МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ

РЯЗАНСКИЙ ГОСУДАРСТВЕННЫЙ
РАДИОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

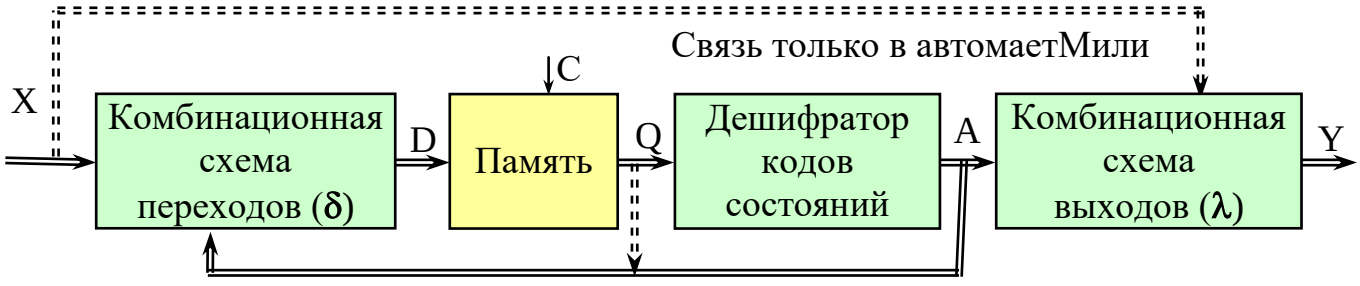
КАФЕДРА ЭВМ

А. В. Кистрин

Проектирование цифровых устройств

Методические указания к лабораторным работам,
практическим и самостоятельным занятиям
по курсам «Схемотехническое проектирование»,
«Проектирование цифровых устройств».

Часть 2



Обобщенная структурная схема конечного автомата

Рязань 2021

Работа 1. Синтез автоматов Мили и Мура

Теоретическая часть

Цифровое устройство, содержащее элементы памяти, называется последовательностной схемой или цифровым автоматом. В таком устройстве последовательность выходных сигналов определяется последовательностями предыдущих состояний схемы и входных сигналами. Цифровые автоматы предназначены для выполнения целенаправленных действий без непосредственного участия человека.

Абстрактный автомат – математическая модель с неограниченными (бесконечными) ресурсами, предназначенная для теоретического анализа функциональных возможностей и методов описания автоматов.

Конечный автомат (Fine State Machine - FSM) – модель реального проектируемого устройства, все параметры которого конечны и определена элементная база. Большое количество типов цифровых устройств с памятью синтезируют как конечные автоматы.

Наиболее полная обобщенная структурная схема конечного автомата (вариант1 на рис. 1.1) содержит память и комбинационные схемы определенного функционального назначения. На схеме показаны все внешние и внутренние сигналы: X - входные сигналы автомата; D -входные сигналы триггеров; Q-выходные сигналы триггеров; A- коды состояний; Y-выходные сигналы триггеров.

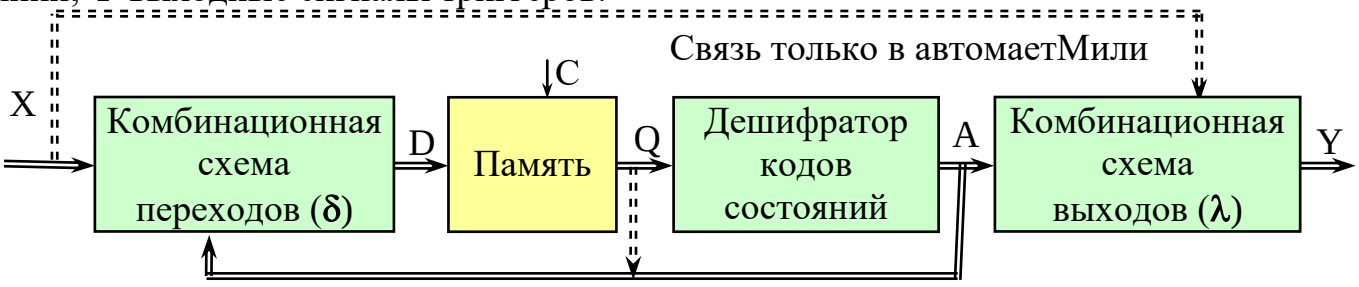


Рис.1.1. Обобщенная структурная схема конечного автомата (вариант 1)

Поясним назначение элементов обобщенной структурной схемы.

Комбинационная схема переходов формирует входные сигналы элементов памяти выбранного типа (например, D), необходимые для перехода автомата из текущего состояния a_m в новое состояние a_s: $d_{ms} = \delta (a_m, x_m)$.

Индекс m от слова master –первый относится к текущему состоянию, **индекс s** от slave- второй – к новому состоянию, d_{ms} обозначает входной сигнал D-триггера дл перехода из текущего состояния в новое.

В качестве сигнала, определяющего состояние, на данную схему может быть подан выходной сигнал триггеров памяти Q.

Память автомата предназначена для хранения кодов состояний, представляет собой набор из n триггеров (например, типа D), позволяющих запомнить не менее 2^n различных состояний. Изменения состояний происходят под управлением входных сигналов (D) в моменты появления импульсов синхронизации C. В качестве элементов памяти используют триггеры с динамическим управлением типов D, T, JK, RS.

Существуют различные способы кодирования состояний автомата. При двоичном кодировании разрядность памяти минимальна. Число триггеров n, используемых для реализации модуля памяти автомата, определяется числом состояний автомата M и способом их кодирования.

При единичном кодировании по принципу: одно состояние — один триггер

(One Hot State) количество триггеров равно количеству состояний n = M.

Дешифратор кодов состояний преобразует двоичные коды, получаемые с выходов триггеров (Q) в единичные коды, определяющие состояниям (A). Наличие в схеме дешифратора кодов состояний позволяет использовать различные способы кодирования выходных сигналов триггеров, а также приводит к упрощению КС переходов и выходов..

Комбинационная схема выходов (λ) формирует выходные сигналы Y.

В автомате Мили (Mealy machine) выходные сигналы в зависят от текущего состояния A и от текущих входных сигналов : $\mathbf{y}_m = \lambda[\mathbf{a}_m, \mathbf{x}_m]$. Автомат Мили содержит связь, показанную па рис. 1.1 пунктирной линией.

В автомате Мура (Moore machine) выходные сигналы зависят только от текущего состояния автомата : $\mathbf{y}_m = \lambda[\mathbf{a}_m]$

В структурах конкретных реализаций могут отсутствовать дешифратор кодов состояний и комбинационная схема выходов.

Приведенная структура является наиболее полной. В структурах конкретных реализаций могут отсутствовать дешифратор кодов состояний и комбинационная схема выходов.

Синтез конечного автомата

Содержит 6 этапов. Рассмотрим пример.

Задано синтезировать автомат, который имеет входные сигналы x1. x2 и выходные сигналы y1, y2, y3. Если x1 = 0, то y1 = y2 = y3 =0. После появления входного сигнала x1=1 автомат должен включить один из выходных сигналов указателей поворота. Если x2=0, то левый – y1, иначе правый y2. Затем должен быть включен выходной сигнал на панели приборов y3. Выключение всех сигналов должно выполняться одновременно.

1 этап. Разработка содержательной ГСА, отмеченной ГСА, кодирование всех сигналов.

Содержательная граф- схема алгоритма **(ГСА)** строится на основе анализа функции, выполняемой автоматом Y=F(X). Операторные вершины ГСА соответствуют выполняемым функциям, а условные вершины соответствуют входным сигналам X. Каждая команда, содержащаяся в операторной вершине, выполняются за один такт синхросигнала. Одна операторная вершина может содержать несколько совместимых микрокоманд, выполняемых одновременно. Вершинам алгоритма «начало» и «конец» назначают выходной сигнал y0, в результате алгоритм становится циклическим. Этим вершинам соответствует начальный сброс памяти автомата, который в ПЛИС выполняется автоматически.

Отмеченная ГСА строится по содержательной ГСА.

На отмеченной ГСА указывают состояния автомата. которые определяют порядок выполнения операторов. В операеорных вершинах указывают выходные сигналы автоиата, а в условных – входные.

В автомате Мили состояния отмечают черточками на дугах . За состояние принимают выход одного, или нескольких исполняемых операторов. Состояние a0 назначают выходу оператора «начало» и входу оператора «конец».

В автомате Мура за состояния принимают исполняемые операторы алгоритма.

Отмеченный алгоритм определяет количество состояний автомата M и разрядность памяти (количество триггеров) m. При двоичном кодировании состояний разрядность памяти минимальна и определяется по формуле . m =]$\log_2$M[, где скобки] [обозначают операцию округления до ближайшего большего целого. - для автомата Мили M = 3, m = 2,а для автомата Мура M = 5, m = 3.

Для сигналов А (см. рис. 1.1) используют единичное кодирование, позволяющее упростить комбинационные схемы переходов и выходов. Дешифратор кодов состояний преобразует двоичный код Q в единичный код состояний А. Разрядность сигналов А определяет количество состояний М.

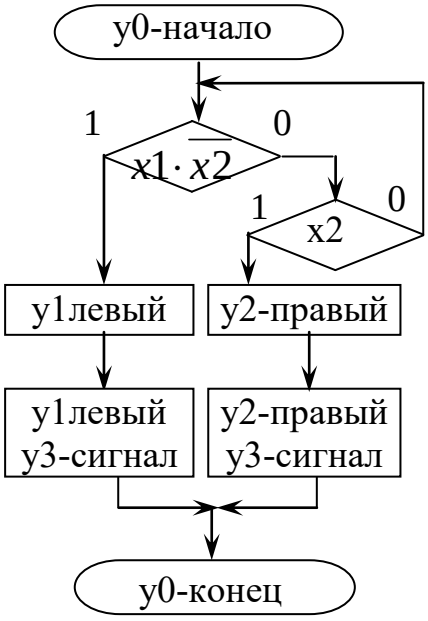


Рис. 1.2.
Содержательная ГСА

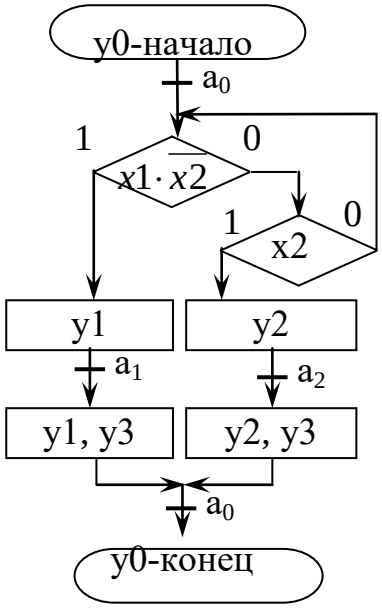


Рис. 1.3. Отмеченная
ГСА Мили

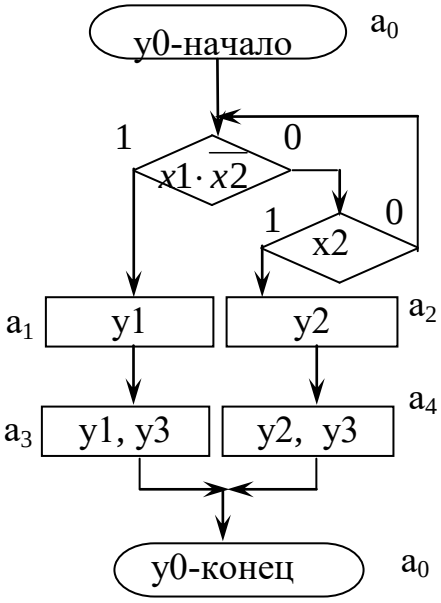


Рис. 1.4. Отмеченная
ГСА Мура

Сигналы X и Yпредставляют в виде отдельных независимых проводников.

2 этап.. Разработка графа автомата.

Граф **автомата составляется по отмеченной ГСА,** содержит вершины, соответствующие состояниям, и дуги, соответствующие переходам.

Для автомата Мили в изображениях вершин указывают состояния, на начале дуги отмечают входной сигнал, а на конце – выходной сигнал. Все возможные пути из одного состояния в другое должны быть отображены дугами.

Для автомата Мура в изображениях вершин указывают состояния и выходные сигналы, а на дугах – входные сигналя, являющиеся условиями перехода. Для безусловных переходов записывают единицу.

Автомат Мили имеет, как правило, меньшее количество состояний, чем автомат Мура. Он обеспечивает экономию аппаратных затрат.

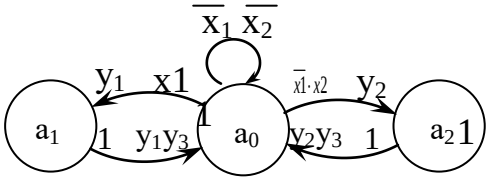
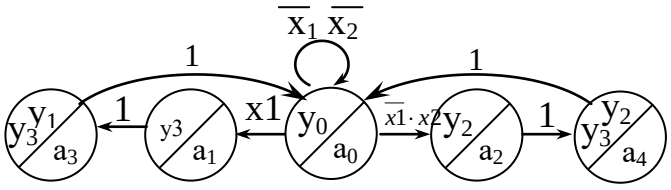


Рис. 1.5. Граф автомата Мили



Г Рис. 1.6 .раф автомата Мура

3 этап. Составление таблицы переходов и выходов по графу.

Этот вид таблицы (называемый «обратная совмещенная») используют при разработке автоматов на основе ПЛИС. Каждая строка таблицы соответствует одной дуге графа. Столбцы соответствуют сигналам (см. рис. 1.1). Столбцы 1 - 4 выделены, они содержат сигналы, представленные на графе, соответствующие описываемой дуге:

Индекс m (Master-первый) обозначает исходное состояние автомата; индекс s (/Slave – второй) обозначает новое состояние.

Столбец x_m – входные сигналы (условия) для текущего состояния. Если переход, безусловный, то в этом столбце записывается прочерк или константа 1.

Столбец a_m – текущие состояния, из которых начинается переход,

Столбец a_s – новые состояния, в которые переходит автомат ;

Столбец y_m – выходной сигнал, автомата в текущем состоянии.

Таблица 1.1

	Сигналы графа Мили				Коды в схеме			
№	1	2	3	4	5	6	7	8
	x_m	a_m	a_s	y_m	q_m $q_1 q_0$	q_s $q_1 q_0$	d_{ms} $d_1 d_0$	a_m $a_2 a_1 a_0$
1	$\overline{x1} \cdot \overline{x2}$	a_0	a_0	-	0 0	0 0	0 0	0 0 0
	x1	a_0	a_1	y1	0 0	0 1	0 1	0 0 0
2	$\overline{x1} \cdot x2$	a_0	a_2	y_2	0 0	1 0	1 0	0 0 0
3	1	a_1	a_0	y_1,y_3	0 1	0 0	0 0	0 1 0
	1	a_2	a_0	y_2,y_3	1 0	0 0	0 0	1 0 0

Столбцы 5, 6 содержат двоичные коды на выходах триггеров в, соответствующие выбранной разрядности.

Столбец 7 содержит входные сигналы триггеров (например, d_{ms}), необходимые для перехода из текущего состояния (q_m) в новое состояние (q_s).

В данном примере выбраны D – триггеры, таблица переходов которых (таблица 1.2) показывает, что для перехода триггера в состояние q_s необходимо подать на его вход точно такой же сигнал ($d_{ms} = q_s$), поэтому коды в столбце 7 повторяют коды столбца 6.

Столбец 8 (может отсутствовать) содержит единичные коды текущих состояний.

Таблица 1.2

	Сигналы графа Мура				Коды в схеме			
№	1	2	3	4	5	6	7	8
	x_m	a_m	a_s	y_m	q_m $q_2 q_1 q_0$	q_s $q_2 q_1 q_0$	d_{ms} $q_2 d_1 d_0$	a_m $a_4 a_3 a_2 a_1 a_0$
	$\overline{x1} \cdot \overline{x2}$	a_0	a_0	0	0 0 0	0 0 0	0 0 0	0 0 0 0 1
1	x1	a_0	a_1	0	0 0 0	0 0 1	0 0 1	0 0 0 0 1
2	$\overline{x1} \cdot x2$	a_0	a_2	0	0 0 0	0 1 0	0 1 0	0 0 0 0 1
3	1	a_1	a_3	y_1	0 0 1	0 1 1	0 1 1	0 0 0 1 0
	1	a_2	a_4	y_2	0 1 0	1 0 0	1 0 0	0 0 1 0 0
	1	a_3	a_0	$y_1 y_3$	0 1 1	0 0 0	0 0 0	0 1 0 0 0
	1	a_4	a_0	$y_2 y_3$	1 0 0	0 0 0	0 0 0	1 0 0 0 0

Логические функции переходов выполняет функцию: $d_{ms} = \delta (\mathbf{x_m}, \mathbf{a_m})$, предназначены для синтеза КС КС переходов (см. рис. 1.1). Входными сигналами являются текущие значения входного сигнала автомата и текущие состояния, а выходными - управляющие сигналы триггеров, обеспечивающие требуемый переход. Для записи логической функции перехода вначале выделяется столбец для одного из разрядов управляющего сигнала триггера (например, d0 в столбце 7 - d_{ms}), в котором определяются строки, содержащие 1. Для каждой строки записывается произведение сигналов, содержащихся в столбцах 1 и 2 (это x_m и a_m). Произведения объединяются операцией дизъюнкции. Данные операции повторяются для всех разрядов сигнала D.

