



|   |   |                         |                    |
|---|---|-------------------------|--------------------|
|  | Tipo de Prova<br>Exame de recurso   | Ano letivo<br>2017/2018 | Data<br>xx-xx-2018 |
|   | Curso<br>Licenciatura em Engenharia Informática / Licenciatura em<br>Segurança Informática em Redes de Computadores | Hora<br>xx:xx           |                    |
|   | Unidade Curricular<br>Sistemas Digitais e Arquitetura de Computadores   | Duração<br>1H:30M       |                    |

#### Observações:

- Preencha todo o cabeçalho da folha(s) de teste: nome completo do estudante, número do estudante, data da realização da prova de avaliação, nome da unidade curricular e nome do curso de licenciatura.
- Os estudantes deverão colocar em cima da mesa onde irão realizar a prova de avaliação o seu cartão de estudante ou outro cartão que os identifique.
- Os estudantes deverão colocar em cima da mesa para a realização da prova de avaliação, salvo indicação em contrário pelo docente, apenas os seguintes materiais, caneta, lápis, borracha. Todo o restante material deverá ser colocado debaixo da mesa.
- Os estudantes não deverão sair da sala de exame sem terem assinado a folha de presenças no caso de um exame final ou de passar o cartão de estudante na máquina de registo de presenças no caso de um momento de avaliação que não exame final.
- Os estudantes só podem sair da sala ao fim de 30 minutos depois do início da prova.
- Caso um estudante queira desistir deverá escrever na folha de exame "Desisto" e colocar por baixo a sua assinatura.
- Apresente a resolução desta prova apenas na(s) folha(s) fornecida(s) para esse fim.
- Justifique convenientemente todas as respostas.
- Qualquer estudante que necessite de mais folhas de teste ou mais folhas de rascunho deverá solicitar as mesmas ao docente.
- Quando um estudante solicitar uma nova folha de teste não deverá esquecer-se de no cabeçalho atualizar o número de folhas de teste. Cada folha de teste é constituída por quatro páginas, assim o número de folhas é 1/1. Caso o estudante solicite nova folha de teste o número de folhas a indicar na primeira folha será 1/2 e na segunda folha 2/2. Para não haver engano na contagem este parâmetro do cabeçalho deve apenas ser preenchido aquando da conclusão da prova de avaliação.
- Não é permitido o uso de qualquer dispositivo eletrónico, tais como por exemplo, máquina de calcular, salvo indicação em contrário, dada pelo docente responsável da unidade curricular.
- Não é permitido o uso de qualquer documentação além da indicada/fornecida pelo docente.
- Na altura da entrega da prova pelo estudante, este deve entregar tudo o que lhe foi entregue pelo docente, folha de teste, folha de rascunho, enunciado, folhas de apoio, etc.

|   |   |                         |                    |
|---|---|-------------------------|--------------------|
|  | Tipo de Prova<br>Exame de recurso   | Ano letivo<br>2017/2018 | Data<br>xx-xx-2018 |
|   | Curso<br>Licenciatura em Engenharia Informática / Licenciatura em<br>Segurança Informática em Redes de Computadores | Hora<br>xx:xx           |                    |
|   | Unidade Curricular<br>Sistemas Digitais e Arquitetura de Computadores   | Duração<br>1H:30M       |                    |

## Grupo I


(Resposta correta = 100%; Resposta errada = - 25%; Resposta em branco = 0%)

|  |
|--|
| <b>1.</b>  |
| O código BCD natural é baseado nas primeiras: <ul style="list-style-type: none"> <li>a) Dezassex combinações do código binário</li> <li>b) Vinte combinações do código binário</li> <li>c) Cinco combinações do código binário</li> <li>d) Nenhuma das anteriores</li> </ul> |
| Cotação: 2 valores   |

|   |
|---|
| <b>2.</b>   |
| Os circuitos sequenciais são circuitos lógicos, segundo os quais, os valores das suas saídas: <ul style="list-style-type: none"> <li>a) Dependem exclusivamente dos valores apresentados nas entradas num determinado momento, mas também dos valores que já estavam presentes anteriormente nas suas saídas</li> <li>b) Dependem exclusivamente dos valores apresentados nas entradas num determinado momento e da sua constituição interna</li> <li>c) Dependem exclusivamente dos valores apresentados nas entradas num determinado momento, mas também dos valores que já estavam presentes anteriormente nas suas saídas e ainda da sua constituição interna</li> <li>d) Nenhuma das anteriores</li> </ul> |
| Cotação: 2 valores  |

