
	Tipo de Prova Teste 2	Ano letivo 2017/2018	Data 29-05-2018
	Curso Licenciatura em Engenharia Informática / Licenciatura em Segurança Informática em Redes de Computadores	Hora 14:00	
	Unidade Curricular Sistemas Digitais e Arquitetura de Computadores	Duração 30Min	

#### 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 Teste 2	Ano letivo 2017/2018	Data 29-05-2018
	Curso Licenciatura em Engenharia Informática / Licenciatura em Segurança Informática em Redes de Computadores	Hora 14:00	
	Unidade Curricular Sistemas Digitais e Arquitetura de Computadores	Duração 30Min	

### Grupo I


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

<b>1.</b>
Os programas desenvolvidos em linguagem de alto nível, têm de ser traduzidos para uma linguagem mais simples denominada:
<ul style="list-style-type: none"> <li>a) Linguagem Assembler</li> <li>b) Linguagem Assembly</li> <li>c) Linguagem microprogramação</li> <li>d) Nenhuma das anteriores</li> </ul>
Cotação: 2 valores

<b>2.</b>
A comunicação entre a unidade de processamento e a unidade de controlo do CPU é executada:
<ul style="list-style-type: none"> <li>a) Através de bits de estado</li> <li>b) Através de palavras de controlo</li> <li>c) Através de palavras de estado</li> <li>d) Nenhuma das anteriores</li> </ul>
Cotação: 2 valores

<b>3.</b>
Uma interrupção Interna:
<ul style="list-style-type: none"> <li>a) É gerada por um dispositivo I/O</li> <li>b) É usada para transferir o controlo para o sistema operativo</li> <li>c) É uma exceção gerada por um programa</li> <li>d) Nenhuma das anteriores</li> </ul>
Cotação: 2 valores

<b>4.</b>
Quando ocorre uma interrupção, é necessário guardar um conjunto de informação. Essa informação é guardada:
<ul style="list-style-type: none"> <li>a) Na memória</li> <li>b) Na Stack</li> <li>c) No CPU</li> <li>d) Nenhuma das anteriores</li> </ul>

	Tipo de Prova Teste 2	Ano letivo 2017/2018	Data 29-05-2018
	Curso Licenciatura em Engenharia Informática / Licenciatura em Segurança Informática em Redes de Computadores	Hora 14:00	
	Unidade Curricular Sistemas Digitais e Arquitetura de Computadores	Duração 30Min	

Cotação: 2 valores

**5.**

Num barramento dedicado:

- a) Os mesmos fios condutores são usados para endereços ou dados
- b) Existem fios condutores para dados e endereços em separado
- c) Existem apenas fios condutores para dados
- d) Nenhuma das anteriores

Cotação: 2 valores

**6.**

Nos barramentos em que cada módulo do sistema contém uma lógica de controlo de acesso e os módulos agem de forma conjunta para partilhar o barramento, são denominados por barramentos:

- a) Centralizados
- b) Normalizados
- c) Descentralizados
- d) Nenhuma das anteriores

Cotação: 2 valores

**7.**

Uma memória que não perde a informação quando é desligada corrente elétrica denomina-se por:

- a) Memória não volátil
- b) Memória volátil
- c) Memória apagável
- d) Nenhuma das anteriores


Cotação: 2 valores

**8.**

Uma cache que utilize mapeamento associativo:

- a) Cada bloco da memória principal é mapeado numa única linha da cache
- b) Um bloco da memória principal é mapeado em qualquer linha de um determinado conjunto
- c) Um bloco da memória principal pode ser mapeado em qualquer linha da cache
- d) Nenhuma das anteriores

Cotação: 2 valores

	Tipo de Prova Teste 2	Ano letivo 2017/2018	Data 29-05-2018
	Curso Licenciatura em Engenharia Informática / Licenciatura em Segurança Informática em Redes de Computadores	Hora 14:00	
	Unidade Curricular Sistemas Digitais e Arquitetura de Computadores	Duração 30Min	

### Grupo II


<b>9.</b>
Elabore um programa que subtraia os conteúdos de três posições de memória consecutivas (1038H, 1039H e 1040H), guarde o resultado da subtração na posição de memória anterior à ocupada pelas parcelas a adicionar.
Cotação: 2 valores

<b>10.</b>
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
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

