

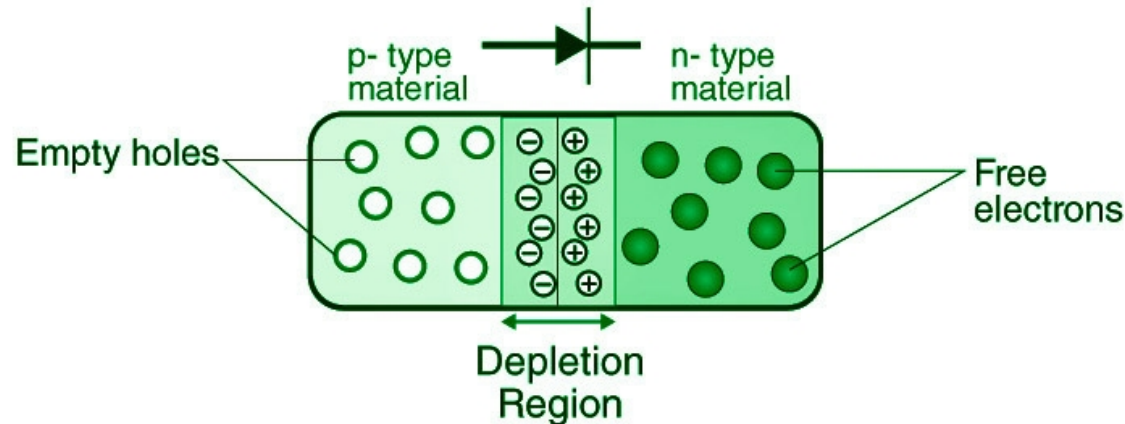


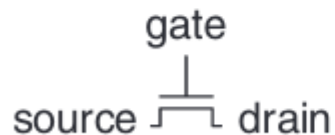
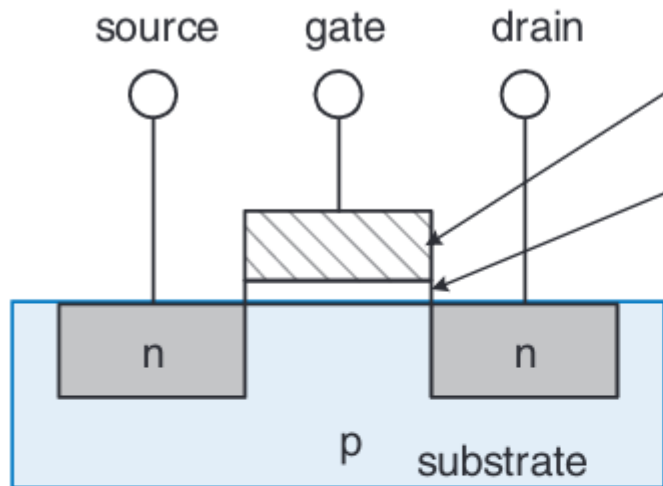
УНИВЕРСИТЕТ ИТМО

«Вводная лекция»

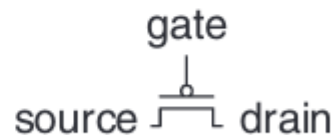
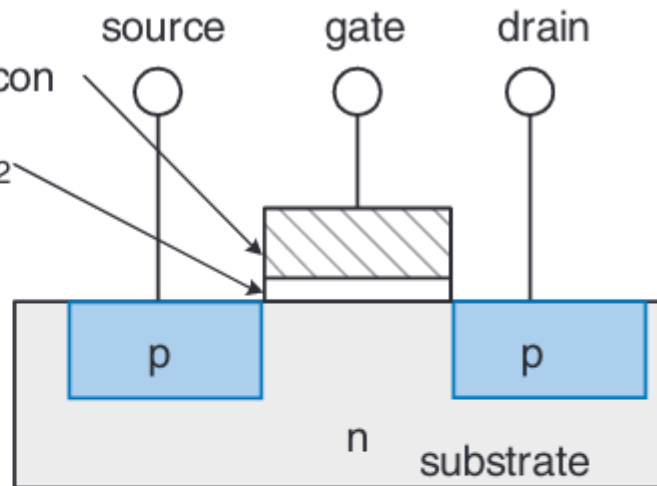
Весна, 2023

- ✓ Диод — электронный компонент, который пропускает электрический ток только в одну сторону;
- ✓ Полупроводник **n-типа**;
- ✓ Полупроводник **p-типа**.

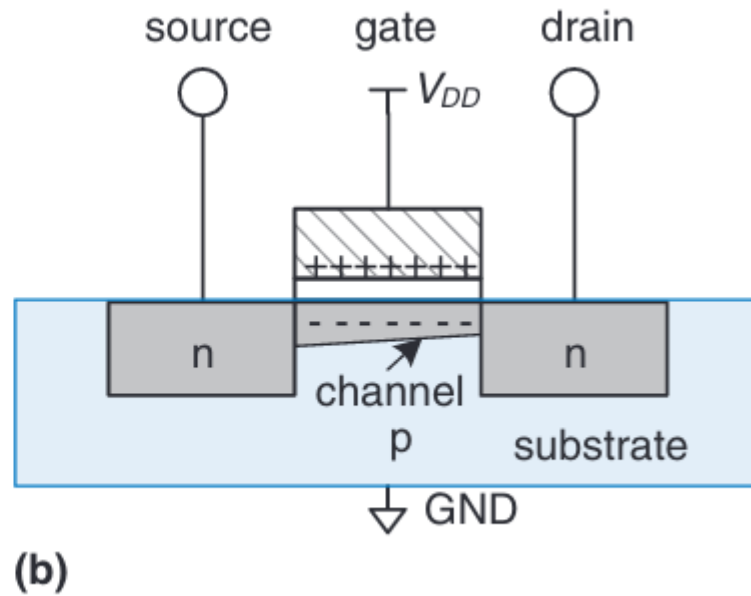
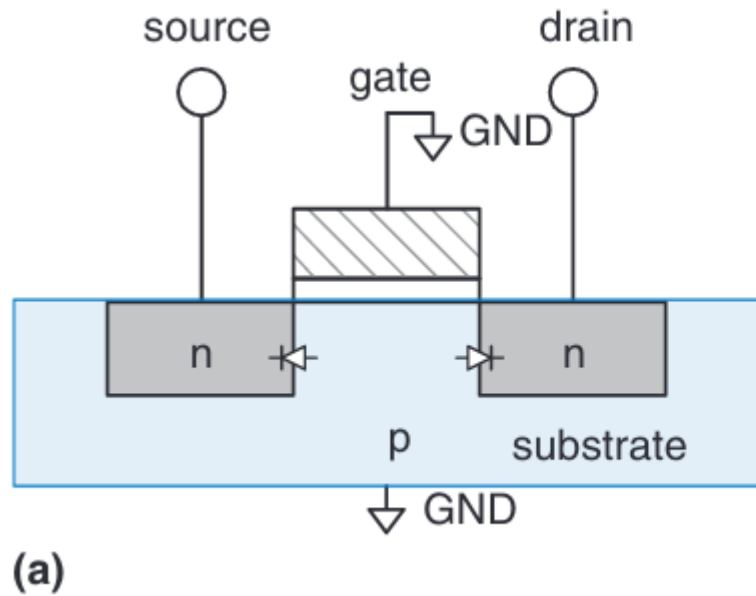




