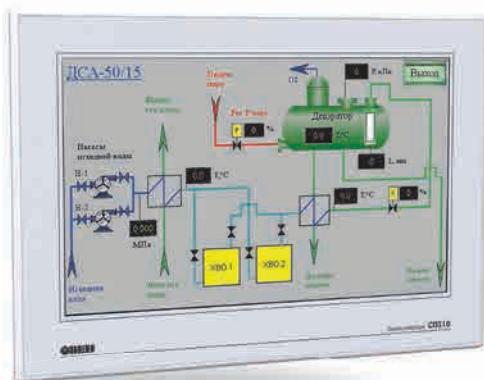


Использование макросов при работе с операторскими панелями ОВЕН СП3xx

Евгений Кислов, инженер ОВЕН

Операторские панели ОВЕН СП3xx применяются для создания человеко-машинных интерфейсов объектов автоматизации. Основной функцией панели в большинстве применений является визуализация и передача данных, полученных от других устройств. Кроме этого, с помощью панели можно создавать алгоритмы управления и обработки данных. Для их реализации необходимо использовать макросы.



Для реализации пользовательских алгоритмов на панели ОВЕН СП3xx используются макросы. Макросы представляют собой небольшие программы, написанные на языке ANSI C (C89). Каждый проект по умолчанию содержит глобальный макрос (рис. 1), который предназначен для объявления глобаль-

ных переменных и функций, доступных в пользовательских макросах.

Для создания пользовательского макроса в дереве проекта выбирается узел **Макросы**, в контекстном меню – команда **Добавить макрос**. Другая команда этого меню – **Компиляция** – используется для проверки корректности синтаксиса всех макросов проекта.

Простейший макрос

На рис. 2 приведен пример простейшего макроса, который суммирует значения регистров PSW300-303 и записывает результат в регистр PSW400. В левой части редактора макросов отображается список системных функций. Их полное описание приведено в Руководстве пользователя и справке программы Конфигуратор СП300.

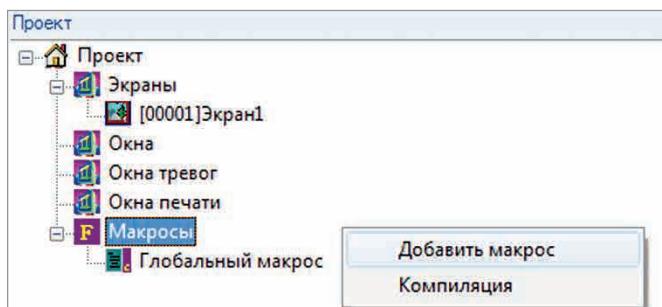


Рис. 1

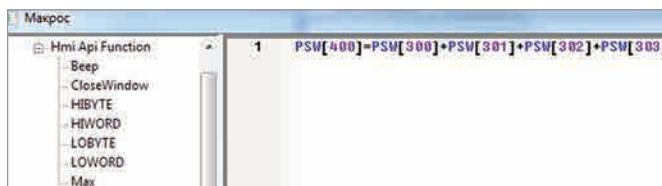


Рис. 2

Вызов макросов

Код, содержащийся в пользовательских макросах, выполняется при их вызове. Вызов осуществляется с помощью элементов **Функциональная кнопка** и **Функциональная область**.

Первый элемент, как правило, применяется для однократного вызова макросов, второй – для циклического (рис. 3). Следует отметить, что при вызове макроса все его переменные инициализируются начальными значениями, поэтому при необходимости сохранения результатов вычислений следует использовать память панели.

Работа с внутренней памятью

Панели СП3xx имеют три независимые области памяти – PSB (биты), PSW (оперативные регистры), PFW (энергонезависимые регистры). Методика

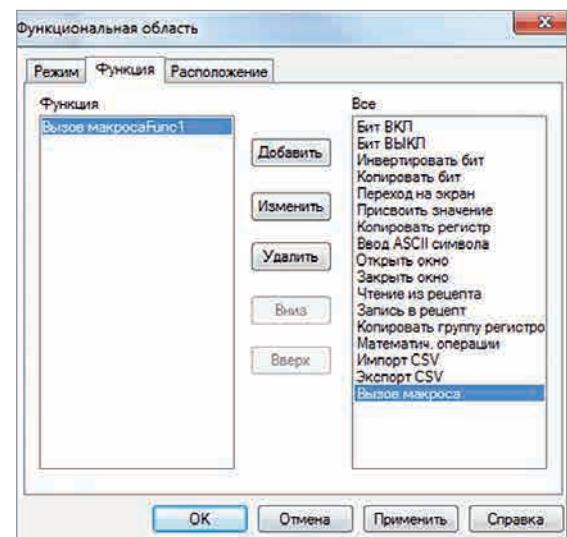


Рис. 3

```
// Код пользовательского макроса
if (GetPSBStatus(300))
{
    SetPSB(400);
}
else
{
    ResetPSB(400);
}
```

Рис. 4

работы с каждой областью памяти в макросах различается.

