Федеральное агентство связи

Государственное федеральное образовательное учреждение высшего профессионального образования

ПОВОЛЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ

ЭЛЕКТРОННАЯ БИБЛИОТЕЧНАЯ СИСТЕМА

Самара

Поволжский государственный университет телекоммуникаций и информатики

Стефанов А. М., Стефанова И.А.

МОДЕЛИРОВАНИЕ ЦИФРОВЫХ УСТРОЙСТВ В СИСТЕМЕ MATLAB + Simulink

Задания и методические указания к лабораторным работам

Самара
2011

ОГЛАВЛЕНИЕ

Введение

Данный цикл содержит пять лабораторных работ, направленных на приобретение навыков синтеза цифровых устройств и освоение приемов их моделирования в системе MATLAB+ Simulink. Цикл может использоваться на лабораторных и практических занятиях по дисциплинам «Вычислительная техника и информационные технологии» и «Цифровые устройства и микропроцессоры» для студентов телекоммуникационных направлений подготовки.

Рекомендуемая литература

1. Хоровиц П. Искусство схемотехники / пер. с англ. – М.: БИНОМ: Мир, 2010.
2. Стефанов А.М. Вычислительная техника и информационные технологии: уч. пособие. – Самара: ПГАТИ, 2006.
3. Дьяконов В. П. MATLAB 6.5 SP1/7+ Simulink 5/6. Основы применения. – М.: СОЛОН-Пресс, 2005.
4. Конспект лекций по дисциплине.

Содержание отчета

1. Название работы.
2. Код группы, фамилия и инициалы студента.
3. Для каждого задания работы приводятся постановка задачи и описание ее решения: подробное описание вывода аналитических выражений, структурная схема устройства, прогноз и собственно результаты моделирования.
4. Выводы относительно приемов анализа и/или синтеза, используемых в данной лабораторной работе.

Ход выполнения лабораторной работы сохраняется во флэш-памяти. Соответствующий файл и отчет предъявляются преподавателю перед тестированием по данной работе.

Сохранение результатов лабораторной работы

Обычно файлы, созданные в системе MATLAB+ Simulink, автоматически сохраняются в рабочей папке C:\MAT-LAB\work. Однако это приемлемо лишь в случае индивидуального пользования персональным компьютером.

В случае же возможности массового доступа к папке work или отсутствии доступа к диску C: рабочую папку следует создать на любом другом из доступных дисков и даже в любой из доступных на них папок. При этом рабочей папке дается уникальное имя, например <номер группы><фамилия>, в котором если используются буквы, то желательно **латиницы.**

Для удобства систематизации файлов по лабораторным работам в рабочей папке целесообразно создать дерево папок (рис. 1), где в именах папок, а также содержащихся в них файлах желательно использовать опять-таки **латинские буквы**. При этом в имена папок удобно включить номера лабораторных работ.

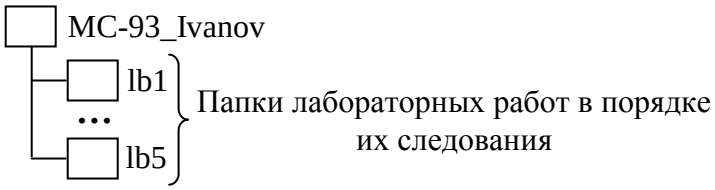


Рис. 1. Систематизация результатов выполнения лабораторных работ

При такой организации после запуска системы MATLAB, прежде всего, необходимо в адресном окне *Current Directory* (рис. 2) указать путь к нужной папке. Для этого удобно воспользоваться окном *Обзор папок*, которое открывается кнопкой *Browse for folder*.

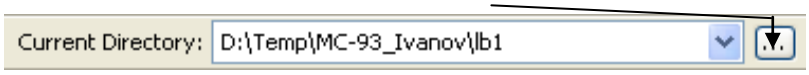


Рис. 2. Фрагмент панели инструментов окна MATLAB

1. ЗАКОНЫ И ТОЖДЕСТВА АЛГЕБРЫ ЛОГИКИ

В синтезе цифровых устройств используются в следующих целях:

– упрощение аналитического описания закономерности их работы, что приводит к предельно простой технической реализации. В результате уменьшаются энергопотребление и стоимость устройства;

– замена логических элементов одного типа логическими элементами другого типа.

Подготовка к работе

По указанной литературе изучить:

– элементарные функции алгебры логики (ФАЛ) и сложные формы их записи;

– типы, условное графическое обозначение, правила работы логических элементов (ЛЭ) и реализуемые ими ФАЛ;

– законы и тождества алгебры логики.

Задания и методические указания к их выполнению

1. Из таблицы в соответствии с номером варианта N (номер по списку группы) выбрать выражение исходной ФАЛ:

N	Выражение	N	Выражение
1	$a\downarrow(b\downarrow c)\vee a\wedge\bar{b}\wedge\bar{c}$	16	$(a\vee\bar{b})\wedge(a\vee\bar{c})$
2	$(\bar{a}\,\|\,b)\,\|\,(a\,\|\,\bar{b})\downarrow\bar{c}\vee c\downarrow(a\sim b)$	17	$\bar{a}\downarrow(b\vee c)$
3	$(\bar{a}\wedge\bar{b}\,\|\,c)\,\|\,(b\,\|\,\bar{c})$	18	$\bar{a}\wedge\bar{b}\,\|\,\bar{c}$
4	$\bar{a}\downarrow(b\wedge c)\wedge(b\vee c)$	19	$\bar{a}\vee(b\oplus c)\vee\bar{b}\wedge c$

5	$\bar{a} \wedge (b \vee \bar{c}) \wedge b \mid \bar{c}$	20	$\bar{a} \vee b \downarrow c$
6	$a \downarrow (\bar{b} \mid \bar{c})$	21	$\bar{a} \wedge (\bar{b} \vee \bar{c}) \vee a \wedge b \wedge c$
7	$\bar{a} \vee b \wedge c$	22	$a \vee \bar{b} \vee \bar{c} \vee a \wedge b \wedge c$
8	$\bar{a} \wedge b \wedge c \vee \bar{a} \downarrow (b \wedge c)$	23	$\bar{a} \mid (b \wedge c)$
9	$a \downarrow \bar{b} \wedge c \vee a \wedge (\bar{b} \vee \bar{c})$	24	$\bar{a} \downarrow (\bar{b} \downarrow \bar{c})$
10	$\bar{a} \mid (b \vee c) \vee b \wedge c$	25	$\bar{a} \wedge b \wedge c$
11	$a \wedge (b \vee c) \mid (b \mid c)$	26	$\bar{a} \vee \bar{b} \mid \bar{c}$
12	$a \downarrow (b \wedge c) \wedge b \downarrow c$	27	$\bar{a} \wedge (b \downarrow c \vee b \wedge c) \vee a \wedge (b \oplus c)$
13	$\bar{a} \vee b \sim c$	28	$\bar{a} \wedge \bar{b} \wedge \bar{c} \vee a \wedge b \vee a \wedge c$
14	$(a \vee b) \downarrow c \vee a \wedge b \vee a \wedge c$	29	$\bar{a} \wedge (\bar{b} \vee \bar{c})$
15	$\bar{a} \wedge b \vee \bar{a} \wedge c \vee a \wedge \bar{b} \wedge \bar{c}$	30	$(a \downarrow b) \downarrow (a \wedge b) \wedge \bar{c} \vee c \wedge (a \vee b)$ $(\bar{a} \vee \bar{b})$

2. Определить тип и количество логических элементов, необходимых для реализации исходной ФАЛ:

Тип	Реализуемая операция	Количество ЛЭ

……

Всего		

Например, реализация ФАЛ $y = a \mid (b \vee \bar{c}) \vee \bar{b} \downarrow c$ требует логические элементы:

Тип	Реализуемая операция	Количество ЛЭ
НЕ	Инверсия	2
ИЛИ	Дизъюнкция	2
И-НЕ	Штрих Шеффера	1
ИЛИ-НЕ	Стрелка Пирса	1
Всего		6

3. Упростить исходное выражение до двух логических операций.

Чтобы получить компактную запись сложного выражения, прежде всего, це-

Таблица 1. Пример работы с ФАЛ в командном окне

Исходная таблица истинности				Командное окно (Command Window) MATLAB
аргументы			функция	
c	*b*	*a*	y	
0	0	0	1	>>A=[0 1 0 1 0 1 0 1]; >>B=[0 0 1 1 0 0 1 1]; >>C=[0 0 0 0 1 1 1 1]; >>Y1= ~(A&(B|~C))|~(~B|C) Y1 = 1 0 1 1 1 1 1 0 >> Y2= ~(A&~xor(B,C)) Y2 = 1 0 1 1 1 1 1 0
0	0	1	0	
0	1	0	1	
0	1	1	1	
1	0	0	1	
1	0	1	1	
1	1	0	1	
1	1	1	0	

лесообразно заменить в нем такие элементарные ФАЛ, как стрелка Пирса, штрих Шеффера, исключающее ИЛИ и эквивалентность их сложной формой записи. При этом необходимо помнить, что порядок вычисления логического выражения определяется тремя основными факторами:

– скобки. Выражения в скобках всегда вычисляются первыми;

– приоритет операций. По убыванию приоритета логические операции располагаются в следующем порядке:

- инверсия отдельного аргумента,
- штрих Шеффера, стрелка Пирса, эквивалентность,
- конъюнкция,
- операции типа дизъюнкции – собственно дизъюнкция и исключающее ИЛИ (неравнозначность);

– когда порядок вычисления выражения не задан скобками, а уровень приоритета операций одинаков, выражение вычисляется слева направо.

