МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

Кафедра № 705Б

Бортовая автоматика беспилотных космических
и атмосферных летательных аппаратов

Б.Н. ПОПОВ, Е.Э. ЦАЛКОВА, Д.Н. ХАНИН

ПРОЕКТИРОВАНИЕ ЦИФРОВЫХ И ЦИФРОАНАЛОГОВЫХ УСТРОЙСТВ БОРТОВОЙ АВТОМАТИКИ. ЛАБОРАТОРНЫЙ ПРАКТИКУМ

Под редакцией Б.Н. Попова

Утверждено
на заседании редсовета
факультета № 7
28.01.2016 г.
(Протокол № 1)

Москва
Издательство МАИ
2016

УДК 621.38.049.77
ББК 34.447
П 58

П 58 Попов Б.Н., Цалкова Е.Э., Ханин Д.Н.
Проектирование цифровых и цифроаналоговых устройств бортовой автоматики. Лабораторный практикум / Под ред. Б.Н. Попова. – М.: Изд-во МАИ, 2016. – 80 с.: ил.

Учебное пособие содержит описание четырех лабораторных работ. Кратко изложен необходимый теоретический материал. Приведены основные сведения по языку описания аппаратуры VHDL (Very high speed integrated circuits Hardware Description Language), цифроаналоговым и аналогово-цифровым преобразователем, а также цифровым устройствам управления электродвигателями. Приведены примеры текстов программ на языке VHDL.

Пособие предназначено для студентов вузов, обучающихся по направлению «Системы управления летательными аппаратами», а также может быть полезным для молодых специалистов данного профиля деятельности и аспирантов.

Рецензенты:
кафедра «Системы автоматического управления» МГТУ им. Н.Э. Баумана (зав. кафедрой д-р техн. наук, академик РАН Микрин Е.А.);
канд. техн. наук Кербер О.Б.

ISBN 978-5-4316-0317-4

СОДЕРЖАНИЕ

ПРЕДИСЛОВИЕ

Настоящее учебное пособие соответствует ряду разделов учебного курса «Микропроцессорная техника в приборах, системах и комплексах» в рамках подготовки дипломированных специалистов, обучающихся по специальности 240506 «Системы управления летательными аппаратами». Учебное пособие, кроме помощи в самостоятельном изучении материала, предназначено для курсового и дипломного проектирования, а также может быть полезным для молодых специалистов данного профиля деятельности и аспирантов.

Авторы благодарят сотрудников МОКБ «Марс» Н.А. Вербицкого и О.В. Абросимова за помощь в подготовке и описании лабораторных работ № 3, 4.

Авторы также благодарят сотрудников МОКБ «Марс» Е.Э. Качалову и Т.В. Кособокову за подготовку рукописи к печати.

Общие указания к лабораторным работам

При проведении лабораторных работ используется персональный компьютер с установленной средой разработки Quartus II. Аппаратная часть представлена демонстрационной платой DE2 фирмы Altera с кабелем питания и кабелем для программирования (USB Blaster) [8, стр. 48–50]. Для лабораторных работ № 2–4 используются дополнительно цифровые вольтметры, цифровой осциллограф, макетные платы и ряд навесных компонентов, устанавливаемых на макетную плату, цифровой фототахометр.

Отчет по лабораторной работе должен содержать: титульный лист, содержание, цель работы, задание, теоретическую и практическую части, схему эксперимента, выводы и ссылки на использованную литературу.

Процесс сдачи лабораторной работы заключается в предъявлении отчета и ответах на контрольные вопросы, которые приведены в конце каждой работы.

Лабораторная работа 1.

Разработка цифровых устройств на основе ПЛИС с использованием языка VHDL

Цель работы – приобретение навыков проектирования комбинационных и последовательностных логических схем на основе программируемых логических интегральных схем (ПЛИС) с использованием языка описания аппаратуры VHDL и текстового ввода проекта. Закрепление навыков использования программной среды Quartus II.

Теоретическая часть

Язык описания аппаратуры (Hardware Description Language) является набором формальных записей, которые могут быть использованы на всех этапах разработки цифровых электронных систем.

Это возможно вследствие того, что язык легко воспринимается как машиной, так и человеком. Он может использоваться на этапах проектирования, верификации, синтеза и тестирования аппаратуры так же, как и для передачи данных о проекте, модификации и сопровождения.

Язык VHDL используется во многих системах автоматизированного проектирования (САПР) для моделирования цифровых систем, проектирования различных цифровых устройств на основе ПЛИС, базовых матричных кристаллов, заказных интегральных микросхем.

Язык VHDL является близким по синтаксису и семантике к современным языкам программирования типа Паскаль, C++ и др.

Основные операторы VHDL

С помощью операторов описываются алгоритмы, определяющие функционирование схемы. Они могут находиться в теле функции, процедуры или процесса. Рассмотрим основные операторы.

- **`Wait until condition`**

Этот оператор приостанавливает выполнение процесса до момента выполнения условия (condition).

- **`Signal <=`** *expression*

Оператор назначения устанавливает сигналу значение, равное выражению справа.

- **`Variable :=`** *expression*

Оператор присваивания устанавливает значение переменной равным выражению справа.

- **`Procedure_`***name (parameter {, parameter(s)})*

Оператор вызова процедуры состоит из имени процедуры и списка фактических параметров.

- **`If`**

\оператор `if\::=if`\условие 1`\then`
`{`\последовательный оператор 1`\}`
[`{elsif`\условие 2`\then`
`{`\последовательный оператор 2`\}]`
[`else`
`{`\последовательный оператор 3`\}]`
`end if`;

Оператор `if` используется для ветвления алгоритма по различным условиям.

- **`Case`**

```
\оператор case\ ::= case \простое выражение\ is
when \альтернативы\ => {\последовательный оператор\}
{when \альтернативы\=> {\последовательный оператор\}
when others =>{\последовательный оператор\}}
end case;
```

Оператор `case` подобно оператору `if` задает ветвление алгоритма. Когда значение выражения встречается в одном из списков значений, выполняется соответствующая последовательность операторов. Если значение выражения не присутствует ни в одном из списков, то выполняется список операторов, соответствующий ветви when others.

- **`Loop`**

```
\оператор цикла\ ::= [\метка\:] [\схема итерации\] loop
{\последовательный оператор\}
{next [\метка\] [when \условие\];}
{exit [\метка \] [when \условие\];}
end loop [\метка\];
\схема итерации\ ::= while \условие\
\for\переменная цикла\in\диапазон\
```

Оператор цикла позволяет многократно выполнять последовательность операторов. Диапазон значений задается в виде `value1 to value2`. Переменная цикла последовательно принимает значения из заданного диапазона. Количество итераций равно количеству значений в диапазоне.

- **`Return expression`**

Этот оператор возвращает значение из функции.

- **`Null`**

Пустой оператор, не выполняет никаких действий.

Операторы VHDL делятся на последовательные и параллельные. Последовательно выполняемые (*последовательные*) операторы

VHDL могут использоваться в описании процессов, процедур и функций. К этим операторам относятся:

– оператор присваивания переменной (**:=**);

– последовательный оператор назначения сигналу (**<=**);

– последовательный оператор утверждения (**assert**);

– условный оператор (**if**);

– оператор цикла (**case**);

– пустой оператор (**null**);

– операторы возврата процедуры (**return**);

– оператор последовательного вызова процедуры.

Язык поддерживает концепции пакетного и структурного программирования. Сложные операторы заключены в операторные скобки: `if – end if`; `process – end process`; `case – end case`; `loop – end loop` и т.д.

Различаются локальные и глобальные переменные. Область «видимости» локальных переменных ограничена пределами блока (процессного, процедурного, операторного блока, оператора описания архитектуры).

Фрагменты описаний, которые могут независимо анализироваться компилятором и при отсутствии ошибок помещаться в библиотеку проекта (рабочую библиотеку Work), называются проектными пакетами design unit. Такими пакетами могут быть: объявление интерфейса объекта проекта (entity), объявление архитектуры (architecture), объявление конфигурации (configuration), объявление интерфейса пакета (package) и объявление тела пакета (package body).

Модули проекта, в свою очередь, можно разбить на две категории: первичные и вторичные. К первичным модулям относятся объявления пакета, объекта проекта, конфигурации. К вторичным – объявление архитектуры, тела пакета. Один или несколько модулей проекта могут быть помещены в один файл, называемый файлом проекта (design file).

Каждый проанализированный модуль проекта помещается в библиотеку проекта (design library) и становится библиотечным модулем (library unit).

Каждая библиотека проекта в языке VHDL имеет логическое имя (идентификатор). По отношению к сеансу работы с VHDL-системой существует два класса рабочих библиотек проекта: рабочие библиотеки и библиотеки ресурсов.

Рабочая библиотека – это библиотека, содержащая модули, ссылка на которые имеется в текущем проекте. В каждый конкретный момент времени пользователь работает с одной рабочей библиотекой и произвольным числом библиотек ресурсов.

Проблемно-ориентированная компонента позволяет описывать цифровые системы в привычных для разработчика понятиях и терминах. Сюда можно отнести:

• понятие модельного времени (now);

• данные типа time, позволяющие указывать время задержки в физических элементах;

• данные вида signal, значения которых изменяются не мгновенно, как у переменных, а с указанной задержкой, а также специальные операции и функции над ними;

• средства объявления объектов (entity) и их архитектур (architecture).

Параллельные операторы VHDL включают:

– оператор процесса (**process**);

– оператор блока (**block**);

– параллельный оператор назначения сигналу (**<=**);

– оператор условного назначения сигналу (**when**);

– оператор селективного назначения сигналу (**select**);

– параллельный оператор утверждения (**assert**);

– параллельный оператор вызова процедуры;

– оператор конкретизации компоненты (**port map**);

– оператор генерации конкретизации (**generate**).

Как видно из этого перечня, последовательные и параллельные операции назначения, вызова процедуры и утверждения различаются контекстно.

Базовым элементом описания систем на языке VHDL является блок. Блок содержит раздел описаний данных и раздел параллельно исполняемых операторов. В рамках описания архитектуры могут использоваться внутренние, вложенные блоки. Наряду со всеми преимуществами блочной структуры программы и ее соответствия естественному иерархическому представлению структуры проекта операторы блока языка VHDL позволяют устанавливать условия охраны (запреты) входа в блок. Только при истинности значения охранного выражения управление передается в блок и инициируется выполнение операторов его тела.

Алфавит языка

Как и любой другой язык программирования, VHDL имеет свой алфавит – набор символов, разрешенных к использованию и воспринимаемых компилятором. В алфавит языка входят:

- латинские строчные и прописные буквы – A, a; B, b; ..., Z, z;
- арабские цифры от 0 до 9;
- символ подчеркивания «_» (код ASCII номер 95).

Из символов, перечисленных выше (и только из них!), могут конструироваться идентификаторы в программе. Написание идентификаторов должно подчиняться следующим правилам:

- идентификатор не может быть зарезервированным словом языка;
- идентификатор должен начинаться с буквы;
- идентификатор не может заканчиваться символом подчеркивания «_»;
- идентификатор не может содержать два и более идущих подряд символов подчеркивания «_».

Примеры корректных идентификаторов: `cont, clock2, full_add`.

Примеры некорректных идентификаторов: `1clock`, `adder_`, `add__sub`, `entity`.

Зарезервированные слова в языке VHDL имеют специальное назначение и приведены ниже.

ABS	ELSE	MOD	RETURN
ACCESS	ELSEIF	NAND	SELECT
AFTER	END	NEW	SEVERITY
ALIAS	ENTITY	NEXT	SIGNAL
ALL	EXIT	NOR	SUBTYPE
AND	FILE	NOT	THEN
ARCHITECTURE	FOR	NULL	TO
ARRAY	FUNCTION	OF	TRANSPORT
ASSERT	GENERATE	ON	TYPE
ATTRIBUTE	GENERIC	OPEN	UNITS
BEGIN	GUARDED	OR	UNTIL
BLOCK	IF	OTHERS	USE
BODY	IMPORT	OUT	VARIABLE
BUFFER	IN	PACKAGE	WAIT
BUS	INITIALIZE	PORT	WHEN
CASE	INOUT	PROCEDURE	WHILE
COMPONENT	IS	PROCESS	WITH
CONFIGURATION	LABEL	RANGE	XOR
CONSTANT	LIBRARY	RECORD	DISCONNECT
LINKAGE	REGISTER	DOWNTO	LOOP
REM	MAP	REPORT	

Следует отметить, что прописные и строчные буквы не различаются, идентификаторы `nik` и NIK эквивалентны.

В алфавит языка также входят: символ «пробел» (код ASCII номер 32), символ табуляции (код ASCII номер 9), символ новой строки

(коды ASCII номер 10 и 13). Данные символы являются разделителями слов в конструкциях языка VHDL. Количество разделителей не имеет значения. Таким образом, следующие выражения для компилятора эквивалентны:

```
- Count:=2+2;
- Count := 2 + 2 ;
- Count := 2
  +
  2;
```

Специальные символы, участвующие в построении конструкций языка, тоже входят в его алфавит:

Символ	Имя	Символ	Имя
"	кавычки	.	точка
#	диез	/	слэш
&	амперсант	:	двоеточие
'	апостроф	;	точка с запятой
(	левая круглая скобка	<	меньше
)	правая круглая скобка	=	равно
*	звездочка, умножить	>	больше
+	плюс	_	нижнее подчеркивание
,	запятая	|	вертикальная черта
-	минус		

При необходимости могут применяться составные символы: **<=** **>=** **=>** **:=** **/=** , которые воспринимаются компилятором как один символ. Разделители между элементами составных символов недопустимы.

Комментарии

Признаком комментария являются два символа тире «--», идущие подряд. Компилятор игнорирует текст, начиная с символов «--» до конца строки, комментарий может включать в себя символы, не входящие в алфавит языка (в частности, русские буквы).

Числа

В стандарте языка VHDL определены числа как целого, так и вещественного типов. Однако средства синтеза ПЛИС допускают применение только целых чисел. Целое число в языке VHDL может быть представлено в одной из четырех систем счисления: двоичной, восьмеричной, десятичной и шестнадцатеричной.

Символы

Запись символа представляет собой собственно символ, заключенный в одиночные кавычки. Например: `'A', '*', ' '`.

В средствах синтеза ПЛИС область применения символов ограничена использованием их в качестве элементов перечислимого типа.

Строки

Строки представляют собой набор символов, заключенных в двойные кавычки. Например: `"A string"`.

Типы переменных

Подобно высокоуровневым языкам программирования, VHDL является языком со строгой типизацией. Каждый тип данных в VHDL имеет определенный набор принимаемых значений и набор допустимых операций.

В языке VHDL используются следующие простые типы переменных:

• **BOOLEAN** (логический) – объекты данного типа могут принимать только два значения: FALSE (ложь) и TRUE (истина), причем FALSE эквивалентно 0, а TRUE эквивалентно 1.

• **INTEGER** (целый) – значения данного типа представляют собой 32-разрядные числа со знаком. Объекты типа INTEGER могут содержать значения из диапазона от $-(2^{31}-1)$ до $2^{31}-1$.

• **BIT** (битовый) – представляет один логический бит. Объекты данного типа могут содержать только значения «0» и «1».

• **STD_LOGIC** (битовый) – представляет один бит данных.

Объекты данного типа могут принимать девять состояний:

«U» – неинициализированное;

«X» – сильная неопределенность;

«W» – слабая неопределенность;

«0» – сильный 0;

«L» – слабый 0;

«1» – сильная 1;

«H» – слабая 1;

«Z» – высокий импеданс;

«–» – не подключен.

Данный тип определен стандартом IEEE 1164 для замены типа BIT.

• **STD_ULOGIC** (битовый) – представляет один бит данных.

Объекты данного типа могут принимать девять состояний. Данный тип определен стандартом IEEE 1164[1] для замены типа BIT.

• **ENUMERATED** (перечислимый) – используется для задания пользовательских типов.

• **SEVERITY_LEVEL** – перечислимый тип, используется только в операторе ASSERT.

• **CHARACTER** – символьный тип.

Примечание. В действительности тип STD_ULOGIC является базовым для типа STD_LOGIC, объекты обоих типов могут принимать одно и то же множество значений и имеют одинаковый набор допустимых операций. Единственное различие между типами заключается в том, что для типа STD_ULOGIC не определена функция разрешения (resolving function). В языке VHDL функция разрешения используется для определения значения сигнала, имеющего несколько источников.

Тип BOOLEAN

Все три типа объектов в VHDL (константы, переменные и сигналы) могут иметь тип BOOLEAN.

[1] Институт инженеров электротехники и электроники (Institute of Electrical and Electronics Engineers – IEEE) – разработчик многочисленных стандартов по радиоэлектронике и электротехнике.

Операторы отношения. Значения типа BOOLEAN могут участвовать в выражениях. Операторы отношения (=, /=, <, <=, >, >=) определены для операндов типа BOOLEAN и одномерных массивов, содержащих элементы типа BOOLEAN. Для всех перечислимых типов операции сравнения над одномерными массивами типа BOOLEAN производятся поэлементно, начиная с крайнего левого элемента. Перечислимый тип – это такой тип данных, при котором количество всех возможных значений конечно.

Логические операторы. Для операндов типа BOOLEAN и одномерных массивов, содержащих элементы типа BOOLEAN, определены все логические операции (AND, OR, NAND, NOR, XOR и NOT). Тип и размер операндов должны быть одинаковыми. Тип и размер результата такой же, как тип и размер операндов.

Оператор конкатенации. Оператор конкатенации также определен для операндов типа BOOLEAN и одномерных массивов, содержащих элементы типа BOOLEAN. Результат выражения представляет собой одномерный массив, содержащий элементы типа BOOLEAN. Размер массива равен сумме размеров операндов.

