Herbert Wertheim
College of Engineering
UNIVERSITY of FLORIDA

POWERING THE NEW ENGINEER TO TRANSFORM THE FUTURE

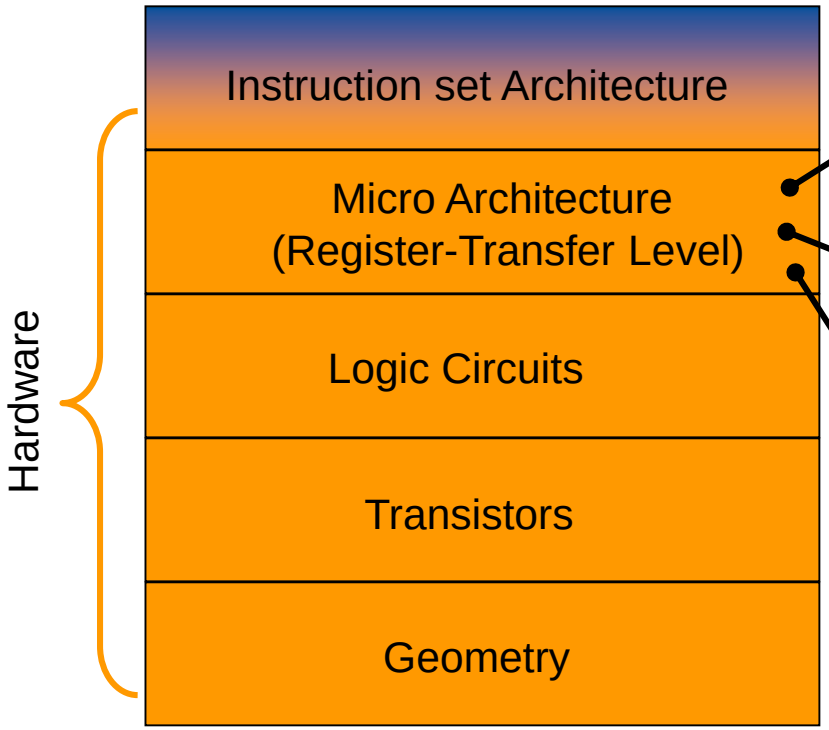
Department of Electrical & Computer Engineering

Digital Logic And Computing Systems

Chapter 05 – RTL Components Memory

Dr. Christophe Bobda

Register-Transfer Level (RTL)



Complex circuit components are made from simple elements

Information flows are represented, processed, transported and stored as words in registers.

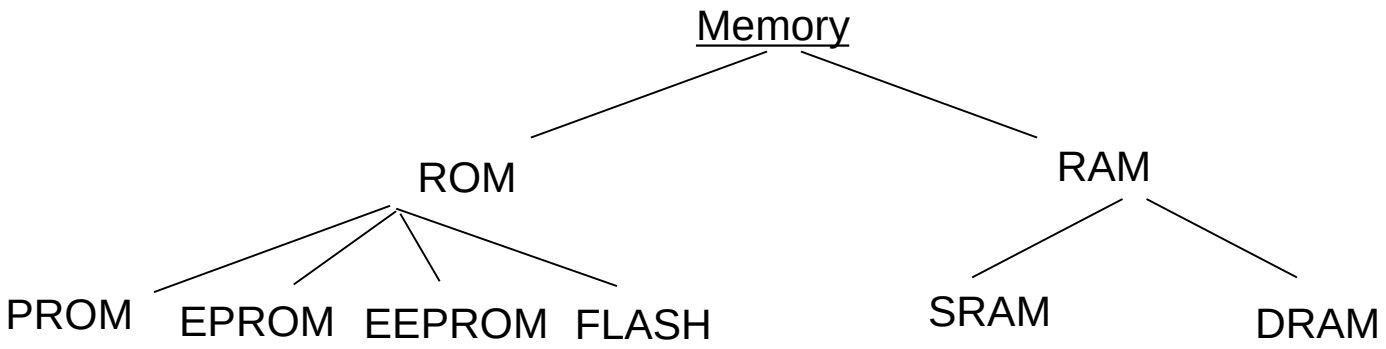
Information flow from register to register through combinational units:

→ Register-Transfer

Agenda

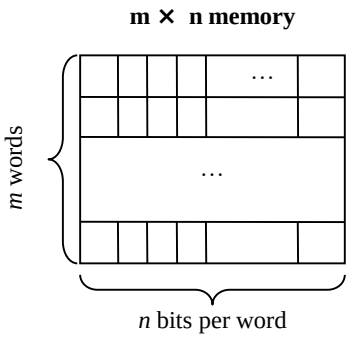
- Buses
- Word (bundle)gates
- Multiplexer / Demultiplexer
- Encoder / Decoder
- Memory
- Arithmetic Circuits

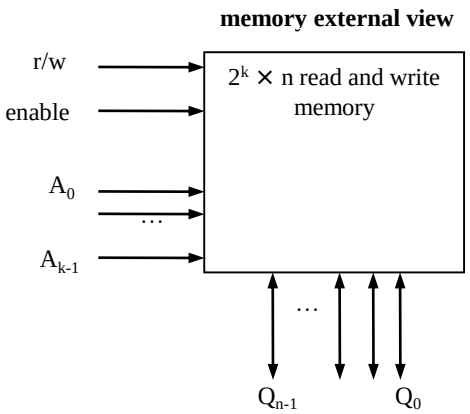
Memory



Memory: basic concepts

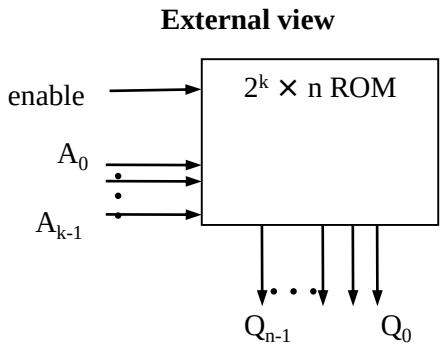
- Stores large number of bits
  - m x n: m words of n bits each
  - k = Log2(m) address input signals
  - or m = 2^k words
  - e.g., 4,096 x 8 memory:
    - 32,768 bits
    - 12 address input signals
    - 8 input/output data signals
  - Memory access
    - r/w: selects read or write
    - enable: read or write only when asserted
    - multiport: multiple accesses to different locations simultaneously



