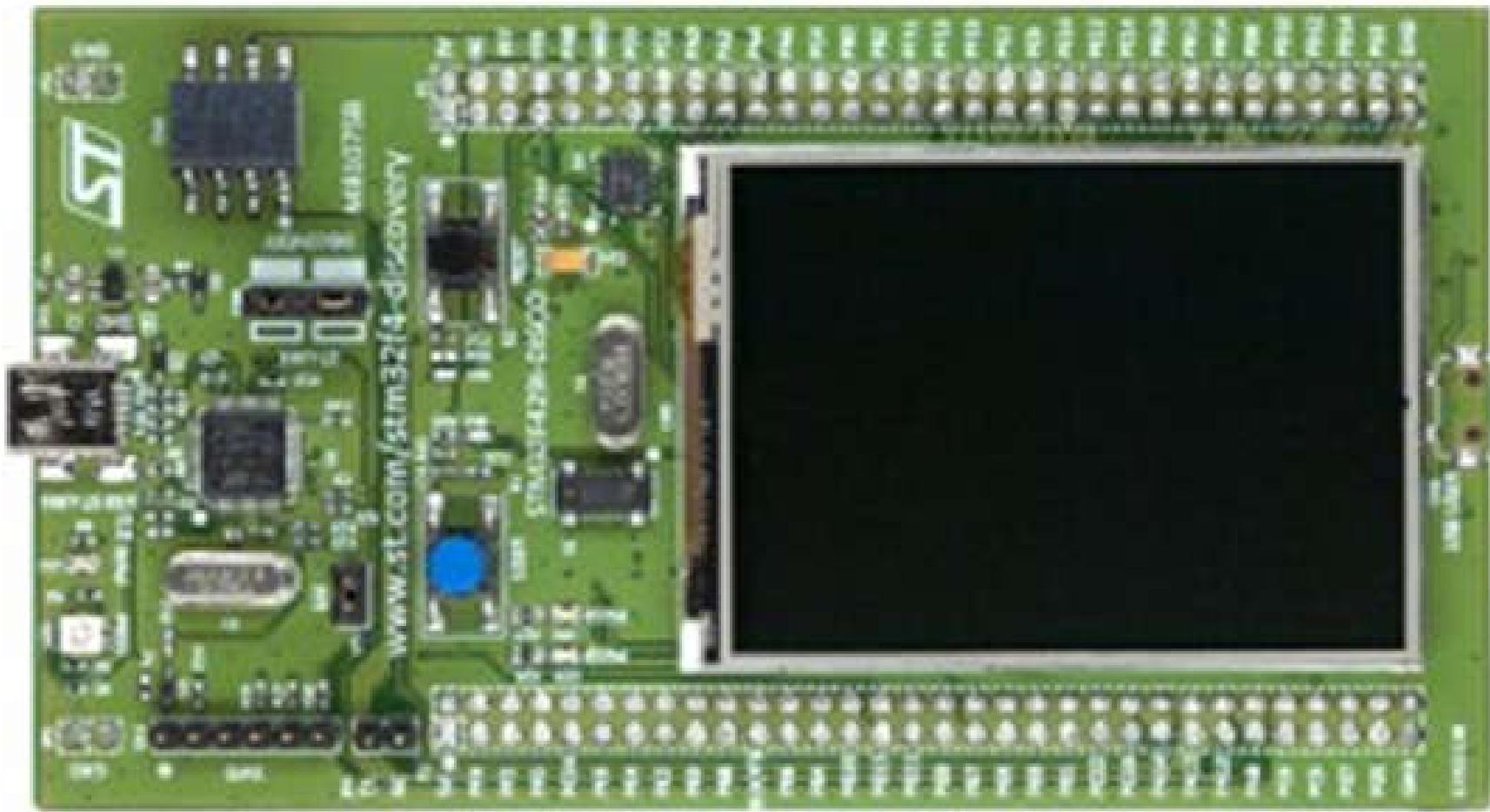
16/32-bit DSC applications

Binary and tool compatible

# ARM Cortex da ST



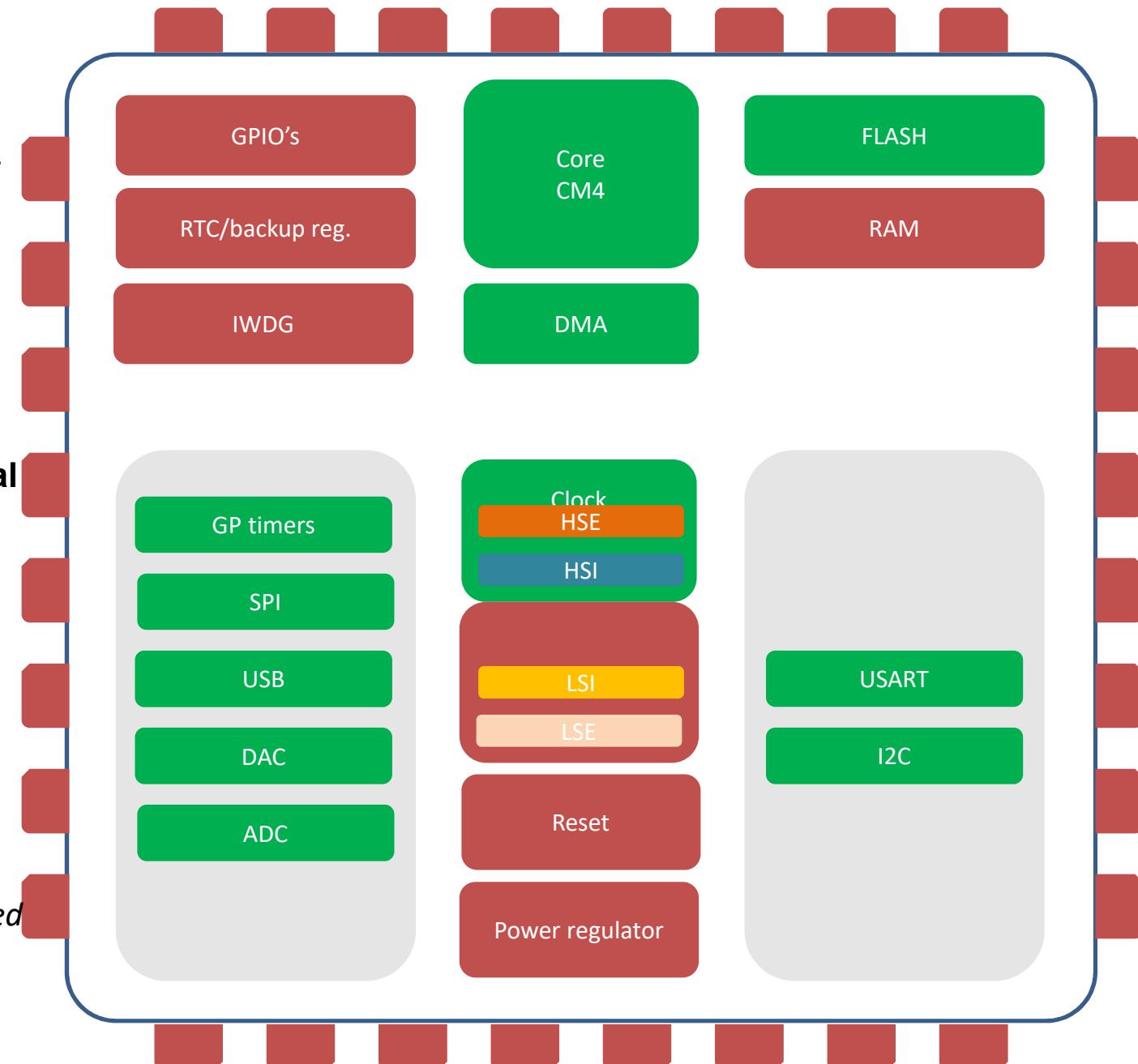
# STM32F429 Discovery (Cortex-M4)



U\$ 29,90 – Digikey 10/17

# STM32F4xx Block Diagram (Cortex-M4)

- 180MHz / 225 DMIPS
- Até 2 MiB Memória Flash
- 256 KiB SRAM (com 64 KiB CCM)
- 4 KiB SRAM de *battery* backup
- 3 ADC de 12 bits (2,4 MSPS)
- 2 DAC de 12 bits
- DMA de propósito geral
- RTC
- Até 17 timers
- Até 168 pinos de I/O que podem gerar interrupções
- Até 21 interfaces de comunicação



OBS: RAM CCM (*Core Coupled Memory*) é usada exclusivamente pela UCP (pilha, por exemplo).

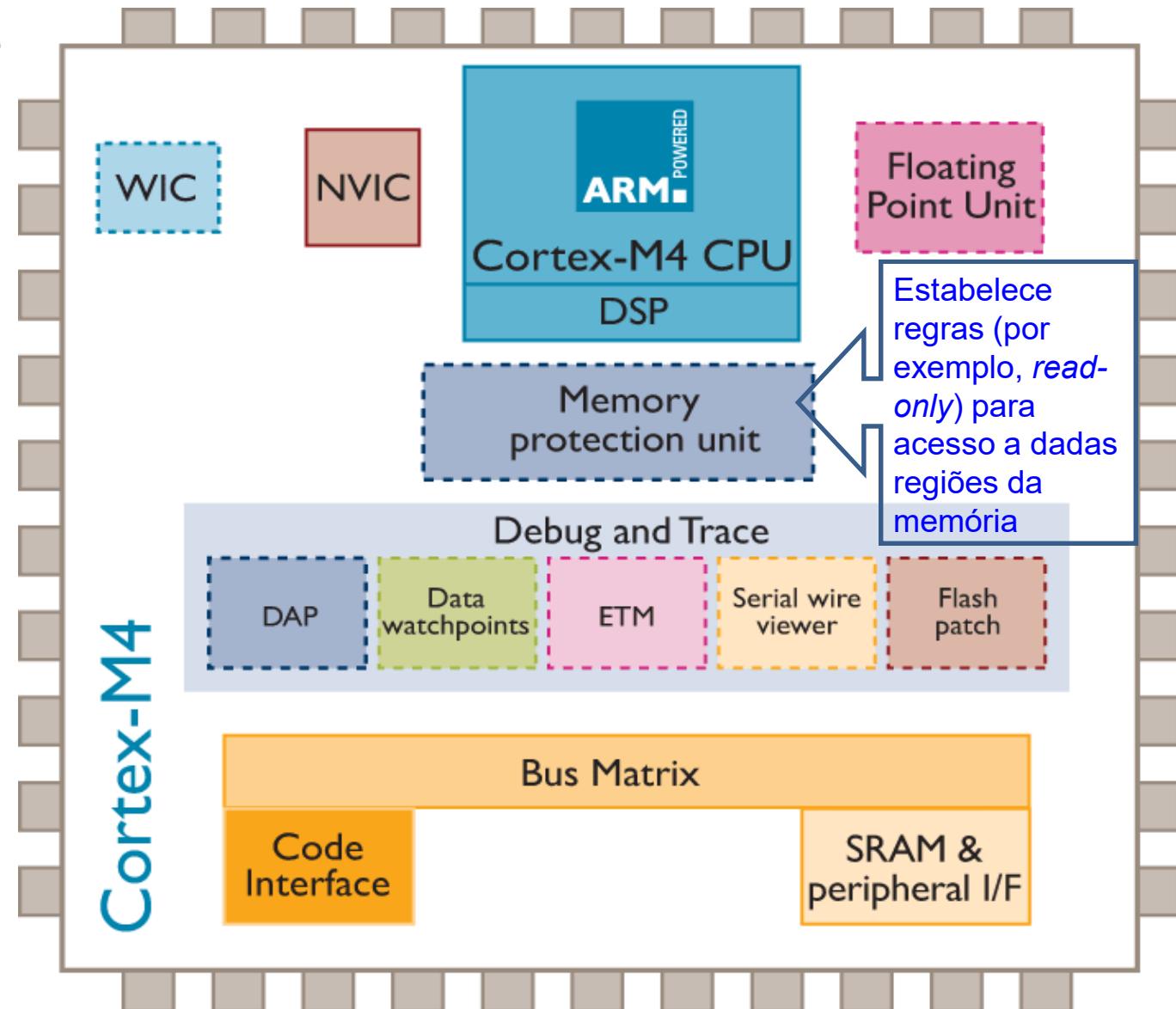
## ULA de ponto flutuante

**WIC:** Wake-up Interrupt Controller  
(modo *low power*)

**NVIC:** Nested Vectored Interrupt Controller

**DAP:** Debug Access Port

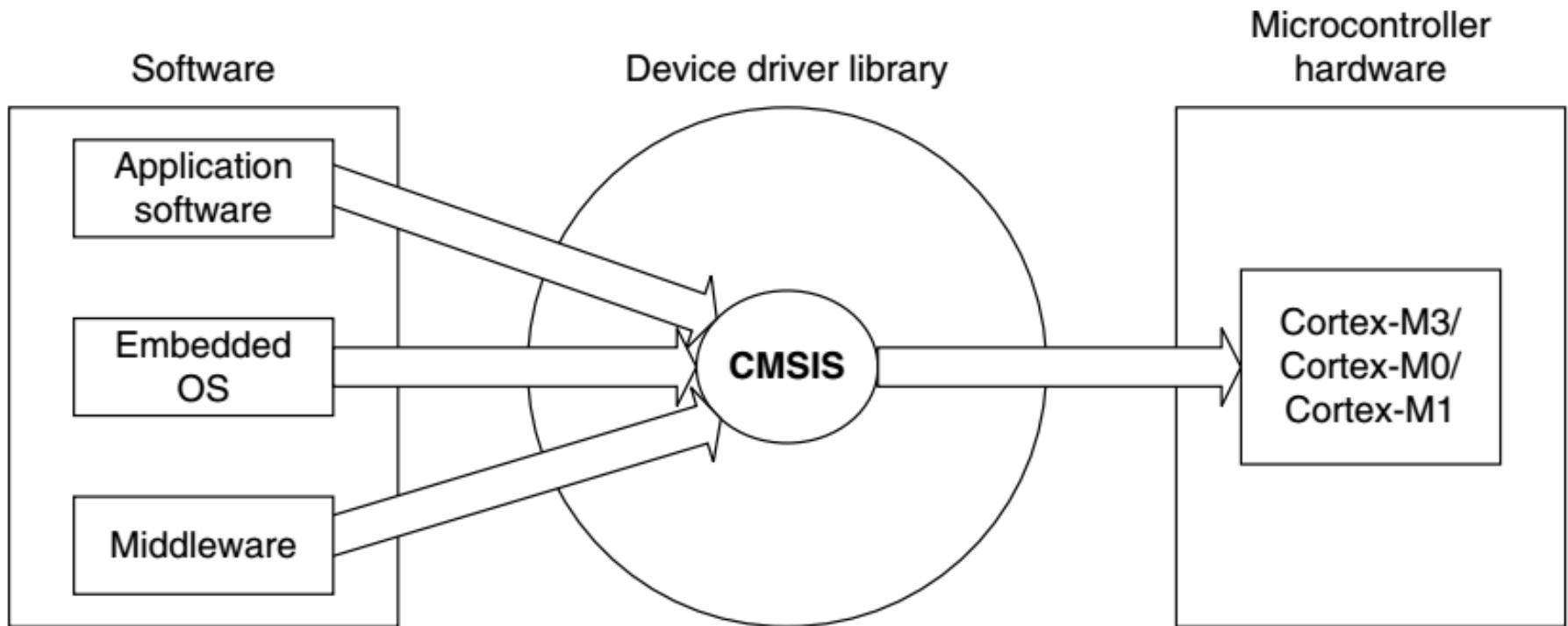
**ETM:**  
Embedded Trace Macrocell



UCPs Cortex possuem sistema de depuração integrado por meio de CMSIS (**Cortex Microcontroller Software Interface Standard**)

# CMSIS - CORE

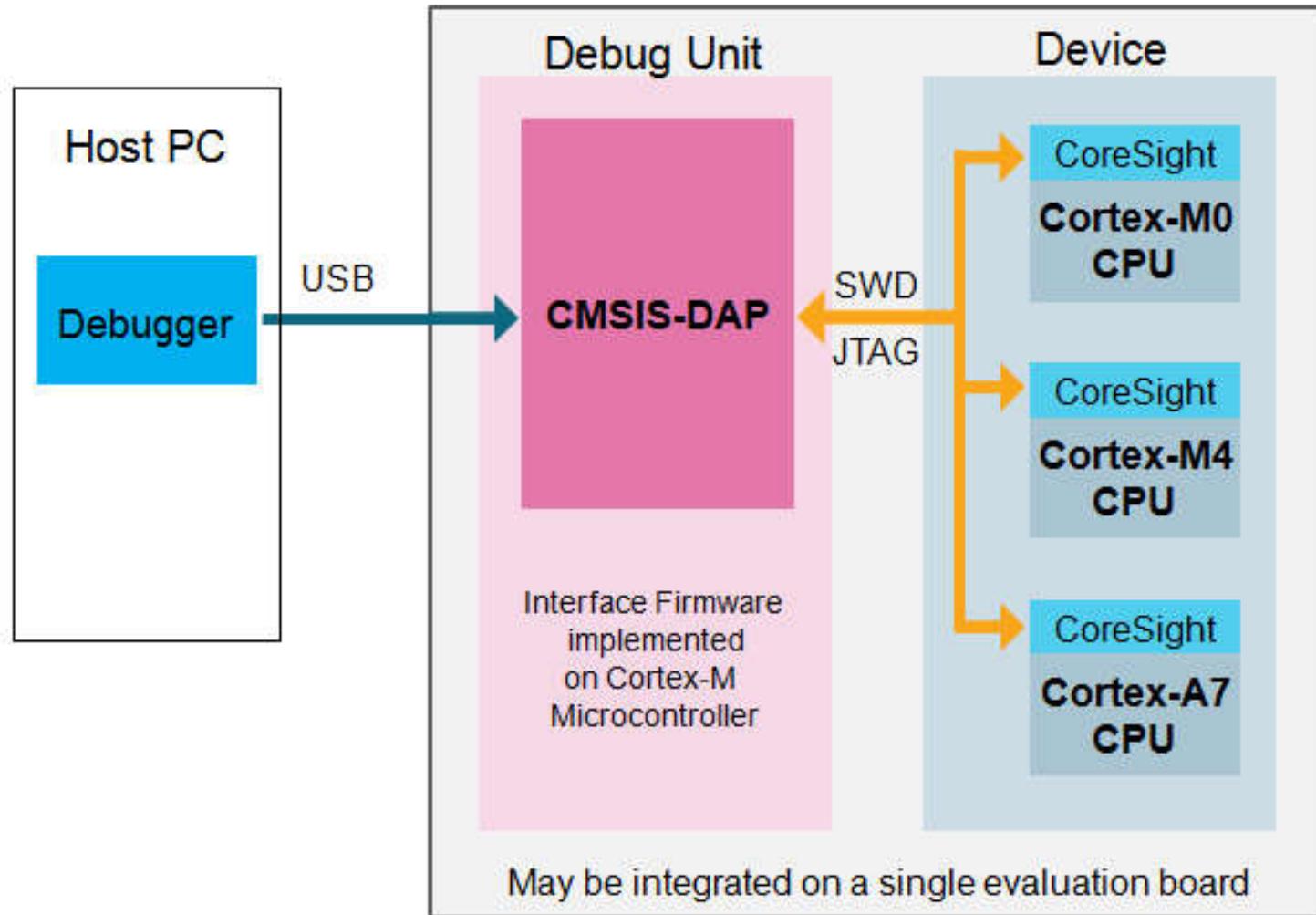
*Cortex Microcontroller Software Interface Standard*



Biblioteca ARM (*Hardware Abstraction Layer*) para o acesso ao processador e aos seus periféricos => Presente em Cls de diferentes fabricantes => facilita portabilidade entre Cls de diferentes fabricantes e entre diferentes CORTEX.

# CMSIS-DAP: Debug Access Port

## Cortex Microcontroller Software Interface Standard



CMSIS-DAP: *firmware* para interface de USB a *Debug Access Port* (DAP) tipicamente por JTAG (5 pinos) ou *Serial Wire Debug* (SWD - 2 pinos: TCLK and TDI )

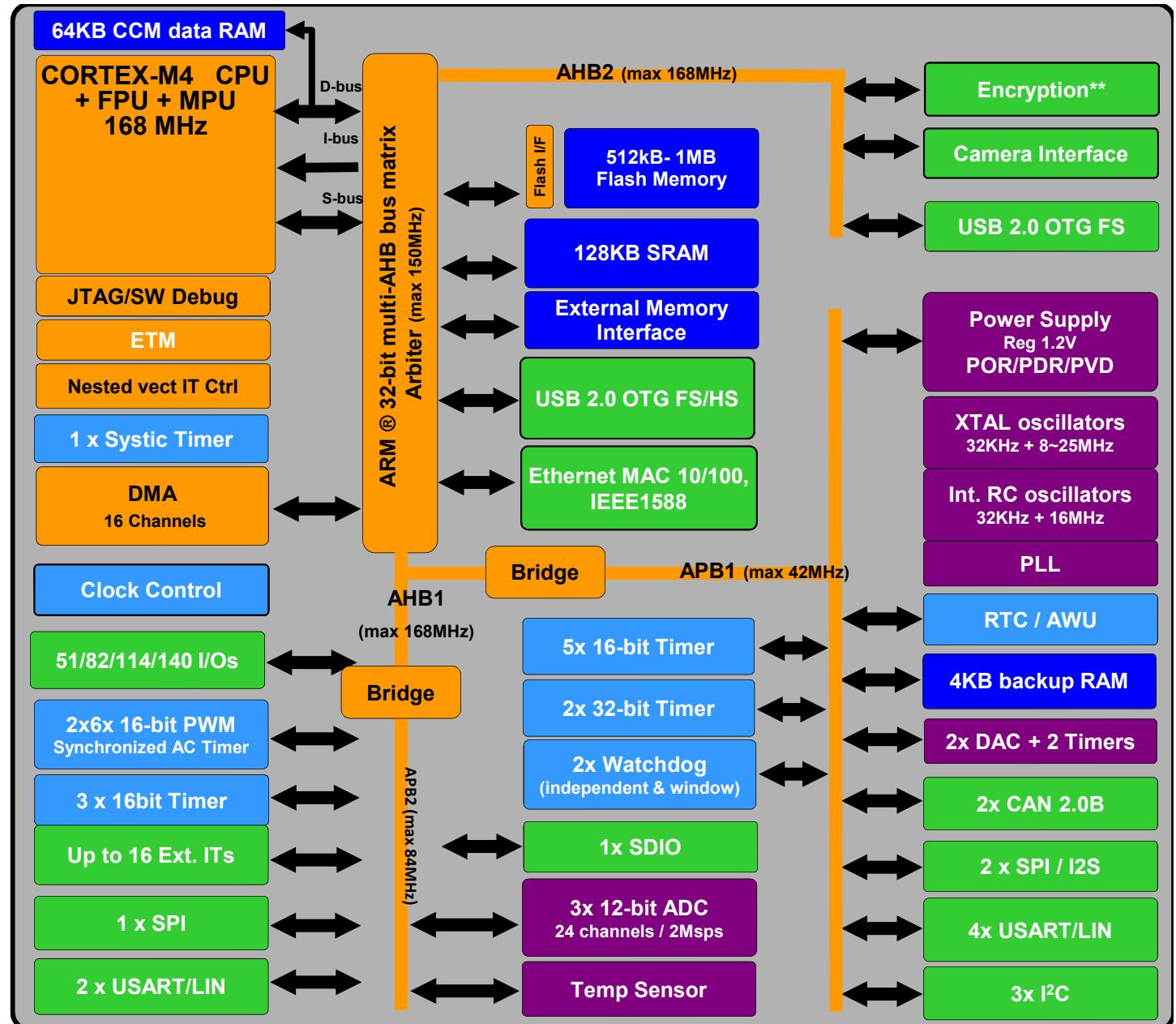
# STM32F4xx

## Block Diagram

A RAM **CCM** (*Core Coupled Memory*) é usada exclusivamente pela UCP (pilha, por exemplo).

M4 tem 3 barramentos:

- *Instruction (I)*, *Data (D)* e *System (S)*.

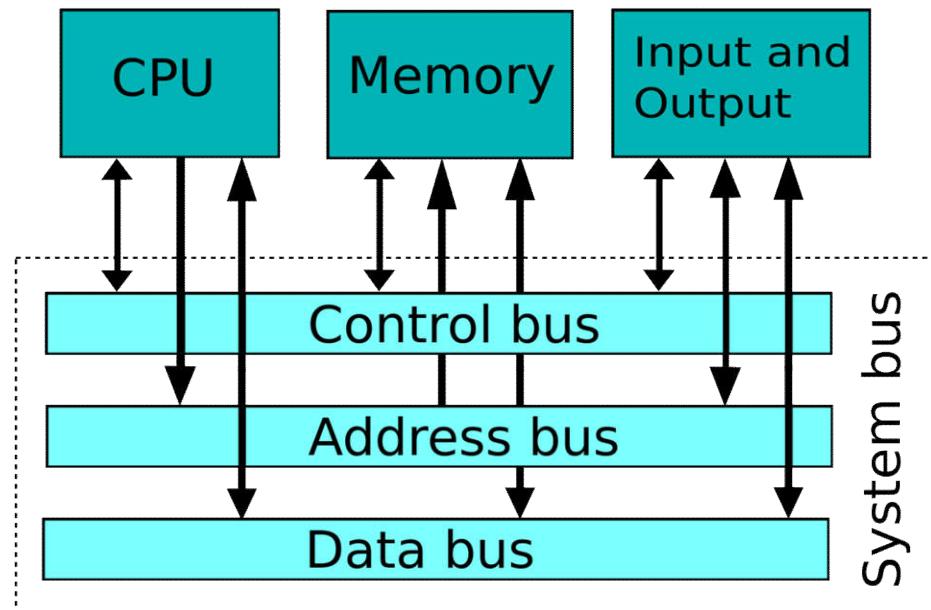


- **I-Bus** : acesso à código (Flash/FMC)
- **D-Bus**: acesso à dados (CCM/SRAM/Flash/FMC)
- **S-Bus**: acesso à periféricos/SRAM/FMC

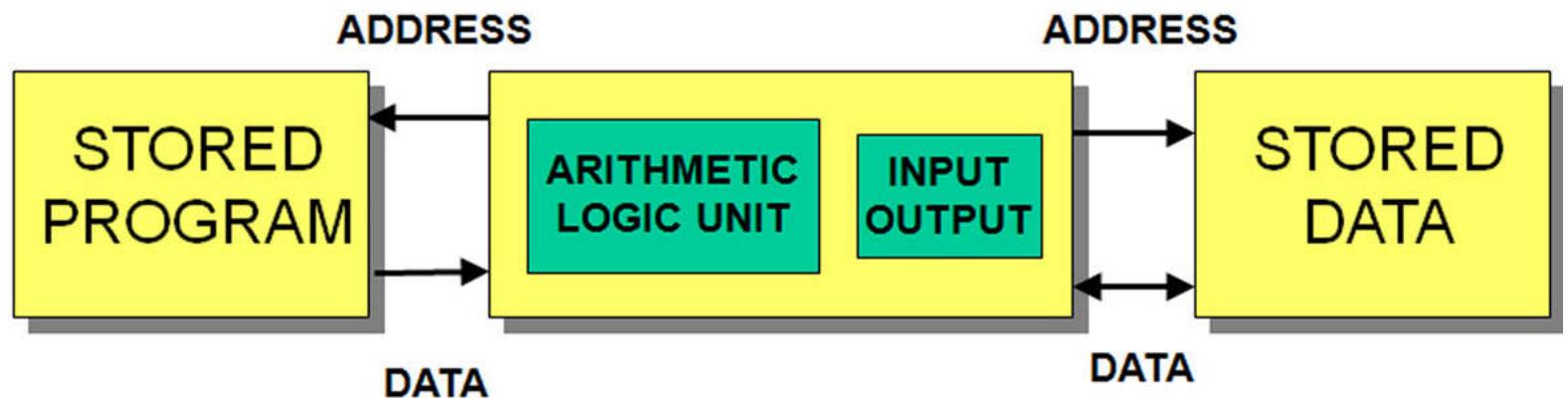
**AHB:** Advanced High-performance Bus  
**APB:** Advanced Peripheral Bus

FMC: Flexible Memory Controller (slide anterior): traduz atividades no AHB em protocolo adequado para acesso à memória externa

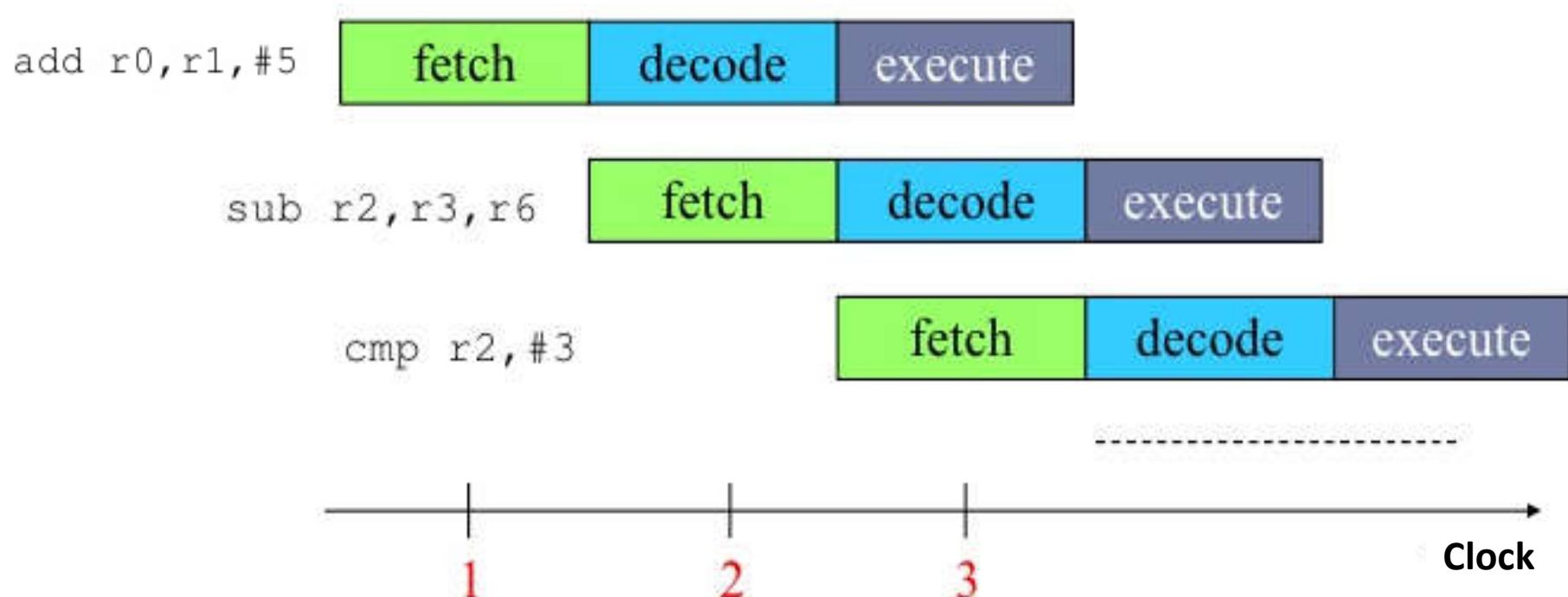
**Arquitetura Von Neuman:**  
Mesmos barramentos para acesso às memórias de dados e código



**Arquitetura Harvard:**  
Barramentos distintos para acesso às memórias de dados e código

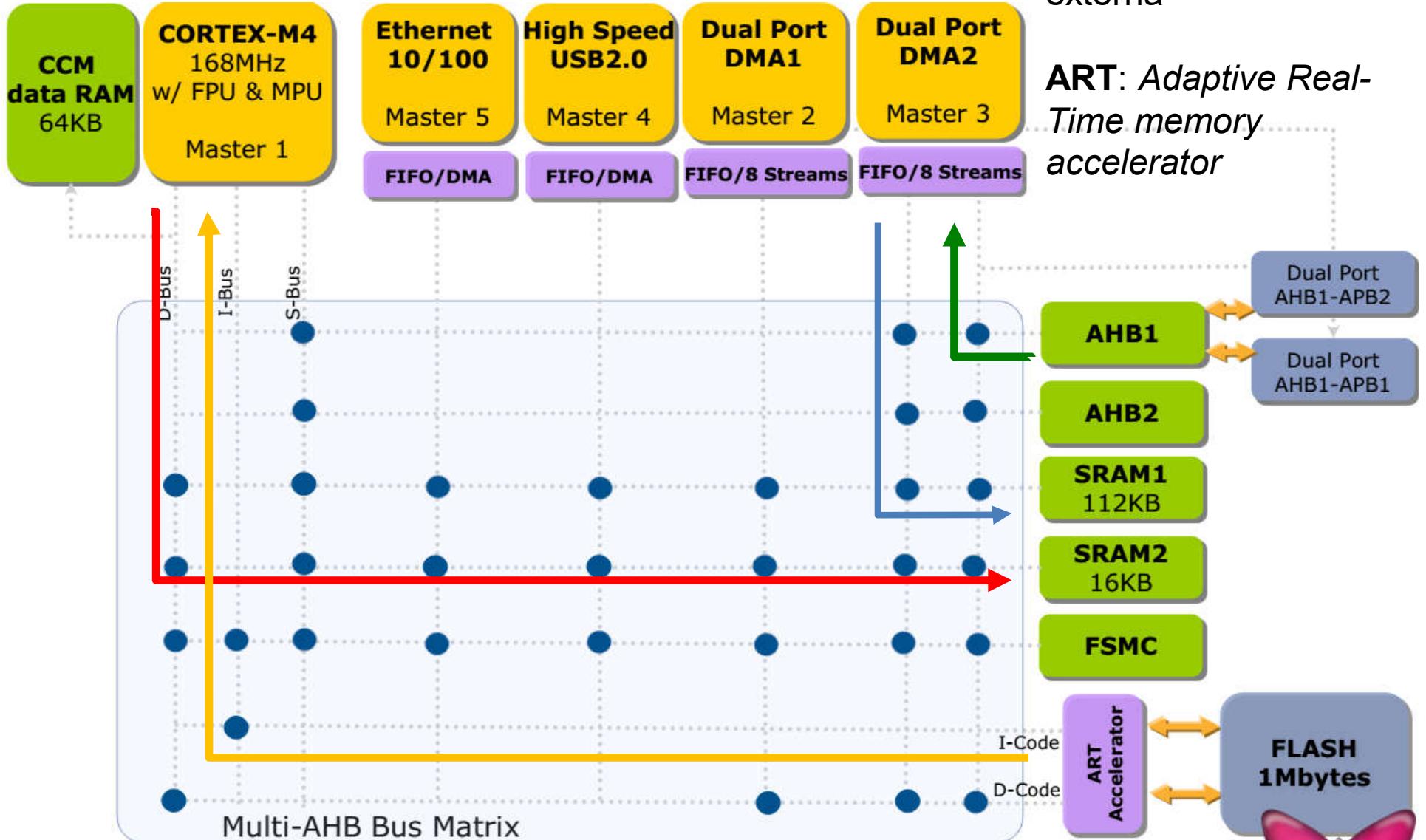


## **3 Stage Pipeline**



# System Architecture

Bus Matrix: Arbitra o acesso a periféricos entre *core* e DMA



- **FSMC:** traduz atividades no AHB em protocolo adequado para periférico externo

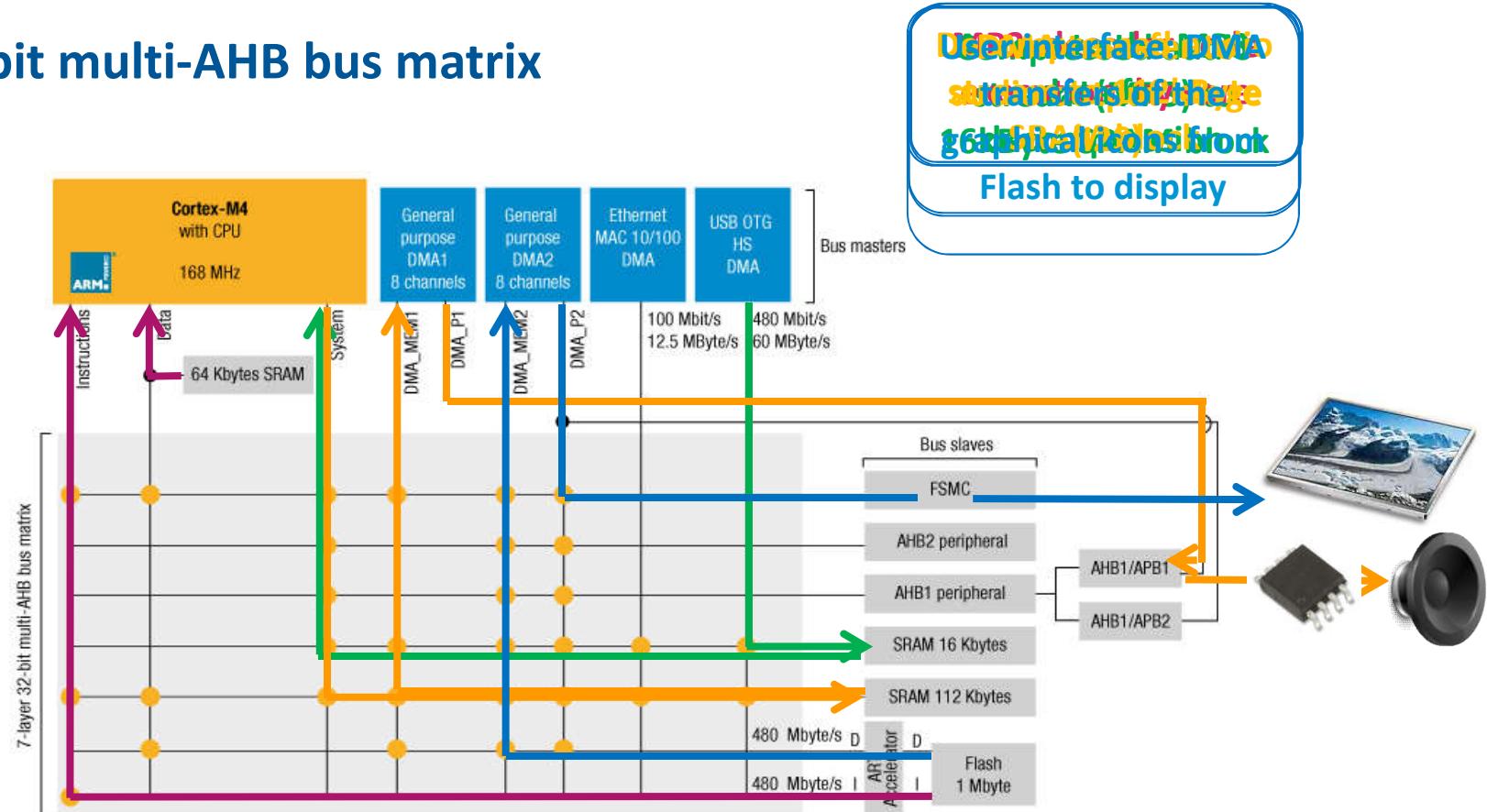
**FSMC:** Flexible Static Memory Controller – Acesso à memória externa

**ART:** Adaptive Real-Time memory accelerator



# Real-time performance

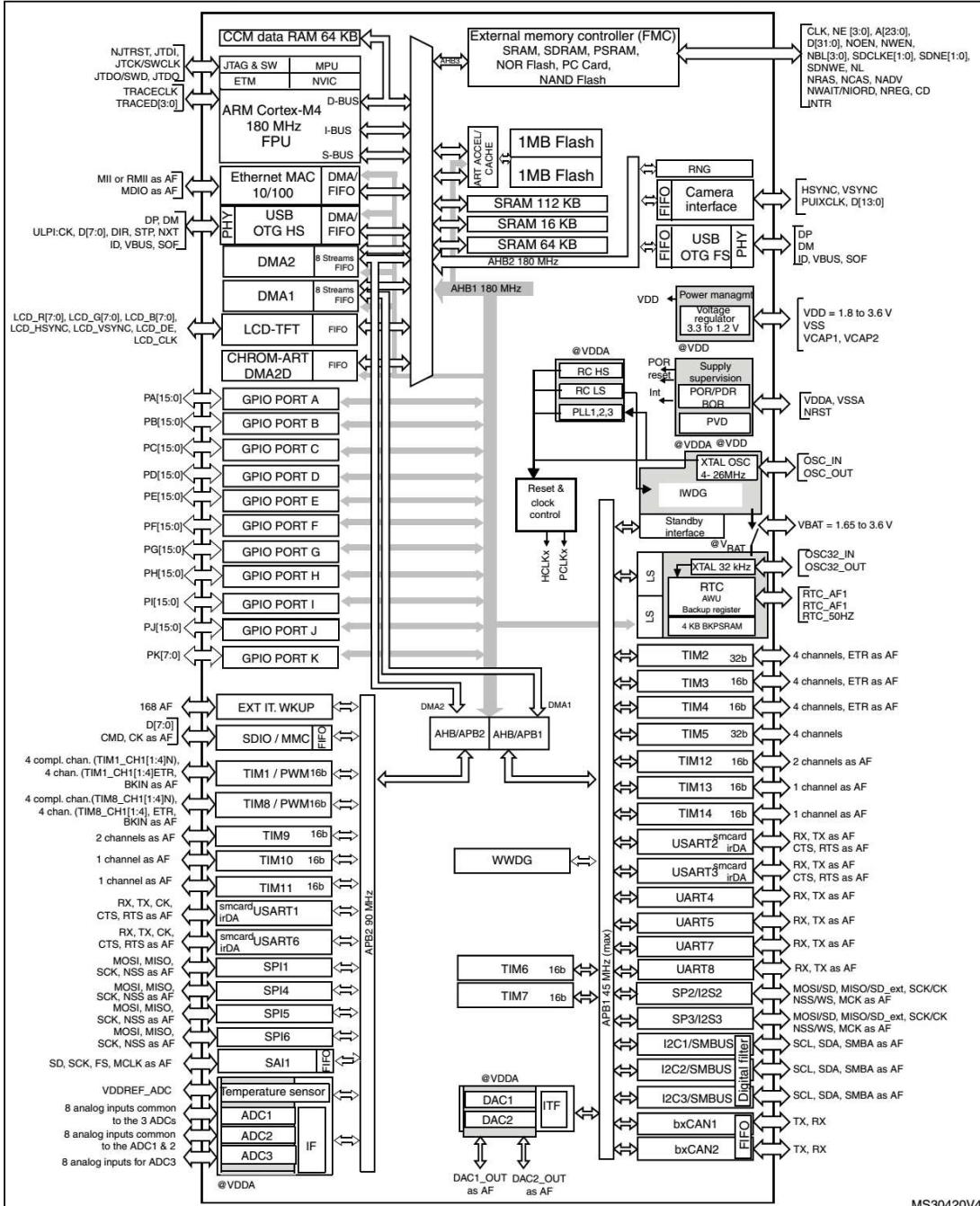
## 32-bit multi-AHB bus matrix



# *Periféricos*

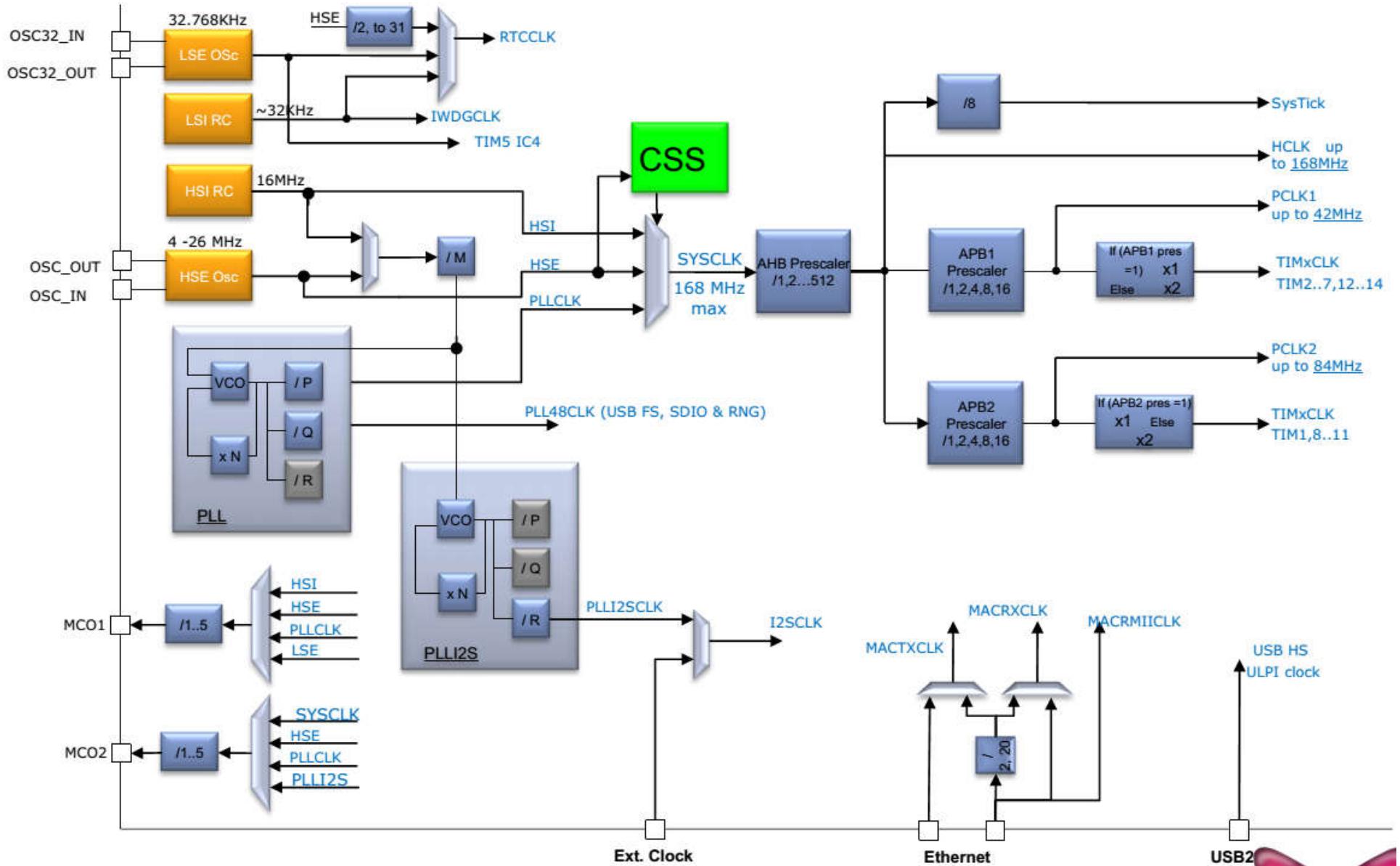
Peripherals	Performance
USB FS / HS	12 Mbit/s / 480 Mbit/s
USART	Up to 11.25 Mbit/s
SPI	Up to 45 Mbit/s
I <sup>2</sup> C	1Mbit/s
GPIO toggling	Up to 90 MHz
3-phase MC timer	180 MHz PWM timer clock input
SDIO	Up to 48 MHz
I <sup>2</sup> S and SAI	From 8 kHz to 192 kHz sampling frequencies
Camera interface	Up to 54 Mbyte/s at 54 MHz (8- to 14-bit parallel)
Crypto/hash processor	AES-256 up to 149.33 Mbyte/s
FMC	Up to 90 MHz (8-/16-/32-bit data bus, supports SRAM, PSRAM, NAND and NOR Flash, SDRAM, parallel graphic LCD)
12-bit ADC / 12-bit DAC	0.41 µs (2.4 MSPS, 7.2 MSPS in Interleaved mode) / 1 MSPS dual DAC
CAN 2.0B	Up to 2 independent CAN
Ethernet	10/100 Mbit/s MAC with hardware IEEE 1588
LCD TFT controller	Display size : QVGA, QWVGA, VGA, SVGA with 2-layer blending and dithering

**Figure 4. STM32F427xx and STM32F429xx block diagram**



1. The timers connected to APB2 are clocked from  $\text{TIM}_x\text{CLK}$  up to 180 MHz, while the timers connected to APB1 are clocked from  $\text{TIM}_x\text{CLK}$  either up to 90 MHz or 180 MHz depending on the  $\text{TIMPRE}$  bit configuration in the  $\text{RCC\_DCKCFGR}$  register.

