

Pontifícia Universidade Católica de Minas Gerais
Instituto de Ciências Exatas e Informática – ICEI
Arquitetura de Computadores I

ARQ1 _ Aula_13

Tema: Introdução à linguagem Verilog e simulação em Logisim (circuitos sequenciais)

Orientação geral:

Atividades previstas como parte da avaliação

Apresentar todas as soluções em apenas um arquivo com formato texto (.txt).

As implementações e testes dos exemplos em Verilog (.v) fornecidos como pontos de partida, também fazem parte da atividade e deverão ter os códigos fontes entregues separadamente. As saídas de resultados, opcionalmente, poderão ser copiadas ao final do código, como comentários.

Atividades extras e opcionais

Outras formas de solução serão opcionais; não servirão para substituir as atividades a serem avaliadas. Se entregues, contarão apenas como atividades extras.

As execuções deverão, preferencialmente, serão testadas mediante uso de entradas e saídas padrões, cujos dados/resultados deverão ser armazenados em arquivos textos. Os resultados poderão ser anexados ao código, ao final, como comentários.

Os *layouts* de circuitos deverão ser entregues no formato (.circ), identificados internamente. Figuras exportadas pela ferramenta serão aceitas como arquivos para visualização, e não terão validade para fins de avaliação. Separar versões completas (a) e simplificadas (b).

Arquivos em formato (.pdf), fotos, cópias de tela ou soluções manuscritas também serão aceitos como recursos suplementares para visualização, e não terão validade para fins de avaliação.

Atividade: Circuitos sequenciais – Flip-Flops – Contadores

Análise e síntese de circuitos sequenciais

As técnicas para análise de circuitos sequenciais que implementam uma certa máquina de estados finitos, em geral, dividem-se em duas etapas:

1. determinar as funções que determinam o próximo estado e as saídas
 - especificar as equações que representem a lógica do circuito e as saídas de cada **flip-flop** (estado corrente);
 - especificar as equações que determinem as transições entre dois pulsos de **clock**;
 - construir a *tabela de transições* para cada uma das combinações das entradas, indicando quais os próximos estados;
 - identificar todas as combinações que representem um mesmo estado e reescrevê-las em uma *tabela de estados*;
2. construir as tabelas de estados/saídas que especifiquem o comportamento do circuito para todas as combinações das entradas e do estado corrente:
 - verificar as funções das saídas em relação às entradas e aos estados correntes;
 - após avaliar todas as combinações de entradas e estados, combinar a tabela de estados com essas informações e criar a tabela de estados/saídas, relacionando cada saída ao próximo estado.

Exemplo 1:

Considerar o circuito abaixo com um **flip-flop** tipo D.

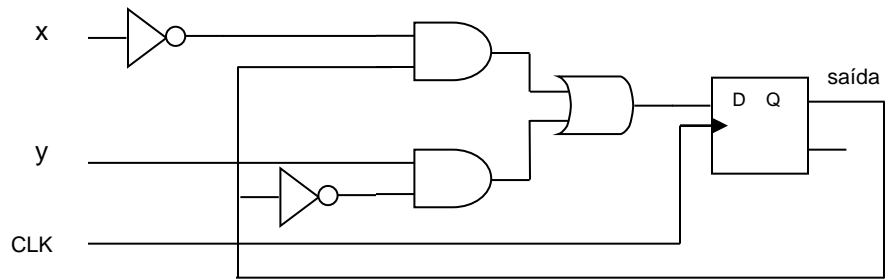


Tabela de transições

$Q_t \backslash xy$	00	01	10	11
0	0	1	0	1
1	1	1	0	0

Q_{t+1}

Equações de transições

$$D = x' \cdot Q + y \cdot Q'$$

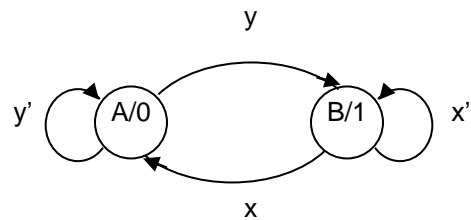
$$Q_{t+1} = x' \cdot Q_t + y \cdot Q'_t$$

Tabela de estados/saídas

$Q_t \backslash xy$	00	01	10	11
A	A,0	B,1	A,0	B,1
B	B,1	B,1	A,0	A,0

$Q_{t+1},$ saída

Diagrama de estados



Considerar o circuito abaixo com dois **flip-flops** tipo JK.

