Учреждение образование

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет издательского дела и полиграфии

Кафедра информационных систем и технологий

Специальность 1-40 01 02-03

Специализация «Издательско-полиграфический комплекс»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**К КУРСОВОЙРАБОТЕ**

по дисциплине «Специализированные информационные системы»

Тема: Преобразование логических формул на язык LD

Исполнитель

Студент 5 курса группы 10 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Романенко И. В.

(подпись, дата)

Руководитель

доцент, канд. техн. наук \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Акунович С. И.

(подпись, дата)

Курсовой проект защищен с оценкой\_\_\_\_\_\_\_\_\_\_\_

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Акунович С. И.

Минск 2013

# РЕФЕРАТ

Курсовая работа: 13 страниц, 7 рисунков, 3 источника

ЯЗЫК ПРОГРАММИРОВАНИЯ DELPHI, ЯЗЫК LD, ЛОГИЧЕСКИЕ ФОРМУЛЫ

Объектом исследования является преобразование логических формул на язык LD.

Цель работы – доработать программу Функции\_LD и выполнить преобразование логических формул с вложенными скобками и без вложенных скобок.

При выполнении работы использовался язык программирования Delphi и его среда разработки Delphi XE 2, а так же IsaGRAF версии 6.2.

В результате работы была доработана программа Функции\_LD и выполнено преобразование логических формул.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись студента)

# СОДЕРЖАНИЕ

[РЕФЕРАТ 3](#_Toc375777373)

[СОДЕРЖАНИЕ 4](#_Toc375777374)

[ВВЕДЕНИЕ 5](#_Toc375777375)

[1. ЯЗЫК LD 6](#_Toc375777376)

[2. ПРОГРАММА ФУНКЦИИ\_LD 8](#_Toc375777377)

[3. ДОРАБОТКА АЛГОРИТМА И ПРОГРАММЫ 9](#_Toc375777378)

[4. ПРЕДСТАВЛЕНИЕ ФУНКЦИЙ НА LD 10](#_Toc375777379)

[ЗАКЛЮЧЕНИЕ 14](#_Toc375777380)

[СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 15](#_Toc375777381)

# ВВЕДЕНИЕ

Современные технологические машины и процессы содержат многочисленные датчики, сигнализирующие об их состояниях. Сигналы с датчиков поступают на вход системы управления, которая должна анализировать текущую ситуацию и вырабатывать адекватные воздействия на исполнительные компоненты оборудования, непосредственно управляющие исполнительными механизмами: переключателями, клапанами, двигателями, насосами, электромоторами. Как сигналы с датчиков, так и управляющие воздействия дискретны, а чаще всего двоичные, т. е. управление становится логическим, и в этом случае системы управления принято называть системами логического управления.

Системы логического управления находят широкое применение не только на производстве, но и на транспорте, в атомной энергетике, сетях связи, информационно-справочных системах, ракетно-космической отрасли, компьютерах и бытовой аппаратуре и т. д.

Разработка любой системы управления всегда начинается с алгоритмического описания процесса управления. Лишь после четкого описания алгоритма, его проверки и отладки средствами моделирования можно перейти к конструированию технической системы, реализующей этот алгоритм. Если в описании алгоритма допущены ошибки, дальнейшая его реализация может оказаться бессмысленной.

В случае обнаружения ошибок во время выполнения системы среда разработки отображает их в своем окне вывода, но чтобы этот процесс стал более удобен, можно выводить их в отдельный текстовый файл, даже не запуская саму среду разработки. Это дает возможность легко скопировать название ошибки и осуществить поиск в Интернете, либо в электронных книгах.

# ЯЗЫК LD

Язык LD – это графический язык - применяется для описания логических выражений различного уровня сложности, графического представления булевых уравнений. Он содержит контакты (входные аргументы) и катушки (выходные переменные). Элементы организуются в сеть релейно-контактных схем. При необходимости можно реализовывать более сложную логику, используя, например элементы языка FBD.

Каждому контакту ставится в соответствие логическая переменная, определяющая его состояние. Ее имя ставится над контактом и служит его названием. Если контакт замкнут, то переменная имеет значение true, если разомкнут – false. Последовательное соединение контактов или цепей соответствует логической операции И/AND, параллельное – ИЛИ/OR. Нормально замкнутый (инверсный) контакт равнозначен логической операции НЕ.

