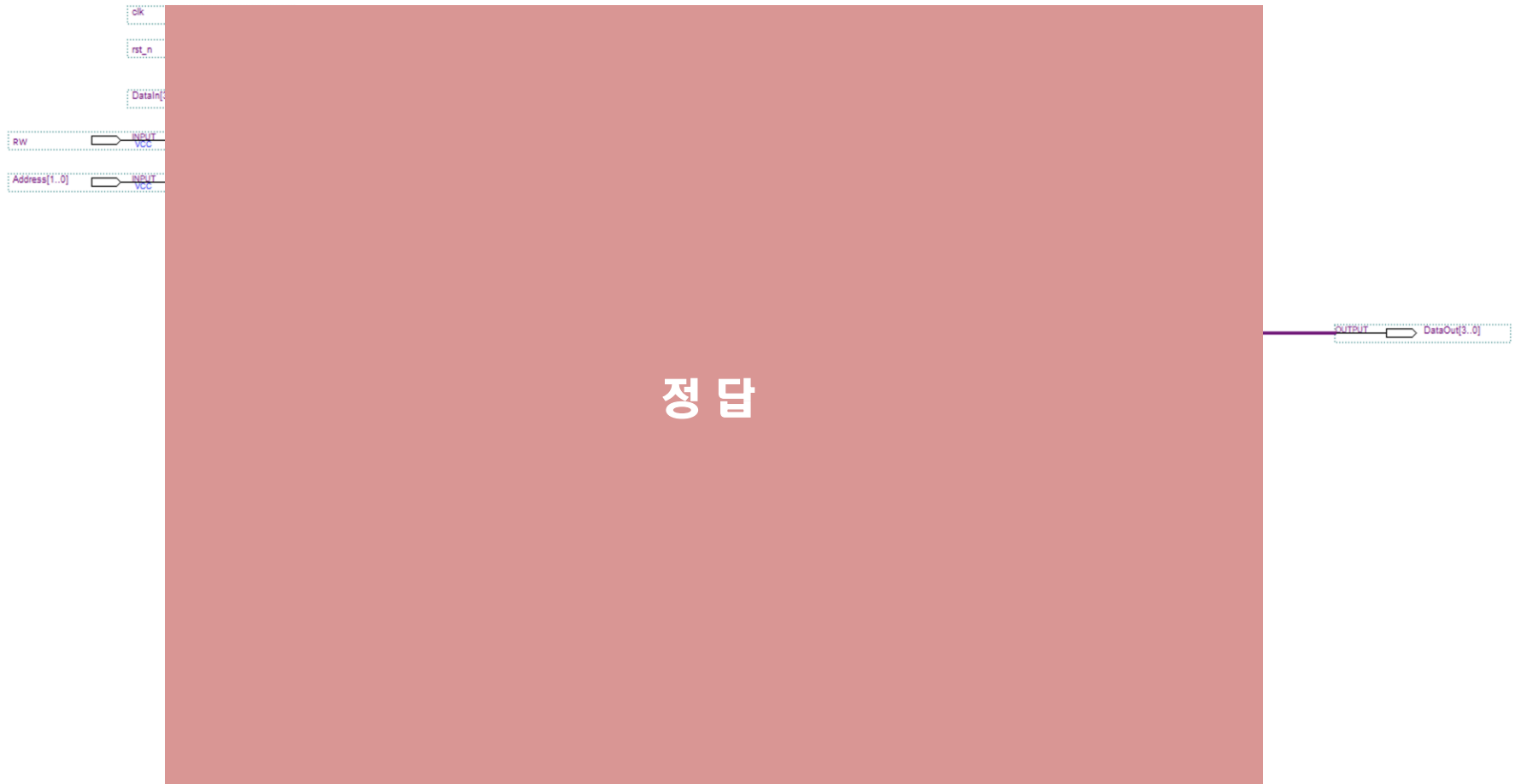


실습 1

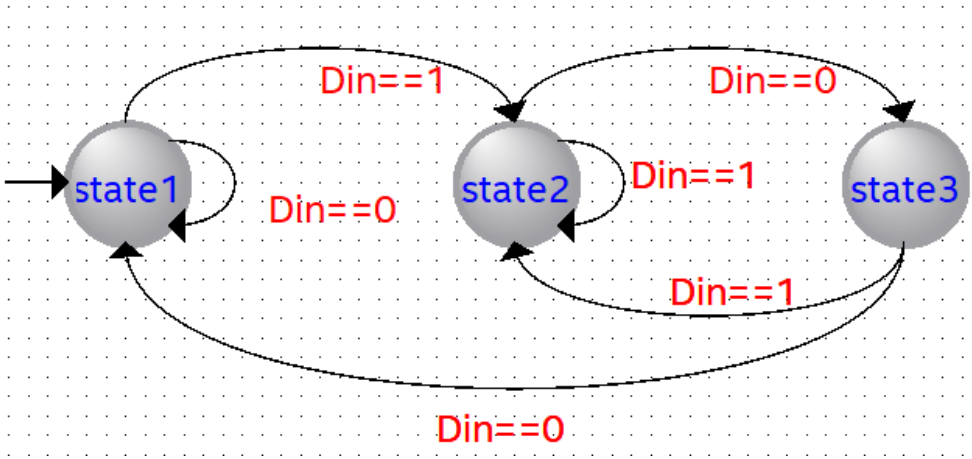


실습 2



과제 1

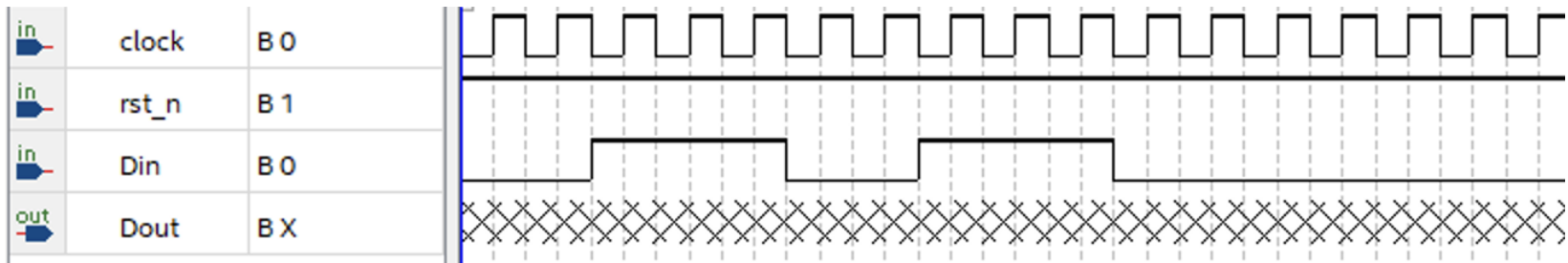
아래 FSM에 대한 입력이 다음과 같다
출력값 Dout을 나타내시오



State Machine Wizard

Inputs	Outputs	States	Transitions	Actions
Output Port	Output Value	In State	Additional Conditions	
Dout	0	state1		
Dout	0	state2		
Dout	1	state3		
< New >				

