Файл **Models_ex_Lexer_Test.mbp** содержит примеры, демонстрирующие работу блока Lexer, его функциональные возможности .

Примеры предназначены для наглядного изучения пользователем функций блоков и возможности экспериментально усвоить, как работают приведённые в примере Models_ex_Lexer_Test.

Рассматриваемые в примере задачи:

- Арифметические операции;
- Математические операции с использованием блоков библиотеки «Математика»
- Ветвление IF и реализованная программа индикации значений 4х-значных чисел
  - а) LEXER с использованием IQ математики
  - б) LEXER с использование FLOAT математики
- Калькулятор, и на его базе, Расчет параметров ПИ регулятора.
- Формирование кривой задания по заданным точкам

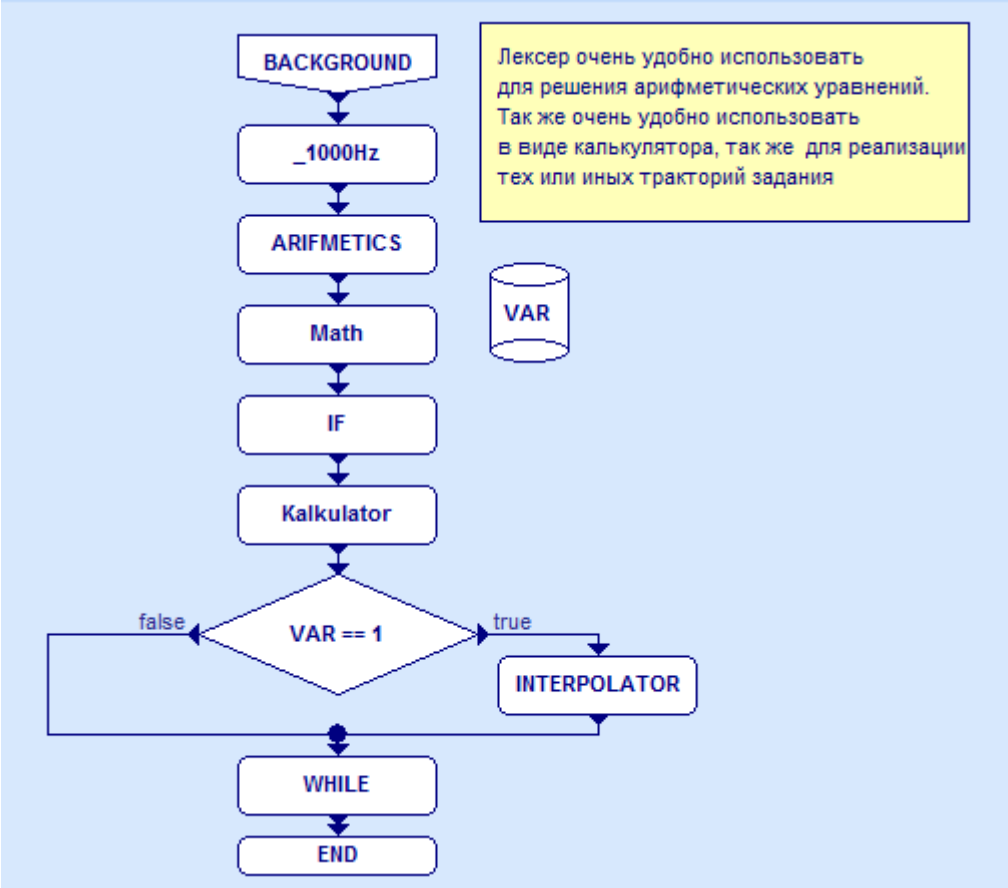

Рис. 1. Корневое поле набора

Формула **_1000Hz** выполняет функцию делителя частоты и создания прерывания на частоте 1000Гц.

ARIFMETICS

LEXER можно использовать для проведения простейших арифметических операций:

a * b - умножение;

a / b - деление;

a + b - арифметический плюс;

a -- b - арифметический минус;

-a – унарный минус;

+a – унарный плюс;

Пример реализации:

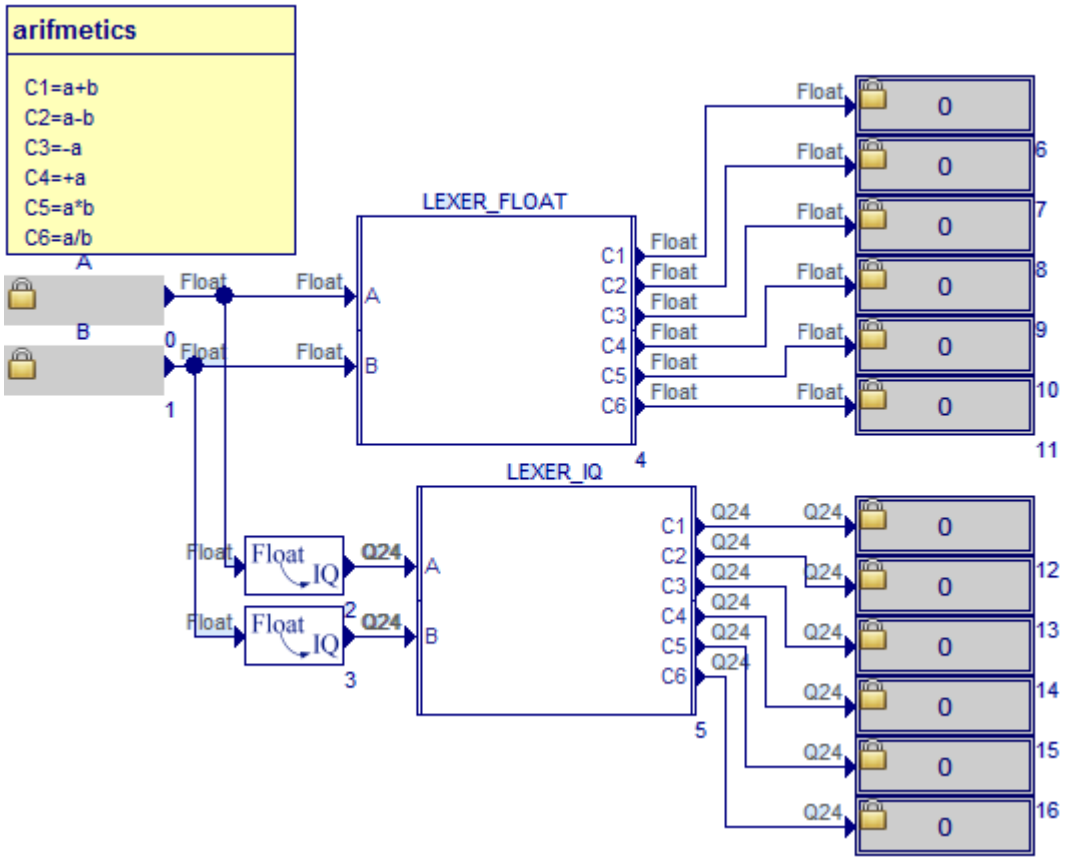

Рис. 2 простейшие арифметические операции

Задание для блока формируются переменными A и B в форматах FLOAT (с плавающей точкой) и IQ (фиксированная точка).

Для настройки блока Lexer нам необходимо объявит входные, выходные, локальные переменные. Особенность локальных переменных: при каждом обращении к блоку LEXER их значения обнуляются.

Текстовая интерпретация исполнительного кода выглядит следующим образом:

```
* Main program *)
(* Supported types: INT, FLOAT, Q1-Q30 *)
PROGRAM main
  (* Input variables *)
  INPUT_VAR
  A : FLOAT;
  B : FLOAT;
  END_VAR
```

```
  (* Output variables *)
  OUTPUT_VAR
  C1 : FLOAT;  (*арифм +*)
  C2 : FLOAT;  (*арифм -*)
  C3 : FLOAT;  (*унарный +*)
  C4 : FLOAT;  (*унарный -*)
  C5 : FLOAT;  (*умножение*)
  C6 : FLOAT;  (*деление*)
      END_VAR
  (* Local variables *)
  LOCAL_VAR
  TMP  :  FLOAT;
  END_VAR
  (* Body text *)
  C1 := A+B;
  C2 := A-B;
  TMP := A;
  TMP := -TMP;
  C3  := TMP;
  TMP := A;
  TMP := +TMP;
  C4  := TMP;
  C5  := A*B;
  C6  := A/B;
END_PROGRAM
```

MATH

LEXER позволяет интегрировать блоки из библиотек MexBIOSa, данная функция расширяет возможности его использования. В примере приведены математические операции с использованием блоков библиотеки «математика»

