



Universidade Tecnológica Federal do Paraná – UTFPR
Bacharelado em Ciência da Computação

BCC34C – Sistemas Microcontrolados

Frank Helbert Borsato

O ATmega328

- **Microcontrolador com arquitetura RISC avançada**
- **131 instruções**
 - Maior parte executada em 1 ou 2 ciclos de clock
 - Poucas em 3 ou 4 ciclos
- **32 registradores de trabalho de propósito geral (8 bits).**
 - Alguns trabalham em par para endereçamentos de 16 bits
- **Operação até 20 MIPS a 20 MHz**
 - Na placa Arduino o ATmega328 trabalha a 16 MHz
- **32 KBytes de memória de programa flash**
 - Acesso por blocos
- **1 KByte de memória EEPROM**
 - Acesso por byte
- **2 KBytes de memória SRAM**

O ATmega328

- **Possui os seguintes periféricos:**
 - 23 entradas e saídas (I/Os) programáveis
 - 2 Temporizadores/Contadores de 8 bits com Prescaler separado, com modo de comparação
 - 1 Temporizador/Contador de 16 bits com Prescaler separado, com modo de comparação e captura
 - Contador de tempo real (com um cristal externo de 32,768 KHz conta precisamente 1 s)
 - **Conversor Analógico-Digital (ADC) com resolução de 10 bits:**
 - 8 canais na versão encapsulamento TQFP
 - 6 canais na versão encapsulamento PDIP
 - **Interface serial para dois fios orientada a byte (TWI)**
 - Compatível com o protocolo I2C
 - **Interface serial USART**
 - **Interface serial SPI Master/Slave**

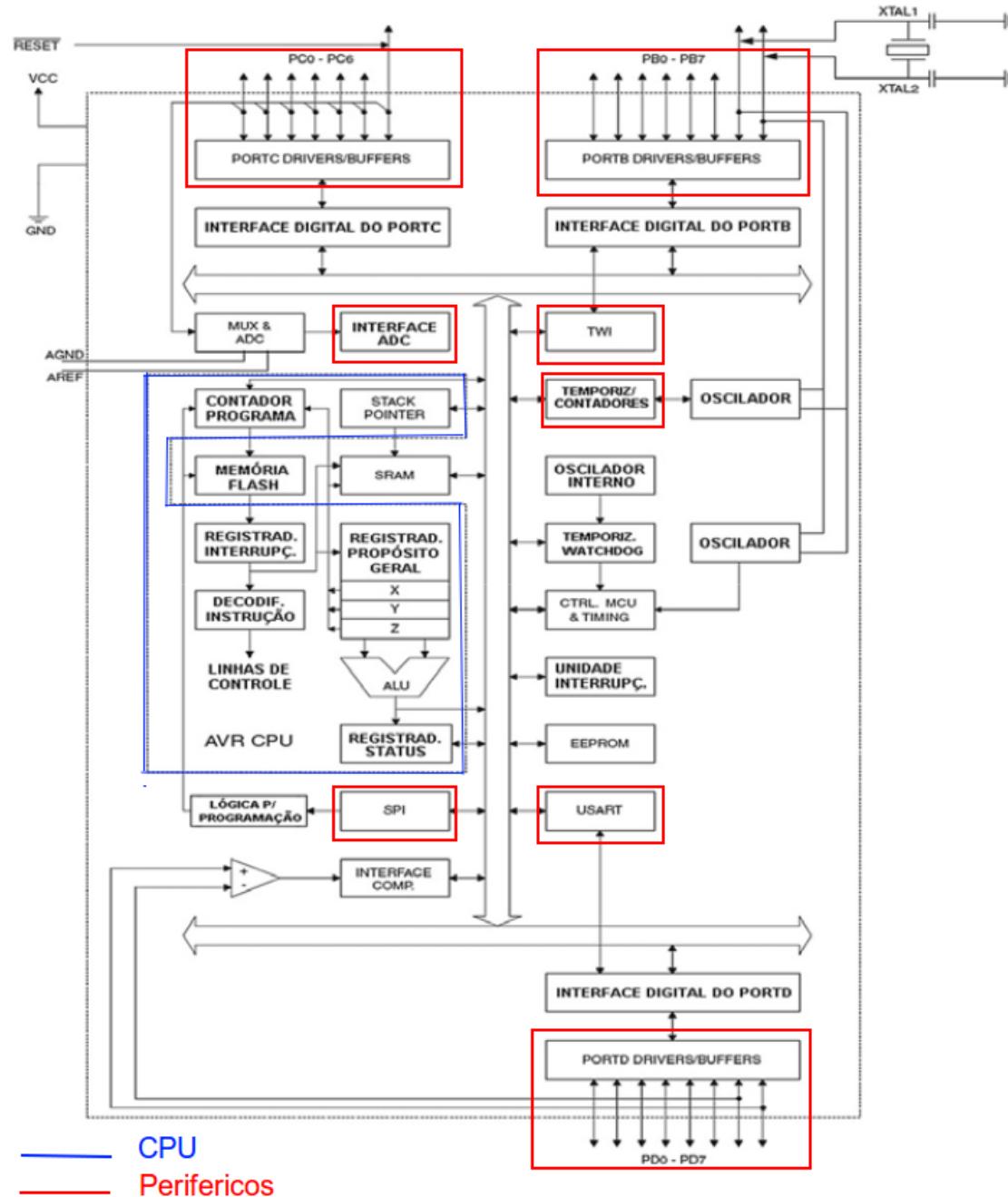


Fig. 2.3 – Diagrama em blocos do ATmega328.

O ATmega328

- **Temporizadores e contadores (T/C)**

- No trabalho com microcontroladores é importante a geração de sinais periódicos e eventos
- Prescaler é um divisor de frequência

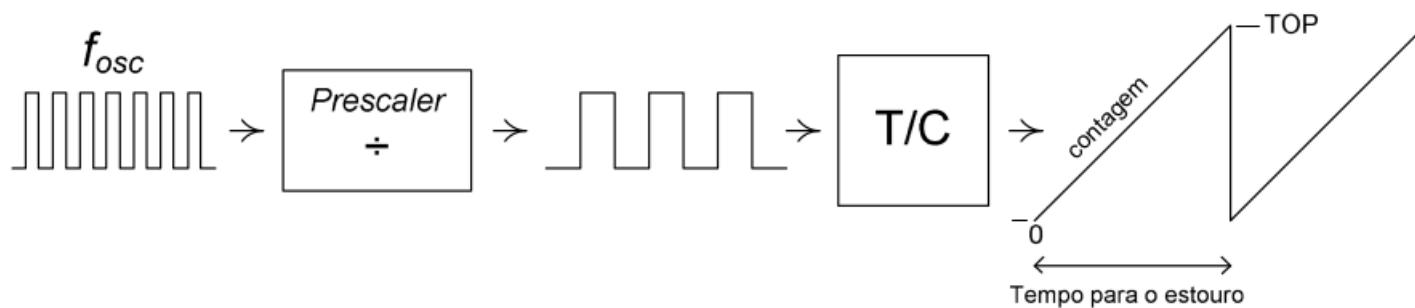
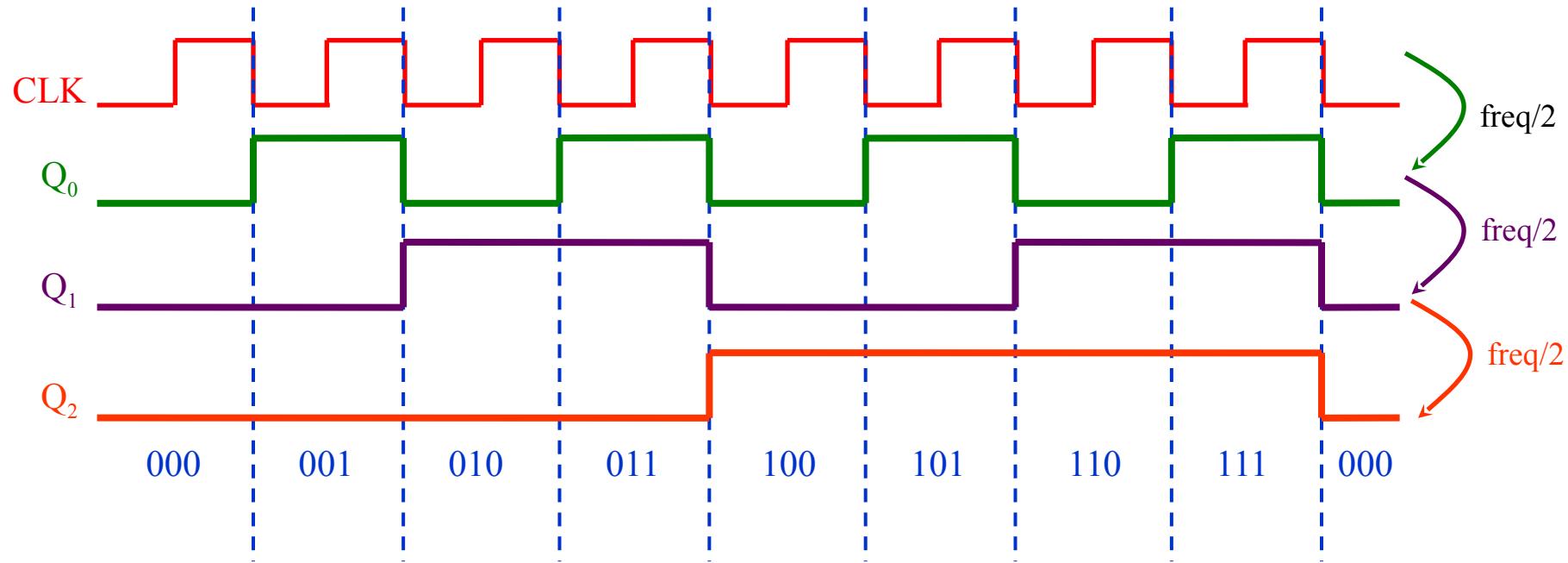
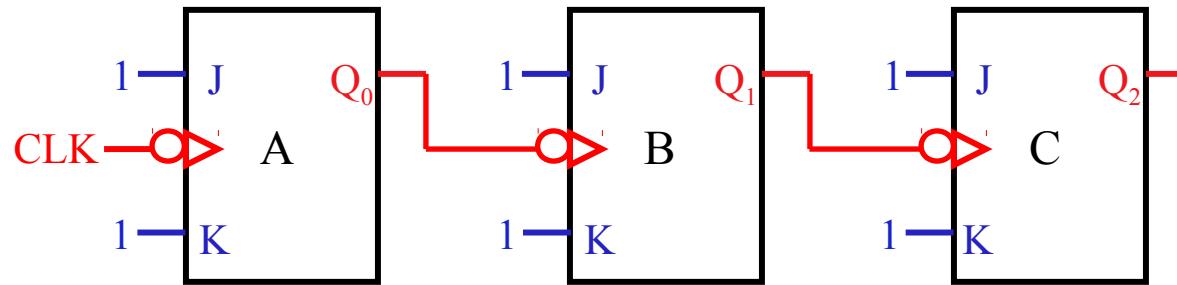


Fig. 9.1 – Funcionamento de um temporizador/contador do ATmega.

Contador/Divisor de Frequência



O ATmega328

- **Conversor Analógico-Digital (ADC)**

- Para o controle de variáveis externas por um sistema digital é necessária a interpretação de grandezas analógicas.
- O emprego de conversores analógico-digitais torna-se imprescindível
 - » O conversor AD do ATmega328 emprega o processo de aproximações sucessivas para converter um sinal analógico em digital.