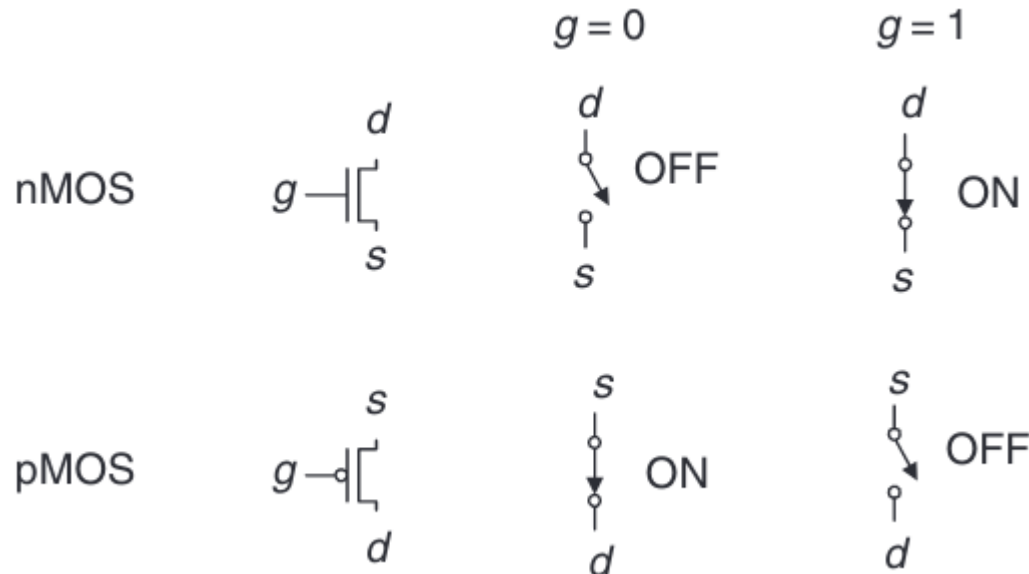
(a) nMOS

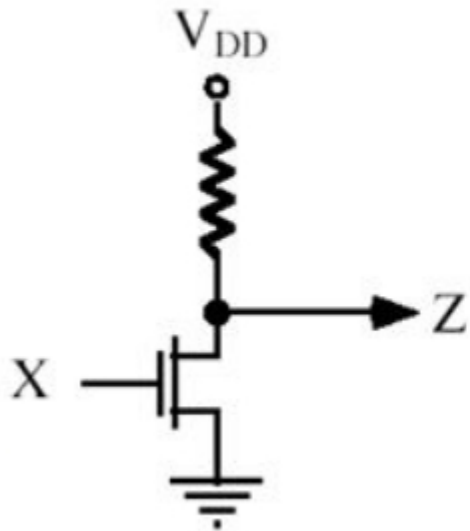


(b) pMOS

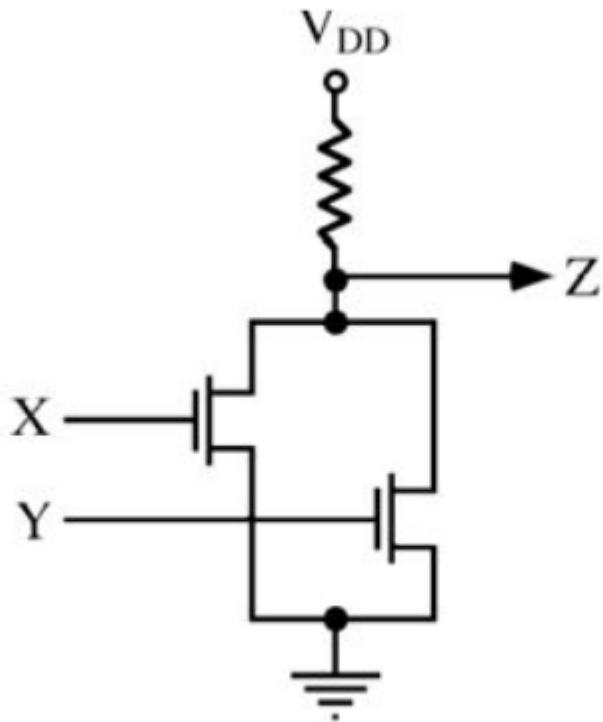


- ✓ Транзистор — контролируемый «ключ», который может замыкаться/размыкаться **в зависимости от прикладываемого напряжения.**

