

Memory Arrays (4H)

Gate Level Design

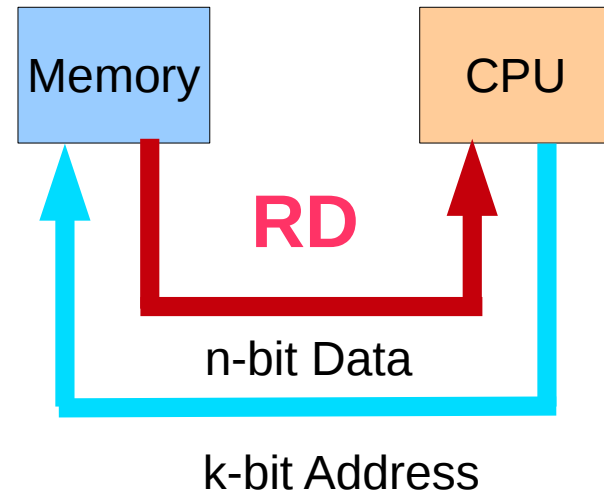
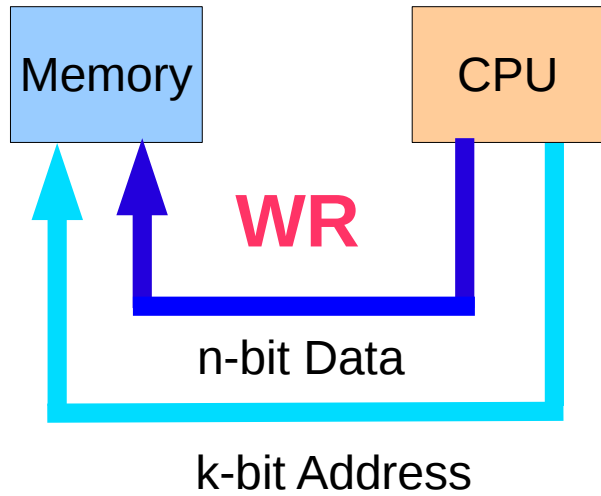
Copyright (c) 2011, 2016 Young W. Lim.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

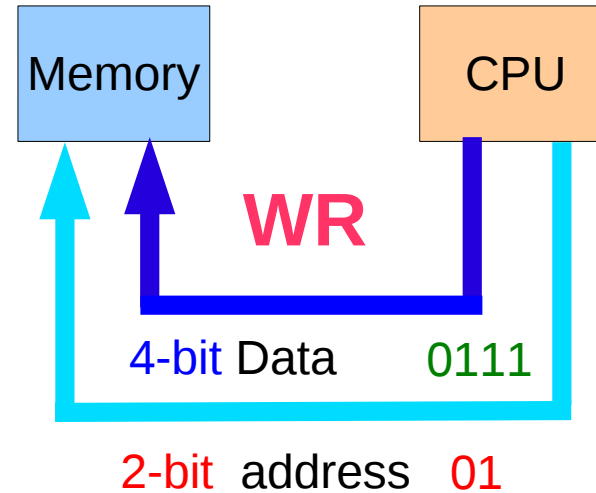
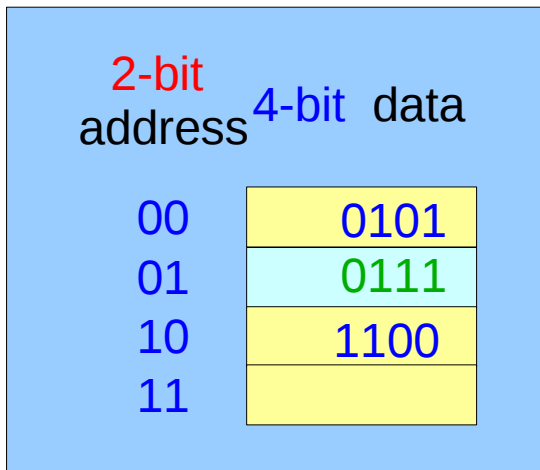
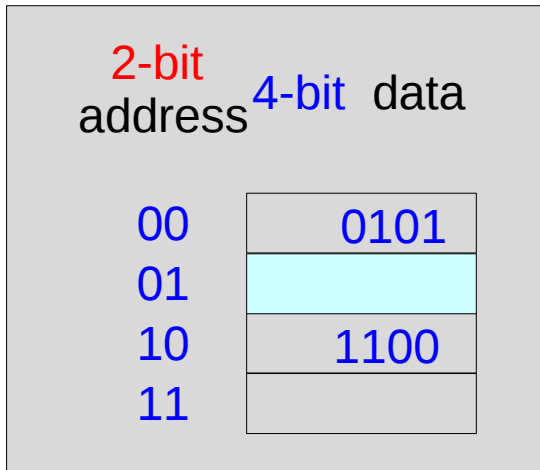
Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using OpenOffice.

