컴퓨터 공학 기초 실험2 보고서

실험제목: Shifter & Counter

실험일자: 2023년 10월 16일 (월)

제출일자: 2023년 10월 29일 (일)

학 과: 컴퓨터정보공학부

담당교수: 이준환 교수님

실습 분반: 월요일 0, 1, 2

학 번: 2020202031

성 명: 김재현

1. 제목 및 목적

A. 제목

Shifter & Counter

B. 목적

이번 실습에서는 flip-flop 과 combinational logic 을 이용하여 sequential logic 인 shifter 와 counter 를 설계하여 보도록 한다. register 에 저장되어 있는 정보를 단방향이나 양방향으로 이동시킬 수 있는 하드웨어를 shifter라 하고, 펄스신호에 따라 어떤 정해진 순서대로 상태의 변이가 진행되는 레지스터를 counter 라 한다.

2. 원리(배경지식)

Shifter는 데이터의 비트를 좌우로 이동시키는 역할을 합니다. Shifter는 데이터의 비트를 왼쪽으로 이동하는 "왼쪽 시프터(Left Shifter)"와 데이터의 비트를 오른쪽으로 이동하는 "오른쪽 시프터(Right Shifter)"로 구분됩니다.
shifter는 이진 데이터의 비트를 한쪽으로 이동시킴으로써 연산을 수행할 수 있습니다. 왼쪽으로 시프트하면 데이터가 2의 거듭제곱만큼 곱해지고 오른쪽으로 시프트하면 데이터가 2의 거듭제곱만큼 나누어집니다. 이러한 연산은 곱셈 및 나눗셈과 같은 연산에서 유용하게 활용됩니다.

Counter는 디지털 논리 회로에서 사용되는 장치로, 이진 또는 다진법으로 표현된 숫자를 세거나 증가시키는 데 사용됩니다. 카운터는 다양한 응용 분야에서 사용되며, 다양한 유형과 동작 방식이 있습니다. 주요 유형에는 이진 카운터, 업/다운 카운터, 동기/비동기 카운터, 모드 카운터 등이 있습니다.

