UNIVERSIDADE da MADEIRA

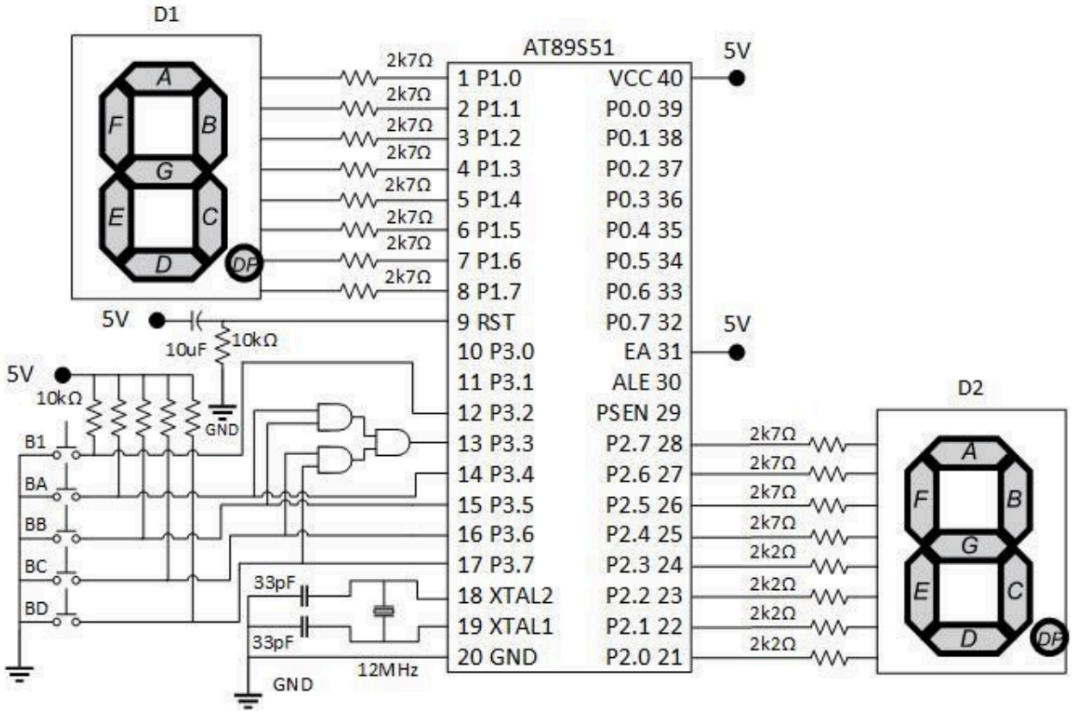
FCEE - LEI

ARQUITETURA DE COMPUTADORES

Sofia Inácio | Dionísio Barros | Pedro Camacho | Dino Vasconcelos

2023/2024

TRABALHO PRÁTICO 3



Trabalho realizado por:

Ana Leonor Freitas - nº2081821

Diogo Miguel Paixão - nº2079921

20 de maio de 2024

Índice

Introdução

No terceiro trabalho prático de Arquitetura de computadores, foi-nos proposto o desenvolvimento e implementação de um programa em linguagem Assembly e C para o microcontrolador 8051, com o intuito de efetuar o registo de tempo e de respostas, para uma plataforma de concursos de perguntas de escolha múltipla. O nosso projeto foi desenvolvido e testado utilizando a plataforma Keil uVision5.

Será apresentado o desenvolvimento do projeto, juntamente com discussões de resultados e fluxogramas em respetivo anexo.

Por fim, este projeto visa aprimorar os conteúdos trabalhados nas aulas práticas acerca da linguagem Assembly, C e o microcontrolador 8051.

Objetivos

O programa desenvolvido tem como objetivo criar um programa em linguagem Assembly e C para o microcontrolador 8051, onde oferece uma solução eficiente, intuitiva e funcional para registar o tempo e as respostas, para uma plataforma de concursos de perguntas de escolha múltipla.

Para alcançar esse objetivo, o projeto é dividido em várias etapas detalhadas, conforme descrito abaixo:

- Estudo das linguagens para o microcontrolador 8051
- Estudo da configuração e programação de interrupções do microcontrolador
- Programação em linguagem assembly e C
- Simulação na aplicação Keil uVision
- Testes e discussão de resultados
- Fluxogramas

A linguagem Assembly tem como principal objetivo realizar um controlo direto do hardware e programar instruções específicas para o microcontrolador. A linguagem C facilita o desenvolvimento e trata das funções mais complexas, oferecendo uma camada de abstração.

O software Keil uVision5 possibilita o bom desenvolvimento da programação e a testagem do nosso projeto, devido aos seus enormes recursos.

Desenvolvimento

Descrição da solução

Linguagem C

A configuração inicial do sistema define várias constantes e variáveis globais, como o tempo inicial de 5 segundos e o valor necessário para contar um segundo utilizando o timer do microcontrolador. Os pinos do microcontrolador são configurados para controlar os botões e os segmentos dos displays de sete segmentos utilizados para mostrar o tempo e a resposta.

A função ***Init*** configura o sistema, ativando as interrupções globais, o ***Timer0*** e as interrupções externas 0 e 1. O ***Timer0*** é configurado no modo 2, que é um modo de 8 bits com auto-reload, adequado para a contagem de tempo necessária no projeto. O ***Timer0*** não começa imediatamente; ele é ativado apenas quando o botão B1 é pressionado.

Duas interrupções externas são configuradas: ***External0*** e ***External1***. A ***External0*** trata o botão de início B1. Quando B1 é pressionado, o sistema alterna entre iniciar e reiniciar o contador de 5 segundos. Se o contador já estiver ativo e B1 for pressionado novamente, o contador é reiniciado para 5 segundos. Se não estiver ativo, o contador começa a decrescer. A ***External1*** trata os botões de resposta (BA, BB, BC, BD). Quando um desses botões é pressionado durante a contagem o pino dessa interrupção ativa (P3.3) e esta registra qual botão foi pressionado e em que momento, interrompendo a contagem.

A interrupção do ***Timer0*** (Timer0_ISR) é chamada a cada 250 microsegundos, incrementando uma variável de contagem. Quando essa variável atinge o valor necessário para 0.1 segundo, o contador de segundos é decrementado. Isso permite que o tempo mostrado no display seja atualizado com precisão de décimos de segundo.

