ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»
МОСКОВСКИЙ ИНСТИТУТ ЭЛЕКТРОНИКИ И МАТЕМАТИКИ им. А.Н. ТИХОНОВА

Руководство программиста на программный модуль

«Программная реализация клеточного автомата Game of Life на базе парадигмы с управлением потоком данных»

Листов7

Руководитель: Салибекян С.М.

Задание принято к исполнению:

Студент группы МКС221 Кутовенко О.А.

Москва 2024

1. Назначение и условия применения программы

Программный модуль представляет собой имитационную модель вычислительного процесса реализации клеточного автомата игра «Жизнь». Данный модуль является составной частью среды ОА программирования и моделирования, эмулирующая работу ОА вычислительной системы. Управление моделью осуществляется с помощью специализированного языка программирования (ОА-язык); компилятор языка входит в состав ОА среды программирования и моделирования. Управление программным модулем (описание конфигурации моделируемой вычислительной системы и загрузка ее параметров, загрузка начальных данных для вычисления, запуск вычисления, запрос выходных данных) осуществляется с помощью специализированного языка. Программный модуль работает на IBM-совместимом компьютере под управлением операционной системы Windows версии 7 и выше. Объем оперативной памяти не менее 1 Гб, объем свободного пространства на жестком диске не менее 2 Гб. Для работы с программным модулем необходимо запустить среду ОА программирования и моделирования.

2. Характеристики программы

Управление программным модулем происходит с помощью специализированного языка программирования (ОА-язык). Компилятор языка входит в состав ОА среды программирования и моделирования. Модель представляет собой набор виртуальных устройств (функциональных устройств (ФУ)), обменивающихся между собой информацией. Модель включает в себя два типы ФУ: менеджер поля АЛУ и потоковый целочисленный АЛУ. Эти устройства служат для организации процесса моделирования. Если они не включены в состав модели, то симуляция клеточного автомата на модели вычислительной системы с управлением потоком данных не производится.

Программный модуль осуществляет моделирование клеточного автомата для заранее заданного числа шагов K. Прочие параметры моделируемой вычислительной системы и моделируемого вычислительного процесса вводятся в программе на специализированном языке (ОА-язык). Например, среди таких параметров N и M – высота и ширина поля соответственно. Программа дает возможность измерения времени выполнения программы.

Параметры имитационной модели задаются в программе на специализированном языке (АО-языка). Настройка вывода результатов работы алгоритма и полученных характеристик имитируемого вычислительного процесса также осуществляется посредством языка программирования.

Программа имеет возможность выводить сообщения об ошибках. Настройка реакции модели на ошибку (например, вывод соответствующего текстового сообщения) также осуществляется с помощью языка программирования. Представление результатов моделирования производится отдельной программой. Программа способна моделировать параллельный вычислительный процесс на многопроцессорной вычислительной системе.

