

# Глава 3

## ***Цифровая схемотехника и архитектура компьютера, второе издание***

---

Дэвид М. Харрис и Сара Л. Харрис

# Цифровая схемотехника и архитектура компьютера

Эти слайды предназначены для преподавателей, которые читают лекции на основе учебника «Цифровая схемотехника и архитектура компьютера» авторов Дэвида Харриса и Сары Харрис. Бесплатный русский перевод второго издания этого учебника можно загрузить с сайта компании Imagination Technologies:

<https://community.imgtec.com/downloads/digital-design-and-computer-architecture-russian-edition-second-edition>

Процедура регистрации на сайте компании Imagination Technologies описана на странице:

<http://www.silicon-russia.com/2016/08/04/harris-and-harris-2/>

# Благодарности









**Перевод данных слайдов на русский язык был выполнен командой сотрудников университетов и компаний из России, Украины, США в составе:**

- Александр Барабанов - доцент кафедры компьютерной инженерии факультета радиофизики, электроники и компьютерных систем Киевского национального университета имени Тараса Шевченко, кандидат физ.-мат. наук, Киев, Украина;
- Антон Брюзгин - начальник отдела АО «Вибро-прибор», Санкт-Петербург, Россия.
- Евгений Короткий - доцент кафедры конструирования электронно-вычислительной аппаратуры факультета электроники Национального технического университета Украины «Киевский Политехнический Институт», руководитель открытой лаборатории электроники Lamra, кандидат технических наук, Киев, Украина;
- Евгения Литвинова – заместитель декана факультета компьютерной инженерии и управления, доктор технических наук, профессор кафедры автоматизации проектирования вычислительной техники Харьковского национального университета радиоэлектроники, Харьков, Украина;
- Юрий Панчул - старший инженер по разработке и верификации блоков микропроцессорного ядра в команде MIPS I6400, Imagination Technologies, отделение в Санта-Кларе, Калифорния, США;
- Дмитрий Рожко - инженер-программист АО «Вибро-прибор», магистр Санкт-Петербургского государственного автономного университета аэрокосмического приборостроения (ГУАП), Санкт-Петербург, Россия;
- Владимир Хаханов – декан факультета компьютерной инженерии и управления, проректор по научной работе, доктор технических наук, профессор кафедры автоматизации проектирования вычислительной техники Харьковского национального университета радиоэлектроники, Харьков, Украина;
- Светлана Чумаченко – заведующая кафедрой автоматизации проектирования вычислительной техники Харьковского национального университета радиоэлектроники, доктор технических наук, профессор, Харьков, Украина.



# Глава 3 :: Темы

- Введение
- Защелки и Триггеры
- Проектирование синхронных логических схем
- Конечные автоматы
- Синхронизация последовательностных схем
- Параллелизм

Application Software	>" he wor.
Operating Systems	
Architecture	
Micro-architecture	
Logic	
Digital Circuits	
Analog Circuits	
Devices	
Physics	

# Введение

- Выходы последовательностной схемы зависят как от текущих, *так и от* прошлых значений на входах - схема имеет **память**.
- Некоторые определения:
  - **Состояние**: Вся информация о схеме, необходимая для определения ее будущего поведения
  - **Защелки и Триггеры**: Элементы, хранящие один бит состояния
  - **Синхронные последовательностные схемы**: За комбинационной схемой следует набор триггеров

# Последовательностные

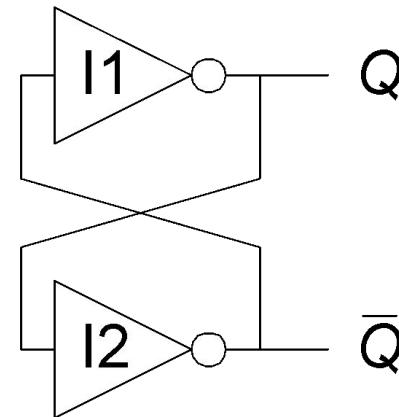
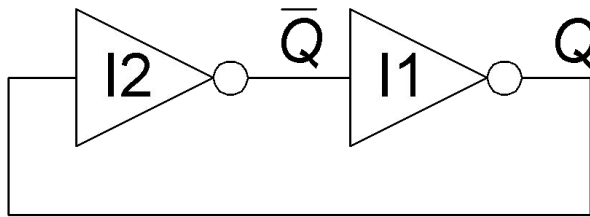
- Выдают последовательность событий
- Имеют (краткосрочную) память
- Для сохранения информации используют обратную связь с выходов на входы

# Элементы, хранящие

- Состояние схемы влияет на ее будущее поведение
- Элементы хранят состояние схемы
  - Бистабильная схема
  - RS-триггер
  - D-защелка
  - D-триггер

# Бистабильная схема

- Основной блок для построения других элементов, хранящих состояние
- Два выхода:  $Q$ ,  $\bar{Q}$
- Входов нет





# Анализ бистабильной схемы

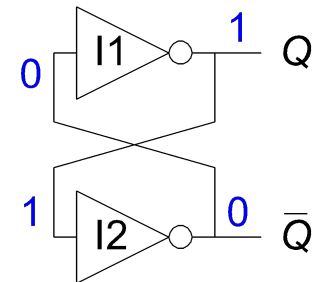
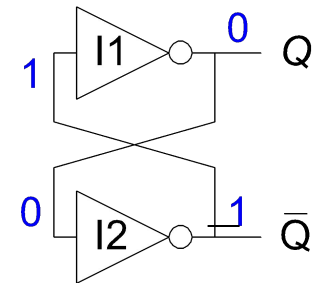
- Рассмотрим два возможных случая:

–  $Q = 0$ :

Тогда  $\bar{Q} = 1$ ,  $Q = 0$   
(нет противоречия)

–  $Q = 1$ :

Тогда  $\bar{Q} = 0$ ,  $Q = 1$   
(нет противоречия)

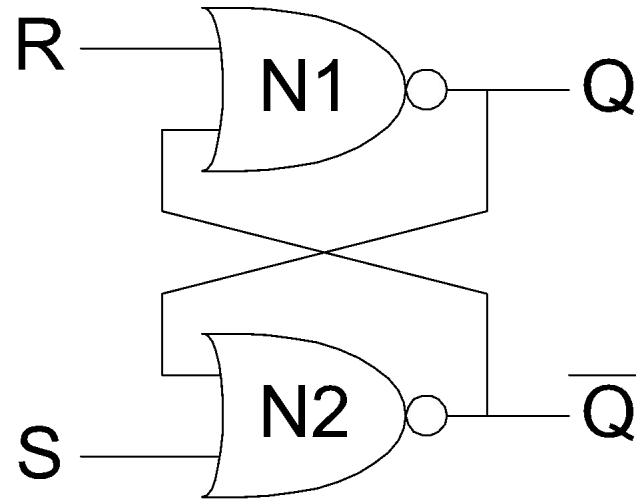


- Хранит 1 бит переменной состояния  $Q$  (или  $\bar{Q}$ )
- **Входы управления состоянием отсутствуют**

# RS (Reset/Set, Сброс/Установка)

ТРИГГЕР

- RS-триггер

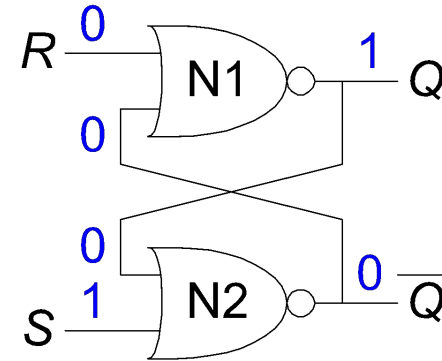


- Рассмотрим четыре возможные случая:
  - $S = 1, R = 0$
  - $S = 0, R = 1$
  - $S = 0, R = 0$
  - $S = 1, R = 1$

# Анализ RS-триггера

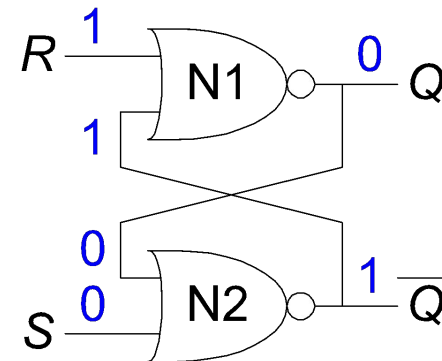
–  $S = 1, R = 0$ :

Тогда  $Q = 1$  и  $\bar{Q} = 0$



–  $S = 0, R = 1$ :

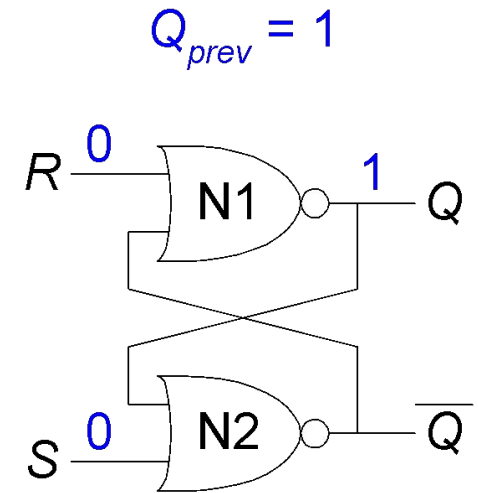
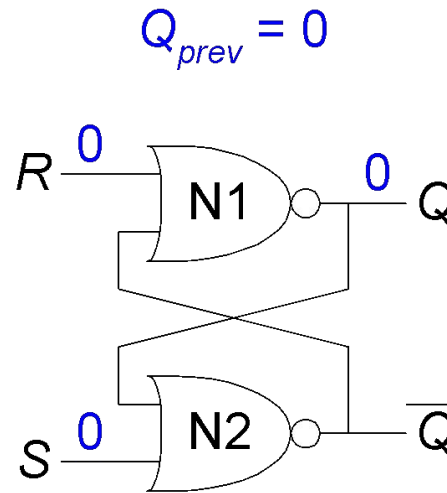
Тогда  $\bar{Q} = 1$  и  $Q = 0$



# Анализ RS-триггера

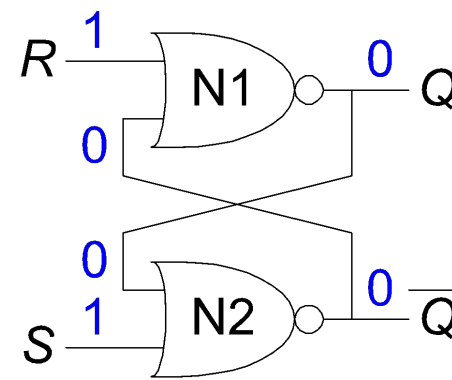
–  $S = 0, R = 0$ :

Тогда  $Q = Q_{prev}$



–  $S = 1, R = 1$ :

Тогда  $Q = 0, \bar{Q} = 0$

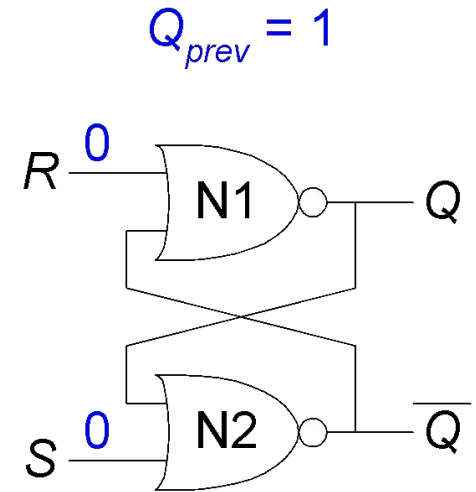
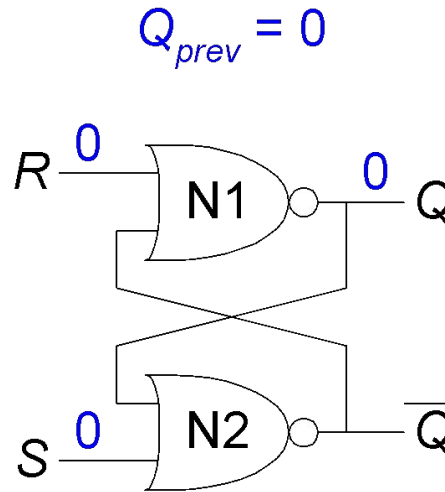


# Анализ RS-триггера

–  $S = 0, R = 0$ :

Тогда  $Q = Q_{prev}$

– **Память!**

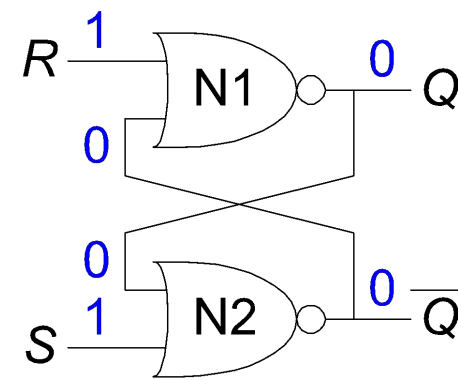


–  $S = 1, R = 1$ :

Тогда  $Q = 0, \bar{Q} = 0$

– **Запрещенное состояние**

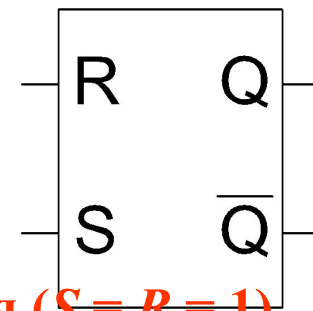
$Q \neq \text{NOT } Q$



# Обозначение RS-триггера

- RS обозначает триггер со входами Reset/Set, Сброс/Установка
  - Хранит один бит состояния ( $Q$ )
- Хранимое значение определяется входами  $S$ ,  $R$ 
  - **Set (Установка):**  
Устанавливает выход в 1  
( $S = 1, R = 0, Q = 1$ )
  - **Reset (Сброс):**  
Устанавливает выход в 0  
( $S = 0, R = 1, Q = 0$ )
- **Нужно предпринять специальные меры для исключения появления запрещенного состояния ( $S = R = 1$ )**

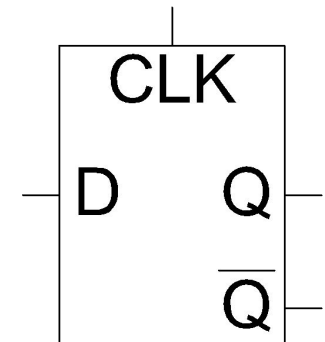
SR Latch  
Symbol



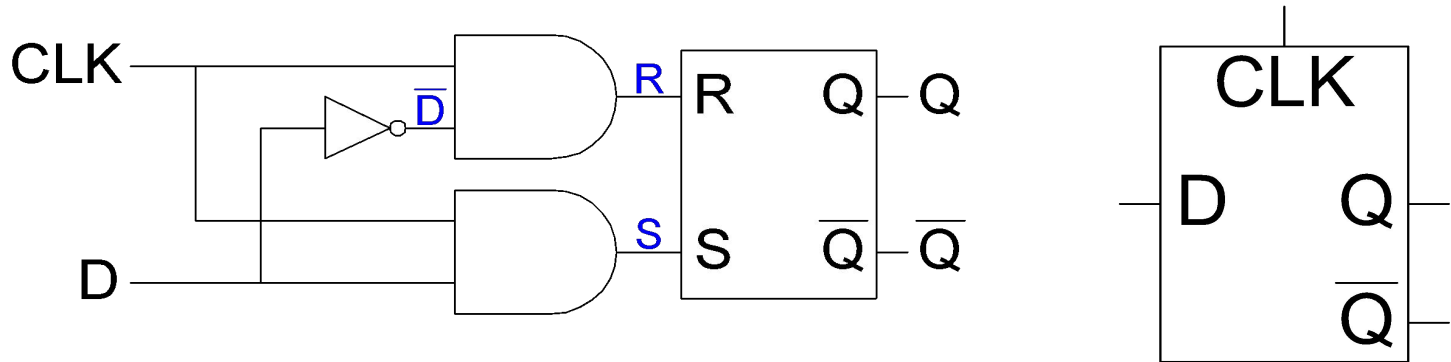
# D-защелка

- Два входа  $CLK$ ,  $D$ 
  - $CLK$ : определяет *когда* выход изменяется
  - $D$  (вход данных): определяет новое *значение* выхода
- Работа
  - Когда  $CLK = 1$ ,  $D$  проходит на выход  $Q$  (защелка *прозрачна*)
  - Когда  $CLK = 0$ ,  $Q$  хранит предыдущее значение (защелка *непрозрачна*)
- Запрещенное состояние не возникает, когда  $Q \neq \text{NOT } \bar{Q}$

D Latch  
Symbol



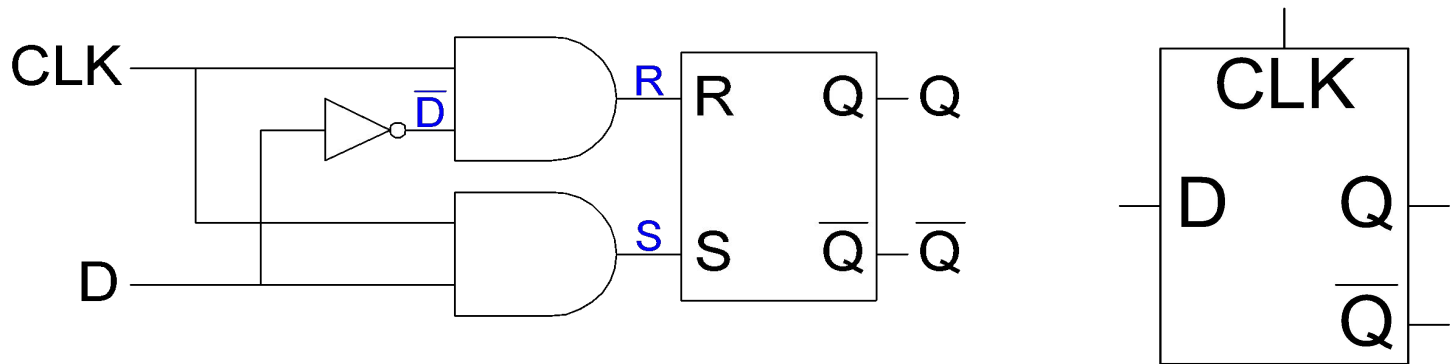
# Внутренняя структура D защелки



$CLK$	$D$	$\overline{D}$	$S$	$R$	$Q$	$\overline{Q}$
0	X					
1	0					
1	1					



# Внутренняя структура D защелки

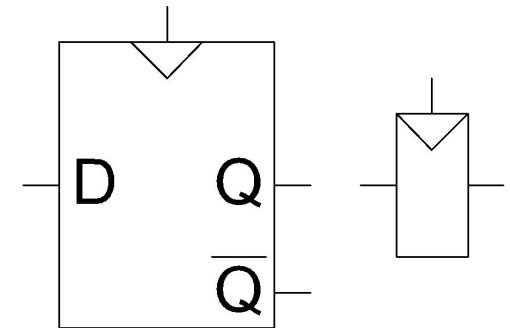


$CLK$	$D$	$\overline{D}$	$S$	$R$	$Q$	$\overline{Q}$
0	X	$\overline{X}$	0	0	$Q_{prev}$	$\overline{Q}_{prev}$
1	0	1	0	1	0	1
1	1	0	1	0	1	0