Например, выражение $a|(b\vee\bar{c})\vee\bar{b}\downarrow c$ следует переписать как $\overline{a\wedge(b\vee\bar{c})}\vee\overline{\bar{b}\vee c}$.

Затем в полученном выражении с помощью законов и тождеств алгебры логики, а также сложной формой записи элементарных ФАЛ добиваются наименьшего числа операций. При этом в случае использования закона двойственности знак общей инверсии преобразуется в скобки.

Так, для приведенного примера:

$$\overline{a\wedge(b\vee\bar{c})}\vee\overline{\bar{b}\vee c}=(\bar{a}\vee\overline{b\vee\bar{c}})\vee b\wedge\bar{c}=\bar{a}\vee\bar{b}\wedge c\vee b\wedge\bar{c}=$$

$$\bar{a}\vee(b\oplus c)=\overline{\overline{\bar{a}\vee(b\oplus c)}}=\overline{a\wedge\overline{b\oplus c}}=\overline{a\wedge(b\sim c)}=a|(b\sim c).$$

4. С помощью системы MATLAB проверить эквивалентность исходной и упрощенной ФАЛ.

Различные ФАЛ от одного и того же количества аргументов эквивалентны, если на всем множестве входных двоичных наборов их значения совпадают.

В системе MATLAB значения ФАЛ можно получить следующим образом. В командном окне (Command Window) системы MATLAB построчно в виде векторов-строк вводятся значения аргументов ФАЛ в соответствии с последовательностью входных двоичных наборов (табл. 1). При этом соседние элементы вектора-строки разделяются пробелом, а командная строка завершается символом «;», который означает запрет вывода на экран результата выполнения данной командной строки после перехода на следующую строку путем нажатия клавиши *Enter*.

После этого в очередной командной строке вводится ФАЛ, логическое выра-

жение которой записывается с помощью стандартных для языка MATLAB операторов или функций (табл. 2). При этом в конце командной строки символ «;» не ставится, что после нажатия клавиши *Enter* обусловливает вывод на экран результатов вычислений в виде вектора-строки с указанием имени функции (см. табл. 1).

Таблица 2. Логические операторы и функции MATLAB

Функция	Название	Оператор
and()	Логическое И	&
or()	Логическое ИЛИ	|
not()	Логическое НЕ	~
xor()	Исключающее ИЛИ	

Пример записи ФАЛ с помощью операторов языка MATLAB приведен в табл. 1. Конечно, ФАЛ можно записать и с помощью только функций MATLAB, например

Y1=or(not(and(A,or(B,not(C)))),not(or(not(B),C))),
Y2=not(and(A,not(xor(B,C)))).

Но тогда выражения становятся громоздкими, что повышает вероятность ошибки как при записи, так и вводе в рабочее поле того или иного окна MATLAB.

5. Определить тип и количество логических элементов, необходимых для реализации полученной ФАЛ. На основании сравнения с таблицей п.2 работы сделать вывод относительно целесообразности упрощения ФАЛ.

2. КОМБИНАЦИОННЫЕ ЦИФРОВЫЕ УСТРОЙСТВА (КЦУ)

Применяются для построения потенциальных автоматов – автоматов без памяти. В таких реализациях значение выходной переменной зависит только от значений входных переменных.

Подготовка к работе

По указанной литературе изучить методику синтеза КЦУ.

Задания и методические указания к их выполнению

1. Из табл. 3 в соответствии с номером варианта N выбрать закон функционирования КЦУ с четырьмя входами и одним выходом.

В таблице в круглых скобках указаны десятичные номера двоичных наборов, на которых функция принимает значение 0, в квадратных скобках – 1, а не указанные наборы являются безразличными.

2. Методами карт Карно и Квайна получить минимальную ФАЛ.

Таблица 3. Закон функционирования КЦУ

N	Скобочная запись таблицы истинности	N	Скобочная запись таблицы истинности
1	[0,1,2,8,9,(11,12,13,14,15)]	16	(0,1,2,8,9,[11,12,13,14,15])

2	[1,2,3,9,10,(8,12,13,14,15)]	17	(1,2,3,9,10,[8,12,13,14,15])
3	[2,3,4,10,11,(7,8,13,14,15)]	18	(2,3,4,10,11,[7,8,13,14,15])
4	[3,4,5,11,12,(6,7,13,14,15)]	19	(3,4,5,11,12,[6,7,13,14,15])
5	[4,5,6,12,13,(0,1,8,14,15)]	20	(4,5,6,12,13,[0,1,8,14,15])
6	[5,6,7,13,14,(0,1,8,9,15)]	21	(5,6,7,13,14,[0,1,8,9,15])
7	[6,7,8,14,15,(0,1,9,10,11)]	22	(6,7,8,14,15,[0,1,9,10,11])
8	[0,1,8,9,10,(4,5,6,13,15)]	23	(0,1,8,9,10,[4,5,6,13,15])
9	[1,3,5,10,11,(2,4,8,12,13)]	24	(1,3,5,10,11,[2,4,8,12,13])
10	[3,5,7,11,12,(4,8,10,14,15)]	25	(3,5,7,11,12,[4,8,10,14,15])
11	[5,7,9,13,14,(6,8,12,11,15)]	26	(5,7,9,13,14,[6,8,12,11,15])
12	[7,9,11,13,15,(4,6,8,10,12)]	27	(7,9,11,13,15,[4,6,8,10,12])
13	[0,2,4,13,15,(5,6,8,10,12)]	28	(0,2,4,13,15,[5,6,8,10,12])
14	[2,4,6,10,12,(8,9,11,13,15)]	29	(2,4,6,10,12,[8,9,11,13,15])
15	[4,6,8,12,14,(1,3,9,13,15)]	30	(4,6,8,12,14,[1,3,9,13,15])

При формировании областей в карте Карно следует начинать с клетки, соответствующей определенному входному набору и в наименьшей степени связанной с другими аналогичными клетками. Этот прием повторяется для остальных еще не сгруппированных клеток, соответствующим определенным входным наборам, и т.д. При таком подходе удается избежать лишних областей.

В методе Квайна пара подбирается к каждому еще не сгруппированному члену ФАЛ. Кроме того, метод Квайна носит циклический характер. То есть к результату преобразования вновь применяется группировка и т.д. В случае сложного конечного результата заново минимизировать ФАЛ, по-другому группируя ее члены. Кроме того, полученный результат иногда можно дополнительно упростить с помощью законов и тождеств алгебры логики.

3. С помощью системы MATLAB проверить эквивалентность минимальной ФАЛ исходной таблице истинности.

Данное задание выполняется аналогично п.4 предыдущей лабораторной работы за одним исключением: в векторах-строках следует использовать только значения аргументов, соответствующие определенным двоичным наборам.

4. Записать минимальную ФАЛ в базисе И-НЕ для четных N и ИЛИ-НЕ для нечетных N.

С этой целью используются законы двойной инверсии и двойственности.

Например, $y = ab \vee c$ в базисе:

– И-НЕ: $y = ab \vee c = \overline{\overline{ab \vee c}} = \overline{\overline{ab} \wedge \overline{c}}$;

– ИЛИ-НЕ: $y = ab \vee c = \overline{\overline{ab}} \vee c = \overline{\overline{a} \vee \overline{b}} \vee c = \overline{\overline{\overline{a} \vee \overline{b}} \vee c}$.

5. В соответствии с ФАЛ, полученными в предыдущем пункте, разработать структурную схему устройства.

6. В системе MATLAB+Simulink создать модель устройства, убедиться в достоверности проведенного синтеза и сохранить модель под именем kcи<№ варианта>.mdl.

Процесс моделирования цифрового устройства можно разделить на 3 этапа:

1. Подготовка поля модели и настройка параметров моделирования.

а). Запустить систему MATLAB, а затем библиотеку Simulink Library Browser (разделы библиотеки компоновщика моделей). Эта библиотека открывается либо командой *Start\Simulink\Library Browser* (кнопка Start находится в левом нижнем углу окна MATLAB), либо нажатием кнопки *Simulink* панели инструментов окна MATLAB:

Каждый раздел библиотеки может содержать подразделы (рис. 3). Управление просмотром содержимого раздела и подраздела аналогично управлению деревом папок в проводнике операционной системы Windows.