Таблица 1.3

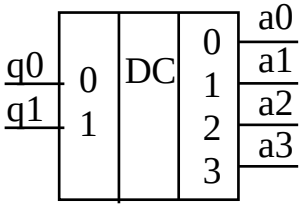
Переход	J	K	R	S	D	T
0→0	0	x	x	0	0	0
0→1	1	x	0	1	1	1
1→0	x	1	1	0	0	1
1→1	x	0	0	x	1	0

Таблицы переходов триггеров

Функции переходов автомата Мили получим: $d_0 = a_0 \cdot x_1$; $d_1 = a_0 \cdot \overline{x_1} \cdot x_2$. (1),

- автомата Мура: $d_0 = a_0 \cdot x_1 \vee a_1; \quad d_1 = a_0 \cdot \overline{x_1} \cdot x_2 \vee a_1; \quad d_2 = a_2.$ (2)

Дешифратор кодов состояний преобразует двоичные коды, получаемые с выходов триггеров (Q) в единичные коды, определяющие состояниям (A). Для кодирования состояний памяти Q могут использоваться различные способы, например, код Грея, позволяющий уменьшить вероятность гонок, либо выбор кодов, содержащих меньшее количество единиц для вершин графа, в которые входит много дуг, что упрощает аппаратную реализацию.



q1 q0	a3 a2 a1 a0
0 0	0 0 0 1
0 1	0 0 1 0
1 0	0 1 0 0
1 0	1 0 0 0

Рис.1.7 Дешифратор кодов состояний

При двоичном кодировании чаще используется выбор выходных кодов памяти из натурального ряда чисел. В этом случае логические известные функции для дешифратора 2-разрядных кодов состояний (рис 1.7) имеют вид:

$$a_3 = q_1 \cdot q_0; \quad a_2 = q_1 \cdot \overline{q_0}; \quad a_1 = \overline{q_1} \cdot q_0; \quad a_0 = \overline{q_1} \cdot \overline{q_0}. \qquad (3)$$

В приведенных примерах используются неполные дешифраторы: для автомата Мили й – 2 входа, 3 выхода, а для автомата Мура – 3 входа, 5 выходов.

Комбинационная схема выходов формирует выходные сигналы Y.

В автомате Мили (который содержит связь, показанную па рис. 1.1 пунктирной линией) выходные сигналы являются функцией текущего состояния и от текущих входных сигналов : $y_m = \lambda[a_m, x_m]$.

Для записи **логической функции выхода** автомата Мили выбирается один из сигналов. Для строк, содержащих выбранный сигнал в столбце 4, записываются произведения сигналов, содержащихся в столбцах 1 и 2 (это x_m и a_m), которые соединяется операцией дизъюнкция.

$$y_1 = a_0 \cdot x_1 \vee a_1; \quad y_2 = a_0 \cdot x_2 \vee a_2; \quad y_3 = a_1 \vee a_2. \qquad (4)$$

В автомате Мура (Moore machine) выходные сигналы зависят только от текущего состояния автомата: $y_m = \lambda[a_m]$, поэтому операция записи погической функции упрощается. Для строк, содержащих выбранный сигнал в столбце 4, записывается сигналы состояний, содержащиеся в столбце 2 (это $\mathbf{a}_m$).

$$y_1 = a_1 \vee a_3; \quad y_2 = a_2 \vee a_4; \quad y_3 = a_2 \vee a_4. \qquad (5)$$

4 этап. Разработка схемы автомата. Схема автомата составляется в соответствии со структурной схемой (рис. 1.1)с учетом выбранного типа триггера и логических функций. Необходимо изобразить терминалы для входных сигналов (с, х) и выходных сигналов (y,q), а также триггеры, к входам которых подключены КС переходов , а к выходам – дешифратор кодов состояний и , затем, КС выходов.

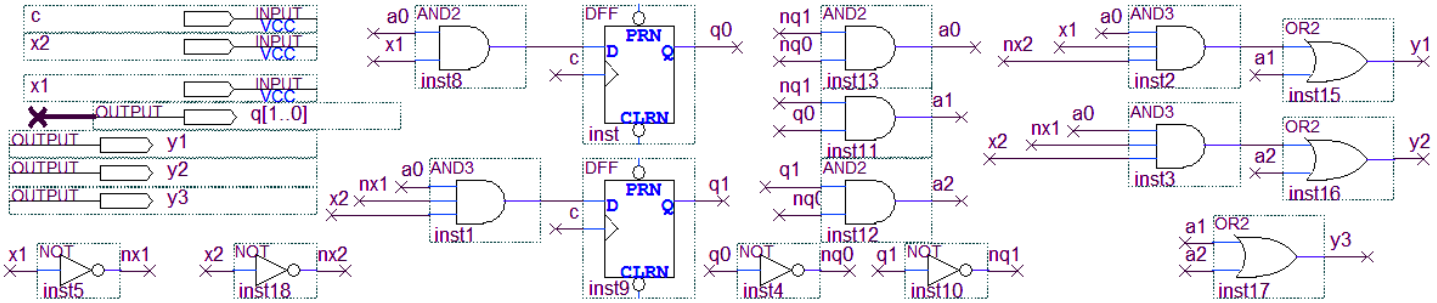


Рис. 1.8. Схема автомата Мили (модуль s11_mealy)

В схеме автомата Мили (рис. 1.8) содержатся терминалы входных и выходных сигналов (с, х, у) в виде отдельных проводников, которые необходимы для включения

автомата в работу. Для тестирования и отладки схемы предназначен терминал шины q выводить сигналы с выхода элементов памяти Q. Для остальных сигналов терминалы могут отсутствовать.

Схема автомата Мура (рис. 1.9) отличается разрядностью памяти.

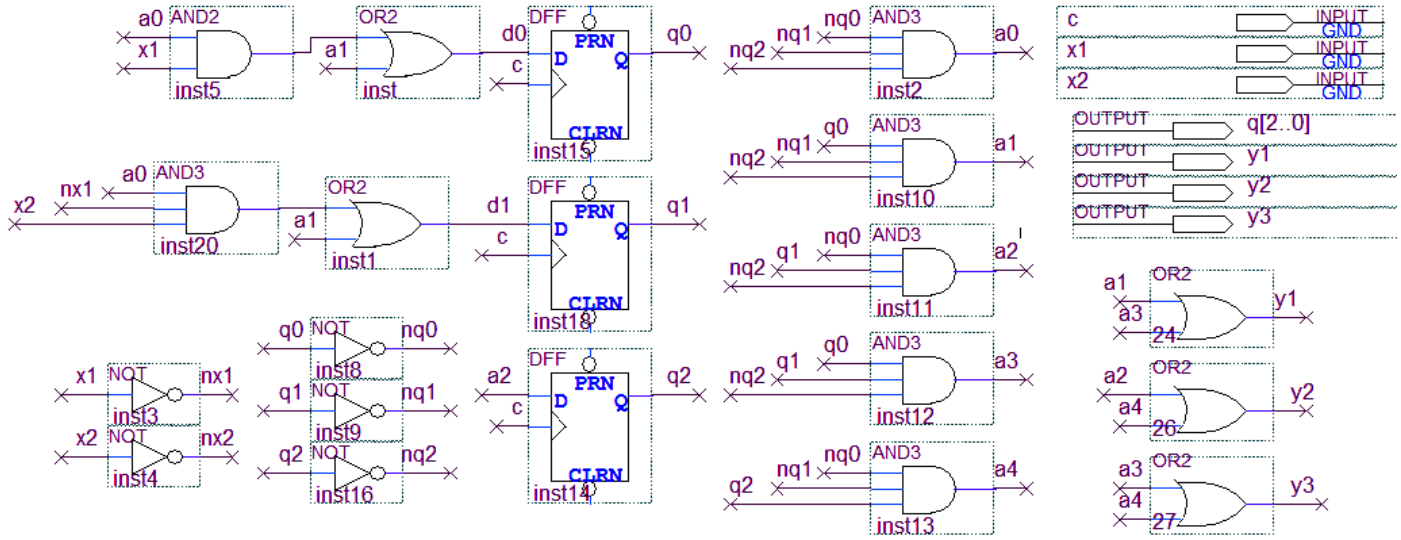


Рис. 1.9. Схема автомата Мура (модуль s11_moore)

5 этап. Тестирование автомата. Предназначено для проверки работы автомата при заданных комбинациях входных сигналов и в соответствии выходных сигналов заданному алгоритму. При моделировании оставляются тестовые сигналы, позволяющие определить возможные неисправности.

Таблица тестовых сигналов

x2	x1	Состояния Мили	Состояния Мура
0	0	0	0
0	1	0 – 1 – 0 – 1 – 0 – 1	0 – 1 – 3 – 0 – 1 – 3 –
1	0	0 – 2 – 0 – 2 – 0 – 2.	0 – 2 – 4 – 0 – 2 – 4 –

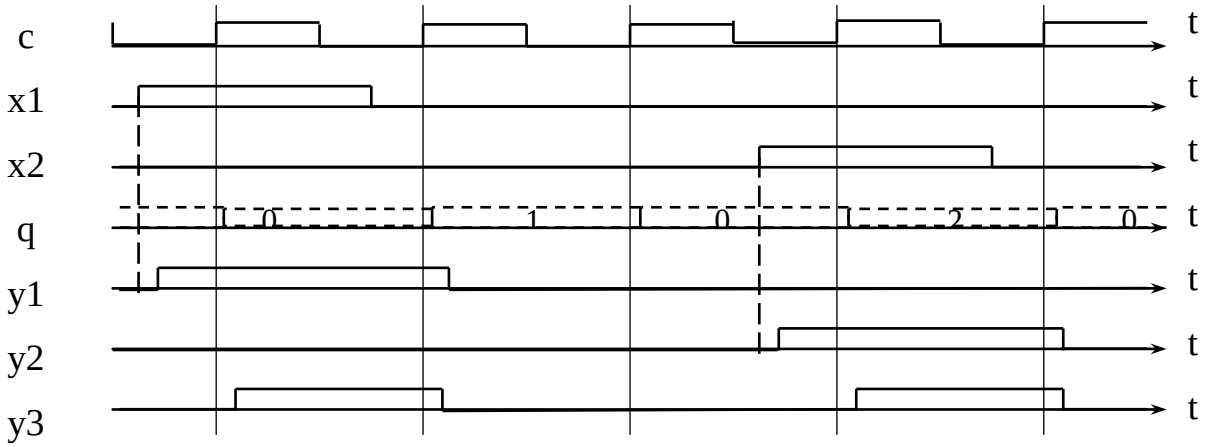


Рис. 1. 10 Теоретические временные диаграммы автомата Мили

1. Проверка правильности рабаты схемы памяти и КС переходов автомата. По графу составляется таблица тестовых сигналов, которая содержит последовательности состояний при заданных входных сигналах. При моделировании используются статические состояния входных сигналов (различные комбинации 0 и 1) и анализируется состояния выходных сигналов памяти.

2. Проверка КС выходов (правильности формирования выходных сигналов). Составляются теоретические временные диаграммы, на которых вначале изображается

синхросигнал – импульсы типа «меандр» и выделяются моменты времени, соответствующие фронтам импульсов. Затем изображаются диаграммы входных импульсов, длительность которых соизмерима с периодом синхросигнала, а начало соответствует моменту времени после переднего фронта синхросигнала.

Диаграммы для выходных сигналов строятся с использованием логических функций выходов в соответствии с физикой работы автомата.

Выходные сигналы автомата Мили начинаются после подачи входных сигналов (с небольшой задержкой в элементах), а заканчиваются по фронту синхросигнала (рис. 1. 10). Выходные сигналы автомата Мура начинаются и заканчиваются в моменты времени, определяемые фронтами синхросигнала(рис. 1. 11).

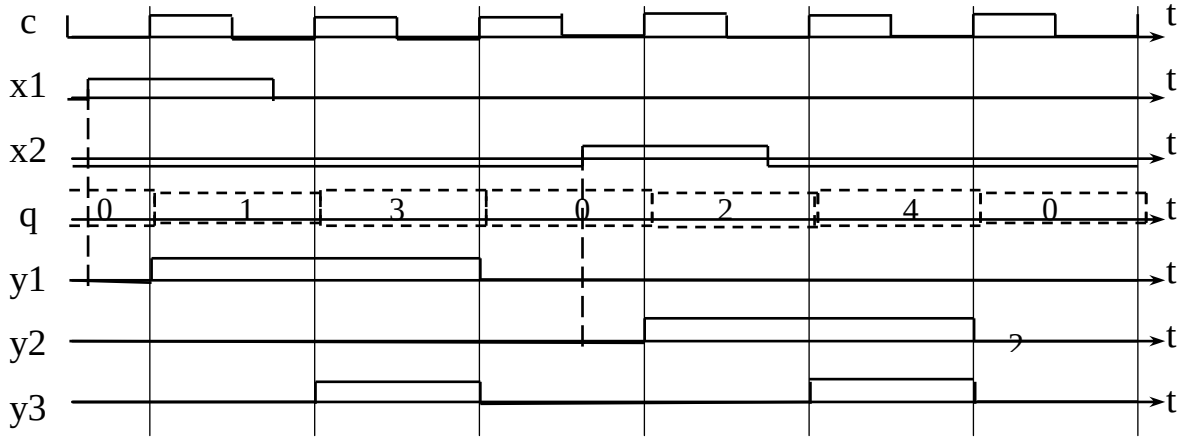


Рис. 1.11 Теоретические временные диаграммы автомата Мура

Задание 1.1. Разработайте иерархический проект по схеме с именем s11_fsm, содержащий два модуля нижнего уровня иерархии, построенные по схемам (рис. 7 и рис. 8) с именами: s11_mealy и s11_moore. Для каждого модуля 1) создайте файл для ввода схемы, 2) введите схему с указанным именем, 3)создайте проект, 4) выполните компиляцию, 5) создайте символ (этапы 1-5 в справочном материале).

Разработайте модуль s11_fsm по схеме (рис. 1.12).

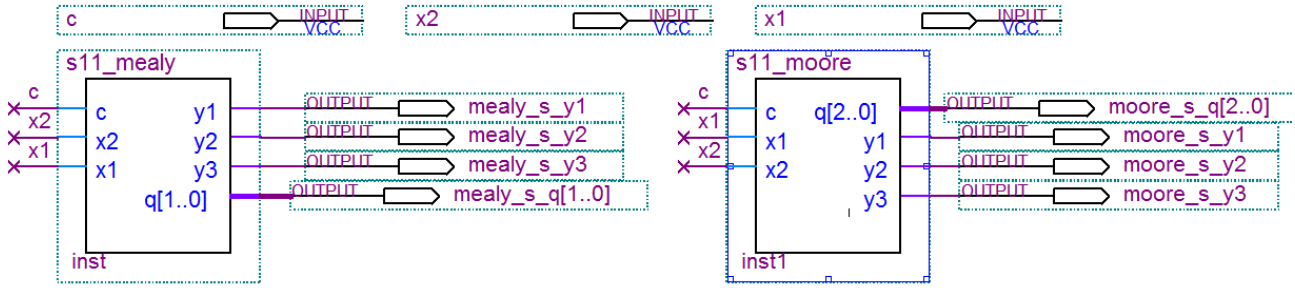


Рис.1.12. Схема проекта верхнего уровня иерархии s11_fsm

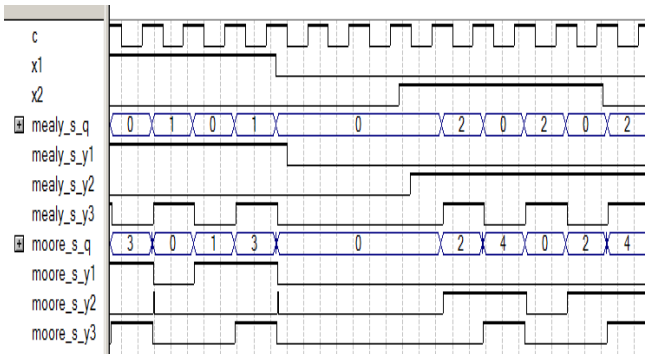


Рис. 1.13. тест памяти и КС переходов

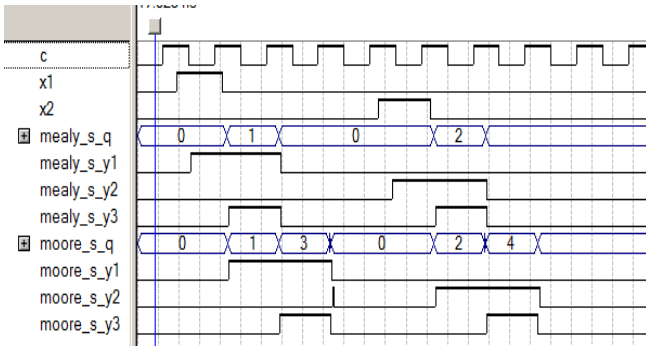


Рис. 1.14. Тест КС выходов

Выполните моделирование, сравните выходные сигналы автоматов Мили и Мура.