Funções auxiliares são utilizadas para atualizar os displays de sete segmentos. A função ***display*** controla quais segmentos devem ser acesos para mostrar números específicos. A função ***displaySegundos*** converte o número de segundos restantes em dezenas e unidades, e chama ***display*** para mostrar esses valores nos displays.

Se o tempo chega a zero sem uma resposta, a função ***semResposta*** é chamada, mostrando "0.0 segundos" e, depois de um segundo, a indicação de resposta indefinida "-.-". Isso é feito através de um loop que altera entre segundos e resposta indefinida até que o botão B1 seja pressionado novamente.

Caso um botão de resposta seja pressionado antes de o tempo acabar, a função ***mostraInformacao*** é chamada, alternando entre mostrar o tempo restante que ficou ao clicar no botão de resposta e a resposta selecionada a cada segundo. Este ciclo continua até que o botão B1 seja pressionado novamente, momento em que o sistema é reiniciado para uma nova tentativa.

Linguagem Assembly

O programa começa definindo diversas constantes e variáveis importantes. O tempo inicial é configurado para 50 unidades (5 segundos), e o sistema usa contadores para medir 0,1 segundo (20 interrupções) e 1 segundo (200 interrupções). Os valores do ***Timer0*** são ajustados para gerar interrupções a cada 250 microsegundos, facilitando a contagem precisa do tempo. As portas P1 e P2 são utilizadas para os displays de sete segmentos, e P3 para os botões.

Na função principal, o sistema é inicializado através da chamada da função ***Init***, que configura os registos e as interrupções. Em seguida, o código entra em um loop infinito que atualiza continuamente os displays com o tempo restante e verifica se uma resposta foi registrada. Este loop é interrompido apenas quando certas condições específicas são atendidas, como o tempo se esgotar ou um botão de resposta ser pressionado.

O código lida com três tipos de interrupções: a interrupção externa 0 é acionada quando o botão B1 é pressionado. Se o botão não foi previamente pressionado, o timer começa a contar o tempo; caso contrário, o timer é reiniciado e o tempo é resetado para o valor inicial. A interrupção do ***Timer0*** é chamada a cada 250 microsegundos para incrementar contadores que rastreiam o tempo. Quando 1 segundo se passa, o contador de segundos é decrementado, a menos que uma resposta tenha sido registrada, garantindo a contagem precisa do tempo. A interrupção externa 1 é acionada por botões de resposta, verificando qual botão foi pressionado e registrando a resposta correspondente, interrompendo o contador de tempo.

Para exibir informações nos displays de sete segmentos, o código utiliza várias funções auxiliares. A função ***display*** configura os displays de sete segmentos com os valores adequados para mostrar os números, utilizando uma tabela de segmentos. A função ***displaySegundos*** divide o tempo restante em dezenas e unidades, ajusta os valores para exibição e chama display para mostrar esses valores nos displays.

A função ***semResposta*** é chamada quando o tempo se esgota sem uma resposta registrada, exibindo alternadamente "0.0" seguido por uma indicação de ausência de resposta "-.-" até que o botão de início seja pressionado novamente.

A função ***mostraInformacao*** alterna a exibição entre o tempo restante e a resposta registrada, até que o botão de início seja pressionado novamente, sinalizando para reiniciar o sistema.

Após a inicialização, o sistema entra em um loop contínuo, verificando e atualizando os displays com o tempo restante. As interrupções garantem que o sistema responda rapidamente a eventos como pressionamentos de botões e a passagem do tempo. O uso de funções auxiliares para exibição facilita a leitura das informações pelo usuário.

Em resumo, este código Assembly implementa de maneira eficiente um sistema de temporização e resposta utilizando interrupções para gerenciar eventos em tempo real e displays de sete segmentos para fornecer uma interface clara ao utilizador. A estrutura modular do código facilita a manutenção e a expansão futura, tornando-o adequado para aplicações educacionais e de prototipagem rápida.

Discussão de resultados

Depois da implementação e teste do sistema, tanto a parte desenvolvida em C quanto em Assembly funcionaram conforme o planeado.

Os botões responderam corretamente às interações, permitindo o registro preciso das respostas dos utilizadores. Além disso, os displays de sete segmentos mostraram as informações de forma clara e correta, exibindo os números e símbolos conforme o esperado.

A temporização e contagem de tempo também operaram de forma precisa, sem desvios perceptíveis, indicando que o sistema é robusto e confiável para o propósito pretendido.

Em resumo, todos os componentes funcionam, atendendo às expectativas iniciais do projeto.

Conclusão

Concluindo, o projeto alcançou seus objetivos de forma satisfatória. O código foi desenvolvido de maneira clara e livre de erros, demonstrando a nossa compreensão e aplicação dos conhecimentos adquiridos, especialmente na linguagem Assembly, C e tudo o que foi abordado nas aulas acerca do microcontrolador.

