Руководство по эксплуатации
панелей оператора
серии

DOP-B

Перевод и адаптация ООО "НПО "СТОИК ЛТД"
www.stoikltd.ru

v.20100531

Содержание

Глава 1. Вступление

1.1 Панели оператора серии DOP-B

Панели оператора серии DOP-B выполнены на основе высокопроизводительного микропроцессора. Они просты в использовании и за счет дружественного программного обеспечения являются совершенным и удобным в применении программируемым интерфейсом.

Для программирования панелей оператора серии DOP-B имеется ряд программ, призванных помочь пользователю настроить панель оператора под конкретые применения:

1. Средство разработки/редактирования программ **Screen Editor v2.xx**. Позволяет создавать многофункциональный и дружественный человеко-машинный интерфейс.
2. DOP Soft Новое программное обеспечение, разработанное специально для работы с панелями серии DOP-B - **DOPSoft v1.xx.** Мощный инструмент для разработки/редактирования программ. Может редактировать программы, созданные в **Screen Editor** (.dpb и .dop).
3. DOP eServer Программное обеспечение для сбора данных. Это удобное решение для постоения информационной сети между офисом и производством. Программа предназначена для непосредственного получения архива данных от панели оператора по сети Ethernet и отображения их в табличной форме на ПК.
4. DOP eRemote Программное обеспечение для управления и мониторинга. Программа способна отображать на экране ПК ту же самую информацию, что в текущий момент отображается на экранах панелей DOP-B. Становится возможным осуществление контроля и управления производственным процессом дистанционно по сети Ethernet с офисного ПК.
5. Программа **WPLSoft v2.3xx**, изначально созданная для разработки/ редактирования программ ПЛК серии DVP, поможет в реализации функци

Direct Link в панелях оператора. (Подробнее см. Приложение E)

Все перечисленные программы работают в среде **Windows** (поддерживают Windows 2000, Windows XP 32bit/64bit, Windows Vista 32bit/64bit, Windows 7 32bit/64bit)

Большинство действий и функций, описанных в данном руководстве описаны в среде **Screen Editor**, эти действия полностью применимы в программе **DOPSoft.**

1.2 Характеристики панелей оператора серии DOP-B

■ Поддержка подключения различных типов контроллеров

Панели оператора серии DOP-B поддерживают работу с внешними контроллерами более чем 20 брэндов, включая Delta, Omron, Siemens, Mitsubishi и т.д.

Полную информацию о поддерживаемых моделях контроллеров можно получить на сайте компании Delta Electronics:

http://www.delta.com.tw/product/em/download/download_main.asp?act=3&pid=3&cid=2&tpid=8

■ Возможность использования различных шрифтов

Screen Editor использует все шрифты, поддерживаемые Windows®.

■ Применение макрокоманд

Screen Editor дает возможность создавать программы для управления процессом сложных вычислений с помощью макросов. Дополнительно имеется возможность, используя коммуникационные макрокоманды связываться с внешними контроллерами через последовательный COM порт.

■ Быстрая загрузка/выгрузка программ по USB

Кроме RS-232, пользователь имеет возможность загружать и выгружать программы через USB порт, что ускоряет эту процедуру.

■ Рецепты

Screen Editor обладает удобной функцией редактирования и хранения рецептов (аналогично Microsoft Excel) с возможностью загрузки выбранного

отредактированного рецепта в контроллер.

- **Непосредственная коммуникация одновременно с тремя контроллерами через три имеющихся последовательных порта**

- **Поддержка одновременного подключения нескольких контроллеров**

  Панели оператора серии DOP-B способны управлять сетью из нескольких контроллеров через порты COM2 и COM3 по интерфейсу RS-485.

- **Режимы симуляции**

  Программа Screen Editor позволяет в режиме симуляции проводить проверку разработанной программы, перед загрузкой её в память панели оператора. Для нормальной работы неоходимо установить разрешение монитора не менее 24 бит.

  Off-line симуляция: После завершения редактирования программы панели пользователь может проверить правильность работы программы с помощью компьютера не подключая контроллер.

  On-line симуляция: После завершения редактирования программы панели пользователь может проверить правильность работы программы на подключенном контроллере с помощью симуляции работы панели на компьютере.

- **USB Host порт**

  Панели оператора серии DOP-B имеют USB для подключения USB диска, картридера и принтера с USB портом. Это даёт возможность пользователю сохранять данные, копировать программы и немедленно выводить на печать требуемую информацию. Кроме того, увеличение объёма памяти позволяет архивировать больший объём данных.

- **Функция печати**

  Панели оператора серии DOP-B поддерживают принтеры с COM и USB портами.

- **Многоуровневая защита паролем программы и экранных элементов.**

Панели оператора серии DOP-B имеют многоуровневую систему паролей обеспечивающие как защиту интеллектуальной собственности (программ) от копирования, так и ограничения несанкционированного доступа к экранным элементам управления для неквалифицированного персонала.

■ Многоязыковая поддержка

Можно настроить интерфейс на любой из 16 языков с возможностью переключения языков без установки многоязычных операционных систем.

Глава 2.
Работа с внешними устройствами

2.1 Применение SD Card и USB Disk

■ Карта памяти SD

SD card (поддерживается SDHC) могут применяться для сохранения и переноса данных. Поддерживаемый формат файлов FAT32. Перед использованием карты памяти SD, её необходимо отформатировать под FAT32 с помощью панели оператора. Только карта памяти SD, отформатированная таким образом, может одновременно использоваться и в панели оператора и в системах с Windows® OS. (Даже если возможны чтение/запись в некоторых других форматах, может возникнуть ошибка в Win95/98/2000/XP.

■ USB носитель

USB диск может использоваться для хранения данных. Он также применяется для копирования данных с панели оператора и его формат, соответственно, должен быть FAT32. При использовании его для сохранения данных, необходимо выбрать USB диск объёмом не более 2GB. Чтобы гарантировать безошибочное сохранение данных на USB диске, необходимо сначала войти в системное меню, и только затем отсоединить USB диск.

Есть два способа безопасного отсоединения USB диска:

1. Войти в системное меню, нажав кнопку SYS в течении 3 секунд, и выбрать функцию отключения USB диска.

2. Создайте экранный кнопочный элемент “Извлечение диска”. После завершения настроек и компиляции этого элемента, он позволит одним нажатием безопасно извлекать USB диск.

Глава 3. Создание и редактирование экранных элементов

Отдел инжиниринга ООО «НПО СТОИК» осуществляет программирование контроллеров DVP и панелей оператора DOP и TP по техническому заданию заказчиков, а также оказывает помощь в выборе оптимального набора оборудования под требования задачи и проектирует комплексные системы управления. При необходимости система управления может быть поставлена в виде готового шкафа, станции или щита управления. Более подробную информацию см. http://www.deltronics.ru/support/engineering/

3.3 Внутренняя память

Возможно использовать 6 различных регистров внутренней памяти (Рис. 3-3-1):

1. внутренние регистры $
2. энергонезависимые внутренние регистры $M
3. регистры косвенной адресации *$
4. регистры номера рецепта RCPNO
5. регистры группы рецептов RCPG
6. регистры рецептов RCP.

Для детального рассмотрения внимательно ознакомьтесь с нижеприведённым описанием.

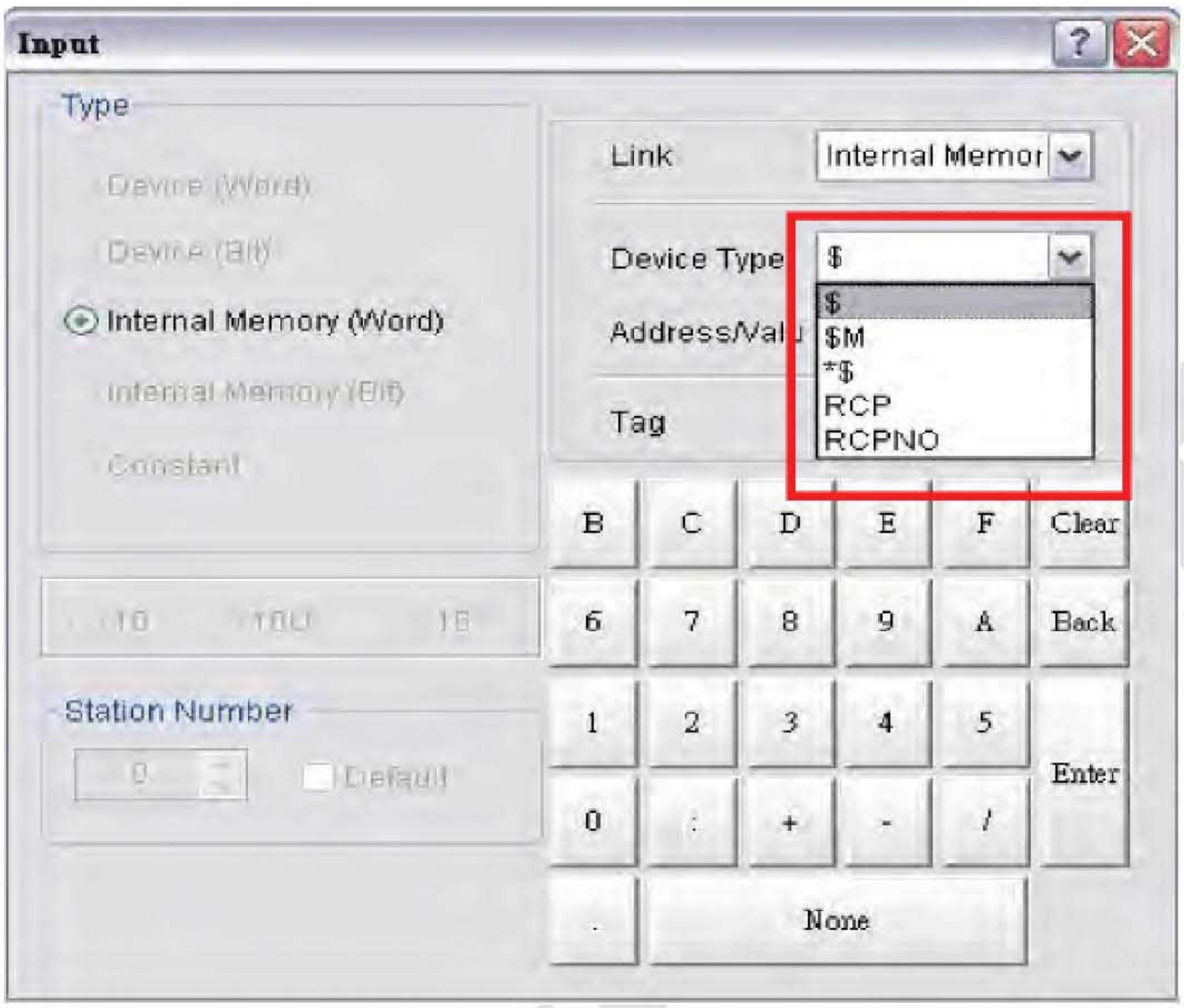


Рис. 3-3-1 Внутренняя память панели

3.3.1 Внутренние регистры (R/W) : $

Обращение к словам: $n (n : 0-65535)

Обращение к битам: $n.b (n : 0-65535, b : 0-15)

Панели оператора серии DOP-B предоставляют в пользование 65536 16-битных внутренних регистров ($0.0 - $65535.15).

При выключении питания данные не сохраняются.

3.3.2 Энергонезависимые внутренние регистры (R/W): $M

Обращение к словам: $Mn (n : 0-1023)

Обращение к битам: $Mn.b (n : 0-1023, b : 0-15)

Панели оператора серии DOP-B предоставляет в пользование 1024 16-битных энергонезависимых внутренних регистров ($M0.0 - $M1023.15).

Все записанные данные этих регистров после выключения питания сохраняются; такие регистры используются для хранения важных данных или их записи.

3.3.3 Регистры косвенной адресации (R/W): *$

Обращение к словам: *$n (n: 0~65535)

Регистр косвенной адресации – это регистр, хранящий значение из адреса одноименного внутреннего регистра. Пользователю необходимо извлечь адрес из $n и затем извлечь значение, сохраненное в этом адресе.

Например: если $7 = 20; $20 = 39; то *$7 = 39.

Общая формула : если $n = m; $m = x, то *$n = x (m : 0-65535). (значение m не может превышать 65535)

При выключении питания данные не сохраняются

3.3.4 Регистр номера рецепта (R/W): RCPNO

Этот регистр используется для назначения номера группе параметров (Рис. 3-3-2)

При загрузке/выгрузке данных в HMI или внешний контроллер, HMI или внешний контроллер использует уставки этого регистра для чтения /записи данных рецепта.

Если RCPNO = 1, это определяет 1-ый номер данных рецепта. Если RCPNO = 4 это определяет 4-ый номер данных рецепта.

В дополнение, при редактировании данных рецепта, пользователь может задать размер данных рецепта. Подробнее об этом можно прочитать в разделе **Регистры рецептов RCO.**

При выключении питания данные не сохраняются

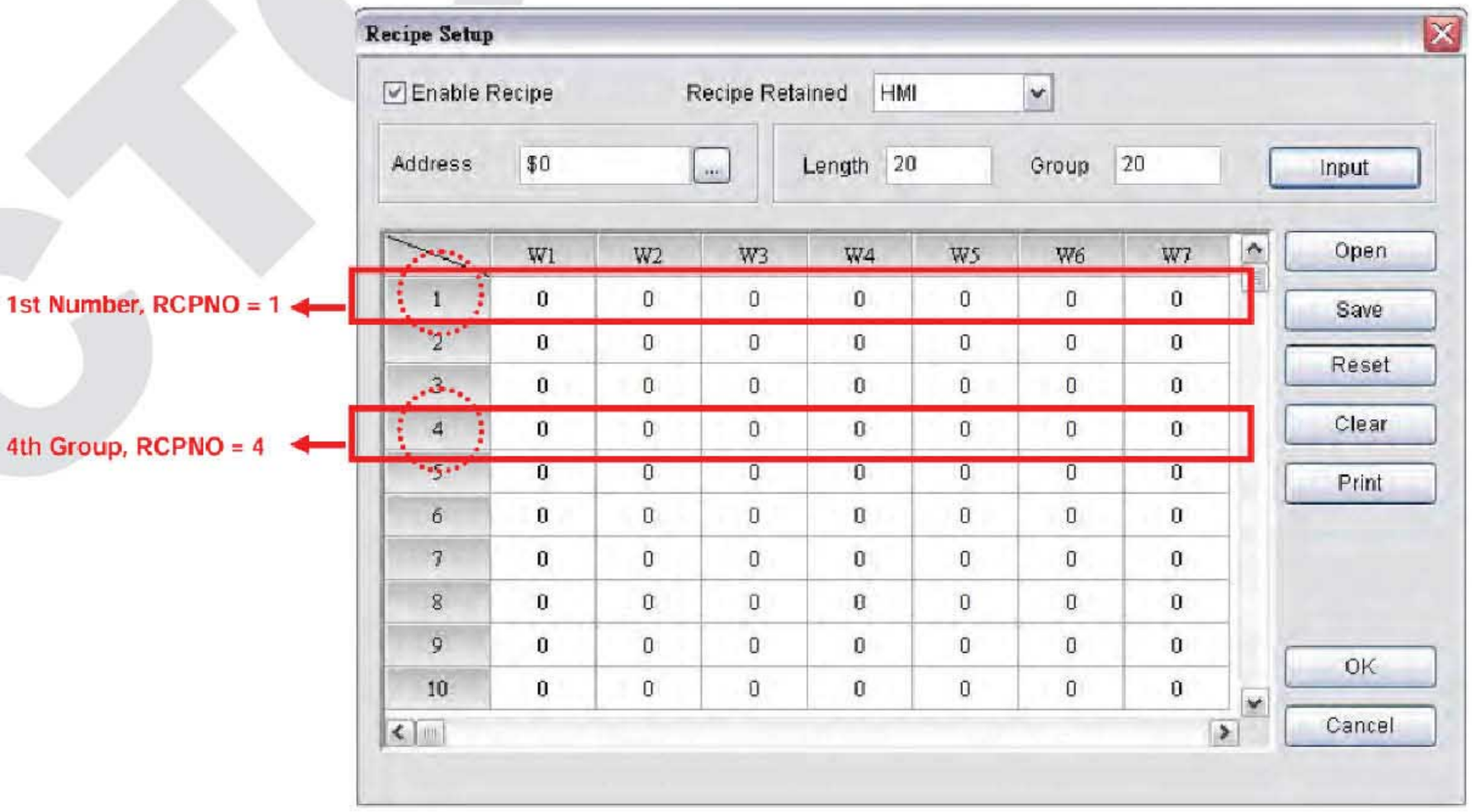


Рис. 3-3-2 Экран редактирования рецептов

3.3.5 Регистры группы рецептов (R/W) : RCPG

Эти регистры используются для доступа к 32-битным группам рецептов (Рис. 3-3-3).

Отличие от RCPNO в том, что при использовании RCPG, задаётся не только номер рецепта, но и номер группы рецептов. При записи/чтении данных 1-ого рецепта 1-ой группы, необходимо установить RCPG = 1 и RCPNO = 1.

Если необходимо обратиться к 4 рецепту 3 группы, то RCPG = 3 и RCPNO = 4.

Номер группы рецептов задаётся при редактировании таблицы рецептов .

Подробнее смотрите в описании Регистров рецептов RCP.

При выключении питания данные не сохраняются

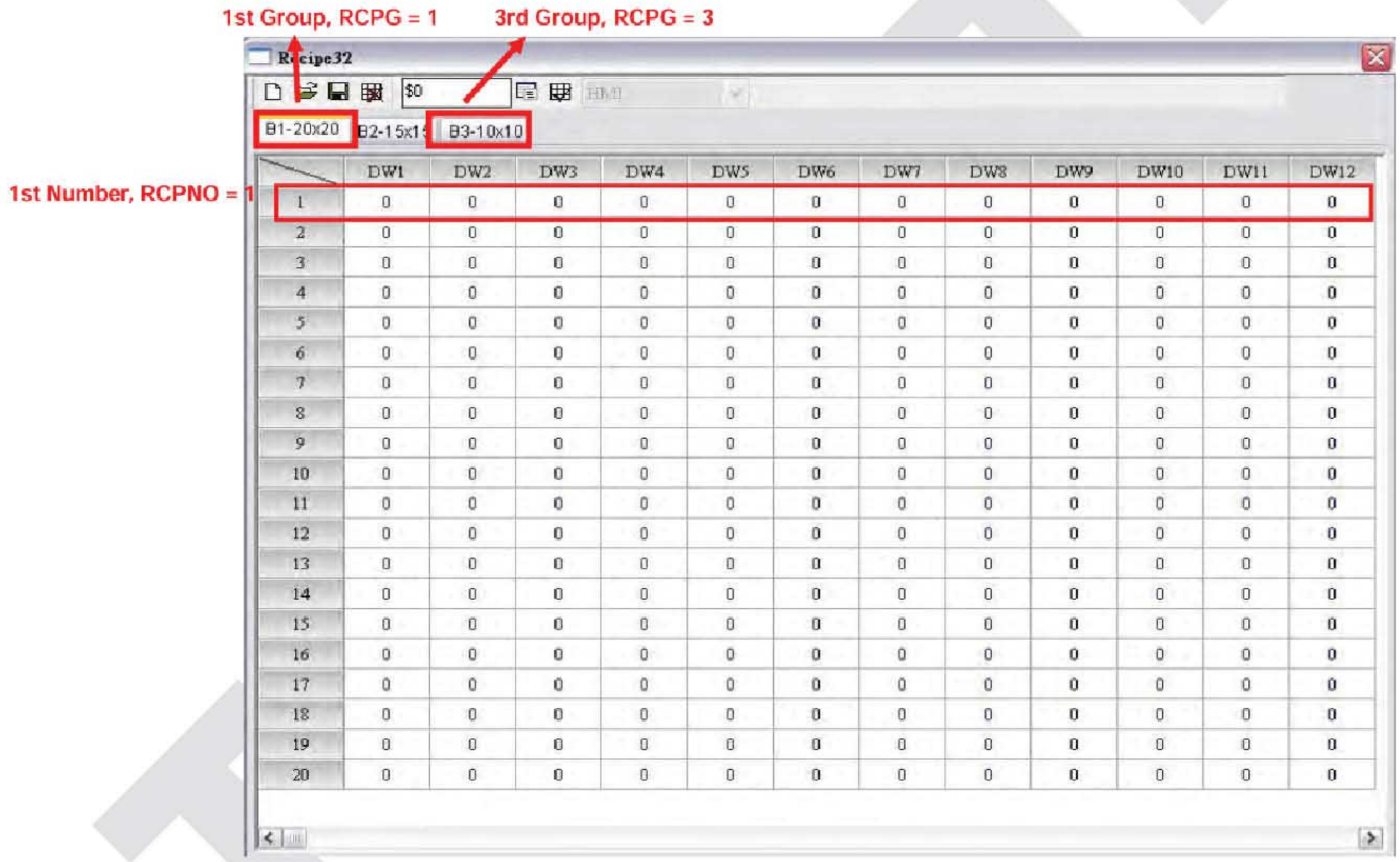


Рис. 3-3-3 Экран регистров группы рецептов

3.3.6 Регистры рецептов (R/W): RCP

Этот регистр используется для сохранения данных рецептов при выходе из **Screen Editor**. Есть два типа таких регистров: 16-битные регистры рецептов и 32-битные регистры рецептов.

16-битный регистр рецептов

Размер такого регистра 16 бит (1 слово).

Если данные хранятся во внешней памяти (USB диск или SM карта), то объём доступной памяти составляет 4MB слов. Если данные хранятся во внутренней памяти, то объём доступной памяти составляет 64K слов.

Например, размер рецепта L и число рецептов N.

Область памяти, занимаемая рецептами составляет L x N слов, т.е. используется L x N регистров.

Панели оператора DOP-B имеют буферную область памяти (Таблица 3-3-2) для хранения выбранных пользователем рецептов. Объём буферной области такой же, как и объём выбранных рецептов. Это также означает, что L рецептов занимает область рецептов. Следовательно, в таблице регистров будет занято L x (N+1) регистров.

Применяется следующий способ доступа к рецептам.

Обращение к словам : RCPn (n : 0-L*(N+1)-**1**).

Число -1 показывает, что нумерация RCP начинается с 0.

	W1	W2
1	36	42
2	12	33
3	27	51

Таблица 3-3-1 Таблица рецептов

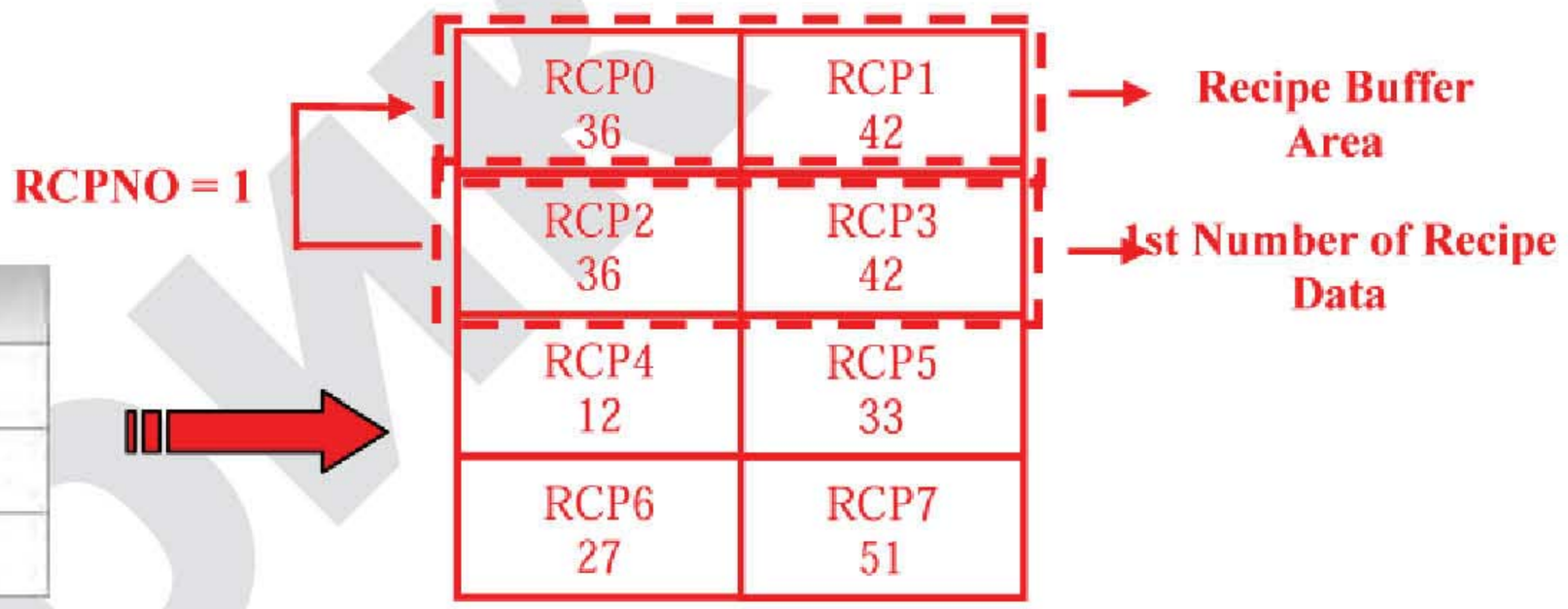


Таблица 3-3-2 Размещение регистров рецептов

Пример 1:

Таблица 3-3-1 показывает как организован доступ к данным. Размер рецепта L = 2 и число рецептов N = 3 определяют размер буферной области. Расположение рецептов показано в таблице 3-3-2, т.е. заняты регистры RCP0~RCP7.

Когда RCPNO =1, содержимое буфера рецептов будет соответствовать первому набору данных.

Когда RCPNO =3, содержимое буфера рецептов будет соответствовать третьему набору данных, как показано в таблице 3-3-4.

	W1	W2
1	36	42
2	12	33
3	27	51

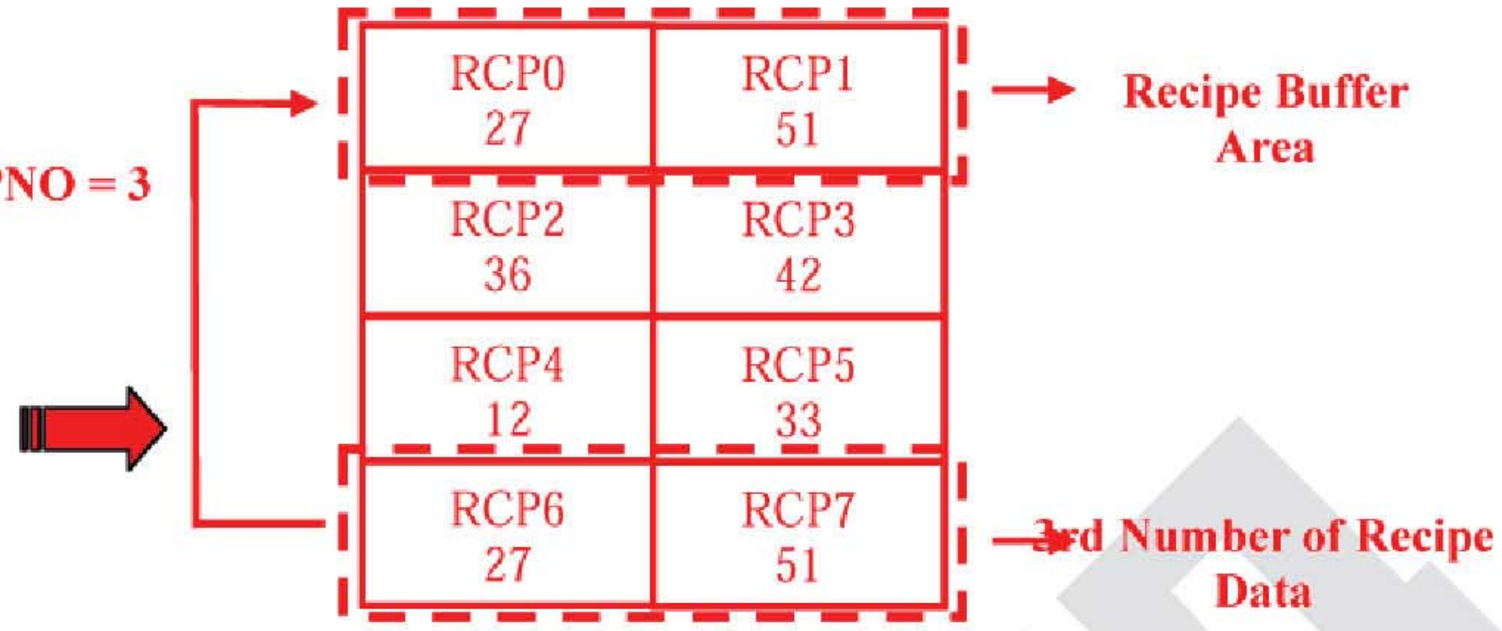


Таблица 3-3-3 таблица рецептов Таблица 3-3-4 Размещение регистров рецептов

Пример 2:

В таблице 3-3-5, заданы L = 3 и N = 2.

Расположение рецептов показано в таблице 3-3-6, т.е.заняты RCP0~RCP8.

Когда RCPNO = 2 содержимое буфера рецептов соответствует второму набору данных .

	W1	W2	W3
1	36	42	1
2	12	33	76

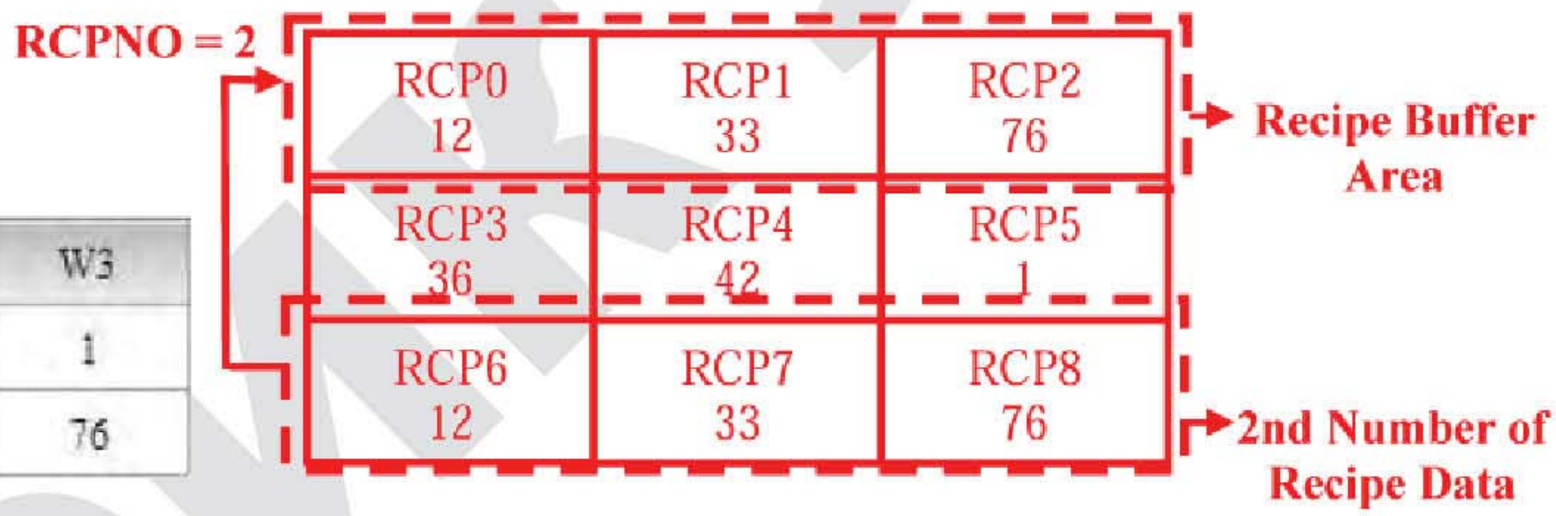


Таблица 3-3-5 Таблица рецептов Таблица 3-3-6 Размещение регистров рецептов

32-битный регистр рецептов

Размер такого регистра 32 бита (2 слова или одно двойное слово).

Если данные хранятся во внешней памяти (USB диск или SM карта), то объём доступной памяти для 32-битных регистров составляет 50MB слов. Если данные хранятся во внутренней памяти, то объём доступной памяти для 32-битных регистров зависит от спецификации флэш памяти панели оператора.

В каталогах DOP-B можно ознакомиться с этими данными. Обратите внимание, что во флэш памяти хранятся не только рецепты, но и другие экранные данные. Общая память рецептов должна быть меньше размера флэш памяти. Для получения детальной информации по распределению памяти можно открыть таблицу распределения памяти (Рис. 3-3-4), войдя в соответствующий раздел, для чего необходимо кликнуть мышью **View > Memory List**.

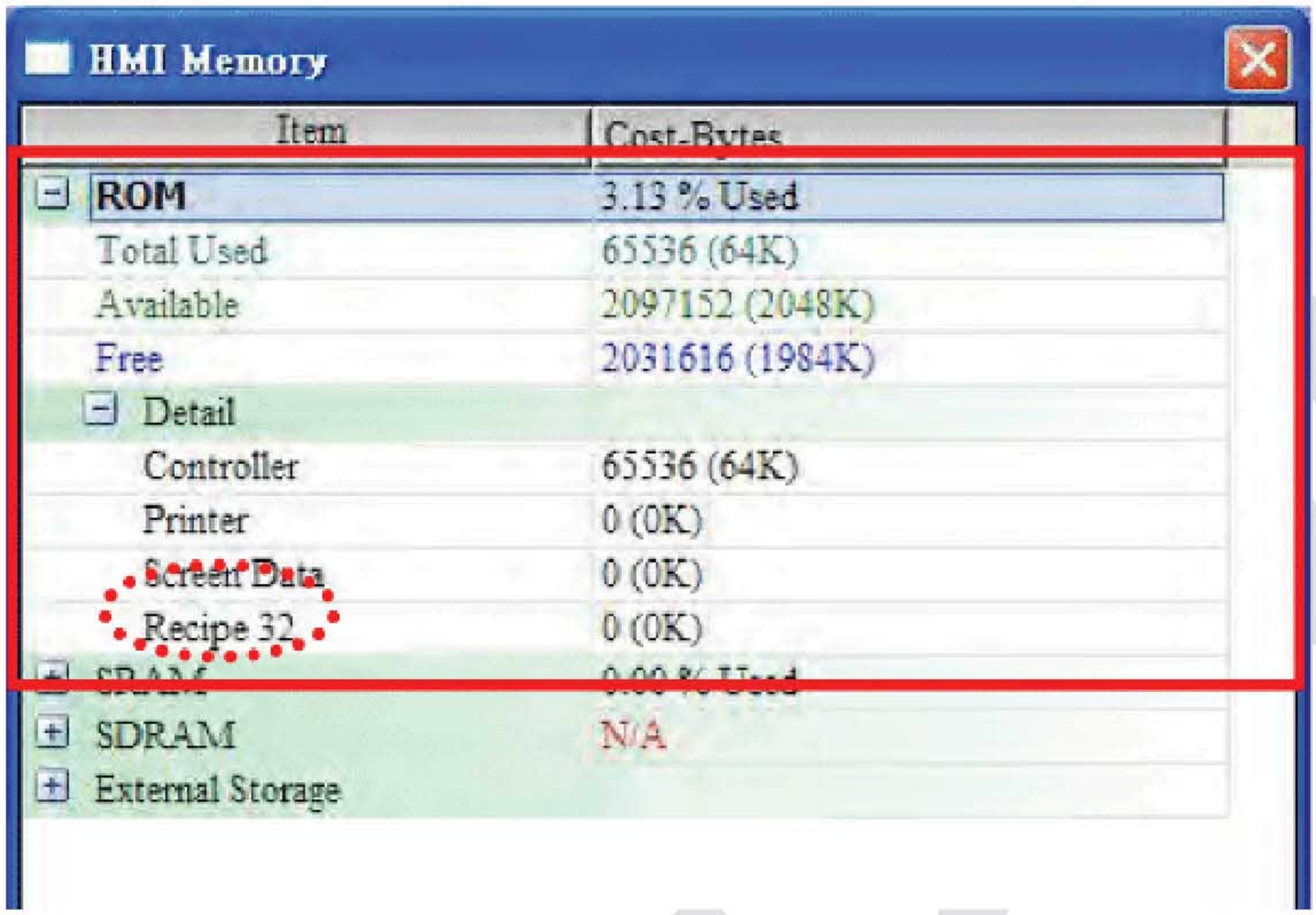


Рис. 3-3-4 Память панели

Например, размер рецепта L и число рецептов N. Область памяти, занимаемая рецептами составляет L x N двойных слов, т.е. используется L x N регистров.

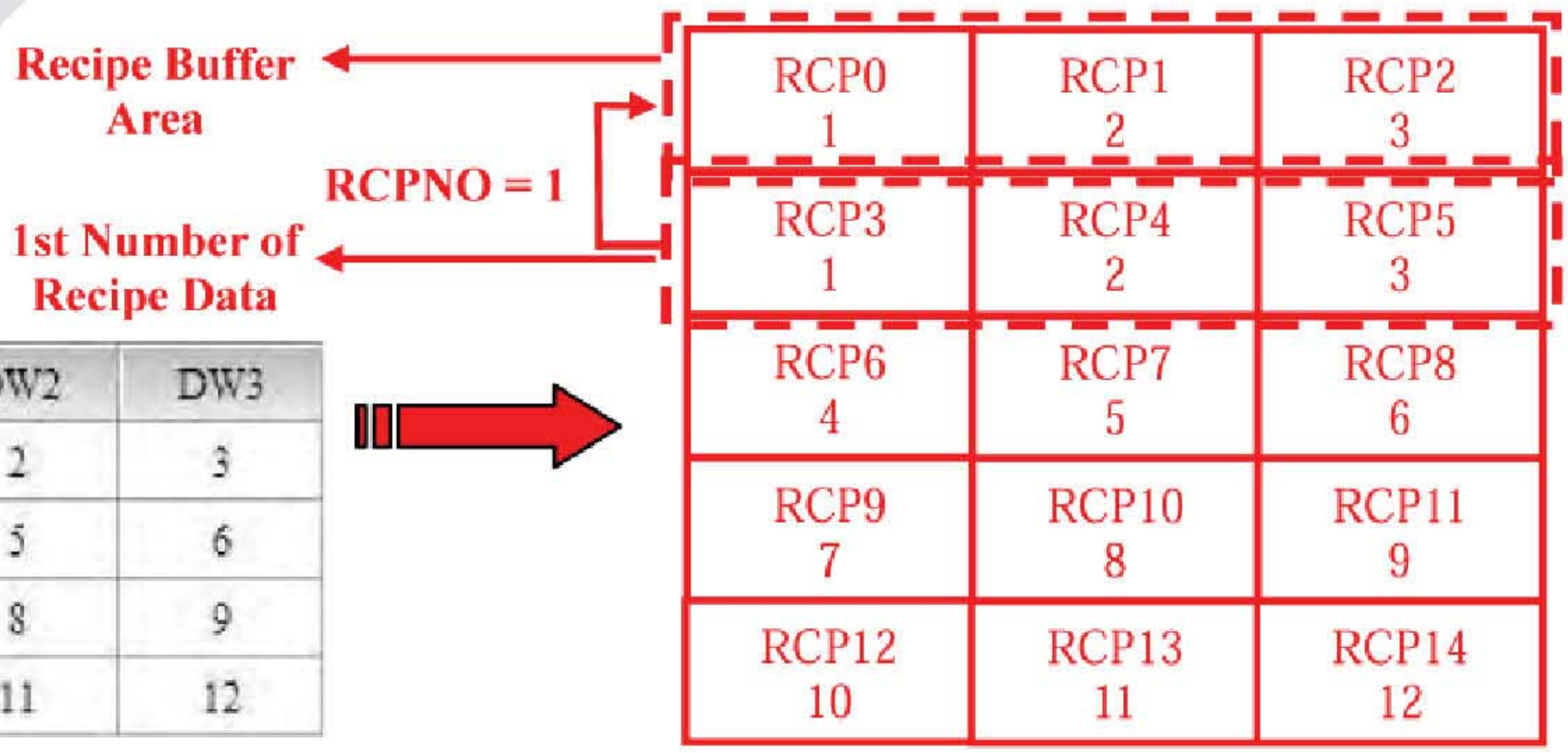
Панель оператора DOP-B имеет буферную область памяти (Таблица 3-3-8) для хранения выбранных пользователем рецептов. Объём буферной области такой же, как и объём выбранных рецептов. Это также означает, что L рецептов занимает буферную область рецептов.

Поэтому, L x (N+1) регистров размещаются в одной таблице рецептов.

Применяется следующий способ доступа к рецептам:

Обращение к двойным словам: RCPn (n : 0-L*(N+1)**-1**)

Число -1 означает, что нумерация RCP начинается с 0.



	DW1	DW2	DW3
1	1	2	3
2	4	5	6
3	7	8	9
4	10	11	12

Таблица 3-3-7 Таблица рецептов

Таблица 3-3-8 Размещение регистров рецептов

Пример 1:

В таблице 3-3-7 показано, как обратиться к первому рецепту (RCPG=1).

Размер L = 3 and число рецептов N= 4. Расположение данных показано виде таблицы 3-3-8, т.е. занято RCP0~RCP14

При RCPNO = 1 содержимое буфера рецептов соответствует первому набору данных.

Если RCPNO =3, содержимое буфера рецептов соответствует третьему набору данных, как показано в Таблице 3-3-10.

	DW1	DW2	DW3
1	1	2	3
2	4	5	6
3	7	8	9
4	10	11	12

Таблица 3-3-9 Таблица рецептов

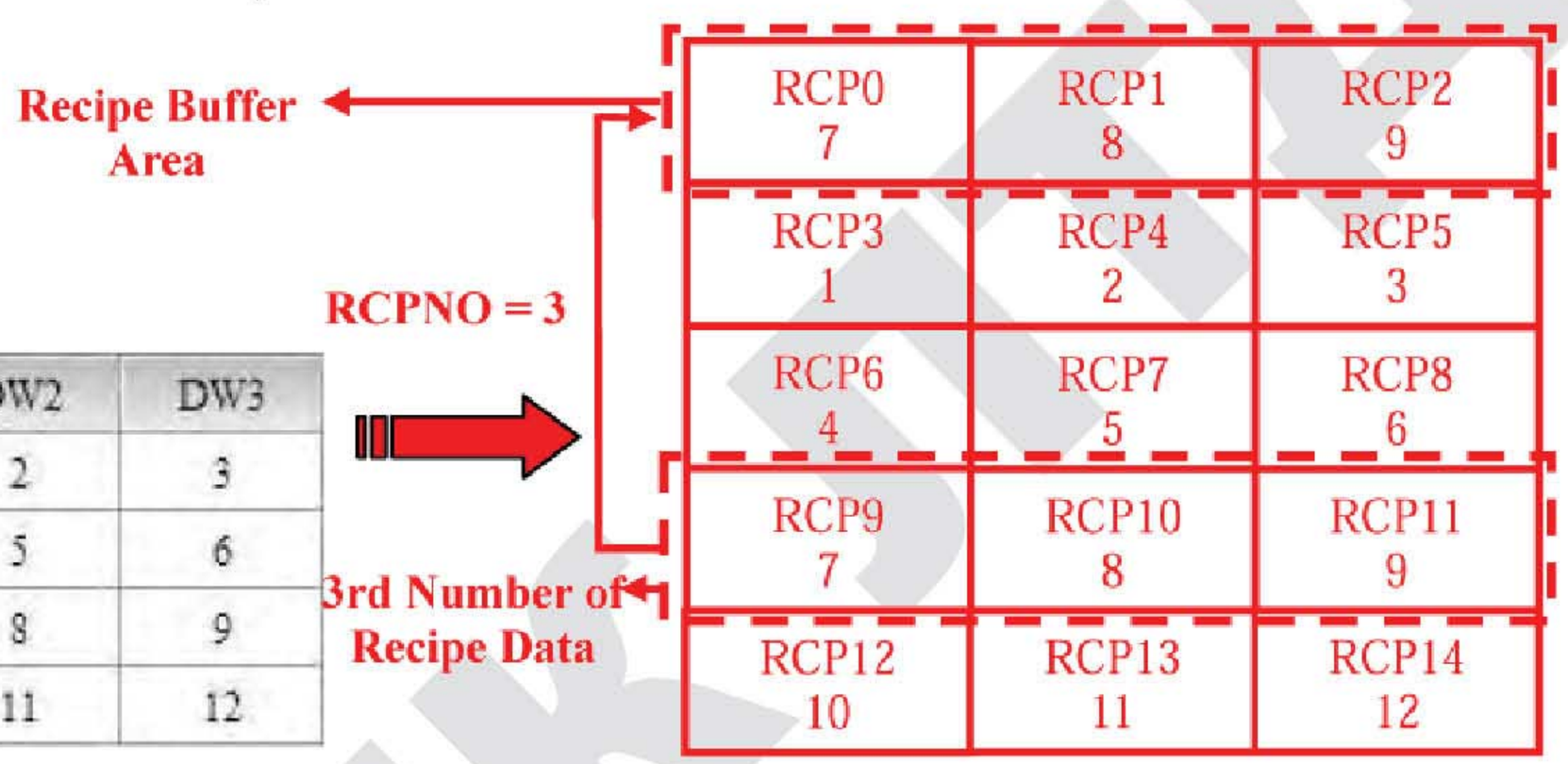


Таблица 3-3-10 Размещение регистров рецептов

Пример 2:

В таблице 3-3-11 показано, что рецепты принадлежат к первой группе, т.е. (RCPG=1).

Размер рецепта L = 2 и число рецептов N = 3.

Расположение данных показано в виде Таблицы 3-3-12, т.е. заняты. RCP0~RCP7.

Когда RCPNO = 3 содержимое буфера рецептов соответствует третьему набору данных.

	DW1	DW2
1	10	11
2	12	13
3	14	15

Таблица 3-3-11 Таблица рецептов

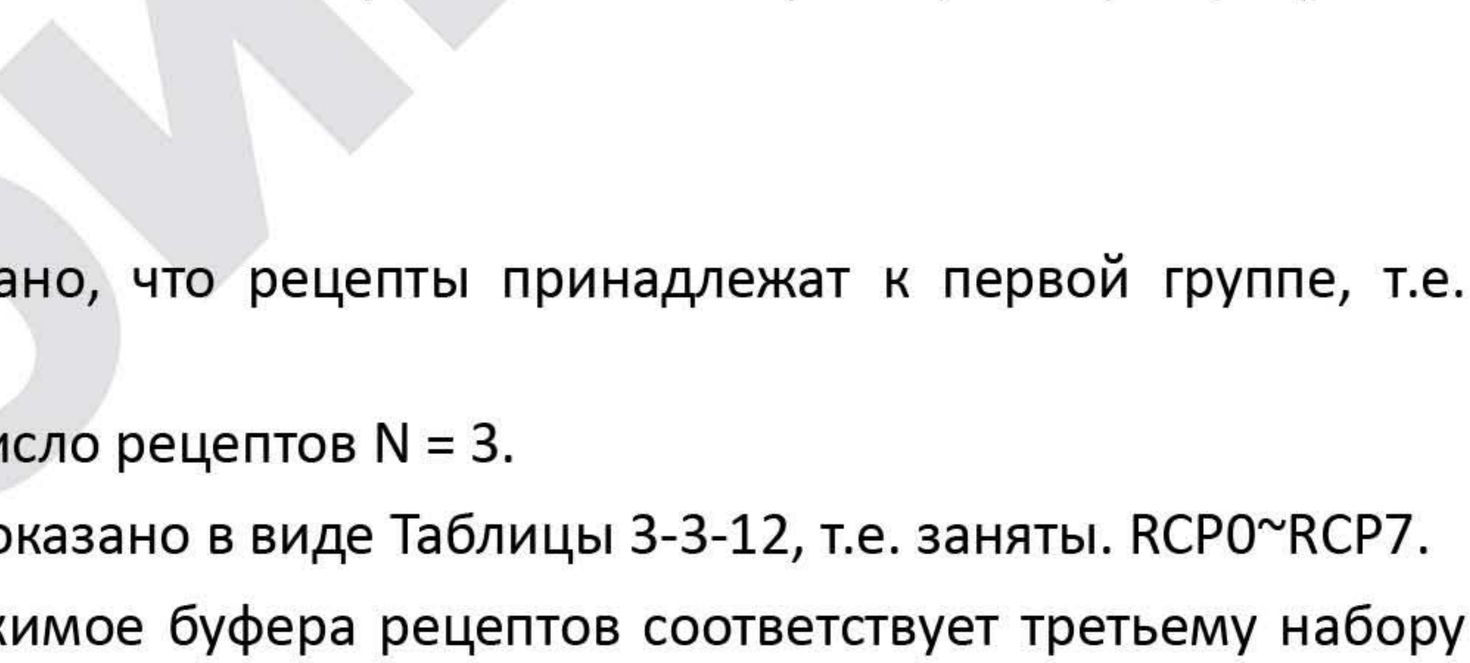


Таблица 3-3-12 Размещение регистров рецептов

3.4 Блок управления и блок состояния

Для реализации двухсторонней связи между панелью DOP и моделями ПЛК других брендов, с целью управления различными функциями панели с контроллера и получения им информации о текущем её состоянии, можно назначить регистры, которые будут выполнять функции блока управления (control block) и блока состояния (status block) панели.

Настройки находятся в разделе **Configuration** (Рис. 3-4-2) Для этого необходимо выбрать закладку **Options > Configuration** (Рис. 3-4-1). Далее приведено более подробное описана процедура настроек.

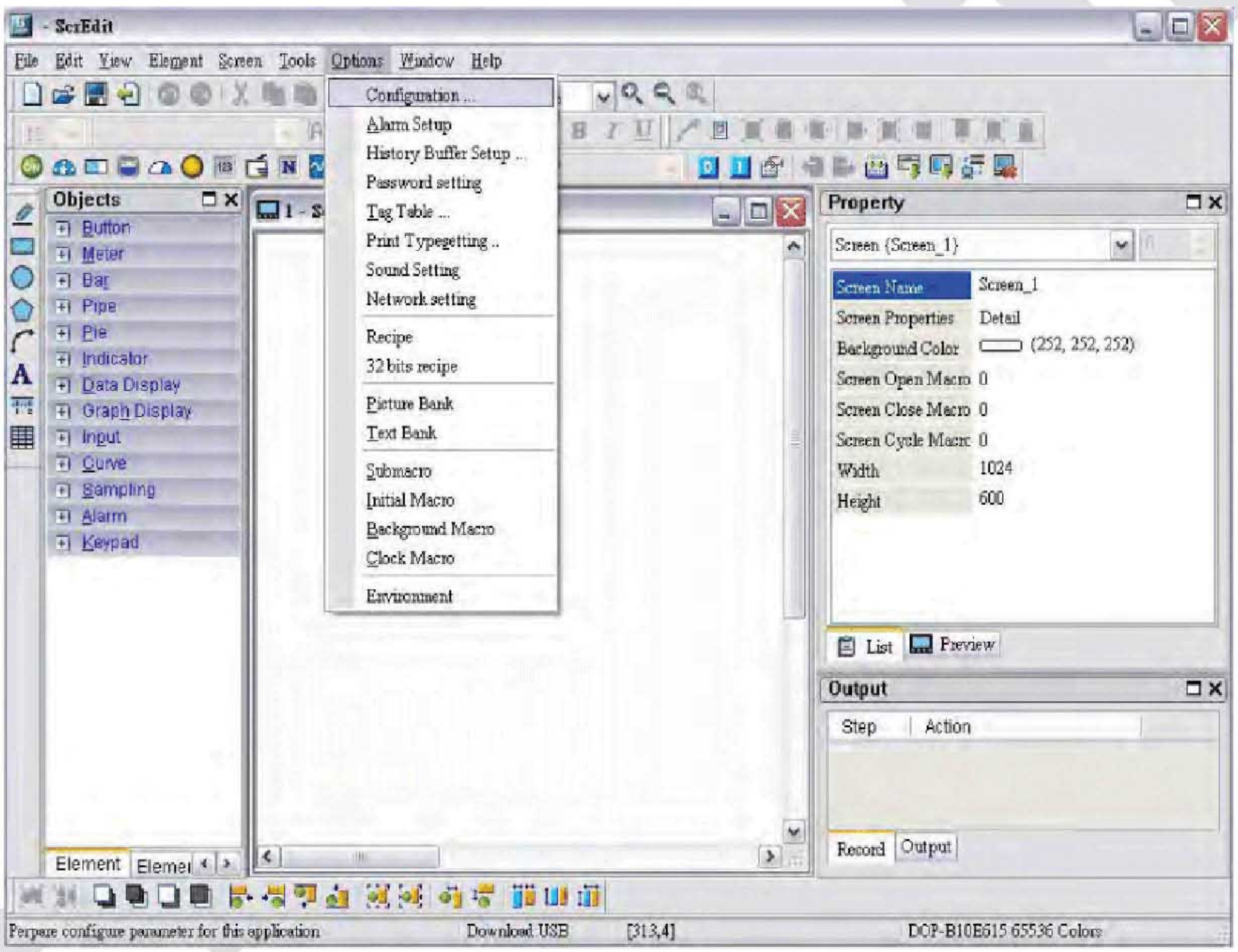


Рис. 3-4-1 Выбор команды Configuration

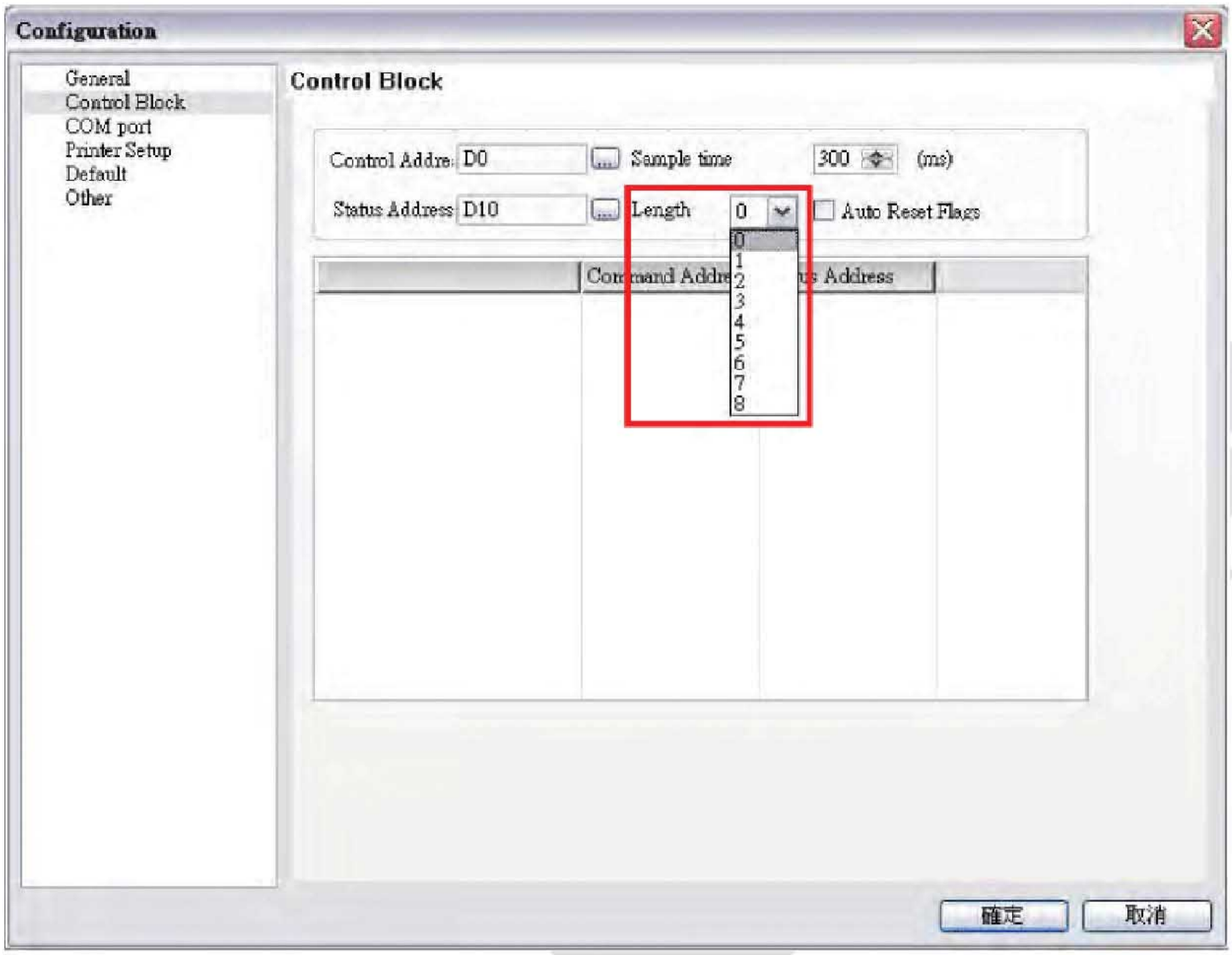


Рис. 3-4-2 Диалоговое окно Configuration

Внимание: Если выбрать **Auto reset flags**, то после выполения желаемой процедуры, произойдёт автоматический сброс флагов блока управления (Bits). Если не выбирать эту функцию, то система будет запрашивать пользователя произвести сброс флагов.

3.4.1 Блок управления

Блок управления даёт возможность с помощью контроллера управлять панелью оператора. С помощью настроек регистров, контроллер, соединённый с панелью оператора может управлять различными функциями панели: переключением экранов, включать заставку, управлять выводом на экран данных в виде графиков, сбрасывать флаги, а также получать информацию о текущем состоянии HMI.

Регистры образуют блок размером от 0 до 8 слов. (Рис. 3-4-2). Конкретная длина задаётся пользователем, исходя из предстоящих задач.

Например, когда используется элемент переключения экранов **Screen Switch** (**Screen Number Designation Register**) длина блока управления должна быть 1 или более. При этом, будет использоваться только **Screen Number Designation Register**.

В другом случае, при использовании журнала событий **History Buffer**

(**Sampling History Buffer Register**), длина блока управления должна быть не менее 4 слов. При этом будет использоваться только **Sampling History Buffer Register**.

При использовании функции многоязыковой поддержки **Multi-language Setting Value Bit (System Control Flag Register)**, длина блока управления должна быть задана равной 8. В этом случае все регистры блока управления могут быть использованы.

При задании нулевой длины блока функция управления панели отключается.

В таблице 3-4-1 приведены функции и описание каждого регистра блока управления. В примере 1 используются регистры внешнего контроллера Delta, адресация регистров соответственно Dn ~ Dn+7 (D0 ~ D7). В примере 2, используются внутренние регистры панели оператора $, адресация регистров соответственно $n ~ $n+7 ($15 ~ $22).

Пользователь выбирает, что использовать для этих целей - контроллер или панель оператора.

Номер слова	Наименование регистра блока управления	Пример 1 (регистр ПЛК)		Пример 2 (регистр HMI)	
		Адрес	Пример	Адрес	Пример
1	Регистр переключения между экранами (SNIR)	Dn	D0	$n	$15
2	Регистр управляющих флагов (CFR)	Dn+1	D1	$n+1	$16
3	Регистр управления построением графиков (CUCR)	Dn+2	D2	$n+2	$17
4	Регистр для выборки данных в буфер архива (HBSR)	Dn+3	D3	$n+3	$18
5	Регистр для очистки буфера архива (HBCR)	Dn+4	D4	$n+4	$19
6	Регистр управления рецептами (RECR)	Dn+5	D5	$n+5	$20
7	Регистр для указания номера рецепта (RBIR)	Dn+6	D6	$n+6	$21
8	Регистр системных управляющих флагов (SCFR)	Dn+7	D7	$n+7	$22

Таблица 3-4-1 Обозначение регистров блока управления

Регистр переключения между экранами (SNIR)

Регистр SNIR (Dn) или ($n) используется для управления переключением между экранами HMI от внешнего ПЛК (D0) или внутренней памяти HMI. Какое значение будет записано в регистре, экран HMI с таким номером и будет отображаться на

дисплее. Например (Таблица 3-4-1), если записать в D0 или $15 число 1, панель переключиться на первый экран. Если записать D0 или $15 число 7, панель переключиться на 7 экран.

Регистр управляющих флагов (CFR)

Бит	Индикация положения соответствующего бита (x)	Функция
0	0000 0000 0000 000x	Разрешение / запрещение коммуникации
1	0000 0000 0000 00x0	Разрешение / запрещение лампы подсветки
2	0000 0000 0000 0x00	Разрешение / запрещение звукового сигнала
3	0000 0000 0000 x000	Очистка буфера аварийных сообщений
4	0000 0000 000x 0000	Очистка счётчика аварий
5	0000 0000 00x0 0000	Обновление USB данных
6-7	0000 0000 xx00 0000	Резерв
8	0000 000x 0000 0000	Установка уровня доступа (Уровень 1)
9	0000 00x0 0000 0000	Установка уровня доступа (Уровень 2)
10	0000 0x00 0000 0000	Установка уровня доступа (Уровень 4)
11-15	xxxx x000 0000 0000	Резерв

- **Разрешение / запрещение коммуникации**

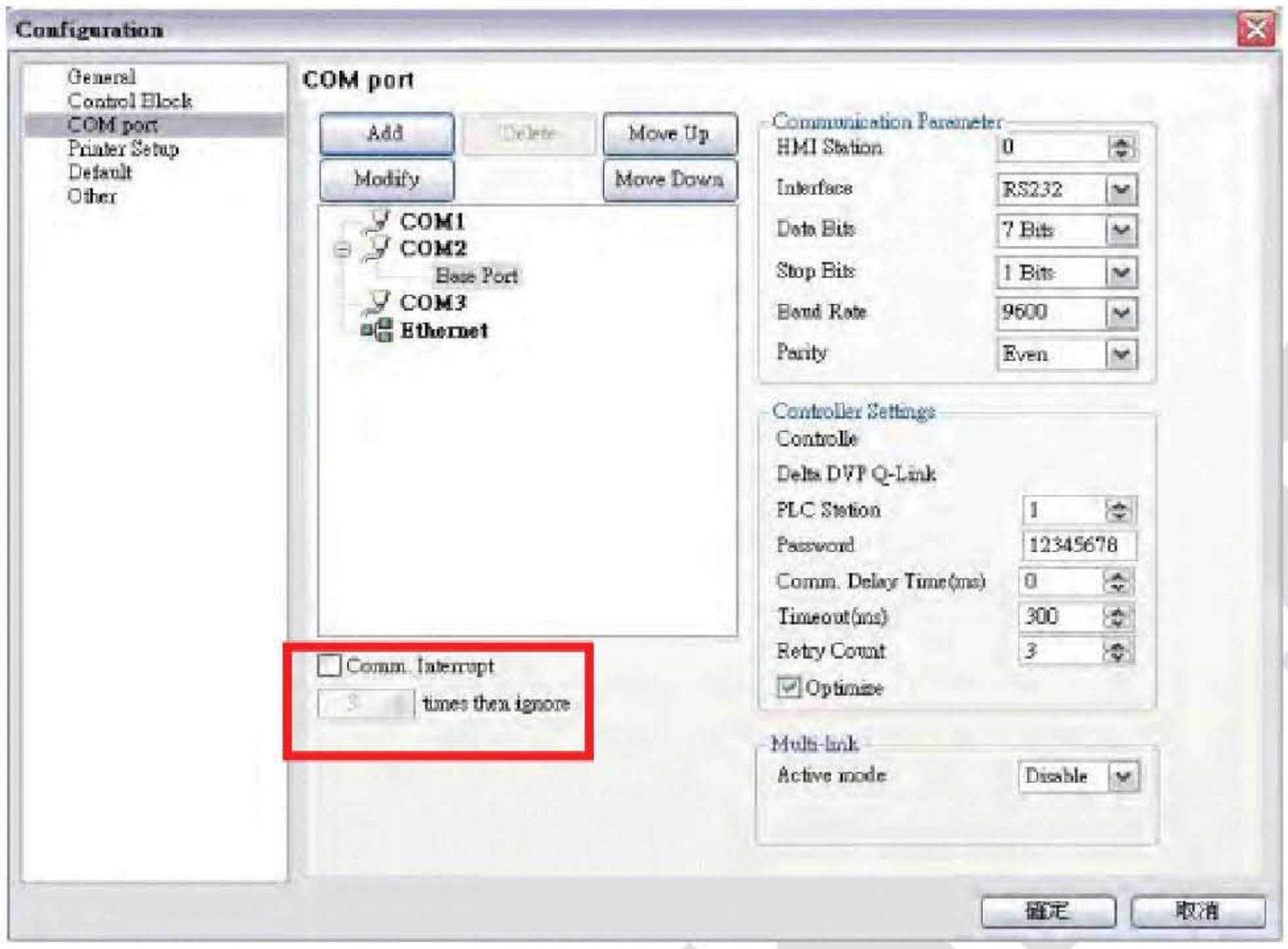
  Бит 0 управляет коммуникацией HMI. Когда bit 0 = ON, связь с HMI будет запрещена. Когда bit0 = OFF, связь с HMI будет разрешена.

  Если выбрать **Communication Interrupt** (Рис. 3-4-3) в разделе **Communication** раздела **Configuration**, когда нарушится связь с определённым контроллером этот бит перейдет в состояние ON, запретит связь автоматически и сообщение об ошибке связи на экране не появится (это не относится к связи между другими контроллерами). Далее, пользователь может сбросить его значение, разрешив связь снова. Если, **Communication Interrupt** не выбран, то этот флаг не активен и связь запрещена. (Используя макрокоманды **OPENCOM/CLOSECOM**, пользователь может управлять работой порта. Более детально об этом написано в разделе 3.14)

  Например (см. таблицу 3-4-1 и рис. 3-4-4), если контроллер, связанный с портом COM2 даёт ошибку связи, и в течении трёх попыток связь не восстановилась, HMI запретит связь с этим контроллером автоматически (но связь с остальными контроллерами не разорвётся) и не будет выдавать сообщений об ошибке связи.

  Одновременно, если D1 или $16.0 равны 0, т.е. коммуникационный

флаг выключен (состояние OFF), связь будет разрешена и панель с этим контроллером снова свяжется. Если связь с контроллером в течении 3 попыток не будет восстановлена этот флаг снова перейдет в состоянии ON. Значение регистра D1=0, означает, что bit0 регистра D1 равен 0 (0000 0000 0000 0000).



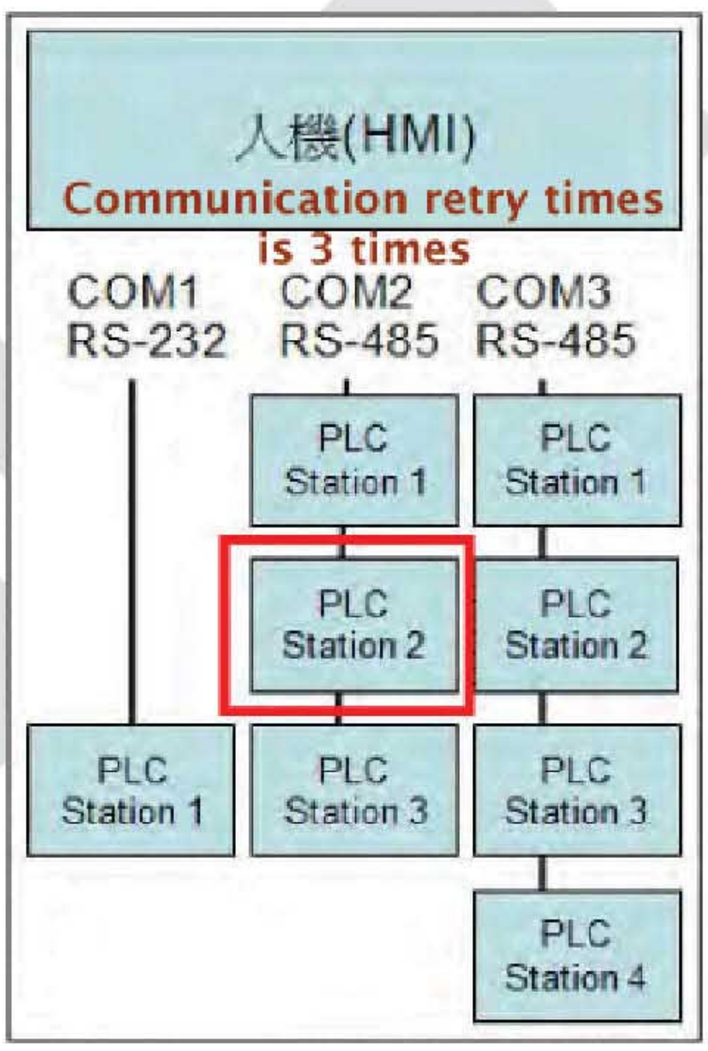


Рис. 3-4-3 Таблица коммуникационных настроек Рис. 3-4-4 Подключения ПЛК

■ Разрешение / запрещение лампы подсветки

Bit 1 управляет подсветкой HMI. Когда bit1=ON, подсветка экрана HMI выключена. Когда bit1=OFF, подсветка включена. Например (см. таблицу 3-4-1), если D1=2 или $16=1 посветка выключена. D1=2 означает, что Bit1 регистра D1 равен 1 (0000 0000 0000 0010).

■ Разрешение / запрещение звукового сигнала

Bit 2 управляет звуковым сигналом HMI. Когда bit 2=ON, звуковой сигнал в HMI выключен. bit 2=OFF, звуковой сигнал включен. Например, (см.таблицу 3-4-1), если D1=4 или $16.2=1, звуковой сигнал HMI будет включен. Если в это время появится сигнал аварии, то раздастся звук. D1=4 означает, что Bit2 регистра D1 равен 1 (0000 0000 0000 0100).

■ Очистка буфера аварийных сообщений

Bit 3 очищает буфер аварийных сообщений.

При использовании буфера аварийных сообщений, когда Bit=3, осуществляется очистка буфера. Для повторной очистки буфера необходимо сбросить бит и повторно его включить. Например (см. таблицу 3-4-1), если D1 = 8 или $16.3 = 1, этот флаг включится и буфер будет очищен. D1 = 8 означает, что Bit3 регистра D1 равен 1 (0000 0000 0000 1000).

■ Очистка счётчика аварий

Bit 4 очищает счётчик аварий. При использовании счётчика аварий установка этого бита очищает его значение. Когда bit 4 = ON, счётчик аварий обнуляется. Для повторного обнуления счетчика необходимо сбросить бит и повторно его включить. Например (см. таблицу 3-4-1), если D1 = 16 или $16.4 = 1, этот флаг включится и счётчик будет обнулён. D1 = 16 означает, что Bit 4 регистра D1 равен 1 (0000 0000 0001 0000).

■ Обновление данных USB диска

Bit 5 производит обновление данных USB диска. При использовании данного флага пользователь может обновить сохранённые данные в буферной области аварий на USB диске. Если буфер аварий, буфер событий или функции рецептов активированы, и область энергонезависимой памяти установлена на USB диске, то, когда этот флаг установлен, HMI производит сохранение данных и их обновление в буферной области диска в реальном времени. Все данные сначала будут сохранены в буферной памяти.

До достижения полного заполнения буфера памяти (емкость его достигает 64 кбайт и определяется пользователем в разделе настроек **Configuration**) система не будет проводить сохранение данных и их запись на USB диск. Для предотвращения повреждения USB диска избегайте частой перезаписи данных.

Если объём данных меньше размера буфера, во избежание потери данных, пользователь может выставить этот флаг и произвести запись данных на USB диск.

Например (см.таблицу 3-4-1), если D1=32 или $16.5=1, этот флаг будет выставлен, функция обновления данных будет разрешена. D1=32 означает, что Bit5 регистра D1 равен 1 (0000 0000 0010 0000).

■ Установка уровней доступа

Bit 8 ... 10 определяют задаваемый пользователем уровень доступа.

В панелях оператора настройки уровня доступа делятся на две большие группы: одна группа уровней доступа 0 (наименьший) до 7 (наивысший

уровень) и другая – самый высокий. Bit 8...10 управляют уровнем доступа 0...7.

Подробнее о том, как задать желаемый уровень доступа показано в таблице:

Уровень	Управление флагами		Двоичный вид
	ON	OFF	
Уровень 0		Bit 8, Bit 9, Bit 10	0000 0000 0000 0000
Уровень 1	Bit 8	Bit 9, Bit 10	0000 0001 0000 0000
Уровень 2	Bit 9	Bit 8, Bit 10	0000 0010 0000 0000
Уровень 3	Bit 8, Bit 9	Bit 10	0000 0011 0000 0000
Уровень 4	Bit 10	Bit 8, Bit 9	0000 0100 0000 0000
Уровень 5	Bit 8, Bit 10	Bit 9	0000 0101 0000 0000
Уровень 6	Bit 9, Bit 10	Bit 8	0000 0110 0000 0000
Уровень 7	Bit 8, Bit 9, Bit 10		0000 0111 0000 0000

Например (см.таблицу 3-4-1), если регистр D1=1280 или биты $16.8 и $16.10 установлены в 1, установливается флаг, задающий уровень доступа 5. Когда регистр D1 = 1280, это означает, что и Bit8 и Bit10 регистра D1 одновременно равны 1 (0000 0101 0000 0000).

Регистр управления построением графиков (CUCR)

Бит	Индикация положения соответствующего бита (x)	Функция
0	0000 0000 0000 000x	Разрешение / запрещение коммуникации
1	0000 0000 0000 00x0	Разрешение / запрещение лампы подсветки
2	0000 0000 0000 0x00	Разрешение / запрещение звукового сигнала
3	0000 0000 0000 x000	Очистка буфера аварийных сообщений
4	0000 0000 000x 0000	Очистка счётчика аварий
5	0000 0000 00x0 0000	Обновление USB данных
6-7	0000 0000 xx00 0000	Резерв
8	0000 000x 0000 0000	Установка уровня доступа (Уровень 1)
9	0000 00x0 0000 0000	Установка уровня доступа (Уровень 2)
10	0000 0x00 0000 0000	Установка уровня доступа (Уровень 4)
11-15	xxxx x000 0000 0000	Резерв

■ Флаг построения кривой

Bit0 … Bit 3 (флаги 1…4) управляют построением трендов или двухкоординатных графиков. Когда флаг выставлен (состояние ON), панель оператора последовательно производит чтение данных по адресам, задаваемым контроллером, преобразует их значение в текущее положение точки на экране и показывает его график на экране. (Более детально о настройки таких объектов описано в разделе 3.8.10). Для повторного отображения новых данных, необходимо сбросить флаг, (OFF) и далее повторно его выставить (ON).

Каждый графический элемент управляется соответствующим флагом, путём установки или очистки соответствующего бита в регистре управления построением графиков. Например (см.таблицу 3-4-1 и Рис. 3-4-5), если D2 или $17.0 =1, график 1 будет активирован, а графики 2…4 не будут. Когда регистр D2=1, это говорит о том, что Bit 0 регистра D2 равен 1 (0000 0000 0000 0001)

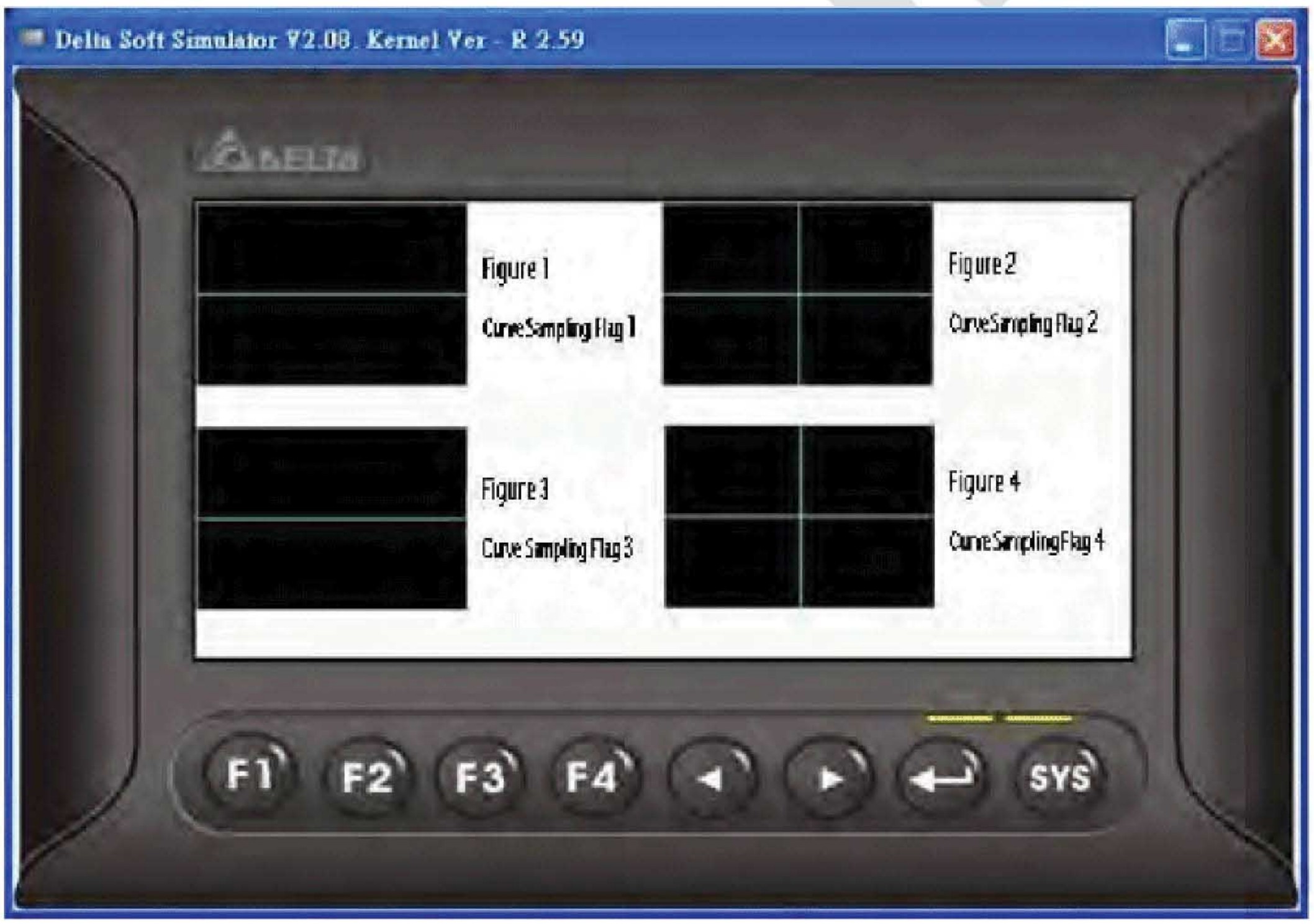


Рис. 3-4-5 Элементы построения графиков на экране

■ Очистка буфера кривой

Bit 8 … 11 (Флаги 1…4) очищают кривую тренда. Или двухкоординатного

графика, когда флаг находится в состоянии ON (более детально настройка описана в разделе 3.8.10). Для управления очисткой буфера с данными необходимо сбросить флаг, а затем снова выставить его в состояние ON)

Например (см.таблицу 3-4-1 и Рис. 3-4-5), если D2 = 512 или $17.9 =1, буфер с данными кривой 3 будет очищен, а с данными для кривых 1, 2 и 4 нет. Когда значение регистра D2=512, это означает, что Bit9 регистра D2 равен 1 (0000 0010 0000 0000).

Регистр для выборки данных в буфер архива (HBSR)

HMI предоставляет возможность записи 12-ти треков архива данных. Выборка данных в HMI может осуществляться через заданные интервалы времени по таймеру или командами ПЛК посредством описываемых здесь флагов.

(См. следующую таблицу и Рис. 3-4-6). Подробнее настройки буфера данных описаны в 3.11.3

Используется для управления выборкой данных в буфере архива данных. См. также описание настройки архива данных **History Setup** в главе 2.

HMI предоставляет возможность записи 12-ти треков архива данных. Выборка данных в HMI может осуществляться через заданные интервалы времени или командами ПЛК посредством описываемых здесь флагов.

Соотвествующая область памяти	Соответствующий флаг		
	Bit	Двоичный вид (x)	Функция
Область памяти 1	0	0000 0000 0000 000x	Флаг управления выборкой данных в буфер архива 1
Область памяти 2	1	0000 0000 0000 00x0	Флаг управления выборкой данных в буфер архива 2
Область памяти 3	2	0000 0000 0000 0x00	Флаг управления выборкой данных в буфер архива 3
Область памяти 4	3	0000 0000 0000 x000	Флаг управления выборкой данных в буфер архива 4
Область памяти 5	4	0000 0000 000x 0000	Флаг управления выборкой данных в буфер архива 5
Область памяти 6	5	0000 0000 00x0 0000	Флаг управления выборкой данных в буфер архива 6
Область памяти 7	6	0000 0000 0x00 0000	Флаг управления выборкой данных в буфер архива 6
Область памяти 8	7	0000 0000 x000 0000	Флаг управления выборкой данных в буфер архива 8

Соотвествующая область памяти	Соответствующий флаг		
	Bit	Двоичный вид (x)	Функция
Область памяти 9	8	0000 000x 0000 0000	Флаг управления выборкой данных в буфер архива 9
Область памяти 10	9	0000 00x0 0000 0000	Флаг управления выборкой данных в буфер архива 10
Область памяти 11	10	0000 0x00 0000 0000	Флаг управления выборкой данных в буфер архива 11
Область памяти 12	11	0000 x000 0000 0000	Флаг управления выборкой данных в буфер архива 12
	12…15	xxxx 0000 0000 0000	Зарезервированы

■ Флаг записи архива

Bits 0 …11 управляют буфером записи архива в панели оператора с помощью контроллера. При установке флага в состояние ON, панель запоминает 1 выборку. Для повторной записи необходимо этот флаг сбросить в состояние OFF и повторно установить в состояние ON.

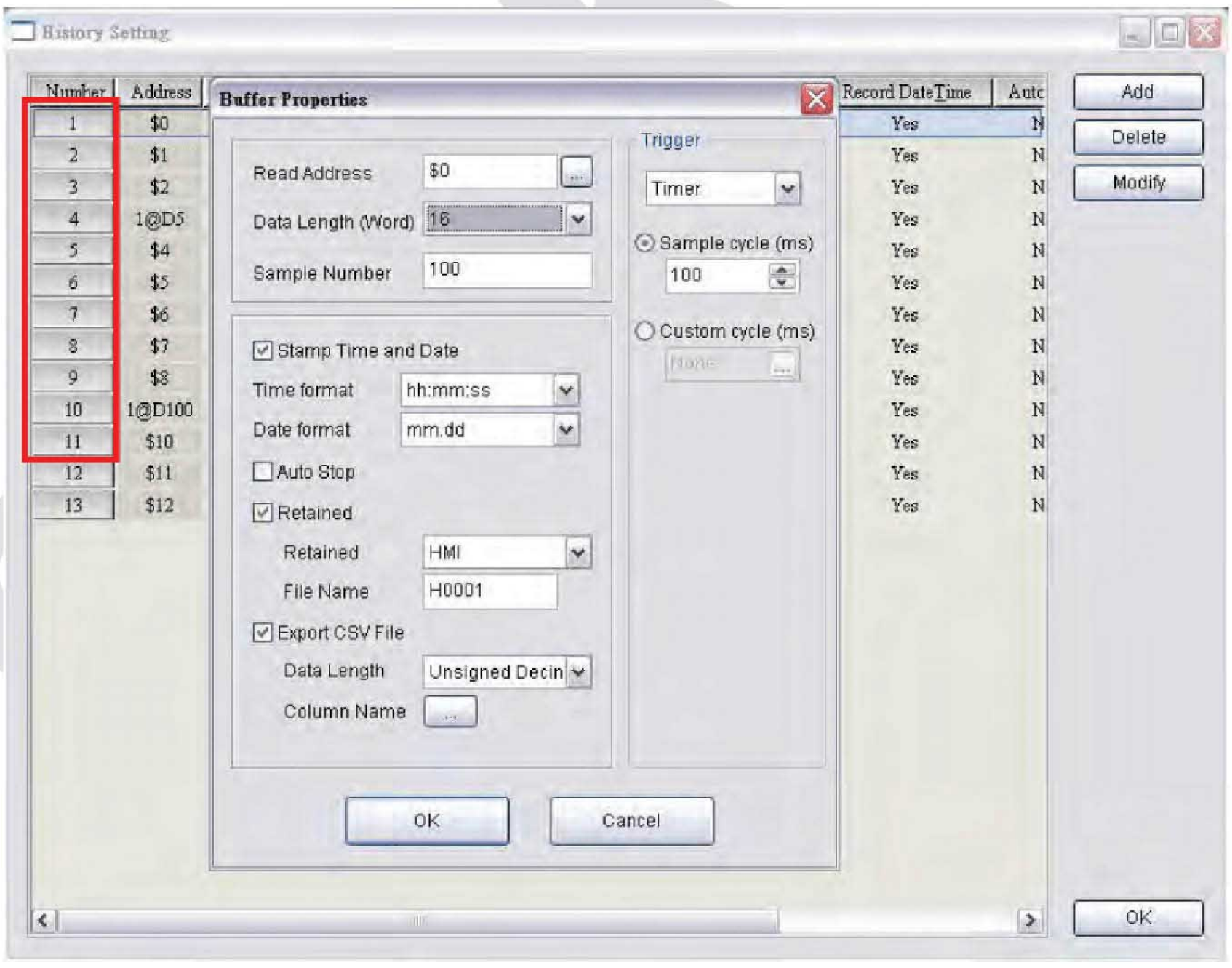


Рис. 3-4-6 Настройки архива

Регистр для очистки буфера архива данных (HBCR)

Буфер архива при управлении от контроллера может очищаться от ранее записанных данных

Соотвествующая область памяти	Соответствующий флаг		
	Bit	Двоичный вид (x)	Функция
Область памяти 1	0	0000 0000 0000 000x	Флаг очистки буфер архива 1
Область памяти 2	1	0000 0000 0000 00x0	Флаг очистки буфер архива 2
Область памяти 3	2	0000 0000 0000 0x00	Флаг очистки буфер архива 3
Область памяти 4	3	0000 0000 0000 x000	Флаг очистки буфер архива 4
Область памяти 5	4	0000 0000 000x 0000	Флаг очистки буфер архива 5
Область памяти 6	5	0000 0000 00x0 0000	Флаг очистки буфер архива 6
Область памяти 7	6	0000 0000 0x00 0000	Флаг очистки буфер архива 7
Область памяти 8	7	0000 0000 x000 0000	Флаг очистки буфер архива 8
Область памяти 9	8	0000 000x 0000 0000	Флаг очистки буфер архива 9
Область памяти 10	9	0000 00x0 0000 0000	Флаг очистки буфер архива 10
Область памяти 11	10	0000 0x00 0000 0000	Флаг очистки буфер архива 11
Область памяти 12	11	0000 x000 0000 0000	Флаг очистки буфер архива 12
	12…15	xxxx 0000 0000 0000	Зарезервированы

- **Флаг очистки буфера архива**

  С помощью Bit 0 … 11 производится очистка буфера при установке этих битов в состояние ON. Для повторной очистки буфера необходимо сбросить эти флаги в состояние OFF а за тем повторно установить в состояние ON.

Регистр управления рецептами (RECR)

Бит	Двоичный вид (x)	Функция
0	0000 0000 0000 000x	Изменение номера рецепта
1	0000 0000 0000 00x0	Чтение рецепта (ПЛК -> HMI) Чтение рецепта из ПЛК и запись в HMI
2	0000 0000 0000 0x00	Запись рецепта (HMI -> ПЛК) Чтение рецепта из HMI и запись в ПЛК
3	0000 0000 0000 x000	Флаг изменения номера группы рецептов
4-7	0000 0000 xxxx 0000	Зарезервировано
8-15	xxxx xxxx 0000 0000	Необходимый номер группы рецептов

■ Флаг изменения номера рецепта

Для смены номера рецепта, пользователь может использовать **Recipe Number Register RCPNO** или использовать этот флаг. Для изменения номера рецепта, запишите номер рецепта в регистр **Recipe Number Designation Register (RBIR)** и затем установите этот флаг в состояние ON (Bit 0=1).

После таких настроек, the RCPNP может изменить значение номера на необходимое автоматически. Повторно перезаписать номер возможно сбросив бит в состоянииеи 0, а затем снова изменив его в состояние ON.

Например (см .таблицу 3-4-1), если установить D6 = 3 и установить D5 или $20.0 в 1 последовательно, номер рецепта станет 3 (RCPNO=3). D6 = 3 означает, что требуется записать номер рецепта 3. Когда D5 = 1, это означает, что Bit 0 регистра D5 равен 1 (0000 0000 0000 0001).

■ Чтение флага рецепта

Bit 1 читает данные рецептов из ПЛК и хранит их в обозначенной области HMI. Для чтения и сохранения данных рецепта задайте номер рецепта, и установите этот флаг в ON. Для повторного чтения рецепта, необходимо перевести флаг в состояние OFF и затем снова установить его в состояние ON.

Например (см. таблицу 3-4-1), если требуется изменить рецепт 4 (RCP-NO=4), то установите D5 = 2 или $20.1 = 1, и данные рецепта, сохраненные в контроллере будут прочитаны и записаны в 4 регистр рецептов. Имеющиеся в этом регистре данные будут обновлены. Когда D5 = 2, это означает, что Bit 1 регистра D5 = 1 (0000 0000 0000 0010).

■ Запись флага рецепта

Bit 2 обеспечивает запись данных рецептов из панели оператора в контроллер. Для этого необходимо задать номер рецепта и установить флаг. Для повторной записи необходимо сбросить флаг в состояние OFF и снова его установить в состояние ON.

Например (см. Таблицу 3-4-1), если номер рецепта 2 (RCPNO=2), установите D5=4 или $20.2 = 1, и данные рецепта, хранимые в панели оператора будут немедленно записаны в регистр контроллера, хранимые там данные будут заменены. Когда D5 = 4, это означает, что Bit 2 регистра D5 = 1 (0000 0000 0000 0100).

■ Изменение номера группы рецепта

Для изменения номера группы рецептов, пользователь может непосредственно использовать регистр номера группы рецептов RCPG, или использовать флаг Bit3. Для записи номера группы регистров. запишите требуемое значение в регистр номера группы рецептов **Designate Recipe Group Number Register (Bits 8 to 15)** и далее установите этот флаг. После этого, в панели будет изменено значение регистра RCPG и автоматически изменится номер группы рецептов.

Для повторной записи необходимо сбросить флаг в состояние OFF и снова его установить в состояние ON. Например (см.таблицу 3-4-1), если установить D5 = 520 или установить $20.3 =1 $20.9 =1 одновременно, номер группы рецептов станет равным 2 (RCPG=2). Когда D5 = 520, это означает. что и Bit 3 и Bit 9 регистра D5 оба равны 1 (0000 0010 0000 1000).

■ Запись номера группы рецептов

Номер группы рецептов записывается битами Bit 8 ... 15.

Далее детально показаны настройки регистра D5 = 520 (0000 0010 0000 1000)

D5 = 520 (0000 0010 0000 1000)	
Старший байт (Регистр номера группы рецептов) 0000 0010	Младший байт (Другие флаги управления) 00001000

При выделении в регистре D5 старшего и младшего байта можно видеть, что старший байт определяет номер группы рецептов. Когда там записано число 0000 0010, это говорит о том, что выбрана 2 группа. Если записать 0000 0011, это говорит о том, что выбрана 3 группа.

Регистр для указания номера рецепта (RBIR)

Этот регистр используется для задания рецепта. Когда установлен флаг ON, система запишет назначенный номер группы рецептов в **Recipe Number Designation Register (RBIR)**. После установки флага смены номера рецепта **Change Recipe Number Flag** в состояние ON, панель оператора изменит значение RCPNO и изменит номер рецепта.

Например (см. таблицу 3-4-1), если значение регистра D6 = 3 и установить D5 =1 или $20.0 = 1 одновременно, Номер рецепта станет равным 3 (RCPNO=3). Когда D6 = 3, это означает, что назначен 3 номер рецепта . Когда D5 = 1, это означает. что в этом регистре Bit 0 = 1 (0000 0000 0000 0001).

Регистр системных управляющих флагов(SCFR)

Бит	Двоичный вид (x)	Функция
0-7	0000 0000 xxxx xxxx	Уставка многоязыковой поддержки
8	0000 000x 0000 0000	Флаг печати
9	0000 00x0 0000 0000	Флаг подачи формы принтера
10-15	xxxx xx00 0000 0000	Резерв

■ Уставка многоязыковой поддержки

В битах 1 … 7 задаётся значение уставок многоязыковой поддержки (Рис. 3-4-7). С помощью этих настроек обеспечивается переключение в панели оператора на соответствующий язык (детальнее об этом можно прочитать в разделе 3.10).

Например (см.таблицу 3-4-1 и Рис. 3-4-7), если уставка 1 соответствует китайскому языку и D7 =1 или $22.0 = 1, то весь текст будет на китайскойм языке. Значение регистра D7 =1, означает, что Bit 0 = 1 (0000 0000 0000 0001).

D7 = 1(0000 0000 0000 0001)	
Старший байт (другие флаги управления) 0000 0000	Младший байт (Уставка многоязыковой поддержки) 0000 0001

При выделении в регистре D7 старшего и младшего байта можно видеть, что младший байт определяет выбор языка (независимо от старшего байта). При значении младшего байта =1 - выбран китайский язык, когда =2 - английский.

Диапазон уставок от 0 до 255.

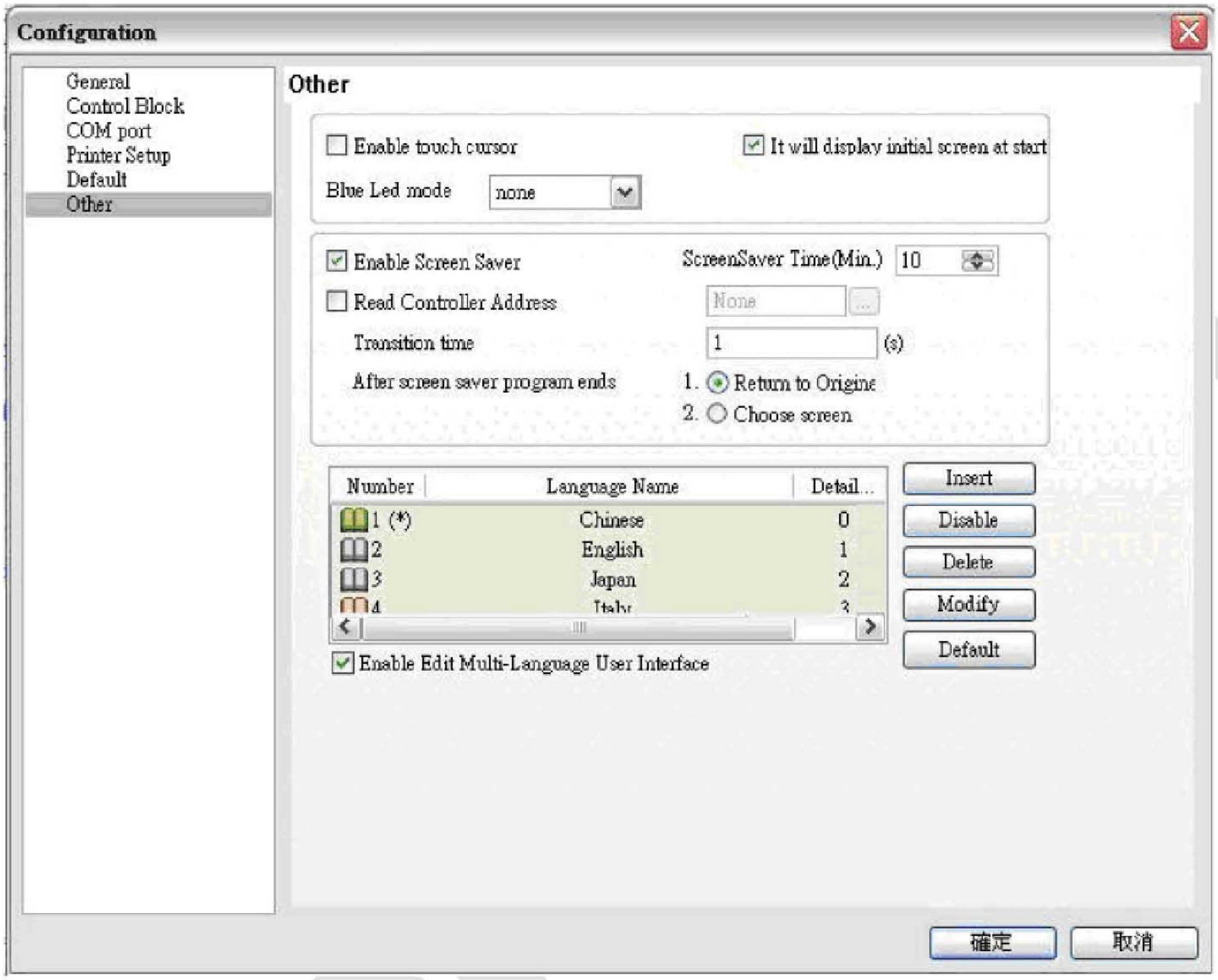


Рис. 3-4-7 Многоязыковая поддержка

■ Флаг печати

Когда этот флаг установлен, то текущий или редактируемый экран может быть распечатан. Когда флаг сброшен, функции печати отключены.

Например (см.таблицу 3-4-1), если значение регистра D7 = 256 или $22.8 =1, панель оператора будет обеспечивать выполнение функций печати. Значение регистра D7 = 256 означает, что в регистре D7 Bit 8 = 1 (0000 0001 0000 0000).

■ Флаг автозагрузки бумаги

Когда этот флаг установлен, принтер загрузит лист бумаги и автоматически подготовится к печати. Когда флаг сброшен, эта функция отключена.

Например (см.таблицу 3-4-1), если значение регистра D7 = 512 или $22.9 =1, панель оператора будет обеспечивать выполнение функций

автозагрузки. Значение регистра D7 = 512 означает, что в регистре D7 Bit 9 = 1 (0000 0010 0000 0000).

3.4.2 Блок состояния

Блок состояния обеспечивает для контроллера возможность получить информацию о состоянии панели оператора. С помощью уставок этих регистров контроллер, соединённый с панелью оператора, может контролировать текущее состояние панели, например, состояние подсветки экрана, переключение экрана состояние флагов, графиков и другую информацию. Блок состояния имеет 8 последовательно расположенных слов. (Рис. 3-4-2)

При задании адреса блока состояния контроллер может читать состояние панели оператора серии DOP-B. Отметим, что если функции блока управления отключены, так же отключены и функции блока состояния. Их адреса должны устанавливаться разными. При настройке необходимо устанавливать разные адреса .

Режимы и описание каждого слова приведены в таблице 3-4-2

В примере 1 показана адресация регистров состояния при использовании регистров памяти контроллера Delta ПЛК Dm ~ Dm+7 (D10 ~ D17).

В примере 2 показана адресация регистров состояния при использовании регистров внутренней памяти панели оператора $m ~ $m+7 ($25 ~ $32).

Пользователь может выбрать где будет хранится адресация регистров состояния - на ПЛК или панели.

Слово	Регистр управления	Пример1 (ПЛК регистр)		Пример 2 (HMI регистр)	
		Адрес	Пример	Адрес	Пример
0	Регистр состояния для общего контроля (GCSR)	Dm	D10	$m	$25
1	Регистр состояния номера экрана (SNSR)	Dm+1	D11	$m+1	$26
2	Регистр состояния построения графиков (CCSR)	Dm+2	D12	$m+2	$27
3	Регистр состояния выборки данных в буфер	Dm+3	D13	$m+3	$28
4	Регистр состояния очистки буфера архива (HCSR)	Dm+4	D14	$m+4	$29
5	Регистр состояния рецептов (RESR)	Dm+5	D15	$m+5	$30
6	Регистр состояния номера рецепта (RBSR)	Dm+6	D16	$m+6	$31

7	Регистр состояния 2 для общего контроля (GCSR2)	Dm+7	D17	$m+7	$32

Таблица 3-4-2 Назначение регистров блока состояния

Регистр состояния для общего контроля (GCSR)		
Бит	**Двоичный вид (x)**	**Функция**
0	0000 0000 0000 000x	Screen Switch Status Flag
1-2	0000 0000 0000 0xx0	Reserved
3	0000 0000 0000 x000	Clear Status of Alarm Buffer
4	0000 0000 000x 0000	Clear Status of Alarm Counter
5-7	0000 0000 xxx0 0000	Reserved
8	0000 000x 0000 0000	User Security Level Flag (Level 1)
9	0000 00x0 0000 0000	User Security Level Flag (Level 2)
10	0000 0x00 0000 0000	User Security Level Flag (Level 4)
11-15	xxxx x000 0000 0000	Reserved

- **Состояние переключения между экранами**

  В процессе переключения экранов этот флаг установлен в состояние ON. После завершения - в OFF

- **Статус очистки буфера аварий**

  В процессе очистки буфера аварий этот флаг установлен в ON. После завершения - в OFF

- **Статус очистки счетчика аварий**

  В процессе очистки счётчика аварий этот флаг установлен в ON. После завершения - в OFF

- **Состояние уровня доступа**

  Bit 8 … 10 описывают текущий уровень доступа заданный пользователем.

Уровень	Состояние управляющих флагов		Двоичный вид
	ON	OFF	
Уровень 0	-	Bit 8, Bit 9, Bit 10	0000 0000 0000 0000

Уровень	Состояние управляющих флагов		Двоичный вид
	ON	OFF	
Уровень 1	Bit 8	Bit 9, Bit 10	0000 0001 0000 0000
Уровень 2	Bit 9	Bit 8, Bit 10	0000 0010 0000 0000
Уровень 3	Bit 8, Bit 9	Bit 10	0000 0011 0000 0000
Уровень 4	Bit 10	Bit 8, Bit 9	0000 0100 0000 0000
Уровень 5	Bit 8, Bit 10	Bit 9	0000 0101 0000 0000
Уровень 6	Bit 9, Bit 10	Bit 8	0000 0110 0000 0000
Уровень 7	Bit 8, Bit 9, Bit 10	-	0000 0111 0000 0000

Регистр состояния номера экрана (SNSR)

В регистре хранится номер последнего открытого пользователем экрана (см. D11 или $26 в таблице 3-4-2).

Регистр состояния построения графиков (CCSR)

Бит	Двоичный вид (x)	Функция
0	0000 0000 0000 000x	Состояние флага построения кривой 1
1	0000 0000 0000 00x0	Состояние флага построения кривой 2
2	0000 0000 0000 0x00	Состояние флага построения кривой 3
3	0000 0000 0000 x000	Состояние флага построения кривой 4
4-7	0000 0000 xxxx 0000	Зарезервировано
8	0000 000x 0000 0000	Состояние флага очистки буфера 1
9	0000 00x0 0000 0000	Состояние флага очистки буфера 2
10	0000 0x00 0000 0000	Состояние флага очистки буфера 3
11	0000 x000 0000 0000	Состояние флага очистки буфера 4
12-15	xxxx 0000 0000 0000	Зарезервировано

- **Curve Sampling Status Flag (Флаг режима построения графиков)**

  При построении в HMI графического тренда или графика X-Y, флаг построения графика (bit 0 … 3) переходит в состояние ON. После построения графика эти флаги переходят в состояние OFF. Флаг построения графика 1 регистра управления соответствует флагу состояния графика 1 регистра состояния.

  Флаг построения графика 2 регистра управления соответствует флагу состояния графика 2.

■ Curve Clear Status Flag
(Флаг режима очистки буфера выводимых данных)

При очистке буфера данных флаг очистки соответствующего буфера (bits 8 … 11) переходит в состояние ON. После завершения очистки эти флаги переходят в состояние OFF. Флаг очистки 1 регистра управления соответствует флагу 1 регистра состояния. Флаг очистки 2 регистра управления соответствует флагу 2 регистра состояния.

Регистр состояния выборки данных в буфер архива (HSSR)

Соотвествующая область памяти	Соответствующий флаг		
	Bit	Двоичный вид (x)	Функция
Область памяти 1	0	0000 0000 0000 000x	Флаг очистки буфер архива 1
Область памяти 2	1	0000 0000 0000 00x0	Флаг очистки буфер архива 2
Область памяти 3	2	0000 0000 0000 0x00	Флаг очистки буфер архива 3
Область памяти 4	3	0000 0000 0000 x000	Флаг очистки буфер архива 4
Область памяти 5	4	0000 0000 000x 0000	Флаг очистки буфер архива 5
Область памяти 6	5	0000 0000 00x0 0000	Флаг очистки буфер архива 6
Область памяти 7	6	0000 0000 0x00 0000	Флаг очистки буфер архива 7
Область памяти 8	7	0000 0000 x000 0000	Флаг очистки буфер архива 8
Область памяти 9	8	0000 000x 0000 0000	Флаг очистки буфер архива 9
Область памяти 10	9	0000 00x0 0000 0000	Флаг очистки буфер архива 10
Область памяти 11	10	0000 0x00 0000 0000	Флаг очистки буфер архива 11
Область памяти 12	11	0000 x000 0000 0000	Флаг очистки буфер архива 12
	12-15	xxxx 0000 0000 0000	Зарезервированы

■ Sampling History Buffer Status Flag
(Флаг состояния выборки данных)

При выборке данных, флаги состояния выборки данных (Bits 0 … 11) переходят в состояние ON. После завершения выборки данных они сбрасываются в OFF.

Регистр состояния очистки буфера архива (HCSR)

Соотвествующая область памяти	Соответствующий флаг		
	Bit	Двоичный вид (x)	Функция
Область памяти 1	0	0000 0000 0000 000x	Флаг очистки буфер архива 1
Область памяти 2	1	0000 0000 0000 00x0	Флаг очистки буфер архива 2

Соотвествующая область памяти	Соответствующий флаг		
	Bit	Двоичный вид (x)	Функция
Область памяти 3	2	0000 0000 0000 0x00	Флаг очистки буфер архива 3
Область памяти 4	3	0000 0000 0000 x000	Флаг очистки буфер архива 4
Область памяти 5	4	0000 0000 000x 0000	Флаг очистки буфер архива 5
Область памяти 6	5	0000 0000 00x0 0000	Флаг очистки буфер архива 6
Область памяти 7	6	0000 0000 0x00 0000	Флаг очистки буфер архива 7
Область памяти 8	7	0000 0000 x000 0000	Флаг очистки буфер архива 8
Область памяти 9	8	0000 000x 0000 0000	Флаг очистки буфер архива 9
Область памяти 10	9	0000 00x0 0000 0000	Флаг очистки буфер архива 10
Область памяти 11	10	0000 0x00 0000 0000	Флаг очистки буфер архива 11
Область памяти 12	11	0000 x000 0000 0000	Флаг очистки буфер архива 12
	12-15	xxxx 0000 0000 0000	Зарезервированы

■ Очистка флага состояния буфера событий

При очистке буфера событий, флаг состояния буфера событий устанавливается в ON (Bits 0 … 11 устанавливаются в состояние ON). После окончания очистки он переключается в состояние OFF.

Регистр состояния рецептов (RESR)

Бит	Двоичный вид (x)	Функция
0	0000 0000 0000 000x	Состояние изменения номера рецепта
1	0000 0000 0000 00x0	Флаг состояния чтения рецепта (ПЛК → HMI)
2	0000 0000 0000 0x00	Флаг состояния записи рецепта (ПЛК ← HMI)
3	0000 0000 0000 x000	Флаг изменения состояния номера группы рецептов
4~7	0000 0000 xxxx 0000	Зарезервированы
8~15	xxxx xxxx 0000 0000	Флаг подтверждения изменения номера рецепта

■ Состояние изменения номера рецепта

При изменении номера рецепта, этот флаг устанавливается в состояние ON. После того, как номер рецепта изменён (его значение RCPNO изменено) он переключается в OFF.

■ Флаг состояния чтения рецептов

Когда HMI читает один рецепт из ПЛК, этот флаг устанавливается в состояние ON. После окончания чтения и запоминания прочитанных данных в HMI этот флаг переключается в состояние OFF.

■ Флаг состояния записи рецептов

Когда HMI записывает рецепт в ПЛК, этот флаг переходит в состояние ON. По окончанию записи в ПЛК этот флаг переключается в OFF.

■ Флаг изменения состояния номера группы рецептов

При изменении номера группы рецептов этот флаг переходит в состояние ON. Когда номер группы изменён и записан данные RCPG изменены, этот флаг переключается в состояние OFF.

■ Флаг подтверждения изменения номера рецепта

Вне зависимости от того, какое значение RCPG назначено ПЛК или HMI, когда меняется номер группы, этот флаг переходит в состояние ON. Таким образом система получае эхо сигнал о изменении номера группы регистров **Designate Recipe Group Number Register**.

Регистр состояния номера рецепт (RBSR)

Вне зависимости от того, какое значение RCPNO назначено ПЛК или HMI, когда меняется номер рецепта, этот флаг переходит в состояние ON. Таким образом, система имеет обратную связь - при изменении номера рецепта он обновляется в регистре назначенных номеров рецептов (Designate Recipe Number Register).

Регистр состояния 2 для общего контроля (GCSR2)

Бит	Двоичный вид (x)	Функция
0-7	0000 0000 xxxx xxxx	Статус уставки многоязыковой поддержки
8	0000 000x 0000 0000	Флаг состояния печати
9	0000 00x0 0000 0000	Флаг состояния формы подачи принтера
10-15	xxxx xx00 0000 0000	Зарезервированы

■ Статус уставки многоязыковой поддержки

Bits 0 … 7 определяют значение номера языка HMI.

■ **Флаг состояния печати**

Когда флаг имеет состояние ON, это показывает, что идёт печать текущего экрана, когда этот флаг переключается в состояние OFF, печать заблокирована.

■ **Флаг состояния формы подачи принтера**
Printer Form Feed Status Flag

Когда флаг имеет состояние ON, это показывает, что обеспечивает автоподачу бумаги, когда этот флаг переключается в состояние OFF, автоподача заблокирована.

3.5 Элементы меню (File)

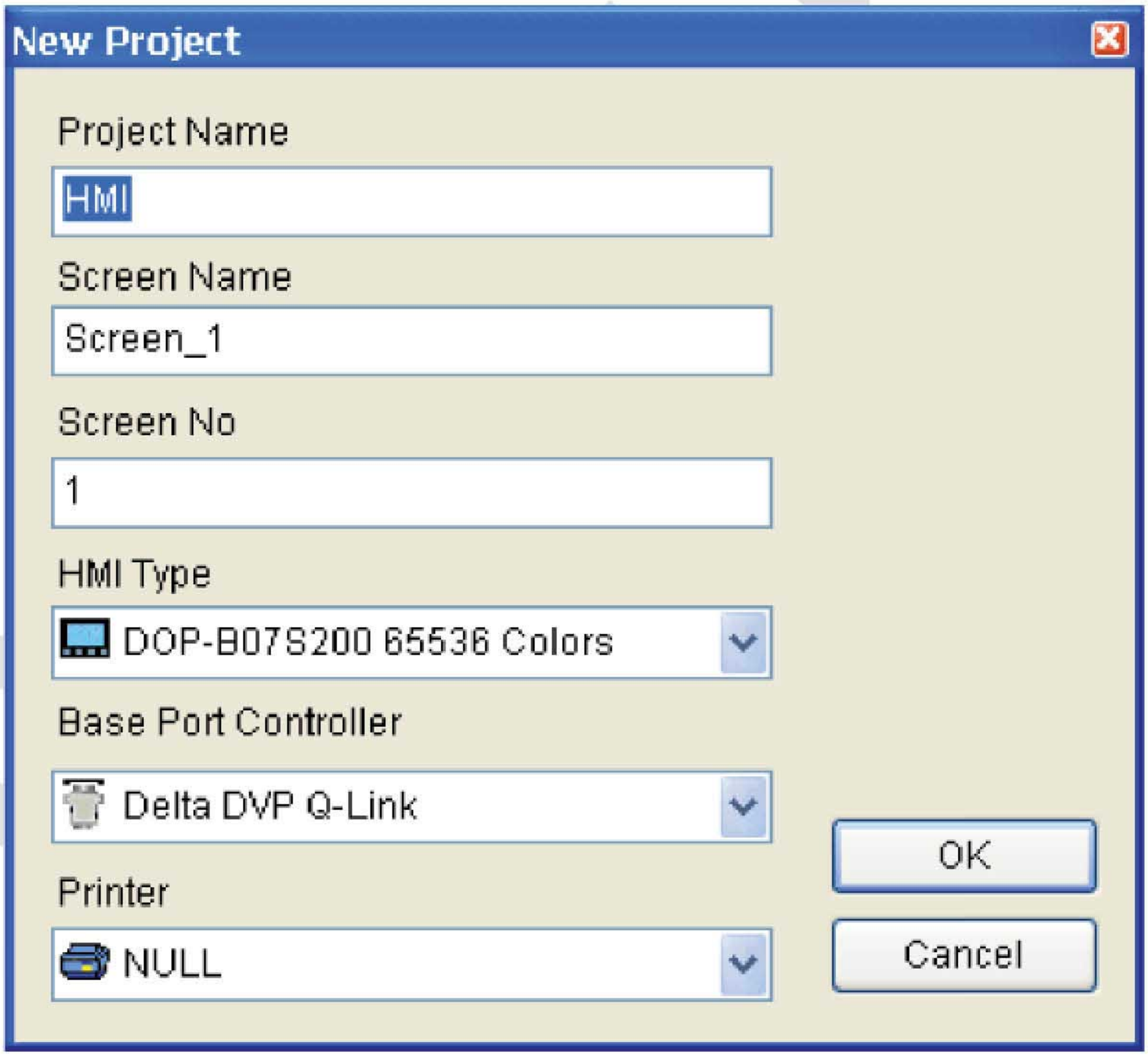


Рис.3.5.1 Новый проект. Открыть новый проект

Для открытия нового проекта выбрать **File > New** (Рис. 3.5.1) или кликнуть на

иконку **New** на стандартной панели инструментов (Рис. 3.5.1.), или нажать **Ctrl + N**.

Если при первом включении нет старой программы, то при открытии нового проекта в диалоговом окне будет предложено ввести имя программы, на экране экрана, номер первого экрана, выбрать модель HMI и тип связи с внешними устройствами (соответствующий драйвер для связи с выбранным контроллером).

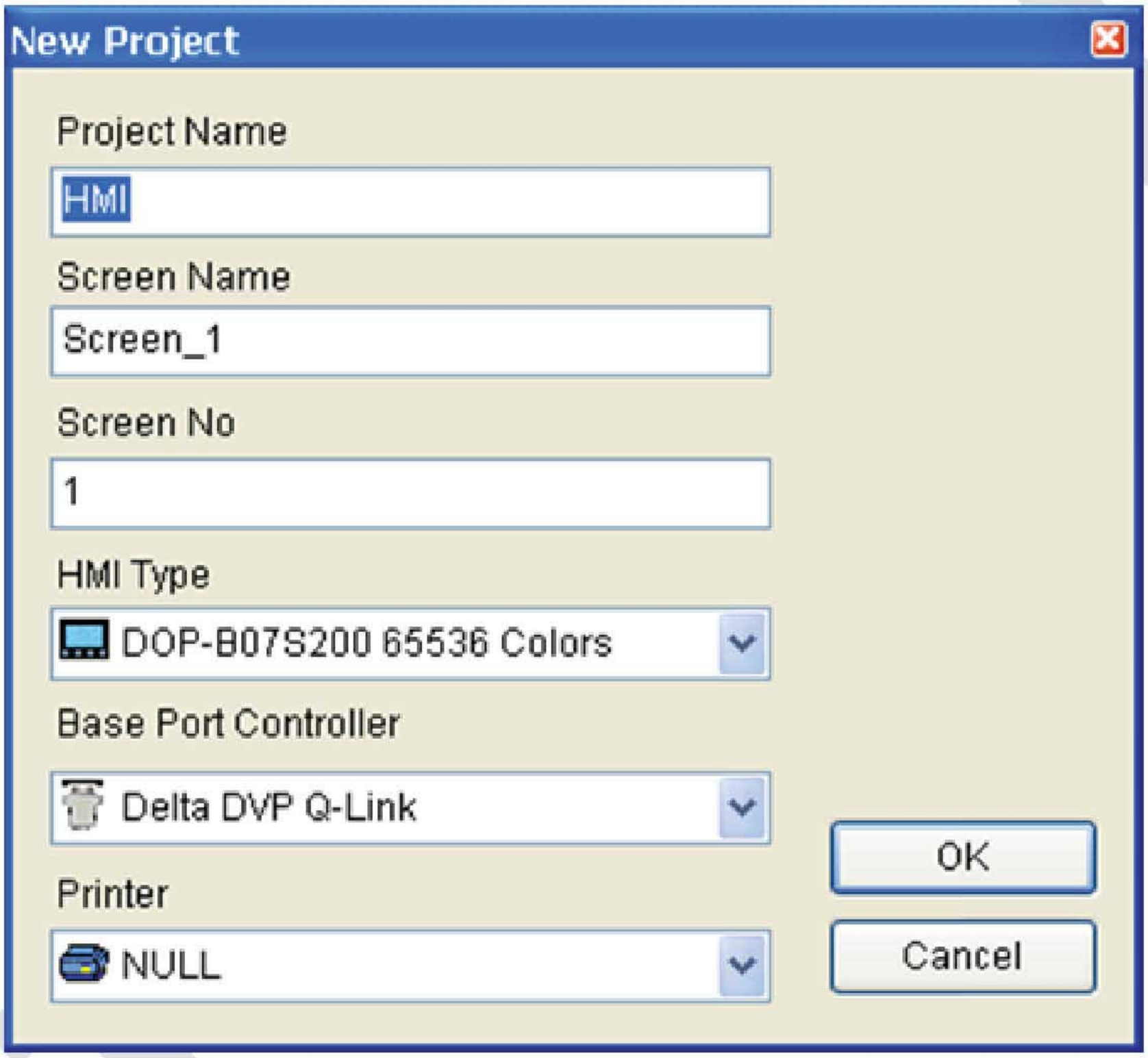


Рис. 3-5-1 Диалоговое окно создания новой прикладной программы

Если в момент создания новой прикладной программы открыт какой-либо файл, то будет предложено его сохранить (Рис. 3.5.2).

Нажмите кнопку **Yes** для сохранения и закрытия текущего файла. Нажав кнопку **No**, текущий файл будет закрыт без сохранения.

Кнопка **Cancel** отменит действие по созданию новой прикладной программы. После нажатия кнопок **Yes** или **No** откроется диалоговое окно (Рис. 3.5.1).

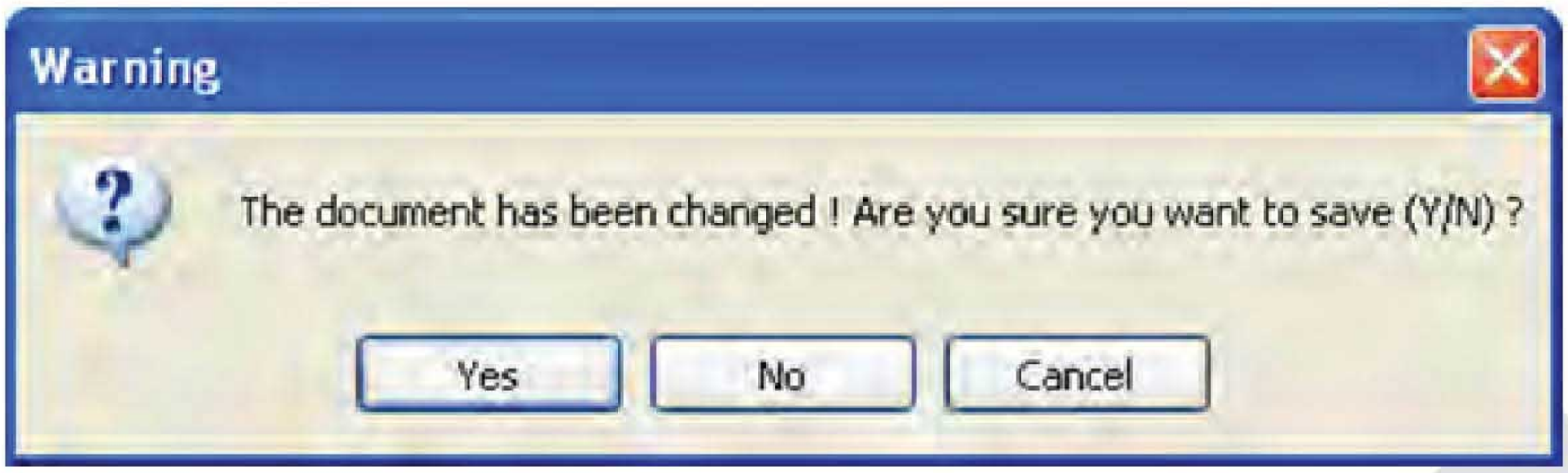


Рис.3-5-2 Диалоговое окно сохранения программы

Введите наименование проекта, имя первого экрана, выберите модель панели, тип связи с внешними устройствами (соответствующий драйвер для связи с выбранным контроллером). (Рис.3-5-3), и нажмите кнопку **OK**.

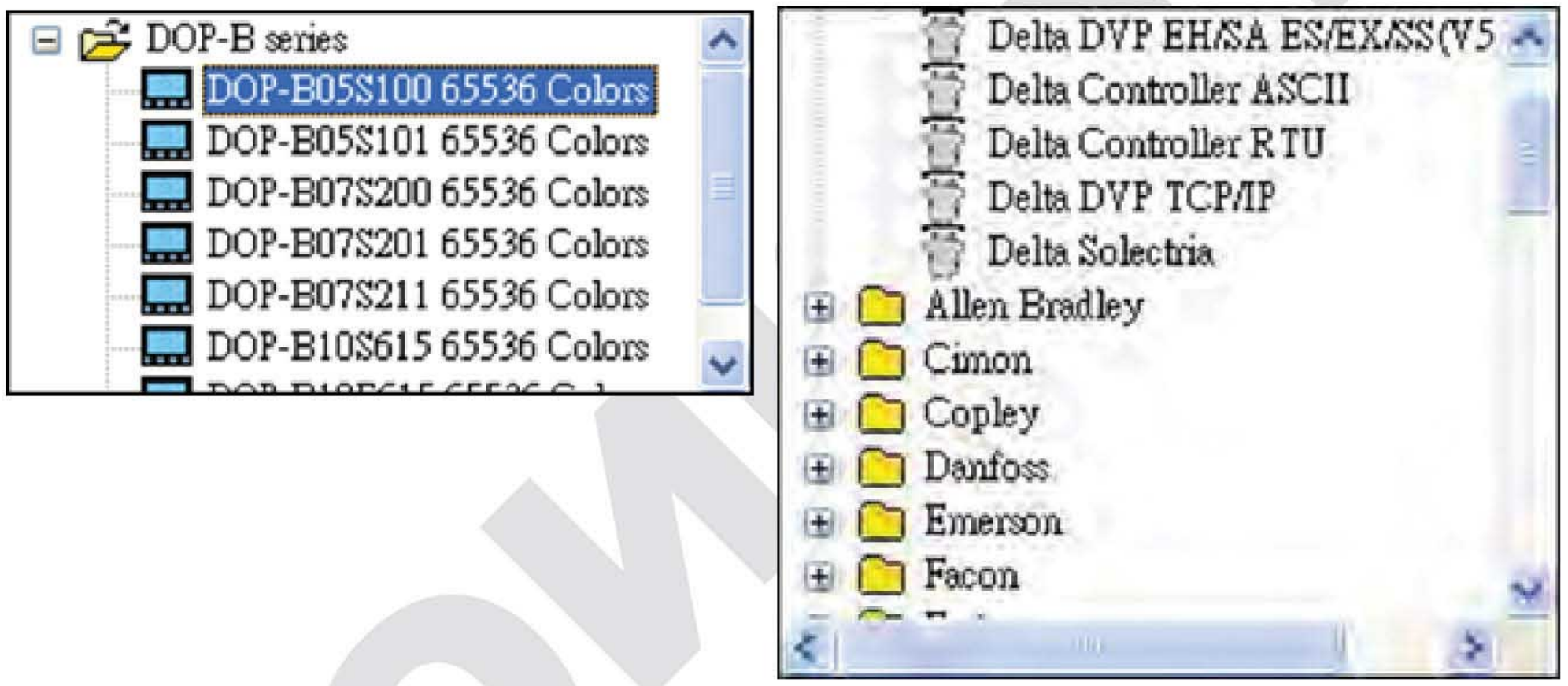


Рис.3-5-3 Выбор модели панели оператора и драйвер связи

3.5.2 Open Открыть старую прикладную программу.

Для открытия прикладной программы выбрать **File > Open** (Рис. 3-5-4) или кликнуть иконку или нажать **Ctrl + O**.

Если будут открыты другие программы, то перед открытием старой программы, пользователь получит предложение сохранить их (Рис. 3-5-2), а затем в открывшемся окне сообщение открыть выбранный файл .dop (Рис. 3-5-4).

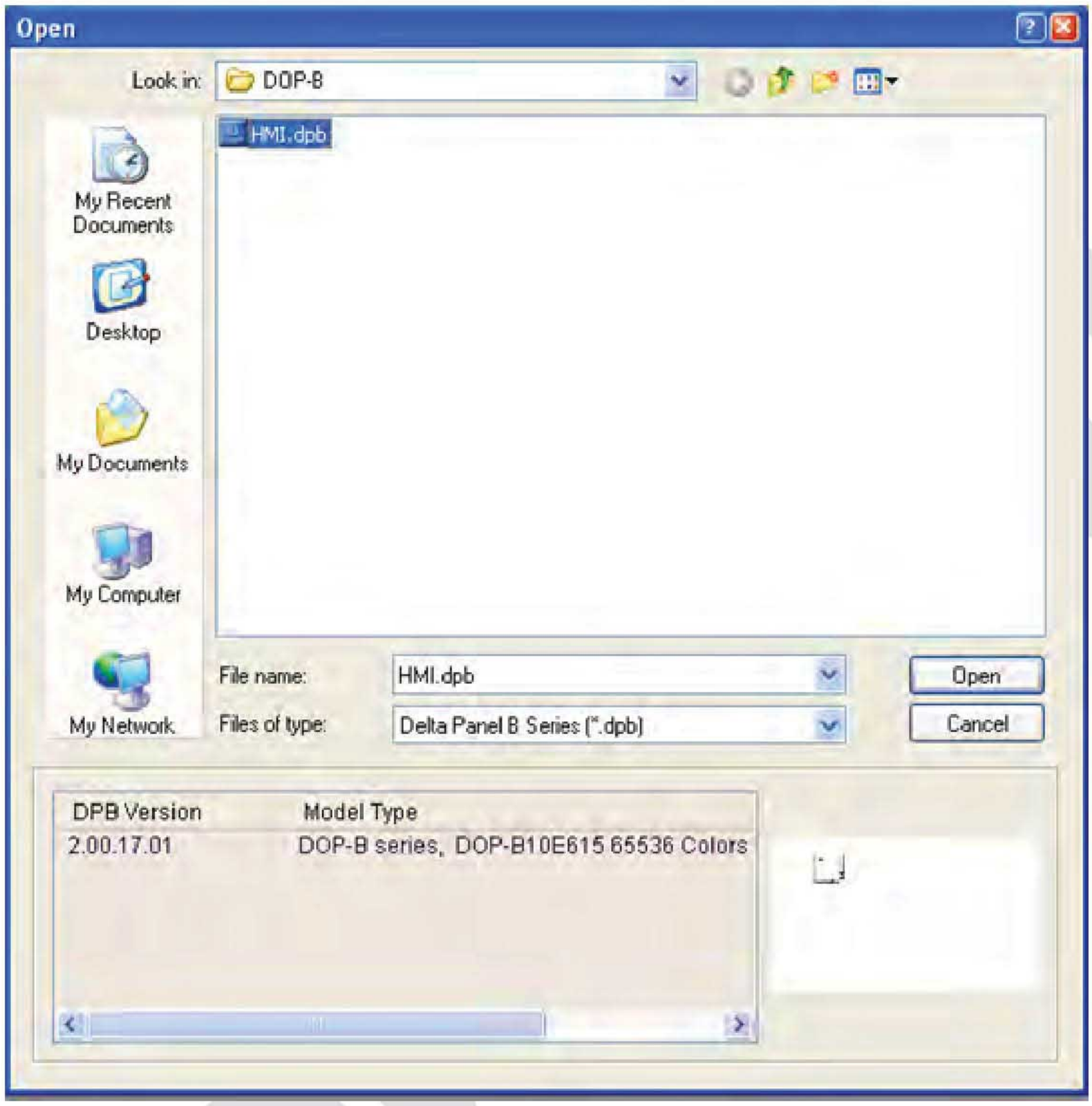


Рис. 3-5-4 Открытие старой прикладной программы

3.5.3 Закрытие программы (Close)

Для закрытия программы выберите **File > Close**. Если до выполнения данной команды текущая прикладная команда не была сохранена, то будет предложено её сохранить (Рис. 3-5-5), нажав клавишу **Yes** .

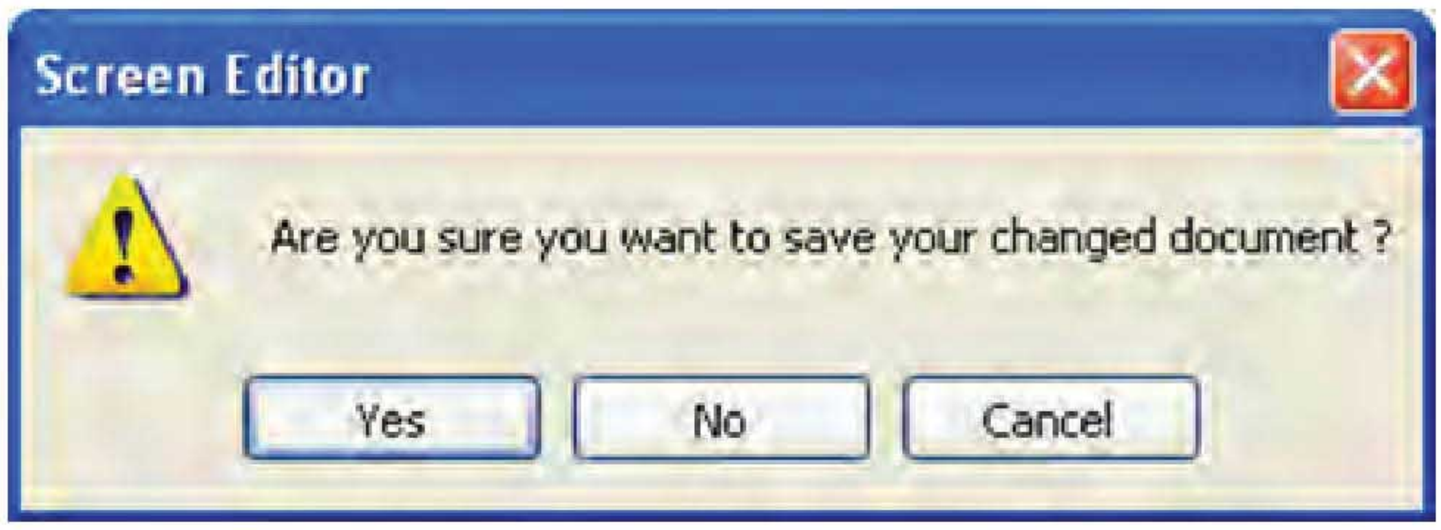


Рис. 3-5-5 Предложение сохранить текущий файл перед его закрытием

3.5.4 Сохранение (Save)

Выберите **File > Save** или значок  на стандартной панели инструментов или используйте горячие клавиши **Ctrl + S**. Если данная программа сохраняется в первые, то откроется диалоговое окно (Рис. 3-5-6) с предложением выбрать имя и место сохраняемого файла. Если сохранение текущей программы ранее выполнялось, то по данной команде она будет немедленно сохранена под старым именем в том же месте жесткого диска ПК.

3.5.5 Сохранение под другим именем (Save as)

Для сохранения текущей программы выберите **File > Save As**. Откроется диалоговое окно (Рис. 3-5-6) с предложением выбрать имя и место сохраняемого файла с расширением .dpb. Это окно появиться автоматически, если прикладная программа сохраняется впервые (командой Save).

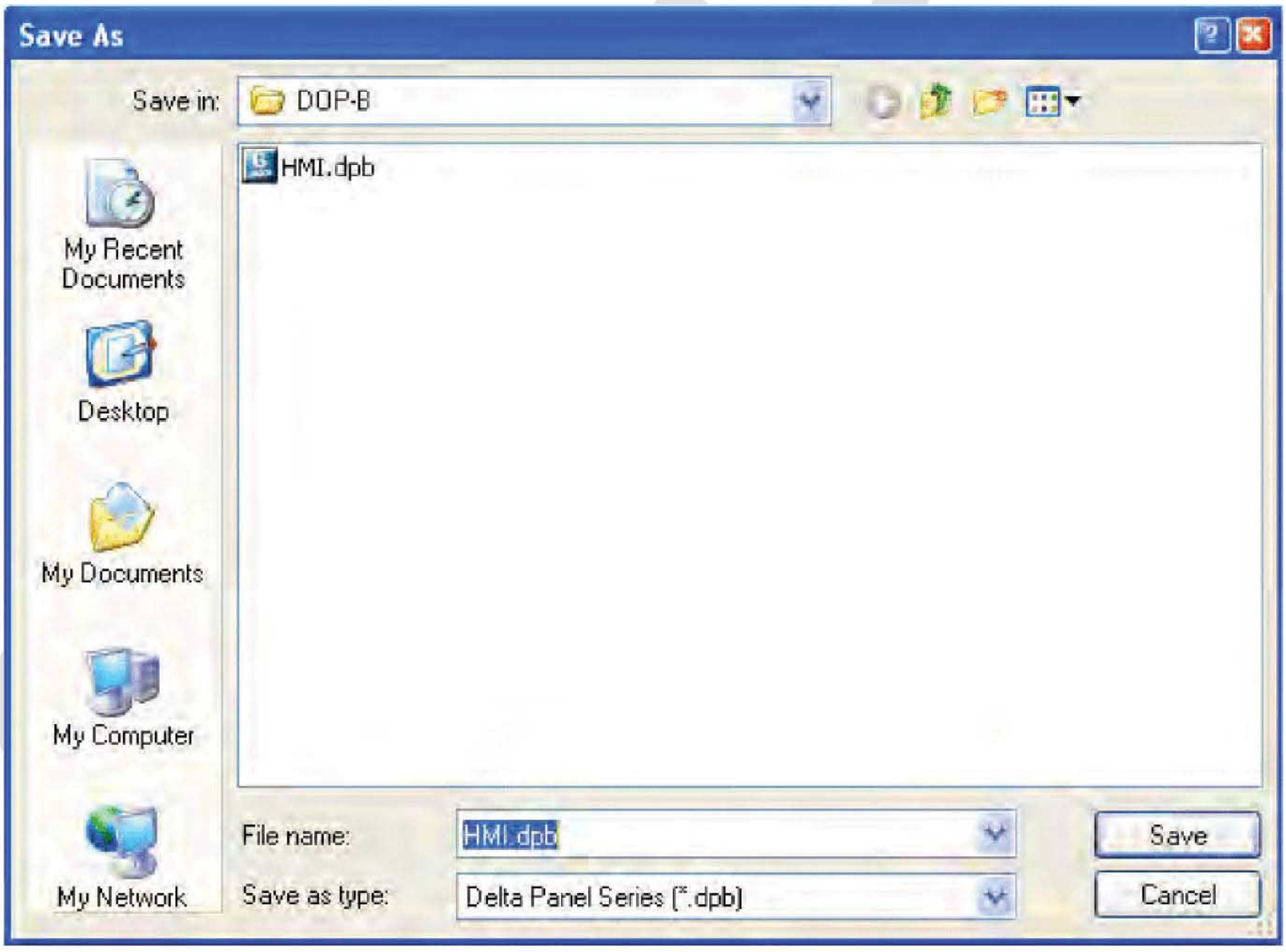


Рис. 3-5-6 Диалоговое окно сохранения

3.5.6 Запись прикладной программы на внешнюю память

Перед использованием данной команды прикладная программа должна быть откомпилирована. Иначе команда не выполнится, а на экране ПК появиться сообщение об ошибке (Рис. 3-5-7). Выполните компиляцию и выберите **File > Make Ext. Memory Data** для копирования текущей прикладной программы на SD карту или USB диск (Рис. 3-5-8).

Если SD карта или USB disk с записанной на неё прикладной программой будет вставлена в HMI, то при подаче питания на панель все данные будут читаться напрямую с SD карты или USB диска.

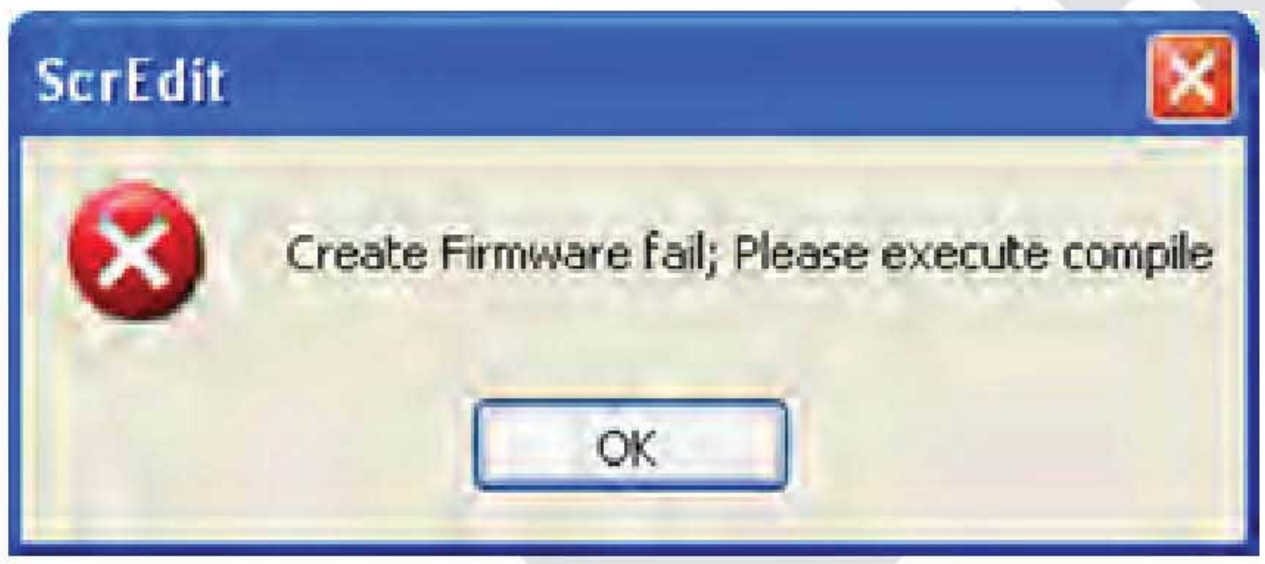


Рис. 3-5-7 Сообщение об ошибке при попытке записать на внешнюю память нескопилированную прикладную программу

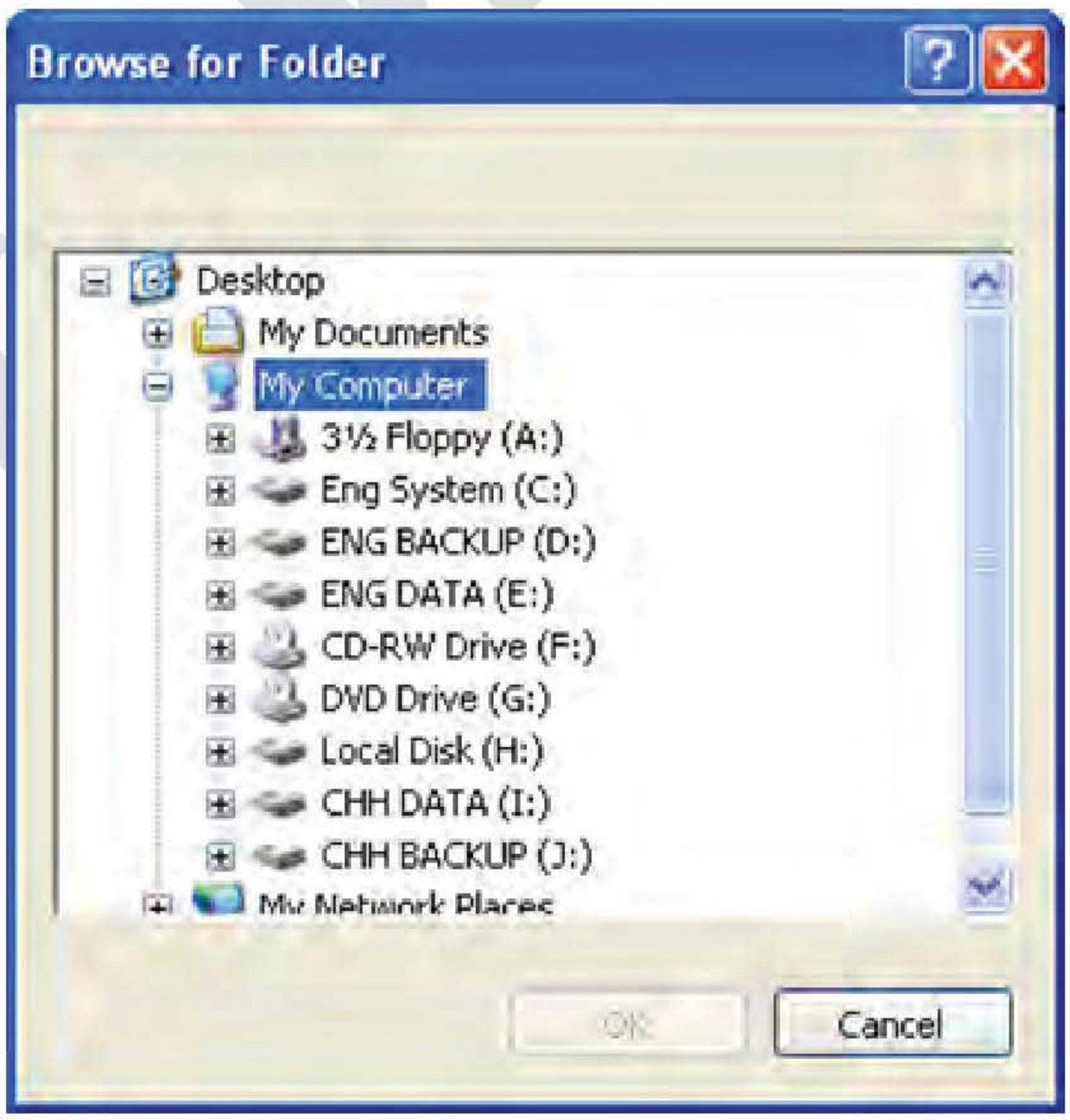


Рис. 3-5-8 Диалоговое окно сохранения

3.5.7 Open Ext. Memory Data

Для открытия внешней памяти, выберите **File > Open Ext. Memory Data** или нажмите **Ctrl+I**. Откроется диалоговое окно и пользователь выбирает требуемый файл.

3.5.8 Защита прикладной программы паролем (Password Protect)

Пользователь может разрешить или запретить функцию защиты паролем прикладной программы выбрав **File > Password protect** (Рис. 3-5-9 & Рис. 3-5-10) Появится диалоговое окно, в котором надо будет подтвердить разрешение функции пароля и рядом с командой **Password Protect** будет установлен символ ☑, означающий что данный файл защищен паролем, без которого его в дальнейшем невозможно будет открыть. Сам пароль должен быть задан в меню **Option > Workstation Setup** (Рис. 3-5-11). При запрещении функции защиты паролем, появится диалоговое окно, показанное на Рис. 3-5-10

Рис. 3-5-9 Функция защиты паролем включена

Рис. 3-5-10 Функция защиты паролем отключена

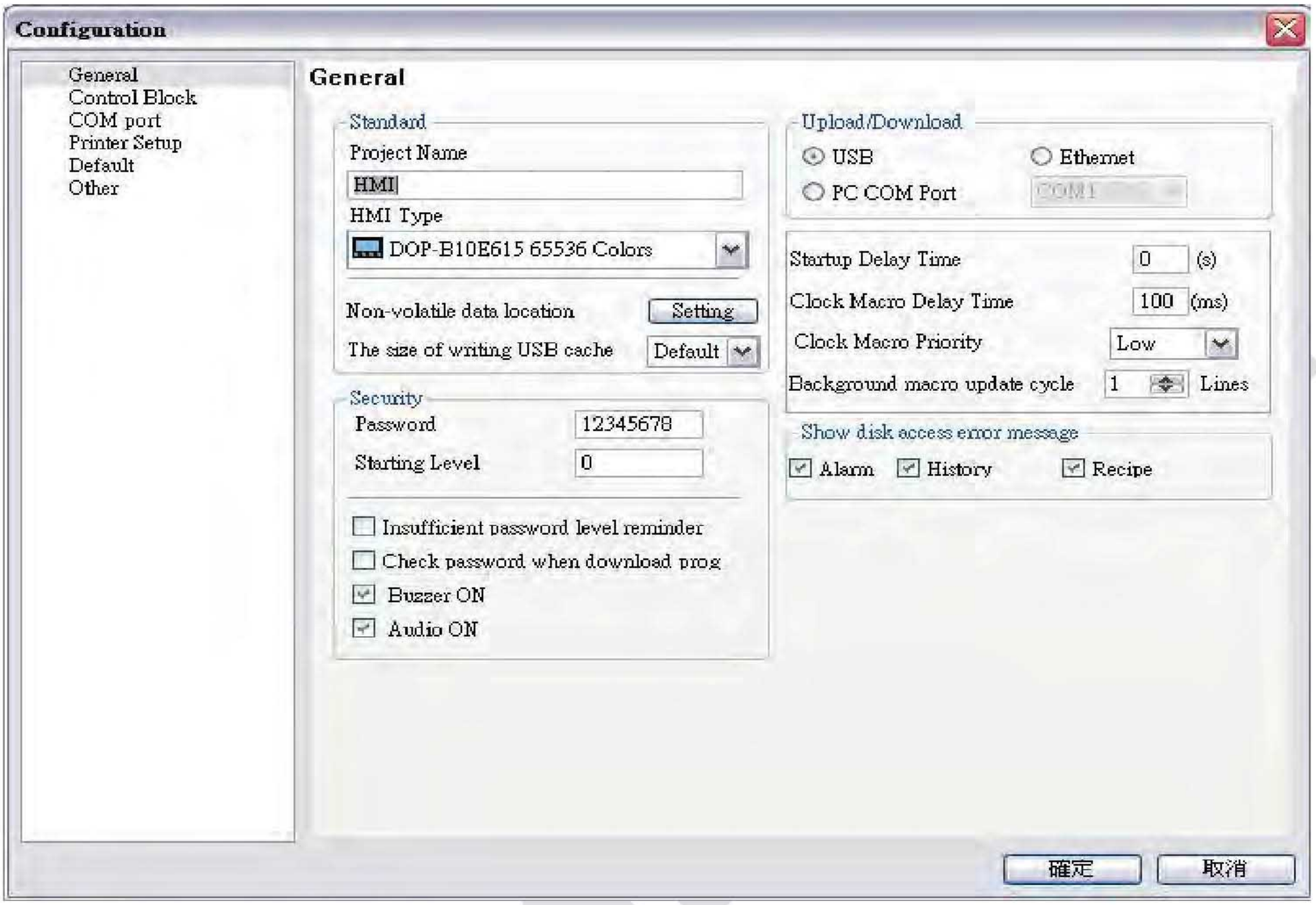


Рис. 3-5-11 Установка пароля –Security–

3.5.9 Print - печать

Для печати текущего экрана выберите **File > Print**, или нажмите на иконку  на панели инструментов, или нажмите **Ctrl + P.**

3.5.10 Предварительный просмотр печатаемых изображений (Print Preview)

Для предварительного просмотра изображения перед посылкой на принтер, выберите **File > Print Preview** (Рис. 3-5-12).

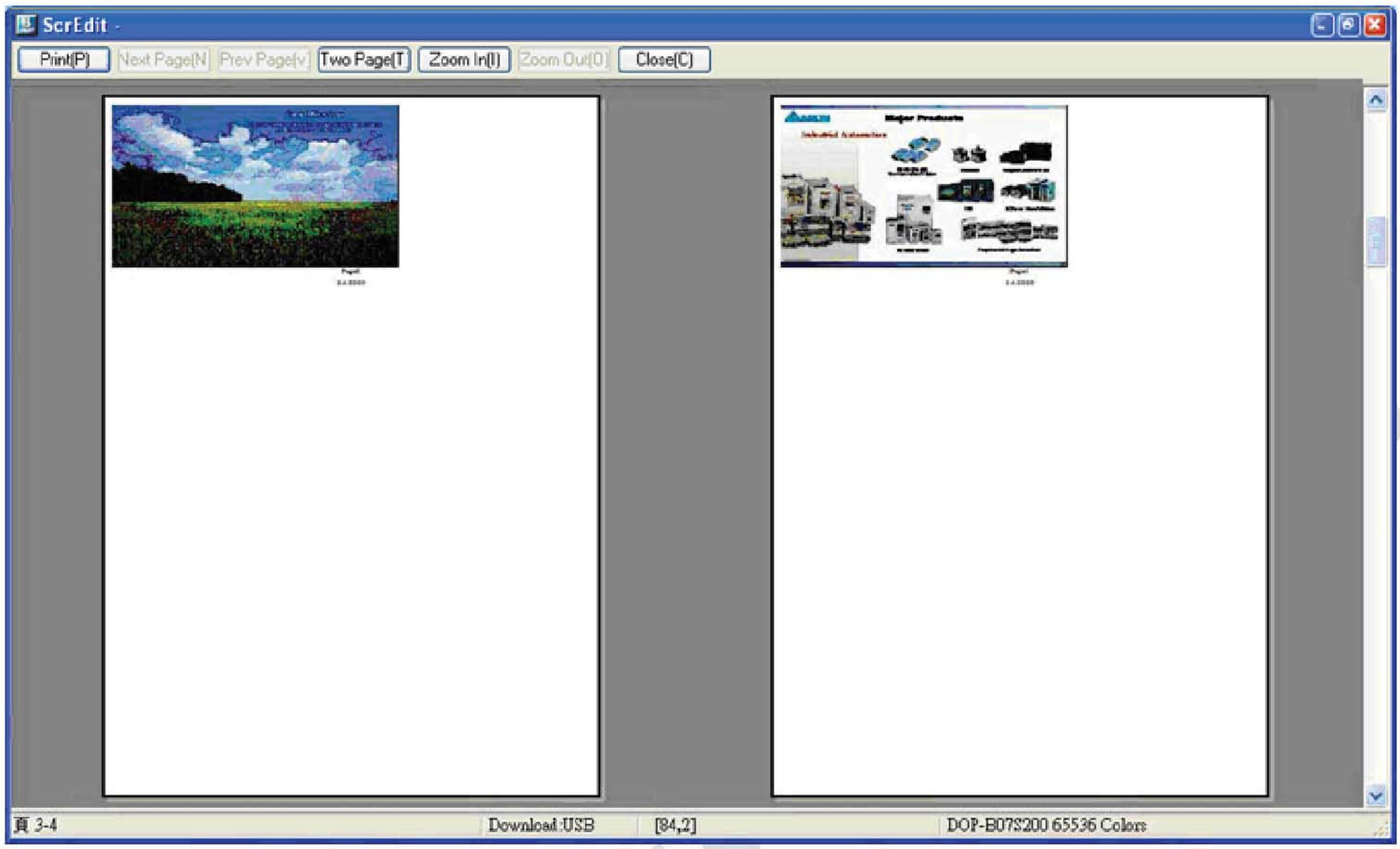


Рис. 3-5-12 Диалоговое окно предварительного просмотра печати

3.5.11 Выбор принтера и настройка параметров печати (Print Setup)

Для выбора принтера и настройки печати, выберите **File > Print Setup** (Рис. 3-5-13).

