NOME:_____ **TURMA** _____

FEUP Universidade do Porto Faculdade de Engenharia

Departamento de Engenharia Electrotécnica e de Computadores

Sistemas Digitais (2000/2001) Correcção 2ª chamada – 25/Janeiro/2001

Duração: 2horas , sem consulta.

Antes de iniciar a prova, tenha em atenção as seguintes recomendações:

- Leia atentamente toda a prova antes de a iniciar.
- Mostre e justifique adequadamente todos os passos das suas respostas.
- A prova deverá ser resolvida no enunciado. Se necessário, utilize o verso para continuar a sua resolução.
- Assine todas as folhas que entregar, indicando em cada uma o número de páginas/folhas que entregou.

1 - Considere $X=11010011_2$ e $Y=E7_{16}$ que representam números inteiros com sinal em complemento para dois com 8 bits.

a) Diga, justificando, se pode ocorrer *overflow* na adição de X com Y.

Na adição de dois números representados em complemento para dois, só pode ocorrer* overflow *se os dois números tiverem o mesmo sinal. Como o número* $X=11010011$ *é negativo e o número* $Y=E7_{16}=11100111_2$ *também é negativo, então pode ocorrer* overflow *na soma desses números.

b) Efectue a adição de X com Y em binário, e indique se ocorre ou não *overflow*.

$$\begin{array}{r} 11010011 \\ +11100111 \\ \hline 110111010 \end{array}$$

O resultado da adição de* X *com* Y *é o número* 10111010*. Como ambos os operandos são negativos e o resultado também é negativo, pode-se concluir que não ocorreu* overflow *e o resultado (correcto!) da adição de X com Y é 10111010.

c) Qual é o maior número negativo representado em complemento para 2 com 8 bits que adicionado ao número X provoca *overflow*? Justifique.

Como o número* X *é negativo, só pode ocorrer* overflow *quando* X *é adicionado com outro número negativo, dando um resultado que é menor (mais negativo) do que o mais negativo que pode ser representado. Como o número mais negativo que pode ser representado em complemento para dois com 8 bits é* $-2^{(8-1)}=-128=10000000_2$*, então o maior número negativo* M *que adicionado com* X *<u>ainda não provoca overflow</u> será dado por* $X+M=(-128)$ *ou* $M=-128-X$*:

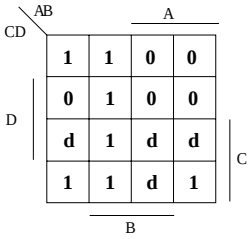
$$\begin{array}{rl} 10000000 & (-128) \\ -11010011 & (-45) \\ \hline 1\ 10101101 & (-83) \end{array}$$

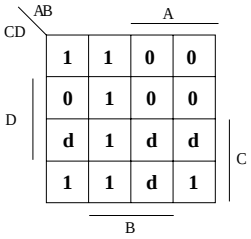
Logo, o número pedido será obtido subtraindo uma unidade a M:

$$M-1=-83-1=-84 = 10101100$$

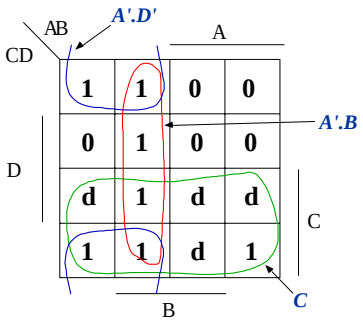
NOME:_____ **TURMA** _____

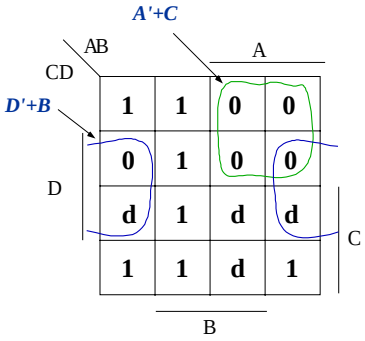
2 Considere a função booleana F(A,B,C,D) representada no seguinte mapa de *Karnaugh*, onde os termos indiferentes (*don't care*) estão representados por **d**.





a) Escreva as expressões simplificadas na forma de soma-de-produtos e produto-de-somas para a função F(A,B,C,D) (utilize um mapa de Karnaugh para obter cada expressão).

