УДК 62519

А.Н. Рыбалев, Ф.А. Николаец

РАЗРАБОТКА И ЭМУЛИРОВАНИЕ АСУ ТП С ИСПОЛЬЗОВАНИЕМ ПРОГРАММ РАЗНЫХ ПРОИЗВОДИТЕЛЕЙ И ТИПОВ

В статье описывается подход к эмуляции автоматизированных систем управления технологическими процессами с использованием таких программ как Matlab, Trace Mode и CoDeSys. Основная цель заключается в замене элементов реальной системы (объекты, датчики, приводы, ПЛК) их виртуальными моделями и решение наиболее распространенные задач управления.

Ключевые слова: автоматизированная система управления технологическим процессом, система имитационного моделирования, программируемый логический контроллер, SCADA-система, стандарт OPC.

DEVELOPMENT AND EMULATE ACS USING THE PROGRAM FROM DIFFERENT MANUFACTURERS AND TYPES MANUFACTURERS

The article describes an approach to emulate automated process control systems with the use of programs such as Matlab, Trace Mode and CoDeSys. The main goal is to replace the real system elements (objects, sensors, actuators, PLCs) on their virtual models and solve the most common control tasks.

Key words: automated process control system, system simulation, PLC, SCADA-system, the OPC.

Введение

Разработка АСУ современных технологических процессов – сложная и ответственная задача, решение которой производится в несколько этапов: от составления математической модели до проектирования человеко-машинного интерфейса. Ошибки проектирования АСУ ТП очень трудно исправить на этапе эксплуатации системы – для этого может потребоваться даже пересмотр базовых концепций, лежащих в ее основе. С другой стороны, ошибки оперативного персонала АСУ могут привести к серьезным последствиям: остановке технологического процесса и авариям оборудования. В связи с этим как проектировщикам, так и оперативному персоналу нужен программный инструмент-симулятор АСУ ТП. Проектировщик с его помощью будет решать следующие задачи:

1) имитационное моделирование технологического процесса в различных режимах работы при воздействиях, программно формируемых управляющей аппаратурой и средствами человеко-машинного интерфейса;

2) отладка технологических программ;

3) выбор наиболее удобных для пользователя средств визуализации технологического процесса и способов формирования управляющих воздействий.

Оперативный персонал задействует программный комплекс на этапе настройки АСУ ТП, а также в целях обучения.

Кроме того, разрабатываемая система, безусловно, будет весьма полезна в учебном процессе по образовательным программам, предусматривающим изучение дисциплин, связанных с проектированием АСУ ТП.

В рамках единого комплекса предлагается задействовать программные средства разных производителей и классов:

система имитационного моделирования – для построения моделей технологического процесса;

система класса PC-based controller – для программной реализации алгоритмов управления на языках программирования промышленных контроллеров;

SCADA-система (supervisory control and data acquisition – система диспетчерского управления и сбора данных) – для визуализации технологических процессов и оперативного управления.

Перечисленные программные средства предназначены для исследования и разработки компонентов АСУ ТП, но используемые по отдельности, не могут решать перечисленные выше задачи.

Для построения прототипа было решено применять следующие программные продукты:

MathWorks® MATLAB®, Simulink® (среда имитационного моделирования);

3S-Smart Software® CODESYS® (PC-эмулятор ПЛК SP PLCWinN, OPC-сервер);

AdAstra Research Group® TRACE MODE® (SCADA-система).

Выбор данных программ обусловлен опытом их применения в учебном процессе.

Межпрограммный обмен

В настоящее время основным стандартом межпрограммного обмена данными в сфере промышленной автоматизации, безусловно, является OPC (OLE for Process Control). OPC – набор повсеместно принятых спецификаций, предоставляющих универсальный механизм обмена данными в системах контроля и управления. OPC-технология обеспечивает независимость потребителей от наличия или отсутствия драйверов или протоколов, что позволяет выбирать оборудование и программное обеспечение, наиболее полно отвечающие реальным потребностям приложения.

OPC-сервер – программа, получающая данные во внутреннем формате устройства или системы и преобразующая эти данные в формат OPC. OPC-сервер является источником данных для OPC-клиентов. По своей сути OPC-сервер – это некий универсальный драйвер физического оборудования, обеспечивающий взаимодействие с любым OPC-клиентом.

