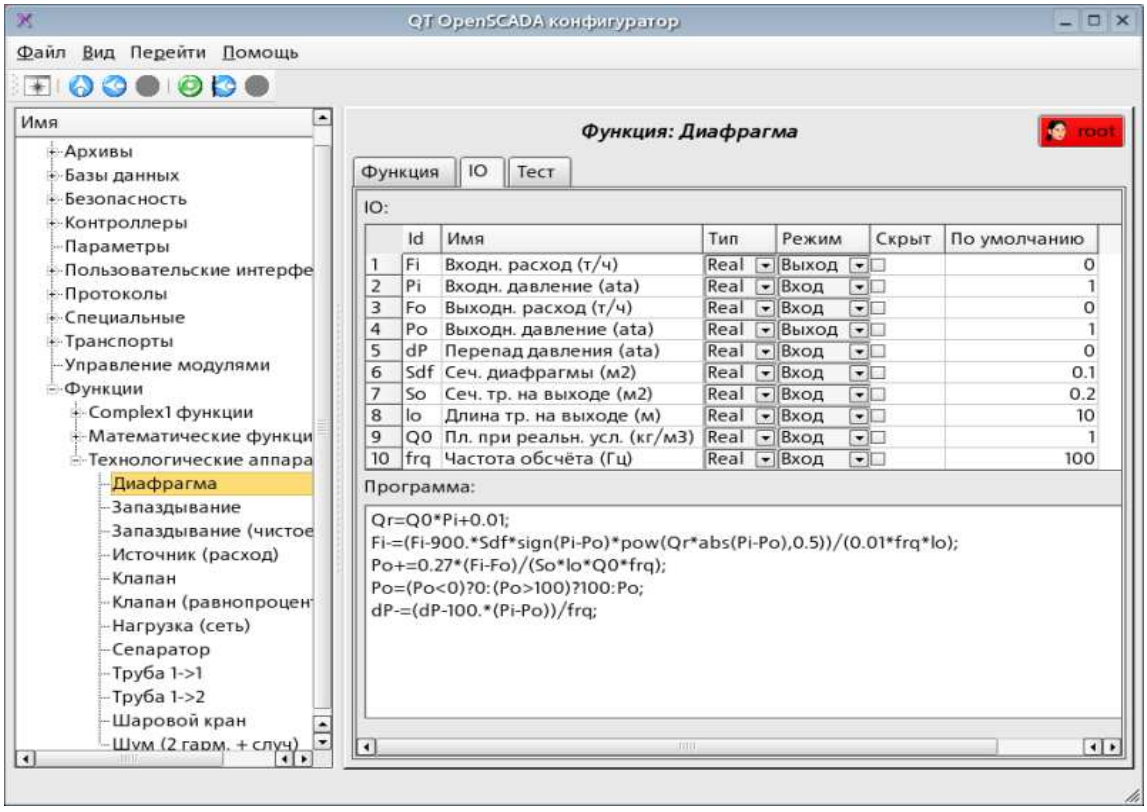
Модуль подсистемы “Контроллеры” <JavaLikeCalc>

Модуль:	JavaLikeCalc
Имя:	Вычислитель на Java-подобном языке.
Тип:	Контроллер
Источник:	cntr_JavaLikeCalc.so
Версия:	0.8.0
Автор:	Роман Савоченко
Описание:	Предоставляет основанные на java подобном языке вычислитель и движок библиотек. Пользователь может создавать и модифицировать функции и библиотеки.
Лицензия:	GPL

Модуль контроллера *JavaLikeCalc* предоставляет в систему механизм создания функций и их библиотек на Java-подобном языке. Описание функции на Java-подобном языке сводится к обвязке параметров функции алгоритмом. Кроме этого модуль наделен функциями непосредственных вычислений путём создания вычислительных контроллеров.

Непосредственные вычисления обеспечиваются созданием контроллера и связыванием его с функцией этого же модуля. Для связанной функции создаётся кадр значений над которым и выполняются периодические вычисления.

Параметры функции могут свободно создаваться, удаляться или модифицироваться. Текущая версия модуля поддерживает до 255 параметров функции в сумме с внутренними переменными. Вид редактора функций в конфигураторе QTCfg показано на рисунке.



После любого изменения программы или конфигурации выполняется перекомпиляция программы с упреждением связанных с функцией объектов значений TValCfg. Компилятор языка построен на известном генераторе грамматики «Bison» который совместим с не менее известной утилитой Yacc.

Язык использует неявное определение локальных переменных которое заключается в определении новой переменной в случае присваивания ей значения. Причём, тип локальной переменной устанавливается в соответствии с типом присваиваемого значения. Например выражение <Qr=Q0*Pi+0.01;> определит переменную Qr с типом переменной Q0.

В работе с различными типами данных язык использует механизм автоматического приведения типов в местах где подобное приведение является целесообразным.

Для комментирования участков кода в языке предусмотрены символы «//». Всё что идёт после данных символов до конца строки игнорируется компилятором.

В процессе генерации кода компилятор языка производит оптимизацию по константам и приведение типов констант к требуемому типу. Под оптимизацией констант подразумевается выполнение вычислений в процессе построения кода над двумя константами и вставка результата в код. Например выражение <y=pi*10;> свернётся в простое присваивание <y=31.4159;>. Под приведением типов констант к требуемому типу подразумевается формирования в коде константы, исключающей приведение типа в процессе исполнения. Например выражение <y=x*10> в случае вещественного типа переменной x преобразуется в <y=x*10.0>.

