UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE ELETRÔNICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

Leonardo Winter Pereira
Rodrigo Yudi Endo

Projeto 02 - Lombada Eletrônica

CURITIBA

2017

Introdução

Enquanto um motorista dirige em ruas na cidade ou em rodovias, a velocidade máxima determinada para seu veículo não deve ser ultrapassada. A violação dos limites de velocidade pode gerar graves acidentes e deve ser punida. Para controlar isso, existem os radares e lombadas eletrônicas, que medem a velocidade do veículo em um determinado trecho e multam o condutor caso ele ultrapasse o limite estabelecido. Esse projeto visa implementar uma solução simples para o problema, utilizando um sensor de distância ultrassónico.

Objetivos

Projetar um protótipo de lombada eletrônica, utilizando um microcontrolador da família 8051, contendo os seguintes requisitos:

- A velocidade deverá ser mostrada em dois displays de sete segmentos.
- O sistema deverá ter um alarme em forma de LED e buzzer quando a velocidade ultrapassar os 40 km/h.
- O sistema utilizara um sensor de distância por ultrassom HC-SR04.

A proposta foi modificada pelo professor Rubens, para utilizar o sensor de distância no lugar das bobinas indutoras.

Esquemático do projeto

O esquemático utilizado para a montagem do projeto pode ser visto na figura 1 abaixo:

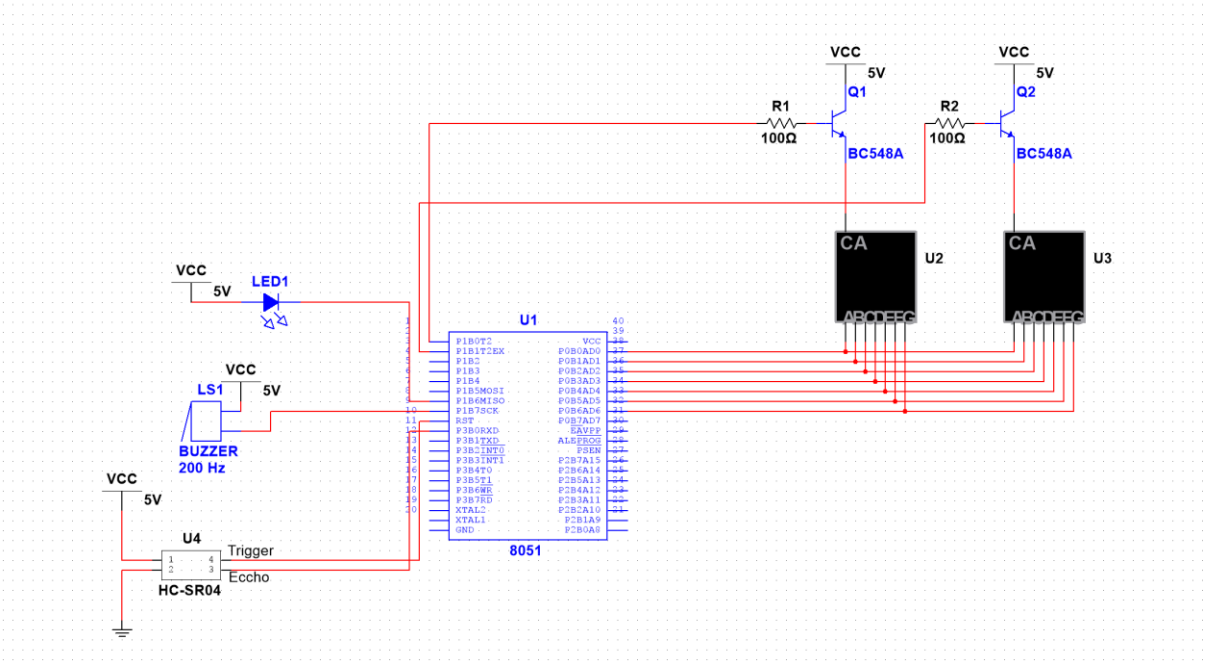

Figura *1 – Esquemático*

Métodos e funções implementadas

Timers e interrupções

Inicialmente, as interrupções foram setadas. A interrupção externa INT0 foi utilizada para o acionamento de um botão que determina o início da leitura do sensor. O TIMER/COUNTER 1 foi utilizado para contar o tempo nas horas necessárias e reaproveitado para enviar pulso para o pino trigger do HC-SR04, sendo resetado para as condições iniciais após ser utilizado pelo sensor.