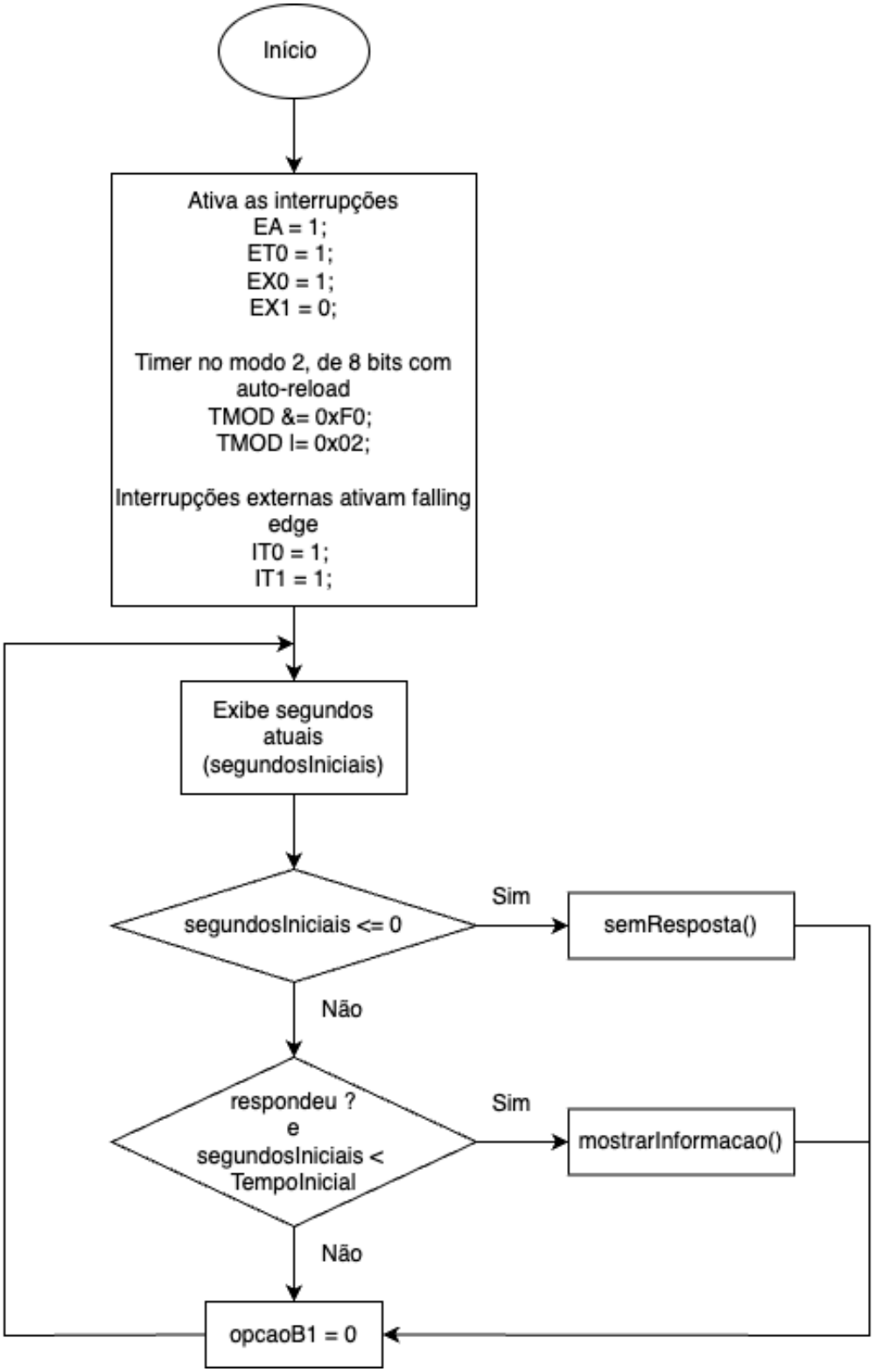
Esta experiência foi valiosa para consolidar os nossos entendimentos teóricos e desenvolver habilidades práticas relevantes, preparando-nos para desafios futuros na área de arquitetura de computadores.

Bibliografia

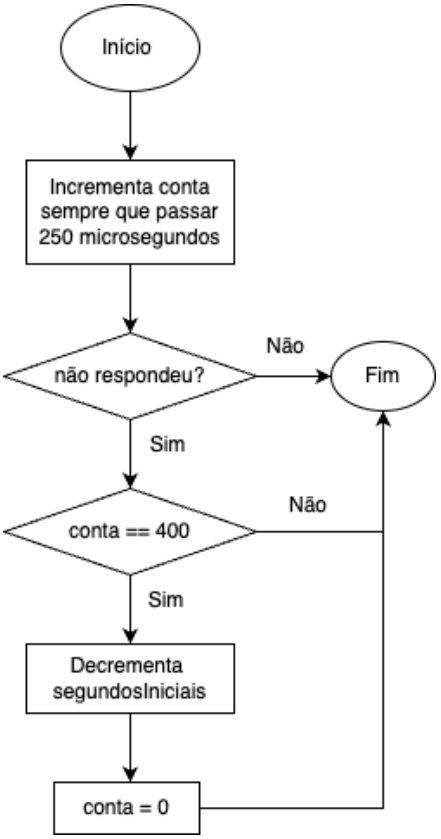
Materiais de referência utilizados incluem as práticas laboratoriais, seus exercícios, e slides fornecidos durante o seu curso.

Anexo A - Fluxogramas

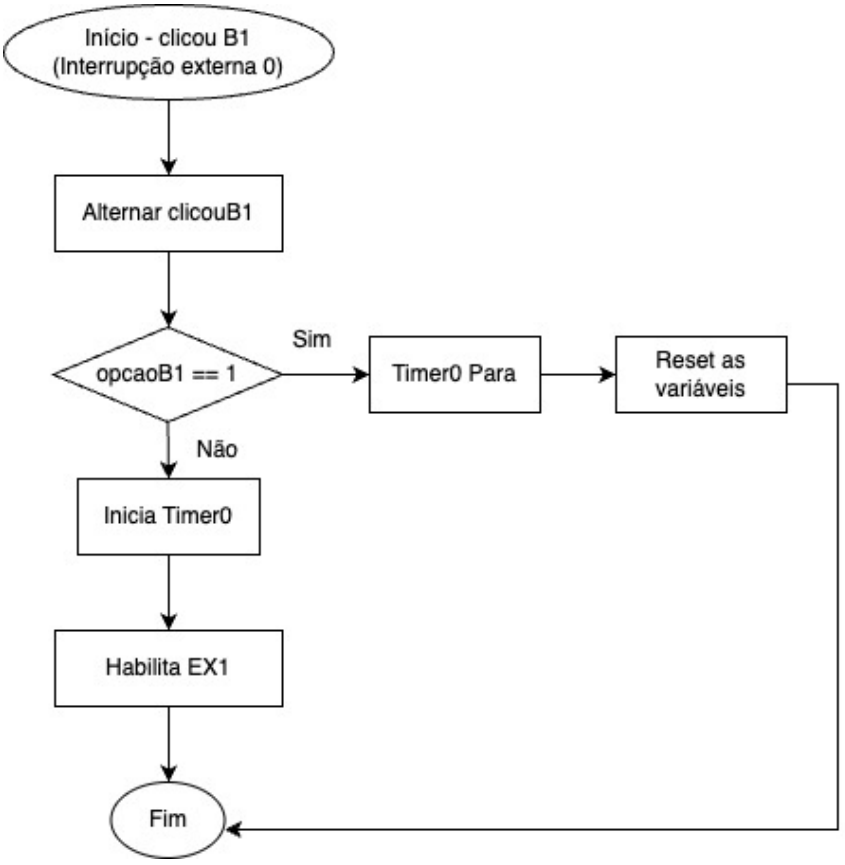
Programa principal:



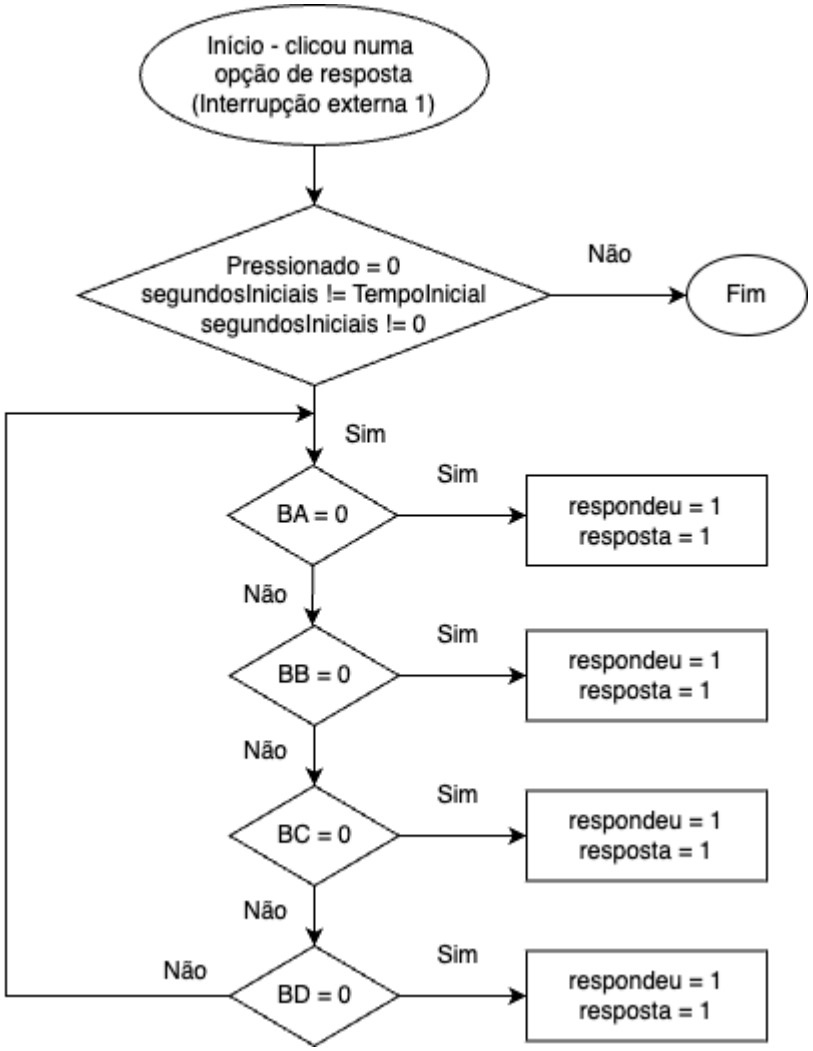
Interrupção Timer0 (Timer0_ISR):



Interrupção Externa 0 (External0):



Interrupção Externa 1 (External1):