Тип INTEGER

Стандарт VHDL определяет тип INTEGER для использования в арифметических выражениях. По умолчанию объекты типа INTEGER имеют размерность 32 бита и представляют целое число в интервале от $-(2^{31}-1)$ до $(2^{31}-1)$, т.е. от –2147483647 до 2147483647. Стандарт языка позволяет также объявлять объекты типа INTEGER, имеющие размер меньше 32 бита, используя ключевое слово RANGE, ограничивающее диапазон возможных значений:

```
SIGNAL X : INTEGER RANGE -127 TO 127
```

Данная конструкция определяет `X` как 8-битовое число.

Можно определить целый тип, используя следующую конструкцию:

`TYPE` имя_типа `IS RANGE` диапазон_индексов.

Диапазон индексов определяется следующим образом:

`m TO n` или `n DOWNTO m`,

где m, n – целочисленные константы, причем m< n .

Операторы отношения. Операторы отношения (=, /= , <, >, <=, >=) определены для операндов типа INTEGER и одномерных массивов, содержащих элементы типа INTEGER. Результат выражения имеет тип BOOLEAN.

Арифметические операторы. Операторы +, – , ABS допустимы в следующих случаях:

• если оба оператора являются константами (CONSTANT);

• если второй операнд является константой и его значение равно 2^n, где $n = 0, 1, 2, 3$.

Применение операторов *, /, MOD, REM недопустимо, если оба операнда являются сигналами (SIGNAL) или переменными (VARIABLE).

Оператор возведения в степень (**), как правило, не поддерживается средствами синтеза ПЛИС.

Тип BIT

Объект данного типа может принимать значение «0» (лог. 0) или «1» (лог. 1). Стандартом IEEE 1164 определена замена типа BIT на более гибкий тип STD_LOGIC, поэтому использование типа BIT в новых разработках не рекомендуется.

Операторы отношения. Значения типа BIT могут участвовать в выражениях. Операторы отношения (=, /= , <, >, <=, >=) определены для операндов типа BIT и одномерных массивов, содержащих элементы типа BIT. Результат выражения имеет тип BOOLEAN. Это справедливо для всех перечислимых типов. Операции сравнения над одномерными массивами типа BIT производятся поэлементно, начиная с крайнего левого элемента.

Логические операторы. Для операторов типа BIT и одномерных массивов, содержащих элементы типа BIT, определены все логические операции (AND, OR, NAND, NOR, XOR, NOT). Тип и размер операндов должны быть одинаковыми. Тип и размер результата такой же, как тип и размер операндов.

Оператор конкатенации. Оператор конкатенации также определен для операндов типа BIT и одномерных массивов, содержащих элементы типа BIT. Результат выражения представляет собой одномерный массив, содержащий элементы типа BIT. Размер массива равен сумме размеров операндов.

Тип STD LOGIC

Тип STD_LOGIC является перечислимым типом. Объекты типа STD_LOGIC могут принимать девять значений: «0», «1», «Z», «–», «L», «H», «U», «X», «W». Для синтеза логических схем используются только первые четыре:

«0» – логический 0;
«1» – логическая 1;
«Z» – третье состояние;
«–» – не подключен.

Операторы отношения. Значения типа STD_LOGIC могут участвовать в выражениях. Операторы отношения (=, /=, <, >, <=, >=) определены для операндов типа STD_LOGIC и одномерных массивов, содержащих элементы типа STD_LOGIC. Результат выражения имеет тип BOOLEAN. Как и для всех перечислимых типов, операции сравнения над одномерными массивами типа STD_LOGIC производятся поэлементно, начиная с крайнего левого элемента.

Логические операторы. Для операндов типа STD_LOGIC и одномерных массивов, содержащих элементы типа STD_LOGIC, определены все логические операции (AND, OR, NAND, NOR, XOR, NOT). Тип и размер операндов должны быть одинаковыми. Тип и размер результата такие же, как тип и размер операндов.

Оператор конкатенации. Оператор конкатенации также определен для операндов типа STD_LOGIC и одномерных массивов, содержащих элементы типа STD_LOGIC. Результат выражения представляет собой одномерный массив, содержащий элементы типа STD_LOGIC. Размер массива равен сумме размеров операндов.

Другие операторы. Другие операции над операндами типа STD_LOGIC не определены.

Тип ENUMERATED

Строго говоря, все описанные выше типы данных являются перечислимыми. Применение перечислимых типов данных преследует две цели:

1) улучшение смысловой читаемости программы;
2) более четкий и простой визуальный контроль значений.

Наиболее часто перечислимый тип используется для обозначения состояний конечных автоматов. Перечислимый тип данных объявляется путем перечисления названий элементов-значений. Объекты, тип которых объявлен как перечислимый, могут содержать только те значения, которые указаны при перечислении. Элементы перечислимого типа должны быть идентификаторами или символами, которые, в свою очередь, должны быть уникальными в пределах одного типа. Повторное использование названий элементов в других перечислимых типах разрешается.

Объявление перечислимого типа имеет вид

```
TYPE имя_типа IS (название_элемента);
```

Операторы отношения. Значения определенных пользователем перечислимых типов могут участвовать в выражениях. Операторы отношения (=, /, <, <=, >, >=) определены как для перечислимых типов, так и для одномерных массивов, содержащих элементы этих типов.

Результат выражения имеет тип BOOLEAN.

Оператор конкатенации. Оператор конкатенации определен для операндов, имеющих перечислимый тип, и одномерных массивов, содержащих элементы перечислимого типа. Оба операнда должны быть одного типа. Результат выражения представляет собой одномерный массив, тип элементов которого равен типу операндов; размер массива равен сумме размеров операндов.

Другие операторы. К операндам перечисленных типов применим оператор указания типа. Данный оператор используется для уточнения типа объекта в случае если одно и то же название элемента используется различными типами.

Тип CHARACTER

Перечислимый тип. Значением объекта данного типа может быть любой символ из набора ASCII (128 первых символов).

Библиотеки

Для облегчения процесса проектирования описание объекта моделирования и различные варианты описания его архитектуры размещают в библиотеках. Они организованы как отдельные файлы. В библиотеках, как правило, размещают отработанные описания объектов, которые могут использоваться одним или несколькими пользователями в рамках одного или нескольких проектов. В библиотеках могут размещаться описания констант, переменных, типов, процедур, функций. Типовой синтаксис имеет вид

```
library library_name {,...},
```

где library_name – имя библиотеки.

Для того чтобы в тексте каждый раз не указывать имя библиотеки часто используемого объекта, можно применять условную ссылку:

```
use library_name. (identifier|all),
```

где identifier – имя объекта из библиотеки library_name. В том случае, если используются все объекты библиотеки, указывают опцию all.

При проектировании цифровых устройств широко применяются библиотеки IEEE (Institute of Electrical and Electronics Engineers) – разработчика многочисленных стандартов по радиоэлектронике и электротехнике.

Структура программы на языке VHDL

Типовая структура программы на языке VHDL состоит из декларативной части и описания архитектуры. В декларативной части описываются используемые библиотеки и связи объекта с внешним миром – входы и выходы объекта. В описании архитектуры определяется функция разрабатываемого устройства – формирование выходных сигналов на основании входных сигналов и внутреннего состояния объекта.

```
library ieee; -- описание используемых библиотек
use ieee.std_logic_1164.all;
```

```
……
entity program is -- назначение выводов
port (
a : in std_logic;
b : out std_logic;
x, y : in std_logic_vector(10 downto 0));
……
end program;
architecture behavior of program is
 signal sum : std_logic_vector(10 downto 0); -- описание глобальных переменных (сигналов)
……
begin
……
общий текст программы
……
process(a,x,y) -- отдельный процесс в программе
variable i : natural; -- описание локальных переменных процесса
……
begin
……
текст программы процесса
……
end process;
end behavior;
```

Заметим, что наличие структурного элемента `process – end process` в программе на языке VHDL не является обязательным и для простых программ, реализующих логические устройства комбинационного типа, отсутствует.

Практическая часть

Часть I

1. Исходя из задания, составить таблицу истинности четырехразрядного комбинационного устройства (см. стр. 65–66 [8]).

2. Минимизировать заданную функцию, используя карты Карно.

3. С помощью текстового редактора программы Quartus II и описанного в [8] маршрута проектирования создать проект для разрабатываемого устройства комбинационного типа. Для отработки проекта на плате предполагается использовать тумблеры SW0–SW3 и светодиодный индикатор LEDR0.

4. Доказать работоспособность схемы с помощью моделирования по временным диаграммам.

5. Прошить проект в ПЛИС платы DE2 и, переключая тумблеры, проверить его работу на соответствие таблице истинности.

Часть II

1. Исходя из задания, написать программу, реализующую четырехразрядный регистр.

2. Промоделировать работу регистра с получением временных диаграмм.

3. Прошить проект в ПЛИС и провести проверку его работоспособности.

Часть III

1. Исходя из задания, написать программу, реализующую счетчик с произвольным модулем счета.

2. Промоделировать работу счетчика с получением временных диаграмм.

3. Прошить проект в ПЛИС и провести проверку его работоспособности.

Варианты заданий (частей II–III) приведены в табл. 1.

Таблица 1

№ варианта	Задание
1	4-разрядный регистр с параллельной записью и считыванием; суммирующий счетчик по модулю 12
2	4-разрядный регистр с последовательной записью и параллельным считыванием; суммирующий счетчик по модулю 13
3	4-разрядный регистр с параллельной записью и последовательным считыванием; вычитающий счетчик по модулю 14
4	4-разрядный регистр с параллельной записью и последовательным считыванием; суммирующий счетчик по модулю 19
5	4-разрядный регистр с параллельной записью и считыванием; вычитающий счетчик по модулю 20
6	4-разрядный регистр с последовательной записью и последовательным считыванием; вычитающий счетчик по модулю 21
7	4-разрядный регистр с последовательной записью и параллельным считыванием; вычитающий счетчик по модулю 24
8	4-разрядный регистр с последовательной записью и последовательным считыванием; вычитающий счетчик по модулю 25
9	4-разрядный регистр с последовательной записью и параллельным считыванием; суммирующий счетчик по модулю 26
10	4-разрядный регистр с последовательной записью и последовательным считыванием; вычитающий счетчик по модулю 29
11	4-разрядный регистр с параллельной записью и последовательным считыванием; вычитающий счетчик по модулю 17
12	4-разрядный регистр с параллельной записью и считыванием; вычитающий счетчик по модулю 28
13	4-разрядный регистр с параллельной записью и последовательным считыванием; суммирующий счетчик по модулю 23
14	4-разрядный регистр с последовательной записью и последовательным считыванием; вычитающий счетчик по модулю 22

Окончание таблицы 1

15	4-разрядный регистр с параллельной записью и последовательным считыванием; вычитающий счетчик по модулю 16
16	4-разрядный регистр с параллельной записью и считыванием; суммирующий счетчик по модулю 30
17	4-разрядный регистр с параллельной записью и последовательным считыванием; суммирующий счетчик по модулю 27
18	4-разрядный регистр с последовательной записью и параллельным считыванием; вычитающий счетчик по модулю 26

Пример программы на языке VHDL, реализующий трехразрядный регистр с параллельной записью и последовательным считыванием, приведен ниже.

Результаты моделирования показаны на рис. 1.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;

entity reg_vhd is
port (
 -- тактовый сигнал
clk     :in    std_logic;
 -- параллельный информационный вход регистра
d       :in    std_logic_vector(2 downto 0);
 -- инверсный сигнал записи
v       :in    std_logic;
 -- выход регистра
out_sig :out   std_logic;
 -- вывод входного значения на индикацию
d_out   :out   std_logic_vector(2 downto 0)
);
end reg_vhd;
```

```
architecture arch of reg_vhd is

signal d_tmp : std_logic_vector(2 downto 0);
type counter is range 0 to 4;
signal i : counter;

begin

p1: process(clk)
begin
d_out <= d;
if rising_edge(clk) then
        if v = '0' then
              d_tmp <= d;
              I <= 0;
        end if;

        case i is
        when 0 =>           out_sig <= d_tmp(0);
                            I <= i+1;
        when 1 =>           out_sig <= d_tmp(1);
                            I <= i+1;
        when 2 =>           out_sig <= d_tmp(2);
                            I <= i+1;
        when 3 =>           I <= 4;
                            out_sig <= '0';
                            d_tmp <= "000";
        when others =>      null;
        end case;

end if;
end process p1;

end arch;
```

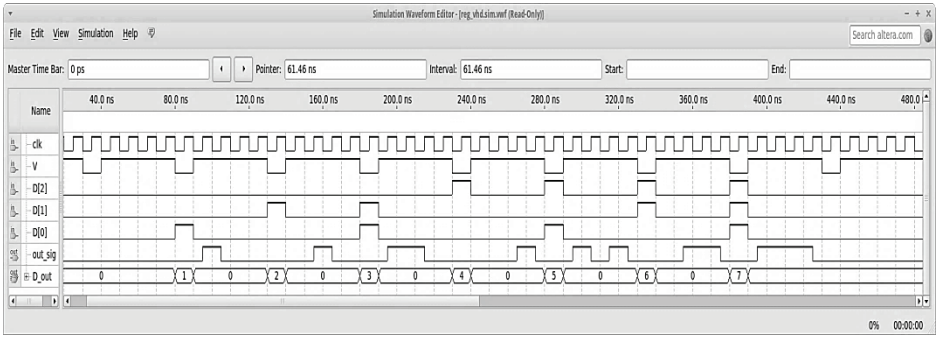


Рис. 1. Временная диаграмма работы трехразрядного регистра с параллельной записью и последовательным считыванием

Ниже приведен пример программы, реализующий четырехразрядный двоичный суммирующий счетчик.

Результаты моделирования показаны на рис. 2.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
use ieee.numeric_std.all;

entity sum4 is
port(
 -- тактовый сигнал
clk     :in    std_logic;
 -- вывод значения счетчика
g:out   std_logic_vector (3 downto 0)
);
end sum4;

architecture bbb of sum4 is

signal st: std_logic_vector (3 downto 0);

begin

process(clk)
```

```
begin
        if (rising_edge (clk)) then
              st <= st+1;
        end if;
end process;

g <= st;

end bbb;
```

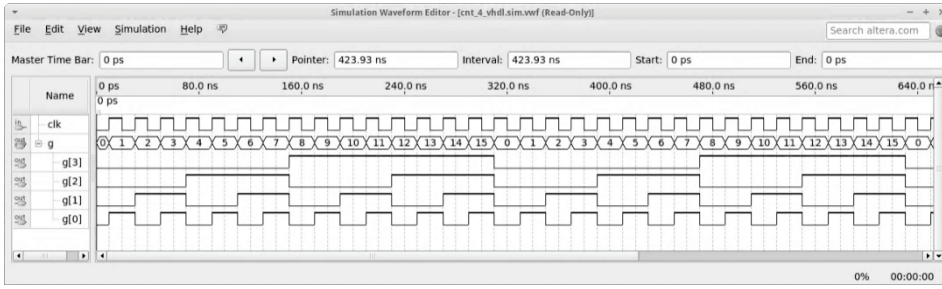


Рис. 2. Временная диаграмма работы четырехразрядного двоичного суммирующего счетчика

Контрольные вопросы

1. В чем преимущество задания проекта в текстовом виде перед графическим заданием?

2. В чем основное отличие любого языка описания аппаратуры от традиционных языков высокого уровня?

3. Назовите основные типы переменных в языке VHDL.

4. Назовите основные элементы алфавита языка VHDL.

5. Приведите пример подключения библиотек.

6. Приведите пример записи портов ввода и вывода.

7. Какие девять типов сигналов поддерживает язык VHDL?

8. Приведите пример типовой программы на языке VHDL. Назовите обязательные структурные элементы программы на языке VHDL.

9. Назовите правила записи идентификаторов.

10. Приведите примеры зарезервированных слов в языке VHDL и назовите их назначение.

ЛАБОРАТОРНАЯ РАБОТА 2.

ИЗУЧЕНИЕ РАБОТЫ ЦИФРОАНАЛОГОВОГО ПРЕОБРАЗОВАТЕЛЯ

Цель работы – закрепление знаний по цифроаналоговому преобразованию и навыков использования программной среды Quartus II.

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

Цифроаналоговый преобразователь (ЦАП) – электронное устройство для преобразования цифрового кода в аналоговый сигнал (ток, напряжение, широтно-импульсный сигнал, перемещение), пропорциональный значению цифрового кода. В устройствах бортовой автоматики применяют несколько типов ЦАП: код – напряжение, код – широтно-импульсный сигнал (см. лаб. раб. № 4) и шаговые двигатели. Особое место занимают преобразователи код – напряжение (ПКН) с суммированием разрядных токов, как наиболее приспособленные к микроминиатюризации. Принцип работы ПКН с суммированием разрядных токов, основанный на применении матриц токовых ключей (МТК), поясняет рис. 3.

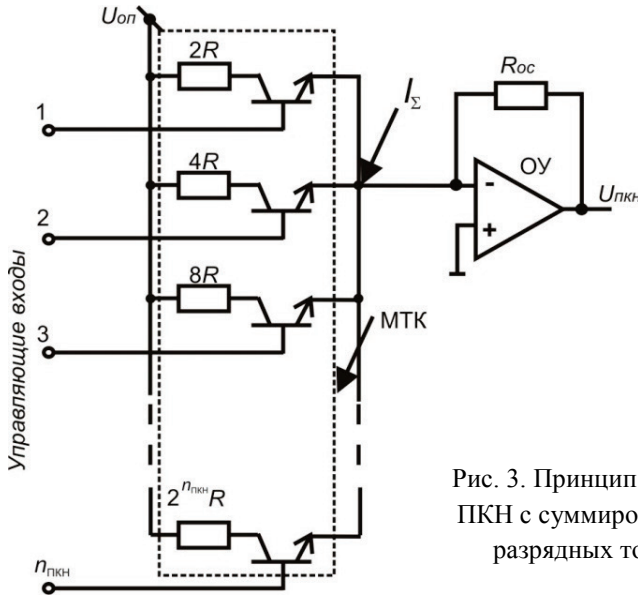


Рис. 3. Принцип работы ПКН с суммированием разрядных токов

Номиналы весовых резисторов подобраны таким образом, что образуют двоично-взвешенный ряд, который обеспечивает деление тока в каждом узле пополам. При этом образуется ряд двоично-взвешенных токов.

