MC404: Organização de Computadores e Linguagem de Montagem

Exame (13/12/2011)

Questão	Valor	Nota
1	0,8	
2	1,2	
3	5,0	
4	5,0	
Nota do exame†:		

Nome:

RA:

Instruções: A duração da prova é de uma hora e quarenta minutos. *Consulta exclusivamente às folhas de resumos de instruções do ARM.* Comente seu código! Qualquer tentativa de fraude será punida com zero para todos os envolvidos.

†: **Nota do exame** $= \frac{(Q_1+Q_2)\times(Q_3+Q_4)}{2}$, onde Q_i é a nota da questão i.

Questão 1. (0,8 pontos)
Determine o **maior** e o **menor** valor que podem ser representados usando-se 15 *bits*. Mostre sua resposta em decimal.

Complemento de 2		Sinal e Magnitude		Complemento de 1		Sem Sinal	
Maior	Menor	Maior	Menor	Maior	Menor	Maior	Menor

Questão 2. (1,2 pontos) Preencha as lacunas em branco da tabela de acordo com a representação da coluna. Preencha o espaço com um traço se o número não puder ser representado no formato da coluna.

Decimal	Binário de 11 *bits*		
	Sem sinal	Complemento de 2	Sinal e Magnitude
2046			
	100 0000 1010		
		100 0000 0000	
			100 0000 0010

Questão 3. (5,0 pontos) Escreva, em linguagem de montagem do ARM, a função

```
int bsearch(int key, int *vetor, int low, int high)
```

que procura um elemento de valor *key* em um vetor de inteiros, ordenado crescentemente, cujo primeiro elemento tem endereço *vetor*. Se esse elemento está presente, a função retorna o índice do elemento no vetor; caso contrário a função retorna -1. Considere que não há elementos repetidos no vetor. A procura do elemento se estende do elemento de índice *low* e o elemento de índice *high*. Por exemplo, para o vetor

```
int vet[8]={1,3,5,7,9,11,12,13,15};
```

a chamada `bsearch(11,&vet,0,7)` retorna 5; e a chamada `bsearch(2,&vet,0,7)` retorna -1.

Questão 4. (5,0 pontos)
Uma empresa de projeto e implementação de *software* aplica uma filosofia de desenvolvimento chamada "minha cara metade" (MCM). Na filosofia MCM um programador divide a tarefa de implementar uma função ou um módulo em *software* com outro programador. A empresa foi contratada para implementar um programa que controla um robô. O programa do robô consiste em um laço principal que lê eventos dos sensores do robô e toma ações de acordo com o evento lido. O programa abaixo está incompleto e foi escrito por um programador da filosofia MCM, a sua tarefa é completar o código do laço princial (apenas o código do laco_principal) abaixo, em linguagem de montagem do ARM, seguindo a seguinte especificação:

- O sistema de *software* deve possuir um laço principal que invoca a função evento para descobrir o evento gerado.
- Se o evento for um `EVENTO_BRONCA`, o robô deve executar a rotina `triste` 2 segundos após o evento.
- Se o evento for um `EVENTO_PIADA`, o robô deve executar a rotina `risada` 1 segundo após o evento.
- Se o evento não for nenhum dos dois acima, você deve chamar a rotina `trata_outros_eventos`.

Suponha que seu programa rode em modo usuário e o sistema ARM já possua um sistema operacional baseado em POSIX/UNIX com as chamadas de sistema `alarm()` e `sigaction()` implementadas. Como visto em um dos trabalhos da disciplina, você pode utilizar as chamadas de sistema `sigaction` e `alarm` para controlar o tempo. A interface (simplificada) das mesmas é descrita abaixo.

- `sigaction`: O número dessa chamada de sistema é 67. Os argumentos são passados através de `r0` e `r1`: `r0` contém o código do sinal e `r1` contém o ponteiro para uma função que será chamada quando o sinal acontecer. O código do sinal SIGALRM é 14.
- `alarm`: O número dessa chamada de sistema é 27. Ela solicita que o sistema operacional dispare um sinal SIGALRM após um determinado tempo. O tempo, especificado em segundos, é informado em `r0`.

```
.set EVENTO_BRONCA   0x01
.set EVENTO_PIADA    0x02

laco_principal:

   bl evento

   b laco_principal

@ Rotina que retorna o evento gerado
@ r0: contem o valor de retorno
evento:
  ...

@ Rotina que faz o robo rir
risada:
  ...

@ Rotina que faz o robor exibir uma expressao triste
triste:
  ...
```