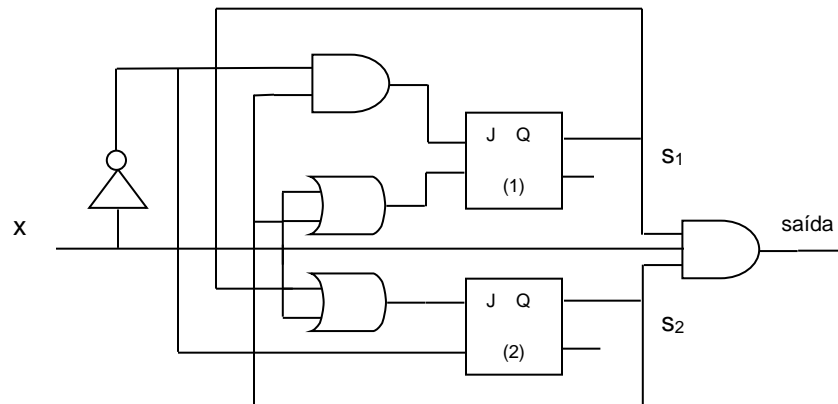


Tabela de transições

S ₁ (t)	S ₂ (t)	X	S ₁ (t+1)	S ₂ (t+1)	saída
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	1	0	0
0	1	1	0	1	0
1	0	0	1	1	0
1	0	1	0	1	0
1	1	0	0	0	0
1	1	1	0	1	1

Equações de transições

$$\text{saída} = S_1 \cdot S_2 \cdot X$$

$$J_1 = S_2 \cdot X' \quad \text{e} \quad K_1 = S_2 + X$$

$$J_2 = S_1 + X \quad \text{e} \quad K_2 = X'$$

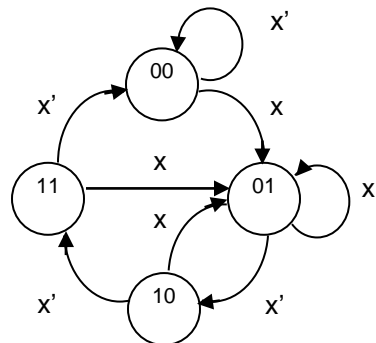
$$\begin{aligned} Q_{t+1} &= J_1 Q_t' + K_1' Q_t \\ S_1 &= S_2 \cdot X' \cdot S_1' + (S_2 + X) \cdot S_1 \\ &= S_2 \cdot X' \cdot S_1' + S_2' \cdot X' \cdot S_1 \\ &= X' \cdot (S_2 \cdot S_1' + S_2' \cdot S_1) \\ &= X' \cdot (S_1 \text{ xor } S_2) \end{aligned}$$

$$\begin{aligned} Q_{t+1} &= J_2 Q_t' + K_2' Q_t \\ S_2 &= (X + S_1) \cdot S_2' + (X')' \cdot S_2 \\ &= (X \cdot S_2') + (S_1 \cdot S_2') + (X \cdot S_2) \\ &= X \cdot (S_2' + S_2) + (S_1 \cdot S_2') \\ &= X + (S_1 \cdot S_2') \end{aligned}$$

Tabela de estados/saídas

S1	S2	x=0		x=1		saída
0	0	0	0	0	1	0
0	1	1	0	0	1	0
1	0	1	1	0	1	0
1	1	0	0	0	1	0/1

Diagrama de estados



Exemplo 2:

Projetar um contador crescente módulo 4 (0-1-2-3-0) com *flip-flops* tipo D.

