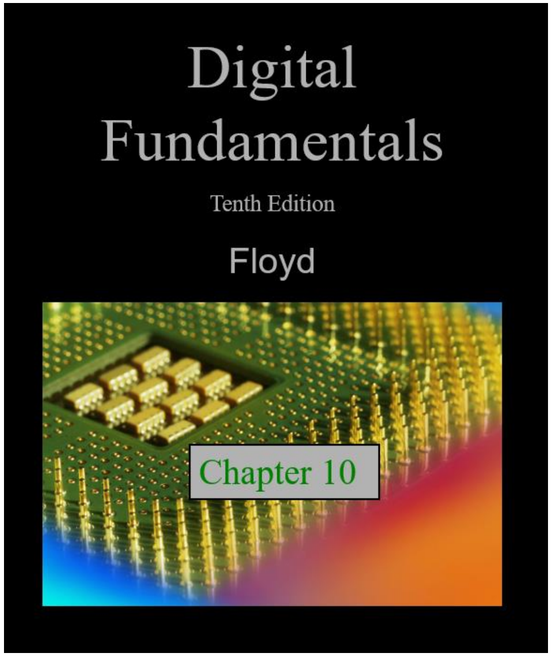
최신디지털공학

Digital Fundamentals

Thomas L. Floyd



Digital Fundamentals

Tenth Edition

Floyd

Chapter 10

Chapter 10

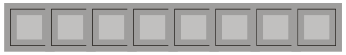
메모리

Memory

Summary

Memory Units

Memories store data in units from one to eight bits. The most common unit is the **byte**, which by definition is 8 bits.



Computer memories are organized into multiples of bytes called words. Generally, a word is defined as the number of bits handled as one entity by a computer. By this definition, a word is equal to the internal register size (usually 16, 32, or 64 bits).

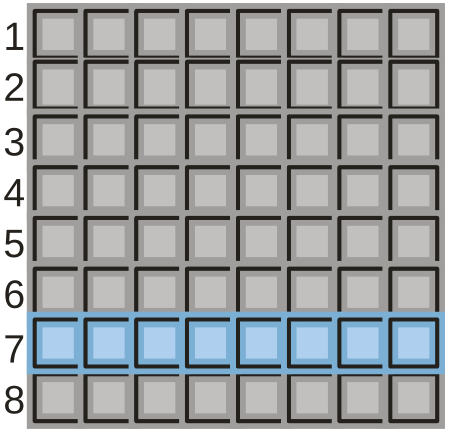
For historical reasons, assembly language defines a word as exactly two bytes. In assembly language, a 32 bit entity is called a double-word and 64 bits is defined as a quad-word.

Summary

Memory Units

The location of a unit of data in a memory is called the **address**. In PCs, a byte is the smallest unit of data that can be accessed.

In a 2-dimensional array, a byte is accessed by supplying a row number. For example the blue byte is located in row 7.

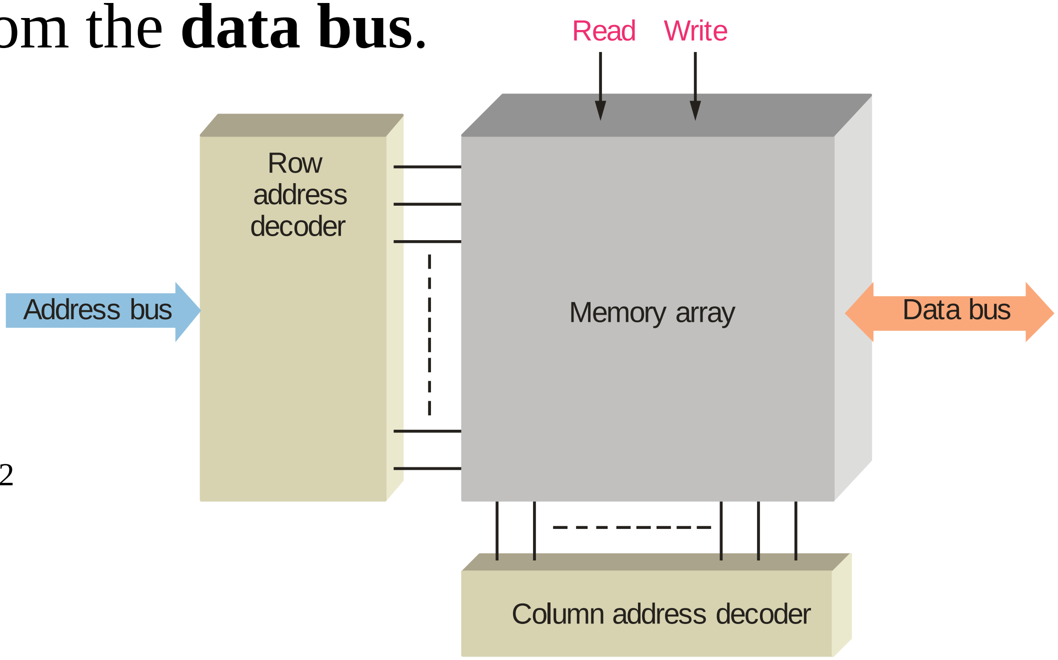


Summary

Memory Addressing

In order to read or write to a specific memory location, a binary code is placed on the **address bus**. Internal decoders decode the address to determine the specific location. Data is then moved to or from the **data bus**.

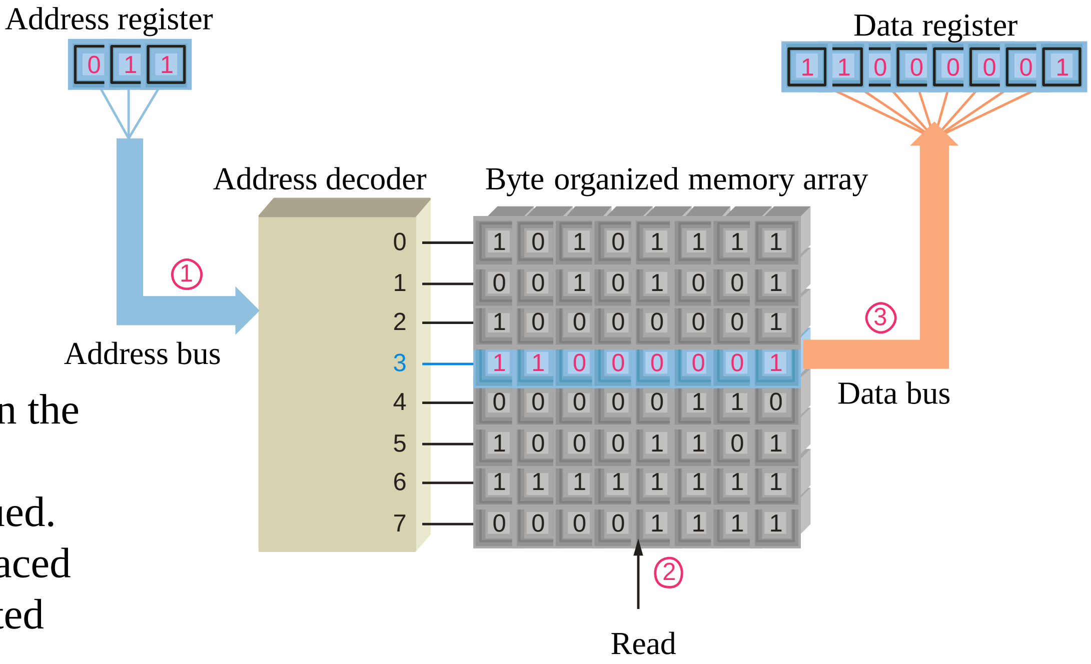
The address bus is a group of conductors with a common function. Its size determines the number of locations that can be accessed. A 32 bit address bus can access 2^{32} locations, which is approximately 4G.



