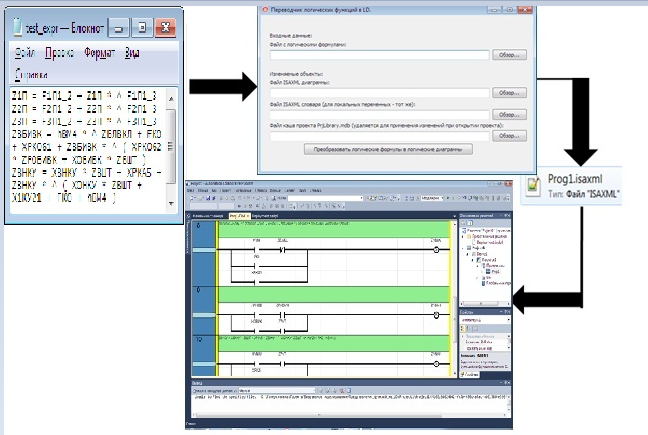
##### Программа Функции\_LD

###### Принцип работы программы

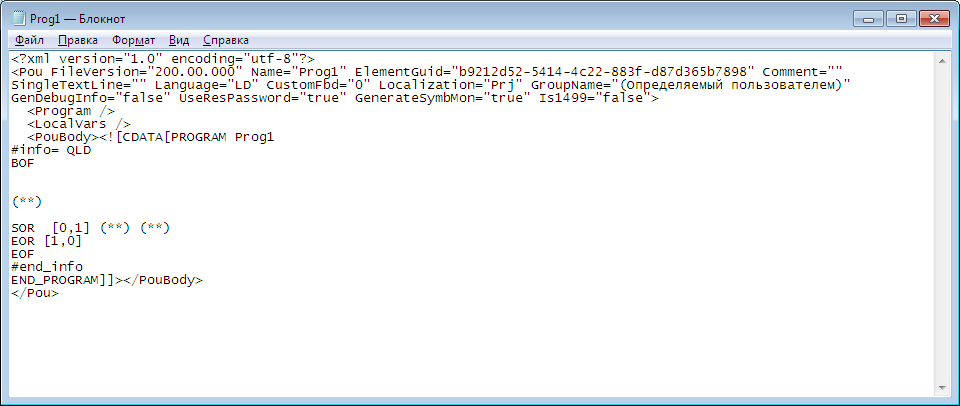
Принцип работы программы *Функции\_LD* заключается в преобразовании логических формул, записанных в текстовом виде, в файл .isaxml, который среда ISaGRAF отображает в графической форме.



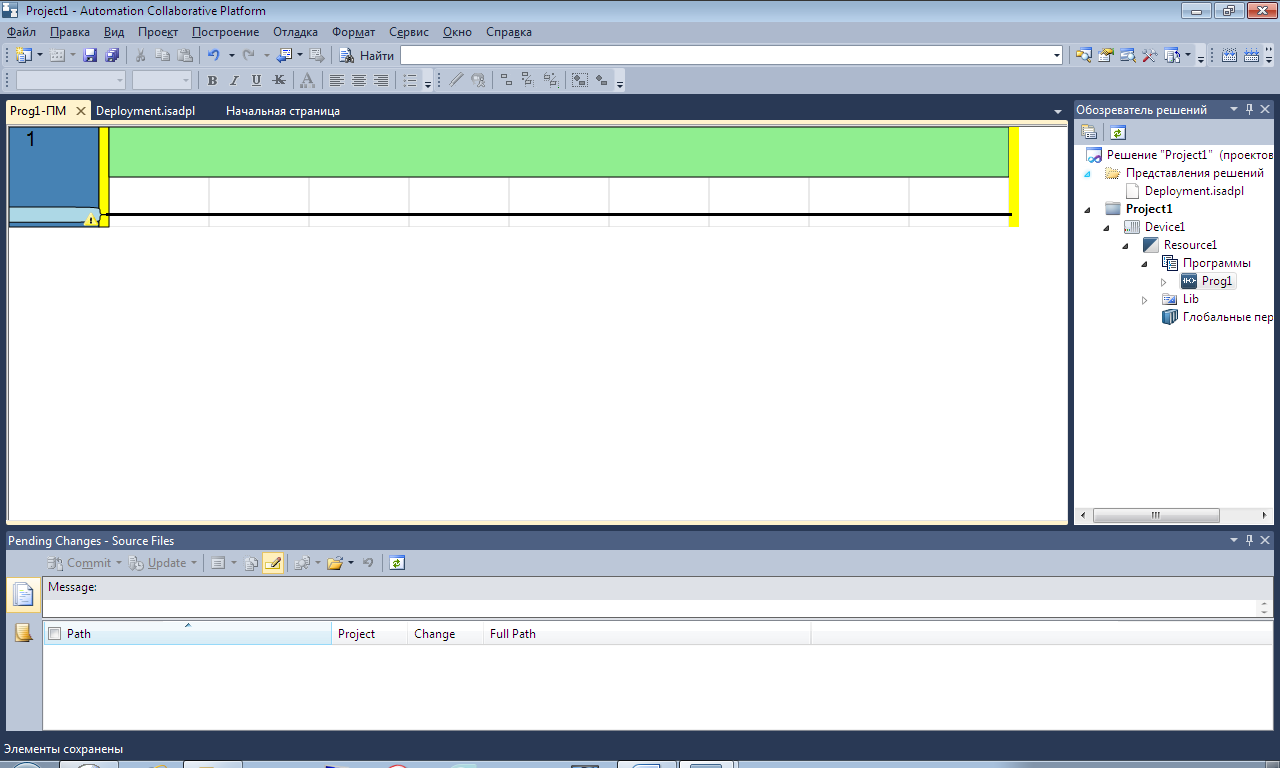
Язык логических диаграмм LD позволяет представить логические формулы в наглядной графической форме и исполняемом виде.

