УДК 681.335 (07)

МОДЕЛИРОВАНИЕ АЦП ПОРАЗРЯДНОГО УРАВНОВЕШИВАНИЯ

© А.Ю. Потлов, Д.А. Семенов, О.А. Остапенко,
О.О. Голубятников, Е.В. Власова, Е.И. Глинкин

Ключевые слова: аналогово-цифровые преобразователи; программная среда LabVIEW; таблица состояний; матричные схемы; временные диаграммы; метод аналогии.

Систематизированны методы синтеза и анализа БИС в технологию проектирования АЦП поразрядного уравновешивания в программной среде LabVIEW.

1. СТРУКТУРА АЦП

АЦП поразрядного уравновешивания относятся к преобразователям параллельно-последовательного действия, их отличает высокая коммуникабельность и оперативность амплитудно-дискретной обработки информации [1].

Сущность способов поразрядного уравновешивания заключается в непосредственном представлении амплитуды в код с взвешенными основаниями число-импульсной последовательности. За период формирования последовательности количество знакомест импульсов организуют соответственно числу позиций оператора счисления, включающего оценку по операторам исчисления уровня исследуемого сигнала с интегралом эквивалентных мер для выявления значимости знакоместа. При положительной оценке формируют на адресе знакоместа импульс в виде потенциала высокого уровня, принимаемого за логическую единицу, в противном случае на адресуемом интервале инициируют потенциалом низкого уровня логический нуль.

АЦП поразрядного уравновешивания синтезируют по принципу аналогии из преобразователей последовательного отсчета, организованных по кольцевой структуре канала регулирования из последовательного соединения компаратора (К), счетчика и цифроаналогового преобразователя (ЦАП). Повышают гибкость структуры заменой счетчика (С) с аппаратным управлением на регистр (Р) и знакогенератор (ЗГ) на ПЗУ с упорядоченной архитектурой программируемой матрицы (рис. 1).

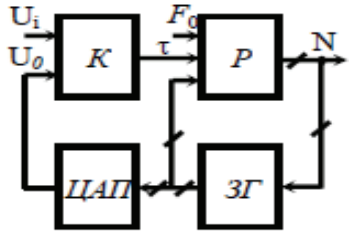


Рис. 1. Структура АЦП поразрядного уравновешивания

Проанализирован АЦП поразрядного уравновешивания на уровне функциональной схемы, поясняющей физику работы для моделирования в среде LabVIEW его компонент схемо- и мнемотехники, математического обеспечения и метрологических средств.

2. ПРОГРАММНАЯ СРЕДА LABVIEW

Для моделирования АЦП поразрядного уравновешивания была выбрана среда графического программирования LabVIEW (Laboratory Virtual Instrument Workbench – среда разработки лабораторных виртуальных приборов). LabVIEW используется в системах сбора и обработки данных, а также для управления техническими объектами и технологическими процессами. Идеологически LabVIEW очень близка к системам диспетчерского управления и сбора данных (SCADA-системам), но в отличие от них в большей степени ориентирована на решение задач не столько в области автоматизированных систем управления технологическим процессом (АСУ ТП), сколько в области автоматизированных систем научных исследований (АСНИ).

LabVIEW имеет обширные библиотеки функций для решения различных задач: ввод/вывод, обработка, анализ и визуализация сигналов; контроль и управление технологическими объектами; статистический анализ и комплексные вычисления и др.

3. СТРУКТУРА ПРОГРАММЫ АЦП

Структура включает в себя 9 файлов с расширением .vi (виртуальные приборы). Верхним ее уровнем является «Главный файл». Именно его нужно открывать для запуска всей программы. Этот файл находится в тесном взаимодействии с файлами «АЦП» и «ПЗУ АЦП» (рис. 2). При работе программы они непрерывно обмениваются информацией друг с другом.

Файлы «АЦП» и «ПЗУ АЦП», в свою очередь, находятся в непрерывном обмене информацией с починенными файлами. Файл «АЦП» отвечает за моделирование работы АЦП без использования ПЗУ. С файлом «АЦП» (рис. 3) в непрерывном информационном обмене находятся файлы «Конвертирование 10 в 2», «Модель черного ящика», «Временные диаграммы», «Матричная

Рис. 2. Иерархия всей программы

схема». Названия этих файлов отражают их назначение. Файл «Матричная схема помимо обмена информацией с файлом «АЦП» еще обменивается ей с файлом «Стрелка». Они служат для построения матричной схемы по ГОСТ.