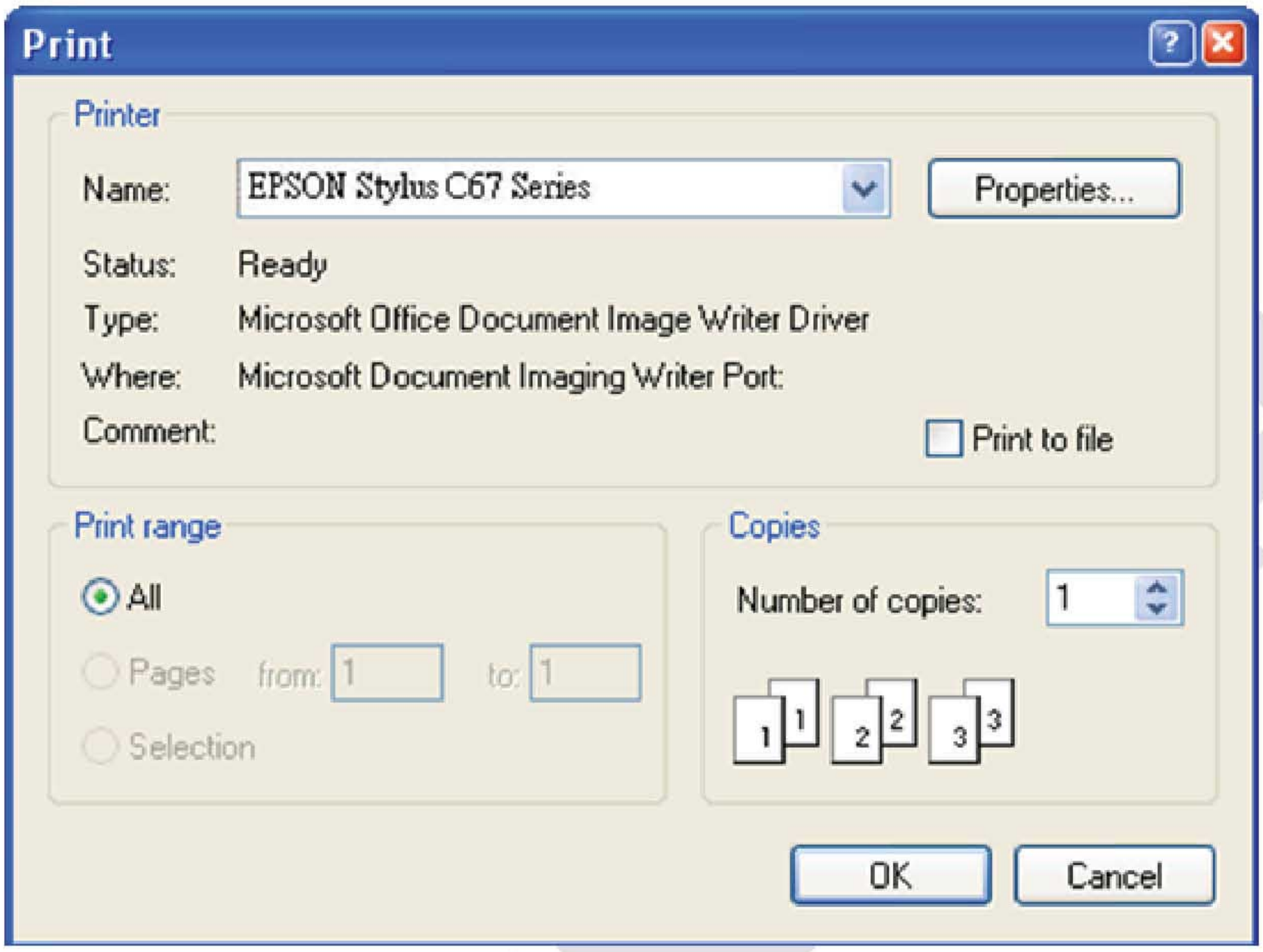


Рис. 3-5-13 Настройки печати

3.5.12 Быстрый доступ к файлам (File Quick Access)

По умолчанию, внизу выпадающего меню **File** предоставлен лист из четырех последних открываемых файлов для быстрого доступа к ним (Рис. 3-5-14). Кликнув мышкой на названии данного файла, можно быстро его открыть. Эта функция аналогична команде **Open**, описание которой находится на странице 3-43. Если путь сохранения файла очень длинный, то будет отображаться многоточие “….”.

Пользователь сможет увидеть полное имя .dpb файла.

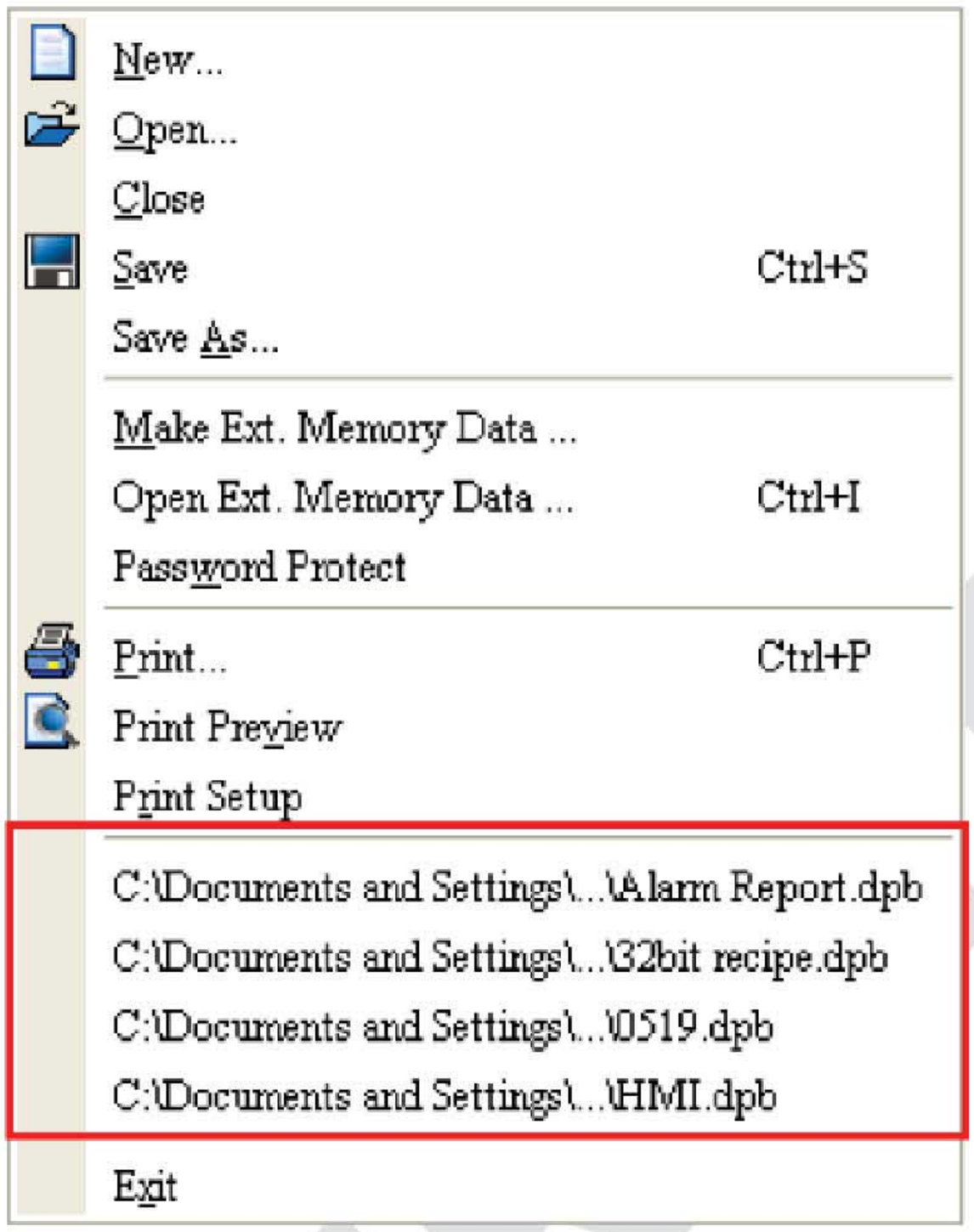


Рис. 3-5-14 Последние открываемые файлы

3.5.13 Закрытие программы (Exit)

Эта команда закрывает все открытые в данный момент прикладные программы, предварительно предложив их сохранить, и полностью закрывает программу ScrEdit. Выберите **File > Exit**. Если текущий файл не был предварительно сохранен, откроется диалоговое окно (Рис. 3-5-2)

Эта команда закрывает все открытые в данный момент прикладные программы, предварительно предложив их сохранить, и полностью закрывает программу ScrEdit. Выберите **File > Exit.** Если текущий файл не был предварительно сохранен, откроется диалоговое окно (Рис. 3-5-4) с предложением сохранить его на жесткий диск ПК.

При нажатии на кнопку **Cancel** команда **Exit** будет отменена.

При нажатии на кнопку **Yes** текущий файл будет сохранен и программа ScrEdit закрыта. Нажав кнопку **No**, текущий файл будет закрыт без сохранения. После нажатия кнопки **Yes** откроется диалоговое окно (Рис. 3-5-4) для выбора пути сохранения файла.

3.6 Элементы меню (Edit)

Выпадающее меню выполнено в стиле Microsoft Office

3.6.1 Undo - отменить последнее действие

Для отмены последнего действия выберите **Edit > Undo** или значок  на стандартной панели инструментов (Рис.2.4.3), или используйте горячие клавиши **Ctrl + Z**. Все действия регистрируются в выходном окне.

3.6.2 Redo – отмена действия команды Undo

Для отмены действия команды undo выберите **Edit > Redo** или значок  на стандартной панели инструментов или используйте горячие клавиши **Ctrl + Y**. Все

действия регистрируются в выходном окне.

3.6.3 Cut – вырезать

Позволяет удалить выбранный компонент на редактируемом экране и сохранить его в буфере обмена для вставки в другое место. Выберите **Edit > Cut** или значок  на стандартной панели инструментов или используйте горячие клавиши **Ctrl + X**.

3.6.4 Copy - копировать

Позволяет скопировать выбранный компонент на редактируемом экране в буфер обмена для вставки в другое место. Выберите **Edit > Copy** или значок  на стандартной панели инструментов), или используйте горячие клавиши **Ctrl + C**.

3.6.5 Paste - вставить

Позволяет вставить компонент на редактируемой странице из буфера обмена. Выберите **Edit > Paste** или значок  на стандартной панели инструментов или используйте горячие клавиши **Ctrl + V**.

3.6.6 Delete - удалить

Позволяет удалить выбранный компонент. Выберите **Edit > Delete** или используйте клавишу **Del** на клавиатуре ПК.

3.6.7 Select All – выбрать все

Используется для выбора всех объектов на редактируемом экране. Выберите **Edit > Select All** или используйте горячие клавиши **Ctrl + A**. Когда выбраны все объекты, объект, расположенный в левом верхнем углу, будет выделен синей рамкой, как базовый элемент, остальные объекты будут выделены белой рамкой.

Базовый элемент используется для выравнивания и изменения размеров.

3.6.8 Find content – найти содержимое

Используется для поиска различных компонентов прикладной программы

по заданным критериям. Выберите **Edit > Find content** или используйте горячие клавиши **Ctrl + F**. Критериями поиска могут быть текст, адрес чтения, адрес записи или адрес памяти на текущем экране, или на всех экранах (Рис. 3-6-1). (Результаты поиска будут отображены в выходном окне, “кликните” мышкой по объекту для перехода к данному объекту на экране в ScrEdit (Рис. 3-6-2)

Find what (Найти что:)

В этом окне пользователь пишет что найти: слово или фразу.

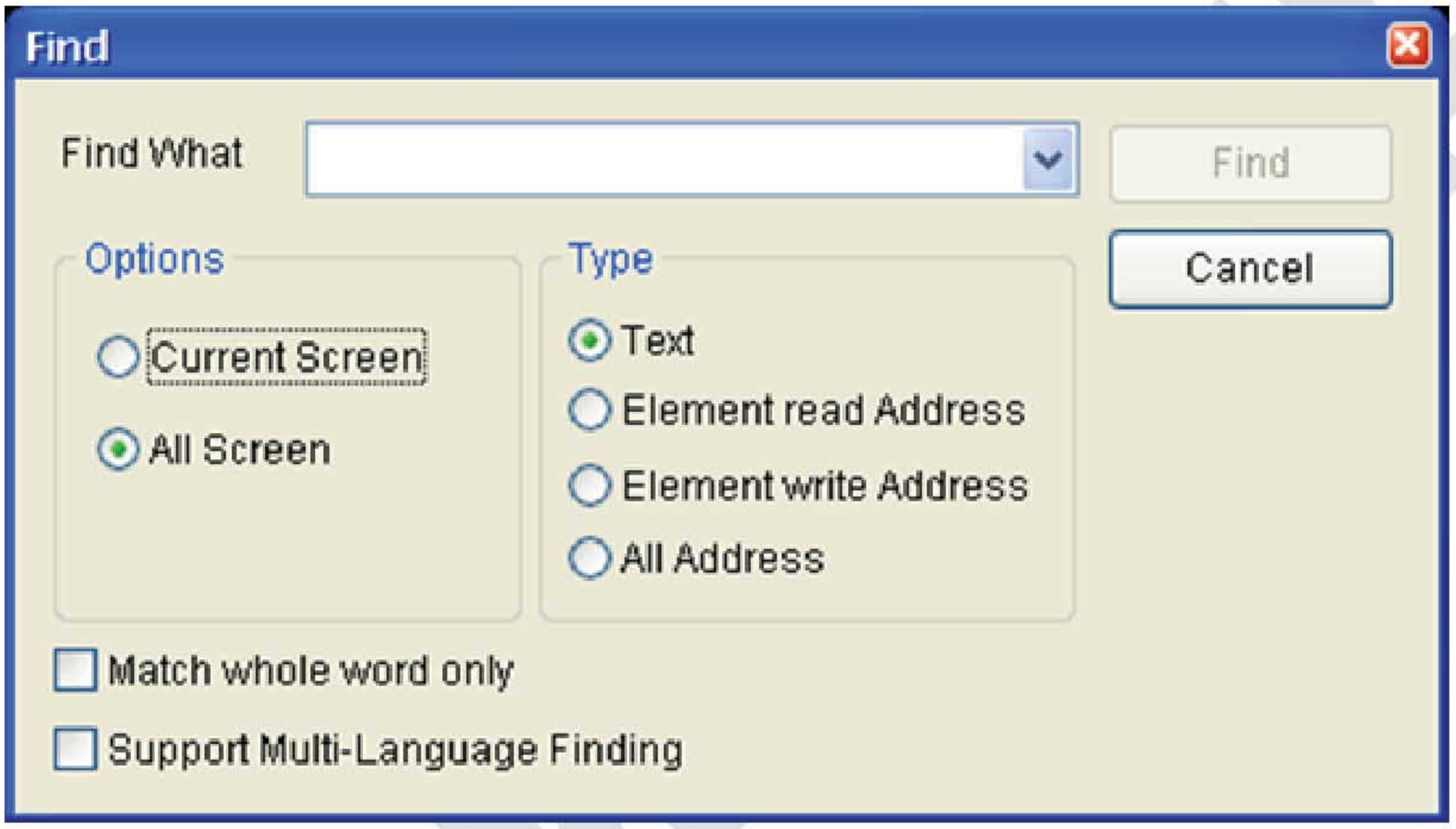


Рис. 3-6-1 диалоговое окно поиска содержимого

Options:

Current Screen (на текущем экране)

Поиск выполняется только на текущем экране. Результаты поиска с координатами расположения найденного объекта будут отображены в выходном окне (output window), дважды “кликнув” по которым мышкой будет совершен переход к данному объекту на экране в ScrEdit. (См. Рис. 2.4.15)

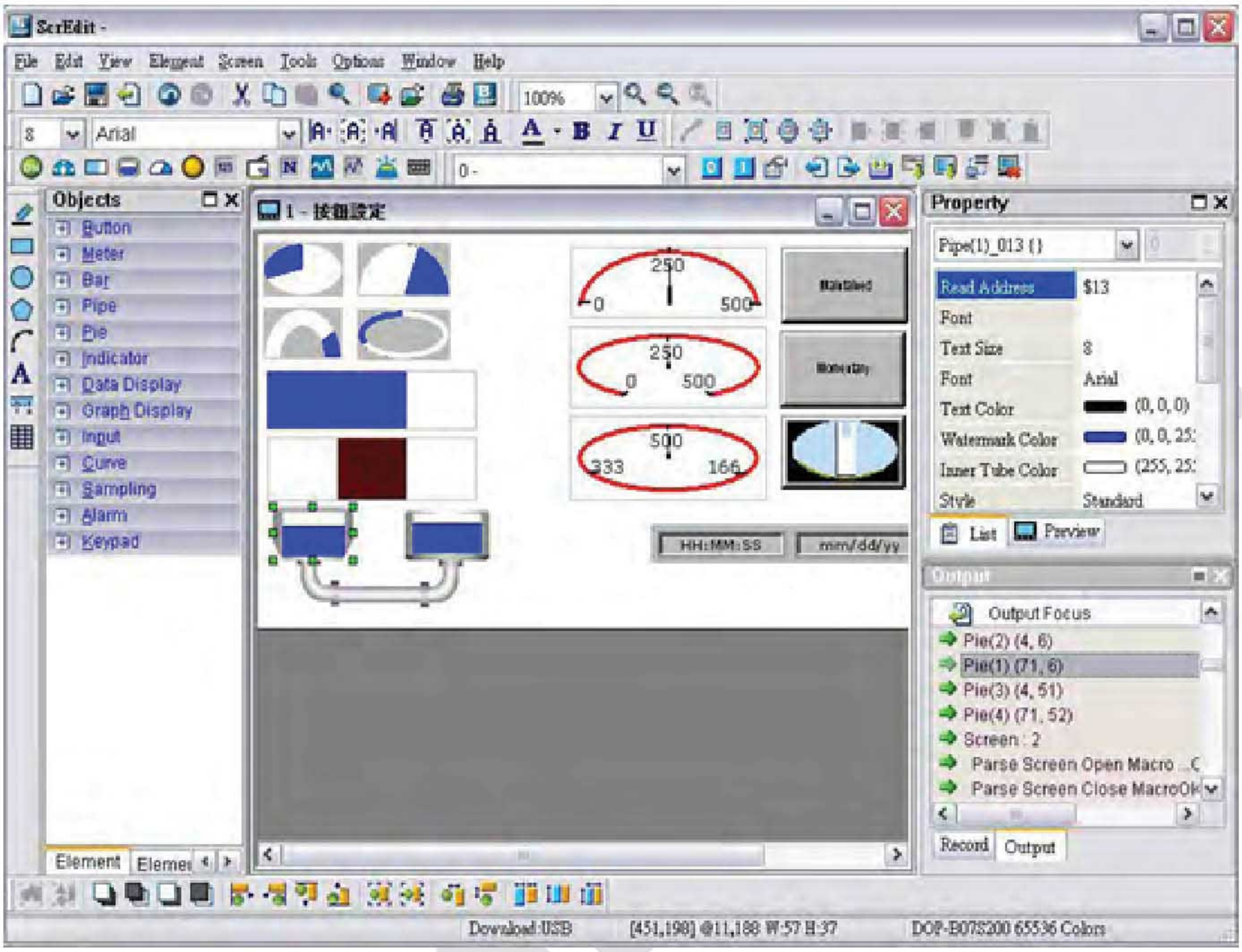


Рис. 3.6.2

All Screen (на всех экранах)

Поиск выполняется на всех экранах ScrEdit. Результаты поиска с координатами расположения найденного объекта будут отображены в выходном окне (output window), дважды “кликнув” по которым мышкой будет совершен переход к соответствующему экрану и данному объекту на нем.

Type:

Text

Поиск объектов в прикладной программе по тексту (только введенному пользователем).

Read Address

Поиск объектов в прикладной программе по адресу чтения (read address).

Write Address

Поиск объектов в прикладной программе по адресу записи (write address).

All Address

Поиск объектов в прикладной программе по адресу чтения и записи.

Check Box:

Match whole word only

Система находит только слова или фразы, которые введены пользователем. Если пользователь не выберет эту функцию, то система выберет всё содержимое, в котором содержатся эти введённые им слова или фразы.

Multi language Finding

Система находит все содержимое многязычного текста в котором содержатся введённые им слова или фразы.

3.6.9 Replace - заменить

Для замены контента, совпадающего с заданными критериями, выберите **Edit > Replace** или используйте горячие клавиши **Ctrl + R**.

Используется для поиска различных компонентов прикладной программы по заданным критериям и замены их на другие. Команда может применяться для замены текста, адреса чтения, адреса записи на текущем экране, или на всех экранах. Эта функция аналогична команде **Find**, только к функции поиска добавлена функция замена найденных компонентов на другие. Тип заменяемых данных может быть: бит, слово, двойное слово.

С помощью этой функции можно менять данные в разделах: Элемент, Макрос, Блок управления, Блок состояния, Буфер событий, Авария и Рецепт.

При выборе адресов чтения и записи пользователь может выбрать место где найти и заменить содержимое, соответствующее заданным критериям замены.

Replace

Find What

В этом поле надо ввести текс или адрес, которые необходимо будет найти.

Replace With

В этом поле надо ввести текс или адрес, на которые надо будет заменить компоненты, заданные в поле поиска (Find What)

Options

Current Screen

Поиск и замена выполняются только на текущем экране.

All Screen

Поиск и замена выполняются на всех экранах ScrEdit.

Type

Text

Поиск объектов в прикладной программе по тексту (только введенному пользователем).

Read Address

Поиск объектов в прикладной программе по адресу чтения (read address).

Write Address

Поиск объектов в прикладной программе по адресу записи (write address).

Data Type

Bit, WORD, DWORD

Здесь выбирается тип заменяемых данных (бит, слово или двойное слово). Функция активна только при замене адресов.

Filter (Replace Criteria)

Фильтр заменяемого объекта.

При выборе кнопки читать или писать адрес необходимо выбрать критерий замены - тип заменяемых данных: Element, Macro, Control Block, Status Block, History Buffer, Alarm и Recipe.

Replace / Replace All

При нажатии на кнопку **Replace** замену каждого найденного элемента необходимо будет подтверждать или пропускать. Например, для замены адреса $157 на $158, введите $157 в поле **Find What**, и $158 в поле **Replace With** и нажмите кнопку **Replace**. После нахождения адреса появится окно ScrEdit в котором надо будет нажать кнопку **Yes** для подтверждения замены.

При нажатии на кнопку **Replace All** будет произведена автоматическая замена всех найденных элементов.

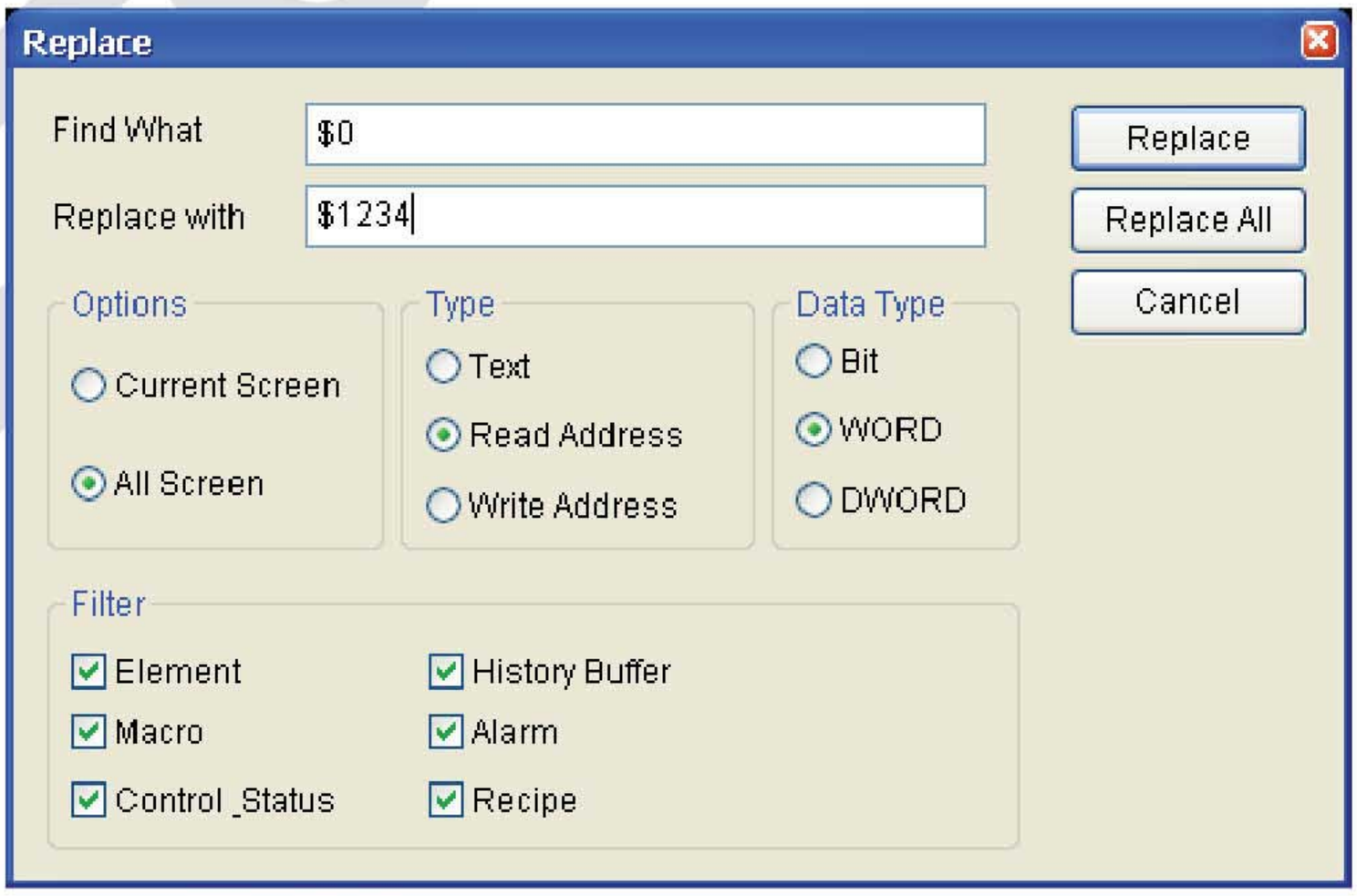


Рис. 3-6-4 Диалоговое окно Replace

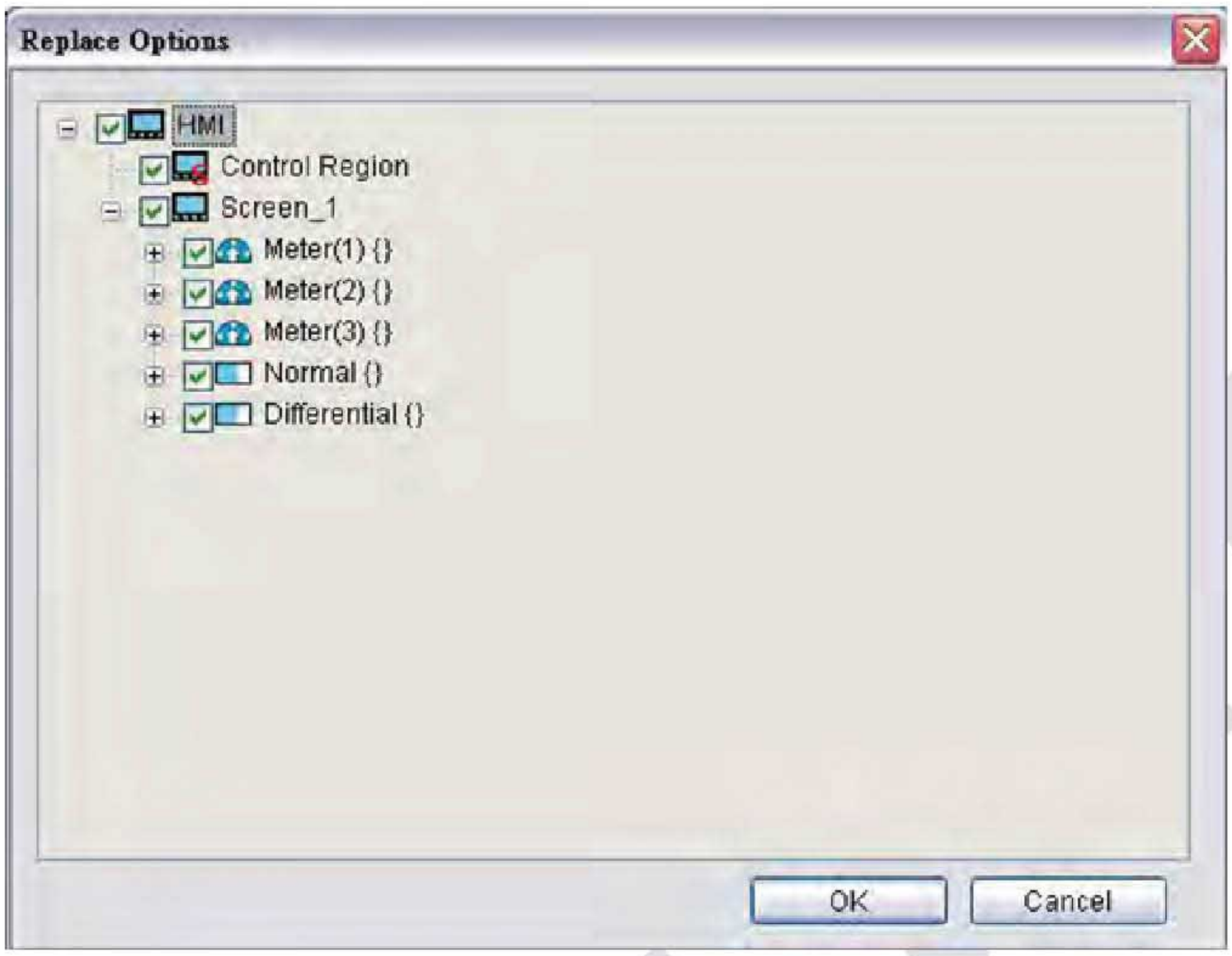


Рис. 3-6-5

3.6.10 Station Replace - замена адреса устройства

Эта функция обеспечивает замену в используемых элементах номера устройства (ПЛК адрес контроллера).

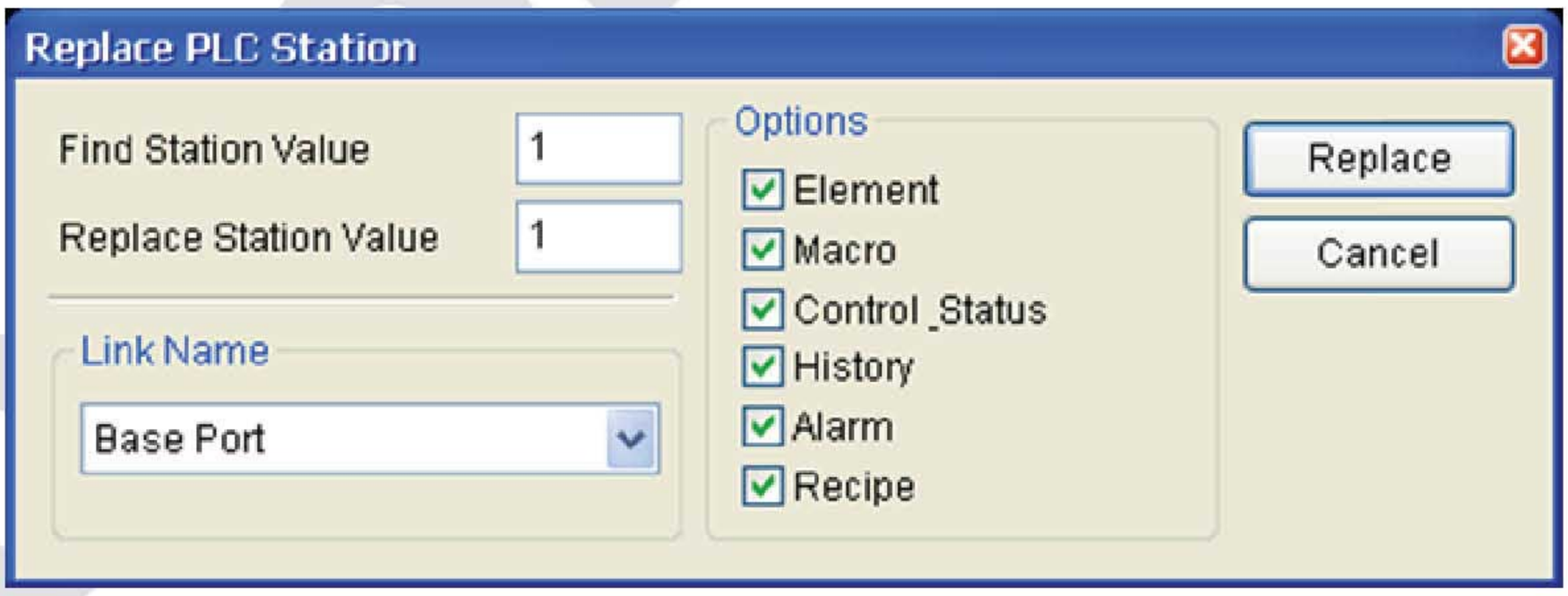


Рис. 3-6-6

3.6.11 Group - группировка элементов

Используется для группировки выбранных объектов на экране. Выберите **Edit > Group** (Рис. 2.4.17) или значок  на панели инструментов (Рис.2.4.18). Когда

два или более объектов объединены в группу, они будут вести себя как один объект при перемещении, изменении размеров и т.д.

3.6.12 Ungroup - разгруппировать

Используется для разгруппировки выбранных объектов на экране.

Выберите **Edit > Ungroup** или значок  на панели инструментов.

3.6.13 Order

Функция используется для определения порядка наложения объектов друг на друга. Выберите **Edit > Order** и значки  в меню или на панели инструментов.

 **Bring to Top**. Переместить выбранный объект в верхний слой, когда два или более объектов перекрыты.

 **Send to Bottom**. Переместить выбранный объект в нижний слой, когда два или более объектов перекрыты.

 **Bring Forward**. Переместить выбранный объект на одну позицию вперед.

 **Send Backward**. Переместить выбранный объект на одну позицию назад.

3.6.14 Align

Применяется для выравнивания расположения выбранных объектов на экране. Выберите **Edit > Align** (Рис. 2.4.23) и соответствующие значки в меню или на панели инструментов (Рис.2.4.24):

 : **Align Left**. выравнивание по левому краю объекта;

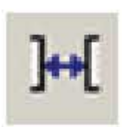
 : **Align Right**. выравнивание по правому краю объекта;

 : **Align Top**. выравнивание по верхнему краю объекта;

 : **Align Bottom**. выравнивание по нижнему краю объекта;

 : **Align Center Vertically**. вертикальное выравнивание по центру страницы;

 : **Align Center**. горизонтальное выравнивание по центру страницы;

 : **Across Space Evenly**. равномерно поперёк пространства;

 : **Down Space Evenly**. равномерно продольному пространству.

Команды **Align Left**, **Align Right**, **Align Top** и **Align Bottom** возможны, когда выбрано не менее двух элементов, так выравнивание производится относительно другого объекта.

Команды **Align Center Vertically** и **Align Center Horizontally** возможны при выборе одного или нескольких объектов, так как объекты выравниваются относительно центра страницы.

Команды **Across Space Evenly** и **Down Space Evenly** выполняются только при выборе трех и более объектов.

После выполнения команд **Align** (выравнивание) координаты этих элементов заменяются новыми.

3.6.15 Make Same Size – уравнять размеры объекта

Применяется для выравнивания размеров объектов. Выберите **Edit > Make Same Size** и соответствующие значки в меню или на панели инструментов Эта функция возможна при выборе двух и более объектов. По объекту, выбранному первым, будут заданы размеры остальным выбранным объектам.

3.6.16 Text Process - обработка текста

Используется для выравнивания, задания направления и импорта текста в ScrEdit. Выберите **Edit > Text Process** и соответствующие значки в меню или на панели инструментов.

Флаг  разрешает соответствующую команду обработки текста. В диалоговом окне **Import Text** пользователь может выбрать текст из предварительно созданного текстового банка **Text Bank**. Для создания и редактирования текстового банка выберите **Option > Text Bank**.

3.6.17 Picture

 следующее за командой **Picture command** означает, что эта функция выбрана. Для создании изображения выберите **Edit > Picture** или воспользуйтесь значком на панели инструментов. Можно воспользоваться значками на панели **Bitmap Toolbar.**

3.6.18 Duplicate - размножить

Применяется для размножения объекта. Выберите в меню **Edit > Duplicate**, откроется диалоговое окно Рис. 3.6.7, в котором пользователь может выбрать число копий объекта по вертикали и по горизонтали. Число копий должно быть не менее 2, так как копируемый объект так же входит в задаваемое здесь количество. Сняв соответствующий флажок, можно будет располагать копии только по вертикали или только по горизонтали.

Spacing (pixels): Эта опция задает расстояние (в пикселях) между копиями объекта.

Increase / Decrease Address: Эта опция используется для выбора последовательного увеличения или уменьшения адресов копий. Адрес может быть **Word** или **Bit**.

X-direction / Y-direction: Эта опция задает направление увеличения адресов копий: по горизонтали (X-direction) или по вертикали (Y-direction). Пример 1 и 2 на Рис. 3-6-8 и Рис. 3-6-9)

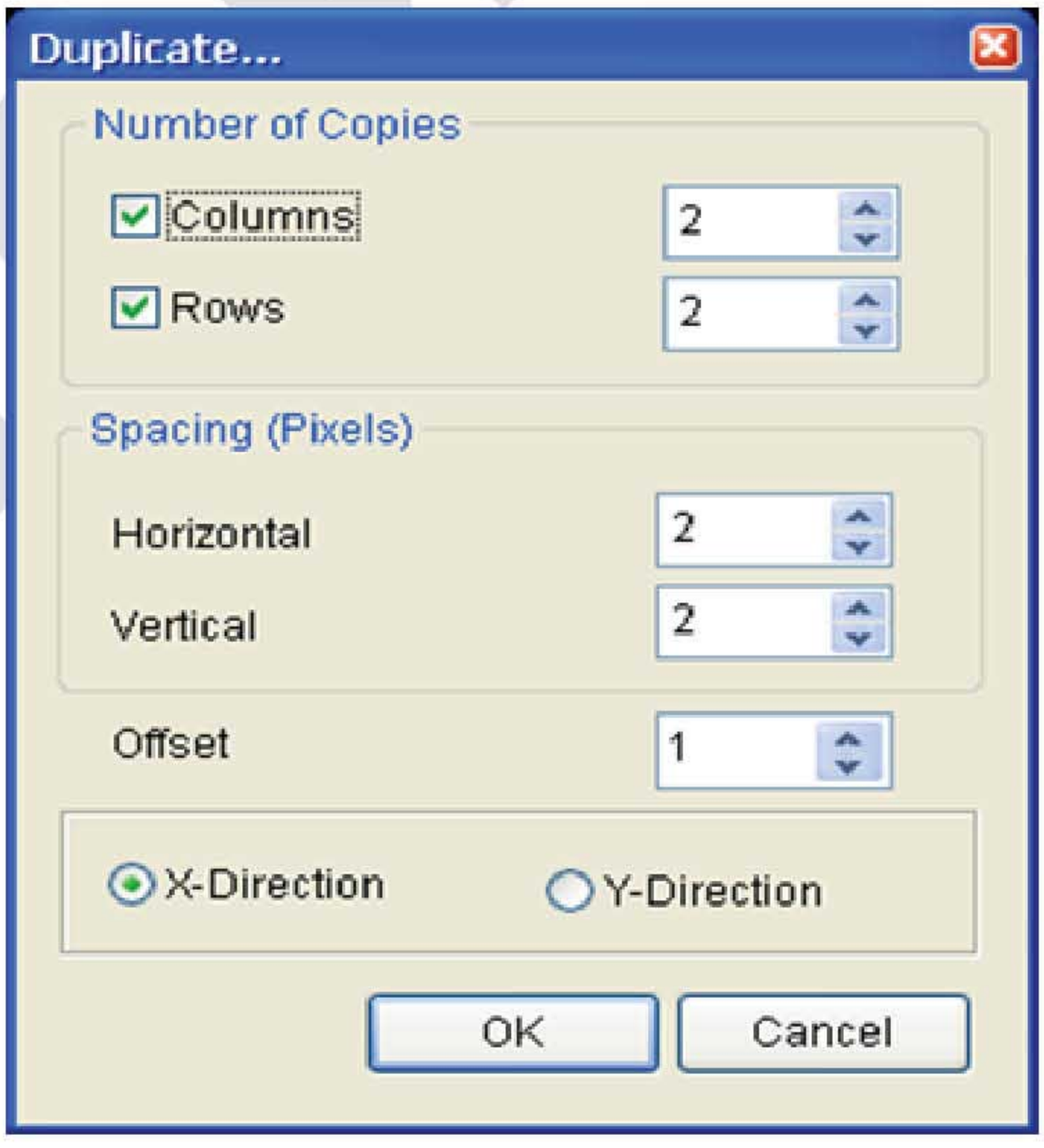


Рис. 3-6-7 Диалоговое окно Duplicate

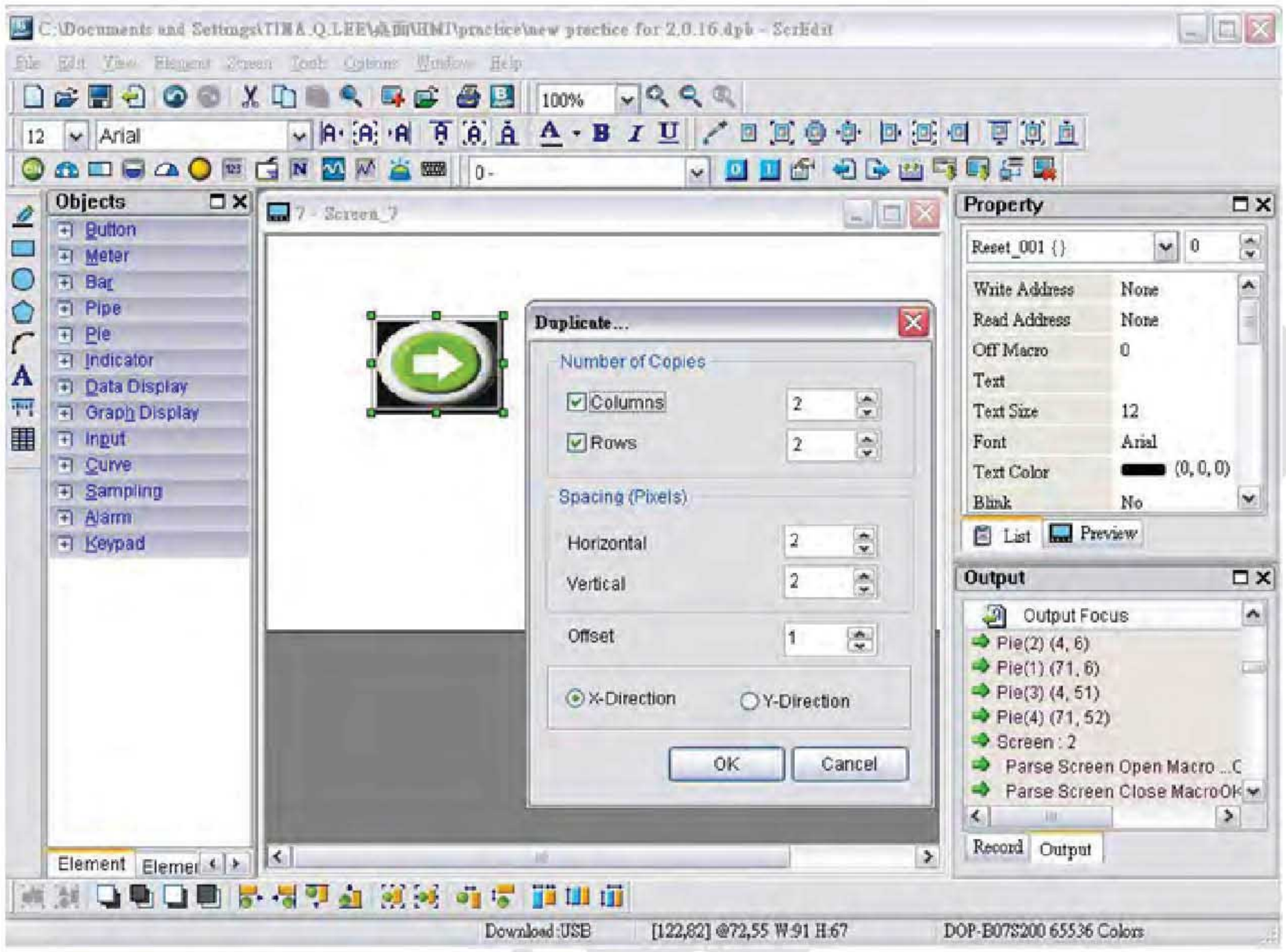


Рис. 3-6-8 **Duplicate** Пример 1

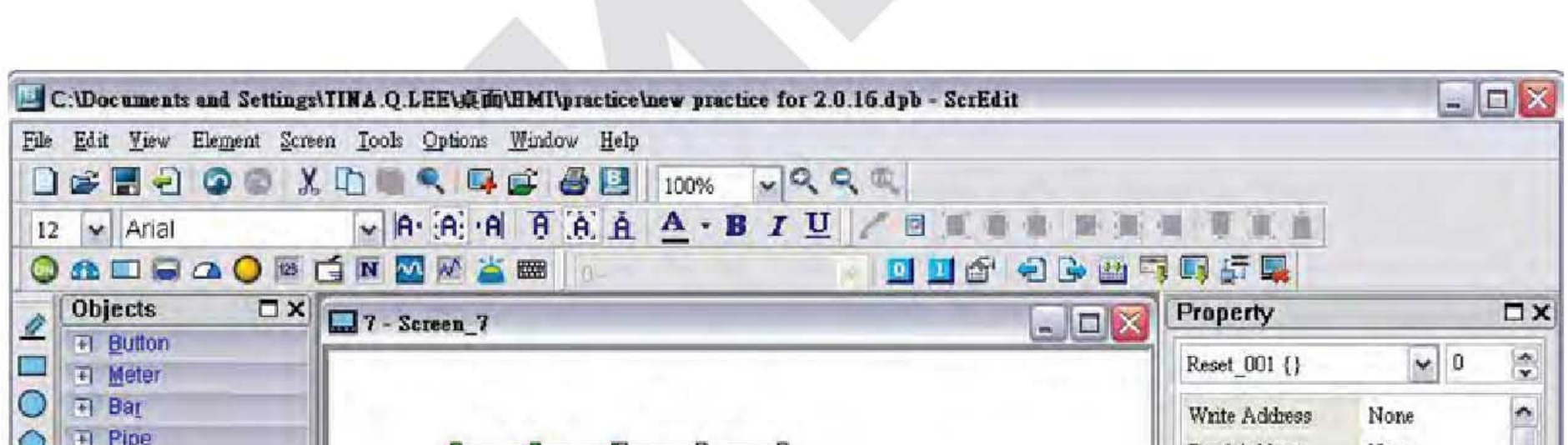
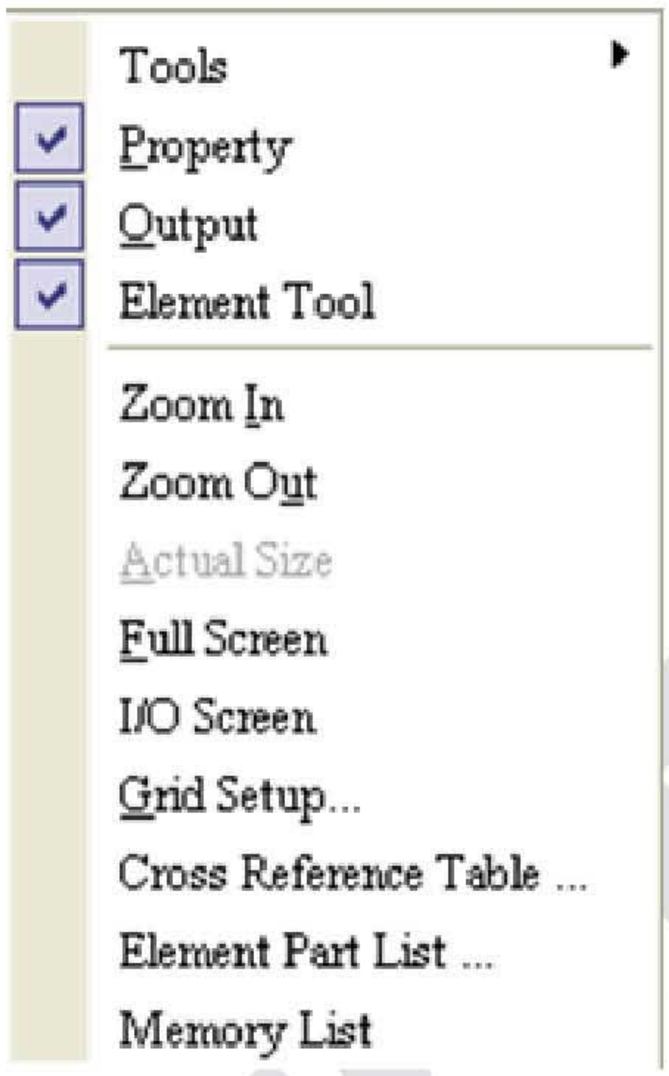


Рис. 3-6-9 **Duplicate** Пример 2

3.7 Меню Вид (View)

Элементы меню **View**



В разделе **View option**, можно выбрать какую панель инструментов и окна с информацией выбрать и настроить для удобства работы. Если  не показан над панелью, это говорит о том, что панель инструментов скрыта и не отображается на экране.

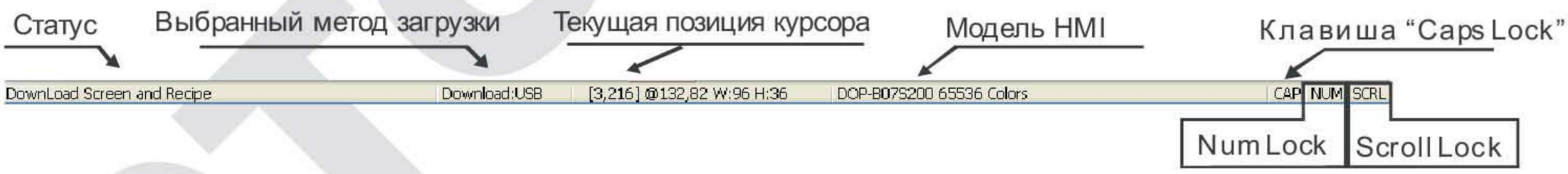
3.7.1 Tools- панель инструментов

Standard Toolbar – стандартная панель инструментов

Значок	Функция	Описание
	New	Создать новую прикладную программу
	Open	Открыть старую прикладную программу
	Save	Сохранить текущую прикладную программу
	Export	Экспортировать прикладную программу в BMP-формат

Значок	Функция	Описание
	Undo	Отменить последнее действие (на шаг назад)
	Redo	Отменить действие команды Undo (на шаг вперед)
	Cut	Вырезать выделенные элементы
	Copy	Копировать выделенные элементы
	Paste	Вставить элементы, предварительно вырезанные или скопированные
	Find	Найти в прикладной программе текст, адреса чтения и записи
	New Screen	Создать новый экран в прикладной программе
	Open Screen	Открыть экран в прикладной программе
	Print	Печать прикладной программы
	About	Показать версию ПО "Screen editor"

Status Bar - панель состояния



Text Toolbar – панель инструментов форматирования текста

Значок	Функция	Описание
	Font Size	Выбор размера шрифта
	Font Name	Выбор типа шрифта
	Align Text to Left	Выравнивание текста по левому краю

Значок	Функция	Описание
	Text Center Horizontally	Горизонтальное выравнивание текста по центру
	Align Text to Right	Выравнивание текста по правому краю
	Align Text to Top	Выравнивание текста по верхнему краю
	Text Center Vertically	Вертикальное выравнивание текста по центру
	Align Text to Bottom	Выравнивание текста по нижнему краю
	Text Color	Выбор цвета шрифта
	Bold	Жирный шрифт
	Italic	Курсив
	Underline	Подчеркивание

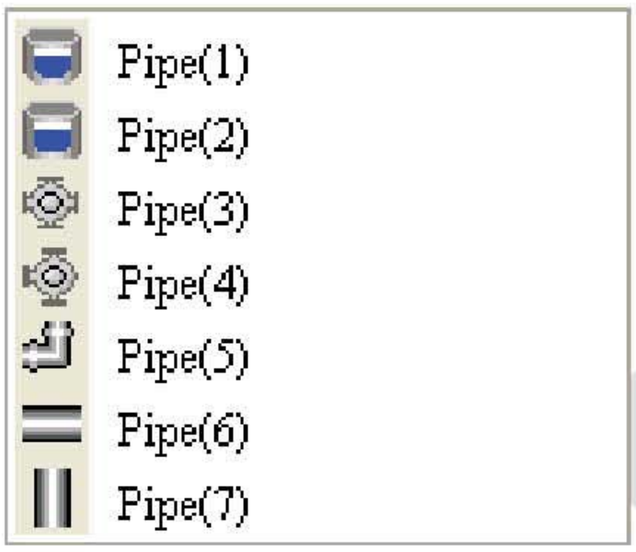
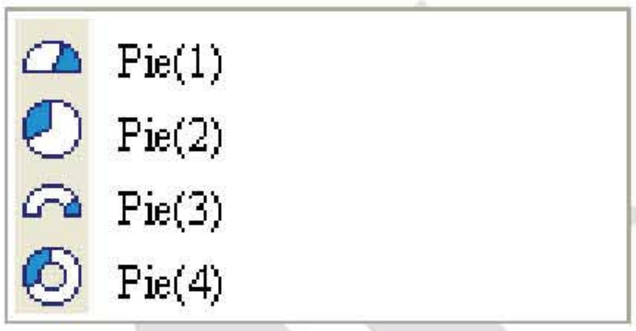
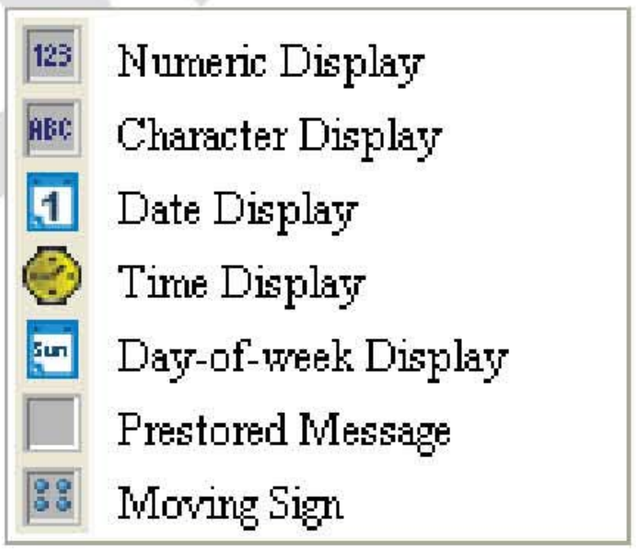
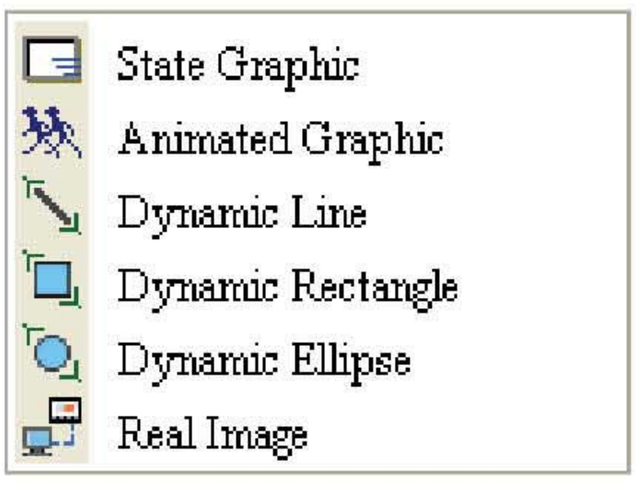
Bitmap Toolbar - панель изображений

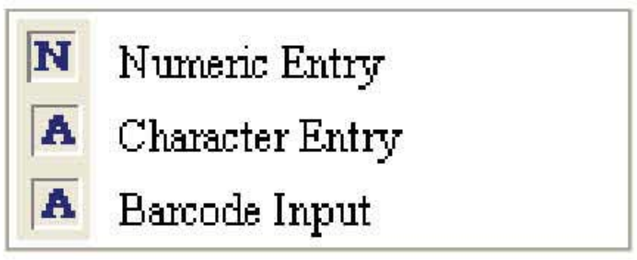
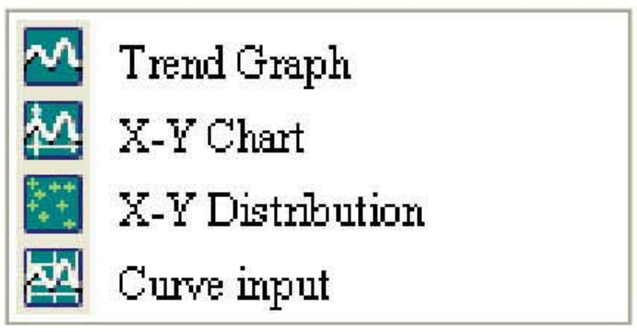
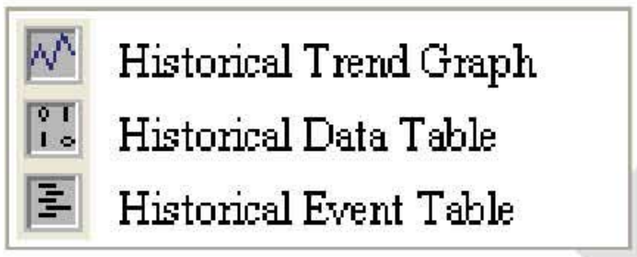
Значок	Функция	Описание
	Select Transparent Color	Используется для выбора цвета на картинке, который должен быть заменен на прозрачный.
	Chang Mode for process all state pictures	Когда эта функция активна (значок нажат), то команды растяжения, выравнивания и изменения размеров будут действовать на картинки, относящиеся ко всем состояниям выбранного объекта.
	Picture Stretch All	Растягивает (сужает) картинку до границ объекта.
	Picture Stretch 1:1	Масштабирует картинку относительно оригинального размера
	Original Picture	Изменение размеров картинки на фактические
	Picture Align Left	Выравнивание картинки по левому краю
	Picture Center Horizontally	Горизонтальное выравнивание картинки по центру

Значок	Функция	Описание
	Picture Align Right	Выравнивание картинки по правому краю
	Picture Align Top	Выравнивание картинки по верхнему краю
	Picture Center Vertically	Вертикальное выравнивание картинки по центру
	Picture Align Bottom	Выравнивание картинки по нижнему краю

Element Toolbar - панель объектов

Значок	Функция	Описание
	Button	

Значок	Функция	Описание
	Meter	 Meter(1) Meter(2) Meter(3)
	Bar	 Normal Differential
	Pipe	 Pipe(1) Pipe(2) Pipe(3) Pipe(4) Pipe(5) Pipe(6) Pipe(7)
	Pie	 Pie(1) Pie(2) Pie(3) Pie(4)
	Indicator	 Multistate Indicator Range Indicator Simple Indicator
	Display	 Numeric Display Character Display Date Display Time Display Day-of-week Display Prestored Message Moving Sign
	Graphic	 State Graphic Animated Graphic Dynamic Line Dynamic Rectangle Dynamic Ellipse Real Image

Значок	Функция	Описание
	Input	 Numeric Entry Character Entry Barcode Input
	Curve	 Trend Graph X-Y Chart X-Y Distribution Curve input
	Sampling	 Historical Trend Graph Historical Data Table Historical Event Table
	Alarm	 Alarm History Table Active Alarm List Alarm Frequency Table Alarm Moving Sign
	Keypad	 KeyPad(1) KeyPad(2) KeyPad(3)

Drawing Toolbar - панель инструментов для рисования

Значок	Функция	Описание
	Line	Нарисовать линию

Значок	Функция	Описание
	Rectangle	Нарисовать прямоугольник
	Circle	Нарисовать круг
	Polygon	Нарисовать многоугольник
	Arc	Нарисовать линию
	Text	Текстовое сообщение
	Scale	Нарисовать шкалу
	Таблица	Открыть таблицу

Layout Toolbar 1 - Панель расположения элементов 1

Значок	Функция	Описание
	Current Element State	Название текущего состояния элемента
	View State OFF/0	Переключение и анализ состояния OFF/0
	View State ON/1	Переключение и анализ состояния ON/1
	Display All Read/Write Address	Индикация всех адресов всех экранных элементов
	Previous windows	Выбор предыдущих окон.
	Next windows	Выбор последующих окон
	Compile	Компилляция программы
	Download Screen and Recipe	Загрузка программы и рецептов
	Download Screen	Загрузка программы
	On-line Simulation	Редактирование и тестирование программы в компьютере реальном подключённым конроллером

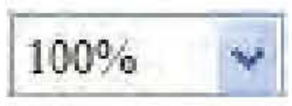
Значок	Функция	Описание
	Off-line Simulation	Редактирование и тестирование программы на компьютере без контроллера

Layout Toolbar 2 - Панель расположения элементов 2

Значок	Функция	Описание
	Group	Сгруппировать элементы
	Ungroup	Разгруппировать элементы
	Bring to Front	Расположить выбранный элемент поверх остальных элементов
	Send to Bottom	Расположить выбранный элемент под остальными элементами
	Bring Forward	Расположить выбранный элемент поверх элемента на первой позиции
	Send Backward	Расположить выбранный элемент под элементом на первой позиции
	Align Left	Выровнять выбранные элементы по левому краю
	Align Right	Выровнять выбранные элементы по правому краю
	Align Top	Выровнять выбранные элементы по верху
	Align Bottom	Выровнять выбранные элементы по низу
	Align Center Vertically	Установить элемент по центру (в вертикальном направлении)
	Align Center Horizontally	Установить элемент по центру (в горизонтальном направлении)
	Across Space Evenly	Расположить элементы равномерно по горизонтали
	Down Space Evenly	Расположить элементы равномерно по вертикали
	Make Same Width	Выровнять элементы по ширине
	Make Same Height	Выровнять элементы по высоте

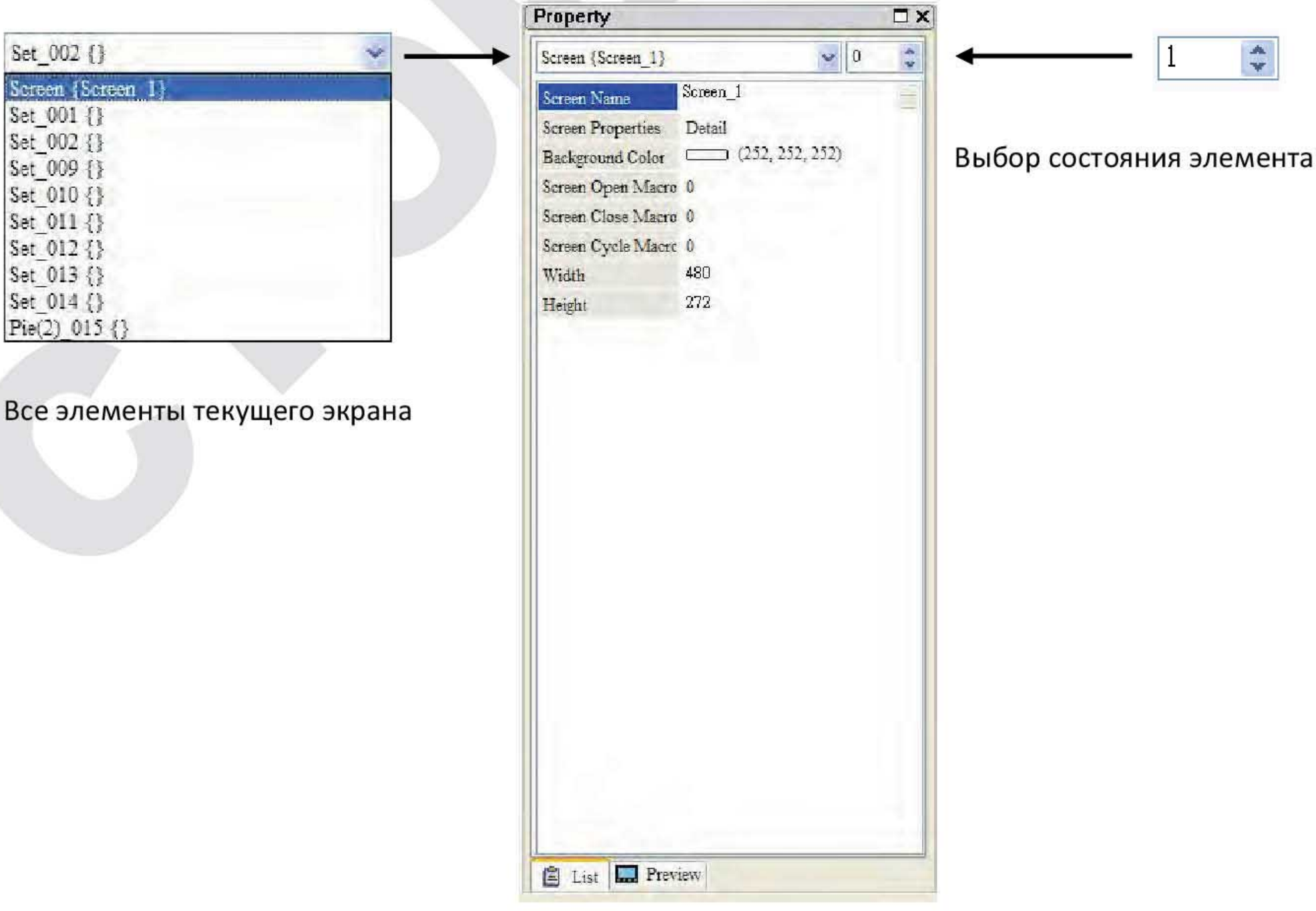
Значок	Функция	Описание
	Make Same Size	Выровнять элементы и по ширине и по высоте

Zoom Toolbar - панель масштабирования

Значок	Функция	Описание
	Display Level	Задание масштаба, включая: 25%, 50%, 75%, 100%, 150%, 200% и 300%
	Zoom In	Увеличение масштаба, включая 150%, 200% и 300%.
	Zoom Out	Уменьшение масштаба, включая 25%, 50% и 75%
	1:1	Установка исходного масштаба (100%)

3.7.2 Property Таблица - таблица настроек элементов

Для каждого элемента приводятся настройки (Рис. 3-7-2).



Все элементы текущего экрана

Выбор состояния элемента

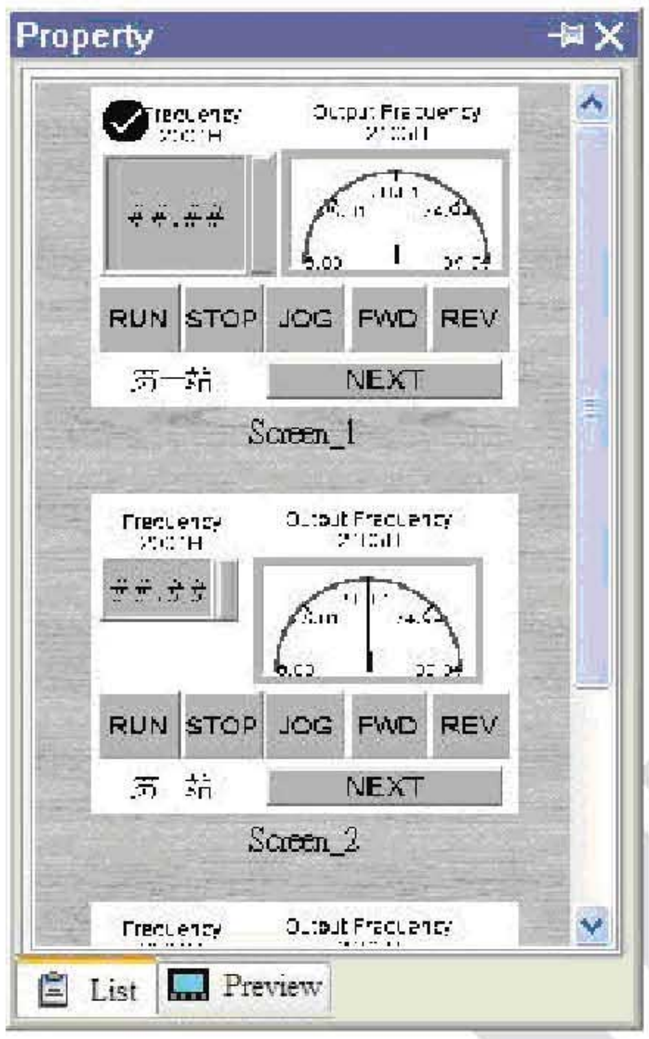


Рис. 3-7-2 Таблица настроек и просмотр редактируемого экрана

3.7.3 Record and Output Window - окно выходного результата

Окно выходного результата отображает все действия при редактировании, выходные сообщения при компиляции. При компиляции HMI программы, система автоматически определяет программу. В случае ошибки в окне выходного результата появляется сообщение об ошибке. Для получения подробной информации об ошибке, кликните курсором мыши на сообщение об ошибке (Рис. 3-7-3, Рис. 3-7-4, Рис. 3-7-5, Рис. 3-7-6).

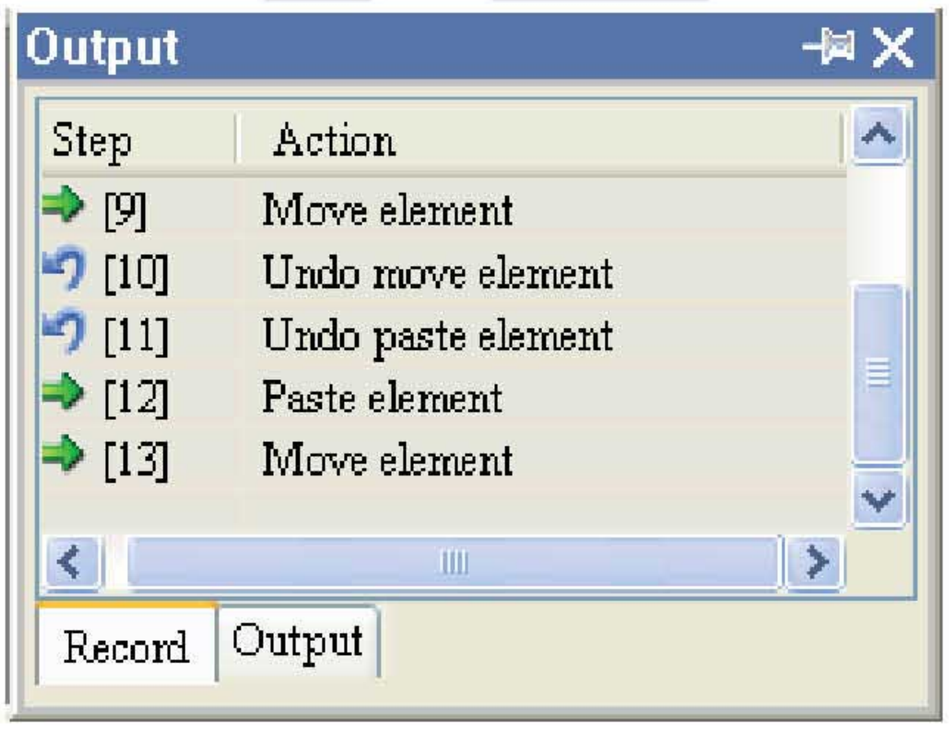


Рис. 3-7-3 Окно записи

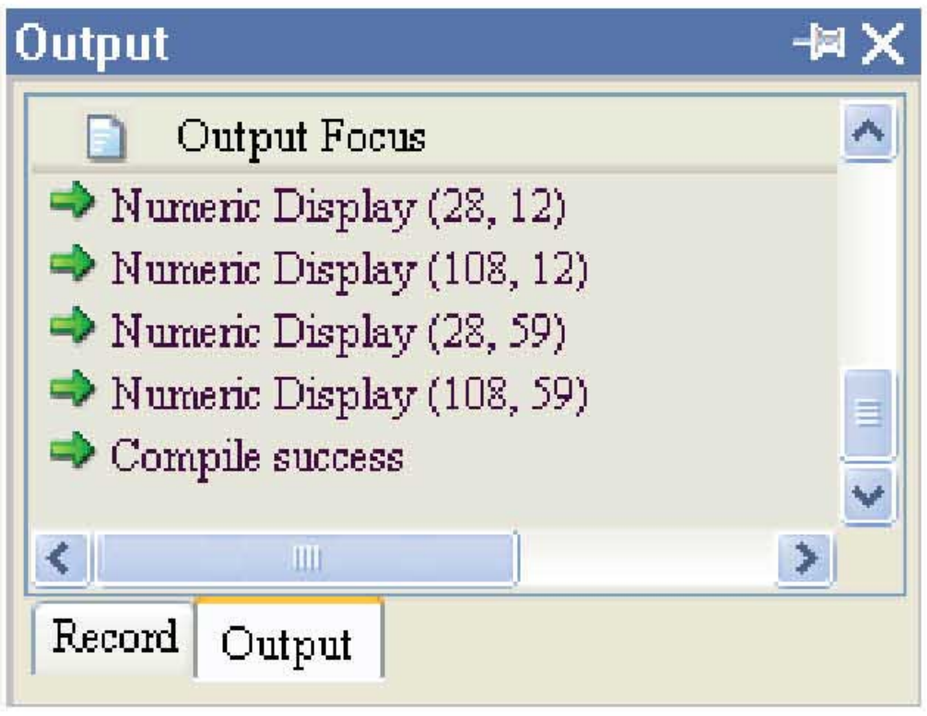


Рис. 3-7-4 Вывод результата

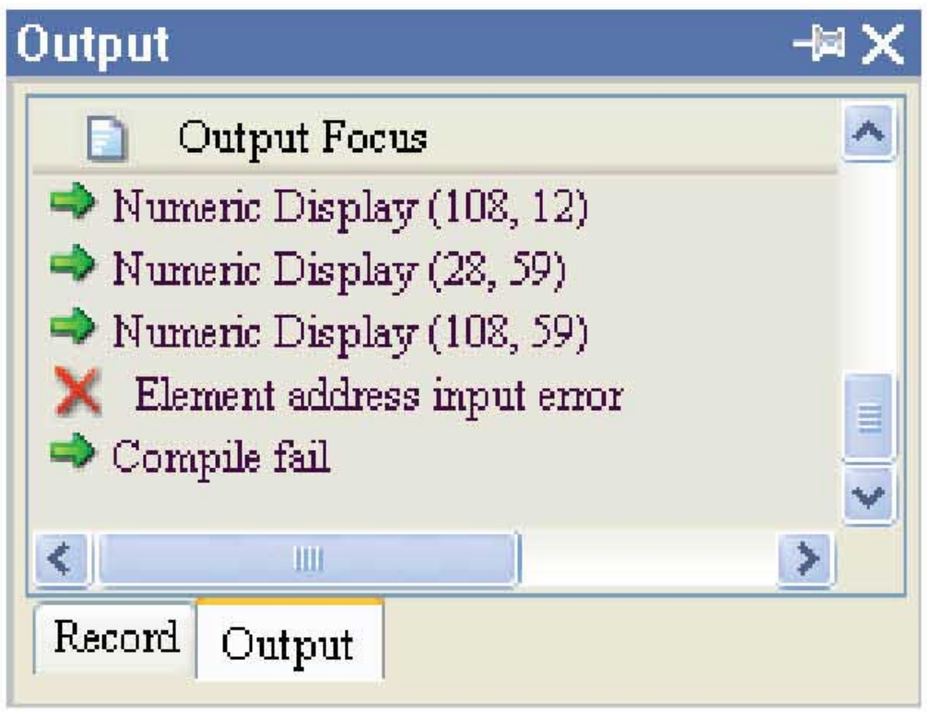


Рис. 3-7-5 Результаты компиляции

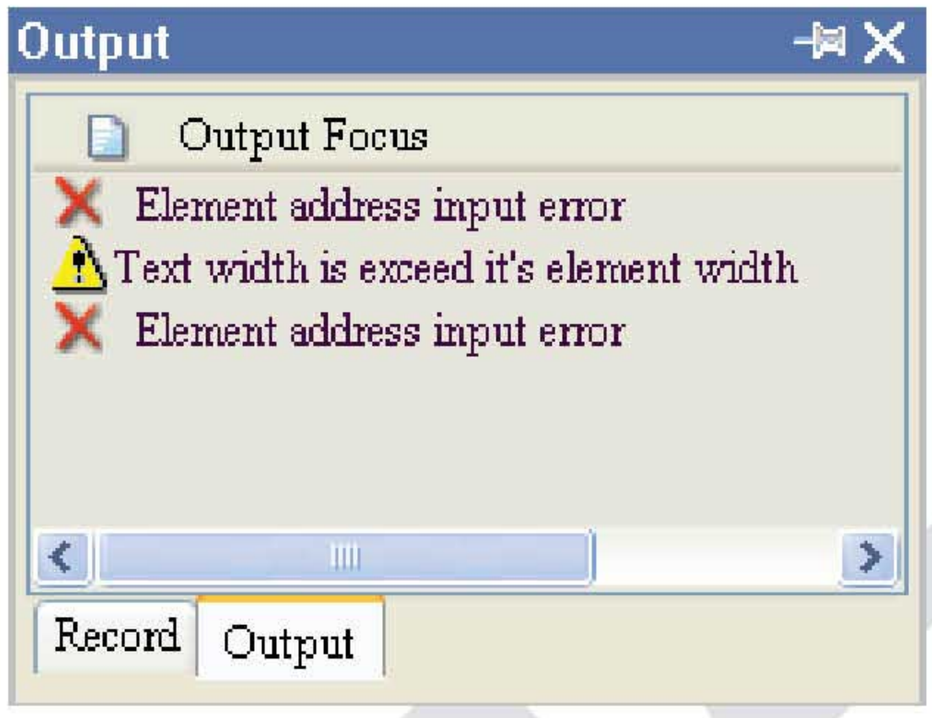


Рис. 3-7-6 Ошибки и предупреждения

3.7.4 Zoom In – приближение

Увеличение рабочего экрана в ScrEdit и всех объектов на один уровень (Рис. 3-7-7, Рис. 3-7-8).

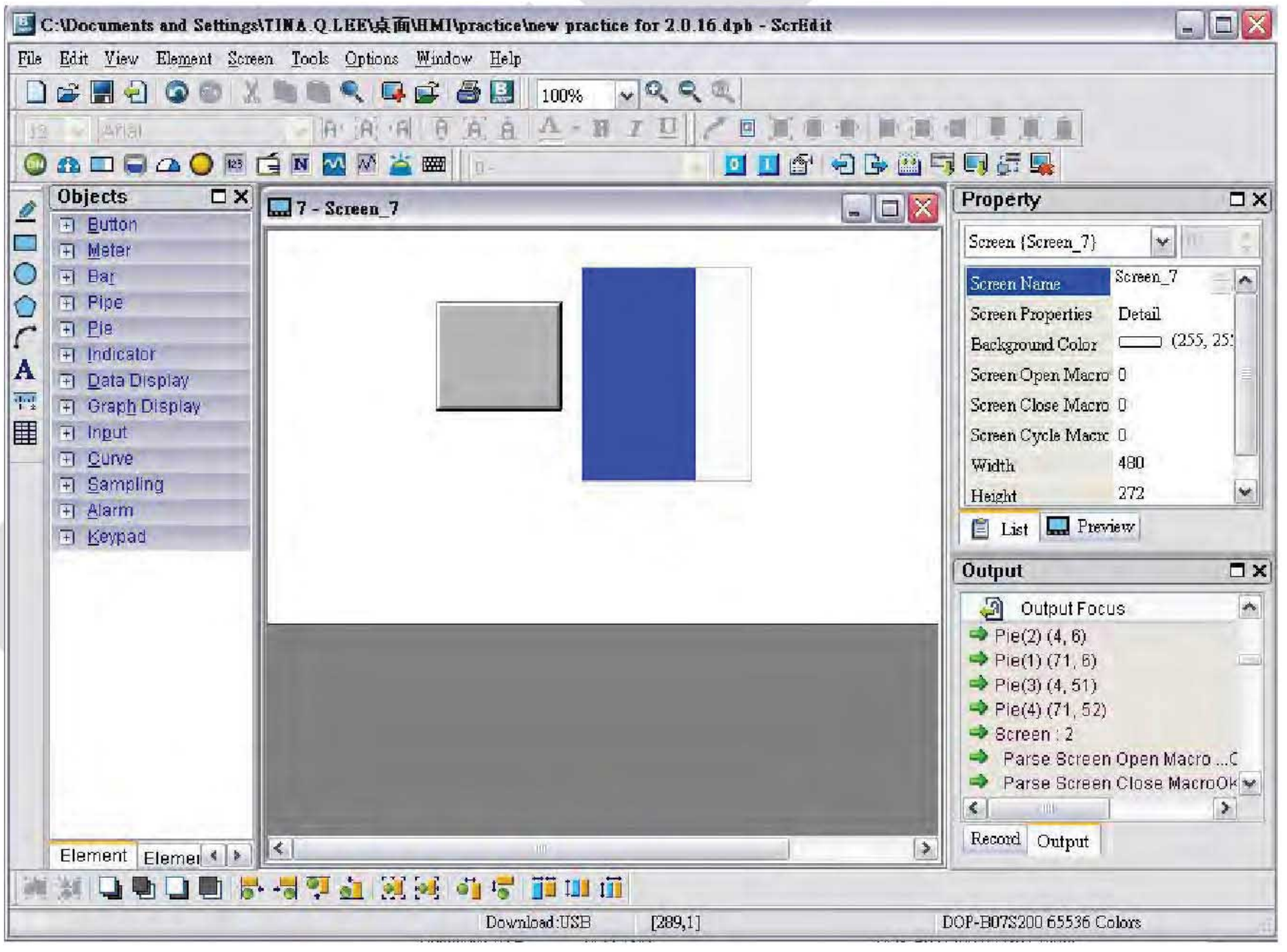


Рис. 3-7-7 Размер = 100% (До выбора команды Zoom In)

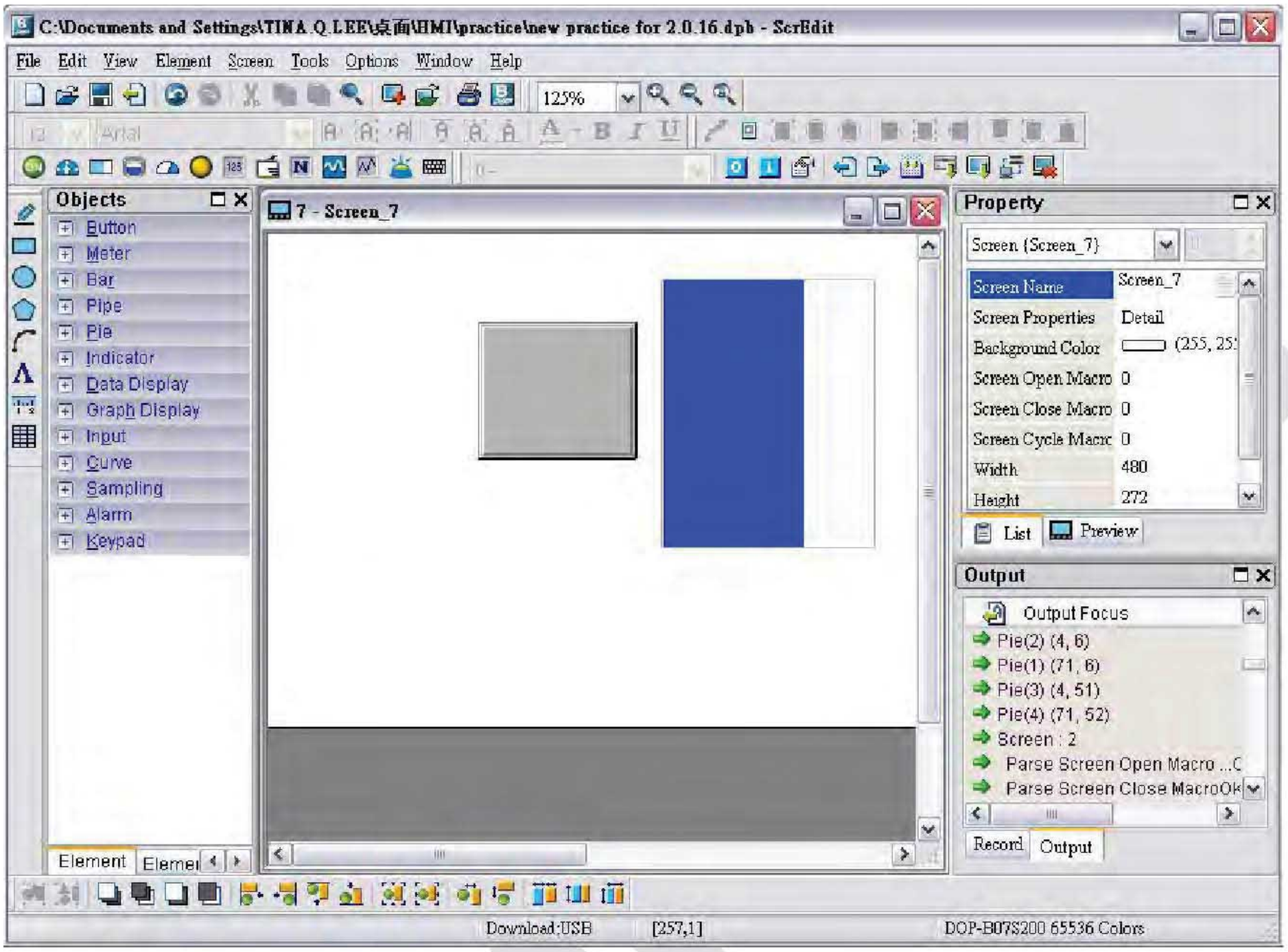


Рис. 3-7-8 Размер = 100% (После выбора команды Zoom In)

3.7.5 Zoom Out - отдаление

Уменьшение рабочего экрана в ScrEdit и всех объектов на один уровень (Рис. 3-7-9).

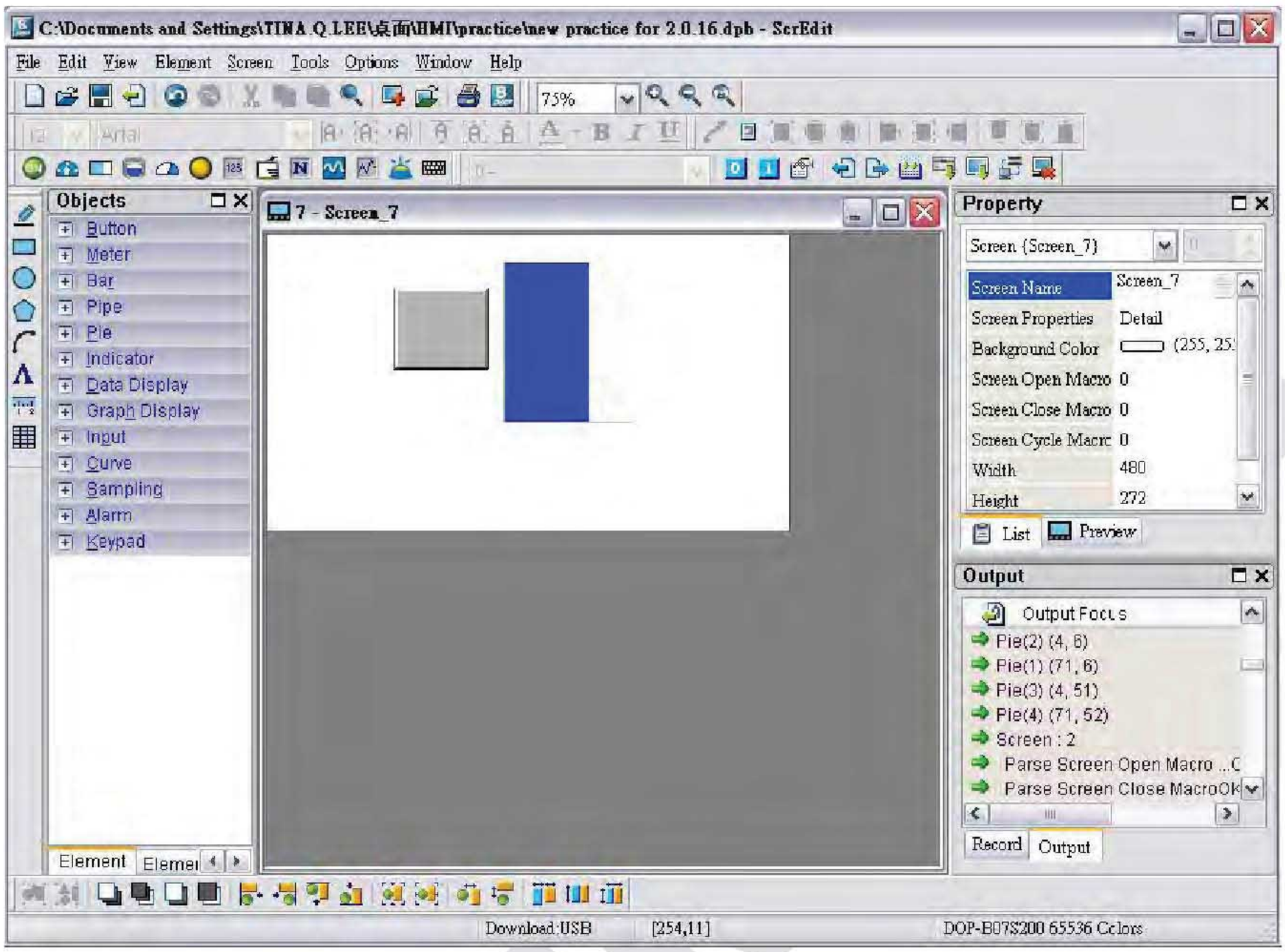


Рис. 3-7-9 Уровень масштабирования = 75%
(После выполнения команды Zoom Out)

3.7.6 Actual Size – актуальный размер экрана

Установка исходного масштаба рабочего экрана (100%).

Пользователь может выбирать масштаб рабочего экрана напрямую из списка: 25%, 50%, 75%, 100%, 150%, 200% и 300% (Рис. 2.5.16) или используя значки или

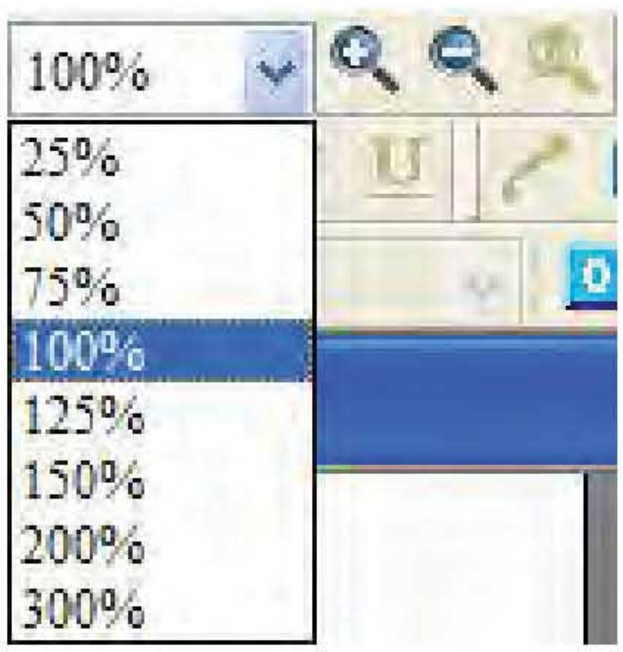


Рис. 3-7-10 Исходного размер экрана

3.7.7 Full Screen – полноэкранный режим

См. Рис. 3-7-11. Полноэкранный режим предоставляет максимальную рабочую область ScrEdit. В полноэкранном режиме будут скрыты все панели инструментов и дополнительные окна ScrEdit, но будет показано количество строк макропрограммы.

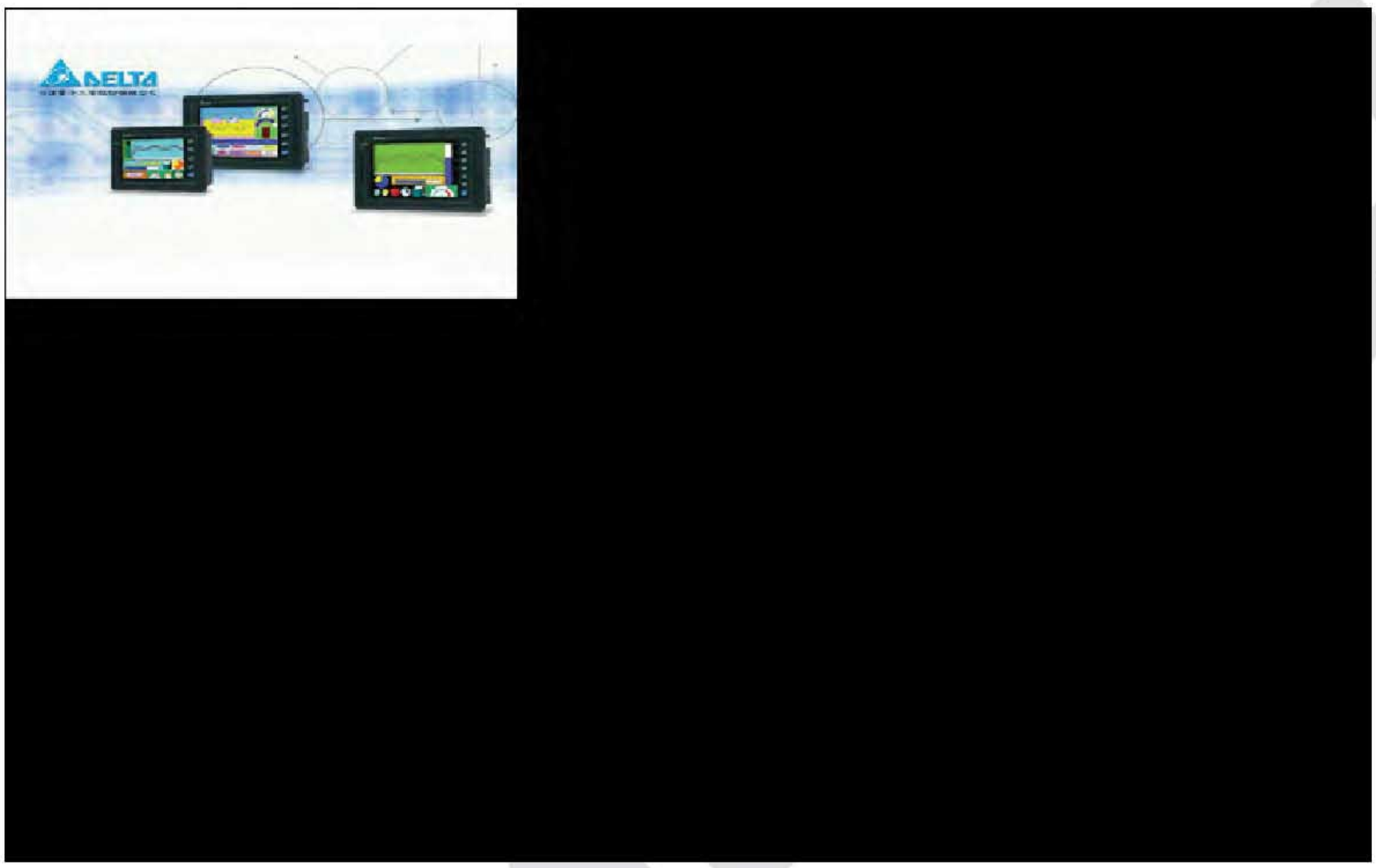


Рис. 3-7-11 Полноэкранный режим
(Нажмите клавишу “Esc” или левую кнопку мыши
для выхода из полноэкранного режима)

3.7.8 I/O Screen I/O Screen – полноэкранный режим с индикацией адресов чтения/записи

Функция вызывает полноэкранный режим, но помимо рабочей области и количества макрокоманд, будут отображаться адреса чтения/записи всех объектов на экране.

Рис. 3-7-12 Режим I/O Screen
(Нажмите клавишу “Esc” или левую кнопку мыши
для выхода из полноэкранного режима)

3.7.9 Grid Setup – установка сетки

Координатная сетка помогает пользователю выравнивать и размешать объекты проще и точнее. Расстояние между точками сетки (spacing) может быть задано в диалоговом окне Рис. 3-7-13 и Рис. 3-7-14).

Show Grid: Показывать сетку на экране.

Snap to Grid: Привязывает объекты к координатной сетке.

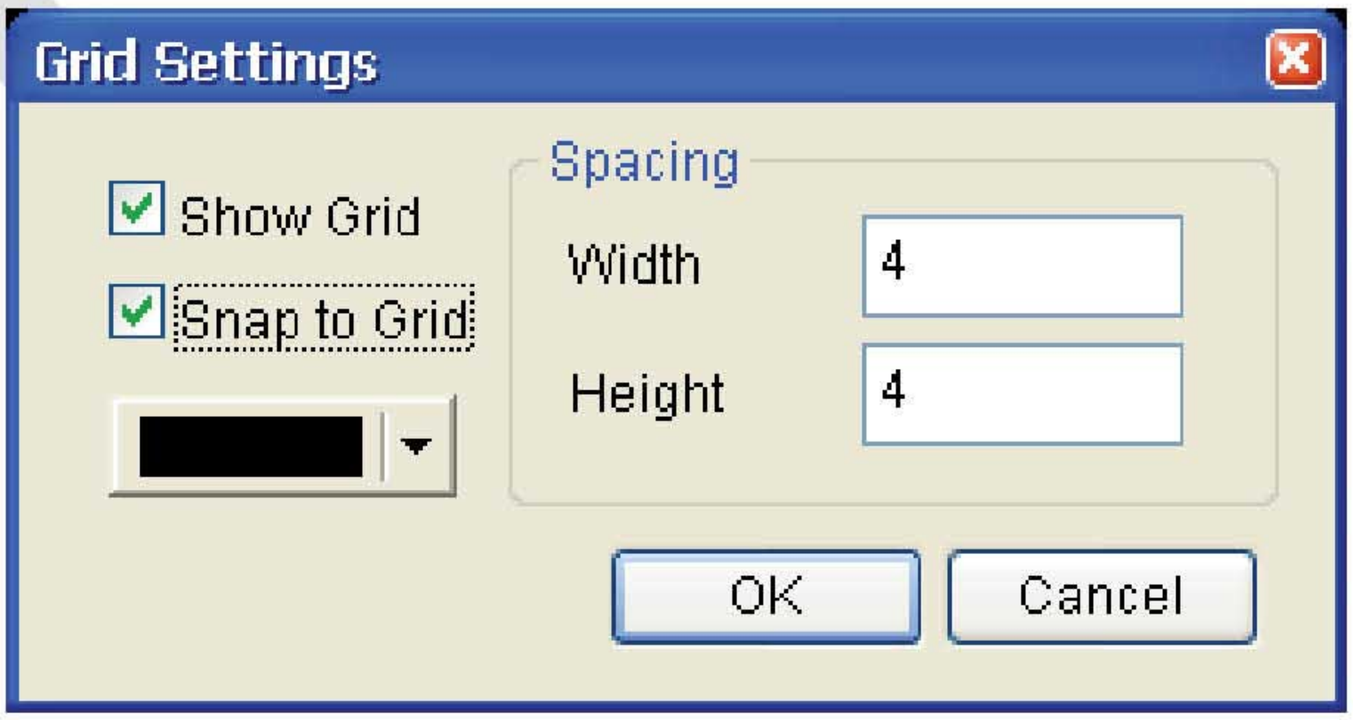


Рис. 3-7-13 Диалоговое окно “Gird Settings”

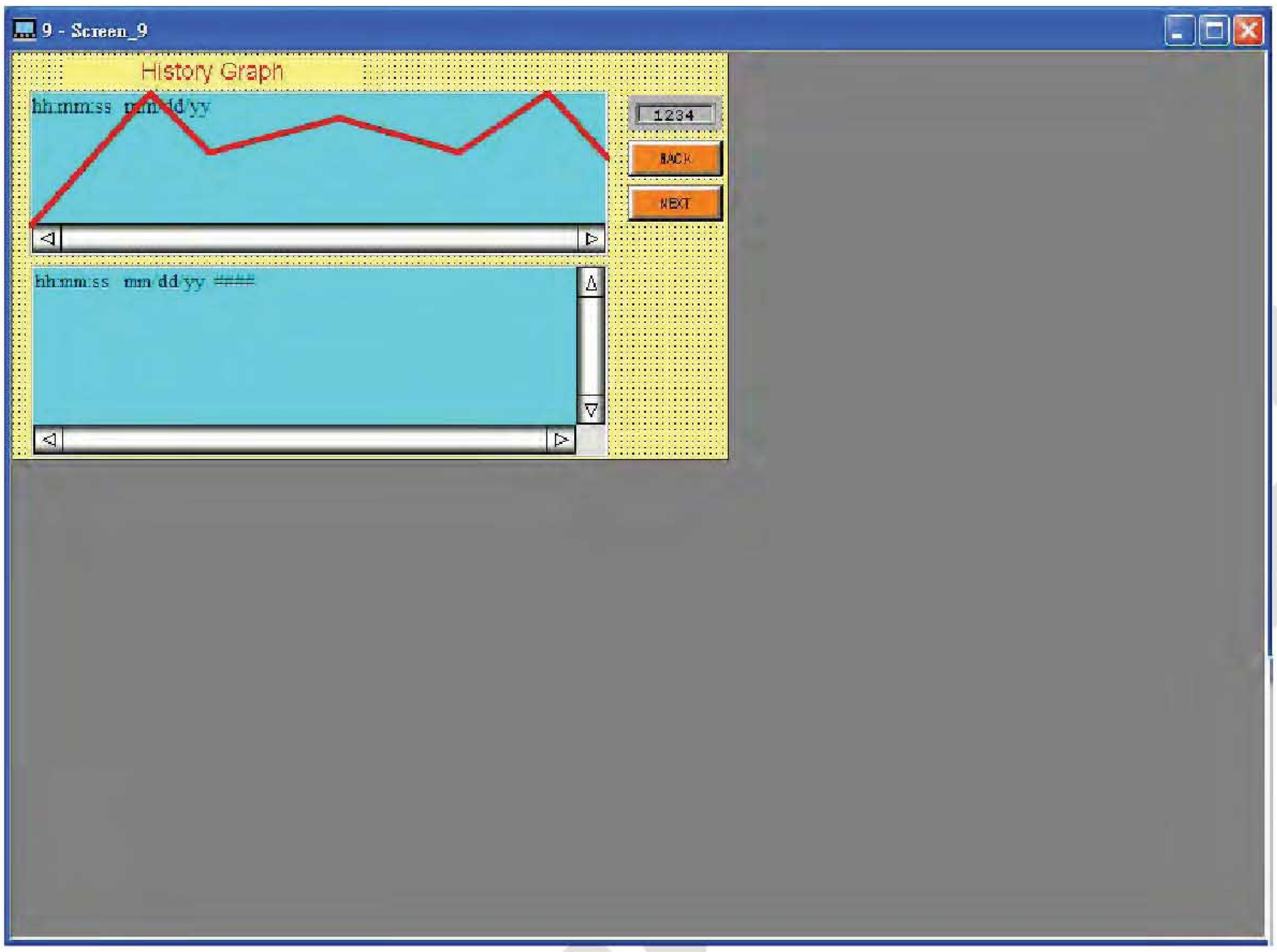


Рис. 3-7-14 Координатная сетка на рабочем экране

3.7.10 Cross Reference Table - таблица взаимосвязанных объектов

Когда создается и редактируется большое количество различных видов объектов, может происходить повторение используемых адресов. Чтобы избежать этой ситуации, в ScrEdit есть функция “**Cross Reference Table**”, с помощью которой пользователь может быстро и удобно отыскать повторяющиеся адреса чтения и записи всех объектов. Пользователь сможет увидеть адреса чтения и записи выбранного объекта во взаимосвязи с адресами других объектов, макрокоманд или системной областью управляющих регистров. Функция **Cross Reference Table** вызывается в меню **View**.

Первая строка таблицы перекрестных ссылок показывает ссылку к выбранному объекту, а другие строки – ссылки к объектам, имеющим такие же адреса. Двойным кликом мыши по ссылке пользователь может автоматически перейти к объекту, на который указывает ссылка.

На Рис. 3-7-15, мы можем увидеть, что адреса регистра аварий и и макрокоманды используют один и тот же адрес $50.

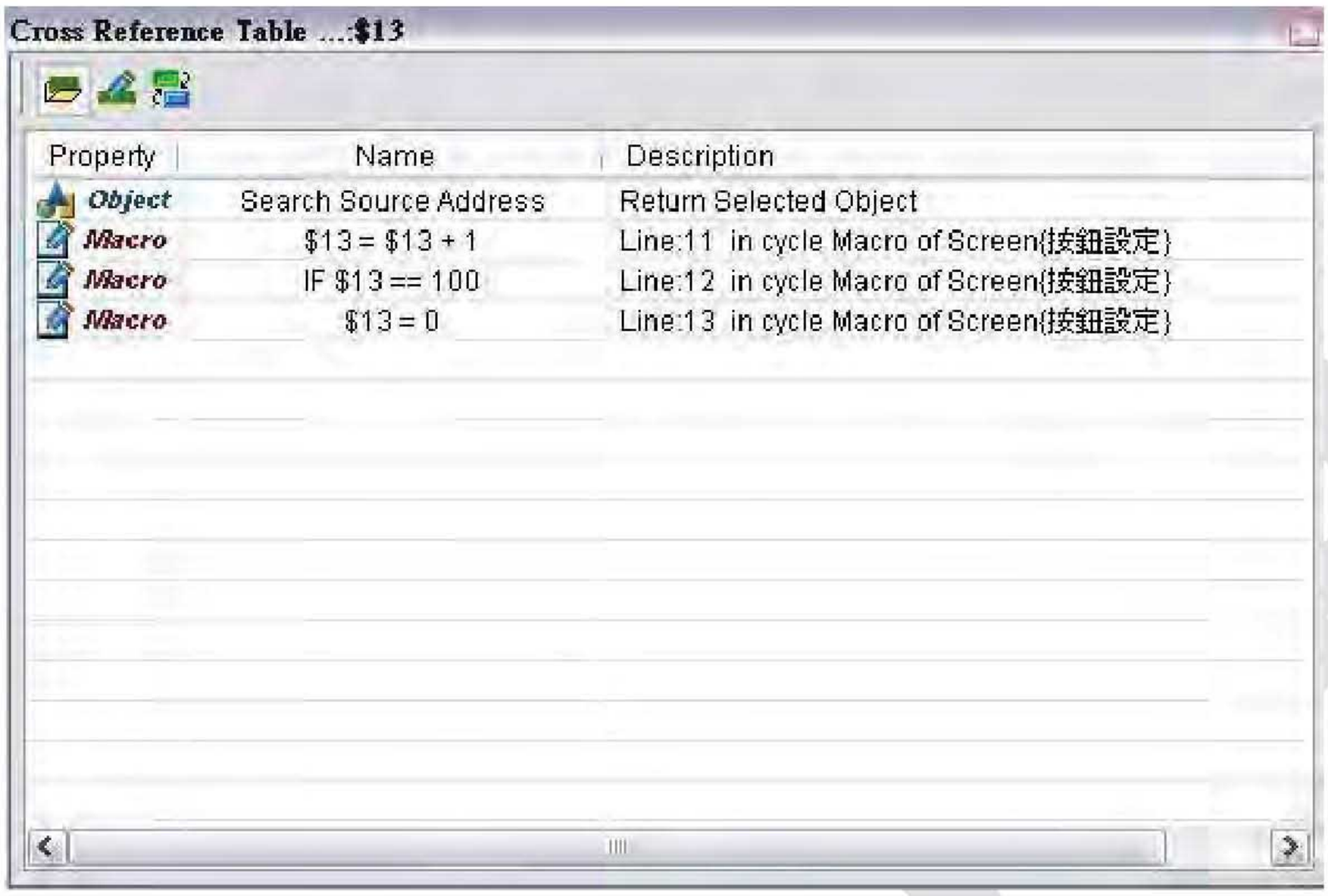


Рис. 3-7-15 Cross Reference Table

3.7.11 Element Part List

Когда выбрана функция **Element Part List**, ScrEdit будет сортировать и классифицировать все элементы на текущем экране. Объекты сортируются по типу и классифицируются по следующим свойствам (Name, Describe, Write / Read address, Trigger address, Trigger type, Interlock и Level). В каждой таблице Пользователь может редактировать свойства объектов как здесь, так и в таблице свойств объектов.

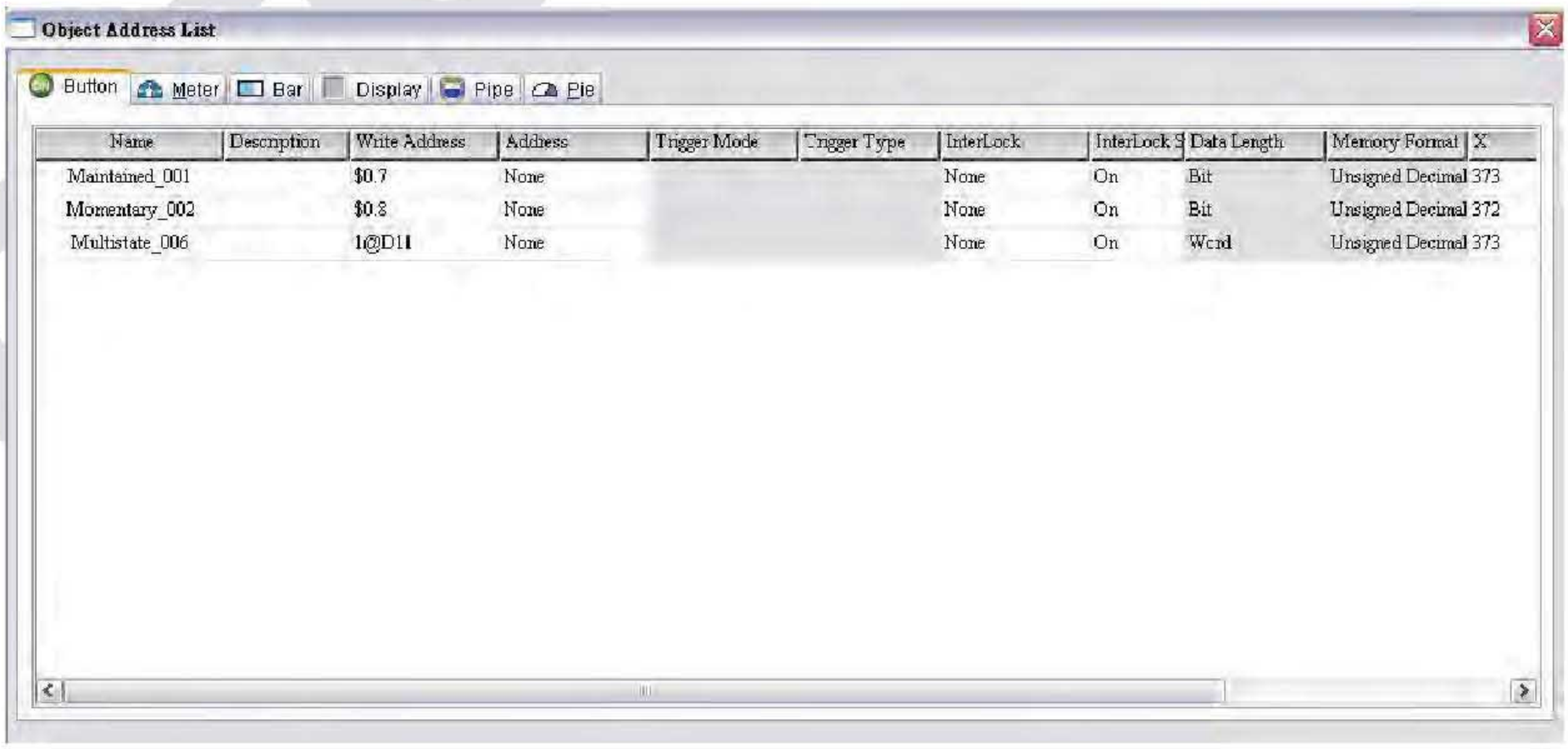


Рис. 3-7-16 Диалоговое окно Element Part List

3.7.12 Memory List – лист распределения памяти

В панелях оператора серии DOP-B имеется четыре вида памяти:

1. **ROM**: область памяти для хранения рецептов и экранных данных.
2. **SRAM**: энергонезависимая память, где можно хранить архивы данных и аварий, данные проектов. Пользователь может контролировать её объём.
3. **SDARM**: память для хранения выполняемой программы. Для получения этой информации пользователь должен открыть проект и скомпилировать его.
4. **External Storage**: показывает, что для хранения энергонезависимых данных используется USB диск. Данные из SRAM будут перегружаться в эту область памяти.

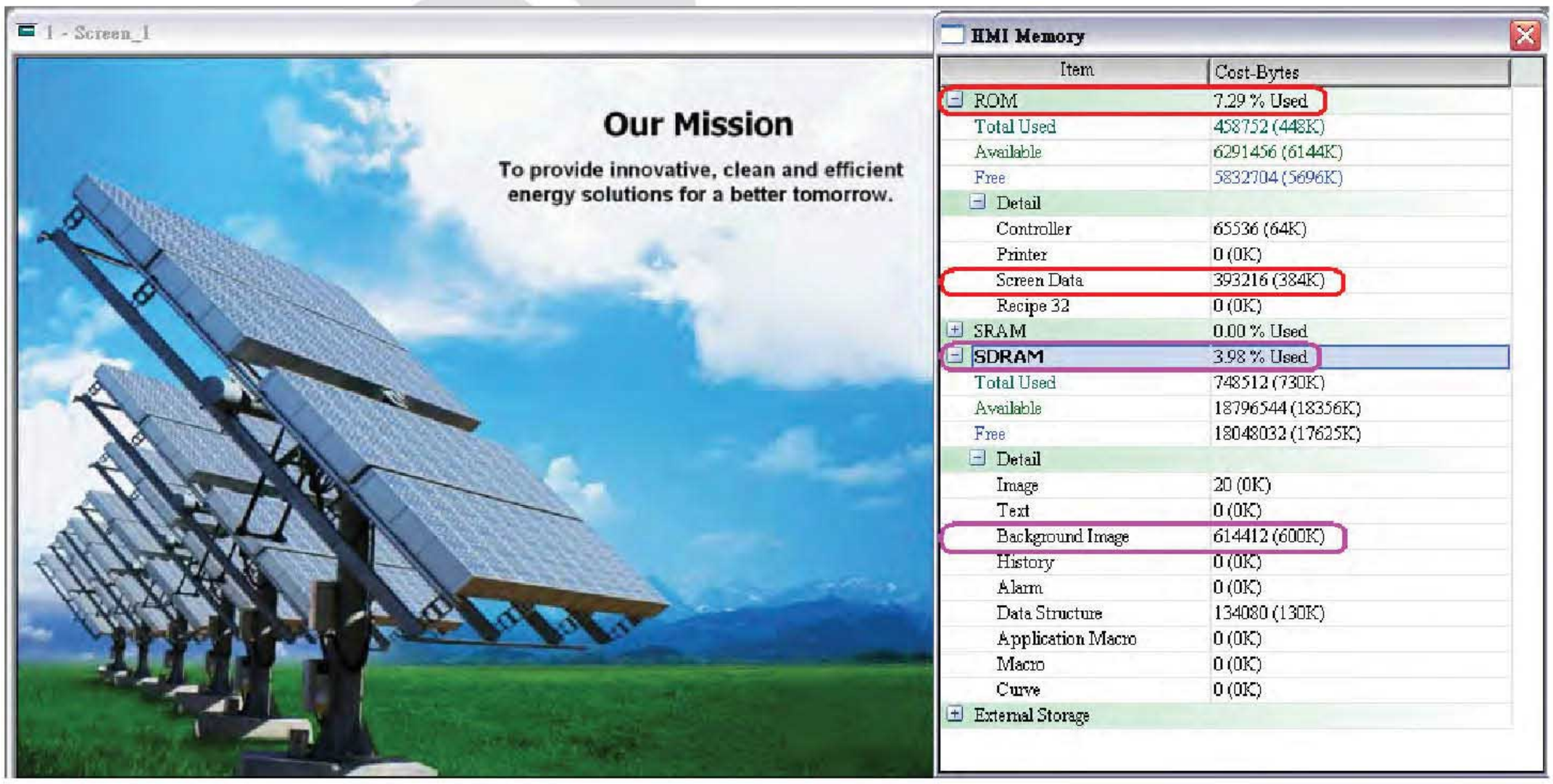
Примеры размещения

(1). Распределение памяти после импортирования картинки.

(2). Распределение памяти после создания истории редактирования (history data).

(1) Импорт изображения

При открытии проекта распределение памяти выглядит так: ROM=3.13%, SDAM=0.71%, а после импортирования картинки: ROM=7.29%, SDRAM=3.98%



Таким образом, объём памяти ROM увеличился до 256K (384K-128K), а объём памяти SDRAM до 600K (600K-0K).

(2) Распределение памяти после открытия архива.

В этом случае, первоначальное распределение памяти выглядит следующим образом:

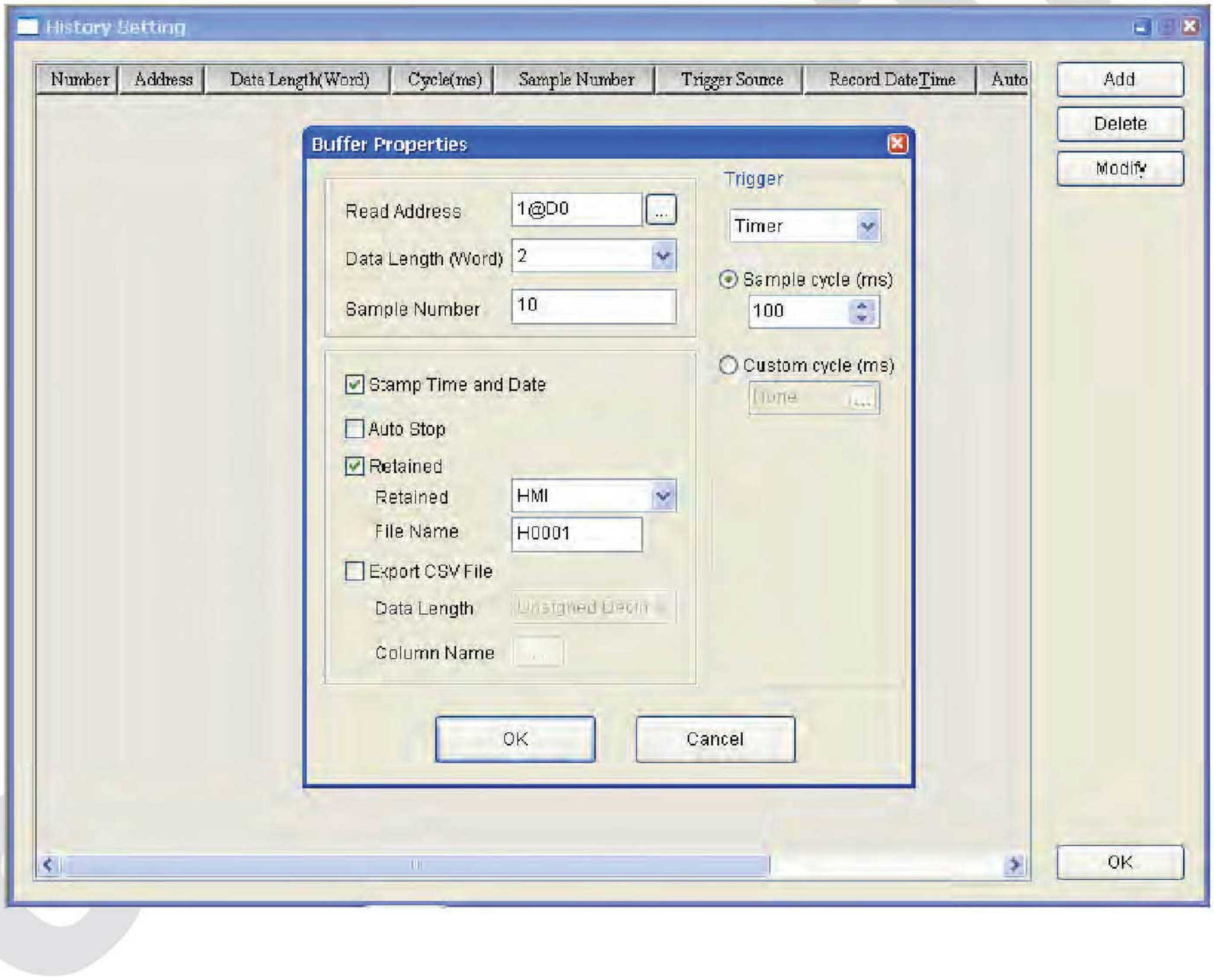
ROM=7.29%,

SDAM=3.98%,

SRAM=0%,

внешняя память=0%

Откройте, как показано ниже, архив из двух слов. Убедитесь, что архив сохранился в памяти.



После настройки архива, учитывая, что каждая запись даты и времени занимает 6 байт, каждая запись данных занимает 2 слова (4 байта).

SRAM = (6+4) x 10 +10 байт (для системы) = 110 байт

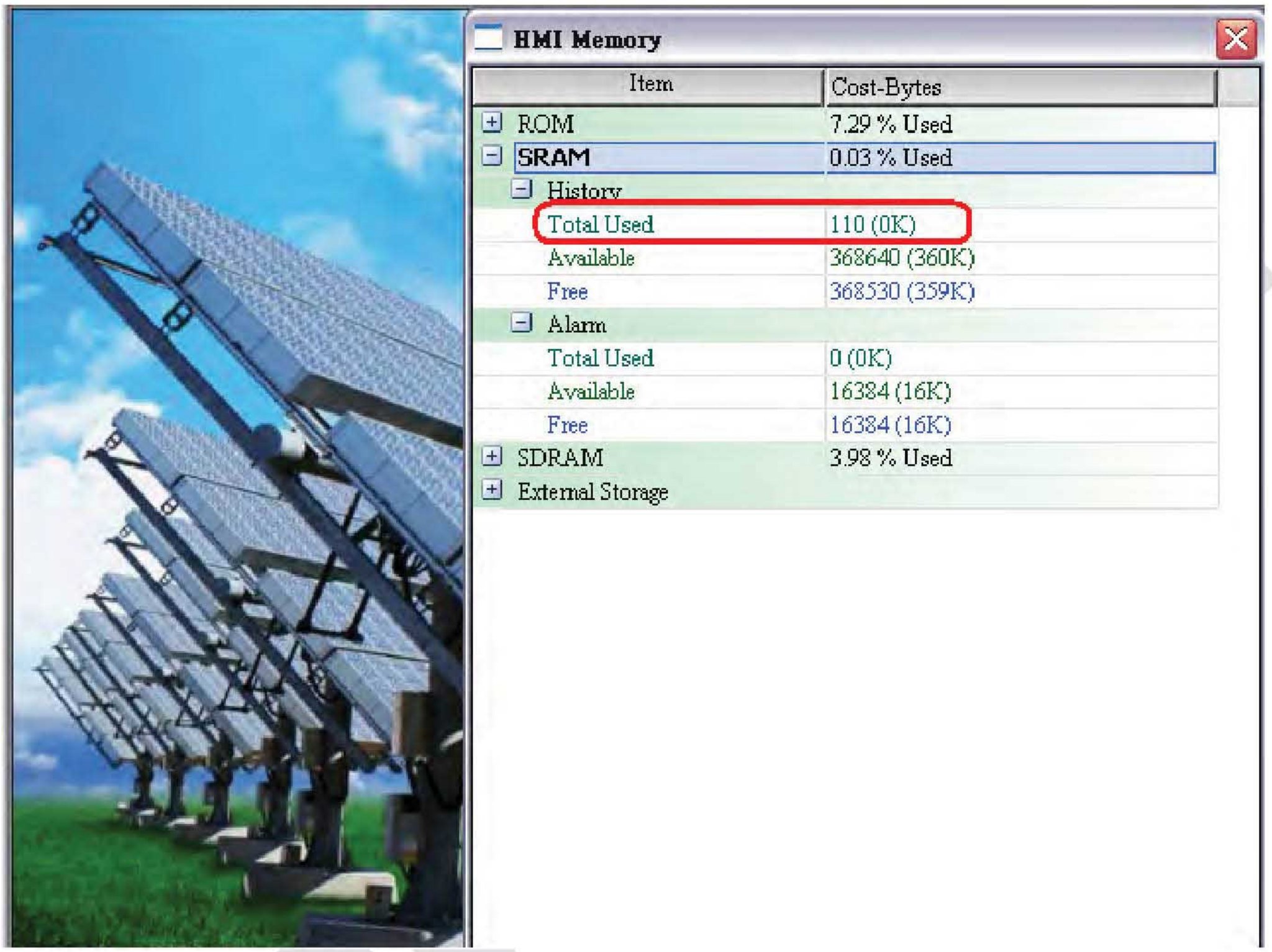
Распределение памяти при этом стало:

ROM=7.29%,

SDAM=3.98%,

SRAM=0.03%,

внешняя память =0%



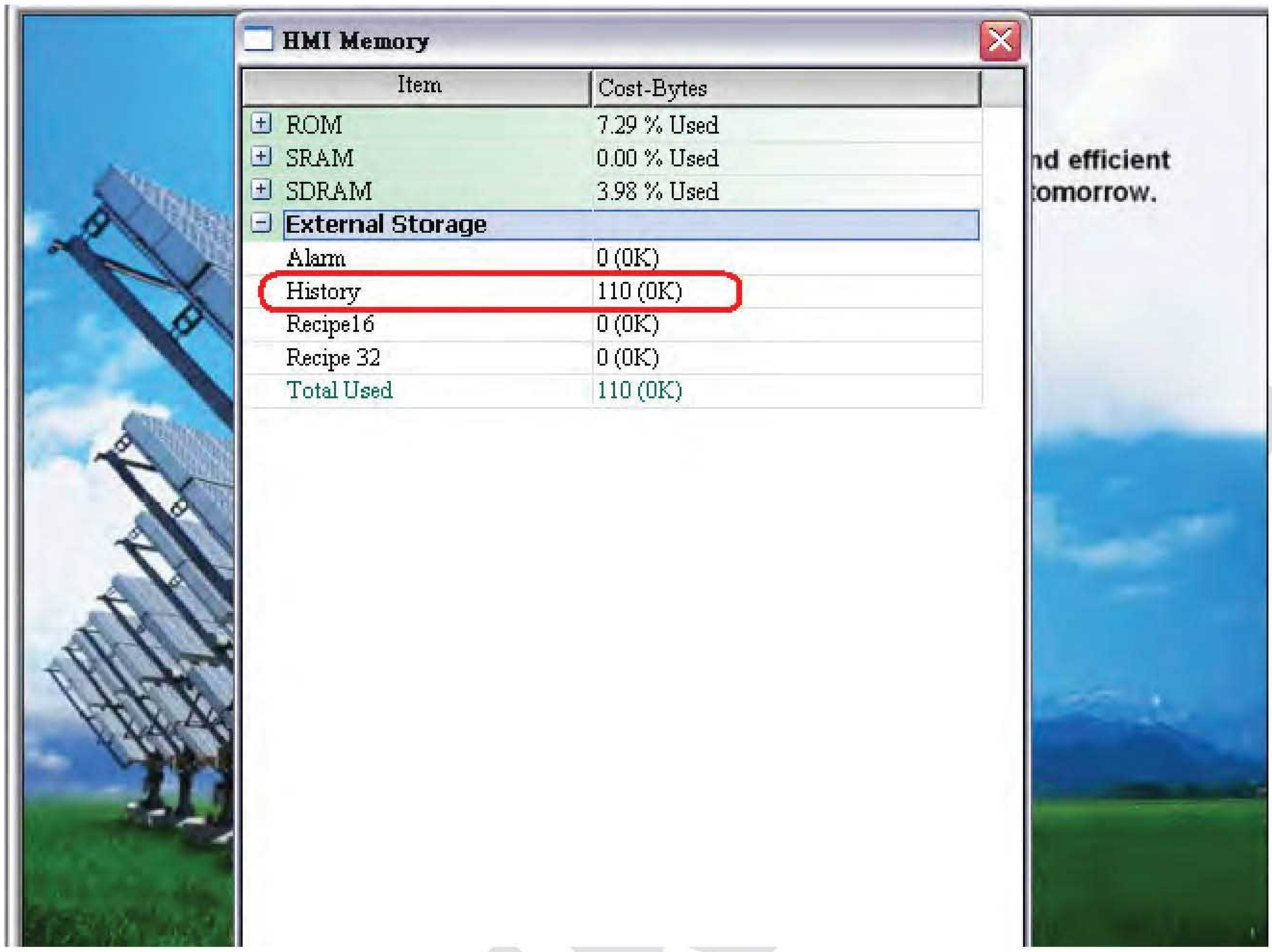
При сохранении архива на USB диске (Внешняя память) занятая память изменилась следующим образом:

ROM=7.29%,

SDAM=3.98%,

SRAM=0%,

Внешняя память=110байт



Подробнее о настройках архива аварий и событий можно прочитать в разделе 3.11.3 (Глава 3).

3.14 Макрофункции

Панели оператора серии Delta DOP-B имеют большой выбор макрокоманд: арифметические, логические, обмен данными, преобразование данных, их сравнение, управление программными циклами, управление битовыми операндами, настройка последовательных портов (Рис. 3.14.1).

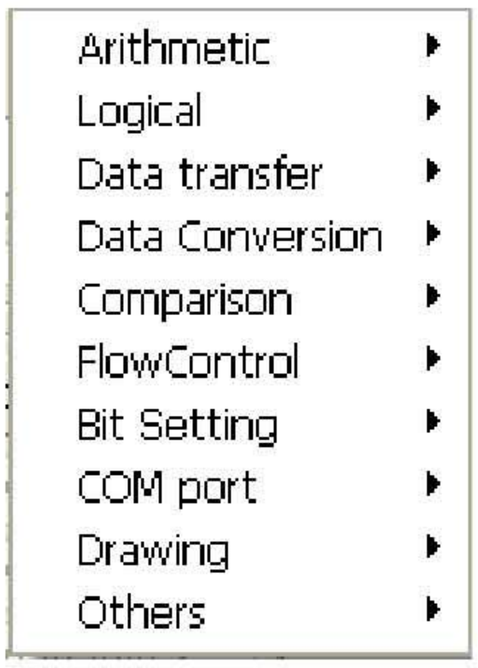


Рис. 3-14-1 Макрокоманды

Применение макрокоманд даёт пользователю возможность:

1. Оптимизировать экранные данные.
2. Уменьшать объём программирования контроллера
3. Преодолевать программные и аппаратные ограничения

Макрофункции являются мощным инструментом пользователя. До загрузки программы в память панели имеется возможность в режиме on-line или off-line симуляции проводить проверку правильности функционирования программы.

3.14.1 Типы макросов

Существует 11 типов макрокоманд которые можно разделить на 4 категории

1. **On / Off Macro**: ими обеспечены все битовые элементы, которые могут быть элементами ввода, такие как кнопки Maintained и Momentary.
2. **Before / After Execute Macro**: ими обеспечены все элементы ввода (знако/ буквенного и кнопочные элементы (включая системные гнопки)
3. **Screen Open / Screen Close / Cycle Macro**: используют экран как функциональный блок. Каждый экран может иметь индивидуальную макропрограмму.
4. **Initial / Background / Clock / Sub Macro**: используют систему как функциональный блок. Каждая программа может иметь индивидуальную макропрограмму.

Таблица 3-14-1 Таблица макрокоманд

Двоичный вид (x)	Функция
On Macro Макрос включения	Пользователь может использовать **On Macro** для всех битовых объектов Button. Выполняется при изменении состояния бита с OFF на ON.
Off Macro Макрос выключения	Пользователь может использовать **Off Macro** для всех битовых объектов Button. Выполняется при изменении состояния бита с ON на OFF.
Before Execute Macro Макрос предваритетельного исполнения	Пользователь может использовать **Before Execute Macro** для всех битовых объектов Button и объектов ввода цифр и букв. Каждый элемент имеет один макрос **Before Execute Macro**.
After Execute Macro Макрос последующего исполнения	Пользователь может использовать **After Execute Macro** для всех битовых объектов **Button** и объектов ввода цифр и букв. Каждый элемент имеет один макрос **After Execute Macro**.

Двоичный вид (x)	Функция
Screen Open Macro Макрос открытия экрана	**Screen Open Macro** может использоваться только один раз при открытии экрана (или переключении на новый экран) и экранные элементы не появятся до окончания выполнения **Screen Open Macro**. Поэтому при создании программы важно, чтобы при выполнении макроса не происходило зацикливание программы макроса, не рекомендуется делать макрос очень длинным.
Screen Close Macro Макрос закрытия экрана	Screen **Close Macro** может использоваться только один раз при закрытии экрана и никакой другой макрос не будет выполняться до окончания выполнения **Screen Close Macro**. Поэтому при создании программы важно, чтобы при выполнении макроса не происходило зацикливание программы макроса.
Screen Cycle Macro Циклический экранный макрос	Макрос выполняется непрерывно в течении всего времени, пока открыт экран.
Initial Macro Макрос инициализации	Макрос выполняется только один раз в программе **Initial Macro** до открытия экрана. Полезен при задании параметров инициализации контроллера и панели.
Background Macro Фоновый макрос	При выполнении **Background Macro** производится выполнение одной или нескольких команд последовательно. Выполнение других макросов, например **Cycle Macro** не оказывает на его работу никакого влияния. Этот макрос не требует работы в цикле, так как он выполняется непрерывно.
Clock Macro Циклический макрос	Циклический макрос **Clock Macro** будет однократно выполняться, а повторное выполнение будет происходить соответствии с уставкой периода **Clock cycle** в экранном меню **Options > Standard**. Такие типы макросов как **Cycle Macro** не рекомендуется делать длинным.
Sub-macro Суб-макрос	Имеется 512 подпрограмм, выполняющие повторяющиеся фрагменты программы, благодаря которым экономится время редактирования программы. Для вызова подпрограммы используется команда **CAL**L, например, **CALL 1** вызывает первую подпрограмму.

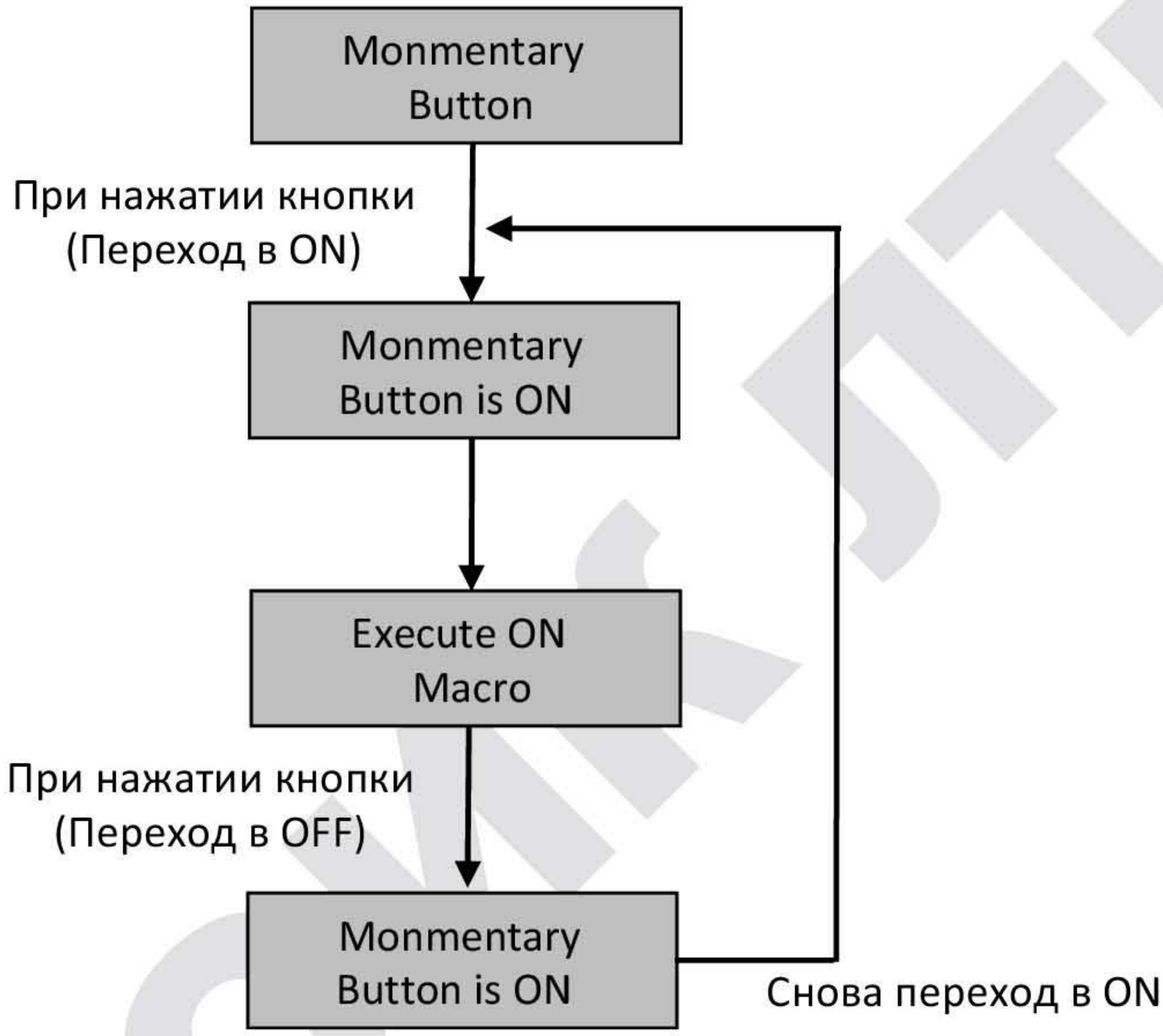
3.14.1.1 On Macro

Этот макрос связан с элементом кнопка (button). Пользователь может использовать для каждого кнопочного элемента, адресуемого битом. Он называется **On Macro** так как выполняется только один раз при изменении состояния бита с OFF на ON.

Пока кнопка не будет повторно переключена в состояние ON, повторное выполнение макроса не произойдёт.

Однако, выполнение макроса будет происходить только тогда, когда кнопка нажата и бит изменил своё состояние с OFF на ON.

Макрос не будет выполняться, если бит будет переведён в состояние ON, без нажатия кнопки.



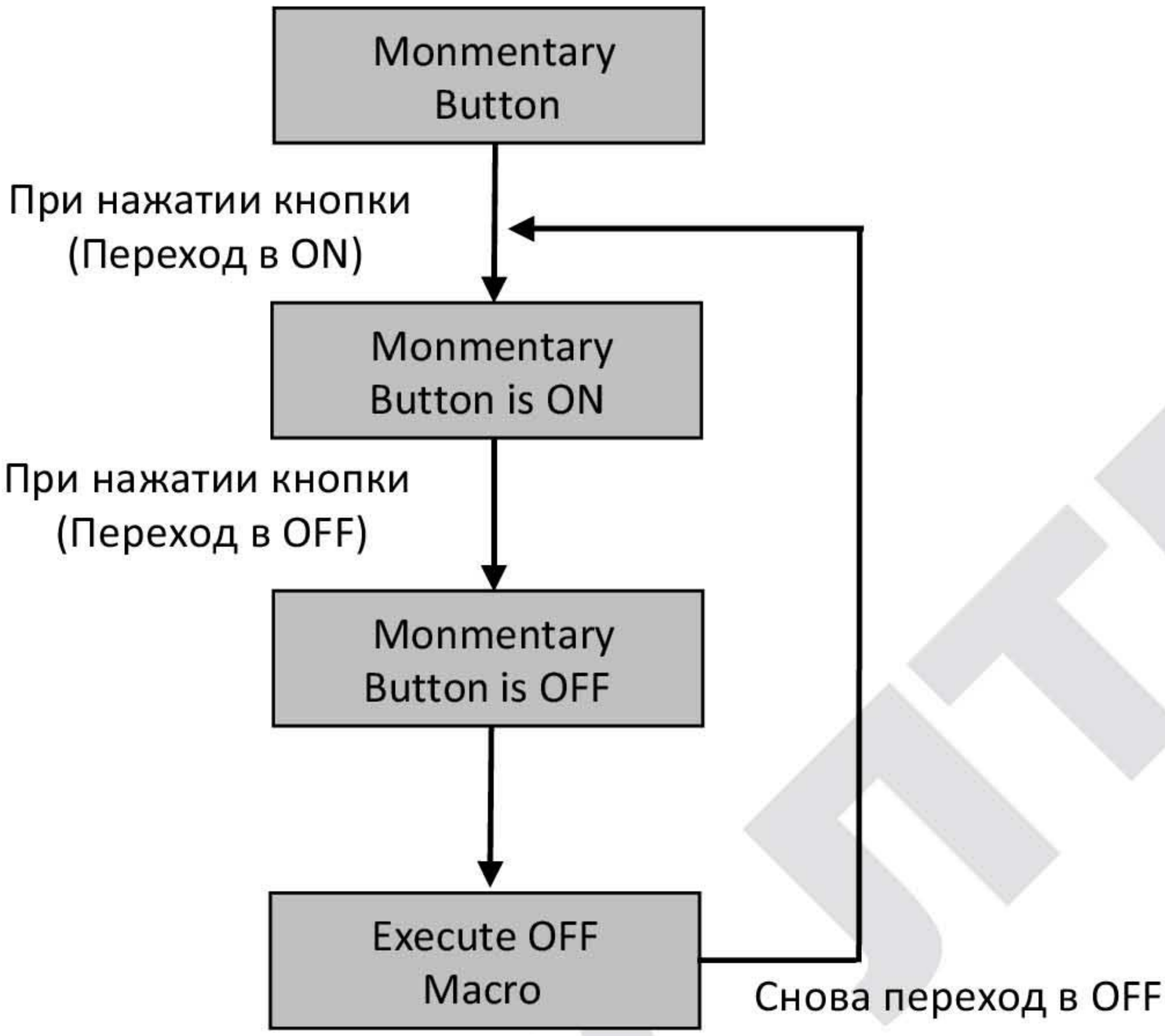
3.14.1.2 Off Macro

Этот макрос связан с элементом кнопка. Пользователь может использовать для каждого кнопочного элемента, адресуемого битом. Он называется **Off Macro** так как выполняется только один раз при изменении состояния бита с ON на OFF.

Пока кнопка не будет повторно переключена в состояние OFF повторное выполнение макроса не произойдёт.

Однако, выполнение макроса будет происходить только тогда, когда кнопка нажата и бит изменил своё состояние с ON на OFF.

Макрос не будет выполняться если бит будет переведён в состояние OFF без нажатия кнопки.



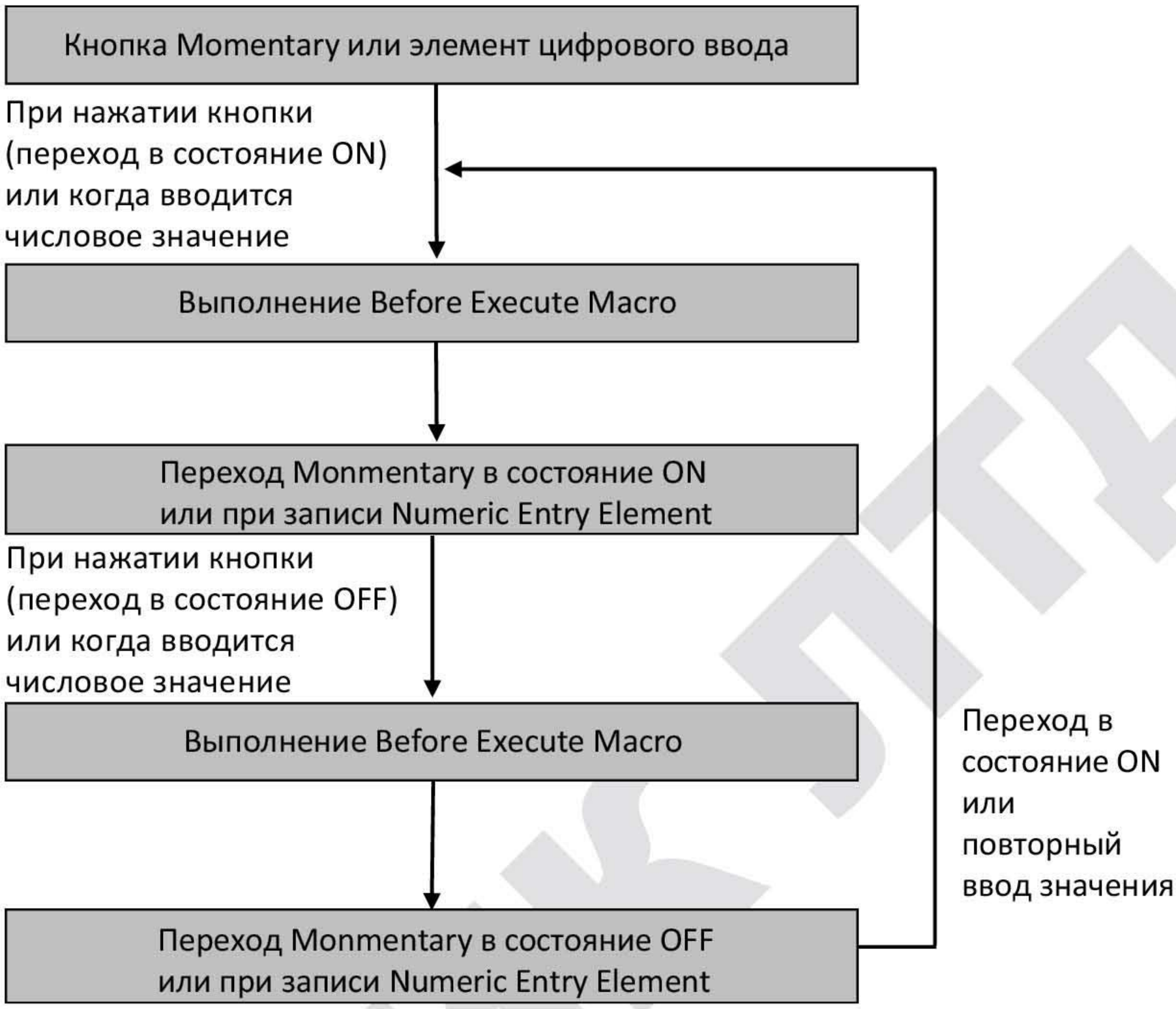
3.14.1.3 Before Execute Macro

Этот макрос связан с элементом кнопка или элементом цифрового /знакового ввода, причём каждому элементу соответствует один макрос **Before Execute Macro**.

Когда кнопочный элемент нажат, сначала выполняются команды макроса, а далее выполняется действие данного элемента.

Но, если изменение состояние элемента не производится нажатием, (а производится макросом или внешним сигналом), то команды макроса не выполняются.

Ниже приводится процедура при нажатии элемента **Momentary** или элемента цифрового ввода:



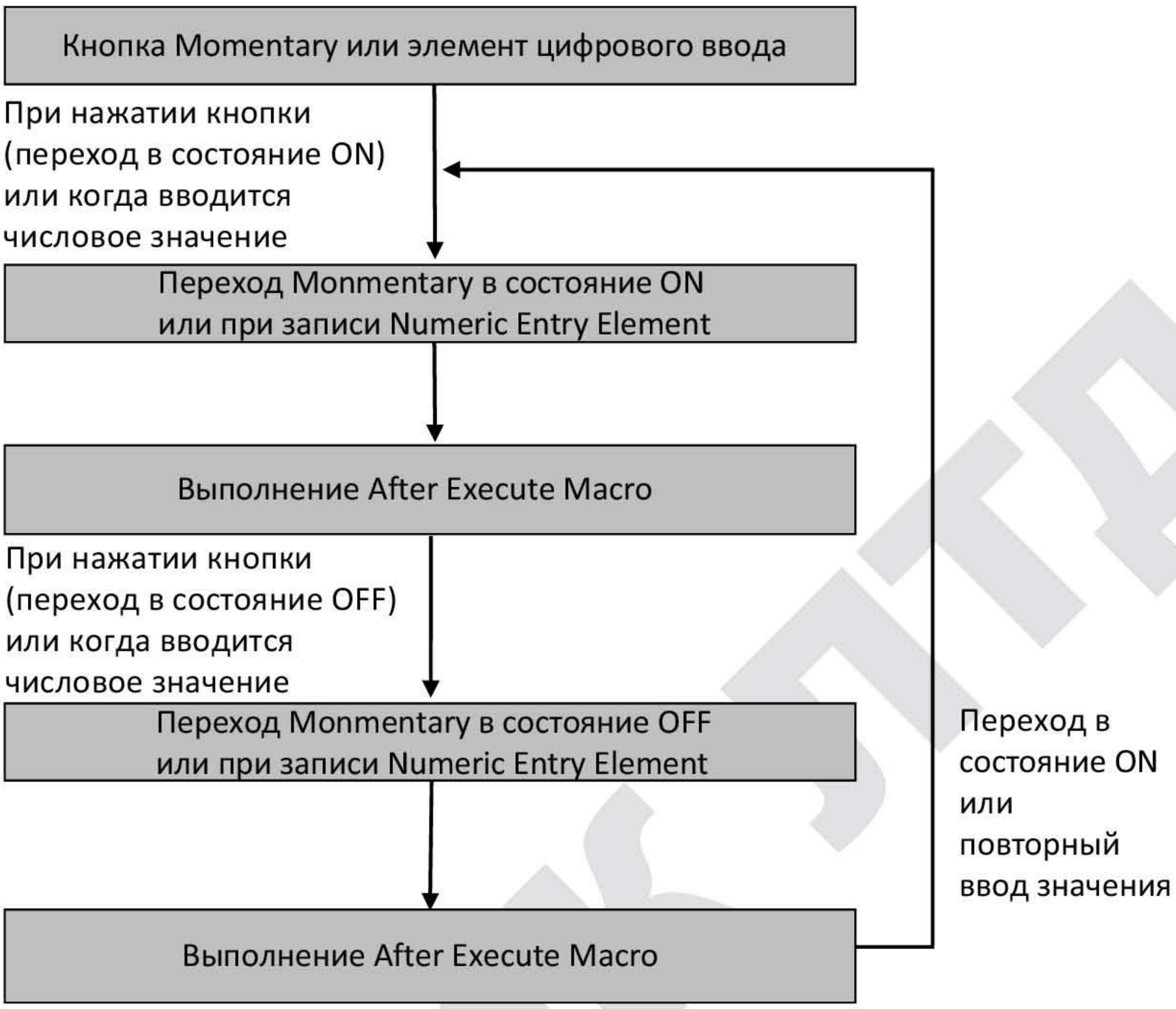
3.14.1.4 After Execute Macro

Этот макрос связан с элементом кнопка или элементом цифрового /знакового ввода, причём каждому элементу соответствует один макрос **After Execute Macro**.

Когда кнопочный элемент нажат, сначала выполняются команды макроса, а далее выполняется действие данного элемента.

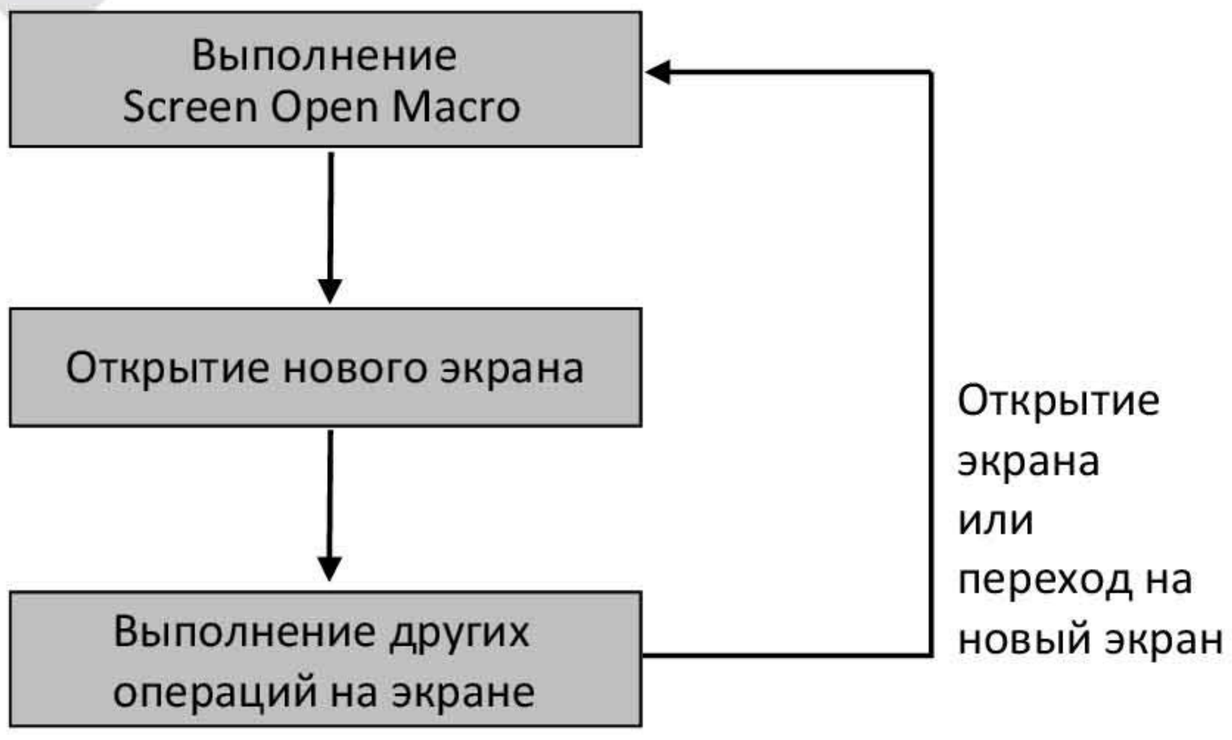
Но, если изменение состояние элемента не производится нажатием, (а производится макросом или внешним сигналом), то команды макроса не выполняются.