	Tipo de Prova Teste 2	Ano letivo 2017/2018	Data 29-05-2018
	Curso Licenciatura em Engenharia Informática / Licenciatura em Segurança Informática em Redes de Computadores	Hora 14:00	
	Unidade Curricular Sistemas Digitais e Arquitetura de Computadores	Duração 30Min	

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						$[A] \leftarrow [[pr]]$ Load A using implied addressing by BC (pr=B) or DE (pr=D)
STAX	pr						$[[pr]] \leftarrow [A]$ Store A using implied addressing by BC (pr=B) or DE (pr=D)
MOV	r,M						$[r] \leftarrow [[HL]]$ Load any register using implied addressing by HL
MOV	M,r						$[[HL]] \leftarrow [r]$ Store any register using implied addressing by HL
LDA	addr						$[A] \leftarrow [addr]$ Load A using direct addressing
STA	addr						$[addr] \leftarrow [A]$ Store A using direct addressing
LHLD	addr						$[L] \leftarrow [addr]$ and $[H] \leftarrow [addr+1]$ Load H and L registers using direct addressing
SHLD	addr						$[addr] \leftarrow [L]$ and $[addr+1] \leftarrow [H]$ Store H and L registers using direct addressing

	Tipo de Prova Teste 2		Ano letivo 2017/2018	Data 29-05-2018
	Curso Licenciatura em Engenharia Informática / Licenciatura em Segurança Informática em Redes de Computadores			Hora 14:00
	Unidade Curricular Sistemas Digitais e Arquitetura de Computadores			Duração 30Min


MOV	r,r						$[r] \leftarrow [r]$ Move any register to any register
XCHG							$[D] \leftrightarrow [H]$ and $[E] \leftrightarrow [L]$ Exchange DE with HL
SPHL							$[HL] \leftarrow [SP]$ Move HL to SP
LXI	pr,double						$[pr] \leftarrow \text{double}$ Load 16 bits immediate data into BC (pr=B), DE (pr=D), HL (pr=H), SP (pr=SP)
MVI	M,byte						$[[HL]] \leftarrow \text{byte}$ Load 8 bit immediate data into memory location with implied addressing by HL
MVI	r,byte						$[r] \leftarrow \text{byte}$ Load 8 bit immediate data into any register

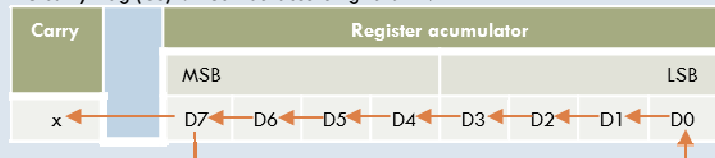
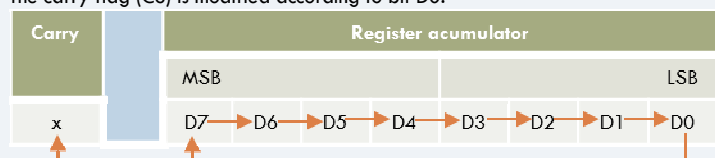
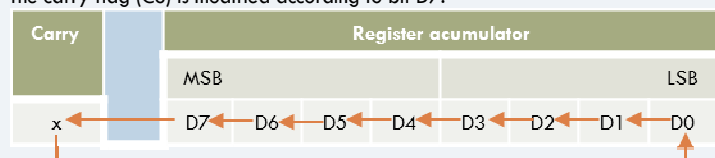
## Grupo aritmético, lógico e de rotação

Instrução	Operandos	Status do registo de flags					Operação realizada
		CS	AC	Z	S	P	
ADD	M	x	x	x	x	x	$[A] \leftarrow [A] + [[HL]]$ Add register A with implied addressing by HL and store the result in register A
ADC	M	x	x	x	x	x	$[A] \leftarrow [A] + [[HL]] + [CS]$ Add register A with carry with implied addressing by HL and store the result in register A
SUB	M	x	x	x	x	x	$[A] \leftarrow [A] - [[HL]]$ Subtract register A with implied addressing by HL and store the result in register A
SBB	M	x	x	x	x	x	$[A] \leftarrow [A] - [[HL]] - [CS]$ Subtract register A with carry with implied addressing by HL and store the result in register A
ANA	M	0	1	x	x	x	$[A] \leftarrow [A] \text{ AND } [[HL]]$ AND between register A with implied addressing by HL and store the result in register A
XRA	M	0	0	x	x	x	$[A] \leftarrow [A] \text{ XOR } [[HL]]$ Exclusive-OR between register A with implied addressing by HL and store the result in register A
ORA	M	0	0	x	x	x	$[A] \leftarrow [A] \text{ OR } [[HL]]$ OR between register A with implied addressing by HL and store the result in register A
CMP	M	x	x	x	x	x	$[A] - [[HL]]$ Compare register A with implied addressing by HL If register A < [[HL]] then the carry flag is set (1) If register A = [[HL]] then the zero flag is set (1)