Файл «ПЗУ АЦП» (рис. 4) отвечает за моделирование работы АЦП на ПЗУ. С файлом «ПЗУ АЦП» в непрерывном информационном обмене находятся файлы «Генератор комбинаций ПЗУ», «Конвертирование 10 в 2», «Модель черного ящика», «Временные диаграммы», «Матричная схема». Названия этих файлов отражают их назначение. Файл «Матричная схема ПЗУ» помимо обмена информацией с файлом «ПЗУ АЦП» еще обменивается ей с файлом «Стрелка» (рис. 3). Они служат для построения матричной схемы по ГОСТ. Файл «Генератор комбинаций ПЗУ» также обменивается информацией с файлом «Конвертирование 10 в 2», это необходимо для его корректной работы.

Подробно рассмотрим «Генератор комбинаций ПЗУ». Он выполнен в виде отдельного устройства (виртуального прибора).

3.1. ВИРТУАЛЬНЫЙ ПРИБОР

Работа виртуального прибора регулируется параметром под названием «Размер». Формируем 5 массивов:

– одномерный массив «Такт». В нем хранятся значения тактового сигнала, т. е. чередующиеся нули и единицы;

– двумерный массив «Входы». В нем хранятся значения подающихся на вход сигналов;

– двумерный массив «Выходы». В нем хранятся значения сигналов на выходе;

– одномерный массив «Проверка». Содержит такую же информацию что и массив «Выходы», только в 10-м коде;

– одномерный массив «Заголовки». Строковый массив, содержащий в себе заглавие таблицы.

Информация из этих 5 массивов при правильном объединении дает таблицу состояний ПЗУ.

Рассмотрим принцип работы данного виртуального прибора.

Так как в ПЗУ вначале программируется нулевая комбинация и лишь потом все остальные, то очевидно, что в генераторе комбинаций удобнее выделить 3 части:

1) часть, которая генерирует матрицу ПЗУ размером 0;

2) часть, которая генерирует матрицу ПЗУ размером 1;

3) часть, которая генерирует любую матрицу ПЗУ размером более одного.

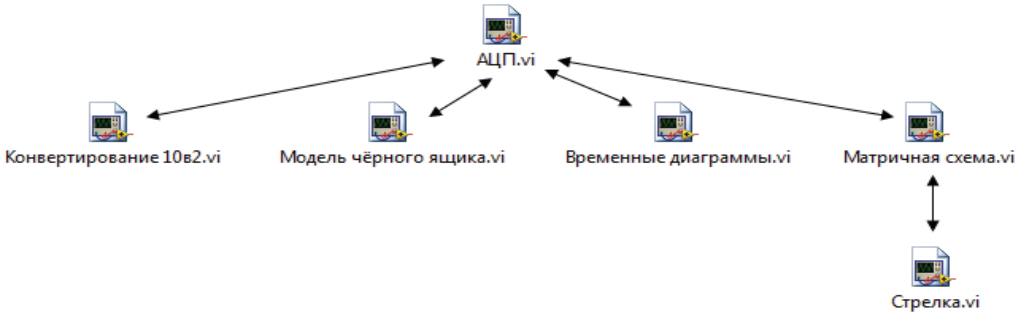


Рис. 3. Иерархия программы моделирования АЦП

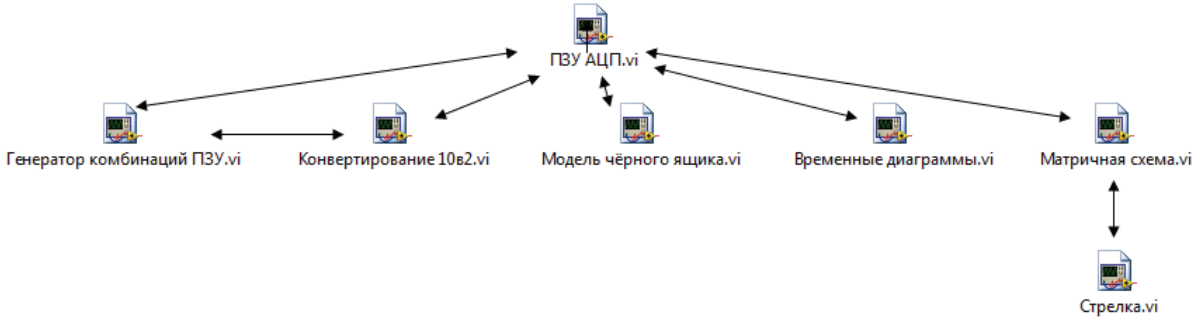


Рис. 4. Иерархия программы моделирования ПЗУ АЦП

3.2. НУЛЕВАЯ МАТРИЦА

На рис. 5 показана часть программы генерации матрицы ПЗУ размером 0. При генерировании данной матрицы известны значения, которые должны получиться во всех 5 массивах, поэтому они создаются заранее.

