

DR. BOB DAVIDOV

Программные имитаторы сигналов OPC сервера

Цель работы: Изучить принципы построения сигналов произвольной формы OPC сервера.

Задача работы: Проектирование выходных сигналов ModBUS OPC сервера компании ИнСАТ.

Приборы и принадлежности: Персональный компьютер. Установочная программа OPC сервера.

ОБЩИЕ СВЕДЕНИЯ

OPC DA (Data Access) сервер обеспечивает обмен данными в реальном времени между клиентами (SCADA системы, МатЛАБ, LabView, электронные таблицы, и др. программные среды) с Программируемыми Логическими Контроллерами (ПЛК), Распределенными Системами Управления (PCY), Человеко-Машинными Интерфейсами (ЧМИ), оборудованием с Числовым Программным Управлением (ЧПУ) и другими устройствами.

Modbus RTU/ASCII/TCP OPC сервер компании ИнСАТ имеет бесплатную конфигурацию на одно рабочее место на 32 тега. Сервер способен работать в двух режимах – DA (Data Access) и HDA (Historical Data Access), последний режим предоставляет доступ к уже сохраненным данным. Для организации хранения архивов опрашиваемых переменных OPC использует встроенный SQL сервер.

OPC имеет в своем составе поддержку простого сценарного языка, что позволяет проводить предварительную обработку данных после их считывания из внешних устройств, а также перед записью в них. Возможно использование сценариев для написания новых драйверов (как для Modbus протоколов, так и любых других), сохранения архивов в SQL-сервере, написания имитаторов сигналов, вычисления косвенных параметров, работы с признаками качества и т.п. Сценарии могут использоваться на уровне коммуникационных узлов, устройств и подустройств, отдельных тегов. Встроенный редактор обеспечивает стандартный сервис - подсветку ключевых слов, удобную работу с тегами и библиотеками.

Для облегчения тиражирования OPC также поддерживает возможность экспорта и импорта конфигураций устройств. В основную поставку OPC включены все приборы фирм ОВЕН и ICP DAS работающих по протоколу Modbus. Пользователь может создавать, сохранять и распространять собственные библиотеки устройств.

Средства и возможности сценарного языка OPC сервера включают:

- встроенный редактор с подсветкой ключевых слов, доступом к тегам и библиотекам;
- поддержку простой разработки OPC DA и HDA серверов для любых протоколов связи;
- вычисление значения переменной после чтения или перед записью;

- работу с признаками качества;
- поддержку разработки симуляционных устройств (используя скрипт);
- передачу данных в любой SQL сервер

ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Задание 1. Написание имитаторов сигналов OPC сервера.

1. Запустите конфигурацию ex2.mbc демонстрационного OPC сервера.
2. Наблюдайте изменение синусоидального сигнала.
3. Остановите сервер.
4. Откройте редактором готовый скрипт сигнала **sin**.