Summary

Read and Write Operations

The read operation is actually a “copy” operation, as the original data is not changed. The data bus is a “two-way” path; data moves *from* the memory during a read operation.



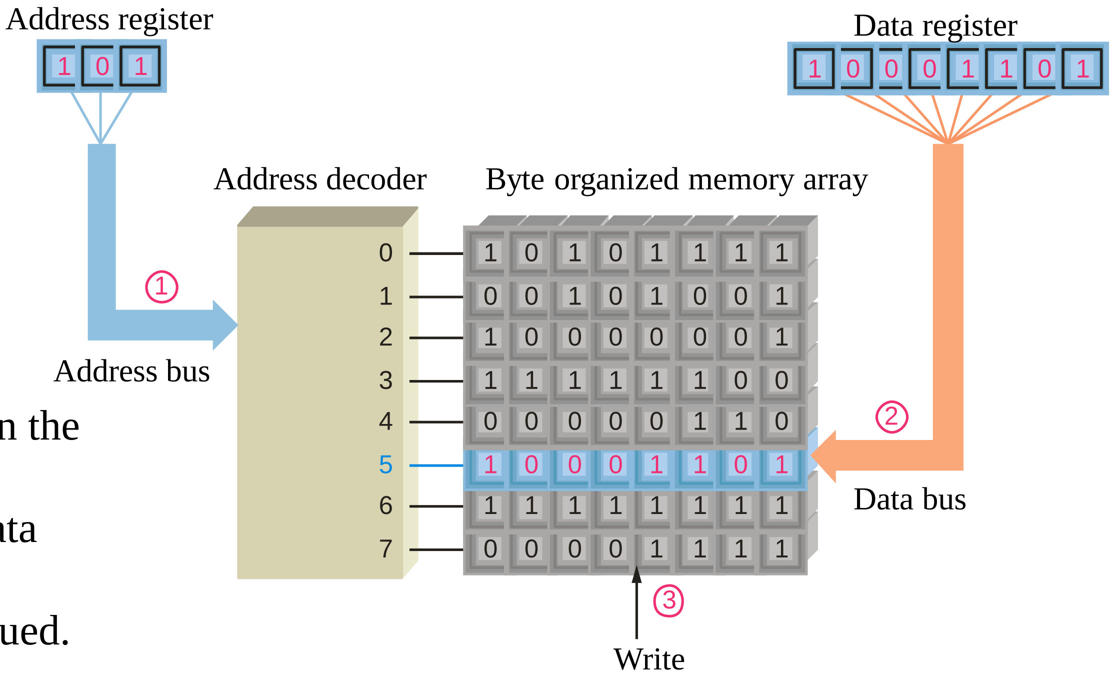
0	1	0	1	0	1	1	1	1
1	0	0	1	0	1	0	0	1
2	1	0	0	0	0	0	0	1
3	1	1	0	0	0	0	0	1
4	0	0	0	0	0	1	1	0
5	1	0	0	0	1	1	0	1
6	1	1	1	1	1	1	1	1
7	0	0	0	0	1	1	1	1

1. The address is placed on the address bus.
2. A read command is issued.
3. A copy of the data is placed in the data bus and shifted into the data register.

Summary

Read and Write Operations

The two main memory operations are called **read** and **write**. A simplified write operation is shown in which new data overwrites the original data. Data moves *to* the memory.

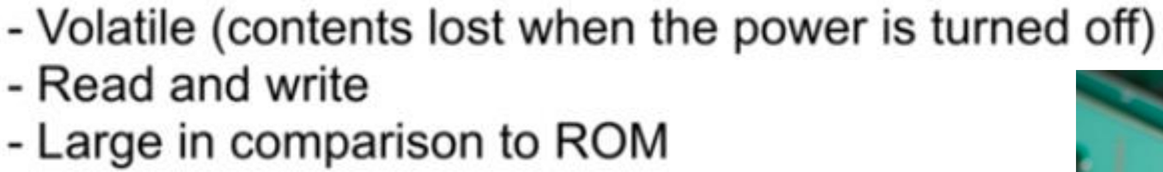


0	1	0	1	0	1	1	1	1
1	0	0	1	0	1	0	0	1
2	1	0	0	0	0	0	0	1
3	1	1	1	1	1	1	0	0
4	0	0	0	0	0	1	1	0
5	1	0	0	0	1	1	0	1
6	1	1	1	1	1	1	1	1
7	0	0	0	0	1	1	1	1

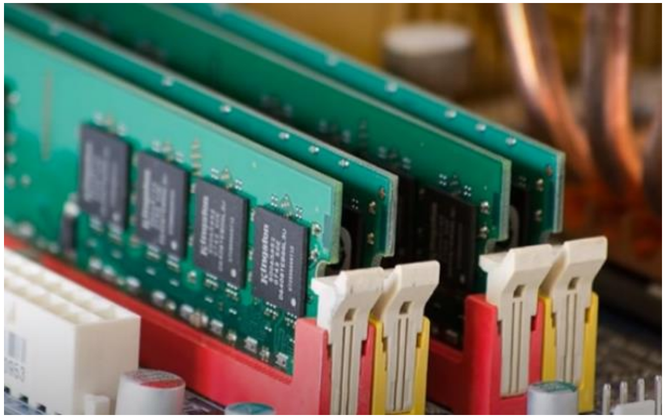
1. The address is placed on the address bus.
2. Data is placed on the data bus.
3. A write command is issued.