Display de Sete Segmentos

Os segmentos de A a H foram conectados no port P0 do microcontrolador e os displays foram ligados em cascata. Os ports P1.0 e P1.1 foram utilizados para escolher em qual display o dado seria escrito. Dessa forma, se o P1.0 fosse setado, o digito da unidade seria escrito e, se o P1.1 fosse setado, a dezena. Para escrever o digito correto no display, uma tabela que relaciona o valor hexadecimal mandado para o port com a combinação de segmentos que devem ser ativados para formar o caractere desejado no display [3].

Sensor de distância HC-SR04

O modulo de distância HC-SR04 funciona emitindo um sinal de ultrassom, recebendo o sinal ecoado e calculando o tempo entre esses dois eventos. O sinal de

saída é criado como uma forma de onda cujo tempo em alta é proporcional a distância. Um pulso com 10µs de tempo em alta deve ser transmitido para o pino do trigger e a saída pode ser vista no pino do echo [2]. Para calcular a distância em centímetros, a seguinte formula foi utilizada:

> Distancia em cm = Largura do pulso no echo em µS/58

Buzzer

Para o acionamento do buzzer, inicialmente o port P1.7 foi setado como o pino do buzzer. Para acionar o componente, o mesmo foi ligado a uma fonte de 5V e o pino correspondente foi setado. Para aciona-lo nos momentos necessários, o port recebeu uma instrução CLR e, após um período de tempo desejado, SETB.

Calculo da velocidade

O sensor que mede a distância é acionado por uma interrupção externa ativada por um botão. O HC-SR04 é então lido novamente depois de um segundo. O cálculo de (distância final – distância inicial) é feito e temos, com isso, a velocidade em cm/s. Essa velocidade então é comparada com uma velocidade máxima pré-determinada e, caso seja maior, um buzzer e um LED são acionados.

Conclusão

Nesse experimento foi possível aprender mais sobre um novo sensor, o medidor de distância HC-SR04. Enquanto a parte de código foi simples, reutilizando várias coisas aprendidas na sala de aula e nas práticas anteriores, como timers e interrupções, o sensor apresentou problemas. A precisão dele é muito baixa devido à grande área que ele analisa, tornando difícil ter uma leitura precisa. A medida só foi confiável movimentando o sensor em direção a uma parede, sem nenhum obstáculo no caminho. Apesar dessa forma de apresentação fugir da proposta de um radar, cujo sensor deveria ficar fixo na via, os erros aconteceram devido a imprecisão do sensor. Esse fato pode ser corrigido adquirindo um componente mais preciso, porém mais caro, sendo inviável para o projeto.

Referências

1. NICOLASSI, Denys E. C. Microcontrolador 8051 Detalhado. São Paulo: Editora Erica, 6ª edição.
2. HC-SR04 ultrasonic sensor interfacing with 8051 microcontroller. Disponível em: < http://homemaderobo.blogspot.com.br/2012/08/ultrasonic-sensor-interfacing-with-8051.html >. Acesso em 5 de maio de 2017.
3. Interfacing Seven segment display to 8051. Disponível em: < http://www.circuitstoday.com/interfacing-seven-segment-display-to-8051 >. Acesso em 5 de maio de 2017.

