ISS TECHNOLOGY
ИНТЕЛЛЕКТУАЛЬНЫЕ СИСТЕМЫ БЕЗОПАСНОСТИ

Программирование в VideoInspecotr Global
Язык скриптов
Настройки через реестр

VideoInspector global

Руководство программиста

Rev. 1.2

119899 Москва

Ленинские горы, МГУ, Научный парк

владение 1, строение 77, офис 102 "Центр Нейросетевых Технологий - Интеллектуальные Системы Безопасности", ООО

Телефоны: 930-8860, 930-8861, 930-8106

E-Mail: info@iss.ru

Оглавление

Программы

В системе VideoInspector Global встроен гибкий механизм настройки автоматической или автоматизированной реакции на события, поступающие в систему от реальных конечных устройств охранных систем (камер, датчиков, считывателей и т.д.). Это механизм макрокоманд и скриптов. Кроме того можно обрабатывать события поступающие от виртуальных объектов самой системы VideoInspector Global (смена экрана, выполнение макрокоманды, выполнения реакции и др.).
Макрокоманды уже сами по себе дают вам широкие дополнительные возможности по управлению системой. Однако, этого может оказаться мало. В случае возникновения необходимости воссоздать сложную реакцию системы вам на помощь придёт программа.
Применение встроенного в систему языка скриптов позволит вам вводить практически любые отношения между объектами и событиями системы.

☛ *Внимание! Язык скриптов даёт широкие возможности по управлению объектами и реакциями системы. Вследствие этого возрастает вероятность нестабильной работы (вплоть до зависания системы) при ошибках в программировании. Будьте внимательны!!!*

☛ *Язык скриптов предназначен для управления уже созданными объектами системы. Создание новых объектов посредством данного инструмента невозможно.*

Создание программы

Создайте объект Программа. Этот объект является дочерним по отношению к VideoInspector Global. Панель настройки созданного объекта Программа и является тем окном, где нужно будет писать программу на языке скриптов VideoInspector Global.

Структура языка скриптов для написания программ

Практически ко всеми объектами системы "VideoInspector Global" привязываются понятия: Идентификатор типа, Номер(ID), События, Действия (Реакции), Состояния.

- ✔ **Идентификатор типа.** Название объекта в системе "VideoInspector Global". Всегда пишется заглавными латинскими буквами (Например: **CAM**, **DIALOG**, **DISPLAY**). Используется в командах скриптов **OnEvent**, **DoReact** и **CheckState** в первом параметре (Например **DoReact** (**"CAM"**, "...","...")).
- ✔ **Номер (ID).** Номер объекта в системе "VideoInspector Global" данного типа . Берется в кавычки (Например: "`1`"). Используется в команде скриптов **OnEvent** и **DoReact** во втором параметре (Например **OnEvent** (**"CAM"**, "`1`","**ARM**")).
- ✔ **События.** Название событий объекта в системе "VideoInspector Global". Всегда пишется заглавными латинскими буквами (Например: **ARM**, **DISARM**, **REC**). Используется в команде скриптов **OnEvent** в третьем параметре (Например **OnEvent** (**"CAM"**, "...","**ARM**")).
- ✔ **Действия (Реакции).** Название действий над объектом в системе "VideoInspector Global". Всегда пишется заглавными латинскими буквами (Например: **ARM**, **DISARM**, **REC**). Используется в команде скриптов **DoReact** в третьем параметре (Например **DoReact** (**"CAM"**, "...","**MUX1**")).

- ✔ **Состояния.** Название состояния объекта в системе "VideoInspector Global". Всегда пишется заглавными латинскими буквами (Например: **ARMED**, **DETACHED**). Используется в команде скриптов **CheckState** в третьем параметре (Например **CheckState** ("**CAM**", "...","**DETACHED**")).

Структура программы и процедуры - обработчики событий

Программа состоит из набора процедур - обработчиков событий. Каждая процедура работает независимо от остальных. Таким образом вы можете включать в один объект "Программа" несколько обработчиков событий как логически связанных между собой, так и логически независимых.

- ✔ **OnInit()** - выполняется один раз в момент запуска системы (используется для инициализации переменных).
- ✔ **OnTime(W,D,X,Y,H,C,S)**- выполняется по расписанию.
  где:
  W - день недели (0 - понедельник, 6 - воскресенье);
  D - дата в формате "число-месяц-год", 16 августа 2001 года это "16-08-01"
  X,**Y** - зарезервировано
  H - час
  C - минуты
  S - секунды
- ✔ **OnEvent(идентификатор типа источника, номер, идентификатор события)** - выполняется по событию от объектов системы.

Каждая процедура, имеющая параметры, может встречаться в коде много раз с различными параметрами. При возникновении события, система выполнит те из них, параметры которого совпадут с параметрами возникшего события.
Далее приведена общая структура программы.

```
OnInit()       // Инициализация переменных
{
  ...
}
OnTime (...)  // Операторы, выполняемые по расписанию
{
  ...
}
...
OnEvent(...)  // Операторы, выполняемые по событию
{
  ...
}
...
```

Пример 1

Выполнится при замыкании луча 1

```
OnEvent("RAY","1","ON")
{
...
}
```

Пример 2

Выполнится при замыкании любого луча.
Переменной N присваивается значение номера луча.

```
OnEvent("RAY",N,"ON")
{
 if (strequal(N, "1"))
  {
   ...
  }
...
}
```

Пример 3

Выполнится при появлению движения в поле зрения камеры №3.

```
OnEvent("CAM","3","MD_START")
{
 ...
}
```

Пример 4

Выполнится каждую секунду

```
OnTime(N,N,N,N,N,N,N)
{
 ...
}
```

Пример 5

Выполнится16 августа 2001 года в 11 часов 11 минут 30 секунд

```
OnTime(W,"16-08-01",X,Y,"11","11","30")
{
 ...
}
```

Пример 6

Выполнится каждый день в 11 часов 11 минут 30 секунд

```
OnTime(W,D,X,Y,"11","11","30")
{
 ...
}
```

Пример 7

Выполнится 16 августа 2001 года каждую секунду

```
OnTime(W,"16-08-01",X,Y,H,C,S)
{
 ...
}
```

Пример 8

Выполнится 16 августа 2001 года с 11 часов 11 минут по 11 часов 12 минут каждую секунду

```
OnTime(W,"16-08-01",X,Y,"11","11",S)
{
 ...
}
```

Пример 9

Выполнится каждый понедельник в 21 часов 00 минут 00 секунд

```
OnTime ("0",D,X,Y,"21","0","0")
{
 ...
}
```

Переменные и константы

Параметр любой процедуры может быть константой или переменной. Константы заключаются в кавычки, например, - "`1`", "**RAY**" и т.п. Имена переменных должны содержать латинские буквы, - `CHANNEL, N, i` и т.п.

☛ *Значения всех переменных строковые!*

Если требуется работать с целочисленными значениями используйте функцию **atof**() или **str**() .
В случае, если аргументом процедуры является переменная, - процедура будет выполнена для всех событий для которых его можно определить и соответствующее значение будет присвоено переменной.
В языке скриптов присутствуют системные переменные:

- ✔ **Date** - переменная равная системной дате.
- ✔ **Time** - переменная равная системному времени.

Глобальные переменные

В скриптах имеет место механизм глобальных переменных. Суть механизма в том, что глобальные переменные имеют одинаковое значение на всех компьютерах системы. При изменении значения из скрипта на одном из компьютеров - соответствующая переменная автоматически принимает новое значение и на всех остальных компьютерах системы.
Для работы с глобальными переменными используются следующие функции:

- ✔ **GlobalVarSet(name,value)** - устанавливает для глобальной переменной, имеющей имя **name**, значение **value**.
- ✔ **GlobalVarGet(name)** - возвращает значение глобальной переменной, имеющей имя **name**.

Механизм CellBox

CellBox - это объект языка скриптов представляющий собой ящик с N ячейками. В каждую ячейку можно поместить любое значение. Когда все ячейки заполнены - ящик “переворачивается”. Если в течении t секунд после заполнения первой ячейки все ячейки еще не заполнены - ящик переворачивается в другую сторону. При перевороте все ячейки ящика освобождаются.
События, генерящиеся системой при “перевороте” ящика в ту или иную сторону:

- ✔ **CELLBOX|x|TIMEOUT** - ящик перевернулся по таймауту.
- ✔ **CELLBOX|x|FILLEDUP** - ящик заполнен. Параметры: **count** - число ячеек, число **cell0** - **cellN** (где N=0...count) - значение из ячейки с соответствующим индексом.