- » Для работы с **PSB** используются функции: GetPSBStatus – чтение бита; SetPSB – установка бита в TRUE; ResetPSB – установка бита в FALSE. В качестве примера представлен макрос, который записывает значение бита PSB300 в PSB400 (рис. 4).
- » Работа с **PSW** производится с помощью оператора присваивания `==>`: **PSW[400]=PSW[300];**
- » Работа с **PFW** осуществляется с помощью функций **Read/Reads** и **Write/Writes**. На рис. 5 приведен код макроса, который считывает значение PFW300 в локальную переменную **wVar** и записывает ее значение в PFW400. С помощью функций **Reads/Writes** можно считывать и записывать массивы данных.

Пример использования макросов – работа с СОМ-портом

Для записи в регистр битовой маски выходов модулей ОВЕН МУ110 и МК110 нужно применять функцию Modbus 0x10 (Write Multiple Registers). Однако при использовании графических элементов панели запись одного регистра всегда производится функцией 0x06 (Write Single Register), поэтому при работе с модулями МУ/МК обмен реализуется через макросы.

В качестве примера рассмотрим работу с модулем ОВЕН МК110. Объявим в глобальном макросе функцию **MK_MASK_WRITE**. Ее аргументами являются адрес модуля, адрес записываемого регистра, значение и номер бита, в который записывается статус обмена.

Код функции показан на рис. 6.

Функция **MK_MASK_WRITE** выполняет три действия:

- » формирует запрос к модулю по протоколу Modbus;
- » отправляет этот запрос в PLC-порт, считывает ответ и освобождает порт;

```
// Код пользовательского макроса
WORD wVar;

Read (HMI_LOCAL_MCH, 0, TYPE_PFW, 300, 0, TYPE_WORD, &wVar);
Write (HMI_LOCAL_MCH, 0, TYPE_PFW, 400, 0, TYPE_WORD, wVar );
```

Рис. 5

```
// Код глобального макроса
void MK_MASK_WRITE(BYTE DevAdr, WORD RegAdr, WORD Mask, WORD Error)
{
    BYTE DataReceive[8];
    BYTE DataSend[11];

    DataSend[0]=DevAdr;
    DataSend[1]=0x10;
    DataSend[2]=HIBYTE(RegAdr);
    DataSend[3]=LOBYTE(RegAdr);
    DataSend[4]=0x00;
    DataSend[5]=0x01;
    DataSend[6]=0x02;
    DataSend[7]=HIBYTE(Mask);
    DataSend[8]=LOBYTE(Mask);
    DataSend[9]=HIBYTE(Crc(DataSend,9));
    DataSend[10]=LOBYTE(Crc(DataSend,9));

    Enter(PLC);
    Send(PLC,DataSend,11);
    Receive(PLC,DataReceive,8,1000,6);
    Leave(PLC);
    Delay(100);

    if (DataReceive[1]!=0x10)
    {
        SetPSB(Error);
    }
    else
    {
        ResetPSB(Error);
    }
    return;
}
```

Рис. 6

```
// Код пользовательского макроса
WORD UserMask=PSW[300];

MK_MASK_WRITE(70, 50, UserMask, 400);
```

Рис. 7

- » выставляет флаг ошибки в заданный бит при некорректном ответе.

Функцию **MK_MASK_WRITE** следует вызывать в пользовательском макросе (рис. 7). В результате выполнения данного макроса в модуль МК110 с адресом 70 в регистр 50 (регистр битовой маски выходов) будет записано значение регистра панели PSW300. В случае ошибки биту PSB400 присваивается значение TRUE.

Обращаем внимание, что при работе с портом нет необходимости задавать его параметры, так как они настраиваются в Конфигураторе СП300.

Использование порта в макросе не влияет на опрос через графические элементы.

Заключение

Макросы представляют собой удобный инструмент для решения нестандартных задач. Использование макросов значительно расширяет возможности панелей СП3xx. Подробное описание макросов приведено в Руководстве пользователя и справке программы Конфигуратор СП300. Пример работы с модулями МУ/МК доступен для скачивания на сайте owen.ru.