С регулятора Размер подается размер ПЗУ, затем с помощью оператора =0 это значение сравнивается с нулем. Если оно равно нулю, то формируется логический сигнал 1 (Правда), который подается на оператор условия Case Structure, и далее работает только ветвь Case Structure с пометкой True (рис. 4) и операторы, находящиеся на ее выходах. В ней с помощью оператора построения массива (Build Array) создается одномерный массив размерностью 4 × 1, данные из этого массива подаются на виртуальный прибор – конвертирование 10 в 2. Вирутальный прибор служит для конвертирования одномерного массива из десятичного формата чисел в двумерный массив из чисел двоичного формата. В данном случае использование этого прибора не является обязательным, т. к. ноль что в 2-й, что в 10-й системе счисления записывается одинаково. Массивы используются только для придания данным нужного формата. Далее информация поступает на Array Subset – оператор вычленения из массива подмассива. С помощью этого оператора путем указания точных адресов подмассивов (константы 1 и 0) выделяются 2 подмассива, которые потом транспонируются и становятся массивами Входы и Выходы. Затем снимаются данные в массивы Такт и Проверка. Для создания массива Заголовки, имеющего строковый формат, используется оператор Build Array, создающий массив размером 1 × 4, в котором на входы подаются строковые константы a0, a1, Q1, Q0.

Для проверки правильности работы ПЗУ после оператора Case Structure объединяются данные из массива «Заголовки» и переведенные в строковый вид данные для массивов «Такт», «Входы», «Выходы», «Проверка». Для этого данные с конвертора подаются на следующий блок, состоящий из двух For Loop (цикл с параметром) и оператора Format Into String со строковым параметром %d для обозначения формата. Затем значения с этого блока транспонируются Transpose 2D Array и передаются на оператор Build Array, производящий объединение данных, которые, в свою очередь, передаются на Таблица ПЗУ.

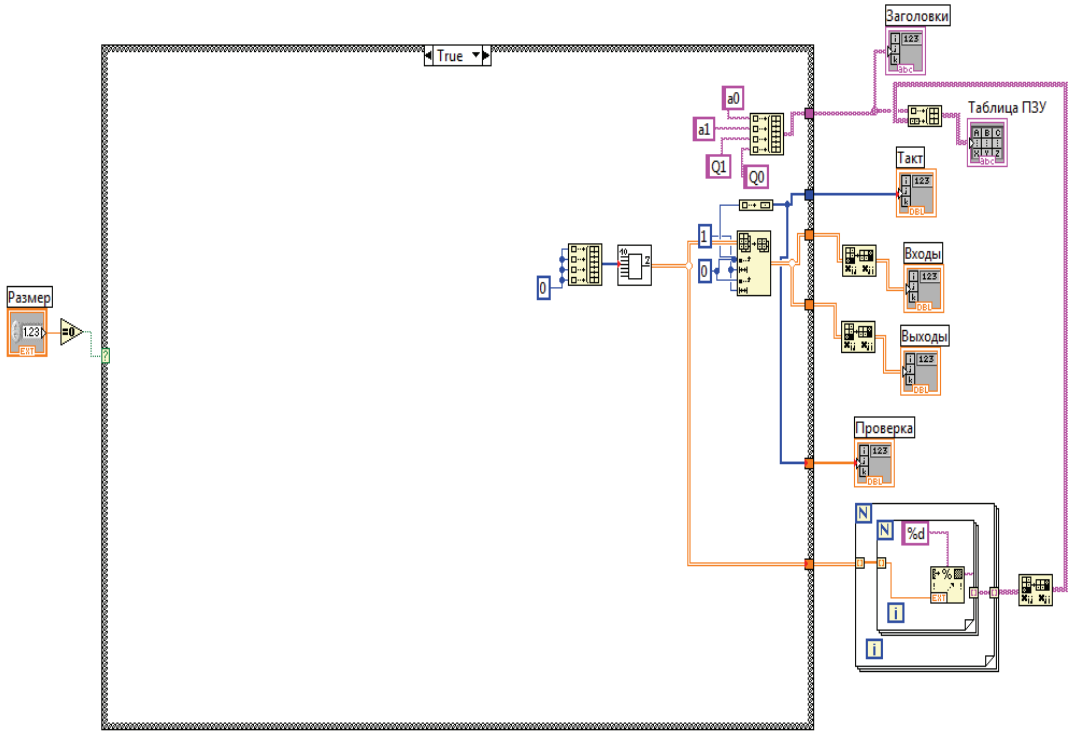
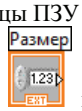


Рис. 5. Структура генерирования ПЗУ размером 0

3.3. ЕДИНИЧНАЯ МАТРИЦА

На рис. 6 представлена структура программы генерирования матрицы ПЗУ размером один.

