실험 7 쉬프트 레지스터(shift register)

실험 목표

쉬프트 레지스터의 동작을 이해하고 응용하여 PRBS(pseudo-random binary sequence) 발생기를 구성한다. VHDL을 사용하여 쉬프트 레지스터를 제작하고, 기존의 쉬프트 레지스터와 연결하여 링 카운터로써 data의 이동을 확인한다.

실험 부품

74LS86 (XOR gate),
74LS194 (shift register)
Clock pulse generator
Breadboard
Clock pulse generator
디지털 멀티미터(digital multimeter)
DIP switch module
LED array module
FPGA 실험 보드(EP4CE6)
USB Blaster Ⅱ
Quartus Ⅱ

관련 이론

레지스터는 두 개 이상의 플립플롭의 클럭 신호를 공통으로 연결한 것이다. 이것은 논리회로의 출력 데이터를 일시적으로 기억하여 디지털 시스템의 주요부와 입·출력부를 잇는 역할을 하고, 2진 레지스터를 이용하면 보수를 취하는 연산, 그리고 곱셈 및 나눗셈과 같은 연산 동작을 하는 중요한 논리 기능을 가진 회로이다.

1. 쉬프트 레지스터(shift register)

CLK 신호가 입력 될 때마다 저장된 데이터가 오른쪽 혹은 왼쪽으로 이동한다. 쉬프트 레지스터는 데이터 입력 방법에 따라 직렬입력, 병렬입력으로 나누고 데이터 출력 방법에 따라 직렬출력, 병렬출력으로 구분된다. 또한 데이터 입·출력 방법을 조합해서 “직렬 입·출력", "직렬입력, 병렬출력", "병렬입력, 직렬출력", "병렬 입·출력"으로 나눌 수 있으며, 데이터의 이동 방향에 따라 쉬프트 레지스터는 쉬프트-라이트(shift-right), 쉬프트-레프트(shift-left), 양방향(bi-directional)으로 구분된다.

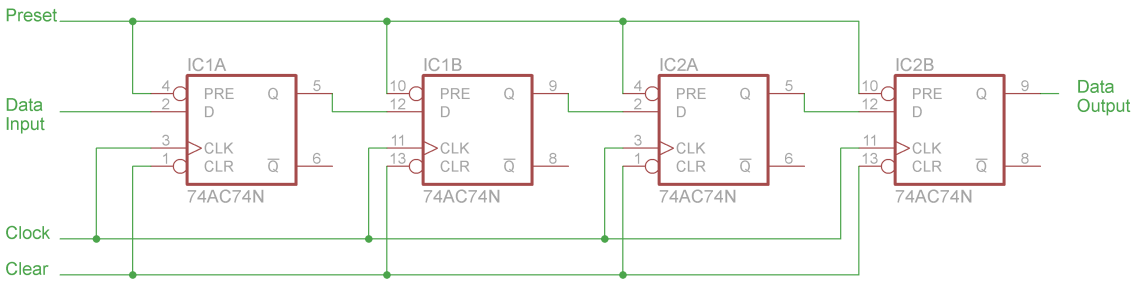


그림 7-1 Serial-In Serial-Out Shift Right Register

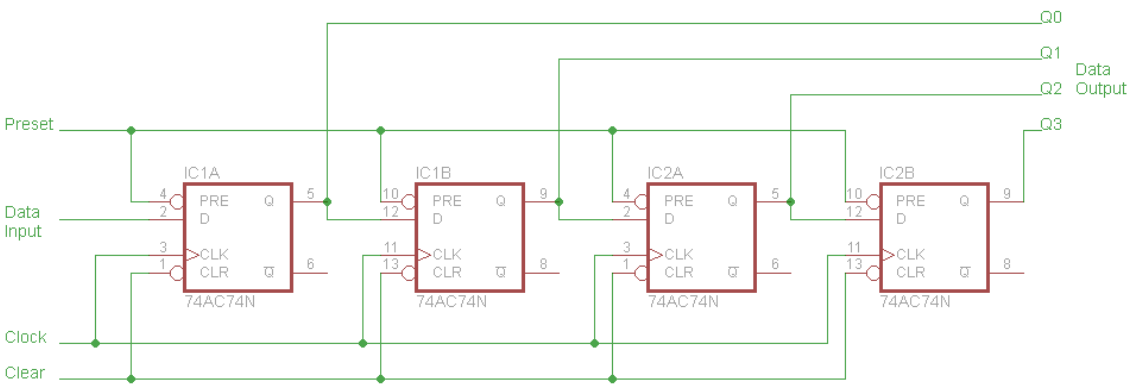


그림 7-2 Serial-In Parallel-Out Shift Right Register

2. 링 카운터(ring counter / Johnson counter)

