|  |  |  |  |
| --- | --- | --- | --- |
| 디지털시스템실험 결과보고서 | | | |
| 이름 : 정성훈 | | | 학번 : 2019171009 |
| **실험제목** | Latch, Flip-Flop, Shift Register 설계 및 구현 | | |
| **실험목표** | 1. BCD Ripple Counter 구현 2. 4-bits Register 구현 3. 4-bits Shift Register 구현 | | |
| **실험결과** | | | |
| **0. SR Latch, D Flip Flop, JK Flip Flop**  이번 Counter와 Register를 구현하기 위해서는 JK Flip Flop을 구현해야 했다. 그러기 위해 SR Latch, D Flip Flop이 필요했다.    세 회로의 회로도는 위와 같고, 이를 위해 아래와 같이 Verilog로 코딩했다.    **1. BCD Ripple Counter**  1. 구현하고자 하는 논리회로    위와 같이 JK Flip Flop 4 개를 연결해 0000 ~ 1001 까지 순환하는 BCD Ripple Counter를 설계하고자 하였다.  2. Verilog 코드    그림에 그려진 논리회로대로 JK FF에 선을 연결해 BCD Ripple Counter를 설계했다. 이를 검증하기 위한 Test bench code는 아래와 같다.    3. 시뮬레이션 파형  JK FF의 초기값이 설정되어 있지 않아 ModelSim의 simulator에서 모든 FF 값이 x가 나와 첨부하지 않았다.  4. FPGA 검증    LED 1~4에 각 Counter의 Q값을 mapping했다. 0000 ~ 1001까지 순서대로 LED가 켜지다가 1001 다음에는 다시 0000이 켜지는 걸 확인할 수 있었다.  **2. 4-bits Register**  1. 구현하고자 하는 논리회로    위과 같이 D FF 4개로 이루어져 CLK값이 인가될때 들어온 D값을 저장하는 4-bits register를 설계하고자 하였다.  2. Verilog Code    왼쪽은 register 구현을 위한 코드고, 오른쪽은 동작을 검증하기 위한 testbench 코드다. for loop를 이용해 모든 D input에 대한 output을 검증하도록 했다.  3. 시뮬레이션 파형    위는 test bench code를 simulation한 결과다. D FF가 posedge triggered였기 때문에 clk가 posedge일 때 Q가 D값을 저장했다.  4. FPGA 검증    Q3~0을 LED 5~7에 mapping 했고, input값인 D 3~0을 keypad 1, 4, 7, \*에 mapping했다.(위에서부터 내려오는 순서)  왼쪽 사진은 \*만 눌렀을 때이고, 오른쪽 사진은 1과 4를 같이 눌렀을 때다. 각각에 해당하는 LED만 On 된 것을 확인할 수 있었다.  **3. 4-bits Shift Register**  1. 구현하고자 하는 논리회로    위와 같이 Serial Input을 cnt의 주파수에 따라 한 비트씩 shift되는 4-bits Shift Register를 구현하고자 하였다.  2. Verilog Code    왼쪽은 shift register 구현을 위한 코드고, 오른쪽은 동작 검증을 위한 testbench 코드다. 첫 번째 FF에 input을 잇고, 이후에 wire를 이용해 나머지 FF들을 serial로 이었다.  3. 시뮬레이션 파형    Serial input으로 101101을 주었다. i에 clock의 한 주기마다 101101 신호를 주었다. input이 들어가고 나서 4번의 주기동안 shift가 이루어진 후 O에 serial output으로 101101이 나온것을 확인할 수 있었다.  4. FPGA 동작검증    LED 1~4에 각 FF에서 나오는 output을 mapping했고, keypad 1에 input i를 mapping했다. clock의 주기에 따라 한 비트씩 shift 되었고, clock의 posedge 타이밍에 keypad를 누르면 LED 1에 불이 들어왔다. | | | |
| **토의** | | Modelsim은 초기값을 항상 x로 두고, 이를 따로 초기화해주지 않으면 x가 들어간 논리연산은 확실한 값이 아니면 x로 나왔기 때문에 BCD Ripple Counter를 제대로 시뮬레이션 할 수 없었다.  이를 시뮬레이션 하기 위해서는 JK FF에 따로 reset을 만들어서 reset이 켜졌을때 FF의 값이 초기화가 되도록 해야겠다고 생각했다. | |