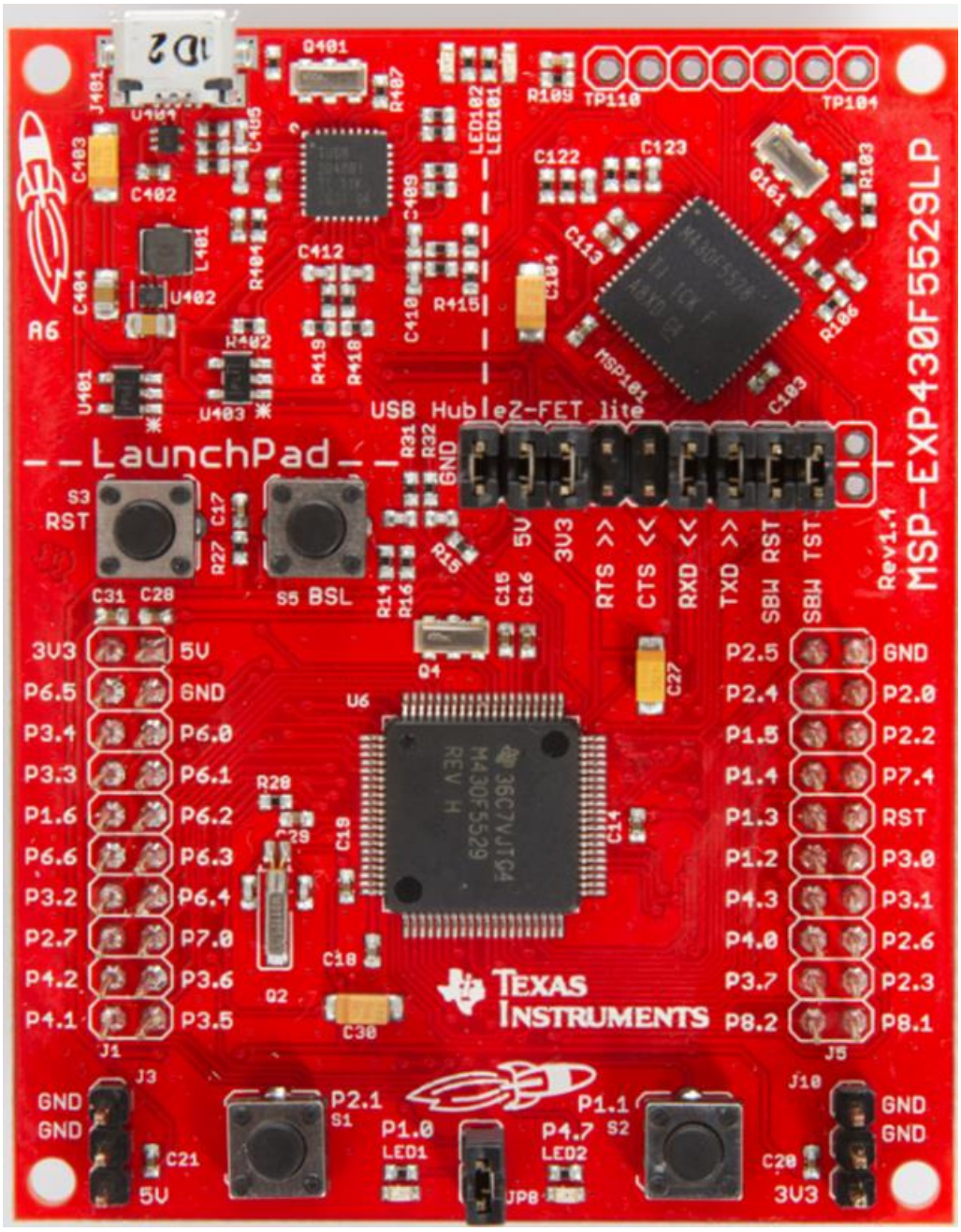
DOCUMENTAÇÃO DO

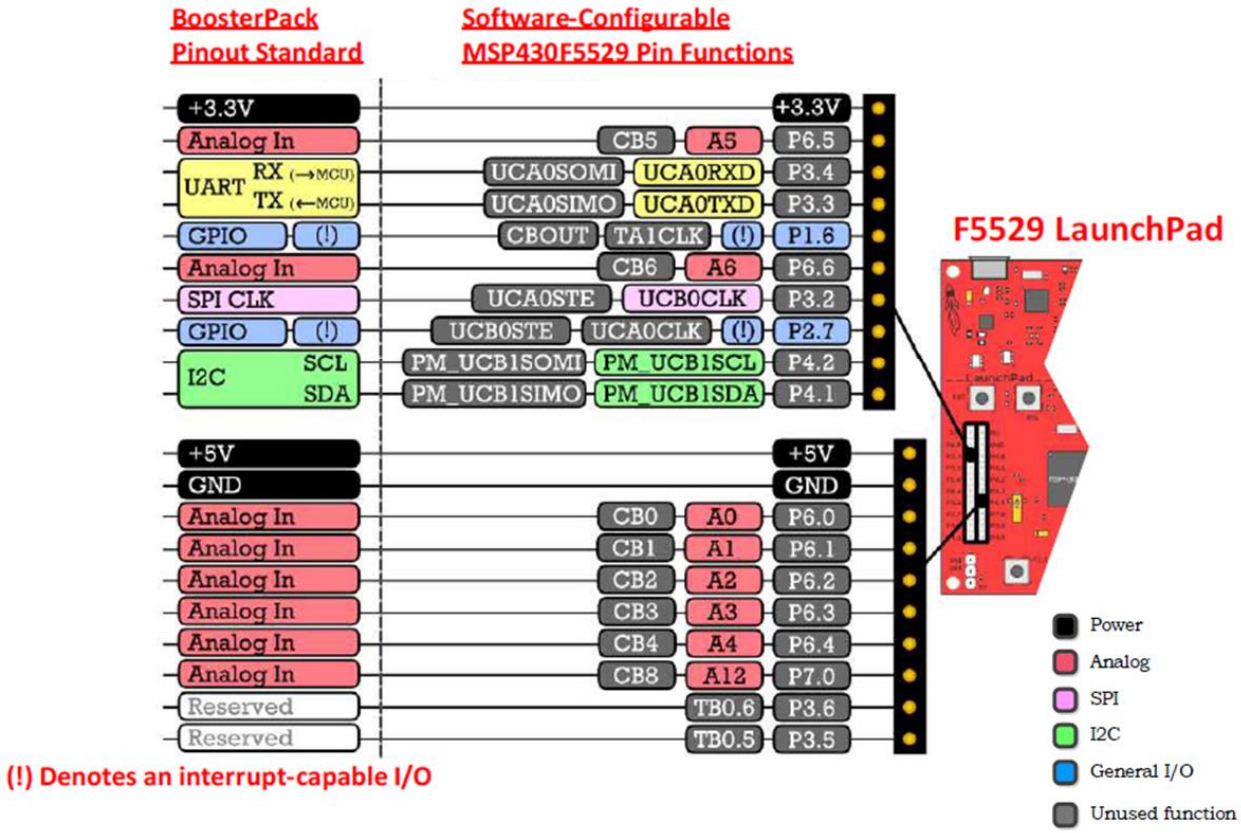
LABORATÓRIO DE SISTEMAS MICROPROCESSADOS



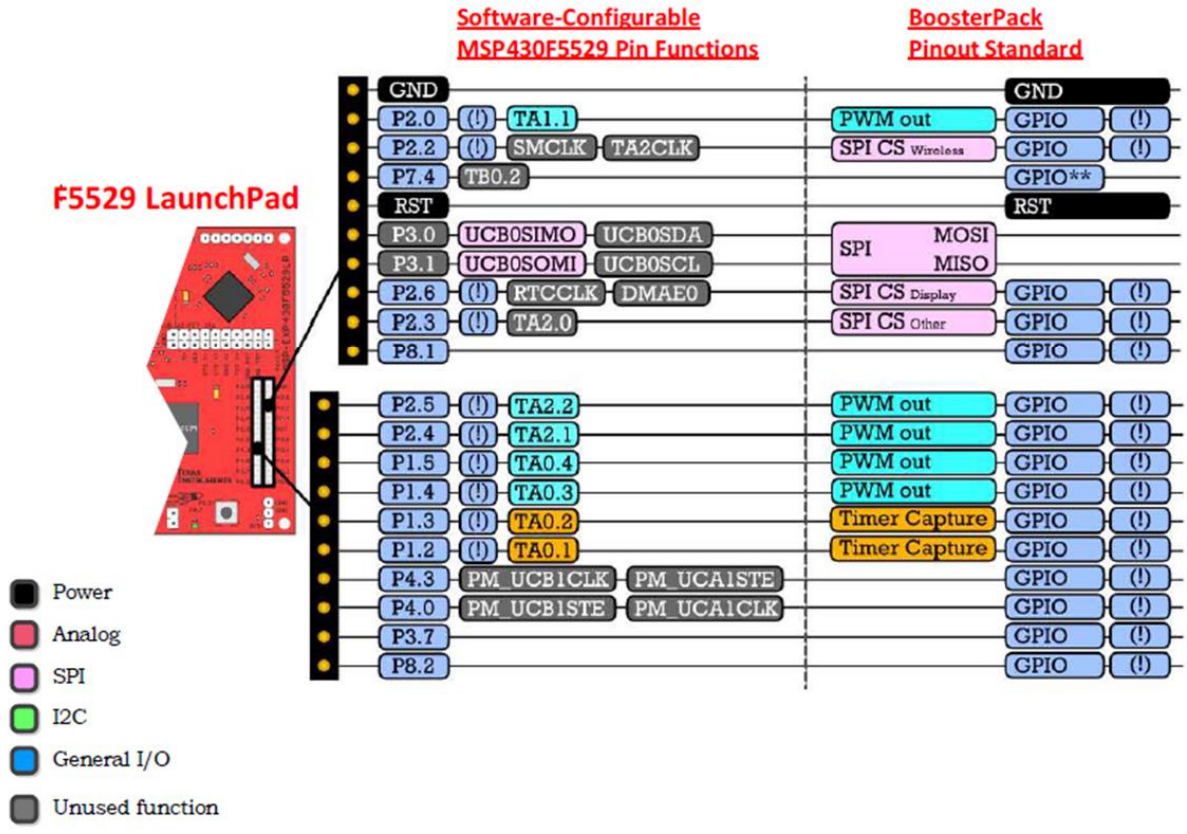
NÃO PODE SER RETIRADO DO LOCAL

Ricardo Zelenovsky

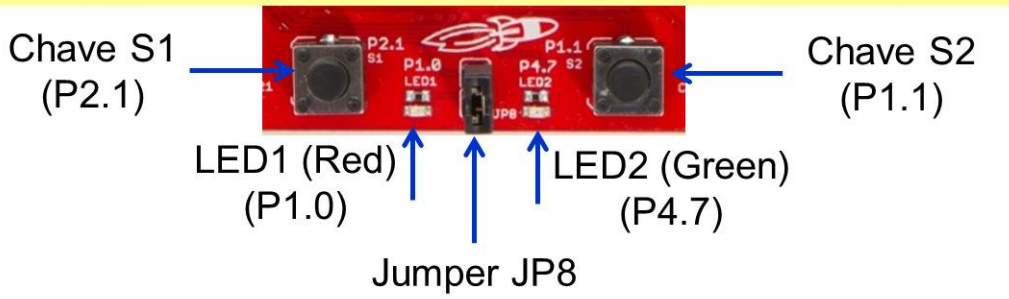
MSP430 – Launch Pad

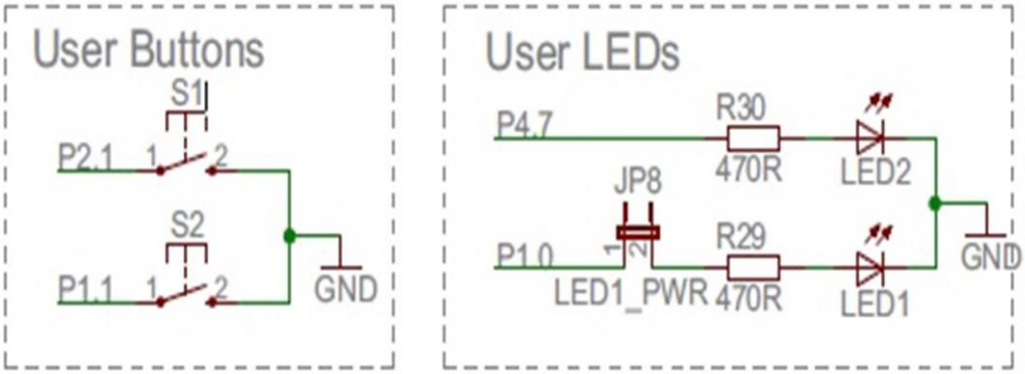


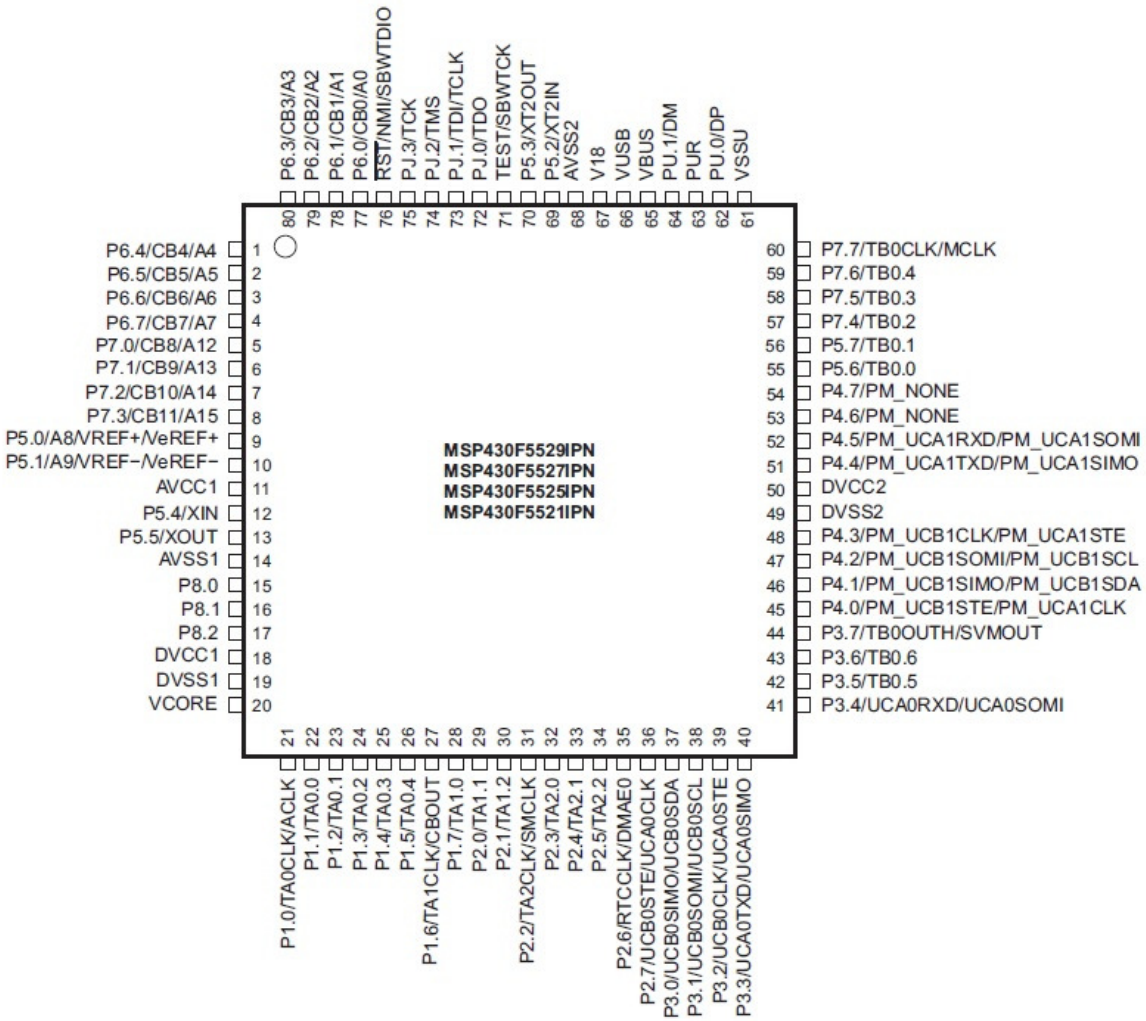
MSP430 – Launch Pad

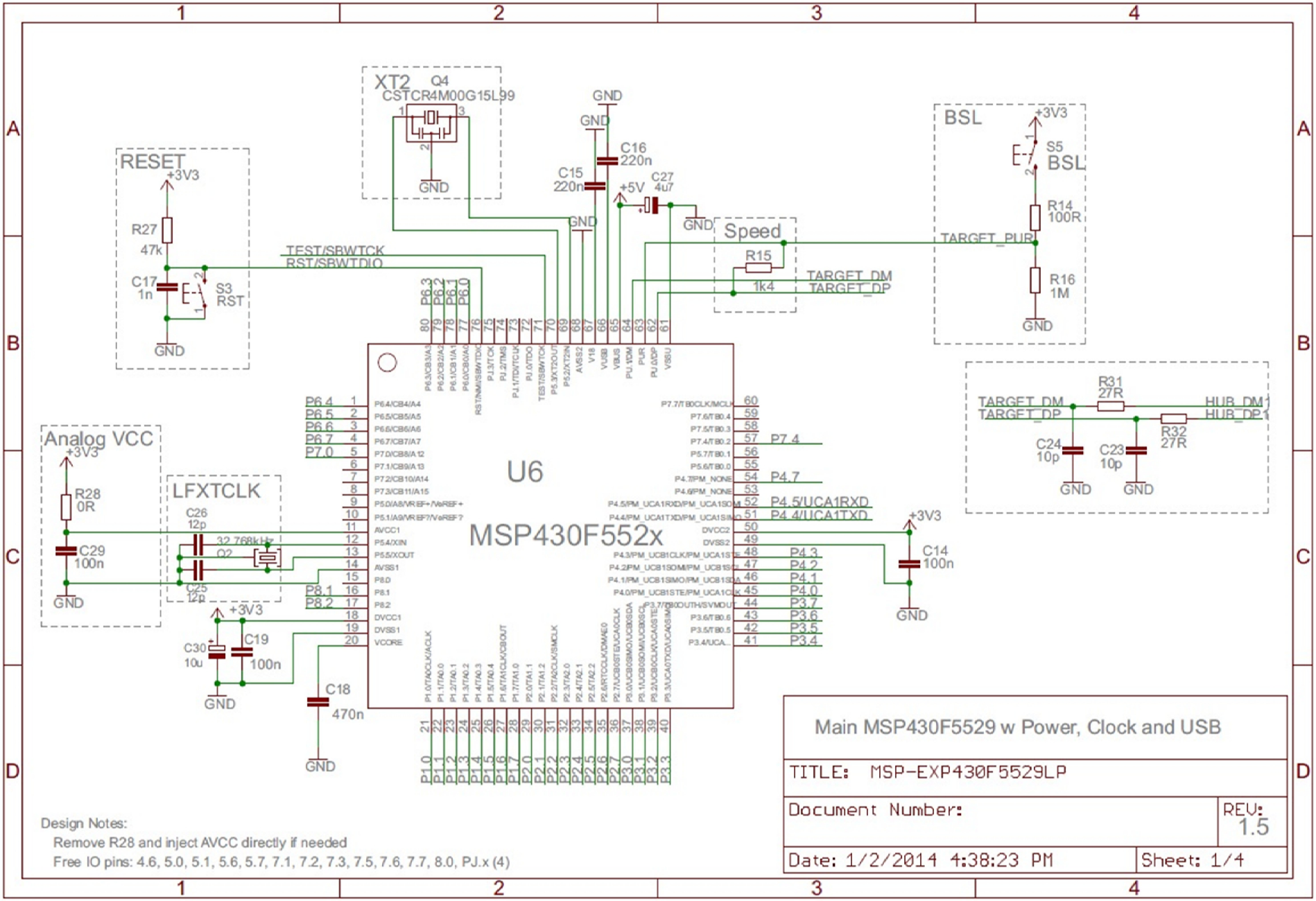


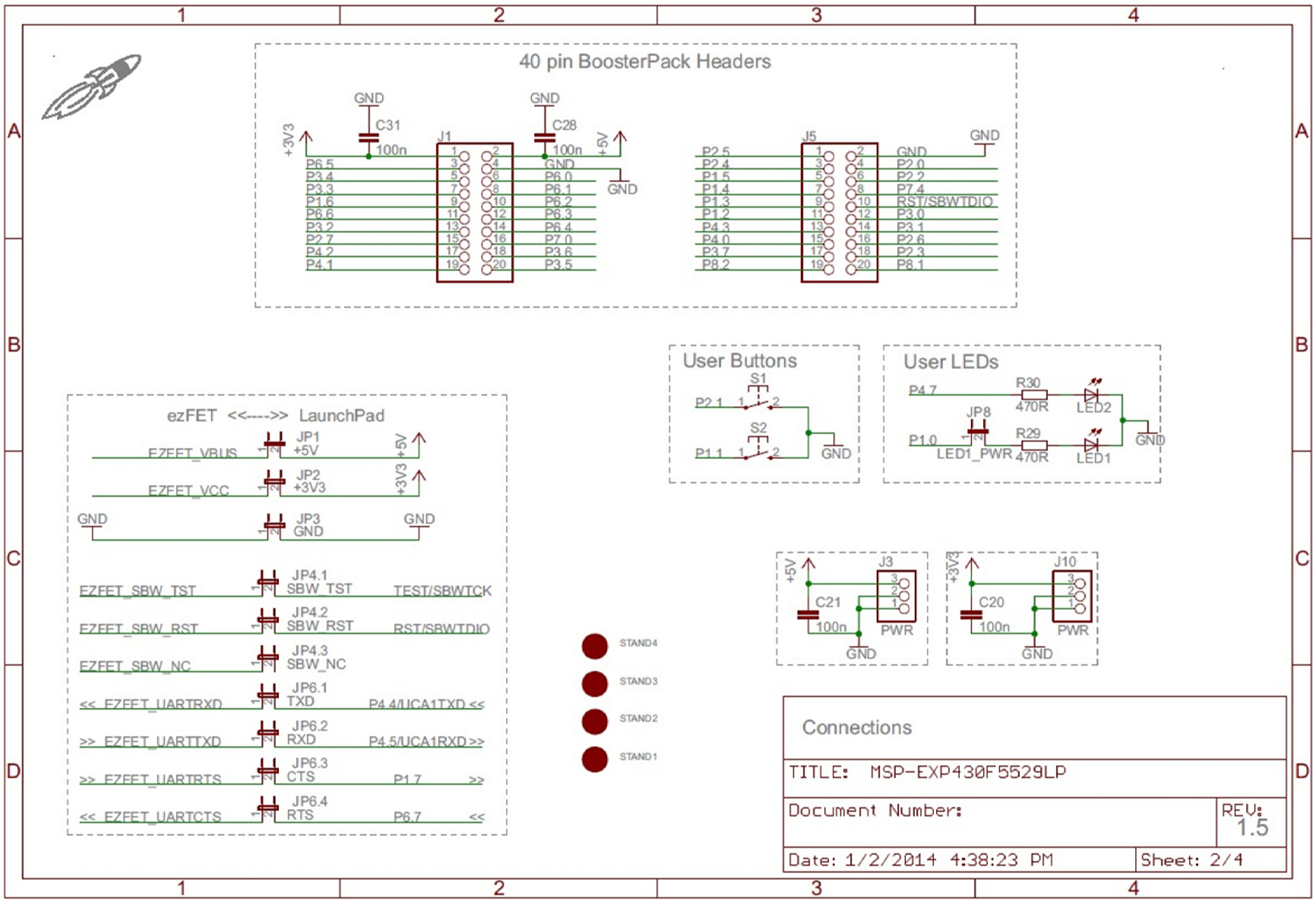
MSP430 – GPIO – LEDs e Chaves

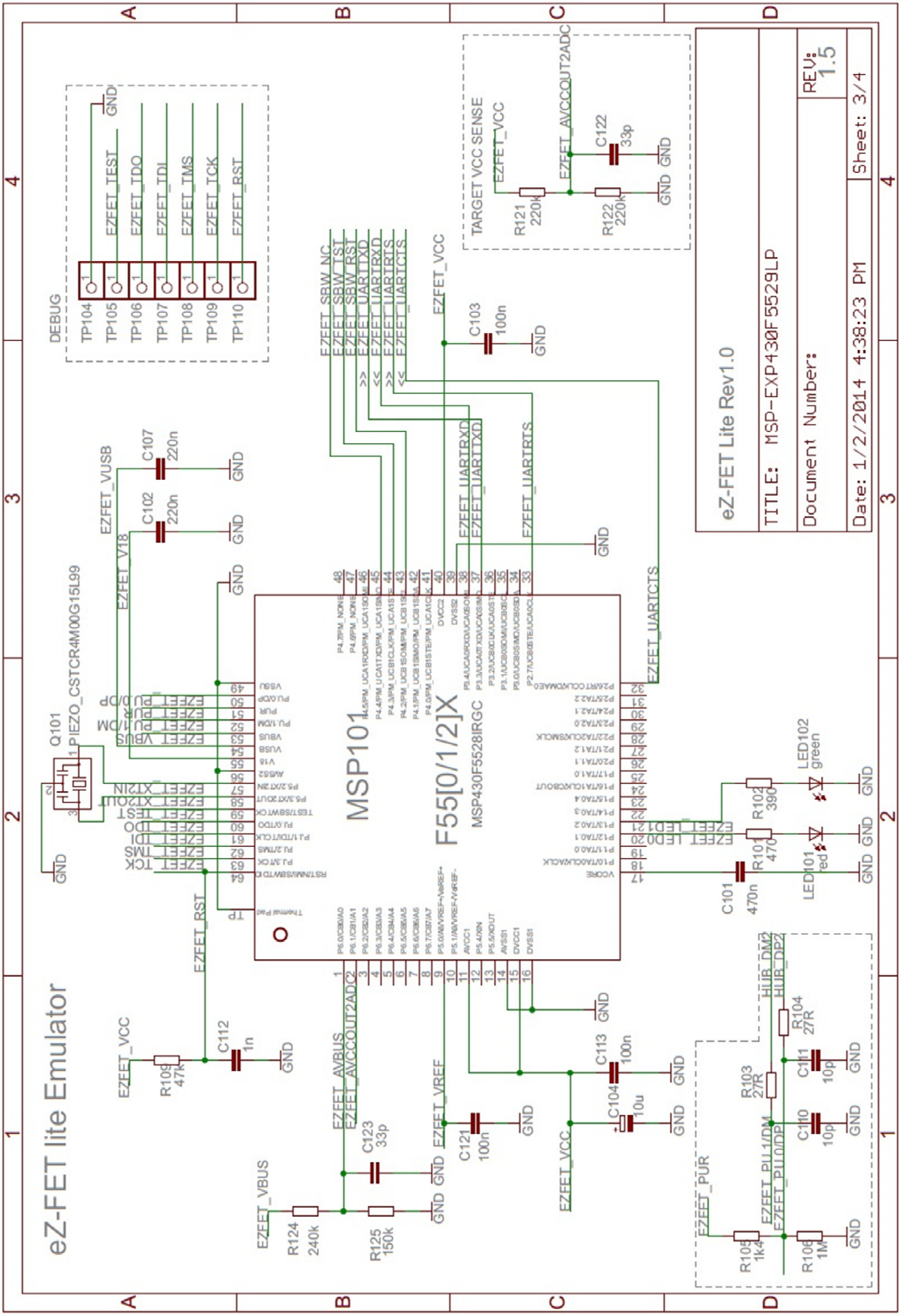


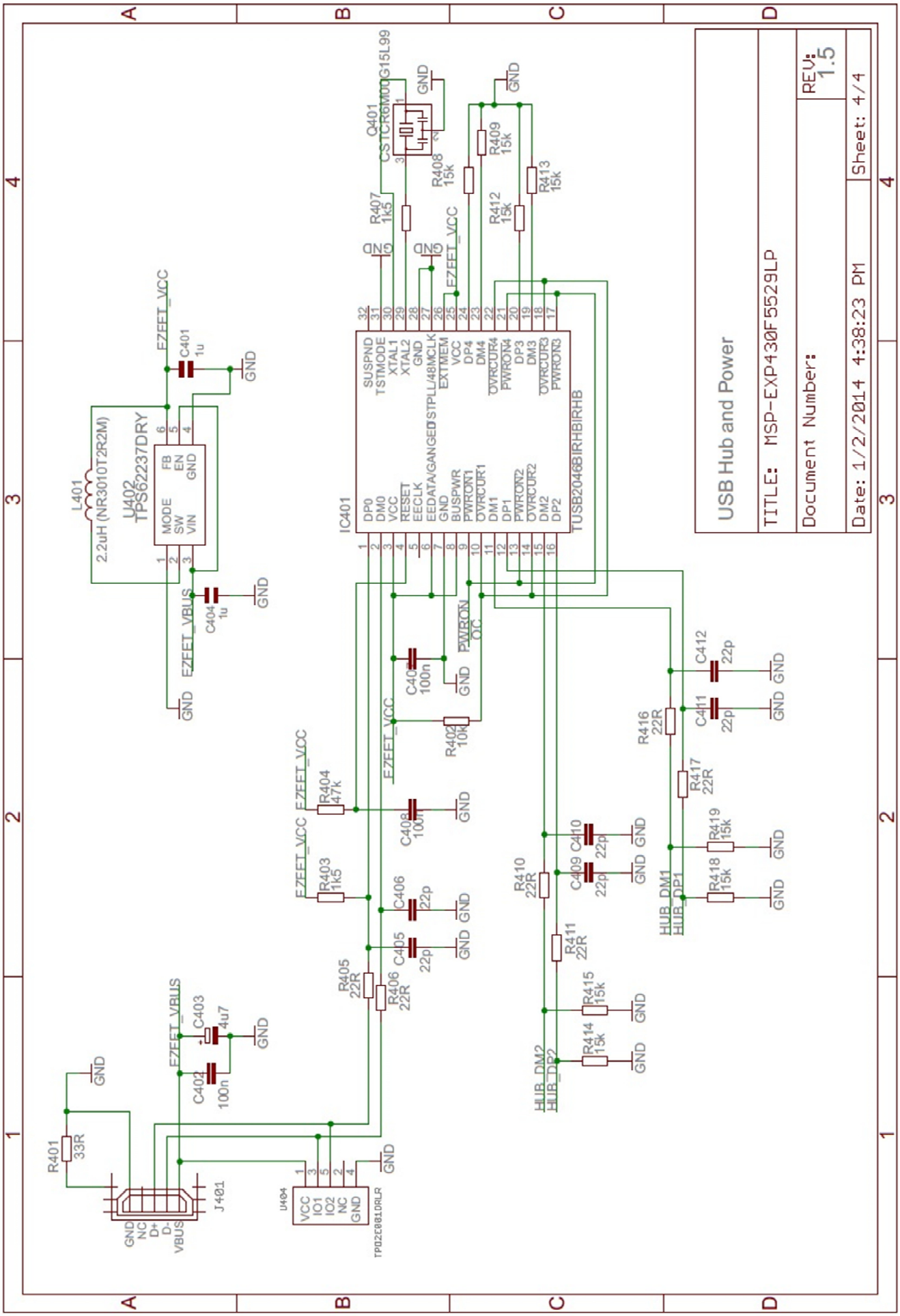




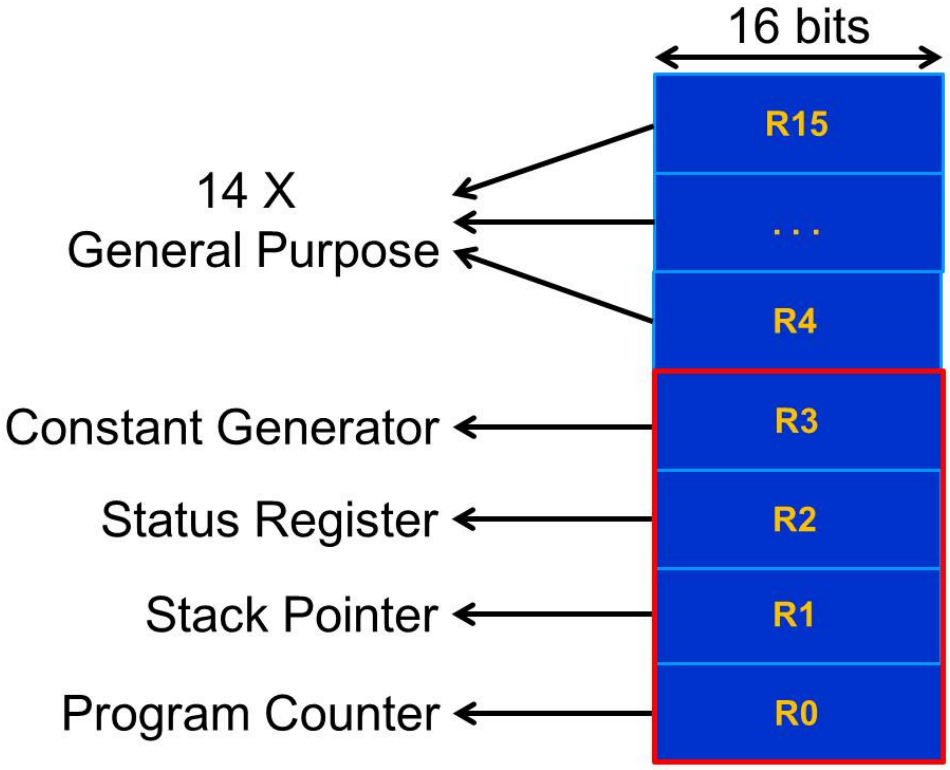

XT2 Q4
CSTCR4M00G15L99
C16 220n
C15 220n
C27 4u7
+5V
GND
RESET
+3V3
R27 47k
C17 1n
S3 RST
TEST/SBWTCK
RST/SBWTDIO
Speed
R15 1k4
TARGET_DM
TARGET_DP
BSL
S5 BSL
R14 100R
R16 1M
TARGET_PUR
U6
MSP430F552x
Analog VCC
R28 0R
C29 100n
LFXTCLK
C26 12p
C25 12p
32.768kHz
Q2
C30 10u
C19 100n
C18 470n
C14 100n
R31 27R
R32 27R
C24 10p
C23 10p
HUB_DM
HUB_DP
P7.4
P4.7
P4.5/UCA1RXD
P4.4/UCA1TXD
P4.3
P4.2
P4.1
P4.0
P3.7
P3.6
P3.5
P3.4
Main MSP430F5529 w Power, Clock and USB
TITLE: MSP-EXP430F5529LP
Document Number:
REV: 1.5
Date: 1/2/2014 4:38:23 PM
Sheet: 1/4
Design Notes:
Remove R28 and inject AVCC directly if needed
Free IO pins: 4.6, 5.0, 5.1, 5.6, 5.7, 7.1, 7.2, 7.3, 7.5, 7.6, 7.7, 8.0, PJ.x (4)


40 pin BoosterPack Headers
GND
C31
100n
+3V3
J1
C28
+5V
P6.5
P3.4
P3.3
P1.6
P6.6
P3.2
P2.7
P4.2
P4.1
P6.0
P6.1
P6.2
P6.3
P6.4
P7.0
P3.6
P3.5
J5
P2.5
P2.4
P1.5
P1.4
P1.3
P1.2
P4.3
P4.0
P3.7
P8.2
P2.0
P2.2
P7.4
RST/SBWTDIO
P3.0
P3.1
P2.6
P2.3
P8.1
User Buttons
S1
S2
P2.1
P1.1
User LEDs
P4.7
R30
470R
LED2
JP8
P1.0
LED1_PWR
R29
LED1
ezFET <<---->> LaunchPad
JP1
+5V
EZFET_VBUS
JP2
+3V3
EZFET_VCC
JP3
EZFET_SBW_TST
JP4.1
SBW_TST
TEST/SBWTCK
EZFET_SBW_RST
JP4.2
SBW_RST
RST/SBWTDIO
EZFET_SBW_NC
JP4.3
SBW_NC
JP6.1
TXD
P4.4/UCA1TXD
JP6.2
RXD
P4.5/UCA1RXD
JP6.3
CTS
P1.7
JP6.4
RTS
P6.7
J3
C21
PWR
J10
C20
STAND4
STAND3
STAND2
STAND1
Connections
TITLE: MSP-EXP430F5529LP
Document Number:
REV: 1.5
Date: 1/2/2014 4:38:23 PM
Sheet: 2/4


eZ-FET lite Emulator
eZ-FET Lite Rev1.0
TITLE: MSP-EXP430F5529LP
Document Number:
REV: 1.5
Date: 1/2/2014 4:38:23 PM
Sheet: 3/4
MSP101
F55[0/1/2]X
MSP430F5528IRGC
DEBUG
TARGET VCC SENSE
EZFET_VCC
EZFET_AVCCOUT2ADC
EZFET_TEST
EZFET_TDO
EZFET_TDI
EZFET_TMS
EZFET_TCK
EZFET_RST
EZFET_SBW_NC
EZFET_SBW_TST
EZFET_SBW_RST
EZFET_UARTTXD
EZFET_UARTRXD
EZFET_UARTRTS
EZFET_UARTCTS
EZFET_VUSB
EZFET_V18
EZFET_VBUS
EZFET_AVBUS
EZFET_VREF
EZFET_PUR
EZFET_PU.1/DM
EZFET_PU.0/DP
EZFET_XT2IN
EZFET_XT2OUT
EZFET_LED0
EZFET_LED1
HUB_DM2
HUB_DP2
Q101
PIEZO_CSTCR4M00G15L99
LED101 red
LED102 green
TP104
TP105
TP106
TP107
TP108
TP109
TP110
R101 470
R102 390
R103 27R
R104 27R
R105 1k4
R106 1M
R107 47k
R109 47k
R121 220k
R122 220k
R124 240k
R125 150k
C101 470n
C102 220n
C103 100n
C104 10u
C107 220n
C110 10p
C111 10p
C112 1n
C113 100n
C121 100n
C122 33p
C123 33p
GND


USB Hub and Power
TITLE: MSP-EXP430F5529LP
Document Number:
REV: 1.5
Date: 1/2/2014 4:38:23 PM
Sheet: 4/4
IC401
TUSB2046BIRHB|RHB
U402
TPS62237DRY
L401
2.2uH (NR3010T2R2M)
Q401
CSTCR6M00G15L99
J401
U404
TPD2E001DRLR
EZFET_VCC
EZFET_VBUS
HUB_DM1
HUB_DP1
HUB_DM2
HUB_DP2
PWRON
OC
GND

MSP430 – 16 Registradores

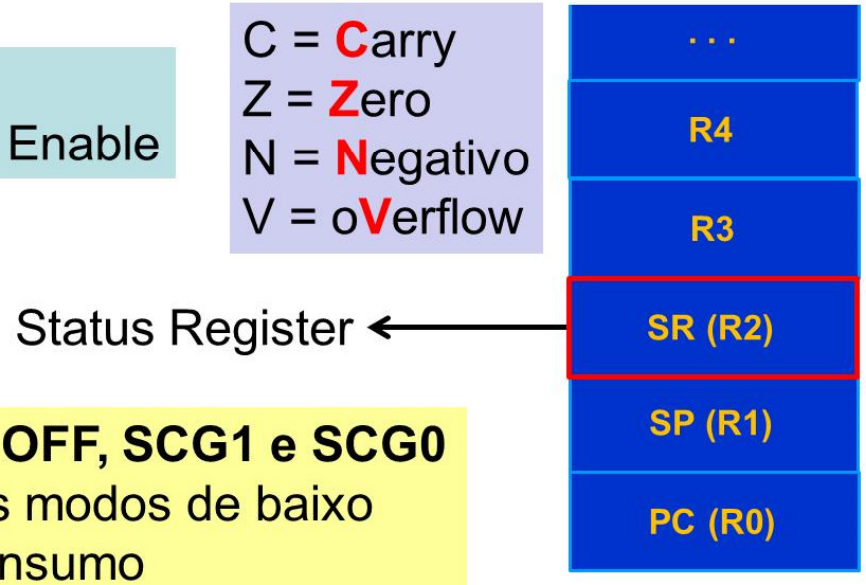


MSP430 – Status Register

15 ... 9	8	7	6	5	4	3	2	1	0
Reservados	V	SCG 1	SCG 0	OSC OFF	CPU OFF	GIE	N	Z	C

GIE
Global Interrupt Enable

C = **C**arry
Z = **Z**ero
N = **N**egativo
V = o**V**erflow



CPU OFF, OSC OFF, SCG1 e SCG0
Controlam os modos de baixo consumo

MSP430 – Modos de Endereçamento

1. Register Mode	**Rn**	Rn = operando
2. Indexed Mode	**X(Rn)**	(Rn+X) = operando
3. Symbolic Mode	**X**	(PC+X) = operando
4. Absolute Mode	**&ADDR**	(ADDR) = operando
5. Indirect Reg. Mode	**@Rn**	(Rn) = operando
6. Indirect Autoincr.	**@Rn+**	(Rn) = oper. e Rn++
7. Immediate Mode	**#N**	N é o operando

Não podem ser usados como destino

MSP430 – Instruções

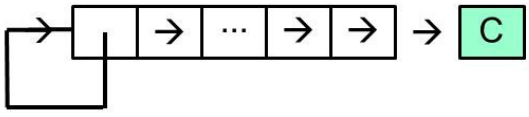
Rotações e Deslocamentos

20 **rla dst**

Arithmetic Shift

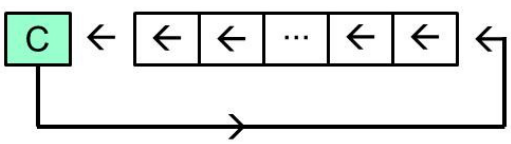
16 **rra dst**

Arithmetic Shift



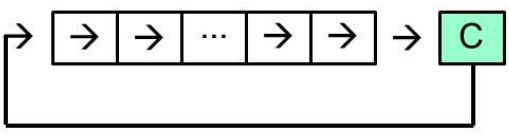
21 **rlc dst**

Rotation



17 **rrc dst**

Rotation



MSP430 – Instruções

Saltos Condicionais **Controle de Fluxo**

Executar antes a instrução que atualiza status

cmp src,dst **dst - src**

24	**jeq label**	**jz**	Se dst = src, **PC + label → PC**
25	**jne label**	**jnz**	Se dst ≠ src, **PC + label → PC**
21	higher or same **jhs label**	**jc**	Se dst ≥ src, **PC + label → PC**
22	lower **jlo label**	**jnc**	Se dst < src, **PC + label → PC**

Resumo Saltos Condicionais

Comparações para números sem sinal:

cmp src,dst → dst - src

Instr	Jump if ...	Cond.	Equiv.
jeq	Equal	**dst** = src	**jz**
jne	Not equal	**dst** ≠ src	**jnz**
jhs	Higher or same	**dst** ≥ src	**jc**
jlo	Lower	**dst** < src	**jnc**

Notem que a referência sempre é o **dst** !

MSP430 – Instruções

Saltos Condicionais **Controle de Fluxo**

Para números com sinal

Executar antes a instrução que atualiza status

cmp src,dst **dst - src**

greater or equal

26 **jge label** **jc** Se dst ≥ src, **PC + label → PC**

less than

27 **jl label** **jnc** Se dst < src, **PC + label → PC**

Resumo Saltos Condicionais

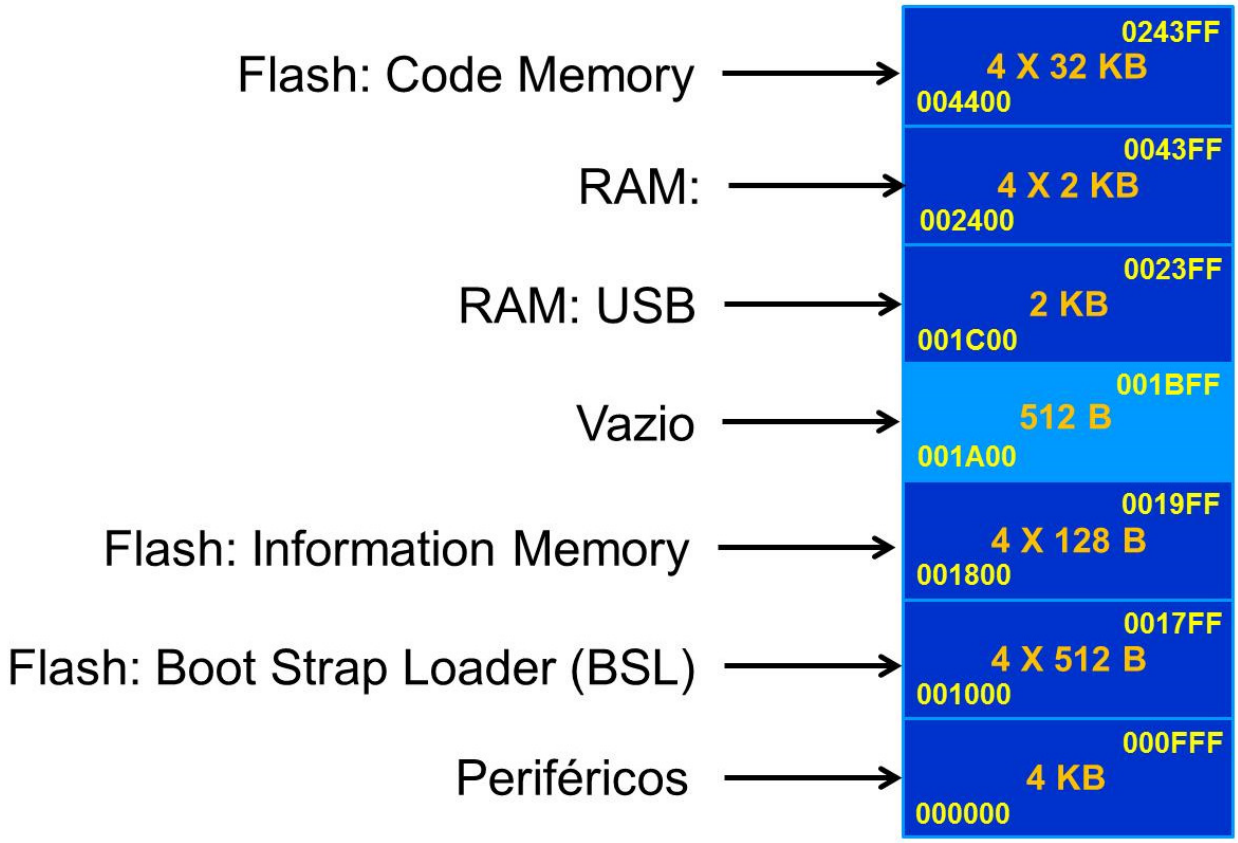
Comparações para números com sinal:

cmp srcd,dst → dst - src

Instr	Jump if ...	Flag	Cond
jge	Greater or equal	N⊕V = 0	dst ≥ src
jl	Less than	N⊕V = 1	dst < src

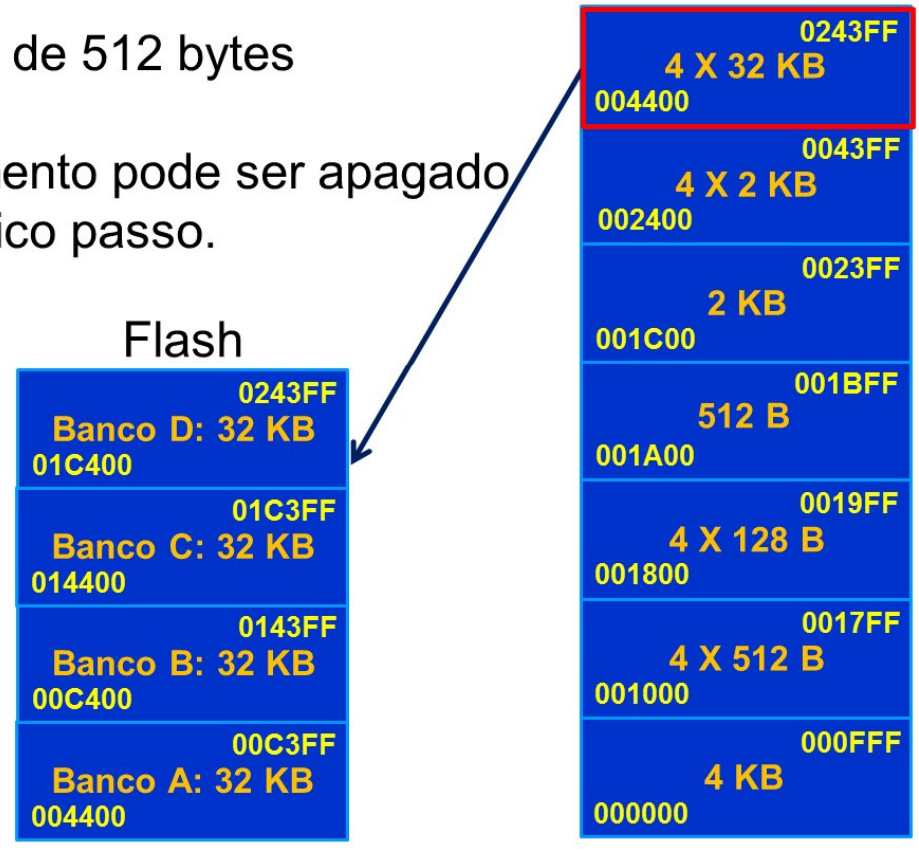
Notem que a referência sempre é o **dst** !

MSP430 – Memória



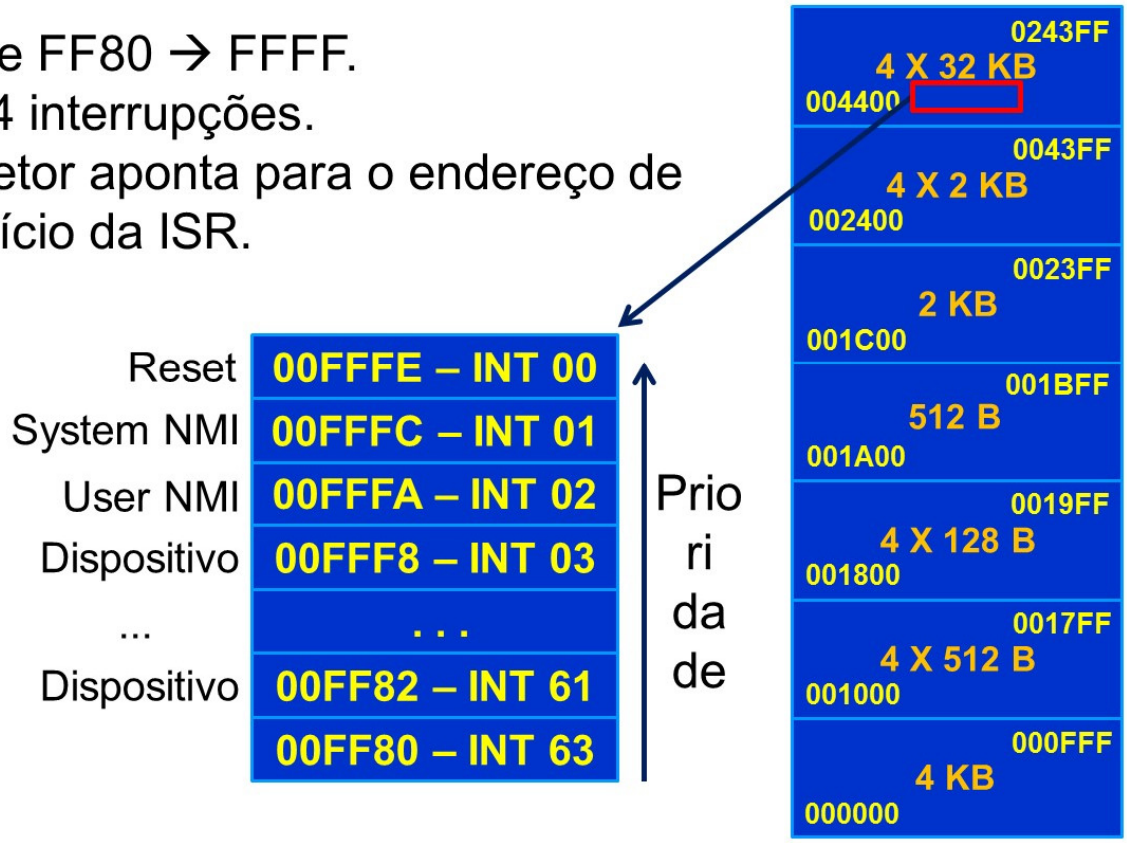
MSP430 – Code

- Segmentos de 512 bytes
- Cada segmento pode ser apagado com um único passo.

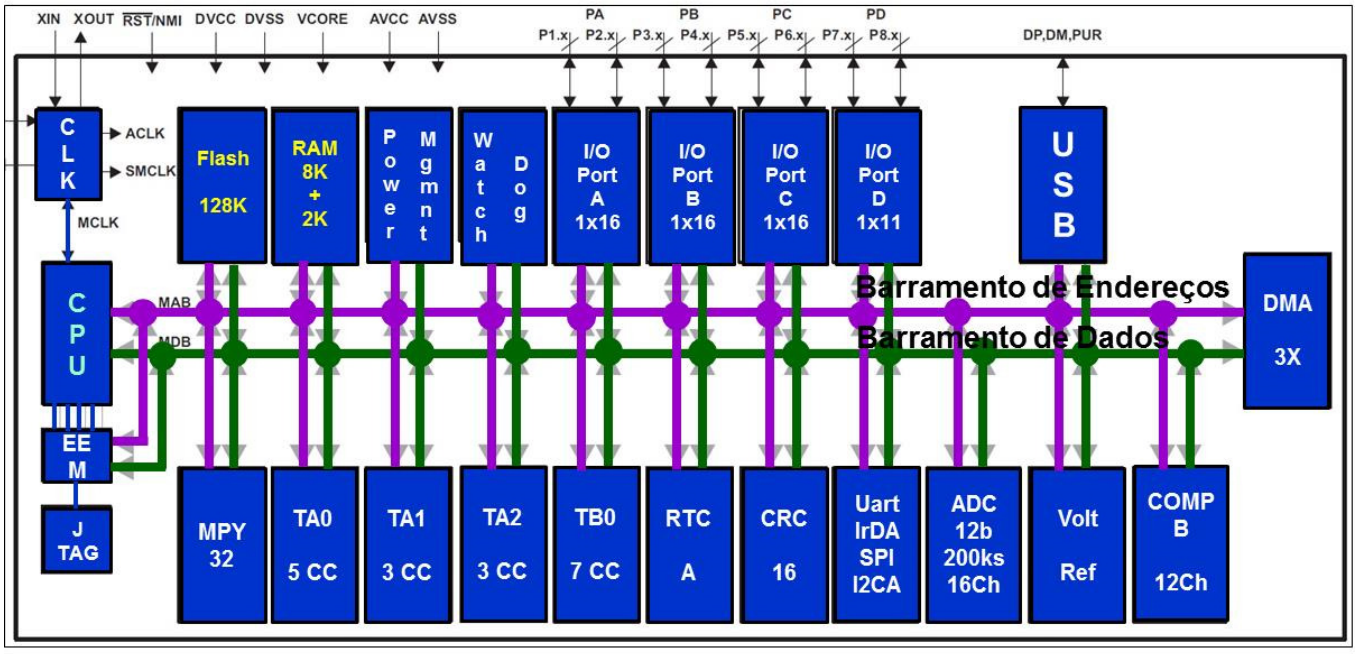


MSP430 – Vetores de Interrupção

- De FF80 → FFFF.
- 64 interrupções.
- Vetor aponta para o endereço de início da ISR.

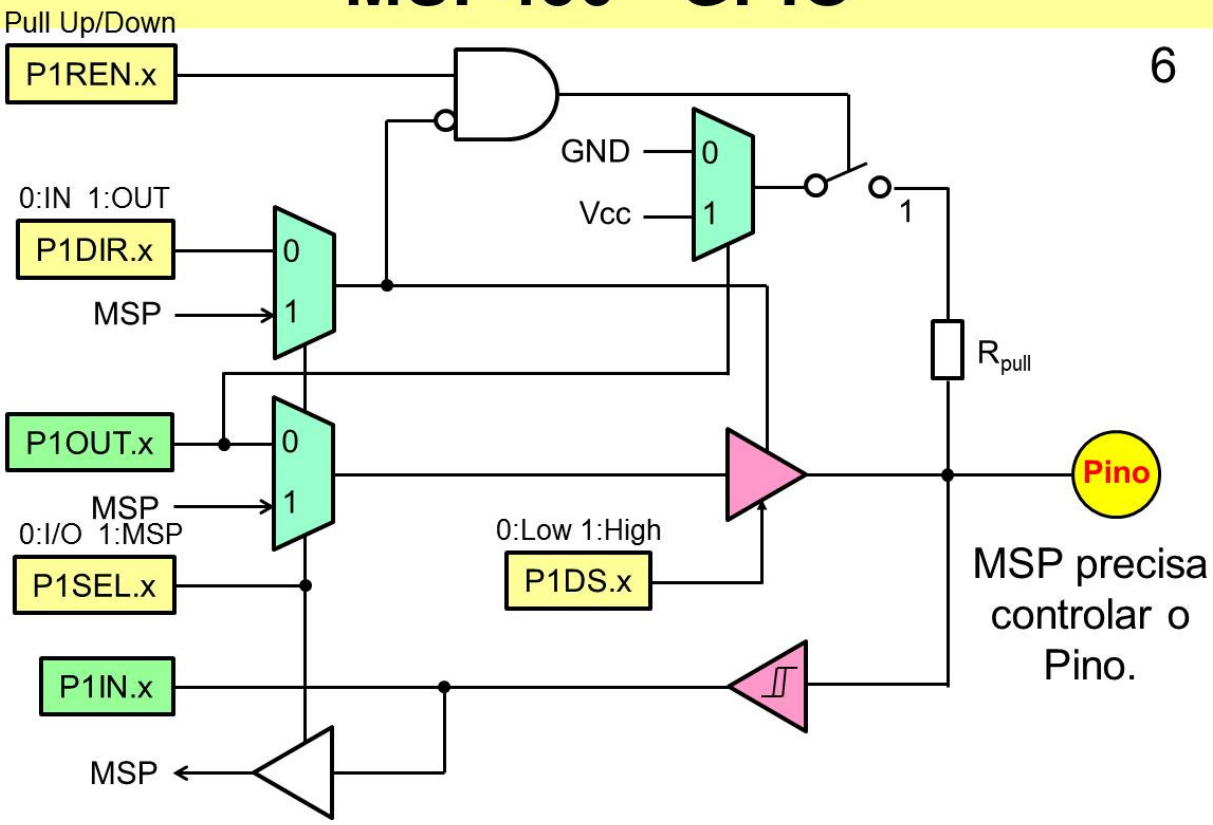


MSP430 – Diagrama de Blocos

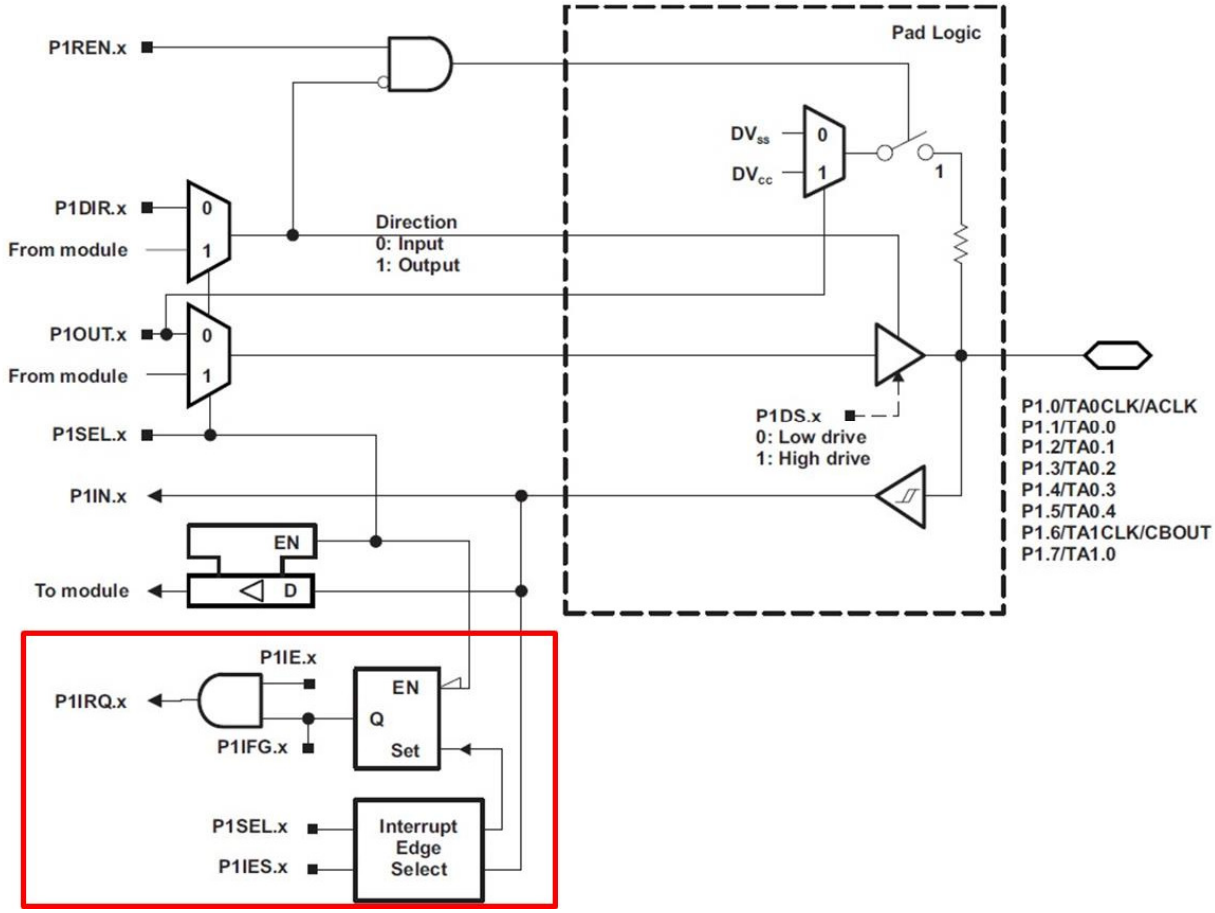


EEM = Embedded Emulation Module

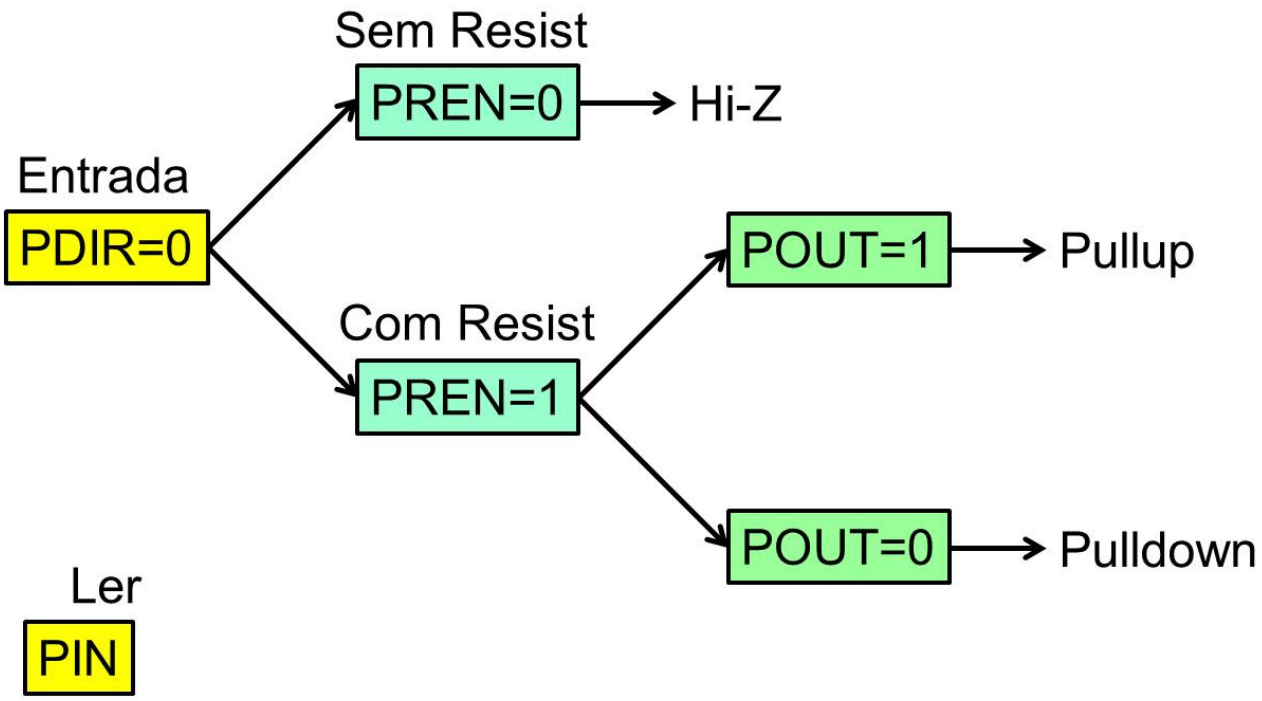
MSP430 - GPIO



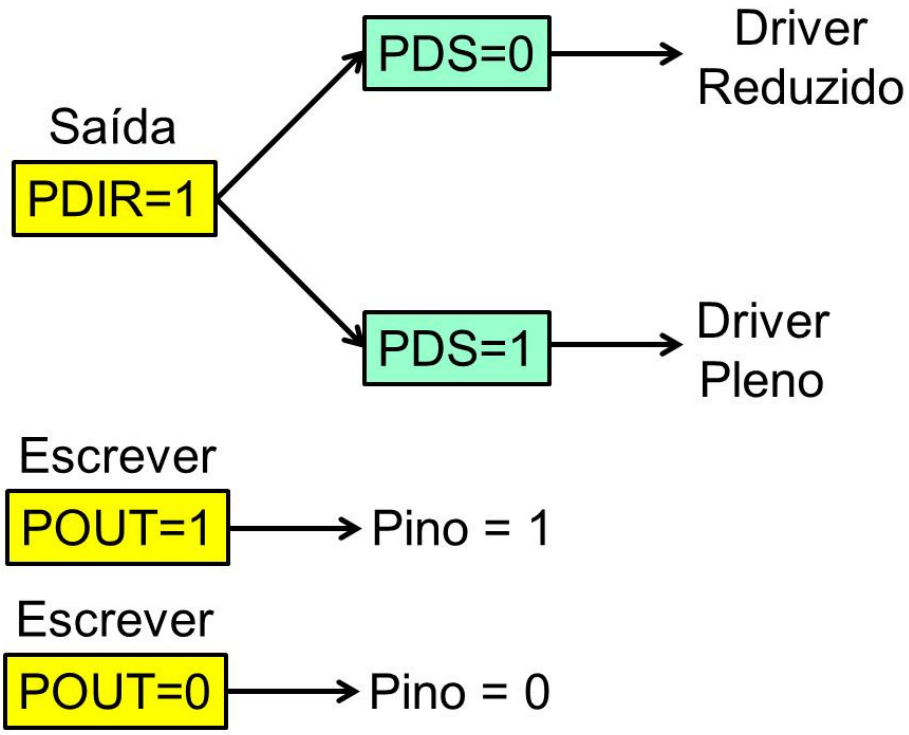
Port P1, P1.0 to P1.7, Input/Output With Schmitt Trigger



MSP430 – GPIO – Entrada: Resumo



MSP430 – GPIO – Saída: Resumo



MSP430 - GPIO

Registradores de Controle e Operação:

PxIN → (Modo Entrada) Ler o valor do pino

PxOUT → (Modo Saída) Escrever no pino
(Modo Entrada) 1 = Pullup, 0 = Pulldown

PxDIR → Direção: 0 = entrada, 1 = saída

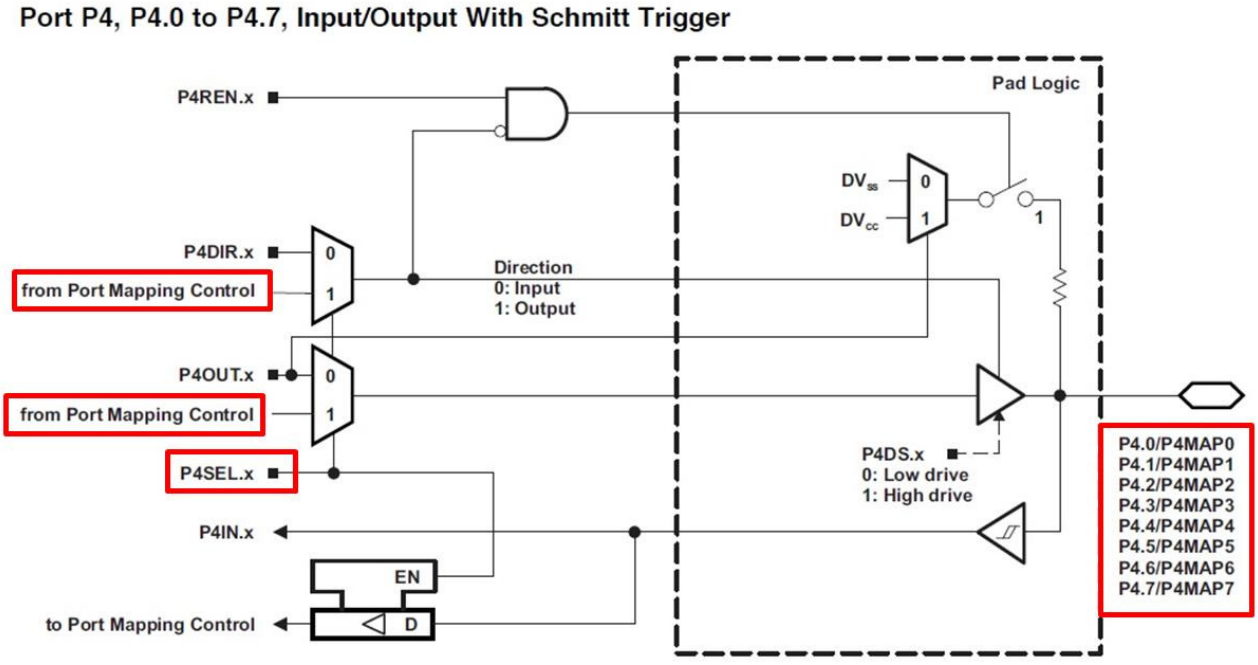
PxREN → Resistores: 0 = desabilita, 1 = habilita

PxDS → Driver: 0 = reduzido, 1 = pleno

PxSEL → Função: 0 = I/O, 1 = função

MSP430 – Mapeamento de P4

Diversos recursos podem ser mapeados na P4



MSP430 – Mapeamento de P4

Como mapear P4.7 para saída OUT0 do Timer B0?