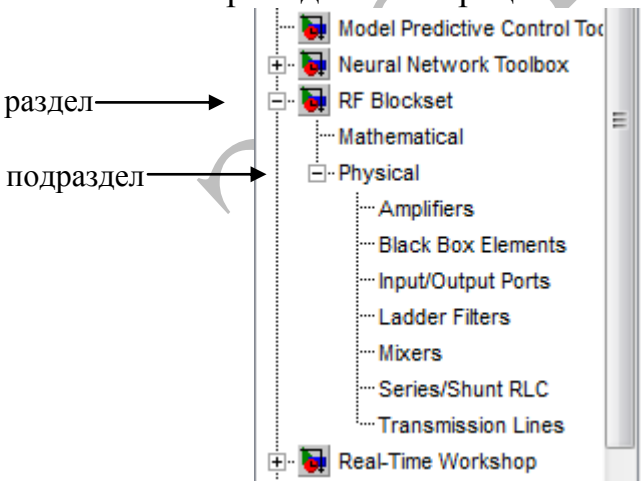
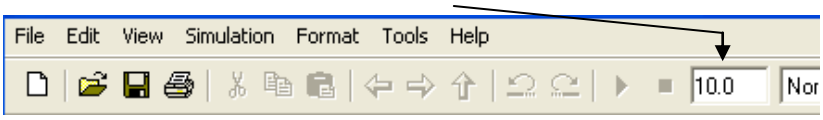


Рис. 3. Структура Simulink Library Browser

б). Открыть окно модели, для чего в окне Simulink Library Browser открыть пункт *File* оконного меню и в опции *New* выбрать команду *Model* (*Script*).

в). С целью настройки параметров моделирования в окне модели выполнить следующие действия:



– в окне *Simulation stop time* (конечное время моделирования) панели инструментов задать необходимое время моделирования – количество тактов работы устройства. Обычно оно определяется числом определенных состояний устройства и подсчитывается, начиная с нуля;

– в пункте меню *Simulation* выбрать команду *Configuration Parameters*...(конфигурация параметров …). В списке *Solver* (разрешение) открывшегося окна выбрать *discrete* (*no continuous states*) (дискретное (не непрерывные состояния)) и кнопкой *OK* закрыть данное окно.

2. Построение модели.

Модель должна содержать блок ввода исходных двоичных наборов, структурную схему устройства и блок регистрации результатов моделирования. Все эти блоки располагаются в рабочем поле окна модели в указанном порядке слева направо путем перетаскивания их компьютерной мышью из соответствующих разделов и/или подразделов библиотеки Simulink. В данной работе целесообразны следующие блоки:

а) From workspace (из рабочего поля) – находится в подразделе Sources (источники) раздела Simulink (окно Simulink Library Browser). Блок читает значения аргументов ФАЛ (входные двоичные наборы) из командного окна MATLAB.

При моделировании схемы устройства с числом входов (аргументов ФАЛ) k >1 требуется k блоков From workspace.

Для настройки параметров блока используется соответствующее окно диалога (открывается двойным щелчком компьютерной мышью по его изображению или командой …*Parameters*… контекстного меню), где:

– в строке *Data* ввести имя аргумента **в строгом соответствии** с форматом его записи в командном окне MATLAB;

– в строке *Sample time* задать шаг изменения времени (тактов), в случае цифровых устройств равный 1;

– в списке *Form output after-final data value by* (значение данных после последнего такта) выбрать *Setting to zero* (установить в 0).

Значения аргументов записываются в виде матрицы:

$$X = \left|\begin{array}{ccc|c} 0 & 0 & \ldots & 1 \\ \ldots & \ldots & \ldots & \ldots \\ n\text{-}1 & 1 & \ldots & 0 \end{array}\right|,$$

номер такта: $x_0 \quad \ldots \quad x_{N-1}$

аргументы ФАЛ

где n – число **определенных** двоичных наборов, а N – количество аргументов ФАЛ (входов устройства). При этом, во-первых, достаточно указать только определенные двоичные наборы, что вполне оправдано. Во-вторых, соответствующие такты модели имеют сквозную нумерацию (0, 1, 2, …) в независимости от номеров определенных двоичных наборов.

При вводе в командное окно MATLAB значения каждого аргумента (столбца матрицы) располагаются в соответствующем векторе-строке. Каждый элемент этого вектора представляется в формате:

<номер такта><пробел><значение аргумента в такте>

и отделяется от соседнего элемента символом «;».

Пример матриц входных двоичных наборов для частично определенного устройства с двумя входами *a, b* и одним выходом *y* приведен в табл. 4;

б) Logical Operator (логический оператор) – расположен в подразделе Logic and Bit Operations (логика и побитные операции) раздела Simulink библиотеки. Блок реализует логические элементы типа И (AND), ИЛИ (OR), НЕ (NOT), И-НЕ (NAND), ИЛИ-НЕ (NOR), исключающее ИЛИ (XOR).

Таблица 4. Пример матриц исходных данных для блоков From workspace

№ такта модели	Исходная таблица истинности			Командное окно (Command Window)
	Аргументы		Функция	
	b	a	y	
0	0	0	0	>>A=[0 0;1 0;2 1];
	0	1	~	>>B=[0 0;1 1;2 1];
1	1	0	0	>> Y' % вывод результата в строку
				ans =

Для настройки параметров блока на вкладке *Main* (главная) соответствующего окна диалога выполнить следующие действия:

– в списке *Operator* выбрать нужный оператор;

– в строке *Number of input ports* задать нужное количество входов;

в) To workspace (в рабочее поле) – находится в подразделе Sinks (приемники) раздела Simulink библиотеки. Блок регистрирует результаты моделирования, но лишь по одному из выходов устройства или по одной его шине. При этом в случае шины данные представляются в блоке матрицей, *i*-я строка которой соответствует *i*-у такту работы устройства. В случае же отдельного выхода устройства данные представляются вектором-столбцом.

Для настройки параметров блока в соответствующем окне диалога выполнить следующие действия:

– в строку *Variable name* (имя переменной) ввести имя выходной переменной (ФАЛ);

– в списке *Save format* выбрать *Array* (массив);

– снять флаг *Log fixed-point data as a fi object*.

Наконец, блоки соединяются путем протягивания компьютерной мышью с нажатой левой ее клавишей выхода одного блока к входу другого. Соединение организуются с помощью левой клавиши мыши (протягивание от выхода одного блока до входа другого или наоборот), а узел с разветвлением – правой клавишей. Так, модель для примера, приведенного в табл. 4, показана на рис. 4.

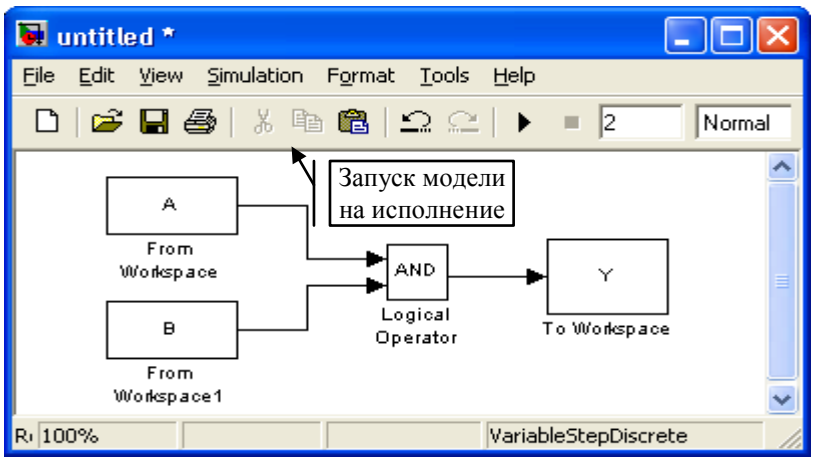


Рис. 4. Моделирование логического элемента И

3. Запуск модели на исполнение и просмотр результатов моделирования.

Созданная модель запускается на исполнение кнопкой *Start Simulation*, расположенной на панели инструментов окна модели (см. рис. 4).

По окончании заданного цикла работы в окне Workspace системного окна MATLAB образуются папки под именами выходных переменных, помеченные галочкой. Доступ к их содержимому обеспечивает двойной щелчок компьютерной мыши по соответствующей пиктограмме.

Другим и более простым способом просмотра результатов является их вывод в командное окно MATLAB как показано в табл. 4, где апостроф в имени выходной переменной означает транспонирование матрицы или вектора-столбца блока To workspace.

7. Сравнить результаты моделирования с исходной таблицей истинности и сделать вывод относительно достоверности проведенного синтеза.

3. ТИПОВЫЕ КЦУ

Применяются для преобразования двоичных наборов в унитарный код (активным может быть только один выход устройства) и наоборот (активным может быть только один вход устройства), образования каналов в едином цифровом потоке, реализации арифметических операций, преобразования кода одной системы кодирования в код другой системы кодирования.

Подготовка к работе

По указанной выше литературе изучить:

– условное графическое обозначение и правила работы типовых КЦУ: дешифраторов, шифраторов и мультиплексоров;

– особенности синтеза указанных типовых КЦУ;

– принципы и методы построения многоразрядных дешифраторов и мультиплексоров на базе соответствующих интегральных микросхем.

