Работа 4. Синтез конечных автоматов

Цели работы

- Закрепление навыков структурного синтеза конечных автоматов (КА);
- Закрепление знаний о характеристиках и режимах работы триггеров основных типов;
- Получение практических навыков тестирования и управления КА;
- Получение навыков ввода проекта в графическом редакторе пакета QP, тестирования и отладки проекта и анализа временных характеристик КА;
- Знакомство с редактором КА пакета QP и анализ результатов синтеза;
- Получение навыков отладки цифровых устройств класса КА на физической модели: конфигурирование ПЛИС и экспериментальная проверка работы КА при использовании лабораторного стенда.

Системные требования

- САПР QP V15.1 и выше;
- стенд miniDiLaB-CIV с ПЛИС Cyclone IV EP4CE6E22C8N.

Трудоемкость работы

6 часов (3 часа самостоятельной работы и 3 часа в лаборатории).

Необходимая подготовка студентов для выполнения работы

- Курс ОВТ;
- Выполнение вводной работы;
- Выполнение работы 3 «Исследование триггеров».

1. Абстрактный конечный автомат

Абстрактный конечный автомат КА соответствует "пятерке" <**A, B, R, δ, λ**>, где **A, B, R** – множества состояний входа, выхода и внутренних, а δ и λ – функции переходов и выхода.

Для представления состояний автомата требуется память **П**, а реализация функций δ и λ осуществляется комбинационными схемами КС1 и КС2. Соответствующая структура КА представлена на рисунке 4.1. В ней обозначены: **X** – входные сигналы КА, **A** – множество значений вектора **X**; **Y** – выходные сигналы, **B** – множество значений вектора **Y**; **R** – множество внутренних состояний автомата (множество значений вектора **Q**); **Ф** – сигналы управления элементами памяти.

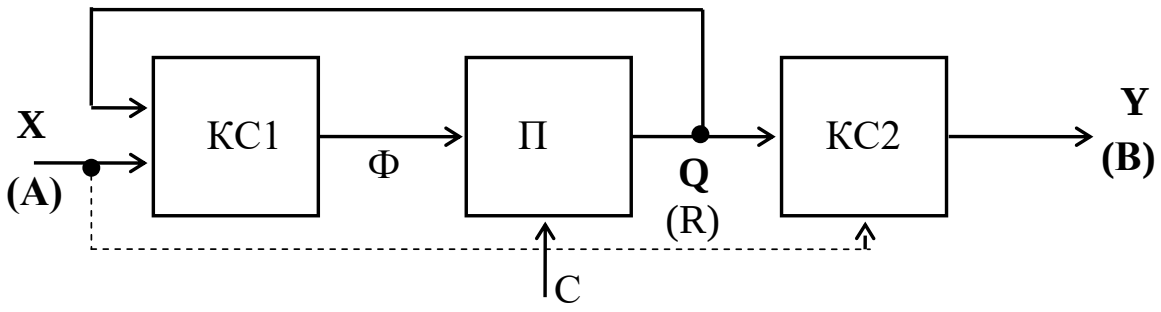


Рисунок 4.1 - Структура абстрактного КА

Если показанная на рисунке 4.1 пунктирная связь входа **X** с комбинационной схемой КС2 присутствует, то такой КА называют автоматом Мили. Его особенностью является наличие зависимости выходных сигналов **Y** от входных сигналов **X**. Если пунктирная связь отсутствует, то такой автомат называют автоматом Мура. Любой алгоритм может быть представлен как автоматом Мили, так и автоматом Мура.

Примечание: Абстрактный КА – это математическая модель, построенная в теоретико-множественных понятиях, которая используется при синтезе цифровых устройств.

При описании работы синхронных КА непрерывное время делят на такты, длительность которых определяется периодом следования синхронизирующих импульсов С на входе блока П. В течение одного такта КА находится в одном состоянии. Моменты переходов КА из одного состояния в другое нумеруют числами 0, 1, 2, 3, ... натурального ряда. При этом говорят, что КА работает в дискретном времени. Для отсчета времени используется безразмерная переменная t, соответствующая номеру такта.

Таким образом, алгебраическое представление работы КА можно привести к системе двух уравнений:

для автомата Мили $r^{t+1} = \delta(a^t, r^t)$; $b^t = \lambda(a^t, r^t)$

и для автомата Мура $r^{t+1} = \delta(a^t, r^t)$; $b^t = \lambda(r^t)$,

где a^t, b^t, r^t – состояния входа, выхода и внутреннее в такте t; r^{t+1} – внутреннее состояние в следующем такте (состояние перехода КА); δ, λ – функции перехода и выхода.

Задать КА означает определить множества **A**, **B**, **R** и функции δ, λ. Функции δ и λ задают отображения $\mathbf{A}*\mathbf{R} \rightarrow \mathbf{R}$ и $\mathbf{A}*\mathbf{R} \rightarrow \mathbf{B}$ соответственно. Таким образом, полным заданием КА являются таблицы переходов и выходов. Поле такой таблицы соответствует декартову произведению множеств **A** * **R**.

Рассмотрим пример задания автомата Мили.

δ: $\mathbf{A} * \mathbf{R} \rightarrow \mathbf{R}$ λ: $\mathbf{A} * \mathbf{R} \rightarrow \mathbf{B}$

a^t	r^t		
	r_1	r_2	r_3
a_1	r_2	r_3	r_2
a_2	r_3	r_2	r_1

a^t	r^t		
	r_1	r_2	r_3
a_1	b_1	b_3	b_3
a_2	b_2	b_1	b_1

$r^{t+1} = \delta(a^t, r^t)$ $b^t = \lambda(a^t, r^t)$

В рассматриваемом примере $\mathbf{A}=\{a_1, a_2\}$, $\mathbf{B}=\{b_1, b_2, b_3\}$, $\mathbf{R} = \{r_1, r_2, r_3\}$.

Один вход в таблицу – имена столбцов – перечисление внутренних состояний r^t, другой вход – имена строк – перечисление состояний входа. Для упрощения записей в таблицах часто записываются только индексы соответствующих переменных.

У автомата Мура таблица выходов содержит всего одну строку, в которой приводятся значения b^t, соответствующие r^t. У не полностью определенного КА в клетках с неопределенным (безразличным) состоянием r^t либо b^t ставится "Н" либо "-".

Другим распространенным способом задания КА служит граф переходов. Это ориентированный граф, вершины которого соответствуют внутренним состояниям, а дуги – направлениям переходов. Дуги помечены условием перехода и состоянием выхода (у автомата Мили). У автомата Мура состояния выхода записывают у соответствующих вершин графа.

На рисунке 4.2. приведен граф переходов автомата Мили, заданный приведенными выше таблицами переходов и выхода.

