실험 4 PLD/FPGA를 이용한 조합논리회로 설계 Ⅱ

실험 목표

디코더(decoder),인코더(encoder),멀티플렉서(multiplexer),디멀티플렉서(demultiplexer)의 동작원리를 이해하고, 'Quartus Ⅱ' software와 FPGA 소자를 사용하여 구현한다.

실험 부품

FPGA 실험 보드(EP4CE6)
USB Blaster Ⅱ
Quartus Ⅱ

관련 이론

1. 인코더(encoder)

인코더는 부호화되지 않은 2^n(또는 그 보다 작은)비트의 입력을 받아서 이를 n 비트로 부호화하여 출력하는 조합논리회로이다. 그림 4-1는 4비트의 입력을 받아 2비트로 부호화하여 출력하는 4-to-2 인코더 회로이며, 이에 대한 진리치표는 표 4-1과 같다. 이를 보면 인코더는 한 가지 제한을 갖는데, 그것은 오직 한 입력만이 1이 될 수 있다는 것이다. 만약 두 입력이 동시에 1이 되면 출력은 정의되지 않은 조합을 만들어 내게 된다.

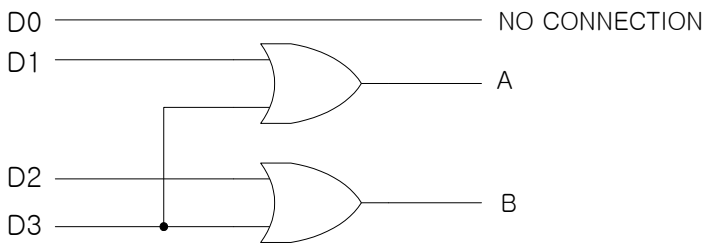


그림 4-1 encoder의 구조

D3	D2	D1	D0	B	A
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

표 4-1 encoder의 진리치표

2. 디코더(decoder)

디코더는 인코더와는 반대로 부호화 된 n 비트의 입력을 받아 2^n(혹은 그 보다 작은) 비트로 출력하는 조합논리회로이다. 그림 4-2는 부호화 된 2비트의 입력을 받아 4비트로 복

호(해독)하여 출력하는 2-to-4 디코더 회로이며, 이에 대한 진리치표는 표 4-2과 같다. 이를 보면 한 출력만이 1이 될 수 있기 때문에 서로 배타적인 것을 알 수 있다.

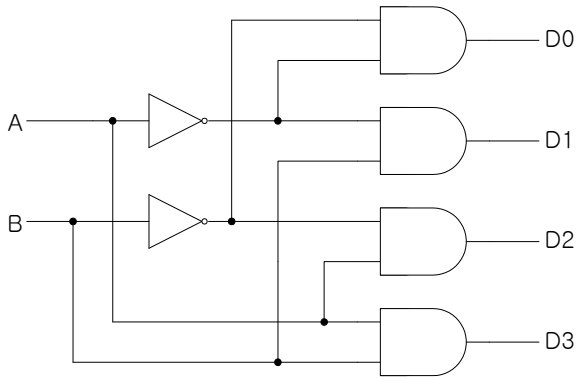


그림 4-2 decoder의 구조

A	B	D3	D2	D1	D0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

표 4-2 decoder의 진리치표

3. 멀티플렉서(multiplexer)

디지털 멀티플렉서(digital multiplexer)는 MUX라고도 불리며, 여러 개의 입력 가운데 하나를 선택하여 단일 출력으로 내보내는 조합논리 회로로서 선택기이다. 여러 개의 입력 가운데 특정한 하나의 입력을 선택하기 위한 '선택선(select)'이 존재하며, 일반적으로 2^n개의 입력선에 대해 n개의 선택선이 필요하다. 이들 선택선들의 비트 조합에 의해 입력선의 선택이 결정된다. 이때 선택 비트(select bit)를 만들어 내기 위해 디코더(decoder)가 이용된다.