Релейная схема представляет собой 2 вертикальные шины питания, между которыми расположены горизонтальные цепи из контактов и катушек реле. Графические символы языка LD соответствуют элементам электрических цепей и имеют те же названия и обозначения.

LD-программа выполняется последовательно слева направо и сверху вниз. В каждом рабочем цикле однократно выполняются все цепи, входящие в сеть. Любая переменная в рамках одной цепи всегда имеет одно и то же значение. Если даже реле в цепи изменит переменную, то новое значение поступит на контакты только в следующем цикле. Цепи, расположенные ниже, получат новое значение переменной сразу, а расположенные выше - только в следующем цикле.

Строгий порядок выполнения цепей очень важен. Благодаря жесткому порядку выполнения LD-программы сохраняют устойчивость при наличии обратных связей.

Словарь ISaGRAF – это средство для редактирования внутренних переменных, переменных ввода/вывода, функциональных блоков и макросов приложения.

Переменные, функциональные блоки и макросы должны быть объявлены в словаре перед выполнением LD-программы. Переменные и макросы могут использоваться в любом языке: SFC, FBD, LD, ST, и IL.

Переменные сортируются в соответствии со своей областью действия и типом.

Основные типы области видимости переменных:

* Глобальная – может быть использована любой программой текущего проекта;
* Локальная – может быть использована только одной программой.
* Основные типы переменных:
* Булевская – true/false двоичные величины;
* Аналог – действительные или целые величины;
* Таймер – временные величины;
* Сообщение – символьные строки.
* Переменная идентифицируется именем, комментарием, атрибутами, сетевым адресов и другими специальными полями. Ниже приведены основные атрибуты переменных:
* Внутренняя – переменная в памяти;
* Вход – переменная, связанная с устройством ввода;
* Выход – переменная, связанная с устройством вывода;
* Константа – внутренняя переменная только для чтения (с начальным значением).

Каждая переменная имеет имя, заданное по принципу транслитерации переменных из ТЗ. Также каждая переменная имеет алиас, который представляет собой имя этой переменной в ТЗ.

Функции памяти (триггеры) представляют собой переменные, хранящиеся в памяти, и их следующее значение зависит от предыдущего значения. Поэтому их значение определяется с помощью двух блоков: блока SET (включить) и блока RESET (выключить). Если сигнал доходит до блока SET, то переменная принимает значение TRUE.Если сигнал доходит до блока RESET, то переменная принимает значение FALSE.А если же сигнал доходит и до SET и до RESET, то переменная принимает значение FALSE.Это происходит потому, что блок RESET логически находится после блока SET, соответственно он выполняется последним (преобладание RESET, RS-триггер).

# ПРОГРАММА ФУНКЦИИ\_LD

Принцип работы программы *Функции\_LD* заключается в преобразовании логических формул, записанных в текстовом виде, в файл .isaxml, который среда ISaGRAF отображает в графической форме.

Язык логических диаграмм LD позволяет представить логические формулы в наглядной графической форме и исполняемом виде.

В среде разработки ISaGRAF диаграмма LD сохраняется в файле с расширением .isaxml.

При запуске ISaGRAF создается шаблон файла Prog1.isaxml.

В среде разработки ISaGRAF шаблону соответствует пустая диаграмма LD.

Программа *Функции\_LD* заполняет этот шаблон в соответствии с заданной системой логических формул.

Файл Prog1.isaxml имеет структуру XML документа и начинается тегом объявления версии языка, на которой написан документ, и кодировки документа:

<?xmlversion="1.0" encoding="utf-8"?>

В следующем тегеPou определены атрибуты версии файла (FileVersion), имени проекта (Name), комментария (Comment), языка программы проекта (Language), генерации отладочной информации (GenDebugInfo) и различные другие атрибуты.

Следующий Тег <LocalVars> представляет собой словарь переменных проекта.

Каждая переменная описывается рядом параметров через атрибуты в теге <Variable /> имя переменной (Name), тип данных (DataType), начальное значение (InitialValue), комментарий (Comment), адрес (Address), направление (Kind), алиас (Alias), права доступа (AccessRights), размер строки (StringSize) и другие атрибуты, которые можно задать в редакторе переменных среды ISaGRAF.