Memory Access Operations



A Memory Write Example



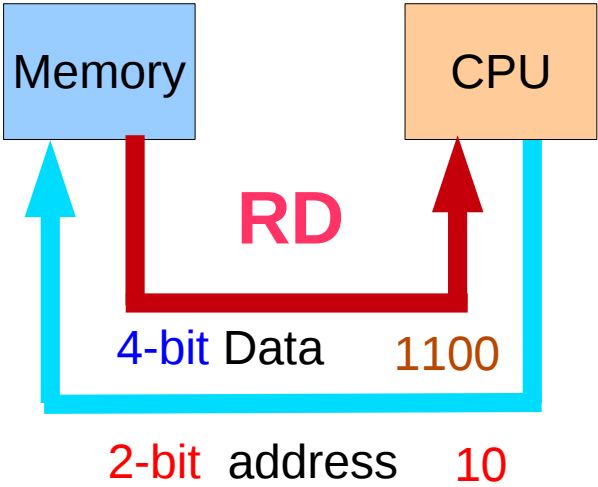
WR

Address: 01

Data: 0111

A Memory Read Example

2-bit address	4-bit data
00	0101
01	0111
10	1100
11	

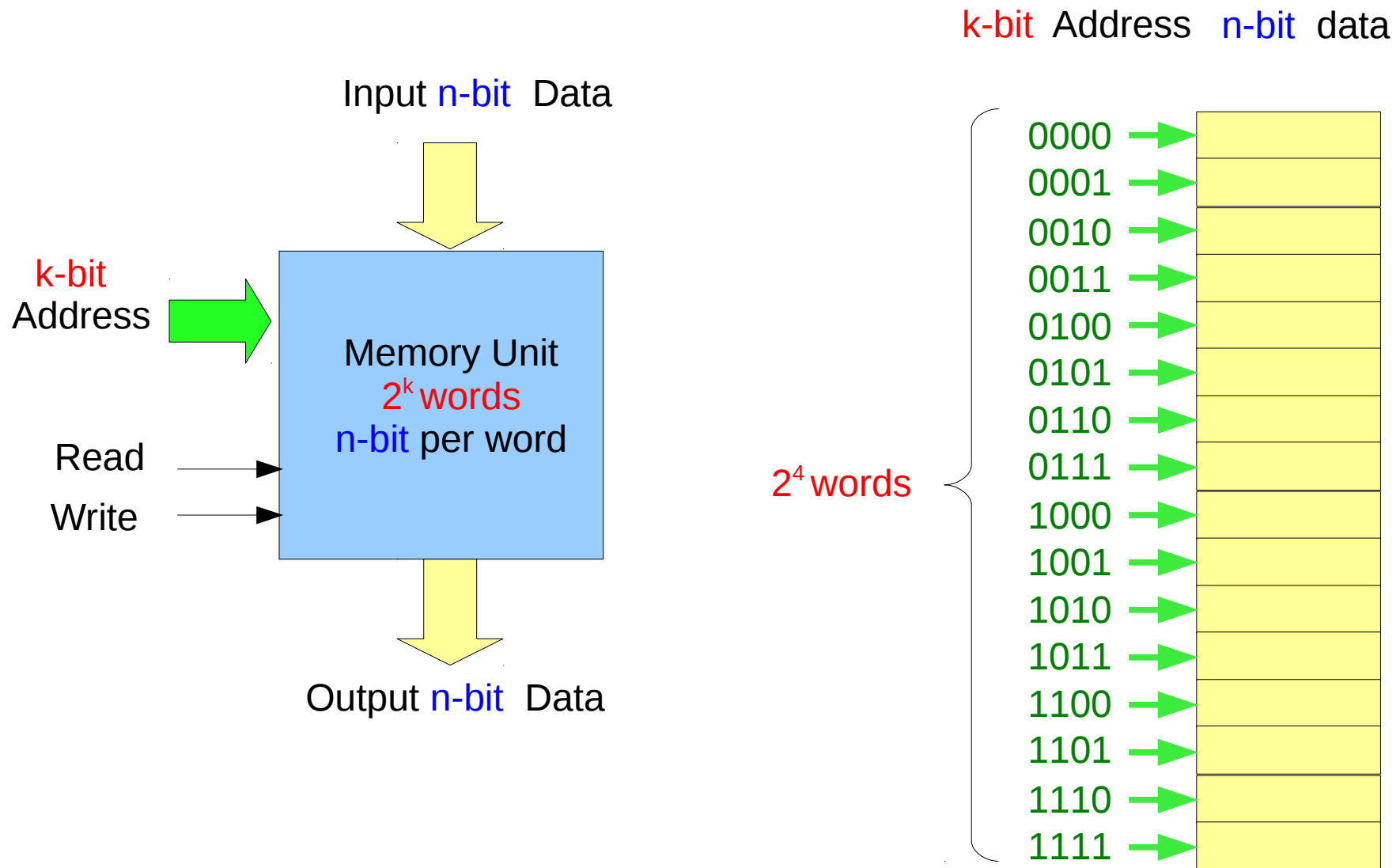


RD

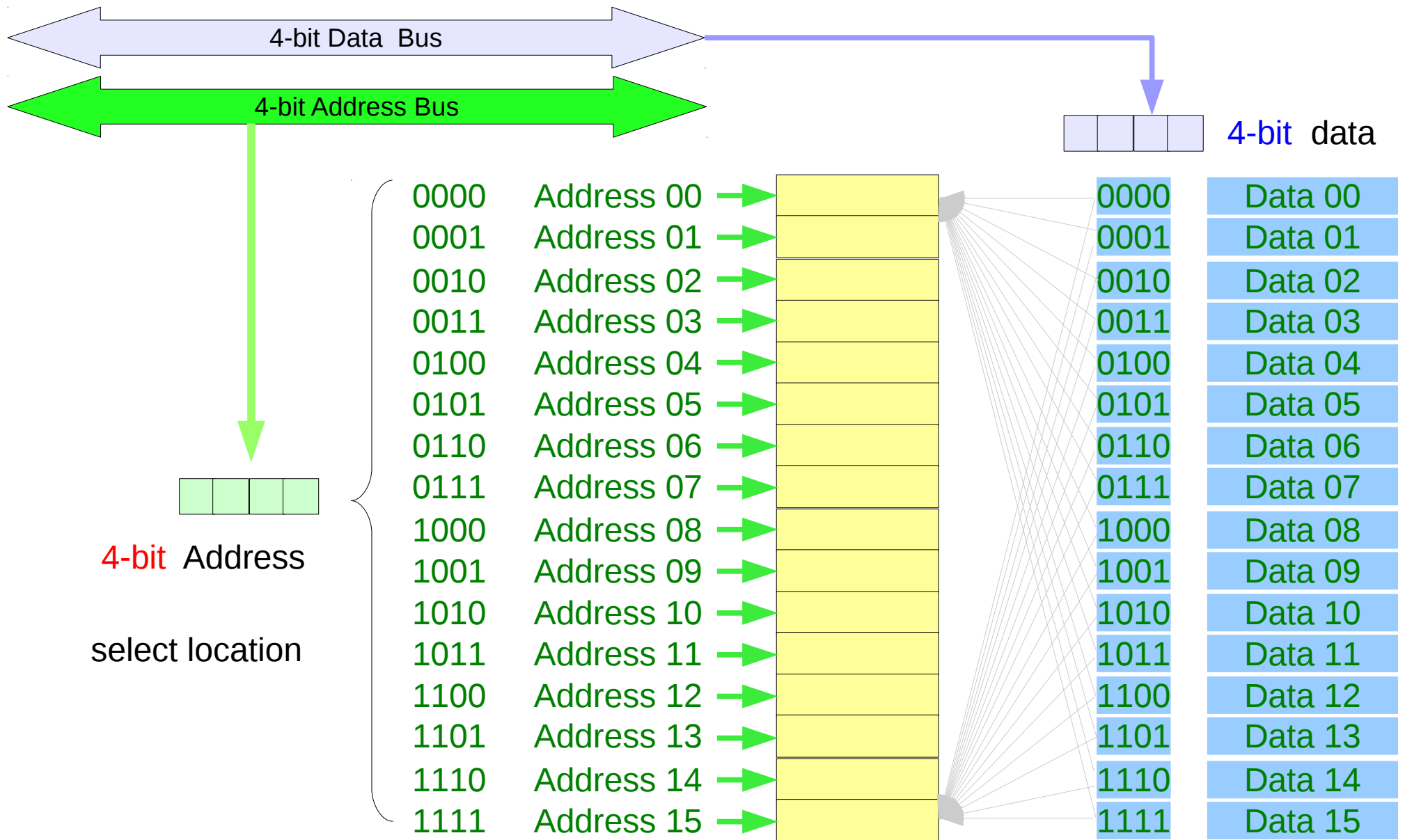
Address: 10

Data: → 1100

A Memory and A Memory Map



Address Bits and Data Bits



Increasing Memory Locations

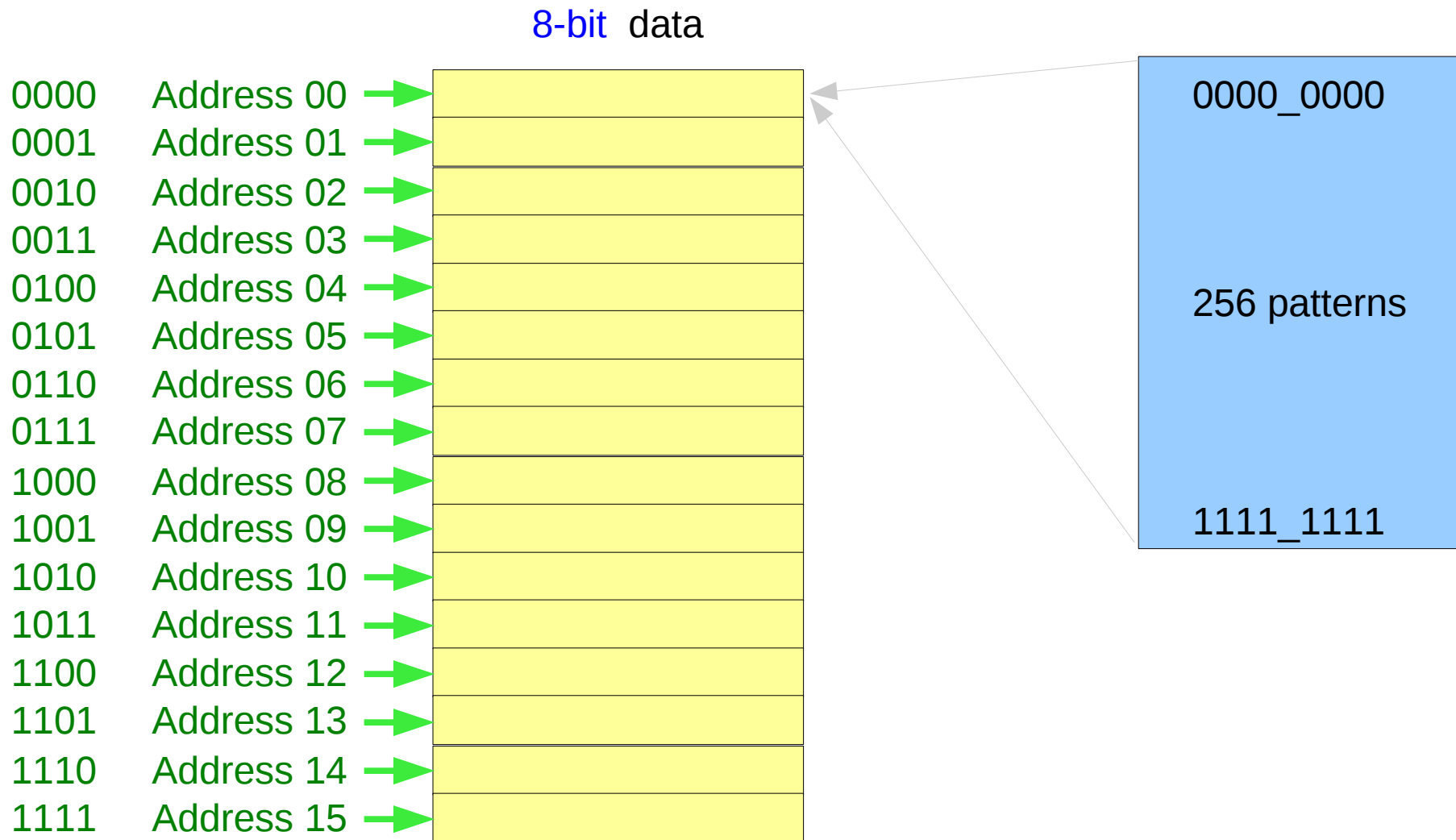
5-bit Address

00000	Address 00	→	
00001	Address 01	→	
00010	Address 02	→	
00011	Address 03	→	
00100	Address 04	→	
00101	Address 05	→	
00110	Address 06	→	
00111	Address 07	→	
01000	Address 08	→	
01001	Address 09	→	
01010	Address 10	→	
01011	Address 11	→	
01100	Address 12	→	
01101	Address 13	→	
01110	Address 14	→	
01111	Address 15	→	

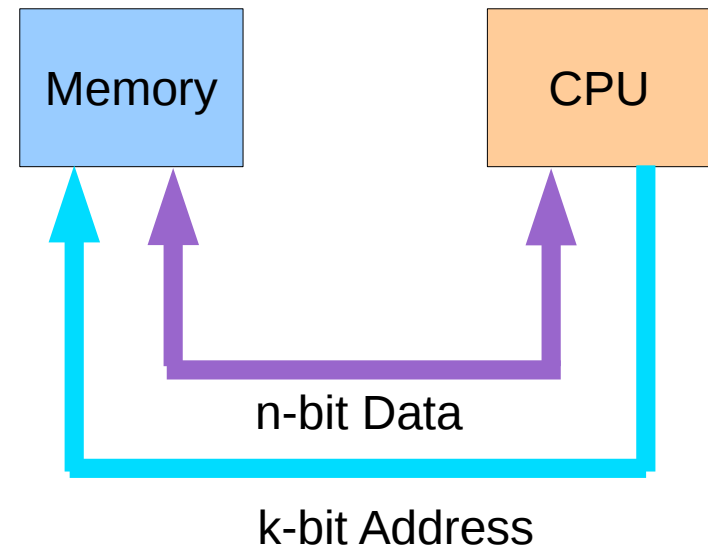
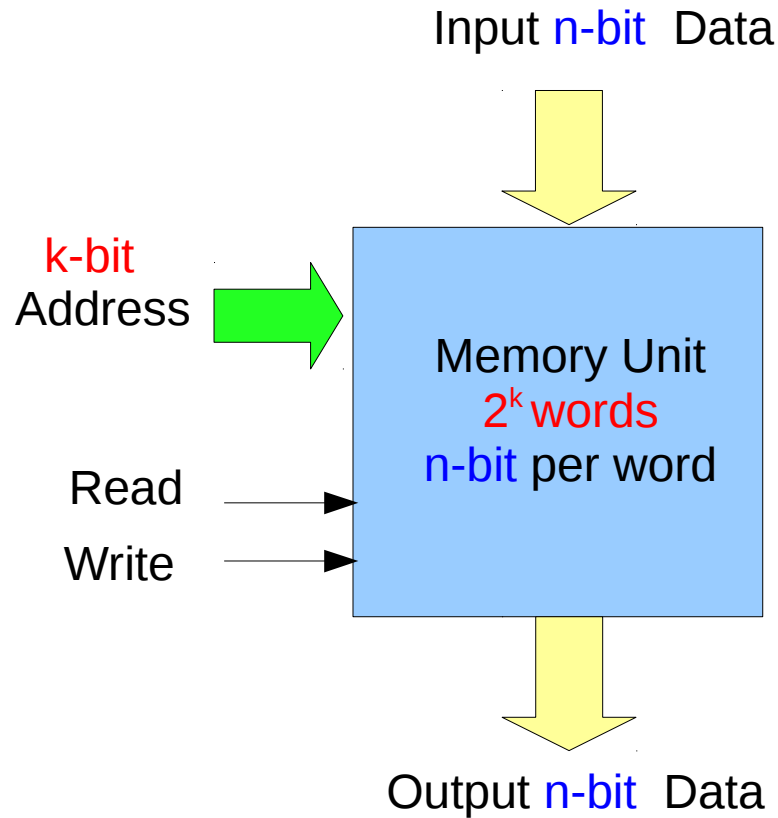
5-bit Address

10000	Address 16	→	
10001	Address 17	→	
10010	Address 18	→	
10011	Address 19	→	
10100	Address 20	→	
10101	Address 21	→	
10110	Address 22	→	
10111	Address 23	→	
11000	Address 24	→	
11001	Address 25	→	
11010	Address 26	→	
11011	Address 27	→	
11100	Address 28	→	
11101	Address 29	→	
11110	Address 30	→	
11111	Address 31	→	

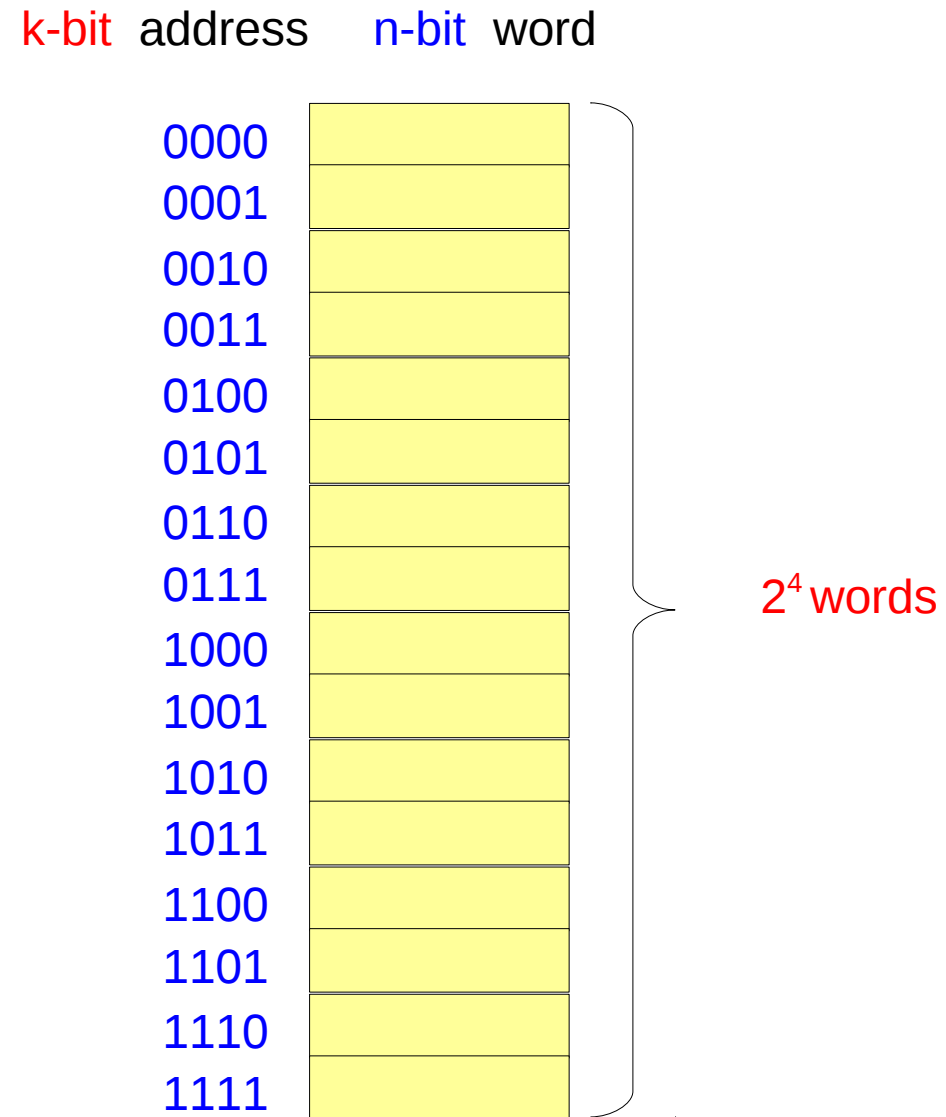
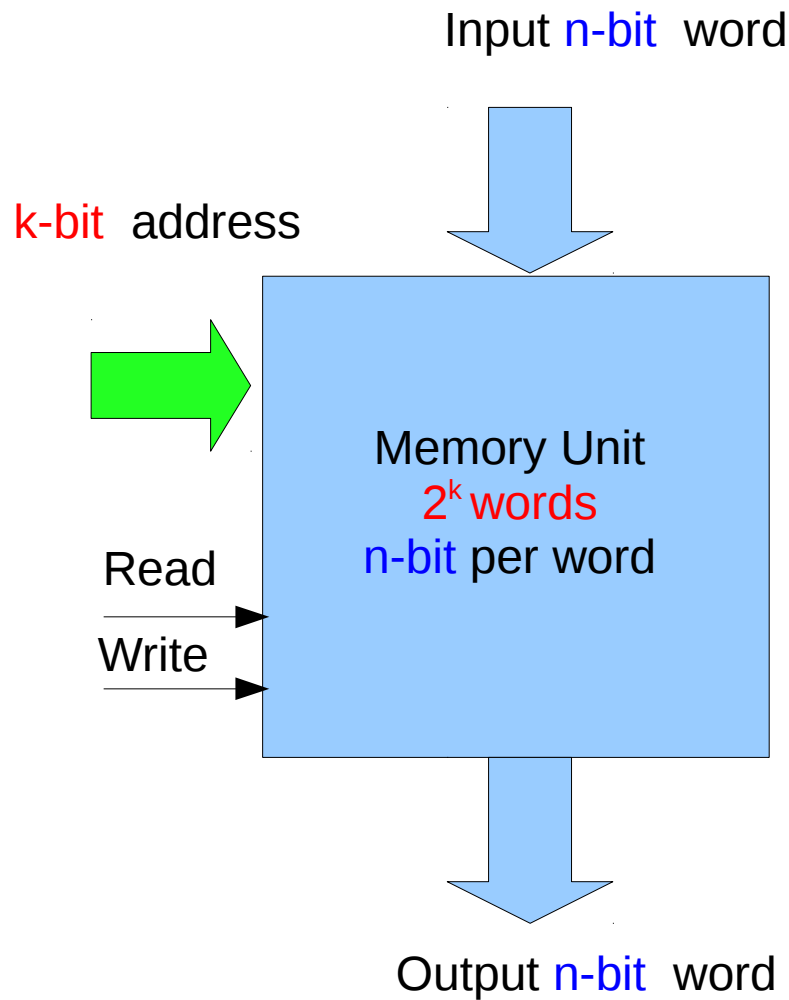
Increasing Memory Width