Summary

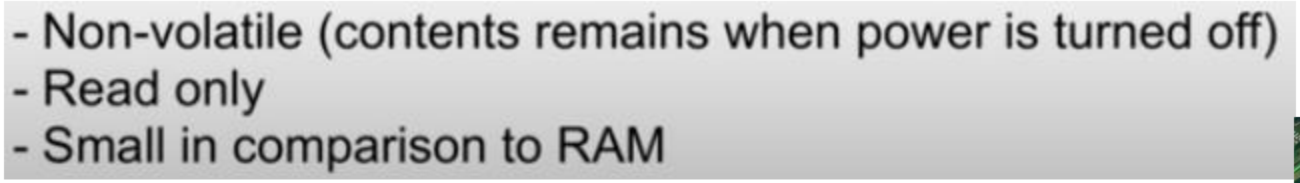
RAM(Random Access Memory)



- Volatile (contents lost when the power is turned off)
- Read and write
- Large in comparison to ROM



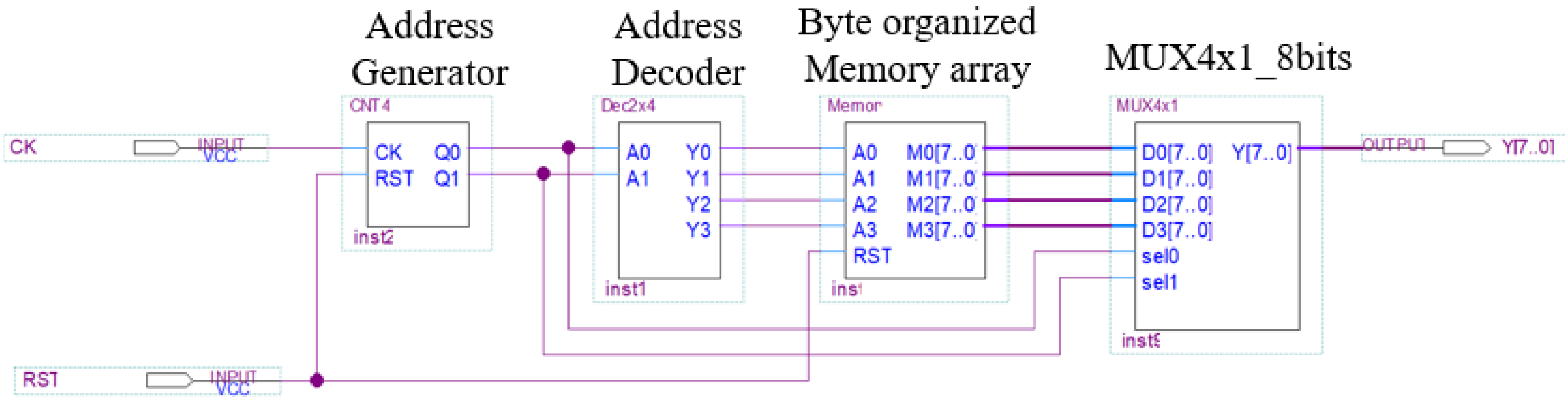
ROM(Read Only Memory)



- Non-volatile (contents remains when power is turned off)
- Read only
- Small in comparison to RAM

H/W #1

아래의 Memory controller 회로를 설계하고
그 내용을 설명하시오.