Для работы с CellBox в языке скриптов VideoInspector Global предназначены следующие функции:

- ✔ **CellBoxCreate(name,cells,timeout)** - создание ящика **name** - имя (строка), **cells** - количество ячеек (число 0-65536), **timeout** - таймаут в миллисекундах (число). В случае, если ящик с таки именем уже существует - его содержимое сбрасывается, устанавливается новое число ячеек и таймаут.
- ✔ **CellBoxPut(name,ncell,value)** - Положить значение в ячейку. **name** - имя (строка), **ncell** - индекс ячейки (нумерация с 0), **value** - помещаемое значение (строка). В случае, если ящика с таким именем не существует - выполнение программы прерывается с ошибкой "CellBox doesn't exists".

- ✔ **CellBoxDestroy(name)** - уничтожение CellBox с именем **name**. Если CellBox с таким именем необходимо использовать в дальнейшем - его нужно создать заново.
- ✔ **CellBoxReset(name)** - сброс CellBox с именем **name** в исходное состояние. Все ячейки очищаются. Срабатывание по **timeout** отменяется.

Пример использования - принять решение о пропуске персоны на основании совместного срабатывания считывателя Аполло и распознавания номера авто.

Процедуры действий

Для инициации из программы действий объектов системы, используется специальная процедура:

- ✔ **DoReact(идентификатор типа объекта, номер объекта, идентификатор действия [, параметры])** - выполнить действие объекта заданного типа и номера с параметрами. Параметры могут отсутствовать.
- ✔ **DoReactGlobal** - аналогична **DoReact**, но ещё передаёт команду на другие ядра системы.
- ✔ **NotifyEvent("CORE", номер ядра, идентификатор действия,[параметры], "objtype<идентефикатор объекта>, [параметры]")** - послать сообщения локальным ядрам произвести указанное действие над объектом.
- ✔ **DoCommand(команда)**, **DoCommandAndWait(команда)** - обе процедуры выполняют командную строку (вторая процедура при этом ещё и ждёт до окончания выполнения команды).

Имеются также процедуры выдержки временных интервалов и проверки состояния объектов:

- ✔ **Wait(кол-во секунд)** - ждать заданное количество секунд. Используя данную процедуру следует брать в квадратные скобки все процедуры и операторы, которые как-либо связаны с данной задержкой - это значит: выполнять в отдельном потоке, не задерживая другие потоки обработки событий системы.
- ✔ **Sleep(кол-во миллисекунд)** - аналогична **Wait**, только интервал в миллисекундах.

☛ *Внимание! После каждой процедуры действий, а также других операторов в теле процедуры-обработчика событий ставится точка с запятой (;)*

Пример 10

Включить камеру на запись.
DoReact ("**CAM**","1","**REC**");

Пример 11

По тревоге от луча 1 реле 1 будет замыкаться на 5 секунд с интервалом 3 секунды в течении тревоги от луча но не более 80 секунд. Реле 1 может управлять, к примеру, ревуном.

```
OnEvent("RAY","1","ALARM") {
  [For( i=0; i<10 && CheckState("RAY","1","ALARM"); i=i+1) {
   [DoReact("RELE","1","ON");
   Wait(5);
   DoReact("RELE","1","OFF");
   Wait(3);]
  }]
}
```

В качестве параметров могут быть также и выражения:
Например: **DoReact**("**MACRO**",**str**(i)+3,"**RUN**");

Процедура состояния

В языке скриптов присутствует одна процедура состояния, позволяющая определить состояния объекта:

✔ **CheckState(идентификатор типа объекта, номер объекта, идентификатор состояния)** - результат будет равен 1 если состояние объекта соответствует указанному, иначе 0.

Операторы и выражения

Имеются условный оператор и оператор цикла, а также обычный набор операторов, которые могут использоваться в выражениях.

Таблица 1: Операторы языка скриптов

Условный оператор	if (выражение) {
	операторы, выполняемые если результат выражения не 0
	}
	else
	{
	операторы, выполняемые если результат выражения равен 0
	}
	Часть оператора else {} может отсутствовать

```
OnEvent("OLXA_LINE","5","ACCU_STOP")
{
if(!n2v) {DoReact("CAM","2","REC_STOP");}
else {DoReact("CAM","2","REC");}
}
```

Оператор цикла	**Перед началом цикла выполняется выражение 1. Цикл выполняется до тех пор, пока значение выражения 2 не равно 0. В конце каждого цикла выполняется выражение 3.** [for(выражение 1; выражение 2; выражение 3) { ... }]

Таблица 1: Операторы языка скриптов

```
OnEvent("MACRO","1","RUN")
{
m=7;
[for(i=2;i<m;i=i+1)
{
DoReact("MACRO",i,"RUN");
Wait(2);
}]
}
```

По выполнению макрокоманды 1 выполнятся макрокоманды со 2 по (m-1) с задержкой 2 секунды.

Оператор присвоения	=

```
m=3;
```

Оператор логического отрицания	!

```
if !(n2v){...}
```

Оператор логическое И	&&

```
for(i=0;i<10&&!bReset;i=i+1)
```

Операторы сравнения	> **- больше** < **- меньше**

```
if (atof(a)>atof(b))
```

Арифметические операции	+ **- сложение** - **- вычитание** * **- умножение** / **- деление** % **- остаток от целочисленного деления** ( )**- группа арифметических операций**

```
DoReact("MACRO",str(i)+3,"RUN");
```

Тернарная операция	**Результатом этой операции будет выражение2 в случае если выражение1 не равно 0, иначе - выражение3.** выражение1 ? выражение2 : выражение3

```
s = "10";
k = (s>10)? "12":"199";

DoReact("DIALOG","msgbox","RUN","msg_text<" + k + ">");
```

Таблица 1: Операторы языка скриптов

Выделение отдельного потока	[] Код, записанный между квадратными скобками, будет выполнен в отдельном потоке.

```
OnEvent("MACRO","1","RUN")
{
m=7;
[for(i=2;i<m;i=i+1)
{
DoReact("MACRO",i,"RUN");
Wait(2);
}]
}
```

Функции языка скриптов

Ниже приведены функции языка скриптов которые не были рассмотрены ранее в соответствующих специализированных разделах. Функции рассортированы по типу (для работы с объектами системы, математические, для работы со строками и т.д.).

Объектные функции

Ниже приведён перечень функций языка скриптов для работы с объектами системы VideoInspector Global.

Таблица 2: Объектные функции

Функция	Описание
GetObjectNum(ObjType)	Возвращает количество объектов типа ObjType в системе
GetObjectName(ObjType, ObjID)	Возвращает имя запрашиваемого объекта (как строку)
GetObjectParam(ObjType, ObjID, Param)	Возвращает значение запрашиваемого параметра объекта (как строку)
GetEventDescription(ObjType, Event)	Возвращает описание запрашиваемого события для указанного объекта (как строку)
GetObjectId(ObjType, Index)	Возвращает идентификатор объекта типа ObjType, имеющего номер Index по порядку в списке объектов (0<=Index<GetObjectNum(ObjType))

Математические функции

В приведённой ниже таблице собраны математические функции, которые вы можете использовать при написании скриптов.

Таблица 3: Математические функции

Функция	Аргументы	Возвращаемое значение	Описание	Дополнение
sin(x)	x - угол в радианах	значение от -1 до +1	вычисляет синус угла	
cos(x)	x - угол в радианах	значение от -1 до +1	вычисляет косинус угла	
tan(x)	x - угол в радианах	число от -∞ до +∞	вычисляет тангенс угла	
asin(x)	x - число от -1 до +1	угол в радианах	вычисляет арксинус	
acos(x)	x - число от -1 до +1	угол в радианах	вычисляет арккосинус	
atan(x)	x - число от -∞ до +∞	угол в радианах	вычисляет арктангенс	
sinh(x)	x - число от -∞ до +∞	число от -∞ до +∞	вычисляет гиперболический синус	
cosh(x)	x - число от -∞ до +∞	число от -∞ до +∞	вычисляет гиперболический косинус	
tanh(x)	x - число от -∞ до +∞	число от -∞ до +∞	вычисляет гиперболический тангенс	
exp(x)	x - число от -∞ до +∞	число от 0 до +∞	вычисляет экспоненту	
log(x)	x - число от -∞ до +∞	число от -∞ до +∞	вычисляет натуральный логарифм	
log10(x)	x - число от -∞ до +∞	число от -∞ до +∞	вычисляет десятичный логарифм	
sqrt(x)	x - число от 0 до +∞	число от 0 до +∞	вычисляет квадратный корень	
floor(x)	x - число от -∞ до +∞	целое число от -∞ до +∞	округляет до ближайшего меньшего целого	
ceil(x)	x - число от -∞ до +∞	целое число от -∞ до +∞	округляет до ближайшего большего целого	
abc(x)	x - число от -∞ до +∞	число от 0 до +∞	модуль числа	
deg(x)	x - угол в радианах	угол в градусах	перевести радианы в градусы	
rad(x)	x - угол в градусах	угол в радианах	перевести градусы в радианы	
int(x)	x - число от -∞ до +∞	целое число	производит округление до целого	
max(x,y)	x - любое число y - любое число	большее из чисел	определяет большее из двух чисел	

Таблица 3: Математические функции

Функция	Аргументы	Возвращаемое значение	Описание	Дополнение
min(x,y)	x - любое число y - любое число	меньшее из чисел	определяет меньшее из двух чисел	

Строковые функции

Ниже дан перечень строковых функций, которые вы можете использовать при написании скриптов.