쉬프트 레지스터를 응용한 간단한 카운터로서 직렬입력, 병렬출력 쉬프트 레지스터의 최종 출력을 다시 입력에 귀환시킨 순환 쉬프트 레지스터이다.

아래의 회로에서 네 개의 플립플롭을 모두 0으로 초기화 시킨 후에 클럭 펄스를 공급하면 $Q_0Q_1Q_2Q_3$의 상태는 0000, 1000, 1100, 1110, 1111, 0111, 0011, 0001, 0000...을 반복하며 한 번에 한 비트씩만 변하는 Gray Code를 만들어 낸다.

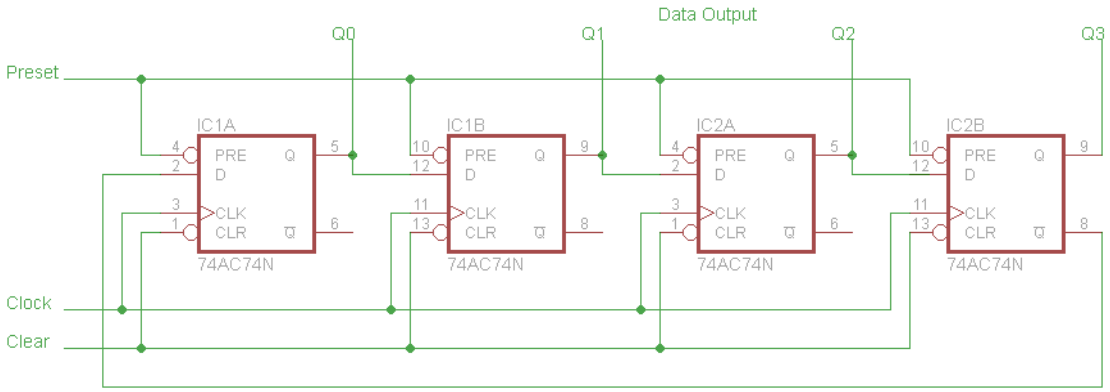


그림 7-3 Johnson Counter

3. Pseudo Random Number Generator

난수발생기라고 불리며, 난수 발생이 결과적으로 주기적으로 반복되기 때문에 pseudo random이라 이름 붙여졌다. 그리고 N개의 플립플롭으로 발생시킬 수 있는 난수의 개수는 2^N-1개이다.