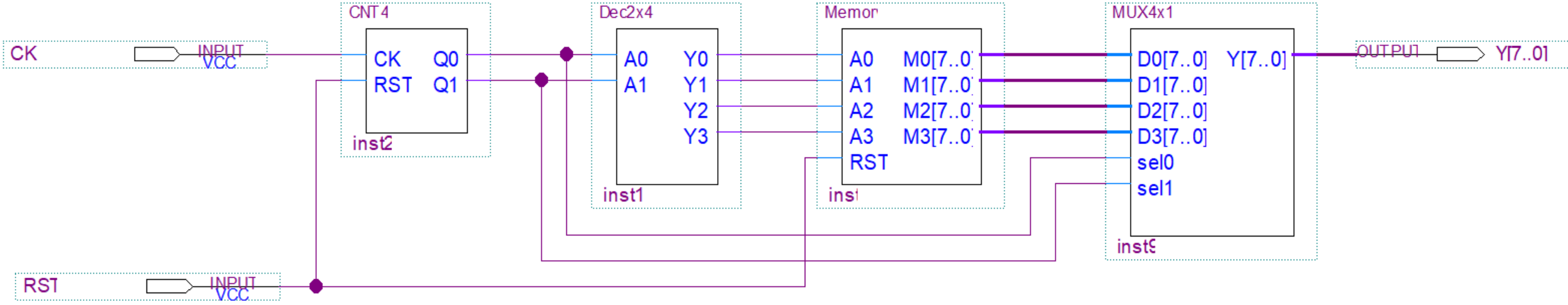


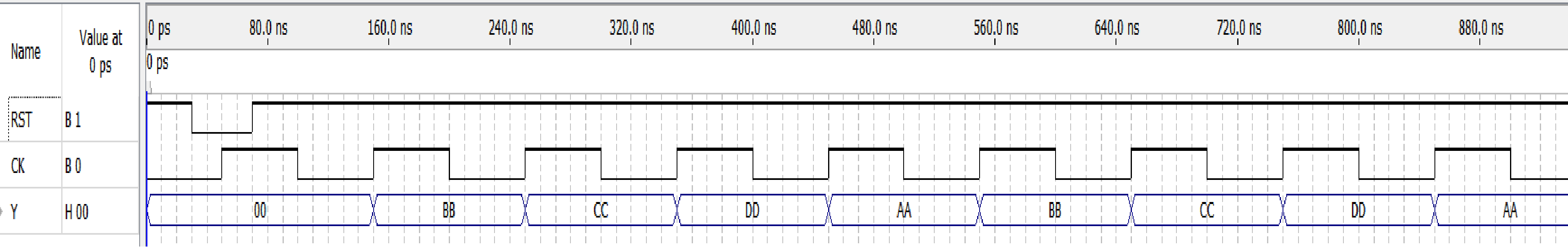
Address Generator

Address Decoder

Byte organized Memory array

MUX4x1_8bits





Memory 출력 data

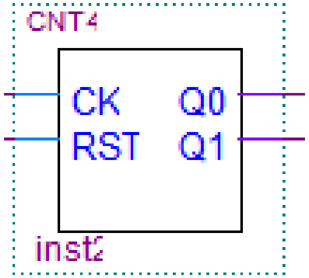
Address 0 : AA

Address 1 : BB

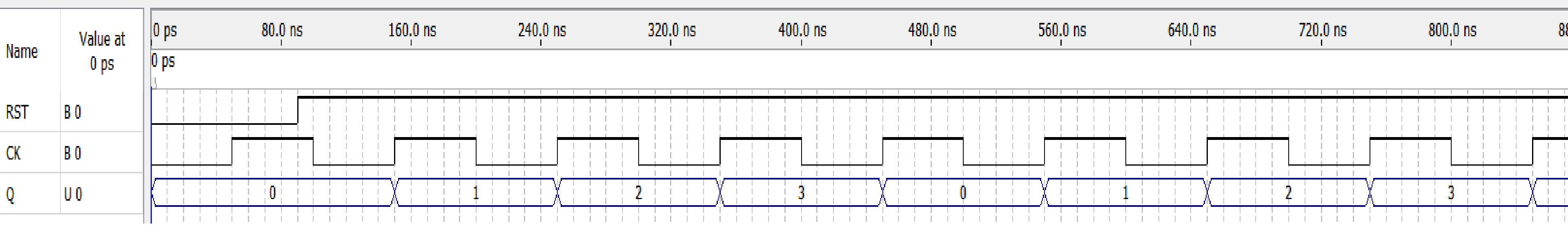
Address 2 : CC

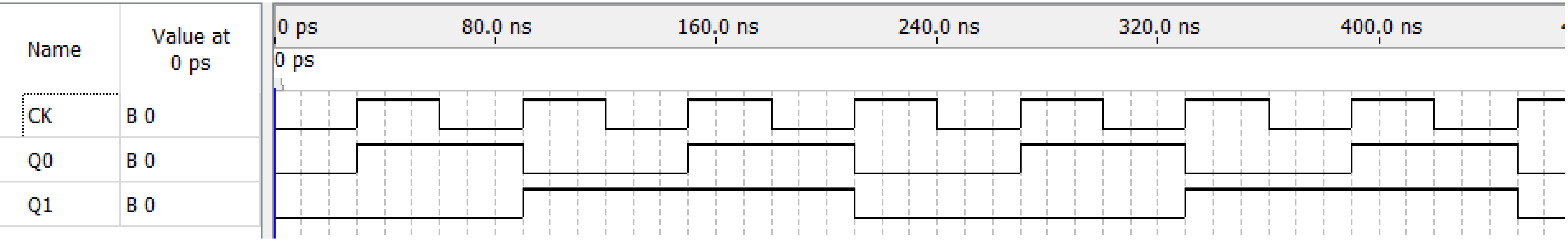
Address 3 : DD

Address Generator

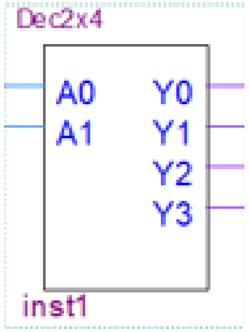


4진 Counter

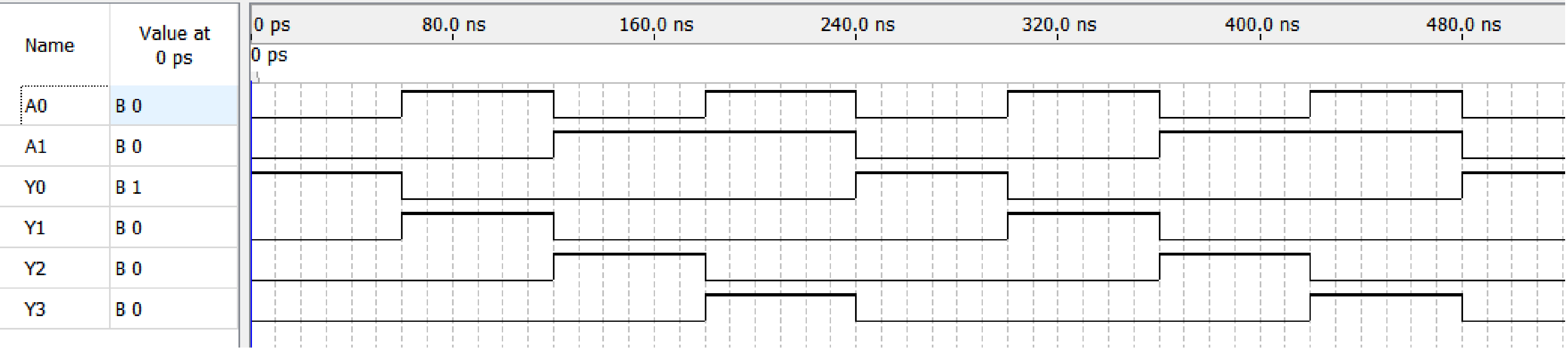




Address Decoder



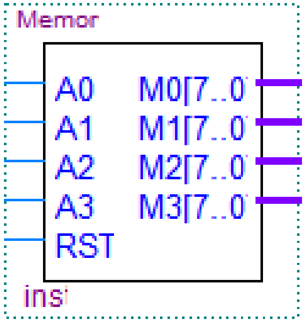
2X4 Decoder



Name	Value at 0 ps
A0	B 0
A1	B 0
Y0	B 1
Y1	B 0
Y2	B 0
Y3	B 0

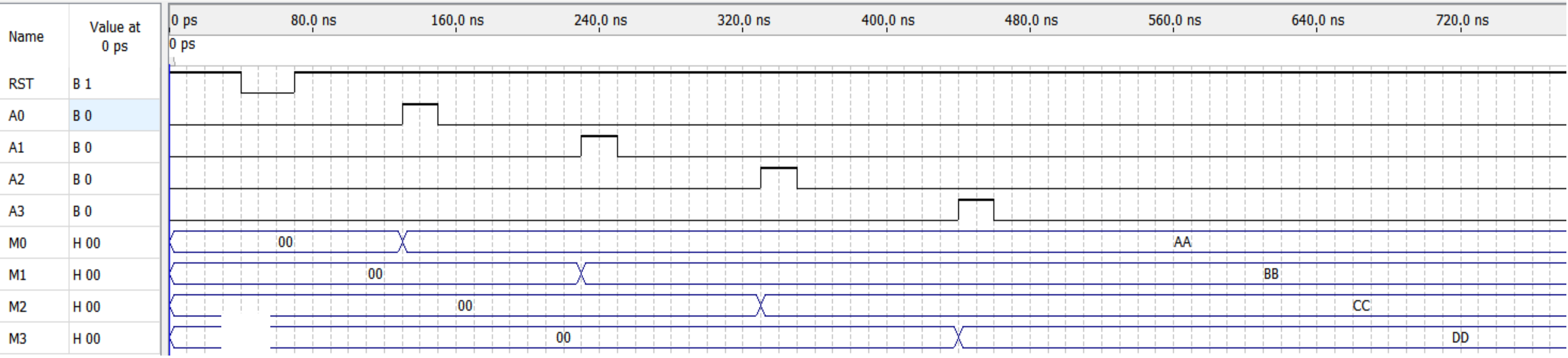
Byte organized Memory array

4-byte Memory array