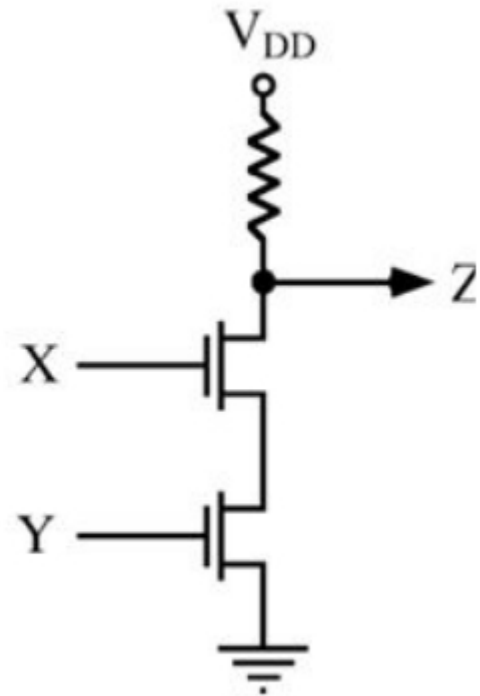




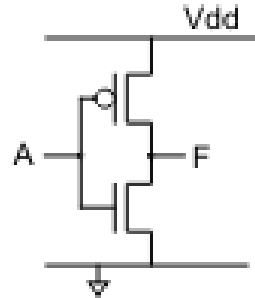
NMOS Inverter



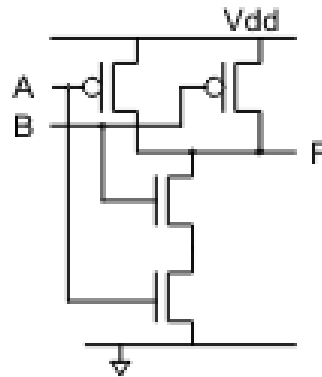
NMOS NOR Gate



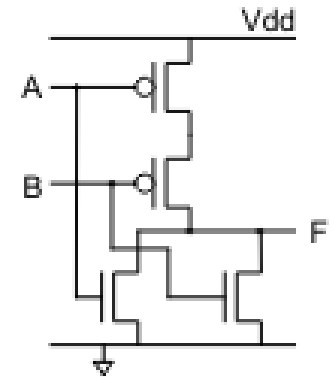
NMOS NAND Gate



CMOS INVERTER

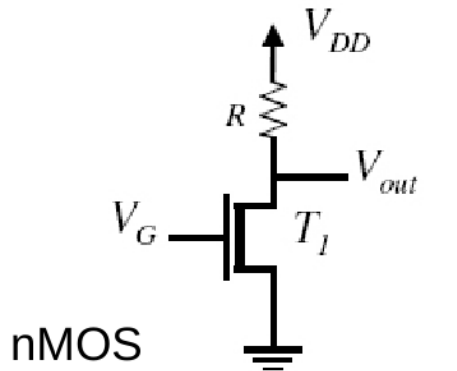


CMOS NAND

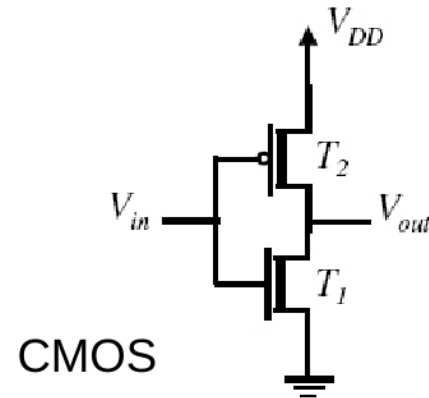


CMOS NOR

- ✓ Ток протекает от источника питания в «землю», когда транзистор «открыт»;
- ✓ Повышенное энергопотребление;
- ✓ Побочные эффекты на различных уровнях.