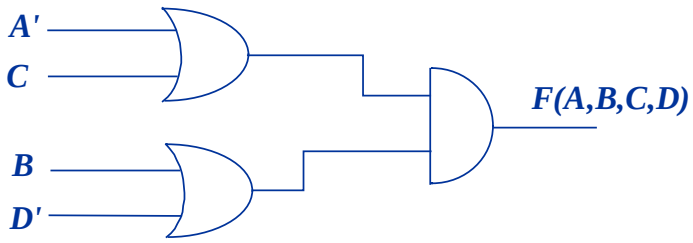




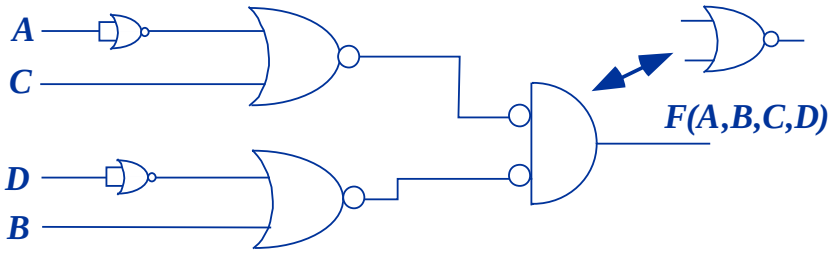
Expressão mínima soma-de-produtos: `F(A,B,C,D) = A'.D' + A'.B + C`
Expressão mínima produto-de-somas: `F(A,B,C,D) = (A'+C).(D'+B)`

b) Desenhe um circuito lógico que realize a função F(A,B,C,D) utilizando um número mínimo de portas lógicas do tipo NAND de duas entradas, ou de portas lógicas do tipo NOR de duas entradas.

Pelas expressões mínimas obtidas na alínea anterior, pode-se concluir que o circuito mais simples com portas NAND ou NOR resultará da expressão mínima produto-de-somas. O circuito OR-AND correspondente a essa expressão é:



Transformando-o de forma a conter apenas portas lógicas do tipo NOR com duas entradas:



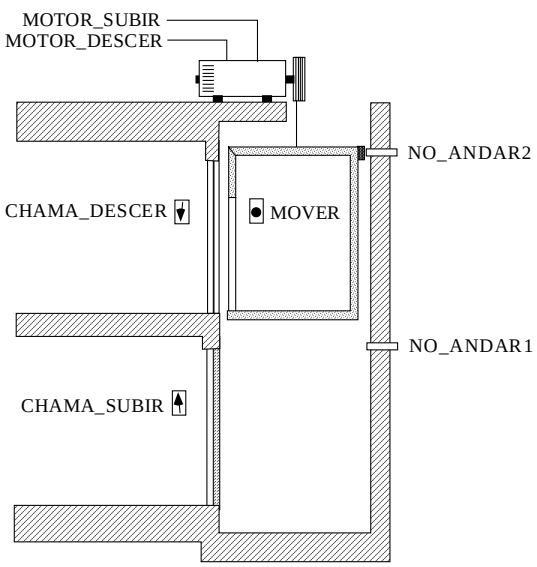
NOME:_____ **TURMA** _____

3 – Pretende-se projectar o sistema de controlo de um monta-cargas que se desloca entre dois andares (ver figura). Para controlar o monta-cargas dispõe-se das seguintes entradas para o sistema de controlo:

- um botão no interior do monta-cargas (MOVER) que é activado para deslocar o monta-cargas para o outro andar.
- dois botões exteriores de chamada, um em cada andar (CHAMA_DESCER e CHAMA_SUBIR), que são activados quando se pretende deslocar o monta-cargas para o andar respectivo.
- dois sensores (NO_ANDAR1 e NO_ANDAR2) que são activados sempre que o monta-cargas está correctamente posicionado no andar respectivo.

e das saídas do sistema de controlo:

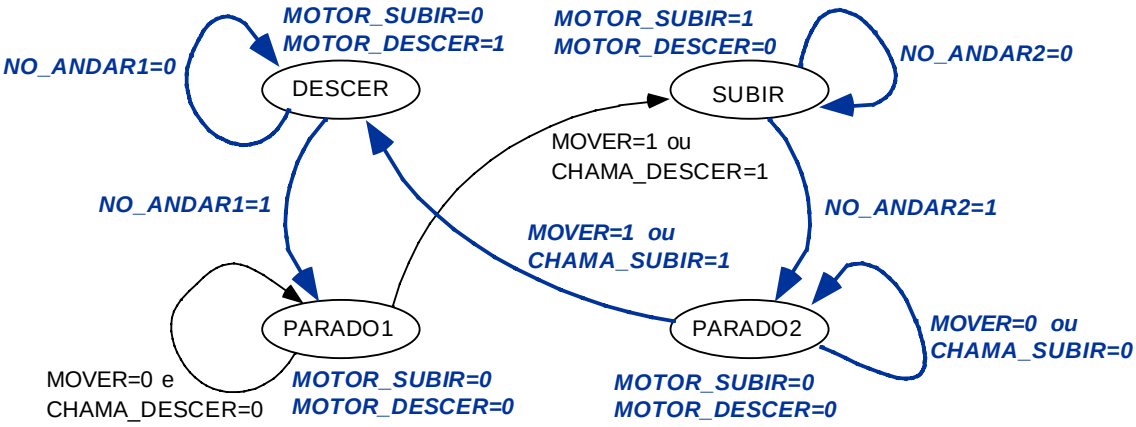
- MOTOR_SUBIR e MOTOR_DESCER que quando activadas provocam o movimento do monta-cargas no sentido respectivo



Admita que o sistema de controlo só aceita comandos provenientes dos botões quando o monta-cargas está parado num dos andares.

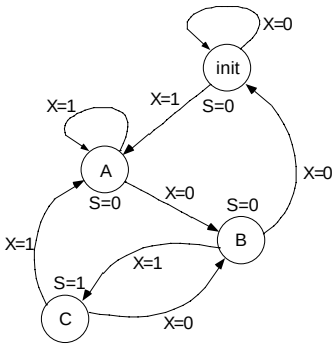
Complete o diagrama de transição de estados do sistema descrito, utilizando apenas os estados já representados e os nomes simbólicos referidos no texto para as entradas e saídas.

NOTA: O diagrama de estados incompleto apresentado no enunciado original tinha um erro: as duas transições de estado apresentadas tinham associada a condição* CHAMA_SUBIR=0 *e* CHAMA_SUBIR=1, *quando deveria ser* CHAMA_DESCER=0 *e* CHAMA_DESCER=1 *como se mostra na figura abaixo.



NOME:_____ **TURMA** _____

4 – O diagrama de transição de estados da figura representa uma máquina de Moore com uma entrada X e uma saída S.



a) Construa a tabela de transição de estados, atribuindo uma codificação apropriada aos estados.

Utilizando primeiro os nomes simbólicos atribuídos aos estados, a tabela de transição pedida é:

	próximo estado		
estado actual	*X=0*	*X=1*	*saída S*
init	*init*	*A*	*0*
A	*B*	*A*	*0*
B	*init*	*C*	*0*
C	*B*	*A*	*1*

Codificando agora os estados como: init=00, A=01, B=10 e C=11, serão necessários 2* flip-*flops do tipo D e a tabela de transição de estados fica:

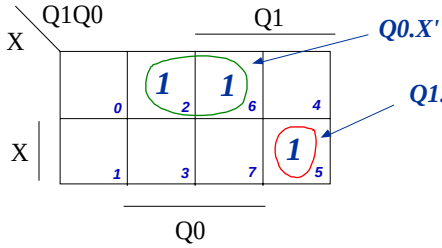
	próximo estado		
estado actual	*Q1* Q0**		
Q1 Q0	*X=0*	*X=1*	*saída S*
0 0	*0 0*	*0 1*	*0*
0 1	*1 0*	*0 1*	*0*
1 0	*0 0*	*1 1*	*0*
1 1	*1 0*	*0 1*	*1*

NOME:_____ **TURMA** _____

b) Desenhe o esquema do circuito lógico que implementa a máquina de estados, utilizando *flip-flops* do tipo D.