На управляющие входы транзисторов поступает цифровой $n_{\text{ПКН}}$-разрядный двоичный код. При поступлении на вход транзисторного ключа логической единицы соответствующий разрядный ток поступает в суммирующую точку, в противном случае – не поступает. В суммирующей точке формируется ток I_{Σ}, пропорциональный поданному на вход цифровому коду. Далее этот ток преобразуется с помощью операционного усилителя в напряжение постоянного тока. Работу ПКН можно описать уравнением

$$U_{\text{ПКН}} = U_{\text{ОП}} \sum_{i=1}^{n_{\text{ПКН}}} \alpha_i 2^{-i}, \qquad (1)$$

где $U_{\text{ОП}}$ – номинальное значение опорного напряжения; i – номера разрядов преобразователя, начиная со старшего; α_i – значение i-го разряда преобразователя (0 или 1); $n_{\text{ПКН}}$ – число разрядов ПКН.

Таким образом, собственно преобразованию подвергается опорный аналоговый сигнал, а кодовый цифровой сигнал является модулирующим.

В состав ПКН функционально входят четыре блока (рис. 4): буферный регистр (БР), матрица токовых ключей (МТК), выходной операционный усилитель (ОУ) и источник опорного напряжения (ИОН).

БР предназначен для временного хранения цифрового кода, который требуется преобразовать в напряжение.

МТК преобразует код, поступающий на ее вход с выхода БР, в пропорциональное этому коду значение тока.

ИОН формирует стабилизированное опорное напря-

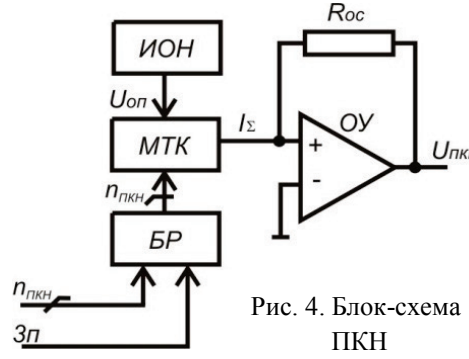


Рис. 4. Блок-схема ПКН

жение $U_{ОП}$ постоянного тока, которое подается на МТК. Как правило, $U_{ОП}$ применяется в двух номиналах: 10,23 В и 5,11 В, что позволяет легко определять разрешающую способность преобразователя. ИОН может быть программируемым, т.е. формировать $U_{ОП}$ в функции некоторого кода. При этом выходной сигнал ПКН будет равен произведению опорного напряжения $U_{ОП}$ = VAR и входного цифрового кода. Такие ПКН называют умножающими.

Выходной ОУ преобразует ток, поступающий с МТК, в однополярное напряжение постоянного тока. Для получения с выхода ОУ напряжения обоих знаков необходимо ввести второй ОУ либо другими мерами обеспечить симметричное смещение статической характеристики в отрицательную область.

МТК, показанная на рис. 3, на практике не применяется. Это обусловлено следующим. При увеличении числа разрядов ПКН в силу технологического разброса номиналов резисторов существенно возрастает методическая погрешность преобразования. При числе разрядов 10 и более технологический допуск на «старший» резистор даже для высокоточных однопроцентных резисторов превышает значения трех «младших» резисторов. Таким образом, одиннадцатиразрядный ПКН обладает гарантированной точностью в семь разрядов. Интересным и более технологичным вариантом резистивной матрицы является цепная схема типа *R*-2*R* (рис. 5).

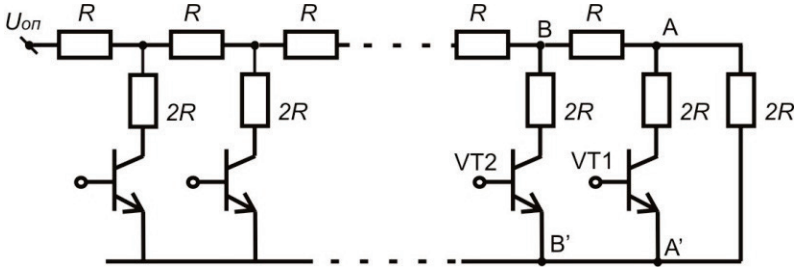


Рис. 5. МТК типа *R*-2*R*

Рассмотрим принцип работы МТК типа *R*-2*R*.

Ток, втекая в узел А при открытом транзисторе VT1, поделится пополам. Сопротивление $R_{общ}$ между точками А и А': $R_{A\text{-}A'} = R$. Тогда

$R_{B\text{-}A\text{-}A'} = 2R$. Поэтому ток, втекая в узел В при открытом транзисторе VT2, тоже поделится пополам и т.д. Таким образом, реализуется ряд двоично-взвешенных токов в коллекторных цепях транзисторных ключей.

Достоинством резистивной матрицы R-$2R$ по сравнению с матрицей резисторов, показанной на рис. 5, является использование резисторов только двух номиналов. На структурной схеме МТК можно представить как элемент с многоступенчатой релейной характеристикой.

В устройствах бортовой автоматики коды, поступающие на вход ПКН, обычно имеют знак и представлены в дополнительном либо обратном кодах.

Например, с помощью восьми разрядов можно представить числа в диапазоне от −128 до +127. При вводе чисел в ПКН этот диапазон чисел обычно сдвигают до 0...255 путем прибавления 128. Числа, большие 128, при этом считаются положительными, а числа, меньшие 128, – отрицательными. Среднее число 128 соответствует нулю. Такое представление чисел со знаком называется смещенным кодом. Прибавление числа, составляющего половину полной шкалы данной разрядности (в данном примере это 128), можно легко выполнить путем инверсии старшего (знакового) разряда. Соответствие рассмотренных кодов иллюстрируется табл. 2.

Таблица 2

Десятичный	Дополнительный	Смещенный	Аналог $U/U_{макс}$
127	01111111	11111111	127/255
1	00000001	10000001	1/255
0	00000000	10000000	0
−1	11111111	01111111	−1/255
−127	10000001	00000001	−127/255
−128	10000000	00000000	−128/255

Чтобы получить выходной сигнал с правильным знаком, необходимо осуществить обратный сдвиг путем вычитания тока или напряжения, составляющего половину шкалы преобразователя.

Статический коэффициент передачи ПКН, отношение максимальной выходной величины ($U_{\text{ОП}}$) к максимальной входной величине $2^{n_{\text{ПКН}}} - 1$, определяет разрешающую способность преобразователя:

$$K_{\text{ПКН}} = \frac{U_{\text{ОП}}}{2^{n_{\text{ПКН}}-1}} = \delta U_{\text{ПКН}} \text{ [В]}. \quad (2)$$

Статическая характеристика ПКН показана на рис. 6.

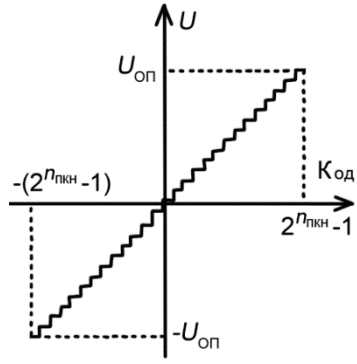


Рис. 6. Статическая характеристика ПКН

Важными статическими параметрами ПКН также являются:

1) Погрешность смещения нуля – часть общей погрешности, характеризующая параллельный сдвиг статической характеристики реального преобразователя по отношению к идеальному (рис. 7, *а*). На рисунке погрешность смещения нуля равна половине единицы младшего разряда (ЕМР).

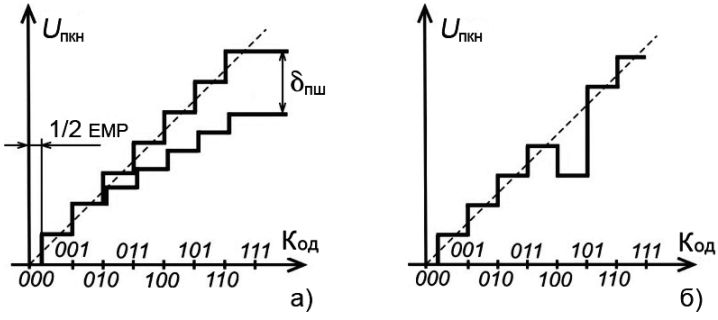


Рис. 7. Статические параметры ПКН

2) Абсолютная погрешность преобразования (погрешность шкалы $\delta_{\text{пш}}$) – отклонение значения выходного напряжения от номинального значения, соответствующего конечной точке характеристики (рис. 7, *а*).

3) Температурная погрешность – составляющая статической погрешности, возникающая при изменении температуры окружающей среды в заданном диапазоне.

4) Монотонность статической характеристики – идентичность знака приращения мгновенных значений входного и выходного сигналов преобразователя (рис. 7, *б*).

Основной динамической характеристикой ПКН является время установления выходного напряжения – интервал времени от подачи кода на вход буферного регистра до установления напряжения на выходе ОУ, соответствующего поданному коду.

Согласование разрядных сеток вычислителя и цифроаналогового преобразователя

Различное число разрядов специализированного микропроцессорного вычислителя (СМВ) и ЦАП приводит к необходимости согласования разрядных сеток этих устройств. Рассмотрим наиболее типичный случай, для которого характерным является $n_{\text{СМВ}} > n_{\text{ЦАП}}$.

Для сохранения точности, определяемой числом разрядов СМВ, согласование разрядных сеток должно осуществляться со стороны младших разрядов (рис. 8). При этом, однако, так как старшие разряды СМВ физически не соединяются с ЦАП, возможно появление так называемых «ложных нулей» (рис. 9, *а*). Действительно, коды, равные $2^{n_{\text{ЦАП}}}$, $2^{n_{\text{ЦАП}}+1}$, $2^{n_{\text{ЦАП}}+2}$, … $2^{n_{\text{СМВ}}-n_{\text{ЦАП}}}$, ЦАП воспринимает как нуль, а коды $2^{n_{\text{ЦАП}}}+1$, $2^{n_{\text{ЦАП}}+1}+1$, $2^{n_{\text{ЦАП}}+2}+1$, … $2^{n_{\text{СМВ}}-n_{\text{ЦАП}}}+1$ – как единицу и т.д.

Естественная характеристика пары СМВ–ЦАП показана на рис. 9, *а*. $К_У$ – код управления.

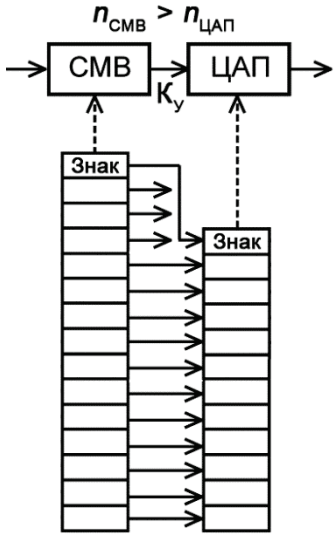


Рис. 8. Согласование разрядных сеток СМВ и ЦАП

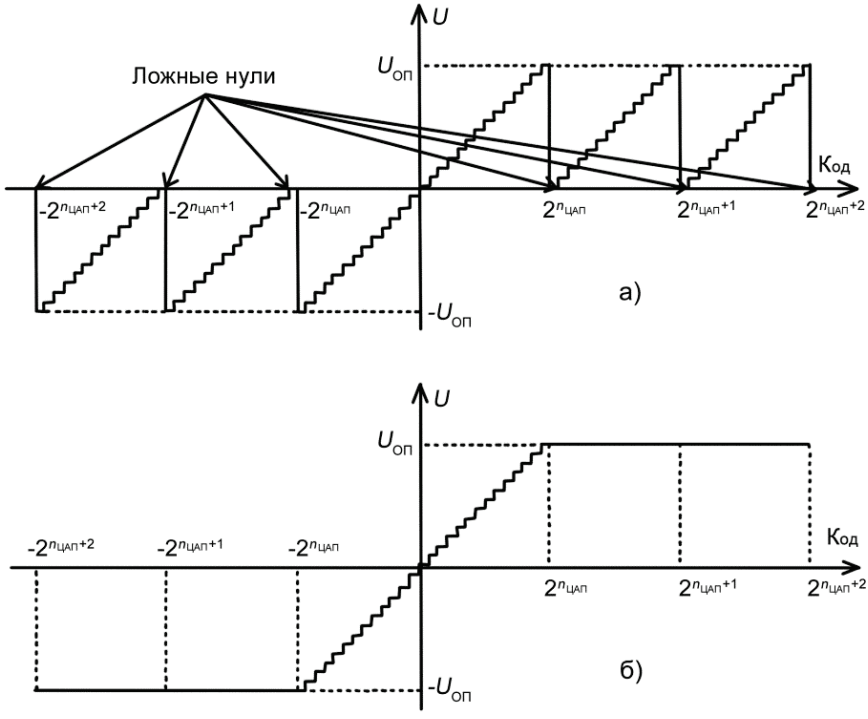


Рис. 9. Естественная (*а*) и искусственная (*б*) характеристики СМВ–ЦАП

Для исключения «ложных нулей» вводится специальный алгоритм:

$$K_y = \begin{cases} K_y, \text{если } K_y < |2^{n_{ПКН}} - 1|; \\ (2^{n_{ПКН}} - 1)Sign\ K_y), \text{ если } K_y \geq |2^{n_{ПКН}} - 1|. \end{cases} \quad (3)$$

Искусственная характеристика пары СМВ–ЦАП с учетом алгоритма (3) показана на рис. 9, *б*. Алгоритм (3) является очень важным, так как обеспечивает однозначность нуля.

Практическая часть

При выполнении данной лабораторной работы используется один канал трехканального 10-разрядного высокоскоростного видео-ЦАП ADV 7123, расположенного на плате DE 2.

Выводы ЦАП приведены на рис. 10, функциональная схема ЦАП – на рис. 11.

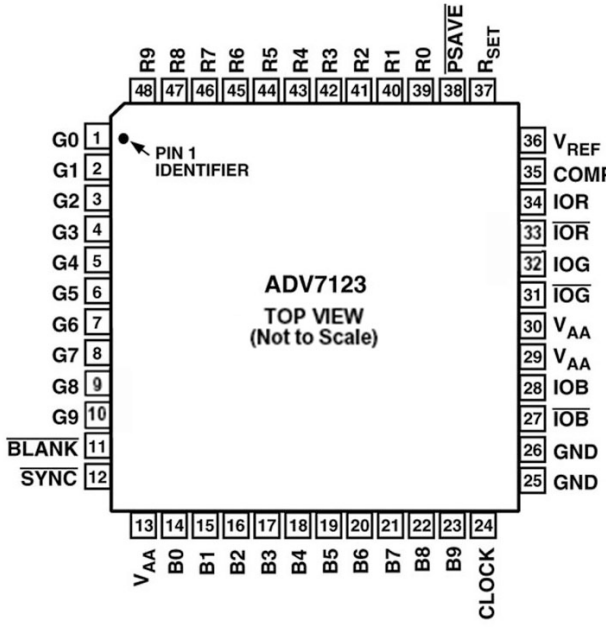


Рис. 10. Внешний вид микросхемы ADV 7123

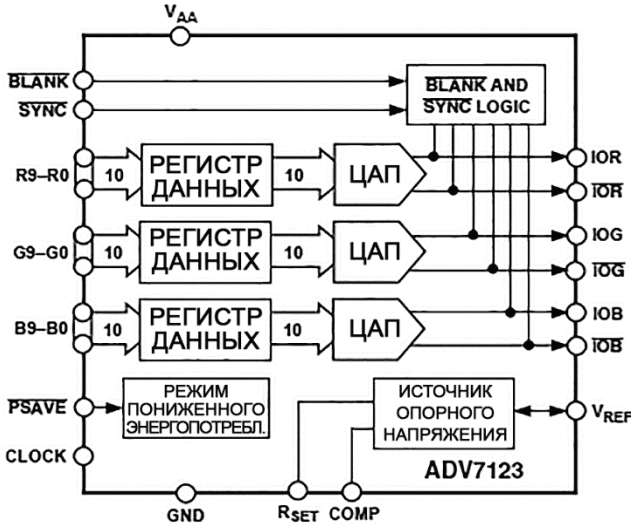


Рис. 11. Функциональная схема ADV7123

Из всего функционала данного ЦАП для лабораторной работы понадобятся только:

1) входы R0-R9 ЦАП, которые следует подключить к ПЛИС;

2) вход CLOCK, который подключается к генератору тактовой частоты 50 МГц (clock_50 => PIN_N2);

3) вход BLANCK, нужный для разрешения выходного сигнала (при значении BLANCK = 1 разрешен);

4) выходы IOR и GND для подключения цифрового вольтметра через контакты 1 и 5 разъема VGA (рис. 6 [8]) с помощью жгута (рис 12).

Схема эксперимента показана на рис. 13.