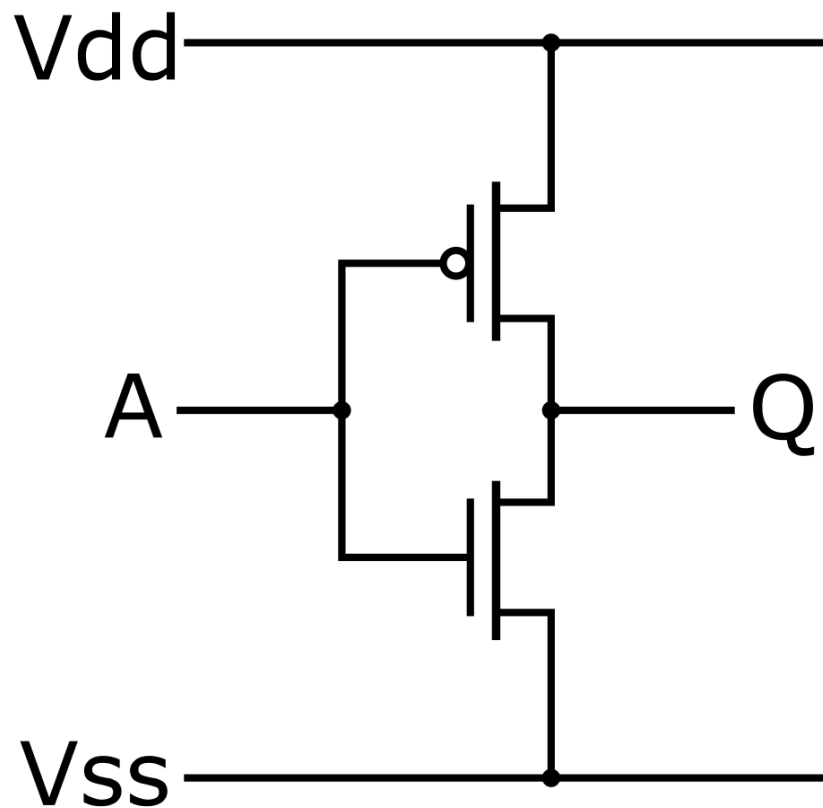


V_G	T_1	V_{out}
0	<i>off</i>	1
1	<i>on</i>	0

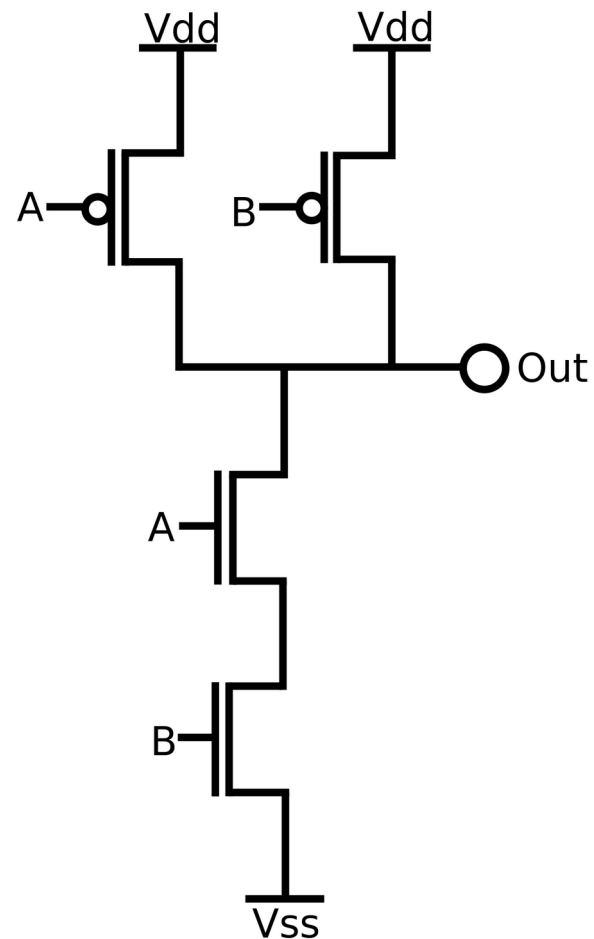


V_{in}	T_1	T_2	V_{out}
0	<i>off</i>	<i>on</i>	1
1	<i>on</i>	<i>off</i>	0

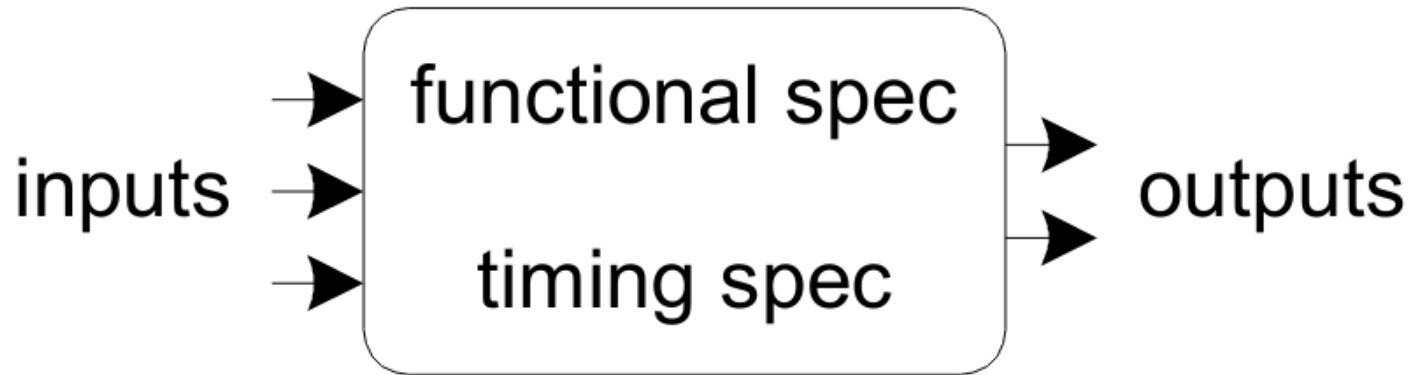
A	Q
0	1
1	0



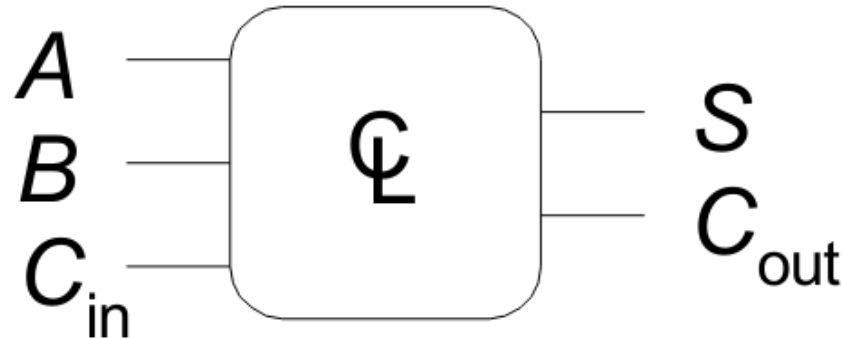
A	B	Out
0	0	1
0	1	1
1	0	1
1	1	0



- ✓ Входные порты;
- ✓ Выходные порты;
- ✓ Функциональная спецификация;
- ✓ Временная спецификация



- ✓ Функциональная спецификация — функция выходных значений от входных.