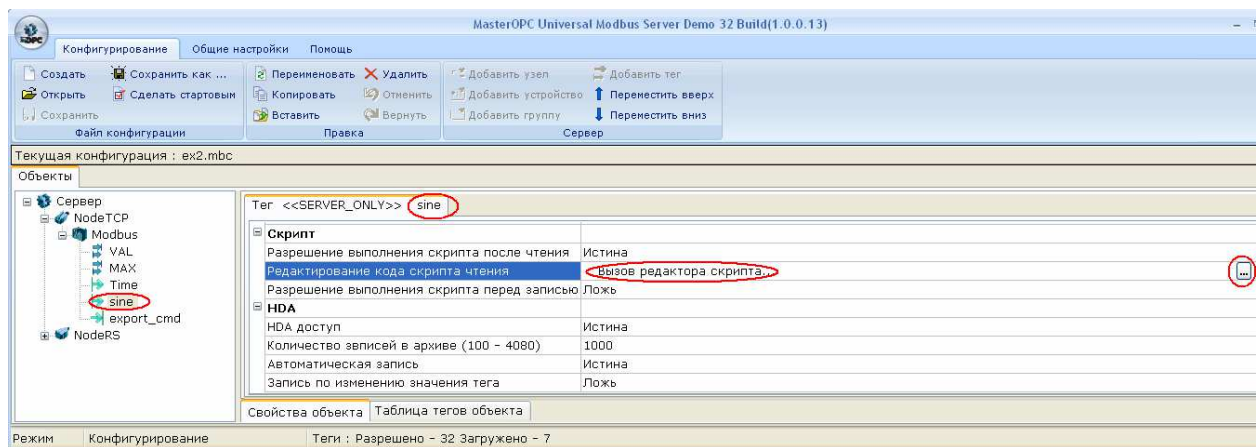


Рис. 1. Вызов редактора скрипта

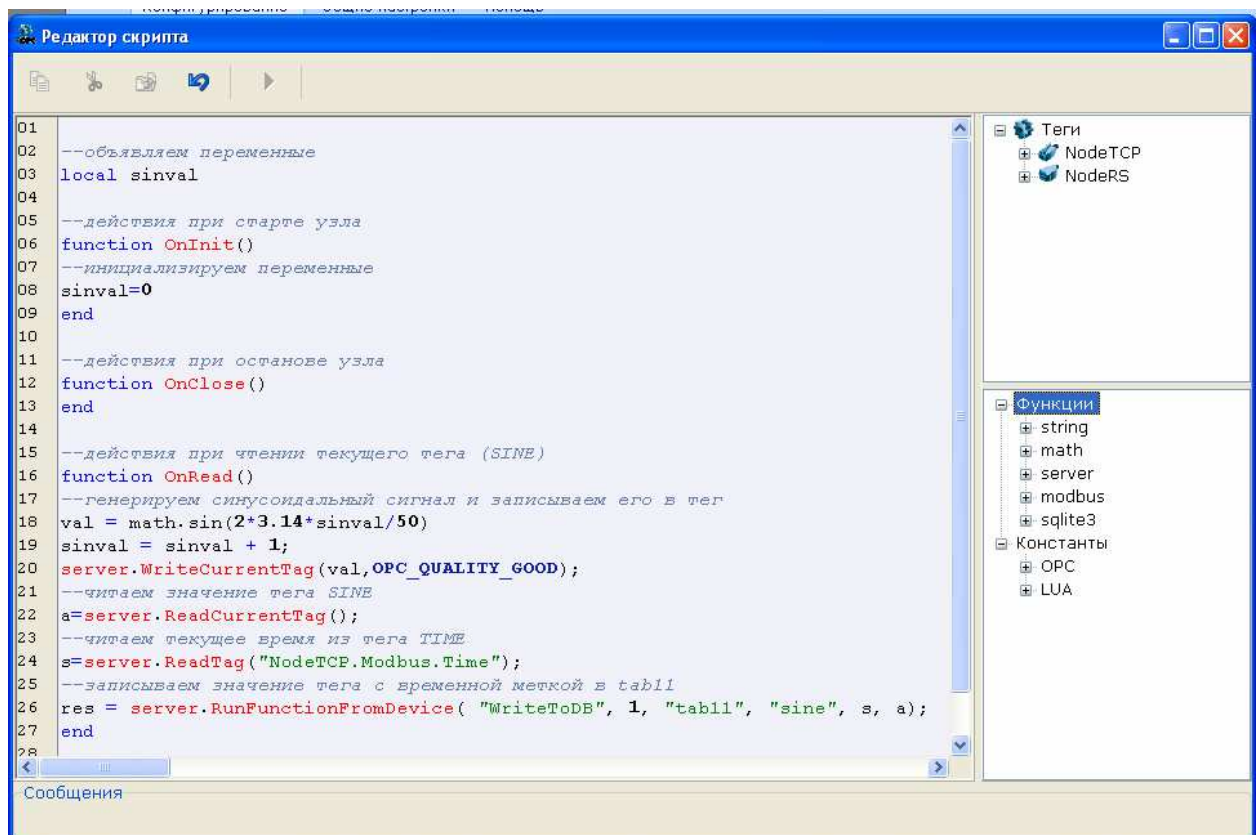


Рис. 2. Скрипт имитатора синусоидального сигнала.

5. Рассмотрите элементы кода генератора сигнала.
6. Ознакомьтесь со следующими функциями по работе с тегами.

ФУНКЦИИ РАБОТЫ С ТЕГАМИ (из раздела “ПОМОЩЬ” OPC сервера)

server.GetCountTags

Возвращает общее количество тегов узла, устройства или подустройства (в зависимости от уровня расположения скрипта).

Пример

```
CT = server.GetCountTags();
```

В скрипте тега возвращает 1.

server.ReadCurrentTag

Возвращает значение, качество и время текущего тега.

Пример

```
TV,TQ,TS = server.ReadCurrentTag();
```

server.ReadTag

Возвращает значение, качество и время произвольного тега по его имени.

Пример

```
TV,TQ,TS = server.ReadTag("Node.Device.Group.Tag");
```

В данной функции, как и в функции *server.WriteTag* , могут использоваться следующие псевдо-имена для текущего узла, устройства или подустройства:

- **\$(N)** – текущий узел;
- **\$(D)** – текущее устройство;
- **\$(S)** – текущее подустройство.