Ниже приводится процедура при нажатии элемента **Momentary** или элемента цифрового ввода:



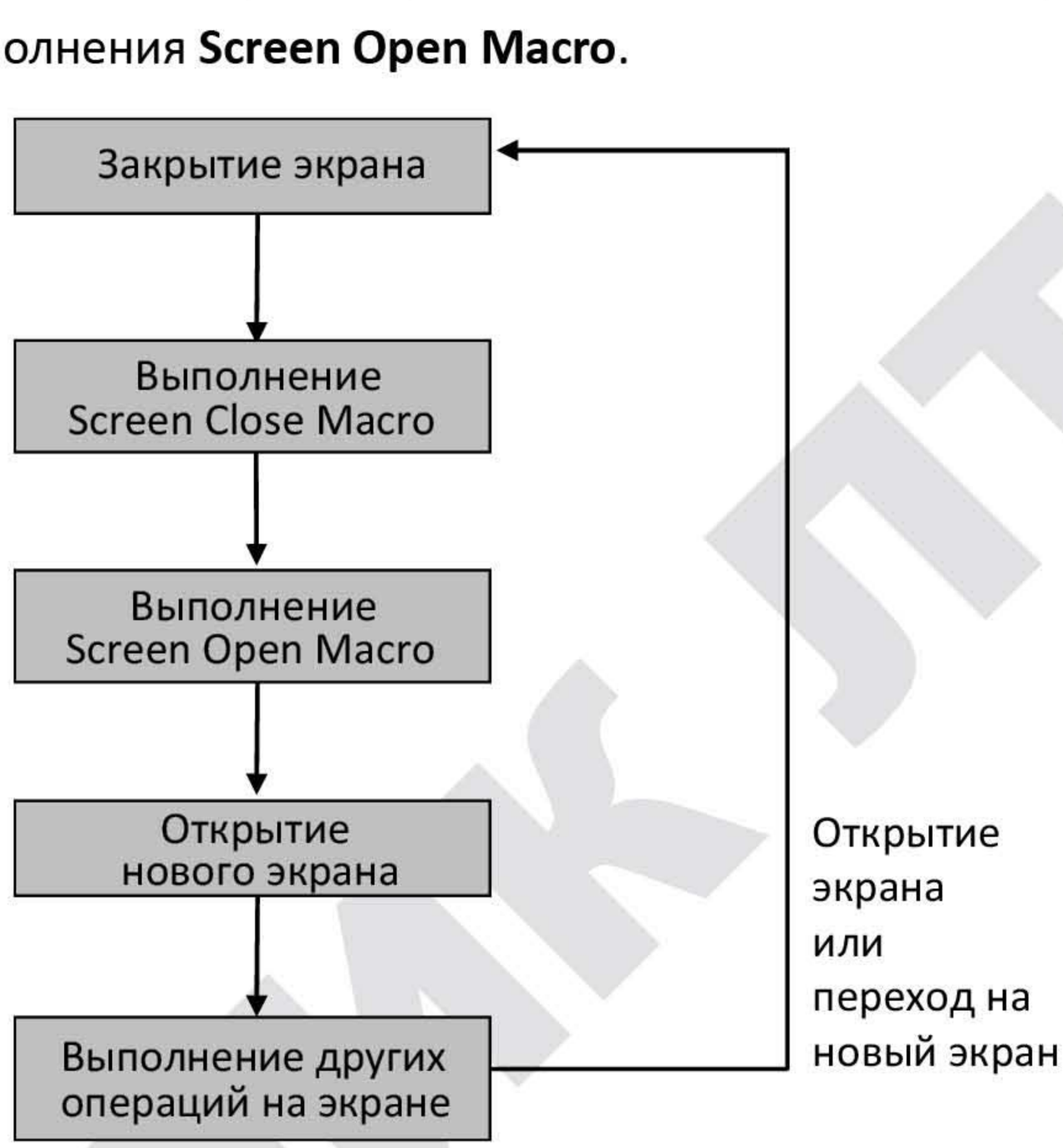
3.14.1.5 Screen Open Macro

Пользователь может использовать **Screen Open Macro** при открытии каждого экрана. Этот макрос выполняется однократно при открытии экрана или при переключении на новый экран. Экранные элементы выполняются после выполнения **Screen Open Macro**.



3.14.1.6 Screen Close Macro

Пользователь может использовать **Screen Close Macro** при закрытии каждого экрана. Этот макрос выполняется однократно при закрытии экрана или при переключении на новый экран. Экранные элементы нового экрана выполняются только после выполнения **Screen Open Macro**.



3.14.1.7 Screen Cycle Macro

Пользователь может использовать **Screen Cycle Macro** при открытом экране для любого экрана. Для каждого экрана может периодически выполняться **Screen Cycle Macro**. Пользователь может задать время **Cycle Delay Time**, то есть время задержки между завершением выполнения макроса и повторным его запуском в диалоговом окне **Screen Properties** (См.Рис. 3-14-2). Значение **Cycle Delay Time** по умолчанию установлено 100ms.

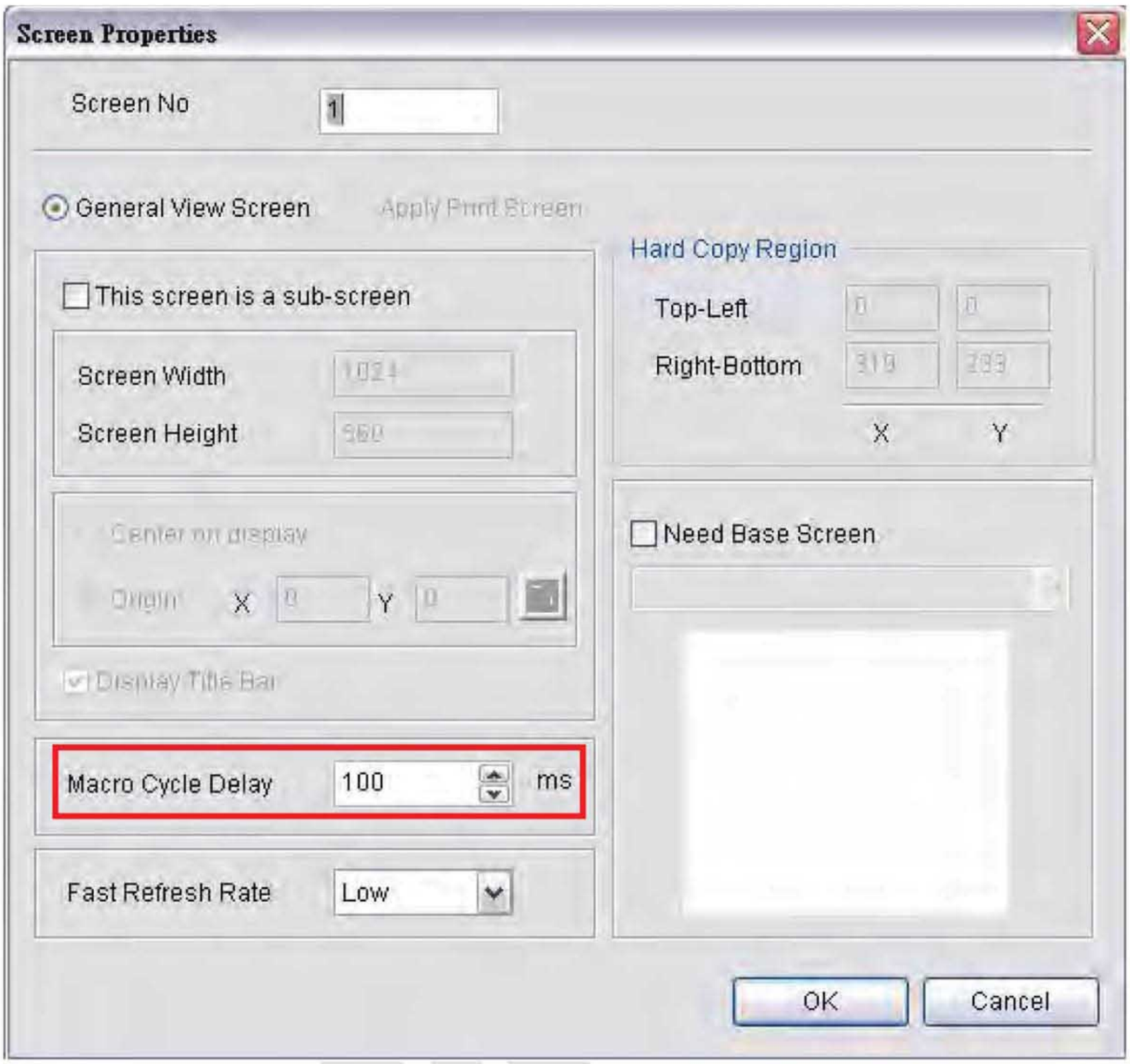
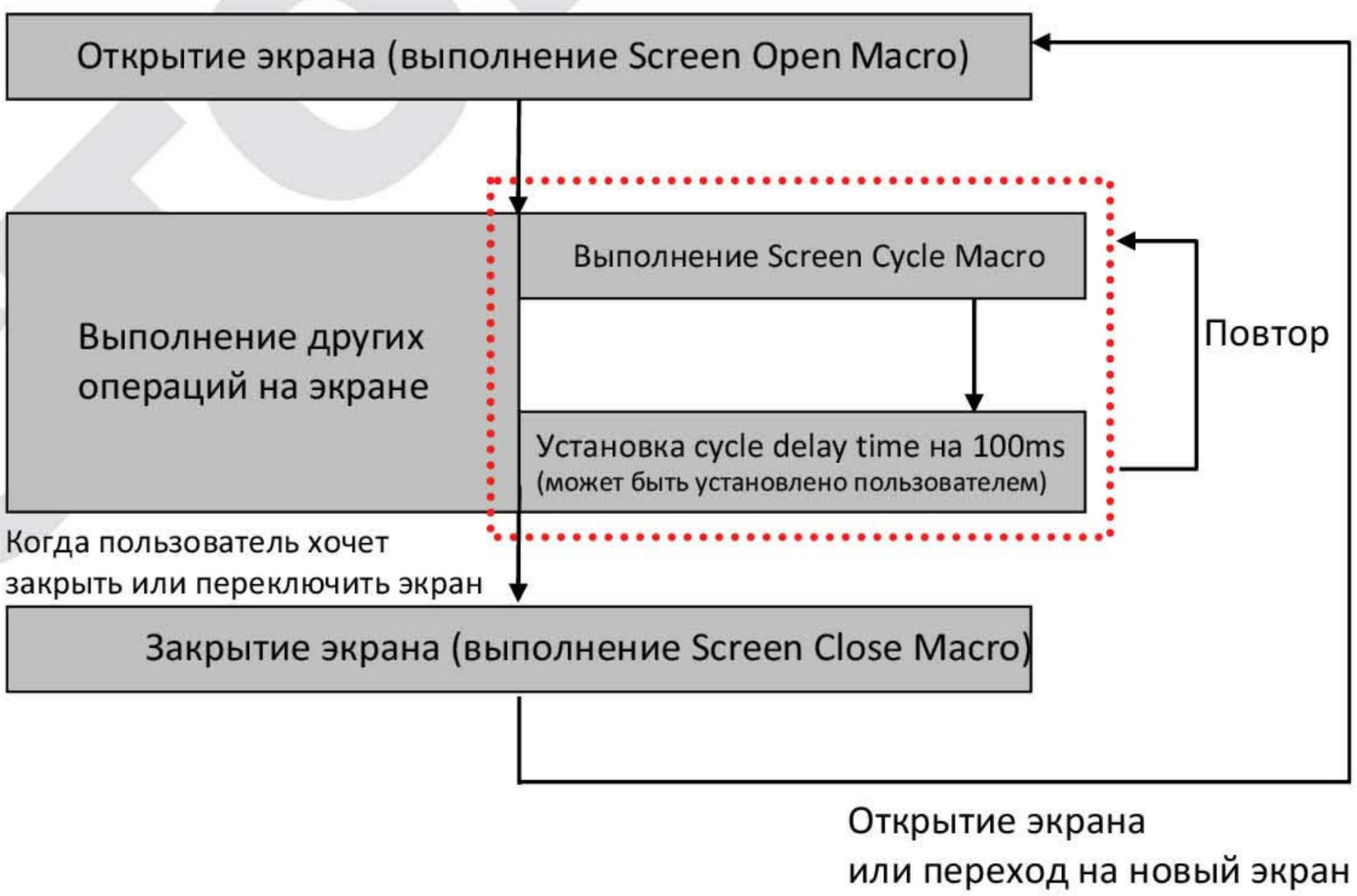


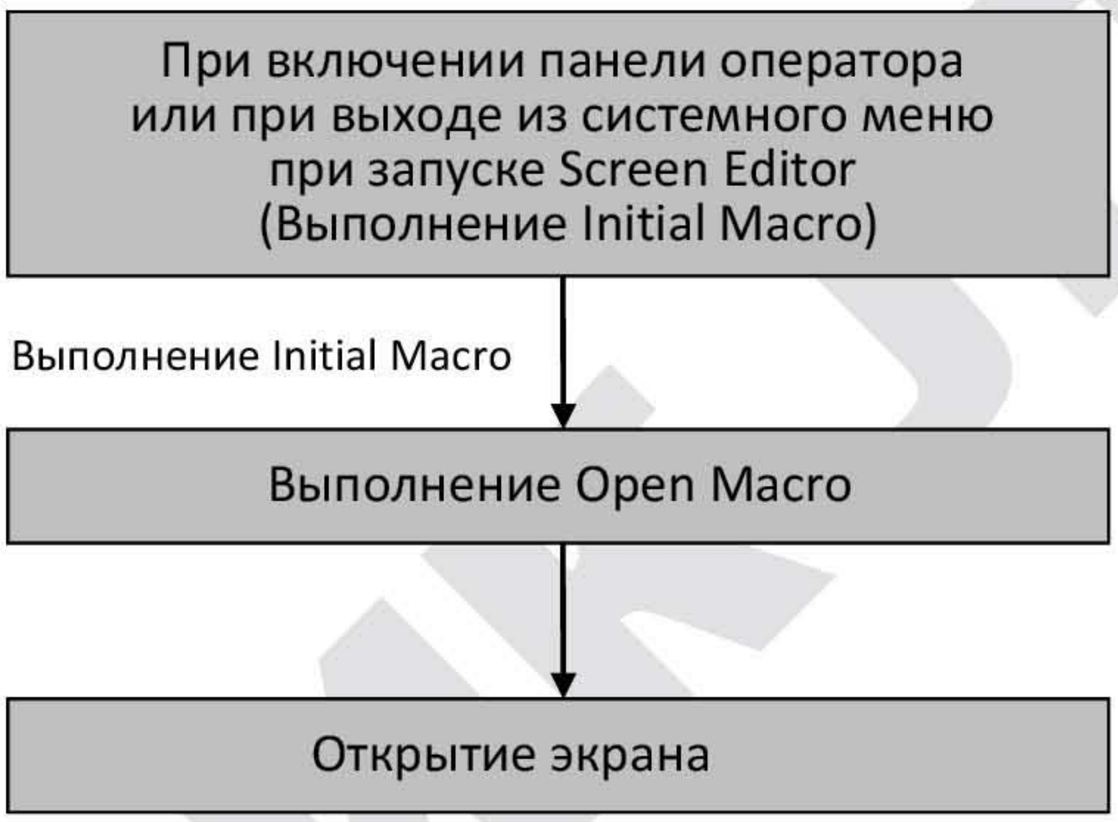
Рис. 3-14-2 Время задержки Macro Cycle



3.14.1.8 Initial Macro

В программе может выполняться один макрос **Initial Macro**. Он выполняется первым в начале программы. Пользователь может установить в начале работы необходимые начальные настройки, вместо того, чтобы устанавливать их при выполнении программы. Таким образом можно предотвратить выполнение программы с неопределёнными начальными параметрами.

Так же, при необходимости установить в специальных регистрах внешнего контроллера начальные значения может быть использован этот макрос. Хорошо спроектированный **Initial Macro** позволяет сократить время создания программы.



3.14.1.9 Background Macro

В программе может выполняться только один макрос **Background Macro**.

Он выполняется всегда при работе панели оператора. Одна или несколько команд выполняются непрерывно, причём после завершения их выполнения происходит повторный старт **Background Macro**.

Кликнув мышью **Options > Configuration > General tab** и спользуя опцию **Background macro update** можно задать число выполняемых строк за 1 скан программы (Рис . 3-14-3)

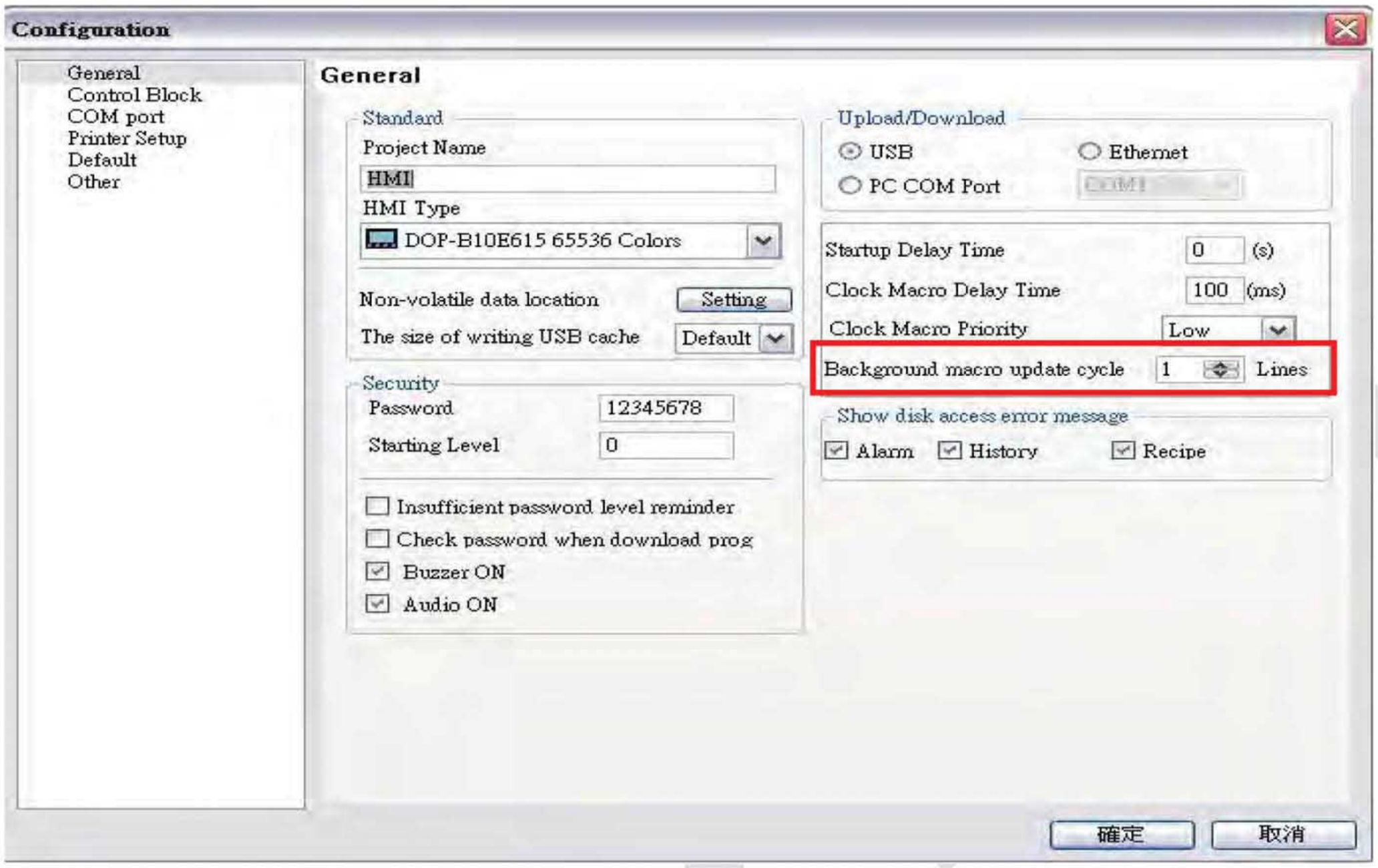


Рис. 3-14-3 Цикл обновления Background macro

Например, если имеется 25 адресуемых элементов, и **Background** использует 5 строк, то при установке цикла обновления 2, при выполнении **background macro** панель оператора сначала прочитает 25 адресов, далее выполнит последовательно 2 строки.

3.14.1.10 Clock Macro

В програме имеется только один макрос **Clock Macro**. Аналогично экранным макросам **Cycle Macro**, он может выполняться постоянно, в соответствии с установленным периодом. Пользователь может задать время **Cycle Delay Time**, то есть время задержки между завершением выполнения макроса и повторным его запуском в диалоговом окне **Configuration > General** (Рис. 3-14-4). По умолчанию установлено 100ms.

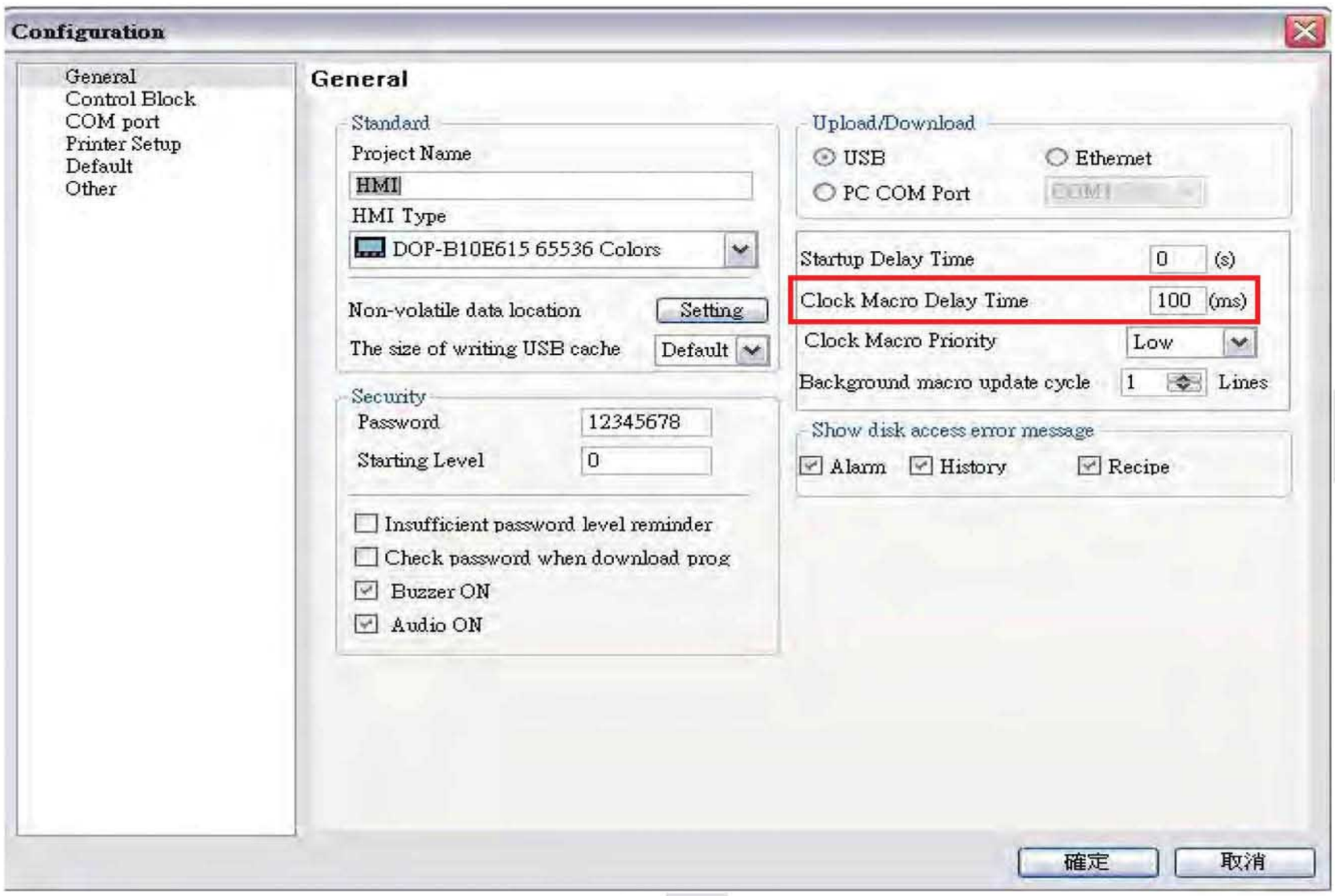
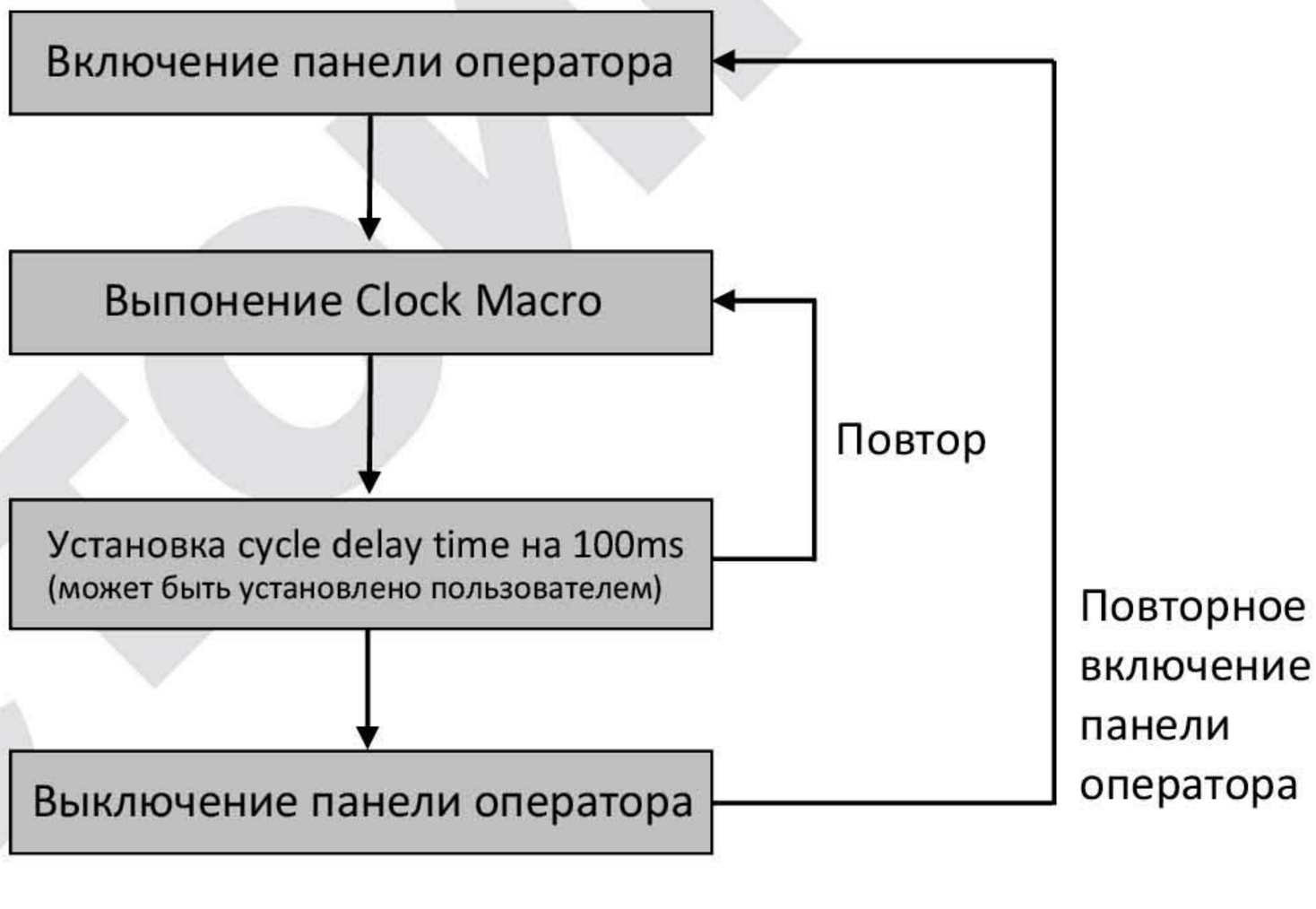


Рис. 3-14-4 Время задержки Clock Macro



3.14.1.11 Sub-Macro

Каждый проект может иметь в своём составе до 512 суб-макросов, с адресацией по умолчанию 1 … 512 (Рис. 3-14-5, Рис. 3-14-6). **Sub-macro** аналогично подпрограммам в программе.

Для сокращения времени написания макросов пользователь может записать

простые повторяющиеся действия в **sub-macro**. Например, для обращения к суб-макросу 2, для его выполнения пользователю в программе или макросе необходимо написать **CALL 2**.

Суб-макросы можно вызывать и из других суб-макросов, причём уровень вложенности не может быть более 6-ти.

По умолчанию суб-макросы обозначены числами 1 … 512, но пользователь может их именовать иначе, по своему усмотрению.

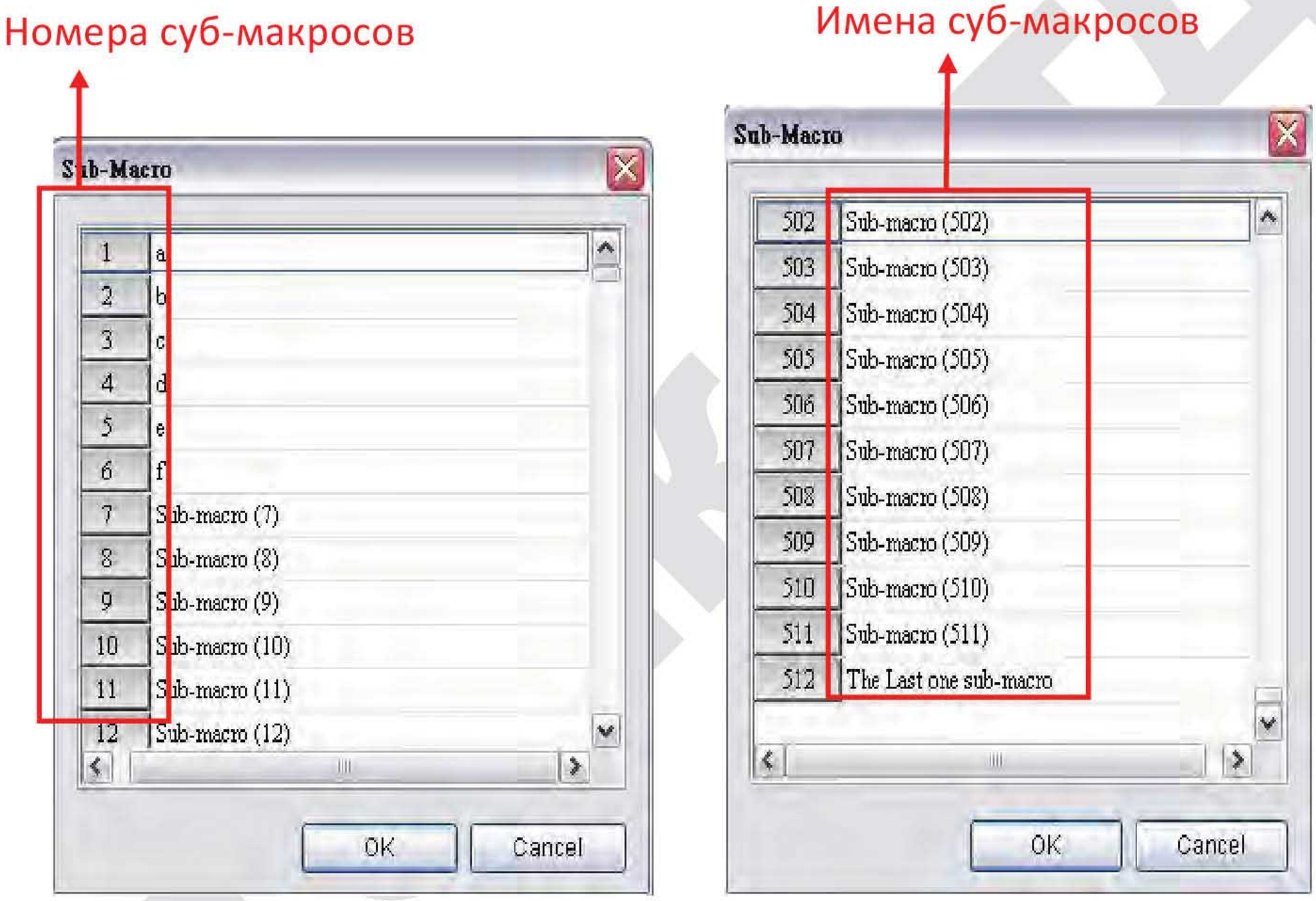


Рис. 3-14-5 Суб-макросы Экран 1

Рис. 3-14-6 Суб-макросы Экран 2

Пример:

Открыть **Screen Open Macro** и **Sub-Macro 1** (Рис.3-14-7, Рис. 3-14-8).

Процесс выполнения **Screen Open Macro** показан на Рис. 3-14-9.

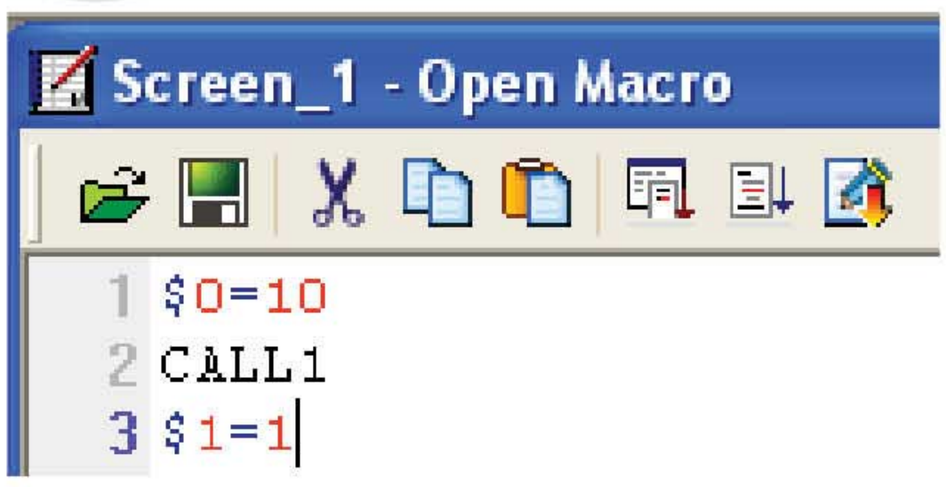


Рис. 3-14-7 Экран Open Macro

Рис. 3-14-8 Sub-Macro 1

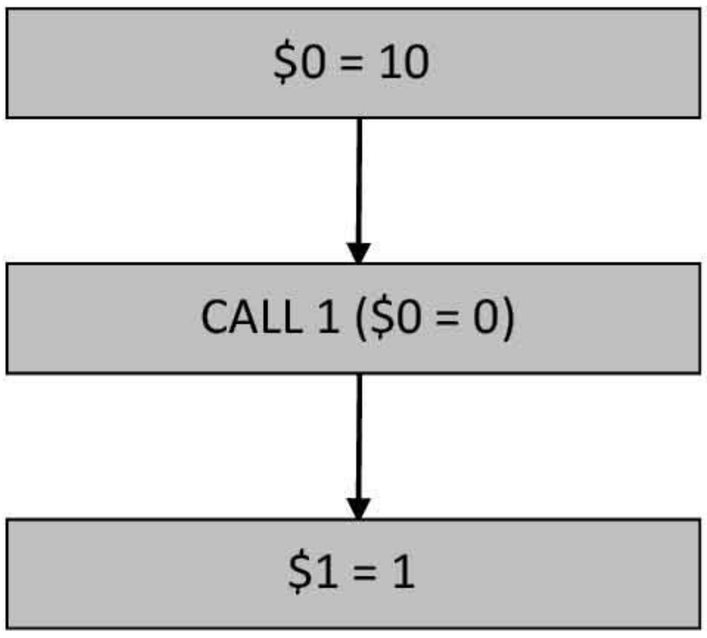


Рис. 3-14-9 Процесс выполнения Screen Open Macro

Выполнение **CALL 1** означает, что выполняется **Sub-Macro 1**, после чего продолжается выполнение **Screen Open Macro** до конца.

В результате, $0 = 0 и $1 = 1.

ПРИМЕЧАНИЕ

При редактировании макросов необходимо обратить особое внимание на циклические макросы и программы, где при невыполнении какого-либо условия программа может остановиться и панель оператора начнет работать неправильно. Рекомендуется проверять работу с помощью симулятора.

3.14.2 Редактирование макросов

3.14.2.1 Окно редактирования макросов и панель инструментов

После выбора необходимой макрокоманды, пользователь, кликнув мышью по любой строке в окне редактирования макросов может начать редактирование как показано на рис. 3-14-10.

Каждый макрос может содержать до 512 командных строк длиной не более 128 символов.

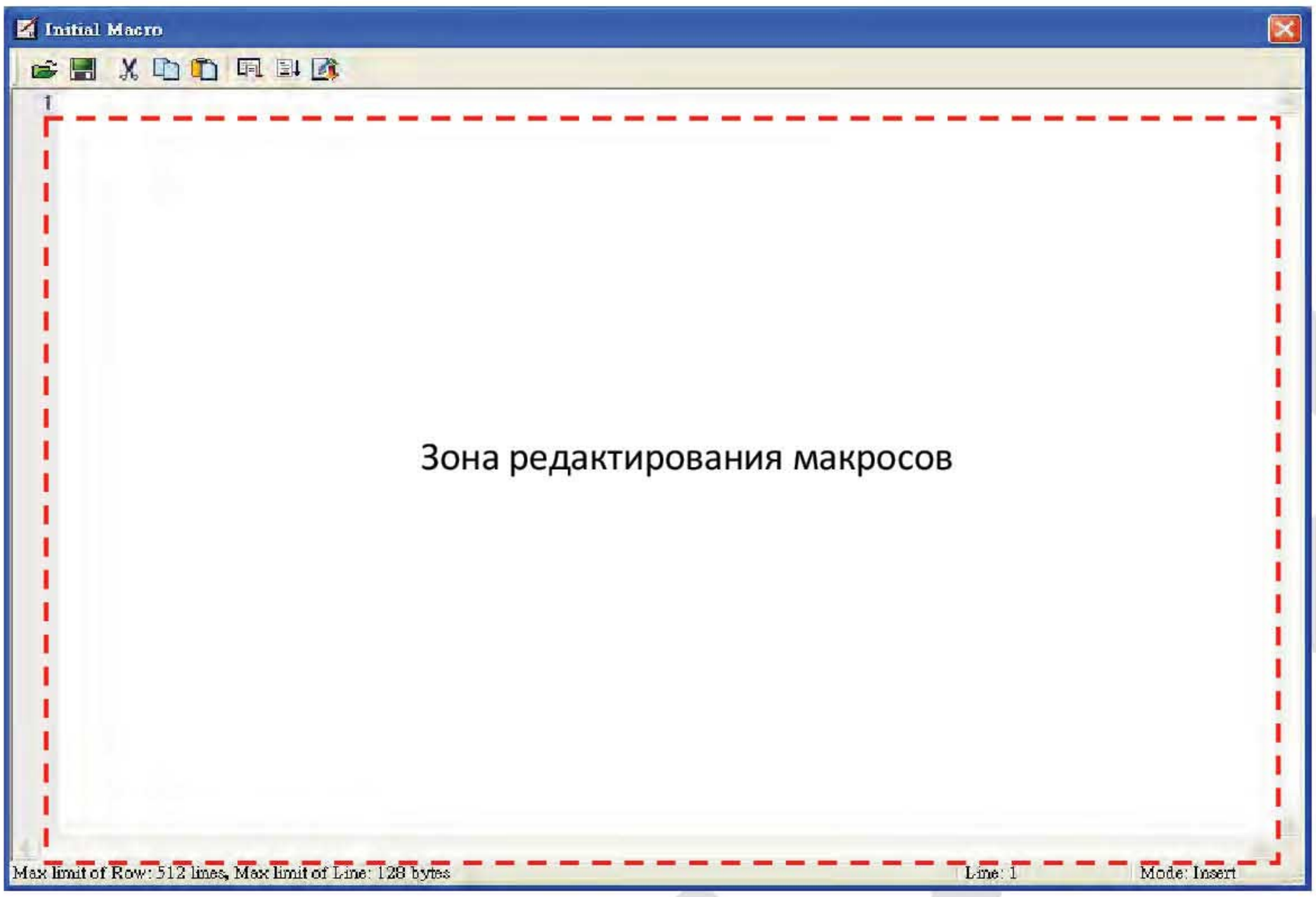


Рис. 3-14-10 Окно Macro Editing

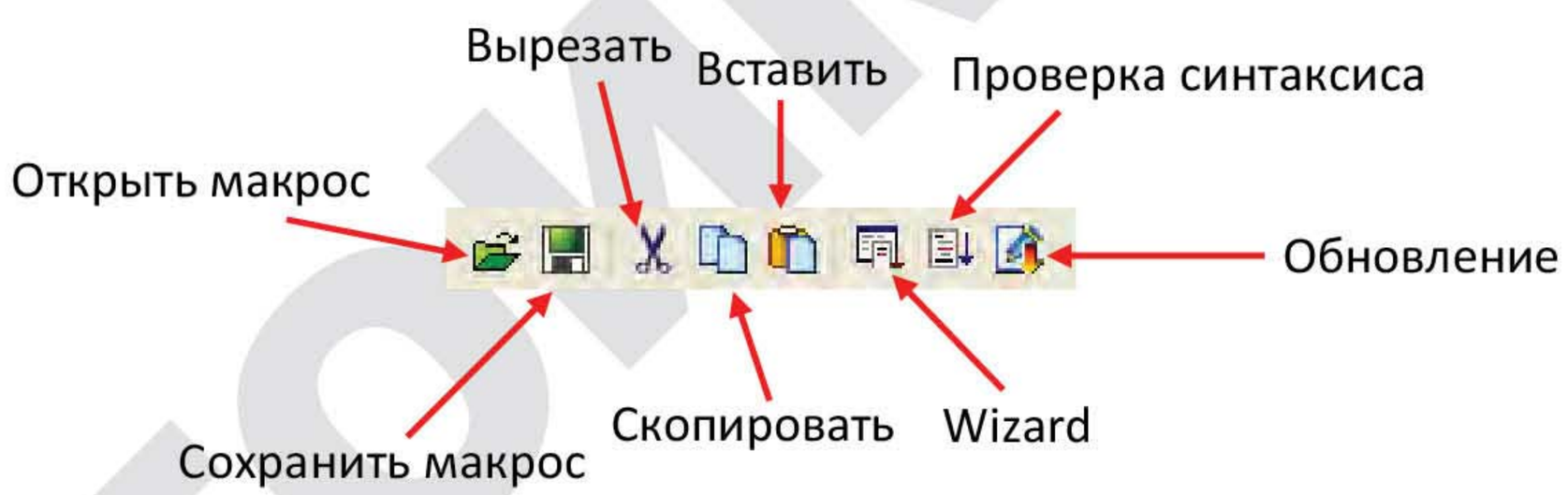


Рис. 3-14-11 Панель инструментов

Open

Этой командой открывается файл макроса.

Пользователь может открывать эти файлы когда используются различные контроллеры, что уменьшает время редактирования. Кликнув по значку для открытия файла появится диалоговое окно, как показано на Рис. 3-14-12

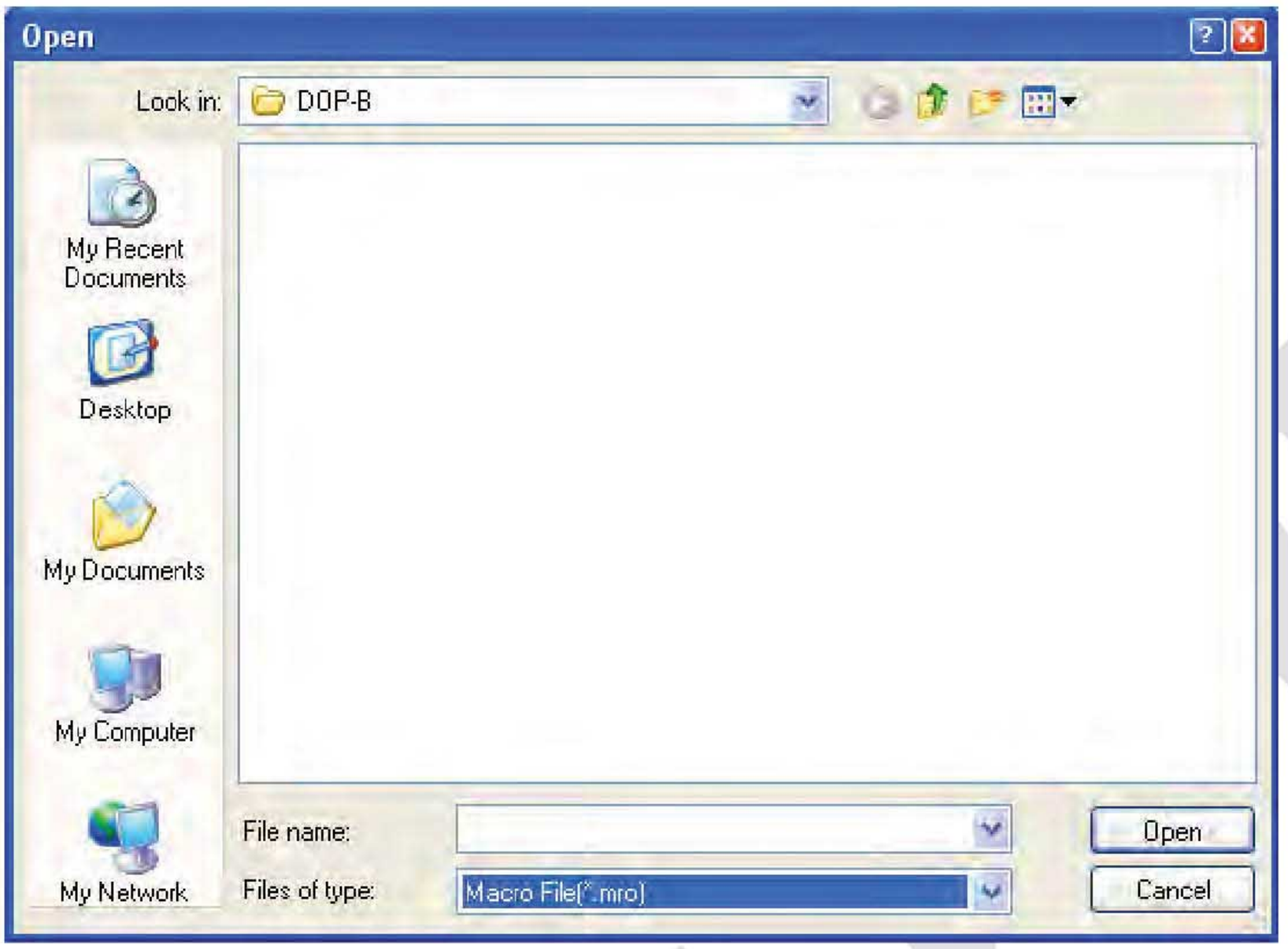


Рис. 3-14-12 Открытие макроса

Save

Этой командой сохраняется файл макроса.

Пользователь может переименовать файл. Это даёт возможность сохранить макрос и сократить время ввода других макросов. (Рис. 3-14-13).

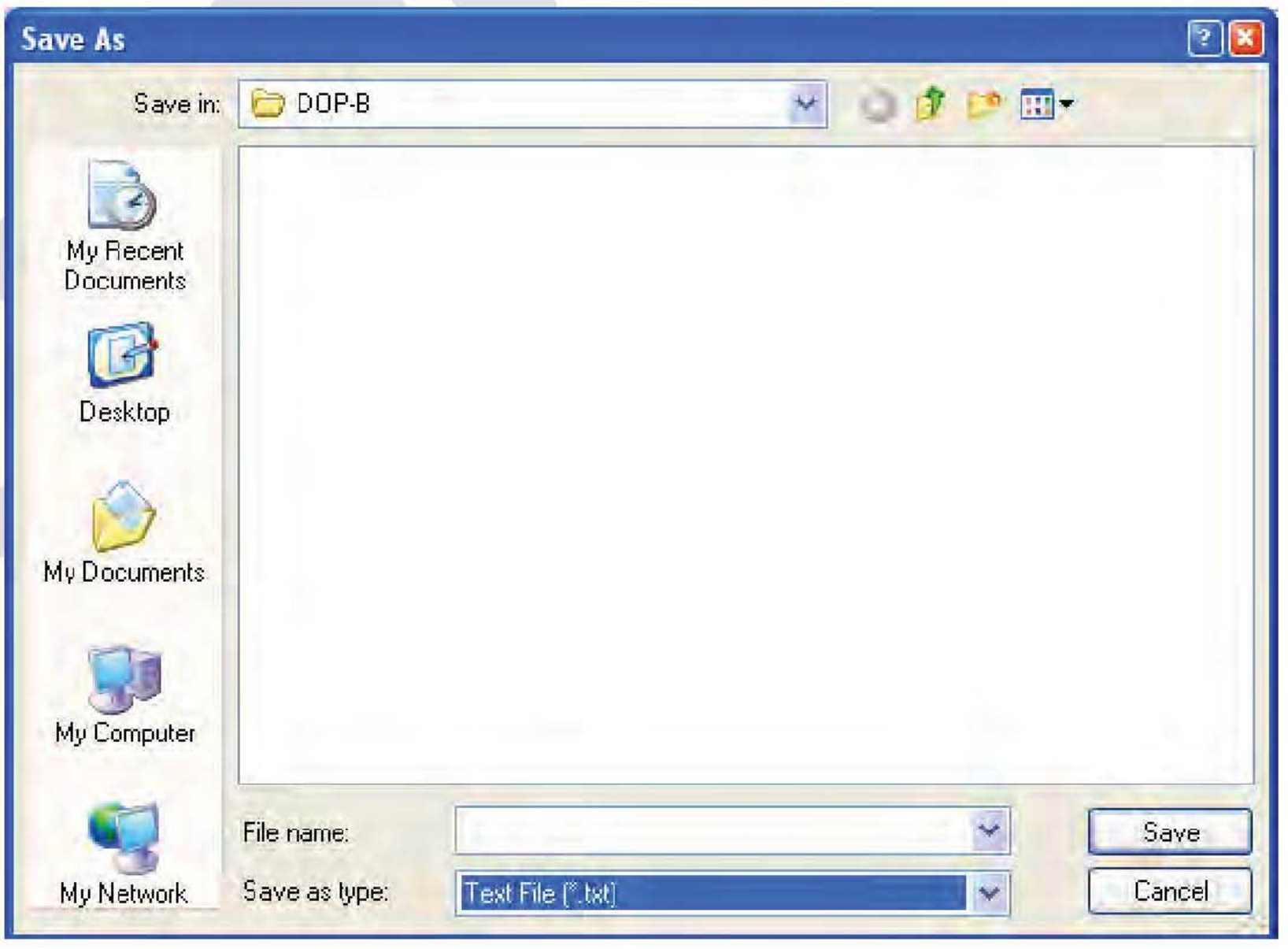


Рис. 3-14-13 Сохранение макроса

Cut

 Этой командой открывается файл макроса.

Также это можно выполнить нажатием **Ctrl+X**.

Copy

 Этой командой копируется файл макроса.

Также это можно выполнить нажатием **Ctrl+C**.

Paste

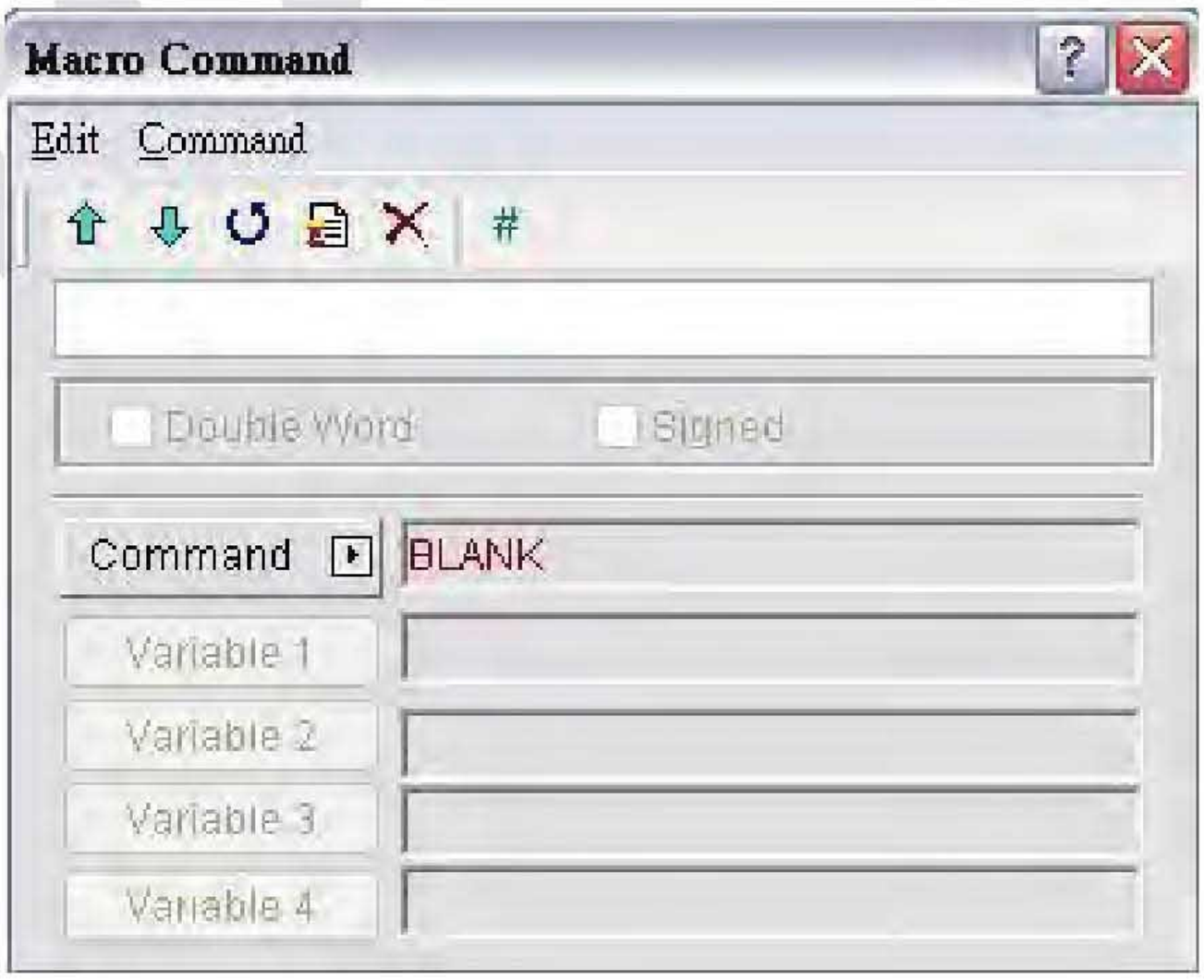
 Этой командой вставляется файл макроса.

Также это можно выполнить нажатием **Ctrl+V**.

Wizard

 Этой командой открывается диалоговое окно редактирования макросов.

Выбрав данный значок, появится диалоговое окно для редактирования макросов. При использовании в макросе адреса внешнего контроллера, он будет в скобках, для отличия его от адресов внутренней памяти.



Пользователь может ввести команду непосредственно или воспользоваться кнопкой **Command** для выбора нужной команды. Подробнее. см. раздел 3.14.2.2.

Syntax check

 С помощью этой команды проверяется правильность синтаксиса макроса.

При выявлении ошибки появляется диалоговое сообщение об ошибке с указанием строки, которую необходимо проверить.

Проверка синтаксиса и компиляция макроса являются разными операциями. Компилляция макроса проводится при выполнении компиляции программы панели оператора.

Update

 Этой командой подтверждается запись изменений в макрос.

Появится следующее сообщение при нажатии кнопки **Update**.

При выборе **Yes** система проверит синтаксис макроса. Если ошибок нет то произойдёт его запись и сохранение. В противном случае появится диалоговое окно с сообщением об ошибке и будет предложено пользователи проверить соответствующую строку.

3.14.2.2 Диалоговое окно макрокоманд

Если кликнуть мышью на  , то на дисплее автоматически появится диалоговое окно **Macro Command** (Рис. 3-14-14).

Панель инструментов редактирования макросов

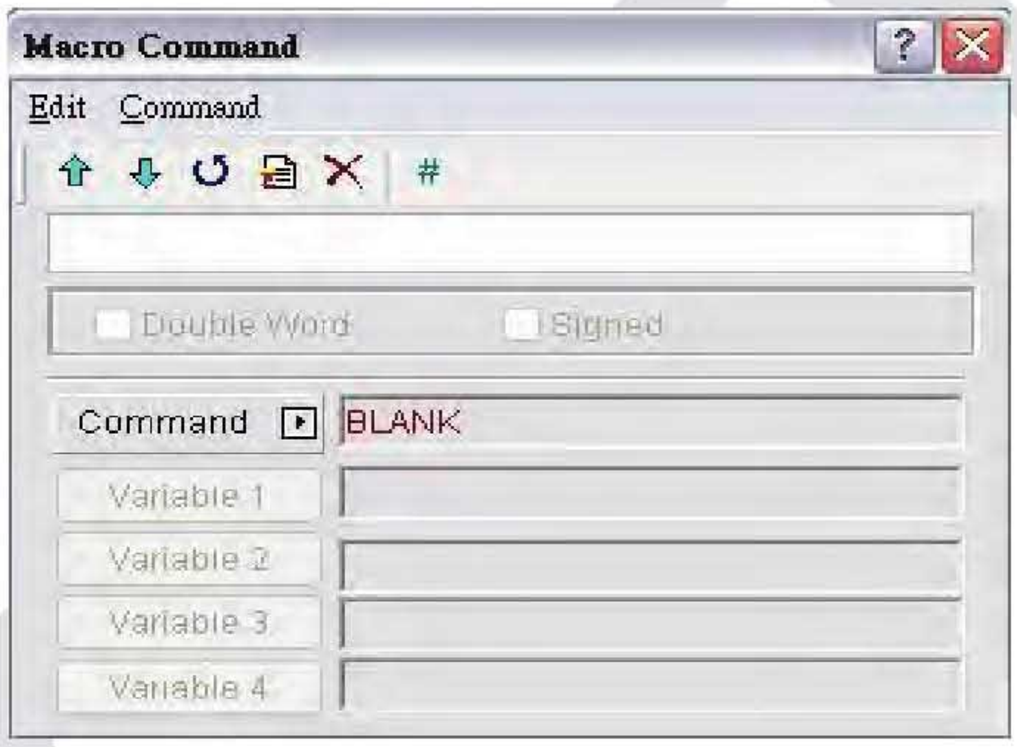


Рис. 3-14-14 Диалоговое окно Macro Command

Edit

Пользователь может редактировать файл макроса с помощью меню редактирования в диалоговом окне Macro Command (Рис. 3-14-15) или пользуясь панелью инструментов. (Рис. 3-14-16).

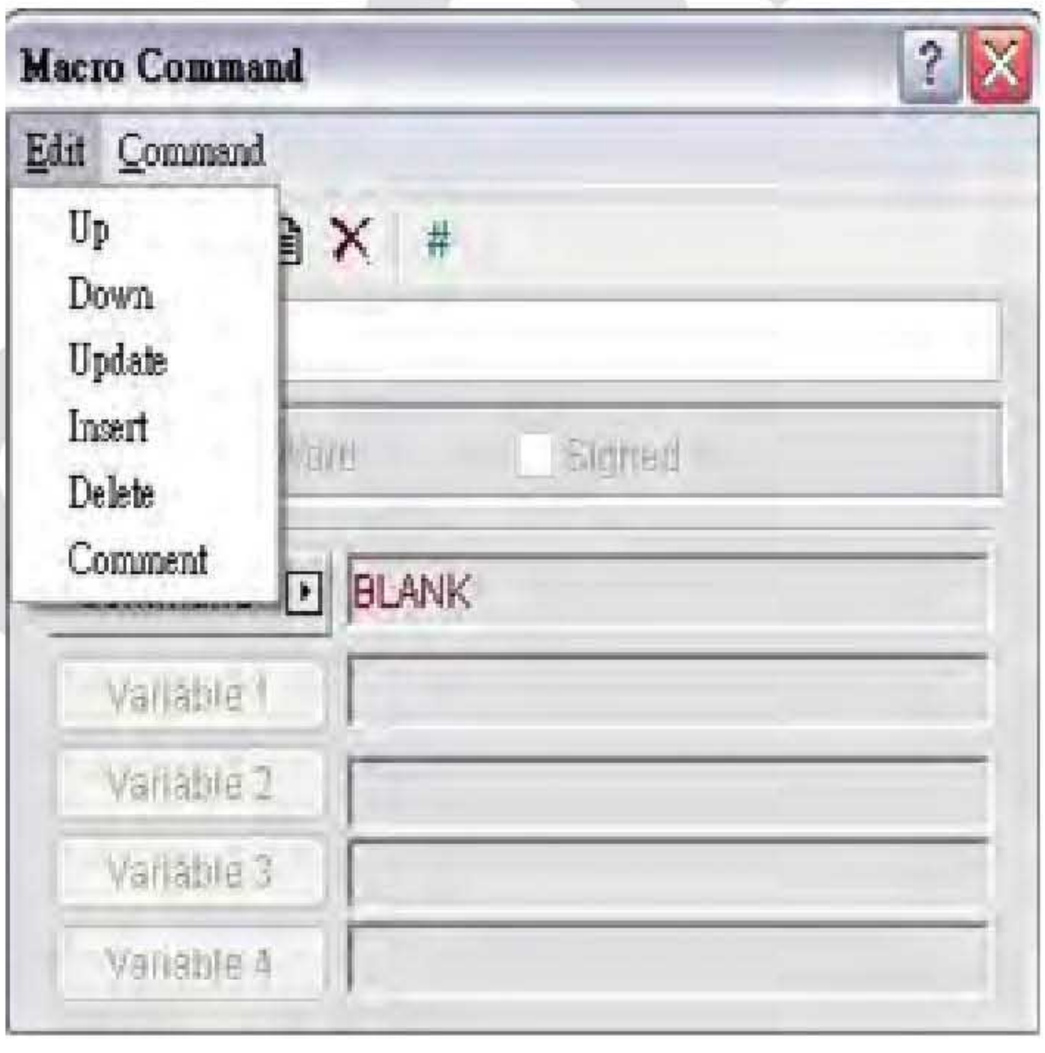


Рис. 3-14-15 Редактирование через меню

Рис. 3-14-16 Редактирование через панель инструментов

Previous

Previous  Эта команда смещает вверх редактируемую строку

Next

Next  Эта команда смещает вниз редактируемую строку

Update

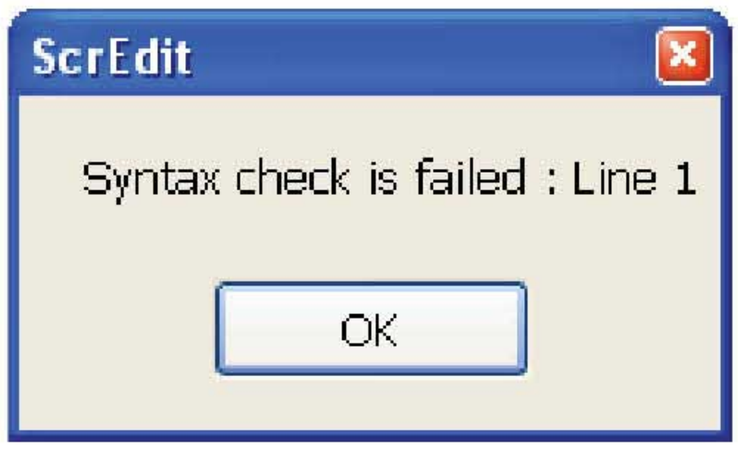
Update 

Эта команда обеспечивает обновление файлов после редактирования. Если эту функцию не использовать, то изменения не будут записаны в макрос. Если кликнуть по значку, то система проверит правильность синтаксиса макроса. Если ошибок не выявлено, то отредактированный макрос будет записан.

При попытке закрыть диалоговое окно **Macro Command** не нажимая кнопку **Update** пользователь получит предупреждающее сообщениеr.

После нажатия кнопки **Yes** система проверит правильность синтаксиса макроса. Если ошибок не выявлено, то отредактированный макрос будет записан.

При выявлении ошибок появится предупреждающее сообщение, с указанием места ошибки.



Insert

Update 

С помощью этой команды можно вставить строку с макрокомандой. Все остальные строки сдвинутся вниз на одну строку

Delete

Update 

С помощью этой команды можно удалить выбранную строку, после чего нижние строки сдвинутся вверх на одну строку

Comment

Update #

Для записи комментариев пользователь может использовать значок (#) перед началом сообщения. Можно записать сообщение длиной до конца строки. Содержимое комментария не выполняется программой.

Command

Пользователь может использовать команды для редактирования макросов. Команды и уравнения могут непосредственно записываться, или выбираться из меню (Рис. 3-14-17) или выбираться с помощью кнопки **Command** (Рис. 3-14-18).

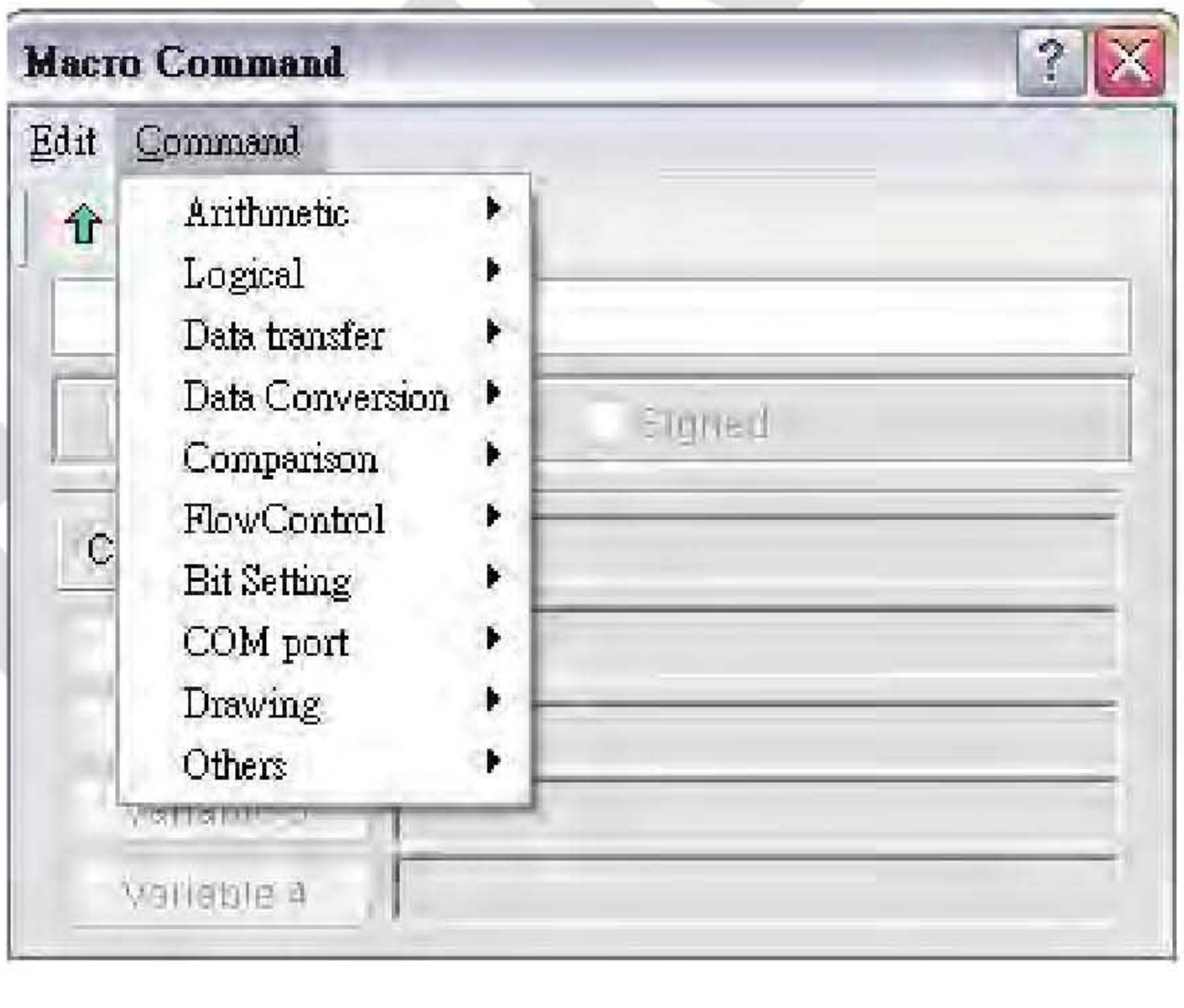


Рис. 3-14-17 Команды меню

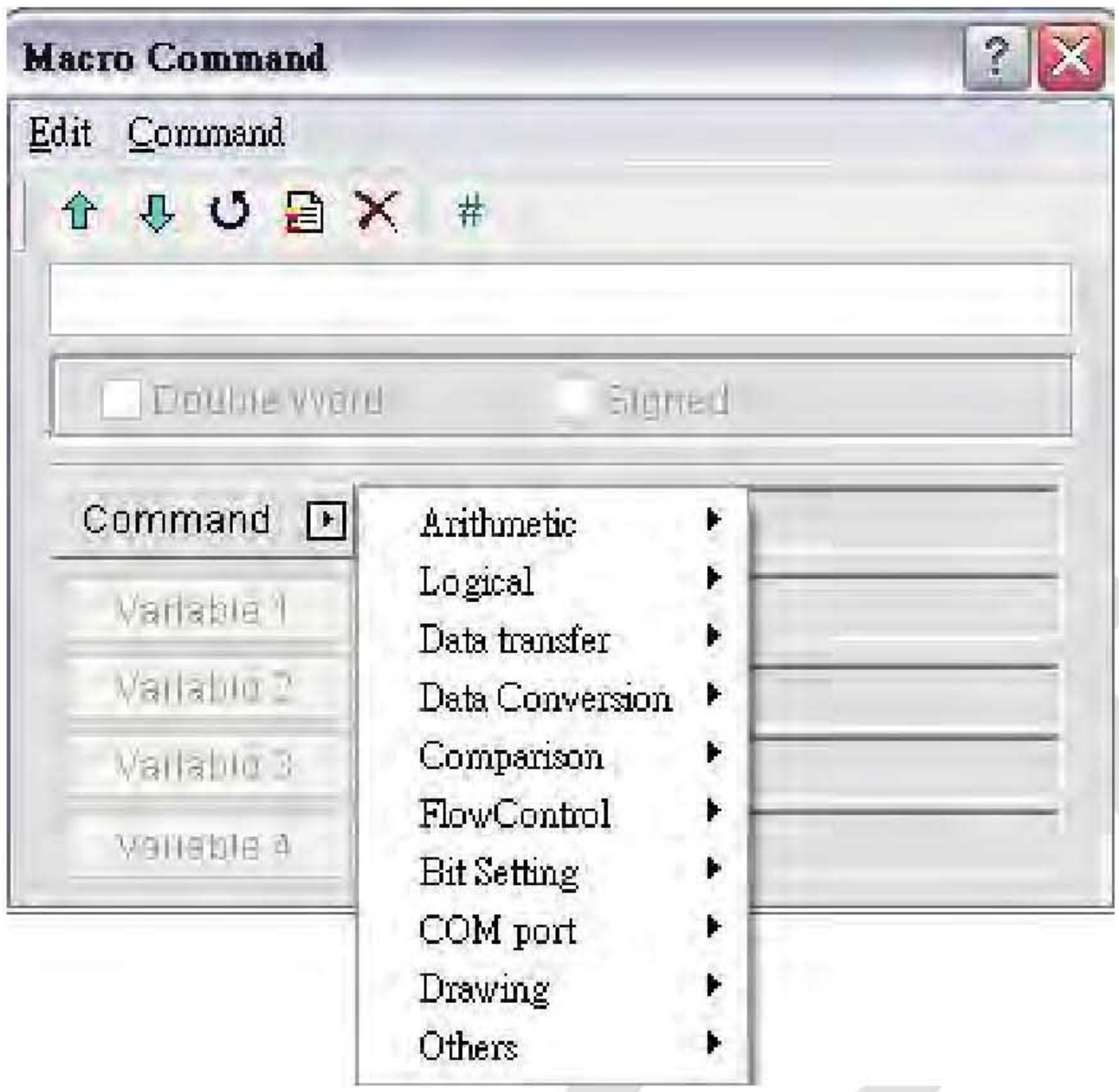


Рис. 3-14-18 Окно команд

На Рис. 3-14-20 ~ Рис. 3-14-29 показаны все доступные команды. Детальное их описание приведено в разделе 3.14.3.

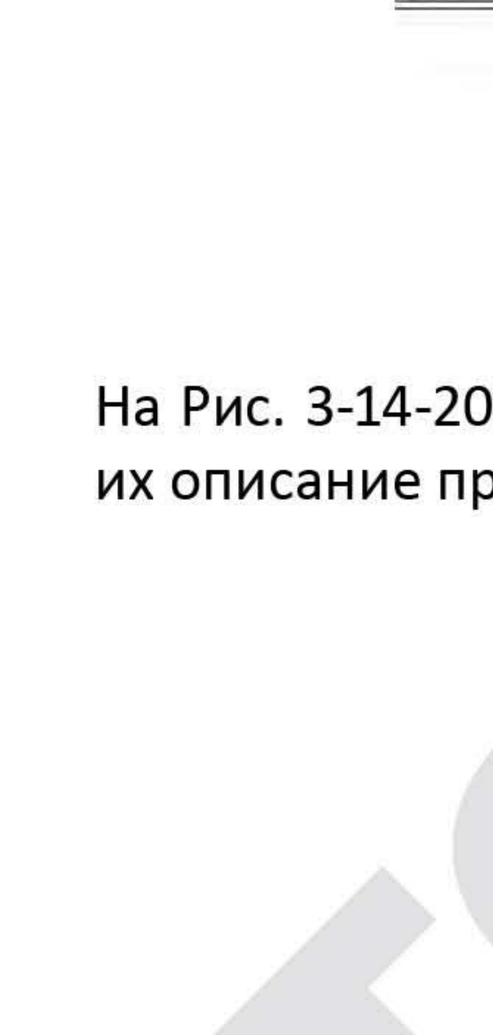


Рис. 3-14-19
Группа Команды

Рис. 3-14-20
Арифметические

Рис. 3-14-21
Логические

Рис. 3-14-22
Передача данных

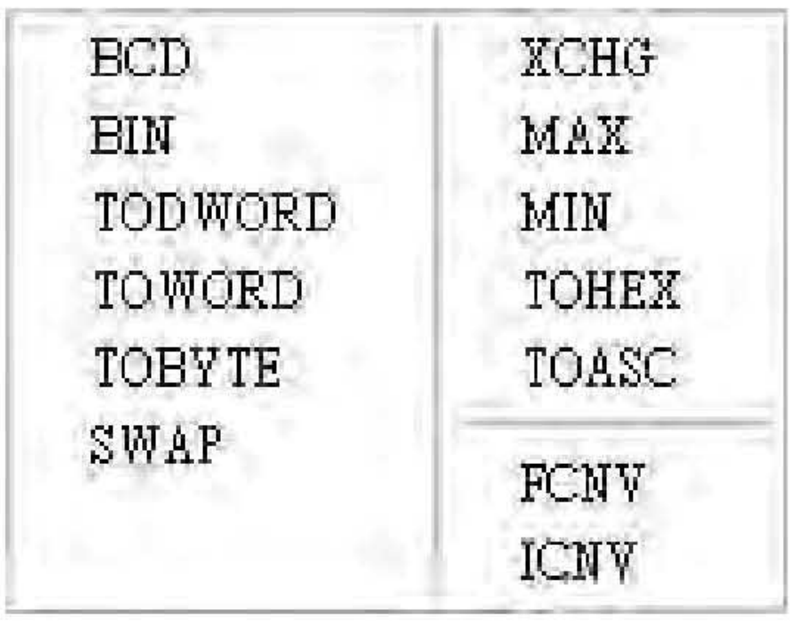


Рис. 3-14-23
Преобразование данных

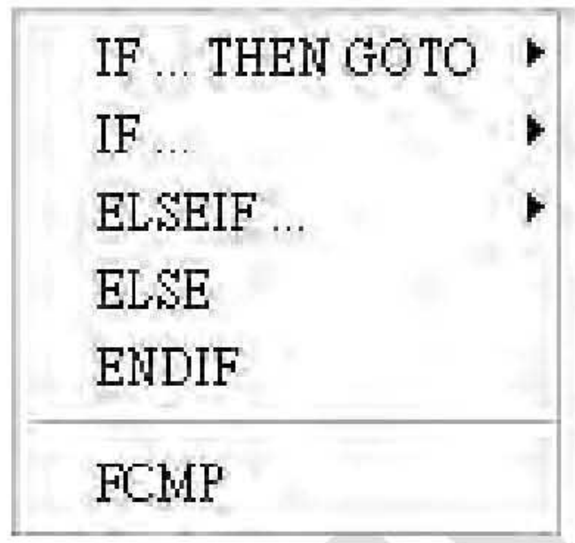


Рис. 3-14-24
Сравнение

Рис. 3-14-25
Управление циклами

Рис. 3-14-26
Установка битов

Рис. 3-14-27
Настройка COM портов

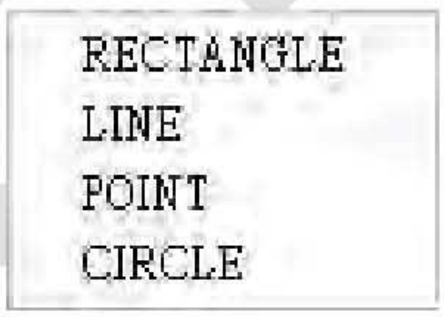


Рис. 3-14-28
Управление графикой

Рис. 3-14-29
Прочие

Задать необходимые значения операндов после выбора макрокоманды, можно, кликнув кнопку **Variable** (Var1 ~ Var4) (Рис. 3-14-30, Рис. 3-14-31).

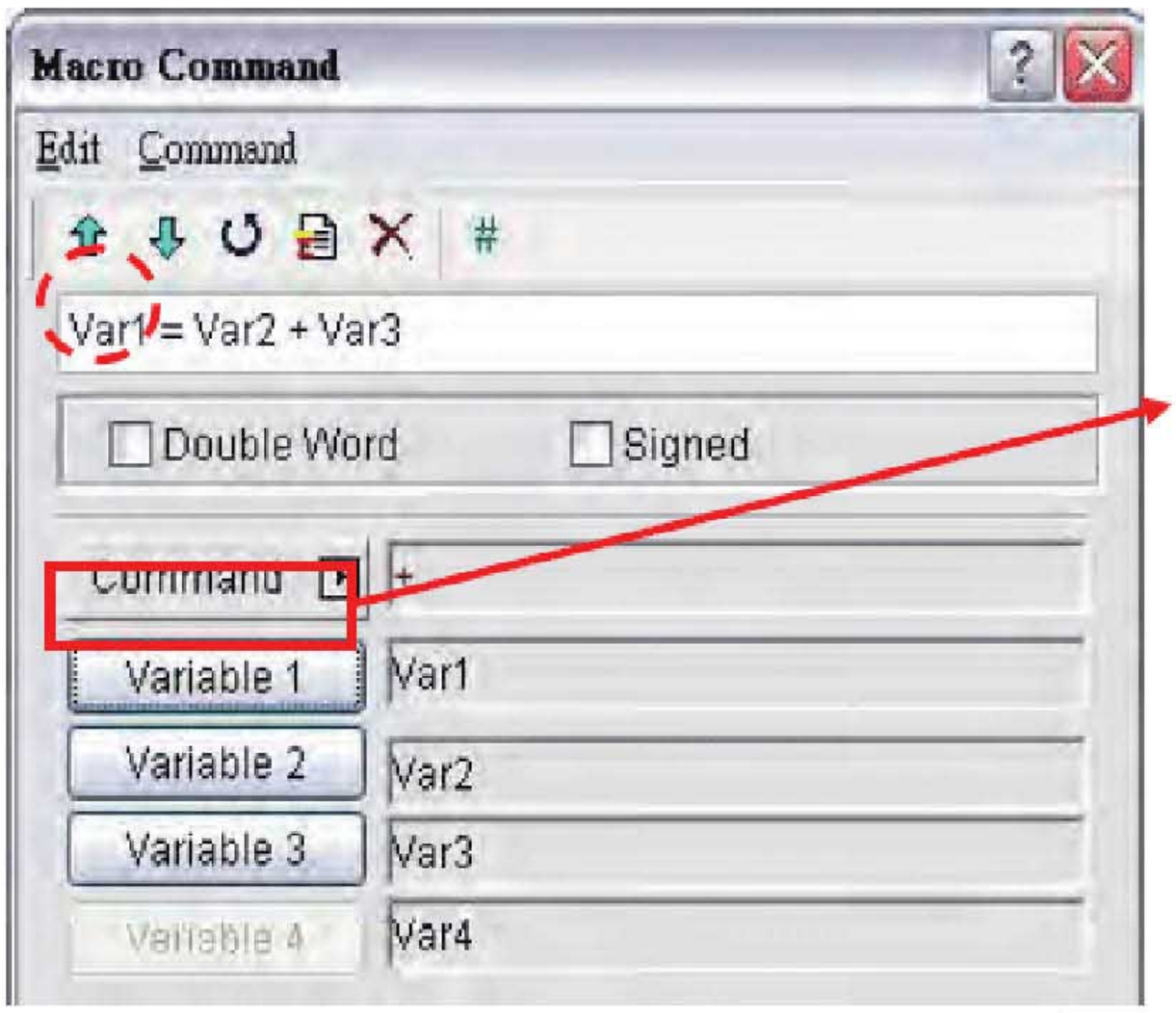


Рис. 3-14-30

Рис. 3-14-31

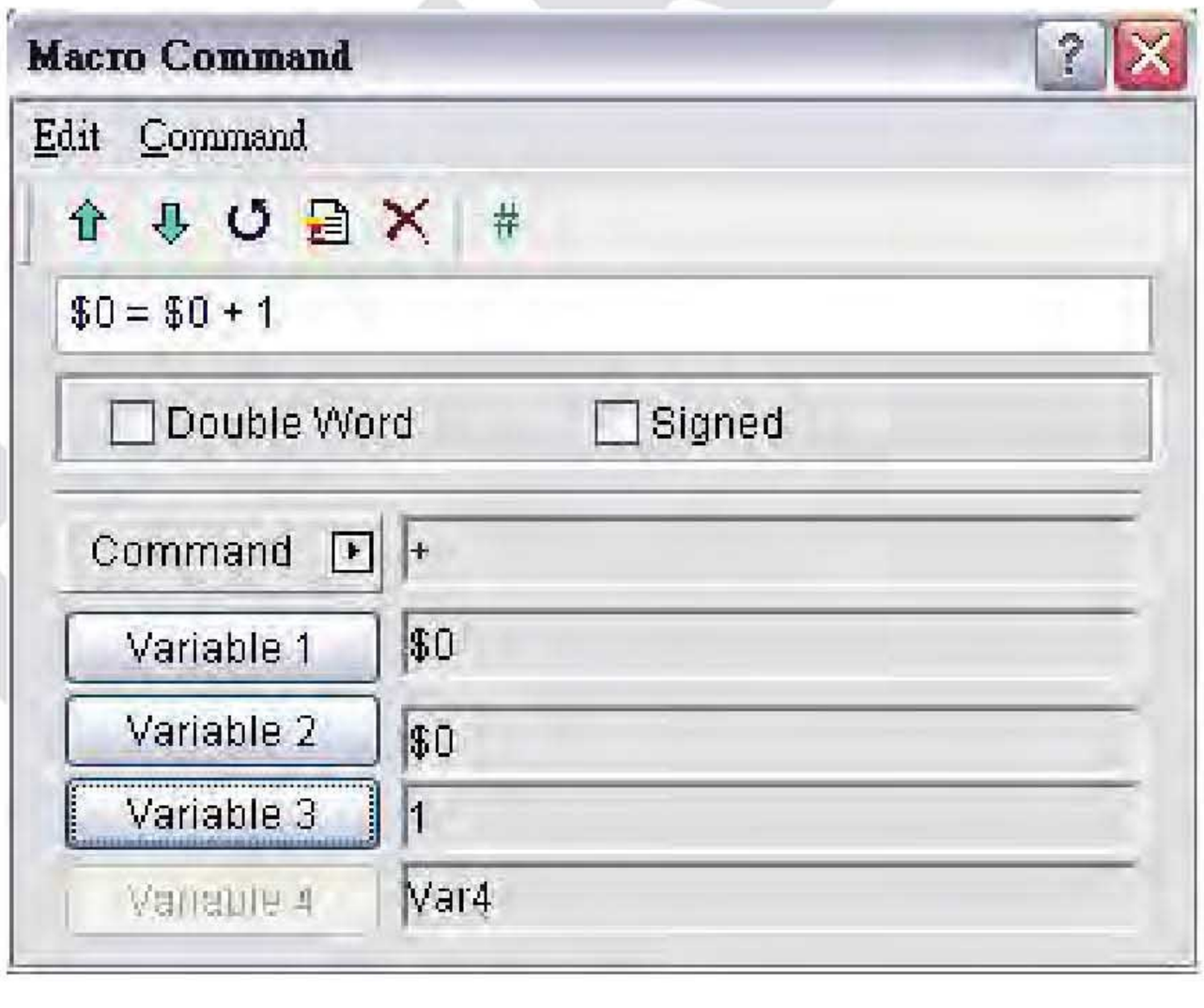


Рис. 3-14-32 Задание опций операндов: Двойное слово и знак

Таблица 3-14-2 Таблица форматов операндов

WORD (слово)	Слово состоит из 16 бит данных (или 2 байта), b15~b0. В шестнадцатеричной системе слово представляется в диапазоне 0000~FFFF.

DWORD, DW (двойное слово)	Двойное слово состоит из 32 бит данных (или 2 последовательных слова), b31~b0. В шестнадцатеричной системе двойное слово представляется в диапазоне 00000000~FFFFFFFF
BYTE (байт)	Байт состоит из 8 бит данных, b7~b0. В шестнадцатеричной системе слово представляется в диапазоне 00~FF
Signed (Число со знаком)	Данный формат позволяет учитывать знаки (+, -) чисел в операциях макропрограммирования. Если данная опция не выбрана, обрабатываемые числа могут быть только положительными.

Если после макрокоманды показано DW, это означает, что используются 32 битных операндов. Отсутствие этого означает использование 16 битных операндов. При использовании данных в формате двойных слов, этот означает использование двух регистров для его хранения.

Если задать формат данных регистра $0 как DW, то для хранения этих данных используются $0 и $1.

Если после макрокоманды показано *Signed*, это означает, что команда имеет числовое значение со знаком.

Например, когда макрокоманда записана как $0 = $2(DW), это означает, что по адресу $2 хранятся данные в виде двойного слова и при этом используются регистры $2 и $3.

При нажатии кнопки **Update**, как показано Рис.3-14-33, в окно редактирования будет записана макрокоманда.

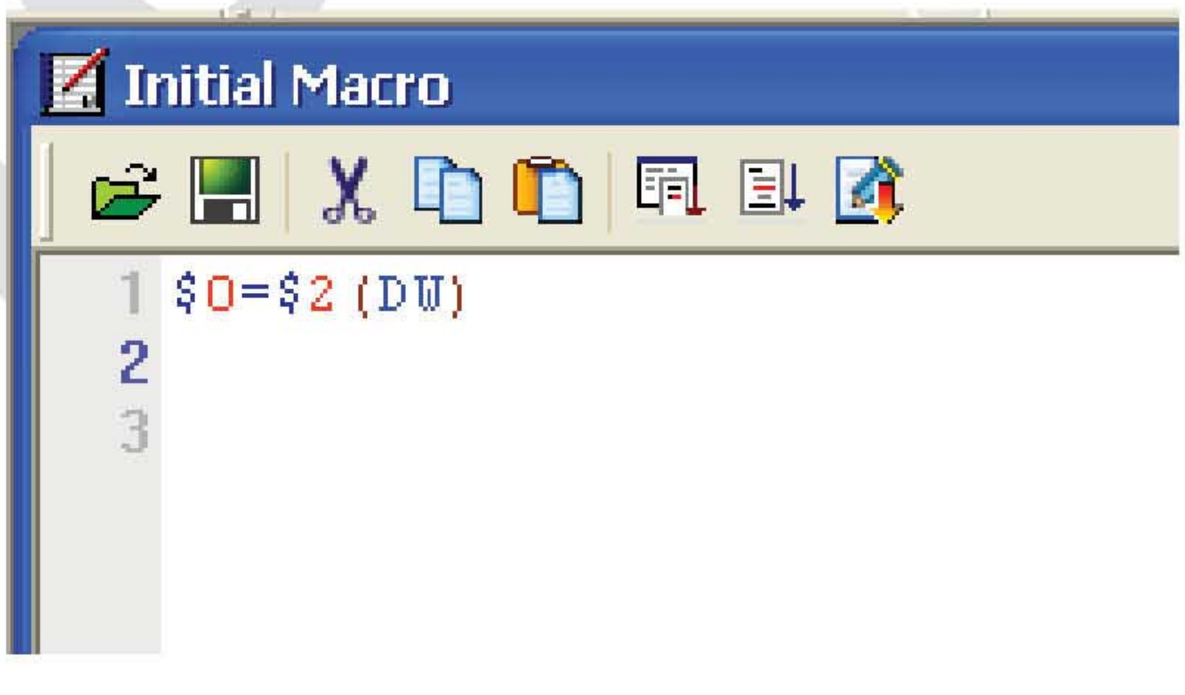


Рис. 3-14-33 Ввод макро команды

После редактирования необходимо провести проверку синтаксиса (**Syntax Check**) и выполнение команды **Update**.

Подробнее, описание функций **Syntax Check** и **Update** приведено в разделе 3.14.2.1.

3.14.2.3 Ввод с клавиатуры

Для удобства редактирования имеется возможность вводить команды с клавиатуры. Система автоматически будет проводить проверку правильности ввода и при ошибке выдаст сообщение. Ввод лишних пробелов ошибками не считается. Дополнительно, появляется перечень макрокоманд при ручном наборе команд, что позволяет пользователю быстро ввести команду (Рис. 3-14-33).

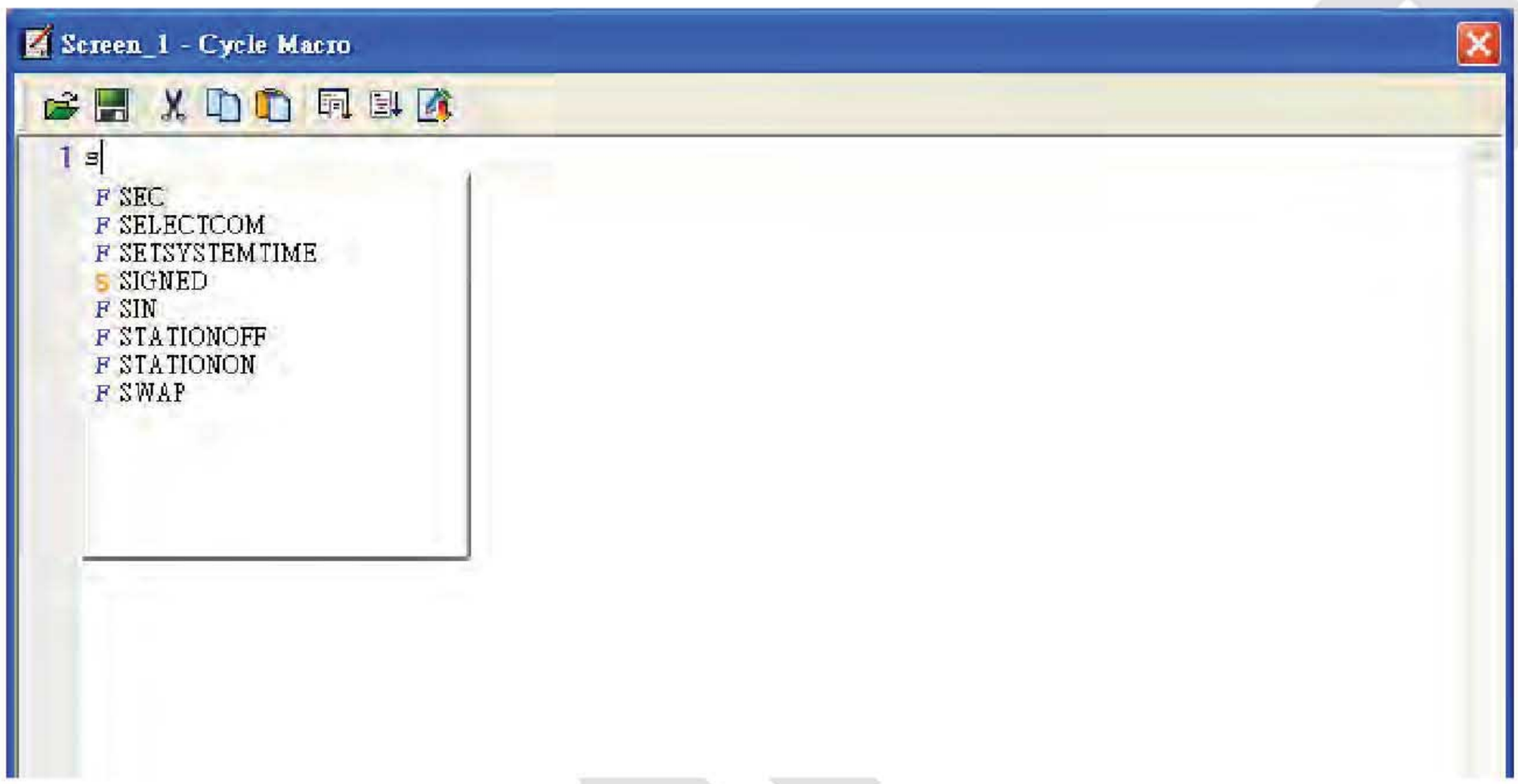


Рис. 3-14-33

После ввода макрокоманды экран выглядит так, как показано на Рис. 3-14-34.

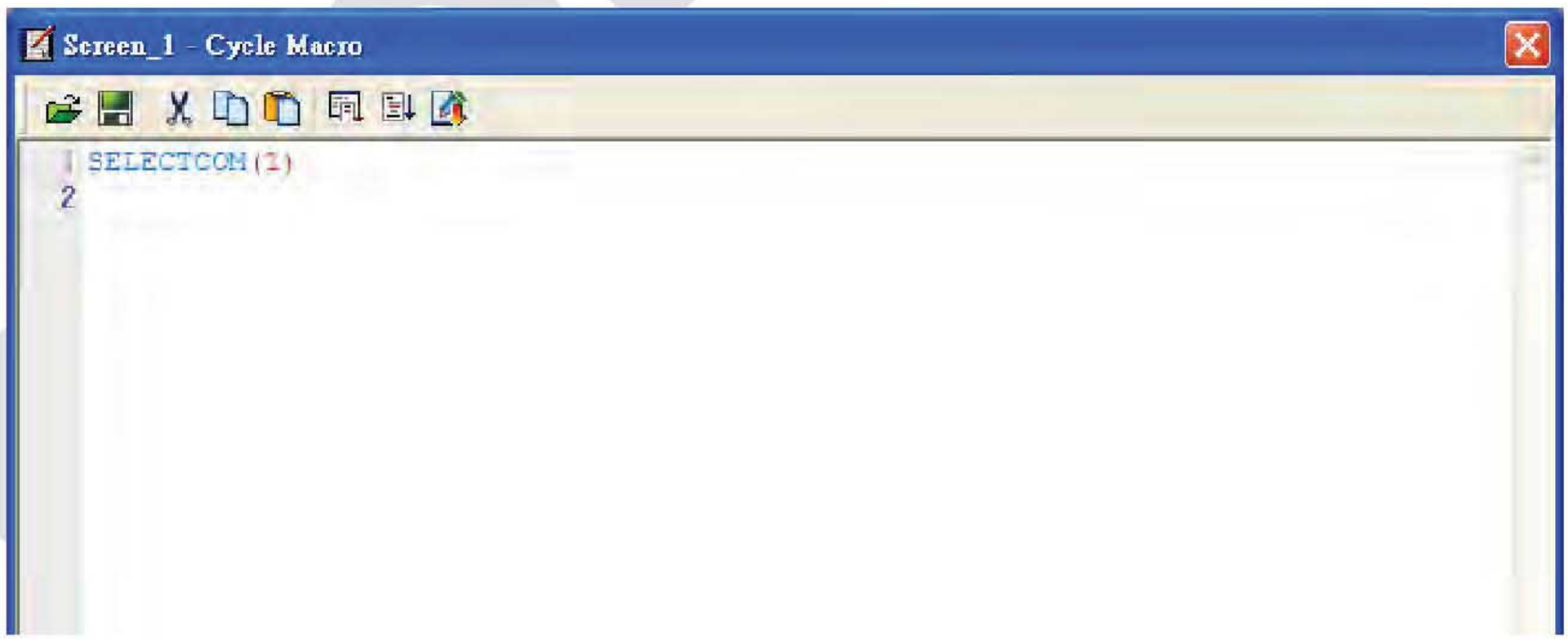


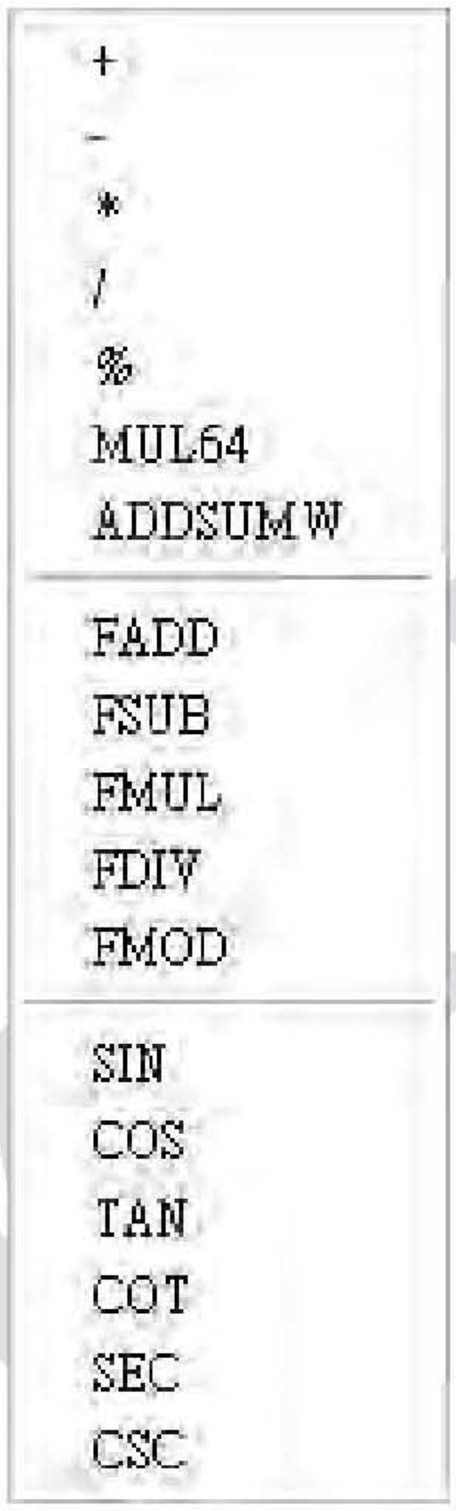
Рис. 3-14-34

После редактирования необходимо провести проверку синтаксиса (**Syntax Check**) и выполнение команды **Update**.

Подробнее, описание функций **Syntax Check** и **Update** приведено в разделе 3.14.2.1.

3.14.3 Макрооперации

3.14.3.1 Арифметические операции



Два типа арифметических операций – с целыми числами и с плавающей запятой. Каждый операнд может размещаться во внутренней памяти или быть константой, а результат - только во внутренней памяти

Для примера ниже приводится таблица и примеры написания

■ **+ (Addition)** Операция сложения

Выражение: Var1 = Var2 + Var3

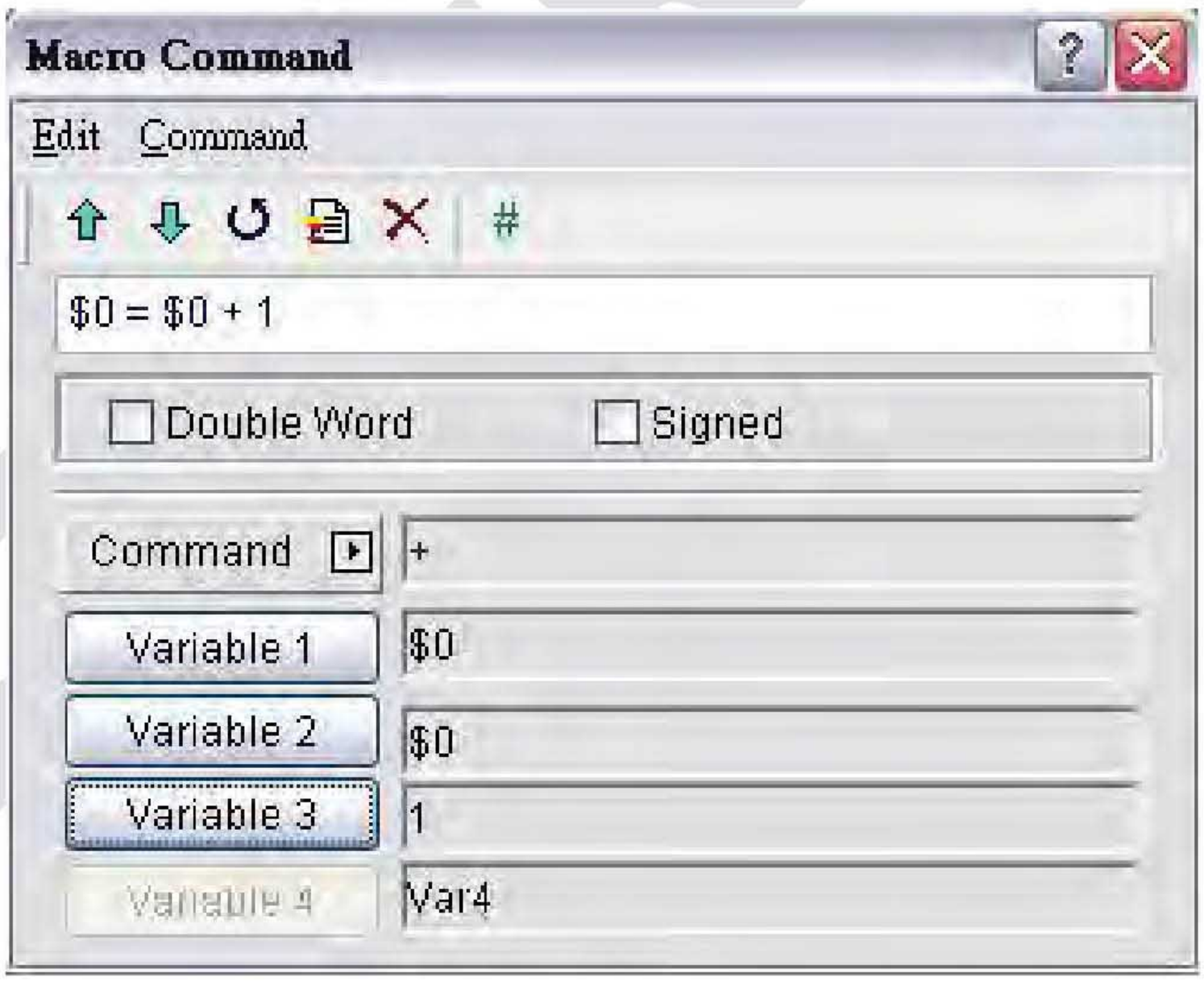
Описание: Произвести сложение Var2 и Var3, сохранить результат в Var1.

Замечания:

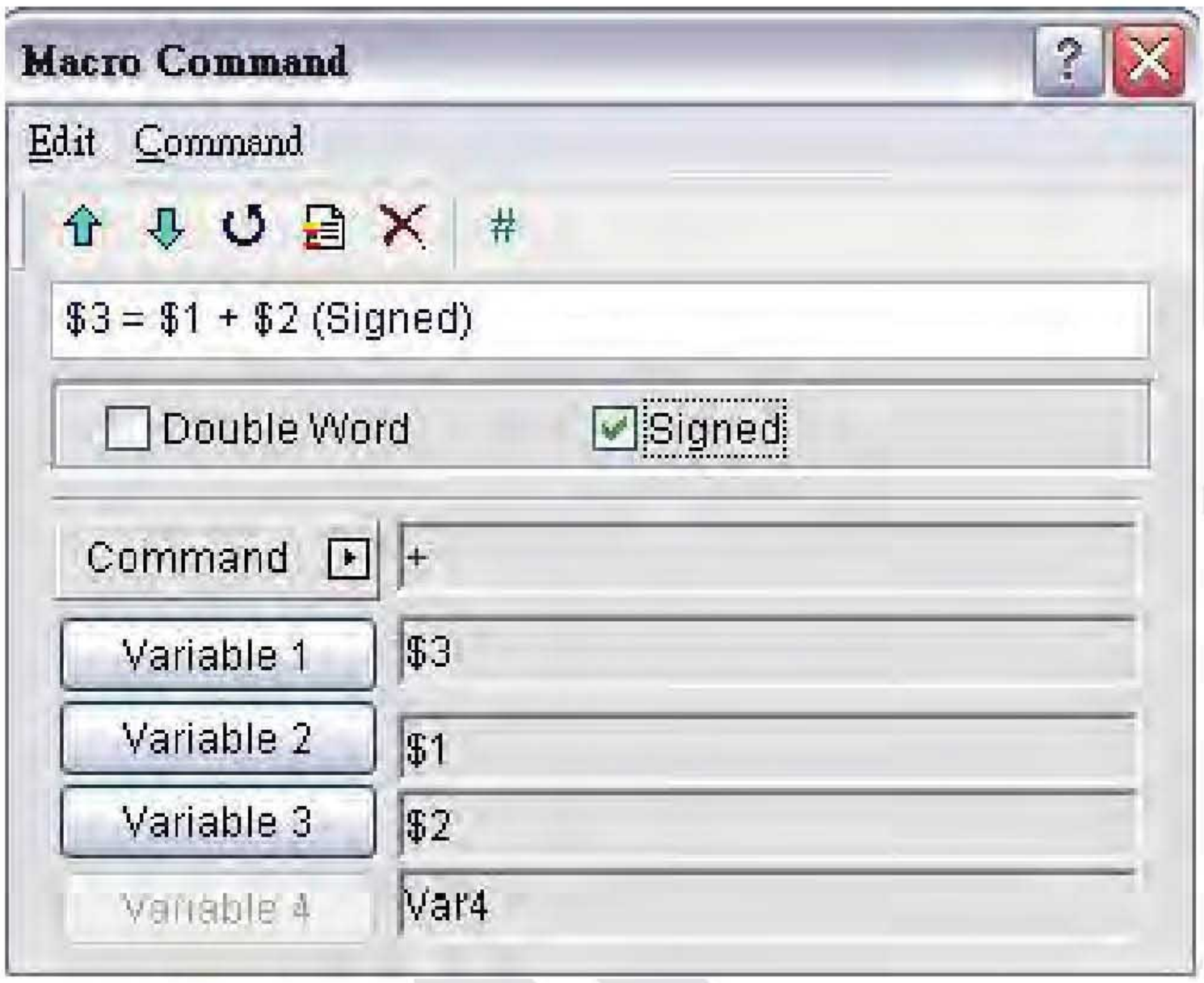
- Вычисленный результат может сохранён как числом со знаком так и числом без знака, в формате WORD или DWORD. При превышении данными заданной длины, та часть, которая превышает длину, будет отброшена.
- Операнды Var1 могут быть только во внутренней памяти, Var2 и Var3 - во внутренней памяти или в виде константы.

Пример

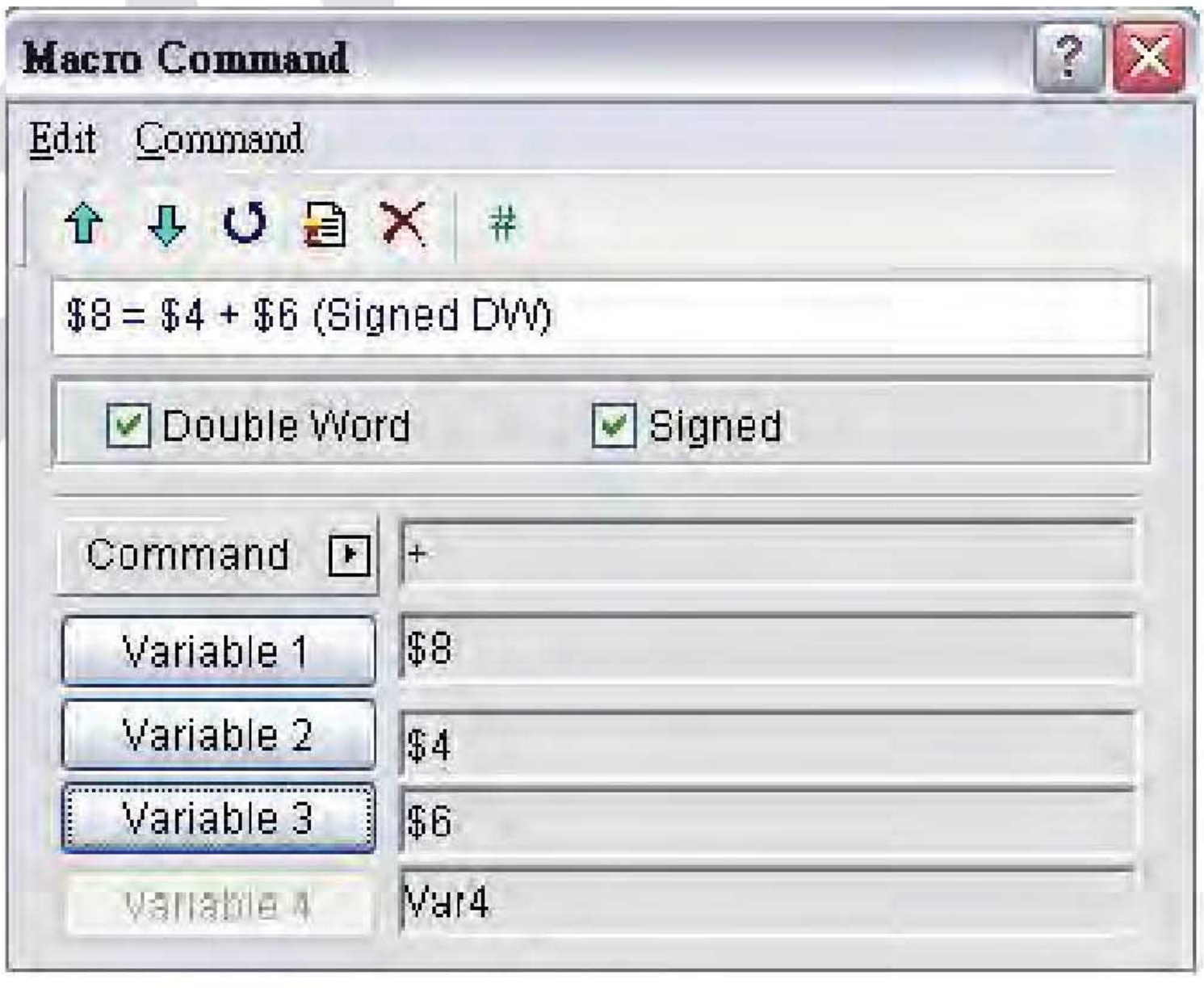
Число 1 складывается с содержимым регистра внутренней памяти $0 и сохраняется в регистре $0. Эта операция проводится с 16-ю разрядными целыми операндами без знака.



Складываются содержимое регистров $1 и $2 и результат сохраняется в регистре $3. (Эта операция проводится с 16-ю разрядными целыми операндами со знаком).



Складывается содержимое регистров $4 и $6 и результат сохраняется в регистре $3. (Эта операция проводится с 32-х разрядными целыми операндами со знаком).



■ **- (Subtraction)** Операция вычитания

Выражение: Var1 = Var2 - Var3

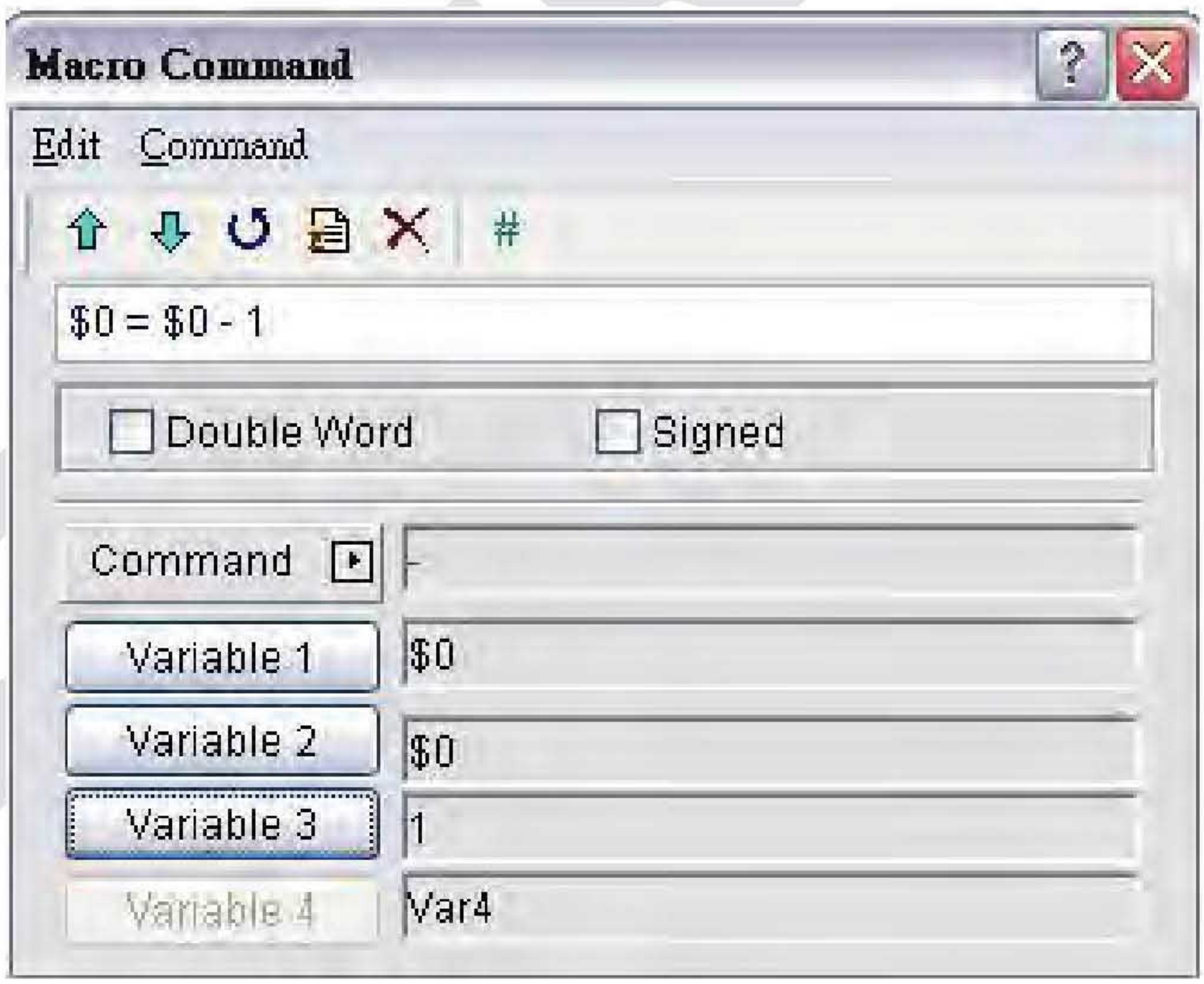
Описание: Произвести вычитание из содержимого Var2 содержимого Var3, а результат сохранить Var1.

Замечания:

- Результат вычислений может быть сохранён в виде числа со знаком или без знака в формате WORD или DWORD. При превышении данными заданной длины, та часть, которая превышает длину, будет отброшена.
- Операнды Var1 могут быть только во внутренней памяти, Var2 и Var3 - во внутренней памяти или в виде константы.

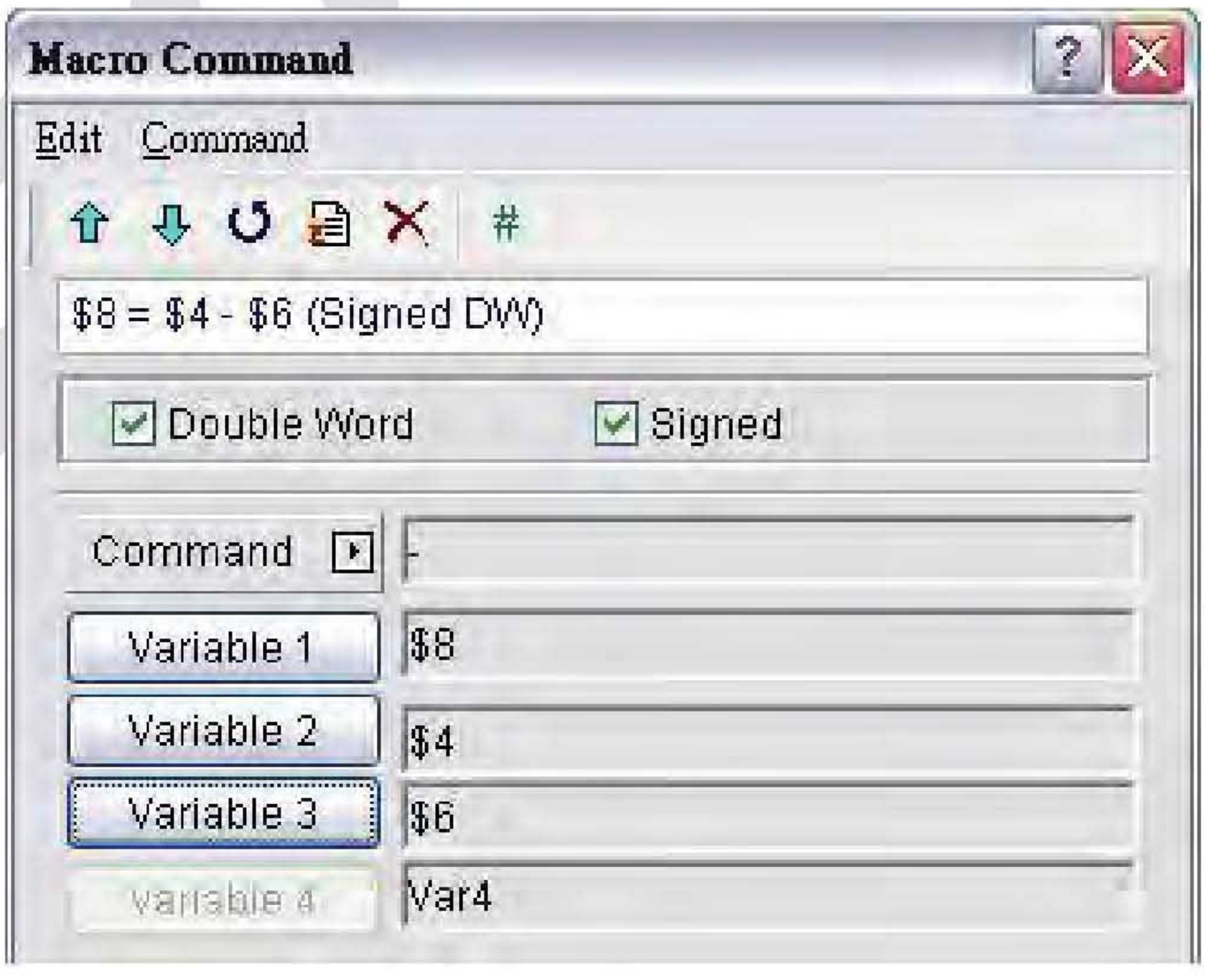
Пример

Число 1 вычитается из содержимого регистра $0 и результат сохраняется в регистре $0 (Эта операция проводится с 16-ю разрядными целыми операндами без знака)



Содержимое регистра $2 вычитается из содержимого регистра $1 и результат сохраняется в регистре $3. (Эта операция проводится с 16-ю разрядными целыми операндами со знаком)

Содержимое регистра $6 вычитается из содержимого регистра $4 и результат сохраняется в регистре $8. (Эта операция проводится с 32-х разрядными целыми операндами со знаком)



■ *** (Multiplication)** Операция умножения

Выражение: Var1 = Var2 * Var3

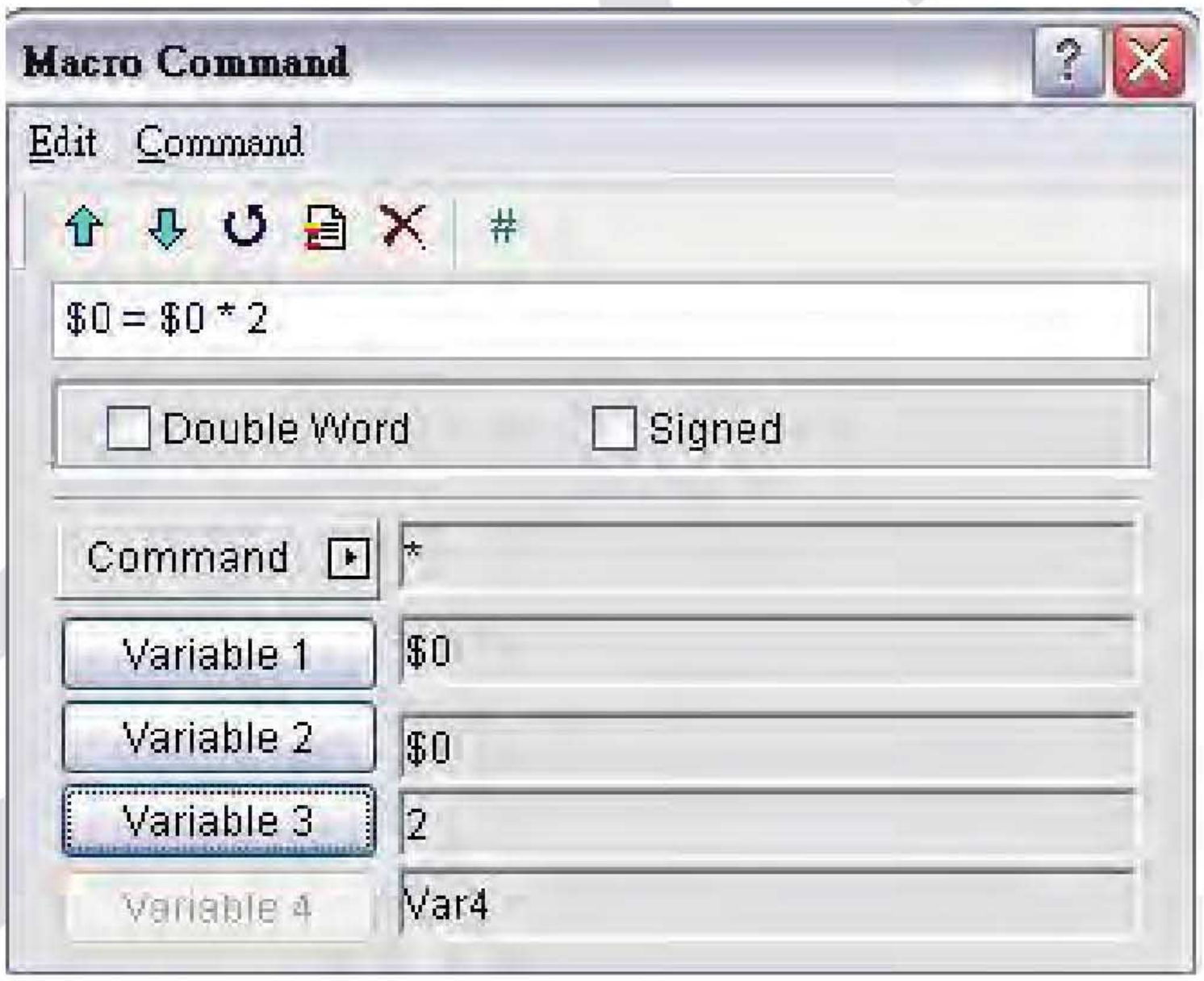
Описание: Перемножить Var2 и Var3, результат умножения сохранить в Var1.

Замечания:

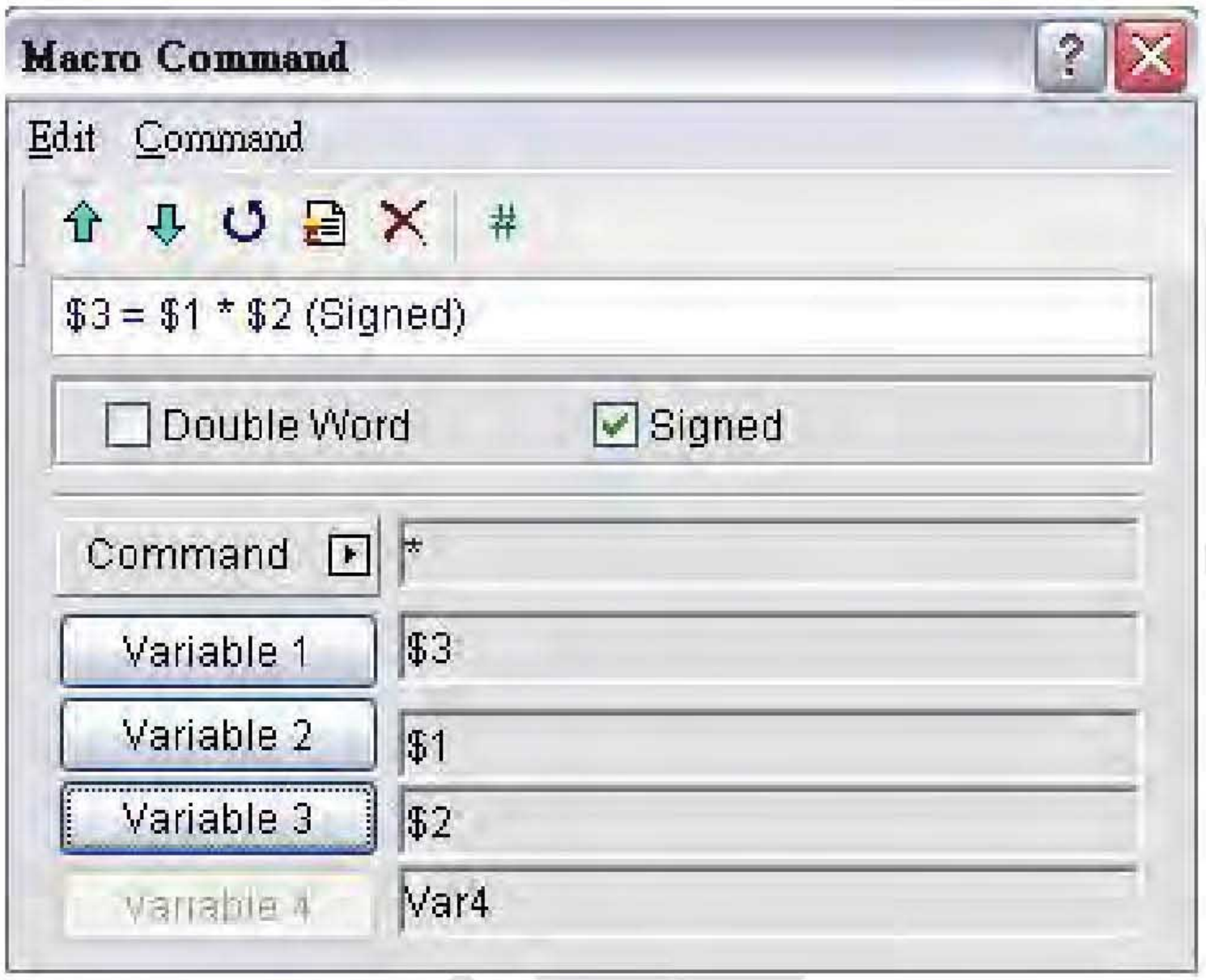
- Результат вычислений может быть целым числом со знаком или без знака в формате WORD или DWORD. При превышении данными заданной длины, та часть, которая превышает длину, будет отброшена.
- Операнды Var1 могут быть только во внутренней памяти, Var2 и Var3 - во внутренней памяти или в виде константы.

Пример

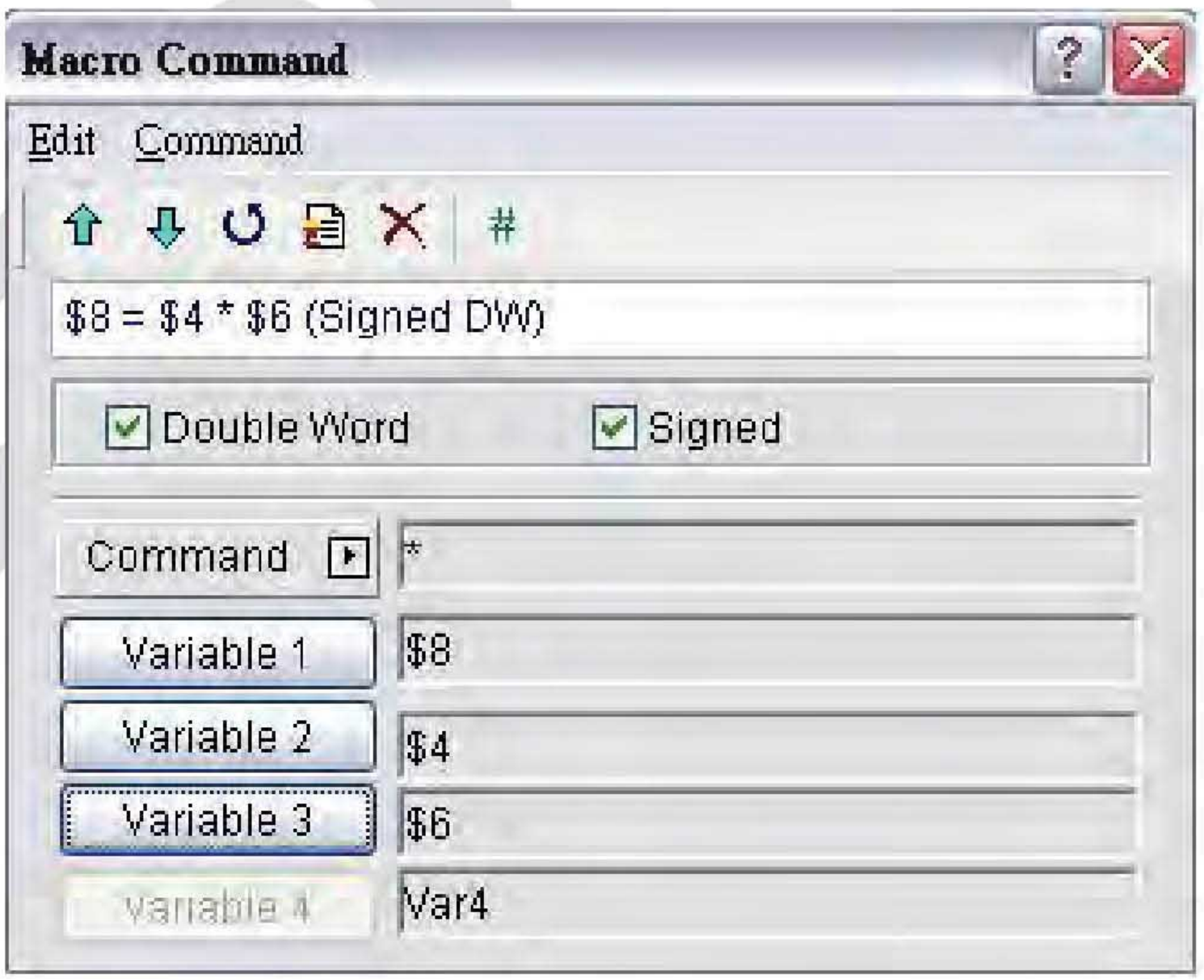
Содержимое регистра $0 умножить на 2 и сохранить результат в $0. (Эта операция проводится с 16-ю разрядными целыми операндами без знака)



Содержимое регистров $1 и $2 перемножить и сохранить результат в $3. (Эта операция проводится с 16-ю разрядными целыми операндами со знаком.)



Содержимое регистров $4 и $6 перемножить и сохранить результат в $8. (Эта операция проводится с 32-х разрядными целыми операндами со знаком)



■ **/ (Division)** Операция деления

Выражение: Var1 = Var2 / Var3

Описание: Произвести деление операнда Var2 на значение Var3, и сохранить результат вычислений в Var1.

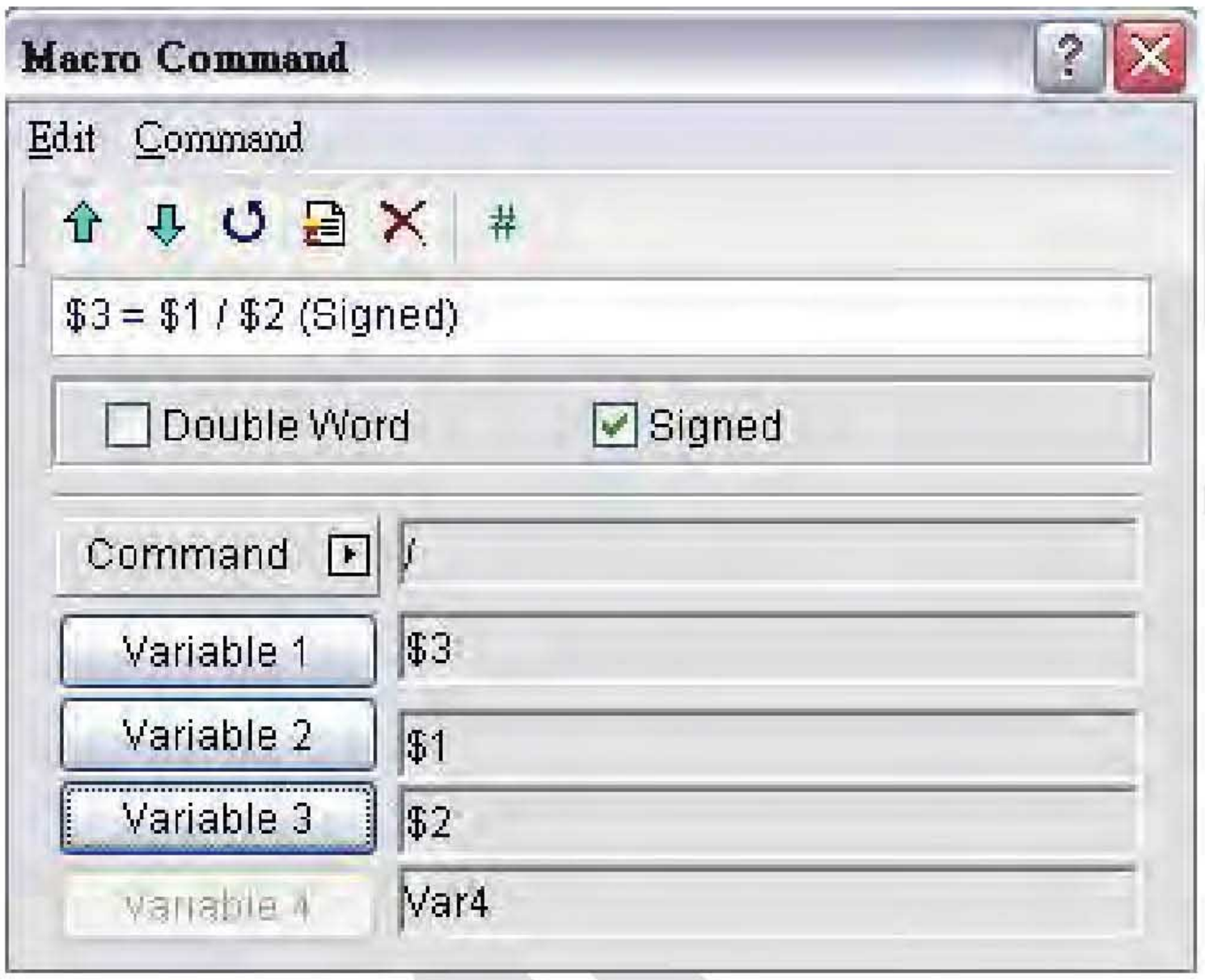
Замечания:

- Результат вычислений может быть целым числом со знаком или без знака в формате WORD или DWORD. При превышении данными заданной длины, та часть, которая превышает длину будет отброшена.
- Операнды Var1 могут быть только во внутренней памяти, Var2 и Var3 - во внутренней памяти или в виде константы.
- Значение операнда Var3 не может быть равно 0

Пример

Содержимое регистра $0 разделить на 2, и сохранить результат вычислений в регистре $0. (Эта операция проводится с 16-ю разрядными целыми операндами без знака)

Содержимое регистра $1 разделить на содержимое регистра $2 и сохранить результат вычислений в регистре $3. (Эта операция проводится с 16-ю разрядными целыми операндами со знаком)



Содержимое регистра $4 разделить на содержимое $6 и результат вычислений сохранить в регистре $8. (Эта операция проводится с 32-х разрядными целыми операндами со знаком)

■ **% (Get Remainder)** Вычисление остатка

Выражение: Var1 = Var2 % Var3

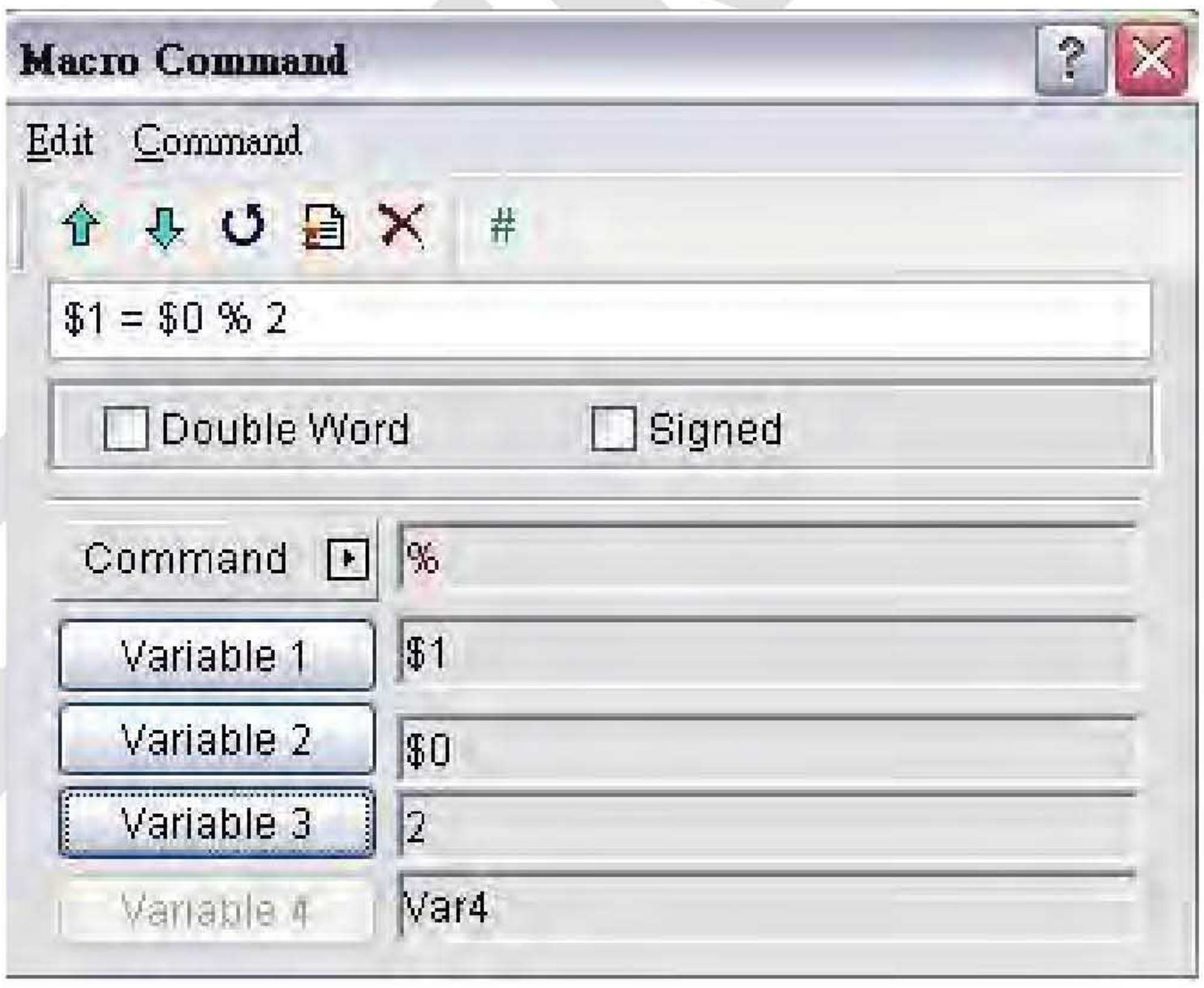
Описание: Разделить содержимое Var2 на Var3, и сохранить остаток в Var1.

Замечания:

- Результат вычислений может быть целым числом со знаком или без знака в формате WORD или DWORD. При превышении данными заданной длины, та часть, которая превышает длину, будет отброшена.
- Операнды Var1 могут быть только во внутренней памяти, Var2 и Var3 - во внутренней памяти или в виде константы.
- Значение операнда Var3 не может быть равно 0

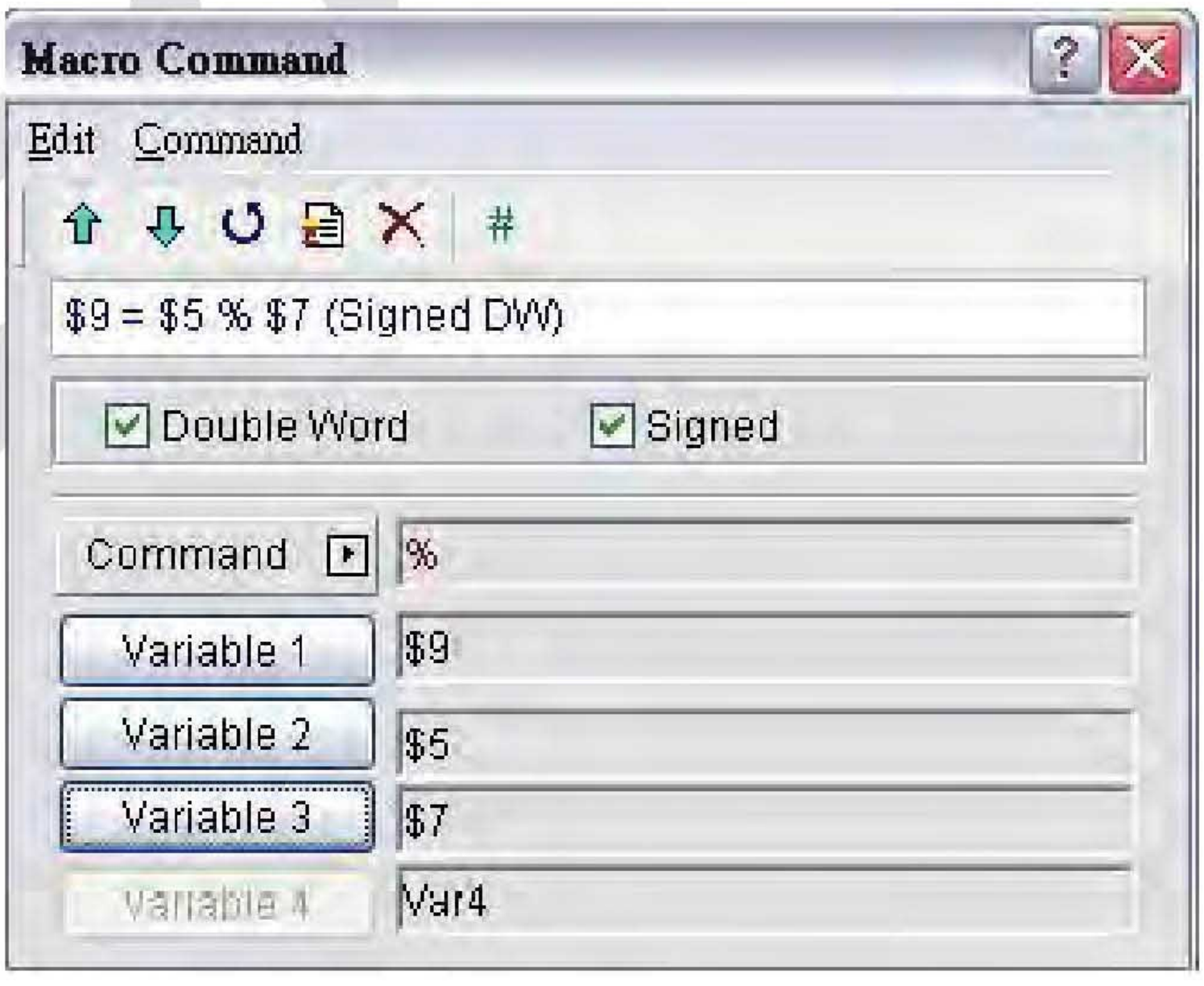
Пример

Содержимое регистра $0 разделить на 2 и значение остатка сохранить в регистре $0. (Эта операция проводится с 16-ю разрядными целыми операндами без знака)



Содержимое регистра $2 разделить на содержимое регистра $3 и значение остатка сохранить в регистре $4. (Эта операция проводится с 16-ю разрядными целыми операндами со знаком)

Содержимое регистра $5 разделить на содержимое регистра $7 и значение остатка сохранить в регистре $9. (Эта операция проводится с 32-х разрядными целыми операндами со знаком)



■ **MUL64 (64-bit Multiplication)** Операция 64-битного умножения

Выражение: Var1 = MUL64 (Var2, Var3)

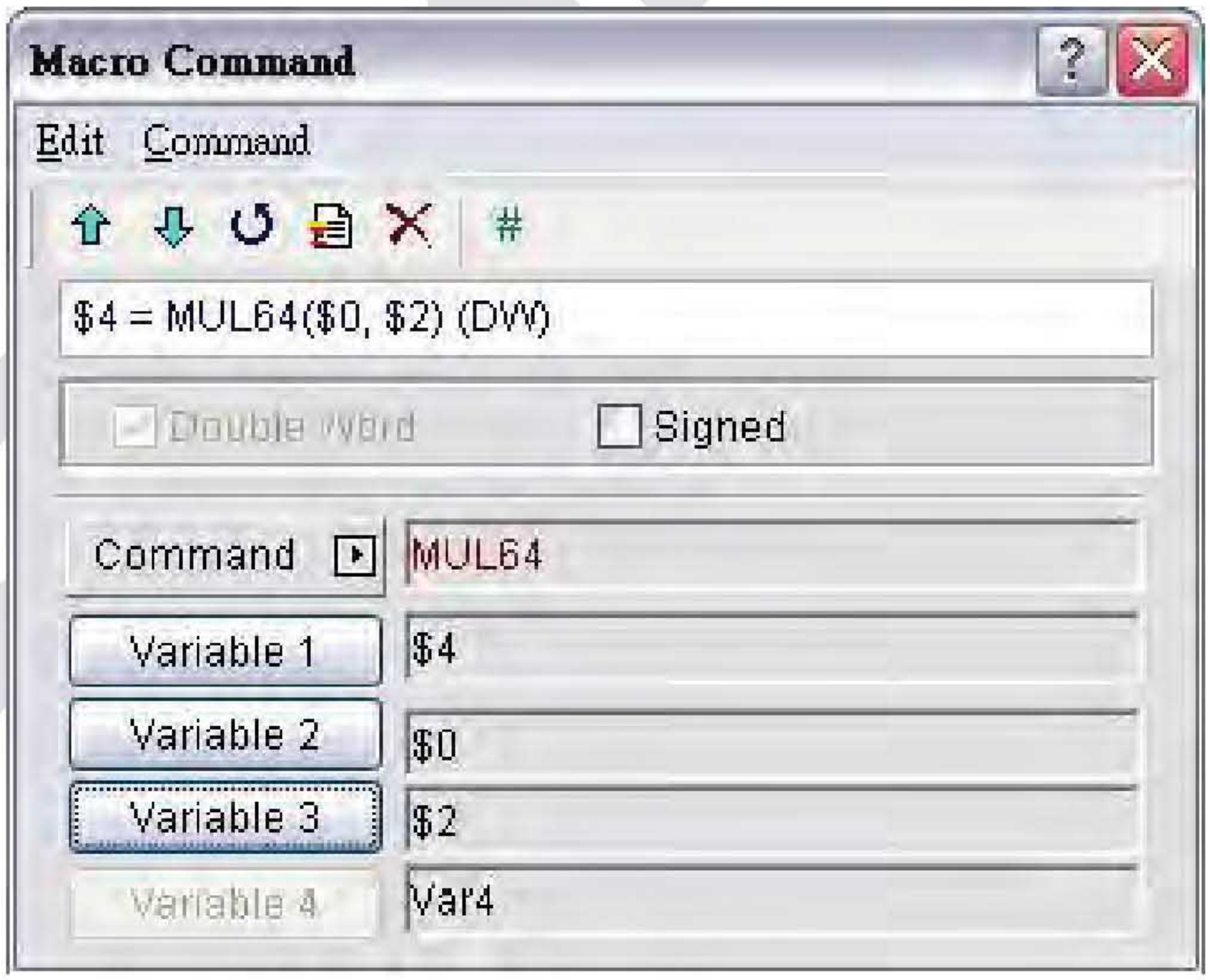
Описание: Содержимое регистра Var2 умножить на содержимое Var3 и сохранить результаты вычислений в Var3.

Замечания:

- Это 32-битная операция.
- Результат вычислений - число в формате DWORD (со знаком или без знака)
- Результат Var1 занимает 4 слова, операнды Var2 и Var3 занимают 2 слова.
- Var1 может храниться только во внутренней памяти, Var2 и Var3 - во внутренней памяти или в виде константы.

Пример

Содержимое регистра $0 умножить на содержимое регистра $2 и сохранить результаты в регистре $4. (Эта операция проводится с числами в 32-х битном формате без знака)



Содержимое регстра $8 умножить на содержимое $10 и сохранить результаты в регистре $12. (Эта операция проводится с числами в 32-х битном формате со знаком).