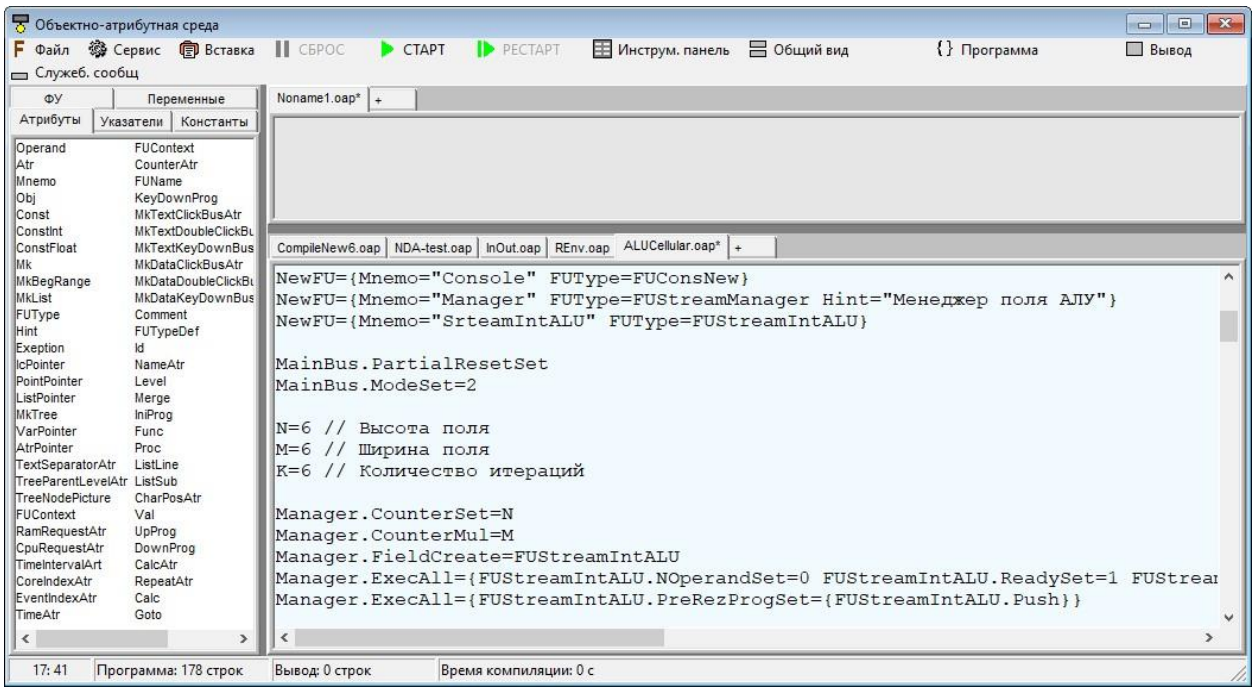

Рис. 1. Среда ОА программирования и моделирования

3. Обращение к программе

Программа разделена на три модуля: компилятор ОА-языка, Программа моделирования, и программа визуализации результатов моделирования.

Для написания программы на ОА-языке применяется программа millicom.exe, представляющая собой текстовый среду разработки программ для ОА вычислительной системы. Интерфейс среды разработки представлен на рис. 1. Внизу расположено поле программы, сверху поле вывода, слева – различные информационные панели. Для корректной работы компилятора необходимо сделать нижнее поле программным корневым. Для этого нужно нажать на вкладку с названием программы (на рис. это «ALUCellular.oap) правой кнопкой мыши и в появившемся всплывающем меню выбрать пункт «Root tab» (рис. 2). Если все сделано корректно, то поле программы окрасится в белый цвет.