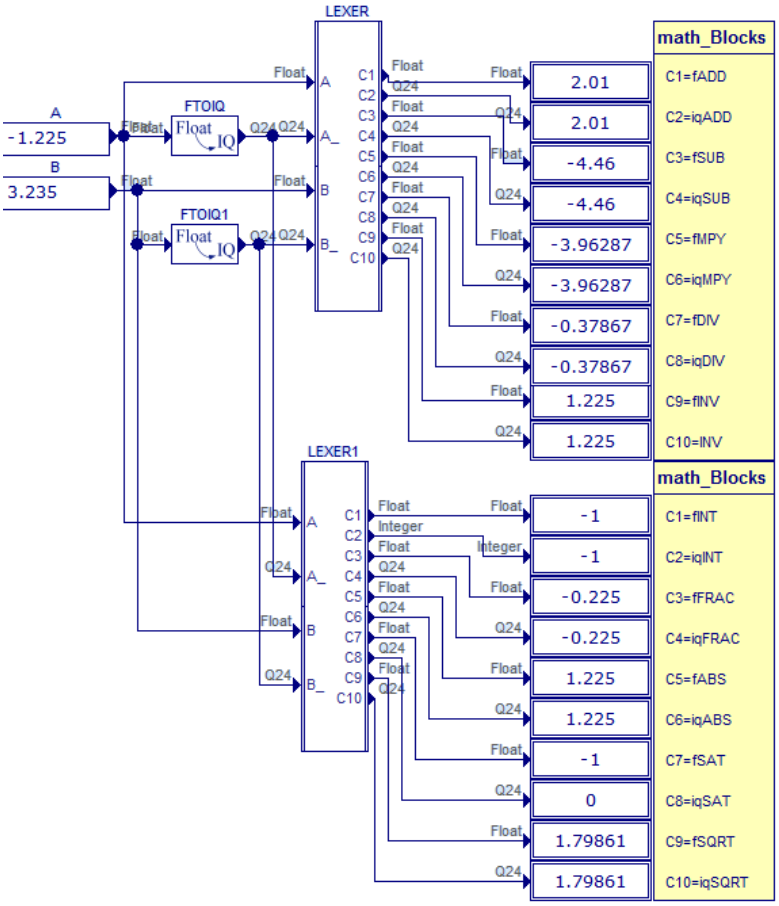

Рис. 3 Пример использования блоков библиотеки

Листинг программы блока LEXER использование блоков из библиотеки «математика»:

```
(* Main program *)
(* Supported types: INT, FLOAT, Q1-Q30 *)
PROGRAM main
  (* Input variables *)
  INPUT_VAR
  A : FLOAT;
  A_: Q24;
  B : FLOAT;
  B_: Q24;

  END_VAR
  (* Output variables *)
  OUTPUT_VAR
  C1 : FLOAT;   (*A+B*)
  C2 : Q24;     (*A+B Q24*)
  C3 : FLOAT;   (*A-B*)
```

```
  C4 : Q24;     (*A-B Q24*)
  C5 : FLOAT;   (*A*B*)
  C6 : Q24;     (*A*B Q24*)
  C7 : FLOAT;   (*A/B*)
  C8 : Q24;     (*A/B Q24*)
  C9 : FLOAT;   (*A=-A*)
  C10: Q24;     (*A=-A Q24*)
      END_VAR
  (* Local variables *)
  LOCAL_VAR
  END_VAR
  (* Body text *)
  C1:= fADD(A,B);
  C2:= ADD(A_,B_);
  C3:=fSUB(A,B);
  C4:=SUB(A_,B_);
  C5:= fMPY(A,B);
  C6:= iqMPY(A_,B_,INT(24));
  C7:= fDIV(A,B);
  C8:= iqDIV(A_,B_,INT(24));
  C9:= fINV(A);
  C10:= INV(A_);
END_PROGRAM
```

Листинг программы блока LEXER2:

```
(* Main program *)
(* Supported types: INT, FLOAT, Q1-Q30 *)
PROGRAM main
  (* Input variables *)
  INPUT_VAR
  A : FLOAT;
  A_: Q24;
  B : FLOAT;
  B_: Q24;
  END_VAR
  (* Output variables *)
  OUTPUT_VAR
  C1 : FLOAT;  (*INT(A)*)
  C2 : INT;  (*INT(A) INT*)
  C3 : FLOAT;  (*REAL(A)*)
  C4 : Q24;  (*REAL(A) Q24*)
  C5 : FLOAT;  (*ABS(A)*)
  C6 : Q24;  (*ABS(A) Q24*)
  C7 : FLOAT;(*SAT(A) *)
  C8 : Q24;  (*SAT(A) Q24*)
  C9 : FLOAT;(*SQRT(A) *)
  C10: Q24;  (*SQRT(A) *)

      END_VAR
  (* Local variables *)
  LOCAL_VAR
  END_VAR
  (* Body text *)

  C1:= fINT(A);
  C2:= iqINT(A_,INT(24));
  C3:=fFRAC(A);
```

```
  C4:=iqFRAC(A_,INT(24));
  C5:= fABS(A);
  C6:= ABS(A_);
  C7:= fSAT(A,1,-1);
  C8:= iqSAT(A_,INT(1),INT(-1),INT(24));
  C9:= fSQRT(B);
  C10:= iqSQRT(B_,INT(24));
END_PROGRAM
```

Реализация ветвлений IF в LEXER:

Рассмотрим пример реализации программы формирования сдвига точки для 4х символьного сегментного индикатора В двух вариантах. В формате Float, и в формате Q10.