Tabela de transições

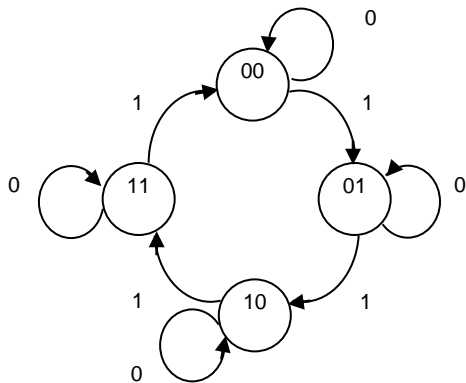


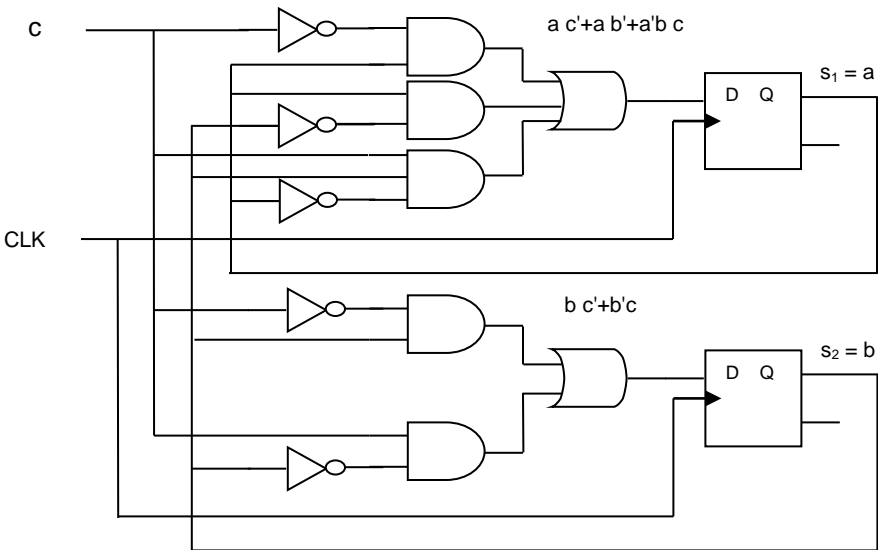
Diagrama de estados

	S1 (t)	S2 (t)	evento c	S1 (t+1)	S2 (t+1)
	a	b		a	b
0	0	0	0	0	0
1	0	0	1	0	1
2	0	1	0	0	1
3	0	1	1	1	0
4	1	0	0	1	0
5	1	0	1	1	1
6	1	1	0	1	1
7	1	1	1	0	0

Equações de transições

sinais	SoP	mintermos	simplificação
S1	3,4,5,6	$a'bc+ab'c'+ab'c+abc'$	$ac'+ab'+ab'c$
S2	1,2,5,6	$a'b'c+a'bc'+ab'c+abc'$	$bc'+b'c$

Circuito



Exemplo 3:

Projetar um contador decrescente módulo 4 (0-3-2-1-0) com **flip-flops** tipo D.