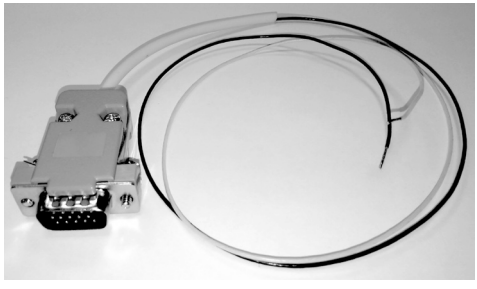


Рис. 12. Жгут для подключения

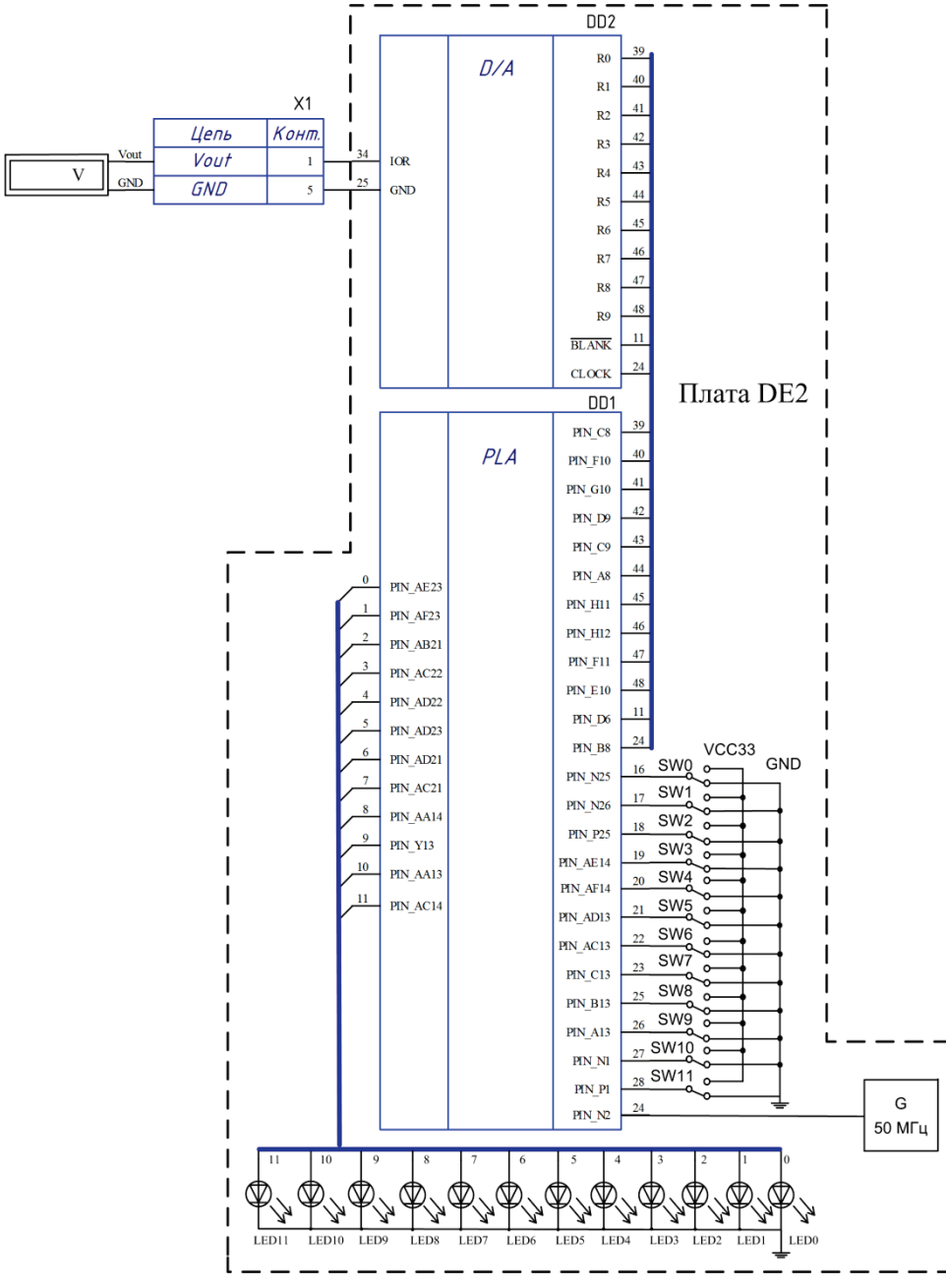


Рис. 13. Схема эксперимента для исследования ЦАП

Ниже приведен текст программы для работы с ЦАП на языке VHDL.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity dac is
port(
 -- ввод тактового сигнала с генератора на плате DE2
clk     :in    std_logic;
 -- входной код с тумблеров SW0 - SW15
sw_in   :in    std_logic_vector(15 downto 0);
 -- вывод входного кода из ПЛИС на ЦАП
dac_out :out   std_logic_vector(9 downto 0);
 -- вывод входного кода на индикацию для удобства
считывания
led_out :out   std_logic_vector(15 downto 0);
 -- вывод тактового сигнала на ЦАП
vga_clk :out   std_logic;
 -- вывод сигнала гашения на ЦАП
blank   :out   std_logic
);
end dac;

architecture arch1 of dac is

begin

blank   <= '1';
led_out <= sw_in;
dac_out <= sw_in(9 downto 0);
vga_clk <= clk;

end arch1;
```

Методика выполнения лабораторной работы

1. Собрать схему эксперимента в соответствии с рис. 13 . При этом через разъем X1 (VGA, см. рис. 6 [8]) с помощью жгута (рис. 12, сигнальный вывод – белый провод, заземление – черный провод) подключить к выводам ЦАП цифровой вольтметр.

2. Ввести приведенную программу, откомпилировать, загрузить в ПЛИС с помощью режима *programmer*.

3. Изменяя входной код с помощью тумблеров-переключателей SW0-SW9 (младший разряд SW0) и снимая показания с вольтметра, заполнить табл. 3.

Таблица 3

№ п/п	Входной код	Выходное напряжение

4. Построить график зависимости напряжения от кода. Убедиться в наличии «ложных нулей».

5. Доработать исходную программу так, чтобы «ложные нули» отсутствовали.

6. Повторно выполнить пункты 2, 3, 4. Убедиться в отсутствии «ложных нулей».

7. Используя данные любой из заполненных таблиц, определить статический коэффициент передачи ЦАП.

Варианты заданий приведены в табл. 4.

Таблица 4

№ варианта	Числовой диапазон входного кода	Количество точек
1	0-2100	20
2	0-2200	20
3	0-2300	20
4	0-2400	20
5	0-2500	20
6	0-2600	20

Окончание таблицы 4

7	0-2700	20
8	0-2800	20
9	0-2900	20
10	0-3000	20
11	0-3100	20
12	0-3200	20
13	0-3300	20
14	0-3400	20
15	0-3500	20
16	0-3600	20
17	0-3700	20
18	0-3800	20

Контрольные вопросы

1. Расскажите о принципе работы ПКН с суммированием токов.

2. Расскажите о принципе работы матрицы R-2R.

3. Назовите основные характеристики ПКН.

4. Из каких основных блоков состоит ПКН?

5. Чему равен статический коэффициент передачи ПКН?

6. Напишите зависимость, связывающую значения опорного напряжения, входного кода и напряжения на выходе ПКН.

7. В чем суть «ложных нулей» при согласовании разрядных сеток вычислителя и ЦАП?

8. $U_{ОП}$ = 10 В, на выходе шестиразрядного ПКН установилось напряжение $U_{ПКН}$ = 2 В. Какой двоичный код поступил на вход преобразователя?

9. $U_{ОП}$ = 10 В, на вход преобразователя поступил восьмиразрядный код: 11001100. Какое напряжение $U_{ПКН}$ установилось на выходе ПКН?

10. Что такое «смещенный код», и для каких целей он используется?

Лабораторная работа 3.

Изучение работы аналого-цифрового преобразователя

Цель работы – закрепление знаний по принципам аналого-цифрового преобразования и навыков по использованию макетных плат и программной среды Quartus II.

Теоретическая часть

Аналого-цифровые преобразователи (АЦП) широко используются в устройствах бортовой автоматики. Наибольшее применение находят микроэлектронные преобразователи напряжение – код (ПНК) в составе датчиков первичной информации и датчиков обратных связей. Промышленностью выпускаются ПНК нескольких типов. Рассмотрим более подробно ПНК, основанные на методах мгновенного кодирования и последовательного приближения. Одним из важнейших элементов ПНК являются компараторы напряжения – специализированные операционные усилители с дифференциальным входом и цифровым выходом, выполняющие функции пороговых устройств.

ПНК мгновенного кодирования

Функциональная схема ПНК мгновенного кодирования показана на рис. 14.

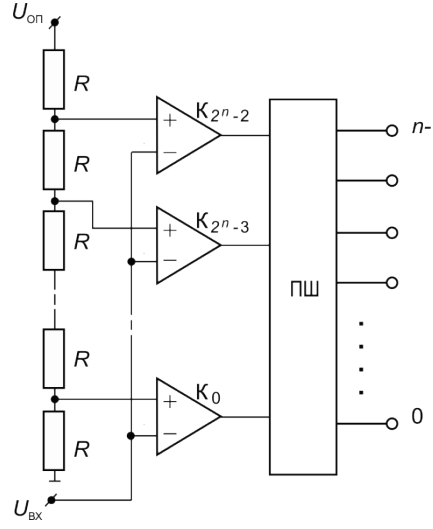


Рис. 14. Функциональная схема ПНК мгновенного кодирования

На рис. 14 приняты следующие обозначения:

$U_{\text{ОП}}$ – стабилизированное опорное напряжение постоянного тока;

$U_{\text{ВХ}}$ – входной сигнал;

К_i – компаратор – операционный усилитель, имеющий три каскада и положительную обратную связь;

ПШ – приоритетный шифратор.

Резистивная матрица, состоящая из резисторов одинакового номинала и собранная по схеме делителя напряжений, обеспечивает систему опорных напряжений $U_{\text{ОП}_i}$, имеющую линейную зависимость.

При методе мгновенного кодирования (рис. 14) первые входы всех $(2^{n_{\text{ПНК}}} - 1)$ компараторов объединены и на них поступает входной аналоговый сигнал $U_{\text{ВХ}}$.

Вторые входы компараторов подключены к $(2^{n_{\text{ПНК}}} - 1)$ источникам опорных напряжений $U_{\text{ОП}_i}$.

Уравнение, описывающее работу компаратора, можно записать как

$$\text{К}_i = \begin{cases} 0, \text{если } U_{\text{ВХ}} \le U_{\text{ОП}_i}; \\ 1, \text{если } U_{\text{ВХ}} > U_{\text{ОП}_i}. \end{cases} \quad (4)$$

На выходе компараторов формируется $(2^{n_{\text{ПНК}}} - 1)$-разрядный код в соответствии с (4).

ПШ преобразует $(2^{n_{\text{ПНК}}} - 1)$-разрядный код в выходной $n_{\text{ПНК}}$-разрядный код.

В качестве примера для трехразрядного ПНК мгновенного кодирования приведем таблицу соответствия (табл. 5) между поступающим входным напряжением $U_{\text{ВХ}}$, $(2^{n_{\text{ПНК}}} - 1)$-разрядным кодом, формируемым компараторами, и $n_{\text{ПНК}}$-разрядным выходным кодом, формируемым ПШ.

Метод отличает очень высокое быстродействие – типовое значение времени преобразования составляет 1...30 нс. К недостаткам следует отнести малое число разрядов (8–10), которого достигают при современном уровне развития технологии.

Таблица 5

$U_{\text{ВХ}}$	$(2^{n_{\text{ПНК}}}-1)$-разрядный код	$n_{\text{ПНК}}$-разрядный код
0	0000000	000
$U_{\text{ОП}}/7$	0000001	001
$2U_{\text{ОП}}/7$	0000011	010
$3U_{\text{ОП}}/7$	0000111	011
$4U_{\text{ОП}}/7$	0001111	100
$5U_{\text{ОП}}/7$	0011111	101
$6U_{\text{ОП}}/7$	0111111	110
$U_{\text{ОП}}$	1111111	111

ПНК последовательных приближений

В основе ПНК последовательных приближений лежит принцип **дихотомии** (от греч. dichotomia – разделение надвое) – поиск решения методом последовательного деления числового отрезка пополам.

В состав ПНК, основанного на методе последовательных приближений, входят сдвиговый регистр последовательных приближений (РПП) с параллельным выходом, преобразователь код – напряжение (ПКН), функции БР в котором выполняет РПП, генератор импульсов (ГИ), компаратор – К (рис. 15). При этом ПНК может быть собран как из отдельных функциональных блоков, так и представлять собой функционально законченное устройство.

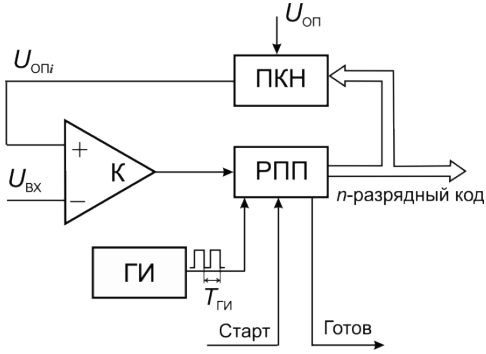


Рис. 15. Функциональная схема ПНК последовательных приближений

Метод последовательных приближений основан на формировании сдвиговым регистром последовательных приближений пробных кодов, которые, поступая на ПКН, преобразуются в систему опорных напряжений. Входной аналоговый сигнал с помощью компаратора последовательно сравнивается с этой системой опорных напряжений. На выходе компаратора в зависимости от результата сравнения формируется либо логическая единица, либо нуль, который поступает на последовательный вход РПП и заносится в тестируемый разряд. Сравнение входного сигнала с опорным ведется, начиная со значения $U_{\text{ОП}}/2$, т.е. представляет собой двоичный поиск с середины; n-разрядное преобразование выполняется за n+1 такт.

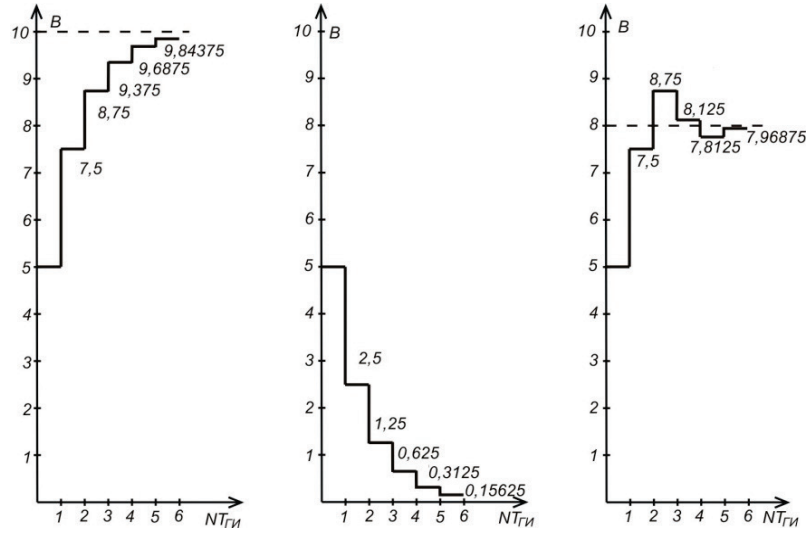
ГИ формирует бесконечную последовательность импульсов. После прихода сигнала «Старт» по первому импульсу с ГИ в РПП записывается код, равный половине от максимального: 1000…0. Этот код поступает на выход и одновременно на вход ПКН. На выходе ПКН формируется напряжение, равное $U_{\text{ОП}}/2$, которое сравнивается с $U_{\text{ВХ}}$ на компараторе. По результатам сравнения (4) компаратор формирует «1» или «0», который по второму импульсу с ГИ записывается в старший разряд, а остальные разряды, включая «тестовую» единицу, сдвигаются на один бит вправо.

Этот процесс циклически повторяется до тех пор, пока «тестовая» единица не «выталкивается» из РПП на выход «Готов». В этот момент времени на выходе РПП сформируется код, который «уравновесит» $U_{\text{ВХ}}$.

Время преобразования $t_{\text{ПР}} = (n_{\text{ПНК}}+1)T_{\text{ГИ}}$, где $n_{\text{ПНК}}$ – число разрядов преобразователя; $T_{\text{ГИ}}$ – длительность одного периода импульса с ГИ.

Рис. 16 иллюстрирует работу ПНК последовательных приближений при условиях: $n_{\text{ПНК}} = 5$, $U_{\text{ОП}}$=10 В для трех значений входного сигнала $U_{\text{ВХ1}} = 10$ В (*а*); $U_{\text{ВХ2}} = 0$ В (*б*); $U_{\text{ВХ3}} = 8$ В (*в*).

Табл. 6 поясняет работу ПНК последовательных приближений для примера, соответствующего рис. 16, *в*.



а) б) в)

Рис. 16. Формы напряжения на выходе ПКН

Таблица 6

Номер такта	Текущий код РПП	Напряжения на выходе ПКН, $U_{\text{ОП}_i}$ [В]	Значение сигнала «Готов»
1	**1** 0 0 0 0	5	0
2	1 **1** 0 0 0	7,5	0
3	1 1 **1** 0 0	8,75	0
4	1 1 0 **1** 0	8,125	0
5	1 1 0 0 **1**	7,8125	0
6	1 1 0 0 1	7,96875	**1**

Преобразователи этого типа оперируют мгновенными значениями входного аналогового сигнала, поэтому погрешность преобразования определяется величиной изменения входного сигнала за время преобразования. В связи с этим для ПНК последовательных приближений очень нежелательны выбросы и помехи во входном сигнале.

ПНК имеют те же статические характеристики, что и ПКН.

Коэффициент передачи ПНК определяется как

$$K_{\text{ПНК}} = \frac{2^{n_{\text{ПНК}}-1}}{U_{\text{оп}}} \text{ [1/В]}. \quad (5)$$

Величина, обратная коэффициенту передачи ПНК – $\delta U_{\text{ПНК}}$, определяет разрешающую способность преобразования.

Основной динамической характеристикой является время преобразования $t_{\text{ПР}}$.

Согласование разрядных сеток АЦП и СМВ

При проектировании устройств бортовой автоматики очень важным является вопрос согласования разрядных сеток АЦП и СМВ. Согласование разрядных сеток АЦП и СМВ осуществляется со стороны младших разрядов (рис. 17). Только в этом случае точность, определяемая числом разрядов АЦП, не изменяется. При этом, так как обычно выполняется условие $n_{\text{СМВ}} > n_{\text{АЦП}}$, в старшие разряды СМВ заносится знаковый разряд АЦП. Обычно данные от АЦП поступают в обратном коде. Поэтому для положительных кодов происходит доопределение «незначащим» нулем, а для отрицательных – «незначащей» единицей.

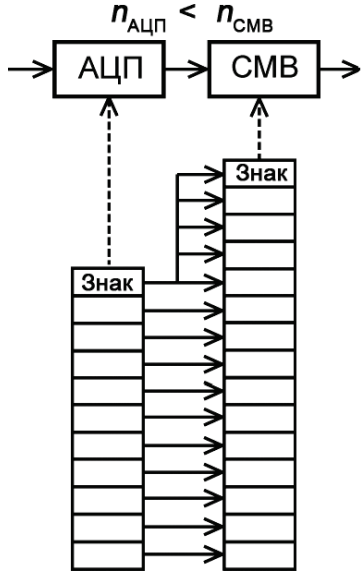


Рис. 17. Согласование разрядных сеток АЦП и СМВ

Практическая часть

Изучение принципов работы АЦП происходит на примере аналого-цифрового преобразователя AD7896ANZ. В качестве задатчика входного напряжения используется переменный резистор.

Принципиальная схема исследуемого АЦП приведена на рис. 18.

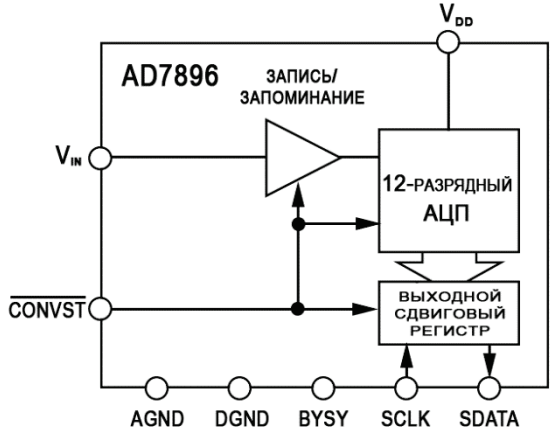


Рис. 18. Принципиальная схема АЦП AD7896ANZ

Назначение входов/выходов АЦП:

1) V_{IN} – входное аналоговое напряжение;

2) V_{DD} – опорное напряжение АЦП, может использоваться напряжение в диапазоне от 2,7 до 5,5 В, в лабораторной работе – 5 В;

3) AGND – аналоговая земля;

4) SCLK – тактовый сигнал, используемый для передачи информации через последовательный канал;

5) BUSY – сигнал «занятости» АЦП, находится в высоком уровне непосредственно во время преобразования;

6) CONVST – сигнал, формируемый на ПЛИС, по переднему фронту которого начинается преобразование;

7) DGND – цифровая земля;

8) SDATA – последовательный канал передачи данных.

Выводы и внешний вид АЦП показаны на рис. 19, *а*. Внешний вид переменного резистора показан на рис. 19, *б*, причем красным цветом обведен неиспользуемый вывод.

а) б)

Рис. 19. Выводы и внешний вид: *а* – АЦП AD7896ANZ; *б* – переменного резистора

Аналого-цифровой преобразователь и переменный резистор устанавливаются на макетной плате (рис. 20).

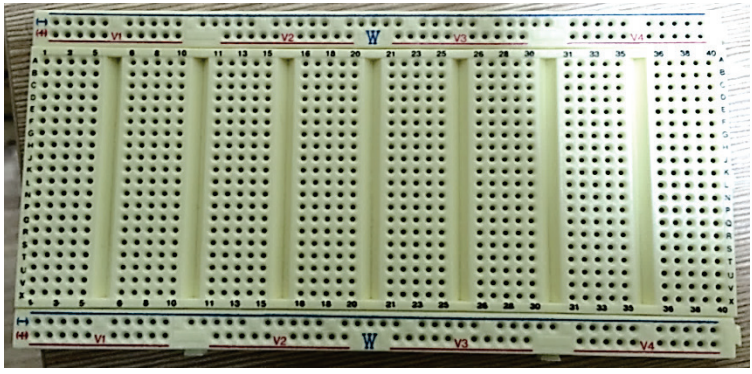


Рис. 20. Внешний вид макетной платы

При работе с макетной платой необходимо иметь в виду следующее:

1) восемь секций изолированы друг от друга;

2) внутри каждой секции уложены токопроводящие шины параллельно от линии A до X;

3) с двух сторон относительно секций уложены две токопроводящие шины W (на плате обозначены синим цветом) и восемь токопроводящих шин V1…V4 (на плате обозначены красным цветом).

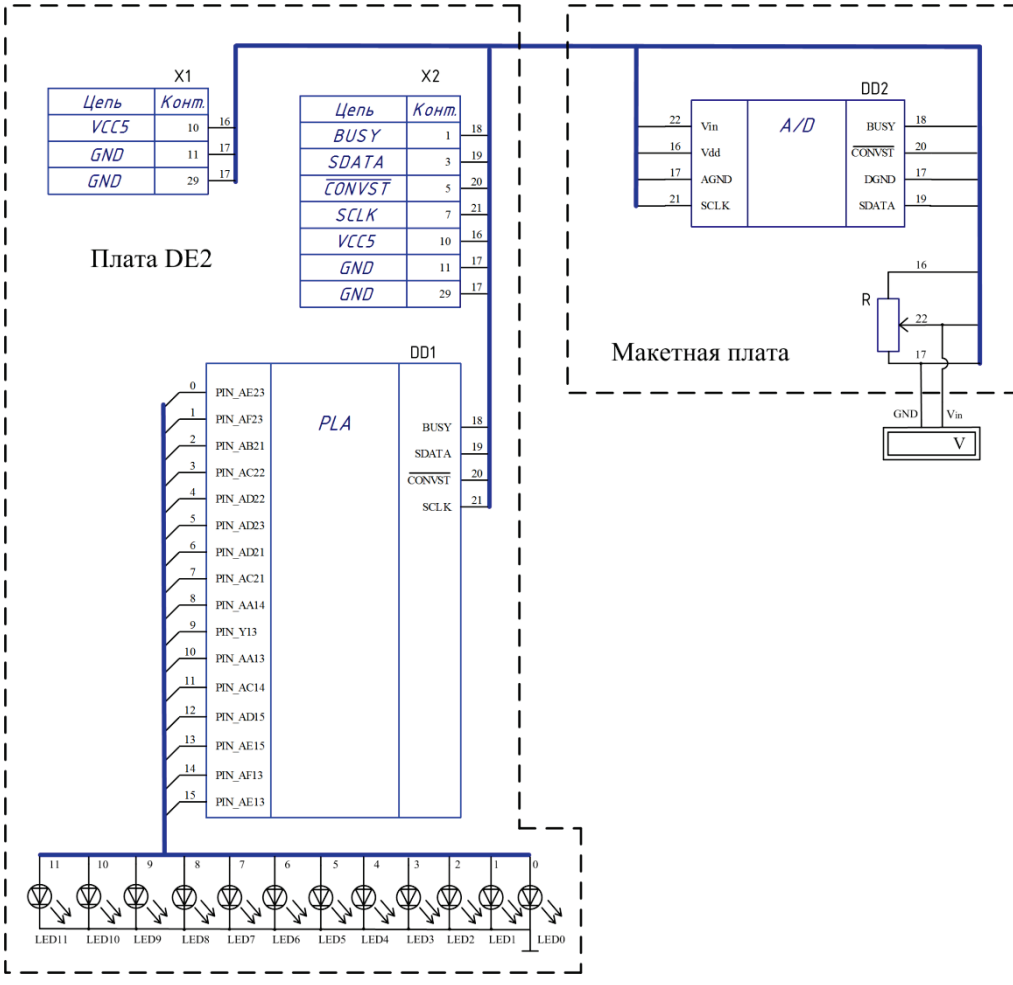


Рис. 21. Схема эксперимента

Схема лабораторной работы показана на рис. 21. Ниже приведен текст программы исследования работы АЦП.

