Portas de Entrada e Saída

GPIO do ATmega328

Prof. Roberto de Matos

roberto.matos@ifsc.edu.br

INSTITUTO FEDERAL
Santa Catarina
Câmpus São José

Objetivo

Objetivo

- Entender como funcionam as portas de entrada e saída do AVR
- Entender como simular entradas e saídas na IDE

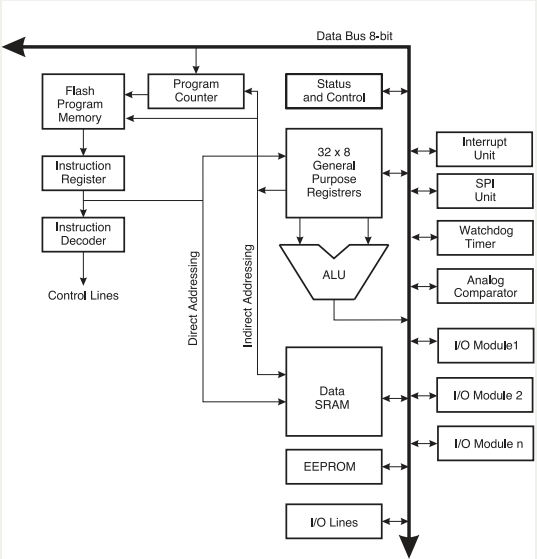
Leitura Recomendada:

AVR e Arduino: Técnicas de Projeto, 2ª ed. (Capítulo 5)[1]

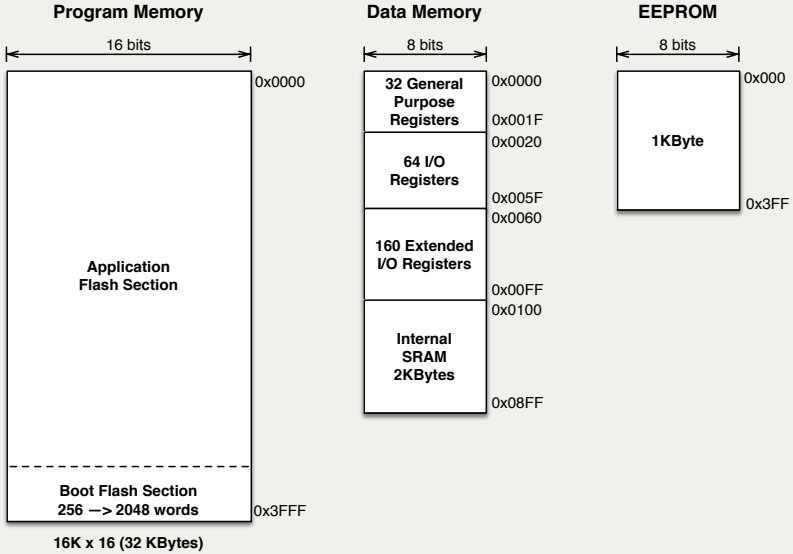
[1] Alguns exemplos e imagens dessa apresentação são retirados desse livro.

Relembrando ...