Partindo da tabela de transição de estados construída na alínea anterior, vamos obter as equações de excitação dos dois* flip-flops*, como expressões do tipo soma-de-produtos:

Q1*(Q1,Q0,X)

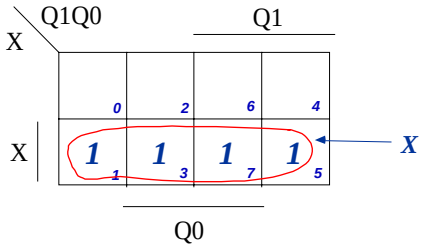


Q0.X'

Q1.Q0'.X

Q1*(Q1,Q0,X) = Q0.X' + Q1.Q0'.X

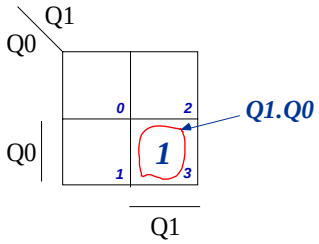
Q0*(Q1,Q0,X)



X

Q0*(Q1,Q0,X) = X

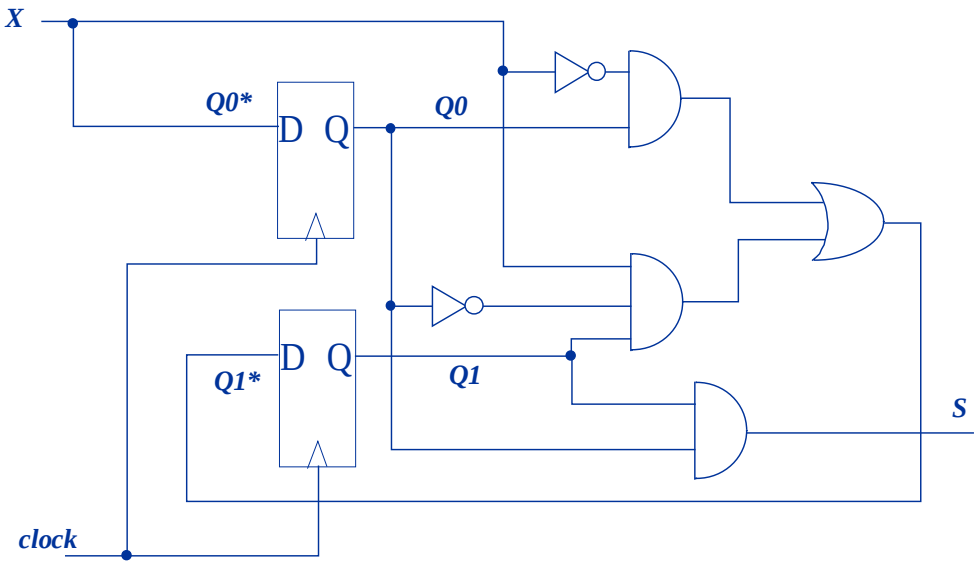
S(Q1,Q0)



Q1.Q0

S(Q1,Q0) = Q1.Q0

um circuito lógico que realiza esta máquina de estados é:



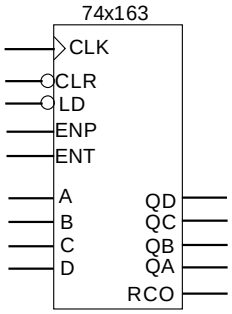
NOME:_____ **TURMA** _____

5 –

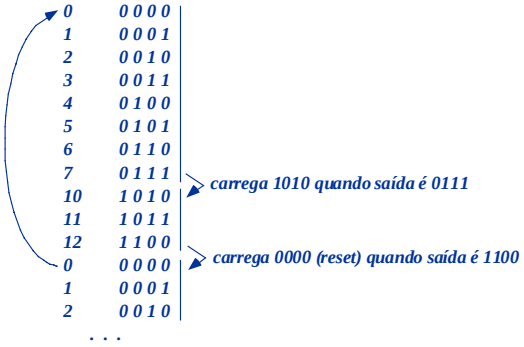
a) Construa um circuito síncrono baseado num contador binário (74x163) e em circuitos lógicos adicionais capaz de gerar, nas saídas Q_D,Q_C,Q_B,Q_A do contador, a seguinte sequência de valores (admitindo que o estado inicial é igual a Q_D,Q_C,Q_B,Q_A=0000):

0, 1, 2, 3, 4, 5, 6, 7, 10, 11, 12, 0, 1, 2, ...