```
library ieee;

use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
```

```
entity adc is
port(
 -- тактовый сигнал
clk          :in    std_logic;
 -- последовательный канал передачи данных из АЦП
sdata        :in    std_logic;
 -- сигнал “занятости” АЦП
busy         :in    std_logic;
 -- тактовый сигнал последовательного канала
sclk         :buffer      std_logic;
 -- сигнал начала преобразования
convst       :out   std_logic;
 -- вывод полученного значения на индикацию
output       :out   std_logic_vector(0 to 15)
);
end adc;

architecture telo of adc is

signal counter            :integer range 0 to 6;
signal counter_shet       :integer range 0 to 31;
signal counter_shet1      :integer range 0 to 1000;
signal counter_reg        :integer range 0 to 15;
signal counter_convst     :integer range 0 to 100;
signal reg                :std_logic_vector (0 to 15);
signal sbros_shet         :std_logic;
signal clk1               :std_logic;
signal clk2               :std_logic;
signal shet               :std_logic;

begin

process(clk)
begin
output <= reg;
sclk   <= clk2;
```

```
if rising_edge(clk) then
       if counter = 5 then
              counter <= 0;
              clk1 <= not clk1;
       else
              counter <= counter+1;
       end if;

       if counter_shet1 = 0 then
              convst <= '0';
              counter_shet1 <= counter_shet1+1;
       elsif counter_shet1 = 10 then
              convst <= '1';
              counter_shet1 <= counter_shet1+1;
       elsif counter_shet1 = 999 then
              counter_shet1 <= 0;
       else
              counter_shet1 <= counter_shet1+1;
       end if;

end if;


if rising_edge(clk1) then
       if shet = '1' then
              if counter_shet = 31 then
                     clk2 <= '0';
                     sbros_shet <= '1';
                     counter_shet <= 0;
              else
                     counter_shet <= counter_shet+1;
                     clk2 <= not clk2;
              end if;
       end if;
end if;

if rising_edge(clk2)then
```

```
      if counter_reg = 15 then
            counter_reg <= 0;
            reg(counter_reg) <= sdata;
      elsif counter_reg > 3 then
            counter_reg <= counter_reg+1;
            reg(counter_reg) <= sdata;
      else
            counter_reg <= counter_reg+1;
      end if;
end if;

if sbros_shet = '1' then
      shet <= '0';
elsif falling_edge(busy) then
      shet <= '1';
end if;

if sbros_shet = '1' then
      sbros_shet <= '0';
end if;

