

로직웍스(Logic Works) 5 사용법

- 01. 로직웍스 소개와 화면 구성
- 02. 회로 그리기 및 시뮬레이션 실행 과정
- 03. 로직웍스에서 지원하는 입출력 소자
 - 04. 부품 라이브러리 제작
- 05. 부품 라이브러리 제작 예
- 06. 기타 사항



로직웍스 소개와 화면 구성

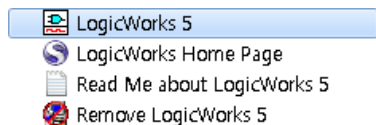
1 로직웍스 소개

로직웍스에서 제공하는 소자로는 74 시리즈, 입출력 및 전원 관련 소자, 아날로그 소자, 디지털 게이트, 조합논리회로, 스파이스(spice) 회로, VHDL(VHSIC Hardware Description Language) 등을 사용할 수 있다. 드래그 앤 드롭(drag and drop) 형식으로 모든 소자를 가져다 쓸 수 있어서 사용하기 편리하다. 또한 회로의 최종 부분에서뿐만 아니라 회로의 중간에서도 중간 결과를 볼 수 있도록 되어 있어서 시뮬레이션 도중에 쉽게 디버그할 수 있다.

좀 더 많은 정보는 로직웍스 홈페이지(<http://www.capilano.com>)에서 얻을 수 있다. 같은 회사의 제품인 디자인웍스(Design Works)는 기능이 훨씬 더 강력하다. 로직웍스의 윈도우용 데모 버전은 없으며, 필요하다면 디자인웍스의 데모 버전을 받아서 실행해 보는 것도 좋은 방법이다. 30일 데모 버전이지만 모든 기능을 다 사용할 수 있다. 로직웍스로 작성한 파일은 디자인웍스에서도 완벽하게 호환된다.

로직웍스 5.0 버전에서는 VHDL도 가능하며, 자신만의 라이브러리를 구성할 수도 있다. 특히 다른 툴에 비해 사용 방법이 간단하며, 기능이 강력하다. 단시간에 익힐 수 있으며, 실제 실험하기 전에 회로를 구성하여 시뮬레이션해 볼 수 있어서 시행 착오를 줄일 수 있다. 여기서는 로직웍스의 사용법을 알아본다.

2 로직웍스 화면 구성



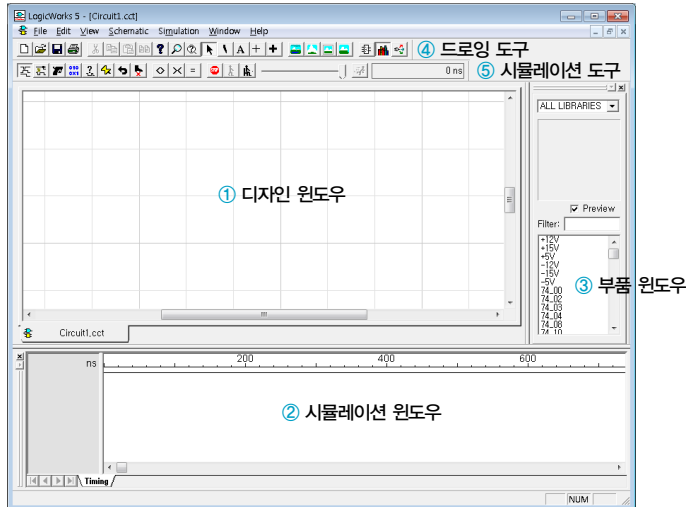
(a) 시작 메뉴에서 실행



(b) 아이콘

[그림 8-1] 로직웍스 실행

시작 메뉴에서 [LogicWorks 5]를 클릭하거나 아이콘을 더블클릭하여 로직웍스를 실행하면 다음과 같은 화면이 나온다.



[그림 B-2] 로직웍스 초기 화면

- ① 디자인 윈도우(Design Window) : 회로를 그리는 영역
- ② 시뮬레이션 윈도우(Simulation Window) : 회로의 시뮬레이션 결과를 출력하는 영역
- ③ 부품 윈도우(Part Window) : 사용할 부품(Part)을 선택하는 영역
- ④ 드로잉 도구(Drawing Tools) : 회로를 그리는 도구들로 구성된 툴 바(tool bar)
- ⑤ 시뮬레이션 도구(Simulation Tools) : 시뮬레이션 작업을 수행하는 툴 바

드로잉 툴 바의 주요 기능



[그림 B-3] 드로잉 툴 바

- ① 포인팅 커서(Pointing Cursor) : 부품을 선택하거나 선을 연결할 때 사용
- ② 삭제용 커서 : 잘못된 부분을 삭제할 때 사용
- ③ 텍스트 입력 커서 : 입출력 Port의 이름이나 레이블을 달 때 사용
- ④ 신호선 입력 커서 : 선을 연결할 때 사용
- ⑤ 버스 입력 커서 : 버스(여러 개의 묶음 선)를 연결할 때 사용

- ⑥ 디자인 윈도우 축소 보기(Zoom out)
- ⑦ 디자인 윈도우 확대 보기(Zoom in)
- ⑧ 윈도우 크기에 맞게 확대
- ⑨ 일반 크기로 확대

시뮬레이션 툴 바의 주요 기능

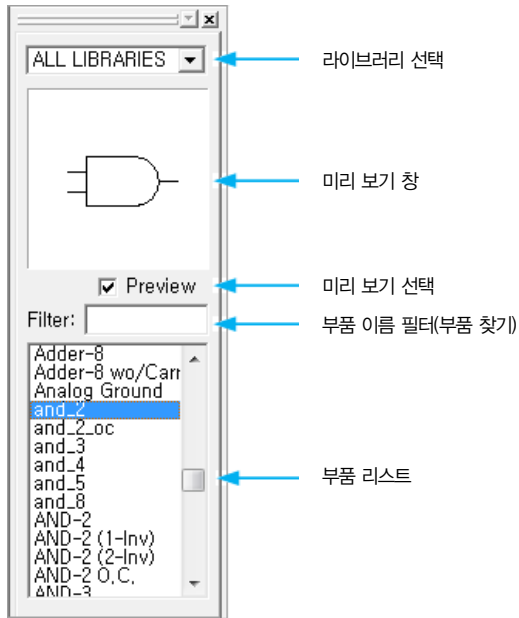


[그림 B-4] 시뮬레이션 툴 바

- ① Show/Hide Simulation : 시뮬레이션 결과 보이기/숨기기
- ② Add Signals to Timing : 시뮬레이션 윈도우에 다른 시그널(Signal) 추가
- ③ Trigger : 특정 신호 위주로 시뮬레이션
- ④ I/O panel : 입출력 현황을 볼 수 있음
- ⑤ Simulation Parameters : 클록의 파형 조절, 디바이스의 지연시간 조절에 사용
- ⑥ Stick/Unstick Signals : 0이나 1로 값을 고정시킴
- ⑦ Reset Simulator : 시뮬레이션 윈도우를 지움
- ⑧ Clear Unknown : 잘못된 부분을 지움
- ⑨ Zoom in Simulation Window : 시뮬레이션 윈도우를 확대
- ⑩ Zoom out Simulation Window : 시뮬레이션 윈도우를 축소
- ⑪ Normal Size : 시뮬레이션 윈도우를 정상 크기로 함
- ⑫ Stop Simulator : 시뮬레이터를 정지
- ⑬ Single Step Simulator : 싱글 스텝 시뮬레이터
- ⑭ Pause Stop Simulator : 시뮬레이터를 잠시 멈춤
- ⑮ Simulator Speed : 시뮬레이션 속도 조절용 바(bar)
- ⑯ Run Simulator : 시뮬레이션 수행

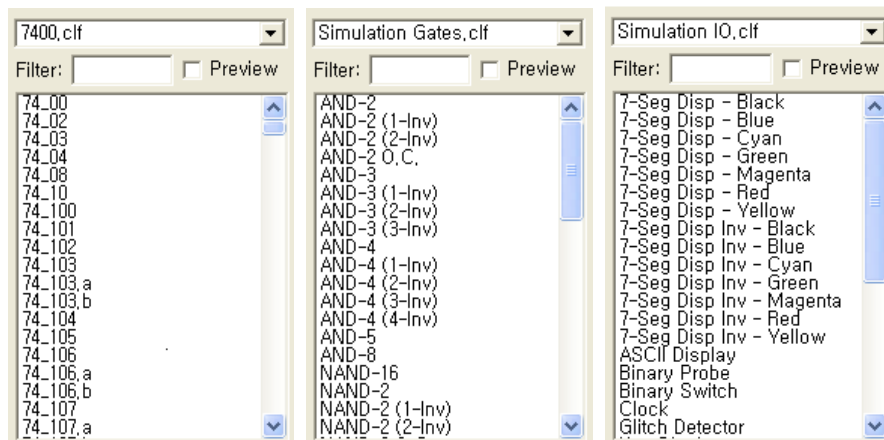
부품 윈도우의 주요 기능

부품 윈도우(Part Windows)는 시뮬레이션에서 사용할 부품(Part)을 선택하는 영역으로 구성 내용은 다음 그림과 같다. [ALL LIBRARIES]의 ▼를 누르면 로직웍스에서 제공하는 모든 라이브러리를 볼 수 있다.