- **ADDSUMW (Repeated Addition)** Многократное сложение

Выражение: Var1 = ADDSUMW (Var2, Var3)

Описание: Сложить с содержимым Var3 содержимое регистров начиная с Var2 и сохранить результат Var1

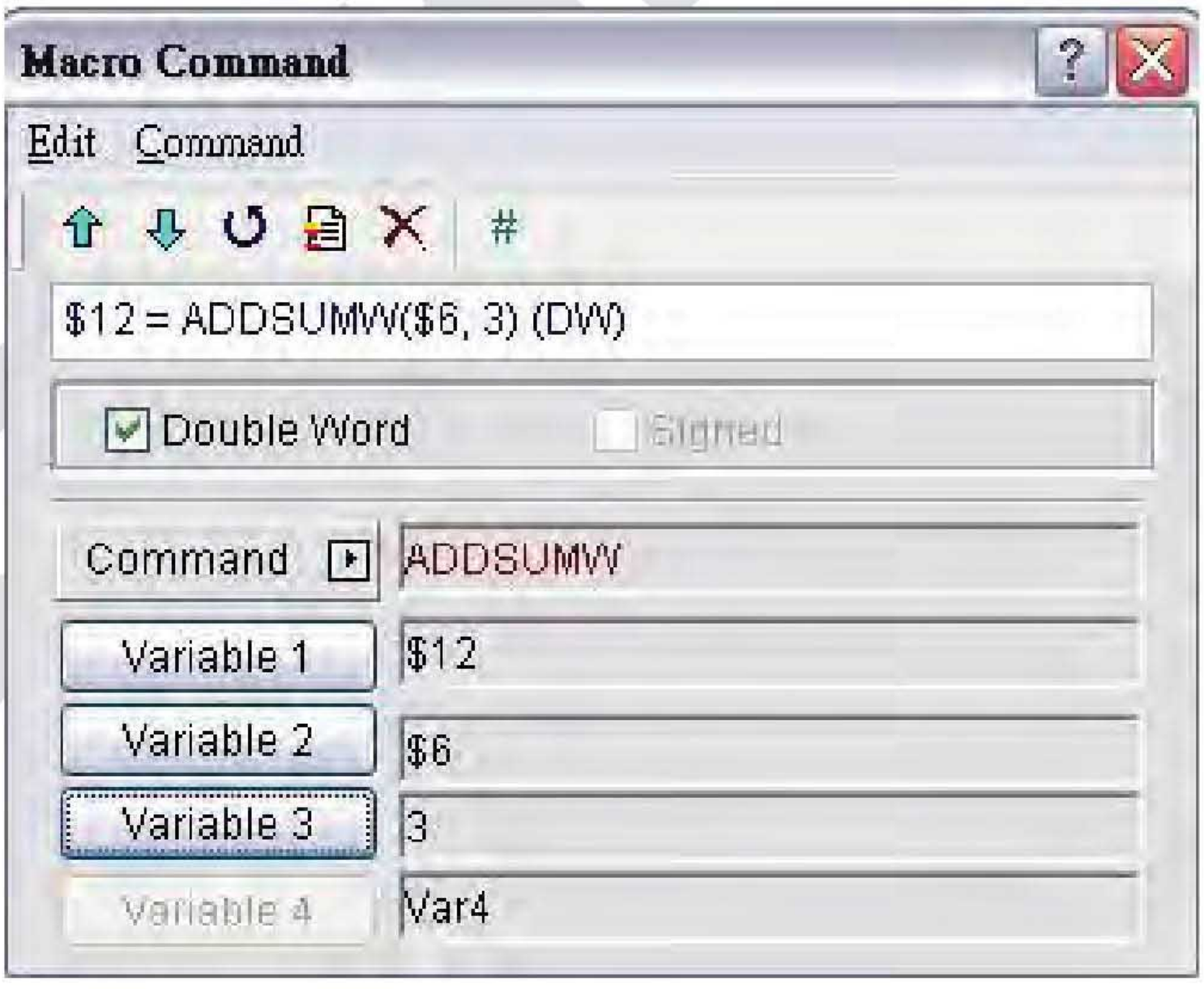
Замечания:

- Эта операция производится с целыми числами без знака. Вычисленный результат сохраняется в виде числа без знака в формате WORD или DWORD. При превышении данными заданной длины, та часть, которая превышает длину, будет отброшена.
- Var1 и Var2 могут храниться только во внутренней памяти и Var3 - во внутренней памяти или в виде константы.

Пример

Сложить содержимое регистров внутренней памяти $0, $1, $2, $3 $4 и результат сохранить в регистре $5. (Эта операция проводится с 16-ю битными числами без знака)

Сложить содержимое регистров в нутренней памяти $6, $8 и $10, и результат вычисления сохранить в регистре $12 (Эта операция проводится с 32-х битными числами без знака.)



■ **FADD (Floating Addition)** Сложение чисел с плавающей запятой

Выражение: Var1 = FADD (Var2, Var3)

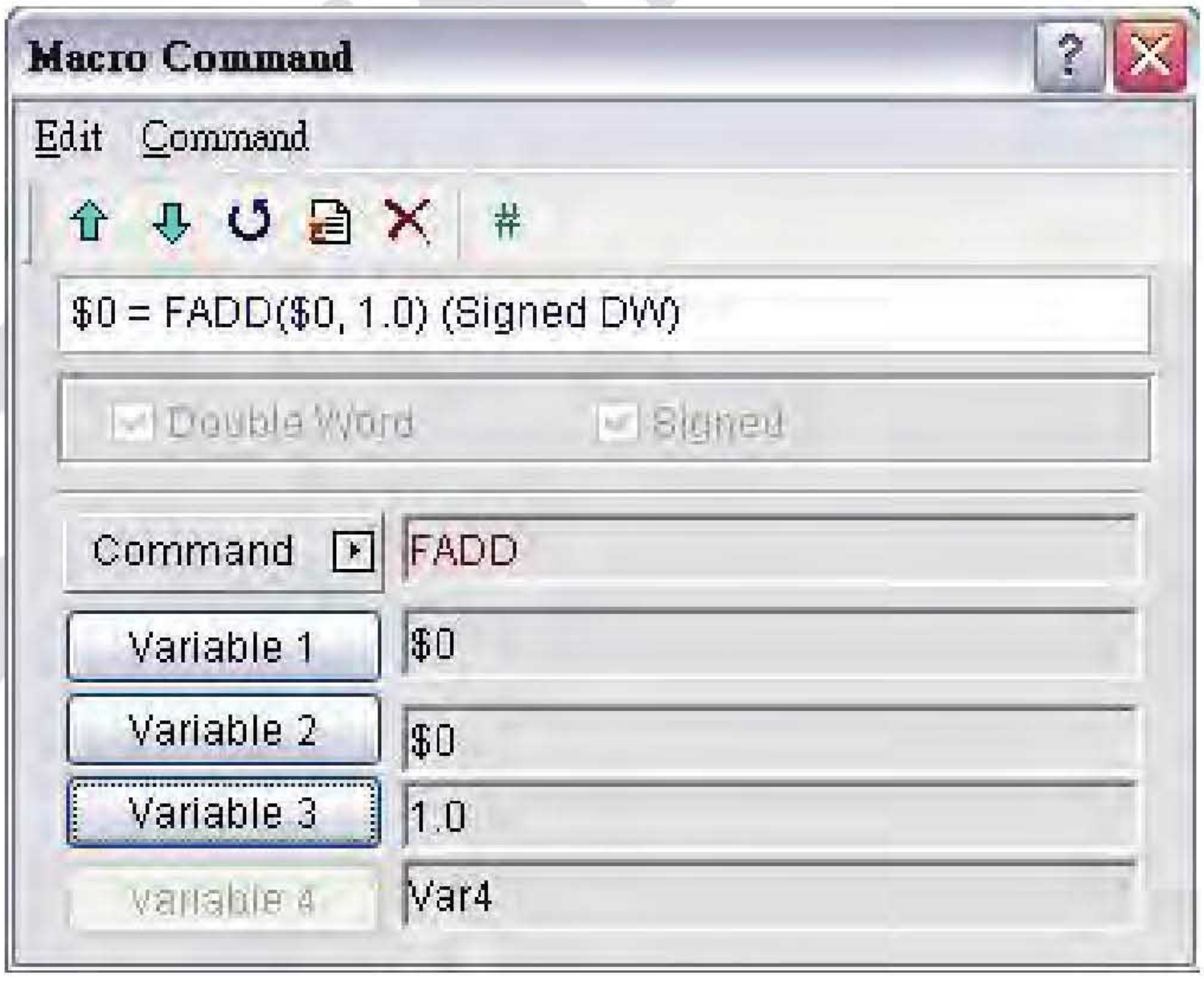
Описание: Сложить содержимое Var2 и Var3, результат вычислений сохранить в Var1.

Замечания:

- Эта операция производится с 32-битными числами со знаком.
- Вычисленный результат сохраняется в виде числа без знака в формате DWORD. При превышении данными заданной длины, та часть, которая превышает длину будет отброшена.
- Var1 могут храниться только во внутренней памяти, Var2 и Var3 - во внутренней памяти или в виде константы.

Пример

Сложить 1.0 с содержимым регистра $0 и результат вычислений сохранить в регистре $0. (Эта операция производится с 32-х битными числами с плавающей запятой со знаком.)



Сложить содержимое регистра $4 с содержимым регистра $2 и результат вычислений сохранить в регистре $6. (Эта операция производится с 32-х битными числами с плавающей запятой со знаком)

■ **FSUB (Floating Subtraction)** Вычитание чисел с плавающей запятой

Выражение: Var1 = FSUB (Var2, Var3)

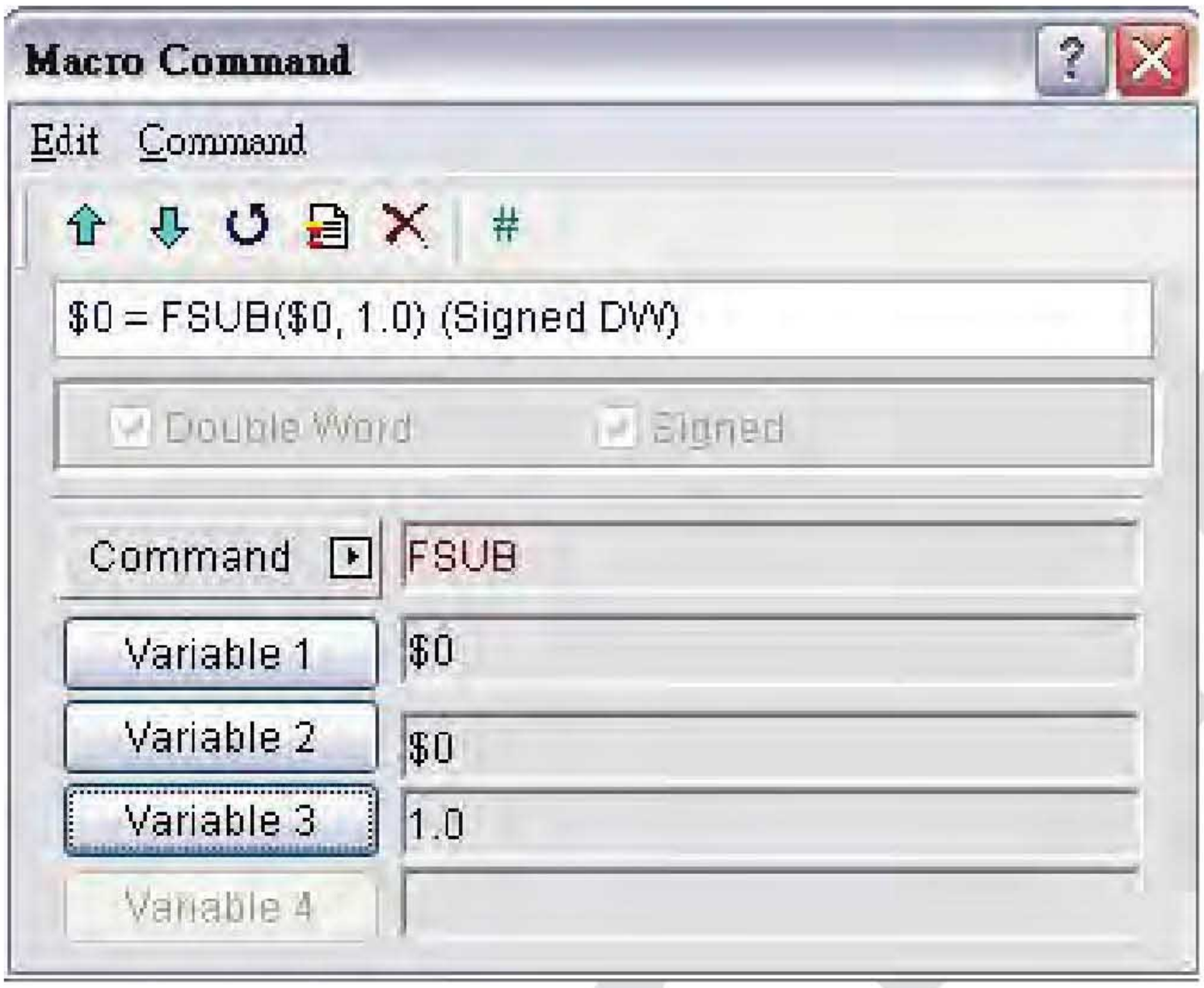
Описание: Вычесть содержимое регистра Var2 из содержимого регистра Var3, и сохранить результат вычислений в регистре Var1.

Замечания:

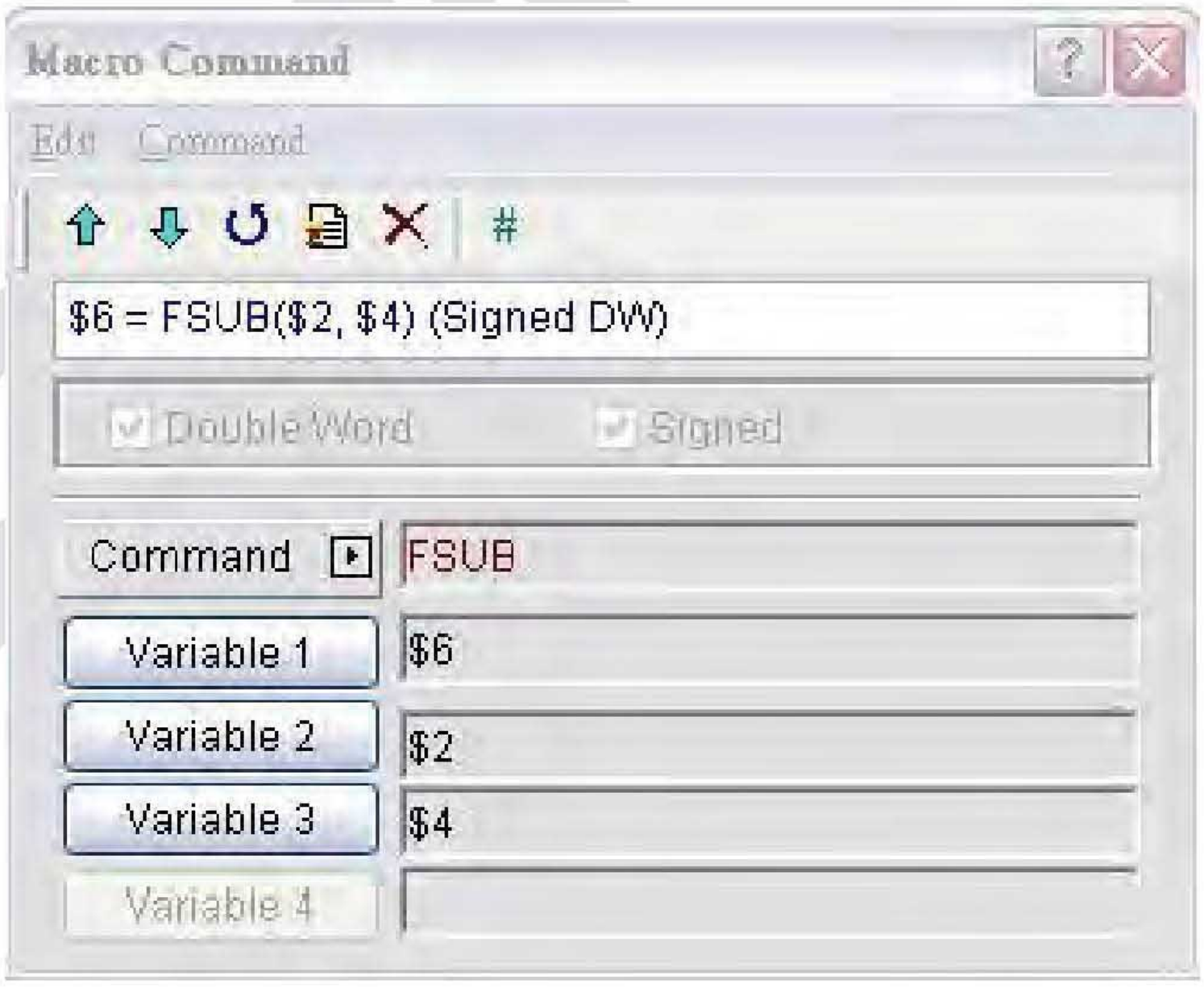
- Эта операция производится с 32-х битными числами с плавающей запятой со знаком.
- Вычисленный результат сохраняется в виде числа со знаком в формате DWORD. При превышении данными заданной длины, та часть, которая превышает длину, будет отброшена.
- Var1 могут храниться только во внутренней памяти, Var2 и Var3 - во внутренней памяти или в виде константы.

Пример

Вычесть число 1.0 из содержимого регистра $0 и сохранить результат вычислений в регистре $0 (Эта операция производится с 32-х битными числами с плавающей запятой со знаком).



Вычесть число $2 из содержимого регистра $4 и сохранить результат вычислений в регистре $6 . (Эта операция производится с 32-х битными числами с плавающей запятой со знаком)



■ **FMUL (Floating Multiplication)** Умножение чисел с плавающей запятой

Выражение: Var1 = FMUL (Var2, Var3)

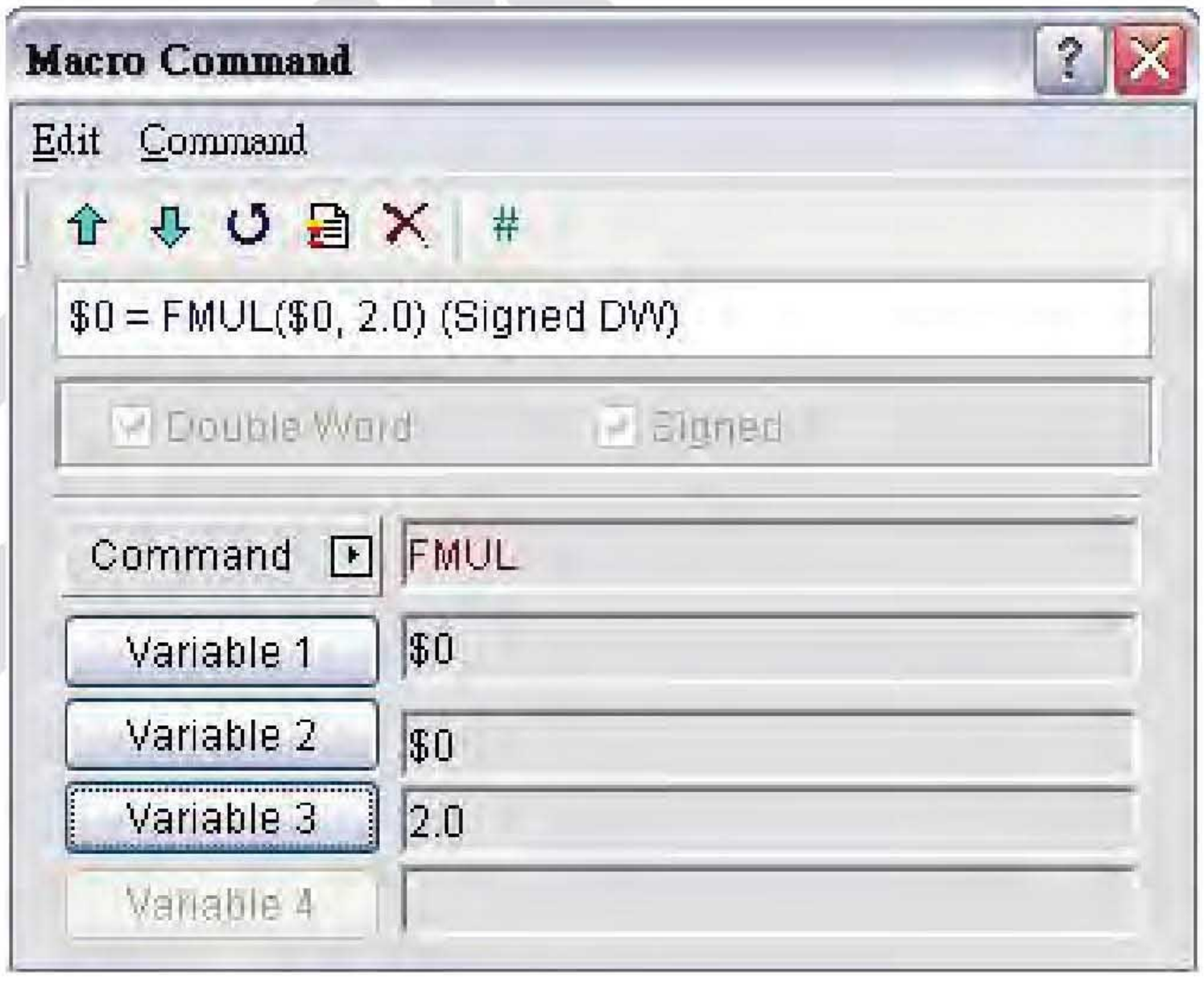
Описание: Умножить содержимое регистра Var2 и содержимое регистра Var3, результат вычислений сохранить в регистре Var3.

Замечания:

- Эта операция производится с 32-х битными числами с плавающей запятой со знаком.
- Результат вычислений сохраняется в виде числа с плавающей запятой в формате DWORD. При превышении данными заданной длины, та часть, которая превышает длину, будет отброшена.
- Var1 могут храниться только во внутренней памяти, Var2 и Var3 - во внутренней памяти или в виде константы.

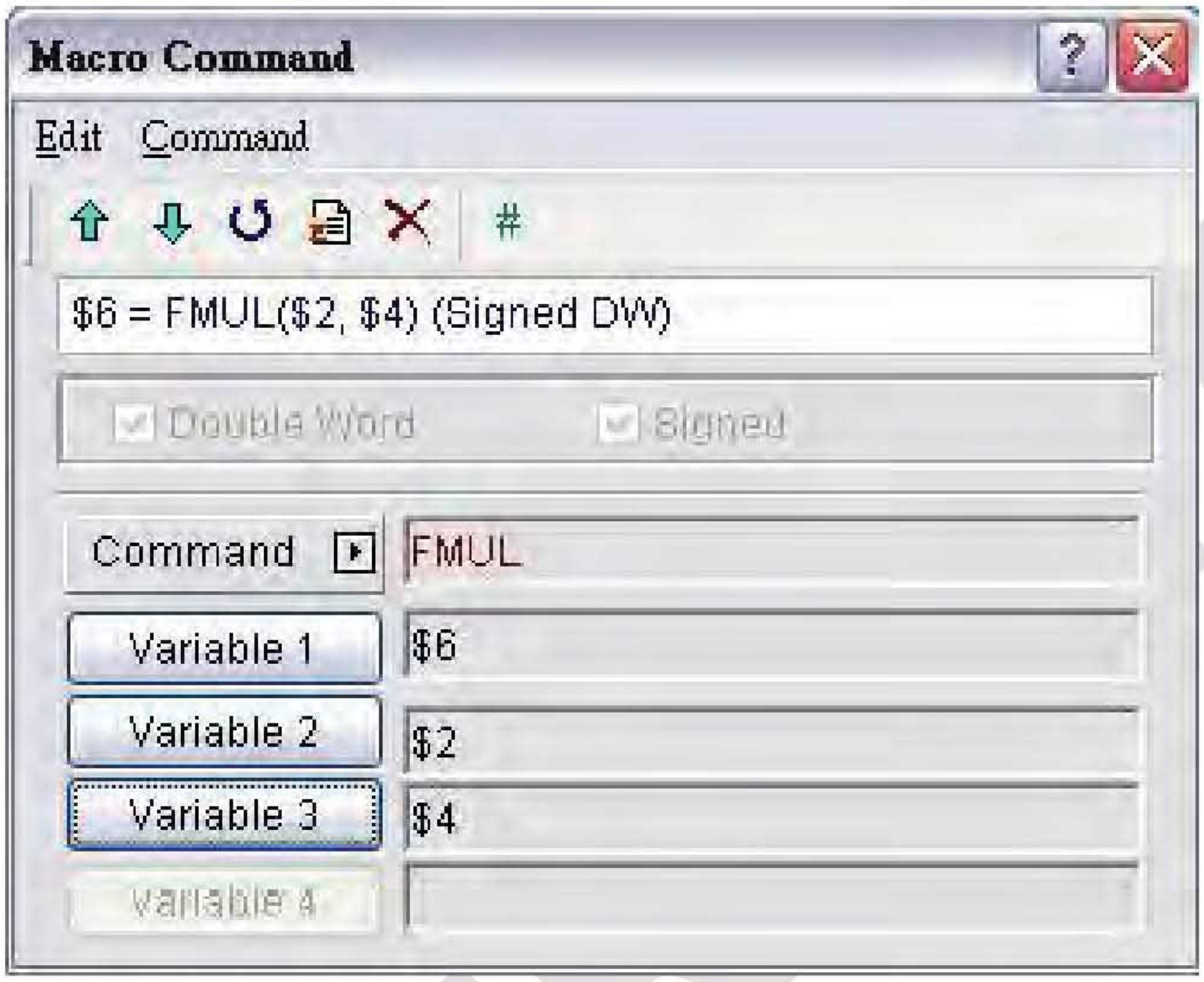
Пример

Содержимое регистра $0 умножить на 2.0 результат вычислений сохранить в $0. (Эта операция производится с 32-х битными числами с плавающей запятой со знаком.)



Содержимое регистра $2 умножить на содержимое регистра $4, результат вычислений сохранить в регистре $6. Эта операция производится с

32-х битными числами с плавающей запятой со знаком (Эта операция производится с 32-х битными числами с плавающей запятой со знаком)



- **FDIV (Floating Division)** Деление чисел с плавающей запятой

  Выражение: Var1 = FDIV (Var2, Var3)

  Описание: Разделить содержимое регистра Var2 на содержимое регистра Var3, результат вычислений сохранить в регистре Var3.

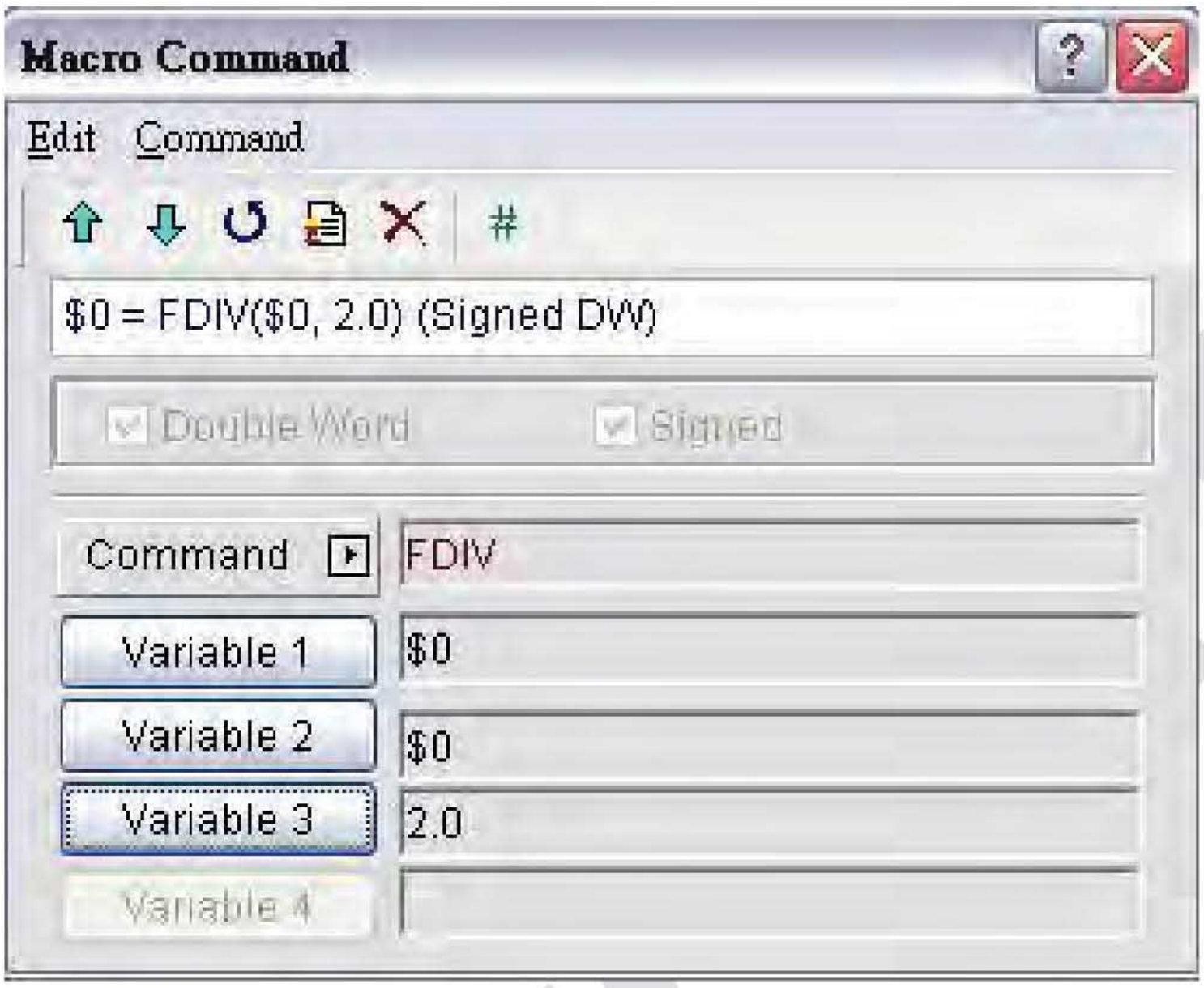
  Замечания:

  - Эта операция производится с 32--х битными числами с плавающей запятой со знаком.
  - Результат вычислений сохраняется в виде числа с плавающей запятой в формате DWORD. При превышении данными заданной длины, та часть, которая превышает длину, будет отброшена.
  - Var1 могут храниться только во внутренней памяти, Var2 и Var3 - во внутренней памяти или бать константой
  - Содержимое регистра Var3 не должно быть равно нулю

  Пример

  Содержимое регистра $0 разделить на 2.0 результат вычислений сохранить в $0. (Эта операция производится с 32--х битными числами с плавающей

запятой со знаком.)



Содержимое регистра $2 разделить на содержимое регистра $4, результат вычислений сохранить в регистре $6. (Эта операция производится с 32-х битными числами с плавающей запятой со знаком.)

■ **FMOD (Get Floating Remainder)** Вычисление остатка деления чисел с плавающей запятой

Выражение: Var1 = FMOD (Var2, Var3)

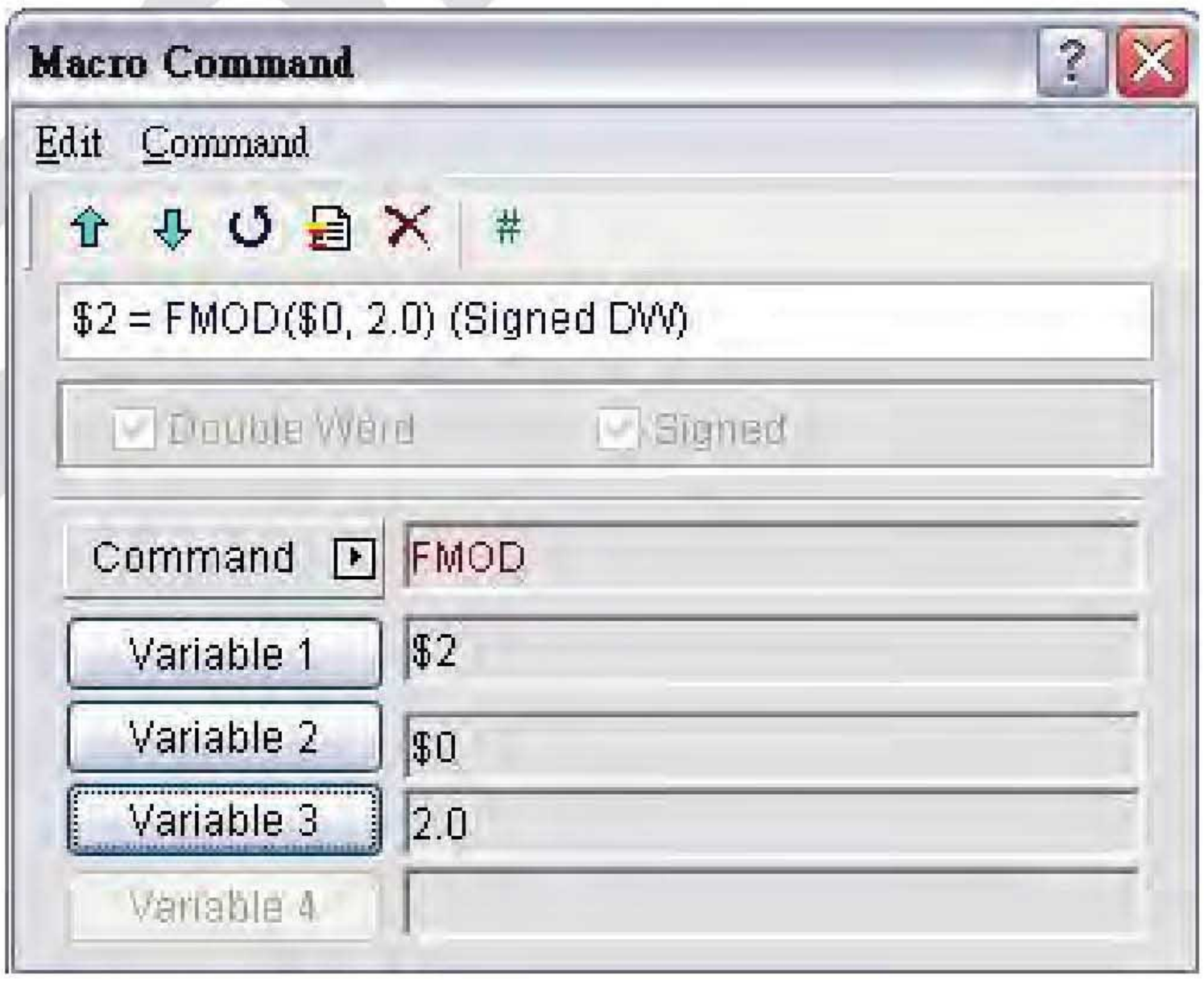
Описание: Разделить содержимое регистра Var2 на содержимое регистра Var3, значение остатка сохранить в регистре Var3.

Замечания:

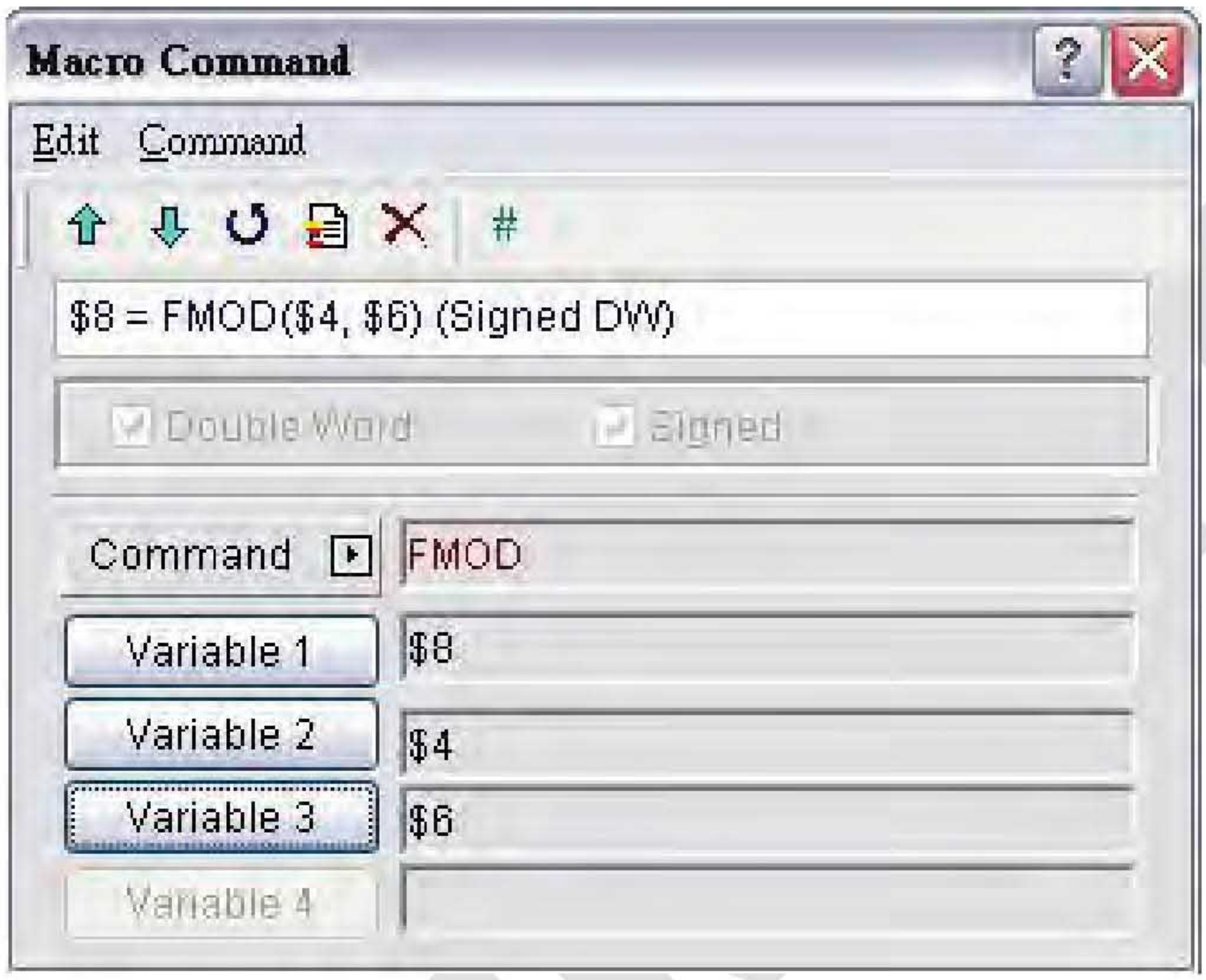
- Эта операция производится с 32-х битными числами с плавающей запятой со знаком.
- Результат вычислений сохраняется в виде числа с плавающей запятой в формате DWORD. При превышении данными заданной длины, та часть, которая превышает длину, будет отброшена.
- Var1 могут храниться только во внутренней памяти, Var2 и Var3 - во внутренней памяти или в виде константы.
- Содержимое регистра Var3 не должно быть равно нулю.

Пример

Содержимое регистра $0 разделить на 2.0 результат вычислений сохранить в $0. (Эта операция производится с 32-х битными числами с плавающей запятой со знаком.)



Содержимое регистра $4 разделить на содержимое регистра $6 результат вычислений сохранить в регистре $8. (Эта операция производится с 32-х битными числами с плавающей запятой со знаком.)



■ **SIN (Sine Function)** Вычисление синуса

Выражение: Var1 = SIN (Var2)

Описание: Вычислить функцию синуса Var2, сохранить результат вычислений Var1.

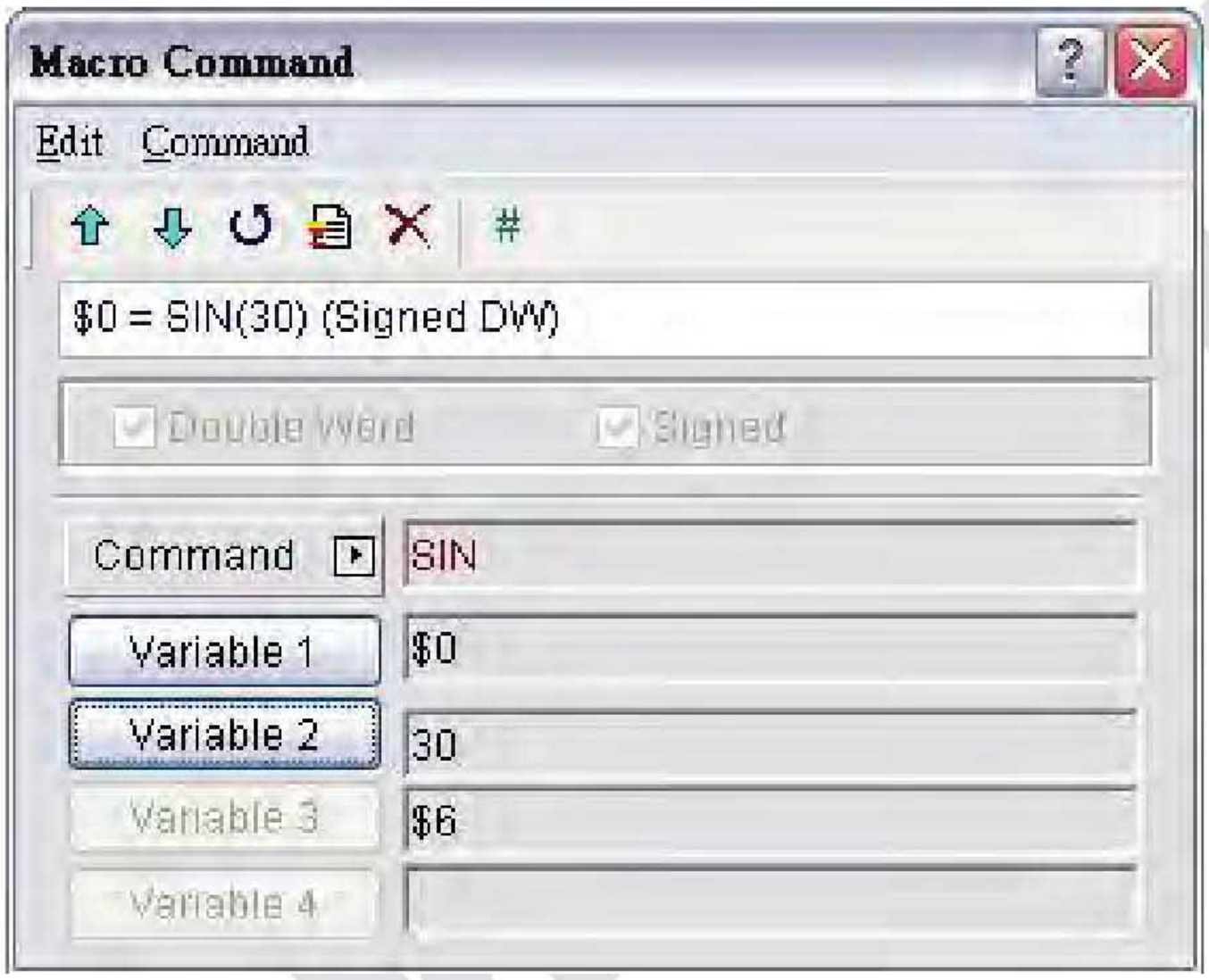
Замечания:

- Значение Var2 задаётся в десятичном целочисленном формате в градусах.
- Эта операция производится с 32-х битными числами с плавающей запятой со знаком.
- Результат вычислений сохраняется в виде числа с плавающей запятой в формате DWORD. При превышении данными заданной длины, та часть, которая превышает длину, будет отброшена.
- Var1 могут храниться только во внутренней памяти, Var2 - во внутренней памяти или в виде константы.
- Отображаемый формат переменных должен быть всегда с плавающей запятой.

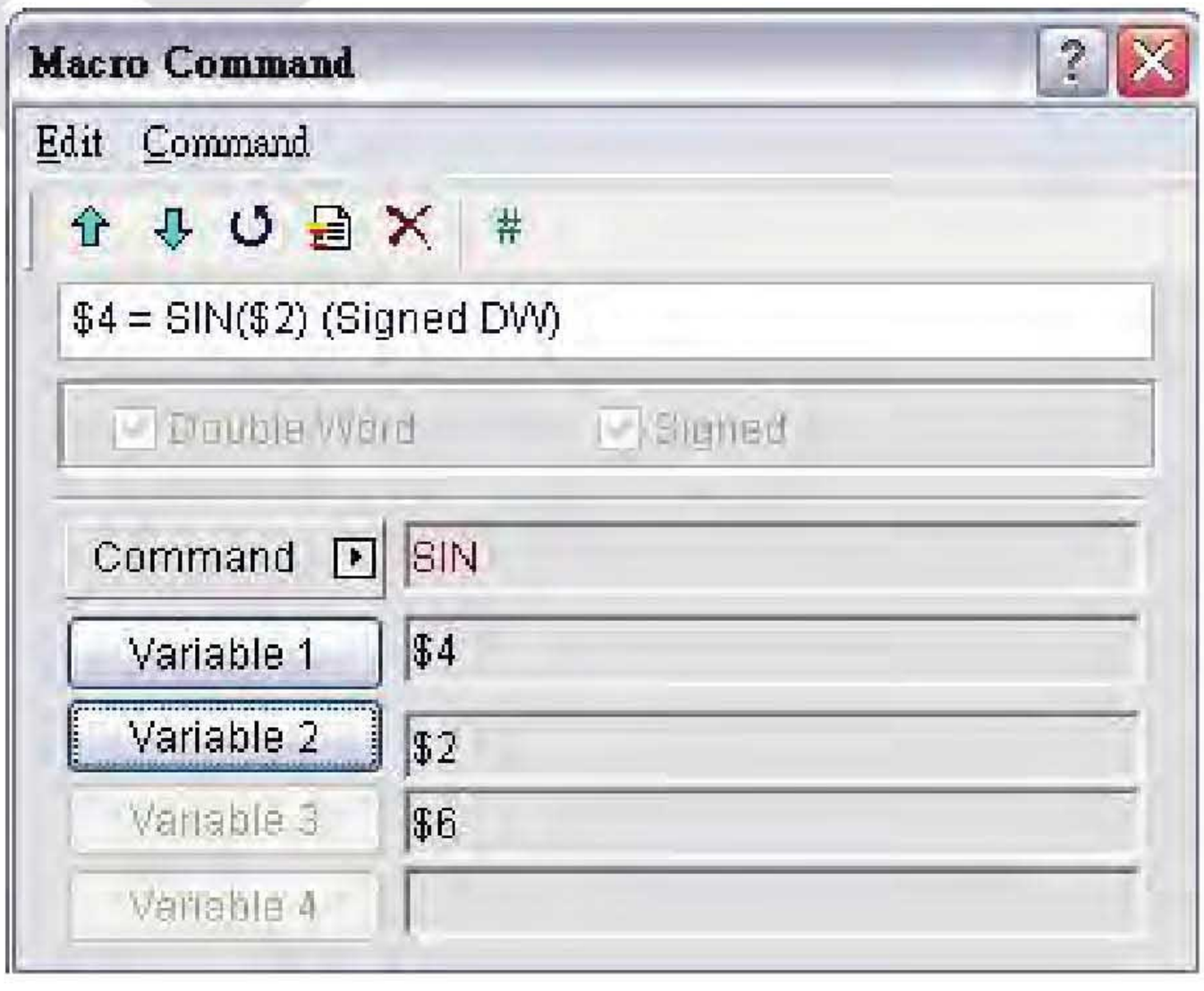
- Входная переменная задаётся в десятичном целочисленном формате со знаком.

Пример

Вычислить SIN30° и сохранить результат в $0. (Эта операция производится с 32-х битными числами со знаком.)



Вычислить синус содержимого регистра $2 и сохранить результат в регистре $4. (Эта операция производится с 32-битными числами с плавающей запятой со знаком.)



■ **COS (Cosine Function)** Вычисление косинуса

Выражение: Var1 = COS(Var2)

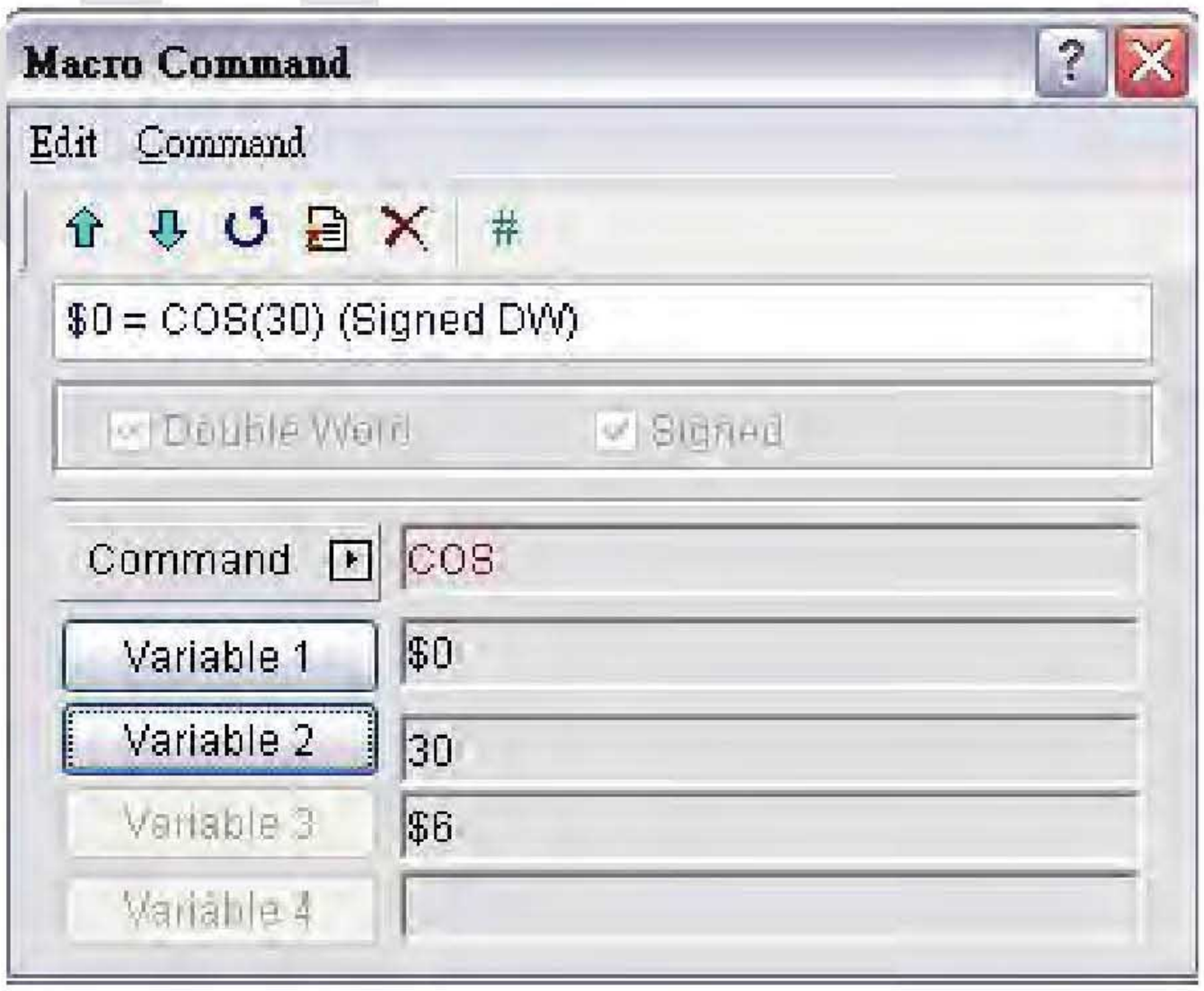
Описание: Вычислить функцию косинуса Var2, сохранить результат вычислений Var1.

Замечания:

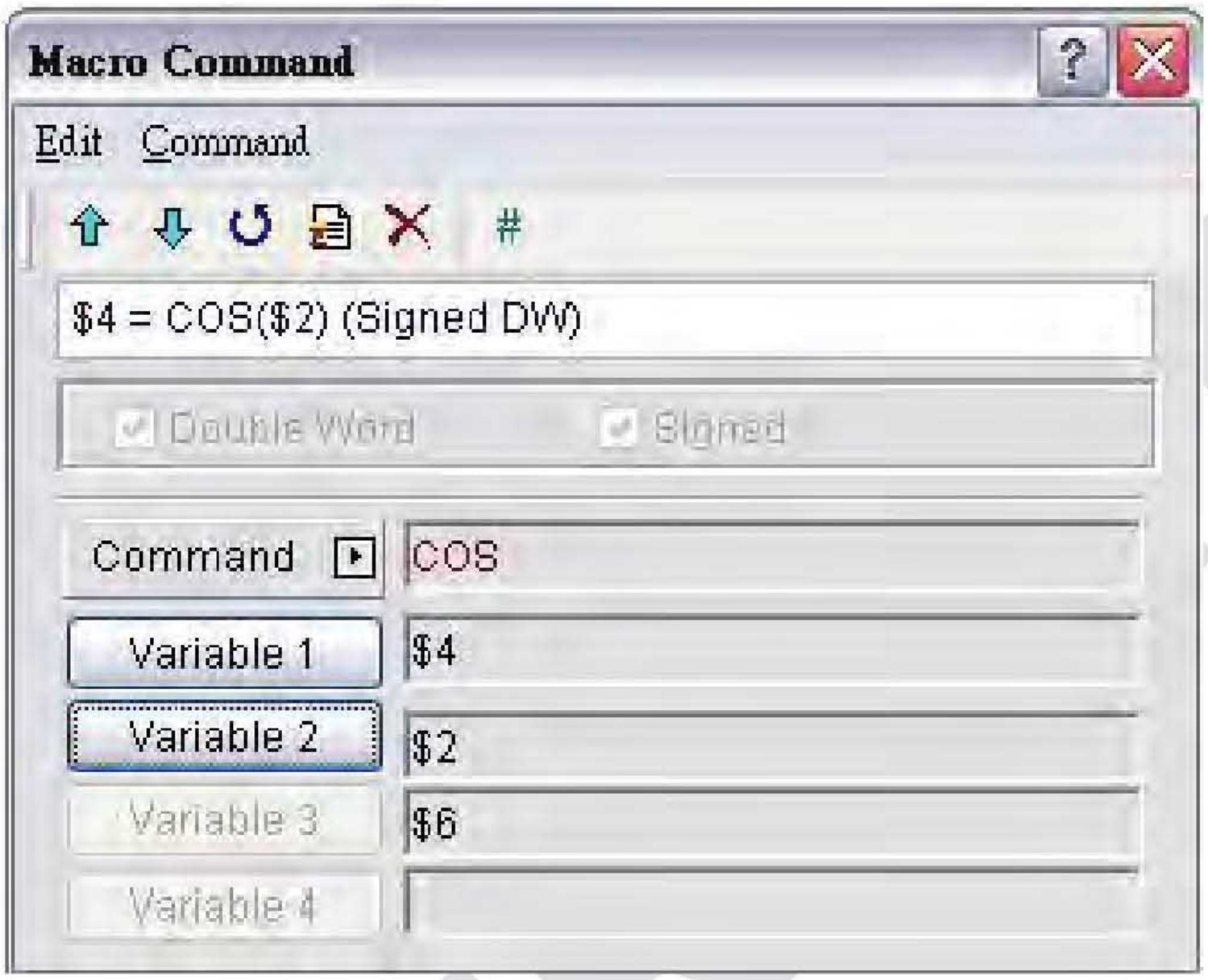
- Значение Var2 задаётся в десятичном целочисленном формате в градусах.
- Эта операция производится с 32-х битными числами со знаком.
- Результат вычислений сохраняется в виде числа с плавающей запятой в формате DWORD. При превышении данными заданной длины, та часть, которая превышает длину, будет отброшена.
- Var1 могут храниться только во внутренней памяти, Var2 - во внутренней памяти или в виде константы.
- Отображаемый формат переменных должен быть всегда с плавающей запятой.
- Входная переменная задаётся в десятичном целочисленном формате со знаком.

Пример

Вычислить COS30º и сохранить результат в регистре $0. (Эта операция производится с 32-х битными числами со знаком.)



Вычислить косинус содержимого регистра $2 и сохранить результат в регистре $4. (Эта операция производится с 32-х битными числами со знаком.)



- **TAN (Tangent Function)** Вычисление тангенса

Выражение: Var1 = TAN (Var2)

Описание: Вычислить функцию косинуса Var2, сохранить результат вычислений Var1.

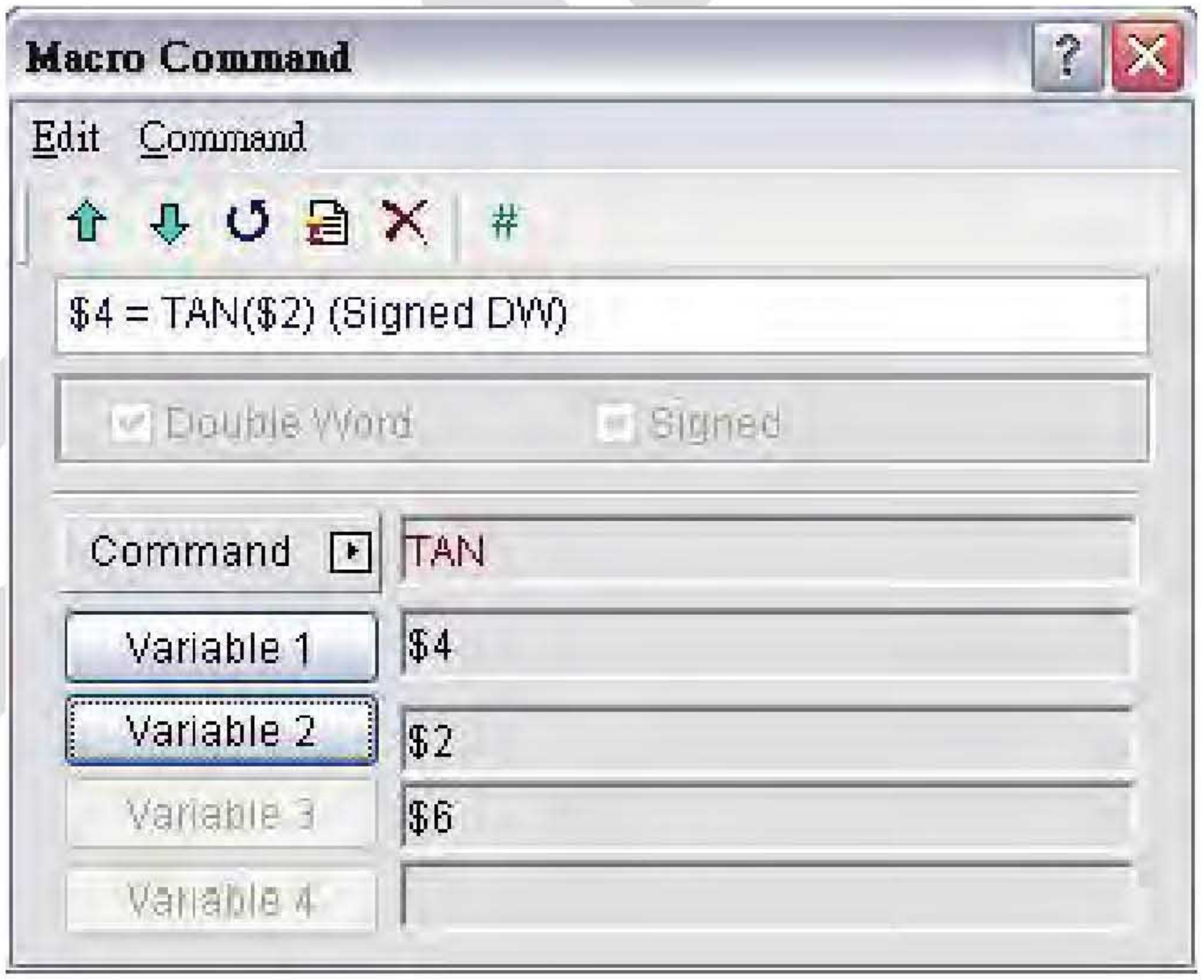
Замечания:

- Значение Var2 задаётся в десятичном целочисленном формате в градусах.
- Эта операция производится с 32-х битными числами со знаком.
- Результат вычислений сохраняется в виде числа в формате DWORD. При превышении данными заданной длины, та часть, которая превышает длину, будет отброшена.
- Var1 может храниться только во внутренней памяти, Var2 - во внутренней памяти или в виде константы.

Пример

Вычислить TAN30º, сохранить результат вычислений в $0.

Вычислить тангенс содержимого регистра $2 результат вычислений сохранить в регистре $4. (Эта операция производится с 32-х битными числами со знаком.)



■ COT (Cotangent Function) Вычисление котангенса

Выражение: Var1 = COT (Var2)

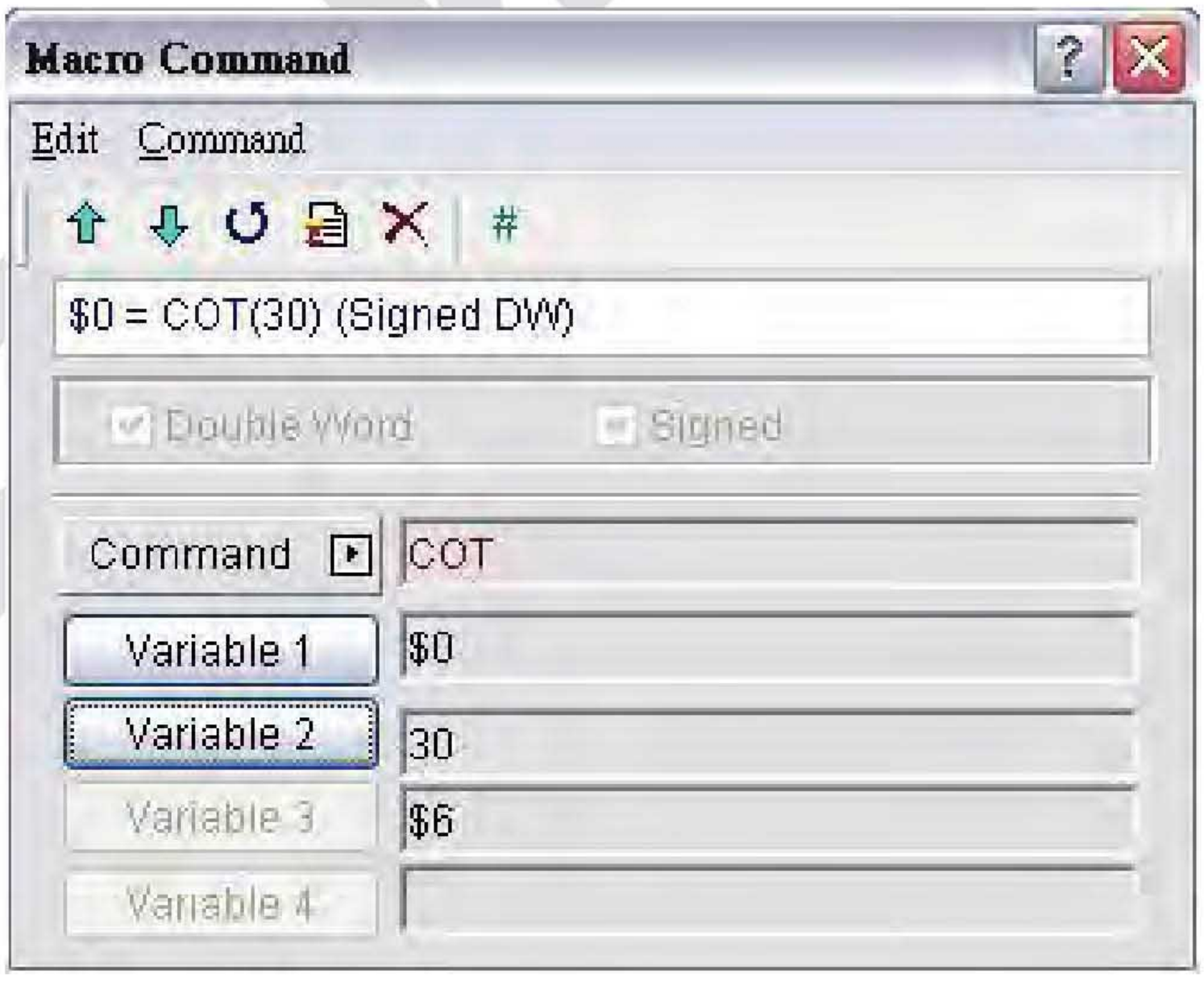
Описание: Вычислить функцию котангенса Var2, сохранить результат вычислений Var1.

Замечания:

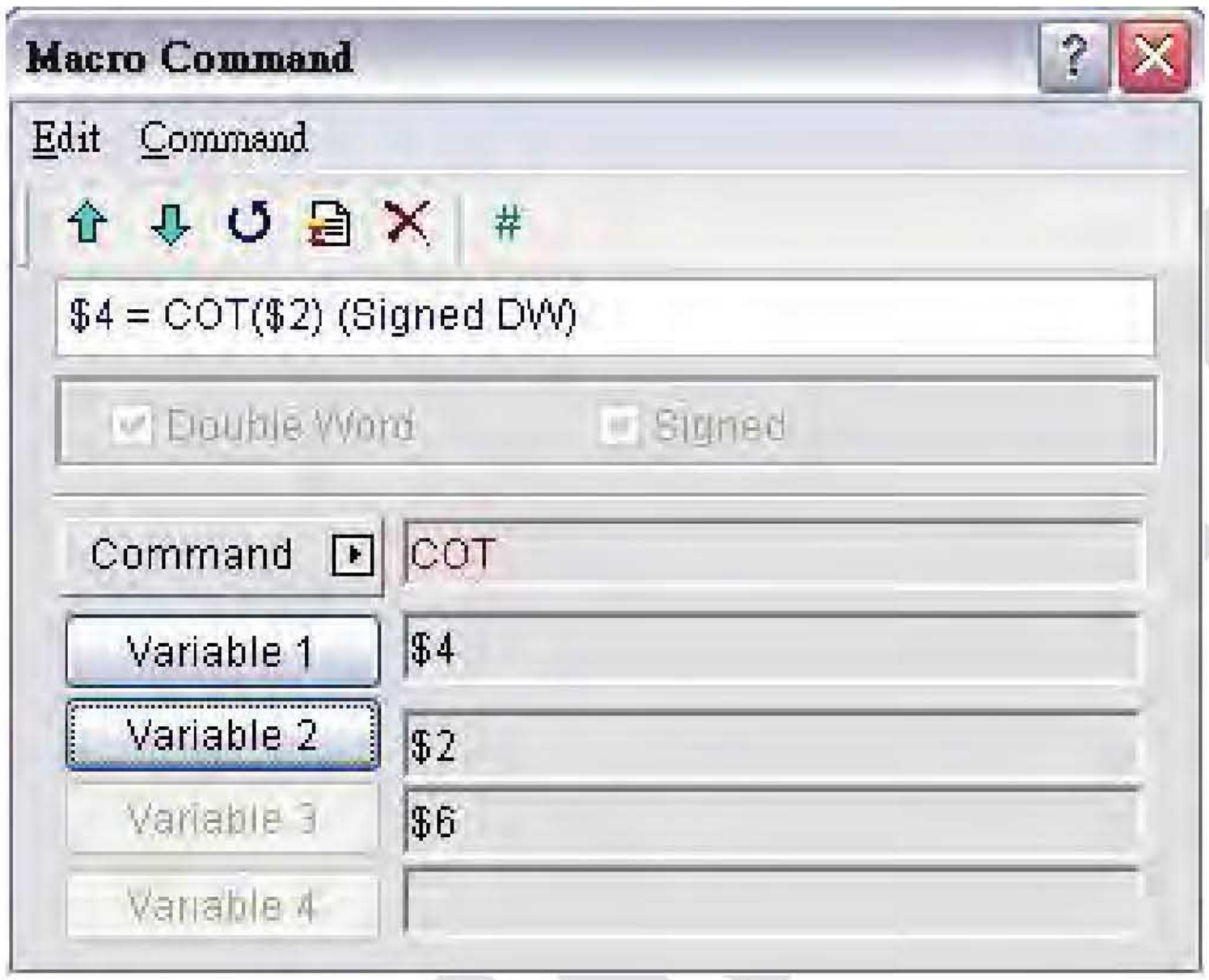
- Значение Var2 задаётся в десятичном целочисленном формате в градусах.
- Эта операция производится с 32-х битными числами с со знаком.
- Результат вычислений сохраняется в виде числа с плавающей запятой в формате DWORD. При превышении данными заданной длины, та часть, которая превышает длину, будет отброшена.
- Var1 может храниться только во внутренней памяти, Var2 - во внутренней памяти или в виде константы.

Пример

Вычислить COT30º, результаты вычислений сохранить в $0. (Эта операция производится с 32-х битными числами с плавающей запятой со знаком.)



Вычислить котангенс содержимого регистра $2, результат вычислений сохранить в $4. (Эта операция производится с 32-х битными числами с плавающей запятой со знаком.)



■ **SEC (Secant Function)** Вычисление секанса

Выражение: Var1 = SEC (Var2)

Описание: Вычислить секанс содержимого регистра Var2, результат вычислений сохранить в регистре Var1.

Замечания:

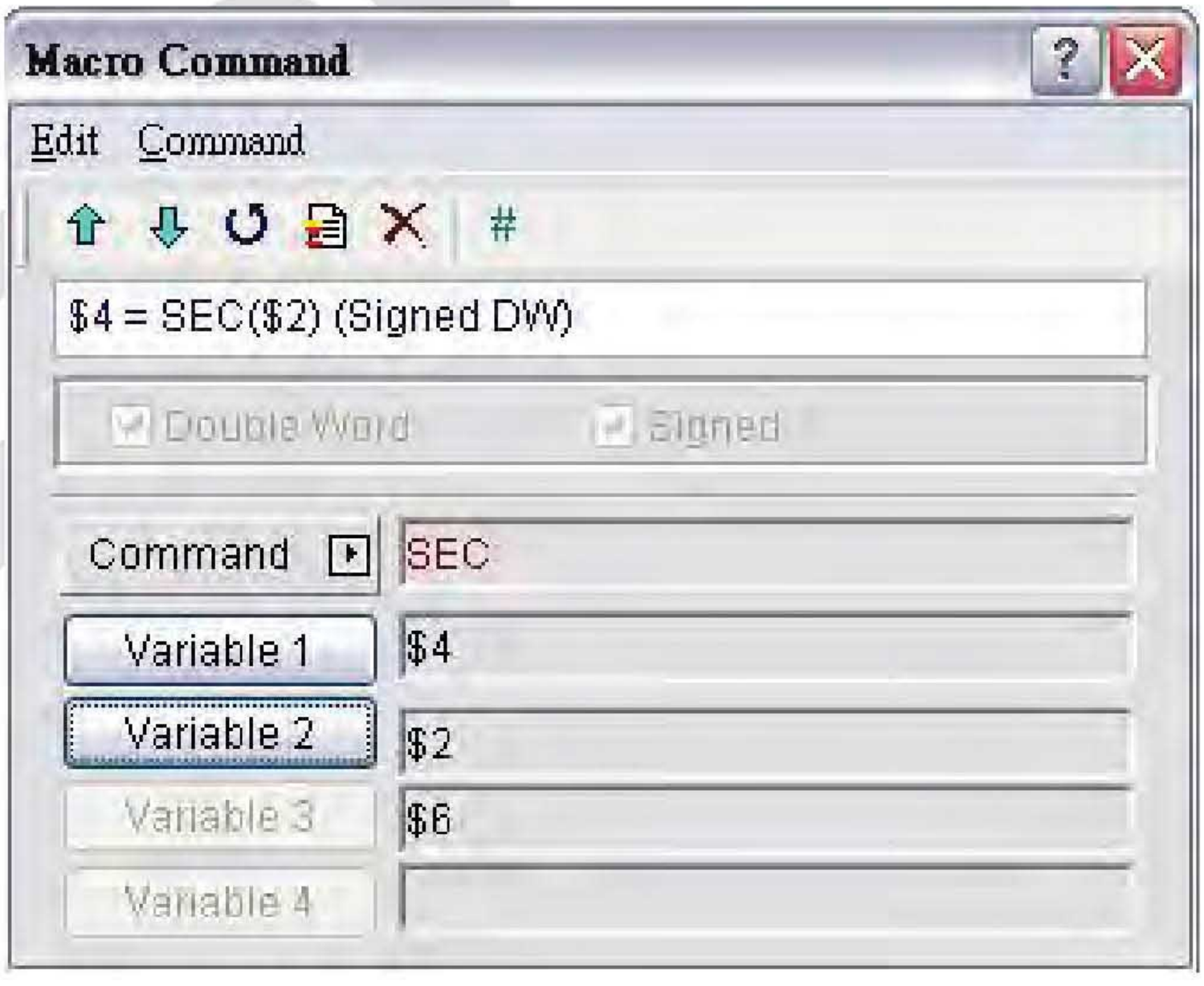
- Значение Var2 задаётся в десятичном целочисленном формате в градусах.
- Эта операция производится с 32-х битными числами со знаком.
- Результат вычислений сохраняется в виде числа со знаком в формате DWORD. При превышении данными заданной длины, та часть, которая превышает длину, будет отброшена.
- Var1 может храниться только во внутренней памяти, Var2 - во внутренней памяти или в виде константы.

Пример

Вычислить SEC30º, результат вычислений сохранить в регистре $0. (Эта операция производится с 32-х битными числами с плавающей запятой со

знаком.)

Вычислить секанс содержимого регистра $2 и сохранить результат в регистре $4 (Эта операция производится с 32-х битными числами с плавающей запятой со знаком.)



■ **CSC (Cosecant Function)** Вычисление косеканса

Выражение: Var1 = CSC (Var2)

Описание: Вычислить косеканс содержимого регистра Var2, результат вычислений сохранить в регистре Var1.

Замечания:

- Значение Var2 задаётся в десятичном целочисленном формате в градусах.
- Эта операция производится с 32-х битными числами со знаком.
- Результат вычислений сохраняется в виде числа со знаком в формате DWORD. При превышении данными заданной длины, та часть, которая превышает длину, будет отброшена.
- Var1 может храниться только во внутренней памяти, Var2 - во внутренней памяти или в виде константы.

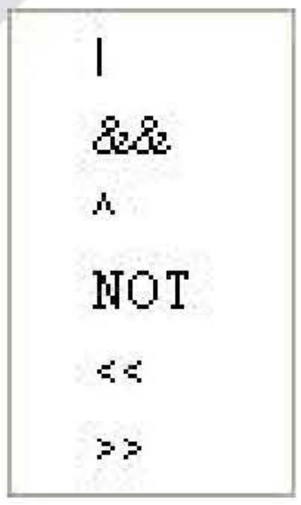
Пример

Вычислить CSC30º, результат вычислений сохранить в регистре $0. (Эта операция производится с 32-х битными числами с плавающей запятой со знаком.)

Вычислить косеканс содержимого регистра $2 и сохранить результат в регистре $4. (Эта операция производится с 32-х битными числами с

плавающей запятой со знаком.)

3.14.3.2 Логические команды



Имеется 6 логических команд **OR**, **AND**, **XOR**, **NOT**, **Shift-left** и **Shift-right**. Каждая команда имеет три операнда, каждый операнд может быть содержимым регистра внутренней памяти или константой, но результат должен содержаться только во внутренней памяти. Формат чисел - **Word** или **Double Word**.

■ **| (Logical OR operation)** Логическое сложение «ИЛИ»

Выражение: Var1 = Var2 | Var3

Описание: Выполнить операцию логическое сложение «ИЛИ» над операндами Var2 и Var3, результат вычислений сохранить в регистре Var1.

Замечания:

- Результат вычислений может быть сохранён в формате WORD или DWORD.
- Var1 может храниться только во внутренней памяти, Var2 и Var3 - во внутренней памяти или в виде константы.

Пример

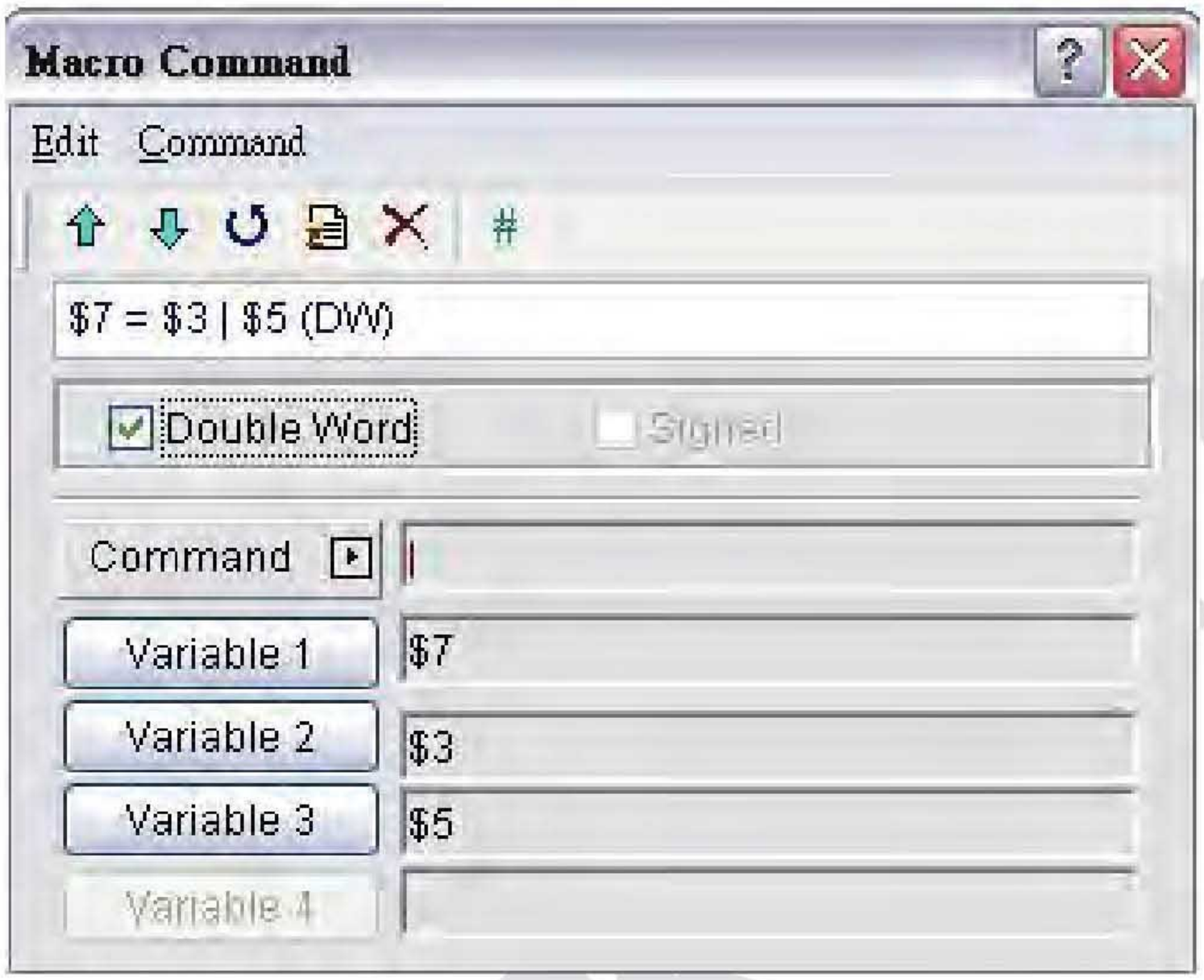
Провести операцию логического сложения «ИЛИ» над содержимым регистров $0 and $1, сохранить результат в регистре $2 (эта операция выполняется над 16-ю битными числами без знака)

$2 = $0 | $1 Сохранить результат в регистре $2.

Если $0 = F000H, $1 = 0F00H, тогда $2 = FF00H.

Провести операцию логического сложения «ИЛИ» над содержимым регистров $3 and $5, сохранить результат в регистре $7 (эта операция

выполняется над 32-х битными числами без знака)



$7 = $3 | $5 (DW) Сохранить результат регистра $7.

Если $3 = F000F000H, $5 = 0F000F00H, тогда $7= FF00FF00H.

- **&& (Logical AND operation)** Логическое умножение «И»

  Выражение: Var1 = Var2 && Var3

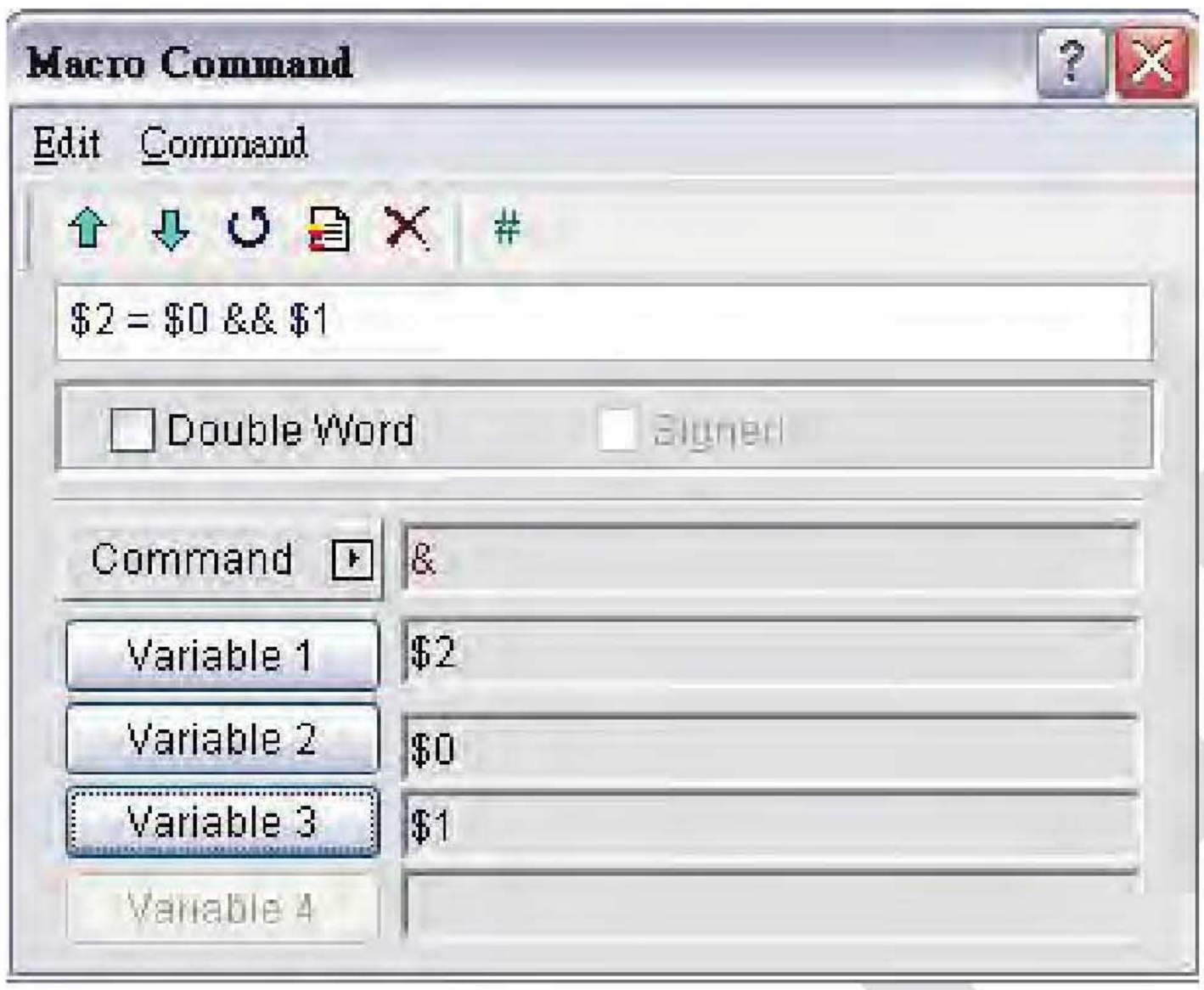
  Описание: Выполнить операцию логическое умножение «И» над операндами Var2 и Var3, результат вычислений сохранить в регистре Var1.

  Замечания:

  - Результат вычислений может быть сохранён в формате WORD или DWORD.
  - Var1 может храниться только во внутренней памяти, Var2 и Var3 - во внутренней памяти или в виде константы.

Пример

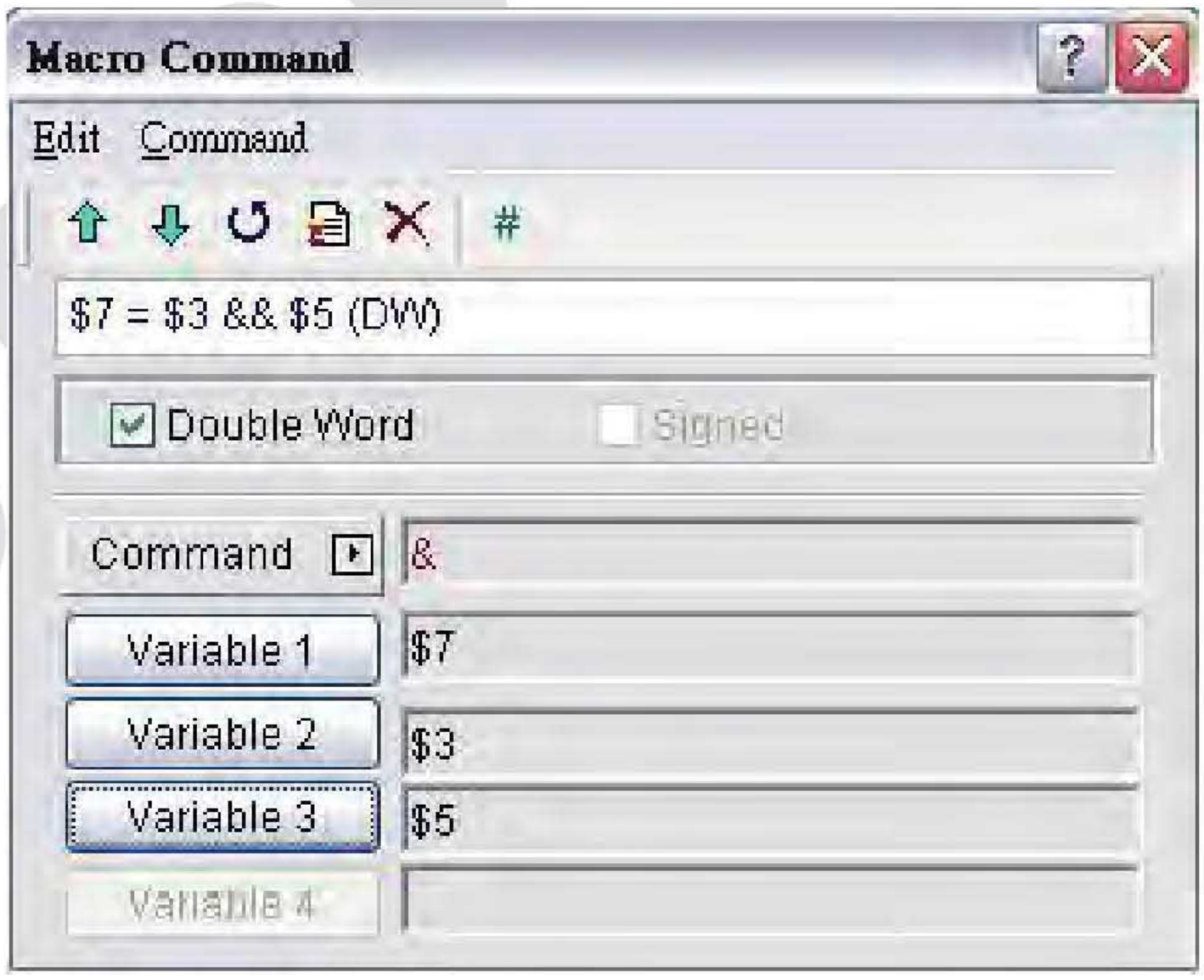
Провести операцию логического умножения «И» над содержимым регистров $0 and $1, сохранить результат в регистре $2 (эта операция выполняется над 16-ю битными числами без знака)



$2 = $0 && $1 Результат сохранить в регистре $2.

Если $0 = F000H, $1 = 0F00H, то $2 = 0000H.

Провести операцию логического умножения «И» над содержимым регистров $3 and $5, сохранить результат в регистре $7. (Эта операция выполняется над 32-х битными числами без знака)



$7 = $3 && $5 (DW) Результат сохранить в регистре $7.

Если $3 = F000F000H, $5 = 0F000F00H, то $7 = 00000000H.

■ **^ (Logical XOR operation)** Исключающее «ИЛИ»

Выражение: Var1 = Var2 ^ Var3

Описание: Выполнить операцию исключающее «ИЛИ» над операндами Var2 и Var3, результат вычислений сохранить в регистре Var1.

Замечания:

- Результат вычислений может быть сохранён в формате WORD или DWORD.
- Var1 может храниться только во внутренней памяти, Var2 и Var3 - во внутренней памяти или в виде константы.

Пример

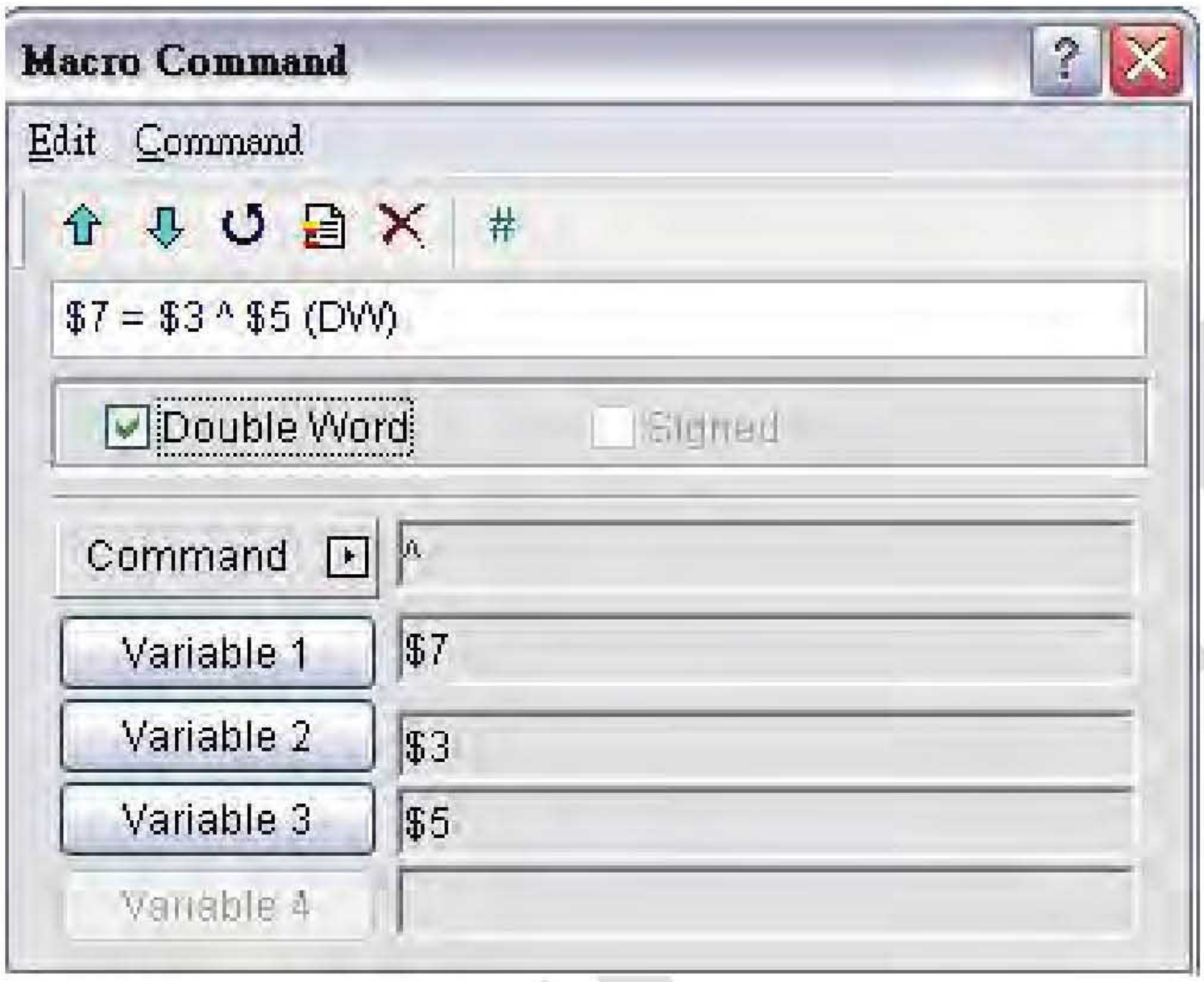
Провести операцию исключающее «ИЛИ» над содержимым регистров $0 и $1, сохранить результат в регистре $2 (эта операция выполняется над 16-ю битными числами без знака)

$2 = $0 ^ $1, результат сохранить в $2.

Если $0 = F100H, $1 = 0F00H, то $2 = FE00H.

Провести операцию исключающее «ИЛИ» над содержимым регистров $3 и $5, сохранить результат в регистре $7 (Эта операция выполняется над 32-х

битными числами без знака)



$7 = $3 ^ $5 (DW) Сохранить результат в регистре $7.

Если $3 = F100F100H, $5 = 0F000F00H, то $7= FE00FE00H.

- **NOT (Logical NOT operation)** Логическое отрицание (поразрядное инвертирование)

  Выражение: Var1 = NOT Var2

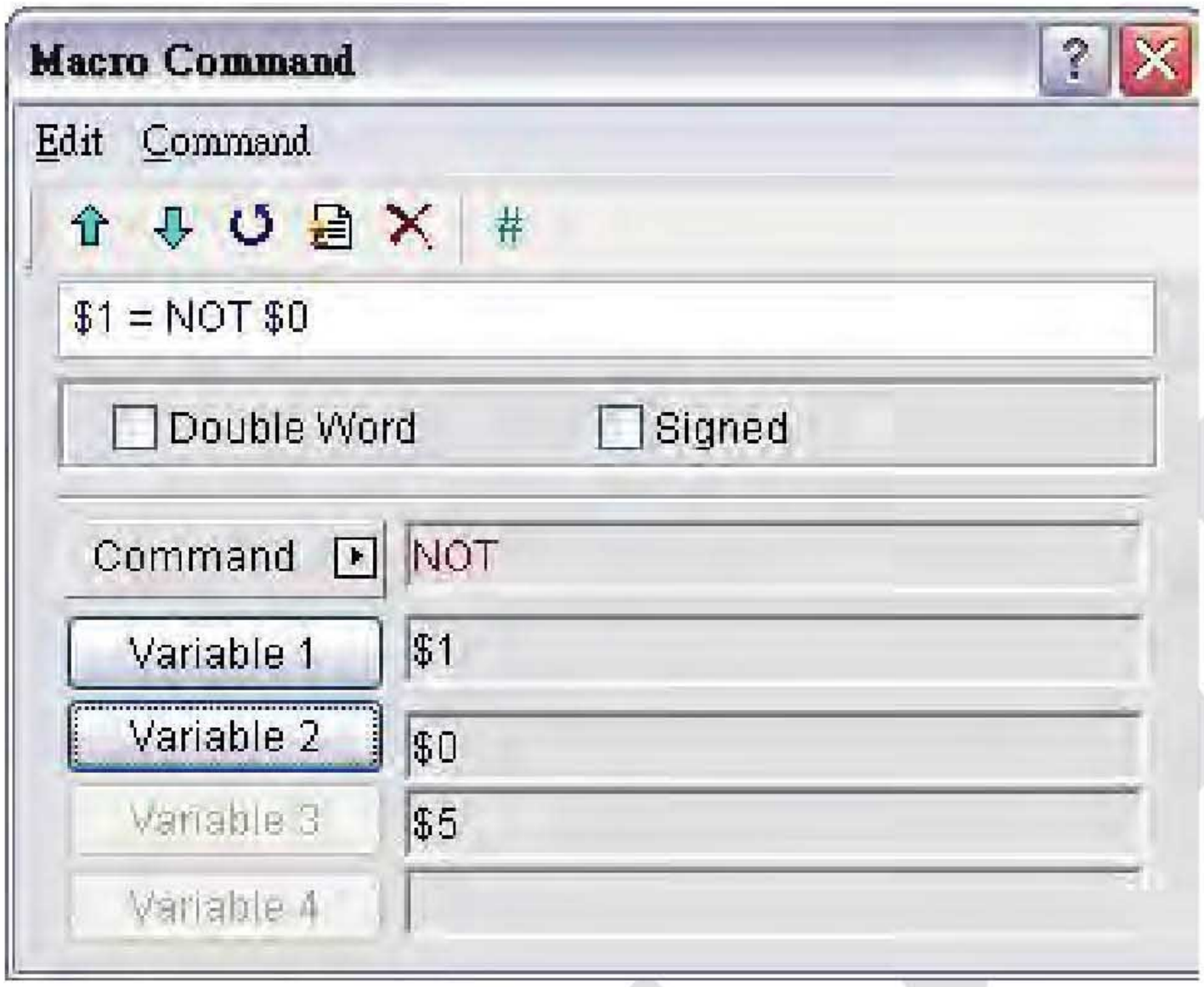
  Описание: Выполнить операцию инвертирования переменной Var2, результаты сохранить в переменной Var1.

  Замечания:

  - Результат вычислений может быть сохранён в формате WORD или DWORD.
  - Var1 может храниться только во внутренней памяти, Var2 - во внутренней памяти или в виде константы.

Пример

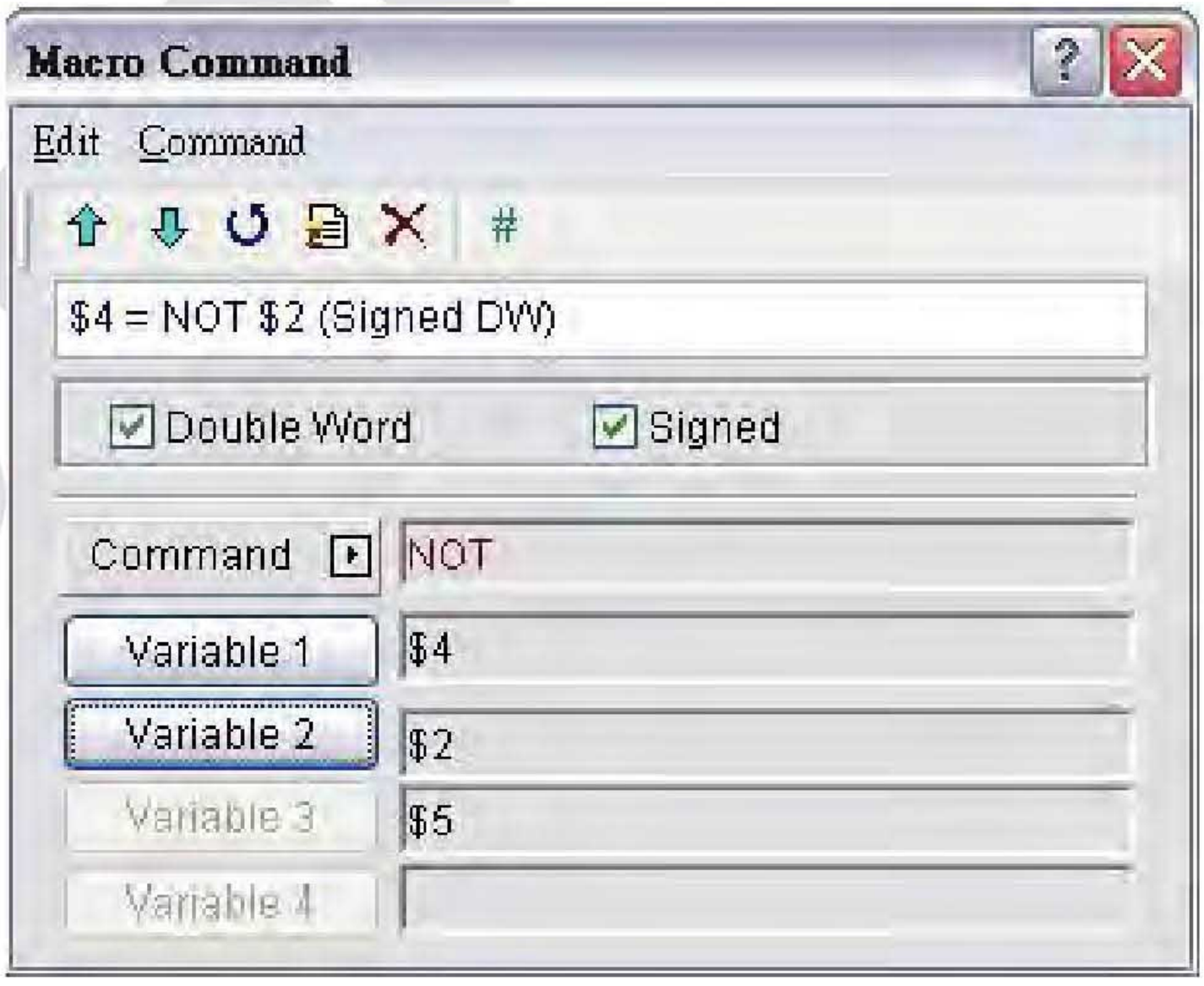
Произвестии операцию логического отрицания над содержимым регистра $0, результат сохранить в регистре $1 (эта операция выполняется над 16-ю битными числами без знака)



$1 = NOT $0. Сохранить результат в регистре $1.

Если $0 = F100H, то $1 = 0EFFH.

Произвести операцию логического отрицания над содержимым регистра $2, результат сохранить в регистре $4 (Эта операция выполняется над 32-х битными числами без знака)



$4 = NOT $2 (Signed DW). Сохранить результат в регистре $4.

Если $2 = F100 F100H, то $4 = 0EFF 0EFFH.

■ **<< (SHL, Logical Shift-left operation)** Операция логического сдвига влево

Выражение: Var1 = Var2 << Var3

Описание: Сдвиг влево содержимого Var2 (WORD/DWORD) (величина сдвига задаётся переменной Var3), результат сохраняется в переменной Var1.

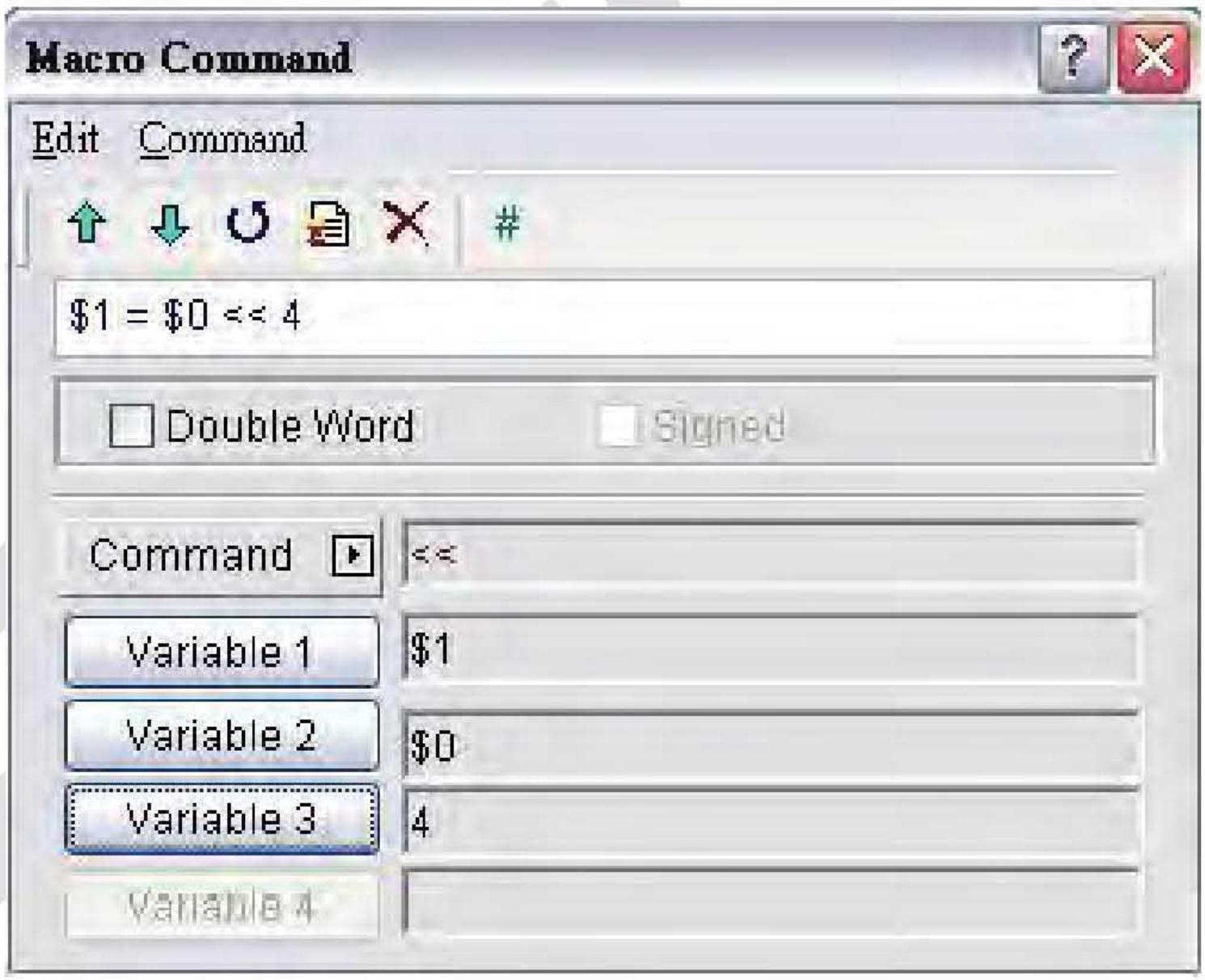
Замечания:

- Результат сохранячется в формате WORD или DWORD.
- Var1 может храниться только во внутренней памяти, Var2 и Var3 - во внутренней памяти или в виде константы.

Пример

$0 = F100H

Сдвиг влево на 4 разряда содержимого $0. Содержимое $1 = 1000H. (Эта операция выполняется над 16-ю битными числами без знака)



$2 = F1000000H

Сдвиг влево на 4 разряда содержимого $2. Содержимое $4 = 10000000H. (Эта операция выполняется над 32-х битными числами без знака).

- **>> (SHR, Logical Shift-right operation)** Операция логического сдвига вправо

Выражение: Var1 = Var2 >> Var3

Описание: Сдвиг вправо содержимого Var2 (WORD/DWORD) (величина сдвига задаётся переменной Var3), результат сохраняется в переменной Var1.

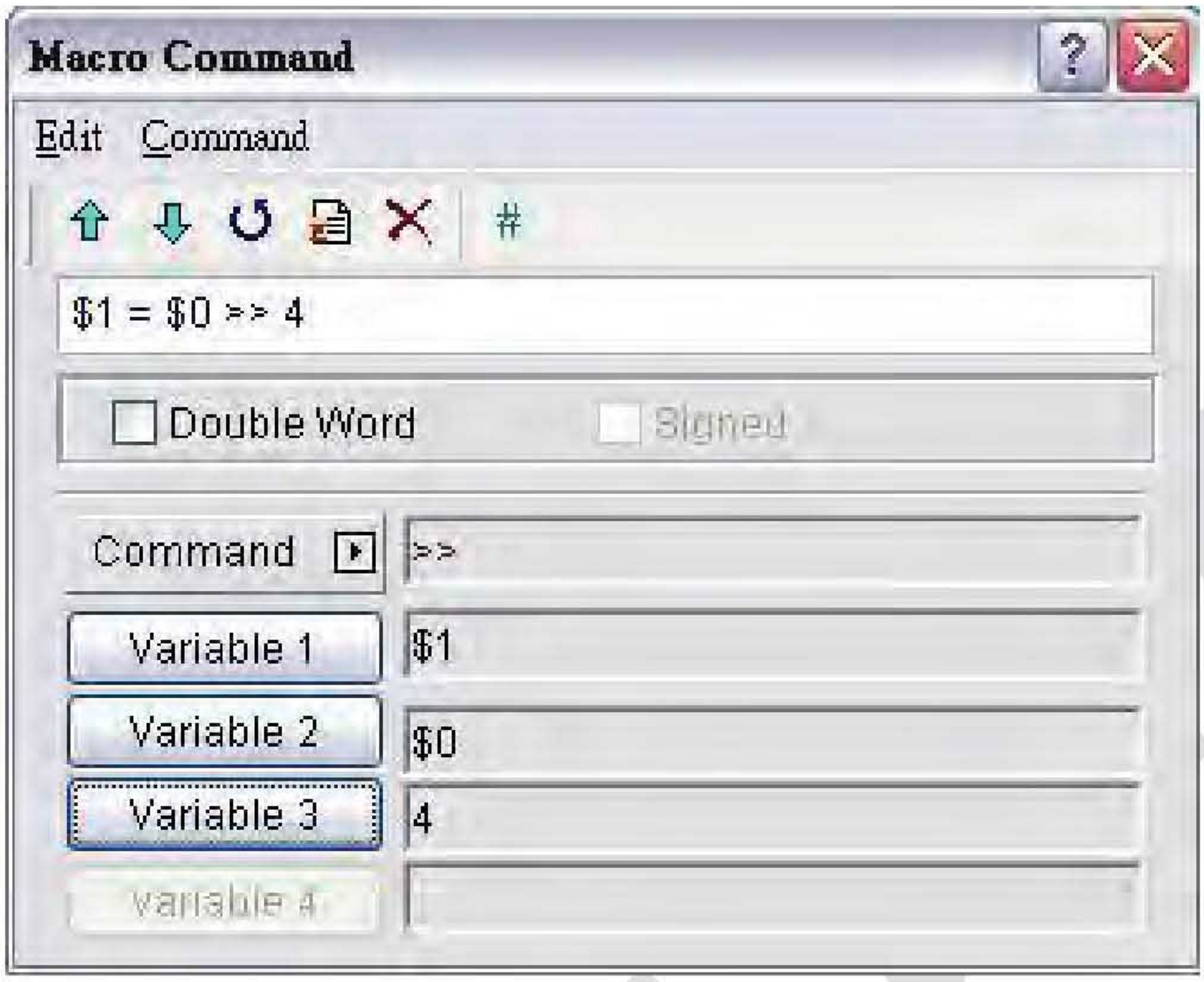
Замечания:

- Результат сохраняется в формате WORD или DWORD.
- При превышении сдвигаемых данных длины адреса, та часть, которая превышает длину, будет отброшена.
- Var1 может храниться только во внутренней памяти, Var2 и Var3 - во внутренней памяти или в виде константы.

Пример

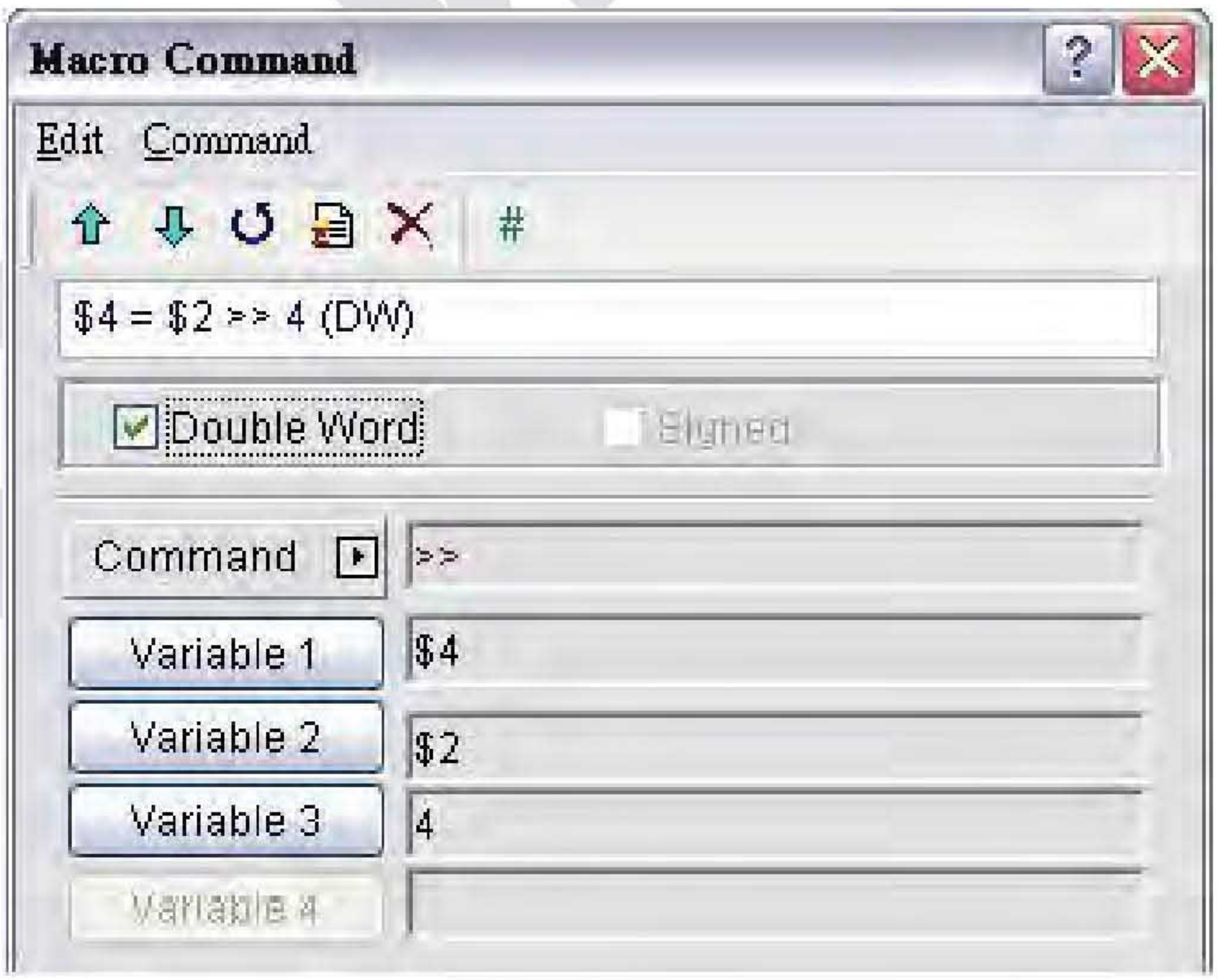
$0 = F100H

Сдвиг вправо на 4 разряда содержимого $0. Содержимое $1 = 0F10H. (Эта операция выполняется над 16-ю битными числами без знака)



$2 = F1000000H

Сдвиг вправо на 4 разряда содержимого $2. Содержимое $1 = 0F10H $4 = 0F100000H. (Эта операция выполняется над 32-х битными числами без знака)



3.14.3.3 Data Transfer - Команды передачи данных

Имеется пять команд передачи данных **MOV**, **BMOV**, **FILL**, **FILLASC** и **FMOV**.

■ **MOV (Transfer Data)** Передача данных

Выражение: Var1 = Var2

Описание: Передача значения операнда Var2 в операнд Var1. Значение операнда Var2 после выполнения команды не изменится.

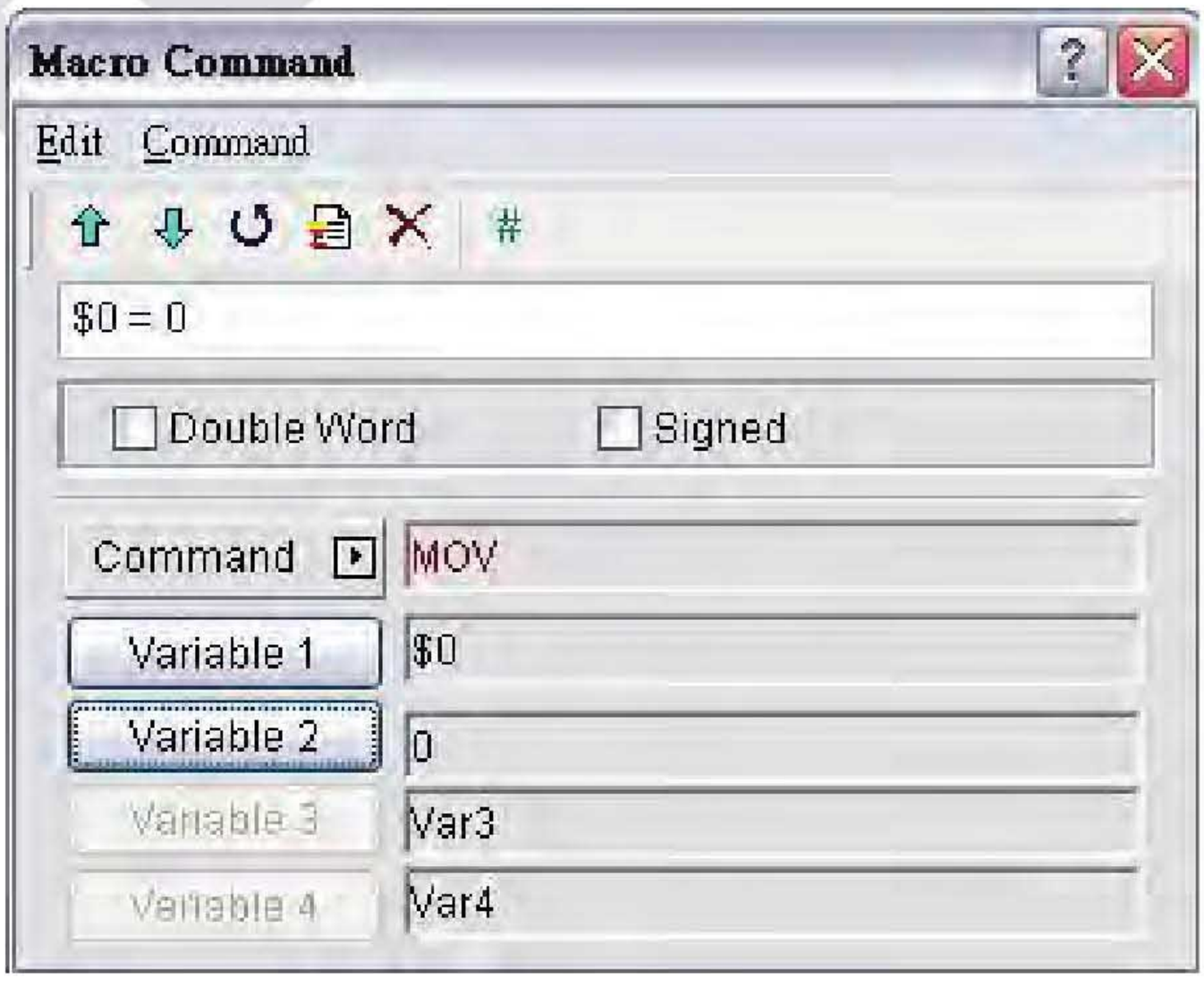
Замечания:

- Результат сохраняется в формате WORD или DWORD.
- Var1 может храниться только во внутренней памяти, Var2 - во внутренней памяти или в виде константы.

Пример

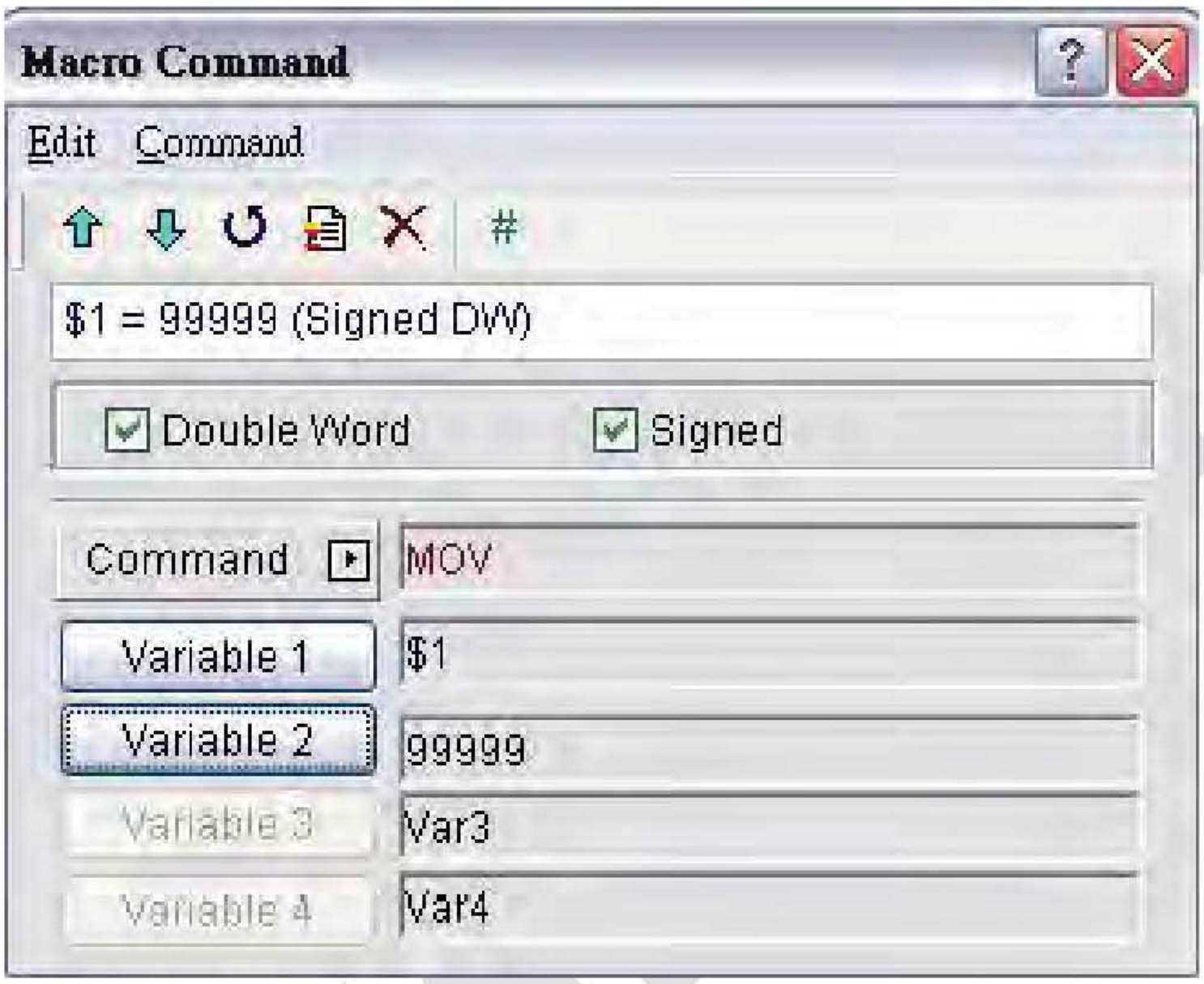
$0 = 0

Запись в регистр внутренней памяти $0 числа 0 (операция выполняется над 16-ю битными числами без знака).



$1 = 99999 (Signed DW)

Запись в регистр внутренней памяти $1 числа 99999 (операция выполняется над 32-х битными числами со знаком).



- **BMOV (Block Move Copy Block)** Передача блока данных

Выражение: BMOV (Var1, Var2, Var3)

Описание: BMOV (Var1, Var2, Var3) означает копирование блока данных из нескольких последовательных регистров (количество задаётся операндом Var3), начиная с адреса операнда Var2 в регистры, начиная с адреса Var1.

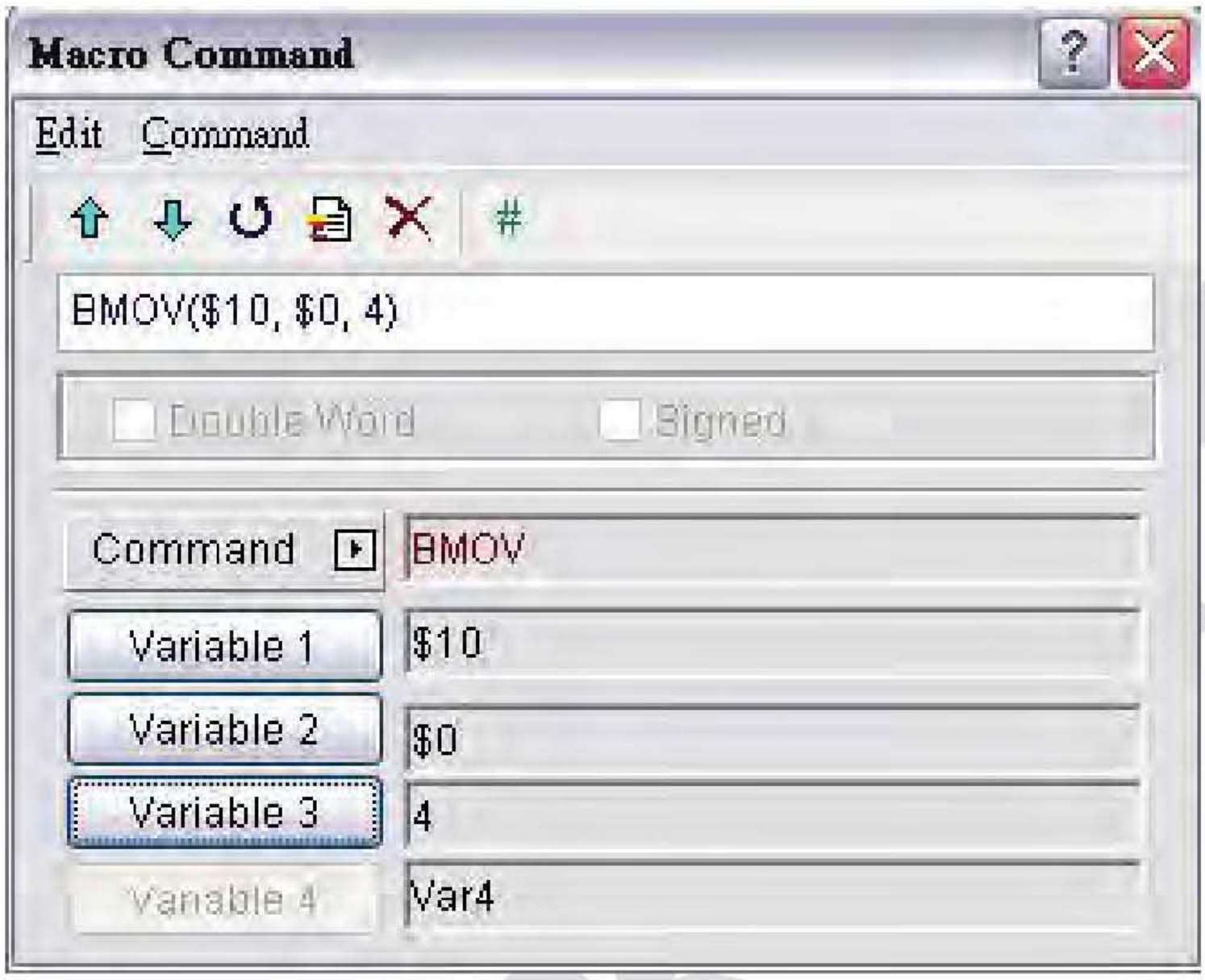
Замечания:

- Результат сохраняется в формате WORD.
- Если длина блока выходит за пределы внутренней памяти или макс. числа регистров ПЛК, при компиляции будет выдано соответствующее сообщение.
- Var1 и Var3 могут храниться только во внутренней памяти, Var2 - во внутренней памяти или в виде константы.

Пример

Скопировать данные регистров $0, $1, $2, $3, $4 в регистры $10, $11, $12, $13

(операция выполняется над 16-ю битными числами без знака)



■ **FILL (Fill the Memory)** Размножение данных

Выражение: FILL (Var1, Var2, Var3)

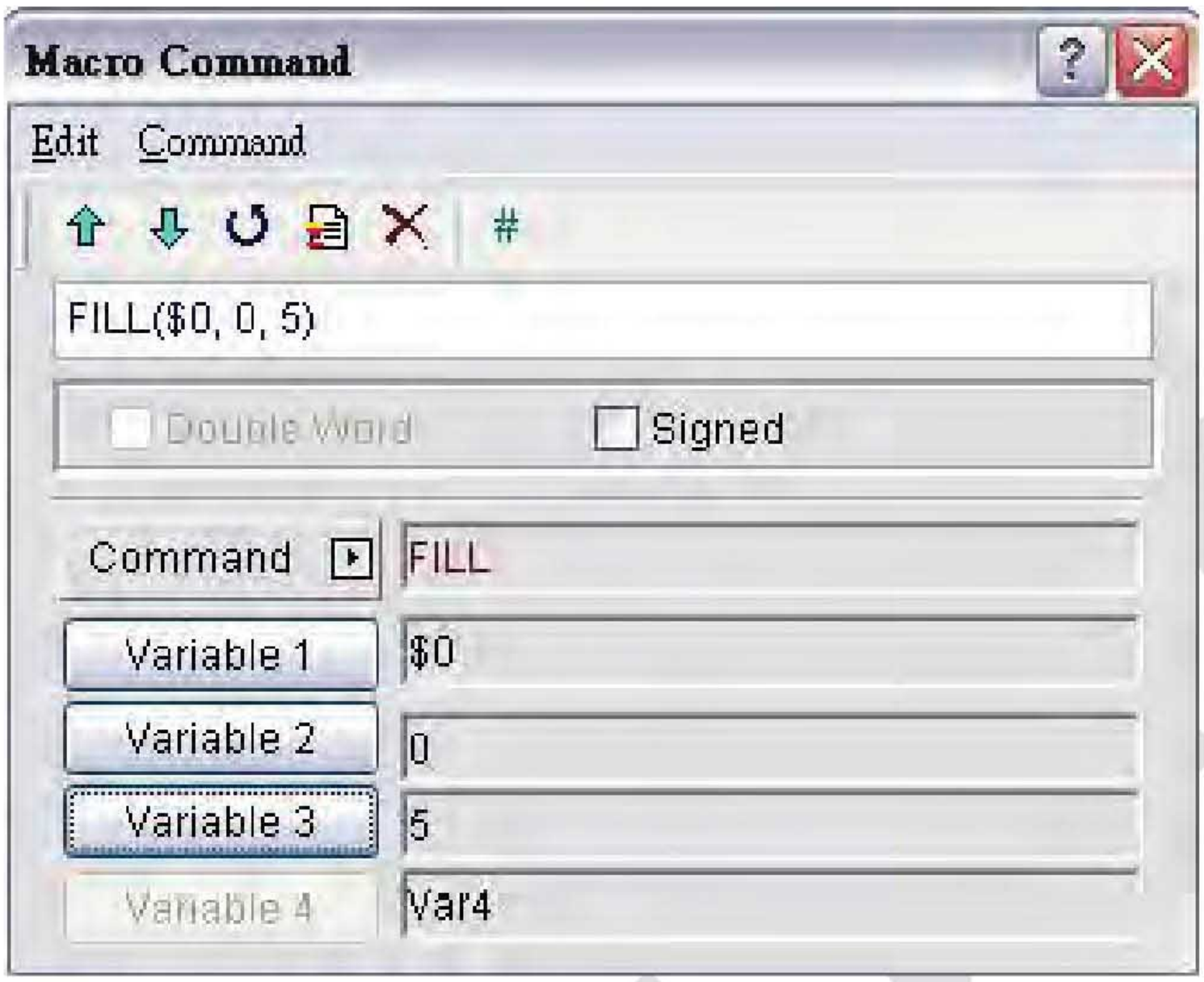
Описание: FILL (Var1, Var2, Var3) означает копирование данных из одного операнда Var1 в несколько (количество определяется значением операнда Var3).

Замечания:

- Результат сохраняется в формате WORD.
- Если длина блока выходит за пределы внутренней памяти или макс. числа регистров ПЛК, при компиляции будет выдано соответствующее сообщение.
- Var1 может храниться только во внутренней памяти, Var2 - во внутренней памяти или в виде константы.

Пример

Записать в регистры $0, $1, $2, $3, $4 число 0. (Операция выполняется над 16-ю битными числами без знака.)



- **FILLASC (Convert Text to ASCII code)** Преобразование текста в ASCII код.

Выражение: FILLASC (Var1, “Var2”)

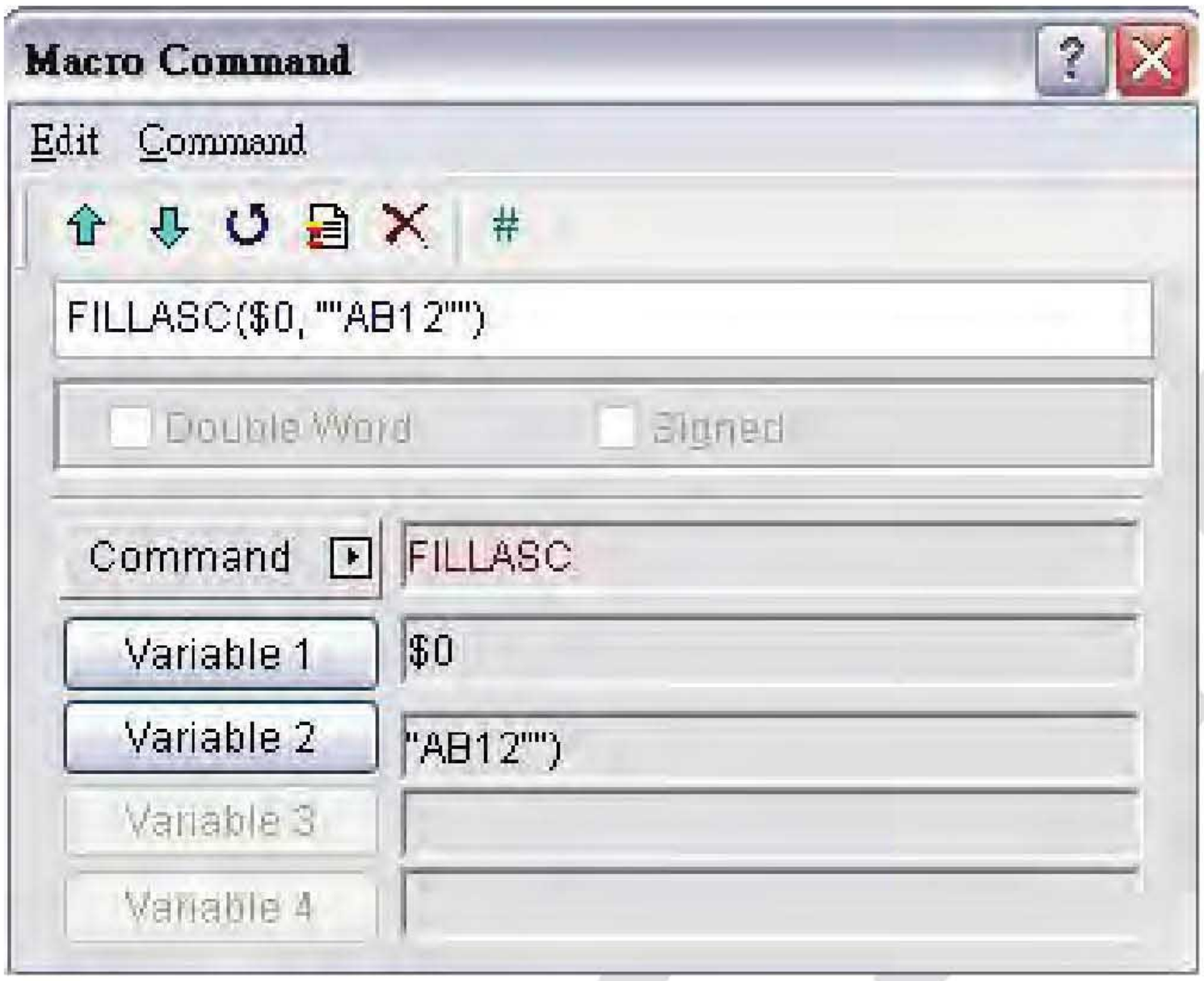
Описание: FILLASC (Var1, “Var2”) означает преобразование текста содержащегося в операнде Var2 в ASCII код и сохранение его в операнде Var1.

Замечания:

- Результат сохраняется в формате WORD без знака.
- Максимальный размер блока 128 слов.
- По одному адресу сохраняются два слова. Остальные слова размещаются по последующим адресам.
- При записи конвертированного текста, младшие и старшие байты ASCII кодов будут заменены местами.
- Var1 может храниться только во внутренней памяти, Var2 - во внутренней памяти или в виде константы.

Пример

После выполнения команды FILLASC, в регистре $0 будет записано число 4241H, а в регистре $1 - 3130H. (Результат сохраняется в формате WORD без знака).



- **FMOV (Transfer Floating Point Data)** Передача данных с плавающей запятой

Выражение: Var1 = FMOV (Var2)

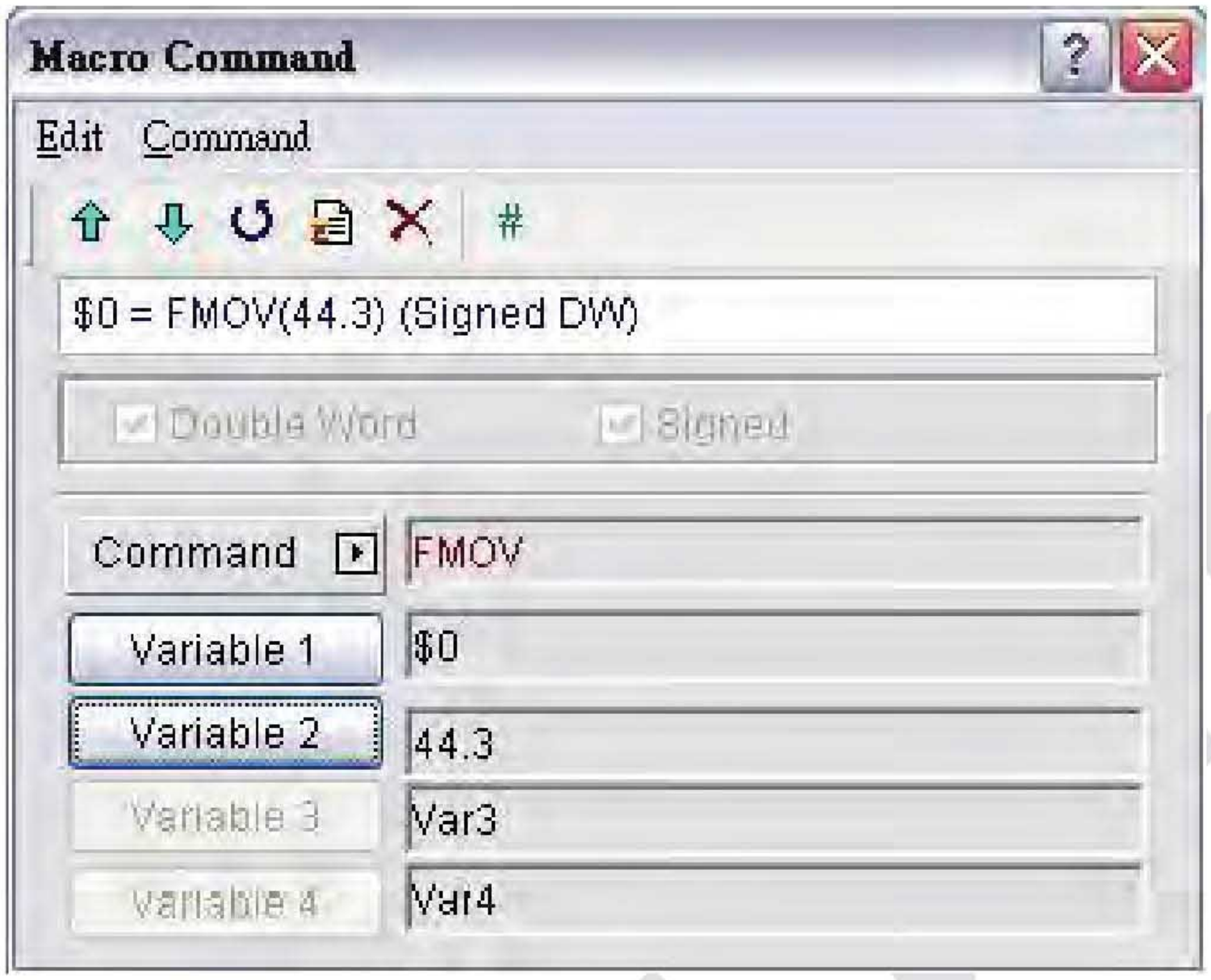
Описание: Передача, содержащихся в операнде Var2 данных с плавающей запятой в операнд Var1. Содержимое операнда Var1 не меняется после выполнения операции.

Замечания:

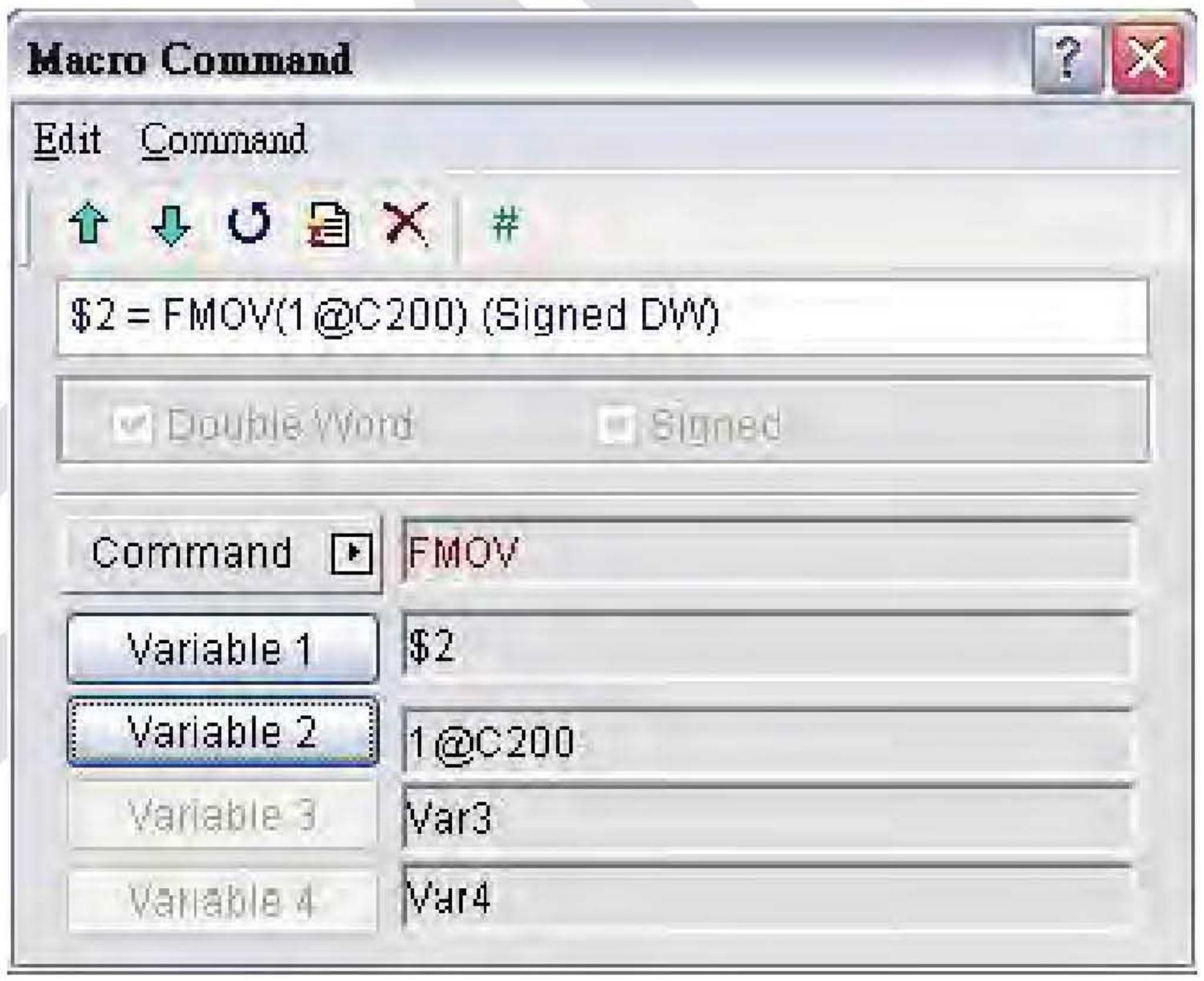
- Результат сохраняется в формате DWORD со знаком.
- Var1 может храниться в адресе контроллера или во внутренней памяти. Var2 - в адресе контроллера, во внутренней памяти или в виде константы.

Пример

Записать число 44.3 в регистр внутренней памяти $0. (Операция выполняется над 32-х битными числами со знаком).



Передать данные из регистра контроллера ПЛК 1@C200 в регистр внутренней памяти $2 (операция выполняется над 32-х битными числами со знаком).



3.14.3.4 Data Conversion - Преобразование данных

В разделе описаны несколько команд, выполняющих преобразование данных.

- **BCD (Convert BIN Data into BCDValue)** Преобразование BIN (двоичных данных) в формат BCD (двоичнодесятичный)

  Выражение: Var1 = BCD (Var2)

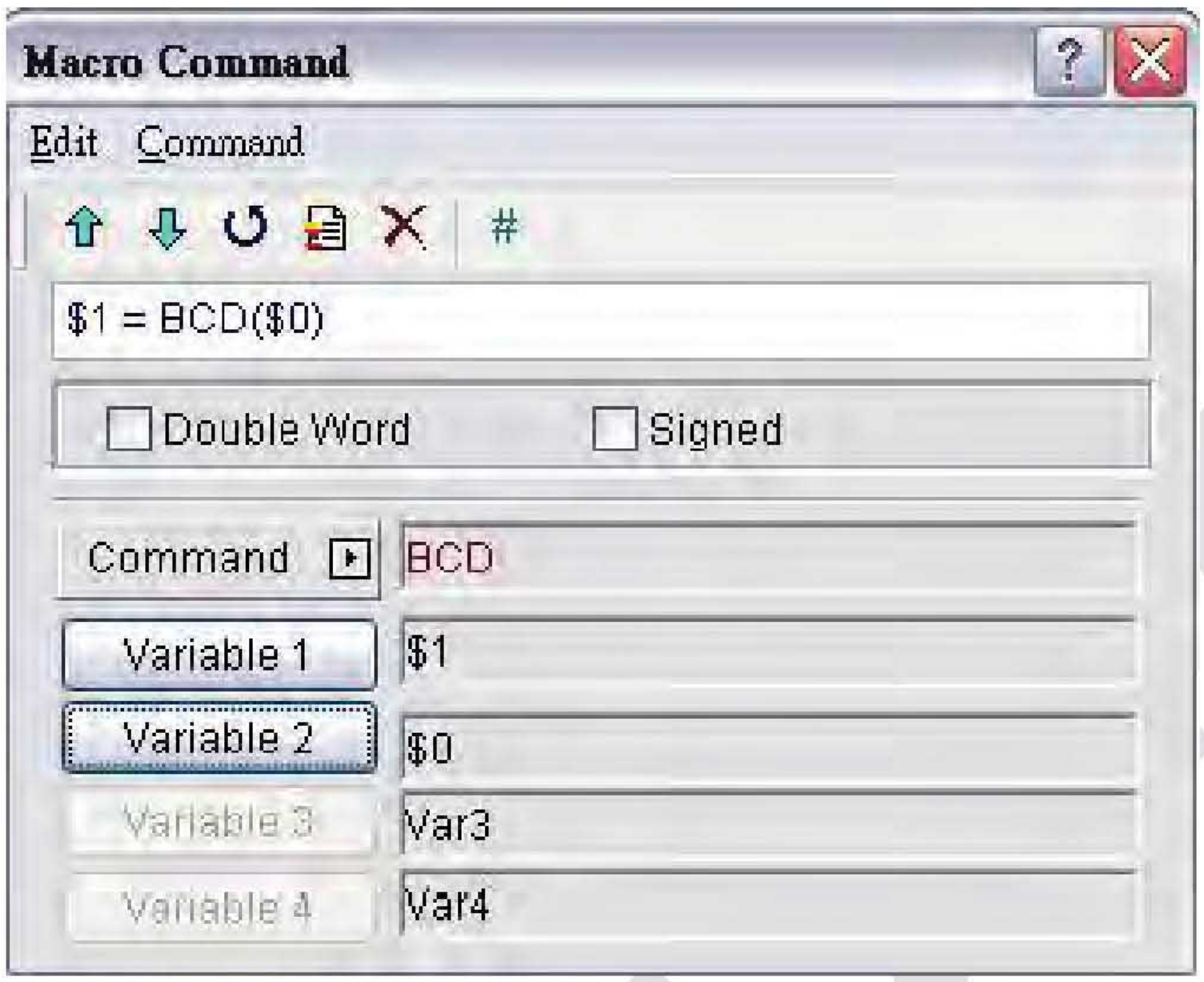
  Описание: Двоичные данные, содержащиеся в операнде Var2 преобразуются в двоичнодесятичный формат и сохраняются в операнде Var1.

  Замечания:

  - Результат сохраняется в формате DWORD или WORD, как со знаком, так и без знака.
  - Операнды Var1 и Var2 могут содержаться только во внутренней памяти.

Пример

После выполнения BCD команды, двоичные данные из регистра $0 преобразуются в формат BCD и сохраняются в регистре $1. (Операция выполняется над 16-ю битными числами без знака).



- **BIN (Converts BCD Data into BIN Value)** Преобразование данных в BCD формате (двоичнодесятичных) в формат BIN (двоичный)

Выражение: Var1 = BIN (Var2)

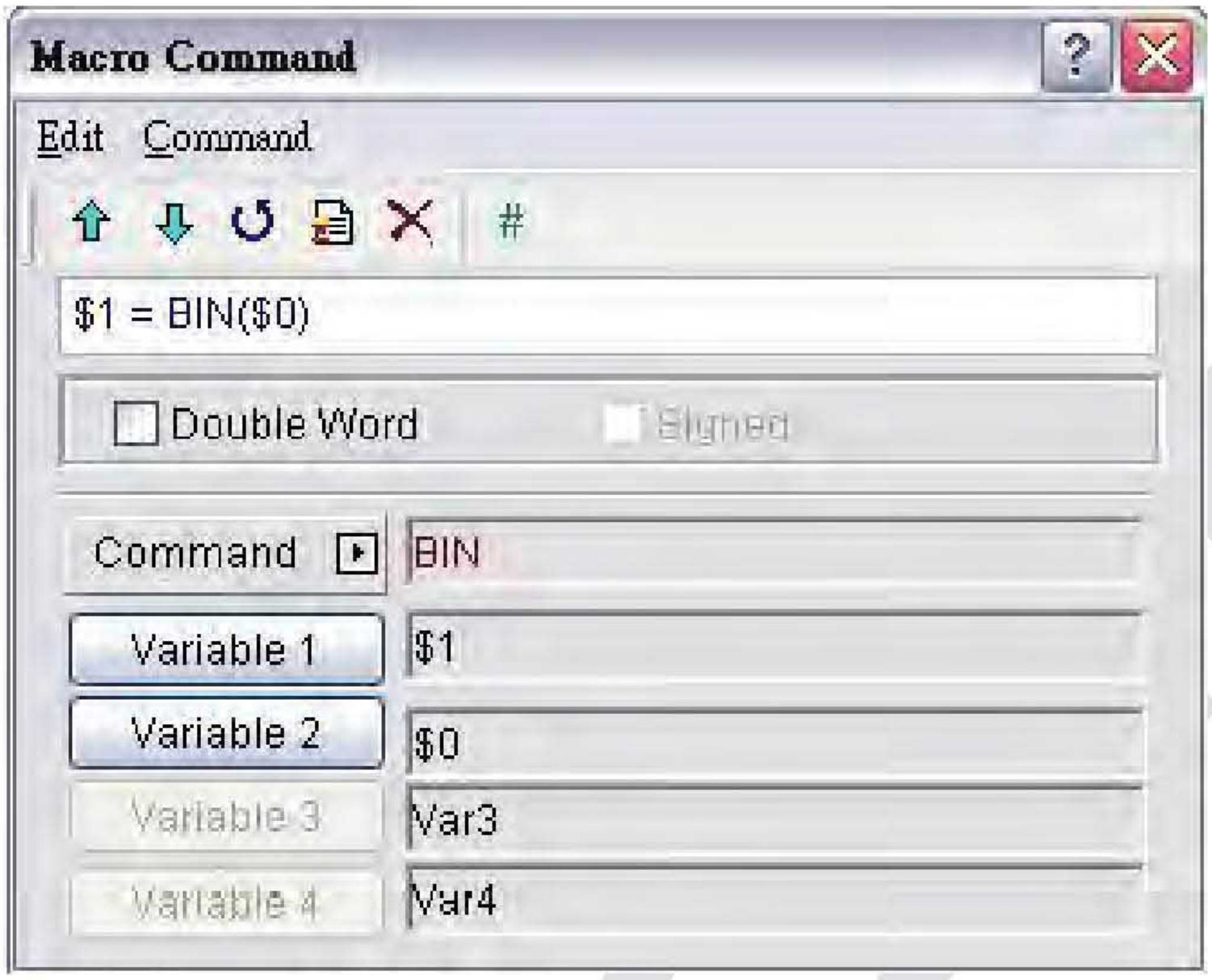
Описание: BCD данные, содержащиеся в операнде Var2 преобразуются в двоичный формат и сохраняются в операнде Var1.

Замечания:

- Результат сохраняется в формате DWORD или WORD, как со знаком, так и без знака.
- Операнды Var1 и Var2 могут содержаться только во внутренней памяти

Пример

После выполнения BIN команды, двоичнодесятичные данные из регистра $0 преобразуются в двоичный формат BIN и сохраняются в регистре $1. (Операция выполняется над 16-ю битными числами без знака).