Определите временные задержки. Рассчитайте максимальную частоту, при которой сохраняется работоспособность. Экспериментально проверьте полученный результат.

В отчете подробно опишите полученные временные диаграммы.

6 этап. Описание конечного автомата на языке Verilog. При описании используют второй вариант структурной схемы автомата (рис. 1.15), в котором сокращено количество внутренних сигналов, при этом состояния автомата определяют выходные коды триггеров. Описание конечного автомата содержит две части, это описание памяти и описание комбинационных схем.

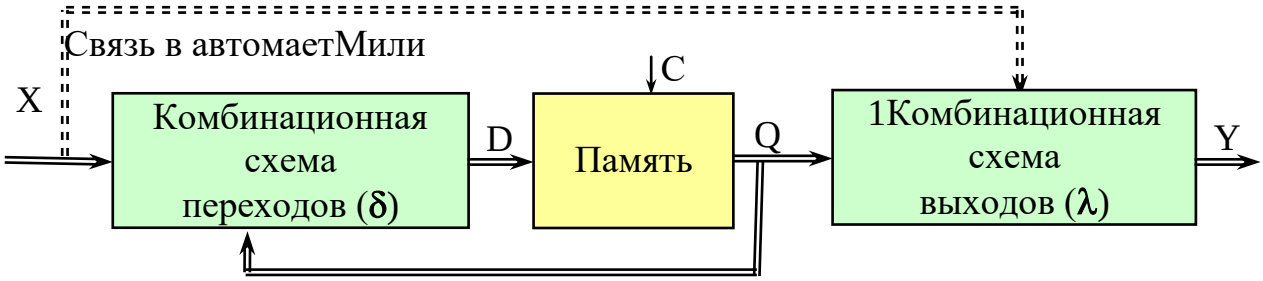


Рис.1.15. Сокращенная структурная схема конечного автомата (вариант 2)

Описание памяти, которая построена на синхронных триггерах с динамическим управлением, выполняется последовательными операторами с ключевым словом always, для которых указывают срабатывание по фронту синхросигнала. После ключевого слова могут содержаться операторные скобки begin – end и операторы case и if.

Простое для восприятия описание получается, если для переходов автомата из одного состояния в другое используется оператор варианта case, в котором в качестве селектора используются исходные состояния автомата, а в качестве выполняемого действия - присваивания нового состояния. Селектор оператора варианта – это переменная, записываемая в скобках после ключевого слова case, которая определяет выбранный вариант. Присваивание выполняется, если селектор равен 1. Строки оператора case составляются по графу автомата. Для безусловного перехода из одного состояния в другое строка имеет вид: <константа – номер исходного состояния> <двоеточие> <оператор присваивания номера нового номера>. Для записи номеров может использоваться десятичная система.

Если переход автомата из одного состояния в другое зависит от входного сигнала, то используется условный оператор if, содержащий в качестве условия входной сигнал. В результате операторные скобки case – endcase будут содержать описание работы памяти и КС переходов (см.рис.1.1).

Описание комбинационных схем выходов конечного автомата выполняют параллельные операторы с ключевым словом «assign», составленные по логическим функциям выходов. Для упрощения описания из структуры автомата (рис. 1.1) исключается дешифратор кодов состояний. В функции переходов (4) и (5) вместо сигналов a_m подставляются соответствующие сигналы q_m. В описании будут использованы следующие функции.

Для автомата Мили функции $y_m = \lambda[q_m, x_m]$ имеет вид:

$$y_1 = \overline{q_1} \cdot \overline{q_0} \cdot x_1 \vee \overline{q_1} \cdot q_0; \quad y_2 = \overline{q_1} \cdot \overline{q_0} \cdot x_2 \vee a_2 = q_1 \cdot \overline{q_0}; \quad y_3 = \overline{q_1} \cdot q_0 \vee a_2 = q_1 \cdot \overline{q_0}. \quad (6)$$

Для автомата Мура функции $y_m = \lambda[q_m]$:

$$y_1 = q_2 \cdot q_1 \cdot q_0 \vee q_2 \cdot q_1 \cdot q_0; \quad y_2 = q_2 \cdot q_1 \cdot q_0 \vee q_2 \cdot q_1 \cdot q_0; \quad y_3 = q_2 \cdot q_1 \cdot q_0 \vee q_2 \cdot q_1 \cdot q_0. \quad (7)$$

```
// Описание  автомата Мили
module v12_mealy
(c,x1, x2,q,y1,y2,y3);
input c, x1, x2;
output [1:0]q;   reg [1:0]q;
output y1,y2,y3;
always @ ( posedge c)
case (q)
2'b00:  if(~x1&~x2) q=2'b00;
                else if(x1&~x2) q=2'b01;
                              else   q=2'b10;
2'b01: q=2'b00;
2'b10: q=2'b00;

endcase
assign
y1=~q[1]&~q[0]&x1&~x2|~q[1]&q[0];
assign
y2=~q[1]&~q[0]&~x1&x2|q[1]&~q[0];
assign y3=~q[1]&q[0]|q[1]&~q[0];
endmodule
```

```
// Описание автомата Мура
module v12_moore(c,x1, x2,q,y1,y2,y3);
input c, x1, x2;
output [2:0]q;   reg [2:0]q;
output y1,y2,y3;
always @ ( posedge c)
case (q)
3'b000:  if(~x1&~x2) q=3'b000;
        else if(x1&~x2) q=3'b001;
                        else   q=3'b010;
3'b001: q=3'b011;
3'b011: q=3'b000;
3'b010: q=3'b100;
3'b100: q=3'b000;
endcase
assign y1=~q[2]&~q[1]&q[0]|
~q[2]&q[1]&q[0];
assign y2=~q[2]&q[1]&~q[0]|
q[2]&~q[1]&~q[0];
assign y3=~q[2]&q[1]&q[0]|
q[2]&~q[1]&~q[0];
endmodule
```

Задание 1.2. Разработайте иерархический проект по схеме с именем s12_fsm, содержащий модули нижнего уровня иерархии, построенные по описаниям v12_mealy и v12_moore. Для каждого модуля нижнего уровня 1) создайте файл для описания с именем, записанным в описании, 2) создайте проект, 3) введите описание, 4) выполните компиляцию, 5) создайте символ. Создайте схему проекта верхнего уровня в соответствии с рис. 1.15. Выполните моделирование.

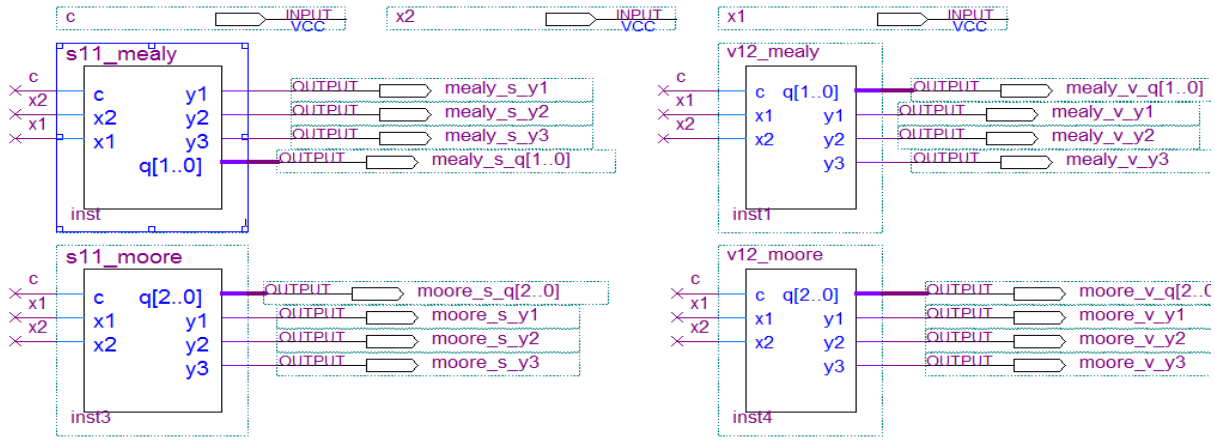


Рис. 1. 16. Схема проекта верхнего уровня иерархии s12_fsm

Задание 1.3. Разработайте проект автомата Мура по схеме с именем s13_fsm, используя сокращенную структуру автомата (рис. 1.15) и триггеры типа T. Выполните моделирование. Опишите полученные временные диаграммы.

Контрольные вопросы

1. Изобразите обобщенную структурную схему конечного автомата.
2. Поясните назначение и функции всех модулей конечного автомата.
3. Перечислите этапы синтеза конечного автомата.
4. Что такое содержательная ГСА?
5. Как составляется отмеченная ГСА?
6. Как составляются графы автоматов Мили и Мура по ГСА.
7. Как составляется таблица переходов по графу?
8. Как составляются логические функции переходов и выходов по таблице переходов автомата?
9. Поясните методику составления описания конечного автомата на языке Verilog.
10. Как составляется таблица тестовых сигналов?
11 Как составляются теоретические временные диаграммы?

Работа 2. Синтез счетчиков

Теоретическая часть

Метод синтеза конечных автоматов позволяет проектировать различные счетчики с заданным модулем счета и любым порядком смены состояний, например, двоичные, двоично-десятичные, реверсивные, счетчики в коде Грея. В качестве элементов памяти используются триггеры с динамическим управлением.

Задание 2.1. Синтезируйте двоичный реверсивный счетчик с модулем счета 3, на триггерах типа JK в виде автомата Мура. Направление счета определяет входной управляющий сигнал m (от слова mminus), при m = 1 вычитание, при m = 0 – суммирование. Используйте в качестве выходных сигналов автомата выходные сигналы триггеров Q, которые будут также определять состояния.

Разработайте иерархический проект, содержащий два модуля нижнего уровня иерархии, один из которых с именем s21_cnt_mod3 построен по схеме, а второй с именем v21_cnt_mod3 – по описанию на Verilog. Для каждого модуля нижнего уровня создайте файл для ввода данных, создайте проект, введите данные, выполните компиляцию, создайте символ (этапы 1-5 в справочном материале).

Создайте проект верхнего уровня с именем sv21_cnt_mod3. Выполните моделирование, сравните выходные сигналы этих модулей. Определите временные задержки. Рассчитайте максимальную частоту, при которой сохраняется работоспособность. Экспериментально проверьте полученный результат.

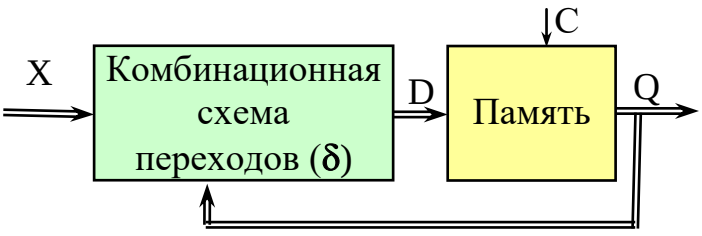


Рис.2.1. Структурная схема счетчика (вариант 3)

Рассмотрим пример синтеза реверсивного счетчика с модулем счета 3

1 этап. Разработка алгоритма, определение разрядностей и способов кодирования всех сигналов.

В данном случае структура автомата предельно упростится (Рис. 2.2.1). Для выходных двоичных кодов заданного реверсивного счетчика с модулем 3 в режиме суммирования выберем последовательность q_0=00, q_1=01, q_2=10, q_0=00. а в режиме вычитания – другую последовательность: $q_0 – q_2 – q_1 – q_0$. В алгоритме этим кодам соответствуют исполняемые операторы.

Алгоритм является циклическим, вершины начало и конец считаются соединенными. Каждому исполняемому оператору в автомате Мура отмечают определенное состояние, определяемое кодом Q (рис. 2.2). Входные сигналы автомата содержатся в условных операторах алгоритма.

По алгоритму определяется количество состояний, количество триггеров памяти и разрядность всех сигналов: количество состояний – 3, количество триггеров и разрядность сигналов j, k, q – 2.

2. Гграф автомата Мура**,** в соответствии с алгоритмом, содержит вершины, соответствующие состояниям и дуги, отображающие входные сигналы - условия переходов автомата из одного состояния в другое. Граф реверсивного счетчика с модулем три имеет три состояния.

3. Таблица переходов автомата (таблица .2.1) составляется по графу. Столбцы 1 - 3 таблицы выделены, они содержат сигналы, показанные на графе. Каждая строка таблицы соответствует одной дуге графа.

Столбцы 4–5 содержат коды выходных сигналов триггеров, записаны в соответствии с состояниями (столбцы 2, 3). В столбцах 6-7 записываются коды входных сигналов триггеров, необходимое для перехода из текущего состояния (q_m) в новое состояние (q_s). Для определения входных сигналов триггеров используются текущее и новое значения сигналов и таблица переходов используемого триггера (Таблица 1.2 в работе 1).

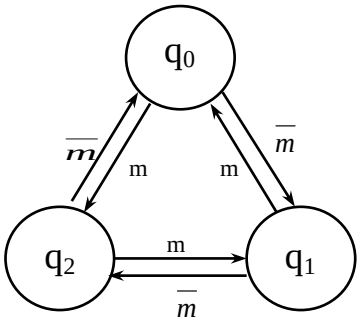


Рис. 2.3. Граф автомата Мура

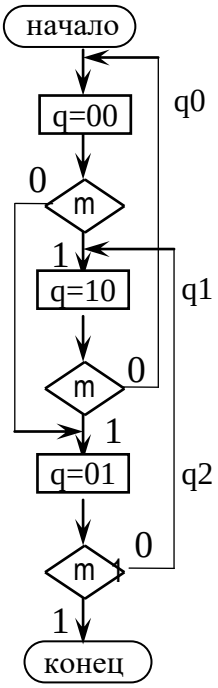


Рис. 2.2. Отмеченная ГСА счетчика

Пояснение. Первая строка таблицы 2.1 соответствует состоянию сброса счетчика, коды на выходах триггеров $q_m = q_s = 0$. Оба триггера выполняют переход 0→0, для которого, в соответствии с таблицей 1.2, требуется подать j = 0, k = x. В строке 2 для старшего разряда требуется переход 0→0, поэтом у $j_1=0$, $k_1 = x$, а для младшего разряда требуется переход 0→1, поэтом у $j_0=1$, $k_0 = x$. Подобным образом записаны остальные строки.

Составление логических функций переходов. Выделяется столбец значений для одного из входных сигналов триггера. Для строк, содержащих значения 1 записываются произведения $x_m \cdot q_m$, которые соединяются знаками дизъюнкции. Получается логическая функция в СДНФ. Неопределенные значения «х» для сигналов j_1 и j0 заменяются, нулями, , а для $к_1$ и $к_0$ единицами.

$$j0 = \overline{m} \cdot \overline{q1} \cdot \overline{q0} \vee m \cdot q1 \cdot \overline{q0}; \qquad j1 = \overline{m} \cdot \overline{q1} \cdot q0 \vee m \cdot \overline{q1} \cdot \overline{q0}; \qquad k0 = k1 = 1. \quad .$$

4 этап. Разработка схемы конечного автомата.

Таблица 2.1

№	x_m	q_m	q_s	q_m	q_s	$jk\_a_m a_s$```	
				q_1 q_0	q_1 q_0	j_1 k_1	j_0 k_0
1	$\overline{m}$	q_0	q_1	0 0	0 1	0 x	1 x
2	$\overline{m}$	q_1	q_2	0 1	1 0	1 x	x 1
3	$\overline{m}$	q_2	q_0	1 0	0 0	x 1	0 x
4	m	q_0	q_2	00	1 0	1 x	0 x
5	m	q_2	q_1	10	0 1	x 1	1 x
6	m	q_1	q_0	01	0 0	0 x	x 1
	1	2	3	4	5	6	7

Схема синтезированного счетчика (Рис. 1.5) содержит два JK-триггера, к входам которых подключены комбинационные схемы, составленные в соответствии с логическими выражениями для функций переходов. В качестве выходных сигналов счетчика используются двоичные коды с выходов триггеров.

6. Описание конечного автомата на языке Verilog.

```
//Реверсивный счетчик  mod3
module v21_cnt_mod3 (c, m,  q);        //2
input c, m;                            //3
output [1:0]q;                         //4
reg [1:0]q;                            //5
always @ (posedge c)                   //6
case (q)                               //7
0:      if(m) q=2; else  q=1;          //9
1:      if(m) q=0; else  q=2;          /10
default: if(m) q=1; else  q=0;         /11
endcase       endmodule                //12
```

Описание конечного автомата на языке Verilog выполняется по графу с использованием последовательного оператора с ключевым словом always с и операторов варианта case и операторов условного оператора if.

В приведенном описании реверсивного счетчика, в строке 2 указано имя модуля и перечислены все входные и выходные сигналы. В строке 3 определены входные сигналы, по умолчанию они будут одноразрядными типа «wire». В строках 4, 5 указан выходной сигнал q в виде 2-разрядного вектора типа «reg». Этот тип сигнала всегда формируется последовательными операторами (строки 6 – 11). В строке 11 исходное значение селектора указано как «default». В результате оператор варианта будет срабатывать при любом из оставшихся неиспользованными значениях селектора Q(это 3 и 4.) Такая запись повышает помехоустойчивость устройства. При выполнении этой строки будет выбран любой из вариантов, возникающий при включении схемы, или в результате сбоя. Рабочий цикл будет выполняться.

