## UFC - Quixadá - QXD0005 - Arquitetura de Computadores- Prof. Cristiano - Trabalho AP2

## 1) Descrição

Implementar um programa (em C ou C++) que simule a execução de instruções mostradas na tabela de decodificação fornecida na Seção 4. Considere que o processador possui 16 bits e os registradores R0-R7, PC, IR, SP e FLAGS.

### 2) Especificação

a) O simulador deverá receber como entrada um arquivo de texto contendo o código a ser executado, em notação hexadecimal. Este arquivo representa a memória de programa, onde cada linha corresponde a um endereço com alinhamento de 16 bits, no formato <endereço>:<conteúdo>.

### Exemplo:

Código	Conteúdo do arquivo de texto
Inicio:	0000: 0x1803
MOV R0, #3	0002: 0x1905
MOV R1, #5	0004: 0x4220
ADD R2, R1, R0	0006: 0x0009
PSH R2	0008: 0x0702
POP R7	000A: 0x52E4
SUB R2, R7, R1	000C: 0x005F
CMP R2, R7	000E: 0x0FC1
JEQ Inicio	0010: 0x1040
MOV R0, R2	0012: 0x0FF8
fim:	
JMP fim	

- b) O programa deve executar até encontrar uma instrução HALT OU uma instrução de formato indefinido OU atingir o final do arquivo.
- c) Ao final da execução, o simulador deverá apresentar o conteúdo (em notação hexadecimal, indicada pelo prefixo '0x') dos seguintes componentes:
- $-\,Registradores$
- $\rightarrow$  R0-R7, PC, LR e SP.

- Memória de dados
- → No mesmo formato do arquivo de entrada (<endereço>:<conteúdo>)
- → Assuma que toda a faixa de endereços está disponível e que a memória está inicialmente zerada. Devem ser exibidas apenas as posições que forem acessadas pelo código (desconsiderando a pilha).
- Pilha
- → No mesmo formato do arquivo de entrada (<endereço>:<conteúdo>)
- → Assuma que o ponteiro de pilha com o valor inicial de 0x82000000 e que pilha possui tamanho de 16 bytes
- Flags
- → Exiba os valores finais das flags C, Ov, Z, S
- → Assuma que as todas iniciam zeradas
- d) Sempre que ocorrer uma instrução NOP, o simulador deve exibir as mesmas informações descritas no item c.

#### 3) Avaliação e Entrega

- a) Cada trabalho será apresentado em sala e submetido a 3 casos de teste. Serão verificadas as informações mostradas ao final da execução (e sempre que ocorrer uma instrução NOP).
- → Cada caso de teste vale de 0,0 a 3,0 pontos;
- → A cada verificação será contabilizada uma pontuação igual a 3,0 / (n+1), onde n é o número de instruções NOP presentes no código.
- → Será descontado 1,0 ponto para cada item que não exiba os valores corretos ou que não atenda **integralmente** as especificações, até no máximo 3,0 pontos por caso de teste.
- b) Será adicionada a pontuação 0,0 a 1,0 conforme as características gerais do projeto, a critério do professor.
- → Será atribuída nota total 0,0 para os trabalhos que apresentarem semelhanças com indícios de cópia.
- c) A pontuação total da equipe será igual aos pontos obtidos multiplicados pelo total de membros da equipe. Essa pontuação deverá ser dividida entre os membros para compor a nota individual de cada um, a critério da equipe.
- d) Entregar até dia 28/02/2025

