**UNIVERSIDADE DE SÃO PAULO**

**ESCOLA POLITÉCNICA**

**DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E SISTEMAS DIGITAIS**

**PCS3635 - LABORATÓRIO DIGITAL I**

****

**PLANEJAMENTO DA EXPERIÊNCIA 3**

Felipe Luis Korbes - NUSP: 13682893

Henrique Eduardo dos Santos de Souza - NUSP: 13679972

João Felipe de Souza Melo - NUSP: 13682913

**Turma:** 5

**Bancada:** B5

**Professor:** Reginaldo Arakaki

**Data da** **experiência**: 24/janeiro/2024

São Paulo

2024

1. **Objetivos**

Os objetivos dessa experiência envolvem o desenvolvimento prático de projetos de circuitos digitais, abrangendo a simulação de componentes Verilog, a elaboração e teste de um fluxo de dados, e a síntese em uma placa FPGA. A linguagem Verilog será empregada, e os resultados serão validados por meio de simulações e testes práticos, incluindo um desafio proposto pelo professor.

1. **Atividades pré-laboratório**

**Análise do contador 163:**

Para a realização da experiência 3, foram disponibilizados diversos materiais, incluindo códigos em Verilog, sendo um deles o arquivo "contador\_163.v", cujo código está detalhado no apêndice. Este código descreve um contador de 4 bits com várias entradas importantes. A entrada "clk" é responsável pelo pulso de clock do contador. A entrada "clr" atua na operação de clear, zerando o contador na borda de subida. Já a entrada "ld" está associada à operação de load, carregando um valor específico da entrada "D" para o contador. As entradas "ent" e "enp" são responsáveis pela habilitação da contagem, sendo "ent" associada à contagem direta e "enp" ao ENABLE do contador. A entrada "D" é um barramento de 4 bits que recebe o valor a ser carregado no contador durante a operação de load. A saída do contador é representada por "Q", e o sinal de saída "rco" é ativado quando o contador atinge seu último valor (1111), e o ENABLE ("enp") opera em nível lógico alto.  
 Após as declarações de entrada e saída há dois blocos always. O primeiro é sensível à borda de subida do clock, ou seja, sempre que houver uma borda de subida, o bloco é executado. Sempre que a estrada “clr” estiver LOW e houver uma borda de subida do clock, a saída Q do comparador recebe 0. O nível ativo da operação clear é quando clr está em LOW por conta da complementaridade no argumento do if em que há “~clr”. Caso “clk” esteja em nível HIGH, parte-se para a próxima comparação que é responsável pela operação de load. A operação de load é ativa em nível lógico baixo por conta do argumento do else-if que é o complemento de “ld”, ou seja, “~ld”, portanto, caso haja uma borda de subida de clk e clr esteja em HIGH e “ld” em LOW, ocorre a operação de load. A próxima comparação está relacionada com a contagem, caso os sinais “ent” e “enp” estejam em HIGH então o contador incrementa o seu valor, caso nenhuma dessas comparações sejam verdadeiras então a entrada do contador se mantém constante.

Já o segundo bloco always é responsável por controlar a saída SCO. Este bloco é sensível à entrada Q ou ao sinal de contagem ENT. Há uma operação lógica realizada neste bloco, se o sinal de controle “ent” estiver em HIGH e Q for seu valor final, então o valor de SCO será HIGH. Caso contrário será LOW. Após essa análise, percebe-se que este código em verilog simula o comportamento do CI 74163 utilizado na experiência 2.

Abaixo são os testes realizados para a familiarização do circuito.