O ATmega328

- **Seção opcional para código de boot para programação In-System por boot loader**

- Boot loader é um pequeno programa que pode ser escrito no início ou no final da memória de programa e serve para que o microcontrolador gerencie a gravação de sua memória
- No AVR o espaço destinado ao boot loader fica no final da memória de programa e a comunicação se dá através de um dos seus periféricos de comunicação, a USART

O ATmega328

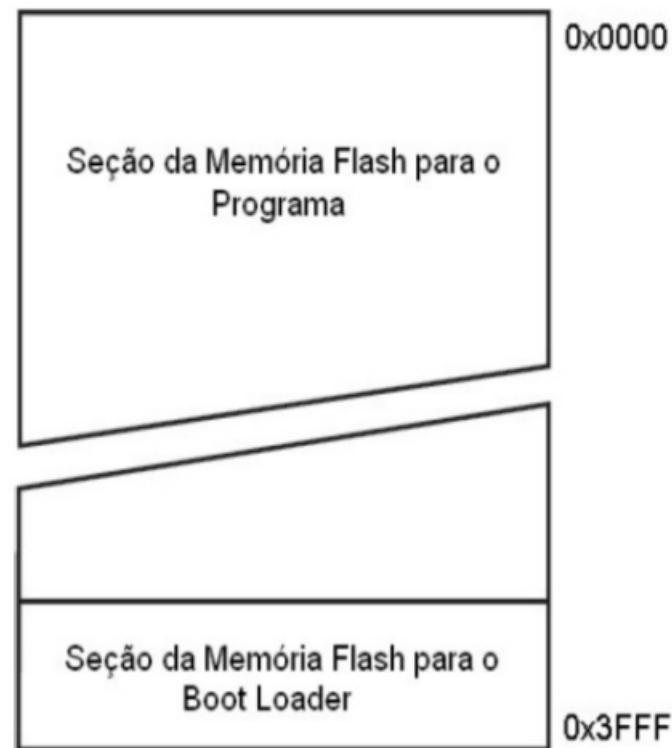
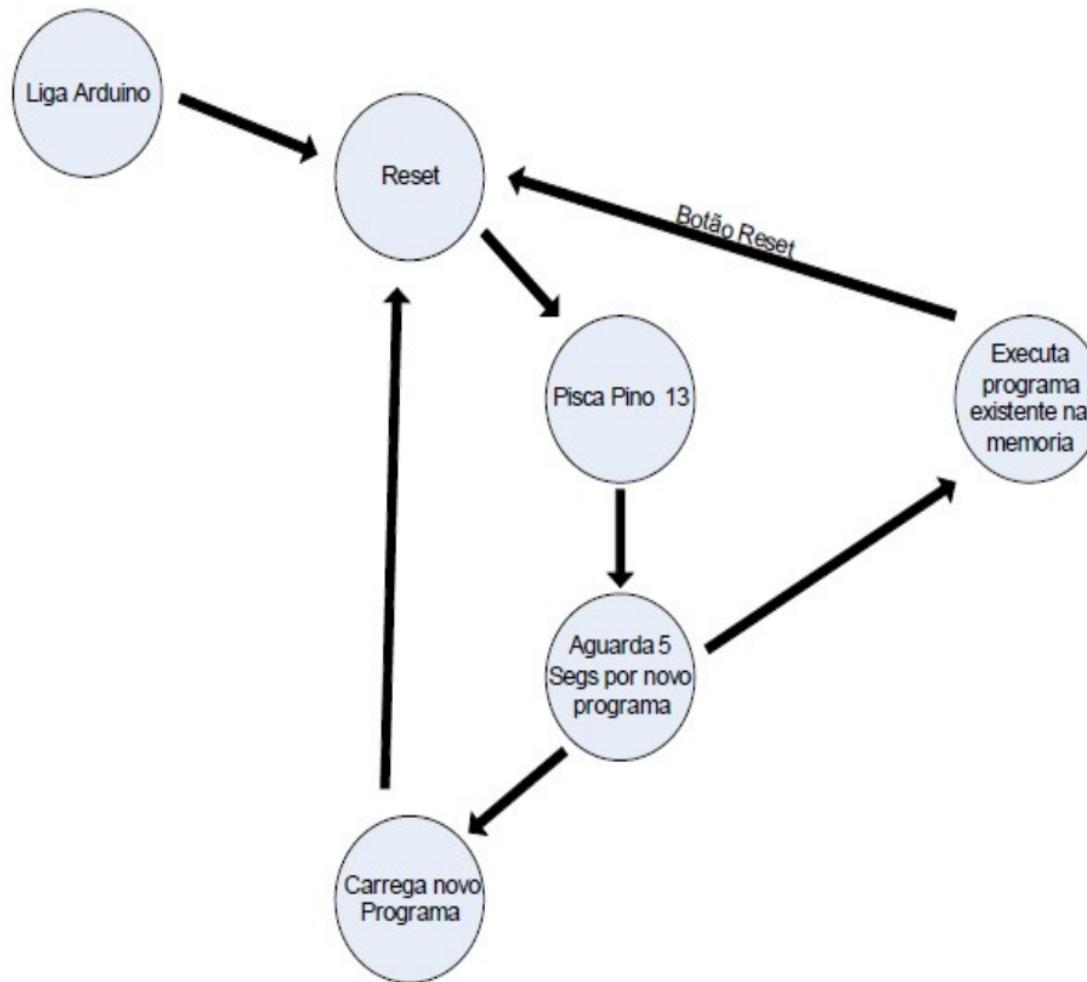


Fig. 2.6 – Organização da memória de programa.

O ATmega328



Boot loader e os estados de funcionamento do Arduino (ATmega328)

O ATmega328

- As instruções do ATmega são de 16 ou 32 bits (a maioria é de 16 bits)
 - Cada instrução consome 2 ou 4 bytes na memória de programa (um byte par e um ímpar)
 - O acesso às posições de memória pelo contador de programa (PC – Program Counter), é realizada de dois em dois bytes
 - Para a memória de 32 Kbytes do ATmega328 são necessários 14 bits de endereçamento ($2^{14} = 16.386$ endereços)

Microcontrolador ATmega328

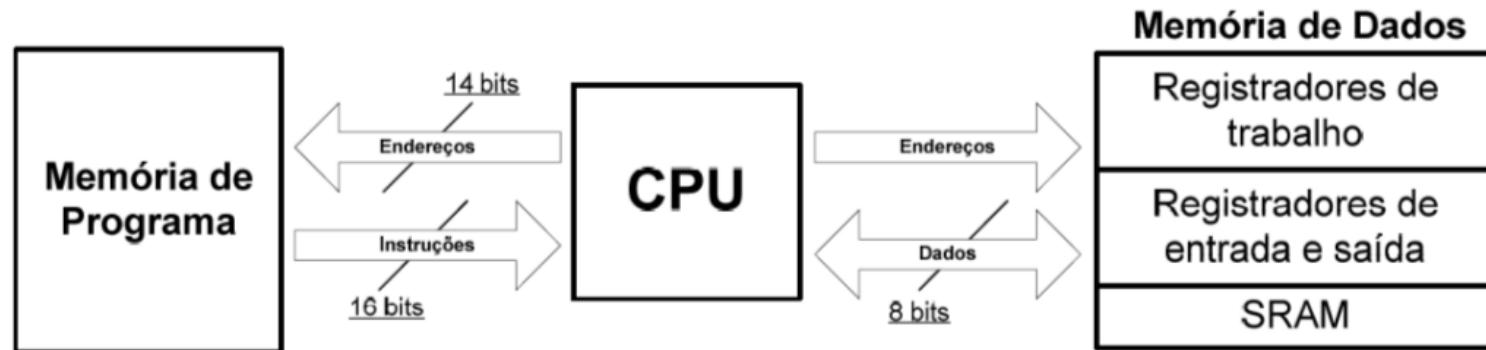


Fig. 2.1 – Diagrama esquemático da estrutura de um microcontrolador ATmega328.

O ATmega328

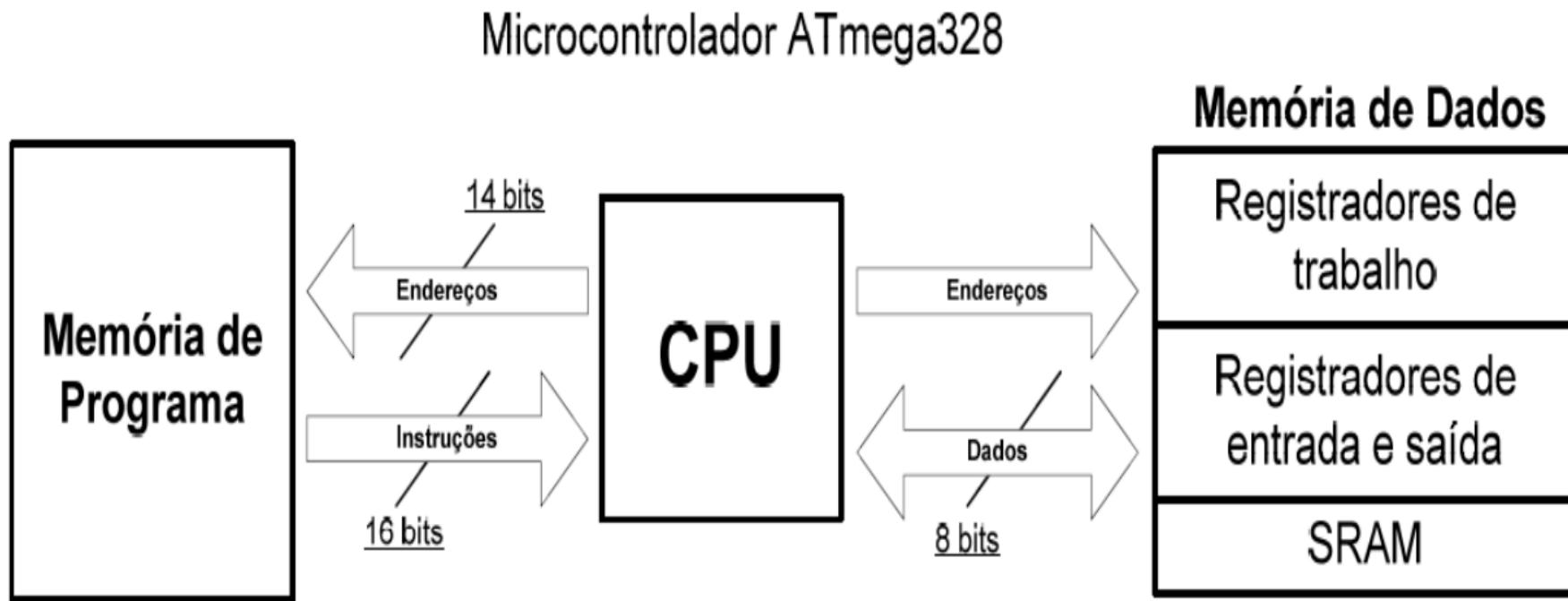
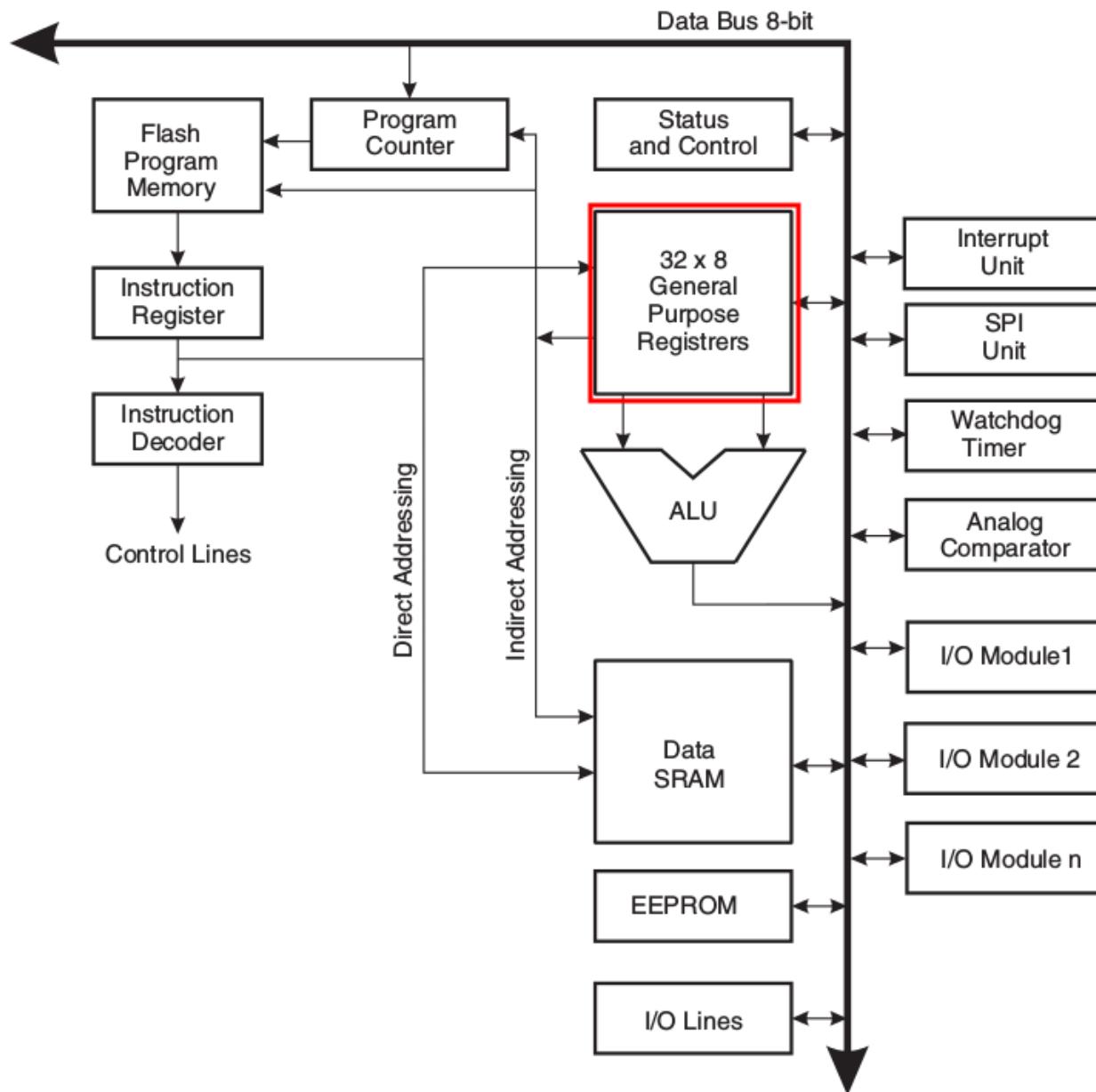


Fig. 2.1 - Diagrama esquemático da estrutura de um microcontrolador ATmega328.