Anexo – Código implementado

```
1   ////////////////////////////////////////////////////////////////////////////
2   //                                                                        //
3   //                  PROJETO 02 - LOMBADA ELETRONICA                       //
4   //                                                                        //
5   // Requisitos:                                                            //
6   // - mostre a velocidade em dois displays de sete segmentos vermelhos     //
7   // - A medida de velocidade e estruturada a partir de um sensor           //
8   // ultrassonico sr04                                                      //
9   // - Esse sensor calcula a distancia entre ele e o obstaculo, permitindo  //
10  // assim calcular a velocidade                                            //
11  // - O sistema devera, ainda, ter um alarme em forma de led e buzzer quando //
12  // a velocidade ultrapassar os XX cm / s.                                 //
13  //                                                                        //
14  // CONSIDERE QUE 1 ESTADO = 0.375 us                                      //
15  //                                                                        //
16  // @author: Leonardo Winter Pereira                                       //
17  // @author: Rodrigo Yudi Endo                                             //
18  //                                                                        //
19  //  http://www.circuitstoday.com/ultrasonic-range-finder-using-8051       //
20  //
    https://www.google.com.br/search?q=display+de+7+segmentos&source=lnms&tbm=isch&sa=X&ved=0ahUKEwjp35vrotTT
    AhXHAsAKHRujBmEQ_AUICigB&biw=1920&bih=988#imgrc=5R8audCeFus40M:
21  ////////////////////////////////////////////////////////////////////////////
22
23  org 000h // Origem do codigo
24  ljmp __STARTUP__
25
26  org 003h // Inicio do codigo da interrupcao externa INT0
27  ljmp INT_EXT0
28
29  org 00Bh // Inicio do codigo da interrupcao interna gerada pelo TIMER/COUNTER 0
30  ljmp INT_TIMER0
31
32  org 013h // Inicio do codigo da interrupcao externa INT1
33  ljmp INT_EXT1
34
35  org 01Bh // Inicio do codigo da interrupcao interna gerada pelo TIMER/COUNTER 1
36  ljmp INT_TIMER1
37
38  org 023h // Inicio do codigo da interrupcao SERIAL
39  ljmp INT_SERIAL
40
41  ////////////////////////////////////////////////
42  //       TABELA DE EQUATES DO PROGRAMA        //
43  ////////////////////////////////////////////////
44
45  PORT_DISPLAY                EQU P0
46
47  DISPLAY_UNIDADE             EQU P1.0
48  DISPLAY_DEZENA              EQU P1.1
49
50  PINO_LED_VERMELHO           EQU P1.6
51
52  BUZZER                      EQU P1.7
53
54  TRIGGER_ULTRASSOM_RECEPTOR  EQU P3.0
55  TRIGGER_ULTRASSOM_ENVIO     EQU P3.1
56
57  // LEDS DA PLACA
58  LED_SEG                     EQU P3.6
59  LED1                        EQU P3.7
60
61  FLAG_MEDIR_DISTANCIA_INIT   EQU 37h
62  FLAG_CALCULAR_VELOCIDADE    EQU 38h  // Se essa flag esta em 1 -> calcula a velocidade e mostra no
    display de 7 segmentos
63
64  // Velocidade em cm/s
65  VELOCIDADE_VEICULO_UNIDADE  EQU 39h
66  VELOCIDADE_VEICULO_DEZENA   EQU 40h
67  VELOCIDADE_VEICULO          EQU 41h
68
69  VELOCIDADE_LIMITE           EQU 42h // em cm/s
70
```

```
  71  TIMER_MEDICOES_LOW          EQU 43h
  72
  73  // Registradores disponibilizados para medir tempo com variacao de 1 ms
  74  // Com esses 2 registradores (configurados na rotina INT_TIMER1) e possivel registrar um tempo de
      medicao de ate 65 s
  75  TIMER_MEDICOES_LSB          EQU 44h
  76  TIMER_MEDICOES_MSB          EQU 45h
  77
  78  DISTANCIA_ANTERIOR          EQU 46h
  79  DISTANCIA_ATUAL             EQU 47h
  80
  81  ////////////////////////////////////////////////