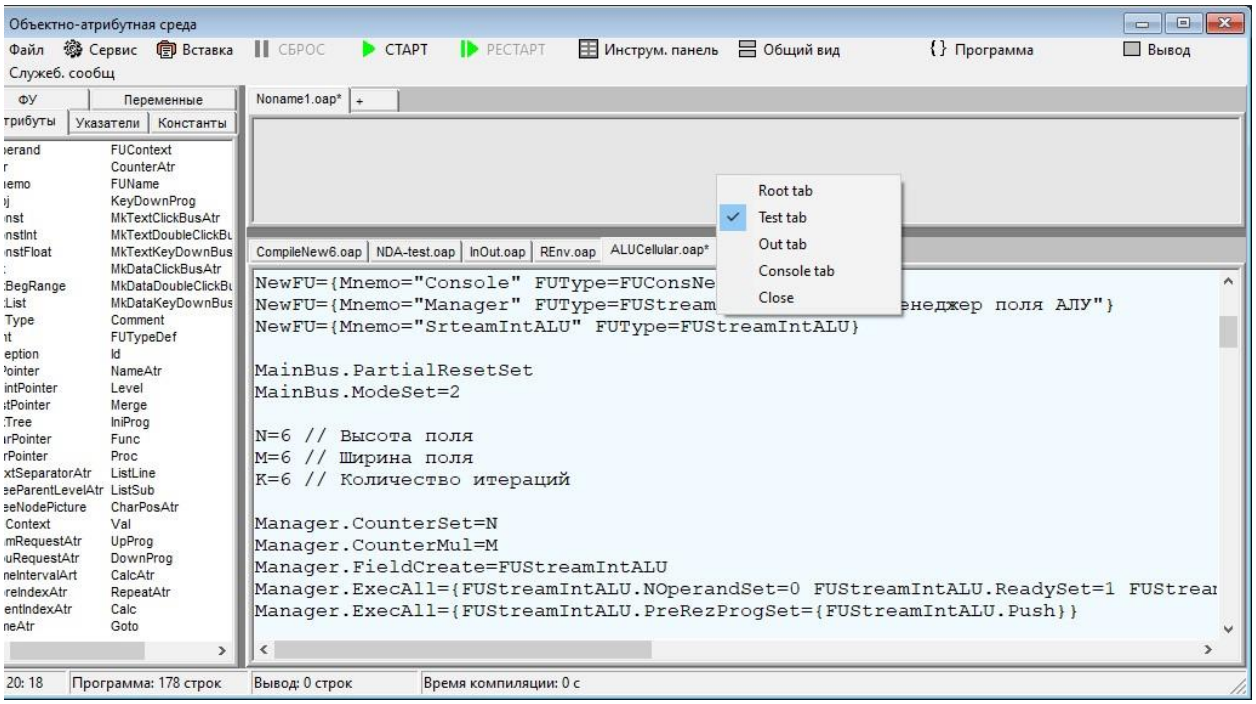

Рис. 2. Контекстное меню вкладки

Далее в программную вкладку производится ввод текста ОА-программы. Для компиляции необходимо сначала скомпилировать программу с помощью клавиши F9 (при этом фокус ввода Windows должен находиться на программном

поле) или с помощью кнопки «Рестарт», расположенной в верхней панели. Далее необходимо создать индексный файл, необходимый для запуска в среде программирования и моделирования. Это можно сделать с помощью сочетания клавиш «Ctrl+Alt+i» или с помощью пункта меню
«Файл→Создать индексный файл» (рис. 3). В появившемся контекстном меню необходимо задать имя индексного файла и его расположение. Обычно для индексного файла указывается расширение «.ind».

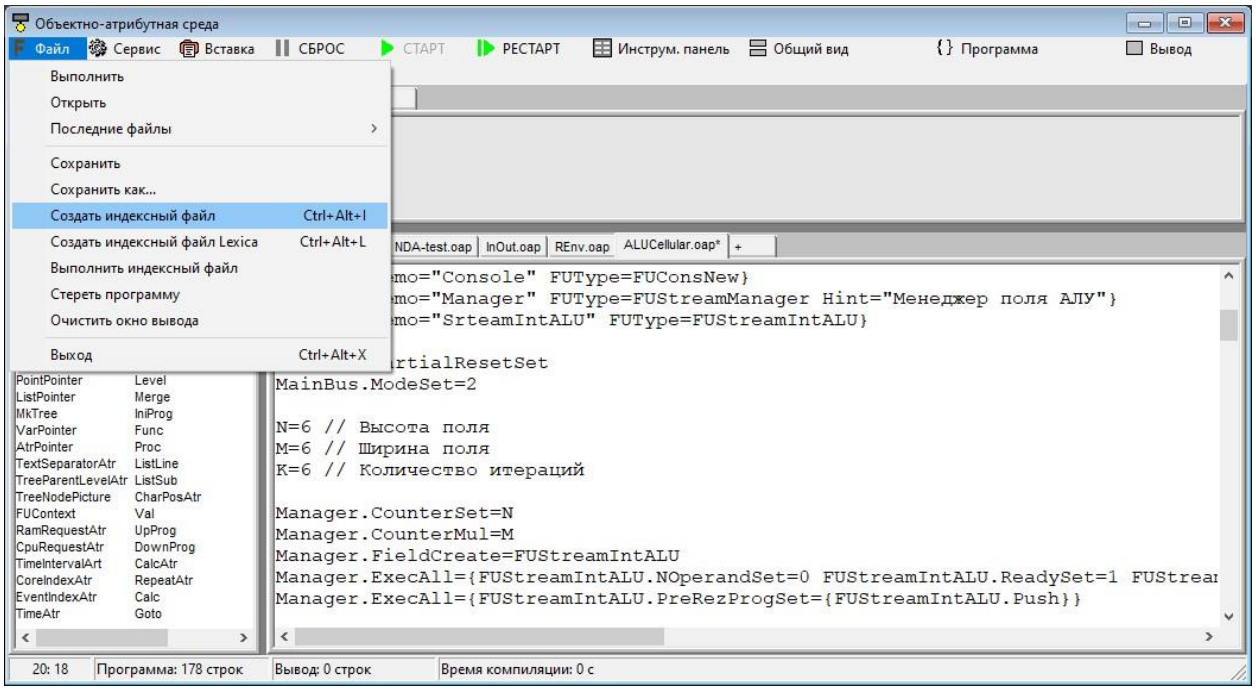

Рис. 3. Создание индексного файла

Среда ОА программирования и моделирования запускается с помощью файла millicom.exe, далее указывается имя индексного файла, который необходимо выполнить. Результаты выполнения программы из индексного файла выдаются либо на консоль, либо в файл. Консоль является удобным инструментом для визуализации модели клеточного автомата игра «Жизнь», поскольку клетки имеют два состояния.