O ATmega328

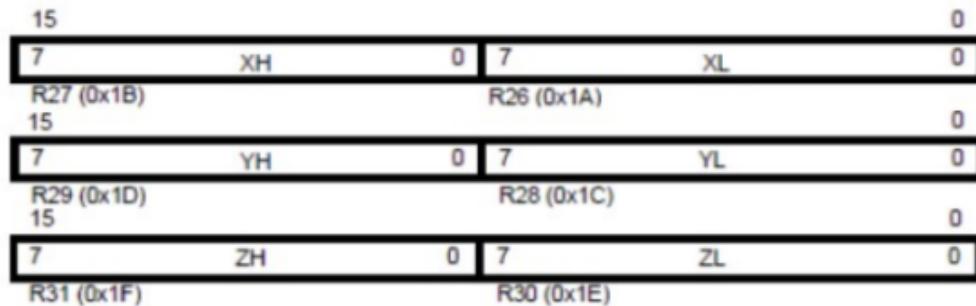
- O núcleo AVR utiliza 32 registradores de trabalho diretamente conectados à ULA
 - Dois registradores independentes podem ser acessados com uma simples instrução em um único ciclo de clock
 - Seis desses registradores podem ser usados como registradores de endereçamento indireto de 16 bits (ponteiros para o acesso de dados), denominados X, Y e Z
 - Os 32 registradores não podem ser empregados em todas as instruções do microcontrolador, pois algumas instruções empregam registradores específicos para o seu trabalho.
 - As cinco instruções lógicas e aritméticas entre uma constante e um registrador SB_{CI}, SUBI, CPI, ANDI e ORI, e a instrução para carga de constante imediata, LDI
 - Essas instruções se aplicam somente a metade superior dos registradores de uso geral (R16...R31)

Figure 7-1. Block Diagram of the AVR Architecture



O ATmega328

Registradores de trabalho



R0	0x00
R1	0x01
R2	0x02
...	
R13	0x0D
R14	0x0E
R15	0x0F
R16	0x10
R17	0x11
...	
X — R26 XL	0x1A
X — R27 XH	0x1B
Y — R28 YL	0x1C
Y — R29 YH	0x1D
Z — R30 ZL	0x1E
Z — R31 ZH	0x1F

Endereços

Fig. 2.2 – Registradores de trabalho da CPU do ATmega.

O ATmega328

- A arquitetura do AVR permite a busca e execução de instruções em paralelo devido ao emprego da técnica de Pipeline
- O desempenho alcançado pode chegar a 1 MIPS (Millions of Instructions Per Second) por MHZ
- Uma instrução é executada enquanto a próxima é lida (buscada e decodificada)

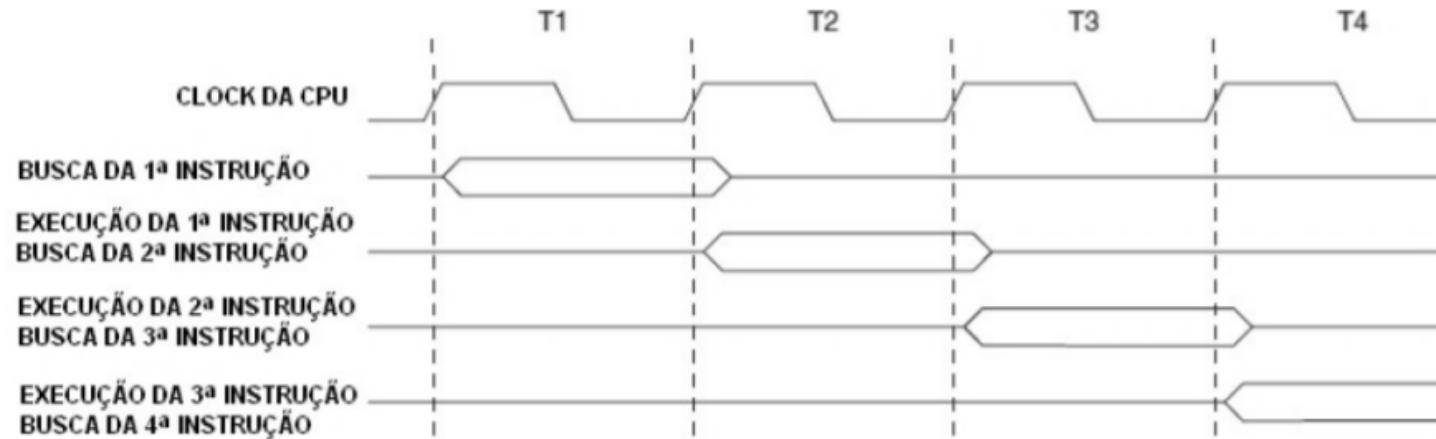


Fig. 2.4 – Diagrama de tempo para a busca e execução de instruções no ATmega.

O ATmega328

- A memória SRAM do ATmega328 é linear, começando no endereço 0 e indo até 0x8FF
 - 32 posições pertencem aos registradores de uso geral (0x000 até 0x01F)
 - 64 aos registradores de entrada e saída (0x020 até 0x05F)
 - 2048 bytes pertencem à memória SRAM (0x060 até 0x8FF)
 - Os 160 primeiros endereços são empregados para a extensão dos registradores de entrada e saída
 - Isso é necessário porque o número de periféricos no ATmega328 é superior ao que pode ser suportado pelo 64 registradores originais (dos primeiros ATmegas)
 - » Também para permitir o acréscimo de futuras funcionalidades

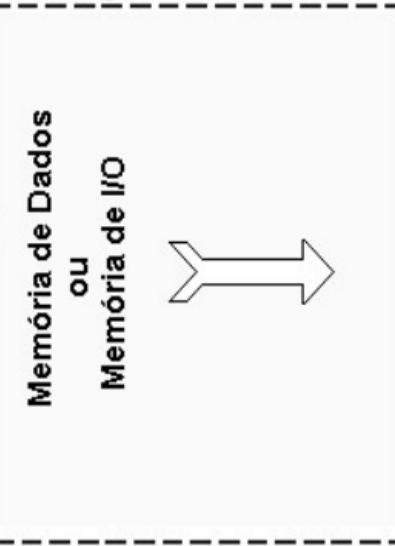
Registradores de Trabalho

R0
R1
R2
...
R29
R30
R31

Espaço dos Endereços de Dados

0x000
0x001
0x002
...
0x01D
0x01E
0x01F

Registradores de I/O



0x020
0x021
0x022
...
0x05D
0x05E
0x05F

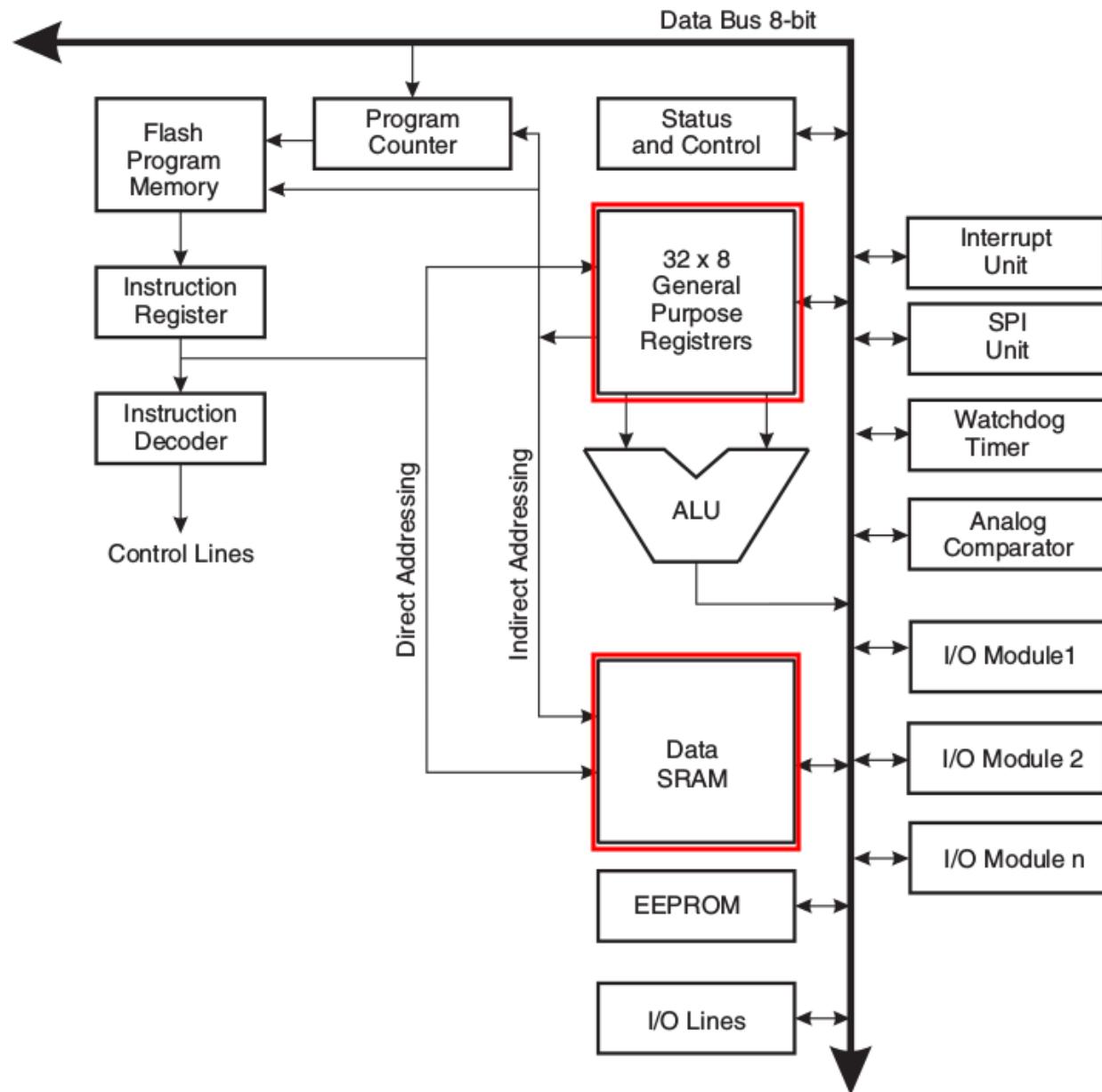
SRAM Interna

0x060
...
0xOFF
0x100
...
0x8FF

Memória Linear

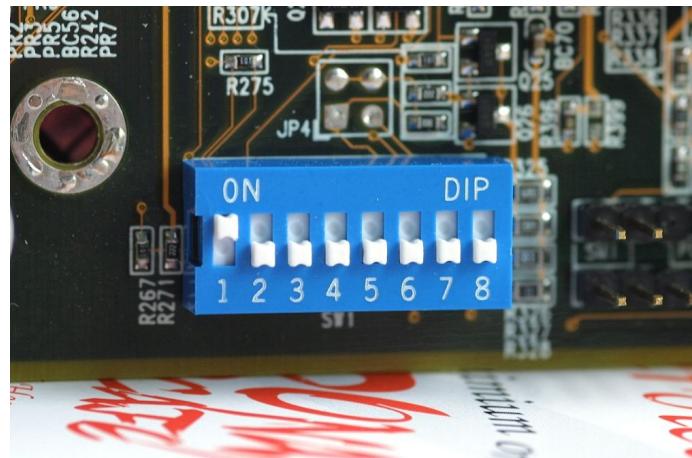
Fig. 2.5 – Memória de dados e memória SRAM do ATmega328.

Figure 7-1. Block Diagram of the AVR Architecture



O ATmega328

- Os Registradores de I/O são o painel de controle do microcontrolador
 - Todas as configurações de trabalho, incluindo acesso às entradas e saídas, se encontram nessa parte da memória
 - É com esses registradores que o programador terá que se familiarizar para trabalhar com os periféricos
 - As “chaves” que ligam e desligam tudo por software...