|  |
|--|
| <b>3.</b>  |
| Os microprogramas são: <ul style="list-style-type: none"> <li>a) Os responsáveis apenas pela execução das diversas etapas de manipulação de dados</li> <li>b) Os responsáveis pela execução das diversas etapas de manipulação de dados e transferências entre registos</li> <li>c) Os responsáveis apenas pela execução das diversas etapas de transferência entre registos</li> <li>d) Nenhuma das anteriores</li> </ul> |
| Cotação: 2 valores   |

|                                 |
|---------------------------------|
| <b>4.</b>                       |
| O registo Instruction Register: |

|   |   |                         |                    |
|---|---|-------------------------|--------------------|
|  | Tipo de Prova<br>Exame de recurso   | Ano letivo<br>2017/2018 | Data<br>xx-xx-2018 |
|   | Curso<br>Licenciatura em Engenharia Informática / Licenciatura em<br>Segurança Informática em Redes de Computadores | Hora<br>xx:xx           |                    |
|   | Unidade Curricular<br>Sistemas Digitais e Arquitetura de Computadores   | Duração<br>1H:30M       |                    |

- a) Contém a instrução que está a ser executada pelo processador
- b) Contém o endereço da próxima instrução armazenada em memória a ser executada
- c) É um apontador para uma zona reservada da memória denominada Stack
- d) Nenhuma das anteriores

Cotação: 2 valores

**5.**

Uma interrupção externa:

- a) É uma exceção gerada por um programa
- b) É usada para transferir o controlo para o sistema operativo
- c) É gerada por um dispositivo I/O
- d) Nenhuma das anteriores

Cotação: 2 valores

**6.**

Barramentos:

- a) São o conjunto de fios condutores para colocar informação nesses mesmos fios
- b) São o conjunto de fios condutores e regras para colocar e retirar informação desses mesmos fios
- c) São o conjunto de fios condutores para colocar e retirar informação desses mesmos fios
- d) Nenhuma das anteriores

Cotação: 2 valores

**7.**


Num disco magnético, o tempo necessário para mover a cabeça de leitura/escrita (quando a cabeça de leitura é móvel) para a pista correta, denomina-se por:

- a) Seek Time
- b) Latência rotacional
- c) Tempo de transferência
- d) Nenhuma das anteriores

Cotação: 2 valores

**8.**

Na multiplicação de números binários, quando se executa repetidamente operações de *shift* e adição ou

|   |   |                         |                    |
|---|---|-------------------------|--------------------|
|  | Tipo de Prova<br>Exame de recurso   | Ano letivo<br>2017/2018 | Data<br>xx-xx-2018 |
|   | Curso<br>Licenciatura em Engenharia Informática / Licenciatura em<br>Segurança Informática em Redes de Computadores | Hora<br>xx:xx           |                    |
|   | Unidade Curricular<br>Sistemas Digitais e Arquitetura de Computadores   | Duração<br>1H:30M       |                    |

subtração, estamos a utilizar a técnica de:

- a) Deslocar e somar
- b) Adicionar repetidas vezes
- c) Algoritmo de Booth
- d) Nenhuma das anteriores

Cotação: 2 valores

## Grupo II

9.

Projete um circuito que implemente a seguinte função lógica  $F$ ,  $F = X.Y + Y.Z + X.Z$ , utilizando um multiplexer de quatro entradas de informação. (Mantenha a seguinte ordem das variáveis  $X$ ,  $Y$  e  $Z$ . Caso seja necessário descartar alguma variável, descarte a variável  $Z$ .)

Cotação: 2 valores

10.


Elabore um programa que coloque dois bytes em hexadecimal, A9H e 7BH, nos registos B e C, e calcule a respetiva soma. Se a soma for maior que 8 bits, ou seja, se produzir carry, então coloque o número 00H no porto PORT2, cujo endereço é 02H, e na posição de memória 1080H. Caso contrário, guarde a soma apenas na posição de memória 1080H.