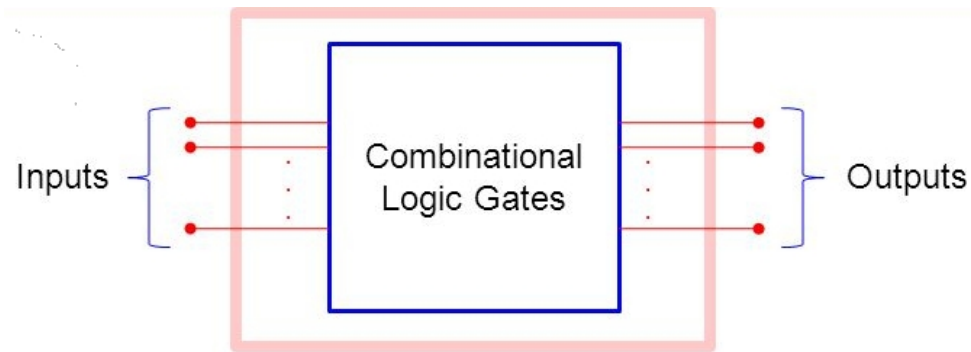
$$\begin{aligned} S &= A \oplus B \oplus C_{in} \\ C_{out} &= AB + AC_{in} + BC_{in} \end{aligned}$$

✓ Комбинационная логика:

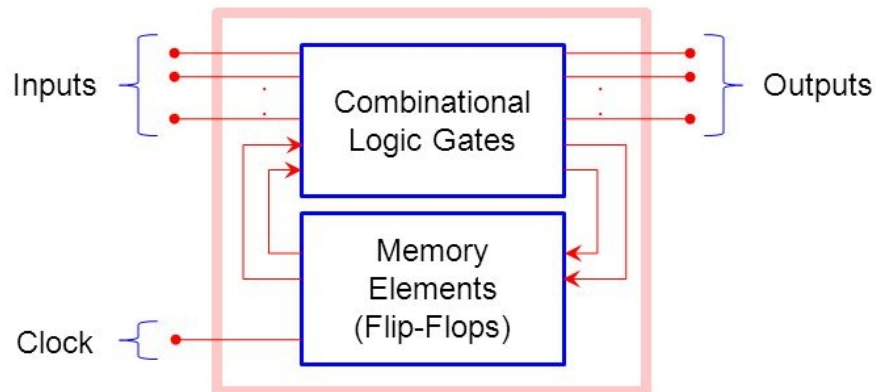
- ✓ Не содержит запоминающих элементов;
- ✓ Выходные сигналы — функция от текущих входных сигналов.

✓ Последовательностная логика:

- ✓ Содержит запоминающие элементы;
- ✓ Выходные сигналы — функция от текущих и предыдущих входных сигналов.

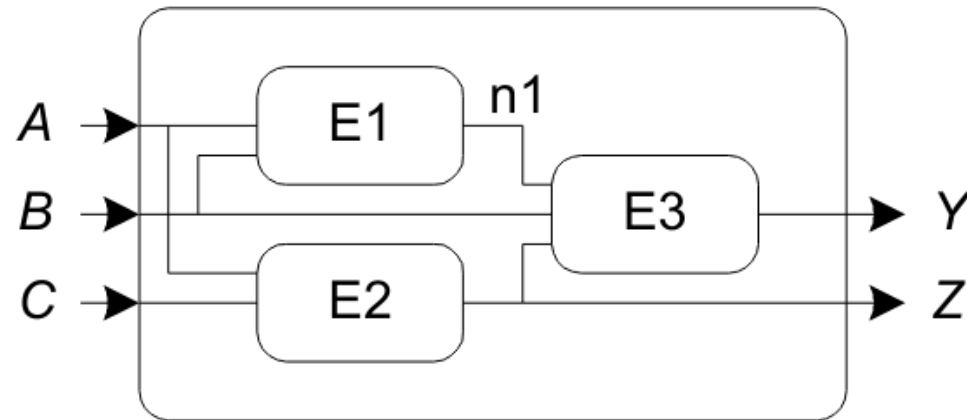


Комбинационная логика

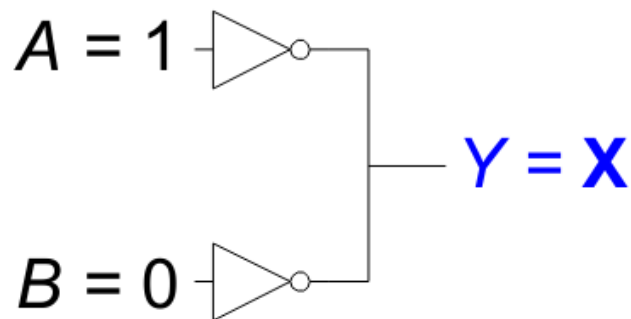


Последовательностная логика 12

- ✓ Любой составной элемент — комбинационный;
- ✓ Каждый входной контакт может иметь **не более** одного источника;
- ✓ Не имеет циклических путей.

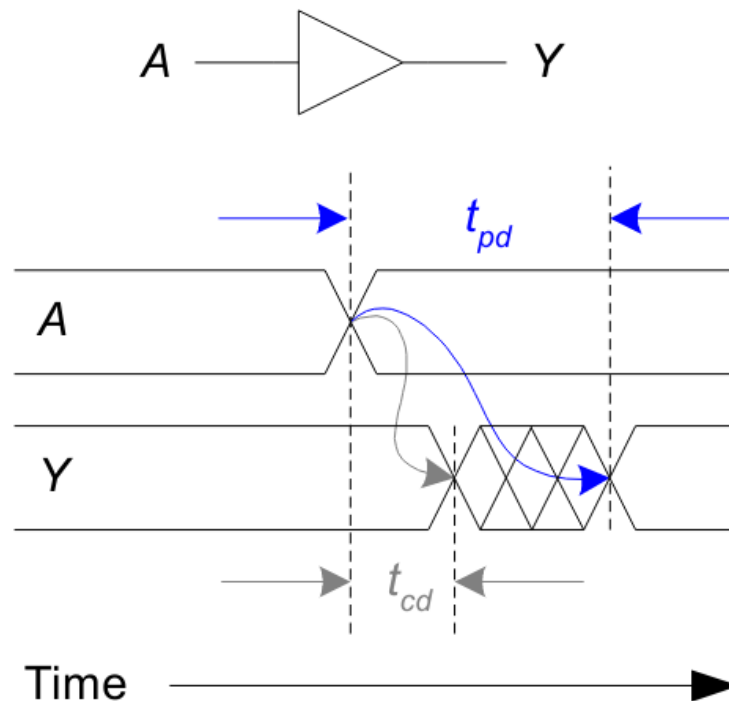


- ✓ Состояние конкурирования — на вход одновременно подаётся «1» и «0»;
- ✓ Реальное значение попадает в неопределенную область;
- ✓ Высокое энергопотребление.