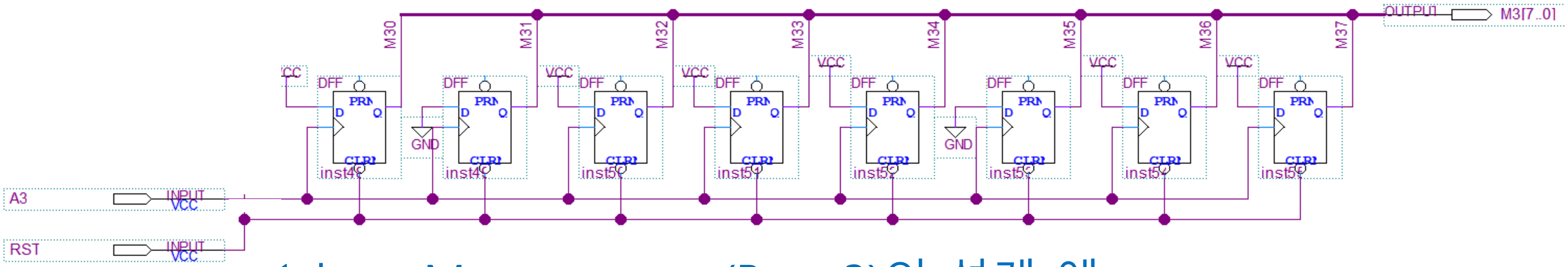


Memory 출력 data

Byte 0 : AA
Byte 1 : BB
Byte 2 : CC
Byre 3 : DD

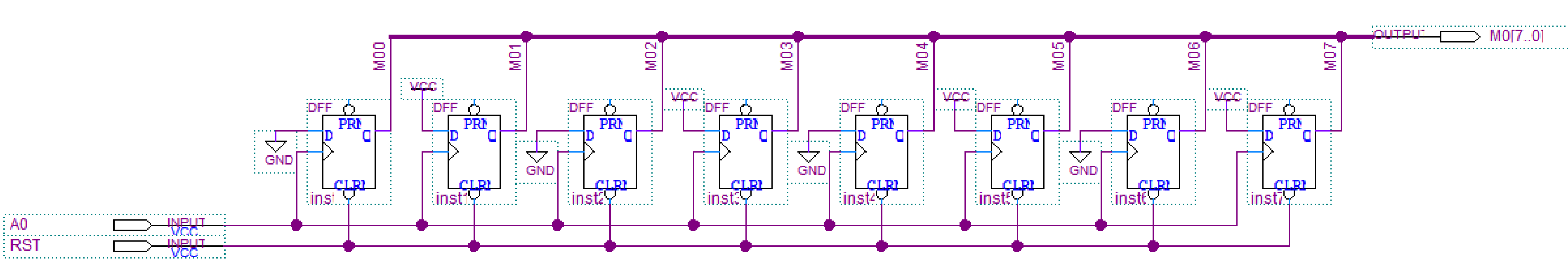


Name	Value at 0 ps
RST	B 1
A0	B 0
A1	B 0
A2	B 0
A3	B 0
M0	H 00
M1	H 00
M2	H 00
M3	H 00



1-byte Memory array(Byte 3)의 설계 예 (DD : 11011101)

1-byte Memory array의 설계 예 (AA : 10101010)



Q0 Q1 Q2 Q3 Q4 Q5 Q6 Q7

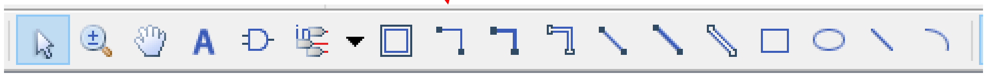
0 1 0 1 0 1 0 1 : AA

LSB MSB

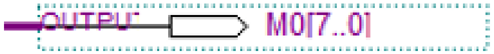
Quartus에서 BUS 사용법

1) 선이 굵은 BUS ICON을 눌러 드래그 하면 굵은 선의 BUS가 그려진다

Wire(bit) BUS

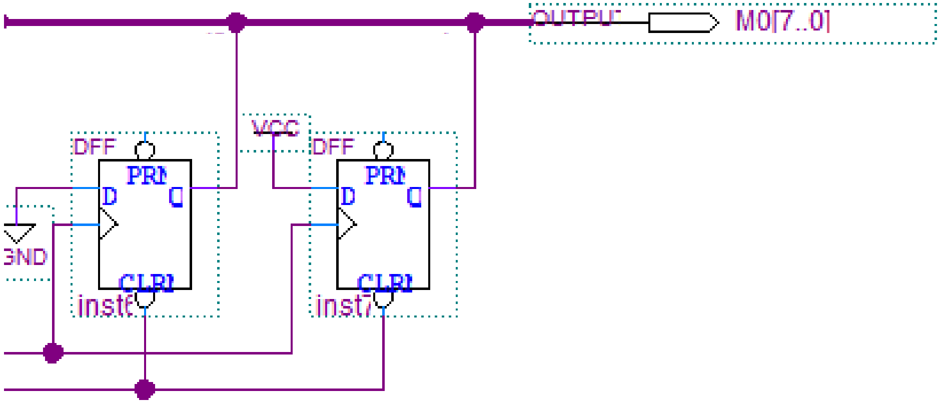


2) Input, Output pad에 BUS의 크기를 적는다. M0[7..0] : 8bit Bus

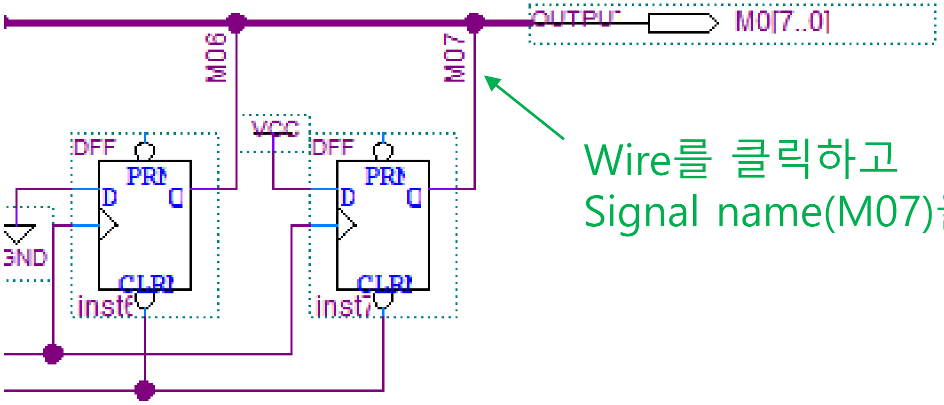


[7..0] 형식에 유의할 것

3) D F/F을 BUS에 wire로 연결한다.