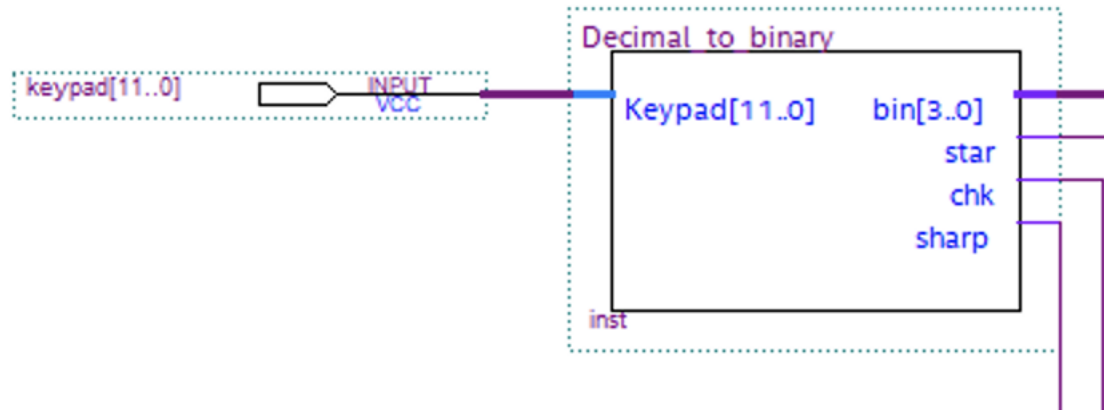
OK Close Apply Help



키패드로 SRAM에 값을 저장하고 읽는 Control Unit을 만들려고 한다
부분 모듈의 동작과 전체 모듈의 동작을 읽고 이해한 후,
그와 같은 동작을 하도록 Control Unit의 빈칸에 알맞은 값을 넣으시오

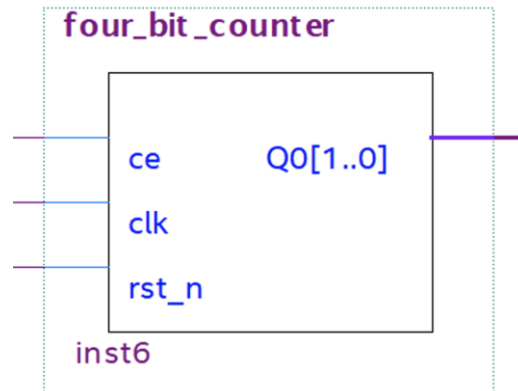
부분 모듈 BCD는 일반적인 키패드의 0~9, *, #을 입력으로 가진다
[제공 파일을 심볼화해서 사용] - Decimal_to_binary.bdf

- 1) 숫자를 누르면 해당하는 숫자의 2진수값이 Bin으로 나온다
- 2) 아무 숫자나 누르면 chk에 1이 나온다
- 3) *을 누르면 star에 1이 나온다
- 4) #을 누르면 sharp에 1이 나온다
- 5) *, #을 누를 땐 chk에는 아무 값도 안 나온다
- 6) 입력은 12비트(11번째 비트는 *, 12번째 비트는 #)



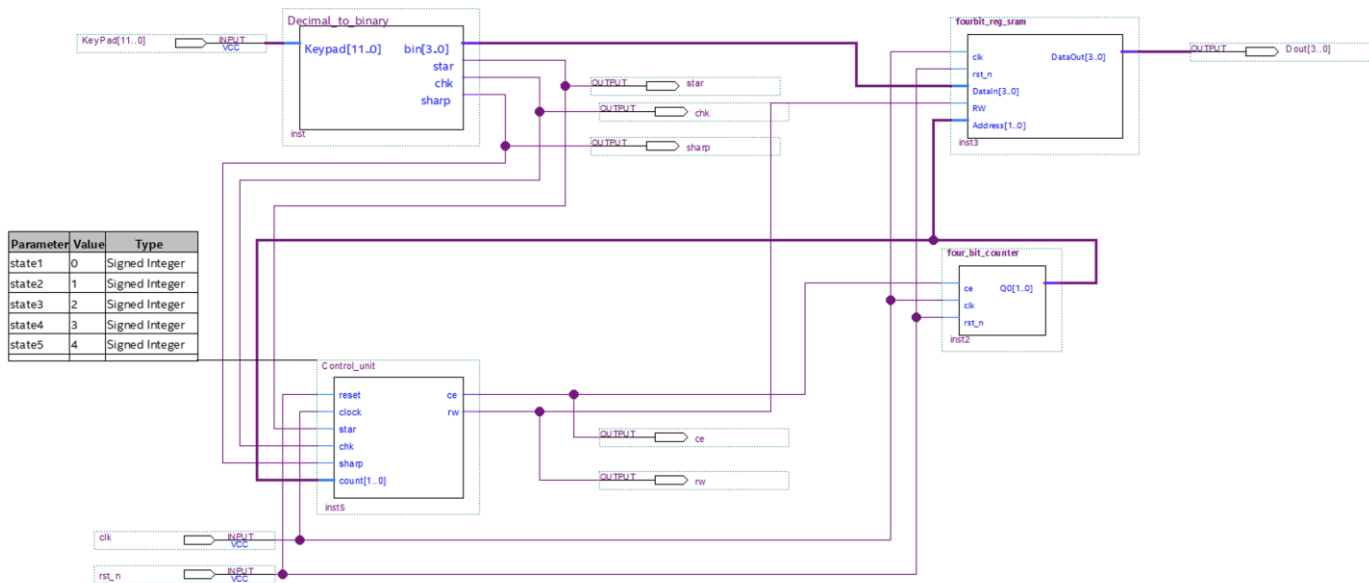
부분 모듈 Counter4_ce는 4진 카운터의 일종이다 [제공 파일을 심볼화해서 사용] – four_bit_couter.bdf