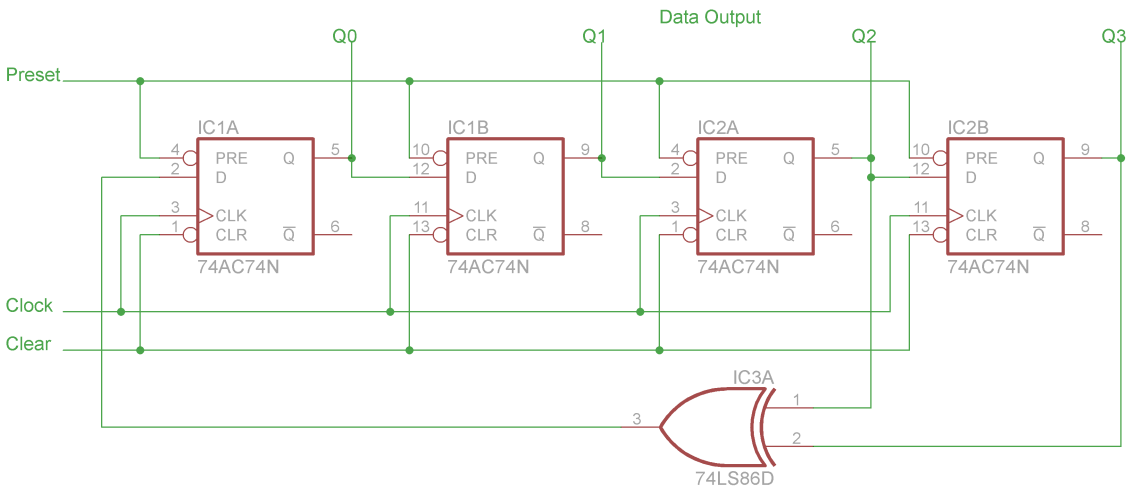


그림 7-4 Pseudo Random Number Generator

실험 순서

예비 실험 SN74LS194(4-Bit Bidirectional Universal Shift Register)의 기능 분석

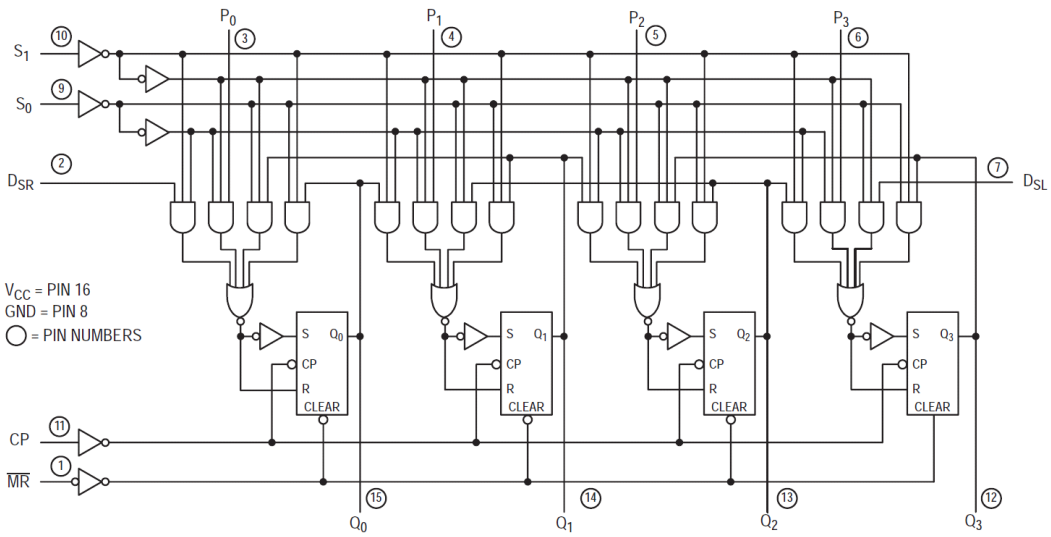
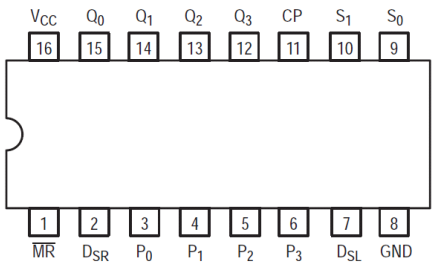


그림 7-5 74LS194 Logic Diagram



PIN NAMES

S_0, S_1	Mode Control Inputs
$P_0 - P_3$	Parallel Data Inputs
D_{SR}	Serial (Shift Right) Data Input
D_{SL}	Serial (Shift Left) Data Input
CP	Clock (Active HIGH Going Edge) Input
$\overline{MR}$	Master Reset (Active LOW) Input
$Q_0 - Q_3$	Parallel Outputs

그림 7-6 74LS194 Connection Diagram

OPERATING MODE	INPUTS						OUTPUTS			
	$\overline{MR}$	S_1	S_0	D_{SR}	D_{SL}	P_n	Q_0	Q_1	Q_2	Q_3
Reset	L	X	X	X	X	X	L	L	L	L
Hold	H	l	l	X	X	X	q_0	q_1	q_2	q_3
Shift Left	H H	h h	l l	X X	l h	X X	q_1 q_1	q_2 q_2	q_3 q_3	L H
Shift Right	H H	l l	h h	l h	X X	X X	L H	q_0 q_0	q_1 q_1	q_2 q_2
Parallel Load	H	h	h	X	X	P_n	P_0	P_1	P_2	P_3

p_n (q_n) = Lower case letters indicate the state of the referenced input (or output) one set-up time prior to the LOW to HIGH clock transition.