Задания и методические указания к их выполнению

1. В соответствии с номером варианта N табл. 5 для приоритетного ***шифратора*** *5×3 без стробирования с прямыми входами для четных N и инверсными входами для нечетных N на основе только анализа таблицы истинности и законов алгебры логики вывести систему минимальных ФАЛ.*

Таблица 5. Закон функционирования шифратора

N	Требуемые выходные двоичные наборы	N	Требуемые выходные двоичные наборы	N	Требуемые выходные двоичные наборы
1	3, 4, 5, 6, 7	11	2, 4, 5, 6, 7	21	2, 3, 4, 5, 7
2	0, 4, 5, 6, 7	12	0, 3, 5, 6, 7	22	0, 3, 4, 5, 6
3	0, 1, 5, 6, 7	13	0, 1, 4, 6, 7	23	0, 1, 3, 5, 6
4	0, 1, 2, 6, 7	14	0, 1, 2, 5, 7	24	2, 3, 4, 5, 6
5	0, 1, 2, 3, 7	15	0, 1, 2, 3, 6	25	0, 2, 3, 5, 7
6	2, 3, 5, 6, 7	16	2, 3, 4, 6, 7	26	0, 1, 3, 4, 7
7	0, 3, 4, 6, 7	17	0, 3, 4, 5, 7	27	1, 3, 5, 6, 7
8	0, 1, 4, 5, 7	18	0, 1, 4, 5, 6	28	0, 2, 3, 4, 6
9	0, 1, 2, 5, 6	19	0, 1, 2, 4, 6	29	1, 2, 4, 6, 7

Сократить время разработки цифрового устройства можно за счет упрощения процедуры вывода минимальных ФАЛ. Для этого достаточно воспользоваться особенностями таблицы истинности, характерными для данного устройства.

В качестве примера рассмотрим задачу синтеза приоритетного шифратора 5×3 без стробирования с прямыми входами и требуемыми двоичными наборами на выходе 0, 2, 4, 5, 7.

В соответствующей таблице истинности (табл. 6) приведена двойная индексация аргументов: без скобок отвечает номерам выходных двоичных наборов, а в скобках – естественной последовательности входов. Аналогично для функций: без скобок отвечает номерам двоичных разрядов, а в скобках – естественной последовательности выходов.

Таблица 6. Таблица истинности шифратора примера

Аргументы (входы)					Функции		
x_7	x_5	x_4	x_2	x_0	y_2	y_1	y_0
(x_5)	(x_4)	(x_3)	(x_2)	(x_1)	(y_3)	(y_2)	(y_1)
0	0	0	0	1	0	0	0
0	0	0	1	~	0	1	0
0	0	1	~	~	1	0	0
0	1	~	~	~	1	0	1
1	~	~	~	~	1	1	1

Имена переменных в скобках удобны при программном моделировании, поскольку в MATLAB нумерация элементов матриц начинается с 1.

Как видно из табл. 6, $y_0 = 1$, если x_5 или x_7 равны 1. Действительно, остальные аргументы могут принимать значение как 0, так и 1. Вследствие этой неоднозначности определяющую роль в значении y_0 они выполнять не могут. Таким образом, $y_0 = x_5 \vee x_7$.

Далее, первое сверху единичное значение y_1 (см. табл. 6) определяется аргументами x_2, x_4, x_5, x_7, а второе – только x_7. На этом основании можно записать ФАЛ в ДНФ: $y_1 = \overline{x}_7\overline{x}_5\overline{x}_4x_2 \vee x_7$. Отсюда после применения законов двойной инверсии и двойственности окончательно получаем: $y_1 = \overline{x}_5\overline{x}_4x_2 \vee x_7$.

Рассуждая аналогично, нетрудно получить минимальную ФАЛ для последнего выхода устройства: $y_2 = x_4 \vee x_5 \vee x_7$.

2. В командном окне системы MATLAB с помощью соответствующей программы убедиться в достоверности полученного аналитического описания работы шифратора.

Программирование в командном окне является еще одним способом моделирования. Однако приемлем он лишь для относительно простых устройств.

В таком моделировании, прежде всего, необходимо определиться относительно вида представления входных и выходных данных. В системе MATLAB их можно задать в виде вектора-строки или матрицы. В первом случае значения функции придется получать последовательно, всякий раз задавая следующий по порядку входной двоичный набор. В результате потребуется *n* прогонов программы, где *n* – число строк таблицы истинности. Во втором случае достаточно одного прогона, что и определяет выбор вида представления данных.

Далее необходимо решить вопрос о структуре входных двоичных наборов. Этот параметр должен отвечать единственному требованию – естественным образом отвечать особенностям моделируемого устройства. В данном случае эта особенность заключается в возможности активного состояния сразу нескольких входов. Следовательно, во входных наборах безразличные значения аргументов необходимо заменить активными.

Например, с учетом сказанного и в соответствии с табл. 6 матрицы входных (X) и выходных (Y) данных будут иметь вид:

$$X = \begin{vmatrix} 1\,0\,0\,0\,0 \\ 1\,1\,0\,0\,0 \\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot \\ 1\,1\,1\,1\,1 \end{vmatrix},\ Y = \begin{vmatrix} 0\,0\,0 \\ 0\,1\,0 \\ \cdot\ \cdot\ \cdot \\ 1\,1\,1 \end{vmatrix}.$$

Далее на основании полученных матриц и аналитического описания работы устройства разрабатывается программа на языке MATLAB, которая вводится в командное окно системы. При этом следует пользоваться именами переменных, индексированными в естественной последовательности входов и выходов устройства (для табл. 6 – в скобках).

Например, рассмотренному шифратору соответствует следующее из возможных программных решений:

```
>>X=[1 0 0 0 0;1 1 0 0 0;1 1 1 0 0;1 1 1 1 0;1 1 1 1 1];
>>for i=1:5
Y(i,3)=X(i,5)|X(i,4);
Y(i,2)=X(i,5)|not(X(i,4))&not(X(i,3))&X(i,2);
Y(i,1)=X(i,5)|X(i,4)|X(i,3);
end;
```

>>Y

3. В соответствии с номером варианта N табл. 7 для ***дешифратора*** *3×5 без стробирования с прямыми выходами для четных N и инверсными выходами для нечетных N на основе только анализа таблицы истинности и законов алгебры логики вывести систему минимальных ФАЛ.*

Таблица 7. Закон функционирования дешифратора

N	Определенные входные двоичные наборы	N	Определенные входные двоичные наборы	N	Определенные входные двоичные наборы
1	3, 4, 5, 6, 7	11	2, 4, 5, 6, 7	21	2, 3, 4, 5, 7
2	0, 4, 5, 6, 7	12	0, 3, 5, 6, 7	22	0, 3, 4, 5, 6
3	0, 1, 5, 6, 7	13	0, 1, 4, 6, 7	23	0, 1, 3, 5, 6
4	0, 1, 2, 6, 7	14	0, 1, 2, 5, 7	24	2, 3, 4, 5, 6
5	0, 1, 2, 3, 7	15	0, 1, 2, 3, 6	25	0, 2, 3, 5, 7
6	2, 3, 5, 6, 7	16	2, 3, 4, 6, 7	26	0, 1, 3, 4, 7
7	0, 3, 4, 6, 7	17	0, 3, 4, 5, 7	27	1, 3, 5, 6, 7
8	0, 1, 4, 5, 7	18	0, 1, 4, 5, 6	28	0, 2, 3, 4, 6
9	0, 1, 2, 5, 6	19	0, 1, 2, 4, 6	29	1, 2, 4, 6, 7
10	1, 2, 3, 5, 7	20	1, 2, 3, 4, 6	30	1, 2, 4, 5, 6

Данная задача решается аналогично предыдущей задаче. Тем не менее, имеет смысл кратко повторить рассуждения.

В качестве примера рассмотрим задачу синтеза дешифратора 3×5 без стробирования с инверсными выходами и определенными входными двоичными наборами 1, 2, 4, 5, 7. Соответствующая таблица истинности имеет вид:

Аргументы (входы)			Функции (выходы)				
x_2	x_1	x_0	y_7	y_5	y_4	y_2	y_1
(x_3)	(x_2)	(x_1)	(y_5)	(y_4)	(y_3)	(y_2)	(y_1)
0	0	1	1	1	1	1	0
0	1	0	1	1	1	0	1
1	0	0	1	1	0	1	1
1	0	1	1	0	1	1	1
1	1	1	0	1	1	1	1

Из таблицы видно, что $y_1 = 0$, если $x_2 = x_1 = 0$, поскольку в других входных наборах такой комбинации нет. Следовательно, $y_1 = x_2 \vee x_1$.