Acesso ao mapeamento é habilitado com chave.

Tranca automaticamente após 32 ciclos.

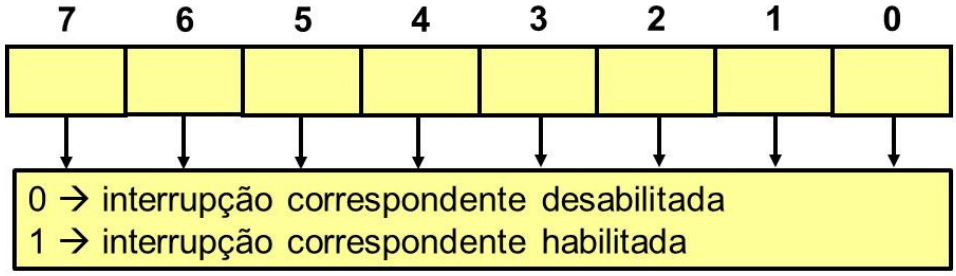
```
P4DIR |= BIT7;        //P4.7 como saída
P4SEL |= BIT7;        //P4.7 recebe saída alternativa

PMAPKEYID = 0X02D52;  //Liberar mapeamentp
P4MAP7 = PM_TB0CCR0A;        //Sair por P4.7
```

Table 10. Port Mapping Mnemonics and Functions

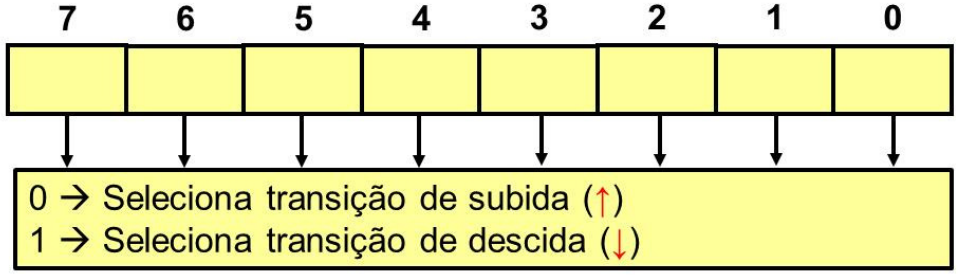
VALUE	PxMAPy MNEMONIC	INPUT PIN FUNCTION	OUTPUT PIN FUNCTION
0	PM_NONE	None	DVSS
1	PM_CBOUT0	-	Comparator_B output
	PM_TB0CLK	TB0 clock input	
2	PM_ADC12CLK	-	ADC12CLK
	PM_DMAE0	DMAE0 input	
3	PM_SVMOUT	-	SVM output
	PM_TB0OUTH	TB0 high impedance input TB0OUTH	
4	PM_TB0CCR0A	TB0 CCR0 capture input CCI0A	TB0 CCR0 compare output Out0
5	PM_TB0CCR1A	TB0 CCR1 capture input CCI1A	TB0 CCR1 compare output Out1
6	PM_TB0CCR2A	TB0 CCR2 capture input CCI2A	TB0 CCR2 compare output Out2
7	PM_TB0CCR3A	TB0 CCR3 capture input CCI3A	TB0 CCR3 compare output Out3
8	PM_TB0CCR4A	TB0 CCR4 capture input CCI4A	TB0 CCR4 compare output Out4
9	PM_TB0CCR5A	TB0 CCR5 capture input CCI5A	TB0 CCR5 compare output Out5
10	PM_TB0CCR6A	TB0 CCR6 capture input CCI6A	TB0 CCR6 compare output Out6
11	PM_UCA1RXD	USCI_A1 UART RXD (Direction controlled by USCI - input)	
	PM_UCA1SOMI	USCI_A1 SPI slave out master in (direction controlled by USCI)	
12	PM_UCA1TXD	USCI_A1 UART TXD (Direction controlled by USCI - output)	
	PM_UCA1SIMO	USCI_A1 SPI slave in master out (direction controlled by USCI)	
13	PM_UCA1CLK	USCI_A1 clock input/output (direction controlled by USCI)	
	PM_UCB1STE	USCI_B1 SPI slave transmit enable (direction controlled by USCI)	
14	PM_UCB1SOMI	USCI_B1 SPI slave out master in (direction controlled by USCI)	
	PM_UCB1SCL	USCI_B1 I2C clock (open drain and direction controlled by USCI)	
15	PM_UCB1SIMO	USCI_B1 SPI slave in master out (direction controlled by USCI)	
	PM_UCB1SDA	USCI_B1 I2C data (open drain and direction controlled by USCI)	
16	PM_UCB1CLK	USCI_B1 clock input/output (direction controlled by USCI)	
	PM_UCA1STE	USCI_A1 SPI slave transmit enable (direction controlled by USCI)	
17	PM_CBOUT1	None	Comparator_B output
18	PM_MCLK	None	MCLK
19 - 30	Reserved	None	DVSS
31 (0FFh)(1)	PM_ANALOG	Disables the output driver as well as the input Schmitt-trigger to prevent parasitic cross currents when applying analog signals.	

MSP430 – Interrupções P1 e P2

PnIE → Habilita interrupção da porta n (1 ou 2)

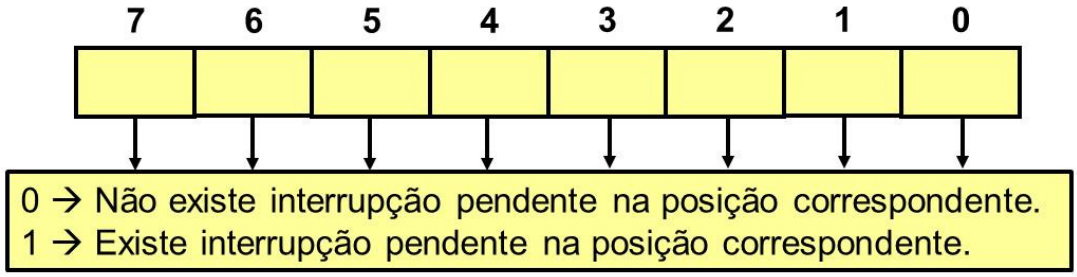


PnIES → Selector do Flanco n (1 ou 2)



MSP430 – Interrupções P1 e P2

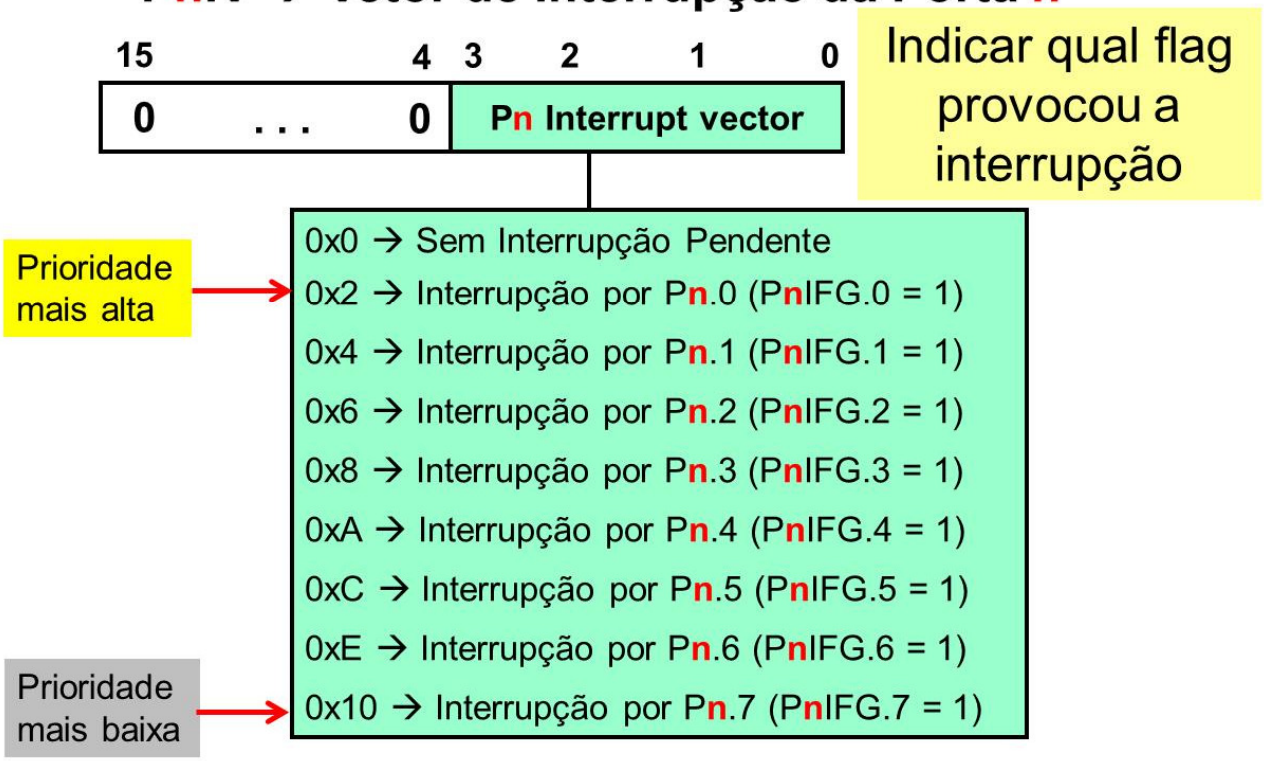
PnIFG → Interrupção pendente na porta n (1 ou 2)



- É possível colocar qualquer um dos flags **PnIFG.x** em 1 e com isso provocar uma interrupção por software.
- Alterações em **PnOUT**, **PnDIR**, **PnREN** podem ativar o flag de interrupção correspondente.

MSP430 – Interrupções TAx

PnIV → Vetor de Interrupção da Porta n



MSP430 – Vetores de Interrupções

Pri	ASM	Vetor ISR em C	Fonte	Flag	IV Reg	Endereço
63	".int63"	RESET_VECTOR	Reset	WDTIFG, KEYV	SYSRSTIV	0xFFFE
62	".int62"	SYSNMI_VECTOR	System NMI	SVMLIFG, ...	SYSSNIV	0XFFFC
61	".int61"	UNMI_VECTOR	User NMI	NMIIFG, OFIFG, ...	SYSUNIV	0xFFFA
60	".int60"	COMPB_VECTOR	Comp_B	CBIV	CBIV	0xFFF8
59	".int59"	TIMER_B0_VECTOR	TB0	TB0CCR0 CCIFG0	-	0xFFF6
58	".int58"	TIMER_B1_VECTOR	TB0	CCIFG1 a CCIFG6	TB0IV	0xFFF4
57	".int57"	WDT_VECTOR	Watch Dog	WDTIFG	-	0xFFF2
56	".int56"	USCI_A0_VECTOR	USCI_A0 T=Rx ou Tx	UCA0RXIFG, UCA0TXIFG	UCA0IV	0xFFF0
55	".int55"	USCI_B0_VECTOR	USCI_B0 T=Rx ou Tx	UCB0RXIFG, UCB0TXIFG	UCB0IV	0xFFEE

MSP430 – Vetores de Interrupções

Pri	ASM	Vetor ISR em C	Fonte	Flag	IV Reg	Endereço
54	“.int54”	ADC12_VECTOR	ADC12_A	ADC12IFG0 a ADC12IFG15	ADC12IV	0XFFEC
53	“.int53”	TIMER0_A0_VECTOR	TA0	TA0CCR0 CCIFG0	-	0xFFEA
52	“.int52”	TIMER0_A1_VECTOR	TA0	CCIFG1 a CCIFG6	TA0IV	0xFFE8
51	“.int51”	USB_UBM_VECTOR	USB_UBM	USB interrupts	USBIV	0xFFE6
50	“.int50”	DMA_VECTOR	DMA	DMA0IFG, DMA1IFG, DMA2IFG	DMAIV	0xFFE4
49	“.int49”	TIMER1_A0_VECTOR	TA1	TA1CCR0 CCIFG0	-	0xFFE2
48	“.int48”	TIMER1_A1_VECTOR	TA1	CCIFG1 a CCIFG6	TA1IV	0xFFE0

MSP430 – Vetores de Interrupções

Pri	ASM	Vetor ISR em C	Fonte	Flag	IV Reg	Endereço
47	“.int47”	PORT1_VECTOR	I/O Port 1	P1IFG.0 a P1IFG.7	P1IV	0xFFDE
46	“.int46”	USCI_A1_VECTOR	USCI_A1 T=Rx ou Tx	UCA1RXIFG, UCA1TXIFG	UCA1IV	0XFFDC
45	“.int45”	USCI_B1_VECTOR	USCI_B1 T=Rx ou Tx	UCB1RXIFG, UCB1TXIFG	UCB1IV	0xFFDA
44	“.int44”	TIMER2_A0_VECTOR	TA2	TA2CCR0 CCIFG0	-	0xFFD8
43	“.int43”	TIMER2_A1_VECTOR	TA2	CCIFG1 a CCIFG6	TA2IV	0xFFD6
42	“.int42”	PORT2_VECTOR	I/O Port 2	P2IFG.0 a P2IFG.7	RTCIV	0xFFD4
41	“.int41”	RTC_VECTOR	RTC_A	RTCRDYIFG, RTCTEVIFG, ...	-	0xFFD2
...	-	-	-	-	-	-
0	-	-	-	-	-	-

Table 4. Interrupt Sources, Flags, and Vectors

INTERRUPT SOURCE	INTERRUPT FLAG	SYSTEM INTERRUPT	WORD ADDRESS	PRIORITY
System Reset Power-Up External Reset Watchdog Timeout, Password Violation Flash Memory Password Violation	WDTIFG, KEYV (SYSRSTIV)[1][2]	Reset	0FFFEh	63, highest
System NMI PMM Vacant Memory Access JTAG Mailbox	SVMLIFG, SVMHIFG, DLYLIFG, DLYHIFG, VLRLIFG, VLRHIFG, VMAIFG, JMBNIFG, JMBOUTIFG (SYSSNIV)[1]	(Non)maskable	0FFFCh	62
User NMI NMI Oscillator Fault Flash Memory Access Violation	NMIIFG, OFIFG, ACCVIFG, BUSIFG (SYSUNIV)[1][2]	(Non)maskable	0FFFAh	61
Comp_B	Comparator B interrupt flags (CBIV)[1][3]	Maskable	0FFF8h	60
TB0	TB0CCR0 CCIFG0[3]	Maskable	0FFF6h	59
TB0	TB0CCR1 CCIFG1 to TB0CCR6 CCIFG6, TB0IFG (TB0IV)[1][3]	Maskable	0FFF4h	58
Watchdog Timer_A Interval Timer Mode	WDTIFG	Maskable	0FFF2h	57
USCI_A0 Receive or Transmit	UCA0RXIFG, UCA0TXIFG (UCA0IV)[1][3]	Maskable	0FFF0h	56
USCI_B0 Receive or Transmit	UCB0RXIFG, UCB0TXIFG (UCB0IV)[1][3]	Maskable	0FFEEh	55
ADC12_A	ADC12IFG0 to ADC12IFG15 (ADC12IV)[1][3][4]	Maskable	0FFECh	54
TA0	TA0CCR0 CCIFG0[3]	Maskable	0FFEAh	53
TA0	TA0CCR1 CCIFG1 to TA0CCR4 CCIFG4, TA0IFG (TA0IV)[1][3]	Maskable	0FFE8h	52
USB_UBM	USB interrupts (USBIV)[1][3]	Maskable	0FFE6h	51
DMA	DMA0IFG, DMA1IFG, DMA2IFG (DMAIV)[1][3]	Maskable	0FFE4h	50
TA1	TA1CCR0 CCIFG0[3]	Maskable	0FFE2h	49
TA1	TA1CCR1 CCIFG1 to TA1CCR2 CCIFG2, TA1IFG (TA1IV)[1][3]	Maskable	0FFE0h	48
I/O Port P1	P1IFG.0 to P1IFG.7 (P1IV)[1][3]	Maskable	0FFDEh	47
USCI_A1 Receive or Transmit	UCA1RXIFG, UCA1TXIFG (UCA1IV)[1][3]	Maskable	0FFDCh	46
USCI_B1 Receive or Transmit	UCB1RXIFG, UCB1TXIFG (UCB1IV)[1][3]	Maskable	0FFDAh	45
TA2	TA2CCR0 CCIFG0[3]	Maskable	0FFD8h	44
TA2	TA2CCR1 CCIFG1 to TA2CCR2 CCIFG2, TA2IFG (TA2IV)[1][3]	Maskable	0FFD6h	43
I/O Port P2	P2IFG.0 to P2IFG.7 (P2IV)[1][3]	Maskable	0FFD4h	42
RTC_A	RTCRDYIFG, RTCTEVIFG, RTCAIFG, RT0PSIFG, RT1PSIFG (RTCIV)[1][3]	Maskable	0FFD2h	41
Reserved	Reserved[5]		0FFD0h	40
			⋮	⋮
			0FF80h	0, lowest

MSP430 – Exemplo Interrupção P2.1

ASM

```
LED1        .equ    BIT0
LED1_OUT    .equ    P1OUT
LED2        .equ    BIT7
LED2_OUT    .equ    P4OUT
SW1         .equ    BIT1
SW1_OUT     .equ    P2OUT
            ;Preparar Leds e chave
            BIS.B   #LED1,&P1DIR
            BIC.B   #LED1,&LED1_OUT
            BIS.B   #LED2,&P4DIR
            BIC.B   #LED2,&LED2_OUT
            BIC.B   #SW1,P2DIR    ;SW1 como entrada
            BIS.B   #SW1,P2REN    ;SW1 hab resistor
            BIS.B   #SW1,P2OUT    ;SW1 com pull up
            BIS.B   #SW1,P2IES    ;SW1 flanco descida
            BIS.B   #SW1,P2IE     ;SW1 interrupção habilitada
            NOP
            EINT
            NOP
```

MSP430 – Exemplo Interrupção P2.1

ASM

```
P2_HND:     ADD     &P2IV,PC      ;Saltar na Tabela de jumps
            RETI
            JMP     P2_0_HND
            JMP     P2_1_HND
            JMP     P2_2_HND
            JMP     P2_3_HND
            JMP     P2_4_HND
            JMP     P2_5_HND
            JMP     P2_6_HND
            JMP     P2_7_HND
P2_0_HND:   RETI
P2_2_HND:   RETI
P2_3_HND:   RETI
P2_4_HND:   RETI
P2_5_HND:   RETI
P2_6_HND:   RETI
P2_7_HND:   RETI
;-------------------------------------------------------------------------------
; Interrupt Vectors
;-------------------------------------------------------------------------------
        .sect   ".reset"            ; MSP430 RESET Vector
        .short  RESET

          .sect   ".int42"          ; P2 interrupção
          .short  P2_HND
```

MSP430 –Interrupção em C

Pragmatic = pragmático, prático — Manual slau132o

#pragma vector = xxx_VECTOR Pag 111

- Indica que a função a seguir será usada como ISR.

__interrupt void **port2** (void) Pag 91 e 135 (6.7.2)

- Prepara a função para ser usada como ISR.
- Salva todos os registradores usados.
- Se a ISR chama alguma outra função, então salva todos os registradores da CPU.
- Retorno feito com RETI.

MSP430 –Interrupção em C

__even_in_range (x, num); Pag 111

- Retorna o valor de x.
- x deve ser **par**, na faixa de 0 até **num**.
- Facilitar uso do switch para vetores interrupção.

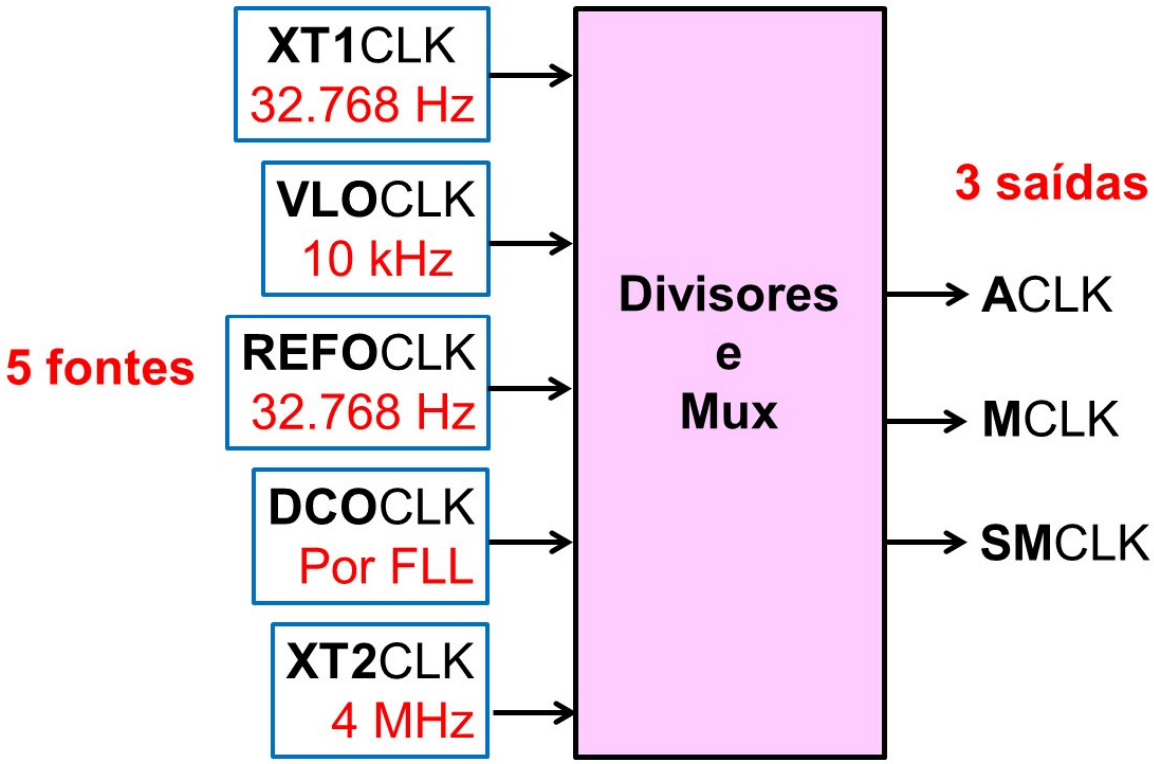
__enable_interrupt ( ); ou **__enable_interrupts ( )**

- Corresponde à instrução EINT. Pag 137

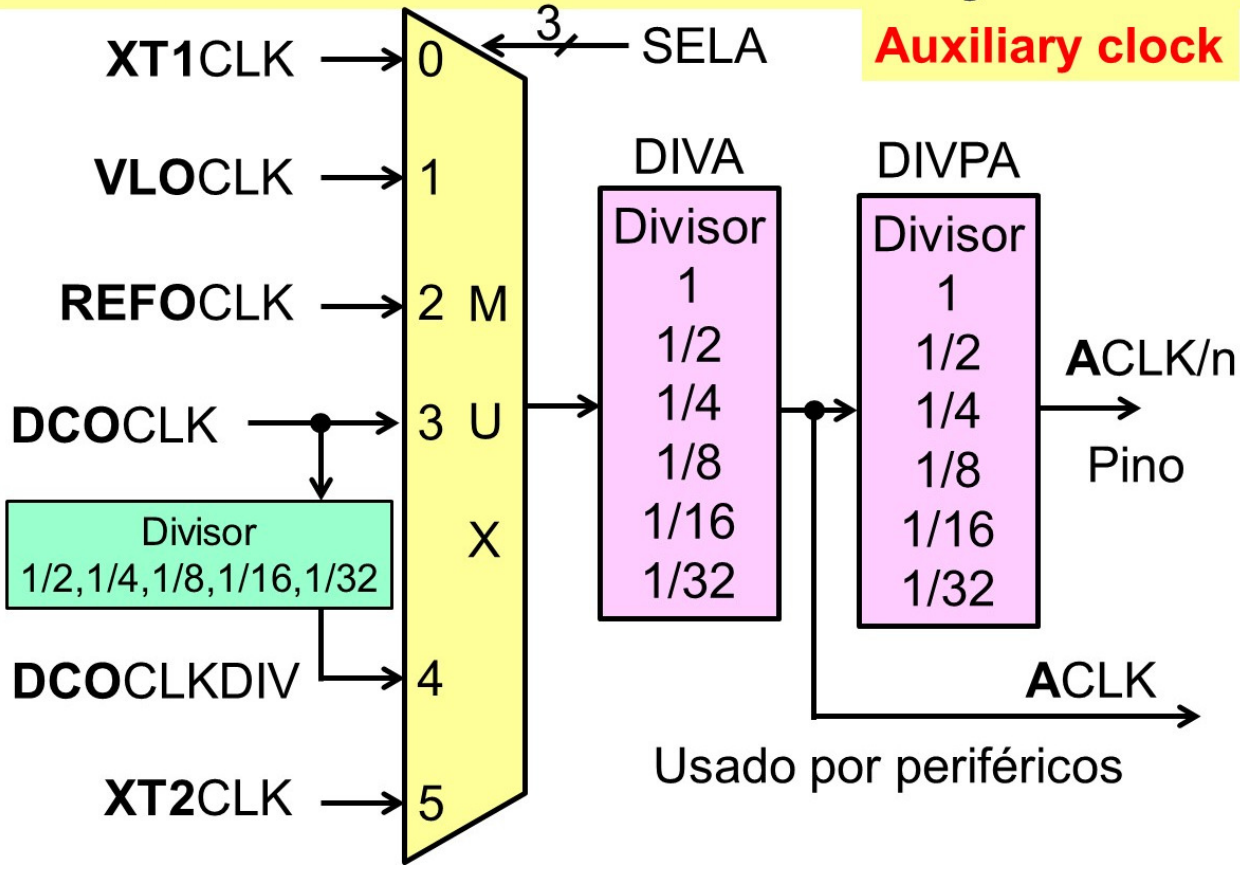
__disable_interrupt ( ); ou **__disable_interrupts ( )**

- Corresponde à instrução DINT. Pag 136

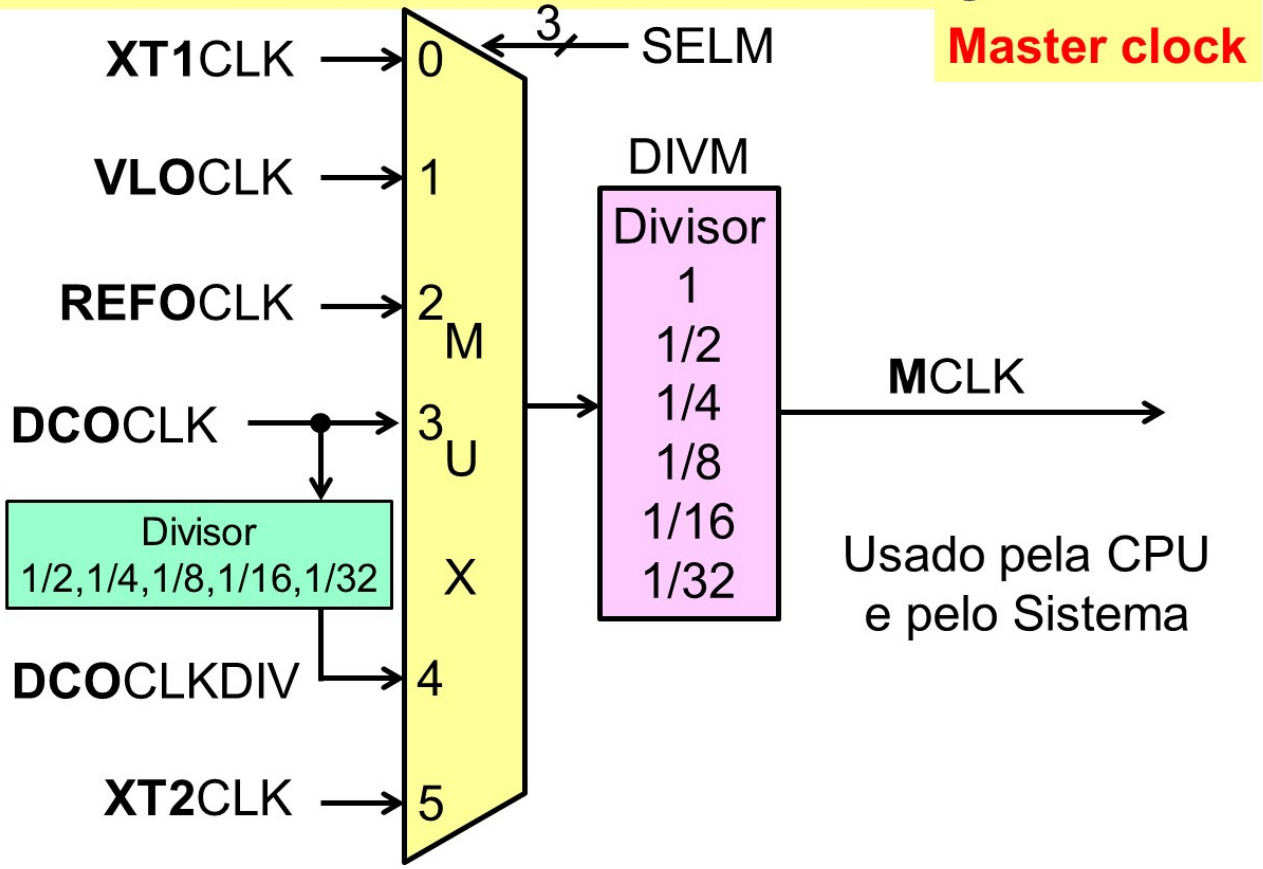
MSP430 – UCS:Unified Clock System



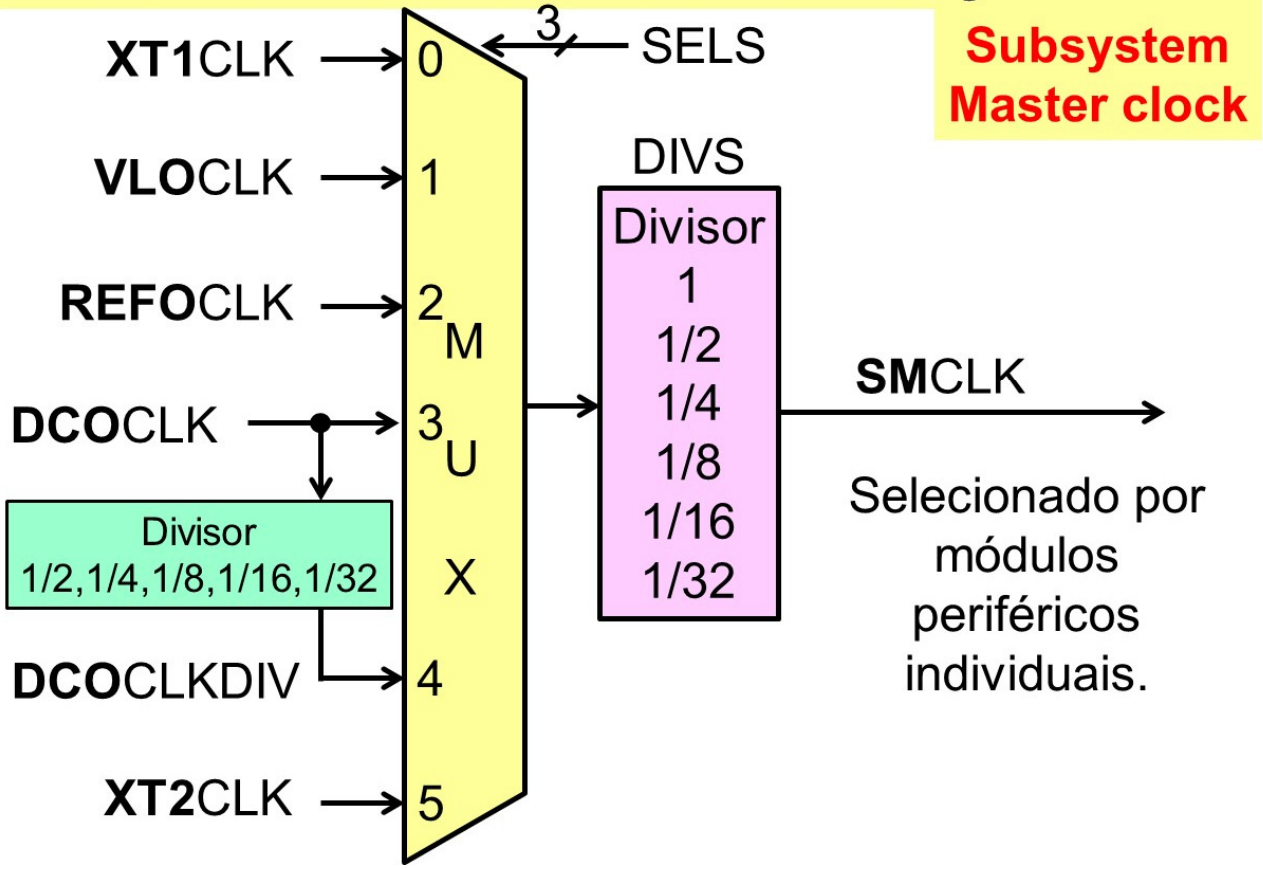
MSP430 – Unified Clock System



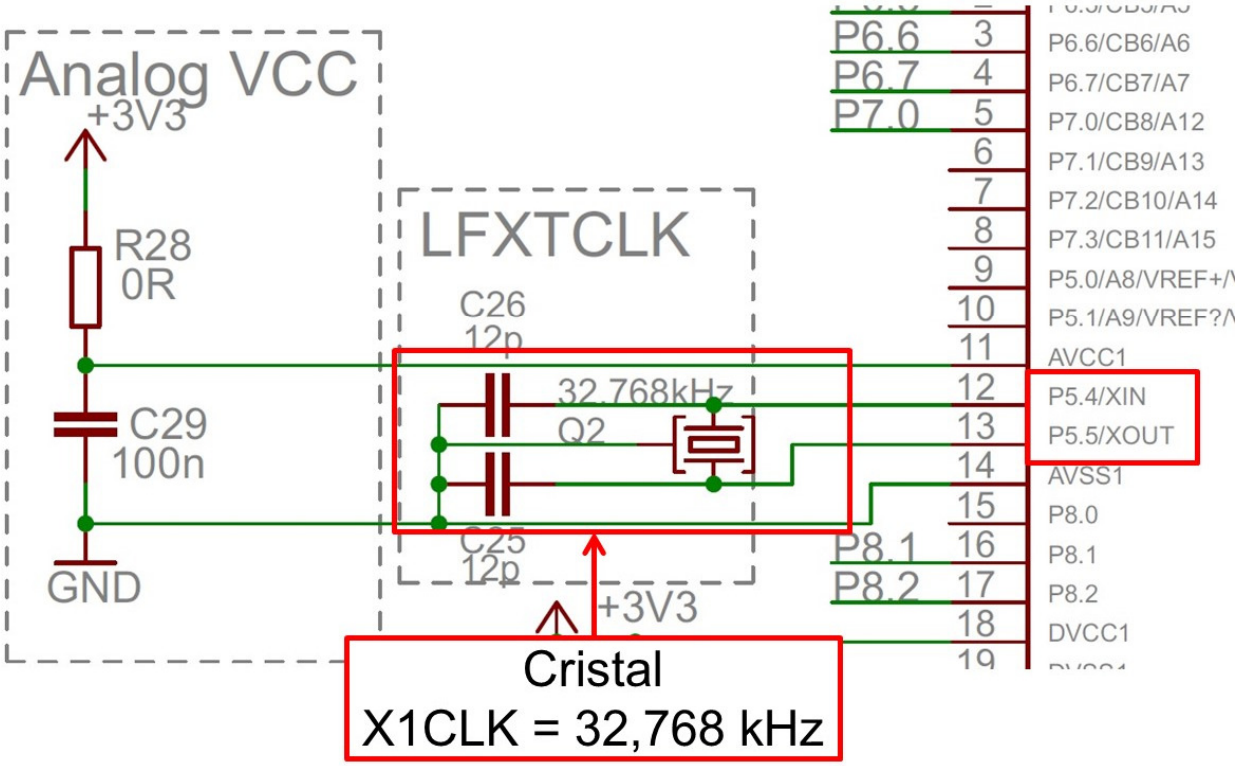
MSP430 – Unified Clock System



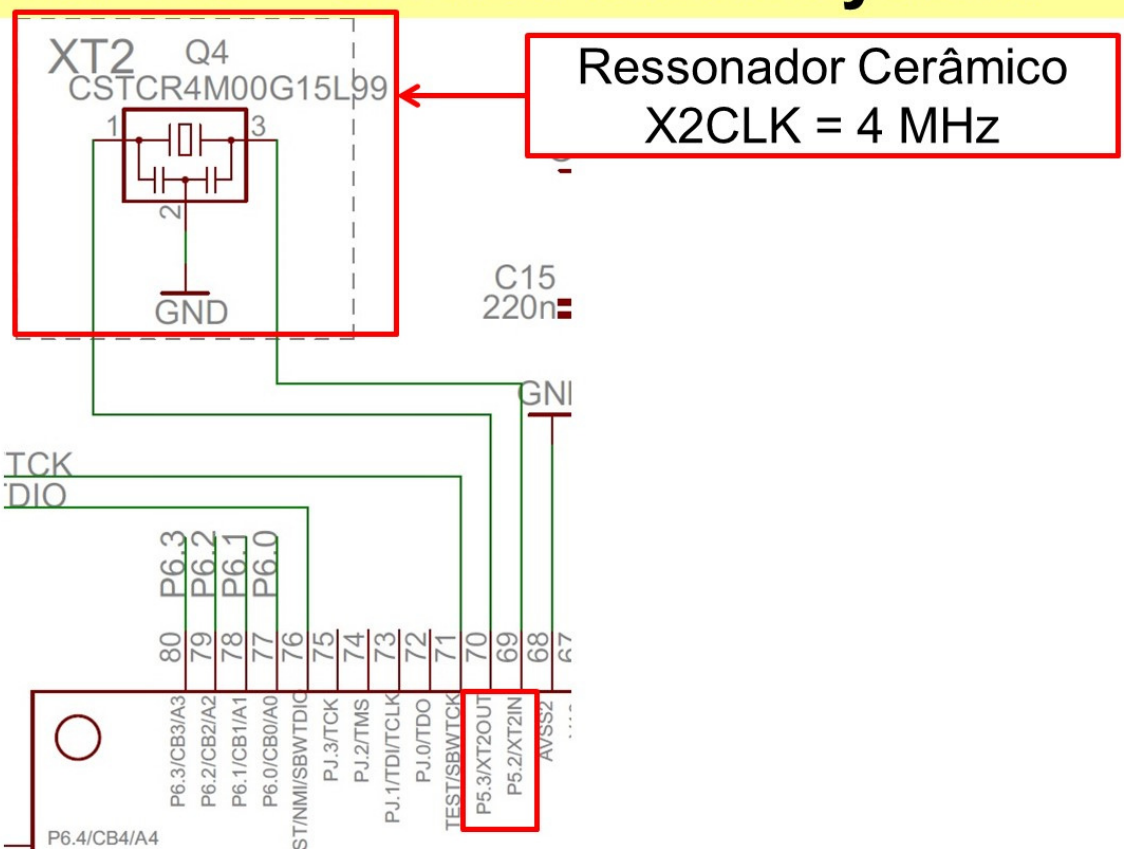
MSP430 – Unified Clock System



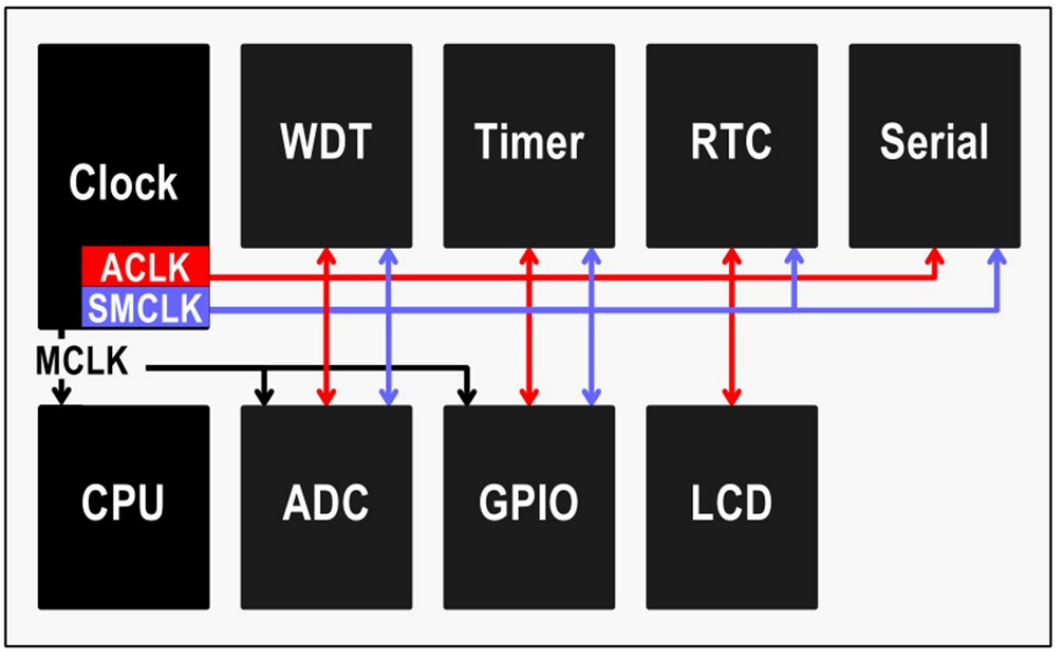
MSP430 – Unified Clock System



MSP430 – Unified Clock System



MSP430 – Unified Clock System



MSP430 – Unified Clock System

UCS: **U**nified **C**lock **S**ystem – Valores **Originais**

CPU rodando a 1 MHz

5 fontes | **Divisores e Mux** | **3 saídas**

XT1CLK → **32,768 kHz**

VLOCLK → **+/- 12 a 20 kHz**

REFOCLK → **32 kHz on-chip**

DCOCLK → **Variável**

XT2CLK → **4 MHz**

Fonte = REFO

ACLK = 32.768 Hz →

Ft = DCO, FLL

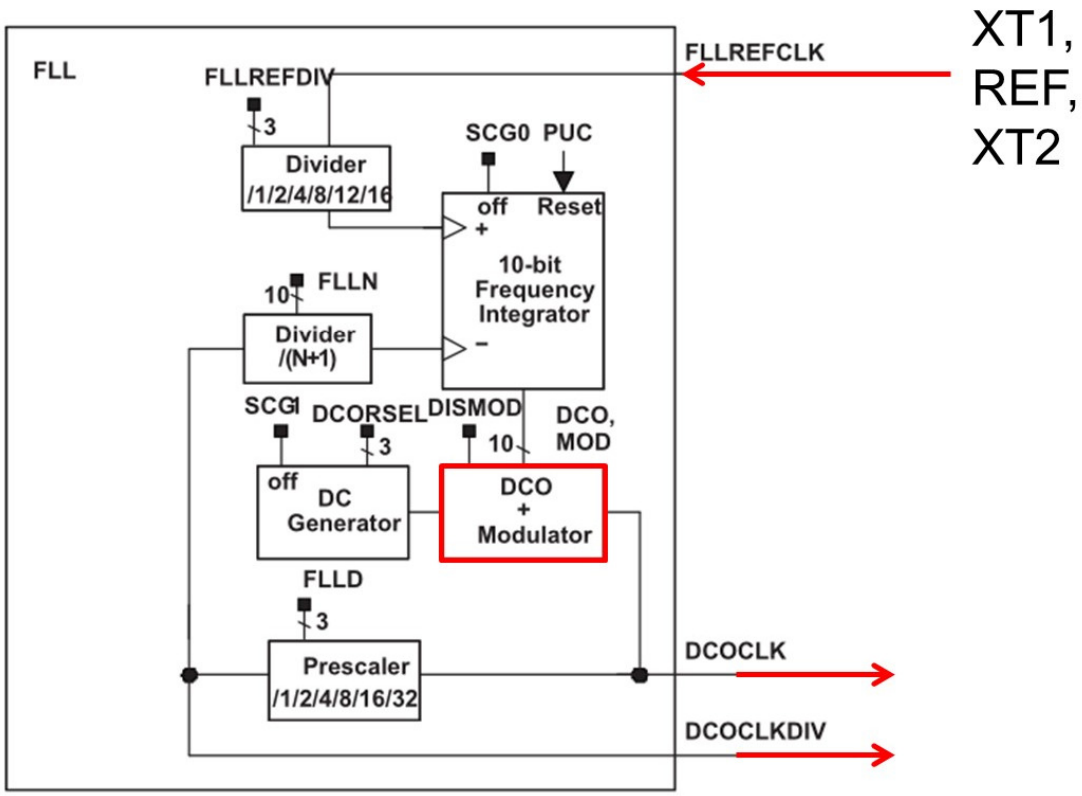
MCLK = 1048576 Hz →

Ft = DCO, FLL

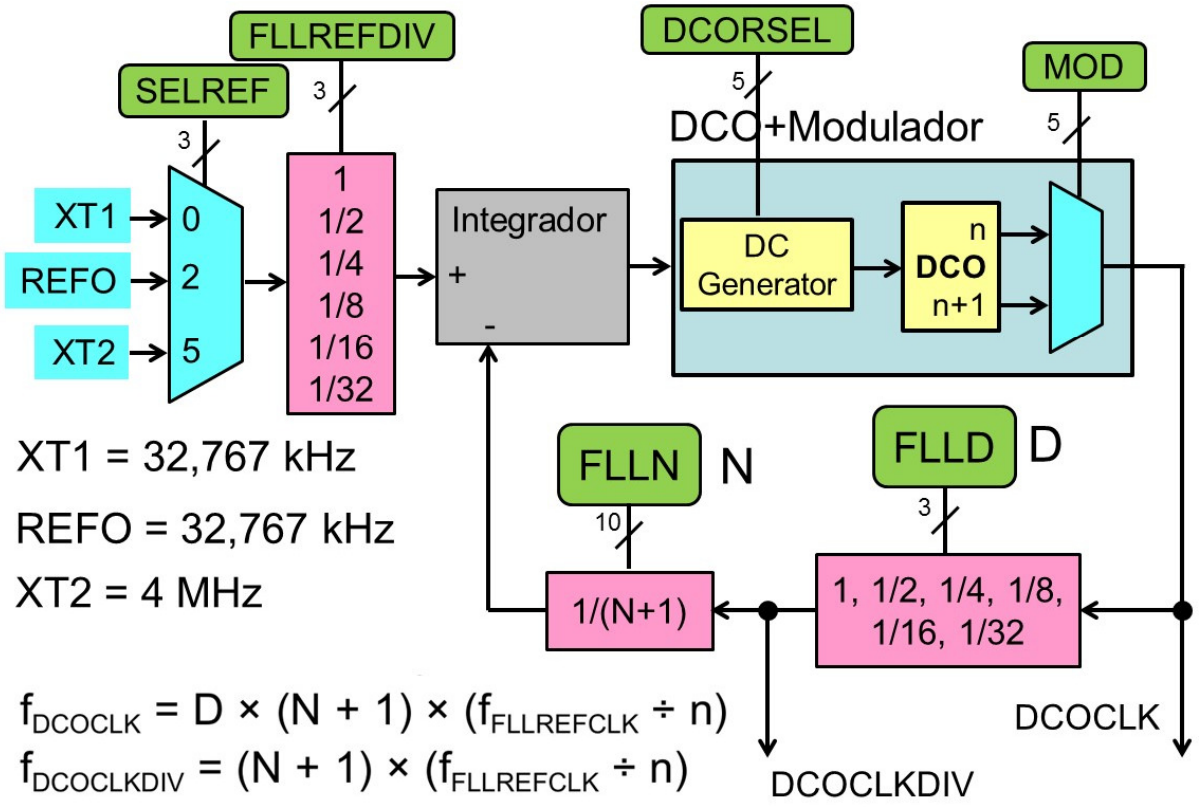
SMCLK = 1048576 Hz →

2^{20} = 1.048.576

MSP430 – Emprego do FLL

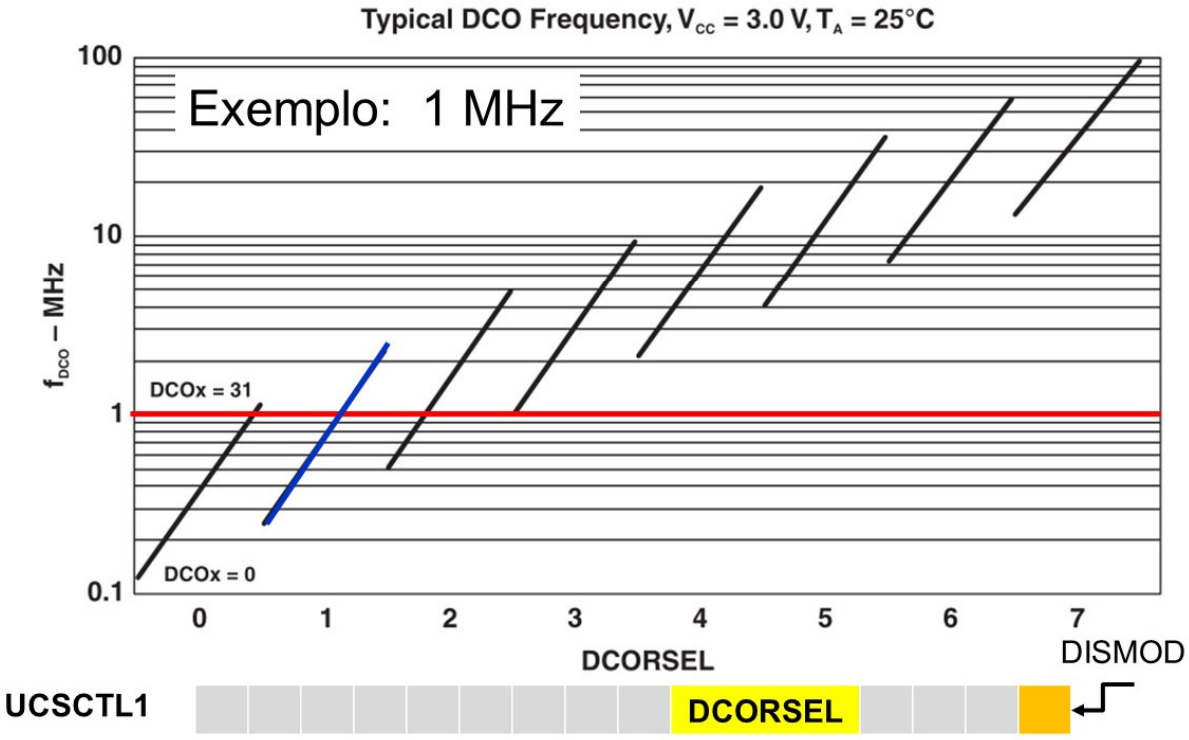


MSP430 – Loop FLL



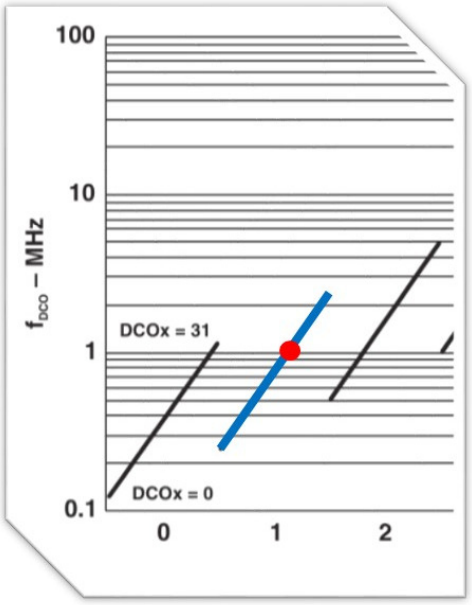
MSP430 – Como usar FLL

1) Escolher a faixa de frequência: **DCORSEL**



MSP430 – Como usar FLL

2) Selecionar um valor para **DCO**



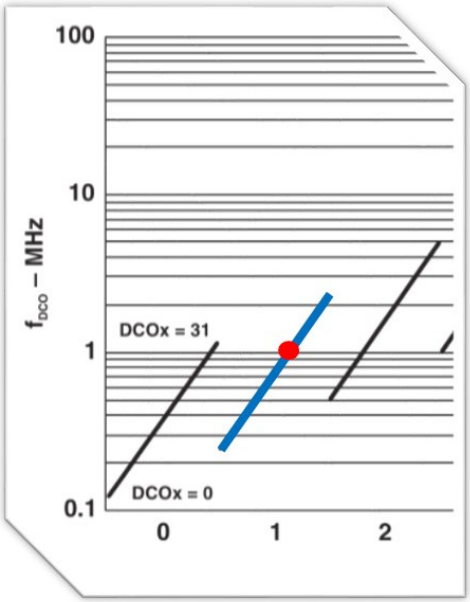
- **DCORSEL = 1**
- **DCO = 18**

A faixa é de 0 até 31.
Parece estar um pouco além da metade.

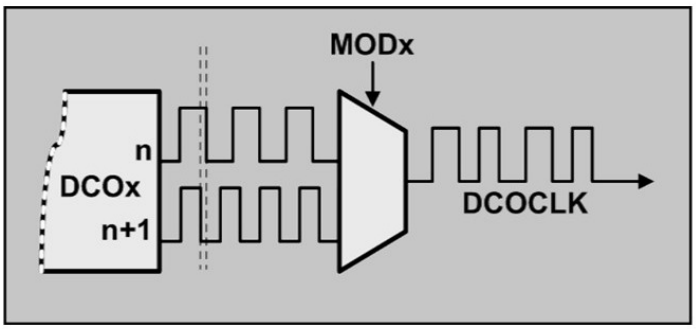
Não é um valor crítico!

MSP430 – Como usar FLL

3) Selecionar modulação: **MOD**



- **DCORSEL = 1**
- **DCO = 18**
- **MOD(n)** = interpola DCO(n) e DCO(n+1)



UCSCTL0

			DCO	DCO	MOD	MOD	

MSP430 – Como usar FLL

3) Selecionar modulação: **MOD**

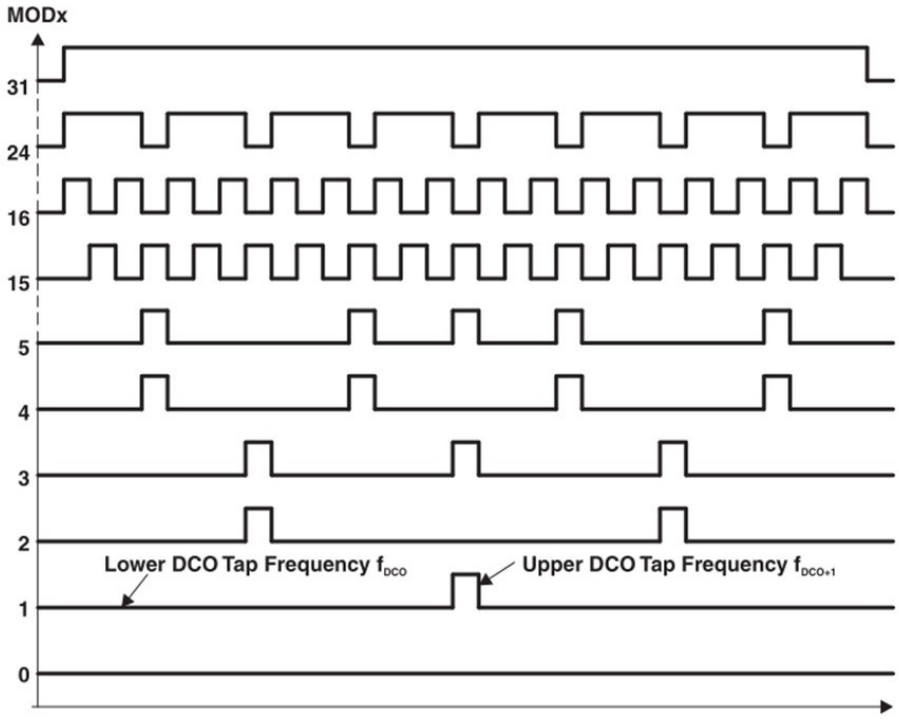
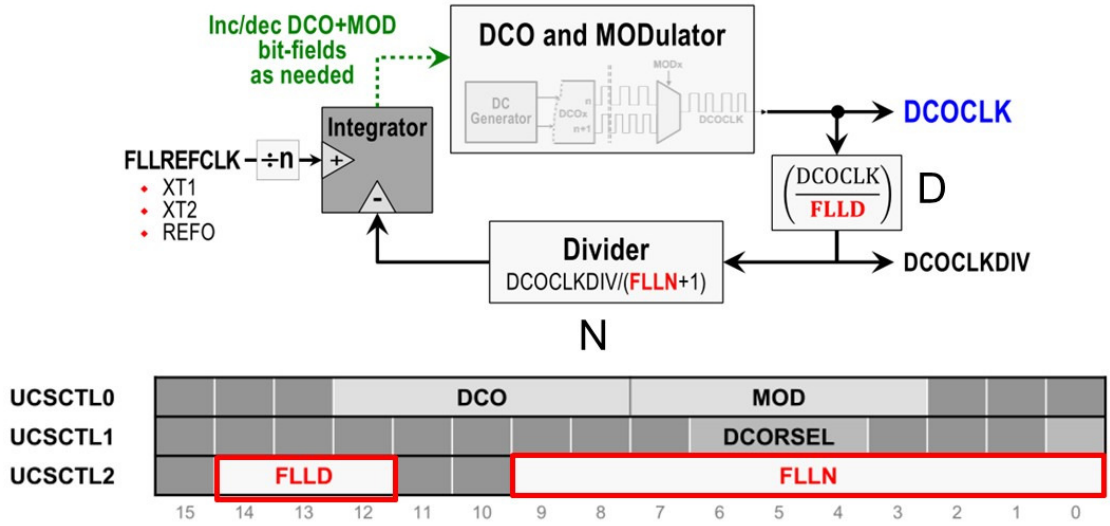


Figure 1-2. Modulator Patterns

MSP430 – Como usar FLL

4) Calcular valores de FLLD (**D**) e FLLN (**N**)

$$f_{DCOCLK} = D \times (N + 1) \times (f_{FLLREFCLK} \div n)$$

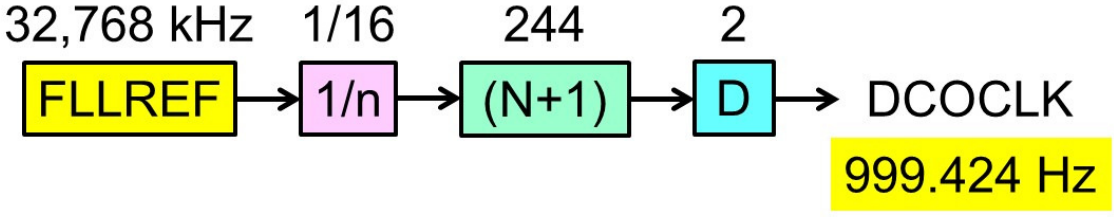


MSP430 – Como usar FLL

4) Calcular valores de FLLD (**D**) e FLLN (**N**)

$$f_{DCOCLK} = D \times (N + 1) \times (f_{FLLREFCLK} \div n)$$

Sugestão para 1 MHz



MSP430 – Ligar um oscilador xtal

Conexões dos cristais:

X1 (32.768 Hz) → P5.4 e P5.5

X2 (4 MHz) → P5.2 e P5.3

1) Configurar esses pinos para usar os cristais:

P5SEL |= BIT2 | BIT3 | BIT4 | BIT5;

2) Configurar drives e ligar osciladores:

UCSCTL6

XT1DRIVE = 0	XT2DRIVE = 0
XT1OFF = 0	XT2OFF = 0
XT1BYPASS = 0	XT2BYPASS = 0
XCAP=3	XTS = 0

MSP430 – Ligar um oscilador xtal

Conexões dos cristais:

X1 (32.768 Hz) → P5.4 e P5.5

X2 (4 MHz) → P5.2 e P5.3

1) Configurar esses pinos para usar os cristais:

P5SEL |= BIT2 | BIT3 | BIT4 | BIT5;

2) Configurar drives e ligar osciladores:

UCSCTL6

XT1DRIVE = 0	XT2DRIVE = 0
XT1OFF = 0	XT2OFF = 0
XT1BYPASS = 0	XT2BYPASS = 0
XCAP=3	XTS = 0

MSP430 – Ligar um oscilador xtal

5) Loop para esperar osciladores partirem, ou seja, esperar até flags de falta irem a zero

```
do {
        //Zerar os flags de falta XT1, XT2 e DCO
        UCSCTL7 &= ~(XT2OFFG  | XT1LFOFFG | DCOFFG);

        //Zerar flag de falta
        SFRIFG1 &= ~OFIFG;              //Zerar OFIFG
}       while (SFRIFG1 & OFIFG);        //OFIFG = 1?
```

- SFRIFG1.OFIFG → flag de interrupção para quando há falta no oscilador.
- Faz resumo das faltas do 3 osciladores.
- Flag é colocado em zero e ele volta a 1 se há falta.

