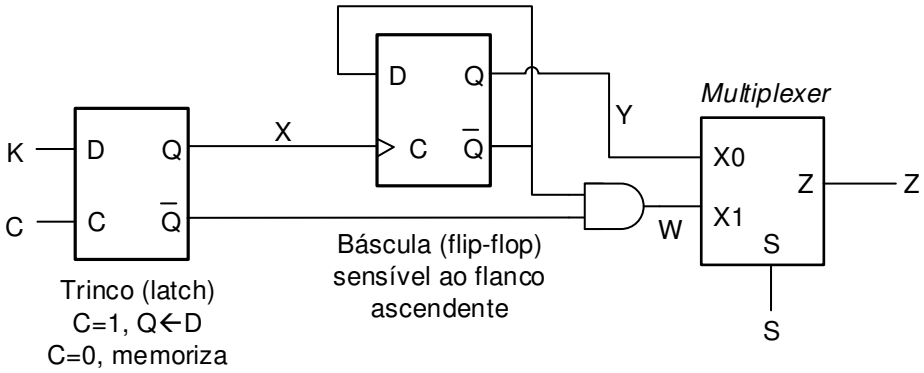
Repescagem - 1.º Teste de Introdução à Arquitetura de Computadores — IST – LEIC-T

1.º Semestre 2014/2015 — Duração: 60 minutos — 16 janeiro 2015

NOME		NÚMERO	

1. (2 valores) Considere o seguinte circuito. Assumindo que os sinais C, K e S evoluem ao longo do tempo da forma indicada na tabela seguinte, acabe de preencher o resto da tabela (o sombreado é apenas para melhor visualização).



C	1	0	0	1	0	1	0	0	1	0	1	0	0	1	0	0	1	0	0	1	0	1	0	1
K	1	1	0	0	1	1	1	0	0	0	1	1	0	0	0	1	1	1	0	0	0	1	1	0
S	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1
X	**1**	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**1**	**1**	**0**
Y	0	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**
W	**0**	**0**	**0**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**
Z	**0**	**0**	**0**	**1**	**1**	**0**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**1**

2. (2+2+2 valores) A subtração A-B é equivalente a somar A com o simétrico de B. A principal vantagem da notação de complemento para 2 é poder tratar os números positivos e negativos de igual forma, de modo que a subtração é desnecessária. Suponha que pretende efetuar a subtração 421 – 2781 (números em base 10), através de uma soma, mas em binário, usando a notação de complemento para 2 com 16 bits.

   a) Calcule os valores em binário com 16 bits, usando a notação de complemento para 2, e faça a soma.

0	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**0**	**1**	**0**	**0**	**1**	**0**	**1**	1.º operando
1	**1**	**1**	**1**	**0**	**1**	**0**	**1**	**0**	**0**	**1**	**0**	**0**	**0**	**1**	**1**	2.º operando
1	**1**	**1**	**1**	**0**	**1**	**1**	**0**	**1**	**1**	**0**	**0**	**1**	**0**	**0**	**0**	Soma

   b) Idem, mas agora em hexadecimal, com 32 bits.

0	**0**	**0**	**0**	**0**	**1**	**A**	**5**	H 1.º operando
F	**F**	**F**	**F**	**F**	**5**	**2**	**3**	H 2.º operando
F	**F**	**F**	**F**	**F**	**6**	**C**	**8**	H Soma