С регулятора 

подается размер ПЗУ, затем с помощью оператора  это значение сравнивается с нулем. Если оно не равно нулю, то формируется логический сигнал 0 (Ложь), который подается на оператор условия Case Structure, и далее работает только ветвь Case Structure с пометкой False и операторы, находящиеся на ее выходах. Но в этом операторе условия есть внутренний Case Structure, на который подается логический результат сравнения размера ПЗУ с единицей (оператор «не равно» – ). Размер ПЗУ в данном случае передается в виде локальной переменной 

. Если размер ПЗУ равен 1, то работать будет часть программы, показанная на рис. 5.

Для значений, которые будут в массиве «Входы», создается пустой массив с помощью элемента уже знакомого оператора  с 4 входами и 1 выходом. На 1-й и 2-й входы подается 0. На 3-й подается размер ПЗУ (в данном случае 1) из локальной переменной 

. На 4-й подается массив чисел, снятых с цикла, ответственного за генерирование ряда чисел в порядке уменьшения с двойным повторением, размером от размера, введенного пользователем, до 1 (в данном случае данные на него не подаются).

3.3.1. МАССИВ «ТАКТ»

Рассмотрим подробно этот блок. В его основе цикл с условием. В качестве условия выбрано то, что декремент входящего значения больше 1. Операторы – Декремент  , больше  . При работе цикла для хранения промежуточных данных используются регистры сдвига (оранжевые прямоугольники со стрелкой вниз).

Для значений, которые будут в массиве «Выходы», также создается пустой массив с помощью оператора  с 4 входами и 1 выходом. На 1-й вход подается 0. На 2-й и 3-й подается размер ПЗУ (в данном случае 1) из локальной переменной 

. На 4-й подается массив чисел, снятых с цикла, ответственного за генерирование ряда чисел в порядке уменьшения с двойным повторением, размером от размера, введенного пользователем, до 1.

Затем данные из этих массивов конвертируются в 2-й код (виртуальный прибор ) и транспонируются  . Копия данных для массива «Выходы» оставляется в 10-м коде, эти данные необходимы для массива «Проверка» и для создания массива «Такт». Вначале определяется количество элементов в массиве «Проверка» (одномерный массив) с помощью оператора Array Size (Размеры массива)  . Затем это значение делится на 2, но не с помощью обычной операции деления  , а с помощью оператора Quotient & Remainder (целая часть и остаток)  . Получившееся число целой части будет размерностью для массива,

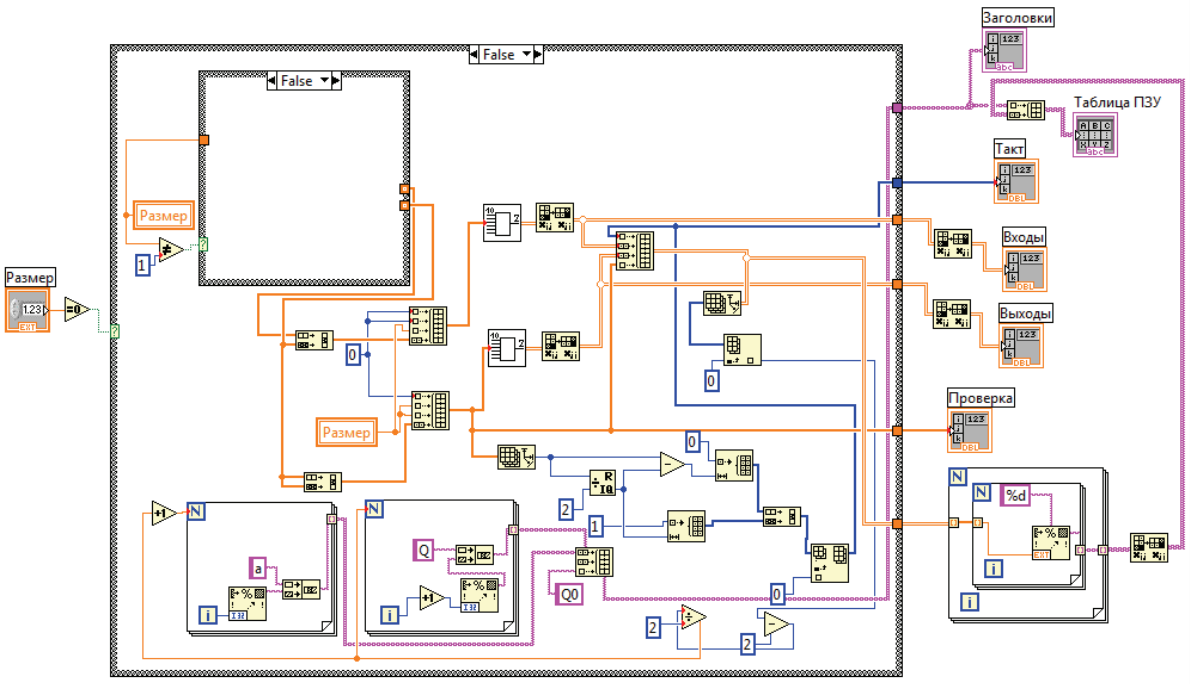


Рис. 6. Структура генерирования ПЗУ размером один

который содержит только нули, а разность между количеством элементов, определенном в , и числом целой части – размерность массива, который содержит только единицы. Эти массивы создаются с помощью оператора Initialize Array (определить массив) . На входы подаются размерность и элемент. Затем эти 2 массива объединяются с помощью оператора Interleave 1D Arrays (объединить одномерные массивы с поэлементным чередованием) , и с помощью оператора Insert Into Array (добавить в массив) добавляется ноль на первую позицию . Так получается массив «Такт».

