8 장

Shift Register & Counter

1. 실험 목표

▣ 쉬프트 레지스터의 동작을 이해한다.

▣ 양방향(Bi-directional) 쉬프트 레지스터를 구현하고 동작을 확인한다.

▣ 카운터의 구조와 동작을 이해한다.

2. 실험 이론

디지털 시스템에서 데이터를 저장하는 1-비트 기본 소자인 D-FF 을 여러 개 이용하여 n-비트 단위의 데이터 저장 레지스터를 설계한다. 레지스터는 CPU 내에 사용이 되는 기본적인 데이터 저장 소자이며, 8-비트 레지스터의 경우에는 8 개의 D-FF 과 공통으로 사용되는 제어신호를 연결하여 단위 처리가 가능한 8-비트 레지스터를 구성한다. 이러한 레지스터는 데이터를 읽고

쓰기 동작을 지원하며, 저장된 내부 데이터 비트를 필요에 따라 좌/우로 이동시킬 수 있는 쉬프트(Shift) 레지스터를 설계할 수 있다. 이러한 구조는 컴퓨터 시스템에서 직렬 데이터를 전송하는 응용, 즉 마우스, 키보드 등의 직렬 통신이 요구되는 곳에 사용이 된다. 레지스터의 다른 응용의 예인 카운터 (Counter)는 고정된 순서의 상태를 클럭에 맞추어 단계적으로 이동해 가는 구조이며, 가장 기본적인 구조가 2 진 카운터를 생각할 수 있다. 이 실험에서는 기본적인 쉬프트 레지스터와 양방향 쉬프트 레지스터, 그리고 존슨 카운터에 대해 구조와 동작을 실험하게 된다.

가) 쉬프트 레지스터(Shift Register)

쉬프트 레지스터는 데이터 통신이나 컴퓨터 시스템의 데이터를 전송하거나 저장하는 블록으로 주로 이진 정보를 이동 및 저장한다. 데이터의 초기 입력이나 이동시, 레지스터에 정보가 저장될 때는 하나의 플립플롭에 하나의 비트가 저장된다. 쉬프트 레지스터는 클럭에 동기를 맞추어 저장된 입력이 이진 정보들이 차례대로 인접의 플립플롭에 한 클럭에 한번씩 이동되도록 동작하며, 필요한 시점에 저장된 데이터를 읽거나, 저장하거나 할 수 있도록 설계한다. 데이터가 한번에 하나씩 차례대로 좌/우로 쉬프트 되도록 한 것이 직렬형 쉬프트 레지스터이다.

[그림 104-1(a)]는 에지 구동형 D-FF 을 이용하여 구성한 6 비트의 직렬형 쉬프트 레지스터이다. 6 비트의 이진 정보를 저장할 수 있고 한 비트의 정보를 입력하는 데에 한 클럭 주기의 시간이 필요하므로 이 직렬형 쉬프트 레지스터를 채우는 데에는 6 클럭의 주기가 필요하다.

[그림 10-1(b)]에는 그림과 같은 쉬프트 레지스터에 데이터 101101 을 입력하여 이동시키는 과정을 입력 신호와 출력 신호의 파형으로 나타내었다. [그림 10-2]는 에지 구동형 D-FF 을 사용하여 직렬형 쉬프트 레지스터의 설계시 모델을 Verilog HDL 로 표현한 것이다.