[그림 B-5] 부품 윈도우

[그림 B-6]에는 주요 라이브러리의 리스트를 나타내었는데, ‘7400.clf’는 7400 TTL 게이트들이 있으며, ‘Simulation Gates.clf’에는 조합논리회로의 소자들이 있다. 또한 ‘Simulation IO.clf’에는 로직웍스에서 지원하는 입출력 소자들이 있다. 따라서 시뮬레이션하려는 회로의 부품이나 구성 요소를 부품 윈도우에서 찾아서 더블클릭하여 그린다.

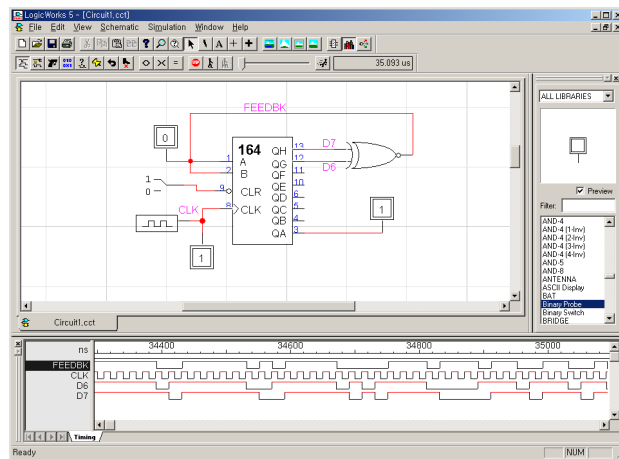


[그림 B-6] 부품 윈도우에서 라이브러리별 부품 보기



회로 그리기 및 시뮬레이션 실습 과정

그림과 같이 74164 소자를 이용한 간단한 예제를 통해 회로 그리기와 시뮬레이션을 실습해보자.

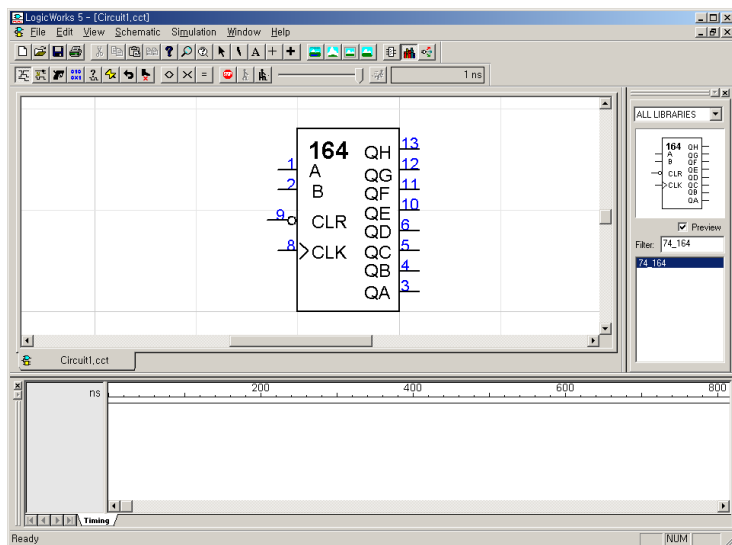


[그림 B-7] 74164를 이용한 예제 화면

1 부품 윈도우에서 74164 소자를 불러오기

- ① 부품 윈도우의 라이브러리 리스트에서 74164를 선택하기 위해 ‘7400.cdf’ 리스트에서 74_164소자를 찾거나, Filter 항목에 ‘74_164’를 입력한다. 찾은 부품을 더블클릭한다.
- ② 디자인 윈도우로 커서를 옮기고, 키보드의 화살표를 눌러 소자의 방향을 선택한다.
- ③ 적당한 위치를 지정하고 클릭하면 74_164 소자를 지정할 수 있다. 여러 소자를 그릴 경우 원하는 위치에 놓고 클릭하면 계속하여 소자가 그려진다.
- ④ 그리기를 멈추려면 스페이스 키를 누르거나 툴 바의 포인팅 커서를 선택하여 포인트 모드 상태로 전환한다.

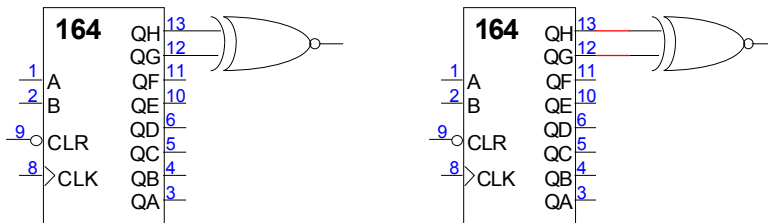
- ⑤ 또는 부품 윈도우의 부품 이름을 마우스로 누른 상태에서 디자인 윈도우로 갖다 놓으면 쉽게 하나의 부품만 그릴 수 있게 된다.



[그림 B-8] 디자인 윈도우에 부품 가져다 놓기

2. XNOR 게이트 삽입하기

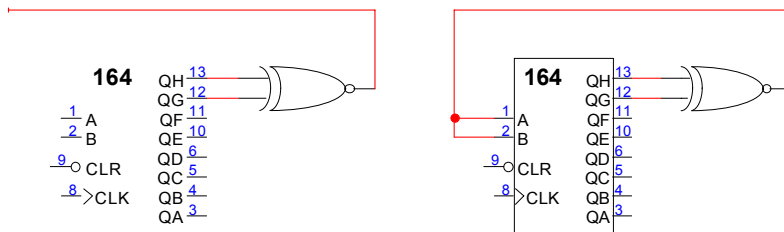
- ① 부품 윈도우에서 'Simulation Gates.clf'를 선택하여 'XNOR-2'를 더블클릭한다.
- ② 키보드의 화살표로 방향을 잡은 후 XNOR-2 게이트의 두 입력단과 74_164의 출력단 QG, QH의 끝이 닿게 하여 클릭한다.
- ③ 스페이스 키를 눌러 포인트 모드로 전환한다.
- ④ XNOR 게이트를 약간 오른쪽으로 드래그하여 74_164와 간격을 띄워준다.



[그림 B-9] XNOR 게이트를 선택하여 넣기

③ 시그널 연결하기

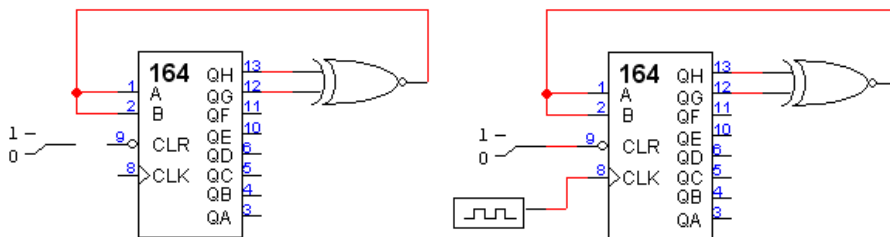
- ① XNOR 게이트의 출력단 끝을 드래그하여 시그널을 만든다.
- ② 74_164의 입력단 A에 연결한다. 이때 **[Alt]** 키나 **[Ctrl]** 키를 눌러 연결 방법을 다르게 할 수 있으며, 연결된 빨간색 시그널을 클릭하여 제대로 연결되었는지 확인할 수 있다.
- ③ 74_164의 입력단 B에서 시그널을 시작하여 A로의 입력 시그널에 연결한다.