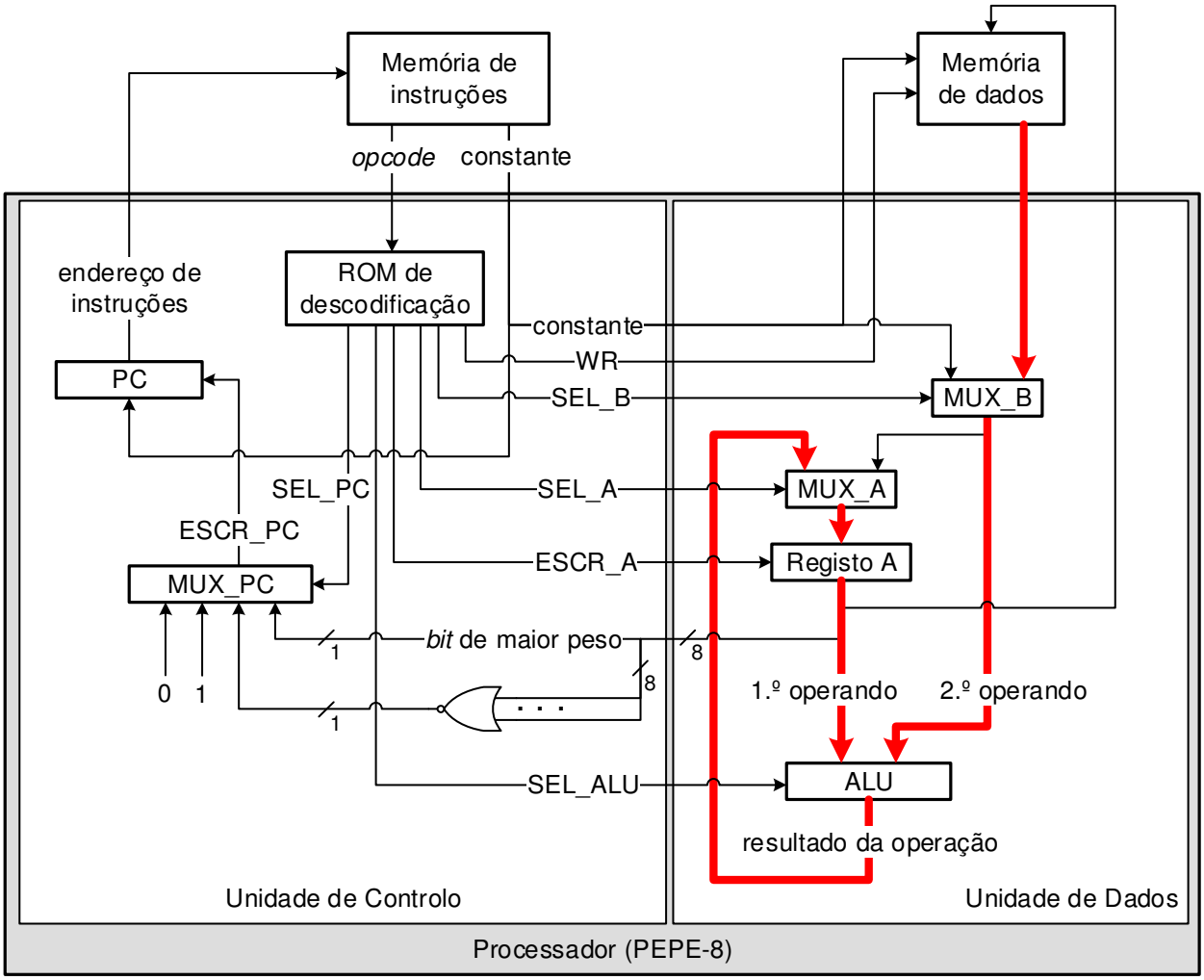
   c) Qual é o menor número de bits dos operandos com que é possível efetuar corretamente esta conta, sempre em notação de complemento para 2? Justifique.

13. Menos que isso faz o 2.º operando ficar com valor errado (o 13.º é o do sinal). O 1.º operando é menos limitativo (poderia ter apenas 10 bits).

3. (2 valores) Quantos valores diferentes é possível representar com 25 bits? Use a notação de K, M, G, etc.

32 M

4. (1,5+1,5 valores) A figura seguinte representa o diagrama de blocos básico do PEPE-8, processador de 8 bits, bem como as memórias a que está ligado.



a) Uma das instruções que o PEPE-8 consegue realizar é ADD [*endereço*], somando o registo A com uma célula de memória e colocando o resultado no registo A. Sobre a figura, reforce com a caneta o percurso dos dados (e apenas estes) durante a execução desta instrução.

b) Na tabela seguinte estão referidos os sinais que a Unidade de Controlo gera para controlar a Unidade de Dados. Preencha esta tabela, especificando para cada sinal qual o seu papel concreto (ou indicação de que não interessa) no caso específico da execução da instrução referida na alínea anterior.