Moore FSM

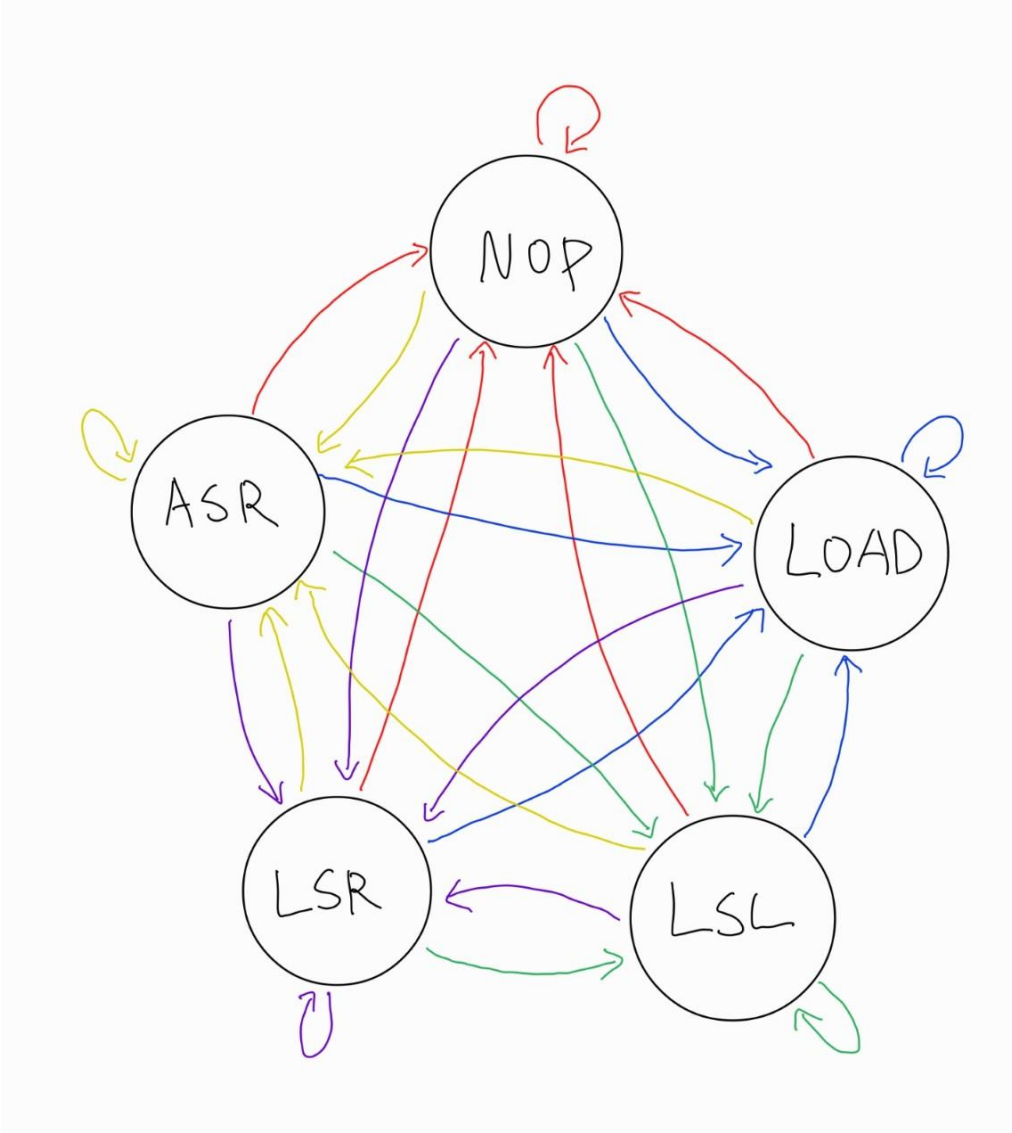
- 장점:
  1) 출력이 현재 상태에만 의존하기 때문에, 출력이 상태에 종속된 경우에 유용합니다. 즉, 출력이 일관성을 가지며 상태 변화와 무관한 경우에 유용하다고 할 수 있습니다.
  2) 설계와 디버깅이 Mealy FSM에 비해 상대적으로 간단합니다.
- 단점:
  1) 출력을 상태에만 종속하는 것은 모든 입력에 대한 반응이 같음을 의미하므로 이로 인해 시스템의 융통성이 떨어질 수 있습니다.
  2) 출력 변경이 상태 전이 이후에 발생하기 때문에, 조합 논리 회로를 추가하여 원하

는 출력을 얻을 때 지연이 발생할 수 있다.