Núcleo do AVR



Memórias ATmega328p



Painel de controle

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x3F (0x5F)	SREG	I	T	H	S	V	N	Z	C
0x3E (0x5E)	SPH	–	–	–	–	–	(SP10)	SP9	SP8
0x3D (0x5D)	SPL	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0
0x37 (0x57)	SPMCSR	SPMIE	(RWWSB)	SIGRD	(RWWSRE)	BLBSET	PGWRT	PGERS	SPMEN
0x35 (0x55)	MCUCR	–	BODS	BODSE	PUD	–	–	IVSEL	IVCE
0x34 (0x54)	MCUSR	–	–	–	–	WDRF	BORF	EXTRF	PORF
0x33 (0x53)	SMCR	–	–	–	–	SM2	SM1	SM0	SE
0x30 (0x50)	ACSR	ACD	ACBG	ACO	ACI	ACIE	ACIC	ACIS1	ACIS0
0x2E (0x4E)	SPDR	SPI Data Register							
0x2D (0x4D)	SPSR	SPIF	WCOL	–	–	–	–	–	SPI2X
0x2C (0x4C)	SPCR	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0
0x2B (0x4B)	GPIOR2	General Purpose I/O Register 2							
0x2A (0x4A)	GPIOR1	General Purpose I/O Register 1							
0x28 (0x48)	OCR0B	Timer/Counter0 Output Compare Register B							
0x27 (0x47)	OCR0A	Timer/Counter0 Output Compare Register A							
0x26 (0x46)	TCNT0	Timer/Counter0 (8-bit)							
0x25 (0x45)	TCCR0B	FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00
0x24 (0x44)	TCCR0A	COM0A1	COM0A0	COM0B1	COM0B0	–	–	WGM01	WGM00
0x23 (0x43)	GTCCR	TSM	–	–	–	–	–	PSRASY	PSRSYNC
0x22 (0x42)	EEARH	(EEPROM Address Register High Byte)							
0x21 (0x41)	EEARL	EEPROM Address Register Low Byte							
0x20 (0x40)	EEDR	EEPROM Data Register							
0x1F (0x3F)	EECR	–	–	EEPM1	EEPM0	EERIE	EEMPE	EEPE	EERE
0x1E (0x3E)	GPIOR0	General Purpose I/O Register 0							
0x1D (0x3D)	EIMSK	–	–	–	–	–	–	INT1	INT0
0x1C (0x3C)	EIFR	–	–	–	–	–	–	INTF1	INTF0
0x1B (0x3B)	PCIFR	–	–	–	–	–	PCIF2	PCIF1	PCIF0
0x17 (0x37)	TIFR2	–	–	–	–	–	OCF2B	OCF2A	TOV2
0x16 (0x36)	TIFR1	–	–	ICF1	–	–	OCF1B	OCF1A	TOV1
0x15 (0x35)	TIFR0	–	–	–	–	–	OCF0B	OCF0A	TOV0
0x0B (0x2B)	PORTD	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0
0x0A (0x2A)	DDRD	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0
0x09 (0x29)	PIND	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0
0x08 (0x28)	PORTC	–	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0
0x07 (0x27)	DDRC	–	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0
0x06 (0x26)	PINC	–	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0
0x05 (0x25)	PORTB	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0
0x04 (0x24)	DDRB	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0
0x03 (0x23)	PINB	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0