Таблица 4: Строковые функции

Функция	Аргументы	Возвращаемое значение	Описание	Дополнение
str0(x)	x - любое число	строка	переводит числовое значение в строковое	если x=0, то возвращает пустую строку
str(x)	x - любое число	строка	переводит числовое значение в строковое	работает как и str(), но в случае x=0 возвращает “0“ как строку
atof(x)	x - строка	double	переводит строку в число	
val(x)	x - строка	double	переводит строку в число	
date_frm(d1, d2)	d1 - дата в формате 24-12-2004 d2 - формат в виде, в котором он используется в функции GetDateFormat (из MSDN)	строка	форматирует дату по указанному формату	
gtd2date(x)	x - строка содержащая символ “/“	строка	преобразует номер ГТД в дату	
scalar2date(x)	x - целое число	строка в виде 04-02-01	преобразует количество дней с начала нашей эры в дату	
scalar(x)	x - строка в виде 02-03-04	целое число	преобразует дату в число дней с начала нашей эры	если ввести пустую строку, то используется текущее системное время

Таблица 4: Строковые функции

Функция	Аргументы	Возвращаемое значение	Описание	Дополнение
long2time(x)	x - число секунд	строка в формате ЧЧ-ММ-СС	переводит число секунд в пределах суток во временной формат	если число секунд превышает количество секунд в сутках, то возвращается пустая строка
time2long(x)	x - строка вида 13:32:11 (время)	число (секунды)	переводит время в стандартном формате в число секунд с начала суток	
ExecuteCmd(x)	x - shell команда, например: format C: /y	- Несмотря на то, что нет возвращаемого значения, функцию нужно присвоить любой строковой перемнной	выполняет shell команду переданную через параметр	
timcmp(s1, s2)	s1 - первая строка s2 - вторая строка	1 если s1>s2 0 если s1=s2 -1 если s1<s2	сравнивает две строки содержащих дату и время (или только дату и только время)	формат: 25-19-1977 12:30:12
strcmp(s1, s2)	s1 - строка s2 - строка	>0 если s1>s2 =0 если s1=s2 <0 если s1<s2	сравнивает строки по их лексикографическим значениям	
strreplace(s1, s2, s3)	s1, s2, s3 - строки	преобразованная строка s1	заменяет все вхождения строки s2 в строке s1 на строку s3	
strequal(s1, s2)	s1, s2 - строки	1 - если строки равны 0 - если строки не равны	сравнивает две строки	
strsub(s1, s2)	s1, s2 - строки	число символов	подсчитывает число символов с начала строки s1 до первого вхождения в неё строки s2 и прибавляет к этому числу 1	
strempty(x)	x - строка	1 - строка пустая или состоит из пробелов 0 - строка не пустая	определяет пустая ли переданная строка	
straleft(s1, x)	s1 - строка x - число	строка	дополняет строку s1 пробелами до тех пор, пока её длинна не станет равной числу x	
straright(s1, x)	s1 - строка x - число	строка	действует так же как и straleft, но пробелы добавляет слева	

Таблица 4: Строковые функции

Функция	Аргументы	Возвращаемое значение	Описание	Дополнение
strlen(s1)	s1 - строка	число	вычисляет длину строки	
strmid(s1, x, y)	s1 - строка x - число y - число	преобразованная строка s1	возвращает из сроки s1 символы начиная с позиции x и до позиции y (включительно)	нумерация символов идёт с левого края строки и начинается с 0
strleft(s1, x)	s1 - строка x - число	преобразованная строка s1	возвращает строку состоящую из первых x символов слева в строке s1	
strright(s1, x)	s1 - строка x - число	преобразованная строка s1	возвращает строку состоящую из x первых символов справа в строке s1	
strnleft(s1, x)	s1 - строка x - число	преобразованная строка s1	обратна функции strleft - возвращает все символы за исключением x символов слева	
strnright(s1, x)	s1 - строка x - число	преобразованная строка s1	обратна функции strright - возвращает все символы за исключением x символов справа	
get_substr(s1, s2, s3)	s1, s2, s3 - строки	преобразованная строка s1	возвращает из строки s1 символы начиная с вхождения s2 и до первого вхождения s3	если после s2 нет вхождений s3, то возвращается строка до конца
extract_substr(s1, l, n)	s1 - строка l - один символ n - число	строка	возвращает из строки s1 символы в промежутке между n и n+1 вхождением символа l	
get_words(s1, n1, n2)	s1 - строка n1 - число n2 - число	строка	возвращает из строки s1 n2 слов, начиная со слова n1	отсчёт слева с 1, разделитель для слов - пробел
strltrim(s1)	s1 - строка	строка	отрезает пробелы слева	
strrtrim(s1)	s1 - строка	строка	отрезает пробелы справа	
stratrim(s1)	s1 - строка	строка	отрезает пробелы слева и справа	
strupr(s1)	s1 - строка	строка	делает uppercase	
strlwr(s1)	s1 - строка	строка	делает lowercase	
set_at(s, n, c)	s - строка n - число c - один символ	строка	заменяет в строке s символ с индексом n символом c	отсчёт слева с нуля

Таблица 4: Строковые функции

Функция	Аргументы	Возвращаемое значение	Описание	Дополнение
ger_at(s, n)	s - строка n - число	один символ	возвращает из строки s символ с индексом n	отсчёт слева с нуля
decl_num(n, s)	n - число s - строка типа блаблабла#N	модифицированная строка	возвращает исходную строку до символа # плюс число n дополненное нулями до количества знаков N (если n=23, а N=4, то добавится 0023)	если строка не содержит # и не может быть преобразована к числу, то будет возвращено число n; если же строка может быть преобразована к числу, то результатом будет число n дополненное нулями до числа в строке s ((123, “5”) ->00123)
iif(s1, s2, s3)	s1, s2, s3 - строки	строка	если s1 пустая или состоит из пробелов, возвращается s3, в противном случае s2	

Дополнительные функции

Ниже дан перечень функций которые не вошли не в один из предыдущих разделов.

Таблица 5: Дополнительные функции

Функция	Аргументы	Возвращаемое значение	Описание	Дополнение
number_frm(n1, n2)	n1, n2 - числа	число	возвращается n1 дополненное n2 знаками после запятой (например (2.34, 4)->2.3400)	если n1 целое, то нули будут убраны; если n2=0, то вернётся n1 без изменений (но знаков после запятой не больше 6)
int_frm(n1, n2)	n1, n2 - числа	число	если n1 >= n2, то возвращается n1; если n2>n1, то возвращается n1 дополненное нулями слева (например: (245, 5)->00245)	
currency(s)	s - строка	строка	заменяет в строке s первое вхождение символа “.“ на “-“	

Таблица 5: Дополнительные функции

Функция	Аргументы	Возвращаемое значение	Описание	Дополнение
convert(n)	n - число	строка	результат - введённое число прописью по русски (дробная часть отбрасывается)	
convert_cur(n)	n - число	строка	после целой части числа добавляется слово “рублей“, после дробной, округлённой до 2-х цифр - “копеек“	
evaluate(s)	s - строка математического выражения (например “2+3“)	число	вычисляет математические выражения	
end(n)	n - число	число	если n<0 - возвращает 100; иначе возвращает n	
IsFileExist(filename)	filename - путь и имя файла.	0 - нет файла 1 - файл найден	проверяет наличие файла на диске	
RGB(red, green, blue)	red - число в пределах 0-255 представляет красный цвет green - 0-255 зелёный blue - 0-255 синий	целое значение	возвращает целое число представляющее RGB схему	
MessageBox(caption, text, type)	caption - текст заголовка text - текст в окне сообщений type - число - тип окна сообщений	- Несмотря на то, что нет возвращаемого значения, функцию нужно присвоить любой строковой перемнной	выводит на экран окно сообщений	число, определяющее, какие элементы будут на панели окна: 0 - кнопка OK 1 - OK и Cancel 2 - Abort, Retry, Ignore 3 - Yes, No, Cancel 4 - Yes, No 5 - Retry, Cancel 16 - иконка “Рука“ 32 - “Вопрос“ 48 - “Воскл. знак“ 64 - “Звёздочка“ 16384 - кнопка “Help“ Это три группы. Складывая значения флагов из разных групп, можно получить общий флаг для получения нескольких элементов на панели окна (по одному из каждой группы).