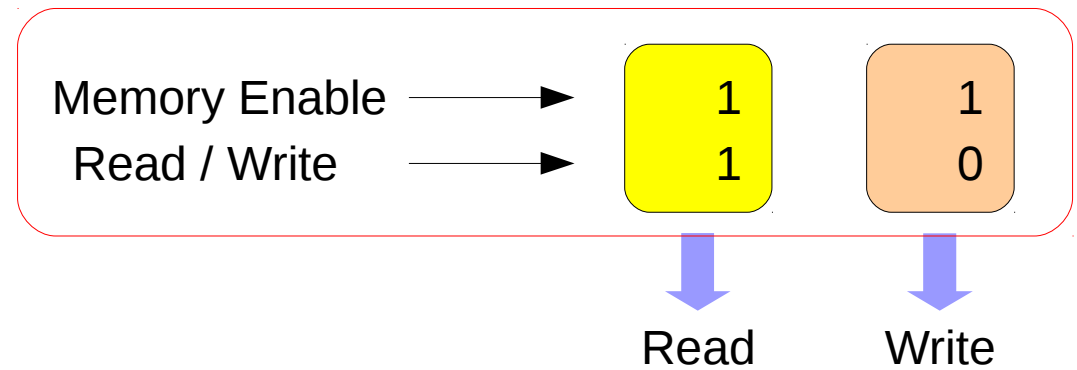
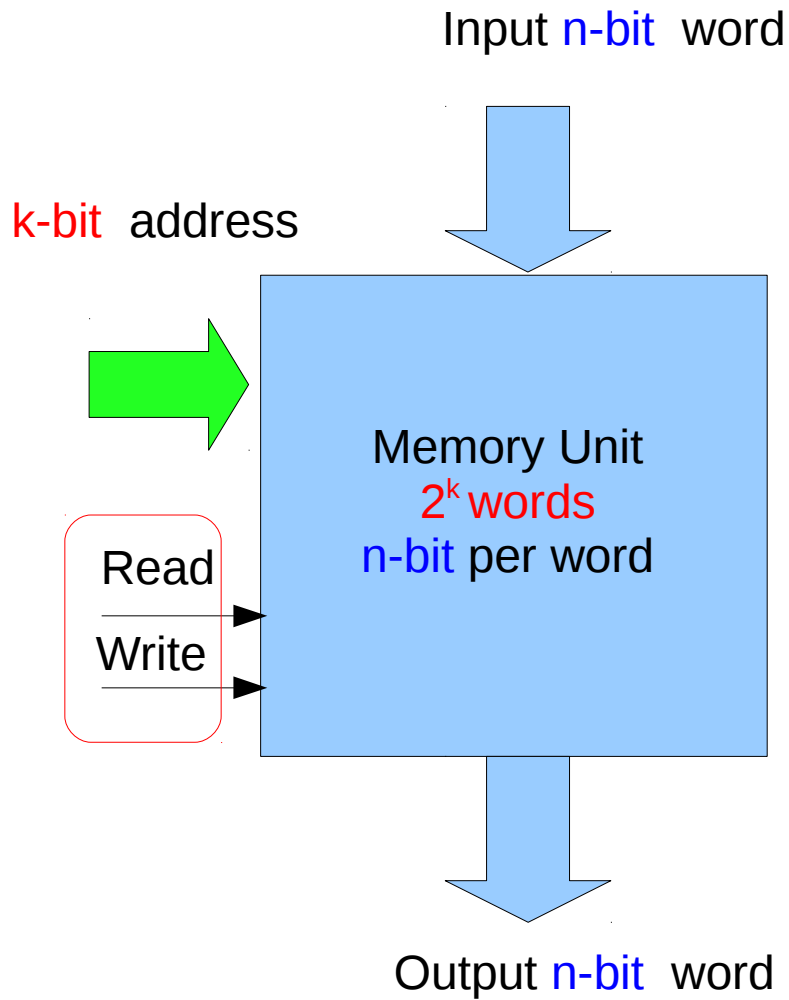
The Inputs and Outputs of a Memory



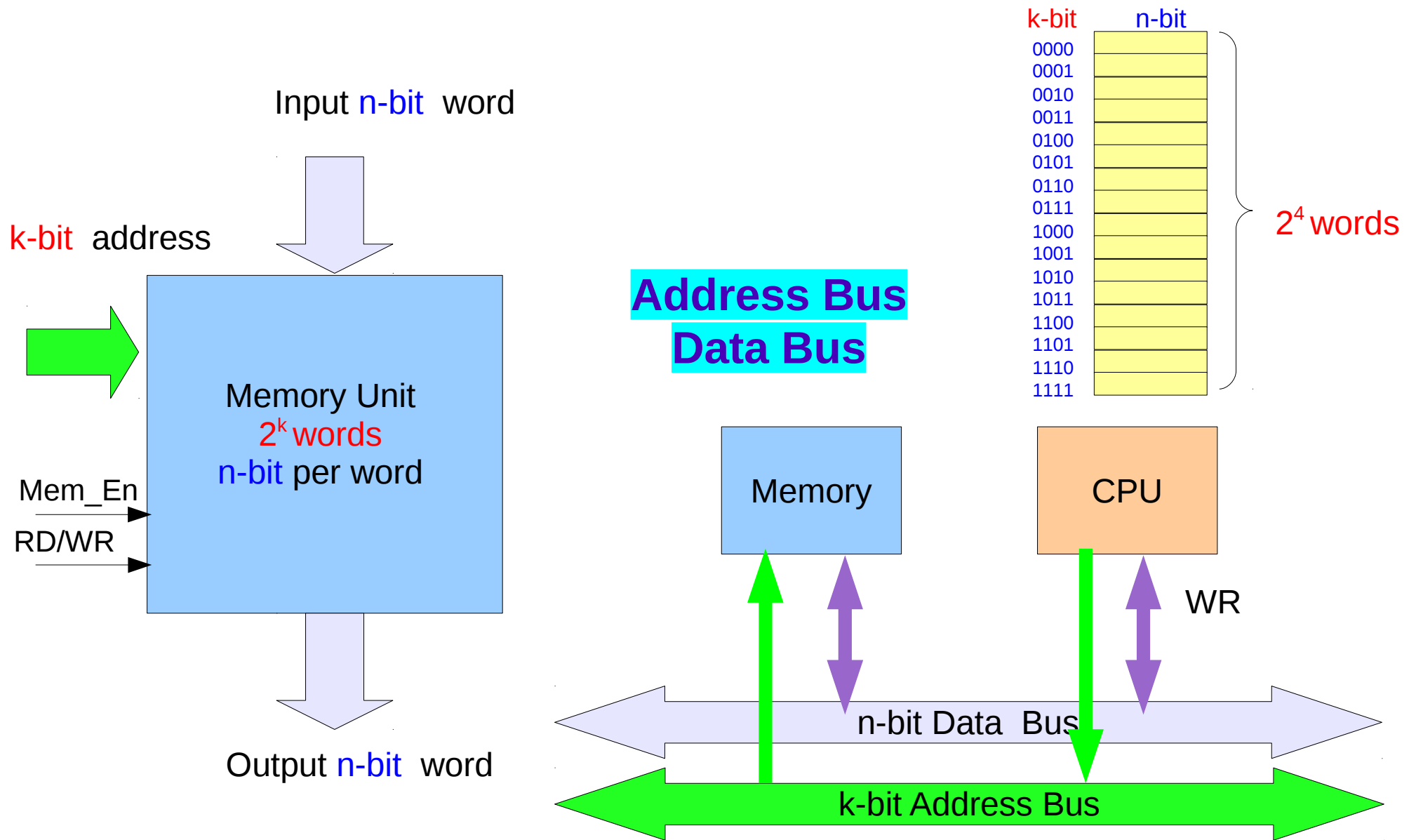
Block Diagram of a Memory Unit



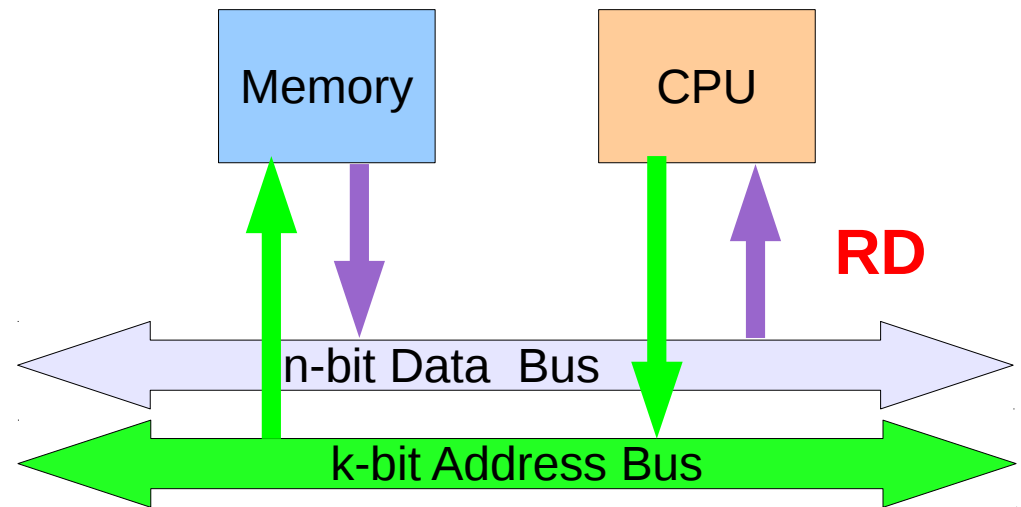
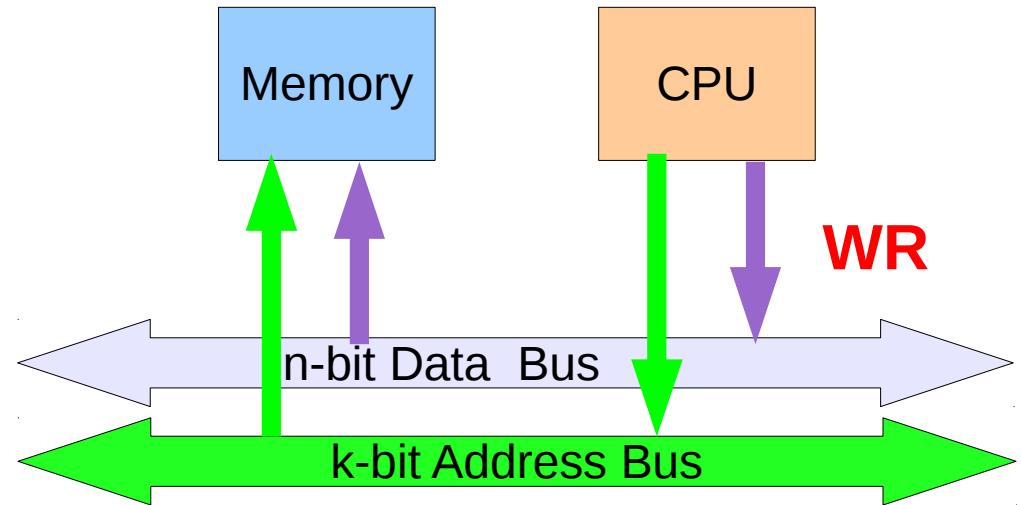
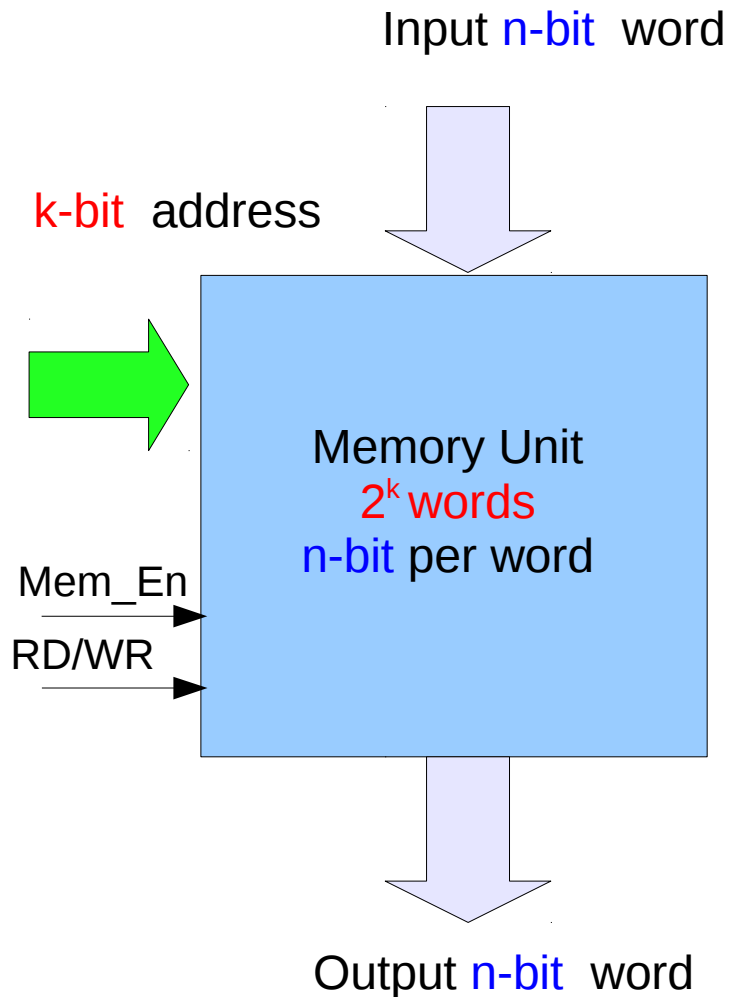
Memory Control Lines



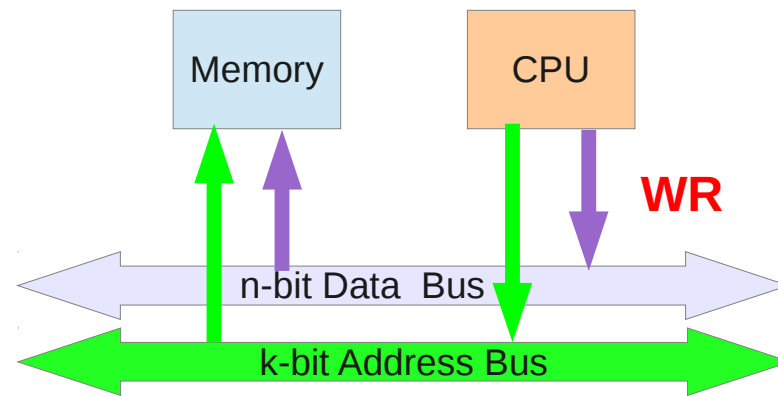
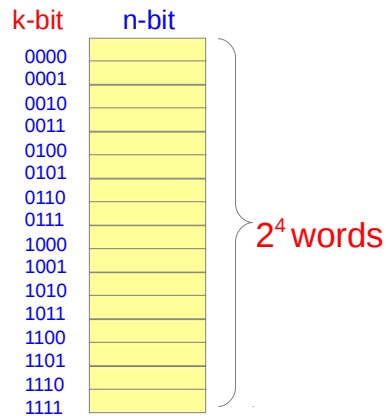
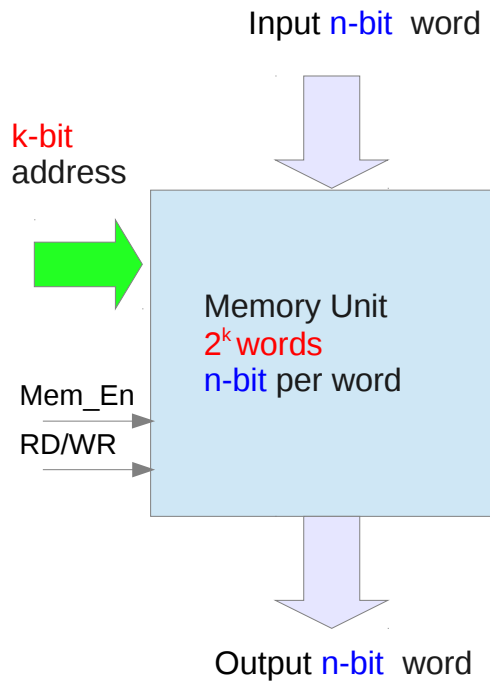
Address & Data Buses