Cotação: 2 valores

## Instruções do microprocessador da INTEL 8085

Nomenclatura:


|     | Legenda                             |
|-----|-------------------------------------|
| pr  | Par de registos: HL, BC, DE, SP, PC |
| reg | Registo: A, B, C, D, E, H, L        |

|   |   |                         |                    |
|---|---|-------------------------|--------------------|
|  | Tipo de Prova<br>Exame de recurso   | Ano letivo<br>2017/2018 | Data<br>xx-xx-2018 |
|   | Curso<br>Licenciatura em Engenharia Informática / Licenciatura em<br>Segurança Informática em Redes de Computadores | Hora<br>xx:xx           |                    |
|   | Unidade Curricular<br>Sistemas Digitais e Arquitetura de Computadores   | Duração<br>1H:30M       |                    |

|         |   |
|---------|---|
| M       | Posição de memória  |
| addr    | Endereço de 16 bits de uma posição de memória                               |
| x       | O bit do registo de flags é afetado   |
| byte    | Constante, ou expressão lógica/aritmética que representa um dado de 8 bits  |
| double  | Constante, ou expressão lógica/aritmética que representa um dado de 16 bits |
| [ ]     | Conteúdo do que se encontra dentro de parênteses retos                      |
| [ [ ] ] | Conteúdo do conteúdo do que se encontra dentro de parênteses retos          |
| CS      | Flag de carry   |
| label   | Endereço de uma posição de memória  |
| port    | Endereço de um dispositivo I/O  |

## Grupo de transferência de dados


| Instrução | Operandos | Status do registo de flags |    |   |   |   | Operação realizada  |
|-----------|-----------|----------------------------|----|---|---|---|---|
|           |           | CS                         | AC | Z | S | P |   |
| LDAX      | pr        |                            |    |   |   |   | $[A] \leftarrow [[pr]]$<br>Load A using implied addressing by BC (pr=B) or DE (pr=D)  |
| STAX      | pr        |                            |    |   |   |   | $[[pr]] \leftarrow [A]$<br>Store A using implied addressing by BC (pr=B) or DE (pr=D) |
| MOV       | r,M       |                            |    |   |   |   | $[r] \leftarrow [[HL]]$<br>Load any register using implied addressing by HL           |
| MOV       | M,r       |                            |    |   |   |   | $[[HL]] \leftarrow [r]$<br>Store any register using implied addressing by HL          |

|   |   |  |                         |                    |
|---|---|--|-------------------------|--------------------|
|  | Tipo de Prova<br>Exame de recurso   |  | Ano letivo<br>2017/2018 | Data<br>xx-xx-2018 |
|   | Curso<br>Licenciatura em Engenharia Informática / Licenciatura em<br>Segurança Informática em Redes de Computadores |  |                         | Hora<br>xx:xx      |
|   | Unidade Curricular<br>Sistemas Digitais e Arquitetura de Computadores   |  |                         | Duração<br>1H:30M  |


|      |           |  |  |  |  |  |   |
|------|-----------|--|--|--|--|--|---|
| LDA  | addr      |  |  |  |  |  | $[A] \leftarrow [addr]$<br>Load A using direct addressing   |
| STA  | addr      |  |  |  |  |  | $[addr] \leftarrow [A]$<br>Store A using direct addressing  |
| LHLD | addr      |  |  |  |  |  | $[L] \leftarrow [addr]$ and $[H] \leftarrow [addr+1]$<br>Load H and L registers using direct addressing         |
| SHLD | addr      |  |  |  |  |  | $[addr] \leftarrow [L]$ and $[addr+1] \leftarrow [H]$<br>Store H and L registers using direct addressing        |
| MOV  | r,r       |  |  |  |  |  | $[r] \leftarrow [r]$<br>Move any register to any register   |
| XCHG |           |  |  |  |  |  | $[D] \leftrightarrow [H]$ and $[E] \leftrightarrow [L]$<br>Exchange DE with HL                                  |
| SPHL |           |  |  |  |  |  | $[HL] \leftarrow [SP]$<br>Move HL to SP   |
| LXI  | pr,double |  |  |  |  |  | $[pr] \leftarrow \text{double}$<br>Load 16 bits immediate data into BC (pr=B), DE (pr=D), HL (pr=H), SP (pr=SP) |
| MVI  | M,byte    |  |  |  |  |  | $[[HL]] \leftarrow \text{byte}$<br>Load 8 bit immediate data into memory location with implied addressing by HL |
| MVI  | r,byte    |  |  |  |  |  | $[r] \leftarrow \text{byte}$<br>Load 8 bit immediate data into any register                                     |