그림 7-7 74LS194 Mode Select

실험 A 4-bit shift register

1. 그림 7-8과 같이 74LS194를 사용하여 4-bit shift register 회로를 구성하고, /CLR 입력을 사용하여 네 개의 플립플롭을 모두 clear 시킨 후, QA, QB, QC, QD 출력을 측정하여 표 7-1의 1번 행에 기록한다.

2. Parallel Load 기능
   1) 74LS194를 parallel input mode로 설정
   2) A, B, C, D에 "0100"의 값을 입력
   3) 하나의 클럭 펄스를 공급
   4) QA, QB, QC, QD 출력을 측정하여 표 7-1의 2번 행에 기록한다.

3. Shift Right 기능
   1) 74LS194를 shift right mode로 설정
   2) SR에 '1'의 값을 입력
   3) 하나의 클럭 펄스를 공급하여 레지스터의 내용을 오른쪽으로 1회 shift 시킨다.
   4) QA, QB, QC, QD 출력을 측정하여 표 7-1의 3번 행에 기록한다.

4. Shift Right 기능
   1) 74LS194를 shift right mode로 설정
   2) SR에 '0'의 값을 입력
   3) 네 개의의 클럭 펄스를 공급하여 레지스터의 내용을 오른쪽으로 4회 shift 시킨다.
   4) 매 클럭 펄스를 가하고 난 후의 QA, QB, QC, QD 출력을 표 7-1의 4번 행~7번 행에 기록한다.

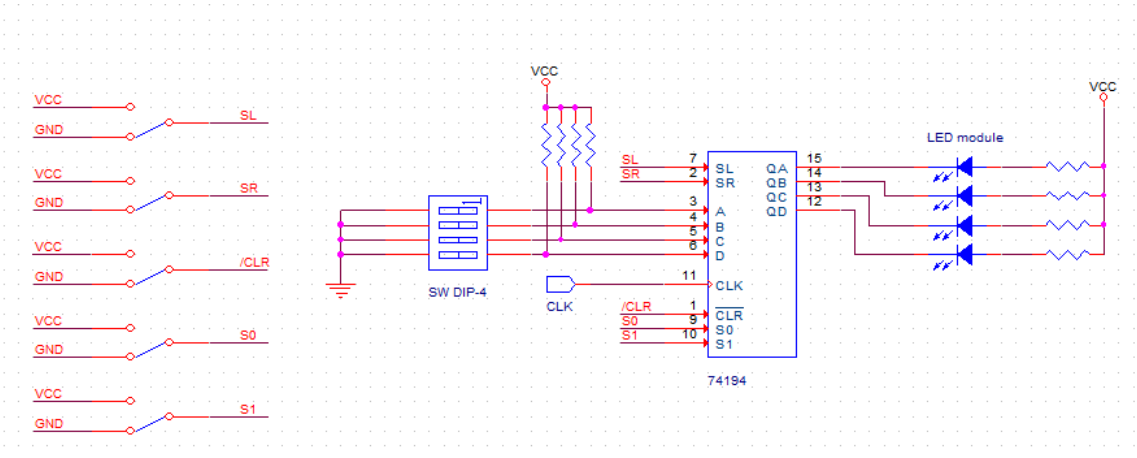


그림 7-8

실험 B Pseudo Random Number Generator

74LS194와 74LS86을 이용하여 그림 7-9와 같이 회로를 구성한다. DIP switch를 사용하여 QA, QB, QC, QD에 “1000” 값이 저장되도록 한다. Shift register를 shift right mode로 설정한 뒤, clock pulse를 공급하여 한 비트씩 shift 시키면서 QA, QB, QC, QD의 출력 값을 표 7-2에 기록한다.