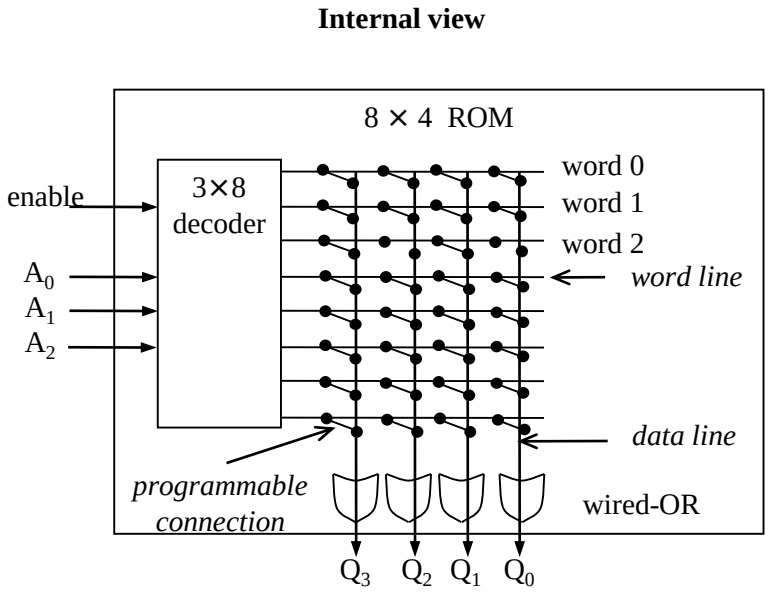


ROM: "Read-Only" Memory

- Nonvolatile memory
- Can be read from but not written to, by a processor in an embedded system
- Traditionally written to, "programmed", before inserting to embedded system
- Uses
  - Store software program for general-purpose processor
  - program instructions can be one or more ROM words
  - Store constant data needed by system
  - Implement combinational circuit



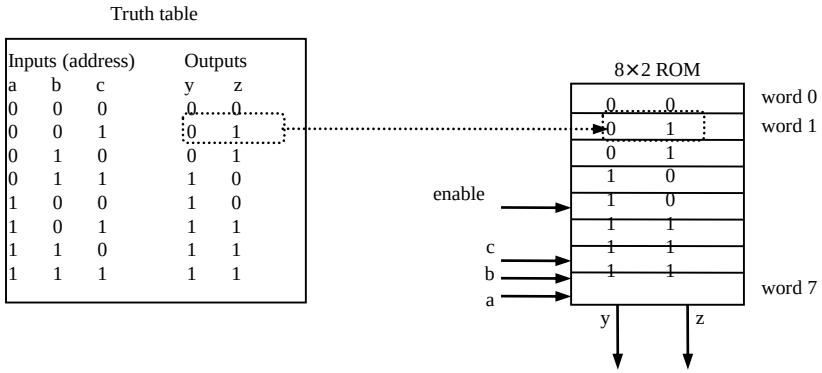
Example: 8 x 4 ROM



- Horizontal lines = words
- Vertical lines = data
- Lines connected only at circles
- Decoder sets word 2's line to 1 if address input is 010
- Data lines Q3 and Q1 are set to 1 because there is a "programmed" connection with word 2's line
- Word 2 is not connected with data lines Q2 and Q0
- Output is 1010

Implementing combinational function

- Any combinational circuit of n functions of same k variables can be done with 2^k x n ROM



Mask-programmed ROM

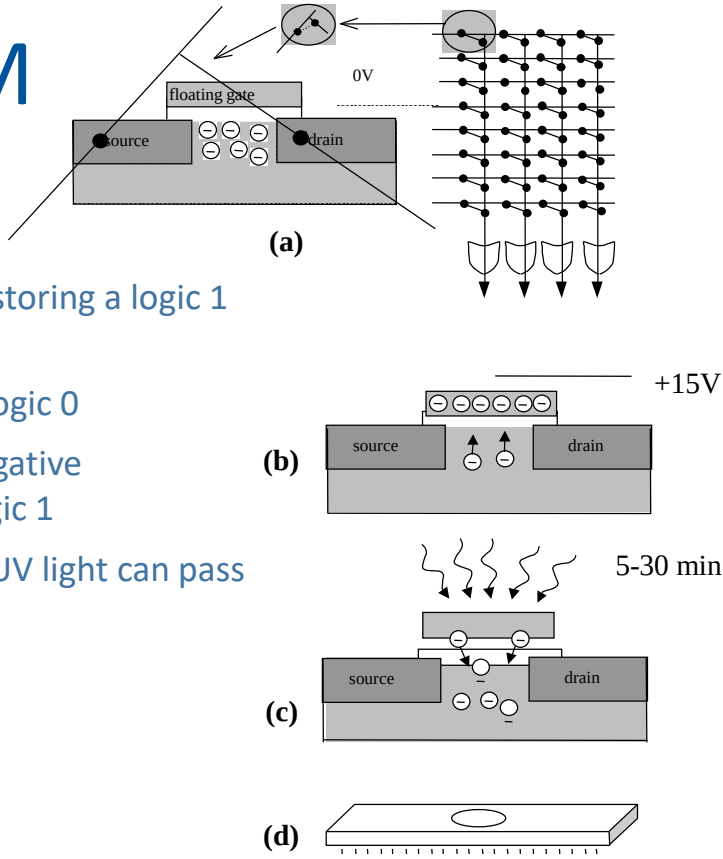
- Connections "programmed" at fabrication
  - set of masks
- Lowest write ability
  - only once
- Highest storage permanence
  - bits never change unless damaged
- Typically used for final design of high-volume systems
  - spread out NRE cost for a low unit cost

