COMP2611: Computer Organization

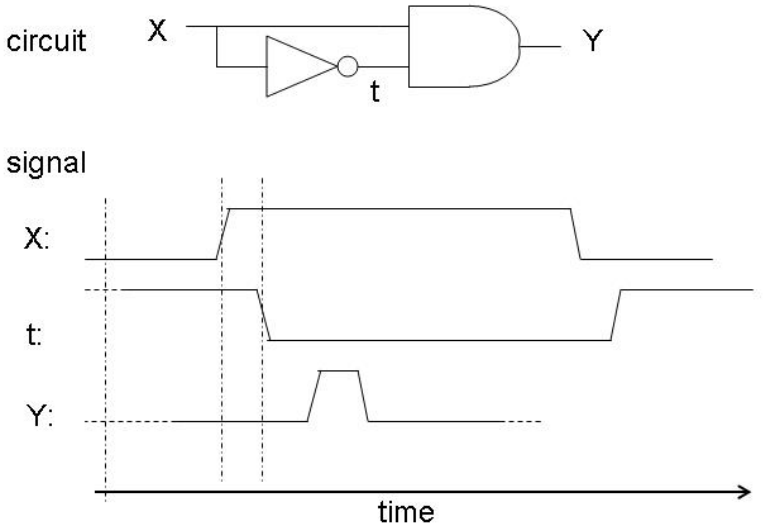
Introduction to Digital Logic

Sequential Logic

Time

- Till now, we have essentially ignored the issue of time. We assume digital circuits:
  - Perform their computations instantaneously
  - Stateless: once you present a new input, they forget everything about previous inputs

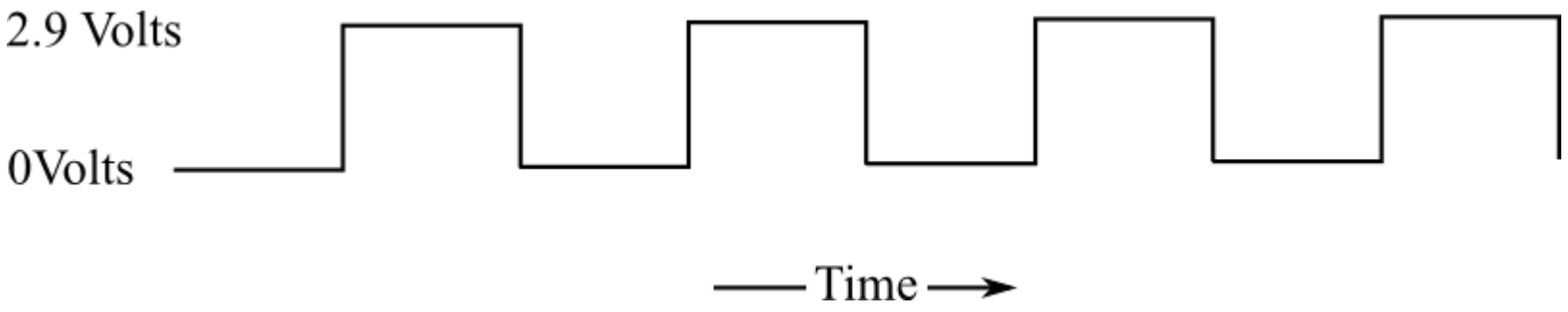
- In reality, time is an important issue:
  - Logic gate induces a small amount of delay (on the order of a few nanoseconds)
  - We want our circuit to have some form of memory



Clocks

- A microprocessor is composed of many different circuits that are operating simultaneously – if each circuit X takes in inputs at time T_{in}, takes time T_{exe} to execute the logic, and produces outputs at time T_{out}, imagine the complications in coordinating the tasks of every circuit

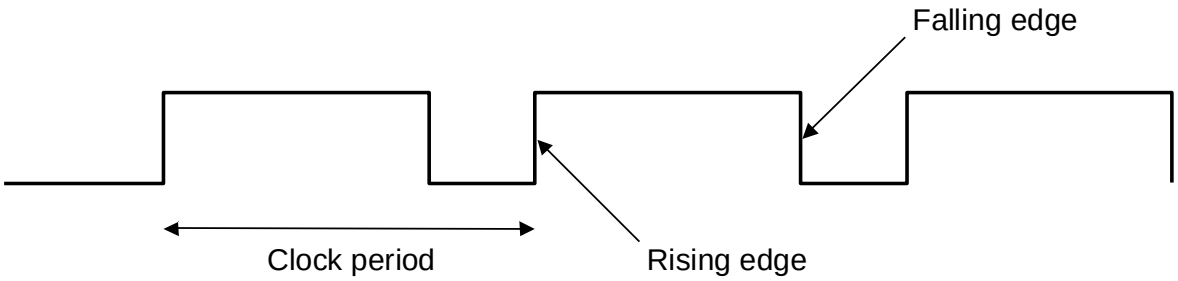
- Solution: all circuits on the chip share a clock signal (a square wave) that tells every circuit when to accept inputs, how much time they have to execute the logic, and when they must produce outputs