Painel de controle

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x3F (0x5F)	SREG	I	T	H	S	V	N	Z	C
0x3E (0x5E)	SPH	–	–	–	–	–	(SP10)	SP9	SP8
0x3D (0x5D)	SPL	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0
0x37 (0x57)	SPMCSR	SPMIE	(RWWSB)	SIGRD	(RWWSRE)	BLBSET	PGWRT	PGERS	SPMEN
0x35 (0x55)	MCUCR	–	BODS	BODSE	PUD	–	–	IVSEL	IVCE
0x34 (0x54)	MCUSR	–	–	–	–	WDRF	BORF	EXTRF	PORF
0x33 (0x53)	SMCR	–	–	–	–	SM2	SM1	SM0	SE
0x30 (0x50)	ACSR	ACD	ACBG	ACO	ACI	ACIE	ACIC	ACIS1	ACIS0
0x2E (0x4E)	SPDR	SPI Data Register							
0x2D (0x4D)	SPSR	SPIF	WCOL	–	–	–	–	–	SPI2X
0x2C (0x4C)	SPCR	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0
0x2B (0x4B)	GPIOR2	General Purpose I/O Register 2							
0x2A (0x4A)	GPIOR1	General Purpose I/O Register 1							
0x28 (0x48)	OCR0B	Timer/Counter0 Output Compare Register B							
0x27 (0x47)	OCR0A	Timer/Counter0 Output Compare Register A							
0x26 (0x46)	TCNT0	Timer/Counter0 (8-bit)							
0x25 (0x45)	TCCR0B	FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00
0x24 (0x44)	TCCR0A	COM0A1	COM0A0	COM0B1	COM0B0	–	–	WGM01	WGM00
0x23 (0x43)	GTCCR	TSM	–	–	–	–	–	PSRASY	PSRSYNC
0x22 (0x42)	EEARH	(EEPROM Address Register High Byte)							
0x21 (0x41)	EEARL	EEPROM Address Register Low Byte							
0x20 (0x40)	EEDR	EEPROM Data Register							
0x1F (0x3F)	EECR	–	–	EEPM1	EEPM0	EERIE	EEMPE	EEPE	EERE
0x1E (0x3E)	GPIOR0	General Purpose I/O Register 0							
0x1D (0x3D)	EIMSK	–	–	–	–	–	–	INT1	INT0
0x1C (0x3C)	EIFR	–	–	–	–	–	–	INTF1	INTF0
0x1B (0x3B)	PCIFR	–	–	–	–	–	PCIF2	PCIF1	PCIF0
0x17 (0x37)	TIFR2	–	–	–	–	–	OCF2B	OCF2A	TOV2
0x16 (0x36)	TIFR1	–	–	ICF1	–	–	OCF1B	OCF1A	TOV1
0x15 (0x35)	TIFR0	–	–	–	–	–	OCF0B	OCF0A	TOV0
0x0B (0x2B)	PORTD	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0
0x0A (0x2A)	DDRD	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0
0x09 (0x29)	PIND	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0
0x08 (0x28)	PORTC	–	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0
0x07 (0x27)	DDRC	–	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0
0x06 (0x26)	PINC	–	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0
0x05 (0x25)	PORTB	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0
0x04 (0x24)	DDRB	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0
0x03 (0x23)	PINB	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0

Fluxograma

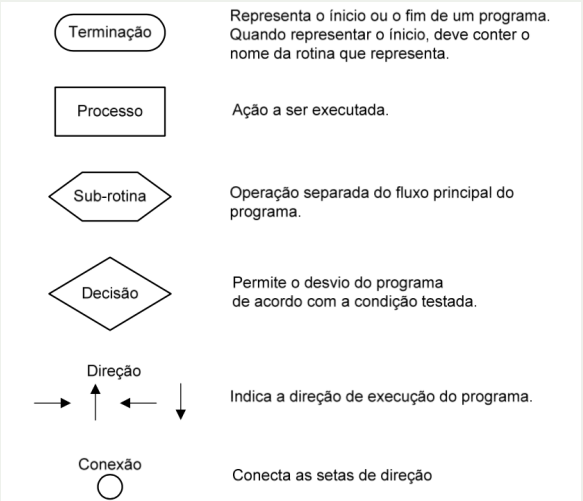


Fig. 4.1– Símbolos básicos para o desenho de fluxogramas.

Portas de Entrada e Saída

Entradas e Saídas

- Portas de entrada e saída de uso geral (GPIO) são os periféricos mais simples dos microcontroladores e a forma mais simples de interagir com sensores (entrada) e atuadores (saída).
- O ATmega328 possui 3 conjuntos de pinos de entrada e saída (I/Os): PORTB, PORTC e PORTD; respectivamente, pinos PB7 .. PB0, PC6 .. PC0 e PD7 .. PD0, todos com a função Lê - Modifica - Escreve.
- Cada pino pode ser configurado individualmente como entrada ou saída.
- Todas os pinos têm resistores de *pull-up* internamente e diodos de proteção entre o VCC e o terra, além de uma capacitância de 10 pF.