Anexo B - Código

Código em linguagem C

```
#include <reg51.h>

// Tempo inicial de 5.0 segundos
#define TempoInicial 50
// Valor da variavel 'conta' para 1 segundo
#define segundo 4000

/*
O valor maximo da contagem de tempo e "FF + 1" = 256 microsegundos (Timer no modo 2 tem 8 bits)
Um cilo maquina tem 6 estados e cada estado tem 2 periodos do oscilador, logo 12 periodos
Periodo = 1/12*(10^6) = 1/12 microsegundos , 1 ciclo maquina = 12*Periodo = 12*(1/12) = 1 microsegundo
250 microsegundos : 256-250 = 6 = 6 microsegundos (Sendo 06 -> TH0 e 06 -> TL0)
*/
#define TempoH 0x06
#define TempoL 0x06

/*
Da inicio a contagem decrescente para o participante responder (Caso o tempo para a contagem esteja a ser mostrado nos displays com o tempo inicial de 5.0 segundos)
Repoe a visualizacao do tempo inicial de 5.0 segundos nos displays (Caso a informacao com o tempo/resposta do participante esteja a ser mostrada nos displays)
*/
sbit B1 = P3^2;

/*
Ao ser pressionado qualquer um dos botoes de opcao de resposta ha uma transicao logica de 1 para 0
Enquanto qualquer um dos botoes de opcao de resposta continuar pressionado, e colocado o valor logico 0
*/
sbit Pressionado = P3^3;

// Opcoes de resposta
sbit BA = P3^4;
sbit BB = P3^5;
sbit BC = P3^6;
sbit BD = P3^7;
```

```
// Display D1
sbit D1A = P1^0;
sbit D1B = P1^1;
sbit D1C = P1^2;
sbit D1D = P1^3;
sbit D1E = P1^4;
sbit D1F = P1^5;
sbit D1G = P1^6;
sbit D1DF = P1^7;

// Display D2
sbit D2A = P2^0;
sbit D2B = P2^1;
sbit D2C = P2^2;
sbit D2D = P2^3;
sbit D2E = P2^4;
sbit D2F = P2^5;
sbit D2G = P2^6;
sbit D2DF = P2^7;

// conta = 1 -> 250 microsegundos
// conta = 400 -> 0.1 segundo
// Conta interrupcoes do timer 0
unsigned int conta = 0;

// Tempo inicial de 5.0 segundos
int segundosIniciais = TempoInicial;

// Resposta do participante (1->A, 2->B, 3->C ou 4->D)
int resposta = 0;

// Variavel de controlo que se mudar de valor o botão B1 foi clicado
bit clicouB1 = 0;

/*
Variavel de controlo para o botao B1 para escolher a acao a ser tomada
Esta a '0' se esta a ser mostrado o tempo inicial nos displays (Nao foi clicado no botao B1)
Esta a '1' se informacao com o tempo/resposta do participante esteja a ser mostrada nos
displays (Ja foi clicado no botao B1)
*/
bit opcaoB1 = 0;

/*
```

```
Variavel de controlo para o botao B1 para escolher a acao a ser tomada
Esta a '0' se o participante ainda nao respondeu (Não clicou numa das opcoes de resposta durante os 5.0 segundos)
Esta a '1' se o participante ja respondeu (Clicou numa das opcoes de resposta durante os 5.0 segundos)
*/
bit respondeu = 0;

// Segmentos dos displays com todos os numeros, simbolos e letras
code unsigned segments[22][8] = {
        {1, 1, 1, 1, 1, 1, 0, 0}, // -.
        {0, 0, 0, 0, 0, 0, 1, 0}, // 0.
        {1, 0, 0, 1, 1, 1, 1, 0}, // 1.
        {0, 0, 1, 0, 0, 1, 0, 0}, // 2.
        {0, 0, 0, 0, 1, 1, 0, 0}, // 3.
        {1, 0, 0, 1, 1, 0, 0, 0}, // 4.
        {0, 1, 0, 0, 1, 0, 0, 0}, // 5.
        {1, 1, 1, 1, 1, 1, 0, 1}, // -
        {0, 0, 0, 0, 0, 0, 1, 1}, // 0
        {1, 0, 0, 1, 1, 1, 1, 1}, // 1
        {0, 0, 1, 0, 0, 1, 0, 1}, // 2
        {0, 0, 0, 0, 1, 1, 0, 1}, // 3
        {1, 0, 0, 1, 1, 0, 0, 1}, // 4
        {0, 1, 0, 0, 1, 0, 0, 1}, // 5
        {0, 1, 0, 0, 0, 0, 0, 1}, // 6
        {0, 0, 0, 1, 1, 1, 1, 1}, // 7
        {0, 0, 0, 0, 0, 0, 0, 1}, // 8
        {0, 0, 0, 0, 1, 0, 0, 1}, // 9
        {0, 0, 0, 1, 0, 0, 0, 1}, // A
        {1, 1, 0, 0, 0, 0, 0, 1}, // B
        {0, 1, 1, 0, 0, 0, 1, 1}, // C
        {1, 0, 0, 0, 0, 1, 0, 1}, // D
};

// Inicializacao do sistema
void Init (void)
{
        // Configuracao do registo
        EA = 1; // Ativa as interrupcoes globais
        ET0 = 1; // Ativa a interrupcao timer 0
        EX0 = 1; // Ativa a interrupcao externa 0
        EX1 = 0; // Desativa a interrupcao externa 1

        // Timer no modo 2, de 8 bits com auto-reload
```

```
	TMOD &= 0xF0; // Limpa os bits menos significativos
	TMOD |= 0x02; // Timer 0 no modo 2

	TH0 = TempoH; // Inicializa o valor de TH0
	TL0 = TempoL; // Inicializa o valor de TL0 (6 microsegundos)

	IT0 = 1; // Interrupcao externa 0 activa a falling edge
	IT1 = 1; // Interrupcao externa 1 activa a falling edge
	TR0 = 0; // Timer 0 nao comeca
}

// Interrupcao externa 0
void External0 (void) interrupt 0
{
	// Inverte o valor da variavel de controlo 'clicouB1'
	clicouB1 = ~clicouB1;

	// Se a variavel de controlo 'opcao1' estiver a 1
	if(opcaoB1){
		IE0 = 0; // Limpa a flag da interrupcao externa 0
		EX1 = 0; // Desativa a interrupcao externa 1
		TR0 = 0; // Timer0 para de contar o tempo
		segundosIniciais = TempoInicial; // Repoe o tempo inicial de 5.0 segundos
		conta = 0; // Repoe a contagem de interrupcoes do timer 0
	}
	else {
		EX0 = 0; // Desativa a interrupcao externa 0
		IE1 = 0; // Limpa a flag da interrupcao externa 1
		EX1 = 1; // Ativa a interrupcao externa 1
		TR0 = 1; // Timer0 comeca a contar tempo
	}
}

// Interrupcao externa 1
void External1 (void) interrupt 2
{
	EX1 = 0; // Desativa a interrupcao externa 1
	// Enquanto o botao de opcao de resposta estiver pressionado, o tempo tiver comecado
a contar e nao tiver terminado
	while(!Pressionado && segundosIniciais != TempoInicial && segundosIniciais != 0){
		// Se a opcao de resposta A for pressionada
		if (!BA){
			respondeu = 1; // A variavel de controlo 'respondeu' passa a 1
			resposta = 1; // A resposta do participante e a opcao A
```

```
		} else if (!BB){ // Se a opcao de resposta B for pressionada
			respondeu = 1;
			resposta = 2;

		} else if (!BC){ // Se a opcao de resposta C for pressionada