# Geração de clock para diferentes barramentos e periféricos

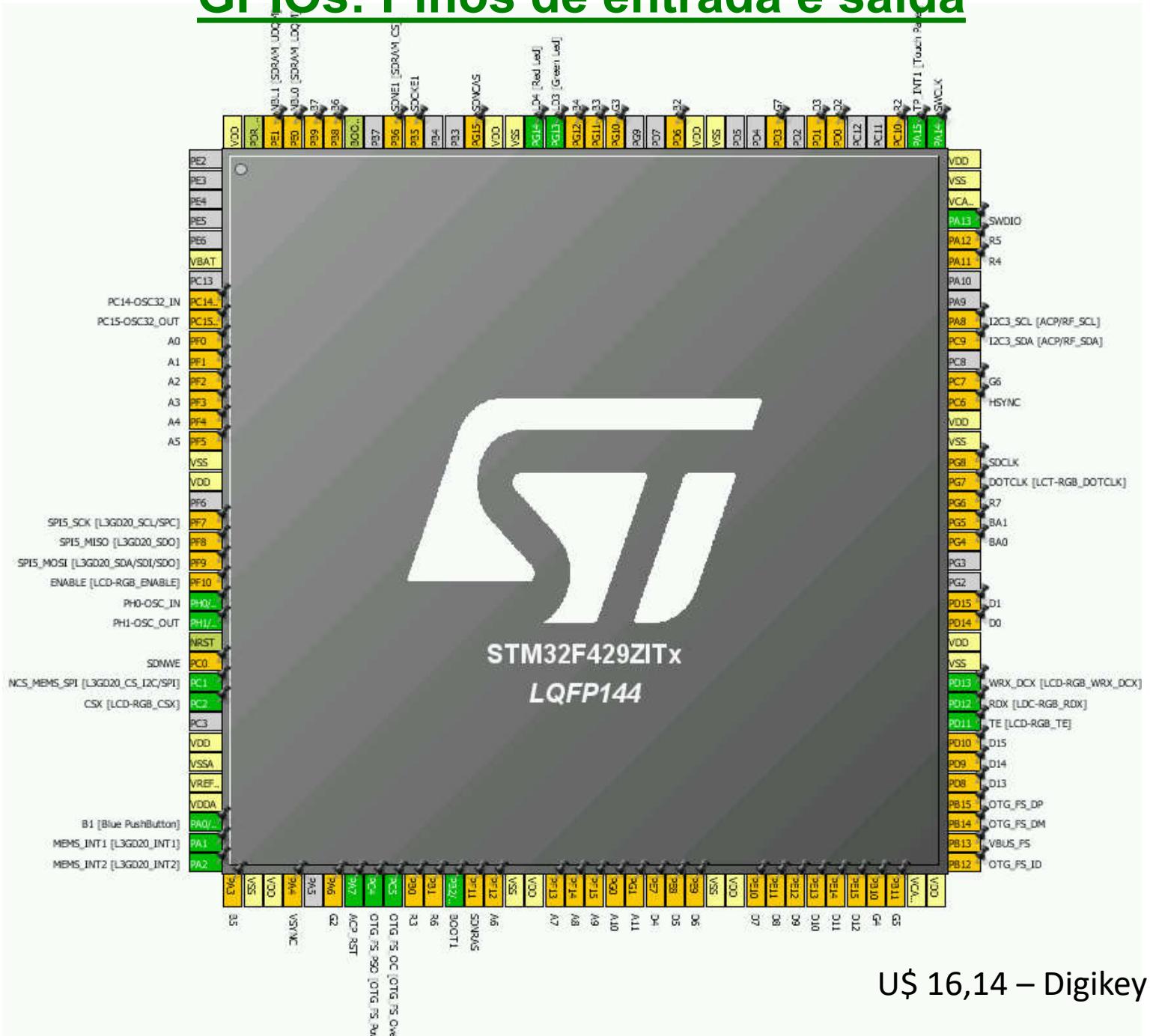


# Geração de clock - Legenda

- **HSE** (*High Speed External oscillator*): 4MHz a 26MHz
- **HSI** (*High Speed Internal RC*): oscilador RC interno - 16MHz +/- 1
- **LSI** (*Low Speed Internal RC*): oscilador RC interno – 32kHz
  - usado para IWDG e, opcionalmente, para o RTC
- **LSE** (*Low Speed External oscillator*): 32,768kHz

*Independent Watchdog Timer* (IWDG): reiniciado pelo processador em intervalos regulares; estouro do WDG revela travamento do sistema, forçando reset.

# GPIOs: Pinos de entrada e saída



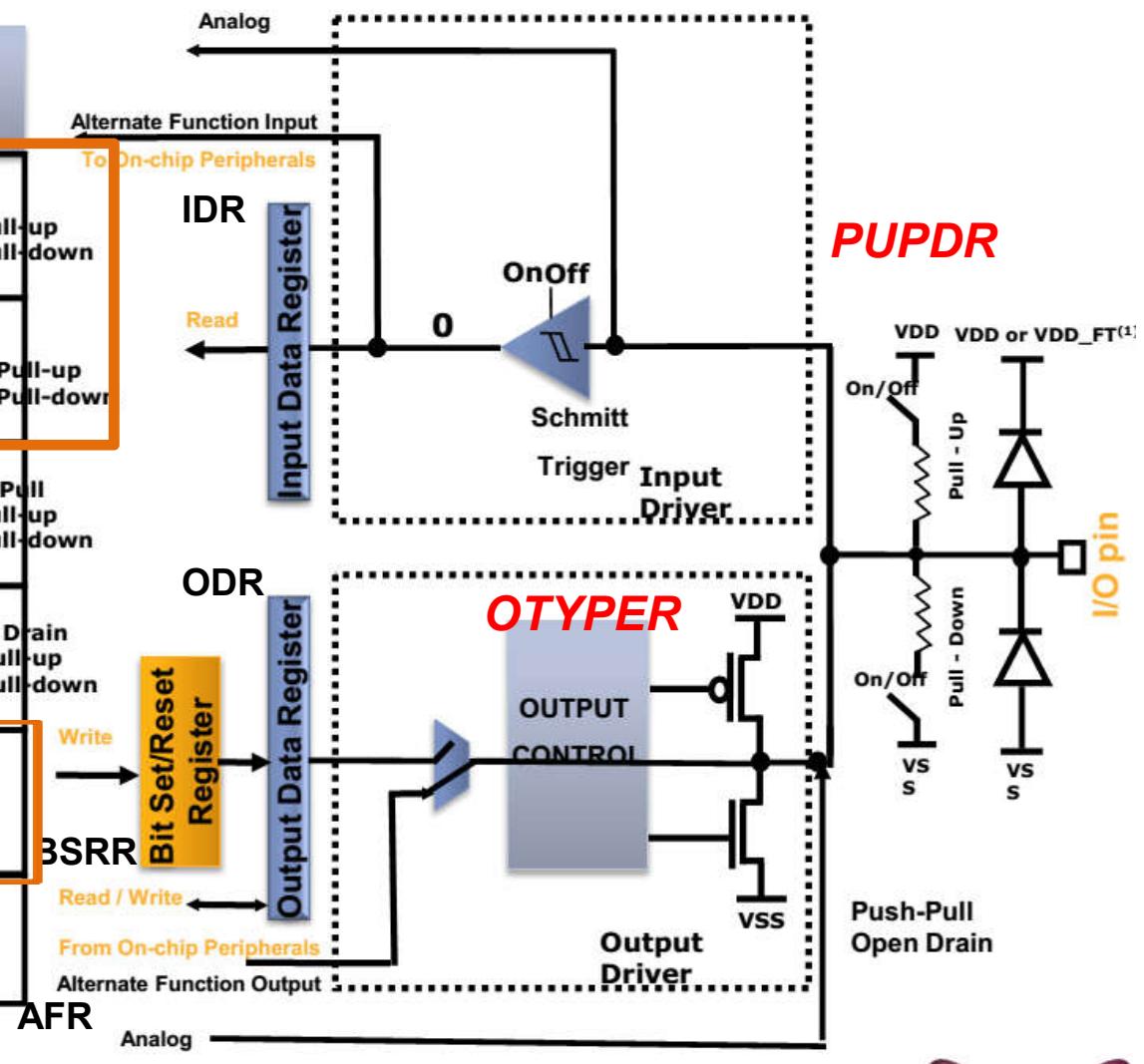
U\$ 16,14 – Digikey 10/17

# GPIOs: Portas de entrada e saída

- Até 168 pinos bi-directionais de I/O (STM32F4) que podem ser configurados para receber interrupção externa (16 simultaneamente)
- Até 11 portas (GPIOA..GPIOK). Cada uma com 16 pinos de entrada e saída.
- Cada porta possui 10 registradores de configuração:
  - 4 de 32 bits para configuração (GPIOx\_MODER, GPIOx\_OTYPER, GPIOx\_OSPEEDR e GPIOx\_PUPDR)
  - 2 de 16 bits para leitura e escrita de dados (GPIOx\_IDR e GPIOx\_ODR, respectivamente)
  - 1 de 32-bit para *set/reset* de pinos individualmente (GPIOx\_BSRR): BSRR[i] =1 seta; BSRR[i+16]=1 reseta
  - 2 de 32 bits para especificar função alternativa (GPIOx\_AFRH e GPIOx\_AFRL)
  - 1 de 32-bit para *locking* (GPIOx\_LCKR): faz retenção de configuração de pino até reset

# GPIO – Modos de Configuração

	MODER(i) [1:0]	OTYPER(i) [1:0]	PUPDR(i) [1:0]	I/O configuration
01	0	0 0 0 1 1 0		Output Push Pull Output Push Pull with Pull-up Output Push Pull with Pull-down
	1	0 0 0 1 1 0		Output Open Drain Output Open Drain with Pull-up Output Open Drain with Pull-down
10	0	0 0 0 1 1 0		Alternate Function Push Pull Alternate Function PP Pull-up Alternate Function PP Pull-down
	1	0 0 0 1 1 0		Alternate Function Open Drain Alternate Function OD Pull-up Alternate Function OD Pull-down
00	x	0 0 0 1 1 0		Input floating Input with Pull-up Input with Pull-down
11	x	x		Analog mode



\* In output mode, the I/O speed is configurable through OSPEEDR register: 2MHz, 25MHz, 50MHz or 100 MHz

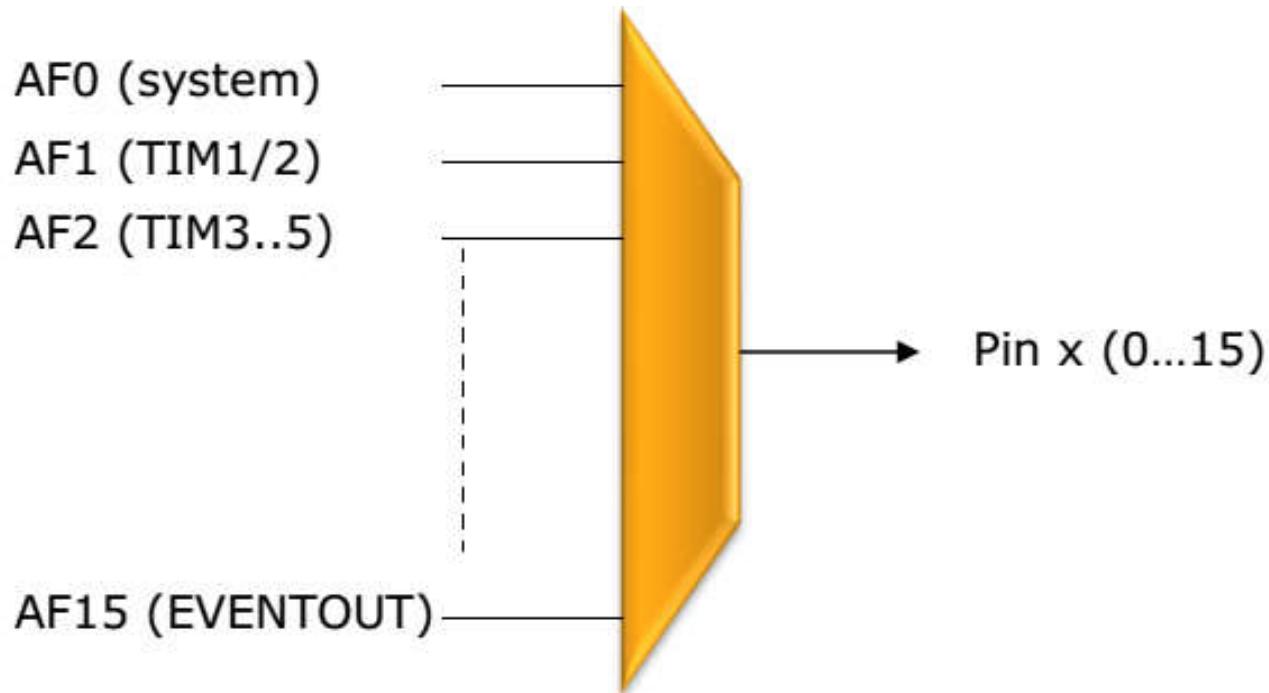
(1) VDD\_FT is a potential specific to five-volt tolerant I/Os and different from VDD.

**Alternate Function (AFx)** multiplexa pinos entre os diferentes periféricos (USARTx, TIMERx, I2Cx, SPIx, ...)

- Para setar/resetar pinos individualmente (**GPIOx\_BSRR**): BSRR[i]=1, seta; BSRR[i+16]=1, reseta.

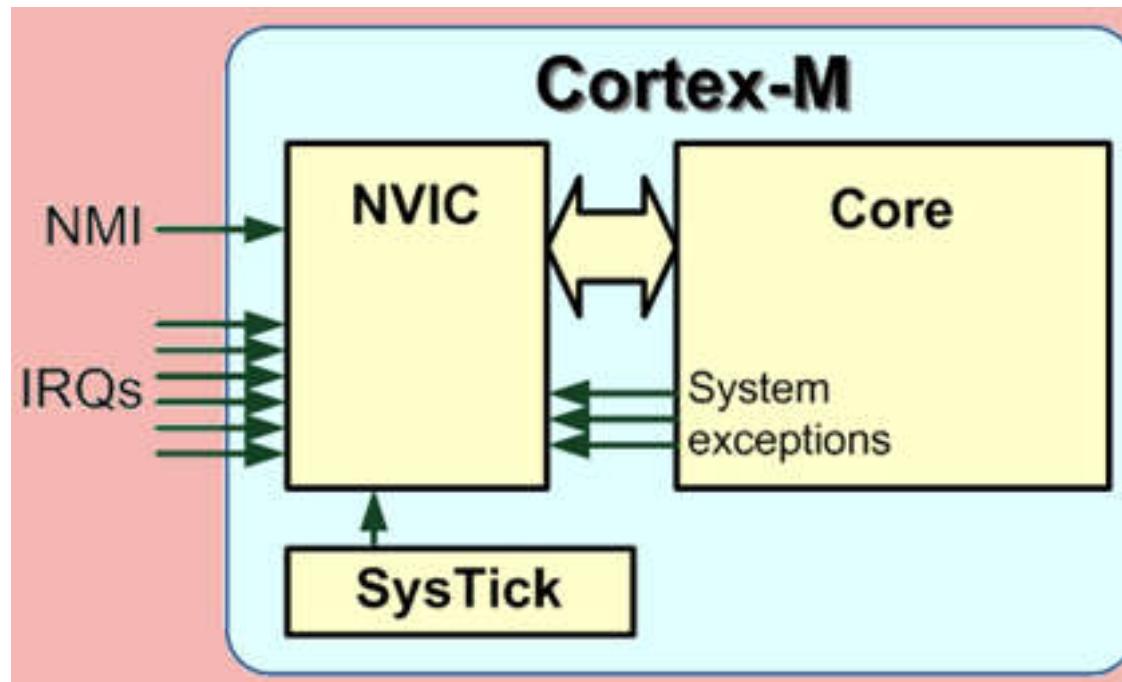


- Registradores Alternate Function (GPIOx\_AFRL e GPIOx\_AFRH):** permitem compartilhar GPIO com diferentes periféricos (4 flip-flops de configuração para cada pino:  $2^4$ : 16; 4 flip-flops x 16 pinos = 64 flip-flops => 2 registradores de 32 flip-flops)

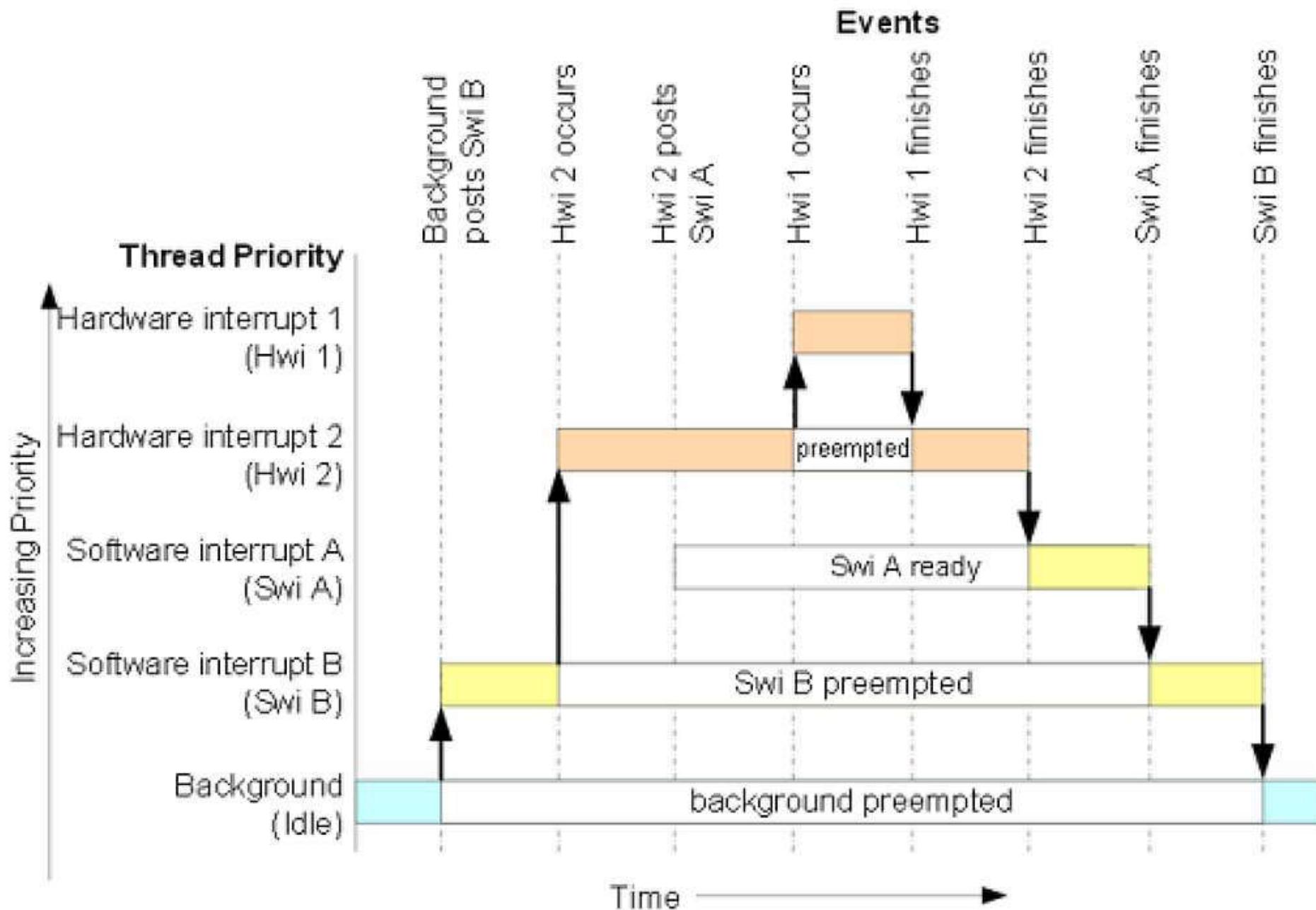


## NVIC-ARM: Nested Vectored Interrupt Controller

(latência de 12 ciclos de clock entre recepção e atendimento)



- O NVIC do Cortex-M4 tem uma interrupção não mascarável (NMI) e até 240 linhas de interrupção externa (IRQs) com 256 níveis de prioridade
- Há 16 fontes de interrupção adicionais dentro do núcleo Cortex que são usadas para exceções do próprio Cortex
- *System timer* (SysTick): Utilizado para fornecer base de tempo para RTOS ou para gerar interrupções periódicas para tarefas agendadas
- O NVIC do STM32F4xx foi implementado com 16 (core) + (até) 91 canais mascaráveis de interrupção com 16 níveis de prioridade

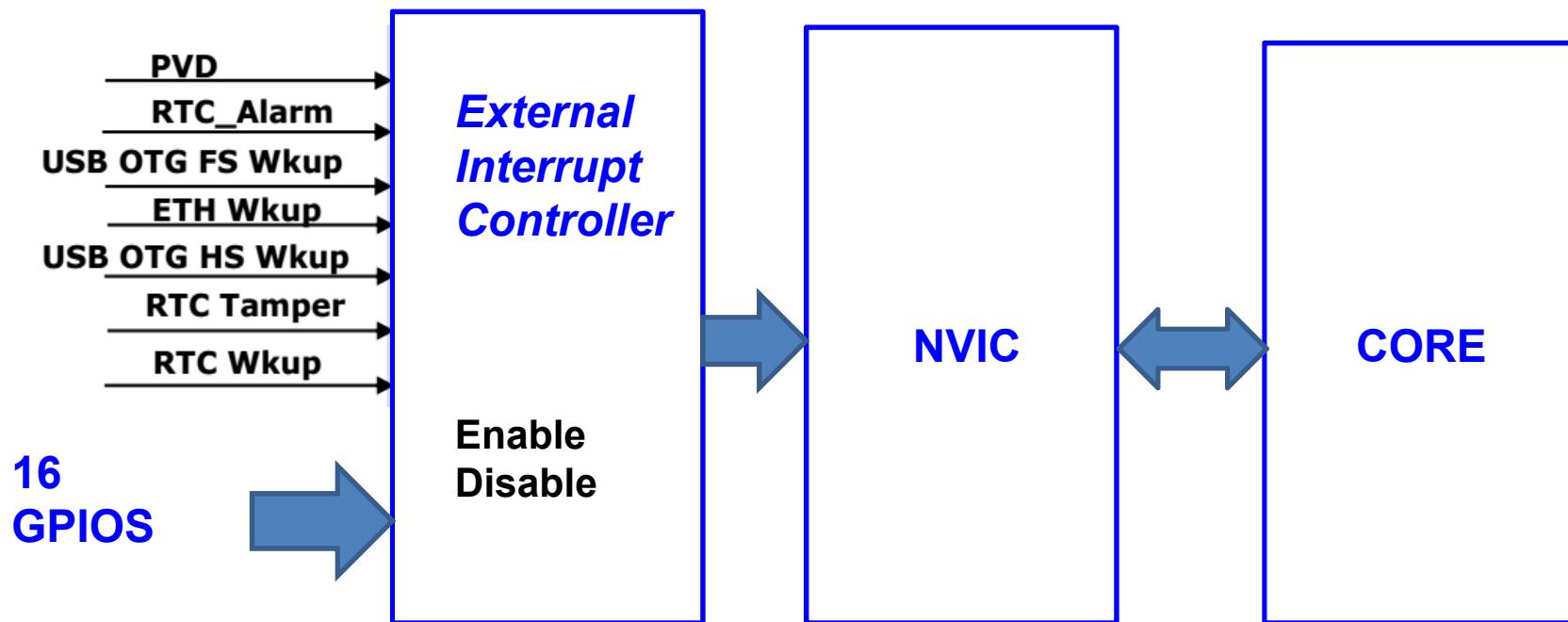


# NVIC – Table

- 16 primeiras fontes de interrupção reservadas para o core
- Endereço inicial da tabela: 0x4;
- OBS: 4 primeiros bytes armazenam endereço do *stack pointer*.

Vector Number	Exception Type	Priority	Vector address	Descriptions
1	Reset	-3	0x04	Reset
2	NMI	-2	0x08	Non-Maskable Interrupt
3	Hard Fault	-1	0x0C	Error during exception processing
4	Memory Management Fault	Programmable	0x10	MPU violation
5	Bus Fault	Programmable	0x14	Bus error (Prefetch or data abort)
6	Usage Fault	Programmable	0x18	Exceptions due to program errors
7-10	Reserved	-	0x1C - 0x28	
11	SVCall	Programmable	0x2C	SVC instruction system service call
12	Debug Monitor	Programmable	0x30	Exception for debug
13	Reserved	-	0x34	
14	PendSV	Programmable	0x38	Pendable request for SVC
15	SysTick	Programmable	0x3C	System Tick Timer
16 and above	Interrupts	Programmable	0x40	External interrupts (Peripherals)

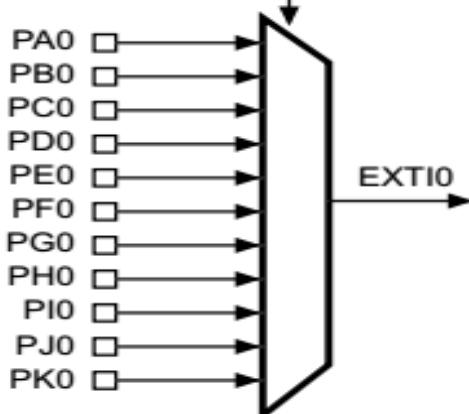
- ❑ A unidade ***External Interrupt Controller*** possui 23 entradas conectadas a vetores de interrupção do NVIC;
- ❑ 16 entradas atendem aos GPIOs, aceitando interrupções por borda (descida, subida e ambos)
- ❑ As outras 7 atendem ao RTC, USB, *Power Voltage Detector* (PWD) e Ethernet (ETH).



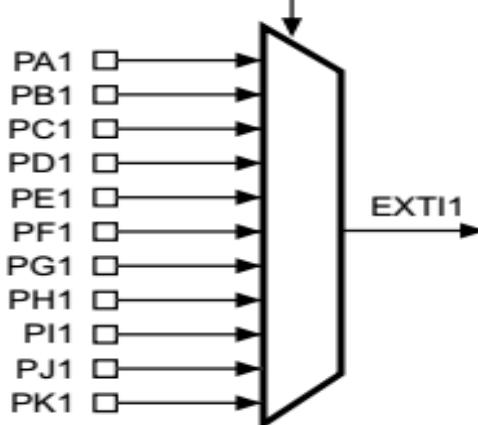
**OBS:** PVD monitora alimentação e o compara a limiares superior e inferior.

Variação da alimentação (+/-) em relação a limiares gera interrupção para proteção do chip

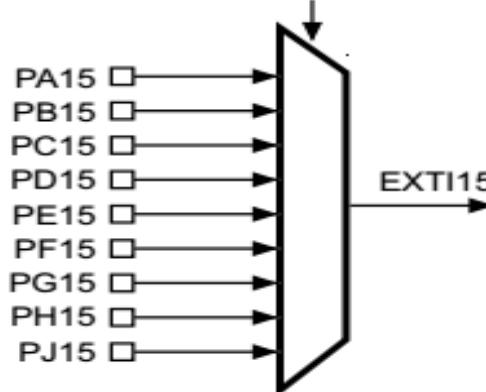
EXTI0[3:0] bits in the SYSCFG\_EXTICR1 register



EXTI1[3:0] bits in the SYSCFG\_EXTICR1 register



EXTI15[3:0] bits in the SYSCFG\_EXTICR4 register



**Seleção de GPIOs para gerar até 16 interrupções  
Folha de dados – Figure 43**

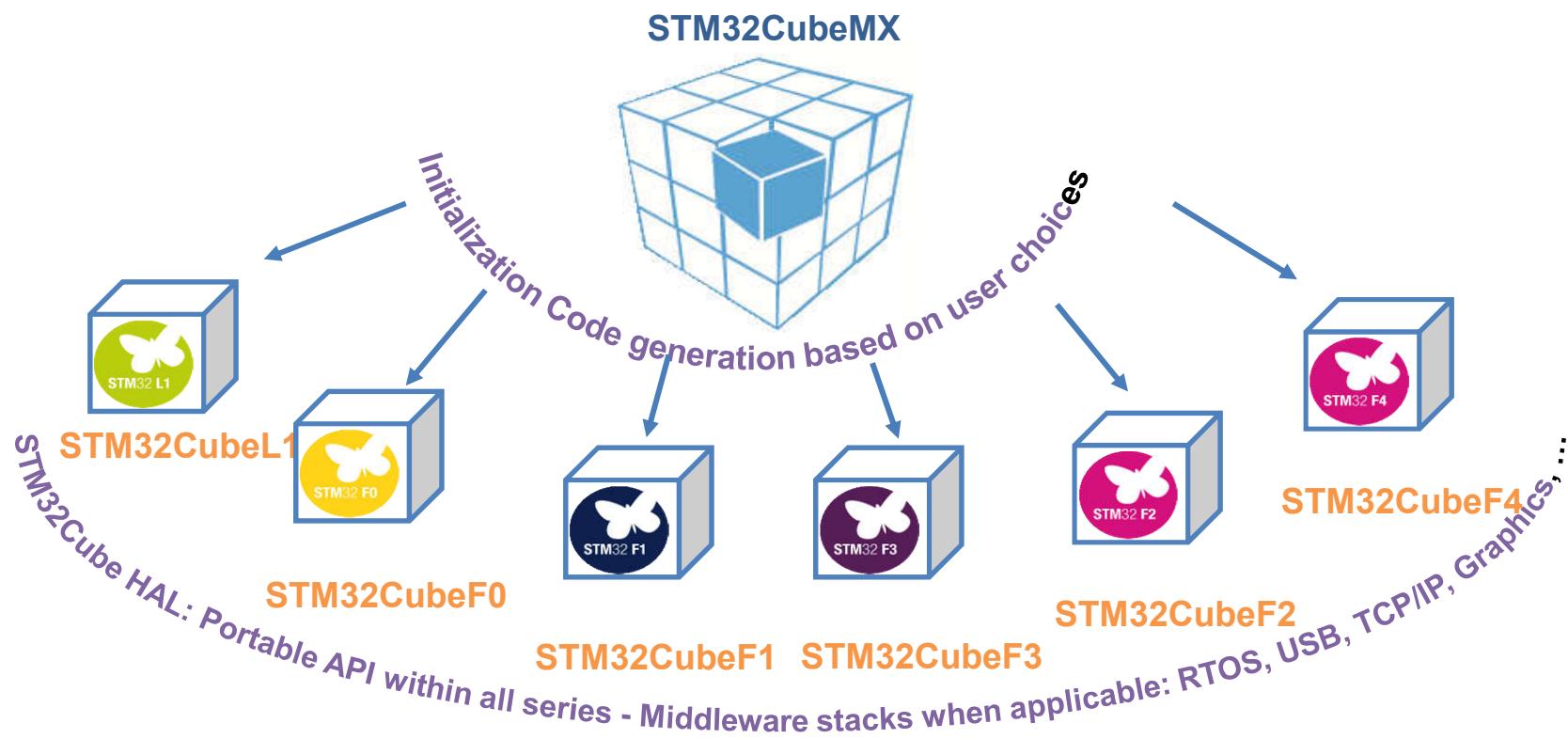
**7 outras fontes**

- EXTI line 16 connected to the PVD output
- EXTI line 17 connected to the RTC Alarm event
- EXTI line 18 connected to the USB OTG FS Wakeup event
- EXTI line 19 connected to the Ethernet Wakeup event
- EXTI line 20 connected to the USB OTG HS (configured in FS) Wakeup event
- EXTI line 21 connected to the RTC Tamper and TimeStamp events
- EXTI line 22 connected to the RTC Wakeup event

# STM32Cube

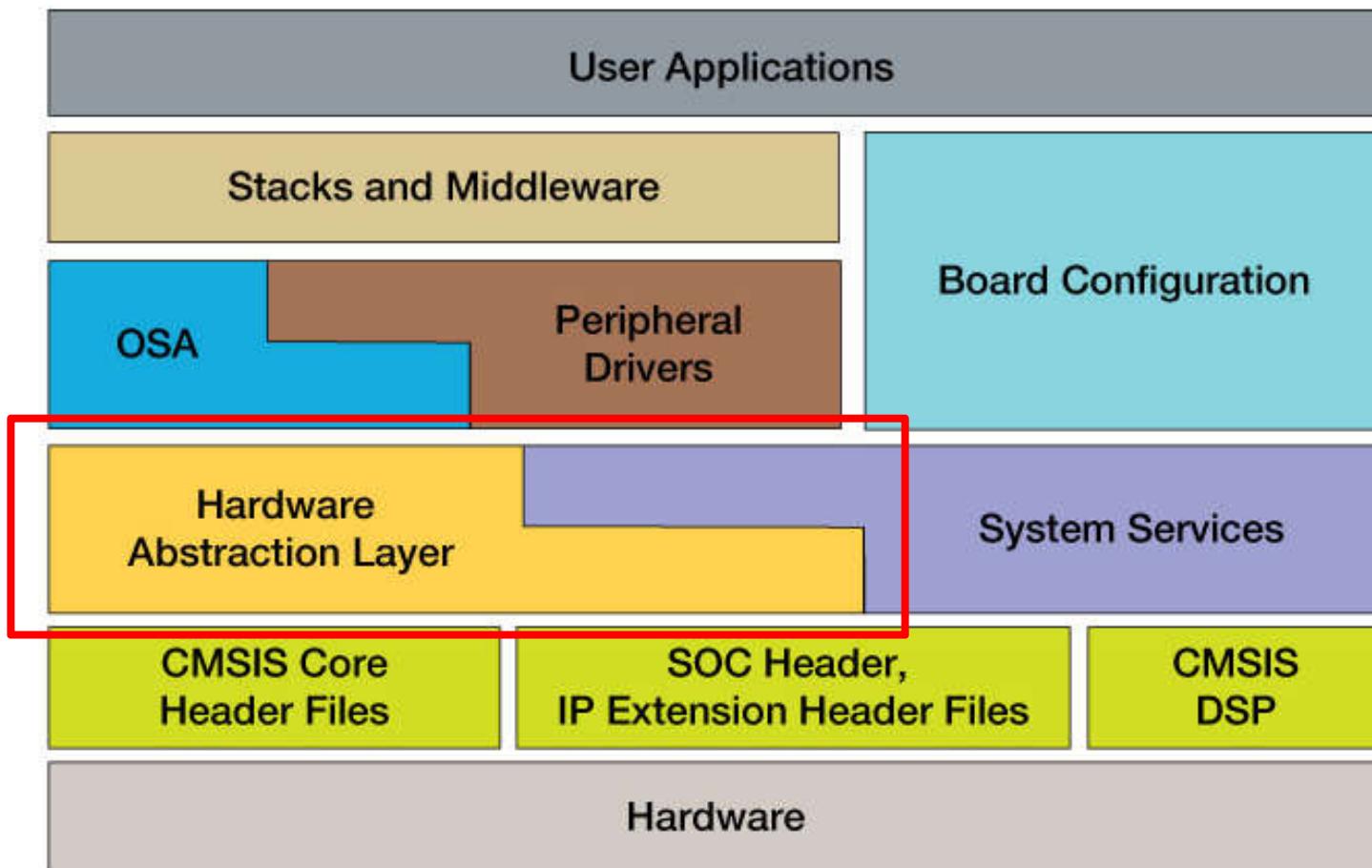
□ Compreende:

- STM32CubeMX (MX – MicroeXplorer): Ferramenta de configuração para gerar código de inicialização a partir das opções do usuário
- Bibliotecas com:
  - STM32 Hardware Abstraction Layer : STM32Cube HAL
  - Conjunto de Middlewares (abstração de detalhes do hardware voltado para gerenciamento e comunicação de dados): RTOS, USB, TCP/IP, Graphics, ...



## **Hardware Abstraction Layer (HAL) no STM32Cube**

- **User friendly APIs:** para abstração da complexidade de programação dos registradores, focando em sua funcionalidade
- **Portable APIs:** permite migrar programa do usuário entre diferentes famílias de microcontroladores.