O ATmega328

- Os Registradores de I/O são o painel de controle do microcontrolador
 - Veja o exemplo dos Registradores de I/O que controlam o PORTB:
 - **PORTB**: registrador de dados, usado para escrever nos pinos do PORTB
 - **DDRB**: registrador de direção, usado para definir se os pinos do PORTB são entrada (0) ou saída (1)
 - **PINB**: registrador de entrada (IN), usado para ler o conteúdo dos pinos do PORTB

O ATmega328

PORTs DE I/O

PORTB - PORT B Data Register

Bit	7	6	5	4	3	2	1	0
PORTB	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0
Lê/Escrive	L/E							
Valor Inicial	0	0	0	0	0	0	0	0

DDRB - PORT B Data Direction Register

Bit	7	6	5	4	3	2	1	0
DDRB	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0
Lê/Escrive	L/E							
Valor Inicial	0	0	0	0	0	0	0	0

PINB - PORT B Input Pins Address

Bit	7	6	5	4	3	2	1	0
PINB	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0
Lê/Escrive	L/E							
Valor Inicial	0	0	0	0	0	0	0	0

File Project Build Edit View Tools Debug Window Help

Trace Disabled

Processor

Name	Value
Program Counter	0x000003
Stack Pointer	0x08FF
X pointer	0x0000
Y pointer	0x0000
Z pointer	0x0000
Cycle Counter	4
Frequency	
Stop Watch	
SREG	1 TH 5 V N Z C
Registers	
R00	0x00
R01	0x00
R02	0x00
R03	0x00
R04	0x00
R05	0x00
R06	0x00
R07	0x00
R08	0x00
R09	0x00
R10	0x00
R11	0x00
R12	0x00
R13	0x00
R14	0x00
R15	0x00
R16	0xFF
R17	0x00
R18	0x00
R19	0x00
R20	0x00
R21	0x00
R22	0x00
R23	0x00
R24	0x00
R25	0x00
R26	0x00

C:\users\paulocg\Meus Documentos\pisca_led.asm

```

// AVR e Arduino: Técnicas de Projeto, 2a ed. - 2012.
//
.include "m328pdef.inc"
.equ LED = PB5 //LED é o substituto de PB5 na programação

.ORG 0x000 //endereço de inicio de escrita do código

INICIO:
    LDI R16,0xFF //carrega R16 com o valor 0xFF
    OUT DDRB,R16 //configura todos os pinos do PORTB como saídas

PRINCIPAL:
    SBI PORTB, LED //coloca o pino PB5 em 5V
    //RCALL ATRASO //chama a sub-rotina de atraso
    CBI PORTB, LED //coloca o pino PB5 em 0V
    //RCALL ATRASO //chama a sub-rotina de atraso
    RJMP PRINCIPAL //volta para PRINCIPAL

ATRASO:
    DEC R3 //decrementa R3, começa com o valor 0x00
    BRNE ATRASO //enquanto R3 > 0 fica decrementando R3, d
    DEC R2 //decrementa R2, começa com o valor 0x00
    BRNE ATRASO //enquanto R2 > 0 volta decrementar R3
    RET //retorno da sub-rotina

```

I/O View

Name	Address	Value	Bits
AD_CONVERTE			
ANALOG_COM			
CPU			
CLKPR	na (0x61)	0x00	██████████
GPIO0	0x1E (0x3E)	0x00	██████████
GPIO1	0x2A (0x4A)	0x00	██████████
GPIO2	0x2B (0x4B)	0x00	██████████
MCUCR	0x35 (0x55)	0x00	██████████
MCUSR	0x34 (0x54)	0x00	██████████
OSCCAL	na (0x66)	0x00	██████████
PRR	na (0x64)	0x00	██████████
SMCR	0x33 (0x53)	0x00	██████████
SP	0x3D (0x5D)	0x08FF	██████████
SPMCSR	0x37 (0x57)	0x00	██████████
SREG	0x3F (0x5F)	0x00	██████████
EEPROM			
EXTERNAL_IN1			
PORTB			
DDRB	0x04 (0x24)	0xFF	██████████
PINB	0x03 (0x23)	0x00	██████████
PORTB	0x05 (0x25)	0x20	██████████
PORTC			
PORTD			
SPI			
TIMER_COUNT			
TIMER_COUNT			
TIMER_COUNT			
TWI			
USART0			
WATCHDOG			

Project Processor

C:\users\paulocg\Meus Documentos\pisca_led.asm

Message

I loaded objectfile: C:\users\paulocg\Meus Documentos\pisca_led.nhi

Message Find in Files Breakpoints and Tracepoints

ATmega328P AVR Simulator Auto Stopped Ln 17, Col 1 CAP NUM OVR

O ATmega328

- **Cada endereço de memória de programa possui 2 bytes, pois as instruções do AVR são de 16 ou 32 bits**
 - A memória possui um total de 16386 endereços (de 0x0000 até 0x3FFF), correspondendo a 32 Kbytes (2 bytes por endereço = 16 bits)
 - A memória flash suporta, no mínimo, 10 mil ciclos de escrita e apagamento
- **A memória EEPROM é de 1 Kbytes e é organizada separadamente**
 - Cada byte individual pode ser lido ou escrito e a memória suporta, no mínimo, 100 mil ciclos de escrita e apagamento

O ATmega328

- Um dos registradores mais importantes do painel de controle do microcontrolador é o SREG
 - O SREG (Status Register) indica, através de bits individuais:
 - O estado das operações lógicas e aritméticas da CPU
 - Permite habilitar ou não as interrupções (chave geral)
 - Seus bits são empregados na programação para tomadas de decisões e na realização de operações lógico aritméticas

SREG – STATUS REGISTER

Bit	7	6	5	4	3	2	1	0
SREG	I	T	H	S	V	N	Z	Z
Lê/Escrve	L/E							
Valor Inicial	0	0	0	0	0	0	0	0

O ATmega328

SREG – STATUS REGISTER

Bit	7	6	5	4	3	2	1	0
SREG	I	T	H	S	V	N	Z	Z
Lê/Escrive	L/E							
Valor Inicial	0	0	0	0	0	0	0	0

Bit 7 – I: *Global Interrupt Enable*.

Bit 6 – T: *Bit Copy Storage*.

Bit 5 – H: *Half Carry Flag*.

Bit 4 – S: *Sign Bit*, $S = N \oplus V$.

Bit 3 – V: *Two's Complement Overflow Flag*.

Bit 2 – N: *Negative Flag*.

Bit 1 – Z: *Zero Flag*.

Bit 0 – C: *Carry Flag*.

O ATmega328

- **Bit 7 – I Global Interrupt Enable**

- Esse bit é a chave geral para habilitar as interrupções
- Cada interrupção individual possui seus registradores de controle
- O bit I é limpo quando uma interrupção ocorre (impedindo que outras ocorram simultaneamente) e volta a ser ativo quando se termina o tratamento da interrupção (instrução RETI)

- **Bit 6 – T Bit Copy Storage**

- Serve para copiar o valor de um bit de um registrador ou escrever o valor de um bit em um registrador
- Instruções BLD e BST

SREG – *STATUS REGISTER*

Bit	7	6	5	4	3	2	1	0
SREG	I	T	H	S	V	N	Z	Z
Lê/Escrve	L/E							
Valor Inicial	0	0	0	0	0	0	0	0

O ATmega328

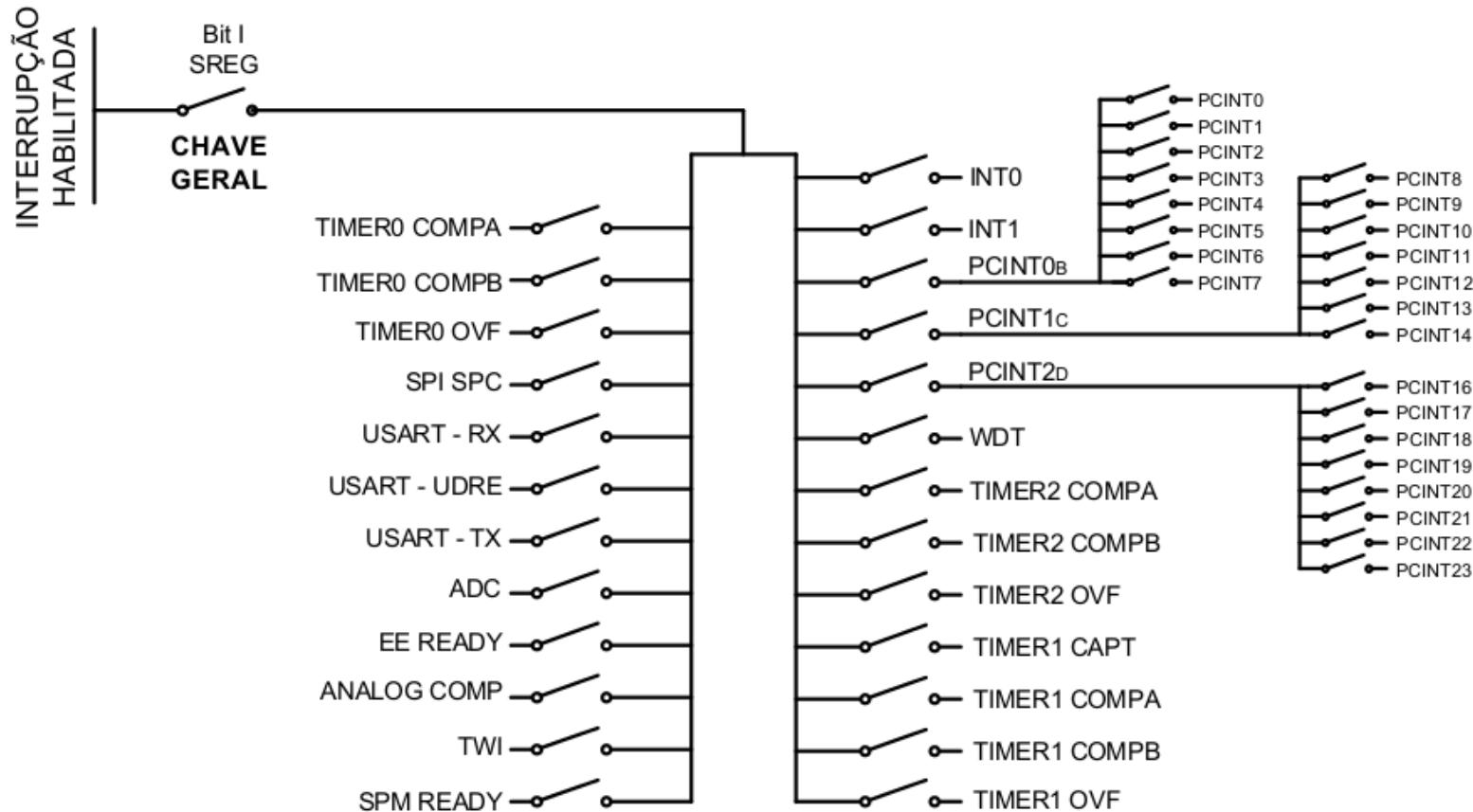


Fig. 6.1 – Chaves de habilitação das interrupções.

O ATmega328

- **Bit 5 – H Half Carry Flag**

- Indica quando um Carry auxiliar (em um nibble) ocorreu em alguma operação aritmética.

- **Bit 4 – S Sign Bit, $S = N \oplus V$**

- O bit S é o resultado de uma operação ou-exclusivo entre o bit de sinalização negativo N e o bit de sinalização de estouro do complemento de dois V

SREG – *STATUS REGISTER*

Bit	7	6	5	4	3	2	1	0
SREG	I	T	H	S	V	N	Z	Z
Lê/Escrve	L/E							
Valor Inicial	0	0	0	0	0	0	0	0

O ATmega328

- **Bit 3 – V Two's Complement Overflow Flag**
 - O bit de sinalização de estouro do complemento de dois, ajuda na aritmética em complemento de dois
- **Bit 2 – N Negative Flag**
 - O bit de sinalização negativo indica quando uma operação aritmética ou lógica resulta em um valor negativo

SREG – *STATUS REGISTER*

Bit	7	6	5	4	3	2	1	0
SREG	I	T	H	S	V	N	Z	Z
Lê/Escrve	L/E							
Valor Inicial	0	0	0	0	0	0	0	0

O ATmega328

- **Bit 1 – Z Zero Flag**
 - O bit de sinalização zero indica quando uma operação aritmética ou lógica resulta em zero
- **Bit 0 – C Carry Flag**
 - O bit de sinalização de Carry indica quando houve um estouro numa operação aritmética

SREG – *STATUS REGISTER*

Bit	7	6	5	4	3	2	1	0
SREG	I	T	H	S	V	N	Z	Z
Lê/Escrve	L/E							
Valor Inicial	0	0	0	0	0	0	0	0

O ATmega328

Mnemonics	Operands	Description	Operation	Flags	#Clocks
BRANCH INSTRUCTIONS					
RJMP	k	Relative Jump	$PC \leftarrow PC + k + 1$	None	2
IJMP		Indirect Jump to (Z)	$PC \leftarrow Z$	None	2
JMP ⁽¹⁾	k	Direct Jump	$PC \leftarrow k$	None	3
RCALL	k	Relative Subroutine Call	$PC \leftarrow PC + k + 1$	None	3
ICALL		Indirect Call to (Z)	$PC \leftarrow Z$	None	3
CALL ⁽¹⁾	k	Direct Subroutine Call	$PC \leftarrow k$	None	4
RET		Subroutine Return	$PC \leftarrow STACK$	None	4
RETI		Interrupt Return	$PC \leftarrow STACK$	I	4
CPSE	Rd,Rr	Compare, Skip if Equal	if (Rd = Rr) $PC \leftarrow PC + 2$ or 3	None	1/2/3
BREQ	k	Branch if Equal	if (Z = 1) then $PC \leftarrow PC + k + 1$	None	1/2
BRNE	k	Branch if Not Equal	if (Z = 0) then $PC \leftarrow PC + k + 1$	None	1/2
BRCS	k	Branch if Carry Set	if (C = 1) then $PC \leftarrow PC + k + 1$	None	1/2
BRCC	k	Branch if Carry Cleared	if (C = 0) then $PC \leftarrow PC + k + 1$	None	1/2
BRSH	k	Branch if Same or Higher	if (C = 0) then $PC \leftarrow PC + k + 1$	None	1/2
BRLO	k	Branch if Lower	if (C = 1) then $PC \leftarrow PC + k + 1$	None	1/2

O ATmega328

- O Stack Pointer (SP, ponteiro de pilha) é um registrador que armazena um endereço correspondente a uma posição da memória RAM
- Esta posição da memória RAM é utilizada na forma de uma pilha para armazenagem temporária de dados:
 - Variáveis e endereços de retorno após chamadas de sub-rotinas
 - Interrupções

O ATmega328

- O endereço do SP é pós-decrementado toda vez que um dado é colocado na pilha
 - Assim, a cada novo dado colocado na pilha, o endereço do SP apresenta um valor menor que o anterior
- Quando um dado é retirado da pilha, o endereço do SP é pré incrementado
 - Sempre apontando uma posição acima do último dado válido da pilha.