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

При запуске ISaGRAF создается шаблон файла Prog1.isaxml:

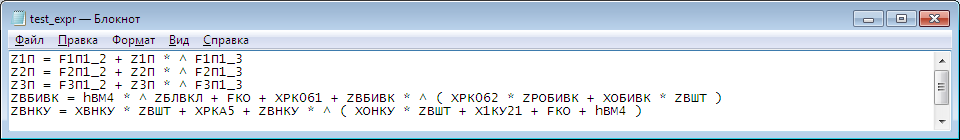


В среде разработки ISaGRAF шаблону соответствует пустая диаграмма LD:



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

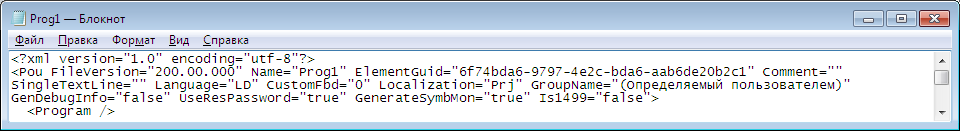
Рассмотрим подробнее структуру файла Prog1.isaxml после заполнения тестовой системой логических формул:



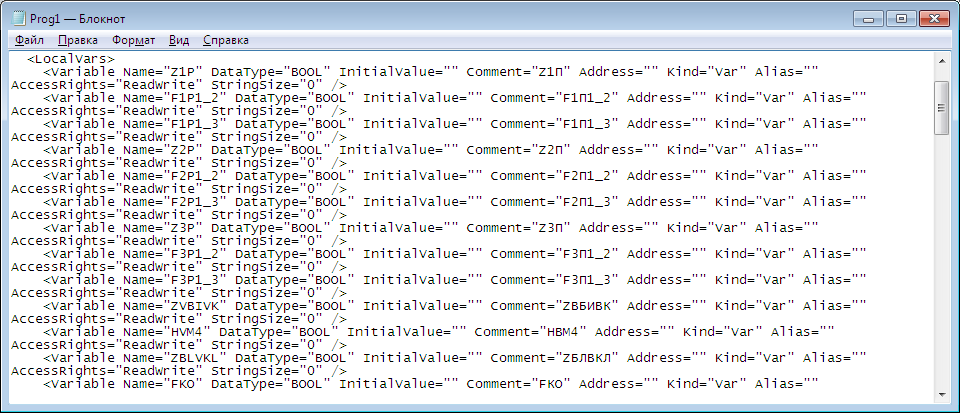
Файл Prog1.isaxml имеет структуру XML документа и начинается тегом объявления версии языка, на которой написан документ, и кодировки документа:

<?xml version="1.0" encoding="utf-8"?>

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



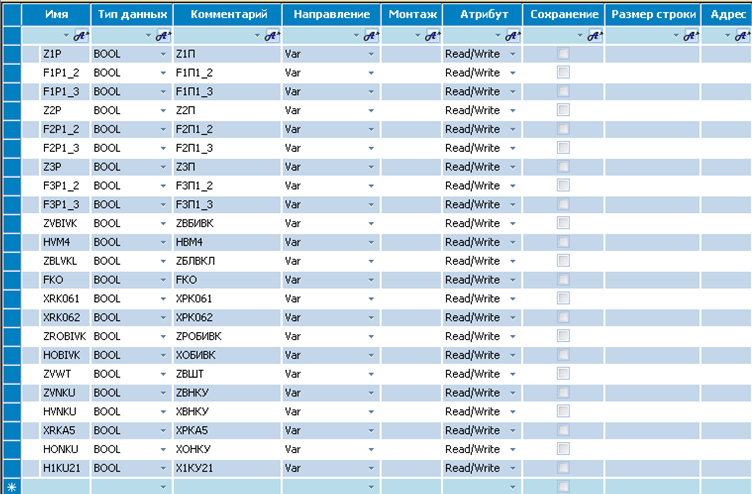
Следующий Тег <LocalVars> представляет собой словарь переменных проекта:



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)
* другие атрибуты, которые можно задать в редакторе переменных среды ISaGRAF.

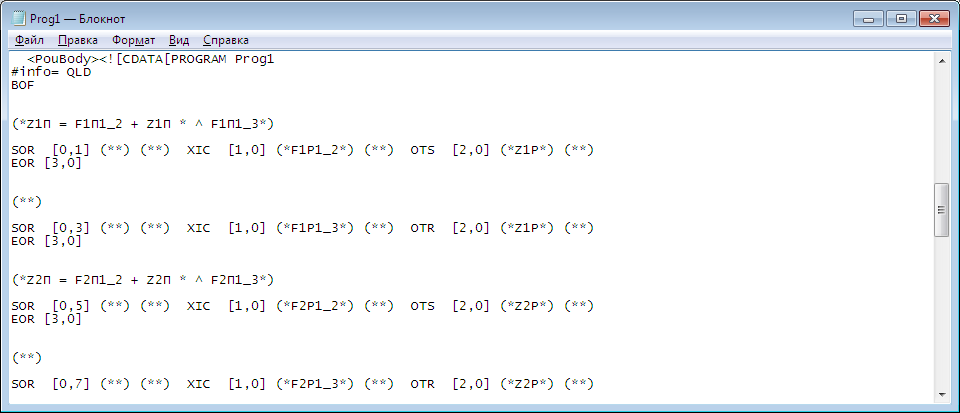
В среде разработки ISaGRAF словарь проекта имеет вид:



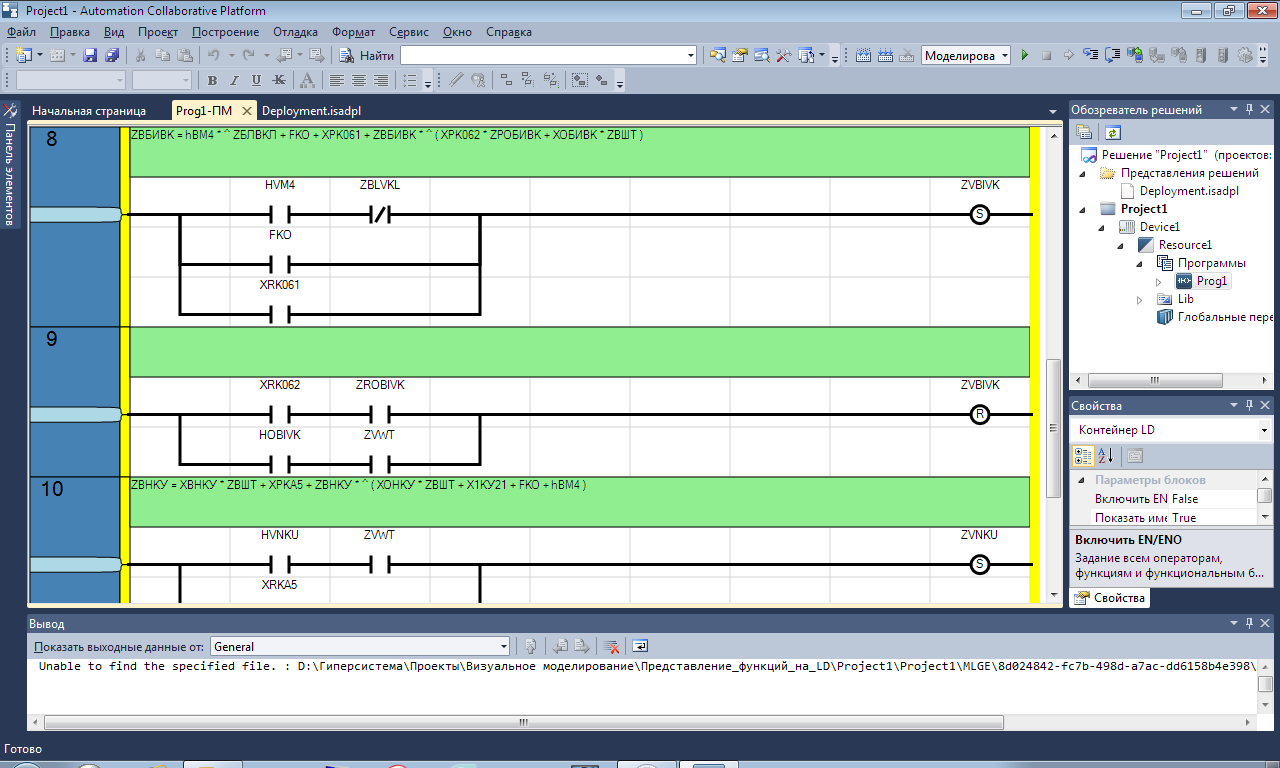
Далее идет тег <PouBody> и в нем секция CDATA:

<PouBody><![CDATA[

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



В среде разработки ISaGRAF система формул принимает вид:

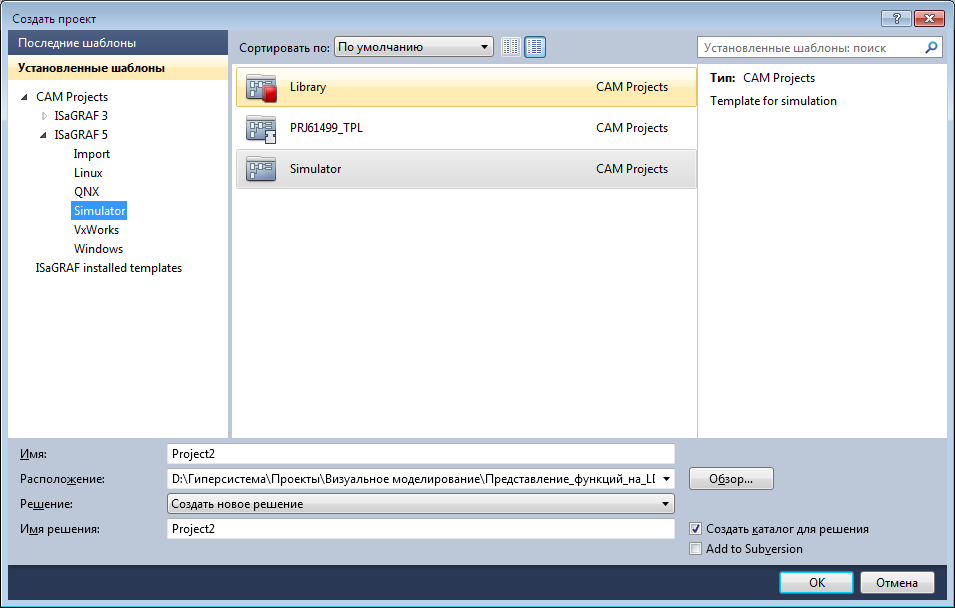


###### Подготовка исходных данных

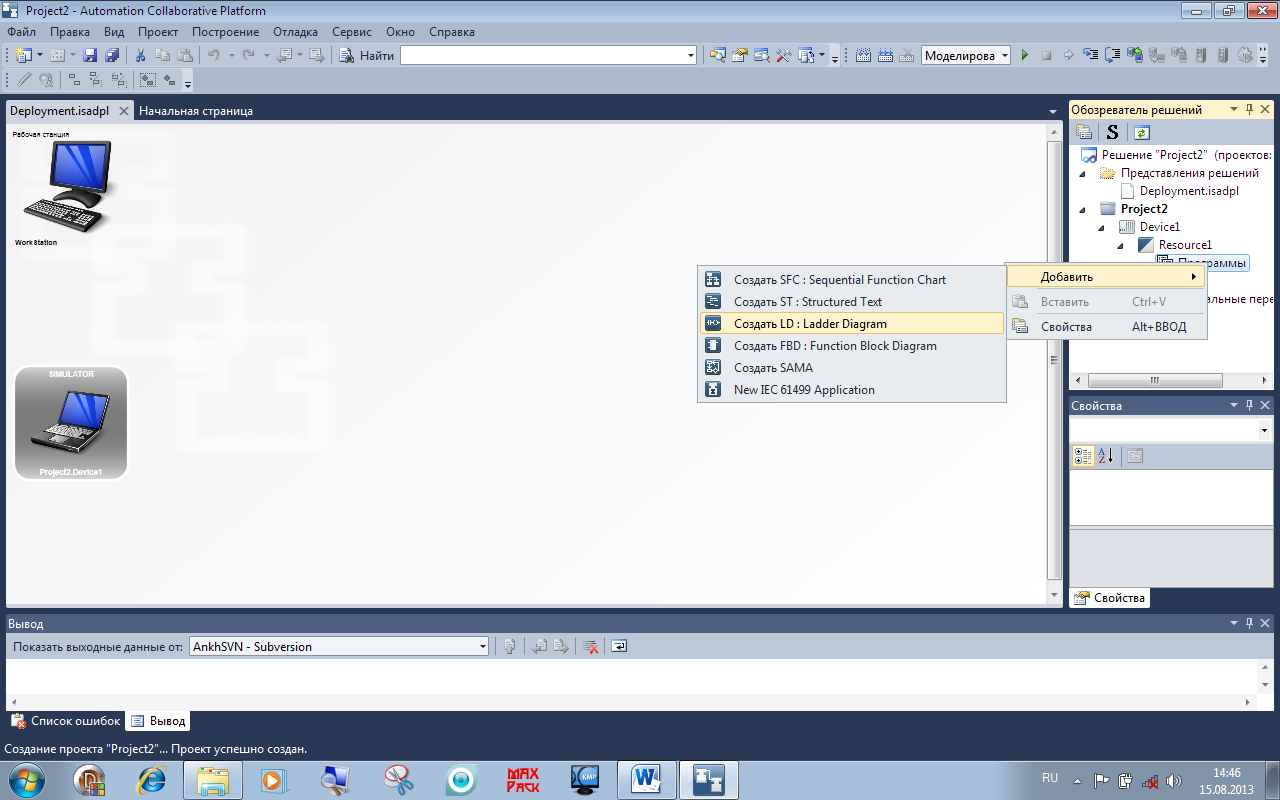
Прежде, чем начать работу с программой *Функции\_LD*, следует подготовить исходные файлы.