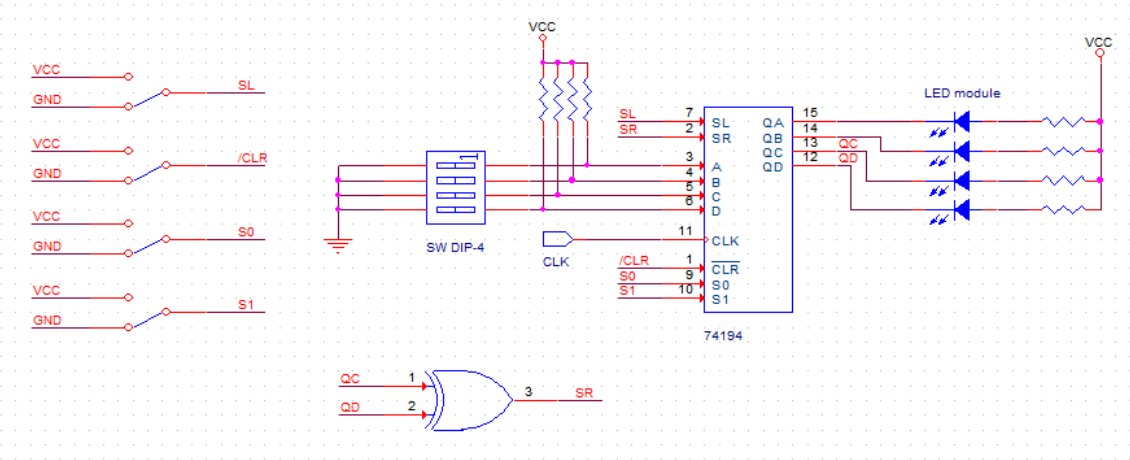


그림 7-9

실험 C FPGA/VHDL을 이용한 4-bit Parallel-in Parallel-out Register 구현

1. 그림 7-10과 표 7-1에 나타낸 바와 같이 Asynchronous Master Reset ($\overline{MR}$), Synchronous Load ($\overline{LD}$), Parallel Input (P3, P2, P1, P0) 입력 신호와 Parallel Out (Q3, Q2, Q1, Q0) 출력 신호를 가지는 4-Bit Parallel-In Parallel-Out Register (PIPOR)를 일반 논리 회로 소자를 사용하여 설계하고, Logisim을 사용하여 검증하시오.

2. FPGA/VHDL을 사용하여 이 PIPOR을 구현하시오. 입출력 핀 할당(pin mapping)은 아래의 그림을 참조하시오.

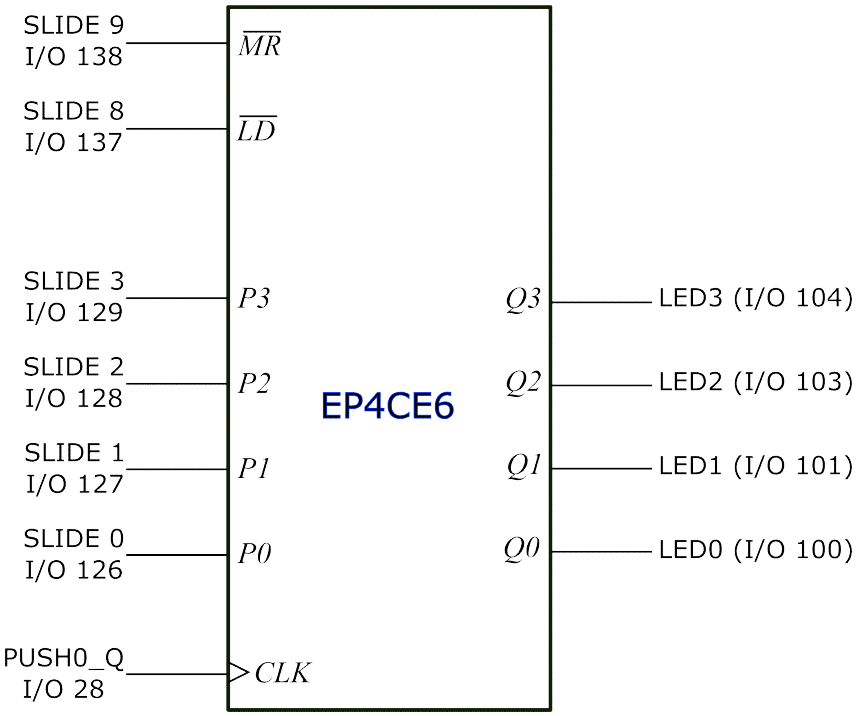


그림 7-10 FPGA로 구현된 4-bit Parallel-in Parallel-out Register

MR	CLK	LD	Q3,Q2,Q1,Q0
L	X	X	L,L,L,L
H	↑	H	No change
H	↑	L	P3,P2,P1,P0

(H: High level, L: Low level, X: Don't care, ↑: Rising-edge)