			respondeu = 1;
			resposta = 3;

		} else if (!BD){ // Se a opcao de resposta D for pressionadaß

			respondeu = 1;
			resposta = 4;
		}
	}
}

// Interrupcao do timer 0
void Timer0_ISR (void) interrupt 1
{
	conta++; // Incrementa a variavel 'conta' a cada interrupcao do timer 0

	// Se a variavel de controlo 'respondeu' estiver a 0
	if (!respondeu){
		// Se a contagem de interrupcoes do timer 0 for igual a 400 (0.1 segundo)
		if (conta == 400){
			segundosIniciais--; // Decrementa o tempo atual
			conta=0; // Recomeca a contagem de interrupcoes do timer 0
		}
	}
}

// Mostra o caracter correspondente ao lugar do segmento no display
void display (int num1, int num2)
{
	// Segmentos do display D1
	D1A = segments[num1][0];
	D1B = segments[num1][1];
	D1C = segments[num1][2];
	D1D = segments[num1][3];
	D1E = segments[num1][4];
	D1F = segments[num1][5];
	D1G = segments[num1][6];
	D1DF = segments[num1][7];
```

```
	// Segmentos do display D2
	D2A = segments[num2][0];
	D2B = segments[num2][1];
	D2C = segments[num2][2];
	D2D = segments[num2][3];
	D2E = segments[num2][4];
	D2F = segments[num2][5];
	D2G = segments[num2][6];
	D2DF = segments[num2][7];
}

// Mostra os segundos nos displays
void displaySegundos (int num)
{
	// Calcula as dezenas e unidades do numero
   int dezenas = num / 10;
   int unidades = num % 10;

	// Mostra o numero nos displays
	display(dezenas+1, unidades+8);
}

// Mostra a resposta nos displays
void semResposta (void)
{
	bit opcao = 0; // Variavel de controlo para a opcao a ser mostrada
	conta = 0; // Reseta a contagem de interrupcoes do timer 0
	respondeu = 1; // A variavel de controlo 'respondeu' passa a 1

	clicouB1 = 1; // A variavel de controlo 'clicouB1' passa a 1
	IE0 = 0; // Limpa a flag da interrupcao externa 0
	EX0 = 1; // Ativa a interrupcao externa 0
	EX1 = 0; // Desativa a interrupcao externa 1

	TR0 = 1; // Timer0 comeca a contar tempo

	// Mostra o valor final do tempo nos displays (0.0)
	display(1,8);

	// Enquanto o botao B1 nao for clicado
	while(clicouB1){
		// Se a contagem de interrupcoes do timer 0 for igual a 4000 (1 segundo)
		if (conta == segundo){
```

```
                        // Se a opcao for 0
                        if (opcao == 0){
                                // Mostra a resposta indefinida nos displays
                                display(0, 7);
                                opcao = 1; // A opcao passa a 1
                        } else { // Se a opcao for 1
                                // Mostra o valor final do tempo nos displays (0.0)
                                display(1, 8);
                                opcao = 0; // A opcao passa a 0
                        }
                        conta = 0; // Reseta a contagem de interrupcoes do timer 0
                }
                opcaoB1 = 1; // A variavel de controlo 'opcaoB1' passa a 1
        }

        TR0 = 0; // Timer0 para de contar tempo
        conta = 0; // Reseta a contagem de interrupcoes do timer 0
        respondeu = 0; // A variavel de controlo 'respondeu' passa a 0
}

// Mostra a informacao nos displays
void mostraInformacao(void)
{
        bit opcao = 0; // Variavel de controlo para a opcao a ser mostrada
        conta = 0; // Reseta a contagem de interrupcoes do timer 0

        clicouB1 = 1; // A variavel de controlo 'clicouB1' passa a 1
        IE0 = 0; // Limpa a flag da interrupcao externa 0
        EX0 = 1; // Ativa a interrupcao externa 0
        EX1 = 0; // Desativa a interrupcao externa 1

        TR0 = 1; // Timer0 comeca a contar tempo

        // Enquanto o botao B1 nao for clicado
        while (clicouB1){
                // Se a contagem de interrupcoes do timer 0 for igual a 4000 (1 segundo)
                if (conta == segundo){
                        // Se a opcao for 0
                        if (opcao == 0){
                                // Mostra o tempo em que clicou na resposta nos displays
                                displaySegundos(segundosIniciais);
                                opcao = 1;
                        } else { // Se a opcao for 1
                                // Mostra a resposta do participante nos displays
```

```
                        display(0, resposta+17);
                        opcao = 0;
                    }
                    conta = 0; // Reseta a contagem de interrupcoes do timer 0
                }
                opcaoB1 = 1; // A variavel de controlo 'opcaoB1' passa a 1
        }

        TR0 = 0; // Timer0 para de contar tempo
        conta = 0; // Reseta a contagem de interrupcoes do timer 0
        respondeu = 0; // A variavel de controlo 'respondeu' passa a 0
}

// Funcao principal
void main (void)
{
        // Inicializacao do sistema
        Init();

        // Ciclo infinito
        while(1){

                // Mostra o tempo atual nos displays
                displaySegundos(segundosIniciais);

                // Se o tempo chegar ao fim
                if (segundosIniciais <= 0){
                        // Funcao que mostra a resposta indefinida e o tempo final nos displays
                        semResposta();
                }

                // Se clicou em algum dos botoes de opcao de resposta e o tempo ainda nao
chegou ao fim
                if (respondeu && segundosIniciais < TempoInicial){
                        // Funcao que mostra a resposta e o tempo de resposta nos displays
                        mostraInformacao();
                }

                opcaoB1 = 0; // A variavel de controlo 'opcaoB1' passa a 0
        }
}
```