- **TODWORD** (Преобразование данных в формате WORD в формат DWORD)

Выражение: Var1 = TODWORD (Var2)

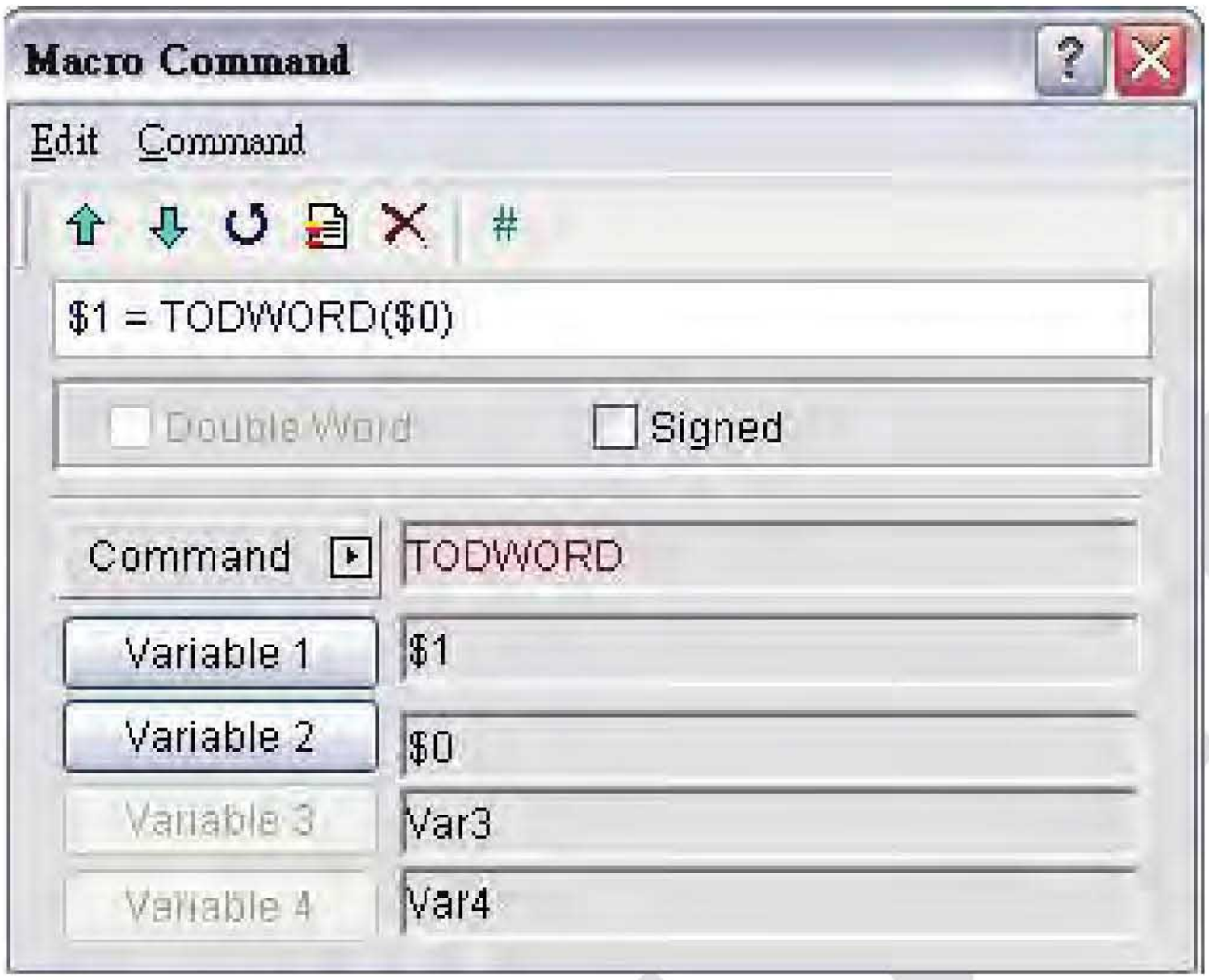
Описание: Содержимое операнда Var2 в формате WORD преобразуется в формат DWORD и сохраняется в операнде Var1.

Замечания:

- Результат сохраняется в формате DWORD или WORD, как со знаком, так и без знака.
- Операнды Var1 и Var2 могут содержаться только во внутренней памяти

Пример

После выпорлнения команды TODWORD, значение числа WORD в регистре $0 преобразуется в значение числа DWORD и сохраняется в регистре $1. Это означает, что число в формате DWORD занимает регистры $1 и $2. (Эта операция производится над положительными 16-ю разрядными числами).



■ **TOWORD** (Преобразование данных в формате BYTE в формат WORD)

Выражение: Var1 = TOWORD (Var2, Var3)

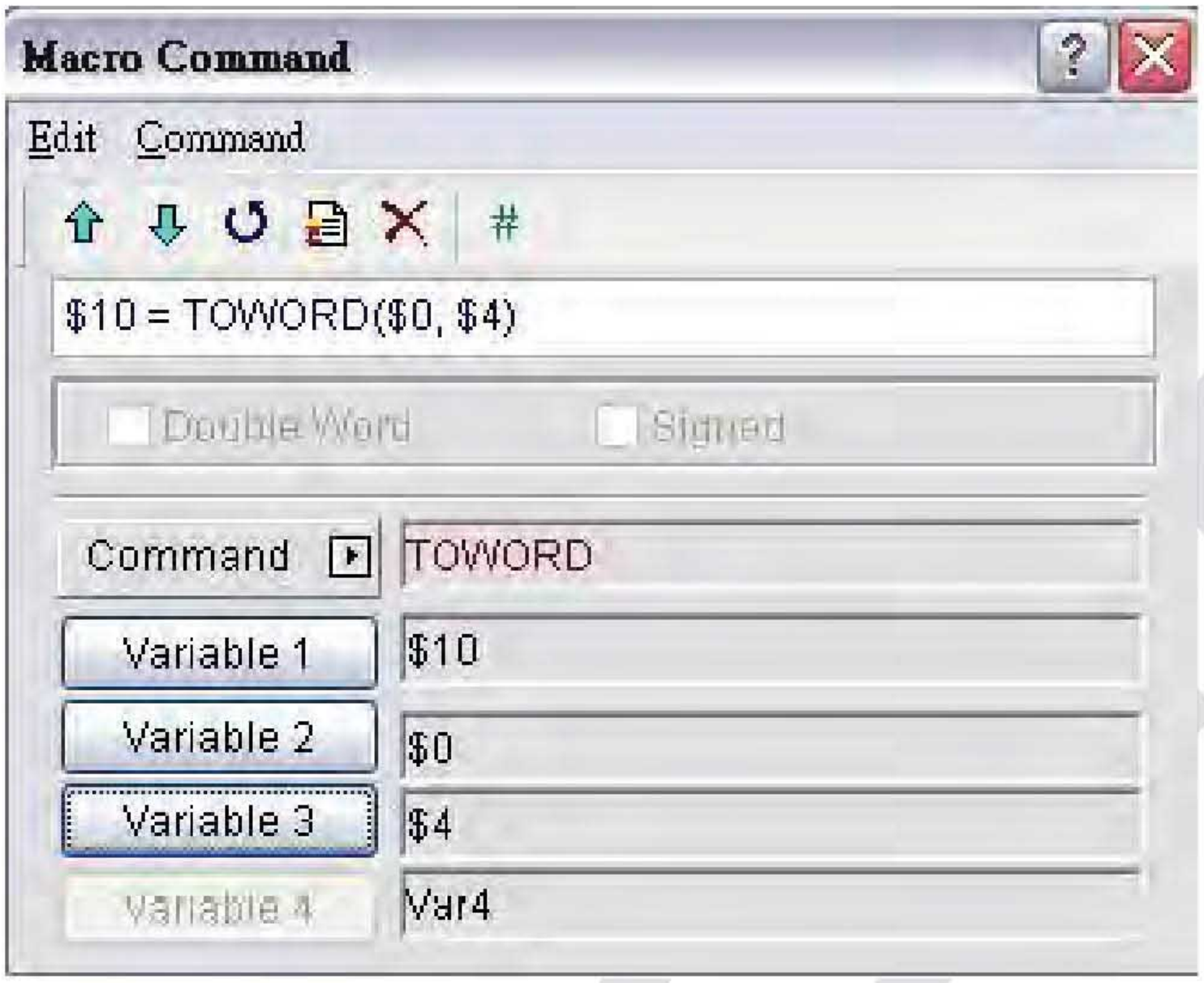
Описание: Содержимое операнда Var2 в формате BYTE преобразуется в формат DWORD и сохраняется в операнде Var1. Преобразовать байты данных (их число задаётся в Var3) операнда Var2 в слова WORD и результат операции сохранить Var1.

Замечания:

- Если Var 3=0, то будет преобразовываться только старший байт.
- Результат сохраняется как положительное число в формате WORD.
- Операнды Var1 и Var2 могут быть только из внутренней памяти, Var3 может быть из внутренней памяти или константой.
- Формат Var2 - WORD. Каждое слово WORD операнда Var2.
- После завершения преобразования данных, старший и младший биты преобразованного Слова поменяются местами.

Пример

Если $0 =12, то 12 BYTES (6 WORDS) будут преобразованы в 12 слов формата WORD, начиная с регистра $4, и результат будет сохранен в регистрах с $10 по $12 (Операция выполняется над 16-ю битными числами без знака).



- **TOBYTE** (Преобразование данных в формате WORD в формат BYTE)

Выражение: Var1 = TOWORD (Var2, Var3)

Описание: Преобразовать слова данных (их число задаётся в Var3) операнда Var2 в формат BYTE и результат операции сохранить в Var1. Старшие байты Var2 будут отброшены.

Замечания:

- Результат будет сохранён в формате положительного числа WORD.
- Операнды Var1 и Var2 могут храниться только во внутренней памяти, операнд Var3 может быть храниться во внутренней памяти или в виде константы.
- После завершения преобразования данных, старший и младший биты преобразованного Слова поменяются местами.

Пример

Если $0 =12, то 12 слов формата WORDS, начиная с младшего байта числа, хранящегося в регистре $4 будут преобразованы в 12 BYTES (6 WORDS), а результат сохранится в регистрах с $10 по $16. (Эта операция производится над положительными 16-ю разрядными числами).

■ **SWAP** (Перестановка байт в регистре)

Выражение: SWAP (Var1, Var2, Var3)

Описание: Старшие и младшие байты слов (число слов определяется операндом Var3) начиная с адреса, определяемого операндом Var2, и сохраняются в памяти, начиная с адреса, определяемого операндом Var1.

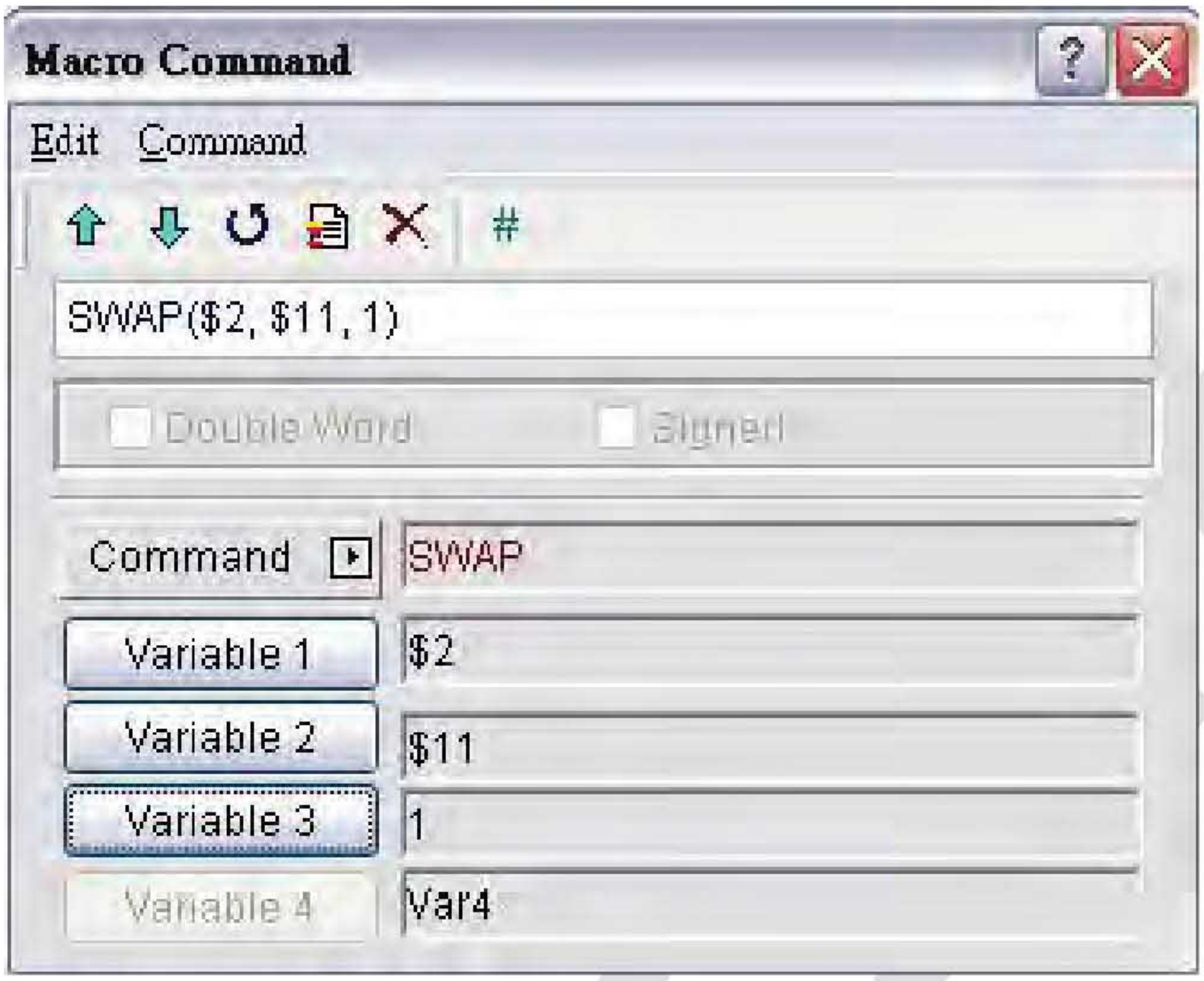
Замечания:

- Результат вычислений будет сохранён виде положительного числа в формате WORD.
- Операнды Var1 и Var2 могут храниться только во внутренней памяти., операнд Var3 может быть храниться во внутренней памяти или в виде константы.

Пример

Поменять местами байты в регистре $11 и результат сохранитить в регистре $2 (Операция выполняется над 16-ю битными числами без знака).

Если $11 = 1234H, то после выполнения команды SWAP $2 = 3412H.



- **XCHG** (Обмен данными)

Выражение: XCHG (Var1, Var2, Var3)

Описание: Обмен значениями операндов (их число определено в Var3), начиная с адреса Var2 и операндов, начиная с адреса Var1.

Замечания:

- Результат вычислений будет сохранён виде положительного числа в формате WORD.
- Операнды Var1 и Var2 могут храниться только во внутренней памяти, операнд Var3 может быть храниться во внутренней памяти или быть константой.

Пример

Произвести обмен содержимым регистров $11 и $2. (Операция с положительными 16-разрядными числами).

Если $11 = 1234H и $2 = 5678H, то после выполнения команды XCHG

$2 = 1234H и $1 = 5678H

- **MAX** (Выделить максимальное значение из значений данных, хранящихся в регистрах)

Выражение: Var1 = MAX (Var2, Var3)

Описание: Выделить максимальное значение содержимого регистров, определяемых Var2 и Var3 и сохранить результат в Var1.

Замечания:

- Результат может быть сохранён виде положительного или отрицательного числа в формате WORD или DWORD.
- Операнд Var1 могут храниться только во внутренней памяти, операнды Var2 и Var3 могут храниться во внутренней памяти или в виде константы.

Пример

Выделить максимальное значение из значений данных регистров $0 и $1, и сохранить результат в регистре $2 (Операция выполняется над 16-ю битными числами без знака).

Если $0 = 2 и$1 = 10, то после выполнения команды MAX $2 = 10 .

- **MIN** (Выделить минимальное значение из значений данных, хранящихся в регистрах)

Выражение: Var1 = MIN (Var2, Var3)

Описание: Выделить минимальное значение содержимого регистров, определяемых Var2 и Var3 и сохранить результат в Var1.

Замечания:

- Результат может быть сохранён виде положительного или отрицательного числа в формате WORD или DWORD.
- Операнд Var1 могут храниться только во внутренней памяти, операнды Var2 и Var3 могут храниться во внутренней памяти или быть константой.

Пример

Выделить минимальное значение из значений данных регистров $0 и $1, и сохранить результат в регистре $2 (Операция выполняется над 16-ю битными числами без знака).

Если $0 = 2 и $1 = 10, то после выполнения команды MIN $2 = 2.

- **TOHEX** (Преобразование кода ASCII (4 слова) в 4-х разрядное целое шестнадцатиричное число.)

Выражение: Var1 = TOHEX (Var2)

Описание: Преобразовать код ASCII операнда Var2 и следующие 3 слова (4 WORD) в шестнадцатиричное значение и сохранить результат в Var1.

Замечания:

- Результат вычислений может быть сохранён виде положительного числа WORD.
- Операнды Var1 и Var2 могут храниться только во внутренней памяти.

Пример

Преобразовать код ASCII регистра $0 и следующих 3-х слов (4 WORD) to a hex value в шестнадцатиричное значение и сохранить результат в $10 (Операция выполняется над 16-ю битными числами без знака).

Если

$0 = 0034H (ASCII 4),

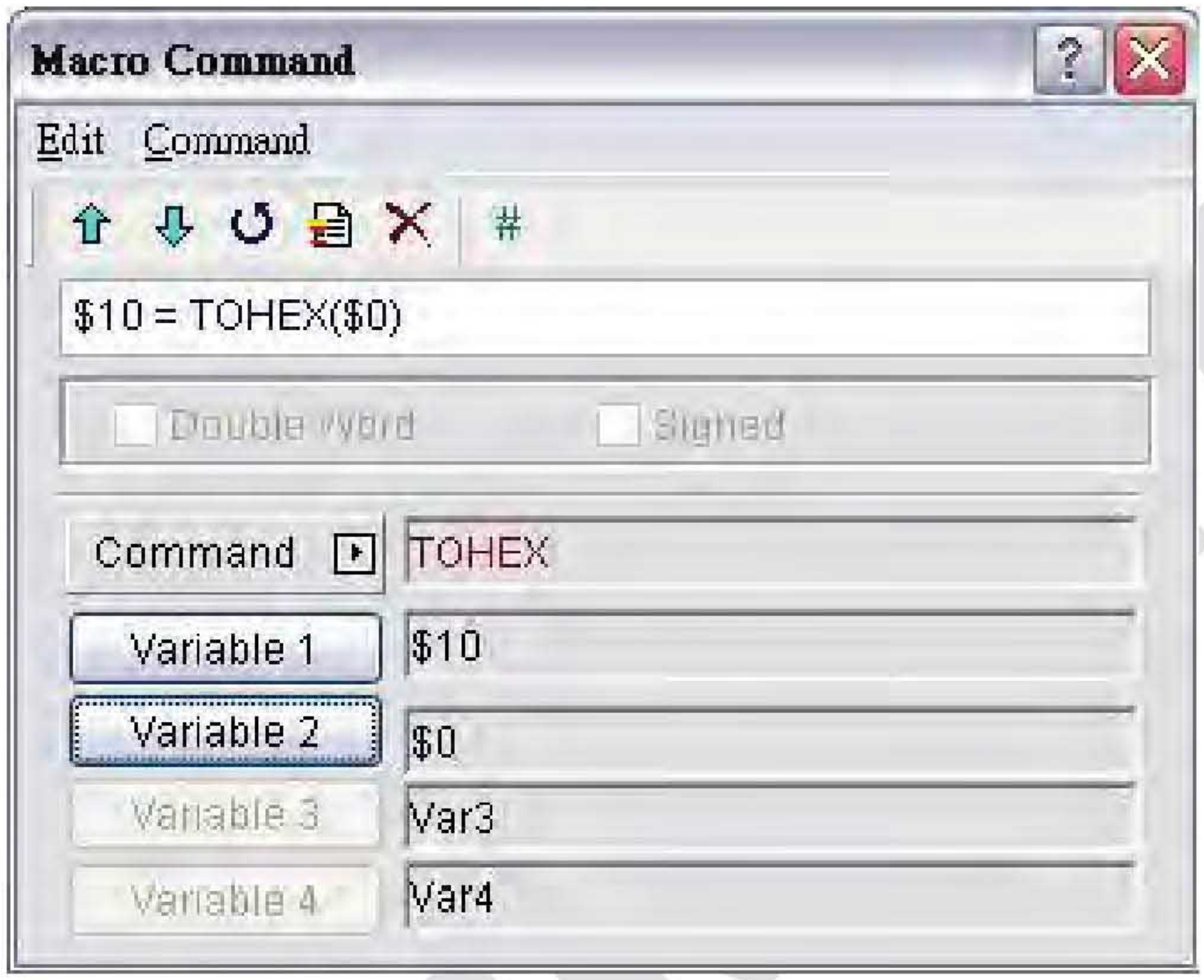
$1 = 0033H (ASCII 3),

$2 = 0036H (ASCII 6),

$3 = 0038H (ASCII 8),

$10 = TOHEX ($0),

то после выполнения команды TOHEX, $10 = 4368H.



- **TOASC** (Преобразование 4-х разрядного целого шестнадцатиричного числа (Var1) в ASCII (4 слова), начиная с адреса Var1.)

  Выражение: Var1 = TOASC (Var2)

  Описание: Преобразование Var2 (1 WORD в шестнадцатиричном значении) в код ASCII (4 WORDS) и сохранить результат в Var1.

  Замечания:

  - Результат хранится виде положительного числа в формате WORD.
  - Операнды Var1 и Var2 могут храниться только во внутренней памяти

  Пример

  Преобразовать данныерегистра $0 (1 WORD в шестнадцатиричном значении) в код ASCII (4 WORDS) и сохранить результат в $10 (Операция выполняется над 16-ю битными числами без знака).

  Если $0 = 1234H и $10 = TOASC ($0), то после выполнения команды TOASC

  $10=0031H (ASCII 1)

  $11=0032H (ASCII 2)

  $12=0033H (ASCII 3)

$13=0034H (ASCII 4)

■ **FCNV** (Преобразование целого числа в формат с плавающей точкой)

Выражение: Var1 = FCNV (Var2)

Описание: Преобразовать значение операнда Var2 (целое число) в формат числа с плавющей точкой и сохранить результат в Var1.

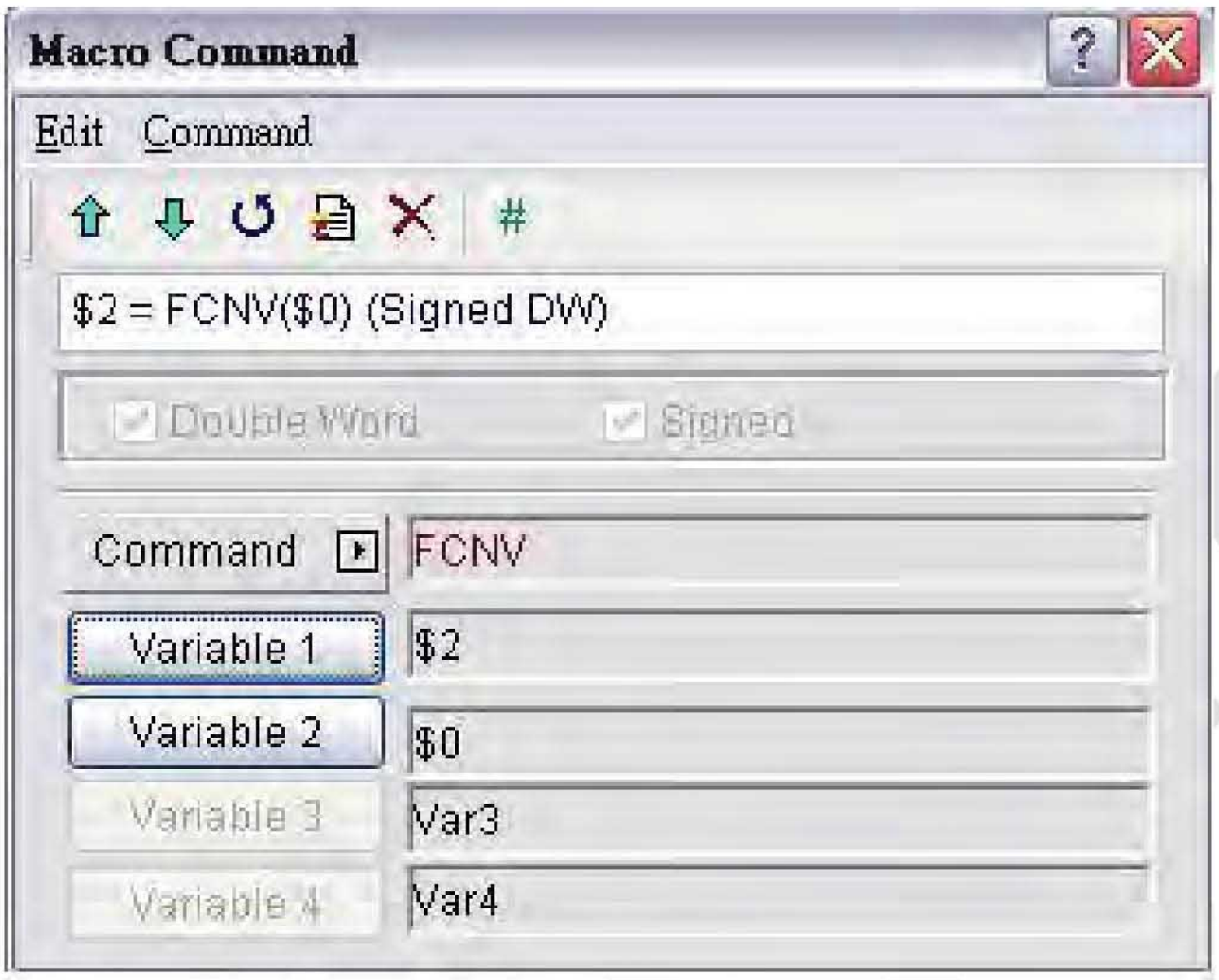
Замечания:

- Результат сохраняется виде числа со знаком в формате DWORD.
- Операнд Var1 может храниться только во внутренней памяти, операнд Var2 может храниться во внутренней памяти или в виде константы.

Пример

Преобразовать целое чило в регистре $0 в число с запятой и сохранить в регистре $2 (Операция выполняется над 16-ю битными числами без знака).

Если $0 = 100, то после выполнения команды FCNV $2 = 100.0



■ **ICNV** (Преобразование числа в формате с плавающей точкой в целое число.)

Выражение: Var1 = ICNV (Var2)

Описание: Преобразование числа в формате с плавающей точкой операнда Var2 в целое число и сохранить результат в Var1.

Замечания:

- Результат сохраняется виде числа со знаком в формате DWORD.
- Операнд Var1 может храниться только во внутренней памяти, операнд Var2 может храниться во внутренней памяти или быть константой.

Пример

Преобразовать число с запятой в регистре $0 в целое чило и сохранить в регистре $2. (Операция выполняется над 16-ю битными числами без знака). Если $0 = 100.0, то после выполнения команды ICNV $2 = 100.

3.14.3.5 Операции сравнения

В этом разделе описаны операции сравнения.

- **IF…THEN GOTO …**

  Выражение: IF *выражение* THEN GOTO LABEL *идентификатор(метка)*

  Описание: Если условия сравнения выполняются, то происходит выполнение команд, по адресу, задаваемому номером метки.

  Замечания:

  - Перечень команд, где проверяется выполнение условий, приведён в таблице.

  Пример

  Если содержимое регистра $2 больше или равно 10, то выполняются команды, определяемые меткой LABEL 1.

Выражение: IFB *выражение* THEN GOTO LABEL *идентификатор(метка)*

Описание: Если условия сравнения состояния битового операнда выполняются, то происходит выполнение команд, находящихся по адресу, задаваемому меткой LABEL (для регистра контроллера).

Замечания:

- Перечень команд, где проверяется выполнение условий, приведён в таблице.

Пример

Если бит контроллера 1@M0 находится в состоянии ON, то происходит выполнение команд находящихся по адресу, задаваемому идентификатором (меткой LABEL 1) .

Выражение: IF *выражение* THEN CALL *подпрограмма*

Описание: Если условия сравнения выполняются, то выполняется подпрограмма 1.

Замечания:

- Перечень команд, где проверяется выполнение условий, приведён в таблице.

Пример

Если содержимое регистра $2 = 10, то выполняется подпрограмма 1.

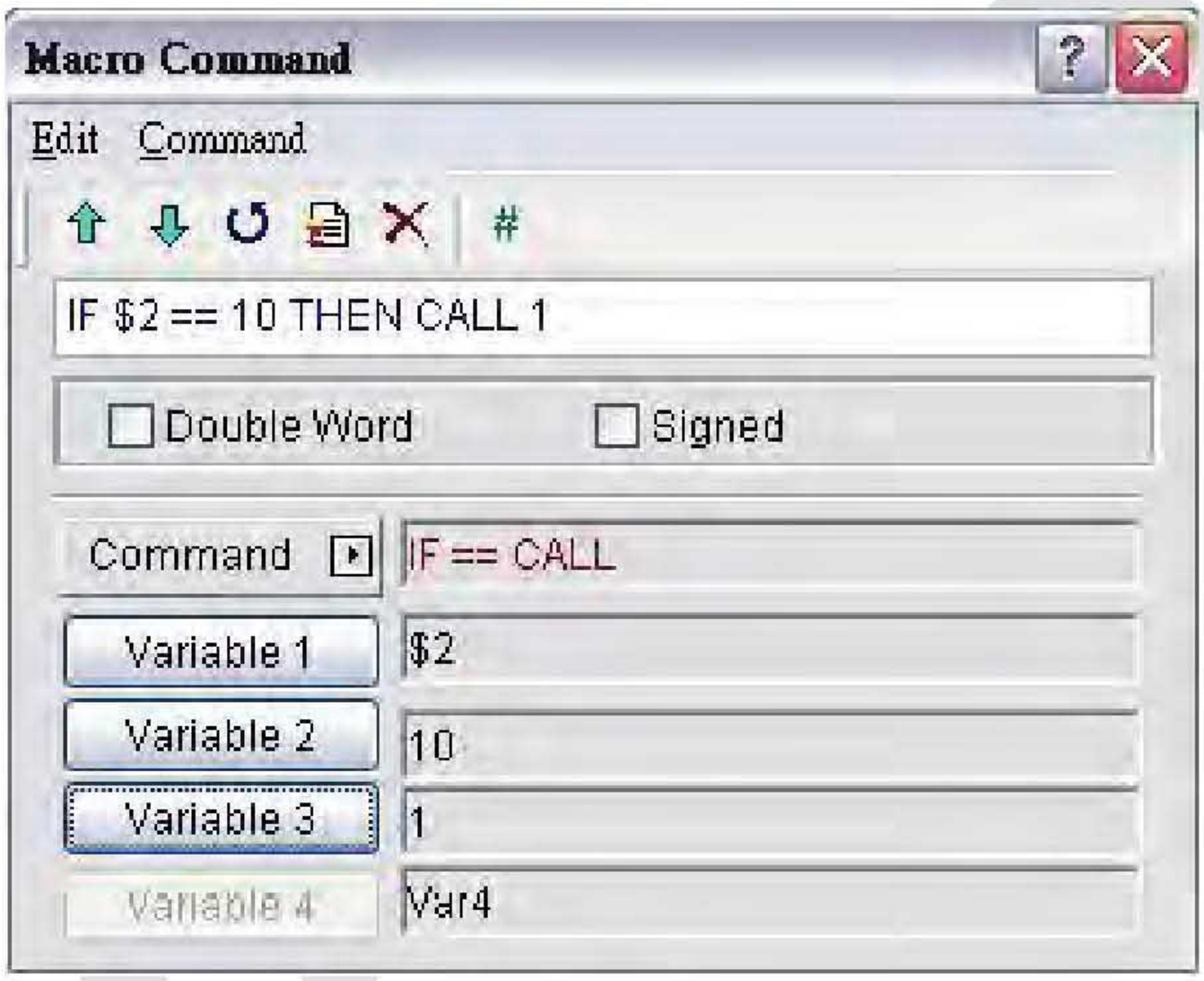


Таблица 3-14-3 Таблица команд сравнения

Выражение	Описание	Примечания
Var1 == Var2	Var1 равно Var2	Var1 и Var2 могут быть регистрами внутренней памяти или константами
Var1 != Var2	Var1 не равно to Var2	
Var1 > Var2	Var1 больше чем Var2	
Var1 >= Var2	Var1 равно или больше Var2	
Var1 < Var2	Var1 меньше чем Var2	
Var1<= Var2	Var1 равно или меньше чем Var2	
Var1 && Var2 == 0	Выполнить команду логическое «И» над операндами Var1 и Var2 и результат равен 0	
Var1 && Var2 != 0	Выполнить команду логическое «И» над операндами Var1 и Var2 и результат не равен нулю	

Выражение	Описание	Примечания
Var1== ON	Битовая перемення Var1 =ON	Var1 битовая переменная памяти контроллера или внутренней памяти панели
Var1== OFF	Битовая перемення Var1 =OFF	

■ IF…ELSEIF…ELSE…ENDIF

Выражение:

IF *условие 1*

Действие 1

ELSEIF *условие 2*

Действие 2

ELSE

Действие 3

ENDIF

Описание: Условный переход с несколькими условиями. При выполнении одного условия выполняется действие 1, при его невыполнении –проверяется выполнение условия 2. При выполнении условия 2 действие 2 выполняется. Иначе, при невыполнении условий 1 и 2 будет выполнено действие 3

ENDIF обязательно завершает последовательность команд сравнения IF.

Замечания:

- Перечень команд, где проверяется выполнение условий, приведён в таблице.

Пример

IF $0 == 0
$10 = 0
ELSEIF $0 == 1
$10 = 1
ELSE
$10 = 2
ENDIF
	IF $0 == 0
	$10 = 0
	ELSEIF $0 ==1
	$10 = 1
	ELSE
	$10 = 2
	ENDIF
		Если $0 = 0,
		То $10 = 0;
		Если $0 = 1, но не равно 0,
		то $10 = 1;
		Если $0 не равно ни 0, ни 1,
		то $10 = 2.

■ **FCMP** (Команда сравнения операндов с плавающей запятой)

Выражение: Var1 = FCMP (Var2, Var3)

Описание: Сравнить значение операндов Var2 и Var3 и результат сохранить в Var1 в формате DWORD со знаком.

Виды сравнения включают:

Var1=0, если Var2 = Var3

Var1=1, если Var2 > Var3

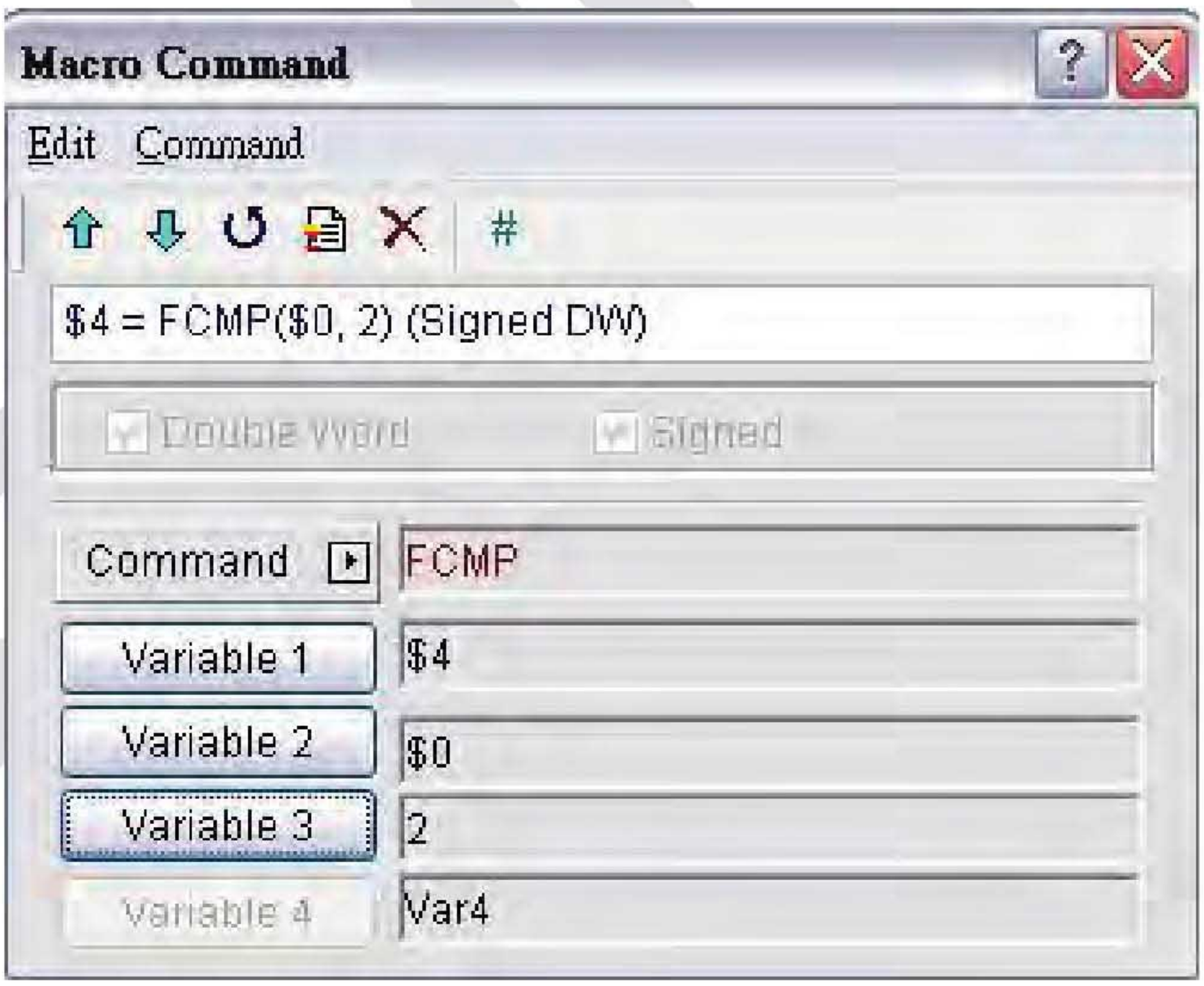
Var1=2, если Var2 < Var3

Замечания:

- The calculation result can be stored as signed DWORD.
- Операнд Var1 может храниться только во внутренней памяти, Var2 и Var3 – во внутренней памяти или быть константой.

Пример

Сравнить содержимое регистров $0 и $2, результат сравнения сохранить в регистре $4. ((this is an operation of signed 32-bit data).



3.14.3.6 Команды управления потоком (Flow Control)

GOTO
LABEL
CALL
RET
FOR
NEXT
END

В данном разделе приводится описание различных типов команд управления потоком.

- **GOTO** (Безусловный переход к заданной метке Label). Команда GOTO всегда осуществляет безусловный переход к метке Label Var1)

  Выражение: GOTO LABEL Var1

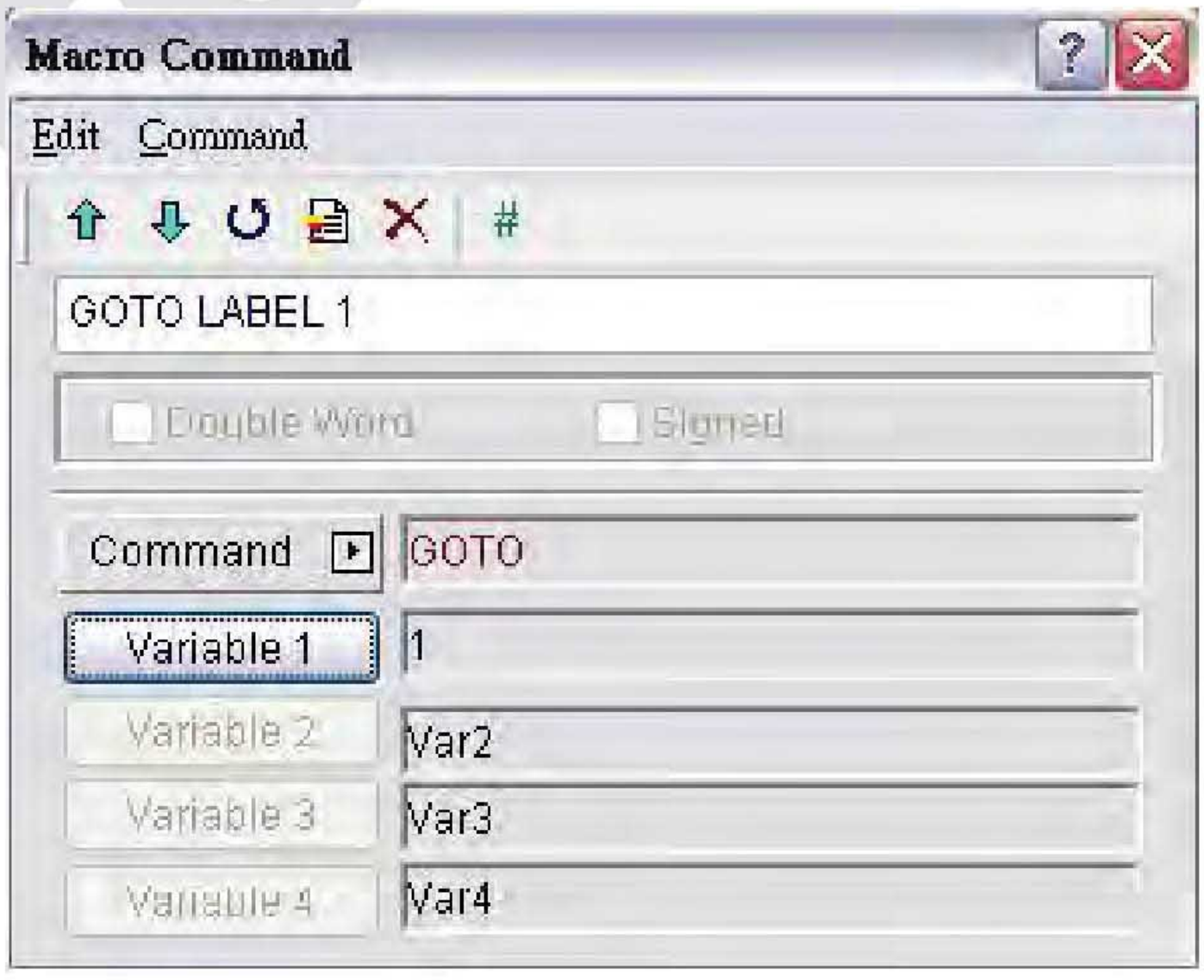
  Описание: Переход к метке Label Var1 без условий.

  Замечания:

  - Операнда Var1 может быть только константой.

Пример

Перейти по адресу, определяемому меткой Label 1 и без проверки каких либо условий выполнять программу.



■ **LABEL** (Такая же метка, как Label Var1)

Выражение: LABEL Var1

Описание: Операция GOTO сдвигает выполнение программы на метку LABEL без условий.

Замечания:

- Операнд Var1 может быть только константой.
- Прользователь может использоватьметку LABEL 1 только один раз в одном макросе LABEL 1

Пример

LABEL 1.

■ **CALL..RET** (вызов подпрограммы)

Выражение: CALL Var1…RET

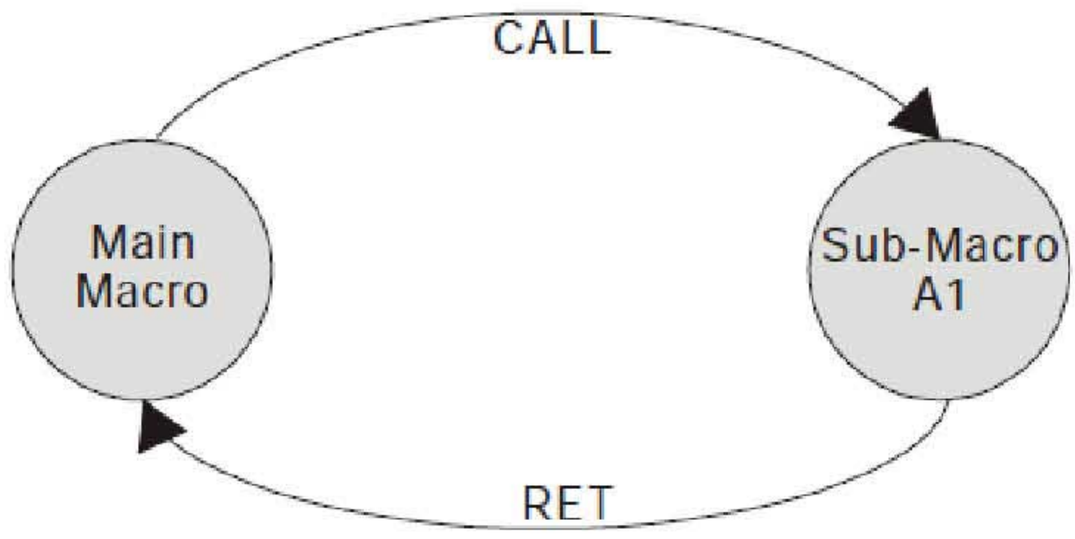
Описание: “CALL Var1” означает вызвать подпрограмму Var1. Var1 определяет номер подпрограммы. “RET” означает выход из подпрограммы и возврат в основную программу Var1. Размещается в конце подпрограммы Var1 .

Замечания:

- Номер подпрограммы лежит в диапазоне 001 – 512.

- Операнд Var1 должен быть константой или регистром внутренней памяти.

Пример



При выполнении макрокоманды CALL Var1 управление программой передается в подпрограмму.

После выполнения и завершения подпрограммы, посредством команды RET управление программой передаётся в основную программу. Номер подпрограммы может лежать в диапазоне 1 … 512.

Пользователь может присвоить имя подпрограмме произвольно. Из подпрограммы пользователь может вызвать другую подпрограмму, но глубина вложений не должна быть более 6.

■ **FOR…NEXT** (Программный цикл)

Выражение: FOR Var1 Действие NEXT

Описание: Для вложенных циклов команда ”FOR” определяет начало цикла, “NEXT” - конец.

Когда эта команда выполняется, последовательно выполняются все команды, причём операнд Var1 определяет число повторений. Внутри цикла можно изменить число повторений, но изменения вступят в силу только при следующем повторении цикла.

Замечания:

- Число вложенных циклов не более 3.
- Операнд Var1 должен быть константой или регистром внутренней памяти
- Вычисление может содержать группу макрокоманд и находиться внутри вложенного цикла.

Пример

```
FOR 10
$0 = $0 + 1
NEXT
```

Если $10=10, $0=0, то после выполнения программы $0 = 10

Если вставить $10 = 2 между FOR и NEXT в макросе, показанном выше, цикл FOR … NEXT будет работать 10 раз, даже учитывая, что значение регистра $10 было изменено на 2 во время первого цикла.

- **END** (Конец макрокоманды)

Выражение: *Действие 1* END *Действие 2*

Описание: Команда END используется для завершения макро программы, все действия после END не выполняются. Если команда END в главной программе, то производится возврат на выполнение 1 строки. Если эта команда в подпрограмме, то выполнение подпрограммы завершится и выполняться будет предыдущая подпрограмма.

Замечания:

- Вычисление может содержать группу макрокоманд и находиться внутри вложенного цикла.
- Если в подпрограмме имеется команда END, то подпрограмма здесь и завершается. Если пользователь хочет вернуться в предыдущую подпрограмму, то необходимо использовать команду RET.

Пример

```
$1 = 10
$1 = $1 + 1
END
$1 = $1 + 1
```

После выполнения команды результатом будет $1 = 11, а не $1 = 12, так как команда END завершает выполнение макропрограммы.

3.14.3.7 Установка состояния битов.

В разделе подробно описаны типы команд, выполняющие задание состояния битовых операндов.

- **BITON** (Установить бит в состояние ON)

Выражение: BITON Var1

Описание: Данная команда устанавливает операнд Var1 в состояние ON.

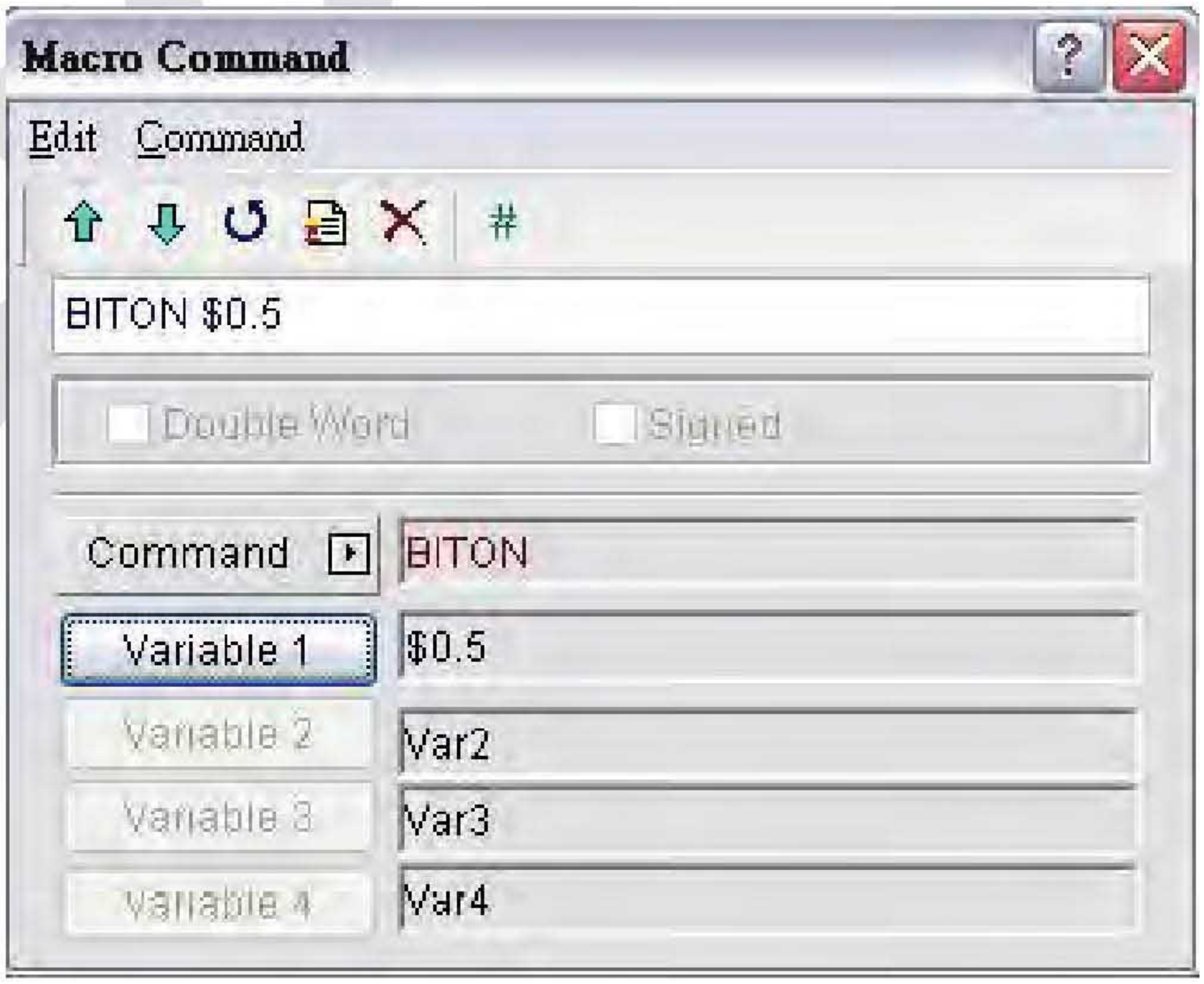
Замечания:

- Операнд Var1 долже быть адресом контроллера или внутренней памяти (BIT).

Пример

Установить 5-ый бит регистра $0 в состояние ON.

Если $0 = 0000000000000000, то после выполнения операции $0 = 0000000000010000



■ **BITOFF** (Установить бит в состояние OFF)

Выражение: BITOFF Var1

Описание: Данная команда устанавливает операнд Var1 в состояние OFF.

Замечания:

- Операнд Var1 должен иметь адрес контроллера или внутренней памяти (BIT).

Пример

Установить 5-ый бит регистра $0 в состояние OFF.

Если $0 = 1111111111111111, то после выполнения операции

$0 = 1111111111101111

■ **BITNOT** (Инвертирование состояния бита (ON в OFF, OFF в ON))

Выражение: BITNOT Var1

Описание: Данная командаинвертирует состояние определенного бита. ON в OFF, OFF в ON.

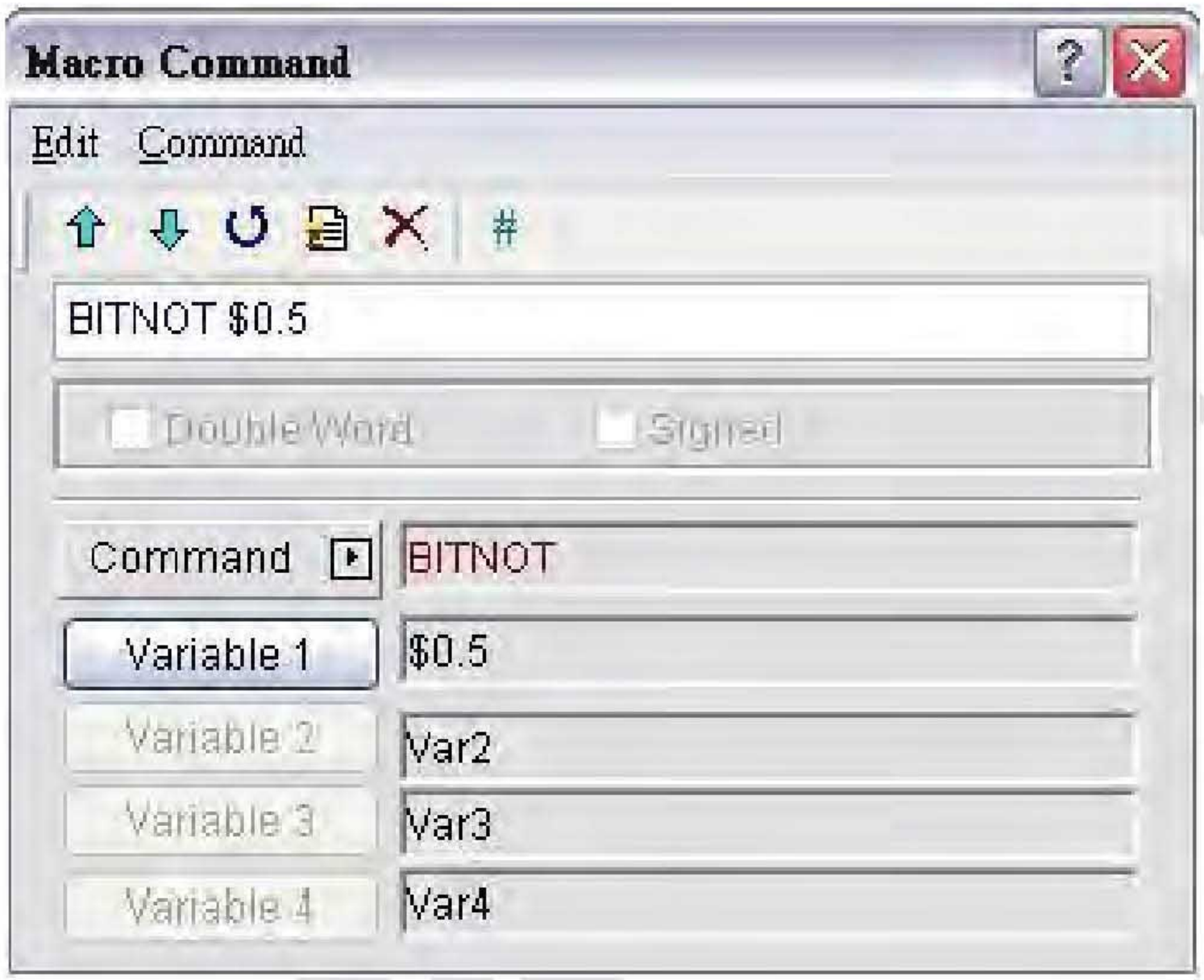
Замечания:

- Операнд Var1 должен иметь адрес контроллера или внутренней памяти (BIT).

Пример

Инвертирование состояния 5-ого бита регистра внутренней памяти $0.

Если $0 = 1111111111111111, то после выполнения операции
$0 = 1111111111101111



- **GETB** (Считать состояние бита)

  Выражение: Var1 = GETB Var2

  Описание: Эта команда считывает состояние битового операнда Var2 и сохраняет его в операнде Var1.

  Замечания:

  - Операнды Var1 и Var2 должны быть адресом контроллера или внутренней памяти (BIT).

  Пример

  Считать состояние 5-ого бита регистра внутренней памяти $0 и сохранить его в 5-ом бите регистра внутренней памяти $1.

  Если $0 = 1111111111111111 и $1 = 0000000000000000, то после выполнения этой операции $1= 0000000000010000

3.14.3.8 Команды коммуникации

В разделе подробно описаны типы команд, выполняющие управление коммуникациями с внешними устройствами.

■ INITCOM (Настройка COM-порта)

Выражение: Var1 = INITCOM (Var2)

Описание: Эта команда производит начальную настройку COM порта и установку коммникационного протокола. После того, как пользователь настроит каждый порт команда запустит процесс коммуникации и сохранит настройки в операнде Var1.

Замечания:

- Var1: Результат обмена данными:
  1: Закончен нормально
  0: Обрыв соединения
- Для настройки операнда Var2, ознакомьтесь с настройками коммуникации Var2 в INITCOM (Коммуникационный протокол).

Настройки Var2 (Communication Protocol)

Элемент	Тип	Код	Замечания
Выбор COM-порта	COM1	0	
	COM2	1	
	COM3	2	
Вид интерфейса	RS232	0	
	RS422	1	
	RS485	2	
Число бит в слове	7 Bits	0	
	8 Bits	1	
Проверка чётности	None	0	
	Odd	1	
	Even	2	
Наличие стоповых бит	1 Bits	0	
	2 Bits	1	

Скорость обмена	300	0	
	600	1	
	900	2	
	1200	3	
	2400	4	
	4800	5	
	9600	6	
	14400	7	
	19200	8	
	28800	9	
	38400	10	
	57600	11	
	115200	12	
Вид управления обменом данными	No Flow Control	0	Ознакомьтесь с пояснениями
	CTS RTS Flow Control	1	
	DTR DSR Flow Control	2	
	Xon Xoff Flow Control	3	

No Flow Control: Функция контроля обмена запрещена.

Flow Control: Скорость обмена и достоверность передачи увеличиваются при обмене, благодаря новым ментодам сжатия и кодирования.

Новые технологии даже на повышенных скоростях делают обмен дольше, поэтому, для гарантирования защищённости данных и завершения обмена при передаче данных через последовательный порт, необходимо осуществлять постоянный контроль его проведения.

Для подтверждения завершения обмена пользователь может использовать следующие функции контроля обмена:

CTS / RTS: Аппаратное управление обменом. Используется квитирующий сигнал для управления входящих и исходящих данных. It uses hand-shaking signal to control receiving and sending data. Контроль происходит через внутренний или внешний модем, соединенный с панелью через соединительный кабель.

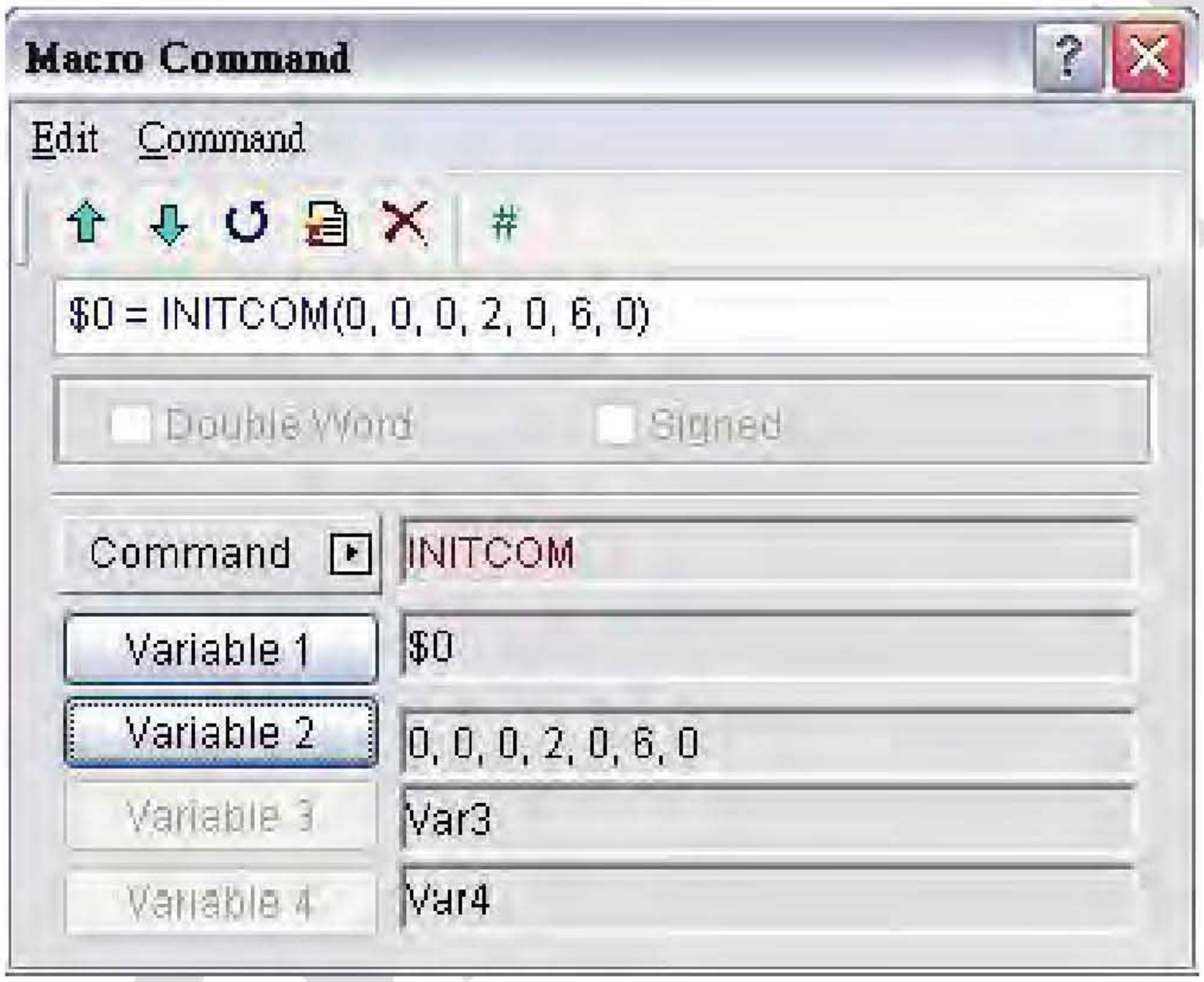
DSR / DTR: Также аппаратное управление обменом. Применяется при обмене данными с компьютером, соединённым кабелем.

XON / XOFF: Программное управление обменом. Применяется только для

модемов со скоростью не более 2400 бит в секунду. Способ управления основан на генерировании контрольного кода и добавлении его к передаваемым данным.

Пример

Когда для порта COM1 установлен протокол RS232, 7, Even, 1, 9600, и режим обмена No Flow Control, результаты обмена сохраняются в регистре $0. На экране это быдет выглядеть следующим образом:



- **ADDSUM** (Вычисление контрольной суммы методом сложения)

Выражение: Var1 = ADDSUM (Var2, Var3)

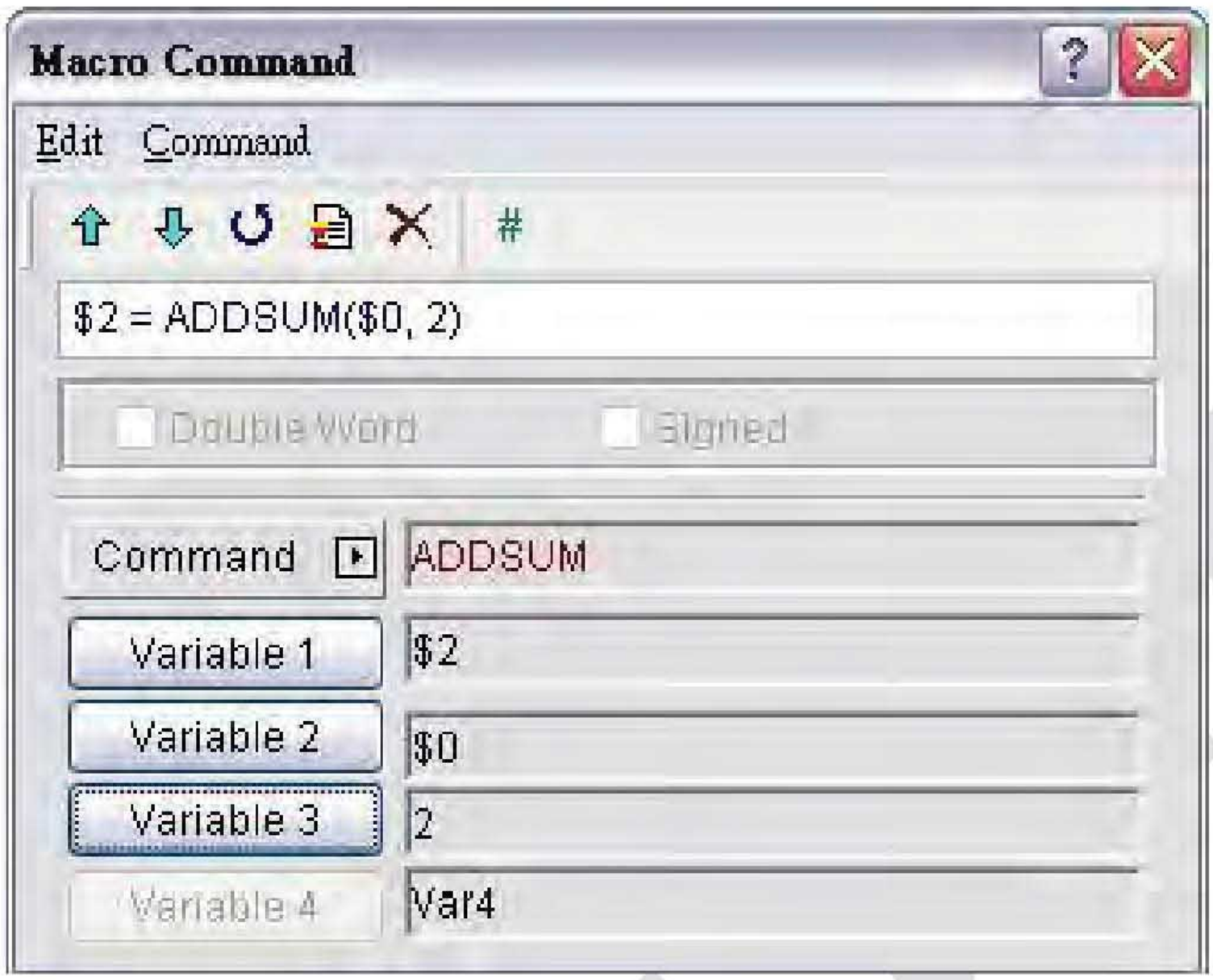
Описание: Эта команда используется для расчёта контрольной суммы методом сложения. Операнд Var1 хранит вычисленное значение, Var2 начальный адрес вычисления, Var3 длину данных.

Замечания:

- Вычисленный результат сохраняется в формате WORD без знака.
- Операнды Var1 и Var2 могут храниться только в внутренней памяти. Var3 быть константой или храниться во внутреней памяти.

Пример

Начальный адрес $0, длина данных равна 2. После передачи данных рассчитанная по методу сложения контрольная сумма сохраняется в регистре $2 (операция над 16-ти битными числами без знака).



■ **XORSUM** (Использование метода XOR для расчёта контрольной суммы)

Выражение: Var1 = XORSUM (Var2, Var3)

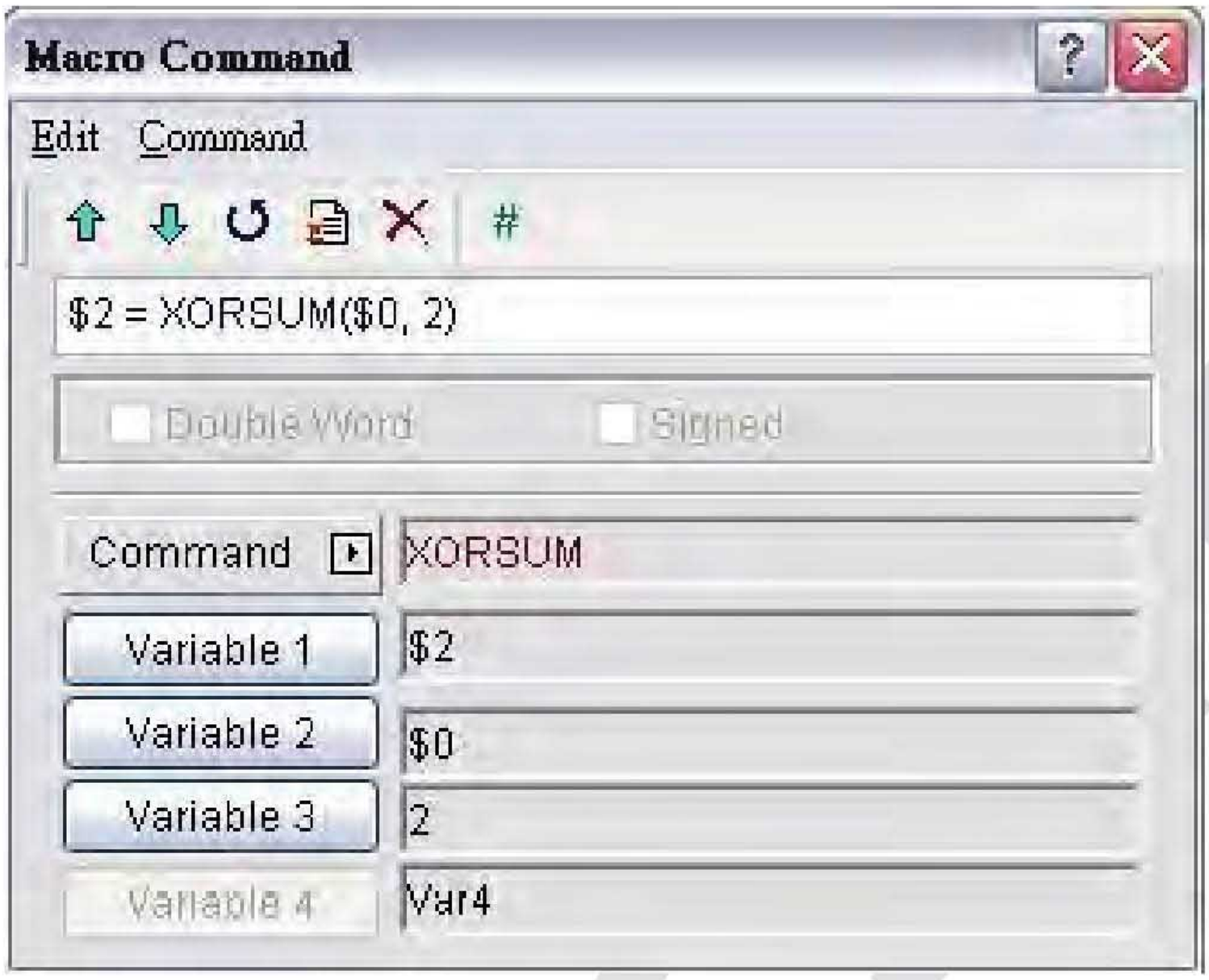
Описание: Эта команда используется для расчёта контрольной суммы методом XOR. Операнд Var1 хранит вычисленное значение, Var2 начальный адрес вычисления, Var3 длину данных

Замечания:

- Вычисленный результат сохраняется в формате WORD без знака.
- Операнды Var1 и Var2 могут храниться только в внутренней памяти. Операнд Var3 быть константой или храниться во внутреней памяти.

Пример

Начальный адрес $0, длина данных равна 2. После передачи данных рассчитанная по методу XOR контрольнаясумма сохраняется в регистре $2 (операция над 16-ти битными числами без знака).



■ **PUTCHARS** (Передача данных в COM-порт)

Выражение: Var1 = PUTCHARS (Var2, Var3, Var4)

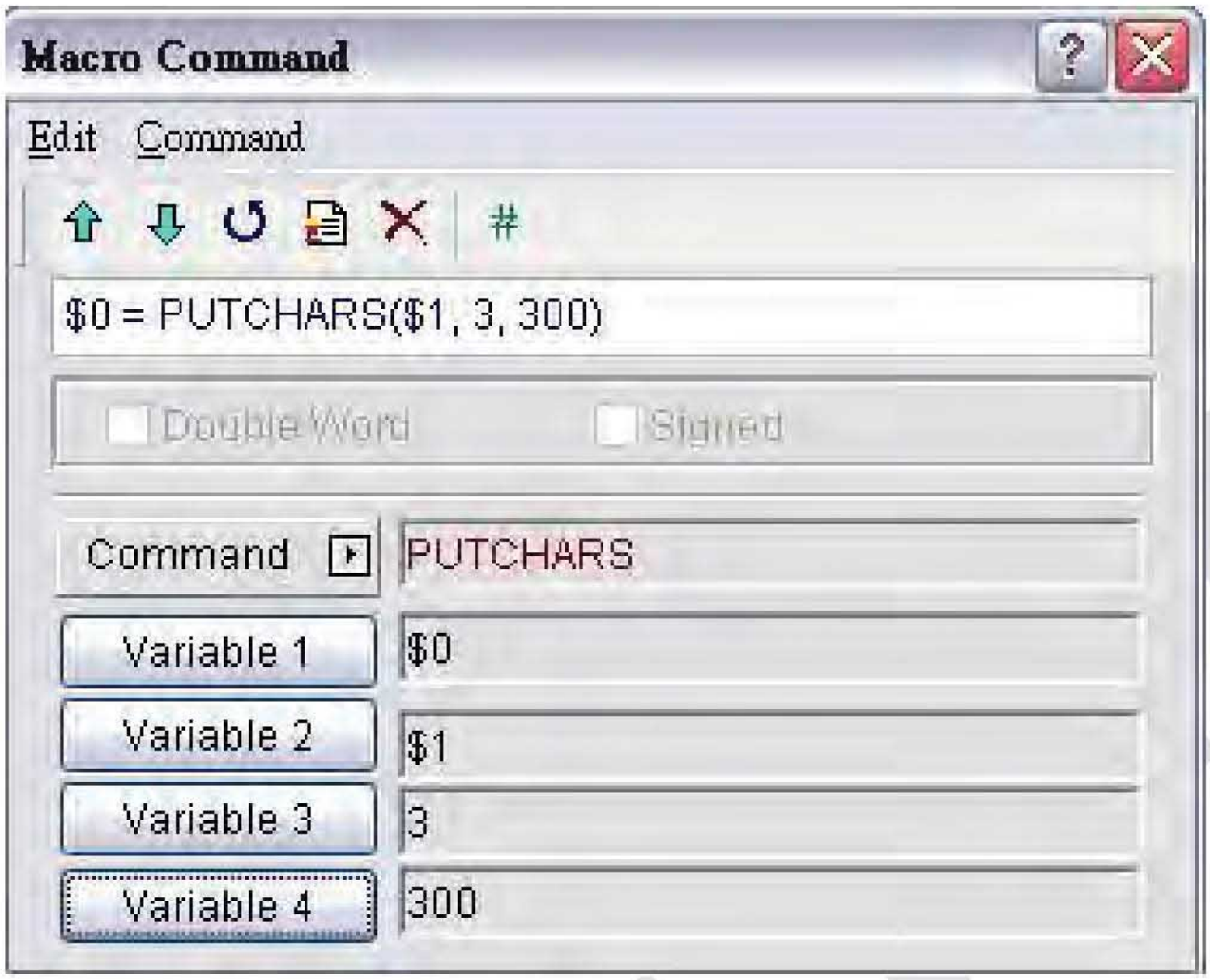
Описание: Передача данных в COM порт. Var1 ответное сообщение после передачи, Var2 стартовый адрес передачи данных, Var3 длина передаваемых данных, Var4 допустимое время передачи (в мс). Результат сохранён в Var 1.

Замечания:

- Var1: Ответное сообщение:
  1: Обмен закончен
  0: Обмен прерван
- Операнды Var1 и Var2 могут храниться только в внутренней памяти. Var3 и Var4 быть константой или храниться во внутреней памяти.

Пример

Передача в порт трёх слов WORD начиная с $1 и сохранение результата в $0.



■ **GETCHARS** (Приём данных из COM-порта

Выражение: Var1 = GETCHARS (Var2, Var3, Var4)

Описание: Приём данных из COM порта.

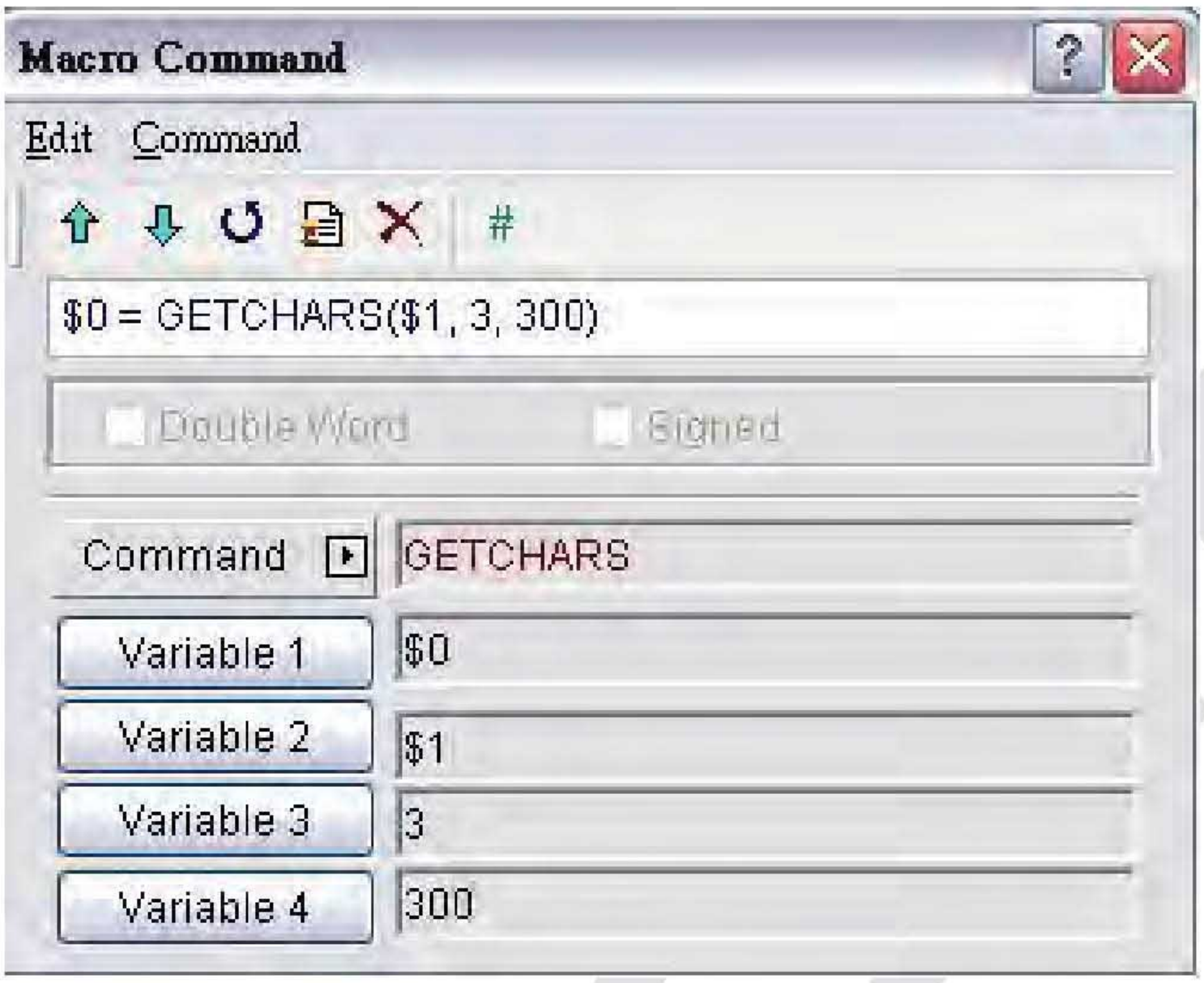
Var1 - ответное сообщение после приёма, Var2 - начальный адрес принимаемых данных, Var3 длина данных и Var4 - допустимое время приёма (в мс). Результат будет сохранён в Var1.

Замечания:

- Var1: Ответное сообщение после приёма
  1: Обмен закончен
  0: Обмен прерван
- Операнды Var1 и Var2 могут храниться только в внутренней памяти. Var3 и Var4 быть константой или храниться во внутреней памяти.

Пример

Получить три последовательных слова начиная с адреса $1 и сохранить их начиная с $0.



- **SELECTCOM** (Выбор COM Port)

Выражение: SELECTCOM (Var1)

Описание: Команда служит для выбора COM порта.

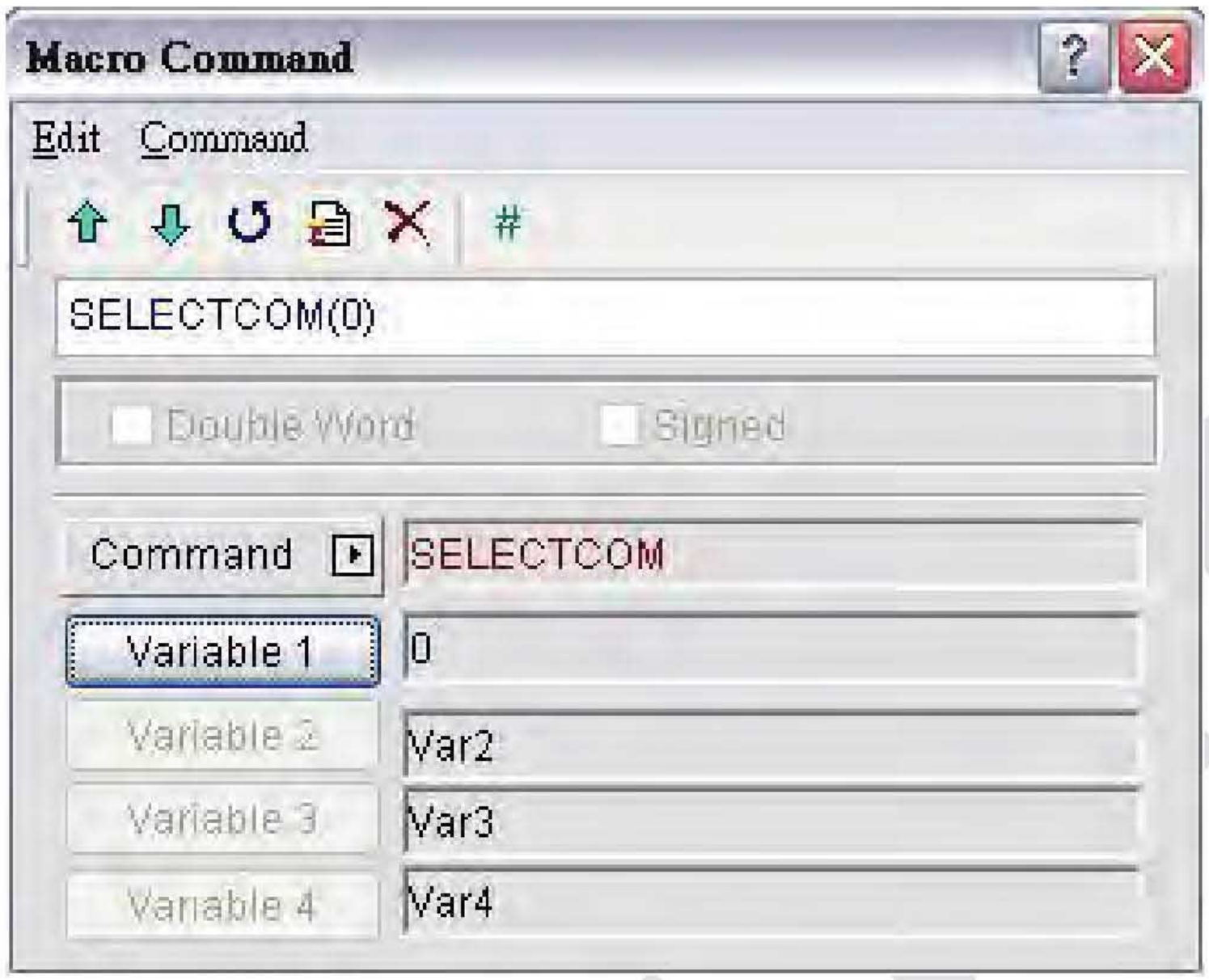
Когда внешний контроллер не установлен (ПЛК выбран как NULL) в меню **Options > Configuration** в программе Screen Editor, пользователь может использовать два COM порта одновременно, установив значение операнда Var1 (0:COM1, 1:COM2, 3:COM3 (для некоторых моделей)

Замечания:

- Var1 – только константа.
- После выполнения данной команды все обращения будут выполняться через установленный ею порт выполнения. Макросы не поддерживают такую команду или выполняются с искажениями.

Пример

Выбор COM1 .



■ **CLEARCOMBUFFER** (Очистка буфера COM-порта)

Выражение: CLEARCOMBUFFER (Var1, Var2)

Описание: Команда обеспечивает очистку буферов приёма или передачи данных.

Var1 – номер COM порта и представляет собой константу 0(COM1), 1(COM2) или 2(COM3).

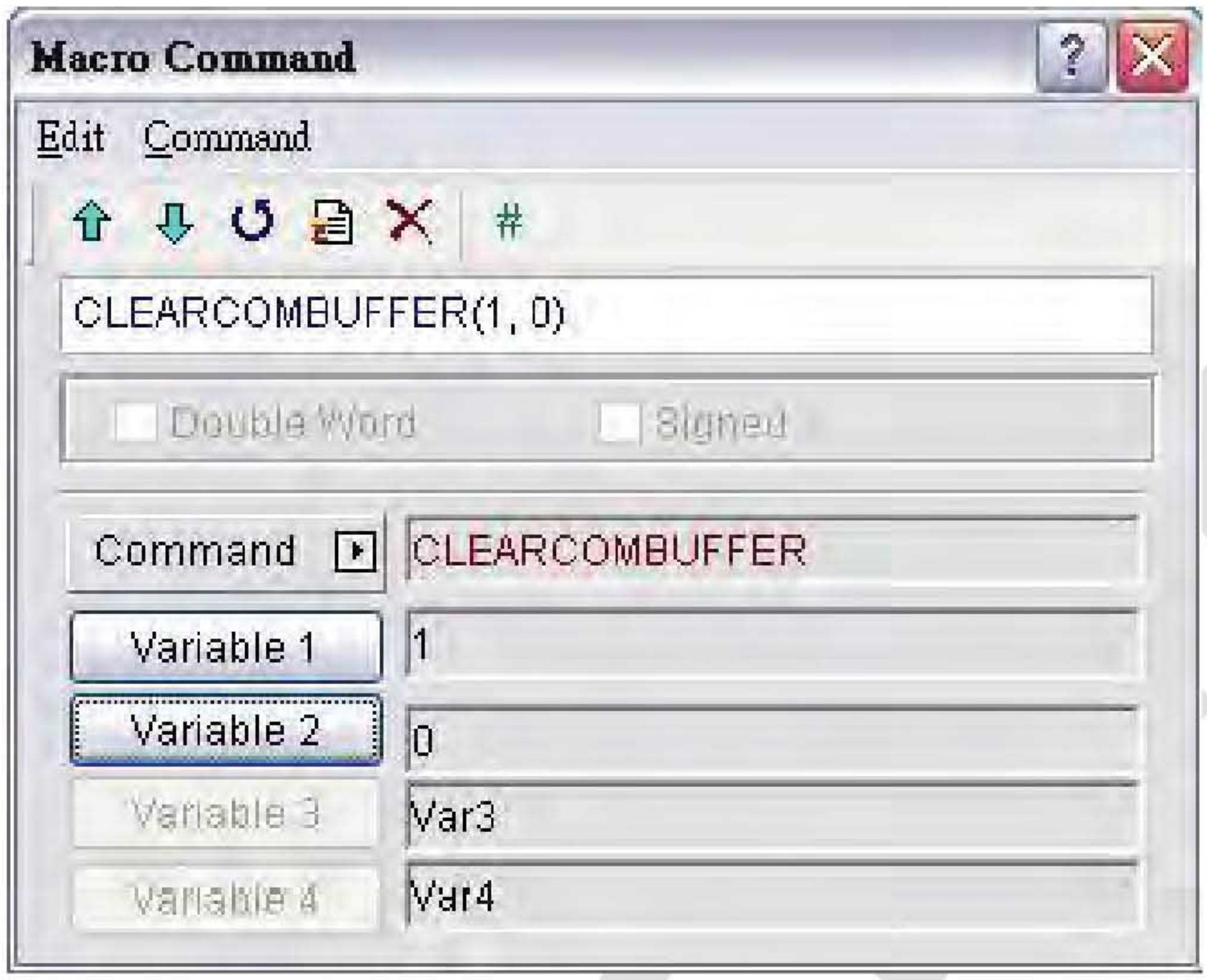
Var2 определяет тип буфера и представляет собой константу 0 (для принимаемых данных) или 1 (для передаваемых данных).

Замечания:

- Операнды Var1 и Var2 должны быть только константами.

Пример

Очистить буфер приёма COM2.



■ **CHRCHKSUM** (Расчёт длины текста или символов и контрольной суммы)

Выражение: Var1 = CHRCHKSUM (“Var2”, Var3, Var4)

Описание: Команда используется для расчёта длины текста или символов и контрольной суммы.

Var1 – адрес внутренней памяти, где сохраняется длина строки текста, который хранится в операнде Var2.

Var2 – строка текста

Var3 – адрес внутренней памяти, где хранится контрольная сумма строки текста Var2.

Var4 – длина данных контрольной суммы, которая хранится в Var3.

1 - означает байт

2 - означает слово

Алгоритм преобразования:

Все текстовые символы конвертируются в коды ASCII и суммируются.

Например, символ ’2’ конвертируется в ASCII ‘31H’, ’4’ в ASCII код‘34H’ и контрольная сумма равна 31Hex + 34Hex = 65H.

Замечания:

- Операнды Var1 и Var3 могут храниться только во внутренней памяти. Операнд Var2 должен быть только строкой текста, а Var4 должен быть

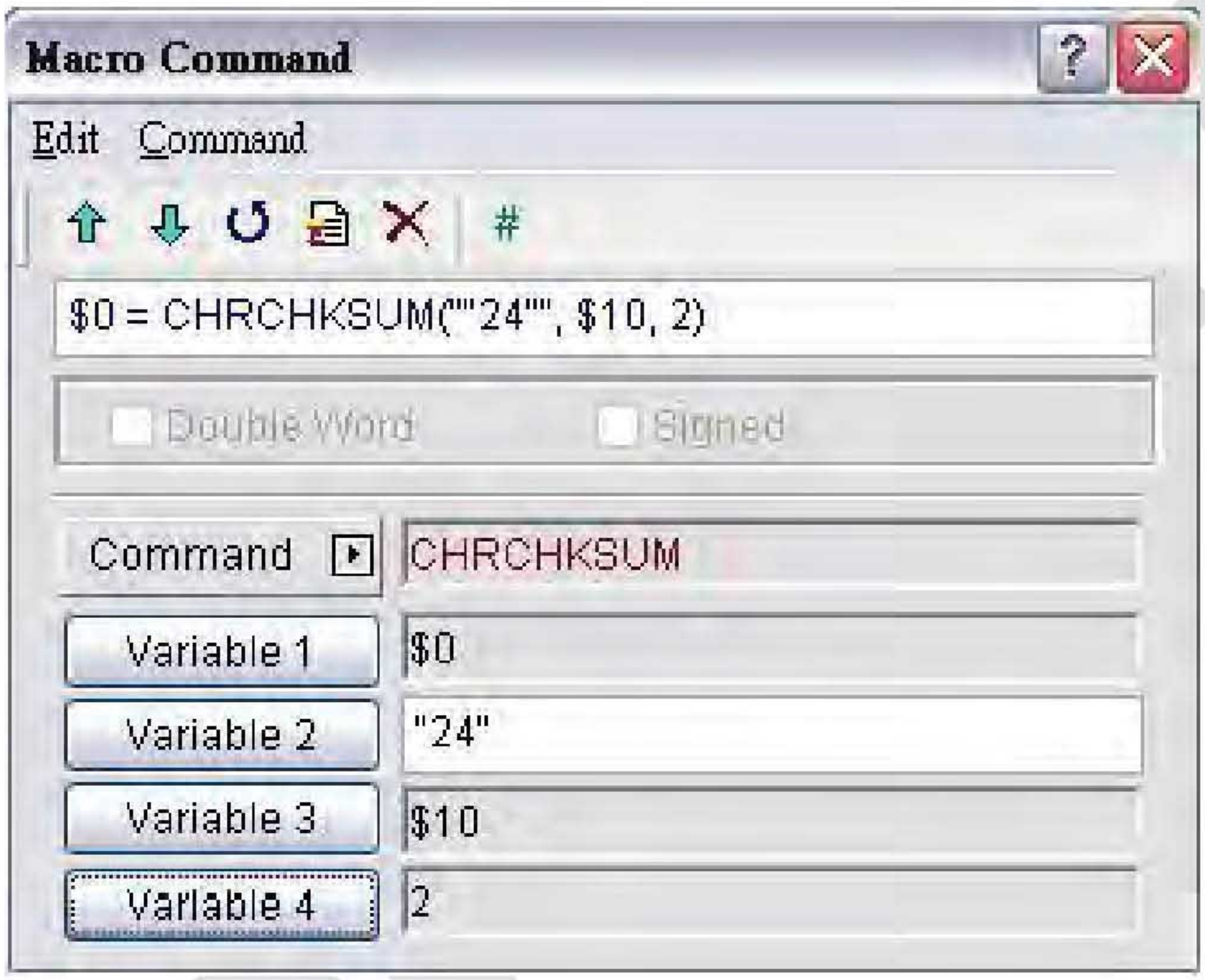
только константой.

Пример

Рассчитать длину данных “24” и контрольной суммы.

$0 = CHRCHKSUM(“24”, $10, 2)

После выполнениякоманды, 4 сохраняется в $0, что соответствует 2 словам. Контрольная сумма сохраняется в $10 как 65H.



- **LOCKCOM / UNLOCKCOM** (Запирание/отпирание COM-порта)

Выражение:

Lock COM Port: Var1 = LOCKCOM (Var2, Var3)

Unlock COM Port: Var1 = UNLOCKCOM (Var2)

Описание: Команда **LOCKCOM** закрывает COM-порт.

Var1 – ответ после коммуникации.

Var2 определяет COM-порт и может принимать значения: 0(COM1), 1(COM2) или 2(COM3).

Var3 – время отклика (в мс). При его достижении, если COM порт не закрылся, то появляется ошибка. При значении Var3=0 панель оператора переходит в режим ожидания без ограничения по времени.

Команда **UNLOCKCOM** открывает COM-порт.

Var2 определяет COM-порт и может принимать значения: 0(COM1), 1(COM2)

или 2(COM3).

Замечания:

- Var1 ответ после коммуникации
  1: Обмен выполнен
  0: Обмен прерван
- Операнд Var1 может храниться только в внутренней памяти. Операнды Var2 и Var3 должны быть только константой.
- При использовании коммуникационных команд в таких макросах, как, Screen Cycle Macro, Clock Macro, Background Macro, Befor/After Execute Macro, ON/OFF Macro, если не применять команды LOCKCOM and UN-LOCKCOM могут возникать ошибки. Для обеспечения устойчивой работы и непрерывного обмена рекомендуется использовать эти две команды.
- Когда операнд Var3 = 0, команда LOCKCOM выполняется дважды и панель оператора будет находиться в режиме ожидания без ограничения по времени. В это время панель для всех становится недоступна.

Пример

```
$0 = LOCKCOM(0, 200)
$1 = PUTCHARS($1, 3, 300)
$2 = GETCHARS($1, 3, 300)
UNLOCKCOM(0)
```

Пояснения к команде LOCKCOM		
Background Macro $0 = LOCKCOM(0,0) $1 =PUTCHARS($1,3,300)	On Macro $0 = LOCKCOM(0,0) $1 = GETCHARS($1,3, 300)	Screen Cycle Macro $0 = LOCKCOM(0,0) $1 = PUTCHARS($1,3,300)
В вышеприведённых подпрограммах используются все коммуникационные команды. Когда выполняется **Background Macro**, порт COM1 будет закрыт. При выполнении команды **LUCKCOM** (0,0), выполнение макросов **ON Macro** и **Screen Cycle Macro** будет приостановлено. После выполнения команды **UNLOCKCOM** в макросе **Background Macro**, выполнение макросов **ON Macro** и **Screen Cycle Macro** может быть снова запущено. Это позволит избежать помех в коммуникации и неточностей в подсчетах.		

Пояснения к команде UNLOCKCOM	
Backgroun Macro $0 = LOCKCOM(0,0) $1 = PUTCHARS($1, 3, 300)	On Macro UNLOCKCOM(0)
При макрокомандах показанных выше пользователь может закрыть COM-порт и передавать данные используя **Background Macro**, и открыть COM-порт, используя **ON Macro**. Это означает разделение операций открытия и закрытия COM-порта.	

■ STATIONON (Station ON)

Выражение: STATIONON (Var1, Var2)

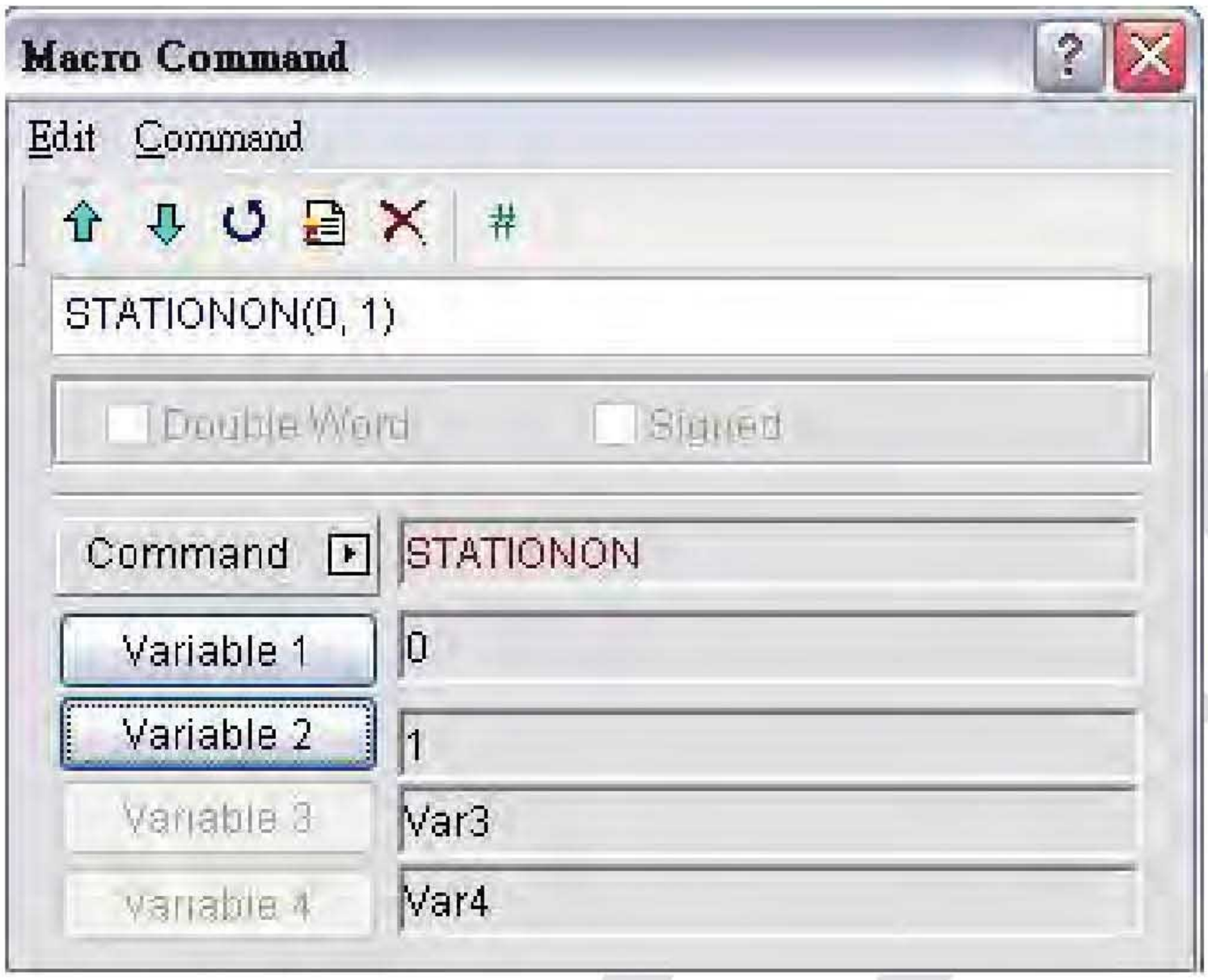
Описание: Эта команда разрешает работу устройства с номером, определяемым операндом Var2, подключённого к порту COM, номер которого задан операндом Var1 (константа, принимающая значения 0(COM1), 1(COM2) или 2(COM3, только для некоторых моделей).

Замечания:

- Операнды Var1 и Var2 могут размещаться во внутренней памяти или быть константой.
- Эта макрокоманда не может быть использована, когда стоит флаг **When Communication Interrupt then ignore** (**Options > Configuration > COM Port**).

Пример

Разрешить работу устройства 1, подключённого к COM1.



■ STATIONOFF (Station OFF)

Выражение: STATIONOFF (Var1, Var2)

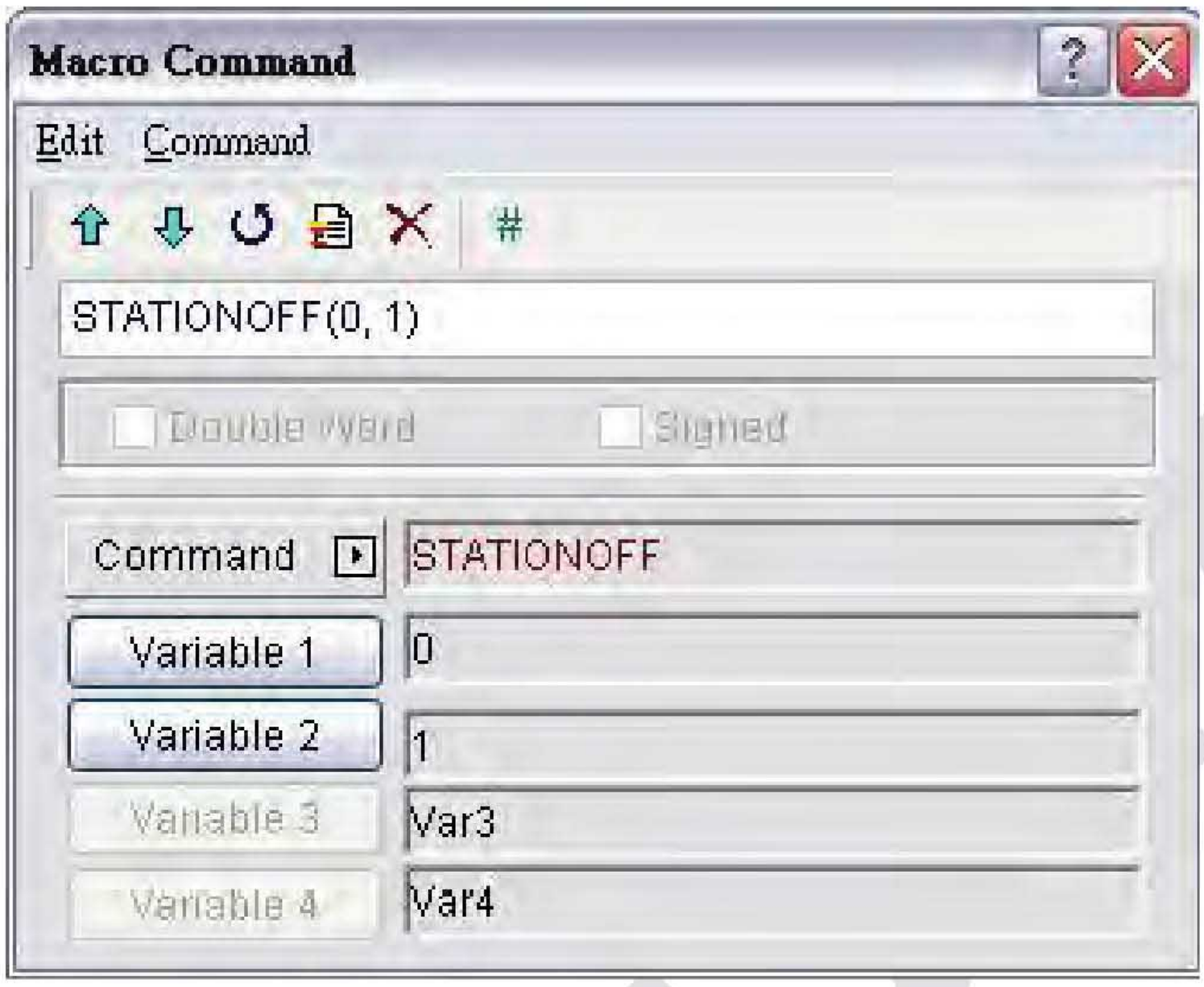
Описание: Эта команда запрещает работу устройства с номером, определяемым операндом Var2, подключённого к порту COM, номер которого задан операндом Var1 (константа, принимающая значения 0(COM1), 1(COM2) или 2(COM3, только для некоторых моделей).

Замечания:

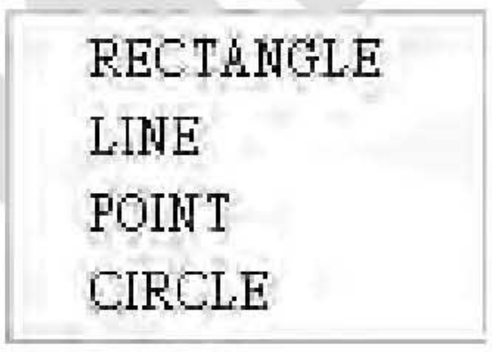
- Операнды Var1 и Var2 могут размещаться во внутренней памяти или быть констнтой.
- Эта макрокоманда не может быть использована, когда стоит флаг **When Communication Interrupt then ignore** (**Options > Configuration > COM Port**).

Пример

Отключить устройство 1, подключённое к COM1.



3.14.3.9 Рисование фигур (Drawing)



Приводятся различные команды, обеспечивающие рисование на экране

- **RECTANGLE** (Рисование прямоугольника)

  Выражение: RECTANGLE (Var1)

  Описание: Команда применяется для рисования на экране прямоугольника, операнды задают координаты углов:

  Var1 - верхняя левая X-кордината

  Var1+1 – верхняя левая Y-координата.

  Var1+2 – ширина прямоугольника

  Var1+3 –высота прямоугольника

  Var1+4 –цвет прямоугольника.

Замечания:

- Операнд Var1 может размещаться только во внутренней памяти

Пример

Если подпрограмма **Screen Open Macro** имеет следующий вид:

```
$0 = 20
$1 = 20
$2 = 100
$3 = 100
$4 = 25
```

и **Screen Cycle Macro** содержит команду

```
RECTANGLE($0),
```

то на экране будет изображён синий прямоугольник:

■ **LINE (Draw a Line)** Рисование линии

Выражение: LINE (Var1)

Описание: Команда применяется для рисования прямой линии.

Var1 –начальная X-координата.

Var1+1 –начальная Y-координата.

Var1+2 –конечная X-координата.

Var1+3 –конечная Y- координата.

Var1+4 –ширина линии.

Var1+5 – цвет.

Замечания:

- Операнд Var1 может размещаться только во внутренней памяти.

Пример

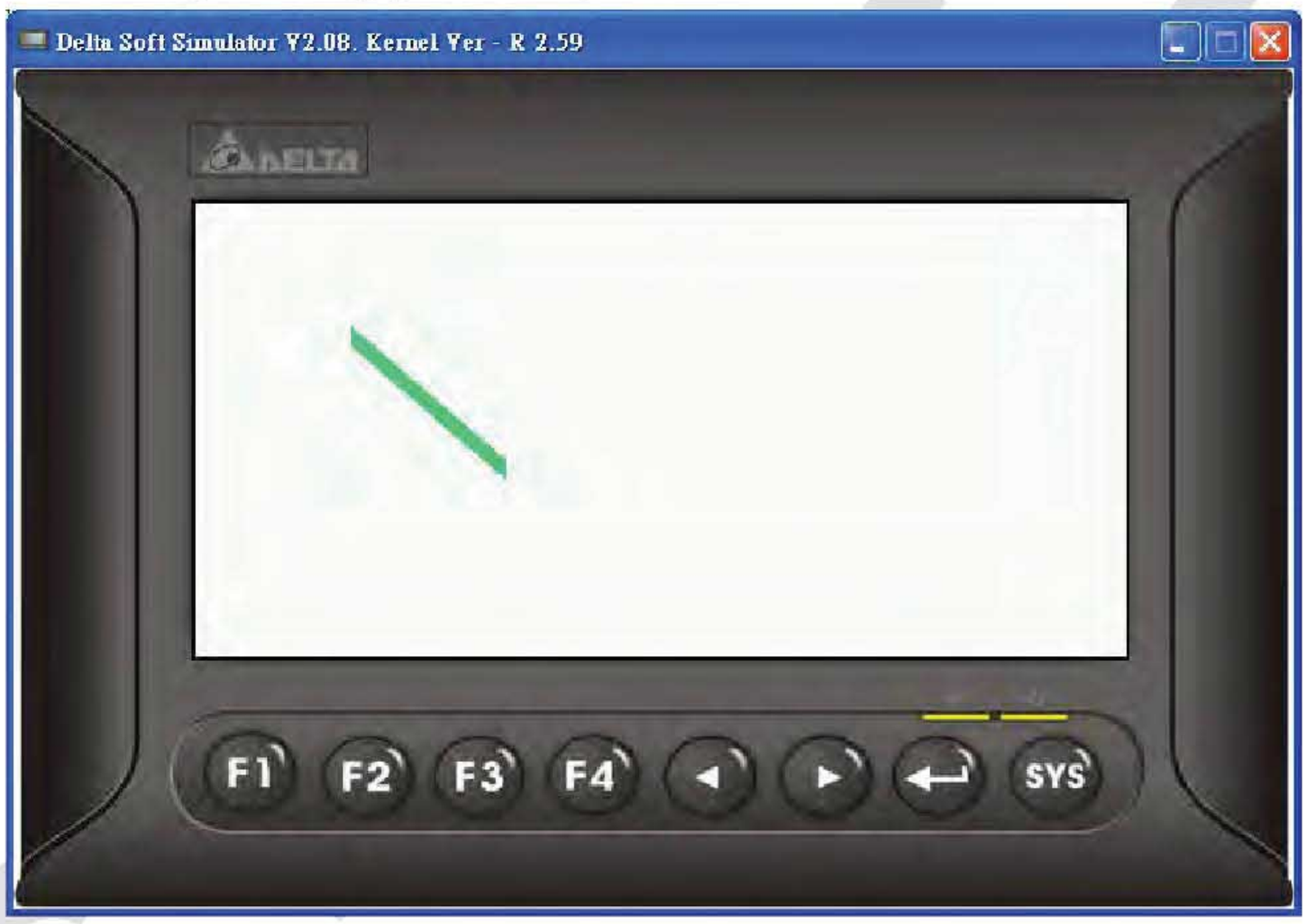
Если подпрограмма **Screen Open Macro** имеет следующий вид:

```
$0 = 80
$1 = 80
$2 = 160
$3 = 160
$4 = 10
$5 = 10000
```

и **Screen Cycle Macro** содержит команду

```
LINE($0),
```

то на экране будет изображёна зелёная линия:



■ **POINT (Draw a Point)** Рисование точки

Выражение: POINT (Var1)

Описание: Команда применяется для рисования точки.

Var1 – X-координата.

Var1+1 – Y-координата.

Var1+2 – цвет точки.

Замечания:

- Операнд Var1 может размещаться только во внутренней памяти.

Пример

Если подпрограмма **Screen Open Macro** имеет следующий вид:

```
$0 = 80
$1 = 80
$2 = 1000
```

и **Screen Cycle Macro** содержит команду

```
POINT($0)
```

,

то на экране будет изображена точка:

■ **CIRCLE (Draw an Ellipse)** Рисование эллипса

Выражение: CIRCLE (Var1)

Описание: Команда применяется для рисования эллипса.

Var1 – X-координата центра

Var1+1 – Y-координата центра

Var1+2 – длина эллипса.

Var1+3 – ширина эллипса

Var1+4 – цвет

Замечания:

- Операнд Var1 может размещаться только во внутренней памяти.

Пример

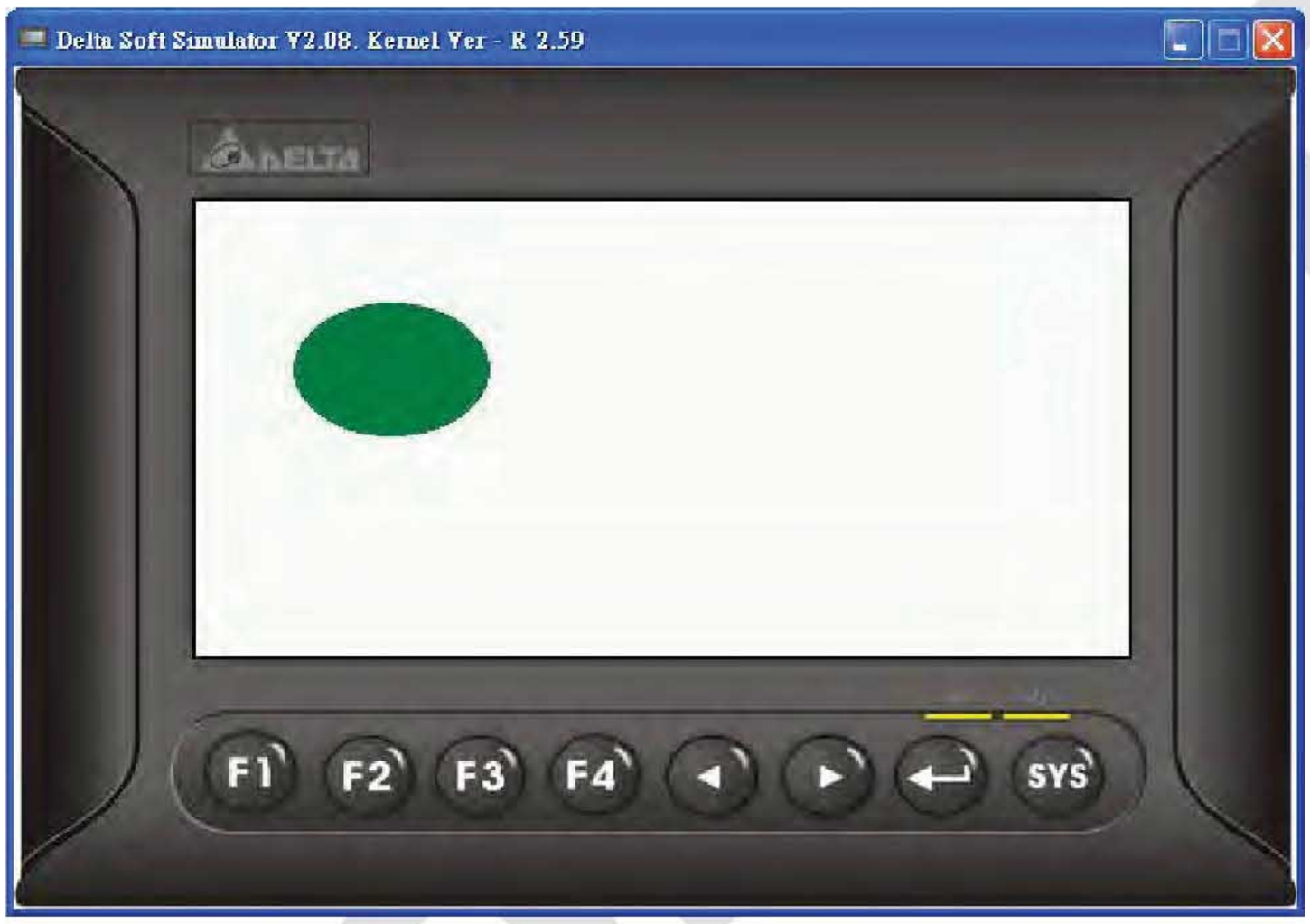
Если подпрограмма **Screen Open Macro** имеет следующий вид:

```
$0 = 100
$1 = 100
$2 = 50
$3 = 40
$4 = 1000
```

и **Screen Cycle Macro** содержит команду

```
CIRCLE($0),
```

то на экране будет изображен зелёный эллипс:



3.14.3.10 Прочие команды

- Time Tick
- GETLASTERROR
- Comment
- Delay
- GETSYSTEMTIME
- SETSYSTEMTIME
- GETHISTORY
- EXPORT

■ **Time Tick** (Измерение текущего времени от момента включения)

Выражение: Var1=TIMETICK

Описание: Команда позволяет считать время, от момента включения в мс и сохранять в заданном операндом Var1 регистре.

Замечания:

- Операнд Var1 может размещаться только во внутренней памяти.

Пример

Хранение в регистре $0 текущего времени.

■ **GETLASTERROR** Считать величину последней ошибки

Выражение: Var1 = GETLASTERROR

Описание: Команда даёт возможность считать значение последней ошибки, при отсутствии ошибок результат её выполнения будет равен 0.

При выполнении макросов одновременно, на их работу сообщение об ошибках не повлияет.

Подробная информация по кодам ошибок приведена в разделе 3.14.4.

Замечания:

- Операнд Var1 может размещаться только во внутренней памяти.

Пример

Записать значение последней ошибки в $0.

■ **# (Comment)** Комментарий

Выражение: # *Комментарий*

Описание: Эта команда даёт возможность делать описание макросов, комментировать шаги, исключать из исполнения команды макросов.

Пользователю необходимо только поместить в начале строки значок #.

Замечания:

- Выражение может быть текстовой строкой или строкой команд.

Пример

Строка текста как комментарий `#This is a Comment.`

Строка команд как комментарий `#$0 = $ 0 + 1`

■ **Delay (System Delay)** Задержка

Выражение: Delay (Var1)

Описание: Эта команда обеспечивает задержку (в мс) при установке системных уставок.

Установка большого значения снижает время отклика панели оператора.

Замечания:

- Операнд Var1 может размещаться во внутренней памяти или быть константой
- После выполнения команды Delay панель оператора остановит работу, и продолжит по окончанию заданного интервала.

Пример

Задержка 2 секунды.

```
Delay(2000)
```

■ **GETSYSTEMTIME** Считывание системного времени (Get System Time)

Выражение: Var1 = GETSYSTEMTIME

Описание: Команда обеспечивает считывание и запоминание значения часов реального времени.

Операнд Var1 задаёт начальный адрес записи 7 слов во внутренней памяти.

Var1 отображает год

Var1 + 1 - месяц

Var1 + 2 - день

Var1 + 3 - неделю

Var1 + 4 - час

Var1 + 5 - минуту.

Var1 + 6 - секунду.

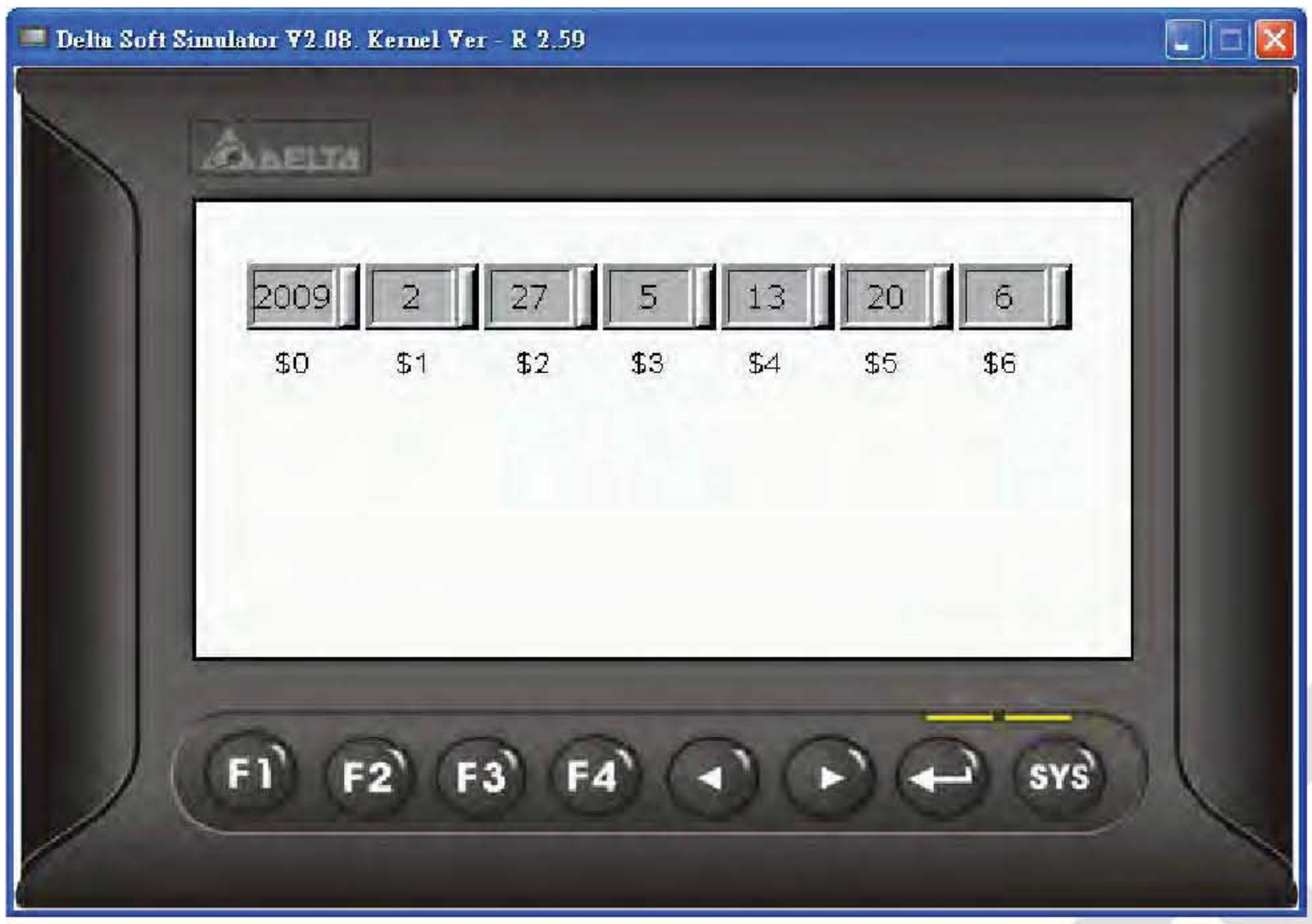
Замечания:

- Операнд может Var1 размещаться во внутренней памяти или быть константой.

Пример

Пример записи текущего системного времени 2009/02/27 FRI 13:20:06.

Команда сохраняет его значение в регистрах $0 … $6.



■ **SETSYSTEMTIME** Установка системного времени (Set System Time)

Выражение: SETSYSTEMTIME (VAR1)

Описание: Установка системного времени.

Операнд Var1 задаёт начальный адрес записи 7 слов во внутренней памяти.

Var1 задаёт год

Var1 + 1 - месяц

Var1 + 2 - день

Var1 + 3 - неделю

Var1 + 4 - час

Var1 + 5 - минуту.

Var1 + 6 - секунду.

Замечания:

- Операнд Var1 может размещаться во внутренней памяти или быть константой.

Пример

Предположим, что необходимо установить системное время 2009/02/27 FRI 13:25:34., тогда подпрограмма будет выглядеть следующим образом:

```
$0 = 2009
$1 = 02
$2 = 27
$3 = 5
$4 = 13
$5 = 25
$6 = 34
SETSYSTEMTIME($0)
```

- **GETHISTORY (Get History Data)** Вывод архива

Выражение: Var1 = GETHISTORY (Var2, Var3, Var4, Var5, Var6)

Описание: Вывод данных архива.

Операнд Var1 определяет адрес регистра, где хранится длина блока данных

Var2 – номер буфера архива

Var3 - начальный адрес выборки данных (хранится во внутренней памяти или константа)

Var4 –Количество читаемых точек

Var5 – адрес хранения данных

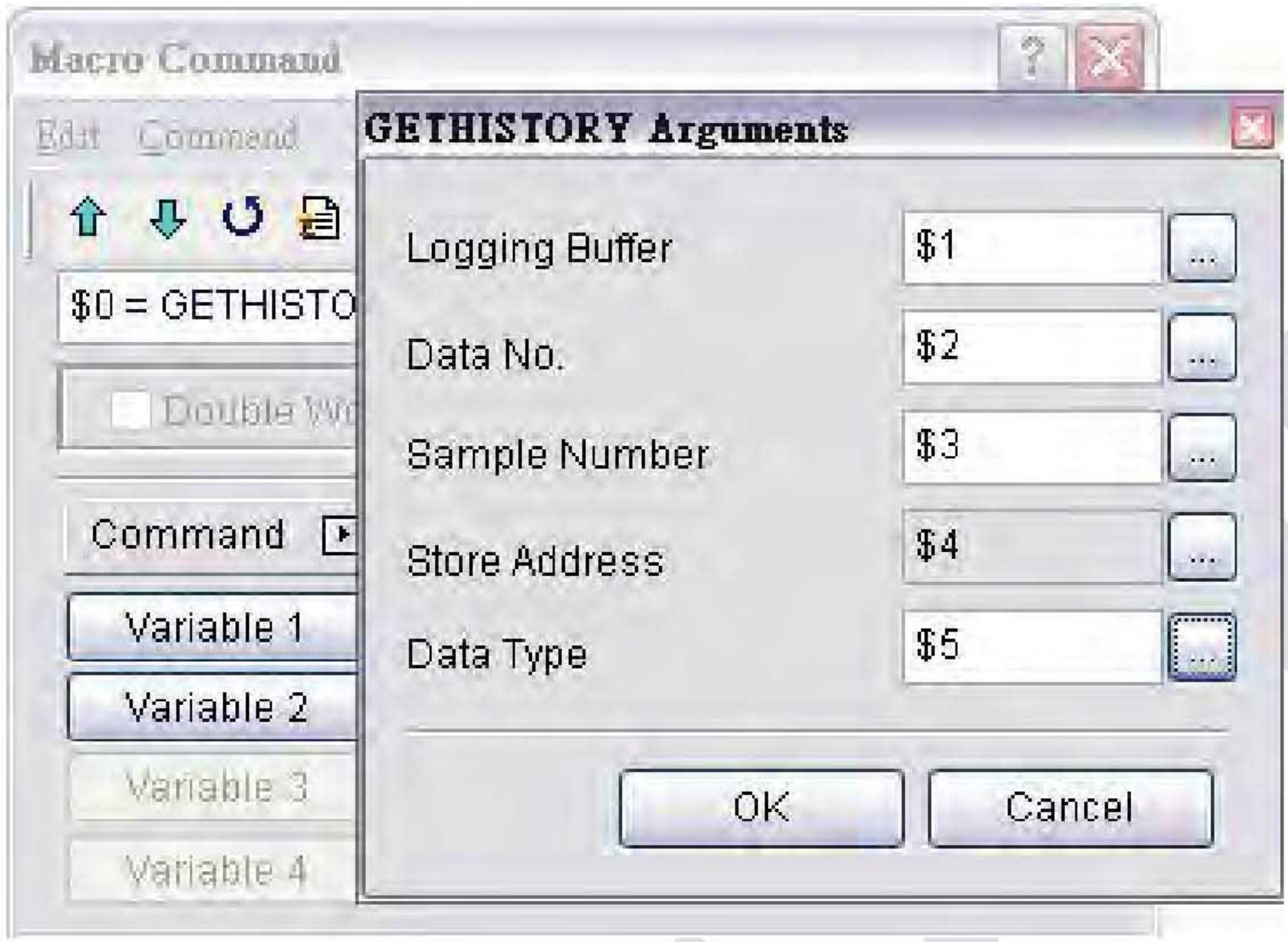
Var6 – адрес регистра где записан тип данных.

Замечания:

- Операнд Var1 может размещаться только во внутренней памяти
- Операнды Var2, Var3 и Var4 могут размещаться во внутренней памяти или быть константой.
- Var5 может быть адресом внешнего контроллера или внутренней памяти
- Var6 может быть адресом внутренней памяти или константой, где записан тип данных
  0: Data,
  1: Time,
  2: Time and Data

Пример

Когда выражение $0 = GETHISTORY ($1, $2, $3, $4, $5), панель оператора обеспечивает чтение архива



■ **EXPORT (Output Data)** Экспорт данных

Выражение: EXPORT (Var1)

Описание: Команда EXPORT выполняет те же функции, что и экранный элемент **Report List**.

Операнд Var1 определяет внешнее устройство:

0: SD card

1: USB диск

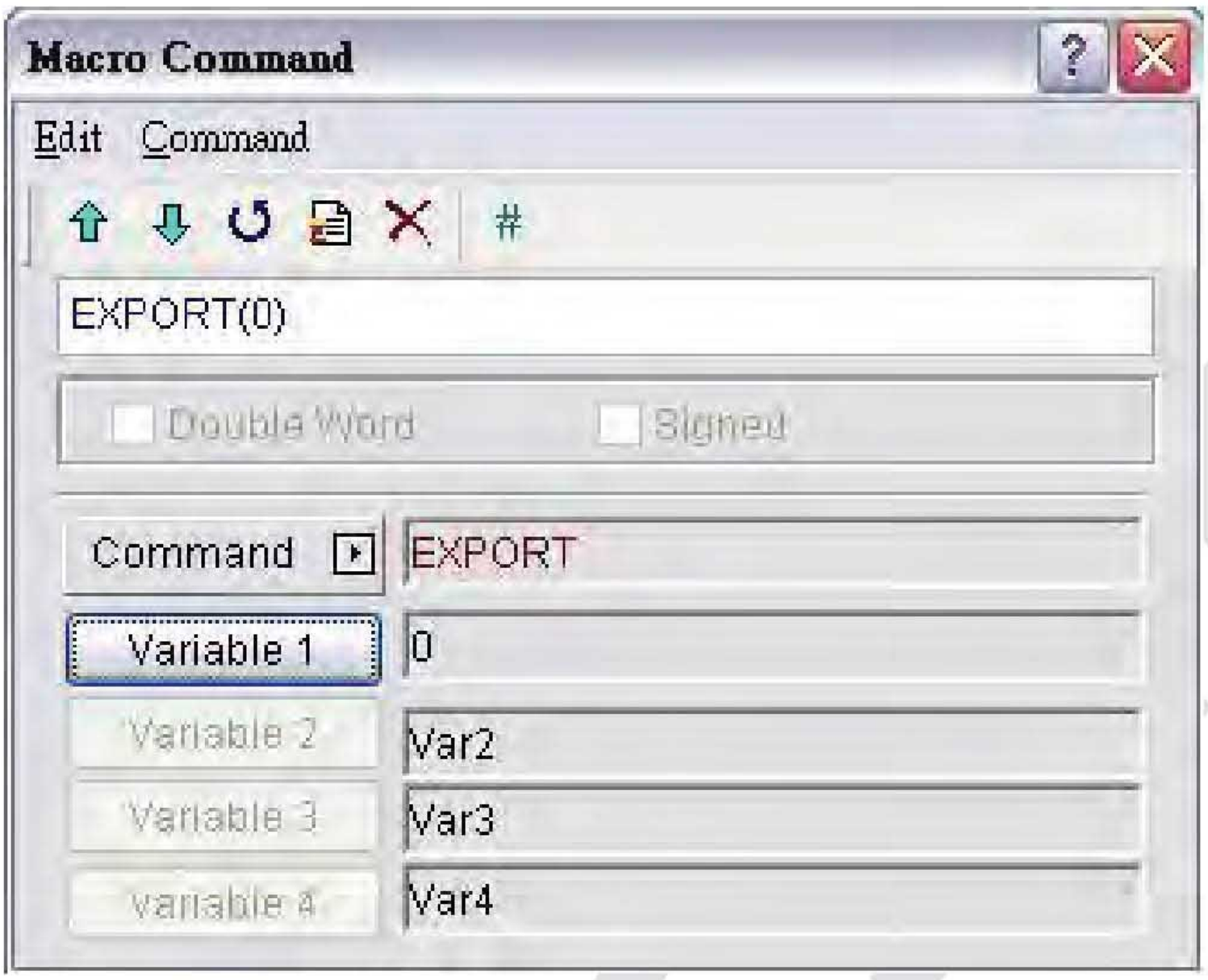
2: Принтер

Замечания:

- Операнд Var1 может быть только константой.

Пример

Команда EXPORT(0), обеспечивает экспорт данных на SD карту.



3.14.4 Сообщения об ошибках

При компиляции, в окне выходного результата будут отображаться обнаруженные ошибки. Ошибки могут появляться из-за невнимательности, неаккуратности пользователя или из-за недостатка входных команд. Некоторые ошибки в программе могут быть легко и быстро найдены. Но поиск ошибок может оказаться трудным в длинных макропрограммах. Для помощи в отладке программы, **ScrEdit** предоставляет систему поиска ошибок и сообщения о них. При выполнении макрокоманд, наличие ошибок в макросе не влияет на выполнение других макросов.

Сообщения при редактировании

- **Code – 100: LABEL cannot be found.**

Нет метки перехода для команды GOTO.

- **Code – 101 Recursion occurs**

Это сообщение обычно появляется в подпрограммах (sub-macro). Способность субмакроса вызывать самого себя называется рекурсией. Независимо от того, как происходит вызов: прямо или косвенно. В основном, рекурсия неприменима для подпрограмм. Для этих целей пользователи могут применять команды GOTO или FOR (много раз).

■ Code – 102 More than 10 nested FOR is used

Это сообщение предупреждает пользователя о том, что число вложенных циклов FOR более 10, чтобы избежать чрезмерного использования памяти. Для этих целей пользователи могут применять команды GOTO или IF.

■ Code – 103 Sub-macro does not exist

Это сообщение предупреждает пользователя об обращении к несуществующей подпрограмме. Например, CALL 5 должна вызвать субмакрос 5. Если подпрограмма sub-macro 5 не создана, то появится данное сообщение об ошибке.

■ Code – 104 Number of NEXT is less than the number of FOR

Количество команд NEXT и FOR не совпадает. Это сообщение указывает пользователю, что он забыл в программе поставить команду NEXT.

■ Code – 105 Number of FOR is less than the number of NEXT

Количество команд NEXT и FOR не совпадает. Это сообщение указывает пользователю, что он забыл в программе поставить команду FOR.

■ Code – 106 Repeated LABEL

Это сообщение указывает пользователю, что в одной макропрограмме повторяется одна и та же метка (LABEL).

■ Code – 107 There is RET in Macro

Это сообщение обозначает, что в основной макропрограмме использована команда RET, которая должна применяться только в подпрограммах (sub-macro) для возврата в главную программу. В главной программе должна применяться команда END, а не RET.

Сообщения при ошибках в макросе

Сообщение об ошибке в одном макросе не будет влиять на выполнение других макропрограмм.

■ Code – 10 GOTO Error

Это сообщение означает, что в макропрограмме имеется ошибка безусловного перехода GOTO.

■ Code – 11 Stack Overflow

Это сообщение означает, что стек в макропрограмме полон. Причиной этого, может быть использование слишком большого количества подпрограмм или одновременное выполнение большого количества макросов. Сообщение служит для того, что бы избежать чрезмерного использования памяти.

■ Code – 12 Empty Sub-macro

Это ошибка выполнения подпрограммы. Подпрограмма (sub-macro), которая вызвана, не должна быть пустой. Это сообщение позволяет избежать неожиданной ошибки.

- **Code – 13 Data Read Error**

Ошибка чтения данных. Иногда это может быть вызвано ошибкой чтения данных оперативной памяти, но в большинстве случаев - ошибкой чтения данных из ПЛК.

- **Code – 14 Data Write Error**

Ошибка записи данных. Иногда это может быть вызвано ошибкой записи данных в оперативную память, но в большинстве случаев - ошибкой записи данных в ПЛК.

- **Code – 15 Divisor is 0**

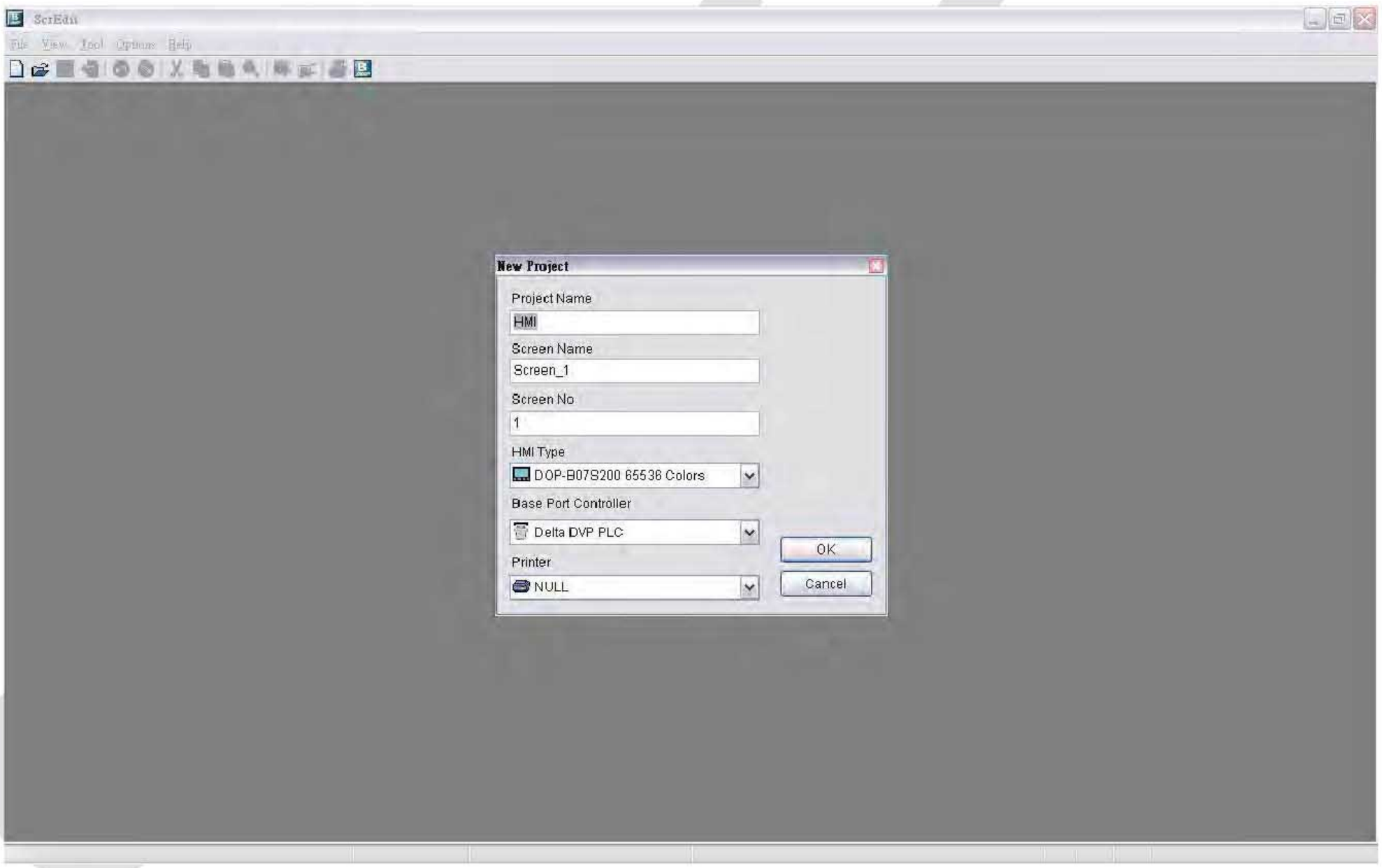
Ошибка деления на ноль.

Глава 4. Примеры

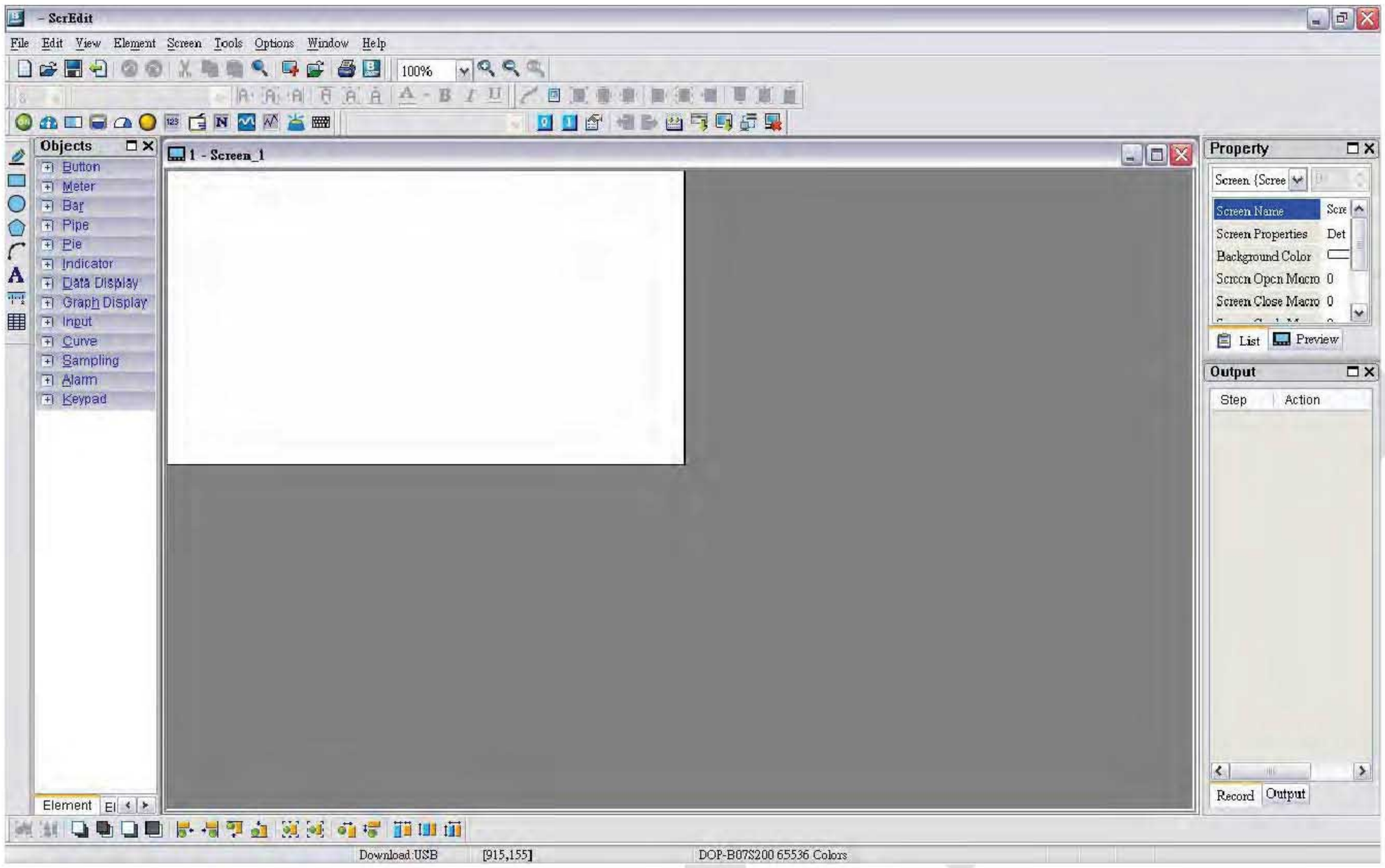
4.1 Создание 16-битные рецептов

Открыть новый проект, для чего кликнуть мышью на значок  или **File > New**.

Диалоговый экран приведён ниже:

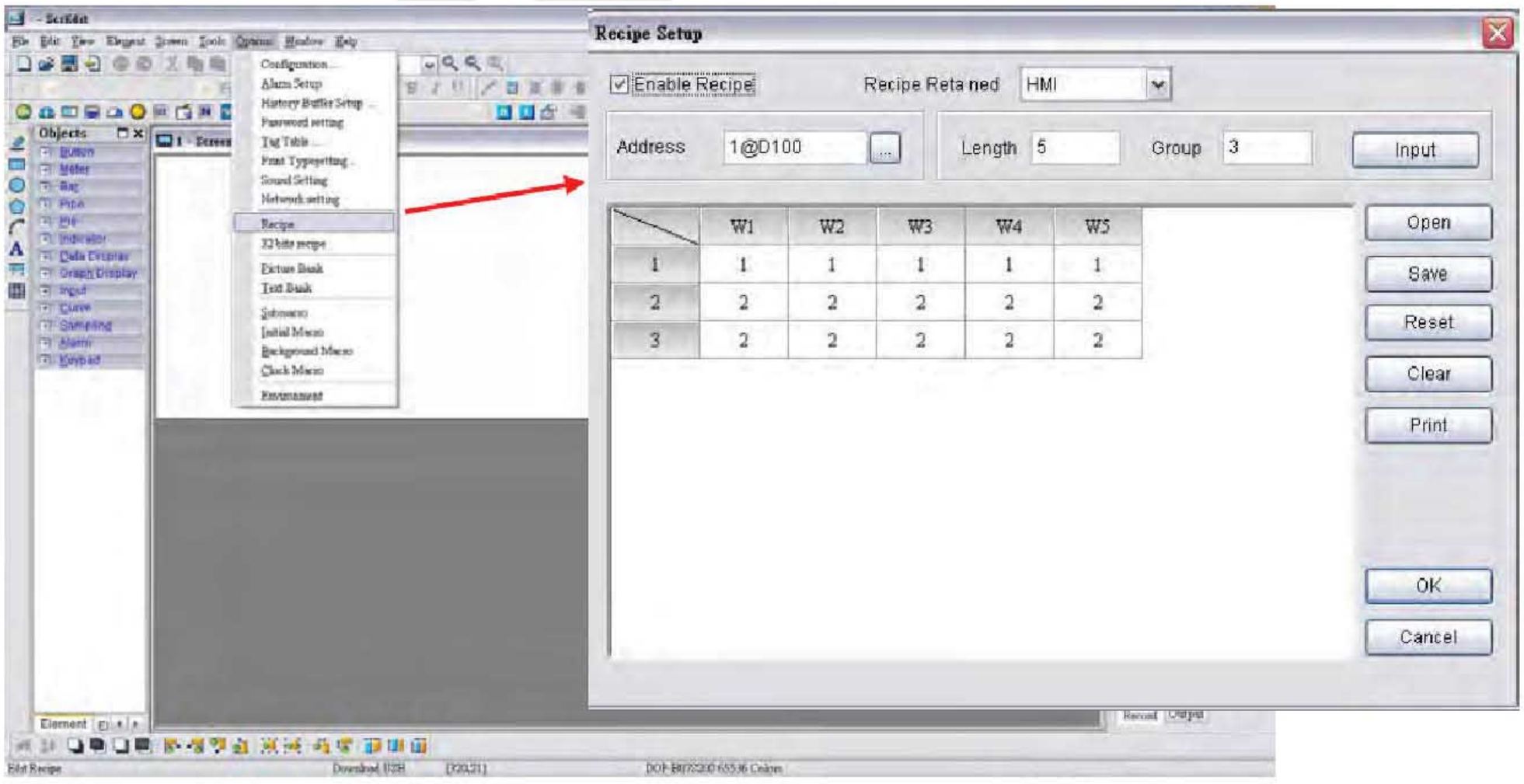


Необходимо ввести наименование проекта, наименование экрана, номер экрана определить тип панели, контроллера или принтера, далее подтвердить выбор, кликнув мышью ОК. Как показано ниже, в программе **Screen Editor** будет открыт новый проект.



Кликните мышью **Options > Recipe,** чтобы открыть меню настройки рецептов.

Разрешить функцию рецептов и выбрать в панели внешнюю память. В приводимом примере выбраны следующие настройки - длина рецепта 5, число групп 3, адрес записи 1@D100. Нажать кнопку **Input** . Ниже показан пример экрана, отображающий введённые данные:

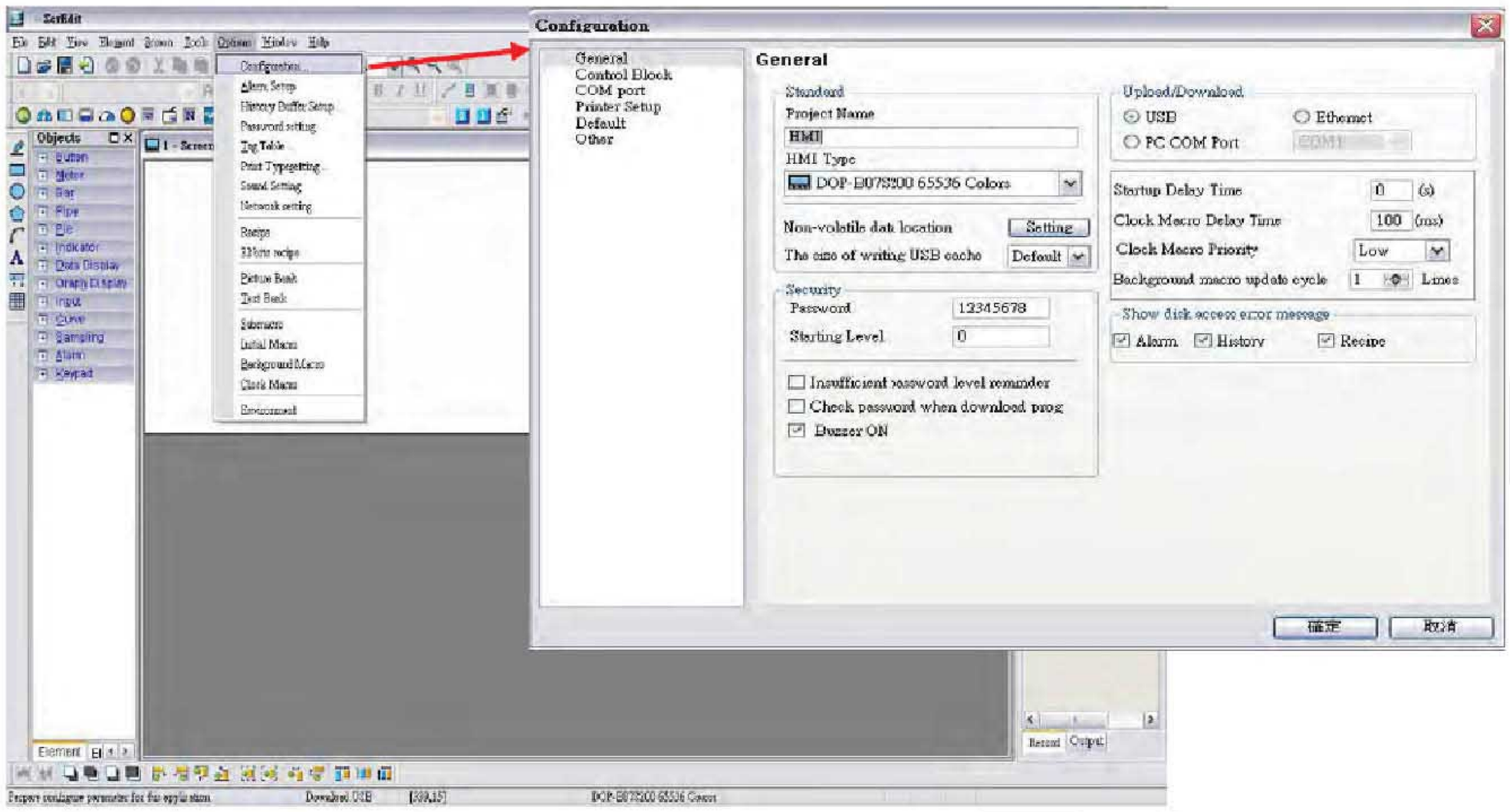


Для завершения процедуры настройки рецептов, нажать кнопку **OK.**

Выбрать **Control Block**, кликнув мышью **Options > Configuration**.

