

Программатор ТРИТОН

Содержание

<u>Введение</u>	4
<u>Программаторы ТРИТОН</u>	5
<u>Интерфейс и питание программатора V5.8TU</u>	8
<u>Интерфейс и питание программатора V5.7TU</u>	9
<u>Установка драйверов и программы</u>	11
<u>Ручная установка USB драйвера</u>	18
<u>Настройка USB драйвера</u>	22
<u>Устранение проблем с подключением</u>	24
<u>Начало работы</u>	27
<u>Программа TriSoft.exe</u>	28
<u>Меню программы</u>	29
<u>Меню Файл</u>	30
<u>Меню Правка (Буфер)</u>	31
<u>Меню Проект</u>	34
<u>Меню Программатор</u>	36
<u>Выбор COM порта</u>	36
<u>Обновление программы</u>	36
<u>Параметры и Тесты</u>	38
<u>Интерфейс</u>	38
<u>Программатор</u>	39
<u>Статистика</u>	40
<u>Калибровка</u>	40
<u>Тесты</u>	41
<u>Меню Микросхема</u>	43
<u>Выбор</u>	43
<u>Серийный номер</u>	43
<u>Управление блоками</u>	45
<u>Коррекция ошибок</u>	47
<u>Команда ERASE</u>	48
<u>Команда PROGRAMM</u>	49
<u>Команда TEST BLANK</u>	51
<u>Команда CHECKSUM</u>	51
<u>Команда READ CHIP</u>	51
<u>Команда VERIFY CHIP</u>	53
<u>Команды для работы с EEPROM</u>	54
<u>Меню Вид</u>	55
<u>Меню Окно</u>	56
<u>Меню Справка</u>	57
<u>Окно проекта</u>	58
<u>Выбор микросхемы</u>	59
<u>Выбор файла</u>	62
<u>Сохранение файла</u>	64
<u>Закладка Сообщения</u>	66
<u>Закладка Буфер</u>	67
<u>Закладка Конфигурация</u>	68
<u>Закладка Параметры</u>	69
<u>Закладка Программатор</u>	72
<u>Настройка программы</u>	75
<u>Блок параметров микросхемы</u>	76
<u>Формат файла микросхем</u>	78
<u>Формат файла конфигурации</u>	79
<u>Формат файла калькулятора</u>	84

Виртуальный процессор	87
Система команд	91
Описание ошибок и сообщений программы	98
Особенности работы с микросхемами	103
Внутрисхемное программирование	104
Микросхемы памяти EPROM 27x	107
Микросхемы памяти EEPROM 28x	108
Микросхемы памяти FLASH	109
Микросхемы памяти FWH и LPC-Flash	111
Микросхемы статического ОЗУ	112
Микросхемы NAND-Flash	113
Микросхемы Serial DATA-Flash	117
Микросхемы SEEPROM 17x	118
Микросхемы SEEPROM 24x	119
Микросхемы SEEPROM 25x	120
Микросхемы SEEPROM 93x	122
Микроконтроллеры AVR	123
Микроконтроллеры Holtek	124
Микроконтроллеры MSC-51	125
Микроконтроллеры Philips P89LPC9xx	127
Микроконтроллеры Winbond	128
Микросхемы Pic10xxx, Pic12xxx, Pic16C5x	129
Микросхемы Pic12F6xx, Pic16Cxxx, Pic16Fxxx	131
Микросхемы Pic17	133
Микросхемы Pic18C, Pic18F, Pic24, dsPIC	134
Автономный режим	135
Подготовка к работе	136
Назначение кнопок	137
Выбор микросхемы и файла	138
Работа с микросхемой	139
Меню программатора	140
Лицензионное соглашение	141
Гарантийные обязательства	143
Техническая поддержка	144

Это Руководство не только описывает работу с программой, но и содержит ответы практически на все вопросы, которые были у других пользователей, и могут возникнуть у Вас во время работы с программатором.

Все программаторы проходят предпродажную проверку, поставляются в исправном состоянии и полностью готовы к работе. Программное обеспечение работает со всеми версиями WINDOWS и протестировано на операционных системах Windows XP, VISTA, Windows 7, 8, 10 и 11 (32- и 64-разрядных).

Перед обращением в службу технической поддержки настоятельно рекомендуется изучить этот документ и выполнить все изложенные здесь рекомендации.

Программаторы ТРИТОН

Программаторы ТРИТОН - профессиональные высокоскоростные программаторы микросхем с USB интерфейсом и возможностью автономной работы. Предназначены для разработки, серийного производства, технического обслуживания и ремонта любой электронной аппаратуры. Программаторы поддерживают широкий спектр современных микросхем памяти и микроконтроллеров, обеспечивают качественную запись микросхем, высокую скорость работы, и имеют уникальный набор функций, которые отсутствуют у большинства конкурентов.