Código em linguagem Assembly

```
; Displays
D1 EQU P1
D2 EQU P2

TempoInicial EQU 50          ; Tempo inicial em segundos
TempoConta EQU 20            ; Conta ate 20 interrupcoes para decrementar 0.1 segundos
segundo EQU 200              ; Conta ate 200 interrupcoes para decrementar 1 segundo

TempoH0 EQU 0x06             ; Tempo inicial do timer 0
TempoL0 EQU 0x06             ; Tempo inicial do timer 0 (6 microsegundos)

ClicouB1 EQU 40H             ; Bit de controlo para indicar se o botao B1 foi clicado
OpcaoB1 EQU 42H              ; Bit de controlo para indicar a opcao a fazer quando clicar
no botao B1
Respondeu EQU 44H            ; Bit de controlo para indicar se o utilizador clicou num
botao de resposta

; Definicoes de portas
B1 EQU P3.2
Pressionado EQU P3.3
BA EQU P3.4
BB EQU P3.5
BC EQU P3.6
BD EQU P3.7

CSEG AT 0300H
; Tabela de segmentos para mostrar no display (-., 0., 1., 2., 3., 4., 5., -, 0, 1, 2, 3, 4, 5, 6, 7, 8,
9, A, B, C, D)
Segmentos:
   DB  0x3F, 0x40, 0x79, 0x24, 0x30, 0x19, 0x12, 0xBF, 0xC0, 0xF9, 0xA4, 0xB0, 0x99,
0x92, 0x82, 0xF8, 0x80, 0x90, 0x88, 0x83, 0xC6, 0xA1

; Rotina Principal
CSEG AT 0000H
   JMP main

; Interrupcao externa 0
CSEG AT 0003H
        JMP External0_Handler
```

```
; Interrupcao timer 0
CSEG AT 000BH
        JMP Timer0_Handler

; Interrupcao externa 1
CSEG AT 0013H
        JMP External1_Handler

CSEG AT 0050H

; Rotina Principal
main:
   CALL Init                              ; Inicializa o sistema
main_loop:
        CLR OpcaoB1                        ; Reset OpcaoB1
   CALL displaySegundos                    ; Mostra os segundos atuais nos displays

   CJNE R2, #0, CheckResposta              ; Se segundosIniciais for diferente de 0,
salta para CheckResposta

   CALL semResposta                        ; Se segundosIniciais for 0, mostra o tempo
final (0.0) e a resposta indefinida
   JMP main_loop                           ; Repete o loop
CheckResposta:
   JNB Respondeu, main_loop                ; Se não respondeu, salta para main_loop

   CJNE R2, #TempoInicial, CheckMostraInformacao   ; Se respondeu e
segundosIniciais != TempoInicial, salta para CheckMostraInformacao

   JMP main_loop                           ; Se respondeu e segundosIniciais ==
TempoInicial, repete o loop
CheckMostraInformacao:
   CALL mostraInformacao                   ; Mostra a resposta e o tempo de resposta
nos displays
        JMP main_loop                      ; Repete o loop

; Inicializa o sistema
Init:
   MOV R1, #0                              ; Inicializa registrador para a conta de
interrupcoes do timer0
   MOV R2, #TempoInicial                   ; Inicializa registrador para os segundos
iniciais que irao decrementar
   MOV R4, #0                              ; Inicializa registrador resposta dada pelo
utilizador
```

```
        CLR ClicouB1                    ; Inicializa ClicouB1
        CLR OpcaoB1                     ; Inicializa OpcaoB1
        CLR Respondeu                   ; Inicializa Respondeu

        ; Configuracoes iniciais e habilitacao das interrupcoes
   MOV IE, #83H                                    ;EA=1, ET1=0, EX1=0,
ET0=1 e EX0=1 -> IE=10000011
        MOV IP, #00H                                    ;IP = 0

        ;Configuracao Registo TMOD
        MOV TMOD, #00000010b                ;Timer 0 no modo 2 (8 bit - auto
reload)

   MOV TL0, #TempoL0                ; Inicializa TL0 (6 microsegundos)
   MOV TH0, #TempoH0                ; Inicializa TH0

   CLR TR0                          ; Comeca o timer0 desligado
        SETB IT0                    ; Habilita interrupcao externa 0
        SETB IT1                    ; Habilita interrupcao externa 1
        RET

; Interrupcao externa 0
External0_Handler:
   CPL ClicouB1                     ; Inverte ClicouB1
        JB OpcaoB1, External0_Clicked       ; Se OpcaoB1 for 1, salta para
External0_Clicked
        JMP External0_NotClicked            ; Se OpcaoB1 for 0, salta para
External0_NotClicked
External0_Clicked:
        CLR IE0                     ; Limpa a flag de interrupcao externa 0
        CLR EX1                     ; Desabilita interrupcao externa 1
   CLR TR0                          ; Timer0 para de contar o tempo
   MOV R2, #TempoInicial                ; Reinicia segundosIniciais
   MOV R1, #0                       ; Reinicia conta
   RETI                             ; Retorna da interrupcao externa 0
External0_NotClicked:
        CLR EX0                     ; Desabilita interrupcao externa 0
        CLR IE1                     ; Limpa a flag de interrupcao externa 1
        SETB EX1                    ; Habilita interrupcao externa 1
   SETB TR0                         ; Timer0 comeca a contar o tempo
   RETI                             ; Retorna da interrupcao externa 0

; Interrupcao do timer 0
```

```
Timer0_Handler:
   INC R7                                ; Incrementa o contador de 20 interrupcoes
   CJNE R7, #20, Timer0_End                    ; Se R7 for difrente de 20, salta para
Timer0_End

   INC R1                                ; Incrementa a conta
   MOV R7, #0                              ; Reinicia o contador de 20 interrupcoes

   JB Respondeu, Timer0_End                    ; Se respondeu, salta para Timer0_End
   CJNE R1, #TempoConta, Timer0_End                ; Se conta for diferente de
TempoConta, salta para Timer0_End

   DEC R2                                 ; Decrementa os segundos iniciais
   MOV R1, #0                              ; Reinicia a conta
Timer0_End:
   RETI                                 ; Retorna da interrupcao do timer 0


External1_Handler:
        CLR EX1                             ; Desabilita interrupcao externa 1
   CJNE R2, #TempoInicial, SecondsNotZero          ; Pula para SecondsNotZero se
segundosIniciais for diferente de TempoInicial
   CJNE R2, #0, SecondsNotZero                 ; Pula para SecondsNotZero se
segundosIniciais for diferente de 0
        JMP External1_End                      ; Se segundosIniciais for 0, salta para
External1_End
SecondsNotZero:
        JNB BA, AnswerA                        ; Se BA for pressionado (0), define
resposta como 1
        JNB BB, AnswerB                        ; Se BB for pressionado (0), define
resposta como 2
        JNB BC, AnswerC                        ; Se BC for pressionado (0), define
resposta como 3
        JNB BD, AnswerD                        ; Se BD for pressionado (0), define
resposta como 4
        JMP External1_Handler                  ; Se nenhum botao foi pressionado,
volta para o inicio da interrupcao
AnswerA:
   SETB Respondeu                          ; Define Respondeu como 1
   MOV R4, #1                            ; Define resposta (R4) como 1 (A)
   JMP External1_End                       ; Salta para External1_End
AnswerB:
   SETB Respondeu                          ; Define Respondeu como 1
   MOV R4, #2                            ; Define resposta (R4) como 2 (B)
```

```
   JMP External1_End                    ; Salta para External1_End