Пример

Ниже приведены примеры функций в том случае, если дерево объектов имеет следующий вид:

```

NODE1
  DEVICE1
    SUBDEVICE1
      TEG1
    SUBDEVICE2
      TEG1
  TEG1
```

--скрипт устройства **DEVICE1**, суммирует 2 тега

--и записывает результат в тег **NODE1.DEVICE1.TEG1**

```
t1,q1,ts1 = server.ReadTag("$ (N).$(D).SUBDEVICE1.TEG1");
```

```
t2,q2,ts2 = server.ReadTag("$ (N).$(D).SUBDEVICE2.TEG1");
```

```
server.WriteTag("$ (N).$(D).TEG1",t1+t2,192);
```

--скрипт подустройства **SUBDEVICE1**,

--читает тег **NODE1.DEVICE1.TEG1**

```
t1,q1,ts1 = server.ReadTag("$ (N).$(D).TEG1");
```

--скрипт тега **SUBDEVICE1.TEG1**,

--читает тег **NODE1.DEVICE1.TEG1**

```
t1,q1,ts1 = server.ReadTag("$ (N).$(D).TEG1");
```

server.ReadTagByNumber

Возвращает значение, качество и время тега с указанным номером.

Пример

```
TV,TQ,TS = server.ReadTagByNumber(1);
```

В скрипте тега возвращает значение текущего тега вне зависимости от номера.

server.ReadTagByRelativeName

Возвращает значение, качество и время тега по его относительному имени.

Пример

```
t, q, ts = server.ReadTagByRelativeName("NameTag");
```

При ошибке значение равно **nil**, качество равно 0, в **ts** записывается текст ошибки.

server.WriteCurrentTag

Записывает значение в текущий тег.

Пример

```
server.WriteCurrentTag(TV,TQ);
```

Возвращаемых значений нет.

server.WriteCurrentTagToDevice

Записывает значение и качество в текущий тег с обращением к устройству.

Пример

```
server.WriteCurrentTagToDevice(TV,TQ);
```

Возвращаемых значений нет.

server.WriteCurrentTagToHda

Записывает в архив HDA текущего тега заданные значение, качество и, опционально, метку времени (метка времени задается как 8-байтовый массив — см. *server.StringToTimeStamp* и *server.TimeToTimeStamp*). Для тега должен быть установлен флаг записи в архив HDA (HDA доступ = ИСТИНА).

Пример

Запись 120 с качеством OPC_QUALITY_GOOD и текущим временем ОС:

```
res=server.WriteCurrentTagToHda(120,192);
```

Запись 120 с качеством OPC_QUALITY_GOOD и заданной меткой времени:

```
ts = server.StringToTimeStamp("2010-05-03 12:34:45.100");
```

```
res= server.WriteCurrentTagToHda(120,192,ts);
```

Запись 120 с качеством OPC_QUALITY_COMM_FAILURE и заданной меткой времени:

```
timesec = os.time{year=2010,month=05,day=03,hour=12,min=34,sec=45};
```

```
timemsec=200;
```

```
ts = server.TimeToTimeStamp(timesec,timemsec);
```

```
server.WriteCurrentTagToHda(120,OPC_QUALITY_COMM_FAILURE,ts)
```

В случае успешного выполнения возвращает true, при ошибке – false.

server.WriteTag

Записывает значение и качество в тег с указанным полным именем.

Пример

```
server.WriteTag("Node.Device.Group.Tag",TV,TQ);
```

Возвращаемых значений нет.

В данной функции, как и в функции *server.ReadTag* , могут использоваться псевдоимена для текущего узла, устройства или подустройства.

server.WriteTagByNumber

Записывает значение и качество в тег с указанным номером.

Пример

```
server.WriteTagByNumber(1,TV,TQ);
```

Возвращаемых значений нет. В скрипте тега записывает значение в текущий тег.

server.WriteTagByNumberToDevice

Аналог *server.WriteTagByNumber* с обращением к устройству.

server.WriteTagByNumberToHda

Записывает в архив HDA тега с указанным номером заданные значение, качество и, опционально, метку времени (метка времени задается как 8-байтовый массив. Для тега должен быть установлен флаг записи в архив HDA.

Пример

Запись 5 с качеством OPC_QUALITY_BAD и текущим временем ОС в архив тега с номером 2:

```
res=server.WriteTagByNumberToHda(2, 5, OPC_QUALITY_BAD);
```

Запись 5 с качеством OPC_QUALITY_BAD и заданным временем в архив тега с номером 2:

```
ts = server.StringToTimeStamp("2010-05-03 12:34:45.100");  
res=server.WriteTagByNumberToHda(2, 5, OPC_QUALITY_BAD,ts);
```

В случае успешного выполнения возвращает true, при ошибке – false.

server.WriteTagByRelativeName

Записывает значение и качество в тег по его относительному имени.

Пример

```
res = server.WriteTagByRelativeName("NameTag", t, q);
```

В случае успешного выполнения возвращает true, при ошибке – false.

server.WriteTagByRelativeNameToDevice

Записывает значение и качество в тег по его относительному имени и инициирует запись в устройство.