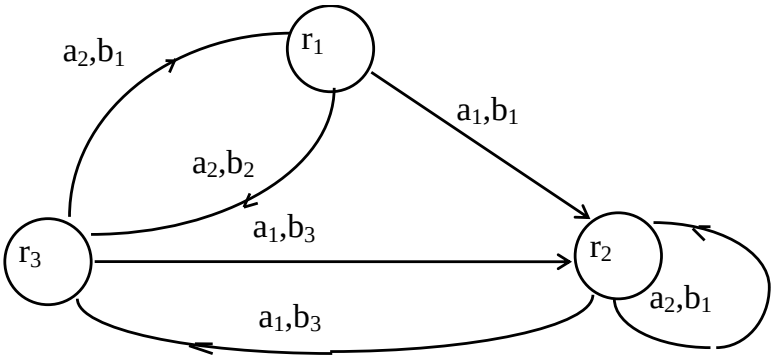


Рисунок 4.2 - Граф переходов автомата Мили

2. Триггеры – элементы блока памяти конечного автомата

Память для хранения внутренних состояний КА реализуется с помощью триггеров. Моделью триггера является элементарный КА, имеющий два внутренних состояния и один или несколько входов. Этот элементарный автомат (ЭА) должен удовлетворять условию полноты системы переходов, то есть должны существовать входные сигналы, обеспечивающие возможные переходы четырех типов: 0→0, 0→1, 1→0, 1→1. ЭА также должен удовлетворять требованию полноты системы выходов, то есть мощность множества внутренних состояний должна совпадать с мощностью множества состояний выходов и должно быть взаимно-однозначное соответствие между элементами этих множеств.

Условиям функциональной полноты удовлетворяют одновходовые триггеры D и T-типа и двухвходовые RS и JK-типа. Условные обозначения таких синхронных триггеров приведены на рисунке 4.3.

В таблице 4.1 для этих триггеров приведены функции возбуждения, то есть сигналы, которые нужно подать на входы триггера для перевода его из исходного состояния Q(t) в состояние Q(t+1). Функции возбуждения однозначно определяются таблицами переходов, полученных при исследовании триггеров в работе 3. Таблица 4.1 используется при синтезе блока управления памятью КС1 (на рисунке 4.1).

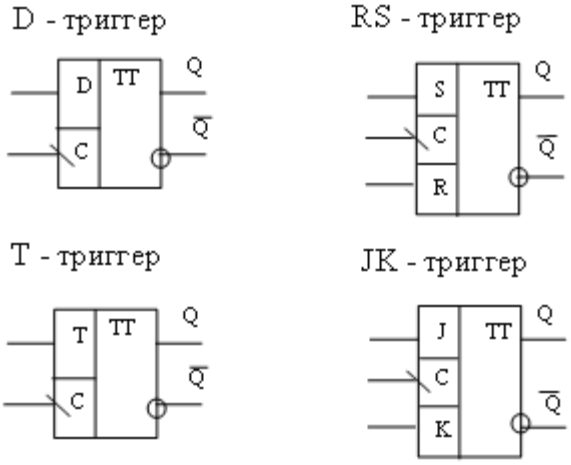


Рисунок 4.3 - Условные обозначения триггеров, синхронизируемых перепадом clk из 1 в 0

Таблица 4.1

Управление синхронными триггерами (активные уровни – высокие)

		D-триггер	T-триггер	RS-триггер	JK-триггер
Q(t)	**Q(t+1)**	***D***	***T***	***S R***	***J K***
0	0	0	0	0 Н	0 Н
0	1	1	1	1 0	1 Н
1	0	0	1	0 1	Н 1
1	1	1	0	Н 0	Н 0

Триггер D называют триггером-задержкой, так как следующее состояние Q(t+1) полностью определяется значением D(t), и функция возбуждения для D-триггера совпадает с состоянием Q(t+1).

Функция возбуждения для T-триггера (счётного триггера) определяется как T=1, если триггер должен изменить свое состояние на противоположное (0→1 или 1→0) и T=0, если триггер должен сохранить прежнее состояние (0→0 или 1→1).

RS-триггер имеет два входа (R – установка в 0, S – установка в 1), то есть возможны четыре различные комбинации сигналов. Одна из комбинаций (R,S)=(1,1) запрещена, так как теоретически приводит к неопределенному состоянию. Однако, в практических реализациях триггеров состояние перехода, как правило, известно и определено в спецификации. Если триггер должен сохранить прежнее состояние, то необходимо обеспечить S=0 (хранение 0) или R=0 (хранение 1) при безразличном состоянии другого входа, что отмечено в табл. 1 символом "Н".

Универсальный JK-триггер совмещает функции RS и T-триггеров. При (J,K)=(0,0), (0,1), (1,0) он ведет себя аналогично RS-триггеру, если K=R, J=S. При (J,K)=(1,1) состояние триггера изменяется на противоположное, то есть он ведет себя как T-триггер. Соответственно изменяется таблица функций возбуждения JK-триггера.

3. Метод структурного синтеза синхронных автоматов

При структурном синтезе КА решаются следующие задачи:

- определение необходимого объема памяти КА;
- выбор типа триггеров и кодирование внутренних состояний;
- определение системы функций переходов и выходов;
- минимизация системы функций;
- синтез комбинационных схем КС1 и КС2
- и построение принципиальной схемы КА.

Минимальное количество триггеров m в памяти автомата определяется из числа его состояний R как

$$m = \lceil \log_2 R \rceil,$$

где $\lceil \ \rceil$ – функция округления до ближайшего целого в большую сторону.

Способы кодирования состояний КА во многом определяются используемой элементной базой.

Если при этом $2^m > R$, то КА следует доопределить, включив в него "лишние" 2^m-R состояний. Их следует связать безусловным переходом с состоянием автомата, принятым за начальное. При нормальном функционировании КА "лишние" состояния не используются. Но либо в результате сбоя, либо при включении питания КА может случайно оказаться в одном из этих состояний. Если не предусмотреть выход из них, то произойдет "зависание" автомата. *В условиях лабораторной работы для выхода из этой ситуации целесообразно организовать начальную установку КА, используя асинхронные входы триггера.*

В синхронных КА, как уже указывалось выше, в качестве элементов памяти используются триггеры, синхронизируемые перепадом потенциала, что исключает влияние состязания сигналов. Поэтому выбор конкретного типа триггера осуществляется из условия минимизации схемы КС1. Так как у триггеров D- и T-типа всего один вход, а у RS- и JK – два, то число функций, реализуемых КС1, в первом случае в два раза меньше, чем во втором. С другой стороны, в одновходовых триггерах при реализации того или иного перехода состояние входа однозначно задано, в то время как для JK-триггера состояние одного или другого входа часто оказывается безразличным (см. таблицу 4.1). Поэтому, хотя функций возбуждения в последнем случае больше, они обычно проще. Сравнительная оценка сложности КС1 для этих случаев зависит от конкретного графа переходов. В рамках лабораторной работы тип используемых триггеров задан.

Важным фактором, определяющим экономичность схемы КС1, является кодирование внутренних состояний КА. В результате процесса кодирования с каждым внутренним состоянием автомата сопоставляется определенная кодовая комбинация, образованная значениями переменных, характеризующих состояние памяти (для уменьшения числа триггеров памяти используются кодовые комбинации двоичного кода).

Для экономичного кодирования можно использовать эвристический алгоритм, направленный на повышение возможности склеивания термов в функциях возбуждения триггеров, реализуемых КС1. В соответствии с этим алгоритмом для кодирования состояний автомата используют два правила.

П1. Те состояния r^t, из которых есть переход в одно и то же состояние r^{t+1} при одном состоянии входа, следует кодировать соседними кодами.

П2. Те состояния r^{t+1}, переход в которые осуществляется из одного и того же состояния r^t, следует кодировать соседними кодами.