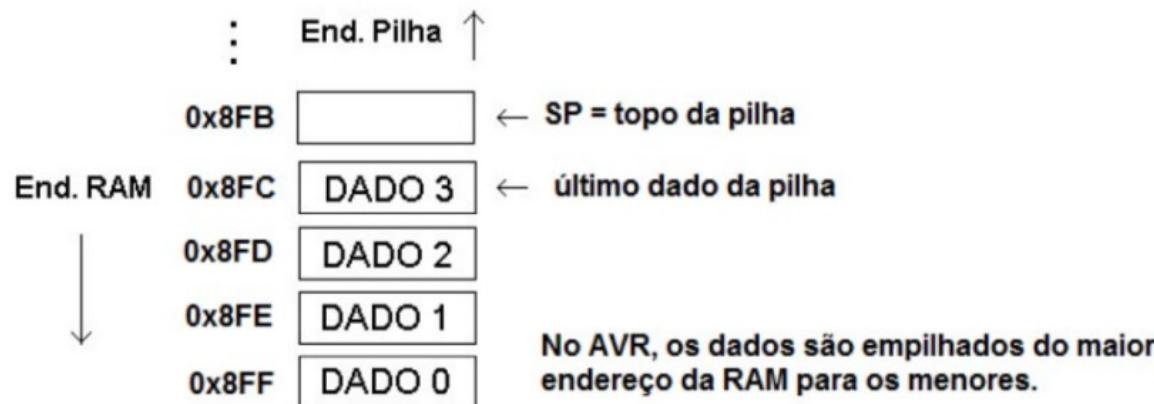


Fig. 2.7 – Stack Pointer.

O ATmega328

Tab. 2.2: Instruções do ATmega que afetam o Stack Pointer.

Instrução	Stack Pointer	Descrição
PUSH	Decrementa 1	Um dado é colocado na pilha (1 byte).
ICALL RCALL	Decrementa 2	O endereço de retorno é colocado na pilha quando uma chamada de sub-rotina ou interrupção acontece (o endereço possui 2 bytes).
POP	Incrementa 1	O dado do topo da pilha é retirado (1 byte).
RET RETI	Incrementa 2	O endereço de retorno é retirado da pilha quando se retorna de uma sub-rotina ou interrupção (o endereço possui 2 bytes).

O ATmega328

- **Dado o funcionamento do SP, na sua inicialização ele deve apontar para o endereço final da RAM**
 - No caso do ATmega328 o endereço é 0x8FF (seu valor padrão após a energização)
- **Existem microcontroladores da família ATmega que precisam ter o SP inicializado pelo programador**
 - Quando o programa é escrito em assembly a inicialização deve ser feita pelo programador de forma explícita
 - Essa inicialização é feita automaticamente quando o programa é escrito na linguagem C
 - » O compilador C se encarrega da tarefa

O ATmega328

- **Como o SP armazena um endereço da RAM, ele deve ter um número de bits suficiente para isso**
- **Como o ATmega possui registradores de 8 bits, são necessários dois registradores para o SP**
 - Um armazena a parte baixa do endereço (SP Low)
 - Outro armazena a parte alta do endereço (SP High)
 - Resultando num registrador de 12 bits (os 4 bits mais significativos do SPH não são utilizados)

Bit	15	14	13	12	11	10	9	8
SPH	-	-	-	-	SP11	SP10	SP9	SP8
SPL	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0
Bit	7	6	5	4	3	2	1	0
Lê/Escrve	L/E	L/E	L/E	L/E	L/E	L/E	L/E	L/E
	L/E	L/E	L/E	L/E	L/E	L/E	L/E	L/E
Valor Inicial	-	-	-	-	1	0	0	0
	1	1	1	1	1	1	1	1

Fig. 2.8- Detalhamento dos registradores do *Stack Pointer* (valor inicial 0x8FF).

O ATmega328

DESCRÍÇÃO DOS PINOS

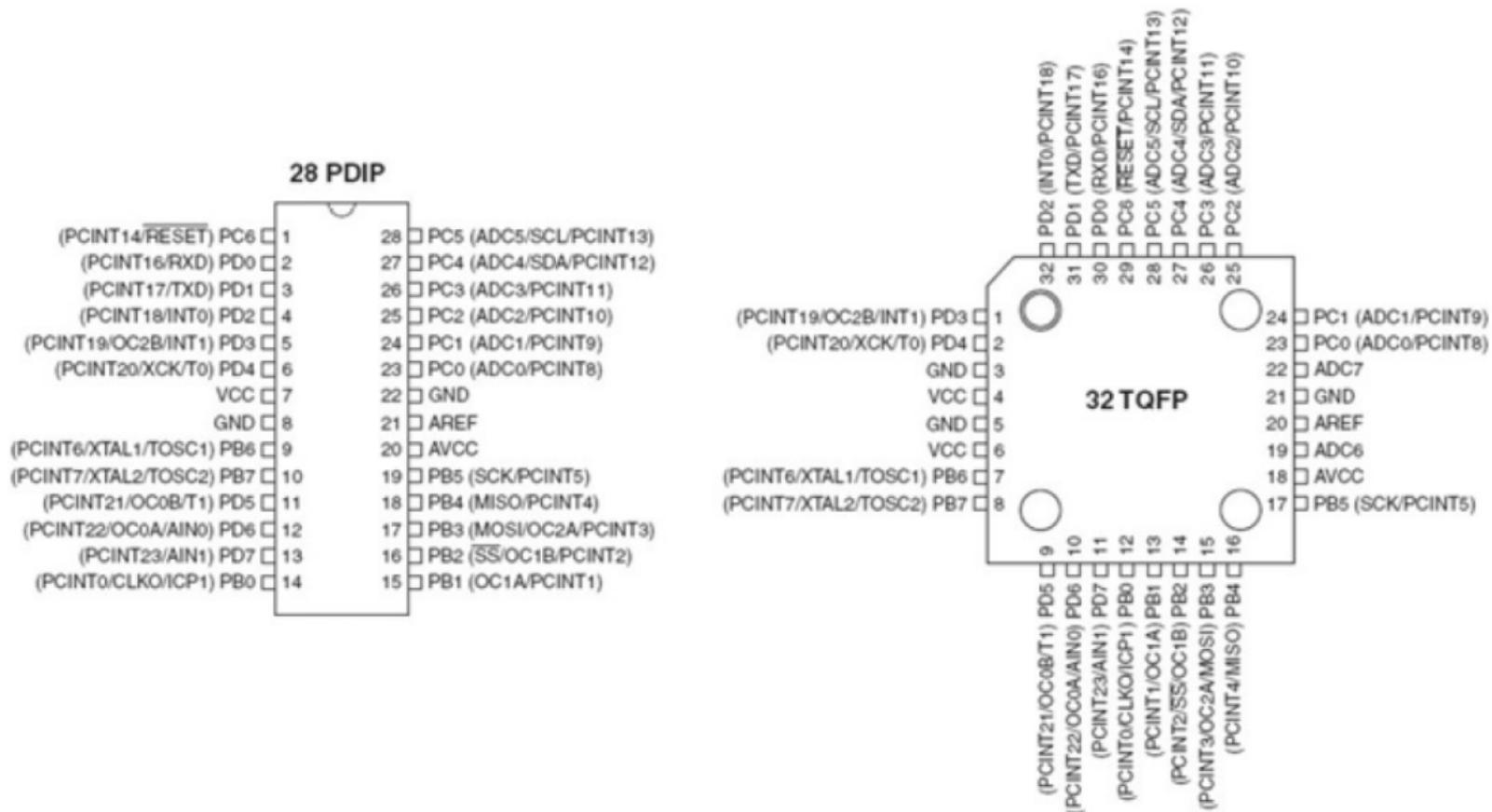


Fig. 1 - Encapsulamentos PDIP e TQFP para o ATmega328.

Tab. 1 - Descrição dos pinos do ATmega328.

PINOS DE ALIMENTAÇÃO	
VCC	Tensão de alimentação.
AVCC	Pino para a tensão de alimentação do conversor AD. Deve ser externamente conectado ao VCC, mesmo se o ADC não estiver sendo utilizado.
AREF	Pino para a tensão de referência analógica do conversor AD.
GND	Terra.

PORTB	
PB0	ICP1 - entrada de captura para o Temporizador/Contador 1. CLKO – saída de <i>clock</i> do sistema. PCINT0 – interrupção 0 por mudança no pino.
PB1	OC1A – saída da igualdade de comparação A do Temporizador/Contador 1 (PWM). PCINT1 - interrupção 1 por mudança no pino.
PB2	SS – pino de seleção de escravo da SPI (<i>Serial Peripheral Interface</i>). OC1B - saída da igualdade de comparação B do Temporizador/Contador 1 (PWM). PCINT2 - interrupção 2 por mudança no pino.
PB3	MOSI – pino mestre de saída e escravo de entrada da SPI. OC2A - saída da igualdade de comparação A do Temporizador/Contador 2 (PWM). PCINT3 - interrupção 3 por mudança no pino.
PB4	MISO – pino mestre de entrada e escravo de saída da SPI. PCINT4 - interrupção 4 para mudança no pino.
PB5	SCK – pino de <i>clock</i> da SPI. PCINT5 - interrupção 5 por mudança no pino.
PB6	XTAL1 – entrada 1 do oscilador ou entrada de <i>clock</i> externa. TOSC1 – entrada 1 para o oscilador do temporizador (RTC). PCINT6 - interrupção 6 por mudança no pino.
PB7	XTAL2 – entrada 2 do oscilador. TOSC2 – entrada 2 para o oscilador do temporizador (RTC). PCINT7 - interrupção 7 por mudança no pino.

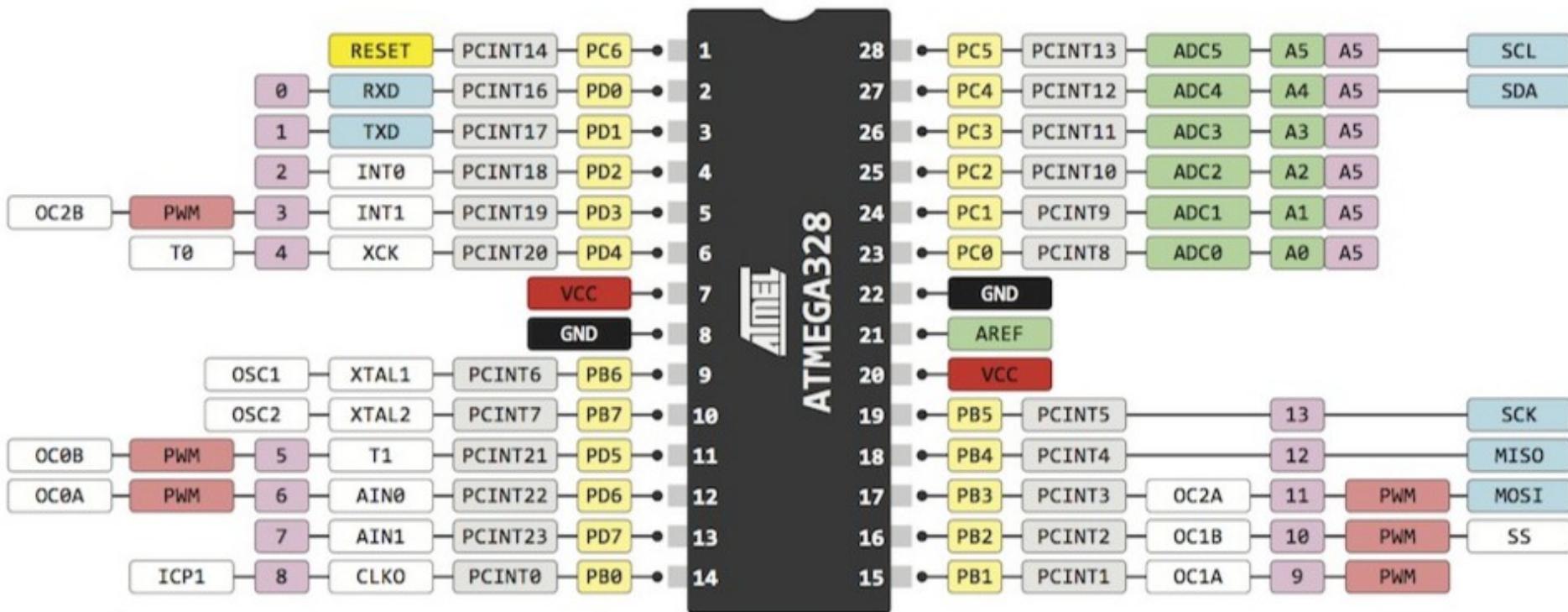
PORTC	
PC0	ADC0 – canal 0 de entrada do conversor AD. PCINT8 - interrupção 8 por mudança no pino.
PC1	ADC1 – canal 1 de entrada do conversor AD. PCINT9 - interrupção 9 por mudança no pino.
PC2	ADC2 – canal 2 de entrada do conversor AD. PCINT10 - interrupção 10 por mudança no pino.
PC3	ADC3 – canal 3 de entrada do conversor AD. PCINT11 - interrupção 11 por mudança no pino.
PC4	ADC4 – canal 4 de entrada do conversor AD. SDA – entrada e saída de dados da interface a 2 fios (TWI – I2C). PCINT12 - interrupção 12 por mudança no pino.
PC5	ADC5 – canal 5 de entrada do conversor AD. SCL – <i>clock</i> da interface a 2 fios (TWI – I2C). PCINT13 - interrupção 13 por mudança no pino.
PC6	RESET – pino de inicialização. PCINT14 - interrupção 14 por mudança no pino.