  82  // REGIAO DA MEMORIA DE PROGRAMA COM AS STRINGS //
  83  ////////////////////////////////////////////////
  84
  85  org 030h
  86  TAB7SEG:
  87      DB 3FH, 06H, 5BH, 4FH, 66H, 6DH, 7DH, 07H, 7FH, 6FH, 77H, 7CH, 39H, 5EH, 79H, 71H
  88
  89  ////////////////////////////////////////////////
  90  //              INICIO DO PROGRAMA              //
  91  ////////////////////////////////////////////////
  92
  93  __STARTUP__:
  94          LCALL   SETA_VARIAVEIS_INICIAIS
  95          LCALL   INT_CONFIGURA_INTERRUPCOES
  96
  97  ESPERA_TRIGGER_CALCULAR_DISTANCIA_INIT:
  98          MOV     A, FLAG_MEDIR_DISTANCIA_INIT
  99          CJNE    A, #01h, ESPERA_TRIGGER_CALCULAR_DISTANCIA_INIT
 100
 101          MOV     R1, #DISTANCIA_ANTERIOR
 102          LCALL   MEDIR_DISTANCIA
 103
 104          // Necessario reprogramar os timers e as interrupcoes devido ao calculo da distancia pelo sensor
 105          LCALL   TIMER_CONFIGURA_TIMER
 106          LCALL   INT_CONFIGURA_INTERRUPCOES
 107          LCALL   RESETA_TIMER_MEDICOES
 108
 109          SETB    TR1
 110
 111  ESPERA_FLAG_MEDIR_DISTANCIA_2:
 112          MOV     A, FLAG_CALCULAR_VELOCIDADE
 113          CJNE    A, #01h, ESPERA_FLAG_MEDIR_DISTANCIA_2
 114
 115          MOV     R1, #DISTANCIA_ATUAL
 116          LCALL   MEDIR_DISTANCIA
 117
 118          LCALL   CALCULA_VELOCIDADE
 119
 120          AJMP    __STARTUP__
 121
 122  ////////////////////////////////////////////////////
 123  // NOME: SETA_VARIAVEIS_INICIAIS                  //
 124  // DESCRICAO:                                     //
 125  // P.ENTRADA:                                     //
 126  // P.SAIDA: -                                     //
 127  // ALTERA:                                        //
 128  ////////////////////////////////////////////////////
 129  SETA_VARIAVEIS_INICIAIS:
 130          SETB    PINO_LED_VERMELHO
 131          SETB    BUZZER
 132
 133          MOV     VELOCIDADE_LIMITE,          #020d   // velocidade limite de 20 cm/s
 134          MOV     FLAG_MEDIR_DISTANCIA_INIT,  #00h
 135          MOV     FLAG_CALCULAR_VELOCIDADE,   #00h
 136          MOV     VELOCIDADE_VEICULO_UNIDADE, #00h
 137          MOV     VELOCIDADE_VEICULO_DEZENA,  #00h
 138          MOV     DISTANCIA_ANTERIOR,         #00h
 139          MOV     DISTANCIA_ATUAL,            #00h
 140
 141          RET
 142
```

```
143   ////////////////////////////////////////////////
144   // NOME: RESETA_TIMER_MEDICOES                //
145   // DESCRICAO:                                 //
146   // P.ENTRADA:                                 //
147   // P.SAIDA: -                                 //
148   // ALTERA:                                    //
149   ////////////////////////////////////////////////
150   RESETA_TIMER_MEDICOES:
151           MOV     TIMER_MEDICOES_LOW,     #00h
152           MOV     TIMER_MEDICOES_LSB,     #00h
153           MOV     TIMER_MEDICOES_MSB,     #00h
154
155           MOV     VELOCIDADE_VEICULO,     #00h
156
157           RET
158
159   ////////////////////////////////////////////////
160   // NOME: MEDIR_DISTANCIA                      //
161   // DESCRICAO:                                 //
162   // P.ENTRADA: R1 -> ponteiro para endereco de dist. //
163   // P.SAIDA:                                   //
164   // ALTERA: R1, A                              //
165   ////////////////////////////////////////////////
166   MEDIR_DISTANCIA:
167           CLR     DISPLAY_UNIDADE
168           CLR     DISPLAY_DEZENA
169
170           MOV     PORT_DISPLAY, #00h
171
172           CLR     TRIGGER_ULTRASSOM_RECEPTOR  // P3.0 configurado para enviar sinal de trigger
173           SETB    TRIGGER_ULTRASSOM_ENVIO     // P3.1 configurado para receber sinal de trigger
174
175           MOV     TMOD, #00100000B    // seta timer 1 para o modo 02