Исходный файл с описанием логических формул создается программой *Ввод данных ТЗ* на этапе *Ввод данных модели* как обычный текстовый файл с Функции.txt в папке **Предметные конструкции модели**.

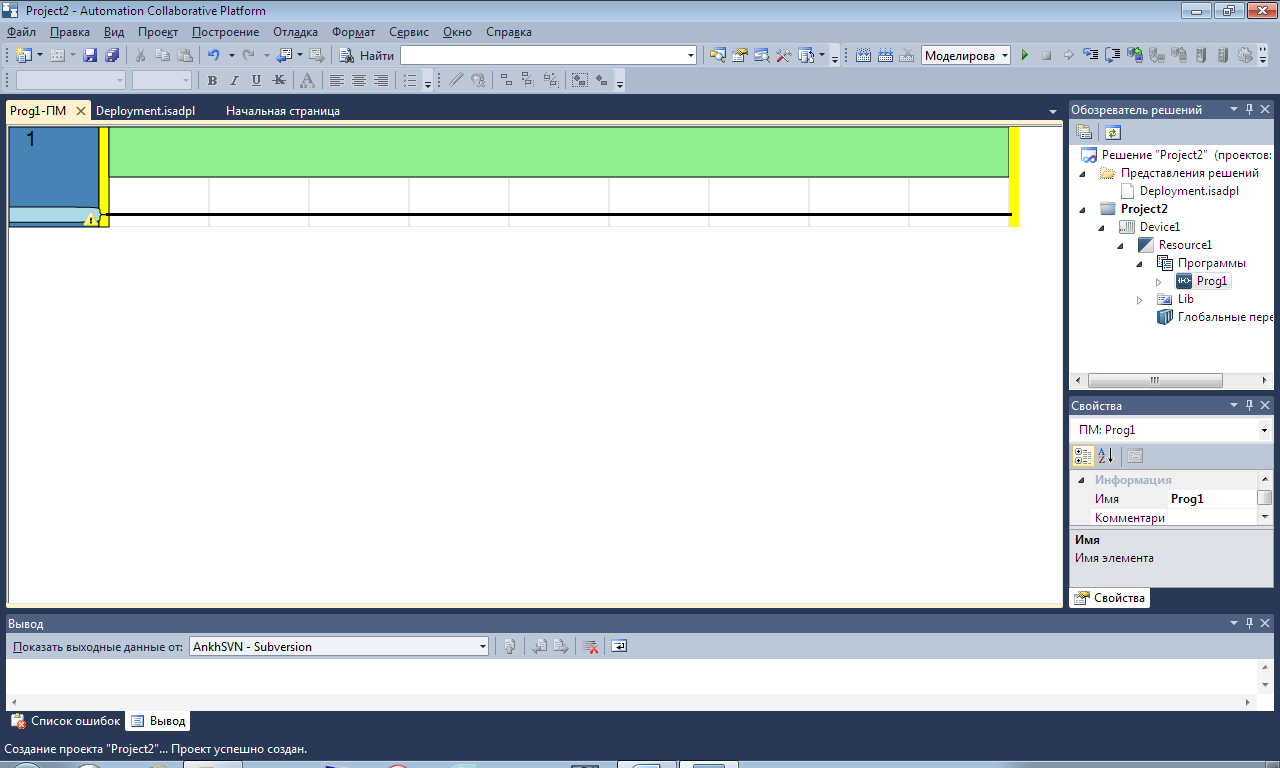
Для получения второго файла Prog1.isaxml, следует создать в среде разработки ISaGRAF новый проект.



В этом проекте нужно создать новую программу Prog1 на языке LD.



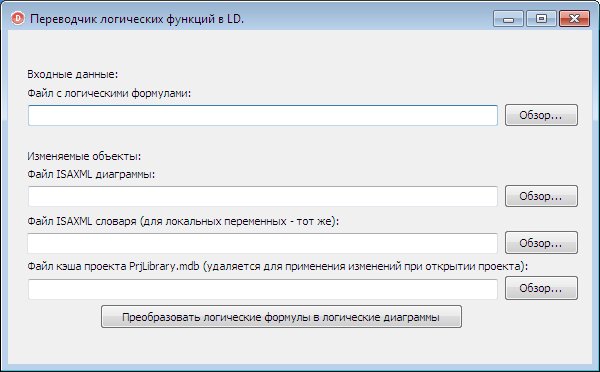
Далее нужно открыть созданную программу Prog1 для автоматической инициализации программы.



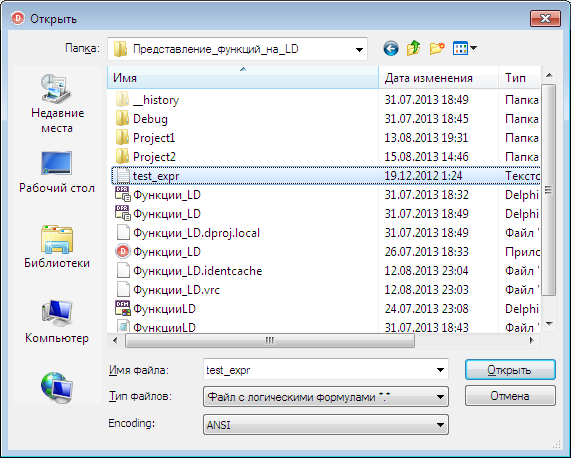
Затем следует сохранить и закрыть проект, в результате чего будет создан Шаблонный файл Prog1.isaxml, подлежащий заполнению программой Функции\_LD.