[그림 B-10] 시그널 연결하기

④ 스위치와 클록 연결하기

- ① 부품 윈도우에서 'Simulation IO.clf'를 선택하여 'Binary Switch'를 더블클릭하고 74_164의 CLR단에 연결한다(스위치나 키패드 등 사용자 입력을 받는 소자를 드래그하여 이동할 경우에는 **[Shift]** 키를 누른 상태에서 드래그해야 한다).
 - ② 부품 윈도우에서 'Simulation IO.clf'를 선택하여 'Clock'을 더블클릭하고 74_164의 CLK단에 연결한다.
- (*주의 : 클록과 스위치 같은 소자를 입력하다 보면 자동으로 시뮬레이션이 실행되는 경우가 있다. 이때는 당황하지 말고 <시뮬레이터 정지(Stop Simulator)> 버튼을 누른다.)



[그림 B-11] 스위치와 클록 입력 연결하기

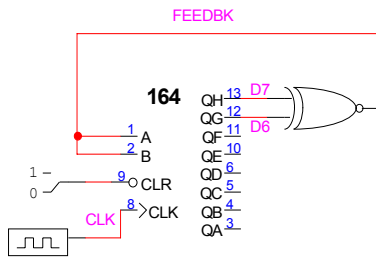
5 시그널에 레이블(Label) 붙이기

- ① 드로잉 톨 바에서 텍스트 입력 톨을 선택하면 커서는 연필 모양으로 변한다. 클록의 빨간색 시그널을 연필 모양의 커서 끝으로 클릭하여 레이블을 붙이고 레이블을 원하는 곳으로 드래그한다.



[그림 B-12] 텍스트 톨 선택

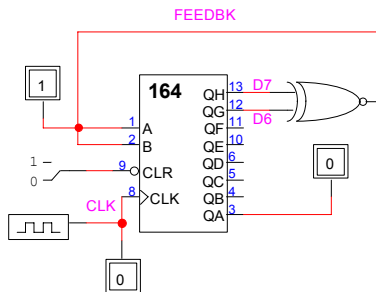
- ② 'CLK'를 입력하고 [Enter] 키를 누른다.
- ③ 레이블을 클릭하여 시그널과 연결 상태를 확인한다.
- ④ 이와 같은 방법으로 'FEEDBK', 'D7', 'D6' 등의 레이블을 붙인다(레이블이 붙은 시그널은 차후 시뮬레이션을 통해 그 값을 알아볼 수 있다).



[그림 B-13] 레이블 붙이기


6 프로브(Probe: 특정 부분의 시그널 검사) 설정

부품 윈도우에서 'Simulation IO.clf'를 선택하여 'Binary Probe'를 더블클릭한 후, 값을 알고자 하는 시그널에 연결한다.





[그림 B-14] 출력값을 관찰하기 위한 프로브 연결

7 시뮬레이션 실행

- ① 시뮬레이션 툴 바에서 시뮬레이션 실행(Run Simulation) 버튼()을 클릭하면 시뮬레이션이 시작되어 시뮬레이션 윈도우에 타이밍(Timing)이 그려진다.




[그림 B-15] 시뮬레이션 실행 버튼

- ②   버튼을 클릭하면, 타이밍 윈도우의 스케일이 확대 또는 축소된다. 해상도는 4 픽셀/시간단위(pixels/time unit)에서 100픽셀/시간단위로 변한다. 여기서 'time unit'은 1ns(나노초)로 생각하면 편리하다.

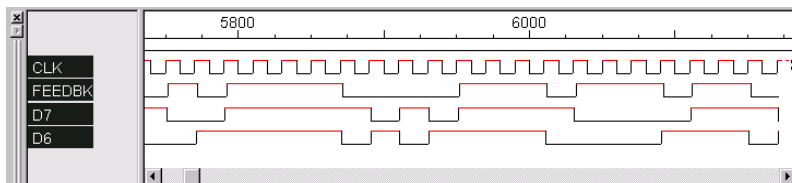


[그림 B-16] 타이밍 윈도우의 크기 확대/축소 버튼

- ③ 시뮬레이션을 0에서부터 다시 시작하려면  버튼을 클릭한다.




[그림 B-17] 시뮬레이션 재시작 버튼



[그림 B-18] 타이밍 윈도우

8 싱글 스텝(Single Step) 실행

시뮬레이션 툴 바에서 싱글 스텝 버튼()을 클릭하면 버튼을 한 번 누를 때마다 클록에 동기하여 한 주기씩 실행되어 시뮬레이션 윈도우에 타이밍이 그려진다.



[그림 B-19] 싱글 스텝 버튼


9. 시뮬레이션 속도 조절

시뮬레이션 툴 바에서 시뮬레이션 속도 조절기를 조절하면 시뮬레이션 속도가 조절된다.



[그림 B-20] 시뮬레이션 속도조절 버튼

10. 시뮬레이션 중단하기

시뮬레이션 툴 바에서 시뮬레이션 멈춤(Stop Simulator) 버튼()을 클릭하면 시뮬레이션이 중단된다.



[그림 B-21] 시뮬레이션 멈춤 버튼

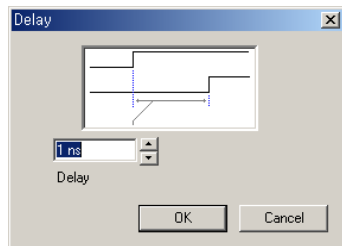
11. 소자의 파라미터 수정

① 먼저 X-NOR 게이트를 클릭하면, 다음과 같이 X-NOR이 선택된다.




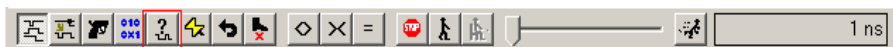
[그림 B-22] 게이트 선택

② [Simulation menu]-[Simulation Params...(Ctrl + K)] 메뉴를 선택하면 아래와 같은 창이 열린다.



[그림 B-23] 게이트의 지연시간 설정 화면

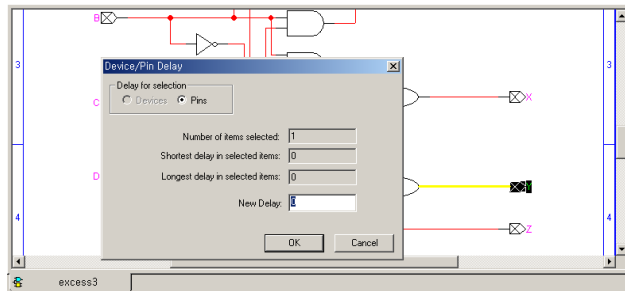
③ 이와 같은 동작은 시뮬레이션 툴 바에서  버튼을 클릭해도 된다.



[그림 B-24] 게이트 지연시간 설정 버튼

- ④ 핀(pin)의 지연시간(delay)은 보통 0으로 설정되어 있어 입력과 출력이 동일한 시점에서 변하는 모양을 관찰할 수 있기 때문에 이해하기 쉽다. 그러나 실제 회로의 동작에서는 어느 정도 지연시간이 존재하므로 정교한 회로의 동작을 고려하는 경우에는 실제 지연 시간을 입력하여 시뮬레이션해야 한다.

다음 그림과 같이 출력선과 출력 포트를 드래그하여 설정한 다음 핀의 지연시간을 설정할 수 있다.



[그림 B-25] 핀의 지연시간 설정 화면


12 회로 수정

- ① 삭제용 커서인 잭(ZAP: delete)을 사용하여 시그널 선을 지운다.
- ② 잭을 선택한 후, 지우려는 선 위에 커서를 놓고 클릭하면 삭제된다.
- ③ 바로 이전에 작업한 내용을 취소하려는 경우 [Edit]-[실행취소(Undo: **Ctrl** + **Z**)] 메뉴를 클릭한다.