Tabela de transições

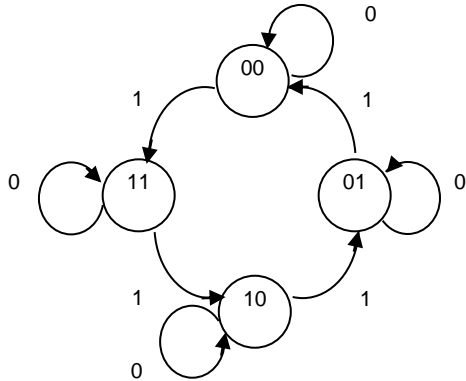


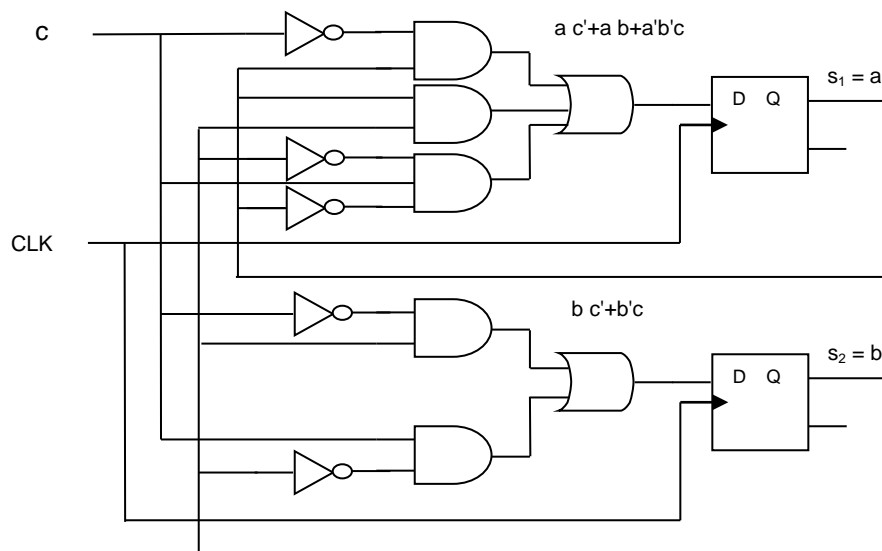
Diagrama de estados

	S ₁ (t)	S ₂ (t)	evento c	S ₁ (t+1)	S ₂ (t+1)
	a	b		a	b
0	0	0	0	0	0
1	0	0	1	1	1
2	0	1	0	0	1
3	0	1	1	0	0
4	1	0	0	1	0
5	1	0	1	0	1
6	1	1	0	1	1
7	1	1	1	1	0

Equações de transições

sinais	SoP	mintermos	simplificação
S ₁	1,4,6,7	$a'b'c+ab'c'+abc'+abc$	$ac'+ab+a'b'c$
S ₂	1,2,5,6	$a'b'c+a'bc'+ab'c+abc'$	$bc'+b'c$

Circuito



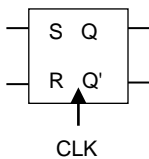
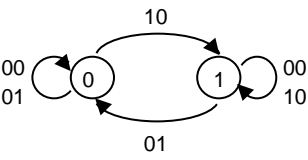
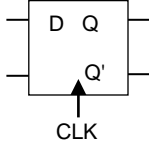
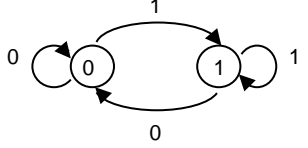
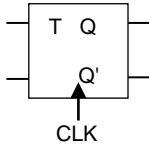
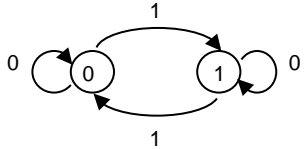
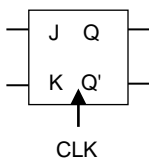
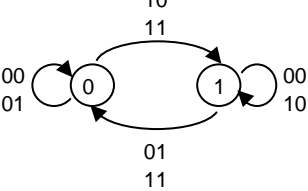
Exercícios

- 01.) Projetar e descrever em Logisim e Verilog um módulo, com portas e flip-flops tipo JK apenas, para implementar um contador assíncrono decrescente com 6 bits de comprimento.
DICA: Ver modelo anexo.
- 02.) Projetar e descrever em Logisim e Verilog um módulo com portas e flip-flops tipo JK apenas, para implementar um contador assíncrono crescente com 6 bits de comprimento.
- 03.) Projetar e descrever em Logisim e Verilog um módulo, com portas lógicas e flip-flops tipo JK apenas, para implementar um contador assíncrono decádico crescente com 5 bits de comprimento.
DICA: Ver modelo anexo.
- 04.) Projetar e descrever em Logisim e Verilog um módulo com portas e flip-flops tipo JK apenas, para implementar um contador assíncrono decádico decrescente com 5 bits de comprimento.
- 05.) Projetar e descrever em Logisim e Verilog um módulo, com portas e flip-flops tipo T apenas, para implementar um contador síncrono módulo 7.
DICA: Ver modelo anexo.