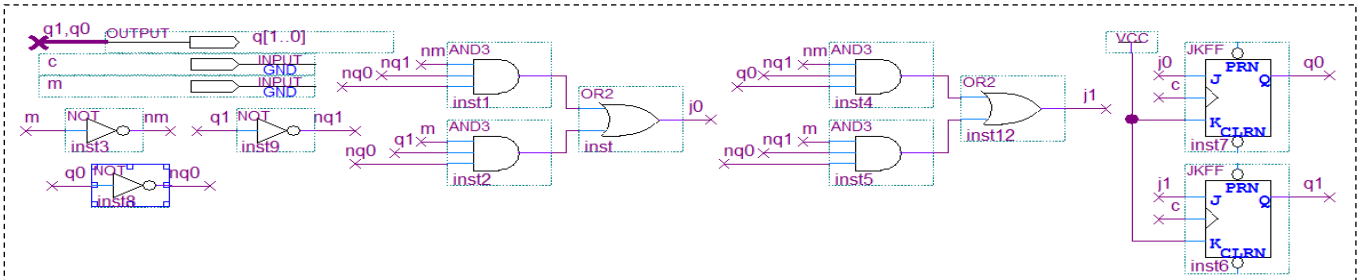


Рис. 2.4. Схема проекта s21_cnt_mod3

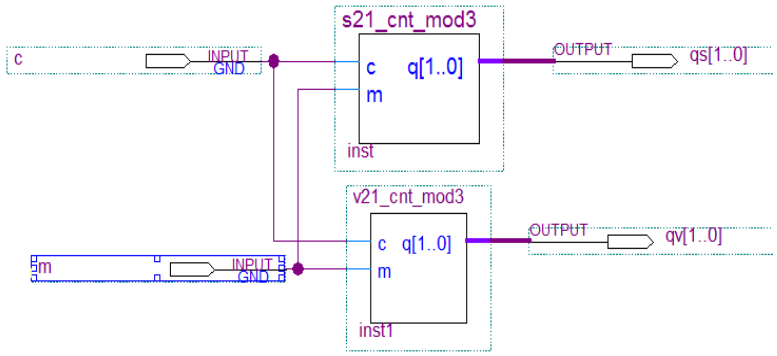


Рис. 2.4. Схема проекта sv21_cnt_mod3

Таблица 2.2

Таблица тестовых сигналов

n	Выходы q
0	0
1	0
0	10 – 01 – 10 - 00 – 01..
1	10 – 10 – 01 - 00 – 10..

7 Тестирование автомата.

Для тестирования автомата необходимо по графу составитель таблицу смены состояний графа при всех комбинациях управляющих сигналов (Таблица 2.2). Данной таблице должны соответствовать временные диаграммы, полученные при моделировании автомата.

Задание 2.2. Синтезируйте двоичный реверсивный счетчик в виде автомата Мура. Модуль счета, и тип триггеров заданы в таблице.

Разработайте иерархический проект, содержащий два модуля нижнего уровня иерархии, один из которых построен по схеме, а второй – по описанию на Verilog. Определите временные задержки. Рассчитайте максимальную частоту, при которой сохраняется работоспособность, полученный результат проверьте экспериментально.

Номер бригады	1	2	3	4	5	6	7	8	9	10
Мщдуль	5	6	7	5	6	7	5	6	7	5
Триггер	JK	T	D	RS	JK	T	D	RS	JK	T

Контрольные вопросы

1. Поясните определение абстрактного автомата.
2. Поясните определение конечного автомата.
3. Какие параметры задают конечный автомат?
4. Изобразите структурную схему конечного автомата.
5. Что такое содержательная ГСА?
6. Что такое отмеченная ГСА?
7. Как составляется граф автомата Мура по ГСА.
8. Как составляется таблица переходов по графу?
9. Как составляются логические функции переходов и выходов по таблице переходов автомата?
10. Поясните методику составления описания конечного автомата на языке Verilog.

Работа 3. Синтез автомата для управления светофором

Задание. 3.1. Выполните синтез конечного автомата Мура для управления светофором в соответствии с структурой варианта 2 (рис. 1.1.15).. При сигнале «старт» рваном 0 должен быть постоянно включен сигнал «желтый». (Для получения мигающего желтого сигнала впоследствии добавляют отдельную схему.) При сигнале «старт» рваном 1 автомат должен периодически переключать сигналы: желтый – зеленый – желтый – зеленый – красный. Если нажата кнопка «К», то после сигнала «красный» должен дополнительно включиться сигнал «переход».

Разработайте иерархический проект с именем sv13_light, содержащий разработанный по схеме модуль s13_light и модуль по описанию v13_light .

Синтез автомата.

1 этап. Работка алгоритма. Выбор способа кодирования состояний автомата, разрядности типа элементов памяти.

В автомате Мура исполняемые операторы алгоритма формируют выходные сигналы, их отмечают как состояния автоматq, и в результате поучают отмеченную граф-схему алгоритма (рис. 3.1).

q	q0	q1	q2	q3	q4
q2 q1 q0	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0

Для состояния q0 выбран желтый сигнал светофора. Сигнал «старт» обозначен через s. При s = 0 автомат должен возвращаться в состояние q0. При s = 1 выполняется циклическое переключение состояний и выдача сигналов в соответствии с заданием. Для автомата определено 5 состояний, которые обозначены как $q_0 – q_4$. Каждому состоянию соответствует определенный сигнал светофора. Заметим, для включения желтого сигнала предусмотрено два различных состояния, так как в этих ситуациях имеем различную предысторию состояний. Сигнал от кнопки k включает сигналы «красный » и «переход».

По алгоритму получено, что автомат имеет 5 состояний. При выборе двоичного кодирования разрядность памяти n = 3. При построении схем на ПЛИС в качестве элемента памяти автомата предпочтение отдают D-триггеру.

Из таблицы переходов для D-триггера следует, что на вход необходимо подать сигнал D, равный требуемому в результате перехода значению Q.

Для кодирования состояний автомата выберем единичные коды, а для соответствующих выходных сигналов триггеров – двоичные.

2 этап. Граф автомата. Вершины графа автомата Мура отображают выходные сигналы и состояния автомата, а дуги – переходы из одного состояния в другое. Над дугами указывают условия переходов (входные сигналы), при которых переход выполняется(рис. 3.2).

3 этап. Составление таблицы переходов и логических функций переходов и выходов.

Выделенная часть таблицы 3.1 (столбцы 1-4) содержат сигналы, показанные на рис. 1.1. Столбцы 5–7 содержат двоичные коды выходных и входных сигналов триггеров: qm – код вершины исходного состояния для описываемой дуги; qs – код вершины нового состояния, в которую входит дуга; d_{ms} - код, который необходимо подать на входы D триггеров, для того, чтобы произошел переход из состояния a_m в состояние a_s, описанный в данной строке таблицы. Этот столбец записывается с учетом функции и таблицы переходов для D – триггера, в соответствии с которой новое состояние, в которое переходит триггер q_s Раино входному сигналу, поданному в текущем такте d_{ms}. Поэтому столбец кодов для d_{ms} является повторением столбца кодов для q_s.

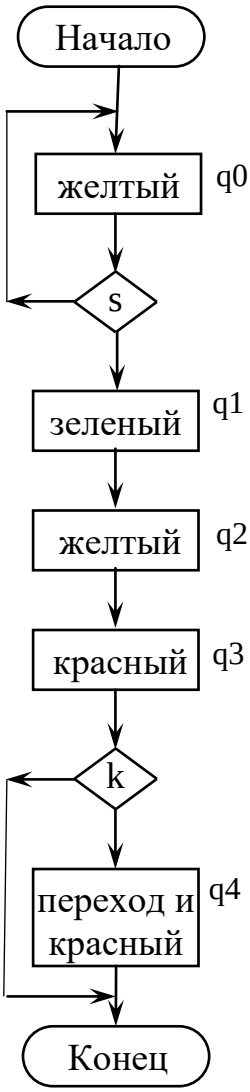


Рис. 3.1. Отмеченная ГСА

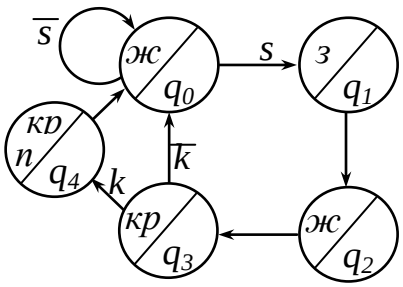


Рис. 3.2. Граф автомата управления

Первая строка таблицы описывает дугу, которая из вершины a_0 возвращается в a_0 при сигнале s = 0, поэтому в первом столбце сигнал s записан с отрицанием. Вторая строка описывает переход из a_0 в a_1 при входном сигнале s=1.

По таблице переходов составляются логические выражения для функций переходов d = δ (q_m, x), предназначенные для построения комбинационной схемы переходов (см. рис. 1.15).

Запишем выражения для сигнала d2, которому соответствуют крайние левые двоичные цифры в столбце 7, выделим строки, в которых d2=1. Это всего одна строка 6, в соответствии с которой будет сформирован сигнал d2=1, если k=1 и q_m = 011. Выражение для d2 имеет вид произведения сигналов из столбцов 1 и 2, принадлежащих данной строке : $d2 = k \cdot \overline{q2} \cdot q1 \cdot q0$.

Для записи выражения для сигнала d1 необходимо рассмотреть в столбце 7 двоичные цифры, соответствующие первому разряду, и выделить строки, в которых он равен 1. Сигнала d1 содержит две единицы в строках 3 и 4 таблицы, поэтому выражение для d1 будет равно дизъюнкции двух переменных: $d1 = q1 \vee q2$.

Сигналу d0, который равный 1 во второй и четвертой строках таблицы, соответствует дизъюнкция двух произведений: $d0 = s \cdot a0 \vee a2$.

Таблица 3.1

	x_m	q_m	q_s	y_m	q_m q2 q1 q0	q_s q2 q1 q0	d_{ms} d2 d1 d0
	1	2	3	4	5	6	7
1	$\bar{s}$	q_0	q_0	y	0 0 0	0 0 0	0 0 0
2	s	q_0	q_1	y	0 0 0	0 0 1	0 0 1
3	-	q_1	q_2	g	0 0 1	0 1 0	0 1 0
4	-	q_2	q_3	y	0 1 0	0 1 1	0 1 1
5	k	q_3	q_0	r	0 1 1	0 0 0	0 0 0
6	$\bar{k}$	q_3	q_4	r	0 1 1	1 0 0	1 0 0
7	-	q_4	q_0	p	1 0 0	0 0 0	0 0 0

По таблице переходов составляются логические выражения для функций переходов и выходов.

$$a0 = \overline{q2} \cdot \overline{q1} \cdot \overline{q0}; \quad a1 = \overline{q2} \cdot \overline{q1} \cdot q0; \quad a2 = \overline{q2} \cdot q1 \cdot \overline{q0};$$
$$a3 = \overline{q2} \cdot q1 \cdot q0; \quad a4 = q2 \cdot \overline{q1} \cdot \overline{q0}.$$

Выходные сигналы, зависят от состояний автомата. Если выходной сиграл формируется несколькими состояниями, то записывается дизъюнкция состояний. **Составление логических функций для дешифратора кодов состояний.**

С учетом выбранных кодов можно записать логические функции для дешифратора кодов состояний (см. рис. 1.1).

4 этап. Разработка схемы конечного автомата.

Схема конечного автомата управления светофором (Рис. 1.4), содержит 3 триггера, образующие память, комбинационные схемы формирования сигналов переходов и комбинационные схемы формирования выходных сигналов (g, r, p, y).

5 этап. Описание конечного автомата на языке Verilog.

Память конечных автоматов, построенную на синхронных триггерах с динамическим управлением, которые срабатывают по фронту импульсов синхронизации, описывают последовательные операторы с ключевым словом always, для которых указывают срабатывание по фронту синхросигнала. Последовательные операторы могут содержать операторные скобки begin – end и операторы case и if.

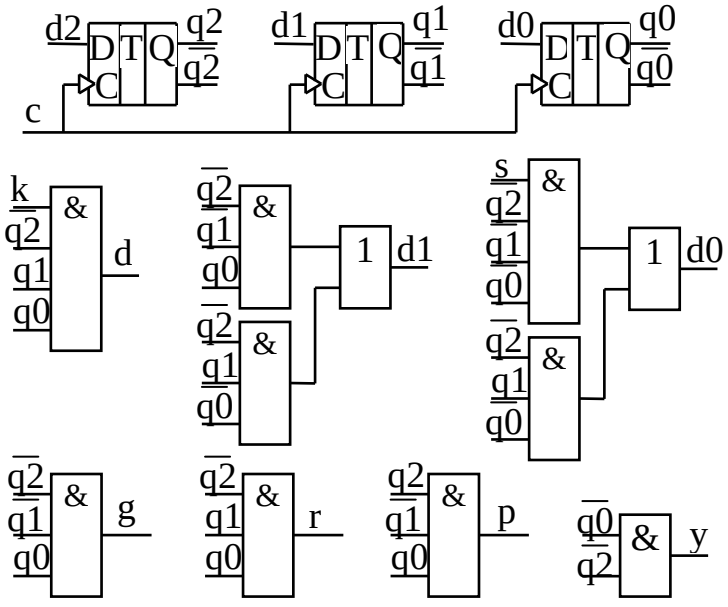


Рис. 3.3. Схема автомата управления

Простое для восприятия описание автомата получается, если для переходов автомата из одного состояния в другое используется оператор варианта case, в котором в качестве селектора используется исходное состояние автомата, а в качестве выполняемого действия – присваивание нового состояния. Селектор оператора варианта – это переменная, записываемая в скобках после ключевого слова case. Присваивание

выполняется, если селектор равен 1. Если переход автомата из одного состояния в другое зависит от входного сигнала, то используется оператор if, содержащий в качестве условия входной сигнал.

```
// Описание  автомата светофора
module v31_light (c,s,k,q,r,y,g,p);//2
input c,s,k;                        //3
output [2:0]q;                      //4
reg [2:0]q;                         //5
output r,y,g,p;                     //6
always @ (posedge c)                //7
case (q)                            //8
3'b000: if(s) q=3'b001;
            else  q=3'b000;     ///9
3'b001: q=3'b010;                   //10
3'b010: q=3'b011;                   //11
3'b011: if(k)  q=3'b100;
                    else  q=3'b000; //12
default: q=3'b000; endcase          //13
assign r =  ~q[2] & q[1] & q[0];//14
assign y = ~q[2] & ~q[0];           //15
assign g = ~q[2] & ~q[1] & q[0]; //16
assign p =  q[2] & ~q[1] & ~q[0];//17
endmodule                           //18
```

Описание комбинационной схемы выходов, которая не содержит элементов памяти, выполняют параллельными операторами с ключевым словом «assign».

В приведенном описании автомата управления светофором, в строке 2 указано имя модуля и перечислены все входные и выходные сигналы, в строке 3 определены входные сигналы, по умолчанию они будут одноразрядными типа «wire». В строках 4,5 указан выходной сигнал q в виде 3-разрядного вектора (или шины) типа «reg». Этот тип сигнала необходим, чтобы использовать последовательные операторы в строках 8 - 13.

Выходные сигналы r, y, g, p указаны в строке 6 как одноразрядные типа «wire». Для формирования этих сигналов использованы параллельные операторы (строки 14 - 17). В строке 13 исходное значение селектора указано как «default». При выполнении этой строки будет выбран любой вариант, отсутствующий среди перечисленных ранее вариантов. Это не только состояние 100, в котором может находиться автомат, но и все возможные другие состояния, возникающие при включении схемы, или в результате сбоя. Таким образом повышается помехоустойчивость.

Коды определяющие исходные и новые состояния автомата записаны как константы в двоичной системе счисления. Первый элемент записи константы – десятичное число, равное количеству разрядов; второй элемент – апостроф; третий элемент – буква, определяющая систему счисления (b- двоичная); четвертый элемент – число в указанной системе счисления.

6 этап. Тестирование автомата.

Таблица 3.2

s	k	состояния	выходы
0	0	0	y
0	1	0	y
1	0	0 – 1 – 2 – 3 - 0 – 1..	y-g-y-r-y-
1	1	0 – 1 – 2 – 3 – 4 - 0..	y-g-y-r-p-y-

Для тестирования конечного автомата необходимо проверить правильность изменения состояний автомата при всех возможных комбинациях входных сигналов. По графу необходимо составить таблицу 3.2 - изменений состояний и выходов автомата и выделить циклы повторяющихся состояний. При отладке и тестировании необходимо проверить выполнение данных циклов смены состояний.

Задание 3.2. Разработка модифицированного проекта автомата управления светофором.

Разработайте альтернативный вариант рассмотренного проекта при различных вариантах типа используемых триггеров и различных способах кодирования состояний в соответствии с заданным вариантом. При составлении таблицы переходов автомата необходимо учитывать свойства и таблицу переходов заданного триггера.