	Tipo de Prova Teste 2	Ano letivo 2017/2018	Data 29-05-2018
	Curso Licenciatura em Engenharia Informática / Licenciatura em Segurança Informática em Redes de Computadores	Hora 14:00	
	Unidade Curricular Sistemas Digitais e Arquitetura de Computadores	Duração 30Min	

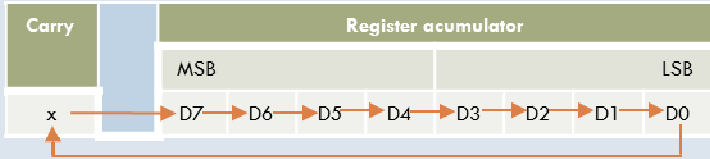
							If register A > [[HL]] then the carry and zero flags are reset (0)
INR	M	x	x	x	x	x	$[[HL]] \leftarrow [[HL]] + 1$ Increment memory
DCR	M	x	x	x	x	x	$[[HL]] \leftarrow [[HL]] - 1$ Decrement memory
ADI	byte	x	x	x	x	x	$[A] \leftarrow [A] + \text{byte}$ Add register A with 8 bit immediate data and store the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

	Tipo de Prova Teste 2		Ano letivo 2017/2018	Data 29-05-2018
	Curso Licenciatura em Engenharia Informática / Licenciatura em Segurança Informática em Redes de Computadores			Hora 14:00
	Unidade Curricular Sistemas Digitais e Arquitetura de Computadores			Duração 30Min

XRA	r	0	0	x	x	x	$[A] \leftarrow [A] \text{ XOR } [r]$ Exclusive-OR between register A with any register and store the result in register A
ORA	r	0	0	x	x	x	$[A] \leftarrow [A] \text{ OR } [r]$ OR between register A with any register and store the result in register A
CMP	r	x	x	x	x	x	$[A] - [r]$ Compare register A with any register If register A < r then the carry flag is set (1) If register A = r then the zero flag is set (1) If register A > r then the carry and zero flags are reset (0)
INR	r		x	x	x	x	$[r] \leftarrow [r] + 1$ Increment any register
DCR	r		x	x	x	x	$[r] \leftarrow [r] - 1$ Decrement any register
CMA							$[A] \leftarrow \overline{[A]}$ 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. 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. 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. 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. Bit D7 is placed in the position of D0 as well as in the Carry flag. The carry flag (CS) is modified according to bit D7. <div data-bbox="678 1321 1396 1478">  </div>
RRC		x					Each binary bit of the register accumulator is rotated right by one position. Bit D0 is placed in the position of D7 as well as in the Carry flag. The carry flag (CS) is modified according to bit D0. <div data-bbox="678 1568 1396 1724">  </div>
RAL		x					Each binary bit of the register accumulator is rotated left by one position through the carry flag. Bit D7 is placed in the carry flag, and the carry flag is placed in the least significant position D0. The carry flag (CS) is modified according to bit D7. <div data-bbox="678 1870 1396 2027">  </div>




	Tipo de Prova Teste 2	Ano letivo 2017/2018	Data 29-05-2018
	Curso Licenciatura em Engenharia Informática / Licenciatura em Segurança Informática em Redes de Computadores	Hora 14:00	
	Unidade Curricular Sistemas Digitais e Arquitetura de Computadores	Duração 30Min	