Sinal	Papel concreto do sinal nesta instrução
Constante	**Indica o endereço especificado na instrução**
WR	**Inativo (indica leitura da memória)**
SEL_B	**Seleciona a entrada direita do multiplexer B**
SEL_A	**Seleciona a entrada esquerda do multiplexer A**
ESCR_A	**Ativo (memoriza o resultado no registo A)**
SEL_ALU	**Especifica soma**

5. (2+2+3 valores) Considere o seguinte programa em linguagem *assembly* do PEPE-16. Para facilitar, fornece-se a descrição interna das instruções CALL e RET.

CALL Etiqueta	SP ← SP-2 M[SP]←PC PC ← Endereço da Etiqueta
RET	PC ← M[SP] SP ← SP+2

Endereços				
	N	EQU	4	
0000H		**MOV**	**SP, Z**	
0002H		MOV	R0, X	
0004H		MOV	R1, [R0]	
0006H		CALL	RotA	
0008H		MOV	R0, Y	
000AH		MOV	[R0], R1	
000CH	fim:	JMP	fim	
	; RotA - Recebe R1 como parâmetro e devolve resultado em R1			
000EH	RotA:	**PUSH**	R2	
0010H		MOV	R2, R1	
0012H	proxA:	ADD	R2, -1	
0014H		CMP	R2, 1	
0016H		JLE	saiA	; salta se R2 for menor que ou igual a 1
0018H		CALL	RotB	
001AH		JMP	proxA	
001CH	saiA:	**POP**	**R2**	
001EH		RET		
	; RotB - Recebe R1 e R2 como parâmetros e devolve resultado em R1			
0020H	RotB:	**PUSH**	R2	
0022H		**PUSH**	R3	
0024H		MOV	R3, 0	
0026H	proxB:	ADD	R2, -1	
0028H		JN	saiB	; salta se R2 tiver ficado negativo
002AH		ADD	R3, R1	
002CH		JMP	proxB	
002EH	saiB:	MOV	R1, R3	
0030H		**POP**	**R3**	
0032H		**POP**	**R2**	
0034H		RET		
		PLACE	100H	
0100H	X:	WORD	N	
0102H	Y:	WORD	0	
0104H	pilha:	TABLE	30H	
0164H	Z:			

a) Preencha os endereços de cada instrução (lado esquerdo, preencha apenas as linhas em que tal faça sentido) e os espaços no programa. Considere que todos os MOVs ocupam apenas uma palavra.

b) Indique quais as funções matemáticas (relação entre resultados e parâmetros de entrada) implementadas pelas rotinas RotA e RotB? Justifique, descrevendo sucintamente o funcionamento de cada uma delas.

A rotina RotB implementa a multiplicação (R1 ← R1 * R2). Soma repetidamente R1 a R3 (tantas vezes quanto o valor de R2). No final coloca o valor do produto em R1

A rotina RotA implementa o fatorial de R1. Usando a rotina RotB, multiplica N por N-1, N-2, etc, diminuindo sucessivamente R2, até chegar a 1. O produto é sucessivamente acumulado em R1.

c) Acabe de preencher a tabela com informação sobre os acessos de dados à memória feitos pelo programa, de leitura (L) ou escrita (E). **Para este efeito, considere que rotina RotB é apenas chamada uma vez (ou seja, que a instrução JMP proxA não faz nada)**. *Use uma cruz ou um traço na última coluna caso não se possa saber o valor em causa*. Use apenas as linhas que necessitar.

Endereço da instrução que faz o acesso	Endereço acedido	L ou E	Valor lido ou escrito
0004H	0100H	L	0004H
0006H	0162H	E	0008H
000EH	0160H	E	---------
0018H	015EH	E	001AH
0020H	015CH	E	0003H
0022H	015AH	E	---------
0030H	015AH	L	---------
0032H	015CH	L	0003H
0034H	015EH	L	001AH
001CH	0160H	L	---------
001EH	0162H	L	0008H
000AH	0102H	E	000CH