Пример

```
res = server.WriteTagByRelativeNameToDevice("NameTag", t, q);
```

В случае успешного выполнения возвращает true, при ошибке – false.

server.WriteTagByRelativeNameToHda

Записывает в архив HDA тега с указанным относительным именем заданные значение, качество и, опционально, метку времени (метка времени задается как 8-байтовый массив). Для тега должен быть установлен флаг записи в архив HDA.

Примеры

Запись 5 с качеством OPC_QUALITY_BAD и текущим временем ОС в архив HDA тега Ter1 из скрипта узла Узел1, в который входит устройство Устройство1:

```
res=server.WriteTagByRelativeNameToHda("Устройство1.Ter1", 5,  
OPC_QUALITY_BAD);
```

Запись 5 с качеством OPC_QUALITY_BAD и заданным временем в архив HDA тега Ter1 из скрипта устройства Устройство1 или из скрипта тега устройства Устройство1:

```
ts = server.StringToTimeStamp("2010-05-03 12:34:45.100");  
res=server.WriteTagByRelativeNameToHda("Ter1", 5, OPC_QUALITY_BAD,ts);
```

В случае успешного выполнения возвращает true, при ошибке – false.

server.WriteTagToDevice

Записывает значение и качество в тег по его полному имени с обращением к устройству.

Пример

```
server.WriteTagToDevice("Node.Device.Group.Tag",TV,TQ);
```

Возвращаемых значений нет.

server.WriteTagToHda

Записывает в архив HDA тега с указанным полным именем заданные значение, качество и, опционально, метку времени (метка времени задается как 8-байтовый). Для тега должен быть установлен флаг записи в архив HDA.

Пример

Запись 5 с качеством OPC_QUALITY_BAD и текущим временем ОС:

```
res=server.WriteTagToHda("Узел1.Устройство1.Тег1", 5, OPC_QUALITY_BAD);
```

Запись 5 с качеством OPC_QUALITY_BAD и заданным временем:

```
ts = server.StringToTimeStamp("2010-05-03 12:34:45.100");
```

```
res=server.WriteTagToHda("Узел1.Устройство1.Тег1", 5, OPC_QUALITY_BAD,ts);
```

В случае успешного выполнения возвращает **true**, при ошибке – **false**.

server.UndoCurrentTag

Отменяет изменение значения текущего тега (восстанавливает предыдущее значение).

Пример

Если в теге задан скрипт, приведенный ниже, то после перевода сервера в режим исполнения и присвоения тегу значения 101 значение тега вновь станет равным 10, а качество – OPC_QUALITY_GOOD.

```
function OnInit()
```

```
    server.WriteCurrentTag(10, OPC_QUALITY_UNCERTAIN);
```

```
end
```

```
function OnClose()
```

```
end
```

```
function OnRead()
```

```
    if server.ReadCurrentTag( ) > 100 then
```

```
        server.UndoCurrentTag( );
```

<pre>end</pre> <pre>end</pre> Возвращаемых значений нет.

7. Доработайте скрипт синусоидального сигнала (см. Рис.1 и Рис.2) так чтобы генератор сигнала OPC сервера выдавал новый сигнал, например, пилообразный.
8. Проверьте работу генератора сигнала.
9. Используя функции представленные в таблице выше сохраните текущее значение тега в архиве HDA.
10. Убедитесь, что сохранение прошло успешно.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Для чего предназначен OPC сервер?
2. Как осуществляется мониторинг значений переменных сервера?
3. Какие типы данных поддерживает OPC сервер?
4. Как обеспечить преобразование переменной в OPC сервере?

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. InSAT. Программное обеспечение систем управления, диспетчеризации и АСУТП / OPC-СЕРВЕРЫ - OPC DA, OPC HDA. OPC-SERVER - ТЕХНОЛОГИЯ OPC / OPC СЕРВЕРЫ КОМПАНИИ ИНСАТ / ПРОТОКОЛ MODBUS RTU И TCP. MODBUS OPC СЕРВЕРЫ.
http://www.insat.ru/products/?category=399&_openstat=ZGlyZWN0LnIhbmRleC5ydTszNTc2Mzc2OzQzNzQ0MDk0O3lhbmRleC5ydTpwcmlVtaXVt
2. InSAT. Программное обеспечение систем управления, диспетчеризации и АСУТП / OPC-СЕРВЕРЫ - OPC DA, OPC HDA. OPC-SERVER - ТЕХНОЛОГИЯ OPC / OPC СЕРВЕРЫ КОМПАНИИ ИНСАТ / ПРОТОКОЛ MODBUS RTU И TCP. MODBUS OPC СЕРВЕРЫ.
<http://masteropc.ru/prices/info.php?pid=6944>
3. Dr. Bob Davidov. Компьютерные технологии управления в технических системах
<http://portalnp.ru/author/bobdavidov>