В общем случае OPC-сервер может быть запущен как компонент любой из трех программ (имитационного моделирования, контроллера или SCADA-системы) или быть внешним по отношению к ним. В системе может быть задействовано и более одного сервера. Каждый из вариантов имеет свои преимущества и недостатки.

В прототипе используется OPC-сервер CoDeSys, связанный с контроллером CoDeSys SP PLCWinNT через «общий» шлюз типа TCP/IP. Список переменных для обмена формируется в контроллере. Matlab и Trace Mode являются OPC-клиентами (рис.1).

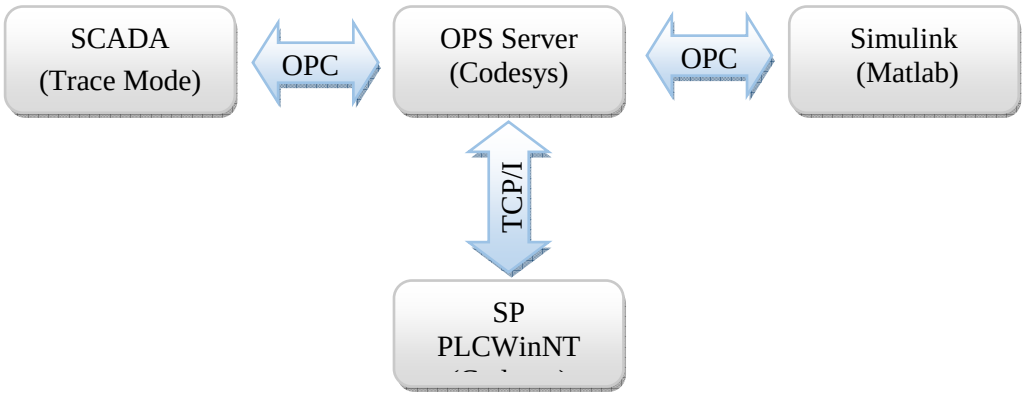

Рис. 1. Взаимодействие программ.

Выбор такой конфигурации связан исключительно с простой ее настройки.

Система имитационного моделирования

Одним из важных этапов проектирования АСУ ТП является создание математической модели объекта управления. Для разработки имитационных моделей объектов в работе был использован пакет Simulink, интегрированный в MATLAB [1]. Simulink – это графическая среда имитационного моделирования, позволяющая строить динамические модели, включая дискретные, непрерывные и гибридные, нелинейные и разрывные системы, при помощи блок-диаграмм в виде направленных графов. На рис. 2 показана Simulink-модель простейшего теплового объекта, описываемого передаточной функцией первого порядка с запаздыванием. Блоки OPC Configuration, OPC Read и OPC Write обеспечивают обмен данными с OPC-сервером CoDeSys.

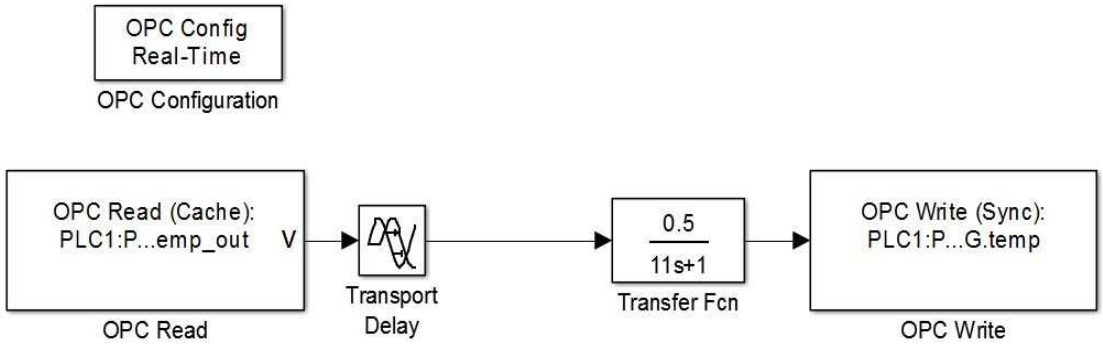

Рис. 2. Модель процесса.