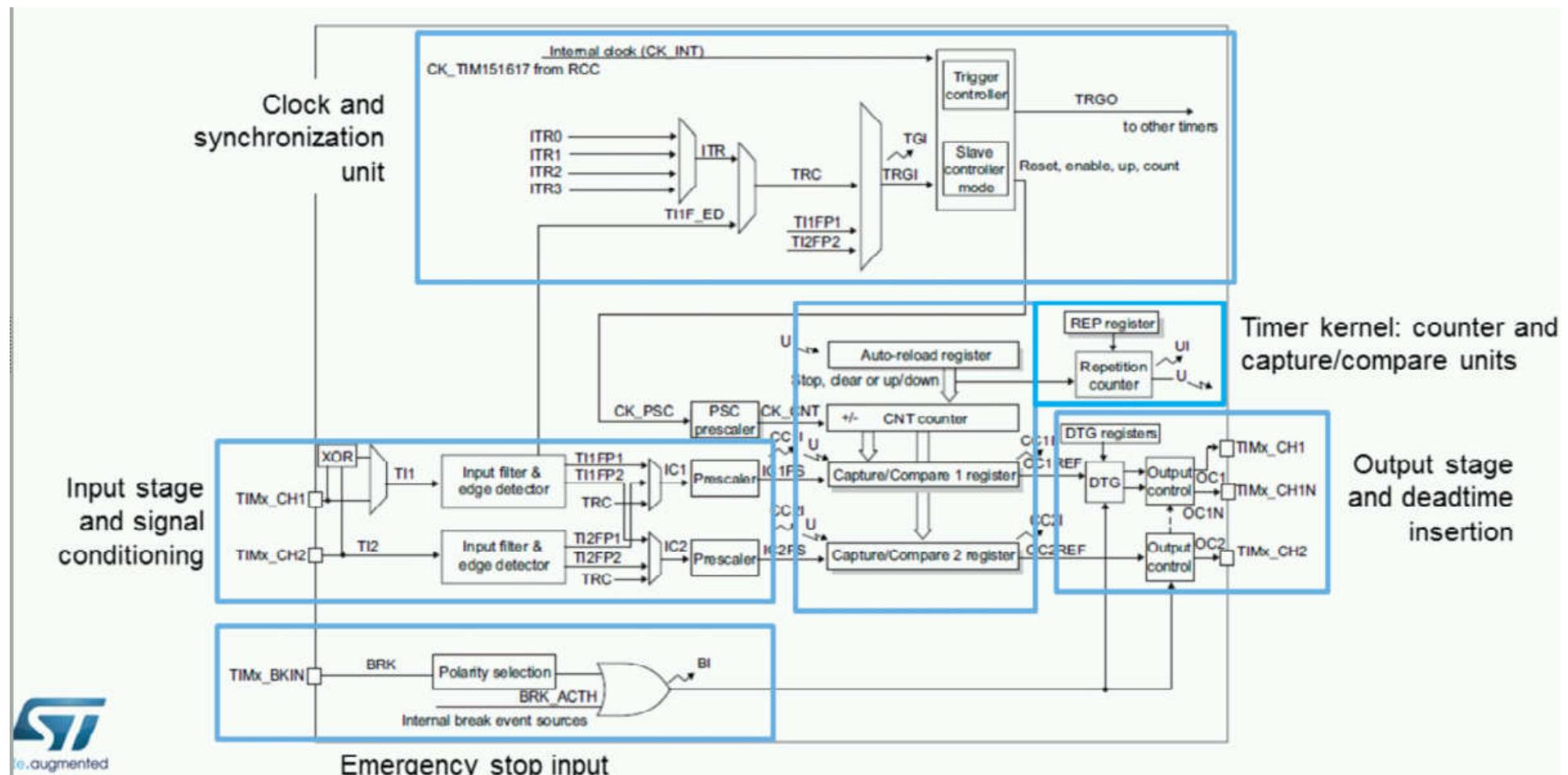
## ***Hardware Abstraction Layer (HAL) - Exemplo***

Exemplos:

- HAL\_GPIO\_Init()
- HAL\_GPIO\_DeInit()
- HAL\_GPIO\_ReadPin()
- HAL\_GPIO\_WritePin()
- HAL\_GPIO\_TogglePin()
- Quando pino é configurado para aceitar interrupção externa, a função “HAL\_GPIO\_EXTI\_Callback()” é criada para a inserção de código do usuário.

# STM32F4 Timers

ST Microelectronics AN4776 & AN4013 - Application Note



- Desabilita saídas PWM em caso de falha

## General-Purpose Timers (GPT) e Advanced Control Timers (ADCT)

15 Timers

8x GPT - 16bits (TIM3, 4, 9, ..., 14)

2x GPT - 32bits (TIM2,5)

2x ADCT - 16bits (TIM1,8)

2x para DAC - 16bits (TIM6,7)

1x system timer - 24bits  
(SysTick)

GPT (16-bits ou 32-bits): Todos com prescaler de 16 bits.

ADCT: Possuem características adicionais para controle de motores

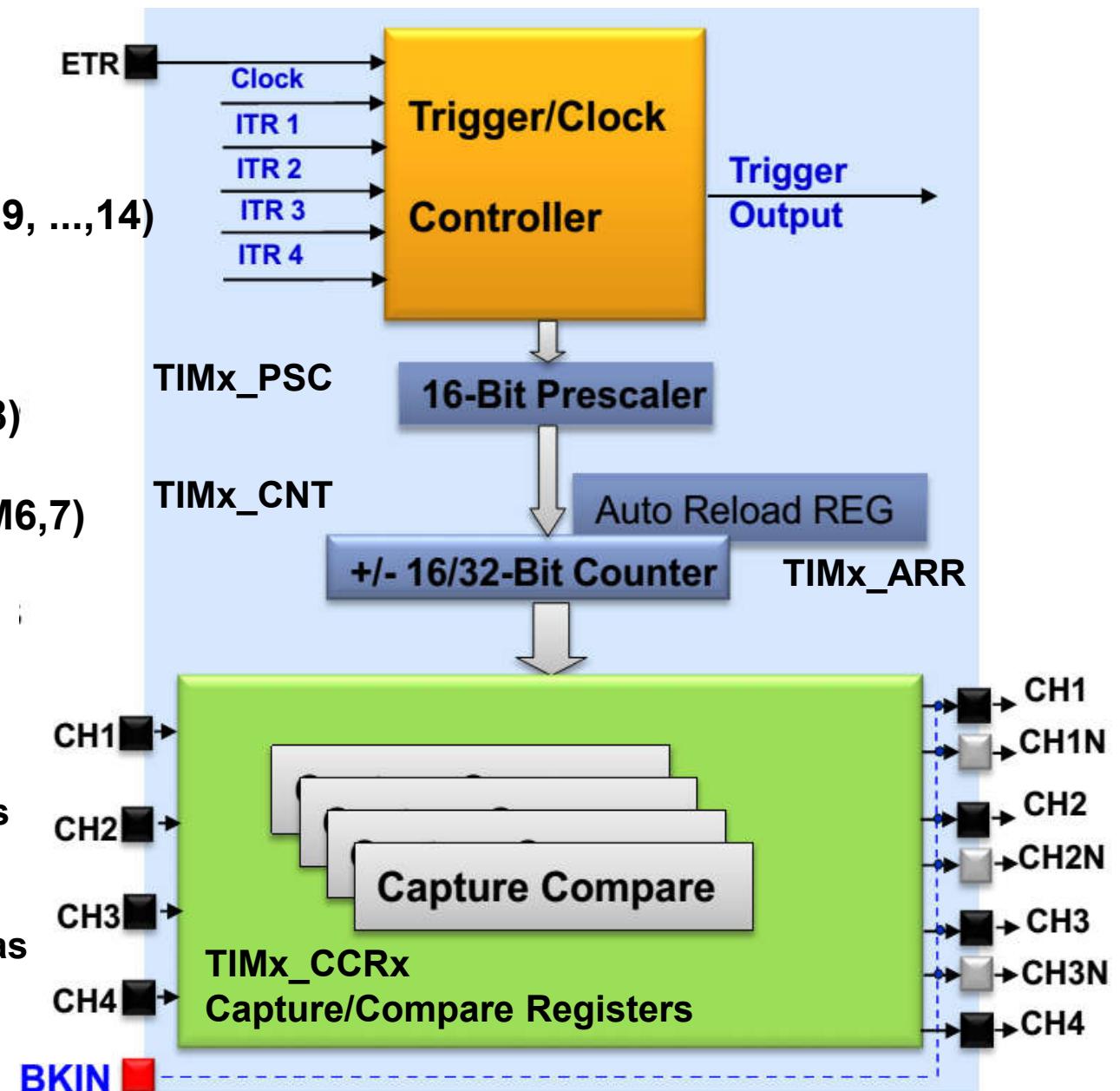


Table 6. Timer feature comparison

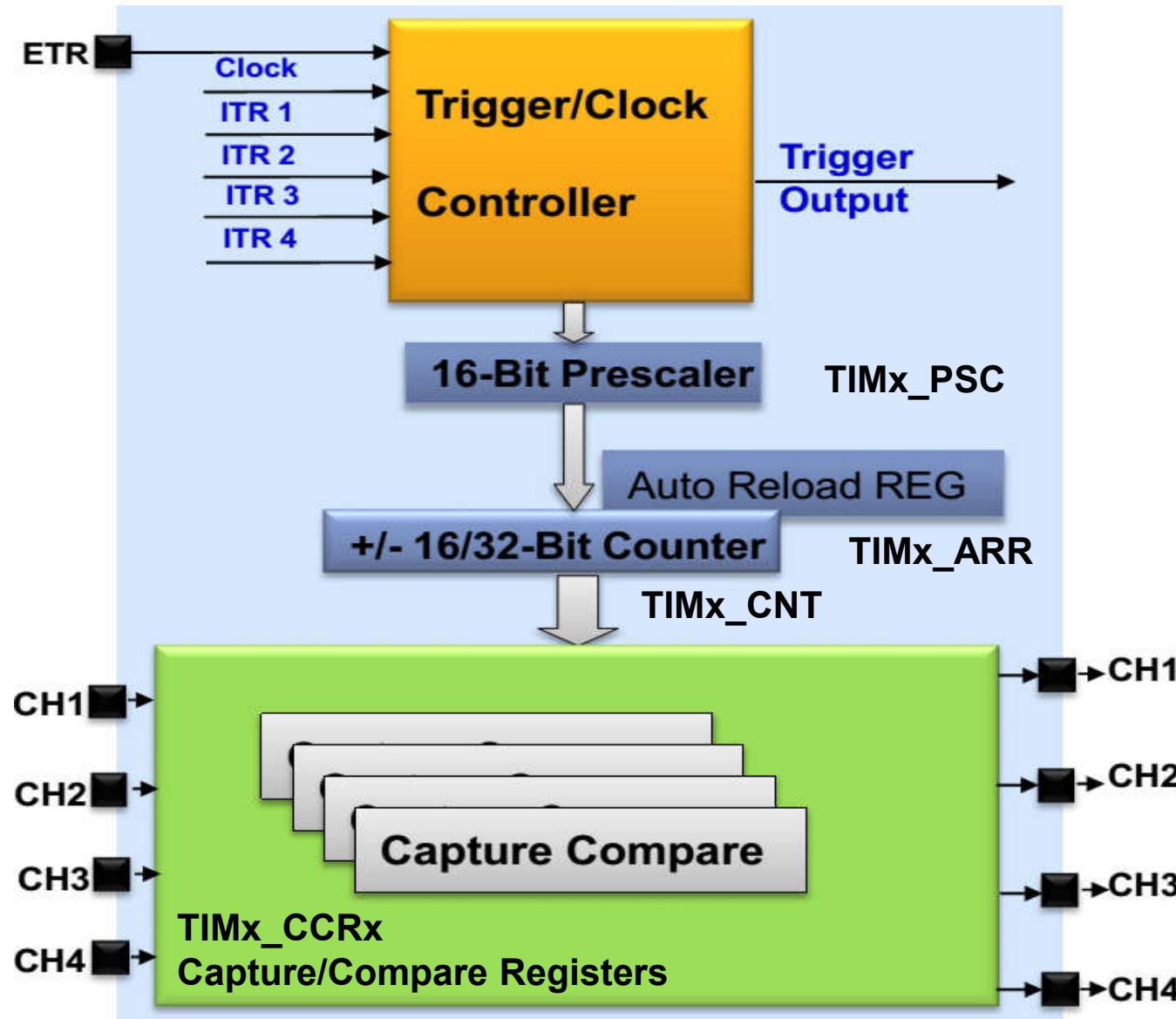
Timer type	Timer	Counter resolution	Counter type	Prescaler factor	DMA request generation	Capture/compare channels	Complementary output	Max interface clock (MHz)	Max timer clock (MHz) (1)
Advanced -control	TIM1, TIM8	16-bit	Up, Down, Up/down	Any integer between 1 and 65536	Yes	4	Yes	90	180
General purpose	TIM2, TIM5	32-bit	Up, Down, Up/down	Any integer between 1 and 65536	Yes	4	No	45	90/180
	TIM3, TIM4	16-bit	Up, Down, Up/down	Any integer between 1 and 65536	Yes	4	No	45	90/180
	TIM9	16-bit	Up	Any integer between 1 and 65536	No	2	No	90	180
	TIM10 , TIM11	16-bit	Up	Any integer between 1 and 65536	No	1	No	90	180
	TIM12	16-bit	Up	Any integer between 1 and 65536	No	2	No	45	90/180
	TIM13 , TIM14	16-bit	Up	Any integer between 1 and 65536	No	1	No	45	90/180
Basic	TIM6, TIM7	16-bit	Up	Any integer between 1 and 65536	Yes	0	No	45	90/180

1. The maximum timer clock is either 90 or 180 MHz depending on TIMPRE bit configuration in the RCC\_DCKCFGR register.

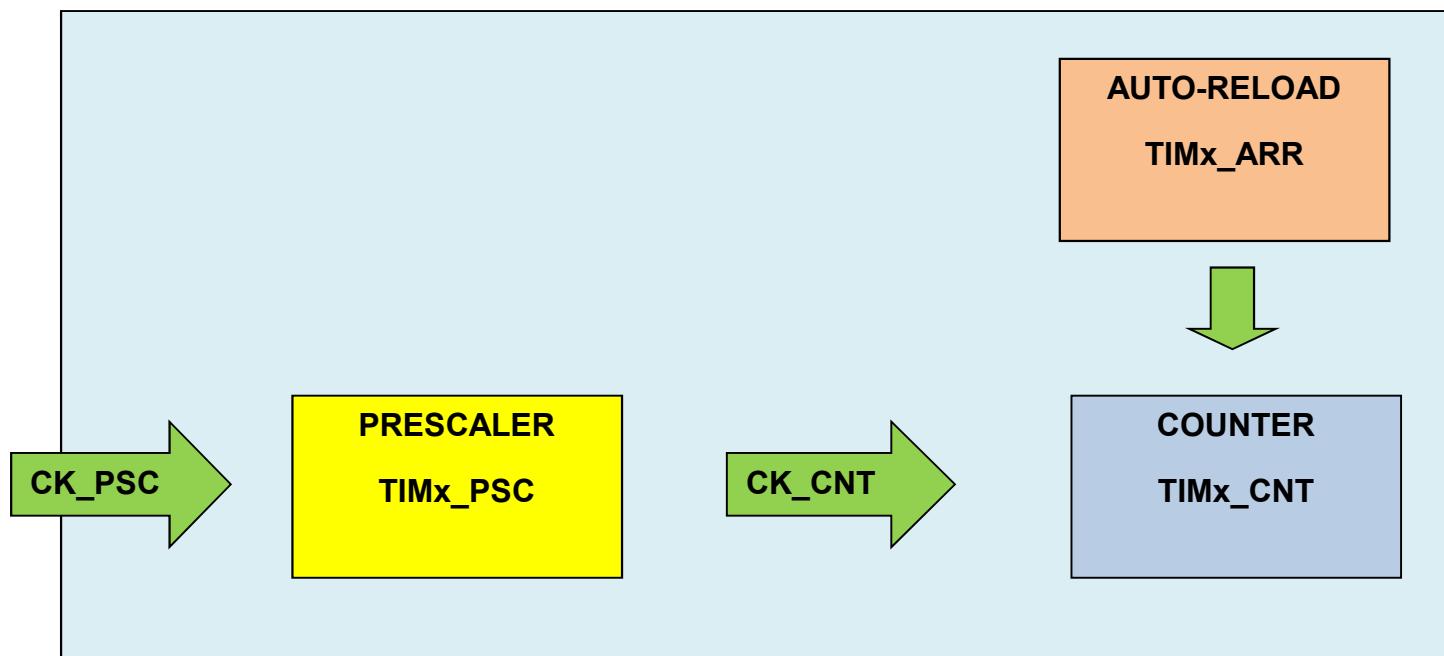
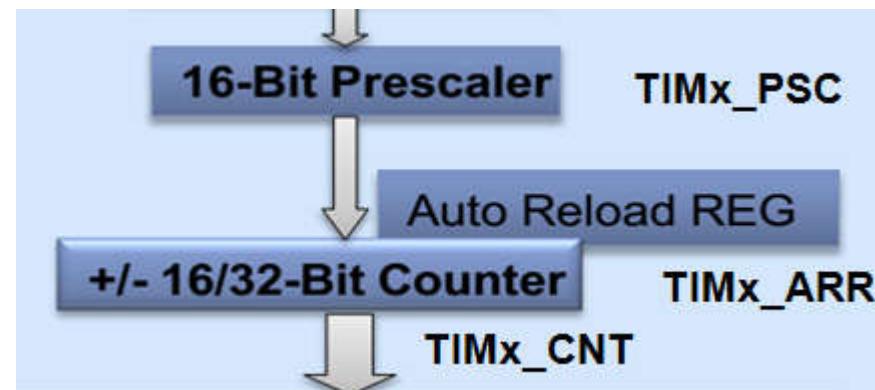
# Características dos TIMERS

## General-Purpose Timers

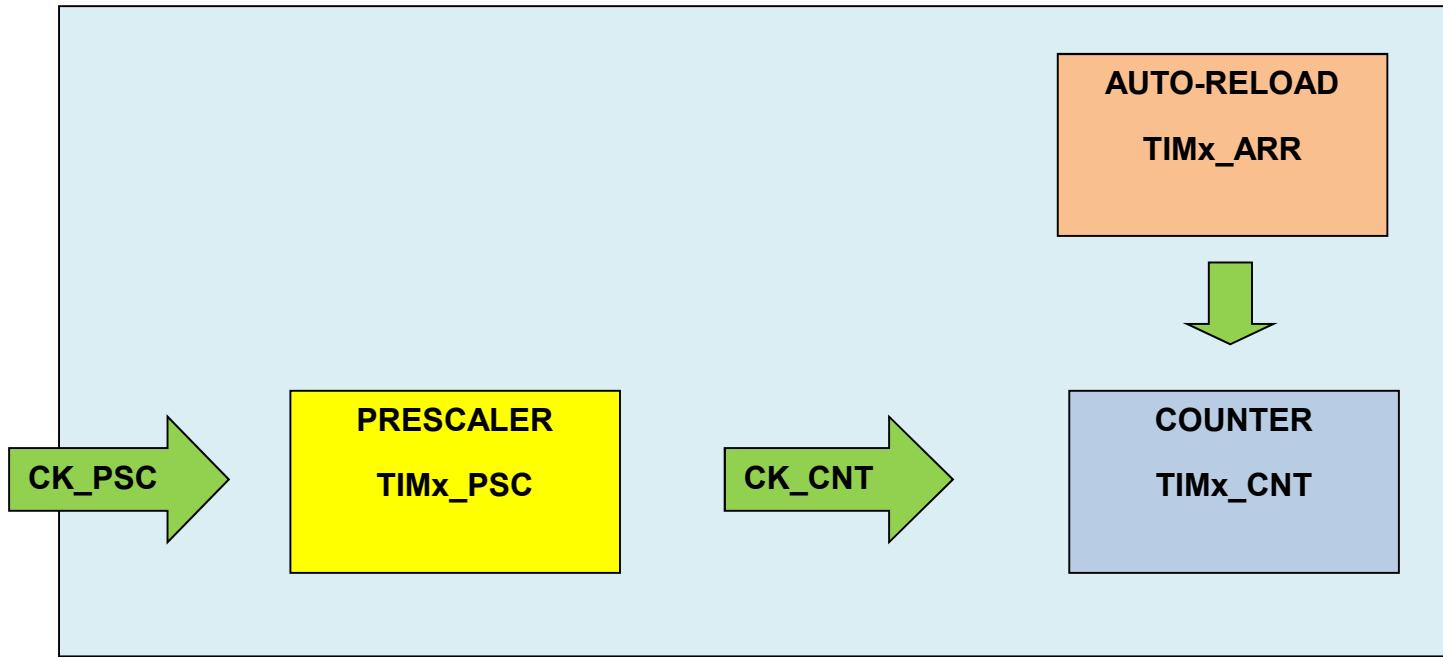
- ❑ Prescaler de 16-bits; Contador de 16-bits (**TIM3,4,9,..,14**) ou de 32-bits (**TIM2,5**): Auto-recarga;



# Timers: Counter Unit

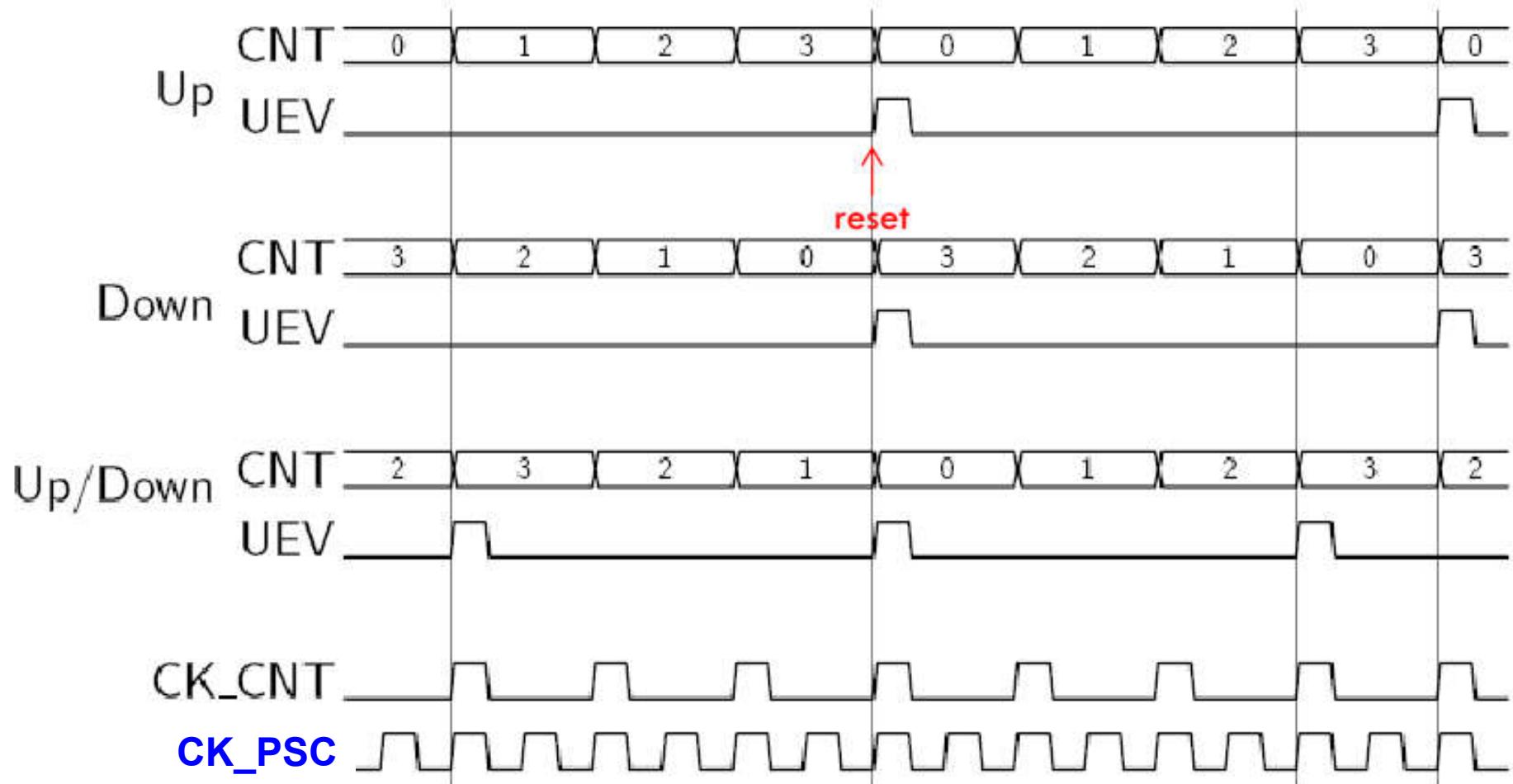


## General-Purpose Timers: Counter Unit



- ❑ A contagem pode ser por incremento, decremento e centrada (incrementa e decrementa).
- ❑ **Modo incremento**: parte de 0, incrementando até o valor do registrador de auto-recarga; reinicia de 0 e gera evento de *overflow* (flag UEV – *update event*).
- ❑ **Modo decremento**: parte do valor de auto-recarga, decrementando até 0; reinicia com valor do registrador de auto-recarga e gera evento de *underflow* (flag UEV – *update event*).
- ❑ **Modo centrado**: parte de 0, incrementando até o valor do registrador de auto-recarga , gerando evento de *overflow*. Passa então, a decrementar a partir do valor do registrador de auto-recarga até 0, gerando evento *underflow*. *Volta a contar a partir de 0*.

# Modos de Contagem



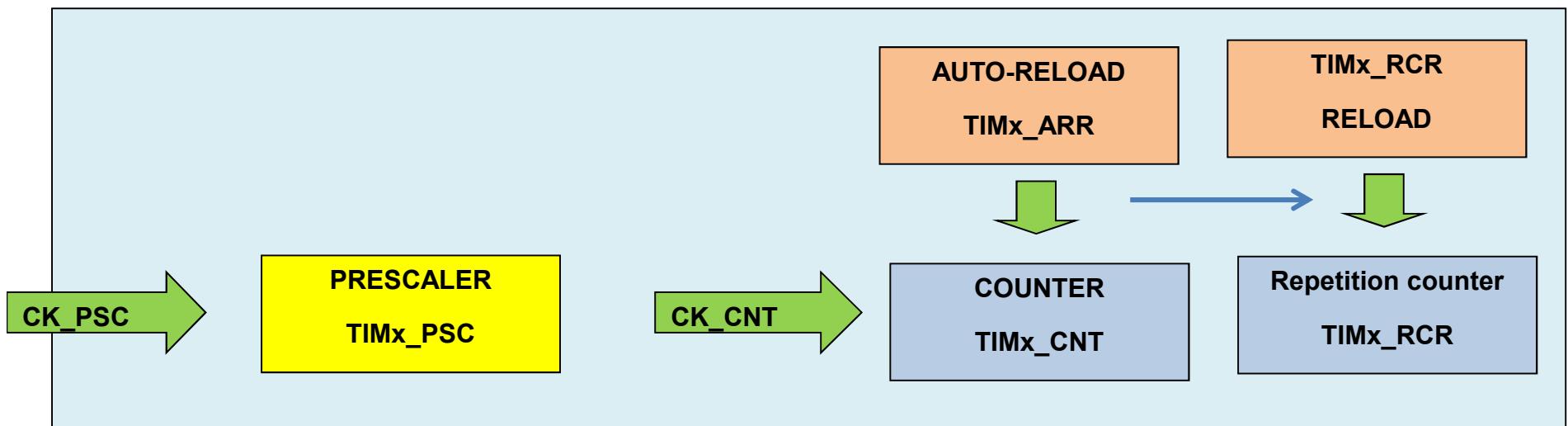
Counter Modes (ARR=3, PSC=1)

**CK\_PSC** conta de 0 a (**PSC + 1**);

GPT não possuem  
*repetition counter*

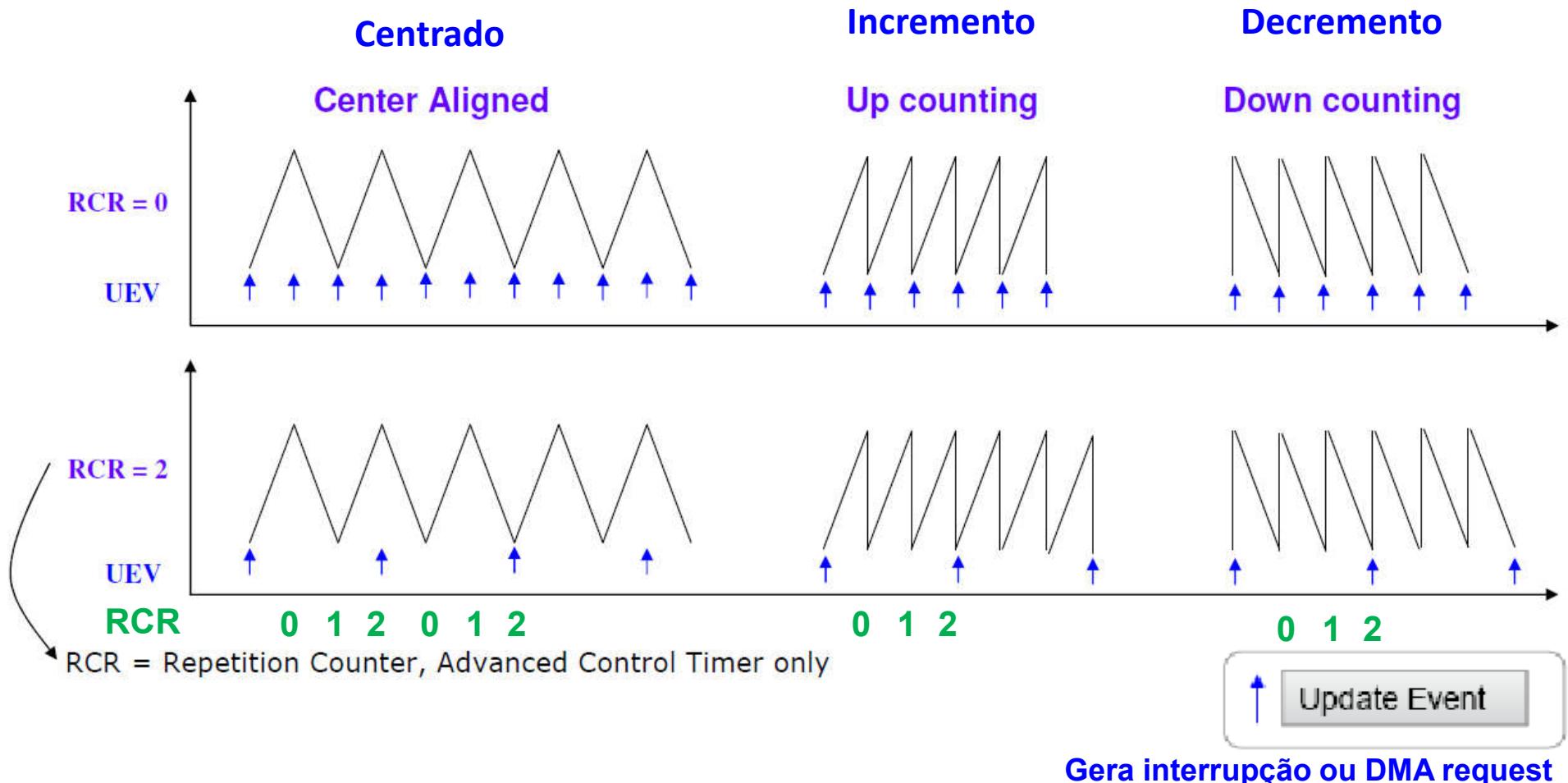
## Advanced Control Timers (TIM1 e TIM8): Counter Unit

- **Modo incremento:** parte de 0, incrementando até o valor do registrador de auto-recarga; reinicia de 0, gerando evento de *overflow* (flag UEV – *update event*).
- **Modo decremento:** parte do valor de auto-recarga, decrementado até 0; reinicia do valor do registrador de auto-recarga, gerando evento de *underflow* (flag UEV – *update event*).
- **Modo centrado:** parte de 0, incrementando até o valor do registrador de auto-recarga , gerando evento de *overflow*. Passa então, a decrementar a partir do valor do registrador de auto-recarga até 0, gerando evento *underflow*. *Volta a contar a partir de 0.*



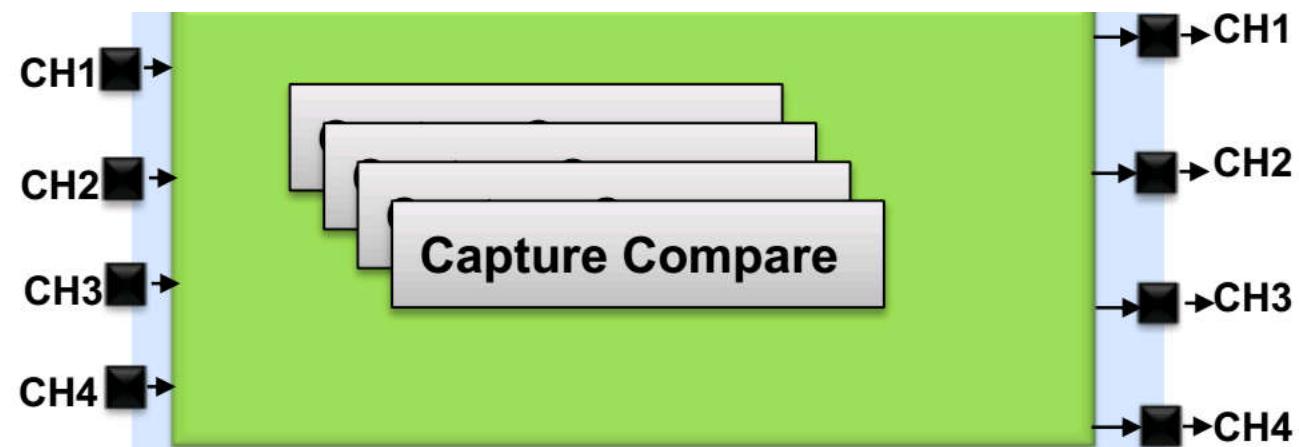
❖ Se o registrador de repetição for empregado (TIMx\_RCR), evento (UEV) é gerado após nro de contagens programado no registrador de repetição (TIMx\_RCR). Caso contrário, o evento é gerado a cada contagem.

# Modos de Contagem



- Update event (UEV)** é gerado após o número de contagens programado no registrador de repetição (TIMx\_RCR - Existe apenas em ADCTs).

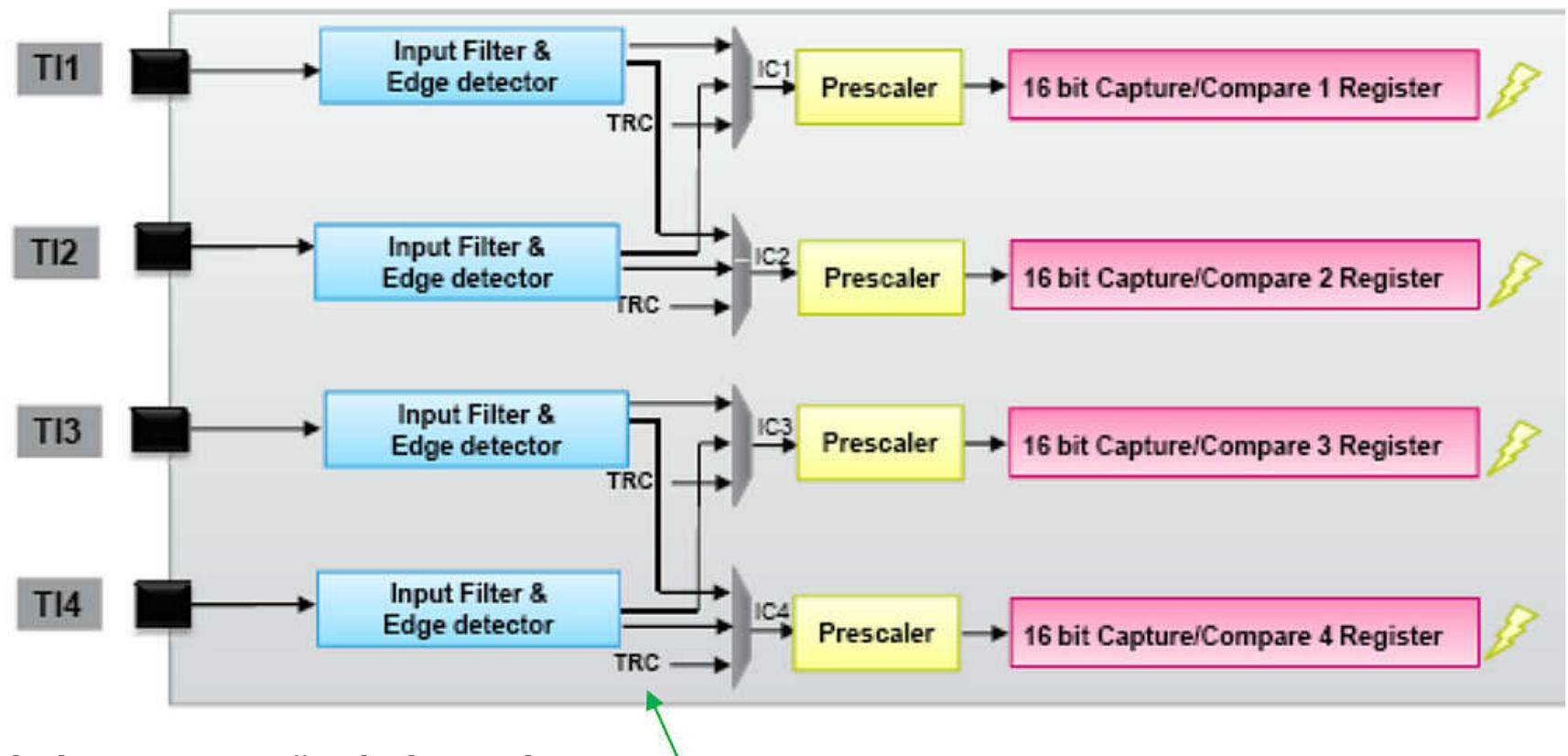
## *Timers: Capture/Compare Unit*



## General-Purpose Timers: Capture/Compare Unit

- Possuem até 4 canais independentes que operam nos seguintes modos:

- Captura de entrada (*Input Capture*)
- Comparação de saída (*Output Compare*)
- Saída de um pulso (*One Pulse*)
- PWM (*Pulse Width Modulation*)

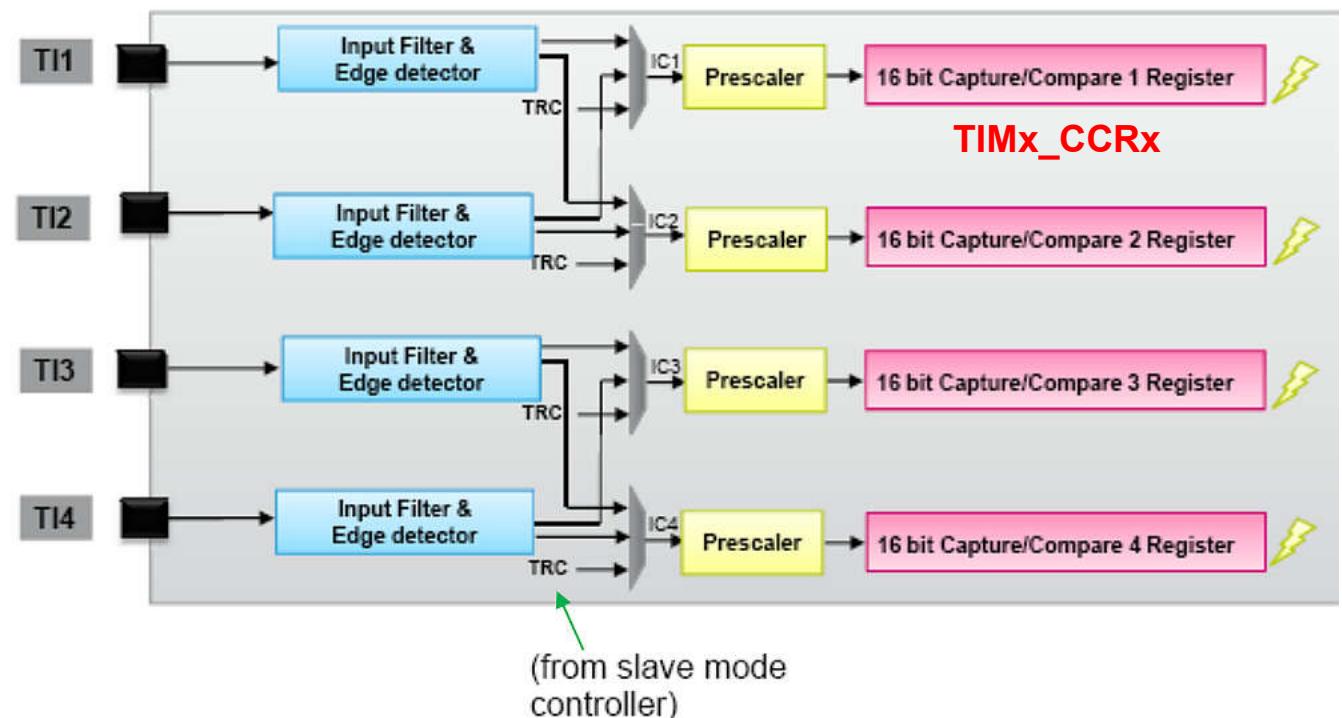


- Fitro digital para remoção de *bouncing* e ruído (from slave mode controller)

## Modo *Input Capture*

- Quando borda é detectada nos pinos TIx (subida ou descida):

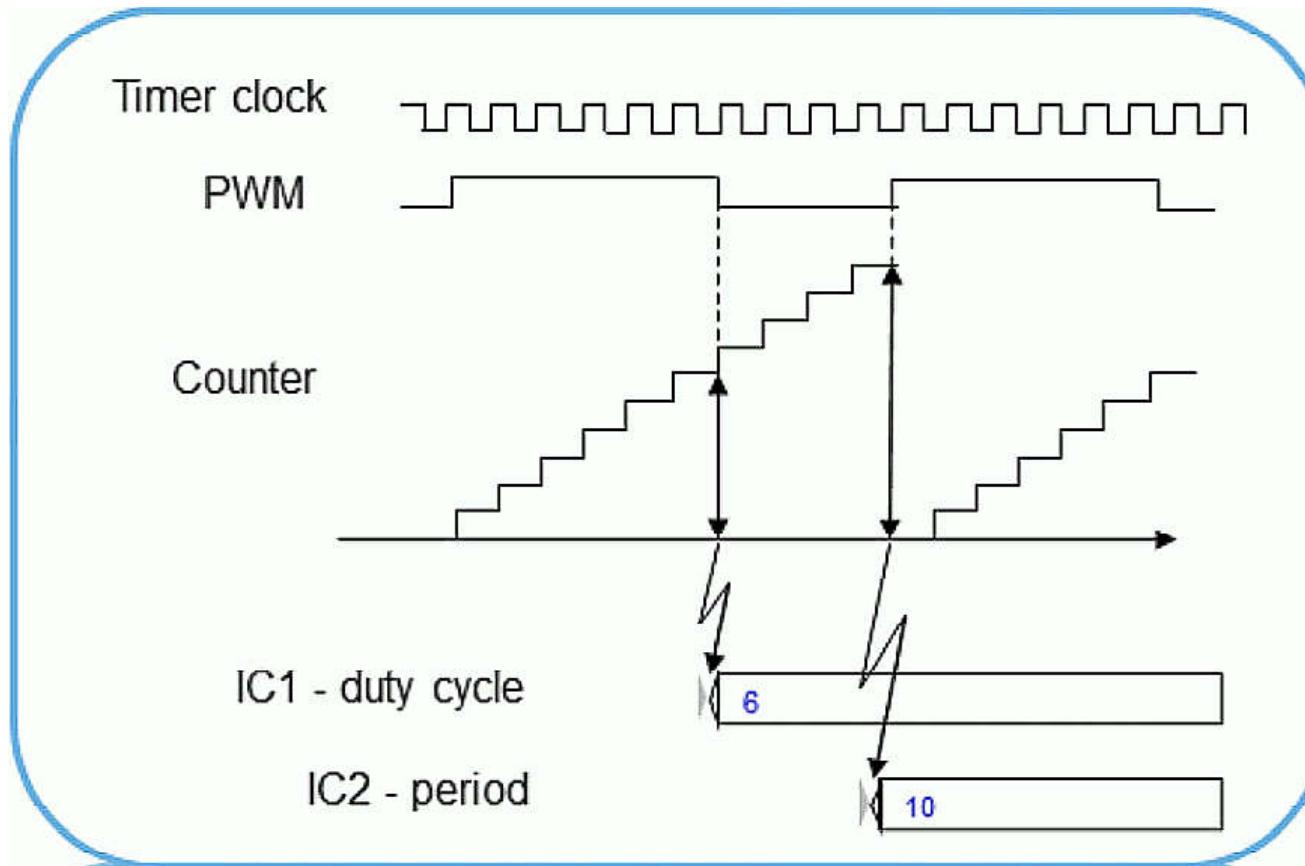
- Atual valor da contagem (TIMx\_CNT) é carregado no registrador de captura/comparação do canal correspondente (TIMx\_CCRx )
- Contador (TIMx\_CNT) pode ser resetado ou parado
- Interrupção ou transferência de dado pelo DMA pode ser requisitada
- “Overcapture” flag é setado se uma segunda captura ocorre sem que captura anterior seja processada (lida)



- Cada pino TIx pode controlar 2 registradores de captura
- TRC gerado por outro timer ou TI1 (ver slide à frente).