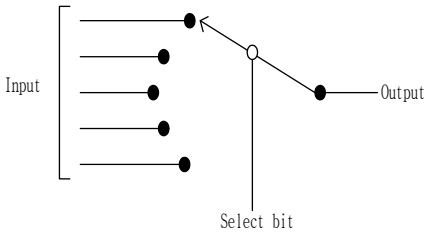


그림 4-3 MUX의 개념

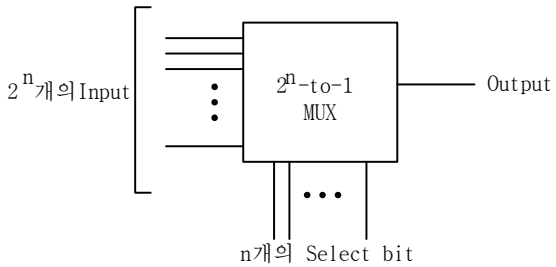


그림 4-4 MUX의 구조

4-to-1 멀티플렉서를 아래에 나타냈다.

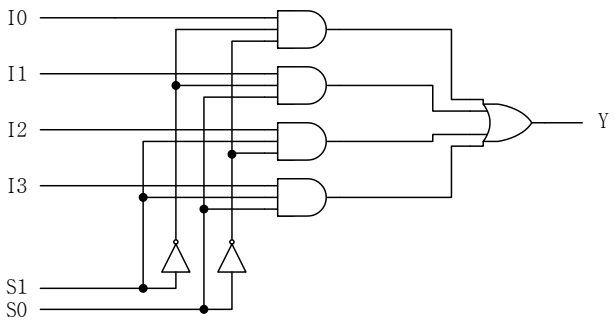


그림 4-5 Logic diagram

S1	S0	Y
0	0	I0
0	1	I1
1	0	I2
1	1	I3

표 4-3 Function table

4. 디멀티플렉서(demultiplexer)

디지털 디멀티플렉서(digital demultiplexer)는 DEMUX라고도 불리며, 멀티플렉서와는 반대로 하나의 입력선을 통해 들어오는 입력 정보를 여러 개의 출력선 가운데 하나를 선택하여 출력하는 조합논리 회로로서 분배기이다. 여기에도 선택을 위한 선택선이 존재한다. 일반적으로 2^n개의 출력선에 대해 n개의 선택선이 필요하며, 이들 선택선들의 비트 조합에 의해 출력선의 선택이 결정된다.

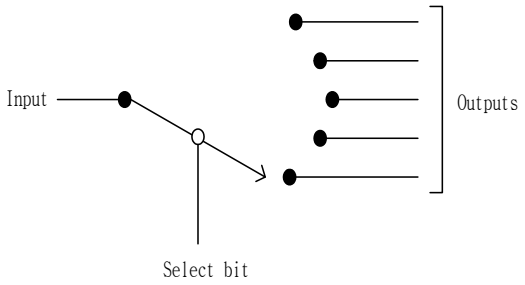


그림 4-6 DEMUX의 개념

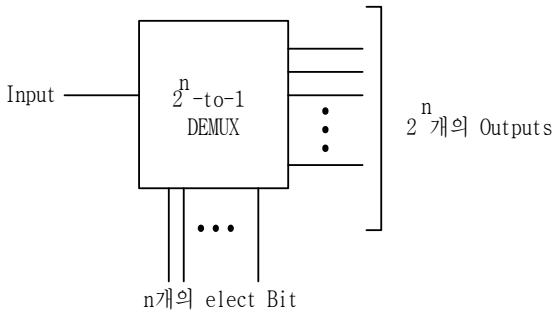


그림 4-7 DEMUX의 구조

1-to-4 디멀티플렉서를 아래에 나타냈다.