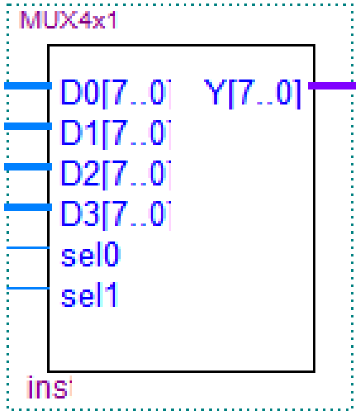


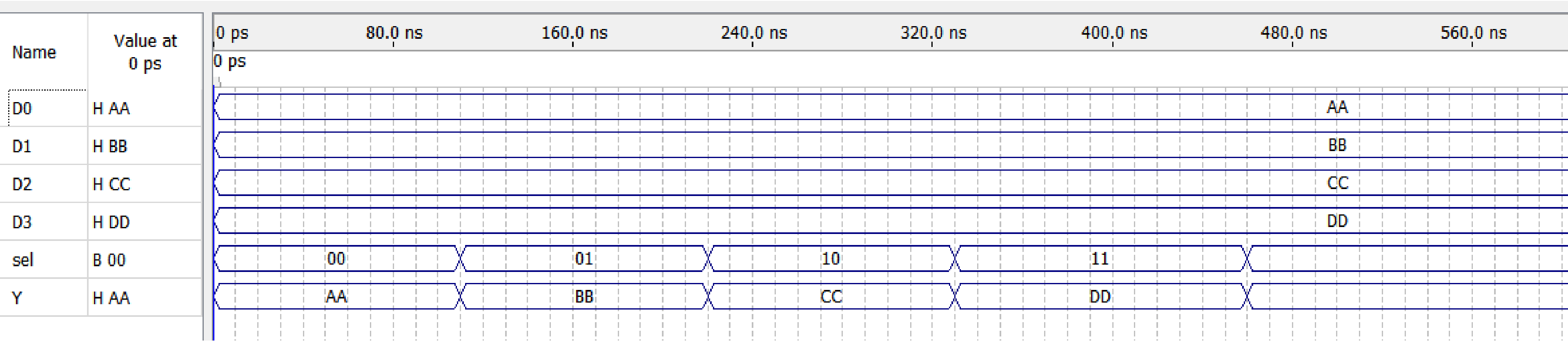
4) BUS에 연결한 각각의 wire에 name을 추가한다.(M00 ~ M07)



Wire를 클릭하고
Signal name(M07)을 타이핑 한다.

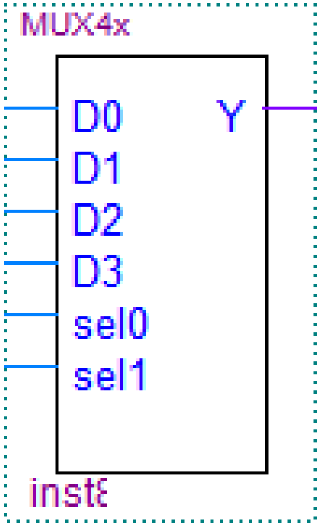
MUX4x1_8bits

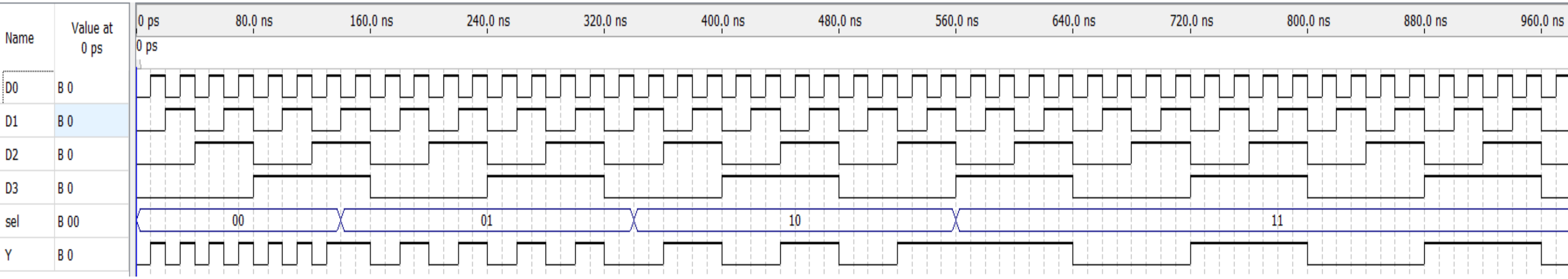




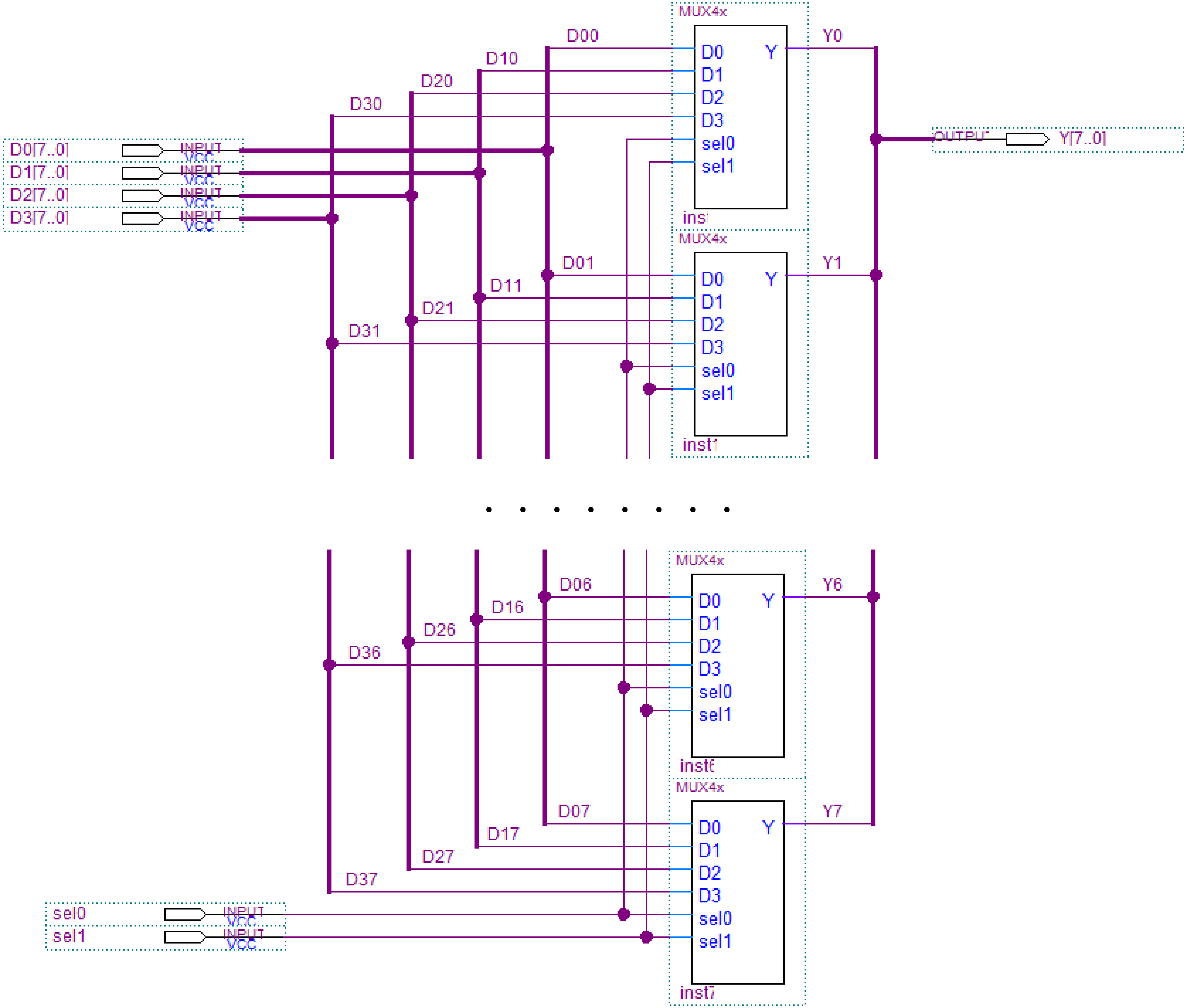
Name	Value at 0 ps
D0	H AA
D1	H BB
D2	H CC
D3	H DD
sel	B 00
Y	H AA