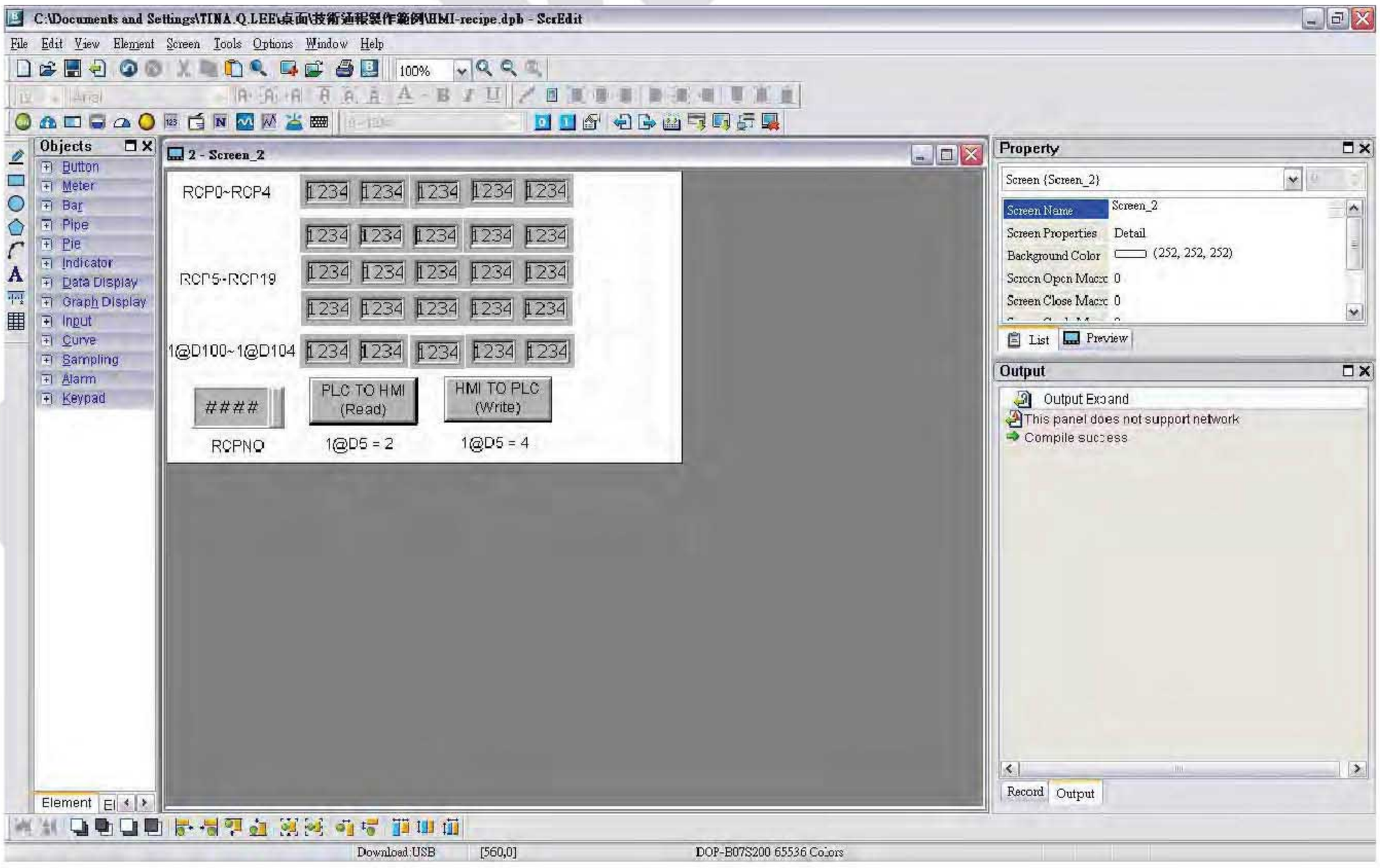
Для управления операциями чтения/записи рецептов, установить адрес управления (Control Address) 1@D0 и значение длины равным 8. Вид экрана

приведён ниже:



Для завершения настроек регистра управления нажать **ОК**.

В области редактирования рецептов открыть элементы цифрового ввода для индикации буфера рецептов (RCP0~RCP4), таблицы рецептов (RCP5~RCP19) и адресов записи рецептов (1@D100~1@D104). Дополнительно открыть элементы ввода номера целевого регистра (RCPNO), кнопку записи рецепта 1@D5=2) , кнопку чтения рецепта (1@D5=4) как показано на экране ниже:

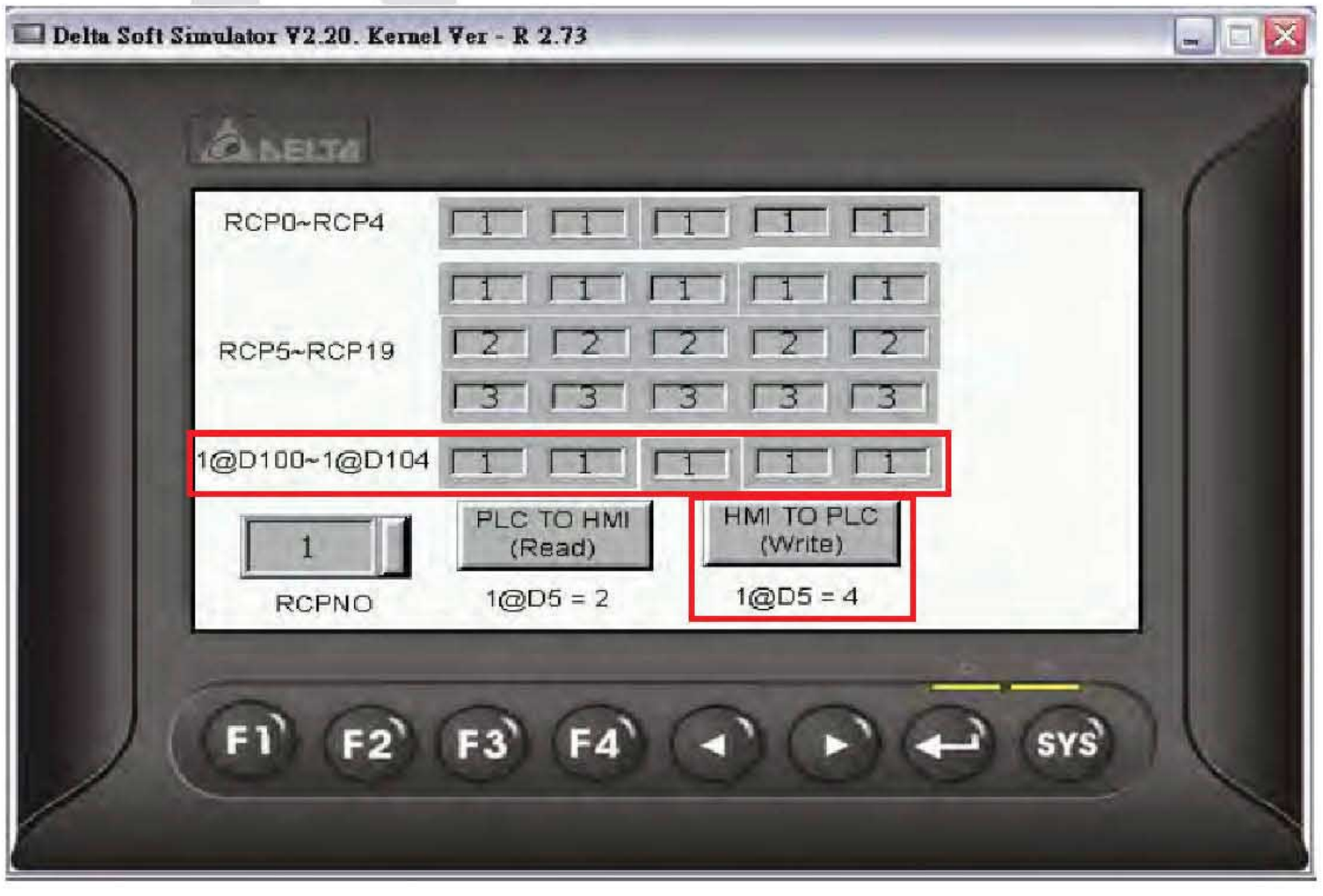


Произвести компиляцию и загрузить программу в память панели оператора.

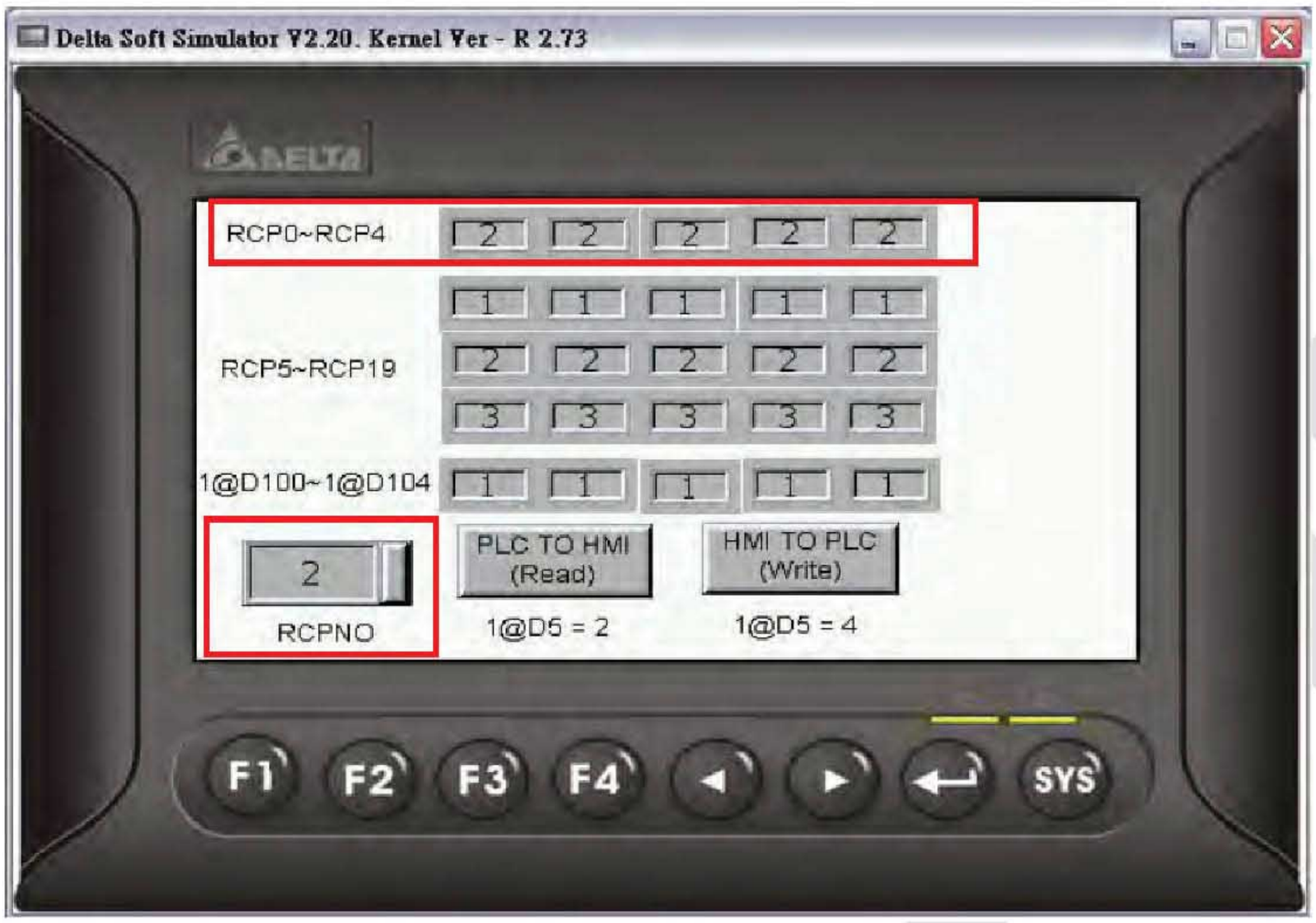
Пример пользовательского экрана будет выглядеть как показано ниже. Так как значение по умолчанию RCPNO =1, то в буфере RCP0~RCP4 индицируется 1 номер данных. Данные рецептов ешё не записаны в контроллер по адресам 1@D100~1@D104. Поэтому их содержимое равно 0.

Нажать кнопку **Write** (запись) и 1 номер данных запишется в регистры 1@D100~1@D104.

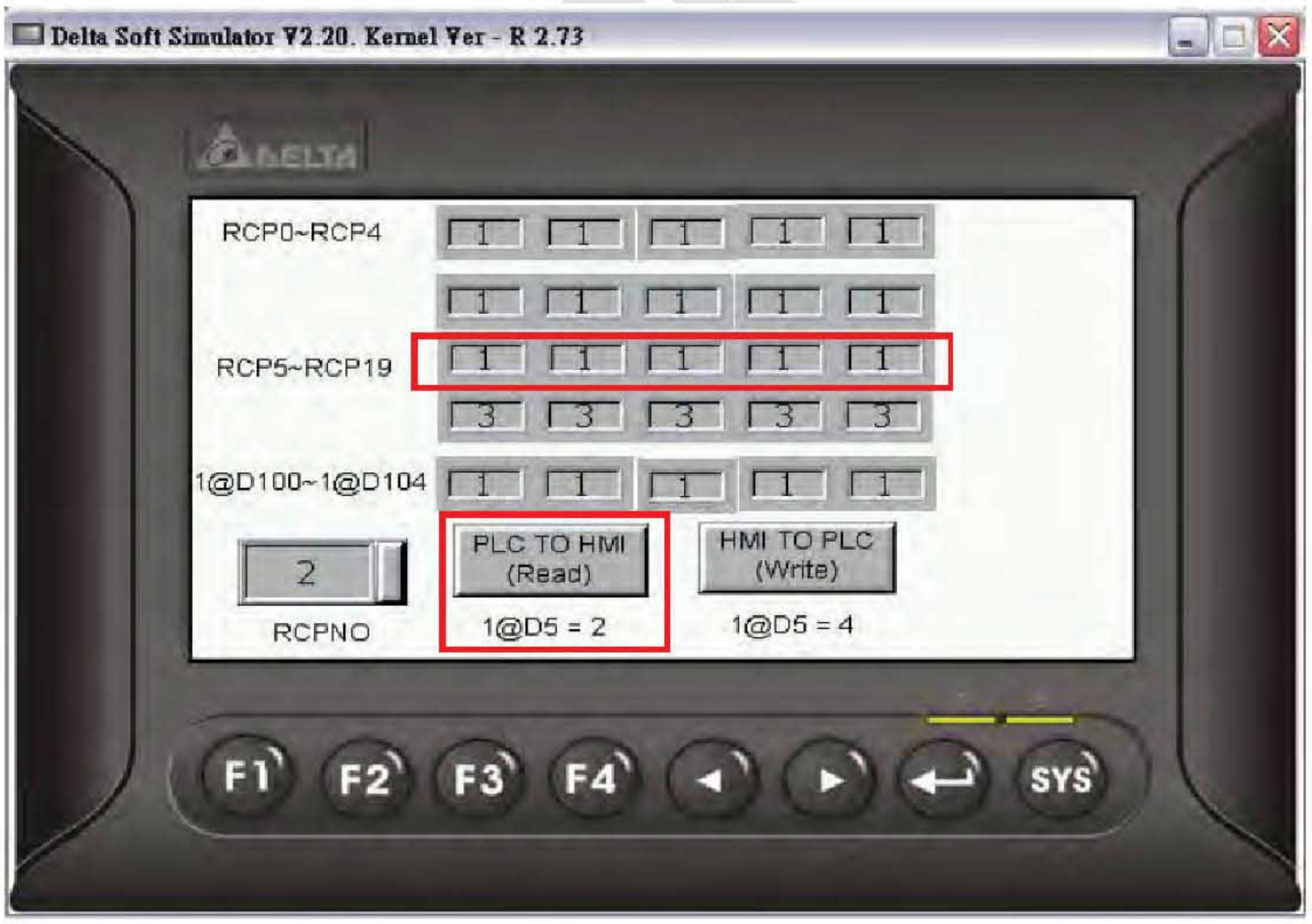
В этот момент содержимое регистров 1@D100~1@D104 станет 1, как показано рисунке ниже:



Установить RCPNO = 2 и значение содержимого RCP0~RCP4 изменится на 2.



Нажать кнопку **Read** (чтение) и после чтения данных рецептов из контроллера они сохранятся во втором наборе регистров данных.

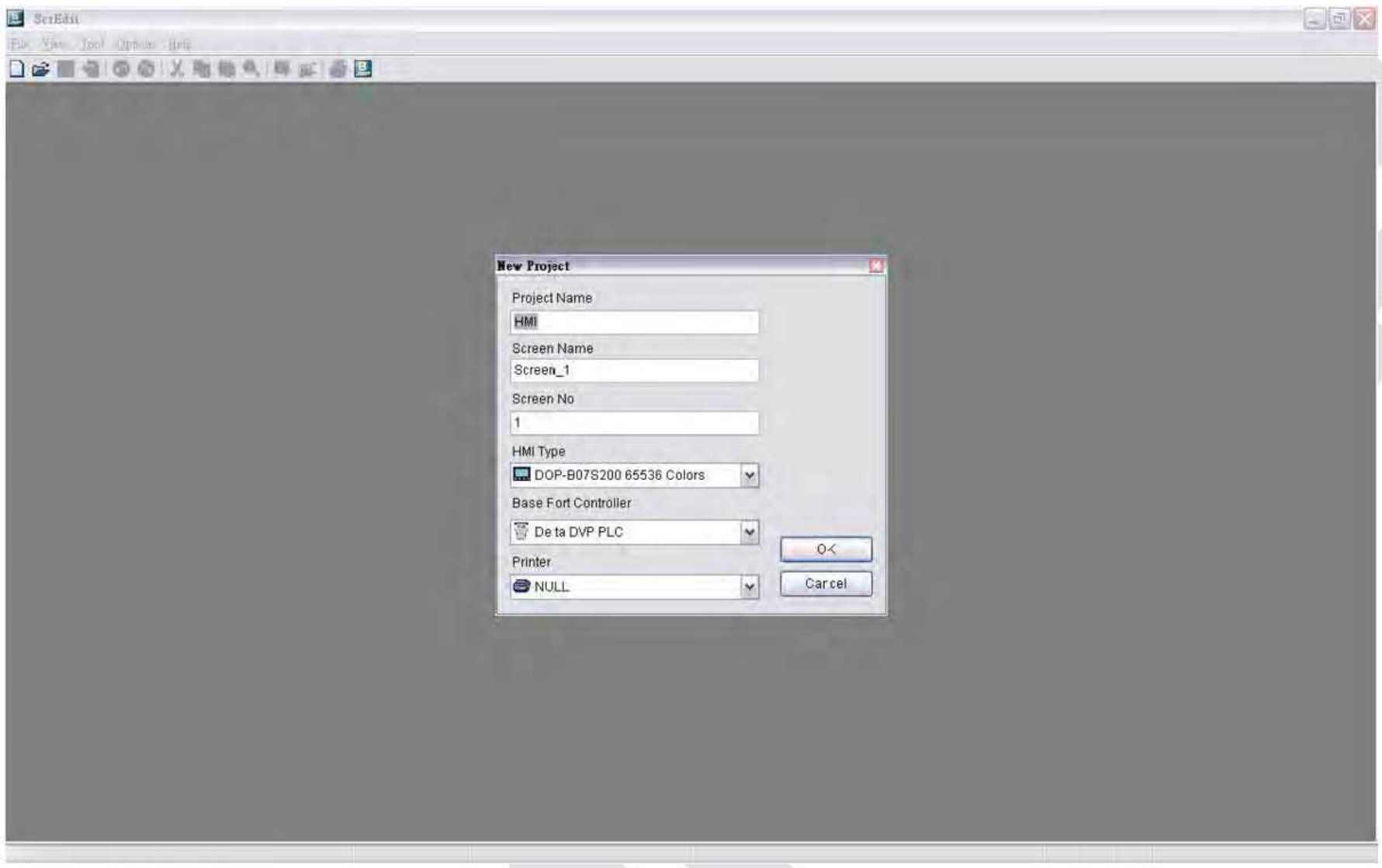


Далее, значение второй группы данных будет изменено.

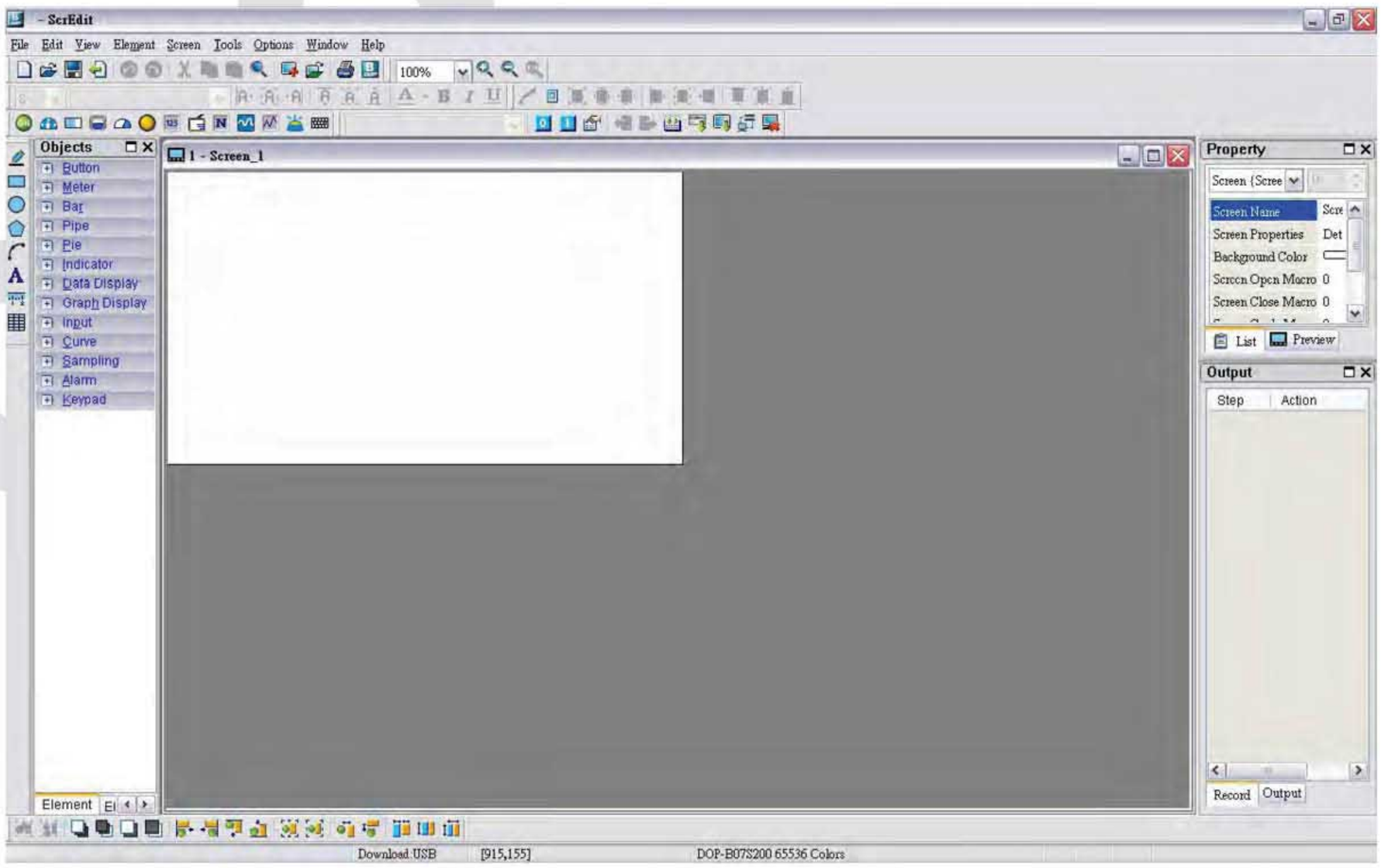
4.2 Как создать 32-битные рецепты

Открыть новый проект, для чего кликнуть мышью значок  или **File > New**.

Диалоговый экран приведён ниже:



Необходимо ввести наименование проекта, наименование экрана, номер экрана определить тип панели, контроллера или принтера, далее подтвердить выбор, кликнув мышью **OK**. Как показано ниже, в программе **Screen Editor** будет открыт новый проект



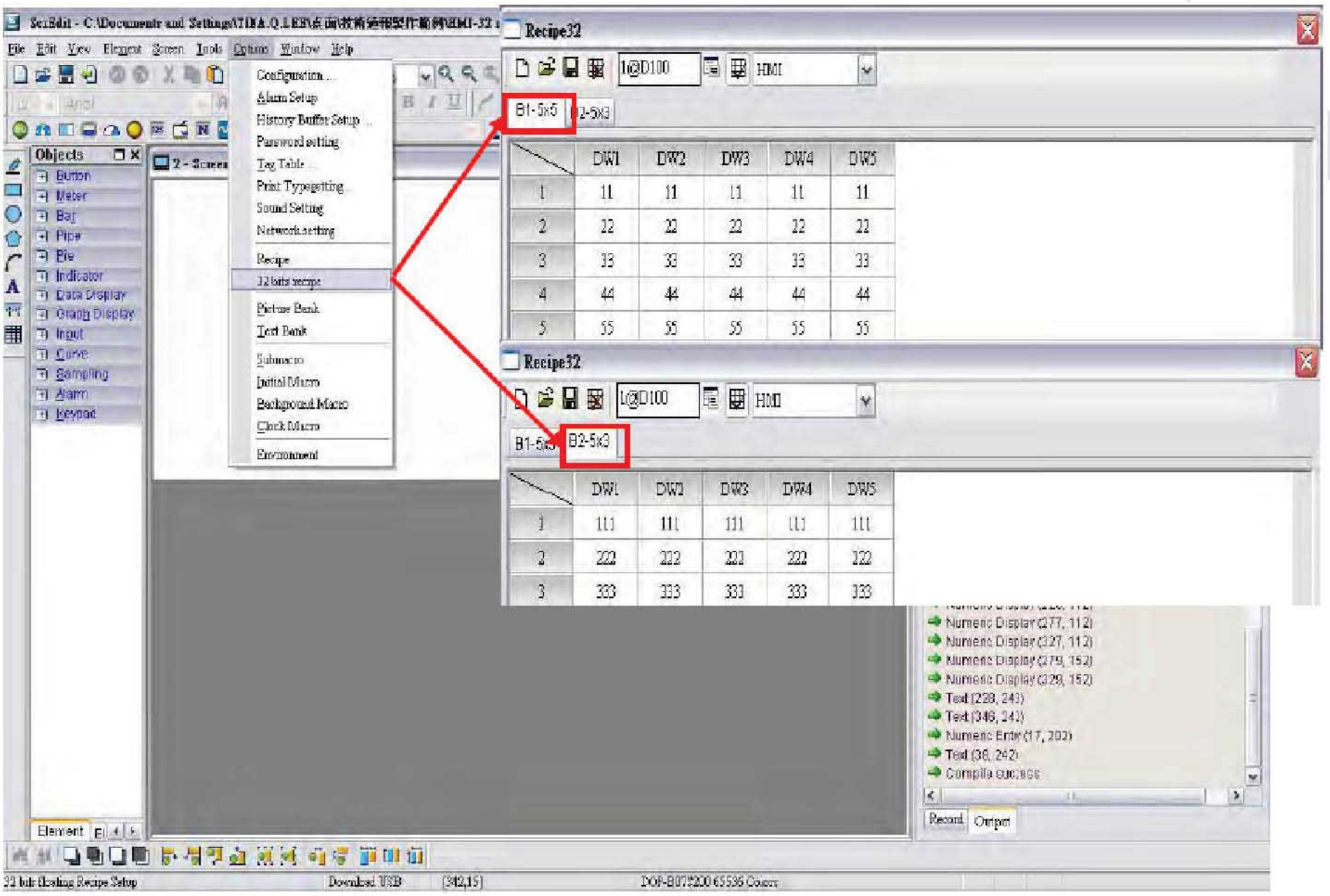
Кликнуть мышью **Options > 32bits recipe**, откроется экран настроек 32-битных

рецептов. В этом примере открыто две таблицы рецептов, адрес записи установлен 1@D100.

Задать в таблице 1 размер рецепта равным 5, длину группы 5.

Задать в таблице 2 размер рецепта равным 5. Длину группы равной 3.

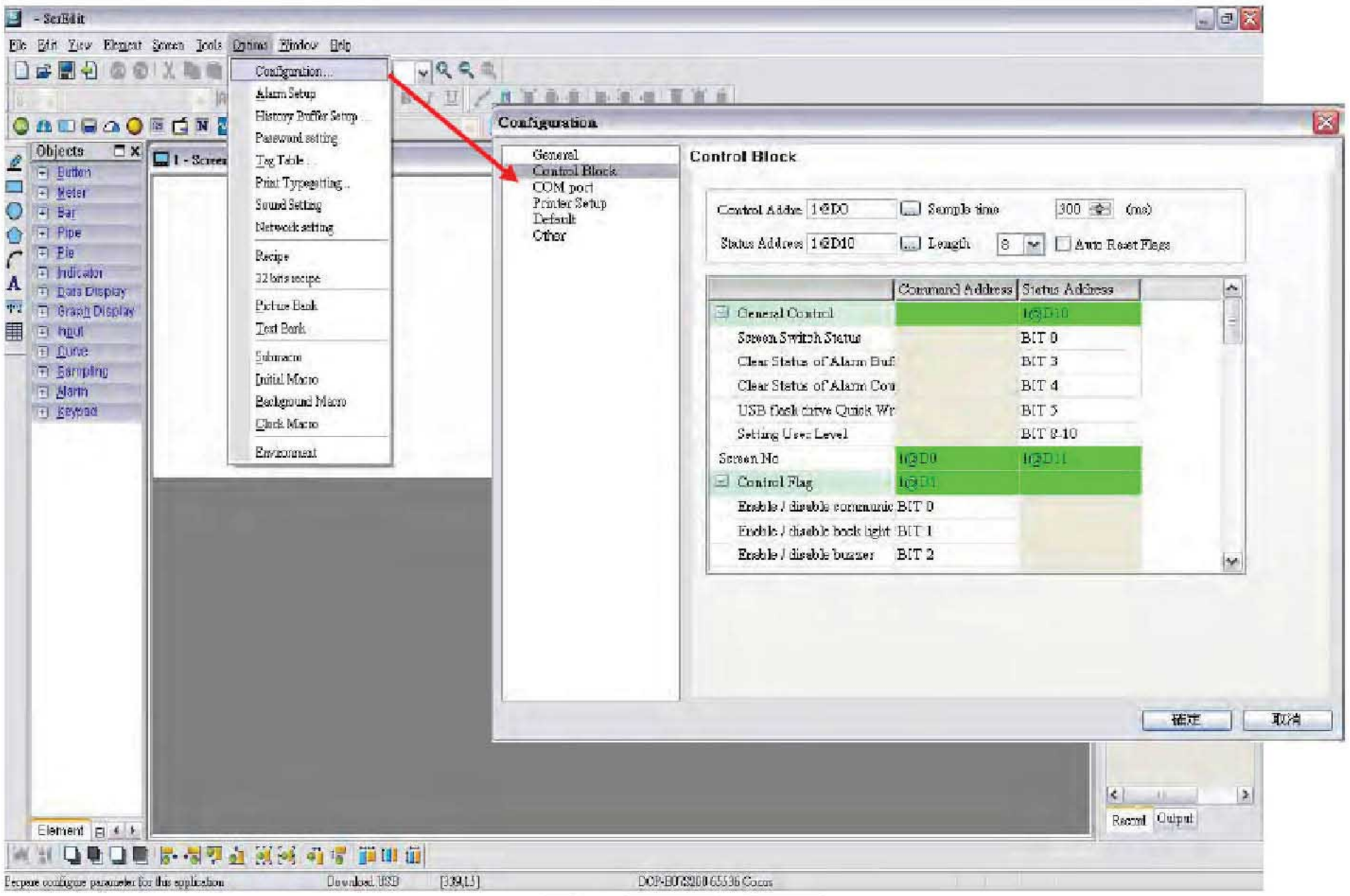
Для завершения процедуры настройки рецептов, нажать кнопку **OK**.



Далее, кликнуть значок **Save** для сохранения настроек.

Выбрать **Control Block**, кликнув мышью **Options > Configuration**.

Для управления операциями чтения записи рецептов, установить адрес блока управления (**Control Address**) 1@D0 и значение длины блока равным 8. Вид экрана приведён ниже:

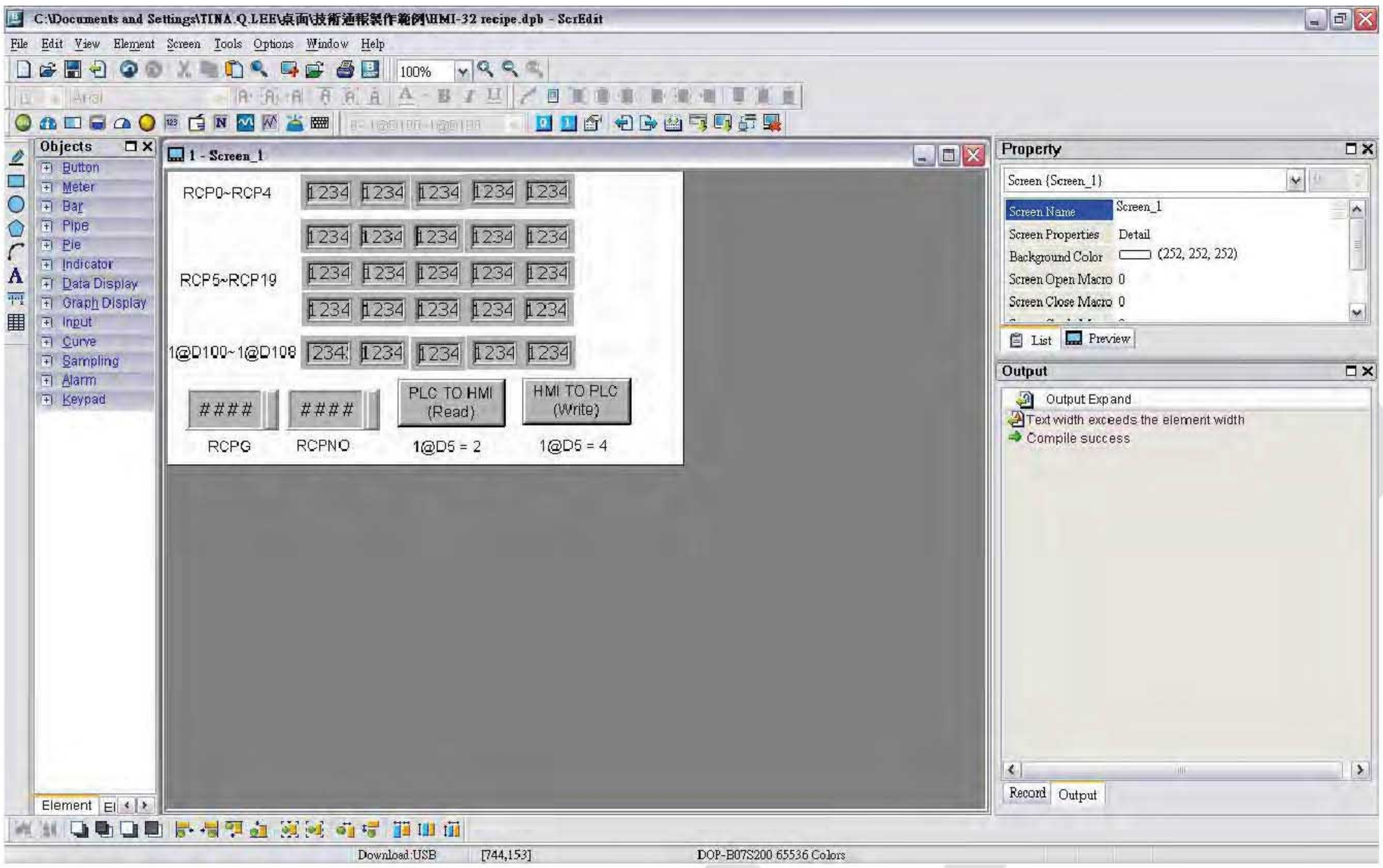


Для завершения настроек регистра управления нажать **ОК**.

В области редактирования рецептов открыть элементы цифрового ввода для индикации буфера рецептов (RCP0~RCP4), таблицы рецептов (RCP5~RCP19) и адресов записи рецептов (1@D100~1@D108).

Дополнительно, открыть элемент цифрового ввода номера целевого регистра (RCPNO), кнопку записи рецепта (1@D5=2), кнопку чтения рецепта (1@D5=4), как показано на экране ниже.

Дополнительно открыть элементы ввода номера целевого регистра (RCPNO) и номера группы регистра целевого (RCPG), кнопку записи рецепта (1@D5=2), кнопку чтения рецепта (1@D5=4), как показано на экране ниже.



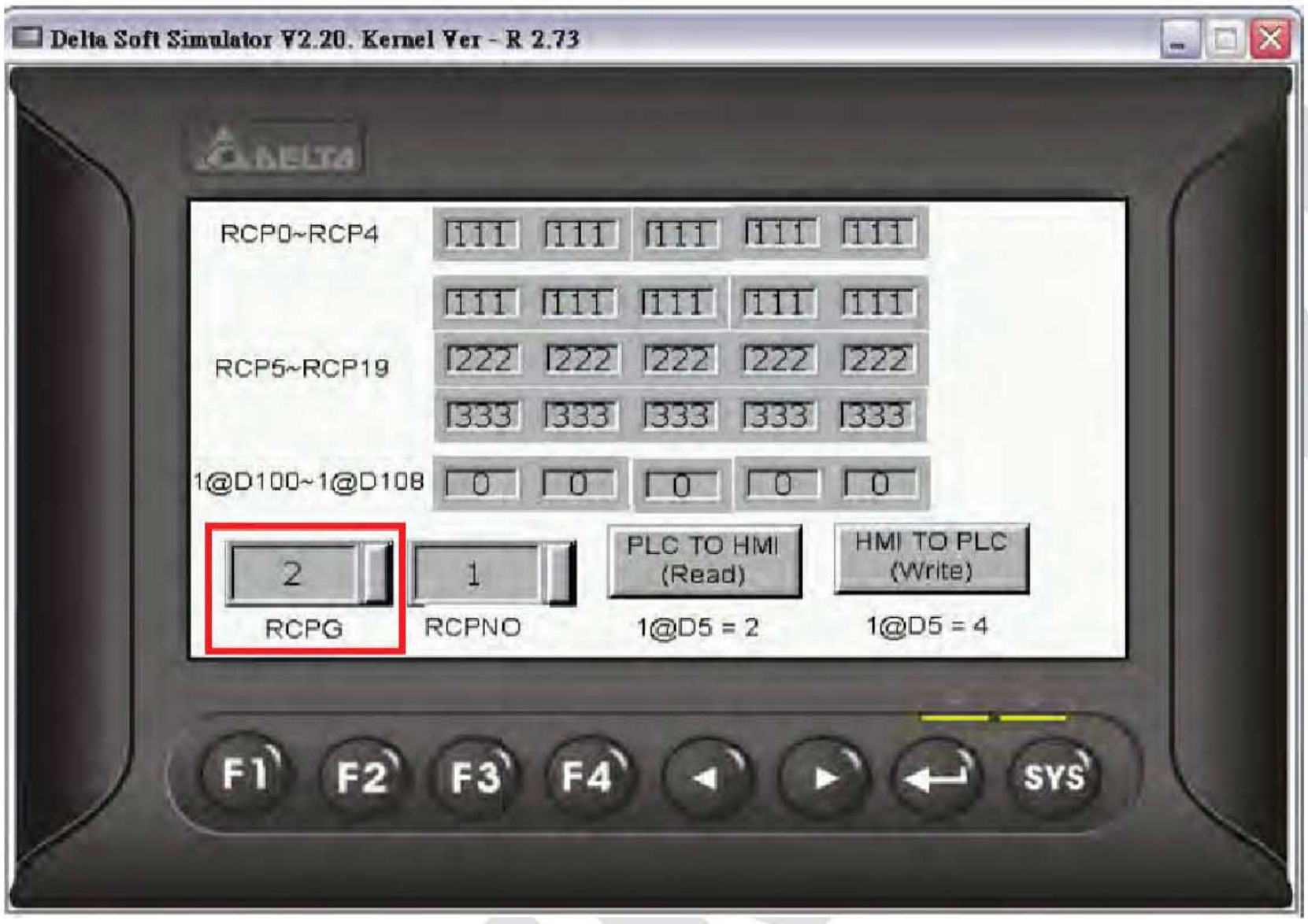
Произвести компиляцию (**Compile**) и загрузить программу в память панели оператора.

Пример пользовательского экрана будет выглядеть как показано ниже. Так как значение RCPG =0 и RCPNO =1, то содержимое буфера RCP0~RCP4 и таблицы рецептов RCP5~RCP19 будет показано как 0.

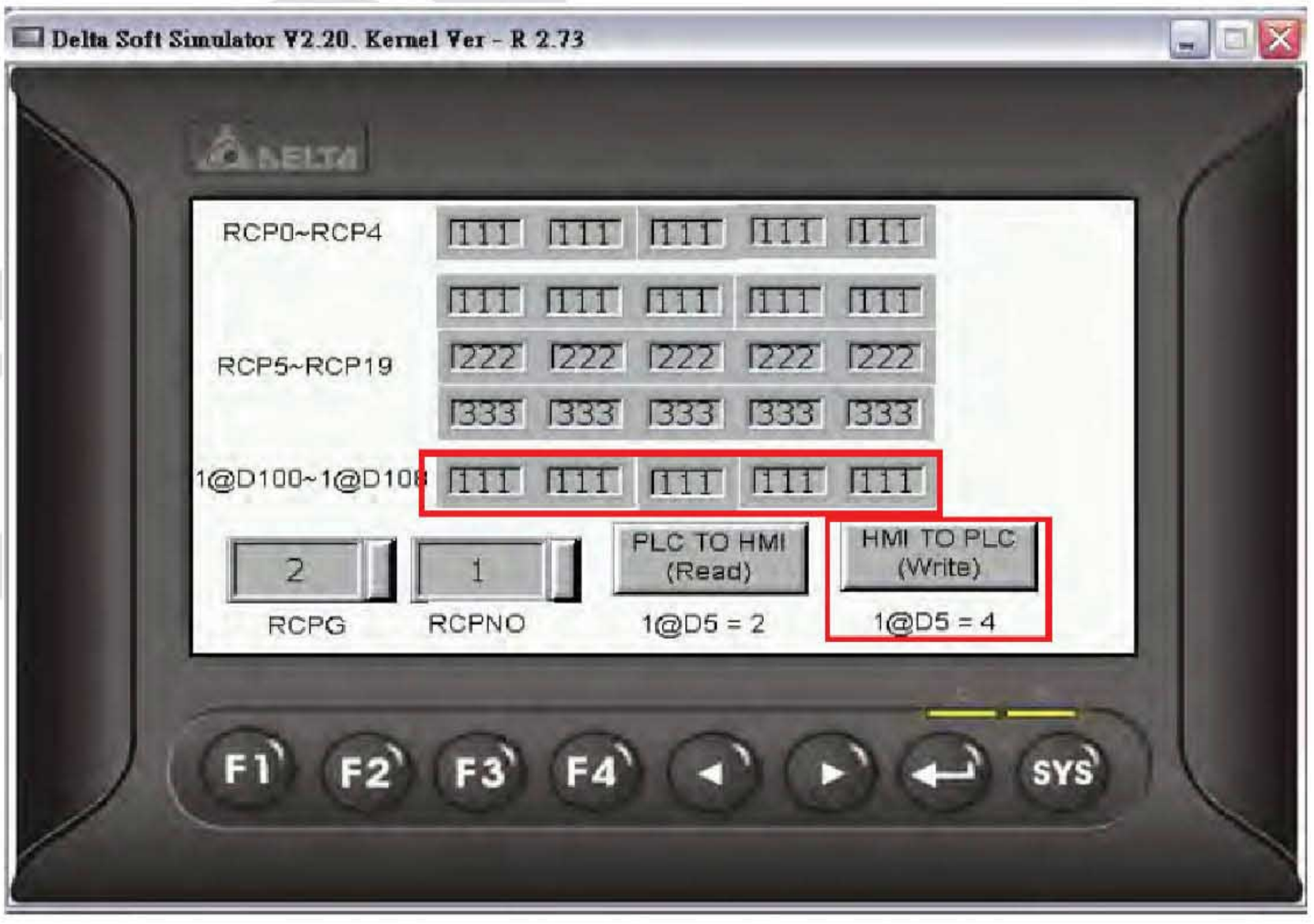
В это время, данные рецептов не записаны в контроллер по адресам 1@D100~1@D108. Поэтому содержимое регистров 1@D100~1@D108 равно 0.

Установить RCPG=2, таблица рецептов RCP5~RCP19 будет отображать данные 2 группы рецептов и буфер RCP0~RCP4 будет отображать данные 1 группы рецептов.

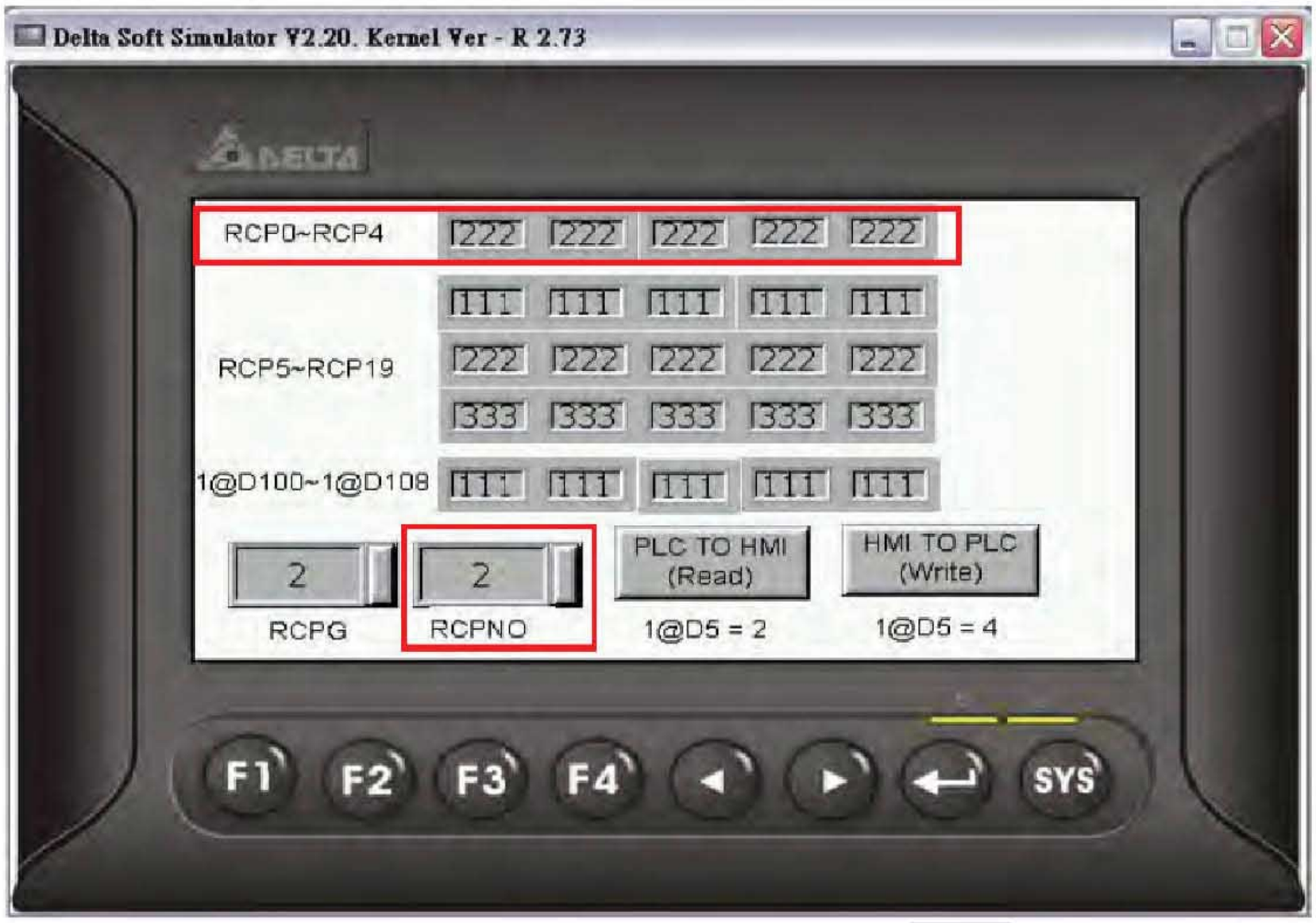
Ниже приведён соответствующий вид экрана.



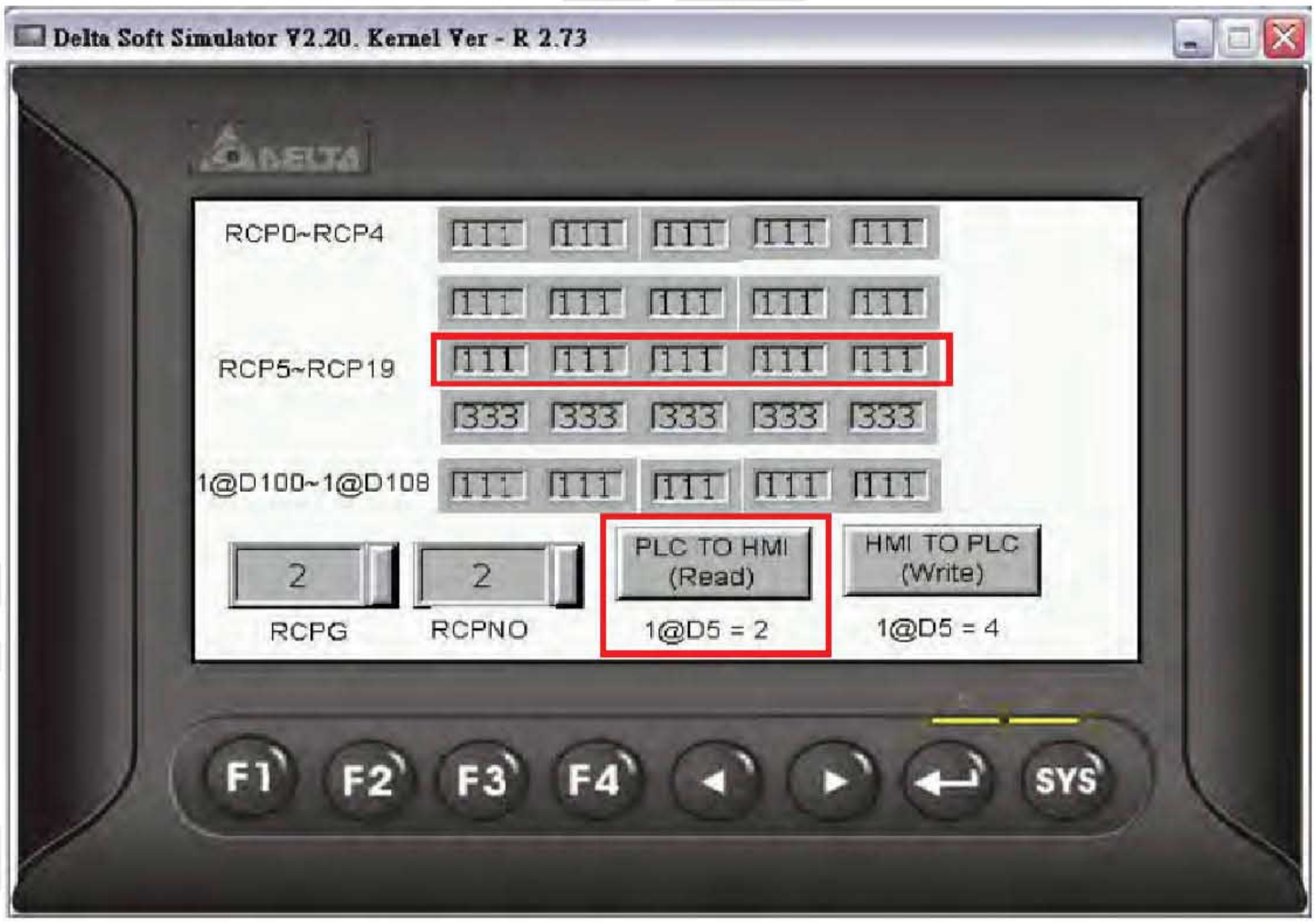
Нажать кнопку **Write** (запись) на экране и данные первой группы рецептов запишутся в контроллер по адресам 1@D100~1@D108. Одновременно, содержимое этих регистров, равное 111, будет отображено на экране



Установить RCPNO = 2, вторая группа рецептов будет отображена в буфере RCP0~RCP4 как показано ниже:



Нажать кнопку **Read** (чтение) и панель оператора прочитает данные из контроллера и сохранит их во 2 группе рецептов.



Далее, значение данных 2-ой группы рецептов будет изменено.

4.3 Как использовать Windows Excel CSV файл

Два способа редактирования рецептов в разных форматах - в виде RCP файла и в виде CSV файла.

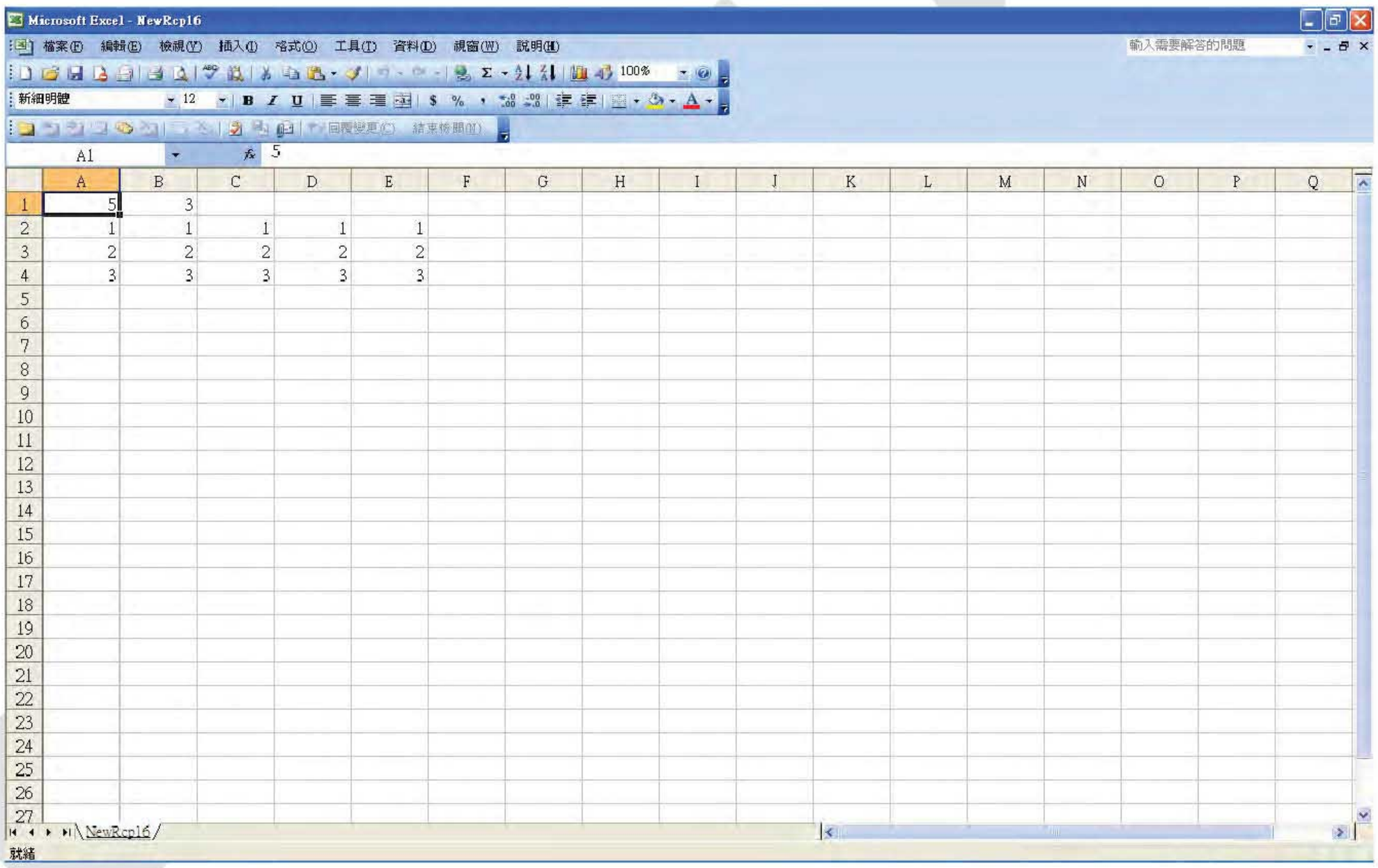
RCP файл может редактироваться только в программе Screen Editor.

CSV файл может редактироваться в программе Microsoft Windows Excel.

В данном разделе описана процедура редактирования CSV файла в программе Microsoft Windows Excel и сохранения отредактированных рецептов в виде файла Windows Excel.

16-битные рецепты

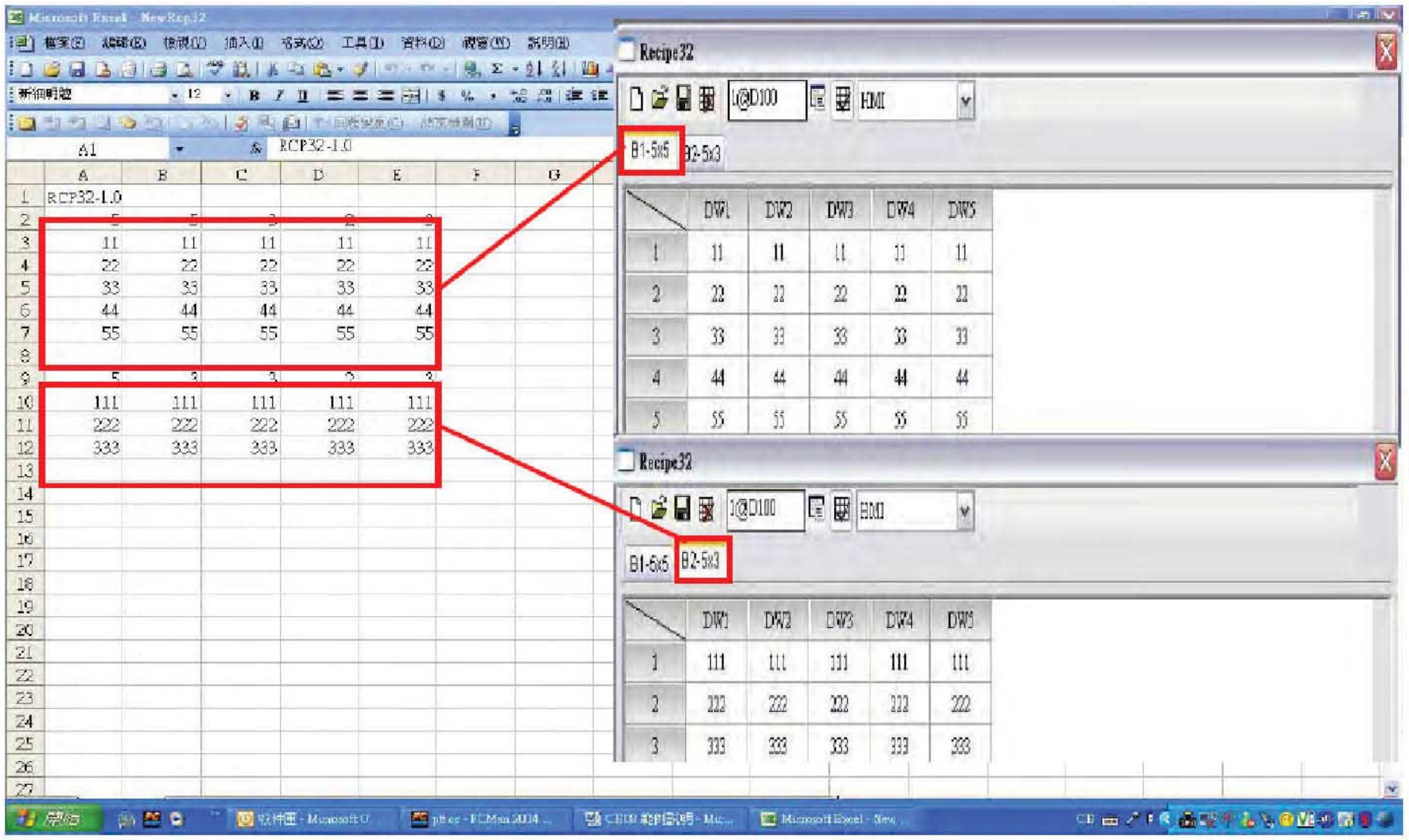
При открытии 16-битного CVS файла рецептов в Microsoft Windows Excel в первой строке записана длина рецепта и число рецептов в группе, в других строках записаны данные рецептов. В данном примере 3 группы по 5 рецептов итого 15 данных.



	A	B	C	D	E
1	5	3			
2	1	1	1	1	1
3	2	2	2	2	2
4	3	3	3	3	3

32-битные рецепты

При открытии 32-битного CVS файла рецептов в Microsoft Windows Excel, в первой строке записана версия 32-битных рецептов, то есть RCP32-1.0.



На экране, с левой стороны показано поле редактирования Microsoft Windows Excel, а с правой - диалоговый экран установки параметров 32-битных рецептов в Screen Editor.

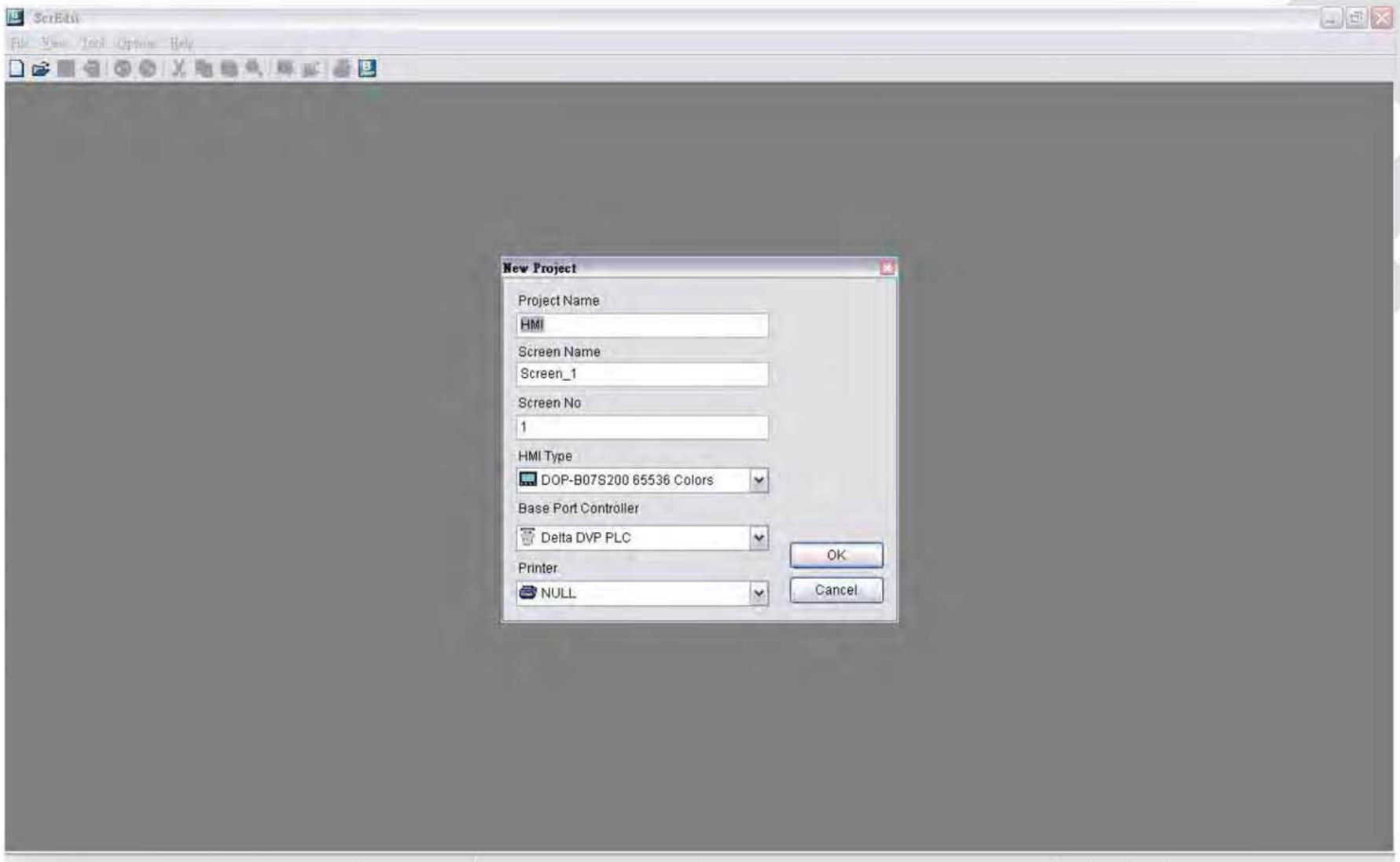
В верхней части этой области располагается данные 1 группы рецептов, а в нижней -данные 2 группы. **Между таблицами должна быть пустая строка**.

В таблице ниже приведены правила соответствия данных в таблице Excel и таблицах рецептов в Screen Editor.

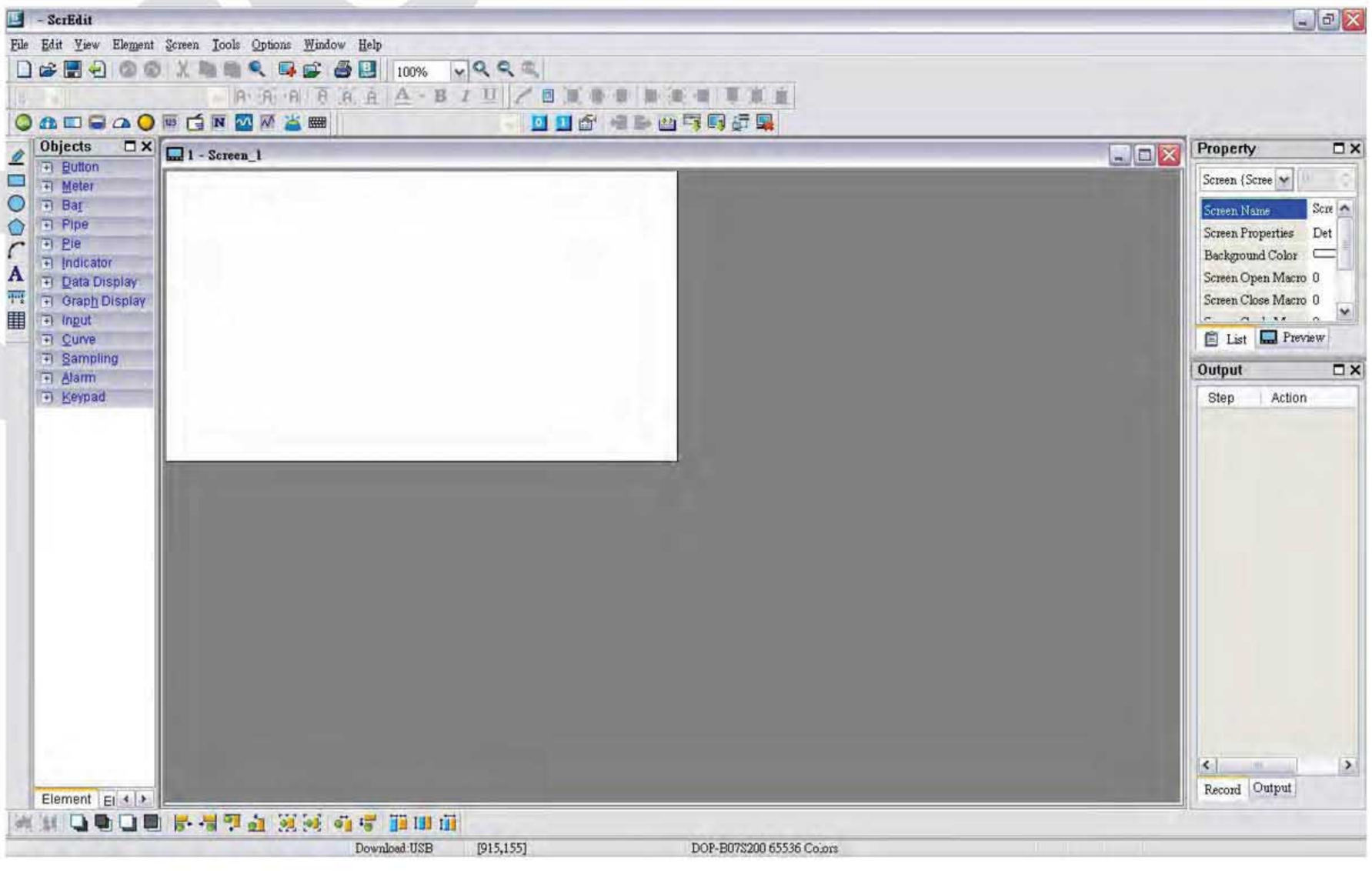
Колонки Excel файла	Соответствующие значения данных рецептов
A	Длина
B	Группа
C	Целые числа (Натуральное десятичное число в формате с плавающей запятой)
D	Дробные числа (Натуральное десятичное число в формате с плавающей запятой)
E	Формат чисел: 2: Десятичный со знаком 3: Десятичный без знака 6: С плавающей запятой

4.4 Как использовать функцию мультиязычной поддержки

Открыть новый проект. Для чего кликнуть мышью значок  или **File > New**. Диалоговый экран приведён ниже

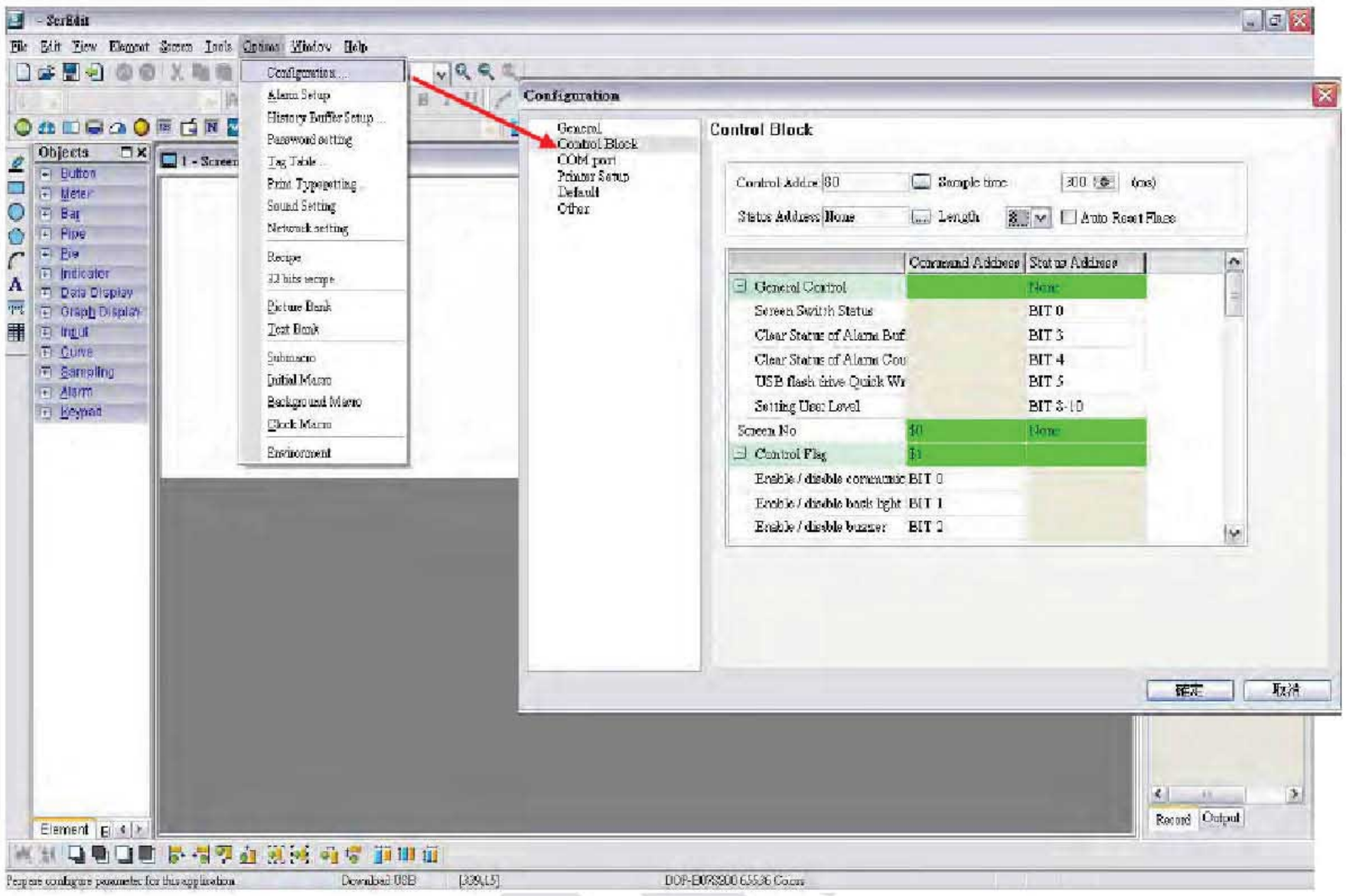


Необходимо ввести наименование проекта, наименование экрана, номер экрана определить тип панели, контроллера или принтера, далее подтвердить выбор, кликнув мышью **OK**. Как показано ниже, в программе **Screen Editor** будет открыт новый проект:



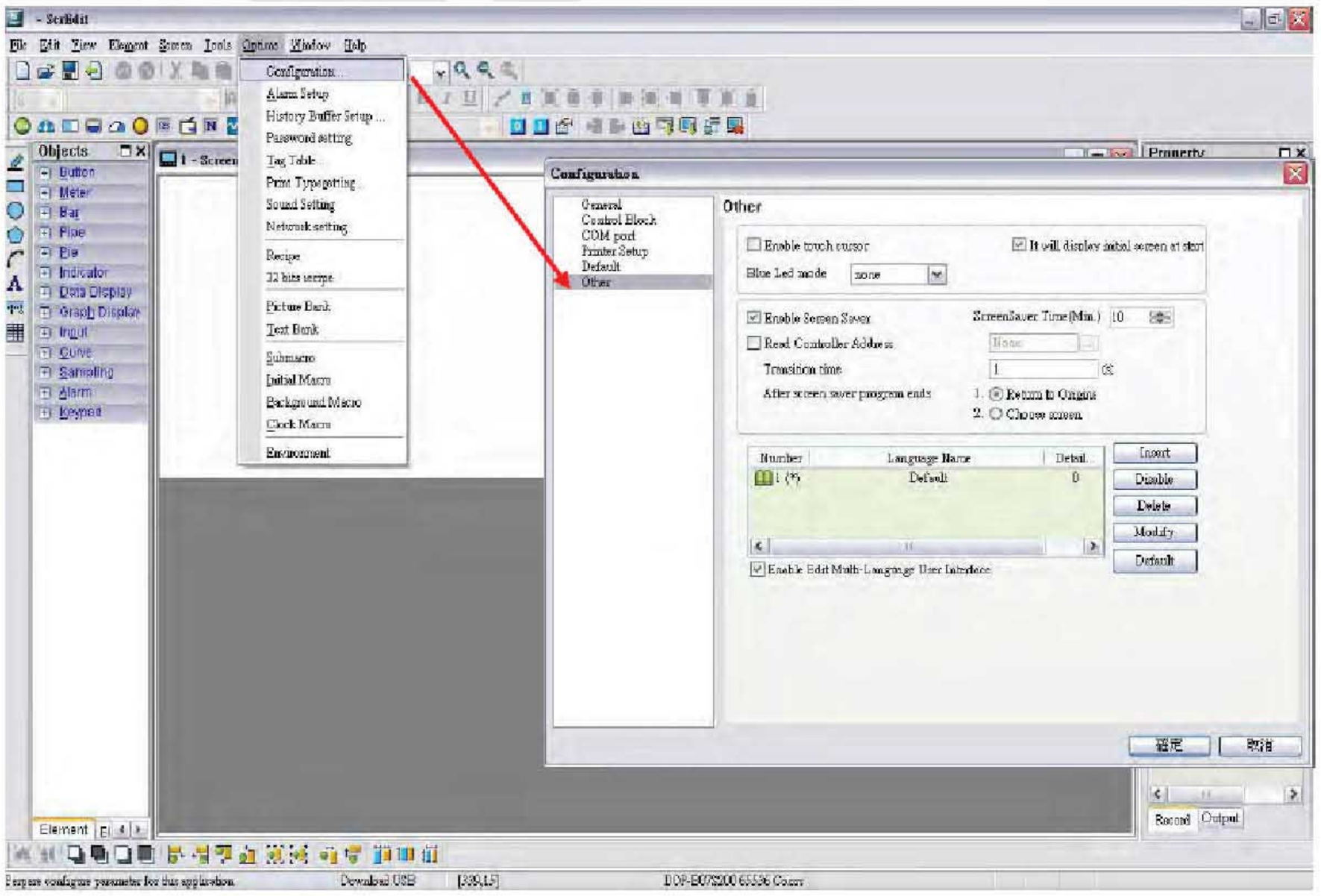
Выбрать **Control Block**, кликнув мышью **Options > Configuration.**

Для управления операциями чтения и записи рецептов, установить адрес блока управления (**Control Address**) 1@D0 и значение длины блока равным 8. Вид экрана приведён ниже:

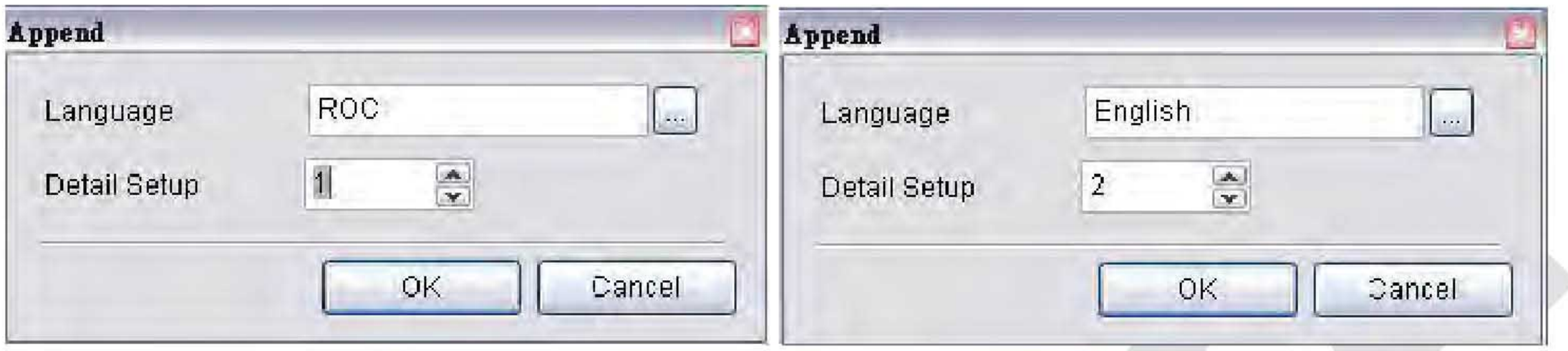


Для завершения настроек блока управления нажать **OK**.

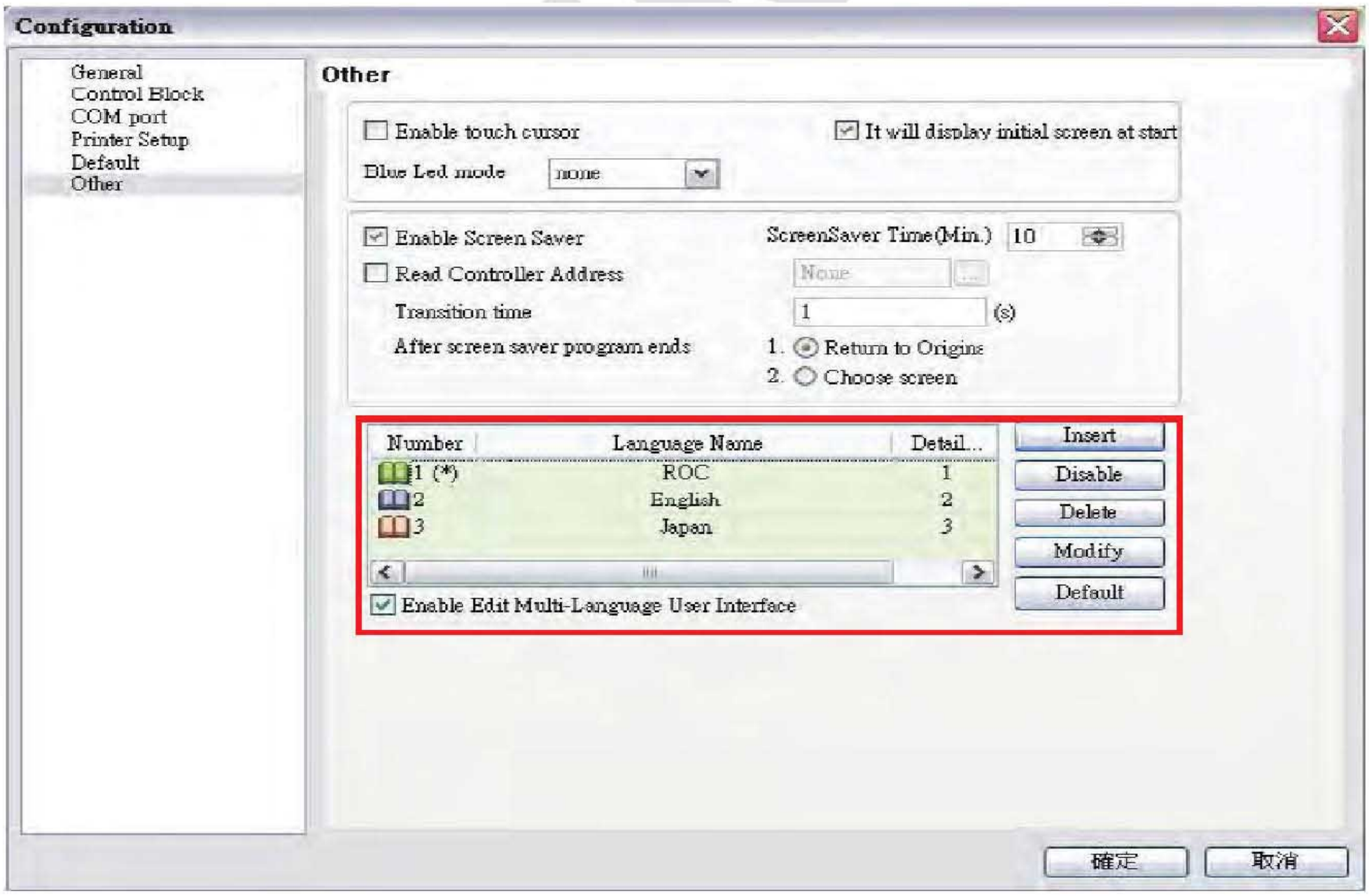
Далее выбрать **Other**, кликнув мышью **Options > Configuration**.



Используя кнопку **Insert** для добавления китайского языка Chinese (ROC), английского English (english), японского Japanese (japan). Для каждого языка записывается соответствующая уставка (1, 2 и 3).

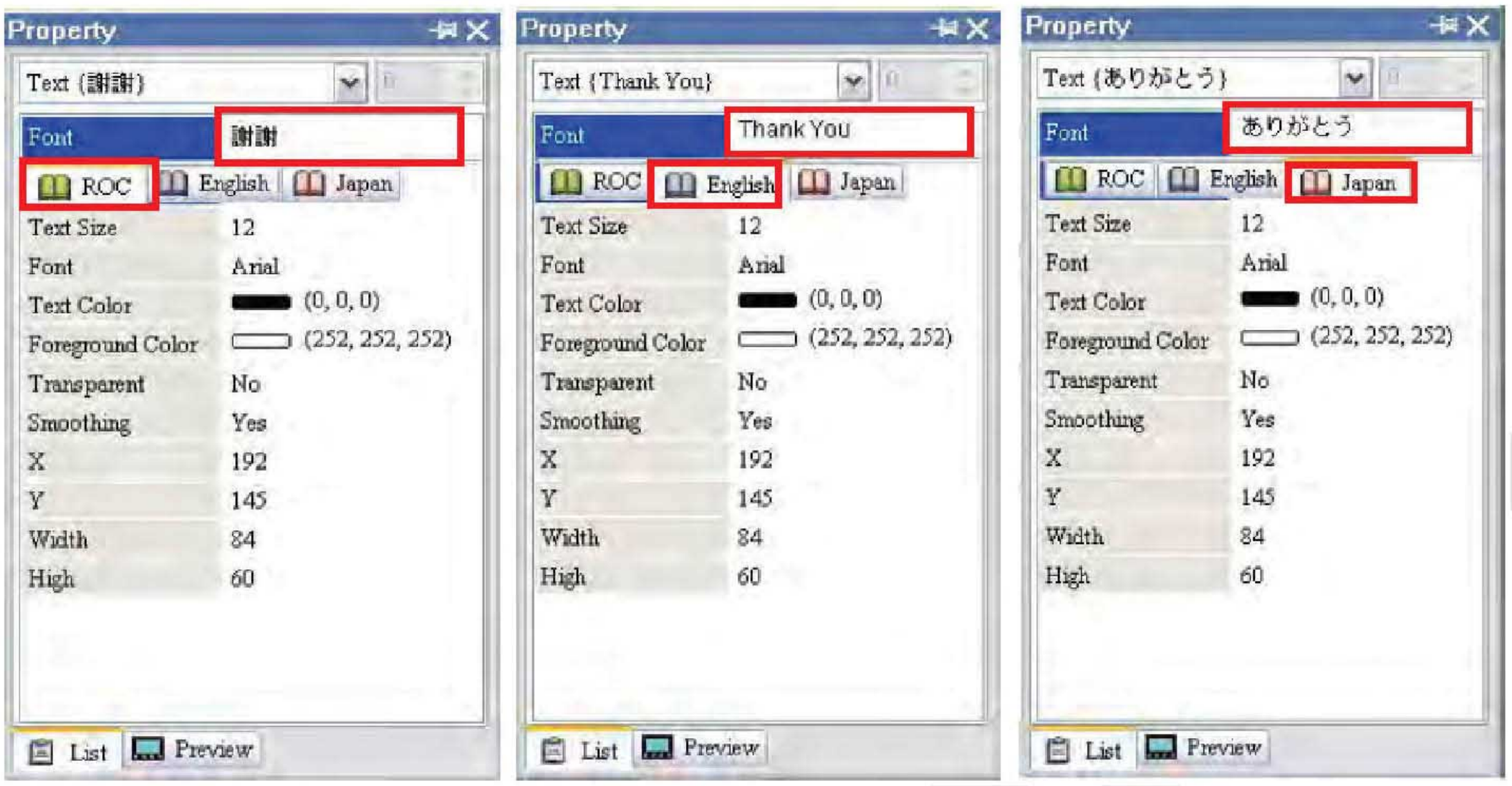


В данном примере по умолчанию установлен Chinese, в таблице показаны и другие языки:



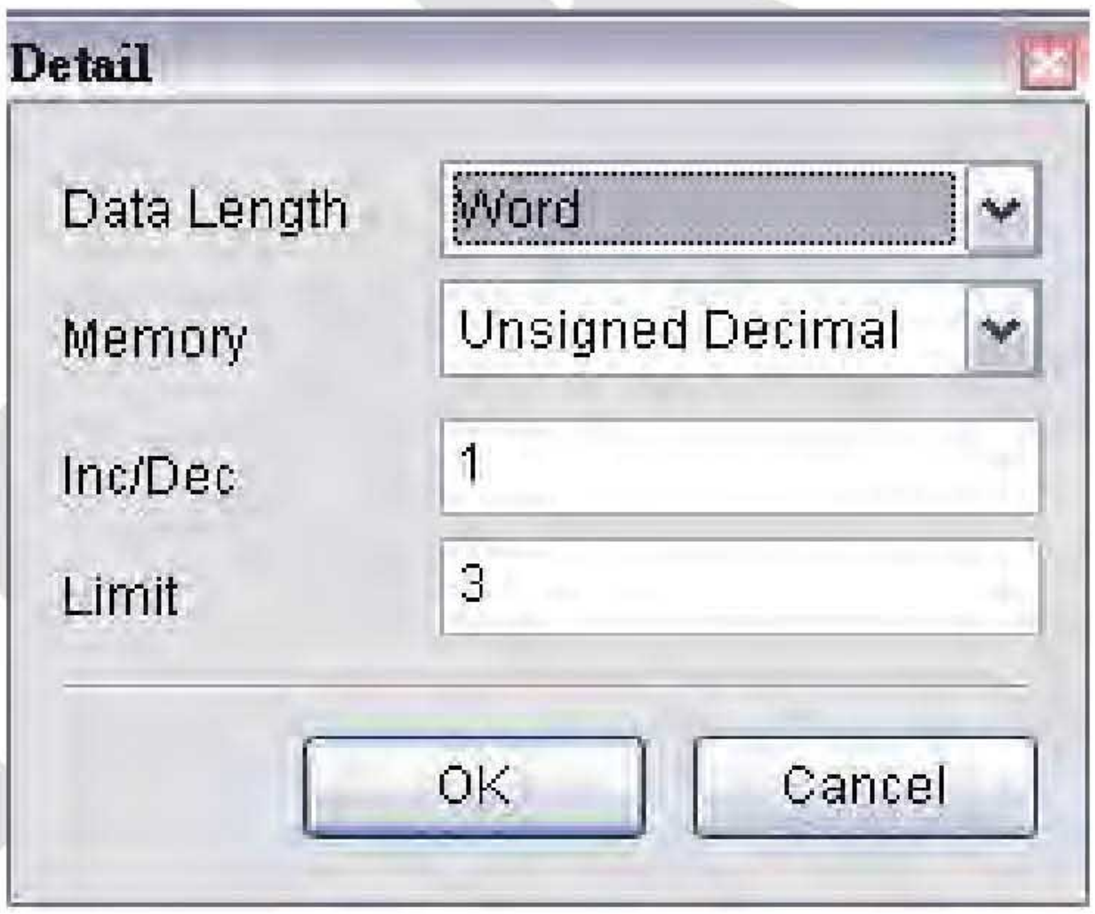
Для завершения настройки нажать **OK**.

Для создания текстового элемента на экране открыть на экране **Text**. Далее, в разделе **Property** записать сообщение на разных языках.



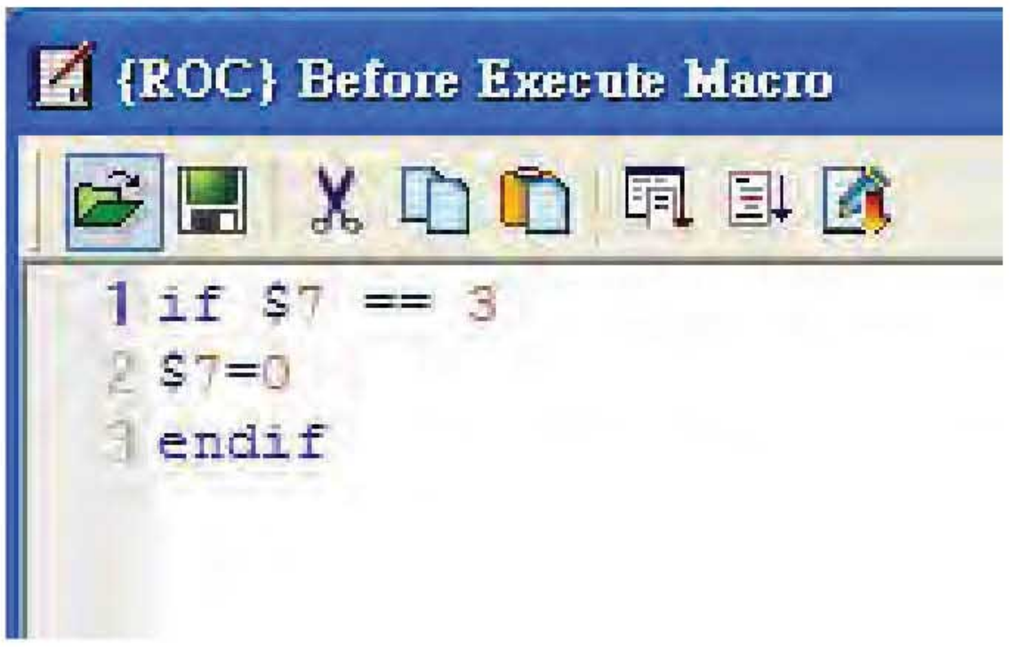
Можно установить пользовательское значение шрифта, цвета, размера.

Дополнительно, создайте на экране элемент-кнопку **Increment**, в свойствах задайте регистр внутренней памяти $7 как адрес записи.

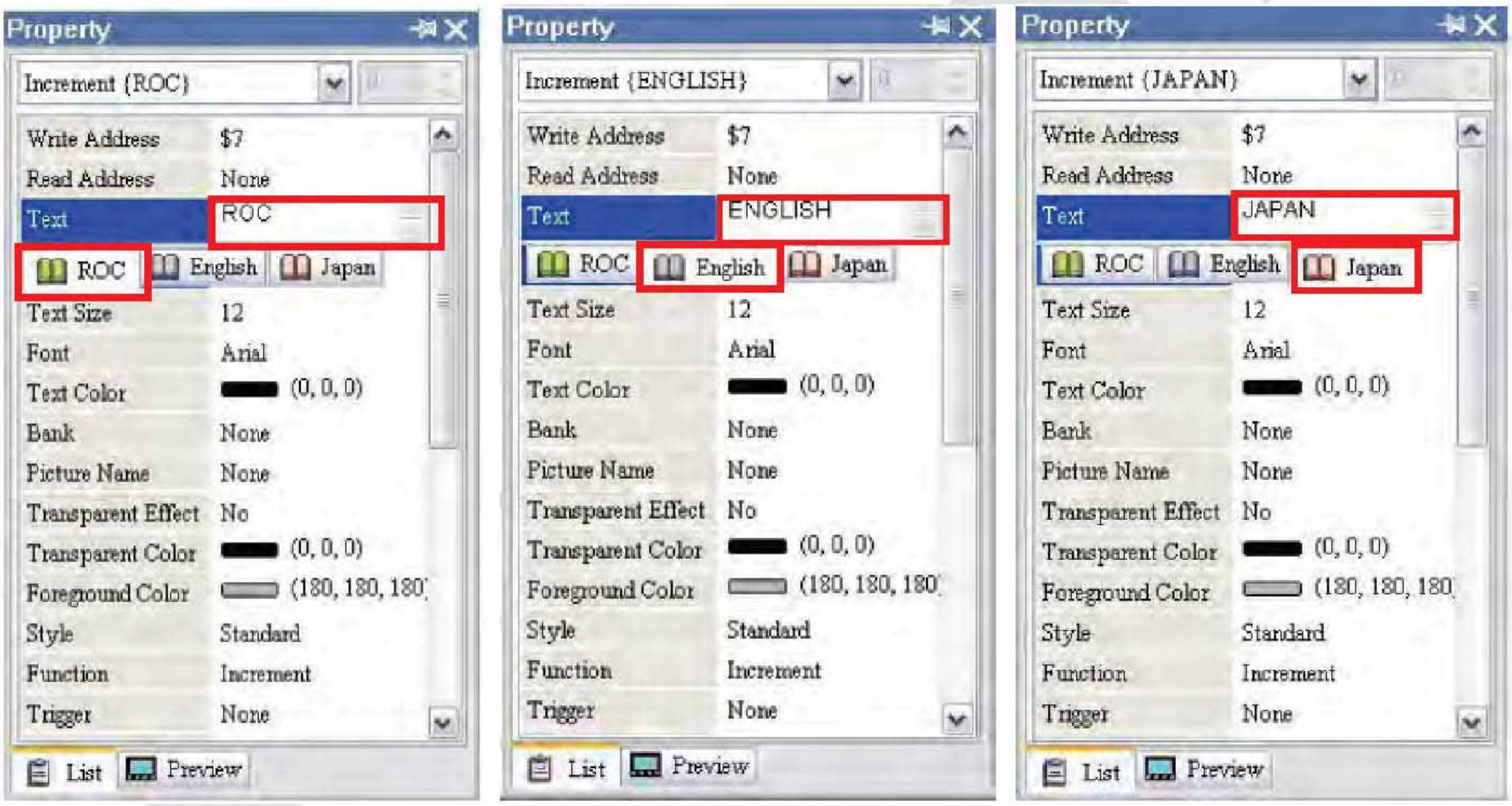


В данном примере используется три языка, поэтому число состояний (**Limit**) установлено на 3.

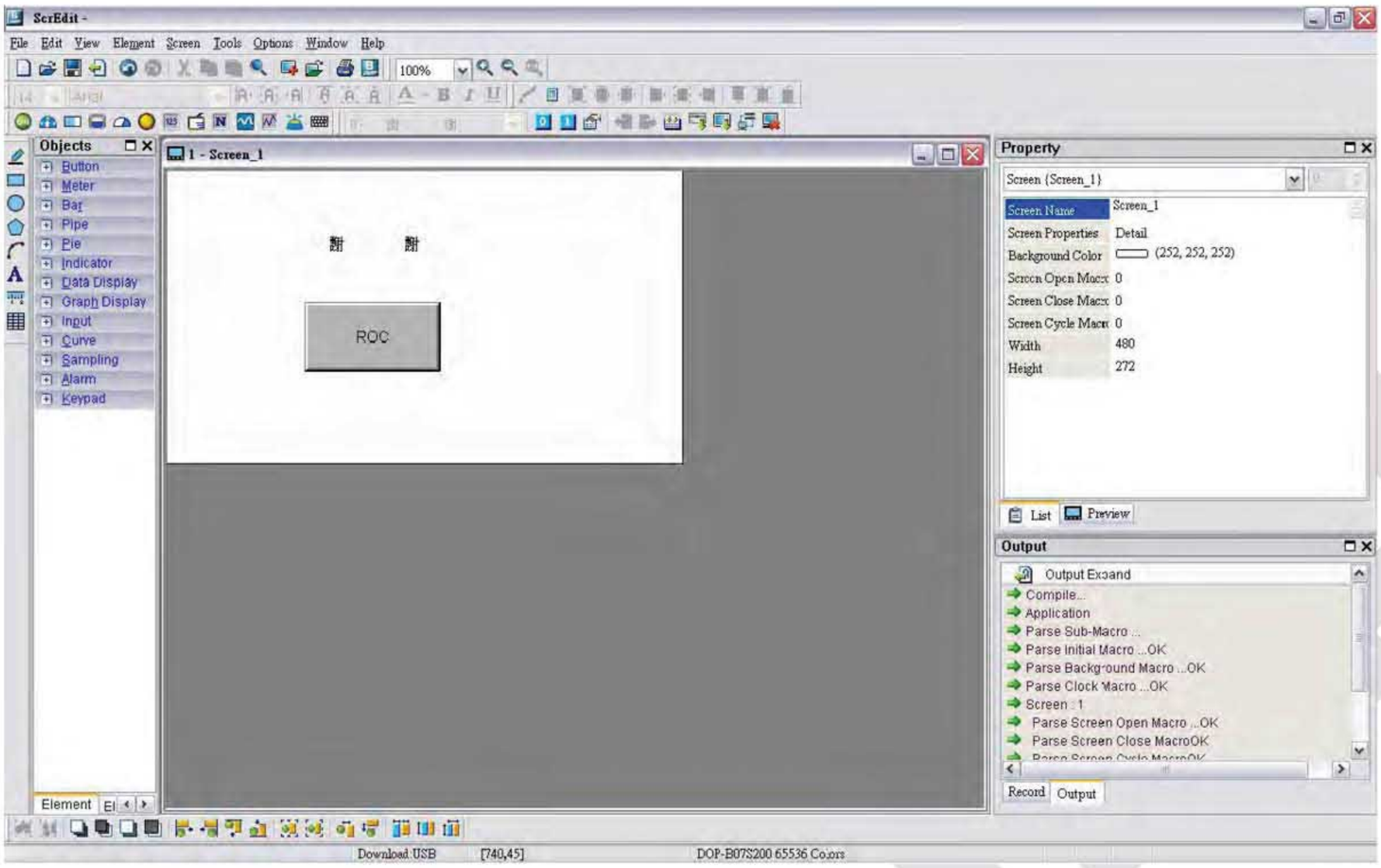
В макросе **Before Execute Macro** задайте настройки как показано ниже. Это обеспечит нужный порядок переключения с одного языка на другой. После переключения на последний язык будет происходить возврат на первый язык.



В настройках элемента-кнопки **Increment**, кликните двойным щелчком мыши по нужному ярлыку языка и введите текст на соответствующем языке в поле **Text**.

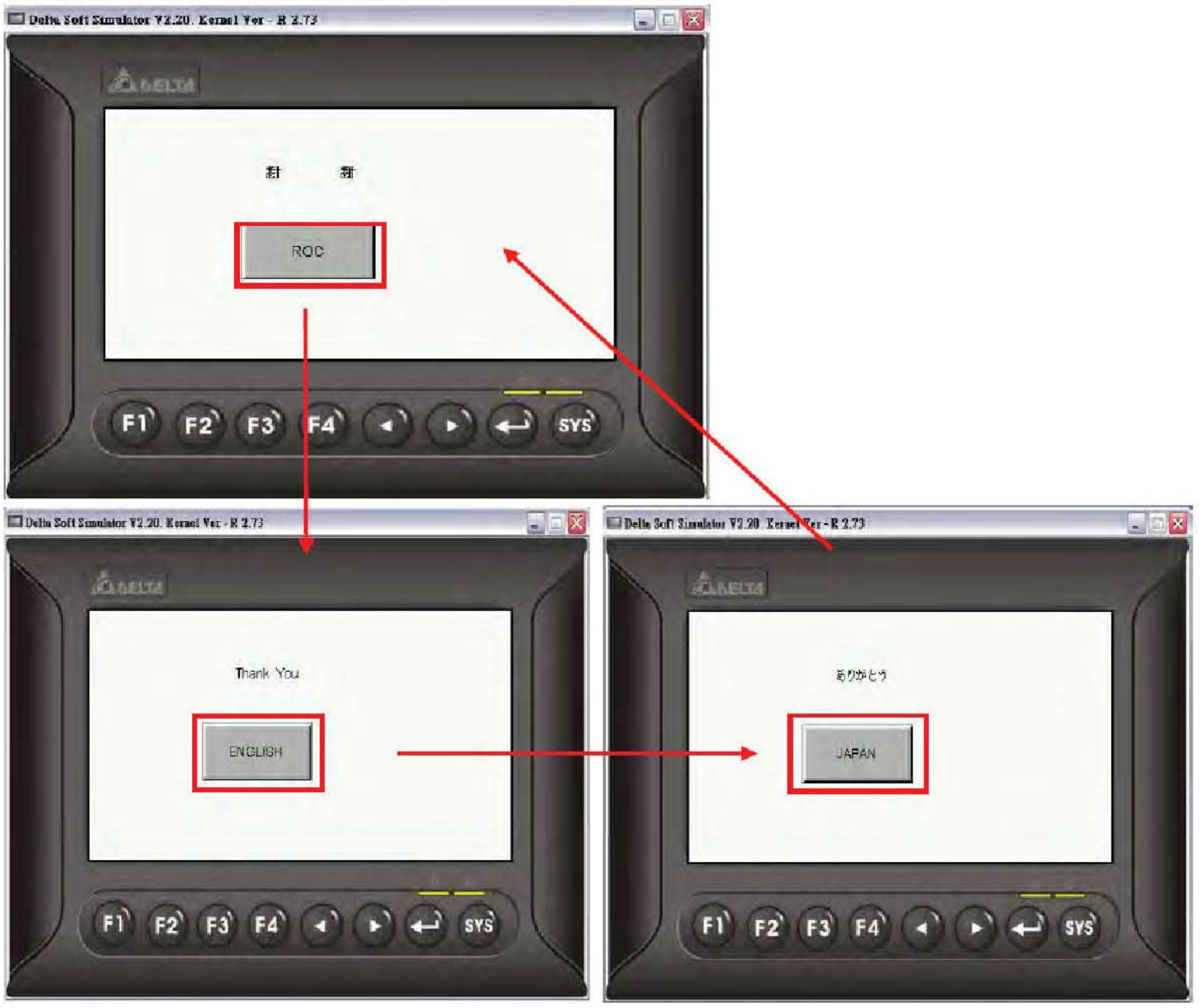


После завершения редактирования пользовательский экран будет выглядеть следующим образом:



Выберите комманду **Compile** для компиляции программы и загрузки её в память панели оператора.

Ниже показано как это будет выглядеть на экране панели оператора. По умолчанию установлен китайский язык. Смена языка происходит при нажатии на экранный элемент-кнопку.

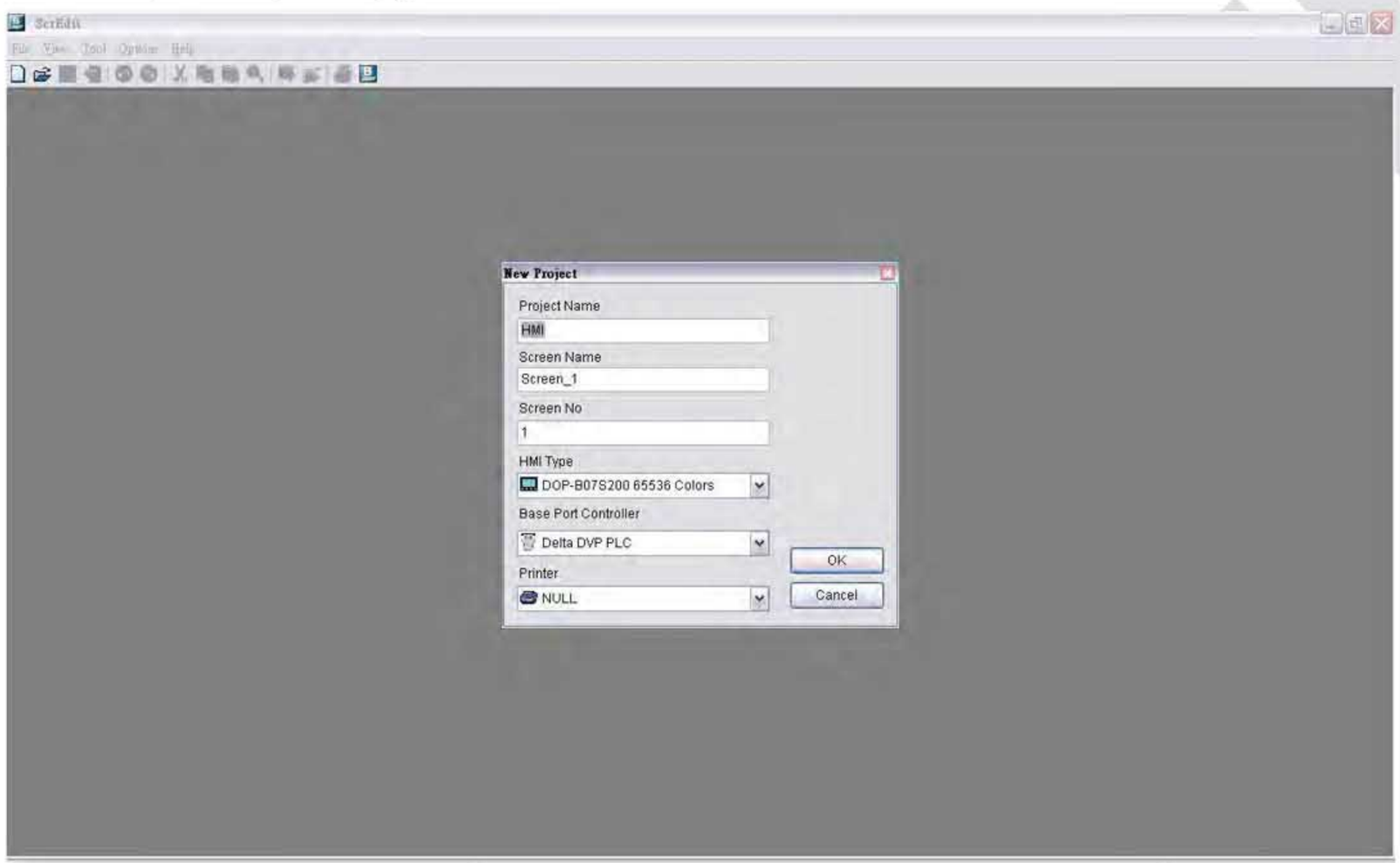


4.5 Как использовать функцию Flash Transfer

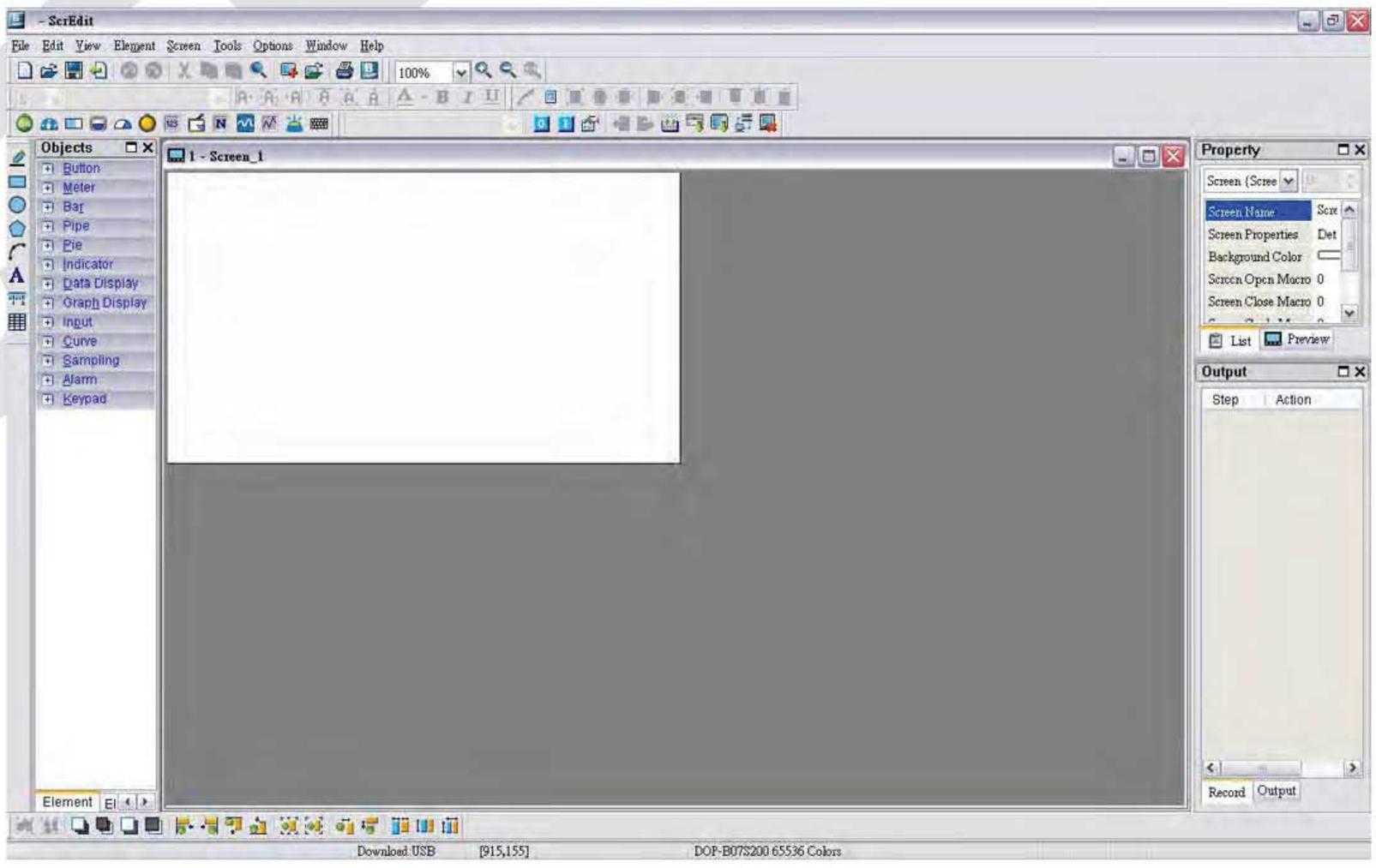
Flash Transfer – этот программа чтения данных из **Historical Trend Graph**, **Historical Data Таблица** и **Alarm data**. В этом случае, Historical Trend Graph and Alarm Historical Таблица используются для пояснения.

Открыть новый проект, для чего кликнуть мышью значок  или **File > New**.

Диалоговый экран приведён ниже

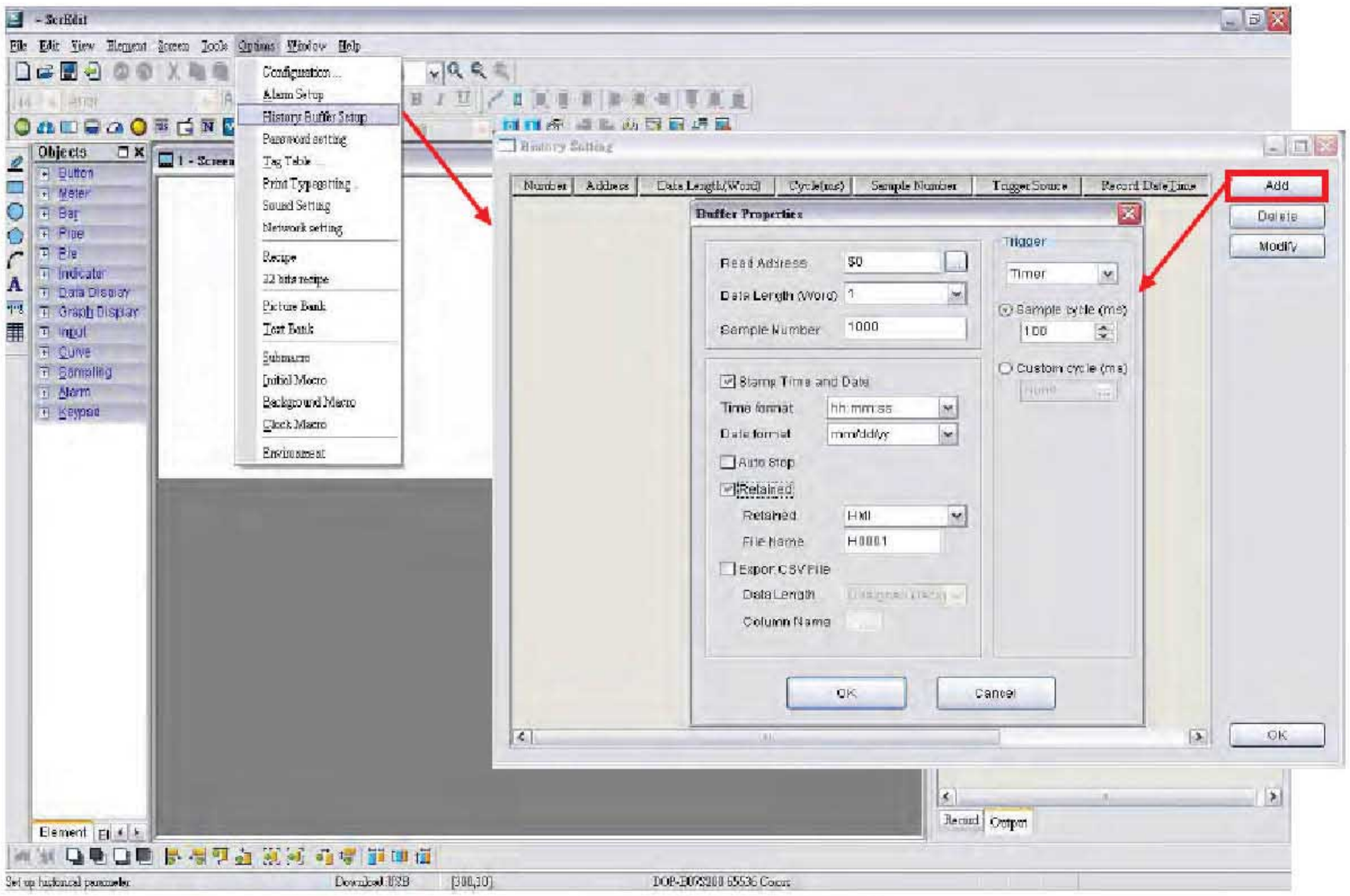


Необходимо ввести наименование проекта, наименование экрана, номер экрана определить тип панели, контроллера или принтера, далее подтвердить выбор, кликнув мышью **OK**. Как показано ниже, в программе **Screen Editor** будет открыт новый проект

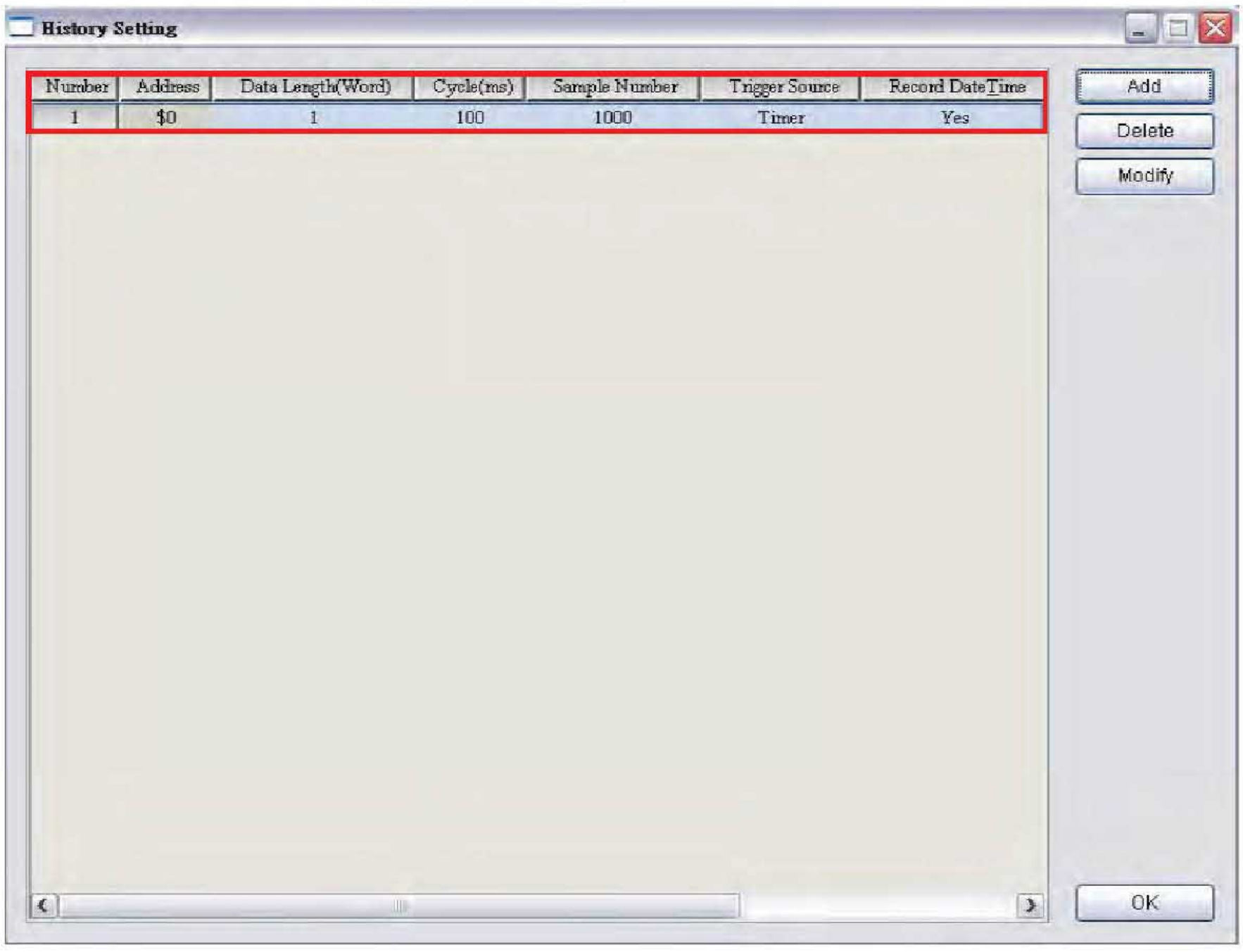


.Для того, чтобы прочитать буфер в данные с соответствующего адреса

контроллера необходимо кликнуть мышью **Options > History Buffer Setup**, установить адрес чтения $0 и выбрать энергонезависимую память панели (**Retained**), как показано ниже:

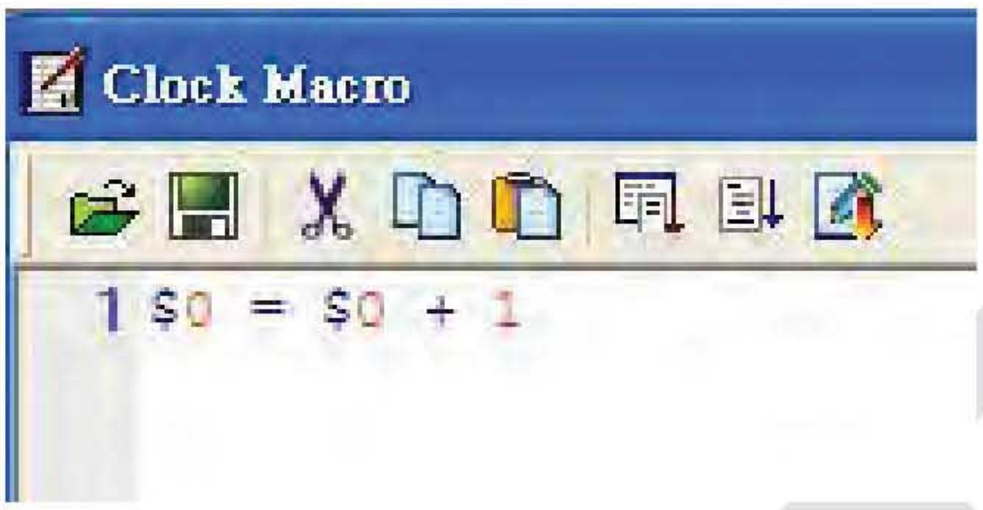


После подтверждения выбора настроек нажатием кнопки **OK** откроется диалоговый экран настроек **History Setting**, вид которого приведён ниже.

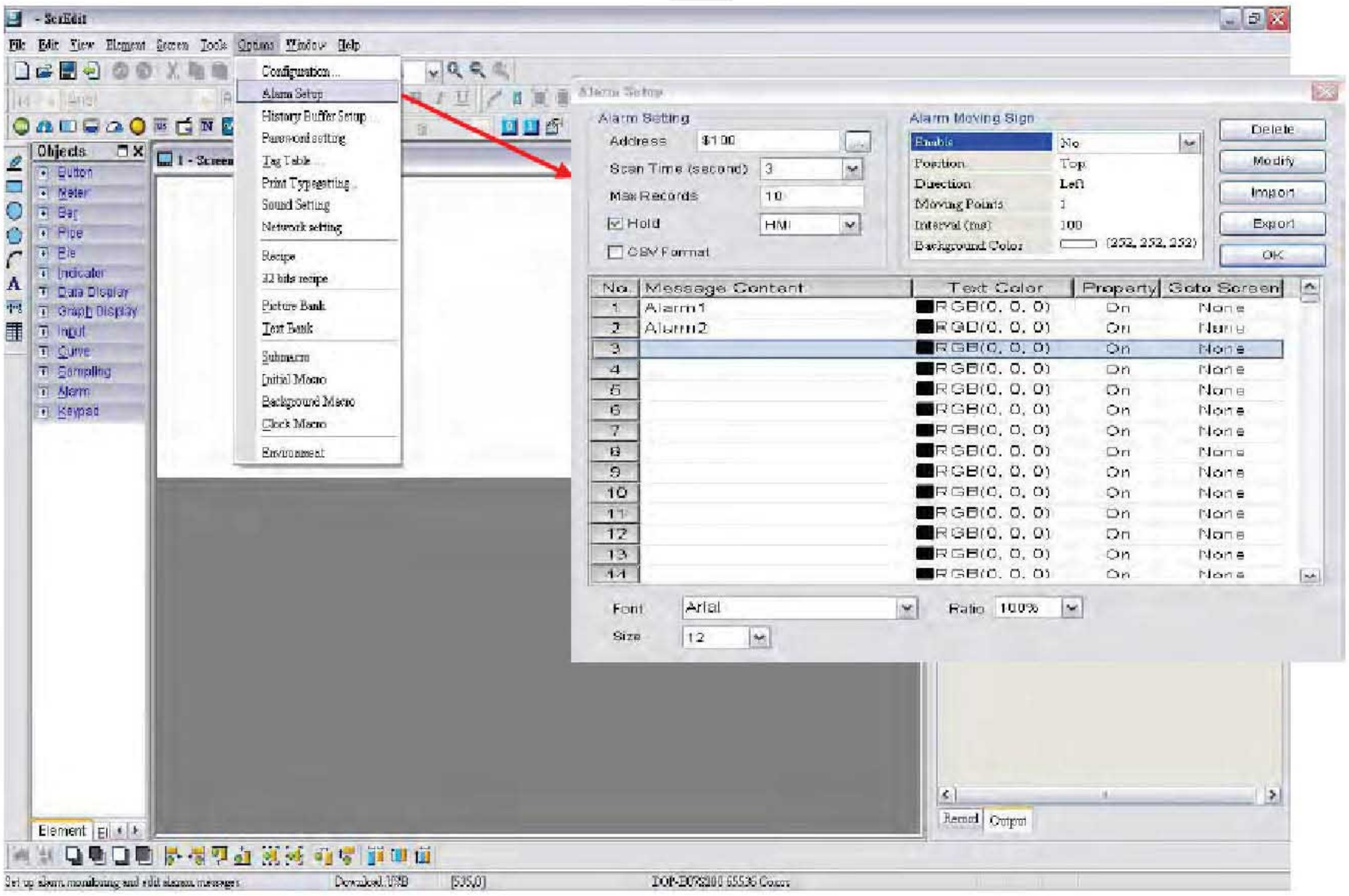


Для подтверждения ввода необходимо нажать кнопку **OK**.

Далее, кликнув мышью **Options > Clock Macro** необходимо войти в редактирование **Clock Macro**. После включения панели оператора циклический макрос выполняется с заданным периодом.

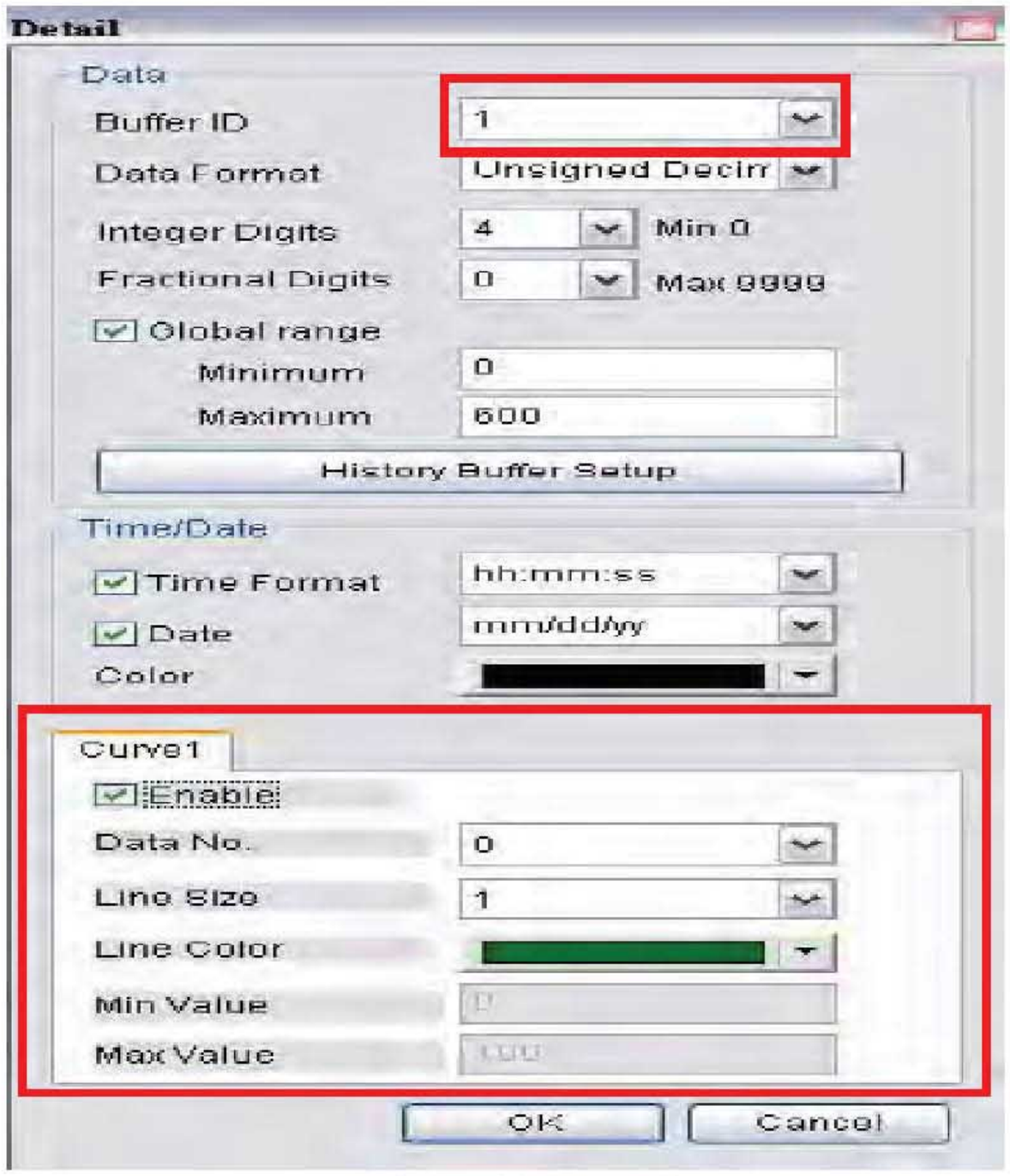


Далее, кликнув мышью **Options > Alarm Setup** установить аварийные сообщения, добавив два сообщения, адресуемые битами внутренней памяти $100.0 и $100.1, установить энергонезависимую память панели для архива аварий, как показано ниже .

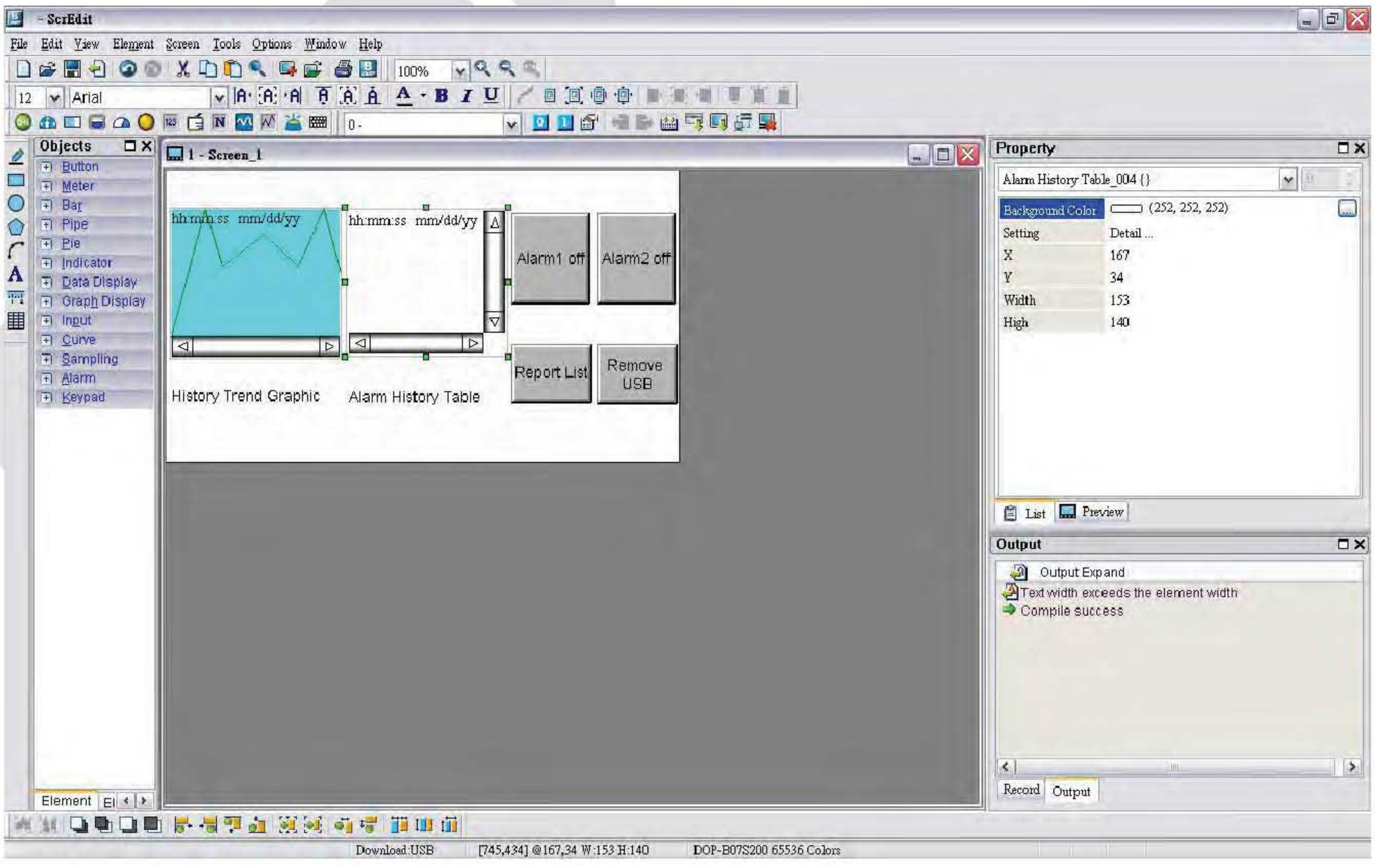


Нажатием кнопки **OK** подтвердить настройки архива аварий.

Открыть экранные объекты **Historical Trend Graph** и **Alarm Data Table**. Завершить настройки **Historical Trend Graph** как показано ниже. Убедитесь, что установлен номер буфера (**Buffer ID**).



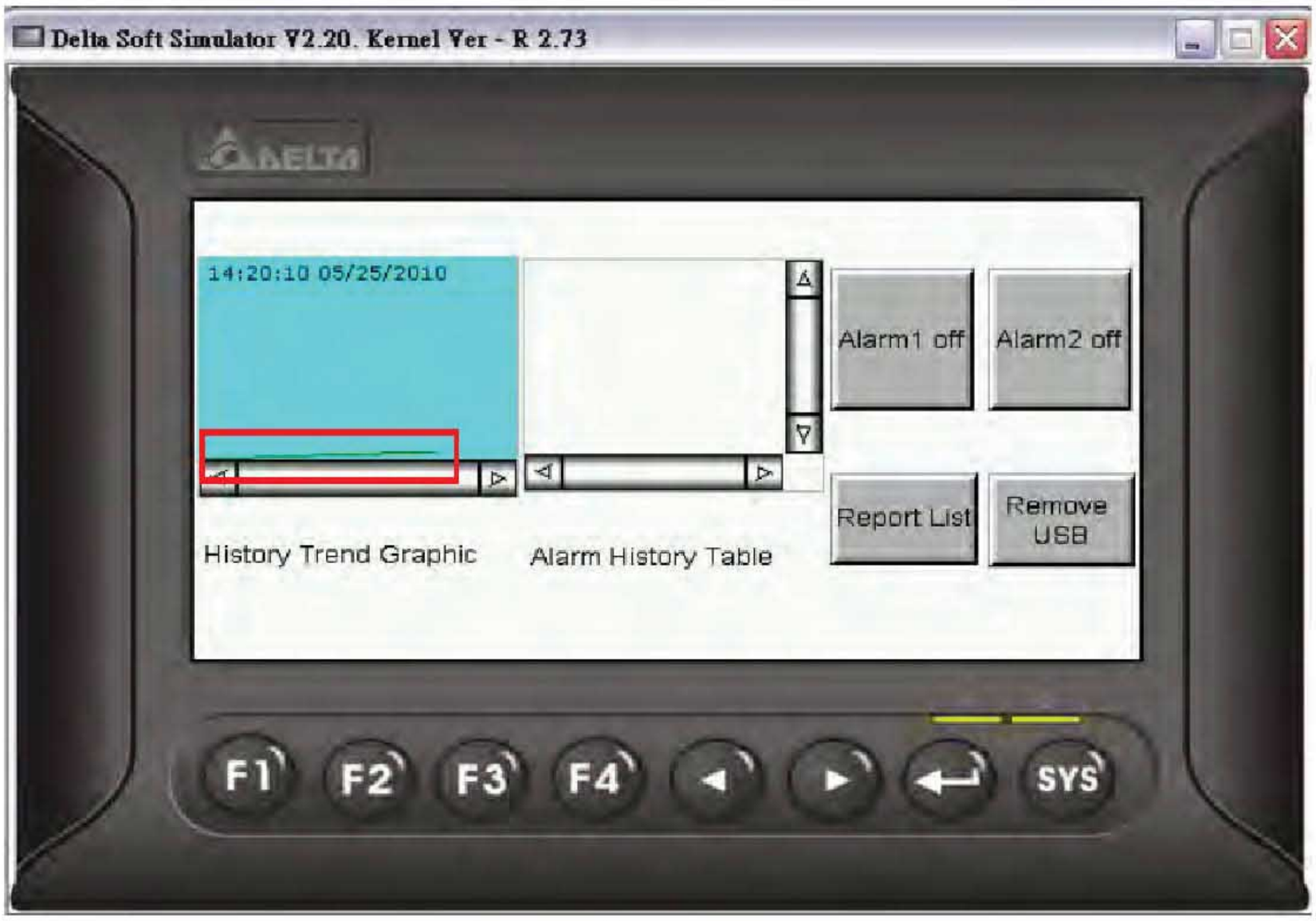
Далее, для переключения аварий **ALARM 1** и **ALARM 2** создать два кнопочных элемента **Momentary**, с адресами соответственно $100.0 and $100.1., создать элементы **Report List** и **Remove storage**. После этого экран будет выглядеть как показано ниже



Произвести компиляцию и загрузить программу в память панели оператора.

Ниже приведён вид экрана панели оператора.

Благодаря работе циклического макроса ($0 = $0 + 1) пользователь может наблюдать все изменения на графиках

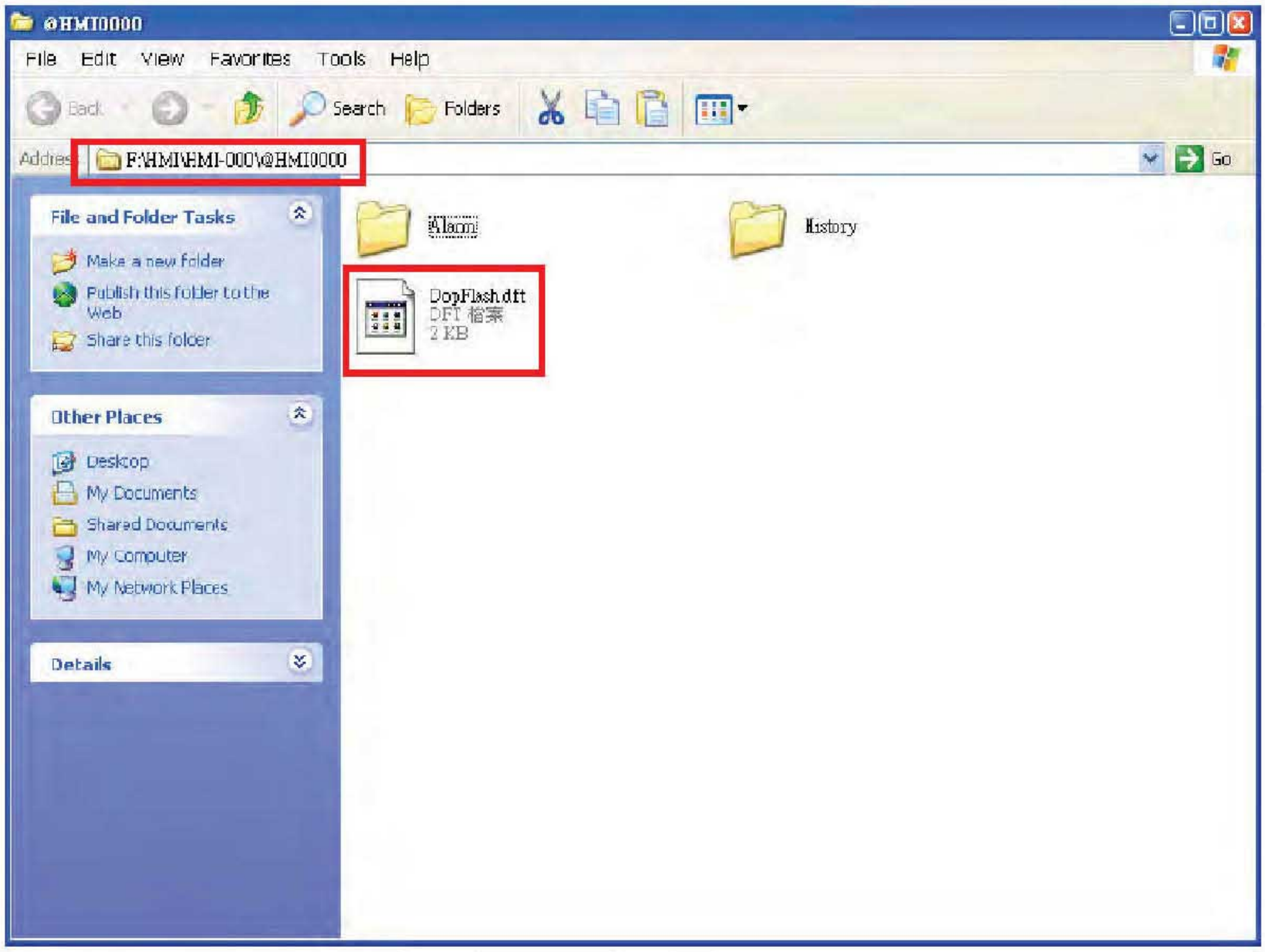


С помощью кнопок **Momentary** можно включать и выключать аварийные сообщения **ALARM 1** и **ALARM 2**, которые будут фиксироваться в **Alarm Data Таблица**

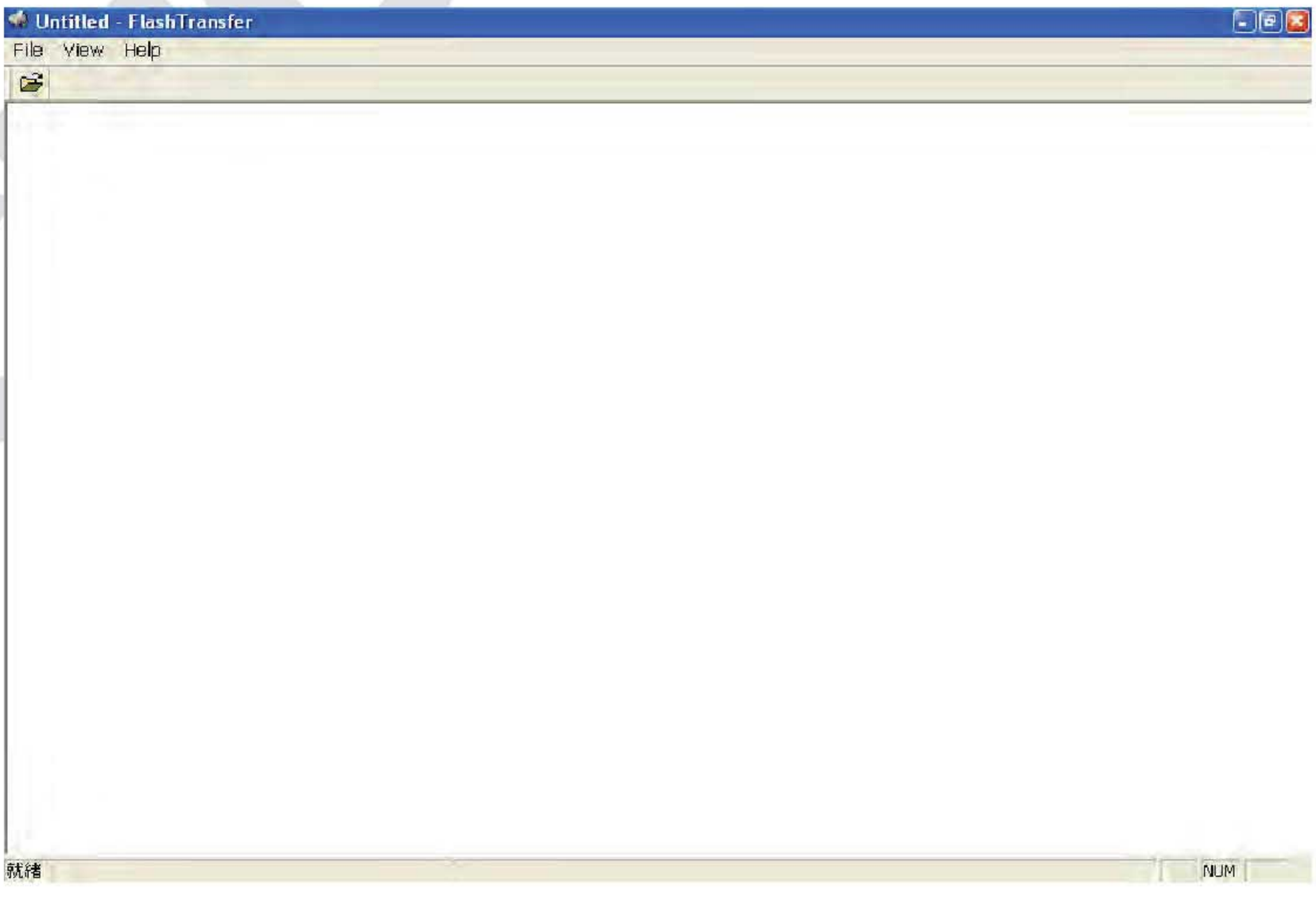
После нажатия экранной кнопки **Report List** в панели произойдёт запись данных архивов в энергонезависимую память на USB диск. После окончания записи необходимо нажать экранную кнопку **Remove storage** и вынуть USB диск из панели

оператора.

Если этот USB диск подключить к компьютеру, то пользователь увидит на нём файл **DopFlash.dft**, который может быть открыт с помощью функции **Flash Transfer**



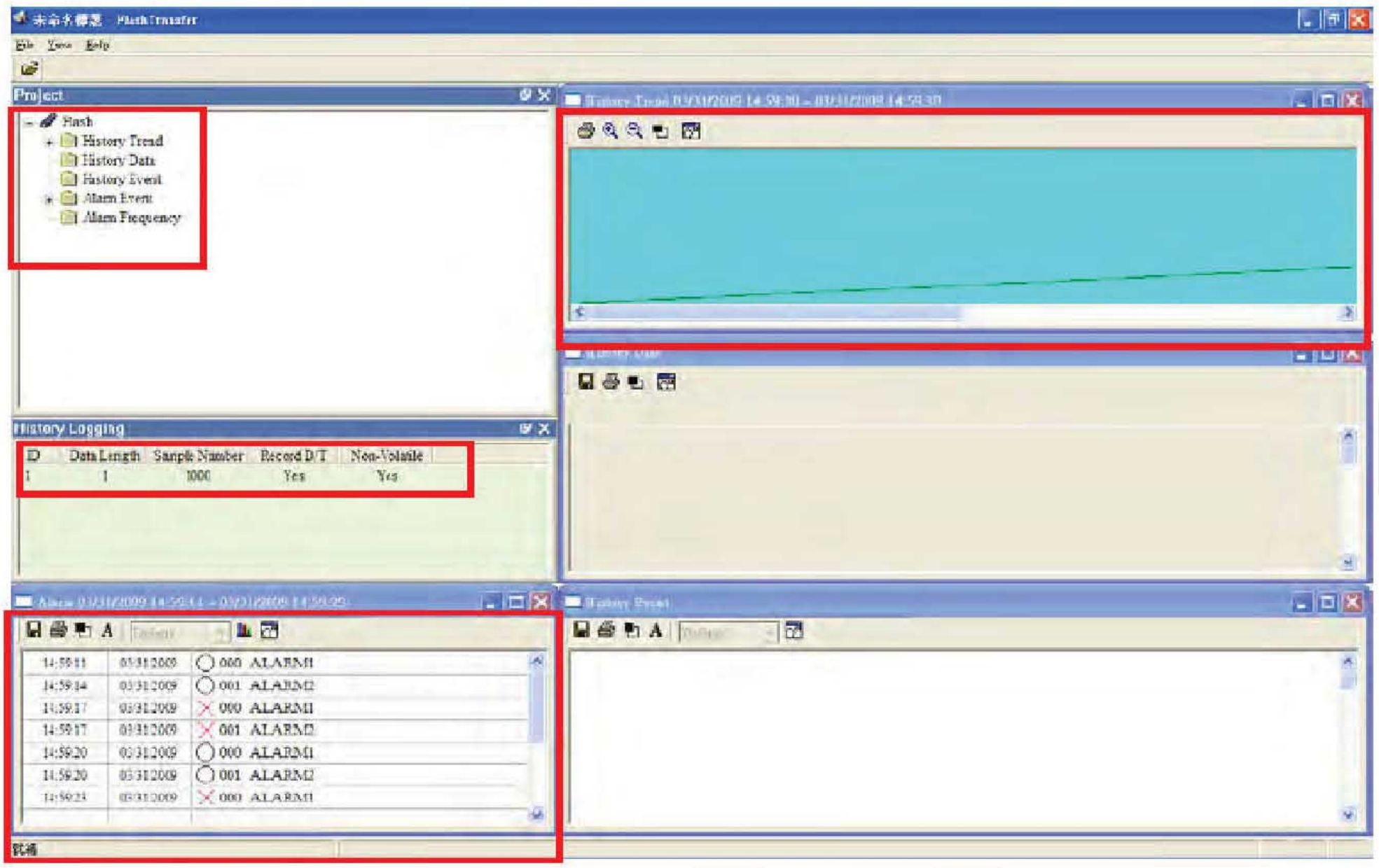
Для реализации функции **Flash Transfer**, необходимо выбрать **Start > Programs > Delta Industrial Automation > HMI > Screen Editor 2.00.07 > Flash Transfer** и откроется экран приведённый ниже.



Кликнуть мышью кнопку  , откроется диалоговый экран

Выбрать папку с **DopFlash.dft**.

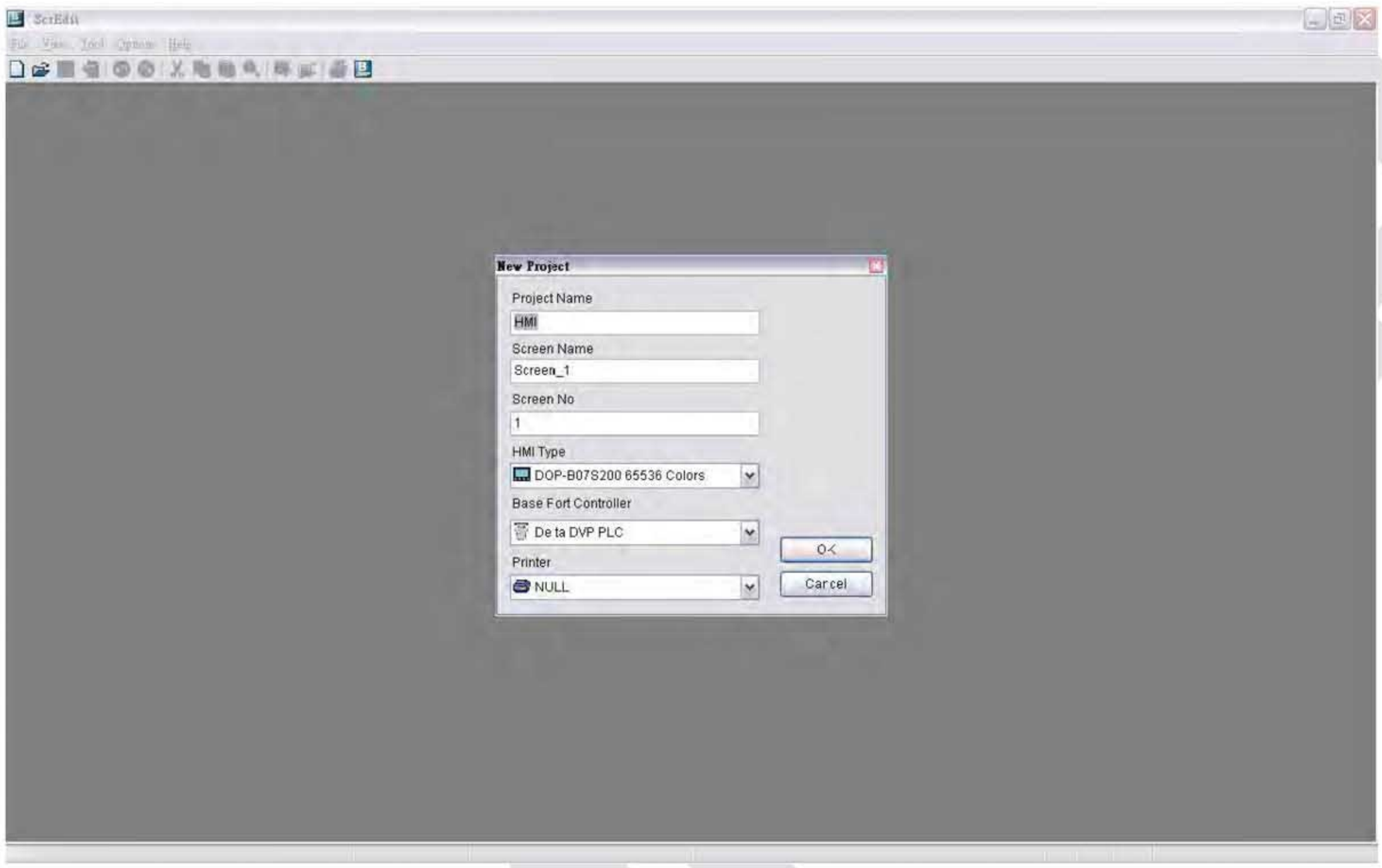
Кликнув **OK** появляется экран с архивами событий и аварий.