Повторяя приведенное рассуждение для остальных выходов, получаем:

$$y_2 = x_2 \vee x_0,\ \ y_4 = x_1 \vee x_0,\ \ y_5 = \bar{x}_2 \vee x_1 \vee \bar{x}_0,\ \ y_7 = \bar{x}_2 \vee \bar{x}_1.$$

Окончательное решение сформулируем, исходя из удобства последующего моделирования:

$$y_1 = x_3 \vee x_2,\ y_2 = x_3 \vee x_1,\ y_3 = x_2 \vee x_1,$$

$$y_4 = \bar{x}_3 \vee x_2 \vee \bar{x}_1,\ y_5 = x_3 \quad (1).$$

4. Создать программный модуль (подпрограмму), соответствующий условиям работы дешифратора и сохранить его под именем DC<№ варианта>.m.

Модульное программирование – другой, более развитый способ программного моделирования. Он позволяет моделировать системы и устройства любой сложности. Кроме того, при этом предоставляются значительно более удобные средства редактирования программы.

В терминах MATLAB подпрограммой является *m*-функция. После сохранения на жестком диске соответствующий файл называют *m*-файлом, который становится частью системы MATLAB и его можно вызывать как из командной строки, так и из другого *m*-файла.

Подпрограмма пишется на языке MATLAB в окне текстового редактора (Editor), которое открывается командой *File/New/M-File*. При этом, как и в п.2 данной работы, прежде всего, определяют вид и структуру представления входных и выходных данных.

Для приведенного <u>примера</u> данные целесообразно представлять в виде матриц:

$$X = \begin{vmatrix} x_{11}\ x_{12}\ x_{13} \\ \cdots\cdots\cdots \\ x_{51}\ x_{52}\ x_{53} \end{vmatrix},\ Y = \begin{vmatrix} y_{11}\ y_{12}\ y_{13}\ y_{14}\ y_{15} \\ \cdots\cdots\cdots\cdots\cdots \\ y_{51}\ y_{52}\ y_{53}\ y_{54}\ y_{55} \end{vmatrix}, \quad (2)$$

где каждая строка соответствует одному из пяти определенных двоичных наборов. Что касается структуры входных наборов, то изменить ее без нарушения закономерности работы устройства нельзя. Поэтому она остается той же, что и в таблице истинности.

Тогда и в соответствии с (1) *m*-функцию можно записать в виде:

```
function Y=DC(X)
for i=1:5
    Y(i,1)=X(i,3)|X(i,2);
    Y(i,2)=X(i,3)|X(i,1);
    Y(i,3)=X(i,2)|X(i,1);
    Y(i,4)=~X(i,3)|X(i,2)|~ X(i,1);
    Y(i,5)=not(X(i,3)&X(i,2));
end;
Y;
```

5. В командном окне системы MATLAB убедиться в достоверности полученного аналитического описания работы дешифратора.

Поскольку *m*-файл является самостоятельной моделью, его можно использовать для исследования поведения соответствующего устройства. С этой целью в командном окне достаточно задать исходные данные, после чего ввести его имя: <имя функции>=<имя *m*-файла>.

Для приведенного <u>примера</u> командные строки будут иметь вид:

```
>>X=[1 0 0;0 1 0;0 0 1;1 0 1;1 1 1];
>>y=DC(X)
```

6. Используя предыдущее устройство, синтезировать ***мультиплексор*** *5×1 без стробирования с прямым выходом.*

Обобщенная структура мультиплексора на основе дешифратора показана на рис. 5, где логический элемент (ЛЭ) служит для организации мультиплексной линии.

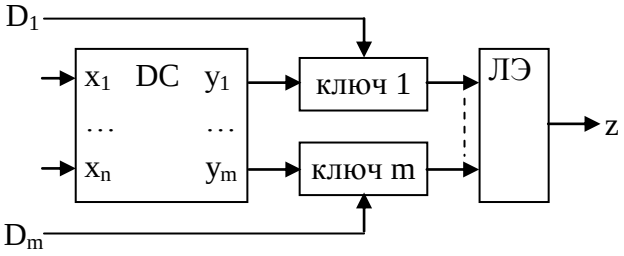


Рис. 5. Обобщенная структурная схема мультиплексора

Поскольку активным является только один выход дешифратора, открытым может быть лишь один из *m* ключей, соответствующий этому выходу. Отсюда определяется тип ЛЭ: сигнал на выходе закрытого ключа не должен изменять реакцию ЛЭ на сигнал с выхода открытого ключа. Таким образом, если в закрытом состоянии на выходе ключа 1, то ЛЭ должен быть типа И, а если 0 – то типа ИЛИ.

Тип логического элемента, используемого в качестве ключа, определяется аналогично, но в зависимости от активного сигнала на выходах дешифратора.

После этого остается записать минимальную ФАЛ $z = f(y_1, \ldots, y_m, D_1, \ldots, D_m)$, чем и завершается синтез данного устройства.

В частности, для приведенного примера активным сигналом на выходах дешифратора является 0. Следовательно, в качестве ключей должны использоваться логические элементы типа ИЛИ. Тогда на выходе закрытого ключа 1. Отсюда ЛЭ должен быть типа И. Следовательно, искомая ФАЛ имеет вид:

$$z = (y_1 \vee D_1) \cdot (y_2 \vee D_2) \cdot \ldots \cdot (y_5 \vee D_5).$$

В случае дешифратора с прямыми выходами в качестве ключей следует использовать логические элементы типа И, а ЛЭ должен быть типа ИЛИ. В этих условиях искомая ФАЛ будет иметь вид:

$$z = y_1 D_1 \vee y_2 D_2 \vee \ldots \vee y_5 D_5.$$

7. Создать подпрограмму, соответствующую ФАЛ $z = f(y_1, \ldots, y_m, D_1, \ldots, D_m)$ *и сохранить ее под именем K<№ варианта>.m.*

Например, в соответствии с (2) и (3) *m*-функцию можно записать в виде:

```
function Z=K(Y,D)
for i=1:5
Z(i)=(Y(i,1)|D(i,1))&(Y(i,2)|D(i,2))&(Y(i,3)|D(i,3))&(Y(i,4)|D
    (i,4))&(Y(i,5)|D(i,5));
end;
Z;
```

8. В командном окне системы MATLAB убедиться в достоверности полученного аналитического описания работы мультиплексора.

Наглядное отображение соответствия модели реальному мультиплексору можно получить, например, путем чередования 1 и 0 на информационных входах модели. В частности, подать на нечетные входы 1, а на четные – 0.

Но здесь есть еще одно обстоятельство: программные модули *DC* и *K* являются звеньями одной сложной системы, причем модуль *K* помимо связи с выходом модуля *DC* имеет самостоятельные информационные входы. Выходом из этой ситуации является последовательный вызов модулей в командном окне с предварительным объявлением соответствующих входных наборов.

С учетом сказанного для приведенного примера командные строки имеют вид:

```
>>X=[1 0 0;0 1 0;0 0 1;1 0 1;1 1 1];
>>Y=DC(X);
>>D=[1 0 0 0 0;1 0 1 1 1;0 0 1 0 0;1 1 1 0 1;0 0 0 0 1];
>>Z=K(Y,D)
```

9. Сделать вывод относительно приемов быстрого вывода минимальных ФАЛ для типовых КЦУ.

4. ТРИГГЕРЫ

Являются базовыми элементами сложных последовательностных цифровых устройств.

Подготовка к работе

По указанной выше литературе изучить:

– классификацию триггеров;

– условное графическое обозначение и правила работы RS-, D-, JK-, T- и TV-триггеров.

Задания и методические указания к их выполнению

1. В соответствии с временными диаграммами рис. 6 получить прогноз работы ***RS-триггера*** *с прямыми входами для четных номеров вариантов N и с инверсными входами – для нечетных N.*

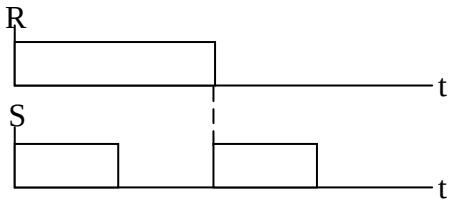


Рис. 6. Сигналы на входах RS-триггера

2. Создать модель как показано на рис. 7 и настроить параметры моделирования (см. п.1,в *работы 2).*

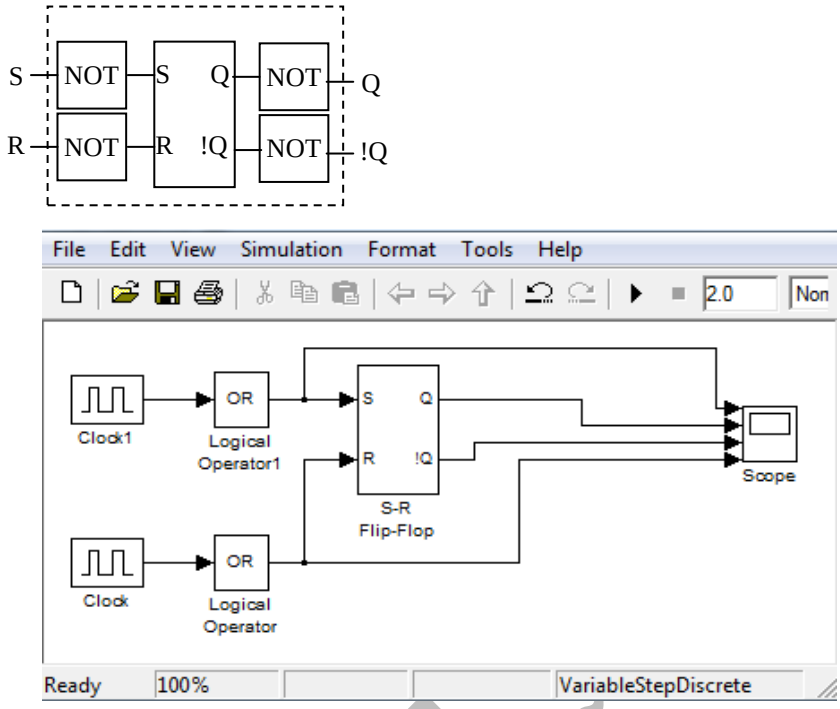


Рис. 7. Окно модели RS-триггера

Триггеры находятся в подразделе Flip Flops раздела Extras библиотеки Simulink Library Browser. В обозначении триггеров символ «!» означает инверсию.