Estrutura Interna

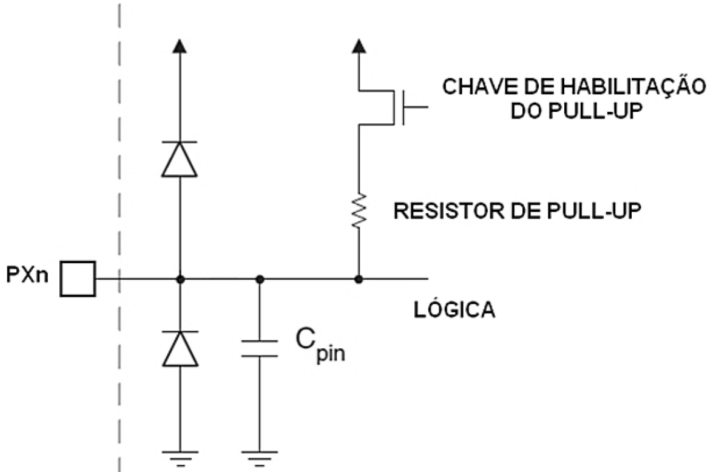


Fig. 5.1 – Esquema geral dos pinos de I/O (PXn) do ATmega.

Registradores: DDRx, PORTx, PINx

- **DDRx**: registrador de direção, usado para definir se os pinos de Px são entrada ou saída.
- **PORTx**: registrador de dados, usado para modificar os pinos de Px, quando configurados como saída, e habilitar o *pull-up*, quando configurados como entrada.
- **PINx**: registrador de entrada, usado para ler o estado dos pinos do Px.

Registradores: DDRx, PORTx, PINx

- **DDRx**: registrador de direção, usado para definir se os pinos de Px são entrada ou saída.
- **PORTx**: registrador de dados, usado para modificar os pinos de Px, quando configurados como saída, e habilitar o *pull-up*, quando configurados como entrada.
- **PINx**: registrador de entrada, usado para ler o estado dos pinos do Px.

- Tabela com os bits de controle:

DDXn	PORTXn	PUD	I/O	Pull-up	Comentário
0	0	x	Entrada	Não	Alta impedância (Hi-Z)
0	1	0	Entrada	Sim	PXn irá fornecer corrente se externamente for colocado em nível lógico 0.
0	1	1	Entrada	Não	Alta impedância (Hi-Z)
1	0	x	Saída	Não	Saída em zero (drena corrente)
1	1	x	Saída	Não	Saída em nível alto (fornece corrente)

Registradores: DDRx, PORTx, PINx

- Endereços dos registradores das portas:

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0B (0x2B)	PORTD	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0
0x0A (0x2A)	DDRD	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0
0x09 (0x29)	PIND	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0
0x08 (0x28)	PORTC	–	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0
0x07 (0x27)	DDRC	–	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0
0x06 (0x26)	PINC	–	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0
0x05 (0x25)	PORTB	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0
0x04 (0x24)	DDRB	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0
0x03 (0x23)	PINB	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0

Mais detalhes:

O registrador PINx é somente leitura, mas existe uma função especial acionada quando se escreve o valor lógico “1” no PINxn: o valor de PORTxn é invertido, independente do DDRxn. Ou seja, com apenas uma instrução `sbi` é possível inverter o pino de uma porta.

Manipulando um LED

Circuito, fluxograma e assembly

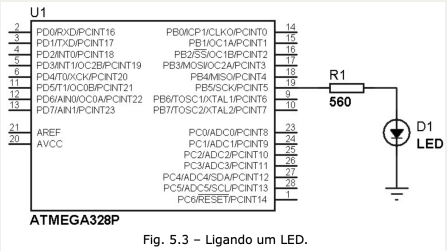
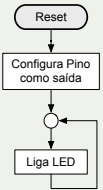


Fig. 5.3 – Ligando um LED.



Circuito, fluxograma e assembly

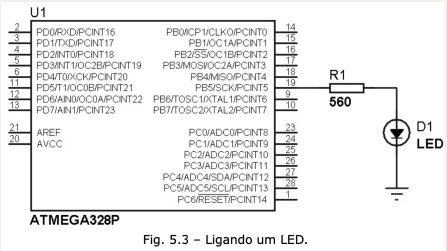
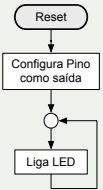


Fig. 5.3 – Ligando um LED.