- ✓ Задержка распространения (t_{pd}) — максимальная задержка;
- ✓ Задержка реакции (t_{cd}) — минимальная задержка.



- ✓ Оператор «assign»;
- ✓ Процедурный блок «always» и блокирующие присваивания.

```
assign result = A & B;

always @(A, B) begin
    result = A & B;
end
```

Логическое «И»

```
assign result = condition ? A : B;

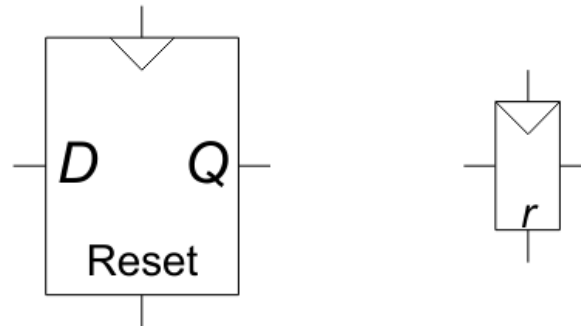
always @(condition, A, B) begin
    if (condition)
        result = A;
    else
        result = B;
end
```

Мультиплексор

```
always @(*) begin
    case (data_in)
        8'b0000_0001: data_out=3'd0;
        8'b0000_0010: data_out=3'd1;
        8'b0000_0100: data_out=3'd2;
        8'b0000_1000: data_out=3'd3;
        8'b0001_0000: data_out=3'd4;
        8'b0010_0000: data_out=3'd5;
        8'b0100_0000: data_out=3'd6;
        8'b1000_0000: data_out=3'd7;
    endcase
end
```

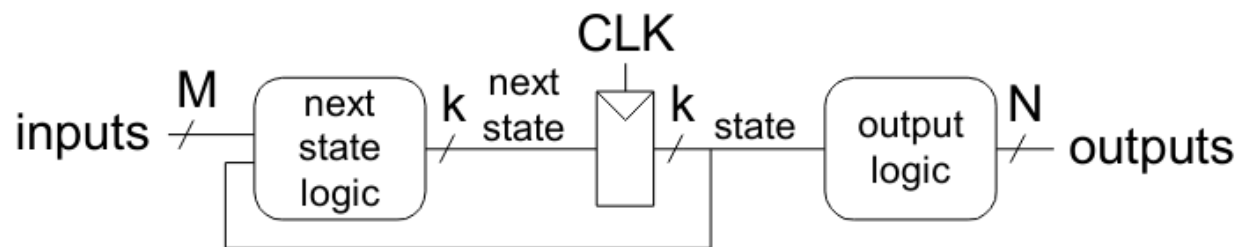
Шифратор

- ✓ Выходы схемы зависят **от текущих и от предыдущих** входных значений;
- ✓ Схемы с «памятью»;
- ✓ Предыдущие значения схемы формируют текущее «состояние» схемы;
- ✓ Текущее «состояние» схемы влияет на её будущее состояние;
- ✓ Последовательность входных сигналов имеет значение;