OTP ROM: One-time programmable ROM

- Connections "programmed" after manufacture by user
  - user provides file of desired contents of ROM
  - file input to machine called ROM programmer
  - each programmable connection is a fuse
  - ROM programmer blows fuses where connections should not exist
- Very low write ability
  - typically written only once and requires ROM programmer device
- Very high storage permanence
  - bits don't change unless reconnected to programmer and more fuses blown
- Commonly used in final products
  - cheaper, harder to inadvertently modify

EPROM: Erasable programmable ROM

- Programmable component is a MOS transistor
  - Transistor has "floating" gate surrounded by an insulator
  - (a) Negative charges form a channel between source and drain storing a logic 1
  - (b) Large positive voltage at gate causes negative charges to move out of channel and get trapped in floating gate storing a logic 0
  - (c) (Erase) Shining UV rays on surface of floating-gate causes negative charges to return to channel from floating gate restoring the logic 1
  - (d) An EPROM package showing quartz window through which UV light can pass
- Better write ability
  - can be erased and reprogrammed thousands of times
- Reduced storage permanence
  - program lasts about 10 years but is susceptible to radiation and electric noise
- Typically used during design development



EEPROM: Electrically erasable programmable ROM

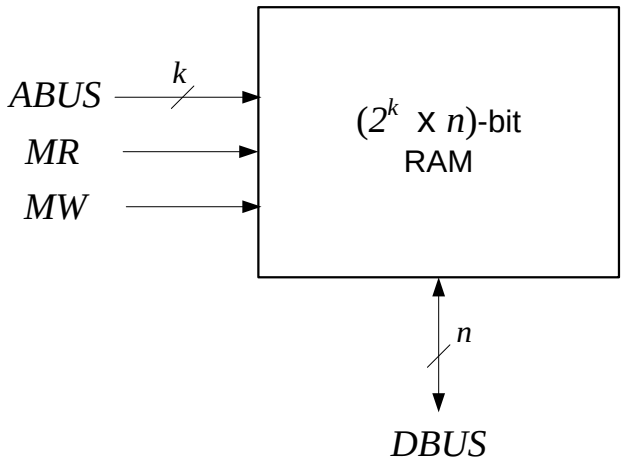
- Programmed and erased electronically
  - typically by using higher than normal voltage
  - can program and erase individual words
- Better write ability
  - can be in-system programmable with built-in circuit to provide higher than normal voltage
    - built-in memory controller commonly used to hide details from memory user
  - writes very slow due to erasing and programming
    - "busy" pin indicates to processor EEPROM still writing
  - can be erased and programmed tens of thousands of times
- Similar storage permanence to EPROM (about 10 years)
- Far more convenient than EPROMs, but more expensive

Flash Memory

- Fast erase
  - Large blocks of memory erased at once, rather than one word at a time
  - Blocks typically several thousand bytes large
- Writes to single words may be slower
  - Entire block must be read, word updated, then entire block written back
- Used with embedded systems storing large data items in nonvolatile memory
  - e.g., digital cameras, TV set-top boxes, cell phones
- Extension of EEPROM
  - Same floating gate principle
  - Same write ability and storage permanence