Установка и настройка OPC-клиента в Matlab могут быть выполнены с помощью специальных функций пакета OPC Toolbox. Ниже показан пример использования некоторых функций (с ответными сообщениями Matlab).

```
>>opcreset;
>> opcregister('install')

Continuing this operation will modify any OPC Foundation files already installed.
Type 'Yes' (exactly as shown) to install the OPC Foundation files

Confirmation string: Yes
>> hostInfo = opcserverinfo('localhost')
hostInfo =
              Host: 'localhost'
          ServerID: {'CoDeSys.OPC.02'}
    ServerDescription: {'OPC Server for CoDeSys V2.0'}
     OPCSpecification: {'DA2'}
    ObjectConstructor: {'opcda('localhost', 'CoDeSys.OPC.02')'}
>> allServers = hostInfo.ServerID'
allServers =
    'CoDeSys.OPC.02'
>> da = opcda('localhost', 'CoDeSys.OPC.02')
da =
Summary of OPC Data Access Client Object: localhost/CoDeSys.OPC.02
```

```
   Server Parameters
      Host      : localhost
      ServerID  : CoDeSys.OPC.02
      Status    : disconnected
      Timeout   : 10 seconds
   Object Parameters
Group     : 0-by-1 dagroup object
      Event Log : 0 of 1000 events
>>opctool
```

Для тех же целей можно использовать графическую утилиту OPCTool (рис. 3).