Типы объектов системы

В состав системы VideoInspector Global входит более сотни объектов, к каждому из которых можно обратиться в ходе написания скриптов. Обращение к объектам идёт по их типам и идентификаторам. При этом для каждого типа объектов существует набор событий (объект может генерить в ходе работы системы), реакций (действий, которые можно совершить с данным типом объекта), и состояний (объект может пребывать стационарно).
В “Приложении1” (см. “Приложение 1 (объекты)” на стр. 27) дано описание типов объектов системы. Для каждого типа дан список генерированных им событий, производимых действий, состояний, а так же наиболее часто употребимых параметров.

☛ *Для установки настроек объектов из программ можно использовать команду **SETUP**. Однако, в отличие от установок, сделанных через панель настроек, установки произведённые через эту команду не записываются в базу данных и будут сброшены после перезагрузки системы.*

Как получить дополнительную информацию об объектах

В предыдущем пункте была дана ссылка на список объекты системы с набором наиболее часто употребимых параметров. Если вам понадобится получить более подробную информацию об объекте (параметры событий и действий) вы можете перевести VideoInspector Global в Debug-mode и получить интересующие вас данные из окна-отладчика.
Для перевода VideoInspector Global в Debug-mode:

- ✔ Выгрузите VideoInspector Global.
- ✔ Загрузите системную программу редактирования реестра (regedit).
- ✔ По адресу `HKLM\\SOFTWARE\\ISS\\Niss400\\debug` проставить флаг 2.
- ✔ Закройте утилиту.
- ✔ Запустите VideoInspector Global.

Теперь VideoInspector Global находится в Debug-mode и в режиме администрирования в нижней части экрана появится дополнительное окно, в котором будут отображаться все внутренние команды системы.
Для получения информации о параметрах какого-либо действия (события), вам нужно сначала добиться того, чтобы это действие было воспроизведено системой (например для команды SETUP объекта, вызовите его панель настроек). Далее в Debug окне найдите строку этого действия (или события) и выделите её левой клавишей мыши. Далее кликните на строке правой клавишей мыши. Как результат, в верхней части экрана появится плавающая панель в которой будет отображено выбранное действие (событие) с полным набором параметров и сделанными установками для данного конкретного случая.

Примеры Скриптов

Ниже приведены примеры скриптов, решающие различные задачи.
Данный список постоянно пополняется.

☛ *Внимание! Применяя скрипт, будьте уверены, что перечисленные в нём объекты уже созданы в вашей системе.*

Скрипт 1

Камера на аналоговый монитор

```
OnEvent ("MONITOR","4","ACTIVATE_CAM")
{
DoReact ("CAM", cam,"MUX1");
}
```

Выводить на аналоговый монитор камеру по клику в ее поле на виртуальном мониторе
Номер монитора может быть другим. Активировать камеру можно также с карты, в данном случае событие:
OnEvent("MAP","1", "ACTIVATE_OBJECT").

Скрипт 2

Включить реле по движению в зоне

```
OnInit()
{
AlarmCount=0;
}
OnEvent("CAM",N,"MD_START")
{[
AlarmCount=AlarmCount+1;
if( AlarmCount<2 ) {
[for( i=0; i<5; i=i+1 )
{
DoReact("RELE","1","ON");
Wait(1);
DoReact("RELE","1","OFF");
Wait(1);
if( AlarmCount>0 ) { i=0; }
}]
}
```

При возникновении движения в поле зрения любой камеры в течении 10 секунд (после последнего движения) вкючать реле номер 1 с периодичностью 1 секунда. На реле может быть заведена к примеру лампа.
Номер камеры может быть фиксированным, номер реле также может быть общим.

Скрипт 3

Камера на весь экран по тревоге

```
OnEvent ("CAM","1","MD_START")
{
DoReact ("DISPLAY","","DEACTIVATE");
DoReact ("DISPLAY","3","ACTIVATE");
DoReact ("MONITOR","4","KEY_PRESSED","key<CAM_ACTIVATE.1>");
DoReact ("MONITOR","4","KEY_PRESSED","key<SCREEN.1>");
}
OnEvent ("CAM","2","MD_START")
{
DoReact ("DISPLAY","","DEACTIVATE");
DoReact ("DISPLAY","3","ACTIVATE");
DoReact ("MONITOR","4","KEY_PRESSED","key<CAM_ACTIVATE.2>");
DoReact ("MONITOR","4","KEY_PRESSED","key<SCREEN.1>");
}
```

По тревоге на данной камере выводить ее на весь экран.
Пример для двух камер с номерами 1 и 2.

Скрипт 4

Камеры на аналоговый монитор

```
OnEvent("MACRO","1","RUN")
{[
AutoScanMode=1;
[for(i=1;AutoScanMode;i=i+1)
{
if(i>6) {
i=1;
}]
Wait(3);
DoReact("CAM",i,"MUX1");
}
]}
//
OnEvent("MACRO","2","RUN") // when an User starts Macro ID=2
// (change ID if need)
{
AutoScanMode=0;
}
```

Последовательно переключать камеры на аналоговый монитор
Запуск последовательного переключения по макрокоманде 1, отключение по макрокоманде 2. Содержимое макрокоманды может быть пустым. Пример для 6 камер, переключение производится каждые 3 секунды.

Скрипт 5

Воспроизведение видео из протокола событий

```
OnEvent("EVENT_VIEWER","1","ACTIVATE_OBJECT")
{
[DoReact ("MONITOR","1","KEY_PRESSED","key<STOP>");
DoReact ("MONITOR","1","KEY_PRESSED","key<MODE_VIDEO>");
Wait(1);
DoReact("MONITOR","1","REPLACE", "slave_id<"+owner+">,audio_type<>,
audio_id<>,cam<"+objid+">,control<1>,name<>");
DoReact("MONITOR","1","ARCH_FRAME_TIME",  "cam<"+objid+">,date<"+date+">,time<"+time+">");
DoReact ("MONITOR","1","KEY_PRESSED","key<PLAY>");]
}
```

По двойному щелчку на строчке в протоколе событий, связанной с любой камерой, проигрывает на активном мониторе ближайшую запись по данной камере
Номера монитора и Протокола событий могут быть другими

Скрипт 6

Проиграть звуковой файл по событию

```
OnEvent ("...","...","...")
{
DoReact("PLAYER","1", "PLAY_WAV","file<c:\niss400\wav\cam_alarm_1.wav>");
}
```

По любому событию проигрывать любой звуковой файл.
wav-файл может быть другим, но обязательно нужно указывать полный путь. При этом проигрываться файл будет через правый канал линейного аудиовыхода, в не зависимости от настроек проигрывателя.

Скрипт 7

Окно телеметрии

```
OnEvent("DISPLAY","1","ACTIVATE")
{
DoReact("DIALOG","telemetry","RUN");
}
OnEvent("DISPLAY","1","DEACTIVATE")
{
DoReact("DIALOG","telemetry","CLOSE");
}
```

Открыть/закрыть окно телеметрии при активации \ деактивации экрана 1
Номер экрана может быть другим.

Скрипт 8

Синхронное видео со звуком

```
OnEvent("OLXA_LINE","5","ACCU_START")
{
n2a=1;
DoReact ("CAM","2","REC");
}
OnEvent("CAM","2","MD_START")
{
n2v=1;
DoReact ("CAM","2","REC");
}
OnEvent("OLXA_LINE","5","ACCU_STOP")
{
n2a=0;
if(!n2v) {DoReact ("CAM","2","REC_STOP"); }
}
OnEvent("CAM","2","MD_STOP")
{
n2v=0;
if(!n2a) {DoReact ("CAM","2","REC_STOP"); }
}
```

По акустопуску писать камеру с привязанным к ней звуком.
Нужно снять с камер атрибут "Запись тревог" и поставить камеры на охрану.
Номера аудиоканалов и видео могут быть другими.