- Ligar o LED:

```
1  .INCLUDE <m328Pdef.INC>
2
3  setup:
4    SBI DDRB, PB5  ;configura o pino PB5 como saída
5
6  loop:
7    SBI PORTB, PB5 ;coloca o pino PB5 em 5V
8
9    RJMP loop
```

Circuito, fluxograma e assembly

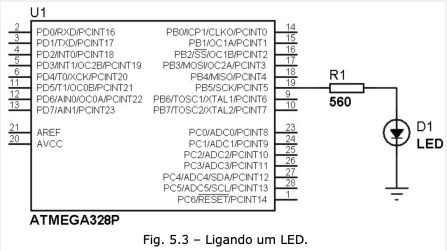
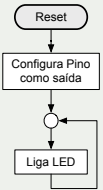


Fig. 5.3 – Ligando um LED.



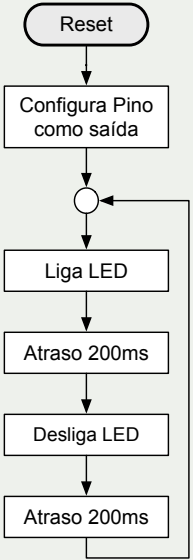
- Ligar o LED:

```
1  .INCLUDE <m328Pdef.INC>
2
3  setup:
4    SBI DDRB, PB5  ;configura o pino PB5 como saída
5
6  loop:
7    SBI PORTB, PB5 ;coloca o pino PB5 em 5V
8
9    RJMP loop
```

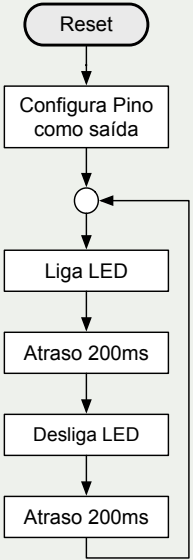
- Desligar o LED, substituir a linha 7 por:

```
7    CBI PORTB, PB5 ;coloca o pino PB5 em 0V
```

Exemplo (“pisca_led.asm”)

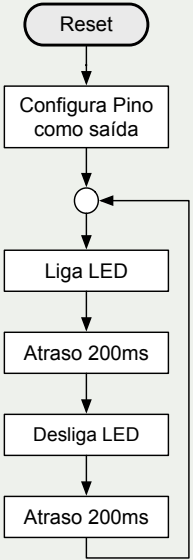


Exemplo (“pisca_led.asm”)




```
1  .INCLUDE <m328Pdef.INC>
2
3  start:
4    SBI DDRB, PB5  ;configura o pino PB5 como saída
5
6  main:
7    SBI PORTB, PB5 ;coloca o pino PB5 em 5V
8
9    ;atraso de aprox. 200ms
10   LDI R19, 16
11   RCALL delay
12
13   CBI PORTB, PB5 ;coloca o pino PB5 em 0V
14
15   ;atraso de aprox. 200ms
16   LDI R19, 16
17   RCALL delay
18
19   RJMP main
```

Exemplo (“pisca_led.asm”)

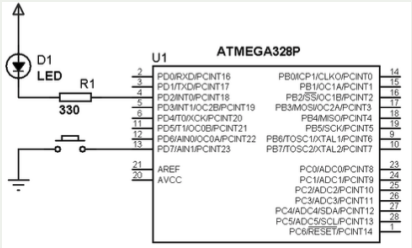



```
1  .INCLUDE <m328Pdef.INC>
2
3  start:
4    SBI DDRB, PB5  ;configura o pino PB5 como saída
5
6  main:
7    SBI PORTB, PB5 ;coloca o pino PB5 em 5V
8
9    ;atraso de aprox. 200ms
10   LDI R19, 16
11   RCALL delay
12
13   CBI PORTB, PB5 ;coloca o pino PB5 em 0V
14
15   ;atraso de aprox. 200ms
16   LDI R19, 16
17   RCALL delay
18
19   RJMP main
```

