Biblioteca Teclas para o Arduino

Timer 1 interrompe com taxa de 100 Hz (10 mseg).

Toda medida é feita com duas leituras do ADC, espaçadas de 10 mseg.

Atualiza tensão do carro (vcar\_val), na taxa de 100/32 Hz (~3Hz).

Atualiza tensão do super cap (vcap\_val), na taxa de 100/32 Hz (~3Hz).

Leitura das teclas na taxa de 100\*(12/32) = 37,5 Hz, uma leitura a cada 27 ms.

|  |  |  |
| --- | --- | --- |
|  | SW\_CIMA |  |
| SW\_ESQ | SW\_SEL | SW\_DIR |
|  | SW\_BAIXO |  |

SW\_SEQ1: SW\_ESQ, SW\_CIMA, SW\_DIR

SW\_SEQ2: SW\_ESQ, SW\_BAIXO, SW\_DIR

Funções para o Teclado

|  |  |  |
| --- | --- | --- |
| void | sw\_qq\_tecla | (void) |
| void | sw\_fila\_limpa | (void) |
| void | sw\_ler | (byte val) |
| void | sw\_busca\_seq1 | (byte val) |
| void | sw\_busca\_seq2 | (byte val) |
| byte | sw\_qual | (byte val) |
| void | sw\_config | (void) |
| byte | sw\_poe | (byte cha) |
| byte | sw\_tira | (byte \*cha) |
| void | sw\_teste | (void) |
| void | sw\_adc | (void) |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

* void **sw\_qq\_tecla** (void)

Espera pelo usuário acionar uma tecla qualquer. Essa tecla é ignorada

* void **sw\_fila\_limpa** (void)

Limpa a fila do teclado. Joga tudo fora!

* void **sw\_ler** (byte val)

Analisa código “val” com o passado das chaves e se for o caso coloca o código da nova tecla na fila circular. Ver fluxograma adiante.

* void **sw\_busca\_seq1** (byte val)

Monitora as teclas, buscando pela Sequência 1, usa a variável sw\_st\_seq1 (ESQ, CIMA, DIR).

* void **sw\_busca\_seq2** (byte val)

Monitora as teclas, buscando pela Sequência 2, usa a variável sw\_st\_seq2 (ESQ, BAIXO, DIR).

* byte **sw\_qual** (byte val)

Retorna o código correspondente à leitura “val” do ADC. Melhorar com o novo teclado.

* void **sw\_config** (void)

Configurar para a leitura das chaves e das tensões do carro (VCAR) e super cap (VCAP).

* byte **sw\_poe** (byte cha)

Colocar na fila o código corresponde a uma chave.

TRUE se conseguiu colocar na fila e

FALSE se a fila está cheia e então escreve mensagem no LCD.

* byte **sw\_tira** (byte \*cha)

Retirar o código de uma tecla da fila do teclado.

TRUE se conseguiu retirar e coloca o código na posição apontada pelo ponteiro “cha” e

FALSE se a fila está vazia.

* void **sw\_teste** (void)

Imprime no LCD e na serial o nome da tecla acionada. ESQUERDA, CIMA, etc.

Só sai com reset

* void **sw\_adc** (void)

Mostrar no LCD e na serial o valor retornado pelo ADC para cada tecla.

Só sai com reset. Desliga a interrupção do Timer 1.

*Gabaritos para configurar ADC (8 bits alinhado pela esquerda, ler apenas ADCH)*

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| **ADMUX** | REFS1 | REFS0 | ADLAR | MUX4 | MUX3 | MUX2 | MUX1 | MUX0 |
| 0,1,1 | 1,1,1 | 1 | 0 | 0 | 0 | 0,0,1 | 0,1,0 |
| **ADCSRA** | ADEN | ADSC | ADATE | ADIF | ADIE | ADPS2 | ADPS1 | ADPS0 |
| 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 |
| **ADCSRB** | - | ACME | - | - | MUX5 | ADST2 | ADST1 | ADST0 |
| - | 0 | - | - | 0 | 1 | 0 | 1 |

Configuração para os diversos canais

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Analógico | Canal ADC | MUX5:0 | REFS1:0 | REF volts |
| Teclado | 0 | 0 0 0 0 0 | 0 1 | AVCC |
| Tensão VCAR | 1 | 0 0 0 0 1 | 1 1 | 2,56 V |
| Tensão VCAP | 2 | 0 0 0 1 0 | 1 1 | 2,56 V |

Teclado: ADMUX = (1<<REFS0) | (1<<ADLAR); //Ref = AVCC

VCAR: ADMUX = (1<<REFS1) | (1<<REFS0) | (1<<ADLAR) | (1<<MUX0); //Canal 1, Ref = 2,56

VCAP: ADMUX = (1<<REFS1) | (1<<REFS0) | (1<<ADLAR) | (1<<MUX1); //Canal 2, Ref = 2,56

Funcionou bem: timer1\_cont = 0, 1, ..., 31, 0, 1, ..., interrupção em 100 Hz

|  |  |  |  |
| --- | --- | --- | --- |
| 0-ADC Start | 8-+Ler, ADC start\* | 16-ADC Start | 24-+Ler, ADC start\* |
| 1-+Ler, ADC start | 9-+Ler, ADC start | 17-+Ler, ADC start | 25-+Ler, ADC start |
| 2-+Ler, ADC start\* | 10-+Ler, ADC start\* | 18-+Ler, ADC start\* | 26-+Ler, ADC start\* |
| 3-+Ler, ADC start | 11-+Ler, ADC start | 19-+Ler, ADC start | 27-+Ler, ADC start |
| 4-+Ler, ADC start\* | 12-Ler, Canal1(VCAR)\* | 20-+Ler, ADC start\* | 28-Ler, Canal 2(VCAP)\* |
| 5-+Ler, ADC start | 13-ADC Start | 21-+Ler, ADC start | 29-ADC Start |
| 6-+Ler, ADC start\* | 14- Ler, ADC start | 22-+Ler, ADC start\* | 30-Ler, ADC start |
| 7-+Ler, ADC start | 15-Ler, Canal 0 | 23-+Ler, ADC start | 31-Ler, Canal 0 |

\* indica a fase para tirar a média na leitura do teclado.