Скрипт 9

Тревожный монитор по движению

```
OnEvent("CAM",N,"MD_START")
{
DoReact("MONITOR","1","ACTIVATE");
}
OnEvent("CAM",N,"MD_STOP")
{
hide=1;
[for(i=0;i<8;i=i+1) { if (CheckState("CAM",i,"ALARMED")) {
hide=0;
}
}]
if(hide) {
DoReact("MONITOR","1","DEACTIVATE");
}
}
```

Показать тревожный монитор если есть хоть одна камера с движением, если движения нет, то скрыть его
Пример для 8 камер. Номер монитора может быть другим.

Скрипт 10

Звук от активной камеры

```
OnInit ()
{
n=0;
}
OnEvent("MACRO","1","RUN")
{
n=1;
}
OnEvent("MONITOR","2","ACTIVATE_CAM")
{
if (n)
{
DoReact("PLAYER","1","START_LISTEN", "audio_id<"+GetObjectParam("CAM",cam,"audio_id")+">");
}
}
OnEvent("MACRO","2","RUN")
{
DoReact("PLAYER","1","STOP_LISTEN");
n=0;
}
```

Прослушивать звук с микрофона привязанного к камере в реальном времени при активации данной камеры в мониторе
Запуск режима по макрокоманде 1, останов по макрокоманде 2.
Номера монитора и аудиопроигрывателя могут быть другими.

Скрипт 11

Тревожный монитор.

```
OnInit()
{
counter=0;
}
OnEvent("CAM",T,"MD_START")
{
if(strequal(counter,"0"))
{
DoReact("MONITOR","2","REMOVE_ALL");
DoReact("MONITOR","2","ADD_SHOW","cam<"+T+">");
}
counter=str(counter+1);
}
OnEvent("CAM",M,"MD_STOP")
{
counter=str(counter-1);
if(strequal(counter,"0"))
{
DoReact("MONITOR","2","ADD_SHOW","cam<"+M+">");
}
}
```

Отображать видео последней тревожной камеры при отсутствии движения по всем камерам
Номер монитора может быть другим.

Скрипт 12

Проигрывание звукового файла

```
OnEvent("MACRO","7","RUN")
{
flag0=1;
[for(i=1;flag0;flag0)
{
DoReact("PLAYER","1","PLAY_WAV","file<C:\"+"Niss400\Wav\cam_alarm_1.wav>");
Wait(3);
}]
}
OnEvent("MACRO","8","RUN")
{
flag0=0;
}
```

Проигрывание звукового файла от прихода одного события, до прихода другого события (В данном случае это запуск макрокоманд).
Звуковой файл должен длиться не больше количества секунд, которое указано в операторе Wait.
События могут быть другими.

Приложение 1 (объекты)

Компьютер

Идентификатор типа для данного объекта - **SLAVE**

События

CONNECTED – Подключение
DISCONNECTED – Отключение
PROGRAM - Изменена программа
PROTOCOL_RCVD – Протокол получен
REGISTER_USER – Регистрация пользователя

Действия

CONNECT_OTHER – Подключиться к ядрам
SYNC_PROTOCOL - Получить протоколы
CONNECT_ONE - Подключиться к компьютеру
DISCONNECT_ONE - Отключиться от компьютера

Состояния

Нет

Отдел

Идентификатор типа для данного объекта - **DEPARTMENT**

События

Нет

Действия

Нет

Состояния

Нет

Временная зона

Идентификатор типа для данного объекта - **TIME_ZONE**

События

ACTIVATE – Начало зоны
DEACTIVATE – Конец зоны

Действия

Нет

Состояния

ACTIVATED – С момента начала до конца зоны
DEACTIVATED – С конца до начала зоны

Область

Идентификатор типа для данного объекта - **ZONE**

События

Нет

Действия

Нет

Состояния

Нет

Раздел

Идентификатор типа для данного объекта - **REGION**

События

ARM_LOCK – На данном разделе произошло охранное блокирование
ARM_UNLOCK – Охранное обокирование раздела снято
FIRE_UNLOCK – На данном разделе произошло пожарное разблокирование
FIRE_RESTORE – Снято пожарное разблокирование раздела
ARM – Раздел поставлен на охрану
DISARM – Раздел снят с охраны
PANIC_LOCK – На данном разделе произошло тревожное блокирование
PANIC_RESTORE – Снято тревожное блокирование

Действия

Нет

Состояния

Нет

Пользователь

Идентификатор типа для данного объекта - **PERSON**

События

REGISTERED – Регистрация пользователя
UNREGISTERED - Завершение работы пользователя

Действия

Нет

Состояния

Нет

Права пользователя

Идентификатор типа для данного объекта - **RIGHTS**

События
Нет
Действия
Нет
Состояния
Нет

Уровень доступа

Идентификатор типа для данного объекта - **LEVEL**

События
Нет
Действия
Нет
Состояния
Нет

Макрокоманда

Идентификатор типа для данного объекта - **MACRO**

События
RUN - Генерируется после выполнения данной макрокоманды
Действия
RUN - Выполнить действия макрокоманды
Состояния
NORM -

Плата видеоввода

Идентификатор типа для данного объекта - **GRABBER**

События

Нет

Действия

SETUP - устанавливает параметры платы видеоввода

☛ *Параметры:*

chan *– номер канала (0,1,2,…)*
mode *– скорость оцифровки (0 – максимальная, 1 – средняя, 2 – минимальная)*
resolution *– разрешение (0– четверть кадра (384x288), 1 – полукадр (768x288), 2 – кадр (768x576))*
format *– формат сигнала(PAL, NTSC)*
drives *– диски для записи видеоархива (DRIVE1:\, DRIVE2:\, … DRIVEN:\)*
cams *– количество подключенных камер*

Пример:
DoReact("**GRABBER**", "**1**", "**SETUP**", "**chan**<1>, **mode**<0>, **resolution**<1>, **format**<PAL>");

SET_DRIVES - устанавливает диски для записи видеоархива

☛ *Параметры:*

drives *– диски для записи видеоархива*

Пример:
DoReact("**GRABBER**", "**1**", "**SET_DRIVES**", "**drives**<D:\,F:\>");

MUX1_OFF - Отключить аналоговый выход 1
MUX2_OFF - Отключить аналоговый выход 2
MUX3_OFF - Отключить аналоговый выход 3

Состояния

Нет

Камера

Идентификатор типа для данного объекта - **CAM**

События

MD_START – Тревога. Генерируется в момент обнаружения движения в кадре детектором движения при поставленной на охрану камере
MD_STOP – Конец тревоги. Снятие сигнала тревоги детектора движения
ATTACH – Подключение. Генерируется в момент появления видеосигнала
DETACH – Обрыв. Генерируется в момент пропадания видеосигнала
ARM – Генерируется в момент постановки камеры на охрану
DISARM – Генерируется в момент снятия камеры с охраны
REC – Генерируется в момент начала записи видео на диск
FILE_REC_ERROR – Ошибка записи на диск
BLINDING – Камера залеплена
UNBLINDING – Камера открыта
PRINT – Печать.Генерируется в момент начала печати видеокадра

Действия

SETUP – устанавливает (изменяет) параметры камеры

☛ Параметры:

rec_priority - *приоритет записи (0 – обычный, 3 – все ресурсы)*
compression - *степень компрессии (0 – компрессия отсутсвует, 1- макс. качество, …, 5 – мин. качество)*
sat_u - *уровень цветности (0 – мин, 10 – макс)*
proc_time - *период дозаписи (с)*
hot_rec_period - *период горячей записи (мс)*
manual - *ручная установка уровней яркости и контрастности (0 – выключено, 1 – включено)*
md_size - *детектор движения – размер (0 – max, 10 – min)*
md_mode - *режим записи пауз (1 – включено, 0 выключено)*
audio_type - *тип звукового сопровождения*
audio_id - *номер микрофона (пустой параметр, если нет микрофона)*
priority - *приоритет обработки (0 – обычный, 1 - 1/3, 2 – 1/2)*
md_contrast - *детектор движения – контраст (0 – max, 10 – min)*
alarm_rec - *запись тревог (1 – включено, 2 – выключено*
pre_rec_time - *период отката (с)*
hot_rec_time - *время горячей записи (с)*
rec_time - *период записи (мс)*
mux - *номер канала (0 – 1 канал, 15 – 16 канал)*
color - *цветность (0 – выключено, 1 – включена)*
contrast - *контрастность (0 – мин, 10 – макс)*
bright - *яркость (0 – мин, 10 – макс)*

Действия (продолжение)