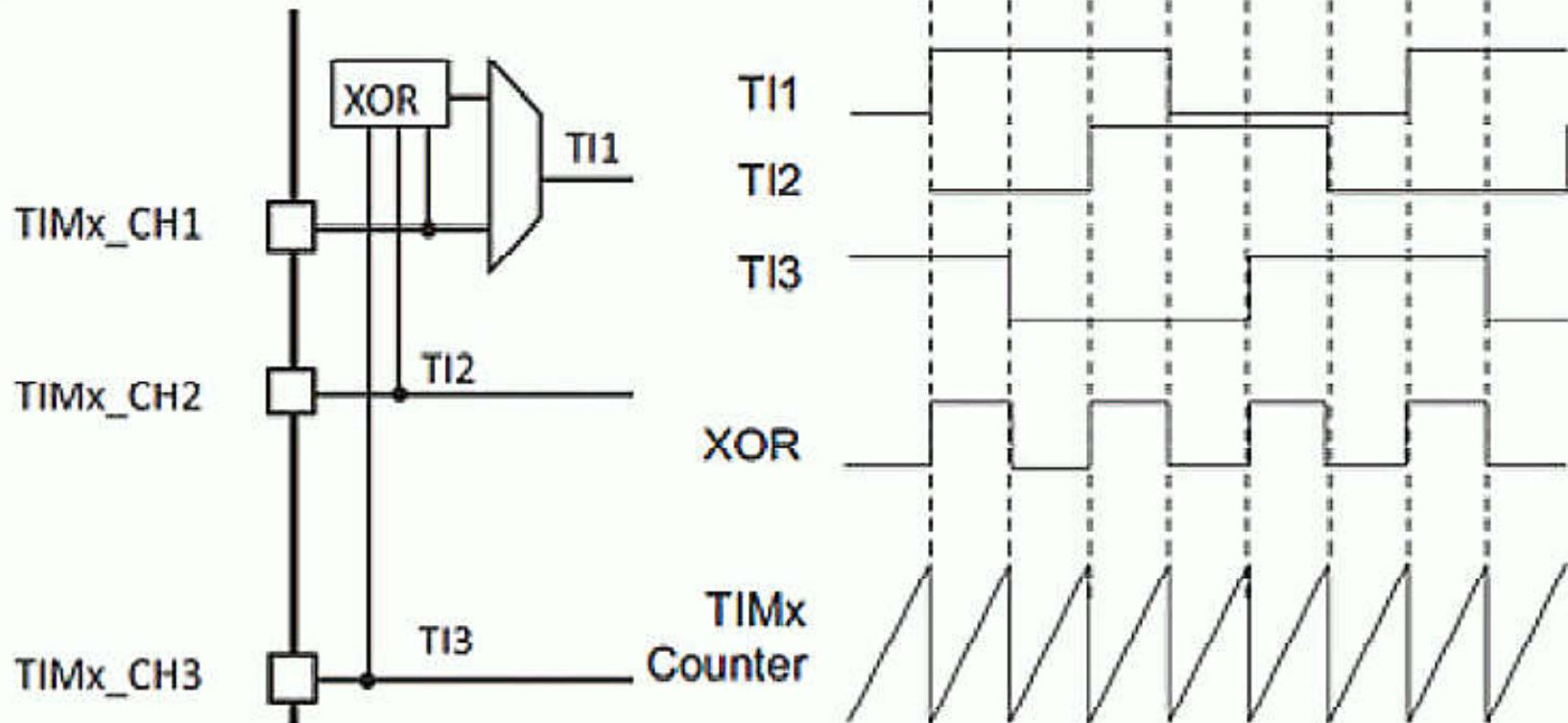
## Modo *Input Capture*

Medindo largura de pulso externo



- TI1 (ou TI2) solicita captura da contagem por IC1 em CCR1 (borda de descida) e por IC2 em CCR2 (borda de subida), resetando então, o contador CNT (*clear-on-capture*) => **Período = CCR2; Largura de pulso (duty cycle) = CCR1.**
- Canais 1 e 2 dos registradores Capture/Compare (CCRx) permitem reset do contador

## Combined Channels: XOR on



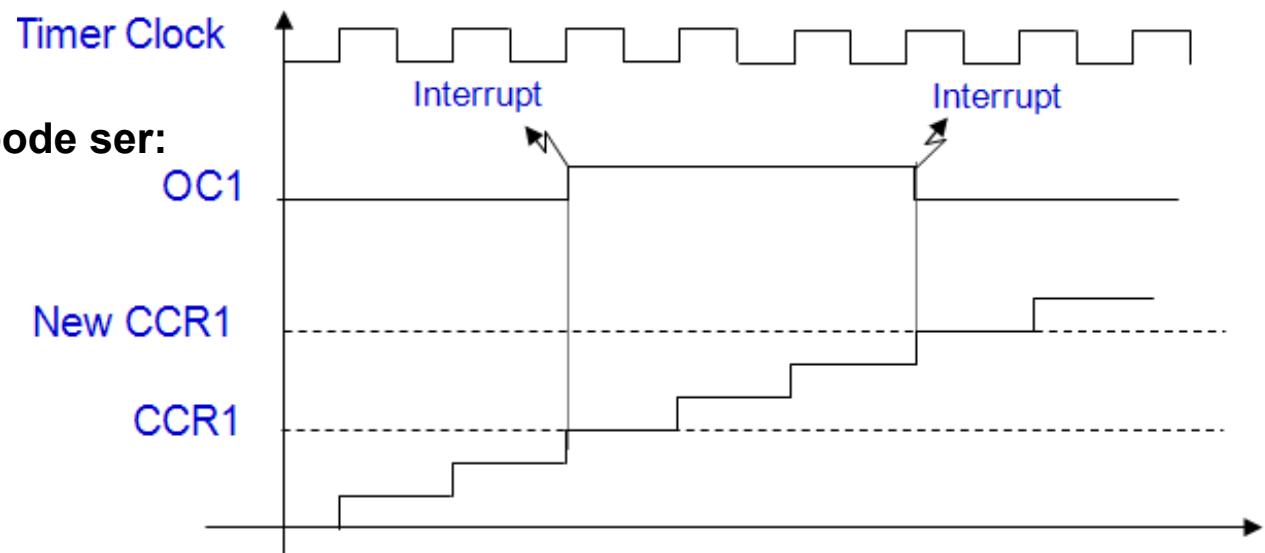
- **Clear-on-capture with XOR function** (borda de descida ou subida reseta a contagem): **Mede largura de pulso entre bordas de até 3 sinais**
- **OBS:** Utilizado com sensores magnéticos (HALL) para controle de velocidade de motores – 3 saídas defasadas de  $120^\circ$

## Modo *Output Compare*

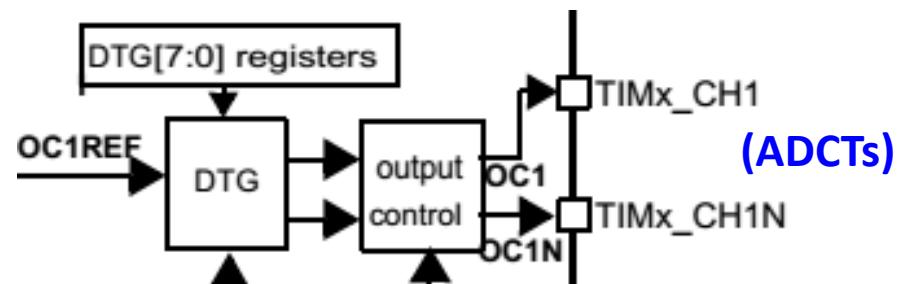
- Gera forma de onda em pino de saída ou indica término de intervalo
- Quando contagem iguala-se a valor contido em registrador *capture/compare* (TIMx\_CCRx):

- Pino de saída do canal pode ser:

- Setado
- Resetado
- Complementado
- Inalterado

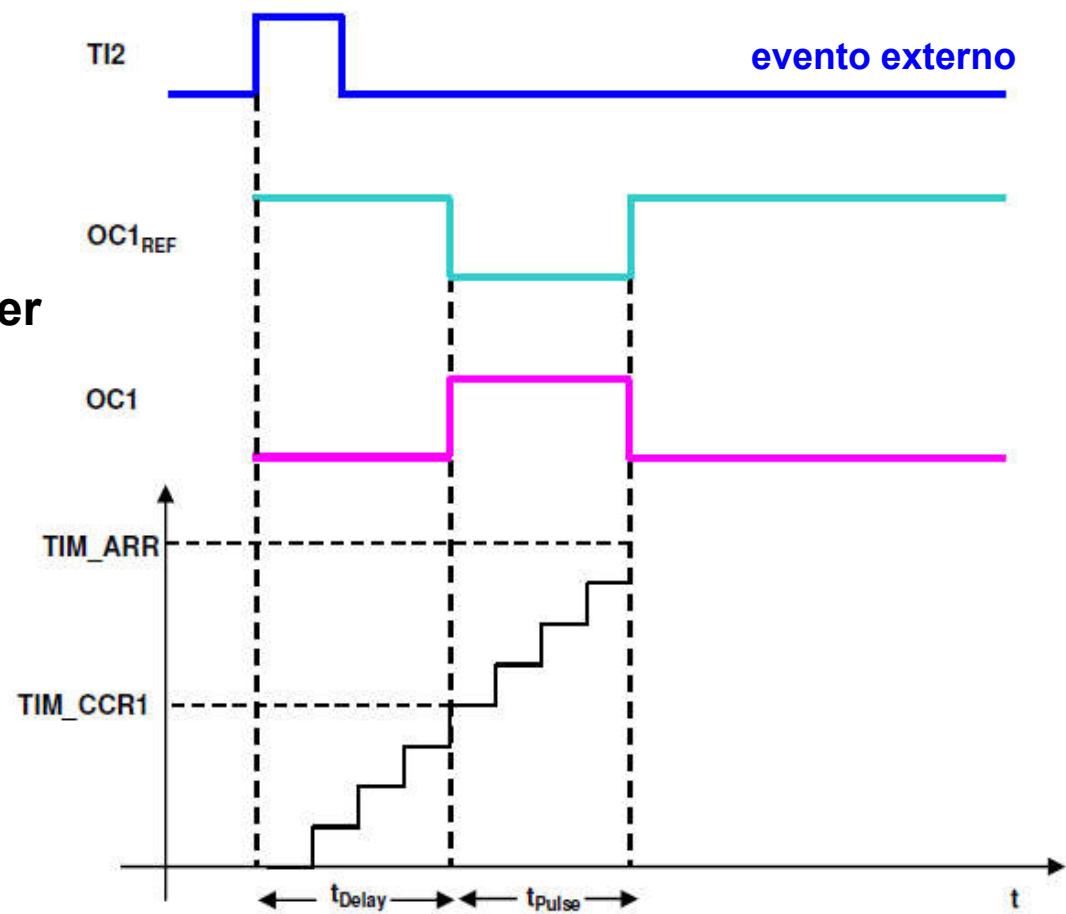
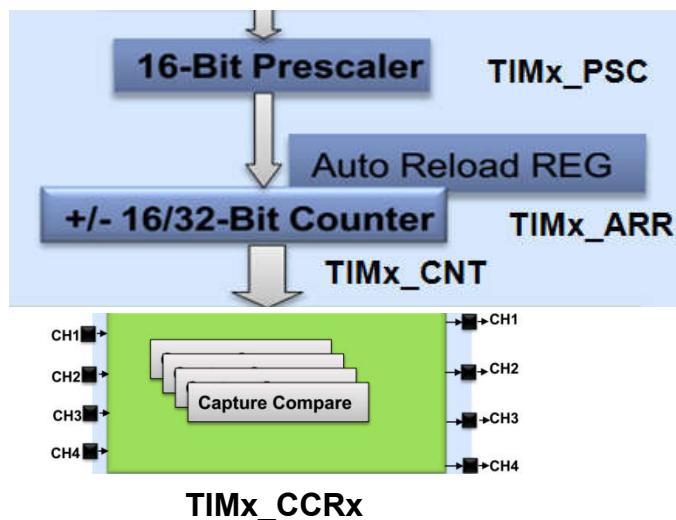


- Flag do *Interrupt Status Register* é setado
- Interrupção gerada (se habilitada)
- Requisição de DMA (se programado)
- Os registradores TIMx\_CCRx podem ser programados para aceitar ou não recarga.



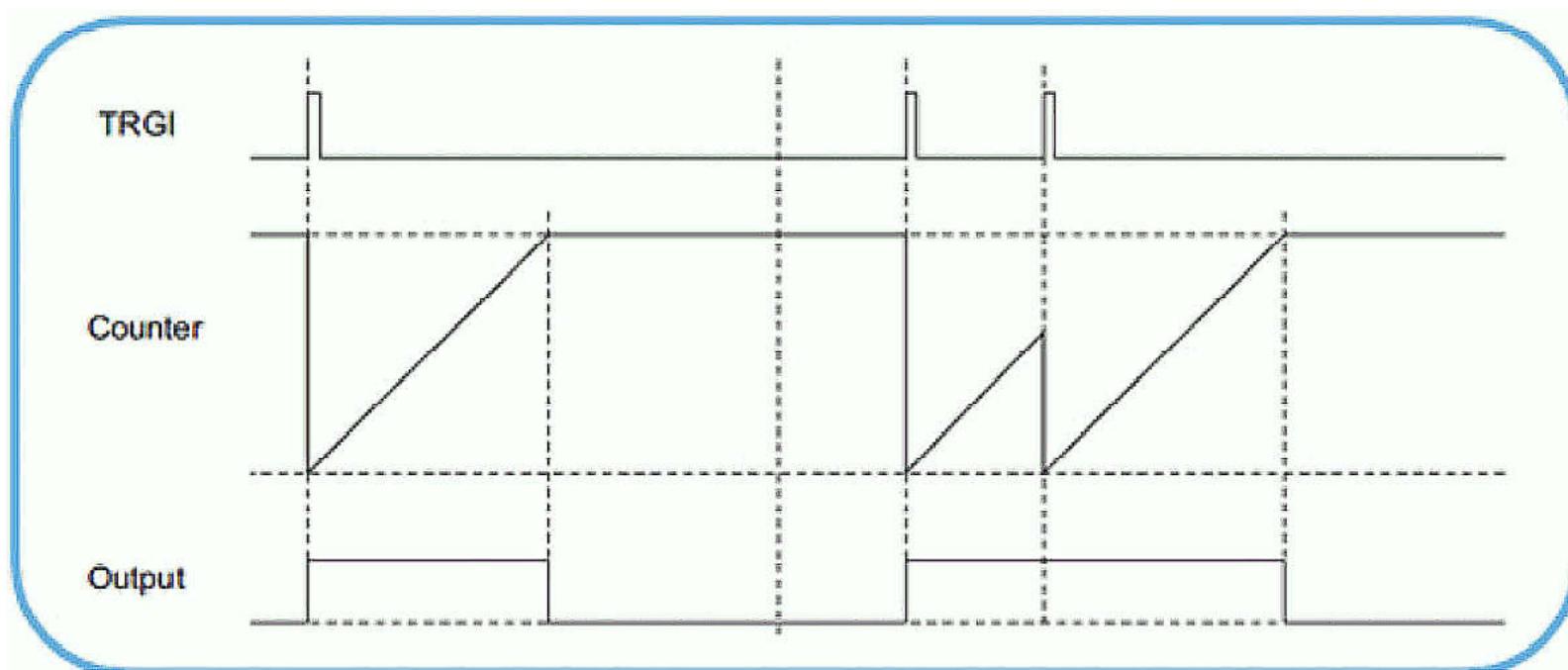
## Modo One Pulse

- Reúne os modos *Output Compare* e *Input Capture*
- Gera forma de onda a partir de solicitação por evento externo (pulso em pino)
- Contagem inicia-se a partir de evento e gera pulso com duração (**TIMy\_ARR** - **TIMy\_CCRx**) programável após atraso (programado em **TIMy\_CCRx**)
- Há 2 opções de geração de Modo One Pulse Mode:
  - Pulso único
  - Trem de pulsos após trigger



## Modo One Pulse

- ☐ **Retriggerable:** Novo trigger antes de fim de contagem, reseta contador e extende pulso



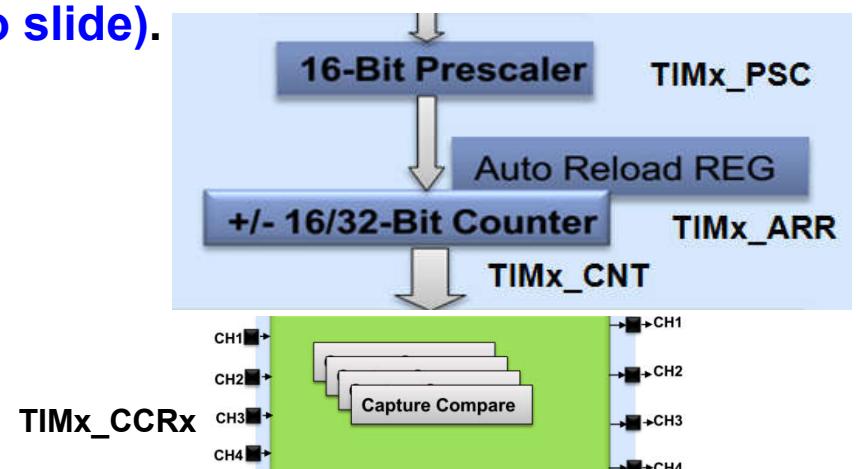
OBS: Disponível para STM32F30x and STM32F3x8

## Modo PWM (*Pulse Width Modulation*)

□ Permite gerar:

- 4 sinais independentes (GPT/ADCT) e 3 complementares com inserção individual de *dead time* (ADCT).
- O período e o *duty cycle* são definidos, respectivamente, por:
  - TIMx\_ARR (registraror de auto-recarga);
  - TIMx\_CCRx (cada canal PWM define seu *duty cycle*)

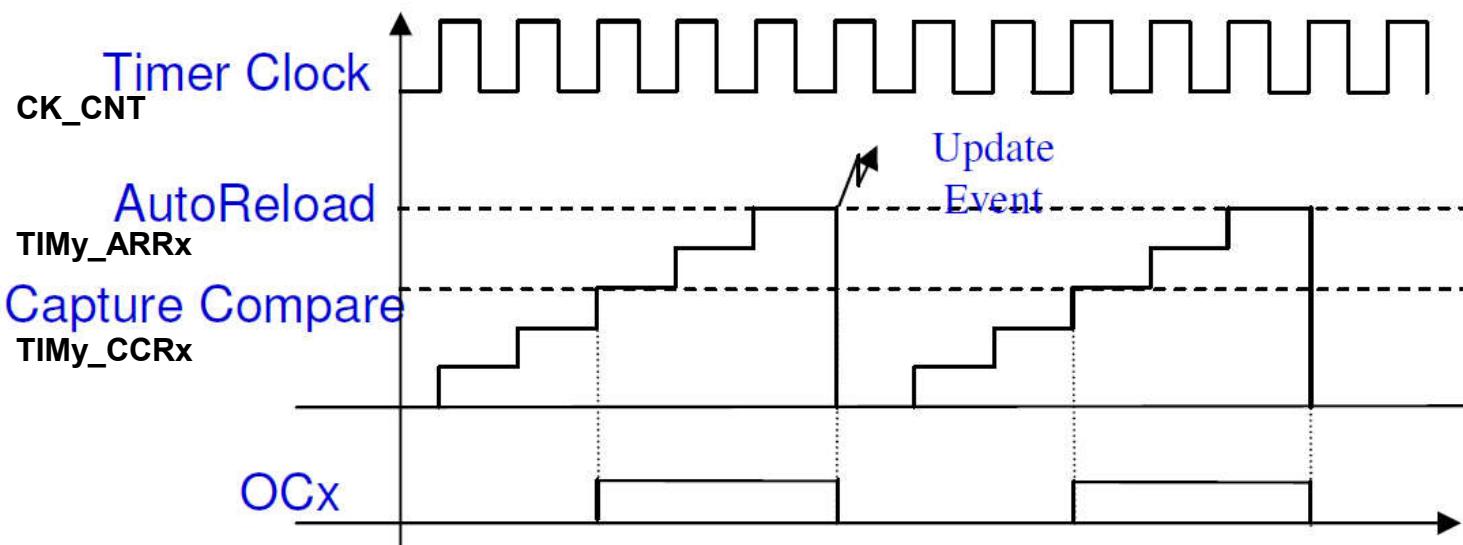
**Exemplo:** Gerar PWM de 45kHz com duty cycle de 50% no TIM1 (clock de 180MHz)  
=> Carregar TIM\_PSC com 1 (TIM1\_CLK/(1+1)), TIM1\_ARR com 1999 e TIM1\_CCRx com 999 (OBS: 90M/2000 = 45k. Ver próximo slide).



□ Há duas configurações para este modo:

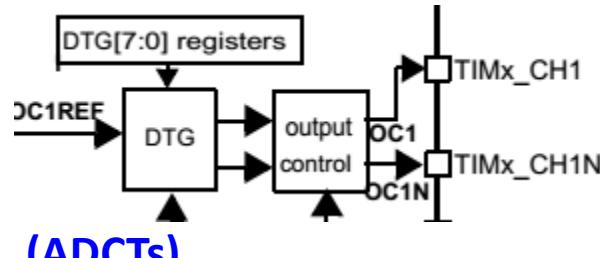
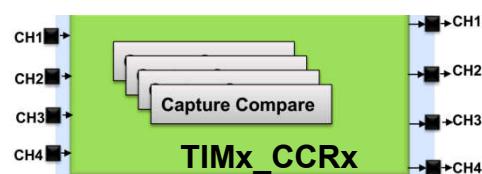
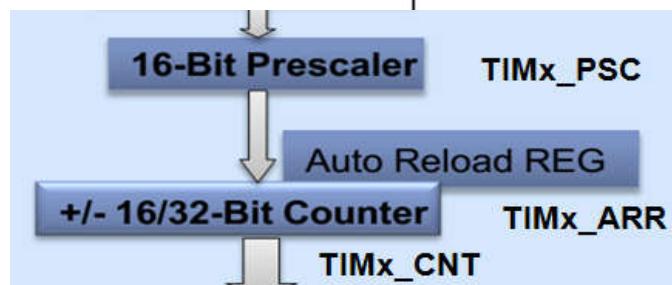
- *Edge-aligned* (contagem *up* ou *down*)
- *Center-aligned*

# Edge-aligned Mode

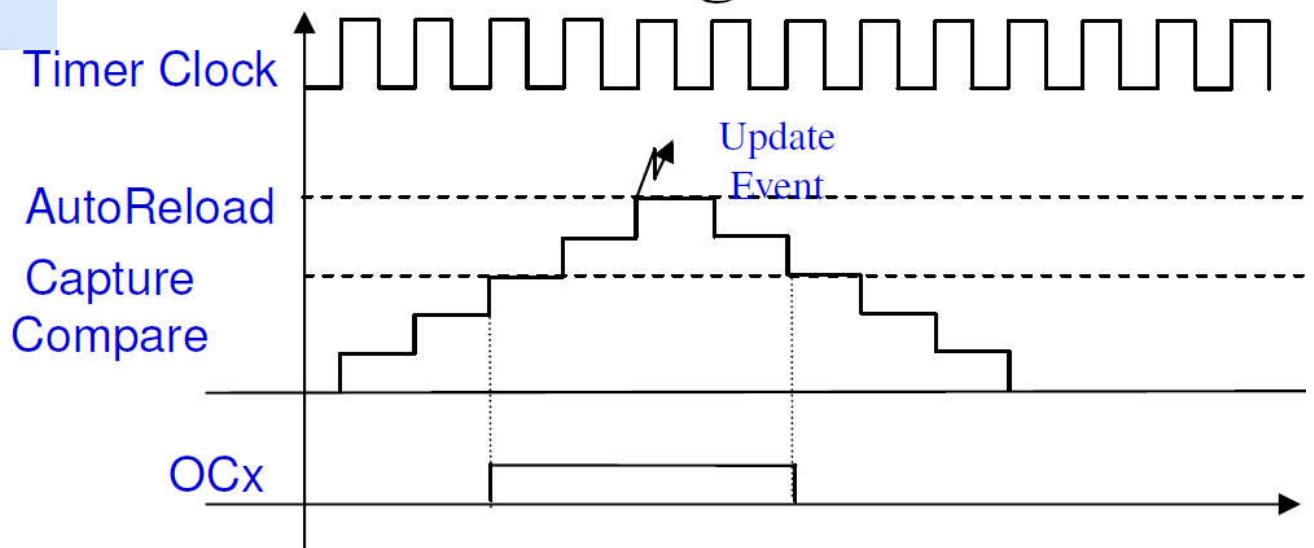


PWM mode 2

(Em PWM  
mode 1, pino  
de saída  
invertido)



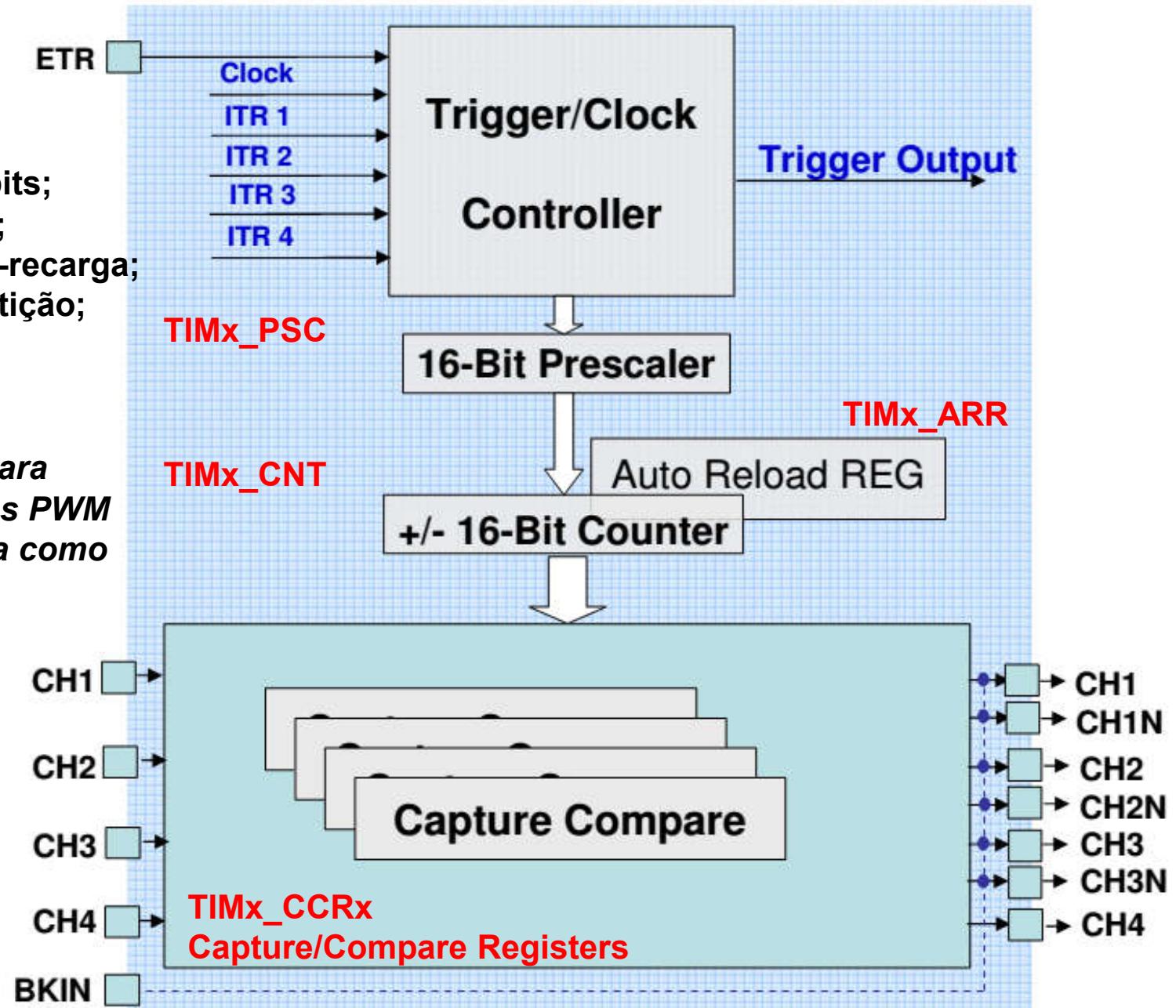
# Center-aligned Mode



## Advanced Control Timers (TIM1 e TIM8): Capture/Compare Unit

- ☐ Prescaler de 16-bits;  
Contador de 16-bits;  
Registrador de auto-recarga;  
Registrador de repetição;  
CLK=180MHz

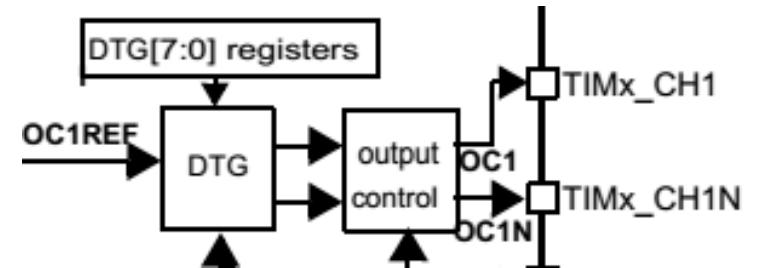
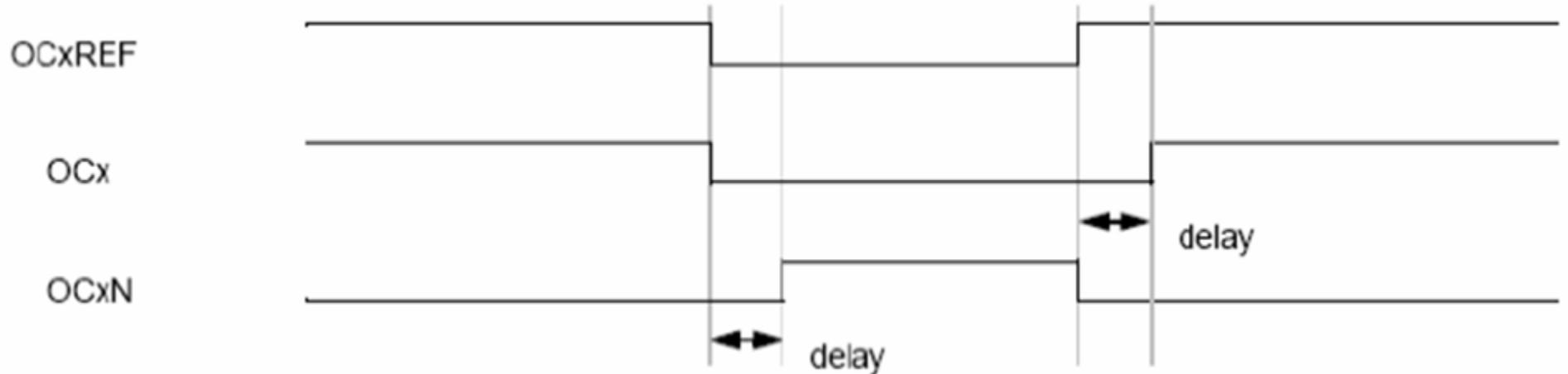
- ☐ BKIN: Utilizado para desabilitar as saídas PWM de forma assíncrona como proteção em dada aplicação



## **Advanced Control Timers (TIM1 e TIM8): Capture/Compare Unit**

- Assim como o GPT, possuem 4 canais independentes que operam nos seguintes modos:
  - Captura de entrada (*Input Capture*)
  - Comparação de saída (*Output Compare*)
  - PWM (*Pulse Width Modulation*)
  - Geração de único pulso (*One pulse*)
- Saídas complementares com *dead-time* programável

## ADCTs: Saídas complementares com *dead-time* programável

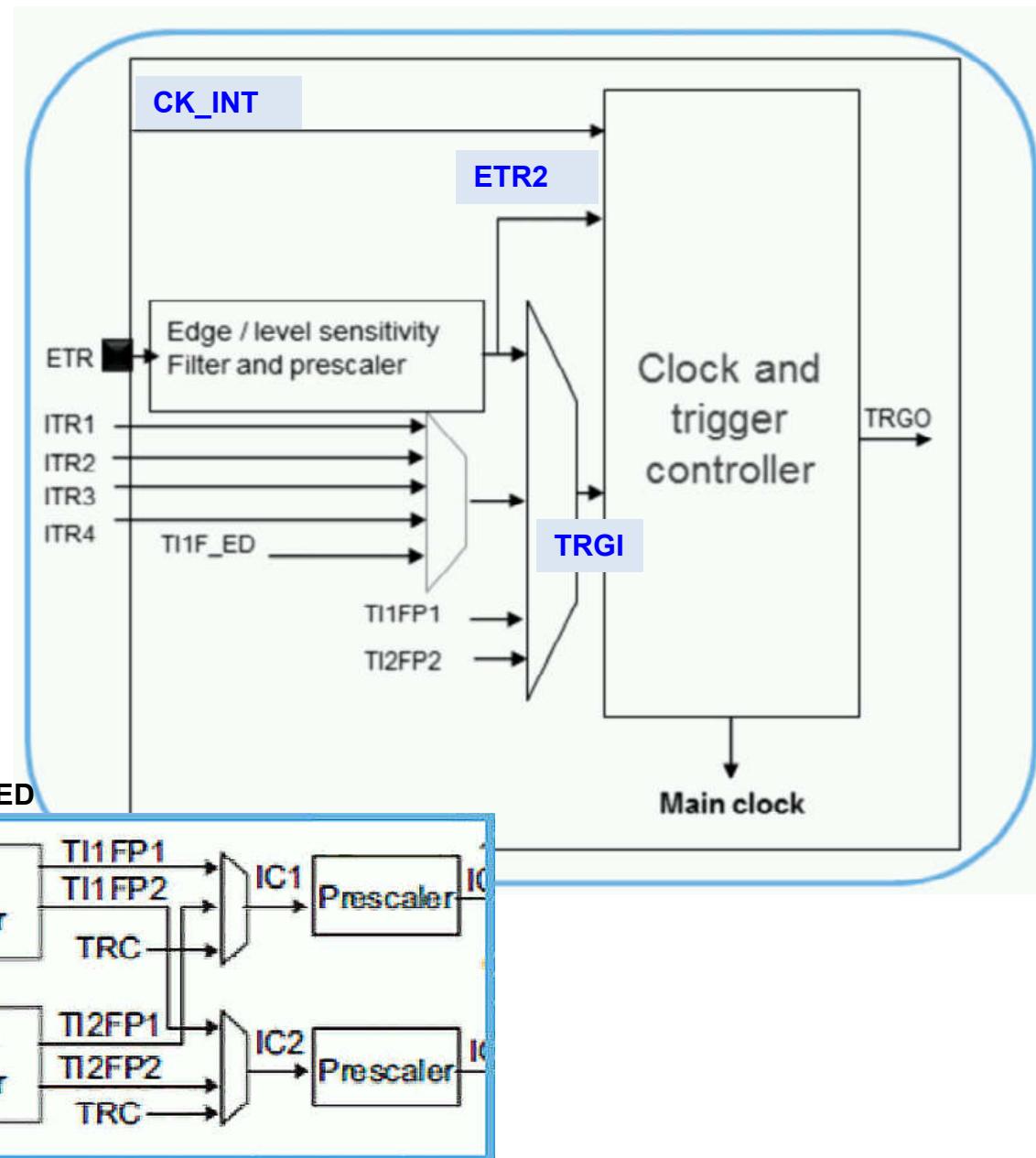


- ❑ 2 saídas independentes em 3 canais
- ❑ Usa mesma forma de onda de referência para gerar 2 saídas em 3 canais: OCx e OCx\_N
- ❑ Inserção de *Dead Time* (registrar TIMx\_BDTR)
  - Bordas de subida de OCx e OCx\_N atrasadas por *dead time* programável

# Timers: Clock and Synchronization Unit

❑ Refs:

- ❑ AN4013
- ❑ AN4776



□ Internal clock (CK\_INT) – clock gerado através do RCC (APB1 ou APB2) (1)

□ External clock mode 1 (TRGI):

▪ ETR1 (1)

▪ 4 Internal Trigger clocks (ITRx): utiliza clock suprido por outro timer (4)

▪ clock aplicado a pinos externos dos Capture/Compare: TI1 ou TI2

Pino TI1: TI1FP1 ou TI1F\_ED (1)

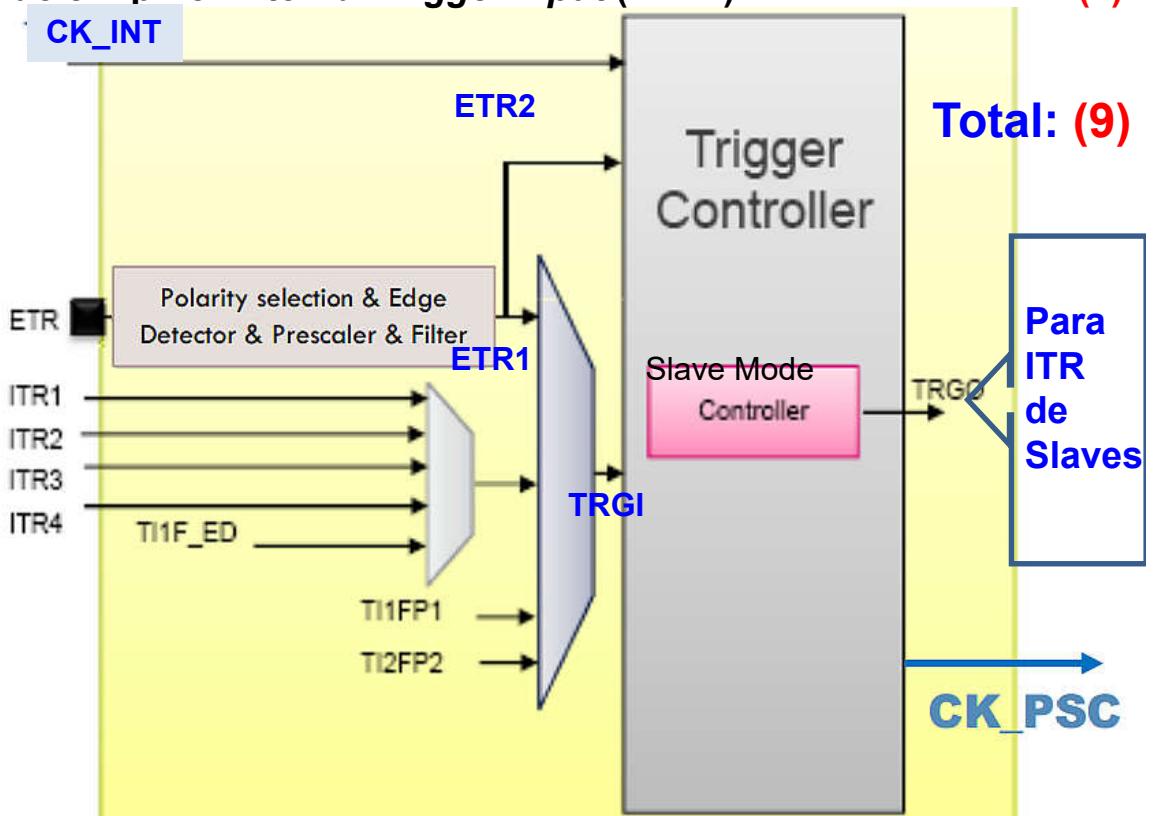
Pino TI2: TI2FP2 (1)

□ External clock mode 2: clock suprido em pino External Trigger Input (ETR2) (1)

▪ ITRx e clocks externos tem acesso ao contador por porta de entrada controlada por trigger externo: pino ETR

**Prescaler:** Divisão por 1,2,4,8

**Filter:** contador em que N eventos (programável) são necessários para identificar mudança de nível do sinal



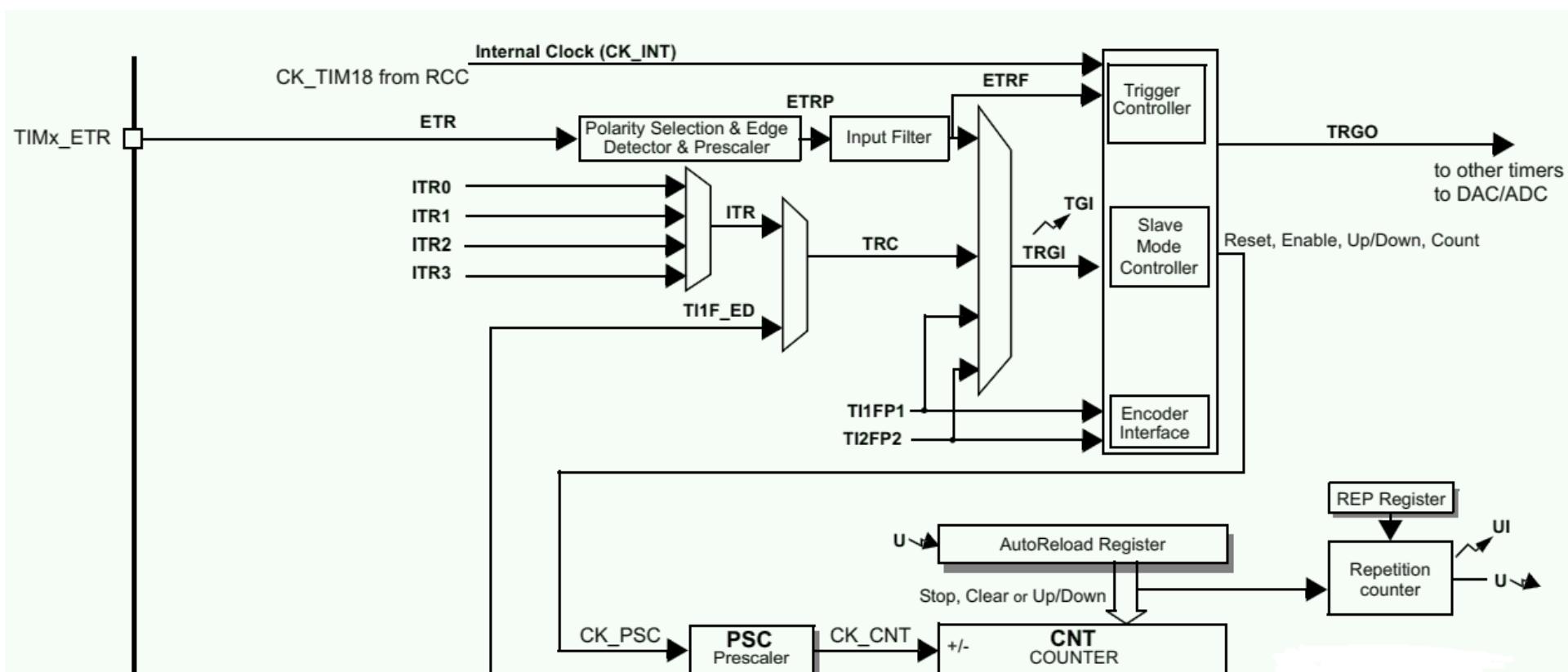
# Timers: Clock and Synchronization Unit