RAR		x					<p>Each binary bit of the register accumulator is rotated right by one position through the carry flag. 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>. The carry flag (CS) is modified according to bit D<sub>0</sub>.</p> 
DAD	pr	x					$[HL] \leftarrow [HL] + [pr]$ Add HL to a register pair BC (pr=B), DE (pr=D), HL (pr=H), SP (pr=SP) and store the result in HL
INX	pr						$[pr] \leftarrow [pr] + 1$ 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)
DCX	pr						$[pr] \leftarrow [pr] - 1$ 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)

## Grupo de controlo e de salto

Instrução	Operandos	Status do registo de flags					Operação realizada
		CS	AC	Z	S	P	
JMP	label						$[PC] \leftarrow \text{label}$ Jump to instruction at address label
PCHL							$[PC] \leftarrow [HL]$ Jump to instruction at address contained in HL
CALL	label						$[[SP]] \leftarrow [PC], [PC] \leftarrow \text{label}, [SP] \leftarrow [SP] - 2$ Jump to subroutine starting at address label
CC	label						$[[SP]] \leftarrow [PC], [PC] \leftarrow \text{label}, [SP] \leftarrow [SP] - 2$ Jump to subroutine starting at address label if the carry flag (CS) equal to 1
CNC	label						$[[SP]] \leftarrow [PC], [PC] \leftarrow \text{label}, [SP] \leftarrow [SP] - 2$ Jump to subroutine starting at address label if the carry flag (CS) equal to 0
CZ	label						$[[SP]] \leftarrow [PC], [PC] \leftarrow \text{label}, [SP] \leftarrow [SP] - 2$ Jump to subroutine starting at address label if the zero flag (Z) equal to 1
CNZ	label						$[[SP]] \leftarrow [PC], [PC] \leftarrow \text{label}, [SP] \leftarrow [SP] - 2$ Jump to subroutine starting at address label if the zero flag (Z) equal to 0
CP	label						$[[SP]] \leftarrow [PC], [PC] \leftarrow \text{label}, [SP] \leftarrow [SP] - 2$ Jump to subroutine starting at address label if the sign flag (S) equal to 0

	Tipo de Prova Teste 2	Ano letivo 2017/2018	Data 29-05-2018
	Curso Licenciatura em Engenharia Informática / Licenciatura em Segurança Informática em Redes de Computadores	Hora 14:00	
	Unidade Curricular Sistemas Digitais e Arquitetura de Computadores	Duração 30Min	


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

	Tipo de Prova Teste 2	Ano letivo 2017/2018	Data 29-05-2018
	Curso Licenciatura em Engenharia Informática / Licenciatura em Segurança Informática em Redes de Computadores	Hora 14:00	
	Unidade Curricular Sistemas Digitais e Arquitetura de Computadores	Duração 30Min	

JM	label						[PC] ← label Jump to instruction at address label if the sign flag (S) equal to 1																								
JPE	label						[PC] ← label Jump to instruction at address label if the parity flag (P) equal to 1																								
JPO	label						[PC] ← label 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> <p>Instruction Restart Address</p> <table><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></table>	RST	0	0000H	RST	1	0008H	RST	2	0010H	RST	3	0018H	RST	4	0020H	RST	5	0028H	RST	6	0030H	RST	7	0038H
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] Input to register acumulator (A) from I/O port
OUT	port						[port] ← [A] Ouput from register acumulator (A) to I/O port
PUSH	pr						[[SP]] ← [pr] , [SP] ← [SP] – 2 Push register pair BC (pr=B), DE (pr =D), H (pr=HL), PSW (pr=PSW) contentes onto stack
POP	pr						[pr] ← [[SP]] , [SP] ← [SP] + 2 Pop stack into register pair BC (pr=B), DE (pr =D), H (pr=HL), PSW (pr=PSW)
XTHL							[HL] ← [[SP]] Exchange HL with top of stack
EI							Enable interrupts following execution of next instruction
DI							Disable interrupts
SIM							Set interrupt mask

	Tipo de Prova Teste 2	Ano letivo 2017/2018	Data 29-05-2018
	Curso Licenciatura em Engenharia Informática / Licenciatura em Segurança Informática em Redes de Computadores	Hora 14:00	
	Unidade Curricular Sistemas Digitais e Arquitetura de Computadores	Duração 30Min	

RIM							Read interrupt mask
NOP							$[PC] \leftarrow [PC] + 1$ No operation but program counter (PC) is incremented
HLT							HALT Stop CPU operation