PORTD	
PD0	RXD – pino de entrada (leitura) da USART. PCINT16 - interrupção 16 por mudança no pino.
PD1	TXD – pino de saída (escrita) da USART. PCINT17 - interrupção 17 por mudança no pino.
PD2	INT0 – entrada da interrupção externa 0. PCINT18 - interrupção 18 por mudança no pino.
PD3	INT1 – entrada da interrupção externa 1. OC2B – saída da igualdade de comparação B do Temporizador/Contador 2 (PWM) PCINT19 - interrupção 19 por mudança no pino.
PD4	XCK – <i>clock</i> externo de entrada e saída da USART. T0 – entrada de contagem externa para o Temporizador/Contador 0. PCINT 20 - interrupção 20 por mudança no pino.
PD5	T1 – entrada de contagem externa para o Temporizador/Contador 1. OC0B - saída da igualdade de comparação B do Temporizador/Contador 0 (PWM). PCINT 21 - interrupção 21 por mudança no pino.
PD6	AIN0 – entrada positiva do comparador analógico. OC0A - saída da igualdade de comparação A do Temporizador/Contador 0 (PWM). PCINT 22 - interrupção 22 por mudança no pino.
PD7	AIN1 – entrada negativa do comparador analógico. PCINT 23 - interrupção 23 por mudança no pino.

Tab. 2 – Correlação entre os pinos do Arduino e do ATmega328.

Arduino	ATmega328	Arduino	ATmega328	Arduino	Atmega328
<i>Analog In</i>	PORTC		PORTD		PORTB
A0	PC0	0	PD0	8	PB0
A1	PC1	1	PD1	9	PB1
A2	PC2	2	PD2	10	PB2
A3	PC3	3	PD3	11	PB3
A4	PC4	4	PD4	12	PB4
A5	PC5	5	PD5	13	PB5
		6	PD6		
		7	PD7		