Defines usados no teclado

#define SW\_FILA\_TAM 10 // tamanho da fila do teclado

#define SW\_TOL 20 // tolerancia para identificar chave

// (8 bits) Valores lidos pelo conversor AD0

#define SW\_VAL\_DIR 0 //antigo 0

#define SW\_VAL\_CIMA 33 //antigo 33

#define SW\_VAL\_BAIXO 80 //antigo 80

#define SW\_VAL\_ESQ 121 //antigo 121

#define SW\_VAL\_SEL 186 //antigo 186

#define SW\_VAL\_NADA 255 //antigo 255

// Codigos para as chaves

#define SW\_NADA 0

#define SW\_CIMA 1

#define SW\_ESQ 2

#define SW\_SEL 3

#define SW\_DIR 4

#define SW\_BAIXO 5

#define SW\_SEQ1 6

#define SW\_SEQ7 7

#define SW\_NAOSEI 99

Variáveis globais para o teclado

volatile char sw\_fila[SER\_TX\_FILA\_TAM]; //Espaço para a fila teclado

volatile byte sw\_pin, sw\_pout; //Ponteiros para usar a fila

volatile byte sw\_1,sw\_2,sw\_n,sw\_v; //Variáveis para detectar teclas acionadas

Lógica para detectar teclas acionadas

sw\_1 e sw\_2 🡪 garantir duas leituras iguais, evita transitórios;

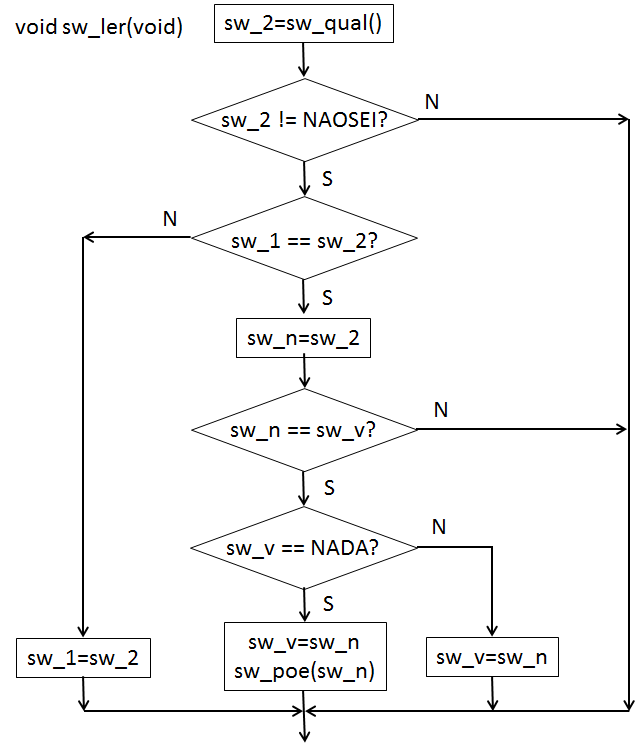
sw\_n 🡪 nova tecla

sw\_v 🡪 tecla velha

Tecla é considerada válida quando duas leituras seguidas são iguais: sw\_1=sw\_2

Neste caso sw\_n recebe o código da tecla.

Se sw\_n<>sw\_v e sw\_v = NADA, nova tecla é colocada no buffer e depois sw\_v=sw\_n.



|  |  |
| --- | --- |
| (a) Circuito | (b) Numérica |

Vi = tensão no adc ao acionar a chave SWi.

🡪

🡪

🡪

|  |  |  |
| --- | --- | --- |
|  | Calculados | Comerciais |
| R1 | 10.000,00 | 10.000 |
| R2 | 2.307,69 | 2.400 |
| R3 | 2.237,76 | 2.200 |
| R4 | 3.232,32 | 3.300 |
| R5 | 5.079,37 | 5.100 |
| R6 | 9.142,86 | 9.100 |
| R7 | 21.333,33 | 22.000 |
|  |  |  |

Uma imagem contendo captura de tela

Descrição gerada automaticamente

% Valores comerciais

R7 = 22000; %21333.33

R6 = 9100; % 9142.86

R5 = 5100; % 5079.37

R4 = 3300; % 3232.32

R3 = 2200; % 2237.76

R2 = 2400; % 2307.69

R1 = 10000; %10000

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
|  | R | V | Hexa |  | R | V | Hexa |  | 3 Bits |
| 1 | 10.000,00 | 0,0 | 0 |  | 10.000 | 0,0 | 00 |  | 000 |
| 2 | 2.307,69 | 0,9375 | 30 |  | 2.400 | 0,9677 | 32 |  | 001 |
| 3 | 2.237,76 | 1,5625 | 50 |  | 2.200 | 1,5753 | 51 |  | 010 |
| 4 | 3.232,32 | 2,1875 | 70 |  | 3.300 | 2,2067 | 71 |  | 011 |
| 5 | 5.079,37 | 2,8125 | 90 |  | 5.100 | 2,8261 | 91 |  | 100 |
| 6 | 9.142,86 | 3,4375 | B0 |  | 9.100 | 3,4424 | B0 |  | 101 |
| 7 | 21.333,33 | 4,0625 | D0 |  | 22.000 | 4,0758 | D1 |  | 110 |
|  |  |  |  |  |  |  |  |  |  |