- 1) 처음 출력값은 0 이다
- 2) Ce에 1이 1 클럭 들어가면 출력값이 1 증가한다
- 3) Ce에 1이 들어갈 때 마다 출력값이 증가하여 0, 1, 2, 3, 0, 1... 을 반복한다
- 4) Ce에 0이 들어가면 출력값은 유지된다
- 5) 출력은 2비트(00~ 11) → **SRAM의 주소로 사용**



전체 모듈은 아래와 같은 데이터패스와 동작을 가진다

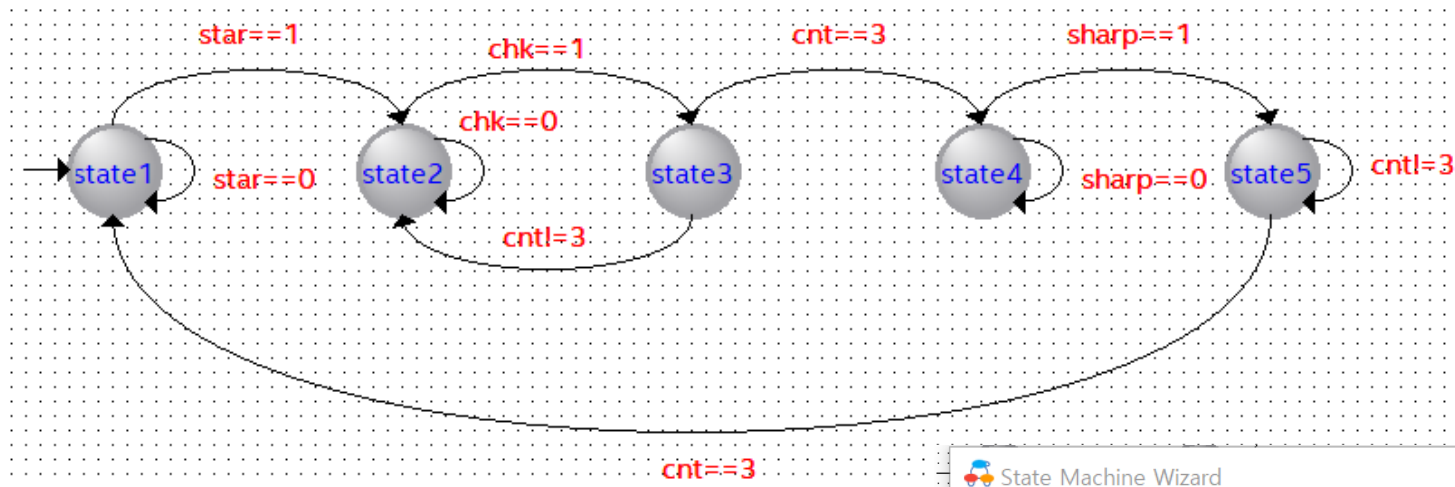
- 1) 처음엔 키패드의 아무 숫자를 눌러도 반응하지 않는다
- 2) *을 한 번 누르고 난 후,
- 3) 숫자를 네 번 누르면 차례대로 SRAM의 0, 1, 2, 3번지에 저장된다
- 4) 이후 #을 누르면 SRAM에서 0, 1, 2, 3번지의 값이 반복해서 읽어진다