[그림 B-26] 잭 툴 선택

13 회로 저장

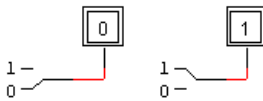
- ① 완성된 회로를 저장하기 위해서는 [File]-[Save] 메뉴를 선택하여 파일 이름을 입력하고 저장한다. 파일의 확장자는 '*.cct'다.
- ② 저장 버튼()을 클릭해도 된다.



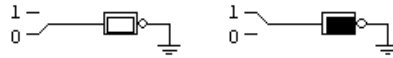
[그림 B-27] 저장 버튼

1 2진 스위치, 2진 프로브, LED

- ① 2진 스위치(binary switch)는 논리 0과 1을 입력하는 소자이며, 2진 프로브(binary probe)는 0과 1을 출력하는 소자이다. 마찬가지로 LED도 0과 1을 출력하는 소자이지만 출력이 0이면 LED는 off 상태, 1이면 on 상태가 된다.
- ② 2진 스위치는 토글(toggle) 형태의 스위치이므로 한번씩 누를 때마다 0에서 1로 또는 1에서 0으로 변환된다.



(a) 2진 스위치와 2진 프로브



(b) 2진 스위치와 LED

[그림 B-28] 2진 스위치, 2진 프로브, LED

- ③ 2진 프로브에서 가능한 형태의 출력은 [표 B-1]과 같다.

[표 B-1] 2진 프로브의 가능한 출력 형태

심볼	의미
0	정상적인 2진 데이터 '0'
1	정상적인 2진 데이터 '1'
Z	연결 없음. 높은 저항(high-impedance) 상태
X	미정. 이 프로브에 연결된 출력은 있으나 확실하지 않음
C	충돌. 이 프로브에 '0'과 '1'의 출력이 연결되어 충돌

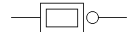
2진 스위치



2진 프로브



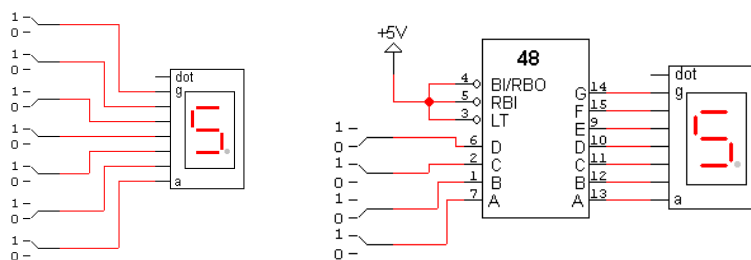
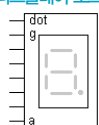
LED



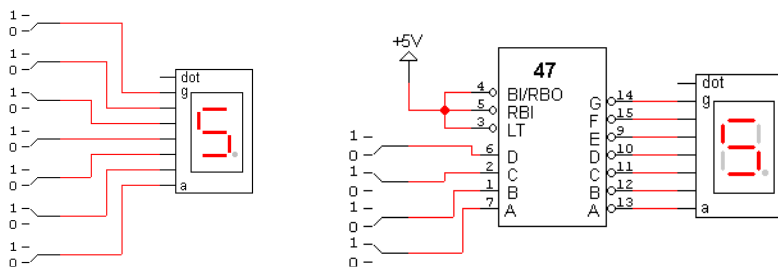
② 7-세그먼트 디스플레이 도트, 7-세그먼트 디스플레이 역 도트

- ① 7-세그먼트 디스플레이 도트(7-Seg Disp)와 7-세그먼트 디스플레이 역 도트(7-Seg Disp Inv)는 2진 데이터를 BCD로 출력하는 소자이다.
- ② 7-세그먼트 디스플레이 도트는 1일 때, 해당되는 LED가 on되는 캐소드-공통 형태의 소자이며, 7-세그먼트 디스플레이 역 도트는 0일 때, 해당되는 LED가 on되는 애노드-공통 형태의 소자이다.
- ③ 7-세그먼트와 디코더를 조합하여 사용할 경우, 7-세그먼트 디스플레이 도트는 7448과 함께 사용되며, 7-세그먼트 디스플레이 역 도트는 7447과 조합하여 사용한다.

7-세그먼트
디스플레이 도트



(a) 7-세그먼트 디스플레이 도트

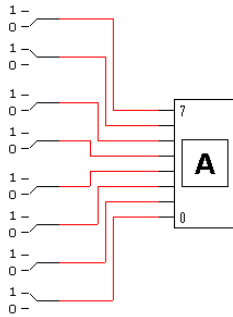


(b) 7-세그먼트 디스플레이 역 도트

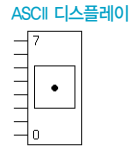
[그림 B-29] 7-세그먼트 디스플레이 동작방법

③ ASCII 디스플레이

- ① ASCII 디스플레이(ASCII Display)는 출력데이터를 ASCII 문자로 출력하는 소자이다.
- ② 예를 들어, 영문자 A의 ASCII 코드는 16진수로 41(2진수로는 0100 0001)이므로 아래와 같은 결과를 얻는다.

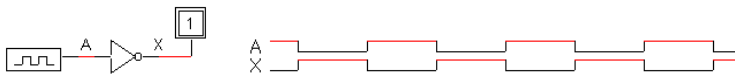


[그림 B-30] ASCII 디스플레이



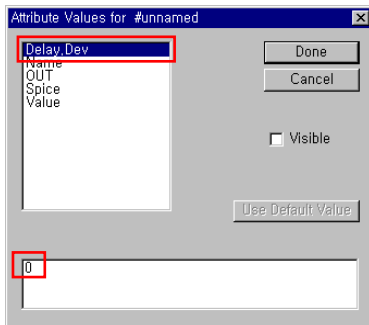
4. 클록

- ① 클록(Clock)은 구형파를 발생하는 입력 소자이다.
- ② 예를 들어, [그림 B-31]과 같은 NOT 게이트의 입력에 클록을 사용한 경우 출력은 반전된 결과를 얻을 수 있다. 여기서 클록의 0(Low)과 1(High)이 지속되는 시간은 기본적으로 각각 10, 10(단위는 ns)으로 설정되어 있다.



[그림 B-31] 클록 입력과 타이밍도

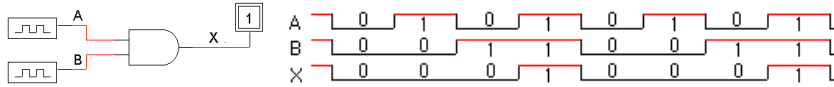
- ③ 이때 입력 A와 출력 X 사이에는 지연시간이 존재하는데, 이를 없애려면 NOT 게이트 위에서 마우스 오른쪽 버튼을 클릭하여 [Attributes...] 메뉴를 선택한다. [그림 B-32]와 같은 화면이 나타나면 'Delay.Dev'를 클릭하고 하단에 지연시간 0을 입력하면 된다.



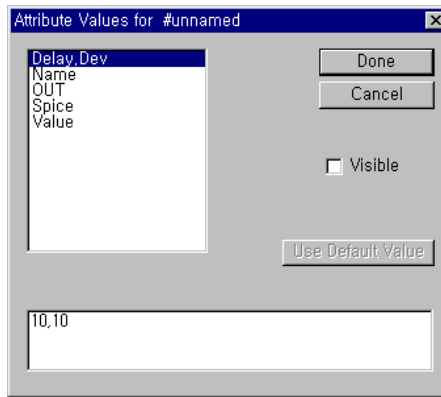
[그림 B-32] 사용자 지연시간 설정 윈도우



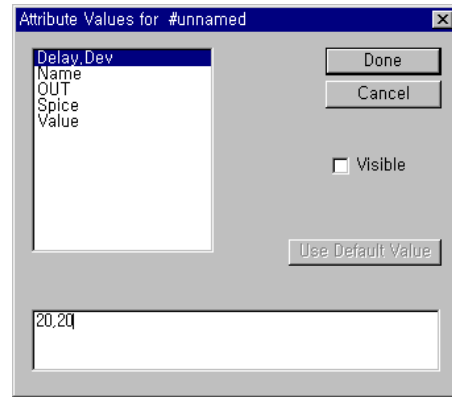
- ④ [그림 B-33]과 같은 AND 게이트의 동작을 시뮬레이션하는 경우, 입력 A와 B의 변화를 진리표에 나타난 형태대로 각각 00, 01, 10, 11의 순서로 하는 경우가 필요하다. 이를 위해 입력 A의 클록 그림 위에 커서를 놓고 오른쪽 마우스를 클릭하여 [Attributes...] 메뉴를 선택하면 [그림 B-34](a)와 같은 화면이 나타나면 ‘Delay.Dev’를 클릭하고 하단에 지연시간 10, 10을 입력한다. 마찬가지로 입력 B는 20, 20으로 입력하면 된다.