## □ Fontes de CLOCK:

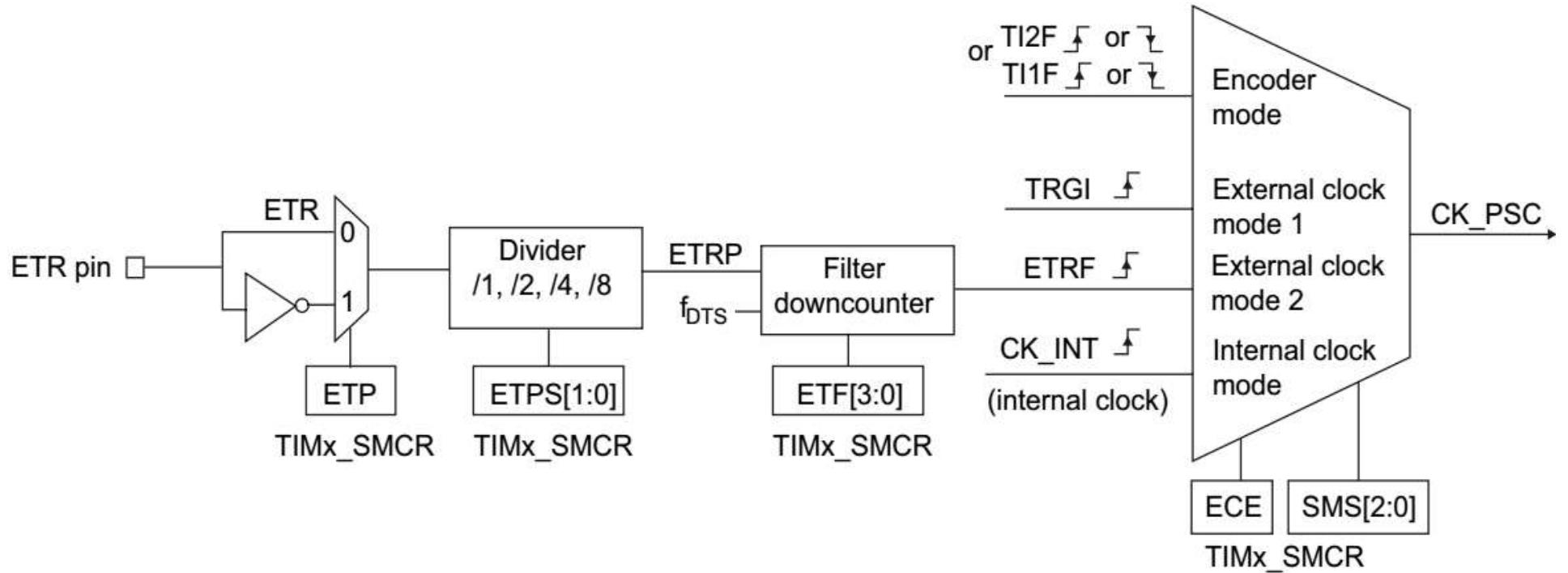
- Internal Clock (CK\_INT)
- External Clock Mode 2 (ETR2)

## □ Fontes de TRIGGER (TRGI):

- ETR1
- Internal Clock (ITRx)
- TI1F\_ED ou TI1\_FP1
- TI2FP2



## External Clock Mode 2 – ETR ou ETR2 (Contador)

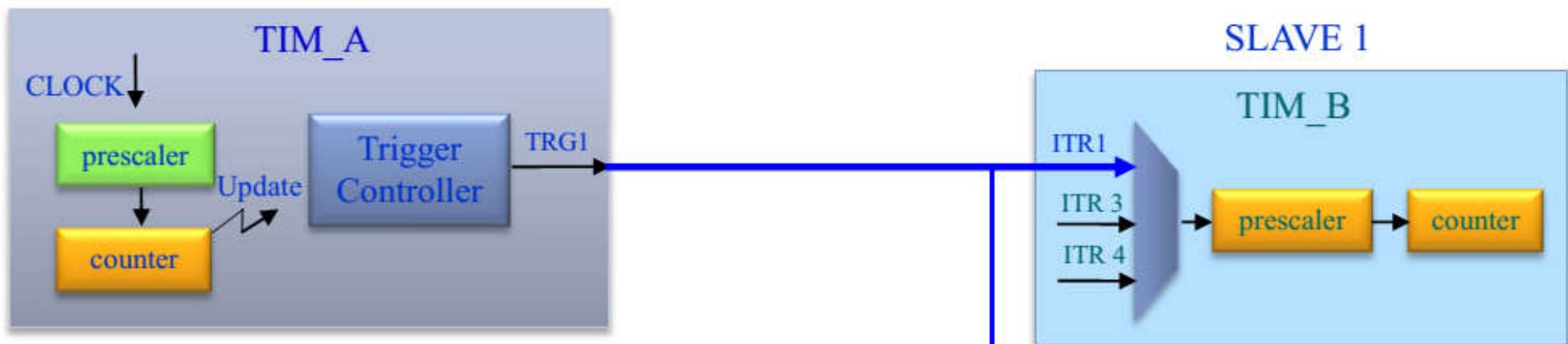


### Clock

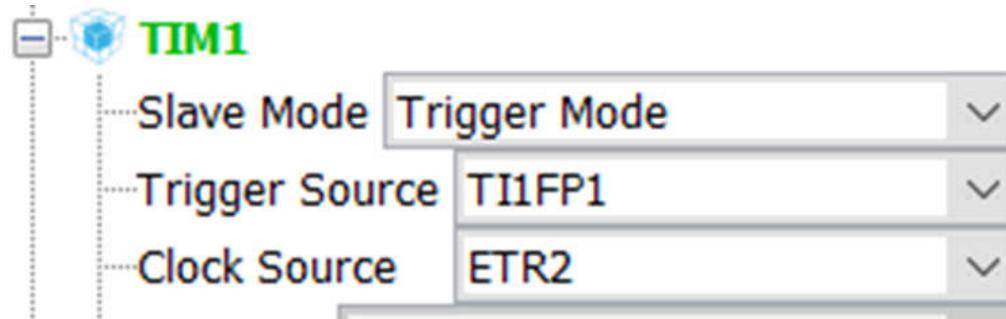
Clock Filter (4 bits value)	0
Clock Polarity	non inverted
Clock Prescaler	Prescaler not used

# Timer Synchronization – I

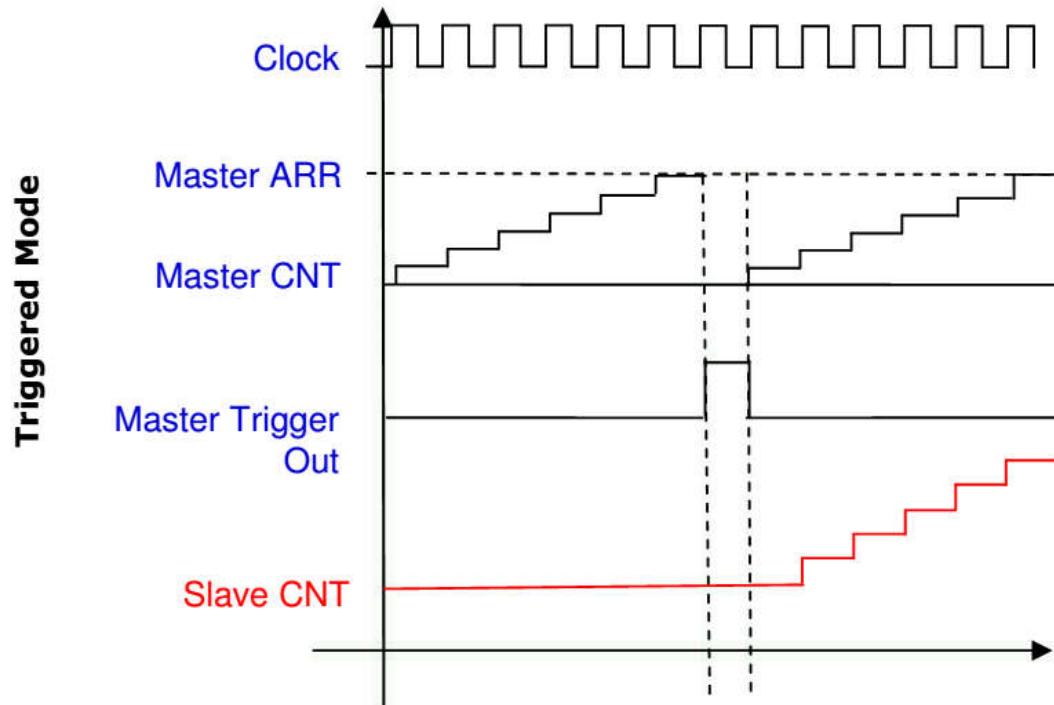
MASTER



- Mestre geraTrigger Output (TRGO)
- Escravo utiliza *internal trigger* (TRGI) para:
  - Reset
  - Gated
  - Trigger



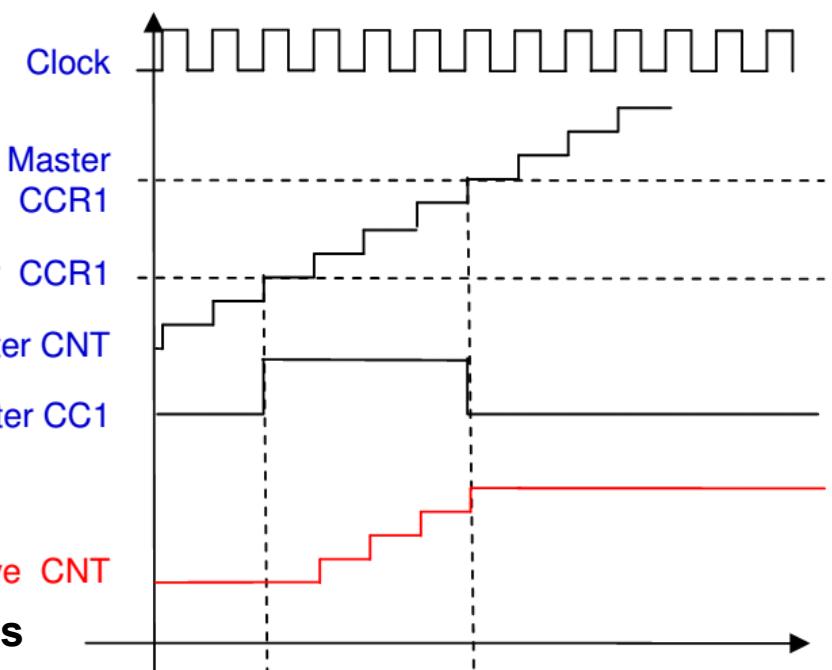
# TIMERS e Sincronização a Evento Externo - I



Timer escravo nos modos:

*Triggered*: dispara início da contagem

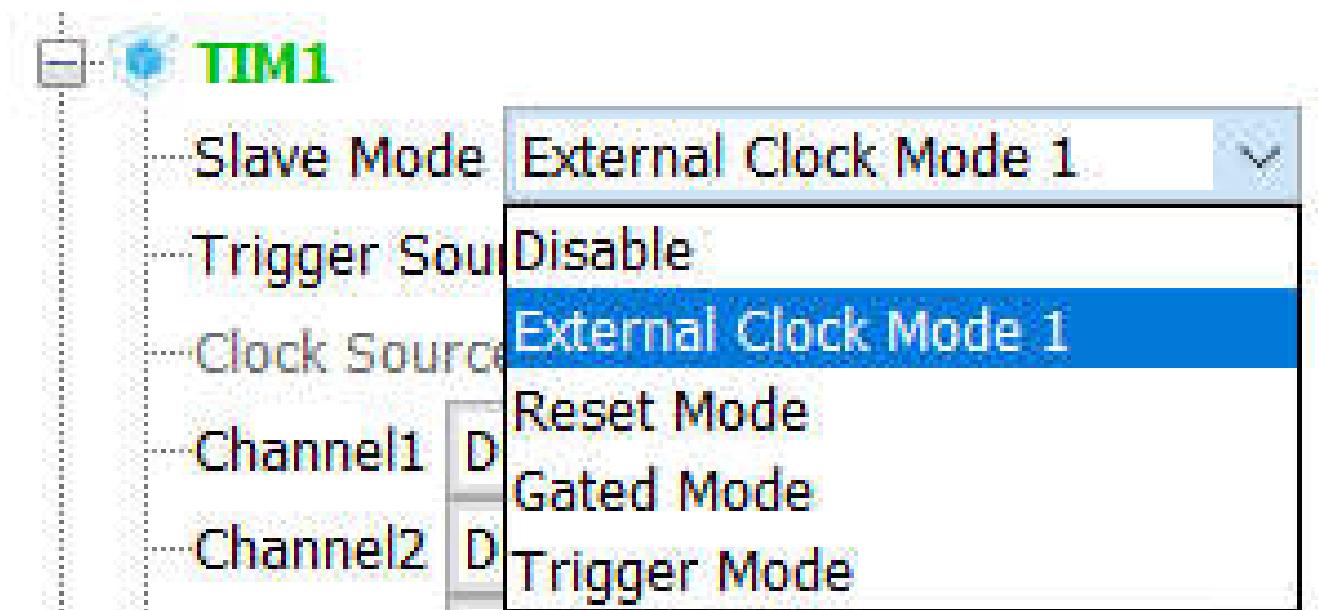
Gated Mode



*Gated* : Início e término da contagem são controlados

# TIMERS e Sincronização a Evento Externo - I

- ☐ Quando timer é configurado como mestre, seu Trigger Output (TRGO) é utilizado como *internal trigger* (ITR) pelo(s) timer(s) escravo(s) podendo resetar, iniciar ou interromper contagem de timer configurado como Escravo



- ☐ Exceto para caso **ECM1**, a fonte de clock do escravo será ETR2 ou *Internal Clock*

# TIMERS e Sincronização a Evento Externo - I

☐ Quando timer é configurado como mestre, seu Trigger Output (TRGO) é utilizado como *internal trigger* (ITR) pelo(s) timer(s) escravo(s) podendo resetar, iniciar ou interromper contagem de timer configurado como Escravo

☐ Exceto para caso **ECM1**, a fonte de clock do escravo será ETR2 ou *Internal Clock*

Escravo
Trigger Input (ITR)
Reset
Contador reinicializado por borda de subida de TRGI
Gated
Contagem ocorre quando TRGI = '1'
Triggered (não reseta)
Contagem é iniciada em resposta a TRGI
Combined Reset + Trigger Mode
Ambos acima (algumas famílias de uPs)
External clock mode 1 ( <b>ECM1</b> )
Contagem incrementada por borda de subida de TRGI

# TIMERS e Sincronização a Evento Externo - I

- ☐ Quando timer é configurado como mestre, seu Trigger Output (TRGO) pode ser utilizado como *internal trigger* (ITR) pelo(s) timer(s) escravo(s)
- ☐ Há oito opções para o sinal apresentado em TRGO

The screenshot shows the 'TIM1 Configuration' dialog box. At the top, there are four tabs: Parameter Settings (checked), User Constants, NVIC Settings, and DMA Settings. Below the tabs, it says 'Configure the below parameters :'. There is a search bar labeled 'Search : Search (Ctrl+F)' and a refresh button. The configuration is divided into sections: 'Counter Settings' and 'Trigger Output (TRGO) Parameters'. Under 'Trigger Output (TRGO) Parameters', the 'Master/Slave Mode' is set to 'Disable (no sync between this TIM (Master) and its S...)' and the 'Trigger Event Selection' is currently set to 'Reset (UG bit from TIMx\_EGR)'. A dropdown menu is open, listing eight options: 'Reset (UG bit from TIMx\_EGR)', 'Enable (CNT\_EN)', 'Update Event', 'Compare Pulse (OC1)', 'Output Compare (OC1REF)', 'Output Compare (OC2REF)', 'Output Compare (OC3REF)', and 'Output Compare (OC4REF)'.

# TIMERS e Sincronização a Evento Externo - I

- ☐ Quando timer é configurado como mestre, seu Trigger Output (TRGO) pode ser utilizado como *internal trigger* (ITR) pelo(s) timer(s) escravo(s)

- ☐ Há oito opções para o sinal apresentado em TRGO

Mestre  
Trigger Output

Reset

Pulso de sincronização em TRGO quando contador for resetado

Enable

Pulso de sincronização em TRGO quando a contagem for iniciada

Update

*Update Event* em TRGO ao término da contagem.  
Fonte de clock para outro timer.

Compare

Pulso de sincronização em TRGO quando contagem iguala-se a registrador Capture/compare

OC1Ref, OC2Ref, OC3Ref, OC4Ref

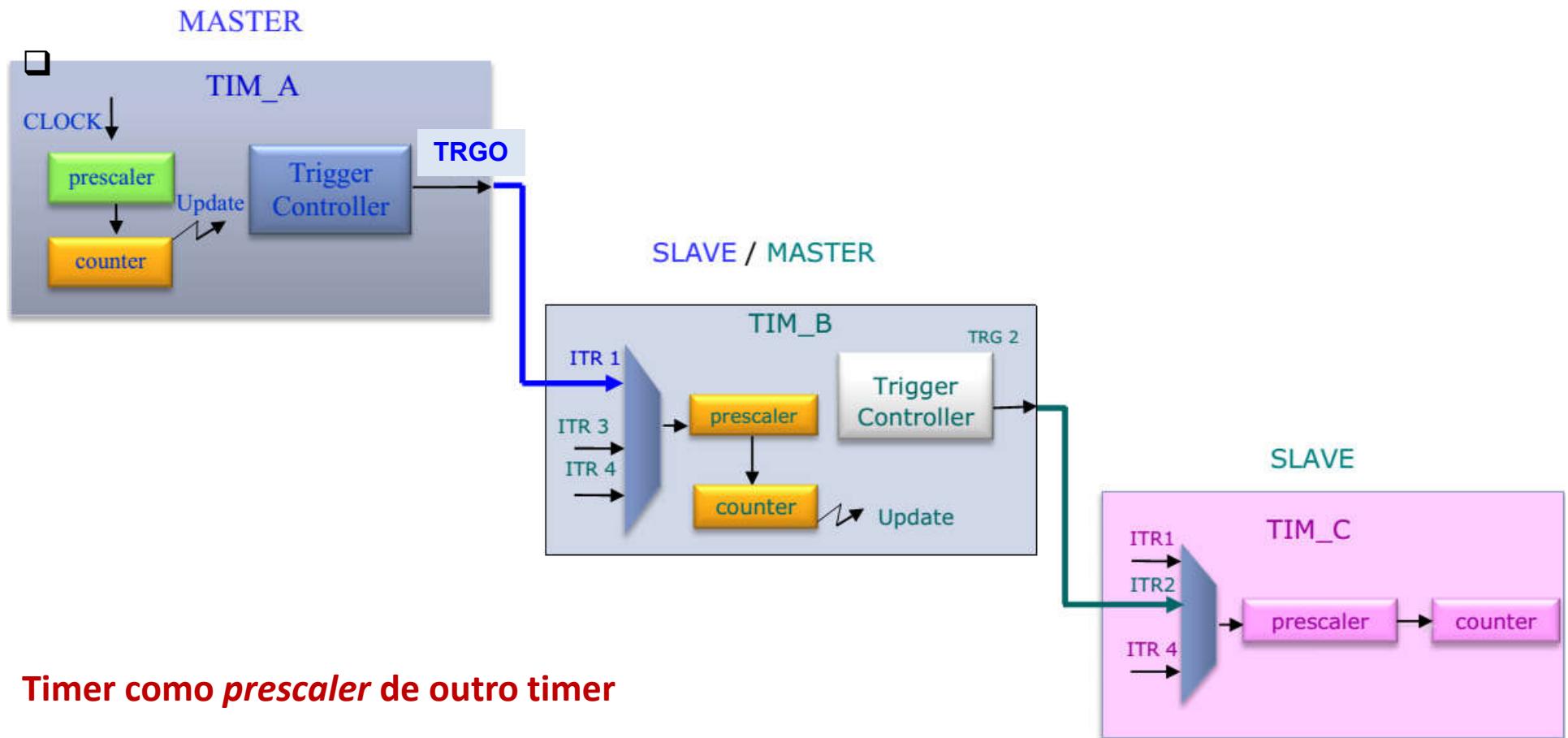
Um destes sinais pode ser apresentado a TRGO

## Timer Synchronization – II

□ Os timers podem ser internamente conectados em cascata.

□ Mestre gera Trigger Output (TRGO)

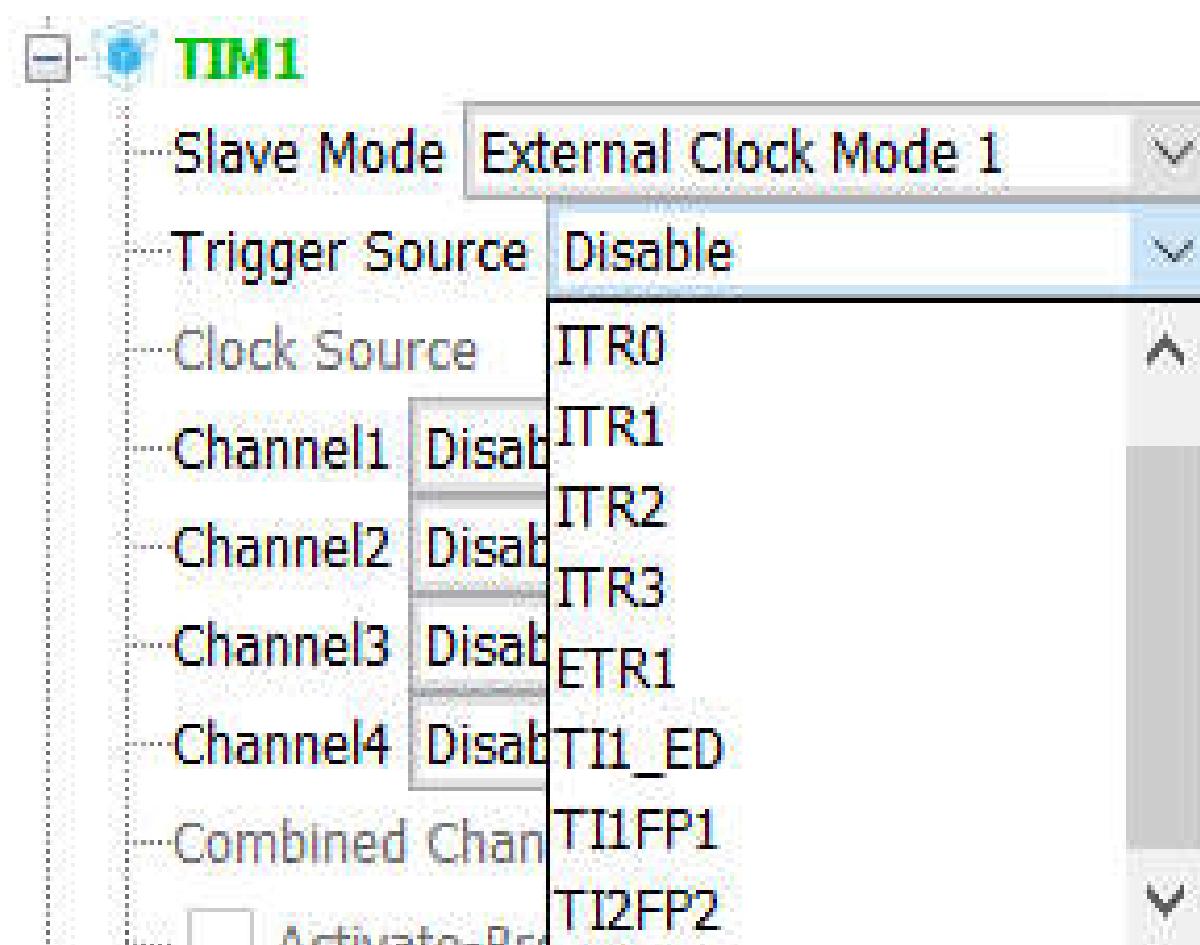
□ Escravo utiliza *internal trigger* (ITR) do mestre como clock



Timer como *prescaler* de outro timer

## **External Clock Mode 1 – Fontes de Trigger**

- ETR1
- ITRx (*Internal TRigger*): utiliza TRGO suprido por outros timers
- Clock aplicado aos pinos *Capture/Compare*: TI1 ou TI2



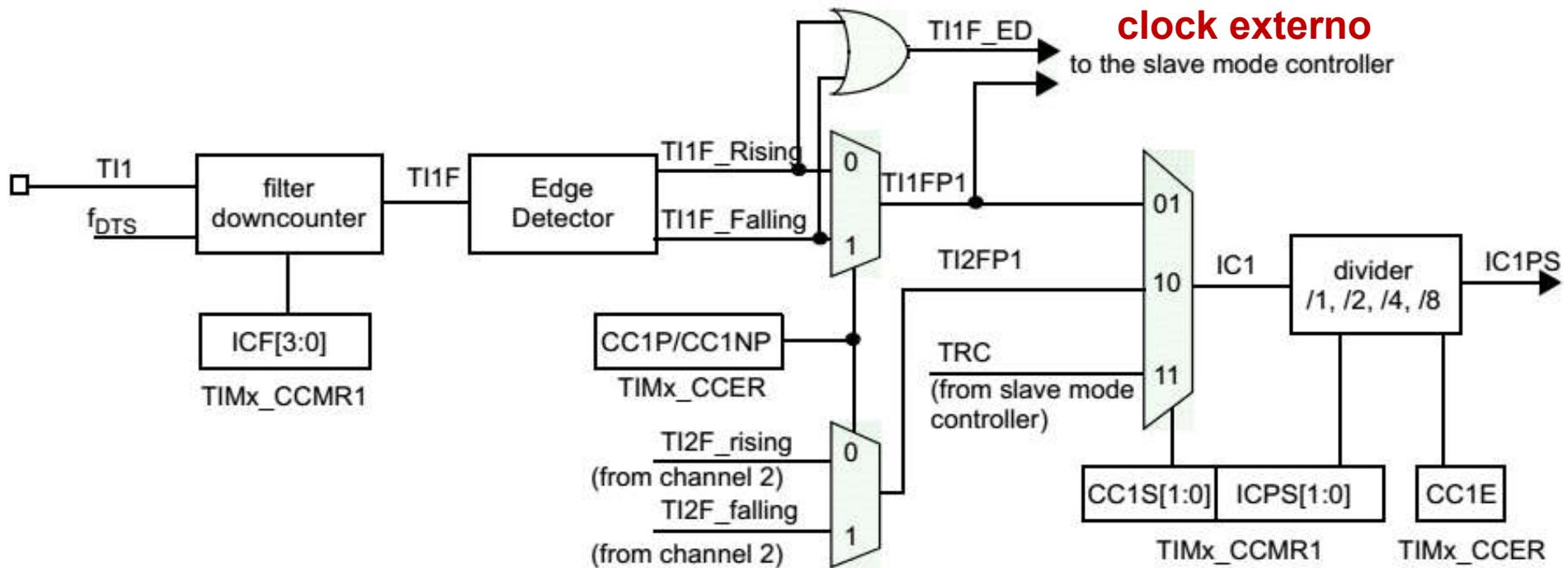
Pino TI1: TI1FP1 ou TI1F\_ED

Pino TI2: TI2FP2

# External Clock Mode 1

- Clock aplicado aos pinos Capture/Compare: TI1 ou TI2

- Pino TI1: TI1FP1 ou TI1F\_ED
- Pino TI2: TI2FP2



**TI<sub>x</sub>:** Entrada de clock externo

**TI<sub>x</sub>F:** Filtered Timer input x (clock externo filtrado)

**TI<sub>x</sub>F<sub>Py</sub>:** Detector de borda com seleção de polaridade (*rising* ou *falling*)

**TI1F\_ED :** TI1 Filtered Edge Detector (detecta borda de descida e subida)

- Encoders são utilizados para medir posição e velocidade de sistemas em movimento (linear ou angular). Geram pulsos defasados de 90 graus para permitirem identificar direção de deslocamento.

## Encoder Interface

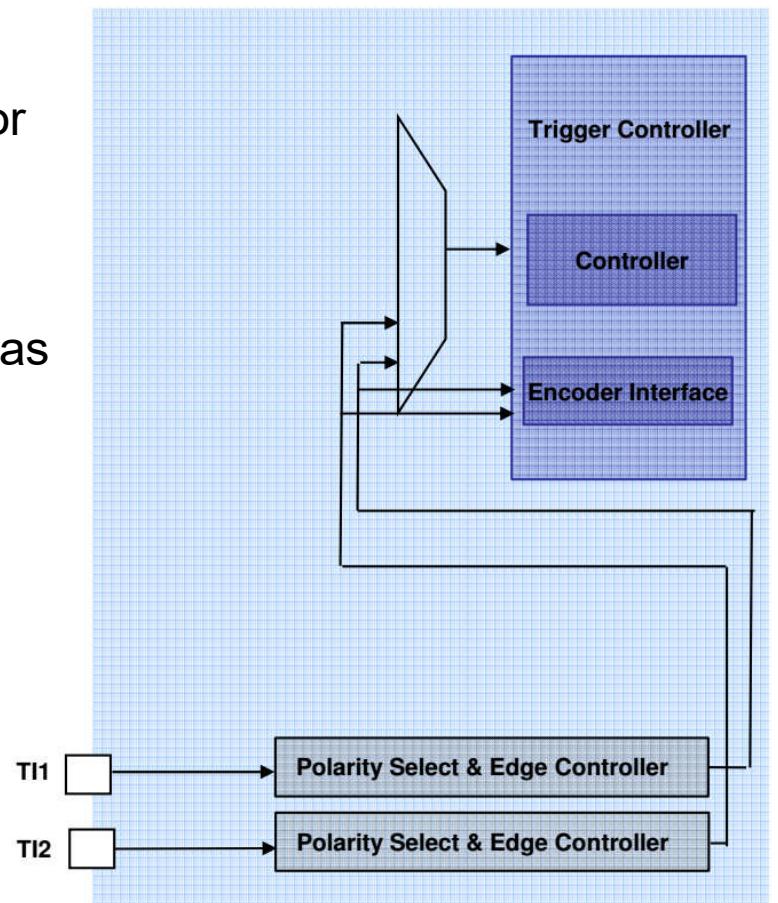
- Esta interface atua como receptora de clock externo com seleção de direção

- O contador supre informação da posição atual (por exemplo, posição angular de rotor elétrico)

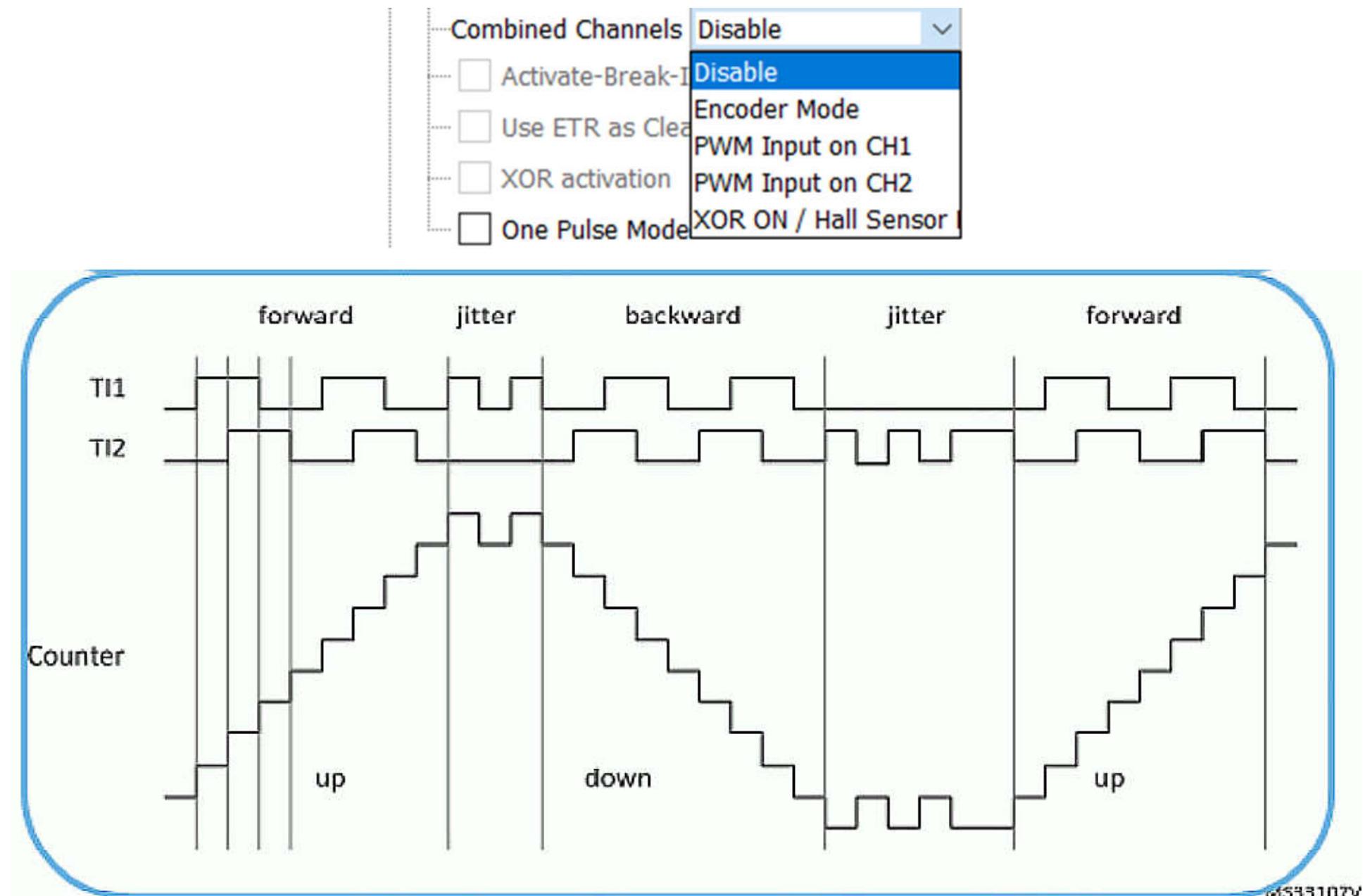
- Informações de velocidade e aceleração são obtidas a partir da contagem do número de pulsos em dado período (estabelecido por um outro timer)

- Exemplo de interface de encoder com microcontrolador:

OBS: A terceira saída do encoder (indica posição de referência (zero)) pode ser conectado a interrupção externa que resete a contagem.



# Exemplo de sinais gerados por *Encoder*



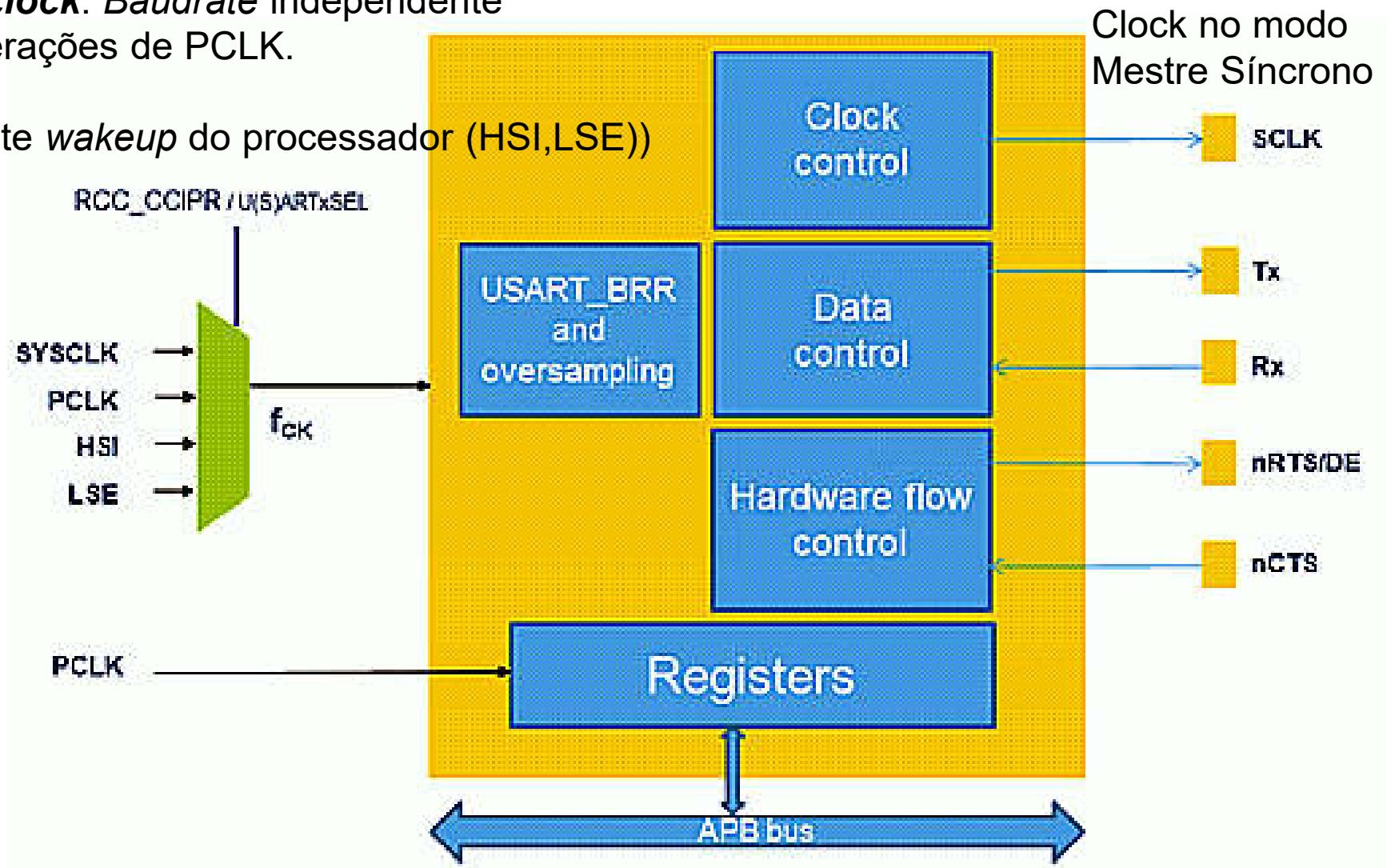
## **USART** – *Universal Synchronous/Asynchronous Receiver/Transmitter Interface*

- ❑ 4 USARTs e 4 UARTs
- ❑ Gerador fracional de taxa de transmissão/recepção (*Baud Rate*)
- ❑ Tamanho programável da palavra a ser transmitida : 8 ou 9 bits
- ❑ Número de *Stop Bits* configurável:
  - **1 stop bit**
  - **2 Stop bits**
  - **0,5 stop bit: em modo Smartcard**
  - **1,5 stop bits: em modo Smartcard**
- ❑ Suporte para:
  - Modem - controle de fluxo (CTS/RTS)
  - SPI mestre
  - Comunicação Multi-processadores
  - LIN (*Local Interconnection Network*),
  - Protocolo *Smartcard*
  - IrDA (*Infrared Data Association*)

# USART – Diagrama de Blocos

**Dual clock:** Baudrate independente de alterações de PCLK.

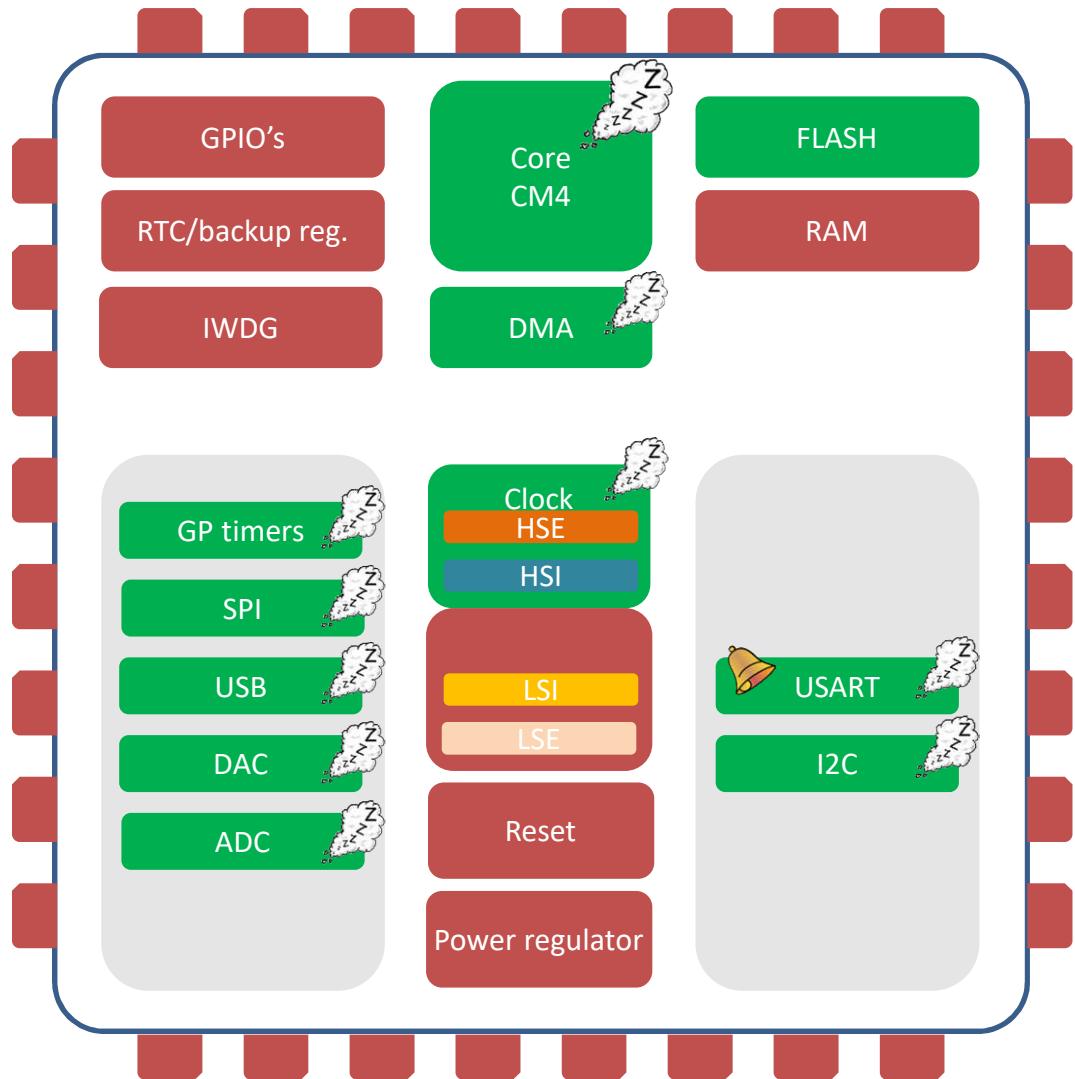
(Permite wakeup do processador (HSI,LSE))



PCLK: *Peripheral clock*  
(APB clock - Clock default)