Mealy FSM

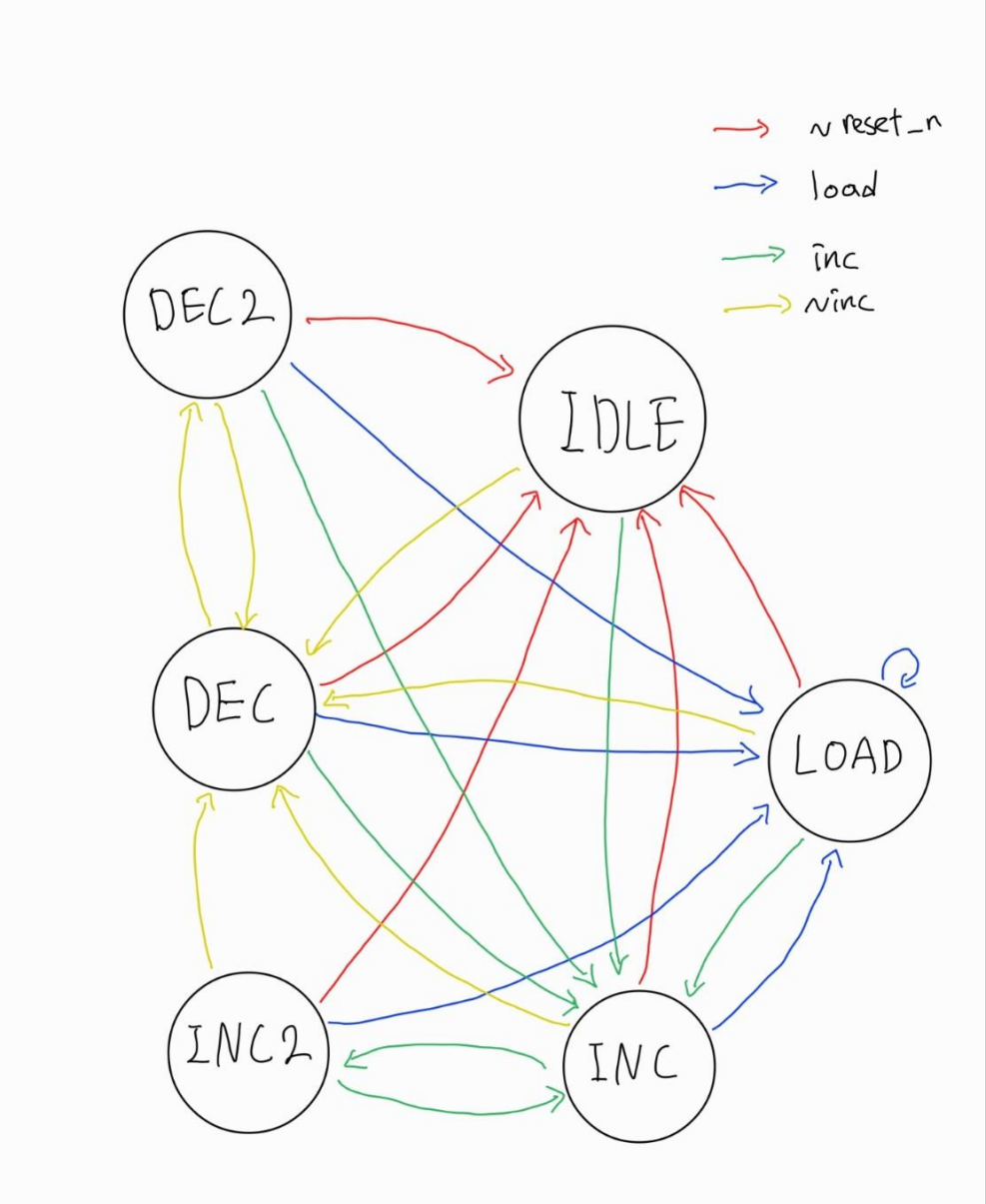
- 장점:
  1) 출력이 현재 상태 뿐만 아니라 현재 입력에도 의존하므로, 입력 신호의 변화에 더 민감하게 반응할 수 있습니다.
  2) Moore FSM에 비해 더 큰 융통성을 제공하며, 그로 인해 다양한 응용 분야에 적합합니다.
- 단점:
  1) 설계와 디버깅이 Moore FSM에 비해 복잡할 수 있다. 상태 전이와 출력은 상호 연결되어 있으므로 조심해서 처리해야 한다.

3. 설계 세부사항



shifter diagram state

각 state별 이름들을 parameter에 선언해준다. 그 선언한 이름들을 이용하여 case 문의 조건 판단에 이용한다. 그리고 nop일 경우 어떻게 진행될지 구현하고, load 일 경우에도 어떻게 진행될지 등의 대해서 코드로 구현한다. 이 구현을 cc_logic 에서 한 후 lsl, lsr, asr인지 판단하여 각 module에서 계산해서 나온 값을 대입하여 그 다음 진행을 위한 register값에 저장한다. 이 저장된 결과값을 이용하여 register에서 연산을 진행하고, 최종 결과값을 얻게 된다.