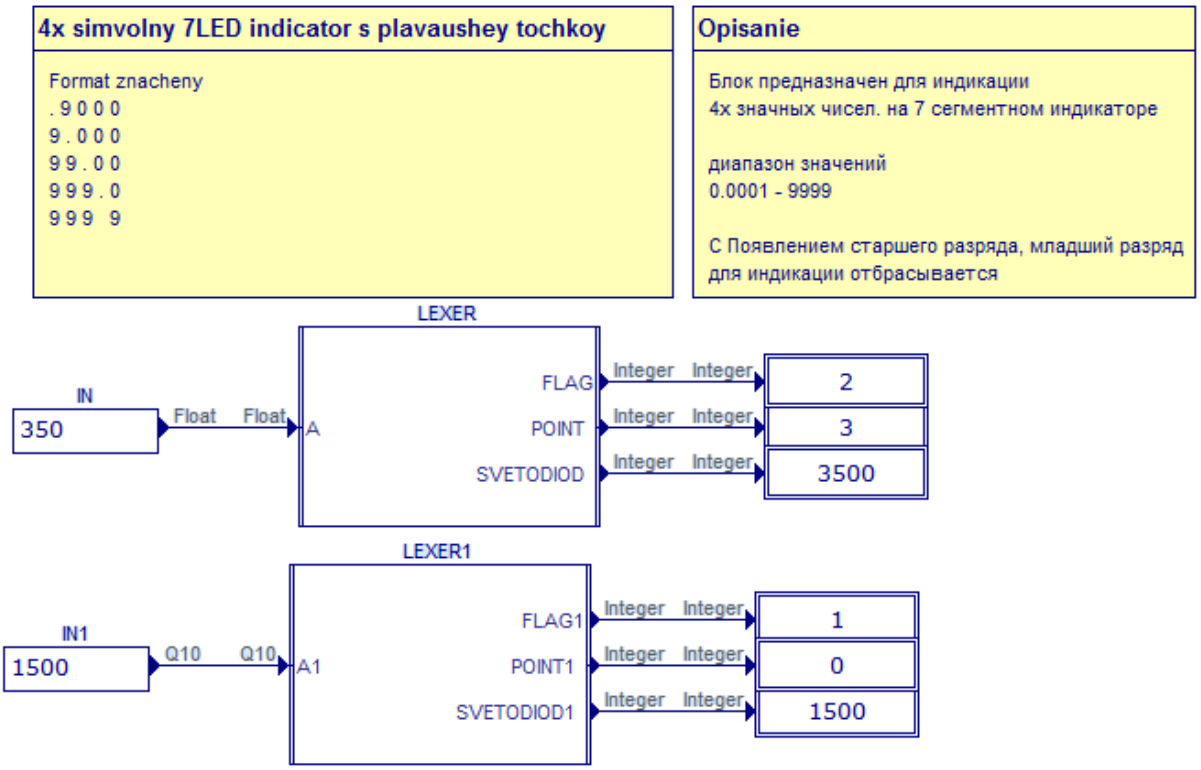

Рис. 4 Изображение: использования LEXER ветвления

Алгоритм программы можно описать следующим образом:

На вход блока подается значение в диапазоне [9999.999-0.001] на индикацию отправляются только 4 старших разряда, следовательно необходимо произвести сдвиг значений и выставить точки (POINT), отделяющие целую часть от дробной:

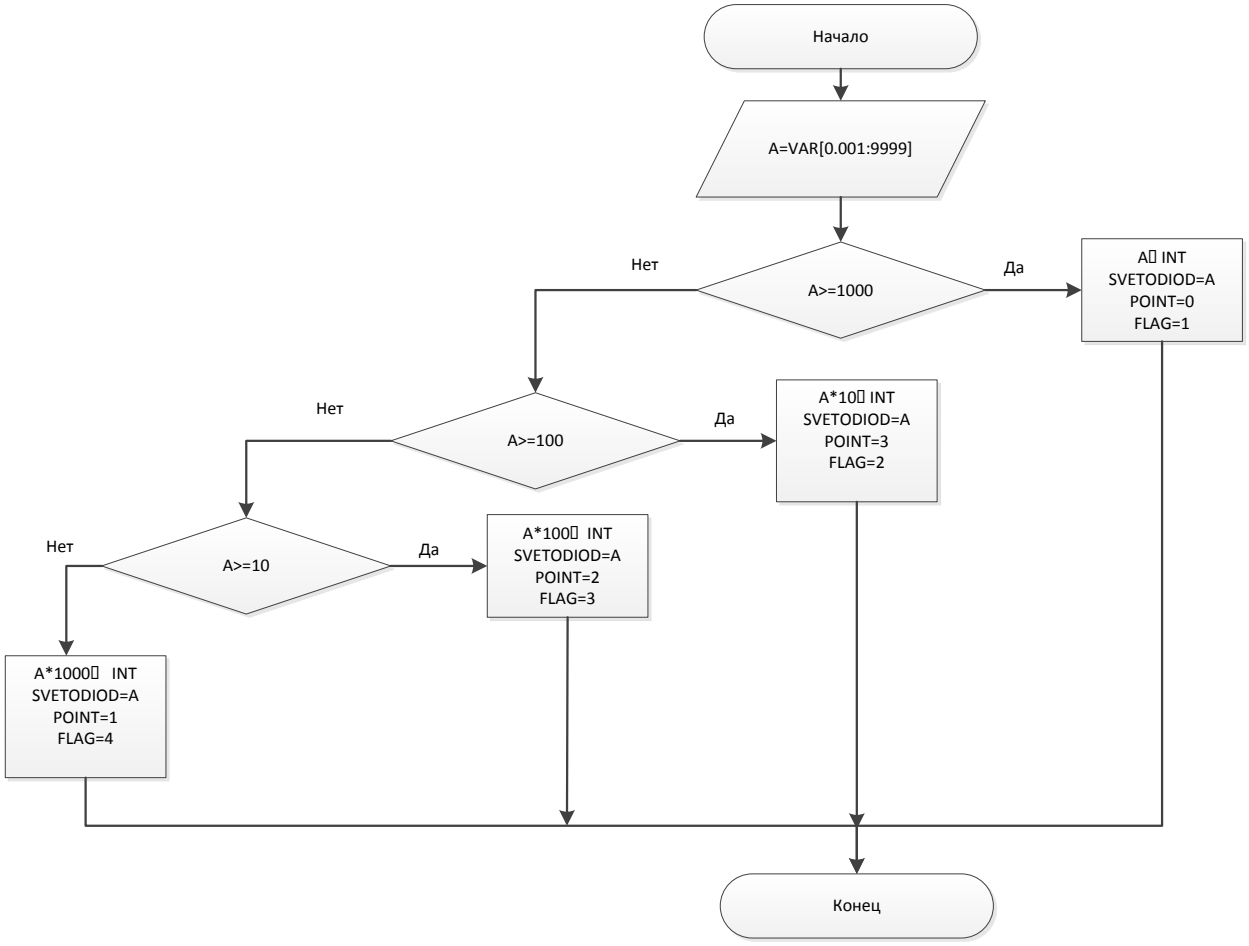

Рис. 5 Алгоритм работы программы формирования данных для 4х старших разрядов 7 сегментного индикатора