# D-триггер

- **Входы:**  $CLK$ ,  $D$
- **Работа**
  - Фиксирует значение  $D$  по переднему фронту  $CLK$ 
    - Когда  $CLK$  изменяется от 0 к 1,  $D$  проходит на выход  $Q$
    - В противном случае  $Q$  хранит предыдущее значение
  - $Q$  изменяется только по переднему фронту  $CLK$
- Такой триггер *управляется фронтом*
- Активируется фронтом тактового сигнала

D Flip-Flop Symbols



# Внутренняя структура D триггера

- Две последовательно соединенные защелки (L1 и L2), которые управляются комплементарными тактовыми сигналами

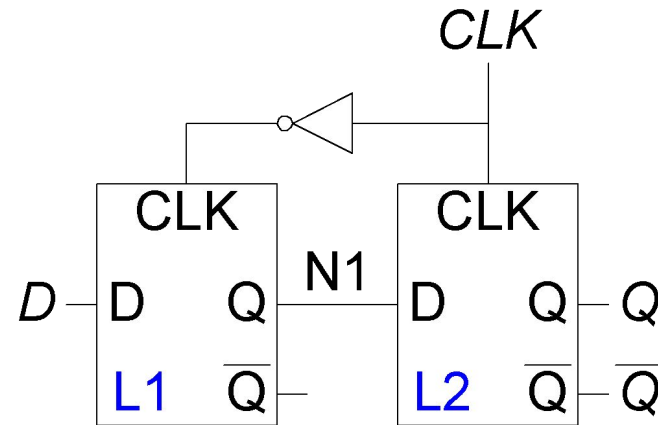
- Когда  $CLK = 0$

- L1 прозрачна
- L2 непрозрачна
- $D$  проходит до N1

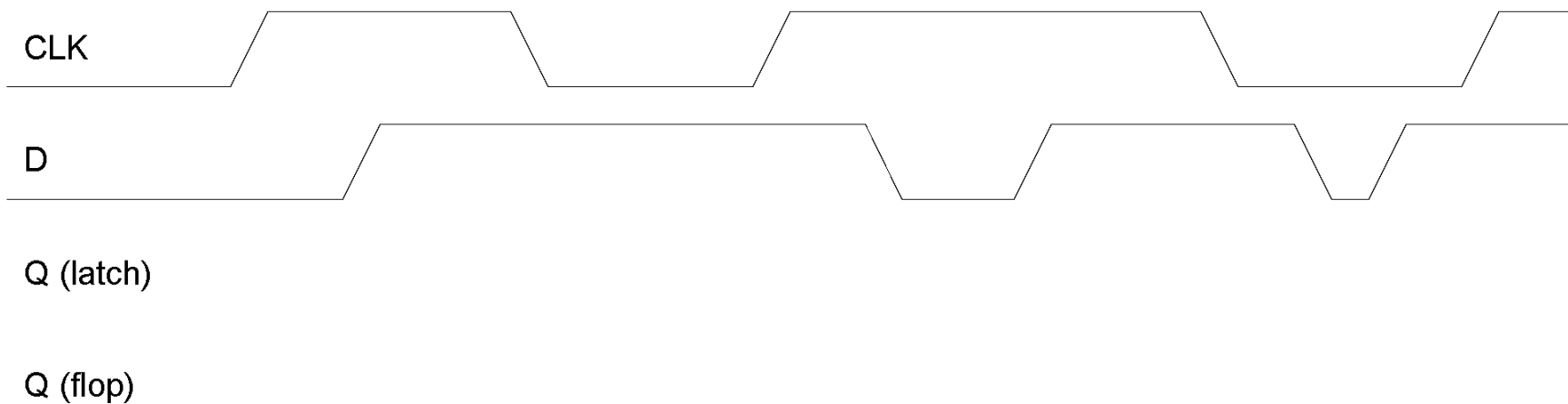
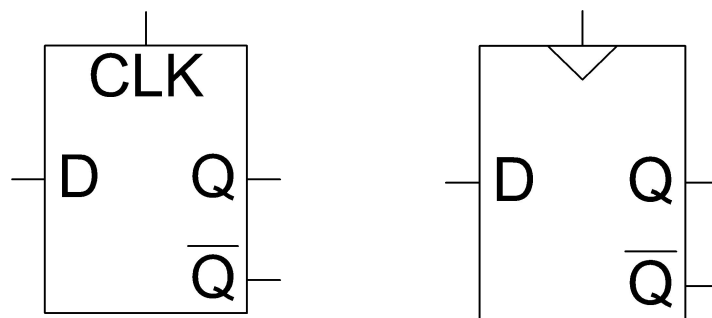
- Когда  $CLK = 1$

- L2 прозрачна
- L1 непрозрачна
- N1 проходит до  $Q$

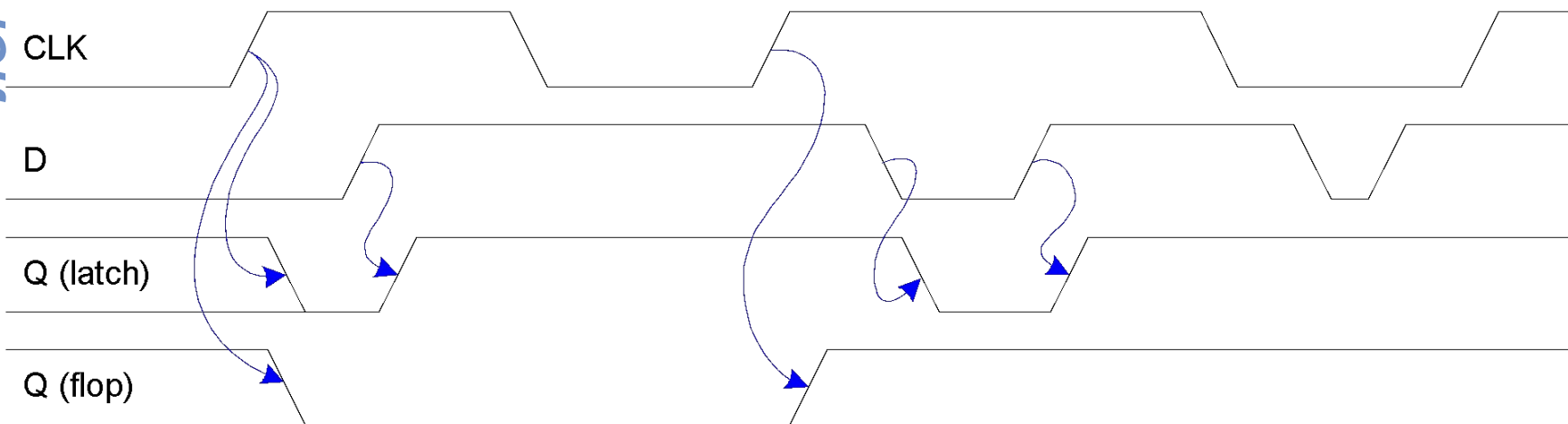
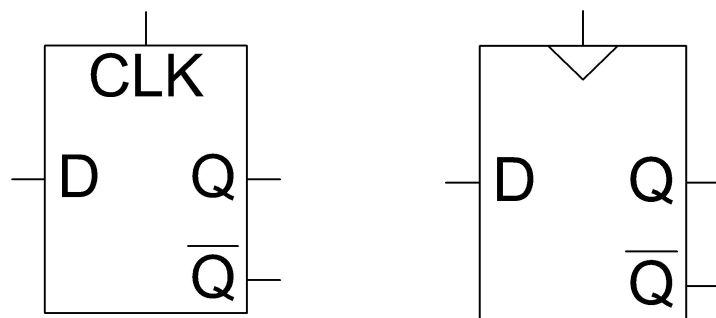
- Следовательно, по фронту тактового сигнала (когда  $CLK$  меняется от 0 к 1)
  - $D$  проходит до  $Q$



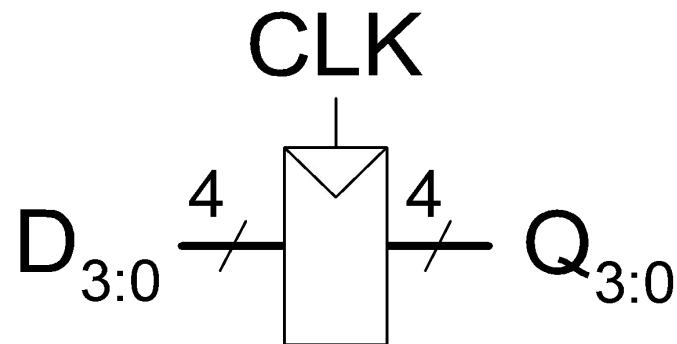
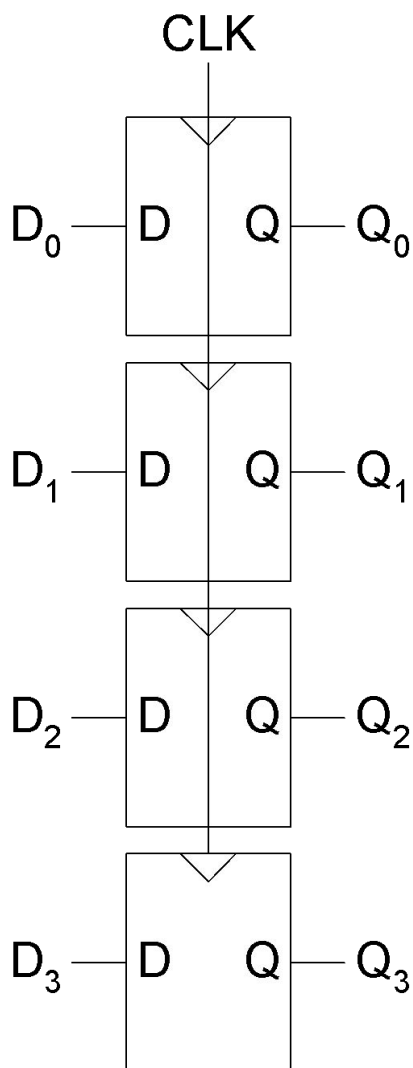
# Сравнение D-защелки и D-триггера



# Сравнение D-защелки и D-триггера



# Регистры



# D-триггер с функцией разрешения

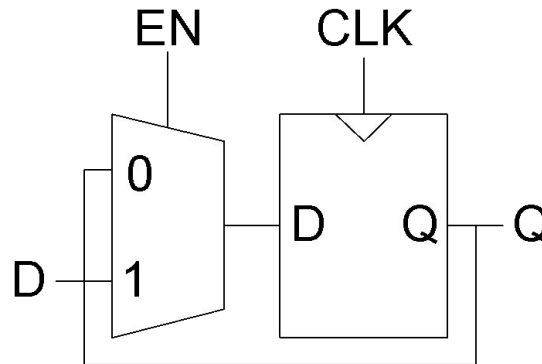
- **Входы:**  $CLK$ ,  $D$ ,  $EN$

- Вход разрешения ( $EN$ ) контролирует, будут ли сохраняться новые данные ( $D$ )

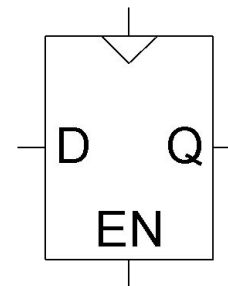
- **Работа**

- $EN = 1$ :  $D$  проходит на  $Q$  по фронту тактового сигнала
- $EN = 0$ : триггер сохраняет предыдущее состояние

Internal  
Circuit



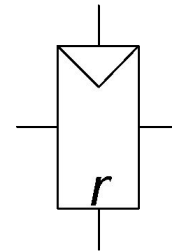
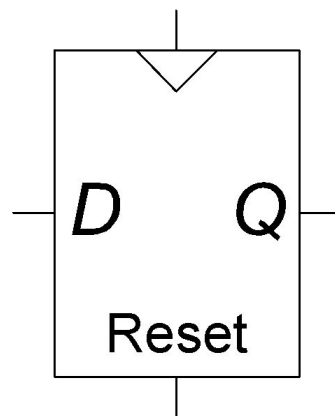
Symbol



# D-триггер с функцией сброса

- **Входы:**  $CLK$ ,  $D$ ,  $Reset$
- **Работа**
  - $Reset = 1$ :  $Q$  устанавливается в 0
  - $Reset = 0$ : Триггер ведет себя как обычный D триггер

## Symbols



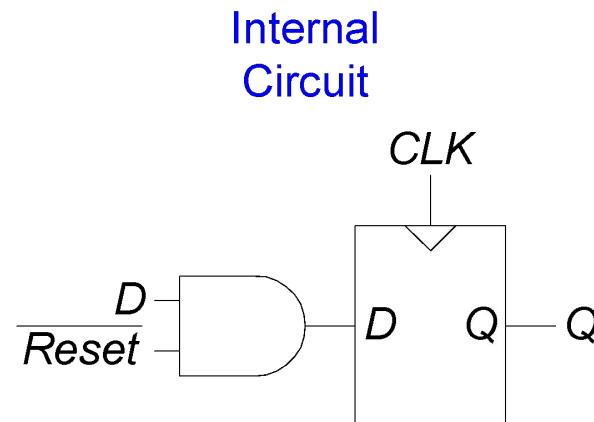


# D-триггер с функцией сброса

- Два типа:
  - **Синхронный:** Сброс происходит только по фронту тактового сигнала
  - **Асинхронный:** сброс происходит сразу после установки *Reset* в 1
- Асинхронно сбрасываемые триггеры требуют отличаются от обычных внутренней структурой
- Триггер с функцией синхронного сброса?

# Триггеры с функцией сброса

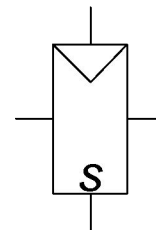
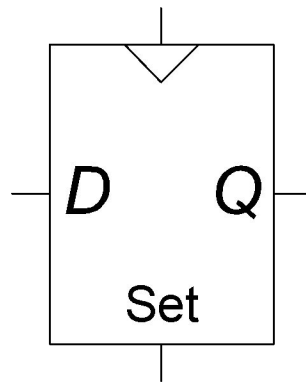
- Два типа:
  - **Синхронный:** Сброс происходит только по фронту тактового сигнала
  - **Асинхронный:** сброс происходит сразу после установки *Reset* в 1
- Асинхронно сбрасываемые триггеры требуют отличаются от обычных внутренней структурой
- Триггер с функцией синхронного сброса?



# Триггеры с функцией

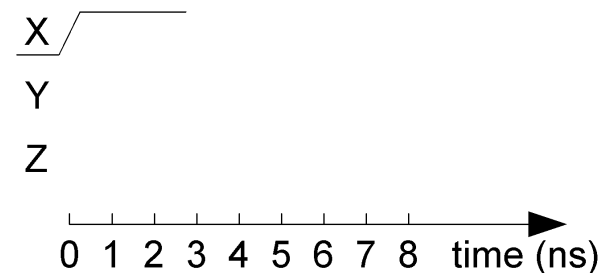
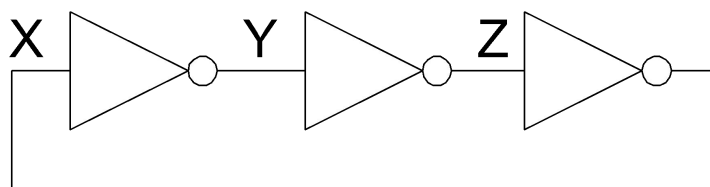
- **Входы:**  $CLK$ ,  $D$ ,  $Set$
- **Работа:**
  - $Set = 1$ :  $Q$  устанавливается в 1
  - $Set = 0$ : Триггер ведет себя как обычный D триггер

## Symbols



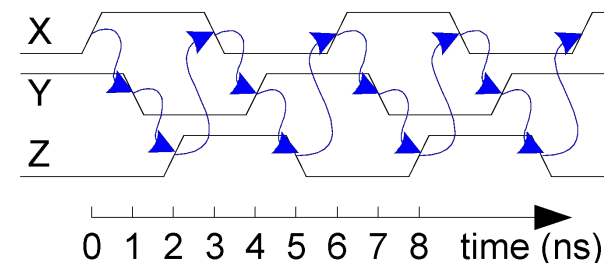
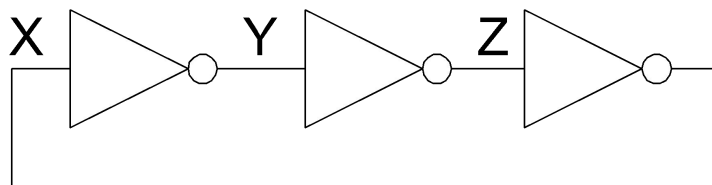
# Последовательные цифровые схемы

- Последовательные схемы - все схемы, которые не являются комбинационными
- Проблемная схема:



# Последовательные цифровые схемы

- Последовательные схемы - все схемы, которые не являются комбинационными
- Проблемная схема:



- Входов нет, выходов 3
- Нестабильная схема, осциллятор
- Период зависит от задержек инверторов
- Схема имеет циклический путь, выход поступает на вход

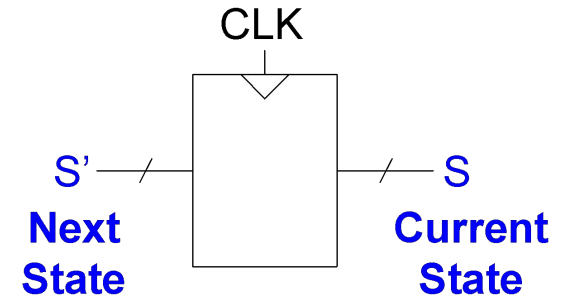
- Разрушение циклических путей с **путем добавления регистров**
- Регистры сохраняет **состояние** схемы
- Состояние изменяется по фронтам тактового сигнала, система **синхронизируется** этим сигналом
- Правила построения синхронных последовательностных схем:
  - Каждый элемент схемы является либо регистром, либо комбинационной схемой
  - Как минимум один элемент схемы является регистром
  - Все регистры тактируются единственным тактовым сигналом
  - В каждом циклическом пути присутствует как минимум один регистр
- Два основные типа синхронных последовательностных схем
  - Конечные автоматы (FSM)
  - Конвейеры

# Конечные автоматы

- Состоят из:

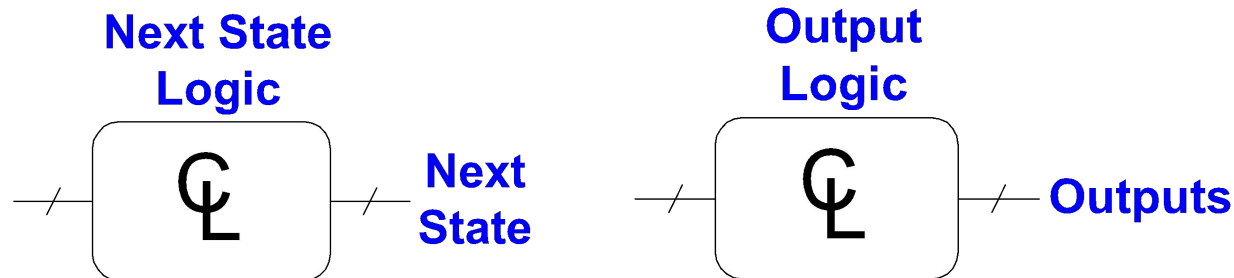
## – Регистра состояний

- Хранит текущее состояние
- По фронту тактового импульса загружает следующее состояние