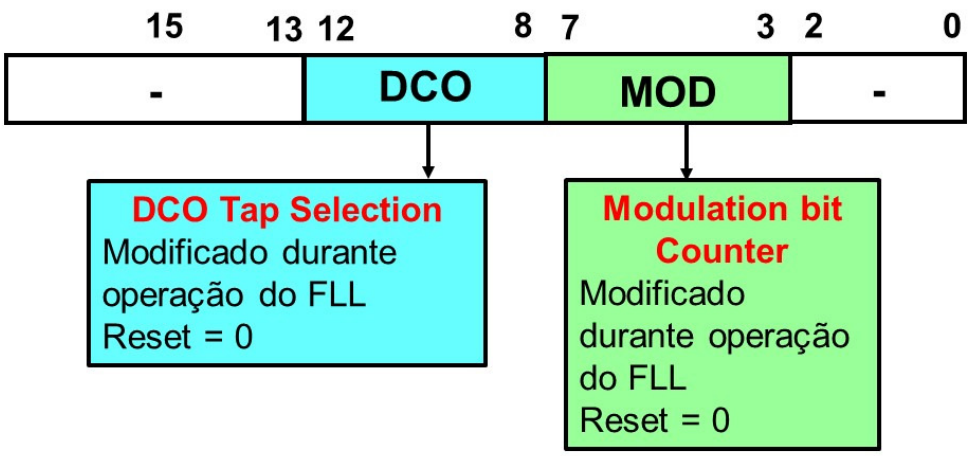
MSP430 – Registradores UCS

UCSCTL0	→	Operação do FLL
UCSCTL1	→	DCO (range e modulação)
UCSCTL2	→	Divisor do FLL (N e D)
UCSCTL3	→	Referência e divisor do FLL
UCSCTL4	→	**M**CLK, **A**CLK e **S**CLK
UCSCTL5	→	Divisor **M**CLK, **A**CLK e **S**CLK
UCSCTL6	→	Controle osciladores xtal
UCSCTL7	→	Fault dos osciladores
UCSCTL8	→	Clock request
UCSCTL9	→	Input range

MSP430 – Registradores UCS

UCSCTL0 → Controle 0 do UCS

Alterado durante operação

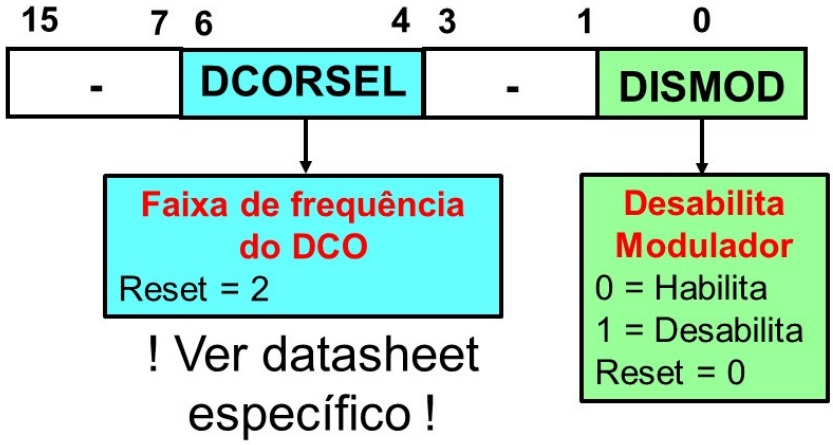


Se MOD passa de 0 → 31, então DCO++

Se MOD passa de 31 → 0, então DCO--

MSP430 – Registradores UCS

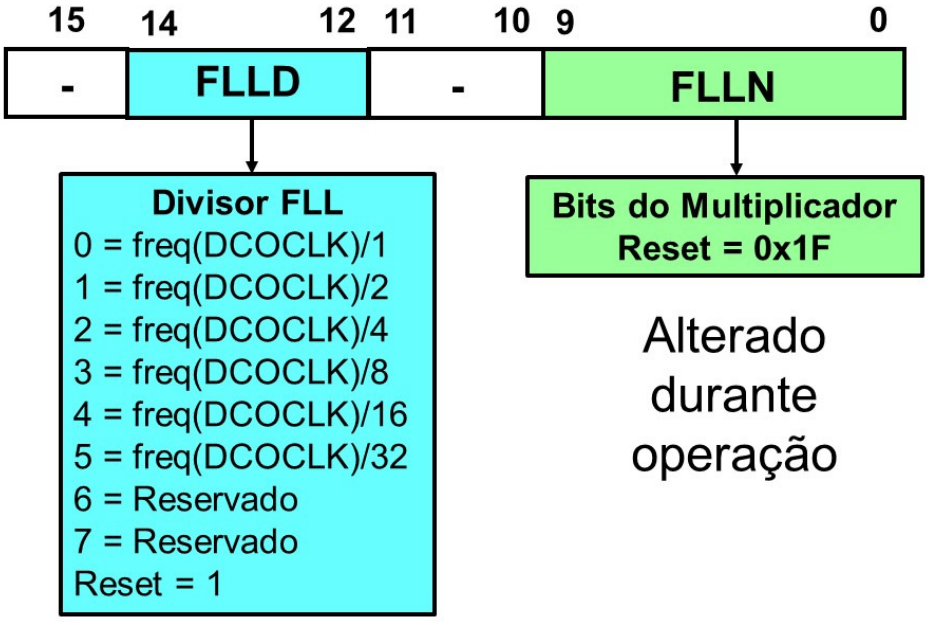
UCSCTL1 → Controle 1 do UCS



! Ver datasheet específico !

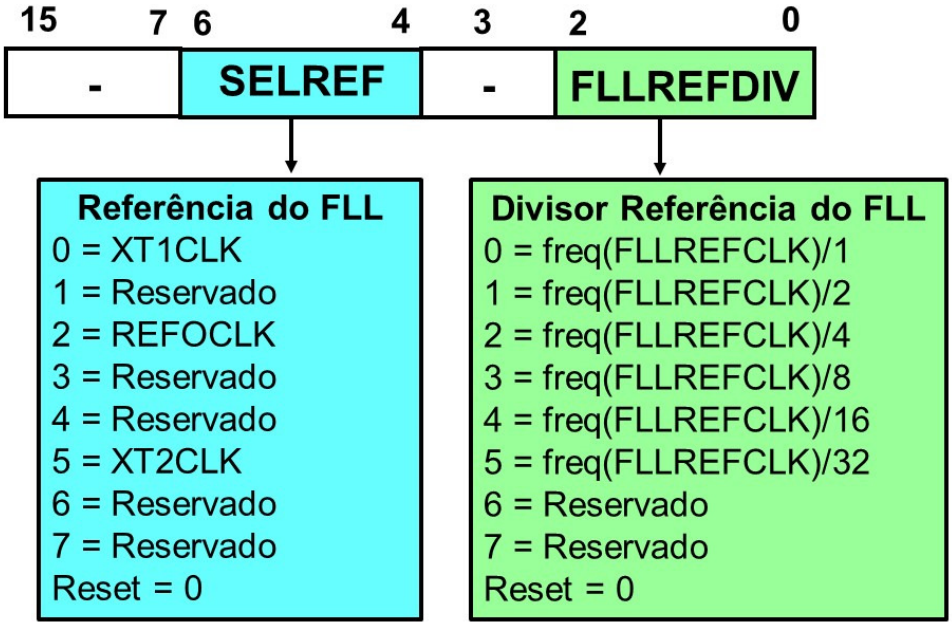
MSP430 – Registradores UCS

UCSCTL2 → Controle 2 do UCS



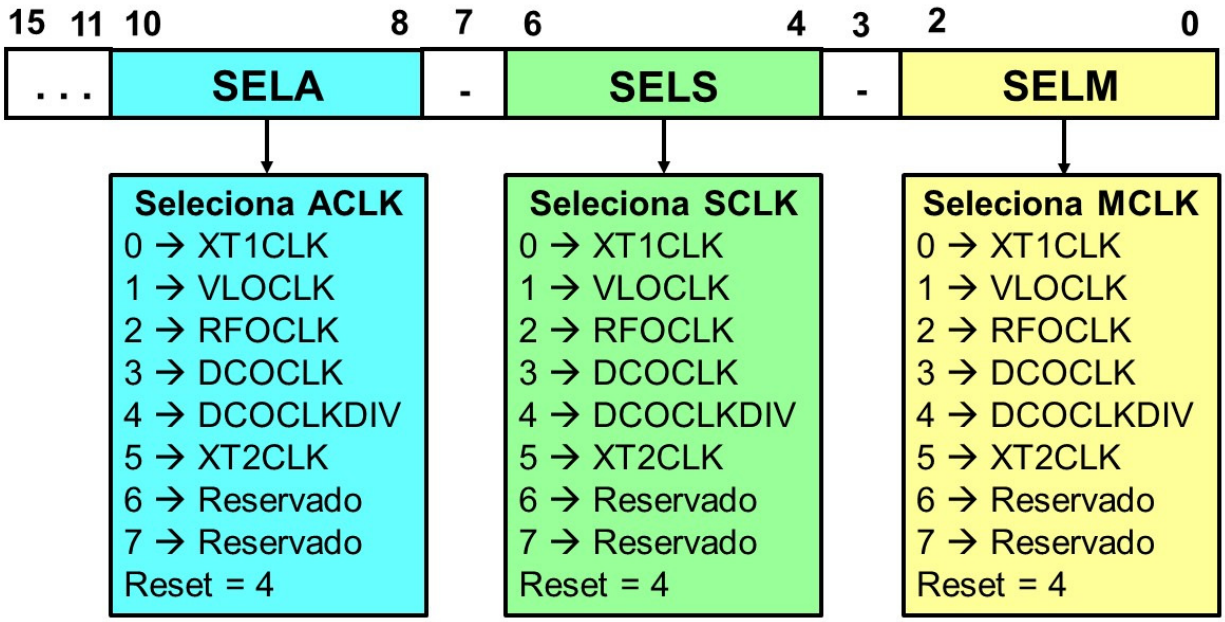
MSP430 – Registradores UCS

UCSCTL3 → Controle 3 do UCS



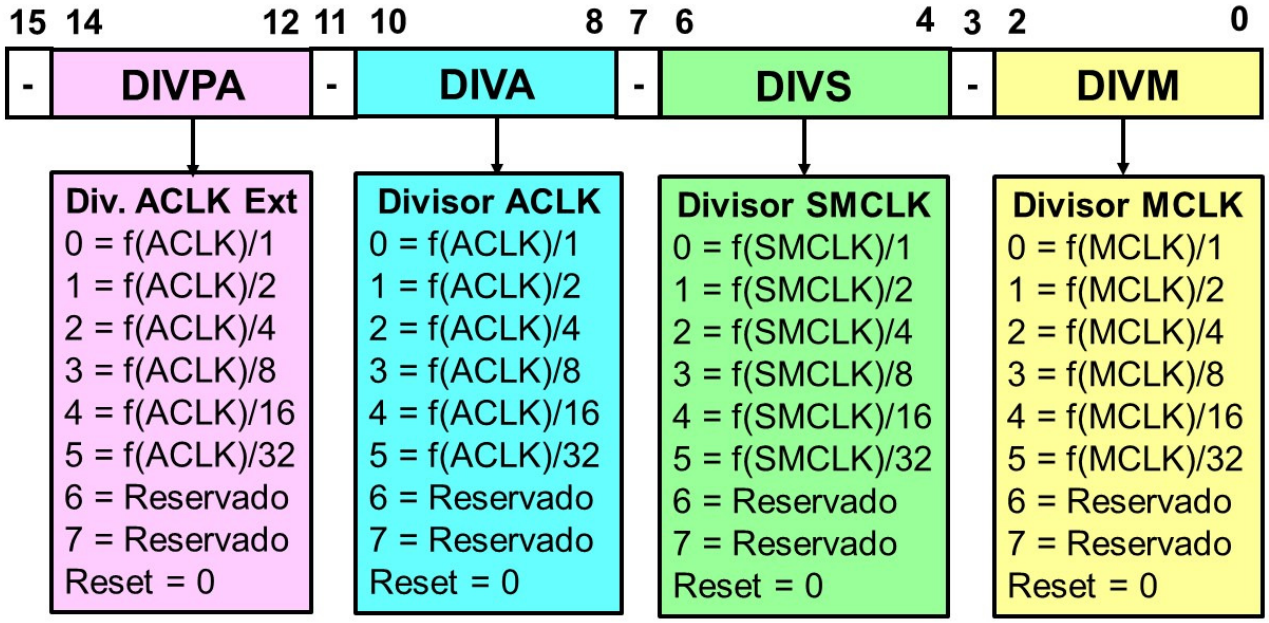
MSP430 – Registradores UCS

UCSCTL4 → Controle 4 do UCS (Repetido)



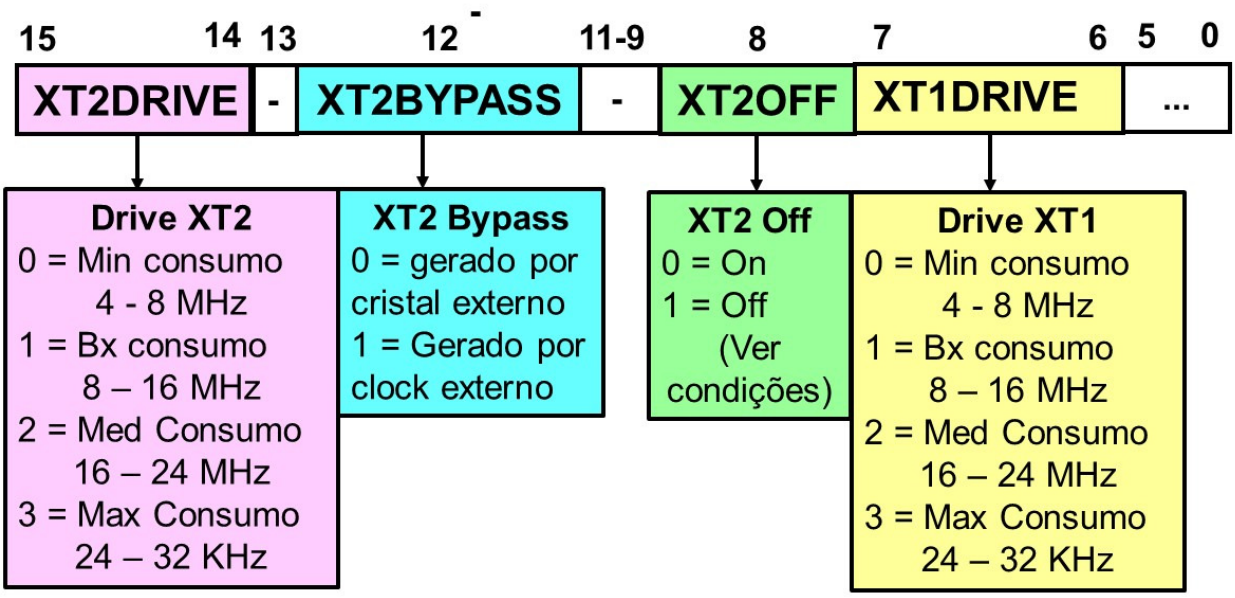
MSP430 – Registradores UCS

UCSCTL5 → Controle 5 do UCS



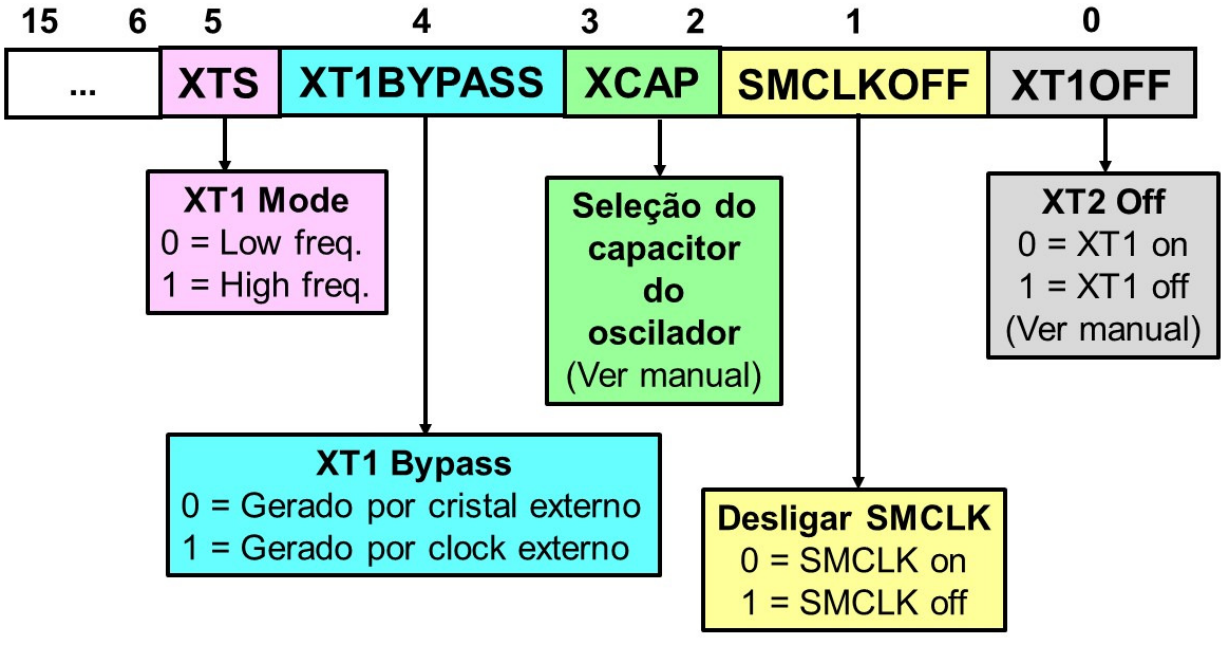
MSP430 – Registradores UCS

UCSCTL6 → Controle 6 do UCS



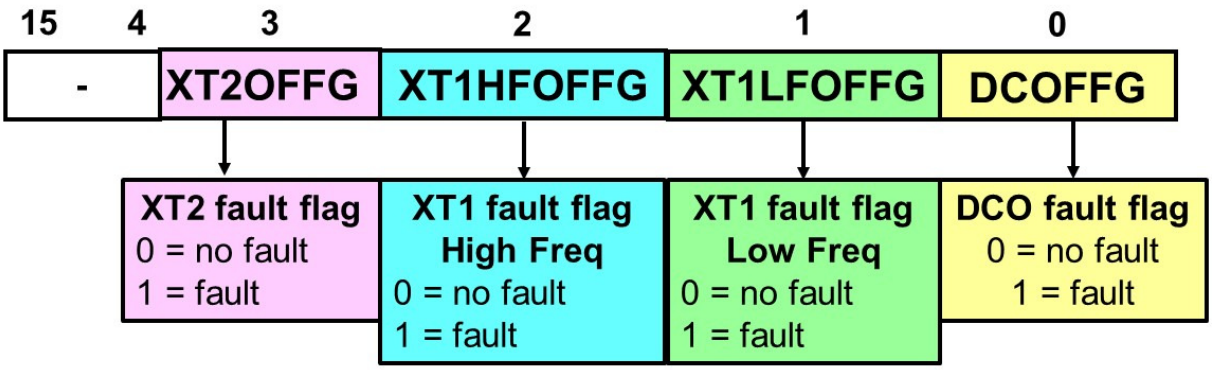
MSP430 – Registradores UCS

UCSCTL6 → Controle 6 do UCS



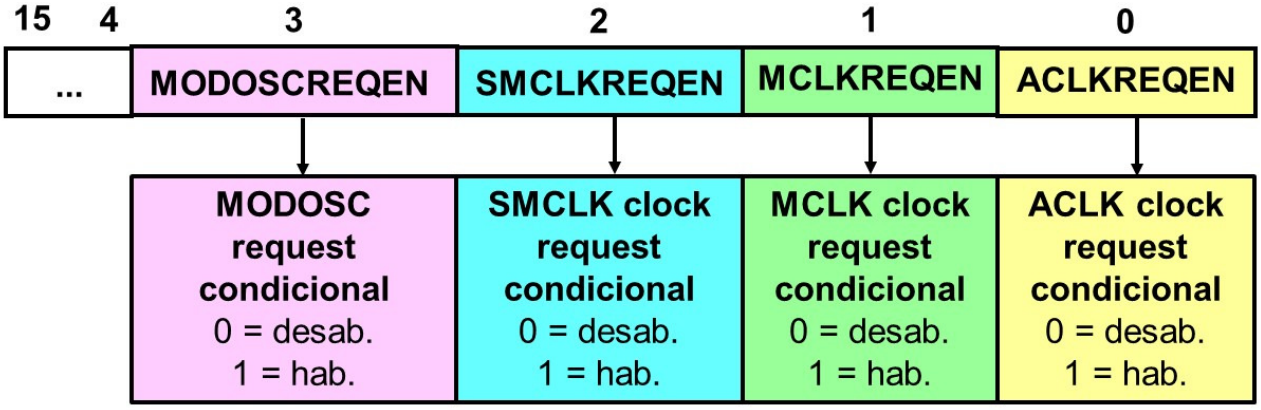
MSP430 – Registradores UCS

UCSCTL7 → Controle 7 do UCS



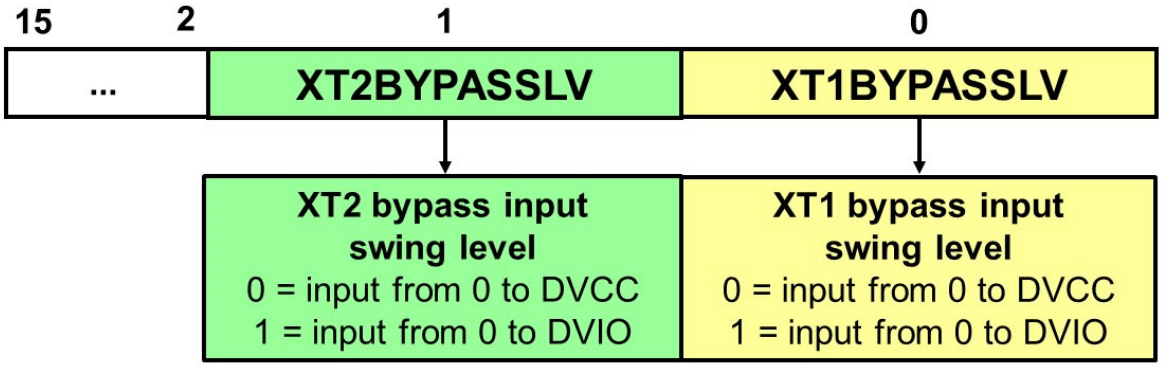
MSP430 – Registradores UCS

UCSCTL8 → Controle 8 do UCS

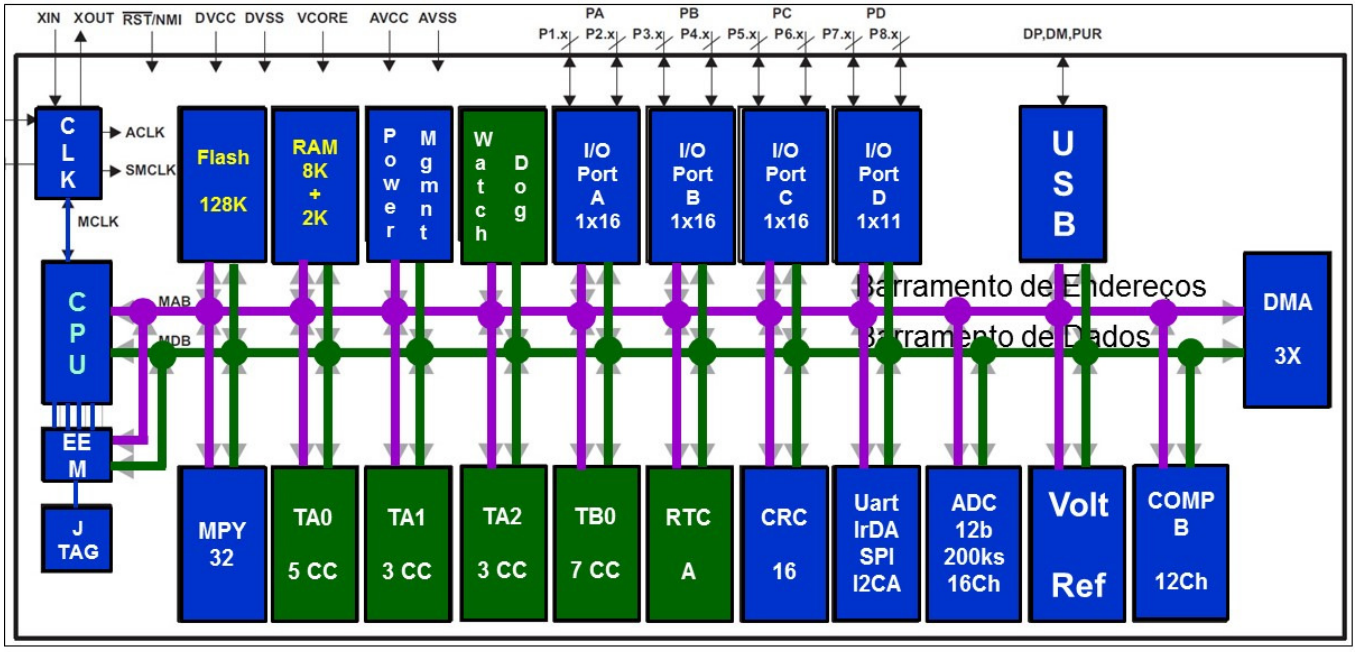


MSP430 – Registradores UCS

UCSCTL9 → Controle 9 do UCS



MSP430 – Diagrama de Blocos



EEM = Embedded Emulation Module

MSP430 – F5529 Timers

Vários Temporizadores disponíveis:

Timer A com 3 instâncias → TA0 com 5 Comparadores; TA1 com 3 Comparadores; TA2 com 3 Comparadores

Timer B com 1 instância → TB0 com 7 Comparadores

Real-time clock 1 instância → RTC_A

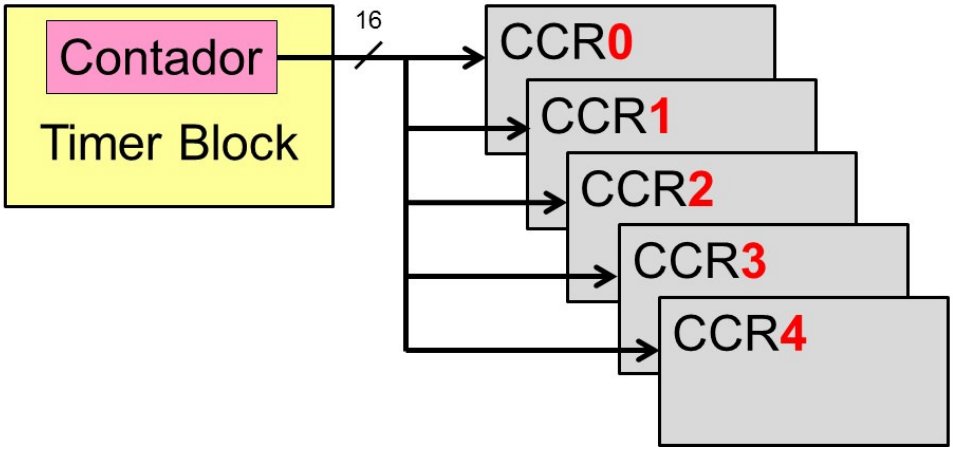
Watchdog timer 1 instância → WDT_A

MSP430 – F5529 Timers

Nomenclatura

TAx → Instância x do Timer A x = 0, 1, 2, ...

CCRn → Instância n do Capture/Compare Register n = 0, 1, 2, ...

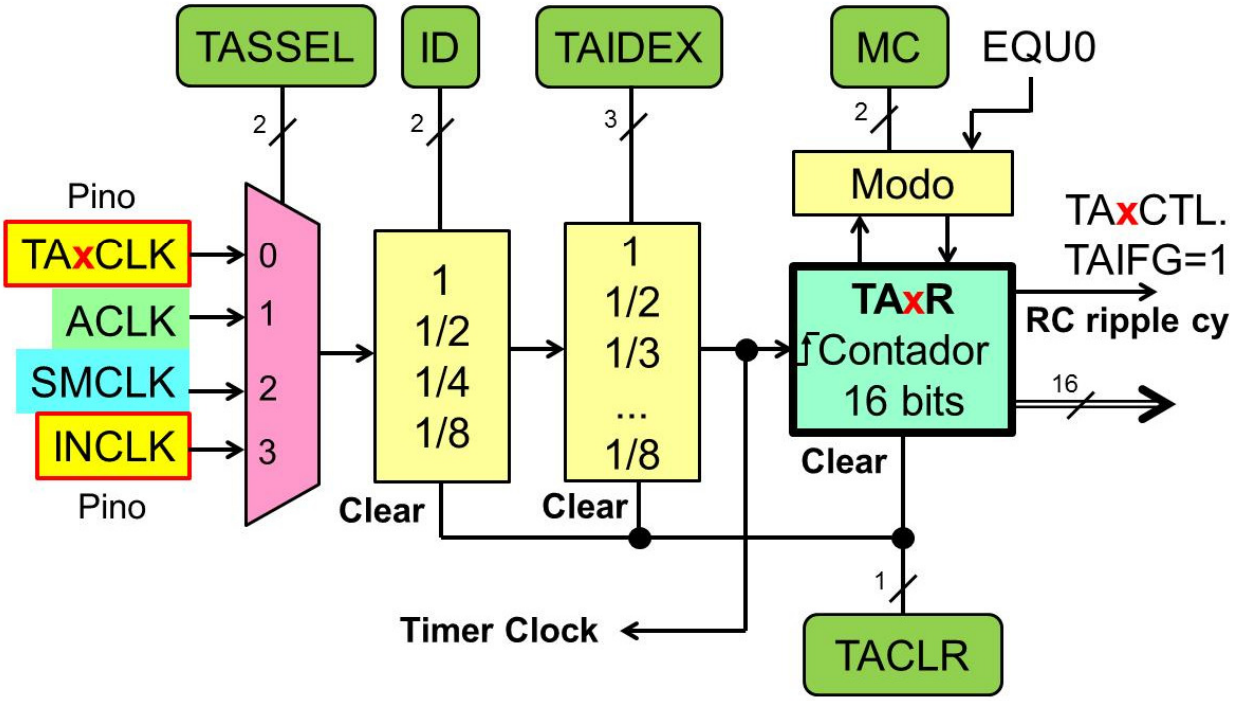


MSP430 – F5529 Timers

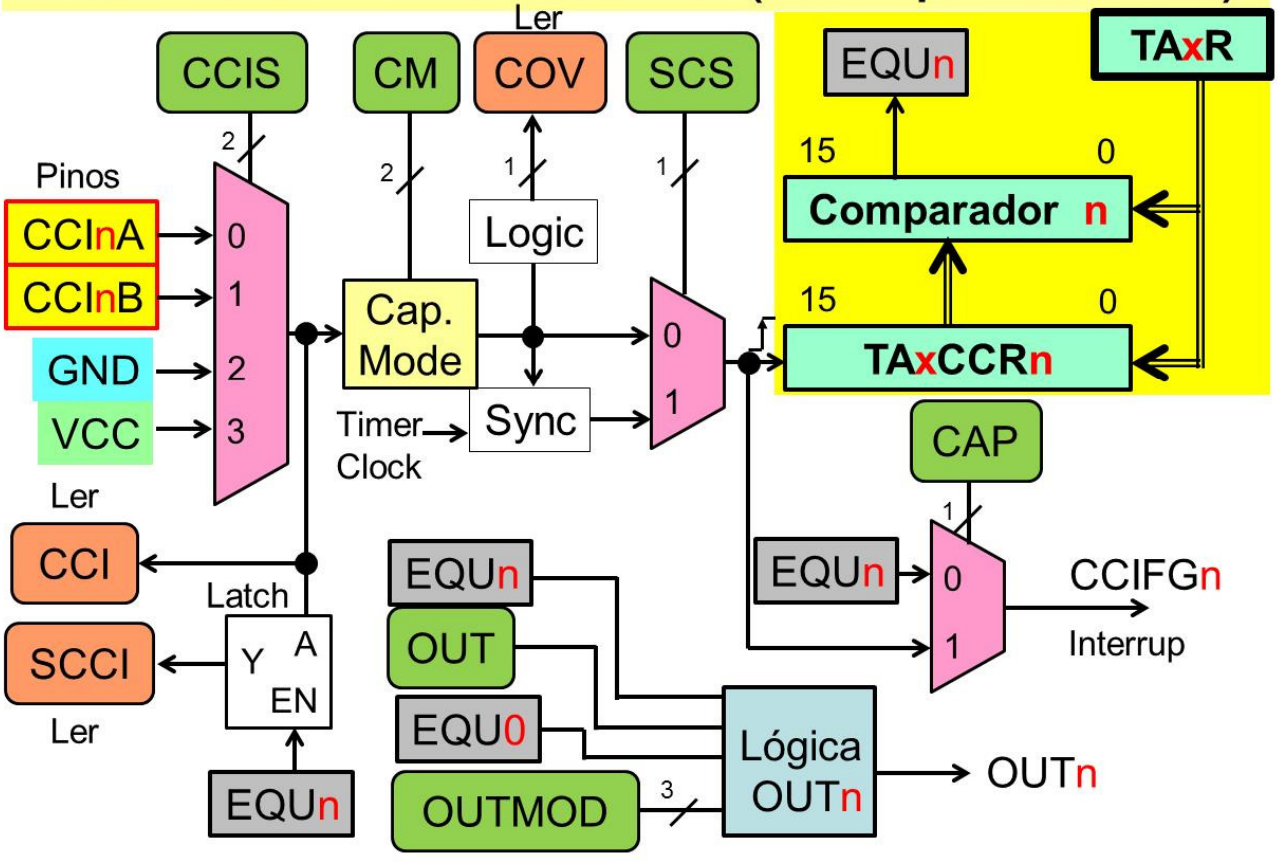
TA0	TA1	TA2	TB0
Instância 0 do Timer A	Instância 1 do Timer A	Instância 2 do Timer A	Instância 0 do Timer B
TA0R	TA1R	TA2R	TB0R
TA0CCR0	TA1CCR0	TA2CCR0	TB0CCR0
TA0CCR1	TA1CCR1	TA2CCR1	TB0CCR1
TA0CCR2	TA1CCR2	TA2CCR2	TB0CCR2
TA0CCR3			TB0CCR3
TA0CCR4			TB0CCR4
			TB0CCR5
			TB0CCR6

MSP430 – Bloco TAx (instância x)

Seleção do clock para TAxR

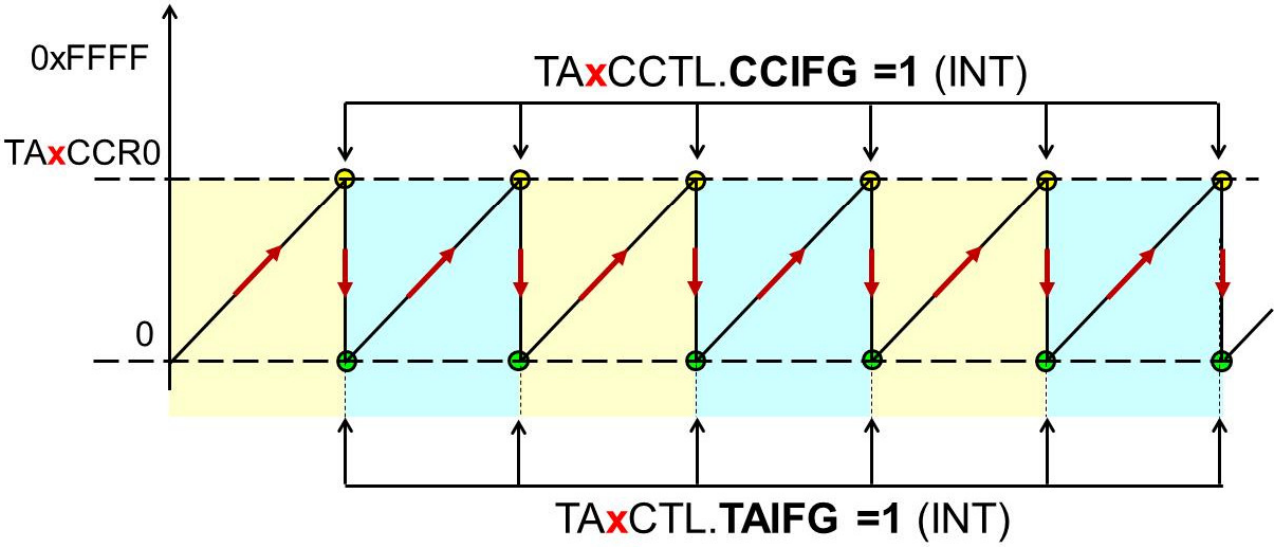


MSP430 – TAx CCRn (Comparador n)



MSP430 –Timer A

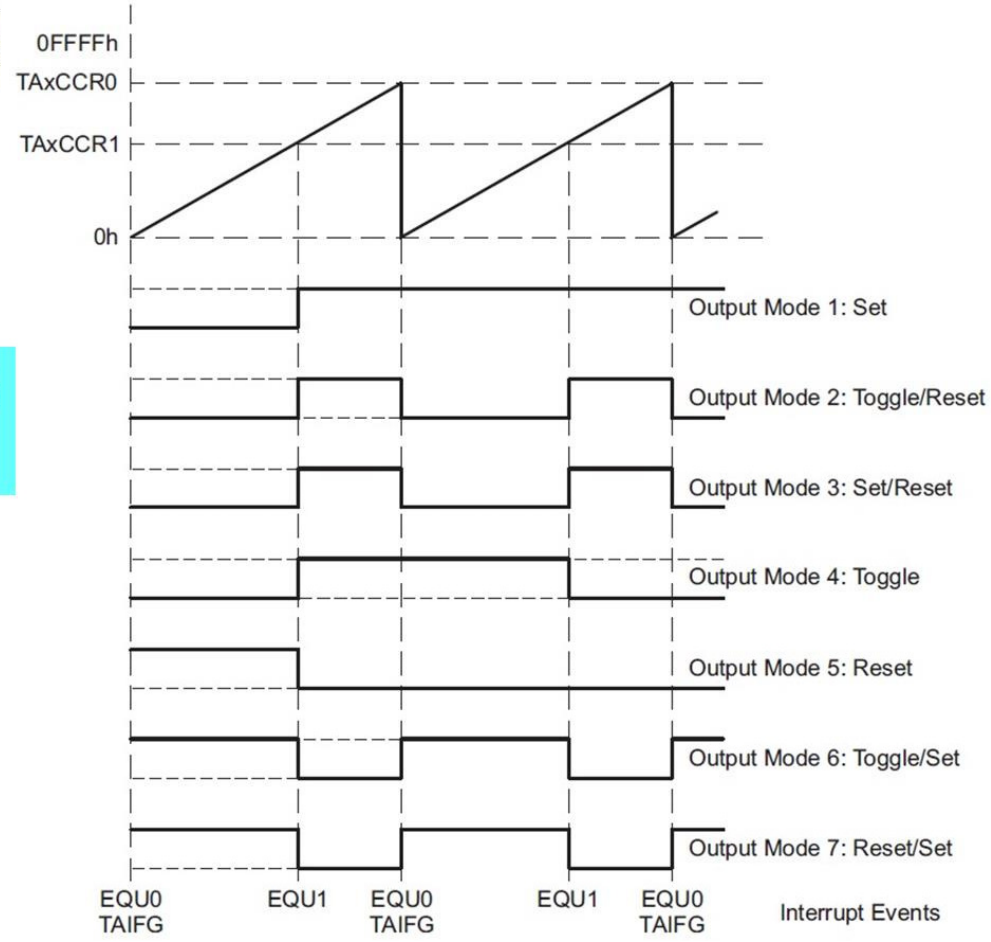
MC = 1 → Modo Up



Contagens por período = TAxCCR0 + 1

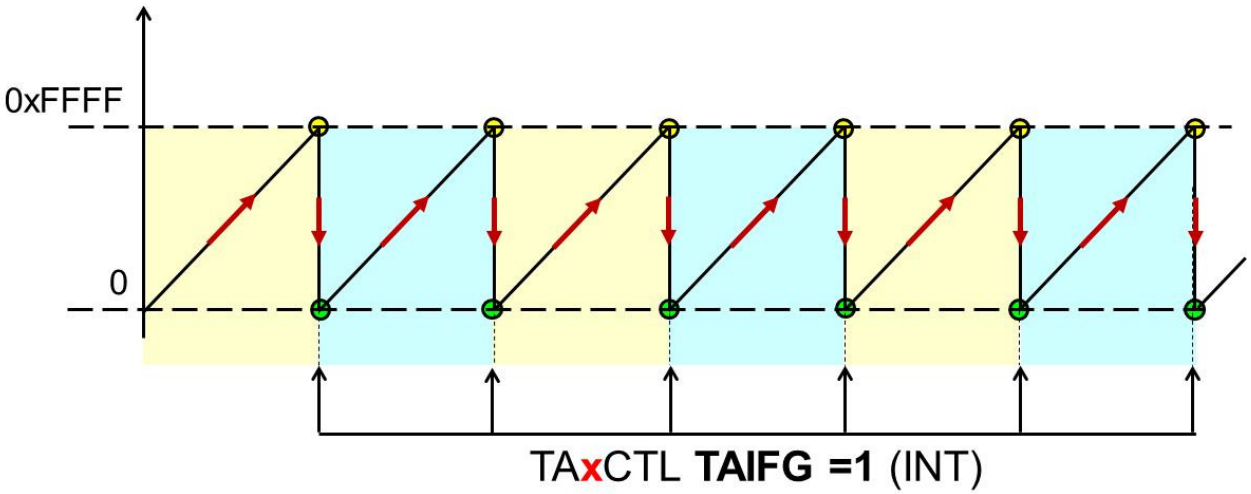
MS

Modo UP



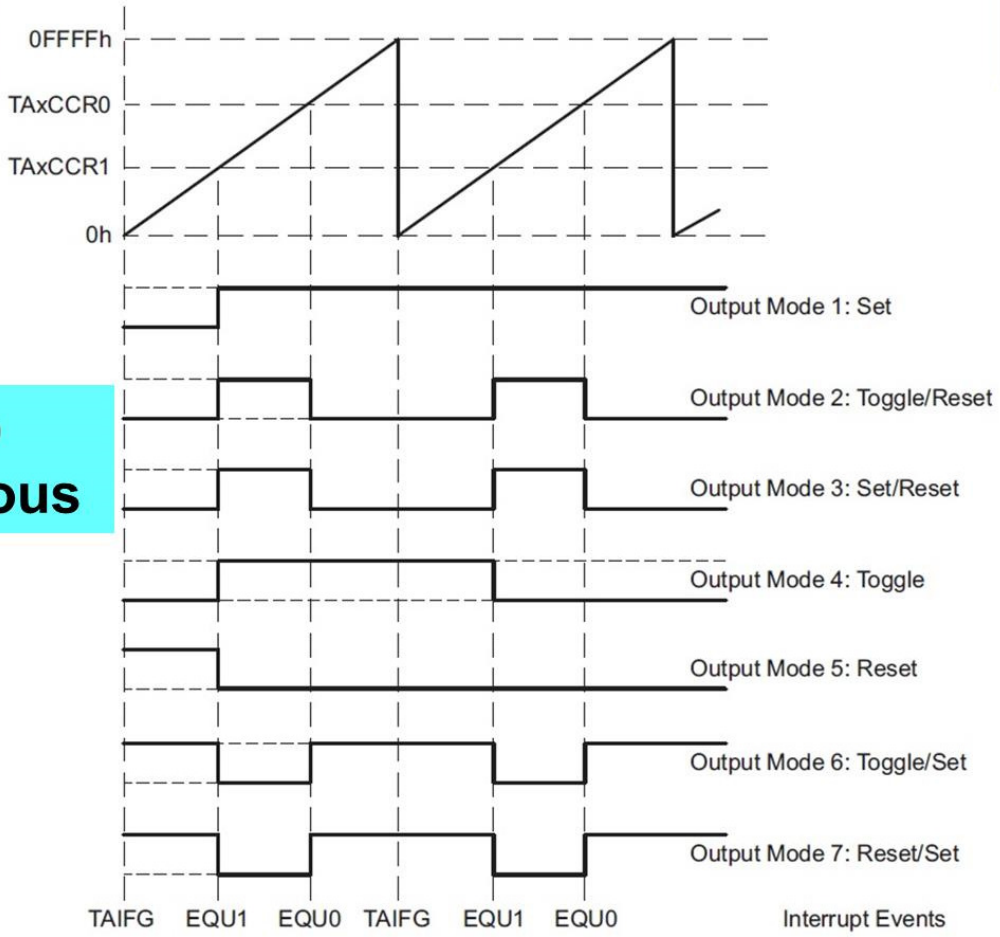
MSP430 –Timer A

MC = 2 → Modo Continuous



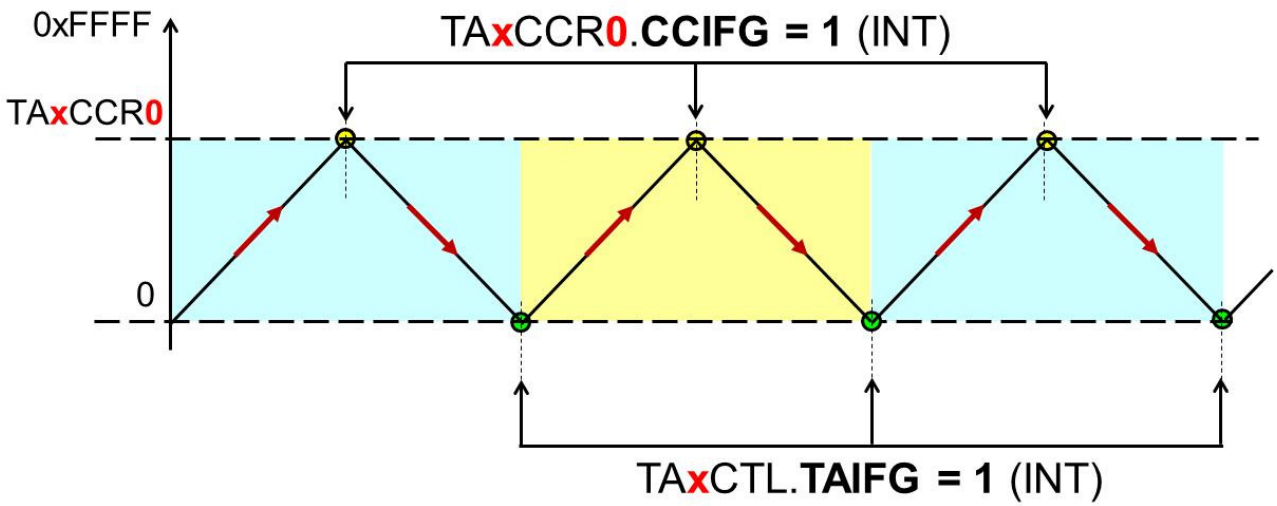
MSP4

Modo Continuous



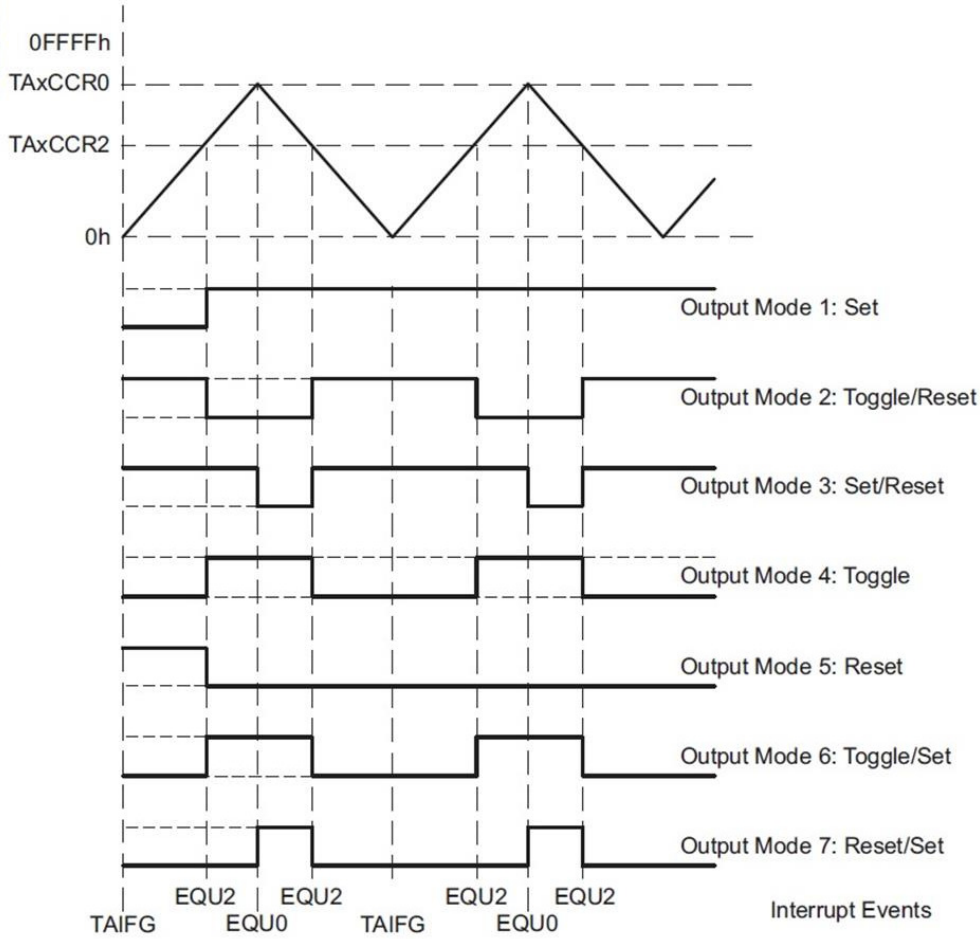
MSP430 –Timer A

MC = 3 → Modo Up/Down



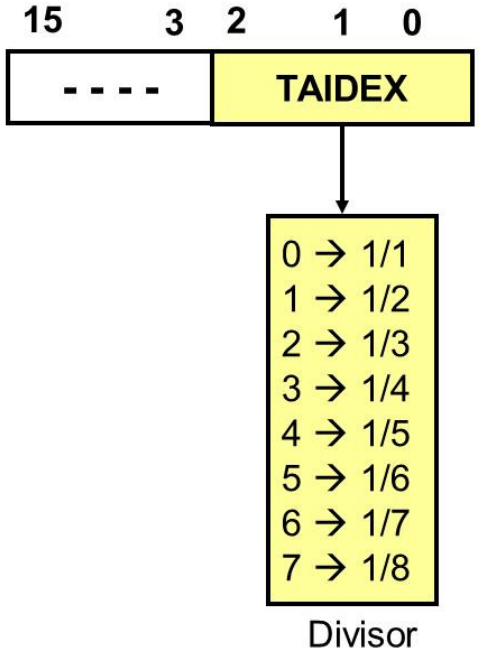
MSP43

Modo Up Down



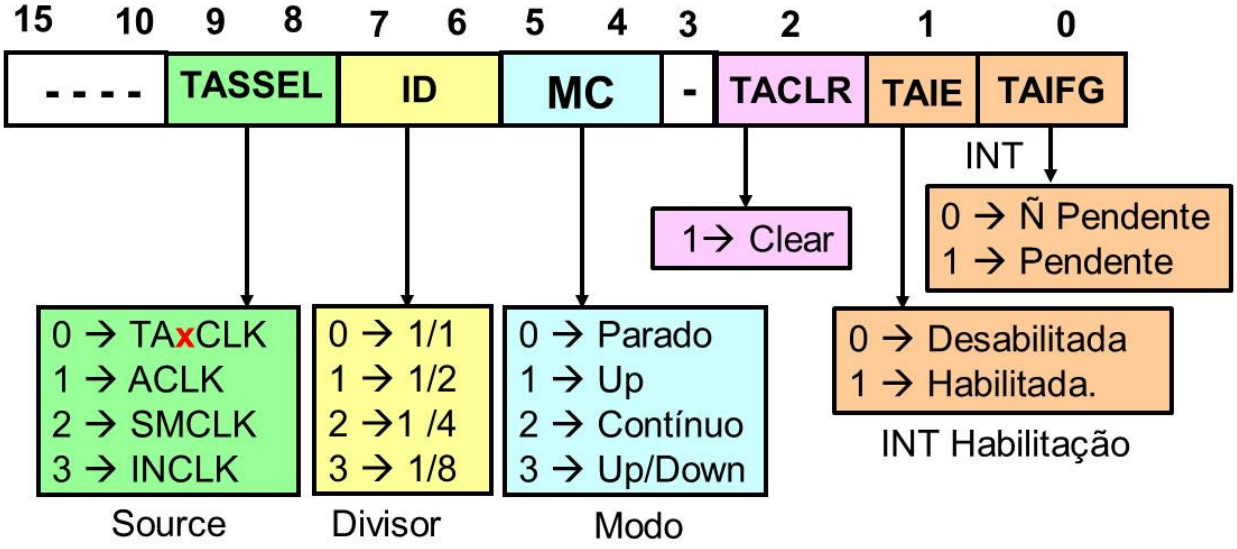
MSP430 – Registradores TAx

TAxEX0



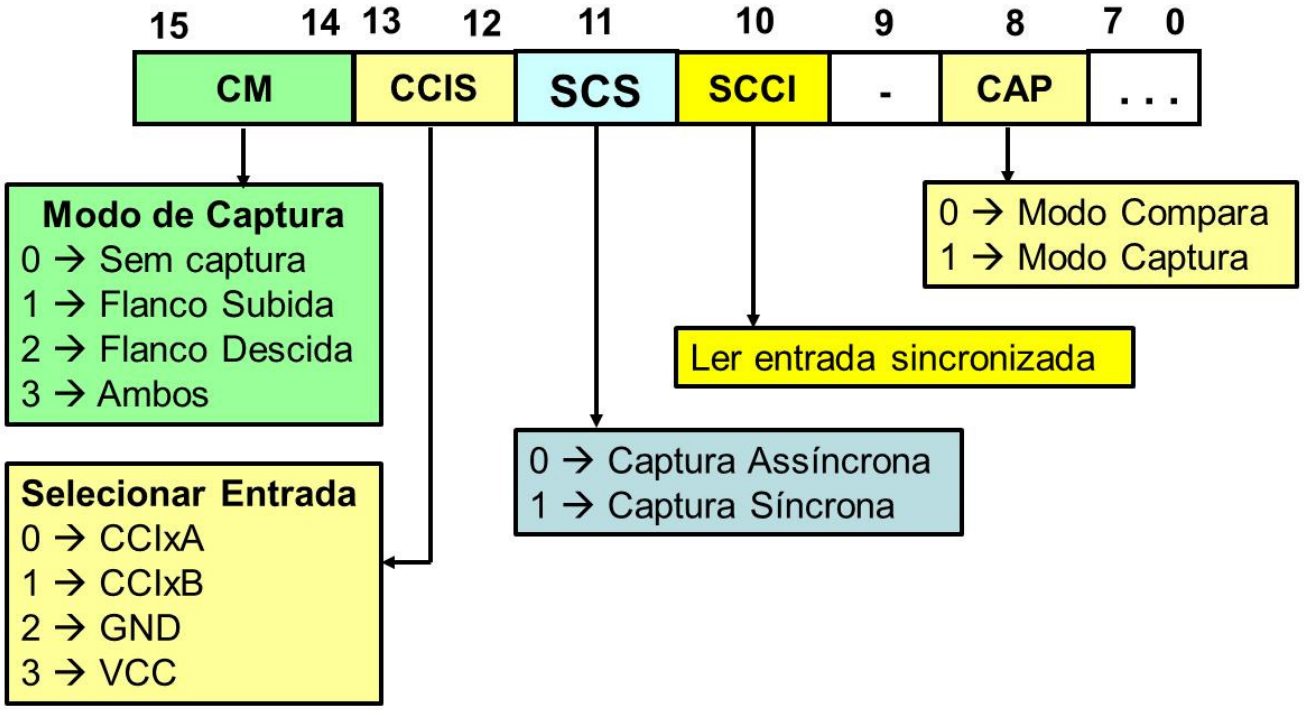
MSP430 – Registradores TAx

TAxCTL → Controle do Timer Ax



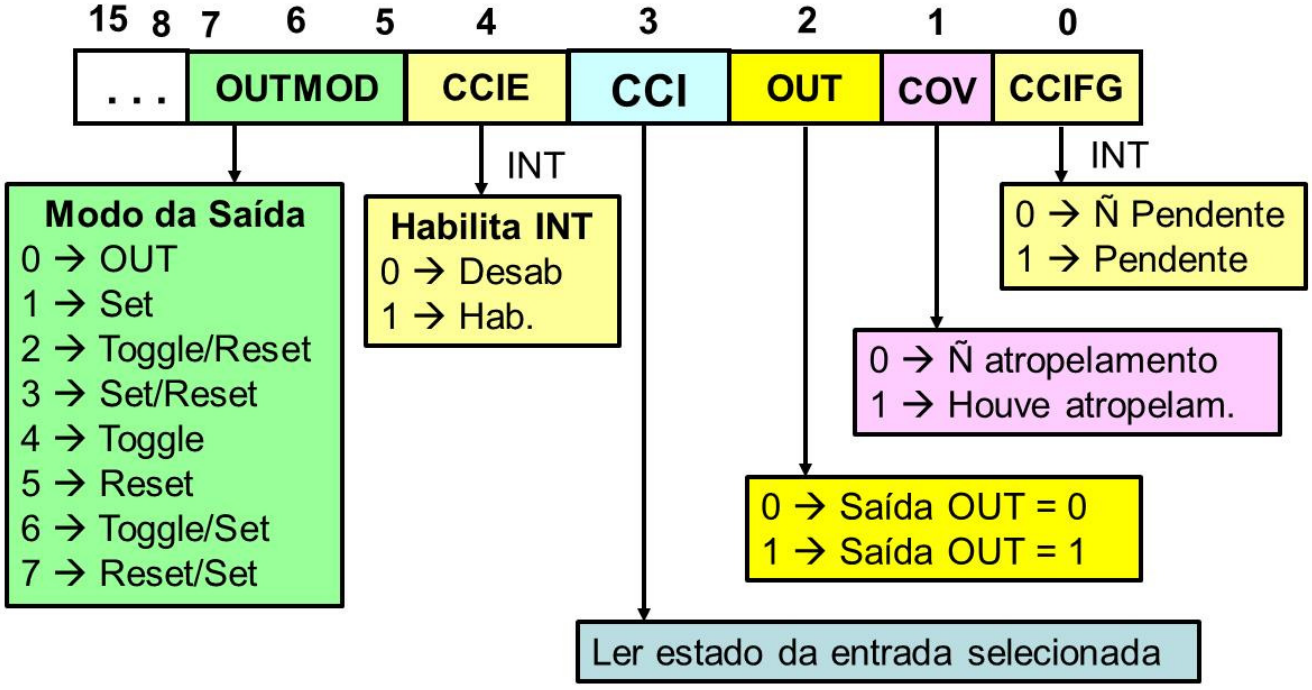
MSP430 – Registradores TAx

TAxCCTLn → Controle do Comparador/Captura n



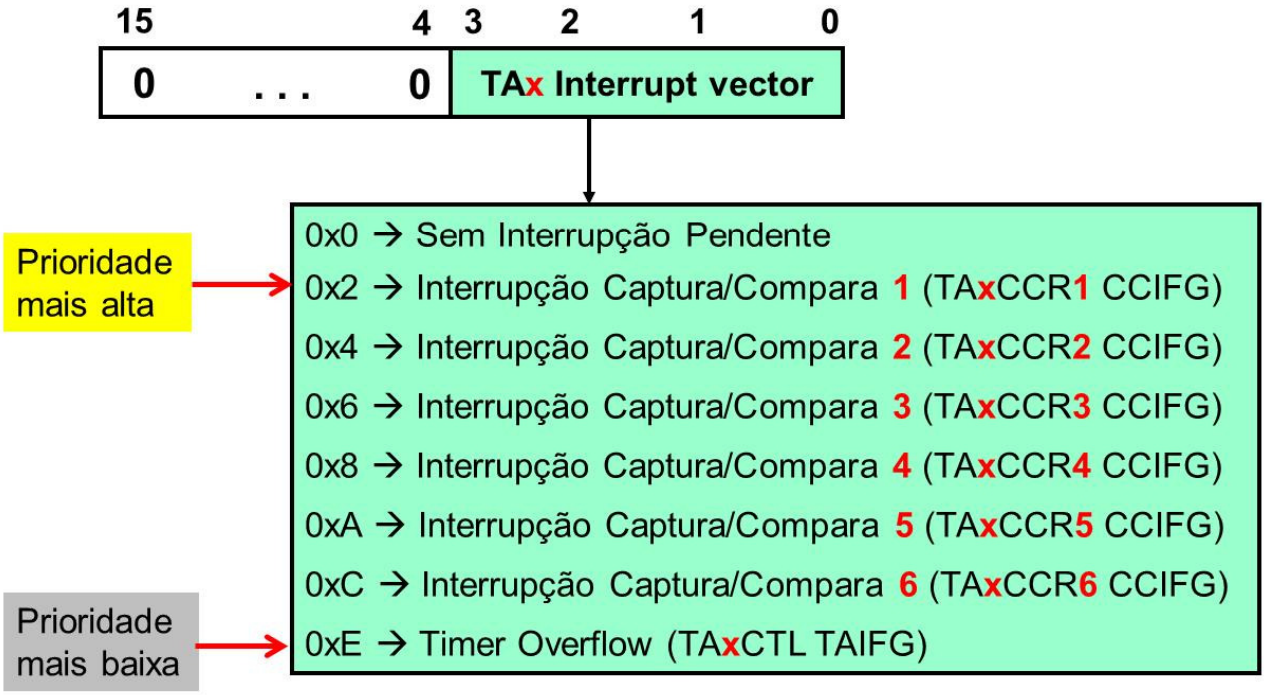
MSP430 – Registradores TAx

TAxCCTLn → Controle do Comparador/Captura n

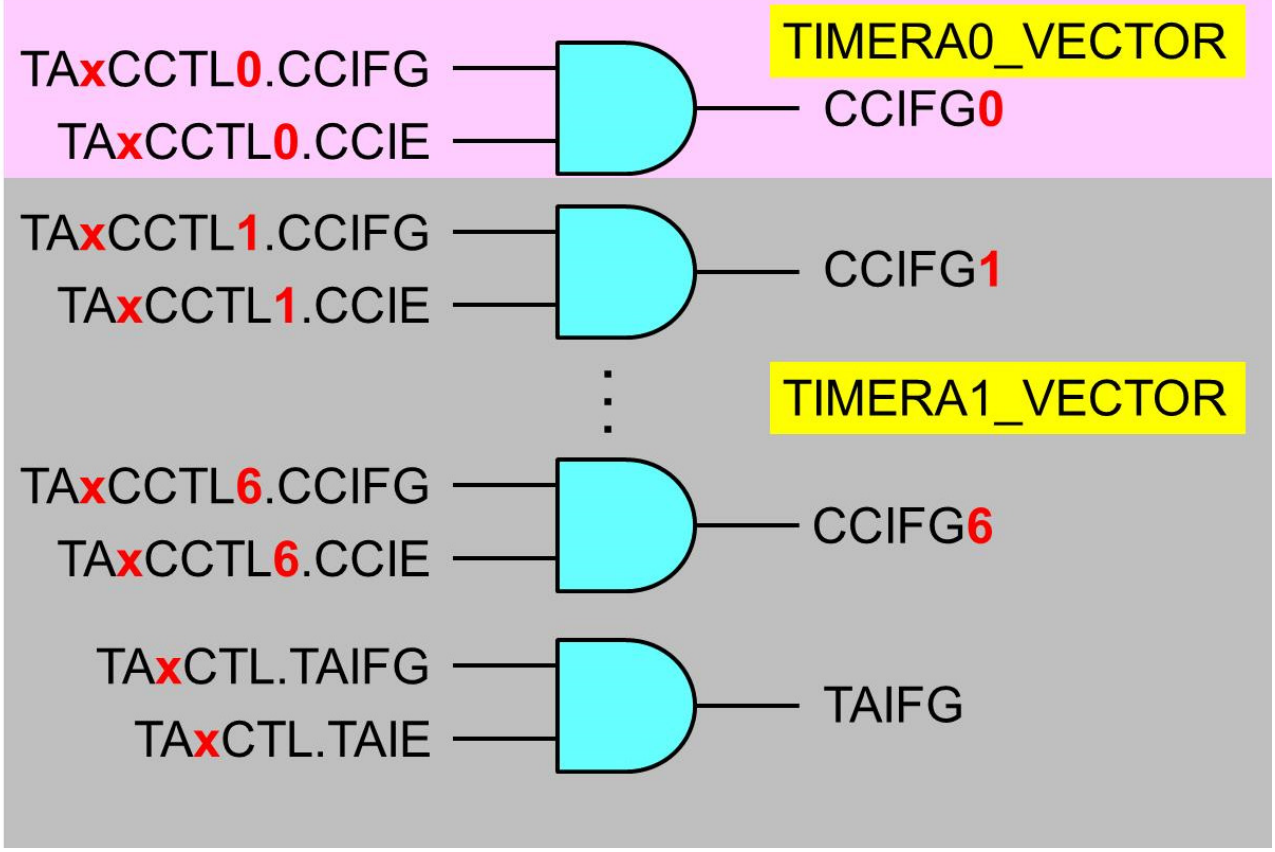


MSP430 – Registradores TAx

TAxIV → Vetor de Interrupção do Timer Ax



MSP430 – Interrupções TAx



```
; Interrupt handler for TA0CCR0 CCIFG.                          Cycle
CCIFG_0_HND
;           ...           ; Start of handler Interrupt latency    6
            RETI                                                  5

; Interrupt handler for TA0IFG, TA0CCR1 through TA0CCR6 CCIFG.

TA0_HND     ...                     ; Interrupt latency           6
            ADD       &TA0IV,PC     ; Add offset to Jump table    3
            RETI                    ; Vector  0: No interrupt     5
            JMP       CCIFG_1_HND   ; Vector  2: TA0CCR1          2
            JMP       CCIFG_2_HND   ; Vector  4: TA0CCR2          2
            JMP       CCIFG_3_HND   ; Vector  6: TA0CCR3          2
            JMP       CCIFG_4_HND   ; Vector  8: TA0CCR4          2
            JMP       CCIFG_5_HND   ; Vector 10: TA0CCR5          2
            JMP       CCIFG_6_HND   ; Vector 12: TA0CCR6          2

TA0IFG_HND                          ; Vector 14: TA0IFG Flag
            ...                     ; Task starts here
            RETI                                                  5

CCIFG_6_HND                         ; Vector 12: TA0CCR6
            ...                     ; Task starts here
            RETI                    ; Back to main program        5

CCIFG_5_HND                         ; Vector 10: TA0CCR5
            ...                     ; Task starts here
            RETI                    ; Back to main program        5
```

C

```
#pragma vector = PORT2_VECTOR
__interrupt void port2(void){
   int n;
   n = __even_in_range(P2IV,0x10);
   switch(n){
      case 0x0: break;                  case 0x2: break;
      case 0x4: isr_sw1();  break;      case 0x6: break;
      case 0x8: break;                  case 0xA: break;
      case 0xC: break;                  case 0xE: break;
      case 0x10: break;
   }
}

void isr_sw1(void){
   LED1_OUT ^= LED1;
}
```

MSP430 –Interrupção em C

Pragmatic = pragmático, prático — Manual slau132o

#pragma vector = xxx_VECTOR Pag 111

- Indica que a função a seguir será usada como ISR.