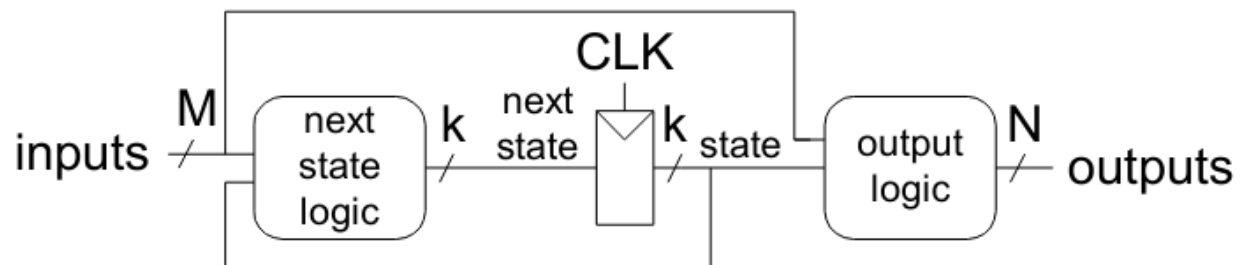




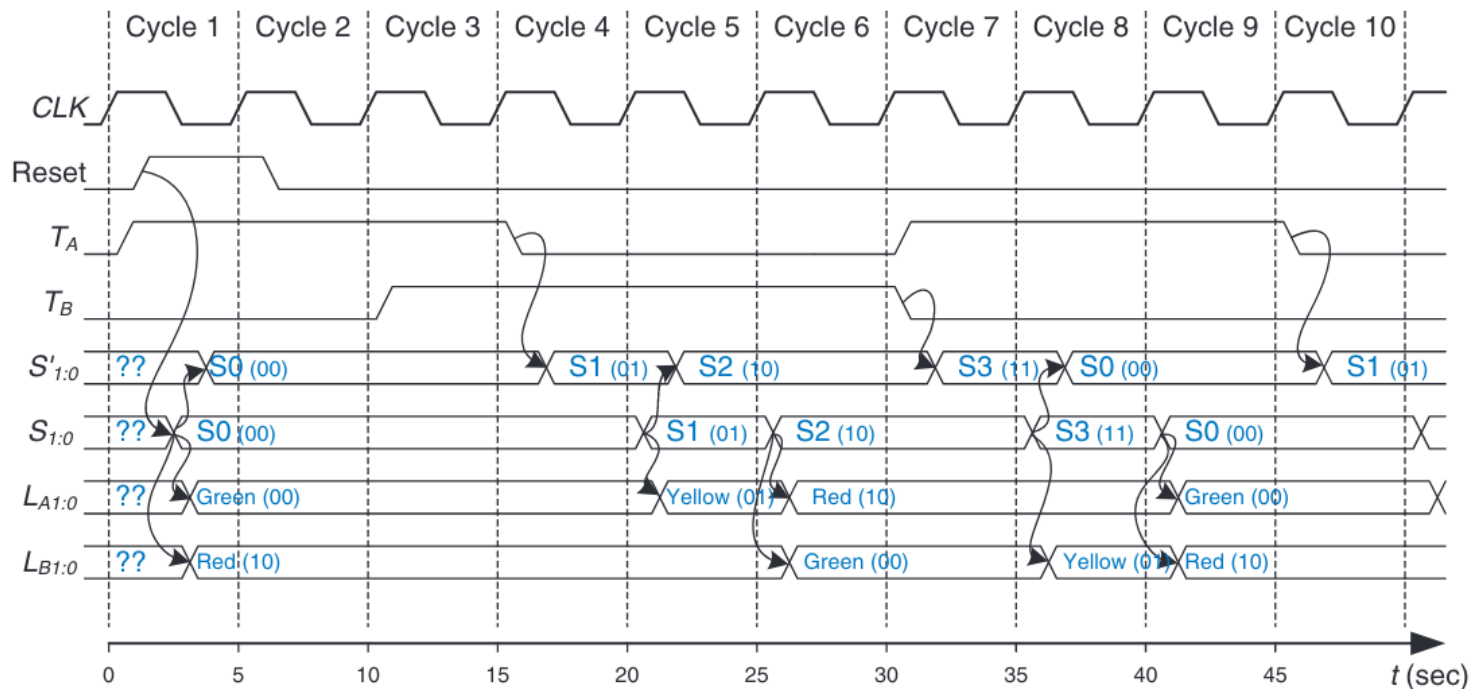
Moore FSM



Mealy FSM

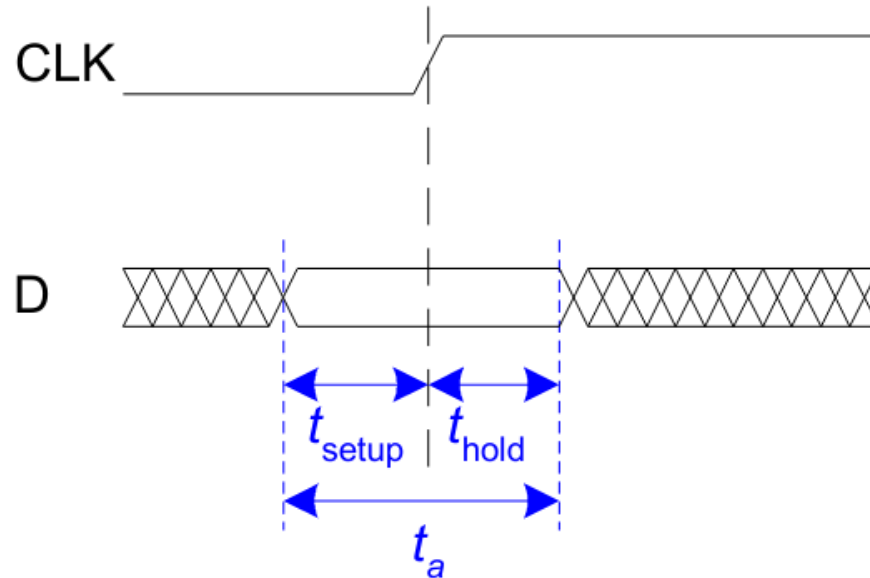


- ✓ Поступает во все запоминающие элементы;
- ✓ Периодичен — имеет собственный тактовый период и тактовую частоту;

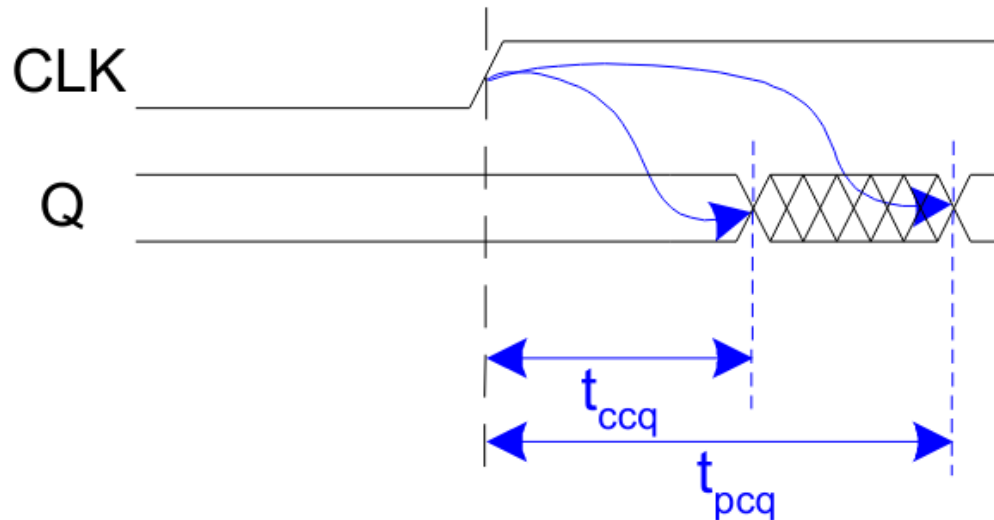


- ✓ Триггер принимает значение по фронту тактового сигнала;
- ✓ Значение не должно меняться в момент «фиксации»;
- ✓ Значение не должно меняться в некоторой окрестности от фронта тактового сигнала;
- ✓ В противном случае, записываемое в триггер значение может быть не определено.