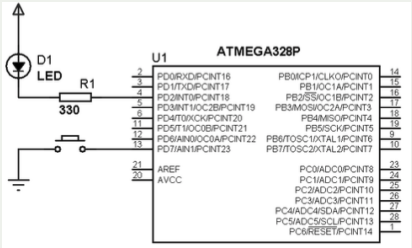
```
20   ;SUB-ROTINA DE ATRASO Programável
21   ; Ex.: - R19 = 16 --> 200ms
22   ;      - R19 = 80 --> 1s
23   ;--------------------------------
24   delay:
25     PUSH r17
26     PUSH r18
27     IN r17,SREG
28     PUSH r17
29
30     CLR r17
31     CLR r18
32   loop:
33     DEC  R17
34     BRNE loop
35     DEC  R18
36     BRNE loop
37     DEC  R19
38     BRNE loop
39
40     POP r17
41     OUT SREG, r17
42     POP r18
43     POP r17
44
45     RET
```

Lendo um Botão

Circuito

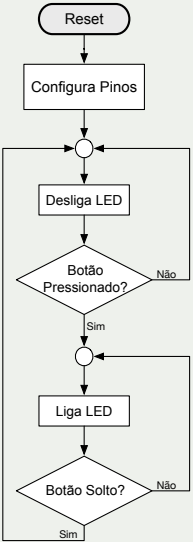

D1
LED
R1
330
U1
ATMEGA328P
PD0/RXD/PCINT16
PD1/TXD/PCINT17
PD2/INT0/PCINT18
PD3/INT1/OC2B/PCINT19
PD4/T0/XCK/PCINT20
PD5/T1/OC0B/PCINT21
PD6/AIN0/OC0A/PCINT22
PD7/AIN1/PCINT23
AREF
AVCC
PB0/ICP1/CLKO/PCINT0
PB1/OC1A/PCINT1
PB2/SS/OC1B/PCINT2
PB3/MOSI/OC2A/PCINT3
PB4/MISO/PCINT4
PB5/SCK/PCINT5
PB6/TOSC1/XTAL1/PCINT6
PB7/TOSC2/XTAL2/PCINT7
PC0/ADC0/PCINT8
PC1/ADC1/PCINT9
PC2/ADC2/PCINT10
PC3/ADC3/PCINT11
PC4/ADC4/SDA/PCINT12
PC5/ADC5/SCL/PCINT13
PC6/RESET/PCINT14

Circuito

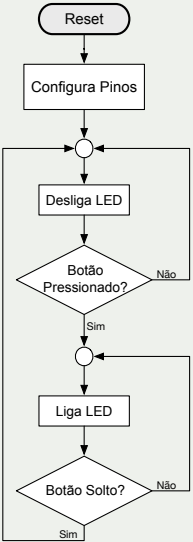


- LED acende colocando ‘0’ no PD2.
- Chave pressionada, PD7 = ‘0’.

Fluxograma e assembly



Fluxograma e assembly




```
1  .INCLUDE <m328Pdef.INC>
2
3  setup:
4    SBI DDRD, PD2        ;configura o pino PD2 como saída
5
6    CBI DDRD, PD7        ;configura o pino PD7 como entrada
7    SBI PORTD, PD7       ;habilita o pull-up para o botão
8
9  ;-----------------------
10 ;LAÇO PRINCIPAL
11 ;-----------------------
12 naoPress:               ;loop botão não pressionado (pull-up)
13   SBI  PORTD,PD2        ;desliga LED
14   SBIC PIND,PD7         ;verifica se o botão foi pressionado,
15   RJMP naoPress         ;senão volta e fica preso no laço naoPress
16
17 press:                  ;loop botão pressionado
18   CBI  PORTD,PD2        ;liga LED
19   SBIS PIND,PD7         ;verifica se o botão foi solto, senão
20   RJMP press            ;senão, aguarda.
21
22   RJMP naoPress         ;vai para o loop do "botão pressionado"
```


Experimentos

Experimentos no MPLAB IDE

- Simule e teste na placa todos os trechos de códigos apresentados.
- Faça a atividade proposta.

Fim!