[그림 B-33] 클록의 타이밍을 다르게 설정하여 실행한 화면



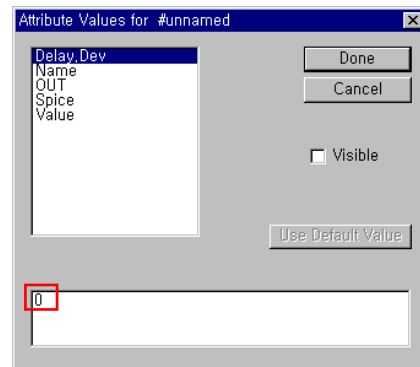
(a) 입력 A의 경우



(b) 입력 B의 경우

[그림 B-34] 클록 주기의 기본값 변경 윈도우

- ⑤ 이때 입력 A, B와 출력 X 사이에는 지연 시간이 존재하는데, 이를 없애려면 AND 게이트 위에 커서를 놓고 마우스 오른쪽 버튼을 클릭하여 [Attributes...] 메뉴를 선택하면 [그림 B-35]와 같은 화면이 보이는데, 여기서 ‘Delay.Dev’를 클릭하고 하단에 지연시간 0을 입력하면 된다.

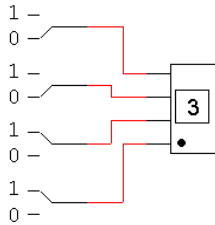


[그림 B-35] 지연시간 설정 변경

- ⑥ 입력이 A, B, C 3개인 경우 A는 10,10을, B는 20,20를 C는 40,40을 ‘Delay.Dev’ 항목에 입력하면 된다. 입력이 4개이면 마지막 입력 클록에 80, 80을 입력한다.

5- 16진 디스플레이

16진 디스플레이(Hex Display)는 2진 데이터를 16진수(HEX)로 출력하는 소자이다.



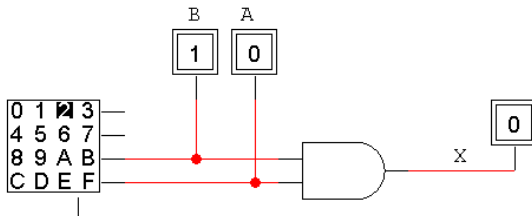
[그림 B-36] 16진 디스플레이

16진 디스플레이



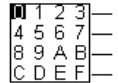
6- 16진 키보드

- ① 2진 스위치는 동시에 여러 입력을 변경하기는 어렵다. 이에 반해 16진 키보드(Hex Keyboard)는 여러 입력을 동시에 변경시키고 싶을 때 사용하는 입력소자이다.
- ② ‘Simulation IO’ 라이브러리에 있는 16진 키보드를 클릭하고 마우스로 2를 누르면 A=0, B=1이 입력된다(위쪽 출력 핀이 MSB, 아래쪽이 LSB).
- ③ 같은 스위치를 여러 번 누르면, 누를 때마다 시뮬레이션이 진행된다.



[그림 B-37] 16진 키보드

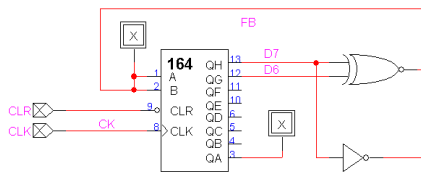
16진 키보드



1 내부회로 작성

- ① [그림 B-38]과 같은 특정 회로를 작성하여 ‘mysample.cct’라는 이름으로 저장한다.
저장 이름은 사용자 임의로 결정한다.
- ② 라이브러리를 만들 경우 외부와 인터페이스 되는 단자는 아래와 같이 ‘connector.clf’ 라이브러리 파일의 ‘Port Bidir’ 모듈을 사용하여 연결한다.
- ③ 연결하는 모듈은 마우스 오른쪽 버튼을 클릭하여 이름을 붙여 넣는다. 차후 부품 라이브러리로 사용할 때 여기서 설정한 이름을 핀 이름으로 사용하므로 적절한 이름을 기입해야 한다.

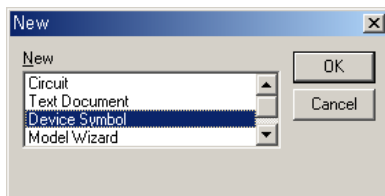
예) Port In: CLR, CLK/Port Out : OUT



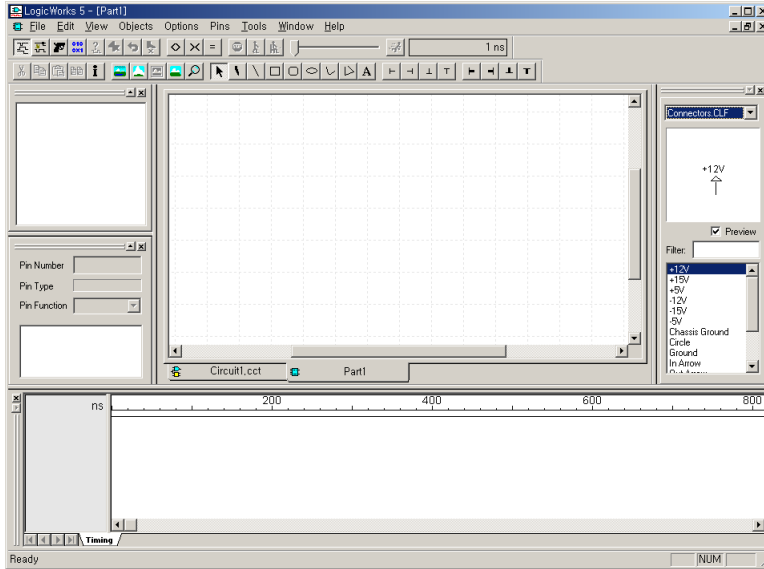
[그림 B-38] 부품 라이브러리에서 내부회로 작성 예

2 DevEditor 상태

[File]-[New] 메뉴를 클릭하고 [New] 대화상자에서 ‘Device Symbol’을 선택하면 ‘DevEditor’ 상태로 들어간다.



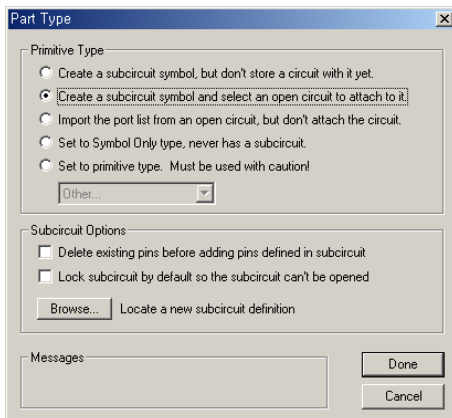
[그림 B-39] 디바이스 심볼 만들기



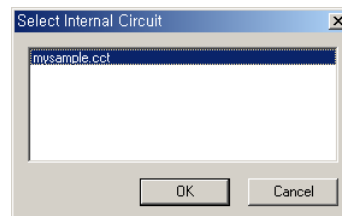
[그림 B-40] 디바이스 심볼 초기화면

3. 내부회로 삽입

- ① [Options]-[Subcircuits and Part Type...] 메뉴를 선택하여 'Create a sub-circuit symbol and select an open circuit to attach to it' 항목을 선택한다.
- ② 삽입하려는 회로(mysample.cct)를 선택하고, 부품 타입(Part Type)을 완료한다.