DE (*Driver Enable*): compartilha pino de IO do nRTS; usado no protocolo RS-485

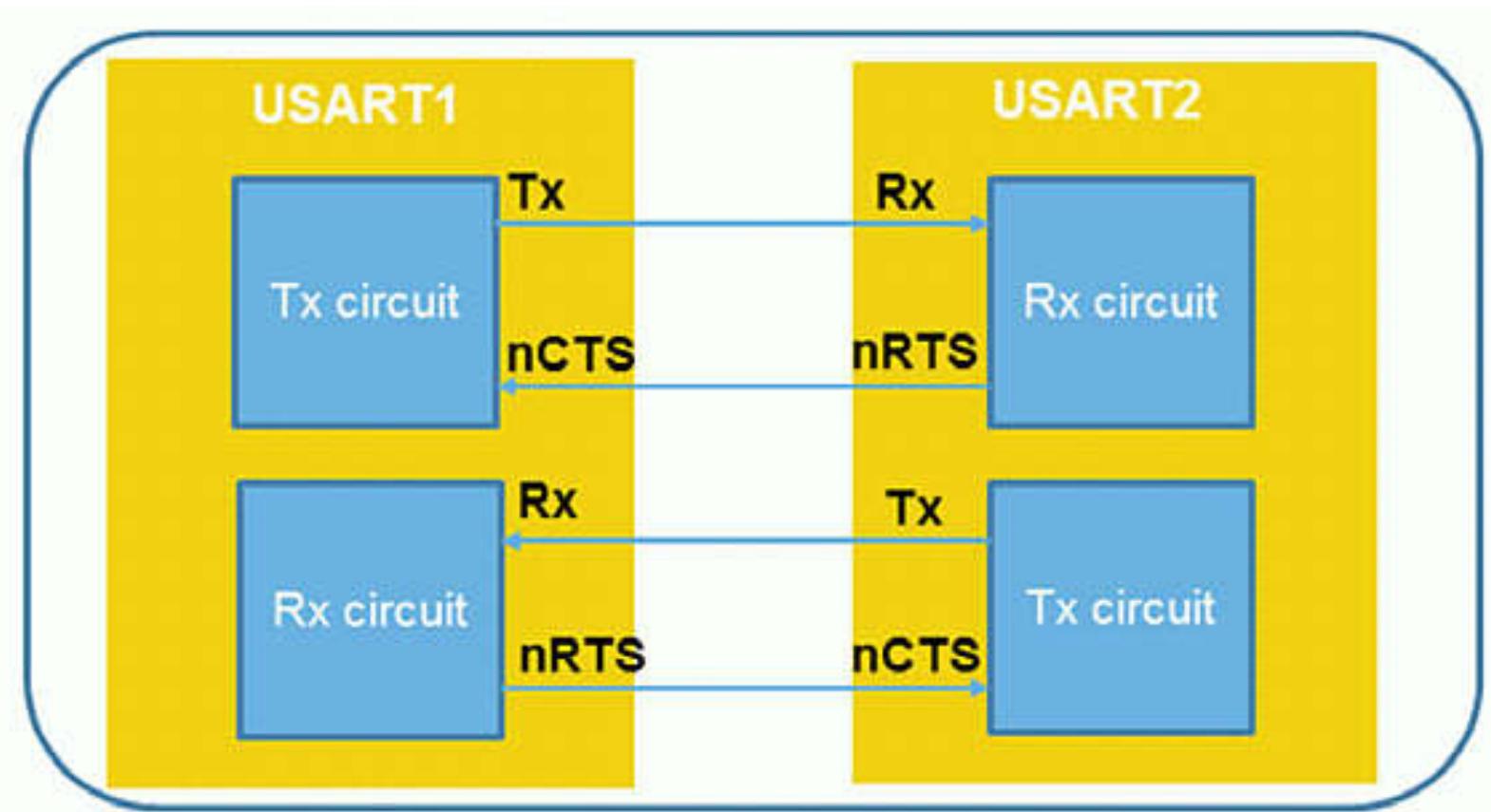
# STOP Mode



- Core é suspenso
- Clock HSE é suspenso
- Conteúdo da SRAM e registradores é preservado
- Periféricos com clock HSI, LSI, LSE podem funcionar
- GPIO's retém configuração

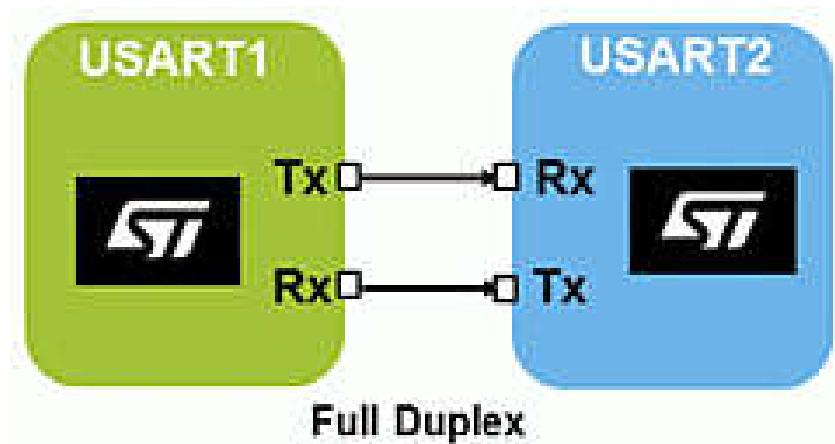


## USART: *Flow control por handshaking*

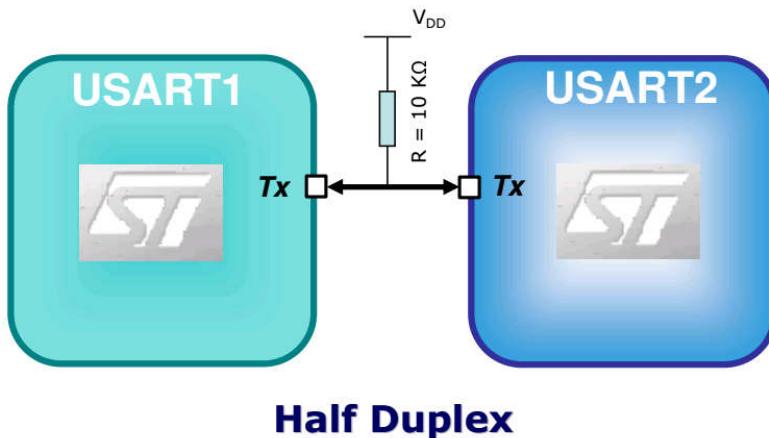
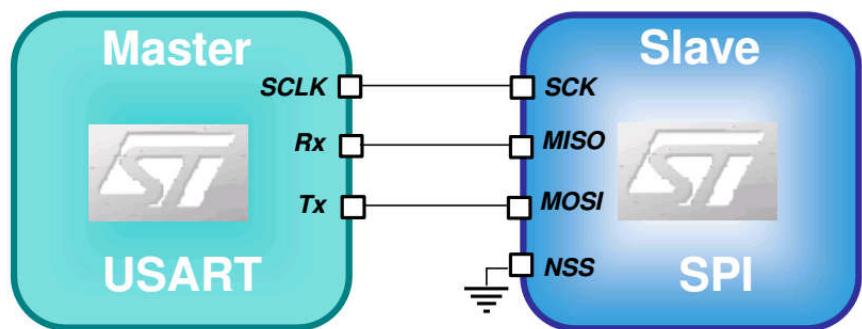


## USART: Modos Full e Half Duplex

### Modo assíncrono *Full Duplex*

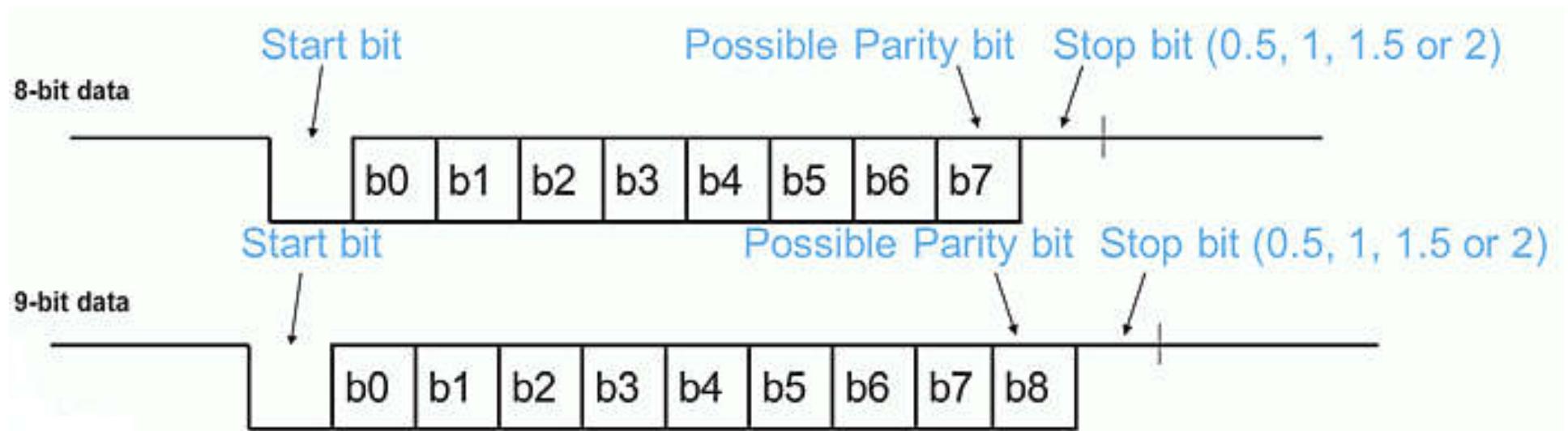


### *Full-duplex three-wire síncrono* (apenas para USART em modo Mestre) saída de clock em SCLK



### *Single wire Half duplex*: Utiliza apenas Tx (Tx e Rx são conectados internamente)

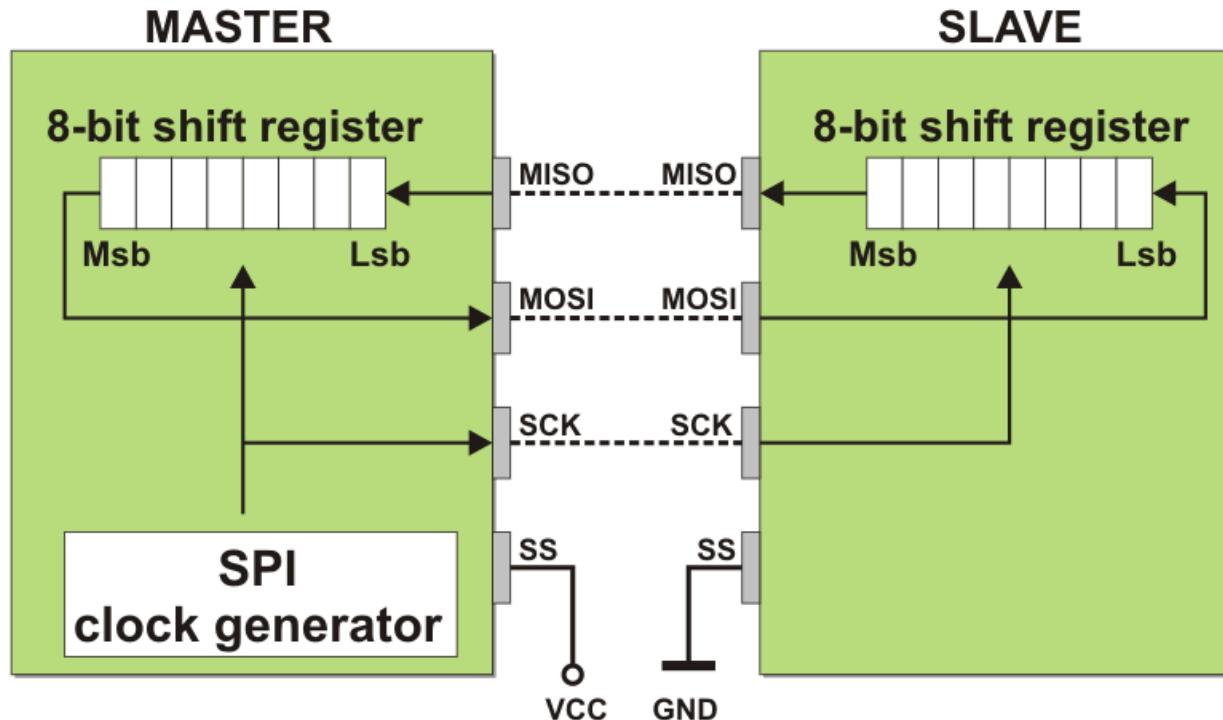
## USART: Modo Assíncrono



- **Parity control:** Bit de paridade gerado na transmissão e verificado na recepção (fazer PCE='1' no registrador USART\_CR1).
- Parity error => PE flag = '1' e interrupção é gerada

# SPI: *Serial Peripheral Interface*

## Transmissão Serial Síncrona



**MOSI:** *Master Out Slave In*

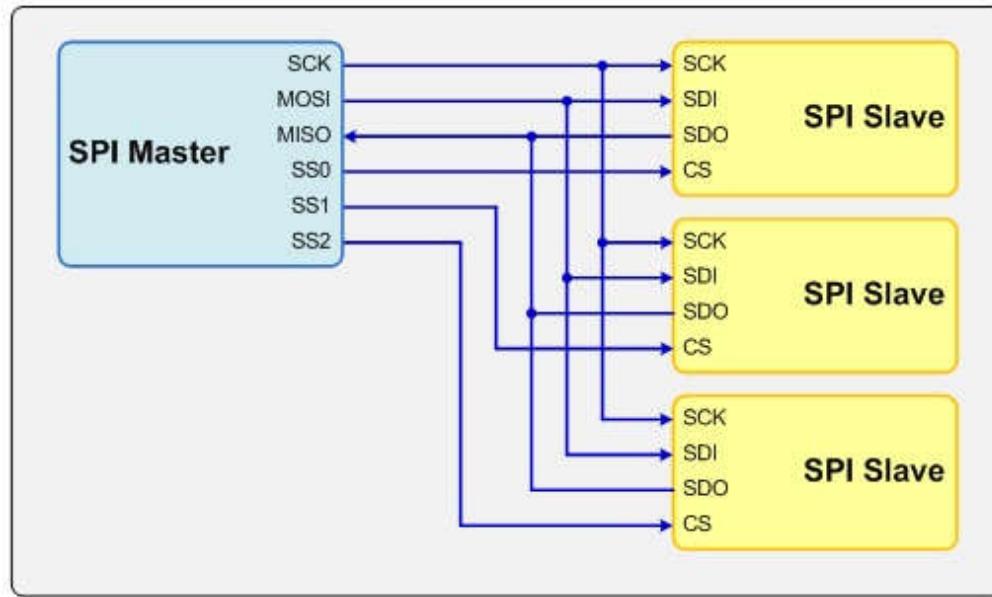
**MISO:** *Master In Slave Out*

**SCK:** *Serial Clock (suprido pelo mestre)*

**SS:** *Slave Select*

- Full Duplex, sem controle do fluxo de dados (para receber, necessário enviar dado)
- Velocidade de transmissão de dados de dezenas de Mbits/s

# SPI - Conexão com múltiplos escravos



**SCK:** Serial Clock (suprido pelo mestre)

**MOSI:** Master Out Slave In

**MISO:** Master In Slave Out

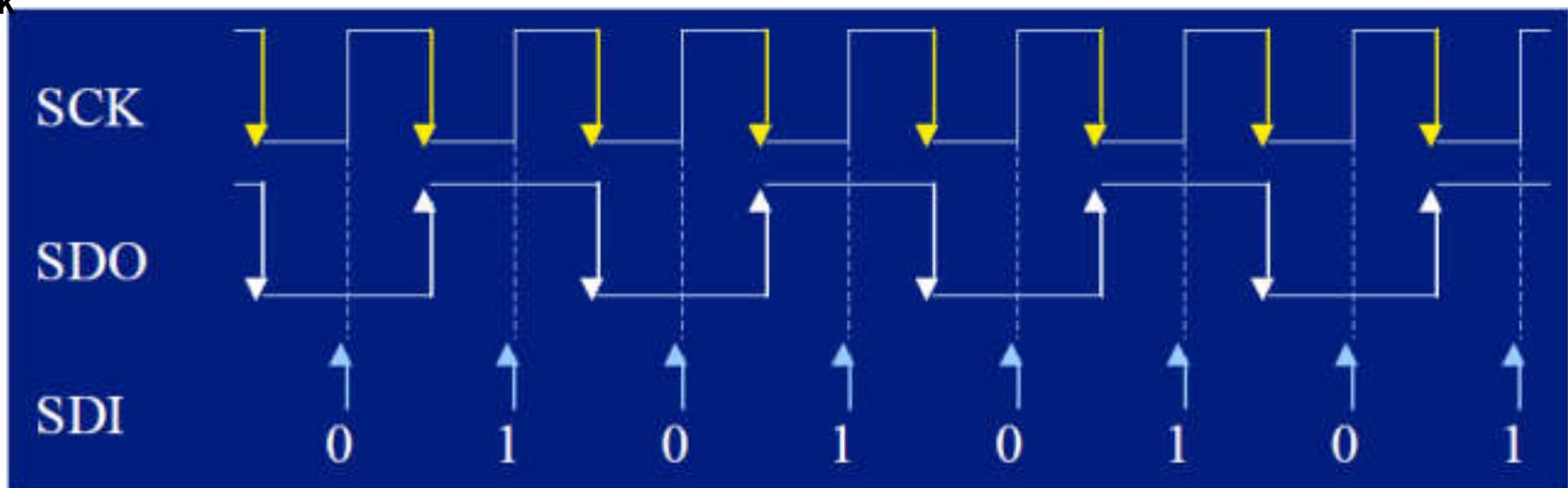
**SDI:** Serial Data In

**SDO:** Serial Data Out

**CS, SSx:** Chip Select, Slave Select

## SPI – Exemplo de envio/recepção de dados – CPOL = CPHA = 1

- Dado atualizado pelo mestre na borda de descida; dado amostrado pelo escravo na borda de subida do clock



**CPOL: 0 - clock normalmente em '0'**

**CPHA: 0 - dado amostrado pelo escravo na borda de subida do clock  
(dado atualizado pelo mestre na borda de descida)**

**CPOL: Polaridade do Clock**  
**CPHA: Fase do Clock**

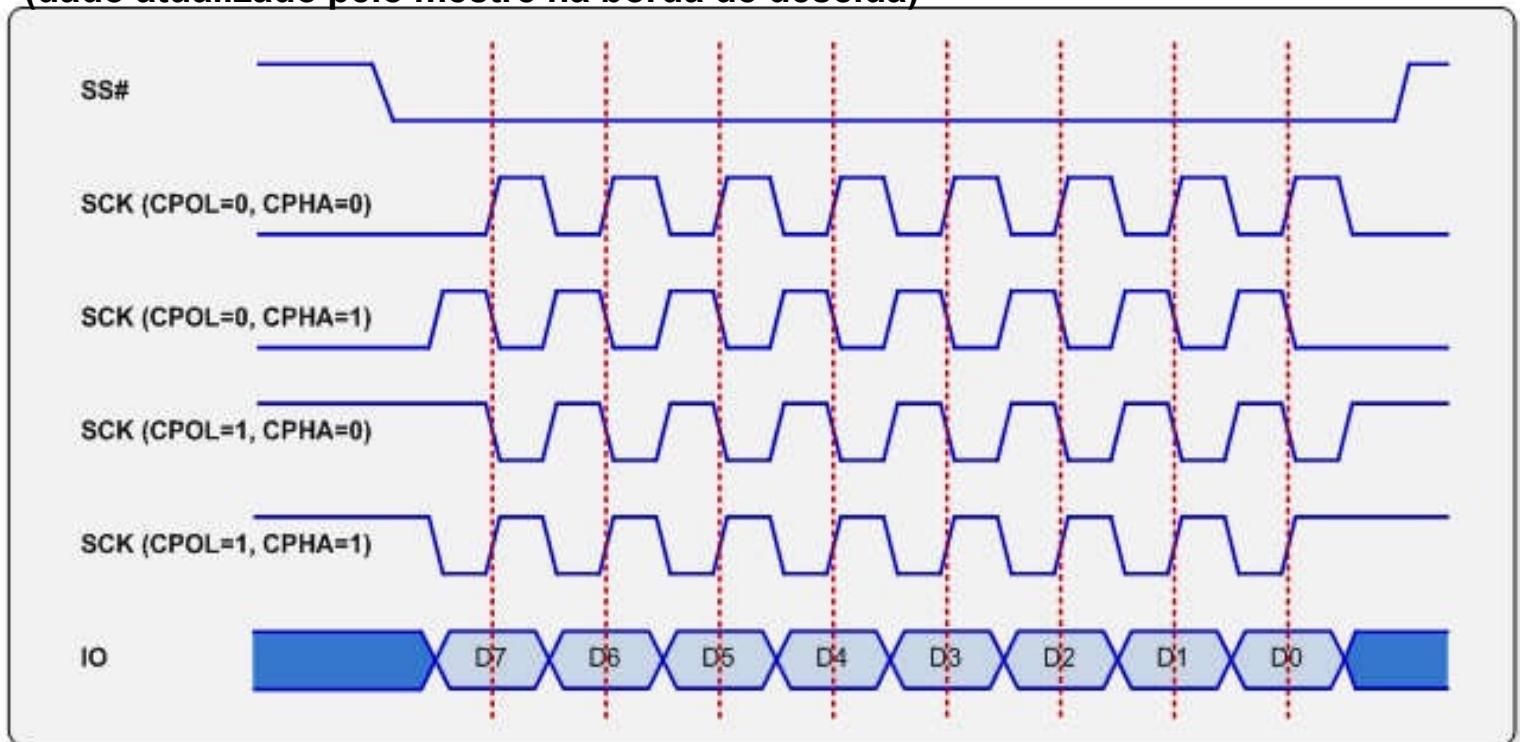
**CPHA: 1 - dado amostrado pelo escravo na borda de descida do clock  
(dado atualizado pelo mestre na borda de subida)**

**CPOL: 1 - clock normalmente em '1'**

**CPHA: 0 - dado amostrado pelo escravo na borda de descida do clock  
(dado atualizado pelo mestre na borda de subida)**

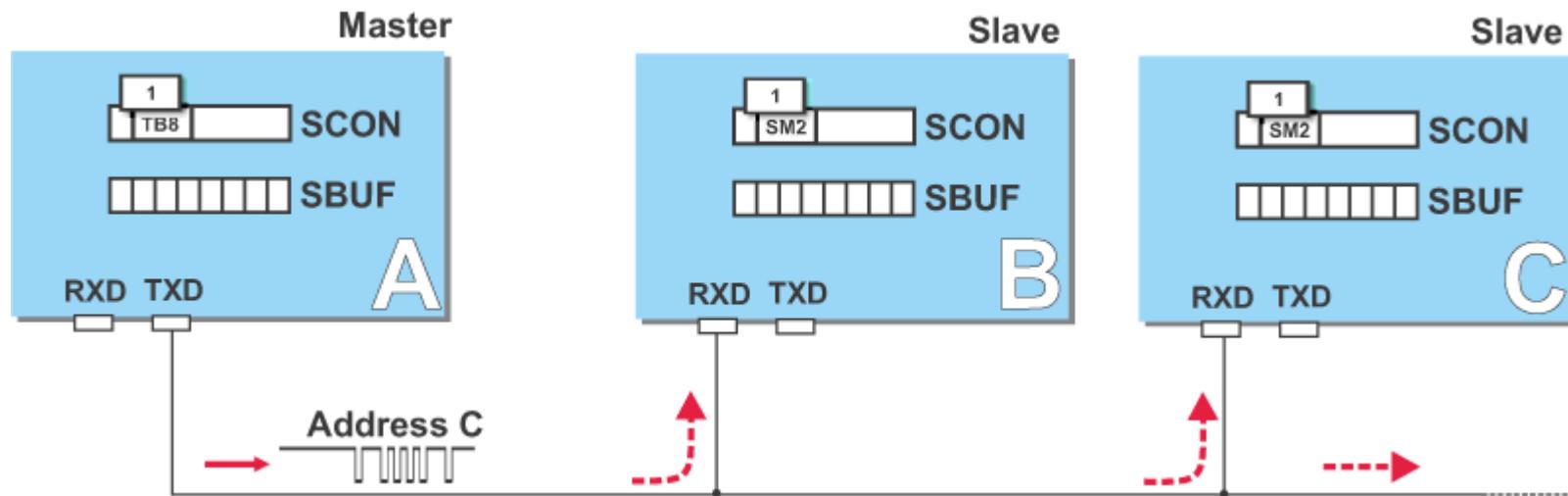
**CPHA: 1 – dado amostrado pelo escravo na borda de subida do clock  
(dado atualizado pelo mestre na borda de descida)**

Mode	CPOL	CPHA
0	0	0
1	0	1
2	1	0
3	1	1

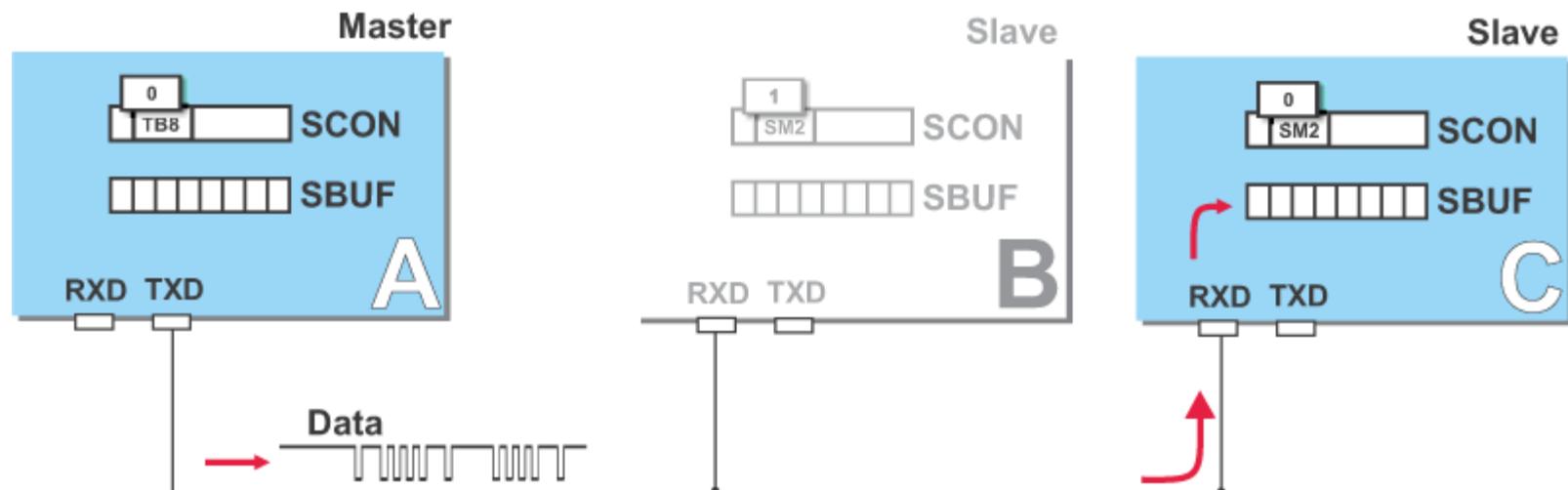


# Comunicação Multi-processadores - 8051

## Transmissão de endereço para selecionar 8051



## Comunicação com 8051 selecionado



# Comunicação Multi-processadores – STMF429 - I

- ❑ Dispositivos não endereçados podem ser colocados em *mute mode* ( $RWU=‘1’$ ):
  - Bits de status da recepção não são setados
  - Interrupções da recepção são inibidas
- ❑ A USART sai ou entra do *mute mode* por meio de:
  - *Idle Line detection* (*Wake bit* = ‘0’ em `USART_CR1`)

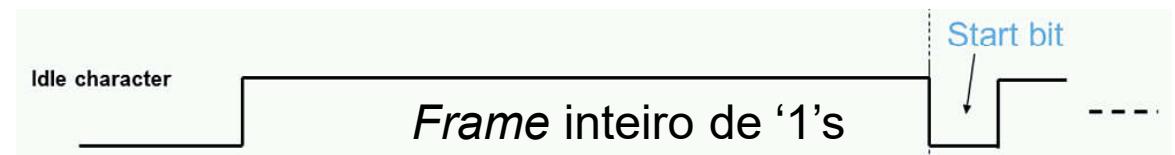
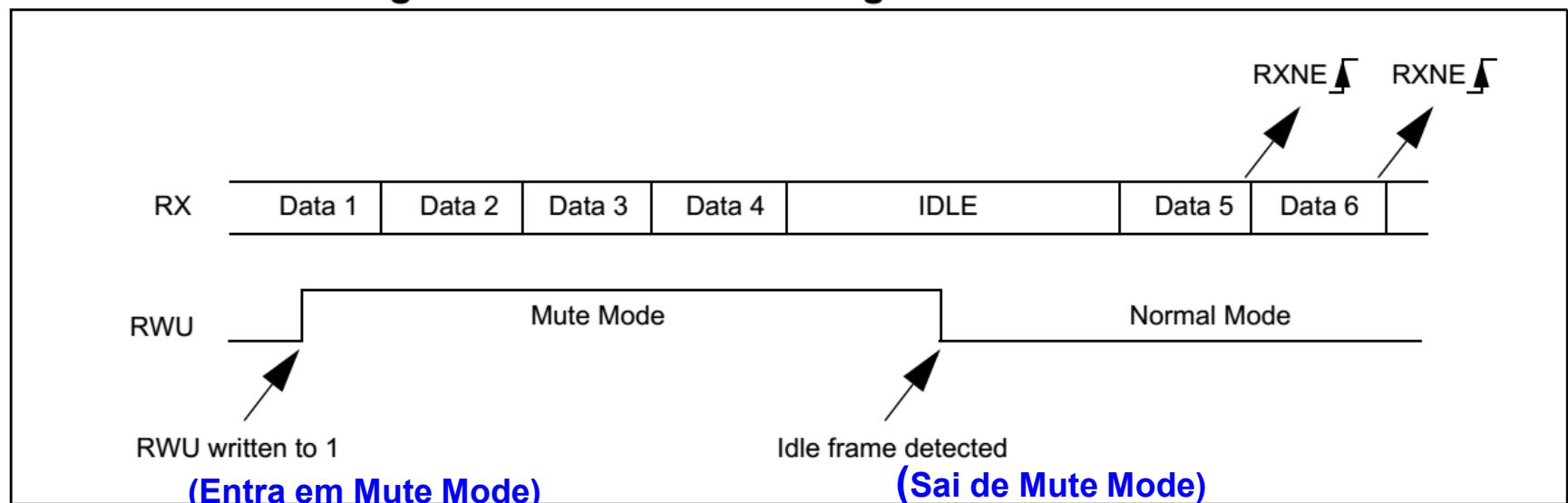


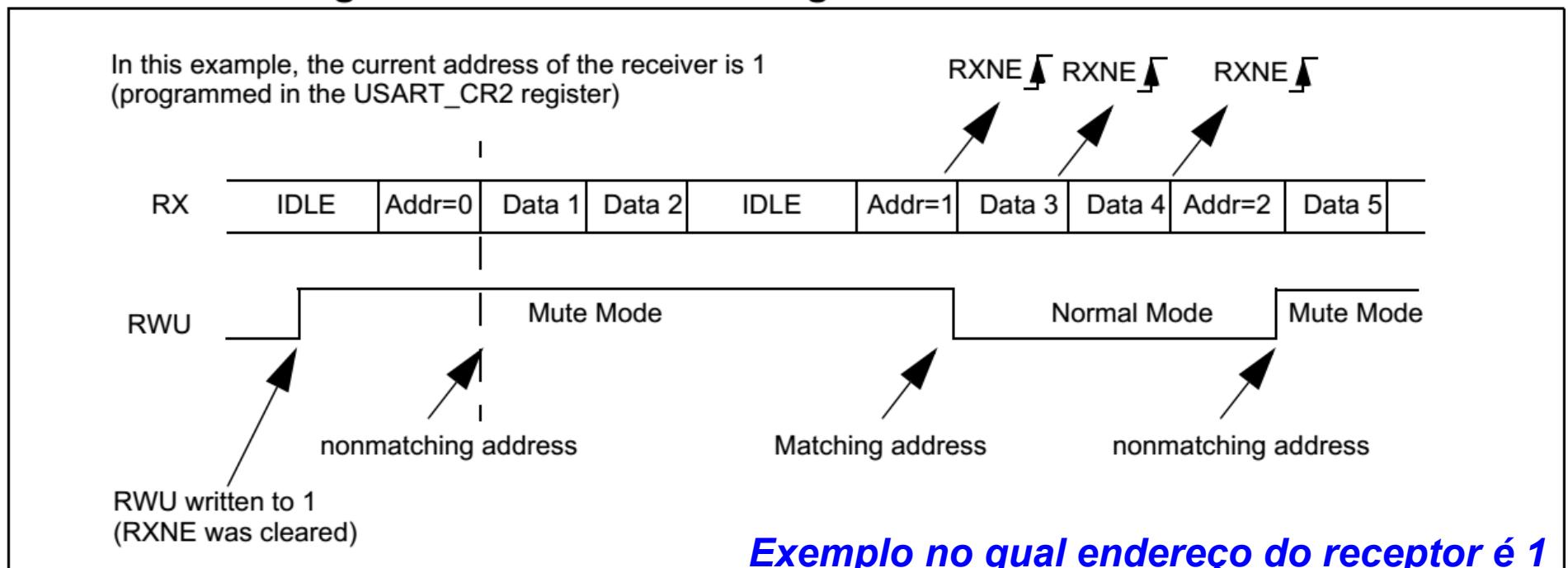
Figure 303. Mute mode using Idle line detection



# Comunicação Multi-processadores – STMF429 - II

- *Address Mark detection (Wake bit = '1' em USART\_CR1)*
- Dado contém endereço se MSB = '1': endereço do receptor visado nos 4 LSB
  - Receptor compara endereço com aquele com o qual foi programado (USART\_CR2)
  - A USART entra em *mute mode* quando endereço recebido não corresponde ao seu

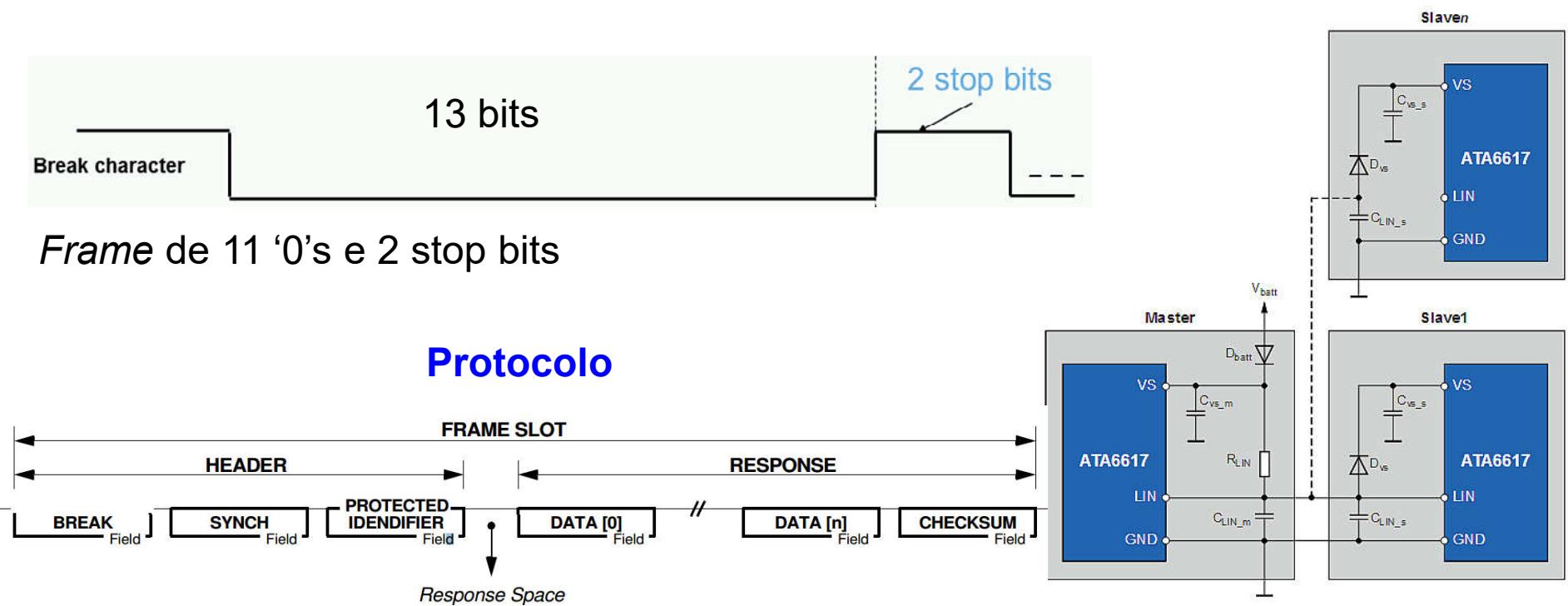
**Figure 304. Mute mode using address mark detection**



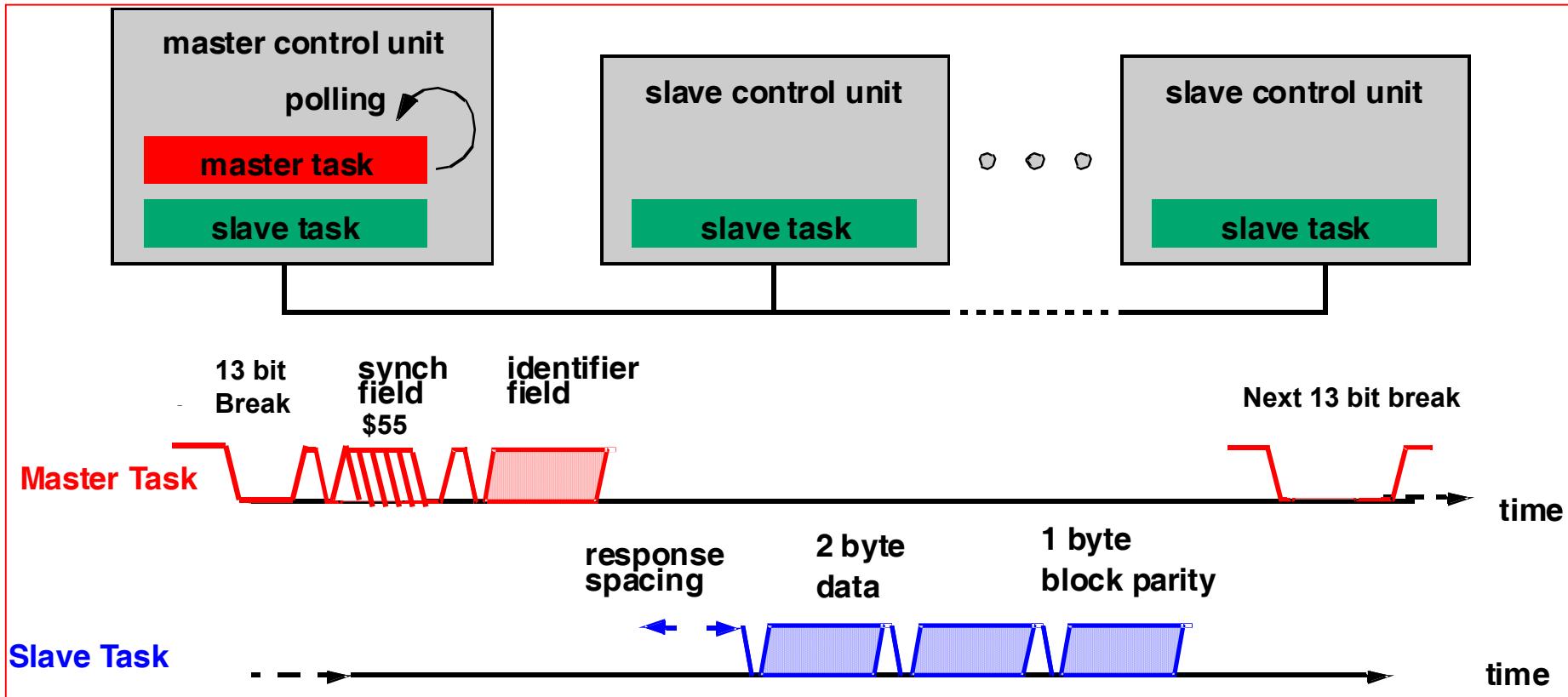
# Comunicação Multi-processadores

*Local Interconnection Network*

- ❑ O LIN bus (protocolo baseado em comunicação serial) foi proposto pela indústria automobilística para aplicações simples de chaveamento (sensor de chuva, trava porta, teto solar e outros)
- ❑ Trata-se de um barramento *single master / multiple slaves* que transmite dados via um único fio
- ❑ Até 40m, máxima taxa de 19,2 kibits/s (reduzir EMI), até 16 escravos.