전체 모듈의 데이터패스

앞서 전체 모듈의 동작이 가능하도록 Control Unit을 완성하시오

- 1) 처음엔 키패드의 아무 숫자를 눌러도 반응하지 않는다
- 2) *을 한 번 누르고 난 후,
- 3) 숫자를 네 번 누르면 차례대로 SRAM의 0, 1, 2, 3번지에 저장된다
- 4) 이후 #을 누르면 SRAM에서 0, 1, 2, 3번지의 값이 읽어진다



State Machine Wizard

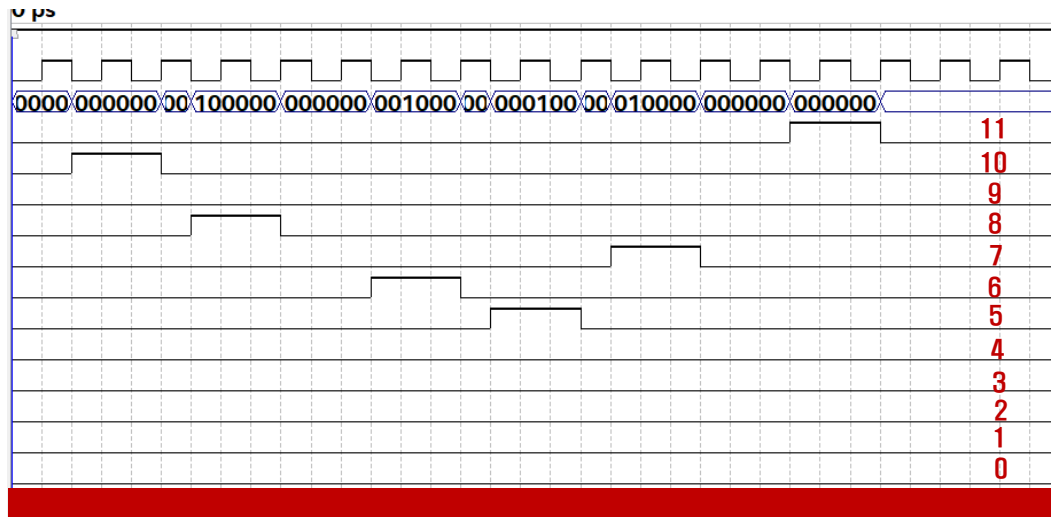
Inputs	Outputs	States	Transitions	Actions
Output Port	Output Value	In State	Additional Conditions	
ce	0	state1		
rw	0	state1		
ce	0	state2		
rw	0	state2		

State 3~ 5 까지의 Action 유추하여 설계

과제 2(시뮬레이션 입력 및 출력)