###### Работа с программой

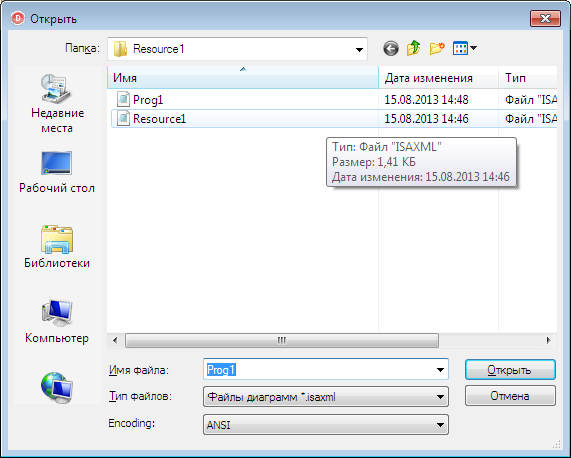
Окно программы Функции\_LD имеет вид:



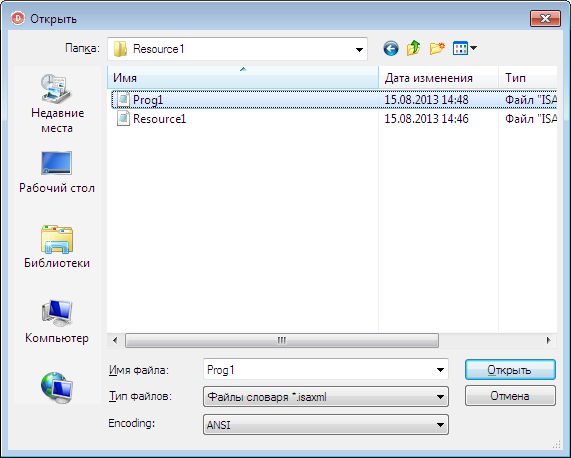
Открываем тестовый файл с логическими формулами:



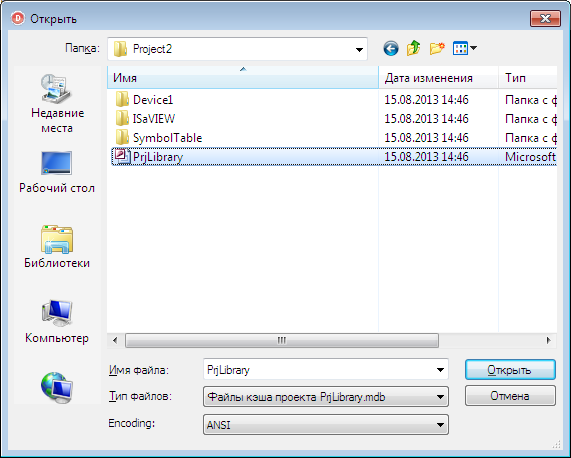
Открываем файл шаблона диаграмм LD:



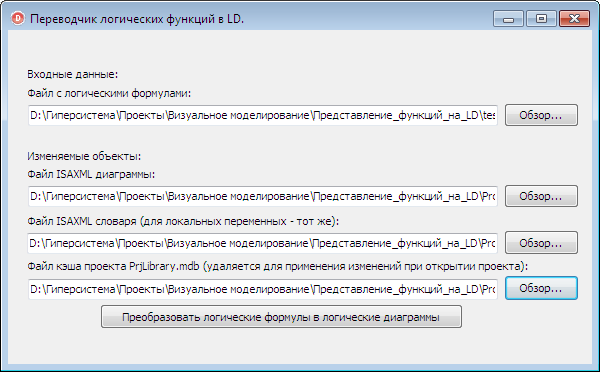
Открываем файл шаблона словаря:



Открываем файл кэша проекта:



Главное окно после выбора файлов имеет вид:

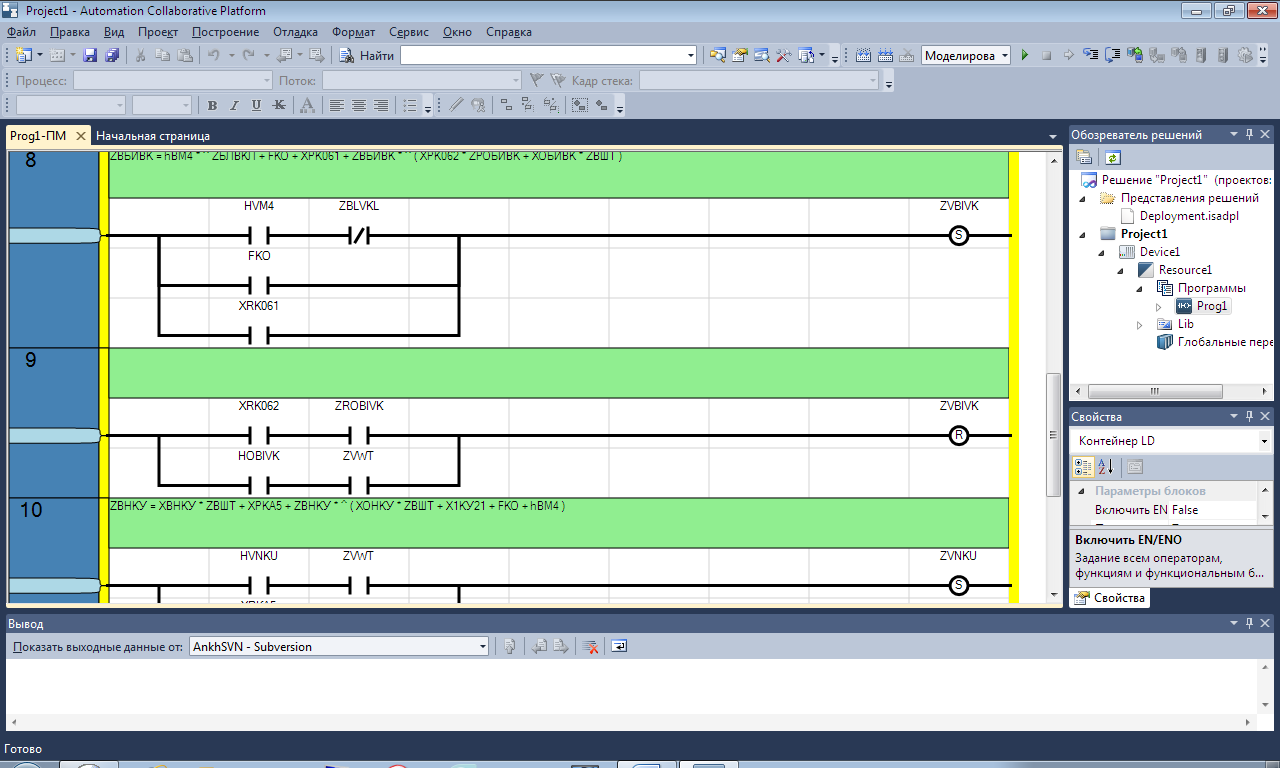


Выполняем команду *Преобразовать логические формулы в логические диаграммы.* На этом работа с программой *Функции\_LD* завершена.