Если всем условиям удовлетворить нельзя, предпочтение следует отдать правилу П1. Ниже, в примере синтеза КА проиллюстрировано применение этого алгоритма.

Для формирования функций возбуждения и выхода необходимо построить таблицы переходов и выходов, в которых учтены результаты кодирования и функции возбуждения выбранного типа триггеров. Методика формирования системы функций изложена ниже на примере структурного синтеза синхронного конечного автомата с четырьмя состояниями и двумя управляющими входами.

В среде QP реализована функция синтеза КА по поведенческому описанию, которая может быть использована как альтернатива описанному выше структурному синтезу. При этом существует возможность выбора способа кодирования внутренних состояний КА при компиляции из следующих вариантов:

- **автоматически (Auto)** – компилятором выбирается лучший способ кодирования, используется по умолчанию;
- **код Грея (Gray)** – использует минимально возможное число бит, кодировка каждого состояние отличается от соседних только в одном разряде;
- **код Джонсона (Johnson)** – число бит для кодирования равно половине числа состояний (округленное до ближайшего большего целого), кодировка каждого состояние отличается от соседних только в одном разряде, код для состояния генерируется сдвигом кода предыдущего состояния вправо на 1 разряд, старший разряд кода получается инверсией младшего разряда кода предыдущего состояния;
- **минимальное число бит (Minimal Bits)** – использует минимально возможное число бит;
- **код 1 из N (One-Hot)** – модифицированный вариант кода, начальное состояние кодируется всеми нулями (чтобы по включению питания КА оказался в начальном состоянии), для остальных состояний один или два бита из всех равны единице. Каждое состояние может быть определено состоянием одного бита (как и для классического кода 1 из N), такой код удобно использовать в случае возможных переходов КА в неопределённое состояние;
- **последовательный двоичный код (Sequential)** – использует минимально возможное число бит, код является двоичным номером состояния;
- **пользовательская кодировка (User-Encoded)** – при синтезе КА из описания на языке описания аппаратуры используется кодирование, описанное пользователем.

Выбор способа кодирования состояний КА осуществляется с помощью настройки State Machine Processing. Просмотр кодировки для реализованного КА доступен через State Machine Viewer. Описание соответствующих команд QP см. в п. 6.

4. Пример структурного синтеза синхронного автомата

Пусть автомат Мура задан следующими таблицами переходов и выхода.

$r^{t+1} = \delta(a^t, r^t)$ $y^t = \lambda(r^t)$

$a^t=(x_2,x_1)^t$	r^t			
	0	***1***	***2***	***3***
0 0	0	3	1	2
0 1	H	H	H	H
1 0	H	0	3	1
1 1	3	2	0	3

r^t	***0***	***1***	***2***	***3***
$b^t=(y_2,y_1)^t$	01	11	00	10

Соответствующий этим таблицам граф переходов КА приведен на рисунке 4.4.

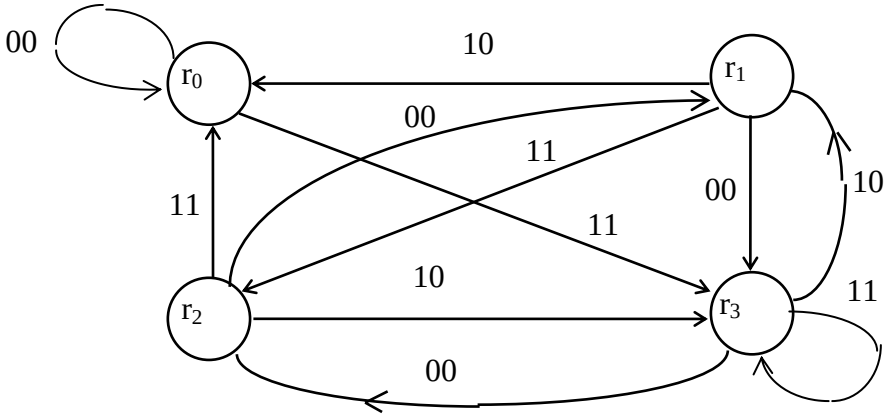


Рисунок 4.4 - Граф переходов синтезируемого автомата

Входной алфавит состоит из четырех абстрактных состояний входа, которые уже закодированы двумя входными сигналами x_2 и x_1. Выходной алфавит состоит из четырех состояний выхода и закодирован двумя выходными сигналами y_2 и y_1.

Для экономичного кодирования внутренних состояний воспользуемся описанным в разделе 2 алгоритмом. Определим пары состояний, которые желательно кодировать соседними кодами.

По правилу П1: (0, 3).

По правилу П2: (0, 2), (2, 3), (1, 0), (1, 3), (1,2).

Выберем вариант размещения номеров состояний, удовлетворяющий правилу П1 и, по возможности, правилу П2.

r_i	0	1	2	3
(Q_2, Q_1)	(0, 0)	(0, 1)	(1, 1)	(1, 0)

Примечание: *Кодирование по приведенному алгоритму неоднозначно. Заслуживает внимания вариант кодирования внутренних состояний по таблице выходов, который не удовлетворяет правилу П1, однако исключает необходимость использования схемы КС2, так как в этом случае* $y_1 = Q_1$, $y_2 = Q_2$.

Заменив в исходной таблице переходов состояния их кодами, получим закодированную таблицу переходов синтезируемого КА:

$(Q_2,Q_1)^{t+1} = \delta((Q_2,Q_1)^t, (x_2,x_1)^t)$.

$(x_2,x_1)^t$	$(Q_2,Q_1)^t$			
	00	***01***	***11***	***10***
0 0	00	10	01	11
0 1	Н	Н	Н	Н
1 0	Н	00	10	01
1 1	10	11	00	10

Для реализации памяти будем использовать JK-триггеры, синхронизируемые перепадом потенциала. Тогда дальнейший синтез КА сводится к синтезу двух комбинационных схем, КС1 – реализующей логические функции возбуждения триггеров J_2, K_2, J_1, K_1, и КС2 – логические функции выходов Y_2, Y_1.

Составим таблицу функций возбуждения памяти. Переходы триггеров памяти происходят под действием сигналов функций возбуждения, поступающих на их входы. Для определения того, что нужно подать на вход ЭА, чтобы перевести его из 0 в 1, обратимся в табл. 1 к функции возбуждения JK-триггера. Видно, что на J-вход надо подать сигнал 1, на K-вход – безразлично какой. Аналогично определяются входные сигналы для других переходов и заносятся в соответствующие строки таблицы функций возбуждения. Например, если автомат находился в состоянии (1, 0), то при входном сигнале x_2=1, x_1=0 согласно таблице переходов он перейдет в состояние (0, 1), то есть оба триггера поменяют свои состояния на противоположные. Чтобы этот переход произошёл, согласно таблице управления JK-триггера на его входы необходимо подать следующие сигналы управления: J_2=Н, K_2=1, J_1=1, K_1=Н.