Clock Terminology

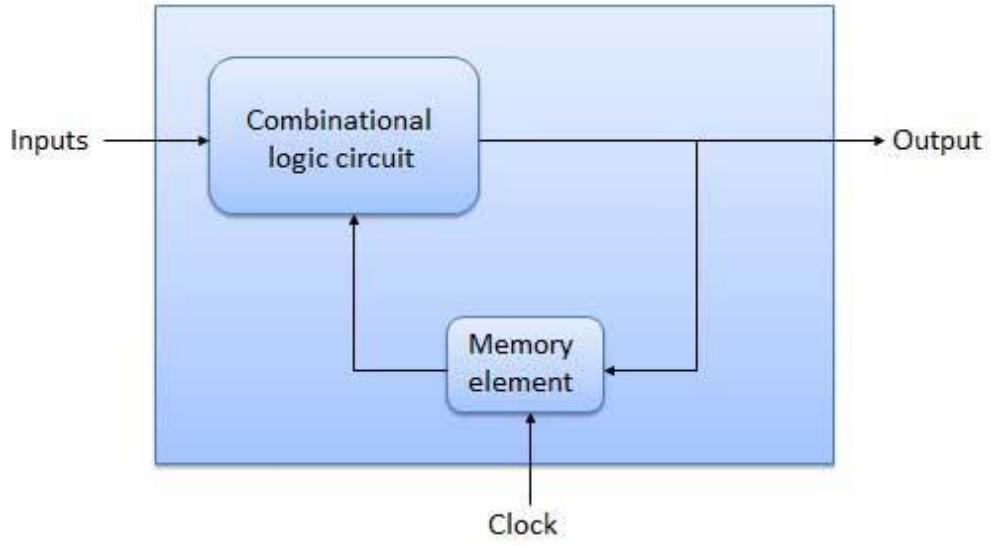
- A **clock** is a free-running signal with a fixed **cycle time** (called **clock period**) or, equivalently, a fixed **clock frequency** (i.e., inverse of the cycle time).

- **Edge-triggered clocking**:
  - Design methodology for sequential logic circuits in which all state changes occur on a clock edge (**rising edge** or **falling edge**).

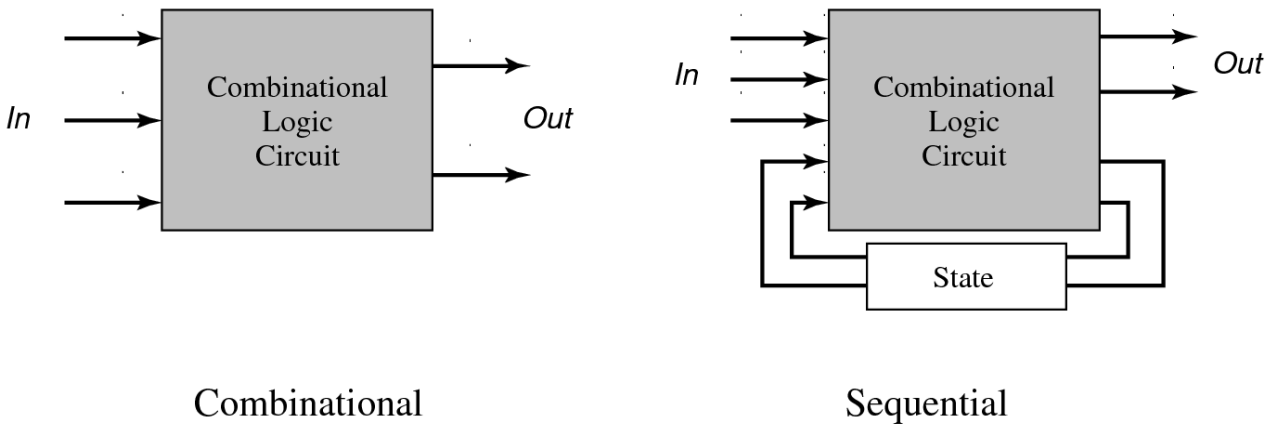


- Example $4\ GHZ = clock\ speed = \frac{1}{cycle\ time} = \frac{1}{250\ ps}$

- Clocked systems are also called **synchronous systems**.
- **Sequential logic circuits** are circuits whose output depends on **both** the current input and the value stored in the memory of the circuit (called **state**)
- Sequential logic circuits often depend on a clock to determine when the memory or state element of the circuit is updated.