Далее идет тег <PouBody> и в нем секция CDATA:

<PouBody><![CDATA[

В этой секции и размещается структура логических формул на языке LD, которая отображается при открытии проекта в ISaGRAF.

# ДОРАБОТКА АЛГОРИТМА И ПРОГРАММЫ

Предлагаемая программа была не совместима с IsaGRAF версии 6.2. После преобразования формулы проект становился не работоспособным.

Ошибка заключалась в том, что временный файл программы не удалялся. Для исправления ошибки в алгоритм функции ЗаписьФайлов() был добавлен код удаления временного файла. Так же была добавлена возможность выбора временного файла программы (рисунок 3.1).

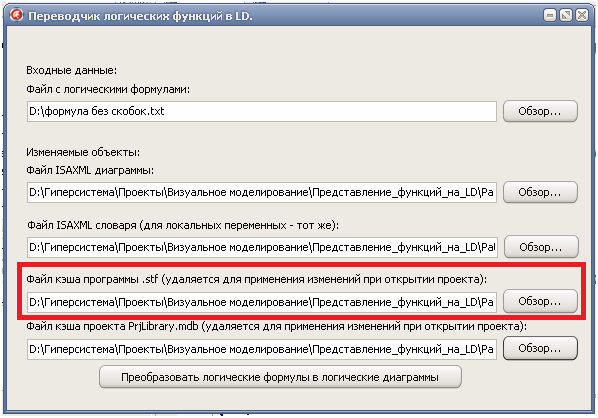


Рисунок 3.1. Главное окно программы

# ПРЕДСТАВЛЕНИЕ ФУНКЦИЙ НА LD

Представим формулу Q1 = ( Q2 + Q3 ) \* ( Q4 + Q5 ) + ^ Q6 с помощью программы Функции\_LD на языке LD.

Сначала выбираем файл диаграмм и файл словаря переменных (рисунок 4.1). Затем выберем временный файл программы (рисунок 4.2) и временный файл проекта (рисунок 4.3).