- *Параметры (***SETUP,** *продолжение)*:
  telemetry_id - *идентификатор модуля телеметрии (id поворотника)*
  parent_id - *идентификатор видеоканала, которому принадлежит камера*

DELATE – Отключить камеру
START_VIDEO – включает видеопоток для текущей камеры

- Параметры:
  slave_id - *имя компьютера, к которому подключена камера*
  compress - *степень компрессии*
  register_only -

STOP_VIDEO – выключает видеопоток для текущей камеры

- *Параметры:*
  slave_id *– имя компьютера, к которому подключена камера*

REQUEST_MASK –

- *Параметры:*
  mask *– маска*

ACTIVATE – Вывести на монитор

- *Параметры:*
  monitor *– номер монитора*

REC – Начать запись (Параметр - время "горячей" записи, сек.)
REC_STOP – Остановить запись видеоданных
ARM – Поставить на охрану
DISARM – Снять с охраны
MUX1,2,3 – Коммутировать сигнал данной камеры на аналоговый выход (1, 2 или 3) платы захвата видео
STOP_VIDEO - Остановить видеопоток
REC_ROLLBACK - Начать запись с откатом

Состояния

ALARMED – Тревозное (Зафиксировано движение в поле зрения видеокамеры)
ARMED – Поставлена на охрану
DETACHED – Отсутствует видеосигнал от камеры
DISARMED – Камера снята с охраны
RELAXED – Камера включена и нормально работает

Видеошлюз

Идентификатор типа для данного объекта - **GATE**

События

Нет

Действия

Нет

Состояния

Нет

Луч

Идентификатор типа для данного объекта - **RAY**

События

ON – Замкнут. Генерируется в момент перехода луча из разомкнутого в замкнутое состояние
ARM – Луч поставлен на охрану
OFF – Разомкнут. Генерируется в момент перехода луча из замкнутого в разомкнутое состояние
DISARM – Луч снят с охраны
ALARM – Зафиксировано стабатывание датчика
NOT_VALID_STATE – Зона не готова
CONFIRM – Генерируется в момент принятия тревоги оператором

Действия

ARM – Луч ставится на охрану
CONFIRM – Луч фиксирует принятие тревоги и генерирует событие "тревога принята"
DISARM – Луч снимается с охраны

Состояния

ALARMED – Луч зафиксировал тревогу
ARMED – Луч поставлен на охрану
CONFIRMED – Луч зафиксировал тревогу и тревога принята системой
DISARMED – Луч снят с охраны

Реле

Идентификатор типа для данного объекта - **RELE**

События

ON – Реле включено
OFF – Реле выключено

Действия

ON – Включает реле
OFF – Выключает реле

Состояния

ON – Реле включено
OFF – Реле выключено

Окно запроса оператора

Идентификатор типа для данного объекта - **DIALOG**

События

Нет

Действия

SETUP - Настройка диалогового окна.

☛ *Параметры:* ***x***<>,***y***<>- *координаты вывода*

RUN - Запуск диалогового окна
CLOSE - Закрывает последнее открытое диалоговое окно
CLOSE_ALL - Закрывает все открытые диалоговые окна

Состояния

Нет

Архиватор

Идентификатор типа для данного объекта - **ARCHIVER**

События

Нет

Действия

START - Запускает процесс архивации данных. Если приемник данных - стример, то не производится перемотка ленты в начало
START_BEGIN - Запускает процесс архивации данных. Если приемник данных - стример, лента перематывается в начало перед записью
STOP - Останавливает процесс архивации
SHOW_PANNEL - Отображение панели управления

Состояния

Нет

Клавиатура

Идентификатор типа для данного объекта - **KEYB**

События

Нет

Действия

Нет

Состояния

Нет

Звуковая плата

Идентификатор типа для данного объекта - **OLXA**

События

Нет

Действия

Нет

Состояния

Нет

Микрофон

Идентификатор типа для данного объекта - **OLXA_LINE**

События

ARM – Включён встроенный датчик акустопуска
DISARM – Выключен встроенный датчик акустопуска
REC – Произошло принудительное включение записи сигнала
INCOMING_NUMBER – Входящий телефонный номер
OUTGOING_NUMBER - Исходящий телефонный номер
ACCU_START - Включение акустопуска
ACCU_STOP - Выключение акустопуска
RESET - Подключение микрофона
REC_STOP – Произошло принудительное выключение записи сигнала

Действия

ARM – Включить запись
DISARM – Выключить запись

Состояния

BLUE – Снят с охраны
GREEN - Нет сигнала
YELOW - Поставлен на охрану
RED - Начало записи

Аудиопроигрыватель

Идентификатор типа для данного объекта - **PLAYER**

События

Нет

Действия

Нет

Состояния

Нет

Модуль голосового оповещения

Идентификатор типа для данного объекта - **VDIAL**

События

SENT - Генерится после успешного соединения
SEND_ERROR - Генерится после неудачного соединения

Действия

SEND - Послать.

☛ *Параметры:*
phone *- номер телефона*
file *- полный путь к звуковому файлу.*

Например:
DoReact("**VDIAL**",**ObjID**,"**SEND**","**phone**<1234567>,**file**<X.WAV>");

Состояния

Нет

Почтовый клиент

Идентификатор типа для данного объекта - **MAIL_SENDER**

События

SENT - Генерится после отправки почты
SEND_ERROR - Почта не отправлена

Действия

SETUP - Установки для отправки почтовых сообщений.

☛ *Параметры:*
smtp *- IP-адрес сервера для модемного соединения:*
connection *- имя соединения*
username *- имя пользователя*
password *- пароль*

SEND - Послать.

☛ *Параметры:*
from *- адрес почтового ящика откуда исходит сообщение*
to *- адрес почтового ящика куда отправить сообщение*
cc *- адрес почтового ящика куда отправить копию сообщения*
bcc *- адрес почтового ящика куда отправить слепую копию сообщения*
subject *- тема письма*
body *- тело письма*
attachments *- прикрепленный файл*

Состояния

Нет

Экран

Идентификатор типа для данного объекта - **DISPLAY**

События

Нет

Действия

ACTIVATE - Показывает данный экран на дисплее компьютера
DEACTIVATE - Скрывает данный экран

Состояния

Нет

Карта

Идентификатор типа для данного объекта - **MAP**

События

Нет

Действия

Нет

Состояния

Нет

Протокол событий

Идентификатор типа для данного объекта - **EVENT_VIEWER**

События

ACTIVATE_OBJECT - Генерится в момент двойного щелчка левой кнопки мыши в любом поле протокола событий

Действия

Нет

Состояния

Нет

Монитор

Идентификатор типа для данного объекта - **MONITOR**

События

Нет

Действия

ACTIVATE - Показывает виртуальный монитор на экране
DEACTIVATE - Скрывает монитор

Состояния

Нет

Контроллер телеметрии

События (продолжение)

GUARD_VP - Поставлено на охрану по ВП
UNGUARD_VP - Канал снят с охраны по ВП
UNLOCK - Снята относит. блокировка
NOACCESS_GROUP - Нет доступа по группе
LOCK_APP - Включена апп. блокировка
UNLOCK_APP - Выключена апп. блокировка
ACCESS_IN - Нормальный вход по ключу
ACCESS_OUT - Нормальный выход по ключу
EXIT_OUT_TIME - Выход вне врем. профиля
ALARM_VP - Вход тревоги включен по ВП
UNALARM_VP - Вход тревоги выключен по ВП
ALARM_PC - Вход тревоги включен от ПК
UNALARM_PC - Вход тревоги выключен от ПК
AUTO_REGIM - Переход в автономный режим
ACCESS_IN_FACT - Фактический вход
ACCESS_OUT_FACT - Фактический выход
RELE2_ON - Включено реле 2
RELE2_OFF - Выключено реле 2
RESET_CONTROLLER - Аппаратный сброс контроллера
OPEN_RTE - Открывание двери по RTE
OPEN_PC - Дверь открыта по команде ПК
CLOSE_PC - Дверь закрыта по команде ПК
OPEN_VP - Дверь открыта по ВП
CLOSE_VP - Дверь закрыта по ВП
RELE_ON_PC - Включение реле от ПК
RELE_OFF_PC - Выключение реле от ПК
RELE_ON_VP - Включение реле по ВП
RELE_OFF_VP - Выключение реле по ВП
SIGNAL_OFF - Выкл. сигнал незакрытой двери
EXIT_RTE_FACT - Фактический выход по RTE
CLOSE_OFFLINE - Дверь закрыта по Off-Line
EXIT_DISTANCE - Выход по дистанционной кнопке
TAMPER_SOUND_OFF - Снятие звука тампера кожуха