__interrupt void **port2** (void) Pag 91 e 135 (6.7.2)

- Prepara a função para ser usada como ISR.
- Salva todos os registradores usados.
- Se a ISR chama alguma outra função, então salva todos os registradores da CPU.
- Retorno feito com RETI.

MSP430 –Interrupção em C

__even_in_range (x, num); Pag 111

- Retorna o valor de x.
- x deve ser **par**, na faixa de 0 até **num**.
- Facilitar uso do switch para vetores interrupção.

__enable_interrupt ( ); ou **__enable_interrupts ( )**

- Corresponde à instrução EINT. Pag 137

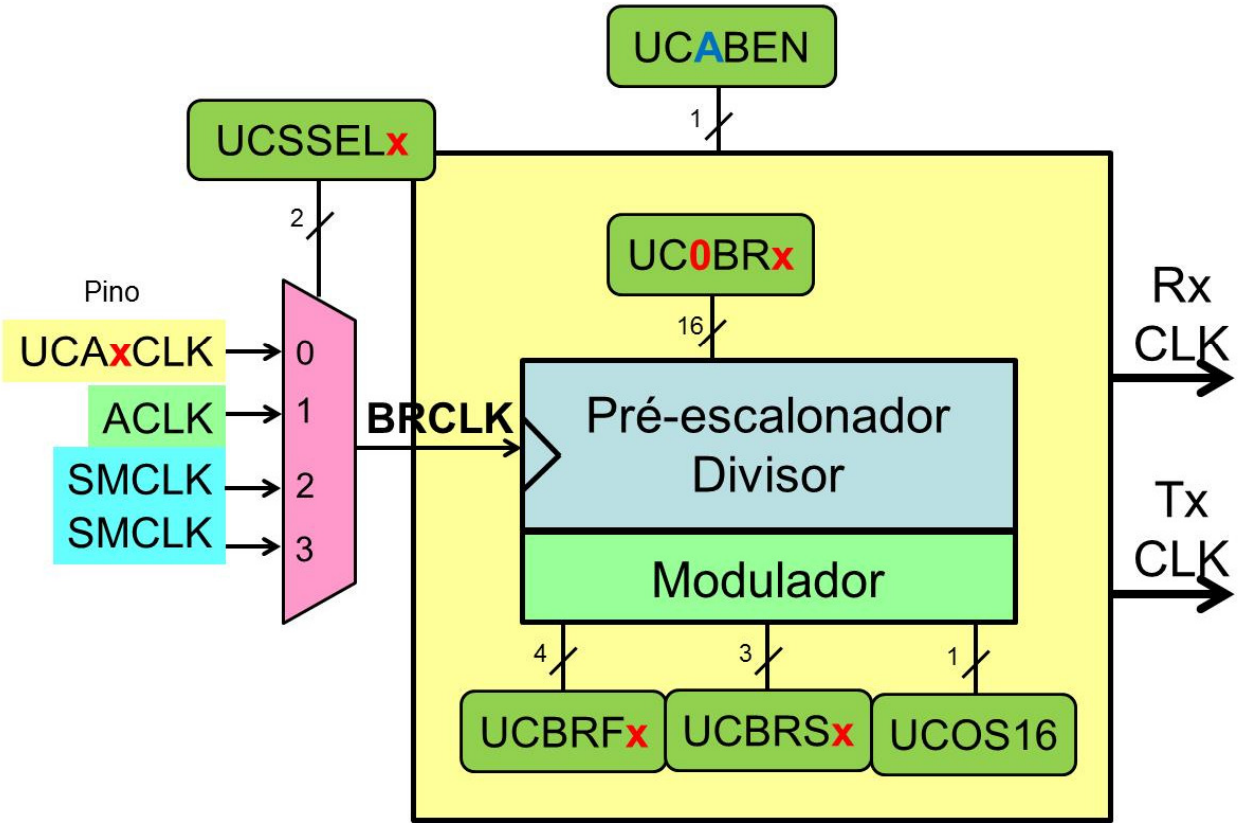
__disable_interrupt ( ); ou **__disable_interrupts ( )**

- Corresponde à instrução DINT. Pag 136

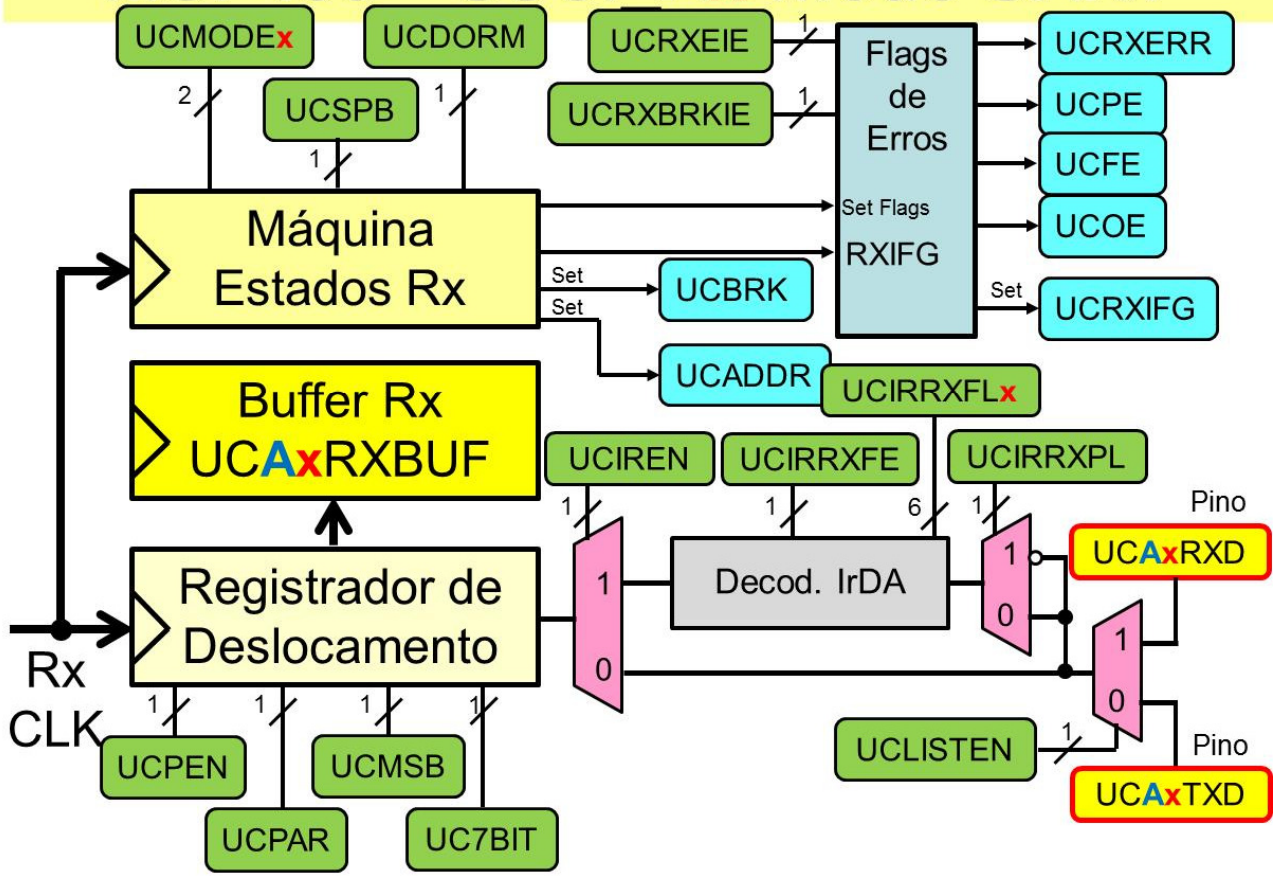
Table 4. Interrupt Sources, Flags, and Vectors

INTERRUPT SOURCE	INTERRUPT FLAG	SYSTEM INTERRUPT	WORD ADDRESS	PRIORITY
System Reset Power-Up External Reset Watchdog Timeout, Password Violation Flash Memory Password Violation	WDTIFG, KEYV (SYSRSTIV)[1][2]	Reset	0FFFEh	63, highest
System NMI PMM Vacant Memory Access JTAG Mailbox	SVMLIFG, SVMHIFG, DLYLIFG, DLYHIFG, VLRLIFG, VLRHIFG, VMAIFG, JMBNIFG, JMBOUTIFG (SYSSNIV)[1]	(Non)maskable	0FFFCh	62
User NMI NMI Oscillator Fault Flash Memory Access Violation	NMIIFG, OFIFG, ACCVIFG, BUSIFG (SYSUNIV)[1][2]	(Non)maskable	0FFFAh	61
Comp_B	Comparator B interrupt flags (CBIV)[1][3]	Maskable	0FFF8h	60
TB0	TB0CCR0 CCIFG0[3]	Maskable	0FFF6h	59
TB0	TB0CCR1 CCIFG1 to TB0CCR6 CCIFG6, TB0IFG (TB0IV)[1][3]	Maskable	0FFF4h	58
Watchdog Timer_A Interval Timer Mode	WDTIFG	Maskable	0FFF2h	57
USCI_A0 Receive or Transmit	UCA0RXIFG, UCA0TXIFG (UCA0IV)[1][3]	Maskable	0FFF0h	56
USCI_B0 Receive or Transmit	UCB0RXIFG, UCB0TXIFG (UCB0IV)[1][3]	Maskable	0FFEEh	55
ADC12_A	ADC12IFG0 to ADC12IFG15 (ADC12IV)[1][3][4]	Maskable	0FFECh	54
TA0	TA0CCR0 CCIFG0[3]	Maskable	0FFEAh	53
TA0	TA0CCR1 CCIFG1 to TA0CCR4 CCIFG4, TA0IFG (TA0IV)[1][3]	Maskable	0FFE8h	52
USB_UBM	USB interrupts (USBIV)[1][3]	Maskable	0FFE6h	51
DMA	DMA0IFG, DMA1IFG, DMA2IFG (DMAIV)[1][3]	Maskable	0FFE4h	50
TA1	TA1CCR0 CCIFG0[3]	Maskable	0FFE2h	49
TA1	TA1CCR1 CCIFG1 to TA1CCR2 CCIFG2, TA1IFG (TA1IV)[1][3]	Maskable	0FFE0h	48
I/O Port P1	P1IFG.0 to P1IFG.7 (P1IV)[1][3]	Maskable	0FFDEh	47
USCI_A1 Receive or Transmit	UCA1RXIFG, UCA1TXIFG (UCA1IV)[1][3]	Maskable	0FFDCh	46
USCI_B1 Receive or Transmit	UCB1RXIFG, UCB1TXIFG (UCB1IV)[1][3]	Maskable	0FFDAh	45
TA2	TA2CCR0 CCIFG0[3]	Maskable	0FFD8h	44
TA2	TA2CCR1 CCIFG1 to TA2CCR2 CCIFG2, TA2IFG (TA2IV)[1][3]	Maskable	0FFD6h	43
I/O Port P2	P2IFG.0 to P2IFG.7 (P2IV)[1][3]	Maskable	0FFD4h	42
RTC_A	RTCRDYIFG, RTCTEVIFG, RTCAIFG, RT0PSIFG, RT1PSIFG (RTCIV)[1][3]	Maskable	0FFD2h	41
Reserved	Reserved[5]		0FFD0h	40
			⋮	⋮
			0FF80h	0, lowest

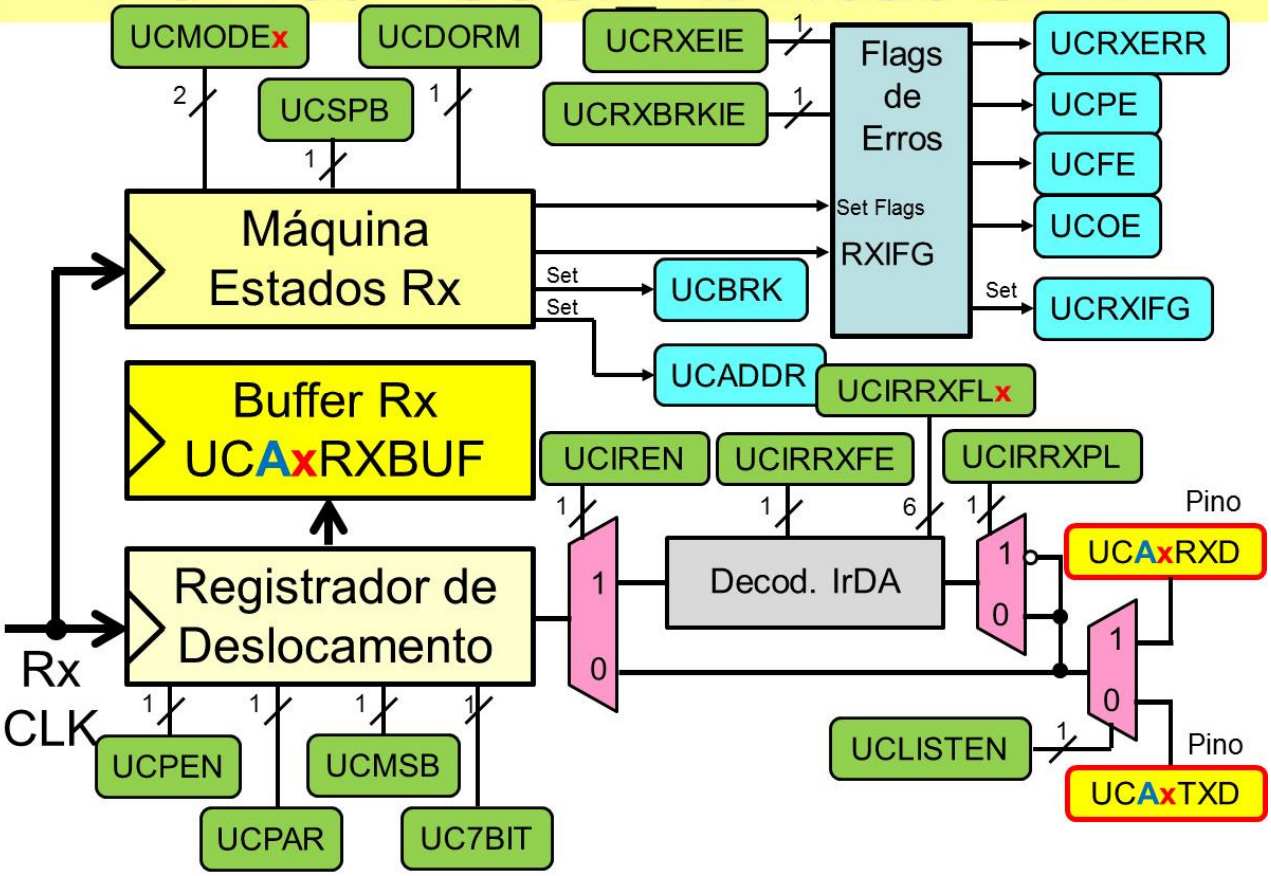
MSP430 – USCI_Ax modo UART



MSP430 – USCI_Ax modo UART



MSP430 – USCI_Ax modo UART



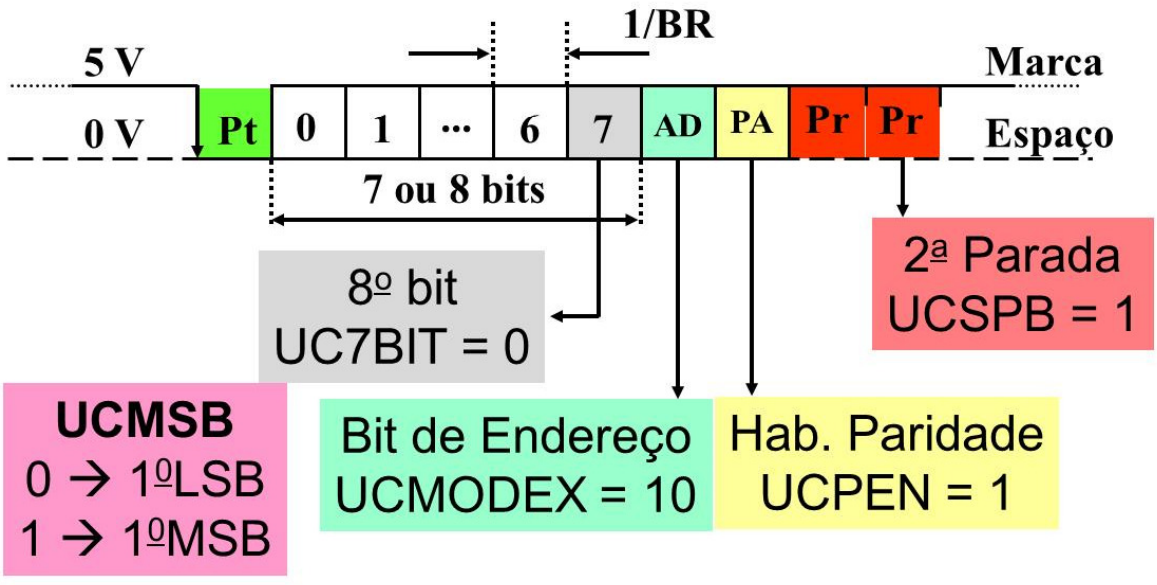
MSP430 – USCI UART - Formato

Pt = Bit de Partida

Pr = Bit de Parada

AD = Bit de Endereço

PA = Bit de Paridade



MSP430 – USCI UART - Inicialização

1) **UCSWRST = 1** → Reset

2) Inicializar registradores e Baudrate.

3) Configurar as portas (saída ou entrada).

4) **UCSWRST = 0** → Sair do Reset

5) Se for o caso, hab. Interrupções UCRXIE e UCTXIE.

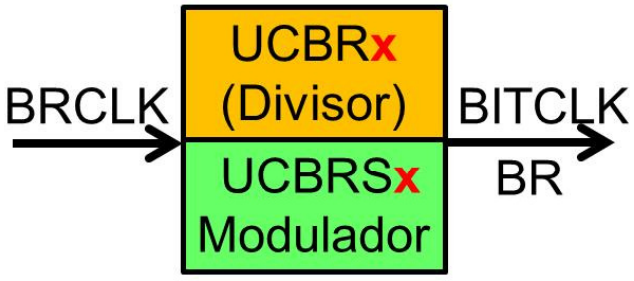
Dado a Transmitir → UCA**TX**BUF → UC**TX**IFG

Dado Recebido ← UCA**RX**BUF UC**RX**IFG ←

MSP430 – USCI Geração de BR

Modo Baixa Frequência: UCOS = 0

RESUMO



Máximo BITCLK:
BITCLK = BRCLK / 3

N = BRCLK / BR

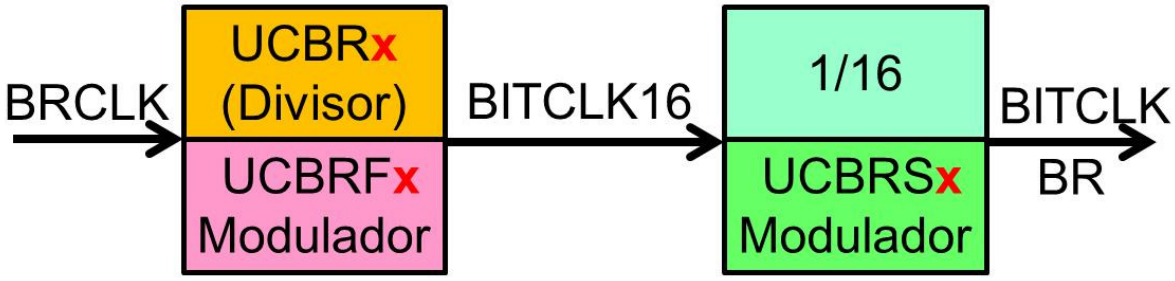
UCBRx = INT (N)

UCBRSx = ROUND [(N – INT(N)) x 8]

MSP430 – USCI Geração de BR

Modo Super Amostragem: UCOS = 1

RESUMO



BITCLK16 = 16 x BR

N = BITCLK16 / BRCLK

UCBRx = INT (N)

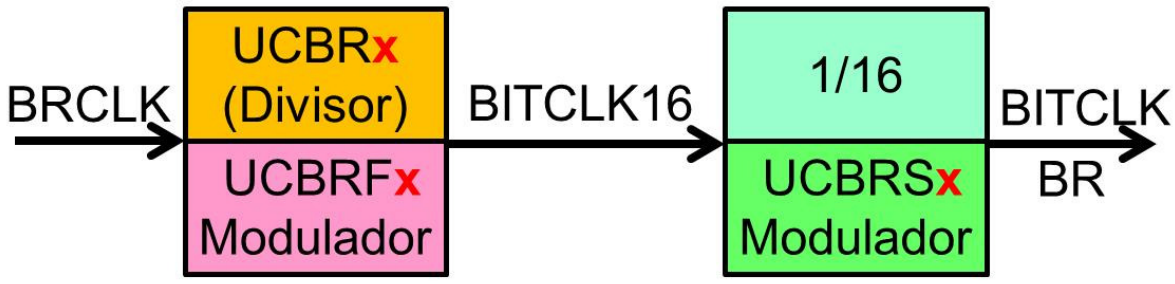
UCBRFx = ROUND [(N – INT(N)) x 16]

Continua ...

MSP430 – USCI Geração de BR

Modo Super Amostragem: UCOS = 1

RESUMO



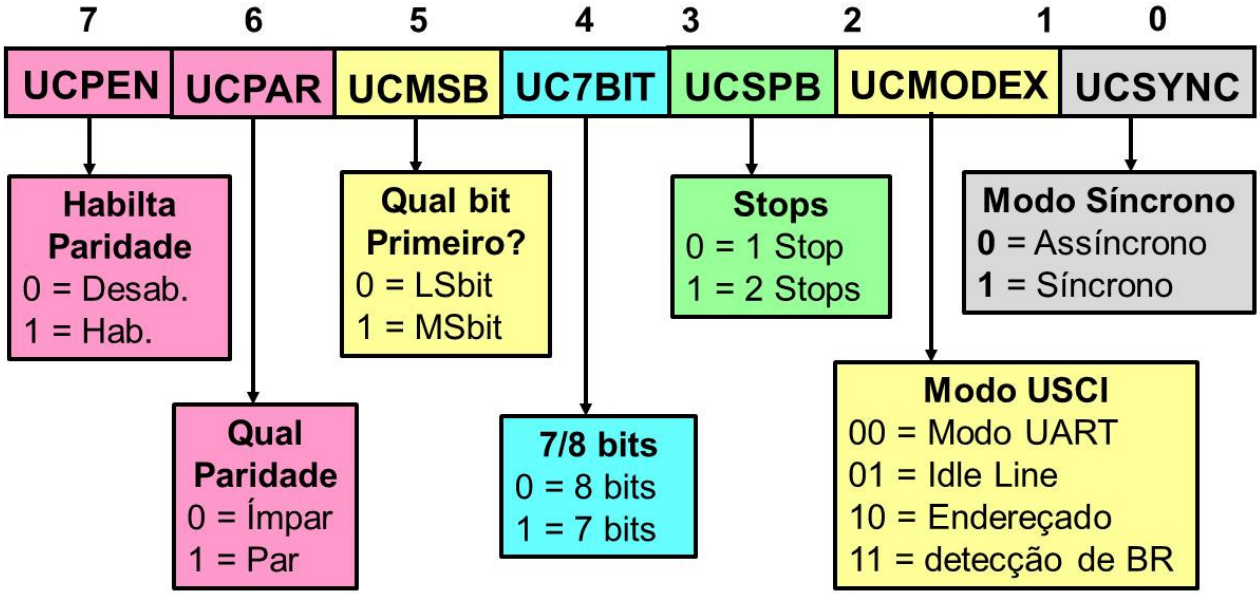
BITCLK16' = BRCLK / (N + UCBRFx / 16)

N' = BITCLK16' / BR → N' = 16,xxxxx

UCBRSx = ROUND [(N' – INT(N')) x 8]

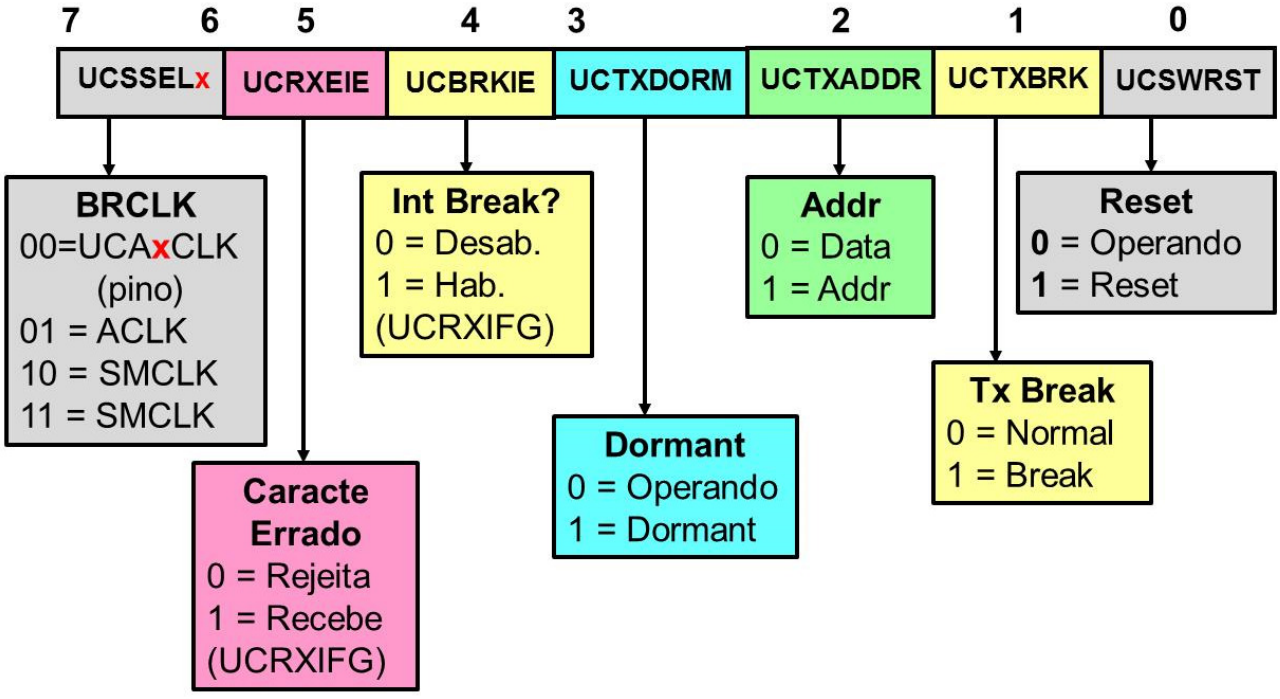
MSP430 – Registradores do USCI

UCAxCTL0 → Controle 0 do USCI_Ax



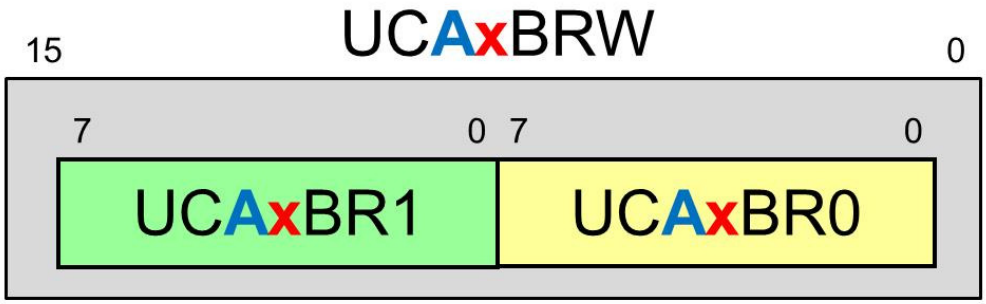
MSP430 – Registradores do USCI

UCAxCTL1 → Controle 1 do USCI_Ax



MSP430 – Registradores do USCI

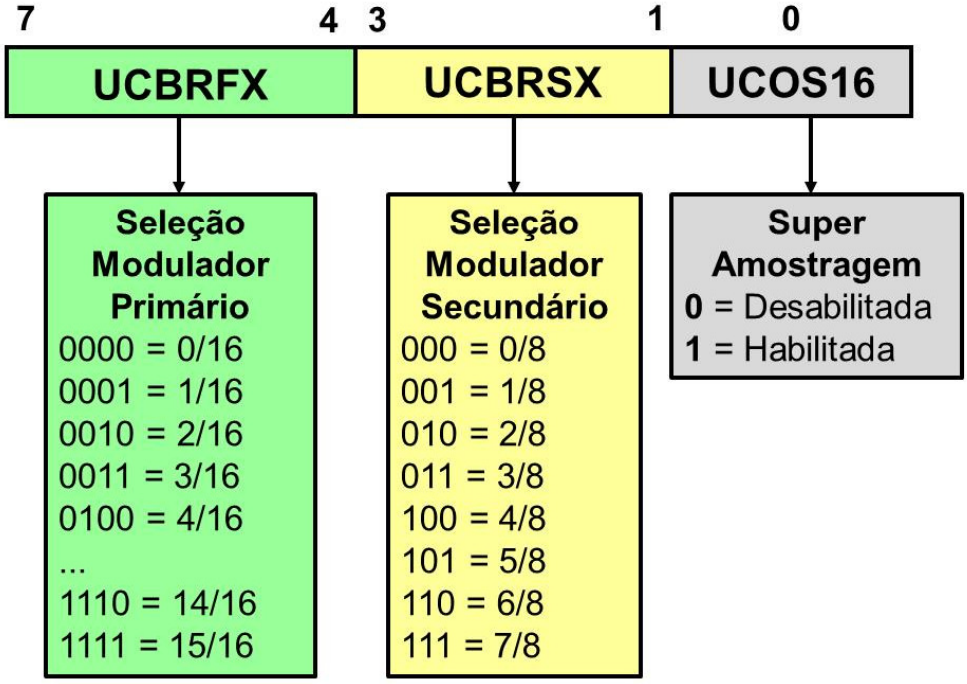
UCAxBRW → Palavra de Controle do BR



Este registrador (16 bits) é o divisor UCBRx

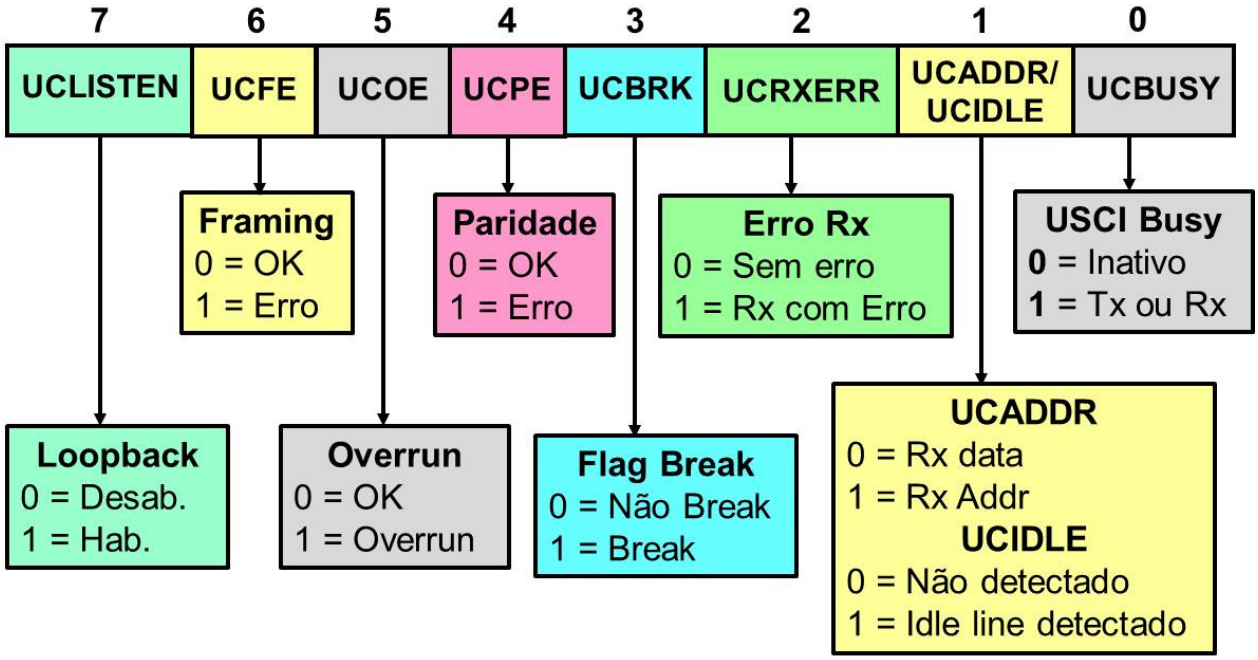
MSP430 – Registradores do USCI

UCAxMCTL → Controle do Modulador de USCI_Ax



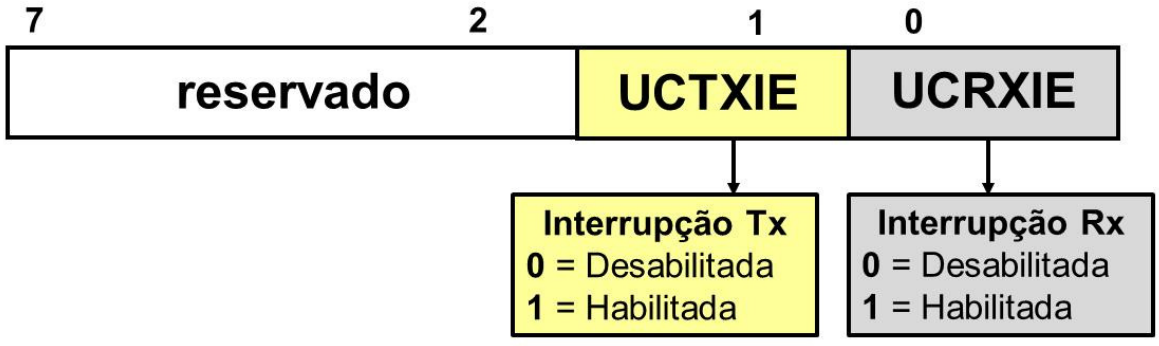
MSP430 – Registradores do USCI

UCAxSTAT → Registrador de Status do USCI_Ax



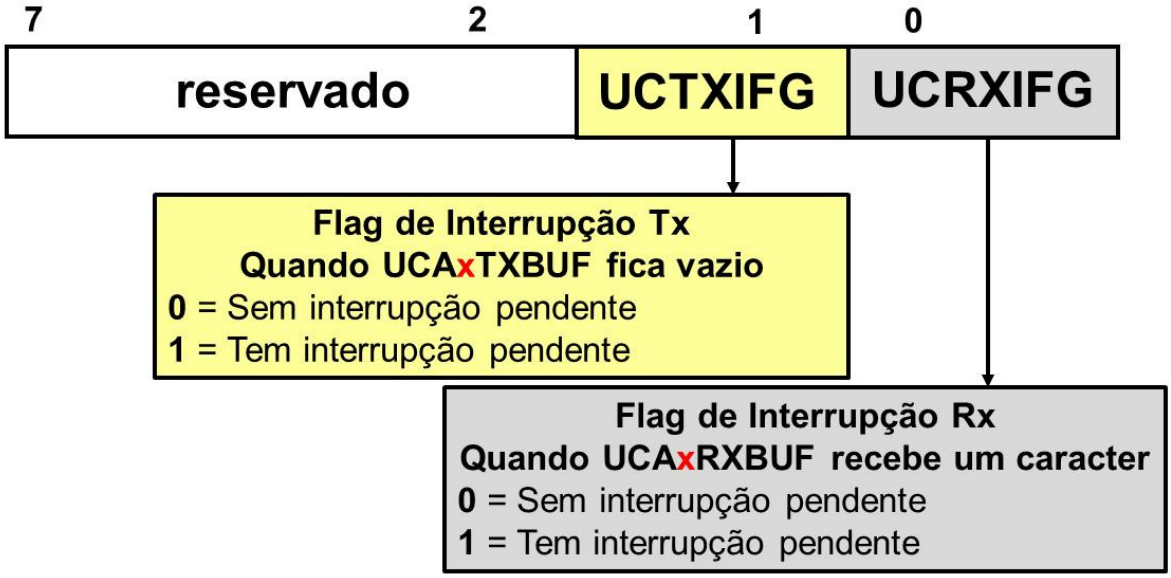
MSP430 – Registradores do USCI

UCAxIE → Registrador Habilita Interrupção



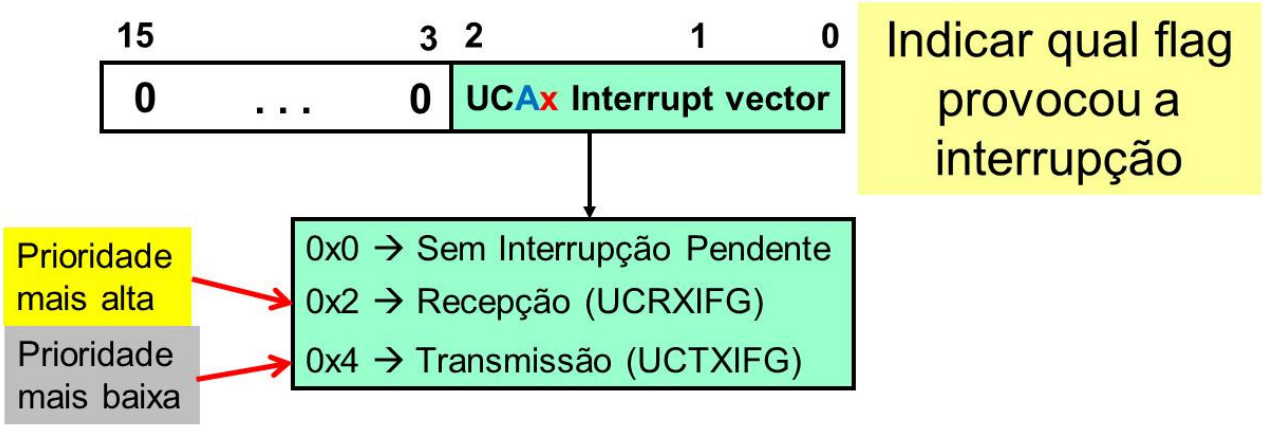
MSP430 – Registradores do USCI

UCAxIFG → Registrador de Flags de Interrupção

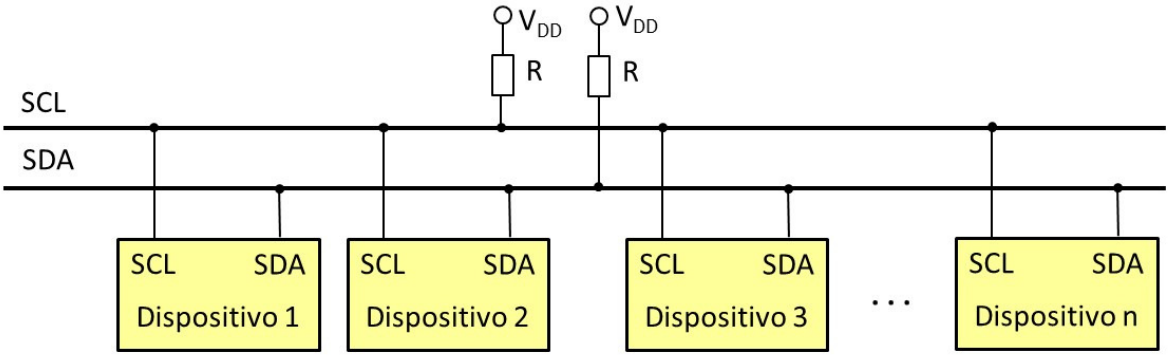


MSP430 – Registradores do USCI

UCAxIV → Vetor de Interrupção do UCA

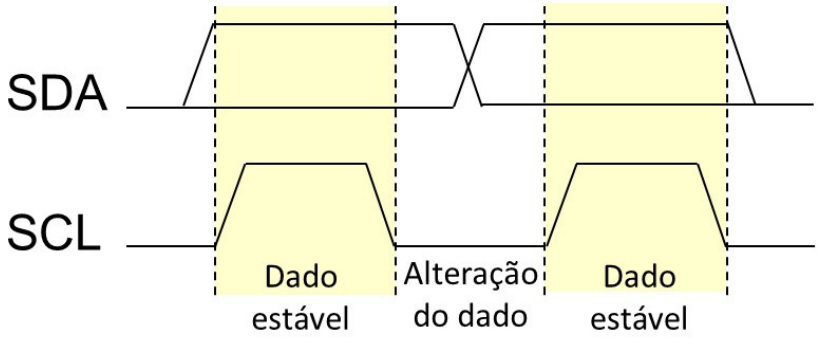


MSP430 – I2C – Terminologia



Termo	Descrição
Mestre	É o dispositivo que inicia e termina a transmissão. Ele é responsável por gerar o relógio SCL.
Escravo	Dispositivo endereçado pelo Mestre.
Transmissor	Dispositivo que coloca dados no barramento.
Receptor	Dispositivo que lê os dados do barramento.

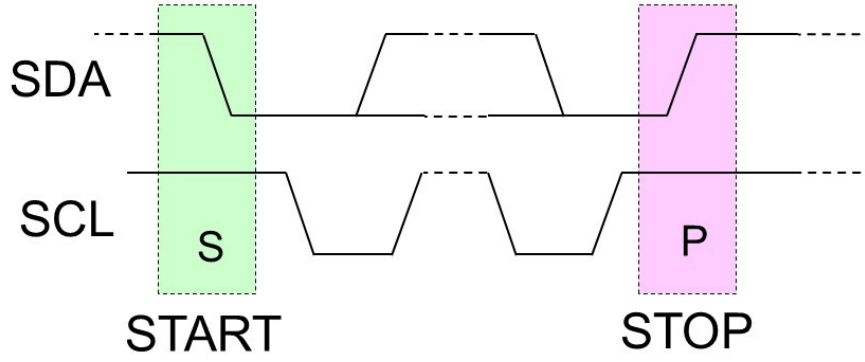
MSP430 – I2C – Transf. de Dado



SCL = 0 → pode alterar o dado.

SCL = 1 → dado estável.

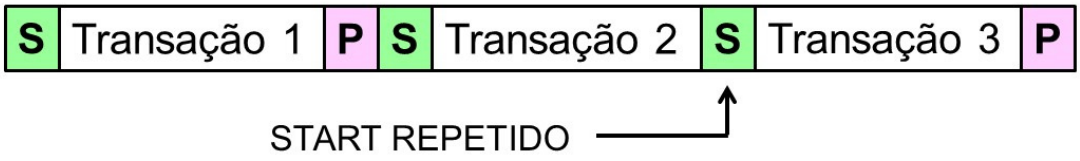
MSP430 – I2C – Start/Stop



START → SCL = 1 e SDA vai de alto para baixo.

STOP → SCL = 1 e SDA vai de baixo para alto.

MSP430 – I2C – Start Repetido



Start Repetido → Mestre iniciou uma nova transação sem liberar o barramento.

Usado geralmente para mudar a direção da comunicação com o escravo.

MSP430 – I2C – Endereço

- Pacote de endereço tem 7 bits + 1 bit de direção

b7	b6	b5	b4	b3	b2	b1	R/$\overline{W}$
1	0	0	1	0	1	0	**x**

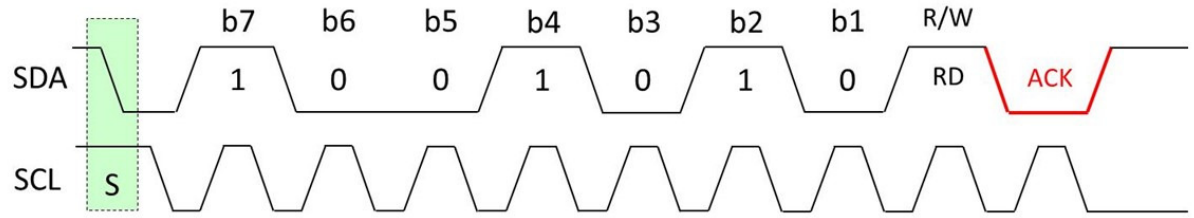
R/$\overline{W}$ = 1→ Leitura
R/W = 0 → Escrita

- 2^7 = 128 endereços possíveis.

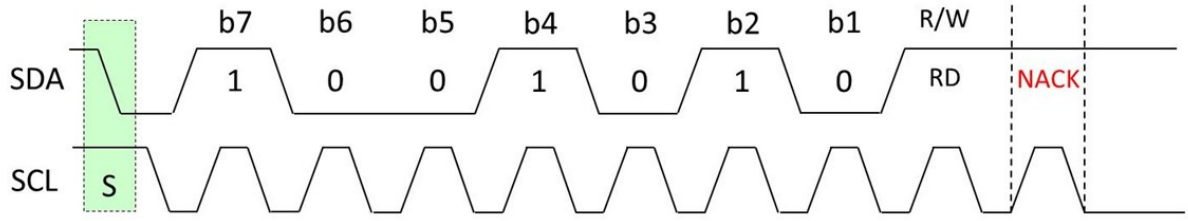
- O dispositivo acima (1001 010**x**) tem 2 endereços:
  - Um para leitura (SLA+R) → 0x95
  - Outro para escrita (SLA+W) → 0x94

MSP430 – I2C – Endereço

- Dispositivo endereçado tem obrigação de responder com um acknowledge (ACK).

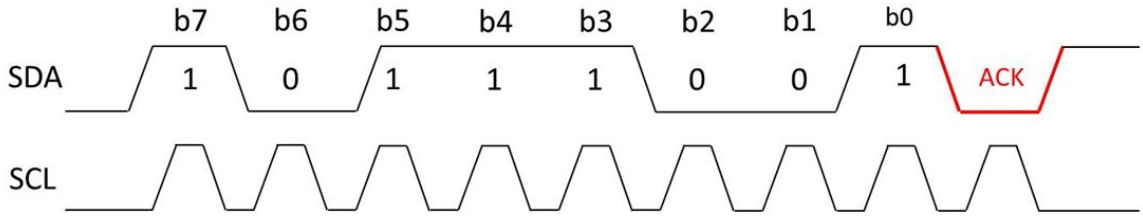


- Caso esteja ocupado, usa o not acknowledge (NAK)

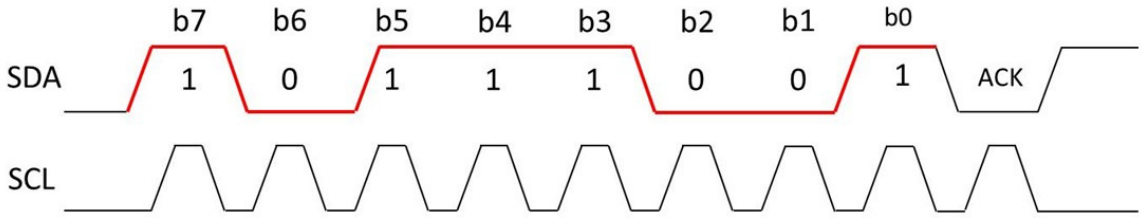


MSP430 – I2C – Endereço

- Mestre → Escravo

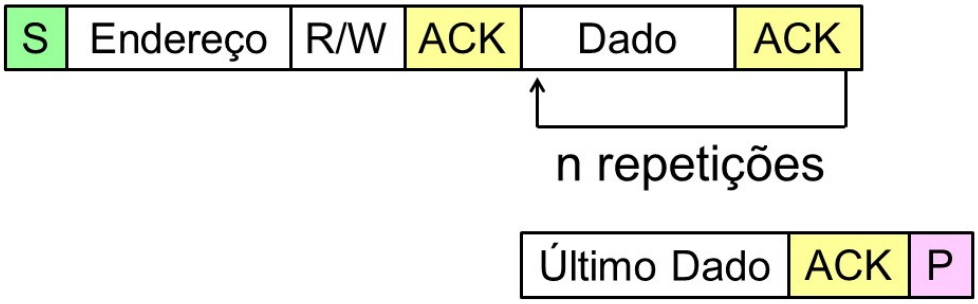


- Escravo → Mestre

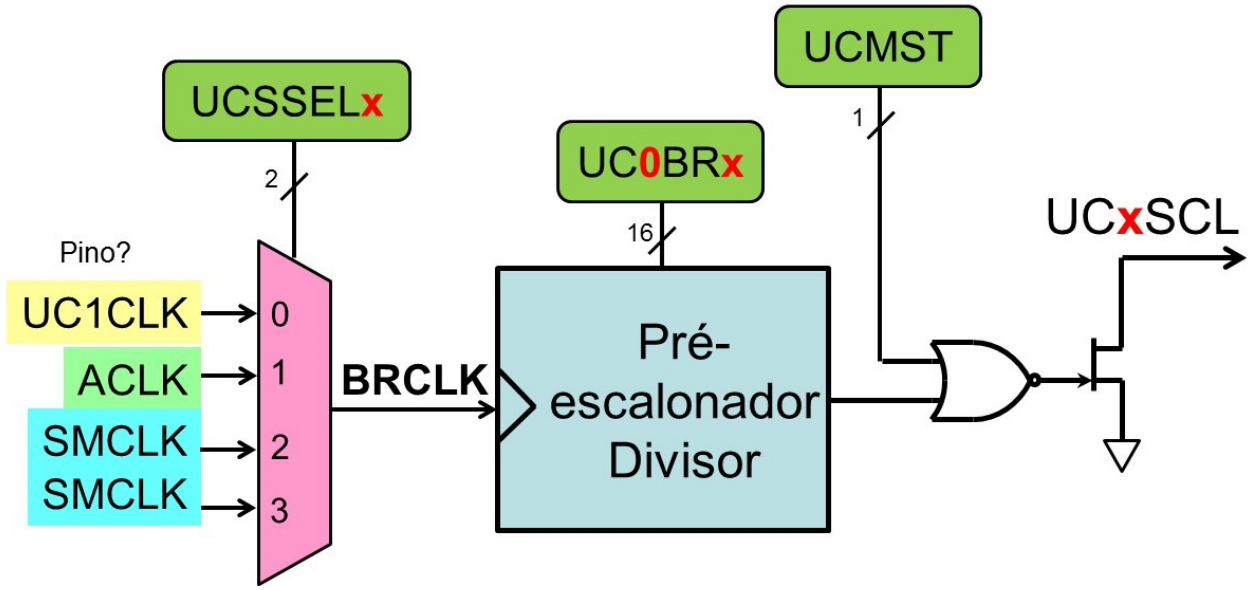


MSP430 – I2C – Típica Transmissão

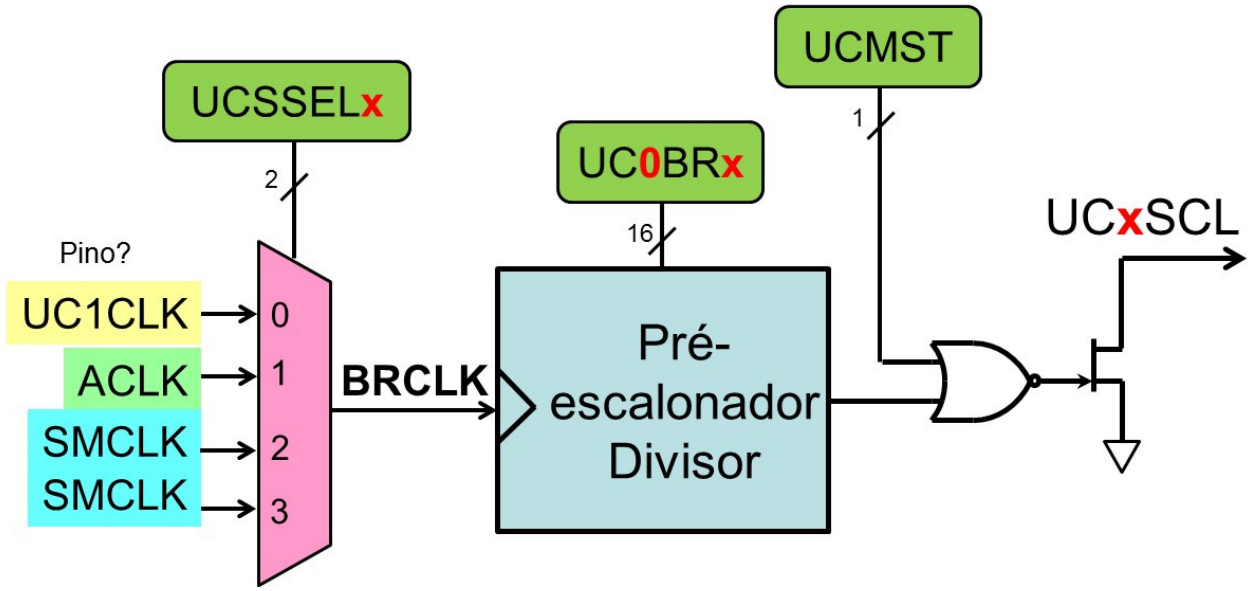
- Mestre → Escravo ou Escravo → Mestre.



MSP430 – USCI_Ax modo I2C



MSP430 – USCI_Ax modo I2C



MSP430 – USCI – I2C Mestre

Para ser configurada como Mestre:

1) UCMODE**x** = 11b

2) UCSYNC = 1

3) UCMST = 1

4) Em caso de multimestre, configurar UCB**x**I2COA Own Address

5) UCA10 = 0 → 7 bits de endereço

6) UCA10 = 1 → 10 bits de endereço

7) UCGCEN = 1 → se for responder ao General Call

MSP430 – USCI – I2C Mestre TX

1) UCB**x**I2CSA = endereço do escravo

2) UCSLA10 = 0 → para endereço 8 bits

3) UCTR = 1 → Modo Transmissor

4) UCTXSTT = 1→ gerar condição de START

a) USCI verifica a disponibilidade do barramento

b) Gera condição de START e transmite endereço

c) UCTXSTT = 0→ escravo gerou ACK

d) Se UCTXIFG = 1, pronto para transmitir

d.1) UCB**x**TXBUF = recebe dados a transmitir

MSP430 – USCI – I2C Mestre TX

UCTXSP = 1 → gerar STOP após ACK do escravo

Escrever em UCBxTXBUF, depois fazer UCTXSP = 1

UCTXSTT = 1→ permite gerar START repetido

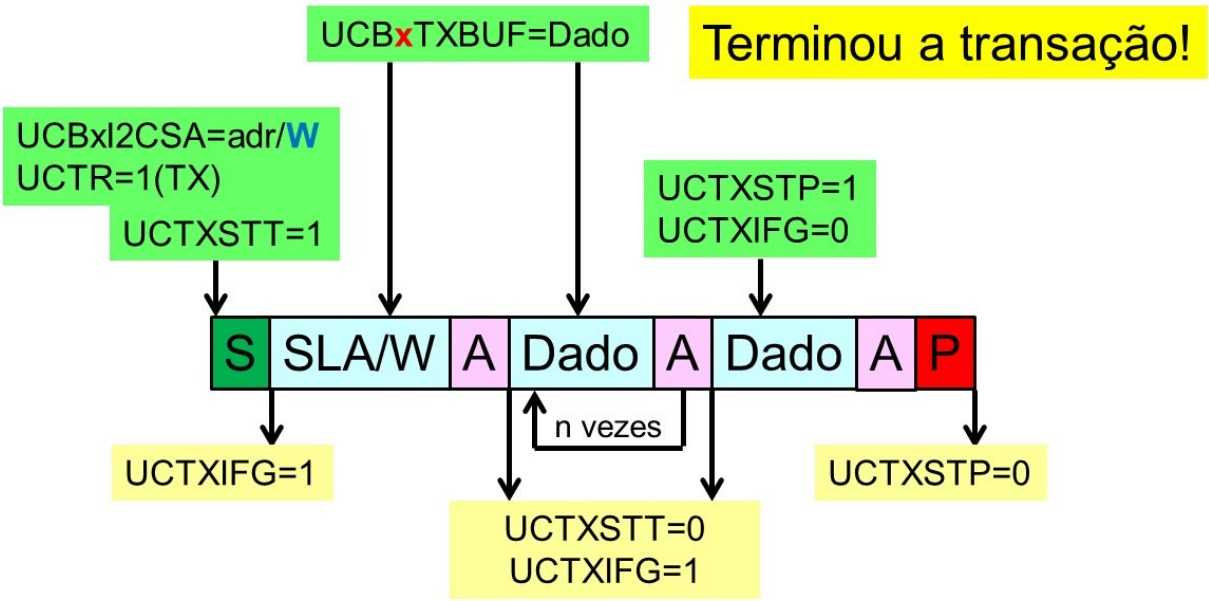
Neste caso UCTR (direção) pode mudar de valor

UCNACIFG = 1→ indica que escravo não gerou ACK

i) Mestre gera um STOP ou

ii) Mestre gera um START repetido

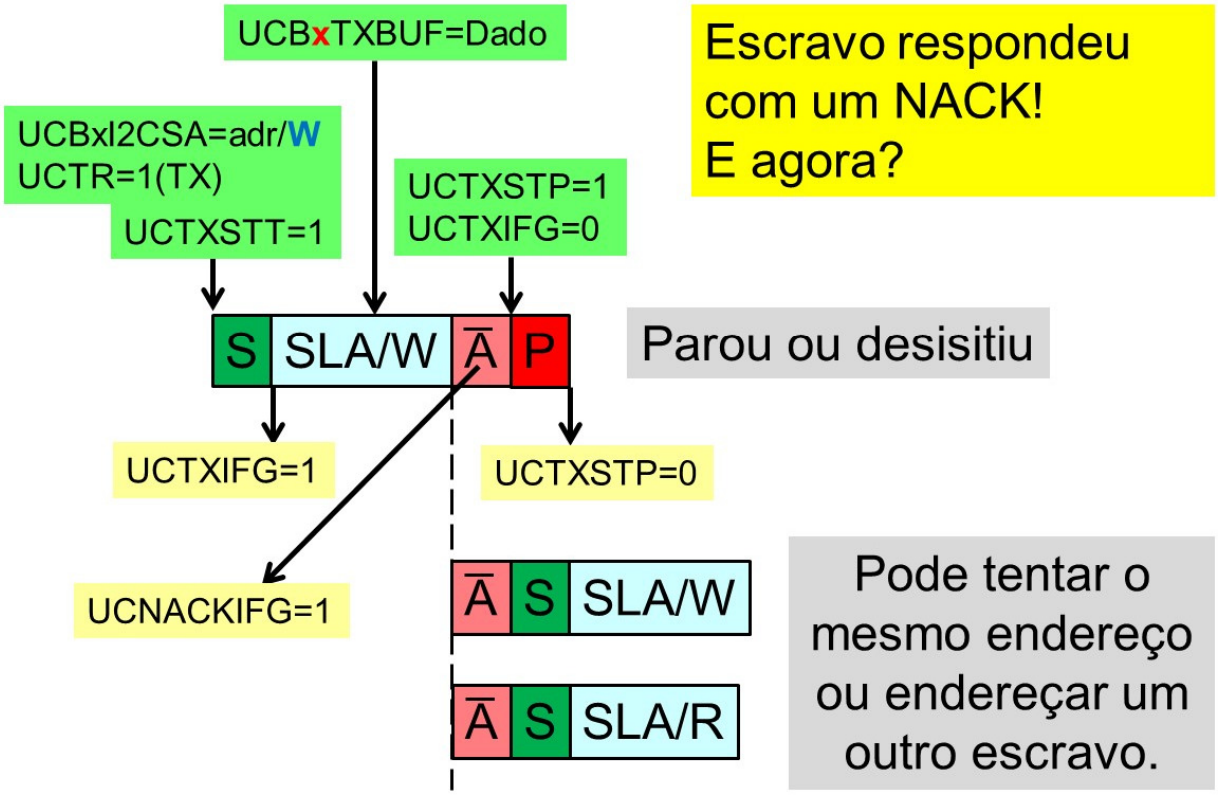
MSP430 – USCI – I2C Mestre TX



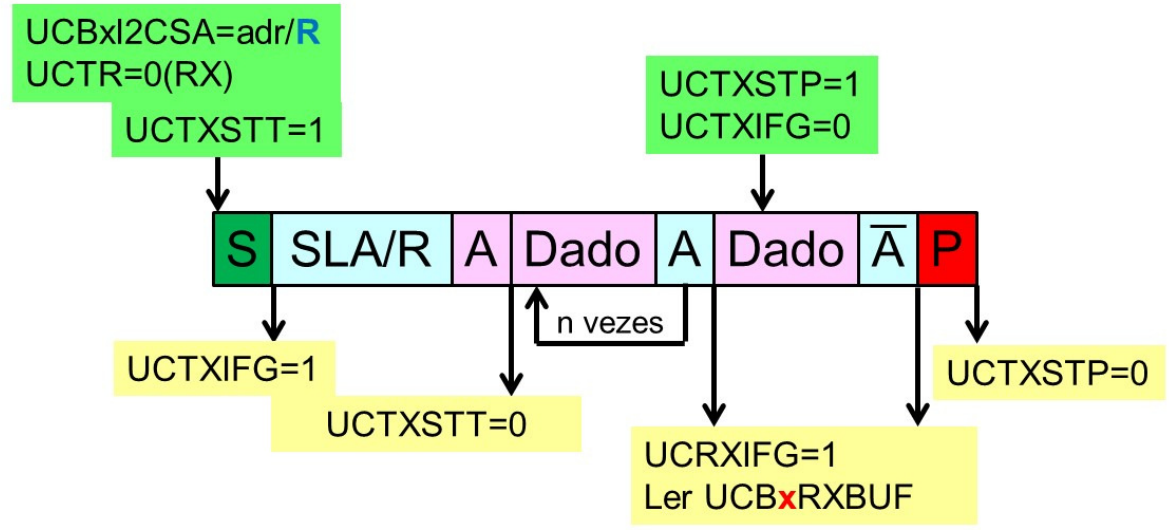
UCTXIFG=1 → pode escrever dado no UCBxTXBUF

Só fazer UCTXSTP=1 após UCTXIFG=1

MSP430 – USCI – I2C Mestre TX



MSP430 – USCI – I2C Mestre RX

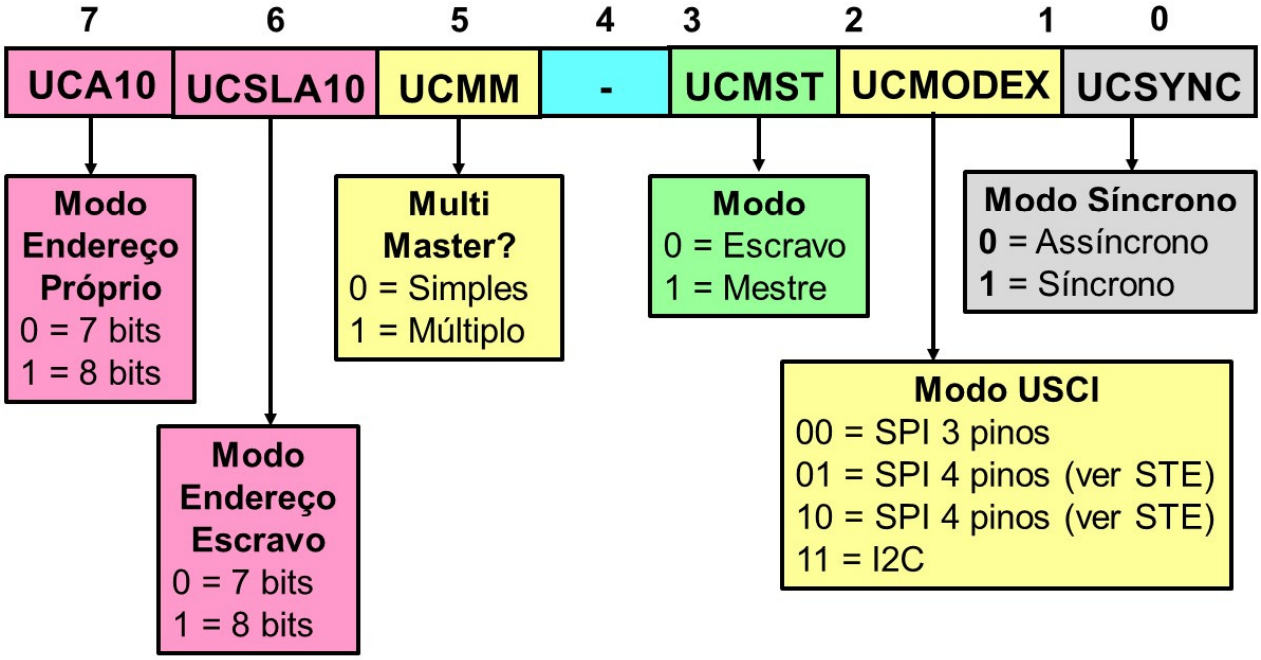


UCRXIFG=1 → pode ler dado no UCBxRXBUF

Terminou a transação!