Самостоятельное расширение списка микросхем. Программное обеспечение позволяет менять и настраивать все параметры, используемые программатором для работы с микросхемой. Встроенный [графический редактор](#) drag-and-drop позволяет переназначать сигналы и использовать любые переходники и адаптеры. Подробная документация и развитая система подсказок в программе, позволяют настраивать конфигурации и добавлять аналоги поддерживаемых микросхем.

Создание собственных алгоритмов программирования. [Виртуальный процессор](#) в программаторе и компилятор скрипт файлов в оболочке, позволяют создать собственный алгоритм работы с любой микросхемой. [Система команд](#) виртуального процессора, разработанная с учетом особенностей программирования микросхем, позволяет писать простой и эффективный код, обеспечивающий высокую скорость работы с микросхемой.

Многооконный шестнадцатеричный редактор. Самый быстрый, компактный и функциональный [HEX редактор](#) среди аналогичных программ. Редактор поддерживает любые форматы файлов и данных, логические и арифметические операции, сравнение данных и файлов, управление данными с помощью [скриптов-калькуляторов](#).

Правильная работа с микросхемами NAND-Flash. Поддержка программой реальных схем обработки дефектных блоков и существующих алгоритмов коррекции ошибок, позволяют программаторам корректно работать с [микросхемами NAND-Flash](#), определять и поддерживать большинство алгоритмов, используемых в телевизорах, ресиверах, навигаторах, магнитолах и другом оборудовании.

Сравнение программаторов V5.8TU и V5.7TU



ТРИТОН+ V5.8TU USB



ТРИТОН+ V5.7TU USB

- 48-выводная заменяемая панелька с полным набором ключей питания и программирования.
 - Логическая матрица с коммутатором сигналов и аппаратной поддержкой основных протоколов.
 - Синхронная запись и чтение микросхемы без участия процессора программатора.
 - Выбор системной частоты (50, 33, 25, 16, 12, 8, 4, 2 МГц) или программное управление.
 - Установка времени выборки данных (30ns, 60ns, 90ns...1920ns).
 - Типы сигналов на панельке программатора:
 - земля, напряжение питания, программирования.
 - открытый сток (Pull-up), открытый исток (Pull-down).
 - активный ноль, активная единица.
 - генератор частоты.
 - Проверка контактов. Все активные сигналы с
- 40-выводная заменяемая панелька с неполным набором ключей питания и программирования.
 - Программная коммутация сигналов и организация протоколов.
 - Формирование сигналов чтения и записи процессором программатора.
 - Не поддерживается.
 - Не поддерживается.
 - Типы сигналов на панельке программатора:
 - земля, напряжение питания, программирования.
 - открытый сток (Pull-up).
 - Проверка контактов. Только шина данных (результат:

индикацией неподключенных выводов.

нормально или ошибка).

- Интерфейс с компьютером: USB High Speed (FT232H).
- скорость передачи данных до 9 МБайт/сек.
- максимальная скорость чтения 4-6 МБайт/сек.

- Интерфейс с компьютером: COM или USB Full Speed (FT245RL).
- скорость передачи данных до 750 КБайт/сек (USB).
- максимальная скорость чтения до 370 КБайт/сек.

- Автономный режим. Поддержка SD карт, объемом до 32ГБайт. Поддерживается только режим чтения карты. Запись на карту не работает.

- Автономный режим. Встроенная память 4МБайта. Поддерживаются все режимы работы: чтение, запись, форматирование.

- Встроенный разъем для внутрисхемного программирования с защитой от статики.

- Переходник для внутрисхемного программирования TSH-ICSP.

- Питание программатора от USB или внешний источник питания 5-12V.

- Питание программатора от внешнего источника питания 8-12V.

Время чтения и записи микросхем:

MX30LF1G08 (132 МБайт)	Чтение+Сверка:	38сек.	MX30LF1G08 (132 МБайт)	Чтение+Сверка:	592сек.
MX30LF1G08 (132 МБайт)	Запись+Сверка:	52сек.	MX30LF1G08 (132 МБайт)	Запись+Сверка:	690сек.
S29GL128S* (16 МБайт)	Чтение+Сверка:	~6сек.	S29GL128N* (16 МБайт)	Чтение+Сверка:	77сек.
S29GL128S* (16 МБайт)	Запись+Сверка:	17сек.	S29GL128N* (16 МБайт)	Запись+Сверка:	495сек.
W25Q128BV (16 МБайт)	Чтение+Сверка:	11сек.	W25Q128BV (16 МБайт)	Чтение+Сверка:	411сек.
W25Q128BV (16 МБайт)	Запись+Сверка:	56сек.	W25Q128BV (16 МБайт)	Запись+Сверка:	600сек.
W78E516D (64 КБайт)	Чтение+Сверка:	<3сек.	W78E516D (64 КБайт)	Чтение+Сверка:	10сек.
W78E516D (64 КБайт)	Запись+Сверка:	15сек.	W78E516D (64 КБайт)	Запись+Сверка:	26сек.