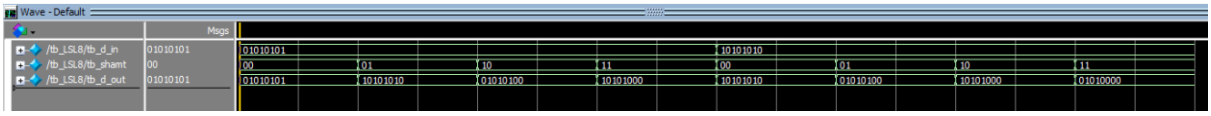


counter module을 구현하는데 state를 직접 그려보면서 그 관계를 이해하고 실제 구현 시 case문을 활용하여 모든 state간의 연결에 대해 생각하며 연결을 구현한다. 또한 parameter를 선언하여 각 state에 맞는 이름과 binary의 코딩 값을 매칭시켜줌으로써 case문에서 parameter로 선언해놓은 state의 이름을 활용하여 구현을 진행하도록 한다.

4. 설계 검증 및 실험 결과

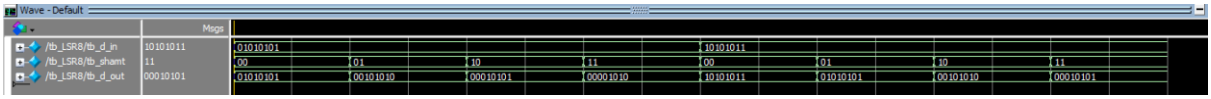
A. 시뮬레이션 결과

1) LSL8



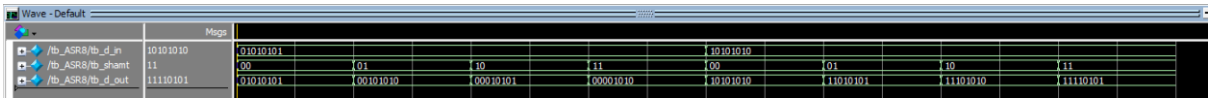
tb_LSL8의 waveform입니다. 8bit data가 shamt만큼 왼쪽으로 이동하고 그로 인해 비게 되는 자리에 0이 채워지는 것을 확인할 수 있습니다.

2) LSR8



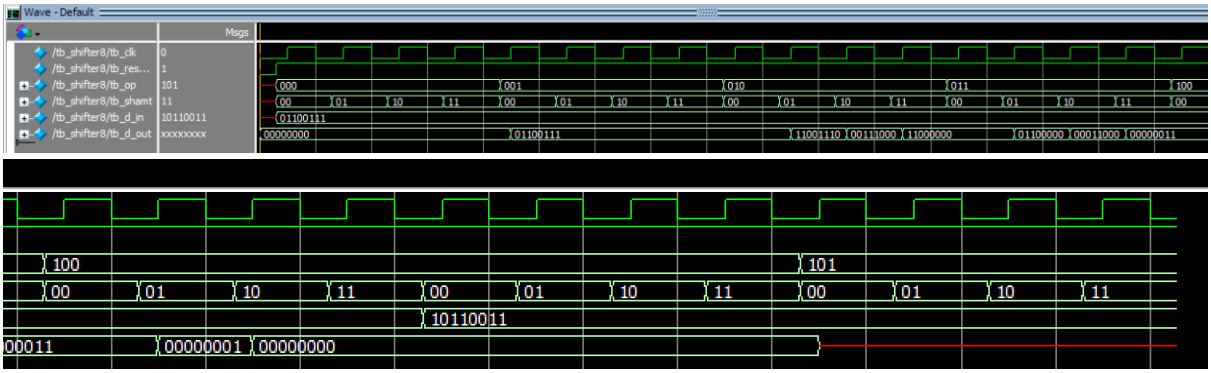
tb_LSR8의 waveform입니다. 8bit data가 shamt만큼 오른쪽으로 이동하고 그로 인해 비게 되는 자리에 0이 채워지는 것을 확인할 수 있습니다.

