Instituto Superior Técnico
Licenciatura em Engenharia Informática e de Computadores

Projecto de Arquitectura de Computadores
2007/2008

Controlo de um Elevador

Índice

Versão	Comentário	Data
1.2	1. O indicador "Carga Total" passa a designar-se "Passageiros × andar". 2. Tempo de abertura e fecho das portas na simulação = 0.	18 de Maio de 2007
1.1	Esclarece um exemplo na secção "Geração Automática de Pedidos". Corrige o opcode da instrução I1OP (1Eh → 17h).	27 de Abril de 2007
1.0	Publicação inicial.	20 de Abril de 2007

1 Objectivo

O objectivo do projecto é a simulação de um controlador de elevador usando o simulador do processador P3 (P3SIM).

Será simulado o ambiente de um prédio com vários andares, com a colocação de pedidos nos vários andares, a deslocação do elevador em resposta aos pedidos, a abertura e fecho das portas de acesso ao elevador e o movimento das pessoas entre andares.

Naturalmente usam-se os recursos do P3SIM para recriar o ambiente do prédio, pelo que os pedidos de deslocação serão colocados através do teclado da janela de interface e o movimento do elevador será mostrado na janela de texto.

O simulador é programado em linguagem *assembly* do P3, sendo também microprogramada uma nova instrução.

2 Simulador de Elevador

2.1 Módulos do Simulador

Como este sistema tem alguma complexidade é conveniente decompô-lo em vários módulos funcionais cooperantes que passamos a descrever. A estrutura do programa a desenvolver deve respeitar a repartição funcional descrita.

2.1.1 Módulo de Controlo

O módulo de controlo actua nos motores dos equipamentos (subida/descida do elevador, abertura/fecho das portas) em resposta à interpretação dos pedidos pendentes. O *algoritmo de controlo* determina o movimento do elevador em função dos pedidos pendentes.

Sugere-se, como primeira aproximação, a realização de um algoritmo simples: O elevador desloca-se para cima ou para baixo enquanto houver pedidos pendentes nos andares que estão no sentido da deslocação. Inicialmente o elevador encontra-se parado no piso 0 com as portas abertas. O sistema pára o elevador com a porta aberta após sair do elevador o último passageiro, não havendo mais pedidos pendentes.

Deixa-se ao critério dos alunos a realização de um algoritmo que "optimize" o comportamento do elevador, devendo, então, ser indicada qual a característica que se pretende melhorar – o tempo de espera das pessoas, a distância percorrida pelo elevador, ou outro. O investimento adicional dos alunos será devidamente valorizado, mas com carácter distintivo.

2.1.2 Módulo de Interface

O módulo de interface captura os pedidos das pessoas. O módulo de controlo comunica com este módulo **exclusivamente** através de uma matriz de estado (estrutura de dados) que armazena os pedidos pendentes.

Neste projecto os dispositivos de interface com os utilizadores serão simulados através das facilidades do simulador do P3:

- Os pedidos são realizados premindo sequencialmente em duas teclas da janela de interface (l_j l_k) correspondendo a `(origem, destino)`, isto é, o andar em que se apresenta o passageiro, seguido do andar de destino. Quando a primeira tecla carregada for "9" (l_9) é gerado automaticamente pelo sistema, aleatoriamente, um par `(origem, destino)`.

2.1.3 Módulo de Visualização

O módulo de visualização representa a consola de monitorização do estado do sistema na janela de texto. Mostra os pedidos pendentes nos pisos, a deslocação do elevador, o estado (aberto/fechado) das portas nos vários pisos, as pessoas que viajam no elevador e outros indicadores do comportamento do sistema. Toda a informação visualizada é obtida a partir da matriz de estado mencionada no ponto anterior e, eventualmente, a partir de outras estruturas de dados.

A imagem na janela de texto do simulador é representada na Figura 1.

```
          1         2         3         4         5
   12345678901234567890123456789012345678901234567890123456789012345678901234567...
 1 |   |   ----------
 2 |   |   |           5
 3 |   |   |
 4 |   |   ----------
 5 |   |   |           4
 6 |   |   |
 7 |   |   ----------
 8 |   |   |           3
 9 |   |   |
10 |   |   ----------
11 |   |   |           2
12 |   |   | 45
13 |   |   ----------
14 |   |   |           1
15 |   |   |
16 |   |   ----------
17 |+--+--+|           R/C     Ciclo:                xxxxx
18 ||13345||                   Andares percorridos: yyyyy
19 |+-----+----------         Passageiros x andar: zzzzz
20 |-------|
21
```

Figura 1.