AnswerC:
   SETB Respondeu                       ; Define Respondeu como 1
   MOV R4, #3                           ; Define resposta (R4) como 3 (C)
   JMP External1_End                    ; Salta para External1_End
AnswerD:
   SETB Respondeu                       ; Define Respondeu como 1
   MOV R4, #4                           ; Define resposta (R4) como 4 (D)
   JMP External1_End                    ; Salta para External1_End
External1_End:
   RETI                                 ; Retorna da interrupcao externa 1


; Funcao para mostrar um caracter nos displays, entre numeros de segundos e letras de
resposta
; Argumentos:
;R5 (lugar da tabela de segmetos do valor a carregar no display 1)
;R6 (lugar da tabela de segmentos do valor a carregar no display 2)
display:
   MOV DPTR, #Segmentos                 ; Carrega o endereco da tabela de
segmentos

   MOV A, R5                            ; Obtem R5
   MOVC A, @A+DPTR                      ; Carrega Segmentos[R5]
   MOV P1, A                            ; Define D1

   MOV A, R6                            ; Obtem R6
   MOVC A, @A+DPTR                      ; Carrega Segmentos[R6]
   MOV P2, A                            ; Define D2

   RET


; Funcao para mostrar os segundos nos displays
; Argumento:
; R2 (segundos atuais)
displaySegundos:
   MOV A, R2                            ; Obtem R2
   MOV B, #10                           ; Obtem o valor 10
   DIV AB                               ; Divide R2 por 10 (Em A fica o quociente que é as
dezenas e em B o resto que é as unidades)

   INC A                                ; Dezenas + 1 para obter o lugar na tabela de
segmentos com o valor das dezenas
```

```
   MOV R5, A                          ; Passa o valor das dezenas para R5

        MOV A, B                          ; Obtem B que é as unidades
   ADD A, #8                          ; Unidades + 8 para obter o lugar na tabela de
segmentos com o valor das unidades
   MOV R6, A                          ; Passa o valor das unidades para R6

   CALL display                       ; Mostra os segundos nos displays
   RET


; Funcao para mostrar a resposta indefinida e o tempo final nos displays
semResposta:

        CLR TR0                           ; Para de contar o tempo
        MOV B, #0                         ; Reseta opcao (B) para 0
   MOV R1, #0                         ; Reseta conta (R1) para 0
   SETB Respondeu                     ; Define Respondeu como 1

        SETB ClicouB1                     ; Define ClicouB1 como 1

        CLR IE0                           ; Limpa a flag de interrupcao externa 0
        SETB EX0                          ; Habilita interrupcao externa 0
        CLR EX1                           ; Desabilita interrupcao externa 1

   SETB TR0                           ; Comeca a contar o tempo

        MOV R5, #1                        ; Dá o valor do lugar na tabela de segmentos
para mostrar 0.
   MOV R6, #8                         ; Dá o valor do lugar na tabela de segmentos para
mostrar 0
   CALL display                       ; Mostra 0.0 nos displays

semResposta_Loop:
        SETB OpcaoB1                      ; Define OpcaoB1 como 1
   JNB ClicouB1, semResposta_Fim           ; Se ClicouB1 for 0, salta para
semResposta_Fim

   CJNE R1, #segundo, semResposta_Loop          ; Se conta for diferente de segundo,
repete o loop

        MOV A, B                          ; Obtem a opcao
   CJNE A, #0, DisplayZero            ; Se opcao for diferente de 0, salta para
DisplayZero
```

```
DisplayIndefinido:
        MOV R5, #0                      ; Dá o valor do lugar na tabela de segmentos
para mostrar -.
   MOV R6, #7                           ; Dá o valor do lugar na tabela de segmentos para
mostrar -
   CALL display                         ; Mostra -.- nos displays
   MOV B, #1                            ; Define opcao (B) como 1
   MOV R1, #0                           ; Reseta conta (R1)
   JMP semResposta_Loop                 ; Repete o loop

DisplayZero:
        MOV R5, #1                      ; Dá o valor do lugar na tabela de segmentos
para mostrar 0.
   MOV R6, #8                           ; Dá o valor do lugar na tabela de segmentos para
mostrar 0
   CALL display                         ; Mostra 0.0 nos displays
        MOV B, #0                       ; Define opcao (B) como 0
        MOV R1, #0                      ; Reseta conta (R1)
        JMP semResposta_Loop            ; Repete o loop

semResposta_Fim:
        CLR TR0                         ; Para de contar o tempo
        MOV R1, #0                      ; Reseta conta (R1)
        CLR Respondeu                   ; Define Respondeu como 0
        RET


; Funcao para mostrar a resposta e o tempo de resposta nos displays
mostraInformacao:

        CLR TR0                         ; Para de contar o tempo
   MOV B, #0                            ; Reseta opcao (B) para 0
   MOV R1, #0                           ; Reseta conta (R1) para 0

        SETB ClicouB1                   ; Define ClicouB1 como 1

        CLR IE0                         ; Limpa a flag de interrupcao externa 0
        SETB EX0                        ; Habilita interrupcao externa 0
        CLR EX1                         ; Desabilita interrupcao externa 1

   SETB TR0                             ; Comeca a contar o tempo

mostraInformacao_Loop:
```

```
        SETB OpcaoB1                        ; Define OpcaoB1 como 1
    JNB ClicouB1, mostraInformacao_Fim      ; Se ClicouB1 for 0, salta para
mostraInformacao_Fim

    CJNE R1, #segundo, mostraInformacao_Loop    ; Se conta for diferente de
segundo, repete o loop

        MOV A, B                            ; Obtem a opcao
    CJNE A, #0, DisplayOption               ; Se opcao for diferente de 0, salta para
DisplayOption
DisplaySeconds:
    CALL displaySegundos                    ; Mostra os segundos restantes nos displays
    MOV B, #1                               ; Define opcao (B) como 1
    MOV R1, #0                              ; Reseta conta (R1)
    JMP mostraInformacao_Loop               ; Repete o loop

DisplayOption:
    MOV R5, #0                              ; Dá o valor do lugar na tabela de segmentos para
mostrar -.
    MOV A, R4                               ; Obtem a resposta
    ADD A, #17                              ; Adiciona 17 para obter o lugar na tabela de
segmentos com o valor da resposta
    MOV R6, A                               ; Passa o valor da resposta para R6
    CALL display                            ; Mostra a resposta nos displays
    MOV B, #0                               ; Define opcao (B) como 0
    MOV R1, #0                              ; Reseta conta (R1)
    JMP mostraInformacao_Loop               ; Repete o loop

mostraInformacao_Fim:
    CLR TR0                                 ; Para de contar o tempo
    MOV R1, #0                              ; Reseta conta (R1)
    CLR Respondeu                           ; Define Respondeu como 0

    RET

END
```