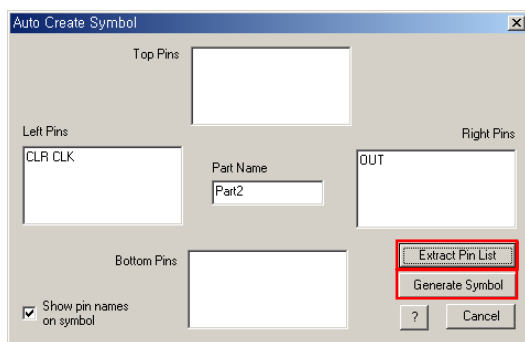
[그림 B-41] 부품 타입 생성 화면



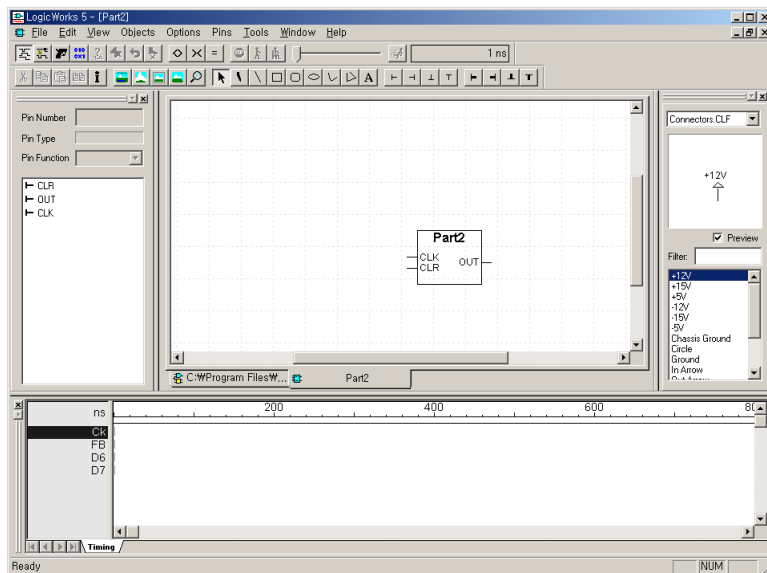
[그림 B-42] 내부회로 선택 화면

4. 핀 모듈(Pin Module)을 링크(Link)

- ① [Options] 메뉴의 [Auto Create Symbol...]과 [Extract Pin List]를 클릭하면 라이브러리 생성 시 배치되는 핀들이 나열된다. 여기서 보이는 핀들은 내부회로 작성 시 작업했던 부품 이름인 것을 알 수 있다.
- ② 핀들의 위치를 조정한 후 'Generate Symbol'을 클릭하면 [그림 B-44]와 같이 라이브러리가 생성된다.

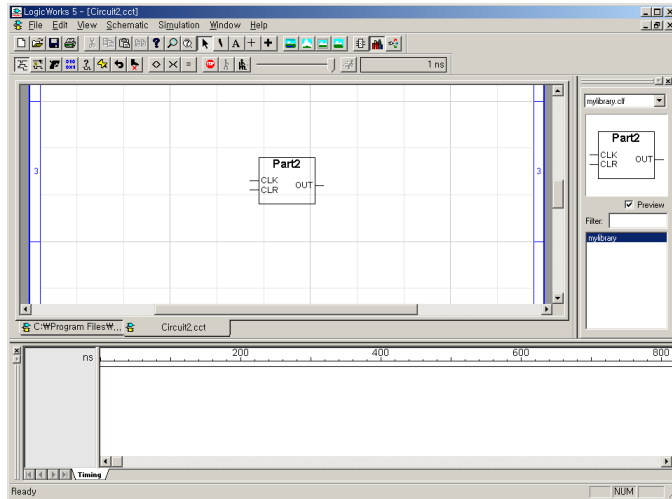


[그림 B-43] 핀 리스트 추출 화면

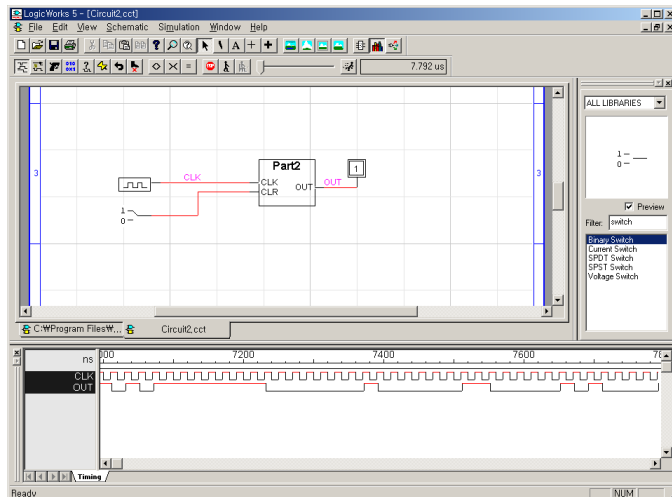


[그림 B-44] 심볼 생성이 완성된 화면

- ① 작성한 라이브러리를 [File]-[Save part as...] 메뉴를 선택하여 특정 라이브러리 파일에 저장한다. 확장자는 '*.clf'이다. 여기서는 'mylibrary.clf'에 저장한다. 다음 그림은 'mylibrary.clf'에서 생성한 모듈을 선택한 것이다.
- ② [그림 B-46]은 작성한 라이브러리를 이용하여 시뮬레이션을 수행하는 화면이다.



[그림 B-45] 저장된 라이브러리에서 불러오기



[그림 B-46] 만들어진 심볼을 이용하여 시뮬레이션 실행하기

- ① BCD 코드를 3초과(excess-3) 코드로 변환하는 회로를 라이브러리 파일을 만드는 과정을 설명한다. 먼저 3초과 코드는 10진수 각 자리의 숫자에 3을 합한 결과를 4비트의 2진수로 표시한 것이다.

[표 B-2] BCD 코드에 대한 3초과 코드의 진리표

10진수	BCD 코드	3초과 코드
	$A B C D$	$W X Y Z$
0	0 0 0 0	0 0 1 1
1	0 0 0 1	0 1 0 0
2	0 0 1 0	0 1 0 1
3	0 0 1 1	0 1 1 0
4	0 1 0 0	0 1 1 1
5	0 1 0 1	1 0 0 0
6	0 1 1 0	1 0 0 1
7	0 1 1 1	1 0 1 0
8	1 0 0 0	1 0 1 1
9	1 0 0 1	1 1 0 0

- ② 카르노 맵을 이용하여 W, X, Y, Z 를 A, B, C, D 의 함수로 간소화한다.

AB \ CD	00	01	11	10
00	0	0	0	0
01	0	1	1	1
11	×	×	×	×
10	1	1	×	×

$$W = A + BC + BD$$

AB \ CD	00	01	11	10
00	0	1	1	1
01	1	0	0	0
11	×	×	×	×
10	0	1	×	×

$$X = \overline{B}C + \overline{B}D + \overline{B}C\overline{D}$$

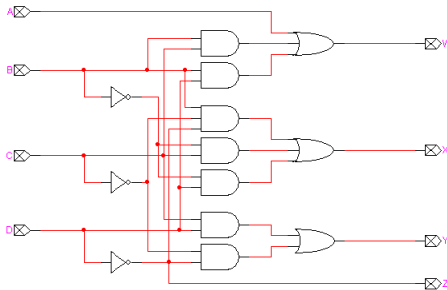
AB \ CD				
	00	01	11	10
00	1	0	1	0
01	1	0	1	0
11	×	×	×	×
10	1	0	×	×

$$Y = CD + \overline{CD}$$

AB \ CD				
	00	01	11	10
00	1	0	0	1
01	1	0	0	1
11	×	×	×	×
10	1	0	×	×

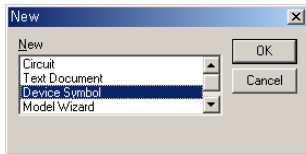
$$Z = \overline{D}$$

③ 간소화한 결과를 디자인 윈도우 상에 그리면 아래 그림과 같다.



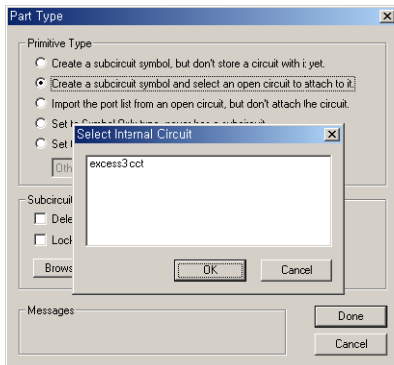
[그림 B-47] BCD 코드를 3초과 코드로 변환하는 회로

④ DevEditor 상태로 변경한다.