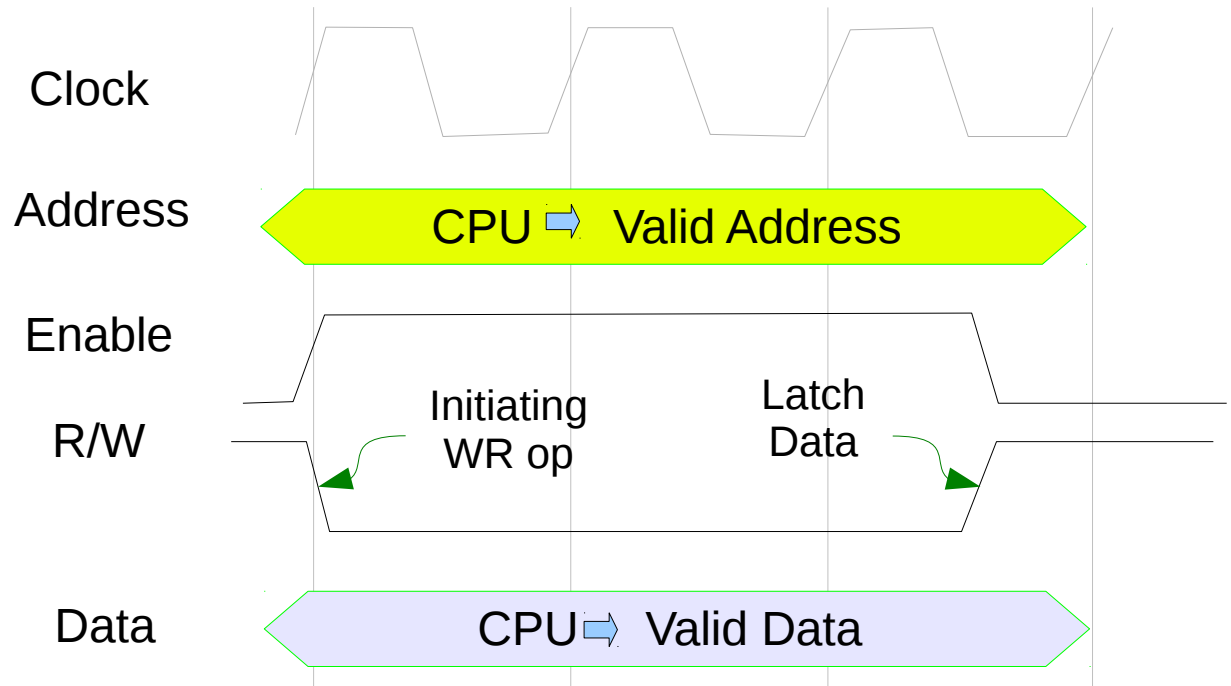
RD & WR Operations



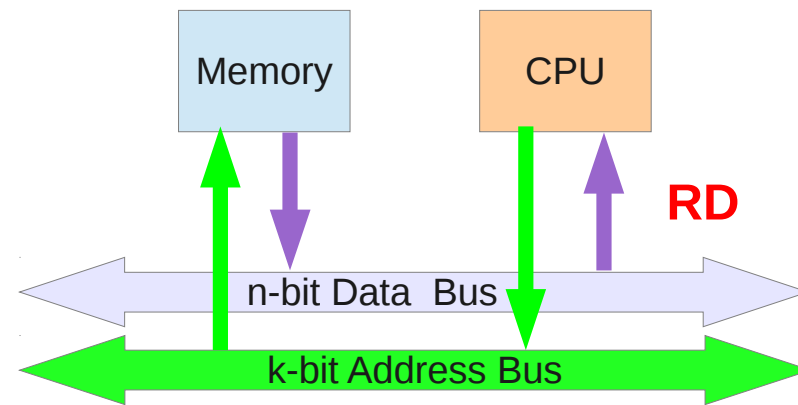
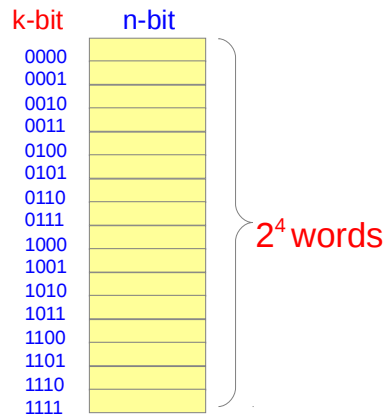
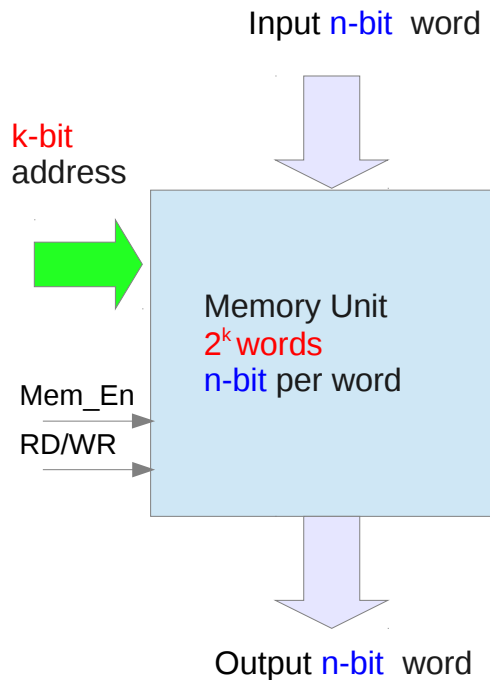
Memory Write Cycle



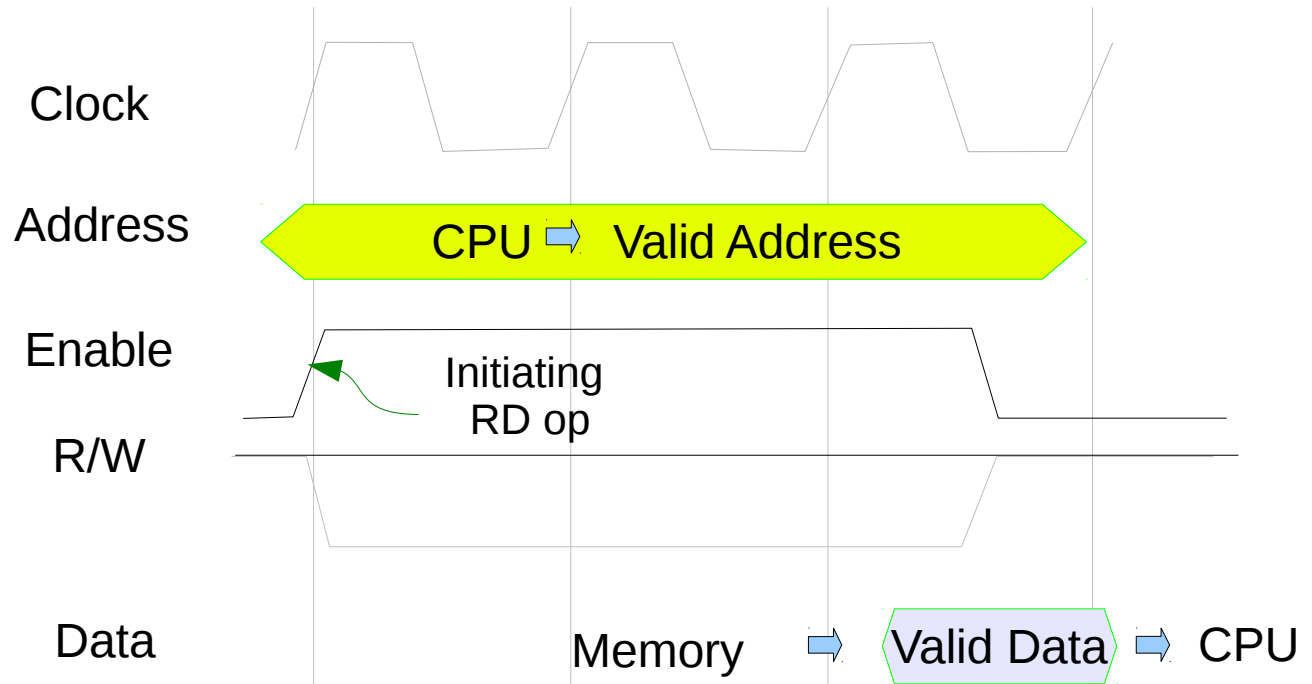
Usually, clock is seldom used. (shown for a reference)



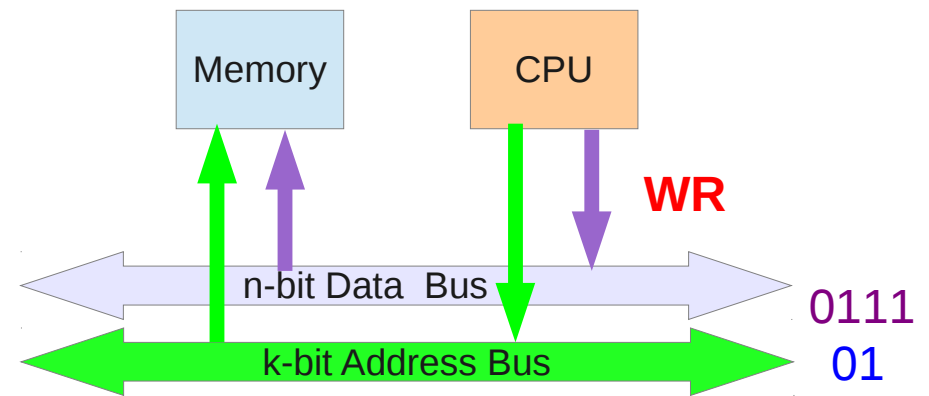
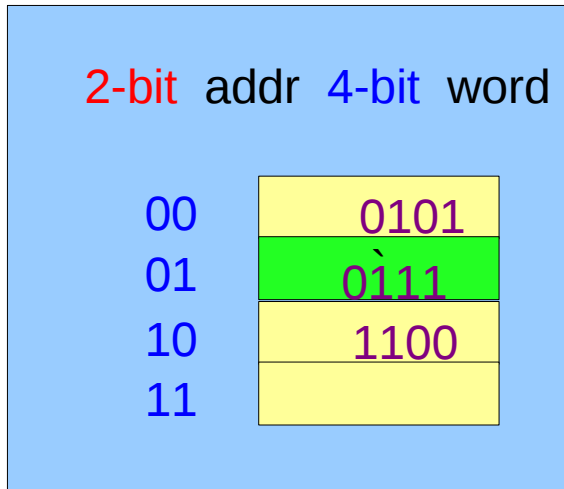
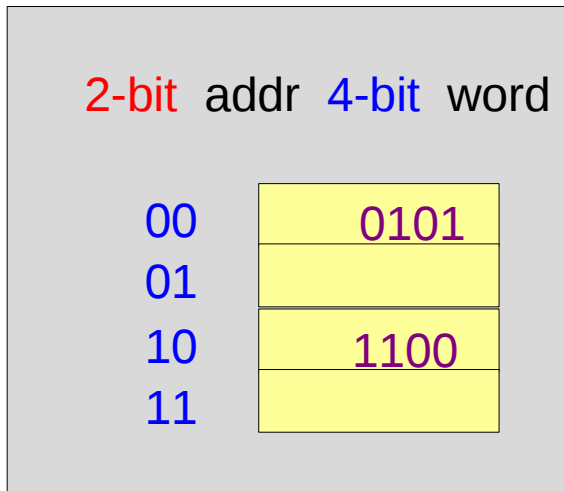
Memory Read Cycle



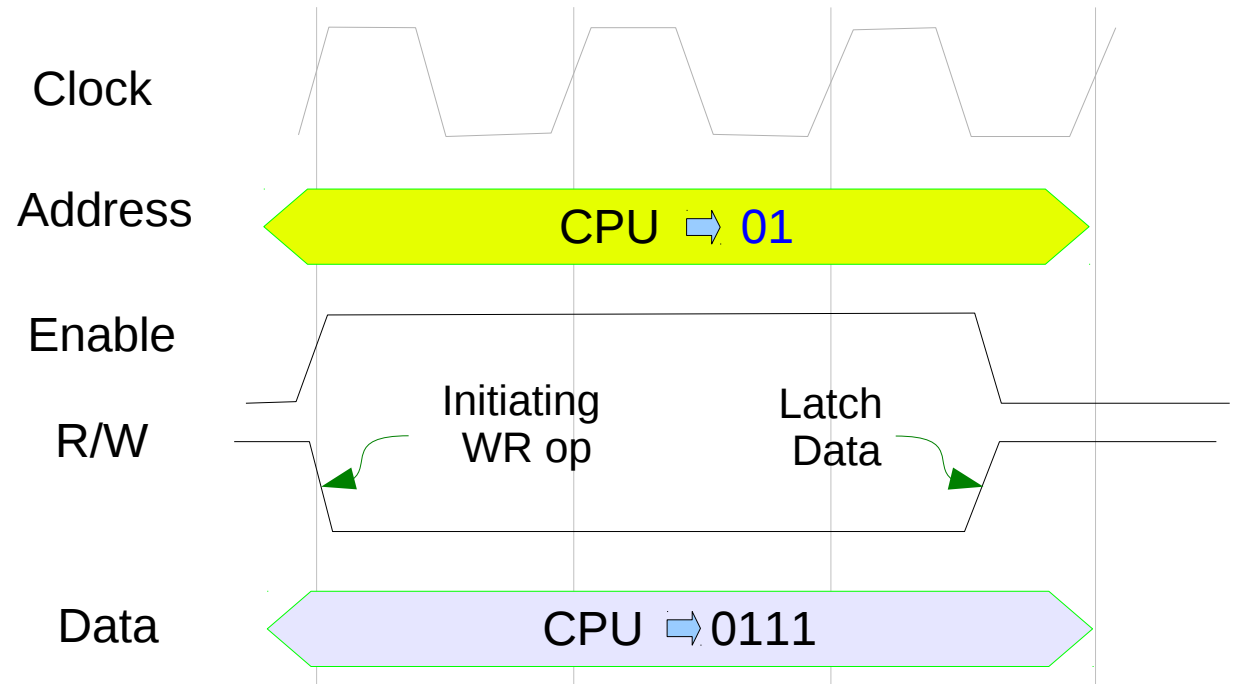
Usually, clock is seldom used. (shown for a reference)