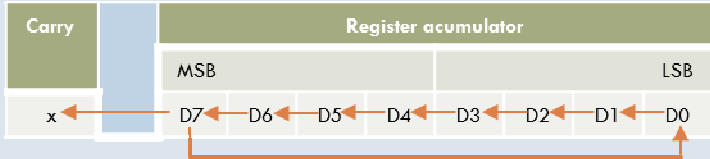
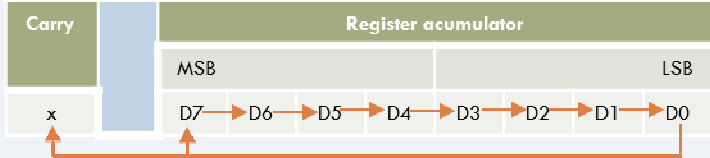
### Grupo aritmético, lógico e de rotação

| Instrução | Operandos | Status do registo de flags |    |   |   |   | Operação realizada  |
|-----------|-----------|----------------------------|----|---|---|---|---|
|           |           | CS                         | AC | Z | S | P |   |
| ADD       | M         | x                          | x  | x | x | x | $[A] \leftarrow [A] + [[HL]]$<br>Add register A with implied addressing by HL and store the result in register A                        |
| ADC       | M         | x                          | x  | x | x | x | $[A] \leftarrow [A] + [[HL]] + [CS]$<br>Add register A with carry with implied addressing by HL and store the result in register A      |
| SUB       | M         | x                          | x  | x | x | x | $[A] \leftarrow [A] - [[HL]]$<br>Subtract register A with implied addressing by HL and store the result in register A                   |
| SBB       | M         | x                          | x  | x | x | x | $[A] \leftarrow [A] - [[HL]] - [CS]$<br>Subtract register A with carry with implied addressing by HL and store the result in register A |


|   |   |  |                         |                    |
|---|---|--|-------------------------|--------------------|
|  | Tipo de Prova<br>Exame de recurso   |  | Ano letivo<br>2017/2018 | Data<br>xx-xx-2018 |
|   | Curso<br>Licenciatura em Engenharia Informática / Licenciatura em<br>Segurança Informática em Redes de Computadores |  |                         | Hora<br>xx:xx      |
|   | Unidade Curricular<br>Sistemas Digitais e Arquitetura de Computadores   |  |                         | Duração<br>1H:30M  |

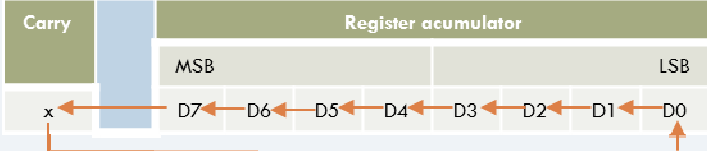
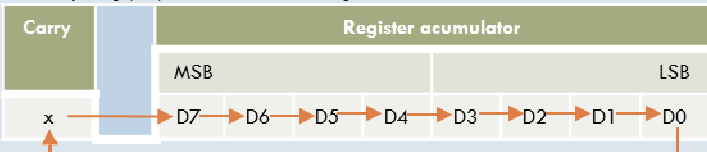
|     |      |   |   |   |   |   |   |
|-----|------|---|---|---|---|---|---|
| ANA | M    | 0 | 1 | x | x | x | $[A] \leftarrow [A] \text{ AND } [[HL]]$<br>AND between register A with implied addressing by HL and store the result in register A   |
| XRA | M    | 0 | 0 | x | x | x | $[A] \leftarrow [A] \text{ XOR } [[HL]]$<br>Exclusive-OR between register A with implied addressing by HL and store the result in register A  |
| ORA | M    | 0 | 0 | x | x | x | $[A] \leftarrow [A] \text{ OR } [[HL]]$<br>OR between register A with implied addressing by HL and store the result in register A   |
| CMP | M    | x | x | x | x | x | $[A] - [[HL]]$<br>Compare register A with implied addressing by HL<br>If register A < [[HL]] then the carry flag is set (1)<br>If register A = [[HL]] then the zero flag is set (1)<br>If register A > [[HL]] then the carry and zero flags are reset (0) |
| INR | M    | x | x | x | x | x | $[[HL]] \leftarrow [[HL]] + 1$<br>Increment memory  |
| DCR | M    | x | x | x | x | x | $[[HL]] \leftarrow [[HL]] - 1$<br>Decrement memory  |
| ADI | byte | x | x | x | x | x | $[A] \leftarrow [A] + \text{byte}$<br>Add register A with 8 bit immediate data and store the result in register A   |
| ACI | byte | x | x | x | x | x | $[A] \leftarrow [A] + \text{byte} + [CS]$<br>Add register A with 8 bit immediate data with carry and store the result in register A   |
| SUI | byte | x | x | x | x | x | $[A] \leftarrow [A] - \text{byte}$<br>Subtract register A with 8 bit immediate data and store the result in register A  |
| SBI | byte | x | x | x | x | x | $[A] \leftarrow [A] - \text{byte} - [CS]$<br>Subtract register A with 8 bit immediate data with carry and store the result in register A  |
| ANI | byte | 0 | 1 | x | x | x | $[A] \leftarrow [A] \text{ AND } \text{byte}$<br>AND between register A with 8 bit immediate data and store the result in register A  |
| XRI | byte | 0 | 0 | x | x | x | $[A] \leftarrow [A] \text{ XOR } \text{byte}$<br>Exclusive-OR between register A with 8 bit immediate data and store the result in register A   |
| ORI | byte | 0 | 0 | x | x | x | $[A] \leftarrow [A] \text{ OR } \text{byte}$<br>OR between register A with 8 bit immediate data and store the result in register A  |
| CPI | byte | x | x | x | x | x | $[A] - \text{byte}$<br>Compare register A with 8 bit immediate data<br>If register A < byte then the carry flag is set (1)<br>If register A = byte then the zero flag is set (1)<br>If register A > byte then the carry and zero flags are reset (0)      |
| ADD | r    | x | x | x | x | x | $[A] \leftarrow [A] + [r]$<br>Add register A with any register and store the result in register A   |
| ADC | r    | x | x | x | x | x | $[A] \leftarrow [A] + [r] + [CS]$<br>Add register A with any register with carry and store the result in register A   |