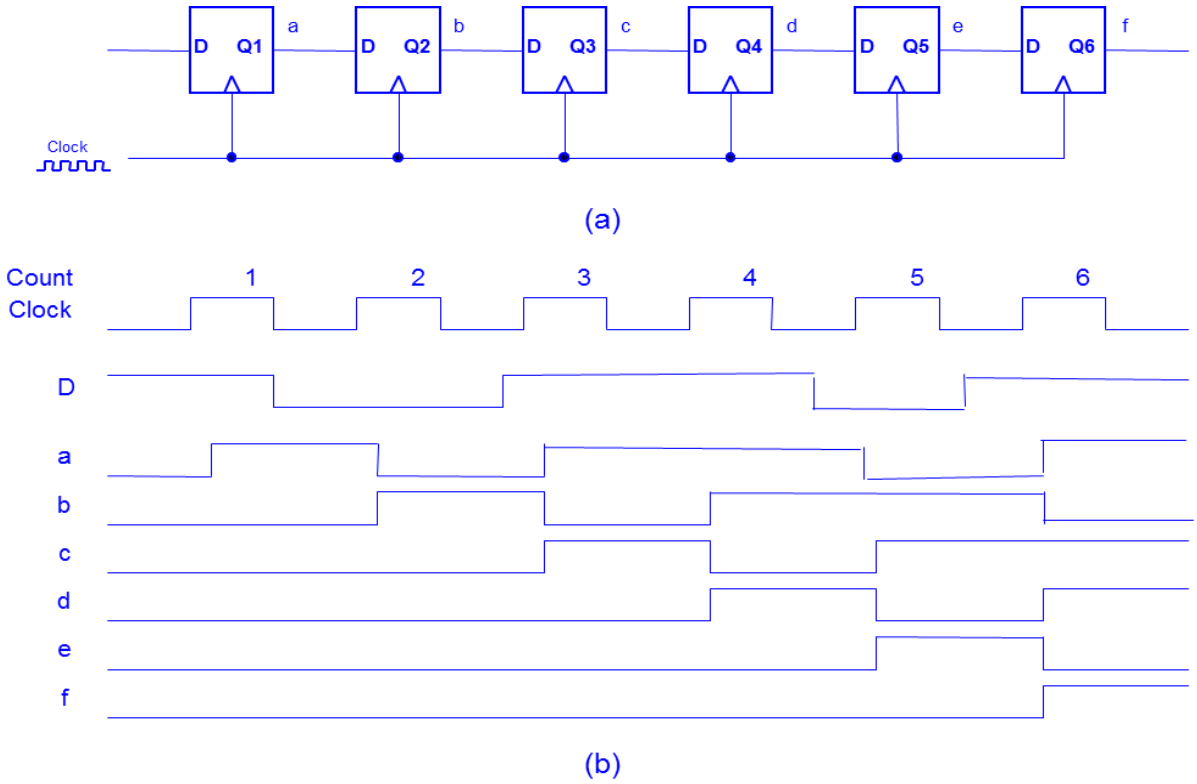


그림 10-1: 6 비트 직렬형 쉬프트 레지스터 (a) 회로도, (b) 101110 이동 과정.

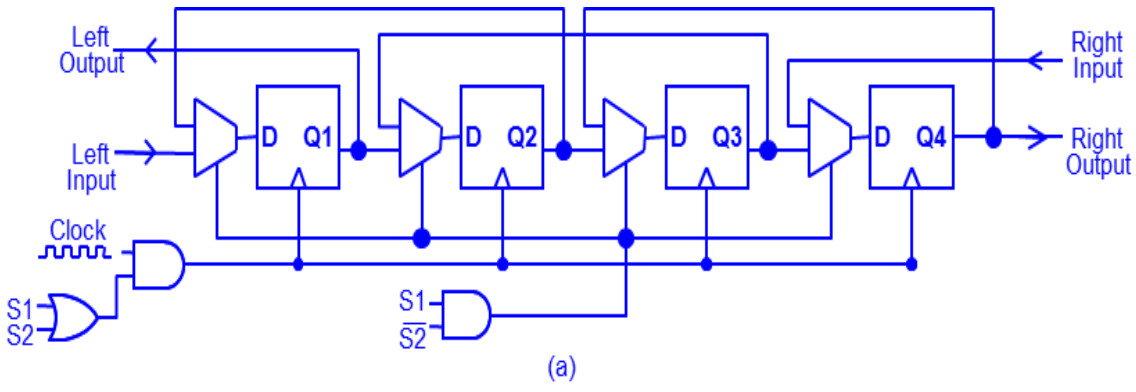
```
module SHIFT_REG_d(D, Clk, Q);

input D, Clk;
output Q;

wire D1, D2, D3, D4, D5;
wire Q1, Q2, Q3, Q4, Q5;

D_FF D_FF1 (Q1, D, Clk);
D_FF D_FF2 (Q2, Q1, Clk);
D_FF D_FF3 (Q3, Q2, Clk);
D_FF D_FF4 (Q4, Q3, Clk);
D_FF D_FF5 (Q5, Q4, Clk);
D_FF D_FF6 (Q, Q5, Clk);

endmodule
```