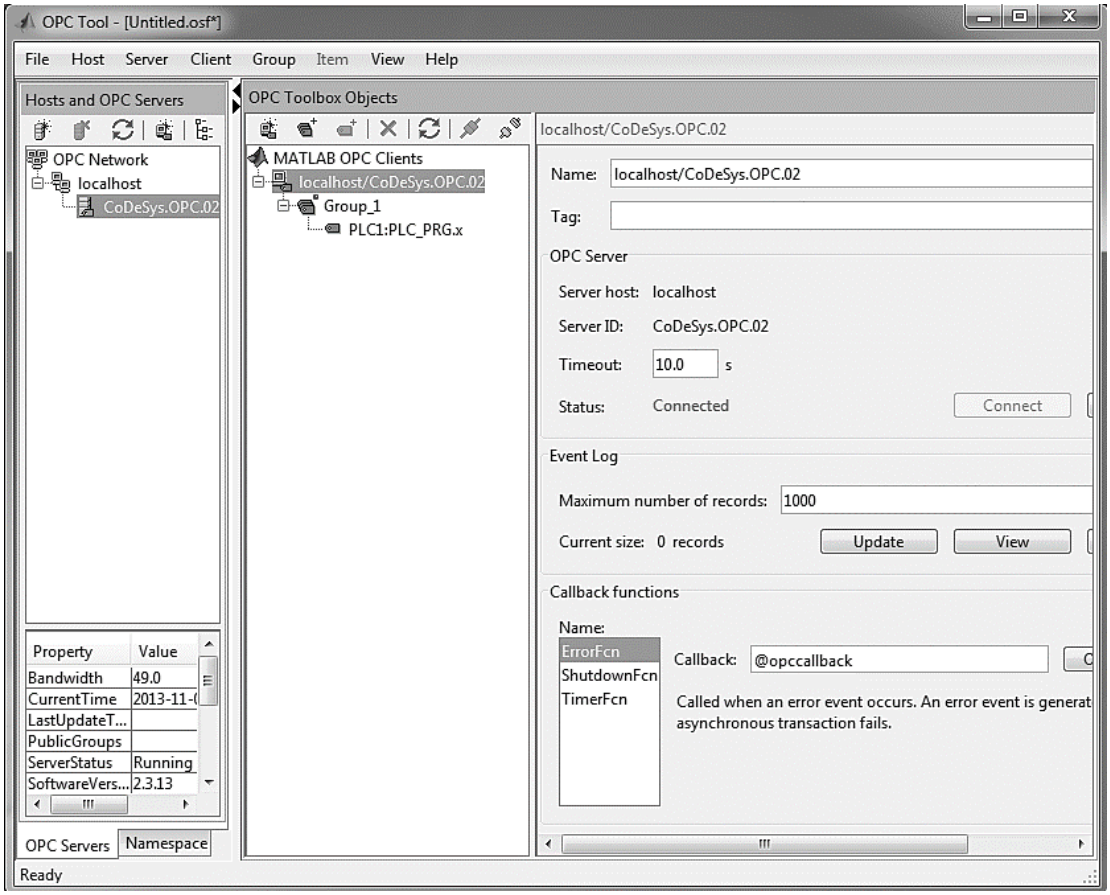

Рис.3. OPCTool.

Программно-эмулируемый контроллер (PC-based Controller)

SP PLCWinNT (рис. 4) – виртуальный контроллер, работающий под управлением ОС Windows. SP PLCWinNT входит в состав программы CoDeSys [2].

Программа для SP PLCWinNT создается и отлаживается в редакторах CoDeSys «обычным» способом (т.е. так, как это делается для реальных контроллеров). При этом доступны все пять определяемых стандартом IEC 61131-3 (МЭК 61131-3) языков программирования:

IL (Instruction List) – ассемблер-подобный язык;

ST (Structured Text) – Pascal-подобный язык;

LD (Ladder Diagram) – язык релейных схем;

FBD (Function Block Diagram) – язык функциональных блоков;

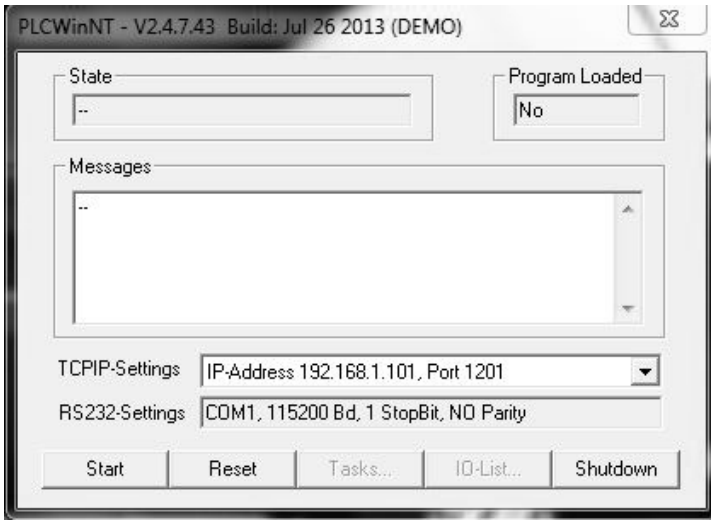

Рис. 4. Виртуальный контроллер CoDeSys SP PLCWinNT.

SFC (Sequential Function Chart) – язык диаграмм состояний.

В дополнение к FBD поддерживается язык программирования CFC (Continuous Function Chart) с произвольным размещением блоков и расстановкой порядка их выполнения.

В приложении приведен листинг программы контроллера, составленной на языке ST и решающей задачу регулирования по ПИД-закону выходной величины виртуального объекта (см. рис. 2).

В отличие от реальных контроллеров SP PLCWinNT не имеет собственных входов и выходов, поэтому обмен данными с внешними устройствами возможен только посредством периферии компьютера, на котором он запущен. Это могут быть как последовательные (COM), так и сетевые (Ethernet) интерфейсы. При наличии специальных плат допускается подключение к развитым промышленным сетям, – например, CAN.

Настройка взаимодействия контроллера с OPC-сервером выполнена в несколько этапов.

1. Создан локальный сервер и настроены параметры связи для передачи данных между программами (рис. 5).