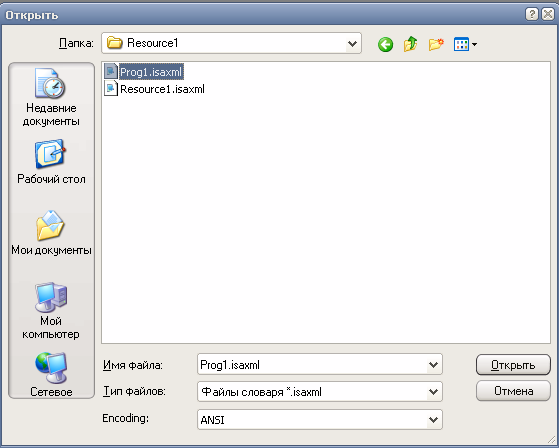


Рисунок 4.1. Выбор файл диаграмм и файл словаря переменных

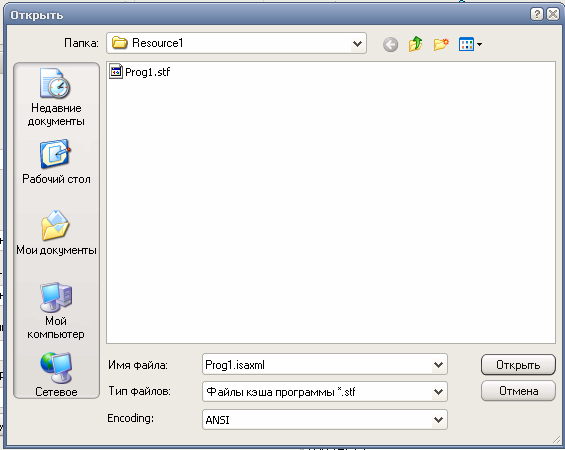


Рисунок 4.2. Выбор временного файла программы

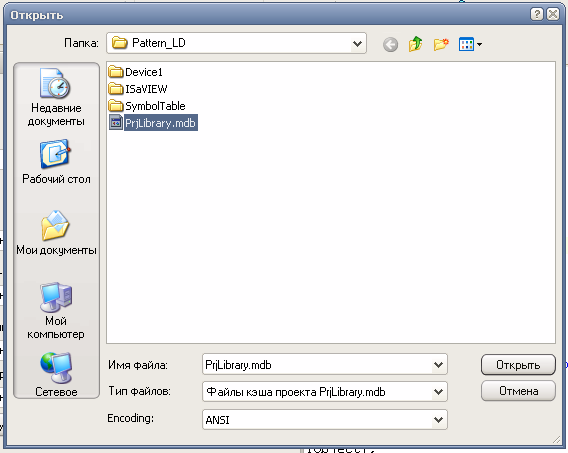


Рисунок 4.3. Выбор временного файла проекта

В главном окне программы нажмем кнопку «Преобразовать логические формулы в логические диаграммы» (рисунок 4.4)

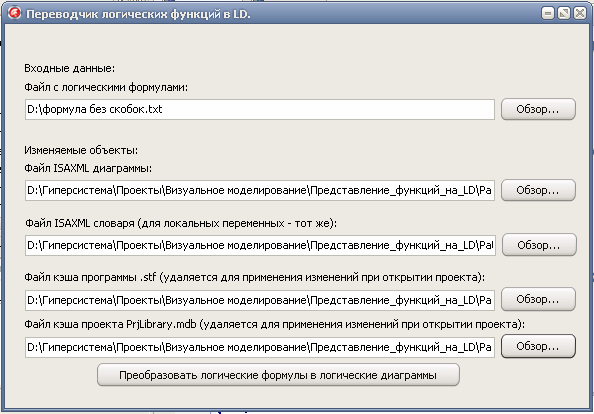


Рисунок 4.4. Главное окно программы

После открытия проекта в IsaGRAF видим результат преобразования (рисунок 3.5).