Текстовая реализация алгоритма представлена в программном коде LEXERa
В FLOAT реализации:

```
(* Main program *)
(* Supported types: INT, FLOAT, Q1-Q30 *)
PROGRAM main
  (* Input variables *)
  INPUT_VAR
  A : FLOAT;
  END_VAR
  (* Output variables *)
  OUTPUT_VAR
  FLAG       : INT;
  POINT      : INT;
  SVETODIOD  : INT;
  END_VAR
  (* Local variables *)
  LOCAL_VAR
  _1000 : FLOAT  :=1000;
  _100  : FLOAT  :=100;
  _10   : FLOAT  :=10;
  TMP_FLOAT   : FLOAT  :=0;
  TMP_INT     : INT    :=0;
```

```
    END_VAR
  (* Body text *)
  IF      A>=_1000 THEN FLAG:=INT(1); POINT:=INT(0);
  TMP_FLOAT:=A; TMP_INT:=FTOIQ(TMP_FLOAT,0); SVETODIOD:=TMP_INT;
  ELSE_IF A>=_100  THEN FLAG:=INT(2); POINT:=INT(3);
  TMP_FLOAT:=A*10; TMP_INT:=FTOIQ(TMP_FLOAT,0); SVETODIOD:=TMP_INT;
  ELSE_IF A>=_10   THEN FLAG:=INT(3); POINT:=INT(2);
  TMP_FLOAT:=A*100; TMP_INT:=FTOIQ(TMP_FLOAT,0); SVETODIOD:=TMP_INT;
  ELSE_IF A<_10    THEN FLAG:=INT(4); POINT:=INT(1);
 TMP_FLOAT:=A*1000; TMP_INT:=FTOIQ(TMP_FLOAT,0); SVETODIOD:=TMP_INT;
  END_IF
END_PROGRAM
```

В IQ реализации. (Так как диапазон входных значений принадлежит диапазону [9999-0.0001] целесообразно выбрать формат расчет в Q10):

```
(* Main program *)
(* Supported types: INT, FLOAT, Q1-Q30 *)
PROGRAM main
  (* Input variables *)
  INPUT_VAR
  A1 : Q10;
  END_VAR
  (* Output variables *)
  OUTPUT_VAR
  FLAG1       : INT;
  POINT1      : INT;
  SVETODIOD1  : INT;
  END_VAR
  (* Local variables *)
  LOCAL_VAR
  _1000 : Q10  :=1000;
  _100  : Q10  :=100;
  _10   : Q10  :=10;
  TMP_FLOAT   : Q10  :=0;
  TMP_INT     : INT  :=0;
  TMP_FRAQ    : Q10  :=0;

    END_VAR
  (* Body text *)

  IF      A1>=_1000 THEN FLAG1:=INT(1); POINT1:=INT(0);
  TMP_FLOAT:=A1;
  TMP_FRAQ:=Q10(0.5)+iqFRAC(TMP_FLOAT,INT(10));

  TMP_FLOAT:=iqINT(TMP_FLOAT,INT(10));
  TMP_FLOAT:=IQTOIQ(TMP_FLOAT,INT(0),INT(10));
  TMP_FLOAT:=TMP_FLOAT+TMP_FRAQ;
  TMP_INT:=iqINT(TMP_FLOAT,INT(10));

  SVETODIOD1:=TMP_INT;

  ELSE_IF A1>=_100  THEN FLAG1:=INT(2); POINT1:=INT(3);
  TMP_FLOAT:=iqMPY(A1,Q10(10),INT(10));
  TMP_FRAQ:=Q10(0.5)+iqFRAC(TMP_FLOAT,INT(10));

  TMP_FLOAT:=iqINT(TMP_FLOAT,INT(10));
  TMP_FLOAT:=IQTOIQ(TMP_FLOAT,INT(0),INT(10));
```

```
  TMP_FLOAT:=TMP_FLOAT+TMP_FRAQ;
  TMP_INT:=iqINT(TMP_FLOAT,INT(10));

  SVETODIOD1:=TMP_INT;

  ELSE_IF A1>=_10    THEN FLAG1:=INT(3); POINT1:=INT(2);
  TMP_FLOAT:=iqMPY(A1,Q10(100),INT(10));
  TMP_FRAQ:=Q10(0.5)+iqFRAC(TMP_FLOAT,INT(10));

  TMP_FLOAT:=iqINT(TMP_FLOAT,INT(10));
  TMP_FLOAT:=IQTOIQ(TMP_FLOAT,INT(0),INT(10));
  TMP_FLOAT:=TMP_FLOAT+TMP_FRAQ;
  TMP_INT:=iqINT(TMP_FLOAT,INT(10));

  SVETODIOD1:=TMP_INT;

  ELSE_IF A1<_10     THEN FLAG1:=INT(4);POINT1:=INT(1);
  TMP_FLOAT:=iqMPY(A1,Q10(1000),INT(10));
  TMP_FRAQ:=Q10(0.5)+iqFRAC(TMP_FLOAT,INT(10));

  TMP_FLOAT:=iqINT(TMP_FLOAT,INT(10));
  TMP_FLOAT:=IQTOIQ(TMP_FLOAT,INT(0),INT(10));
  TMP_FLOAT:=TMP_FLOAT+TMP_FRAQ;
  TMP_INT:=iqINT(TMP_FLOAT,INT(10));

  SVETODIOD1:=TMP_INT;
  END_IF
END_PROGRAM
```

KALKULATOR

LEXER удобно использовать в качестве калькулятора, для текстовой записи арифметических, математических формул. В Качестве примера приведем расчет параметров ПИД регулятора с коррекцией интегрального насыщения

Структурная схема ПИД регулятора с коррекцией интегрального насыщения представлена на Рис. 6 Структурная схема ПИД регулятора с коррекцией интегрального насыщения.

