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

ARQ1 _ Aula_14

Tema: Introdução à linguagem Verilog e simulação em Logisim

Orientação geral:

Os arquivos para simulação em Logisim (.circ) poderão ser entregues como complementação, deverão ser identificados internamente e entregues, acompanhados (ou não) de figuras equivalentes exportadas pela ferramenta. Programas em Verilog deverão ser entregues em formato (.v) com previsão de testes.

Atividade: Circuitos sequenciais – Flip-Flops Todos os circuitos deverão ser simulados no Logisim.

- 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 5 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 5 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 4 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 4 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 8. 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
S Q R Q' CLK	00 00 00 00 00 00 00 00 00 00 00 00 00	$\begin{array}{ c c c c c c c c c }\hline S & R & Q_{t+1} & Q'_{t+1} \\ \hline 0 & 0 & Q_{t} & Q_{t}' \\ \hline 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & ? & ? \\ \hline \end{array}$	$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	Q _{t+1} =S+R'.Q _t
CLK CLK		$ \begin{array}{c cccc} D & Q_{t+1} & Q^t_{t+1} \\ \hline 0 & 0 & 1 \\ 1 & 1 & 0 \\ \hline \end{array} $	$\begin{array}{c cccc} Q_t & Q_{t+1} & D \\ \hline 0 & 0 & 0 \\ \hline 0 & 1 & 1 \\ \hline 1 & 0 & 0 \\ \hline 1 & 1 & 1 \\ \end{array}$	$Q_{t+1} = D$
T Q Q' CLK	0 0 1 0	$ \begin{array}{c cccc} T & Q_{t+1} & Q'_{t+1} \\ 0 & Q_t & Q_t' \\ 1 & Q_t' & Q_t \\ \end{array} $	$\begin{array}{c cccc} Q_t & Q_{t+1} & T \\ \hline 0 & 0 & 0 \\ \hline 0 & 1 & 1 \\ \hline 1 & 0 & 1 \\ \hline 1 & 1 & 0 \\ \end{array}$	$Q_{t+1} = T \oplus Q_t$
J Q K Q' CLK	10 11 00 01 01 01 11	$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$	$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$	$Q_{t+1}=J.Q_t'+K'.Q_t$

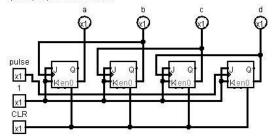
Configurações especiais

Flip-flop	Estados	Característica	Transição	Equação
CCC K Q'	01 0 10	J K Q _{t+1} Q' _{t+1} 0 1 0 1 1 0 1 0	$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	$\begin{aligned} Q_{t+1} &= 1.Q_t' + 0'.Q_t \\ Q_{t+1} &= 1 \\ Q_{t+1} &= 0.Q_t' + 1'.Q_t \\ Q_{t+1} &= 0 \end{aligned}$
T J Q K Q' CLK	00 0 10 00	J K Q _{t+1} Q' _{t+1} 0 0 Q _t Q _t ' 1 1 Q _t ' Q _t	$\begin{array}{ c c c c c c c }\hline Q_t & Q_{t+1} & J=T & K=T\\\hline 0 & 0 & 0 & 0\\\hline 0 & 1 & 1 & 1\\\hline 1 & 0 & 1 & 1\\\hline 1 & 1 & 0 & 0\\\hline \end{array}$	$\begin{aligned} &Q_{t+1}{=}0.Q_t'{+}0'.Q_t\\ &Q_{t+1}{=}0'.Q_t &= Q_t\\ &Q_{t+1}{=}1.Q_t'{+}1'.Q_t\\ &Q_{t+1}{=}1.Q_t' &= Q_t' \end{aligned}$

```
module dff ( output q, output qnot,
                                                           module tff (output q, output qnot,
             input d, input clk);
                                                                       input t, input clk,
reg q, qnot;
                                                                       input preset, input clear );
always @( posedge clk )
                                                          reg q, qnot;
begin
                qnot \leq ~d;
  q \ll d;
                                                          always @( posedge clk or ~preset or ~clear)
end
                                                           begin
endmodule // dff
                                                           if (~clear)
                                                            begin q \ll 0;
                                                                                   qnot <= 1; end
module jkff ( output q, output qnot,
                                                           else
        input j, input k,
                                                            if (~preset)
         input clk, input preset, input clear );
                                                            begin q \ll 1;
                                                                                   qnot \le 0; end
                                                            else
reg q, qnot;
                                                            begin
                                                             if (t) begin q \le -q; qnot \le -qnot; end
always @( posedge clk or preset or clear )
                                                            end
begin
                                                          end
 if (clear)
              begin q <= 0; qnot <= 1; end
 else
                                                           endmodule // tff
  if (preset) begin q <= 1; qnot <= 0; end
  else
                                                           module srff (output q, output qnot,
   if (j \& \sim k) begin q \le 1; q = 0; end
                                                                        input s, input r, input clk);
                                                           reg q, qnot;
    if (\sim j \& k) begin q \ll 0; qnot \ll 1; end
    else
                                                           always @( posedge clk )
                                                          begin
     if ( j & k )
                                                             if (s \& \sim r) begin q \ll 1;
                                                                                            qnot <= 0; end
         begin q <= ~q; qnot <= ~qnot; end
end
                                                              if (\sim s \& r) begin q \le 0;
                                                                                            qnot <= 1; end
                                                              else
endmodule // jkff
                                                               if (s&r)
                                                                begin q <= 0; qnot <= 0; // arbitrary end
                                                           end
```

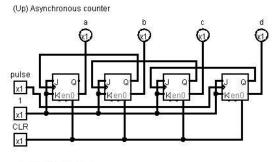
endmodule // srff

(Down) Asynchronous counter

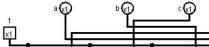


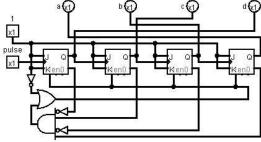
OPS: CLR - 1 - pulse

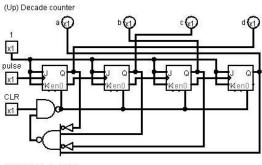
(Down) Decade counter



OPS: CLR - 1 - pulse

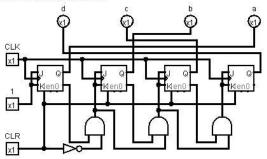




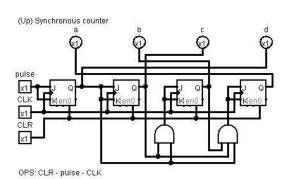


OPS: CLR - 1 - pulse

(Down) Synchronous counter



OPS: CLR - 1 - CLK



Counter base 5 pulso x1 :