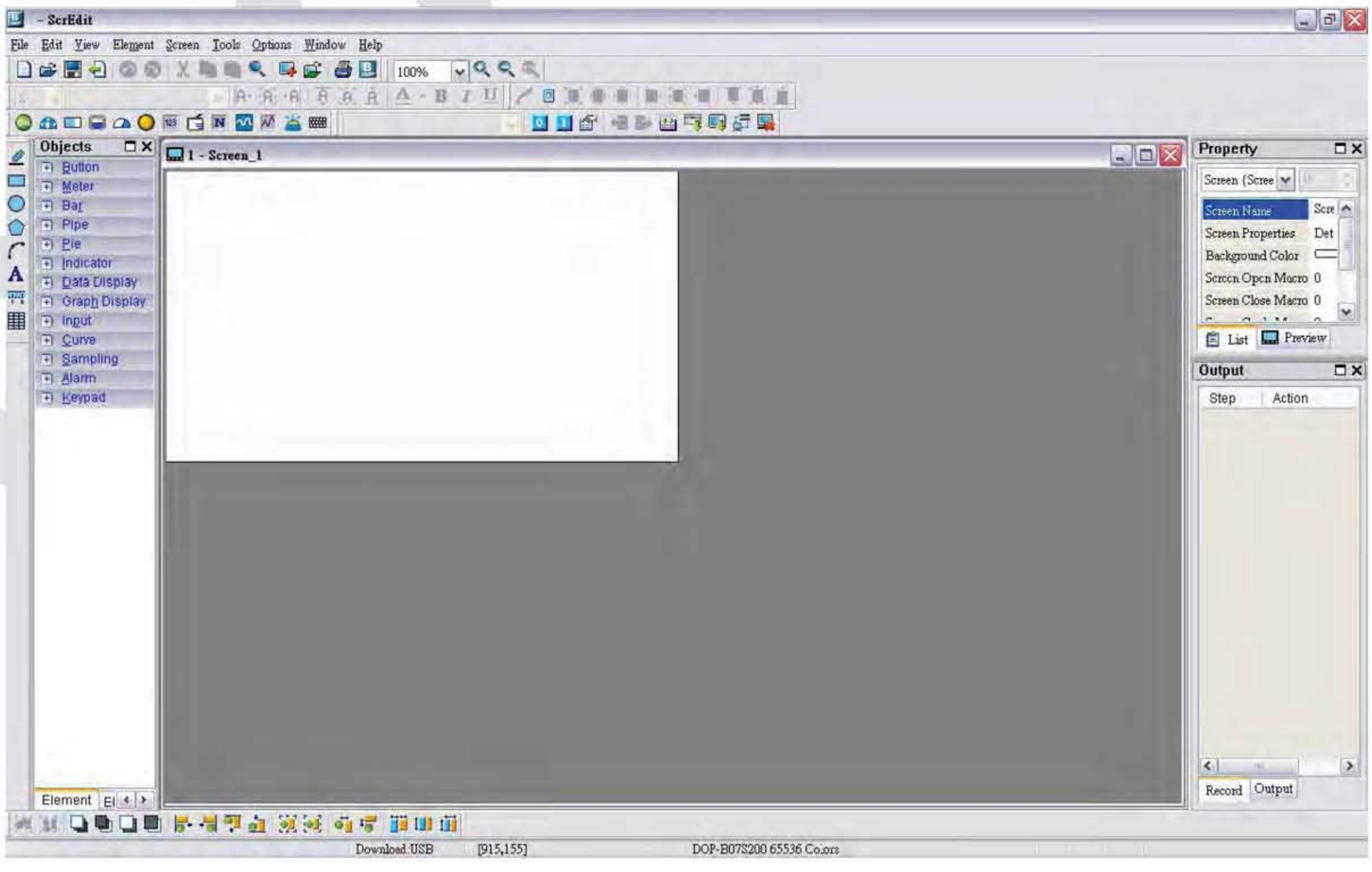
Используя функцию **Flash Transfer** пользователь получает доступ для чтения архивов событий и аварий

4.6 Как использовать объект Real Image

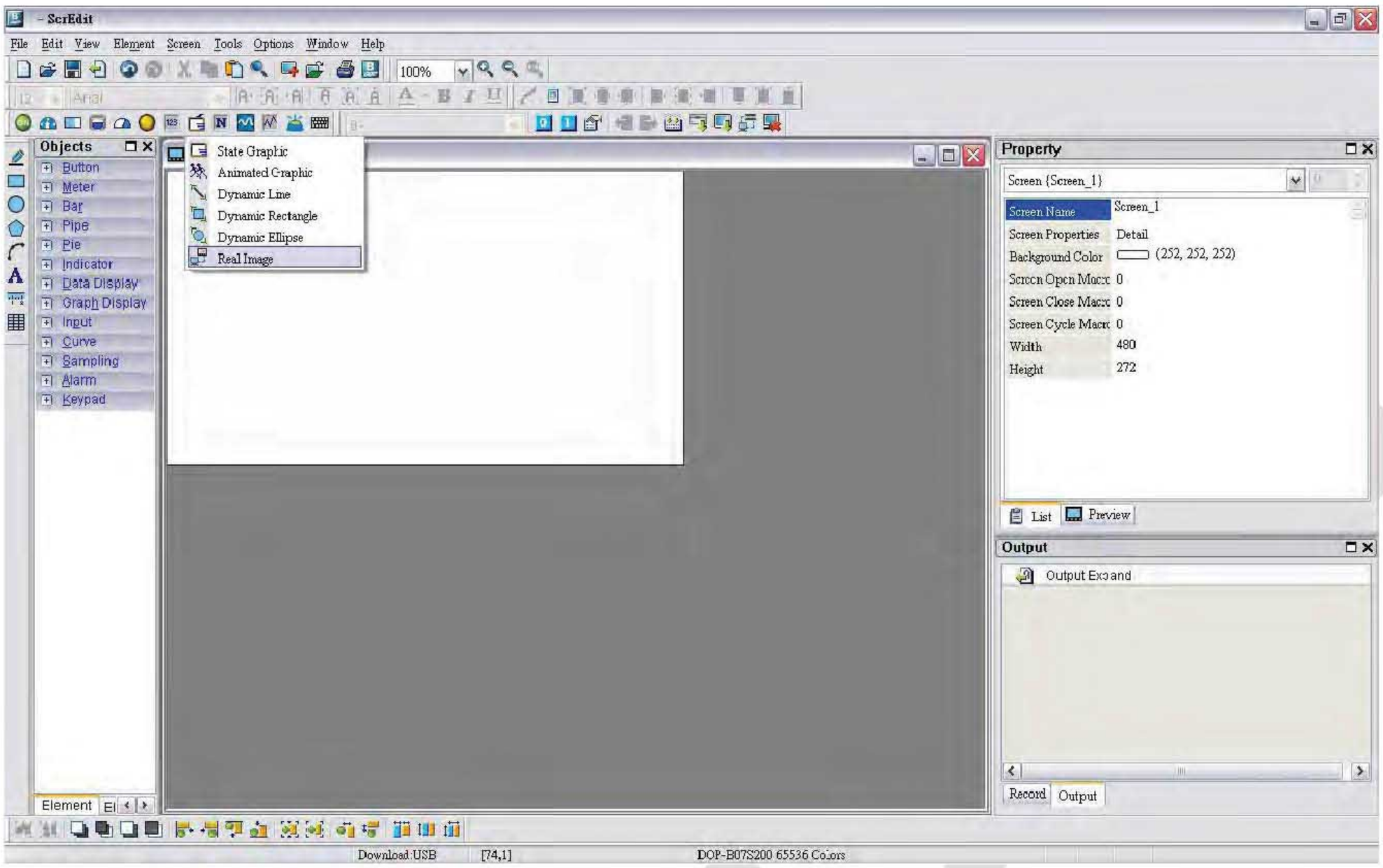
Открыть новый проект, для чего кликнуть мышью значок  или **File > New**. Диалоговый экран приведён ниже



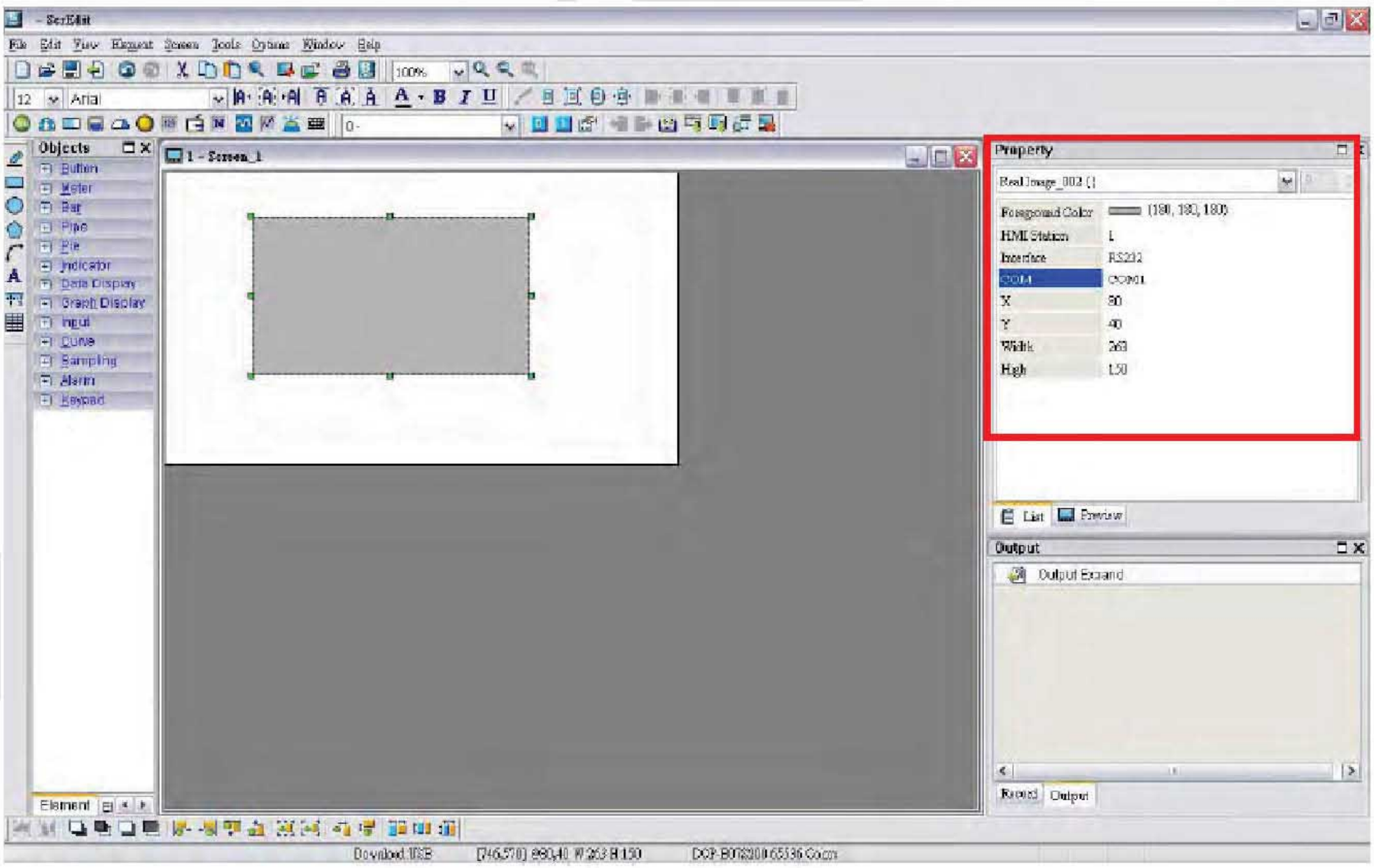
Необходимо ввести наименование проекта, наименование экрана, номер экрана определить тип панели, контроллера или принтера, далее подтвердить выбор, кликнув мышью **OK**. Как показано ниже, в программе **Screen Editor** будет открыт новый проект



Открыть экранный объект **Real Image**.



В свойствах установить параметры коммуникации: **COM** на COM1, **Interface** на RS232.



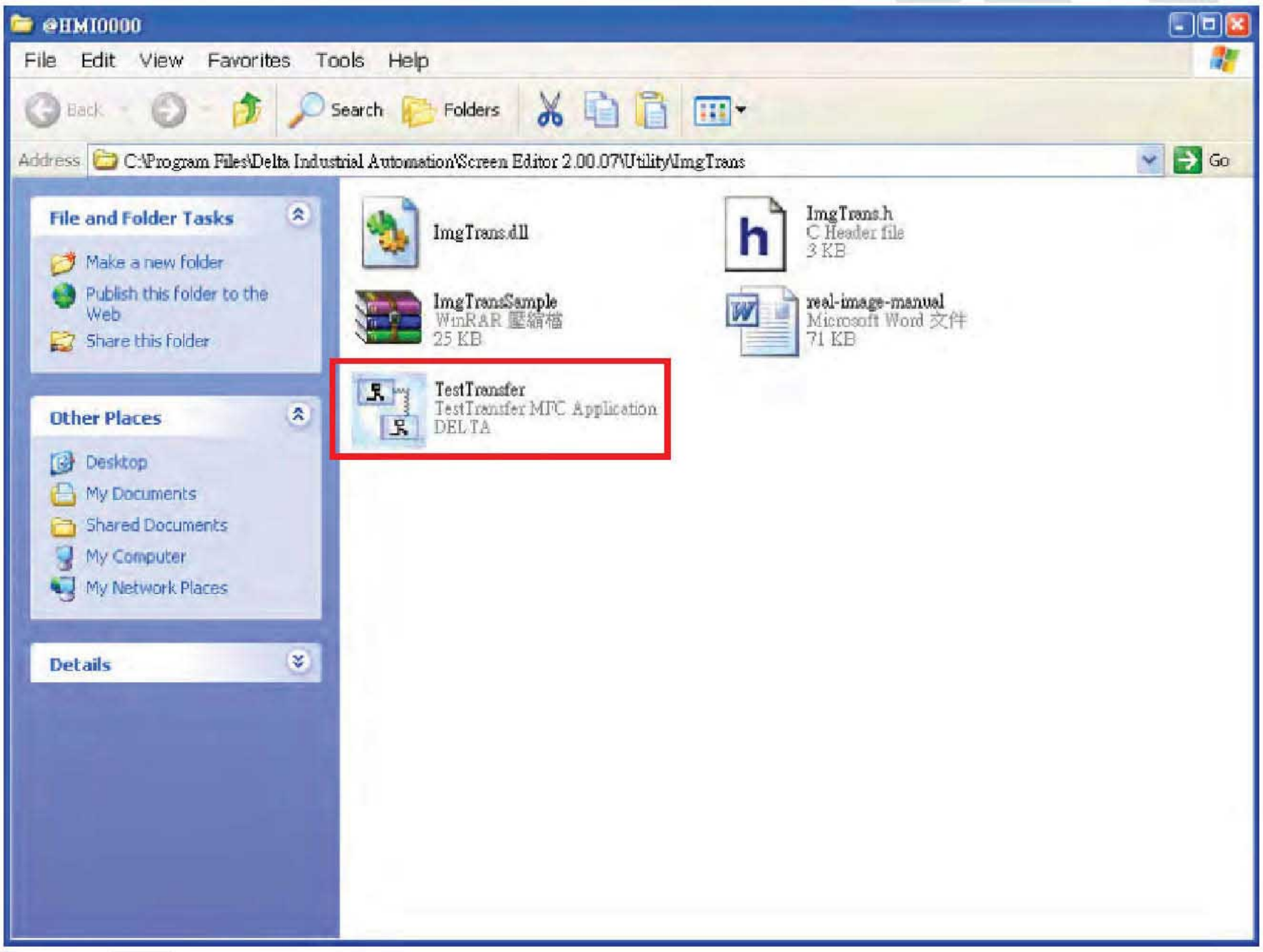
Далее, произвести компиляцию программы и загрузить её в память панели оператора **Tools > Compile**

Выбрать картинку, которую необходимо загрузить, например:

для чего нужна специальная программа **TestTransfer.exe**.

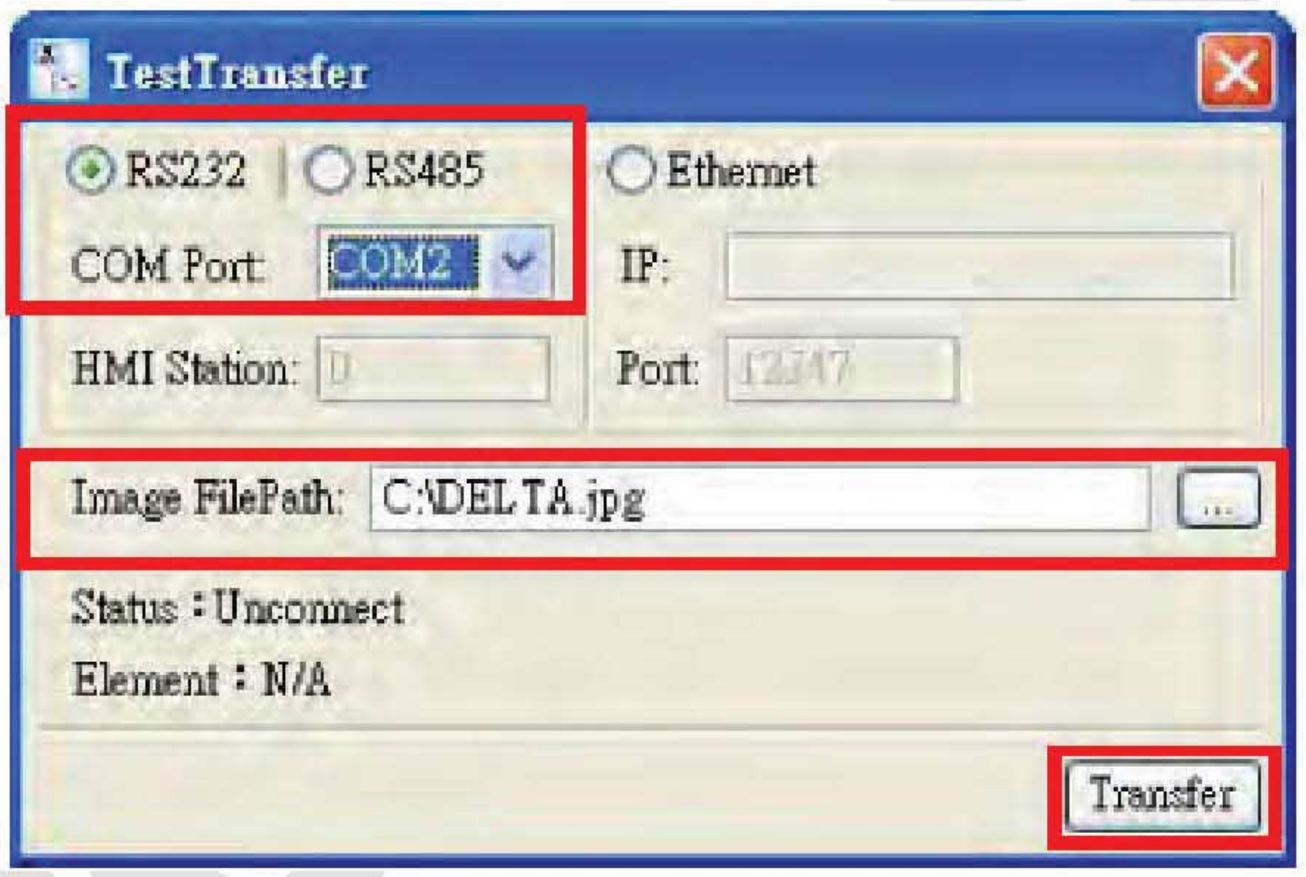
Выполнить её (по умолчанию TestTransfer.exe находится в директории:

C:\Program Files\Delta Industrial Automation\Screen Editor 2.00.07\Utility\ImgTrans)

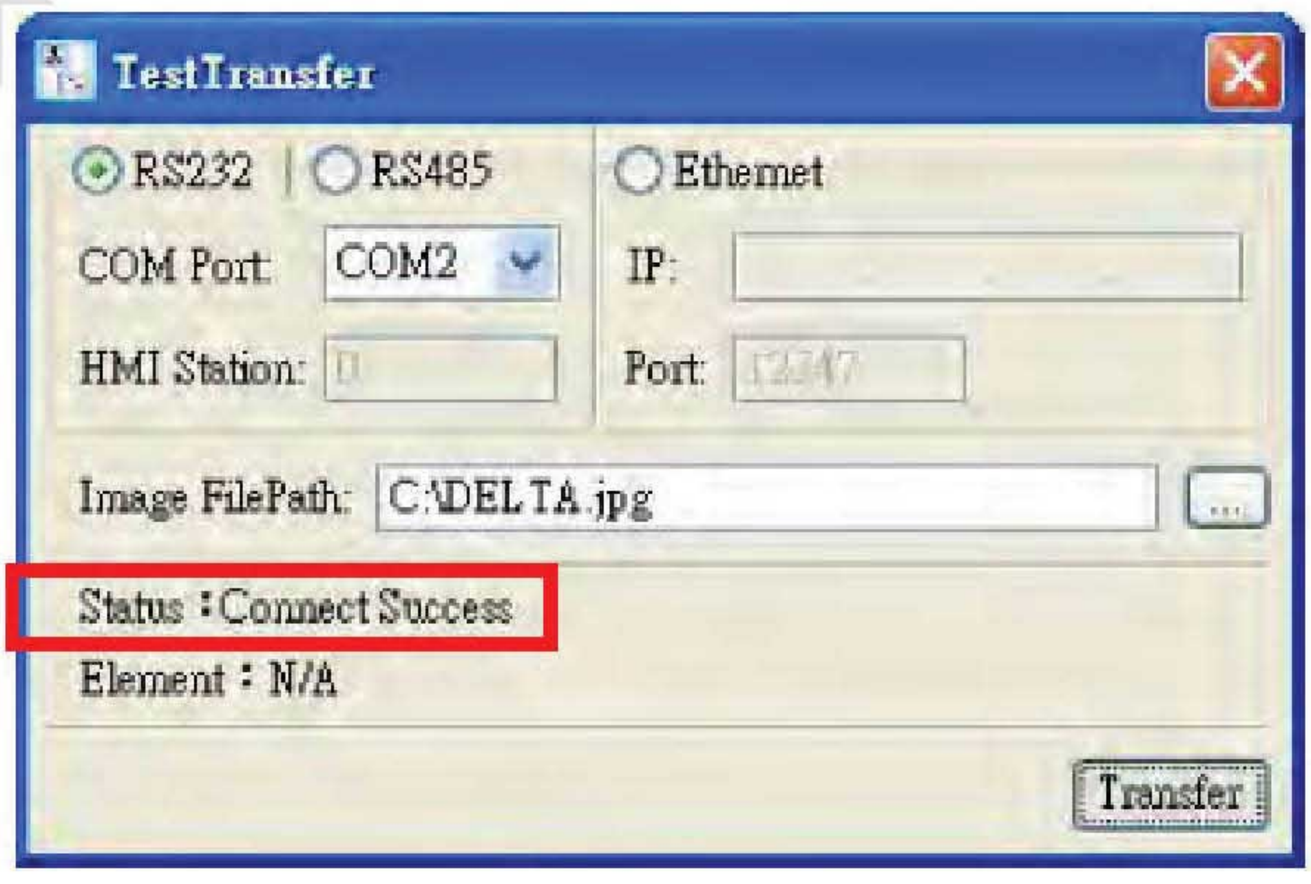


Запустить выполнение, дважды кликнув мышью

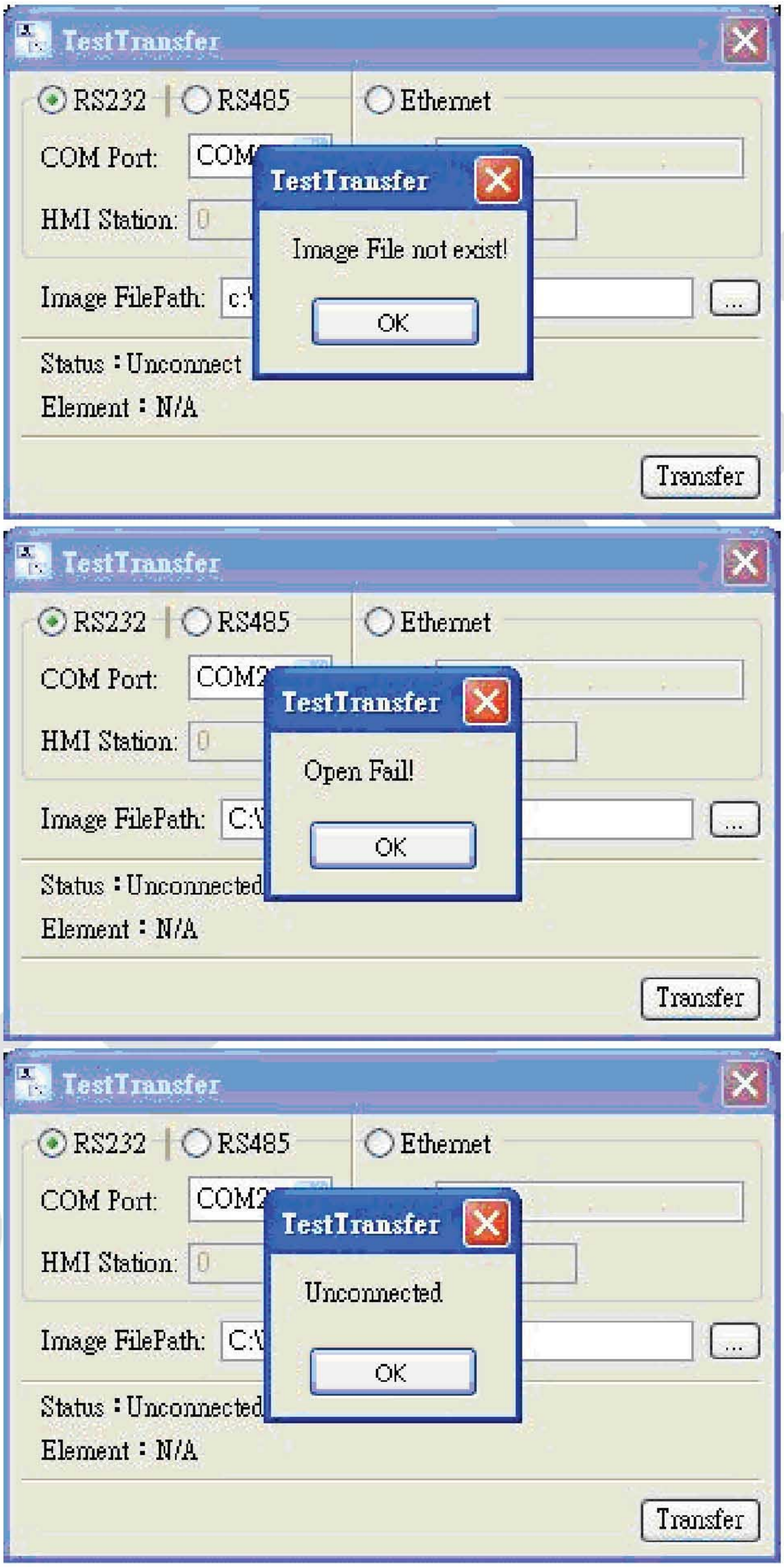
После выбора протокола и интерфейса задать путь к требуемому файлу, нажать экранную кнопку **Transfer** для передачи изображения



При успешной передаче появится сообщение **Connect Success**.



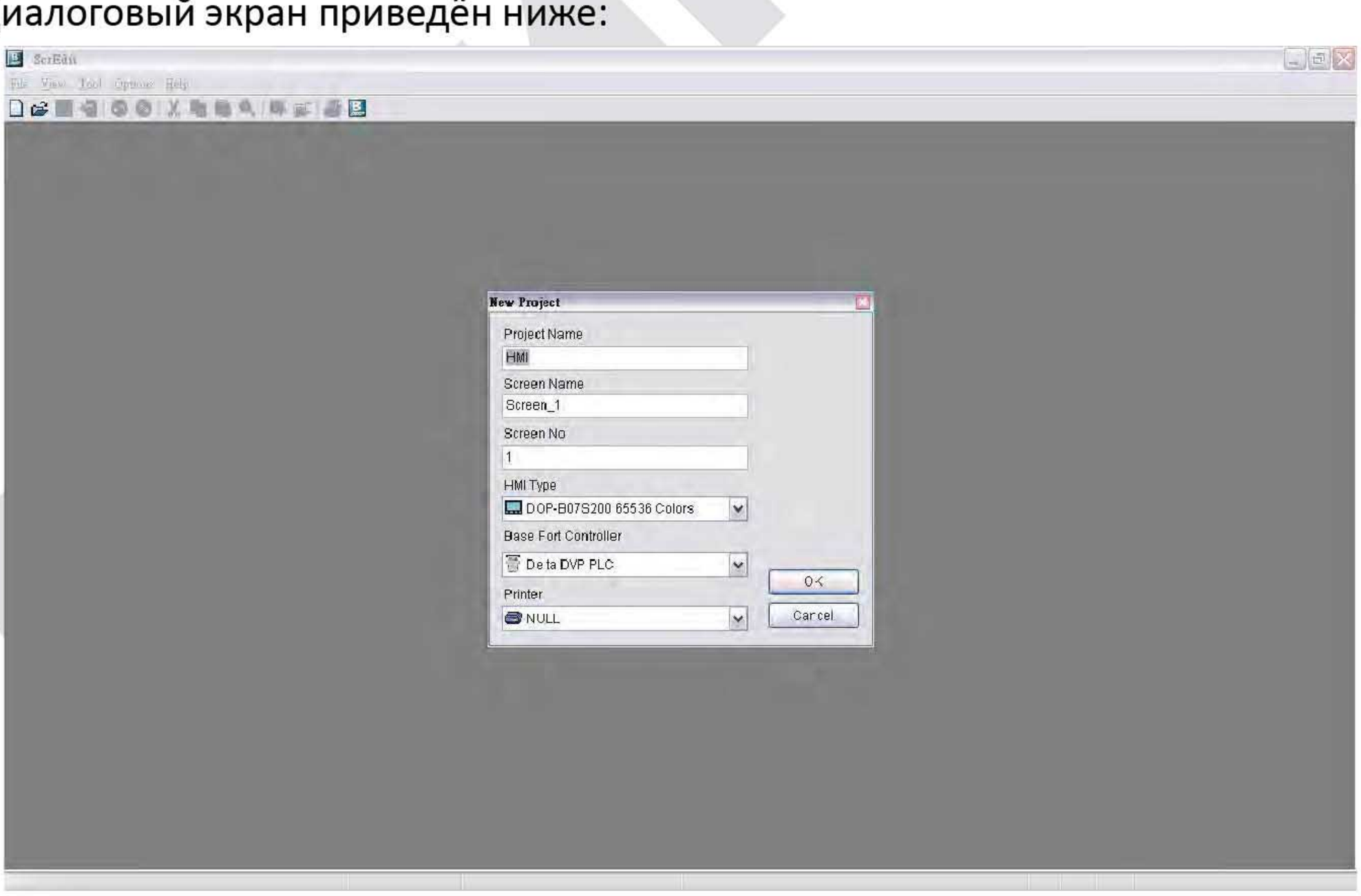
Если передача не произойдёт, появятся сообщения об ошибках



При успешной передаче на экране панели появися изображение.

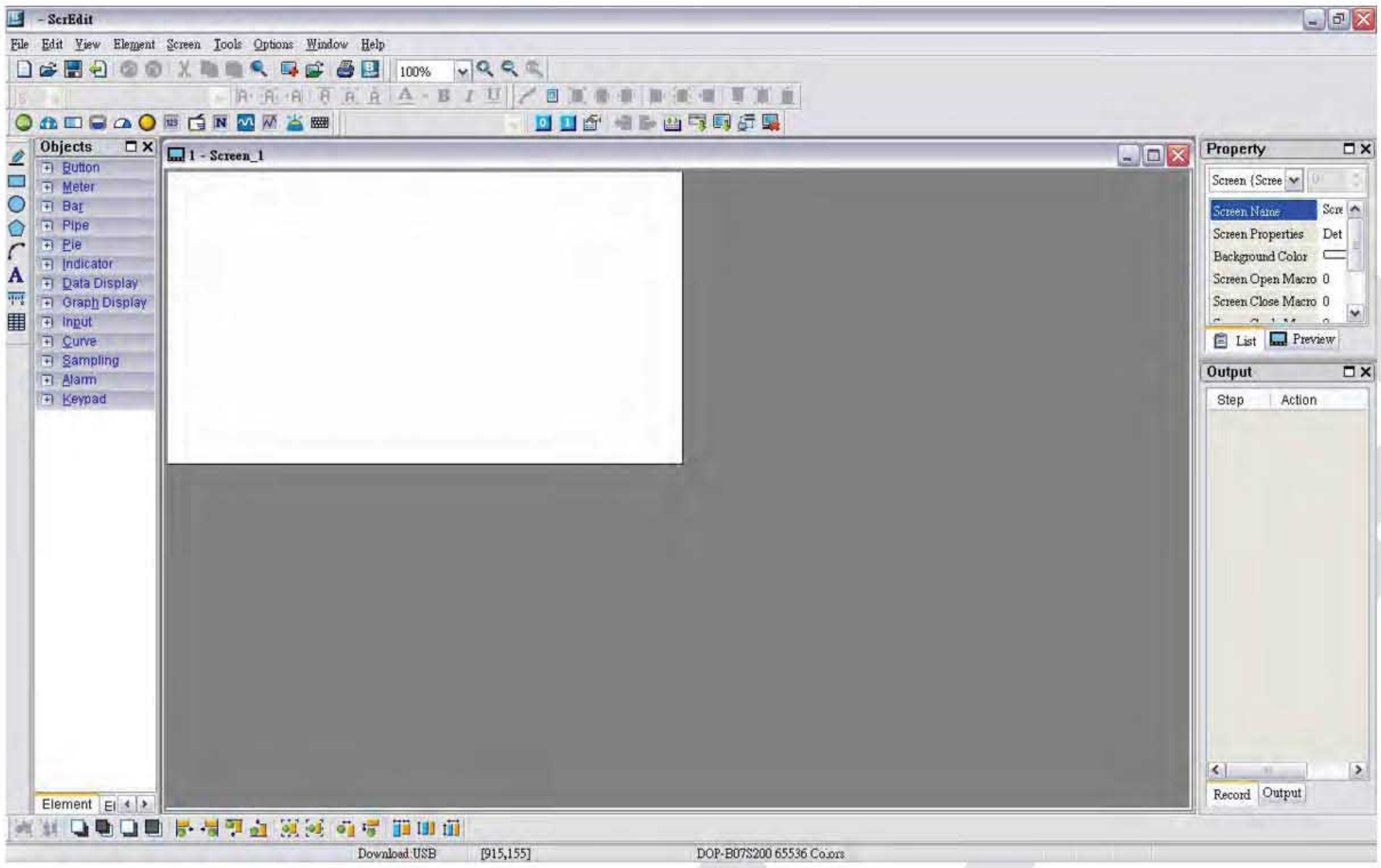
4.7 Как открыть объект Curve Element

Открыть новый проект, для чего кликнуть мышью значок  или **File > New**.
Диалоговый экран приведён ниже:

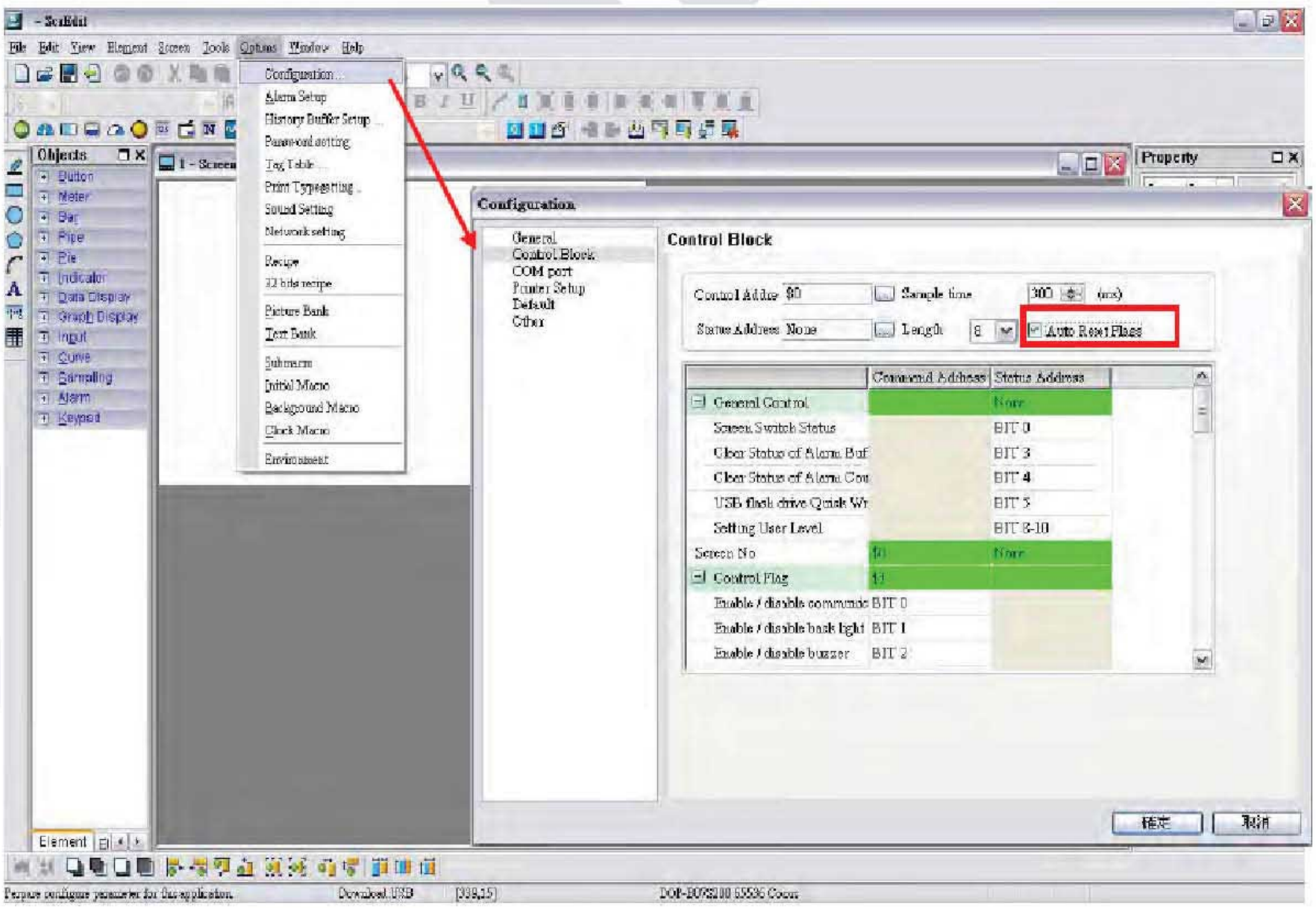


Необходимо ввести наименование проекта, наименование экрана, номер экрана определить тип панели, контроллера или принтера, далее подтвердить выбор, кликнув мышью **OK**. Как показано ниже, в программе **Screen Editor** будет открыт

новый проект:

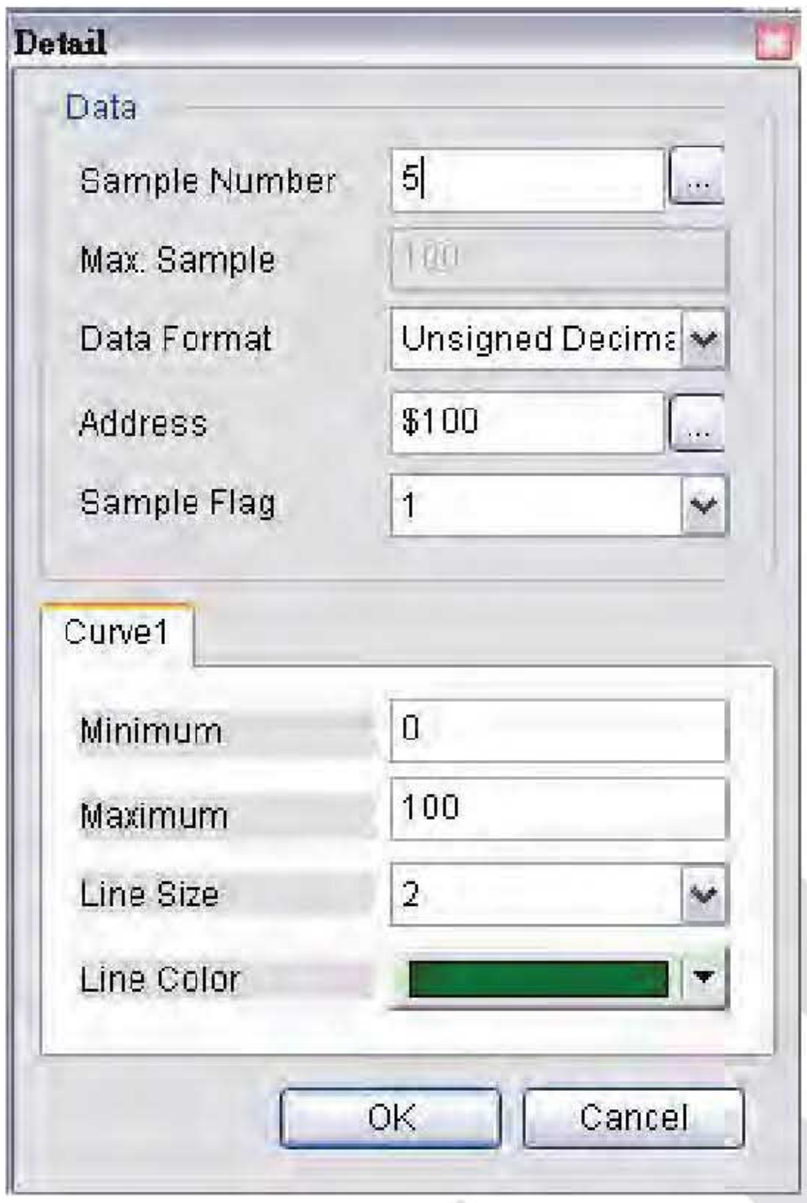


Кликнуть мышью **Options > Configuration**, select **Control Block** и установить флаг **Auto reset**. Задать адрес блока управления $0 и его размер равным 8, как показано на нижеприведённом изображении экрана:



Убедитесь, что отмечено галочкой **Auto reset**. Если эта галочка останется неотмеченной, пользователю самому будет необходимо сбрасывать флаги.

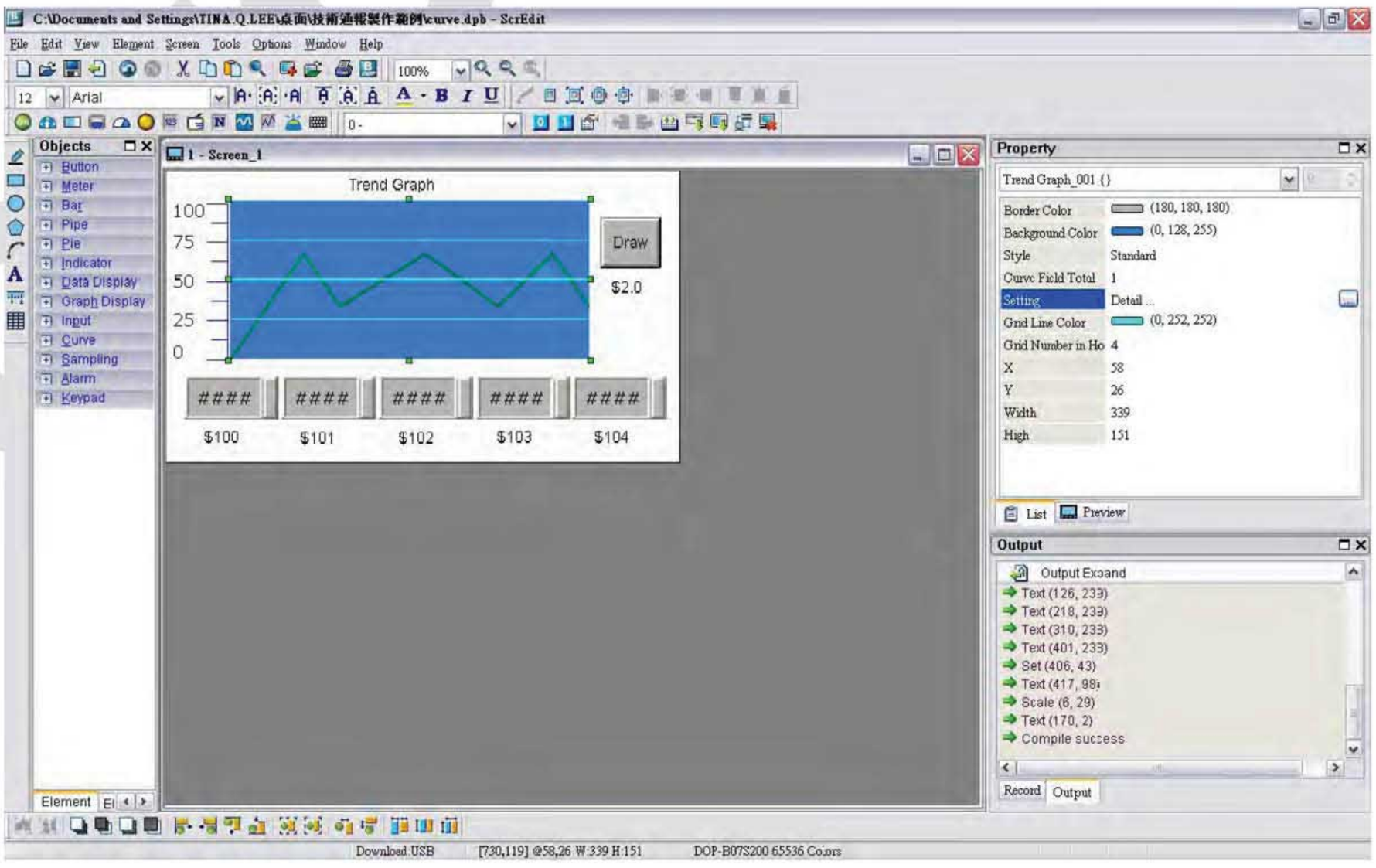
Открыть графический объект вывода графика. Задать **Sample Flag** = 1, **Sample Number** = 5, и адрес чтения $100, как показано ниже:



Так как начальный адрес задан во внутренней памяти, то адреса чтения выборок будут $100, $101, $102, $103 и $104.

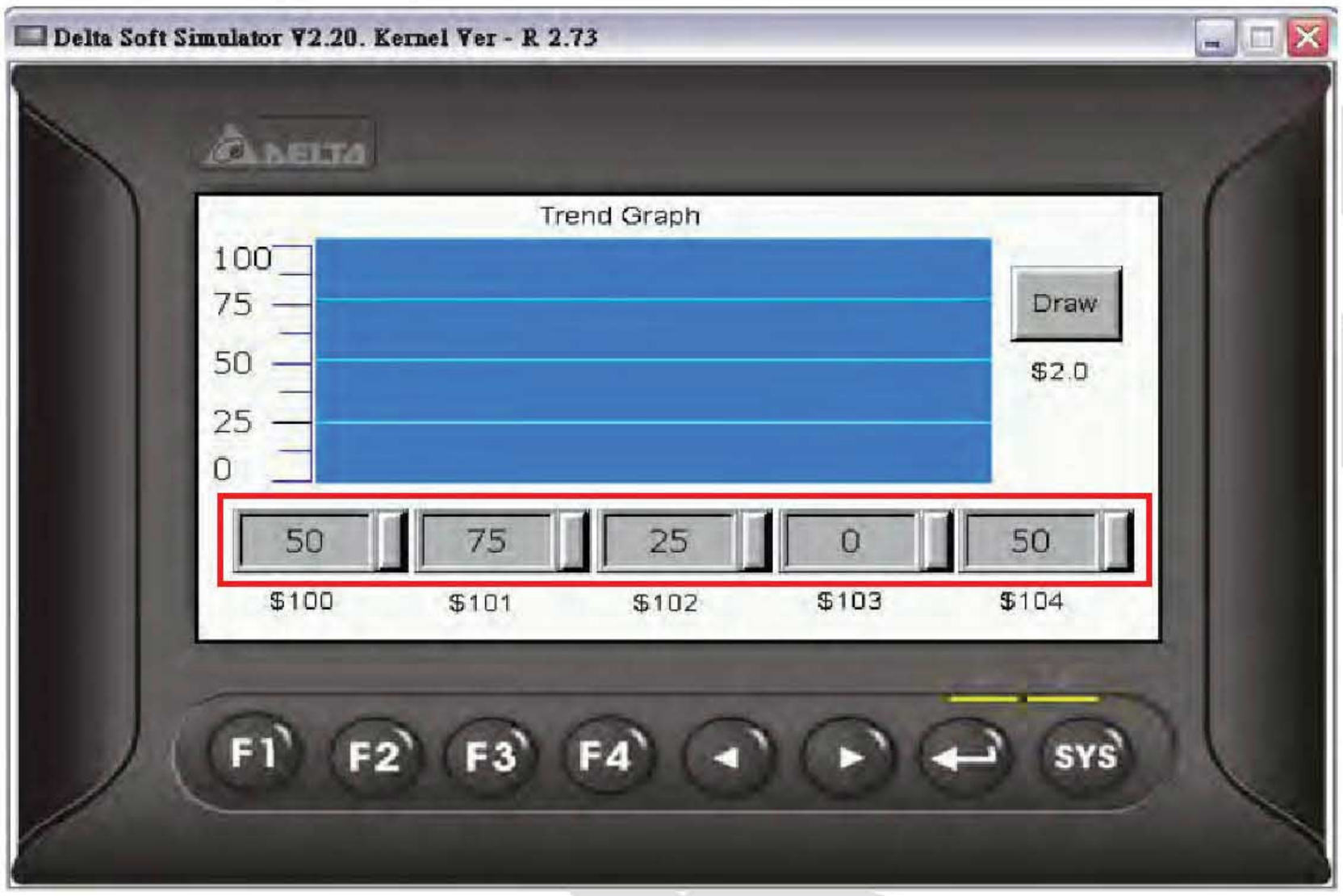
Когда **Sample Flag** = 1, это означает, что управление чтением осуществляется с помощью $2.0 в блоке управления.

Далее, открыть пять элементов цифрового ввода (адреса $100~$104.), и кнопочный элемент **Set ON** (кнопка **Draw**) для включения флага графического элемента .

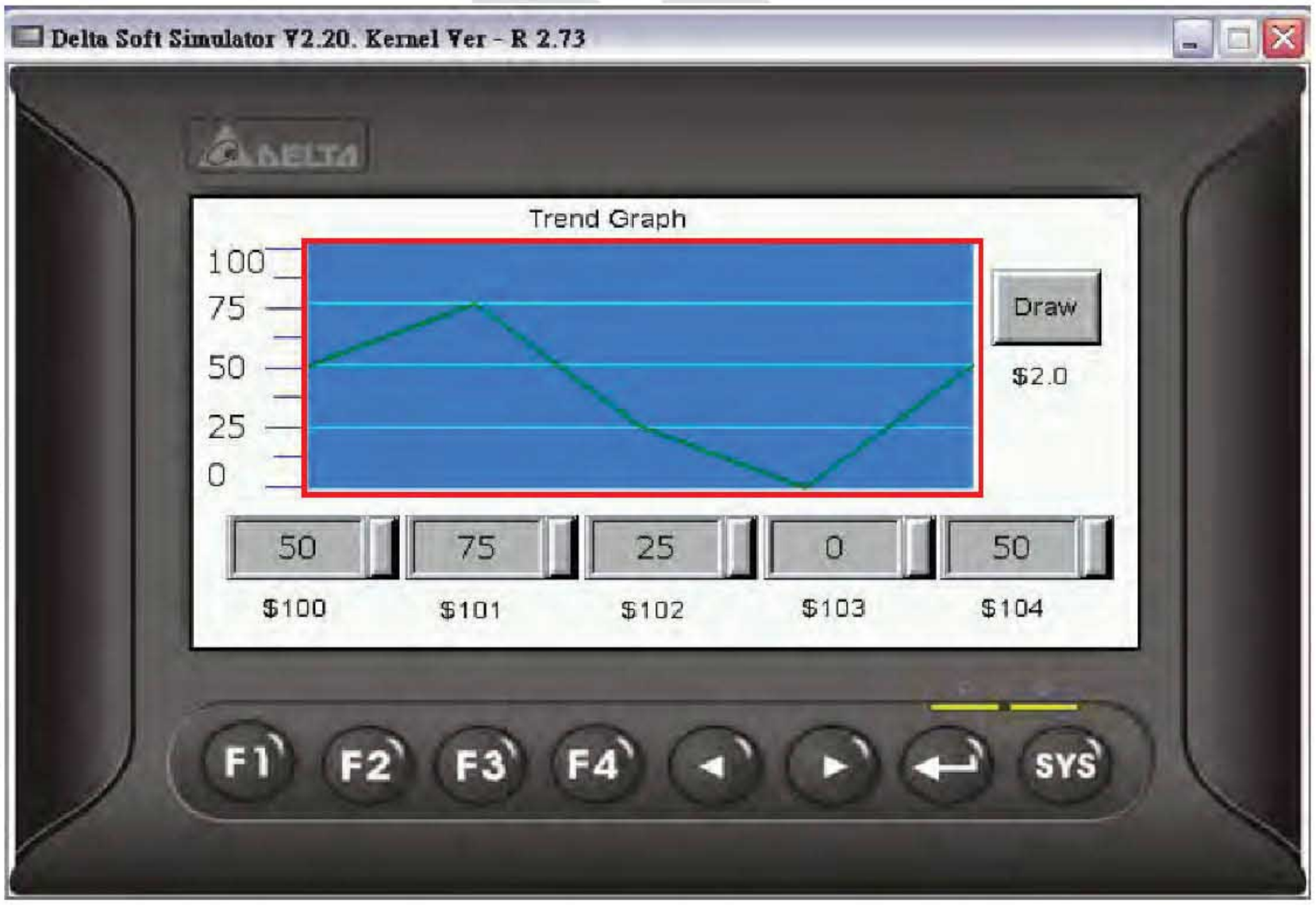


Tools > Compile Скомпилировать программу и записать её в память панели оператора.

Вид экрана после ввода данных



Вид графика после нажатия кнопки **Draw**

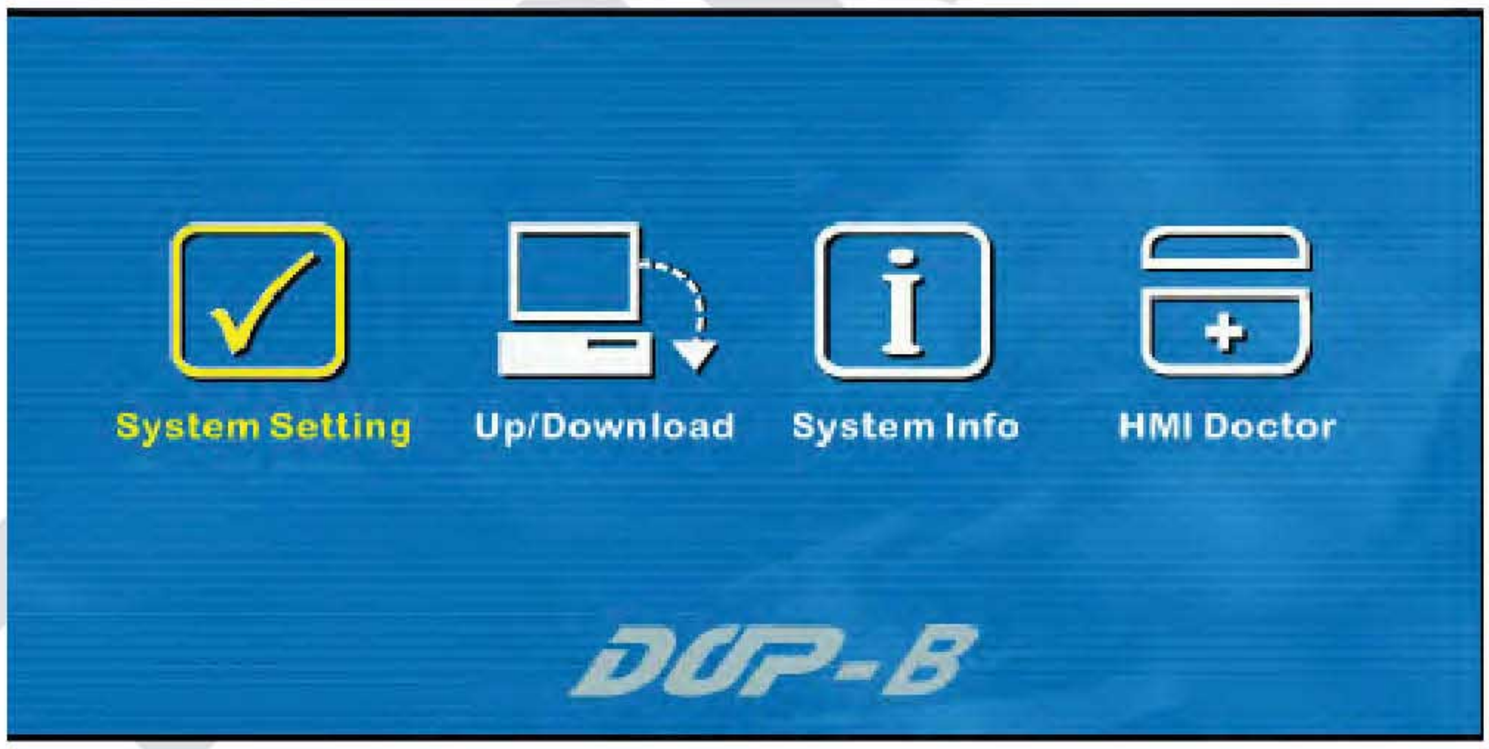


Глава 5. Системное меню

5.1 Введение в Системное Меню

■ Доступ с Системное Меню

1. Нажмите кнопку **SYS** на панели и держите в течение 2 секунд.
2. Нажмите на экран панели.
3. Перед Вами появится Системное Меню, как показано ниже:



■ Выход из Системного Меню

Нажмите кнопку **SYS** на панели и держите в течение 2 секунд.

■ Методы эксплуатации

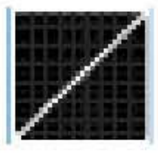
1. Пользователь может нажать на нужное меню непосредственно на экране панели.
2. Пользователь может использовать функциональные клавиши (если таковые имеются).

   Нажатие кнопок  позволит сделать выбор между

необходимым пунктом меню. Нажатие кнопки  позволит войти в выбранное меню.

Выбранный пункт меню будет подсвечен желтым цветом, т.е.  .

■ Схема Системного Меню

 Системные настройки	 Touch Panel	 TP Delay	Установка времени реакции на нажатие
		 TP Force	Задание усилия нажатия
		 TP Calibrate	Калибровка экрана
	 Date/Time	 Date	Установка даты
		 Time	Установка времени
		 Alarm Clock	Установка времени сигнализациии
	 Display	 Contrast	Настройка контраста
		 Brightness	Настройка яркости
		 Gamma	Настройка гаммы цветов

Системные настройки (продолжение)	File Manager	Formatting	HMI	Форматирование памяти панели
			USB Disk	Форматирование USB диска
			CF Card	Форматироване CF карты
			SD Card	Форматирование SD карты
		Copy File		Копирование файлов
		Multi-Screen File		Установка мультиэкранной загрузки экранов
	MISC.			Дополнительные опции
	Security			Установка таблицы паролей
	Audio	Volume		Настройки уровня звука
		Buzzer		Звуковой сигнал
	COM Port			Коммуникационный порт

Up/Download	Standard Mode	COM1	COM1 загрузка/выгрузка
		COM2	COM2 загрузка/выгрузка
	Bypass Mode	Mode 1	COM1 → COM2 байпас
		Mode 2	COM1 → COM3 байпас
		Mode 3	COM2 → COM1 байпас
Системная информация	Системные сообщения		Следующая страница
			Предыдущая страница
HMI Doctor		Белый экран	
		Черный экран	
		Красный экран	
		Зелёный экран	
		Голубой экран	
		Тест рисования линии	
		Проверка зуммера и светодиодного индикатора	
		Проверка аналого-цифрового преобразователя	

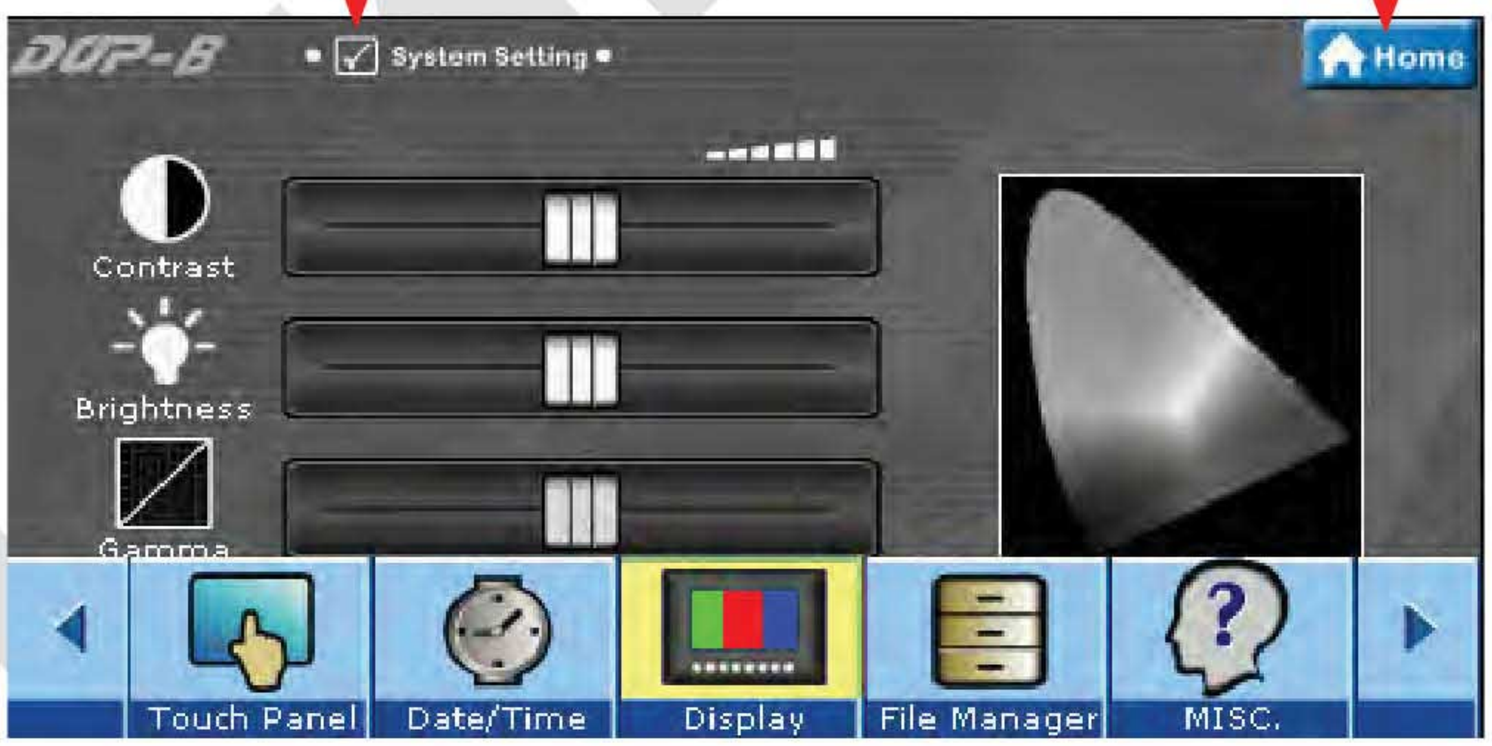
HMI Doctor (продолжение)		тест USB
		Тест кнопок
		Тест цвета

5.2 Настройки системного меню

Если изображение на экране отображено в серых тонах, то это значит, что вы видите только превью экрана данной настройки.

Для входа в меню данной настройки кликните по иконке меню. После этого экран меню должен стать цветным.

Возврат в системное меню



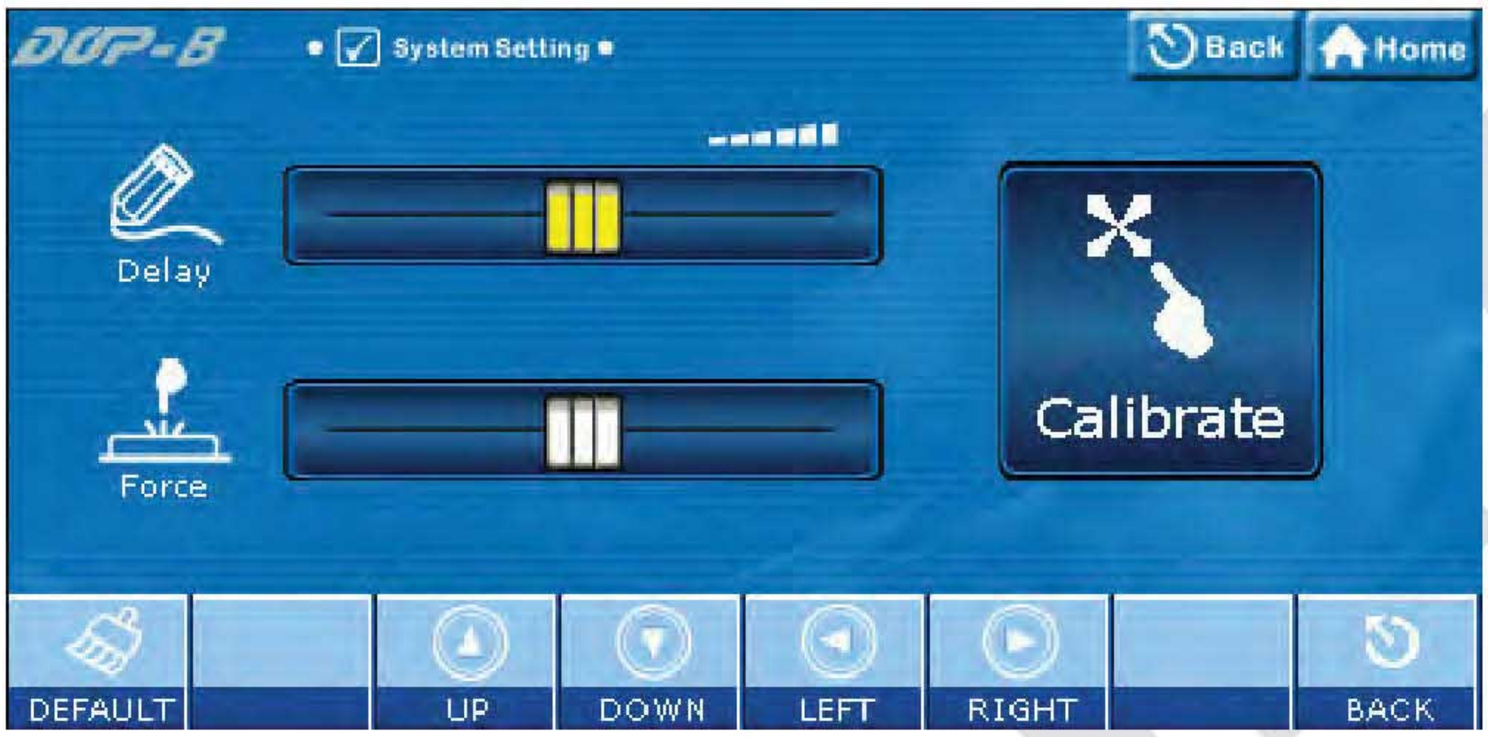
Нажимая  можно переключать опции меню.

 Опция выбрана, когда подложка имеет желтый цвет.

 Опция не выбрана когда подложка имеет голубой цвет

■ Touch Panel (Сенсорный экран)

Опция **Touch Panel** в системном меню позволяет настроить силу нажатия, время реакции экрана и провести калибровку экрана.



Функции панели управления:

Восстановить настройки по умолчанию

Переключение между опциями

Изменение параметров в выбранной опции

Выход

Кнопки на панели оператора и соответствующие им экранные объекты:

DEFAULT	UP	DOWN	LEFT	RIGHT	BACK
F1	F3	F4	◄	►	SYS

TP Delay – Установка времени реакции сенсорного экрана

Обеспечивает возможность настройки чувствительности экрана.

Уменьшение задержки увеличивает чувствительность (RIGHT). Увеличение задержки уменьшает чувствительность (LEFT)

TP Force – Установка силы нажатия

Нажатие на **LEFT** облегчает выбор экранных элементов, нажатие на **RIGHT** заставляет прилагать больше усилий для выбора экранного элемента, предотвращая случайное нажатие.

Calibrate – Калибровка экрана

Эта функция позволяет повысить точность выбора в области экранного элемента.

Калибровка проводится по трём базовым точкам .

Для повышения точности калибровки желательно использовать стилус.

Шаг 1

Шаг 2

Шаг 3

■ **Date/Time** (Установка даты и времени)

Каждое нажатие **Up** или **Down** изменяет значение параметра на 1.

Функции панели управления:

Восстановить настройки по умолчанию

Переключение между опциями

Изменение параметров в выбранной опции

Выход

Кнопки на панели оператора и соответствующие им экранные объекты:

CANCEL	UP	DOWN	LEFT	RIGHT	BACK
F1	F3	F4	◄	►	SYS

Date – Установка даты

2007 08 26 Запись значений Year, Month, Day of Month последовательно нажимая кнопки **UP** и **DOWN**


Time – Установка времени

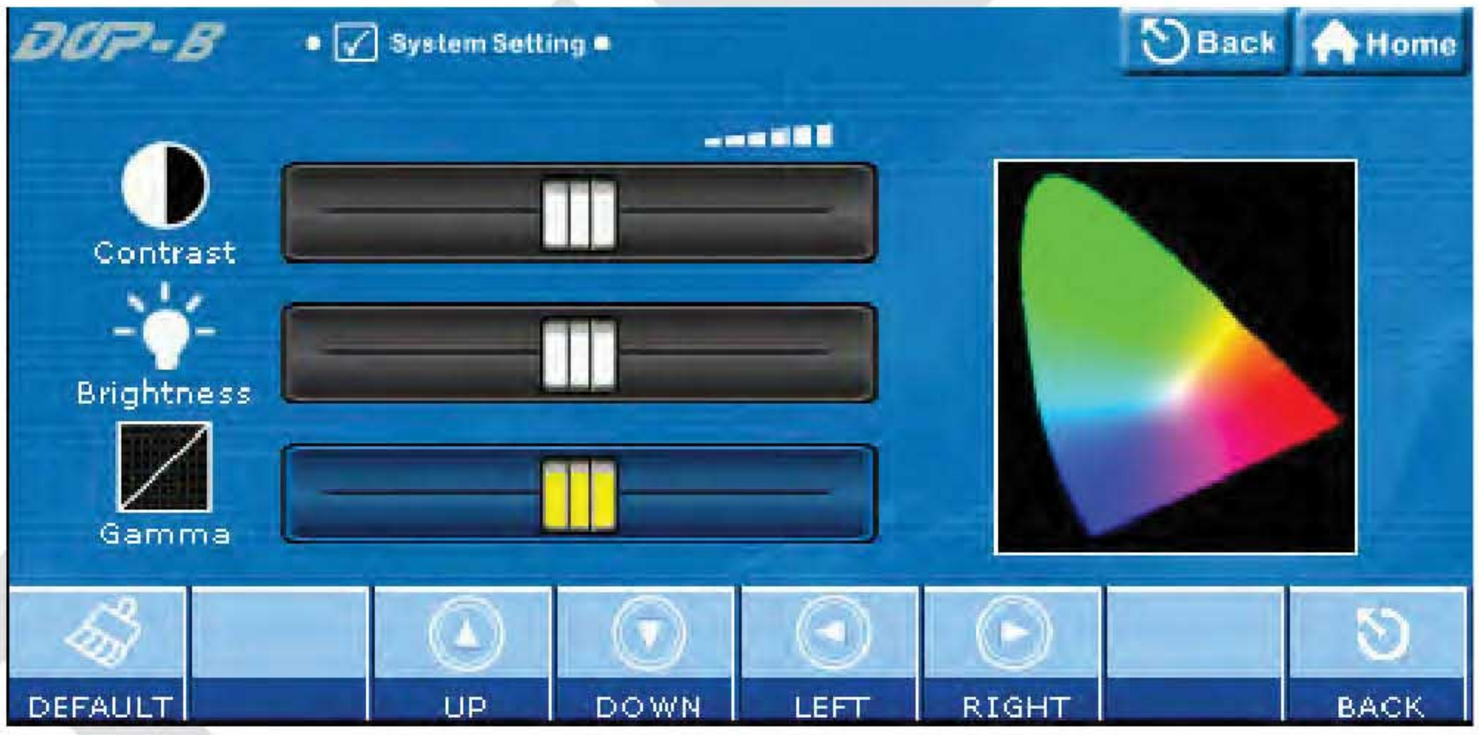
Запись значений Hour, Minute, Second последовательно нажимая кнопки **UP** и **DOWN**


Alarm Clock – Установка будильника (функция зарезервирована)

- 
Display (LCD Display) Настройка дисплея

Черная подложка опции означает, что параметр зарезервирован и не может быть отрегулирован.

Пример на картинке ниже



Функции панели управления:


Восстановить настройки по умолчанию


Переключение между опциями


Изменение параметров в выбранной опции


Выход

Кнопки на панели оператора и соответствующие им экранные объекты:

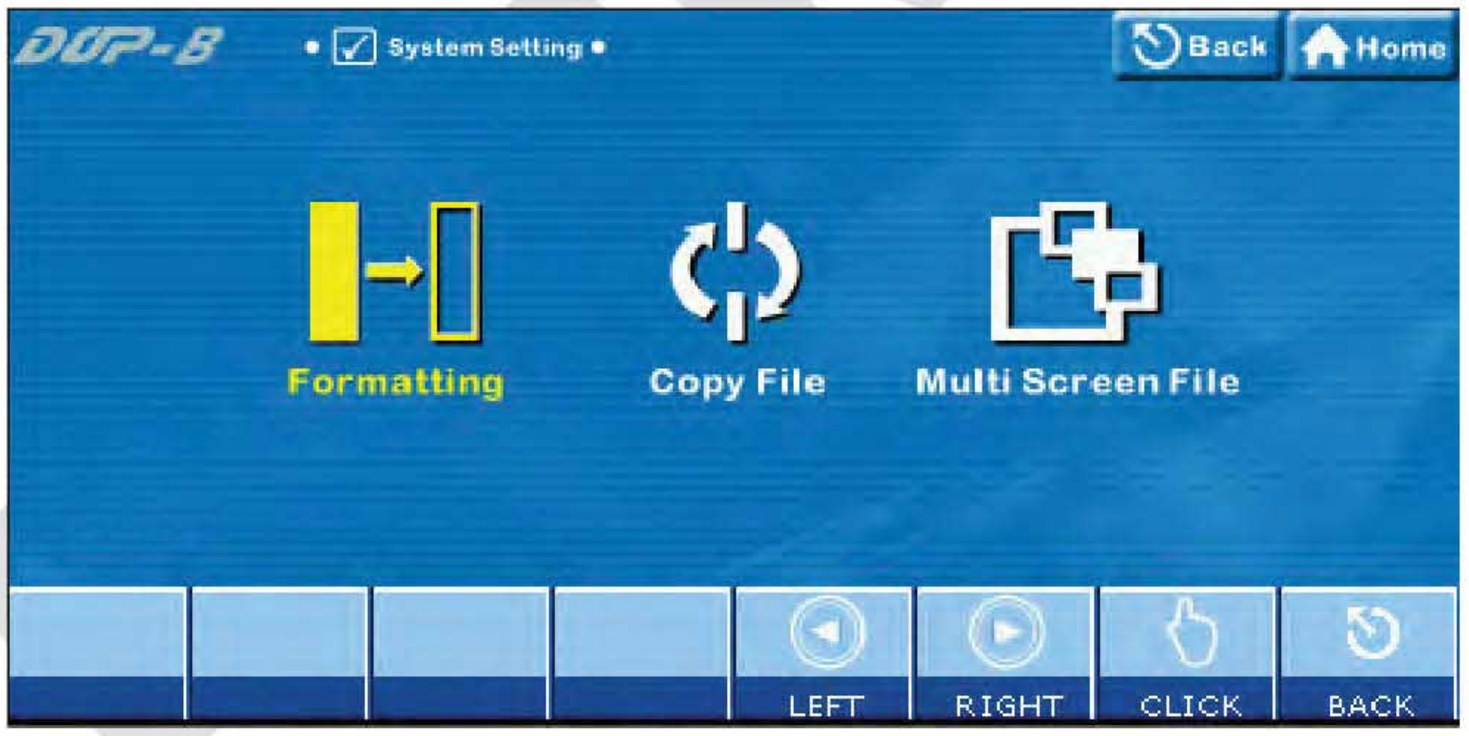
DEFAULT	UP	DOWN	LEFT	RIGHT	BACK
F1	F3	F4	◄	►	SYS

Contrast – Настройка контраста (настройка зарезервирована)

Brightness – Настройка яркости (настройка зарезервирована)

Gamma – Настройка цветовой насыщенности.

- **File Manager** (Файл менеджер)



Функции панели управления:

Переключение между опциями

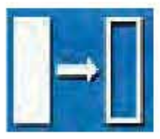
Выбранная опция выделена желтым цветом

Выбор опции

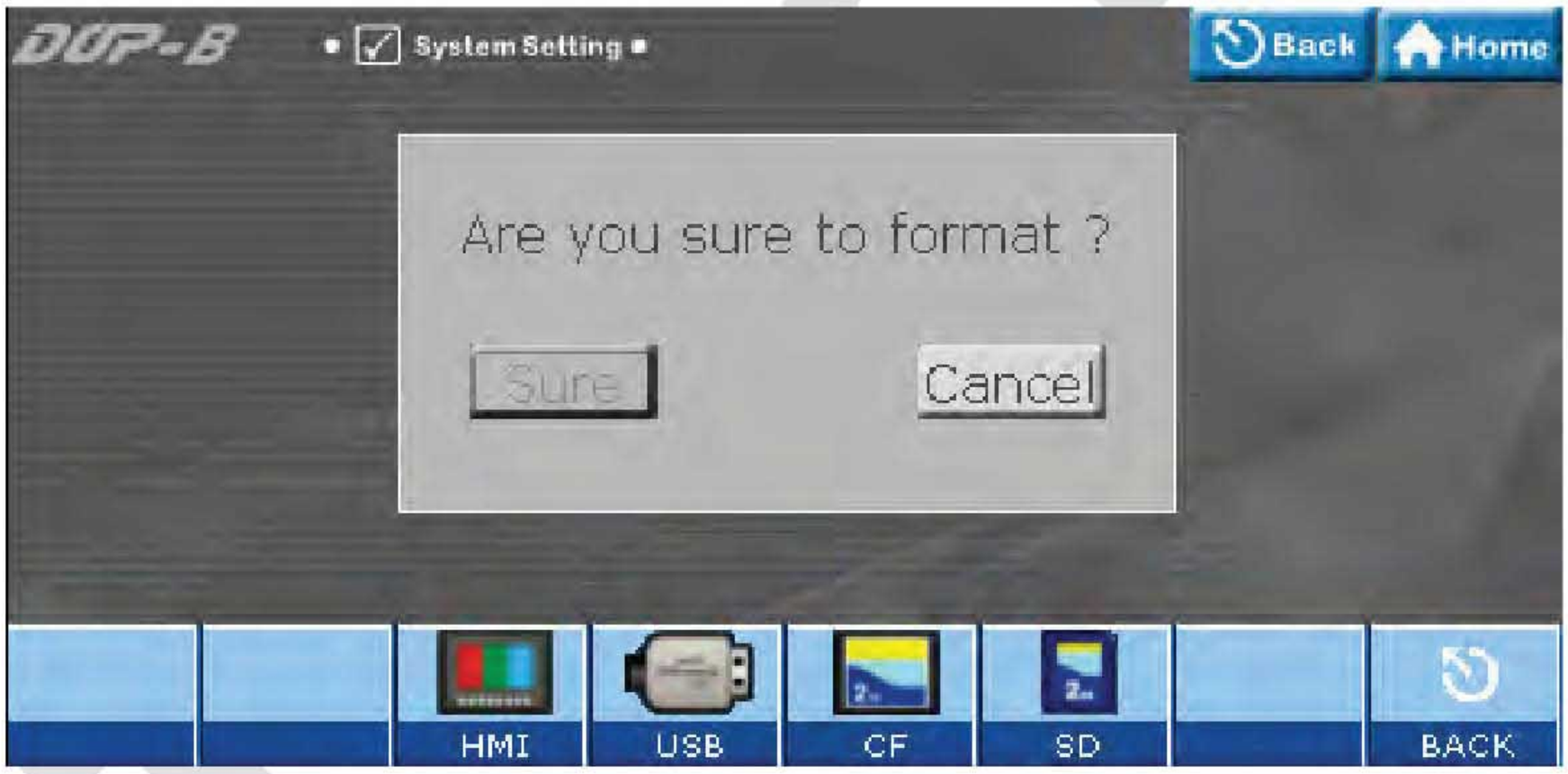
Выход

Кнопки на панели оператора и соответствующие им экранные объекты:

 LEFT	 RIGHT	 CLICK	 BACK
			

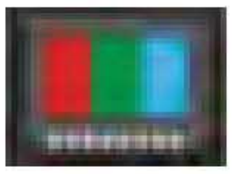
 **Formatting** (Форматирование памяти)

Функция предназначена для форматирования памяти (внутренней или внешней). Если подложка темная, то данная функция не реализована в данной модели.



Кнопки на панели оператора и соответствующие им экранные объекты

 HMI	 USB	 CF	 SD	 BACK
				

 **HMI** – Форматирование внутренней памяти панели оператора

Нажатие на  или  панели оператора приводит к появлению на экране окна подтверждения форматирования памяти. Нажатие экранной кнопки с надписью **SURE** разрешает форматирование, а нажатие кнопки **Cancel** прекращает процедуру форматирования

 **USB Disk** – Форматирование USB диска (функция зарезервирована)

 **CF Card** – Форматирование CF карты (функция зарезервирована)

 **SD Card** – Форматирование SD карты (функция зарезервирована)

Copy File – Копирование файлов

Данная опция позволяет копирование файлов между внутренней и внешней памятью. Для доступа требуется пароль наивысшего приоритета.

Функции панели управления:

Выбор диска

Переключение между источником/приёмником данных

Подтверждение выбора

Выполнение копирования

Выход

Кнопки на панели оператора и соответствующие им экранные объекты:

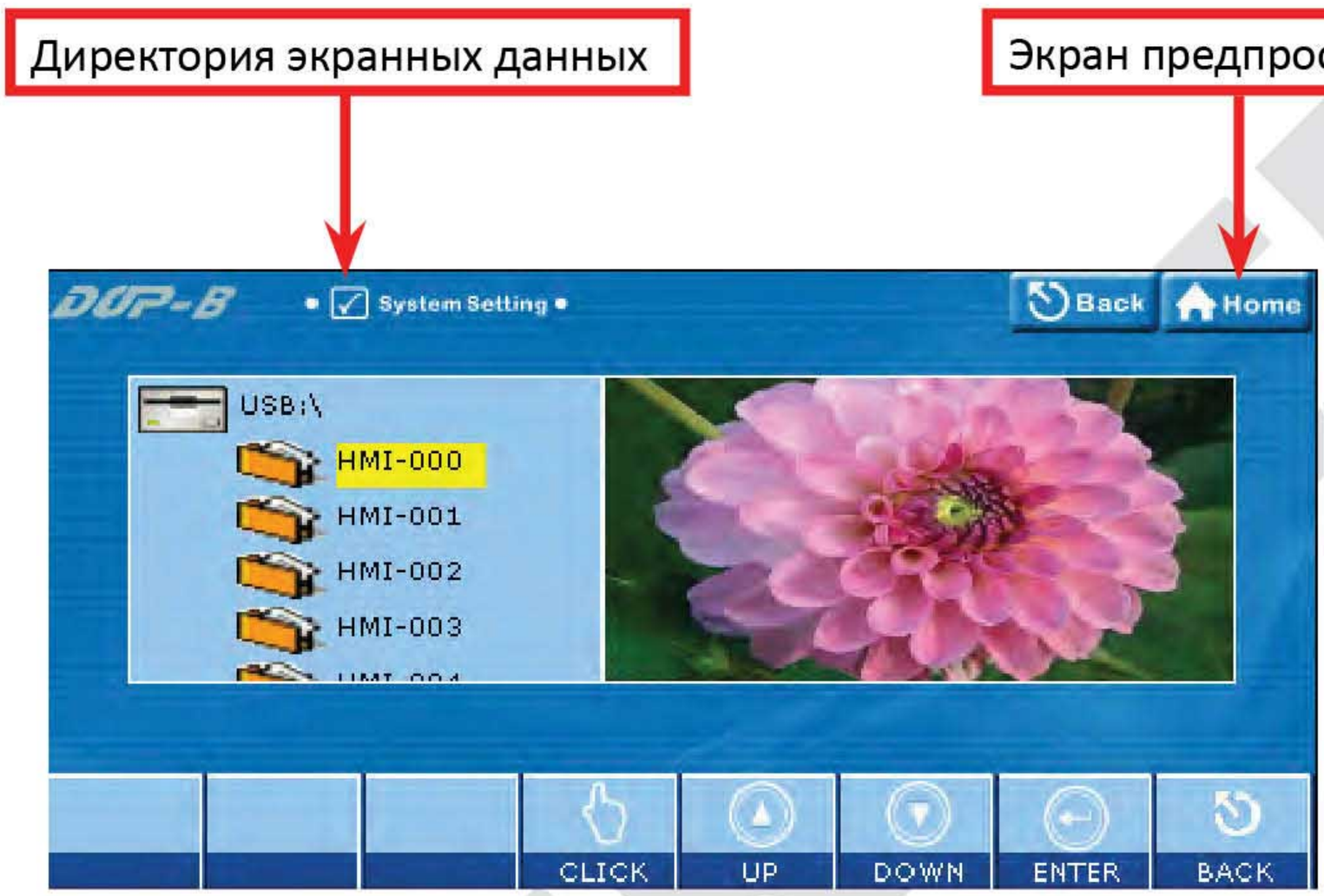
 UP	 DOWN	 CLICK	 LEFT	 RIGHT	 COPY	 BACK
						

Примечания:

- Панель оператора не позволяет непосредственно копировать диск
- Только файлы HMI-000 ~ HMI255 возможно копировать из директории.
- Рекомендуется, при копировании файлов, закрыть целевую директорию, а далее проводить копирование.
- Когда открывается новый файл, то ему присваивается неиспользуемое имя из значений от HMI-000 до HMI-255 и открывается целевая директория (режим NEW).
- Если экранные данные закрыты паролем, то требуется ввести пароль наивысшего уровня, иначе копирование будет запрещено.

Multi-Screen File – Настройки мультиэкранного загрузчика

Эта функция используется для реализации загрузки экрана, из файла, записанного в памяти панели или внешней памяти.



Функции панели управления:

Выбор диска

Подтверждение выбора

Установка данного файла в экран загрузки

Выход

Кнопки на панели оператора и соответствующие им экранные объекты:

CLICK	UP	DOWN	ENTER	BACK
F4	◄	►	↵	SYS

Примечания:

- Только файлы HMI-000 ~ HMI255 возможно копировать из директории.
- Когда панель подключена к питанию и включена, если директория с загрузочным экраном не обнаружена, панель автоматически загрузит файл с экраном, но при этом путь к загрузочному экрану останется прежним. Таким образом, если при следующей загрузке панель найдет директорию с загрузочным экраном, панель по прежнему загрузит этот экран как экран по умолчанию.
- Когда директория с загрузочным экраном хранится на внешнем хранилище данных, энергонезависимые данные также будут храниться в этой директории, независимо от настроек для энергонезависимых данных.

■ **MISC** (Прочие настройки)

Настройки :

- Курсор
- Установка времени хранителя экрана
- Задание задержки загрузки
- Язык интерфейса (по умолчанию)
- Установка функции работы голубого светодиода
- Загрузочный дисплей

Кнопки на панели оператора и соответствующие им экранные объекты:

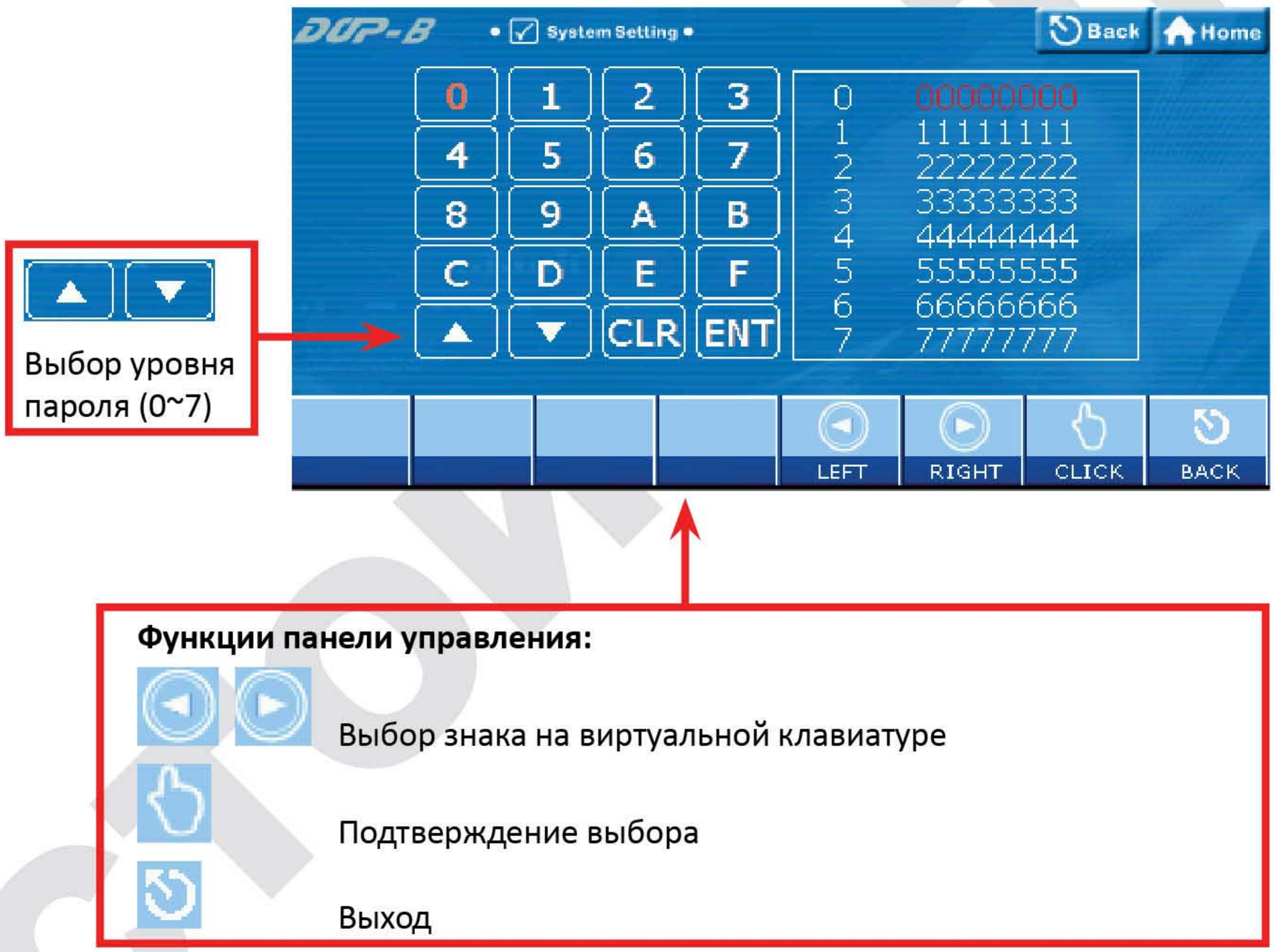
DEFAULT	UP	DOWN	LEFT	RIGHT	BACK
F1	F3	F4	◄	►	SYS

■ **Security (Password Таблица setup)** (Установка таблицы паролей)

Эта опция используется для установки нескольких уровней паролей.

Самый высокий уровень 7, самый низкий 0.

Когда открыта эта таблица, пользователь может видеть или менять пароли с уровнем доступа ниже своего. Пользователь с уровнем доступа 7 может видеть и менять любые пароли.



Кнопки на панели оператора и соответствующие им экранные объекты:

LEFT	RIGHT	CLICK	BACK
			

■ **Audio** (Настройки звука)

Функции панели управления:

Восстановить настройки по умолчанию

Переключение между опциями

Изменение параметров в выбранной опции

Выход

Кнопки на панели оператора и соответствующие им экранные объекты:

DEFAULT	UP	DOWN	LEFT	RIGHT	BACK
F1	F3	F4	◄	►	SYS

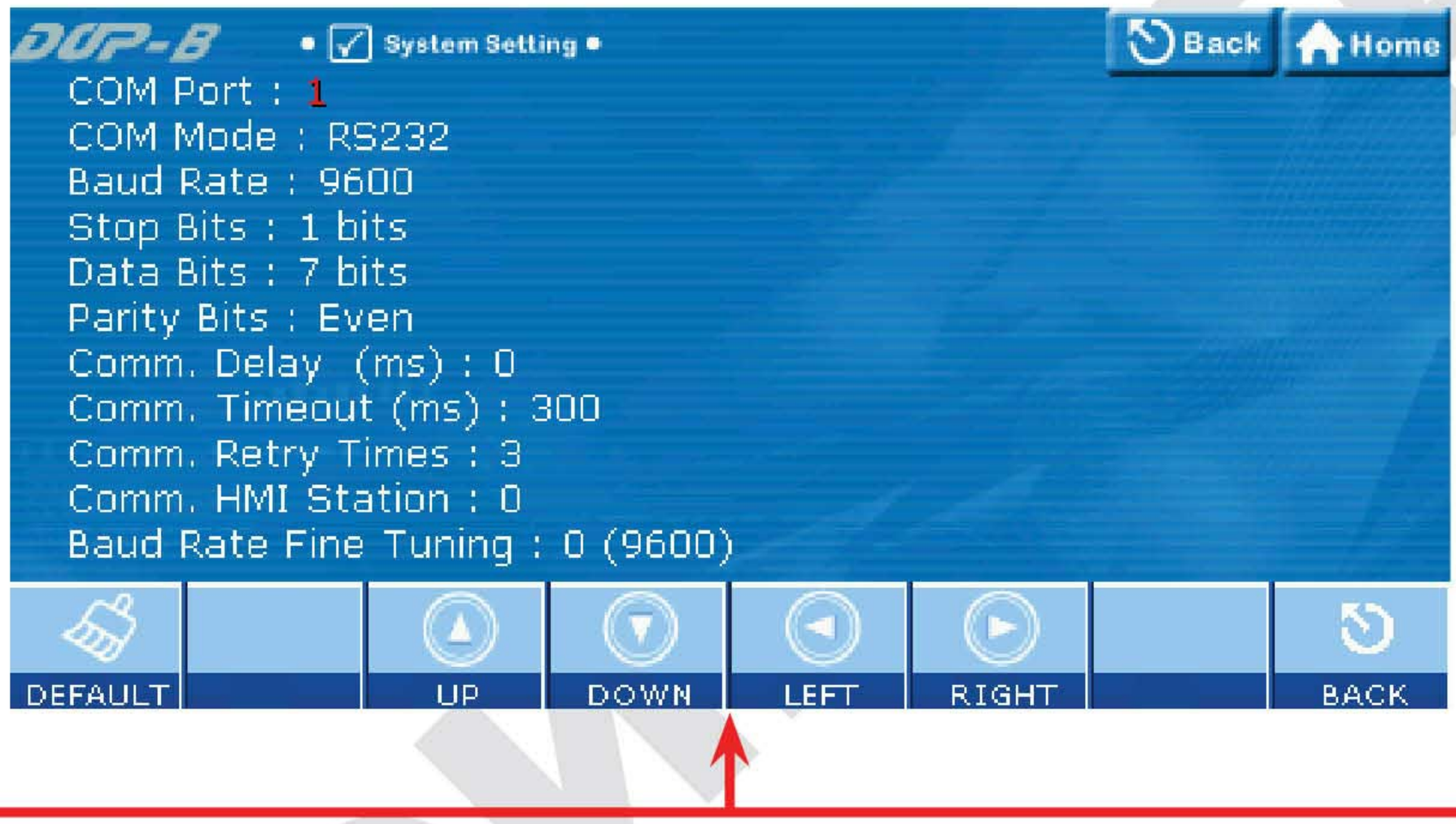
Volume – Настройка уровня громкости.

Эта функция позволяет задавать необходимую громкость звука. Кнопка **LEFT** уменьшает громкость звука, **RIGHT** - увеличивает.


Buzzer – Зуммер/будильник

Кнопка **LEFT** уменьшает громкость звука, **RIGHT** - увеличивает.

- 
COM Port



Функции панели управления:


Восстановить настройки по умолчанию


Переключение между опциями


Изменение параметров в выбранной опции


Выход

Кнопки на панели оператора и соответствующие им экранные объекты:

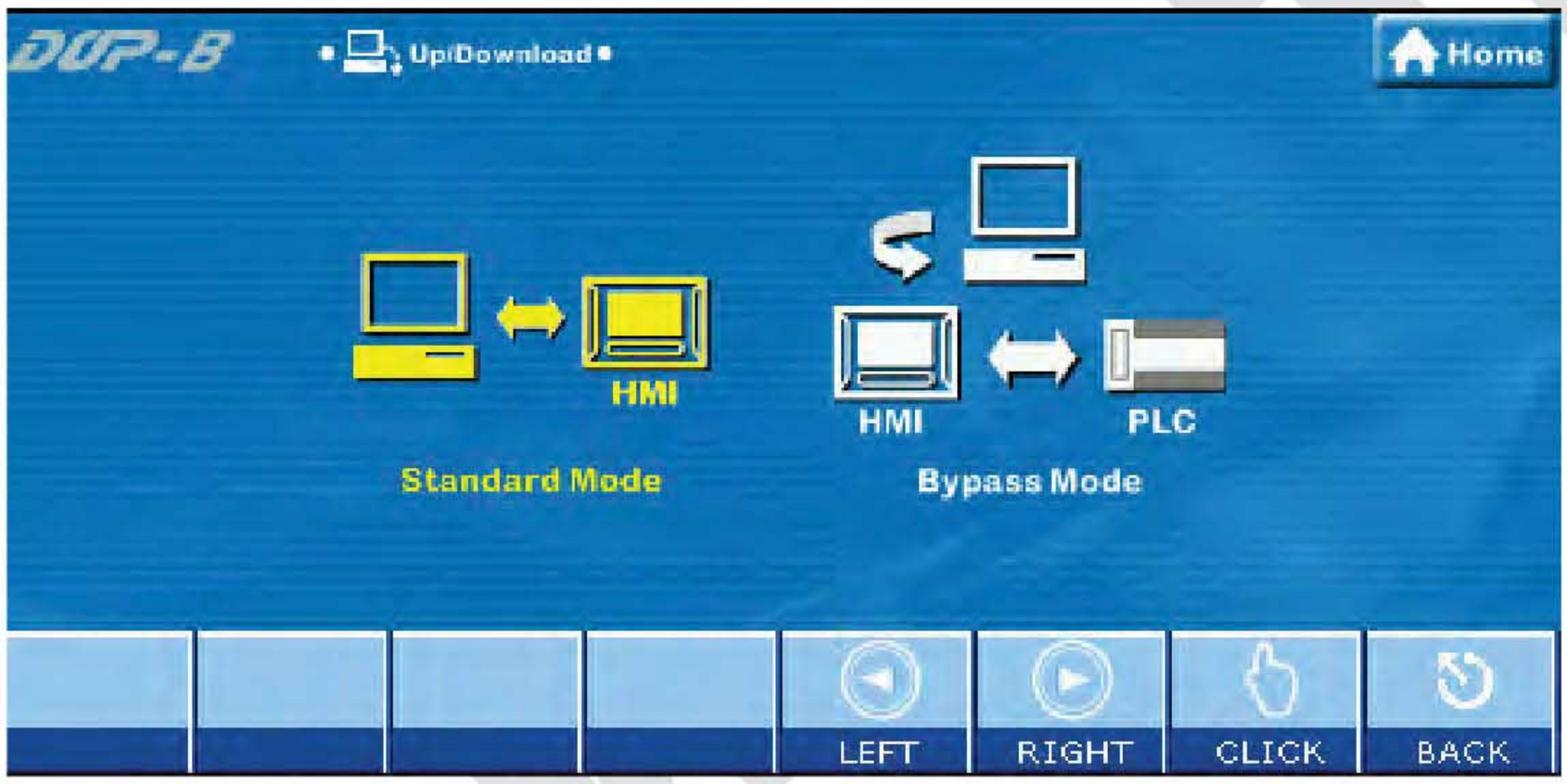
 DEFAULT	 UP	 DOWN	 LEFT	 RIGHT	 BACK
					

5.3 Меню режимов связи

Меню режимов коммуникации с контроллерами

Два типа режимов связи через последовательный порт

1. Стандартный режим (Standart Mode)
2. Режим байпас (Bypass Mode)

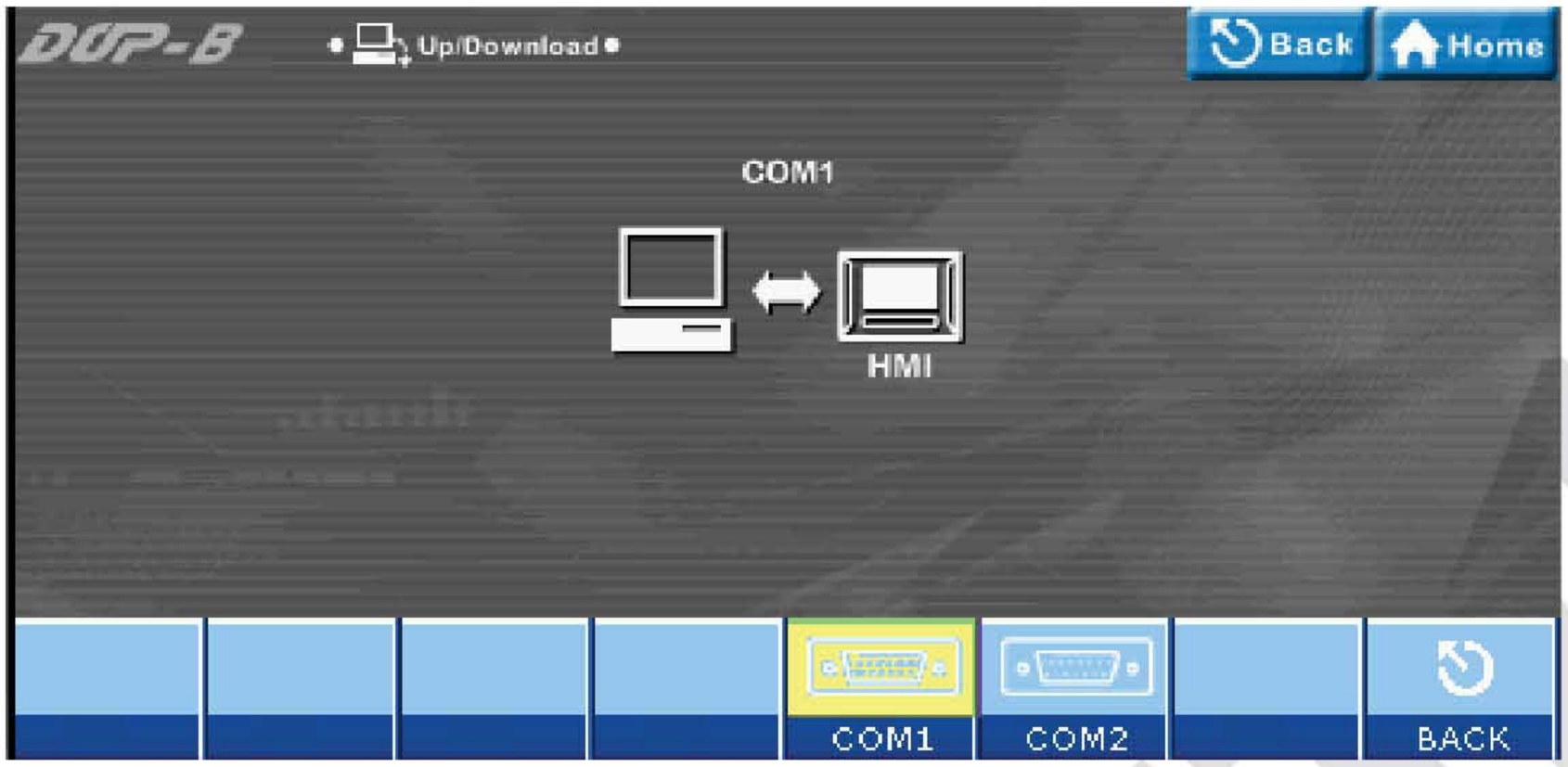


Кнопки на панели оператора и соответствующие им экранные объекты^

LEFT	RIGHT	CLICK	BACK
			SYS

Standard Mode (Стандартный режим)

В этом режиме, COM порт соединён с компьютером и с помощью программы **Screed Editor** производится загрузка/выгрузка программ. Можно выбрать для этого COM1 или COM2.



Кнопки на панели оператора и соответствующие им экранные объекты:

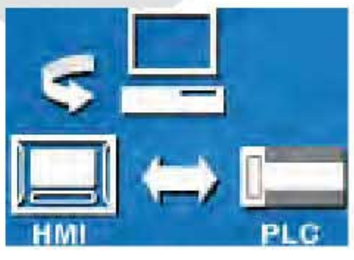
COM1	COM2	BACK
F3	F4	SYS

COM1 (COM1 загрузка/выгрузка)

Если выбран COM1, то это означает, что через него возможна передача данных с помощью настроек в программе **Screen Editor**.

COM2 (COM2 загрузка/выгрузка)

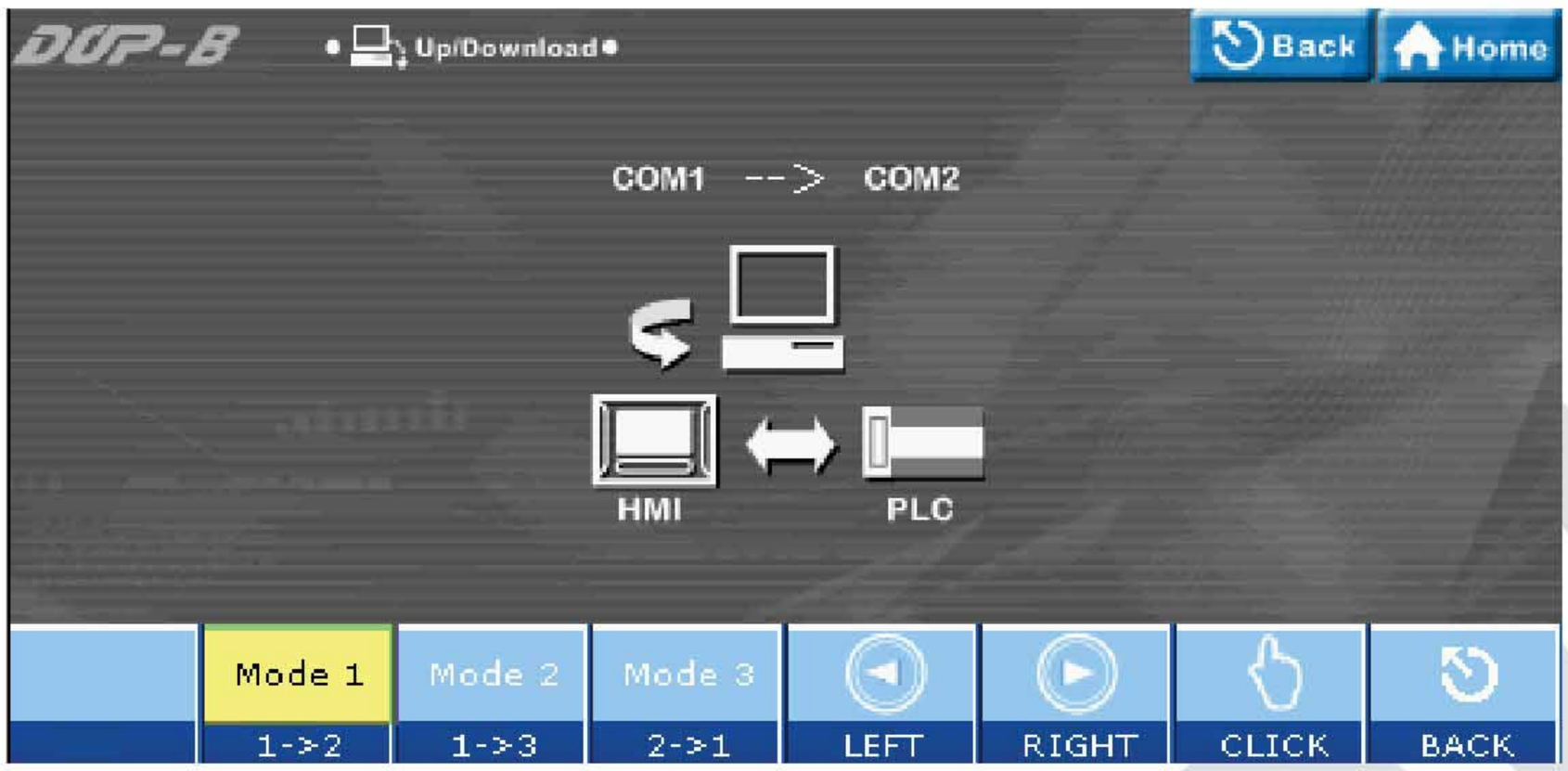
Если выбран COM2, то это означает, что через него возможна передача данных с помощью настроек в программе **Screen Editor**.



Bypass Mode (Режим байпас)

В таком режиме, данные переданные в панель оператора через через COM порт от компьютера, могут быть переданы к контроллеру, подключенному к другому COM порту.

Таким образом, благодаря режиму байпас в системе возможно оперативно проводить обновление программы контроллера.



Кнопки на панели оператора и соответствующие им экранные объекты:

Mode 1 1 → 2	Mode 2 1 → 3	Mode 3 2 → 1	LEFT	RIGHT	ENTER	BACK
F2	F3	F4	◄	►	↵	SYS

Mode 1 Режим COM1 → COM2 байпас.

Выбор этого режима означает, что порт COM2 панели оператора связан с контроллером, а данные от компьютера через порт COM1 панели оператора могут передаваться в контроллер.

(COM1 является источником данных, а порт COM2 - портом передачи данных в контроллер.)

Mode 2 Режим COM1 → COM3 байпас.

Выбор этого режима означает, что порт COM3 панели оператора связан с контроллером, а данные от компьютера через порт COM1 панели оператора могут передаваться в контроллер.

(COM1 является источником данных, а порт COM3 - портом передачи данных в контроллер)

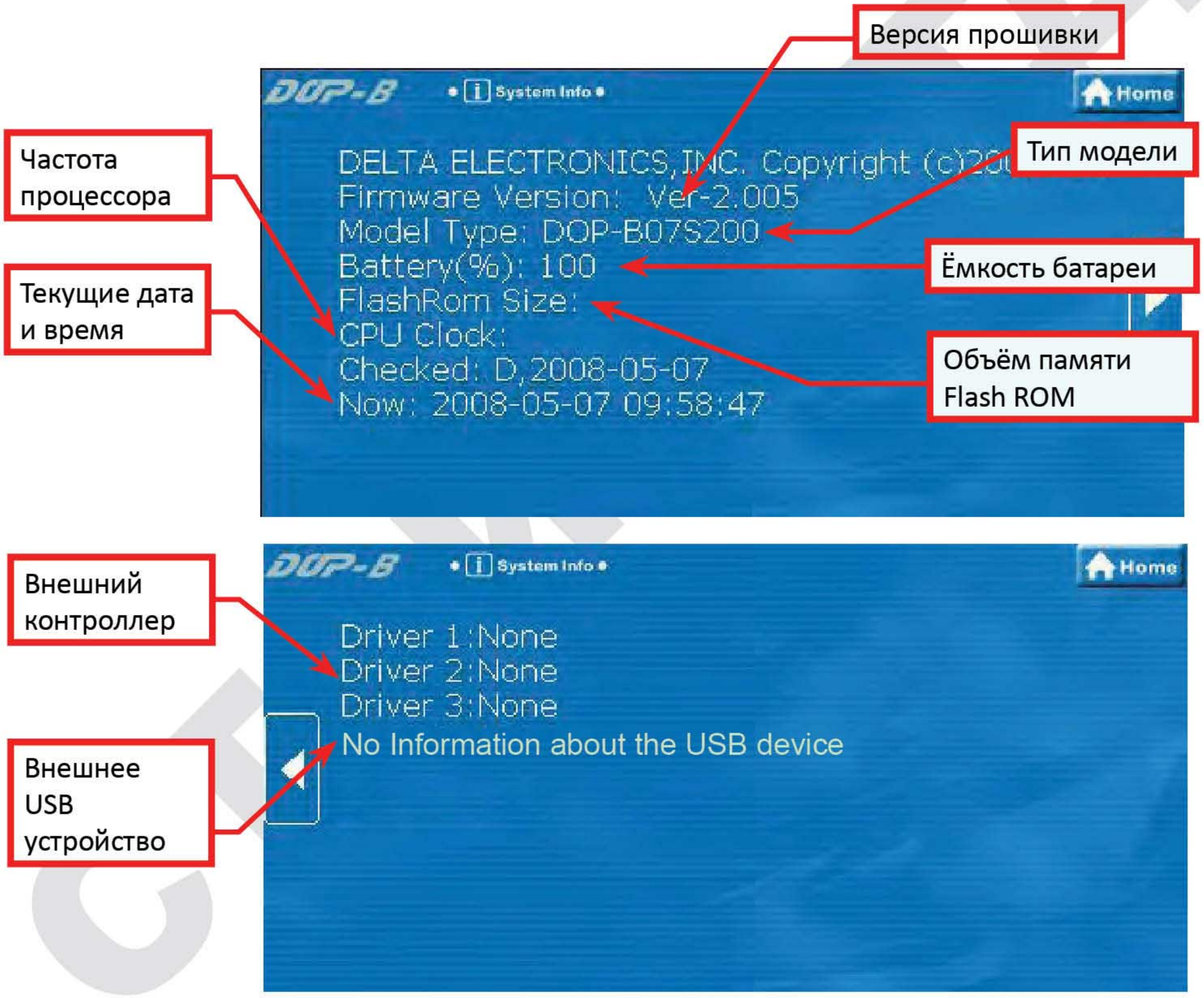
Mode 3 Режим COM2 → COM1 байпас

Выбор этого режима означает, что порт COM1 панели оператора связан с контроллером, а данные от компьютера через порт COM2 панели оператора

могут передаваться в контроллер.

(COM2 является источником данных, а порт COM1 - портом передачи данных в контроллер.)

5.4 Информация о системе.



Функциональные кнопки на панели оператора:

PREVIOUS PAGE Предыдущая страница	**NEXT PAGE** Следующая страница	**EXIT** Выход
		

5.5 HMI Doctor Menu (Тестирование аппаратной части)

Эта опция предоставляет простые текстовые программы для легкого тестирования аппаратной части. Доступны 7 простых опций: LCD, Touch Panel, Buzzer, LED, USB, ADC Channel, и Hot Key.

Если изображение на экране отображено в серых тонах, то это значит, что вы видите только превью экрана данной настройки.

Для входа в меню данной настройки кликните по иконке меню. После этого экран меню должен стать цветным.

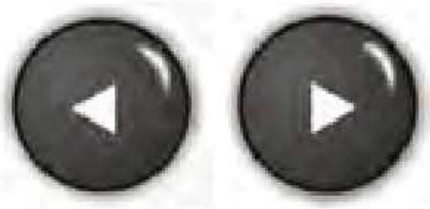
Возврат в системное меню

Нажимая можно переключать опции меню.

Опция выбрана, когда подложка имеет желтый цвет.

Опция не выбрана когда подложка имеет голубой цвет

Функциональные кнопки:

		
Выбор режима	Вход в выбранный режим	Выход из режима тестирования

 **White Screen Test**

Тест белого экрана, позволяет выявить темные пиксели и другие дефекты экрана.

 **Black Screen Test**

Тест черного экрана, позволяет выявить красные, голубые, зелёные или белые яркие пиксели на черном фоне.

 **Red Screen Test**

Тест красного экрана, позволяет выявить тёмные пиксели на красном фоне.

 **Green Screen Test**

Тест зелёного экрана, позволяет тёмные пиксели на зелёном фоне.

 **Blue Screen Test**

Тест голубого экрана, позволяет тёмные пиксели на голубом фоне.

 **Draw Line Test**

Тест рисования линии, позволяет определить рассогласование между местом рисования линии и местом нахождения её изображения на экране. При рассогласовании необходимо провести калибровку экрана.

Buzzer/LED Test

- Тест звучания зуммера.
- Проверка свечения светодиодов: красного, голубого и зелёного .

ADC Test

Тест правильности каналов аналого-цифрового преобразователя, которые применяются для измерения силы нажатия, напряжения питания, температуры и т.д.

USB Test

Проверка правильной работы подключенного USB устройства.

Key Test

Тест функциональных кнопок. При нажатии кнопок  ~  панели оператора соответствующие значки на экране будут высвечиваться. Если реакция на нажатие соответствующей кнопки отсутствует, кнопка считается неисправной.

Color Screen Test

Контроль уровня RGB каналов и насыщенности.

Приложение A.
Новые функциональные возможности

A.1 Новые макрокоманды

■ EXPORT

Функция макрокоманды **EXPORT macro** такая же, как функция экранного элемента **Report List** (отчёт).

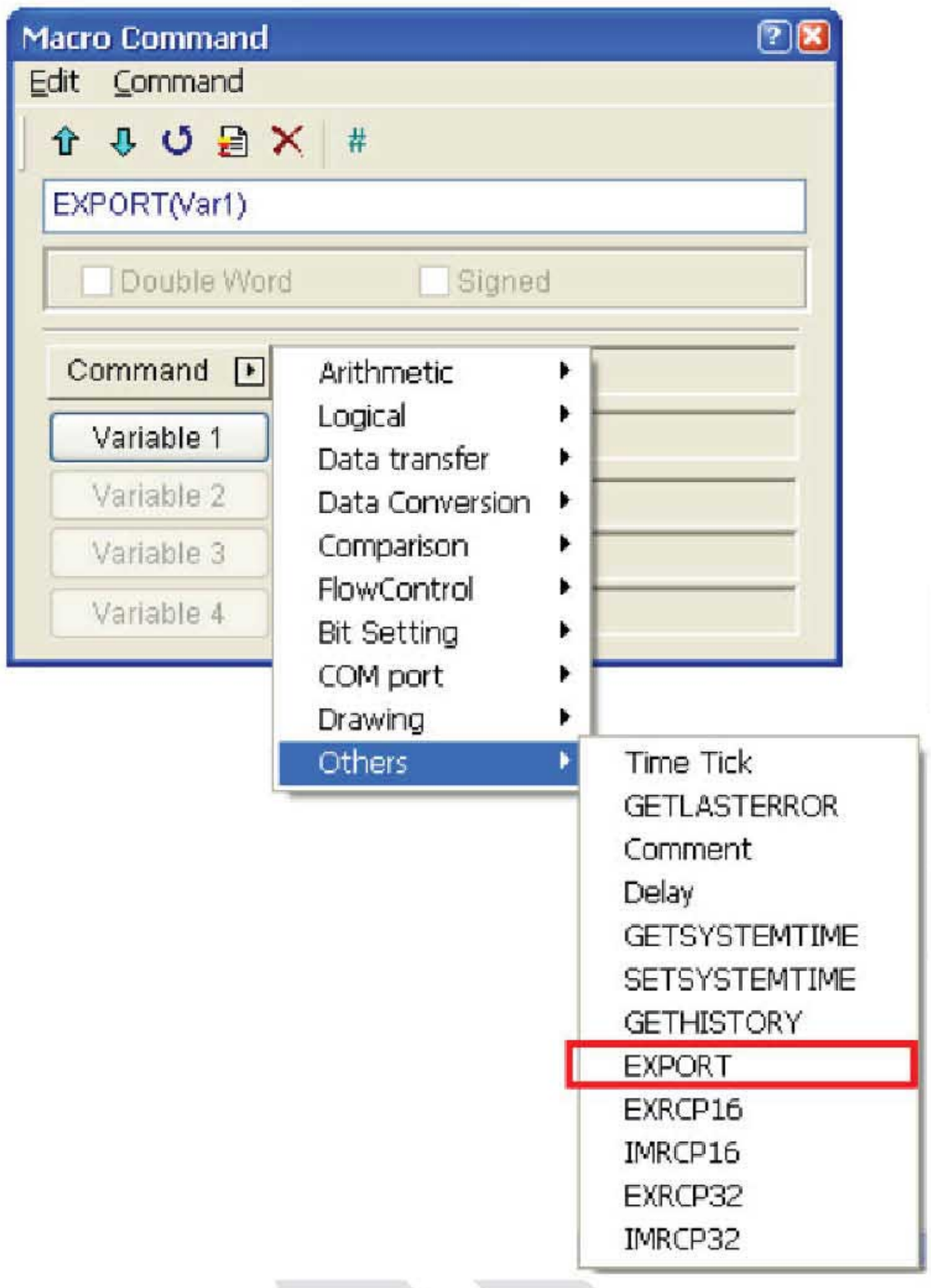
Выражение: EXPORT(Var1)

Переменная Var1: Регистр внутренней памяти или константа.

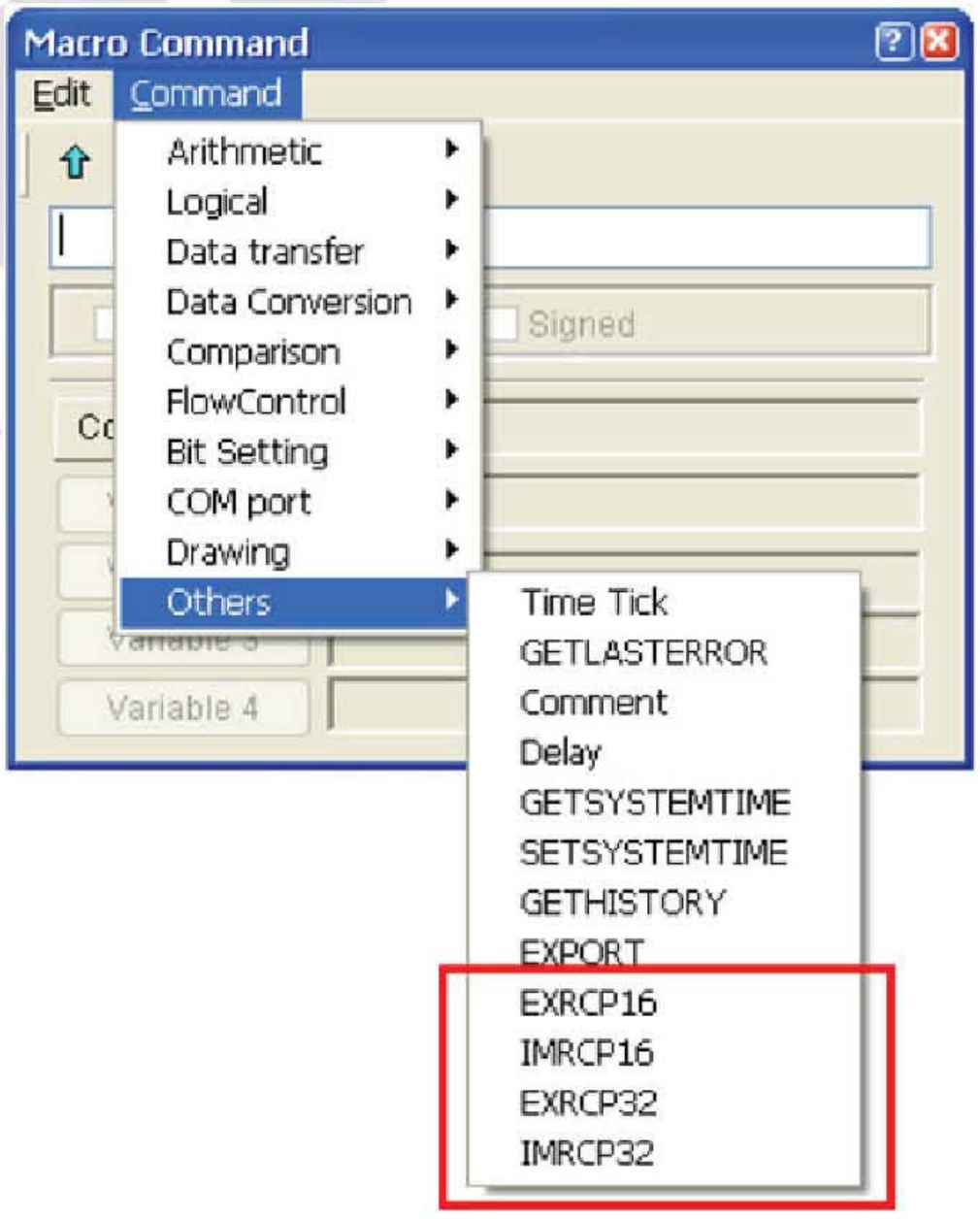
0: запись данных на SD карту

1: Запись данных на USB диск

2: Отправка данных на принтер для их распечатки.



■ Макрокоманды IMRCP16/32, EXRCP16/32



Макрокоманды **IMRCP16/32** и **EXRCP16/32** применяются для записи и чтения рецептов.

Экспорт рецептов в 16-битном формате

Выражение: Var1 = EXRCP16(Var2, Var3)

Экспорт 16-битных данных рецептов из регистров, адресуемых переменной Var2 в регистры Var3 (в виде Excel CSV файла), и сохранение в регистре по адресу Var1.

Импорт рецептов в 16-битном формате

Выражение: Var1 = IMRCP16(Var2, Var3)

Импорт рецептов в 16-битном формате в регистры, адресуемые переменной Var2 из регистров внешней памяти Var3 (в виде Excel CSV файла) и сохранение в регистре по адресу Var1.

Экспорт рецептов в 32-битном формате

Выражение: Var1 = EXRCP32(Var2, Var3)

Экспорт 32-битных данных рецептов из регистров, адресуемых переменной переменной Var2 в регистры Var3 (в виде Excel CSV файла), и сохранение в регистре по адресу Var1.

Импорт рецептов в 16-битном формате

Выражение: Var1 = IMRCP32(Var2, Var3)

Импорт рецептов в 32 битном формате в регистры, адресуемые переменной Var2 из регистров внешней памяти Var3 (в виде Excel CSV файла) и сохранение в регистре по адресу Var1.

Описание:

Var1: Результат после обмена

1: Обмен прошёл нормально

0: Обмен прошёл неправильно

Var2: Адрес импорта /экспорта данных

Var3: Внешняя память

2: USB диск.

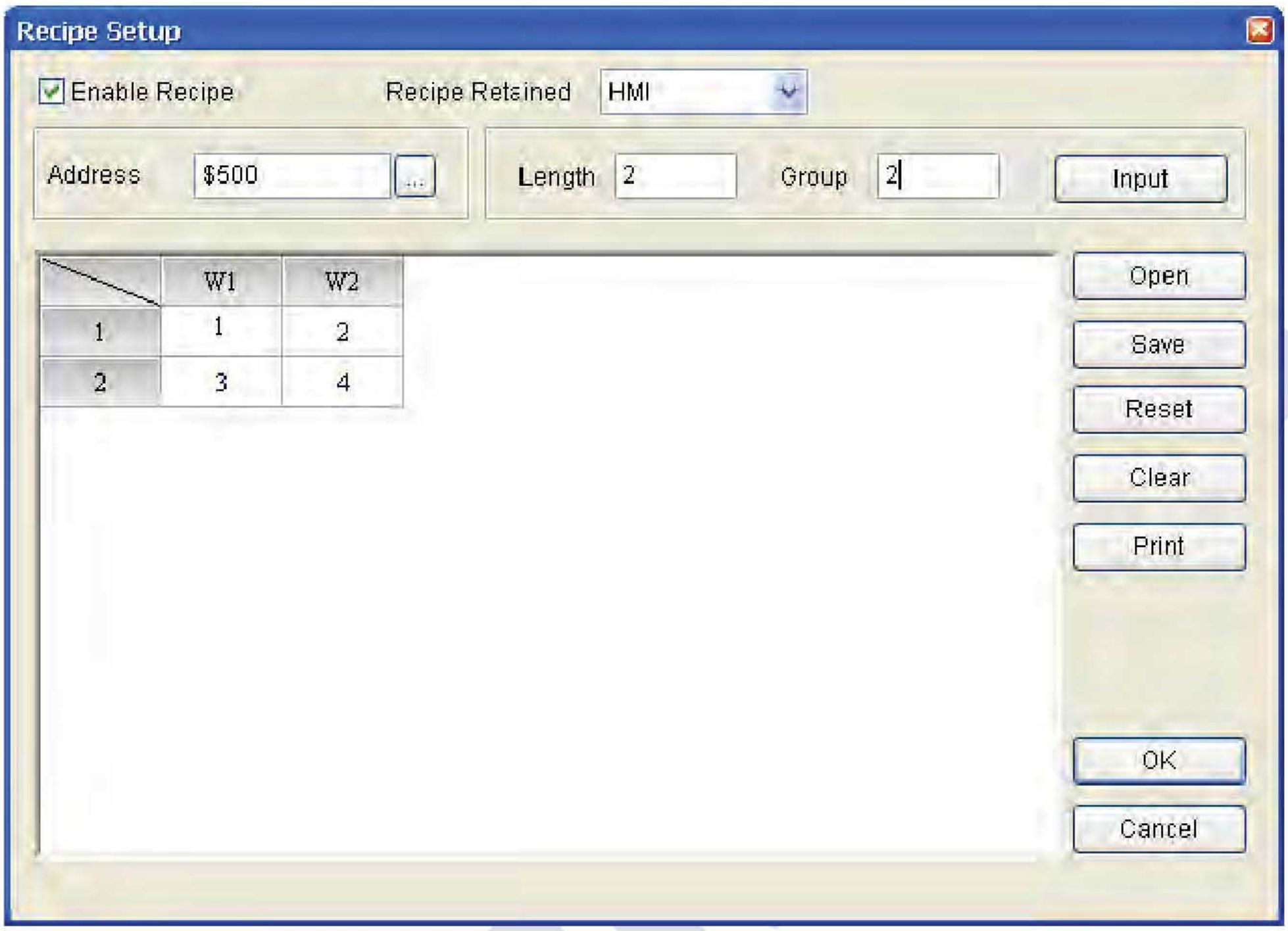
3: SD карта.

Пример

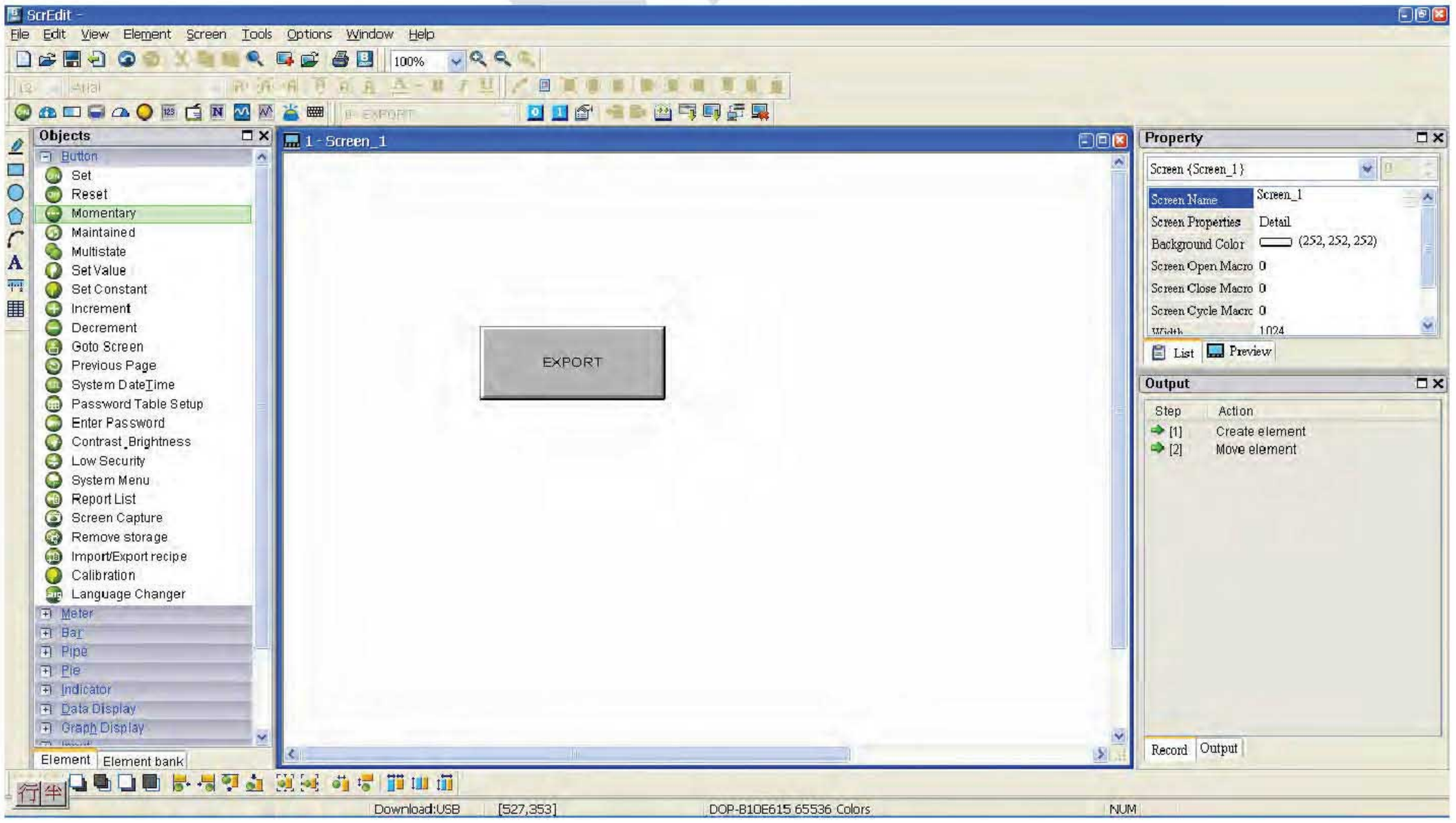
Предположим, что пользователь хочет записать данные рецептов на USB носитель.

Сначала, выделить курсором мыши и подтвердить выбор копкой мыши,

Options > Recipe в меню для открытия данных рецептов в 16-битном формате.

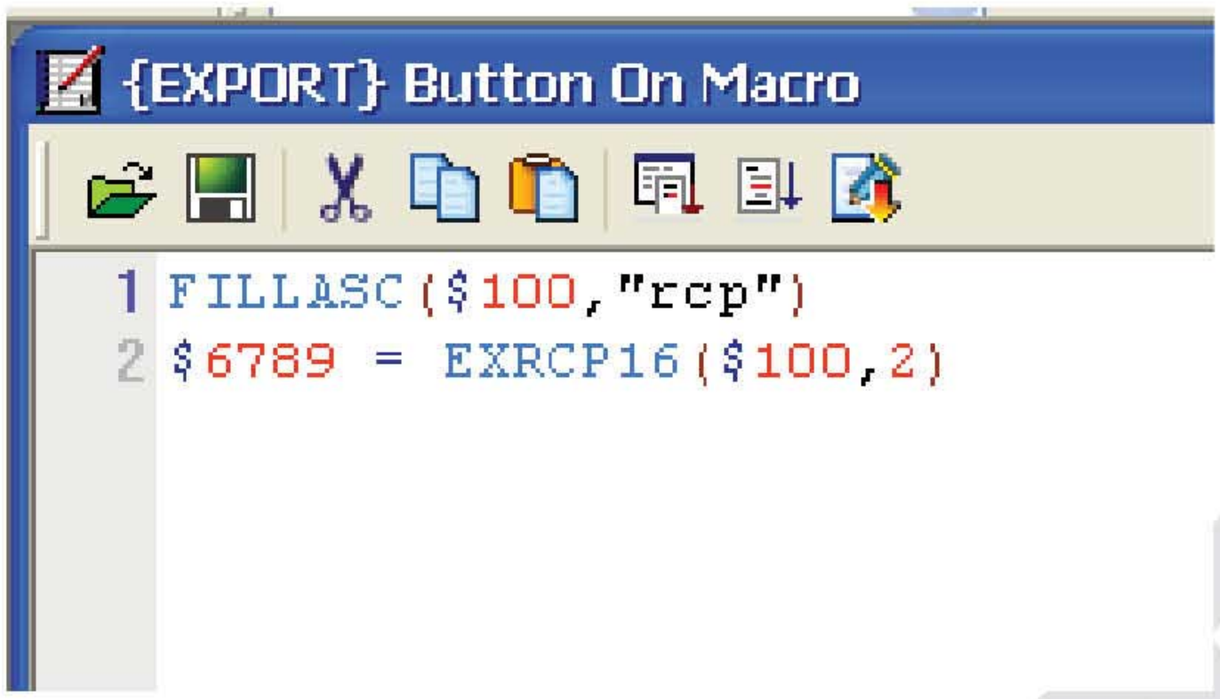


Далее, создадим экранный кнопочный элемент **Momentary**



Для правильной настройки этого элемента, необходимо, как показано на рисунке

записать макрос включения (**On MACRO**)

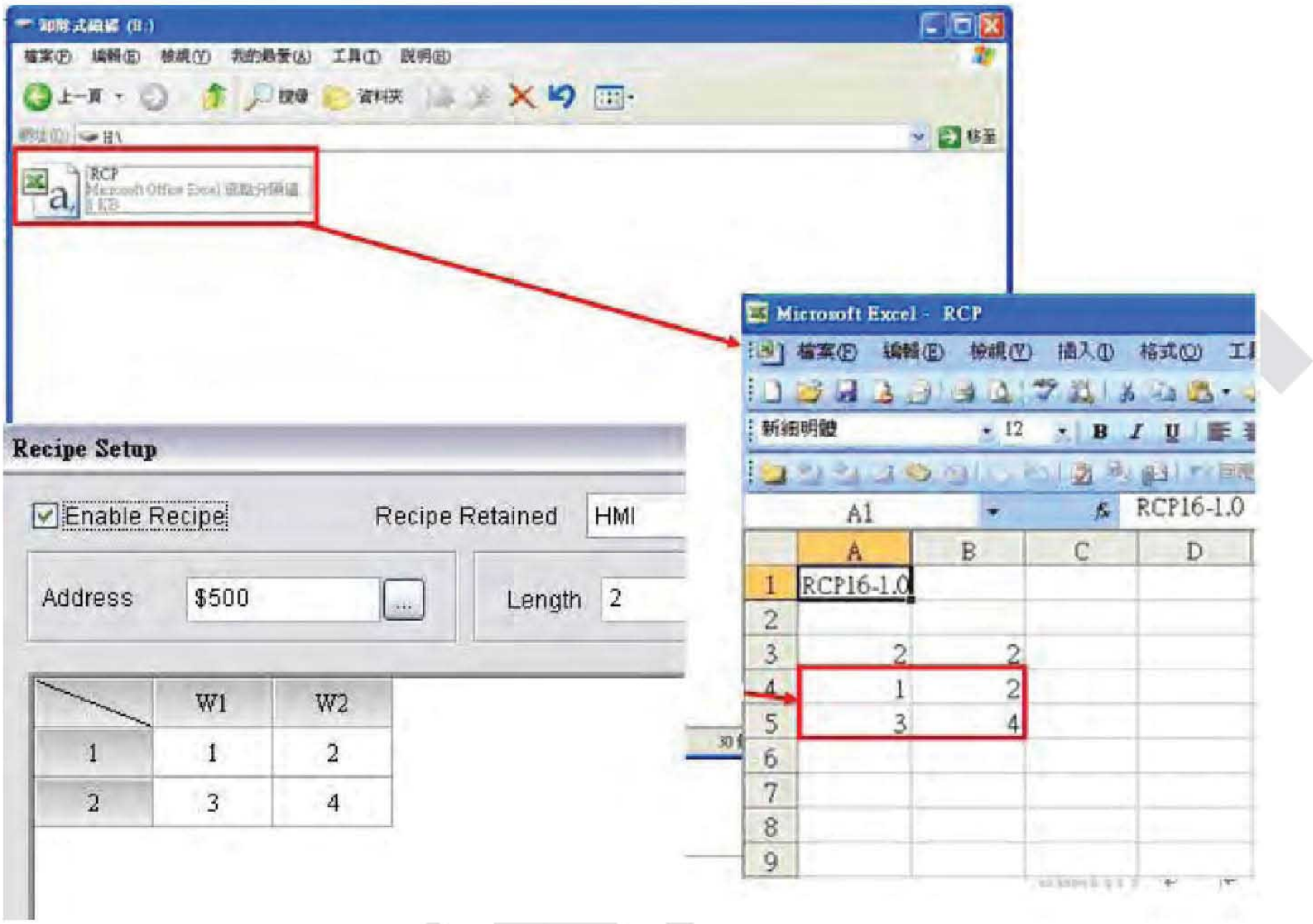


FILLASC($100, “rcp”)	Записать имя файла как RCP.
$6789 = EXRCP16($100, 2)	Экспортировать данные рецепта из регистра $100 на USB носителе и сохранить их в регистре $6789

Провести компиляцию и программу в память панели (вид экрана приведён ниже.)

При нажатии кнопки **EXPORT**, данные из 16-битном формата преобразуются в Excel CSV файл и немедленно запишутся в USB носитель.

Файл рецептов на USB носителе:



■ Макрокоманды STATIONON/STATIONOFF

С помощь макрокоманд STATIONON / STATIONOFF пользователь имеет возможность разрешать или запрещать обмен между панелью оператора и внешними контролерами.

Выражение: STATIONON (Var1, Var2) STATIONOFF (Var1, Var2)

Var1 и Var2 могут задавать адреса внешней памяти или быть константой.

Var1: Коммуникационный порт

0: COM1

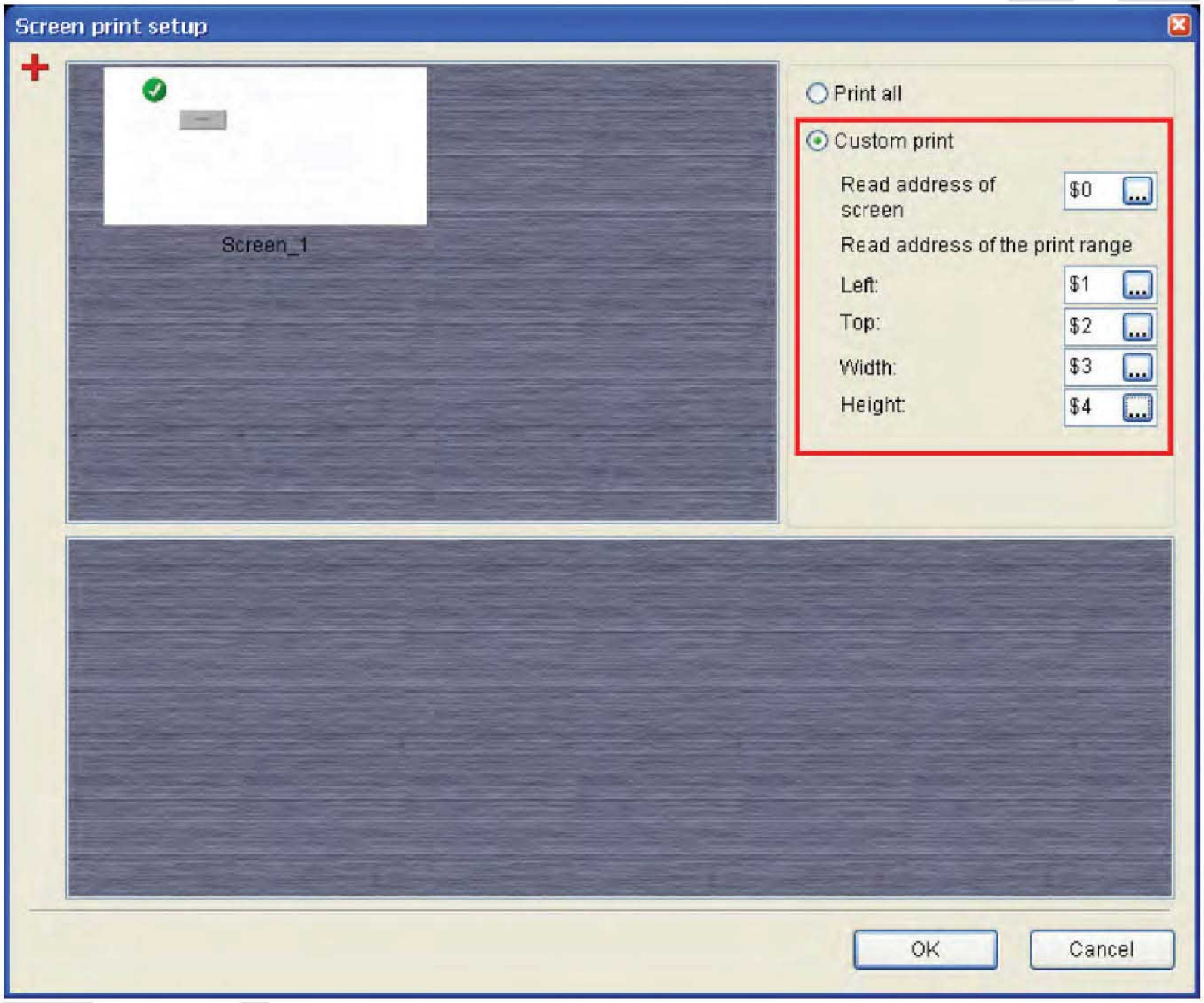
1: COM2

2: COM3

Var2: Адрес контроллера

A.2 Настройка печати - задание области печати

В окне настройки печати **Custom Print** возможно задать пользовательские значения настройки области печати.



Функции каждого регистра приведены в таблице:

$0	Чтение адреса экрана	Задание номера экрана. При значении параметра равным '0' производится печать всех выведенных в диалоговое окно печати экранов.
$1	Нижняя точка	Адрес X координаты
$2	Верхняя точка	Адрес Y координаты
$3	Ширина	Размер области печати по ширине
$4	Высота	Размер области печати по высоте.

Данная функция может применяться совместно с записью отчёта, макрокомандами экспорта данных.

A.3 Новые экранные элементы кнопочного ввода

■ Calibration - Калибровка

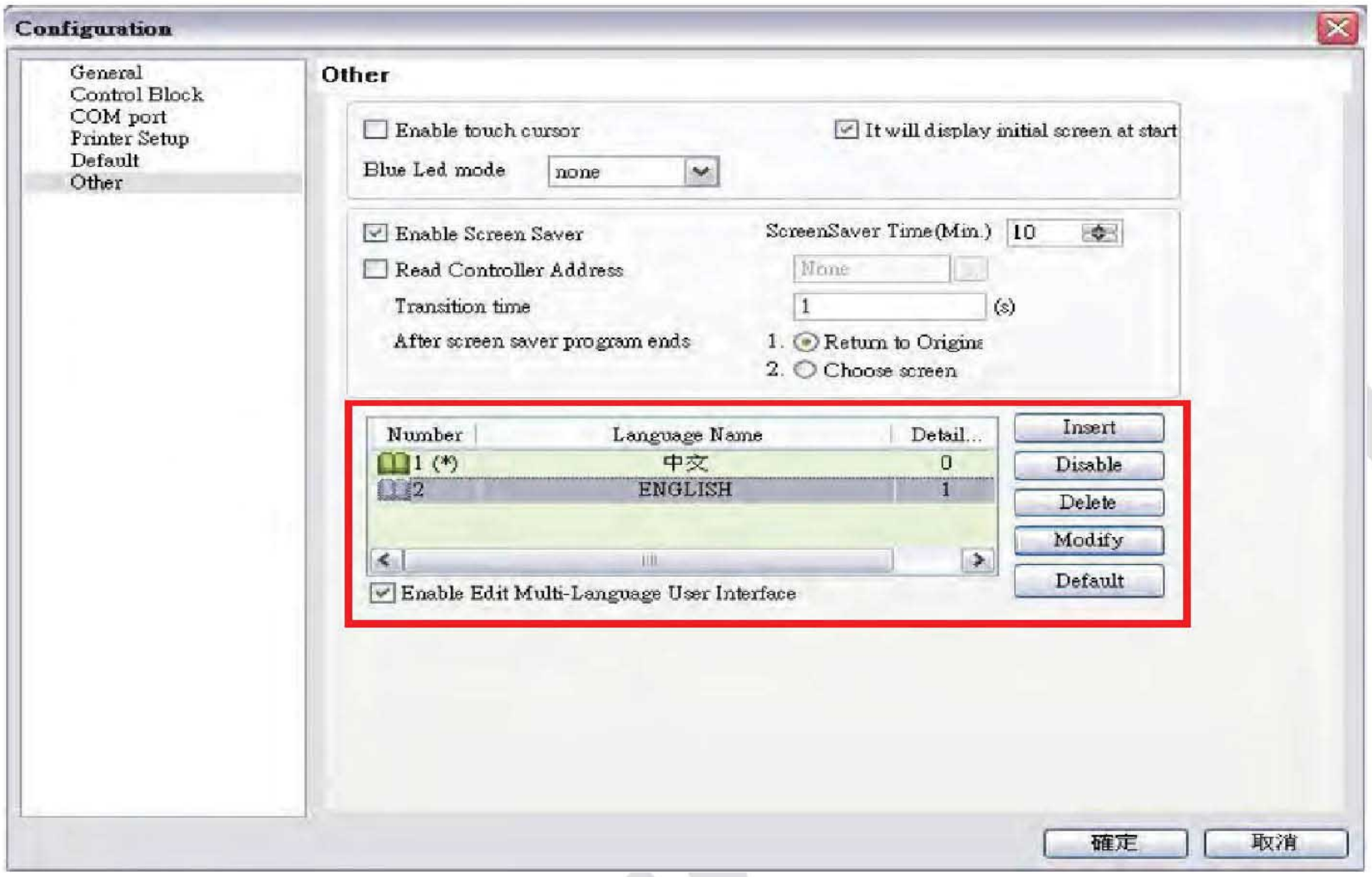
При нажатии кнопки **Calibration** на экране, пользователь получает возможность провести калибровку экрана непосредственно, не входя в системное меню.

■ Language Changer - Смена языка экрана

При нажатии кнопки **Language Changer**, пользователь получает возможность провести переключения языка непосредственно, не входя в системное меню.

Как использовать элемент **Language Changer** (на примере с китайским и английским языками):

Сначала, в диалоговом окне **Options > Configuration** с помощью мыши выберите **Other**, чтобы выбрать мультиязычный режим и добавить новый язык экрана.