Действия

OPEN - Открыть дверь
CLOSE - Закрыть дверь
LOCK - Установить относит. блокировку
UNLOCK - Снять относит. блокировку
ABS_LOCK - Установить абс. блокировку
ABS_UNLOCK - Снять абс. блокировку
GUARD - Поставить на охрану
UNGUARD - Снять с охраны
RELE_ON - Включить доп. реле
RELE_OFF - Выключить доп. реле
RELE2_ON - Включить реле 2
RELE2_OFF - Выключить реле 2

Состояния

NORMAL - Дверь закрыта
OPENED - Дверь открыта
DETACHED - Связь потеряна
ALARMED - Связь потеряна

Идентификатор типа для данного объекта - **TELEMETRY_CARD**

События

Нет

Действия

Нет

Состояния

Нет

Поворотное устройство

Идентификатор типа для данного объекта - **TELEMETRY**

События

Нет

Действия

SET_PRESET - Установить пресет
CLEAR_PRESET - Очистить пресет
GO_PRESET - Перейти в пресет
AUTOFOCUS_ON - Автофокус
UP - Вверх
DOWN - Вниз
LEFT - Влево
RIGHT - Вправо
HOME - Домой
STOP - Стоп
FOCUS_IN - Фокус +
FOCUS_OUT - Фокус -
FOCUS_STOP - Фокус стоп
RIGHT_DOWN - Вниз-вправо
RIGHT_UP - Вверх-вправо
LEFT_DOWN - Вниз-влево
LEFT_UP - Вверх-влево
IRIS_OPEN - Открыть диафрагму
IRIS_CLOSE - Закрыть диафрагму
IRIS_STOP - Остановть диафрагму
PATROL_LEARN - Обучение
PATROL_STOP - Стоп
PATROL_PLAY - Патрулирование
D2ON - Супер Динамик вкл.
D2OFF - Супер динамик откл.
AUTOPAN_START_P - Задание стартовой точки
AUTOPAN_END_P - Задание конечной точки
AUTOPAN_START - Начать автоповорот
AUTOPAN_STOP - Окончить автоповорот

Состояния

Нет

Приложение 2 (Настройки VideoInspector Global через реестр Windows)

Некоторые специальные настройки системы VideoInspector Global вы можете провести через реестр Windows.
Для запуска программы редактирования реестра Windows нажмите кнопку "Пуск" (Start) - "Выполнить" (Run), наберите "regedit" и нажмите кнопку "OK"

☛ *Программа редактирования реестра Windows запускается только если операционная система загружена с правами локального (Administrator) или доменного администратора (Domain Admin).*

☛ *Не рекомендуется изменять какие-либо значения в реестре Windows кроме разделов и параметров, описанных ниже.*

Табл. 6: Http.exe

HKEY_CURRENT_USER\\Software\\ISS\\Niss400\\HttpMod\\VideoI\\

Ключ	Значение по умолчанию	Комментарий
m_IP_VidI	"127.0.0.1"	IP-адрес http-сервера
m_Compress	2	Степень компрессии
iss_user_counter	10	Максимальное количество подключенных пользователей к серверу
m_DefaultPage	"Index.htm"	Имя стартового файла по умолчанию
m_BasePath		Путь к файлам-ресурсам сервера: "<VI Dir>\\webroot"
m_Port_HTTP	80	Порт http-сервера

HKEY_CURRENT_USER\\Software\\ISS\\Niss400\\HttpMod\\VideoI\\

tcpdelay	0	Настройки сокета (IPPROTO_TCP). 1 - запретить использовать алгоритм NAGLE. Алгоритм NAGLE блокирует передачу маленьких пакетов, до приема подтверждения о получении предыдущего фрейма (пакета). Это позволяет за счет увеличения размера пакета исключить перегрузку сети.
tcpsndbuf	0	Настройки сокета (SOL_SOCKET). Определяет размер буфера зарезервированного на отправку пакетов.

Табл. 7: Telemetry.exe

HKLM\\Software\\ISS\\Niss400\\Telemetry\\

Ключ	Тип	Значение по умолчанию	Комментарий
Delay	S	250	Время задержки (мс) между посылкой команд
PriorityDelay	S	30000	Время, по истечении которого, если занятое устройство не используется, считается свободным

Табл. 8: Vdial.exe

HKLM\\Software\\ISS\\Niss400\\VDIAL

Ключ	Тип	Значение по умолчанию	Комментарий
infpath	S		Путь к inf файлу ("<WindowsDirectory>\\Inf")
inf	S		Имя inf файла

HKLM\\Software\\ISS\\Niss400\\VDIAL			
port	S		Порт (например, "COM1")
waitfordialtone	DWORD	1	При наборе ждать
HKLM\\Software\\ISS\\Niss400\\Video			
pulse	DWORD	0	Использовать импульсный набор

Табл. 9: Videoi.exe

HKLM\\Software\\ISS\\Niss400\\Video\\		
Ключ	Значение по умолчанию	Комментарий
Delta	1	Использовать Delta-компрессию
DbMode	0	Регулируется показ ReceiptViewer'а доступ в базу VideoI (2 - запретить, 0 и 1 - разрешить). А также: 0 - разрешить чтение/ запись чеков в БД 1 - запретить запись чеков в БД 2 - запретить чтение/ запись чеков в БД.
FreeMB	100	Минимальное свободное дисковое пространство на диске, на который записывается архив.
MaxFrames	500	Количество фреймов в одном файле при непрерывной записи.
DrawCursor	0	(0, 1) Выключить/ включить отображение курсора средствами VideoInspector Global

HKLM\\Software\\ISS\\Niss400\\Video\\

Index	2	Использование индексов файловой системы
HideMask	0	Запретить редактировать маску детектора движения
NewMD	0	READ ONLY Всегда должен = 0
Compression	1	Разрешить запись компрессированного видео
QueueNormal	1000	Количество сообщений (принимаемое за "нормальный" уровень) в очереди VideoI. При превышения этого уровня генерируется Event: "SLAVE"|GetComputerName()|QUEUE_NORMAL
QueueWarning	2000	Количество сообщений (принимаемое за "тревожный" уровень) в очереди VideoI. При превышения этого уровня генерируется Event: "SLAVE"|GetComputerName()|QUEUE_WARNING
QueueCritical	3000	Количество сообщений (принимаемое за "тревожный" уровень) в очереди VideoI. При превышения этого уровня генерируется Event: "SLAVE"|GetComputerName()|QUEUE_CRITICAL
PostMessageCount	100	Количество сообщений извлекаемых из системной очереди VideoI на каждом шаге обработки внутренних сообщений.

HKLM\\Software\\ISS\\Niss400\\Video\\

sigma	35	Параметры детектора засвета (контраст). Чувствительность выше, чем больше sigma
threshold	1500	Параметры детектора засвета (яркость). Чувствительность выше, чем меньше threshold
cfg		Имя cfg-файла videoi.exe
Ping	1	1 - Разрешить посылку ping
DecompressThread	0	READ ONLY. 1 - применять m_arAsyncDecompressor.
SendCompressed	1	Использовать компрессию при отправке видео
Delay	0	Использовать FIFO по каждой камере при отправке видео
ExportDir	VideoiDir+"Export\\"	Поддиректория каталога экспорта стоп-кадров или клипов
Overlay	1	READ ONLY. Использование старого оверлея.
deinterlace	1	Использовать деинтерлейсинг
ArchiveSubtitles	1	Показывать субтитры из архива при выводе на экран
reg	VideoiDir++"\\ip.reg"	Путь к файлу настроек IP-камеры

HKLM\\Software\\ISS\\Niss400\\Video\\

show_debug_histogram	0	Выводить окно с гистограммой яркости для каждой камеры. Для настройки функции определения засветки/ закрытия камеры. Служебная функция.
noise_thresold	20	(Порог на гистограмме яркости, значения выше которого, учитываются при вычислении эффективной ширины гистограммы.) Уровень шума. При работе детектора закрытия/ засветки точки, в которых значения гистограммы яркости меньше заданного уровня, игнорируются.
adapt_time	120	Время за которое вычисляется средняя гистограмма яркости. Задаётся в S-х долях секунды. Например, значение 120 соответствует одной минуте.
PlayerQueueLenght	20	Число фреймов которые будут храниться в кэше при обращении к архиву. Если видео воспроизводится скачками, то рекомендуется увеличить этот параметр. Например, если архив просматривается со стримера, то оптимально проставить 500.