Язык поддерживает вызовы внешних и внутренних функций. Имя любой функции, вообще, воспринимается как символ, проверка на принадлежность которого к той или иной категории производится в следующем порядке:

- ключевые слова;
- константы;
- встроенные функции;
- внешние функции;
- известные переменные;
- параметры функции;
- создание внутренней переменной.

Формат вызова внешней функции имеет следующий вид: <библиотека.функция>. В случае вызова функции из текущей библиотеки имя библиотеки может опускаться.

1. Элементы языка

Ключевые слова: if, else, true, false.

Постоянные:

- десятичные: цифры 0–9 (12,111, 678);
- восьмеричные: цифры 0–7 (012, 011, 076);
- шестнадцатеричные: цифры 0–9, буквы a-f или A-F (0x12, 0XAB);
- вещественные: 345.23, 2.1e5, 3.4E-5, 3e6;
- логические: true, false;
- строковые: «hello».

Типы переменных:

- целое: -231…231;

- вещественное: 3.4 * 10308;
- логическое: false, true;
- строка: длина любая но без перехода на другую строку.

Встроенные константы: pi = 3.14159265, e = 2.71828182.

2. Операции языка

Операции поддерживаемые языком представлены в таблице ниже. Приоритет операций уменьшается с верху вниз. Операции с одинаковым приоритетом входят в одну цветовую группу.

Символ	Описание
()	Вызов функции.
{}	Программные блоки.
-	Унарный минус.
!	Логическое отрицание.
*	Умножение.
/	Деление.
%	Остаток от целочисленного деления.
+	Сложение
-	Вычитание
>	Больше
>=	Больше или равно
<	Меньше
<=	Меньше или равно
==	Равно
!=	Неравно
|	Поразрядное «ИЛИ»
&	Поразрядное «И»
^	Поразрядное «Исключающее ИЛИ»
&&	Логический «И»
||	Логический «ИЛИ»
?:	Условная операция (i=(i<0)?0:i;)
=	Присваивание.
+=	Присваивание с сложением.
-=	Присваивание с вычитанием.
*=	Присваивание с умножением.
/=	Присваивание с делением.

3. Встроенные функции языка

Для обеспечения высокой скорости работы в математических вычислениях модуль

предоставляет встроенные математические функции которые вызываются на уровне команд виртуальной машины. Встроенные математические функции:

- sin(x) – синус x;
- cos(x) – косинус x;
- tan(x) – тангенс x;
- sinh(x) – синус гиперболический от x;
- cosh(x) – косинус гиперболический от x;
- tanh(x) – тангенс гиперболический от x;
- asin(x) – арксинус от x;
- acos(x) – арккосинус от x;
- atan(x) – арктангенс от x;
- rand(x) – случайное число от 0 до x;
- lg(x) – десятичный логарифм от x;
- ln(x) – натуральный логарифм от x;
- exp(x) – экспонента от x;
- pow(x,y) – возведение x в степень y;
- sqrt(x) – корень квадратный от x;
- abs(x) – абсолютное значение от x;
- sign(x) – знак числа x;
- ceil(x) – округление числа x до большего целого;
- floor(x) – округление числа x до меньшего целого.

4. Операторы языка

Языком модуля поддерживаются два типа условных операторов. Первый это условный оператор для использования внутри выражения второй – глобальный.

Условный оператор для использования внутри выражения строится на операциях «?» и «:». В качестве примера можно записать следующее практическое выражение <st_open=(pos>=100)?true:false;> что читается как «Если переменная pos больше или равна 100 то переменной st_open присваивается значение true иначе false.

Глобальное условие строится на основе ключевых слов «if» и «else». В качестве примера можно привести тоже выражение выражение но другим способом <if (pos>100) st_open=true; else st_open=false;>. Как видно выражение записано по другому но читается также.

5. Примеры программы на языке

Приведём несколько примеров програм:

```
//Модель хода исполнительного механизма шарового крана
if( !(st_close && !com) && !(st_open && com) )
{
  tmp_up=(pos>0&&pos<100)?0:(tmp_up>0&&lst_com==com)?tmp_up-1./frq:t_up;
  pos+=(tmp_up>0)?0:(100.*(com?1.:-1.))/(t_full*frq);
  pos=(pos>100)?100:(pos<0)?0:pos;
  st_open=(pos>=100)?true:false;
  st_close=(pos<=0)?true:false;
  lst_com=com;
}

//Модель клапана
Qr=Q0+Q0*Kpr*(Pi-1)+0.01;
Sr=(S_kl1*l_kl1+S_kl2*l_kl2)/100.;
Ftmp=(Pi>2.*Po)?Pi*pow(Q0*0.75/Ti,0.5):(Po>2.*Pi)?Po*pow(Q0*0.75/To,0.5):pow(abs
(Q0*(pow(Pi,2)-pow(Po,2))/Ti),0.5);
Fi-=(Fi-7260.*Sr*sign(Pi-Po)*Ftmp)/(0.01*lo*frq);
Po+=0.27*(Fi-Fo)/(So*lo*Q0*frq);
Po=(Po<0)?0:(Po>100)?100:Po;
To+=(abs(Fi)*(Ti*pow(Po/Pi,0.02)-To)+(Fwind+1)*(Twind-To)/Riz)/(Ct*So*lo*Qr*frq);
```