그림 10-2: 6 비트 직렬형 쉬프트 레지스터의 Verilog HDL 코드.

일반적으로 직렬형 쉬프트 레지스터는 n-비트의 정보를 저장하기 위해 n 개의 플립플롭이 사용되고, 또한 n-비트의 정보를 저장하기 위해서는 n 클럭 주기만큼의 시간이 레지스터의 각 단계를 통한 정보의 쉬프트 동작을 위해서 필요하다.

나) 양방향 쉬프트 레지스터 (Bi-directional Shift Register)

기본적인 쉬프트 레지스터는 보통 한 방향으로 입력을 받아 다른 한 방향으로 출력을 한다. 따라서 [그림 10-3(a)]와 같이, 양방향으로 입력과 출력을 하기 위해서는 선택기 (Mux)를 이용하여 레지스터의 입력을 왼쪽으로부터 입력되는 비트를 저장할 것인지, 아니면 오른쪽으로부터 입력되는 입력을 저장할 것인지를 결정해 주는 제어 신호가 필요하다. 또한 클럭의 입력이 0 으로 들어가게 하면 쉬프트 레지스터의 값을 현 상태를 그대로 유지할 수 있게 된다. 양방향 쉬프트 레지스터의 동작을 제어하기 위하여 S1 과 S2 는 모드를 설정해 주는 신호이다. [10-3(b)]는 각각의 모드를 설정해 주는 제어 신호의 진리표이다. 그림의 진리표와 같이 S1, S2 두 신호 모두 0 일 때는 클럭의 출력이 0 이 유지되도록 AND 게이트의 입력으로 연결하였고, 좌측 이동 (Shift Left)을 수행할 때에는 Mux 선택 신호를 그림과 같이 우측 D-FF 의 출력을 좌측 D-FF 의 입력단의 Mux 입력에 연결하여 주며, 우측 이동 시에는 좌측 D-FF 출력을 우측 D-FF 입력단 Mux 에 연결해 준다.



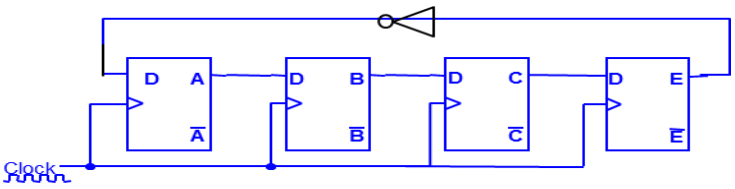
S1	S2	Mode
0	0	Hold
0	1	Shift Right
1	0	Shift Left
1	1	Not Allowed

(b)

그림 10-3: 4 비트 양방향 쉬프트 레지스터 (a) 회로도, (b) 제어신호의 진리표.