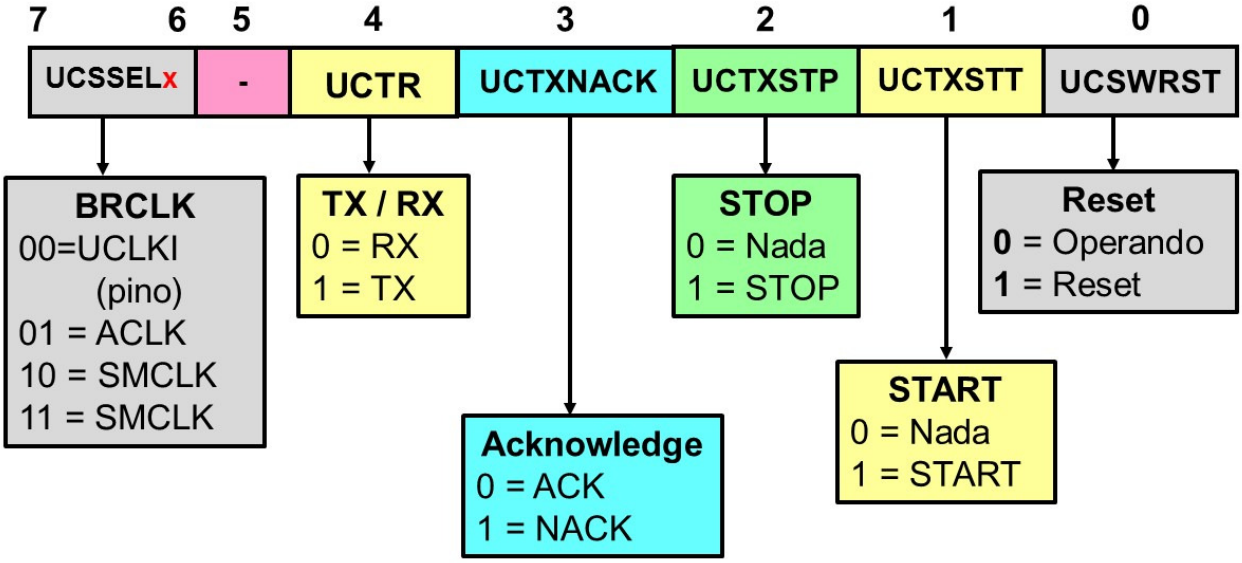
MSP430 – Registradores - USCI - I²C

UCBxCTL0 → Controle 0 do USCI_Bx



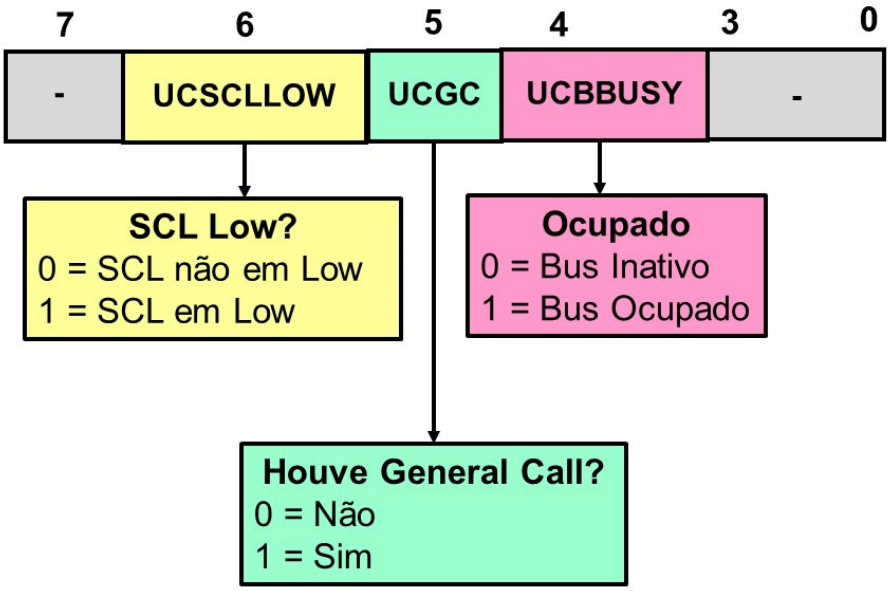
MSP430 – Registradores - USCI - I²C

UCBxCTL1 → Controle 1 do USCI_Bx



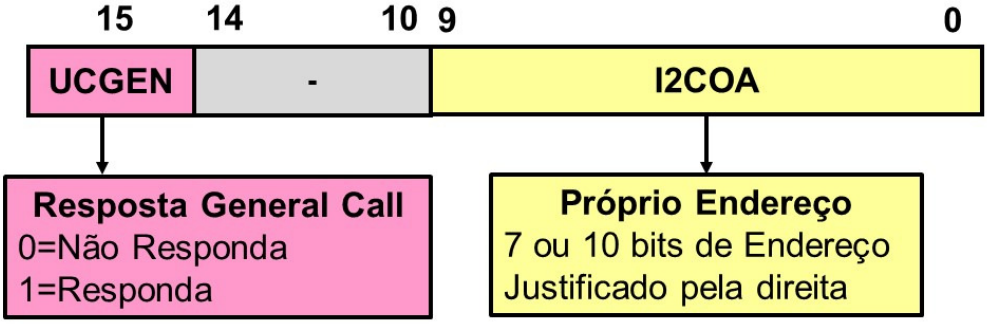
MSP430 – Registradores - USCI - I²C

UCBxSTAT → Registrador de Status do USCI_Bx



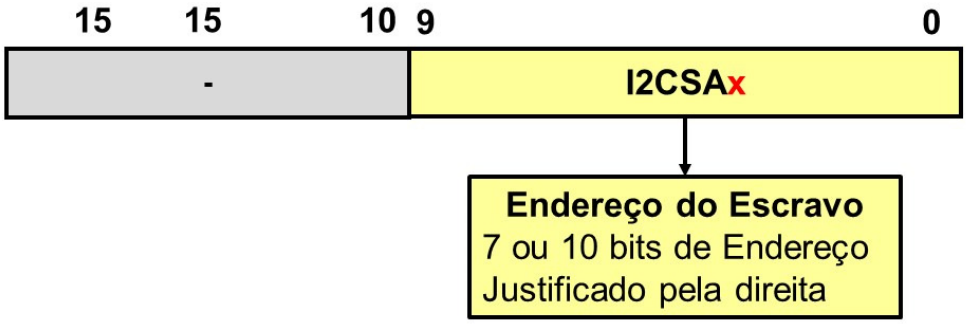
MSP430 – Registradores - USCI - I²C

UCBxI2COA → Próprio Endereço do USCI_Bx



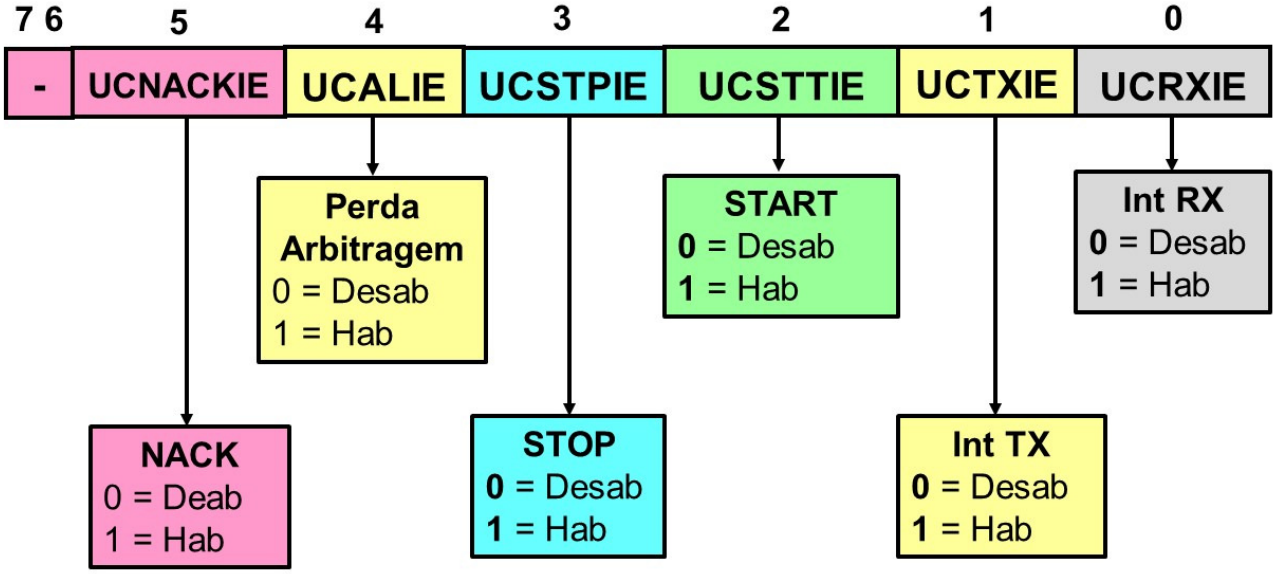
MSP430 – Registradores - USCI - I²C

UCBxI2CSA → Endereço do Escravo do USCI_Bx



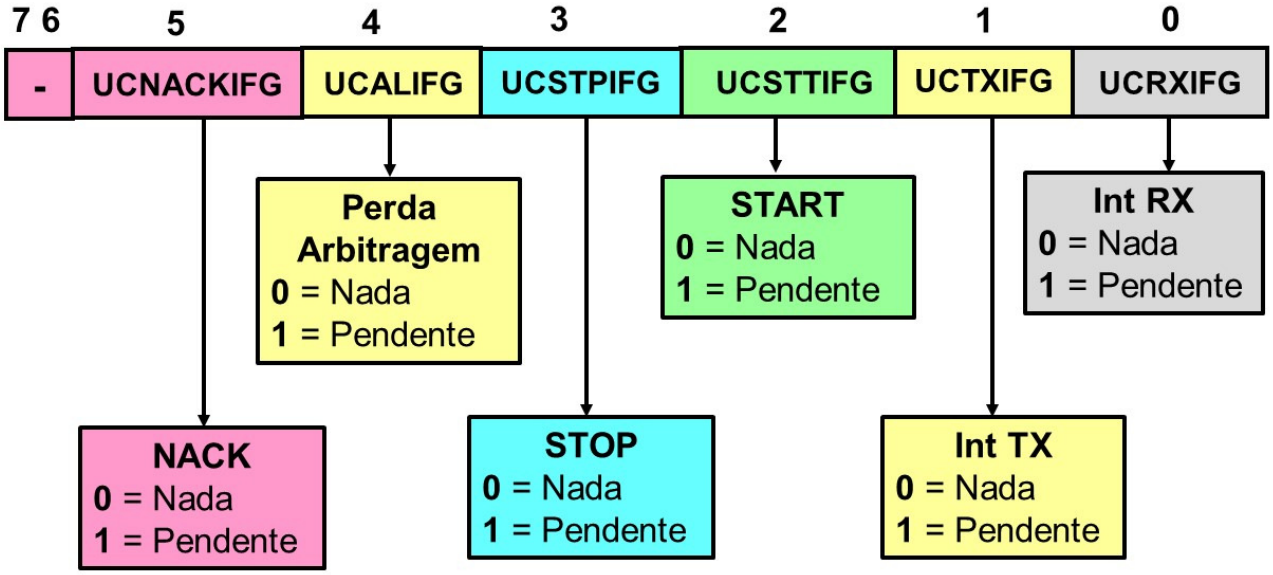
MSP430 – Registradores - USCI - I²C

UCBxIE → Habilitação de Interrupções do USCI_Bx



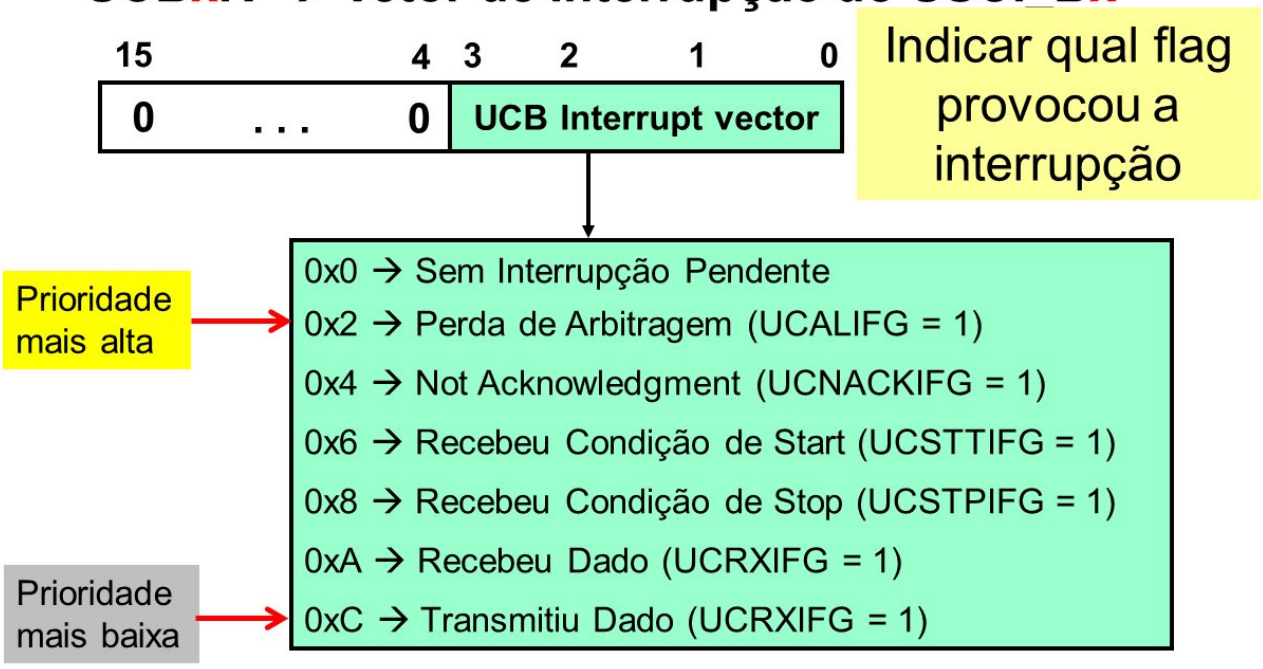
MSP430 – Registradores - USCI - I²C

UCBxIFG → Flags de Interrupções do USCI_Bx



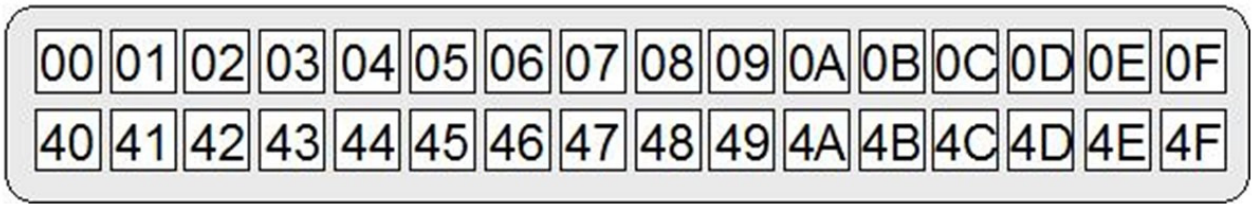
MSP430 – Registradores - USCI - I²C

UCBxIV → Vetor de Interrupção do USCI_Bx

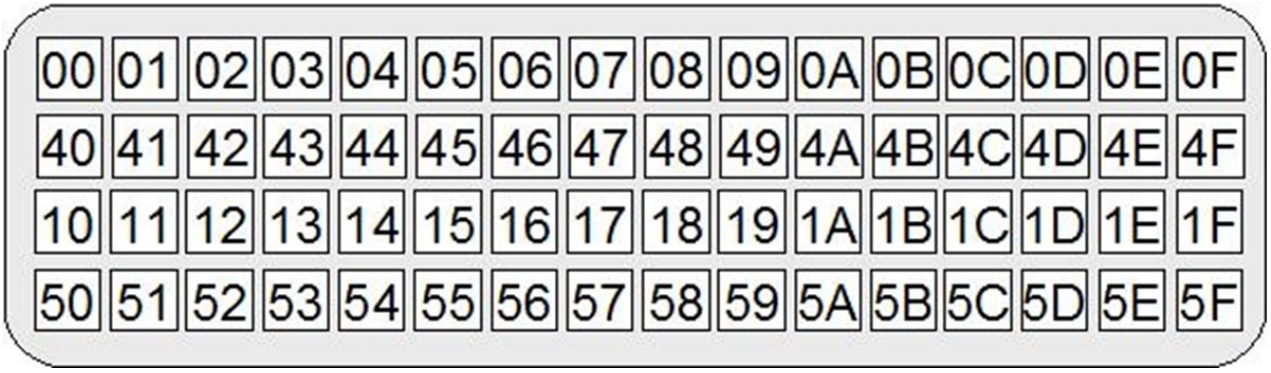


MSP430 – LCD – Formato

2 X 16



4 X 16



MSP430 – LCD – Memórias

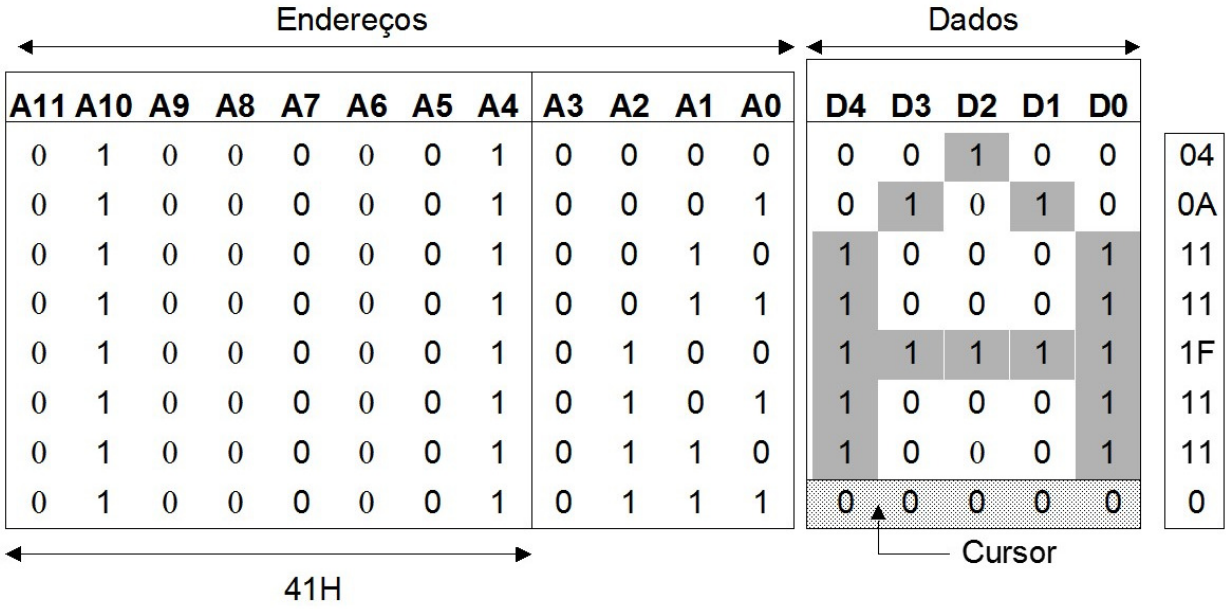
DDRAM ⟶ Memória de 80 bytes: um byte para cada posição do display.
Linha 1: 0x00 → 0x27
Linha 2: 0x40 → 0x67

CGROM ⟶ Memória de 4 KB que indica a matriz de pontos para cada caracter.

CGRAM ⟶ Memória de 256 B com a matriz de pontos especificada pelo usuário para os caracteres de 0 até 7 (repete para 8 até 15).

MSP430 – LCD – Memórias

CGROM ⟶ Matriz 5 X 8 pontos para cada caracter.



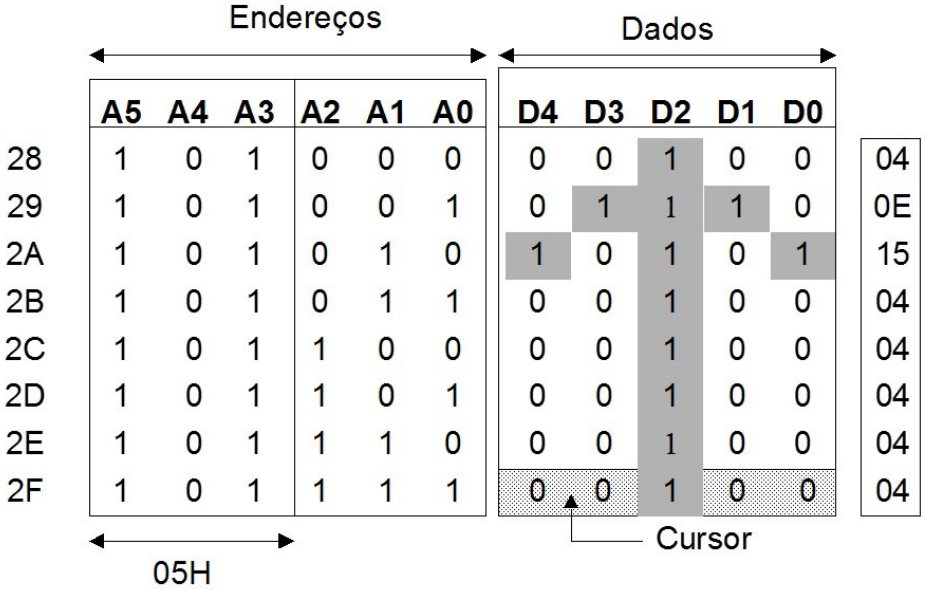
Endereços												Dados					
A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	D4	D3	D2	D1	D0	
0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	04
0	1	0	0	0	0	0	1	0	0	0	1	0	1	0	1	0	0A
0	1	0	0	0	0	0	1	0	0	1	0	1	0	0	0	1	11
0	1	0	0	0	0	0	1	0	0	1	1	1	0	0	0	1	11
0	1	0	0	0	0	0	1	0	1	0	0	1	1	1	1	1	1F
0	1	0	0	0	0	0	1	0	1	0	1	1	0	0	0	1	11
0	1	0	0	0	0	0	1	0	1	1	0	1	0	0	0	1	11
0	1	0	0	0	0	0	1	0	1	1	1	0	0	0	0	0	0

Cursor

41H

MSP430 – LCD – Memórias

CGRAM ⟶ Matriz 5 X 8 pontos para cada caracteres de 0 até 7.



	Endereços						Dados					
	A5	A4	A3	A2	A1	A0	D4	D3	D2	D1	D0	
28	1	0	1	0	0	0	0	0	1	0	0	04
29	1	0	1	0	0	1	0	1	1	1	0	0E
2A	1	0	1	0	1	0	1	0	1	0	1	15
2B	1	0	1	0	1	1	0	0	1	0	0	04
2C	1	0	1	1	0	0	0	0	1	0	0	04
2D	1	0	1	1	0	1	0	0	1	0	0	04
2E	1	0	1	1	1	0	0	0	1	0	0	04
2F	1	0	1	1	1	1	0	0	1	0	0	04

Cursor

05H

MSP430 – LCD – Registradores

IR ⟶ **Registrador de Instruções** (WR)
Recebe as instruções.

DR **Registrador de Dados** (RD / WR)
Recebe o dado a ser escrito na DDRAM ou CGRAM.

AC ⟶ **Contador de Endereços** (RD)
Indica onde o dado será escrito.
Incrementa a cada escrita.
Acessado indiretamente via instrução

MSP430 – LCD – Registradores

RS	R/W	Reg.	Operação
0	0	IR	Escrever uma instrução em IR
1	0	DR	Escrever no DR(operação sobre DDRAM ou CGRAM)
0	1	AC	Ler o bit de ocupado (b7) e o valor de AC (b6 – b0)
1	1	DR	Ler o DR (operação sobre DDRAM ou CGRAM)

- LCD opera com 5 V.
- Muitos adotam solução com apenas escrita
- R/W = 0.

MSP430 – LCD – Instruções (RS=0)

Abrev.	Tempo	Instrução	D7	D6	D5	D4	D3	D2	D1	D0
CLR	1,52 ms	Limpar mostrador	0	0	0	0	0	0	0	1
HOME	1,52 ms	Retornar	0	0	0	0	0	0	1	-
EMS	37 μs	Definir do modo de entrada	0	0	0	0	0	1	I/D	S
DEC	37 μs	Ativar ou desativar mostrador	0	0	0	0	1	D	C	B
CDS	37 μs	Deslocar cursor ou mostrador	0	0	0	1	S/C	R/L	-	-
FS	37 μs	Definir condições	0	0	1	DL	N	F	-	-
SCG	37 μs	Especificar endereço CGRAM	0	1	A5	A4	A3	A2	A1	A0
SDD	0 μs	Especificar endereço DDRAM	1	A6	A5	A4	A3	A2	A1	A0

MSP430 – LCD – Instruções (RS=0 e R/#W=0)

Limpar

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	1

- Preenche toda DDRAM com “brancos” (0x20).
- AC = 0 → Leva cursor para 1ª linha 1ª coluna.
- I/D = 1 → incremento do AC.
- Anula os deslocamentos do passado.
- Pode demorar **1,5 ms**.

MSP430 – LCD – Instruções (RS=0 e R/#W=0)

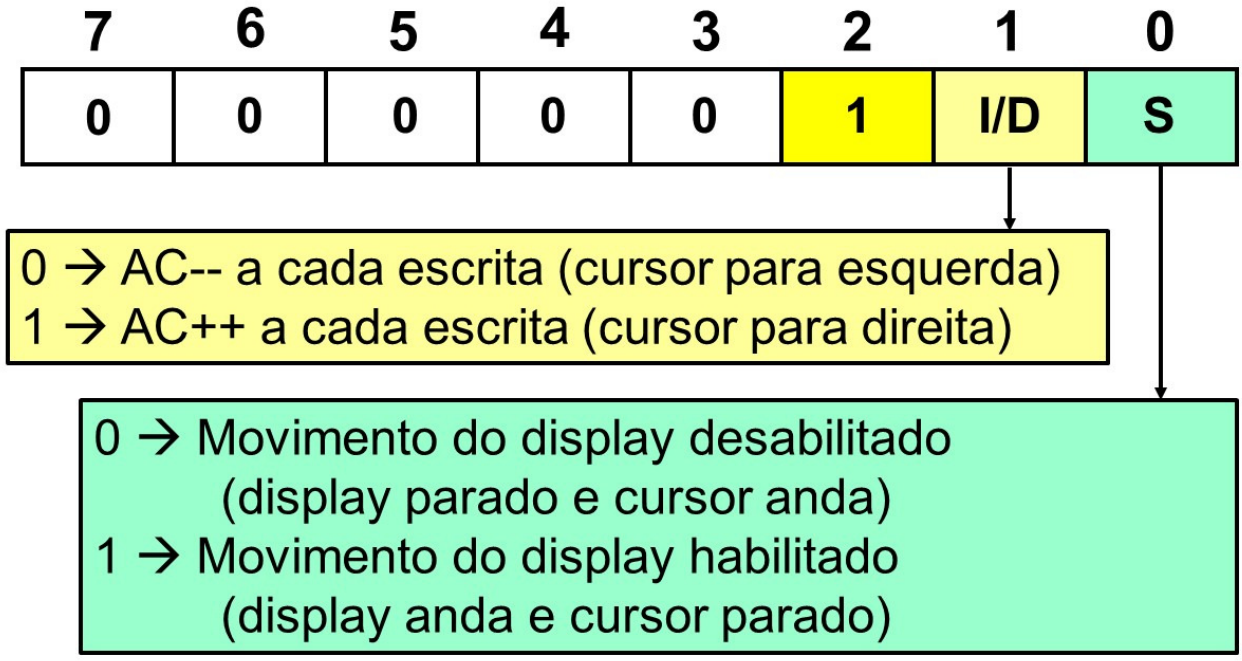
Retornar

7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	-

- AC = 0 → Leva cursor para 1ª linha 1ª coluna.
- Anula os deslocamentos do passado.
- Pode demorar **1,5 ms**.

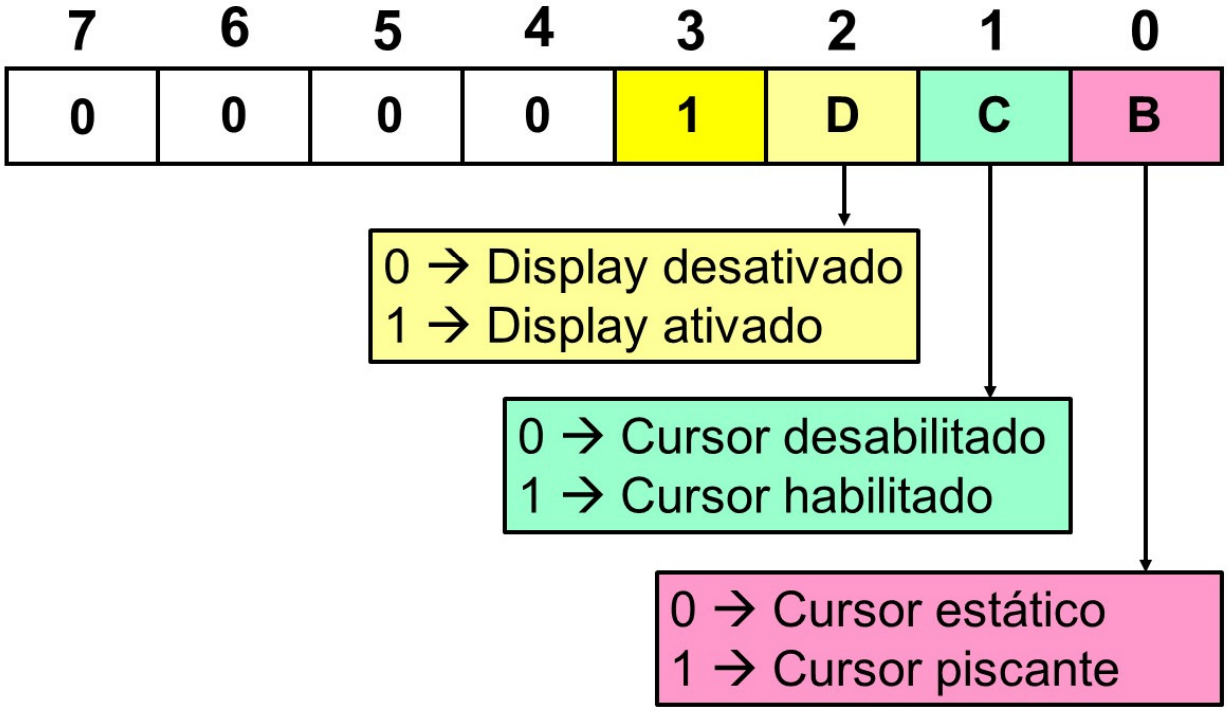
MSP430 – LCD – Instruções (RS=0 e R/#W=0)

Modo de Entrada



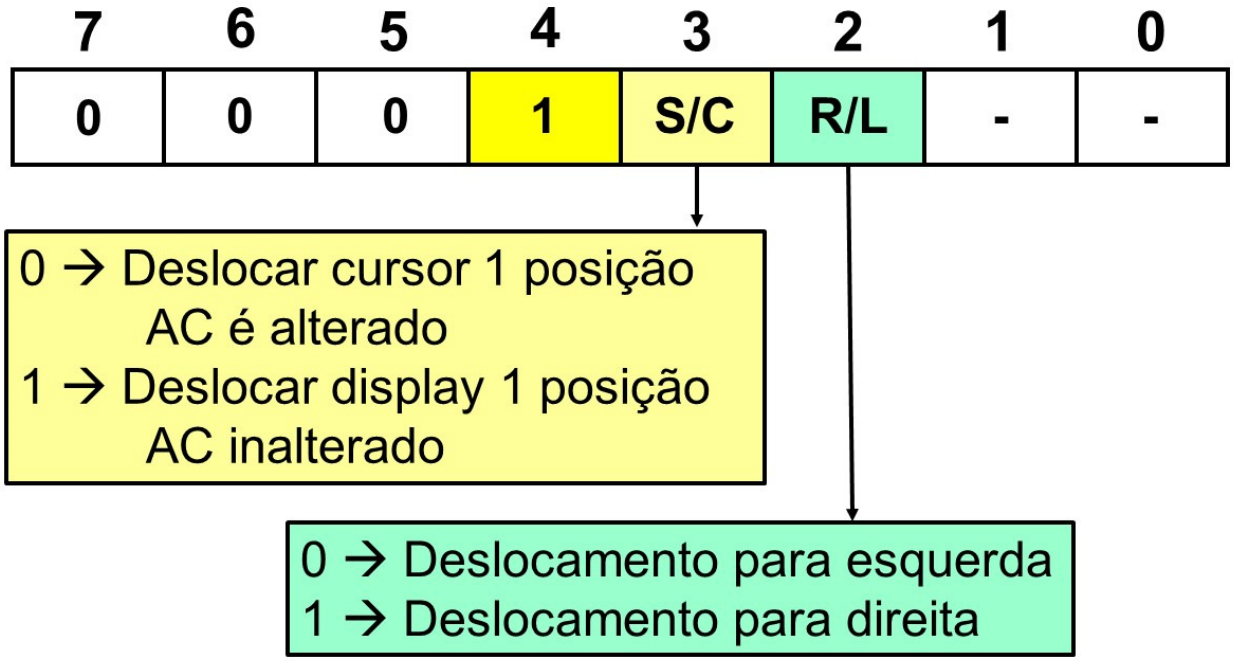
MSP430 – LCD – Instruções (RS=0 e R/#W=0)

Ativar ou Desativar Controles



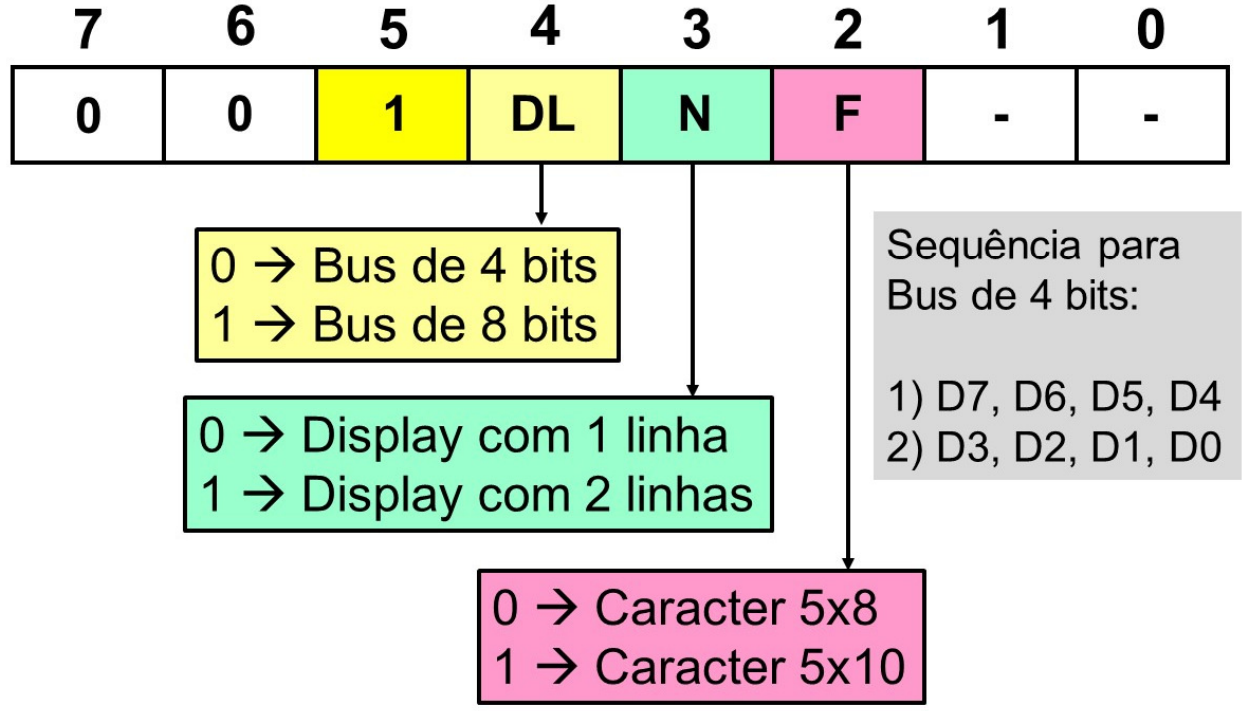
MSP430 – LCD – Instruções (RS=0 e R/#W=0)

Deslocar Cursor ou Display



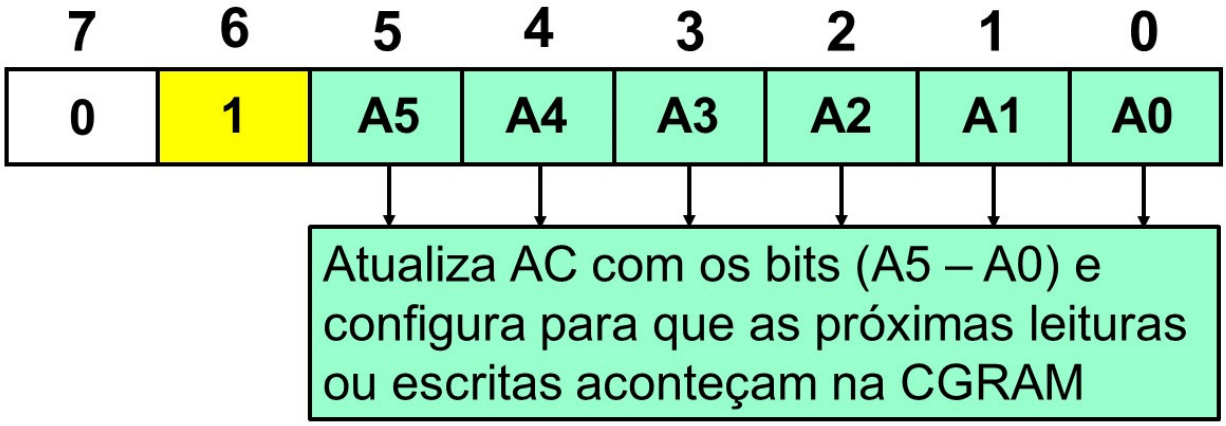
MSP430 – LCD – Instruções (RS=0 e R/#W=0)

Definir Condições



MSP430 – LCD – Instruções (RS=0 e R/#W=0)

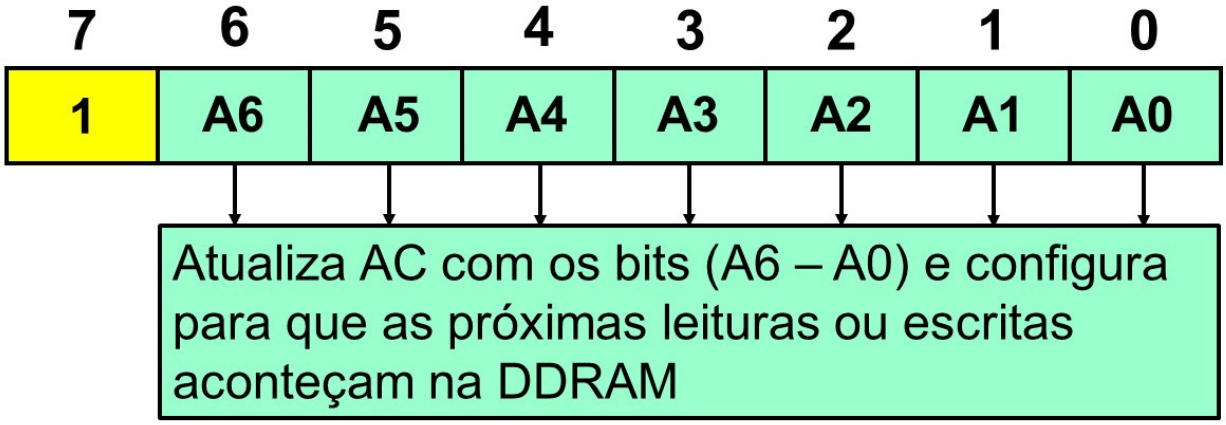
Especificar Endereço da CGRAM



CGRAM memória para os 8 caracteres que são configuráveis pelo usuário.

MSP430 – LCD – Instruções (RS=0 e R/#W=0)

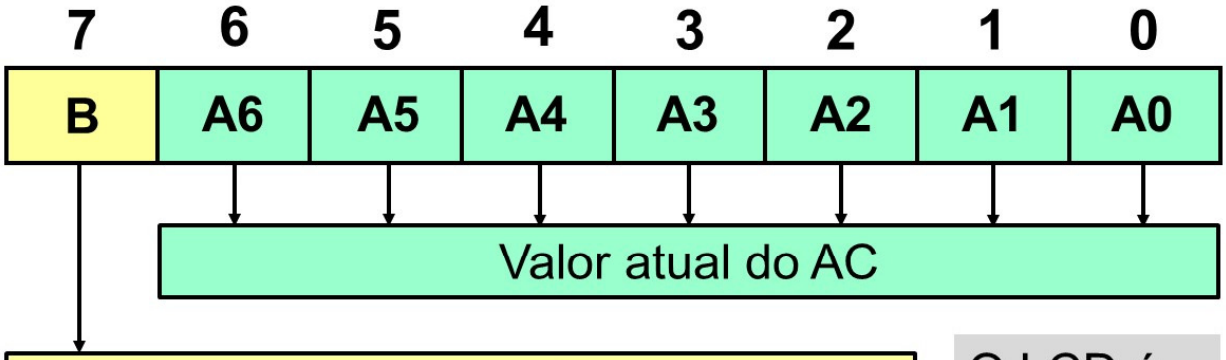
Especificar Endereço da DDRAM



Usado para especificar a nova posição de entrada dos caracteres.
Cursor se move para essa nova posição.

MSP430 – LCD – Leitura (RS=0 e R/#W=1)

Ler conteúdo do AC e bit de ocupado (Busy)



0 → Pode-se acessar o display
1 → Ocupado, não acessar o display

Vamos fazer essa leitura usando o PCF8574 (quasi-bidirecional).

O LCD é lento por isso é importante o bit de ocupado

MSP430 – LCD – Escrita (RS=1 e R/#W=0)

Escrever um dado no LCD

- A escrita acontece no DR.
- Depois o dado é copiado para a DDRAM.

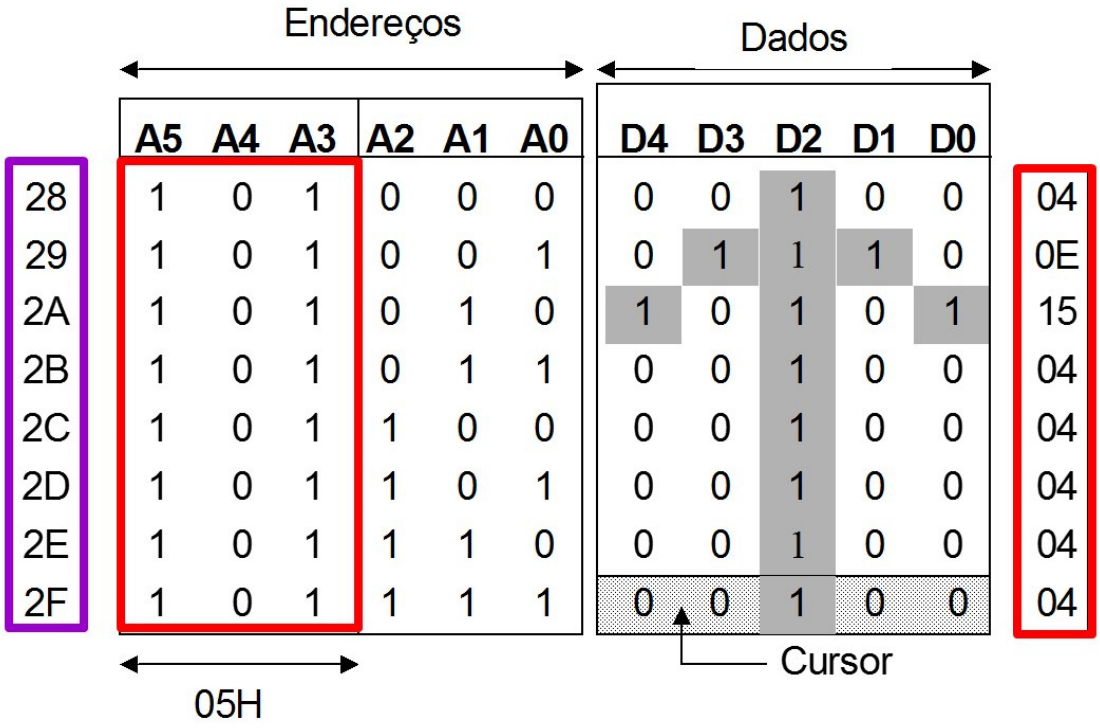
- A escrita acontece no endereço especificado pelo registrador AC.

- A cada escrita o AC é incrementado ou decrementado, de acordo com o que foi programado.

- A instrução anterior define se a escrita acontece na DDRAM ou na CGRAM.

MSP430 – LCD – Criar Caracter

Vamos definir o caracter 5 como seta para cima ↑



MSP430 – LCD – Instruções (RS=R/#W=0)

Especificar Endereço da CGRAM

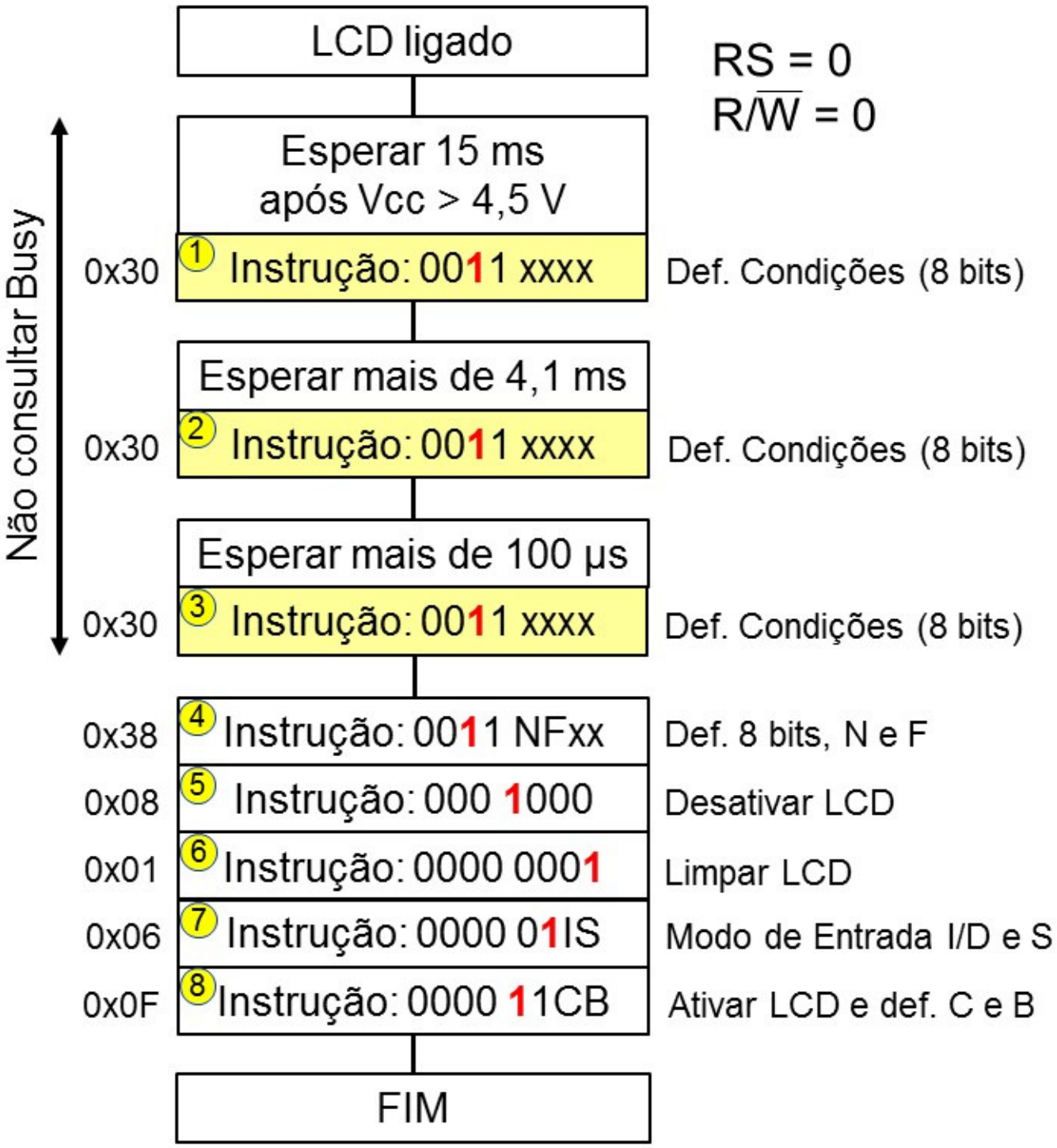
7	6	5	4	3	2	1	0
0	1	A5	A4	A3	A2	A1	A0
0	1	1	0	1	0	0	0

0x28

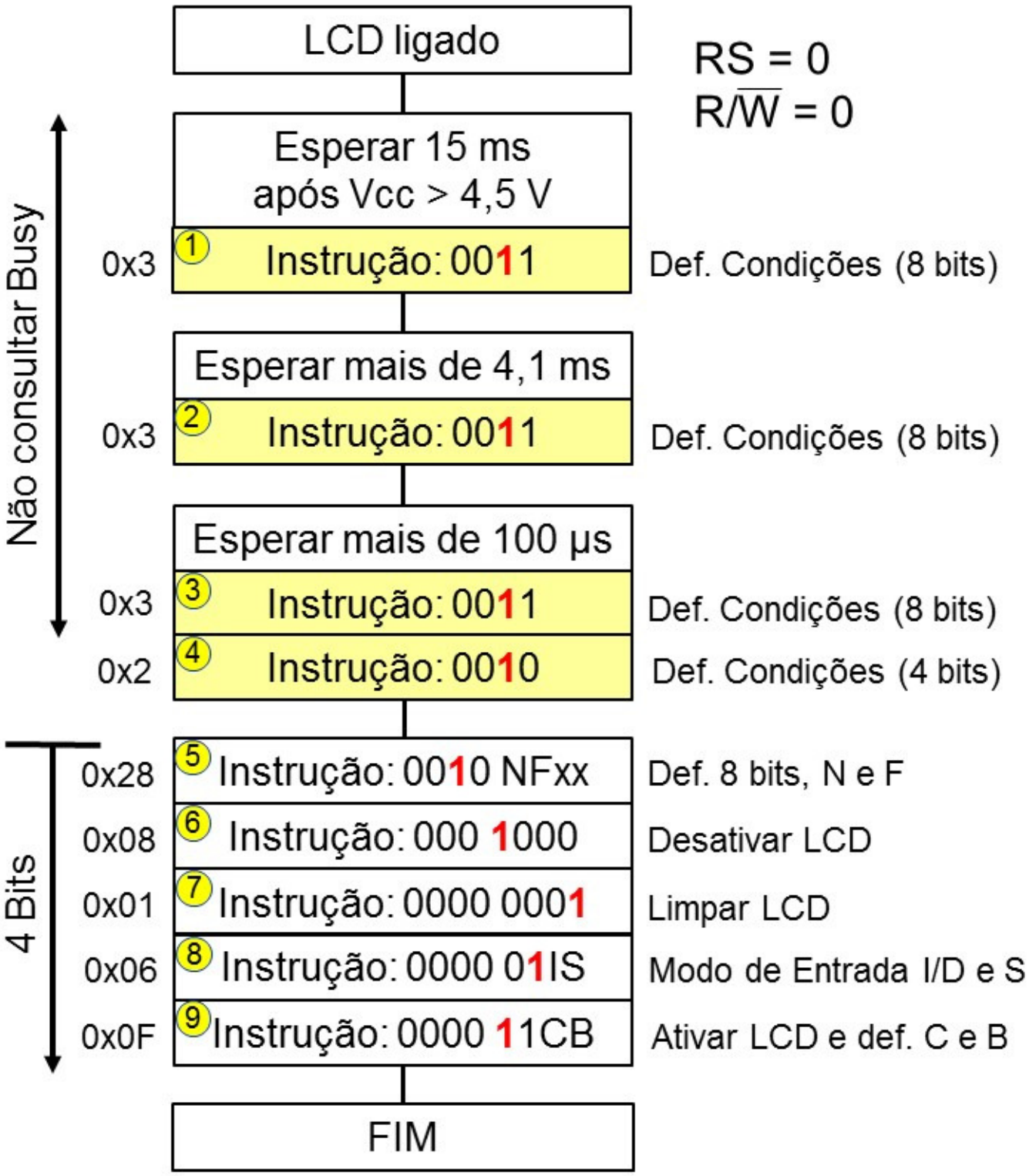
Enviar a instrução **0x68** e depois escrever os bytes:
0x04, 0x0E, 0X15, 0X04, 0X04, 0X04 , 0X04, 0X04.

Quando o usuário escrever o caracter **5**, vai aparecer no display a seta para cima.

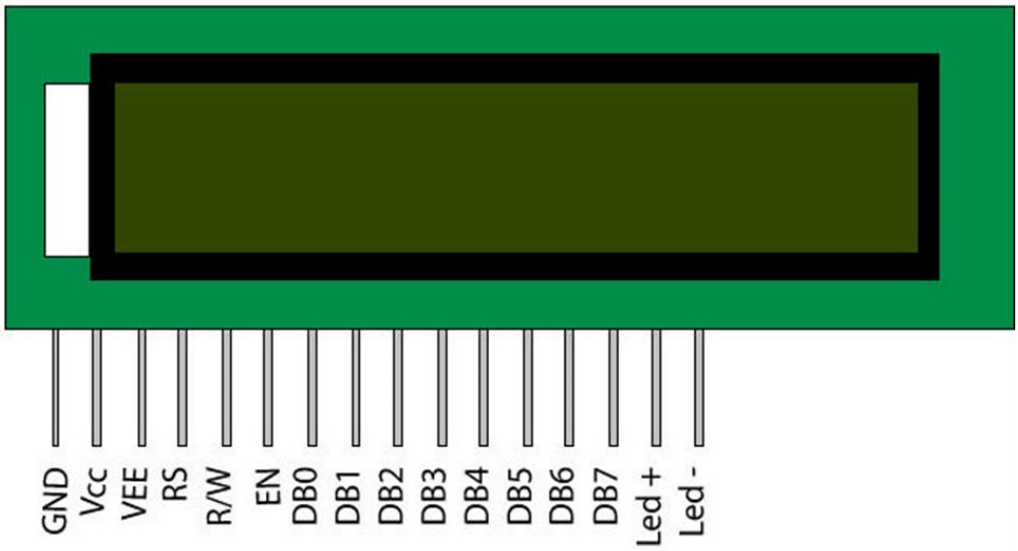
MSP430 – LCD – Inicialização 8 bits



MSP430 – LCD – Inicialização 4 bits



MSP430 – LCD – Pinagem

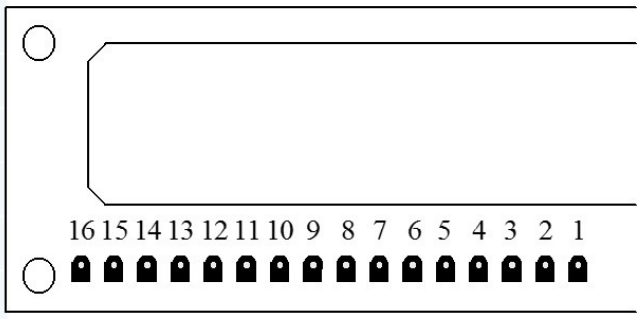


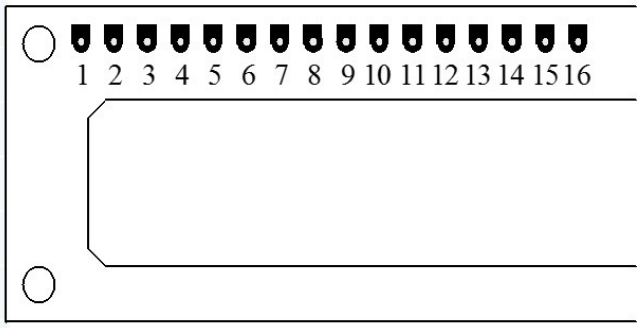
Sobre a interface elétrica

Pino	Nome	Função
1	V_{SS}	Terra
2	V_{DD}	Alimentação
3	V_{EE}	Contraste
4	RS	Seleciona Registrador
5	R/#W	Leitura/Escrita
6	E	Habilitação
7	D0	Dado, Bit 0
8	D1	Dado, Bit 1
9	D2	Dado, Bit 2
10	D3	Dado, Bit 3
11	D4	Dado, Bit 4
12	D5	Dado, Bit 5
13	D6	Dado, Bit 6
14	D7	Dado, Bit 7
15	A	Anodo
16	K	Catodo

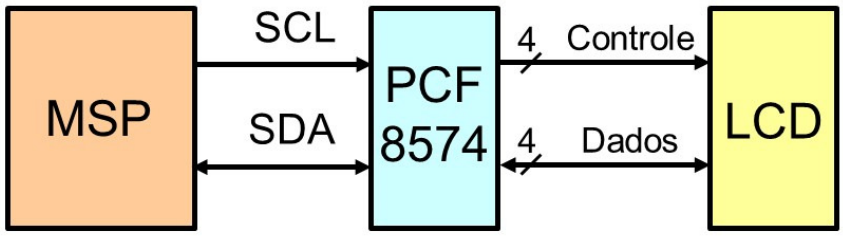
D – Pinagem

Não há disposição padrão. Cuidado!

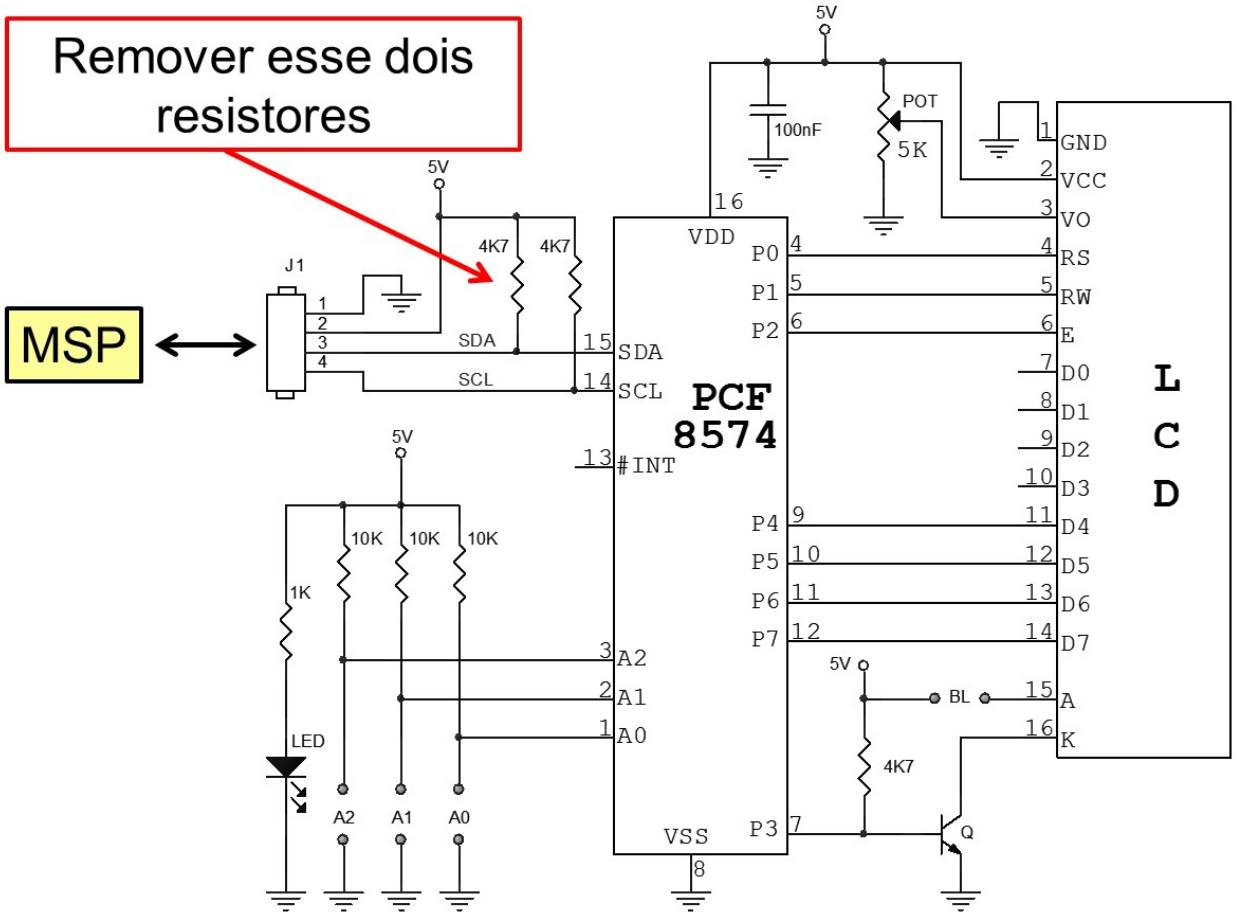




MSP430 – PCF8574

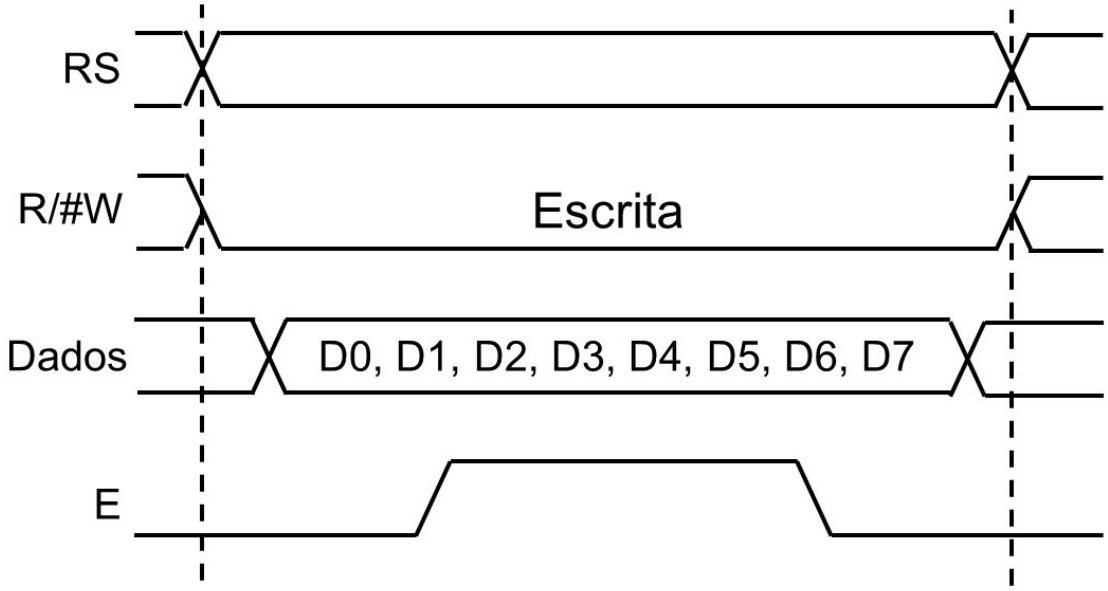


- Porta paralela quase-bidirecional.
- **Saída**: escrever zero ou um em cada pino.
- **Entrada**: antes de ler, escrever um no pino.
- **Endereços**:
  - PCF8574**T** → 0x27
  - PCF8574**AT** → 0x3F

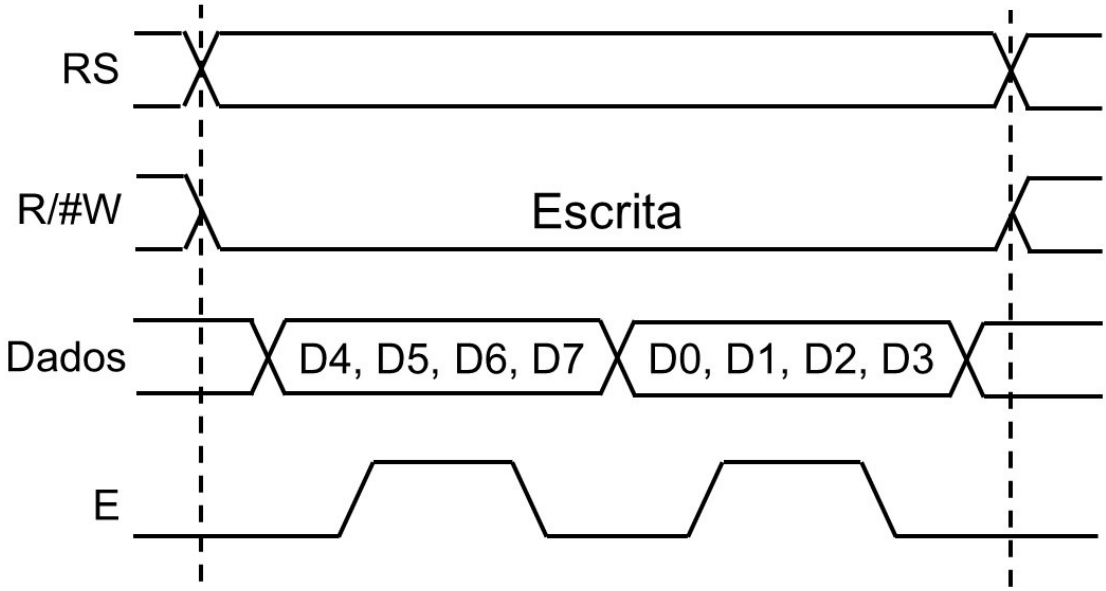


RS	R/W	Reg.	Operação
0	0	IR	Escrever uma instrução em IR
1	0	DR	Escrever no DR(DDRAM ou CGRAM)
0	1	AC	Ler Busy (b7) e o valor de AC (b6 – b0)
1	1	DR	Ler DR (DDRAM ou CGRAM)

MSP430 – LCD – Escrita em 8 bits

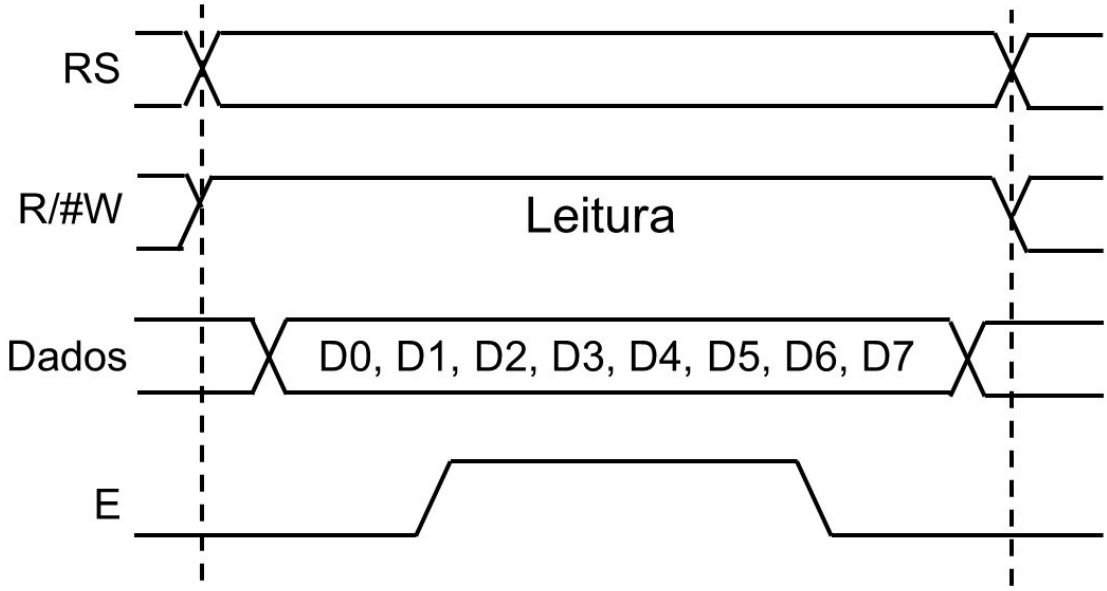


MSP430 – LCD – Escrita em 4 bits

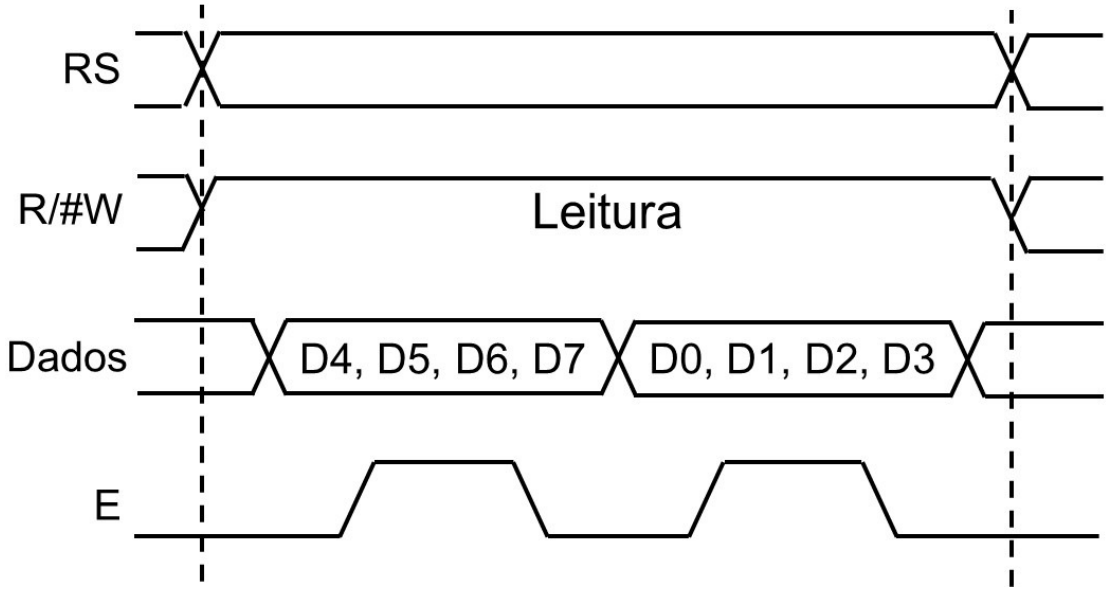


RS	R/W	Reg.	Operação
0	0	IR	Escrever uma instrução em IR
1	0	DR	Escrever no DR(DDRAM ou CGRAM)
0	1	AC	Ler Busy (b7) e o valor de AC (b6 – b0)
1	1	DR	Ler DR (DDRAM ou CGRAM)

MSP430 – LCD – Leitura em 8 bits



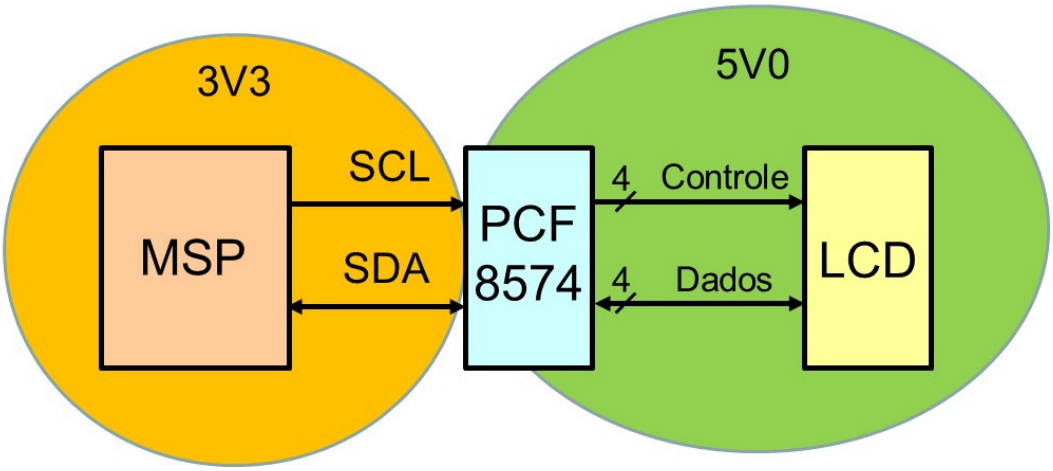
MSP430 – LCD – Leitura em 4 bits



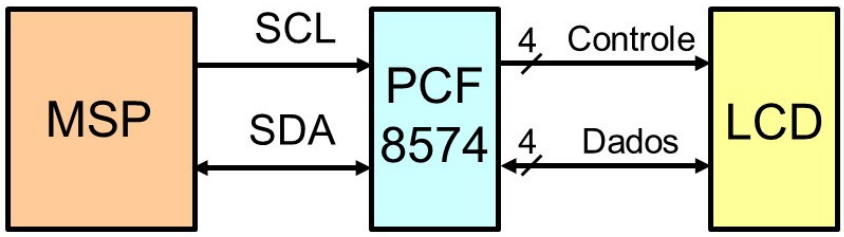
MSP430 – Solução com I²C

Vamos adotar uma solução usando um CI com porta paralela bidirecional e barramento I²C.