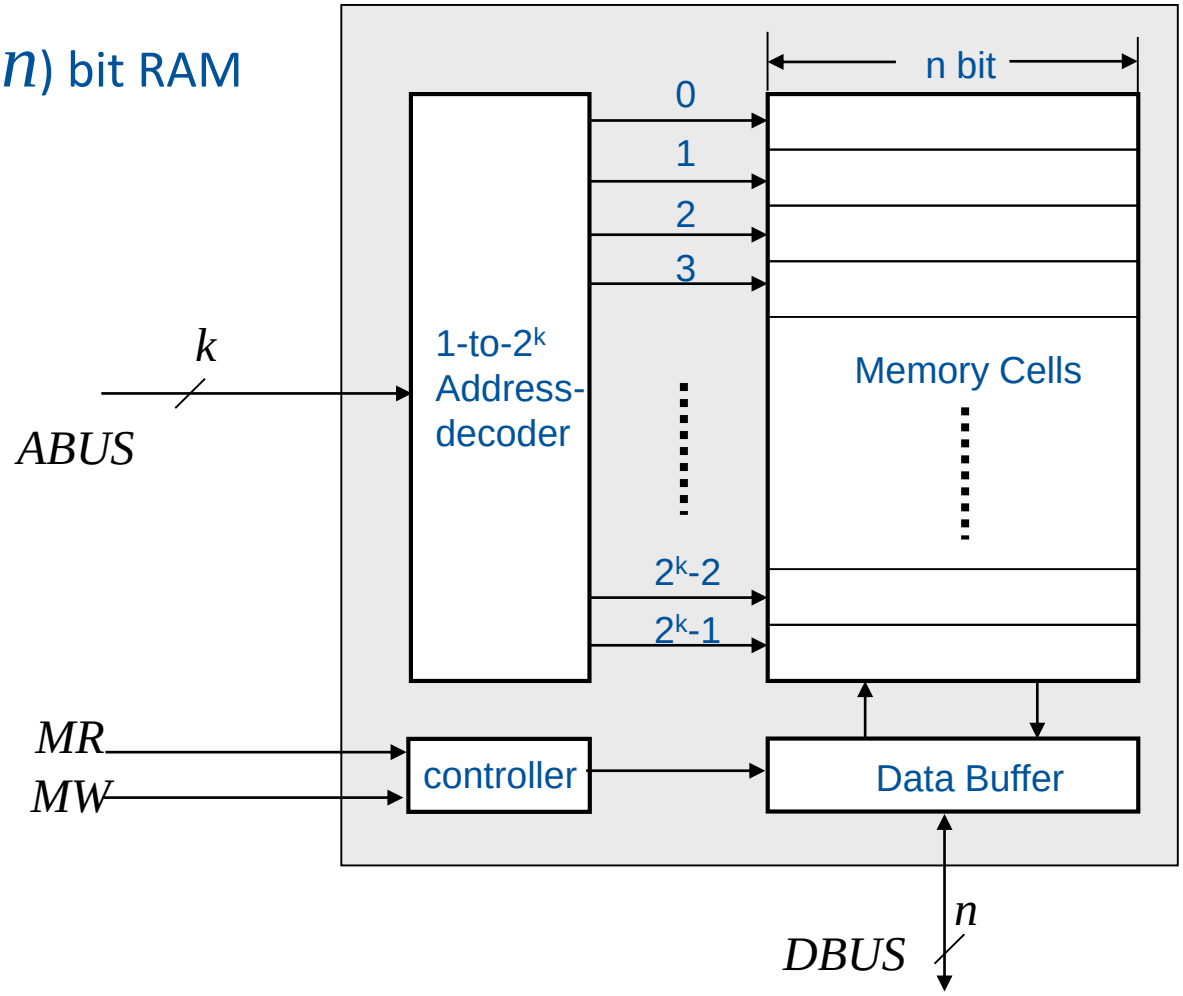
RAM

- RAM (write-read memory) can be randomly written and randomly read
- Inputs and Outputs
  - k bit address bus $ABUS$
  - n bit data bus $DBUS$
  - control signals MR (memory read) and MW (memory write)
    - $MR=1$ means that value at address location defined by ABUS must be place on the bus (read)
    - $MW=1$ means that the value on the bus DBUS must be stored at the location defined by ABUS (write)