## – Комбинационной логической схемы

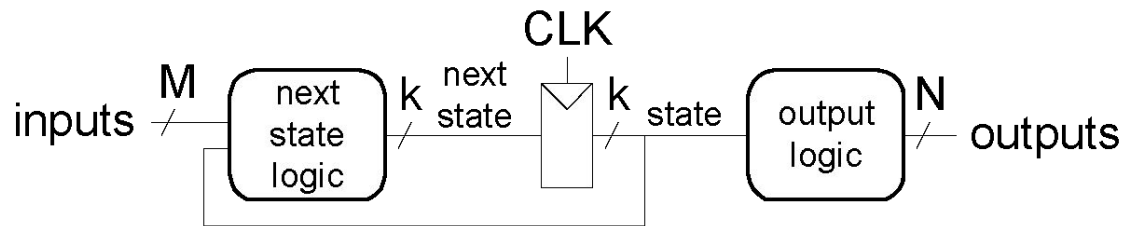
- Определяет следующее состояние
- Определяет выходные сигналы



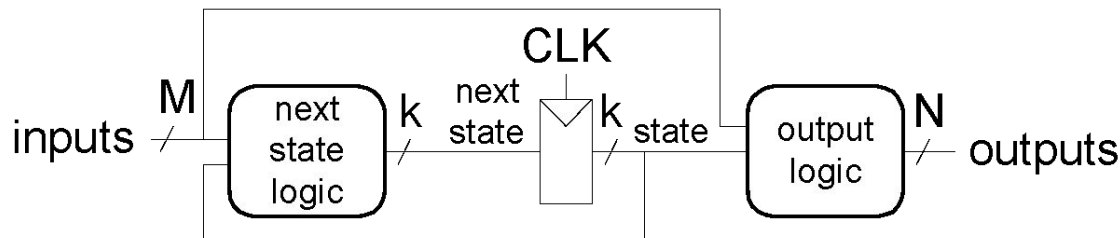
# Конечные автоматы

- Следующее состояние определяется входами и текущим состоянием
- Два класса конечных автоматов отличаются логикой определения выходных сигналов:
  - **Конечные автоматы Мура:** Выход определяется только текущим состоянием
  - **Конечный автомат Мили:** Выход определяется текущим состоянием и входами

## Moore FSM



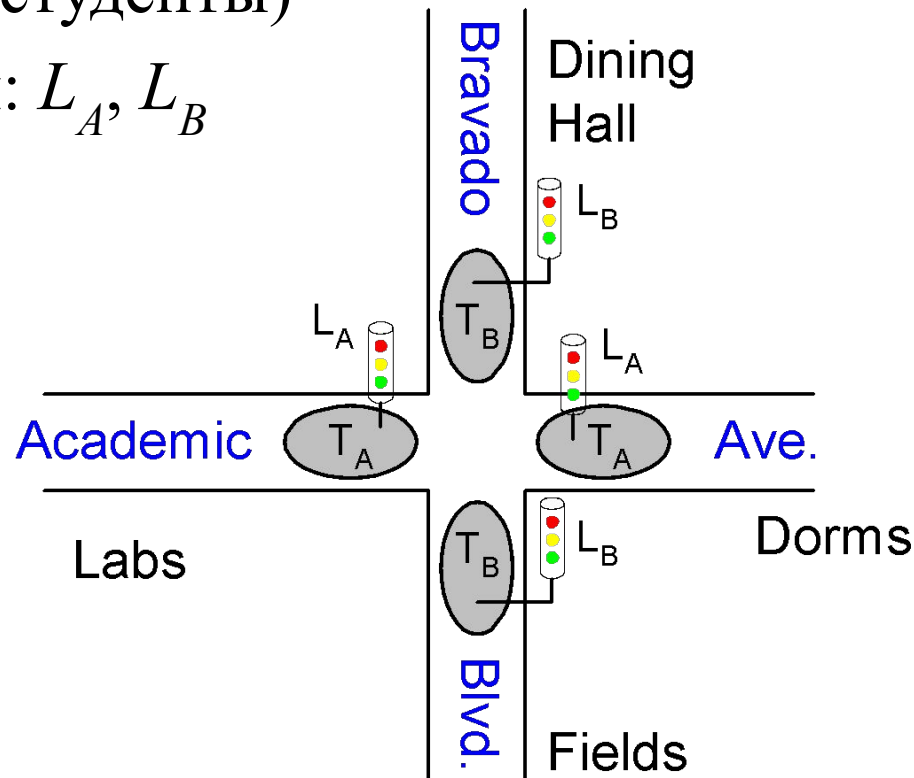
## Mealy FSM





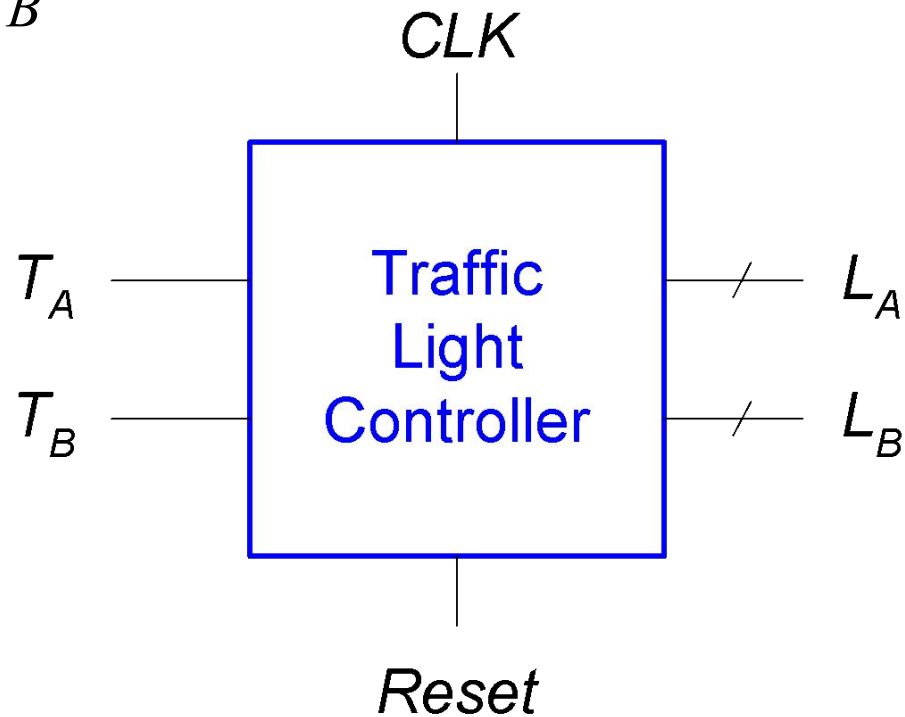
# Пример конечного автомата

- Система управления светофором
  - Датчики движения:  $T_A$ ,  $T_B$  (ИСТИНА, когда на улице есть студенты)
  - Светофоры:  $L_A$ ,  $L_B$



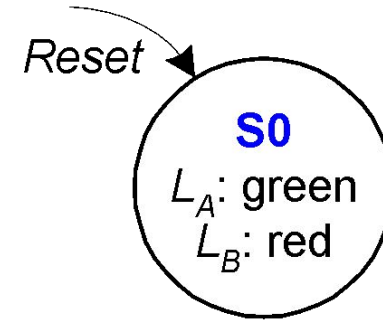
# Черный ящик конечного автомата

- Входы:  $CLK$ ,  $Reset$ ,  $T_A$ ,  $T_B$
- Выходы:  $L_A$ ,  $L_B$



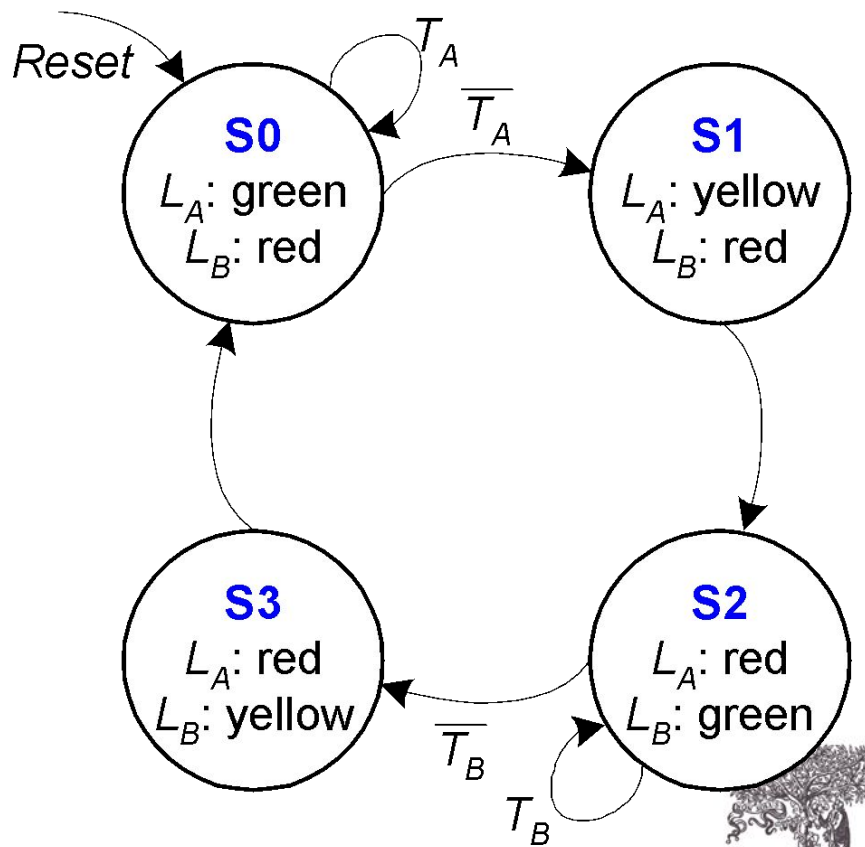
# Диаграмма переходов конечного автомата

- **Конечный автомат Мура:** Значения выходов указаны для каждого состояния
- **Состояния:** Кружки
- **Переходы:** Дуги



# Диаграмма переходов конечного автомата

- **Конечный автомат Мура:** Значения выходов указаны для каждого состояния
- **Состояния:** Кружки
- **Переходы:** Дуги



# Таблица переходов конечного автомата

Текущее состояние	Входы		Следующее состояние
	$T_A$	$T_B$	
$S$			$S'$
S0	0	X	
S0	1	X	
S1	X	X	
S2	X	0	
S2	X	1	
S3	X	X	

# Таблица переходов конечного автомата

Текущее состояние	Входы		Следующее состояние
	$T_A$	$T_B$	
$S$			$S'$
S0	0	X	S1
S0	1	X	S0
S1	X	X	S2
S2	X	0	S3
S2	X	1	S2
S3	X	X	S0

# Таблица переходов конечного автомата с указанием кодирования

Текущее состояние		Входы		Следующее состояние	
$S_1$	$S_0$	$T_A$	$T_B$	$S_1$	$S_0$
0	0	0	X		
0	0	1	X		
0	1	X	X		
1	0	X	0		
1	0	X	1		
1	1	X	X		

Состояние	Кодирование
S0	00
S1	01
S2	10
S3	11



# Таблица переходов конечного автомата с указанием кодирования

Текущее состояние		Входы		Следующее состояние	
$S_1$	$S_0$	$T_A$	$T_B$	$S_1$	$S'_0$
0	0	0	X	0	1
0	0	1	X	0	0
0	1	X	X	1	0
1	0	X	0	1	1
1	0	X	1	1	0
1	1	X	X	0	0

Состояние	Кодирование
S0	00
S1	01
S2	10
S3	11

$$S'_1 = S_1 \oplus S_0$$

$$S'_0 = \overline{S_1} \overline{S_0} \overline{T_A} + S_1 \overline{S_0} \overline{T_B}$$





# Таблица выходов конечного автомата

Текущее состояние		Выходы			
$S_1$	$S_0$	$L_{A1}$	$L_{A0}$	$L_{B1}$	$L_{B0}$
0	0				
0	1				
1	0				
1	1				

Выходы	Кодирование
зеленый	00
желтый	01
красный	10

# Таблица выходов конечного автомата

Текущее состояние		Выходы			
$S_1$	$S_0$	$L_{A1}$	$L_{A0}$	$L_{B1}$	$L_{B0}$
0	0	0	0	1	0
0	1	0	1	1	0
1	0	1	0	0	0
1	1	1	0	0	1

Выходы	Кодирование
зеленый	00
желтый	01
красный	10

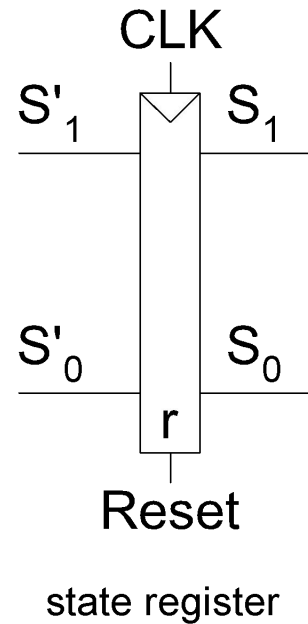
$$L_{A1} = S_1$$

$$L_{A0} = \overline{S_1} S_0$$

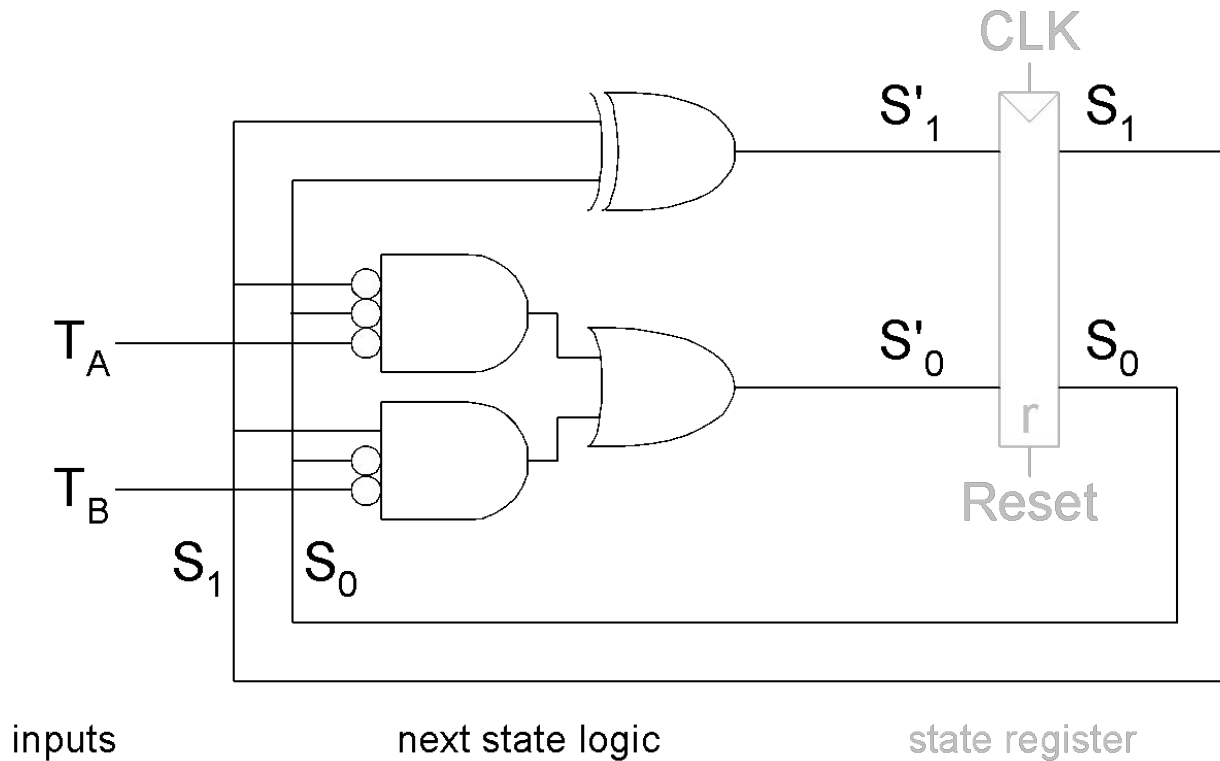
$$L_{B1} = S_1$$

$$L_{B0} = S_1 S_0$$

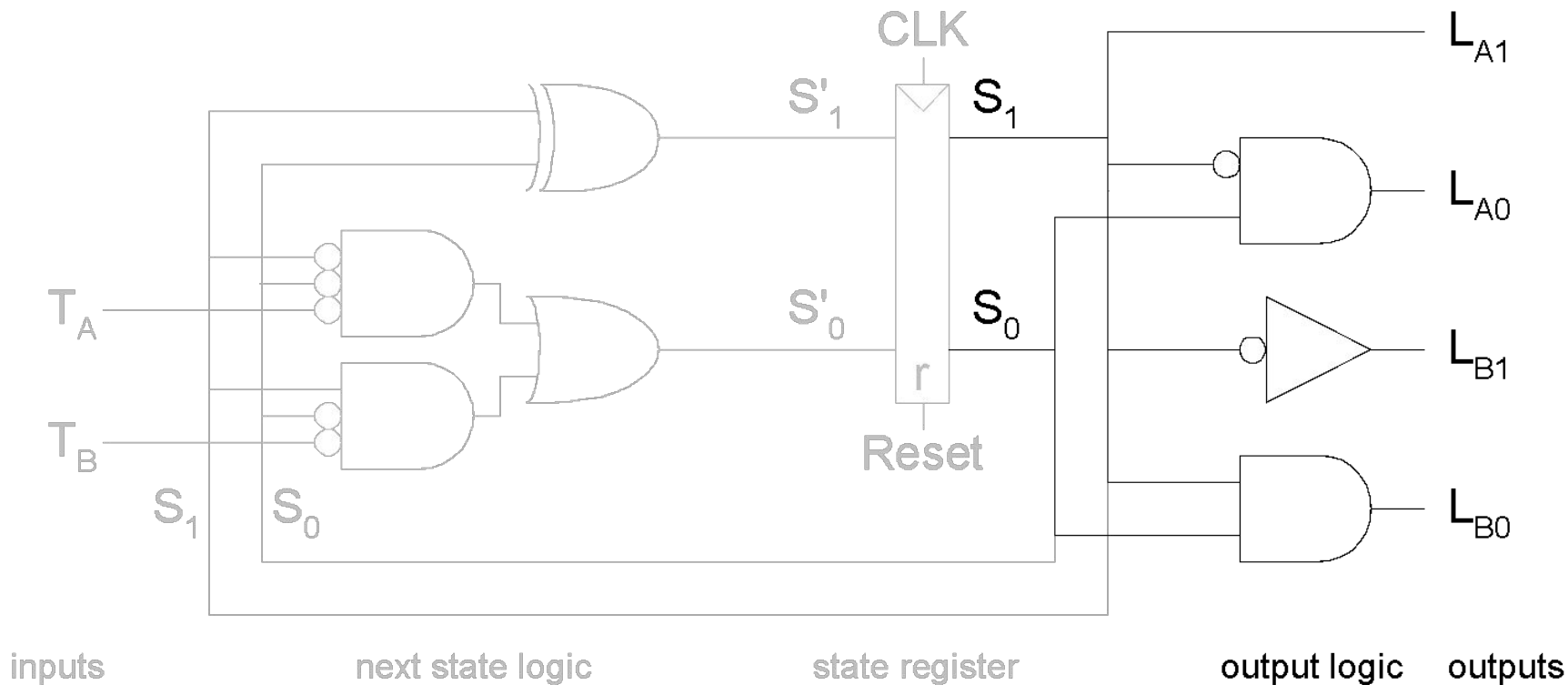
# Схема конечного автомата: Регистр состояний