##### Визуальное моделирование логических функций

После запуска среды IsaGraf можно просматривать созданную LD-программу в разных вариантах:

* LD-программа в среде IsaGraf;



## Генерация словаря

Словарь проекта представляет собой набор глобальных или локальных булевых переменных. Локальные переменные хранятся в секции LocalVars файлов программ:

<?xml version="1.0" encoding="utf-8"?>

<Pou FileVersion="200.00.000" Name="Prog1" ElementGuid="1137c031-7e8a-4c2d-85fd-e8102f55e3d1" Comment="" SingleTextLine="" Language="LD" CustomFbd="0" Localization="Prj" GroupName="(Определяемый пользователем)" GenDebugInfo="false" UseResPassword="true" GenerateSymbMon="true" Is1499="false">

<Program />

**<LocalVars>**

**<Variable Name="Z1P" DataType="BOOL" InitialValue="" Comment="Z1П" Address="" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />**

**<Variable Name="F1P1\_2" DataType="BOOL" InitialValue="" Comment="F1П1\_2" Address="" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />**

**<Variable Name="F1P1\_3" DataType="BOOL" InitialValue="" Comment="F1П1\_3" Address="" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />**

**<Variable Name="Z2P" DataType="BOOL" InitialValue="" Comment="Z2П" Address="" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />**

**<Variable Name="F2P1\_2" DataType="BOOL" InitialValue="" Comment="F2П1\_2" Address="" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />**

**<Variable Name="F2P1\_3" DataType="BOOL" InitialValue="" Comment="F2П1\_3" Address="" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />**

**<Variable Name="Z3P" DataType="BOOL" InitialValue="" Comment="Z3П" Address="" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />**

**<Variable Name="F3P1\_2" DataType="BOOL" InitialValue="" Comment="F3П1\_2" Address="" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />**

**<Variable Name="F3P1\_3" DataType="BOOL" InitialValue="" Comment="F3П1\_3" Address="" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />**

**<Variable Name="ZVBIVK" DataType="BOOL" InitialValue="" Comment="ZВБИВК" Address="" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />**

**<Variable Name="HVM4" DataType="BOOL" InitialValue="" Comment="HВМ4" Address="" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />**

**<Variable Name="ZBLVKL" DataType="BOOL" InitialValue="" Comment="ZБЛВКЛ" Address="" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />**

**<Variable Name="FKO" DataType="BOOL" InitialValue="" Comment="FКО" Address="" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />**

**<Variable Name="XRK061" DataType="BOOL" InitialValue="" Comment="XРК061" Address="" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />**

**<Variable Name="XRK062" DataType="BOOL" InitialValue="" Comment="XРК062" Address="" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />**

**<Variable Name="ZROBIVK" DataType="BOOL" InitialValue="" Comment="ZРОБИВК" Address="" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />**

**<Variable Name="HOBIVK" DataType="BOOL" InitialValue="" Comment="ХОБИВК" Address="" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />**

**<Variable Name="ZVWT" DataType="BOOL" InitialValue="" Comment="ZВШТ" Address="" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />**

**<Variable Name="ZVNKU" DataType="BOOL" InitialValue="" Comment="ZВНКУ" Address="" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />**

**<Variable Name="HVNKU" DataType="BOOL" InitialValue="" Comment="ХВНКУ" Address="" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />**

**<Variable Name="XRKA5" DataType="BOOL" InitialValue="" Comment="XРКA5" Address="" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />**

**<Variable Name="HONKU" DataType="BOOL" InitialValue="" Comment="ХОНКУ" Address="" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />**

**<Variable Name="H1KU21" DataType="BOOL" InitialValue="" Comment="Х1КУ21" Address="" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />**

**</LocalVars>**

<PouBody>…</PouBody>

</Pou>

Глобальные переменные размещаются аналогичным образом в секции GlobalVars файлов ресурсов.

Генерация словаря осуществляется вместе с генерацией диаграмм в процедуре «ЗаписьФайлов». В процедуре сравниваются имена указанных для словаря и для диаграммы файлов. Если имена файлов совпадают, значит требуется создать локальные переменные в файле с диаграммой:

procedure TForm1.ЗаписьФайлов();

var

TS\_src,TS\_res\_varst,TS\_res\_vars,TS\_res\_diag,TS\_vars,TS\_diag: TStringList;

i,l: integer;

TS : TStringList;

s\_all,S\_src: string;

Str\_result: string;

TF\_src: TextFile;

VarsIndex, PouBodyIndex: integer;

begin

TS\_res\_vars := TStringList.Create();

TS\_res\_varst := TStringList.Create();

TS\_res\_diag := TStringList.Create();

TS\_src := TStringList.Create();

s\_all := '';

AssignFile(TF\_src, Edit1.Text);

reset(TF\_src);

l:=0;

while not eof(TF\_src) do begin

readln(TF\_src, S\_src); // в переменную str будут сохраняться строки из файла... по очереди...