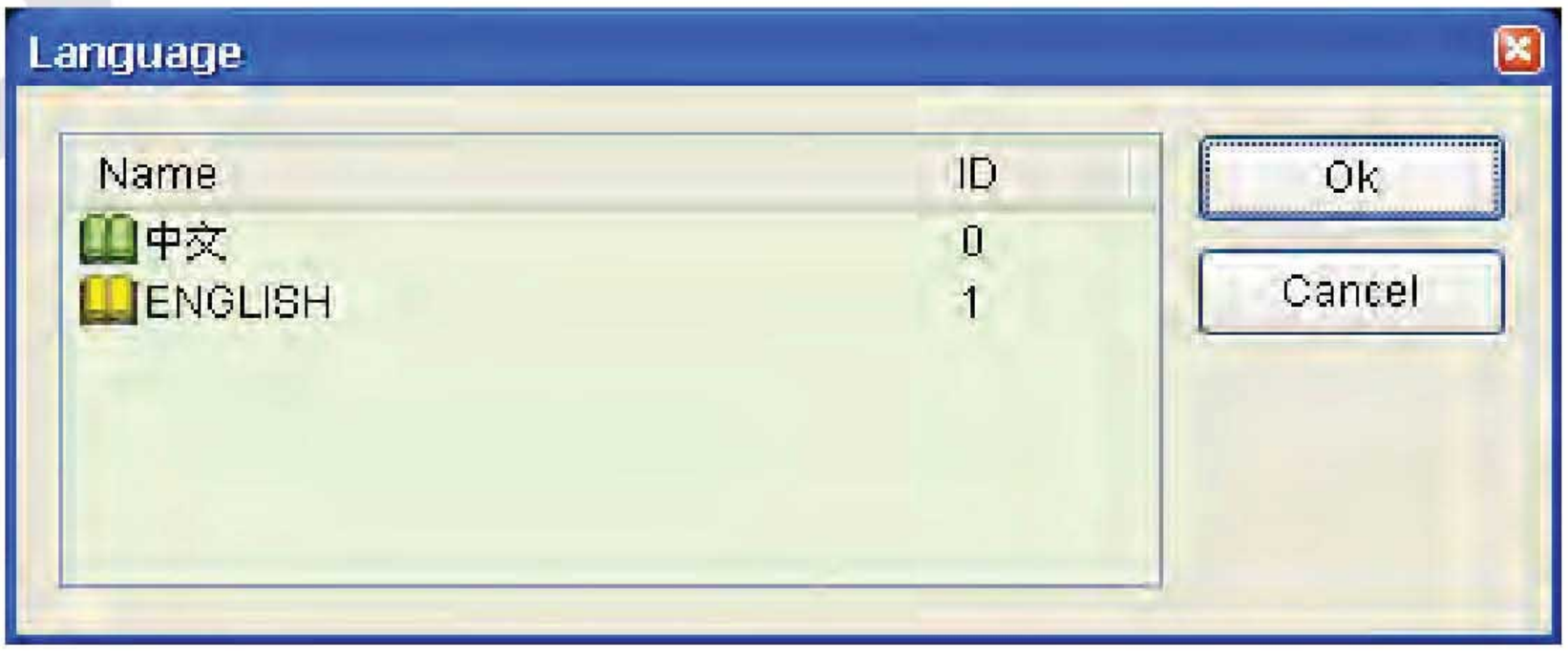
После выполнения этих шагов создайте 3 элемента на экране:

1. Создайте кнопочный элемент **Momentary**. Дважды кликнуть мышью по **Chinese** и ввести “中文版” по-китайски.

Дважды кликнуть мышью по **English** и ввести “ENGLISH VERS” по-английски.

2. Открыть кнопочный элемент **Language Changer**, и в его свойствах задать уставку языка “中文”.

3. Открыть другой кнопочный элемент **Language Changer**, и в его свойствах задать уставку языка “ENGLISH”.



Далее, проведя компиляцию, загрузить программу в память панели оператора и запустить её.

Вид изображения на экране при нажатии кнопки ENGLISH

Вид изображения на экране при нажатии кнопки “中文”

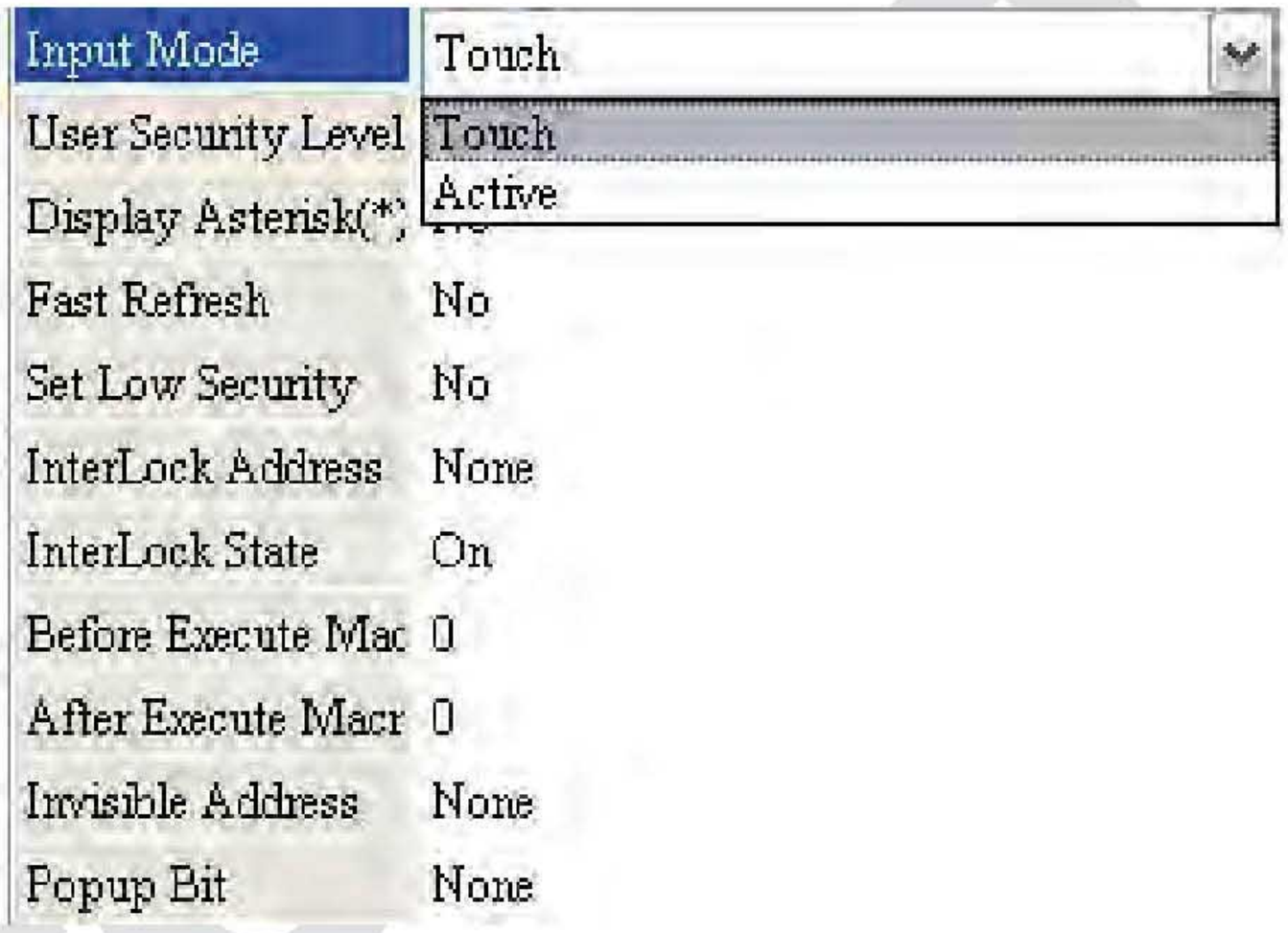
A.4 Новые элементы ввода

■ Ввод штрихкода

Этот экранный объект используется для ввода значений штрих кода, передаваемых сканером штрихкодов через USB интерфейс.

В выпадающем меню режима ввода **Input mode** имеется две опции: **Touch** или **Active**.

Уставка **Touch** режима ввода:

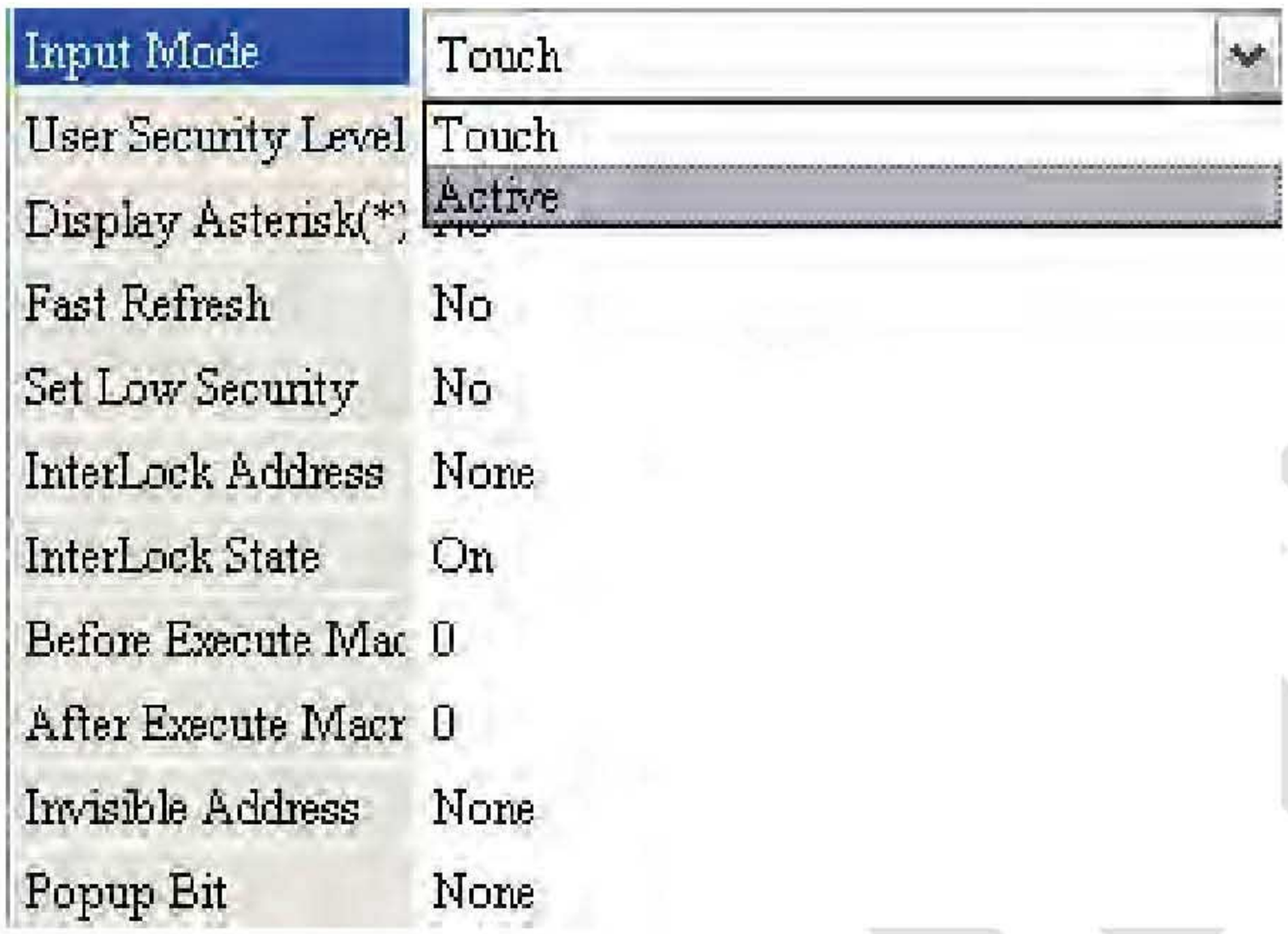


При такой настройке, при касании объекта ввода штрихкода на экране он переходит в режим ожидания данных от сканера штрихкодов.

При повторном касании режим ожидания выключается.

При наличии на экране нескольких таких объектов после приёма данных от первого сканера, система не будет автоматически принимать данные следующего сканирования. Для их приёма пользователю необходимо коснуться другого объекта. При необходимости корректировки введённых данных пользователю нужно установить **Popup Bit** опцию. Когда этот бит будет в состоянии ON на экране появится клавиатура для ввода вручную данных штрихкода.

Уставка **Active** режима ввода:



При такой настройке, сначала необходимо включить **InterLock Address** в состояние **ON**. После этого, объект ввода штрихкода перейдёт в режим готовности принимать штрихкоды.

Если на экране имеется несколько таких объектов, то система будет автоматически принимать штрихкоды до того момента, как будет сброшен **InterLock Address**.

При необходимости корректировки введённых данных пользователю нужно сначала установить экранный объект для вызова на экран клавиатуры.

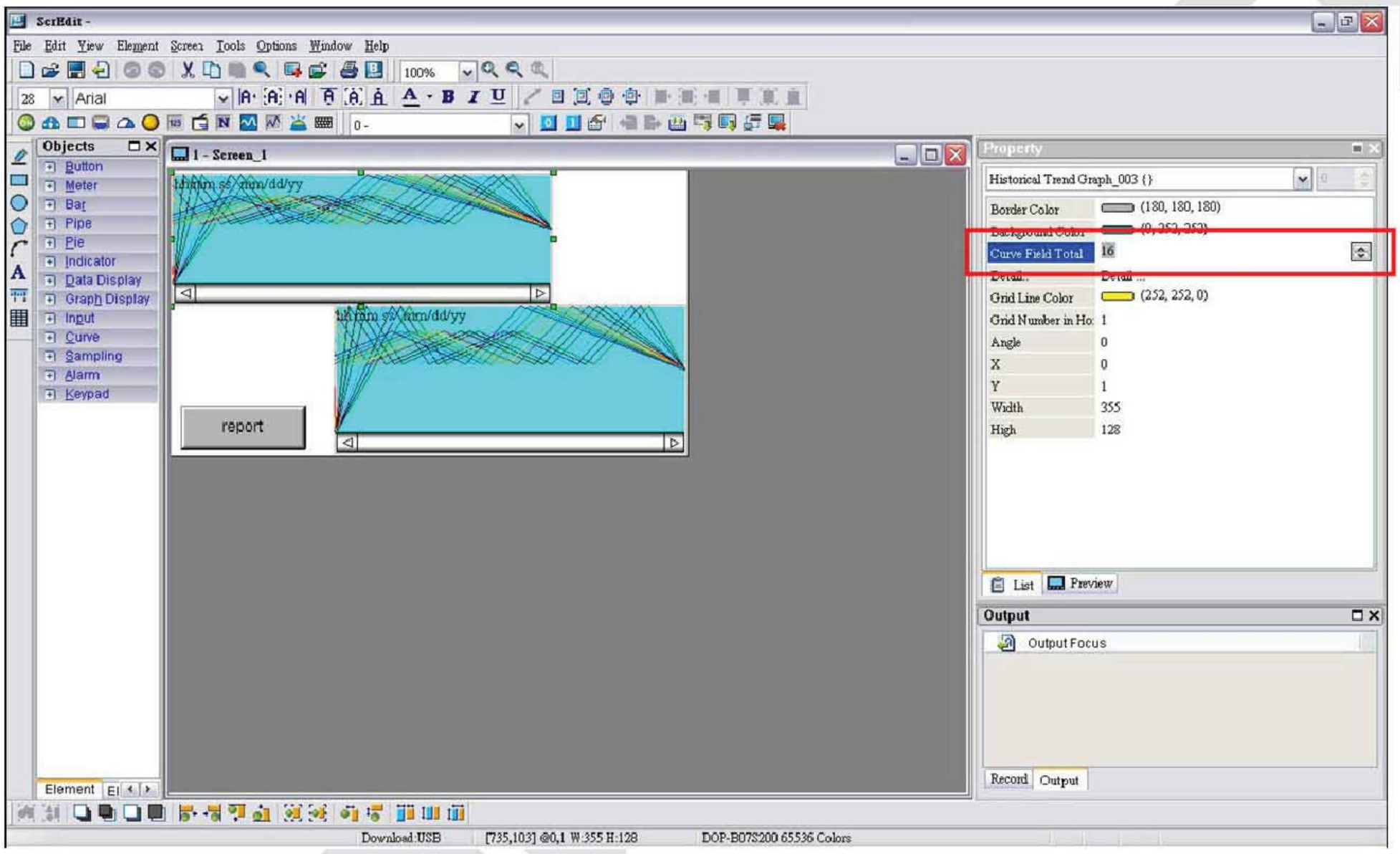
После включения **InterLock Address** пользователь может с помощью клавиатуры вводить новые значения штрихкода

Внимание: опция **Popup Bit** не активна при **Active** в режиме ввода.

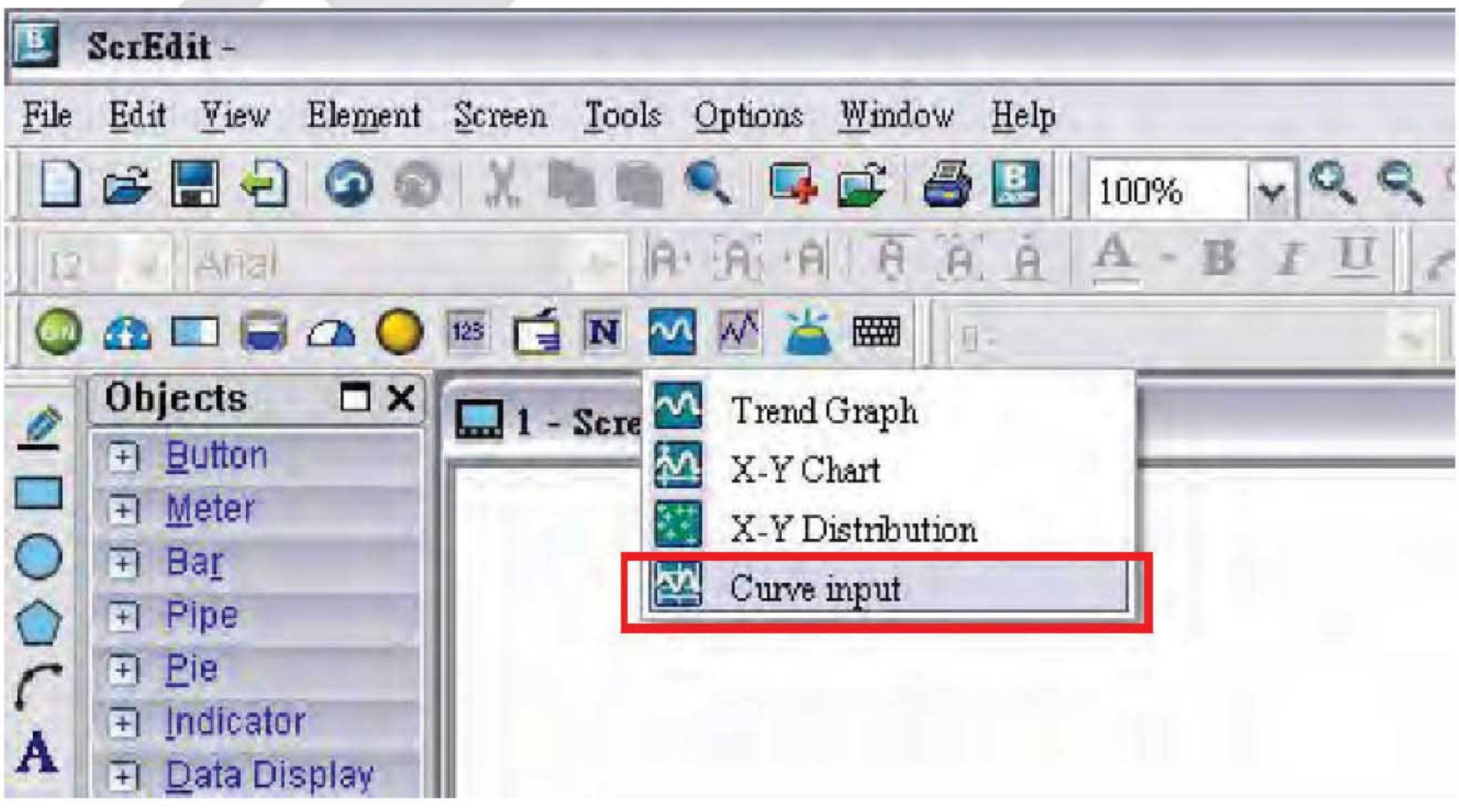
A.5 Графические объекты

■ Графики трендов

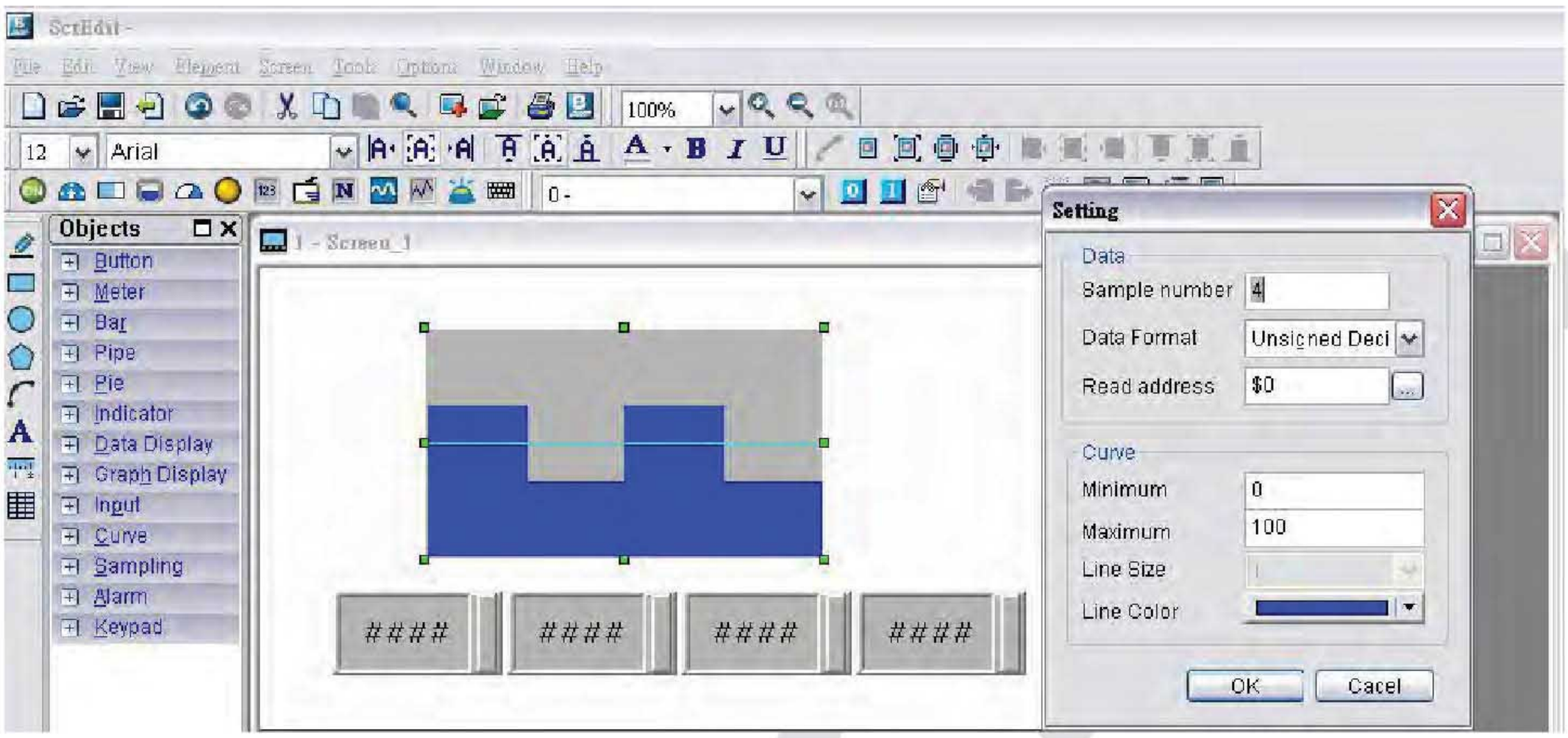
Количество окон с графиками трендов увеличено до 16.



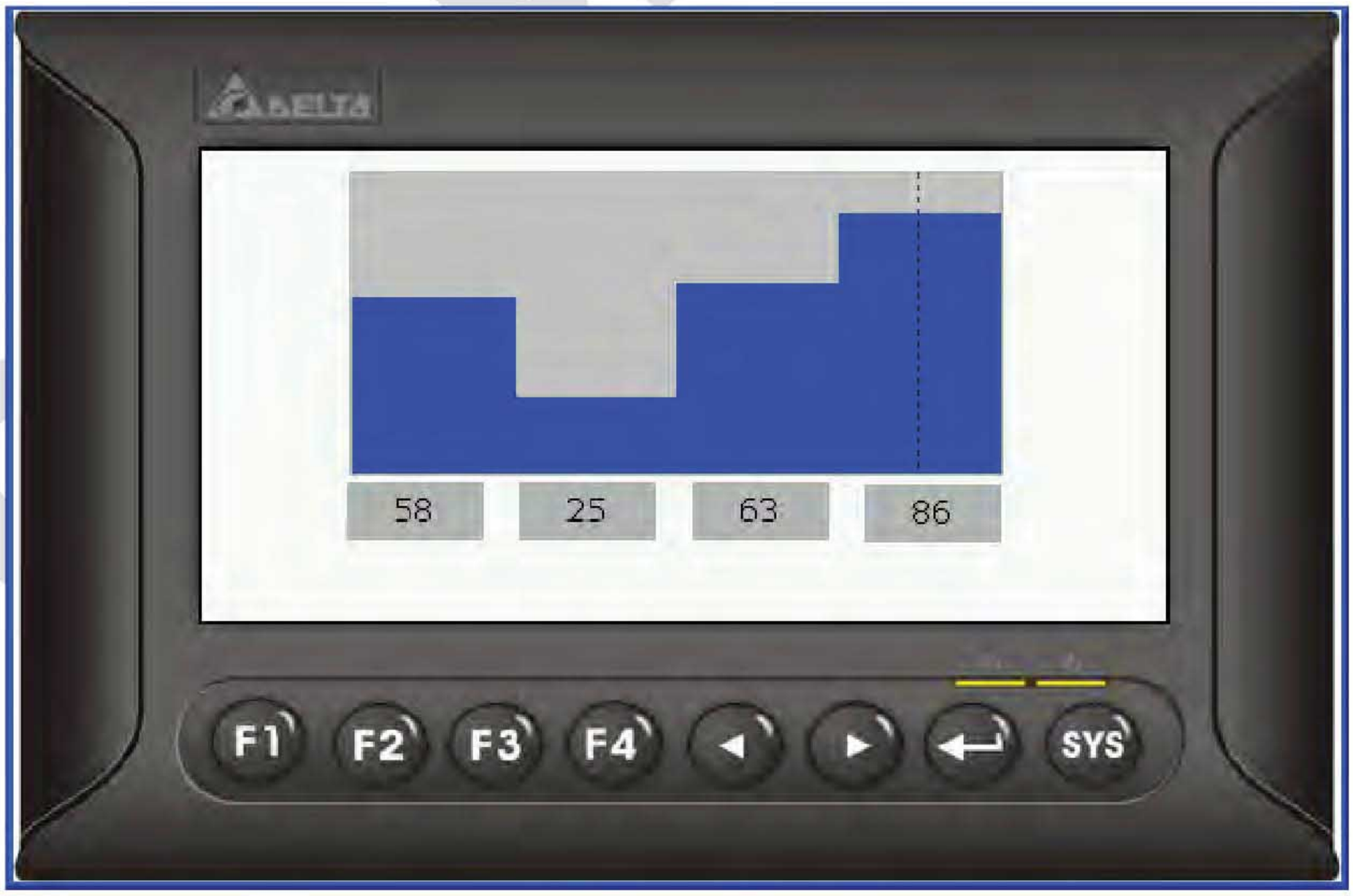
■ Curve Input (Ввод графика)



Открыть экранный объект ввода графика, задать число выборок равным 4 и адрес чтения $0, то есть значения выборок хранятся в регистрах $0, $1, $2, $3 соответственно.

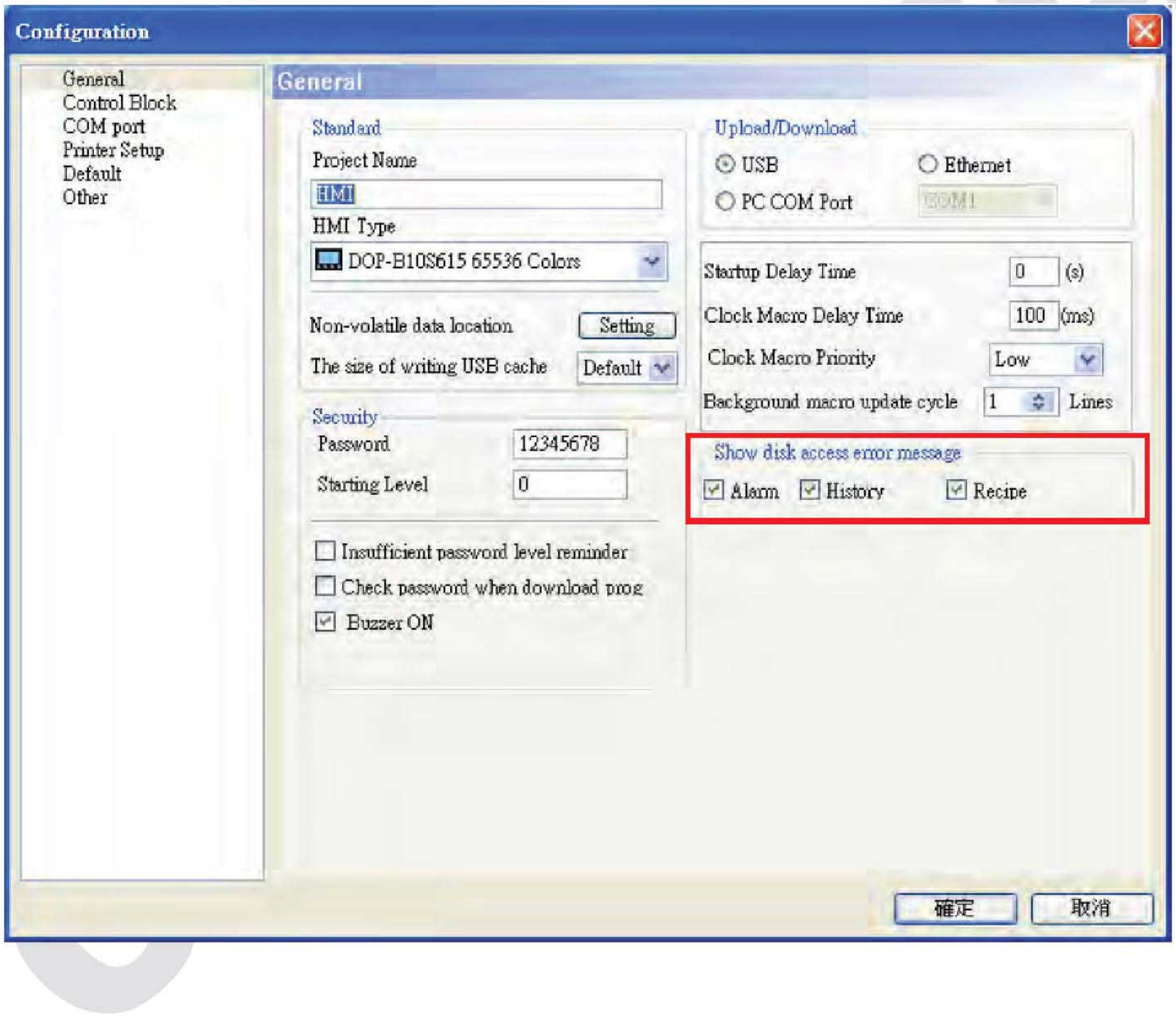


Создать на экране 4 объекта цифрового ввода с адресами $0, $1, $2, $3 соответственно. После настройки свойств, провести компиляцию и загрузить программу в панель оператора. Для ввода значений графика пользователь может либо непосредственно вводить значения на экране либо использовать цифровой ввод для каждой выборки.



A.6 Индикация ошибок, возникающих при записи на USB диск.

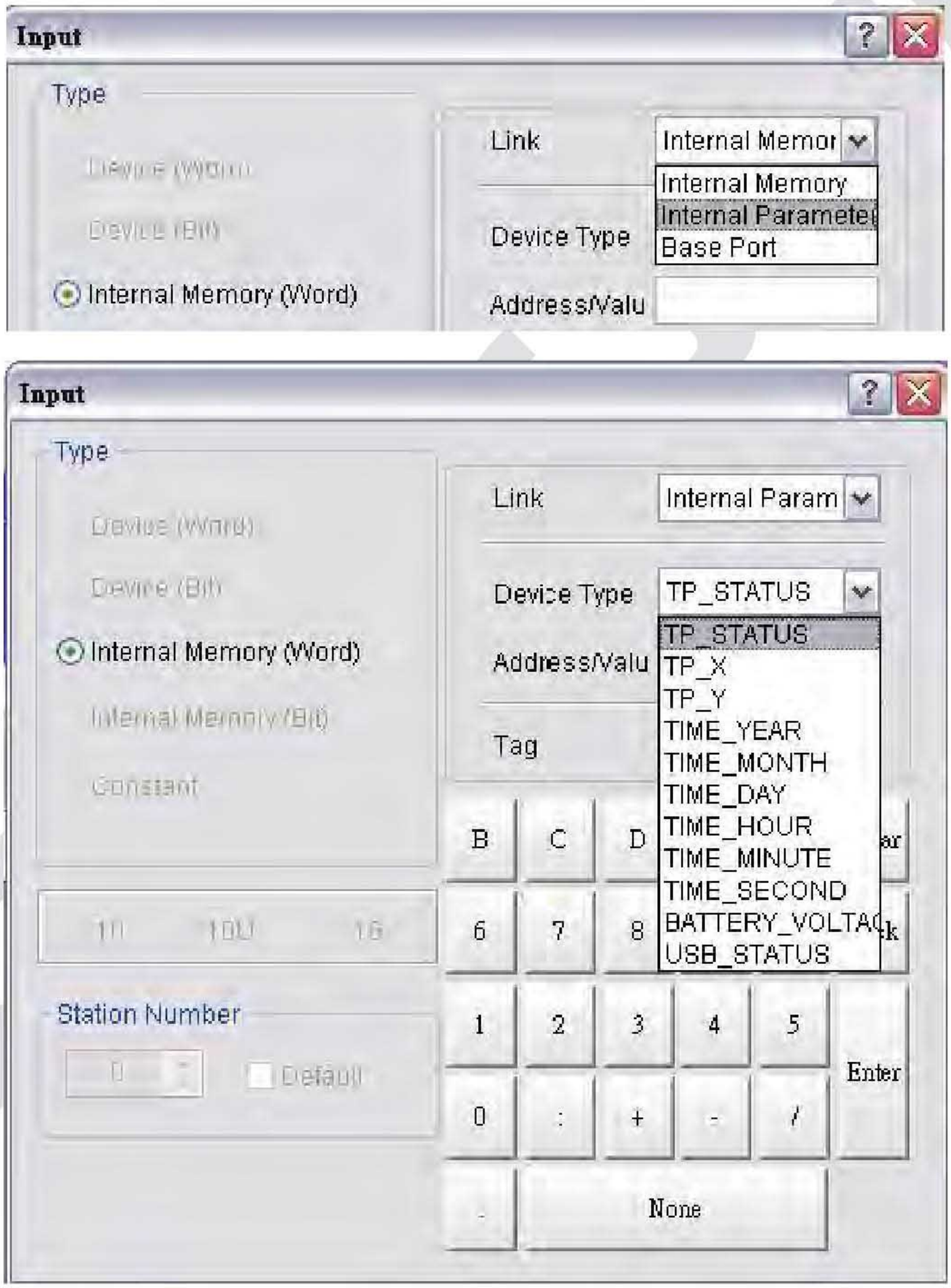
Когда установлена эта опция, если происходит какая либо ошибка при записи на USB диск аварий, рецептов, событий, то сообщение об этой ошибки выводится на экран для информирования пользователя.



A.7 Дополнительные внутренние параметры

В выпадающем меню выводятся некоторые внутренние параметры.

При активации **Device Type** (тип устройства) появляется перечень внутренних параметров, каждый из которых может быть выбран пользователем.

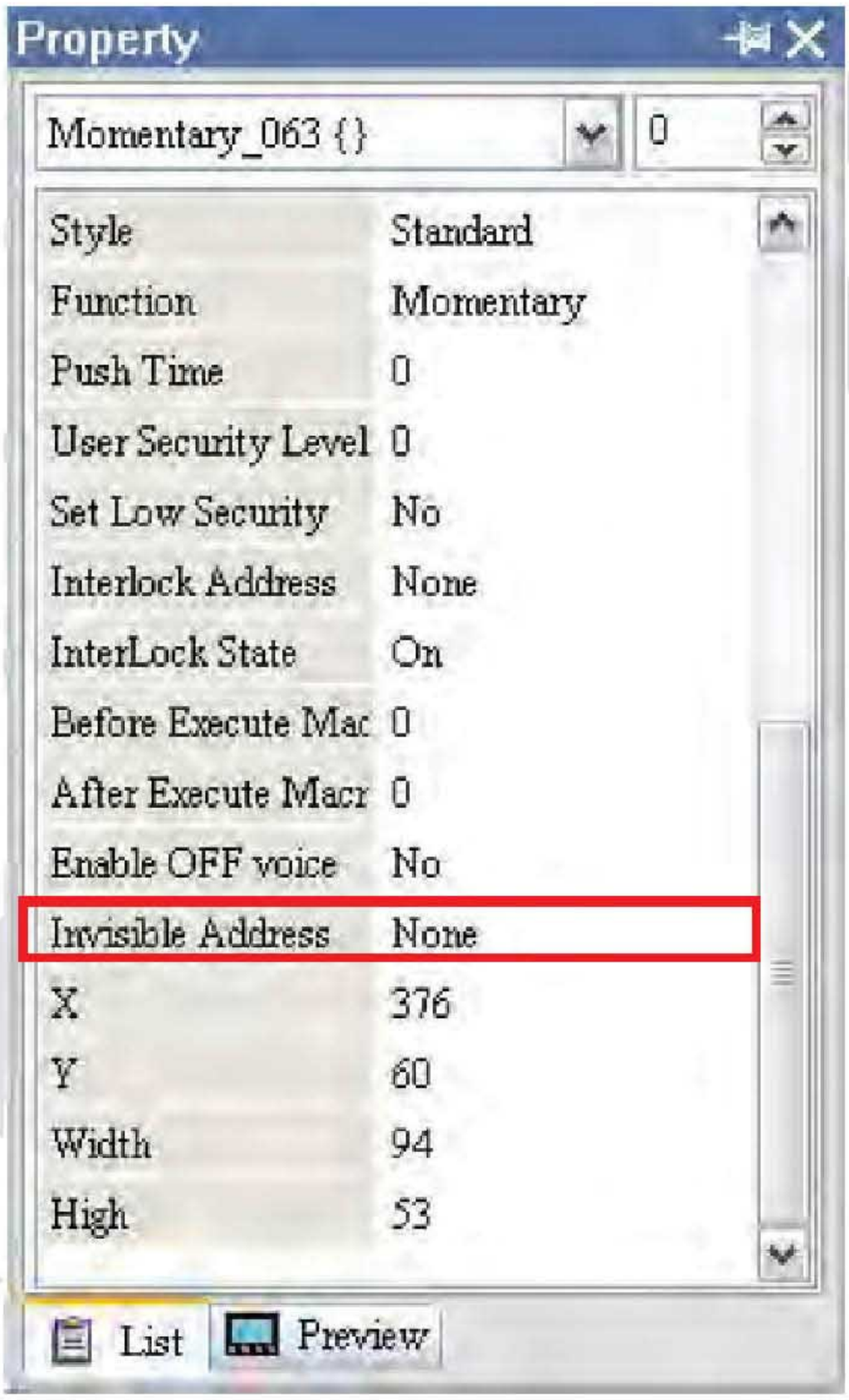


Название	Описание	Атрибут
TP_Status	Состояние Touch	Только для чтения
TP_X	X координата точки касания	Только для чтения

Название	Описание	Атрибут
TP_Y	X координата точки касания	Только для чтения
Time_Year	Год	Чтение/запись
Time_Month	Месяц	Чтение/запись
Time_Day	День	Чтение/запись
Time_Hour	Час	Чтение/запись
Time_Minute	Минута	Чтение/запись
Time_Second	Секунда	Чтение/запись Чтение/запись
Battery_Voltage	Уровень напряжения внутренней батареи (%)	Только для чтения
NET_IP1	IP адрес панели оператора Например, для IP адреса 192.168.0.1: NET_IP1 = 192 NET_IP2 = 168 NET_IP3 = 0 NET_IP4 = 1	Только для чтения
NET_IP2		Только для чтения
NET_IP3		Только для чтения
NET_IP4		Только для чтения

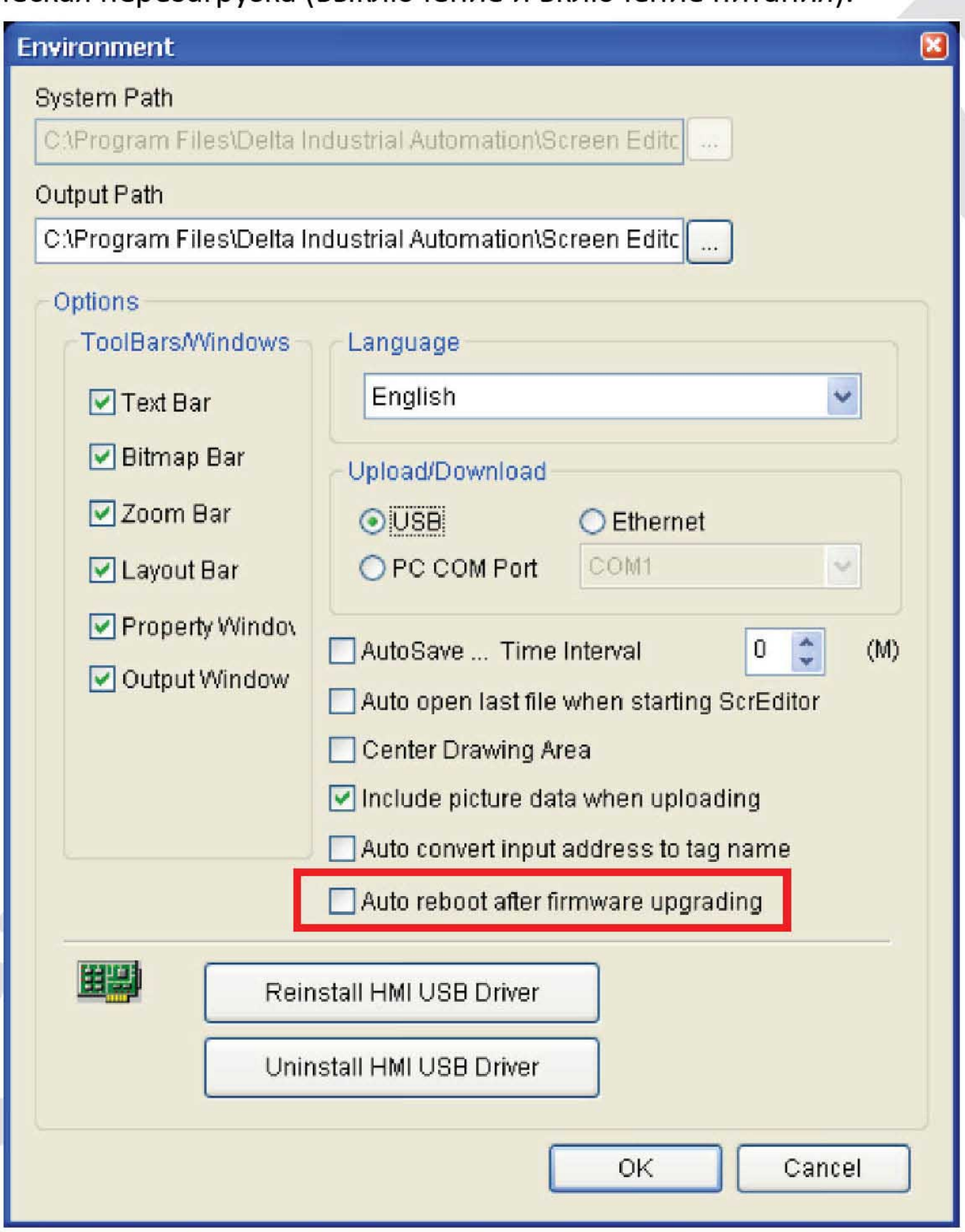
A.8 Адресация невидимых экранных объектов

При установке данной опции в состояние **On**, данный объект становится невидимым на экране. Такую функцию поддерживают все кнопочные объекты, элементы индикации ввода, объекты ввода штрихкодов.



A.9 Перезагрузка после обновления прошивки панели

При выборе такой опции, после завершения обновления прошивки происходит автоматическая перезагрузка (выключение и включение питания).



A.10 USBCommMode (Обмен данными в с USB диском)

USBCommMode - новая опция в системном меню, обеспечивающая обмен данными с USB носителем.

После обновления прошивки, пользователь, нажав системную кнопку SYS, может войти в системное меню.

Для входа в системные настройки **Touch System Setting** используются экранные кнопки и функциональная кнопка . В **System Setting Menu**, выберите **MISC.** (как показано на рисунке). Далее установить значение, USBCommMode = 1 снова соедините панель оператора с USB портом компьютера.

Нажать кнопку MISC.

Выбор опции USBCommMode

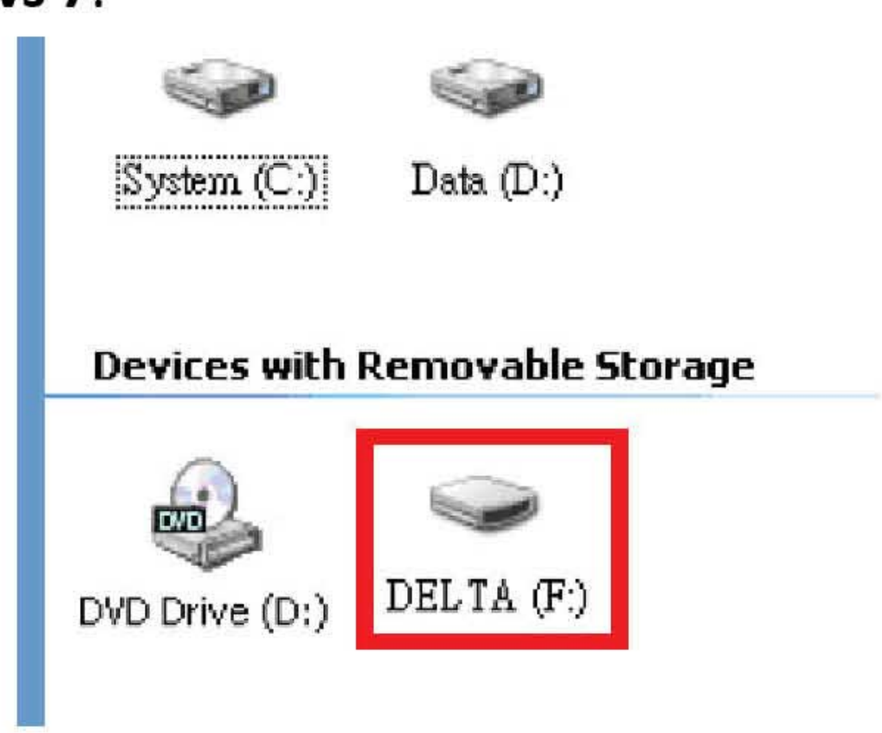
Установка USBCommMode = 0, означает, что выбран обычный режим передачи данных.

Установка USBCommMode = 1, означает выбор нового режима обмена данными с USB носителем. После установки USBCommMode =1, снова соедините панель оператора с USB портом компьютера.

Далее, дважды кликнув кнопкой мыши по значку “**My Computer**”, можно определить на рабочем столе появление нового сменного дискового устройства, называемого DELTA.

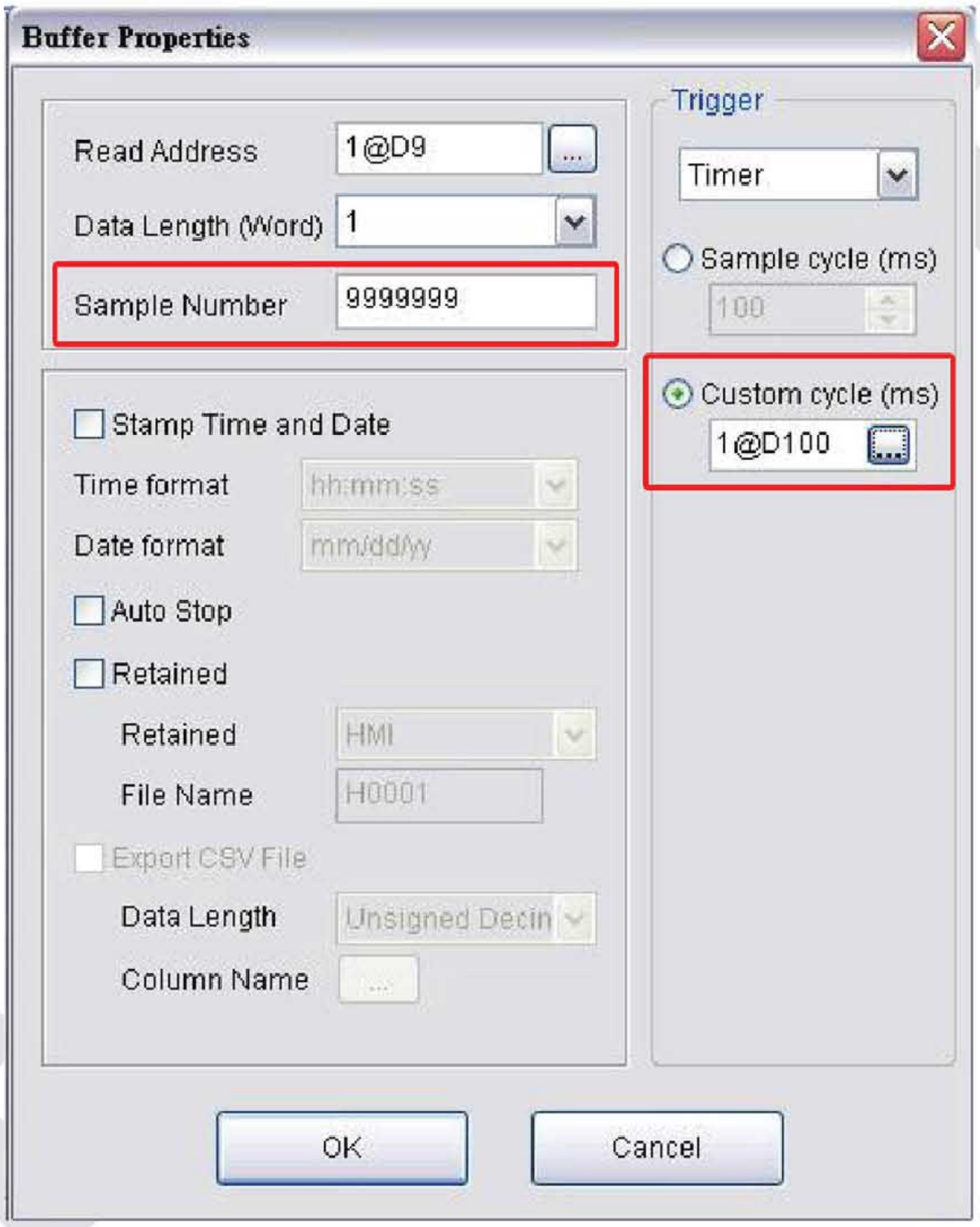
Если новое устройство подключено, то пользователь может загрузить программу с USB диска или записать её в панель с этого диска.

Эта функция совместима с операционными системами **Microsoft Windows XP**, **Windows Vista**, и **Windows 7**.



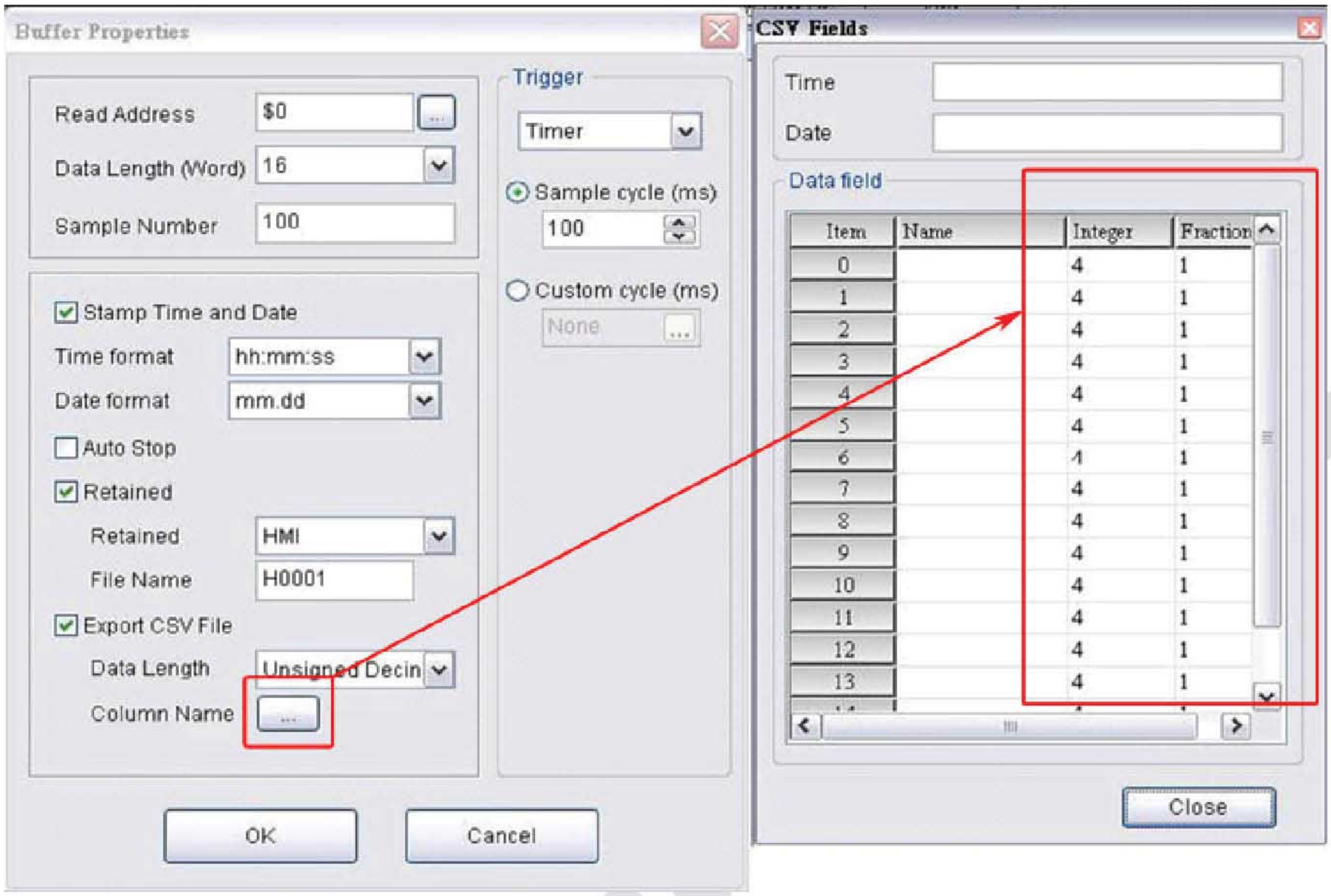
A.11 Новые функции в архиве событий.

- Число выборок увеличено до 9999999.
- Используя адрес регистра (**Register address**) можно установить пользовательское время цикла.

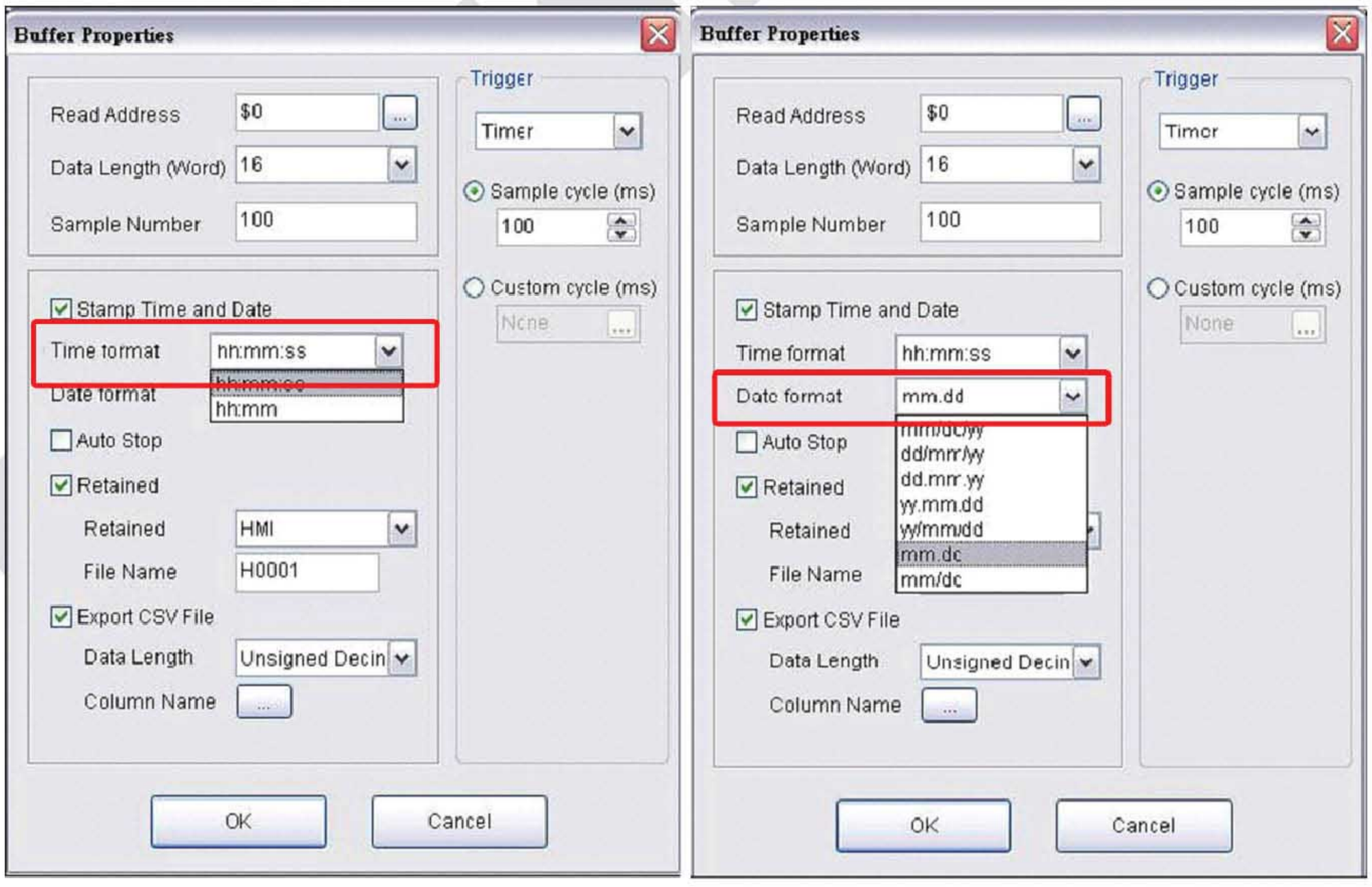


- При экспортировании CSV файла пользователь может присвоить названия колонкам таблицы.
- Добавлена новая функция — задание целой и дробной части чисел в таблице.

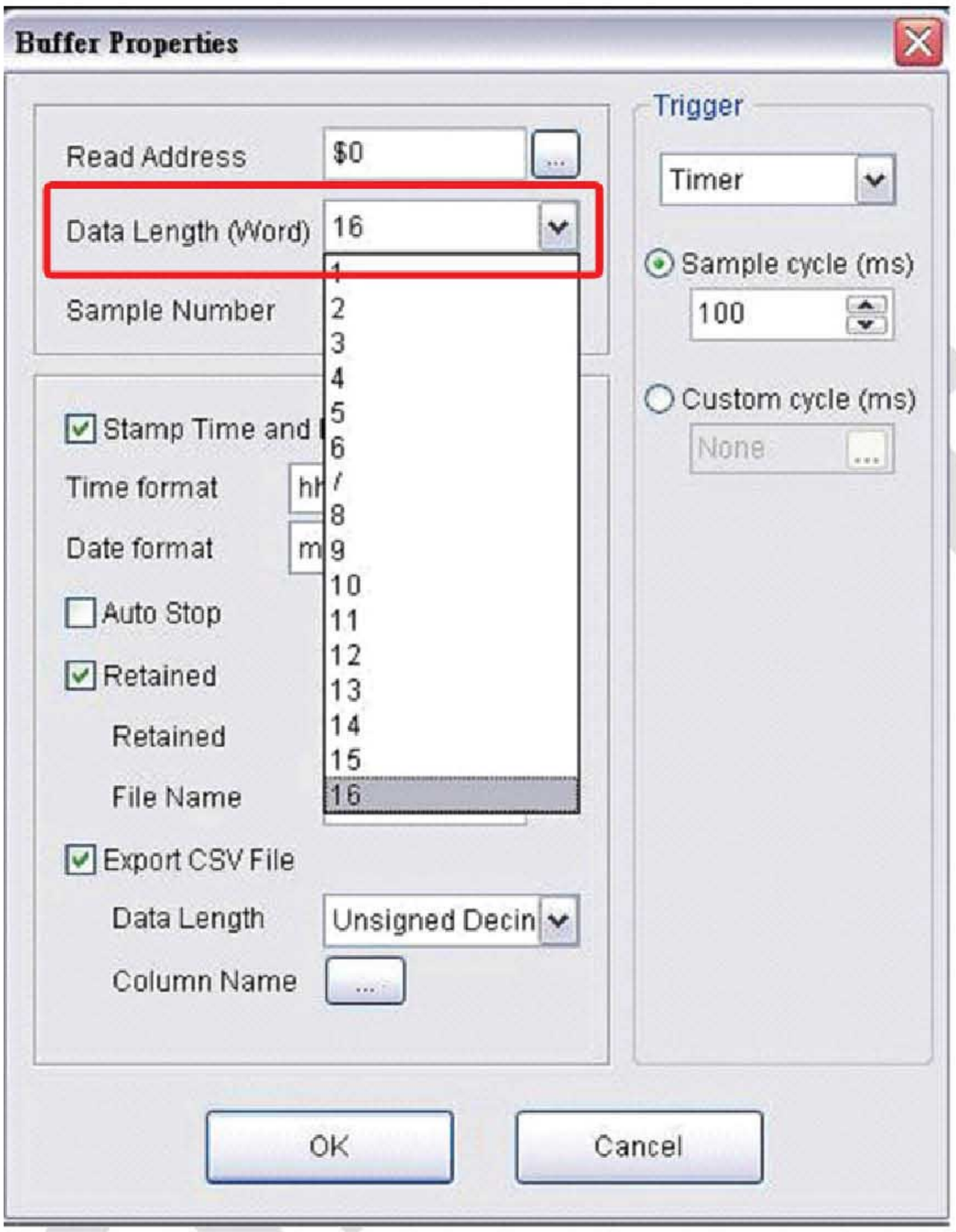
.



- Добавлен новый формат записи даты и времени



- Количество графиков при экспорте в CSV файл увеличено до 16 (в настройках архива событий)



Приложение B.
Новые функции загрузки программ

B.1 Загрузка/выгрузка программ в контроллер серии DVP

Новые кнопки появившиеся в системном меню, загрузка и выгрузка программ, позволяют загружать их в контроллеры серии DVP. Используя эти возможности пользователь может непосредственно загружать программу в контроллер DVP.

Описание процедуры:

1. Для входа в системное меню в течении 2 сек держать нажатой кнопку SYS, нажать экранную кнопку **Up/Download**

2. Нажать кнопку  для входа в **Transfer Mode**.

3. Пользователь должен выбрать **COM Port**, после этого панель оператора определит устройство DVP и автоматически с ним свяжется.

4. После установки связи панели с контроллером, нажав экранную кнопку **Upload,** пользователь может загрузить программу в контроллер, а нажав кнопку **Download -** выгрузить её из контроллера в компьютер.

Примечания:

- Формат записи имени файла программы контроллера: DVP-xxx.dvp (xxx – число 000 ... 999).

- Если программа является новым файлом, то надо выбрать **New..** и далее нажать значок **Upload** для добавления его в каталог.
- Поддерживаемые типы контроллеров и версии прошивок приведены в таблице:

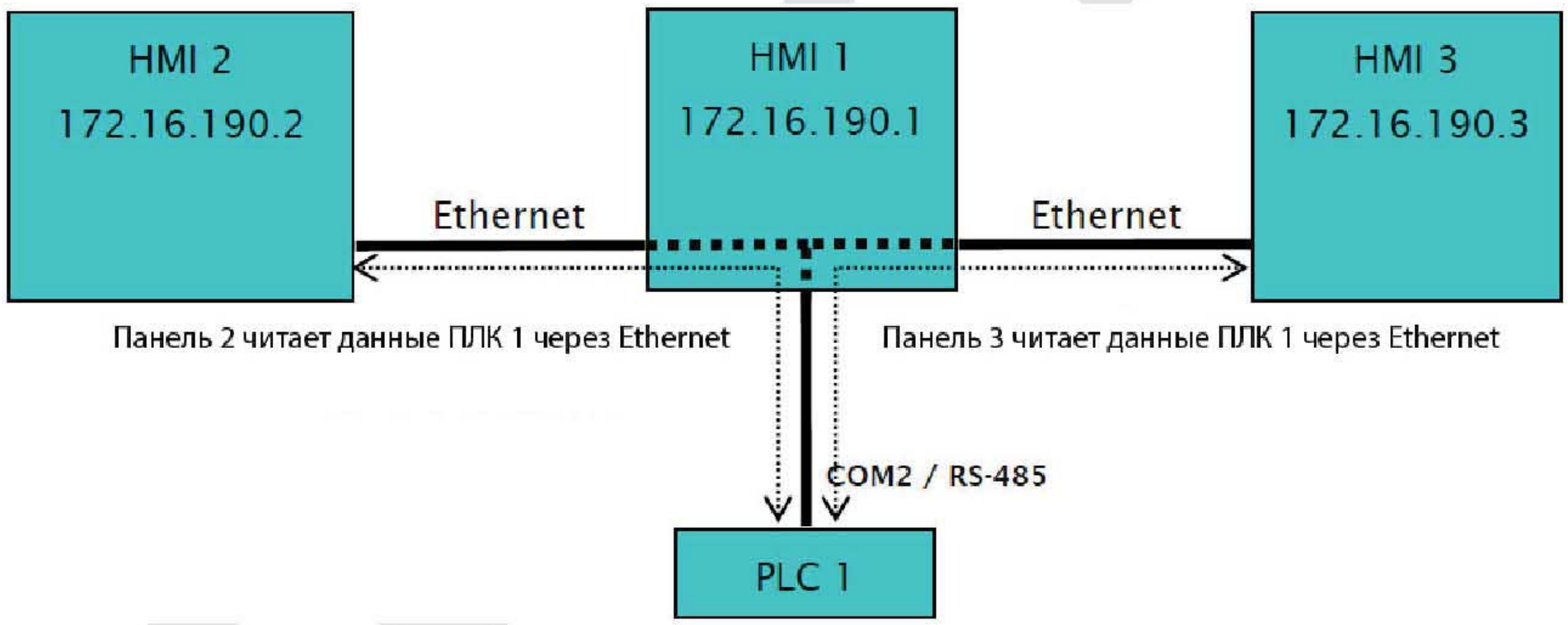
Серия контроллера	Поддержка функции проверки пароля	Поддержка функции проверки верификационного кода
ES	X	X
ES2	V1.0 и последующие	V1.0 и последующие
EX	X	X
EC	X	X
SS	X	X
SA	V 1.7 и последующие	V 1.7 и последующие
SX	V 1.7 и последующие	V 1.7 и последующие
SC	V 1.5 и последующие	V 1.5 и последующие
SV	V 1.2 и последующие	V1.3 и последующие
EH	X	X
EH2	V 1.1 и последующие	V1.3 и последующие
EH2-L	V1.0 и последующие	V1.0 и последующие

5. Режим **Transfer Mode** применим для загрузки/выгрузки **только** программ контроллеров серии DVP (*.dvp) и не может использоваться для загрузки/выгрузки подпрограмм, многоступенчатых диаграмм, последовательных функциональных схем, данных об имени устройства, энергонезависимых данных, структуры меток и символов и др. данных.

Приложение C.
Многоканальное соединение

В разделе описаны методы объединения нескольких панелей оператора с одним или несколькими контроллерами.

Способы объединения показаны на рисунке, где используется связь и по COM2 и по Ethernet .



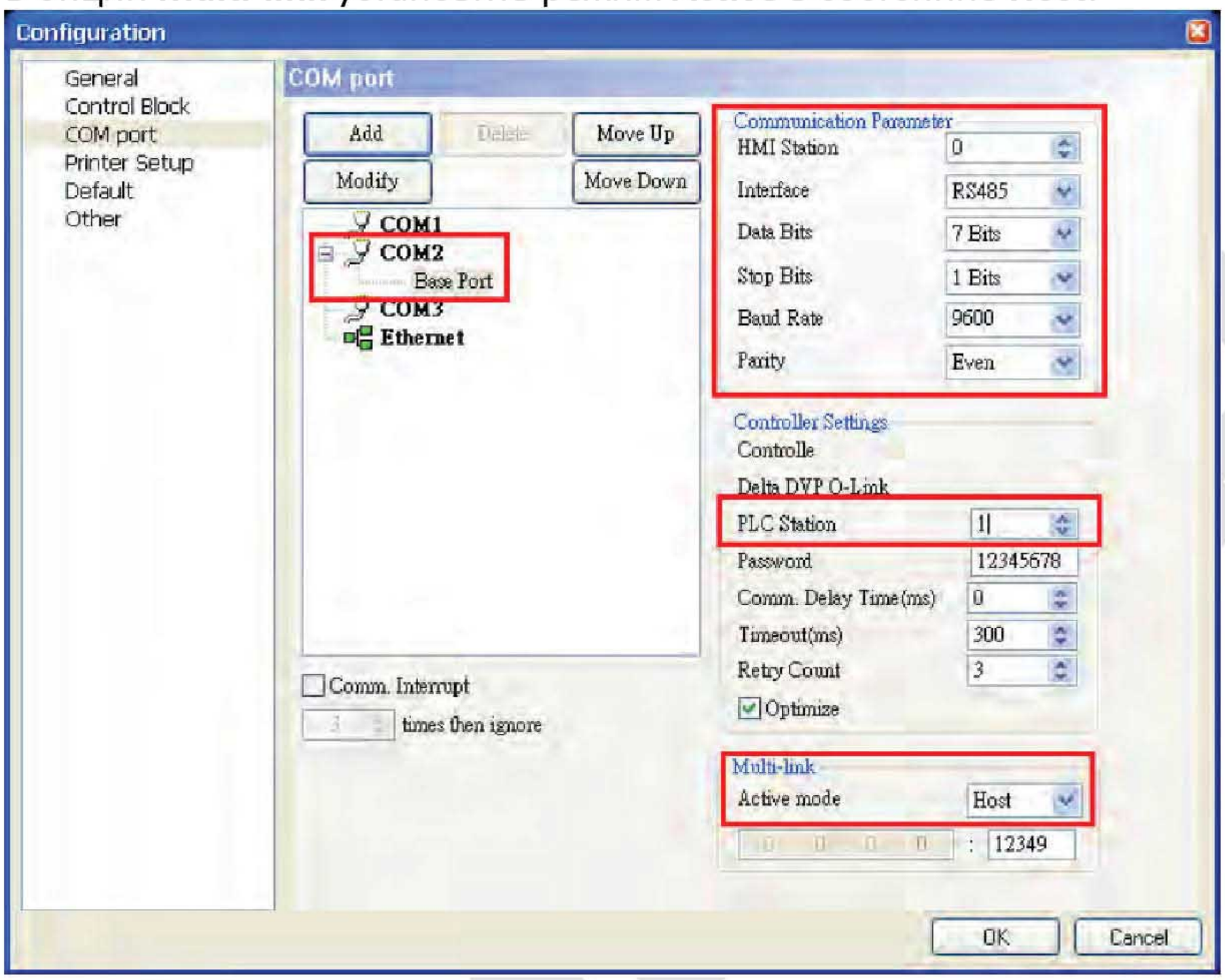
Все панели оператора HMI 1, HMI 2 и HMI 3 соединены по Ethernet, и только паненль HMI 1 соединена через COM2 с контроллером через RS-485. HMI 2 и HMI 3 не соединены ни с одним внешним контроллером. Однако, через 1:N соединение панели HMI 2 и HMI 3 могут иметь доступ к регистрам контроллера.

Способы настройки сетевых устройств:

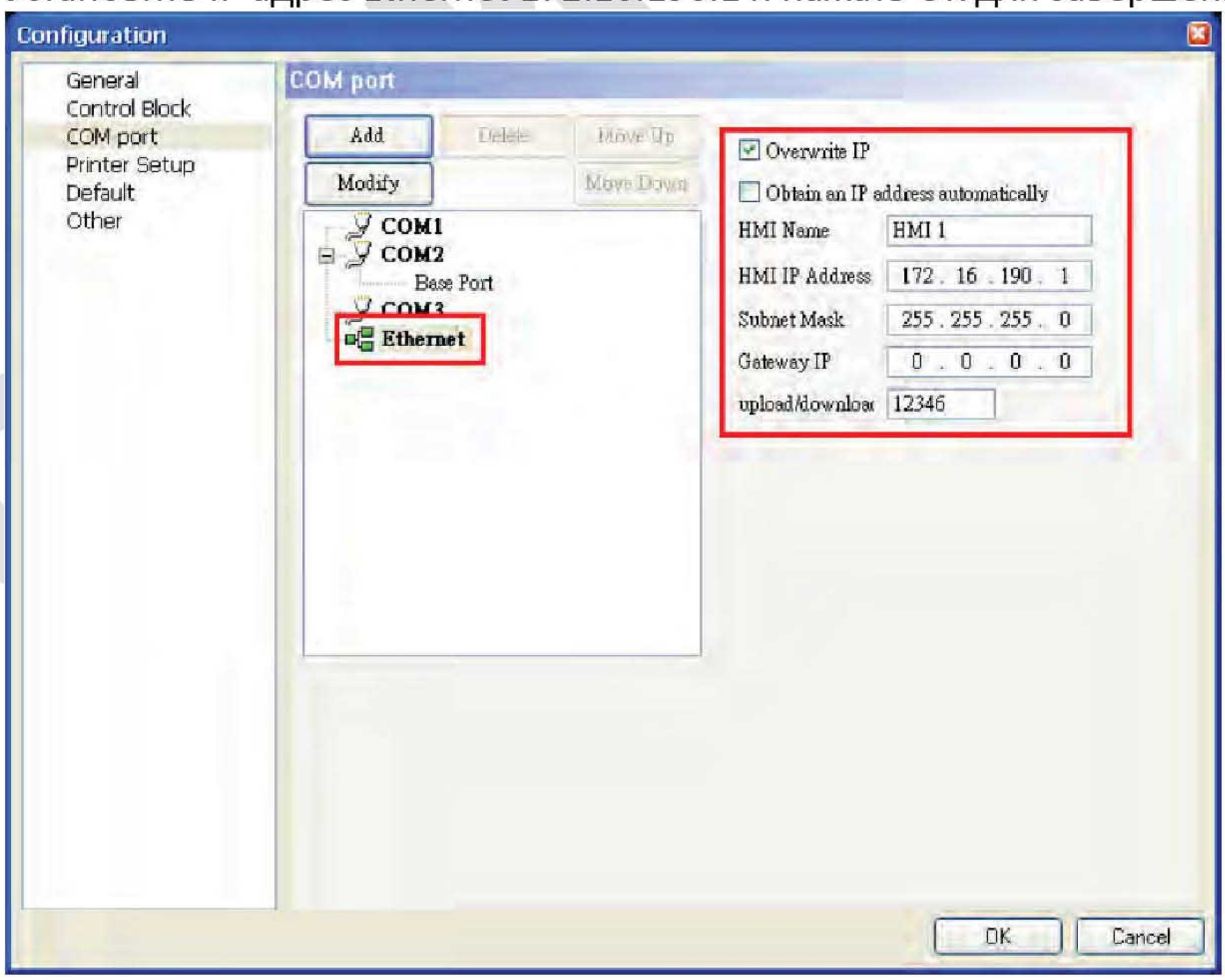
Способы настройки HMI 1

1. С помощью мыши войти в настойки порта **Options > Configuration > COM Port**.
2. Кликнуть мышью кнопку **Add** для добавления контроллера подключенного к порту **COM2**.
3. Задать сетевой номер панели = **1**, выбрать **RS-485** интерфейс и задать настройки коммуникационного протокола [**9600, 7, Even, 1**].

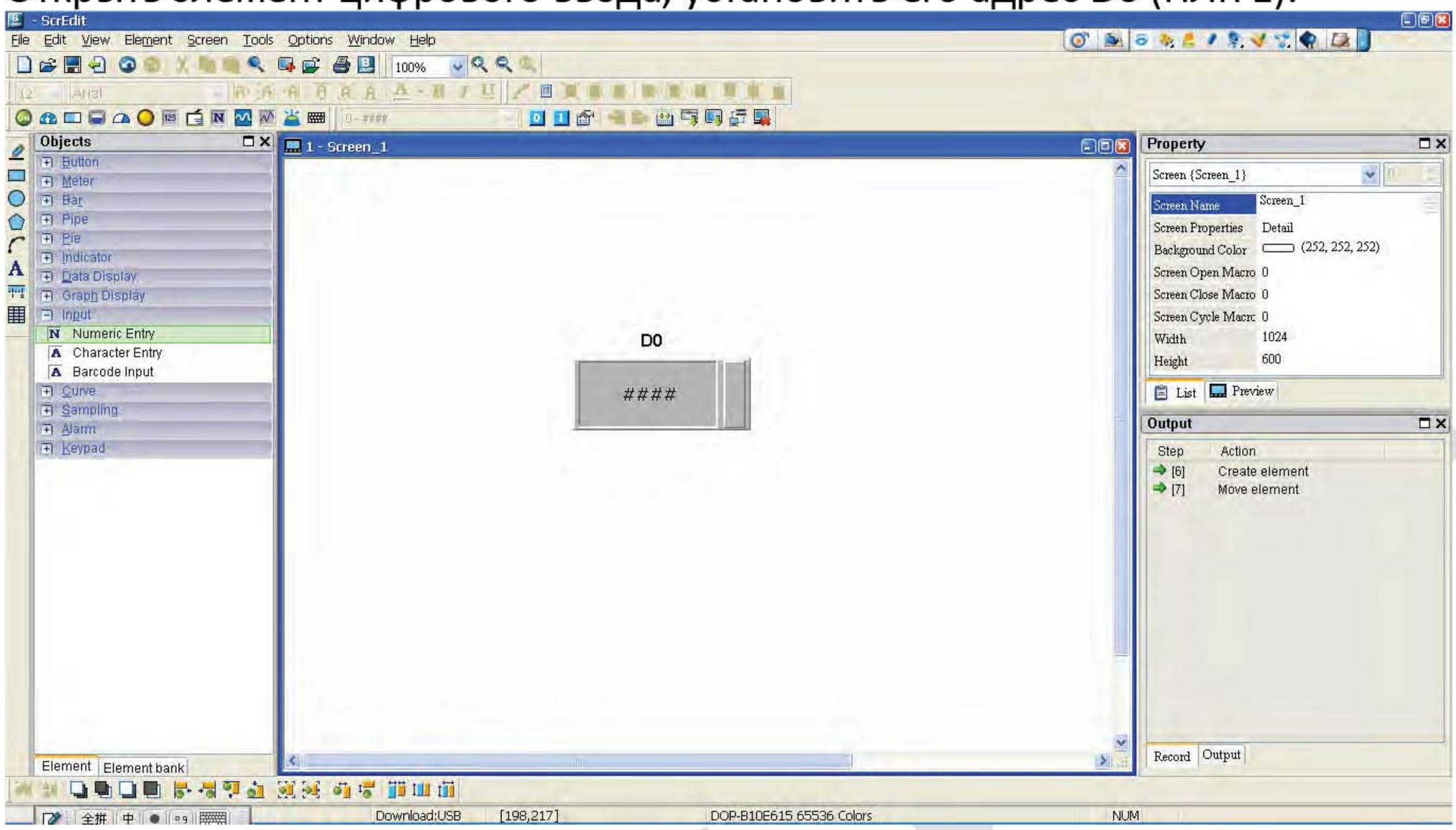
4. В опции **Multi-link** установить режим **Active** в состояние **Host**.



5. Установить IP адрес Ethernet 172.16.190.1 и нажать OK для завершения.



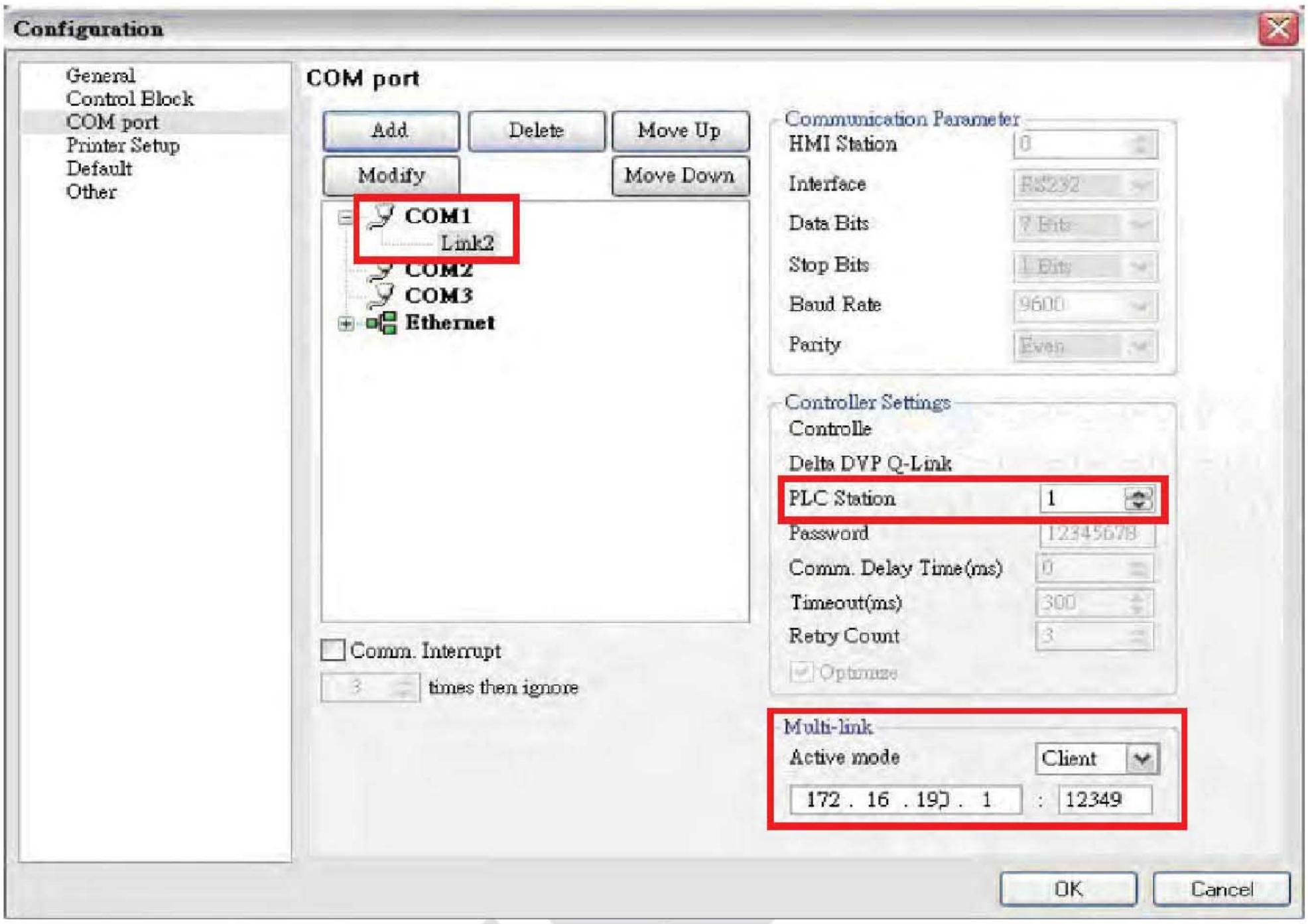
6. Открыть элемент цифрового ввода, установить его адрес **D0** (ПЛК 1).



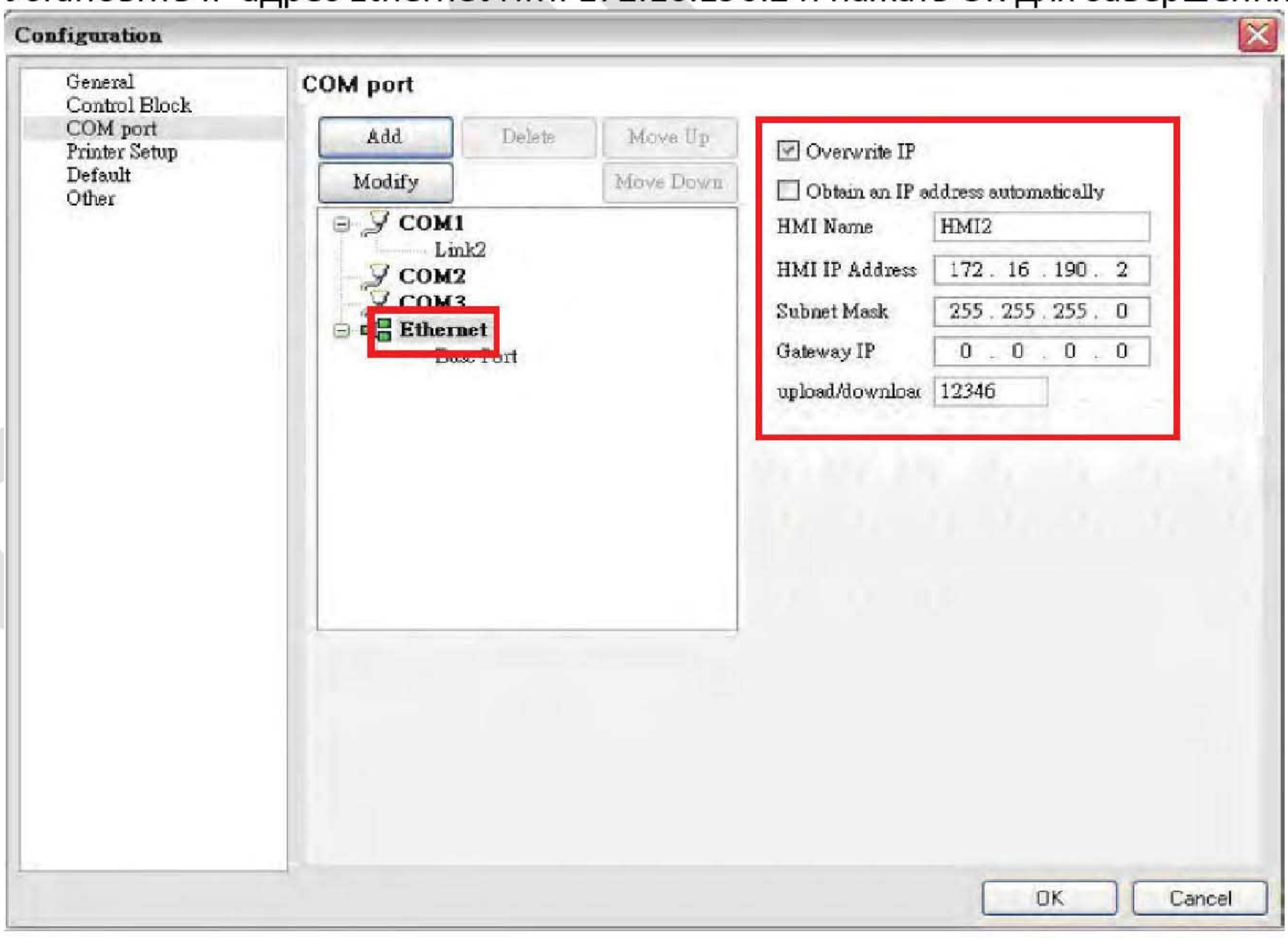
7. После завершения провести компиляцию и загрузить программу в панель HMI 1.

Способы настройки HMI 2

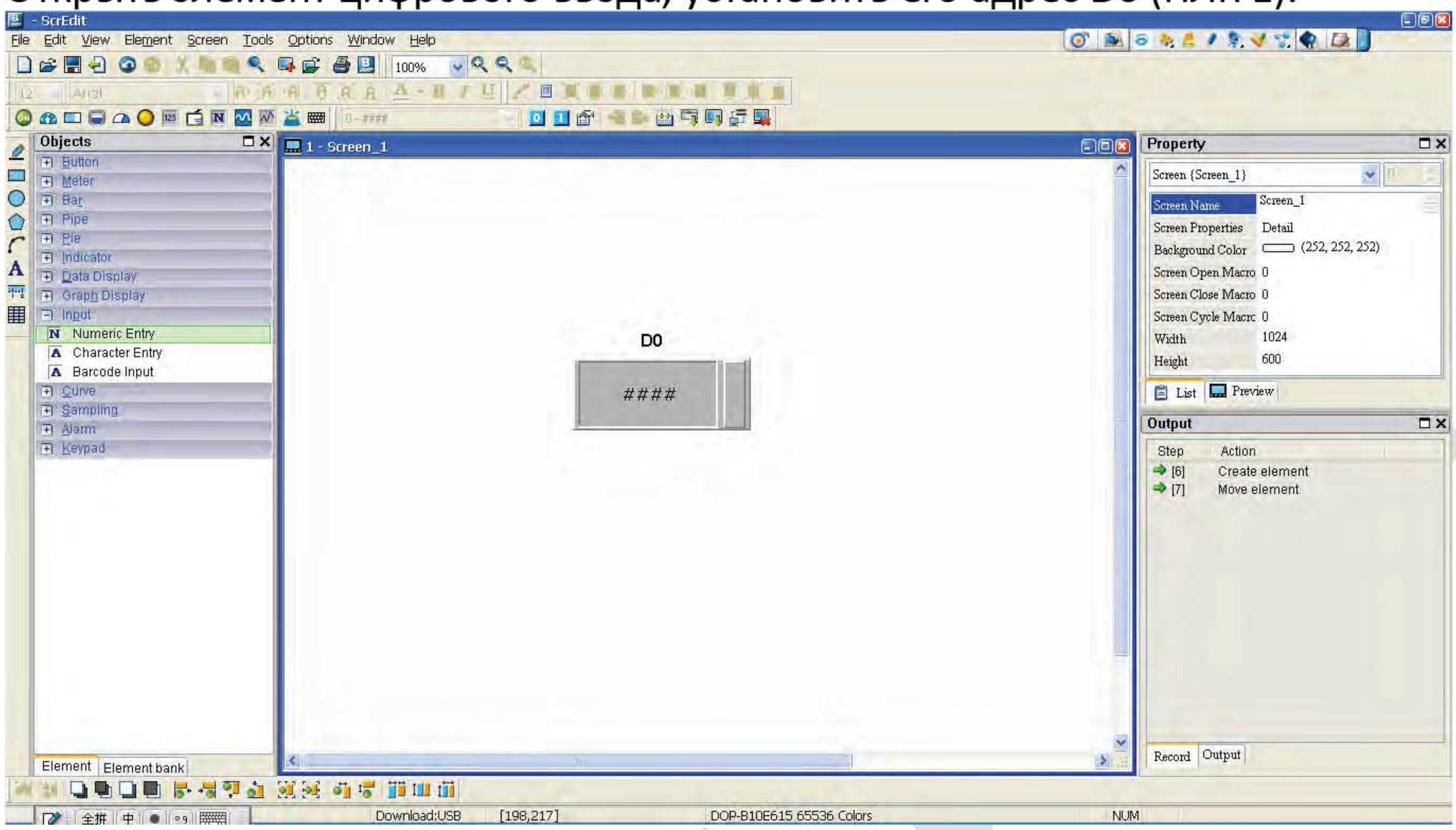
1. С помощью мыши войти в настойки порта **Options > Configuration > COM Port**.
2. Кликнуть мышью кнопку **Add** для добавления контроллера подключенного к порту **COM1**
3. Установить адрес контроллера = **1**.
4. В опции **Multi-link** установить режим **Active** в состояние **Client** и установить IP 172.16.190.1 (IP адрес HMI 1).



5. Установить IP адрес Ethernet HMI 172.16.190.2 и нажать OK для завершения.



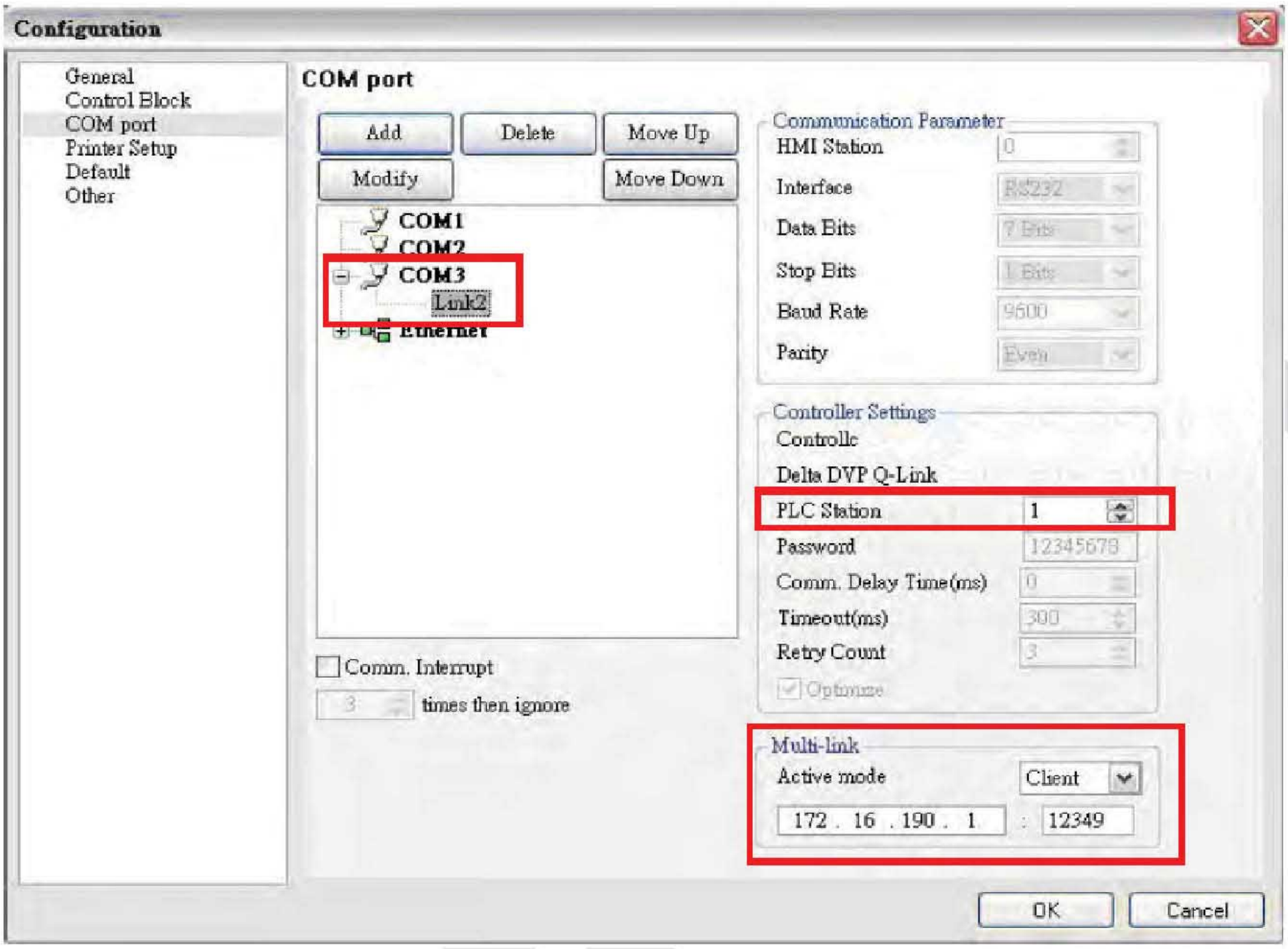
6. Открыть элемент цифрового ввода, установить его адрес **D0** (ПЛК 1).



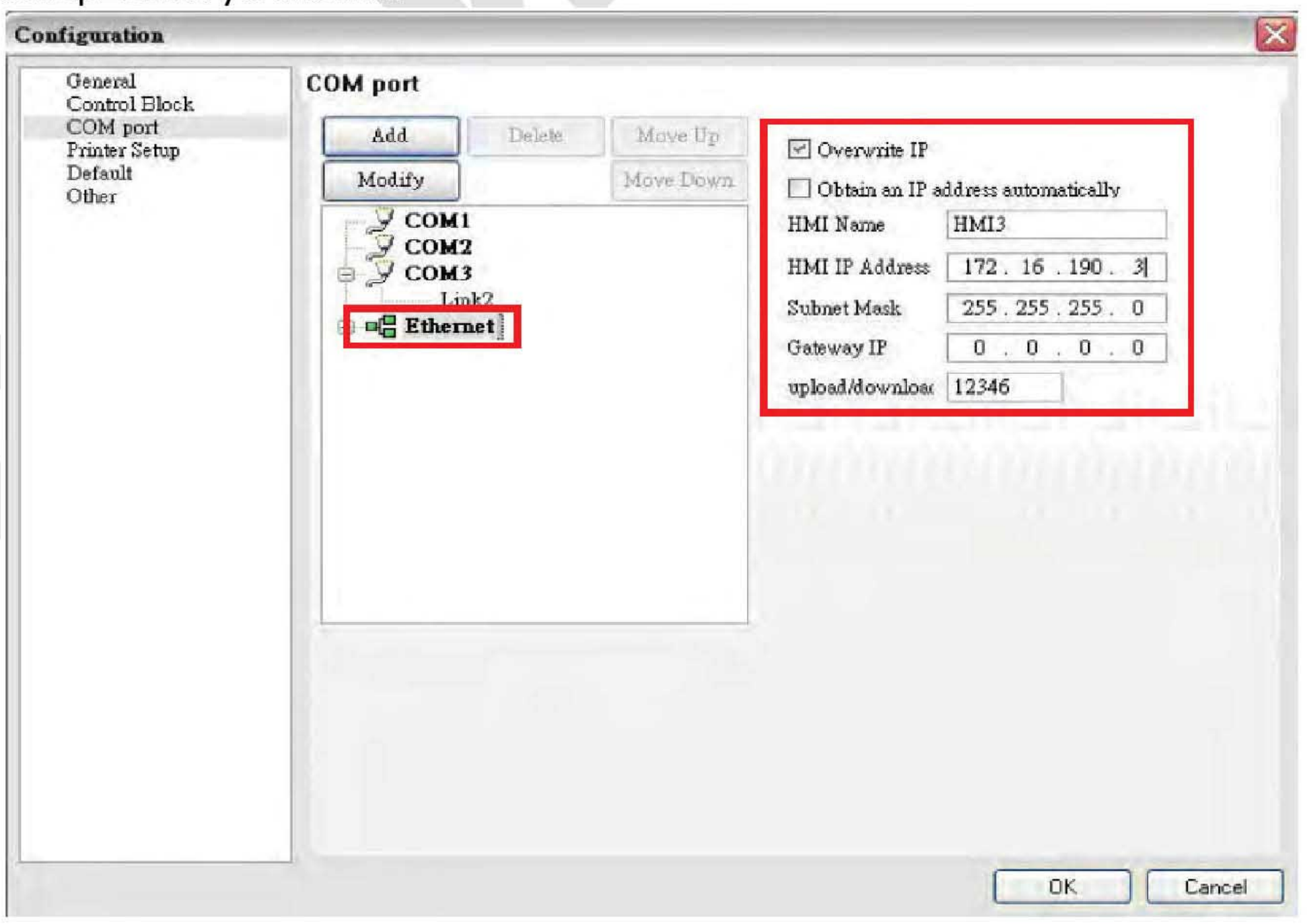
7. После завершения провести компиляцию и загрузить программу в панель HMI 2

Способы настройки HMI 3

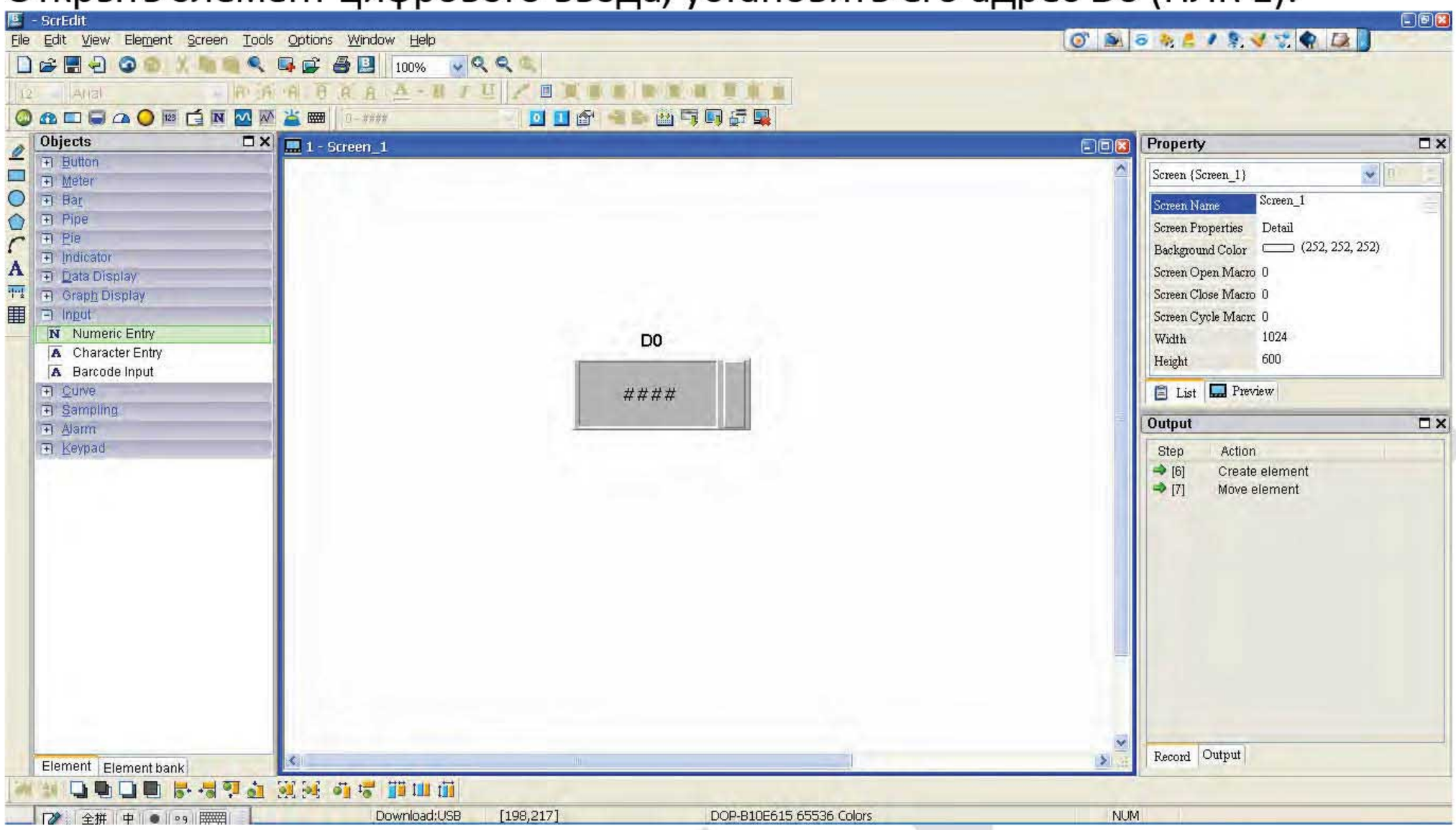
1. С помощью мыши войти в настойки порта **Options > Configuration > COM Port**.
2. Кликнуть мышью кнопку **Add** для добавления контроллера подключенного к порту **COM3**
3. Установить адрес контроллера = **1**.
4. В опции **Multi-link** установить режим **Active** в состояние **Client** и установить IP 172.16.190.1 (IP адрес HMI 1).



5. Установить HMI IP адрес Ethernet to 172.16.190.3 и нажать OK для завершения установки.



6. Открыть элемент цифрового ввода, установить его адрес **D0** (ПЛК 1).



7. После завершения провести компиляцию и загрузить программу в панель HMI 3.

Далее, подсоединить контроллер ПЛК 1 к панели оператора HMI 1, и соединить панели HMI 1 с HMI 2 и HMI 3.

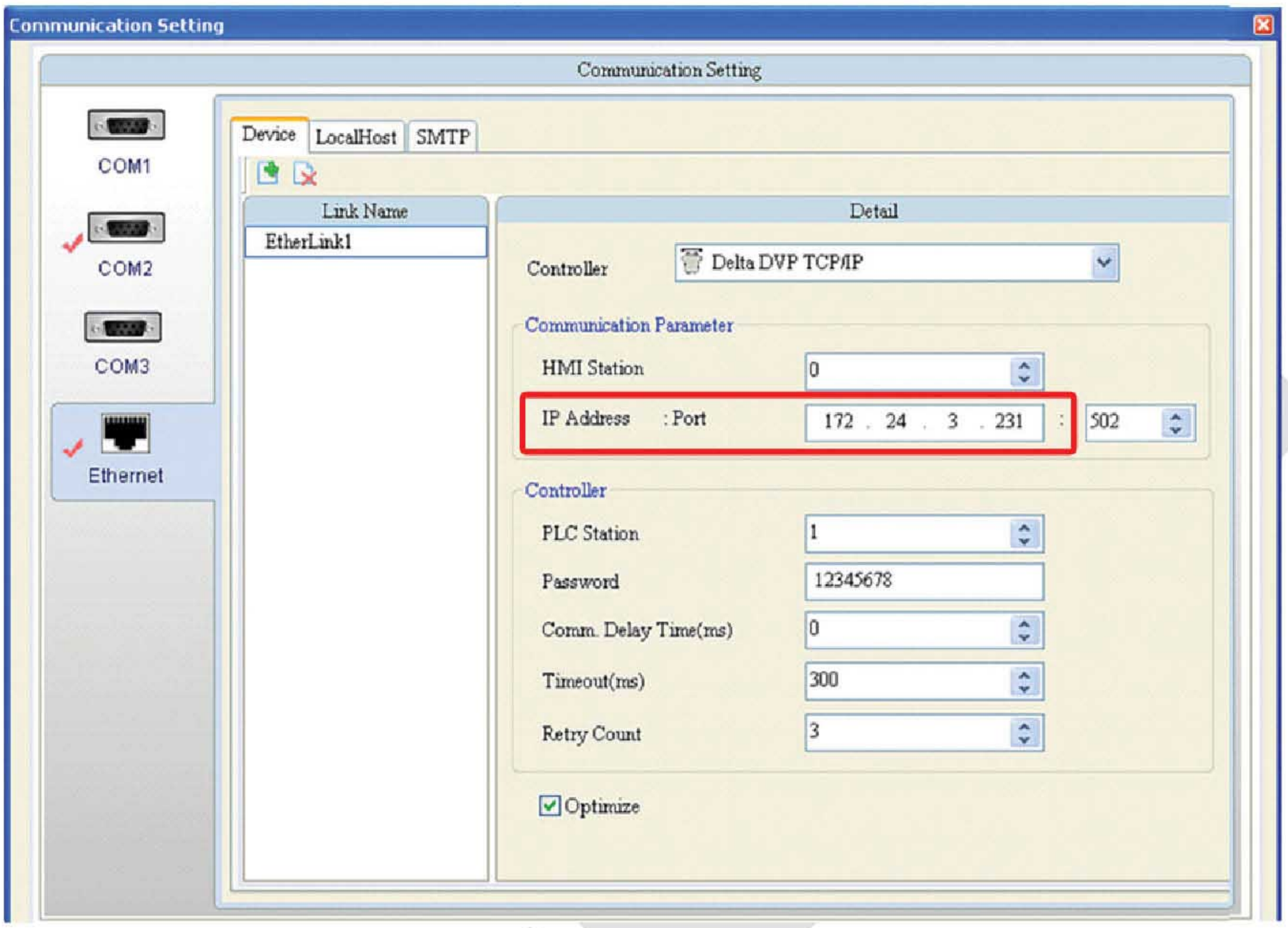
После подачи питания на все эти устройства образуется многоканальная сеть.

Приложение D.
Посылка сообщений об авариях на электронную почту

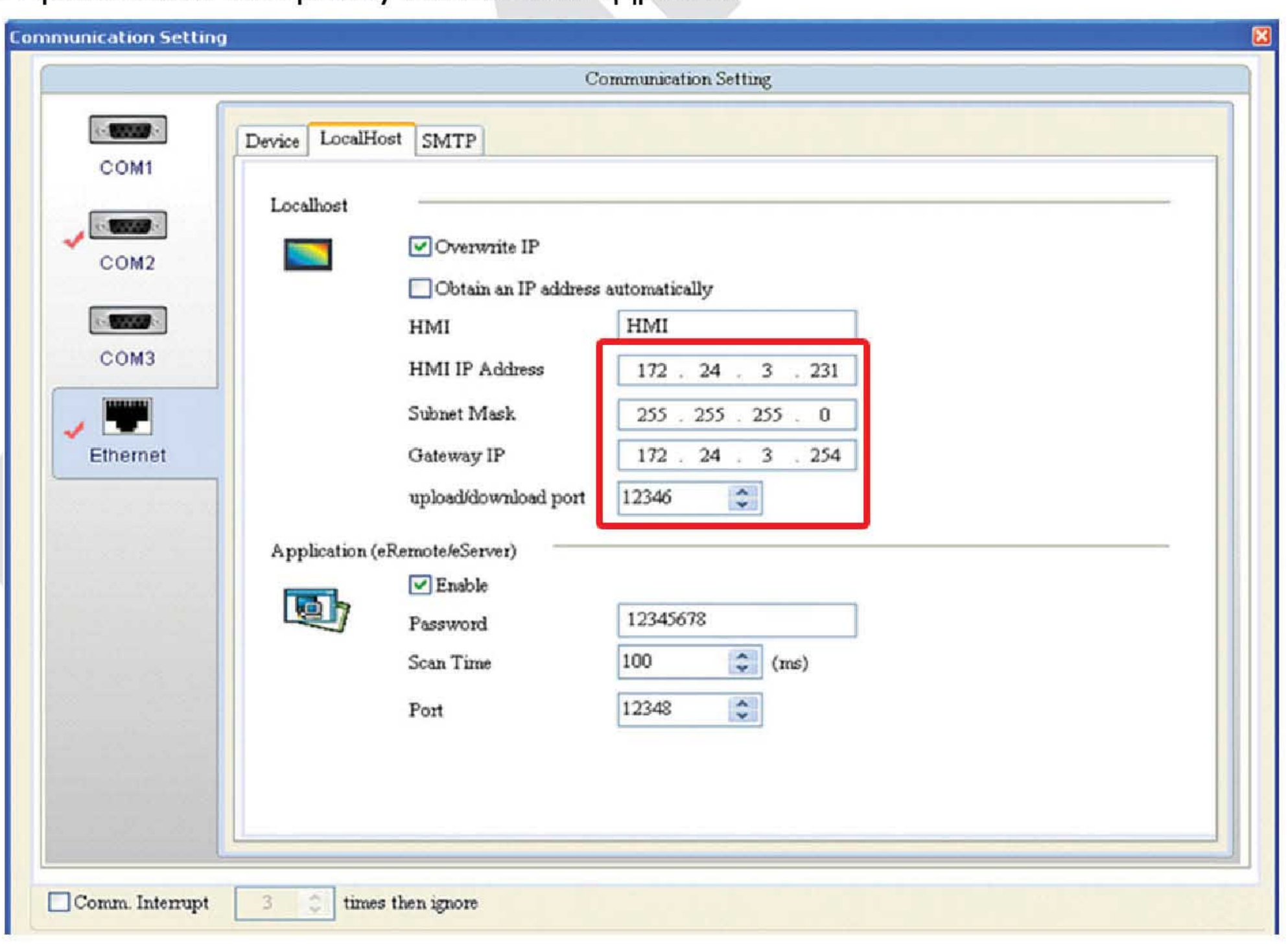
Все панели оператора серии DOP-B **со встроенным Ethernet** имеют возможность при записи аварийного сообщения в архив аварий посылать соответствующее сообщение по электронной почте.

Описание процедуры:

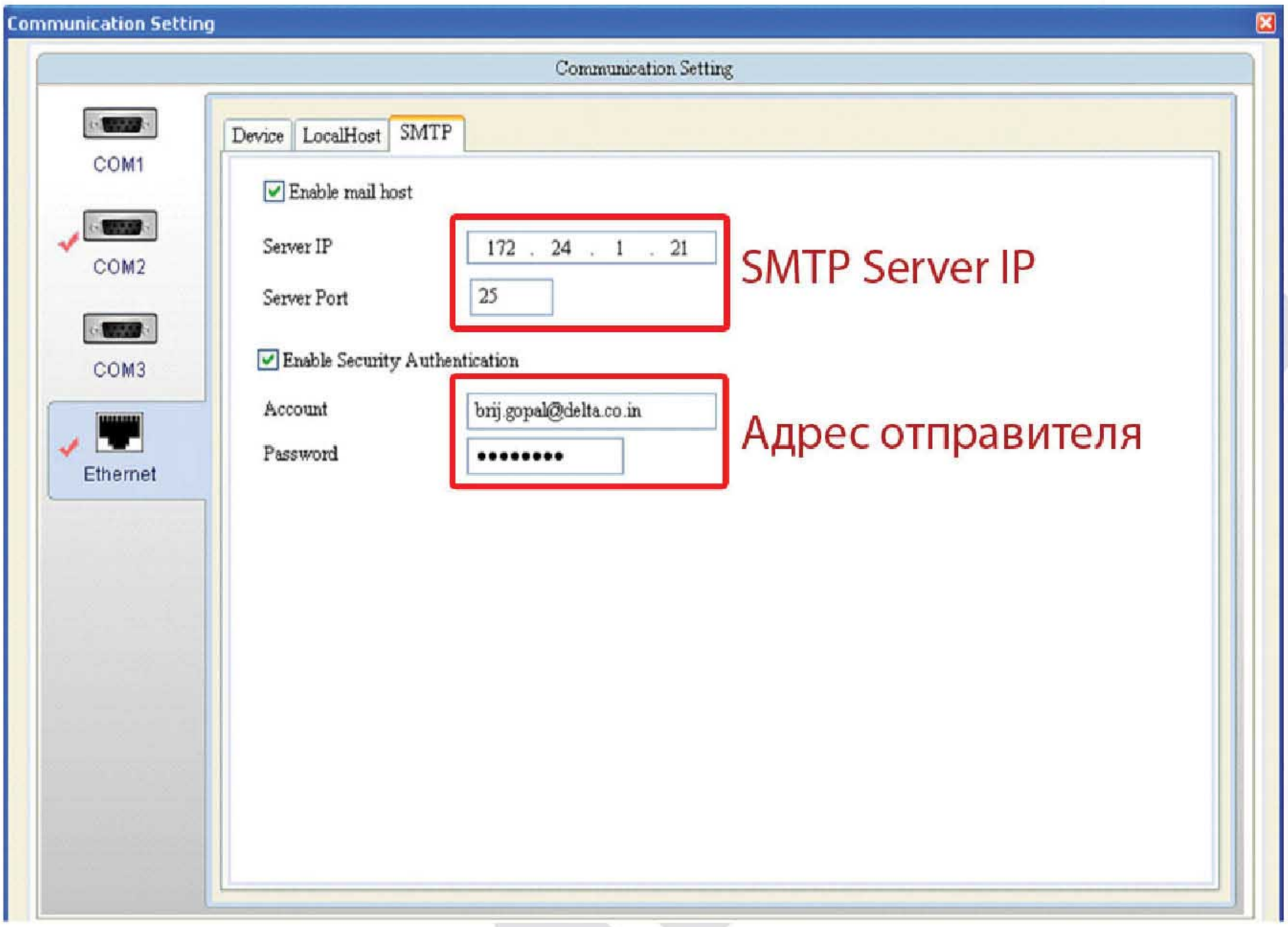
1. Присвоить IP адрес для панели оператора.



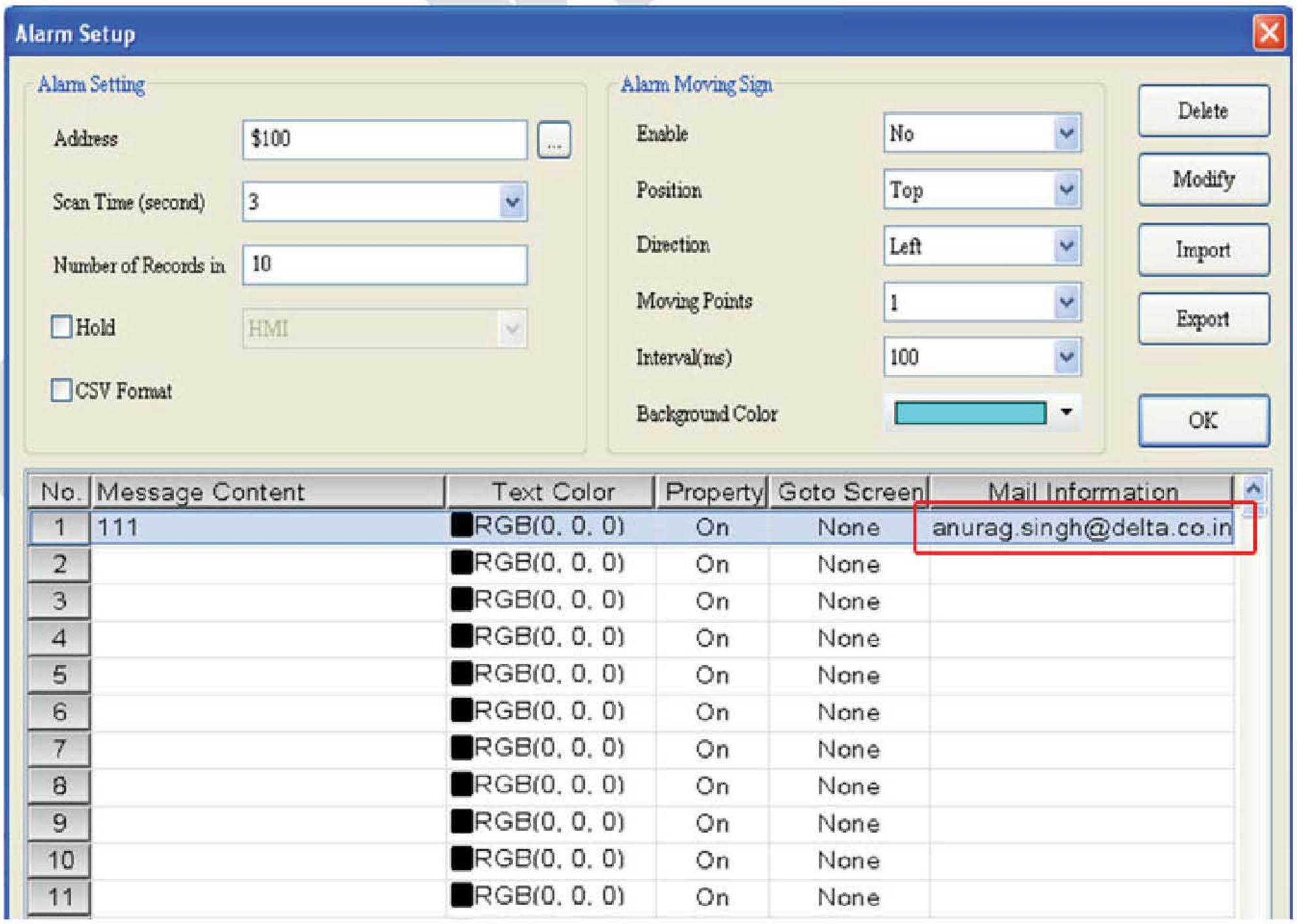
2. Произвести настройку локальных адресов.



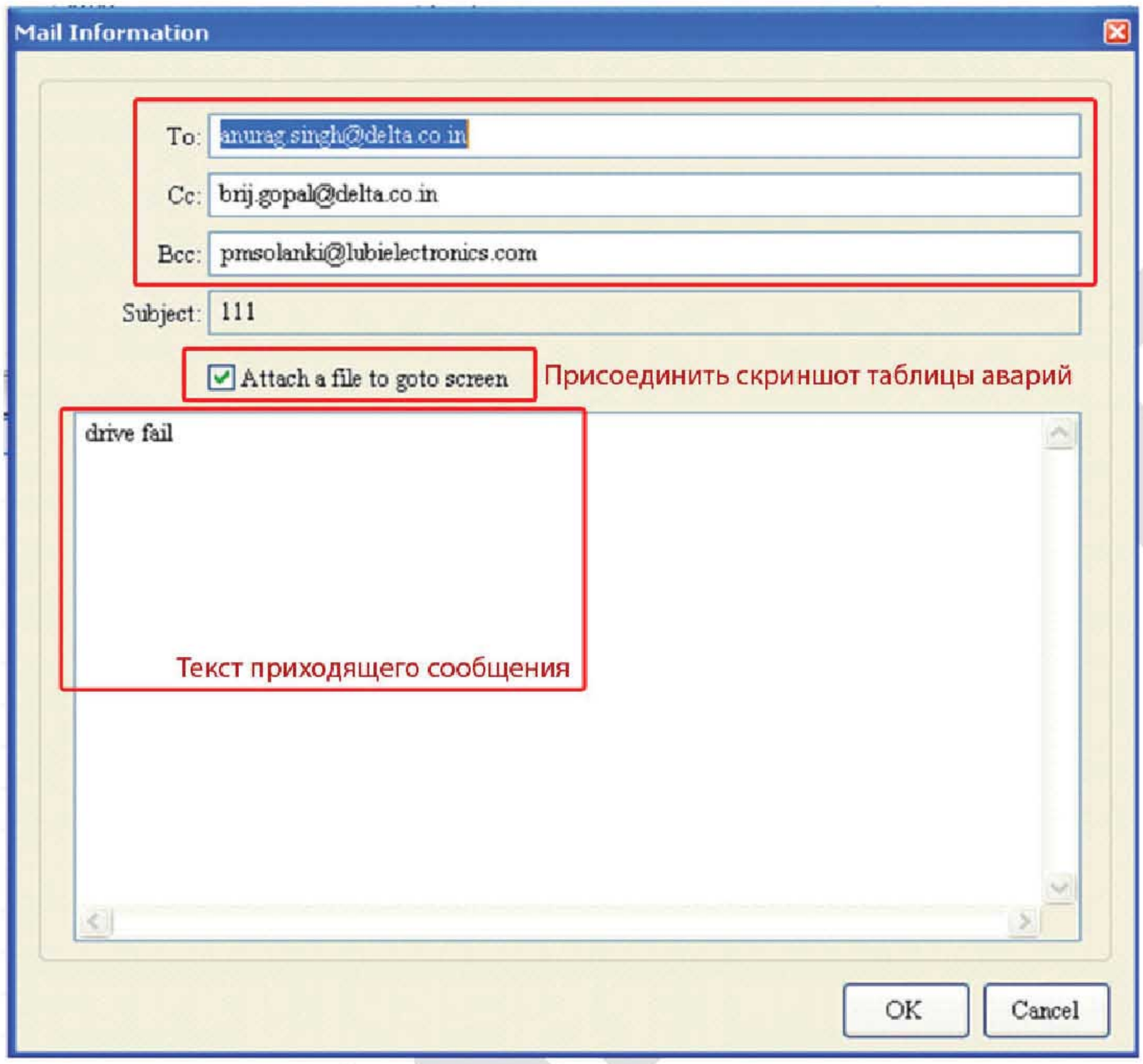
3. Провести настройки SMTP сервера.



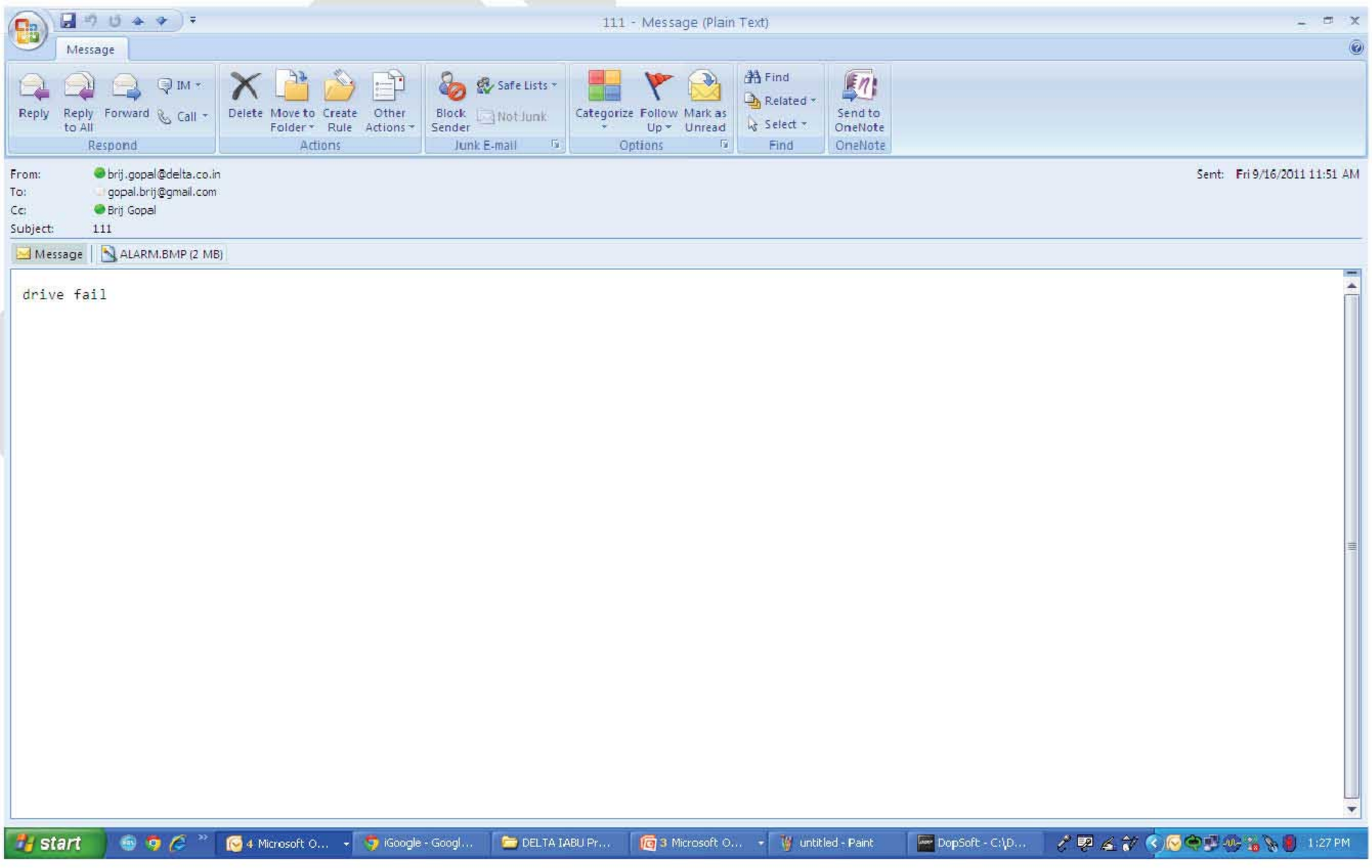
4. Присвоить каждой аварии почтовый адрес для посылки сообщения



5. Настроить отправляемое сообщение.



Полученное сообщение:



Приложение E.
Реализация функции Direct Link

Функция **Direct Link** в программном обеспечении Delta **WPL Soft** может помочь пользователям загружать и выгружать программы в контроллер Delta и контролировать состояние контроллера с помощью панели оператора Delta без непосредственного подключения её к контроллеру.

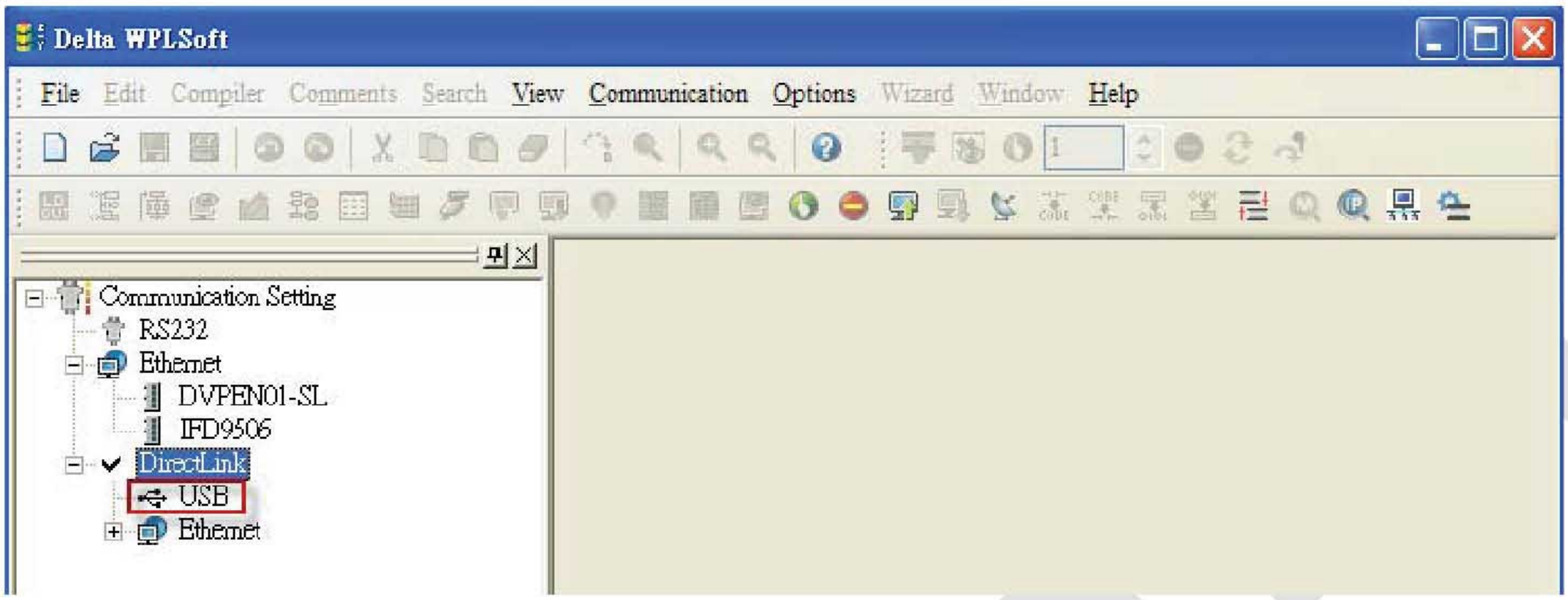
Типы Direct Link коммуникаций:

1. USB Direct Link: Использование USB кабеля для связи с контроллером через панель оператора.
2. Ethernet Direct Link: Использование Ethernet для связи с контроллером через панель оператора.

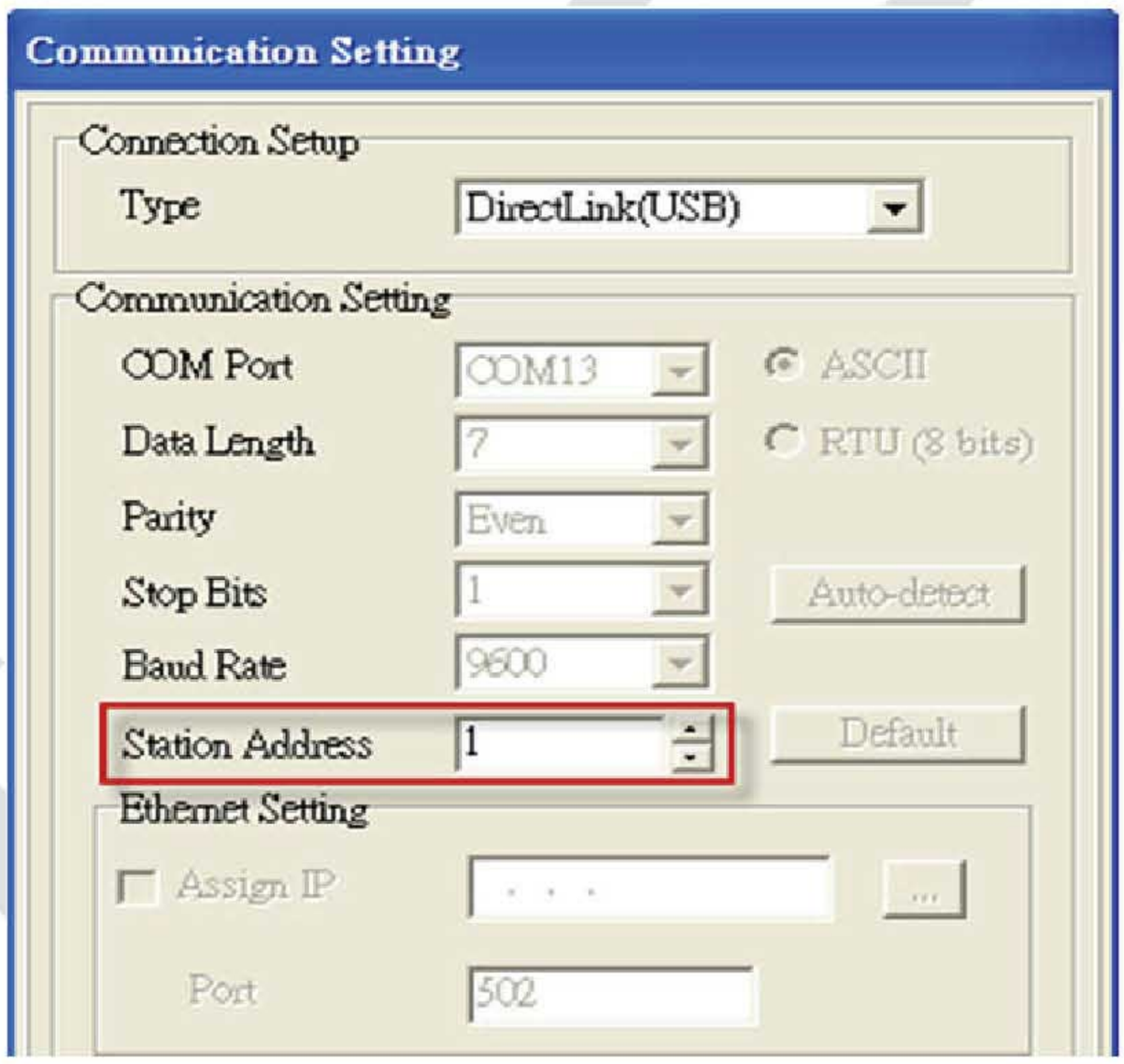
Внимание: Только контроллеры серии Delta DVP поддерживают функцию Direct link.

Панель оператора связывается с контроллером по RS-232/RS-485 и компьютер связывается с панелью оператора с помощью кабеля USB.

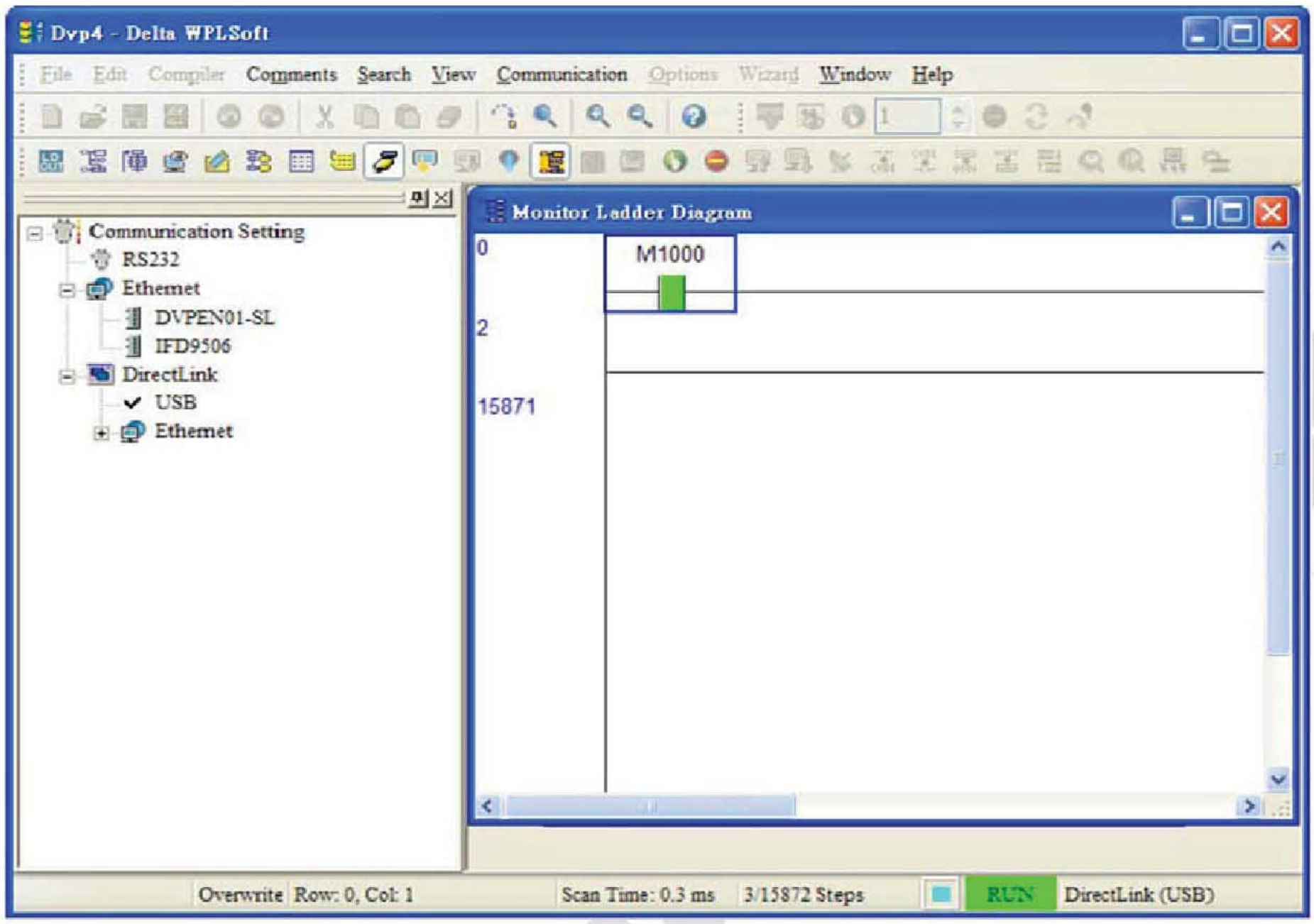
Загрузить программу Delta **WPL Soft** в компьютер и двойным кликом мыши по значку **USB** войти в диалоговое окно **Communication Setting**.



В настройке **Station Address** установить адрес контроллера, который необходимо подключить.



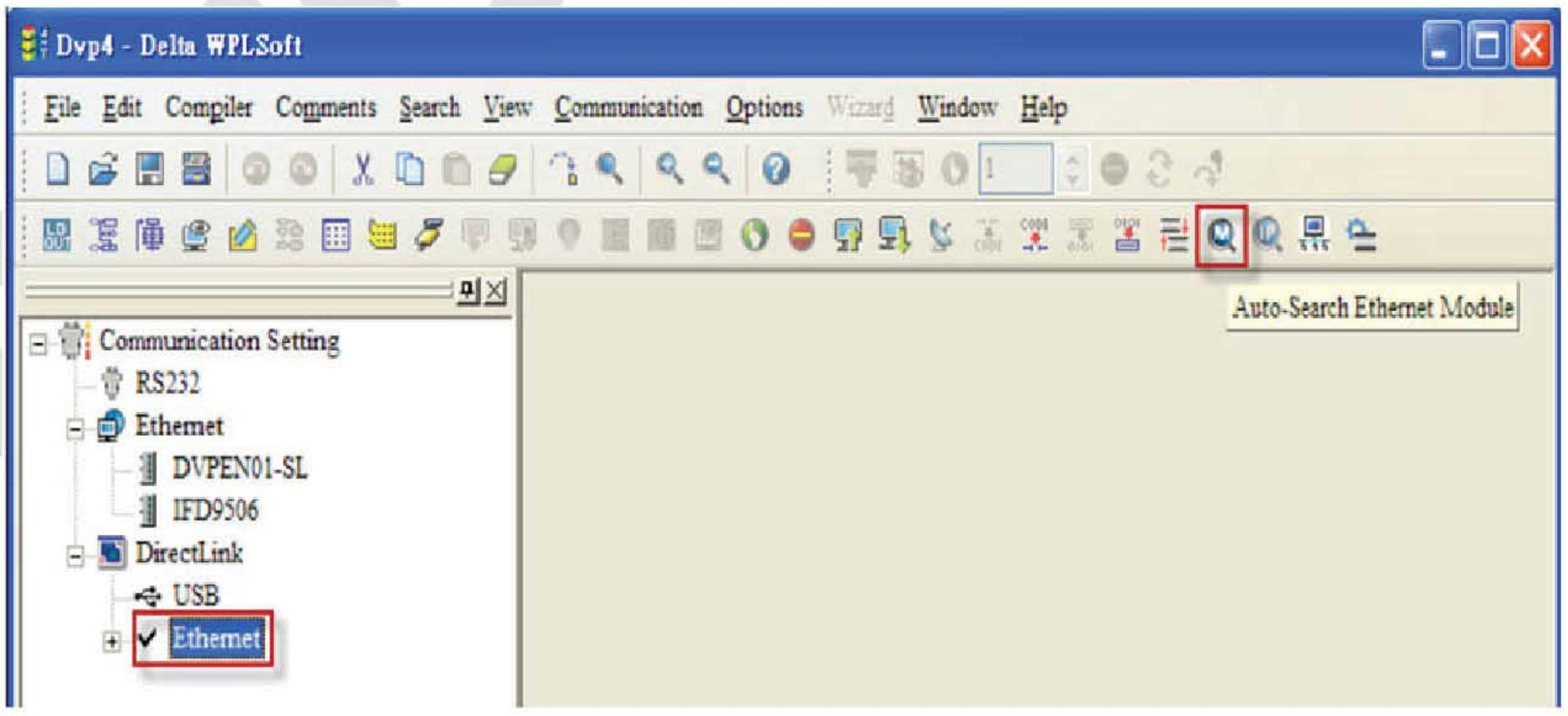
В программе **WPLSoft** мы можем прочитать программу контроллера или создать программу и записать её в контроллер.



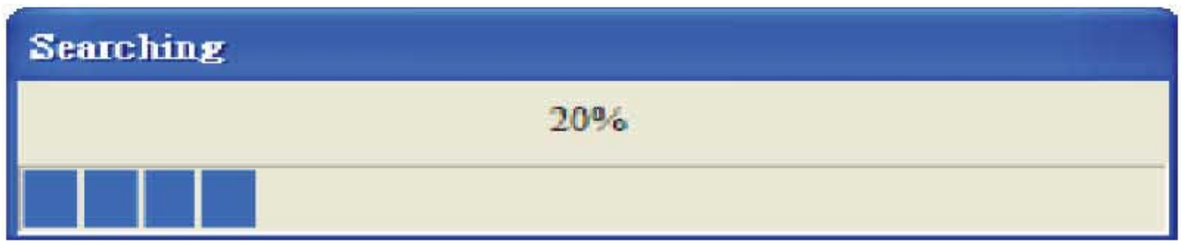
Панель оператора связывается с контроллером по RS-232/RS-485, панель оператора с компьютером находятся в одной локальной сети.

Загрузить программу Delta **WPL Soft** в компьютер и кликнуть мышью по значку 

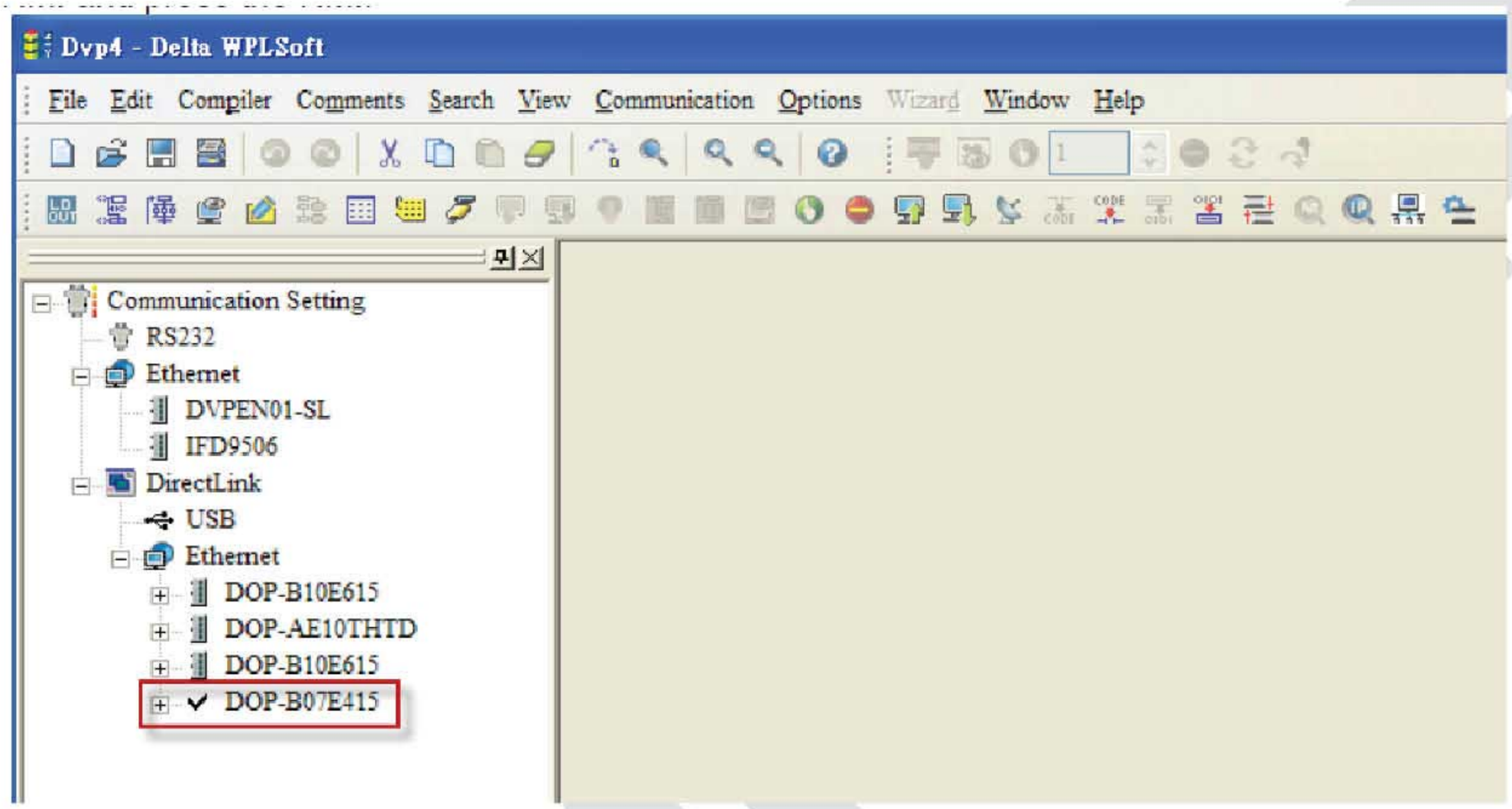
, далее кликом мыши по значку  запустить функцию **Auto-Search Ethernet Module** (автоматический поиск сетевых устройств).



Программа **WPLSoft** начнёт обзор сетевых устройств.



После обзора, можем увидеть перечень панелей оператора, и выбрать необходимую модель.



Все настройки должны сохраниться такими же, как и в режиме **Direct link Ethernet**.

Далее необходимо получить статический IP адрес от провайдера (ISP) и преобразовать в него IP адрес локальной сети (для этого требуется роутер с функциями NAT (трансляции сетевых адресов)

Пример:

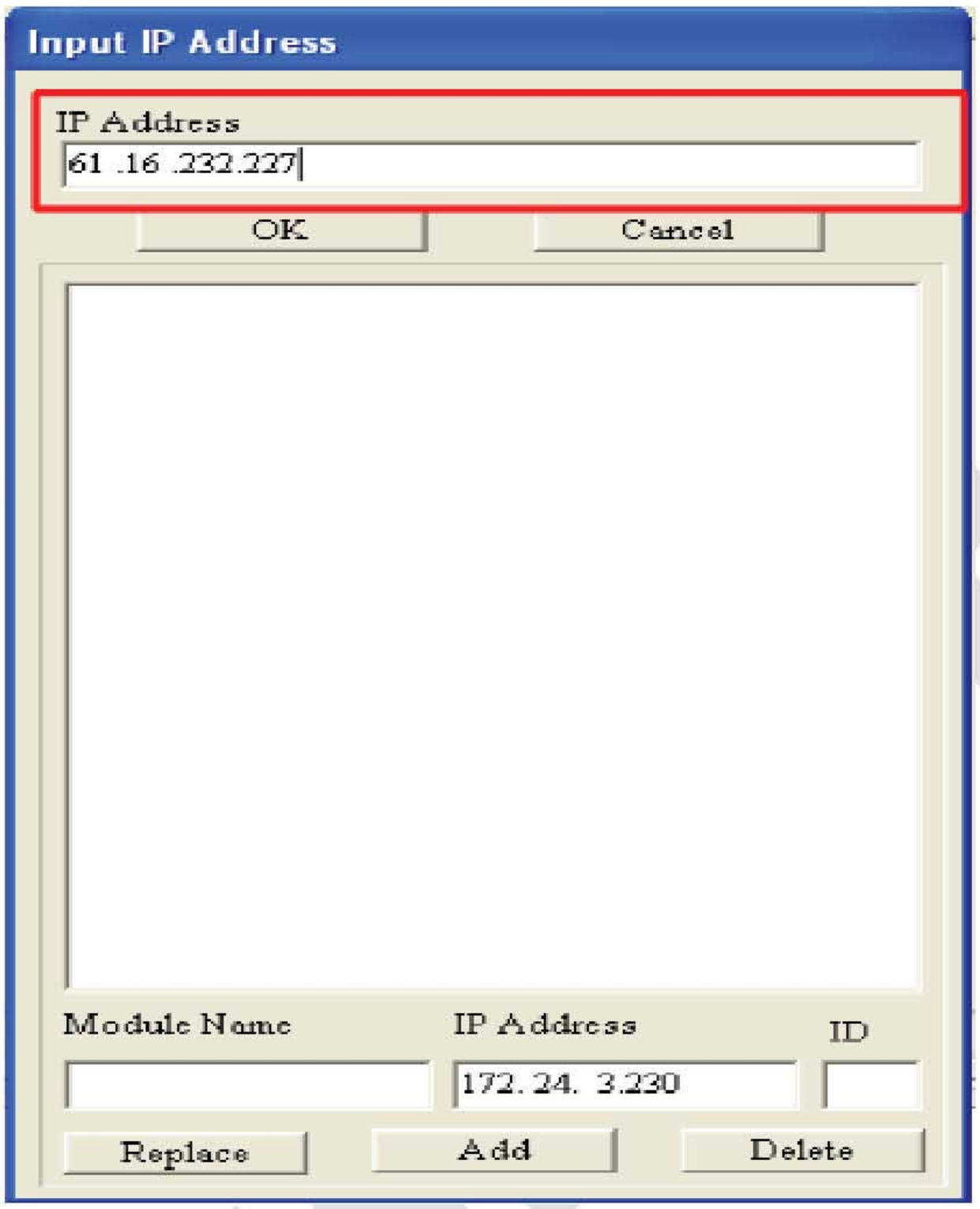
Локальный IP: 172.24.3.231

Статический IP: 61.16.232.227

Мы должны провести трансляцию локального IP в статический IP.

Удалённый компьютер должен быть подключен к сети Internet и фаервол должен быть выключен.

Запустить **WPLSoft** в компьютере. Для ввода IP адреса войти в окно ввода IP адресов для чего кликнуть мышью значок  . Сначала ввести статический IP и нажать **OK**.



Теперь мы можем загружать и выгружать программы в контроллер Delta и из него и контролировать состояние контроллера с помощью панели оператора.