74x163				estado presente				próximo estado			
/CLR	/LD	ENT	ENP	QD	QC	QB	QA	QD*	QC*	QB*	QA*
0	x	x	x	x	x	x	x	0	0	0	0
1	0	x	x	x	x	x	x	D	C	B	A
1	1	0	x	x	x	x	x	QD	QC	QB	QA
1	1	x	0	x	x	x	x	QD	QC	QB	QA
1	1	1	1	N (se N<15)				N + 1			
1	1	1	1	1	1	1	1	0	0	0	0



Analisando a sequência pretendida, representada em binário:

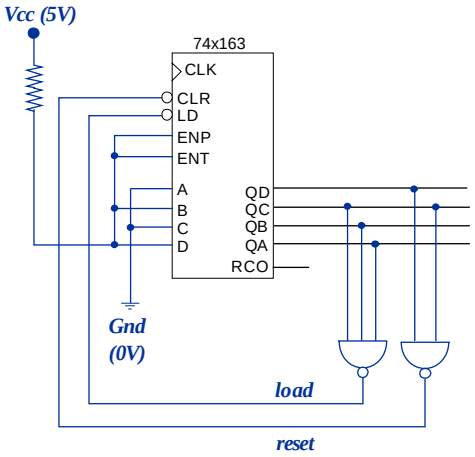


podemos concluir o seguinte:

1 – quando as saídas apresentam o estado 0111 deve ser activada a entrada LD, mantendo as entradas DCBA ligadas permanentemente a 1010.

2 – quando as saídas apresentam o estado 1100 deve ser activada a entrada CLR para iniciar as saídas com 0000

O circuito pretendido é o seguinte:

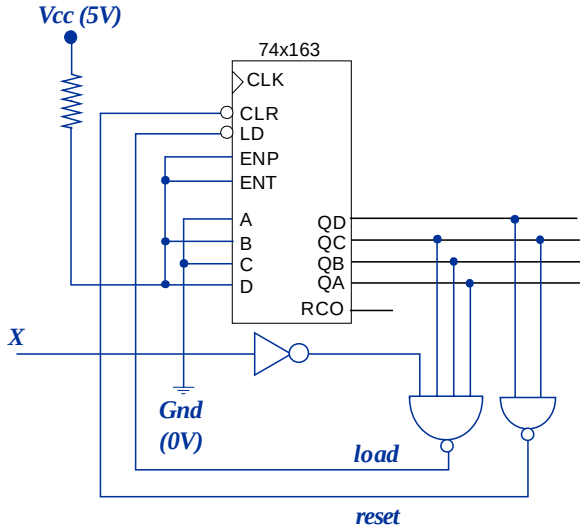


NOME:_____ **TURMA** _____

b) Modifique o circuito que construiu, acrescentando-lhe uma entrada X por forma a que quando X=0 é mantida a sequência de contagem anterior, e quando X=1 passa a ser gerada a sequência:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 0, 1, 2, ...

Para obter a nova sequência de contagem, basta desactivar o sinal de carregamento do contador quando X=1, e manter a mesma função do circuito anterior quando X=0. Uma solução consiste em acrescentar, na porta NAND que produz o sinal de* load*, uma entrada ligada a X': quando X=0, X'=1 e esse NAND produz a mesma função realizada no circuito anterior; quando X=1 (X'=0), o sinal* load *fica permanentemente igual a 1 (desactivado) e nunca é efectuado o carregamento do contador. O circuito resultante é:



NOME:_____ **TURMA** _____

6 – Pretende-se construir uma máquina de estados com uma saída Z que é '1' quando os 4 últimos bits consecutivos colocados na sua entrada X são 1011 (ver exemplo). Após o início do funcionamento da máquina de estados a saída Z só é considerada válida após o 4º ciclo de relógio.