Trim(S\_src); // удаление граничных пробелов

TS\_src.Insert(TS\_src.Count,S\_src);

S\_src := AnsiUpperCase(S\_src); // перевод в верхний регистр

TS\_res\_diag.Add(СтрокаВДиаграмме(S\_src)); //преобразование функции в STF-формат

s\_all := s\_all + s\_src + ' ';

end;

CloseFile(TF\_src);

TS := TStringList.Create();

TS.Delimiter := ' ';

TS.DelimitedText := s\_all;

//----------------копируем уникальные переменные в TS\_result

for i := 0 to TS.Count - 1 do

begin

if (TS.Strings[i] <> '=') and

(TS.Strings[i] <> '(') and

(TS.Strings[i] <> ')') and

(TS.Strings[i] <> '+') and

(TS.Strings[i] <> '\*') and

(TS.Strings[i] <> '^') and

(TS\_res\_vars.IndexOf(TS.Strings[i]) = -1) then begin

TS\_res\_vars.Add(TS.Strings[i]);

TS\_res\_varst.Add(Транслитерация(TS.Strings[i]));

end;

end;

//--------------чтение указанного файла словаря-----------

TS\_vars := TStringList.Create();

TS\_vars.LoadFromFile(Edit3.Text);

for i := 0 to TS\_vars.Count-1 do

TS\_vars[i]:=Trim(TS\_vars[i]);

//--------------запись переменных в файл-----------

if Edit2.Text=Edit3.Text then begin //проверка: хранятся ли переменные в том же файле, что и программа

VarsIndex:=TS\_vars.IndexOf('<LocalVars />')+1;

if VarsIndex<>0 then begin

TS\_vars[VarsIndex-1]:='<LocalVars>';

TS\_vars.Insert(VarsIndex,'</LocalVars>')

end

else begin

VarsIndex:=TS\_vars.IndexOf('</LocalVars>');

end;

end

else begin

VarsIndex:=TS\_vars.IndexOf('<GlobalVars />')+1;

if VarsIndex<>0 then begin

TS\_vars[VarsIndex-1]:='<GlobalVars>';

TS\_vars.Insert(VarsIndex,'</GlobalVars>');

end

else

VarsIndex:=TS\_vars.IndexOf('</GlobalVars>');

end;

**for i := TS\_res\_vars.Count - 1 downto 0 do**

**begin**

**TS\_vars.Insert( VarsIndex,'<Variable Name="'+TS\_res\_varst[i]+'" DataType="BOOL" InitialValue="" Comment="'+TS\_res\_vars[i]+'" Address="" Modifier="NonRetain" Kind="Var" Alias="" AccessRights="ReadWrite" StringSize="0" />');**

**end;**

TS\_res\_vars.Free;

TS\_res\_varst.Free;

TS\_vars.SaveToFile(Edit3.Text);

TS\_vars.Free;

…

end;

Созданные переменные можно просматривать и редактировать внутренним редактором среды IsaGraf (рисунок 4.1).

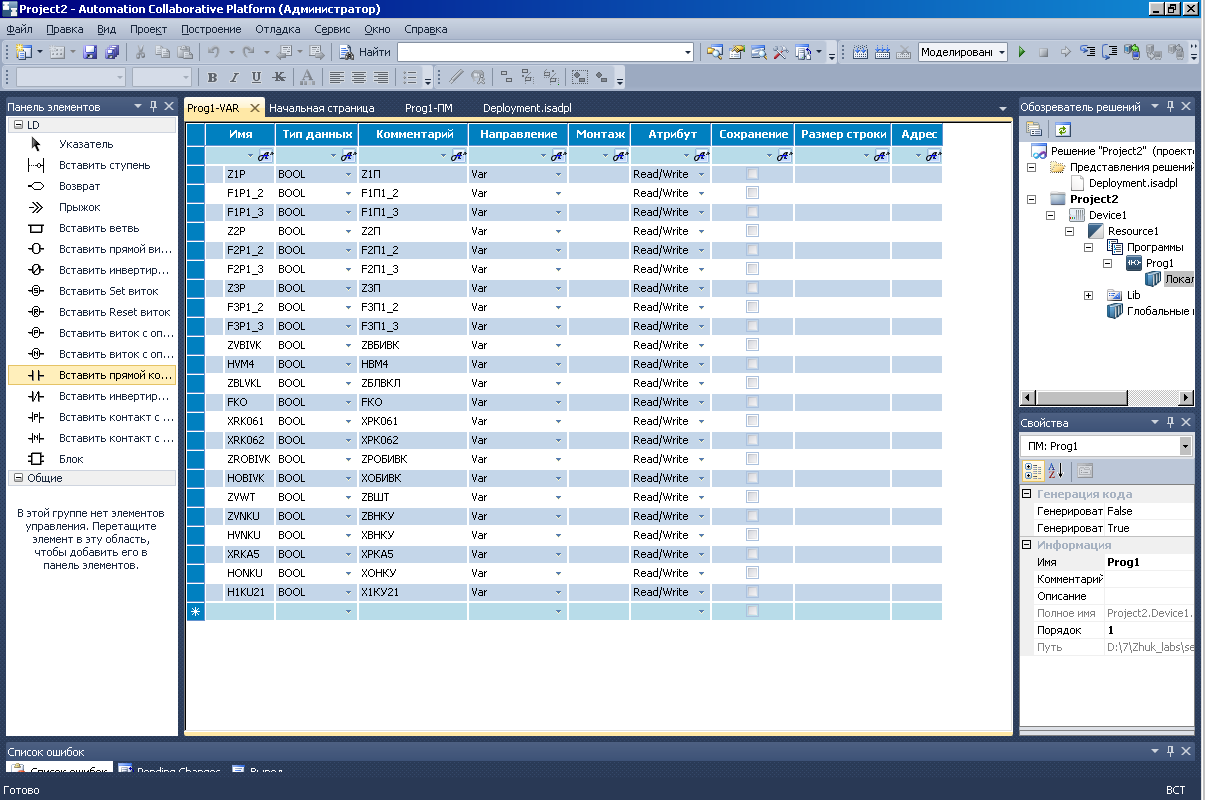


Рисунок 4.1. Словарь в редакторе IsaGraf

## Генерация диаграмм

Диаграммы хранятся в файлах программ в секции Pou:

<?xml version="1.0" encoding="utf-8"?>

<Pou FileVersion="200.00.000" Name="Prog1" ElementGuid="1137c031-7e8a-4c2d-85fd-e8102f55e3d1" Comment="" SingleTextLine="" Language="LD" CustomFbd="0" Localization="Prj" GroupName="(Определяемый пользователем)" GenDebugInfo="false" UseResPassword="true" GenerateSymbMon="true" Is1499="false">

<Program />

…

**<PouBody><![CDATA[PROGRAM Prog1**

**#info= QLD**

**BOF**

**(\*\*)**

**SOR [0,1] (\*\*) (\*\*)**

**EOR [1,0]**

**(\*Z1П = F1П1\_2 + Z1П \* ^ F1П1\_3\*)**

**SOR [0,2] (\*\*) (\*\*) XIC [1,0] (\*F1P1\_2\*) (\*F1П1\_2\*) OTS [2,0] (\*Z1P\*) (\*Z1П\*) EOR [3,0] (\*\*) SOR [0,2] (\*\*) (\*\*) XIC [1,0] (\*F1P1\_3\*) (\*F1П1\_3\*) OTR [2,0] (\*Z1P\*) (\*Z1П\*) EOR [3,0]**