3) ASR8



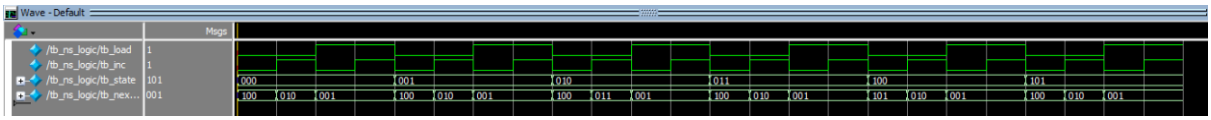
tb_ASR8의 waveform입니다. 8bit data가 shamt만큼 오른쪽으로 이동하고 그로 인해 비게 되는 자리에 data의 MSB가 채워지는 것을 확인할 수 있습니다. 이렇게 작동함으로써 data의 부호를 계속해서 유지할 수 있습니다.

4) shifter8



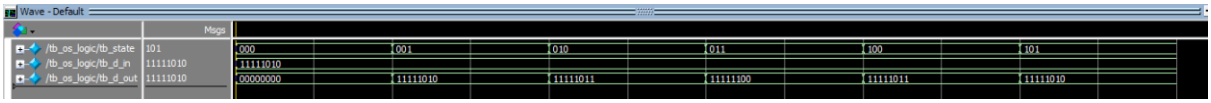
tb_shifter8의 waveform입니다. op가 000일 때는 shamt 값에 상관 없이 아무런 연산을 진행하지 않습니다. op가 001일 때는, d_in의 값을 d_out으로 출력하는 것을 확인할 수 있습니다. op가 010, 011, 100일 때는 각각 LSL, LSR, ASR 연산을 진행하며 비트의 이동 양은 shamt가 결정하는 것을 확인할 수 있습니다. 그리고 op가 100을 초과하여 존재하지 않는 state가 될 경우, x를 출력하는 것을 확인할 수 있습니다.

5) ns_logic



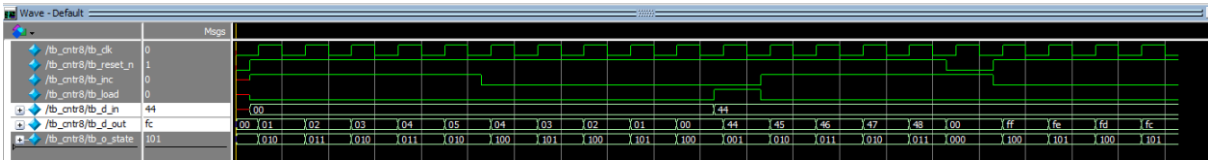
tb_ns_logic의 waveform입니다. load가 inc보다 우선순위가 높기 때문에, load가 0일 때, inc의 값에 따라 state가 inc, inc2, dec, dec2 state로 이동하게 되고, load가 1일 때는 inc의 값에 상관 없이 state가 load state로 이동하는 것을 확인할 수 있습니다.

6) os_logic



tb_os_logic의 waveform입니다. state가 IDLE 일 때는 출력이 0으로 초기화 되고, LOAD 일 때는 d_in 값을 출력하고, INC, INC2, DEC, DEC2 일 때는 출력을 +1, -1 하는 것을 확인할 수 있습니다.