end process;
end telo;
```

Выводы АЦП необходимо подключить к выводам ПЛИС в соответствии с табл. 7.

Таблица 7

Сигнал АЦП	Вывод ПЛИС
BUSY	PIN_K26
CONVST	PIN_M20
SCLK	PIN_M21
SDATA	PIN_M23

Встроенный генератор импульсов с опорной частотой 50 МГц (CLOCK) подсоединить к выводу PIN_N2 ПЛИС.

Сигналы output выводятся на красные светодиоды (табл. П. 2.3 [8], стр. 87).

Методика выполнения лабораторной работы

1. Собрать схему эксперимента в соответствии с рис. 21. При этом подключить цифровой вольтметр к выводам переменного сопротивления с помощью зажимов.

2. Ввести программу, откомпилировать, загрузить в ПЛИС с помощью режима *programmer*.

3. Изменяя входное напряжение с помощью переменного сопротивления и снимая показания с вольтметра и со светодиодов, заполнить табл. 8.

Таблица 8

№ п/п	Входное напряжение	Выходной код

4. Построить график зависимости выходного кода от напряжения.

Варианты заданий приведены в табл. 9.

Таблица 9

№ вар.	Диапазон входного напряжения, В	Количество точек
1	0-5	15
2	0-5	16
3	0-5	17
4	0-5	18
5	0-5	19
6	0-5	20
7	0-5	21
8	0-5	22
9	0-5	23
10	0-5	24
11	0-5	25
12	0-5	26

Окончание таблицы 9

№ вар.	Диапазон входного напряжения, В	Количество точек
13	0-5	25
14	0-5	24
15	0-5	23
16	0-5	22
17	0-5	21
18	0-5	20

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. *Расскажите о принципе работы АЦП мгновенного кодирования.*
2. *Расскажите о принципе работы АЦП последовательных приближений.*
3. *Какова роль компараторов в работе АЦП?*
4. *Как согласовать разрядные сетки вычислителя и АЦП?*
5. *Из каких основных блоков состоит АЦП последовательных приближений?*
6. *Чему равен статический коэффициент передачи АЦП?*
7. *В каком коде передаются данные из АЦП AD7896ANZ в ПЛИС?*
8. *$U_{ОП}$ = 10 В, на вход пятиразрядного преобразователя последовательных приближений поступило 3 В. Нарисуйте эпюру изменения напряжения на выходе ЦАП.*
9. *На выходе пятиразрядного АЦП мгновенного кодирования установился код: 10101. Какое напряжение поступило на вход преобразователя?*
10. *На выходе пятиразрядного АЦП мгновенного кодирования установился код: 11011. Какой код сформировали компараторы на входе приоритетного шифратора?*

Лабораторная работа 4.

Разработка цифрового устройства управления двигателем постоянного тока

Цель работы – закрепление знаний по принципам импульсного управления двигателями постоянного тока, навыков по программированию на языке VHDL, по использованию макетных плат и программной среды Quartus II.

Теоретическая часть

В устройствах бортовой автоматики, например при управлении двигателями-маховиками, применяют импульсные методы регулирования скорости. Вариант структуры цифрового следящего привода (ЦСП), который обеспечивает принцип прямого цифрового управления исполнительным двигателем, показан на рис. 22. В этом случае вместо ЦАП используется управляющий логический автомат (УЛА), функции которого заключаются в реализации законов управления системой «импульсный усилитель мощности – исполнительный двигатель» (ИУМ–ИД). Пунктиром выделен мехатронный модуль, в состав которого, кроме УЛА, ИУМ и ИД, могут входить устройства, реализующие обратные связи (ОС).

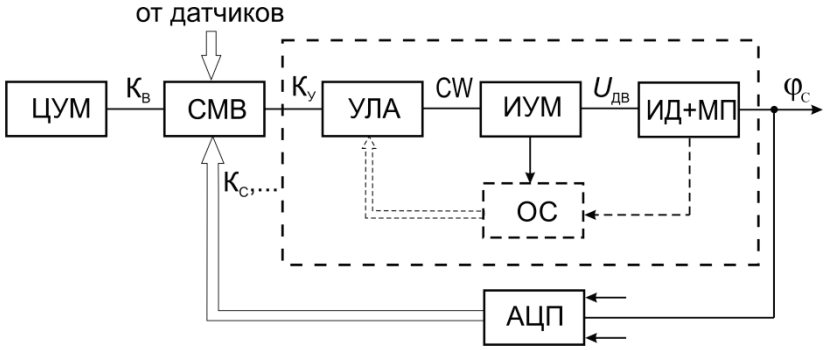


Рис. 22. Блок-схема ЦСП с прямым цифровым управлением

Цифровая управляющая машина (ЦУМ) формирует требуемый закон движения выходного вала привода в виде некоторой функции

$\varphi_{\text{В}}(t)$ и передает его как задающее воздействие на ЦСП в виде n-разрядного двоичного кода $\text{К}_{\text{В}}$. Специализированный микропроцессорный вычислитель (СМВ) принимает этот код, n-разрядный двоичный код $\text{К}_{\text{С}}$ и, возможно, сигналы с каких-либо датчиков первичной информации. На основе полученных сигналов СМВ формирует код ошибки $\text{К}_{\theta} = \text{К}_{\text{В}} - \text{К}_{\text{С}}$, а при необходимости – код управления $\text{К}_{\text{У}}$, включающий, кроме пропорциональной составляющей К_{θ}, например интегральную и дифференциальную. Код управления $\text{К}_{\text{У}}$ в зависимости от порядка астатизма системы несет информацию о желаемом положении, скорости или ускорении. $\text{К}_{\text{У}}$ поступает на УЛА, который может быть либо программируемым, либо реализовывать жесткую программу. УЛА реализует требуемый закон управления процессом коммутации ключевых элементов (КЭ), то есть преобразует код управления $\text{К}_{\text{У}}$ в циклическую последовательность двоичных **управляющих слов** CW (Control Word) или **состояний.** ИУМ преобразует последовательность CW в среднее напряжение $U_{\text{ДВ}}$, которое ИД и механическая передача (МП) преобразуют в перемещение $\varphi_{\text{С}}$. АЦП преобразует угол поворота выходного вала $\varphi_{\text{С}}$ в пропорциональный этому углу n-разрядный двоичный код $\text{К}_{\text{С}}$.

Такая структура ЦСП обеспечивает прямое цифровое управление исполнительным устройством (первой непрерывной координатой при импульсном управлении является ток в обмотках двигателя). При этом сохраняются все преимущества микропроцессорной техники. К ним относятся высокая степень интеграции электронных элементов на кристалле кремния и, как следствие, малая потребляемая мощность, малые веса и габариты, высокая надежность. Эти качества особенно важны для электроприводов летательных аппаратов.

Импульсный метод регулирования скорости в системе ИУМ-ИД основан на циклическом изменении состояний КЭ, обеспечивающих чередование двигательного и тормозного режимов работы [7].

Законом управления процессом коммутации ключевых элементов (в дальнейшем для краткости – **закон коммутации**) в системе ИУМ–ИД будем называть временную циклическую последовательность управляющих слов, которая в общем случае обеспечивает регулируемое по величине и знаку напряжение в точках подсоединения

нагрузки (обмоток двигателя) и, как следствие, регулируемое по направлению и скорости движение вала (штока) двигателя.

К традиционным способам описания процессов коммутации КЭ ИУМ относятся временные диаграммы [7]. Основным достоинством временных диаграмм являются наглядность и возможность достаточно простого перехода к табличному, а затем к аналитическому представлению управляющих логических функций (УЛФ).

Цифровое управление двигателем постоянного тока

Мехатронный модуль на основе двигателя постоянного тока (ДПТ) показан на рис. 23. На рисунке $\Omega_{дв}$ – скорость двигателя.

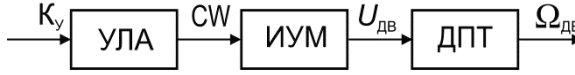


Рис. 23. Блок-схема мехатронного модуля на основе ДПТ

Схема подключения якорной обмотки ДПТ к ИУМ имеет вид, показанный на рис. 24.

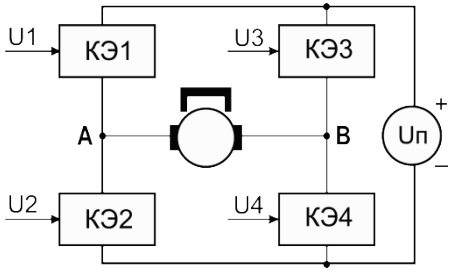


Рис. 24. Схема подключения якорной обмотки ДПТ к ИУМ

КЭ состоит из транзистора *VT*, обратного диода *VD*.

На управляющие входы ключевых элементов ИУМ поступают логические сигналы, принимающие только два значения: 0 и 1. Назовем эти сигналы **управляющими логическими функциями** – $UI(mT_{ГИ})$, где I – номер КЭ, $T_{ГИ}$ – минимальный интервал времени, определяемый генератором тактовых импульсов СМК, m = 0, 1, 2,... Нумерацию КЭ проведем таким образом, чтобы в одной стойке верх-

ний ключ имел нечетный номер ($I_В = 2n - 1$), а нижний – четный ($I_Н = 2n$), n – число стоек ИУМ. Аргумент у УЛФ в целях упрощения выражений далее будем опускать.

Совокупность всех УЛФ, которые поступают одновременно на все управляющие входы КЭ, образует CW. Таким образом, для ДПТ

$$CW = \langle U4\ U3\ U2\ U1 \rangle.$$

Из 16 располагаемых состояний 9 относятся к множеству допустимых и 7 – к множеству запрещенных состояний. Из 9 элементов множества допустимых состояний только 2 обеспечивают подключение источника питания к якорной цепи ДПТ: 0110 (t_6) и 1001 (t_9). Все остальные допустимые состояния обеспечивают отключение источника питания от нагрузки.

Временная циклическая последовательность управляющих слов определяет подключение и отключение источника питания от двигателя, что в итоге обеспечивает регулирование среднего напряжения в точках подключения нагрузки. В любом законе коммутации обязательно должно быть чередование одного из режимов подключения и отключения источника питания.

Из семи состояний, отключающих источник питания от якорной обмотки ДПТ, два состояния: 0101 или 1010 – обеспечивают режим электродинамического торможения. Все остальные (0000, 0001, 0010, 0100, 1000) реализуют режим рекуперативного или генераторного торможения.

К основным функциям УЛА относятся следующие:

1. Выделение из кода управления $К_У$ знака и абсолютного значения $|К_У|$. При этом знак будет определять направление вращения, а модуль – среднюю скорость вращения.

2. Преобразование абсолютного значения кода управления в цифровой сигнал Q, длительность которого пропорциональна этому значению кода управления. Реализация при необходимости других логических переменных.

3. Формирование управляющих логических функций.

1. Выделение из кода управления $К_У$ знака и абсолютного значения $|К_У|$

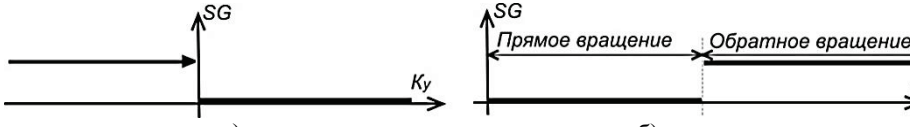
Для определенности будем полагать, что код управления представлен в обратном коде: $К_У = \langle X_{ЗН}\ X_{n-2}\ X_{n-3} \ldots X_2\ X_1\ X_0 \rangle$, тогда согласно основным соотношениям булевой алгебры:

$$|К_У| = \langle X_{ЗН} \oplus X_{n-2}\ \ X_{ЗН} \oplus X_{n-3} \ldots X_{ЗН} \oplus X_2\ \ X_{ЗН} \oplus X_1\ \ X_{ЗН} \oplus X_0 \rangle.$$

Старший (левый) разряд $К_У$ несет информацию о знаке и, как следствие, о направлении вращения двигателя, которое определяется полярностью прикладываемого напряжения $U_П$. Введем логическую переменную:

$$SG = \begin{cases} 0, \text{если } К_У \geq 0 \\ 1, \text{если } К_У < 0. \end{cases} \qquad (6)$$

Будем предполагать, что при $SG = 0$ реализуется прямое вращение, а при $SG = 1$ – обратное вращение. На рис. 25 показана графическая интерпретация выражения (6).



а) б)

Рис. 25. Графическая интерпретация переменной *SG*

Логическая переменная *SG* является функцией координат ЦСП (код управления $К_У$ несет полную информацию о текущем состоянии привода). Между тем, считывание $К_У$ происходит в строго определенные моменты времени, поэтому значение переменной *SG* синхронизировано с формированием логических переменных, являющихся функциями времени. В связи с этим в дальнейшем на временных диаграммах переменная *SG* будет иметь вид, как показано на рис. 25, *б*.

2. Преобразователи код – широтно-импульсный сигнал

При широтно-импульсном управлении ИД основным параметром, характеризующим скорость двигателя, является скважность γ –

отношение длительности импульса сигнала управления $t_И$ к периоду следования этих импульсов T (рис. 26). Значения этого параметра лежат в диапазоне $\mathbf{0 \le \gamma \le 1}$, причем (без учета момента нагрузки) при $\mathbf{\gamma = 0}$ двигатель неподвижен, при $\mathbf{\gamma = 1}$ – вращается с максимальной скоростью. Таким образом, при широтно-импульсном управлении в зависимости от параметра $\mathbf{\gamma}$ существует три принципиально различных режима работы системы ИУМ-ЭД. Временные диаграммы обычно приводят для наиболее общего случая: $\mathbf{0 < \gamma < 1}$.

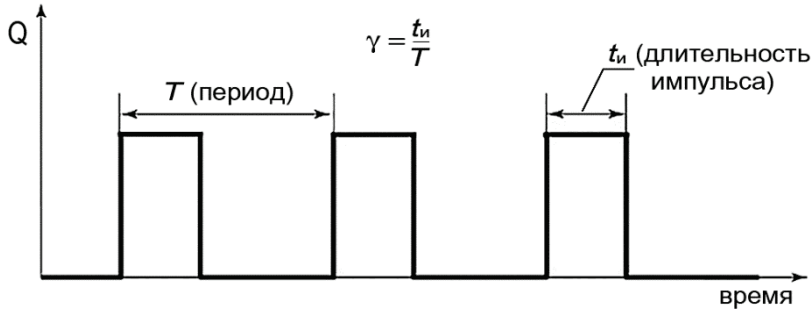


Рис. 26. Графическая интерпретация понятия «скважность»

Преобразователи код – широтно-импульсный сигнал (ПКШИС) находят применение в составе ЦСП (см. рис. 22) в качестве цифроаналоговых преобразователей и реализуются в УЛА.

При широтно-импульсном управлении регулируют длительность (временной интервал), в течение которой к двигателю либо прикладывают напряжение одной полярности или обратной, либо осуществляют динамическое или рекуперативное торможение.

Алгоритм формирования логической переменной Q (широтно-импульсного сигнала), длительность которой пропорциональна модулю кода управления $|K_У|$ на одном периоде T, основан на сравнении модуля кода управления с опорным кодом «пилообразной» формы, который формируется счетчиком (рис. 27).

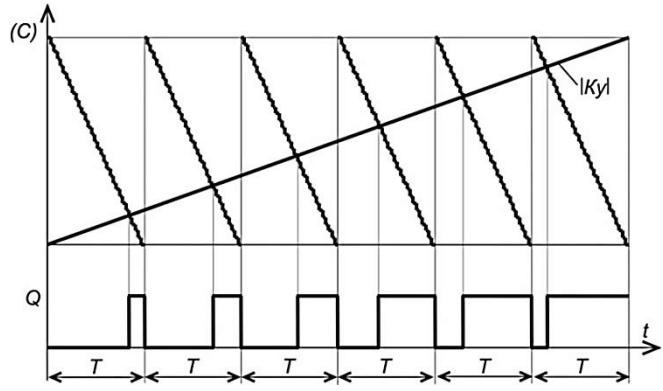


Рис. 27. Принцип формирования ШИС

В практике используется ПКШИС четырех типов (рис. 28).

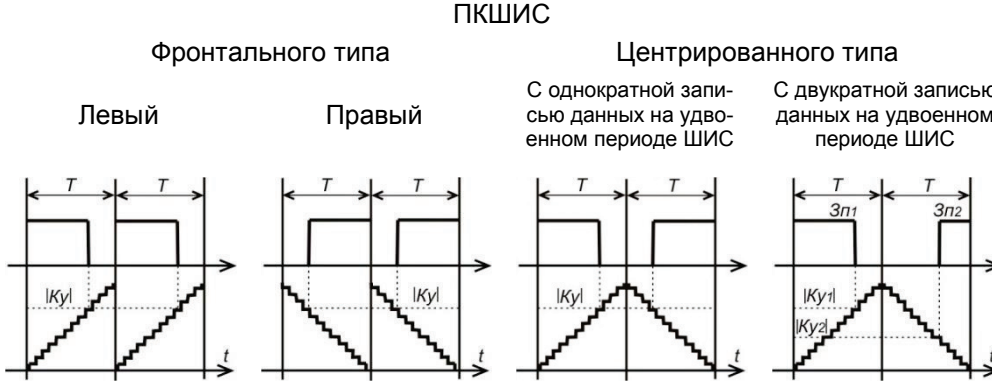


Рис. 28. Типы ШИС

ШИС фронтального типа реализуется с помощью либо суммирующих, либо вычитающих счетчиков. ПКШИС центрированного типа реализуется на основе реверсивных счетчиков.

Принцип работы ПКШИС

Рассмотрим принцип работы ПКШИС (рис. 29).

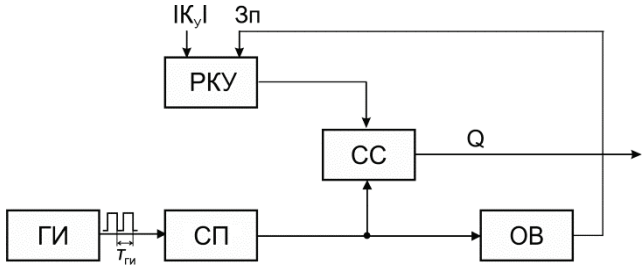


Рис. 29. Блок-схема ПКШИС

На рисунке приняты следующие обозначения: ГИ – генератор импульсов; СП – счетчик периода; РКУ – регистр кода управления; ОВ – одновибратор; СС – схема сравнения.

ГИ является времязадающим устройством и формирует бесконечную последовательность униполярных импульсов с периодом $T_{ГИ}$. РКУ обеспечивает прием сигнала записи Зп и хранение абсолютного значения кода управления $|K_У|$ в течение T. СП формирует опорный сигнал пилообразной формы и обеспечивает постоянство периода ШИС: $T = K_{СП} \cdot T_{ГИ}$, где $K_{СП}$ – модуль счета, т.е. максимальный код, который может быть подсчитан СП; $T_{ГИ}$ – интервал времени, за который содержимое СП изменяется на единицу.

ОВ формирует сигнал записи Зп один раз за период ШИС. СС обеспечивает сравнение опорного пилообразного кода и $|K_У|$ и формирование широтно-импульсного сигнала Q (7), длительность которого на одном периоде ШИС пропорциональна $|K_У|$ и реализует скважность $0 \le \gamma \le 1$.

Логическая переменная Q в случае левого ПКШИС фронтального типа может быть выражена как

$$Q = \begin{cases} 1, \text{если } (k-1)T \le mT_{ГИ}, \le (k-1)T + \left|K_У\right| \cdot T_{ГИ}; \\ 0, \text{если } (k-1)T + \left|K_У\right| \cdot T_{ГИ}, \le mT_{ГИ} \le K \cdot T, \end{cases} \quad (7)$$

где $m = m_1 + (k-1) \cdot K_{СП}$; $m_1 = 1,\ 2,\ 3 \ldots K_{СП}$; k =1, 2, 3 – номер периода.

Управляющие логические функции для закона симметричной коммутации

При законе симметричной коммутации осуществляется торможение противовключением. При этом к двигателю прикладывается напряжение:

$$U_{\text{ДВ}} = \begin{cases} +U_{\text{П}}, \text{ если } U4 \cdot \overline{U3} \cdot \overline{U2} \cdot U1 = 1; \\ -U_{\text{П}}, \text{ если } \overline{U4} \cdot U3 \cdot U2 \cdot \overline{U1} = 1. \end{cases} \quad (8)$$

На рис. 30 приведены временные диаграммы $U_{\text{ДВ}}$, логических переменных *SG*, *Q* и управляющих логических функций *U*4...*U*1.

Табличные значения УЛФ *U*4...*U*1, соответствующие временной диаграмме (рис. 30), представлены в табл. 10.

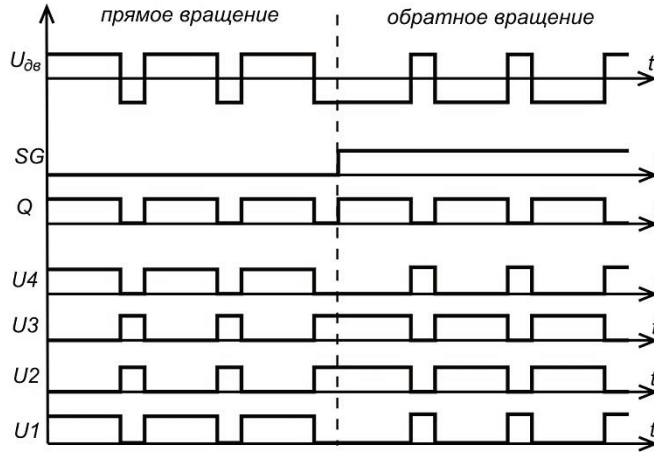


Рис. 30. Временные диаграммы для закона симметричной коммутации

Таблица 10

i	*SG*	*Q*	*U*4	*U*3	*U*2	*U*1	T^2
0	0	0	0	1	1	0	t_6
1	0	1	1	0	0	1	t_9
2	1	0	1	0	0	1	t_9
3	1	1	0	1	1	0	t_6

Минимальные аналитические выражения УЛФ закона симметричной коммутации могут быть получены непосредственно по табл. 10:

$$U1 = U4 = SG \oplus Q;$$
$$U2 = U3 = \overline{SG \oplus Q}. \quad (9)$$

При законе симметричной коммутации отсутствует состояние, соответствующее нулевому сигналу управления: все четыре КЭ переключаются один раз за период широтно-импульсного сигнала при движении в любом из двух направлений при любом сигнале управления. Во многих источниках, в том числе в [7], отмечено, что закон симметричной коммутации энергетически наименее выгоден.

Управляющие логические функции для закона несимметричной коммутации

Закон несимметричной коммутации ключевыми элементами реализует электродинамическое торможение. При этом имеется два альтернативных варианта режима электродинамического торможения: верхними КЭ либо нижними КЭ.

При использовании для электродинамического торможения верхней пары КЭ к двигателю прикладывается напряжение:

$$U_{\text{ДВ}} = \begin{cases} +U_{\text{П}}, \text{ если } U4 \cdot \overline{U3} \cdot \overline{U2} \cdot U1 = 1 \\ 0, \text{если } \overline{U4} \cdot U3 \cdot \overline{U2} \cdot U1 = 1 \\ -U_{\text{П}}, \text{если } \overline{U4} \cdot U3 \cdot U2 \cdot \overline{U1} = 1 \end{cases}. \quad (10)$$

На рис. 31 приведены временные диаграммы $U_{\text{ДВ}}$, логических переменных *SG*, *Q* и управляющих логических функций *U*4...*U*1 для рассматриваемого случая.

Табличное представление УЛФ *U*4...*U*1 для случая электродинамического торможения верхними КЭ, как показано на рис. 30, приведено в табл. 11.