표 7-1 4-bit Parallel-in Parallel-out Register의 기능

- **구현된 PIPOR의 기능 확인 실험**

1. MR 입력을 'L'로 설정하여 네 개의 플립플롭 Q3, Q2, Q1, Q0를 모두 clear 시킨 후, Q3, Q2, Q1, Q0의 출력을 관찰하여 표 7-4에 기록한다.

2. Hold 기능 확인
   1) MR 입력을 'H'로 설정
   2) LD 입력을 'H'로 설정
   3) P3, P2, P1, P0 입력 핀에 "0101"을 입력
   4) CLK 입력에 하나의 클럭 펄스를 공급
   5) Q3, Q2, Q1, Q0의 출력을 측정하여 표 7-4에 기록한다.

3. Parallel Load
   1) MR 입력을 'H'로 설정
   2) LD 입력을 'L'로 설정
   3) P3, P2, P1, P0 입력 핀에 "0101"을 입력
   4) CLK 입력에 하나의 클럭 펄스를 공급
   5) Q3, Q2, Q1, Q0의 출력을 측정하여 표 7-4에 기록한다.

실험 D FPGA/VHDL을 이용한 4-bit Bidirectional Universal Shift Register(BUSR) 구현

- FPGA/VHDL을 사용하여 74LS194와 동일한 동작을 하도록 4-bit Bidirectional Universal Shift Register (BUSR)를 구현하시오.

- 입출력 핀 할당(pin mapping)은 아래의 그림을 참조하시오.

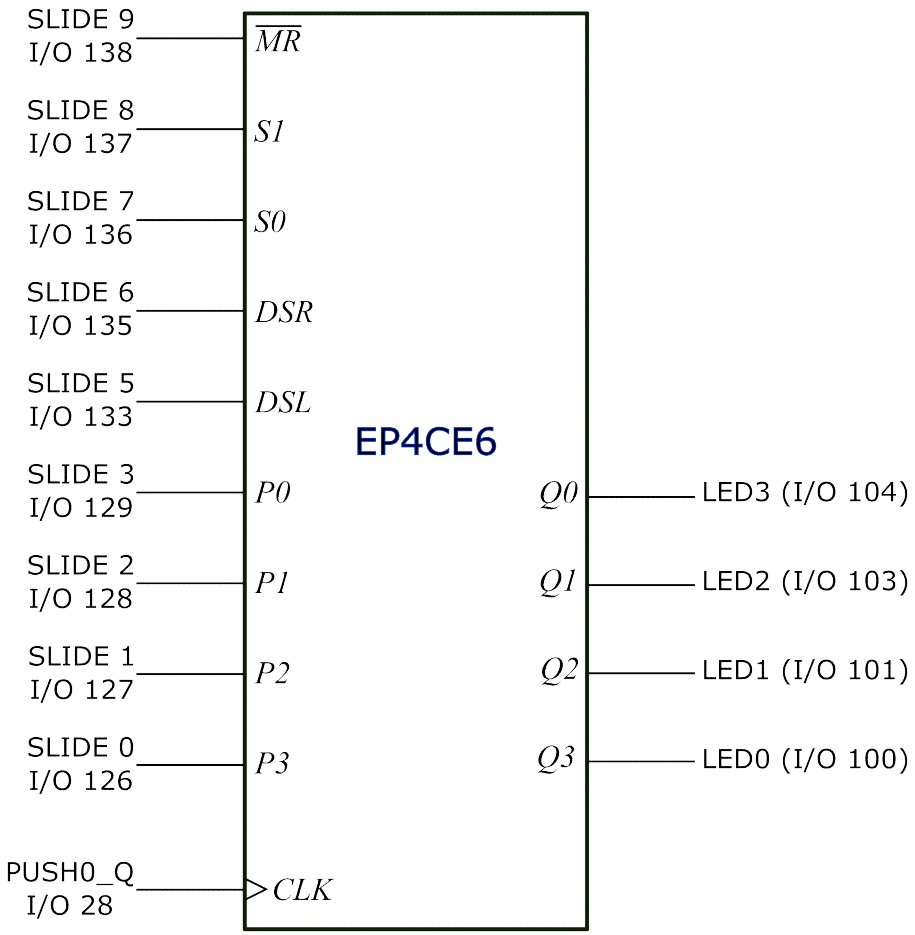


그림 7-11 FPGA로 구현된 4-bit Bidirectional Universal Shift Register (BUSR)