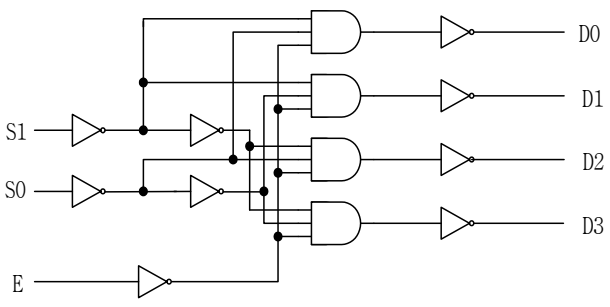


그림 4-8 Logic diagram

S1	S0	D0	D1	D2	D3
0	0	E	0	0	0
0	1	0	E	0	0
1	0	0	0	E	0
1	1	0	0	0	E

표 4-4 Function table

실험 순서

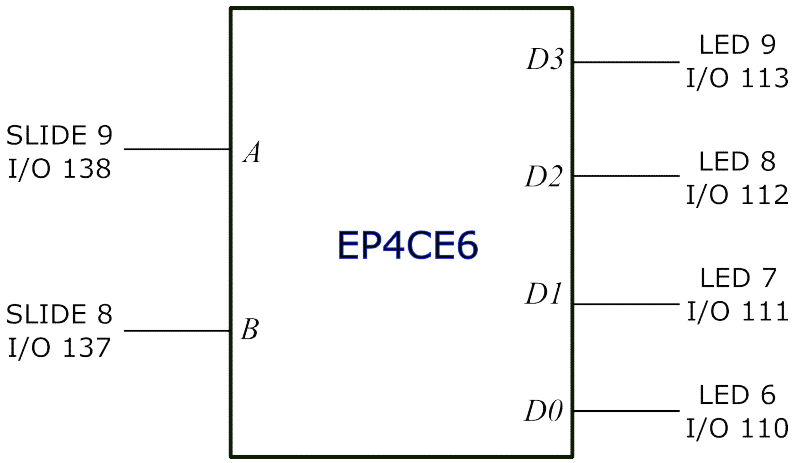
실험 A 2-to-4 Decoder

아래의 그림과 같은 2-to-4 decoder를 VHDL로 설계한 후, 이를 FPGA에 프로그램하여 동작시켜 본다. 단, VHDL로 decoder 설계시

(1) Concurrent signal assignment statement를 사용한다. (부울 대수식만을 사용)

(2) Conditional signal assignment statement를 사용한다.（`when ~ else`）

(3) Selected signal assignment statement를 사용한다.（`with ~ select ~ when`）



입력		출력			
A	B	D3	D2	D1	D0
0	0				
0	1				
1	0				
1	1				

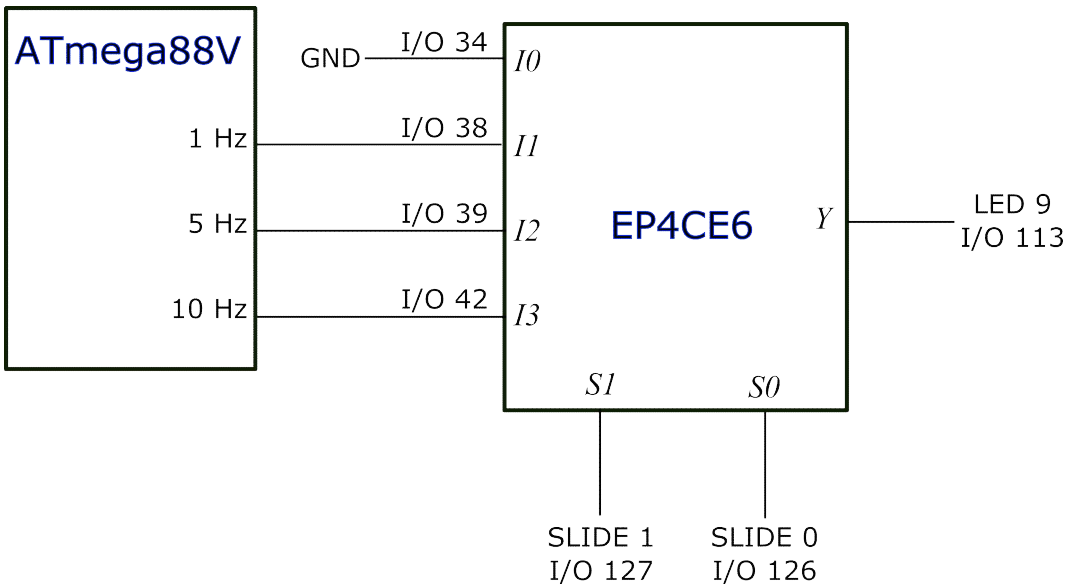
실험 B 멀티플렉서 (Multiplexer)