THE
DEFINITIVE
ATMEGA328
&Arduino
PINOUT DIAGRAM



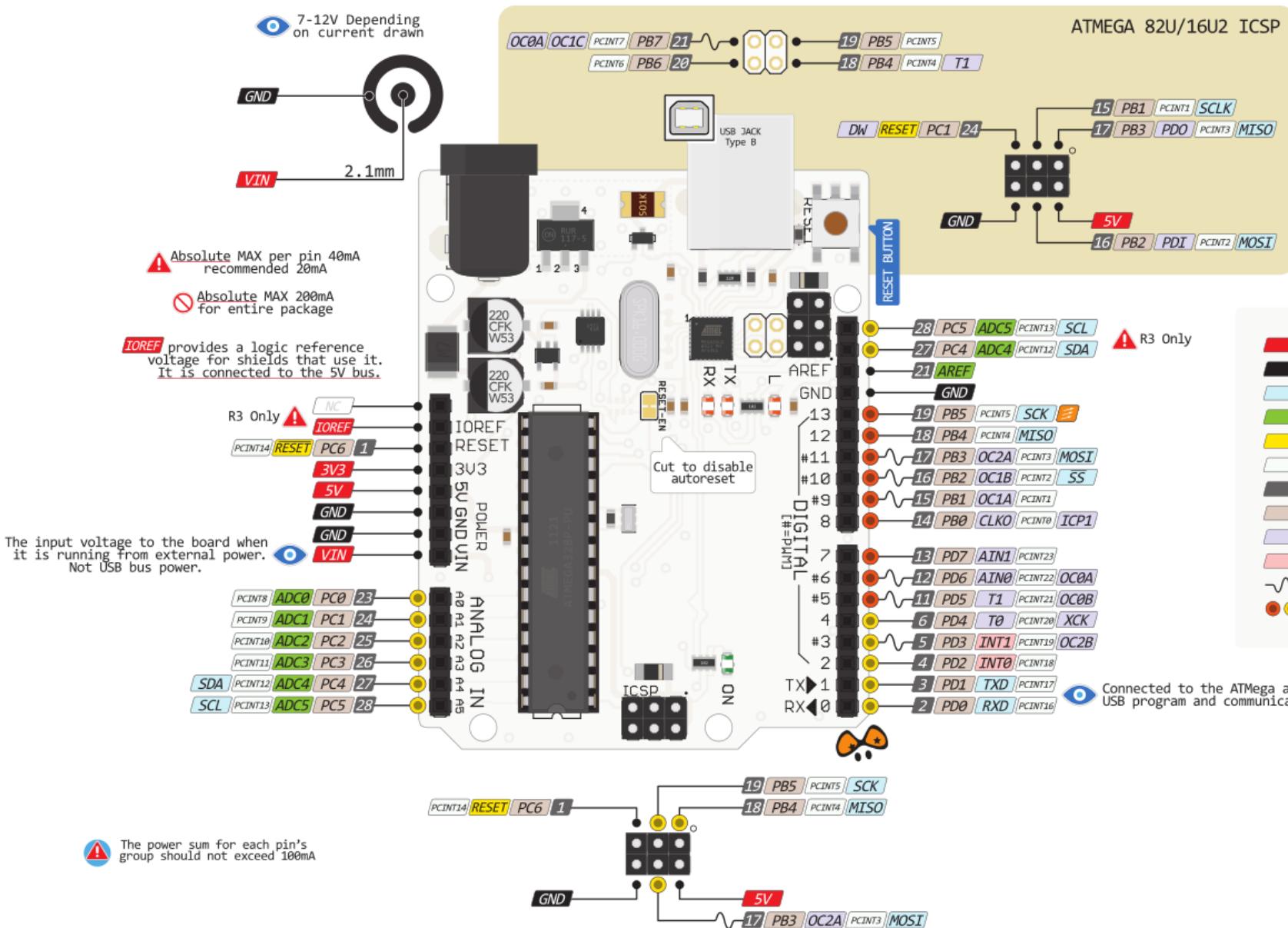
www.pjlxkx.com



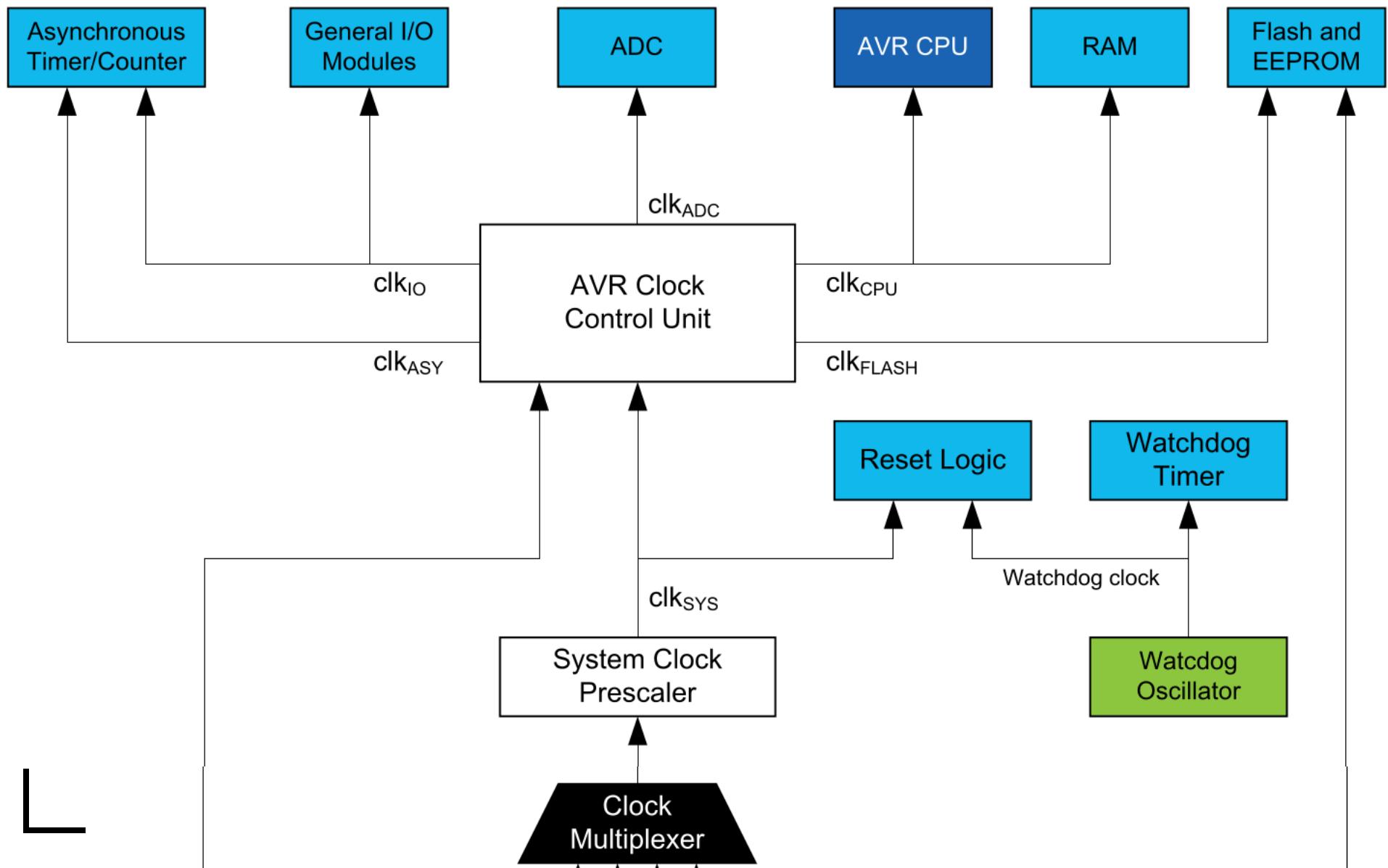
18 FEB 2013

ver 2 rev 0 - 19.02.2013

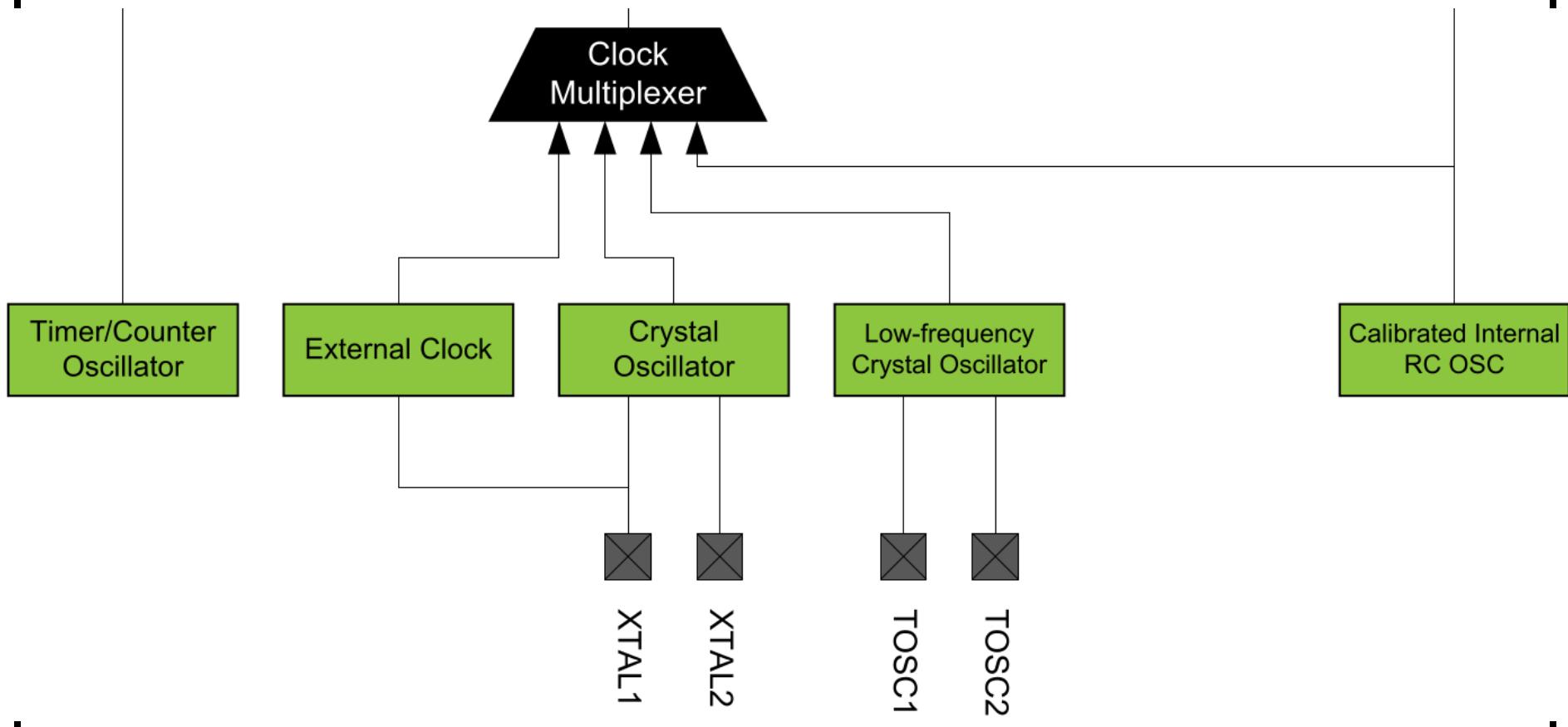
UNO PINOUT



Sistema de clock



Sistema de clock

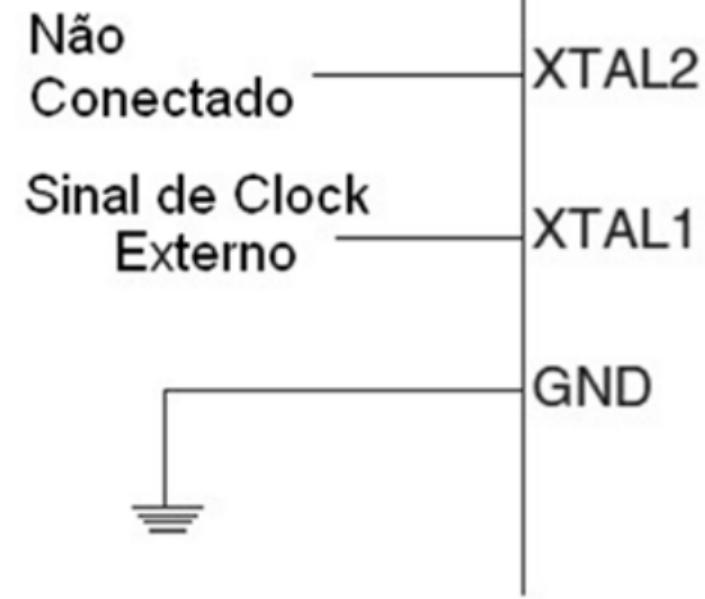
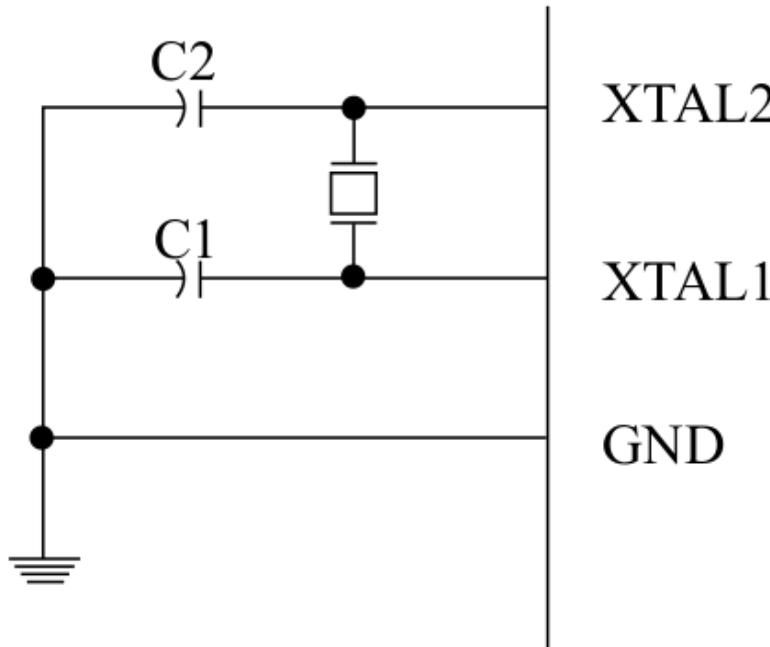


Origem do clock

Table 13-1. Device Clocking Options Select

Device Clocking Option	CKSEL[3:0]
Low Power Crystal Oscillator	1111 - 1000
Full Swing Crystal Oscillator	0111 - 0110
Low Frequency Crystal Oscillator	0101 - 0100
Internal 128kHz RC Oscillator	0011
Calibrated Internal RC Oscillator	0010
External Clock	0000
Reserved	0001

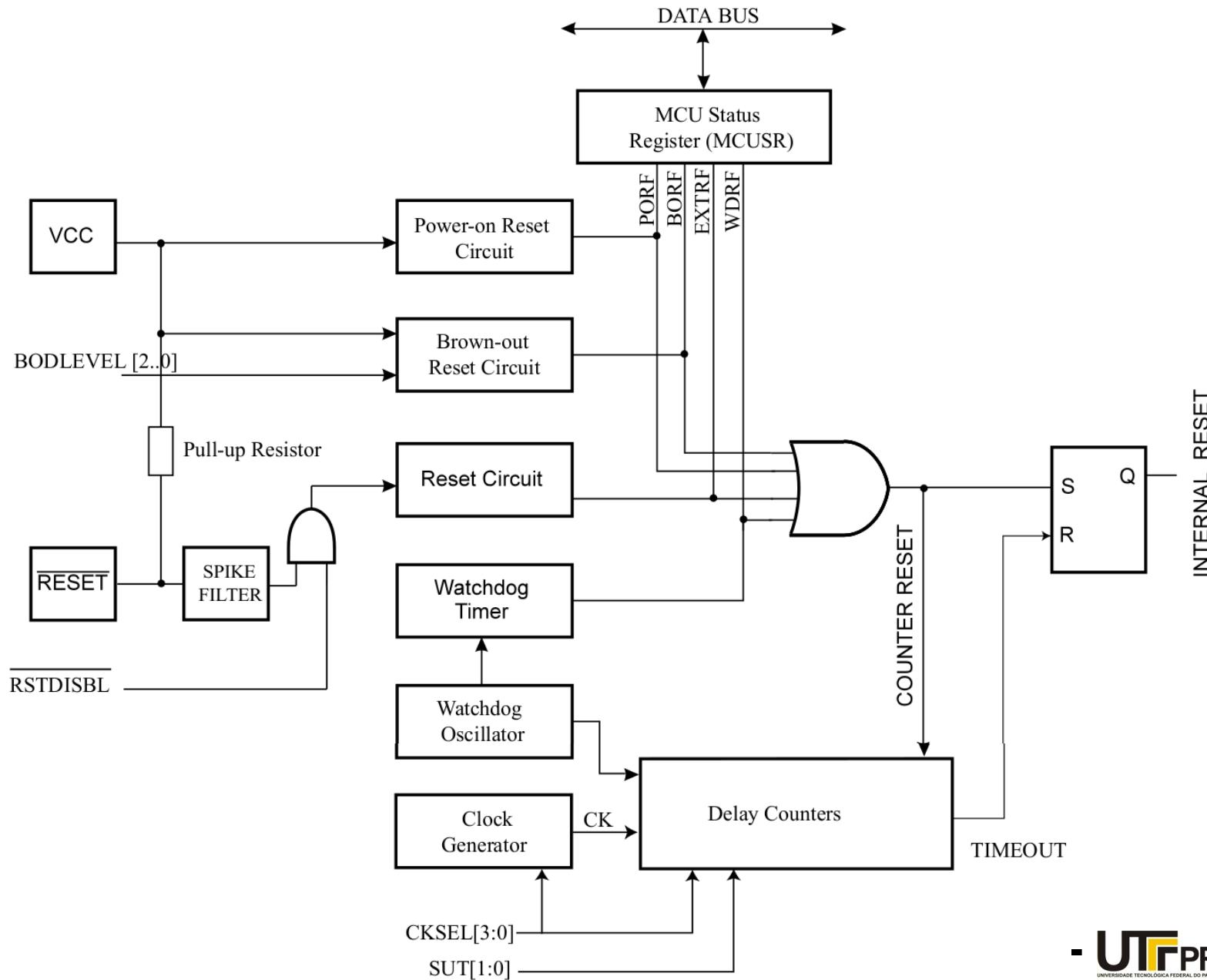
Opções de clock externo



b)

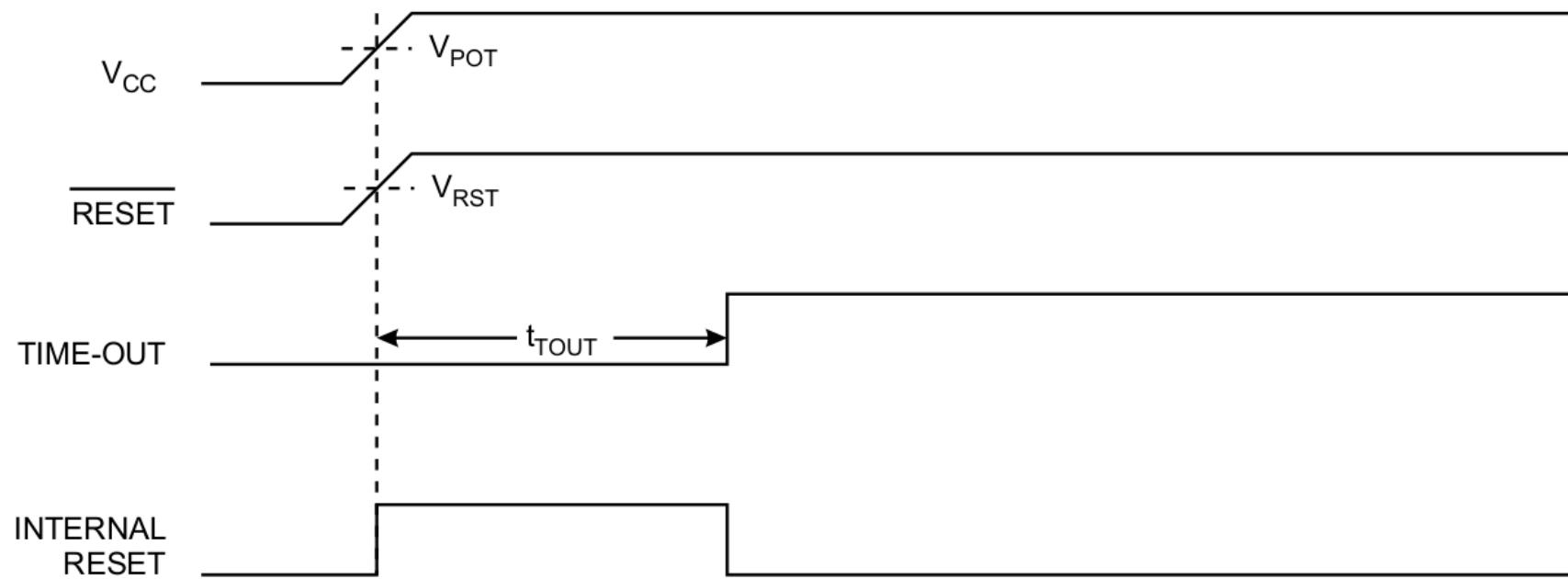
Opções de *clock* externo para o AVR: a) cristal e b) sinal externo.