- 다음 쪽에 설명된 기능 확인 절차에 따라 구현된 BUSR의 기능을 검증할 수 있는 Simulation 파형을 작성 후, Functional Simulation을 수행하고, 그 결과를 검토한 후, 이상이 없으면 FPGA 소자에 프로그램한 후, 아래의 절차대로 다섯 가지 기능을 모두 확인하시오.

- **구현된 BUSR의 기능 확인 실험**

1. $\overline{MR}$ 입력을 사용하여 네 개의 플립플롭 Q0, Q1, Q2, Q3 모두 clear 시킨 후, Q0, Q1, Q2, Q3 출력을 측정하여 표 7-5에 기록한다.

2. Parallel Load
   1) 구현한 BUSR을 Parallel Load mode로 설정
   2) P0, P1, P2, P3 입력 핀에 "0100"의 값을 입력
   3) 하나의 클럭 펄스를 공급
   4) Q0, Q1, Q2, Q3 출력을 측정하여 표 7-5에 기록한다.

3. Shift Right 기능
   1) 구현한 BUSR을 Shift Right mode로 설정
   2) DSR에 '1'의 값을 입력
   3) 하나의 클럭 펄스를 공급하여 레지스터의 내용을 오른쪽으로 1회 shift 시킨다.
   4) Q0, Q1, Q2, Q3 출력을 측정하여 표 7-5에 기록한다.

4. Shift Right 기능
   1) 구현한 BUSR을 Shift Right mode로 설정
   2) DSR에 '0'의 값을 입력
   3) 네 개의 클럭 펄스를 공급하여 레지스터의 내용을 오른쪽으로 4회 shift 시킨다.
   4) 매 클럭 펄스를 가하고 난 후 Q0, Q1, Q2, Q3 출력을 측정하여 표 7-5에 기록한다.

5. Shift Left 기능
   1) 구현한 BUSR을 Shift Left mode로 설정
   2) DSL에 '1''의 값을 입력
   3) 하나의 클럭 펄스를 공급하여 레지스터의 내용을 왼쪽으로 1회 shift 시킨다.
   4) Q0, Q1, Q2, Q3 출력을 측정하여 표 7-5에 기록한다.

6. Shift Left 기능
   1) 구현한 BUSR을 Shift Left mode로 설정
   2) DSL에 '0'의 값을 입력
   3) 네 개의의 클럭 펄스를 공급하여 레지스터의 내용을 왼쪽으로 3회 shift 시킨다.
   4) 매 클럭 펄스를 가하고 난 후 Q0, Q1, Q2, Q3 출력을 측정하여 표 7-5에 기록한다.

7. Hold 기능
   1) 구현한 BUSR을 Hold mode로 설정
   2) 클럭 펄스를 가하고 난 후 Q0, Q1, Q2, Q3 출력을 측정하여 표 7-5에 기록한다.

8. MR 입력을 사용하여 네 개의 플립플롭 Q0, Q1, Q2, Q3 모두 clear 시킨 후, Q0, Q1, Q2, Q3 출력을 측정하여 표 7-5에 기록한다.

실험 E FPGA/VHDL을 이용한 링 카운터(ring counter / Johnson counter) 구현

1. 아래에 설명된 기능 확인 절차에 따라 구현된 Ring Counter의 기능을 검증할 수 있는 Simulation 파형을 작성 후, Functional Simulation을 수행하고, 그 결과를 검토하여 이상이 없으면 FPGA 소자에 프로그램한 후, 아래의 절차대로 출력 Q0, Q1, Q2, Q3를 확인하시오.