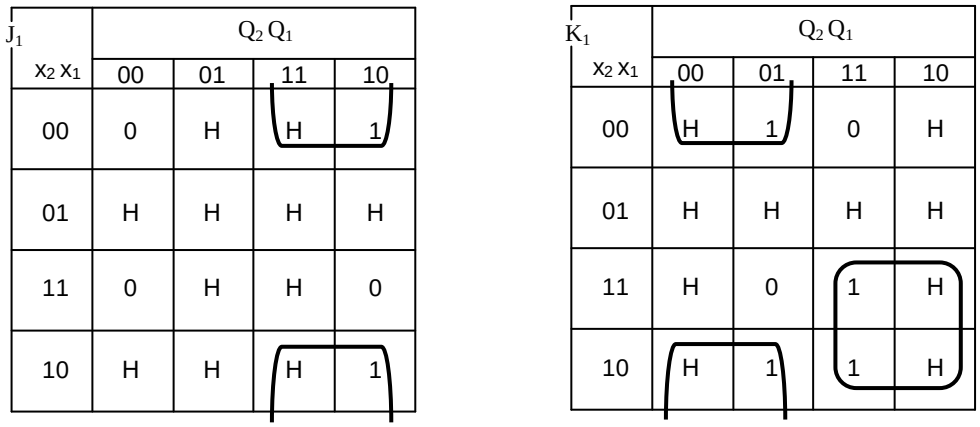
На основании закодированной таблицы переходов КА и функций управления JK-триггеров получаем таблицу истинности (таблица 4.2). После составления таблицы функций управления проводится совместная минимизация этих функций (см. рисунок 4.5).

Примечание*: Функции J и K имеют общую часть и могут быть реализованы совместно.*

Таблица 4.2

Таблица истинности для КС1

$(x_2,x_1)^t$	$(Q_2,Q_1)^t$	J_2	K_2	J_1	K_1
00	***00***	0	Н	0	Н
00	***01***	1	Н	Н	1
00	***11***	Н	1	Н	0
00	***10***	Н	0	1	Н
01	***00***	Н	Н	Н	Н
01	***01***				
01	***11***				
01	***10***				
10	***00***	Н	Н	Н	Н
10	***01***	0	Н	Н	1
10	***11***	Н	0	Н	1
10	***10***	Н	1	1	Н
11	***00***	1	Н	0	Н
11	***01***	1	Н	Н	0
11	***11***	Н	1	Н	1
11	***10***	Н	0	0	Н



J_1	$Q_2 Q_1$			
$x_2 x_1$	00	01	11	10
00	0	H	H	1
01	H	H	H	H
11	0	H	H	0
10	H	H	H	1

K_1	$Q_2 Q_1$			
$x_2 x_1$	00	01	11	10
00	H	1	0	H
01	H	H	H	H
11	H	0	1	H
10	H	1	1	H

$$J_1 = \overline{x_1} Q_2 \qquad K_1 = \overline{x_1}\,\overline{Q_2} + x_2 Q_2$$

$$J_1 = \overline{\overline{x_1} Q_2} \qquad K_1 = \overline{\overline{x_1}\,\overline{Q_2}} \bullet \overline{x_2 Q_2}$$

J_2	$Q_2 Q_1$			
$x_2 x_1$	00	01	11	10
00	0	1	H	H
01	H	H	H	H
11	1	1	H	H
10	H	0	H	H

K_2	$Q_2 Q_1$			
$x_2 x_1$	00	01	11	10
00	H	H	1	0
01	H	H	H	H
11	H	H	1	0
10	H	H	0	1

$$J_2 = \overline{x_2} Q_1 + x_2 x_1 \qquad K_2 = \overline{x_2} Q_1 + x_1 Q_1 + x_2 \overline{x_1}\,\overline{Q_1}$$

$$J_2 = \overline{\overline{x_2} Q_1} + \overline{x_2 x_1} \qquad K_2 = \overline{\overline{x_2} Q_1} \bullet \overline{x_1 Q_1} \bullet \overline{x_2 \overline{x_1}\,\overline{Q_1}}$$

Рисунок 4.5 - Карты Карно для входов JK-триггеров

Составляется также таблица для выходных сигналов, как функций состояния автоматов, и проводится минимизация.

Q_2 Q_1	**y_2 y_1**
0 0	0 1
0 1	1 1
1 1	0 0
1 0	1 0

$$y_1 = \overline{Q_2}$$

$$y_2 = \overline{Q_1} Q_2 + Q_1 \overline{Q_2} = \overline{\overline{Q_1} Q_2}\;\overline{Q_1 \overline{Q_2}}$$

На рисунке 4.6 представлена функциональная схема в графическом редакторе QP, реализующая синтезированный в примере КА. Для реализации КС1 и КС2 используются только элементы Шеффера.