Вариант	1	2	3	4	5	6	7	8	9	10
Код состояний	двоич-ный	Грея	Единич-ный	Двоич-ный	Грея	Единич-ный	Грея	Единич-ный	Двоич-ный	Грея
Триггер	JK	T	D	T	JK	T	D	JK	JK	T

Контрольные вопросы

1. Чем различаются абстрактный и конечный автоматы?.
2. Перечислите назначение параметров, которые задают конечный автомат.
3. Изобразите структурную схему конечного автомата, поясните назначение всех элементов.
4. Как составляется граф автомата Мура по ГСА.
5. Как составляется таблица переходов по графу?
6. Как составляются логические функции переходов по таблице переходов автомата?
7. Как составляются логические функции для дешифратора состояний?
8. Изобразите таблицу переходов для триггеров jk, RS, D, T.

Работа 4. Синтез множительного устройства с микропрограммным управлением

Теоретическая часть

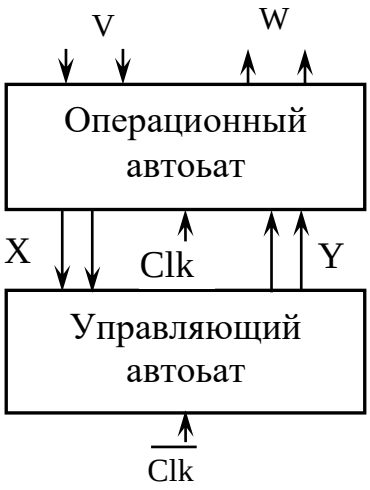


Рис. 4.1. Устройство обрпаботки данных

По методике академика В. М. Глушкова цифровое устройство, предназначенное для выполнения разлмчных вычислительных операций (преобразования кодов, ввода и вывода данных, управления процессами и объектами) представляют как систему, состоящую из двух автоматов (цифровых устройств, содержащих элементы памяти) – операционного автомата - ОА и управляющего автомата - УА (рис. 3.1). Такое представление упрощает синтез.

Операционный автомат выполняет заданную обработку входных данных V и формирует выходные данные W и признаки X под управлением сигналов Y.

Комбинационные схемы ОА (мультиплексоры, дешифраторы, сумматоры, компараторы, АЛУ) выполнят обработку данных и их пересылку в память. Они являются асинхронными, выходной сигнал появляется сразу же после подачи входа сигнала, с определенной задержкой.

Элементы памяти ОА (триггеры, регистры, счетчики запоминающие устройства), предназначены для хранения текущих значений обрабатываемых данных, управляющих сигналов, параметров циклов. В проектах на ПЛИС используют библиотечные модули элементов памяти динамического типа или их описания на Verilog.

Управляющий автомат – это конечный автомат, который обеспечивает выполнение микрокоманд в ОА в соответствии с заданным алгоритмом. На вход УА из ОА поступают признаки результата X, определяющие порядок смены состояний УА и порядок выполнения операций. УА формирует управляющие сигналы Y, под действием которых в модулях ОА выполняются микрокоманды в соответствии с требуемым алгоритмом. С учетом функционального назначения управляющий автомат называют микропрограммным.

Существует два подхода к проектированию управляющего автомата. Первый подход - это использование принципа схемной (жесткой) логики, при котором

последовательность микрокоманд формирует конечный (микропрограммный) автомат. Такие устройства способны обеспечивать наивысшее быстродействие при заданном типе технологии элементов.

Второй подход - это использование принципа программируемой логики, при котором ОА работает под управлением микрокоманд, которые содержатся в памяти в виде программы. В этом случае обеспечивается возможность модификации выполняемых функций изменением содержимого памяти микрокоманд без изменения схемы.

В сложных устройствах, построенных на основе синхронных триггеров с динамическим управлением (которые записывают данные по фронту синхросигнала) используется система синхронизации, обеспечивающая запись только в те моменты времени, когда изменения этих данных закончились.

В цифровых устройствах, содержащих два взаимодействующих автомата (рис. 4.1), целесообразно использовать двухфазную синхронизацию. При этом для записи в устройства памяти ОА выбирается, например, фронт сигнала clk (рис. 3.2), а для записи в устройства памяти УА –спад сигнала clk (который совпадает с фронтом инверсного сигнала). Все остальные операции выполняются комбинационными схемами асинхронно в интервалы, соответствующие состояниям сигнала clk = 0 и clk = 1. Работа устройства обработки будет выполняться следующим образом.

1. Спад сигнала clk (фронт инверсного сигнала). Переход УА в новое состояние. Выдача соответствующих сигналов Y в ОА. Заметим, на УА подается инверсный сигнал clk (см. рис. 3.1)

Рис. 4.2. Импульсы синхронизации

2. Интервал clk = 0. Модули ОА (комбинационного типа) формируют выходные сигналы в соответствии с управляющими сигналами X и входными данными V (с определенной задержкой).

3. Фронт сигнала clk. Запись результата микрооперации текущего такта в устройства памяти ОА.

4. Интервал clk=1. Формирование в УУ функции перехода к новому состоянию.

Важное замечание. Для надежной синхронизации необходимо в качестве УА использовать автомат Мура, в котором смена состояний выполняется синхронно с изменениями синхросигнала.

Синтез устройства обработки, содержащего два взаимодействующих автомата - ОА и УА рассмотрим на конкретном примере.

Задание 4.1. Разработайте устройство с микропрограммным управлением с жесткой логикой для умножения двух целых положительных 4-разрядных двоичных чисел: P = A * B с младших разрядов множителя и сдвигом множимого влево с использованием операций суммирования и сдвига.

Отчет должен содержать следующие пункты.

1. Синтез ОА.
   - 1.1. Анализ постановки задачи, разработка регистровой модели ОА.
   - 1.2. Разработка алгоритма, составление содержательной ГСА.
   - 1.3. Разработка модулей ОА.
2. Синтез УА, обеспечивающего управление работой ОА.
   - 2.1. Разработка отмеченной ГСА по с содержательной ГСА.
   - 2.2. Разработка графа УА.
   - 2.3. Составление таблицы переходов и выходов.
   - 2.4. Составление логических функций переходов и выходов.

2.5. Разработка схемы, тестирование, создание символа модуля УА.

3. Разработка функциональной схемы устройства, содержащей все модули обработки сигналов и управления. Разработка тестовых сигналов. Моделирование. Анализ результатов.

Пояснение проектирования заданного устройства.

1.1.Анализ постановки задачи, разработка регистровой модели ОА.

Регистровая модель отображает все элементы памяти (регистры, счетчики и блоки памяти), необходимые для хранения данных пр решении поставленной задачи (рис. 4.3). Выбор элементов производится с учетом технического задания и уточняется при составлении алгоритма. Связи между элементами и комбинационные схемы, выполняющие логические операции, в регистровой модели не изображаются, они учитываются впоследствии при разработке функциональной схемы.

Для реализации устройства умножения заданным методом (который используется при умножении столбиком) необходимо иметь для множителя регистр reg_b заданной разрядности n=4. Для сумматора sum, регистра произведения reg_pr и множимого reg_a требуется удвоенная разрядность 2 n = 8 (с учетом необходимости сдвига множимого при вычислениях).

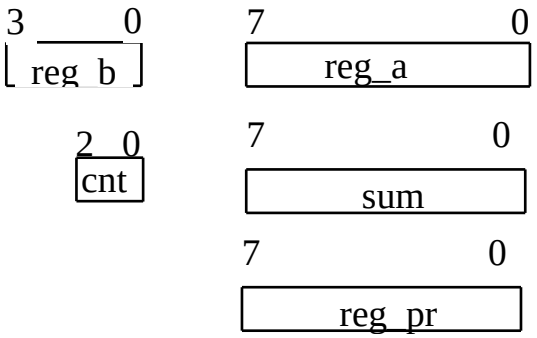


Рис. 4.3. Регистровая модель ОА

отображает все элементы памяти, необходимые для решения задачи. Это ra – регистр множимого; rb – регистр множителя; sum – ссумматор, в котором будут накапливаться частичные суммы; r_pr –регистр произведения; cnt - счетчик циклов. Разрядность регистров определена с учетом сдвига множимого, разрядности суммы и произведения.

1.2. Разработка алгоритма. Алгоритм выполняемый в ОА задачи представляется в виде содержательной ГСА (рис. 4.4), содержащей операторные вершины Y, соответствующие выполняемым микрокомандам, и условные вершины X, содержащие условия переходов.

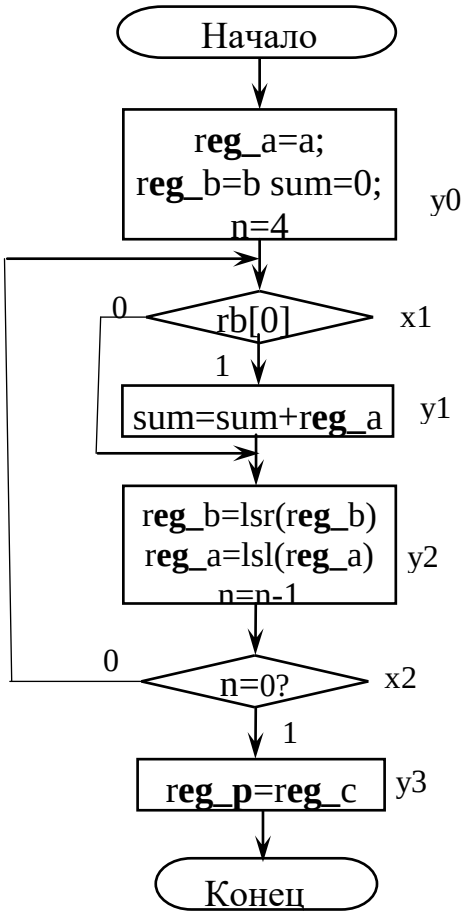


Рис.4.4. Содержательная ГСА

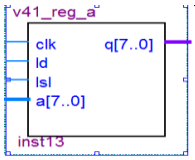
Каждая микрокоманда выполняется под управлением сигнала Y_i за один такт синхросигнала clk, длительность которого выбирается по самой продолжительной микрокоманде. Операторные вершины (микрокоманды) могут содержать несколько совместимых микроопераций, которые будут выполняться одновременно в одном и том же такте, под управлением одного и того же сигнала.

Микрокоманда y0 содержит микрооперации инициализации. Это ввод исходных данных в регистры множимого и множителя a и b; сброс аккумулятора sum, запись в счетчик количества повторения циклов.

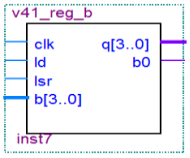
Условная вершина x1 проверка значения младшего вита множителя . Если этот бит rb[0] = 1, то выполняется команда y1 - прибавление к содержимому сумматора sum частичной суммы из регистра множимого, иначе суммирования нет.

Микрокоманда y2 содержит микрооперации

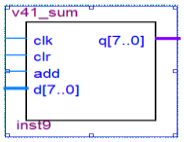
подготовки данных для прибавления следующей частичной суммы. Это логические сдвиги множителя – вправо, множимого – влево, декремент счетчика циклов.




```
module v41_reg_a
(clk, ld, lsl, a, q);
input clk, ld, lsl;
input [7:0]a;
output [7:0] q;
reg [7:0] q;
always @ (posedge clk)
if (~lsl & ld) q=a;
   else if (lsl & ~ld)
    q={q[6:0],1'b0};
endmodule
```




```
module v41_reg_b
(clk, ld, lsr, b,q, b0);
input clk,ld, lsr;
input [3:0] b;
output b0;
output [3:0] q;
reg  [3:0] q;
always @(posedge clk)
if (ld & ~lsr) q=b;
   else if (~ld & lsr)
   q={1'b0,q[3:1]};
assign b0=q[0];
endmodule
```




```
module v41_sum
(clk, clr, add, d, s);
input clk, clr, add;
input [7:0]d;
output [7:0] s;
reg [7:0] s;
always @(posedge clk)
if (~clr & add) s=s+d;
  else if (clr & ~add)
s=0;
endmodule
```

Условная вершина x2 – проверка условия выхода из цикла;

Команда y3 - запись содержимого накапливающего сумматора в буферный регистр произведения.

1.3. Разработка модулей ОА

Для всех модулей, содержащихся в регистровой модели ОА (рис. 4.3), необходимо разработать устройства, выполняющие функции, предусмотренные в алгоритме (рис. 4.4). Это могут быть готовые элементы из библиотеки САПР, или проекты, разработанные по описаниям на Verilog.

Модули ОА, содержащие элементы памяти с динамическим управлением, описываются последовательными операторами, содержащими условные операторы, определяющие алгоритм функционирования.

Регистр множимого reg_a должен выполнять функции загрузки множимого, сдвига и выдачи частичной суммы. Множимое (a) загружается в младшие 4 разряда **reg_a**. Для получения частичных сумм содержимое регистра впоследствии сдвигается 4 раза, поэтому регистр имеет 8 разрядов.

Входные управляющие сигналы: ld (load)загрузка 4-разрядного множимого a при ld =1; lsl – логический сдвиг числа влево на один разряд при входе lsl = 1;

В схеме ОА в соответствии с ГСА (рис. 4.4): на вход ld подается y0, а на вход lsl - сигнал y2.

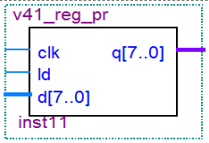
Пояснение. Строка описания q={q[6:0],1'b0}; выполняет логический сдвиг влево. Она содержит оператор присваивания переменной q нового значения в виде объединения записанного в фигурных скобках: разрядов 6-0 старого значения и константы 0. Константа записана в полном формате, который содержит элементы : 1) цифра – количество разрядов, 2) – апостроф, 3) – буква «b» - двоичная система счисления, 4) – число в указанной системе. Попробуйте вместо «1'b0» записать «0». Компилятор выдаст ошибку. Константу «0» он примет как десятичную и отведет для нее 4 двоичных разряда, что в данном случае недопустимо.

Регистр множителя reg_b выполняемые функции: загрузка, сдвиг, выдача на выход младшего разряда. Сигналы: ld (load)загрузка множителя b (при ld =1), lsr – логический сдвиг числа вправо на один разряд (при lsr = 1). ; b0 –младший бит - – выходной сигнал x1, q[3..0] – выход для отладки.

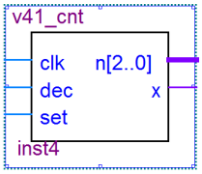
В схеме ОА на вход ld подается y0, на lsr – y2,

Сумматор sum. Выполняемые функции - сброс, загрузка и суммирование с накоплением. Сигналы: d[7..0] -входные данные 4-разрядное множимое в пределах 8-разрядного слова; ld (load) - разрешения загрузки при ld =1; add – разрешение суммирования с накоплением по формуле q = q + d; q[7..0] – выходной код.

Подключение в схеме ОА: d[7..0] - выход reg_a, q – вод reg_pr; clr- y0, add – y1.




```
module v41_reg_pr
(clk,ld, d,q);
input clk, ld;
input [7:0] d;
output  [7:0] q;reg [7:0] q;
always  @ (posedge clk)
if(ld) q=d;
endmodule
```




```
module v41_cnt
(clk, dec, set, n, x);
input clk, dec, set;
output [2:0] n;
reg [2:0] n;
output x;
always @(posedge clk)
if (~set & dec) n=n-1;
   else if (set) n=4;
assign x=~n[1]&~n[0];
endmodule
```

Регистр произведения reg_pr. Это буферный регистр для загрузки и хранения данных. Сигналы: d[7..0] - входные данные с выхода сумматора; ld - разрешение загрузки при ld =1 (сигналом y4); q[7..0] – выходной код;

Счетчик циклов cnt. Выполняемые фикции – установка исходного кода – загрузка константы 4 (при set = 1), декремент (при dec = 1), выдача признака x=1 при n=0.

Счетчик 3-разрядный, так как в данной задаче требуется модуль 5.

Подключение. set – y0, dec – y3, x – x2.

Для каждого из приведенных модулей необходимо создать проект (в одном общем каталоге), выполнить компиляцию, создать символ. Впоследствии эти символы позволят составить схему устройства.

2. Синтез управляющего автомата

Выполняется по методике, описанной в работе 1..

2.1. Построение отмеченной ГСА УА по содержательной ГСА ОА. Вместо микроопераций в операторных вершинах записываются микрокоманды (Y), а вместо логических условий – их обозначения (X). За состояния автомата Мура принимаются исполняемые операторы (рис. 4.5). Вершины «начало» и «конец» считаются соединенными, алгоритм является циклическим.

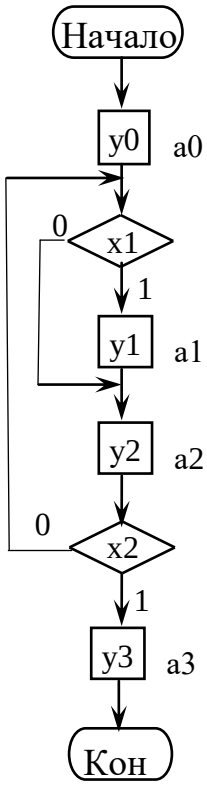


Рис.4.5. Содержательная ГСА

.2.2. Построение графа автомата по отмеченной ГСА. Вершины графа соответствуют состояниям и выходным сигналам автомата, дуги – переходам из состояния a_m в состояние a_s. Над дугой записываются логические условия, определяющие переход из состояния a_m в состояние a_s . Для безусловных переходов записывается константа 1.