|   |   |  |                         |                    |
|---|---|--|-------------------------|--------------------|
|  | Tipo de Prova<br>Exame de recurso   |  | Ano letivo<br>2017/2018 | Data<br>xx-xx-2018 |
|   | Curso<br>Licenciatura em Engenharia Informática / Licenciatura em<br>Segurança Informática em Redes de Computadores |  |                         | Hora<br>xx:xx      |
|   | Unidade Curricular<br>Sistemas Digitais e Arquitetura de Computadores   |  |                         | Duração<br>1H:30M  |

|     |   |   |   |   |   |   |   |
|-----|---|---|---|---|---|---|---|
| SUB | r | x | x | x | x | x | $[A] \leftarrow [A] - [r]$<br>Subtract register A with any register and store the result in register A  |
| SBB | r | x | x | x | x | x | $[A] \leftarrow [A] - [r] - [CS]$<br>Subtract register A with any register with carry and store the result in register A  |
| ANA | r | 0 | 1 | x | x | x | $[A] \leftarrow [A] \text{ AND } [r]$<br>AND between register A with any register and store the result in register A  |
| XRA | r | 0 | 0 | x | x | x | $[A] \leftarrow [A] \text{ XOR } [r]$<br>Exclusive-OR between register A with any register and store the result in register A   |
| ORA | r | 0 | 0 | x | x | x | $[A] \leftarrow [A] \text{ OR } [r]$<br>OR between register A with any register and store the result in register A  |
| CMP | r | x | x | x | x | x | $[A] - [r]$<br>Compare register A with any register<br>If register A < r then the carry flag is set (1)<br>If register A = r then the zero flag is set (1)<br>If register A > r then the carry and zero flags are reset (0)   |
| INR | r |   | x | x | x | x | $[r] \leftarrow [r] + 1$<br>Increment any register  |
| DCR | r |   | x | x | x | x | $[r] \leftarrow [r] - 1$<br>Decrement any register  |
| CMA |   |   |   |   |   |   | $[A] \leftarrow \overline{[A]}$<br>Complement register A  |
| DAA |   | x | x | x | x | x | The contents of the accumulator are changed from a binary value to two 4-bit binary coded decimal (BCD) digits.<br>This is the only instruction that uses the auxiliary flag (AC) to perform the binary to BCD conversion, and the conversion procedure is described below.<br>If the value of the low-order 4-bits in the accumulator is greater than 9 or if AC flag is set (1), the instruction adds 6 to the low-order four bits.<br>If the value of the high-order 4-bits in the accumulator is greater than 9 or if the Carry flag (CS) is set (1), the instruction adds 6 to the high-order four bits. |
| RLC |   | x |   |   |   |   | Each binary bit of the register accumulator is rotated left by one position.<br>Bit D7 is placed in the position of D0 as well as in the Carry flag.<br>The carry flag (CS) is modified according to bit D7. <div data-bbox="683 1630 1396 1787">  </div>   |
| RRC |   | x |   |   |   |   | Each binary bit of the register accumulator is rotated right by one position.<br>Bit D0 is placed in the position of D7 as well as in the Carry flag.<br>The carry flag (CS) is modified according to bit D0. <div data-bbox="683 1881 1396 2038">  </div>  |