Memory Write Example



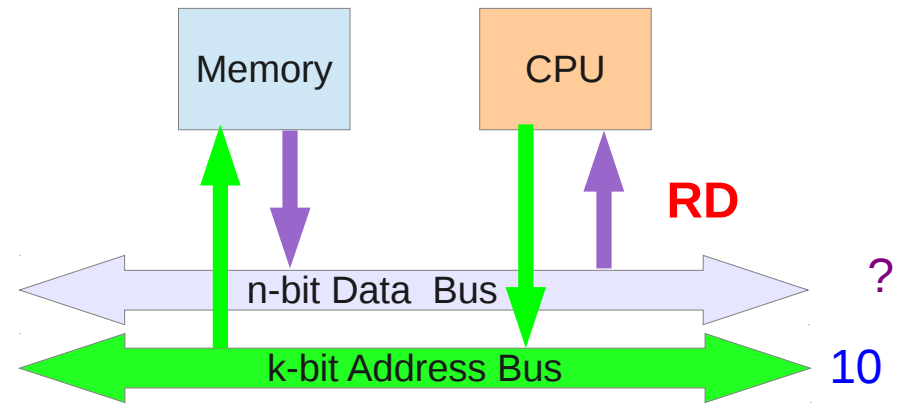
Usually, clock is seldom used. (shown for a reference)



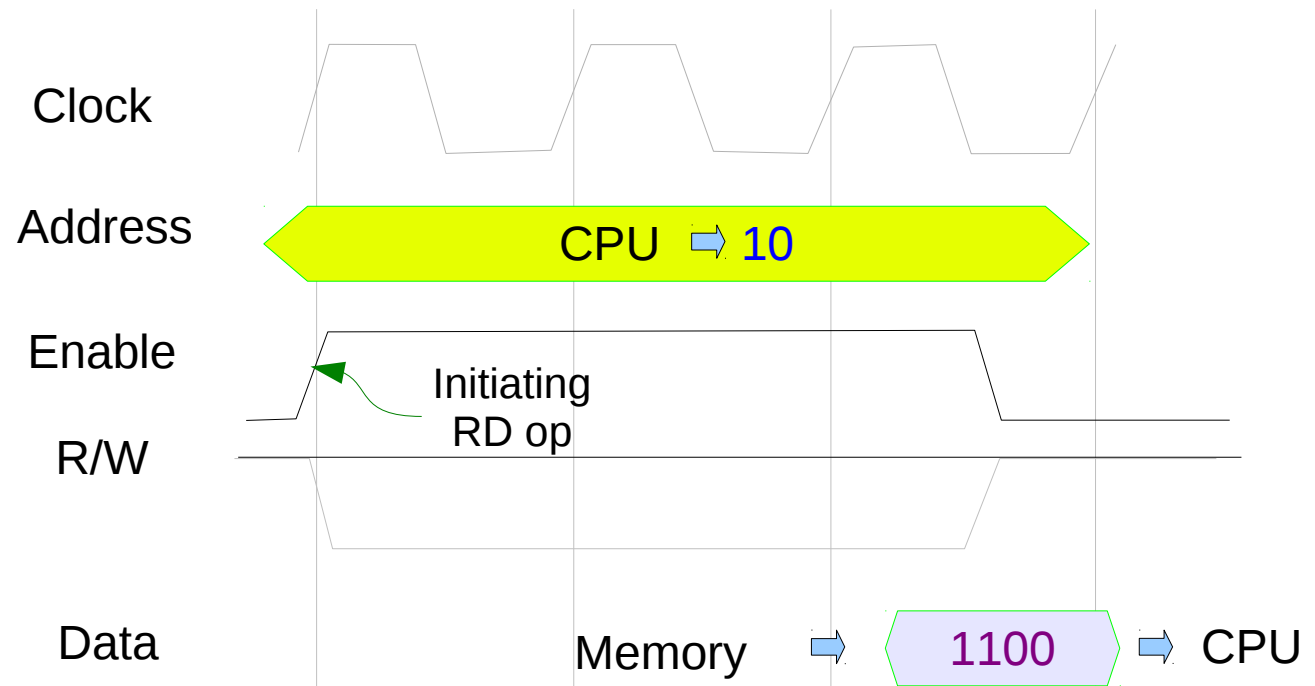
Memory Read Example

2-bit addr 4-bit word

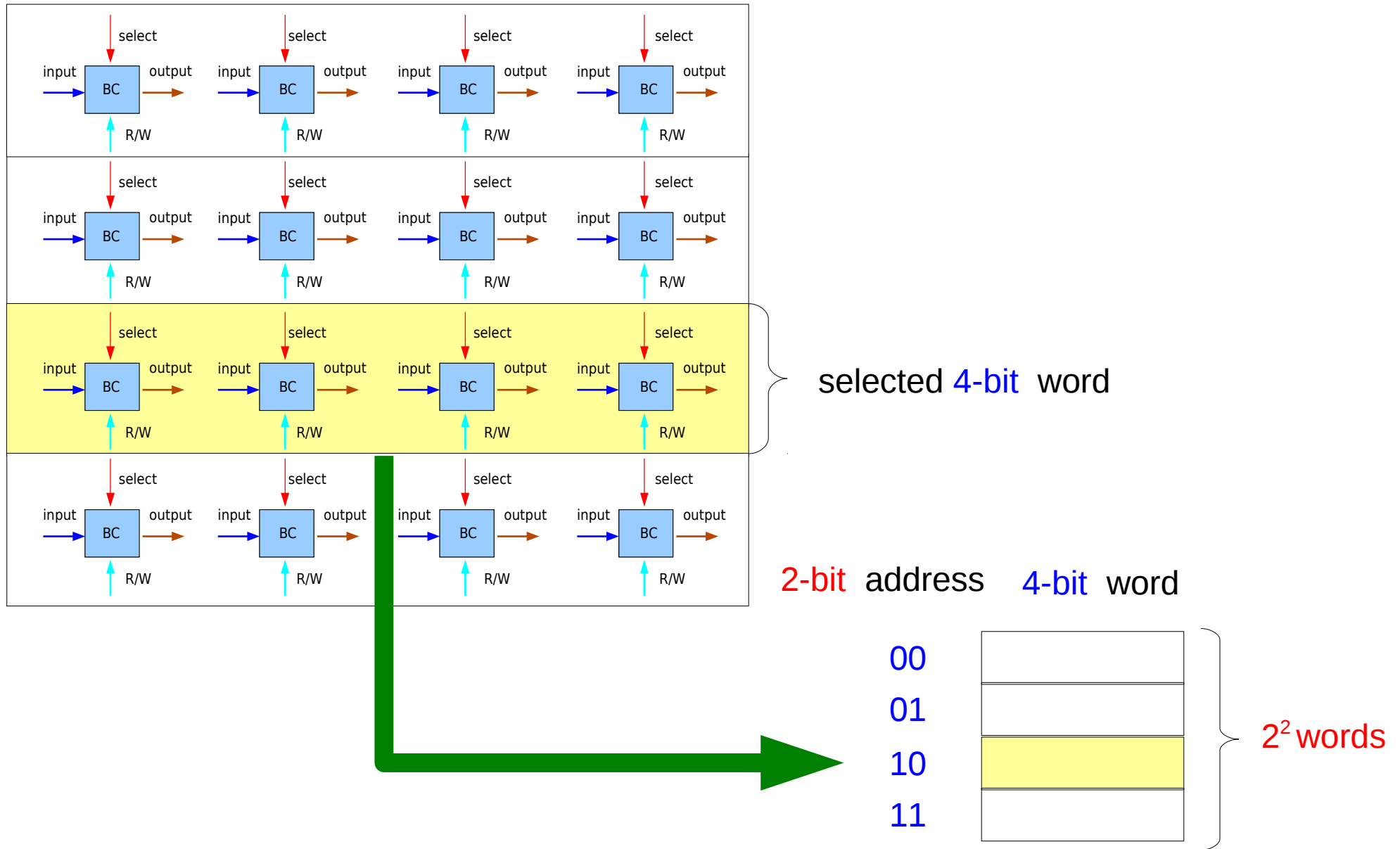
00	0101
01	
10	1100
11	



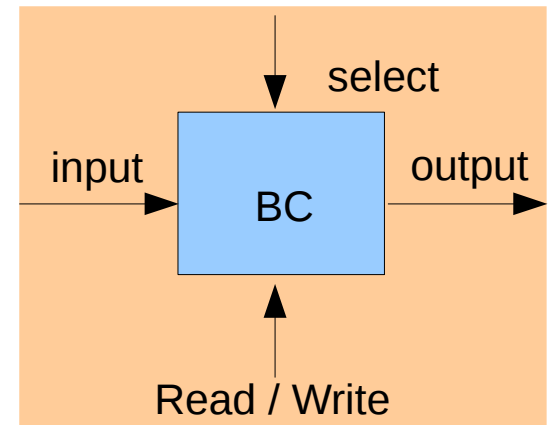
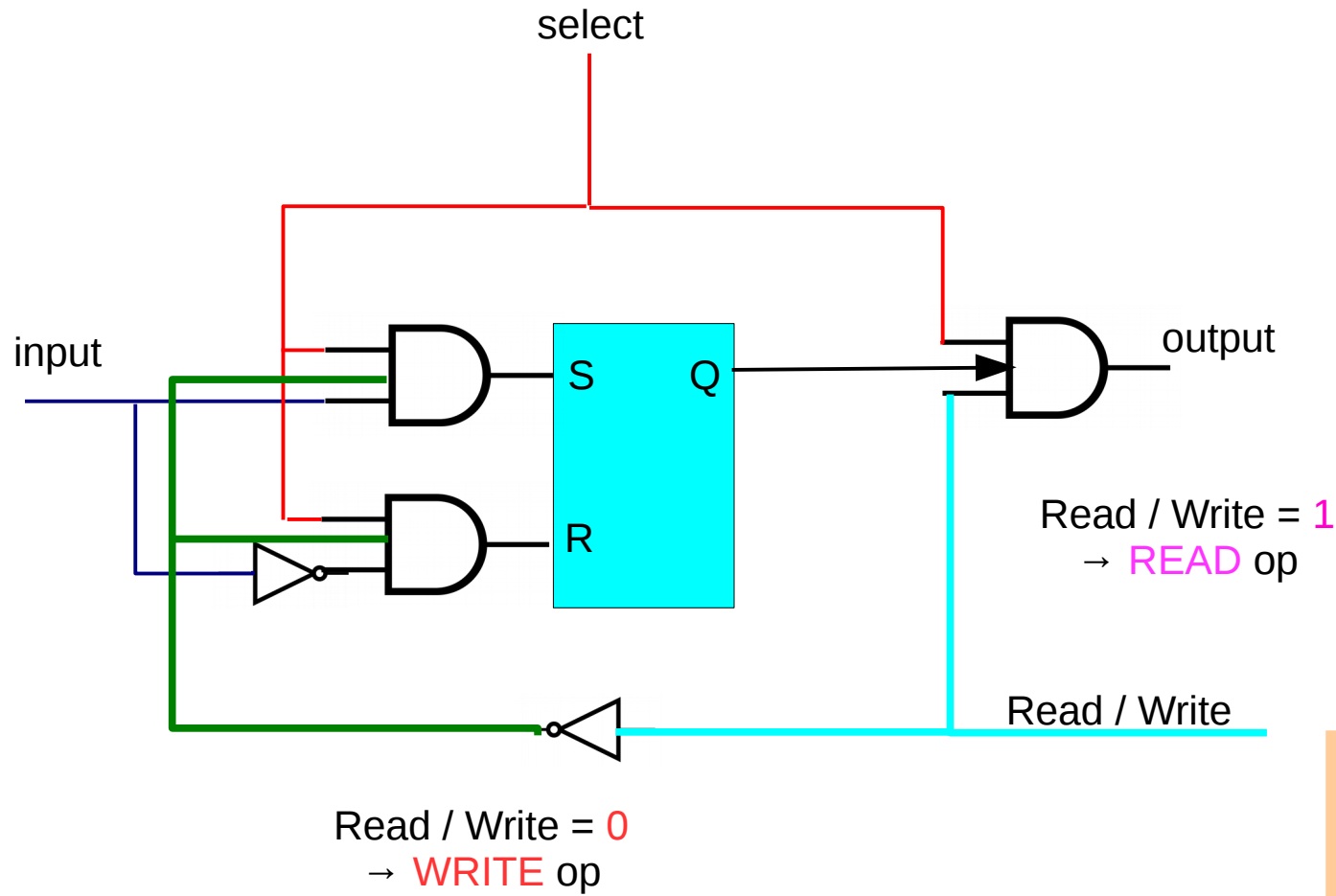
Usually, clock is seldom used. (shown for a reference)