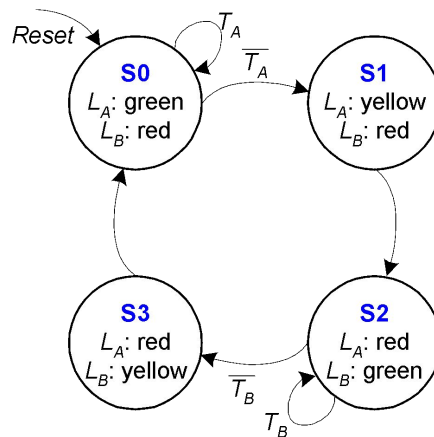
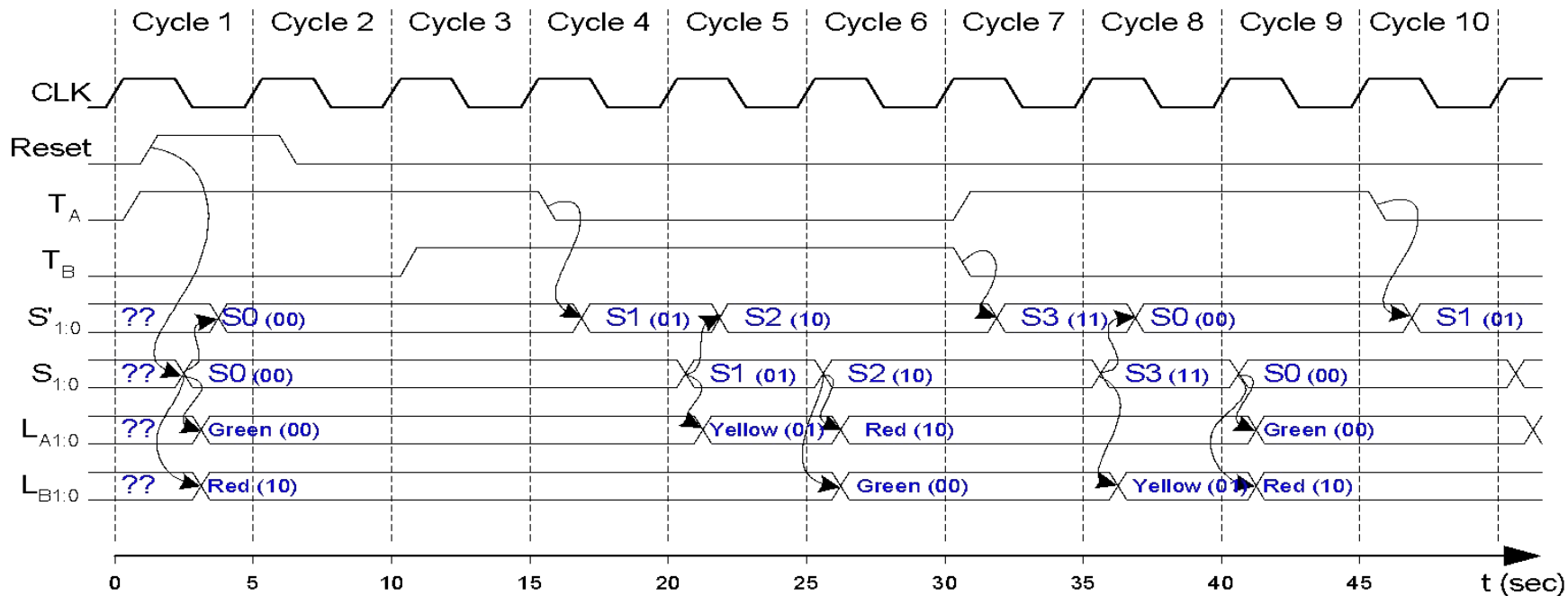
# Схема конечного автомата: Логика следующего состояния



# Схема конечного автомата: Выходная логика



# Временная диаграмма конечного автомата

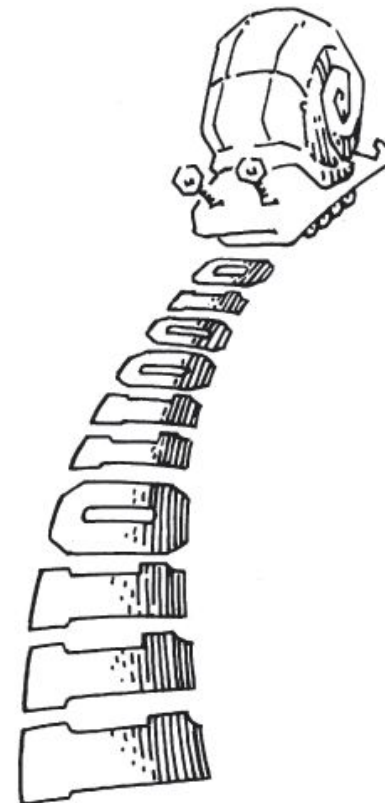


# Кодирование состояний конечного автомата

- **Двоичное** кодирование:
  - Для четырех состояний: 00, 01, 10, 11
- **Прямое (One-hot)** кодирование
  - На каждое состояние один бит
  - Только один бит имеет значение ИСТИНА
  - Для четырех состояний: 0001, 0010, 0100, 1000
  - Используется больше триггеров
  - Часто логика следующего состояния и выходная логика проще

# Сравнение конечных автоматов Мура и Мили

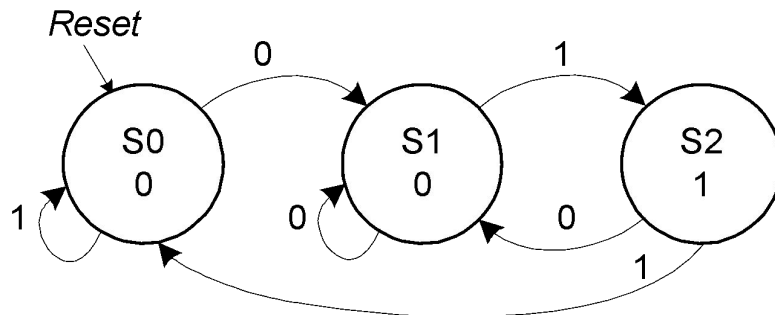
- У Алисы есть улитка, которая ползает по перфоленте, содержащей последовательность нулей и единиц. Улитка улыбается, если последовательность из двух последних бит, через которые она переползла, равна 01. Спроектируйте конечные автоматы Мура и Мили, определяющие, когда улитке нужно улыбнуться.



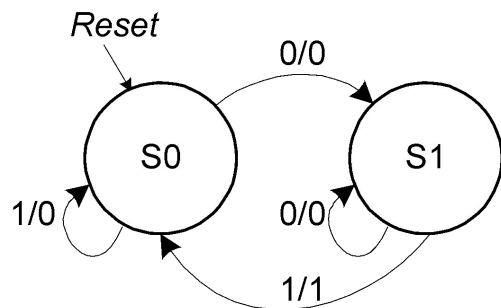


# Диаграмма переходов

## Moore FSM



## Mealy FSM



Конечный автомат Мили: около дуг указаны значения входов/выходов

# Таблица переходов конечного автомата Мура

Текущее состояние		Входы	Следующее состояние	
$S_1$	$S_0$		$S'_1$	$S'_0$
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		

Состояние	Кодирование
S0	00
S1	01
S2	10



# Таблица переходов конечного автомата Мура

Текущее состояние		Входы	Следующее состояние	
$S_1$	$S_0$		$S'_1$	$S'_0$
0	0	0	0	1
0	0	1	0	0
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0

Состояние	Кодирование
S0	00
S1	01
S2	10

$$S'_1 = S_0 A$$

$$S'_0 = \bar{A}$$

# Таблица выходов конечного автомата Мура

Текущее состояние		Выходы $Y$
$S_1$	$S_0$	
0	0	
0	1	
1	0	

$$Y = S_1$$

# Таблица выходов конечного автомата Мура

Текущее состояние		Выходы
$S_1$	$S_0$	
0	0	0
0	1	0
1	0	1

$$Y = S_1$$

# Таблица переходов и выходов конечного автомата Мили

Текущее состояние	Вход	Следующее состояние	Выход
$S_0$	$A$	$S'_0$	$Y$
0	0		
0	1		
1	0		
1	1		

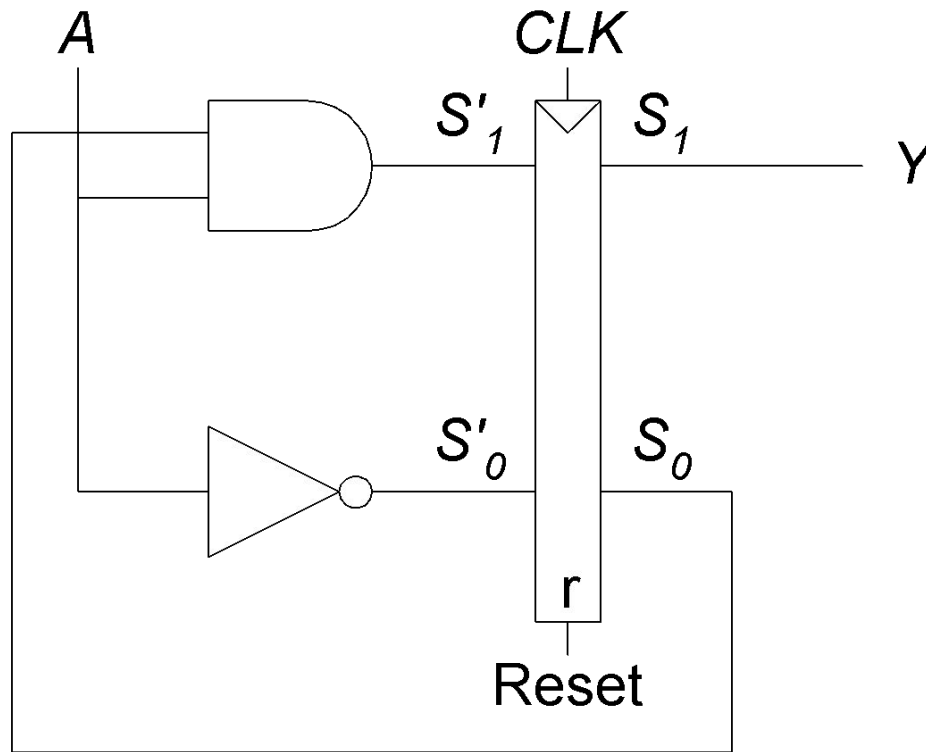
Состояние	Кодирование
S0	00
S1	01

# Таблица переходов и выходов конечного автомата Мили

Текущее состояние	Вход	Следующее состояние	Выход
$S_0$	$A$	$S'_0$	$Y$
0	0	1	0
0	1	0	0
1	0	1	0
1	1	0	1

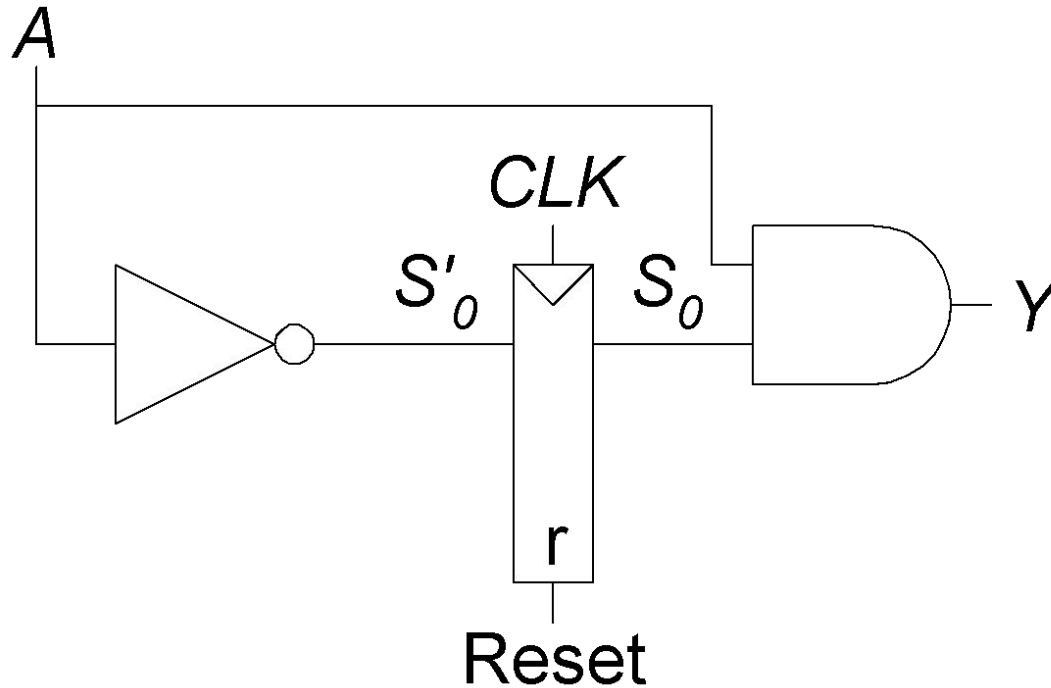
Состояние	Кодирование
S0	00
S1	01

# Схема конечного автомата Мура

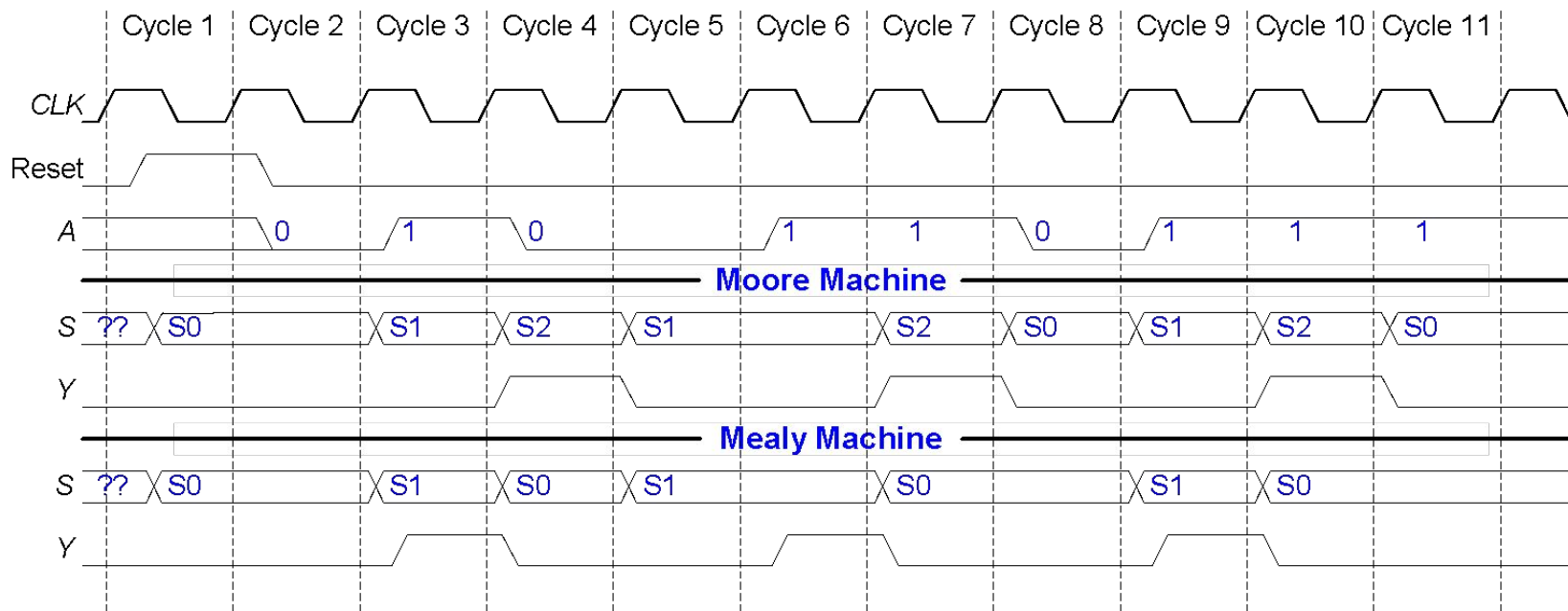




# Схема конечного автомата Мили



# Временные диаграммы конечных автоматов Мура и Мили

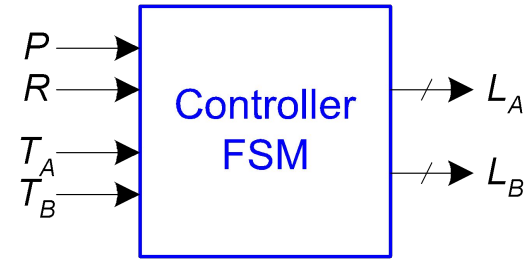


# Декомпозиция конечных автоматов

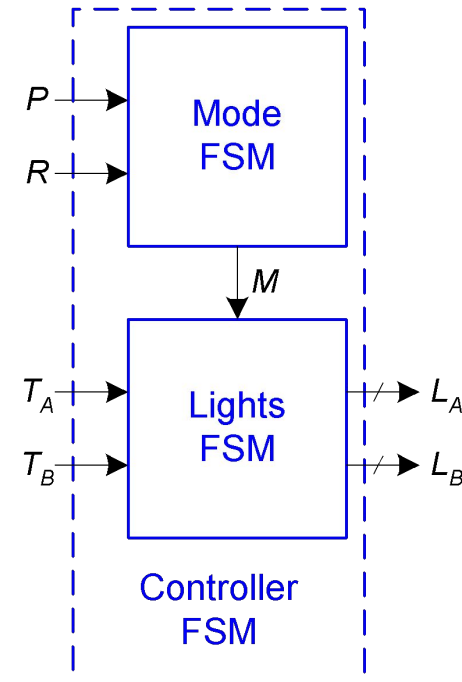
- Разделение сложного конечного автомата на более простые взаимодействующие конечный автоматы
- Пример: Модифицируйте контроллер светофора так, чтобы в нем появился режим «парада»
  - Два дополнительные входа  $P$ ,  $R$
  - Когда  $P = 1$ , автомат входит в режим парада и светофор на Беговой улице остается зеленым
  - Когда  $R = 1$ , автомат выходит из режима парада