Библиотечный RS-триггер имеет прямые входы. Для имитации инверсных входов необходимо, во-первых, заменить повторитель OR инвертором NOT, что легко сделать с помощью вкладки *Main* окна диалога блока Logical Operator (см. п.2,б работы 2). Данное окно открывается двойным щелчком компьютерной мыши по изображению блока. Во-вторых, по обоим выходам триггера установить инверторы NOT. Иначе при активных входах (R = S = 0) на выходах триггера будут не единицы, а нули, что характерно для RS-триггера с прямыми, но не инверсными входами. Таким образом, RS-триггер с инверсными входами на основе библиотечного триггера будет иметь вид:

Блоки Logical Operator выполняют функцию согласования блоков Clock с информационными входами блока S-R Flip-Flop по типу данных.

Блок Clock (генератор тактовых импульсов) находится в том же подразделе Flip Flops библиотеки. Для реализации временных сдвигов сигналов на входах S и R триггера в модели используется два таких блока. Один из них (Clock) настроен на условный период следования тактов, равный 2, а второй (Clock1) – на условный период, равный 1.

Для визуализации результатов моделирования работы устройства удобно использовать блок Scope (осциллограф), расположенный в подразделе Sinks раздела Simulink библиотеки. Для настройки его параметров на панели инструментов одноименного окна диалога (открывается двойным щелчком компьютерной

мыши по изображению блока) следует «нажать» кнопку *Parameters*. В строке *Number of axes* нового окна диалога *'Scope' parameters* установить требуемое число входов блока. При этом помимо выходов моделируемого устройства следует предусмотреть подключение блока или блоков Clock.

3. Запустить модель на исполнение и сравнить результаты моделирования с прогнозом.

4. Сохранить модель в соответствующей папке под именем RStr<№ вар.>.mdl и сделать вывод относительно «запрещенной» комбинации входных сигналов.

5. В соответствии с табл. 8 и временными диаграммами рис. 8 получить прогноз работы ***D-триггера со статическим управлением.***

Таблица 8. Параметры D-триггера со статическим управлением

Номер варианта N	Активный сигнал	
	D-вход	C-вход
1, 5, 9, …	прямой	прямой
2, 6, 10, …	инверсный	прямой
3, 7, 11, …	прямой	инверсный
4, 8, 12, …	инверсный	инверсный

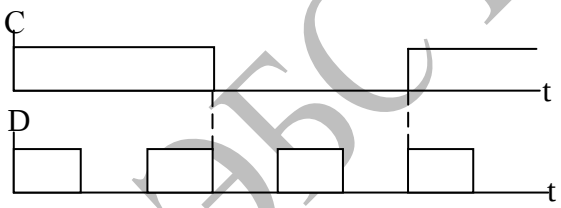


Рис. 8. Сигналы на входах D-триггера со статическим управлением

6. Создать модель как показано на рис. 9 и настроить параметры моделирования.

Библиотечный триггер со статическим управлением имеет прямые входы D и C. Для имитации инверсных входов достаточно использовать инверторы NOT.

Согласно рис. 8 генератор тактов Clock должен быть настроен на условный период следования тактов, равный 3, а Clock1 – на условный период, равный 1.

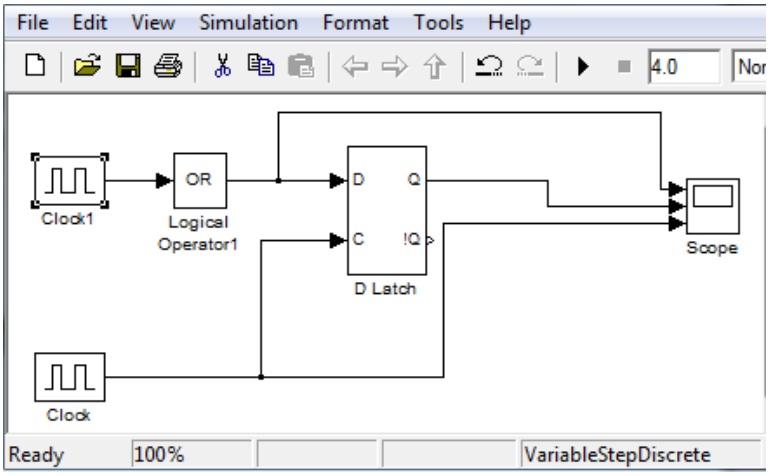


Рис. 9. Окно модели D-триггера со статическим управлением

7. Запустить модель на исполнение и сравнить результаты моделирования с прогнозом.

8. Сохранить модель, под именем DStr<№ вар.>.mdl.

9. В соответствии с табл. 9 и временными диаграммами рис. 10 получить прогноз работы ***D-триггера с динамическим управлением****.*

Таблица 9. Параметры D-триггера с динамическим управлением

Номер варианта N	Активный сигнал	
	D-вход	C-вход
1, 5, 9, …	инверсный	фронт
2, 6, 10, …	прямой	срез
3, 7, 11, …	инверсный	срез
4, 8, 12, …	прямой	фронт

Библиотечный триггер с динамическим управлением имеет прямой D-вход, а активным сигналом по C-входу является фронт тактового импульса. Для имитации противоположных активных сигналов достаточно использовать инверторы NOT.

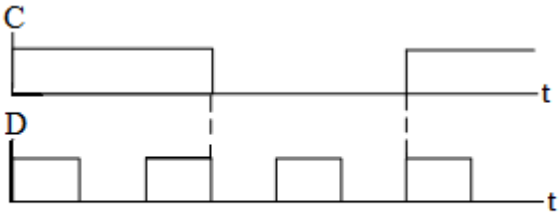
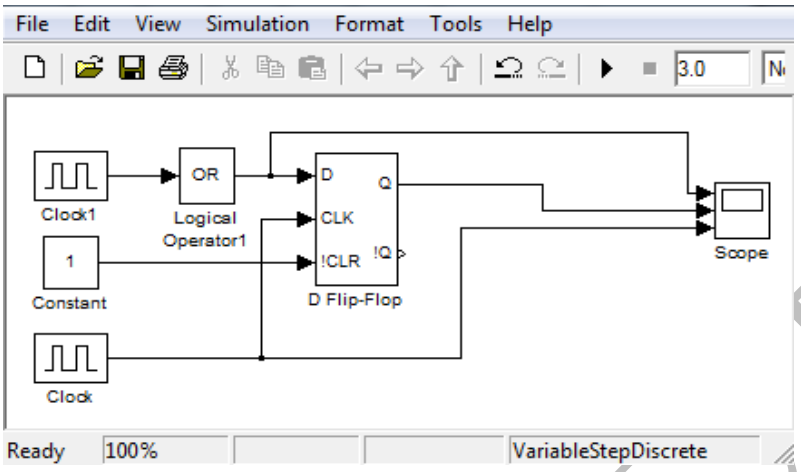


Рис. 10. Сигналы на входах D-триггера с динамическим управлением

10. Создать модель как показано на рис. 11 и настроить параметры моделирования.

Рис. 11. Модель D-триггера с динамическим управлением



Согласно рис. 10 генератор тактов Clock должен быть настроен на условный период следования тактов, равный 1, а Clock1 – на условный период, равный 3.

Поскольку вход CLR (R-вход триггера) в данной работе не используется, на него следует подать пассивный сигнал – 1, которую можно задать блоком Constant, расположенным в подразделе Sources раздела Simulink библиотеки.

11. Запустить модель на исполнение и сравнить результаты моделирования с прогнозом.

12. На основе заданного D-триггера построить T-триггер, в том же окне модели создать и запустить соответствующую модель и сделать вывод относительно соотношения частот на входе и выходе триггера.

13. На основе того же D-триггера построить TV-триггер, в том же окне модели создать и запустить соответствующую модель и сделать вывод относительно сложности реализации, а также типе и месте включения дополнительного логического элемента.

14. Сохранить окно модели под именем DDtr<№ вар.>.mdl.