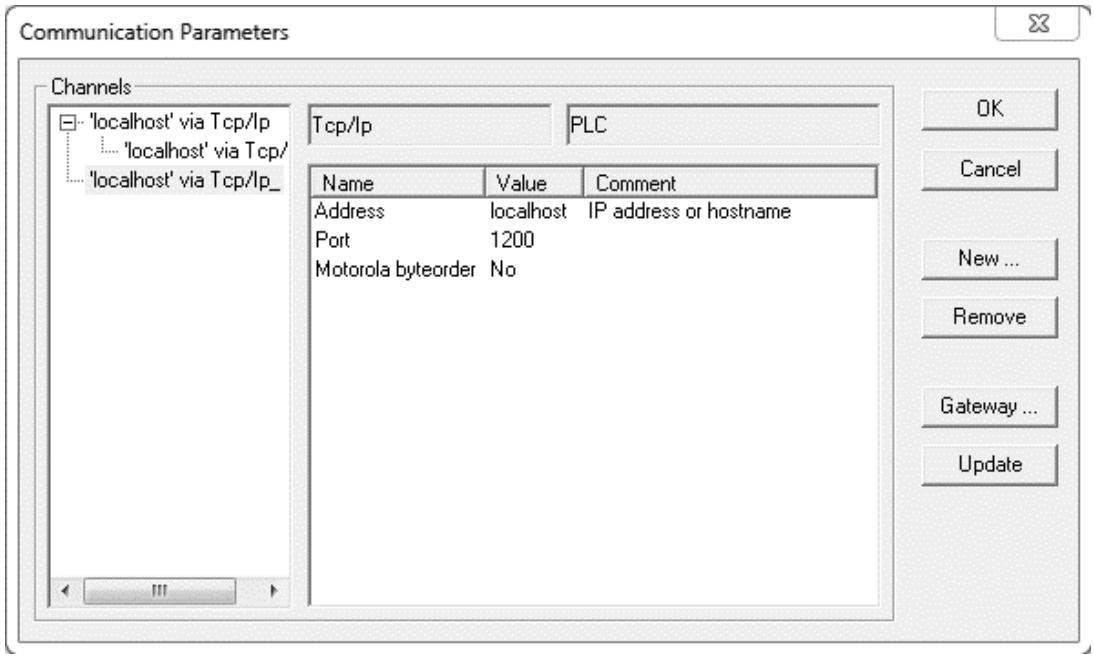

Рис. 5. Параметры связи.

2. В опциях проекта CoDeSys выделены переменные, которые должны быть доступны OPC-серверу (рис 6).