4. Входные и выходные данные

Входные данные оформляются в виде программы на специализированном языке. В программе имеется возможность описания размера сетки и количества итераций. Программа должна начинаться со фрагмента кода, приведенного ниже, который производит настройку компилятора:

```
CapsManager.IndexVectCreate=20000
CapsManager.IpIcIdOutMk=MainBus.IpIcIdSet
CapsManager.IpIcIdOutMk=VariableManager.IpIcIdSet
CapsManager.IpIcIdOutMk=ListSyntez.IpIcIdSet MainBus.ModeSet=1
\\ ------------------------------------ \\
MainBus.FUTypeCorrectSet=-96
```

Далее в программе находится раздел описания ФУ, участвующих в процессе моделирования. Ниже приведено описание ФУ для моделирования вычислительного процесса клеточного автомата:

```
NewFU={Mnemo="Console" FUType=FUConsNew}
NewFU={Mnemo="Manager" FUType=FUStreamManager Hint="Менеджер поля АЛУ"}
NewFU={Mnemo="SrteamIntALU" FUType=FUStreamIntALU}
```

Здесь инициализируются 3 ФУ: программная консоль, менеджер поля АЛУ и потоковый целочисленный АЛУ. После ключевого слова «Mnemo» указывается имя ФУ, после ключевого слова «FUType» – тип ФУ, а после ключевого слова «Hint» – подсказка, которая будет показываться в среде при наведении мыши на имя ФУ.

5. Сообщения

В среде millicom во время компиляции ОА программы могут быть обнаружены синтаксические ошибки. В этом случае сообщения об ошибках выводятся на дополнительном всплывающем окне. В окне выводятся строка с ошибкой и две строчки перед ней. Знак «^» указывает на место ошибки.

Список сообщение об ошибках приведен ниже:

- Wrong expression – неправильное выражением
- '(' is not found – не найден символ "("
- ')' is not finded – не найден символ ")"
- Wrong Mkend description – неправильное описание милликоманды
- Wrong futype description – неправильный тип ФУ
- Wrong Mk description – неправильное описание милликоманды
- Error – общая ошибка
- Wrong Mnemonics – неправильное описание мнемоники
- Wrong attribute description – неправильное описание атрибута
- Mnemo not founded – не найдена мнемоника
- Constant is not founded – не найдена константа
- Variable or constant is not founded – переменная или константа не найдены
- Wrong load description – неправильное описание нагрузки милликоманды
- Wrong Mk description – неправильное описание милликоманд
- Wrong Ic description – неправильное описание информационной капсулы
- Wrong attribute description – неправильное описание атрибута

При успешном создании индексного файла в окне вывода должно появиться сообщение «Index file has created !!!» (рис. 4).

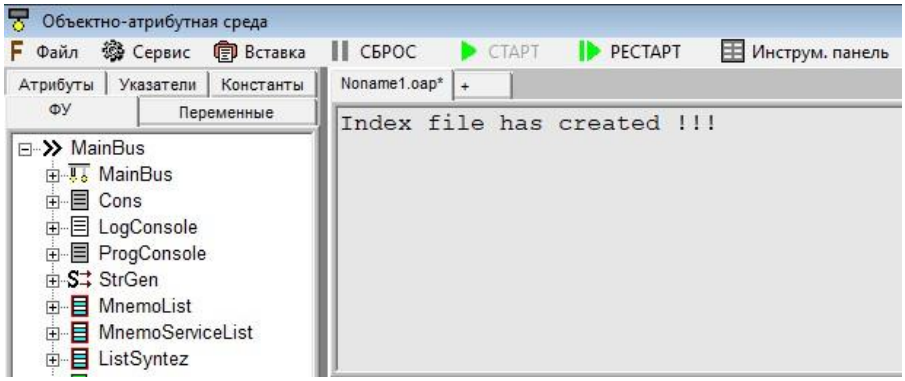

Рис. 4. Успешное создание индексного файла

Настройка системы моделирования осуществляется с помощью милликоманд (совокупность операнда и его атрибута). В ОА языке милликоманда обозначается с помощью знака "=". Слева от него указывает ФУ, которому адресуются данные, а также через знак "." атрибут данных, по которому ФУ идентифицирует их. В приложениях А, Б, В приведены мнемоники милликоманд, используемых для управления ФУ всех трех типов, которые входят в модель реализации модели Game of Life.