Resolve o progblema da compatibilidade entre :
MSP (3,3 V) e o LCD (5 V)



MSP430 – PCF8574



- Porta paralela quase-bidirecional.
- **Saída**: escrever zero ou um em cada pino.
- **Entrada**: antes de ler, escrever um no pino.
- **Endereços**:
  - PCF8574**T** → 0x27
  - PCF8574**AT** → 0x3F

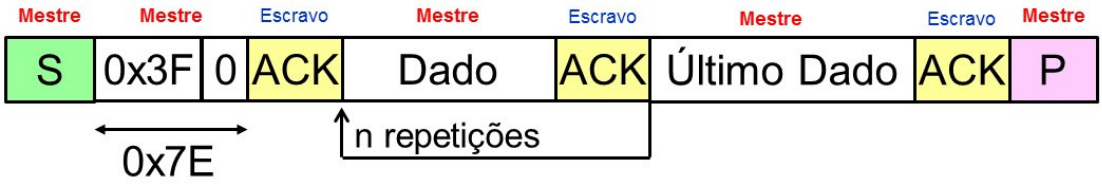
MSP430 – Solução com I²C

Arrumação dos pinos do PCF8574 e do LCD

7	6	5	4	3	2	1	0
D7	D6	D5	D4	BL	E	R/W	RS

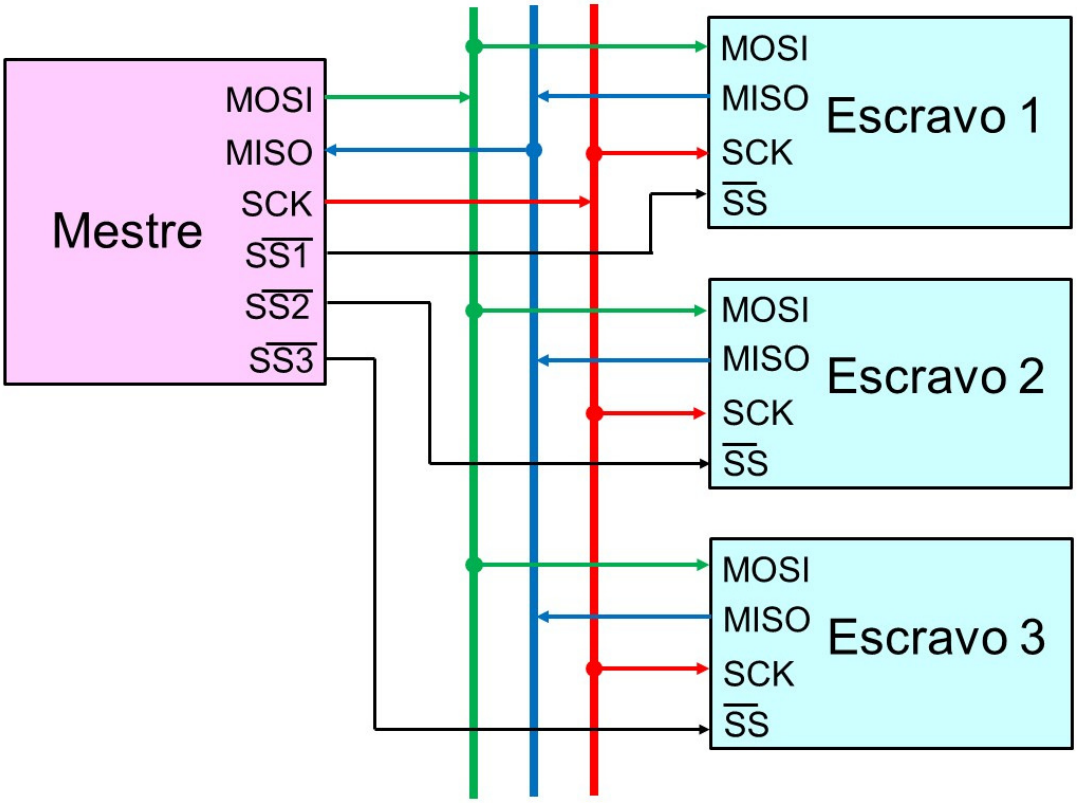
MSP430 – Solução com I²C

MSP enviando dados para o PCF8574



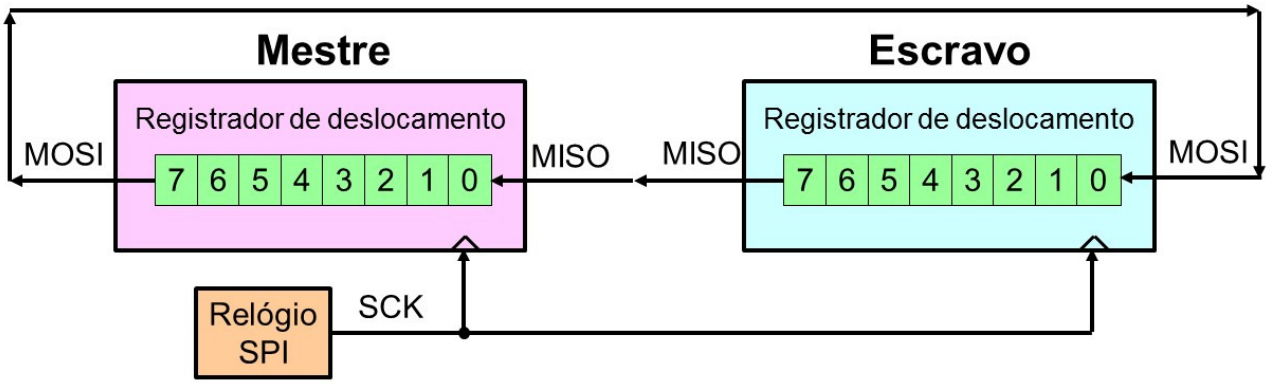
MSP recebendo dados do PCF8574

MSP430 – SPI



MSP430 – SPI

Conexão de dois Registradores de Deslocamento.



Transferência sempre em full-duplex.

Para cada bit que é enviado, um outro é recebido.

No 8º SCK, conteúdos foram trocados.

MSP430 – SPI – Polaridade e Fase

Modos de Operação do barramento SPI

UCCKPL	UCCKPH	Flanco 1	Flanco 2	Modo SPI
0	**0**	Alterar (subida, ↑)	Amostrar (descida, ↓)	0
0	**1**	Amostrar (subida, ↑)	Alterar (descida, ↓)	1
1	**0**	Alterar (descida, ↓)	Amostrar (subida, ↑)	2
1	**1**	Amostrar (descida, ↓)	Alterar (subida, ↑)	3

MSP430 – SPI – Polaridade e Fase

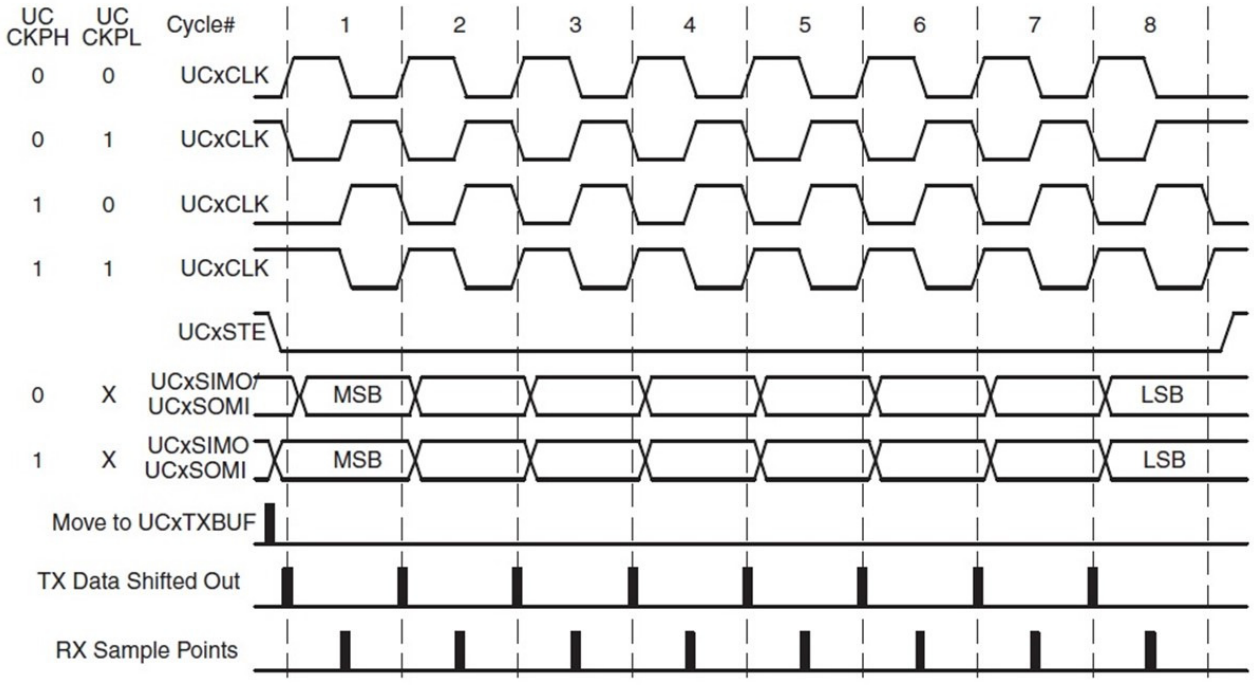


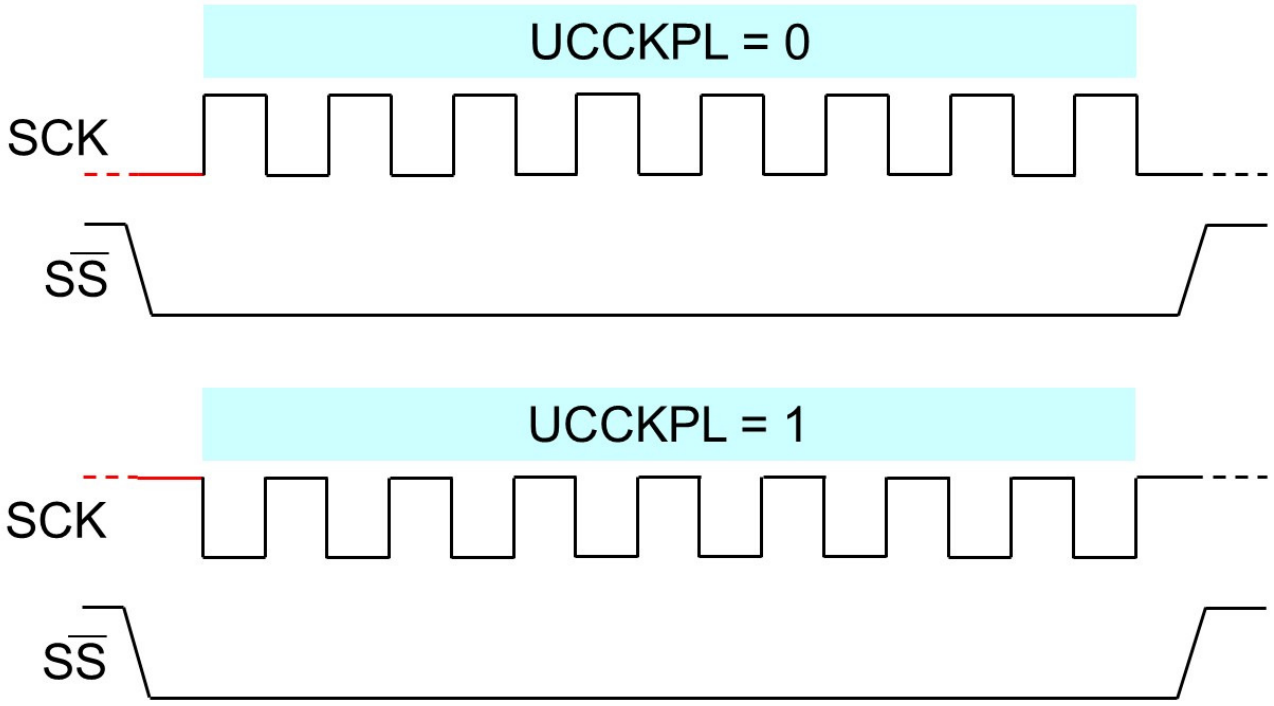
Figure 37-4. USCI SPI Timing With UCMSB = 1

MSP430 – SPI - Polaridade

- A polaridade (**UCCKPL**) diz como a linha SCK está quando ociosa.

- **UCCKPL = 0** → quando ociosa, a linha SCK está em nível baixo e
- **UCCKPL = 1** → quando ociosa, a linha SCK está em nível alto.

MSP430 – SPI - Polaridade



MSP430 – SPI - Fase

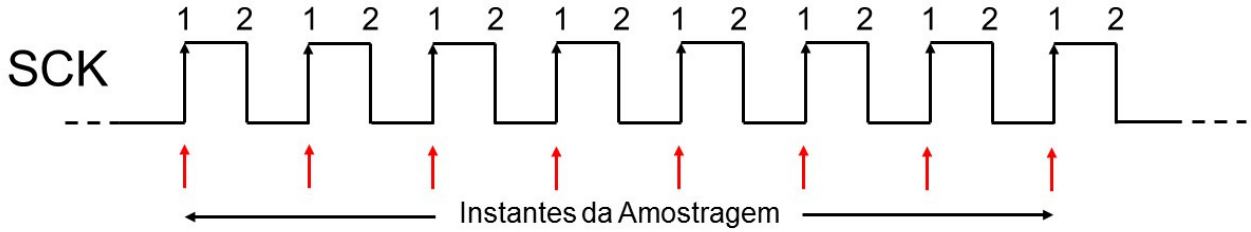
- A fase (UCCKPH) indica o instante em que a linha é amostrada.

- A definição toma como referência o primeiro flanco ou o segundo flanco do relógio (SCK).

- **UCCKPH = 1** → referência da amostragem é o primeiro flanco do relógio SCK
- **UCCKPH = 0** → referência da amostragem é o segundo flanco do relógio SCK

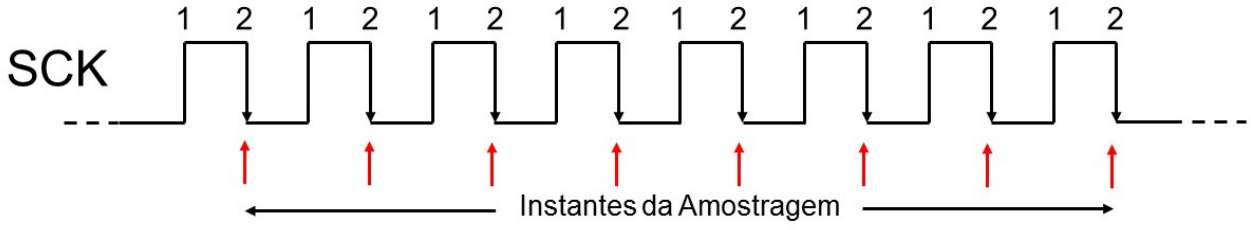
MSP430 – SPI - Fase

CCCKPL = 0 e UCCKPH = 1

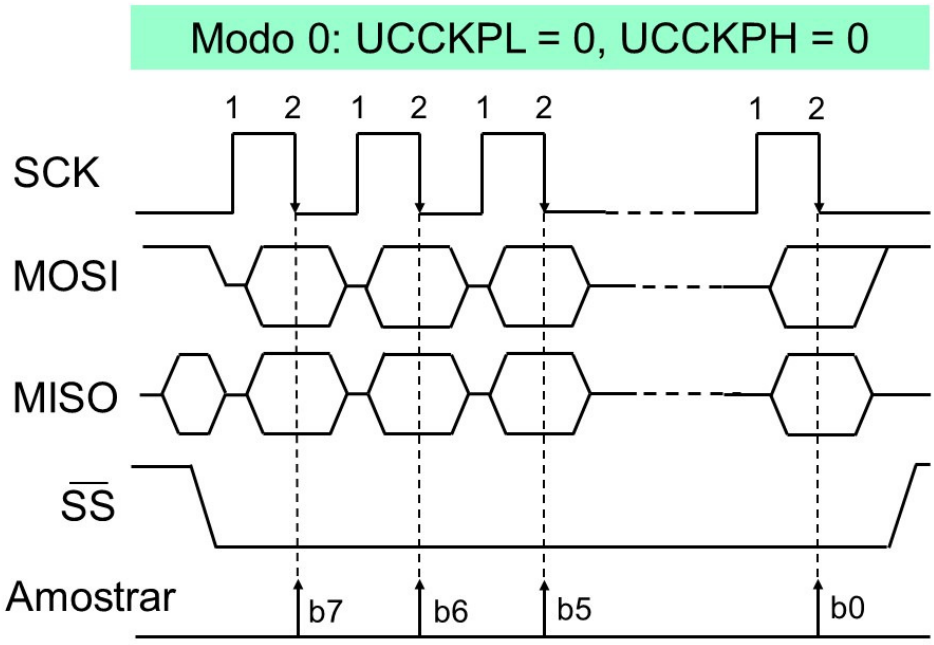


Em um flanco a linha é alterada e no outro é amostrada

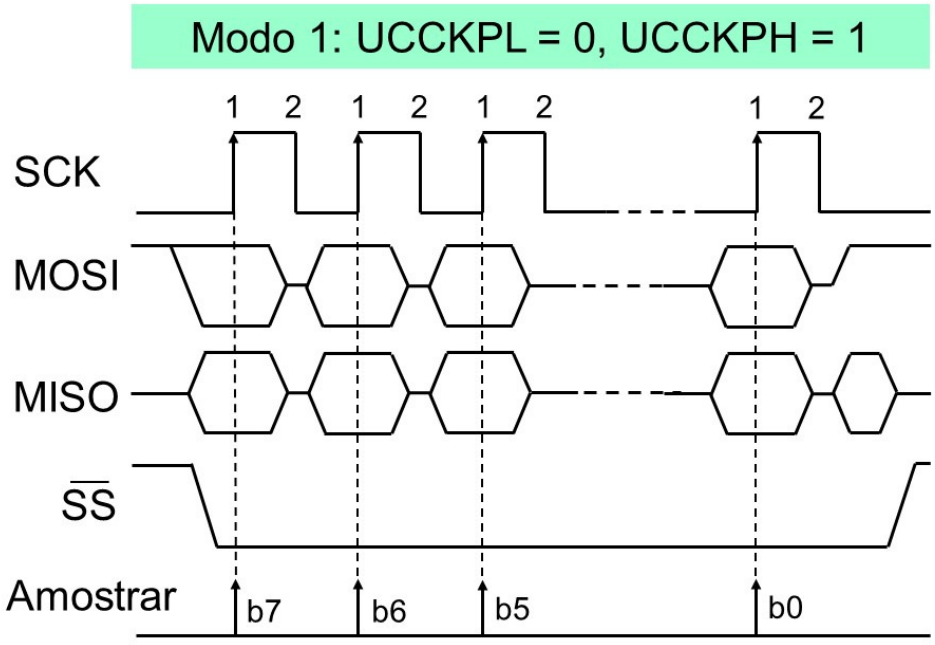
CCCKPL = 0 e UCCKPH = 0



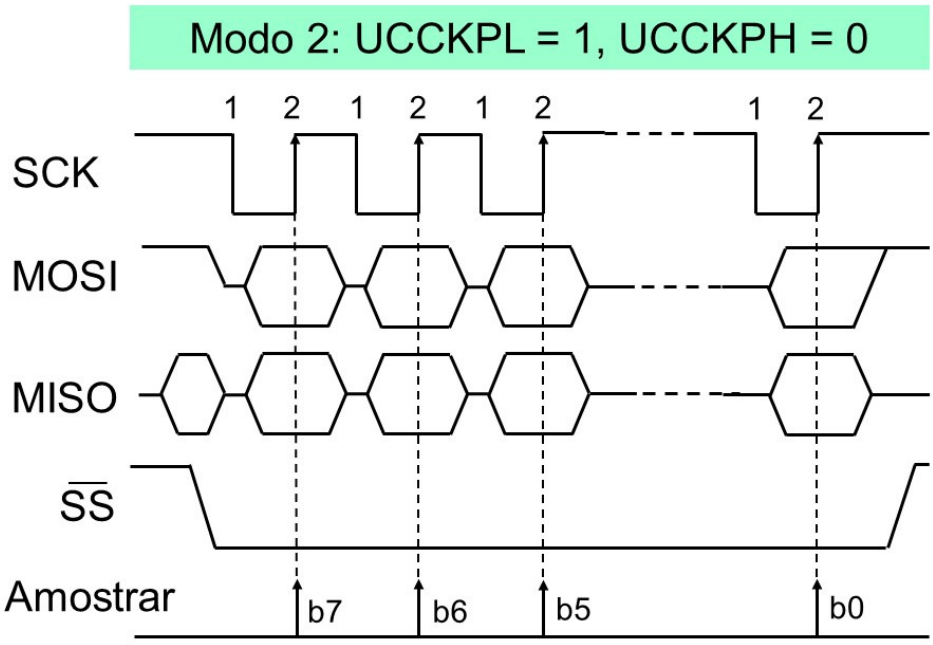
MSP430 – SPI – Polaridade e Fase



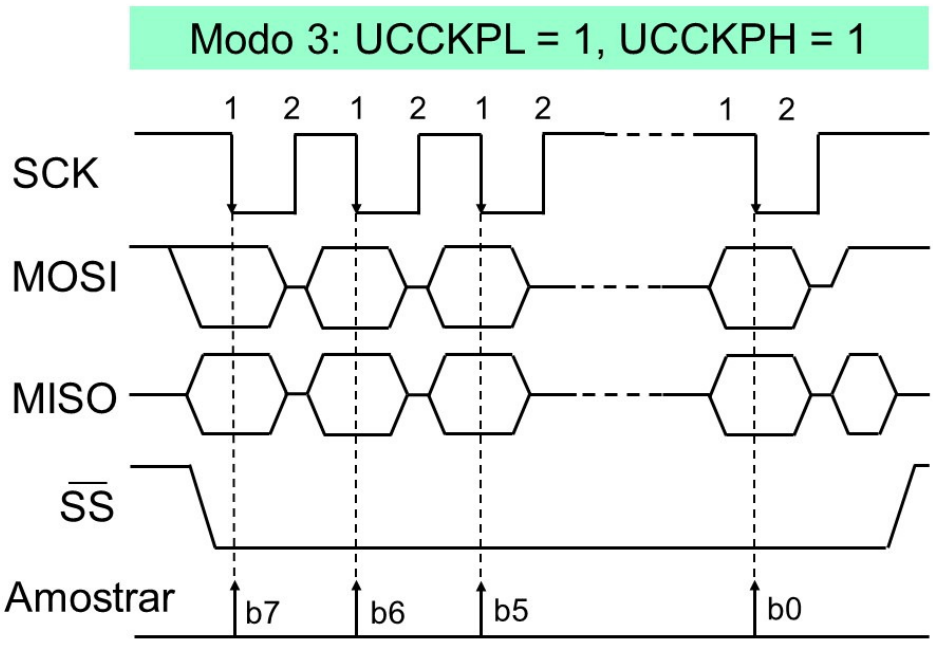
MSP430 – SPI – Polaridade e Fase



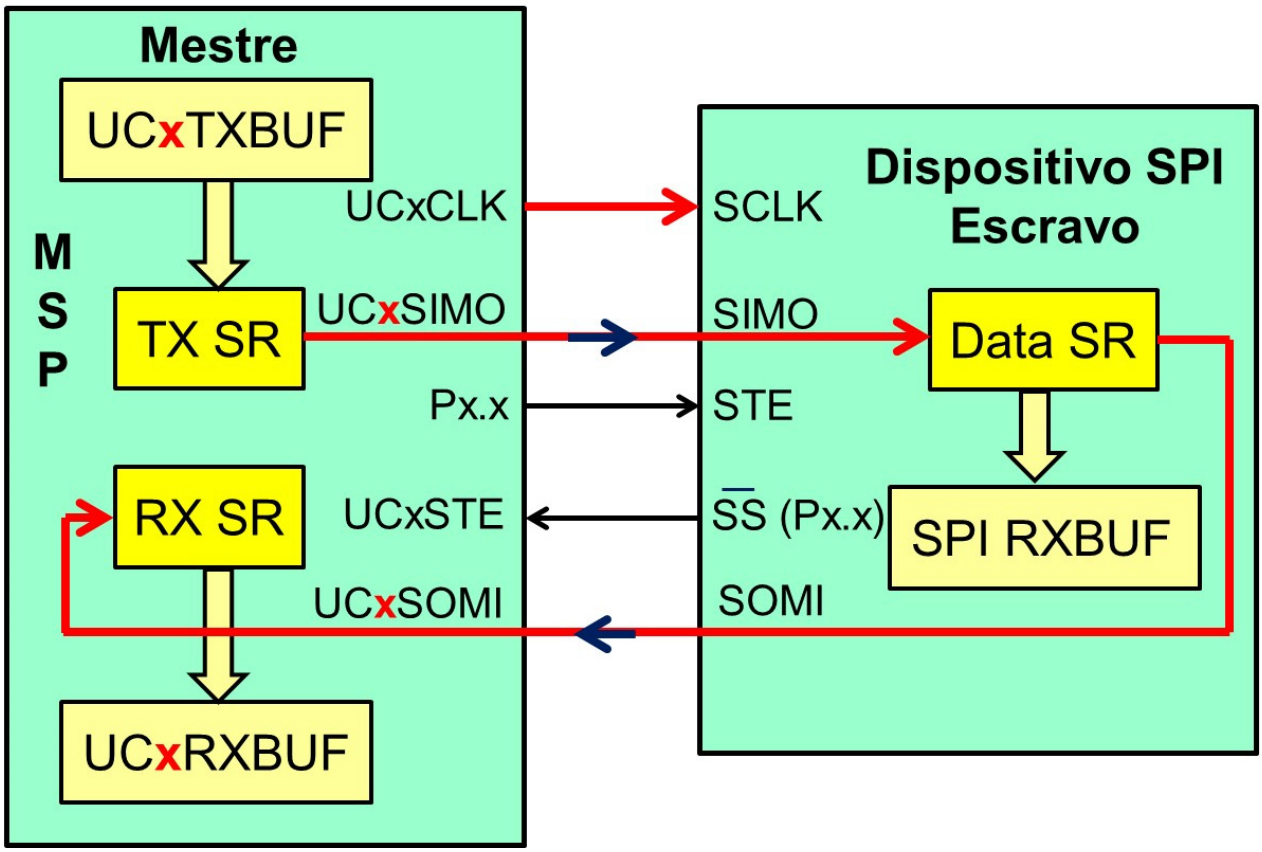
MSP430 – SPI – Polaridade e Fase



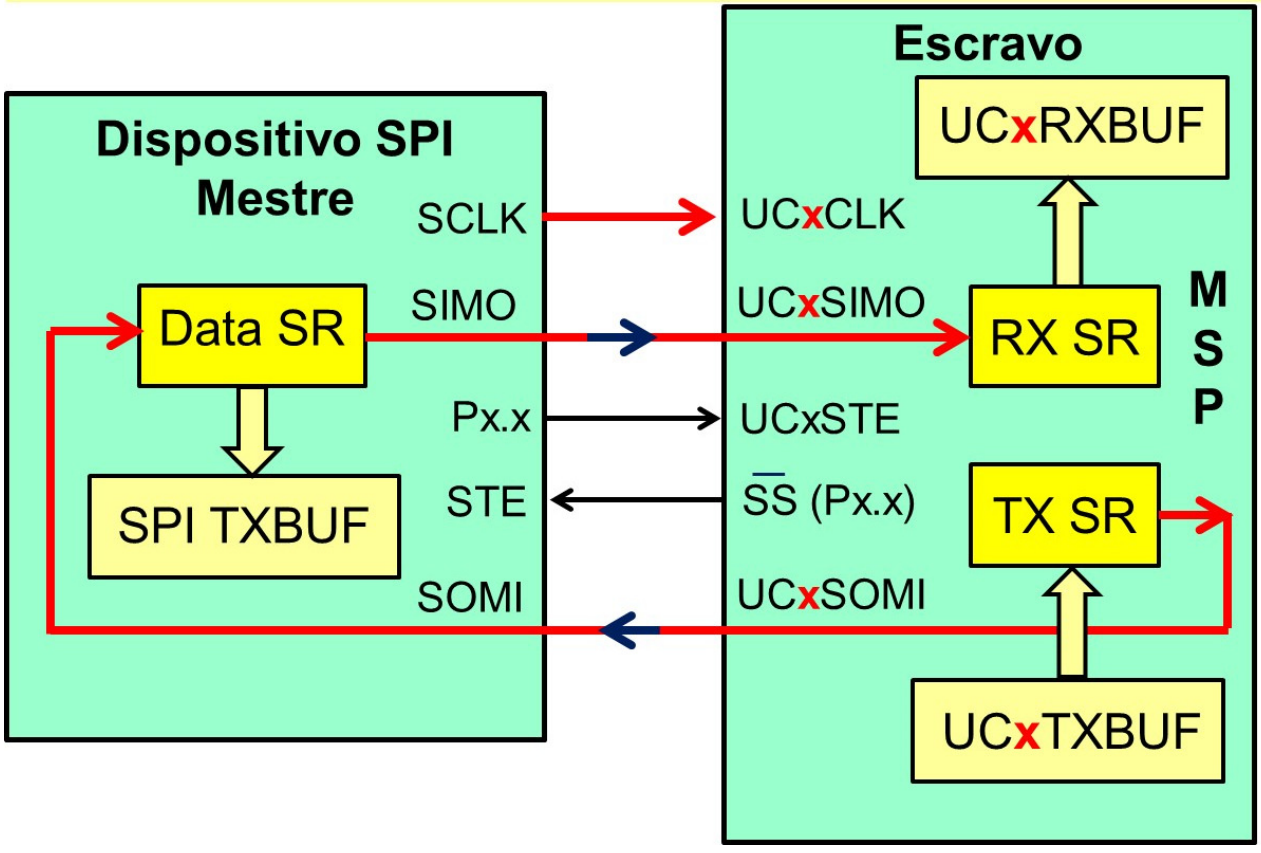
MSP430 – SPI – Polaridade e Fase



MSP430 – SPI - MSP MESTRE

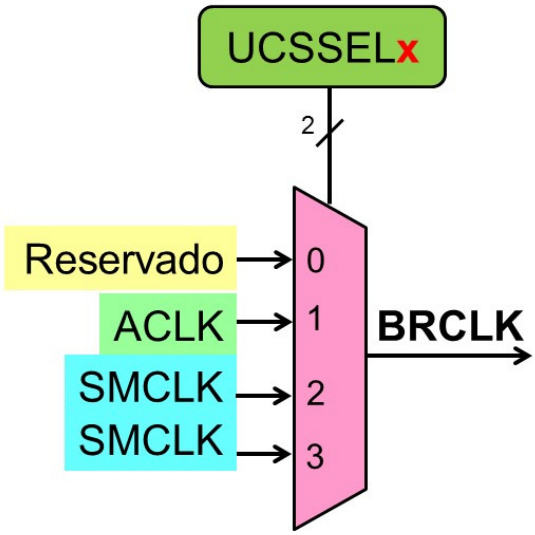


MSP430 – SPI - MSP ESCRAVO



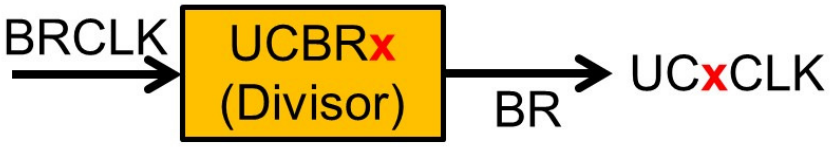
MSP430 – USCI – SPI – CLOCK

Seleção do BRCLK (já visto)



MSP430 – USCI – SPI – CLOCK

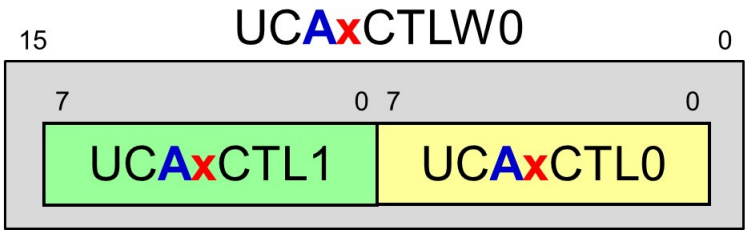
Modo Mestre, UCMST = 1



N = BRCLK / BR

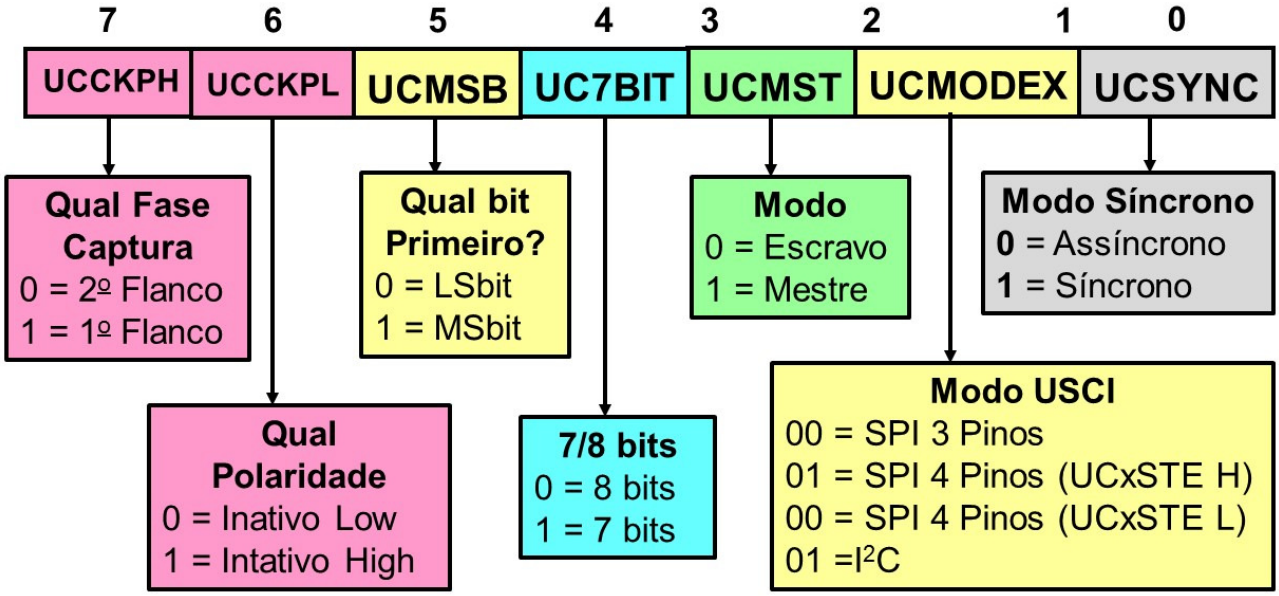
UCBRx = INT (N)

MSP430 – Registradores USCI SPI



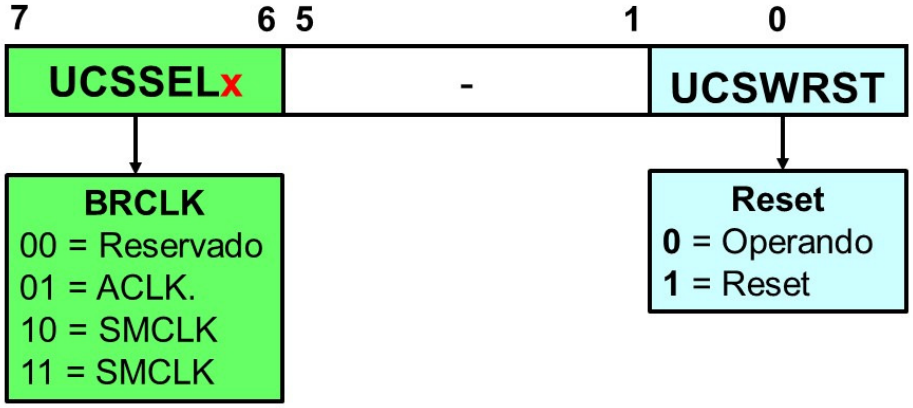
MSP430 – Registradores USCI SPI

UCAxCTL0 → Controle 0 do USCI_Ax



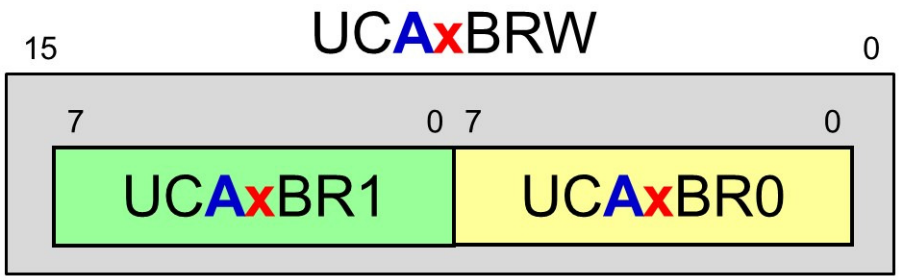
MSP430 – Registradores USCI SPI

UCAxCTL1 → Controle 1 do USCI_Ax



MSP430 – Registradores USCI SPI

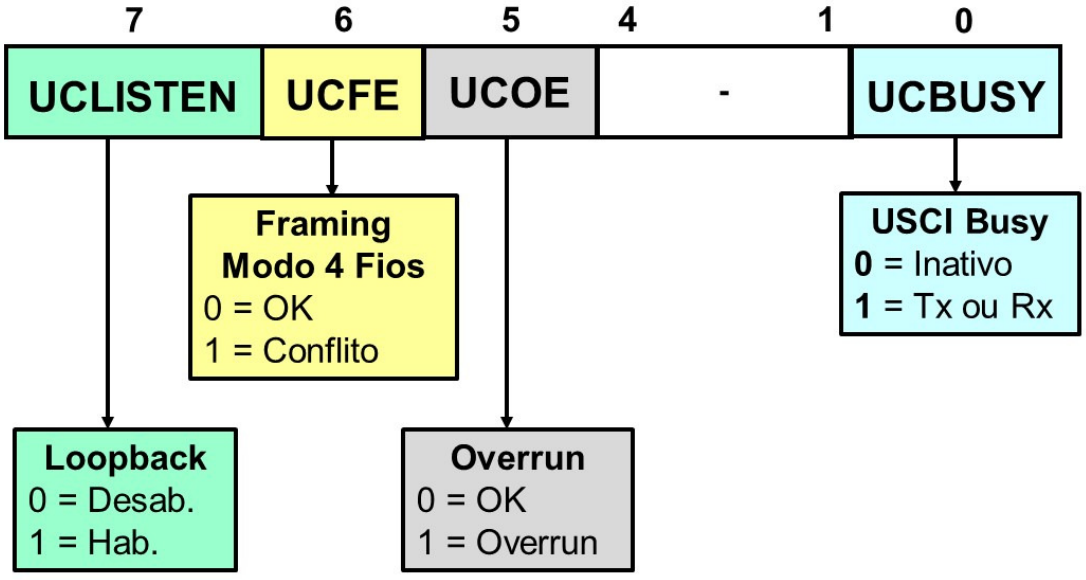
UCAxBRW → Palavra de Controle do BR



Este registrador (16 bits) é o divisor UCBRx

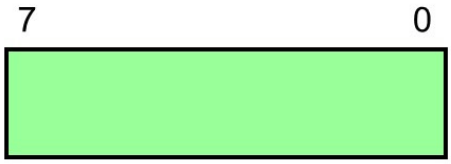
MSP430 – Registradores USCI SPI

UCAxSTAT → Registrador de Status do USCI_Ax



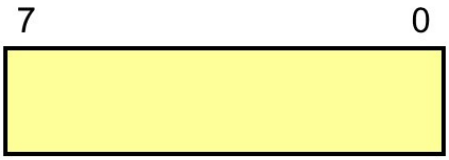
MSP430 – Registradores USCI SPI

UCAxRXBUF → Buffer de Recepção



- Contém o último dado recebido.
- Leitura apaga diversos flags de erros.

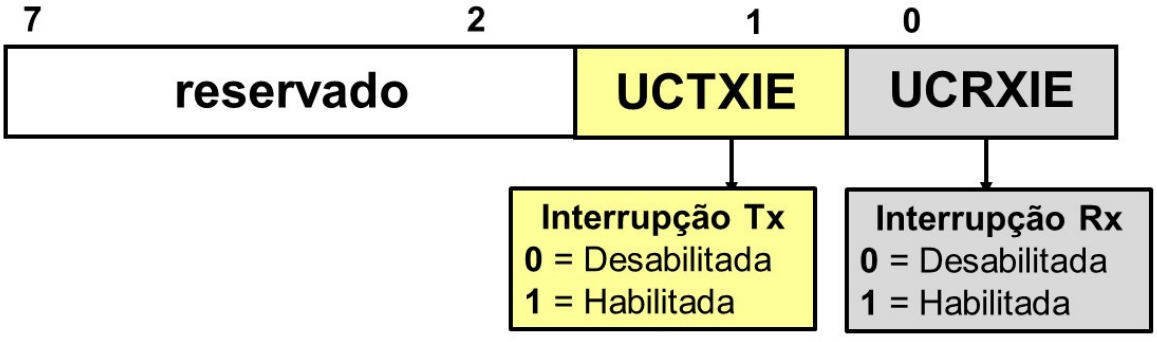
UCAxTXBUF → Buffer de Recepção



- Contém o dado a ser transmitido.
- Escrita apaga flag UCTXIFG.

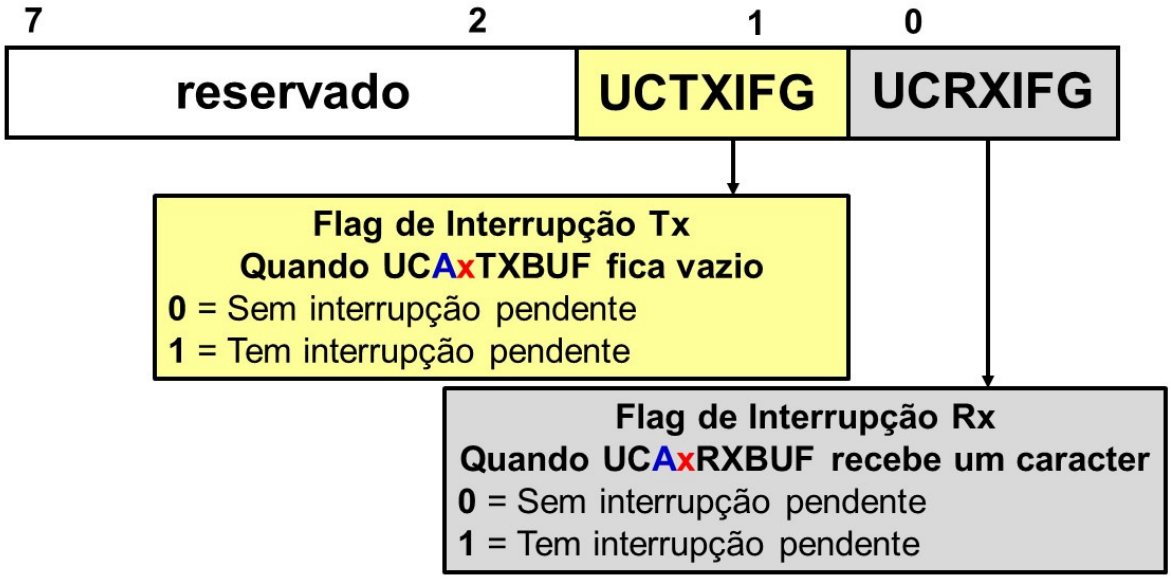
MSP430 – Registradores USCI SPI

UCAxIE → Registrador Habilita Interrupção



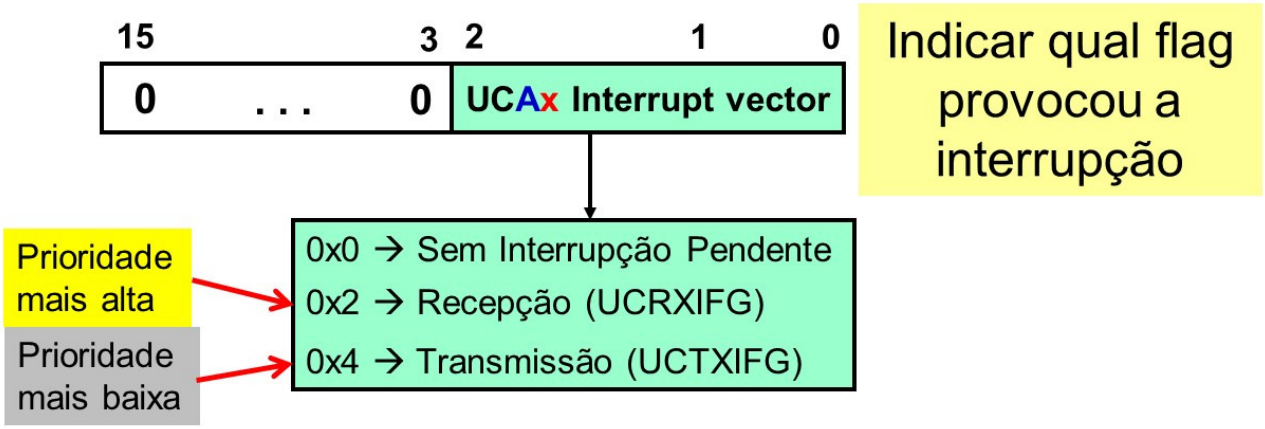
MSP430 – Registradores USCI SPI

USCAxIFG → Registrador de Flags de Interrupção

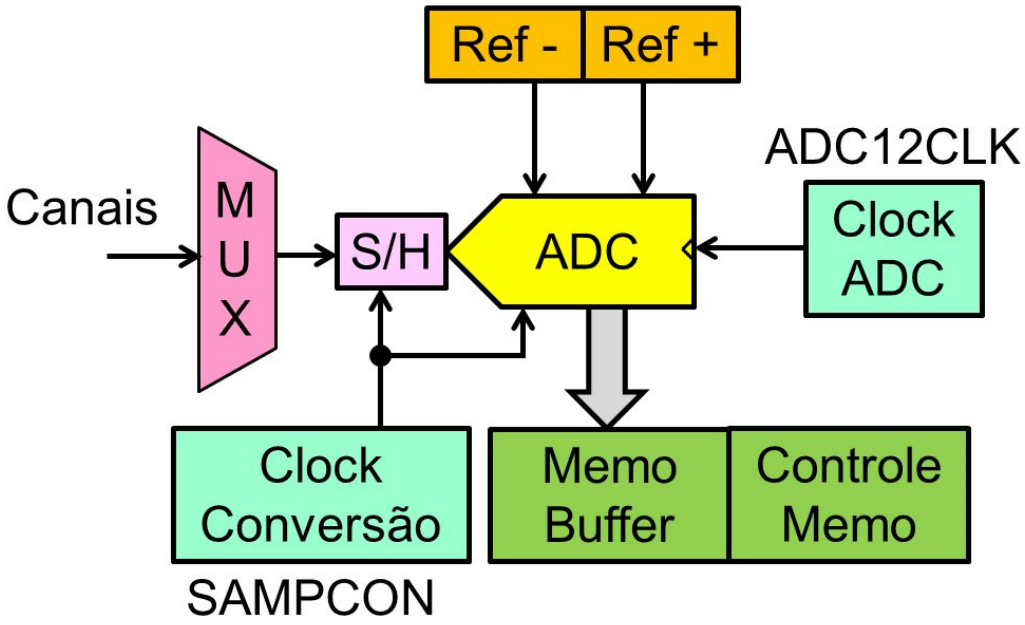


MSP430 – Registradores USCI SPI

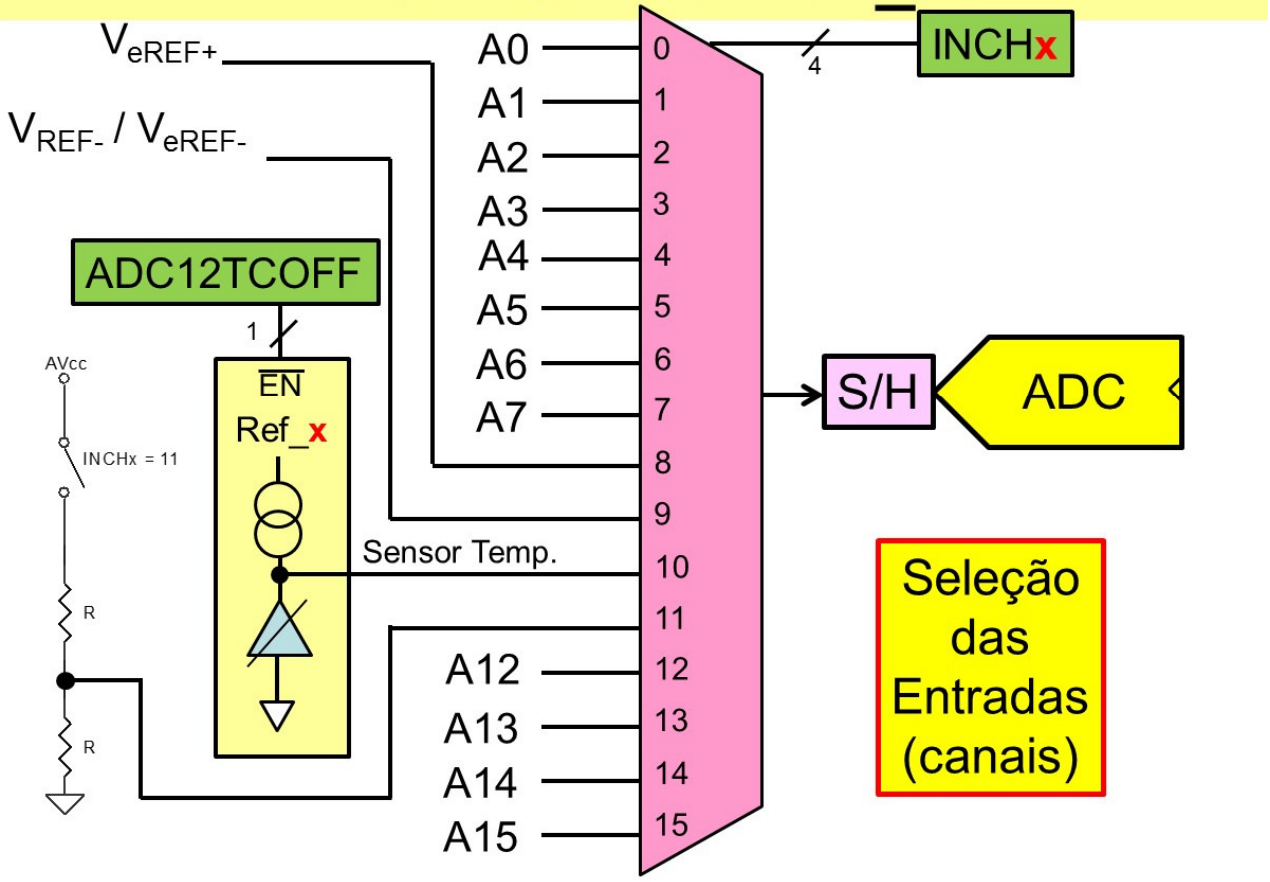
UCAxIV → Vetor de Interrupção do UCA



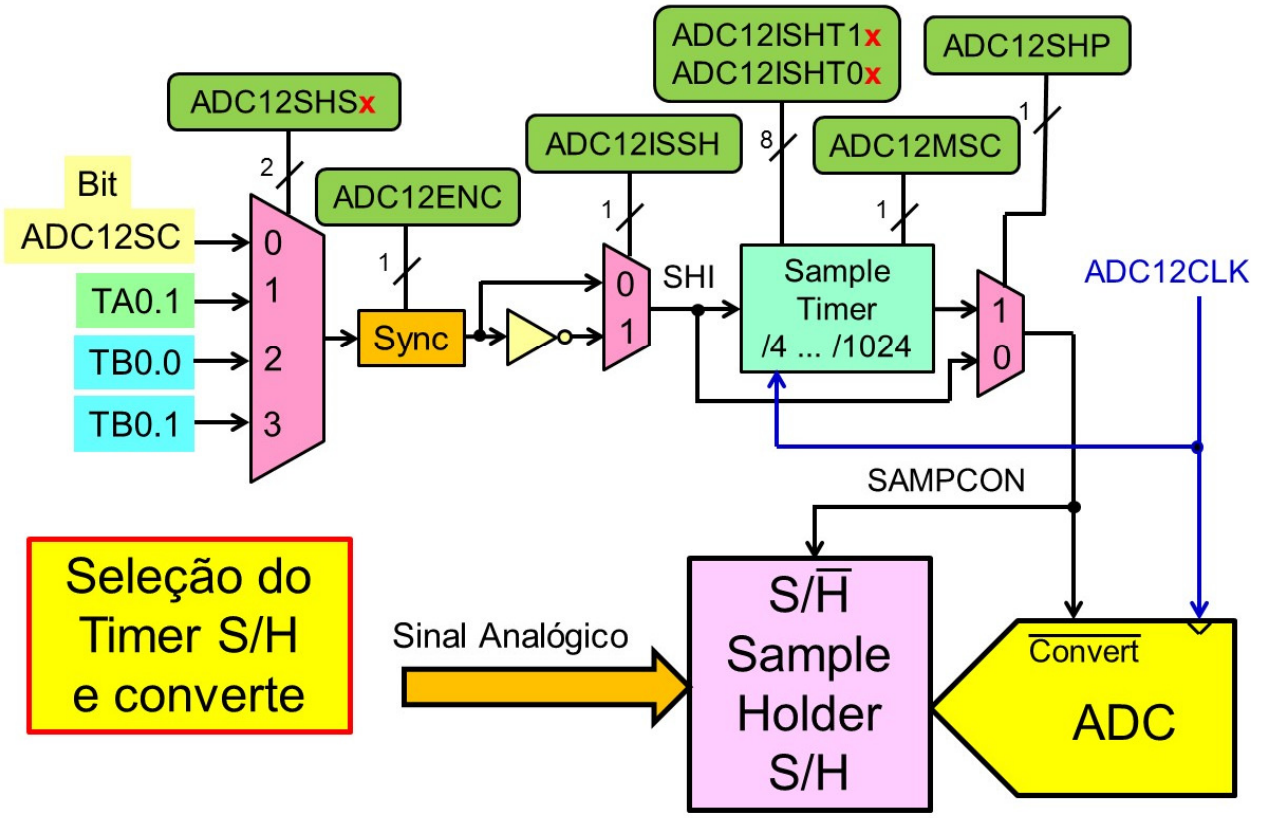
MSP430 – ADC12_A



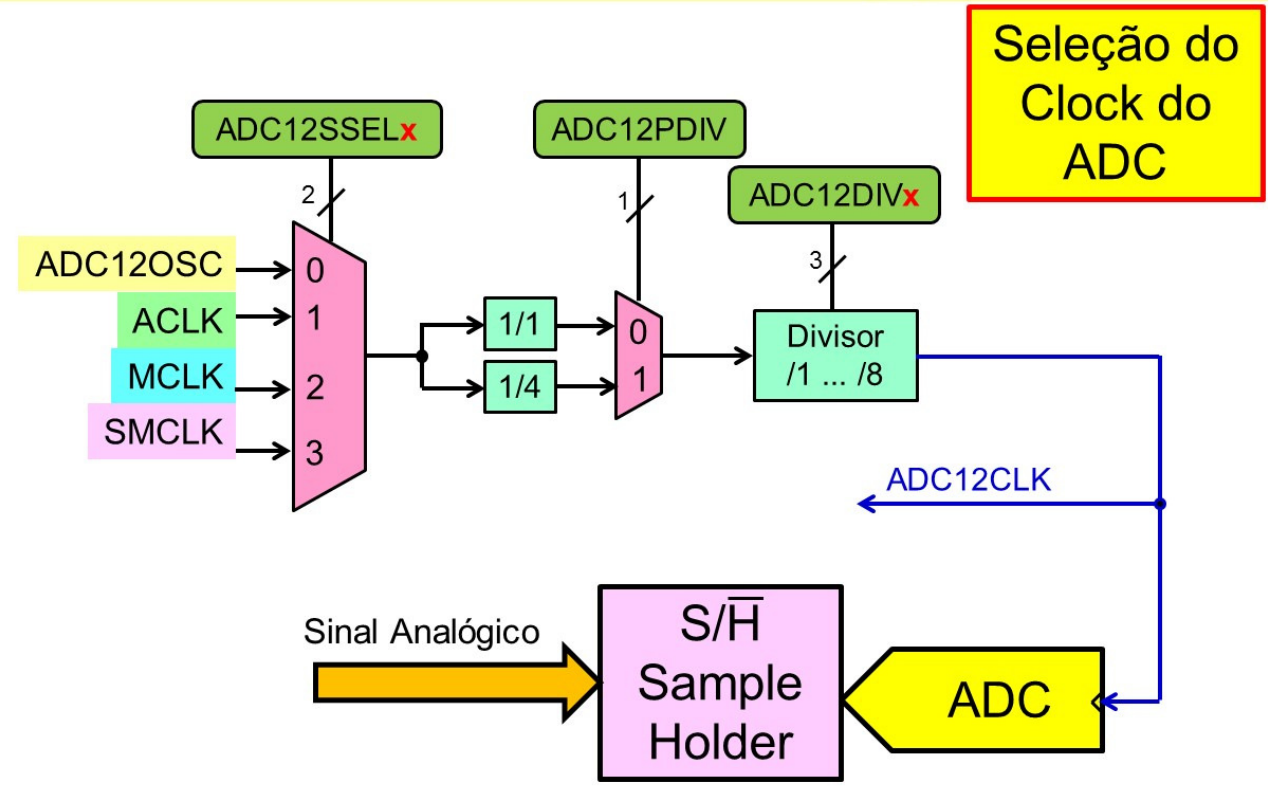
MSP430 – ADC12_A



MSP430 – ADC12_A

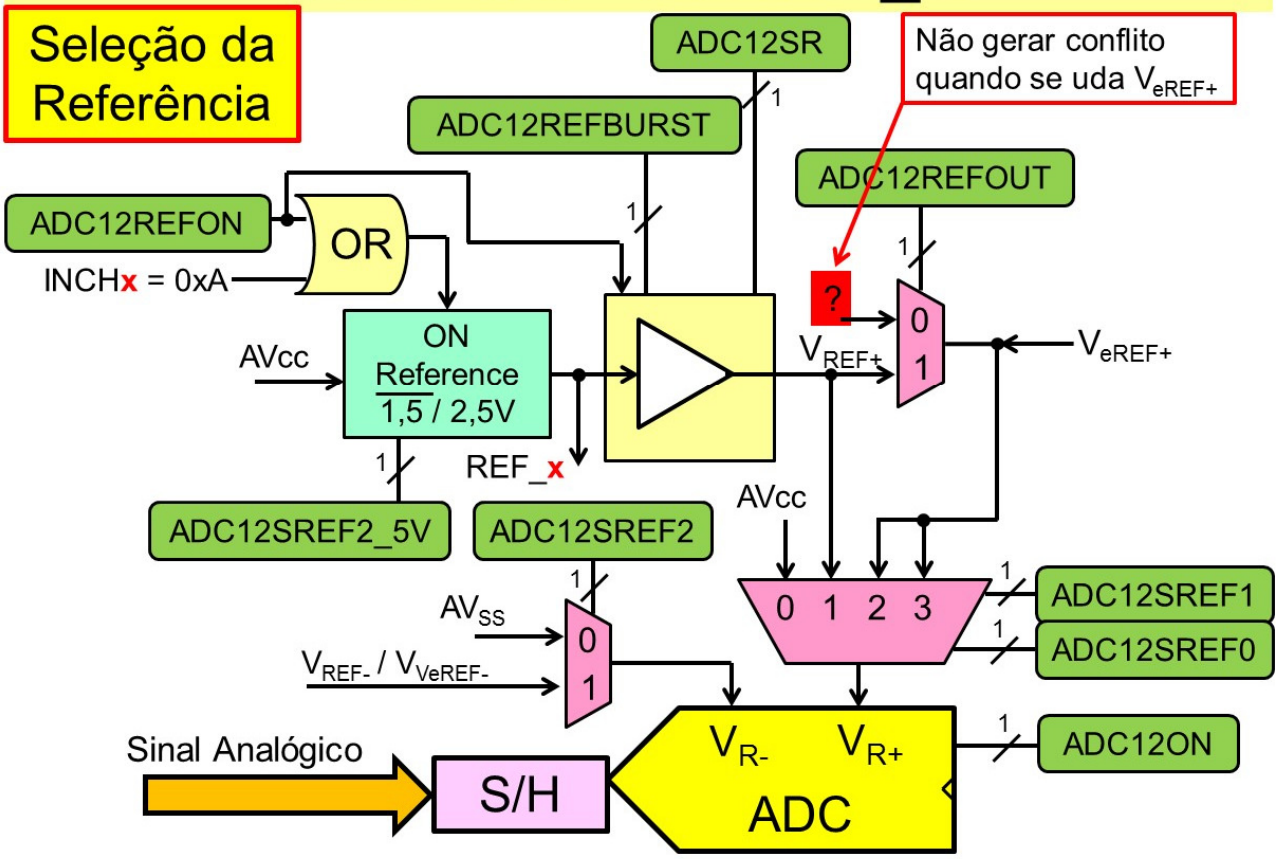


MSP430 – ADC12_A

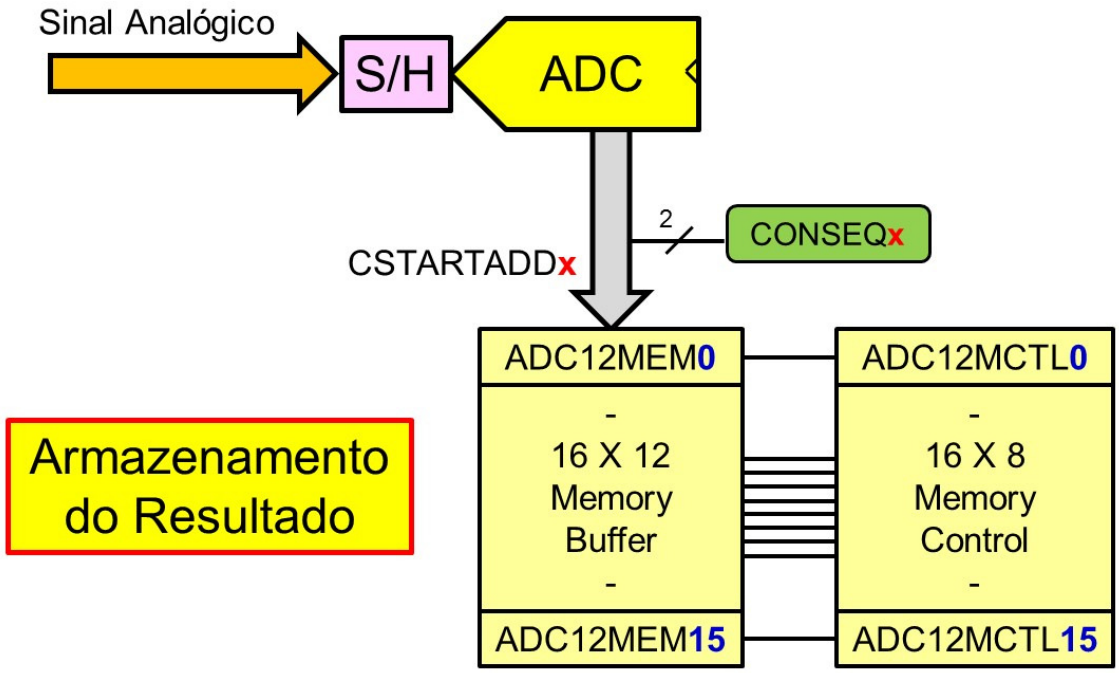


MSP430 – ADC12_A

Seleção da Referência



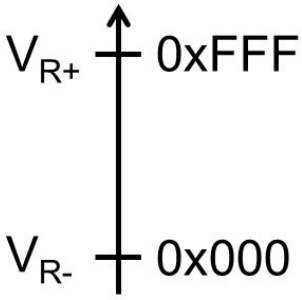
MSP430 – ADC12_A



Armazenamento do Resultado

MSP430 – ADC12_A – Operação

- ADC de 12 bits → N_{ADC} (0 até 0xFFF).
- Armazena na Memória de Conversão.
- V_{R+} → limite superior de conversão.
- V_{R-} → limite inferior de conversão.
- Se **Vin ≥ V_{R+}** → N_{ADC} = 0xFFF.
- Se **Vin ≤ V_{R-}** → NADC = 0x000.



$$N_{ADC} = 4095\,\frac{Vin - V_{R-}}{V_{R+} - V_{R-}}$$

MSP430 – ADC12_A

Formatos de Conversão

ADC12DF = 0 → alinhado pela direita

ADC12DF = 1 → alinhado pela esquerda em comp. a 2.