Sequential Circuit Features

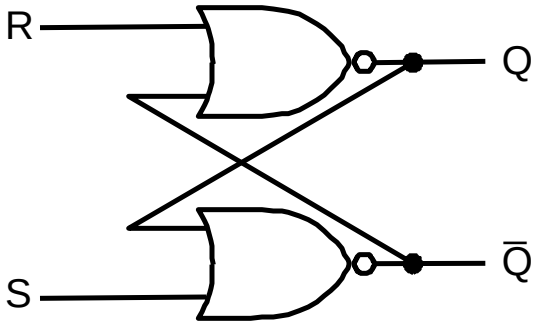


Combinational Sequential

- Consists of combinational circuit and a storage elements
- The rising/falling edge causes the 'state' storage to store some updated values
- This state will not change for an entire cycle (until next rising/falling edge)
- The combinational circuit has some time to accept the value of 'state' and 'inputs' and produce 'outputs'
- Some of the outputs (i.e. the value of the next 'state') may feed back and they're only seen in the next cycle
- Example: counter, register

S-R Latch

- **S-R latches** (set-reset latches):
  - **Unclocked** memory element built from a pair of cross-coupled NOR gates (i.e., OR gates with inverted outputs).

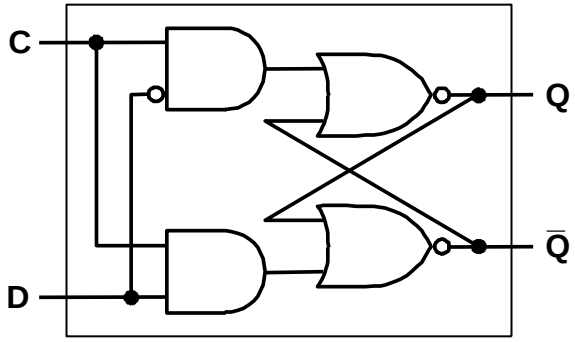


Inputs		Outputs	
R	S	Q	Not Q
0	0	Latch	
1	0	0	1
0	1	1	0
1	1	Invalid Input	

  - **Operation:**
    - When Set is **asserted** (high), a 1 is stored
    - When Reset is asserted, a 0 is stored
    - When both are **de-asserted** (low), the previous state is preserved
    - When both are high, the output is unstable,- this set of input is therefore not allowed

D Latch

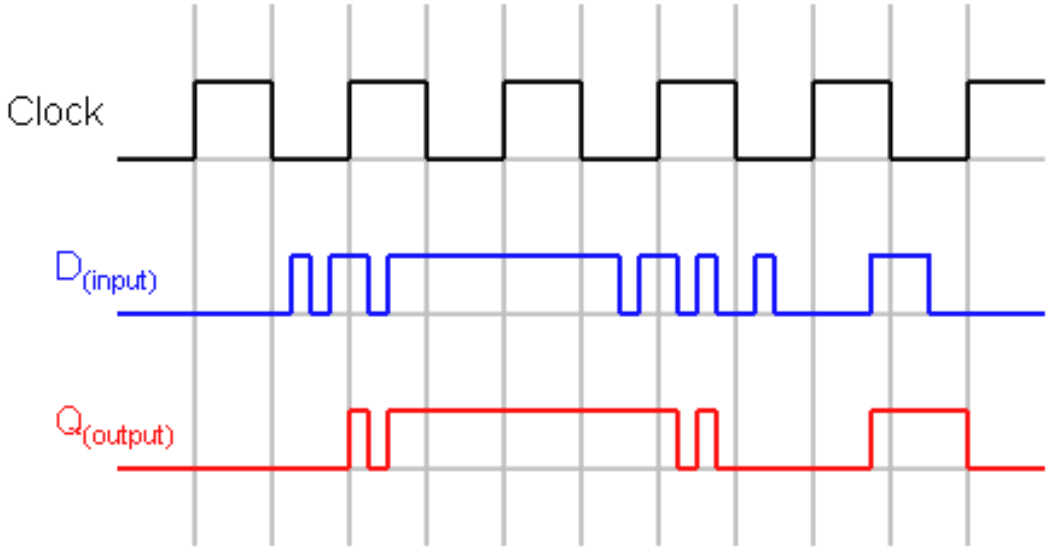
- **D latches** are **clocked** (i.e., state changes are triggered by a clock).
- **Operation:**
  - the value of the input D signal (data) is stored only when the clock signal C is asserted – the previous state is preserved when the clock is de-asserted.