# Comunicação Multi-processadores – LIN



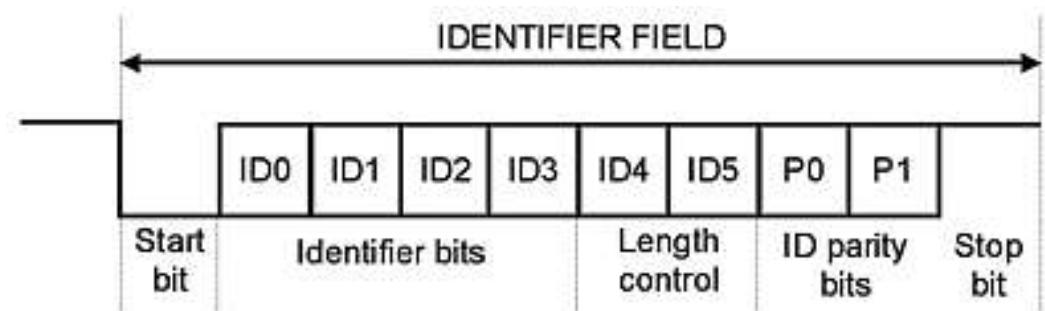
**Identifier bits:** código da mensagem (0 a 15)

**Length control:** Número de bytes esperado em resposta (2, 4 ou 8)

**Parity bits:**

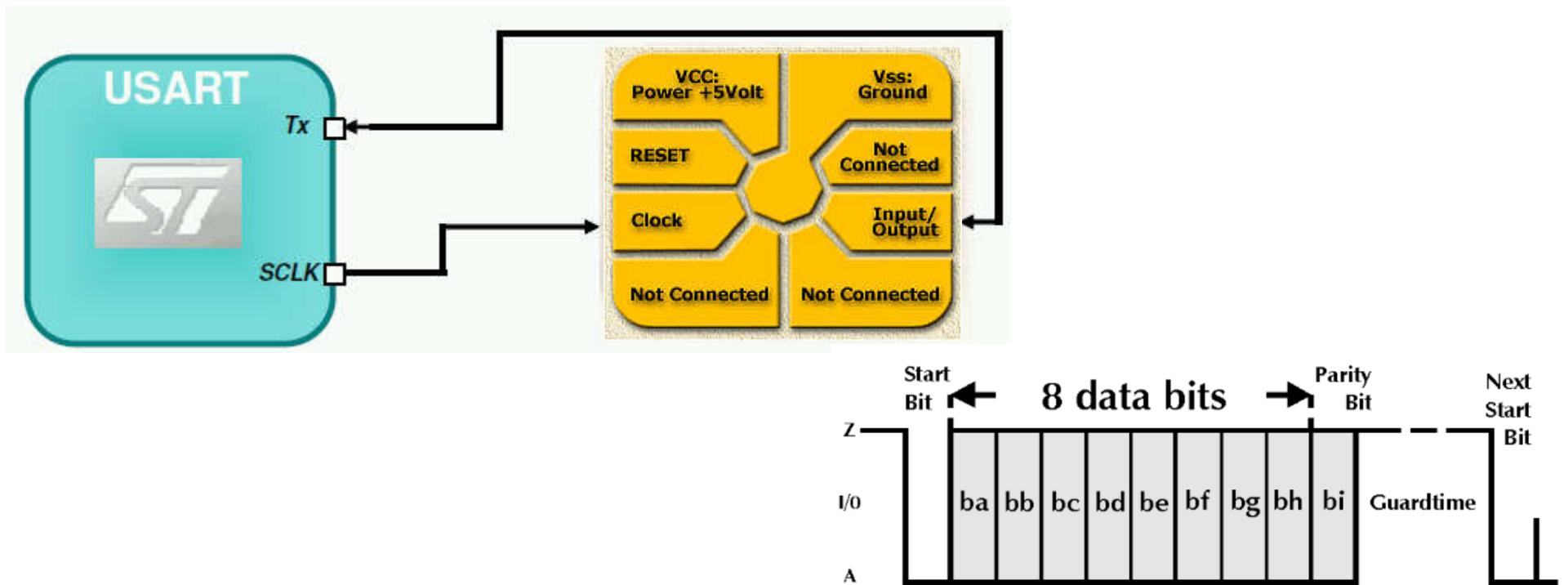
$$P0 = (ID0 \text{ xor } ID1 \text{ xor } ID2 \text{ xor } ID4)$$

$$P1 = ! (ID1 \text{ xor } ID3 \text{ xor } ID4 \text{ xor } ID5)$$



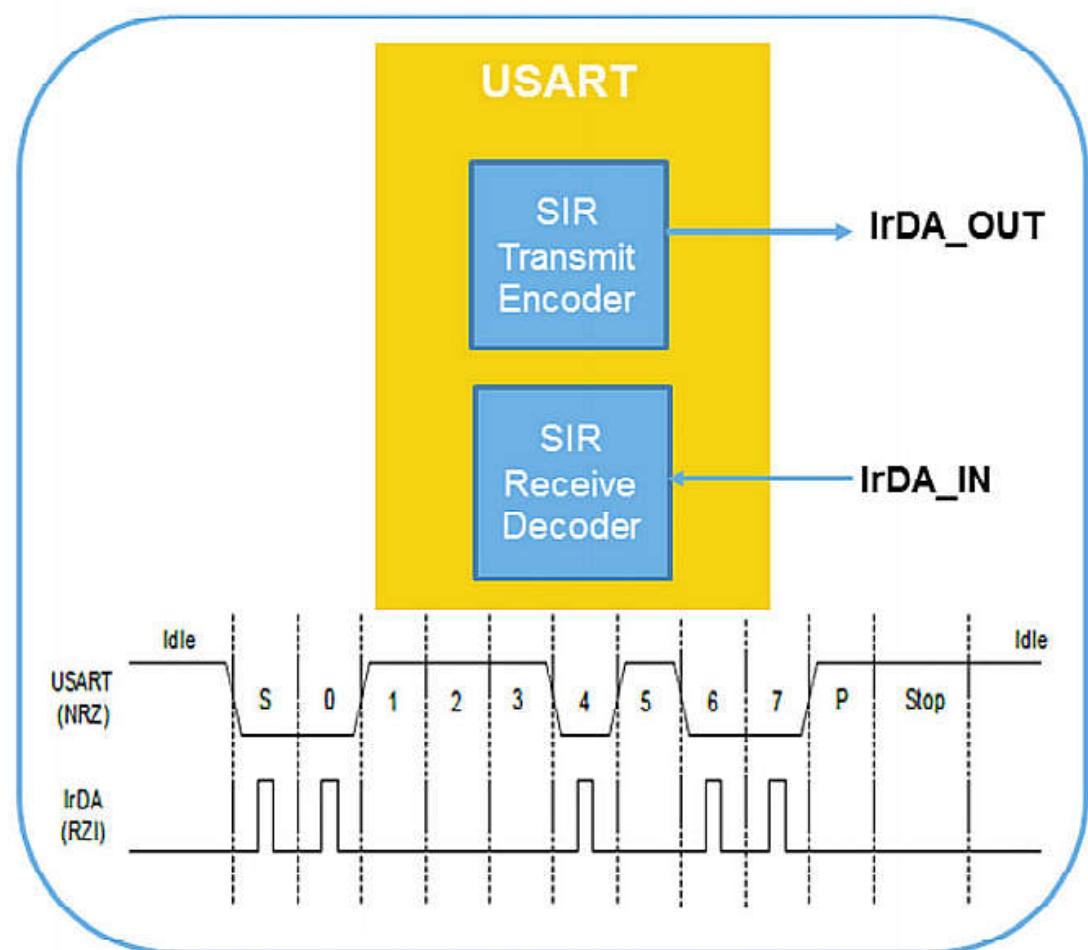
## USART: Smart Card Mode

- USART suporta emulação de *Smart Card* (ISO 7618-3)
- Half-Duplex (Tx) e saída de Clock (SCLK)
- *Guard Time Programável* (similar a *stop bit*)
- 9 bits de dados – 0,5 *Stop bit* na recepção; 1,5 *Stop bits* na transmissão
- Geração de erro de paridade com transmissão de NACK



## USART: IrDA (Infrared Data Association)

- A USART suporta especificações serial IrDA (SIR)
  - Até 115,2 kibits/s (half-duplex)
  - 1 start bit, 8 bits de dados, 1 stop bit
  - Distância padrão: 1m
  - USART => NRZ (*Non Return to Zero*) => nível lógico cte durante  $\Delta t$
  - IrDA => RZI (*Return to Zero Inverted*) => “0” => pulso de curta duração (3/16) durante  $\Delta t$
  - SIR converte NRZ para RZI.



## USART: Auto baud rate

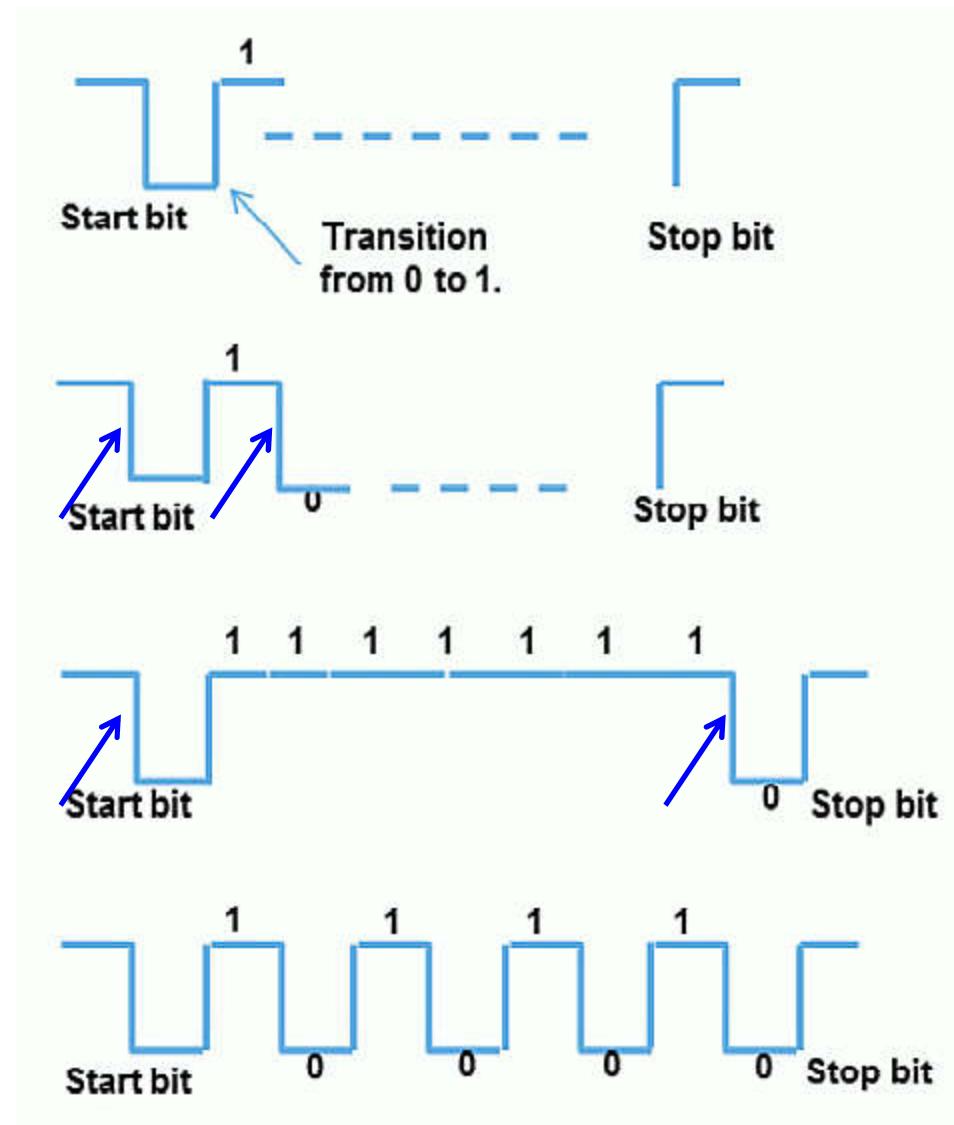
- O USART *receiver* pode detectar automaticamente o *baudrate* da transmissão a partir da recepção de um caractere:

- Que se inicia com '1': mede duração do startbit

- Que se inicia com '10': mede duração entre bordas de descida consecutivas (melhor exatidão no caso de transições lentas)

- 0x7F: mede duração entre bordas de descida consecutivas.

- 0x55: mede duração de cada bit até bit 6



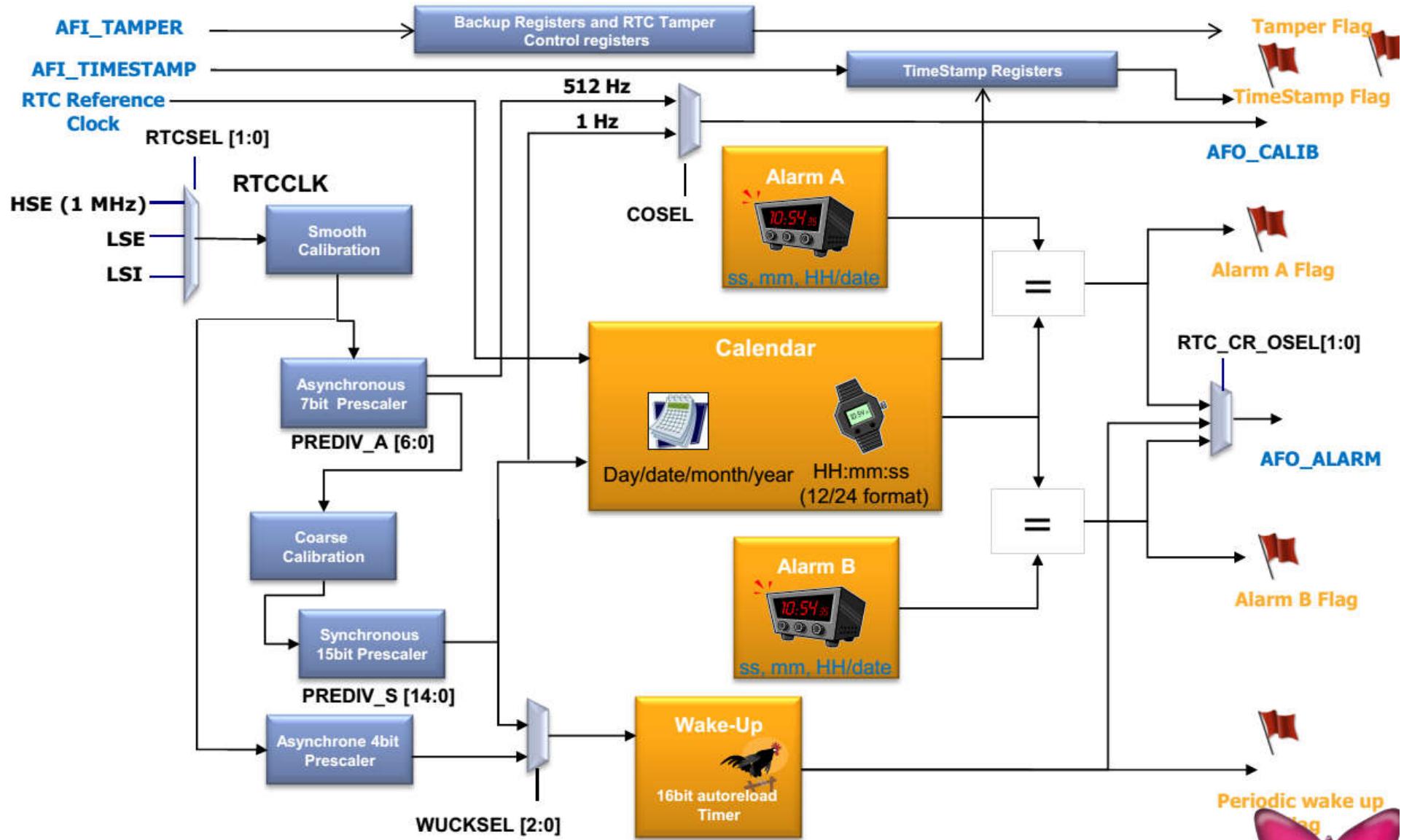
# USART: Fontes de Interrupção

Interrupt event	Event flag	Enable control bit
Transmit Data Register Empty	TXE	TXEIE
CTS Changes	CTS	CTSIE
Transmission Complete	TC	TCIE
Received Data Ready	RXNE	RXNEIE
Overrun Error	ORE	RXNEIE
Idle Line Detected	IDLE	IDLEIE
Parity Error	PE	PEIE
Break Flag	LBD	LBDIE
Noise error	NF	EIE
Framing error	FE	EIE

## Real-time clock e calendário

- 3 fontes de clock: LSE (oscilador dedicado: 32,768 kHz), LSI ou HSE
- Calendário: subseg, seg, min, hora (formato de 12 ou 24 horas), dia, mês, ano
- Possui 2 alarmes programáveis: A e B
- Gera interrupções:
  - Alarme A
  - Alarme B
  - *Wakeup* periódico
  - ***Timestamp detection:*** armazena data em registradores quando de detecção de evento.
  - ***Tamper detection:*** armazena data em registradores *time-stamp* quando da detecção de evento de segurança (adulteração do sistema).

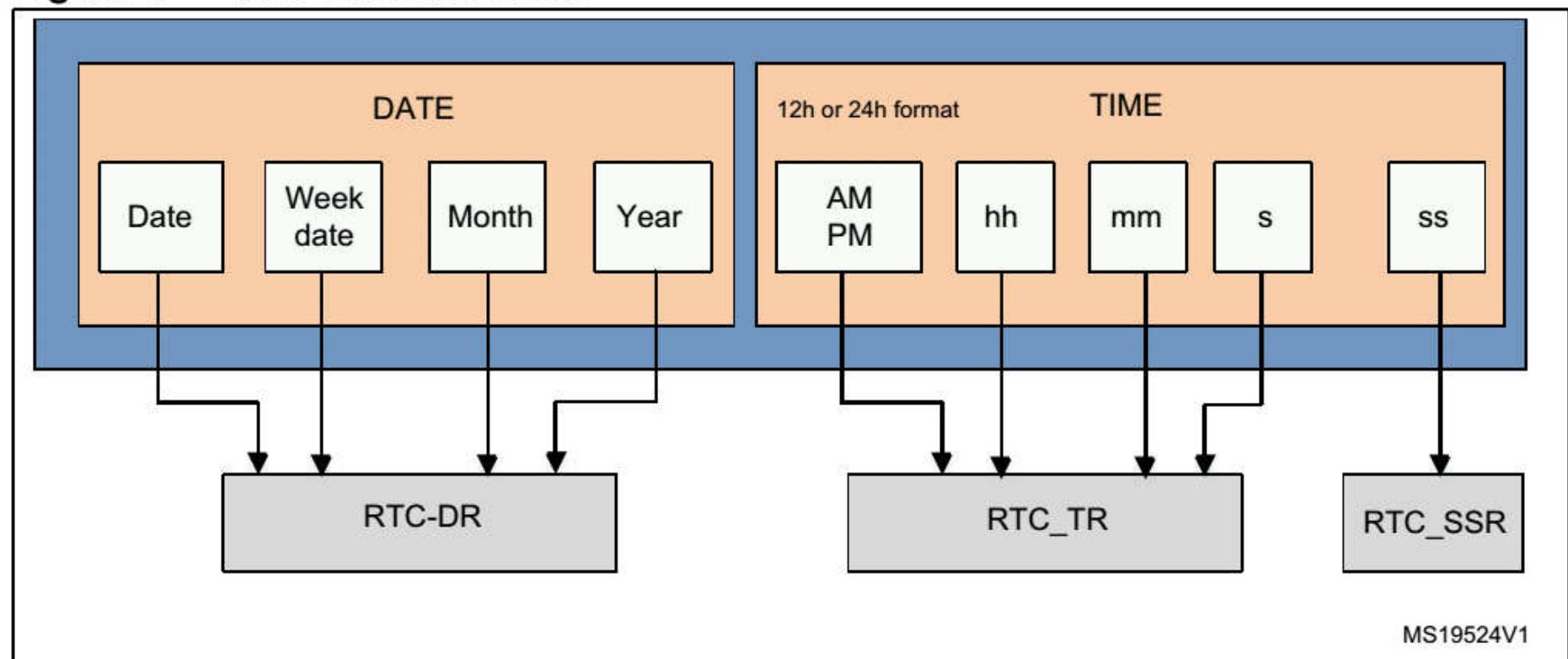
# Real-time clock e calendário



GPIOC\_13 (RTC\_AF1) pode ser usado, como (AF input - *Alternate Function*): AFI\_TAMPER, AFI\_TIMESTAMP

# Real-time clock e calendário

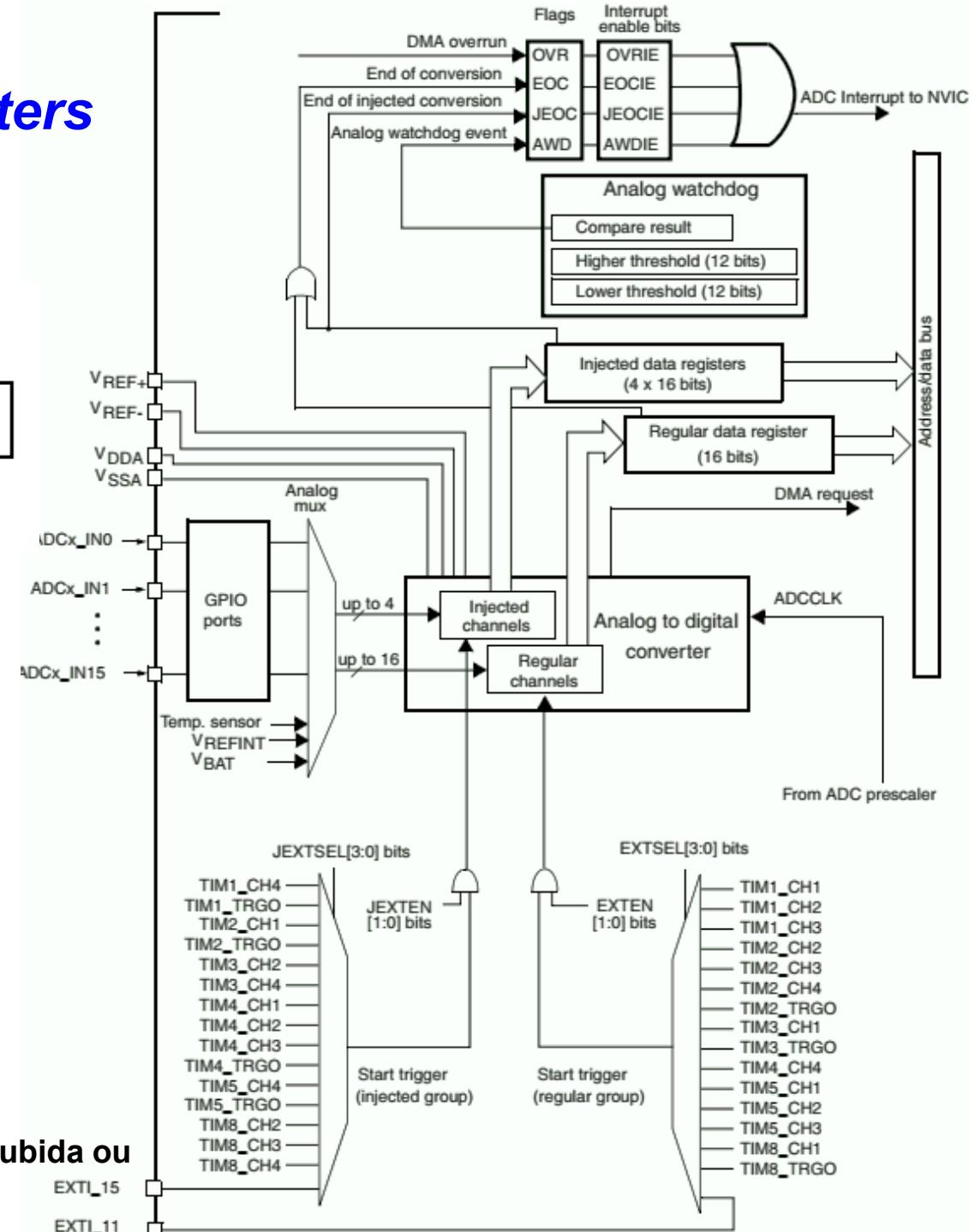
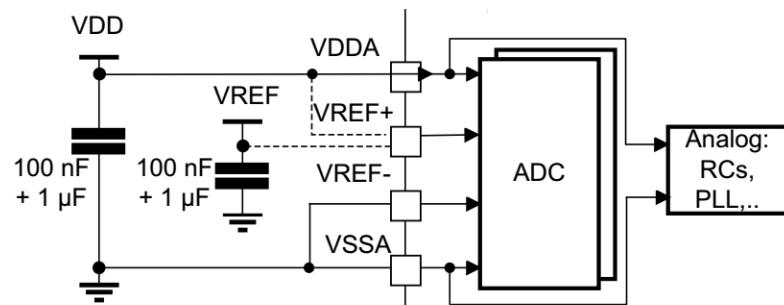
Figure 1. RTC calendar fields



# ADCs

## Analog-to-digital converters

STM32F429



- Trigger externo por borda de descida, subida ou ambos

# ADC - Analog-to-digital converter

ADC1 Configuration

Parameter Settings User Constants NVIC Settings DMA Settings GPIO Settings

Configure the below parameters :

Search : Search (Ctrl+F)

ADCs\_Common\_Settings

Mode	Independent mode
------	------------------

ADC\_Settings

Clock Prescaler	PCLK2 divided by 4
Resolution	12 bits (15 ADC Clock cycles)
Data Alignment	Right alignment
Scan Conversion Mode	Disabled
Continuous Conversion Mode	Disabled
Discontinuous Conversion Mode	Disabled
DMA Continuous Requests	Disabled
End Of Conversion Selection	EOC flag at the end of single channel conversion

ADC-Regular\_ConversionMode

Number Of Conversion	1
External Trigger Conversion Edge	None
Rank	1

ADC\_Injected\_ConversionMode

Number Of Conversions	0
-----------------------	---

WatchDog

Enable Analog WatchDog Mode	<input type="checkbox"/>
-----------------------------	--------------------------

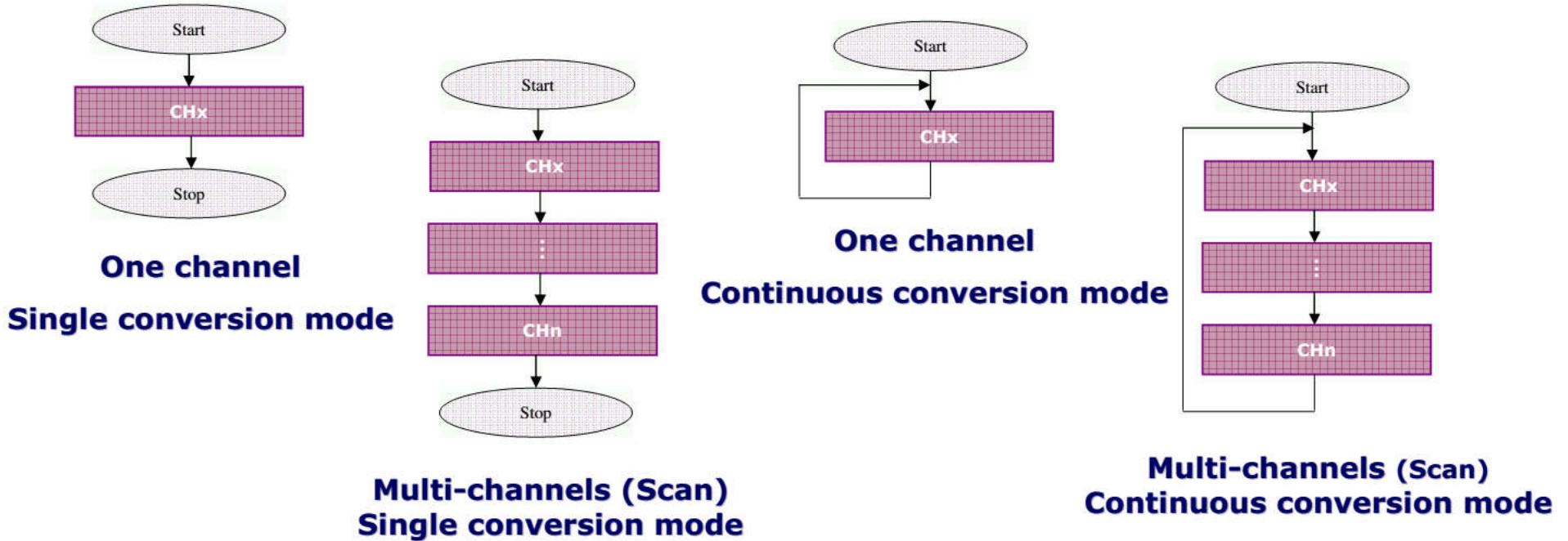
## **ADCs - Analog-to-digital converters**

- **3 × 12-bit ADCs de 2,4 MSPS – Até 19 canais:** ADC1 (mestre), ADC2 e ADC3 (escravos)
- **Alimentação ( $V_{DDA}$ ):** 2,4V a 3,6V.
- **Faixa de tensão para conversão:**  $V_{REF-} \leq V_{IN} \leq V_{REF+}$  ( $1,8 \text{ V} \leq V_{REF+} \leq V_{DDA}$ )
- **Canais dedicados:**
  - ADC1\_IN17: Tensão de referência interno: - VREFINT (1,2V típico)
  - ADC1\_IN18 compartilhado entre (utilizar um ou outro):
    - Sensor de temperatura
    - Monitoramento de bateria:  $V_{BAT}$
- Opções de triggers internos (hardware e software) e externos para conversões regulares e injected (Ver figura acima)

## ADCs – Modos de Conversão

- ❑ Possível organizar conversões em 2 grupos: Regular e Injetado.
- **Grupo:** Sequência de conversões realizadas em quaisquer canais e ordem. Exemplo de sequência de conversão: ADC\_IN3, ADC\_IN8, ADC\_IN2, ADC\_IN2, ADC\_IN0, ADC\_IN2, ADC\_IN2, ADC\_IN15.
- **Regular group:**
  - Comporta até 16 conversões.
  - Realizar leitura do ADC ao final da conversão em cada canal.
- **Injected group :**
  - Comporta até 4 conversões.
- **Injected conversion:**
  - Interrompe conversão regular para amostrar até 4 canais.
  - Conversão regular retomada após *Injected Conversion*
  - Resultados armazenados em registradores dedicados (*Injected data registers*)
  - **Triggered injection:** trigger ou software (SWSTART='1')
  - **Auto injection:** *injected group* é automaticamente convertido após conversão regular

# ADCs – Modos de Conversão : *Single & Continuous*



- **Single**: converte único canal ou múltiplos canais (*Scan*) pré-programados (*Regular Group*) uma única vez.
- **Continuous** (apenas para *Regular Group*): converte único canal ou múltiplos canais (*Scan*) pré-programados (*Regular Group*) repetidamente..

## ADCs – Modos de Conversão : *Discontinuous*

- **Para grupo regular:** Utilizado para converter curta sequência de **n** canais ( $n \leq 8$ ) do grupo que pode ter até 16 canais.
  - Após *trigger*, inicia-se sequência de conversão de **n** canais do grupo regular de forma sucessiva e circular.
  - Exemplo:
    - $n = 3$ ; grupo regular = 0, 1, 2, 4, 5, 8, 9, 11, 12
    - trigger 1: converte-se sequencialmente os canais 0, 1, 2. Gera-se EOC (*end of conversion int.*) a cada conversão.
    - trigger 2: converte-se sequencialmente os canais 4, 5, 8. Gera-se EOC a cada conversão.
    - trigger 3: converte-se sequencialmente os canais 9, 11, 12. Gera-se EOC a cada conversão.
    - trigger 4 : converte-se sequencialmente os canais 0, 1, 2. Gera-se EOC a cada conversão.



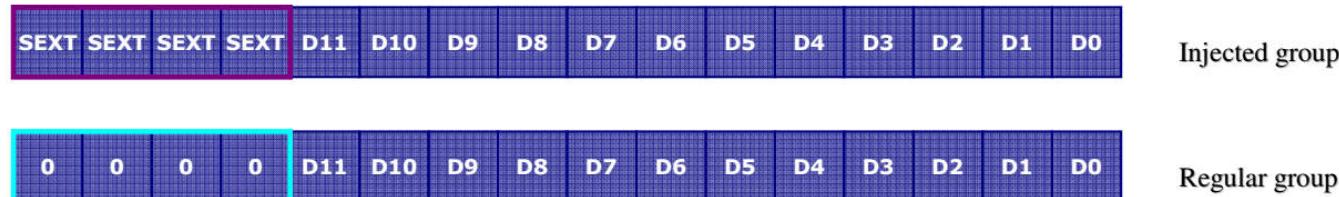
## ADCs – Modos de Conversão : *Discontinuous*

- **Para Grupo *injected*** : Utilizado para sequência de **n** canais (**n** ≤ 3) do grupo que pode ter até 4 canais.
  - Após trigger, inicia-se sequência de conversão de **n** canais do grupo injetado de forma sucessiva e circular.
  - Exemplo:
    - n = 1; grupo *injected* = 1, 2, 3
    - trigger 1: converte-se canal 1.
    - trigger 2: converte-se canal 2.
    - trigger 3: converte-se canal 3. Gera-se JEOC.
    - trigger 4 : converte-se canal 1.

# ADCs – Alinhamento da palavra

- Alinhamento programável à esquerda ou à direita em registradores de 16 bits
- Resultado da conversão do *injected group* pode ser subtraído de valor de *offset* definido pelo usuário (ADC\_JOFRx registers), tal que o valor possa ser negativo.

## Right alignment



## Left alignment

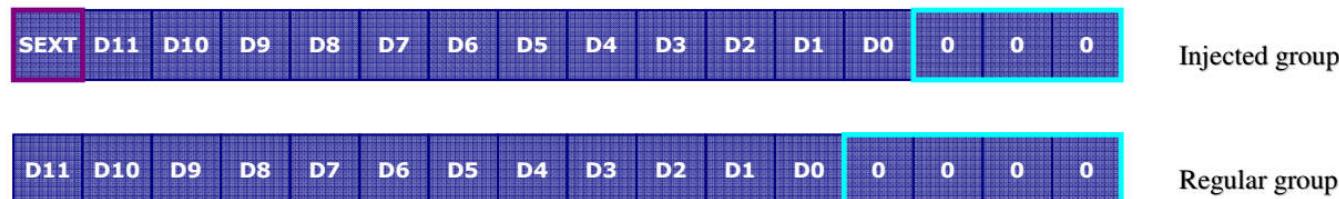


Figure 50. Left alignment of 6-bit data

### Injected group

SEXT	D5	D4	D3	D2	D1	D0	0								
------	------	------	------	------	------	------	------	------	----	----	----	----	----	----	---

### Regular group

0	0	0	0	0	0	0	0	D5	D4	D3	D2	D1	D0	0	0
---	---	---	---	---	---	---	---	----	----	----	----	----	----	---	---

**SEXT:** extended sign value

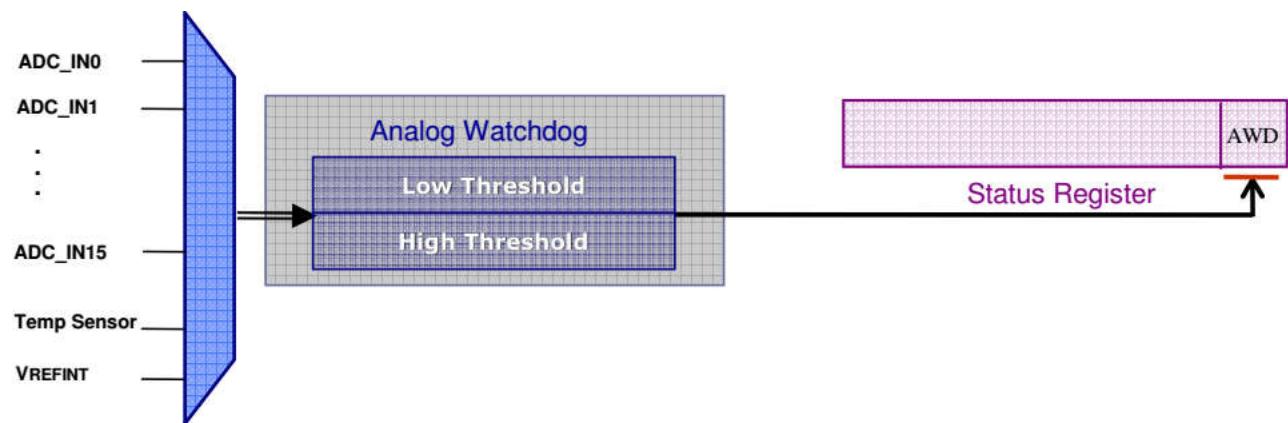
# ADCs - Analog-to-digital converters

## □ Interrupções geradas:

- *End of Conversion* (EOC)
- *End of Injected Conversion* (JEOC)
- *Analog watchdog*
- *Overrun* (amostra anterior é sobrescrita por nova aquisição)

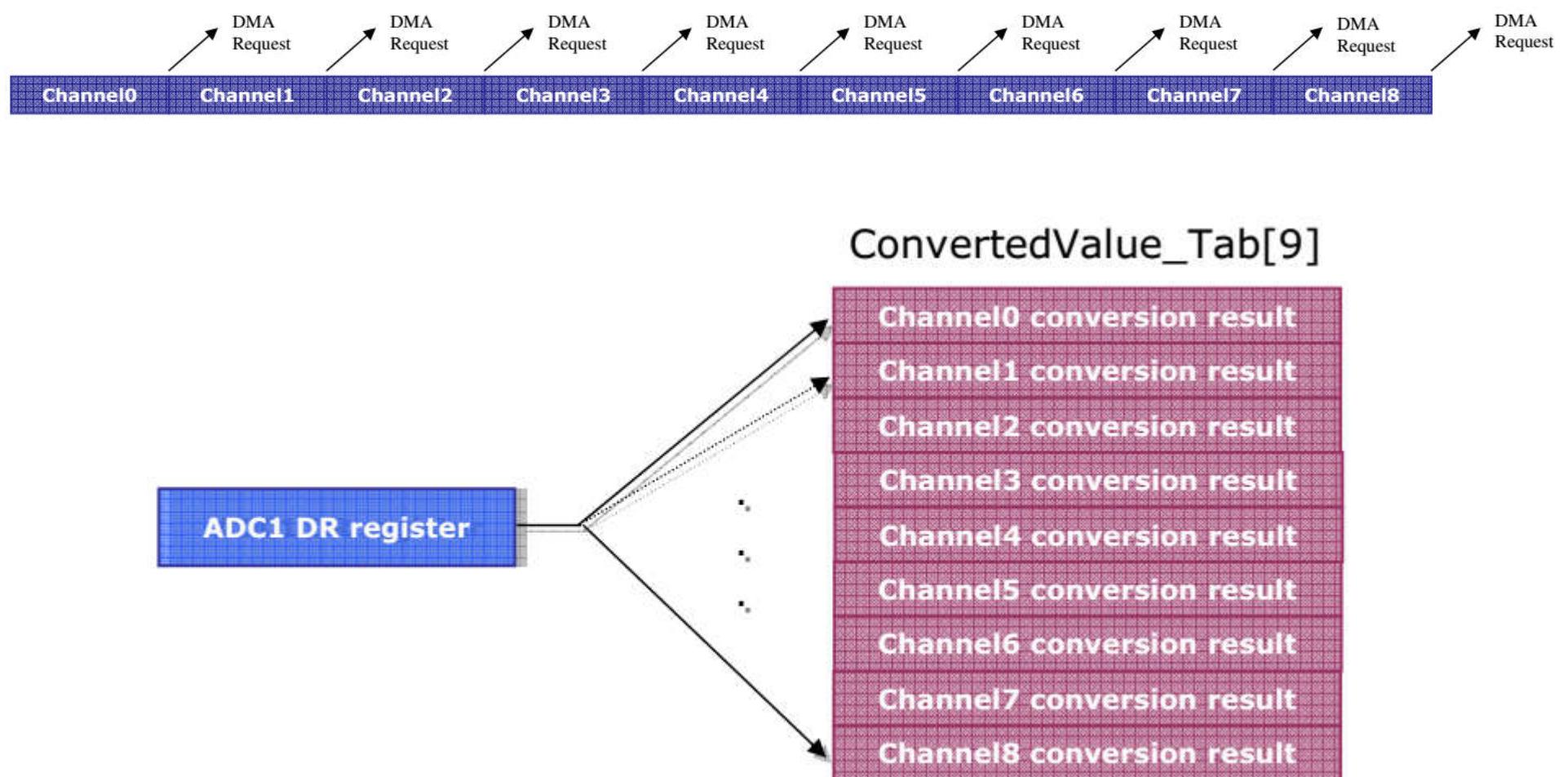
## □ Analog watchdog:

- Gera interrupção se sinal convertido está acima ou abaixo de limiares previamente programados em registradores (ADC\_HTR e ADC\_LTR).
- Habilitado para um ou todos os canais.