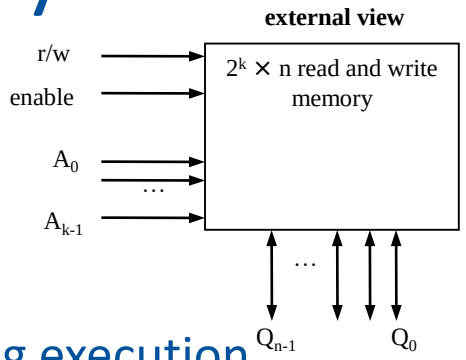


RAM

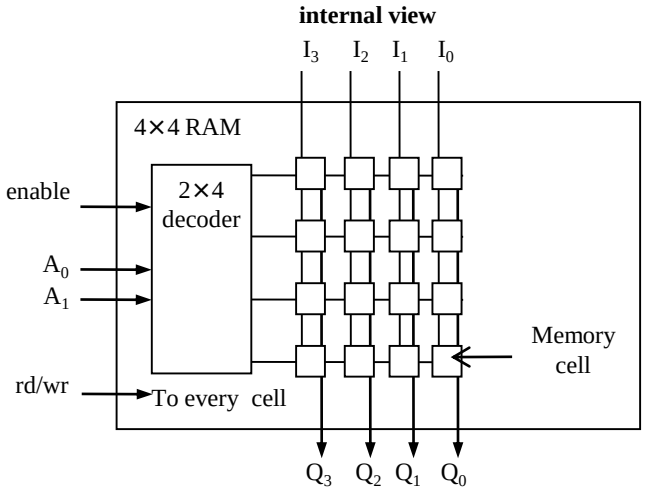
- Architecture of a (2^k x n) bit RAM



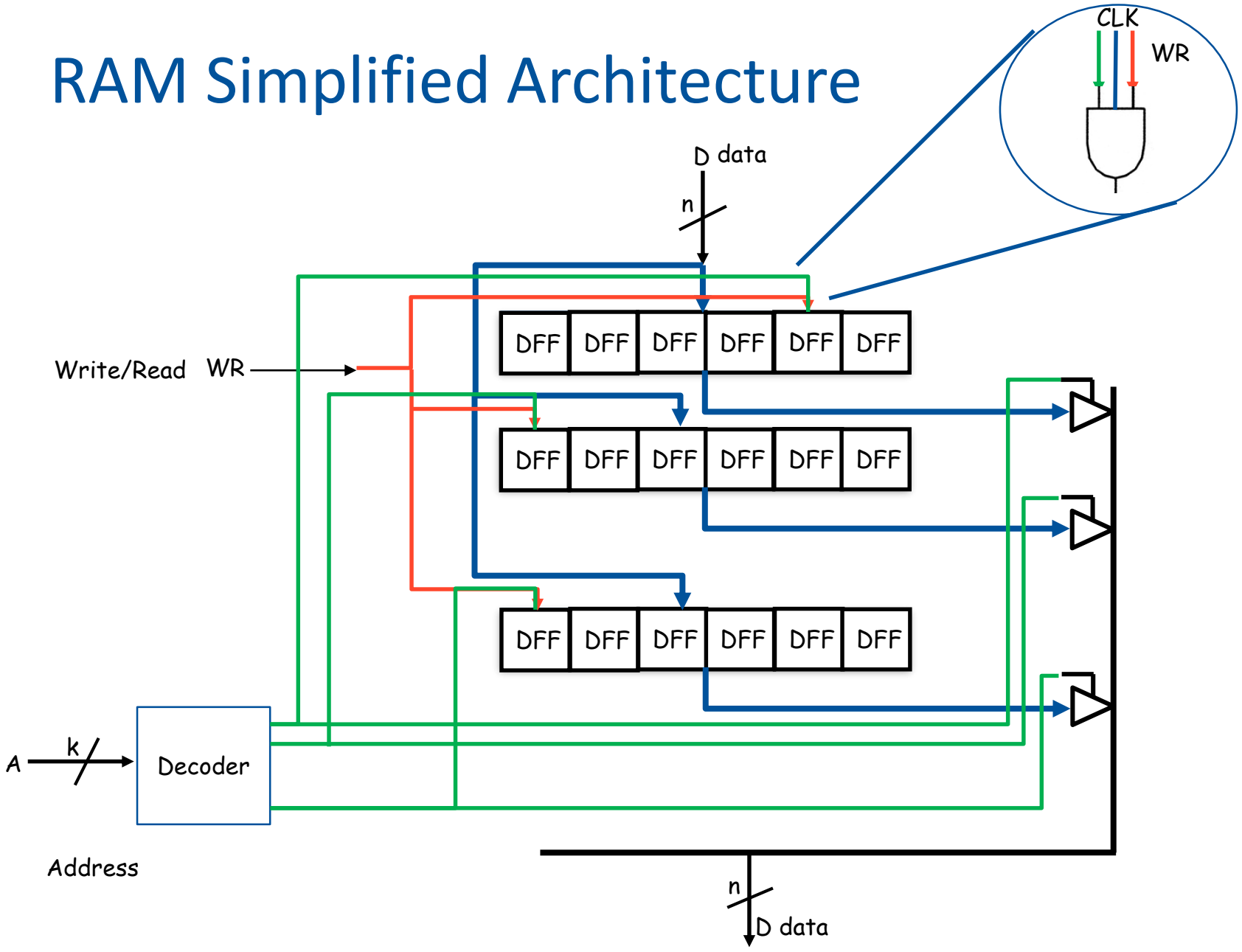
RAM: "Random-access" memory



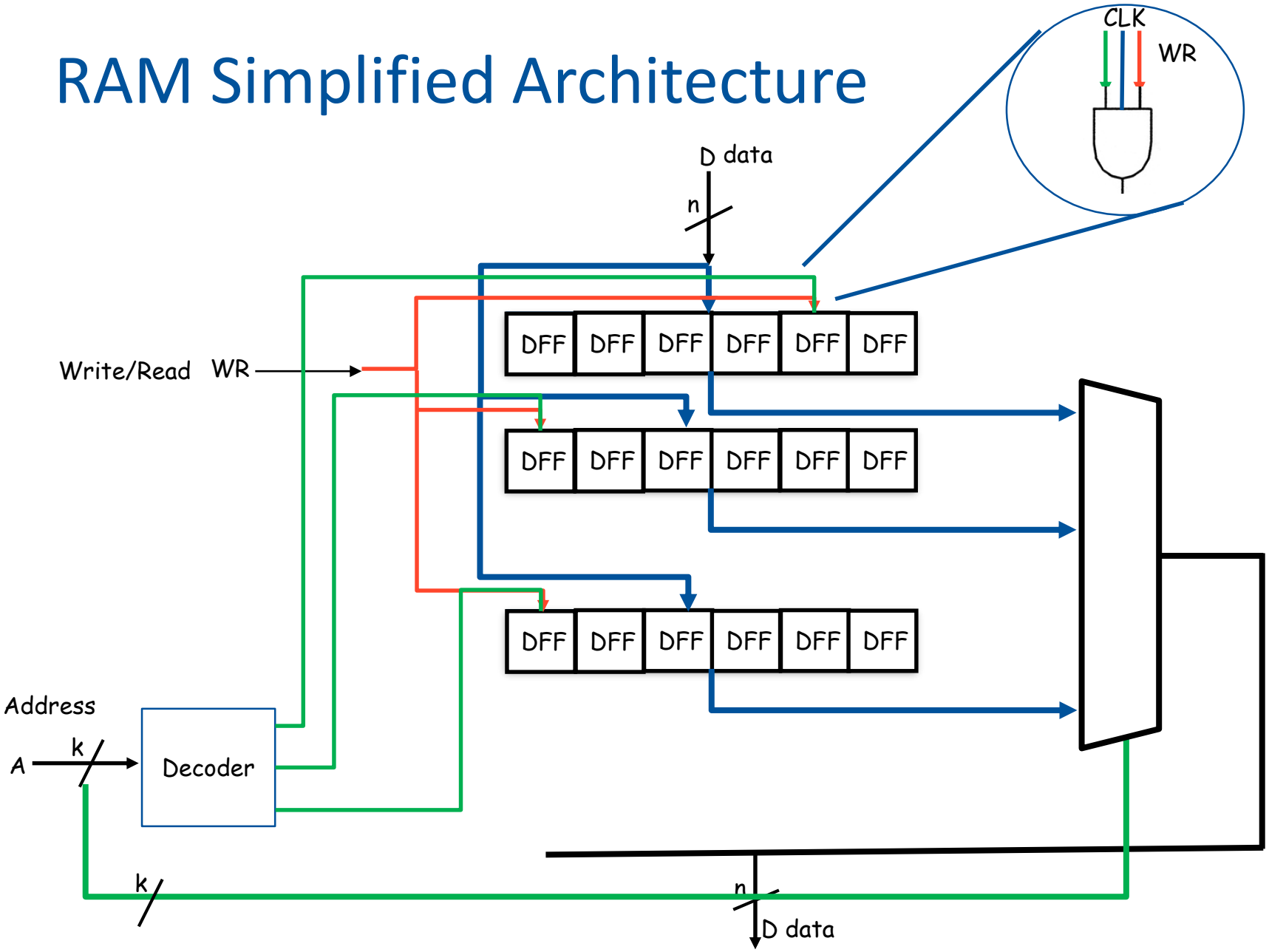
- Typically volatile memory
  - bits are not held without power supply
- Read and written to easily by embedded system during execution
- Internal structure more complex than ROM
  - a word consists of several memory cells, each storing 1 bit
  - each input and output data line connects to each cell in its column
  - rd/wr connected to every cell
  - when row is enabled by decoder, each cell has logic that stores input data bit when rd/wr indicates write or outputs stored bit when rd/wr indicates read