As pessoas são representadas por algarismos que identificam o andar para onde se dirigem. A ordem das pessoas nos andares corresponde à ordem de colocação dos pedidos. Os pedidos mais antigos do andar aparecem à esquerda.

A figura representa o elevador no R/C com 5 pessoas: Uma dirige-se para o 1º piso, duas para o 3º piso, uma para o 4º piso e outra para o 5º piso. Duas pessoas aguardam o elevador no 2º piso. A primeira a colocar o pedido pretende subir para o 4º piso, a outra para o 5º piso.

O elevador desloca-se verticalmente no poço, abrindo-se automaticamente as portas para entrarem ou saírem pessoas. O movimento é simulado deslocando a cabina do elevador e as pessoas que nele viajam linha a linha na janela de E/S.

```
|+--+--+|            R/C
||13345||
|+-----+----------
```

Elevador com as portas fechadas.

```
|+--+--+|            R/C
||13345
|+-----+----------
```

Elevador com as portas abertas.

O módulo de visualização apresenta ainda alguns indicadores gerais sobre o sistema:

- Ciclo de simulação – Conta os passos de simulação.
- Andares percorridos – Número de andares percorridos desde o início da simulação. (Incrementa sempre que há mudança de andar).
- Passageiros × andar[1] – Somatório do número de passageiros transportados de um andar para o andar seguinte. Só contam os passageiros transportados entre andares e não aqueles que, por qualquer eventualidade, permaneçam na cabine com esta parada. Este indicador é uma medida do trabalho executado pelo elevador.

Todos os indicadores são representados por inteiros de 16 bits.

Poderão, opcionalmente, ser apresentados outros indicadores na janela de texto desde que preservem a formatação aqui apresentada, limitando-se, portanto, a usar a área à direita da imagem de referência (da coluna 30 em diante nas primeiras 15 linhas).

- OPÇÃO: O mostrador interno ao elevador, que indica o andar corrente, é apresentado no *display* de sete segmentos mais à direita da janela de interface.

2.1.4 Matriz de Estado

A matriz de estado é a estrutura central de representação do estado do sistema.

Em cada linha da matriz, correspondente a um piso, são armazenados os pedidos pendentes colocados nesse piso. Por exemplo,

`pedido[3,0] ... pedido[3,5]` = 1, 1, 0, 5, FFFFh, FFFFh,

indica que no 3º piso há 4 pedidos pendentes – 2 pedidos de deslocação para o 1º piso, um pedido para o rés do chão (piso 0) e um pedido para o 5º piso. As entradas preenchidas com FFFFh representam a ausência de mais pedidos. Os pedidos estão ordenados por ordem de chegada.

[1] Anteriormente designado "Carga total do elevador".

Quando uma pessoa entra no elevador é retirado o pedido mais à esquerda na linha – `pedido[i,0]` – e os restantes pedidos são deslocados para a esquerda, inserindo pela direita FFFFh.

A matriz é armazenada em memória por linhas, isto é

```
pedido[0,0],pedido[0,1],...,pedido[0,5],pedido[1,0],...,pedido[5,5]
```

são armazenados em endereços crescentes de memória.

Um vector de estado do elevador representa o piso em que a cabina se encontra, o sentido do movimento e os destinos dos passageiros que nela se encontram.

```
Elevador[0] ; Piso
Elevador[1] ; Sentido: 0 – Parado, 1 – Subir, 2 - Descer
```

2.1.5 Módulo Gerador de Pedidos

O gerador de pedidos recebe os pedidos de serviço colocados no teclado da janela de interface do P3SIM ou gerados pelo próprio simulador e insere-os na matriz de estado.

No modo padrão os pedidos são colocados no teclado. Adicionalmente será possível gerar pedidos automaticamente, mantendo-se sempre disponível a colocação através do teclado.

2.1.5.1 Geração Automática de Pedidos

A colocação de pedidos é sequenciada ao ritmo do relógio de simulação.

O gerador de pedidos consulta uma *lista de pedidos* (não confundir com a *matriz de estado*) em que cada pedido é constituído pelo triplo `(atraso, origem, destino)`.

