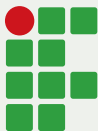


Projetando, codificando e simulando para MCU

Assembly do ATmega328

Prof. Roberto de Matos

roberto.matos@ifsc.edu.br



**INSTITUTO
FEDERAL**

Santa Catarina

Câmpus
São José

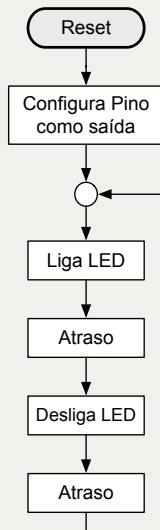
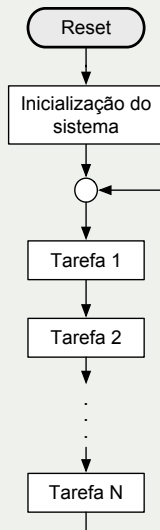
Objetivo

- Entender como projetar um *software* embarcado básico
- Entender como codificar um *software* em Assembly
- Introdução à simulação



***Software* embarcado básico**

- Conhecido como *Round-robin* ou “loopão”
- Simplicidade
- Funcionamento básico:
 - Configura o sistema logo depois do *reset*
 - Executa as tarefas de forma sequencial e repetida
- Exemplo: Pisca-pisca



Fluxograma

Fluxograma - Símbolos básicos

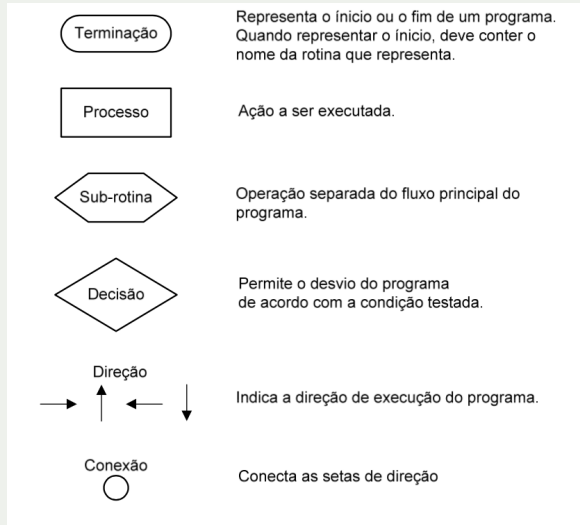
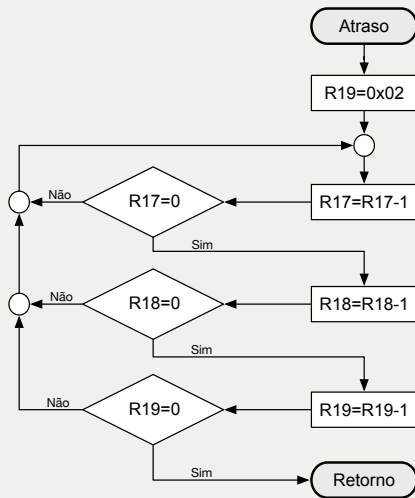
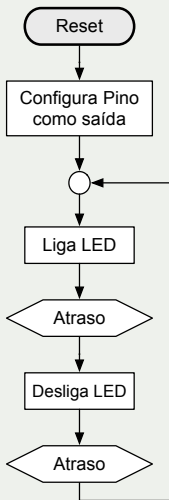


Fig. 4.1– Símbolos básicos para o desenho de fluxogramas.



Fluxograma - Exemplo



Programando em *Assembly*

- Planejar e modelar o código
- Conhecer a arquitetura (hierarquia de memória, periféricos, etc.)
- Conhecer as instruções *assembly*¹
- Conhecer as diretivas do montador² (*assembler*)

¹Manual do conjunto de instruções do AVR

²Manual online do montador



■ Sintaxe do arquivo:

```
1 [Rótulo:] .diretiva [operandos] [comentários]
2 [Rótulo:] instrução [operandos] [comentários]
3 Comentários
4 Linhas em branco
```

■ Entre [] é opcional

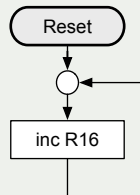
■ Diretivas → Montador

■ Instruções → CPU

■ Todo texto após ";" ou "//" é comentário

■ Exemplo:

```
1 label: .EQU var1=100 ; associa a constante 100 à var1 (Diretiva)
2       .EQU var2=200 ; associa a constante 200 à var2 (Diretiva)
3
4 start: INC r16       ; incrementa o r16 (Instrução)
5       RJMP start    ; salta para o rótulo ``start'' (Instrução)
6               ; linha com comentário
7               // outro comentário
```



- Definição do seguimento de dados ou código: **.DSEG, .CSEG, .ESEG**
- Reserva de bytes na memória SRAM (variável): **.BYTE**
- Reserva de bytes/words (constantes): **.DB, .DW**
- Definir um nome simbólico para um registrador: **.DEF**
- Setar um símbolo a partir de uma expressão: **.EQU, .SET**
- Setar um endereço de origem: **.ORG**
- Incluir um arquivo fonte: **.INCLUDE**

¹Verificar [lista completa de diretivas](#) no manual online do montador.