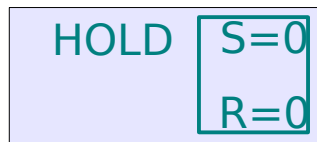
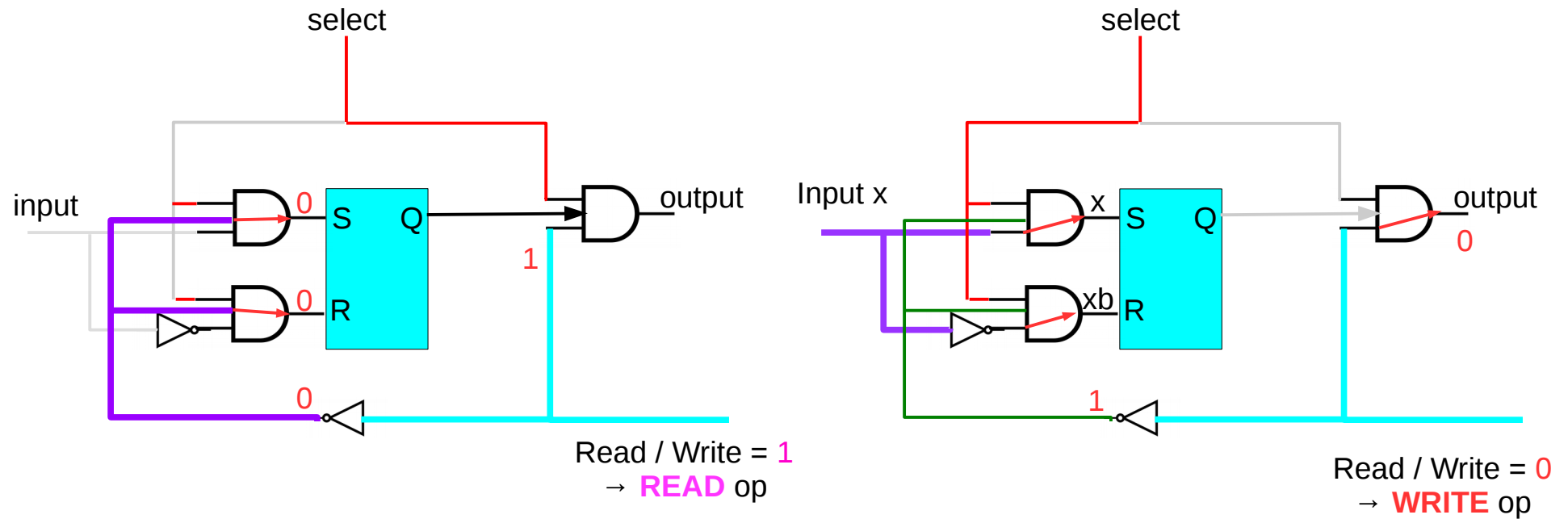
Selecting a Word



Bit Cell



Read & Write Mode of Operations



Q = old Q
 \bar{Q} = old \bar{Q}

Input x=1



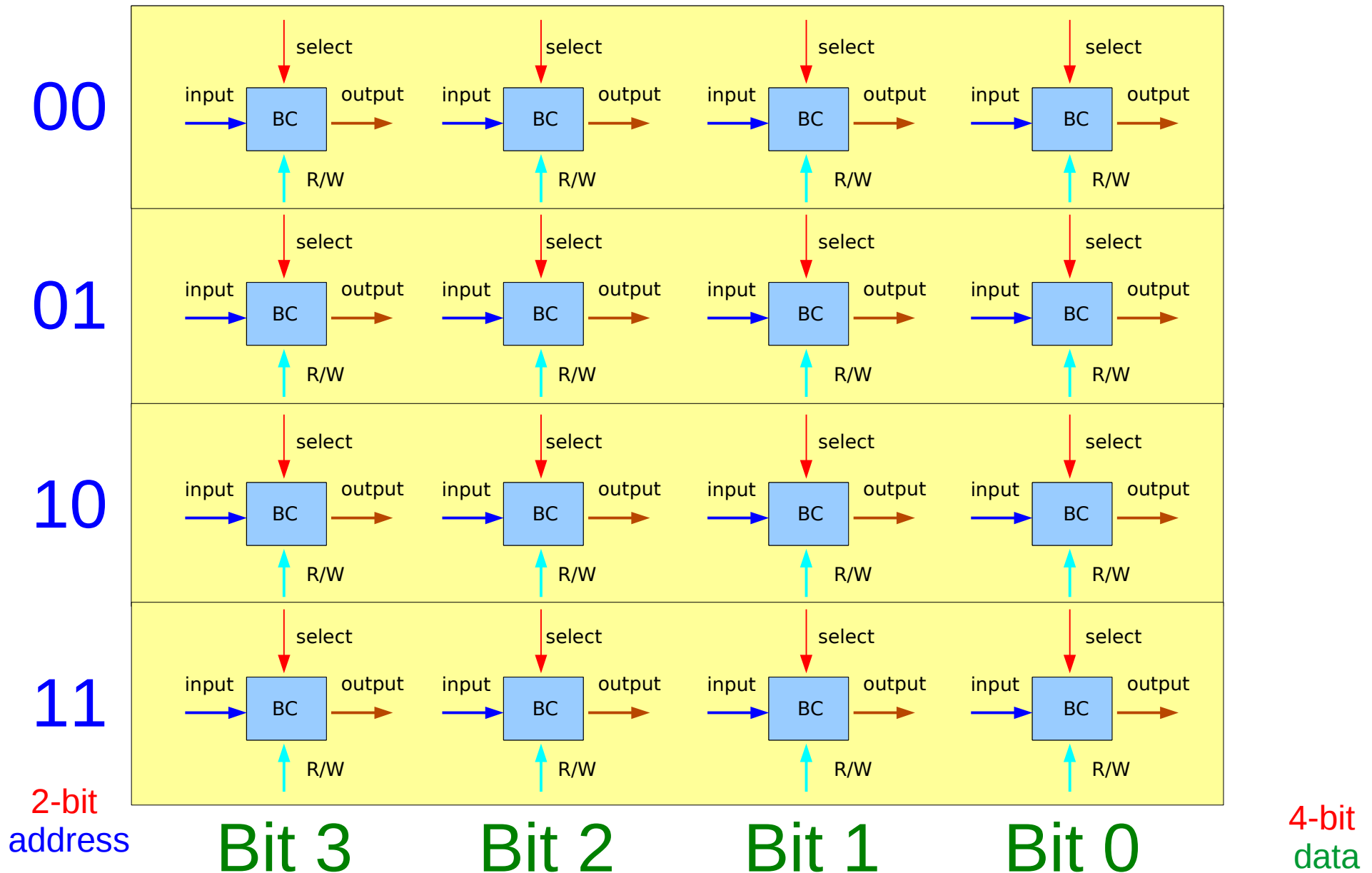
Q = 1
 \bar{Q} = 0

Input x=0



Q = 0
 \bar{Q} = 1

Memory Words and Addresses



4x4 Memory

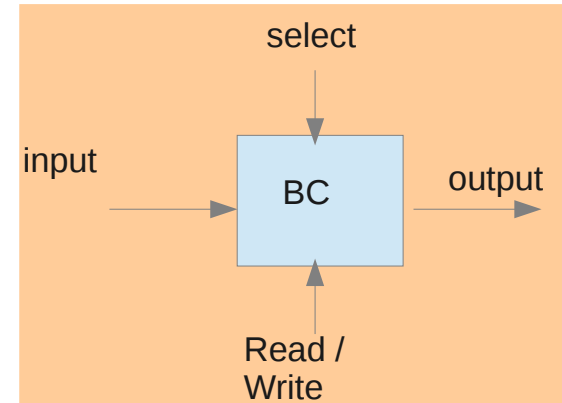
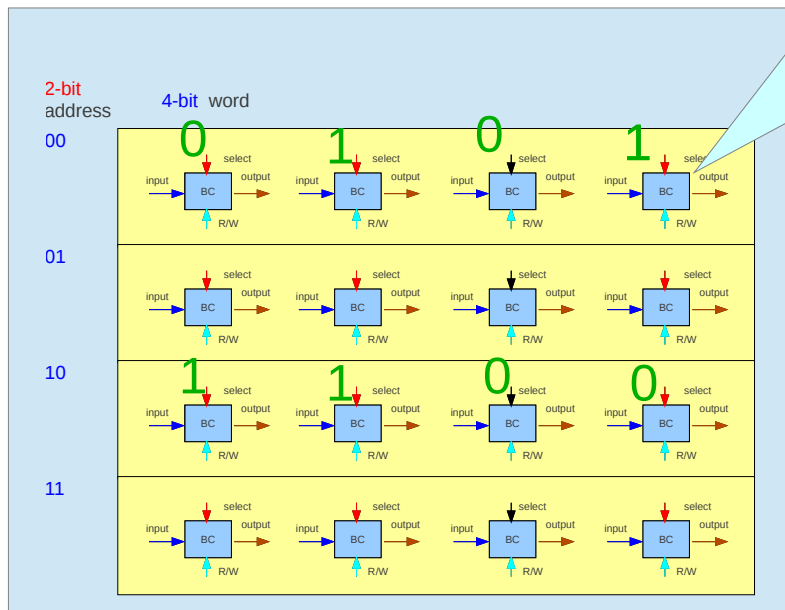
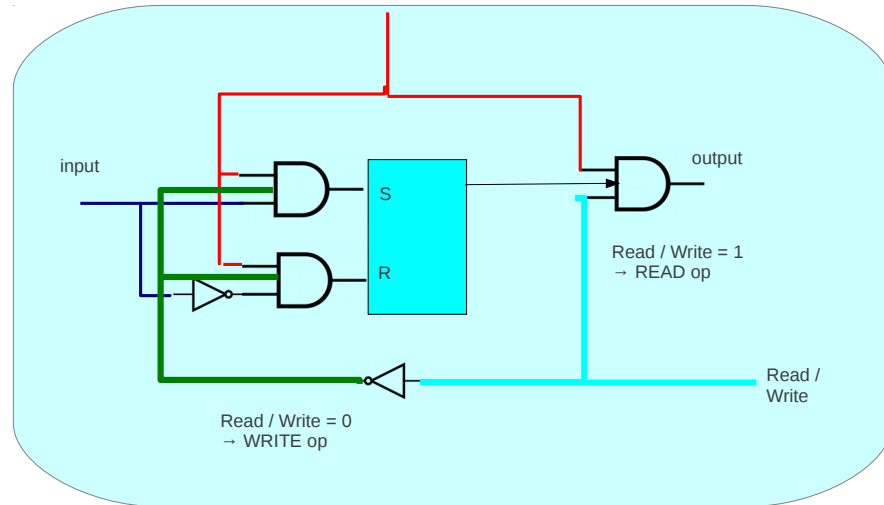
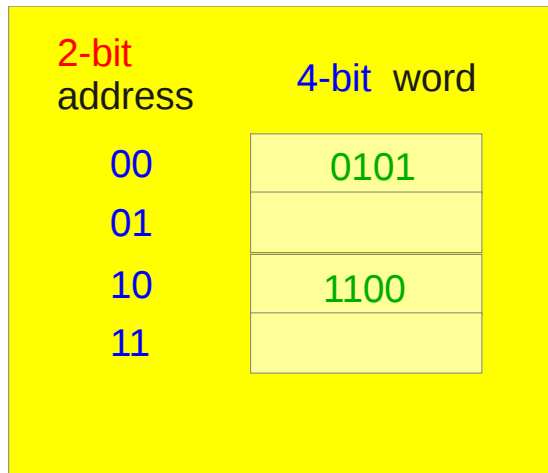
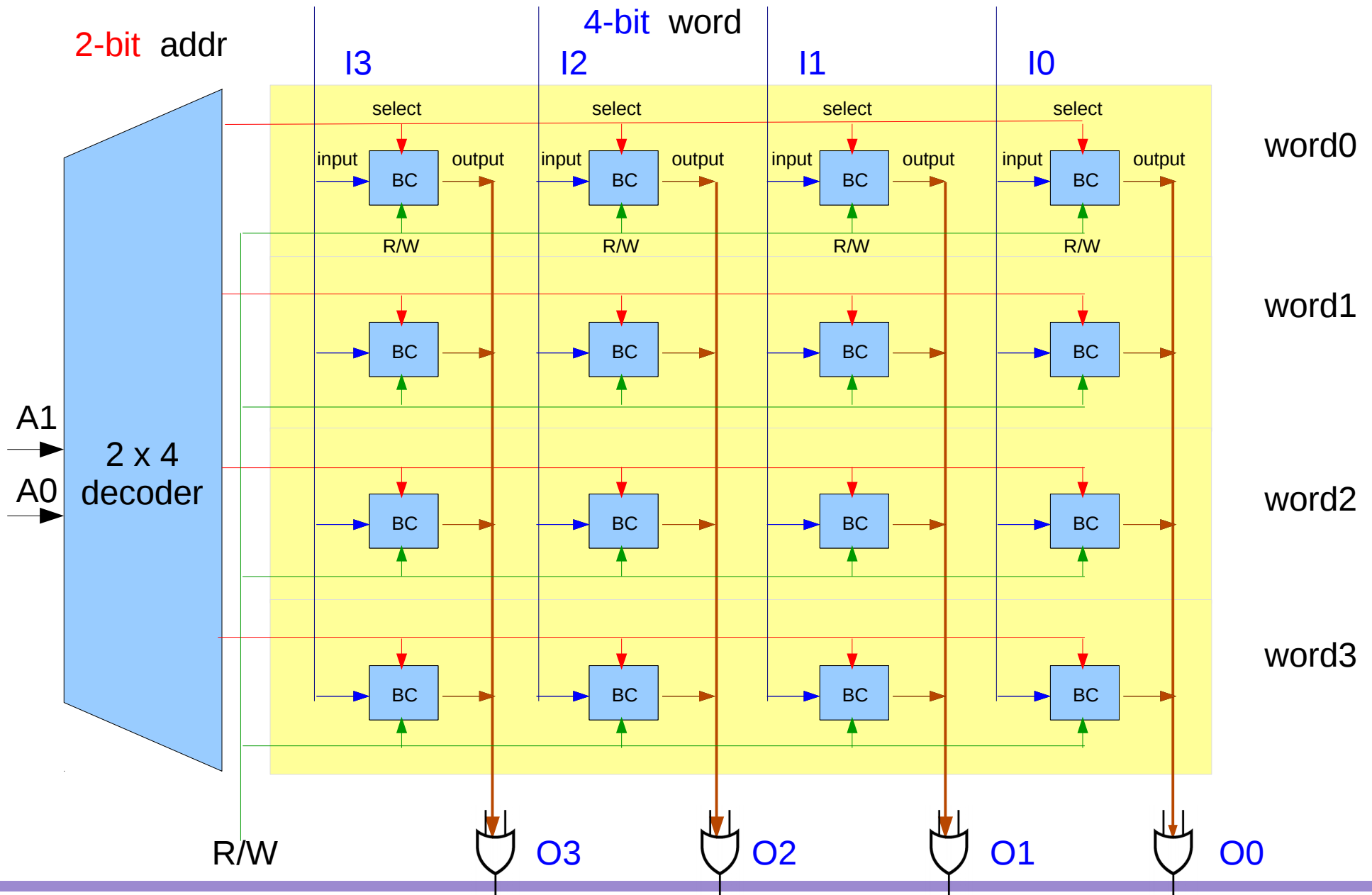