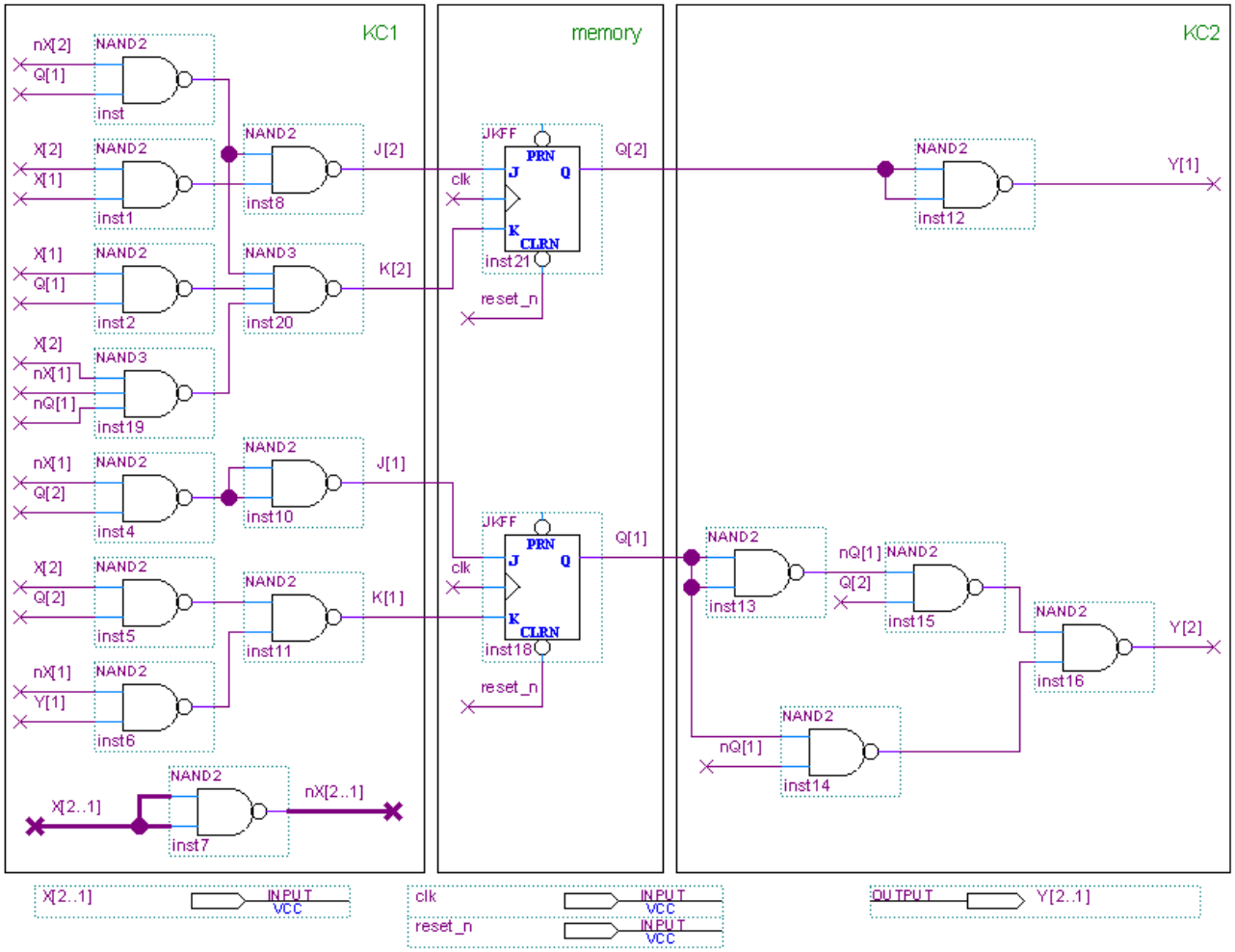


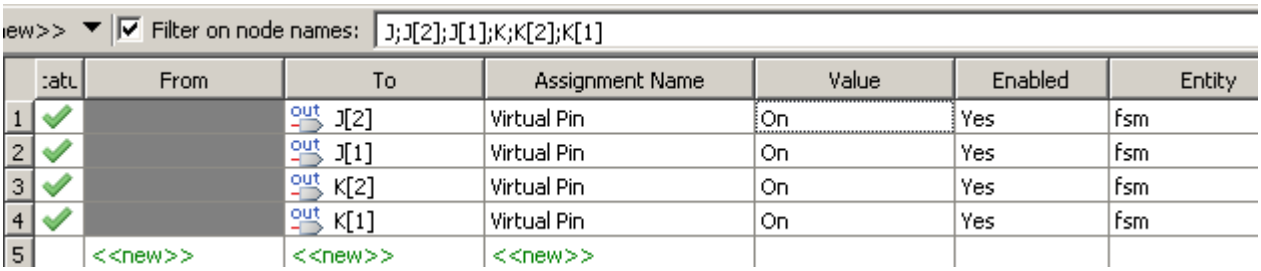
Рисунок 4.6 - Функциональная схема КА

5. Проверка и наладка синтезированного КА

Проверку реализуемого КА осуществляют, заполняя таблицы переходов и выходов в процессе экспериментального исследования схемы на имитационной (моделирование в QP) и физической (лабораторный стенд miniDiLab) моделях и сравнивая эти таблицы с полученными теоретически. Для этого предварительно составляется тест, то есть последовательность входных сигналов, позволяющая последовательно, переходя из одного состояния в другое, осуществить полную проверку таблицы переходов. Переходы, которые не определены в исходной таблице, также исследуются, и в таблице отмечается, как они доопределены. При проверке очередного перехода сначала устанавливается входной сигнал (он может быть и тем же, что и в предыдущем такте), затем подается синхроимпульс и фиксируются состояния триггеров и выходов. Если экспериментальная и теоретическая таблицы переходов и выходов совпали, значит, КА реализует заданный алгоритм работы; если хотя бы для одного перехода нет совпадения, необходимо осуществить наладку собранной схемы. Для контроля таблицы переходов на временной диаграмме наблюдайте состояние триггеров (В окне Node Finder маска фильтра Register: post-fitting).

Для наладки КА при моделировании необходимо наблюдать сигналы управления триггерами. Сформированные комбинационными схемами, они могут не входить в число сигналов, доступных для моделирования. Поэтому для их наблюдения необходимо подключить их к выходам схемы, далее выполнить анализ и синтез, чтобы выходы вошли в список сигналов и затем, локализовав выходы в редакторе назначений (Assignment Editor через контекстное меню элемента ввода/вывода), задать их как виртуальные выводы (см. рисунок 4.7). На временных диаграммах наблюдать сигналы с выходов функциональных

преобразователей, подключенных к виртуальным выводам (см. Technology Map Viewer). После отладки виртуальные выводы уберите, они увеличивают аппаратурные затраты.



	:atu	From	To	Assignment Name	Value	Enabled	Entity
1	✓		out J[2]	Virtual Pin	On	Yes	fsm
2	✓		out J[1]	Virtual Pin	On	Yes	fsm
3	✓		out K[2]	Virtual Pin	On	Yes	fsm
4	✓		out K[1]	Virtual Pin	On	Yes	fsm
5		<<new>>	<<new>>	<<new>>			

Рисунок 4.7 - Назначение виртуальных выходов

Алгоритм работы может не соответствовать теоретическому из-за ошибок в синтезе и (или) из-за ошибок при вводе схемы. Процедуру наладки можно представить в виде следующей последовательности действий:

1. Выводим на индикацию сигналы управления триггеров.
2. Устанавливаем триггеры в состояние ri, из которого осуществляется неправильный переход (если этих состояний несколько, то в любое из них).
3. Устанавливаем входные сигналы, соответствующие неправильному переходу из состояния ri, то есть соответствующие клетке в таблице переходов, не совпавшей с теоретической.
4. Не подавая синхроимпульс, фиксируем сигналы управления триггера, сформированные комбинационной схемой, и сверяем их с соответствующей строкой в таблице управления триггеров, построенной в процессе синтеза КА. Здесь возможны несколько вариантов:

– сигналы управления, полученные в эксперименте, совпали с сигналами, рассчитанными при синтезе. Проверяем теоретически для заданного типа триггеров, какие переходы инициализируют данные сигналы управления. Если теоретически данные сигналы возбуждения должны перевести триггеры в заданное состояние, то причина ошибки в неправильном включении триггера;
– экспериментальные сигналы возбуждения совпали с рассчитанными при синтезе, но эти сигналы теоретически не инициализируют нужный переход. Следовательно, произошла ошибка при синтезе, и устранить ее можно проверкой данного перехода (клетки в таблице переходов) на всех предшествующих составлению таблицы возбуждения этапах синтеза;
– экспериментальные сигналы возбуждения не совпали с рассчитанными при синтезе. Ошибку в этом случае следует искать в синтезе или реализации комбинационной схемы, формирующей сигналы возбуждения триггера. Для выявления ошибок этого типа можно воспользоваться рекомендациями по наладке комбинационных схем.

Если ошибка найдена, необходимо внести исправления и снова снять экспериментальную таблицу переходов. Процедура наладки повторяется, пока не будет достигнуто полное соответствие теоретической и экспериментальной таблиц переходов.

Если экспериментальная таблица переходов соответствует теоретической, а таблица состояния выходов – нет, то ошибку следует искать в синтезе и реализации КС2, формирующей выходной сигнал, согласно методике, изложенной в лабораторной работе по синтезу КС.

6. Синтез КА средствами QP

При описанном выше структурном синтезе получаемое в результате устройство не интерпретируется QP как КА, что в случае сложных устройств может усложнить поиск ошибок в описании и отладку. Пакет QP предоставляет пользователям возможность задания КА на одном из языков описания аппаратных средств и его синтеза как

специального объекта. Для устройств, введенных и синтезированных как КА, доступны дополнительные средства визуализации.

Кроме того, для ввода конечных автоматов в QP может использоваться редактор конечных автоматов. Для доступа к редактору создайте новый файл типа **State Machine File** (см. рисунок 4.8). Созданный файл имеет расширение **smf**.