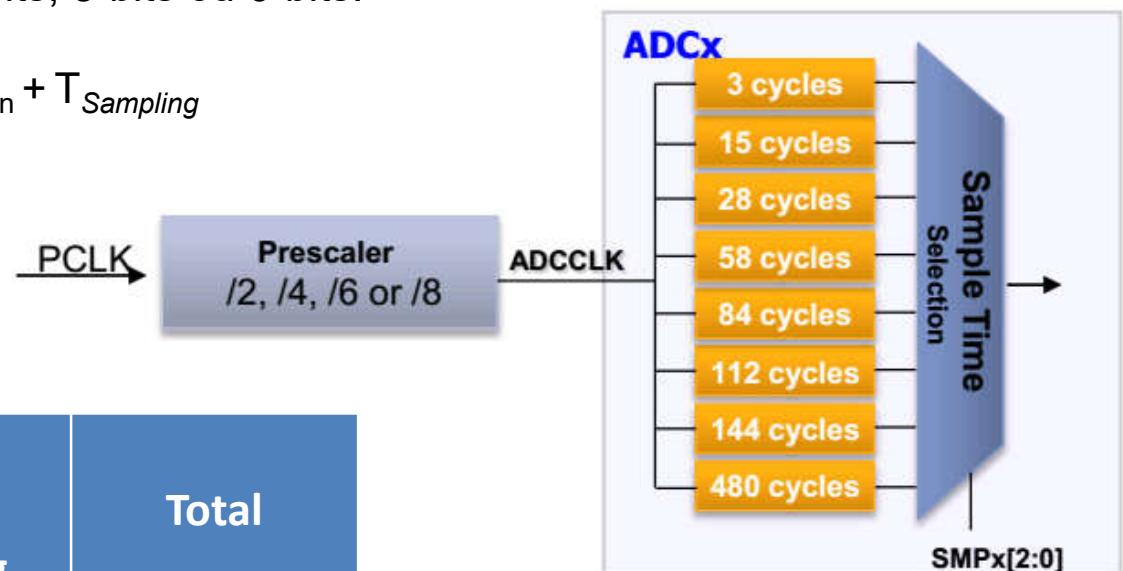
## ADCs - Analog-to-digital converters

- ❑ DMA (transferência de dado de ADC1 para memória): requisitado por *ADC1 End of Conversion* (caso desejado)



## ADCs - Analog-to-digital converters

- Clock do ADC: Até 90MHz - PCLK dividido por prescaler (2, 4, 6 e 8)
- Tempo de amostragem ( $T_{Sampling}$ ) programável para cada canal:
  - 3, 15, 28, 56, 84, 112, 144, 480 ciclos de clock
- Resolução configurável: 12-bits, 10-bits, 8-bits ou 6-bits.
- Tempo Total de Conversão =  $T_{conversion} + T_{Sampling}$



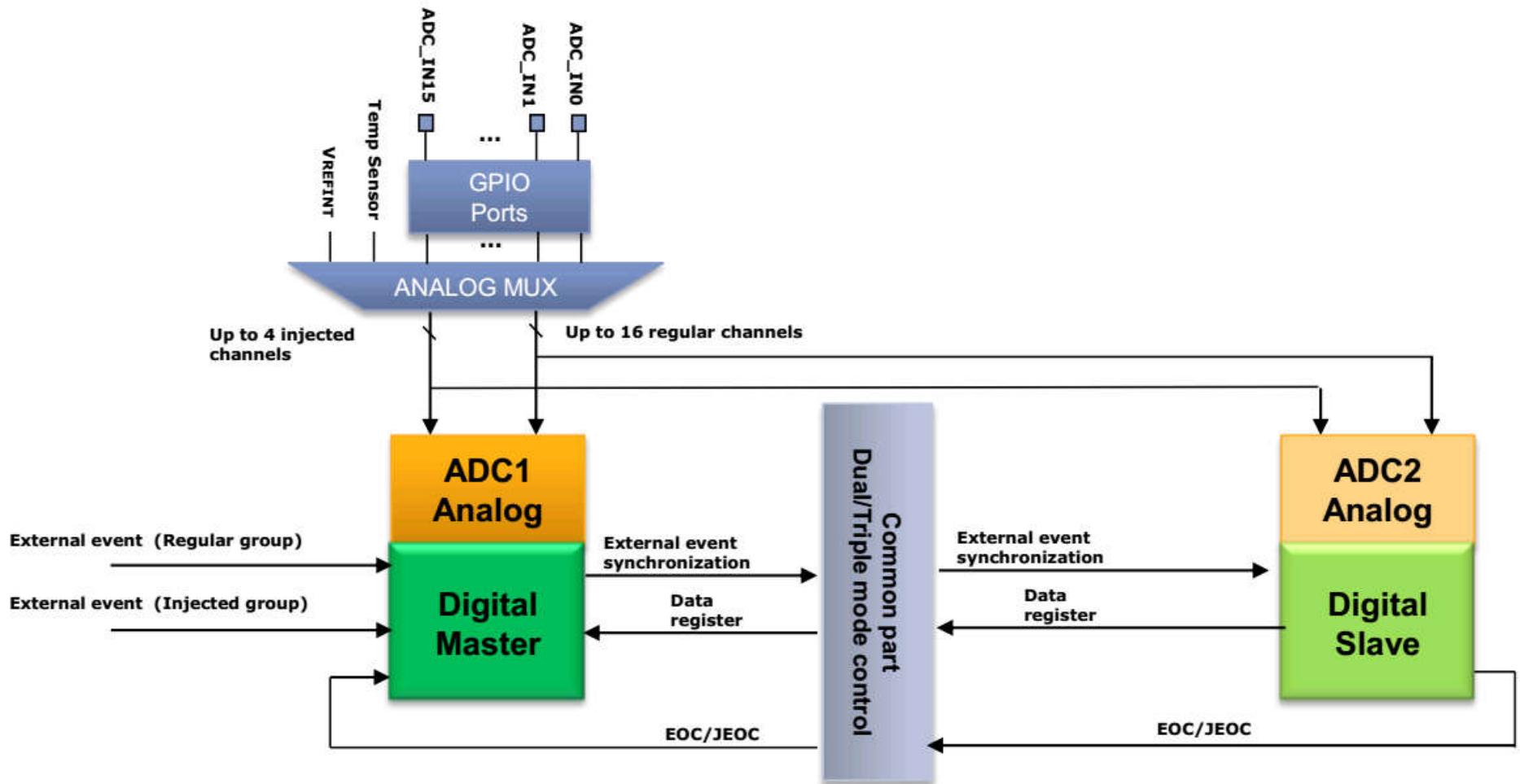
Resolução	ADC Clock Cycles	Clock sampling	Total
12	15	3	18
10	13	3	16
8	11	3	14
6	9	3	12

## Multi ADC Mode

- Dispositivos com 2 ou mais ADCs possibilitam multi ADC modes
  - O início da conversão é disparada de forma simultânea ou alternada nos ADCs do dispositivo
  - OBS: Se trigger se dá por evento externo, utilize apenas trigger do mestre e desabilite trigger dos escravos para prevenir conversões espúrias.
  - 4 modos possíveis:
    - *Injected simultaneous mode*
    - *Regular simultaneous mode*
    - *Interleaved mode*
    - *Alternate trigger mode*
  - Possível utilizar os modos de forma combinada:
    - *Injected simultaneous mode + Regular simultaneous mode*
    - *Regular simultaneous mode + Alternate trigger mode*

## Multi ADC Mode – Dual Mode

- ADC1: *master*; ADC2: *slave*; ADC3: independente.
- Conversão do ADC2 ocorre de forma simultânea (ou alternada, dependendo da programação) à conversão do ADC1 mestre

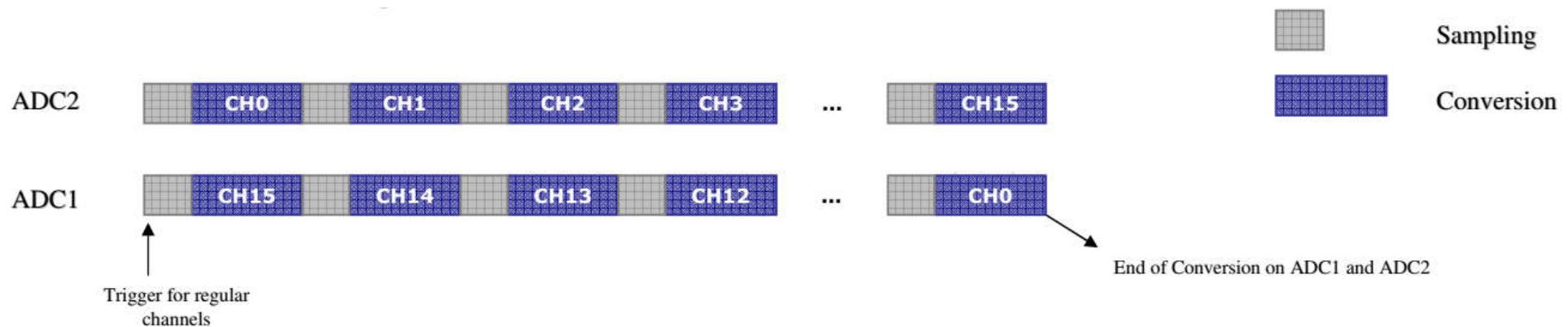


# Multi ADC Mode – Dual Mode

- ❑ Exemplo: ADC1: *master*; ADC2: *slave*.
- ❑ Conversão do ADC2 ocorre de forma simultânea (ou alternada, dependendo da programação) à conversão do ADC1 mestre

## Simultaneous mode

(Pode ser para *injected* (até 4 canais) ou *regular group*)



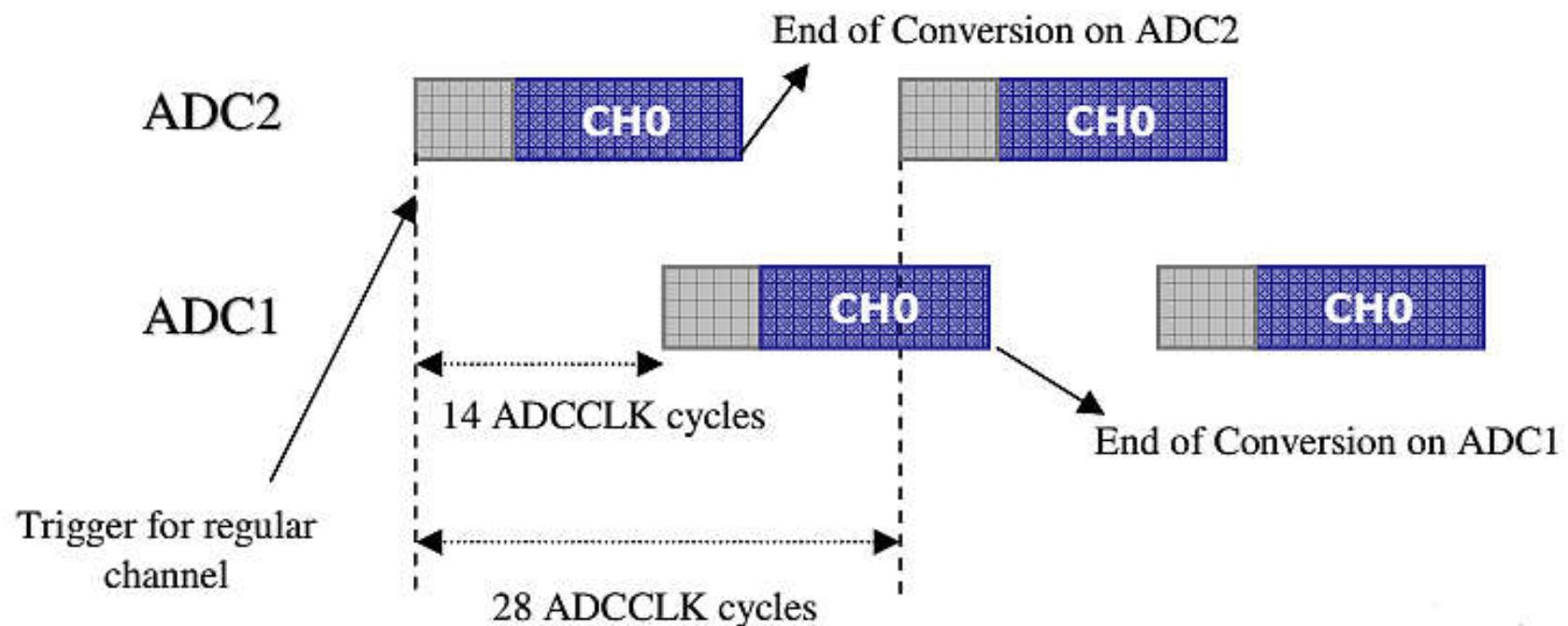
*Note: Do not sample the same channel at the same time on each ADC*

- Utiliza-se DMA para leituras das conversões de grupo regular quando conteúdo de registrador de 32 bits (ADC\_CDR = [ADC2 - ADC1]) deve ser lido.

# Multi ADC Mode – Dual Mode

## *Interleaved mode*

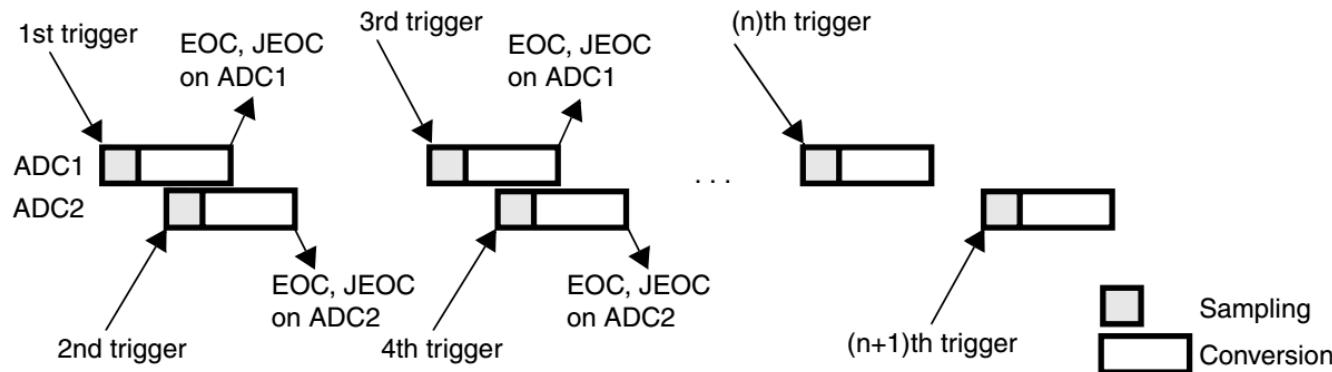
- Apenas para *regular group*
- Usualmente apenas para um canal (aumentar taxa de amostragem)
- Conversões em registrador de 32 bits ( $\text{ADC\_CDR} = [\text{ADC2} - \text{ADC1}]$ )



# Multi ADC Mode – Dual Mode

## Alternate trigger mode

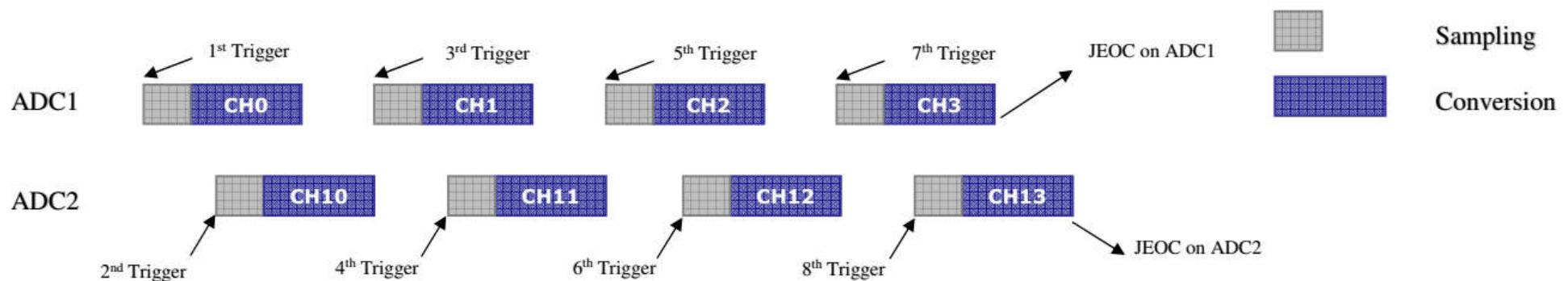
- Apenas para *injected group* (distintos para ADC1 e 2)
- Conversões em registradores de cada ADC
  - trigger 1: converte-se todos os canais do grupo que pertencem ao ADC1
  - trigger 2: converte-se todos os canais do grupo que pertencem ao ADC2
  - trigger 3: converte-se todos os canais do grupo que pertencem ao ADC1...



# Multi ADC Mode – Dual Mode

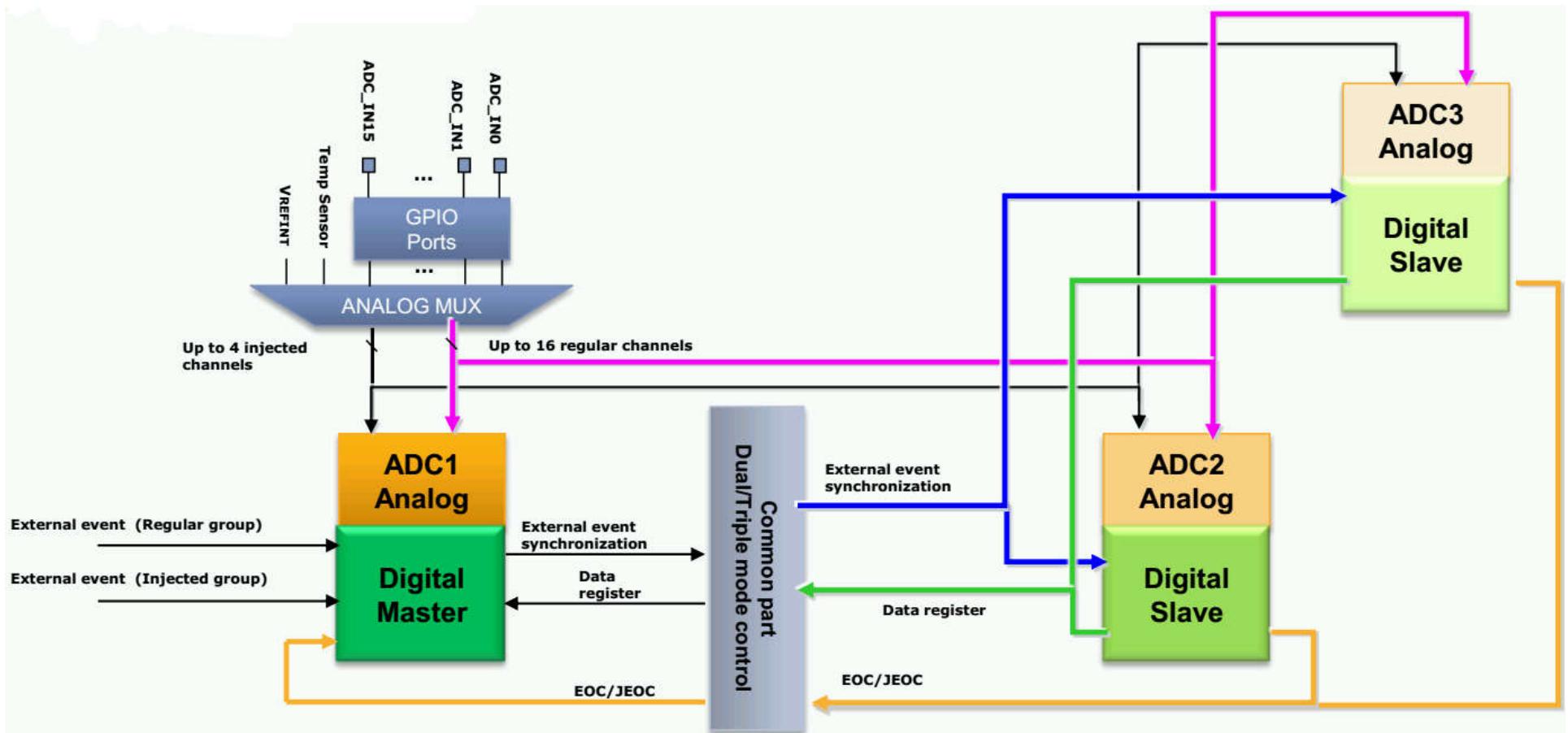
*Alternate trigger mode com discontinuous mode* habilitado para ADC1 e ADC2

- Apenas para *injected group* (exemplo de grupos distintos para ADC1 e ADC2)
- Conversões em registradores de cada ADC
  - trigger 1: converte-se primeiro canal do grupo que pertencem ao ADC1
  - trigger 2: converte-se primeiro canal do grupo que pertencem ao ADC2
  - trigger 3: converte-se segundo canal do grupo que pertencem ao ADC1...



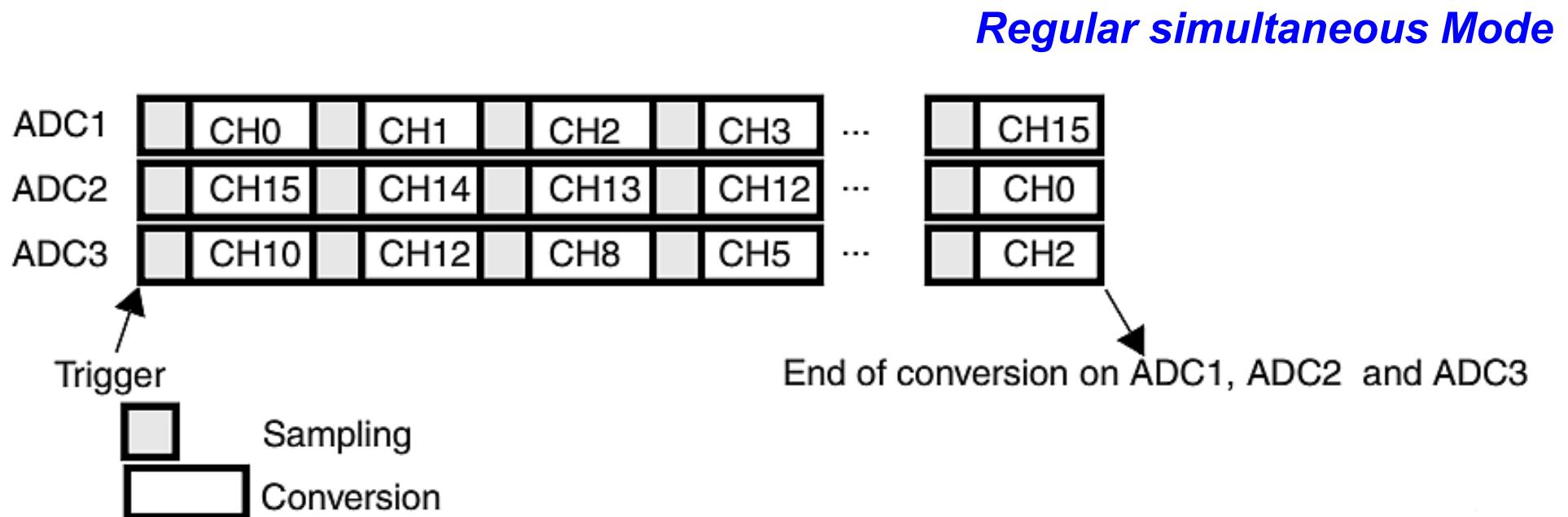
# Multi ADC Mode – *Triple Mode*

- ADC1: *master*; ADC2 e ADC3: *slaves*
- Conversões de ADC2 e ADC3 ocorrem de forma simultânea (ou alternada, dependendo da programação) à conversão do ADC1 mestre



## Multi ADC Mode – *Triple Mode*

- ❑ ADC1: *master*; ADC2 e ADC3: *slaves*. Pode ser *injected* (até 4 canais) ou regular
- ❑ Conversões de ADC2 e ADC3 ocorrem de forma simultânea (ou alternada, dependendo da programação) à conversão do ADC1 mestre



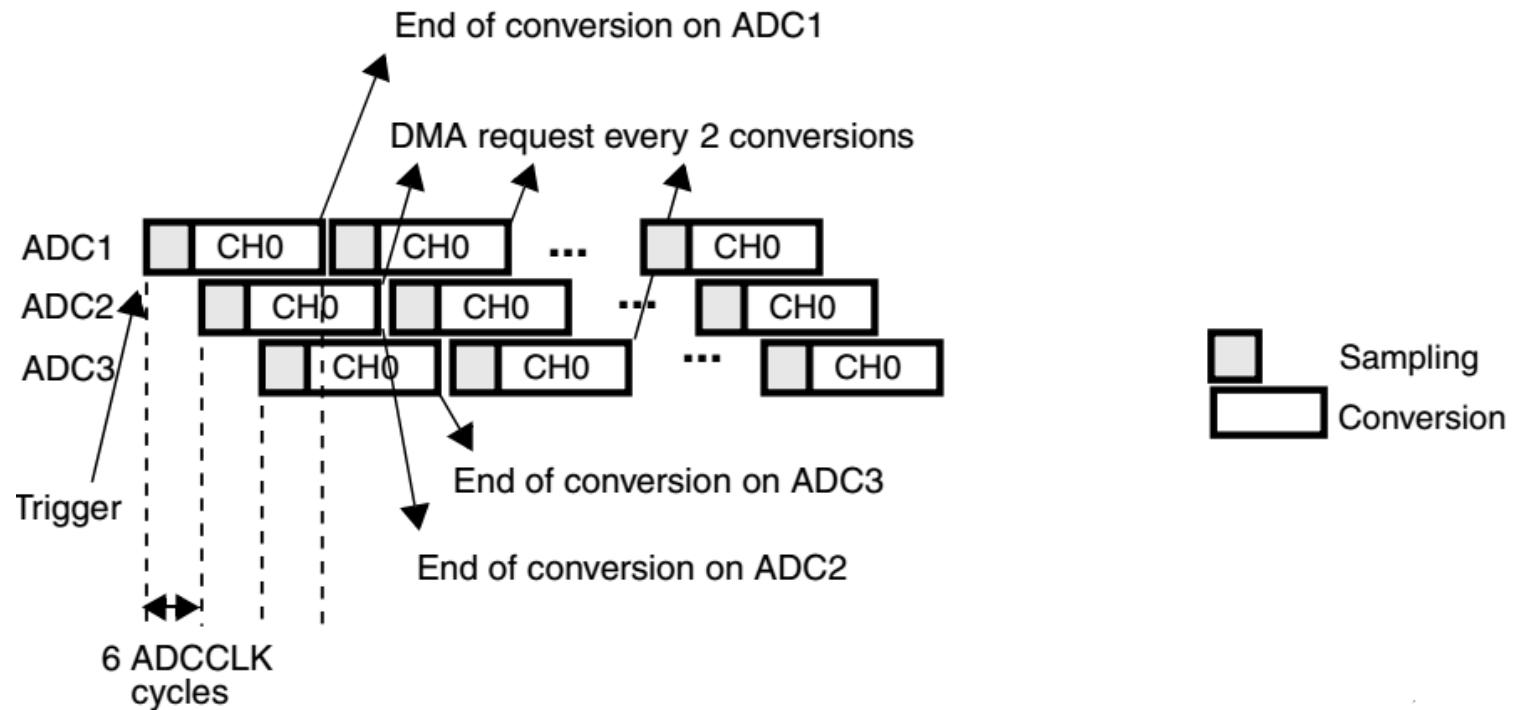
- Utiliza-se DMA para leituras das conversões de grupo regular. São realizadas 3 leituras de registradores de 32 bits (ADC\_CDRs).

# Multi ADC Mode – *Triple Mode*

□ ADC1: *master*; ADC2 e ADC3: *slaves*.

## *Interleaved mode*

- Apenas para *regular group*
- Usualmente apenas para um canal (aumentar taxa de amostragem)
- Conversões em registrador de 32 bits (ADC\_CDR)



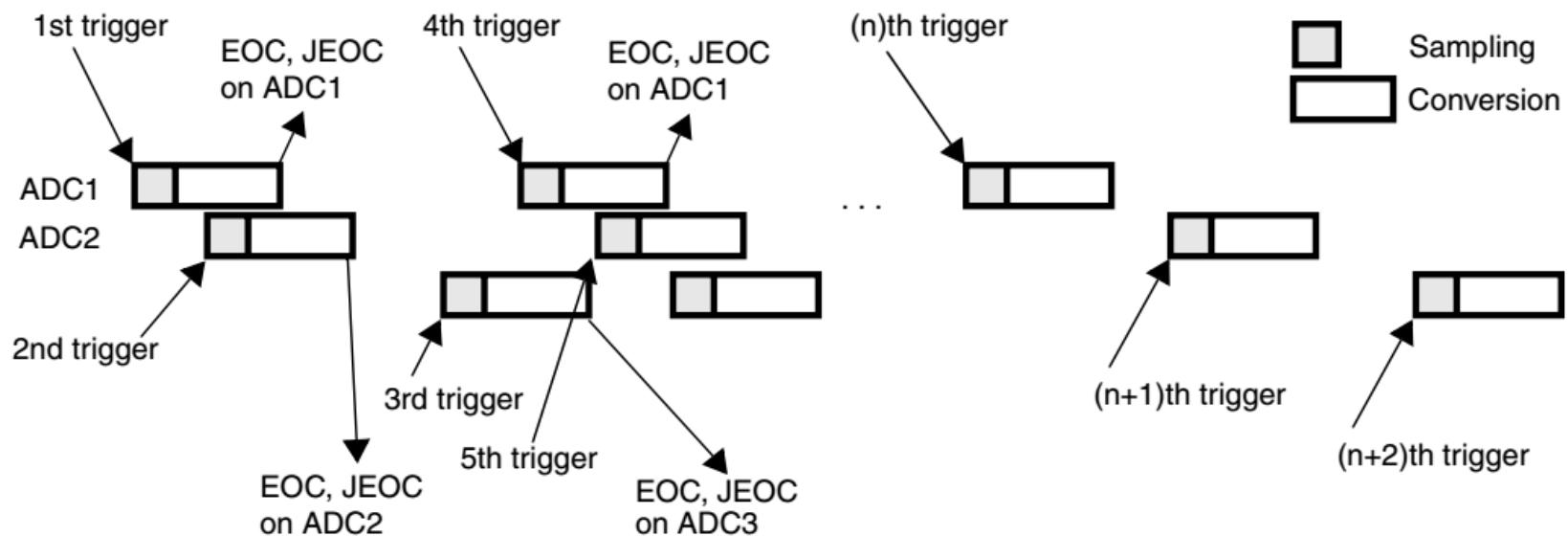
# Multi ADC Mode – *Triple Mode*

□ ADC1: *master*; ADC2 e ADC3: *slaves*.

## *Alternate trigger mode*

- Apenas para *injected group*
- Conversões em registradores de cada ADC

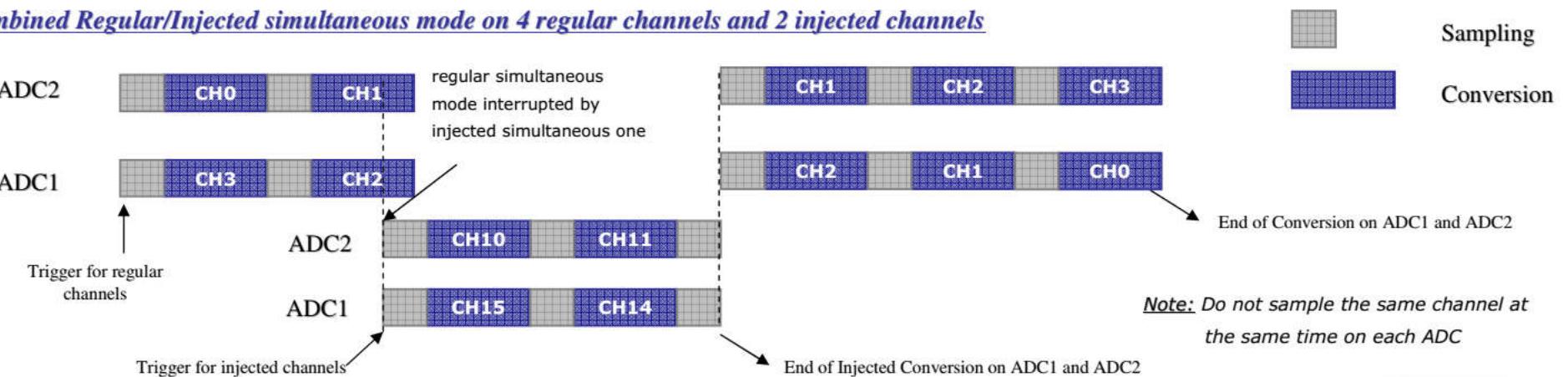
- trigger 1: converte-se todos os canais do grupo que pertencem ao ADC1
- trigger 2: converte-se todos os canais do grupo que pertencem ao ADC2
- trigger 3: converte-se todos os canais do grupo que pertencem ao ADC3
- trigger 4 : converte-se todos os canais do grupo que pertencem ao ADC1...



## Multi ADC Mode – Injected + Regular simultaneous Mode

- ADC1: *master*; ADC2: *slave*.
- Modo *regular simultaneous* interrompido por *Injected simultaneous*
- *End of Injected Conversion* flag gerado ao final das conversões do *Injected group*.
- *End of Conversion* flag gerado ao final das conversões do *regular group*
- Conversões do *Injected group* armazenados em *Injected data registers* de cada ADC
- Conversões do *regular group* armazenados em registrador de 32 bits (ADC\_CDRs)

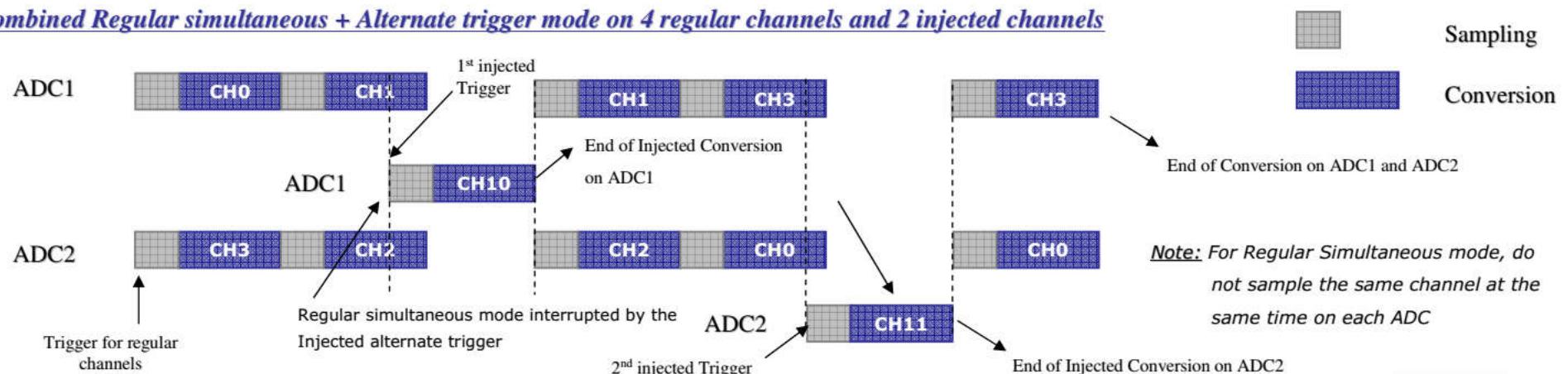
### Combined Regular/Injected simultaneous mode on 4 regular channels and 2 injected channels



## Multi ADC Mode – Regular simultaneous + Alternate trigger Mode

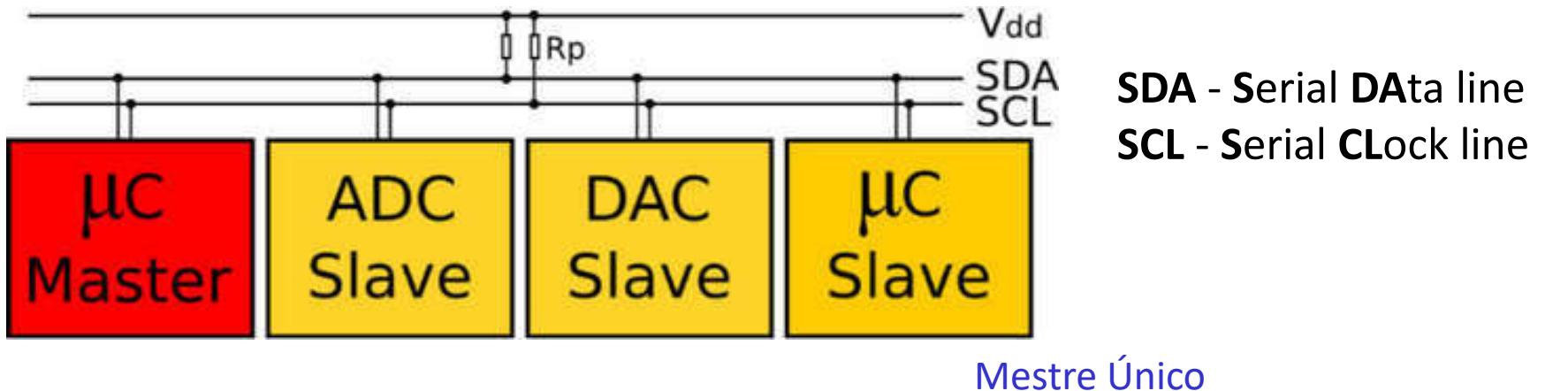
- ❑ ADC1: *master*; ADC2: *slave*.
- ❑ Trigger para conversões *alternate injected mode* interrompe conversões *regular simultaneous*
- ❑ *End of Injected Conversion* flag gerado ao final das conversões do *Injected group*.
- ❑ *End of Conversion* flag gerado ao final das conversões do *regular group*
- ❑ Conversões do *Injected group* armazenados em *Injected data registers* de cada ADC
- ❑ Conversões do *regular group* armazenados em registrador de 32 bits (ADC\_CDRs)

### Combined Regular simultaneous + Alternate trigger mode on 4 regular channels and 2 injected channels

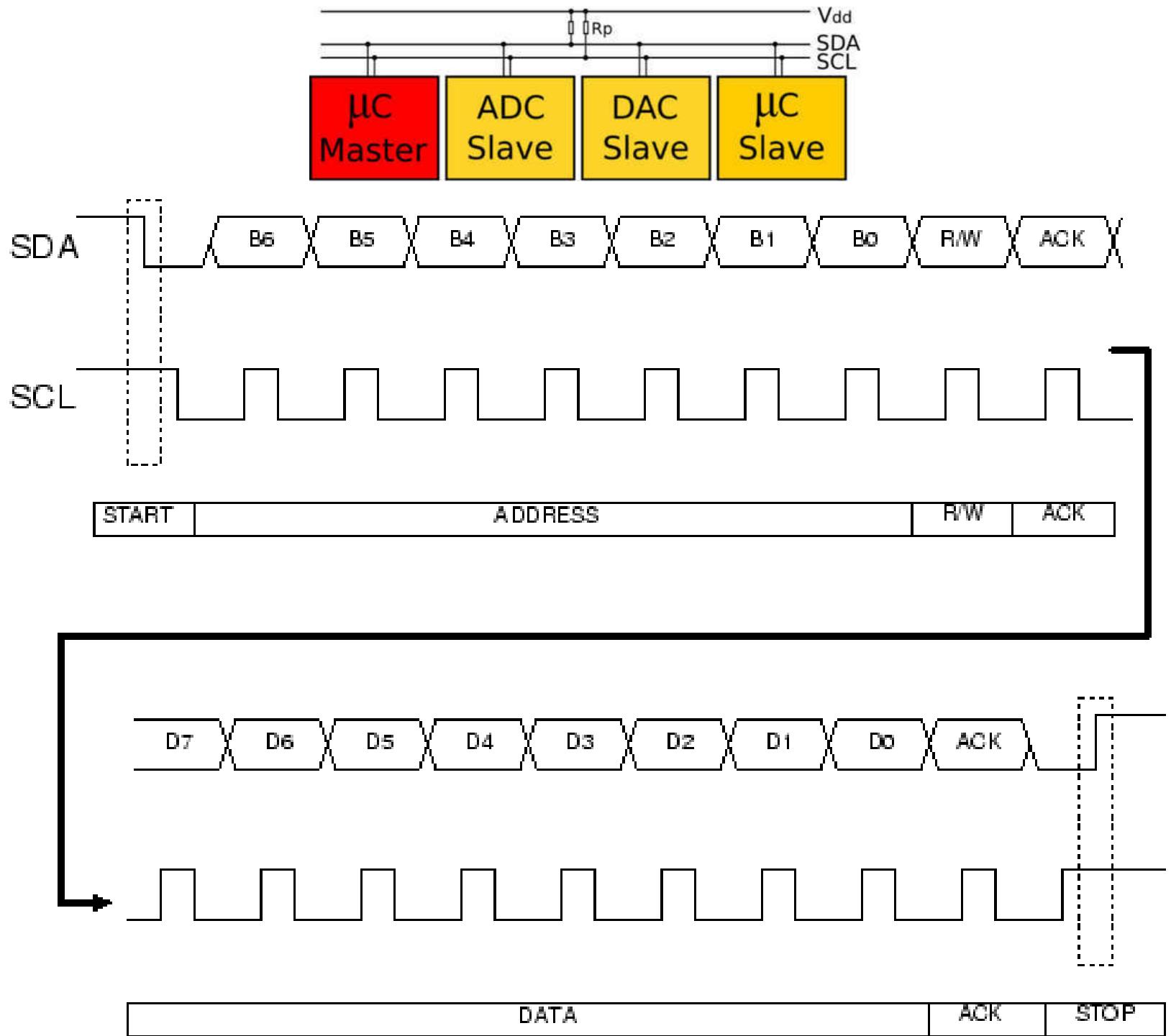


# Transmissão Serial

## I<sup>2</sup>C: *Inter-Integrated Circuit*



- Mestre envia clock e endereço do escravo.
- 7 bits são utilizados para endereçamento; há 16 endereços reservados => máximo de 112 escravos
- CIs tem endereço fixo ou os bits menos significativos do endereço do dispositivo são especificados por pinos
- Velocidade de transmissão de dados => Até 3,4 Mbi/s no modo *High Speed*
- Mestre e escravo podem trocar de papel após stop bit



## I<sup>2</sup>C: Comunicação de dados entre Mestre-Escravo

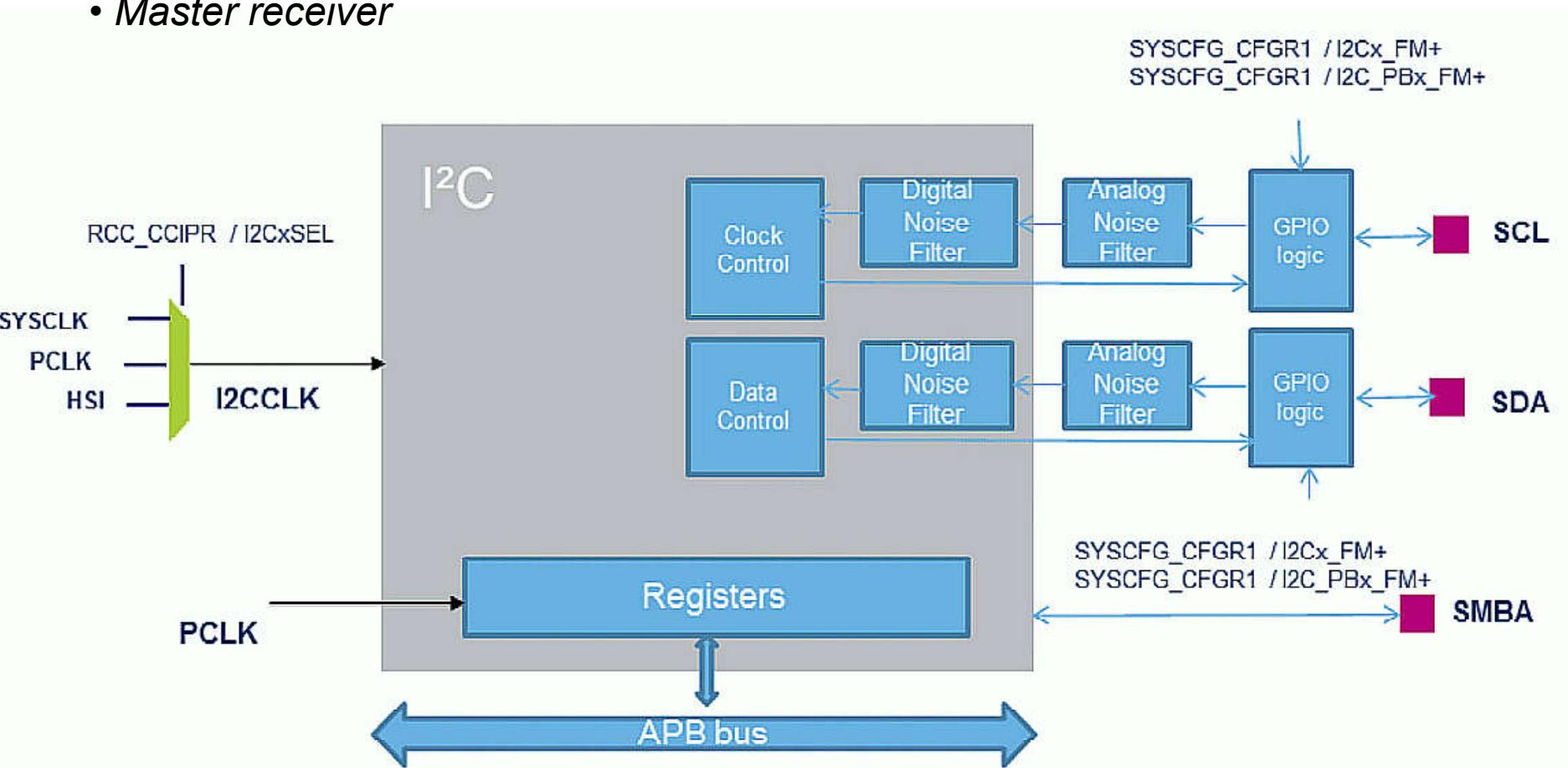
Etapa	Escrita	Leitura	Nro Bits	Campo
1	START bit	START bit	1	START
2	Slave address	Slave address	7	ADDRESS
3	bit 0	bit 1	1	R/W
4	Aguarda bit ‘0’ de “recebido”	Aguarda bit ‘0’ de “recebido”	1	ACK
5	Envia dados	Recebe dados	8	DATA
6	Aguarda bit ‘0’ de recebido	Envia bit ‘0’ de recebido ou vai para passo 7	1	ACK
7	STOP bit	STOP bit	1	STOP

- Os passos 5 e 6 podem ser repetidos tal que múltiplos bytes sejam transmitidos ou recebidos. Ou seja, após passo 6, retorna-se ao passo 5 ou prossegue para o passo 7.

# I<sup>2</sup>C – STM32F4

□ Taxas:

- *Standard Speed (Sm)*: Até 100kHz;
- *Fast Speed (Fm)*: Até 400kHz;
- OBS: Em Fm, padrão I<sup>2</sup>C exige filtro analógico para suprimir spikes de até 50 ns



# I<sup>2</sup>C – STM32F4

## □ Outras características:

- Permite *wakeup* do processador quando de reconhecimento de seu endereço (HSI)
- Filtros programáveis: *Analógico (On/OFF)* e *Digital (nro. de clocks I<sup>2</sup>C)*
- Opera também com endereços de 10-bits
- Compatibilidade com o protocolo *System Management Bus (SMBus)*