Кодирование состояний автомата Мура Используем для данного примера минимальное кодирование состояний. Так как автомат имеет четыре состояния ($N_A = 4$), то минимальное количество элементов памяти n равно:

$$n =] \log_2 |N_A[=] \log_2 4 [= 2.$$

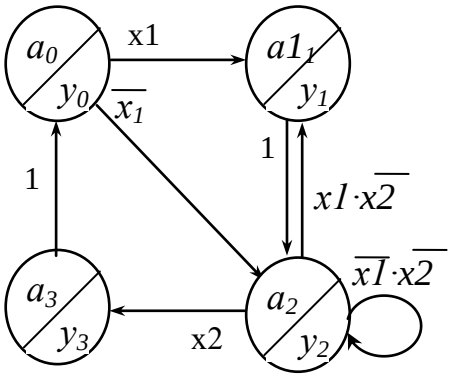


Рис. 4.6. Граф автомата

Выходы триггеров обозначаются соответственно q_1q_0. Значение числа q_1q_0 на этих выходах – это двоичный код состояния автомата.

Для выходных сигналов H и состояний A используются единичные коды, в данном случае они равны Y = A.

$y_0 = 0001_2$, $y_1 = 0010_2$, $y_2 = 0100_2$, $y_3 = 1000_2$.

2.3 Структурная таблица переходов автомата Мура строится по графу

Количество строк в таблице должно быть равно количеству переходов в графе,

а дизъюнкция всех X_m из состояния a_m должна быть равна 1.

Таблица 4.1

№	x_m	a_m	a_s	y_m	q_m	q_s	$d1d_0$
1	x_1	a_0	a_1	y_0	00	01	01
2	$\sim x_1$	a_0	a_2	y_0	00	10	10
3	1	a_1	a_2	y_1	01	10	10
4	$x_1 \cdot \sim x_2$	a_2	a_1	y_2	10	01	01
4	$\sim x_1 \cdot \sim x_2$	a_2	a_2	y_2	10	10	10
5	x_2	a_2	a_3	y_2	10	11	11
6	1	a_3	a_0	y_3	11	00	00

По структурной таблице автомата **составляются логические функции переходов** в соответствии с функцией: $d_{ms} = \delta(\mathbf{x_m}, \mathbf{a_m})$ (см. рис. 1.1). Входными сигналами являются текущие значения входного сигнала автомата и текущие состояния, а выходными - управляющие сигналы триггеров, обеспечивающие требуемый переход. Для записи логической функции перехода вначале выделяется столбец для одного из разрядов управляющего сигнала триггера (например, d0 в столбце сигналов - d_{ms}), в котором определяются строки, содержащие 1. Для каждой строки записывается произведение сигналов, содержащихся в столбцах 1 и 2 (это x_m и a_m). Произведения объединяются операцией дизъюнкции. Данные операции повторяются для всех разрядов сигнала D.

$$d0 = x1 \cdot a0 \vee x1 \cdot \overline{x2} \cdot a2 \vee x2 \cdot a2; \quad d1 = \overline{x1} \cdot a0 \vee a1 \vee \overline{x1} \cdot \overline{x2} \cdot a2 \vee x2 \cdot a2$$

Дешифратор кодов состояний преобразует двоичные коды, получаемые с выходов триггеров (Q) в единичные коды, определяющие состояниям (A).

$$a_3 = q_1 \cdot q_0; \quad a_2 = q_1 \cdot \overline{q_0}; \quad a_1 = \overline{q_1} \cdot q_0; \quad a_0 = \overline{q_1} \cdot \overline{q_0}.$$

Комбинационная схема выходов формирует выходные сигналы Y. **В автомате Мура** выходные сигналы зависят только от текущего состояния автомата. В данном случае $y_m = a_m$.

2.4. Схема автомата (Рис.4.7**)** составляется с ис пользованием полученных логических функций

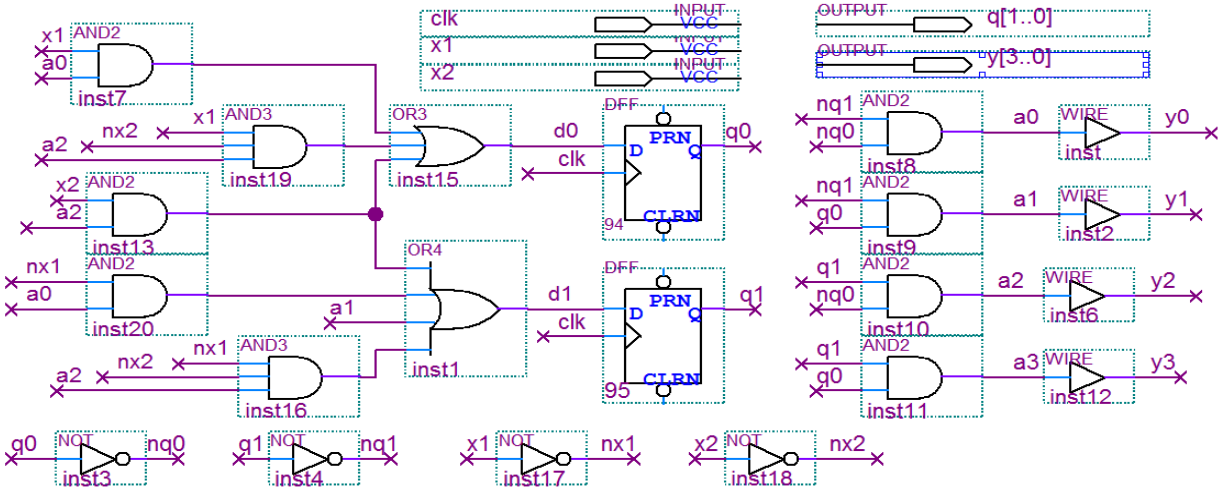


Рис.4.7. Схема управляющего автомата s_41_ua_moore

2.5. Тестирование автомата. При моделировании оставляются тестовые сигналы, позволяющие определить возможные неисправности. По графу составляется таблица тестовых сигналов, которая содержит последовательности состояний при заданных входных сигналах. При моделировании используются статические состояния

Таблица тестовых сигналов

x2	x1	Выходы триггеров q	Состояния Мура
0	0	0-2-2-2-2-2	0001-0100
0	1	0-1-2-1-2-1-2-	0001-0010-0100-
1	0	0-2-3-0-2-3-	0001-0100-1000-
1	1	0-1-2-3-0-1-2-3-	0001-0010-0100-1000-

входных сигналов (различные комбинации 0 и 1) и анализируется состояния выходных сигналов памяти. Таблица тестовых сигналов позволяет анализировать результат моделирования (рис. 4. 8).

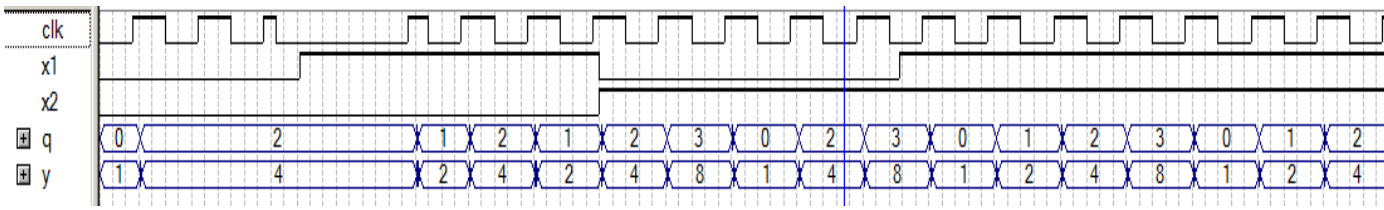


Рис. 4. 8. Результат моделирования управляющего автомата s_41_ua_moore

3. Разработка функциональной схемы устройства.

Функциональная схема устройства разрабатывается на основе регистровой модели ОА и отмеченной ГСА, содержит разработанные ранее модуль Уа и модули ОА, а также связи между модулями и управляющие сигналы (микрокоманды y0, y1, y2, y3).

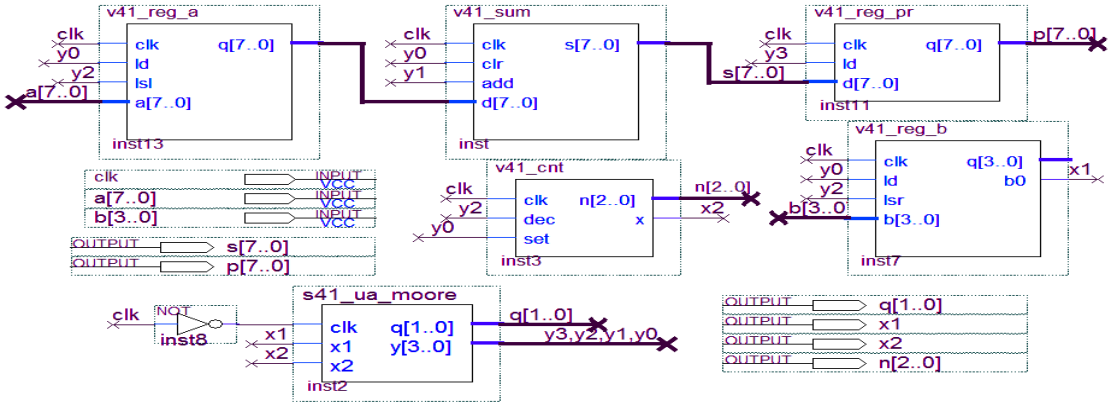


Рис. 4.9. Схема устройства s41_device

Результаты моделирования при различных входных сигналах (рис. 4.10 и 4.11) соответствуют теоретическим значениям.

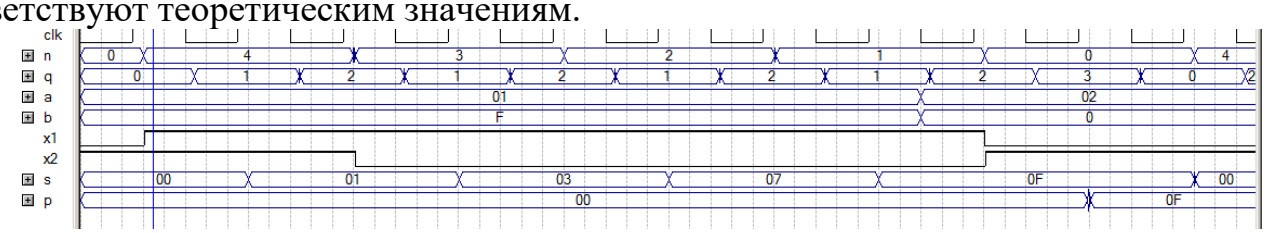
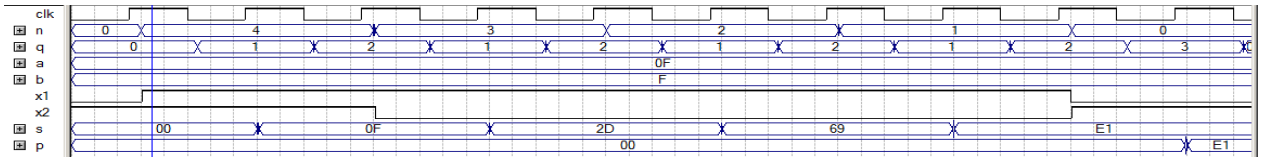


Рис. 4. 10. Результат моделирования устройства s_41_ device, a=1, b=f.



ис. 4. 11. Результат моделирования устройства s_41_ device, a=f, b=f.

Задание 4.2. Разработайте проект УА по описанию на Verilog с именем v42_moore.

Задание 4.3. Разработайте иерархический проект УА по схеме с именем sv43_moore, содержащий модули s41_moore и v42_moore . Выполните моделирование.

Контрольные вопросы

1. Какие элементы содержит ОА?
2. Какие элементы содержит УА?
3. Как взаимодействуют ОА и УА в системе?
4. Как составляются содержательная ГСА и отмеченная ГСА?
5. Как формируются тестовые сигналы?

Работа 5. Синтез модулей интерфейса SPI

Последовательный периферийный интерфейс SPI (Serial Peripheral Interface) выполняет дуплексный обмен данными между ведущим (Master) и ведомым (Slave) модулями. Основу интерфейса SPI составляют два сдвигающих регистра SRM (Shift Register Master) и SRS (Shift Register Slave), соединенные в замкнутый контур посредством линий MOSI (Master Out Slave In) и MISO (Master In Slave Out) (рис. 5.1). В результате соединения n-разрядных сдвигающих регистров образуется кольцевой сдвигающий регистр удвоенной разрядности.

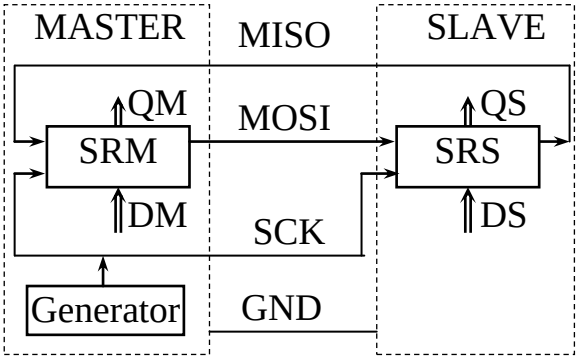


Рис. 5.1. Структура интерфейса SPI

Исходные данные, предназначенные для обмена (DM – Data Master и DS - Data Slave), должны быть заблаговременно записаны в регистры SRM иSRS. Обмен данными запускает модуль Master, который подает n синхроимпульсов частоты SCK (Serial Clock) на оба сдвигающих регистра. В результате содержимое регистра SRM, бит за битом, передается в регистр SRS, а содержимое SRS в SRM. В регистрах будут получены параллельные коды QM = DS и QS = DM. Заметим, результат не зависит от направления сдвига.

Синхронный последовательный интерфейс позволяет простыми техническими средствами осуществить дуплексный обмен данными между различными устройствами (ПЛИС и МК). Современные МК имеют программируемые интерфейсные модули SPI.

При проектировании интерфейса SPI для ПЛИС модули Master и Slave разрабатываются как отдельные проекты, для которых указываются одинаковые основные параметры: разрядность передаваемых данных, порядок передачи разрядов данных (например, от старшего разряда к младшему), скорость передачи и способ синхронизации сигналов. Каждый из проектируемых модулей представляют как систему, состоящую из двух автоматов –операционного автомата - ОА и управляющего автомата – УА, как показано в предыдущей работе на рис. 4.1. Такое представление упрощает синтез.

Проектирование модуля MASTER интерфейса SPI

Разработка регистровой модели, алгоритма и модулей ОА.

Регистровая модель отображает элементы памяти, необходимые для выполнения задачи (Рис.5.2) и является основой для разработки алгоритма (Рис.5.3).

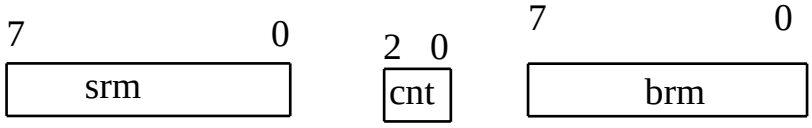


Рис. 5.2. Регистровая модель ОА

Основным элементом ОА является сдвигающий регистр srm (Shift Register Master), выполняющий преобразование параллельного кода в последовательный посредством выполнения n сдвигов. Для подсчета количества сдвигов необъходим счетчик cnt, а для записи результата – буферный регистр brm (Buffer Rtgister Master).

Алгоритм сотавлен на основе анализа выполнения заданной работы. Вначале необходима инициализация, она должна быть выполнена заблаговременно до начала преобразования, поэтому выполняется безусловно после вершины «начало».

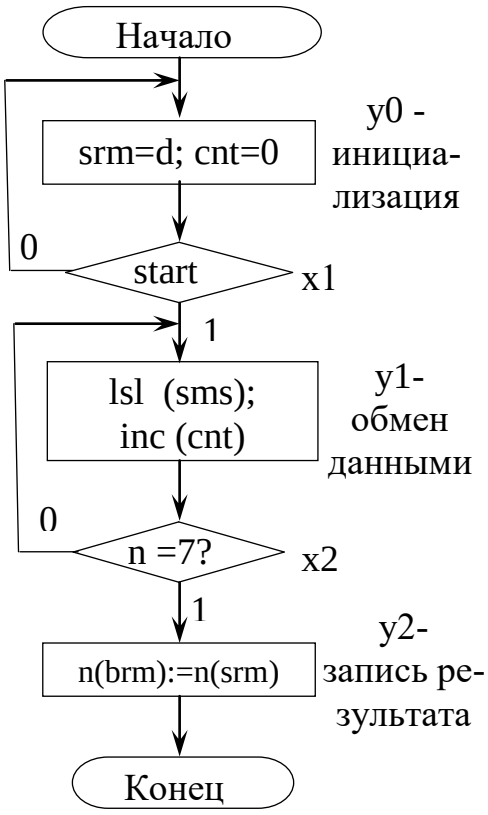


Рис. 5.3. Алгоритм работы мтодуля Master

Модуль Master – ведущий, он определяет начало обмена данными импульсои «старт».

Для обмена данными циклически выполняются логические сдвиги влево, при этом старший бит кода регистра выдается в линию связи mosi, а освобождающийся бит заполняется сигналом miso. Количество сдвигов подсчитывает счетчик, код которого определяет условие выхода из цикла.