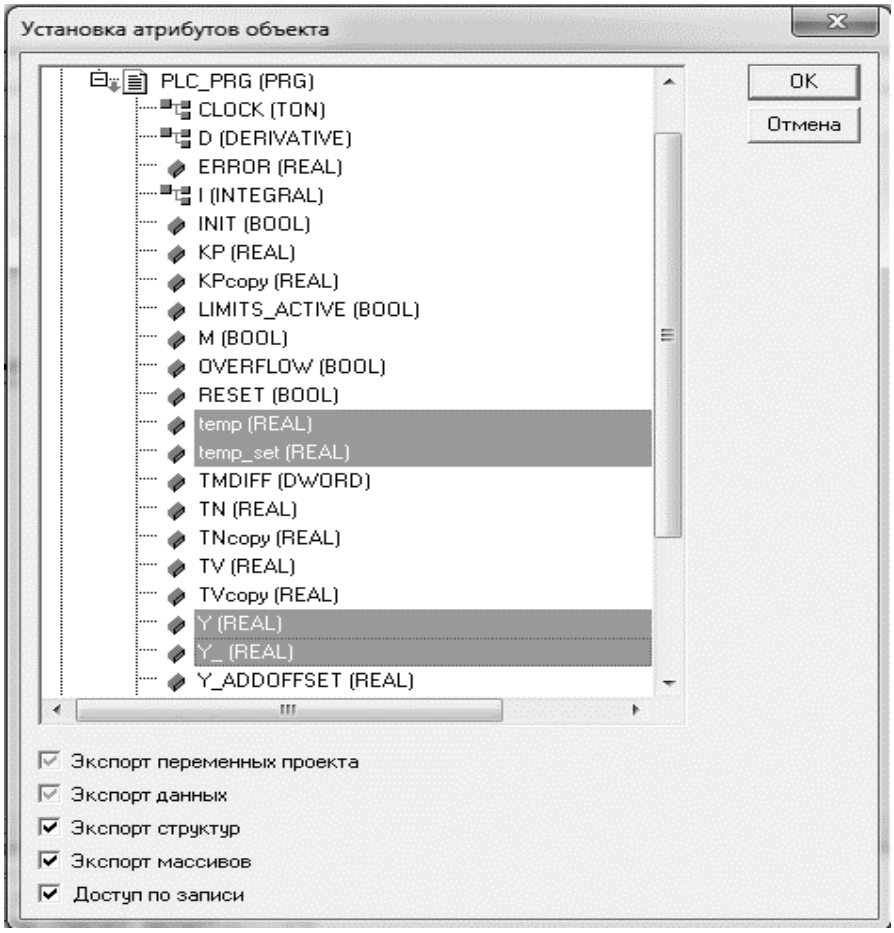

Рис. 6. Установка атрибутов объекта.

3. С помощью утилиты OPCConfig запущен и настроен OPC-сервер (рис. 7).

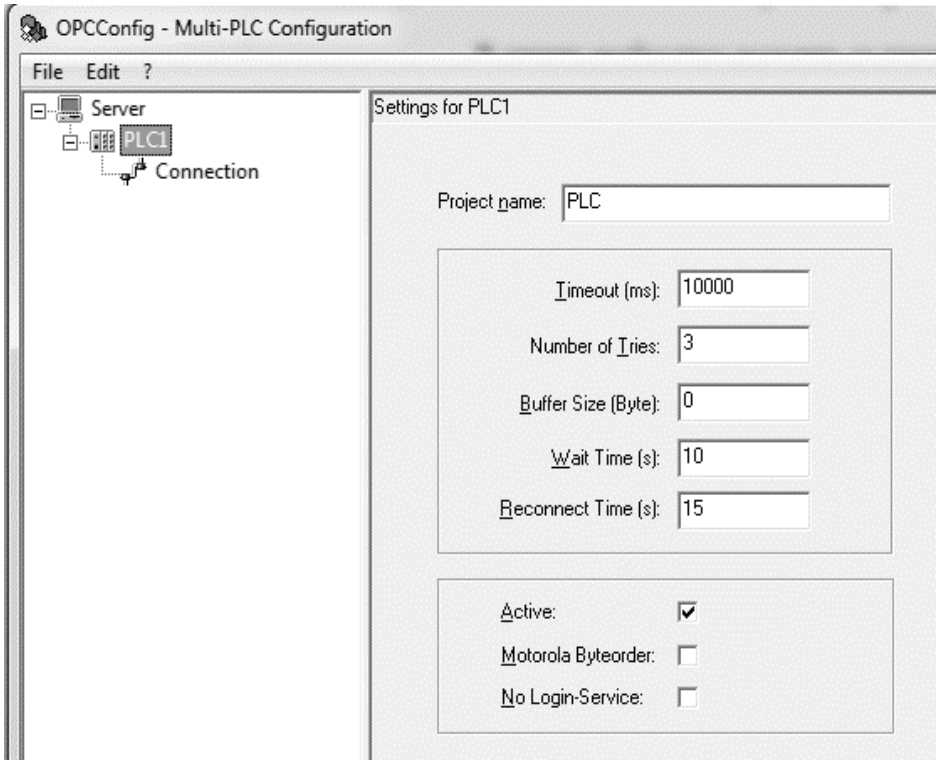

Рис. 7. OPCConfig.

SCADA-система

SCADA – программный пакет, предназначенный для разработки и обеспечения работы в реальном времени систем сбора, обработки, отображения и архивирования информации об объекте мониторинга или управления. SCADA-системы используются во всех отраслях хозяйства, где требуется операторский контроль за технологическими процессами в реальном времени. В качестве SCADA-системы в проекте применялся пакет Trace Mode 6 [3].

Trace Mode 6 включает инструментальную систему (среду разработки) и средства для работы в реальном времени. Инструментальная система Trace Mode 6 – это универсальное средство разработки и отладки приложений для автоматизированных систем управления технологическими процессами (АСУ ТП) и управления производством (АСУ П). В состав системы входит отладочный монитор реального времени – профайлер, позволяющий запускать все экраны проекта на компьютере разработчика.

Как и все современные SCADA-системы, Trace Mode 6 легко настраивается в качестве OPC-клиента (рис. 8).