Далее значения для массивов «Такт», «Входы», «Выходы», «Проверка» объединяются в один массив с помощью операции Build Array.

3.3.2. МАССИВ «ЗАГОЛОВКИ»

Далее создается массив «Заголовки». Это строковый массив. Определяется количество столбцов в объединенном массиве с помощью уже знакомого оператора (Array Size) и оператора Index Array (вычленение элемента из массива по индексу) . В данном случае вычленяется элемент с индексом . Получившееся число показывает количество разрядов в числах массива. На основании этого числа создаются массивы заголовков для входов и выходов. В начале из этого числа вычитается 2 (т. к. массивы «Такт» и «Проверка» имеют особые заголовки). Оператор вычитания . Потом получившееся значение делится на 2 с помощью оператора деления . Полученное значение подается на два For Loop (цикл с параметром) как параметр *N*: на 1-й через инкремент , на второй – в исходном виде. Внутри первого цикла происходит слияние (Concatenate Strings) строковой константы и переведенного в строковый формат значения со счетчика итераций. Внутри второго цикла происходит слияние (Concatenate Strings) строковой константы и переведенного в строковый формат значения со счетчика итераций, подвергшегося предварительному инкрементированию . Затем эти массивы и строковая константа , необходимая для обозначения проверки, объединяются в массив (Build Array), образуя «Заголовки».

Для проверки правильности работы ПЗУ после оператора Case Structure объединаются данные из массива «Заголовки» и переведенные в строковый вид данные для массивов «Такт», «Входы», «Выходы», «Проверка». О проверке подробно рассказано в описании части, которая генерирует матрицу ПЗУ размером 0.

3.4. ПРОГРАММИРУЕМАЯ МАТРИЦА

На рис. 7 структура программы генерирования программируемой матрицы ПЗУ размером более 1. В этой части программы используются все те же компоненты, что и при генерировании матрицы ПЗУ размером 1, только в исходных массивах при создании массивов «Входы» и «Выходы» последний вход задействован. О данных, подающихся на этот вход, рассказано при описании части, которая генерирует матрицу ПЗУ размером один.