| **Teste** | **Descrição** | **Sinais de Entradas** | **Saídas esperadas** | **Saídas encontradas** |
| --- | --- | --- | --- | --- |
| Condições iniciais | Todas as entradas inativadas | CLR=1, LD=1, D=0000, ENP=0, ENT=0, CLOCK=0 | Q=0000, RCO=0 | Q=0000, RCO=0 |
| **1** | Aciona clock 2x com entradas inativadas | CLR=1, LD=1, D=0000, ENP=0, ENT=0, acionar CLOCK 2 vezes | Q=0000, RCO=0 | Q=0000, RCO=0 |
| **2** | Aciona clock (somente) com clear ativado | CLR=0, LD=1, D=0000, ENP=0, ENT=0, acionar CLOCK | Q=0000, RCO=0 | Q=0000, RCO=0 |
| **3** | Aciona clock 5x com sinais de enable ativados | CLR=1, LD=1, D=0000, ENP=1, ENT=1, acionar CLOCK 5 vezes | Q=0101, RCO=0 | Q=0101, RCO=0 |
| **4** | Aciona clock com load ativado e dado=1011 | CLR=1, LD=0, D=1011, ENP=0, ENT=0, acionar CLOCK | Q=1011, RCO=0 | Q=1011, RCO=0 |
| **5** | Aciona clock 4x com sinais de enable ativados | CLR=1, LD=1, D=0000, ENP=1, ENT=1, acionar CLOCK 4 vezes | Q=1111, RCO=1 | Q=1111, RCO=1 |
| **6** | Aciona clock 2x com enp=0 e ent=1 | CLR=1, LD=1, D=0000, ENP=0, ENT=1, acionar CLOCK 2 vezes | Q=1111, RCO=1 | Q=1111, RCO=1 |
| **7** | Aciona clock 2x com enp=1 e ent=0 | CLR=1, LD=1, D=0000, ENP=1, ENT=0, acionar CLOCK 2 vezes | Q=1111, RCO=0 | Q=1111, RCO=0 |
| **8** | Aciona clock 2x com sinais de enable ativados | CLR=1, LD=1, D=0000, ENP=1, ENT=1, acionar CLOCK 2 vezes | Q=0001, RCO=0 | Q=0001, RCO=0 |
| **9** | Aciona clock com clear e load ativados e dado=1001 | CLR=0, LD=0, D=1001, ENP=1, ENT=1, acionar CLOCK | Q=0000, RCO=0 | Q=0000, RCO=0 |
| **10** | Aciona clock com load ativado e dado=1011 | CLR=1, LD=0, D=1001, ENP=1, ENT=1, acionar CLOCK | Q=1001, RCO=0 | Q=1001, RCO=0 |
| **11** | Aciona clock 6x com sinais de enable ativados | CLR=1, LD=1, D=1001, ENP=1, ENT=1, acionar CLOCK 6 vezes | Q=1111, RCO=1 | Q=1111, RCO=1 |

*Tabela XX: bateria de testes para o contador 163*

Os testes realizados foram realizados no Digital e foram condizentes com as saídas esperadas, é possível perceber pelo teste 9 que a operação clear tem preferência em relação à operação de load assim como o componente 74163.

**Análise do comparador 85:**

Para o comparador 85, temos um total de 5 inputs: A e B de 4 bits, que representam os dois números a serem comparados; e ALBi, AGBi e AEBi de 1 bit cada, correspondendo aos sinais de cascateamento que indicam A menor que B input, A maior que B input e A igual a B input, respectivamente. Além disso, há 3 sinais de output: ALBo, AGBo e AEBo, representando A menos que B output, A maior que B output e A igual a B output, respectivamente. Internamente, são utilizados dois wires, CSL e CSG, para carry-save low e carry-save high. Após as declarações, a lógica do circuito é apresentada, envolvendo somas e comparações em complemento de dois com os inputs A, B, ALBi e AGBi. Adicionalmente, a última comparação verifica a igualdade entre os dois números.

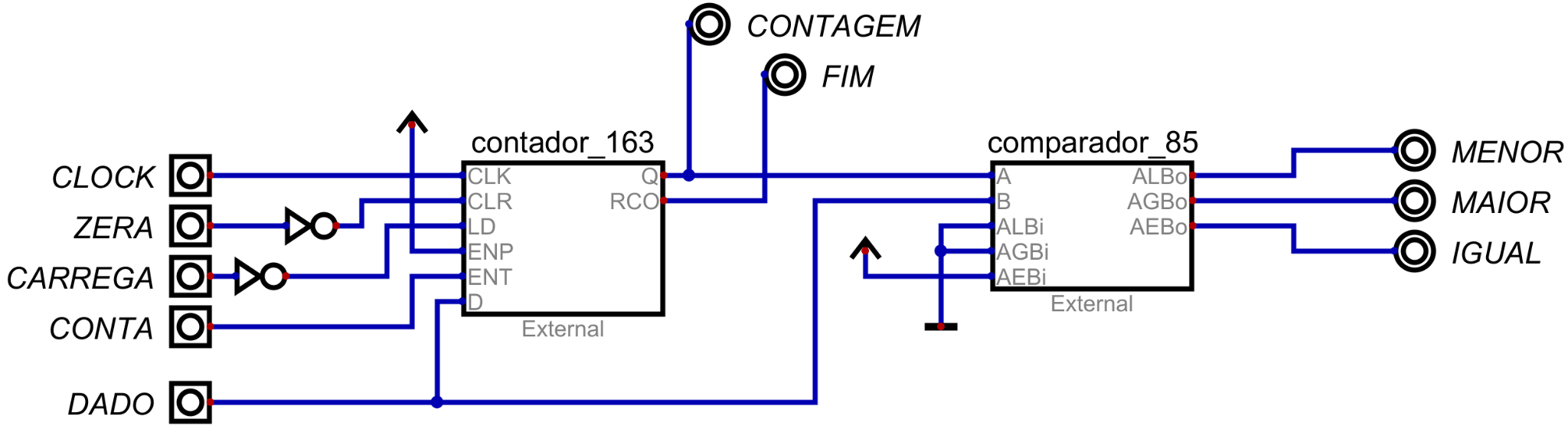
| **Teste** | **Sinais de Entradas** | **Saídas esperadas** | **Saídas encontradas** |
| --- | --- | --- | --- |
| condições iniciais | A>Bin=0, A=Bin=1, A<Bin=0, A=0000, B=0000 | A>Bout=0, A=Bout=1, A<Bout=0 | A>Bout=0, A=Bin=1, A<Bout=0 |
| **1** | A>Bin=0, A=Bin=1, A<Bin=0, A=0110, B=0110 | A>Bout=0, A=Bout=1, A<Bout=0 | A>Bout=0, A=Bin=1, A<Bout=0 |
| **2** | A>Bin=1, A=Bin=0, A<Bin=0, A=0110, B=0110 | A>Bout=1, A=Bout=0, A<Bout=0 | A>Bout=1, A=Bin=0, A<Bout=0 |
| **3** | A>Bin=0, A=Bin=0, A<Bin=1, A=0110, B=0110 | A>Bout=0, A=Bout=0, A<Bout=1 | A>Bout=0, A=Bin=0, A<Bout=1 |
| **4** | A>Bin=0, A=Bin=1, A<Bin=0, A=0001, B=0000 | A>Bout=1, A=Bout=0, A<Bout=0 | A>Bout=1, A=Bin=0, A<Bout=0 |
| **5** | A>Bin=1, A=Bin=0, A<Bin=0, A=0001, B=0000 | A>Bout=1, A=Bout=0, A<Bout=0 | A>Bout=1, A=Bin=0, A<Bout=0 |
| **6** | A>Bin=0, A=Bin=0, A<Bin=1, A=0001, B=0000 | A>Bout=1, A=Bout=0, A<Bout=0 | A>Bout=1, A=Bin=0, A<Bout=0 |
| **7** | A>Bin=0, A=Bin=1, A<Bin=0, A=0011, B=1100 | A>Bout=0, A=Bout=0, A<Bout=1 | A>Bout=0, A=Bin=0, A<Bout=1 |
| **8** | A>Bin=1, A=Bin=0, A<Bin=0, A=0011, B=1100 | A>Bout=0, A=Bout=0, A<Bout=1 | A>Bout=0, A=Bin=0, A<Bout=1 |
| **9** | A>Bin=0, A=Bin=1, A<Bin=0, A=0011, B=1100 | A>Bout=0, A=Bout=0, A<Bout=1 | A>Bout=0, A=Bin=0, A<Bout=1 |

