- Este teste tem 13 questões em 8 páginas, num total de 200 pontos.
- Preencher e entregar as páginas 4 e 5 do enunciado.
- Os problemas 11 e 12 devem ser respondidos em folhas de resposta separadas.

Parte I — Questões de Escolha Múltipla

Cada questão tem uma resposta certa. Respostas erradas não descontam.
As respostas às questões de escolha múltipla devem ser assinaladas com × na grelha da página 8.
Apenas as respostas indicadas na grelha são consideradas para efeitos de avaliação.

[10] 1. O código máquina da instrução `STUR X5,[X4, -1]` é:

A. `0xF81FF0A4` **B.** `0xF81FF085` C. `0xF81010A4` D. `0xF8101085`

[10] 2. Considere o diagrama do processador na figura anexa. Um defeito de fabrico faz com que o bit 30 da saída da memória de instruções fique sempre com o valor 1. Considere os seguintes valores nos registos: `X0` = 100 e `X1` = 50.

Após executar a instrução `add X0, X0, X1`, o resultado em `X0` é:

A. 50 B. Impossível determinar C. 100 D. 150

[10] 3. Assuma que inicialmente `X4` guarda o valor decimal -64. Qual o valor que `X1` passa a guardar depois de se executar o seguinte fragmento de código *assembly*?

```
ASR     X4, X4, 2
MVN     X1, X4
```

A. -15 B. 16 **C. 15** D. -16

[10] 4. Qual das seguintes instruções pode ser executada no caso em que, devido a uma anomalia, os sinais de controlo `Reg2Loc` e `ALUSrc` se encontram fixados, respetivamente, nos valores 1 e 0?

A. `LDUR` B. `AND` **C.** `CBZ` D. `SUB`

[10] 5. Um CPU usa uma memória *cache* do tipo *split cache*. As taxas de faltas da I-Cache e da D-Cache são, respetivamente, t_{if} e t_{df}. Se t_{mem} for a percentagem média de instruções que acedem à memória de dados, qual dos seguintes valores para estes parâmetros levam ao melhor desempenho?

A. $t_{if} = 4\%$, $t_{df} = 3\%$ e $t_{mem} = 90\%$
B. $t_{if} = 8\%$, $t_{df} = 12\%$ e $t_{mem} = 10\%$
C. $t_{if} = 4\%$**,** $t_{df} = 6\%$ **e** $t_{mem} = 30\%$
D. $t_{if} = 8\%$, $t_{df} = 9\%$ e $t_{mem} = 20\%$

[10] 6. Uma memória *cache* é usada para aceder a *doublewords* (64 bits). Os endereços são de 32 bits, as etiquetas são de 13 bits e a *cache* tem 64 blocos. Quantas *doublewords* tem cada um dos blocos?

A. 10 B. 512 C. 8192 **D. 1024**

[10] 7. Um programa científico despende 60 % do seu tempo a executar operações de vírgula flutuante. Para tornar o programa 2 vezes mais rápido pretende-se usar uma nova uma unidade de vírgula flutuante. Quanto mais rápida que a anterior deve ser essa unidade?

A. `2` **B.** `6` C. `8` D. `4`

[10] 8. Qual das afirmações é falsa em relação a memórias *cache* do tipo *write-back*?

A. É possível ler um valor da memória *cache* que seja diferente da memória principal.

B. Existe a possibilidade de serem feitos 2 acessos à memória principal para apenas um acesso à memória *cache*.

C. Pode haver um acesso à memória principal mesmo que o acesso à memória cache seja um acerto (hit).

D. É possível escrever um valor na memória *cache* sem o escrever imediatamente na memória principal.

[10] 9. Assuma que inicialmente as *flags* do processador estão todas a zero. Indique o valor que possuirão após execução do seguinte fragmento de código:

```
MOV   X0, 45
MOV   X1, -90
AND X0, X0, X1
```

A. `N = 0; Z = 1; C = 0; V = 0`

B. `N = 0; Z = 1; C = 1; V = 0`

C. `N = 1; Z = 0; C = 0; V = 1`

D. `N = 0; Z = 0; C = 0; V = 0`

[10] 10. A execução de um programa num computador A o qual funciona com um sinal de relógio de 4 GHz, demora 5s. Um computador B que está a ser desenvolvido é capaz de executar o mesmo programa em 3s, embora consuma 1,2 vezes mais ciclos de relógio do que o computador A a executar o referido programa. Determine a frequência do sinal de relógio de B.