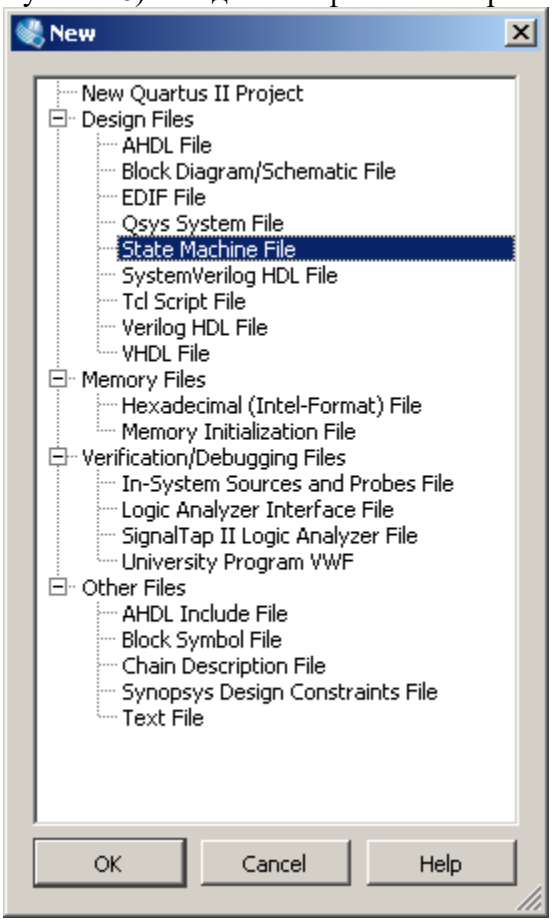


Рисунок 4.8 - Выбор типа State Machine File для нового файла

Щелкните **ОК** – откроется окно редактора ввода КА, как показано на рисунке 4.9.

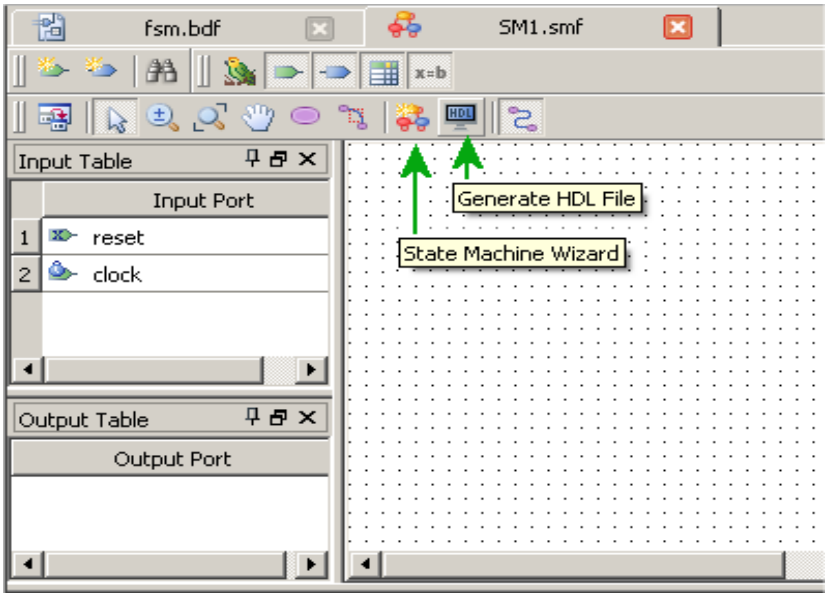


Рисунок 4.9 - Окно графического редактора КА

На вертикальной панели инструментов открывшегося окна выберите **State Machine Wizard**. Следуя инструкциям, задайте состояния автомата, его входные и выходные сигналы.

Обратите внимание на закладку **General** появившегося диалога, где следует задать характеристики начальной установки КА (сброс, Reset) , как показано на рисунке 4.10. Используемые настройки существенно влияют на аппаратные затраты для синтезируемого КА.

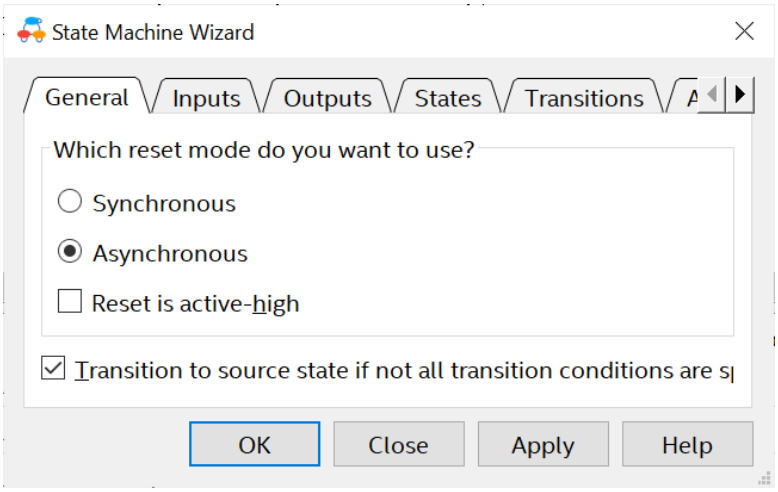


Рисунок 4.10 - Задание режима Reset.

Для показанных на рисунке 10 настроек установка в начальное состояние будет осуществляться асинхронно с активным уровнем сигнала 0. Такая настройка соответствует КА, синтезированному в п.4 работы. Если условия переходов из текущего состояния в другие не заданы, то КА будет оставаться в текущем состоянии.

На вкладках **Inputs** и **Outputs** задайте входные и выходные сигналы КА. На вкладке **States** задайте состояния КА и укажите состояние, в которое автомат будет устанавливаться по команде Reset. На закладках **Transitions** и **Actions** задайте таблицы переходов и выходов, используя следующие обозначения логических операций: **~** – инверсия; **&** – логическое «И»; **|** – логическое «ИЛИ». Вид соответствующих окон **State Machine Wizard** показан на рисунке 4.11.

Синтаксис языка **Verilog** для задания условий перехода:

```
"==" EQUAL                  "&" AND
"!=" INEQUAL                "|" OR
"<=" LESSER THAN            "^" XOR
"<" LESSER                  "~&" NAND
">=" GREATER THAN           "~|" NOR
">" GREATER                 "~^" XNOR
                            "~" NOT
```

General | Inputs | Outputs

Input Port	Controlled Signal
clock	Clock
reset	Reset
x1	No
x2	No
< New >	

General | Inputs | Outputs | States | Tra

Output Port	Registered	Output State
y1	No	Current clock cycle
y2	No	Current clock cycle
< New >		

General | Inputs | Outputs | States | Transitions

Source State	Destination State	Transition (In Verilog or
r0	r0	~x2 & ~x1
r0	r3	x2 & x1
r1	r0	x2 & ~x1
r1	r2	x2 & x1
r1	r3	~x2 & ~x1
r2	r0	x2 & x1
r2	r1	~x2 & ~x1
r2	r3	x2 & ~x1
r3	r1	x2 & ~x1
r3	r2	~x2 & ~x1
r3	r3	x2 & x1

General | Inputs | Outputs | States

State	Reset
r0	Yes
r1	No
r2	No
r3	No
< New >	

Рисунок 4.11 - Настройка КА

После окончания ввода на поле отображения редактора конечных автоматов появится граф переходов введенного автомата. Вид окна для рассматриваемого в работе примера приведён на рисунке 4.12.