1) FPGA/VHDL을 사용하여 그림 7-3과 그림 7-12와 같은 Johnson Counter를 구현한 후
2) $\overline{\text{PRE}}$ 입력을 '1'로 설정하고 $\overline{\text{CLR}}$ 입력을 '0'으로 설정하여 Q0, Q1, Q2, Q3를 모두 clear한다.
3) $\overline{\text{PRE}}$ 입력과 $\overline{\text{CLR}}$ 입력을 모두 '1'로 설정
4) 매 클럭이 입력될 때마다 Q0, Q1, Q2, Q3의 출력 값을 표 7-6에 기록한다.

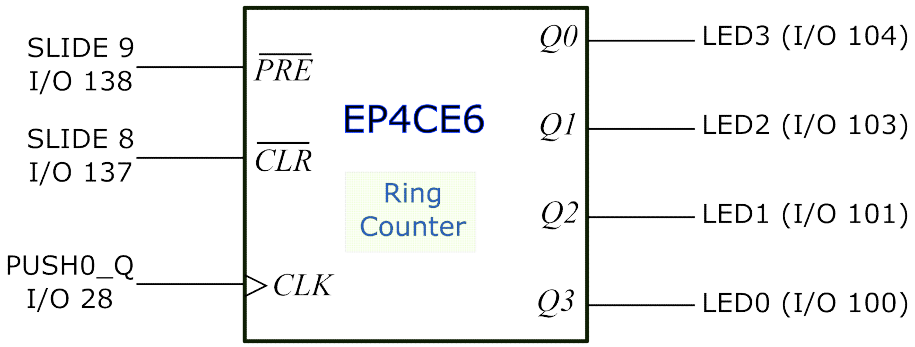


그림 7-12 FPGA로 구현된 Ring Counter

실험 보고서

데이터 및 관찰 내용

실험 A 4-bit shift register

행 번호	동작	P0, P1, P2, P3	SR	Q0	Q1	Q2	Q3
1	clear	-	-				
2	parallel load	0100	-				
3	Shift Right	-	1				
4	Shift Right	-	0				
5	Shift Right	-	0				
6	Shift Right	-	0				
7	Shift Right		0				

표 7-2

실험 B PRBS(pseudo-random binary sequence)

PRBS outputs table				
Number clock pulse	Outputs			
	QA	QB	QC	QD
0	1	0	0	0
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				
17				
18				
19				
20				
21				
22				
23				
24				
25				
26				
27				
28				
29				
30				
31				

표 7-3

실험 C FPGA로 구현한 4-bit Parallel-in Parallel-out Register

아래 표에 나타낸 각 입력 신호를 공급했을 때의 출력 신호(Q3,Q2,Q1,Q0)의 값을 관찰하여 표에 기록하고, 각 행의 동작이 무엇인지를 비고란에 쓰시오.

번호	MR	CLK	LD	P3,P2,P1,P0	Q3,Q2,Q1,Q0	비 고
1	L	X	X	X X X X		
2	H	↑	H	0 1 0 1		
3	H	↑	L	0 1 0 1		

표 7-4

실험 D FPGA로 구현한 4-bit Bidirectional Universal Shift Register(BUSR)

아래의 표에서 모든 입력 항목에는 주어진 동작을 수행하기 위한 값을 '0', '1' 또는 'x' (don't care) 중 하나를 쓰고, 출력 항목에는 클럭 신호가 공급되고 난 후의 관찰된 값을 '0' 또는 '1'로 쓰시오.

번호	동작	S1,S0	P0, P1, P2, P3	$\overline{MR}$	DSR	Q0	Q1	Q2	Q3	DSL
1	Clear									
2	Parallel Load		0100							
3	Shift Right				1					
4	Shift Right				0					
5	Shift Right				0					
6	Shift Right				0					
7	Shift Right				0					
8	Shift Left									1
9	Shift Left									0
10	Shift Left									0
11	Shift Left									0
12	Hold									
13	Clear									

표 7-5

실험 E FPGA/VHDL을 이용한 링 카운터(Ring counter / Johnson counter) 구현

아래의 표에서 모든 입력 항목에는 주어진 동작을 수행하기 위한 값을 '0' 또는 '1' 중 하나를 쓰고, 출력 항목에는 클럭 신호가 공급되고 난 후의 관찰된 값을 '0' 또는 '1'로 쓰시오.

Clock pulse	PR	CLR	Q0	Q1	Q2	Q3
1						
2						
3						
4						
5						
6						
7						
8						

표 7-6

결과 및 결론