- Decimal: 10, 255
- Hexadecimal: 0x0a, \$0a, 0xff, \$ff
- Binário: 0b00110101, 0b1100_1010
- Octal (zero na frente): 010, 077



Ambiente de desenvolvimento e simulação

Abrindo o Projeto

- Descompactar o arquivo mic29004.zip em um local conhecido.
- Abra o MPLAB X IDE
- Clique em *File -> Open Project*
- Selecione o diretório descompactado
- Clique em *Open Project*
- No painel no canto superior esquerdo *Projects*, expanda o projeto e o diretório *Source Files*
- Abra o arquivo `aula1.asm`
- Organize as janelas para a simulação conforme sugestão



Sugestão de organização

MPLAB X IDE v5.45 - mic29004 : programming

3024/509.1MB PC: 0x0 ithsvnz

How do I? (Keywords) Q- Search (36 +)

mic29004

- Header Files
- Important Files
- Linker Files
- Source Files
- Libraries
- Loadables

asm1.asm

```
1 ; Replace with your application code
2
3 start:
4   inc r16
5   rjmp start
6
7
```

mic29004 - Dashboard

- Checksum: NA
- CRC32: 0x541EE624
- Packs
- ATmega_DFP (2.2.108)
- Compiler Toolchain
- AVRASM (v2.2.7) / Applications/microc
- Production Image: Optimization:
- Device support information: ATmega_f
- Memory
 - Data 2,048 (0x800) bytes
 - Data Used: 0 (0x0) Free: 2,048 (0x800)
 - Program 32,768 (0x8000) bytes
 - Program Used: 36 (0x24) Free: 32,732 (0x7FCC)
- Debug Tool
 - Simulator
 - Click for Simulated Peripherals
- Debug Resources
 - Program BP Used: 0 Free: 1000
 - Data BP Used: 0 Free: 1000

Output SRAM Data Memory

Address	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	ASCII
100	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	SRAM Memory
110	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
120	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
130	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
140	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
150	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
160	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Memory SRAM Data ... Format Hex

I/O Memory (SRs)

Address	Name	Hex	Decimal	Bin
000	PC	0x0000	0	0000
001	R0	0x0000	0	0000
002	R1	0x0000	0	0000
003	R2	0x0000	0	0000
004	R3	0x0000	0	0000
005	R4	0x0000	0	0000
006	R5	0x0000	0	0000
007	R6	0x0000	0	0000
008	R7	0x0000	0	0000
009	R8	0x0000	0	0000
00A	R9	0x0000	0	0000
00B	R10	0x0000	0	0000
00C	R11	0x0000	0	0000
00D	R12	0x0000	0	0000
00E	R13	0x0000	0	0000
00F	R14	0x0000	0	0000
010	R15	0x0000	0	0000
011	R16	0x0000	0	0000
012	R17	0x0000	0	0000
013	R18	0x0000	0	0000
014	R19	0x0000	0	0000
015	R20	0x0000	0	0000
016	R21	0x0000	0	0000
017	R22	0x0000	0	0000
018	R23	0x0000	0	0000
019	R24	0x0000	0	0000
01A	R25	0x0000	0	0000
01B	R26	0x0000	0	0000
01C	R27	0x0000	0	0000
01D	R28	0x0000	0	0000
01E	R29	0x0000	0	0000
01F	R30	0x0000	0	0000
020	R31	0x0000	0	0000
021	PINB	0x0000	0	0000
022	DDRB	0x0000	0	0000
023	PORTB	0x0000	0	0000
024	PINC	0x0000	0	0000
025	DDRC	0x0000	0	0000
026	PORTC	0x0000	0	0000
027	DDRD	0x0000	0	0000
028	PORTD	0x0000	0	0000
029	PORTD	0x0000	0	0000
02A	PORTD	0x0000	0	0000
02B	PORTD	0x0000	0	0000
02C	PORTD	0x0000	0	0000
02D	PORTD	0x0000	0	0000
02E	PORTD	0x0000	0	0000
02F	PORTD	0x0000	0	0000
030	PORTD	0x0000	0	0000
031	PORTD	0x0000	0	0000
032	PORTD	0x0000	0	0000
033	PORTD	0x0000	0	0000
034	PORTD	0x0000	0	0000
035	PORTD	0x0000	0	0000
036	PORTD	0x0000	0	0000
037	PORTD	0x0000	0	0000
038	PORTD	0x0000	0	0000
039	PORTD	0x0000	0	0000
03A	PORTD	0x0000	0	0000
03B	PORTD	0x0000	0	0000
03C	PORTD	0x0000	0	0000
03D	PORTD	0x0000	0	0000
03E	PORTD	0x0000	0	0000
03F	PORTD	0x0000	0	0000
040	PORTD	0x0000	0	0000