|   |   |                         |                    |
|---|---|-------------------------|--------------------|
|  | Tipo de Prova<br>Exame de recurso   | Ano letivo<br>2017/2018 | Data<br>xx-xx-2018 |
|   | Curso<br>Licenciatura em Engenharia Informática / Licenciatura em<br>Segurança Informática em Redes de Computadores | Hora<br>xx:xx           |                    |
|   | Unidade Curricular<br>Sistemas Digitais e Arquitetura de Computadores   | Duração<br>1H:30M       |                    |


|     |    |   |  |  |  |   |
|-----|----|---|--|--|--|---|
| RAL |    | X |  |  |  | <p>Each binary bit of the register acumulator is rotated left by one position trough the carry flag.<br/>Bit D<sub>7</sub> is placed in the carry flag, and the carry flag is placed in the least significant position D<sub>0</sub>.<br/>The carry flag (CS) is modified according to bit D<sub>7</sub>.</p>   |
| RAR |    | x |  |  |  | <p>Each binary bit of the register acumulator is rotated right by one position trough the carry flag.<br/>Bit D<sub>0</sub> is placed in the carry flag, and the carry flag is placed in the least significant position D<sub>7</sub>.<br/>The carry flag (CS) is modified according to bit D<sub>0</sub>.</p>  |
| DAD | pr | x |  |  |  | <p>[HL] ← [HL] + [pr]<br/>Add HL to a register pair BC (pr=B), DE (pr=D), HL (pr=H), SP (pr=SP) and store the result in HL</p>  |
| INX | pr |   |  |  |  | <p>[pr] ← [pr] + 1<br/>Increment register pair BC (pr=B), DE (pr=D), HL (pr=H), SP (pr=SP) and store the result in a register pair BC (pr=B), DE (pr=D), HL (pr=H), SP (pr=SP)</p>  |
| DCX | pr |   |  |  |  | <p>[pr] ← [pr] - 1<br/>Decrement register pair BC (pr=B), DE (pr=D), HL (pr=H), SP (pr=SP) and store the result in a register pair BC (pr=B), DE (pr=D), HL (pr=H), SP (pr=SP)</p>  |

## Grupo de controlo e de salto

| Instrução | Operandos | Status do registo de flags |    |   |   |   | Operação realizada   |
|-----------|-----------|----------------------------|----|---|---|---|--|
|           |           | CS                         | AC | Z | S | P |  |
| JMP       | label     |                            |    |   |   |   | [PC] ← label<br>Jump to instruction at address label   |
| PCHL      |           |                            |    |   |   |   | [PC] ← [HL]<br>Jump to instruction at address contained in HL  |
| CALL      | label     |                            |    |   |   |   | [[SP]] ← [PC], [PC] ← label, [SP] ← [SP] - 2<br>Jump to subroutine starting at address label                                   |
| CC        | label     |                            |    |   |   |   | [[SP]] ← [PC], [PC] ← label, [SP] ← [SP] - 2<br>Jump to subroutine starting at address label if the carry flag (CS) equal to 1 |
| CNC       | label     |                            |    |   |   |   | [[SP]] ← [PC], [PC] ← label, [SP] ← [SP] - 2<br>Jump to subroutine starting at address label if the carry flag (CS) equal to 0 |

|   |   |                         |                    |
|---|---|-------------------------|--------------------|
|  | Tipo de Prova<br>Exame de recurso   | Ano letivo<br>2017/2018 | Data<br>xx-xx-2018 |
|   | Curso<br>Licenciatura em Engenharia Informática / Licenciatura em<br>Segurança Informática em Redes de Computadores | Hora<br>xx:xx           |                    |
|   | Unidade Curricular<br>Sistemas Digitais e Arquitetura de Computadores   | Duração<br>1H:30M       |                    |