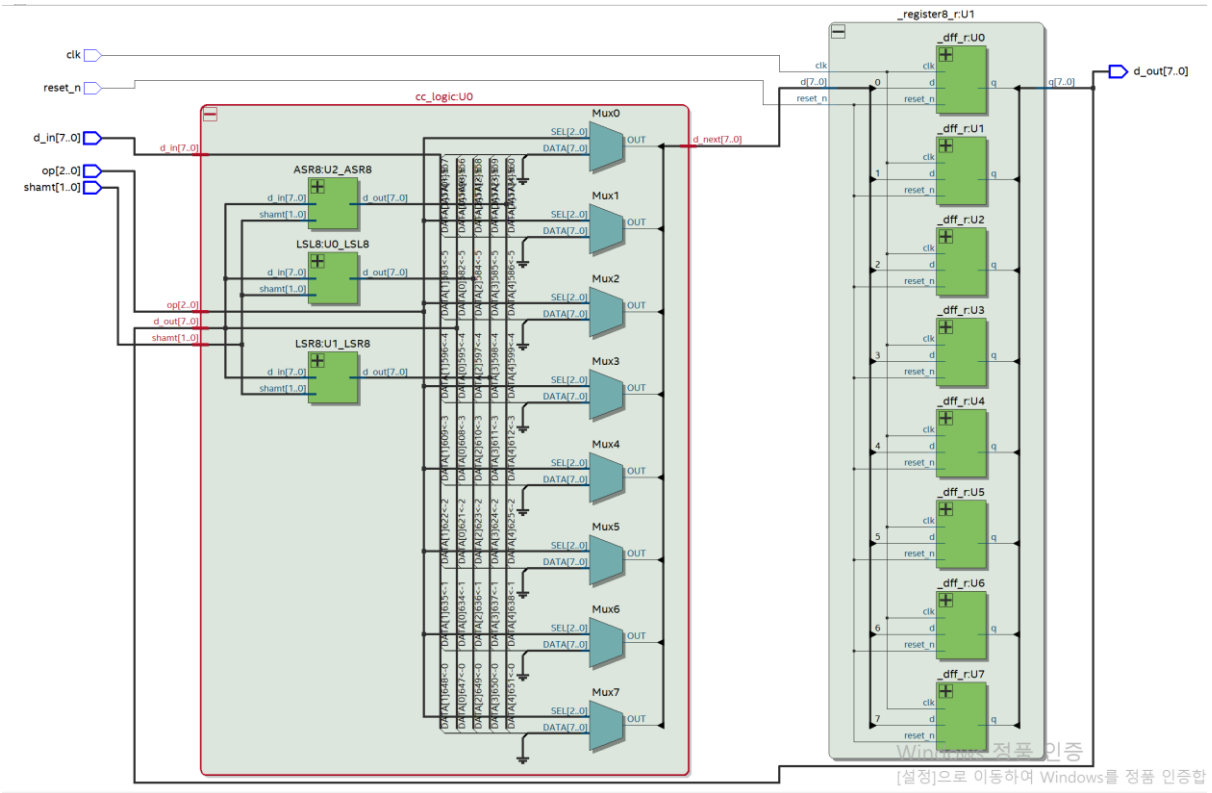
7) cntr8



reset_n이 0일 때, state가 000이 되고, reset_n이 1이면 inc값에 따라 state가 결정됩니다. inc가 0이면 state가 100이, inc가 1이면 state가 010이 됩니다. state가 100일 때 inc가 0이면, state가 101이 됩니다. state가 010일 때 inc가 1이면, state가 011이 됩니다. 매 clock의 posedge에서 현재 state에 따라 data의 값이 0으로 초기화되거나, load되거나, +1 or -1되는 것을 확인할 수 있습니다.

B. 합성(synthesis) 결과

1) shifter8



shifter8의 RTL Viewer입니다. MUX를 통해 op에 따라서 어떤 연산 결과 혹은 d_in 그대로가 d_next로 출력될지 결정됩니다. clk posedge에서 cc_logic에서 연산결과 출력된 값이 d_out으로 출력됩니다.