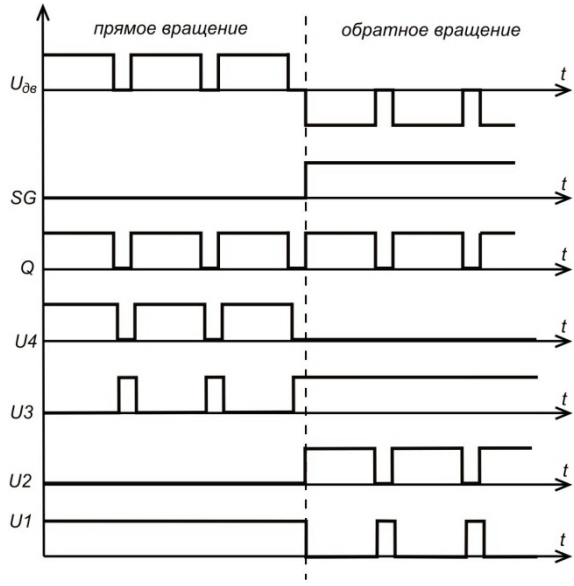


Рис. 31. Временная диаграмма для закона несимметричной коммутации

Таблица 11

i	SG	Q	$U4$	$U3$	$U2$	$U1$	T^2
0	0	0	0	1	0	1	t_5
1	0	1	1	0	0	1	t_9
2	1	0	0	1	0	1	t_5
3	1	1	0	1	1	0	t_6

Минимальные аналитические выражения УЛФ для закона несимметричной коммутации могут быть получены непосредственно по табл. 11:

$$U1 = \overline{SG \cdot Q};\ U2 = \overline{U1};$$
$$U3 = \overline{\overline{SG} \cdot Q}\ ;\ U4 = \overline{U3}. \qquad (11)$$

Наиболее перспективным является реализация УЛА на основе программируемых логических интегральных схем либо базовых матричных кристаллов.

Импульсный усилитель мощности

Схему ИУМ возможно реализовать различными способами. Наибольшее распространение получили мостовые схемы на биполярных или полевых транзисторах. Последние на сегодняшний день применяются для электродвигателей мощностью до 1 кВт.

Для согласования низковольтных логических управляющих сигналов с высоковольтными сигналами управления затворов полевых транзисторов требуются промежуточные устройства согласования – драйверы. В данной работе используются полумостовые драйверы (рис. 32). Один драйвер управляет одной стойкой мостового ИУМ. Драйвер имеет один управляющий вход – *IN*, на который поступает логический сигнал управления верхним КЭ стойки. Выходными сигналами драйвера нелогического уровня являются сигналы *HO* и *LO*, управляющие соответственно верхним и нижним КЭ стойки. Фактически на входы драйверов ИУМ поступает двухразрядное управляющее слово $CW = <U3\ U1>$, а на выходе драйверов формируется уже четырехразрядное слово $LO_2HO_2LO_1HO_1$, эквивалентное $CW = <\overline{U}3\ U3\ \overline{U}1\ U1>$.

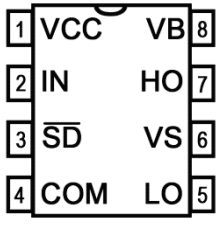


Рис. 32. Назначение выводов полумостового драйвера

Большинство современных полумостовых драйверов (рис. 32) работает таким образом, что при подаче на логический вход *IN* драйвера сигнала высокого логического уровня (3,3 В) выход драйвера, управляющий верхним транзистором *HO*, приводит его в открытое состояние, а выход драйвера, управляющий нижним ключом *LO*, закрывает его. При подаче на логической вход сигнала низкого уровня (0 В) – наоборот. Питание драйвера подводится через вывод *VCC*. Вывод драйвера *COM* соединяется с отрицательным полюсом источника питания драйвера. Через этот вывод разряжается емкость затвора ниж-

него транзистора (рис. 33), когда он должен закрыться. Вывод драйвера *VB* соединяется с конденсатором вольтдобавки и катодом диода цепи вольтдобавки. Через этот вывод замыкается цепь заряда затворной емкости верхнего транзистора стойки, когда нижний транзистор закрыт, а верхний открыт. Вывод *VS* соединяется со средней точкой моста и противоположной обкладкой конденсатора. Через этот вывод разряжается затворная емкость верхнего транзистора, когда он должен закрыться. Логический сигнал $\overline{SD}$ прерывает работу драйвера при подаче на этот вход сигнала низкого уровня и разрешает работу при подаче сигнала высокого уровня.

Драйверы не могут работать без элементов цепи вольтдобавки или так называемой «обвязки». В обвязку драйверов входят такие элементы, как диоды, резисторы и конденсаторы цепи вольтдобавки. В общем случае они необходимы для открытия верхних транзисторов моста. Драйвер работает таким образом, что когда нижний транзистор стойки открыт, а верхний закрыт, происходит заряд конденсатора вольтдобавки через нижний транзистор стойки по цепи: $U_{\text{пит}}$ драйвера, *R5, VD5, C1, VT2*, $-U_{\text{пит}}$ драйвера (рис. 33). Когда нижний транзистор закрыт, то при условии, что напряжения заряженного конденсатора достаточно, открывается верхний транзистор, а диод *VD5* оказывается запертым потенциалом питания силовой схемы и заряженного конденсатора и схема управления верхним плечом питается исключительно разрядным током конденсатора *C*. Когда верхний транзистор открыт, к его затвору прикладывается сумма напряжения заряженного конденсатора и напряжения питания силовой части схемы, а к истоку – только напряжение питания силовой части, следовательно, суммарное напряжение между затвором и истоком есть напряжение заряженного конденсатора. Таким образом, конденсатор вольтдобавки необходим для создания разности напряжения на переходе затвор – исток.

Электрическая схема силовой части системы управления показана на рис. 33.

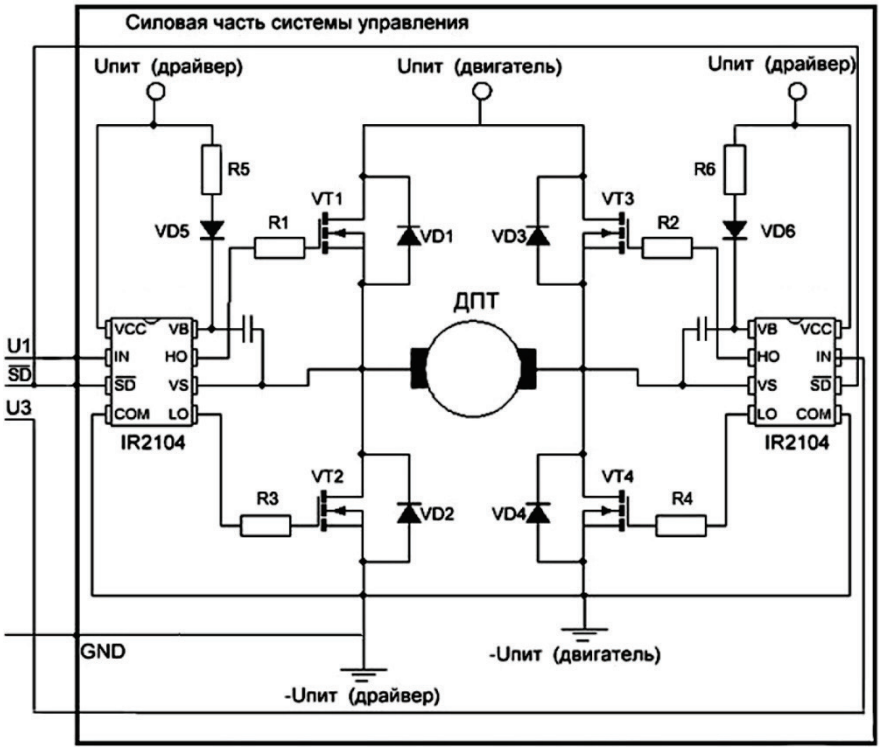


Рис. 33. Электрическая схема силовой части системы управления

На рисунке приняты следующие обозначения:

R5, R6 – резисторы, ограничивающие ток заряда конденсаторов вольтдобавки (производители рекомендуют осуществлять выбор их номиналов в пределах 1–10 Ом), не являются строго рекомендуемыми элементами к монтажу в стандартных системах, работающих по принципу ШИМ;

C1 и C2 – конденсаторы вольтдобавки, отпирают своим потенциалом полевой транзистор, являются необходимыми элементами схемы;

R1–R4 – резисторы, ограничивающие ток заряда затвора транзисторов. Драйверы способны ограничивать свой ток самостоятельно, добавочные сопротивления служат для оптимизации времени переключения и не являются строго рекомендуемыми элементами к монтажу;

VD5–VD6 – диоды вольтдобавки, запирают источник логического напряжения в период открытия верхних транзисторов стойки, являются необходимыми элементами схемы;

VD1–VD4 – обратные диоды.

Достоинства этой схемы заключаются в ее экономичности. Благодаря очень высокому входному сопротивлению, цепь полевых транзисторов расходует крайне мало энергии, так как практически не потребляет входного тока. Усиление по току у полевых транзисторов намного выше, чем у биполярных. Значительно выше помехоустойчивость и надежность работы схемы, поскольку из-за отсутствия тока через затвор транзистора управляющая цепь со стороны затвора изолирована от выходной цепи со стороны стока и истока. У полевых транзисторов на порядок выше скорость перехода, чем у биполярных, между состояниями проводимости и непроводимости тока. Поэтому они могут работать на более высоких частотах, чем биполярные. Более того, полумостовые драйверы, входящие в схему, кроме основных функций формирования управляющего сигнала, обеспечивают и токовую защиту (защиту от короткого замыкания), защиту от пробоя, перенапряжения, оптимизируют скорость открытия транзисторов.

Практическая часть

Методика выполнения лабораторной работы

1. Ввод проекта.

Средствами языка описания аппаратуры VHDL в среде Quartus II v. 9.1 создать устройство – управляющий логический автомат (УЛА), состоящий из делителя частоты, преобразователя код – широтно-импульсный сигнал, а именно счетчик и компаратор для сравнения значений счетчика с входными значениями управляющего сигнала. Управляющий сигнал задавать с помощью тумблеров платы DE2.

2. Компиляция проекта.

Откомпилировать работу проекта. Устранить ошибки и предупреждения компилятора.

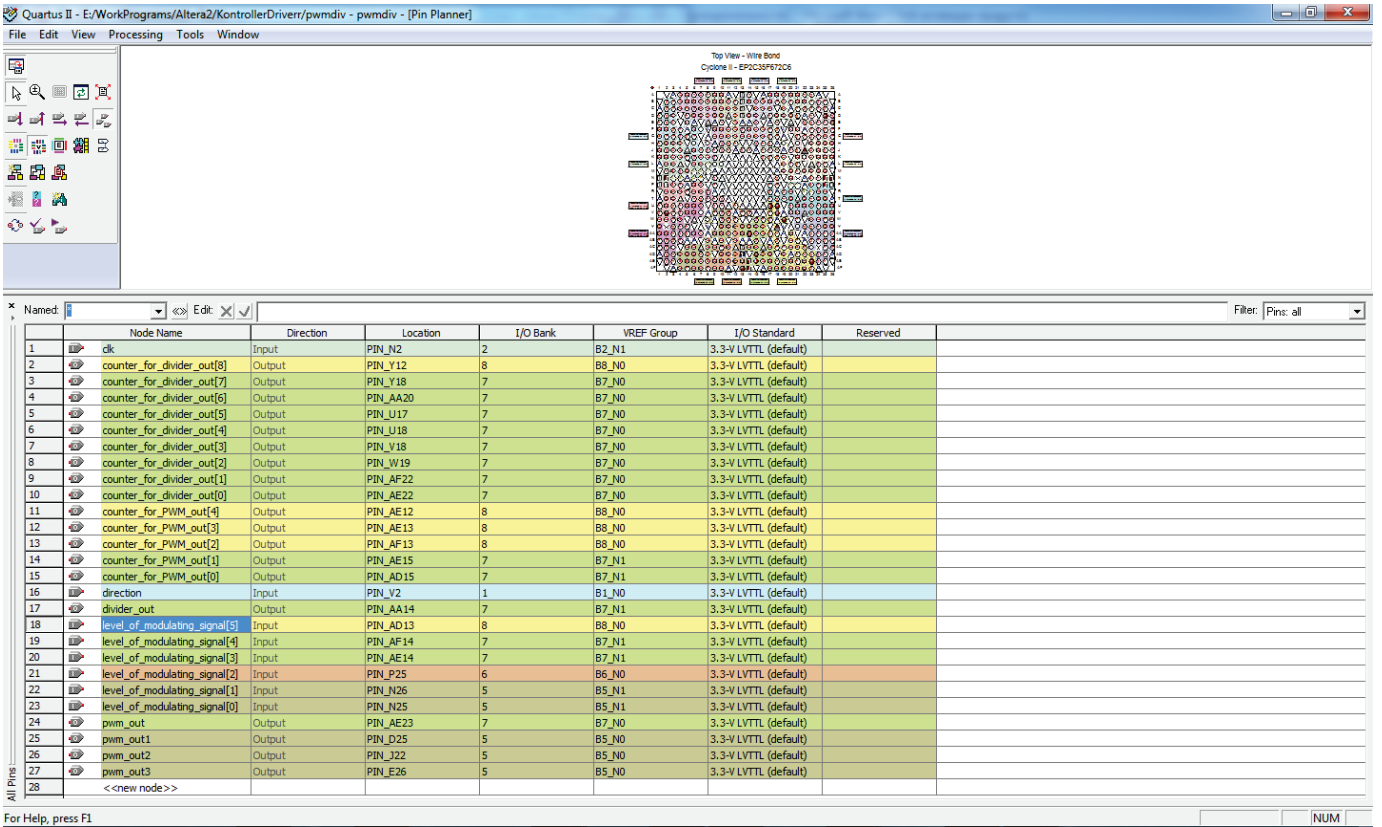
3. Назначение выводов ПЛИС.

Перед загрузкой проекта в ПЛИС нужно назначить всем необходимым входным и выходным сигналам устройства номера их выводов на ПЛИС. Для этого следует сопоставить сигналы из раздела программы *entity* с конкретными физическими выводами микросхемы. Это делается при помощи команды *Assignments=>Pins*. В появившемся окне *Pin Planner* для выводов можно назначить конкретные локации ПЛИС (рис. 34).

В соответствии с приведенным примером кода в проекте будут использованы семь переключателей-тумблеров: один для подачи сигнала изменения направления вращения, пять для задания уровня модулирующего сигнала, один для разрешения работы драйверов; три вывода слота расширения: два для подачи сигнала управления драйверам и один для разрешения работы; и некоторое количество светодиодных индикаторов, к которым будут «привязаны» остальные сигналы из раздела описания внешнего интерфейса модуля. Оставшиеся сигналы, указанные в разделе с описанием внешних характеристик и интерфейса модуля, дублируют сигналы разрядов внутренних регистров, участвующих в работе проекта, и объявлены в этом разделе только для возможности временной симуляции, но и им в соответствие должен быть поставлен физический вывод платы, или компилятор выберет вывод по своему усмотрению, что может нарушить работу всего проекта.

Для управления направлением вращения (сигнал direction) можно использовать тумблер SW17, в ПЛИС номер его локации – PIN_V2. Назначим локацию этого тумблера сигналу direction, для этого выберем ее непосредственно в столбце Location напротив названия сигнала. На рис. 35 выбранный тумблер выделен зеленым цветом.

Запись двоичного числа в регистр модулирующего сигнала (сигналы *level_of_modulating_signal*[0] – *level_of_modulating_signal*[4]) можно осуществлять тумблерами SW0–SW4, в ПЛИС номера локаций этих тумблеров – PIN_N25, PIN_N26, PIN_P25, PIN_AE14, PIN_AF14 соответственно. Назначим необходимым сигналам регистра локации выбранных тумблеров. На рис. 35 эти тумблеры выделены красным цветом.



	Node Name	Direction	Location	I/O Bank	VREF Group	I/O Standard	Reserved
1	clk	Input	PIN_N2	2	B2_N1	3.3-V LVTTL (default)	
2	counter_for_divider_out[8]	Output	PIN_Y12	8	B8_N0	3.3-V LVTTL (default)	
3	counter_for_divider_out[7]	Output	PIN_Y18	7	B7_N0	3.3-V LVTTL (default)	
4	counter_for_divider_out[6]	Output	PIN_AA20	7	B7_N0	3.3-V LVTTL (default)	
5	counter_for_divider_out[5]	Output	PIN_U17	7	B7_N0	3.3-V LVTTL (default)	
6	counter_for_divider_out[4]	Output	PIN_U18	7	B7_N0	3.3-V LVTTL (default)	
7	counter_for_divider_out[3]	Output	PIN_V18	7	B7_N0	3.3-V LVTTL (default)	
8	counter_for_divider_out[2]	Output	PIN_W19	7	B7_N0	3.3-V LVTTL (default)	
9	counter_for_divider_out[1]	Output	PIN_AF22	7	B7_N0	3.3-V LVTTL (default)	
10	counter_for_divider_out[0]	Output	PIN_AE22	7	B7_N0	3.3-V LVTTL (default)	
11	counter_for_PWM_out[4]	Output	PIN_AE12	8	B8_N0	3.3-V LVTTL (default)	
12	counter_for_PWM_out[3]	Output	PIN_AE13	8	B8_N0	3.3-V LVTTL (default)	
13	counter_for_PWM_out[2]	Output	PIN_AF13	8	B8_N0	3.3-V LVTTL (default)	
14	counter_for_PWM_out[1]	Output	PIN_AE15	7	B7_N1	3.3-V LVTTL (default)	
15	counter_for_PWM_out[0]	Output	PIN_AD15	7	B7_N1	3.3-V LVTTL (default)	
16	direction	Input	PIN_V2	1	B1_N0	3.3-V LVTTL (default)	
17	divider_out	Output	PIN_AA14	7	B7_N1	3.3-V LVTTL (default)	
18	level_of_modulating_signal[5]	Input	PIN_AD13	8	B8_N0	3.3-V LVTTL (default)	
19	level_of_modulating_signal[4]	Input	PIN_AF14	7	B7_N1	3.3-V LVTTL (default)	
20	level_of_modulating_signal[3]	Input	PIN_AE14	7	B7_N1	3.3-V LVTTL (default)	
21	level_of_modulating_signal[2]	Input	PIN_P25	6	B6_N0	3.3-V LVTTL (default)	
22	level_of_modulating_signal[1]	Input	PIN_N26	5	B5_N1	3.3-V LVTTL (default)	
23	level_of_modulating_signal[0]	Input	PIN_N25	5	B5_N1	3.3-V LVTTL (default)	
24	pwm_out	Output	PIN_AE23	7	B7_N0	3.3-V LVTTL (default)	
25	pwm_out1	Output	PIN_D25	5	B5_N0	3.3-V LVTTL (default)	
26	pwm_out2	Output	PIN_J22	5	B5_N0	3.3-V LVTTL (default)	
27	pwm_out3	Output	PIN_E26	5	B5_N0	3.3-V LVTTL (default)	
28	<<new node>>						

Рис. 34. Планировка выводов

Для разрешения вывода сигналов непосредственно из ПЛИС (сигнал *level_of_modulating_signal*[5]) можно использовать тумблер SW5, в ПЛИС номер его локации – PIN_AD13, на рис. 35 выделен белым цветом.

В качестве генератора импульсов (сигнал *clk*) использован встроенный 50 МГц осциллятор, номер его локации в ПЛИС – PIN_N2.

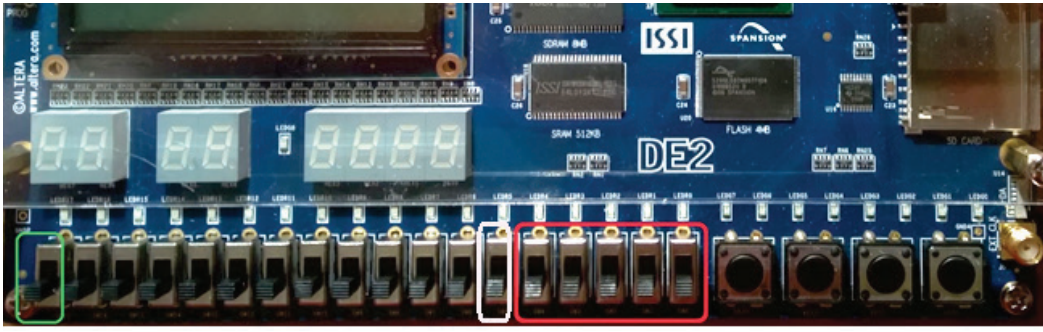
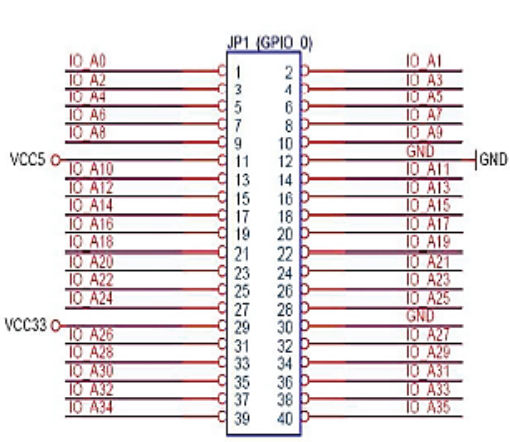


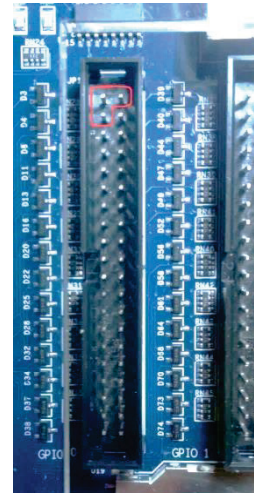
Рис. 35. Тумблеры, используемые при эксперименте

Управляющим сигналам драйверов *pwm_out1*(*U*1) и *pwm_out2*(*U*3) назначим выводы PIN_D25 и PIN_J22 разъема расширения GPIO 0 платы DE2. Порядковые номера этих выводов на рис. 36, *а* – 1, 2 соответственно. Входы *IN* обоих драйверов подключаются к данным выводам. Разрешающему работу драйверов сигналу *pwm_out3* ($\overline{\text{SD}}$) назначим вывод этого же разъема расширения PIN_E26. Порядковый номер этого вывода на рис. 36, *а* – 3. Понятно, что входы обоих драйверов $\overline{\text{SD}}$ подключаются к этому выводу. Для контроля сигналов используется вывод этого же разъема GND, на схеме ему присвоен номер 30 (рис. 36, *а*).

Для проверки работоспособности программы ряд сигналов выведен на светодиоды, номера локаций которых приведены в приложении 2 [8].



а)



б)

Рис. 36. Разъем расширения платы:
а – схематическое изображение выводов разъема расширения GPIO 0;
б – фото разъема GPIO 0 и разъема GPIO 1 на плате Altera DE2 (красным цветом выделены необходимые выводы)

4. Моделирование.

Промоделировать работу УЛА с получением временных диаграмм сигналов.

5. Размещение элементов на макетной плате и сопряжение с ДПТ.

Собрать схему эксперимента в соответствии с рис. 37. При этом подключить цифровой осциллограф с помощью зажимов.

6. Загрузка проекта в ПЛИС, проверка работоспособности системы.

Ввести программу, откомпилировать, загрузить в ПЛИС с помощью режима *programmer*.

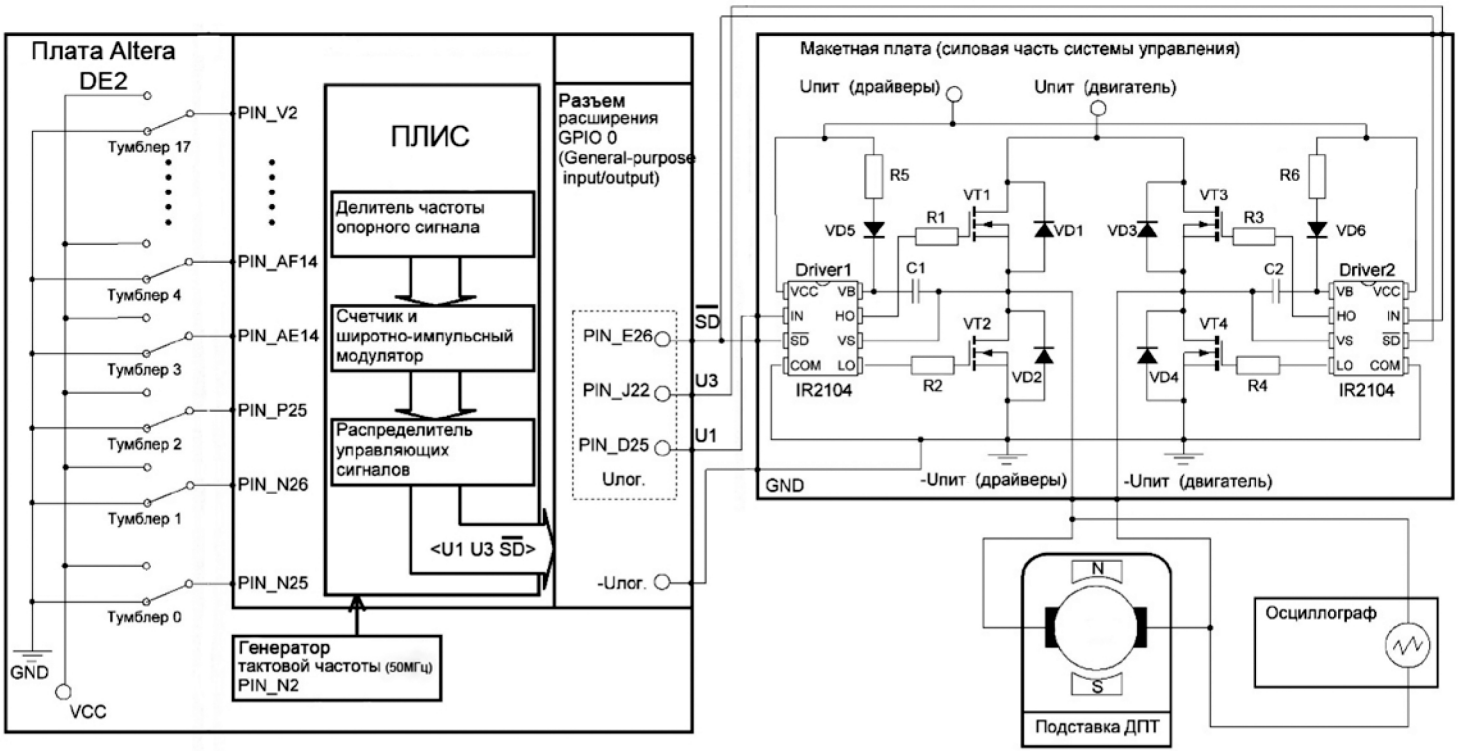


Рис. 37. Схема эксперимента

7. Проверить работу системы, переключая тумблеры, отвечающие за абсолютное значение кода управления и направление вращения. Изменяя входной код с помощью тумблеров-переключателей SW0-SW4 (младший разряд SW0), скопировать с экрана осциллографа напряжение, прикладываемое к ДПТ, для скважностей из диапазона: $0 \le \gamma \le 1$ (задается преподавателем).

8. Изменяя входной код с помощью тумблеров-переключателей SW0-SW4, для каждого кода с помощью бесконтактного фототахометра определить скорость двигателя и заполнить табл. 12. По таблице построить регулировочную характеристику $\Omega_{ДВ}=f(\gamma)$.

Таблица 12

№ п/п	Входной код	Скорость двигателя

9. При скважности $\gamma=0,75$ изменить состояние SW17. Визуально удостовериться в факте реверсирования (изменении направления вращения вала двигателя).

Пример программы, реализующий логический автомат, управляющий скоростью ДПТ, приведен ниже.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;

entity pwmdiv is

port (
 -- тактовый сигнал
clk                :in    std_logic;
 -- величина модулирующего сигнала
```

```
level_of_modulating_signal :in std_logic_vector (5 downto 0);
 -- направление вращения вала
direction                  :in    std_logic;
 -- сигнал делителя частоты
divider_out                :out   std_logic;
 -- регистр делителя частоты
counter_for_divider_out    :out   std_logic_vector (8 downto 0);
 -- регистр ШИ-модулятора
counter_for_pwm_out        :out   std_logic_vector (4 downto 0);
 -- выход ШИМ
pwm_out                    :out   std_logic;
 -- вход первого драйвера
pwm_out1                   :out   std_logic;
 -- вход второго драйвера
pwm_out2                   :out   std_logic;
 -- разрешение работы обоих драйверов
pwm_out3                   :out   std_logic
);
end pwmdiv;

architecture behavioral of pwmdiv is
signal counter_for_divider :std_logic_vector (8 downto 0)
:="000000000";
signal divider             :std_logic := '0';
signal pwm                 :std_logic := '0';
signal counter_for_pwm     :std_logic_vector (4 downto 0)
:="00000";

begin

process(clk)
begin
if rising_edge (clk) then
     if counter_for_divider = "111111111" then
          counter_for_divider <= (others => '0');
     else
          counter_for_divider <= counter_for_divider+1;
     end if;
```

```
end if;
end process;

process(divider)
begin
if rising_edge(divider) then
     if counter_for_pwm = "11110" then
             counter_for_pwm <= (others => '0');
     else
             counter_for_pwm <= counter_for_pwm+1;
     end if;
end if;
end process;

pwm <= '1'   when
     level_of_modulating_signal(4 downto 0) > counter_for_pwm
             else   '0';

divider <= '1'      when
     counter_for_divider(8 downto 0) < "100000000"
             else   '0';

process(direction, level_of_modulating_signal, pwm)
begin
pwm_out1 <= direction and level_of_modulating_signal(5)
and pwm;
pwm_out2 <= (not direction) and level_of_modulating_signal(5)
and pwm;
pwm_out3 <= level_of_modulating_signal(5);
end process;

pwm_out                   <= pwm;
counter_for_divider_out   <= counter_for_divider;
divider_out               <= divider;
counter_for_pwm_out       <= counter_for_pwm;

end behavioral;
```