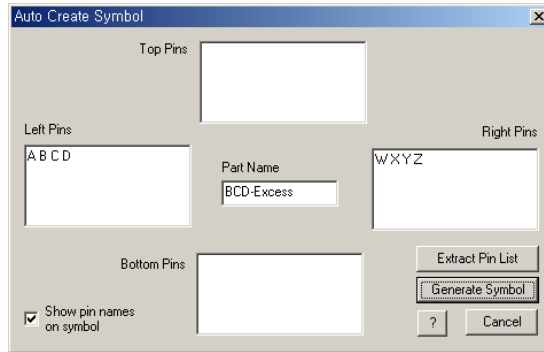
[그림 B-48] 디바이스 심볼 선택

⑤ 내부 회로를 삽입한다.



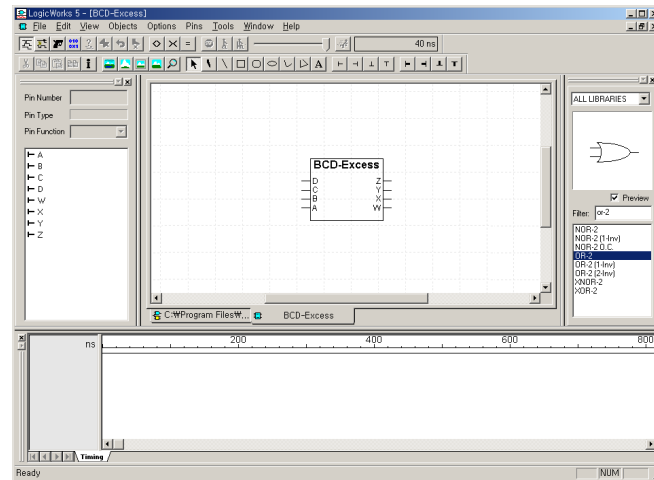
[그림 B-49] 설계한 내부회로 선택

⑥ 핀 모듈을 링크시킨다.



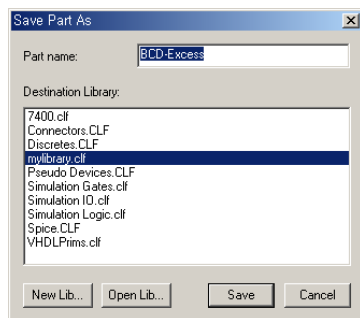
[그림 B-50] 핀 모듈 링크

⑦ 'Generate Symbol'을 선택한다.



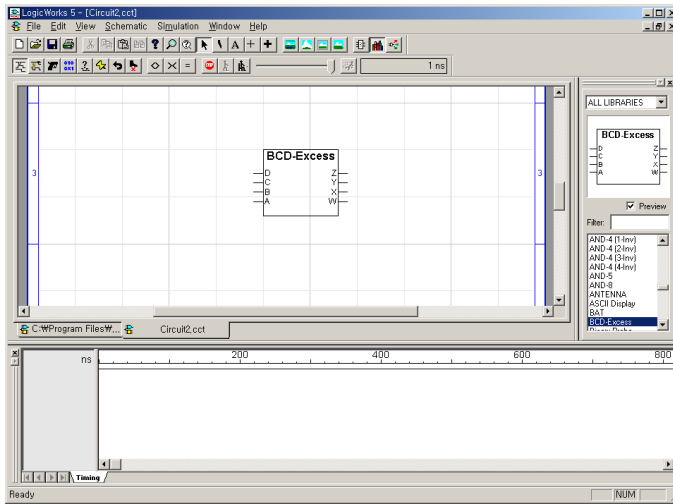
[그림 B-51] 심볼 생성

⑧ [File]—[Save As...] 메뉴를 클릭하면 아래의 화면이 표시된다. 'New Lib...'를 클릭하여 새로운 Library를 만들어 저장하거나 기존에 있는 'mylibrary.clf'에 저장해도 된다.



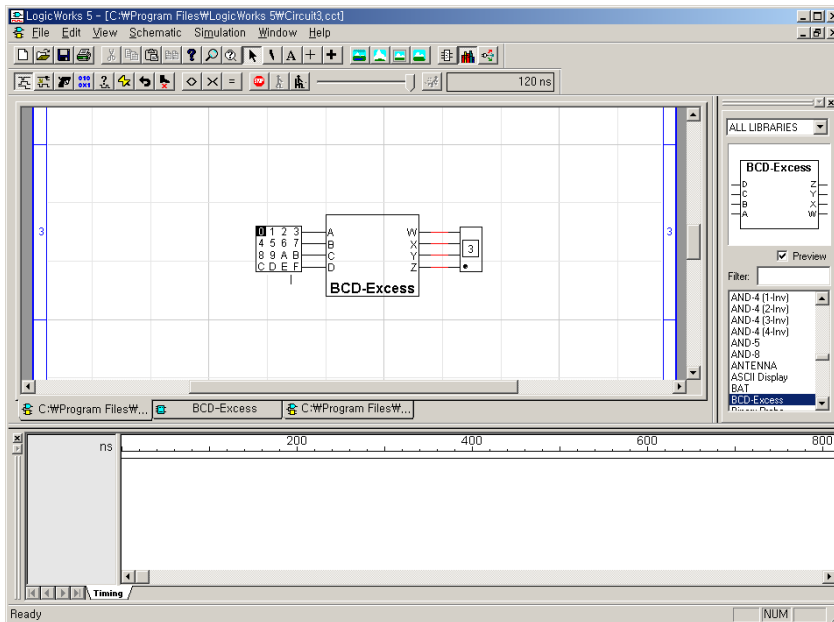
[그림 B-52] 라이브러리에 부품 저장

⑨ ‘mylibrary’에서 만든 모듈을 선택한다.



[그림 B-53] 생성된 심볼 불러오기

⑩ 회로를 만들어서 실행한다.



[그림 B-54] 생성된 심볼을 이용한 시뮬레이션 실행

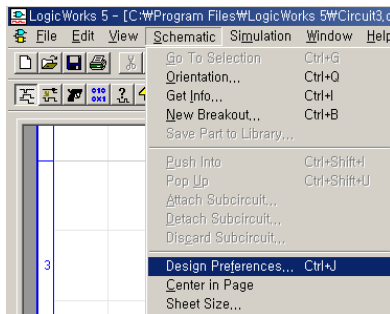
화면을 캡처하려면 키보드의 **[Alt] + [Print Screen]** 키를 같이 누르면 된다.

1 회로도 와 타이밍도의 간편한 편집(보고서 작성 시 필요)

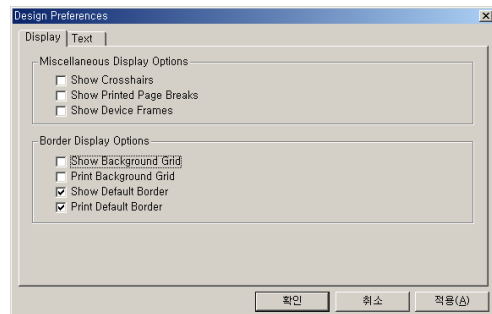
- ① 보고서 작성 시 회로도나 시뮬레이션 결과를 삽입할 필요가 있는데, 이를 위해 회로 작성과 시뮬레이션이 완료된 화면을 캡처한 후, 그림판(MS Paint)에 삽입하면 전체 화면이 그림판에 들어온다.
- ② 그림판에서 필요한 부분을 캡처한 후, 복사하여 워드프로세서에서 붙여넣기로 문서에 삽입한다.

2 그리드(Grid) 제거

- ① [Schematic]-[Design Preference(**Ctrl** + **J**)] 메뉴를 누르고 [그림 B-56]과 같이 ‘Show Background Grid’를 선택 해제한다.



[그림 B-55] 디자인 설정 메뉴

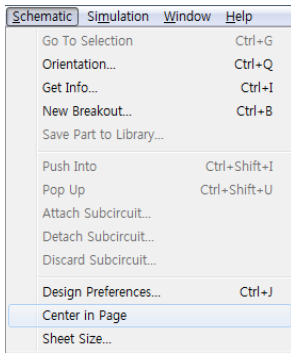


[그림 B-56] 그리드를 제거하는 윈도우

- ② <확인> 버튼을 누르면 드로잉 윈도우에 있는 그리드가 제거된다.