Flow Summary

<<Filter>>

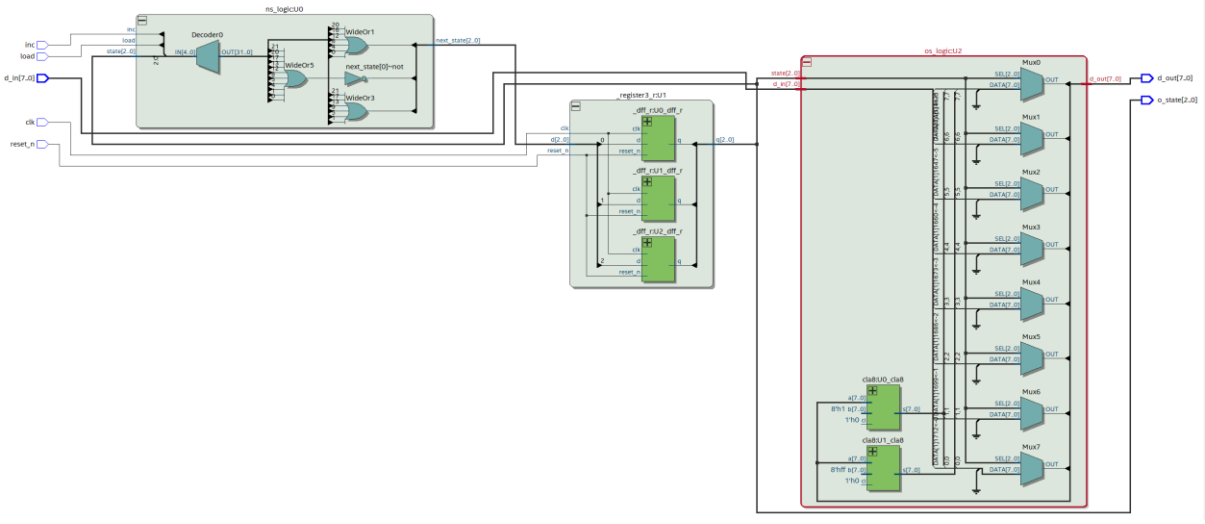
Flow Status	Successful - Sun Oct 29 02:26:34 2023
Quartus Prime Version	18.1.0 Build 625 09/12/2018 SJ Lite Edition
Revision Name	shifter8
Top-level Entity Name	shifter8
Family	Cyclone V
Device	5CSXFC6D6F31C6
Timing Models	Final
Logic utilization (in ALMs)	23 / 41,910 (< 1 %)
Total registers	8
Total pins	23 / 499 (5 %)
Total virtual pins	0
Total block memory bits	0 / 5,662,720 (0 %)
Total DSP Blocks	0 / 112 (0 %)
Total HSSI RX PCSs	0 / 9 (0 %)
Total HSSI PMA RX Deserializers	0 / 9 (0 %)
Total HSSI TX PCSs	0 / 9 (0 %)
Total HSSI PMA TX Serializers	0 / 9 (0 %)
Total PLLs	0 / 15 (0 %)
Total DLLs	0 / 4 (0 %)

shifter8의 Flow Summary입니다.

total pins는 clk 1bit + reset_n 1bit + op 3bits + shamt 2bits + d_in 8bits + d_out 8bits = 23 입니다.

Logic utilization은 23인 것을 확인할 수 있습니다.

2) cntr8



cntr8의 RTL Viewer입니다. 기본적인 틀은 Moore machine과 유사하나, d_in이 os_logic에 입력된다는 점에서 Mealy machine인 것을 알 수 있습니다. d_in이 os_logic에 입력되어야 하는 역할은 LOAD state에서 입력 값을 출력할 때, 입력 값에 대한 정보를 전달하는 것입니다. ns_logic은 inc, load, state 정보를 가지고 next_state를 결정하는 회로입니다. _register3의 역할은 clock의 posedge에서 next_state를 state로 보내주는 것입니다. 이 때 reset_n이 active low면 state가 000으로 초기화됩니다. 따라서 os_logic에 000 state가 전달되고 os_logic에서는 000 state를 IDLE로 받아들여 출력 값을 0으로 초기화합니다. os_logic은 출력 값이 cla8을 통해 연산되고, state가 변경될 때 mux을 통해 적적한 값이 선택적으로 출력됩니다.