`atraso` identifica o número de unidades de tempo de simulação que medeiam entre o pedido anterior e este pedido, `origem` identifica o andar em que o pedido é colocado e `destino` o andar de destino. Em conclusão, a lista de pedidos é o filme dos pedidos ao longo do tempo, a matriz de estado é a fotografia do sistema num dado instante.

Exemplos:

(0,1,2) (2,1,3) ...

representa um pedido de subida do 1º para o 2º piso, colocado em t=0 (início da contagem do tempo), seguido, após duas unidades de tempo, por outro pedido de deslocação do 1º para o 3º piso.

... (4,3,2) (0,1,3) ...

representa dois pedidos simultâneos colocados em $\Delta t = 4$ relativamente ao pedido que os antecede.

Uma entrada a zeros – (0,0,0) – identifica um pedido nulo que não tem impacto no funcionamento do simulador. A entrada (FFFFh, FFFFh, FFFFh) assinala o último pedido da lista.

A lista de pedidos é constituída por uma matriz de inteiros com capacidade para armazenar 32 pedidos e com um apontador que vai avançando com a simulação – `lista_pedidos[32,3]`.

Um pedido é realizado (activado) para simulação quando é inserido pelo módulo gerador de pedidos na matriz de estado. Não há, portanto, inserção directa de pedidos na matriz de estado a partir de interfaces externas (por exemplo, das rotinas que servem o teclado).

2.2 Ciclo de Simulação

O simulador evolui de acordo com um passo de simulação (ciclo de simulação).

Para simplificar a construção do simulador pode-se considerar que a evolução entre pisos demora três passos. As acções necessárias à simulação – captura de pedidos, algoritmo de controlo, visualização do estado – podem ser executadas em qualquer destes três passos.

No mínimo a visualização do estado deve mostrar o elevador a mudar de andar à medida que ele se desloca, isto é, no mínimo deverá ser feito um refrescamento da janela de texto por andar.

- OPÇÃO: Será, no entanto, mais intuitivo, animar o movimento do elevador fazendo-o subir ou descer linha a linha na janela de texto.

Para melhorar a percepção de reacção do elevador é recomendável avaliar os pedidos pendentes no passo imediatamente anterior à execução de alguma acção – paragem do elevador, mudança de direcção, etc. É igualmente recomendável mostrar os pedidos colocados no teclado logo que eles são capturados para melhorar a interactividade do simulador.

2.3 Simulação

É possível seleccionar em execução diferentes modos de funcionamento do simulador.

2.3.1 Modos de funcionamento

Os modos de geração de pedidos determinam a forma como são gerados os pedidos. Existem dois modos:

- *Modo normal* – O sistema reage aos pedidos que são colocados através do teclado. O pedido é capturado do teclado e é inserido na matriz de estado para processamento.
- *Modo automático* – O sistema percorre uma lista de pedidos residente em memória – `lista_pedidos[i,j,k]` – inserindo os pedidos sequencialmente na matriz de estado.

Cada elemento da lista de pedidos é um triplo `(atraso,origem,destino)` que especifica o atraso relativamente ao pedido anterior, medido em unidades de tempo de simulação, o andar onde o pedido é colocado e o andar de destino.

Tipicamente a lista de pedidos é inicializada com um conjunto de pedidos que representa um padrão de teste de uma determinada funcionalidade do sistema.

Mesmo em modo automático podem ser colocados pedidos através do teclado que são enviados para o gerador de pedidos para inserção, sem tempo de espera, na matriz de estado.

O modo de funcionamento é seleccionado num interruptor da janela de interface.

2.3.2 Ritmo da Simulação

O ritmo da simulação define a unidade de tempo correspondente a cada passo da simulação.

- Em *ritmo normal* cada passo de simulação deverá demorar 1 seg.
- Em *ritmo expresso* cada passo de simulação demora 0,2 seg. ou, se tal não for atingível, o menor tempo possível.

2.3.3 Selecção de Modo

Os modos de funcionamento são seleccionáveis por configuração dos interruptores mais à direita na janela de interface.