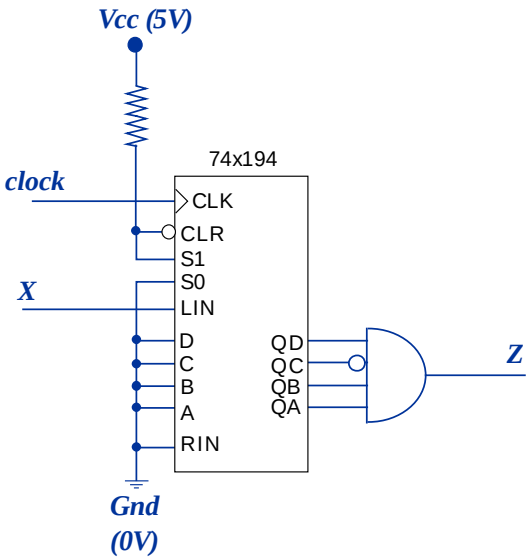
Entrada X: 0001**1011011**00**1011**000
Saída Z: xxxx000**1**00**1**000000**1**000

a) Desenhe um circuito baseado num *shift-register* 74x194 e em circuitos lógicos adicionais capaz de realizar a funcionalidade pretendida para a máquina de estados.

Universal Shift-register 74x194

função	S1	S0	QA*	QB*	QC*	QD*
hold	0	0	QA	QB	QC	QD
shift right	0	1	RIN	QA	QB	QC
shift left	1	0	QB	QC	QD	LIN
load	1	1	A	B	C	D

O circuito pretendido pode ser construído ligando o* shift-register *em configuração* shift-left *(S1=1, S0=0) e um comparador com a constante 1011 nas saídas do* shift-register *(uma porta AND com uma entrada negada), ligando a entrada X à entrada LIN do* shift-register*. Em cada transição de relógio os valores lógicos presentes na entrada X são deslocados para as saídas QD~QA, e a saída Z é activada com 1 sempre que nas saídas aparecer 1011:

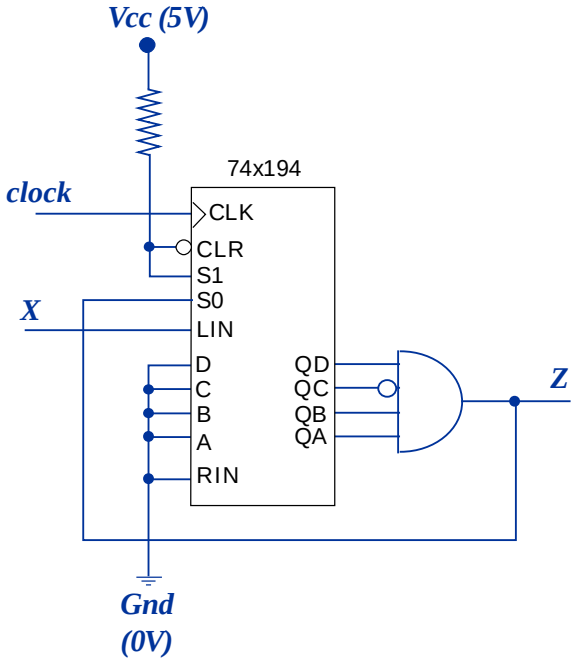


NOME:_____ **TURMA** _____

b) Admita agora que a máquina de estados só deve detectar sequências não sobrepostas (ver exemplo). Modifique o circuito anterior por forma a satisfazer este novo requisito (sugestão: é possível realizar este circuito sem introdução de novos circuitos lógicos ao circuito pedido na alínea anterior).

Entrada X: 0001**1011**01100**1011**000
Saída Z: xxxx000**1**000000000**1**000

Para que apenas sejam detectadas sequências1011 não sobrepostas, deve ser "apagada" a história dos bits anteriores sempre que for detectada uma sequência válida (1011). Uma forma de conseguir este comportamento consiste em limpar (carregar zero) nas saídas do shift-register, o que pode ser conseguido activando a entrada CLR sempre que Z=1. No entanto, como CLR é activo no nível lógico baixo, isso obriga a utilizar um inversor para negar Z. Para não gastar mais circuitos lógicos do que os usados na solução anterior, podemos realizar um load com 0000 (em vez de actuar a entrada CLR), ligando a saída Z à entrada S0: quando Z=0 é feito o shift-left (S1=1 e S0=0); quando é detectada a sequência 1011 a saída Z fica com 1 e é realizado o load(S1=1, S0=1) com 0000. O circuito resultante fica:



- FIM -