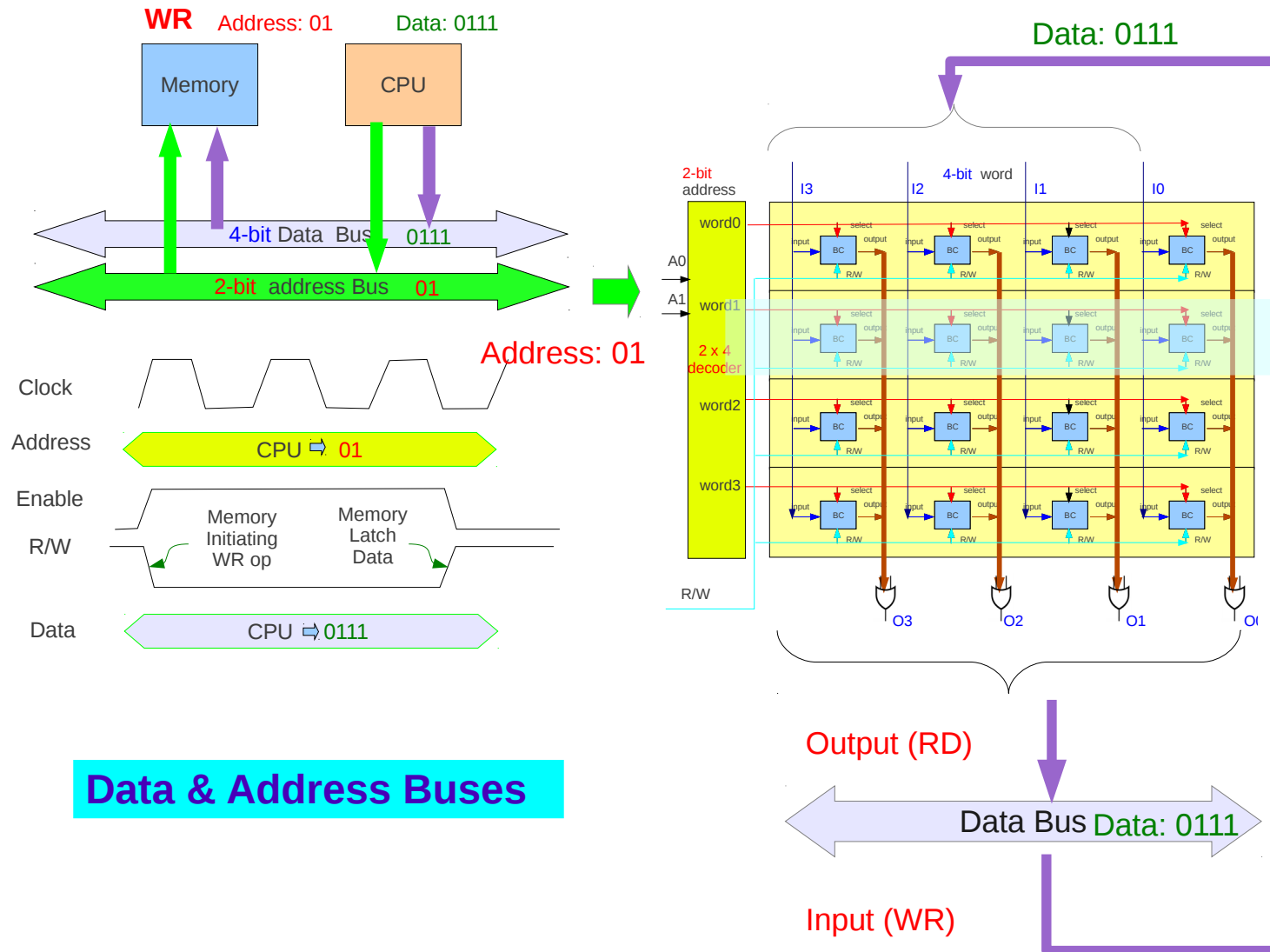
Diagram for a 4x4 Memory



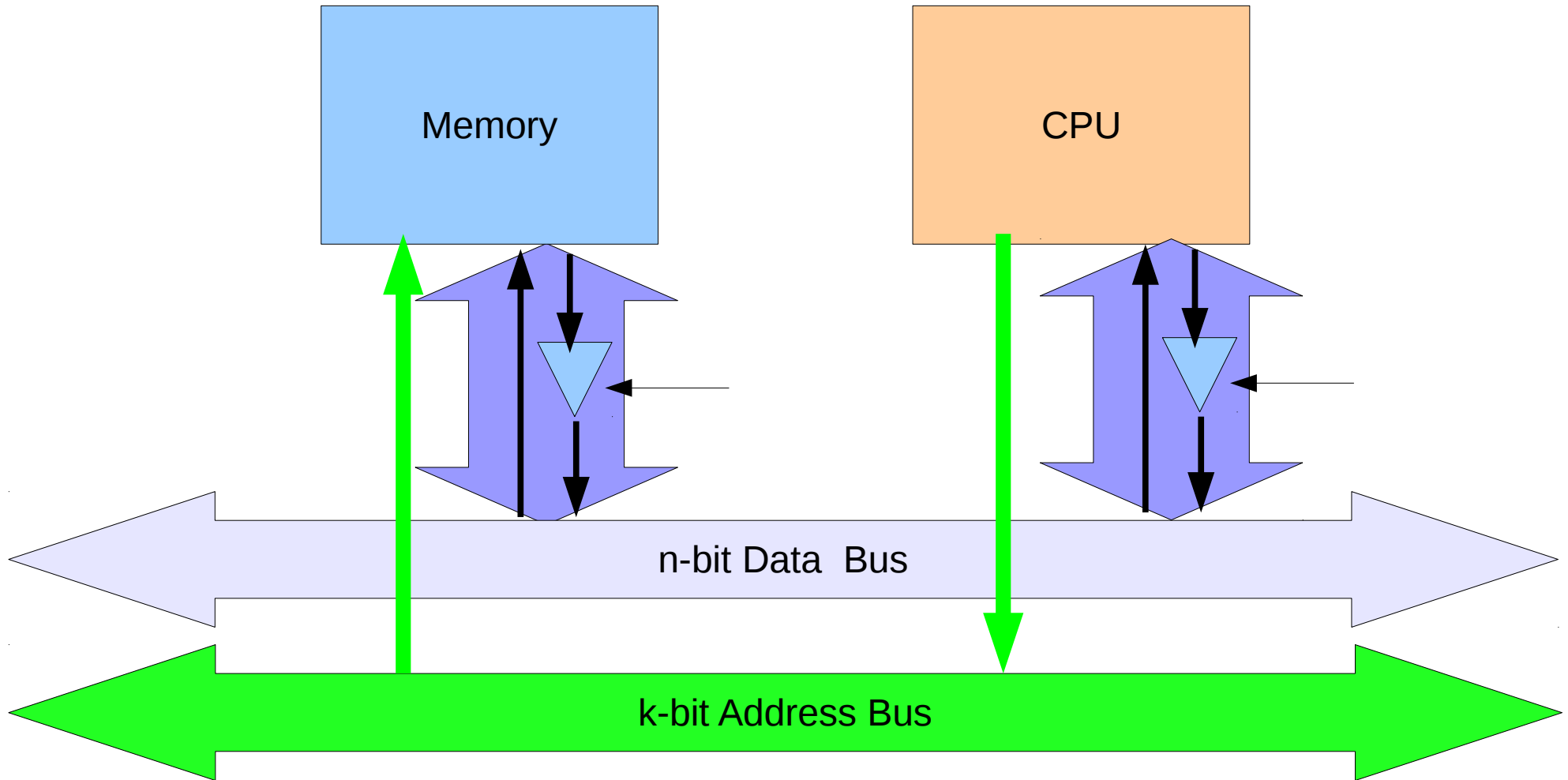
Memory

24

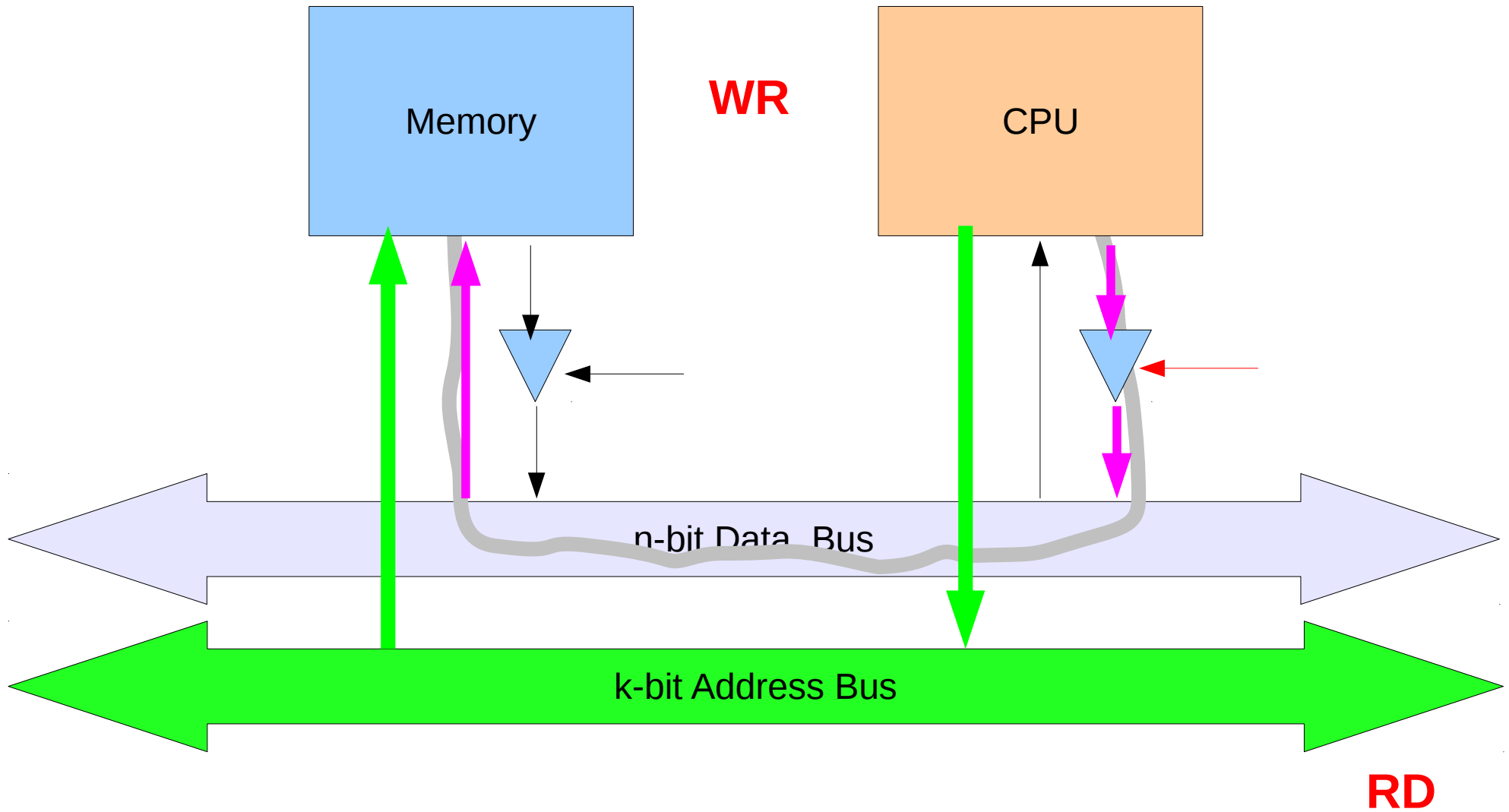
Write Cycle Example for a 4x4 Memory



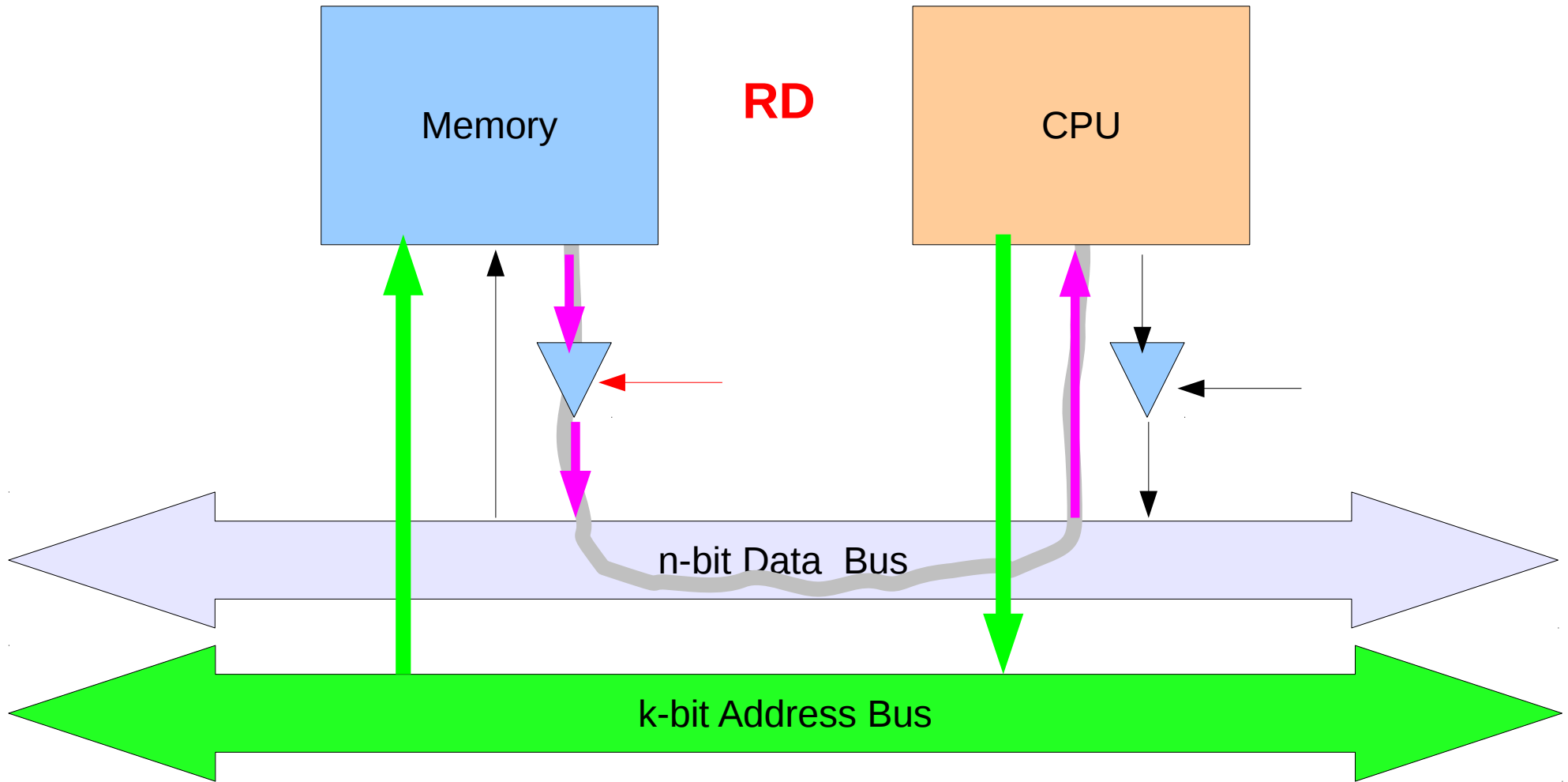
Bi-directional Data Bus



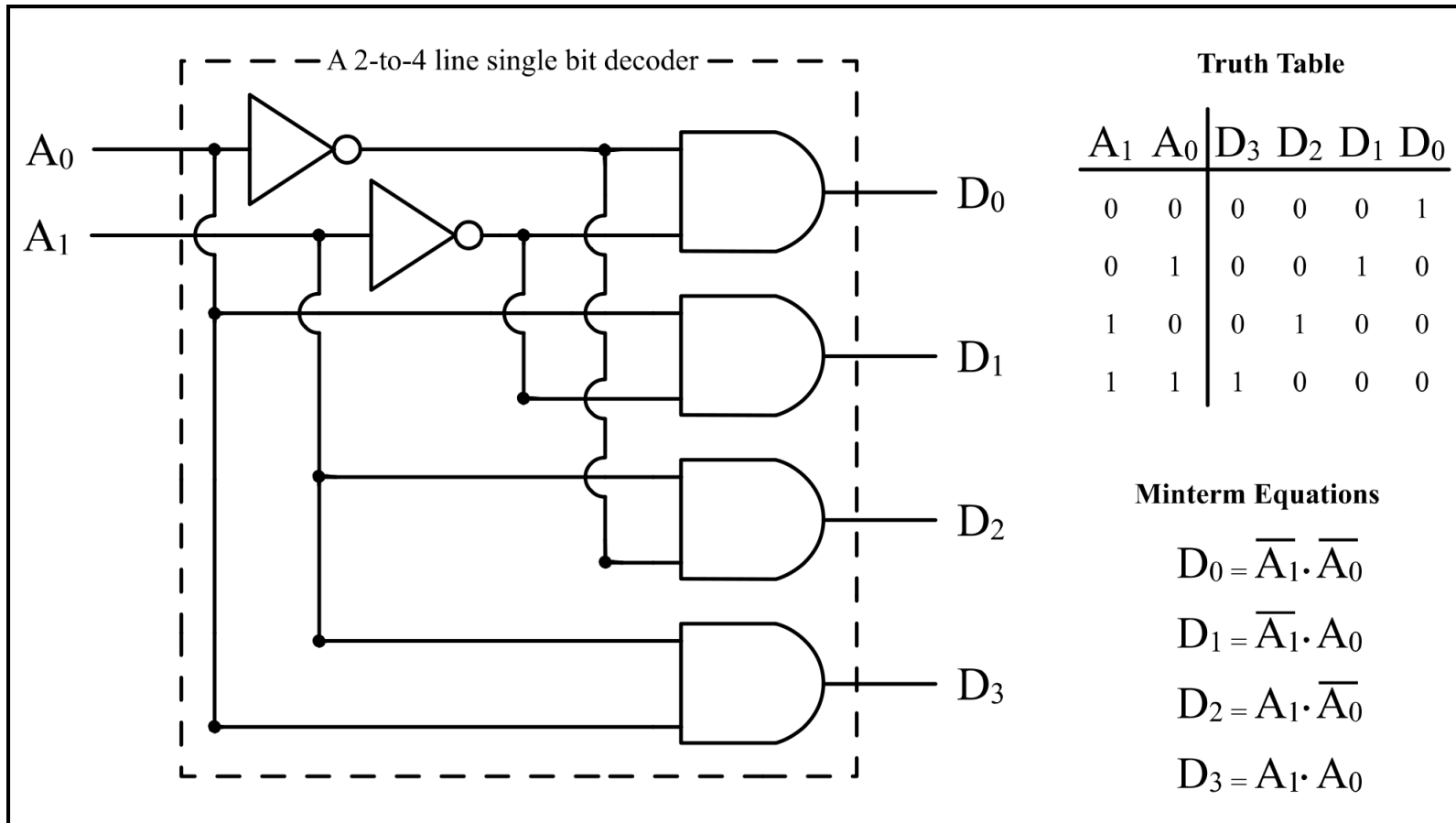
From CPU to Memory



From Memory to CPU



2 to 4 Decoder

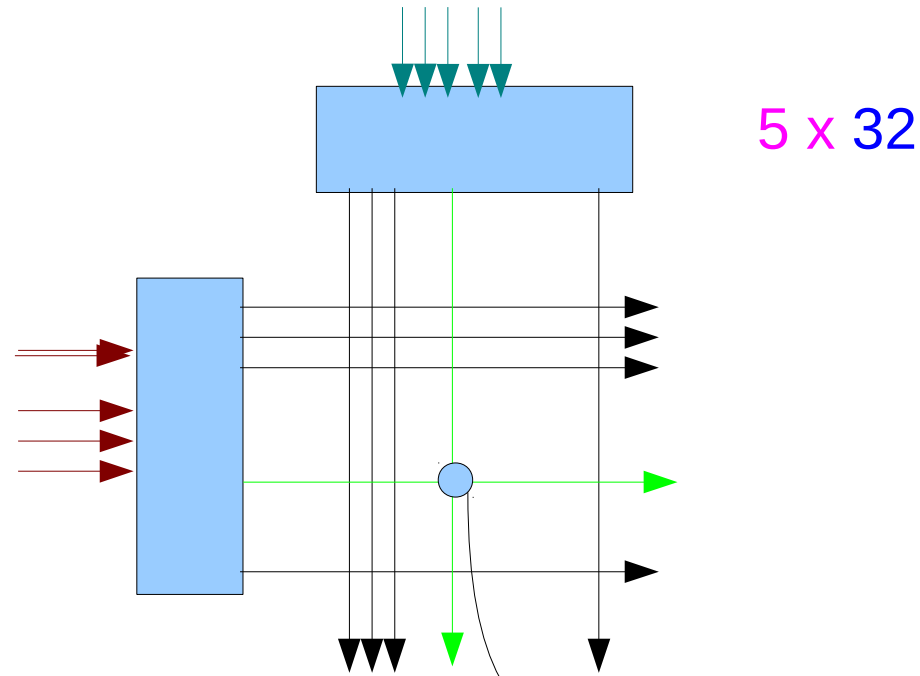
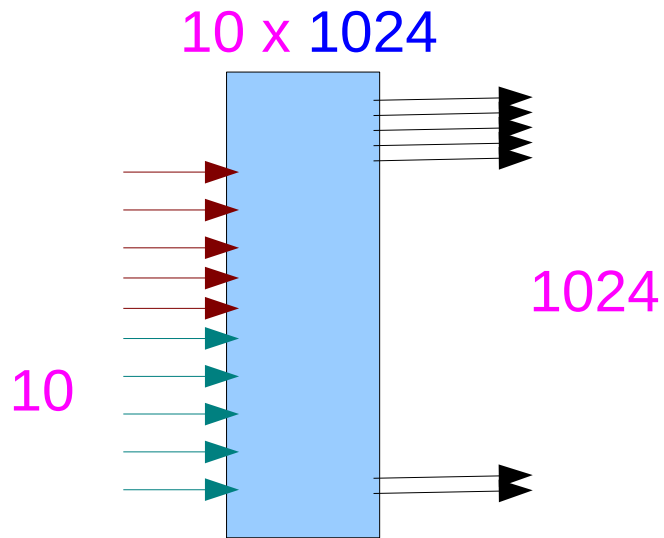
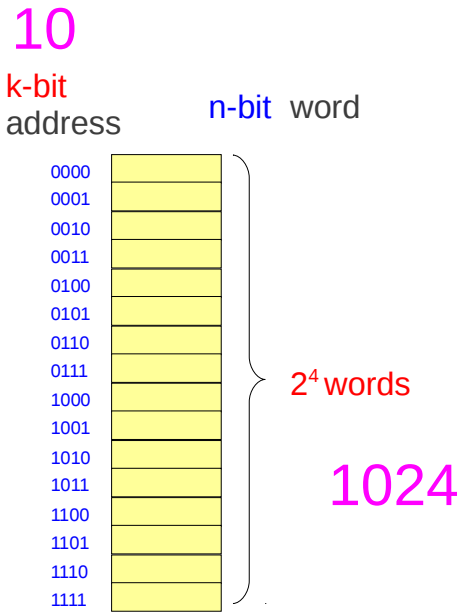


2 x 4 decoder : 4 AND gates

10 x 1024 decoder : 1024 AND gates

4 x 16 decoder : 16 AND gates

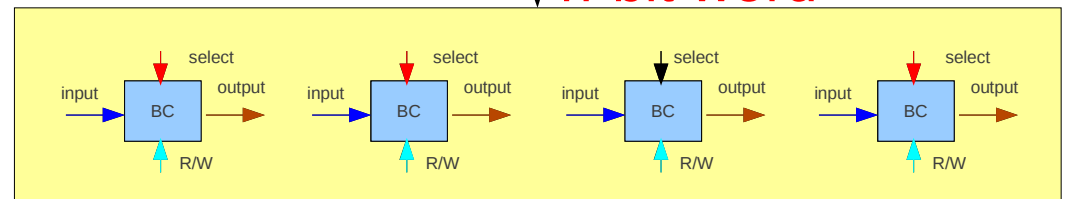
Coincidence Decoder



5 x 32