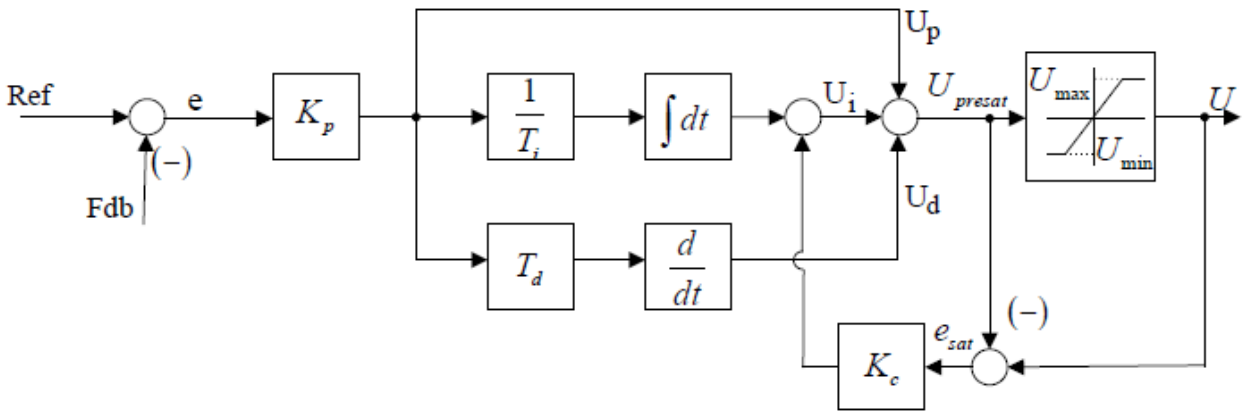

Рис. 6 Структурная схема ПИД регулятора с коррекцией интегрального насыщения.

Дифференциальное уравнение для ПИД регулятора имеет вид[1]:

$$U_{presat}(t) = U_p(t) + U_i(t) + U_d(T) \quad (1)$$

Каждое слагаемое представляет собой:

Пропорциональную составляющую:

$$U_p(t) = K_p \cdot e(t) \quad (2)$$

Дифференциальную составляющую:

$$U_i(t) = \frac{K_p}{T_i} \cdot \int_0^t e(\zeta) d\zeta + K_c \cdot (U(t) - U_{presat}(t)) \quad (3)$$

Выражения 1-4 могут быть дискретизованы согласно первой разностной схеме, тогда:

Уравнение ПИД регулятора:

$$U_{presat}(k) = U_p(k) + U_i(k) + U_d(k) \quad (4)$$

Пропорциональная составляющая:

$$U_p(k) = K_p \cdot e(k) \quad (5)$$

Интегральная составляющая с коррекцией интегрального насыщения

$$U_i(k) = U_i(k-1) + k_p \frac{T_d}{T} \cdot e(k) + K_c \cdot (U(k) - U_{presat}(k)) \quad (6)$$

Дифференциальная оставляющая:

$$U_i(k) = k_p \frac{T_d}{T} \cdot (e(k) - e(k-1)) \quad (7)$$

Для сокращения записи вводим коэффициент $k_i = \frac{T_s}{T_i}$ и $k_d = \frac{T_d}{T_s}$ тогда интегральная составляющая с коррекцией интегрального насыщения и дифференциальная составляющая примут вид:

$$U_i(k) = U_i(k-1) + k_i \cdot U_p(k) + k_c \cdot (U(k) - U_{presat}(k)) \quad (8)$$

$$U_d(k) = k_d \cdot (U_p(k) - U_p(k-1)) \quad (9)$$

Где T_s – время дискретизации (с)

Определение параметров коэффициентов усиления ПИД регулятора:

Параметры системы:

Таблица 1- Параметры системы

Ra, Ом	*La, Гн*	*f_t, Гц*	*k_{tp}*	*N_{zs}*	*I_{max}, A*
73	0.151	5000	46	10	1

Где

Ra,- сопротивление обмотки статора;

La – индуктивность обмотки статора;

f_t – частота расчет контура;

k_{tp} – коэффициент передачи преобразователя DC-bus;

N_{zs} – выход с датчика тока отсчетов в формате Q24;

I_{max} – максимальное значение тока в формате Q24;
Период расчета контура:

$$Ts = \frac{1}{f_t} = \frac{1}{5000} = 0.0002\,c \tag{10}$$

Постоянная времени обмотки

$$Ta = \frac{La}{Ra} = \frac{0.151}{73} = 0.00207\,c \tag{11}$$

Малая некомпенсируемая постоянная времени преобразователя:

$$Tmt = \frac{1}{f_t} = \frac{1}{2*5000} = 0.0001\,c \tag{12}$$

Расчет регулятора:
Коэффициент ОС по току:

$$k_t = \frac{Nz}{I_{\max}} = \frac{10}{1} = 10 \tag{13}$$

Постоянная времени контура:

Коэффициент усиления регулятора:

$$k_{rt} = \frac{T_{rt} \cdot Ra}{k_t \cdot 2 \cdot Tmt \cdot k_{tp}} \cdot = \frac{0.00207 \cdot 73}{10 \cdot 8 \cdot 0.0001 \cdot 46} = 0.415761 \tag{14}$$

$$k_i = \frac{Ts}{T_{rt}} = \frac{0.0002}{0.00207} = 0.0966 \tag{15}$$

Программная реализация в коде для LEXER

```
(* Main program *)
(* Supported types: INT, FLOAT, Q1-Q30 *)
PROGRAM main
  (* Input variables *)
  INPUT_VAR
  R : FLOAT;
  L : FLOAT;
  Kt: FLOAT;
  Ktp : FLOAT;
  Ts : FLOAT;


  END_VAR
  (* Output variables *)
  OUTPUT_VAR
  Kr : FLOAT;
  Ki : FLOAT;
  Kc : FLOAT;
  END_VAR
  (* Local variables *)
  LOCAL_VAR
  Ta : FLOAT:=1;
  Tmt: FLOAT:=1;
  END_VAR
  (* Body text *)
```

```
IF R<>0 THEN
IF L<>0 THEN
IF Ktp<>0 THEN
IF Kt<>0 THEN
IF Ts<>0 THEN

Ta:=L/R;
Tmt:=Ts/2;
Kr:=Ta*R/(Kt*Tmt*8*Ktp);
Ki:=Ts/Ta;
Kc:=Ts/Ta;
END_IF
END_IF
END_IF
END_IF
END_IF

END_PROGRAM
```

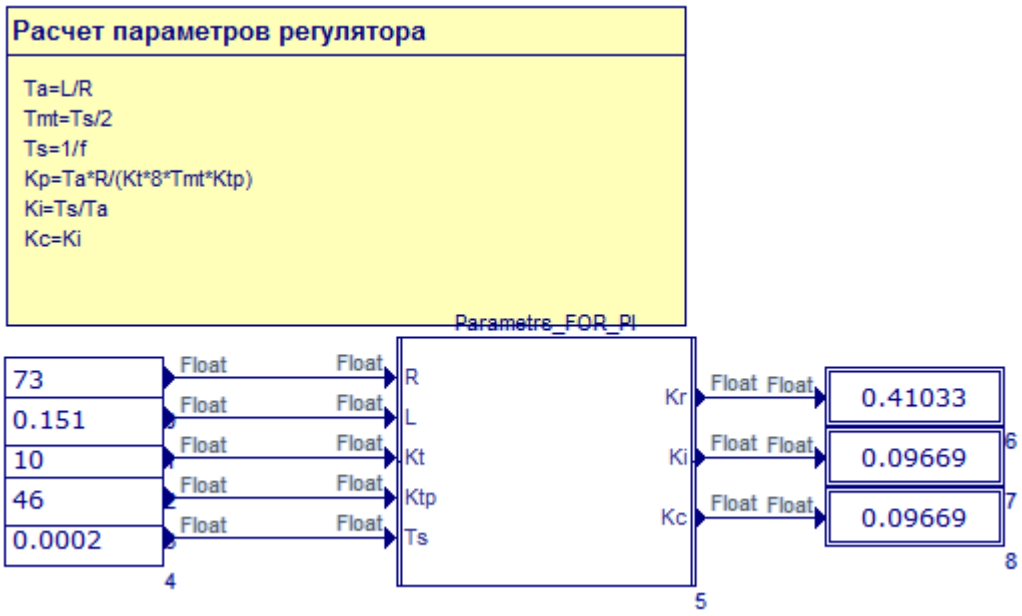

Рис. 7 Расчет параметров регулятора с использованием блока LEXER

INTERPOLATION

В качестве дополнительного примера применения ветвления IF в функциональных возможностях LEXER реализуем блок линейной интерполяции по заданным точкам и входному управляющему воздействию:

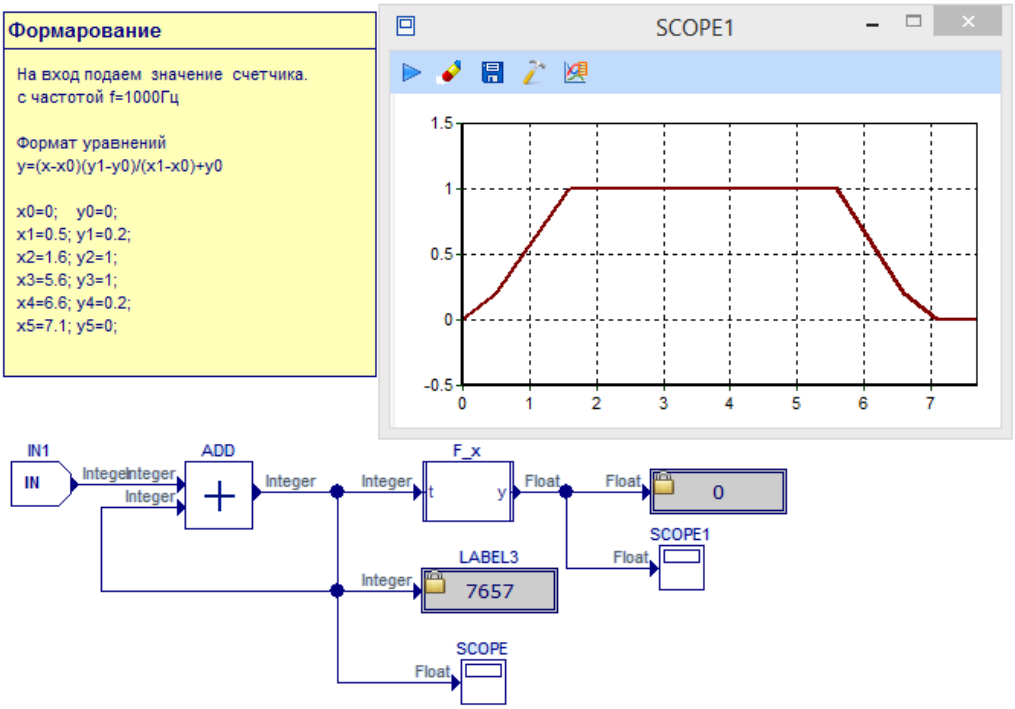

Рис. 8 Реализация блока Линейной интерполяции на базе функциональных возможностей LEXERa