После выполнения n = 8 сдвигов в регистр srm поступит код, принятый из модуля slave, который необходимо сохранить в буферном регистре brm.

Разработка элементов и модулей ОА

Определим выполняемые функции, входные и выходные сигналы, составим описания элементов на Verilog.

Сдвигающий регистр srm должен иметь входы и выходы для параллельного кода (d , q) и для последовательного кода (sin, sout) и выполнять в соответствии с алгоритмом следующие функции.

1. Загрузка параллельного кода, поданного на вход d под управлением сигнала ld = 1.

2. Выдача старшего разряда кода q как сигнала sout, сдвиг кода регистра q влево с заполнением

```
module v51_srm
(clk, lsl, miso, d,q, mosi);
input clk,lsl, miso;
input [7:0] d;
output mosi;
output [7:0] q;
reg  [7:0] q;
assign mosi=q[7]&lsl;
always @(posedge clk)
if (lsl)q={q[6:0],miso} ;
 else q=d;
endmodule
```

```
module v51_cnt
(clk, inc, n, over,sck);
input clk, inc;
output [2:0] n; reg [2:0] n;
output over,sck;
always @(posedge clk)
if (inc)  n=n+1;
   else n=0;
assign over=n==7;
assign sck=clk &(n<=7)&inc;
endmodule
```

```
module v51_srs
(sck,  mosi, d,q, miso);
input sck, mosi;
input [7:0] d;
output miso;
output [7:0] q;
reg  [7:0] q;
assign miso=q[7];
always @(posedge sck)
  q={q[6:0],mosi};
endmodule
```

освобождающегося младшего разряда сигналом sin под управлением сигнала lsl = 1.

Счетчик cnt должен выполнять подсчет импульсов clk, формирование тактовых импульсов sck и сигнала окончания цикла преобразования 0vtr при n = 7.

Для построения иерархического проекта SPI потребуется модуль Slave, который в простейшем случае содержит сдвигающий регистр srs.

Приведенным описаниям соответствуют символы, использованные в схеме (рис.5.6).

Задание 5.1. По приведенным описаниям Для вчех модклуй создайте проекты, выполните компиляцию, создайте символы. Моделирование можно не проводить, оно будет выполнено впоследствии в составе иерархического проекта.

Разработка управляющего автомата

Выполняется в соответствии с алгоритмом (рис. 5.3) в виде автомата

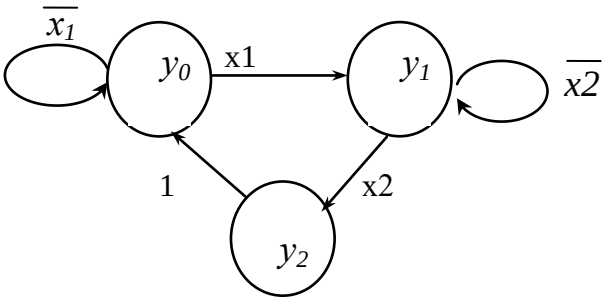


Рис. 5.4. Граф автомата

Мура с единичным кодированием состояний. За состояния приняты исполняемые операторы алгоритма, вершины «начало» и «конец» считаются соединенными, они не формируют никакие выходные сигналы. Состояния автомата определяют формируемые выходные сигналы Y, для которых выбрано единичное кодирование. По алгоритму составлен граф и таблица переходов, что позволило запимать функции переходов и составить схему автомата (рис.5.5)

$$d0 = \overline{x1} \cdot y0 \vee y2;$$
$$d1 = x1 \cdot y0 \vee \overline{x2} \cdot y1;$$
$$d2 = x2 \cdot y1.$$

Таблица 4.1

№	x_m	y_m	y_s	q_{ms}	q_s	d_{ms}
1	~x_1	y_0	y_0	001	001	001
2	x_1	y_0	y_1	001	010	010
3	~x_2	y_1	y_1	010	010	010
4	x_2	y_1	y_2	010	100	100
5	1	y_2	y_0	100	001	001

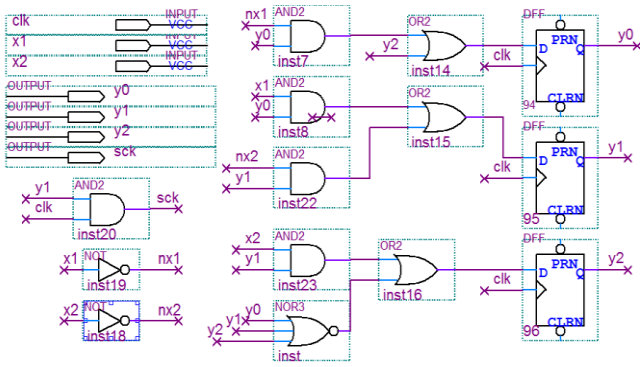


Рис. 5.5. Схема УА s52_ua

Задание 5.2. разработайте проект УА по схеме (рис. 55), выполните 9 этапов проектирования по методике, рассмотренной в предыдущих работах. Это ввод схемы, создание проекта с именем s51_ua, компиляция, создание файла временных диаграмм, разработка тестовых сигналов, моделирование, анализ результатов. В отчете опишите полученные временные диаграммы.

Задание 5.3. Разработайте иерархический проект интерфейса SPI, содержащий на одной схеме (рис. 5.6) модуль Master (ОА и УА) и простейший модуль Slfve (один сдвигающий регистр). Выполните моделирование. Подробно опишите разработку тестовых входных сигналов и результаты моделирования

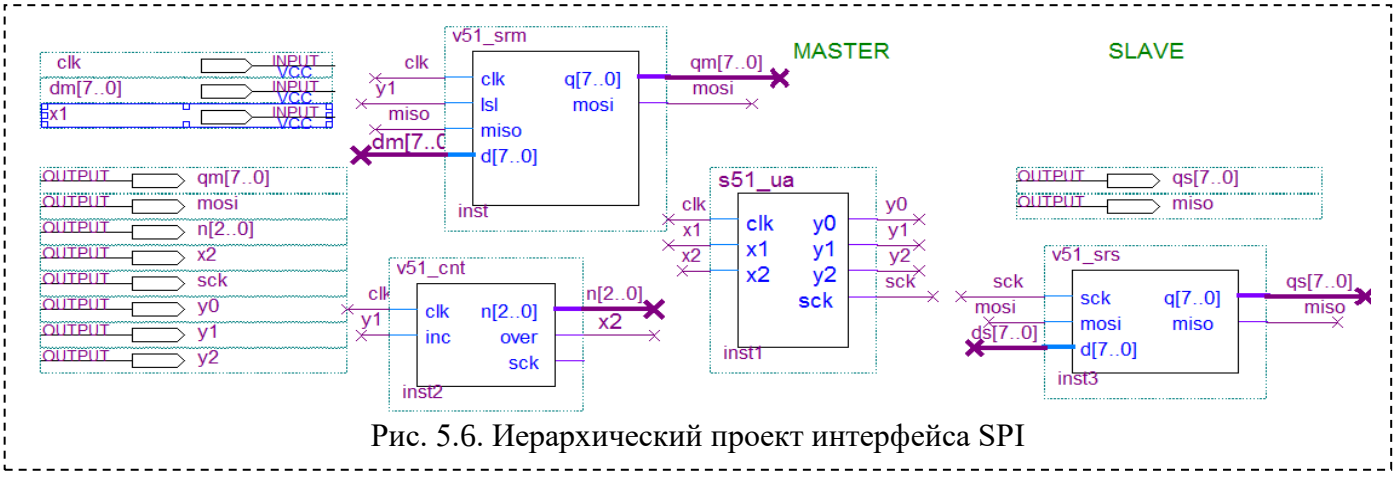


Рис. 5.6. Иерархический проект интерфейса SPI

При моделировании необходимо задать тестовые входные сигналы x1, dm clk.

Сигнал x1 рпобходимо нарисовать выделяя интервалы и устанавливая значения 0 или 1. Сигнал dm устанавливается кнопкой с буквой «С», система счисления – Hexadecimal, начальное значение - 83, множитель – 10, изьенения кода – 50нс.

Сигнал clk устанавливается кнопкой с изображением часов, период 50 нс.

Временные диаграммы упорядочены для удобства восприятия. В верхней части диаграммы сигнала x1 и кодов, затем clk как разделитель, затем сигналы для проверки работы автомата.

В нижней части – сигналы, отображающие последовательные коды.

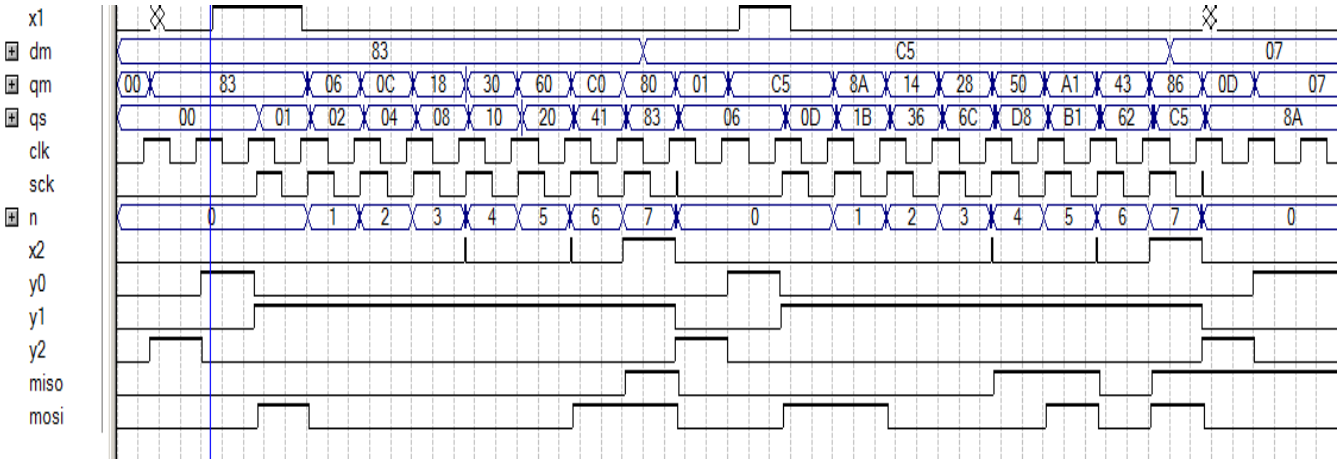


Рис. 5.7. Результат моделированя проекта интерфейса SPI

Задание 5.4. Разработайте модифицированный иерархический проект интерфейса SPI, в котором в предыдущую схему добавлен буферный регистр, в который записывается код регистра srm по сигналу УА y2..

Контрольные вопросы

1. Какие элементы содержит ОА модуля Master SPI?
2. Какие элементы содержит УА модуля Master SPI?
3. Как взаимодействуют ОА и УА в системе модуля Master SPI.
4. Изобразите структурную схему УА модуля Master SPI?.
5. Как составляются содержательная ГСА и отмеченная ГСА?
6. Как формируются тестовые сигналы?
7. Выполните описание управляющего автомата на языке Verilog.

Работа 6. Разработка процессорного ядра аккумуляторного типа

Разработайте проект вычислительного устройства (ядра) аккумуляторного типа для выполнения заданных 4-х операций. Необходимо:

1) разработать структурную схему;
2) составить описание и алгоритм работы,
3) выбрать сигналы синхронизации;
4) описание модулей на языке Verilog;
5) разработать теоретические временные диаграммы;
6) разработать проект в САПР и выполнить моделирование,
7) выполнить анализ результатов моделирования.

Заданы варианты для параметров.

1. Разрядность обрабатываемых данных – 1 вар. - 4; 2вар. – 8; 3вар – 16.
2. Кодирование и титы выполняемых операций

Код\Вариант	1	2	3	4	5	6	7	8	9	0
00	or	nand	and	add	or	and	add	xor	nand	and
01	add	nor	or	xor	and	add	nor	add	xor	xor
10	and	add	xor	and	add	or	and	nor	and	nor
11	xor	xor	add	nand	xor	nand	xor	and	add	add

Мнемоники команд: add – суммирование, and – логическое И, or - логическое ИЛИ .xor - логическое Исключающее ИЛИ, nand – логическое И-НЕ, nor - логическое ИЛИ-НЕ.

3. Формируемые признаки (флажки) результата. 1) cf, zf, nf, pf

Код\Вариант	1	2	3	4	5	6	7	8	9	0
cf	+		+		+		+	+		
zf	+	+	+	+		+		+	+	+
nf	+	+	+	+	+	+	+		+	
pf	+			+			+	+	+	+

Обозначения: cf – признак переноса, zf – признак нуля, nf – признак отрицательного результата, pf – признака нечетности суммы всех единиц.

Номер студента в журнале группы определяет код варианта , который содержит 3 цифры, задающие разрядность, операции и признаки, определяется по.

Номер в журнале	1	2	3	4	5	6	7	8	9	10
Код варианта	180	129	138	147	156	165	174	183	192	101

Номер в журнале	11	12	13	14	15	16	17	18	19	20
Код варианта	215	226	237	248	259	261	272	283	294	205

Номер в журнале	21	22	23	24	25	26	27	28	29	30
Код варианта	310	329	338	347	356	365	374	383	392	301

Рассмотрим выполнение задания с кодм 111.

Разработка структурной схемы.

Схема устройства содержит комбинационную схему и регистры.

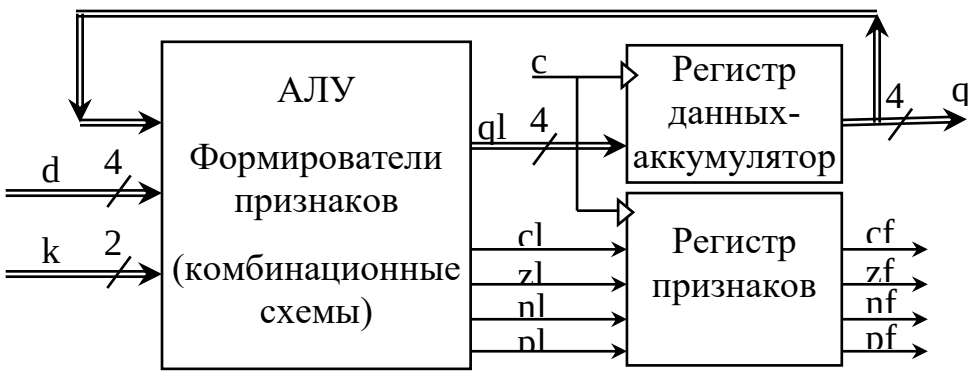
Обозначения: .

c – синхросигнал, k – код команды, d – входные данные,

q - выходные данные, ql выходной логический сигнал АЛУ,

cl, zl, nl, pl – логические сигналы на выходах комбинацинных схем – признаки переноса, нуля, отрицательного результата, нечетности суммы всех единиц соответственно.

cf, zf, nf, pf– признаки (флаги), записанные в регистр состояния.

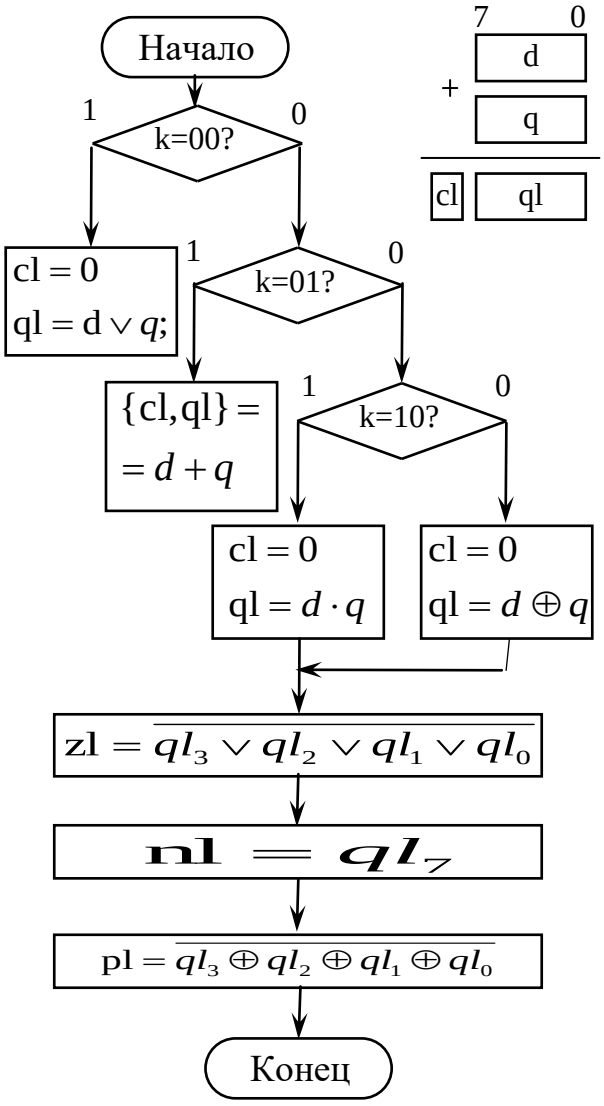


Структурная схема вычислительного устройства

В соответствии с выриантом 111 устройство обрабатывает 4- разрядные данные и формирует признаки cf, zf, nf, pf.