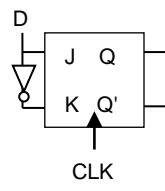
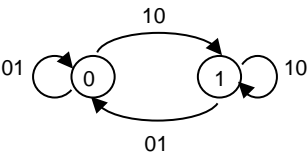
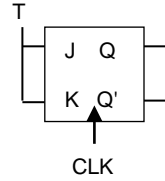
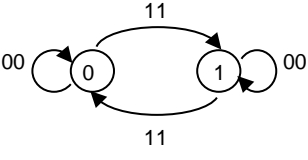
Extras

- 06.) Projetar e descrever em Logisim e Verilog um módulo, com portas e flip-flops tipo JK apenas, para implementar um contador em anel com 6 bits de comprimento.
DICA: Ver modelo anexo.
- 07.) Projetar e descrever em Logisim e Verilog um módulo com portas e flip-flops tipo JK apenas, para implementar um contador em anel torcido com 6 bits de comprimento.
DICA: Ver modelo anexo.

Flip-flops

Flip-flop	Estados	Característica	Transição	Equação																																								
		<table><tr><th>S</th><th>R</th><th>Q_{t+1}</th><th>Q'_{t+1}</th></tr><tr><td>0</td><td>0</td><td>Q_t</td><td>Q'_t</td></tr><tr><td>0</td><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>1</td><td>?</td><td>?</td></tr></table>	S	R	Q _{t+1}	Q' _{t+1}	0	0	Q _t	Q' _t	0	1	0	1	1	0	1	0	1	1	?	?	<table><tr><th>Q_t</th><th>Q_{t+1}</th><th>S</th><th>R</th></tr><tr><td>0</td><td>0</td><td>0</td><td>X</td></tr><tr><td>0</td><td>1</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>X</td><td>0</td></tr></table>	Q _t	Q _{t+1}	S	R	0	0	0	X	0	1	1	0	1	0	0	1	1	1	X	0	$Q_{t+1}=S+R'.Q_t$
S	R	Q _{t+1}	Q' _{t+1}																																									
0	0	Q _t	Q' _t																																									
0	1	0	1																																									
1	0	1	0																																									
1	1	?	?																																									
Q _t	Q _{t+1}	S	R																																									
0	0	0	X																																									
0	1	1	0																																									
1	0	0	1																																									
1	1	X	0																																									
		<table><tr><th>D</th><th>Q_{t+1}</th><th>Q'_{t+1}</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	D	Q _{t+1}	Q' _{t+1}	0	0	1	1	1	0	<table><tr><th>Q_t</th><th>Q_{t+1}</th><th>D</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	Q _t	Q _{t+1}	D	0	0	0	0	1	1	1	0	0	1	1	1	$Q_{t+1} = D$																
D	Q _{t+1}	Q' _{t+1}																																										
0	0	1																																										
1	1	0																																										
Q _t	Q _{t+1}	D																																										
0	0	0																																										
0	1	1																																										
1	0	0																																										
1	1	1																																										
		<table><tr><th>T</th><th>Q_{t+1}</th><th>Q'_{t+1}</th></tr><tr><td>0</td><td>Q_t</td><td>Q'_t</td></tr><tr><td>1</td><td>Q'_t</td><td>Q_t</td></tr></table>	T	Q _{t+1}	Q' _{t+1}	0	Q _t	Q' _t	1	Q' _t	Q _t	<table><tr><th>Q_t</th><th>Q_{t+1}</th><th>T</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	Q _t	Q _{t+1}	T	0	0	0	0	1	1	1	0	1	1	1	0	$Q_{t+1} = T \oplus Q_t$																
T	Q _{t+1}	Q' _{t+1}																																										
0	Q _t	Q' _t																																										
1	Q' _t	Q _t																																										
Q _t	Q _{t+1}	T																																										
0	0	0																																										
0	1	1																																										
1	0	1																																										
1	1	0																																										
		<table><tr><th>J</th><th>K</th><th>Q_{t+1}</th><th>Q'_{t+1}</th></tr><tr><td>0</td><td>0</td><td>Q_t</td><td>Q'_t</td></tr><tr><td>0</td><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>1</td><td>Q'_t</td><td>Q_t</td></tr></table>	J	K	Q _{t+1}	Q' _{t+1}	0	0	Q _t	Q' _t	0	1	0	1	1	0	1	0	1	1	Q' _t	Q _t	<table><tr><th>Q_t</th><th>Q_{t+1}</th><th>J</th><th>K</th></tr><tr><td>0</td><td>0</td><td>0</td><td>X</td></tr><tr><td>0</td><td>1</td><td>1</td><td>X</td></tr><tr><td>1</td><td>0</td><td>X</td><td>1</td></tr><tr><td>1</td><td>1</td><td>X</td><td>0</td></tr></table>	Q _t	Q _{t+1}	J	K	0	0	0	X	0	1	1	X	1	0	X	1	1	1	X	0	$Q_{t+1}=J.Q_t'+K'.Q_t$
J	K	Q _{t+1}	Q' _{t+1}																																									
0	0	Q _t	Q' _t																																									
0	1	0	1																																									
1	0	1	0																																									
1	1	Q' _t	Q _t																																									
Q _t	Q _{t+1}	J	K																																									
0	0	0	X																																									
0	1	1	X																																									
1	0	X	1																																									
1	1	X	0																																									