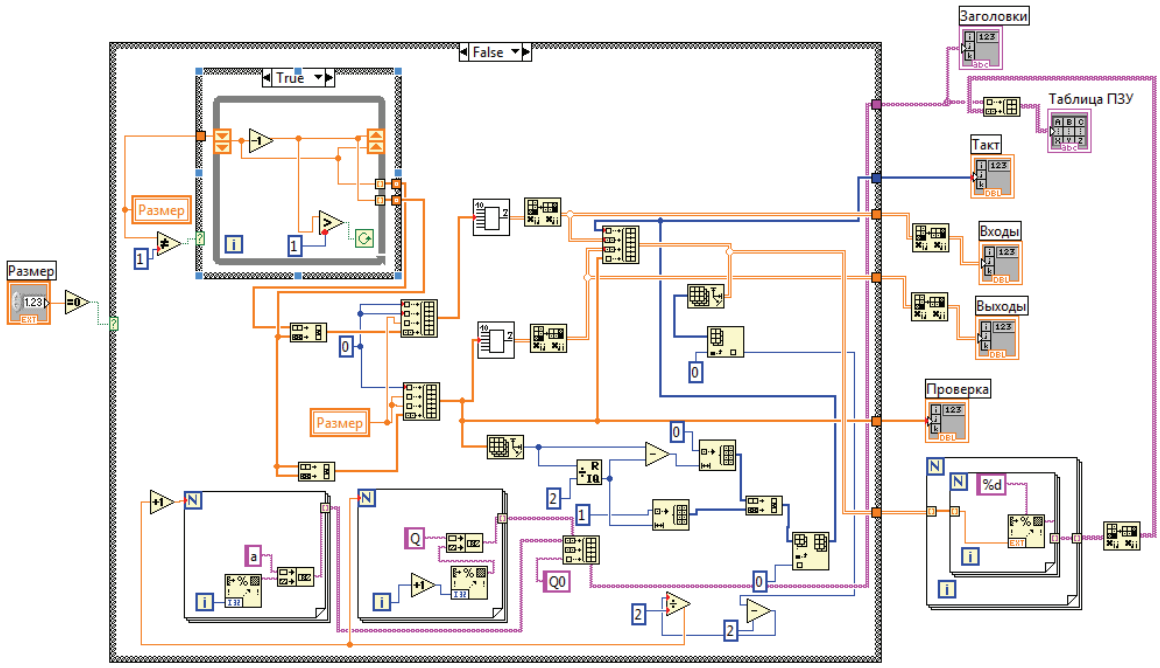


Рис. 7. Структура генерирования любого ПЗУ размером более одного

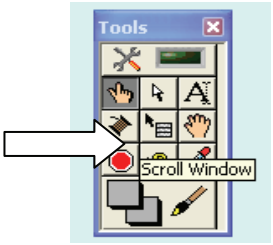


Рис. 8. Инструмент для переходов

Синтезирована мнемосхема таблицы состояний ПЗУ до 128 по методу аналогии на основании математического обеспечения для проектирования схемотехнических и метрологических средств АЦП ПЗУ.

Синтезированы таблицы состояний АЦП и ПЗУ АЦП по методу аналогии из математического обеспечения, архитектуры мнемотехники информационных технологий АЦП.

Спроектировано АЦП поразрядного уравновешивания в программной среде LabView на основных иерархических уровнях: структурном, функциональном и принципиальном.

4. ПРИМЕРЫ

Приведем примеры моделирования АЦП для пояснения на пользовательском уровне. При запуске программы на лицевой панели появляется группа вкладок, первая из которых называется «О проекте». Содержимое этой вкладки не меняется при работе программы.

Чтобы перейти на следующую вкладку, необходимо либо выбрать на панели Tools элемент, показанный на рис. 8, либо сразу включить программу в циклическом режиме (рис. 9).

Следующая вкладка называется «Теория», там содержится сжатый теоретический материал. Содержимое этой вкладки также не меняется при работе программы.