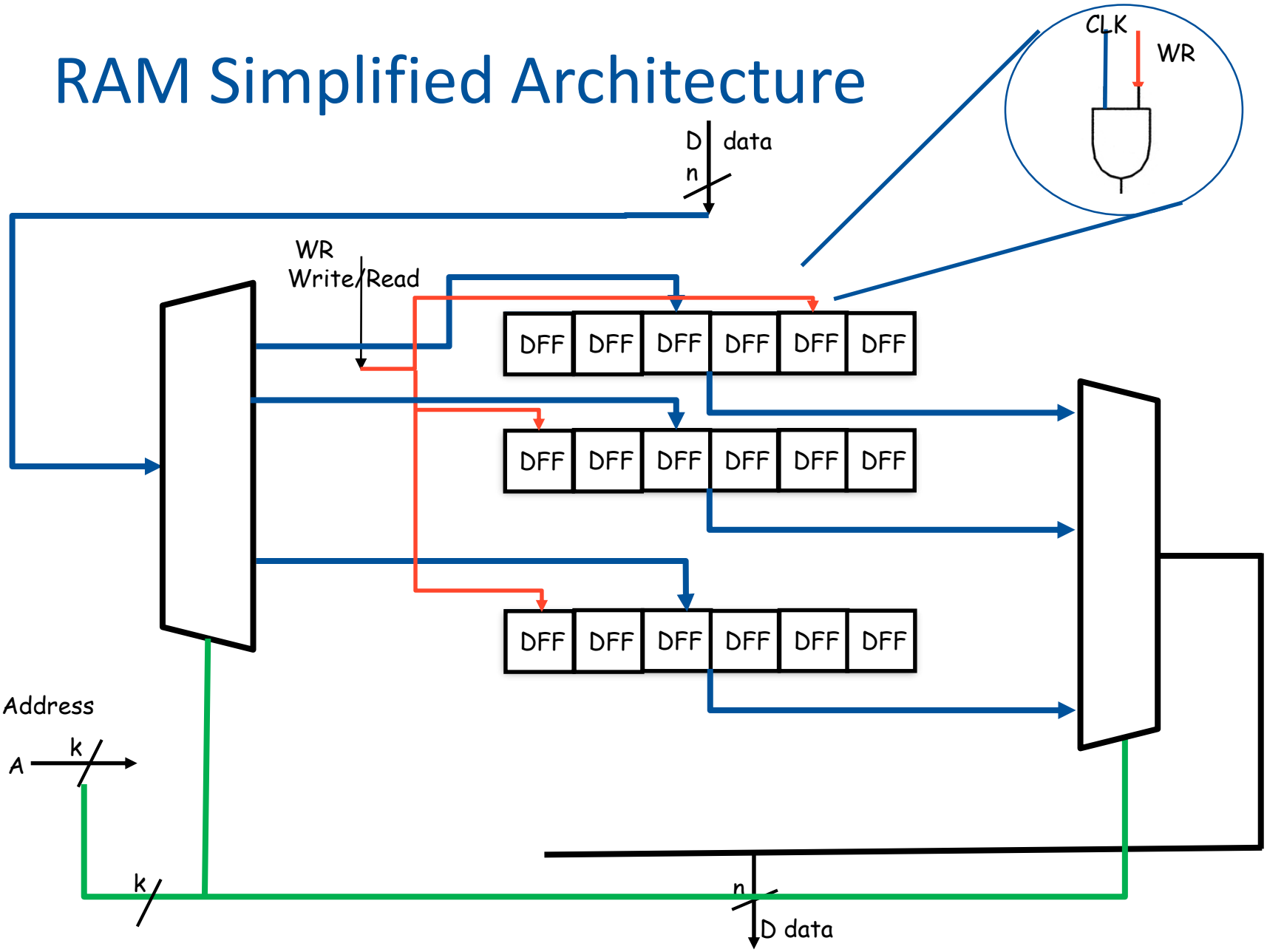
RAM Simplified Architecture



RAM Simplified Architecture



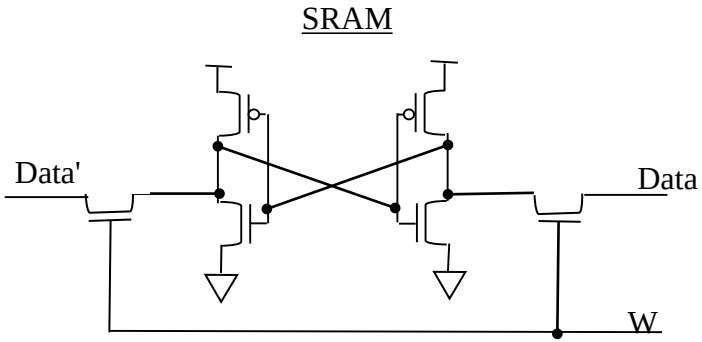
RAM Simplified Architecture



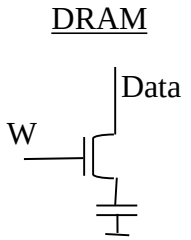
Basic types of RAM

memory cell internals

- SRAM: Static RAM
  - Memory cell uses flip-flop to store bit
  - Requires 6 transistors
  - Holds data as long as power supplied



- DRAM: Dynamic RAM
  - Memory cell uses MOS transistor and capacitor to store bit
  - More compact than SRAM
  - "Refresh" required due to capacitor leak
  - word's cells refreshed when read
  - Typical refresh rate 15.625 microsec.
  - Slower to access than SRAM



RAM

- Important properties
  - Capacity: Number of bits
  - Organization: number of address bits k, number of data bits n
  - Access Time: time to read a word from or write a word in the memory

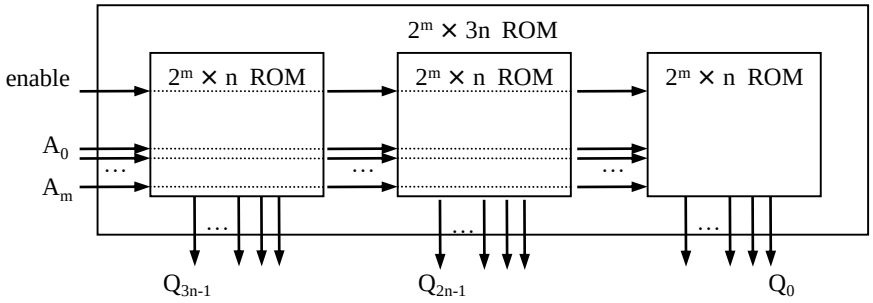
- 2 Types of RAM: SRAM und DRAM.
  - SRAMs (static RAM) have smaller capacity than DRAMs and more expensive, but better access time. The memory content is available for as longer as power is on