**(\*Z2П = F2П1\_2 + Z2П \* ^ F2П1\_3\*)**

**SOR [0,3] (\*\*) (\*\*) XIC [1,0] (\*F2P1\_2\*) (\*F2П1\_2\*) OTS [2,0] (\*Z2P\*) (\*Z2П\*) EOR [3,0] (\*\*) SOR [0,3] (\*\*) (\*\*) XIC [1,0] (\*F2P1\_3\*) (\*F2П1\_3\*) OTR [2,0] (\*Z2P\*) (\*Z2П\*) EOR [3,0]**

**(\*Z3П = F3П1\_2 + Z3П \* ^ F3П1\_3\*)**

**SOR [0,4] (\*\*) (\*\*) XIC [1,0] (\*F3P1\_2\*) (\*F3П1\_2\*) OTS [2,0] (\*Z3P\*) (\*Z3П\*) EOR [3,0] (\*\*) SOR [0,4] (\*\*) (\*\*) XIC [1,0] (\*F3P1\_3\*) (\*F3П1\_3\*) OTR [2,0] (\*Z3P\*) (\*Z3П\*) EOR [3,0]**

**(\*ZВБИВК = hВМ4 \* ^ ZБЛВКЛ + FКО + XРК061 + ZВБИВК \* ^ ( XРК062 \* ZРОБИВК + ХОБИВК \* ZВШТ )\*)**

**SOR [0,5] (\*\*) (\*\*) BST BST XIC [1,0] (\*HVM4\*) (\*HВМ4\*) XIO [2,0] (\*ZBLVKL\*) (\*ZБЛВКЛ\*) NXB XIC [1,1] (\*FKO\*) (\*FКО\*) BND NXB XIC [1,1] (\*XRK061\*) (\*XРК061\*) BND OTS [2,0] (\*ZVBIVK\*) (\*ZВБИВК\*) EOR [3,0] (\*\*) SOR [0,5] (\*\*) (\*\*) BST XIC [1,0] (\*XRK062\*) (\*XРК062\*) XIC [2,0] (\*ZROBIVK\*) (\*ZРОБИВК\*) NXB XIC [1,1] (\*HOBIVK\*) (\*ХОБИВК\*) XIC [2,1] (\*ZVWT\*) (\*ZВШТ\*) BND OTR [3,0] (\*ZVBIVK\*) (\*ZВБИВК\*) EOR [4,0]**

**(\*ZВНКУ = ХВНКУ \* ZВШТ + XРКA5 + ZВНКУ \* ^ ( ХОНКУ \* ZВШТ + Х1КУ21 + FКО + hВМ4 )\*)**

**SOR [0,6] (\*\*) (\*\*) BST XIC [1,0] (\*HVNKU\*) (\*ХВНКУ\*) XIC [2,0] (\*ZVWT\*) (\*ZВШТ\*) NXB XIC [1,1] (\*XRKA5\*) (\*XРКA5\*) BND OTS [2,0] (\*ZVNKU\*) (\*ZВНКУ\*) EOR [3,0] (\*\*) SOR [0,6] (\*\*) (\*\*) BST BST BST XIC [1,0] (\*HONKU\*) (\*ХОНКУ\*) XIC [2,0] (\*ZVWT\*) (\*ZВШТ\*) NXB XIC [1,1] (\*H1KU21\*) (\*Х1КУ21\*) BND NXB XIC [1,1] (\*FKO\*) (\*FКО\*) BND NXB XIC [1,1] (\*HVM4\*) (\*HВМ4\*) BND OTR [2,0] (\*ZVNKU\*) (\*ZВНКУ\*) EOR [3,0]**

**EOF**

**#end\_info**

**END\_PROGRAM]]></PouBody>**

**</Pou>**

Генерация диаграмм осуществляется при помощи разработанных ранее процедур в процедуре «Запись файлов»:

procedure TForm1.ЗаписьФайлов();

var

TS\_src,TS\_res\_varst,TS\_res\_vars,TS\_res\_diag,TS\_vars,TS\_diag: TStringList;

i,l: integer;

TS : TStringList;

s\_all,S\_src: string;

Str\_result: string;

TF\_src: TextFile;

VarsIndex, PouBodyIndex: integer;

begin

TS\_res\_vars := TStringList.Create();

TS\_res\_varst := TStringList.Create();

TS\_res\_diag := TStringList.Create();

TS\_src := TStringList.Create();

s\_all := '';

AssignFile(TF\_src, Edit1.Text);

reset(TF\_src);

l:=0;

while not eof(TF\_src) do begin

readln(TF\_src, S\_src); // в переменную str будут сохраняться строки из файла... по очереди...

Trim(S\_src); // удаление граничных пробелов

TS\_src.Insert(TS\_src.Count,S\_src);

S\_src := AnsiUpperCase(S\_src); // перевод в верхний регистр

TS\_res\_diag.Add(СтрокаВДиаграмме(S\_src)); //преобразование функции в STF-формат

s\_all := s\_all + s\_src + ' ';

end;

CloseFile(TF\_src);

…

//----------------чтение указанного файла диаграмм-----------

TS\_diag := TStringList.Create();

TS\_diag.LoadFromFile(Edit2.Text);

for i := 0 to TS\_vars.Count-1 do

TS\_diag[i]:=Trim(TS\_diag[i]);

//----------------определение положения тела программы в указанном файле диаграмм-----------

PouBodyIndex:=TS\_diag.IndexOf('EOF');

//----------------вставка ступеней диаграммы-----------

//Str\_result := УстановитьКоординаты(Str\_result);

**for i := TS\_res\_diag.Count - 1 downto 0 do begin**

**TS\_diag.Insert(PouBodyIndex, УстановитьКоординаты(TS\_res\_diag.Strings[i],i+2));**

**TS\_diag.Insert(PouBodyIndex,#13#10+'(\*'+TS\_src[i]+'\*)'+#13#10);**

**end;**

TS\_res\_diag.Free;

TS\_diag.SaveToFile(Edit2.Text);

TS\_diag.Free;

DeleteFile(Edit4.Text);