앞서 전체 모듈의 동작이 가능하도록 Control Unit을 완성하시오

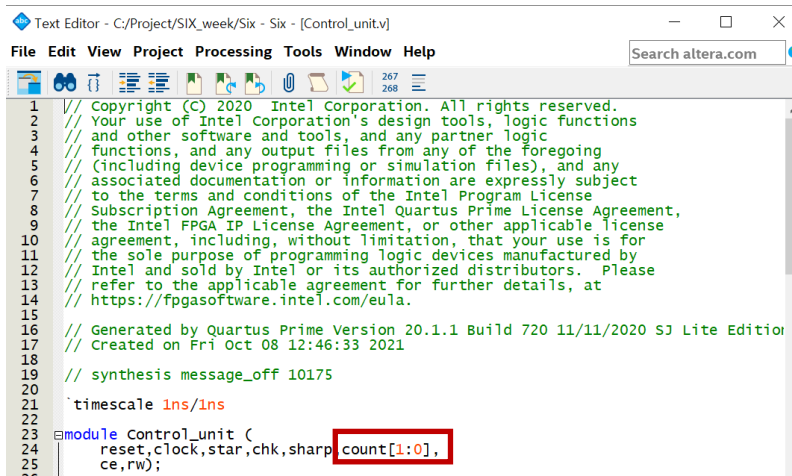
- 1) 처음엔 키패드의 아무 숫자를 눌러도 반응하지 않는다
- 2) *을 한 번 누르고 난 후,
- 3) 숫자를 네 번 누르면 차례대로 SRAM의 0, 1, 2, 3번지에 저장된다
- 4) 이후 #을 누르면 SRAM에서 0, 1, 2, 3번지의 값이 읽혀진다



SRAM의 출력값

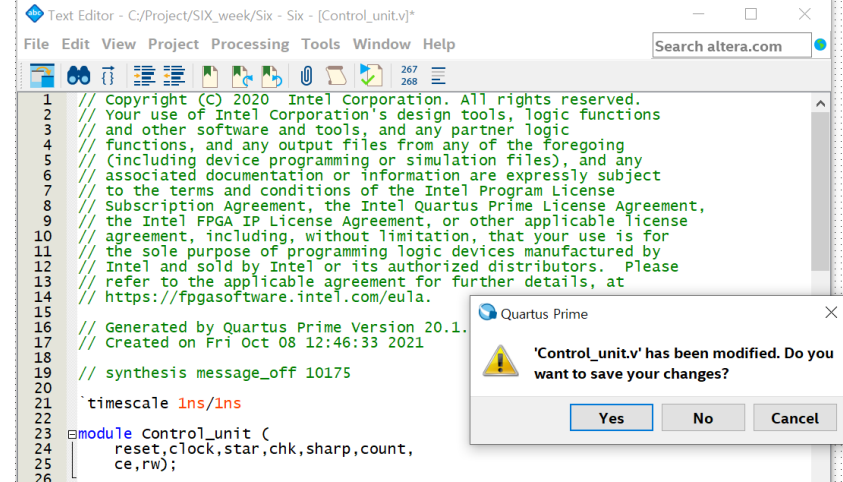
***SRAM은 평소에 00번지의 값을 계속 읽는 것을 염두해 둘 것.**

과제 2의 State machine 같은 경우 멀티비트를 입력으로 받음 **Verilog 파일에서 심볼을 만들 경우 에러가 발생** 해결법 : Verilog 파일의 입력 부분의 멀티비트 부분 삭제 후 심볼 생성



```
1 // Copyright (C) 2020 Intel Corporation. All rights reserved.
2 // Your use of Intel Corporation's design tools, logic functions
3 // and other software and tools, and any partner logic
4 // functions, and any output files from any of the foregoing
5 // (including device programming or simulation files), and any
6 // associated documentation or information are expressly subject
7 // to the terms and conditions of the Intel Program License
8 // Subscription Agreement, the Intel Quartus Prime License Agreement,
9 // the Intel FPGA IP License Agreement, or other applicable license
10 // agreement, including, without limitation, that your use is for
11 // the sole purpose of programming logic devices manufactured by
12 // Intel and sold by Intel or its authorized distributors. Please
13 // refer to the applicable agreement for further details, at
14 // https://fpgasoftware.intel.com/eula.
15
16 // Generated by Quartus Prime Version 20.1.1 Build 720 11/11/2020 SJ Lite Edition
17 // Created on Fri Oct 08 12:46:33 2021
18
19 // synthesis message_off 10175
20
21 timescale 1ns/1ns
22
23 module Control_unit (
24     reset, clock, star, chk, sharp, count[1:0],
25     ce, rw);
26
```