A. `2 GHz` B. `6 GHz` C. `4,5 GHz` **D.** `8 GHz`

Parte II — Questões de Resposta Aberta

- Responder às questões 11 e 12 em folhas de resposta separadas.
- **Justificar todas as respostas.**

11. Considere o CPU ARMv8 simplificado, apresentado na folha de consulta. A latência de componentes usados no CPU é a seguinte (componentes não indicados têm latência nula):

I-Mem	Add	Mux	ALU	Regs	D-Mem	Control	ALU control	
400	100	30	130	250	400	100	40	(ps)

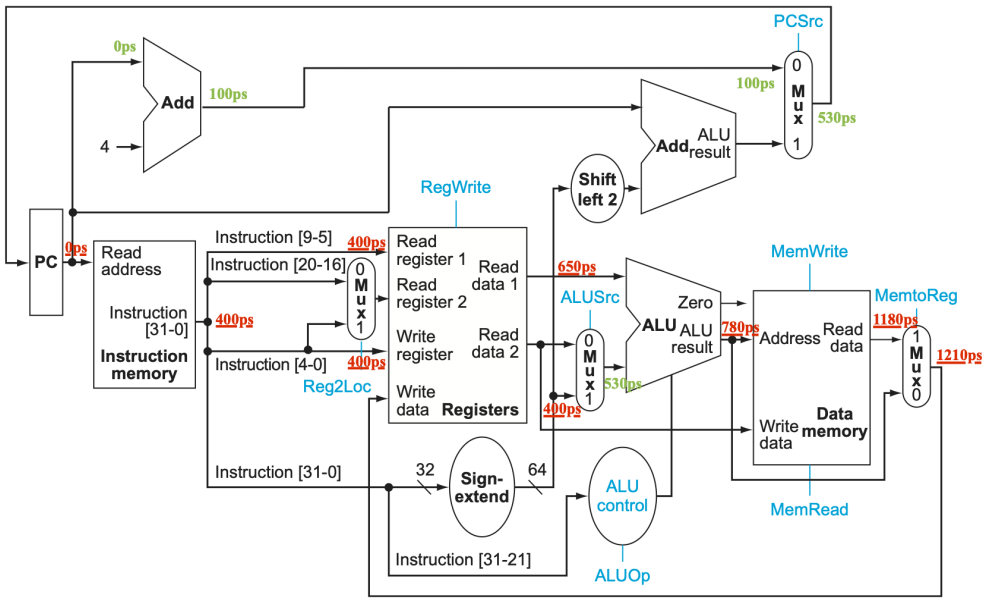
Considere a instrução `LDUR X5, [X9, -8]`.

[10] (a) Assumindo que o valor dos registos X_i é igual a $i + 10$, indique os valores dos seguintes sinais:

- `Read Register 1`
- `Saída do Shift Left 2`
- `Read Data 1`
- `ALU result`

- `Read Register 1`: 9
- `Read Data 1`: $9 + 10 = 19$
- `Saída do Shift Left 2`: $-8 \times 4 = -32$
- `ALU result`: $19 - 8 = 11$

[10] (b) Determine o caminho crítico (caminho de propagação de sinais com maior latência) indicando também o tempo de propagação correspondente.



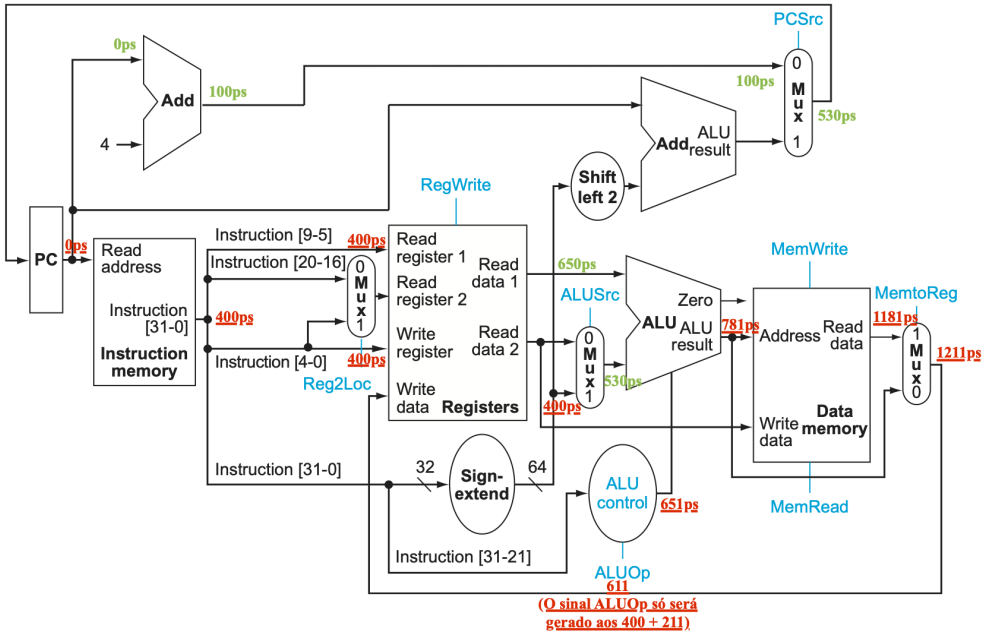
A figura apresenta a latência dos vários sinais importantes para a execução da instrução (`LDUR X5, [X9, -8]`). Analisando-a, conclui-se que o caminho que apresenta a maior latência (caminho crítico) é:

`I-Mem` $\rightarrow$ `Regs` $\rightarrow$ `ALU` $\rightarrow$ `D-MEM` $\rightarrow$ `Mux`

O valor da latência é:

$$400 + 250 + 130 + 400 + 30 = 1210\,\text{ns}$$

[10] (c) Determine qual deveria ser o valor de latência da unidade de controlo de forma a que mesma passe a fazer parte do caminho crítico da instrução da alínea anterior.



Se a unidade de controlo tiver uma latência superior a 210 ns (por exemplo, 211), tanto a mesma como a ALU control passam a fazer parte do caminho crítico (conforme se constata pela figura). Como consequência, o novo caminho crítico será:

`I-Mem` → `Control` → `ALU control` → `ALU` → `D-MEM` → `Mux`.

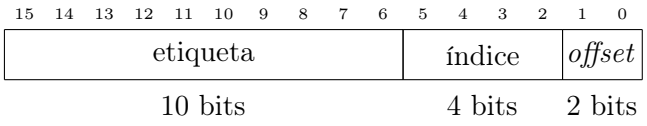
12. Um CPU tem endereços de 16 bits e uma memória *cache* de mapeamento direto para dados (palavras, 32 bits). A memória *cache* é do tipo *write-back*, possui uma palavra por bloco e a respetiva indexação é formada por 4 bits.

    A tabela seguinte mostra parcialmente o conteúdo da *cache* (conteúdo e etiqueta em hexadecimal).

	conteúdo	etiqueta	v	d
...	...	...	...	...
5	7BCDBCD7	241	0	0
6	76543210	1F3	1	0
7	FFFFFFFF	241	1	1
8	80000000	240	1	0
9	26E111A4	241	1	1
10	13012020	199	1	0
...	...	...	...	...

[10] (a) Determine quantos blocos tem a *cache* e quantas palavras em memória podem ser mapeadas em cada entrada da *cache*.

Os endereços decompõem-se nos seguintes campos:



- byte *offset* – formado pelos 2 bits menos significativos, permite designar um dos 4 bytes de cada palavra;
- índice – formado pelos 4 bits que sucedem ao byte *offset*, permite representar os índices, de 0 a 15, dos blocos da *cache*;
- etiqueta – formado pelos 10 bits restantes, permite identificar que palavra de memória está em cada bloco da *cache*.

Portanto, o número de blocos é $2^4 = 16$.

O número de palavras em memória que podem ser mapeadas em cada entrada da cache é dado pelo quociente entre o número de palavras e o número de blocos da cache: $\frac{2^{16}/4}{2^4} = 2^{10} = 1024$.

Alternativamente, este número corresponde ao total de etiquetas possíveis com 10 bits: $2^{10} = 1024$.

[10] (b) Determine o conteúdo de memória nos endereços `0x9020` e `0x9024`.

$$\texttt{0x9020} = \underbrace{\texttt{1001 0000 00}}_{240_\text{H}}\underbrace{\texttt{10 00}}_{8}\texttt{00}$$

O conteúdo presente na entrada 8 da *cache* é válido (`v=1`) e a etiqueta (240_H) é igual à etiqueta do acesso atual. Além disso, como `d=0` significa que o conteúdo em memória é igual ao conteúdo atual em *cache*. Assim, conclui-se que o conteúdo de memória no endereço `0x9020` é `0x80000000`.

$$\texttt{0x9024} = \underbrace{\texttt{1001 0000 00}}_{240_\text{H}}\underbrace{\texttt{10 01}}_{9}\texttt{00}$$

O conteúdo presente na entrada 9 da *cache* é válido (`v=1`) mas a etiqueta (241_H) é diferente da etiqueta do acesso atual. Conclui-se assim que é impossível determinar o conteúdo de memória no endereço `0x9024`.