404 → 01100 10110

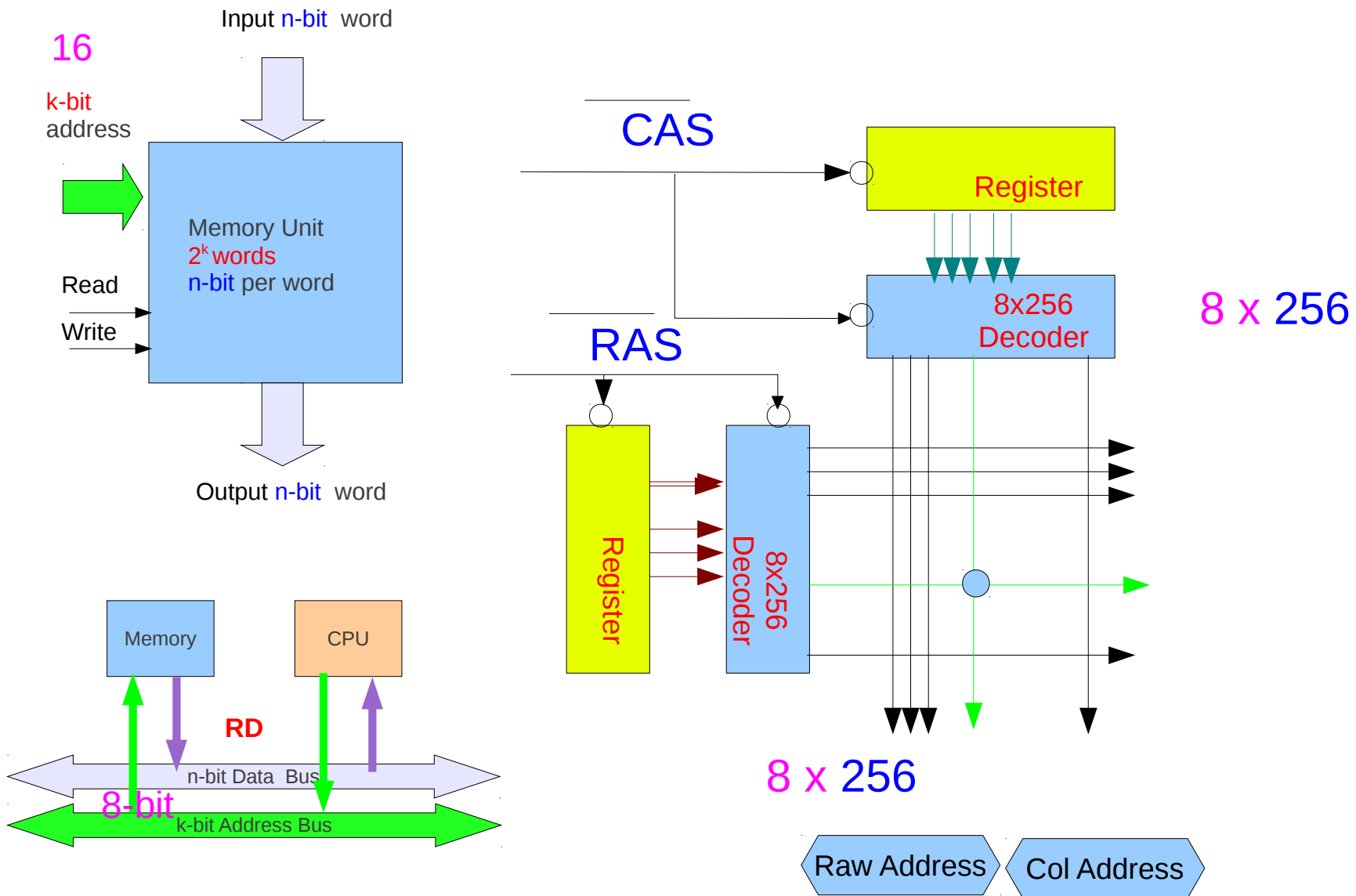
n-bit word



Memory

30

Time Sharing Address Bus



References

- [1] Essential C, Nick Parlante
- [2] Efficient C Programming, Mark A. Weiss
- [3] C A Reference Manual, Samuel P. Harbison & Guy L. Steele Jr.
- [4] C Language Express, I. K. Chun
- [5] https://en.wikiversity.org/wiki/The_necessities_in_SOC_Design
- [6] https://en.wikiversity.org/wiki/The_necessities_in_Digital_Design
- [7] https://en.wikiversity.org/wiki/The_necessities_in_Computer_Design
- [8] https://en.wikiversity.org/wiki/The_necessities_in_Computer_Architecture
- [9] https://en.wikiversity.org/wiki/The_necessities_in_Computer_Organization