Inputs		Outputs	
C	D	Q	Not Q
0	0	Latch	
0	1	Latch	
1	0	0	1
1	1	1	0

- **Digital timing diagram**
  - A representation of a set of signals in the time domain
  - Can contain many rows, usually one of them being the clock

- **Example**: timing diagram of D latch

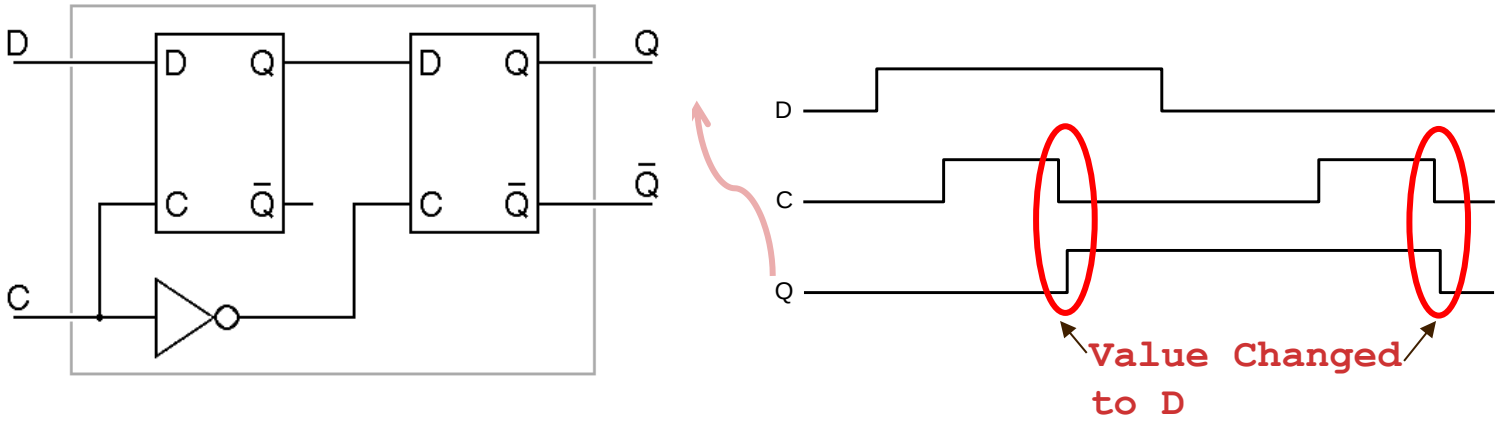


D Flip-Flop

- **D flip-flops**, like D latches, are **clocked**.

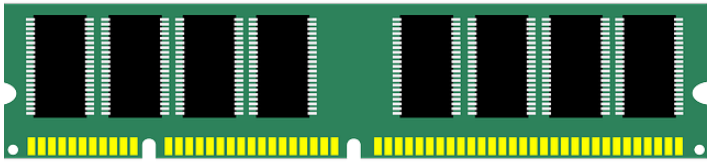
- **Terminology:**
  - Latch: outputs can change any time the clock is asserted (high)
  - Flip-flop: outputs can change only on a clock edge

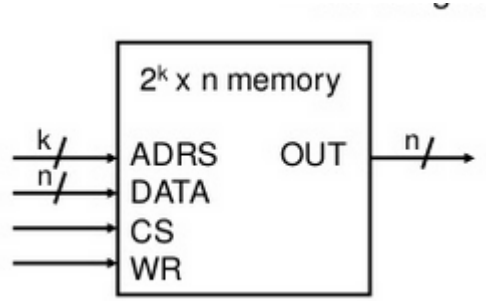
- **Falling-edge-triggered master-slave D flip-flop**:
- Operation:
  - When the clock input C changes from asserted to de-asserted, the Q output stores the value of the D input.



Flip-Flops and Memory elements

- D-Latches and flip-flops can be used to build memory elements used by digital computers to hold data temporarily.
  - **register file** is a structure in the datapath consisting of a set of registers that can be read and written by supplying a register number to be accessed. These represent the hardware "variables"
  - **Static random access memories** (**SRAMs**): relatively small memories used for cache and built with flip-flops