	S1 Geração de pedidos
1	Automático
0	Normal

	S0 Ritmo de simulação
1	Expresso
0	Normal

2.4 Instalação a Simular

A instalação a simular tem as seguintes características:

- Edifício com 6 andares – R/C (piso 0) a 5º piso.
- Características do elevador:
  - Carga máxima do elevador = 5 pessoas.
  - Acesso através de porta dupla (na cabine e no andar) com abertura e fecho sincronizados.
  - Velocidade de movimentação da cabine = 3 ciclos de simulação / piso .
  - Tempo de paragem por andar = ciclos de simulação num piso = (número de saídas + número de entradas) . Isto é, um ciclo por cada entrada ou saída. Portanto, na simulação, as portas abrem e fecham instantaneamente.
  - Inicialmente o elevador está parado no piso 0.
- Acesso ao elevador:
  - Número máximo de pedidos por andar = 6.
  - Ao abrir as portas só entram no elevador as pessoas que se destinem a um andar que esteja no sentido do movimento do elevador, excepto se o elevador

estiver vazio e sem sentido pré-definido, caso em que a primeira pessoa a entrar determina o novo sentido do movimento do elevador.

- A entrada no elevador faz-se por ordem de colocação do pedido (pessoa mais à esquerda primeiro).
- As pessoas saem do elevador para o andar e desaparecem do cenário simulado.

Deverão ser declaradas as seguintes estruturas:

- Matriz de estado: `Matriz TAB 36  ; 6×6`
- Vector de estado do elevador: `Elevador TAB 7`
- Lista de pedidos: `Lista TAB 96  ; 32×3`

Exemplos de configurações de teste:

- Em todos os pisos são colocados simultaneamente pedidos para os pisos "seguintes":

```
Lista STR 0,0,1,0,1,2,0,2,3,0,3,4,0,4,5,0,5,0,0,0,0 ...
```

- Teste à capacidade do elevador – 6 pedidos:

```
Lista STR 0,0,5,0,0,5,0,0,5,0,0,5,0,0,5,0,0,5,0,0,0 ...
```

- Teste do algoritmo de controlo:

```
Lista STR 0,1,0,0,2,0,0,3,0,0,4,0,0,5,0,0,0,0 ...
```

3 Microprogramação

Pretende-se microprogramar uma nova instrução *assembly* – RND (*random*) – que gera uma sequência de números pseudo aleatórios:

RND *op*

RND realiza o seguinte algoritmo (registo de deslocamento modificado com realimentação) que gera uma sequência aparentemente aleatória de números de 16 bits, com distribuição uniforme (isto é, os números são equiprováveis), com um passo de repetição elevado:

Mascara = 1000 0000 0001 0110
if (AND (N_i, 0001h) = 0) /* Testa o bit de menor peso */
    N_{i+1} = rotate_right (N_i);
else
    N_{i+1} = rotate_right (XOR (N_i, Mascara));

Em cada invocação RND lê N_i em *op* e deixa o resultado – N_{i+1} – também em *op*. A raiz desta sequência (N_0) é o valor do operando da primeira vez que a instrução é invocada com esse operando. Este valor deverá ser diferente de zero.

O código (*opcode*) de RND é 010111 (17h).

Esta instrução será utilizada para simplificar a geração dos parâmetros dos pedidos em modo automático.

Por restrição do simulador do P3 a mnemónica da instrução deverá ser I1OP.

4 Plano de Desenvolvimento

4.1 Desenvolvimento do trabalho

Sugere-se o seguinte plano de desenvolvimento:

1. Desenhe o fluxograma que descreve cada um dos procedimentos da aplicação, com especial atenção à relação entre o fluxo do programa principal e as várias rotinas de tratamento de interrupção. Estes fluxogramas e a lógica funcional do programa principal e década módulo deverão ser apresentados na 1ª aula de projecto.
2. Para o conjunto de procedimentos que definiu, identifique claramente as entradas, as saídas e os registos modificados na sua execução.
3. Comente e indente devidamente o código desenvolvido. Inclua nos comentários referências aos fluxogramas para auxiliar a leitura e compreensão do programa.
4. Programe e teste minuciosamente cada uma das rotinas que efectuam a interface com os dispositivos de entrada (teclado e interruptores) e os dispositivos de saída (janela de texto), com especial atenção à passagem de parâmetros entre estas rotinas e o programa principal.
5. Associe as rotinas que realizam a interface com os botões de pressão do teclado com o vector de interrupção respectivo.
6. Configure o temporizador disponibilizado pelo simulador e associe o vector de interrupção respectivo com a rotina a executar periodicamente.
7. Realize a ligação entre os vários procedimentos por forma a obter o comportamento desejado e especificado.
8. Estando o sistema a funcionar correctamente pode incluir as funções opcionais que tenha desenvolvido.