Рекомендации по выполнению лабораторной работы

В качестве генератора тактовых сигналов предлагается использовать встроенный 50 МГц осциллятор.

Следует отметить, что для управления маломощными ДПТ частота управляющего ШИМ-сигнала выбирается обычно не более 10 КГц, а исходя из характеристик используемой элементной базы, рекомендуется понизить частоту ШИМ до 3 КГц. Для соблюдения данного условия нужно ввести в проект, помимо ПКШИС, также делитель опорной частоты ПЛИС. Частота управляющего сигнала ШИМ зависит от модуля счета делителя частоты, а также модуля счета самого ПКШИС.

В коде программы рассмотрен вариант построения делителя частоты на базе 9-разрядного счетчика по модулю 512, а счетчика ШИМ – на базе 5-разрядного по модулю 31. Сигнал, последовательно прошедший через делитель и счетчик ШИМ, уменьшит свою частоту в $512 \cdot 31 = 15872$ раза. Следовательно, результирующая частота будет равна 3,15 КГц, что удовлетворяет ограничениям элементной базы и типу двигателя.

Для задания кода управления в виде двоичного числа возможно использовать 6-разрядный регистр, старший разряд которого реализует функцию сигнала SD и не меняет значения скважности γ.

Также необходимо описать в коде три выходных сигнала, управляющих работой полумостовых драйверов.

Контрольные вопросы

1. *В чем суть импульсного метода управления ДПТ?*
2. *Какие допустимые состояния обеспечивают подключение источника напряжения к обмоткам двигателя постоянного тока?*
3. *Расскажите о принципе работы ПКШИС фронтального типа.*

4. *В чем разница между ПКШИС фронтального и центрированного типов?*
5. *Получите управляющие логические функции для закона симметричной коммутации при управлении ДПТ.*
6. *Получите управляющие логические функции для закона несимметричной коммутации при управлении ДПТ.*
7. *Нарисуйте эпюру напряжения на двигателе для закона симметричной коммутации при прямом и обратном вращении.*
8. *Нарисуйте эпюру напряжения на двигателе для закона несимметричной коммутации при прямом и обратном вращении.*
9. *Что означают термины «управляющая логическая функция» и «управляющее слово»?*
10. *Каково назначение полумостовых драйверов?*

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Алексенко А.Г. Основы микросхемотехники. – М.: Юнимедстайл, 2002. – 448 с.

2. Андреев В.П., Волович Н.В., Глебов В.М. и др. Проектирование и испытание бортовых систем управления: Учебное пособие / Под ред. А.С. Сырова. – М.: Изд-во МАИ-ПРИНТ, 2011. – 344 с.

3. Бусурин В.И., Можаев В.А., Шеленков В.М. Микроэлектронные устройства систем управления летательных аппаратов: Учебное пособие / Под ред. В.И. Бусурина. – М.: Изд-во МАИ-ПРИНТ, 2011. – 200 с.

4. ГОСТ 17021-88. Микросхемы интегральные. Термины и определения. – М.: Изд-во стандартов, 1995. – 12 с.

5. ГОСТ 2.743-91. Обозначения условные графические в схемах. Элементы цифровой техники. – М.: Изд-во стандартов, 1995. – 44 с.

6. ГОСТ 2.755-87. Обозначения условные графические в схемах. Устройства коммутационные и контактные соединения. – М.: Изд-во стандартов, 1995. – 11 с.

7. Попов Б.Н. Цифровые устройства систем приводов летательных аппаратов: Учебное пособие. – М.: Изд-во МАИ-ПРИНТ, 2008. – 124 с.

8. Попов Б.Н., Цалкова Е.Э., Ханин Д.Н. Лабораторный практикум по цифровым схемам и устройствам бортовой автоматики: Учебное пособие к практическим занятиям и лабораторным работам по курсу «Цифровая микроэлектроника». – М.: МОКБ «Марс», 2014. – 92 с.

9. Порешин П.П., Попов Б.Н. Дискретная математика: множества, отношения, логика, автоматы: Учебное пособие. – М.: Изд-во МАИ-ПРИНТ, 2014. – 188 с.

10. Прянишников В.А. Электроника: Полный курс лекций. – 4-е изд. – СПб.: КОРОНА ПРИНТ, 2004. – 416 с.

11. Суворова Е.А., Шейнин Ю.Е. Проектирование цифровых систем на VHDL. – СПб.: БХВ-Петербург, 2003. – 576 с.

12. Угрюмов Е.П. Цифровая схемотехника: Учебное пособие. – 2-е изд. – СПб.: БХВ-Петербург, 2007. – 800 с.

13. Altera DE-2 Development and Education Board. User Manual. – Altera Corporation, 2012.

Тем. план 2016

Попов Борис Николаевич,
Цалкова Елизавета Эдуардовна,
Ханин Дмитрий Николаевич

ПРОЕКТИРОВАНИЕ
ЦИФРОВЫХ И ЦИФРОАНАЛОГОВЫХ УСТРОЙСТВ
БОРТОВОЙ АВТОМАТИКИ.

ЛАБОРАТОРНЫЙ ПРАКТИКУМ

Редакторы: М.С. Винниченко, Т.В. Кособокова
Компьютерная верстка Е.Э. Качаловой

Подписано в печать 19.08.2016
Бум. офсетная. Формат 60x84 1/16. Печать на ризографе.
Усл. печ. л. 4,65. Уч.-изд. л. 5,00. Тираж 100 экз.
Изд. № 426. Заказ 35.

Издательство МАИ
(МАИ), Волоколамское ш., д. 4, Москва, А-80, ГСП-3 125993

Отпечатано в МОКБ «Марс»
127473, г. Москва, 1-й Щемиловский переулок, 16