176           MOV     TL1, #130D
177           MOV     TH1, #130D
178
179           MOV     A, #00h
180
181           SETB    TRIGGER_ULTRASSOM_RECEPTOR  // inicia o pulso para o trigger
182
183           ACALL   TIMER_DELAY_10_US     // o trigger precisa de um pulso de 10 us para funcionar
      corretamente
184
185           CLR     TRIGGER_ULTRASSOM_RECEPTOR
186
187   ESPERANDO_RESPOSTA_ECHO:
188           JNB     TRIGGER_ULTRASSOM_ENVIO, $    // aguarda o recebimento do trigger (sinal de eco)
189
190   ECHO_AINDA_DISPONIVEL:
191           SETB    TR1
192
193           JNB     TF1, $
194
195           CLR     TR1
196           CLR     TF1
197
198           INC     A   // o acumulador ira representar a distancia adquirida pelo sensor de ultrassom
199
200           JB      TRIGGER_ULTRASSOM_ENVIO, ECHO_AINDA_DISPONIVEL  // continua no loop enquanto ainda nao
      tiver recebido um echo como resposta
201
202           MOV     @R1, A  // armazena o valor (referente a distancia) no endereco apontado por R1
203
204           RET
205
206   ////////////////////////////////////////////////
207   // NOME: CALCULA_VELOCIDADE                   //
208   // DESCRICAO:                                 //
209   // P.ENTRADA:                                 //
210   // P.SAIDA:                                   //
211   // ALTERA: C, B, R6, R5                       //
212   ////////////////////////////////////////////////
213   CALCULA_VELOCIDADE:
```

```
214           CLR     C // para poder fazer a comparacao para ver se a velocidade e maior ou menor do que o
      limite permitido
215
216           MOV     A, DISTANCIA_ANTERIOR
217           MOV     B, DISTANCIA_ATUAL
218
219           SUBB    A, B
220
221           JNC     DISTANCIA_ANTERIOR_MAIOR_QUE_ATUAL
222
223   DISTANCIA_ATUAL_MAIOR_QUE_ANTERIOR:
224           MOV     A, DISTANCIA_ATUAL
225           SUBB    A, DISTANCIA_ANTERIOR
226
227           MOV     VELOCIDADE_VEICULO, A
228
229           JMP     DLOOP
230
231   DISTANCIA_ANTERIOR_MAIOR_QUE_ATUAL:
232           MOV     A, DISTANCIA_ANTERIOR
233           SUBB    A, DISTANCIA_ATUAL
234
235           MOV     VELOCIDADE_VEICULO, A
236
237   DLOOP:
238          // Soma a velocidade medida do veiculo com (0xFF - VELOCIDADE_LIMITE)
239          // Se essa soma setar o Carry, e porque a velocidade esta acima do limite
240          // Caso contrario, velocidade abaixo do limite
241          MOV     A, #0FFh
242          SUBB    A, VELOCIDADE_LIMITE
243
244          ADDC    A, VELOCIDADE_VEICULO
245
246          JC      ACIMA_DO_LIMITE
247
248   CONTINUA_DLOOP:
249           MOV     R5,#100d
250           MOV     R6,#15d
251   BACK1:
252           MOV     A, VELOCIDADE_VEICULO
253           MOV     B, #10d     // para poder calcular a velocidade em dezena
254
255           DIV     AB          // armazena em A o valor da unidade
256
257           SETB    DISPLAY_UNIDADE       // ativa o display referente a unidade
258           ACALL   MOSTRA_VELOCIDADE_DISPLAY   // mostra digito no display
259
260           ACALL   TIMER_DELAY_1_MS
261
262           MOV     A,B         // move para o acumulador o restante da divisao anterior
263           CLR     DISPLAY_UNIDADE        // desativa o display referente a unidade
264           SETB    DISPLAY_DEZENA         // ativa o display referente a dezena
265           ACALL   MOSTRA_VELOCIDADE_DISPLAY // mostra digito no display
266
267           ACALL   TIMER_DELAY_1_MS
268
269           CLR     DISPLAY_DEZENA       // desativa o display da dezena
270
271           // De acordo com os valores configurados em R6 e R5, mostra por mais ou menos tempo a
      velocidade no display
272           DJNZ    R6, BACK1
273           MOV     R6, #15h
274           DJNZ    R5, BACK1