Flow Summary

<<Filter>>

Flow Status	Successful - Sun Oct 29 03:32:24 2023
Quartus Prime Version	18.1.0 Build 625 09/12/2018 SJ Lite Edition
Revision Name	cntr8
Top-level Entity Name	cntr8
Family	Cyclone V
Device	5CSXFC6D6F31C6
Timing Models	Final
Logic utilization (in ALMs)	18 / 41,910 (< 1 %)
Total registers	3
Total pins	23 / 499 (5 %)
Total virtual pins	0
Total block memory bits	0 / 5,662,720 (0 %)
Total DSP Blocks	0 / 112 (0 %)
Total HSSI RX PCSs	0 / 9 (0 %)
Total HSSI PMA RX Deserializers	0 / 9 (0 %)
Total HSSI TX PCSs	0 / 9 (0 %)
Total HSSI PMA TX Serializers	0 / 9 (0 %)
Total PLLs	0 / 15 (0 %)
Total DLLs	0 / 4 (0 %)

cntr8의 Flow Summary입니다.

total pins는 clk 1bit + reset_n 1bit + inc 1bit + load 1bit + d_in 8bits + d_out 8bits + o_state 3bits = 23 입니다.

Logic utilization은 18임을 확인할 수 있습니다.

5. 고찰 및 결론

A. 고찰

Loadable Counter

- 장점:
    1) Loadable counter는 초기값을 설정 가능하므로 원하는 시작 값에서부터 카운트를 시작할 수 있습니다.
    2) 임의의 시작값에서 카운트를 시작할 수 있기 때문에 다양한 응용 분야에 적용하기 용이합니다.
- 단점:
    3) Loadable counter는 추가적인 논리 회로와 레지스터를 필요로 하므로 카운터가 상대적으로 복잡할 수 있습니다.

- 응용 분야: 타이머 및 카운터, 통신 프로토콜, 상태 머신 등

Ring Counter
- 장점:
    4) Ring counter는 구조가 단순하기 때문에 적은 논리 회로로 구성되어 있습니다.
    5) Ring counter는 빠른 동작을 제공하여 주기적인 패턴 생성에 적합합니다.
    6) 특정 시퀀스를 생성하기 위해 사용할 수 있으며 순환 구조를 가집니다.
- 단점:
    7) 특정 상태에서 시작하여 다른 상태로 이동하기 어렵습니다. 그러므로 다양한 시작 지점에서 카운트를 시작하는 데는 적합하지 않습니다.
- 응용 분야: shift register, 동기 신호 생성 등

Barrel shifter: 입력 데이터의 비트를 원하는 수만큼 좌나 우로 움직이는 기능을 수행하는 논리 회로입니다. 비트를 좌/우로 이동시키는 shift연산, 비트를 이동시킬 때 넘어가는 비트를 반대편으로 회전시켜 결합하는 rotation연산 등을 수행합니다.

m bits 크기의 데이터를 n bits만큼 shift 하기 위해서는 n bits 크기의 mux가 m개 필요합니다.

B. 결론

cntr8 강의자료에서 os_logic 코드 중 always sensitivity list에 state와 d_in이 포함돼 있었다. 하지만 이대로 코드를 완성시키고 테스트벤치를 돌려보니 오류가 발생했다. clock이 rising 하지 않는데도 d_in의 값이 변경되면 현재 state에 따라 연산을 한번 수행하게되는 것이다. 예를 들어 현재 state가 INC일 때, d_in이 변경되는 순간, d_out에 +1 연산이 진

행되는 것이다. 이러한 오류를 막고 clock의 posedge에서만 연산이 진행될 수 있게끔 always의 sensitivity list에서 d_in을 제외했더니 waveform이 제대로 나오는 것을 확인할 수 있었습니다.

6. 참고문헌

이준환 교수님/디지털논리회로2/광운대학교(컴퓨터정보공학부)/2023

이준환 교수님/컴퓨터공학기초실험2/광운대학교(컴퓨터정보공학부)/2023