Листинг программы:

```
(* Main program *)
(* Supported types: INT, FLOAT, Q1-Q30 *)
PROGRAM main
  (* Input variables *)
  INPUT_VAR
  t : INT;

  END_VAR
  (* Output variables *)
  OUTPUT_VAR
  y : FLOAT;
  END_VAR
  (* Local variables *)
  LOCAL_VAR
  x0: FLOAT:=0;
  x1: FLOAT:=0.5;
  x2: FLOAT:=1.6;
  x3: FLOAT:=5.6;
  x4: FLOAT:=6.6;
  x5: FLOAT:=7.1;
  y0: FLOAT:=0;
  y1: FLOAT:=0.2;
  y2: FLOAT:=1;
  y3: FLOAT:=1;
  y4: FLOAT:=0.2;
  y5: FLOAT:=0;

  tmp_float: FLOAT;
  tmp_int  : INT;

  tmp_x1    : FLOAT;
  tmp_x0    : FLOAT;
  tmp_y1    : FLOAT;
  tmp_y0    : FLOAT;

  END_VAR
  (* Body text *)
tmp_int:=t;
tmp_float:=IQTOF(tmp_int,0)/1000;

IF tmp_float<=x1 THEN
tmp_x0:=x0;
tmp_x1:=x1;
tmp_y0:=y0;
tmp_y1:=y1;
END_IF


IF tmp_float>x1 THEN IF tmp_float<=x2 THEN
tmp_x0:=x1;
tmp_x1:=x2;
tmp_y0:=y1;
tmp_y1:=y2;
END_IF
END_IF

IF tmp_float>x2 THEN IF tmp_float<=x3 THEN
```

```
tmp_x0:=x2;
tmp_x1:=x3;
tmp_y0:=y2;
tmp_y1:=y3;
END_IF
END_IF

IF tmp_float>x3 THEN IF tmp_float<=x4 THEN
tmp_x0:=x3;
tmp_x1:=x4;
tmp_y0:=y3;
tmp_y1:=y4;
END_IF
END_IF

IF tmp_float>x4 THEN IF tmp_float<=x5 THEN
tmp_x0:=x4;
tmp_x1:=x5;
tmp_y0:=y4;
tmp_y1:=y5;
END_IF
END_IF

IF tmp_float>x5 THEN
tmp_x0:=x5;
tmp_x1:=x5;
tmp_y0:=y5;
tmp_y1:=y5;

ELSE
y:=(tmp_float-tmp_x0)*(tmp_y1-tmp_y0)/(tmp_x1-tmp_x0)+tmp_y0;
END_IF

END_PROGRAM
```

Изменение входной величины реализовано на базе счетчика с частотой 1000Гц. Интерполяция осуществляется по формуле: y=(x-x0)(y1-y0)/(x1-x0)+y0. Данная функциональная возможность позволяет формировать задание, нагрузочные характеристики.

WHILE

Использование While в функционале LEXER позволяет проводить оценку выходных переменных, вести статистику. В качестве примера на использования функции WHILE в блоке LEXER рассмотрим подсчет суммы чисел данной формулы:

$$y = \sum_{k=1}^{n} \frac{1}{\mathrm{k}_n!} \qquad (16)$$

Структурный вид программы выглядит следующим образом:

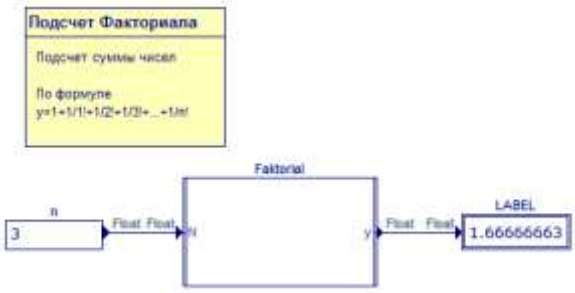

Рис. 9 Структурная схема подсчета суммы чисел

Листинг программы:

```
(* Main program *)
(* Supported types: INT, FLOAT, Q1-Q30 *)
PROGRAM main
  (* Input variables *)
  INPUT_VAR
N: FLOAT;
  END_VAR
  (* Output variables *)
  OUTPUT_VAR
y: FLOAT;
  END_VAR
  (* Local variables *)
  LOCAL_VAR
f1 : FLOAT:=1;
f1_tmp: FLOAT:=1;
y_tmp : FLOAT:=0;
  END_VAR
  (* Body text *)
WHILE N>=f1 THEN
f1_tmp:=f1_tmp* 1/f1;
y_tmp:=y_tmp+f1_tmp;
f1:=f1+1;
y:=y_tmp;
END_WHILE
END_PROGRAM
```