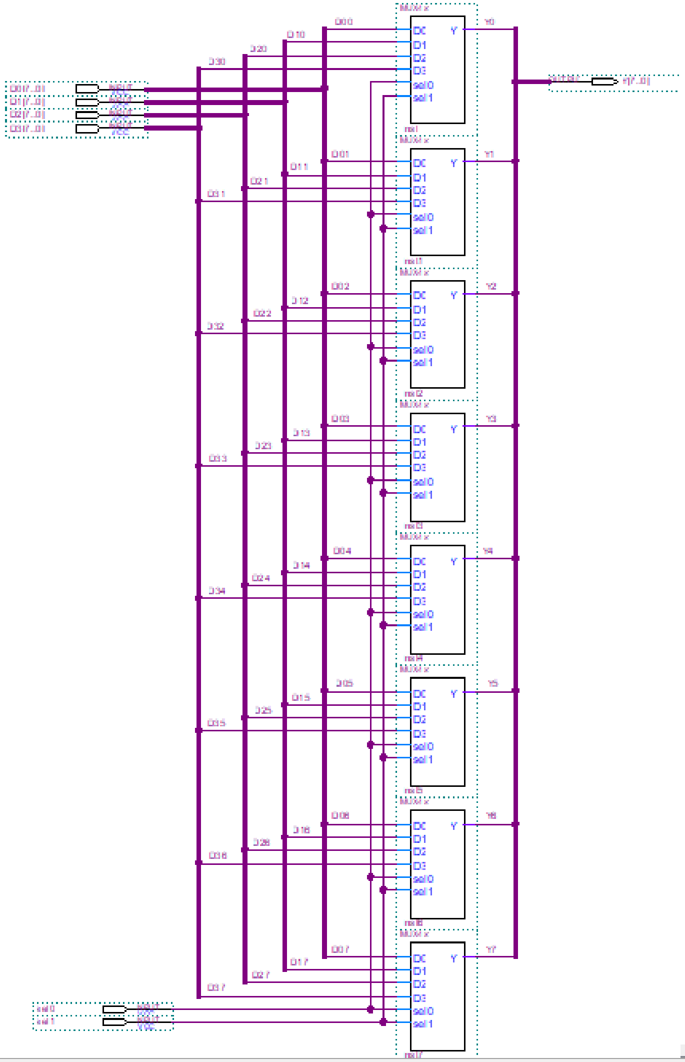
MUX4x1_1bit



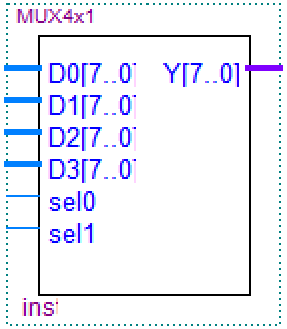


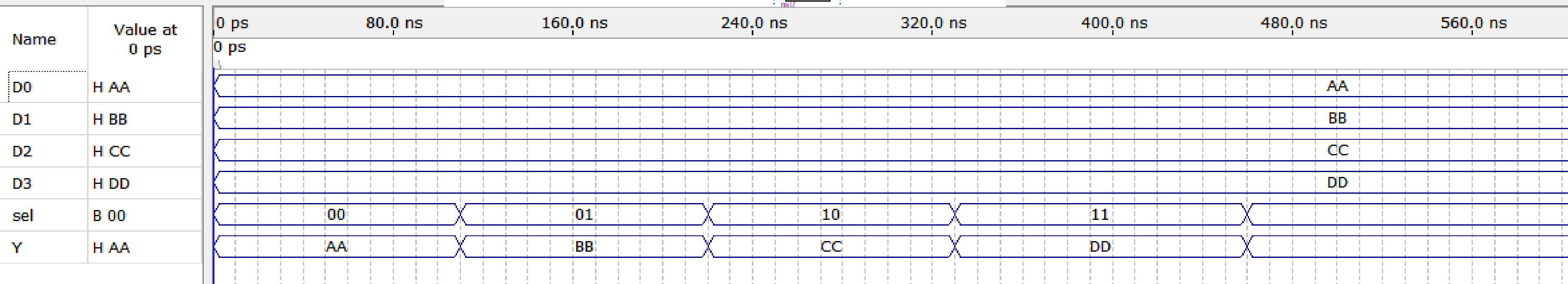
Name	Value at 0 ps
D0	B 0
D1	B 0
D2	B 0
D3	B 0
sel	B 00
Y	B 0


D0[7..0]
D1[7..0]
D2[7..0]
D3[7..0]
INPUT
VCC
D00
D10
D20
D30
MUX4x
D0
D1
D2
D3
sel0
sel1
Y
inst
Y0
OUTPUT
Y[7..0]
D01
D11
D21
D31
inst1
Y1
D06
D16
D26
D36
inst6
Y6
D07
D17
D27
D37
inst7
Y7
sel0
sel1