Composing memory

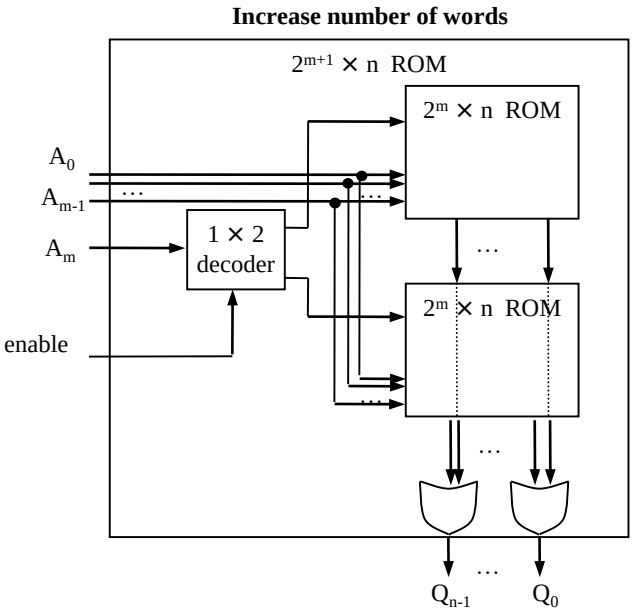
- Memory size needed often differs from size of readily available memories
- When available memory is larger, simply ignore unneeded high-order address bits and higher data lines
- When available memory is smaller, compose several smaller memories into one larger memory
- Connect side-by-side to increase width of words
- Connect top to bottom to increase number of words
- added high-order address line selects smaller memory containing desired word using a decoder
- Combine techniques to increase number and width of words

Composing memory
Parallel Composition

Increase width of words

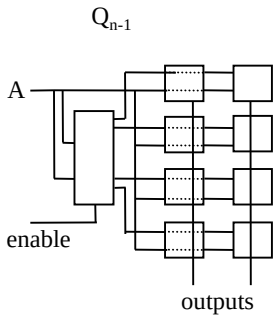


Composing memory
Serial composition



Composing memory
Hybrid Composition

Increase number and width of words



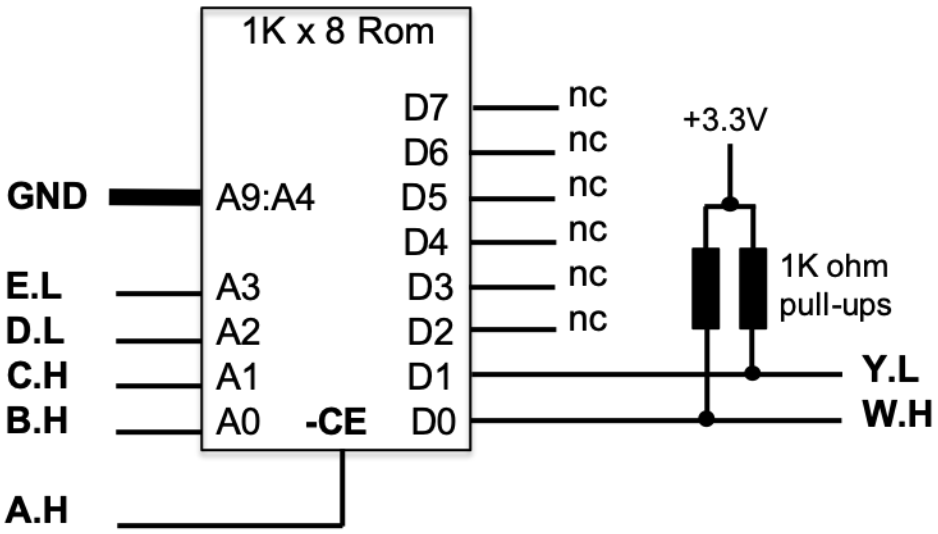
Composing memory

- Briefly define each of the following: mask-programmed ROM, PROM, EPROM, EEPROM, flash EEPROM, RAM, SRAM, DRAM, PSRAM, NVRAM.
- Sketch the internal design of a 4x3 ROM.
- Sketch the internal design of a 4x3 RAM
- Compose 1kx8 ROM's into a 1kx32 ROM (note: 1k actually means 1028 words).
- Compose 1kx8 ROM's into an 8kx8 ROM.
- Compose 1kx8 ROM's into a 2kx16 ROM.
- Show how to use a 1kx8 ROM to implement a 512x6 ROM.

System Implementation with Memory

Note1: -CE = low true chip enable
If –CE = F then data lines tri-state

Note2: **nc** = no connect



ROM Contents

Addr	Data	Addr	Data
0	1	8	1
1	2	9	2
2	1	A	1
3	3	B	3
4	2	C	2
5	3	D	3
6	2	E	2
7	0	F	0

System Impl

1A. Fill in the **Voltage Table** below on the left and the **Logic Truth Table** below on the right. Use '**X**' (don't care) below to reduce the number of table rows. (50 pts.)