|     |       |  |  |  |  |  |   |
|-----|-------|--|--|--|--|--|---|
| CZ  | label |  |  |  |  |  | [[SP]] ← [PC] , [PC] ← label, [SP] ← [SP] – 2<br>Jump to subroutine starting at address label if the zero flag (Z) equal to 1   |
| CNZ | label |  |  |  |  |  | [[SP]] ← [PC] , [PC] ← label, [SP] ← [SP] – 2<br>Jump to subroutine starting at address label if the zero flag (Z) equal to 0   |
| CP  | label |  |  |  |  |  | [[SP]] ← [PC] , [PC] ← label, [SP] ← [SP] – 2<br>Jump to subroutine starting at address label if the sign flag (S) equal to 0   |
| CM  | label |  |  |  |  |  | [[SP]] ← [PC] , [PC] ← label, [SP] ← [SP] – 2<br>Jump to subroutine starting at address label if the sign flag (S) equal to 1   |
| CPE | label |  |  |  |  |  | [[SP]] ← [PC] , [PC] ← label, [SP] ← [SP] – 2<br>Jump to subroutine starting at address label if the parity flag (P) equal to 1 |
| CPO | label |  |  |  |  |  | [[SP]] ← [PC] , [PC] ← label, [SP] ← [SP] – 2<br>Jump to subroutine starting at address label if the parity flag (P) equal to 0 |
| RET |       |  |  |  |  |  | [PC] ← [[SP]] , [SP] ← [SP] + 2<br>Return from subroutine   |
| RC  |       |  |  |  |  |  | [PC] ← [[SP]] , [SP] ← [SP] + 2<br>Return from subroutine if the carry flag (CS) equal to 1                                     |
| RNC |       |  |  |  |  |  | [PC] ← [[SP]] , [SP] ← [SP] + 2<br>Return from subroutine if the carry flag (CS) equal to 0                                     |
| RZ  |       |  |  |  |  |  | [PC] ← [[SP]] , [SP] ← [SP] + 2<br>Return from subroutine if the zero flag (Z) equal to 1                                       |
| RNZ |       |  |  |  |  |  | [PC] ← [[SP]] , [SP] ← [SP] + 2<br>Return from subroutine if the zero flag (Z) equal to 0                                       |
| RM  |       |  |  |  |  |  | [PC] ← [[SP]] , [SP] ← [SP] + 2<br>Return from subroutine if the sign flag (S) equal to 0                                       |
| RP  |       |  |  |  |  |  | [PC] ← [[SP]] , [SP] ← [SP] + 2<br>Return from subroutine if the sign flag (S) equal to 1                                       |
| RPE |       |  |  |  |  |  | [PC] ← [[SP]] , [SP] ← [SP] + 2<br>Return from subroutine if the parity flag (P) equal to 1                                     |
| RPO |       |  |  |  |  |  | [PC] ← [[SP]] , [SP] ← [SP] + 2<br>Return from subroutine if the parity flag (P) equal to 0                                     |
| JC  | label |  |  |  |  |  | [PC] ← label<br>Jump to instruction at address label if the carry flag (CS) equal to 1  |
| JNC | label |  |  |  |  |  | [PC] ← label<br>Jump to instruction at address label if the carry flag (CS) equal to 0  |

|   |   |                         |                    |
|---|---|-------------------------|--------------------|
|  | Tipo de Prova<br>Exame de recurso   | Ano letivo<br>2017/2018 | Data<br>xx-xx-2018 |
|   | Curso<br>Licenciatura em Engenharia Informática / Licenciatura em<br>Segurança Informática em Redes de Computadores | Hora<br>xx:xx           |                    |
|   | Unidade Curricular<br>Sistemas Digitais e Arquitetura de Computadores   | Duração<br>1H:30M       |                    |