아래의 그림과 같은 4-to-1 Multiplexer를 VHDL로 설계한 후, 이를 FPGA에 프로그램하여 동작시켜 본다. 단, VHDL로 multiplexer 설계시

(1) Concurrent signal assignment statement를 사용한다. (부울 대수식만을 사용)

(2) Conditional signal assignment statement를 사용한다. (`when ~ else`).

(3) Selected signal assignment statement를 사용한다. (`with ~ select ~ when`).



입력		출력
S1	S0	Y
0	0	( )Hz
0	1	( )Hz
1	0	( )Hz
1	1	( )Hz

실험 C 2-bit 비교기 (2-bit Comparator)

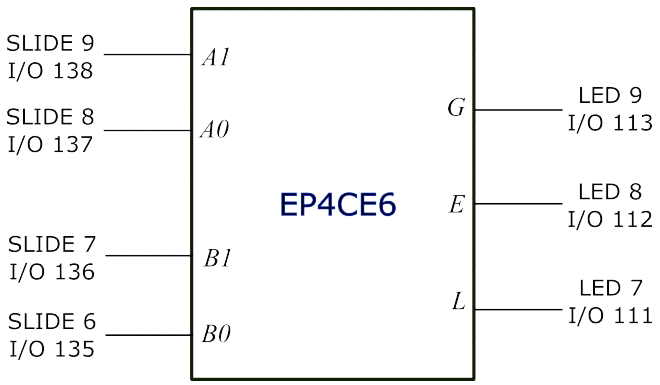
다음 그림과 같이 각각 2-비트로 구성된 두 수 A와 B의 크기를 비교하여

a) 두 수의 크기가 같으면 'E' 출력을 '1' 로,

b) A가 B보다 크면 'G' 출력을 '1' 로,

c) A가 B보다 작으면 'L' 출력을 '1' 로

출력하는 2-비트 비교기를 VHDL로 설계한 후, 이를 FPGA에 프로그램하여 동작시킨다. 단, A1과 B1은 상위 비트이고 A0와 B0는 하위 비트임.



(1) 비교기의 진리치표 작성

위에서 설명한 비교기의 기능이 수행되도록 아래의 진리치표를 완성한다.

입력				출력		
A1	A0	B1	B0	G	E	L
0	0	0	0			
0	0	0	1			
0	0	1	0			
0	0	1	1			
0	1	0	0			
0	1	0	1			
0	1	1	0			
0	1	1	1			
1	0	0	0			
1	0	0	1			
1	0	1	0			
1	0	1	1			
1	1	0	0			
1	1	0	1			
1	1	1	0			
1	1	1	1			

(2) Concurrent signal assignment statement를 사용하여 위의 비교기를
   a) 부울 대수식만을 사용하여 VHDL로 구현하고,
   b) ModelSim을 사용하여 Simulation 한 후,
   c) EP4CE 실험 보드에 프로그램을 다운로드 하여 비교기의 기능을 시험한다.

(3) Conditional signal assignment statement(`when ~ else`)를 사용하여 위의 비교기 구현.
   a) A와 B를 정수(`integer`)로 선언하여 VHDL로 구현하고,
   b) ModelSim을 사용하여 Simulation 한 후,
   c) EP4CE 실험 보드에 프로그램을 다운로드 하여 비교기의 기능을 시험한다.

결과 및 결론