15. В соответствии с табл. 10 и временными диаграммами рис. 12 получить прогноз работы ***JK-триггера.***

Таблица 10. Параметры JK-триггера

Номер варианта N	Активный сигнал		
	вход J	вход K	C-вход
1, 9, 17,	инверсный	прямой	срез

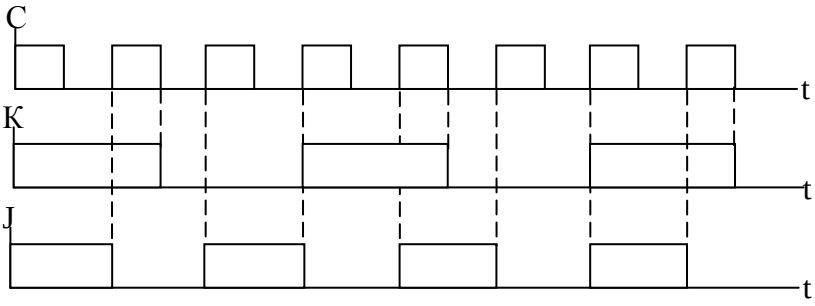
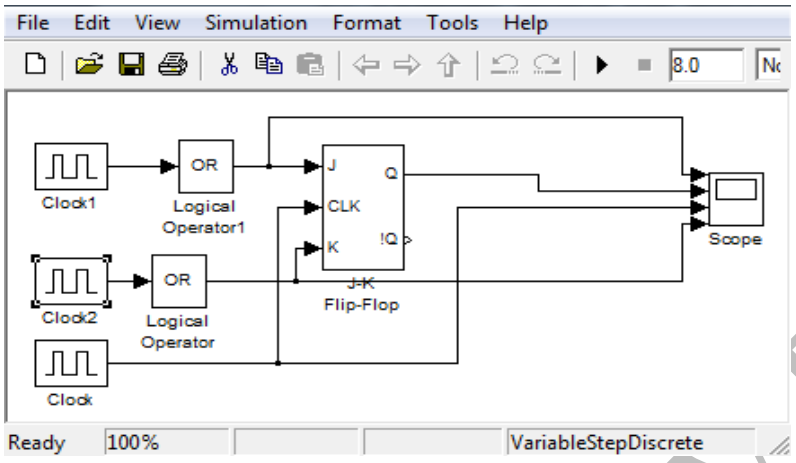


Рис. 12. Сигналы на входах JK-триггера



…			
2, 10, 18, …	прямой	инверсный	фронт
3, 11, 19, …	прямой	инверсный	срез
4, 12, 20, …	инверсный	прямой	фронт
5, 13, 21, …	инверсный	инверсный	срез
6, 14, 22, …	прямой	прямой	фронт
7, 15, 23, …	прямой	прямой	срез
8, 16, 24, …	инверсный	инверсный	фронт

16. Поменять местами сигналы на входах J и K и получить новый прогноз работы заданного триггера.

17. Создать модель как показано на рис. 13 и настроить параметры моделирования.

Библиотечный JK-триггер имеет прямые входы *J* и *K*, а активным сигналом по входу *C* является срез тактового им-

Рис. 13. Модель JK-триггера

пульса. Для имитации противоположных активных сигналов достаточно использовать инверторы NOT.

Согласно рис. 12 генератор тактов Clock должен быть настроен на условный период следования тактов, равный 1, Clock1 – на условный период, равный 2, а Clock2 – на условный период, равный 3.

18. Для каждого из вариантов сигналов на входах J и K запустить модель на исполнение и сравнить результаты моделирования с соответствующими прогнозами.

19. На основе заданного JK-триггера построить T- и TV-триггер, в том же окне модели создать и запустить соответствующие модели и сделать вывод относительно сложности реализации по сравнению с п.п.12 и 13 данной работы.

14. Сохранить окно модели под именем JKtr<№ вар.>.mdl.

5. ПОСЛЕДОВАТЕЛЬНОСТНЫЕ ЦИФРОВЫЕ УСТРОЙСТВА (ПЦУ)

Используются в различных управляющих устройствах, а также для хранения и преобразования информации, представленной в цифровой форме.

Подготовка к работе

По указанной выше литературе изучить:

- модели и способы задания ПЦУ;
- методику синтеза ПЦУ.

Задания и методические указания к их выполнению

1. Из табл. 11 в соответствии с номером варианта N выбрать параметры графа переключений (рис. 14, где t_i – i-й такт цикла работы ПЦУ).

Таблица 11. Параметры графа переключений ПЦУ

N	Состояния ПЦУ				Состояния выхода			
	Q_0	Q_1	Q_2	Q_3	Y_0	Y_1	Y_2	Y_3
1	0	3	4	7	6	4	3	1
2	1	4	5	2	5	3	7	0
3	2	0	6	1	4	7	1	5
4	3	1	7	4	3	0	6	1
5	4	7	2	3	2	5	0	7
6	5	6	3	0	0	6	2	7
7	6	2	1	5	7	1	4	6
8	7	5	0	6	1	2	5	3
9	0	5	4	3	5	0	7	4
10	1	6	3	4	4	5	2	1
11	2	1	0	7	3	6	1	0
12	3	4	5	0	2	7	4	3
13	4	0	2	6	1	2	5	6

14	5	2	7	1	0	1	6	5
15	6	7	1	5	7	4	0	2
16	7	3	6	1	6	3	1	7
17	0	1	6	5	1	4	5	2
18	1	4	3	6	2	3	6	1
19	2	7	0	3	3	1	7	4
20	3	6	1	0	4	7	0	3
21	4	3	7	1	5	6	3	0
22	5	0	2	4	6	2	1	5
23	6	2	5	7	7	0	4	6
24	7	5	4	2	0	5	2	7
25	0	5	2	7	1	6	3	4
26	2	5	6	7	2	5	0	7
27	3	0	6	1	3	4	5	6
28	4	2	1	5	4	5	2	7
29	5	3	2	0	5	6	7	1
30	6	4	3	1	6	4	3	5

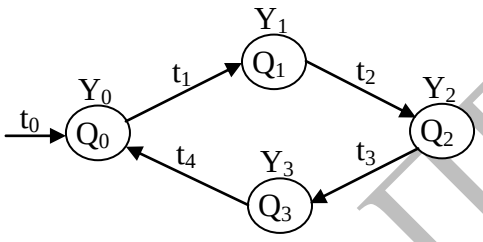


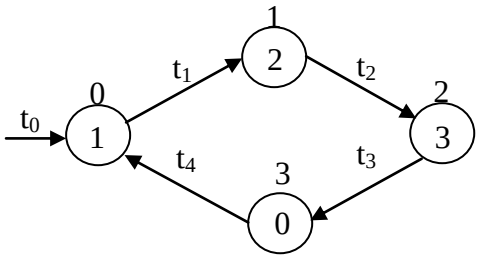
Рис. 14. Граф переключений ПЦУ

2. На основе только анализа автоматной таблицы записать минимальные функции переходов и выходов. При этом в качестве запоминающих элементов ПЦУ использовать D-триггеры с динамическим управлением.

Для примера рассмотрим задачу синтеза ПЦУ, граф переключений которого имеет следующие параметры:

Состояния ПЦУ				Состояния выхода			
Q_0	Q_1	Q_2	Q_3	Y_0	Y_1	Y_2	Y_3
1	2	3	0	0	1	2	3

Соответствующий граф переключений имеет вид:



Как видно, максимальным является третье состояние ПЦУ. Следовательно,

количество запоминающих элементов (триггеров) $N_т = \lceil \mathrm{Log}_2(3) \rceil = 2$.

В качестве модели ПЦУ примем автомат Мура, запоминающими элементами которого, согласно заданию, являются D-триггеры с динамическим управлением. Тогда автоматная таблица, соответствующая графу переключений, будет иметь вид:

№ состояния	Состояние триггеров		Сигналы управления		Состояние выхода	
	q_1	q_0	$a_1 = D_1$	$a_0 = D_0$	y_1	y_0
1	0	1	1	0	0	0
2	1	0	1	1	0	1
3	1	1	0	0	1	0
0	0	0	0	1	1	1

На основании этой таблицы можно сразу записать системы минимальных функций переходов и выходов:

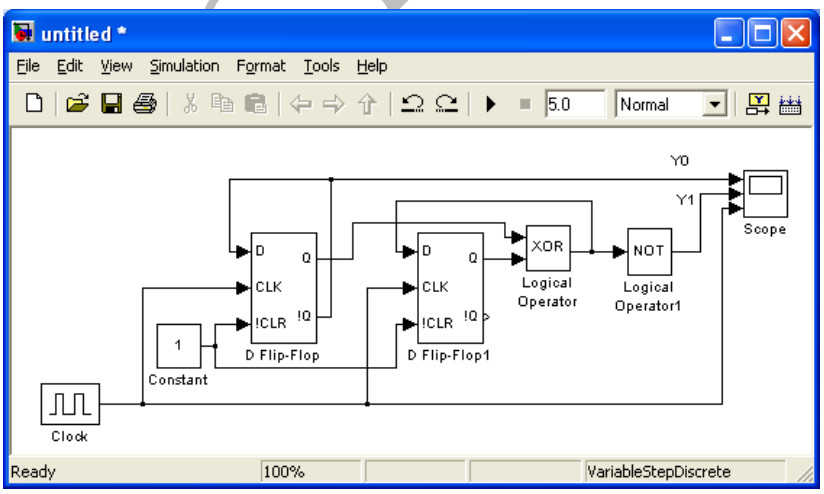
$$\begin{cases} D_0 = \bar{q}_0, \\ D_1 = q_0 \oplus q_1 \end{cases}, \qquad \begin{cases} y_0 = \bar{q}_0 = D_0, \\ y_1 = q_0 \sim q_1 = \bar{D}_1. \end{cases}$$

3. С помощью любого из методов минимизации убедиться в достоверности полученных ФАЛ.