Voltage Table

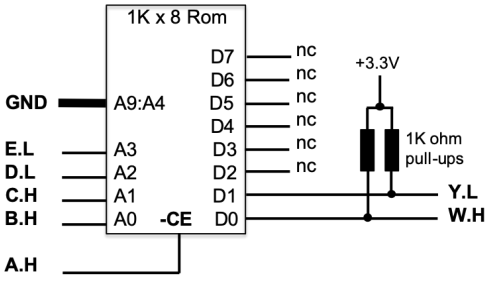
E	D	C	B	A	Y	W

Logic Truth Table

E	D	C	B	A	Y	W

Note1: -CE = low true chip enable
If –CE = F then data lines tri-state

Note2: **nc** = no connect



ROM Contents

Addr	Data	Addr	Data
0	1	8	1
1	2	9	2
2	1	A	1
3	3	B	3
4	2	C	2
5	3	D	3
6	2	E	2
7	0	F	0

1A. Fill in the **Voltage Table** below on the left and the **Logic Truth Table** below on the right.

Use '**X**' (don't care) below to reduce the number of table rows. (50 pts.)

Voltage Table

A3	A2	A1	A0	CE	D1	D0
X	X	X	X	1	Z	Z
0	0	0	0	0	0	1
0	0	0	1	0	1	0
0	0	1	0	0	0	1
0	0	1	1	0	1	1
0	1	0	0	0	1	0
0	1	0	1	0	1	1
0	1	1	0	0	1	0
0	1	1	1	0	0	0
1	0	0	0	0	0	1
1	0	0	1	0	1	0
1	0	1	0	0	0	1
1	0	1	1	0	1	1
1	1	0	0	0	1	0
1	1	0	1	0	1	1
1	1	1	0	0	1	0
1	1	1	1	0	0	0

Logic Truth Table

E	D	C	B	A	Y	W
X	X	X	X	1	Z	Z
1	1	0	0	0	1	1
1	1	0	1	0	0	0
1	1	1	0	0	1	1
1	1	1	1	0	0	1
1	0	0	0	0	0	0
1	0	0	1	0	0	1
1	0	1	0	0	0	0
1	0	1	1	0	1	0
0	1	0	0	0	1	1
0	1	0	1	0	0	0
0	1	1	0	0	1	1
0	1	1	1	0	0	1
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	0	0
0	0	1	1	0	1	0

ROM Contents

Addr	Data	Addr	Data
0	1	8	1
1	2	9	2
2	1	A	1
3	3	B	3
4	2	C	2
5	3	D	3
6	2	E	2
7	0	F	0

ROM Contents

Addr	Data	Addr	Data
0	1	8	1
1	2	9	2
2	1	A	1
3	3	B	3
4	2	C	2
5	3	D	3
6	2	E	2
7	0	F	0

1A. Fill in the **Voltage Table** below on the left and the **Logic Truth Table** below on the right.

Use '**X**' (don't care) below to reduce the number of table rows. (50 pts.)

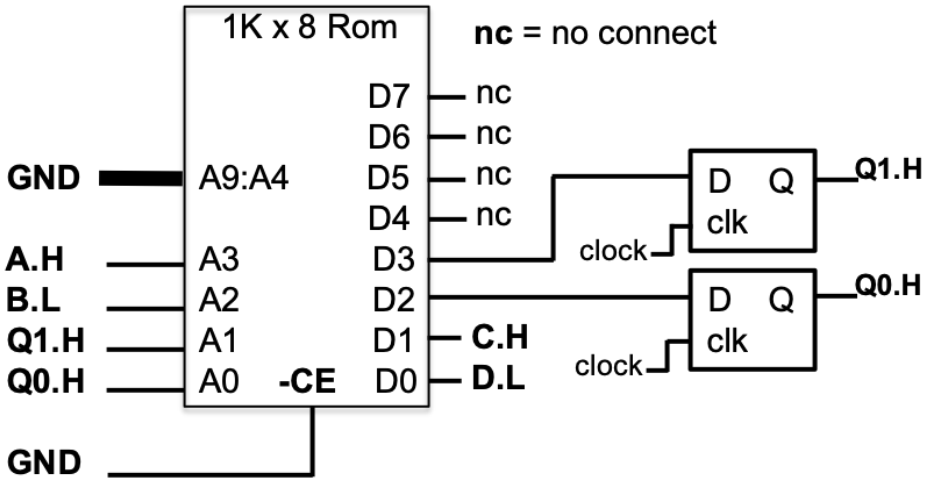
Voltage Table

E	D	C	B	A	Y	W
X	X	X	X	H	Z	Z
L	L	L	L	L	L	H
L	L	L	H	L	H	L
L	L	H	L	L	L	H
L	L	H	H	L	H	l
L	H	L	L	L	H	L
L	H	L	H	L	H	H
L	H	H	L	L	H	L
L	H	H	H	L	L	L
H	L	H	L	L	L	H
H	L	L	H	L	H	L
H	L	H	L	L	L	H
H	L	H	H	L	H	H
H	H	L	L	L	H	L
H	H	L	H	L	H	L
H	H	H	L	L	H	L
H	H	H	H	L	L	L

Logic Truth Table

E	D	C	B	A	Y	W
X	X	X	X	1	Z	Z
1	1	0	0	0	1	1
1	1	0	1	0	0	0
1	1	1	0	0	1	1
1	1	1	1	0	0	1
1	0	0	0	0	0	0
1	0	0	1	0	0	1
1	0	1	0	0	0	0
1	0	1	1	0	1	0
0	1	0	0	0	1	1
0	1	0	1	0	0	0
0	1	1	0	0	1	1
0	1	1	1	0	0	1
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	0	0
0	0	1	1	0	1	0

System Implementation with Memory



ROM Contents

Addr	Data	Addr	Data
0	5	8	0
1	4	9	8
2	7	A	7
3	D	B	D
4	5	C	0
5	4	D	2
6	7	E	7
7	D	F	D

2A. Fill in the **Next State Voltage Table** (below) and the **Next State Logic Truth Table** (below) for the design above. (40 points)

Next State Voltage Table

A	B	Q1	Q0	Q1+	Q0+	C	D

Next State Logic Truth Table

A	B	Q1	Q0	Q1+	Q0+	C	D

UF Herbert Wertheim College of Engineering
UNIVERSITY of FLORIDA

POWERING THE NEW ENGINEER TO TRANSFORM THE FUTURE