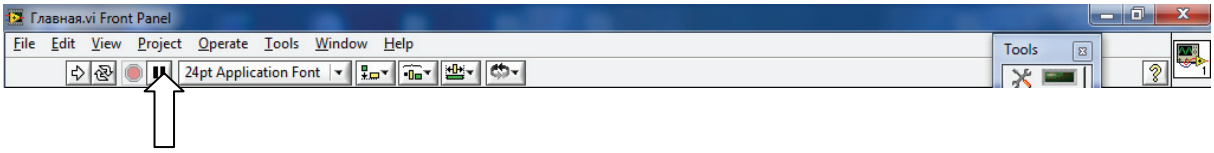


Рис. 9. Циклический запуск программы

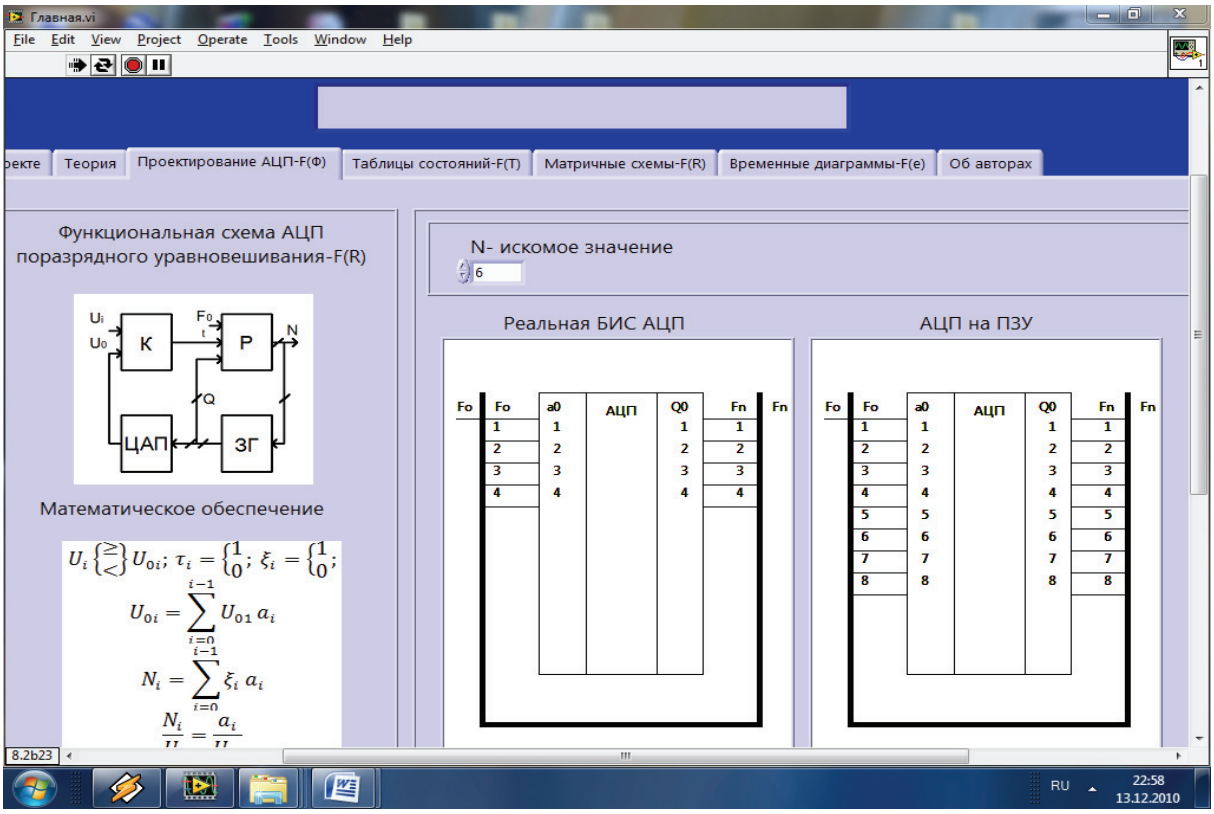


Рис. 10. Вкладка «Проектирование АЦП-F(Ф)»

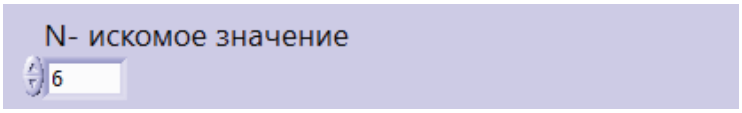


Рис. 11. Управляющий параметр

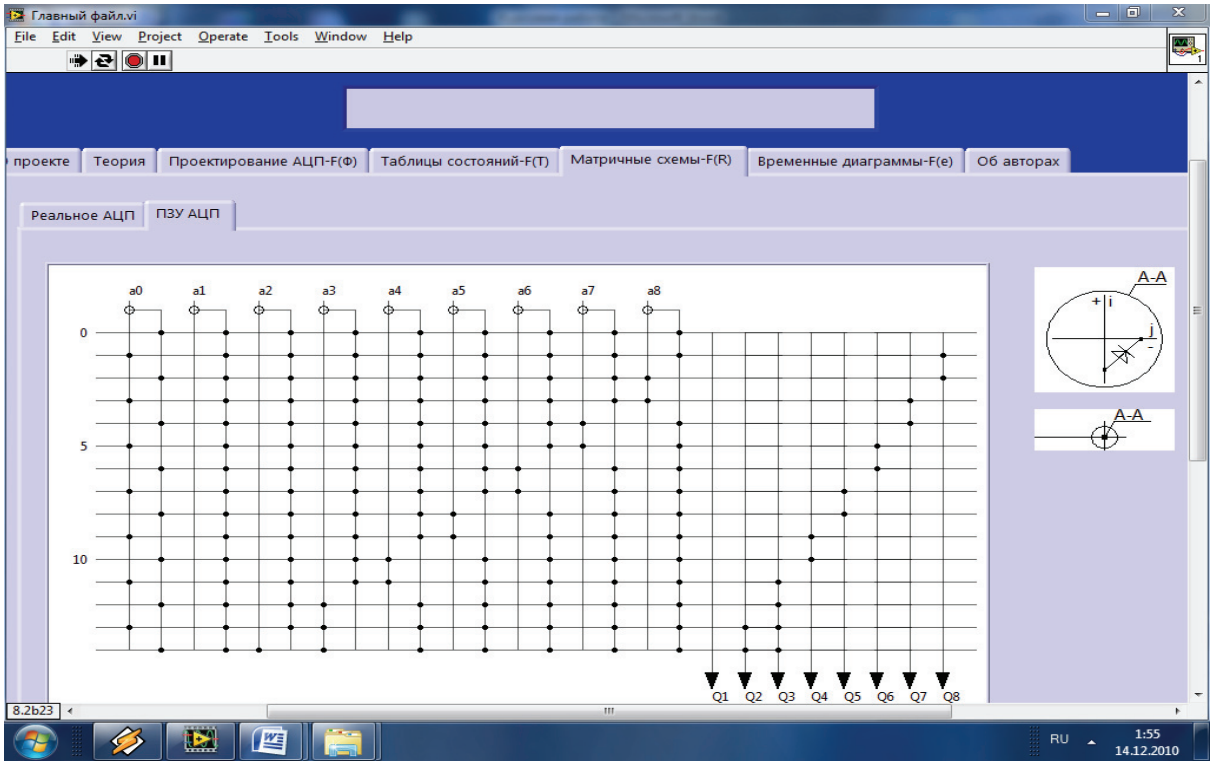


Рис. 12. Матричная схема «ПЗУ АЦП» при $N = 6$

Следующая вкладка называется «Проектирование АЦП-F(Ф)». Это вкладка является основной для работы с программой. В ней располагаются: элементы управления, функциональная схема, сведения о математическом обеспечении и схемы реальной БИС АЦП и АЦП на ПЗУ (которые изменяются в зависимости от управляющих данных). Содержимое этой вкладки при $N = 6$ показано на рис. 10. Эта вкладка служит для управления программой. В качестве управляющего параметра используется искомое значение – N, которое можно либо водить с клавиатуры, либо с помощью мышки. По умолчанию данному параметру присвоено значение 128 (максимальное для ПЗУ АЦП). На рис. 11 показана часть вкладки, ответственная за изменение управляющего параметра.