4. Разработать структурную схему устройства.

5. В системе MATLAB создать соответствующую модель устройства и сохранить ее под именем pcu.mdl.

Так, модель ПЦУ рассмотренного примера приведена на рисунке:



6. Запустить модель и сравнить диаграммы на экране «осциллографа» с данными таблицы выходов.

Процесс настройки параметров моделирования приведен в пункте 6 работы 2, а также в предыдущей работе.

Может случиться, что таблица выходов совпадает с «осциллограммами» лишь через некоторое количество тактов. Это вполне соответствует реальной ситуации, поскольку в момент запуска устройство устанавливается в произвольное состояние, что обусловливает переходный процесс.

6. ТИПОВЫЕ ПЦУ

Применяются для подсчета количества тактовых импульсов, уменьшения частоты их следования в заданное число раз, хранения и/или преобразования двоичных наборов путем поразрядной инверсии либо сдвига.

Подготовка к работе

По указанной выше литературе изучить:

– типы базовых триггеров асинхронных и синхронных счетчиков, буферных регистров и регистров сдвига;

– способы реализации базовых триггеров счетчиков на D- и JK-триггерах;

– принципы построения асинхронных и синхронных счетчиков;

– принципы и особенности реализации счетчиков с произвольным модулем счета и начальным состоянием;

– принципы построения регистров.

Задания и методические указания к их выполнению

1. На D-триггерах разработать структурную схему асинхронного ***счетчика*** *с параметрами, заданными в табл. 12 – 14 согласно номера варианта N. Направление счета – прямое для четных N и обратное для нечетных N.*

Таблица 12. Параметры триггеров счетчика

N	Активный сигнал		Вход D	N	Активный сигнал		Вход D
	C	R			C	R	
1,2,17,18	⌜	0	инверсный	3,4,19,21	⌜	0	прямой
9,10,25,26		1		11,12,27,28		1	
5,6,21,22	⌞	0		7,8,23,24	⌞	0	
13,14,29		1		15,16,30		1	

Таблица 13. Модуль счета $К_с$

N	$К_с$	N	$К_с$	N	$К_с$
1,3,5,7,9	13	11,13,15,17,19	9	21,23,25,27,29	11
2,4,6,8,10	10	12,14,16,18,20	7	22,24,26,28,30	12

Таблица 14. Начальное состояние $Q_н$

N	$Q_н$	N	$Q_н$	N	$Q_н$	N	$Q_н$	N	$Q_н$
1,2	1	7,8	4	13, 14	7	19, 20	10	25, 26	13
3,4	2	9,10	5	15, 16	8	21, 22	11	27, 28	14
5,6	3	11, 12	6	17, 18	9	23, 24	12	29, 30	0

Триггер D Flip-Flop Simulink имеет один вход принудительной установки,

причем в нулевое состояние. Это обстоятельство порождает две особенности в построении счетчиков:

– при необходимости принудительной установки i-го триггера в состояние 1 в качестве соответствующего выхода счетчика следует использовать инверсный выход триггера;

– в асинхронном счетчике, отвечающем требованиям задания, соединять этот i-й триггер с последующим необходимо по схеме, соответствующей противоположному относительно заданному направлению счета.

Например, в суммирующем счетчике с $К_с = 6$ и $Q_н = 2$ выходом среднего разряда счетчика будет инверсный выход соответствующего триггера при следующих межтриггерных связях: $Q_0 \rightarrow C_1$ и $Q_1 \rightarrow C_2$ или в обозначениях Simulink $!Q_0 \rightarrow CLK_1$ и $Q_1 \rightarrow CLK_2$.

Другая особенность – MATLAB «отказывается» моделировать счетчики с произвольным модулем счета, поскольку длительность сигнала на выходе СУНС значительно меньше длинны такта. Выходом из этой ситуации является просто наблюдение сигнала на выходе СУНС без соединения его с входами R(!CLR) триггеров.

Так, модель счетчика примера, учитывающая все отмеченные особенности, приведена на рис. 15.

На рис. 16 приведены временные диаграммы работы счетчика, полученные в результате моделирования. Видно, что активный сигнал на выходе СУНС (логический элемент OR модели) образуется сразу по окончании рабочего цикла счетчика ($Q_к = 2+6-1 = 7$ или 111 в двоичной системе счисления). Следовательно, реализация счетчика удовлетворяет всем требованиям.

Q_0

$!Q_1$

Q_2

OR

Такты

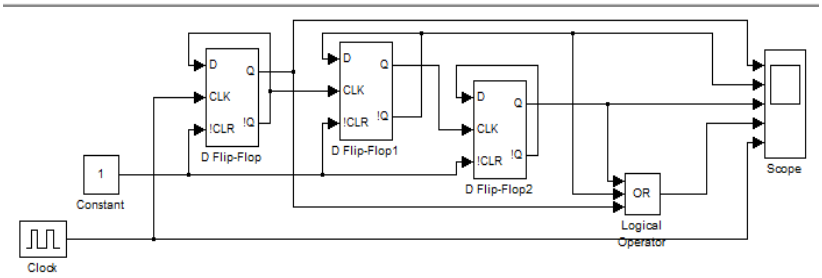


Рис. 15. Модель суммирующего счетчика с $К_с = 6$ и $Q_н = 2$

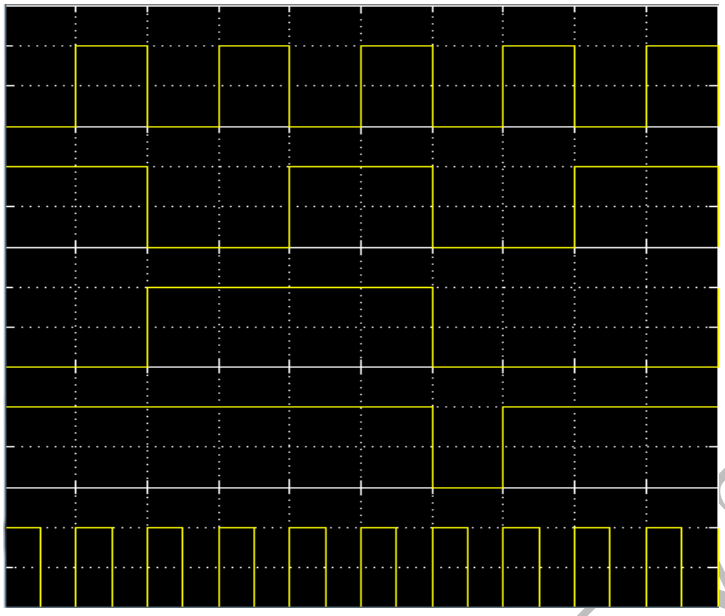


Рис. 16. Результаты моделирования счетчика примера

2. Построить временные диаграммы работы счетчика.

3. В системе MATLAB создать модель разработанного счетчика и сохранить ее под именем ас<№ вар.>.mdl.

4. Запустить модель на исполнение и сравнить диаграммы окна Scope с диаграммами п.2 данной работы.

5. Разработать структурную схему 5-разрядного последовательно-параллельного ***регистра сдвига*** *влево для четных N и вправо для нечетных N на базовых триггерах. При этом для нечетных пар вариантов (1 и 2, 5 и 6, . . .) информационный вход триггеров – прямой, а для четных пар (3 и 4, 7 и 8, ...) – инверсный.*

6. Построить временные диаграммы по вводу в регистр N-го (для вариантов с 1 по 14) и (N+1)-го (для вариантов с 15 по 29) двоичного набора.

7. В системе MATLAB создать модель разработанного регистра и сохранить ее под именем rs<№ вар.>.mdl.

Для чтения вводимого двоичного набора используется блок From workspace (см. стр. 13), причем с информационным входом регистра он должен соединяться через блок Logical Operator (см. стр. 14). Это промежуточное звено необходимо для согласования блоков From workspace и D Flip-Flop по типу данных.

При настройке параметров блока From workspace в списке *Form output after-final data value by* (по окончании данных формировать на выходе значение) со-

ответствующего окна диалога выбрать *Holding final value* (по последнему значению).

8. Запустить модель и сравнить диаграммы окна Scope с диаграммами п.6 данной работы.