# Модифицированный конечный автомат

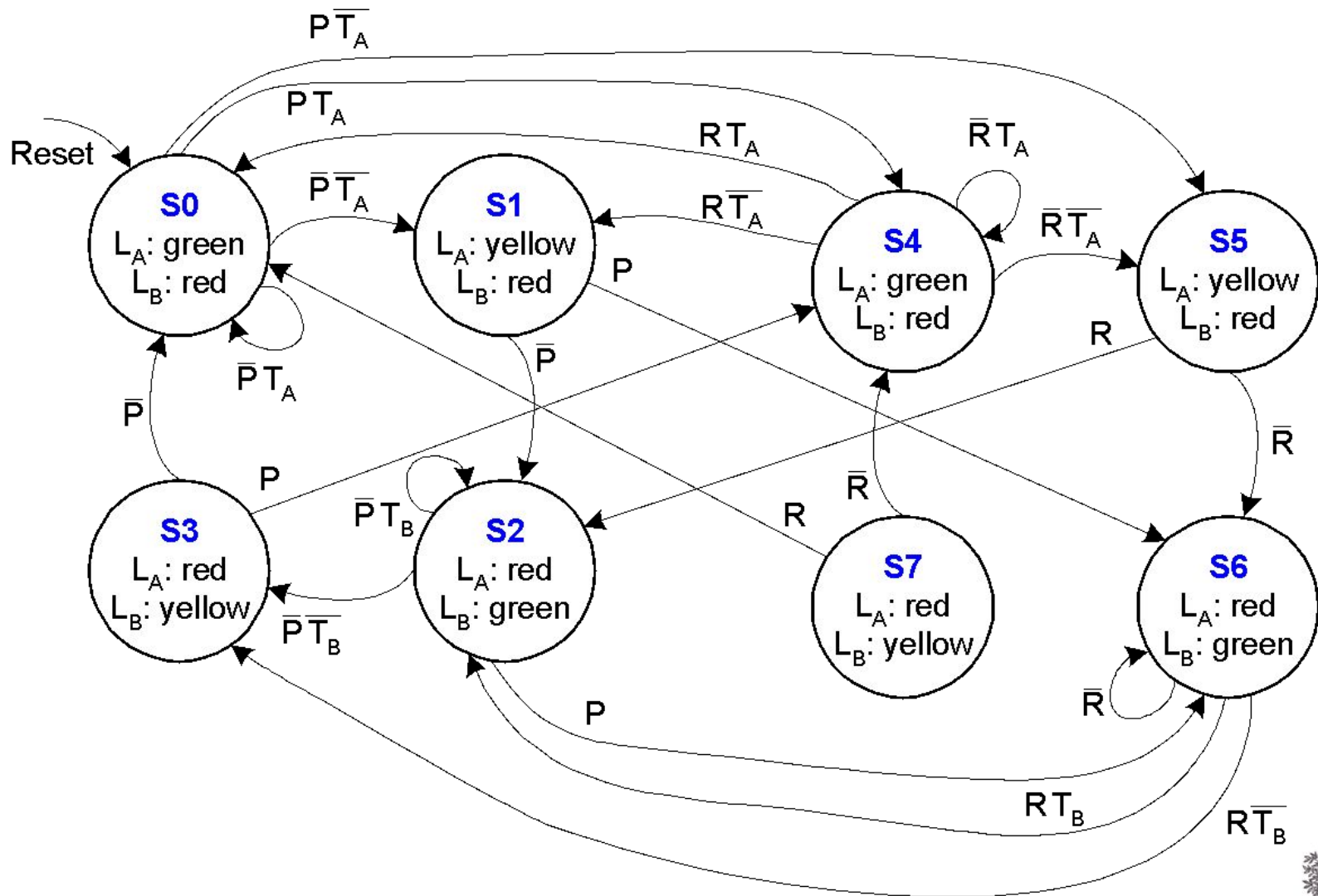
Немодульный  
конечный автомат



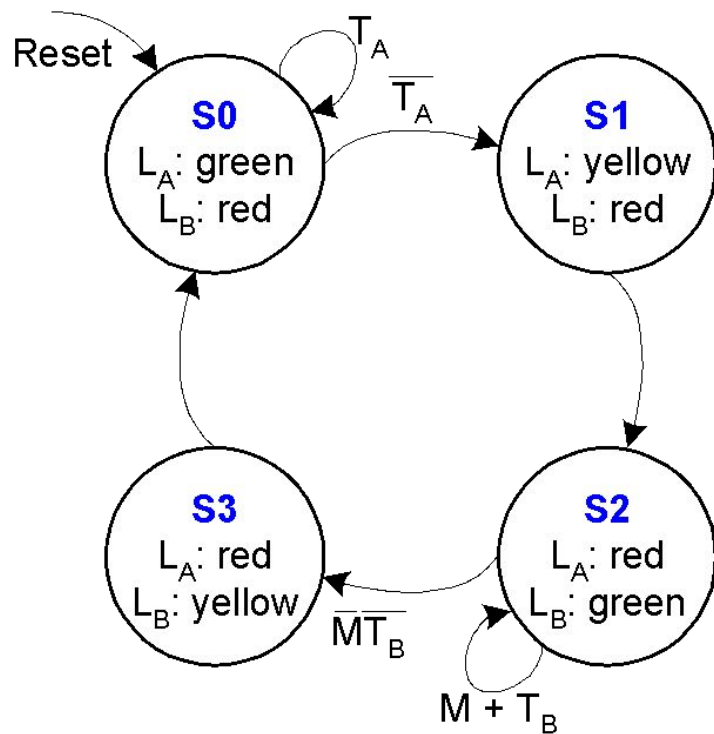
Модульный  
конечный автомат



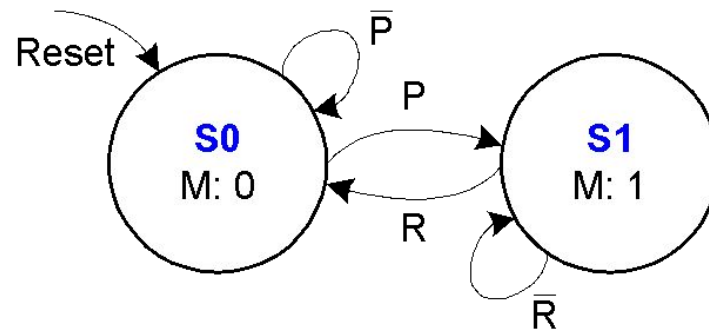
# Немодульный конечный автомат



# Модульный конечный



Lights FSM



Mode FSM

# Проектирование конечного

## автомата

- Определите входы и выходы
- Нарисуйте диаграмму переходов
- Запишите таблицу переходов
- Выберите способ кодирования состояний
- Для конечного автомата Мура:
  - Перепишите таблицу переходов с учетом кодирования состояний
  - Запишите таблицу выходов
- Для конечного автомата Мили:
  - Перепишите объединенную таблицу переходов и выходов с учетом кодирования состояний
- Запишите булевы выражения логики следующего состояния и выходов
- Нарисуйте принципиальную схему

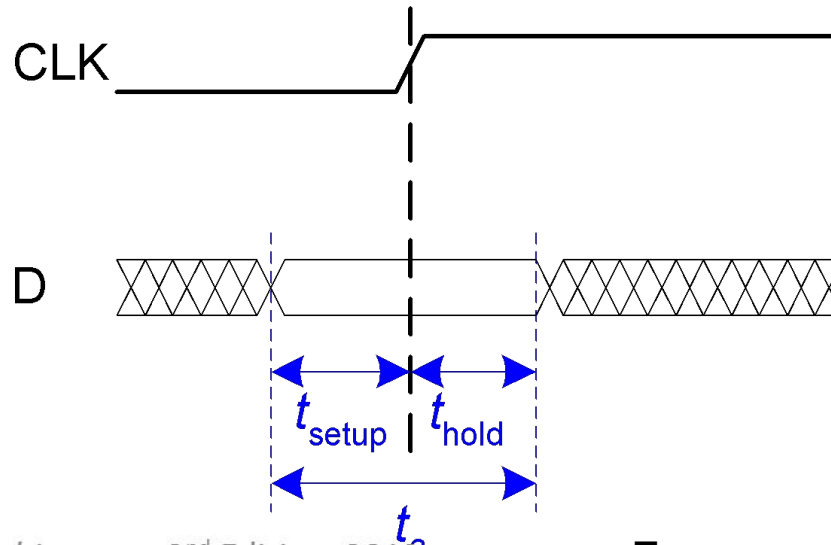
# Синхронизация

- Триггер фиксирует сигнал  $D$  по фронту тактового сигнала
- Сигнал  $D$  должен быть стабильным в процессе фиксации
- Как при фотографировании,  $D$  должен быть стабильным в окрестности фронта тактового сигнала
- Если это не так, может возникнуть метастабильность



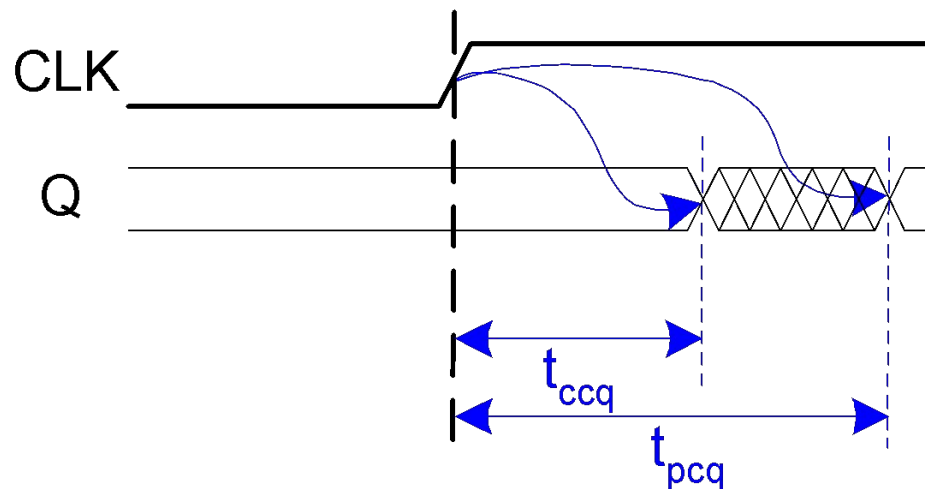
# Ограничение времени изменения входов

- **Время предустановки:**  $t_{\text{setup}}$  = период времени *перед* фронтом тактового сигнала, в течении которого данные должны быть стабильными (т.е. не изменяться)
- **Время удержания:**  $t_{\text{hold}}$  = период времени *после* фронта тактового сигнала, в течении которого данные должны быть стабильными
- **Апертурное время:**  $t_a$  = общее время в окрестности фронта тактового сигнала, в течении которого данные должны быть стабильными ( $t_a = t_{\text{setup}} + t_{\text{hold}}$ )



# Ограничение времени изменения ВЫХОДОВ

- **Задержка распространения  $t_{pcq}$**  = период времени после фронта тактового сигнала, после окончания которого выход  $Q$  будет гарантированно стабильным (т.е., прекратит изменяться)
- **Задержка реакции  $t_{ccq}$**  = период времени после фронта тактового сигнала, после окончания которого  $Q$  может быть нестабильным (т.е., начать изменяться)

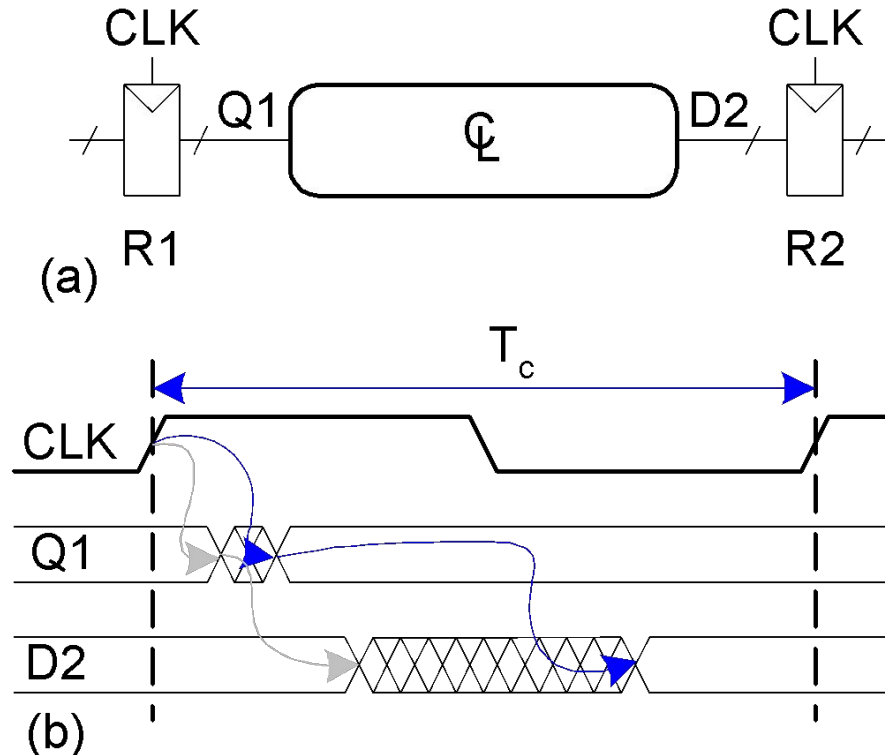


# Динамическая дисциплина

- Входы синхронной последовательностной схемы должны быть стабильны в течение апертурного времени в окрестности фронта тактового сигнала
- Более точно, входы должны быть стабильны
  - по крайней мере в течении времени  $t_{\text{setup}}$  до фронта тактового импульса
  - и по крайней мере в течении времени  $t_{\text{hold}}$  после фронта тактового импульса

# Динамическая дисциплина

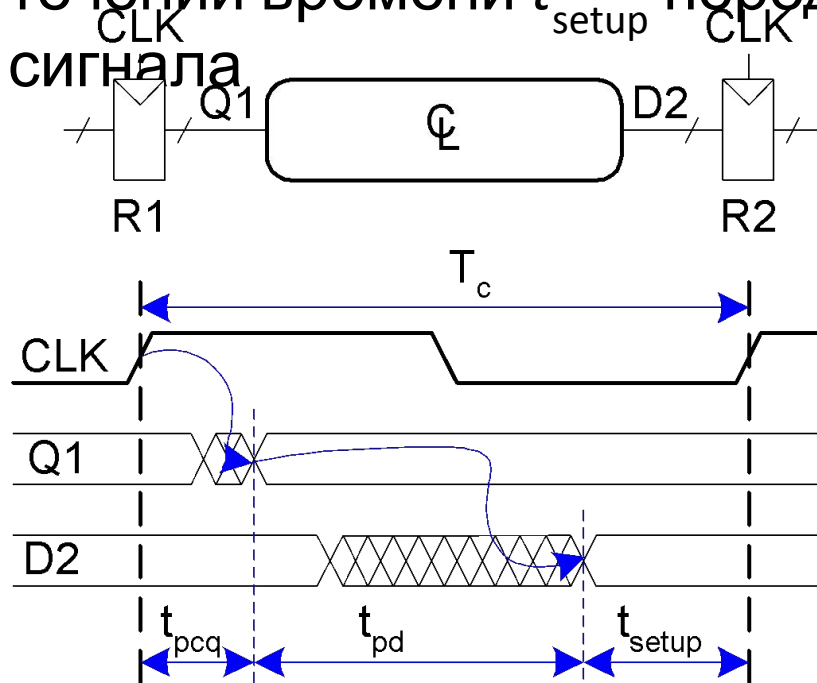
- В зависимости от задержек элементов, общая задержка между регистрами лежит между максимальным и минимальным значениями



# Ограничение времени

## ПРОДУКТОРОВА

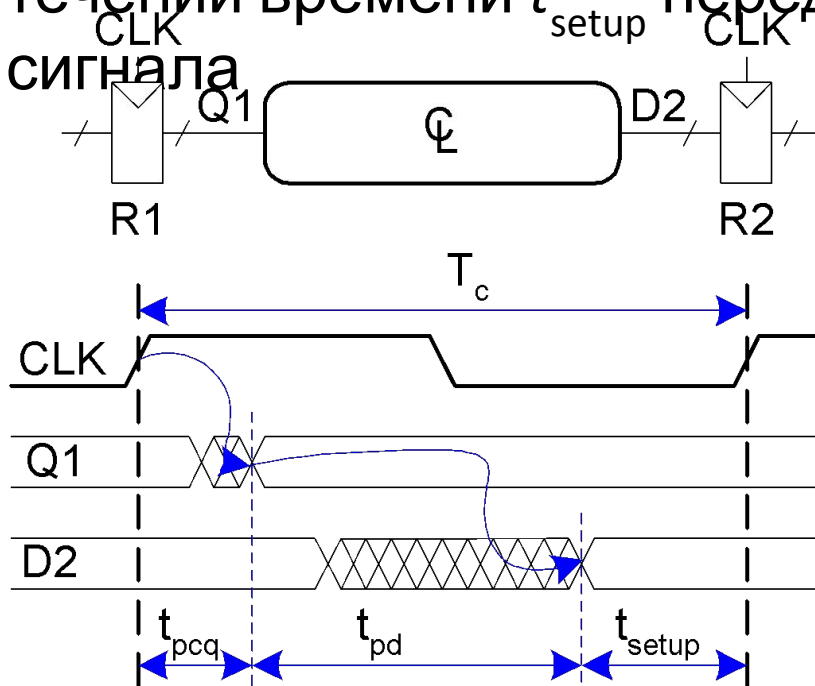
- Зависит от **максимальной** задержки комбинационной логики между регистрами R1 и R2
- Вход регистра R2 должен быть стабильным в течении времени  $t_{setup}$  перед фронтом тактового сигнала



$$T_c \geq$$

# Ограничение времени предустановки

- Зависит от **максимальной** задержки комбинационной логики между регистрами R1 и R2
- Вход регистра R2 должен быть стабильным в течении времени  $t_{setup}$  перед фронтом тактового сигнала

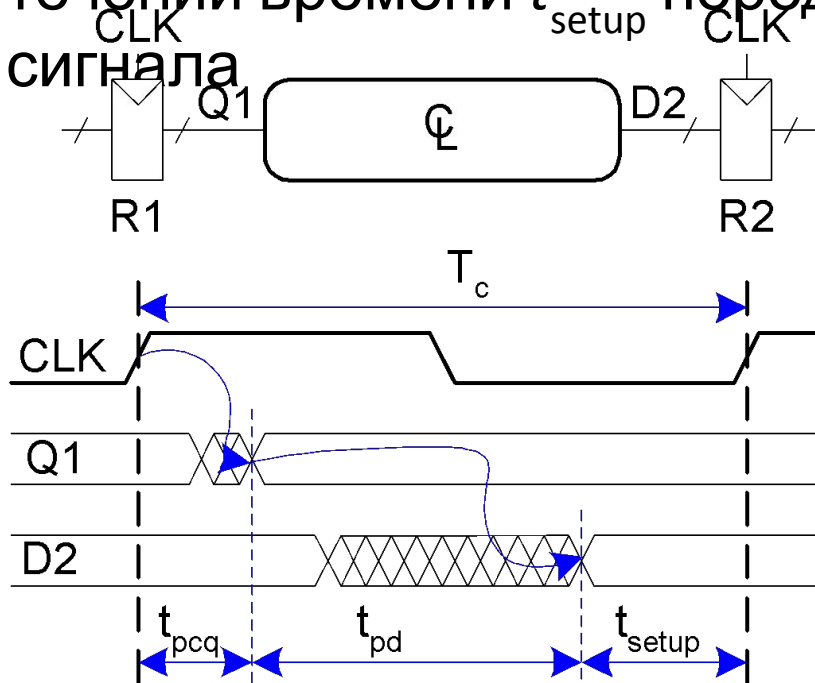


$$T_c \geq t_{pcq} + t_{pd} + t_{setup}$$
$$t_{pd} \leq$$

# Ограничение времени

## продолжительности

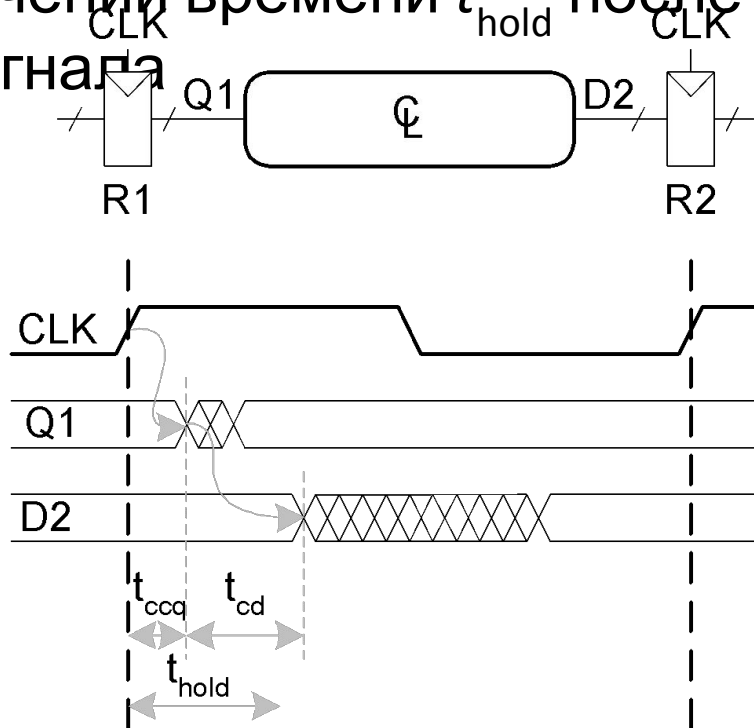
- Зависит от **максимальной** задержки комбинационной логики между регистрами R1 и R2
- Вход регистра R2 должен быть стабильным в течении времени  $t_{setup}$  перед фронтом тактового сигнала