6. Приложения

Приложение А Перечень милликоманд ФУ Консоль

Милликоманда	Комментарий
Out	Вывод
OutLn	Вывод и перевод строки
LnOut	Перевод строки и вывод
LnOutLn	Перевод строки, вывод и перевод строки
Ln	Перевод строки
SepSet	Установить строку-разделитель
EndSet	Установить строку в конце вывода
ArrayBracketStartSet	Установить строку, обозначающую открывающуюся скобку при вывод вектора
ArrayBracketFinSet	Установить строку, обозначающую закрывающуюся скобку при вывод вектора
PrefixSet	Установить файл для вывода
StdOutFileSet	Установить файл для ввода
StdOutFileAppend	Установить файл для дополнения
StdInFileSet	Установить префикс перед строкой

Приложение Б Перечень милликоманд ФУ Менеджер потока

Милликоманда	Комментарий
FieldCreate	Создать поле АЛУ
FieldCreateTempl	Создать поле АЛУ на основе эталона
FieldClear	Очистить поле АЛУ
CreateGoup	Создать группу АЛУ
DevCreate	Создать АЛУ в группе
DevCreateLast	Создать АЛУ в последней группе
DevAdd	Добавить устройство в текущую группу
DevLastAdd	Добавить устройство в последнюю группу
IndGroupSet	Установить индекс группы АЛУ
Ind2Set	Установить второй индекс АЛУ в группе
Ind2GroupSet	Установить второй индекс группы АЛУ
IndSet	Установить индекс АЛУ в группе
IndSwap	Поменять индексы устройств местами (также меняются МК для ФУ)
GroupIndSwap	Поменять индексы групп местами
IndOut	Выдать первый индекс
IndOutMk	Выдать МК с первым индексом
Ind2Out	Выдать второй индекс
Ind2OutMk	Выдать МК со вторым индексом
IndAdd	Прибавить к индексу ФУ
IndGroupAdd	Прибавить к индексу группы
Ind2Add	Прибавить ко второму индексу ФУ
IndSub	Установить индекс АЛУ в группе
IndMul	Установить индекс АЛУ в группе
IndMod	Установить индекс АЛУ в группе
Ind2GroupAdd	Прибавить ко второму индексу группы
MkSet	Установить МК для первого индекса
Mk2Set	Установить МК для второго индекса
CounterSet	Установить сколько раз необходимо создавать устройства
CounterAdd	Прибавить к счетчику (по умолчанию 1)
CounterMul	Умножить счетчик (по умолчанию 2)
CounterSub	Вычесть из счетчика (по умолчанию 1)
ExecCounterSet	Установить счетчик итераций выполнения подпрограммы
ExecCounterAdd	Прибавить к счетчику итераций
ExecCounterSub	Вычесть из счетчика итераций

ExecCounterMul	Умножить счетчик итераций
ExecCounterDiv	Целочисленно разделить счетчик итераций
GroupRefCreate	Создать ссылки на ФУ для группы ФУ
MkSet	Установить МК для первого индекса
Mk2Set	Установить МК для второго индекса
ExecAll	Выполнить программу для всех ФУ поля
ExecGroup	Выполнить программу для всех ФУ группы

ExecDev	Выполнить программу для конкретного ФУ
Exec2Group	Выполнить программу для всех ФУ группы
Exec2Dev	Выполнить программу для конкретного ФУ
MkExec	Выполнить МК для ФУ поля
Mk2Exec	Выполнить МК для ФУ поля по второму индексу
MkBackExec	Выполнить МК для последнего ФУ последней группы поля
Mk2BackExec	Выполнить МК для ФУ поля по второму индексу
MkAllExec	Выполнить МК для всех ФУ поля
MkAllGroupExec	Выполнить МК для всех ФУ текущей группы
MkAllLastGroupExec	Выполнить МК для всех ФУ последней группы
RezVectOut	Выдать вектор результатов всех ФУ поля
RezVectOutMk	Выдать Мк с вектором результатов всех ФУ поля
ReadyVectOut	Выдать вектор готовности результатов всех ФУ поля
ReadyVectOutMk	Выдать Мк с вектором готовности результатов всех ФУ поля
RezGroupVectOut	Выдать вектор результатов всех ФУ группы
RezGroupVectPutMk	Выдать Мк с вектором результатов всех ФУ группы
ReadyGroupVectOut	Выдать вектор готовности результатов всех ФУ группы
ReadyGroupVectOutMk	Выдать Мк с вектором готовности результатов всех ФУ группы