Configurações especiais

Flip-flop	Estados	Característica	Transição	Equação																																								
		<table><tr><th>J</th><th>K</th><th>Q_{t+1}</th><th>Q'_{t+1}</th></tr><tr><td></td><td></td><td></td><td></td></tr><tr><td>0</td><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td><td>0</td></tr><tr><td></td><td></td><td></td><td></td></tr></table>	J	K	Q _{t+1}	Q' _{t+1}					0	1	0	1	1	0	1	0					<table><tr><th>Q_t</th><th>Q_{t+1}</th><th>J/D</th><th>K/D'</th></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td><td>0</td></tr></table>	Q _t	Q _{t+1}	J/D	K/D'	0	0	0	1	0	1	1	0	1	0	0	1	1	1	1	0	$Q_{t+1}=1.Q_t'+0'.Q_t$ $Q_{t+1}=1$ $Q_{t+1}=0.Q_t'+1'.Q_t$ $Q_{t+1}=0$
J	K	Q _{t+1}	Q' _{t+1}																																									
0	1	0	1																																									
1	0	1	0																																									
Q _t	Q _{t+1}	J/D	K/D'																																									
0	0	0	1																																									
0	1	1	0																																									
1	0	0	1																																									
1	1	1	0																																									
		<table><tr><th>J</th><th>K</th><th>Q_{t+1}</th><th>Q'_{t+1}</th></tr><tr><td>0</td><td>0</td><td>Q_t</td><td>Q'_t</td></tr><tr><td></td><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>1</td><td>Q_t'</td><td>Q_t</td></tr></table>	J	K	Q _{t+1}	Q' _{t+1}	0	0	Q _t	Q' _t									1	1	Q _t '	Q _t	<table><tr><th>Q_t</th><th>Q_{t+1}</th><th>J=T</th><th>K=T</th></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td><td>0</td></tr></table>	Q _t	Q _{t+1}	J=T	K=T	0	0	0	0	0	1	1	1	1	0	1	1	1	1	0	0	$Q_{t+1}=0.Q_t'+0'.Q_t$ $Q_{t+1}=0'.Q_t = Q_t$ $Q_{t+1}=1.Q_t'+1'.Q_t$ $Q_{t+1}=1.Q_t' = Q_t'$
J	K	Q _{t+1}	Q' _{t+1}																																									
0	0	Q _t	Q' _t																																									
1	1	Q _t '	Q _t																																									
Q _t	Q _{t+1}	J=T	K=T																																									
0	0	0	0																																									
0	1	1	1																																									
1	0	1	1																																									
1	1	0	0																																									


```

module dff ( output q, output qnot,
             input  d, input clk );
reg q, qnot;

always @( posedge clk )
begin
    q <= d;      qnot <= ~d;
end

endmodule // dff

module jkff ( output q, output qnot,
             input  j, input k,
             input clk, input preset, input clear );

reg  q, qnot;

always @( posedge clk or preset or clear )
begin
    if ( clear )    begin q <= 0; qnot <= 1; end
    else
        if ( preset ) begin q <= 1; qnot <= 0; end
        else
            if ( j & ~k ) begin q <= 1; qnot <= 0; end
            else
                if ( ~j & k ) begin q <= 0; qnot <= 1; end
                else
                    if ( j & k )
                        begin q <= ~q; qnot <= ~qnot; end
end

endmodule // jkff