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

# Ограничение времени удержания

- Зависит от **минимальной** задержки комбинационной логики между регистрами R1 и R2
- Вход регистра R2 должен быть стабильным в течении времени  $t_{hold}$  после фронта тактового сигнала

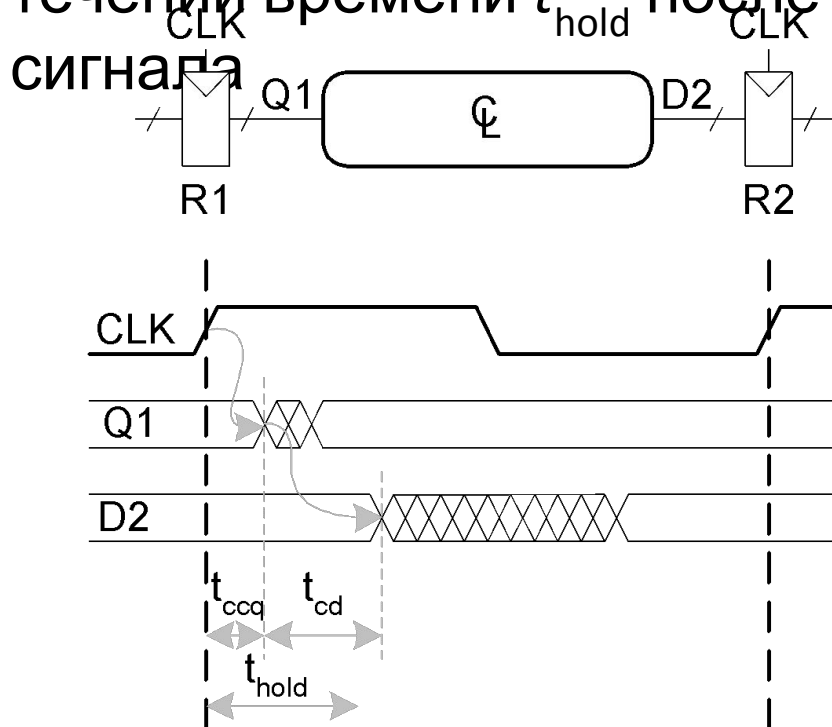


$$t_{hold} <$$



# Ограничение времени удержания

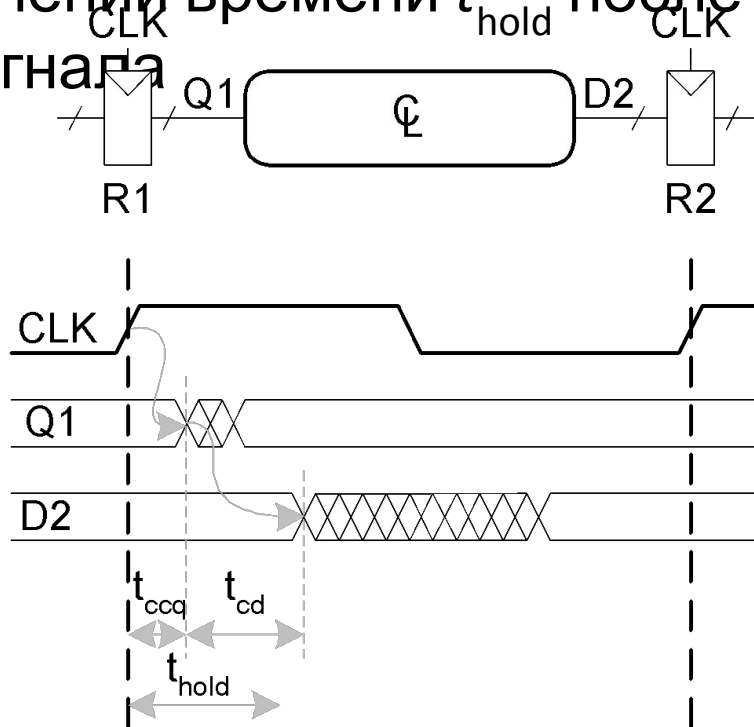
- Зависит от **минимальной** задержки комбинационной логики между регистрами R1 и R2
- Вход регистра R2 должен быть стабильным в течении времени  $t_{hold}$  после фронта тактового сигнала



$$t_{hold} < t_{ccq} + t_{cd}$$
$$t_{cd} >$$

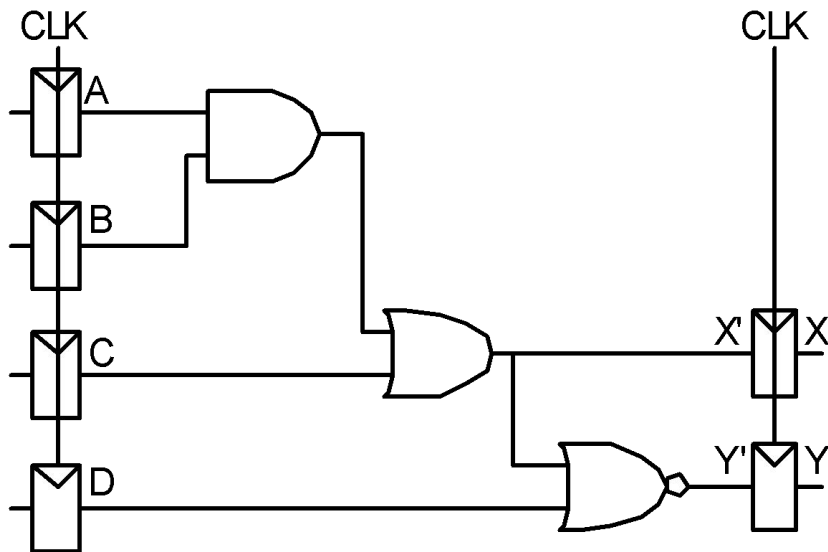
# Ограничение времени удержания

- Зависит от **минимальной** задержки комбинационной логики между регистрами R1 и R2
- Вход регистра R2 должен быть стабильным в течении времени  $t_{hold}$  после фронта тактового сигнала



$$t_{hold} < t_{ccq} + t_{cd}$$
$$t_{cd} > t_{hold} - t_{ccq}$$

# Временной анализ



## Временные характеристики

$$t_{ccq} = 30 \text{ пс}$$

$$t_{pcq} = 50 \text{ пс}$$

$$t_{setup} = 60 \text{ пс}$$

$$t_{hold} = 70 \text{ пс}$$

$$\left. \begin{array}{l} t_{\text{rise}} \\ t_{\text{fall}} \end{array} \right\} = 35 \text{ пс}$$

$$t_{\text{gate}} = 25 \text{ пс}$$

$$t_{pd} =$$

$$t_{cd} =$$

Ограничение времени  
предустановки:

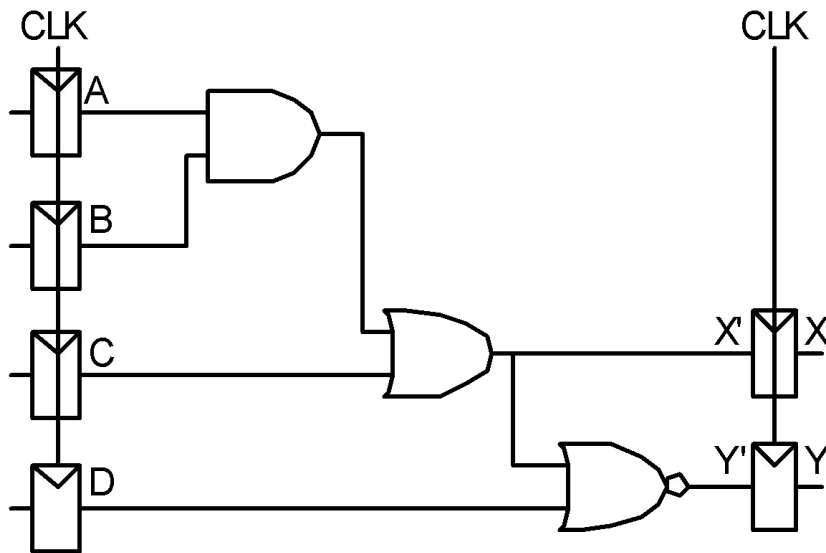
$$T_c \geq$$

$$f_c =$$

Ограничение времени  
удержания:

$$t_{ccq} + t_{cd} > t_{hold} ?$$

# Временной анализ



## Временные характеристики

$$t_{ccq} = 30 \text{ пс}$$

$$t_{pcq} = 50 \text{ пс}$$

$$t_{setup} = 60 \text{ пс}$$

$$t_{hold} = 70 \text{ пс}$$

$$t_{\text{per gate}} = 35 \text{ пс}$$

$$t_{\text{per gate}} = 25 \text{ пс}$$

$$t_{pd} = 3 \times 35 \text{ пс} = 105 \text{ пс}$$

$$t_{cd} = 25 \text{ пс}$$

**Ограничение времени  
предустановки:**

$$T_c \geq (50 + 105 + 60) \text{ пс} = 215 \text{ пс}$$

$$f_c = 1/T_c = 4.65 \text{ ГГц}$$

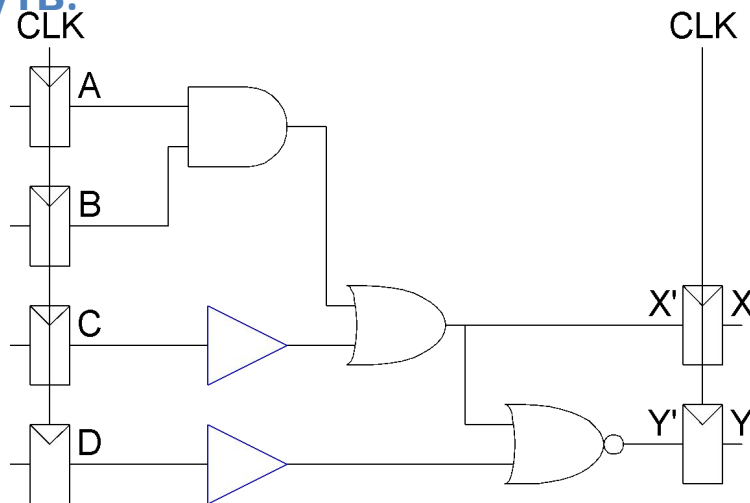
**Ограничение времени  
удержания:**

$$t_{ccq} + t_{cd} > t_{hold} ?$$

$$(30 + 25) \text{ пс} > 70 \text{ пс} ? \text{ Нет!}$$

# Временной анализ

Добавим буфер в кратчайший путь:



$$t_{pd} =$$

$$t_{cd} =$$

Ограничение времени  
предустановки:

$$T_c \geq$$

$$f_c =$$

## Временные характеристики

$$t_{ccq} = 30 \text{ пс}$$

$$t_{pcq} = 50 \text{ пс}$$

$$t_{setup} = 60 \text{ пс}$$

$$t_{hold} = 70 \text{ пс}$$

$$t_{\text{per gate}} = 35 \text{ пс}$$

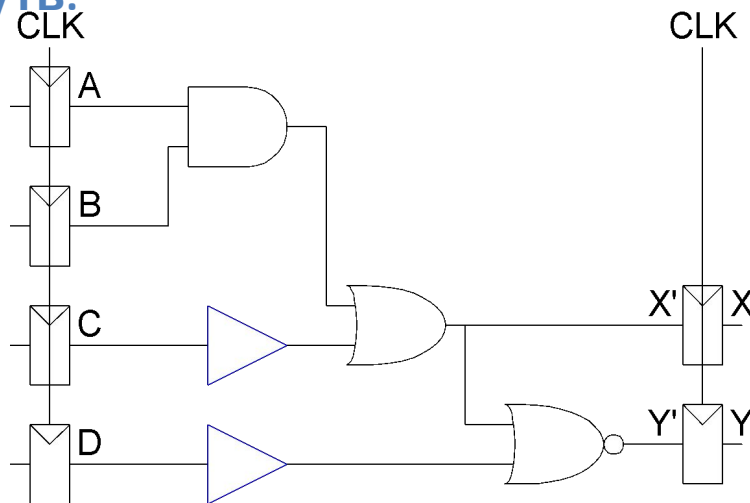
$$t_{\text{per gate}} = 25 \text{ пс}$$

Ограничение времени  
удержания:

$$t_{ccq} + t_{cd} > t_{hold} ?$$

# Временной анализ

Добавим буфер в кратчайший путь:



$$t_{pd} = 3 \times 35 \text{ пс} = 105 \text{ пс}$$

$$t_{cd} = 2 \times 25 \text{ пс} = 50 \text{ пс}$$

Ограничение времени  
предустановки:

$$T_c \geq (50 + 105 + 60) \text{ пс} = 215 \text{ пс}$$

$$f_c = 1/T_c = 4.65 \text{ ГГц}$$

## Временные характеристики

$$t_{ccq} = 30 \text{ пс}$$

$$t_{pcq} = 50 \text{ пс}$$

$$t_{setup} = 60 \text{ пс}$$

$$t_{hold} = 70 \text{ пс}$$

$$t_{\text{per gate}} = 35 \text{ пс}$$

$$t_{\text{buf}} = 25 \text{ пс}$$

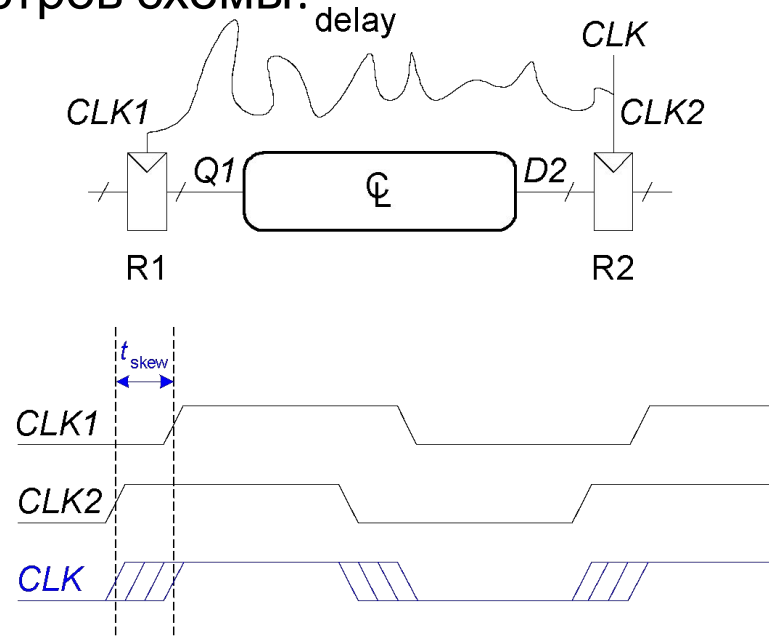
Ограничение времени  
удержания:

$$t_{ccq} + t_{cd} > t_{hold} ?$$

$$(30 + 50) \text{ пс} > 70 \text{ пс} ? \text{ Да!}$$

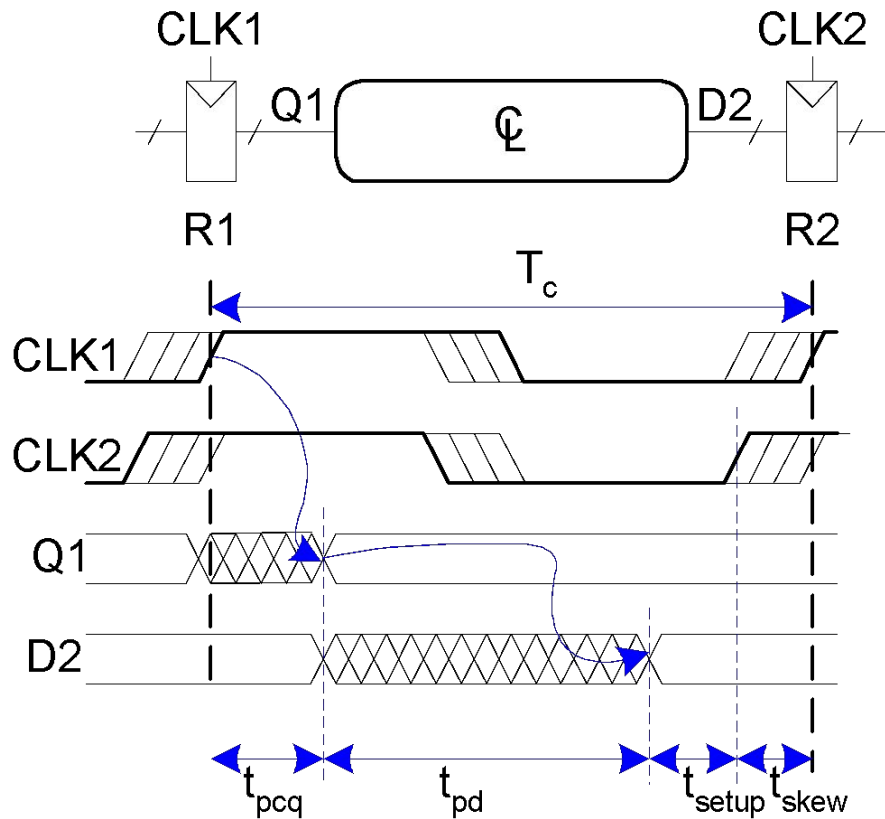
# Расфазировка тактовых сигналов

- Тактовые импульсы поступают на разные регистры схемы не одновременно
- **Расфазировка**: различие во времени между фронтами тактовых сигналов разных элементов
- Следует выполнить **анализ худшего случая**, чтобы гарантировать выполнение динамической дисциплины для всех регистров схемы!