[10] (c) Mostre quantos acessos a memória ocorrem como consequência das seguintes operações:

- ler Mem[`0x7CDC`]
- escrever `0x12345678` em Mem[`0xE020`]

ler Mem[`0x7CDC`]:

$$\texttt{0x7CDC} = \underbrace{\texttt{0111 1100 11}}_{1F3_\text{H}}\underbrace{\texttt{01 11}}_{7}\texttt{00}$$

O conteúdo presente na entrada 7 da *cache* é válido (`v=1`) mas a etiqueta (241_H) é diferente da etiqueta do acesso atual. Perante esta situação de *read-miss*, como `d=1` é necessário fazer *write-back* (escrever conteúdo do bloco 7 da *cache* em memória) antes de concretizar a leitura de memória pretendida. Portanto, conclui-se que ocorrem dois acessos a memória.

escrever `0x12345678` em Mem[`0xE020`]:

$$\texttt{0xE020} = \underbrace{\texttt{1110 0000 00}}_{380_\text{H}}\underbrace{\texttt{10 00}}_{8}\texttt{00}$$

O conteúdo presente na entrada 8 da *cache* é válido (`v=1`) mas a etiqueta (240_H) é diferente da etiqueta do acesso atual. Perante esta situação de *write-miss*, como `d=0` não é necessário fazer *write-back* antes de atualizar o conteúdo da cache. É de notar que a escrita ocorre apenas na cache. Portanto, neste caso não há qualquer acesso a memória.

Conclui-se assim que após a realização das operações indicadas ocorreram 2 acessos a memória.

[10] (d) Assuma que a taxa de faltas da *cache* é 1/9 da taxa de acertos, a penalidade de falta é 80 ciclos e o CPI devido a protelamento no acesso a dados é 2. Nestas condições, determine a percentagem de instruções executadas que acedem a memória.

$\text{taxa}_\text{falta} = \frac{1}{9}\text{taxa}_\text{acerto} = \frac{1}{9}(100 - \text{taxa}_\text{falta})$ pelo que $\text{taxa}_\text{falta} = 10\,\%$

$\text{CPI}_\text{prot} = \text{taxa}_\text{falta} \times \text{N}_\text{acessos/inst} \times \text{p}_\text{f}$
$2 = 0{,}1 \times \text{N}_\text{acessos/inst} \times 80$
$\text{N}_\text{acessos/inst} = 0{,}25$

Percentagem de instruções que acedem a dados é 25 %.

(Teste continua na página seguinte.)

Nome: _____ **Nº de estudante:** _____

- Responder à questão 13 diretamenta na folha de enunciado e entregá-la.
- Preencher a grelha da página 8.

13. A sub-rotina `funcX` recebe um valor inteiro maior que zero e produz um resultado inteiro (também superior a zero). Todos os valores inteiros têm 64 bits (*doublewords).*

```
funcX:      mov   X1, 10
            mov   X2, X0
            sub   X0, X0, X0
L1:         cbz   X2, L2
            add   X0, X0, 1
            udiv  X2, X2, X1
            b     L1
L2:         ret
```

[10] (a) A sub-rotina é invocada com `X0=37`. Quantas vezes é executada a instrução `udiv`? Justificar.

A instrução `udiv` está incluída no ciclo das linhas 4–7. A execução desse ciclo termina quando o valor do registo `X2` chega a 0. Este registo tem inicialmente o valor do argumento. De cada vez, que a instrução `udiv` é executada, o conteúdo de `X2` é dividido por 10 (divisão inteira), ficando em `X2` o quociente da divisão.
Para o valor 37, a primeira execução altera `X2` para ter 3 (quociente da divisão de 37 por 10); a segunda iteração coloca `X2` a 0 (quociente da divisão de 3 por 10). Consequentemente, a instrução é executada 2 vezes.
A sub-rotina produz o valor de `X0`, cujo conteúdo é o número de iterações executadas, i.e, o número de dígitos da representação decimal do argumento.

[10] (b) É verdade que, para todos os valores do argumento pertencentes ao intervalo [1000; 9999], a sub-rotina produz sempre o mesmo resultado? Justificar.

A sub-rotina conta o número de dígitos da representação decimal do valor do argumento (o registo `X0` guarda a contagem). Todos os elementos do intervalo indicado têm 4 dígitos decimais. Portanto, a sub-rotina produz o valor 4 quando é invocada com qualquer um como argumento: a afirmação é verdadeira.

[10] (c) O fragmento de código abaixo deve aplicar a sub-rotina `funcX` a todos os elementos de uma sequência (endereço-base em `X20`, número de elementos em `X19`) e determinar o maior resultado (em `X21`). Completar o fragmento.

```
        mov     X21, 0
Lx:     __cbz__ X19, stop          // terminar?
        ldur    X0, [X20]          // obter elemento da sequência
        add     X20, X20, __8__
        sub     X19, X19, __1__
        __bl__  funcX
        cmp     X0, X21
        b.ls    Lx
        mov     __X21__, X0
        __b__   Lx
stop:   ...
```

Fim das questões

- Preencher a grelha com as respostas às questões 1 a 10.
- **Apenas as respostas indicadas na grelha são consideradas para efeitos de avaliação.**

♣	Questão									
Opção	1	2	3	4	5	6	7	8	9	10
A		×								
B	×						×			
C			×	×	×			×		
D						×			×	×

Pontos: ____ / 100

Fim.