| JZ          | label   |         |  |  |  |  | [PC] ← label<br>Jump to instruction at address label if the zero flag (Z) equal to 1  |             |         |         |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |
|-------------|---------|---------|--|--|--|--|---|-------------|---------|---------|-----|---|-------|-----|---|-------|-----|---|-------|-----|---|-------|-----|---|-------|-----|---|-------|-----|---|-------|-----|---|-------|
| JNZ         | label   |         |  |  |  |  | [PC] ← label<br>Jump to instruction at address label if the zero flag (Z) equal to 0  |             |         |         |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |
| JP          | label   |         |  |  |  |  | [PC] ← label<br>Jump to instruction at address label if the sign flag (S) equal to 0  |             |         |         |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |
| JM          | label   |         |  |  |  |  | [PC] ← label<br>Jump to instruction at address label if the sign flag (S) equal to 1  |             |         |         |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |
| JPE         | label   |         |  |  |  |  | [PC] ← label<br>Jump to instruction at address label if the parity flag (P) equal to 1  |             |         |         |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |
| JPO         | label   |         |  |  |  |  | [PC] ← label<br>Jump to instruction at address label if the parity flag (P) equal to 0  |             |         |         |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |
| RST         | n       |         |  |  |  |  | <p>The RST instruction is equivalent to a 1-byte call instruction to one of eight memory locations depending upon the number. The instructions are generally used in conjunction with interrupts and inserted using external hardware. However these can be used as software instructions in a program to transfer program execution to one of the eight locations. The addresses are:</p> <table><thead><tr><th>Instruction</th><th>Restart</th><th>Address</th></tr></thead><tbody><tr><td>RST</td><td>0</td><td>0000H</td></tr><tr><td>RST</td><td>1</td><td>0008H</td></tr><tr><td>RST</td><td>2</td><td>0010H</td></tr><tr><td>RST</td><td>3</td><td>0018H</td></tr><tr><td>RST</td><td>4</td><td>0020H</td></tr><tr><td>RST</td><td>5</td><td>0028H</td></tr><tr><td>RST</td><td>6</td><td>0030H</td></tr><tr><td>RST</td><td>7</td><td>0038H</td></tr></tbody></table> | Instruction | Restart | Address | RST | 0 | 0000H | RST | 1 | 0008H | RST | 2 | 0010H | RST | 3 | 0018H | RST | 4 | 0020H | RST | 5 | 0028H | RST | 6 | 0030H | RST | 7 | 0038H |
| Instruction | Restart | Address |  |  |  |  |   |             |         |         |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |
| RST         | 0       | 0000H   |  |  |  |  |   |             |         |         |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |
| RST         | 1       | 0008H   |  |  |  |  |   |             |         |         |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |
| RST         | 2       | 0010H   |  |  |  |  |   |             |         |         |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |
| RST         | 3       | 0018H   |  |  |  |  |   |             |         |         |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |
| RST         | 4       | 0020H   |  |  |  |  |   |             |         |         |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |
| RST         | 5       | 0028H   |  |  |  |  |   |             |         |         |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |
| RST         | 6       | 0030H   |  |  |  |  |   |             |         |         |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |
| RST         | 7       | 0038H   |  |  |  |  |   |             |         |         |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |     |   |       |

### Grupo de controlo do CPU, I/O e da Pilha

| Instrução | Operandos | Status do registo de flags |    |   |   |   | Operação realizada   |
|-----------|-----------|----------------------------|----|---|---|---|--|
|           |           | CS                         | AC | Z | S | P |  |
| IN        | port      |                            |    |   |   |   | [A] ← [port]<br>Input to register acumulator (A) from I/O port   |
| OUT       | port      |                            |    |   |   |   | [port] ← [A]<br>Output from register acumulator (A) to I/O port  |
| PUSH      | pr        |                            |    |   |   |   | [[SP]] ← [pr], [SP] ← [SP] - 2<br>Push register pair BC (pr=B), DE (pr=D), H (pr=HL), PSW (pr=PSW) contents onto stack |
| POP       | pr        |                            |    |   |   |   | [pr] ← [[SP]], [SP] ← [SP] + 2<br>Pop stack into register pair BC (pr=B), DE (pr=D), H (pr=HL), PSW (pr=PSW)           |
| XTHL      |           |                            |    |   |   |   | [HL] ← [[SP]]<br>Exchange HL with top of stack   |

|   |   |                         |                    |
|---|---|-------------------------|--------------------|
|  | Tipo de Prova<br>Exame de recurso   | Ano letivo<br>2017/2018 | Data<br>xx-xx-2018 |
|   | Curso<br>Licenciatura em Engenharia Informática / Licenciatura em<br>Segurança Informática em Redes de Computadores | Hora<br>xx:xx           |                    |
|   | Unidade Curricular<br>Sistemas Digitais e Arquitetura de Computadores   | Duração<br>1H:30M       |                    |

|     |  |  |  |  |  |  |
|-----|--|--|--|--|--|--|
| EI  |  |  |  |  |  | Enable interrupts following execution of next instruction                          |
| DI  |  |  |  |  |  | Disable interrupts   |
| SIM |  |  |  |  |  | Set interrupt mask   |
| RIM |  |  |  |  |  | Read interrupt mask  |
| NOP |  |  |  |  |  | $[PC] \leftarrow [PC] + 1$<br>No operation but program counter (PC) is incremented |
| HLT |  |  |  |  |  | HALT<br>Stop CPU operation   |