# Время предустановки с учетом расфазировки тактовых импульсов

- В худшем случае CLK2 опережает во времени CLK1



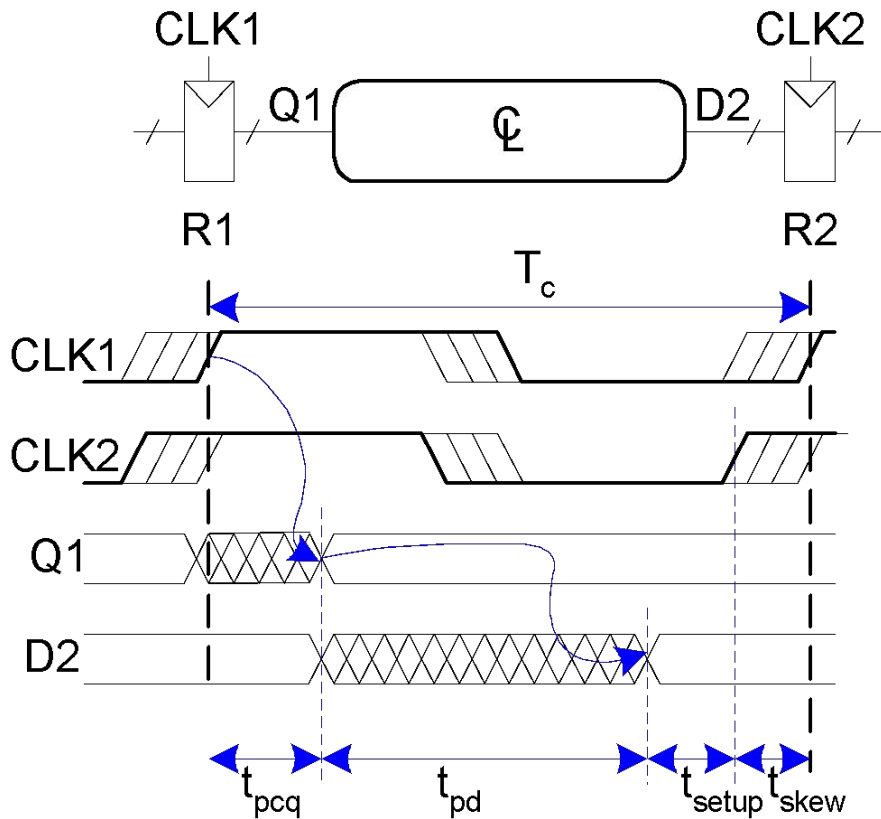
$$T_c \geq$$





# Ограничение времени предустановки с учетом расфазировки тактовых импульсов

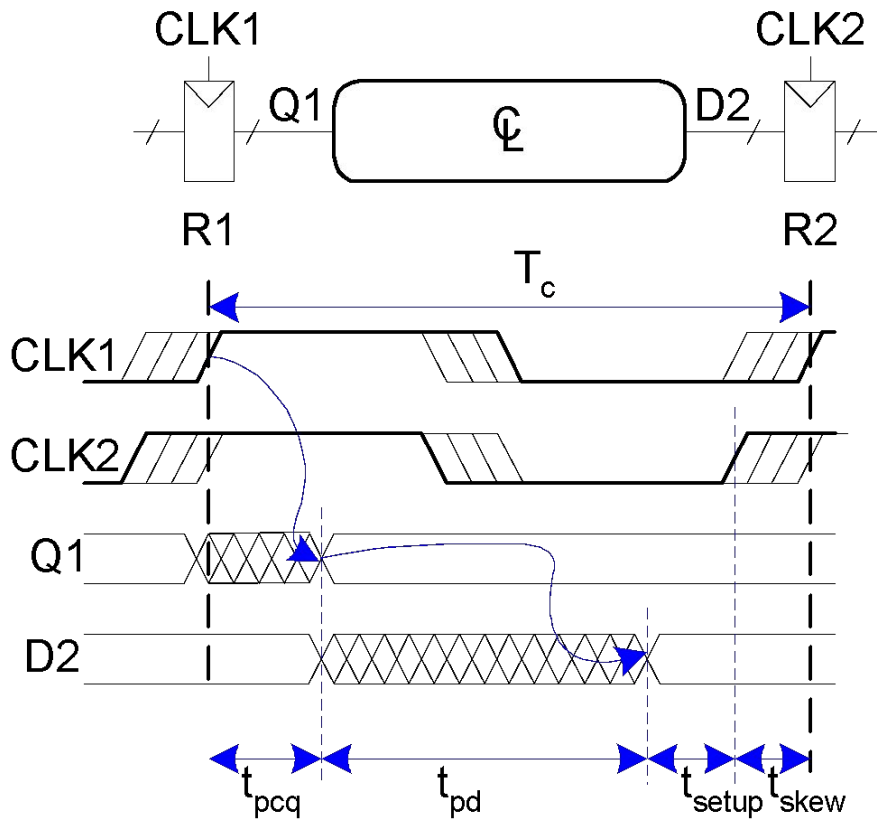
- В худшем случае CLK2 опережает во времени CLK1



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

# Ограничение времени предустановки с учетом расфазировки тактовых импульсов

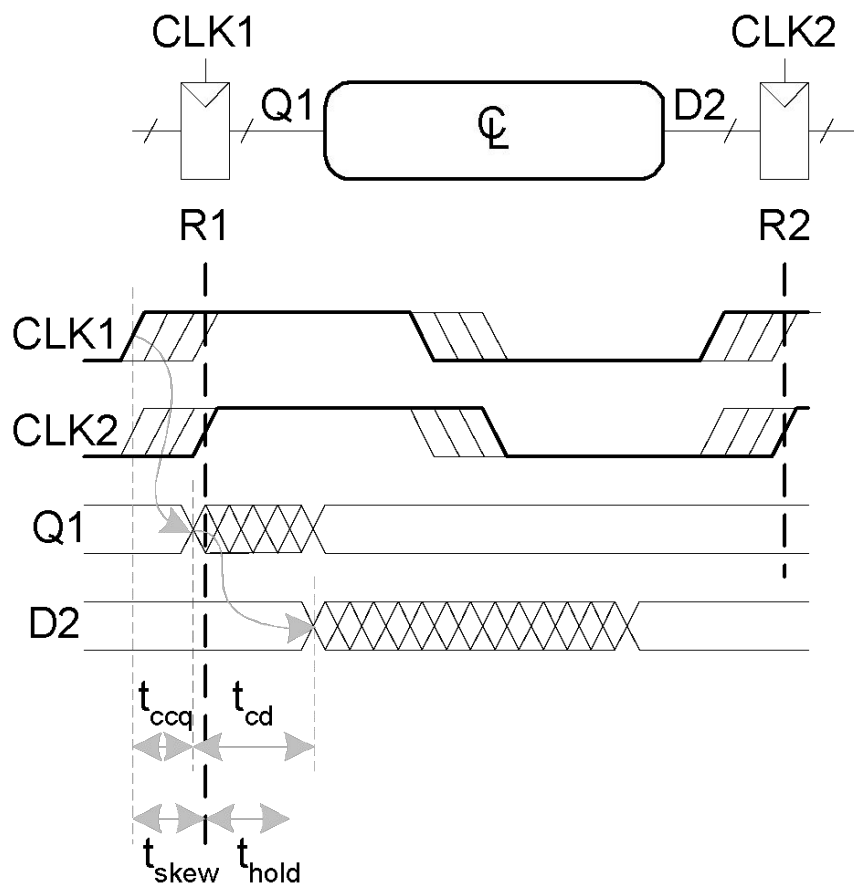
- В худшем случае CLK2 опережает во времени CLK1



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

# Ограничение времени удержания с учетом расфазировки тактовых импульсов

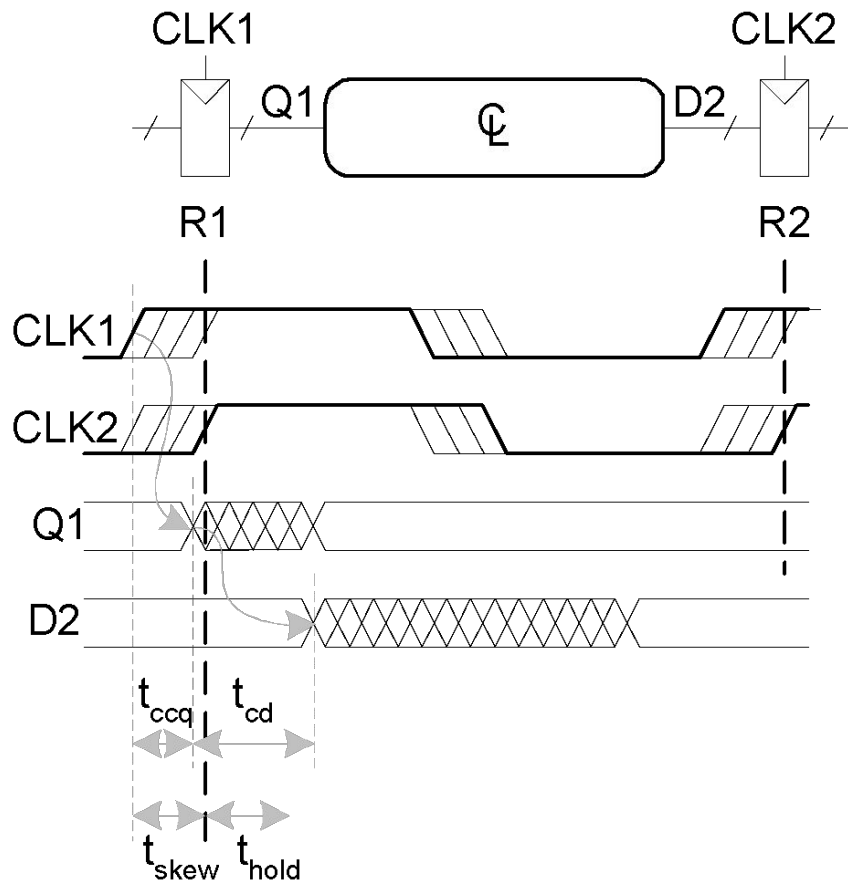
- В худшем случае CLK2 отстает во времени от CLK1



$$t_{ccq} + t_{cd} >$$

# Ограничение времени удержания с учетом расфазировки тактовых импульсов

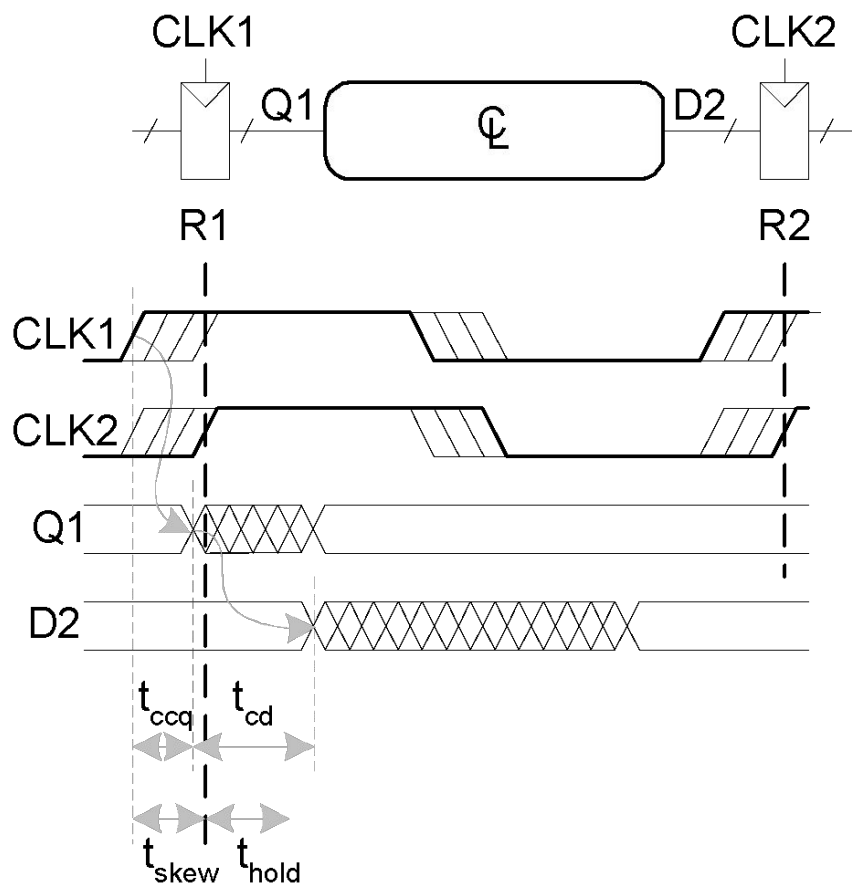
- В худшем случае CLK2 отстает во времени от CLK1



$$t_{ccq} + t_{cd} > t_{hold} + t_{skew}$$
$$t_{cd} >$$

# Ограничение времени удержания с учетом расфазировки тактовых импульсов

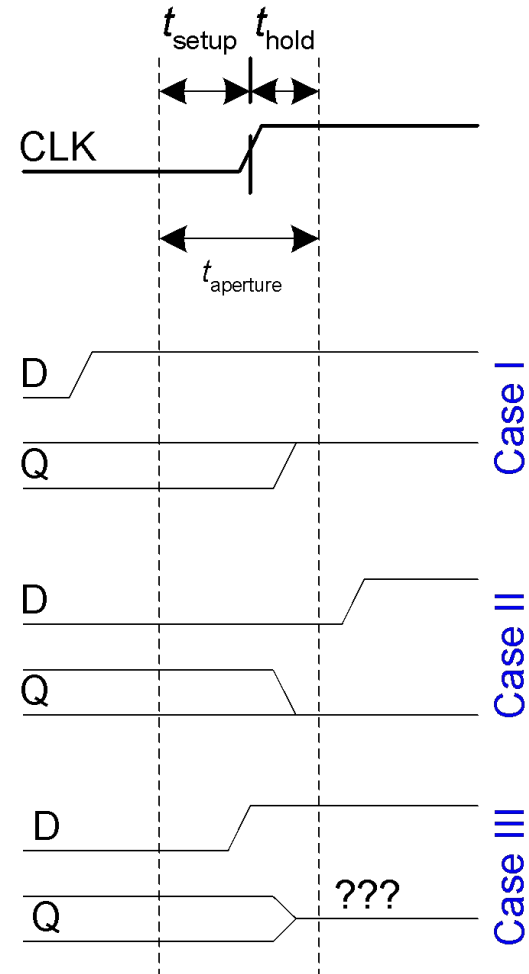
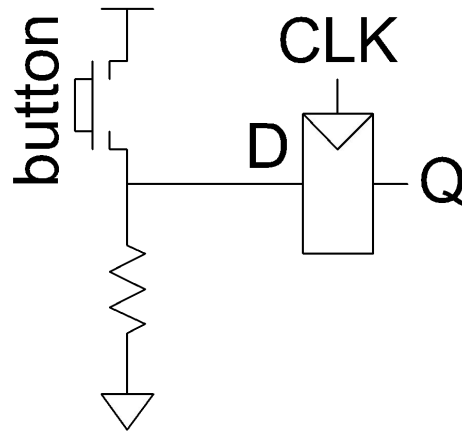
- В худшем случае CLK2 отстает во времени от CLK1



$$t_{ccq} + t_{cd} > t_{hold} + t_{skew}$$
$$t_{cd} > t_{hold} + t_{skew} - t_{ccq}$$

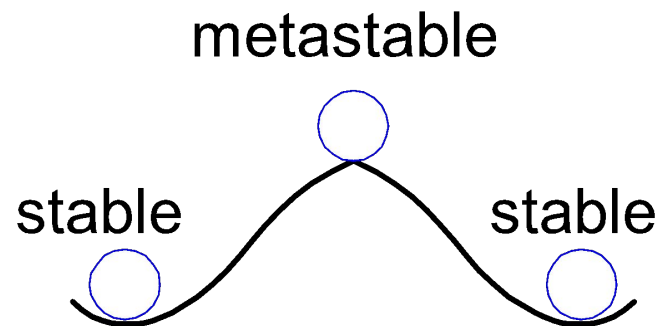
# Нарушение динамической дисциплины

- Асинхронные (например пользовательские) входы могут приводить к нарушению динамической дисциплины



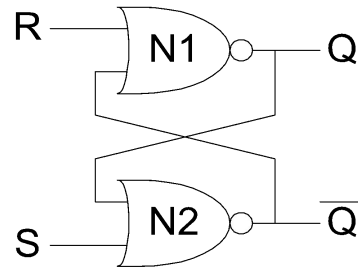
# Метастабильность

- **Бистабильные приборы:** два стабильные состояния и метастабильное состояние между ними
- **Триггер** два стабильные состояния (0 и 1) и одно метастабильное состояние
- Если триггер попадает в метастабильное состояние, он может находится в нем неограниченно долго



# Внутренняя структура

- Триггер имеет **обратную связь**: если  $Q$  находится где-то между 1 и 0, то перекрестно соединенные элементы устанавливают выход в одно из двух состояний (1 или 0)



- Метастабильный сигнал: если он не принял корректное значение 0 или 1
- Если вход триггера изменяется в случайный момент, **вероятность того, что его выход  $Q$  будет в метастабильном состоянии спустя время  $t$ :**

$$P(t_{\text{res}} > t) = (T_0/T_c) e^{-t/\tau}$$

$t_{\text{res}}$  : время разрешения в 1 или 0  $T_0$ ,  $\tau$  : характеристики схемы



# Метастабильность

- **Интуитивно:**

- $T_0/T_c$ : вероятность изменения входа в плохое (апертурное) время

$$P(t_{\text{res}} > t) = (T_0/T_c) e^{-t/\tau}$$

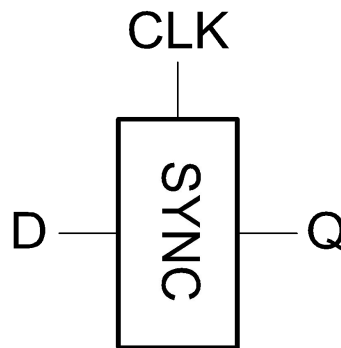
- $\tau$ : константа, определяющая как быстро триггер уходит из метастабильного состояния

$$P(t_{\text{res}} > t) = (T_0/T_c) e^{-t/\tau}$$

- Коротко, если триггер фиксирует метастабильный вход, то после достаточно продолжительного ожидания ( $t$ ) он, с высокой вероятностью, перейдет в стабильное состояние 0 или 1

# Синхронизаторы

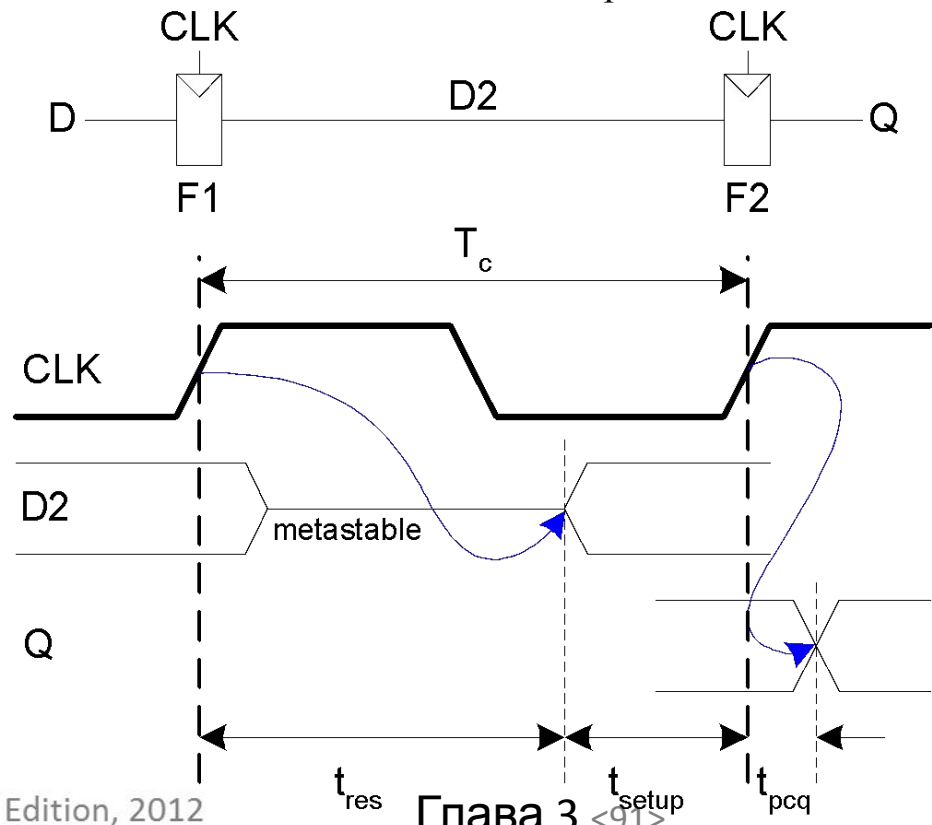
- **Наличие асинхронных входов неизбежно** (интерфейс пользователя, системы с разными тактовыми сигналами и т.д.)
- **Функция синхронизатора:** сделать вероятность сбоя (выход  $Q$  находится в метастабильном состоянии) достаточно малой
- Синхронизатор не может довести вероятность сбоя до 0