*Tabela XX: bateria de testes para o comparador 85*

Os testes realizados foram realizados no Digital e foram condizentes com as saídas esperadas e, as saídas esperadas que foram preenchidas condizem com os resultados realizados no digital.

**Montagem do circuito da atividade 2:**

Abaixo está o circuito digital para a atividade 2, que foi feito no Digital, cujo arquivo pode ser encontrado juntamente com essa entrega.



*Figura XX: circuito digital da atividade 2*

Após montado o circuito, foi executada uma bateria de testes, conforme visto na tabela abaixo, onde todos os resultados observados bateram com o resultado esperado.

| **#** | **Caso de teste** | **Sinais de controle** | **Resultado esperado** | **Resultado observado** |
| --- | --- | --- | --- | --- |
| **c.i.** | Condições iniciais | clock=1 zera=0 carrega=0 conta=0  chaves=0000 | contagem=0, fim=0, maior=0, menor=0, igual=1 | contagem=0, fim=0, maior=0, menor=0, igual=1 |
| **1** | Zerar contador e observar a saída da contagem | zera=1 clock = | contagem=0, fim=0, maior=0, menor=0, igual=1 | contagem=0, fim=0, maior=0, menor=0, igual=1 |
| **2** | Ajustar chaves para 0001 | chaves=0001 | contagem=0, fim=0, maior=0, menor=1, igual=0 | contagem=0, fim=0, maior=0, menor=1, igual=0 |
| **3** | Incrementar contador e chaves=0001 | conta=1 clock=­ | contagem=1, fim=0, maior=0, menor=0, igual=1 | contagem=1, fim=0, maior=0, menor=0, igual=1 |
| **4** | Incrementar contador para 3 e ajustar chaves para 0010 | conta=1 clock=(2x)  chaves=0010 | contagem=3, fim=0, maior=1, menor=0, igual=0 | contagem=3, fim=0, maior=1, menor=0, igual=0 |
| **5** | Ajustar chaves para 0110 | chaves=0110 | contagem=3, fim=0, maior=0, menor=1, igual=0 | contagem=3, fim=0, maior=0, menor=1, igual=0 |
| **6** | Incrementar contador até 1110 | conta=1 clock=(11x) | contagem=14, fim=0, maior=1, menor=0, igual=0 | contagem=14, fim=0, maior=1, menor=0, igual=0 |
| **7** | Incrementar contador | conta=1 clock= | contagem=15, fim=1, maior=1, menor=0, igual=0 | contagem=15, fim=1, maior=1, menor=0, igual=0 |

*Tabela XX: tabela de testes do circuito digital da atividade 2*