다) 존슨 카운터의 구조와 동작

직렬 쉬프트 레지스터를 이용하여 [그림 10-4]와 같이 시프트 레지스터의 마지막 단의 플립플롭의 출력을 역 변환하여 첫 번째 단의 플립플롭의 입력인 D 와 쉬프트 카운터 형태의 존슨 카운터(Johnson counter)를 설계할 수 있다. 연결된 n 개의 플립플롭들은 mod-2N 의 카운터가 된다. [그림 10-4]에서와 같이 4 개의 플립플롭을 연결하면 mod-8 존슨 카운터가 된다. 이러한 카운터들은 입력 상태에 따라서 같은 상태를 반복하는 부당 상태(illegal state)에 이르기도 하기 때문에 카운터의 부당 상태를 피하기 위한 주의가 필요하다. 카운터가 진행하는 8 개의 상태는 [그림 10-4(b)]와 같다. 이 카운터는 가능한 16 개의 상태들 중 8 개의 상태만을 고정 싸이클로 반복 수행하게 된다. 만약 정의 되지 않은 부당 상태가 발생하면 이 카운터는 0101 과 1010 사이만을 계속 반복한다.



(a)

E	C	B	A	State
0	0	0	1	1
0	0	1	1	2
0	1	1	1	3
1	1	1	1	4
1	1	1	0	5
1	1	0	0	6
1	0	0	0	7
0	0	0	0	8
0	0	0	1	1

(b)

그림 10-4: 존슨 카운터 (a) 회로도, (b) 상태 진리표.

[그림 10-4]와 같은 존슨 카운터의 Verilog 코드는 다음의 [그림 10-5]와 같이 구성된다.

```
module mod8(clk, reset, A, B, C, E, EN);
    input clk, reset; output A, B, C, E, EN;
    D_FF D_FF1(EN, reset, clk, A);
    D_FF D_FF2(A, reset, clk, B);
    D_FF D_FF3(B, reset, clk, C);
    D_FF D_FF4(C, reset, clk, E);
    not n1(EN, E);
endmodule

module D_FF(D, reset, clk, Q);
    input D, reset, clk; output Q; reg Q;
    always @(negedge reset or posedge clk)
    begin
```

```
        if (reset == 0)  Q = 1'b0;
        else Q = D;
    end
endmodule
```

그림 10-5: 8 비트 mod-8 쉬프트 카운터의 Verilog HDL 코드.

3. 예비보고서 작성

▣ Parallel Shift Register 에 대해서 조사하시오.

▣ 4 비트 양방향 쉬프트 레지스터를 Verilog HDL 코드로 표현하시오.

▣ 존슨 카운터에 대한 상태 전이도를 구하시오.

4. 실험에 필요한 기기

▣ 실험용 PC

▣ MAX PLUS II 소프트웨어 / Quartus II-Modelsim 소프트웨어

5. 실험 내용과 과정

① 예비보고서 작성 과정에서 조사한 내용들을 바탕으로, 4 비트 양방향 쉬프트 레지스터의 Verilog HDL 을 Verilog HDL 로 모델링 한다.

② 실험 자료 및 안내 자료에 따라 각 gate 별로 delay 를 설정해주고, 모든 Verilog HDL 코드마다 `timescale 1ns/100ps 처럼 시뮬레이션 시간 단위 및 해상도를 정의한다.

③ 실험 결과를 기반으로 표 10-1 의 내용을 완성하고 결과보고서 문제에 따라 이를 설명하시오.

④ 그림 10-5 의 D-FF 을 이용한 존슨 카운터 코드를 따라 구현해본다.

⑤ D-FF 을 이용한 존슨 카운터 코드에서 인버터를 제거한 수행 결과와 비교하여 결과보고서를 작성한다.

표 10-1: 4 비트 양방향 쉬프트 레지스터

Left Input	Right Input	S1	S2	Left Output	Right Output
0	1	0	1		
1	1	0	1		
1	0	0	1		
1	0	1	0		
0	1	1	0		
1	1	1	0		
0	1	0	0		
0	1	0	1		

6. 실험 결과 보고서

① 표 10-1 의 결과를 설명하시오.

② 쉬프트 레지스터의 종류와 용도에 대해서 설명하시오.

③ 존슨 카운터의 인버터를 제거한 수행 결과를 구하시오.