Табл. 10: Backup.exe

HKLM\\SOFTWARE\\ISS\\Niss400\\Backup

Ключ	Тип	Значение по умолчанию	Комментарий
folder	REG_SZ	"C:\\Backup"	Папка куда будет копироваться архив
size	REG_BINARY	650	Ограничение на суммарный размер AVI-файлов
continuing	REG_SZ	1	
fps	REG_SZ	1000	Framerate по умолчанию для продуцируемого AVI. Fps = code 0.5 = “5” 1 = “10” 3 = “30” 5 = “50” 15 = “150” 30 = “300” RealTime = “1000”
Last Backup	REG_BINARY		TimeStamp последнего backup'a
timelimit	REG_DWORD	30 min	Не используется
compvars	REG_BINARY		Параметры кодирования видео
wfx	REG_BINARY		Параметры кодирования аудио

Табл. 11: Глобальные параметры

HKLM\\SOFTWARE\\ISS\\Niss400\\

Ключ	Тип	Значение по умолчанию	Комментарий

HKLM\\SOFTWARE\\ISS\\Niss400\\

debug	S	0	Уровень вывода отладочной информации: 0 – запретить вывод отладочной информации 1 – выводить всю отладочную информацию 2 – выводить предупреждения и ошибки 3 – выводить только ошибки
SyncTime	S	0	(0, 1) Включение/ выключение синхронизации времени
Core IP Address	S	"127.0.0.1"	Адрес текщего запущенного ядра (READ ONLY) (для VideoServer и УРМ-А «127.0.0.1» для УРМ-М – адрес ведущего ядра)
PhotoCores	S		Перечень (через ",") ядер Инспектора, которым пересылаются файлы (стоп-кадры).
ActivateWithoutAKey	S	0	Разрешает активацию панели настройки Инспектора (VideoServer и УРМ-А) и Панели Выбора Экранов (УРМ-М) без нажатия Ctrl
MMF	S	1	Разрешается использование MMF-транспорта
URAttempts	S	3	Количество попыток на ввод пароля без задержки между попытками
URDelay	S	60	Задержка в случае превышения ограничения попыток на ввод пароля (в сек).

HKLM\\SOFTWARE\\ISS\\Niss400\\			
DisableSaveState	S	(0, 1)	Вкл/Выкл сохранение состояния объектов в базу данных
PersonWindowPlacement	S		
HKLM\\SOFTWARE\\ISS\\Niss400\\DNS\\			
<Имя сервера>	S	IP-адрес сервера	Вводится список имён и адресов серверов системы. При потере связи клиента с сервером, он попытается автоматически подключиться к следующему серверу из списка. Имеет смысл для клиентов системы.

Табл. 12: Archiver.exe

HKLM\\Software\\ISS\\Niss400\\Archiver\\			
Ключ	Тип	Значение по умолчанию	Комментарий
Mode	S	1 или 0	1- многопотоковость, 0 - нет
MaxQueryTime	S	180	Время ожидания ответа клиента архиватором (секунды)
Tape	S	0	Номер, подставляемый в шаблон \\\\.\\TAPE%d, который является именем открываемого стримера
BlockSize	S	Минимум из TAPE_GET_DRIVE_PARAMETERS::MaximumBlockSize и 65535	Размер блока данных при считывании/записи в файл на ленте

HKLM\\Software\\ISS\\Niss400\\Archiver\\

m_dwTime	S	0	
m_nCurVolume	S	0	
FreeMB	S	100	Зарезервированный размер свободного места на несъёмных носителях
drv	S	""	

Табл. 13: AudioI.exe

HKLM\\Software\\ISS\\Niss400\\AudioI\\

Ключ	Тип	Значение по умолчанию	Комментарий
\\Olxa Compression	S	4	
\\Olxa\\AON Mode	S	1	Режим работы АОН-а
\\Olxa\\AON Level	S	15	Индекс уровня запроса АОН
\\Olxa\\AON Delay	S	3	Индекс времени до выдачи запроса АОН в линию
\\Olxa\\AON Duration	S	3	Индекс длительности запроса АОН
\\Olxa\\AON Request	S	5	Индекс количества перезарпосов
\\Olxa\\Lines\\<nLine>HookLevel	S		Индекс порога снятия трубки (nLine – номер линии)
\\Olxa\\Lines\\<nLine>RingLevel	S		Индекс порога звонок (nLine – номер линии)

Табл. 14: Keyb.exe

HKLM\\Software\\ISS\\Niss400\\Keyb\\

Ключ	Тип	Значение по умолчанию	Комментарий
Prefix	S	192	Код клавиши READ ONLY

Табл. 15: Map.exe

HKLM\\Software\\ISS\\Niss400\\

Ключ	Тип	Значение по умолчанию	Комментарий
\\DB Data	S	“”	Connection string для соединения с базой данных. Если не задана, строится программно. В реестре хранится в зашифрованном виде. READ ONLY?.
\\Map ActiveMap	S	“”	Имя текущей активной карты. Для внутреннего использования.

Табл. 16: Core.exe

HKLM\\Software\\ISS\\Niss400\\

Ключ	Тип	Значение по умолчанию	Комментарий
sequence_counter	S		Для внутреннего использования
OnlyLocalProtocol	S		Вести протокол только локального компьютера
SyncTime	S		Включить синхронизацию времени между ядрами (React SLAVE| |SET_CLOCK|<date><time>)

HKLM\\Software\\ISS\\Niss400\\

\\DB Data	S		Connection string для соединения с базой данных. Если не задана, строится программно. В реестре хранится в зашифрованном виде. READ ONLY?.
\\DB Protocol	S		В реестре хранится в зашифрованном виде. READ ONLY?.
\\DB Source	S		Имя источника данных. В реестре хранится в зашифрованном виде. READ ONLY?.
\\RAS<Slave ID>	Ключ		READ ONLY?
\\RAS\\<Slave ID>Connection	S		Заносится значение параметра «connection». Если этот параметр передается пустым, то ключ SOFT-WARE\\ISS\\Niss400\\RAS\\<SlaveID> удаляется из реестра
\\RAS\\<Slave ID>Username	S		Имя пользователя по умолчанию
\\RAS\\<Slave ID>Password	S		Пароль по умолчанию
\\RAS\\<Slave ID>DialAttempts	S		Количество попыток дозвона
\\RAS\\<Slave ID>LANFirst	S	(0, 1)	Вкл/Выкл попытки первоначально связаться по LAN
\\DNS<Имя хоста>	S		IP-адрес включенного в конфигурацию хоста <имя хоста>. Если не задан, то используется GetHostByName.
MessageTTL	S	30	Для внутреннего использования для класса MsgLogger. READ ONLY 30

HKLM\\Software\\ISS\\Niss400\\

SyntaxEdit	S	1	Индикатор того, используется для редактирования кода сцинтилловский контрол CsyntaxEdit, или стандартный EditBox. По умолчанию используется CsyntaxEdit. READ ONLY 1

Табл. 17: Slave.exe

HKLM\\Software\\ISS\\Niss400\\

Ключ	Тип	Значение по умолчанию	Комментарий
URAttempts	S	3	Количество попыток на ввод пароля без задержки между попытками
URDelay	S	60	Задержка в случае превышения ограничения попыток на ввод пароля (в сек).

HKLM\\Software\\ISS\\Niss400\\

\\DNS<Имя хоста>	S	<IP-адрес>	Список ведущих серверов. УРМ-М пытается установить связь с ведущим ядром подключаясь по очереди к компьютерам, перечисленным в списке. При разрыве связи с ведущим ядром, УРМ-М пытается использовать в качестве ведущего одно из ядер, перечисленных в списке (продолжает перебор списка). Перебор списка циклический – т.е. при достижении конца начинается с начала.

Некоторые ключи не присутствуют в реестре по умолчанию, но их можно добавить самостоятельно.

Табл. 18: Изменение шрифта

HKLM\\Software\\ISS\\Niss400\\Video\\CustomFont (Позицию CustomFont нужно создавать самостоятельно на основе Video. Так же нужно создавать самому все нижеприведённые ключи)

Ключ	Значение по умолчанию	Комментарий
size		Размер шрифта
name		Тип шрифта (например, Arial CYR)
bold		Жирность шрифта: 0 - не жирный, 1 - жирный.

Табл. 19: Поддержка нескольких мониторов

HKLM\\Software\\ISS\\Niss400\\Video\\

Ключ	Тип	Значение по умолчанию	Комментарий
mon_h	REG_SZ		Высота экрана монитора в пикселях (например, mon_h=1224)
mon_w	REG_SZ		Высота экрана монитора в пикселях (например, mon_w=1280)