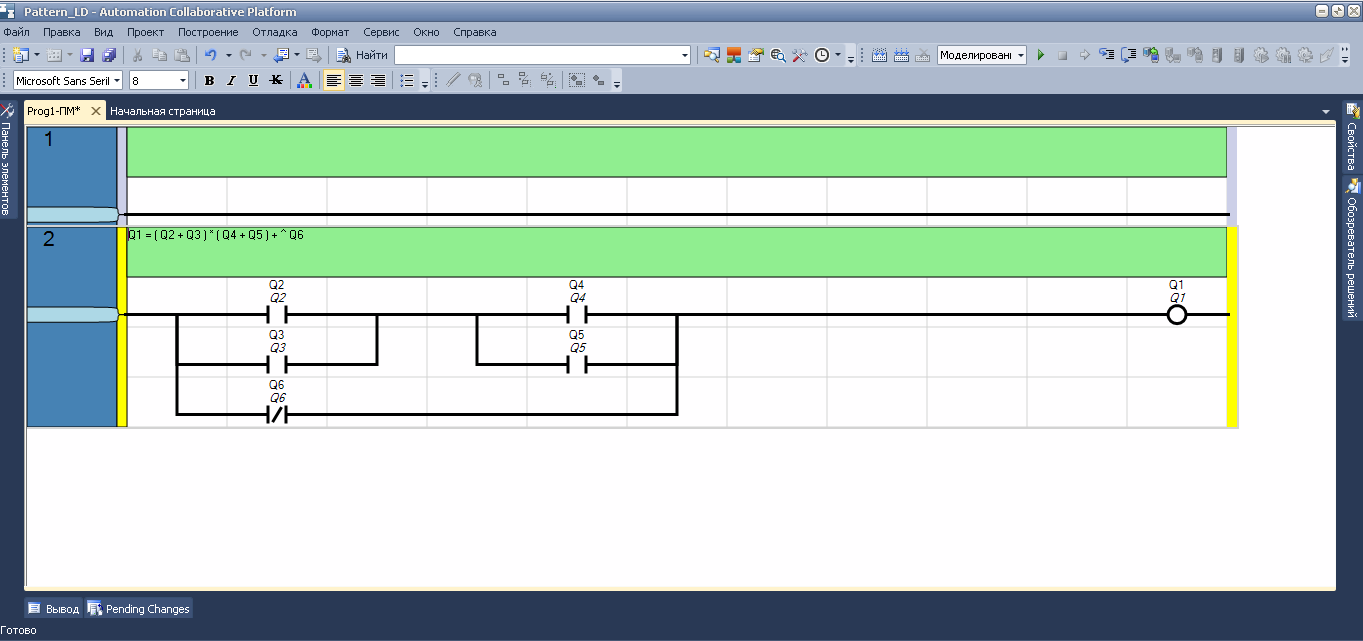


Рисунок 4.5. Результат работы программы

Для формулы с вложенными скобками Q1 = ( Q2 + Q3 ) \* ( ( Q7 + ^ ( Q7 + ^ Q8 ) \* ( Q9 + Q10 ) ) \* ( Q9 + Q10 ) + Q5 ) + ^ Q6 проделаем аналогичные действия и получим результат (рисунок 4.6).

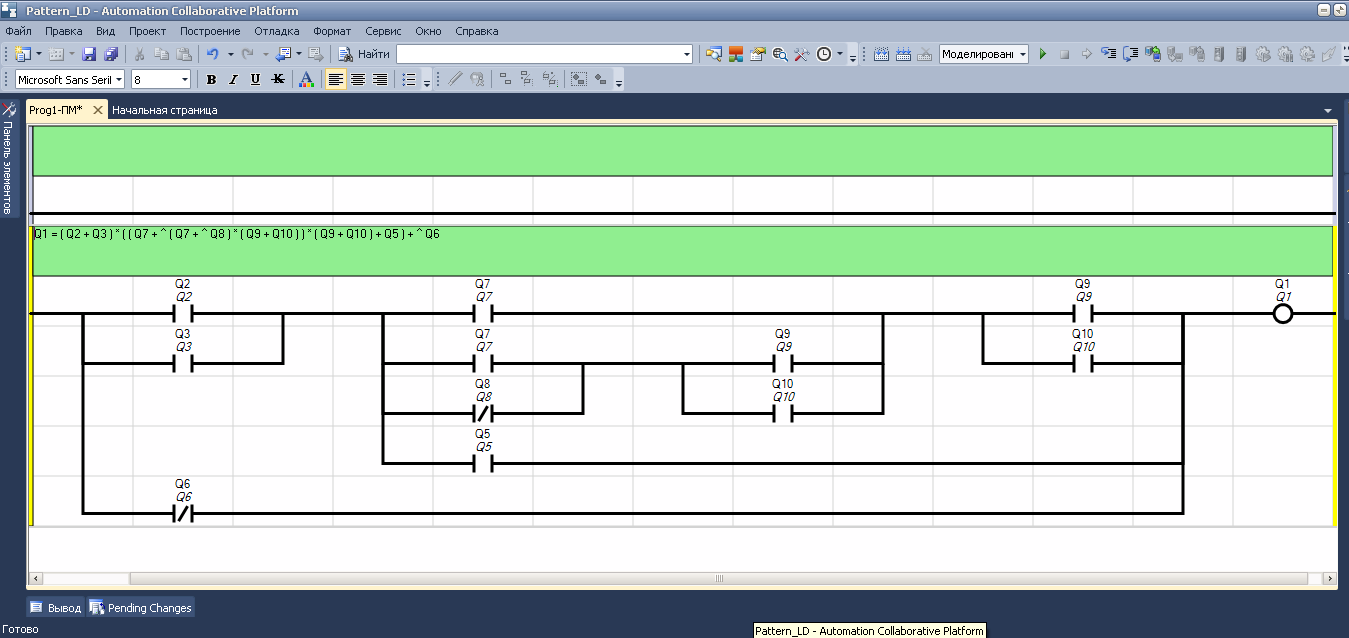


Рисунок 4.6. Результат преобразования

# ЗАКЛЮЧЕНИЕ

Язык LD позволяет удобно отображать логические формулы. В результате выполнения данной курсовой работы была найдена и исправлена ошибка в программе Функции\_LD и выполнено преобразование логических формул на язык LD.

# СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Д. Осипов. Delphi XE 2. – СПб.: БХВ-Петербург, 2012.
2. Е. Марков. Delphi 2005 для Win32. – СПб.: БХВ-Петербург, 2005.
3. http://www.delphibasics.ru.