275
276           // desativa (em baixa) o led vermelho e o buzzer
277           SETB    PINO_LED_VERMELHO
278           SETB    BUZZER
279
280           RET
281
282   ACIMA_DO_LIMITE:
283           LCALL   VELOCIDADE_ACIMA_DO_LIMITE
284
```

```
285           JMP     CONTINUA_DLOOP
286
287           RET
288
289   ////////////////////////////////////////////////
290   // NOME: VELOCIDADE_ACIMA_DO_LIMITE            //
291   // DESCRICAO: ATIVA O LED VEMELHO E O BUZZER   //
292   // P.ENTRADA:                                  //
293   // P.SAIDA:                                    //
294   // ALTERA:                                     //
295   ////////////////////////////////////////////////
296   VELOCIDADE_ACIMA_DO_LIMITE:
297           CLR     PINO_LED_VERMELHO
298           CLR     BUZZER
299
300           RET
301
302   ////////////////////////////////////////////////
303   // NOME: MOSTRA_VELOCIDADE_DISPLAY             //
304   // DESCRICAO:                                  //
305   // P.ENTRADA: A                                //
306   // P.SAIDA:                                    //
307   // ALTERA: A                                   //
308   ////////////////////////////////////////////////
309   MOSTRA_VELOCIDADE_DISPLAY:
310           MOV     DPTR, #TAB7SEG  // Move para o DPTR o endereco dos valores para o display de 7 segmentos
311
312           MOVC    A, @A + DPTR    // busca pelo valor referente ao digito a ser impresso
313           CPL     A               // complementa o digito a ser impresso (necessario para que o display
      seja usado corretamente)
314           MOV     PORT_DISPLAY, A // Envia para o port do display
315
316           RET
317
318   ////////////////////////////////////////////////
319   //        CODIGOS RELACIONADOS AO TIMER       //
320   ////////////////////////////////////////////////
321
322   ////////////////////////////////////////////////
323   // NOME: TIMER_CONFIGURA_TIMER                 //
324   // DESCRICAO:                                  //
325   // P.ENTRADA:                                  //
326   // P.SAIDA:                                    //
327   // ALTERA:                                     //
328   ////////////////////////////////////////////////
329   TIMER_CONFIGURA_TIMER:
330           MOV     TMOD, #00100010b // Seta o TIMER_0 para o modo 02 (08 bits com reset) e o TIMER_1 para
      o modo 02 (8 bits com reset)
331
332           ACALL   TIMER_SETA_VALORES_TIMER_PADRAO
333
334           ////////////////////////////////////////////////////
335           //    Aqui configuramos o TIMER_1 (para as medicoes)  //
336           //       Interrupcao a ser chamada a cada 50 us       //
337           // 50 us = 133 instrucoes (onde 1 executa em 0.375us) //
338           ////////////////////////////////////////////////////
339
340           MOV     TH1, #0FFh
341           MOV     TL1, #122d
342
343           RET
344
345   ////////////////////////////////////////////////
346   // NOME: TIMER_SETA_VALORES_TIMER_PADRAO       //
347   // DESCRICAO:                                  //
348   // P.ENTRADA:                                  //
349   // P.SAIDA:                                    //
350   // ALTERA:                                     //
351   ////////////////////////////////////////////////
352   TIMER_SETA_VALORES_TIMER_PADRAO:
353           // Para o TIMER_0, TH0 e TL0 representam o necessario para um delay de 20ms
354           // Se 1 estado executa em 0.375 us, precisamos de 53330 estados para executar 20ms
355           MOV     TH0, #207d
```

```
356           MOV     TL0, #207d
357
358           RET
359
360   ////////////////////////////////////////////////
361   // NOME: TIMER_DELAY_20_MS                    //
362   // DESCRICAO: INTRODUZ UM ATRASO DE 20 MS     //