В случае если пользователь введет значение, превышающее размер ПЗУ, в самом верху программы высветится сообщение об ошибке.

Схемы реальной БИС АЦП и АЦП на ПЗУ служат для демонстрации того, что при работе АЦП обычно задействованы не все входы. БИС АЦП показывает АЦП таким образом, что все входы и выходы задействованы за счет изменения количества входов и выходов. А АЦП на ПЗУ показывает АЦП с неизменных количеством входов и выходов вне зависимости от параметра N.

Следующая вкладка называется «Таблицы состояний – F(T)». Она содержит 3 вкладки более низкого уровня: «Матрица АЦП», «Матрица ПЗУ АЦП», «Матрица ПЗУ». Во вкладке «Матрица АЦП» показан эталонный вид таблицы состояний для значения N. Во вкладке «Матрица ПЗУ АЦП» показан вид таблицы состояний для значения N, найденного АЦП по матрице ПЗУ. А во вкладке «Матрица ПЗУ» показан вид этой матрицы. Она генерируется автоматически еще до введения N, т. е. при запуске программы.

Следующая вкладка называется «Матричные схемы – F(R)». Она содержит две вкладки более низкого уровня: «Реальное АЦП», «ПЗУ АЦП». Во вкладке «Реальное АЦП» показана матричная схема реального АЦП, а во вкладке «ПЗУ АЦП» показана матричная схема для ПЗУ АЦП, т. е. для АЦП, построенного на матрице ПЗУ (рис. 12).

Далее следует вкладка «Временные диаграммы – F(e)». Она содержит две вкладки более низкого уровня: «Реальное АЦП», «ПЗУ АЦП». Во вкладке «Реальное АЦП» показаны две временные диаграммы. На первой из них первые 8 разрядов временной диаграммы входящих сигналов, на второй – первые 8 разрядов временной диаграммы выходящих сигналов.

Во вкладке «ПЗУ АЦП» тоже показаны две временные диаграммы. На первой из них – временная диа-

грамма входящих сигналов. На второй – временная диаграмма выходящих сигналов.

Последняя вкладка программы посвящена авторам и их вкладу в ее разработку и написание.

ВЫВОДЫ

Проанализирован АЦП поразрядного уравновешивания на уровне функциональной схемы, поясняющей физику работы для моделирования в среде LabVIEW его компонент схемо- и мнемотехники, математического обеспечения и метрологических средств.

Синтезирована мнемосхема таблицы состояний ПЗУ до 128 по методу аналогии на основании математического обеспечения для проектирования схемотехнических и метрологических средств АЦП ПЗУ.

Синтезированы таблицы состояний АЦП и ПЗУ АЦП по методу аналогии из математического обеспечения, архитектуры мнемотехники информационных технологий АЦП.

Синтезированы матричные схемы АЦП и ПЗУ АЦП по методу аналогии из таблицы состояний для изучения информационных технологий АЦП.

Приведено семейство временных диаграмм для АЦП и ПЗУ АЦП по методу аналогии из таблицы состояний для изучения метрологических средств информационных технологий АЦП.

Спроектировано АЦП поразрядного уравновешивания в программной среде LabView на основных иерархических уровнях: структурном, функциональном и принципиальном.

ЛИТЕРАТУРА

1. *Глинкин Е.И.* Схемотехника аналого-цифровых преобразователей. Тамбов: Изд-во ТГТУ, 2001. 160 с.
2. *Глинкин Е.И., Глинкин М.Е.* Схемотехника микропроцессорных средств. Тамбов: Изд-во ТГТУ, 2005. 148 с.
3. *Тревис Д.* LabVIEW для всех. Прибор комплекс, 2005. 544 с.

Поступила в редакцию 5 апреля 2012 г.

Potlov A.Yu., Semenov D.A., Ostapenko O.A., Golubyatnikov O.O., Vlasova E.V., Glinkin E.I. MODELING ADC OF BIT BALANCING

Methods of the synthesis and the analysis BIS in technology engineering of ADC of bit balancing by LabView are systemized.

Key words: analog-digital converters; LabView program environment; states table; matrix schemes; temporary diagrams; analogy method.