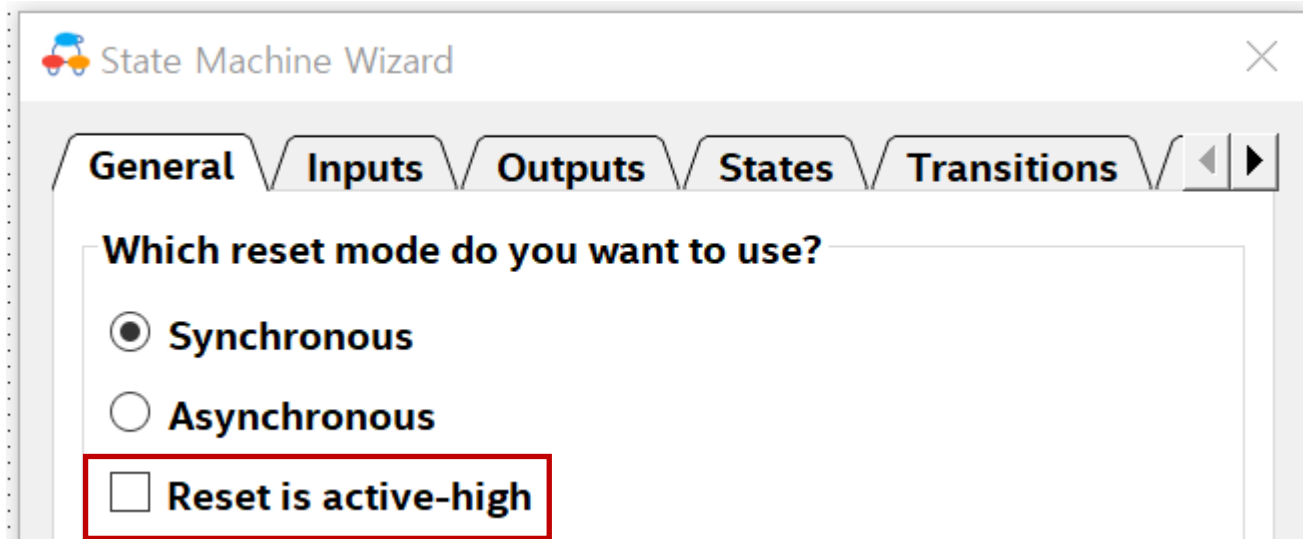
과제 2의 State machine의 멀티비트 입력



```
1 // Copyright (C) 2020 Intel Corporation. All rights reserved.
2 // Your use of Intel Corporation's design tools, logic functions
3 // and other software and tools, and any partner logic
4 // functions, and any output files from any of the foregoing
5 // (including device programming or simulation files), and any
6 // associated documentation or information are expressly subject
7 // to the terms and conditions of the Intel Program License
8 // Subscription Agreement, the Intel Quartus Prime License Agreement,
9 // the Intel FPGA IP License Agreement, or other applicable license
10 // agreement, including, without limitation, that your use is for
11 // the sole purpose of programming logic devices manufactured by
12 // Intel and sold by Intel or its authorized distributors. Please
13 // refer to the applicable agreement for further details, at
14 // https://fpgasoftware.intel.com/eula.
15
16 // Generated by Quartus Prime Version 20.1.
17 // Created on Fri Oct 08 12:46:33 2021
18
19 // synthesis message_off 10175
20
21 timescale 1ns/1ns
22
23 module Control_unit (
24     reset, clock, star, chk, sharp, count,
25     ce, rw);
26
```

멀티비트 명시하는 부분 삭제 후 심볼 생성 진행

과제 2에서 사용되는 4bit SRAM, Counter 등은 Negative clock을 사용함
Control unit의 클럭도 네거티브로 설정



General 탭에서 해당 항목 체크 해제