```

```

module tff ( output q, output qnot,
             input  t, input  clk,
             input  preset, input clear );

reg q, qnot;

always @( posedge clk or ~preset or ~clear)
begin
    if ( ~clear )
        begin  q <= 0;      qnot <= 1;  end
    else
        if ( ~preset )
            begin  q <= 1;      qnot <= 0;  end
        else
            begin
                if ( t ) begin q <= ~q; qnot <= ~qnot; end
            end
end

endmodule // tff

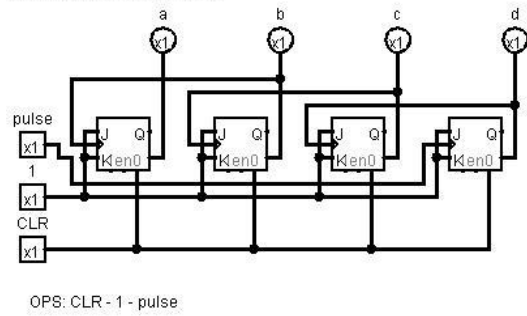
module srff ( output q, output qnot,
             input  s, input  r, input clk );
reg q, qnot;

always @( posedge clk )
begin
    if ( s & ~r ) begin q <= 1;      qnot <= 0; end
    else
        if ( ~s & r ) begin q <= 0;      qnot <= 1; end
    else
        if ( s & r )
            begin q <= 0; qnot <= 0; end // arbitrary
end

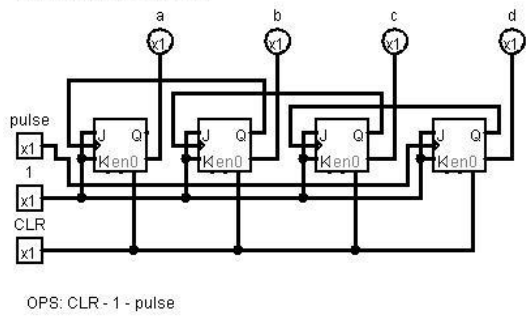
endmodule // srff

```

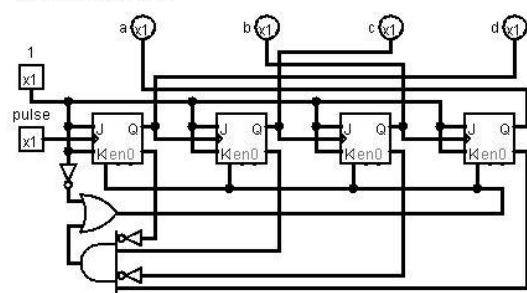
(Down) Asynchronous counter



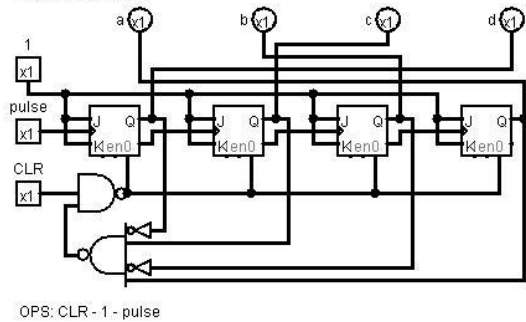
(Up) Asynchronous counter



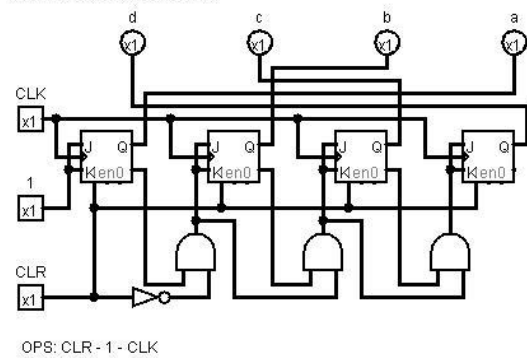
(Down) Decade counter



(Up) Decade counter



(Down) Synchronous counter



(Up) Synchronous counter