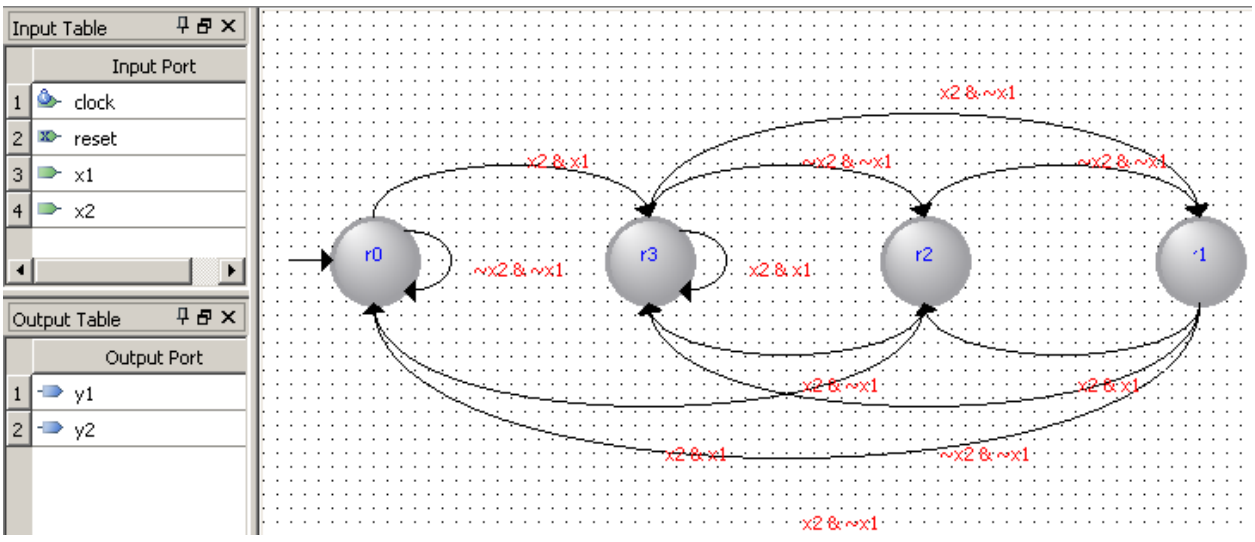


Рисунок 4.12 - Результат графического ввода КА

На вертикальной панели инструментов щелкните кнопку **Generate HDL File**. В появившемся окне выберите язык **Verilog** и нажмите **ОК**. Генерируется логический файл, описывающий КА. Этот файл можно компилировать и для него можно выполнять моделирование. Откройте полученный файл и ознакомьтесь с его содержимым.

Внимание! *Имя создаваемого файла должно отличаться от уже существующих в папке проекта имен.*

Для того, чтобы структура КА была близка к структуре КА, синтезированной в п.4 работы, задайте способ кодирования состояний "**минимальное число бит (Minimal Bits)".** Для этого в окне главного менеджера QP выполните **Assignments => Settings => Compiler Settings => Advanced Analysis & Synthesis Settings => State Machine Processing** и выберите **Minimal Bits.** Эта же настройка может быть установлена через **Assignment Editor**. Выполните полную компиляцию. Это позволит определить аппаратурные затраты и временные характеристики разработанного КА.

Чтобы посмотреть, каким образом компилятором закодированы состояния КА, откройте **Tools => Netlist Viewers => State Machine Viewer**. **State Machine Viewer** представляет граф переходов синтезированного КА, а также таблицу с условиями этих переходов. Начальное состояние, в которое КА переходит по сигналу сброса, отображается первым слева в ряду состояний (всегда закодировано всеми нулями). Информация о кодировании состояний появляется после выполнения анализа и синтеза.

Моделирование работы полученного КА осуществляется так же, как в п.4. Соответствие состояний КА состояниям триггеров определяется в **State Machine Viewer**.

Программа работы

1. Выполните структурный синтез КА:
   - 1.1. Составьте по индивидуальному заданию таблицу переходов и выходов.
   - 1.2. Определите необходимое для реализации КА число триггеров.
   - 1.3. Осуществите кодирование внутренних состояний КА и составьте таблицу возбуждения памяти КА для заданного типа триггеров.
   - 1.4. Осуществите совместную минимизацию логических функций возбуждения триггеров с помощью карт Карно, выберите один из минимальных вариантов и запишите полученные логические функции в базисе Шеффера (для инвертирования используйте элементы NOT).
   - 1.5. Осуществите минимизацию логических функций выхода КА и также запишите их в базисе Шеффера.
2. Выполните исследование синтезированного автомата при реализации на ПЛИС:
   - 2.1. По результатам структурного синтеза введите принципиальную схему КА в QP.
   - 2.2. Определите системное окружение из состава стенда miniDiLab, необходимое для исследования КА на физической модели: управление сигналами на входах КА, отображение выходов и текущего состояния КА. При необходимости модифицируйте вашу схему соответствующим образом и анализ и синтез проекта. Выполните назначение выводов ПЛИС.
   - 2.3. Выполните полную компиляцию. Определите аппаратурные затраты (**Compilation Report => Fitter => Summary**) и временные характеристики для медленной модели **Slow 1200mV 85C** (**Timing Analyzer => Datasheet => Report Fmax Summary** и **Report Datasheet => Clock to Output Times**).
   - 2.4. Средствами моделирования составьте полный тест для проверки таблиц переходов и выходов для всех переходов, указанных в задании, и переходов, доопределённых при синтезе. В отчете на временной диаграмме укажите символьное отображение состояний КА. При выявлении ошибок в работе КА осуществите его настройку. По результатам эксперимента заполните экспериментальные таблицы переходов и выходов.
   - 2.5. Создайте версию проекта для исследования КА на физической модели. Дополните Ваш проект формирующим сигналы тактирования делителем частоты, который позволит наблюдать изменение состояний КА. Выполните компиляцию. Выполните программирование ПЛИС и проверьте работу КА на стенде. Заполните таблицы переходов и выходов экспериментальными данными. При

защите работы на стенде будьте готовы продемонстрировать заданные преподавателем переходы КА по графу состояний.