В графическом редакторе Trace Mode 6 создан простейший экран визуализации (рис. 9), позволяющий наблюдать изменение регулируемого параметра, формировать задание регулятору в автоматическом режиме и сигнал управления объектом – в ручном.

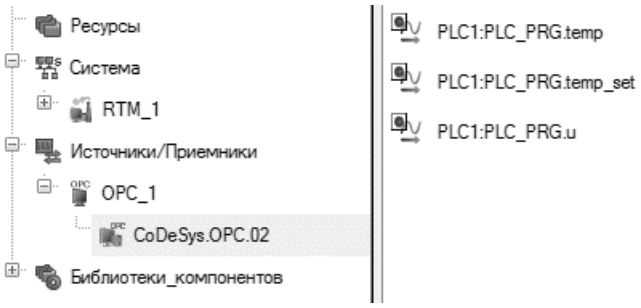

Рис. 8. Настройка OPC-клиента в Trace Mode 6.

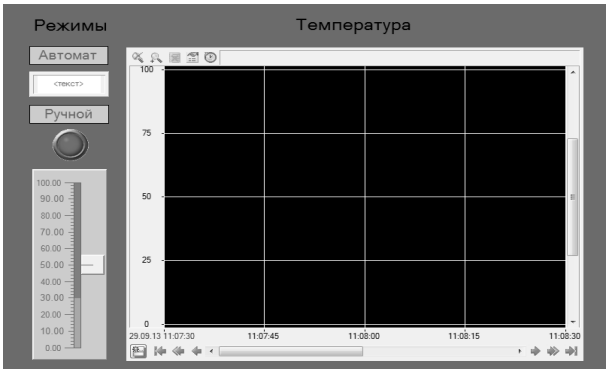

Рис. 9. Интерфейс управления ТП.

Заключение

Разработан прототип программного симулятора АСУ ТП, демонстрирующий параллельную работу и взаимодействие программных средств имитации, управления и визуализации. Сферы применения рассмотренной технологии:

1. Разработка прототипов АСУ ТП промышленных предприятий, включая построение моделей процессов, разработку технологических программ и человеко-машинного интерфейса. Благодаря тому, что большинство современных ПЛК программируется на языках МЭК 61131-3, программы, составленные для PLCWinNT, после минимальных изменений и «привязки» к процессу можно загружать в реальные контроллеры.

2. Разработка программных симуляторов уже имеющихся АСУ ТП для обучения оперативного персонала. Вследствие универсальности стандарта OPC Trace Mode можно заменить той SCADA-системой, что используется на практике.

3. Учебные цели и проекты по направлениям подготовки, связанным с автоматизацией производства, в том числе:

самостоятельная подготовка студентов к выполнению работ на лабораторном оборудовании, предполагающих программирование контроллеров и создание человеко-машинного интерфейса. При

этом основная часть работы, связанная с проектированием, будет проделана студентом внеаудиторно, а реализация системы на реальном оборудовании не займет много времени. Это позволит увеличить количество выполняемых лабораторных работ без увеличения объема аудиторных занятий;

разработка комплексных заданий для государственного экзамена по направлению подготовки. Такие задания можно сформулировать так, чтобы они охватывали материал практически всех изученных выпускником специальных дисциплин. Исходными данными могли бы быть, например, текстовое описание технологического процесса, модели объектов регулирования типа «черный ящик» и требования к проектируемой системе. В ходе выполнения задания выбираются технические средства автоматизации, составляются функциональная и принципиальная схемы, производятся необходимые расчеты и, как итог, создается программный симулятор по описанной выше технологии.

---

1. Ануфриев, И.А. MATLAB 7. Наиболее полное руководство / И.А. Ануфриев, А.Б. Смирнов, Е.Н. Смирнова. – СПб.: БХВ-Петербург, 2005. – 1104 с.

2. Руководство пользователя по программированию ПЛК в CoDeSys 2.3 [Электронный ресурс]. –2006. – 158 с. Режим доступа: http://www.kipshop.ru/CoDeSys/steps/codesys_v23_ru.pdf, свободный. – Загл. с экрана.