DevCountOut	количество АЛУ в поле
DevCountOutMk	Выдать МК с количеством АЛУ в поле
DevGroupCountOut	Выдать количество АЛУ в текущей группе
DevGroupCountOutMk	Выдать количество АЛУ в текущей группе
GroupCountOut	Выдать количество групп
GroupCountOutMk	Выдать МК с количеством групп
DevOut	Выдать контекст текущего АЛУ
DevOutMk	Выдать контекст текущего АЛУ
Dev2Out	Выдать контекст текущего АЛУ по второму индексу
Dev2OutMk	Выдать контекст текущего АЛУ по второму индексу
LastDevOut	Выдать контекст последнего созданного АЛУ
LastDevOutMk	Выдать МК с контекстом последнего созданного АЛУ

LastGroupIndOut	Выдать индекс последней созданной группы АЛУ (-1, если поле пустое)
LastGroupIndOutMk	Выдать МК с индексом последней созданной группы АЛУ (-1, если поле пустое)
LastDevIndOut	Выдать индекс последнего созданного АЛУ (-1, если поле пустое)
LastDevIndOutMk	Выдать МК с индексом последнего созданного АЛУ (-1, если поле пустое)

Приложение В Перечень милликоманд ФУ Потоковое целочисленное АЛУ

Милликоманда	Комментарий
Set	Установить результат
RoutProg	Программа при несовпадении адреса ФУ с его собственным адресом
Out	Выдать результат
OutMk	Выдать МК с результатом
AccumModeSet	Установить аккумуляторный режим вычислений (по умолчанию true)
AngleModeSet	Установить режим измерения угла (0 - радианы, 1 -градусы)
ReadySet	Установить флаг готовности результата (по умолчанию true)
ReadyOut	Выдать флаг готовности результата
ReadyOutMk	Выдать МК с флагом готовности результата
RezOutBlockSet	Установить блокировку выдачи результата (при нулевой нагрузке true)
RezSend	Выслать результат вычислений (+ выполняются программы по флагам)
BufSet	Записать результат в буфер (при нулевой нагрузке записывается из Rez)
BufSend	Разослать результат из буфера
NOperandSet	Установить количество операндов (по умолчанию 2)
NOperandAdd	Увеличить количество операндов (по умолчанию на 1)
ReceiverReset	Сброс установок получателей результата
ReceiverAdd	Установить ссылку на контекст получателя результата
ReceiverMkAdd	Установить МК для получателя результата
ReceiverCountOut	Выдать количество получателей результата
ReceiverCountOutMk	Выдать МК с количеством получателей результата
RezProgSet	Установить программу при получении результата
PreRezProgSet	Установить Программу, запускаемую перед получением результата

Swap	Обменять местами регистр результата и верхний элемент стека
Push	Положить в стек (при нулевой нагрузке в стек помещается Rez)
Pop	Вынуть из стека (при при нулевой нагрузке величина помещается в Rez)
PopMk	Вынуть из стека и выдать МК (при при нулевой нагрузке величина помещается в Rez)

StackOut	Выдать из стека (при при нулевой нагрузке величина помещается в Rez)
StackOutMk	Выдать из стека с МК (при при нулевой нагрузке величина помещается в Rez)
StackCounterOut	Выдать количество элементов в стеке
StackCounterOutMk	Выдать МК с количеством элементов в стеке
ErrProgSet	Установить программу при ошибке
RezStackIsEmpyProgSet	Установить программу при чтении из пустого стека результатов
EqProgSet	Установить программу при ==
NEqProgSet	Установить программу при !=
LessProgSet	Установить программу при <
BiggerProgSet	Установить программу при >
LessEqProgSet	Установить программу при <=
BeggerEqProgSet	Установить программу при >=
ReadyExec	Запуск программы по флагу готовности результата
ReadyNotExec	Запуск программы при сброшенном флаге готовности результата
StackEmptyExec	Выполнить, если стек пустой
StackNotEmptyExec	Выполнить, если стек не пустой
ZeroExec	Выполнить программу при ==
LessExec	Выполнить программу при <
BiggerExec	Выполнить программу при >
LessEqExec	Выполнить программу при <=
BbiggerZExec	Выполнить программу при >=
NZeroExec	Выполнить программу при !=
Equal	Проверка на равенство
NotEqual	Проверка на неравенство