# Внутренняя структура

## цифрового

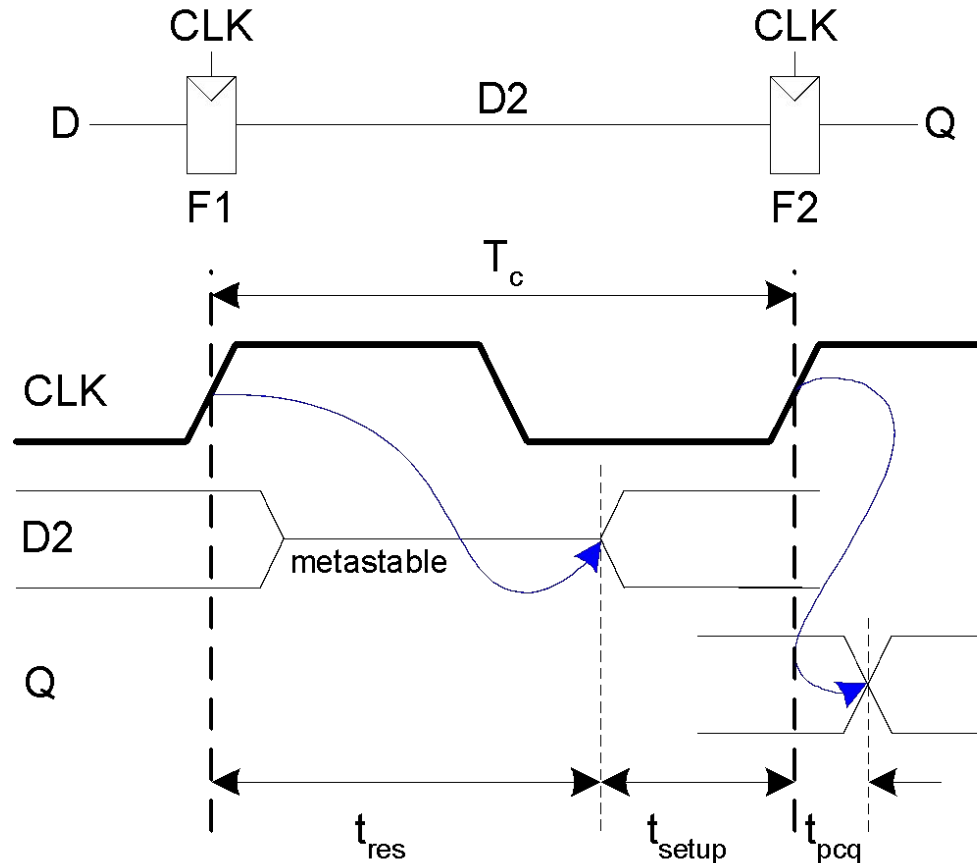
- Синхронизатор: строится из двух последовательно соединенных триггеров
- Предположим, что D изменяется во время фиксации F1
- У внутреннего сигнала D2 есть время  $(T_c - t_{\text{setup}})$  для разрешения в 1 или 0



# Вероятность сбоя

Вероятность сбоя для одиночного изменения входа:

$$P(\text{failure}) = (T_0/T_c) e^{-(T_c - t_{\text{setup}})/\tau}$$



# Среднее время между сбоями синхронизатора

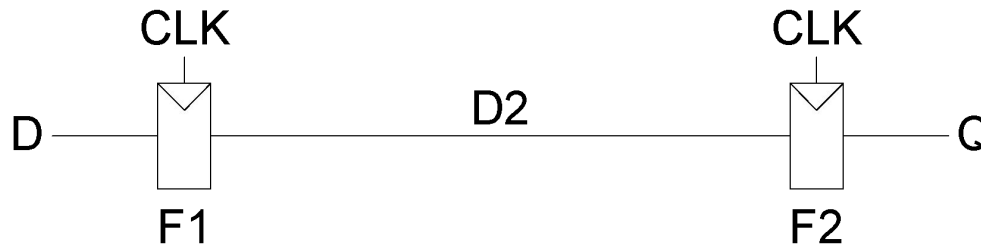
- Если асинхронный вход изменяется один раз за секунду, то вероятность сбоя за одну секунду будет равна  $P(\text{failure})$
- Если вход изменяется  $N$  раз за секунду, то вероятность отказа за секунду будет:

$$P(\text{failure})/c = (NT_0/T_c) e^{-(T_c - t_{\text{setup}})/\tau}$$

- Сбой синхронизатора происходит в среднем  $1/[P(\text{failure})/c]$
- Эта величина называется *средним временем наработки на отказ* (*mean time between failures*, MTBF):

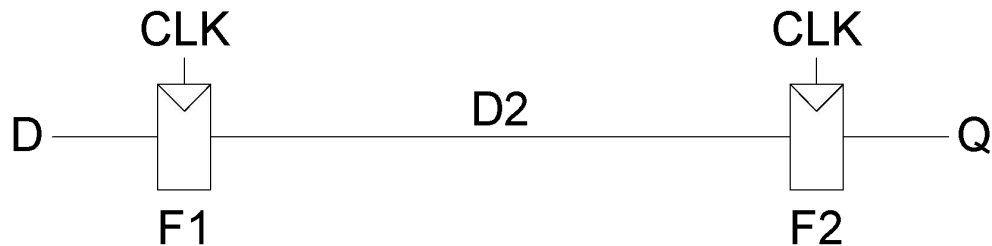
$$\text{MTBF} = 1/[P(\text{failure})/c] = (T_c/NT_0) e^{(T_c - t_{\text{setup}})/\tau}$$

# Пример синхронизатора



- Предположим:  $T_c = 1/500 \text{ МГц} = 2 \text{ нс}$      $\tau = 200 \text{ пс}$   
 $T_0 = 150 \text{ пс}$      $t_{\text{setup}} = 100 \text{ пс}$   
 $N = 1 \text{ событий за секунду}$
- Чему равна вероятность отказа? MTBF?

# Пример синхронизатора



- Предположим:  $T_c = 1/500 \text{ МГц} = 2 \text{ нс}$      $\tau = 200 \text{ пс}$   
 $T_0 = 150 \text{ пс}$      $t_{\text{setup}} = 100 \text{ пс}$   
 $N = 1 \text{ событие за секунду}$

- Чему равна вероятность отказа? MTBF?

$$P(\text{failure}) = (150 \text{ пс} / 2 \text{ нс}) e^{-(1.9 \text{ нс}) / 200 \text{ пс}}$$
$$= 5.6 \times 10^{-6}$$

$$P(\text{failure})/c = 10 \times (5.6 \times 10^{-6})$$
$$= 5.6 \times 10^{-5} / \text{с}$$

$$\text{MTBF} = 1/[P(\text{failure})/c] \approx 5 \text{ часов}$$

# Параллелизм

- **Два типа параллелизма:**
  - **Пространственный параллелизм:**
    - несколько копий аппаратных блоков в одно и то же время выполняют несколько задач
  - **Временной параллелизм:**
    - разбиение задачи на несколько ступеней
    - также называется конвейеризацией
    - например, сборочная линия



# Определения параллелизма

- **Токен (Token):** Набор входной информации, который обрабатывается для того, чтобы получить выходную информацию
- **Латентность (latency):** Время прохождения одного токена через всю систему с ее входа на выход
- **Пропускная способность (throughput):** Количество токенов, которое обрабатывается системой в единицу времени

**Параллелизм увеличивает пропускную способность**

# Пример параллелизма

- Бен готовит печенье для вечеринки, посвященной введению в эксплуатацию его контроллера светофора
- 5 минут уходит на сворачивание печенья на один противень
- В течение 15 минут печенье выпекается
- Чему равна пропускная способность и задержка без параллелизма?

# Пример параллелизма

- Бен готовит печенье для вечеринки, посвященной введению в эксплуатацию его контроллера светофора
- 5 минут уходит на сворачивание печенья на один противень
- В течение 15 минут печенье выпекается
- Чему равна пропускная способность и задержка без параллелизма?

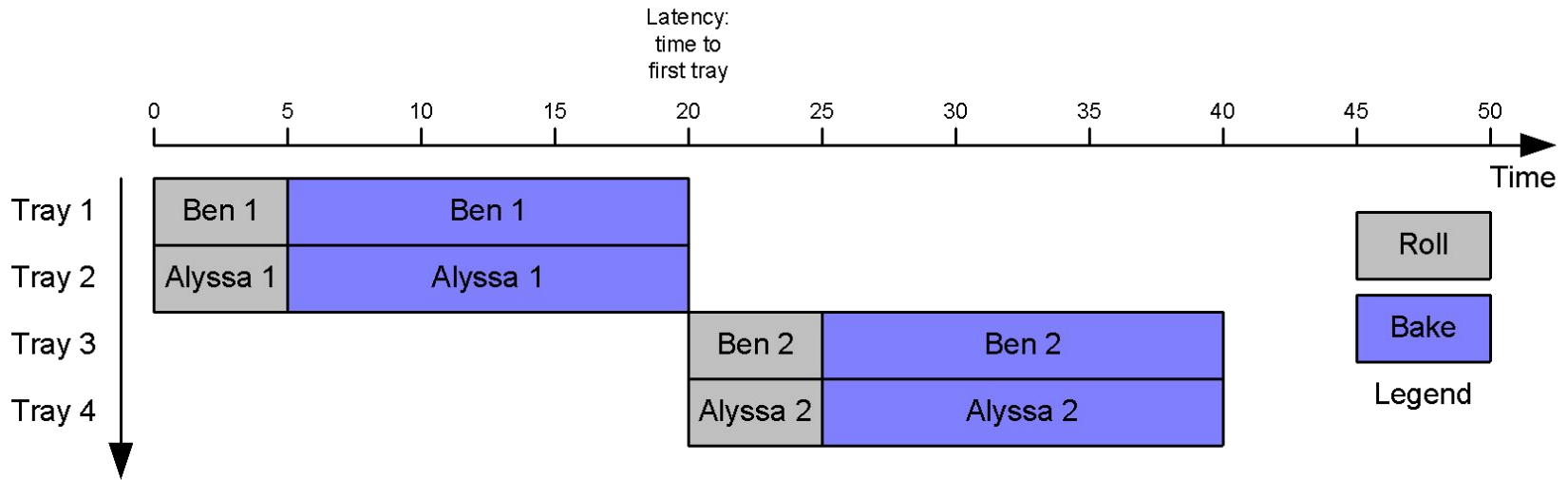
**Латентность** =  $5 + 15 = 20$  минут = **1/3 час**

**Пропускная способность** =  $1 \text{ противень} / 1/3 \text{ час} = 3$   
**противня/час**

# Пример параллелизма

- Чему равна пропускная способность и задержка если Бен использует параллелизма?
  - **Пространственный параллелизм:** Бен просит Алису помочь, используя ее печь
  - **Временной параллелизм:**
    - Две ступени: сворачивание и выпекание
    - Он использует два противня
    - Когда первая партия выпекается, он сворачивает следующую партию и т.д.

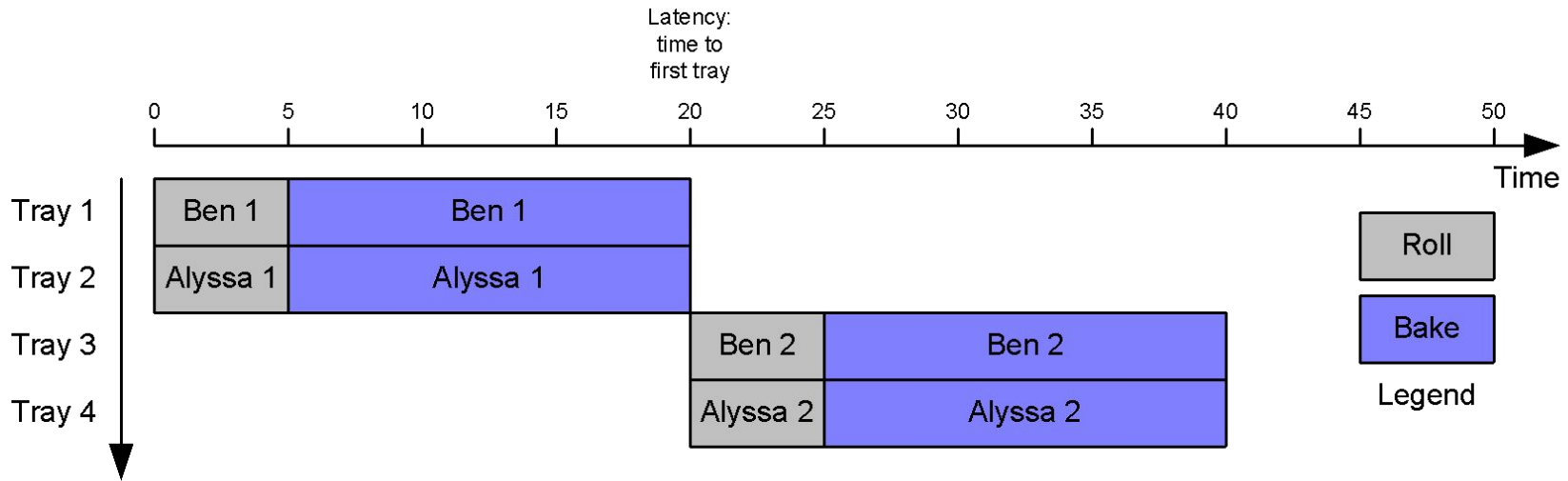
# Пространственный параллелизм:



Латентность = ?

Пропускная способность = ?

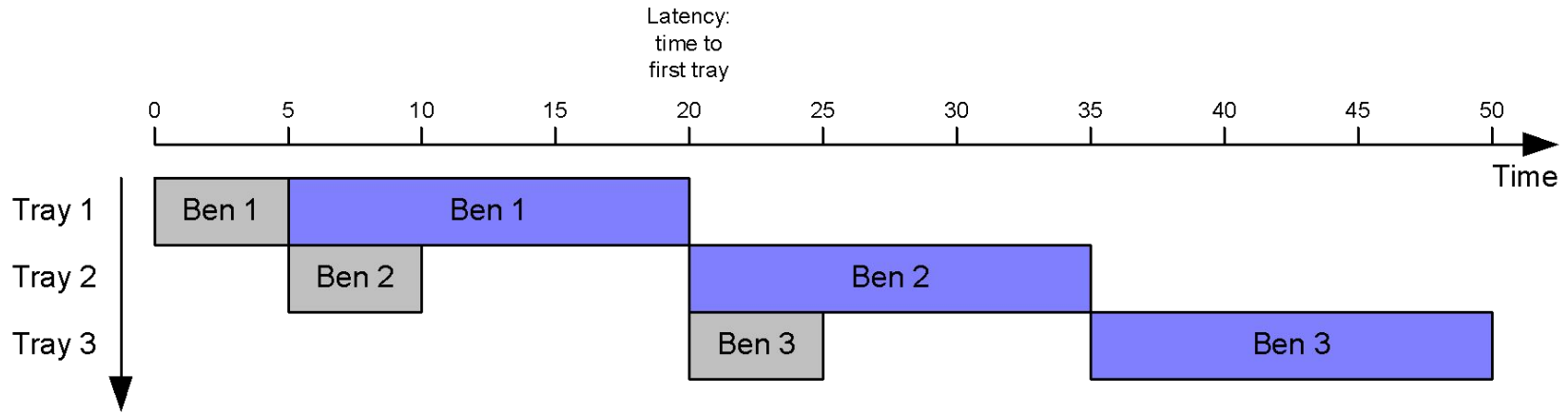
# Пространственный параллелизм:



Латентность =  $5 + 15 = 20$  минут =  $1/3$  час

Пропускная способность =  $2$  противня /  $1/3$  час =  $6$  противней/час

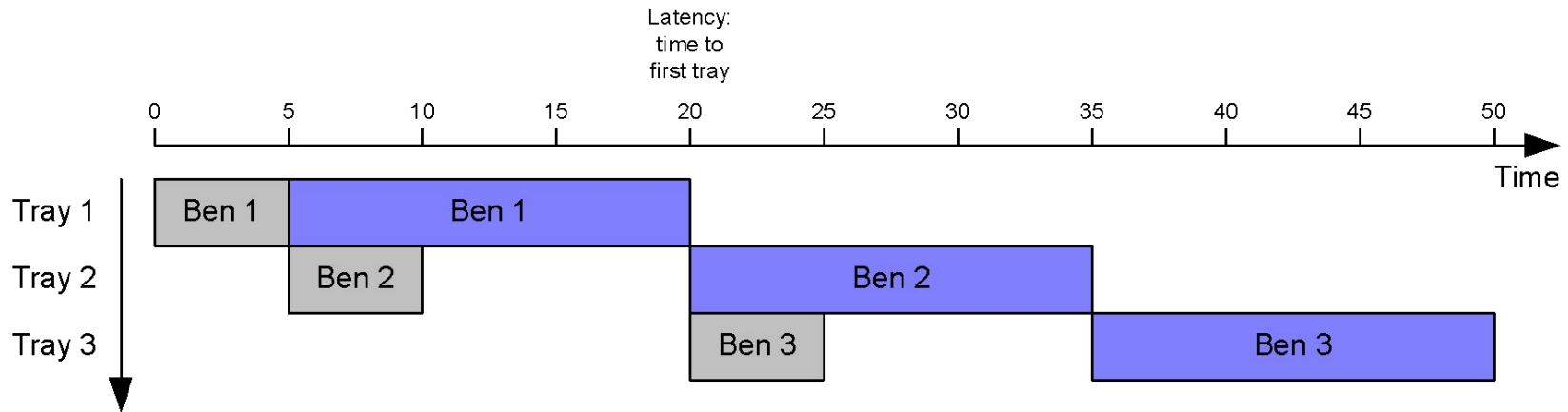
# Временной параллелизм



**Латентность = ?**

**Пропускная способность = ?**

# Временной параллелизм



Латентность =  $5 + 15 = 20$  минут = **1/3 час**

Пропускная способность =  $1 \text{ противень} / 1/4 \text{ час} = 4$   
**противней/час**

С использованием обеих технологий можно достигнуть  
пропускной способности **8 противней/час**