4.2 Faseamento da codificação

Não deve tentar codificar todo o programa de uma só vez. Implemente as várias funcionalidades do programa de uma forma faseada e efectue os testes necessários para verificar o seu correcto funcionamento. Não prossiga para a implementação de funcionalidades mais avançadas sem ter garantido que as que lhe servem de base estão correctamente implementadas.

5 Plano de Entrega

1ª Aula (24 de Abril a 2 de Maio)

Entrega:	• Fluxogramas da aplicação e dos principais procedimentos que lhe servem de base (mesmo que ainda não estejam implementados), com especial atenção à relação entre o fluxo do programa principal, os vários módulos funcionais e as rotinas de tratamento de interrupção.
2ª Aula (17 de Maio a 23 de Maio)[2]	
Entrega:	• Microcódigo da instrução I1OP que gera um número pseudo-aleatório (preencher a tabela do Anexo A). • Simulador do Elevador: Código desenvolvido devidamente comentado.

Demonstração:	• Execução da instrução I1OP.
	• Funcionalidades do Simulador:
	  • Leitura de pedidos através das teclas da janela de interface.
	  • Apresentação do cenário a simular na janela de texto.

3ª Aula (24 de Maio a 30 de Maio)	
Entrega:	• Breve relatório com a descrição do projecto realizado, organização do programa e explicação dos aspectos mais relevantes da implementação. Na conclusão deverá ser feito um balanço do que foi realizado, com indicação dos aspectos nos quais o projecto tenha divergido do enunciado base (funcionalidades adicionais implementadas, funcionalidades não implementadas, outras variações ou divergências, etc.). • Fluxogramas finais da aplicação e dos principais procedimentos que lhe servem de base. • Código desenvolvido devidamente comentado (impresso frente e verso a duas páginas por face).
Funcionalidades:	• Trabalho realizado.

Todas as entregas devem ser feitas num envelope identificado com o dia, hora, turno e número de grupo.

Os documentos referidos na tabela anterior devem ser entregues na aula correspondente.

A entrega final deve conter ainda:

- Relatório.
- Código impresso. Cada ficheiro deverá ser impresso utilizando 2 páginas por face, frente e verso, numeradas; sem linhas cortadas; devidamente comentado e indentado; agrafado; com a identificação do grupo como comentário na primeira página. Para tal deverá utilizar a aplicação **p3print** fornecida no site da cadeira.

[2] Daqui em diante as datas estão sujeitas a confirmação.

- CD-ROM com o código e relatório (devidamente identificado com o número do grupo).

Os projectos que sejam entregues após a data limite serão aceites com uma penalização de 1 valor por cada fracção de dia de atraso (incluindo sábados, domingos, feriados e outros dias não úteis).

Após a entrega dos projectos, será afixado na página da cadeira o respectivo horário de discussões.

As discussões dos projectos realizar-se-ão entre 31 de Maio e 14 de Junho. As discussões dos grupos que entreguem os trabalhos fora do prazo poderão ser adiadas pelos docentes da cadeira para dias diferentes daqueles previstos para a generalidade dos grupos.

Anexo A

(Na página seguinte.)

Microprogramação

GRUPO: _____

	End. Simb.	Transferência de Registos	F	M5		SR1	SR2	IAK	FM				CALU					MA	MB	M2	MRB	RB				WM	WR	MD		MAD	RAD				
								LS	MCOND			CC	LI	LF	CONST / NA																				
(1)	IF0	RI ← M[PC]																																
(2)																																		
(3)																																		
(4)																																		
(5)																																		
(6)																																		
(7)																																		
(8)																																		
(9)																																		
(10)																																		
(11)																																		
(12)																																		
(13)																																		
(14)																																		
(15)																																		
(16)																																		
(17)																																		
(18)																																		
(19)																																		
(20)																																		
(21)																																		
(22)																																		
(23)																																		
(24)																																		
(25)																																		
(26)																																		
(27)																																		
(28)																																		
(29)																																		
(30)																																		
(31)																																		
(32)																																		
(33)																																		
(34)																																		
(35)																																		

End. ROM Controlo	Valor

LEGENDA:

Sinal INACTIVO - 0
Sinal ACTIVO - 1
Sinal INDIFERENTE - *não preencher*

Fluxograma	Usar o verso da folha de resposta

Nº Ciclos de Relógio	Mínimo	Médio	Máximo

Endereço ROM A	Valor

Nuno.Roma@inesc-id.pt