RESET



Power-on RESET

Figure 15-2. MCU Start-up, $\overline{\text{RESET}}$ Tied to V_{CC}



Power-on RESET

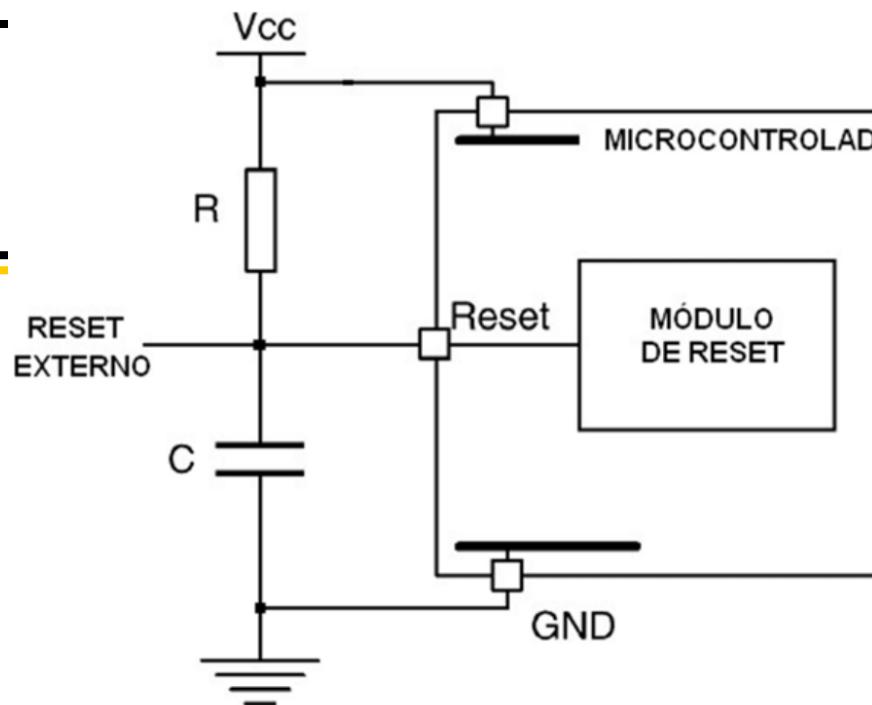
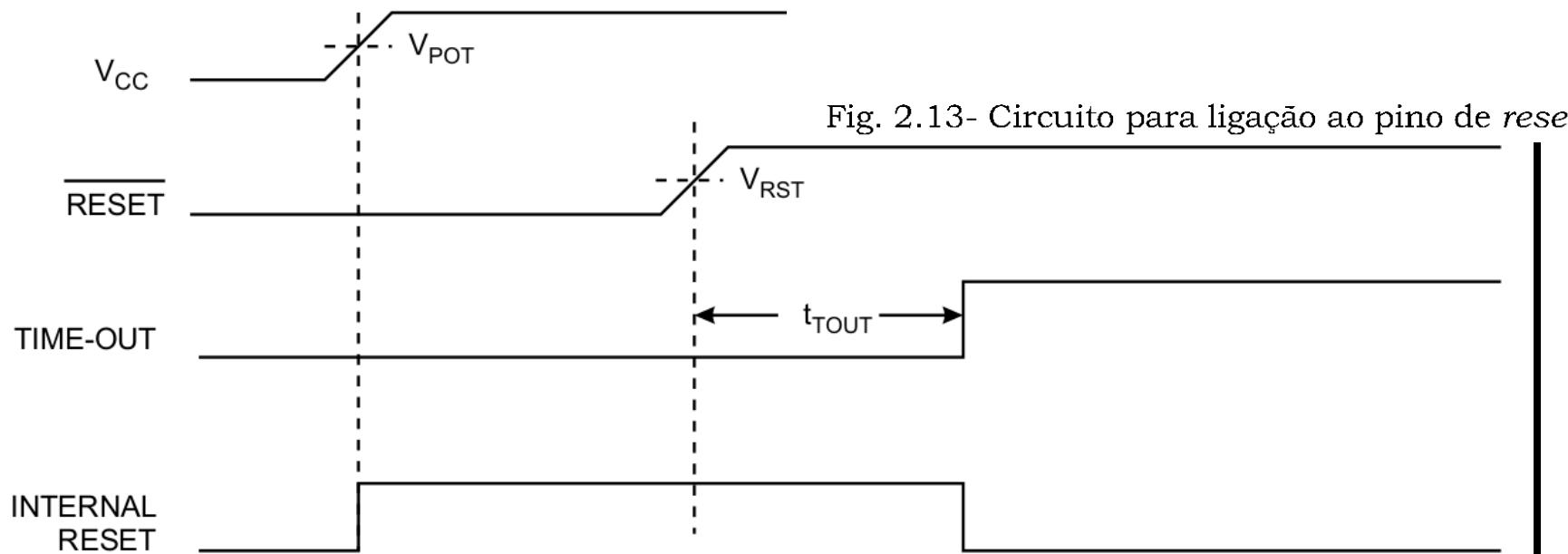
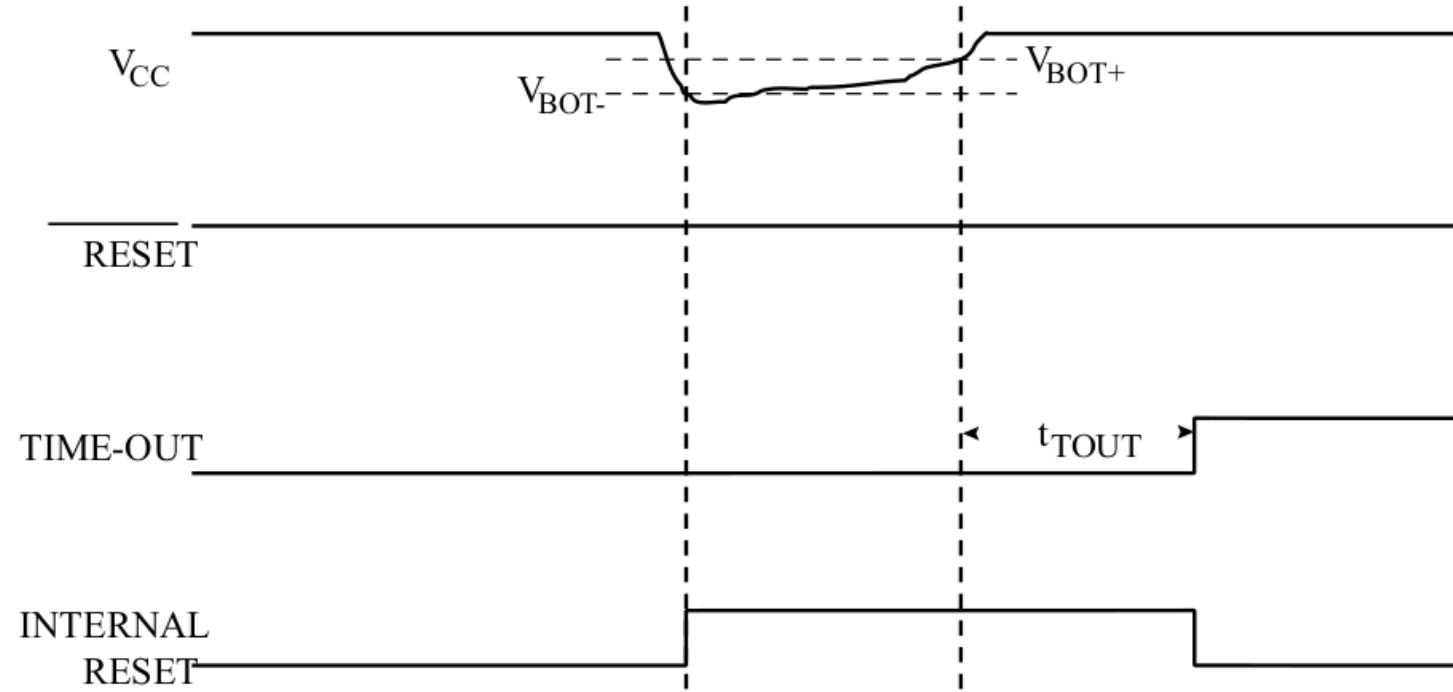


Figure 15-3. MCU Start-up, RESET Extended Externally



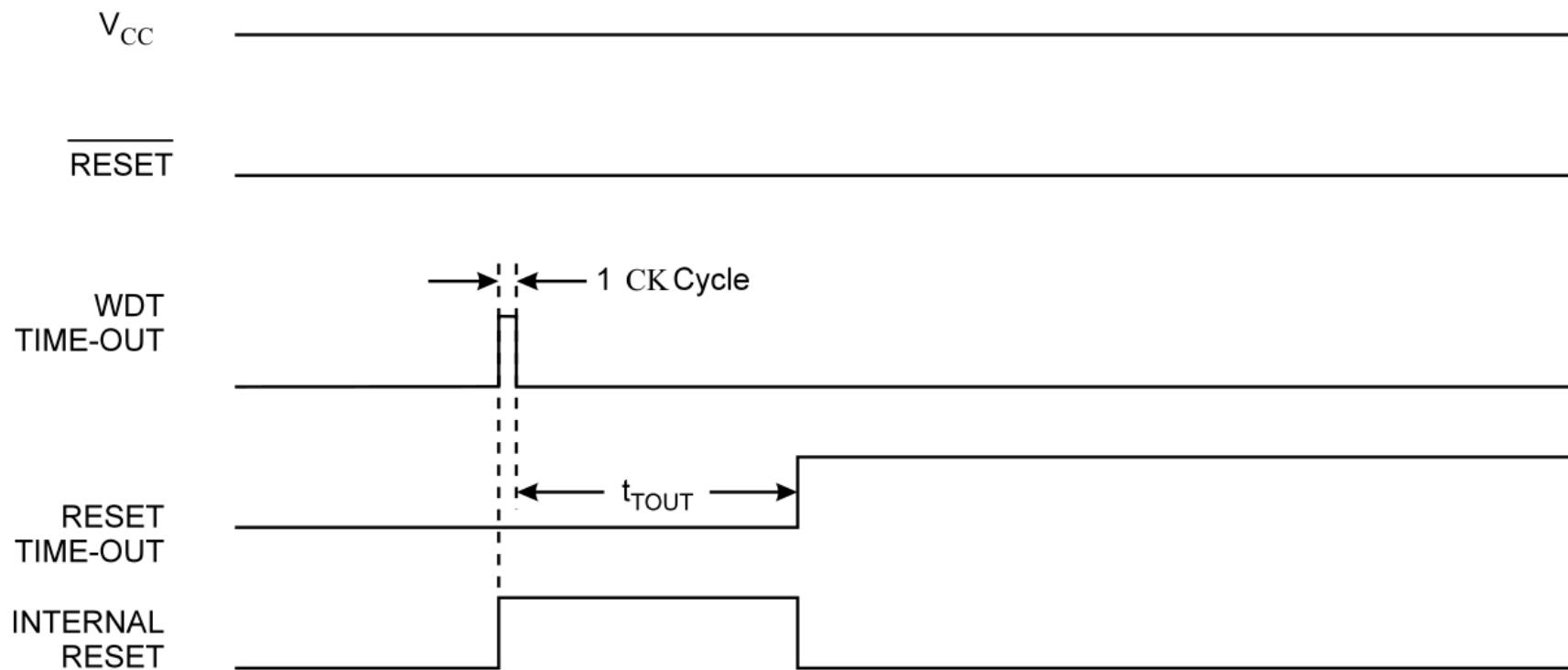
Brown-out RESET

Figure 15-5. Brown-out Reset During Operation



Watchdog RESET

Figure 15-6. Watchdog System Reset During Operation



Gerenciamento de energia

Tab. 2.4 – Fontes desligadas e sinais de ‘despertar’ para os diferentes modos *sleep*.

Modos Sleep	Sinais de <i>clock</i> ativos					Osciladores		Fontes de <i>wake-up</i> (para ‘despertar’)							BOD ⁴ desabilitado por software
	Clock CPU	Clock FLASH	Clock IO	Clock ADC	Clock ASY	Fonte principal de <i>clock</i> habilitada	Oscilador do Temporizador habilitado	INT1, INT0 e mudança nos pinos	Casamento de endereço da interface serial à 2 fios	Temporizador 2	SPM/EEPROM prontos	ADC	Watchdog Timer	Outras I/O	
<i>Idle</i>			X	X	X	X	X ²	X ³	X	X	X	X	X	X	
Redução de Ruído para o ADC			X	X	X	X ²	X ³	X	X ²	X	X	X	X		
<i>Power-Down</i>								X ³	X				X		X
<i>Power-Save</i>					X		X ²	X ³	X	X			X		X
<i>Standby</i> ¹					X			X ³	X				X		X
<i>Extended Standby</i>					X ²	X	X ²	X ³	X	X			X		X

Gerenciamento de energia

- Idle: a atividade da CPU é suspensa, mas a SPI, a USART, o comparador analógico, o ADC, a interface serial à 2 fios, os temporizadores/contadores, o *watchdog timer* e o sistema de interrupção continuam operando. Basicamente, o *clock* da CPU e da memória *flash* são suspensos.
- Redução de Ruído para o ADC: a CPU é parada, mas continuam operando o ADC, as interrupções externas, igualdade de endereço da interface serial à 2 fios, o temporizador/contador 2 e o *watchdog timer* (se habilitado). Esse modo é empregado para reduzir o ruído para o ADC e garantir sua resolução.

Gerenciamento de energia

- Power-down: o funcionamento do oscilador externo é suspenso, mas continuam operando a interface serial à 2 fios, as interrupções externas e o *watchdog timer* (se habilitado). Basicamente, interrompe todos os *clocks* gerados.
- Power-save: igual ao modo *power-down*, com exceção que o contador/temporizador 2 continua trabalhando assincronamente.
- Standby: é idêntico ao modo *power-down*, com exceção que o oscilador é mantido funcionando (válido para oscilador externo a cristal ou ressonador cerâmico). O microcontrolador ‘desperta’ do *sleep* em 6 ciclos de *clock*.
- Extended Standby: idêntico ao modo *power-save*, com exceção que o oscilador é mantido funcionando. Leva 6 ciclos de *clock* para o microcontrolador ‘despertar’.

Referências

- **AVR e Arduino – Técnicas de Projeto.**
 - Capítulo 2. O ATmega328