Описание работы вычислительного устройства.

АЛУ выполняет арифметические или логические операции в



Алгоритм работы комбинационной схемы

соответствии с входным кодом k. Например, если k=1, то выполняется суммирование n-разрядных аодов - содержимого аккумулятора q (это результат выполнения предыдущей команды) и входных данных d. Результатом выполнения текущей операции является логический сигнал ql = q + d. При суммировании возможно появление переноса cl. В зависимости сигнала ql формируются логические сигналы признаков результата zl, nl, pl.

Регистры данных и состояния (с динамическим управлением !) предназначены для записи новых значений выходного кода и признаков по результату текущей операции. По фронте синхросигнала с выполняется запись: q = ql; cf = cl; zf = zl; nf = nl; pf = pl.

Таким образом, устройство выполняет операцию вида: q := q + d, что означает: «новое значение q равно предыдущему значению плюс d».

Выбор сигналов синхронизации

Сигнал синхронизации содержит два состояния, которым соответствуют постоянные значения сигнала (***c*** = 0 и ***c*** = 1) и два изменения состояний - спад и фронт импульса. Необходимо определить действия, выполняемые в моменты, соответствующие различным состояниям синхросигнала.

Пусть спад импульса синхросигнала определяет начало машинного цикла - момент подключения команды на шину k и входных на шину d. Интервал ***c*** = 0 будет предназначен для обработки данных посредством устройств комбинационного типа (АЛУ и формирователей признаков). Фронт импульса синхросигнала будет выполнять запись данных, полученных в результате обработки (ql, cl, zl, nl, pl), в регистры данных и признаков. Интервал *c* = 1 предназначен для использования новых данных (q, cf, zf, nf, pf).

Разработка алгоритма работы комбинационной схемы.

Комбинационная схема содержит АЛУ и формирователи признаков, на выходахсхемы формируются логические сигналы, показанные на структурной схеме. . АЛУ выполняет операцию, определяемую кодом k, в соответствии с заданием, при k = 00 – операция ИЛИ, при k = 01 – суммирование, при k = 10 – И, при k = 11 – Исключающее ИЛИ. При суммировании может возникать перенос (при этом признак переноса cl=1), поэтому для всех команд принят (n+1) – разрядный формат выходного кода ql, lля логических операции cl тождественно равен 0. Для формирования заданных признаков использованы очевидные логические функции. Признак нуля zl = 1 , если все разряды данных равны нулю. Признак отрицательного числа nl – старший разряд кода числа, nl = 1 для отрицательного числа. Признак четности pl = 1, если сумм- мА бит в двоичной записи числа четна.

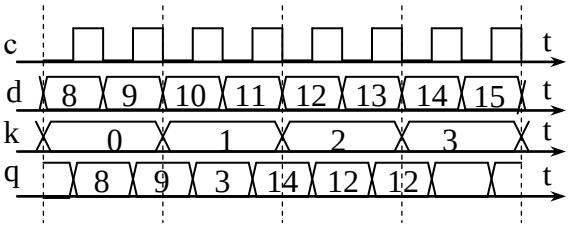
Описание модулей на языке Verilog

В описании комбинационной схемы Выбор варианта операции производится

посредством операторов условного присваивания, выходной сигнал которых содержит для всех операций n+1 разряд,

Для описания которого использованы операторы объединения. Для всех констант использован формат с указанием разрядности и двоичной системы счисления (первый элемент-количество разрядов, второй – апостроф, третий – буква b для двоичной системы, четвертый - двоичное число). Описание регистров обеспечивает запись а триггеры и регистры входных логических сигналов в по фронту синхросигнала c. Для всех выходных сигналов регистров указан тип сигналов reg. Запись сигналов выполняет последовательный оператор (строка 10). Для присваивания нескольких сигналов использованы скобки «begin - tnd».

Разработка теоретических временных диаграмм

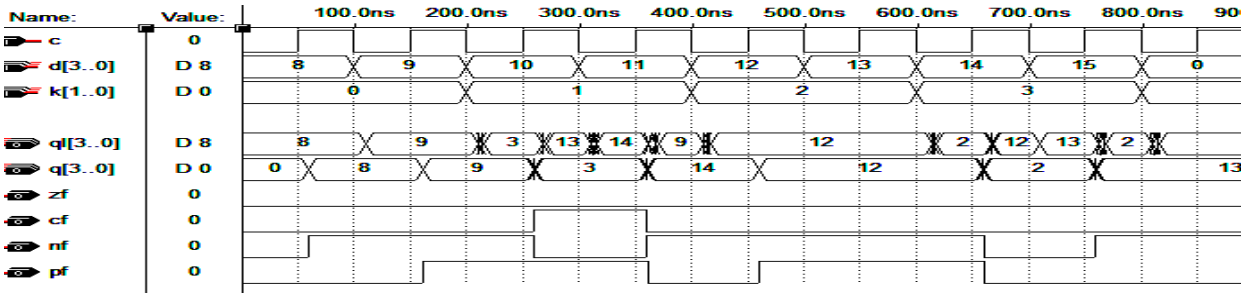


Теоретические временные диаграммы

Для тестирования устройства необходимо подать на его входы синхросигнал c, 4-разрядный код входных данных d, 2-разрядный код команды k. Для перечисленных сигналов целесообразно выбрать различные значения множителя, определяющего масштаб ро оси времени (например, 1, 2, 4). Спад импульсов синхронизации определяет моменты изменения данных и команд (отмечены пунтирными линиями). Выходной сигнал q изменяет значения в моменты его записи в регистр – по фронту синхросигнала.

```
//Описание Комб. схемы 1
module ks (d, q, k, ql,
 cl, zl, nl, pl);                //2
     input [3:0]d, q;                    //3
input [1:0]k ;                   //4
output cl, zl, nl, pl;           //5
output [3:0]ql;                  //6
assign  {cl,ql} =
(k == 2'b00) ? {1'b0,(q | d)}:
(k == 2'b01) ? q + d:
(k == 2'b10) ? {1'b0,(q & d)}:
{1'b0,(q ^ d)};                  //7
assign zl = ~(ql[3] |
 ql[2] |ql[1] | ql[0]);          //8
assign nl = ql[3];               //9
assign pl = ~(ql[3] ^
ql[2] ^ql[1] ^ ql[0]);           //10
endmodule                        //11
```

```
//Описание регистров   1
module registers (c, ql, cl,zl,
nl, pl, q, cf, zf, nf, pf);//2
input c;                         //3
input [3:0]ql;                   //4
input cl, zl, nl, pl ;           //5
output [3:0]q;                   //6
reg [3:0]q;                      //7
output cf, zf, nf, pf;           //8
reg cf, zf, nf, pf;              //9
always @ (posedge c)             //10
begin                            //11
q = ql; zf = zl; nf = nl;        //12
pf = pl; cf = cl;                //13
end                              //14
endmodule                        //15
```



Разработка проекта в САПР и моделирование. Анализ результатов моделирования.

По временным диаграммам необходимо изучить процесс выполнения

вычислений, определить временные задержки в модулях устройства. Используя экспериментальные данные определить максимальную допустимую тактовую частоту, проверить работу на этой частоте.

Работа 7. Устройство дл вычисления коэффициентов ряда Фурье

/Задание. Периодическая функция записывается в ПЗУ в виде таблицы из 16 чисел со знаком в дополнительном коде разрядностью 8 бит.

Устройство должно выполнить вычисление и запись в буферное ОЗУ значений амплитуд гармоник ai, bi для i =1..4.

Теоретическая часть

Вычисление амплитуд гармоник функции, заданной дискретными отсчетами, выполняется по формулам дискретного преобразования Фурье. Значение периодической функции f для точки отсчета n определяется суммой гармоник

$$f_n = \sum_{k=0}^{N/2}(a_k \cos k \cdot \frac{2\pi}{\mathrm{T}} \cdot n \cdot \tau + b_k \sin k \cdot \frac{2\pi}{\mathrm{T}} \cdot n \cdot \tau)\text{; } \omega_1 = 2\pi / \mathrm{T},\ \omega_k = k*2\pi / \mathrm{T}$$

Обозначения:

T – период функции; N – количество отсчетов функции;

n – номер отсчета функции; k – номер гармоники;

2π/T – шаг дискретизации в радианах; 360/T – шаг дискр. в градусах.

Коэффициенты ряда (амплитуды гармоник) определяют формулы

$$a_0 = \frac{2}{N}\sum_{n=0}^{N-1}\mathrm{f(n)};\quad a_n = \frac{2}{N}\sum_{n=1}^{N-1} f(n)\cos n \cdot \frac{2\pi}{\mathrm{T}}\cdot;\quad b_n = \frac{2}{N}\sum_{n=1}^{N-1} f(n)\sin n \cdot \frac{2\pi}{\mathrm{T}}\cdot;$$

Проект вычислительного устройства имеет иерархическую структуру, содержит отдельные устройства памяти для хранения исходных данных и результатов, что позволяет использовать раздельные ыдресные пространства. Рассмотрим назначение элементов устройства.

Обозначения переменных.

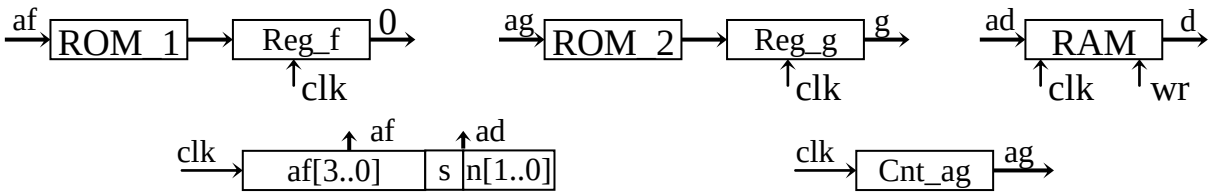


Рис.7.1. Регистровая модель вычислительного устройства

ag[3..0] – адрес гармоники для ПЗУ гарионического сигнала;

g[7..0] – код гармоники с выхода ПЗУ;

af[3..0] - адрес функции для ПЗУ входного сигнала;

f[7..0] - код входной функции;

a_mem - адрес ОЗУ для записи результата;

data – входные данные ОЗУ;

we - разрешенгие записи в ОЗУ.

Гармоническая функция времени g(t) = sin ωt представлена в виде массива в ПЗУ (ROM_1), также содержащего N=16 отсчетов с шагом дискретизации, равным 2π/N = 360/16 = 22,5°. После подачи адреса гармоники ag в регистре гармоники Reg_g фиксируется отсчет гармоники g. Отсчеты приведены в дополнительном коде. Для ввода данных в файл инициализации (*.mif) необходтиа 16-ричная система. При исследовании результатов вычислений целесообразна десятичная система счисления.

n	0	1	2	3	4	5	6	7
n*τ	0	22,5	45	67,5	90	112,5	135	157,5
sin	0	0,38	0,71	0,92	1	0,92	0,71	0,308
Д.к.	0	38	71	92	100	92	71	38
Hex	0	26	47	5c	64	5c	47	26

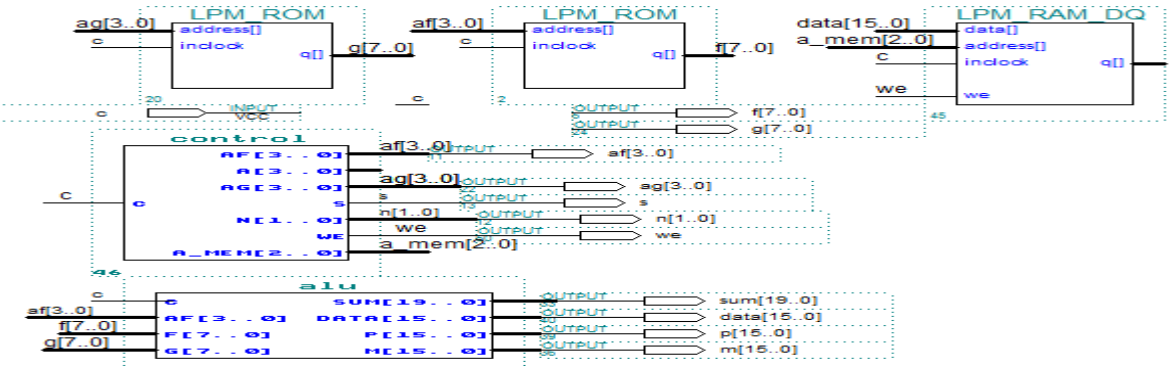
n	8	9	10	11	12	13	14	15
n*τ	180	202?5	225	247,5	270	292,5	315	337,5
sin	0	-0,38	-0,71	-0,92	-1	-0,92	-0,71	-0,38
Д.к.	0	218	185	164	156	164	185	218
Hex	0	da	b9	a4	9c	a4	b9	da

Входной сигнал задан в виде функции времени f(n) представленной в ПЗУ (ROM_2) в виде массива, содержащего N=16 отсчетов с тактом дискретизации T/N. После подачи адреса функции af в регистре функции Reg_f фиксируется отсчет функции f. При моделировании функция задается отсчетами в файле с расширением .mif. Например, для импульсного сигнала с периодом содержащим 16 отсчетов, при этом 8 отсчетов принимают значения +10, а другие 8 отсчетов принимают значение -10, таблица имеет вид.

n	0	1	2	3	4	5	6	7
f	**10**	10	10	10	10	10	10	10
Д.к.	10	10	10	10	10	10	10	10
Hex	0A	0A	0A	0A	0A	0A	0A	0A

n	8	9	10	11	12	13	14	15
f	-10	-10	-10	-10	-10	-10	-10	-10
Д.к.	246	246	246	246	246	246	246	246
Hex	F6	F6	F6	F6	F6	F6	F6	F6

Функции устройства управления (модуль control) выполняет счетчик, содержащий несколько полей. Такое включение существенно упрощает выполнение алгоритма, содержащего три вложенных цикла. В поле af, содержащем 4 разряда, формируется адрес функции, в поле s содержится бит признака анализируемой гармоники – (0 - синус, 1 - косинус). Двухразрядное поле n определяет номер гармоники. Код 0 соответствует 1-й гармонике; код 1 – 2-й; код 2 – 3-й; код 3 – 4-й гармонике.
В начале моделирования по умолчанию s = 0, n = 0, будет вычисляться коэффициент b1 (амплитуда первой гармоники функции sin), после переполнения счетчика адреса установится s = 1 и бдет вычисляться a1 (амплитуда первой гармоники cos). Впоследствии, при значениях n, равных 1, 2, 3 будут вычисляться коэффициенты b2, a2, b3, a3, b4, a4. Признак s используется при формировании адреса при обращении к ПЗУ гармоники. При s = 0 на адресный вход ПЗУ подается адрес, полученный в



```
module control
(c,af, a,ag, s,n, we,a_mem);
input c;
output [3:0] af;      reg [3:0] af;
output [3:0] a;       reg [3:0] a;
output [3:0] ag;
output [2:0] a_mem;
output s, we;         reg s ;
output [1:0] n;       reg [1:0] n;
always @ (posedge c)
begin
{n,s,af} = {n,s,af} +1;
a=a+1+n;
end
assign ag = s?a+4:a;
assign we = af==4'b1111;
assign a_mem = {n,s} ;
endmodule
```

регистре a, а при s = 1 к адресу прибавляется 4. Это ¼ периода, содержащего 16 отсчетов. Таким образом, с выхода ПЗУ будет получена функция cos.

Код, содержащийся в поле n определяет индексирование счетчика адреса гармоник - прибавление к адресу числа 1+n. В результате первой гармонике (n = 0) соответствует приращение адреса на 1, второй гармонике (n = 1) соответствует приращение адреса на 2, и т. д. В модуле АЛУ вычисляются произведения кодов гармоники и входной функции. Переменная m - модуль произведения определяется как произведение модулей сомножителей, которые представлены в дополнительном кое. Переменная p - произведение вычисляется в дополнительном коде. Переменна sum – накапливающий сумматор, его разрядность выбрана с учетом суммирования нескольких 16-разряднвх слагаемых. Выходной сигнал АЛУ – data определяется делением суммы (sum) на 16.

```
module alu (c,af,f,g,sum,data,p,m);
input c;
input [3:0] af;
input [7:0] f, g;
output [15:0] data;
output [19:0] sum;
reg [19:0] sum;
output [15:0] p;
output [15:0] m;
assign m =(f[7]? 256-f :f)*(g[7]? 256-g:g);
assign p = (f[7] ^ g[7]) ? 5'h10000-m:m;
always @ (posedge c)
if (af==0) sum=p; else
sum = sum+p;
assign data = sum[19:4];
endmodule
```

```
module control
(c,af, a,ag, s,n, we,a_mem);
input c;
output [3:0] af;      reg [3:0] af;
output [3:0] a;       reg [3:0] a;
output [3:0] ag;
output [2:0] a_mem;
output s, we;         reg s ;
output [1:0] n;       reg [1:0] n;
always @ (posedge c)
begin
{n,s,af} = {n,s,af} +1;
a=a+1+n;  end
assign ag = s?a+4:a;
assign we = af==4'b1111;
assign a_mem = {n,s} ;  endmodule
```

При моделировании необходимо выбрать шаг При моделировании необходимо выбрать шаг сетки 400 нс. время моделирования 51 мкс.