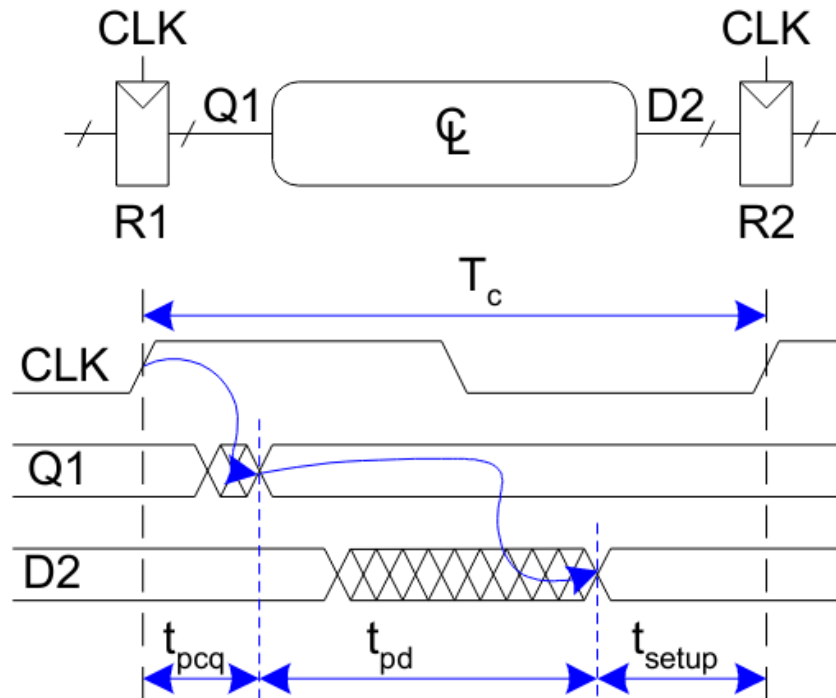
- ✓ Время предустановки — период перед фронтом;
- ✓ Время удержания — период после фронта;
- ✓ Сигнал должен быть стабилен.



- ✓ Задержка распространения (t_{pcq}) — от фронта до конца изменений;
- ✓ Задержка реакции (t_{ccq}) — от фронта до начала изменений;
- ✓ Считать от фронта тактового сигнала!



- ✓ Максимальная задержка прохождения сигнала от регистра R1 через комбинаторику к регистру R2;



$$T_c \geq t_{pcq} + t_{pd} + t_{setup}$$

$$t_{pd} \leq T_c - (t_{pcq} + t_{setup})$$

✓ Always-блок. Реакция на фронт/спад тактового сигнала

```
always @(posedge clk) begin
    op1_ff <= op1_next;
    op2_ff <= op2_next;
end
```

```
always @(posedge clk) begin
    if (enable) begin
        op1_ff <= op1_next;
        op2_ff <= op2_next;
    end
end
```

```
always @(posedge clk) begin
    if (~rst_n) begin
        op1_ff <= 8b'00000000;
        op2_ff <= 8b'00000000;
    end else if (enable) begin
        op1_ff <= op1_next;
        op2_ff <= op2_next;
    end
end
```

- ✓ Не смешивать описания комбинационной и последовательной логики в одном always-блоке;

```
always @(posedge clk) begin
    op1_ff <= A + B;
    op2_ff <= condition ? A : ~A + 1;
end
```

Некорректно

```
wire op1_next;
wire op2_next;

assign op1_next = A + B;
assign op2_next = condition ? A : ~A + 1;

always @(posedge clk) begin
    op1_ff <= op1_next;
    op2_ff <= op2_next;
end
```

Корректно

- ✓ Блокирующие присваивания для комбинационной логики;
- ✓ Неблокирующие присваивания для последовательностной логики.

```
always @(posedge clk) begin
  op1_ff = op1_next;
  op2_ff = op2_next;
end
```

```
always @(condition, A, B) begin
  if (condition)
    result <= A;
  else
    result <= B;
end
```

Некорректно

```
always @(posedge clk) begin
  op1_ff <= op1_next;
  op2_ff <= op2_next;
end
```

```
always @(condition, A, B) begin
  if (condition)
    result = A;
  else
    result = B;
end
```

Корректно

- ✓ Не смешивать блокирующие и неблокирующие присваивания в рамках одного блока.

```
always @(posedge clk) begin
  op_new_ff = opnew_next;
  op_old_ff <= op_new_ff;
end
```

Некорректно

```
always @(posedge clk) begin
  op_new_ff <= opnew_next;
  op_old_ff <= op_new_ff;
end
```

Корректно

- ✓ Всего около 9 практических занятий;
- ✓ Всего 4 лабораторных работы;
- ✓ У каждой лабораторной работы есть срок;
- ✓ Сдача в срок гарантирует максимальные личностные качества;
- ✓ Сдача после истечения срока (без уважительной причины) — минимальный балл за работу в любом случае;
- ✓ **Отчёт в электронной форме** должен быть на почте **перед** защитой.

- ✓ Сроки:
 - ✓ Срок Л.Р №1 — 20.02.23;
 - ✓ Срок Л.Р №2 — 20.03.23;
 - ✓ Срок Л.Р №3 — 17.04.23;
 - ✓ Срок Л.Р №4 — 15.05.23 (**возможно** 29.05.23).

- ✓ Построить КМОП-элемент логического базиса на транзисторах;
 - ✓ На основе построенного элемента построить БОЭ;
 - ✓ Рассчитать задержки;
 - ✓ Рассчитать максимальную частоту работы вашего БОЭ;
 - ✓ Провести эксперимент на максимальной частоте и на частоте превышающей максимальную. Объяснить результаты;
-
- ✓ Описать аналогичную схему в САПР Vivado на языке Verilog;
 - ✓ Верифицировать блок и объяснить результаты.

- ✓ Построить схему полусумматора оперирующего двумя одноразрядными операндами на базисе **И-ИЛИ-НЕ**;
- ✓ Два однобитных входа A и B;
- ✓ Один выход S;
- ✓ $S = A + B$.