Table 28-1. ADC12_A Conversion Result Formats

Analog Input Voltage	ADC12DF	ADC12RES	Ideal Conversion Results	ADC12MEMx
$-V_{REF}$ to $+V_{REF}$	0	00	0 to 255	0000h to 00FFh
	0	01	0 to 1023	0000h to 03FFh
	0	10	0 to 4095	0000h to 0FFFh
	1	00	-128 to 127	8000h to 7F00h
	1	01	-512 to 511	8000h to 7FC0h
	1	10	-2048 to 2047	8000h to 7FF0h

8 bits	10 bits
-128 = 0x80 → 0x8000	-512 = 0x200 → 0x8000
+127 = 0x7F → 0x7F00	+511 = 0x3FF → 0x7FC0

MSP430 – ADC12_A

Modo Amostragem Estendida — ADC12SHP = 0

- SHI controla SAMPCON e determina o intervalo de amostragem.

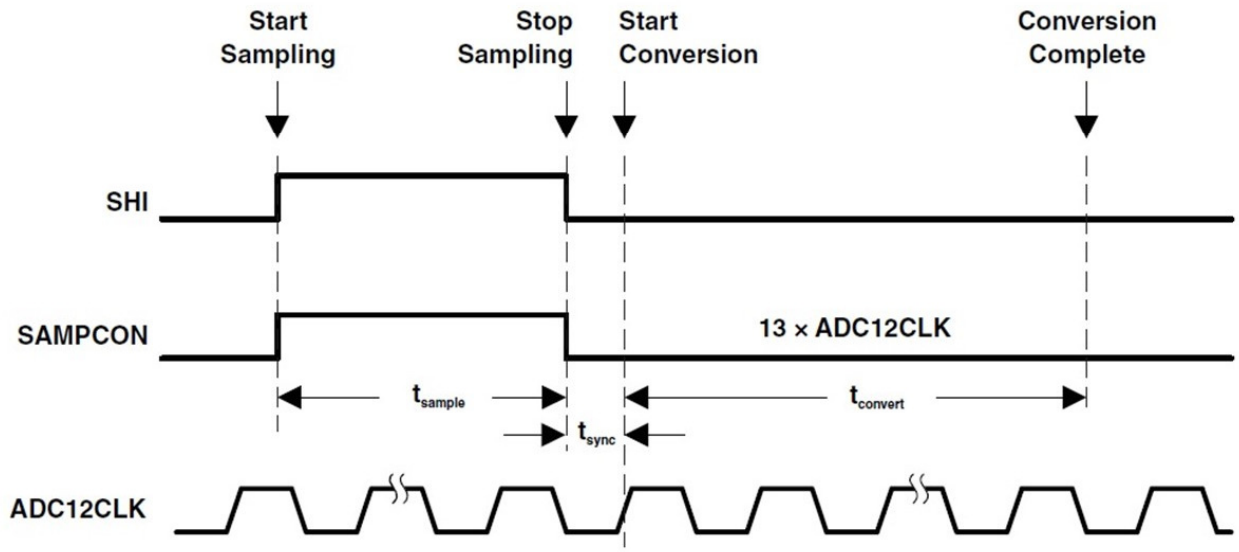


Figure 28-4. Extended Sample Mode

MSP430 – ADC12_A

Modo Amostragem com Pulso — ADC12SHP = 1

- SHI dispara o temporizador de amostragem, que controla o sinal SAMPCON.

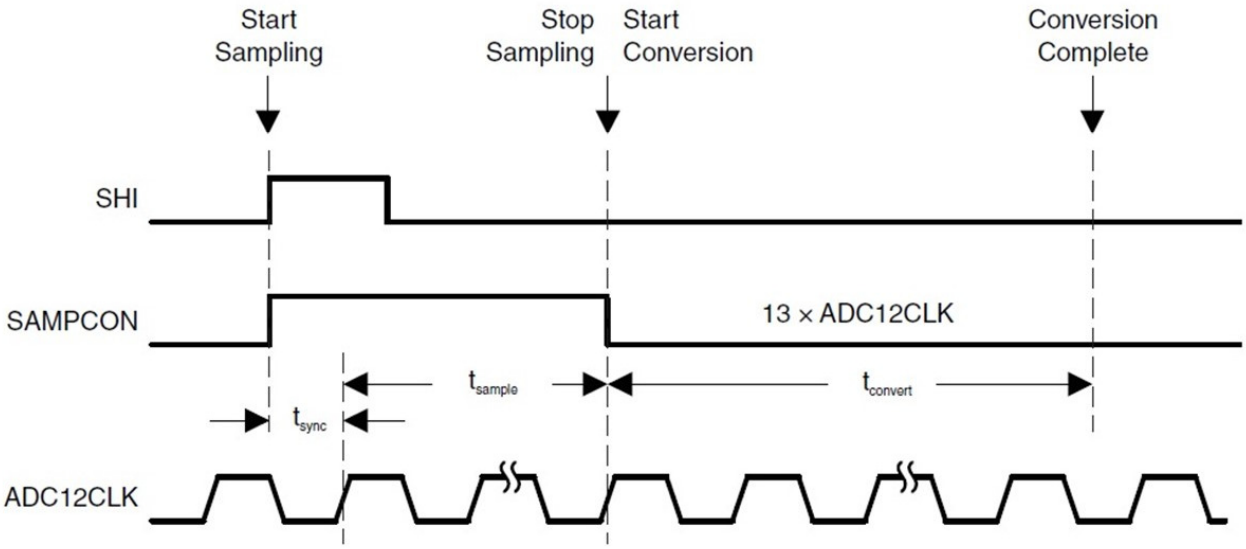


Figure 28-5. Pulse Sample Mode

MSP430 – ADC12_A

Considerações sobre intervalo de amostragem

- SAMPCON = 0 → Entradas Ax em Hi-Z.
- SAMPCON = 1 → Entradas tem modelo abaixo.
- Capacitor do S/H precisa de tempo para se carregar de tal forma que $V_S - V_C < ½$ bit V_S.

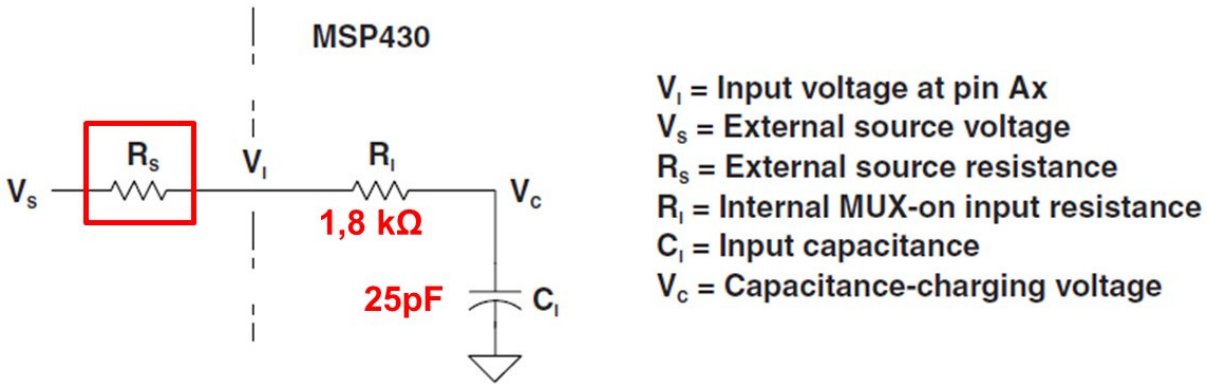


Figure 28-6. Analog Input Equivalent Circuit

MSP430 – ADC12_A

Considerações sobre intervalo de amostragem

- Cálculo do tempo de amostragem depende de R_S

$$t_{sample} > (R_S + R_I) \times ln(2^{n+1}) \times C_I + 800\ ns$$

- Substituindo os valores:

$$t_{sample} > (R_S + 1{,}8\ k) \times ln(2^{n+1}) \times 25\ p + 800\ ns$$

- Por exemplo, para R_S = 10 kΩ e n = 12 bits:

$$t_{sample} > 3{,}46\ \mu s$$

MSP430 – ADC12_A

Referência de Tensão

- V_{R+} → limite superior de conversão.
- V_{R-} → limite inferior de conversão.

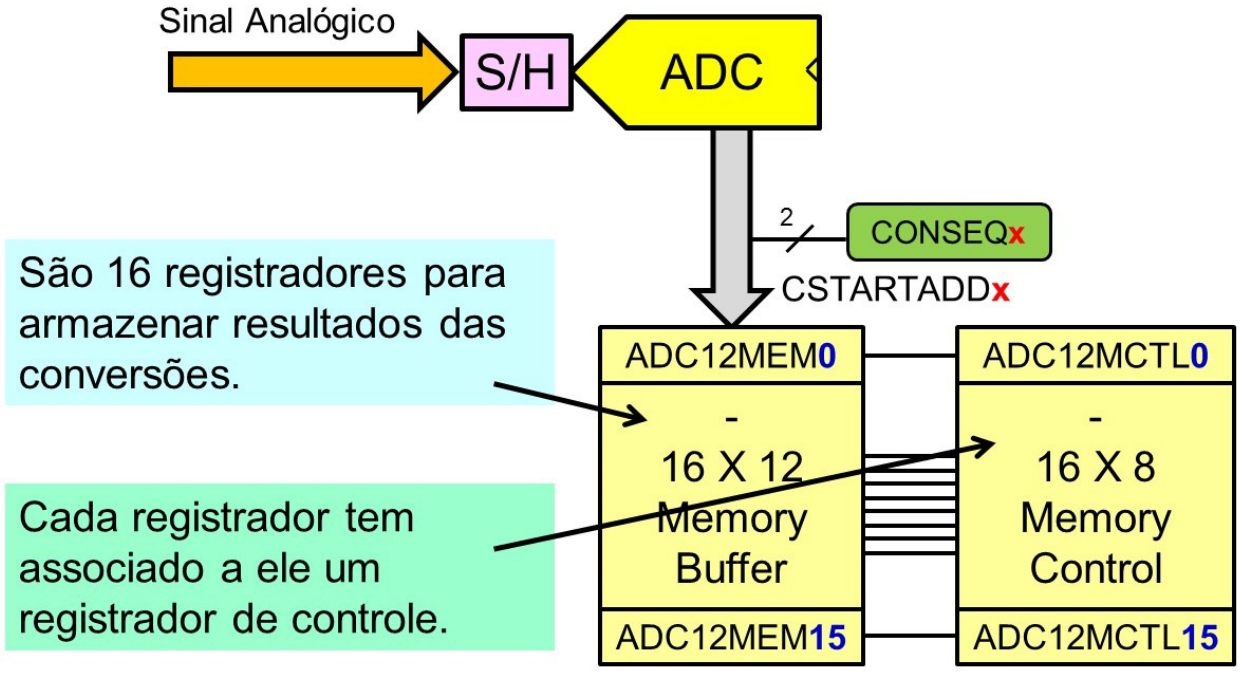
- Podem ser internas ou externas.

- V_{R+} → (interno) 1,5 V, 2,5 V ou AVcc .
- V_{R-} → (interno) AVss.

- ADC trabalha com : (ADC12RES)
  - 8 bits (9 ciclos ADC12CLK)
  - 10 bits (11 ciclos ADC12CLK)
  - 12 bits (13 ciclos ADC12CLK)

MSP430 – ADC12_A

Memória de Conversão



MSP430 – ADC12_A

Modos de Conversão

ADC12CONSEQx	Modo	Operação
00	Um canal, uma conversão	Um único canal é convertido uma única vez
01	Seq. de Canais (Autoscan)	Uma seq. de canais é convertido uma única vez
10	Repete um canal	Um único canal é convertido repetidamente
11	Repete Seq. de Canais (Rep. Autoscan)	Uma seq. de canais é convertida repetidamente

MSP430 – ADC12_A

Memória de Conversão

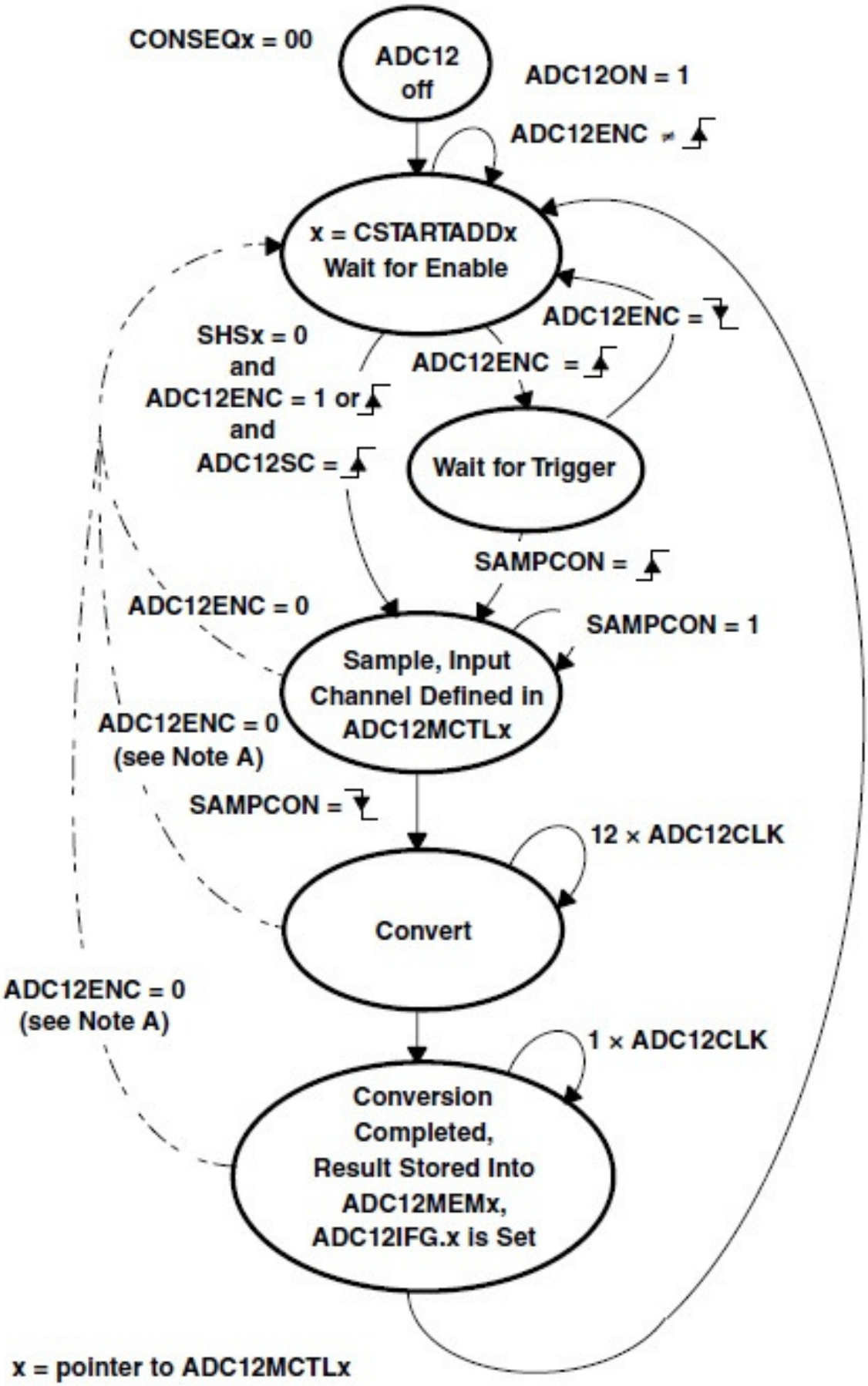
Bit ADC12EOS = 1 → define último registrador de uma conversão em sequência.

Se DC12MEM**15** tem Bit ADC12EOS=0, a sequência rola de volta para ADC12MEM**0**.

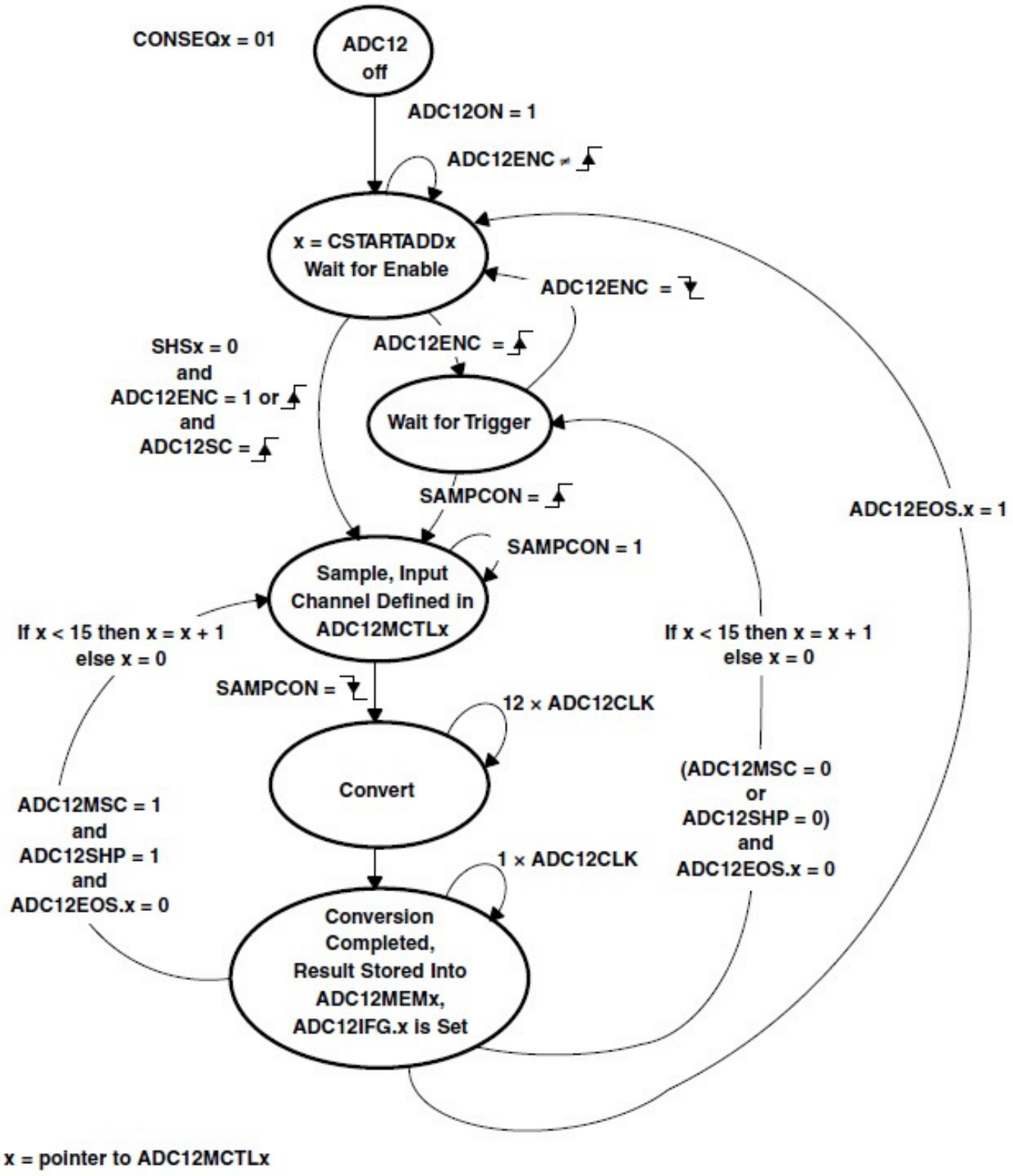
Bit CSTARTADD**x** = 1 → define o primeiro registrador de sequência.

Se está no modo **único canal** simples ou repetido, o registrador marcado com CSTARTADD**x** = 1 é o único usado.

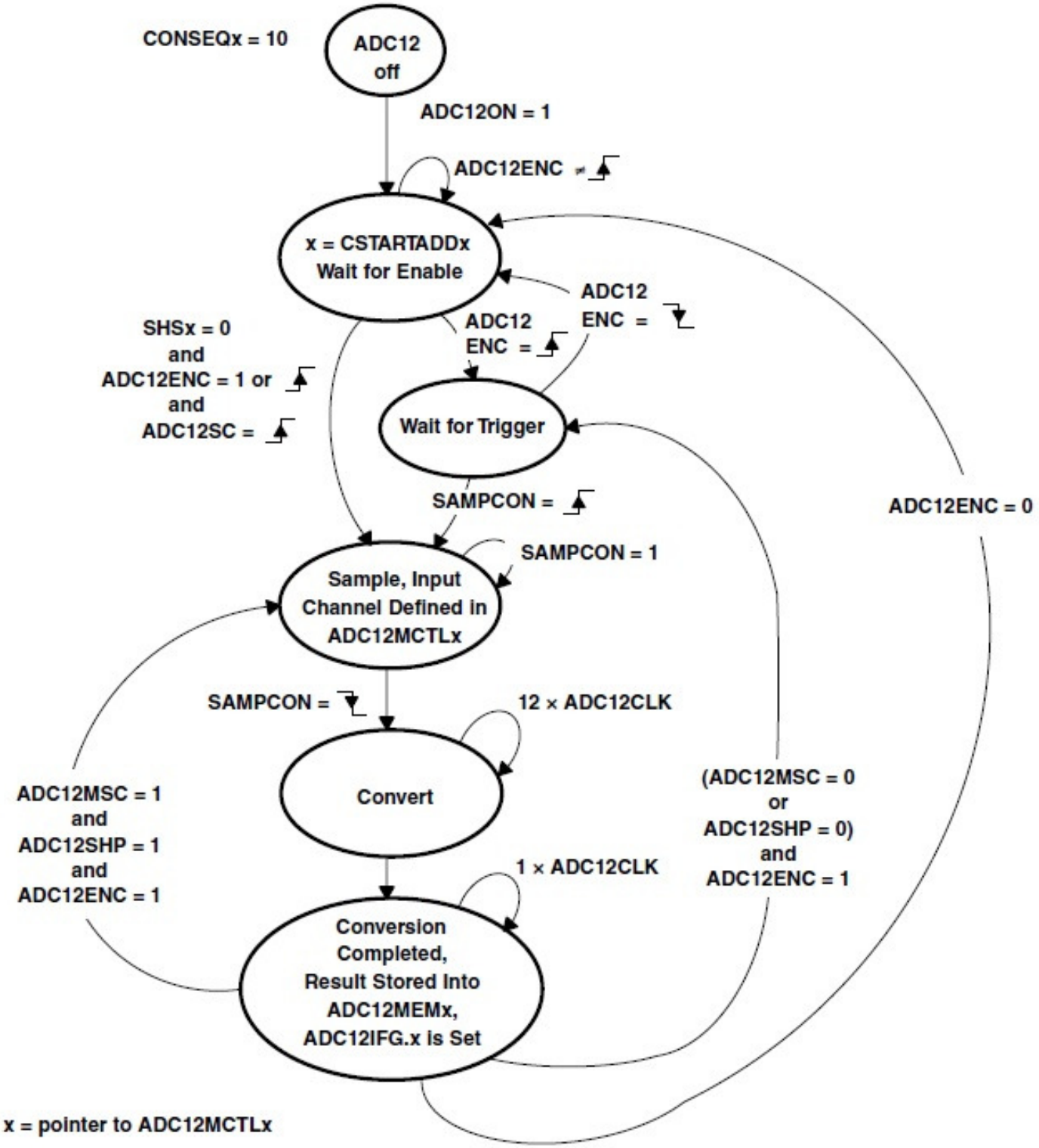
ADC12_A – Modo Simples



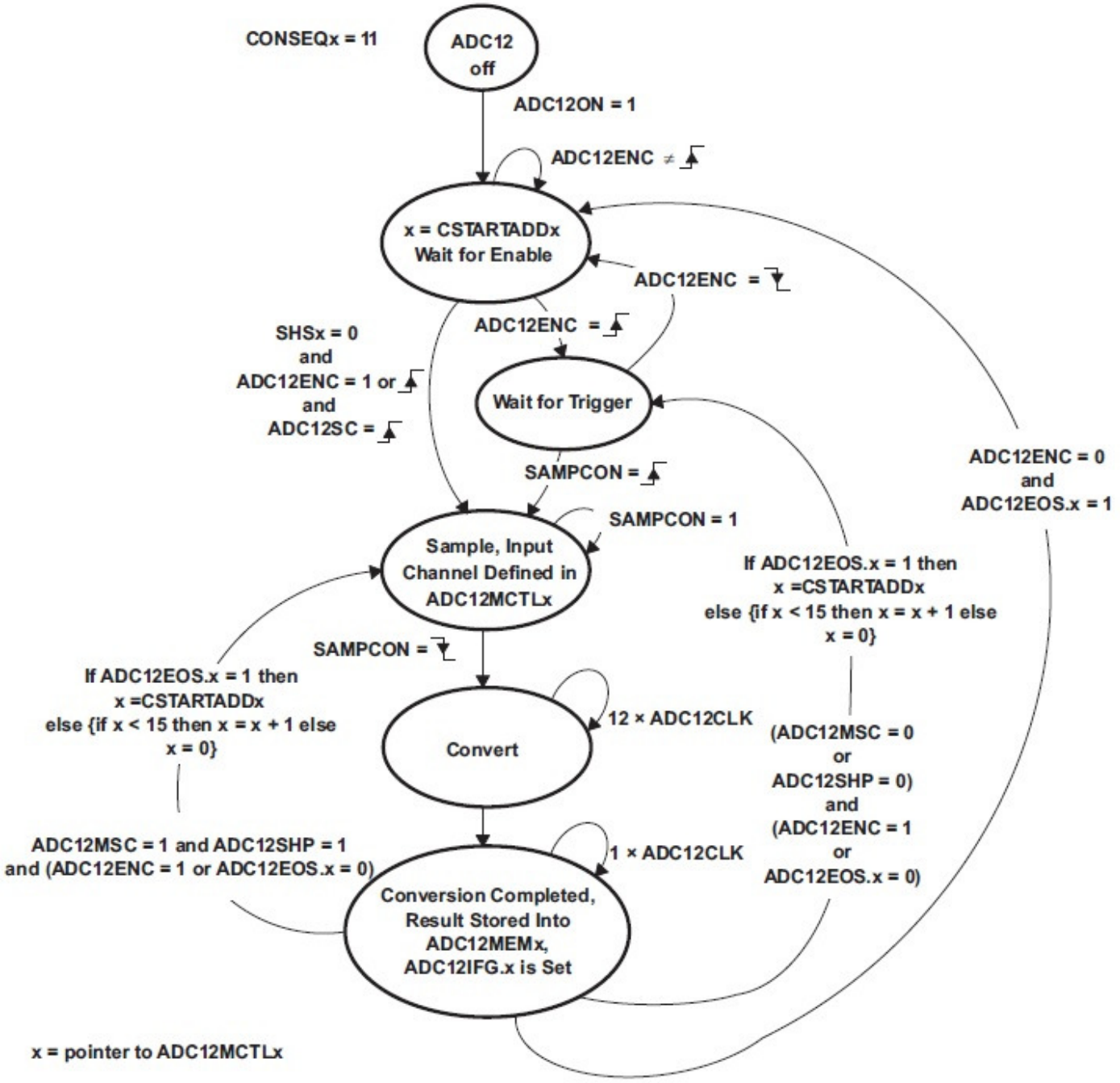
ADC12_A – Modo Sequência (Autoscan)



ADC12_A – Modo Simples Repetido



ADC12_A – Modo Sequência Repetido (Repeat Autoscan)



MSP430 – ADC12_A - Registradores

Offset	Acrônimo	Nome	Word 16
0	ADC12CTL**0**_L	Controle 0	ADC12CTL**0**
1	ADC12CTL**0**_H		
2	ADC12CTL**1**_L	Controle 1	ADC12CTL**1**
3	ADC12CTL**1**_H		
4	ADC12CTL**2**_L	Controle 2	ADC12CTL**2**
5	ADC12CTL**2**_H		
A	ADC12IFG_L	Interrupt Flag	ADC12IFG
B	ADC12IFG_H		
C	ADC12IE_L	Interrupt Enable	ADC12IE
D	ADC12IE_H		
E	ADC12IV_L	Interrupt Vector	ADC12IV
F	ADC12IV_H		

MSP430 – ADC12_A - Interrupções

- ADC12IV → Gerador de Vetor de Interrupção.

- Cuidado → Acesso ao ADC12IV:
- Não zera ADC12IFG**x**,
- Porém, pode zerar ADC12OV ou ADC12TOV.

- ADC12IFG**x**= zerado quando ADC12MEM**x** correspondente é acessado ou por software.

- ADC12IFG**x**=1 → resultado escrito no ADC12MEM**x**.

- ADC12OV = 1 → escrita numa posição antes que o valor prévio seja lido.

- ADC12TOV = 1 → novo pedido de amostrar-e-converter chega antes de terminar a atual conversão.

MSP430 – ADC12_A - Registradores

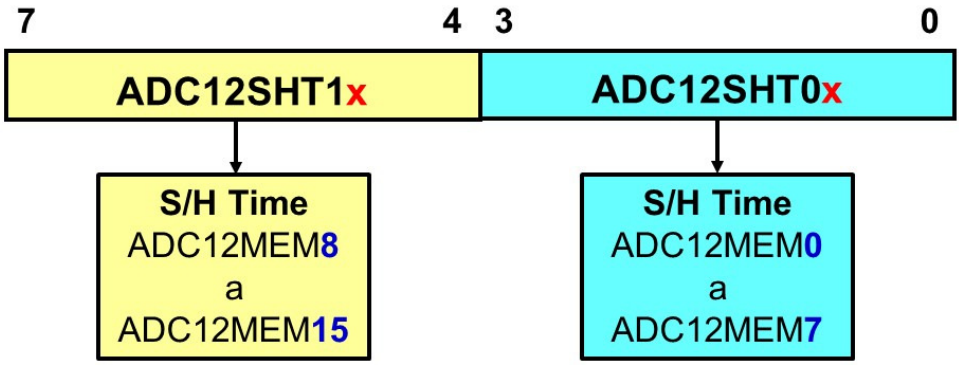
Offset	Acrônimo	Nome	Word 16
20	ADC12MEM0_L	ADC12_A Memory 0	ADC12MEM0
21	ADC12MEM0_H		
22	ADC12MEM1_L	ADC12_A Memory 1	ADC12MEM1
23	ADC12MEM1_H		
24	ADC12MEM2_L	ADC12_A Memory 2	ADC12MEM2
25	ADC12MEM2_H		
...	...	...	...
...	...		
3C	ADC12MEM14_L	ADC12_A Memory 4	ADC12MEM4
3D	ADC12MEM14_H		
3E	ADC12MEM15_L	ADC12_A Memory 5	ADC12MEM5
3F	ADC12MEM15_H		

MSP430 – ADC12_A - Registradores

Offset	Acrônimo	Nome	W16
10	ADC12MCTL0	ADC12_A Memory Control 0	-
11	ADC12MCTL1	ADC12_A Memory Control 1	-
12	ADC12MCTL2	ADC12_A Memory Control 2	-
13	ADC12MCTL3	ADC12_A Memory Control 3	-
...	...	...	-
1D	ADC12MCTL13	ADC12_A Memory Control 13	-
1E	ADC12MCTL14	ADC12_A Memory Control 14	-
1F	ADC12MCTL15	ADC12_A Memory Control 15	-

MSP430 – ADC12_A - Registradores

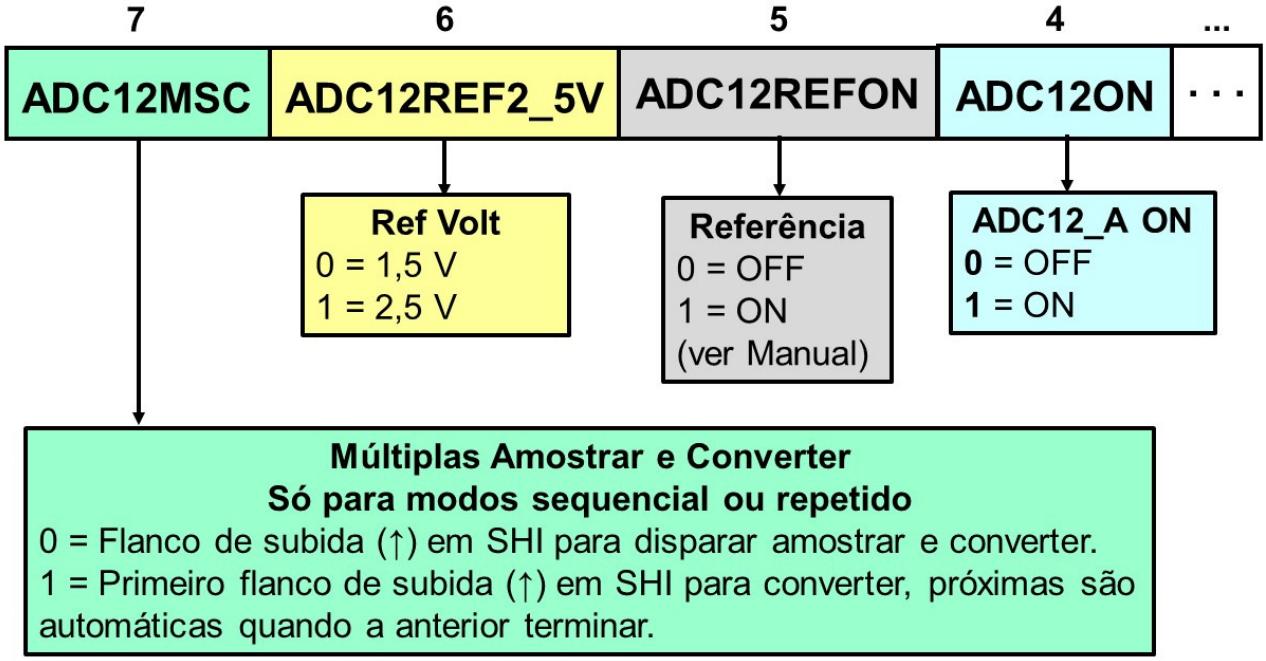
ADC12CTL0_H → Controle 0 (High) ADC12_A



Quantidade de ciclos ADC12CLK			
0 = 4 ciclos	4 = 64 ciclos	8 = 256 ciclos	12 = 1024 ciclos
1 = 8 ciclos	5 = 96 ciclos	9 = 384 ciclos	13 = 1024 ciclos
2 = 16 ciclos	6 = 128 ciclos	10 = 512 ciclos	14 = 1024 ciclos
3 = 32 ciclos	7 = 192 ciclos	11 = 768 ciclos	15 = 1024 ciclos

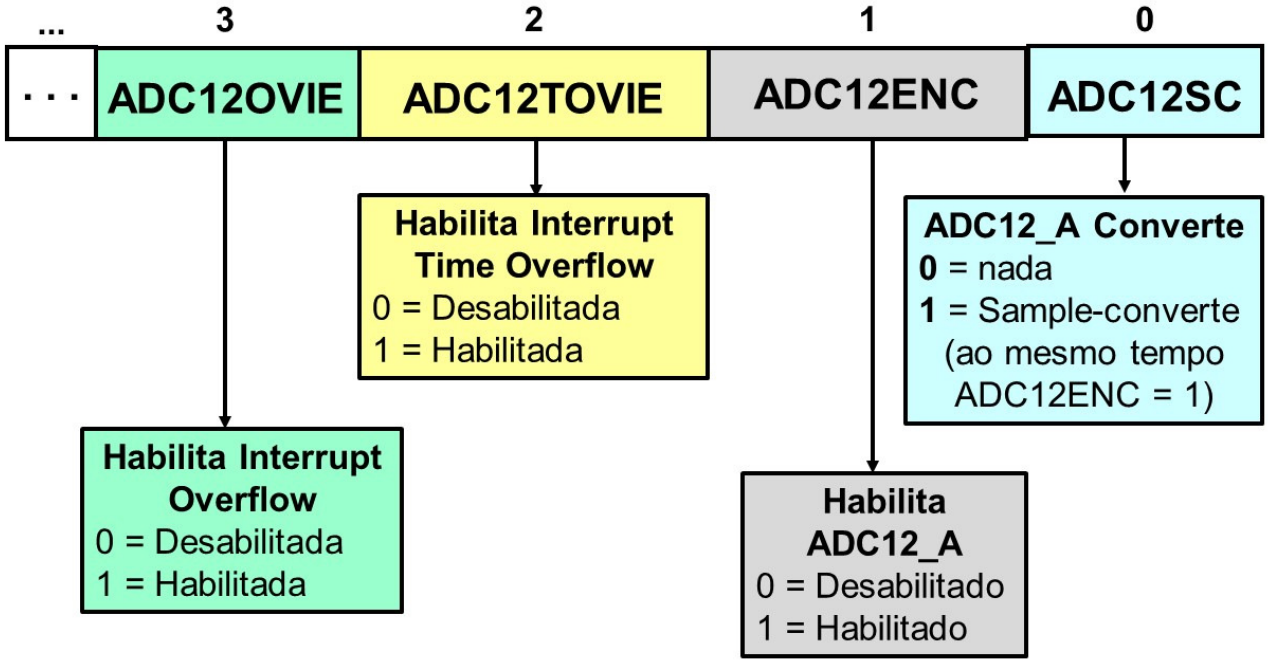
MSP430 – ADC12_A - Registradores

ADC12CTL0_L → Controle 0 (Low) ADC12_A



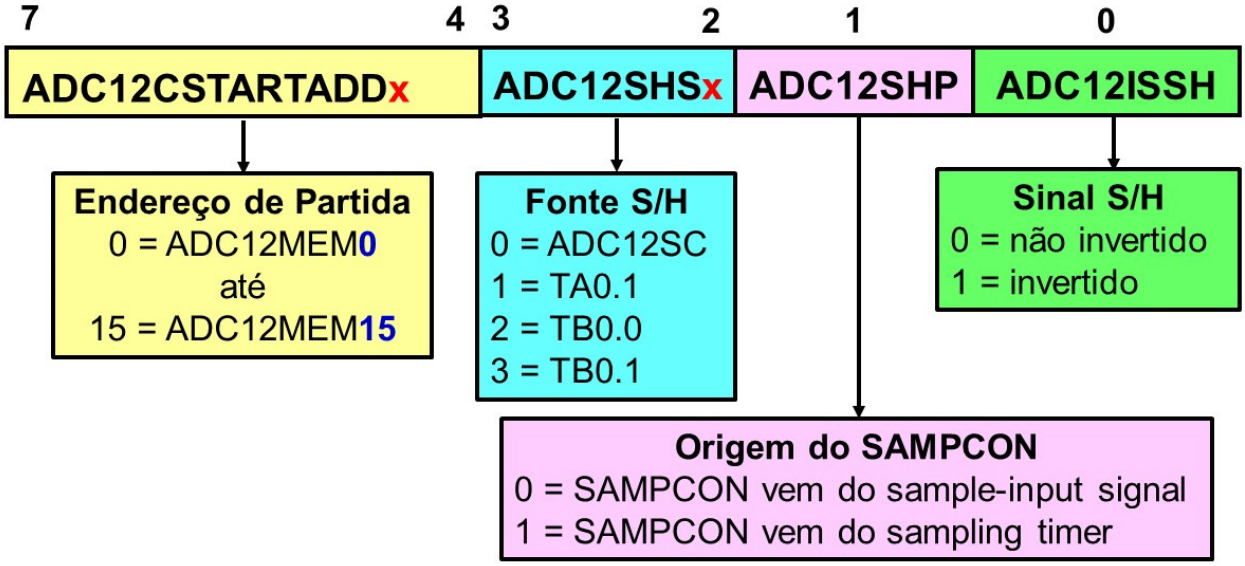
MSP430 – ADC12_A - Registradores

ADC12CTL0_L → Controle 0 (Low) ADC12_A



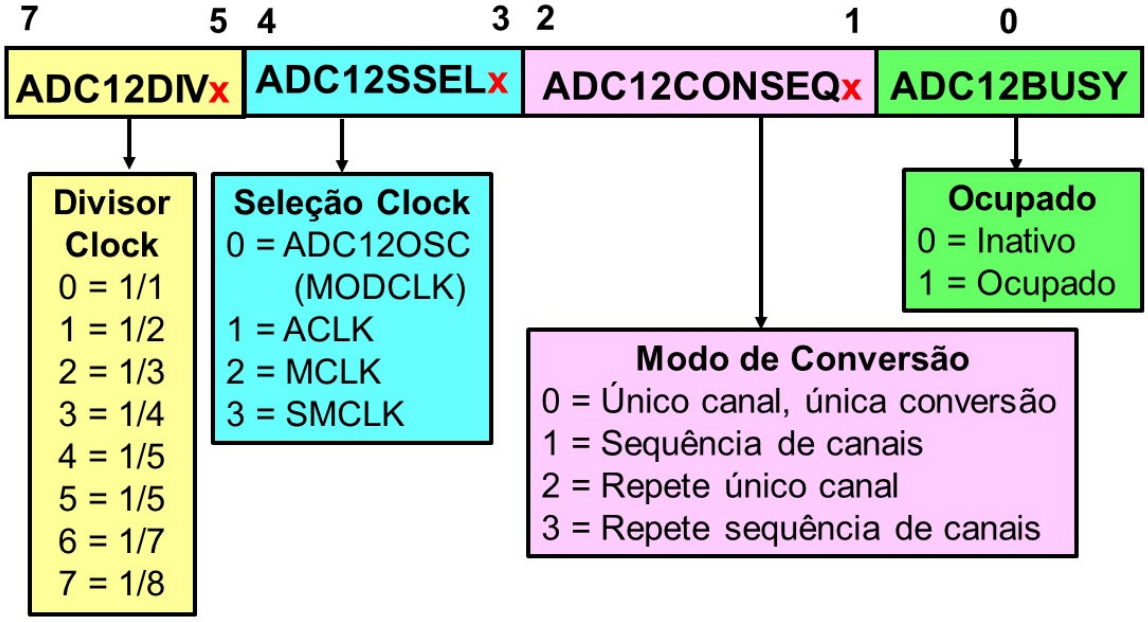
MSP430 – ADC12_A - Registradores

ADC12CTL1_H → Controle 1 (High) ADC12_A

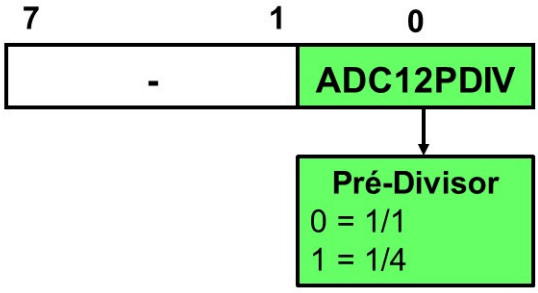


MSP430 – ADC12_A - Registradores

ADC12CTL1_L → Controle 1 (Low) ADC12_A

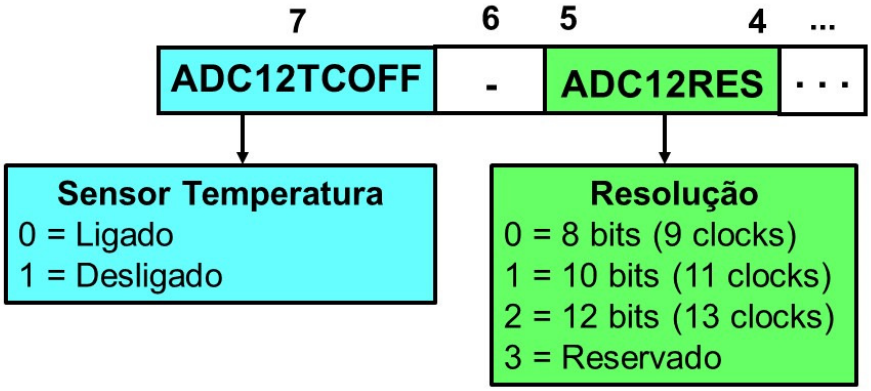


ADC12CTL2_H → Controle 2 (High) ADC12_A



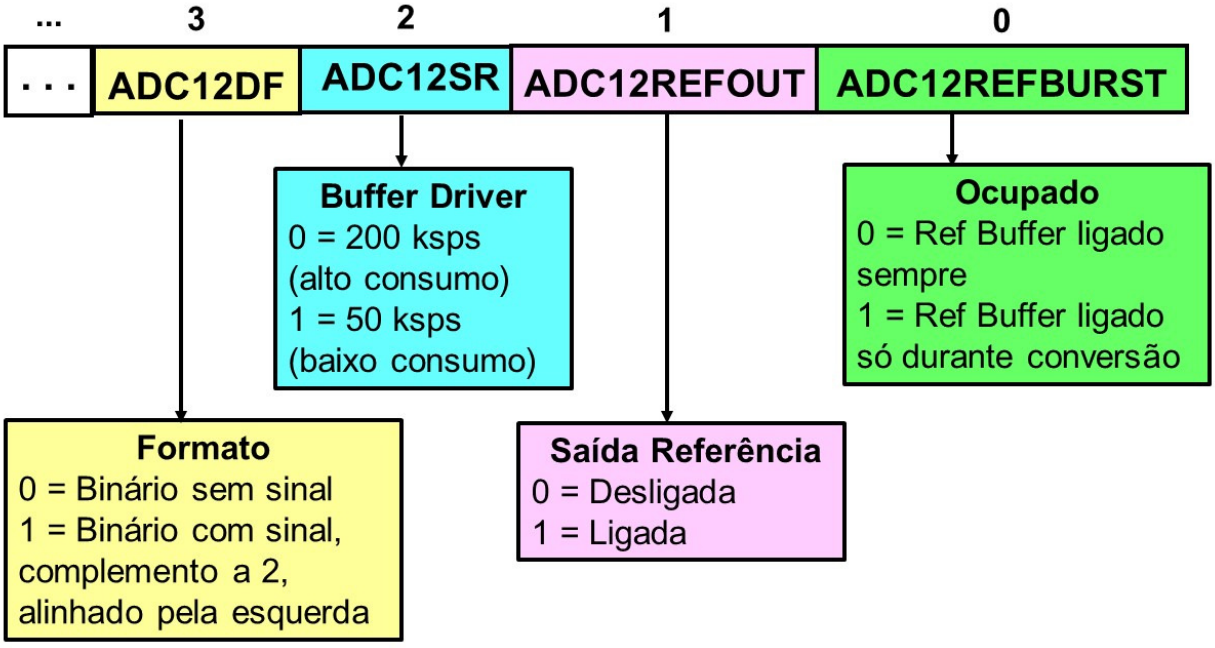
MSP430 – ADC12_A - Registradores

ADC12CTL2_L → Controle 2 (Low) ADC12_A



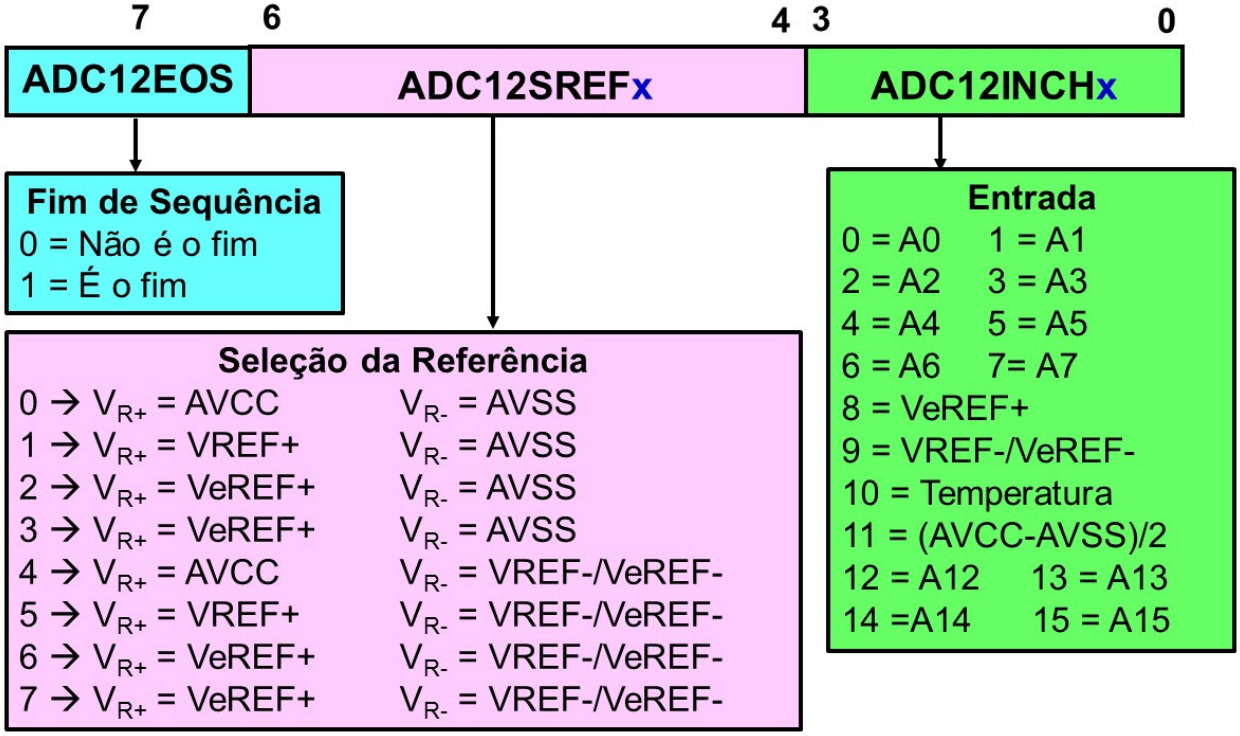
MSP430 – ADC12_A - Registradores

ADC12CTL2_L → Controle 2 (Low) ADC12_A



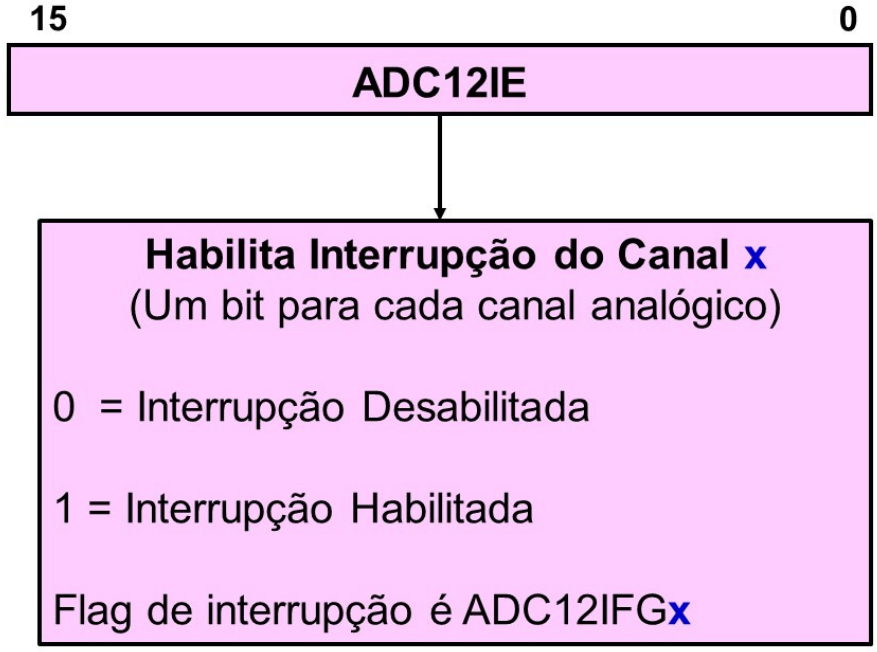
MSP430 – ADC12_A - Registradores

ADC12MCTLx → Controle x Reg de Memória



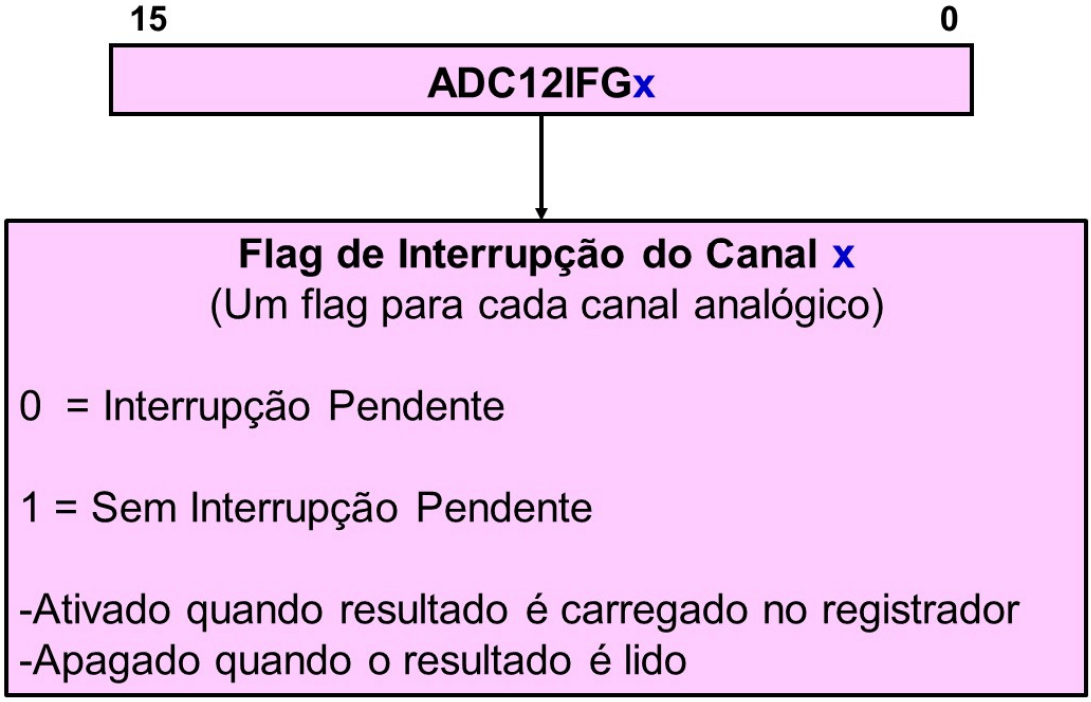
MSP430 – ADC12_A - Registradores

ADC12IE → Habilita Interrupções



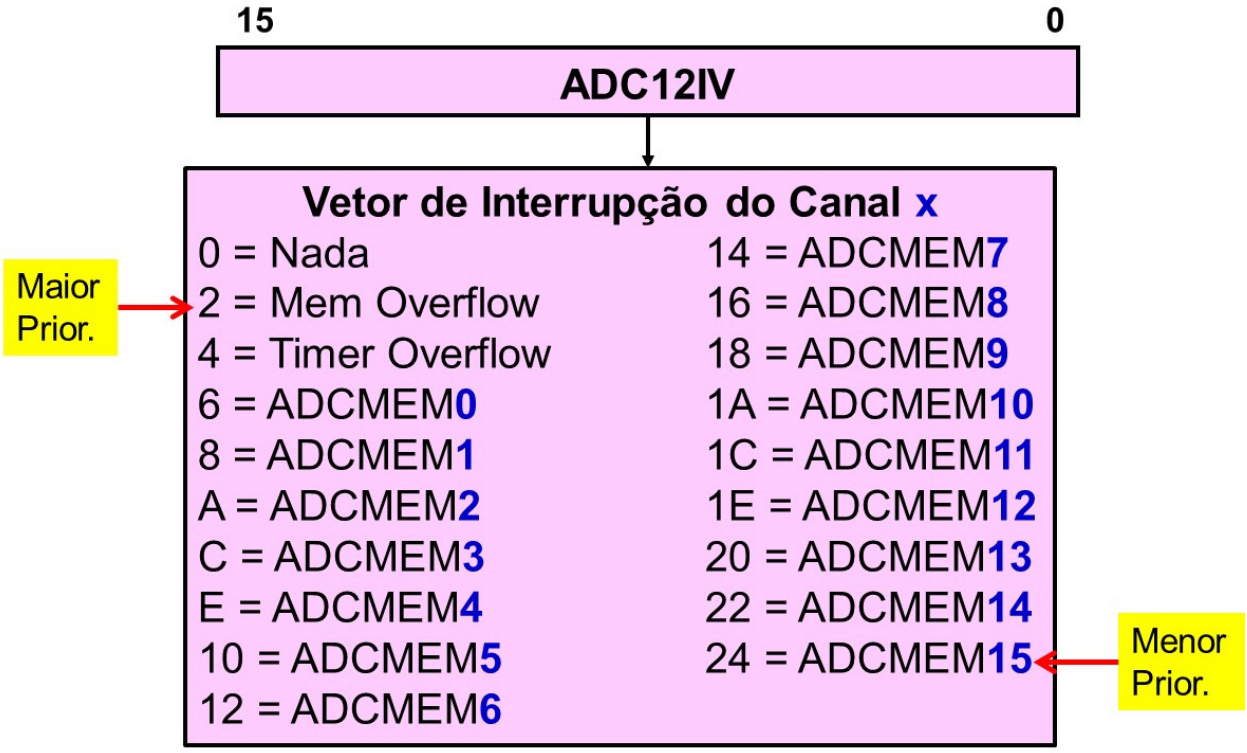
MSP430 – ADC12_A - Registradores

ADC12IFG → Flags de Interrupção



MSP430 – ADC12_A - Registradores

ADC12IFG → Vetores de Interrupção



MSP430 – DMA - Características

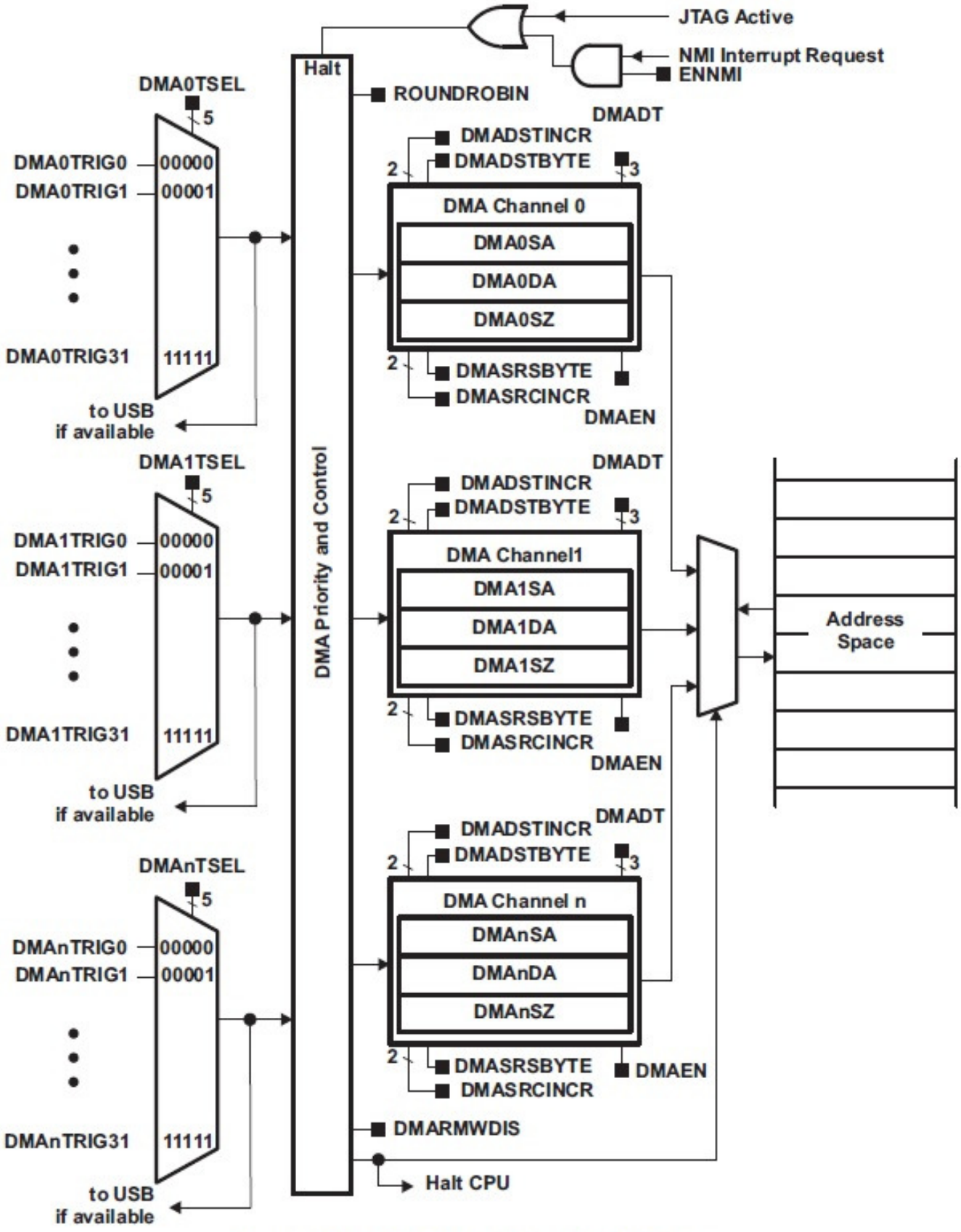


Figure 11-1. DMA Controller Block Diagram

MSP430 – DMA – Endereçamento

São 4 modos de endereçamento

- É possível programar o Modo de Endereçamento de cada canal de forma independente dos demais.