363   // P.ENTRADA: R0 => (R0 x 20) ms              //
364   // P.SAIDA: -                                 //
365   // ALTERA: R0                                 //
366   ////////////////////////////////////////////////
367   TIMER_DELAY_20_MS:
368           MOV     TMOD, #00000001b // Seta o TIMER_0 para o modo 01 (16 bits) e o TIMER_1 para o modo 02
      (8 bits com reset)
369
370           // Para o TIMER_0, TH0 e TL0 representam o necessario para um delay de 20ms
371           MOV     TH0, #HIGH(65535 - 43350)
372           MOV     TL0, #LOW(65535 - 43350)
373
374           CLR     TF0
375           SETB    TR0
376
377           JNB     TF0, $
378
379           CLR     TF0
380           CLR     TR0
381
382           DJNZ R0, TIMER_DELAY_20_MS
383
384           RET
385
386   ////////////////////////////////////////////////
387   // NOME: TIMER_DELAY_1_S                      //
388   // DESCRICAO: INTRODUZ UM ATRASO DE 1 S       //
389   // P.ENTRADA: R1 = y => (y x 1) s             //
390   // P.SAIDA: -                                 //
391   // ALTERA: R1                                 //
392   ////////////////////////////////////////////////
393   TIMER_DELAY_1_S:
394           MOV     R0, #50d
395           CALL    TIMER_DELAY_20_MS
396
397           DJNZ    R1, TIMER_DELAY_1_S
398
399           RET
400
401   ////////////////////////////////////////////////
402   // NOME: TIMER_DELAY_10_US                    //
403   // DESCRICAO: INTRODUZ UM ATRASO DE 10 US     //
404   // P.ENTRADA:                                 //
405   // P.SAIDA: -                                 //
406   // ALTERA: R6                                 //
407   ////////////////////////////////////////////////
408   TIMER_DELAY_10_US:
409           MOV     R6, #2D
410           DJNZ    R6, $
411
412           RET
413
414   ////////////////////////////////////////////////
415   // NOME: TIMER_DELAY_1_MS                     //
416   // DESCRICAO: INTRODUZ UM ATRASO DE 1 MS      //
417   // P.ENTRADA:                                 //
418   // P.SAIDA: -                                 //
419   // ALTERA: R7                                 //
420   ////////////////////////////////////////////////
421   TIMER_DELAY_1_MS:
422           MOV     R7, #250d
423           DJNZ    R7, $
424
425           RET
426
427   ////////////////////////////////////////////////
```

```
428   //  INICIO DOS CODIGOS GERADOS POR INTERRUPCAO  //
429   ////////////////////////////////////////////////
430
431   ////////////////////////////////////////////////
432   // NOME: INT_CONFIGURA_INTERRUPCOES            //
433   // DESCRICAO: CONFIGURA AS PALAVRAS IE, IP E PARTE  //
434   //            DO TCON                          //
435   // P.ENTRADA:                                  //
436   // P.SAIDA:                                    //
437   // ALTERA:                                     //
438   ////////////////////////////////////////////////
439   INT_CONFIGURA_INTERRUPCOES:
440           // Bits da palavra IE - Interrupt Enable
441           SETB    EA
442           SETB    EX0
443           SETB    ET1
444
445           // Bits da palavra IP - Interrupt Priority
446           CLR     PX0     // Baixa prioridade para o SENSOR_EXTERNO_1
447           SETB    PT1     // Alta prioridade para o TIMER/COUNTER 1
448
449           // Bits da palavra TCON - Timer Control
450           CLR     IE0     // Interrupcao por Borda
451           CLR     IT1     // Interrupcao por Nivel
452
453           RET
454
455   ////////////////////////////////////////////////
456   // NOME: INT_EXT0                              //
457   // DESCRICAO:                                  //
458   // P.ENTRADA:                                  //
459   // P.SAIDA:                                    //