3- 중앙에 회로 위치

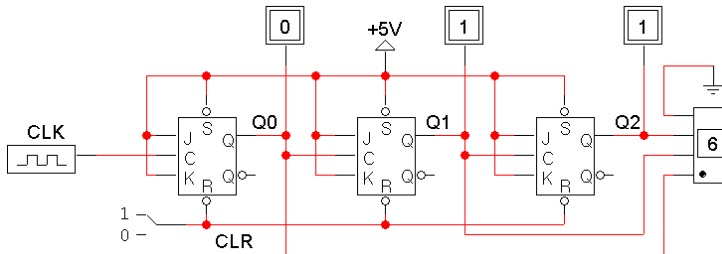
[Schematic]-[Center in Page] 메뉴를 누르면 드로잉 윈도우에 있는 회로가 중앙으로 정렬된다.



[그림 B-57] 화면 중앙에 회로 위치 선택 화면

4- 임의의 타이밍 설정 방법

- ① 그림과 같은 Clear 단자가 있는 8진 카운터를 이용하여, 임의의 타이밍 파일을 Clear 단자(CLR)에 인가하는 경우를 고려한다.

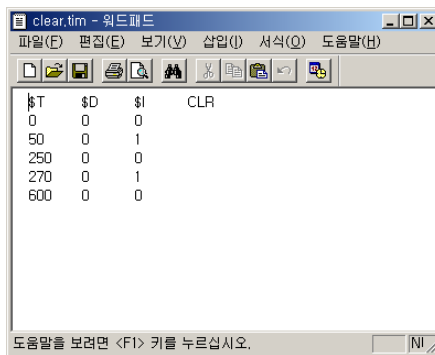


[그림 B-58] 타이밍을 실험할 회로도

- ② 워드패드(Wordpad)나 노트패드(Notepad)를 이용하여 [그림 B-59]와 같은 타이밍 파일(Timing File)을 작성하여 'clear.tim'이라는 파일명으로 저장한다. 여기서 'Clear'는 사용자 임의로 정하지만 확장자는 반드시 '*.tim'으로 해야 한다. 타이밍 파일은 \$T, \$D, \$I의 3개의 헤더(Header)와 하나 이상의 데이터 행으로 구성된다. \$I 헤더 다음에는 CLR과 같은 시뮬레이션에 사용된 신호선 명칭이 표기되어야 한다. [그림 B-59]에

제시한 내용은 0~50 시간단위(Time unit)에서는 CLR 단자에 0이 입력되고, 50~250 시간단위에서는 1이 입력된다. 250~270 시간단위에서는 다시 0이 입력되고, 270~600 시간단위에서는 1이 입력되었다가 600 시간단위 이후에서는 0이 입력된다. 각 헤더의 의미는 다음과 같다.

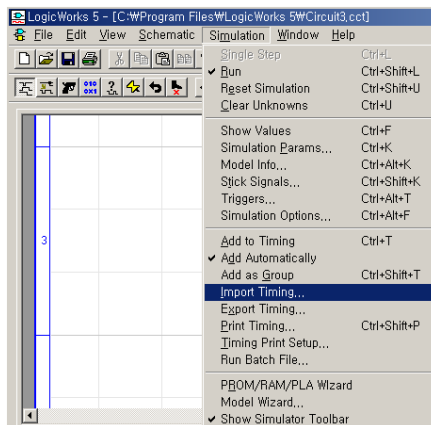
- \$T : 현재 시각을 나타낸다. 일반적으로 로직웍스에서 시간은 $T=0$ 에서부터 시작하며, 각 스텝은 ns(나노초)로 생각할 수 있다.
- \$D : 현재 행과 다음 행 사이의 지연시간을 나타낸다. 이 부분은 보통 0으로 놓는다.
- \$I : 타이밍 파일에서 구동될 각 신호값(0 또는 1)을 나타낸다.



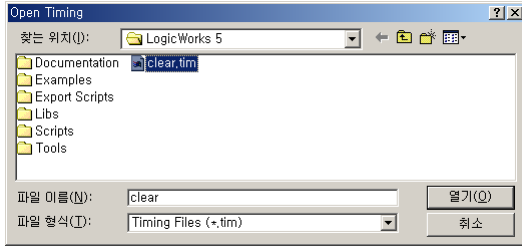
\$T	\$D	\$I	CLR
0	0	0	
50	0	1	
250	0	0	
270	0	1	
600	0	0	

[그림 B-59] 임의의 타이밍 설정

- ③ 다음에는 타이밍 파일인 ‘clear.tim’을 삽입한다. [Simulation]–[Importing Timing...] 메뉴를 클릭한다. ‘clear.tim’을 선택하고 <열기> 버튼을 클릭한다.

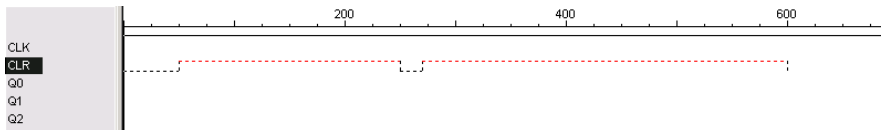


[그림 B-60] 타이밍 설정 파일 불러오기



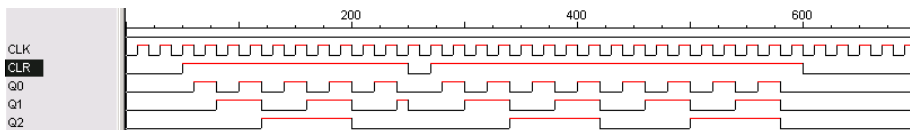
[그림 B-61] 타이밍 파일 선택

- ④ 타이밍 파일인 'clear.tim'이 삽입되면 타이밍 윈도우 창은 [그림 B-62]와 같이 CLR 단자의 타이밍이 점선으로 표시된다.




[그림 B-62] 타이밍 윈도우

- ⑤ 시뮬레이션이 수행된 결과를 [그림 B-63]에 나타내었다. 출력 결과를 분석하면, 0~50 시간단위에서는 CLR 단자가 0이므로 모든 플립플롭의 출력은 0이며, 50~250 시간단위에서는 CLR 단자가 1이므로 카운터는 정상적으로 동작한다. 250~270 시간단위에서는 다시 모든 플립플롭의 출력은 0이었다가 270~600 시간단위에서는 카운터는 정상적으로 동작한다. 마지막으로 600 시간단위 이후에서는 모든 플립플롭의 출력은 0이 된다.

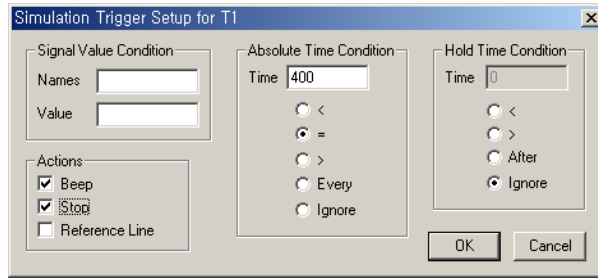


[그림 B-63] 클리어 타이밍이 적용된 타이밍도

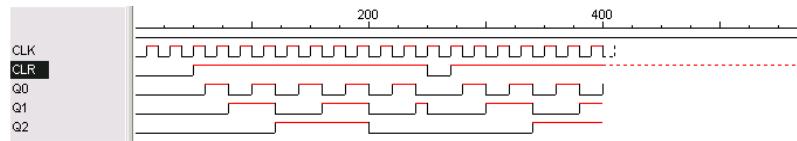
- ⑥ 시뮬레이션에서 트리거(Trigger)는 임의의 시점에서 시뮬레이션을 종료하는 기능으로  버튼을 클릭하면 된다. 400 시간단위에서 시뮬레이션을 종료하려는 경우 그림처럼 설정하고 시뮬레이션을 수행한다.



[그림 B-64] 트리거 버튼



[그림 B-65] 트리거 설정 화면



[그림 B-66] 트리거가 설정된 타이밍 원도