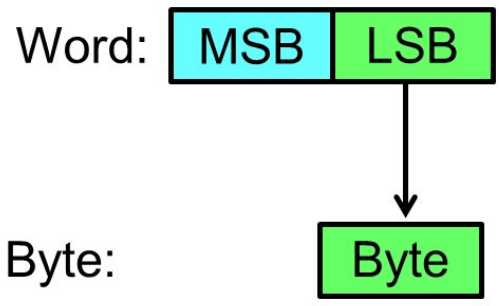
	Source	Destination
• Fixo para Fixo	Fixo	Fixo
• Fixo para Bloco	Fixo	Incr ou Decr
• Bloco para Fixo	Incr ou Decr	Fixo
• Bloco para Bloco	Incr ou Decr	Incr ou Decr

DMA**SRC**INCR
DMA**DST**INCR

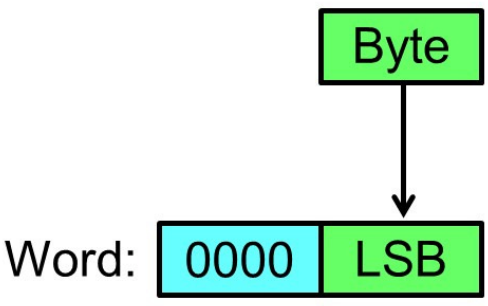
00 ou 01 → Fixo
10 → Decremento
11 → Incremento

MSP430 – DMA – Transferências

- Transferência de **word → byte**: só o LSB



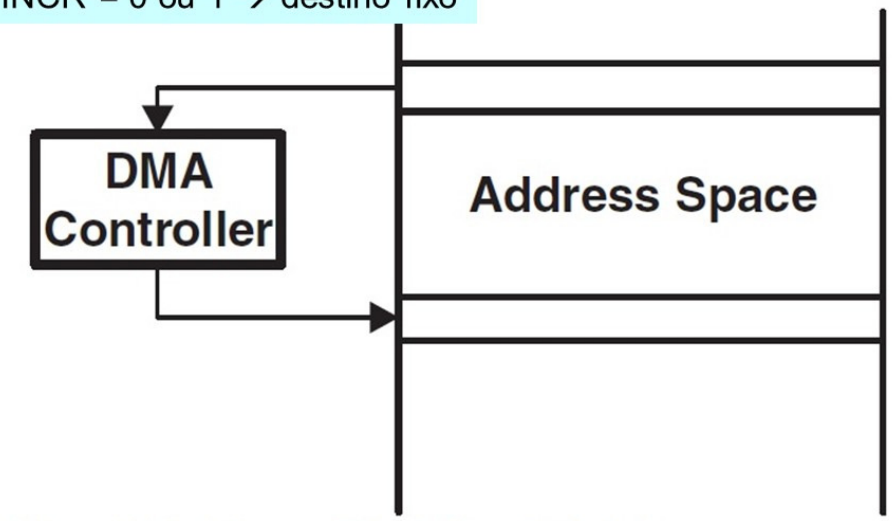
- Transferência de **byte → word**: só o LSB, MSB é zerado.



MSP430 – DMA – Endereçamento

- Fixo para Fixo

DMA**SRC**INCR = 0 ou 1 → fonte fixo
DMA**DST**INCR = 0 ou 1 → destino fixo

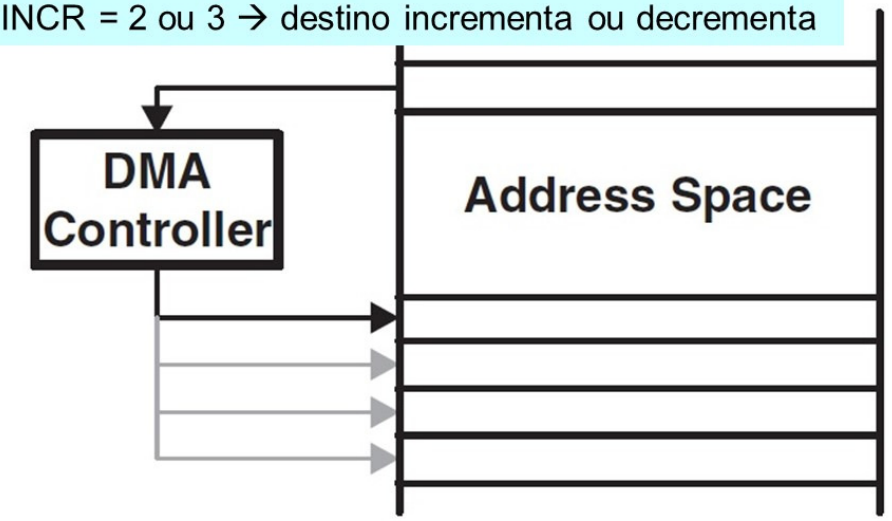


Fixed Address To Fixed Address

MSP430 – DMA – Endereçamento

- Fixo para Bloco

DMA**SRC**INCR = 0 ou 1 → fonte fixo
DMA**DST**INCR = 2 ou 3 → destino incrementa ou decrementa



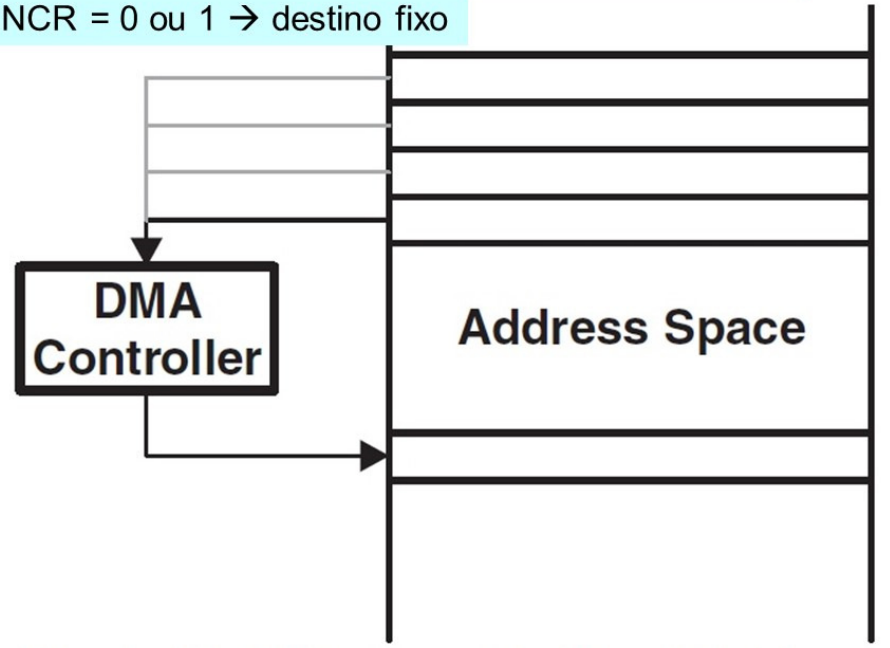
Fixed Address To Block Of Addresses

MSP430 – DMA – Endereçamento

- Bloco para Fixo

DMA**SRC**INCR = 2 ou 3 → fonte incrementa ou decrementa

DMA**DST**INCR = 0 ou 1 → destino fixo



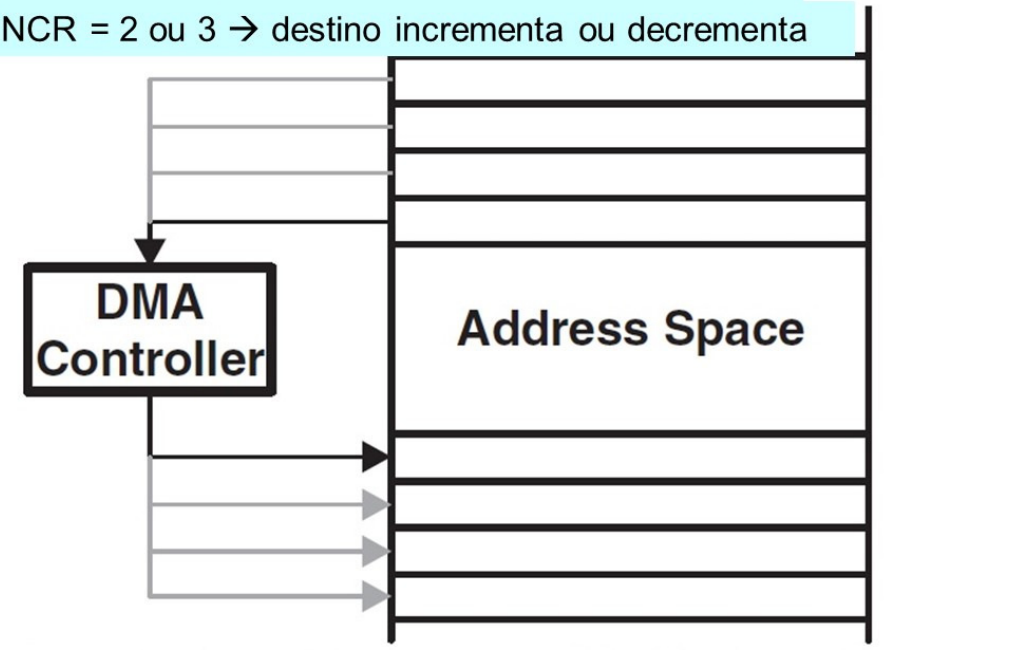
Block Of Addresses To Fixed Address

MSP430 – DMA – Endereçamento

- Bloco para Bloco

DMA**SRC**INCR = 2 ou 3 → fonte incrementa ou decrementa

DMA**DST**INCR = 2 ou 3 → destino incrementa ou decrementa



Block Of Addresses To Block Of Addresses

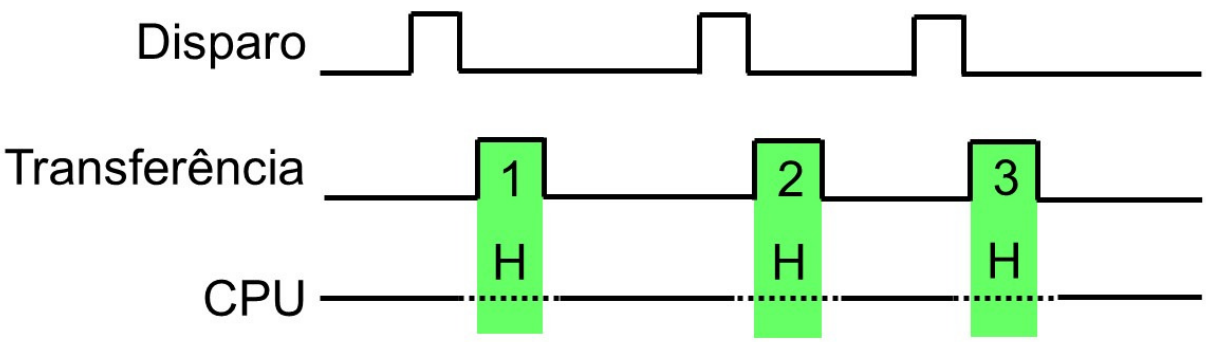
MSP430 – DMA - Modos

DMADT	Modo	Descrição
000	Simples	Um disparo para cada transferência. Se DMAxSZ = 0 → DMAEN = 0.
001	Bloco	Um disparo para transferir todo o bloco. Ao final do bloco → DMAEN = 0.
010 011	Rajada	Um disparo para transferir todo o bloco. CPU intervalada com as transferências. Ao final do bloco → DMAEN = 0.
100	Simples repetido	Um disparo para cada transferência. DMAEN permanece habilitado.
101	Bloco repetido	Um disparo para transferir todo o bloco. DMAEN permanece habilitado.
110 111	Rajada repetida	Um disparo para transferir todo o bloco. CPU intervalada com as transferências. DMAEN permanece habilitado.

DMAxSZ = quantidade de transferências programada.

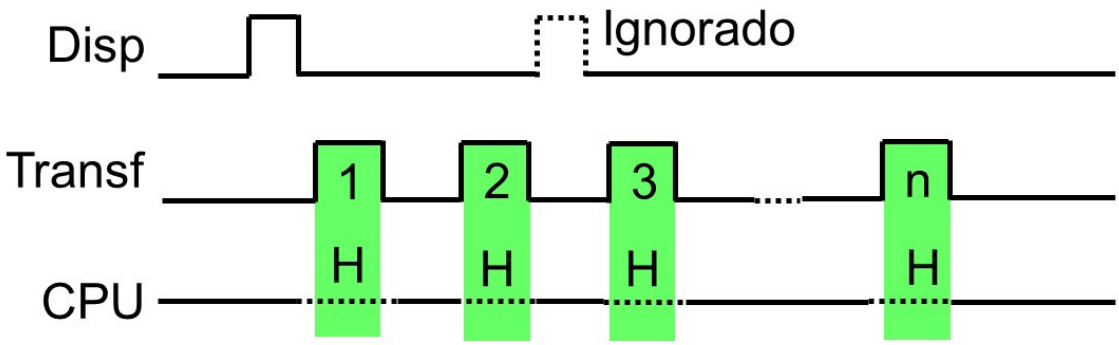
MSP430 – DMA - Simples

- Precisa de um disparo para cada transferência.
- CPU entra em Hold por um curto período.



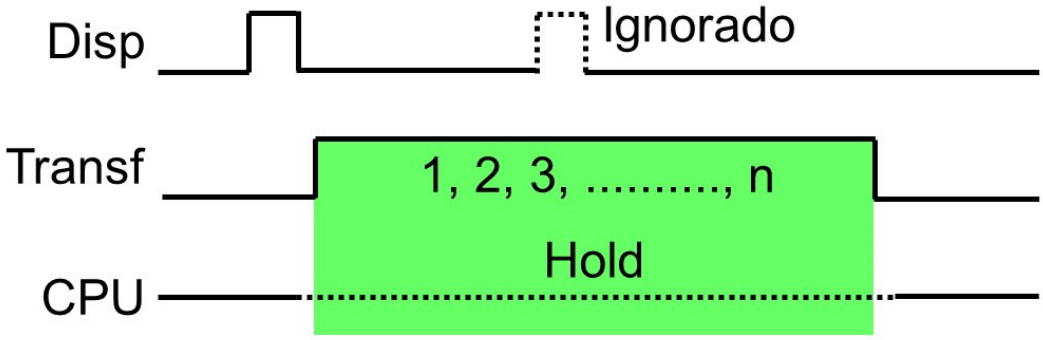
MSP430 – DMA - Rajada

- O primeiro disparo inicia a transferência.
- Só pára quando contador chegar a zero.
- CPU intercalada com as transferências.



MSP430 – DMA - Bloco

- O primeiro disparo inicia a transferência.
- Só pára quando contador chegar a zero.



MSP430 – DMA - Simples

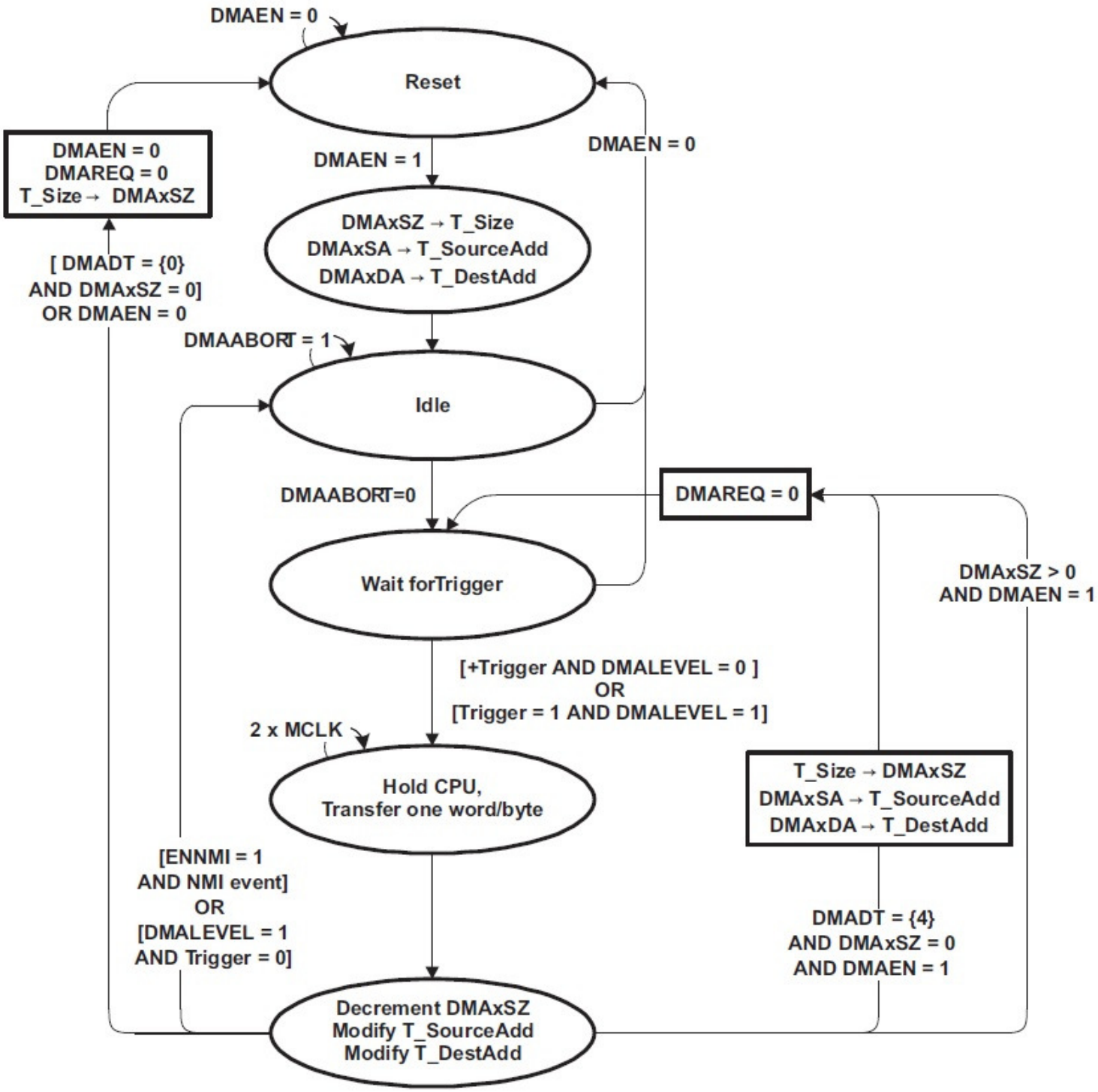


Figure 11-3. DMA Single Transfer State Diagram

MSP430 – DMA - Bloco

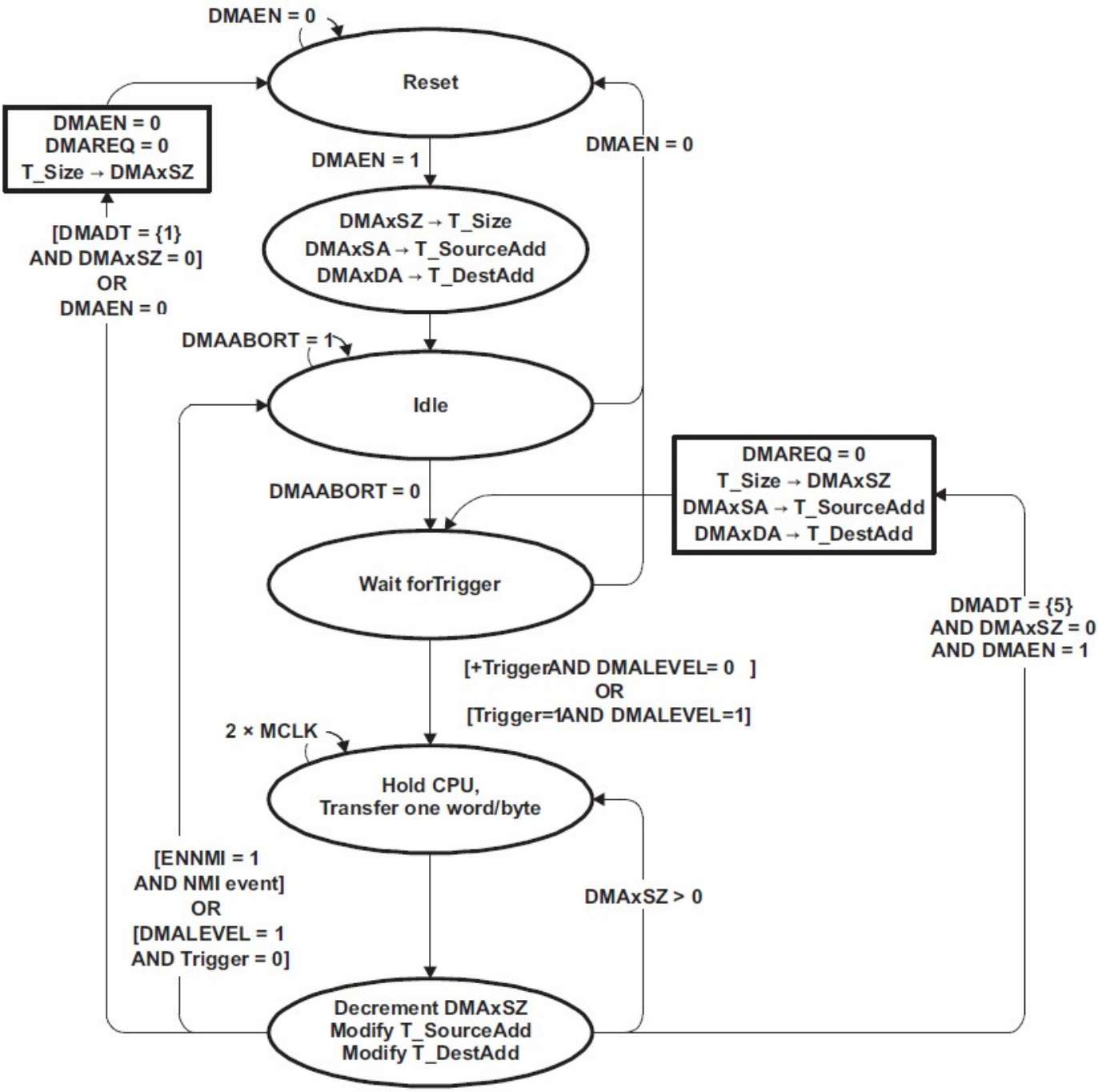


Figure 11-4. DMA Block Transfer State Diagram

MSP430 – DMA - Rajada

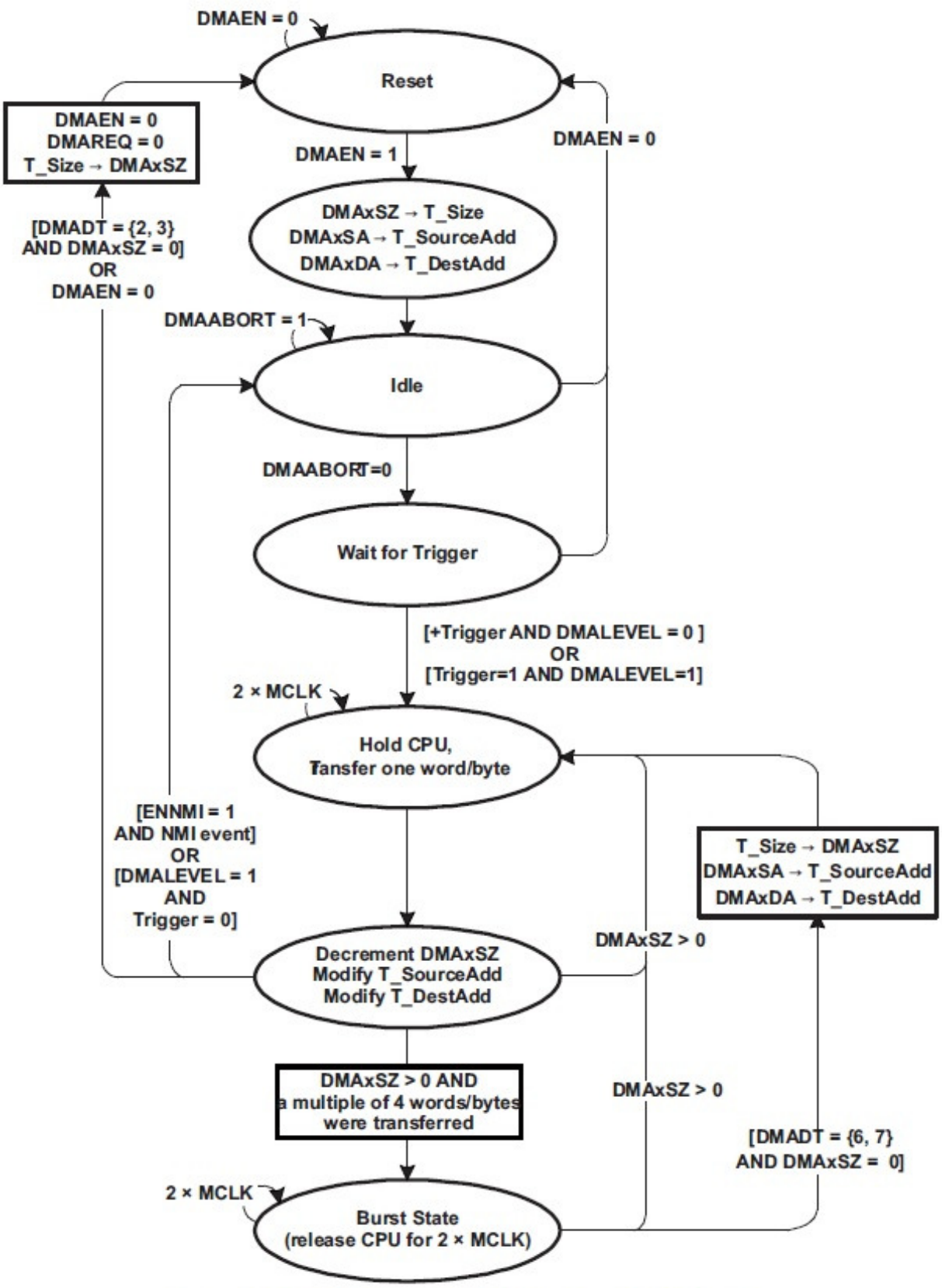


Figure 11-5. DMA Burst-Block Transfer State Diagram

MSP430 – DMA – Resumo Transf.

DMAx**SA** → inc/dec → fonte

DMAx**DA** → inc/dec → destino

DMAx**SZ** → dec → contador (se =0 nada acontece)

DMAx**EN** → habilita

DMA**IFG** → interrupção

	Não Repetido (DMAxEN=0)	Repetido (DMAxEN=1)
Simples	DMAT = 000 DMAxSZ = 0 → DMAxEN = 0 → DMAIFG = 1 1 Disparo → 1 Transferência	DMAT = 100 DMAxSZ = 0 → DMAIFG = 1 1 Disparo → 1 Transferência
Bloco	DMAT = 001 DMAxSZ = 0 → DMAxEN = 0 → DMAIFG = 1 1 Disparo → 1 Bloco	DMAT = 101 DMAxSZ = 0 → DMAIFG = 1 1 Disparo → 1 Bloco
Rajada	DMAT = 010 ou 011 DMAxSZ = 0 → DMAxEN = 0 → DMAIFG = 1 1 Disparo → 1 Rajada	DMAT = 110 ou 111 DMAxSZ = 0 → DMAIFG = 1 Recomeça automaticamente

MSP430 – DMA

- Ao iniciar uma transferência (DMAEN = 1):

- DMA**x**SA,
- DMA**x**DA e
- DMA**x**SZ

→ são copiados para posições temporárias. →

T_DMA**x**SA,
T_DMA**x**DA e
T_DMA**x**SZ

A cada transferência são alteradas: →

T_DMA**x**SA (inc ou dec),
T_DMA**x**DA (inc ou dec),
DMA**x**SZ (dec),

MSP430 – DMA – Fonte de Disparos

DMACTL**x**

Trigger	Canal		
	0	1	2
0	DMAREQ	DMAREQ	DMAREQ
1	TA0CCR0 CCIFG	TA0CCR0 CCIFG	TA0CCR0 CCIFG
2	TA0CCR2 CCIFG	TA0CCR2 CCIFG	TA0CCR2 CCIFG
3	TA1CCR0 CCIFG	TA1CCR0 CCIFG	TA1CCR0 CCIFG
4	TA1CCR2 CCIFG	TA1CCR2 CCIFG	TA1CCR2 CCIFG
5	TA2CCR0 CCIFG	TA2CCR0 CCIFG	TA2CCR0 CCIFG
6	TA2CCR2 CCIFG	TA2CCR2 CCIFG	TA2CCR2 CCIFG
7	TB0CCR0 CCIFG	TB0CCR0 CCIFG	TB0CCR0 CCIFG
8	TB0CCR2 CCIFG	TB0CCR2 CCIFG	TB0CCR2 CCIFG

MSP430 – DMA – Fonte de Disparos

DMACTL**x**

Trigger	Canal		
	0	1	2
9	Reservado	Reservado	Reservado
10	Reservado	Reservado	Reservado
11	Reservado	Reservado	Reservado
12	Reservado	Reservado	Reservado
13	Reservado	Reservado	Reservado
14	Reservado	Reservado	Reservado
15	Reservado	Reservado	Reservado

MSP430 – DMA – Fonte de Disparos

DMACTLx

Trigger	Canal		
	0	1	2
16	UCA0RXIFG	UCA0RXIFG	UCA0RXIFG
17	UCA0TXIFG	UCA0TXIFG	UCA0TXIFG
18	UCB0RXIFG	UCB0RXIFG	UCB0RXIFG
19	UCB0TXIFG	UCB0TXIFG	UCB0TXIFG
20	UCA1RXIFG	UCA1RXIFG	UCA1RXIFG
21	UCA1TXIFG	UCA1TXIFG	UCA1TXIFG
22	UCB1RXIFG	UCB1RXIFG	UCB1RXIFG
23	UCB1TXIFG	UCB1TXIFG	UCB1TXIFG

MSP430 – DMA – Fonte de Disparos

DMACTLx

Trigger	Canal		
	0	1	2
24	ADC12IFGx	ADC12IFGx	ADC12IFGx
25	Reservado	Reservado	Reservado
26	Reservado	Reservado	Reservado
27	USB FNRXD	USB FNRXD	USB FNRXD
28	USB Ready	USB Ready	USB Ready
29	MPY Ready	MPY Ready	MPY Ready
30	DMA2IFG	DMA0IFG	DMA1IFG
31	DMAE0	DMAE0	DMAE0

MSP430 – DMA – Disparo pelo DMA

DMACTL**x** = 0 → DMA**x**CTL.DMAREQ (software)

- Disparo acontece com DMAREQ = 1.
- É zerado (DMAREQ = 0) quando inicia transferência.

DMACTL**x** = 30 → DMA**x**CTL.DMA**x**IFG

- Disparo acontece com DMA**x**IFG = 1.
- DMA**0**IFG → Canal 1.
- DMA**1**IFG → Canal 2.
- DMA**2**IFG → Canal 0.

- DMA**x**IFG não é zerado quando a transferência inicia.

DMACTL**x** = 31 → DMAE0

- Transferência inicia com o disparo externo.

MSP430 – DMA – Disparo pelo Timer

DMACTL**x** = 1 → T**A0**CCR**0**.CCIFG = 1

- Disparo acontece com CCIFG = 1.
- É zerado (CCIFG = 0) quando inicia transferência.
- Se CCIE = 1 (interrupção habilitada), não há disparo.

DMACTL**x** = 24 → ADC12IFG = 1

- Precisa de interrupção desabilitada ADC12IE = 0.
- Conversão simples → ADC12IFG é o trigger.
- Sequência → ADC12IFG (da última) é o trigger.
- Por software, não se dispara fazendo ADC12IFG = 1.
- ADC12IFG são zerados quando o DMA acessa as posições de memória correspondentes.

MSP430 – DMA – Disparo pelo Serial

DMACTL**x** = 16, ..., 23 → USCI_**Ax** ou USCI_**Bx**

- Disparo acontece quando recebe um dado.
- Disparo com UC**Ax**RXIFG = 1 ou UC**Bx**RXIFG = 1.
- Zerado (UC**Xx**RXIFG=0) quando inicia transferência.
- Se UC**Xx**RXIE=1 (interrup. hab.), não há disparo.

- Disparo acontece quando transmite um dado.
- Disparo com UC**Ax**TXIFG = 1 ou UC**Bx**TXIFG = 1.
- Zerado (UC**Xx**TXIFG=0) quando inicia transferência.
- Se UC**Xx**TXIE=1 (interrup. hab.), não há disparo.

MSP430 – DMA – Tempo

Ciclo de tempo do DMA

- DMA usa MCLK.

- Controlador de DMA precisa de 1 ou 2 ciclos de MCLK para se sincronizar antes de iniciar uma transferência (simples, rajada ou bloco).
- Cada transferência (byte ou word) precisa de 2 ciclos de MCLK.
- Após a transferência é preciso de 1 ciclo de MCLK.

- Conta grosseira: precisa de 4 a 5 ciclos de MCLK.
- Transfere mesmo com a CPU desligada.

- Ver casos em que MCLK é desligado.

MSP430 – DMA – Registradores

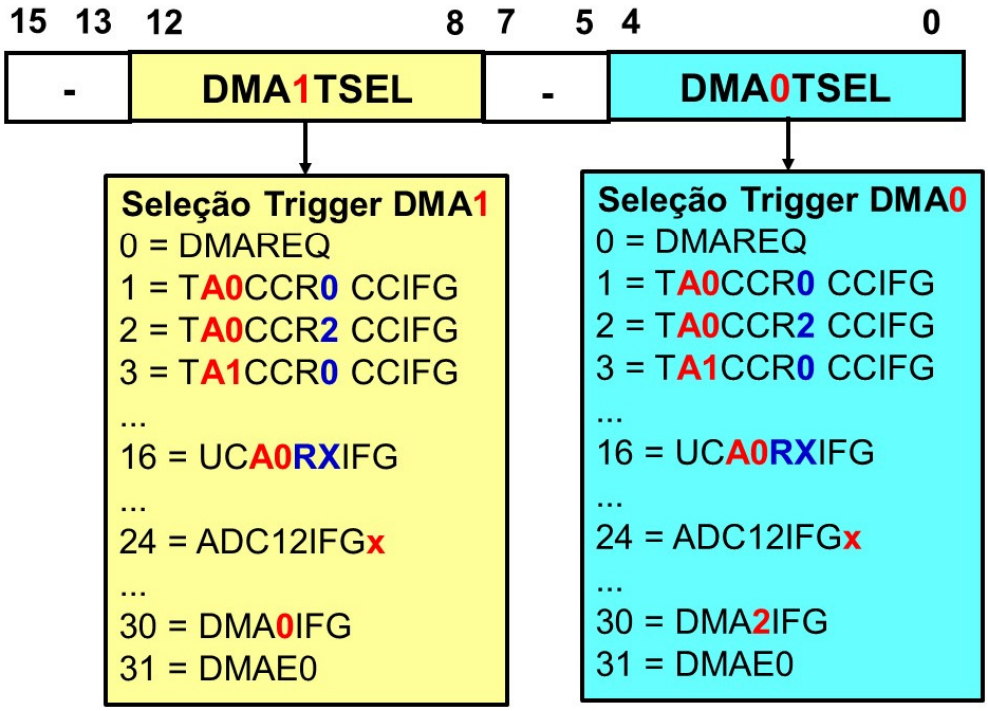
Offset	Acrônimo	Nome	Bits
0	DMACTL**0**	Controle 0	16
2	DMACTL**1**	Controle 1	16
4	DMACTL**2**	Controle 2	16
6	DMACTL**3**	Controle 3	16
8	DMACTL**4**	Controle 4	16
E	DMAIV	DMA Interrupt Vector	16

MSP430 – DMA – Registradores

Offset	Acrônimo	Nome	Bits
0	DMA**0**CTL	Controle Canal 0	16
2	DMA**0**SA	Endereço Fonte Canal 0	16/32
6	DMA**0**DA	Endereço Destino Canal 0	16/32
A	DMA**0**SZ	Quantidade de transferências	16
0	DMA**1**CTL	Controle Canal 1	16
2	DMA**1**SA	Endereço Fonte Canal 1	16/32
6	DMA**1**DA	Endereço Destino Canal 1	16/32
A	DMA**1**SZ	Quantidade de transferências	16
0	DMA**2**CTL	Controle Canal 2	16
2	DMA**2**SA	Endereço Fonte Canal 2	16/32
6	DMA**2**DA	Endereço Destino Canal 2	16/32
A	DMA**2**SZ	Quantidade de transferências	16

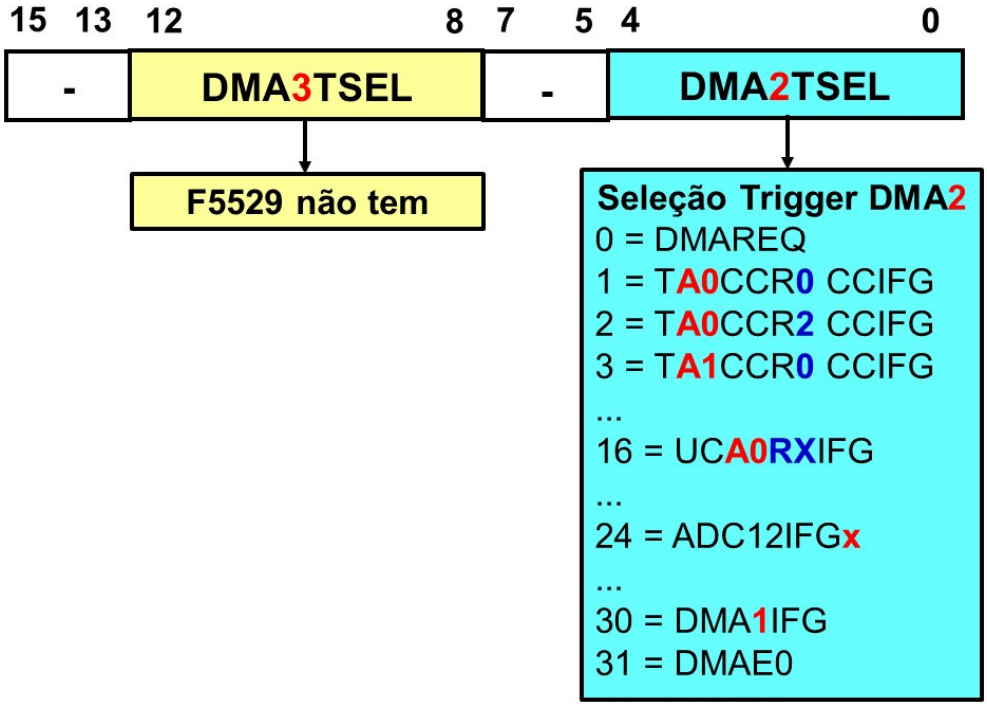
MSP430 – DMA – Registradores

DMACTL0 → Controle 0 do DMA



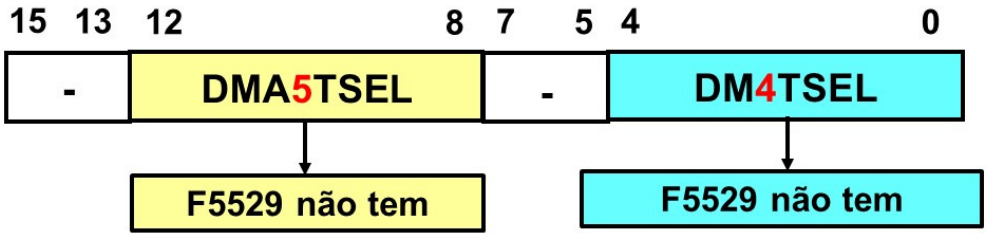
MSP430 – DMA – Registradores

DMACTL1 → Controle 1 do DMA

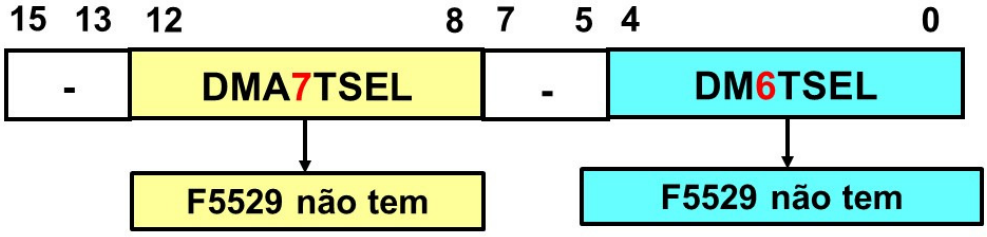


MSP430 – DMA – Registradores

DMACTL2 → Controle 2 do DMA

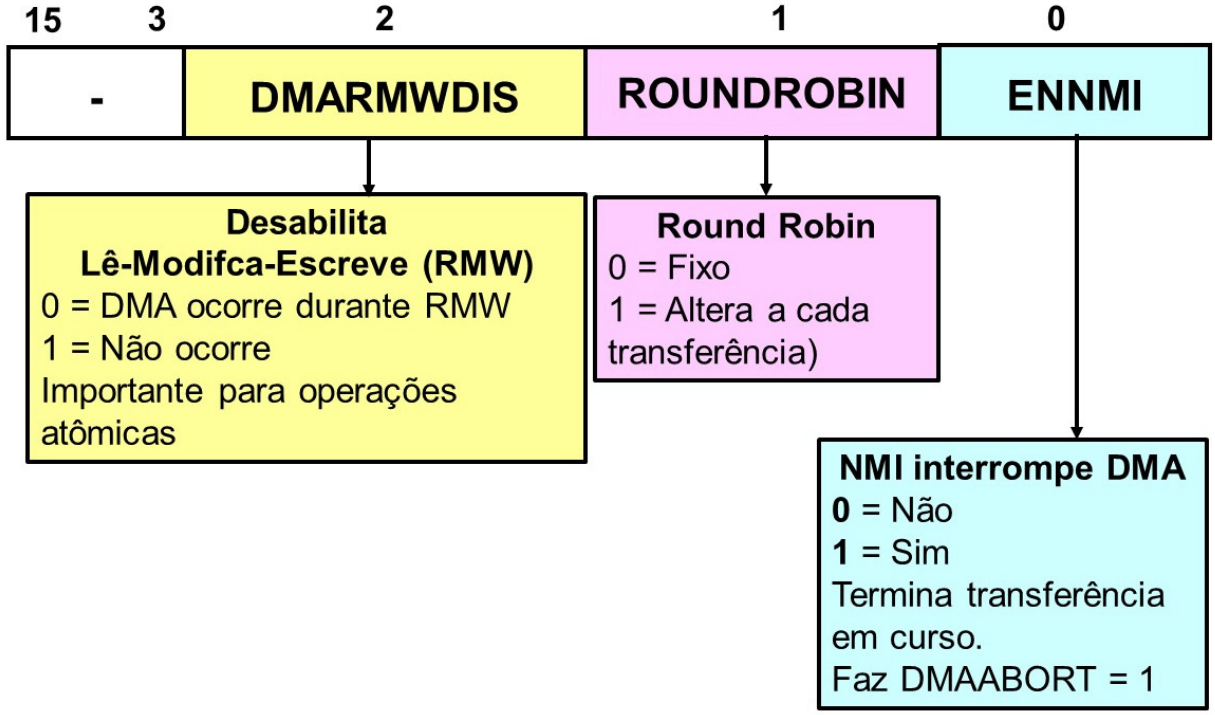


DMACTL3 → Controle 3 do DMA



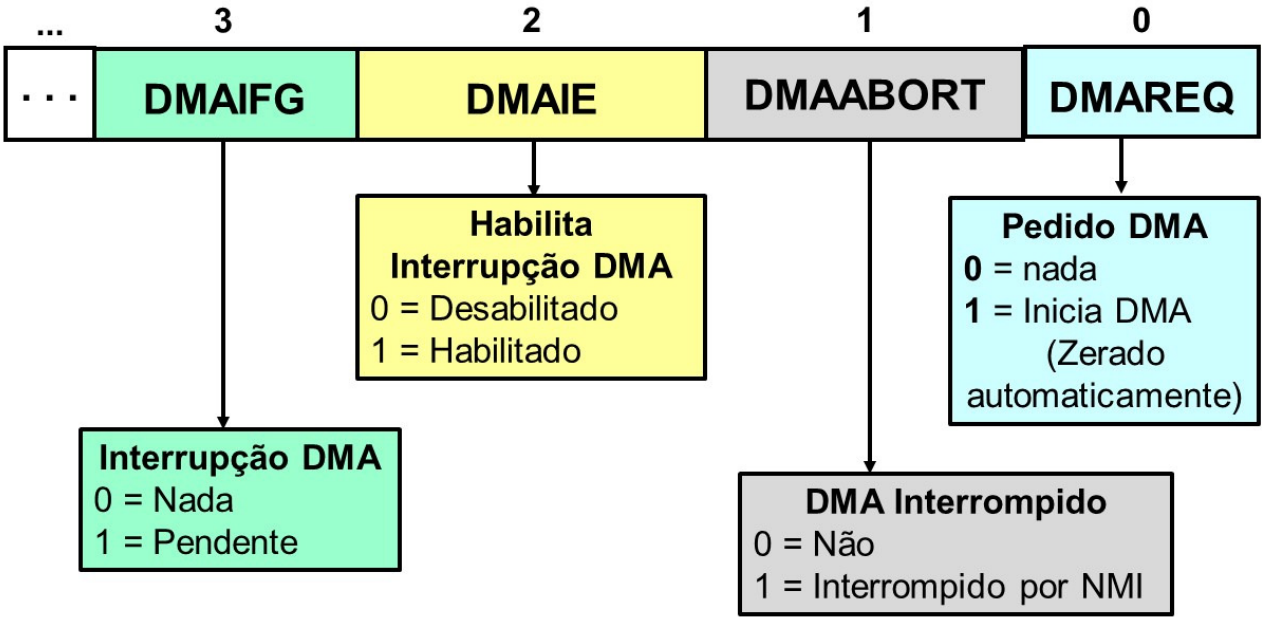
MSP430 – DMA – Registradores

DMACTL4 → Controle 4 do DMA



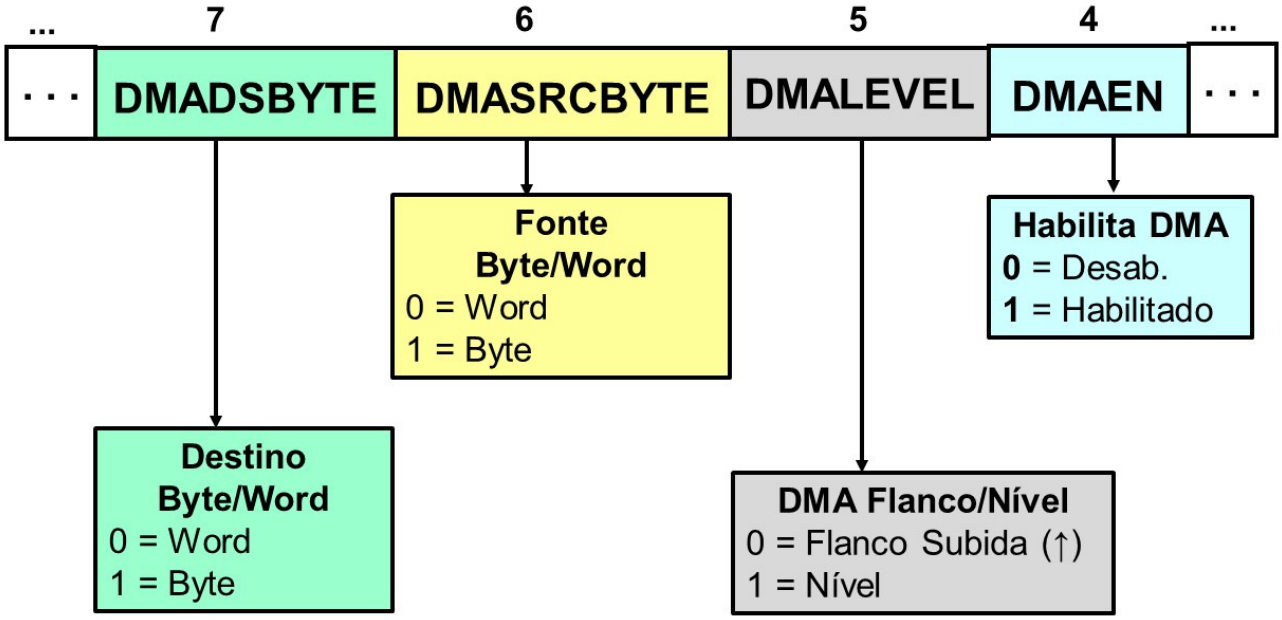
MSP430 – DMA – Registradores

DMAxCTL → Controle do Canal x (1 de 3)



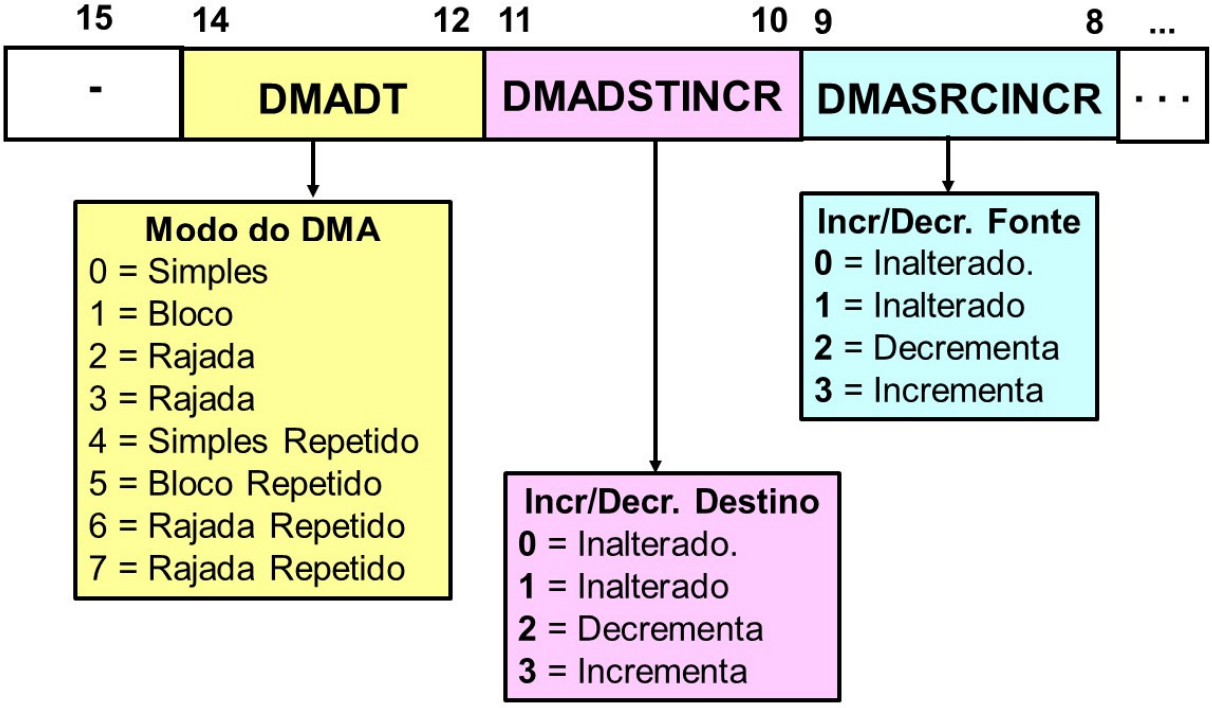
MSP430 – DMA – Registradores

DMAxCTL → Controle do Canal x (2 de 3)



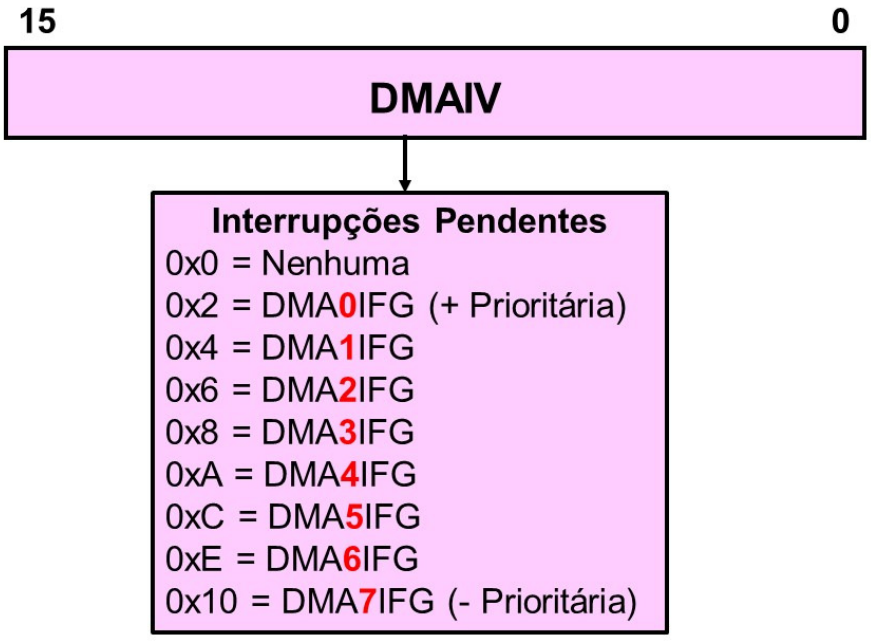
MSP430 – DMA – Registradores

DMAxCTL → Controle do Canal x (3 de 3)



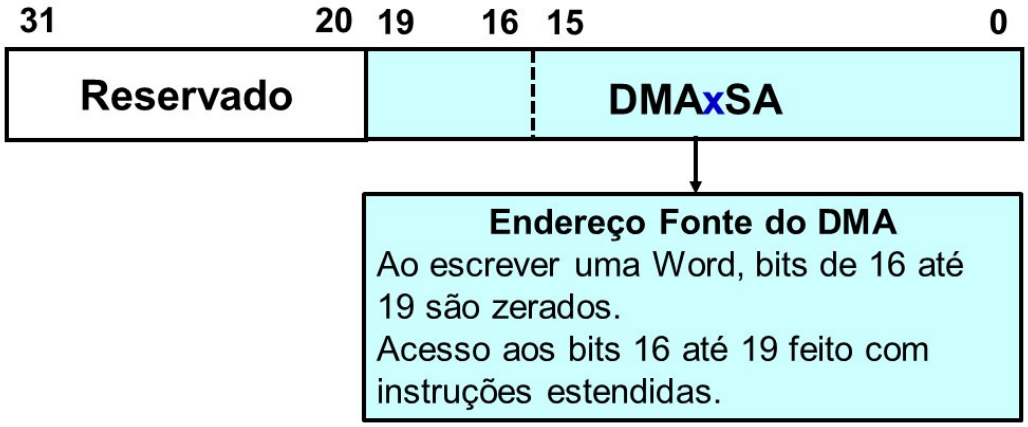
MSP430 – DMA – Registradores

DMAIV → Vetor de Interrupções



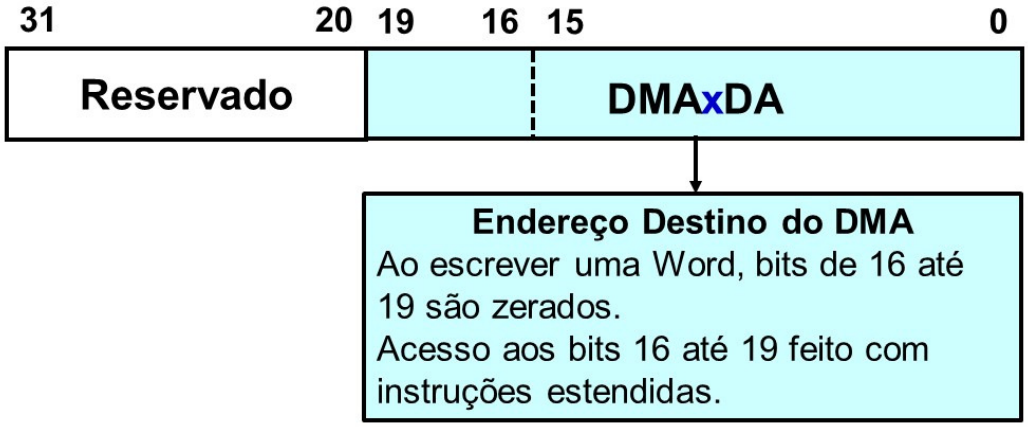
MSP430 – DMA – Registradores

DMAxSA → Endereço Fonte do Canal x



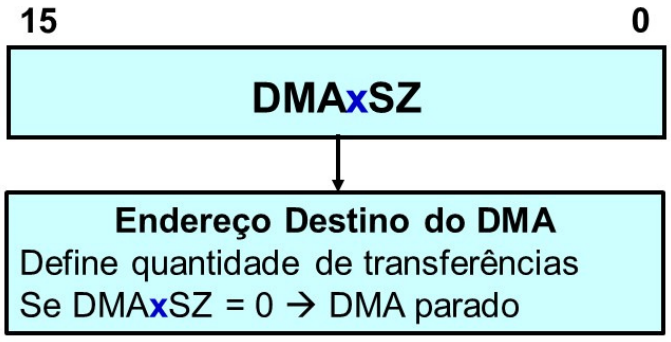
MSP430 – DMA – Registradores

DMAxDA → Endereço Destino do Canal x



MSP430 – DMA – Registradores

DMAxSZ → Quantidade de operações do Canal x



MSP430 – Periféricos

Table 18. Peripherals

MODULE NAME	BASE ADDRESS	OFFSET ADDRESS RANGE
Special Functions (see Table 19)	0100h	000h-01Fh
PMM (see Table 20)	0120h	000h-010h
Flash Control (see Table 21)	0140h	000h-00Fh
CRC16 (see Table 22)	0150h	000h-007h
RAM Control (see Table 23)	0158h	000h-001h
Watchdog (see Table 24)	015Ch	000h-001h
UCS (see Table 25)	0160h	000h-01Fh
SYS (see Table 26)	0180h	000h-01Fh
Shared Reference (see Table 27)	01B0h	000h-001h
Port Mapping Control (see Table 28)	01C0h	000h-002h
Port Mapping Port P4 (see Table 28)	01E0h	000h-007h
Port P1 and P2 (see Table 29)	0200h	000h-01Fh
Port P3 and P4 (see Table 30)	0220h	000h-00Bh
Port P5 and P6 (see Table 31)	0240h	000h-00Bh
Port P7 and P8 (see Table 32)	0260h	000h-00Bh
Port PJ (see Table 33)	0320h	000h-01Fh
TA0 (see Table 34)	0340h	000h-02Eh
TA1 (see Table 35)	0380h	000h-02Eh
TB0 (see Table 36)	03C0h	000h-02Eh
TA2 (see Table 37)	0400h	000h-02Eh
Real-Time Clock (RTC_A) (see Table 38)	04A0h	000h-01Bh
32-Bit Hardware Multiplier (see Table 39)	04C0h	000h-02Fh
DMA General Control (see Table 40)	0500h	000h-00Fh
DMA Channel 0 (see Table 40)	0510h	000h-00Ah
DMA Channel 1 (see Table 40)	0520h	000h-00Ah
DMA Channel 2 (see Table 40)	0530h	000h-00Ah
USCI_A0 (see Table 41)	05C0h	000h-01Fh
USCI_B0 (see Table 42)	05E0h	000h-01Fh
USCI_A1 (see Table 43)	0600h	000h-01Fh
USCI_B1 (see Table 44)	0620h	000h-01Fh
ADC12_A (see Table 45)	0700h	000h-03Eh
Comparator_B (see Table 46)	08C0h	000h-00Fh
USB Configuration (see Table 47)	0900h	000h-014h
USB Control (see Table 48)	0920h	000h-01Fh

Table 45. ADC12_A Registers (Base Address: 0700h)

REGISTER DESCRIPTION	REGISTER	OFFSET
Control register 0	ADC12CTL0	00h
Control register 1	ADC12CTL1	02h
Control register 2	ADC12CTL2	04h
Interrupt-flag register	ADC12IFG	0Ah
Interrupt-enable register	ADC12IE	0Ch
Interrupt-vector-word register	ADC12IV	0Eh
ADC memory-control register 0	ADC12MCTL0	10h
ADC memory-control register 1	ADC12MCTL1	11h
ADC memory-control register 2	ADC12MCTL2	12h
ADC memory-control register 3	ADC12MCTL3	13h
ADC memory-control register 4	ADC12MCTL4	14h
ADC memory-control register 5	ADC12MCTL5	15h
ADC memory-control register 6	ADC12MCTL6	16h
ADC memory-control register 7	ADC12MCTL7	17h
ADC memory-control register 8	ADC12MCTL8	18h
ADC memory-control register 9	ADC12MCTL9	19h
ADC memory-control register 10	ADC12MCTL10	1Ah
ADC memory-control register 11	ADC12MCTL11	1Bh
ADC memory-control register 12	ADC12MCTL12	1Ch
ADC memory-control register 13	ADC12MCTL13	1Dh
ADC memory-control register 14	ADC12MCTL14	1Eh
ADC memory-control register 15	ADC12MCTL15	1Fh
Conversion memory 0	ADC12MEM0	20h
Conversion memory 1	ADC12MEM1	22h
Conversion memory 2	ADC12MEM2	24h
Conversion memory 3	ADC12MEM3	26h
Conversion memory 4	ADC12MEM4	28h
Conversion memory 5	ADC12MEM5	2Ah
Conversion memory 6	ADC12MEM6	2Ch
Conversion memory 7	ADC12MEM7	2Eh
Conversion memory 8	ADC12MEM8	30h
Conversion memory 9	ADC12MEM9	32h
Conversion memory 10	ADC12MEM10	34h
Conversion memory 11	ADC12MEM11	36h
Conversion memory 12	ADC12MEM12	38h
Conversion memory 13	ADC12MEM13	3Ah
Conversion memory 14	ADC12MEM14	3Ch
Conversion memory 15	ADC12MEM15	3Eh

MSP430 – Periféricos – USCI_A0

Table 41. USCI_A0 Registers (Base Address: 05C0h)

REGISTER DESCRIPTION	REGISTER	OFFSET
USCI control 1	UCA0CTL1	00h
USCI control 0	UCA0CTL0	01h
USCI baud rate 0	UCA0BR0	06h
USCI baud rate 1	UCA0BR1	07h
USCI modulation control	UCA0MCTL	08h
USCI status	UCA0STAT	0Ah
USCI receive buffer	UCA0RXBUF	0Ch
USCI transmit buffer	UCA0TXBUF	0Eh
USCI LIN control	UCA0ABCTL	10h
USCI IrDA transmit control	UCA0IRTCTL	12h
USCI IrDA receive control	UCA0IRRCTL	13h
USCI interrupt enable	UCA0IE	1Ch
USCI interrupt flags	UCA0IFG	1Dh
USCI interrupt vector word	UCA0IV	1Eh

MSP430 – Periféricos – USCI_B0

Table 42. USCI_B0 Registers (Base Address: 05E0h)

REGISTER DESCRIPTION	REGISTER	OFFSET
USCI synchronous control 1	UCB0CTL1	00h
USCI synchronous control 0	UCB0CTL0	01h
USCI synchronous bit rate 0	UCB0BR0	06h
USCI synchronous bit rate 1	UCB0BR1	07h
USCI synchronous status	UCB0STAT	0Ah
USCI synchronous receive buffer	UCB0RXBUF	0Ch
USCI synchronous transmit buffer	UCB0TXBUF	0Eh
USCI I2C own address	UCB0I2COA	10h
USCI I2C slave address	UCB0I2CSA	12h
USCI interrupt enable	UCB0IE	1Ch
USCI interrupt flags	UCB0IFG	1Dh
USCI interrupt vector word	UCB0IV	1Eh

MSP430 – Periféricos – USCI_A1

Table 43. USCI_A1 Registers (Base Address: 0600h)

REGISTER DESCRIPTION	REGISTER	OFFSET
USCI control 1	UCA1CTL1	00h
USCI control 0	UCA1CTL0	01h
USCI baud rate 0	UCA1BR0	06h
USCI baud rate 1	UCA1BR1	07h
USCI modulation control	UCA1MCTL	08h
USCI status	UCA1STAT	0Ah
USCI receive buffer	UCA1RXBUF	0Ch
USCI transmit buffer	UCA1TXBUF	0Eh
USCI LIN control	UCA1ABCTL	10h
USCI IrDA transmit control	UCA1IRTCTL	12h
USCI IrDA receive control	UCA1IRRCTL	13h
USCI interrupt enable	UCA1IE	1Ch
USCI interrupt flags	UCA1IFG	1Dh
USCI interrupt vector word	UCA1IV	1Eh

MSP430 – Periféricos – USCI_B1

Table 44. USCI_B1 Registers (Base Address: 0620h)

REGISTER DESCRIPTION	REGISTER	OFFSET
USCI synchronous control 1	UCB1CTL1	00h
USCI synchronous control 0	UCB1CTL0	01h
USCI synchronous bit rate 0	UCB1BR0	06h
USCI synchronous bit rate 1	UCB1BR1	07h
USCI synchronous status	UCB1STAT	0Ah
USCI synchronous receive buffer	UCB1RXBUF	0Ch
USCI synchronous transmit buffer	UCB1TXBUF	0Eh
USCI I2C own address	UCB1I2COA	10h
USCI I2C slave address	UCB1I2CSA	12h
USCI interrupt enable	UCB1IE	1Ch
USCI interrupt flags	UCB1IFG	1Dh
USCI interrupt vector word	UCB1IV	1Eh