# 4) Tabela de decodificação

Instrução	Operação	Tipo	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOP	пор	NOP	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
HALT	halt	HALT	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
MOV Rd, Rm	Rd = Rm	MOV	0	0	0	1	0	Rd,	Rd,			Rm,	Rm <sub>o</sub>	-	-	-	-	-
MOV Rd, #Im	Rd = #lm	MOV	0	0	0	1	1	Rd <sub>2</sub>	Rd,	Rd <sub>o</sub>	Im,	Im <sub>o</sub>	lm₅	Im <sub>4</sub>	Im <sub>3</sub>	Im <sub>2</sub>	Im,	Im <sub>o</sub>
STR [Rm], Rn	[Rm] = Rn	STORE	0	0	1	0	0	-	-	-	Rm <sub>2</sub>	$Rm_1$	Rm <sub>o</sub>	Rn <sub>2</sub>	$Rn_1$	Rn <sub>o</sub>	-	-
STR [Rm], #lm	[Rm] = #Im	STORE	0	0	1	0	1	Im,	Im <sub>6</sub>	lm₅	$Rm_2$	$Rm_1$	Rm <sub>o</sub>	Im <sub>4</sub>	Im <sub>3</sub>	Im <sub>2</sub>	Im <sub>1</sub>	Im <sub>o</sub>
LDR Rd. [Rm]	Rd =[Rm]	LOAD	0	0	1	1	-	Rd <sub>2</sub>	$Rd_1$	$Rd_o$	$Rm_2$	$Rm_1$	Rm <sub>o</sub>	-	-	-	-	-
ADD Rd, Rm, Rn	Rd = Rm + Rn	ULA	0	1	0	0	-	Rd <sub>2</sub>	$Rd_1$	$Rd_o$	Rm <sub>2</sub>	$Rm_1$	Rm <sub>o</sub>	Rn₂	$Rn_1$	Rn₀	-	-
SUB Rd, Rm, Rn	Rd = Rm - Rn	ULA	0	1	0	1	-	Rd <sub>2</sub>	$Rd_1$	$Rd_o$	Rm <sub>2</sub>	$Rm_1$	Rm <sub>o</sub>	Rn <sub>2</sub>	Rn <sub>1</sub>	Rn₀	-	-
MUL Rd, Rm, Rn	Rd = Rm * Rn	ULA	0	1	1	0	-	Rd <sub>2</sub>	$Rd_1$	$Rd_o$	$Rm_2$	$Rm_1$	Rm₀	Rn <sub>2</sub>	$Rn_1$	Rn₀	-	-
AND Rd, Rm, Rn	Rd = Rm and Rn	ULA	0	1	1	1	-	Rd <sub>2</sub>	$Rd_1$	$Rd_o$	Rm <sub>2</sub>	$Rm_1$	Rm₀	Rn <sub>2</sub>	$Rn_1$	Rn₀	-	-
QRR Rd, Rm, Rn	Rd = Rm or Rn	ULA	1	0	0	0	-	Rd <sub>2</sub>	$Rd_1$	$Rd_o$	$Rm_2$	$Rm_1$	Rm <sub>o</sub>	Rn <sub>2</sub>	$Rn_1$	Rn <sub>o</sub>	-	-
NOT Rd. Rm	Rd = ¬Rm	ULA	1	0	0	1	-	Rd <sub>2</sub>	$Rd_1$	$Rd_o$	Rm <sub>2</sub>	$Rm_1$	Rm <sub>o</sub>	-	-	-	-	-
XOR Rd, Rm, Rn	Rd = Rm xor Rn	ULA	1	0	1	0	-	Rd <sub>2</sub>	$Rd_1$	$Rd_o$	$Rm_2$	$Rm_1$	$Rm_0$	$Rn_{2}$	$Rn_1$	$Rn_0$	-	-
PSH Rn	[SP] = Rn; SP	PILHA	0	0	0	0	0	-	-	-	-	-	-	Rn <sub>2</sub>	$Rn_{_1}$	$Rn_0$	0	1
POP Rd	SP++; Rd = [SP]	PILHA	0	0	0	0	0	Rd <sub>2</sub>	$Rd_1$	$Rd_o$	-	-	-	-	   <b>-</b>	-	1	0
CMP Rm, Rn	Z = (Rm = Rn)? 1:0; C = (Rm < Rn)? 1:0	ULA	0	0	0	0	0	-	-	-	$Rm_2$	$Rm_1$	$Rm_0$	Rn <sub>2</sub>	$Rn_{_1}$	$Rn_0$	1	1
JMP #Im	PC = PC + #Im	DESVIO	0	0	0	0	1	Im <sub>8</sub>	Im <sub>7</sub>	Im <sub>6</sub>	Im <sub>5</sub>	Im <sub>4</sub>	lm <sub>3</sub>	$lm_2$	$\operatorname{Im}_{\scriptscriptstyle 1}$	$Im_o$	0	0
JEQ #Im	PC = PC + #Im, se Z = 1 e C = 0	DESVIO	0	0	0	0	1	Im <sub>8</sub>	Im <sub>7</sub>	Im <sub>6</sub>	Im <sub>5</sub>	Im <sub>4</sub>	lm <sub>3</sub>	Im <sub>2</sub>	$\operatorname{Im}_{\scriptscriptstyle 1}$	$Im_o$	0	1
JLT #Im	PC = PC + #Im, se Z = 0 e C = 1	DESVIO	0	0	0	0	1	Im <sub>8</sub>	Im <sub>7</sub>	Im <sub>6</sub>	lm₅	$Im_4$	lm <sub>3</sub>	$lm_2$	$\operatorname{Im}_{\scriptscriptstyle 1}$	$\text{Im}_{\text{o}}$	1	0
JGT #Im	PC = PC + #Im, se Z = 0 e C = 0	DESVIO	0	0	0	0	1	Im <sub>8</sub>	Im <sub>7</sub>	Im <sub>6</sub>	Im <sub>5</sub>	Im <sub>4</sub>	lm <sub>3</sub>	Im <sub>2</sub>	$\operatorname{Im}_{\scriptscriptstyle 1}$	$Im_o$	1	1
SHR Rd, Rm, #Im	Rd = Rm >> #Im	ULA	1	0	1	1	-	$Rd_2$	$Rd_1$	$Rd_o$	$Rm_2$	$Rm_1$	$Rm_0$	$Im_4$	lm <sub>3</sub>	$Im_2$	$\operatorname{Im}_{\scriptscriptstyle 1}$	$Im_o$
SHL Rd, Rm, #Im	Rd = Rm << #Im	ULA	1	1	0	0	-	$Rd_2$	$Rd_1$	$Rd_o$	$Rm_2$	$Rm_1$	$Rm_0$	$Im_4$	lm <sub>3</sub>	$Im_2$	$Im_1$	$Im_o$
ROR Rd, Rm	Rd = Rm >> 1; Rd(MSB) = Rm(LSB)	ULA	1	1	0	1	-	$Rd_2$	$Rd_1$	$Rd_o$	$Rm_2$	$Rm_1$	$Rm_0$	-	-	-	-	-
ROL Rd, Rm	Rd = Rm << 1; Rd(LSB) = Rm(MSB)	ULA	1	1	1	0	-	Rd <sub>2</sub>	$Rd_1$	$Rd_o$	$Rm_2$	$Rm_1$	$Rm_o$	-	-	-	-	-