3. Быстрый старт Tracemode 6 [Электронный ресурс] / AdAstraResearchGroup, Ltd. – М., 2008. – 517 с. Режим доступа: http://www.adastra.ru/files/documents/QUICK_START_v_6a_d.pdf, свободный. – Загл. с экрана.

В качестве приложения приводим листинг программы контроллера:

```
PROGRAM PLC_PRG
VAR_INPUT
        temp :REAL;
        temp_set:REAL;
        KP:REAL;
        TN:REAL;
        TV:REAL;
        Y_:REAL;
        Y_OFFSET:REAL;
        Y_MIN:REAL;
        Y_MAX:REAL;
        M:BOOL;
        RESET:BOOL;
END_VAR
VAR_OUTPUT
        Y:REAL;
        LIMITS_ACTIVE:BOOL:=FALSE;
        OVERFLOW:BOOL:=FALSE;
END_VAR
VAR
        CLOCK:TON;
        I: INTEGRAL;
        D: DERIVATIVE;
        TMDIFF: DWORD;
        ERROR: REAL;
        INIT: BOOL:=TRUE;
        Y_ADDOFFSET: REAL;
```

```
        KPcopy:REAL;
        TNcopy:REAL;
        TVcopy:REAL;
END_VAR

KP:=1;
TN:=10;
Y_MAX:=100;
IF TN>0 AND KP<> 0 AND (NOT OVERFLOW OR RESET OR M) THEN
        ERROR :=temp_set-temp;
 IF RESET OR M OR INIT OR (KP<>KPcopy OR TN<>TNcopy OR TV<>TVcopy)
 THEN
        I(RESET:=TRUE);
        D(RESET:=TRUE);
        OVERFLOW:=FALSE;
        LIMITS_ACTIVE:=FALSE;
        IF RESET OR INIT THEN
          Y := Y_OFFSET;
          INIT:=FALSE;
          Y_ADDOFFSET := 0;
        ELSIF M THEN
          Y := Y_;
          Y_ADDOFFSET := Y_- (Y_OFFSET+KP*(ERROR+I.OUT//TN+D.OUT*TV));
        ELSE
          Y_ADDOFFSET := Y - Y_OFFSET - KP*ERROR;
        END_IF
        TMDIFF:=0;
        CLOCK(IN:=FALSE);
        CLOCK(PT:=t#1h, IN:=TRUE);
        TNcopy := TN;
        TVcopy := TV;
        KPcopy := KP;
 ELSE
        CLOCK;
        TMDIFF:=TIME_TO_DWORD(CLOCK.ET);
 END_IF;
 IF TMDIFF>0 THEN
        CLOCK(IN:=FALSE);
        CLOCK(PT:=t#1h, IN:=TRUE);
        D(IN:=ERROR, TM:=TMDIFF, RESET:=FALSE);
        I(IN:=ERROR, TM:=TMDIFF, RESET:=FALSE);
        OVERFLOW := I.OVERFLOW;
        IF NOT OVERFLOW THEN
                Y:=Y_OFFSET+KP*(ERROR+I.OUT/TN+D.OUT*TV) + Y_ADDOFFSET;
                IF Y>1E30 OR Y<-1E30 THEN
                        OVERFLOW:=TRUE;
```

```
                END_IF;
                LIMITS_ACTIVE:=FALSE;
                IF Y_MAX>Y_MIN AND Y>Y_MAX THEN
                        LIMITS_ACTIVE:=TRUE;
                        IF KP<>0 THEN
                          I(IN:=(Y_MAX-Y)*TN/KP,TM:=1000,RESET:=FALSE);
                        END_IF
                        Y:=Y_MAX;
                END_IF;
                IF Y_MAX>Y_MIN AND Y<Y_MIN THEN
                        LIMITS_ACTIVE:=TRUE;
                        IF KP<>0 THEN
                          I(IN:=(Y_MIN-Y)*TN/KP,TM:=1000,RESET:=FALSE);
                        END_IF
                        Y:=Y_MIN;
                END_IF;
        END_IF;
  ELSE
        CLOCK(PT:=t#1h,IN:=TRUE);
  END_IF;

END_IF;
```