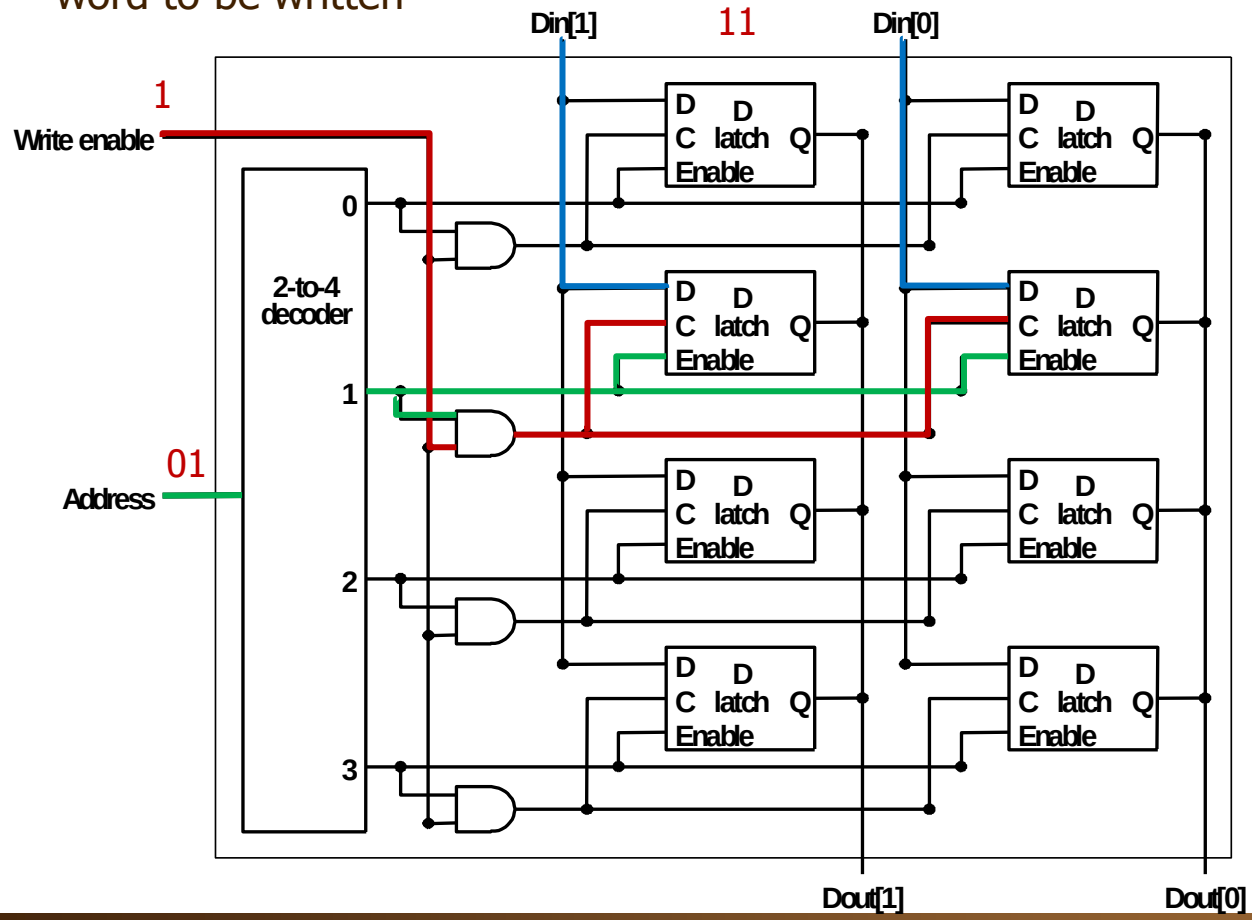




Block diagram of RAM chip

Example: a 4 x 2 SRAM

- Operation
  - Stores 4 words of data each word has 2 bits
  - Reads the content of one word at a time by supplying its address on 2 bits (i.e., word 00, or 01, or 10, or 11)
  - Write data to one word by supplying 2 bits of Data and the 2-bit address of the word to be written



Summary

- Computers use two reference levels of voltage to represent data as binary digits
- **Boolean Algebra** to express any operation on bits as a logic equation or a equivalently a truth table
  - Three basic operations: AND, OR, and NOT
  - Basic rules of Boolean Algebra
- **The transistor (an electronic switch)** is the basis for building logic gates
- **Logic Gates** are basic building blocks of more complex logic circuits

Summary

- Two types of logic circuits:
  - **Combinational circuits**: process the inputs to produce the output
    - Multiplexers
    - Decoders
    - Arithmetic and logic operations
  - **Sequential circuits**: process the input and the **state element** (memory) of the circuit to produce the output
    - Memory, registers
    - Not possible to access (read) and update (write) the state element simultaneously, therefore a clock is necessary to control when operations are done
    - **Edge-triggered clocks** are more efficient and more accurate, than level-based clocks
    - Latches, Flip-flops