460   // ALTERA:                                     //
461   ////////////////////////////////////////////////
462   INT_EXT0:
463           PUSH    ACC
464           PUSH    PSW
465
466           MOV     FLAG_MEDIR_DISTANCIA_INIT, #01h
467
468           MOV     R0, #05h        // R0 x 20 ms de delay - para nao sentir o efeito de bounce no trigger
      da "pistola"
469           ACALL   TIMER_DELAY_20_MS
470
471           POP     PSW
472           POP     ACC
473
474           RETI
475
476   ////////////////////////////////////////////////
477   // NOME: INT_TIMER0                            //
478   // DESCRICAO:                                  //
479   // P.ENTRADA:                                  //
480   // P.SAIDA:                                    //
481   // ALTERA:                                     //
482   ////////////////////////////////////////////////
483   INT_TIMER0:
484           RETI
485
486   ////////////////////////////////////////////////
487   // NOME: INT_EXT1                              //
488   // DESCRICAO:                                  //
489   // P.ENTRADA:                                  //
490   // P.SAIDA:                                    //
491   // ALTERA: A                                   //
492   ////////////////////////////////////////////////
493   INT_EXT1:
494           RETI
495
496   ////////////////////////////////////////////////
497   // NOME: INT_TIMER1                            //
498   // DESCRICAO: ESTA ROTINA SERA CHAMADA A CADA 200US //
499   // TODA VEZ QUE A INT_EXT0 FOR ACIONADA. PERMITE    //
```

```
500   // CALCULAR O TEMPO GASTO PELO VEICULO PARA SAIR DO //
501   // PRIMEIRO SENSOR E ALCANCAR O SEGUNDO            //
502   // P.ENTRADA: -                                    //
503   // P.SAIDA: TIMER_MEDICOES_LOW; TIMER_MEDICOES_LSB  //
504   // ALTERA: TIMER_MEDICOES_LOW; TIMER_MEDICOES_LSB   //
505   ////////////////////////////////////////////////////
506   INT_TIMER1:
507           PUSH    ACC
508           PUSH    PSW
509
510           MOV     TL1, #122d
511           CLR     TF1
512
513           MOV     A, TIMER_MEDICOES_LOW
514           CJNE    A, #020d, INCREMENTA_TIMER_MEDICOES_LOW      // neste ponto, o TIMER_MEDICAO_LSB passa
      a contar 1 ms em cada novo estado
515
516           MOV     A, TIMER_MEDICOES_LSB
517           CJNE    A, #0250d, INCREMENTA_TIMER_MEDICOES_LSB    // neste ponto, o TIMER_MEDICAO_MSB passa a
      contar 250 ms em cada novo estado
518
519           MOV     TIMER_MEDICOES_LOW, #00h
520           MOV     TIMER_MEDICOES_LSB, #00h
521           INC     TIMER_MEDICOES_MSB
522
523           MOV     A, TIMER_MEDICOES_MSB
524           CJNE    A, #02d, FINALIZA_TIMER_1   // neste ponto, o TIMER_MEDICAO_MSB passa a contar 250 ms
      em cada novo estado
525
526           CLR     TR1
527           MOV     FLAG_CALCULAR_VELOCIDADE, #01h
528           AJMP    FINALIZA_TIMER_1
529
530   INCREMENTA_TIMER_MEDICOES_LOW:
531           INC     TIMER_MEDICOES_LOW
532           AJMP    FINALIZA_TIMER_1
533
534   INCREMENTA_TIMER_MEDICOES_LSB:
535           MOV     TIMER_MEDICOES_LOW, #00h
536           INC     TIMER_MEDICOES_LSB
537
538   FINALIZA_TIMER_1:
539           POP     PSW
540           POP     ACC
541
542           RETI
543
544   ////////////////////////////////////////////////////
545   // NOME: INT_SERIAL                                //
546   // DESCRICAO:                                      //
547   // P.ENTRADA:                                      //
548   // P.SAIDA:                                        //
549   // ALTERA:                                         //
550   ////////////////////////////////////////////////////
551   INT_SERIAL:
552           RETI
553
554   FIM:
555           JMP $
556           END
```