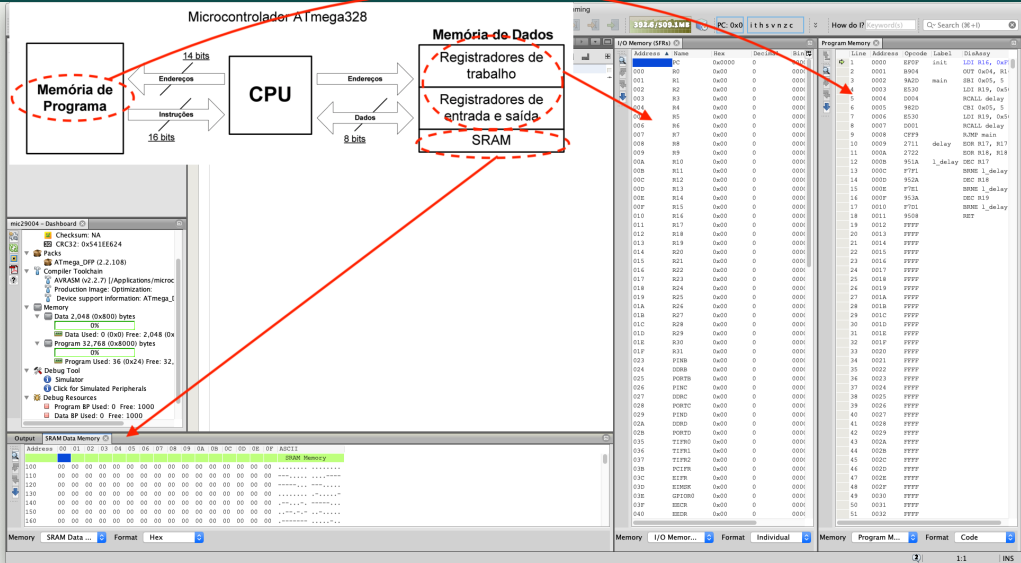
Program Memory

Line	Address	Opcodes	Label	Disassembly
1	0000	RFPD	init	LDI R16, 0x0F
2	0001	R904		OUT 0x04, R1
3	0002	R920	main	SRI 0x05, 5
4	0003	R530		LDI R19, 0x51
5	0004	D004		RCALL delay
6	0005	R820		CHI 0x05, 5
7	0006	R530		RCALL delay
8	0007	D001		RCALL delay
9	0008	CF7F		RJMP main
10	0009	2711	delay	SOR R17, R17
11	000A	2722		SOR R18, R18
12	000B	951A	l_delay	DEC R17
13	000C	F7F1		RNME l_delay
14	000D	952A		DEC R18
15	000E	F7E1		RNME l_delay
16	000F	953A		DEC R19
17	0010	9701		RNME l_delay
18	0011	9508		RET
19	0012	FFFF		
20	0013	FFFF		
21	0014	FFFF		
22	0015	FFFF		
23	0016	FFFF		
24	0017	FFFF		
25	0018	FFFF		
26	0019	FFFF		
27	001A	FFFF		
28	001B	FFFF		
29	001C	FFFF		
30	001D	FFFF		
31	001E	FFFF		
32	001F	FFFF		
33	0020	FFFF		
34	0021	FFFF		
35	0022	FFFF		
36	0023	FFFF		
37	0024	FFFF		
38	0025	FFFF		
39	0026	FFFF		
40	0027	FFFF		
41	0028	FFFF		
42	0029	FFFF		
43	002A	FFFF		
44	002B	FFFF		
45	002C	FFFF		
46	002D	FFFF		
47	002E	FFFF		
48	002F	FFFF		
49	0030	FFFF		
50	0031	FFFF		
51	0032	FFFF		

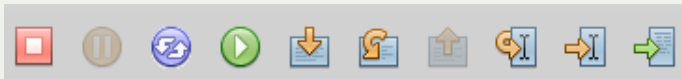
Memory Program M... Format Code

1:1 INS

Arquitetura vs. simulador



- Monte o programa:
 - *Production -> Build Main Project* (atalhos: F11 ou ícone do “martelo”).
- Inicie a simulação:
 - *Debug -> Discrete Debugger Operation -> Launch Debugger Main Project*
- Use os controles do debugger para a simular o código:



Fim!