3. Выполните исследование заданного КА при его синтезе средствами QP:
   - 3.1. Создайте новый проект. Введите КА средствами **State Machine Editor**. Выполните генерацию HDL файла и проанализируйте полученный файл. Выполните компиляцию.
   - 3.2. Откройте **State Machine Viewer** и ознакомьтесь с реализацией синтезированного КА. Определите используемый способ кодирования состояний автомата (закладка Encoding). Сравните его со способом кодирования, использованным вами при структурном синтезе (п. 1.3). Определите аппаратурные затраты и быстродействие
   - 3.3. Задайте способ кодирования КА **Minimal Bits** (**Assignments => Settings => Compiler Settings => Advanced Settings (Synthesis) => State Machine Processing**), наблюдайте, как меняются аппаратурные затраты и быстродействие. Объясните полученные результаты.
   - 3.4. Проведите моделирование работы КА по тесту, составленному для п.2. Выведите на временную диаграмму состояния триггеров памяти конечного автомата (регистровые сигналы в Node Finder).
   - 3.5. Сравните результаты моделирования с результатами, полученными в п.2 программы работы.
4. Оформите отчёт по проделанной работе, включив в него результаты проведённого структурного синтеза (п. 1); принципиальную схему КА, результаты тестирования средствами моделирования, описание и результаты тестов, проведённых на стенде (п. 2); результаты синтеза КА средствами QP (не включая текст сгенерированного HDL файла), результаты анализа синтезированного устройства, результаты сравнения двух полученных реализаций КА (п. 2. и п. 3). В отчёте должны быть отображены результаты всех исследований, проведённых в работе. Выводы по работе должны включать в себя анализ всех полученных в работе результатов.

Контрольные вопросы

1. Укажите особенности использования различных типов синхронных триггеров в КА.
2. Как изменится функционирование КА, если вместо триггеров, синхронизируемых перепадом, использовать триггеры, синхронизируемые уровнем?
3. Чем отличается задание автомата Мили от предложенного вам автомата Мура?
4. Какова максимальная частота тактирования и чем ограничено быстродействие схемы?
5. Как задаётся начальное состояние КА?
6. Где КА хранит своё текущее состояние?
6. Можно ли использовать для реализации памяти КА число триггеров, большее чем выбранное в п.1.2?
7. Могут ли появляться риски сбоя на выходах КА?

Варианты индивидуальных заданий

Номер задания	**Тип триггера**	**Таблица переходов**				**Таблица выходов**			
		x_2x_1=00	**x_2x_1=01**	**x_2x_1=10**	**x_2x_1=11**	**r_0**	**r_1**	**r_2**	**r_3**
		r				**Y**			
	JK	**0 1 2 3**	**0 1 2 3**	**0 1 2 3**	**0 1 2 3**	**2 1**	**2 1**	**2 1**	**2 1**
Пример		**0 3 1 2**	**H H H H**	**H 0 3 1**	**3 2 0 3**	**0 1**	**1 1**	**0 0**	**1 0**
1	D	1 3 2 1	H 0 2 2	H H H H	2 H 0 H	1 1	0 1	0 0	1 0
2	JK	3 0 H 3	1 2 H 2	1 1 0 H	H H H H	0 1	1 0	0 1	1 0
3	D	0 2 0 0	2 H H 2	H H H H	H 3 1 1	0 0	1 0	0 1	1 1
4	JK	2 2 2 1	1 H 1 H	H H H H	3 3 H 0	1 0	1 0	0 1	0 1
5	D	2 3 0 H	0 0 1 H	H H H H	1 0 3 2	0 1	1 1	1 0	0 0
6	JK	3 1 H 0	H H H H	0 0 3 1	1 3 H H	1 0	1 1	0 1	0 0
7	D	0 1 H 0	H H H H	2 1 1 2	H 3 3 H	1 1	0 0	0 1	1 0
8	JK	2 H 0 1	H H H H	3 0 H 2	H 2 3 1	0 1	1 0	1 0	0 1
9	D	0 1 H 0	H H H H	1 H 3 2	H 2 1 0	0 0	1 0	1 1	0 1
10	JK	H H H H	2 H 0 1	3 0 H 2	H 2 3 1	1 1	1 0	0 1	0 0
11	D	1 2 0 H	3 0 1 0	H H H H	2 3 1 1	0 1	1 0	1 1	0 0
12	JK	3 2 H 0	1 2 3 H	H H H H	2 0 3 1	1 0	1 0	0 1	0 1
13	D	2 0 3 2	1 1 H H	3 2 1 0	H H H H	1 0	1 1	0 0	0 1
14	JK	2 3 0 H	0 2 1 H	H H H H	1 0 3 0	1 0	0 1	1 0	0 1
15	D	0 2 0 0	2 H H 2	H H H H	H 3 1 1	1 1	0 0	0 1	1 0
16	JK	1 H 3 H	2 0 1 1	H 2 1 3	H H H H	1 1	0 0	1 0	0 1
17	D	3 H 2 2	2 0 H 0	1 1 3 0	H H H H	0 0	1 0	1 1	0 1
18	JK	3 3 2 2	1 H 0 1	2 0 1 H	H H H H	1 0	0 1	0 1	1 0
19	D	3 0 H 3	1 2 H 2	1 1 0 0	H H H H	1 0	0 1	0 0	1 1
20	JK	H 2 3 H	H H H H	2 0 1 3	1 3 3 0	1 0	0 0	1 1	0 1
21	D	H 2 3 0	H H H H	1 H 0 2	3 0 1 3	0 0	1 0	0 1	1 1
22	JK	3 2 1 H	H H H H	2 3 H 0	1 H 3 2	0 1	1 0	1 0	0 1
23	D	H H H H	2 1 H 3	3 0 1 H	0 2 1 0	0 1	1 1	1 0	0 1
24	JK	H H H H	2 3 H 0	0 1 0 2	3 H 1 1	0 1	0 1	1 0	1 0
25	D	2 0 3 H	2 2 0 1	H H H H	1 H 3 0	1 0	0 0	1 1	0 1

Дополнительные сведения.

Методические указания по моделированию работы КА

1. При моделировании в ModelSim в настройках конфигурации моделирования дополнительно включите библиотеку altera_ver, поддерживающую базовые аппаратные примитивы, используемые в проекте, в том числе dffeas.
2. На рисунке 4.13 представлен пример временной диаграммы, полученной в результате теста КА. На временной диаграмме отображены все штатные переходы в графе КА, а также все необходимые для отладки промежуточные сигналы. Полный тест дополнительно должен включать в себя моделирование доопределённых при синтезе переходов.

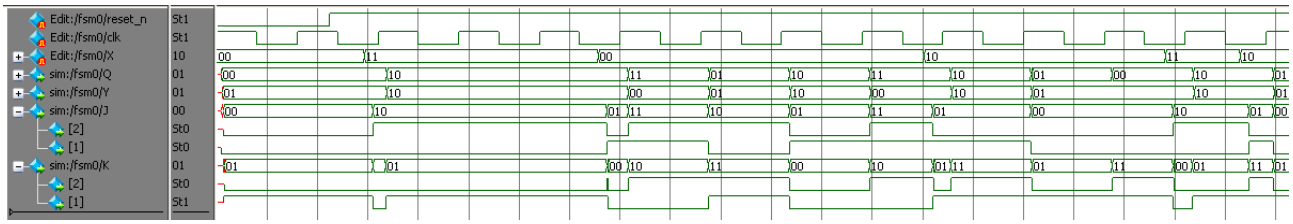


Рисунок 4.13 - Пример результатов полного теста КА

3. ModelSim позволяет отображать состояния КА в символьном виде (user defined radix). Для этого на основе известной информации о кодировании состояний КА необходимо задать символьную таблицу командой radix, например:

```
radix define States {
11'b0000 "r0",
11'b0011 "r1",
11'b0101 "r2",
11'b1001 "r3",
-default hex }
```

Затем объединить наблюдаемые разряды (4 в этом примере) с помощью команды Combine signals и задать способ отображения Radix типа States, определённый показанной выше командой.