MUX4x1_8bits

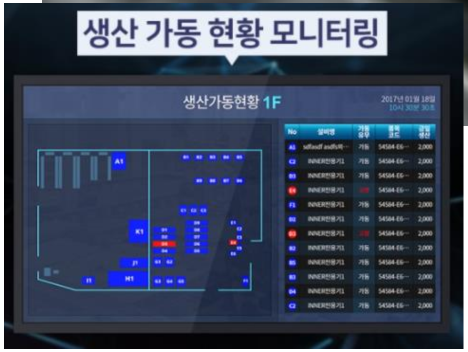




Name	Value at 0 ps
D0	H AA
D1	H BB
D2	H CC
D3	H DD
sel	B 00
Y	H AA

제약회사 직무용어 탐방기

50

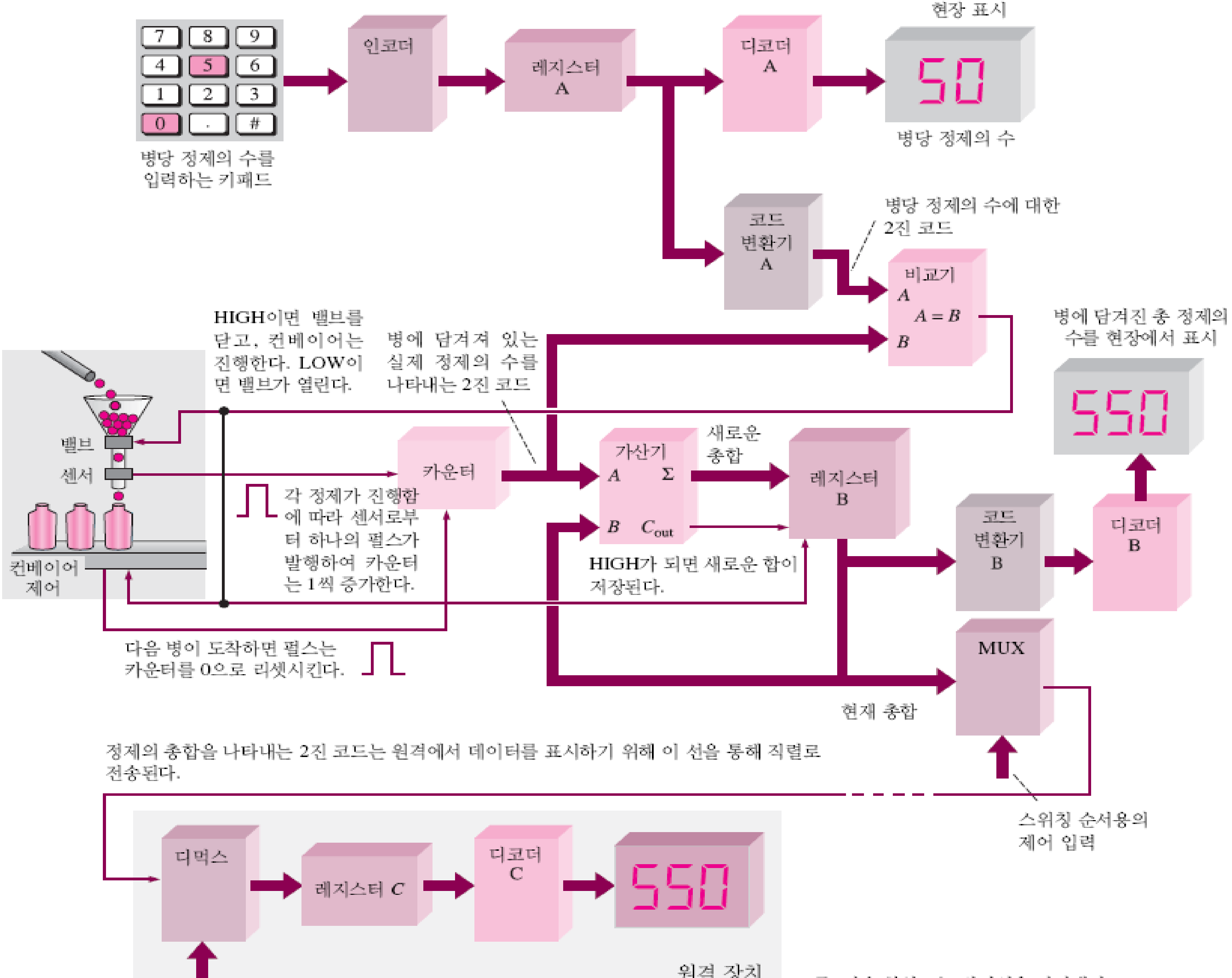


생산 가동 현황 모니터링

생산가동현황 1F

2017년 01월 18일
10시 30분 30초

No	설비명	가동유무	품목코드	금일생산
A1	sdfasdf asdfs希···	가동	54584-E6···	2,000
C2	INNER전용기1	가동	54584-E6···	2,000
B3	INNER전용기1	가동	54584-E6···	2,000
E4	INNER전용기1	[illegible]	54584-E6···	2,000
F1	INNER전용기1	가동	54584-E6···	2,000
B2	INNER전용기1	가동	54584-E6···	2,000
E3	INNER전용기1	[illegible]	54584-E6···	2,000
B2	INNER전용기1	가동	54584-E6···	2,000
B5	INNER전용기1	가동	54584-E6···	2,000
B3	INNER전용기1	가동	54584-E6···	2,000
D4	INNER전용기1	가동	54584-E6···	2,000
C2	INNER전용기1	가동	54584-E6···	2,000


7 8 9
4 5 6
1 2 3
0 . #
병당 정제의 수를 입력하는 키패드
인코더
레지스터 A
디코더 A
현장 표시
50
병당 정제의 수
코드 변환기 A
병당 정제의 수에 대한 2진 코드
비교기
A
A = B
B
병에 담겨진 총 정제의 수를 현장에서 표시
550
HIGH이면 밸브를 닫고, 컨베이어는 진행한다. LOW이면 밸브가 열린다.
병에 담겨져 있는 실제 정제의 수를 나타내는 2진 코드
밸브
센서
컨베이어 제어
카운터
각 정제가 진행함에 따라 센서로부터 하나의 펄스가 발행하여 카운터는 1씩 증가한다.
가산기
A
Σ
B
C_out
새로운 총합
레지스터 B
HIGH가 되면 새로운 합이 저장된다.
코드 변환기 B
디코더 B
다음 병이 도착하면 펄스는 카운터를 0으로 리셋시킨다.
MUX
현재 총합
정제의 총합을 나타내는 2진 코드는 원격에서 데이터를 표시하기 위해 이 선을 통해 직렬로 전송된다.
스위칭 순서용의 제어 입력
디먹스
레지스터 C
디코더 C
550
원격 장치

주: 넓은 화살표는 병렬선을 나타낸다.

수고하셨습니다!!!