Электрические и прочие характеристики программаторов:

- два цифро-аналоговых преобразователя с электронной калибровкой от 1,5v до 16v (26v), с шагом 0,125v;
- прецизионная установка напряжений питания и записи, вне зависимости от тока, потребляемого микросхемой;
- падение напряжения на ключах программирования не более 20mv при токах до 80mA;
- на каждом выводе панельки – формирователи фронта импульса и уровня логического сигнала;
- отсутствие напряжений на контактах панели и антистатическая защита в исходном состоянии;
- автоматическое определение и контроль типа микросхемы, защита от неправильной установки;
- быстродействующая электронная защита ключей программатора от перегрузок и короткого замыкания;
- точное соблюдение всех временных и электрических параметров в соответствии с фирменными спецификациями;
- возможность ручной настройки и сохранения любых параметров программирования;
- чтение, программирование и сверка при различных напряжениях питания;
- поддержка всех режимов работы микросхемы в едином цикле записи или с помощью отдельных команд.

Разъемы и механические характеристики:

- USB разъем для подключения к компьютеру.
- Разъем для внутрисхемного программирования.
- Разъем для установки карты micro SD.
- Разъем для подключения дополнительного БП.
- Размер программатора – 170x100x23мм.
- Вес программатора – 220г.
- USB разъем для подключения к компьютеру.
- Разъем COM порта для подключения к компьютеру.
- Разъем для подключения источника питания.
- Размер программатора – 170x100x23мм.
- Вес программатора – 235г.

Комплект поставки: (переходные панельки и адаптеры в комплект поставки НЕ ВХОДЯТ и приобретаются отдельно)

- Универсальный программатор ТРИТОН V5.8TU.
- Интерфейсный кабель High Speed USB.
- Плоский кабель для внутрисхемного программирования.
- Сетевой блок питания (5v @ 1000mA).
- Компакт-диск с программным обеспечением.
- Гарантийный талон.
- Упаковочная коробка.
- Универсальный программатор ТРИТОН V5.7TU.
- Интерфейсный кабель USB.
- Интерфейсный кабель COM (комплектуется по запросу).
- Сетевой блок питания (12v @ 500mA).
- Компакт-диск с программным обеспечением.
- Гарантийный талон.
- Упаковочная коробка.

- Вес в упаковке – 400г.

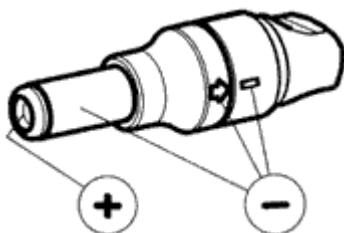
- Вес в упаковке – 460г (530г с COM кабелем).

Интерфейс и питание программатора V5.8TU.

Программатор может быть подключен к любому USB порту компьютера через силовой USB кабель. Для обеспечения стабильной работы рекомендуется использовать кабель длиной не более 1м. При подключении к порту USB 2.0 максимальная скорость передачи данных составляет около 9 МБайт/сек. Реальная скорость работы зависит от типа микросхемы и времени выборки или частоты тактового сигнала.

Питание программатора ТРИТОН V5.8TU может осуществляться через USB разъем или от внешнего источника питания. Потребляемый ток в режиме ожидания не превышает 50mA (90mA для версий с OLED дисплеем). Во время работы, ток зависит от выбранной микросхемы и, для большинства современных микросхем, не превышает 500mA. Вход программатора имеет самовосстанавливающуюся защиту по максимальному току (более 500mA) и защиту от подачи напряжения другой полярности.

При записи микросхем с напряжением программирования более 15v **необходимо** использовать внешний источник питания с напряжением **5-12v**. 12-вольтовый источник питания должен обеспечивать ток не менее 300mA. Внешний 5-вольтовый источник питания должен обеспечивать ток не менее 1A. Напряжение 5-вольтового источника питания под нагрузкой не должно проседать ниже 5V.



Программатор **не имеет гальванической развязки** ни по интерфейсу с компьютером, ни по цепям питания.

Запрещается подавать на вход программатора напряжение более 13v.

Некоторые USB хабы имеют схему защиты от перегрузок, деградирующую со временем. В результате чего, защита срабатывает при более низких значениях тока. Кроме того, компьютер может отключать питание USB устройства при кратковременных перегрузках по току. Поэтому, при появлении таких уведомлений и для обеспечения стабильной работы программатора **рекомендуется ВСЕГДА** использовать внешний источник питания.



Интерфейс и питание программатора V5.7TU.

Программатор имеет два независимых интерфейса и обеспечивает следующие варианты подключения и скорости работы:

	COM - порт	переходник USB - COM	USB
Прошивка Standard	10 кбайт/сек (115200 бод)	до 40 кбайт/сек (460800 бод)	до 250 кбайт/сек
Прошивка USB	не работает	не работает	до 750 кбайт/сек

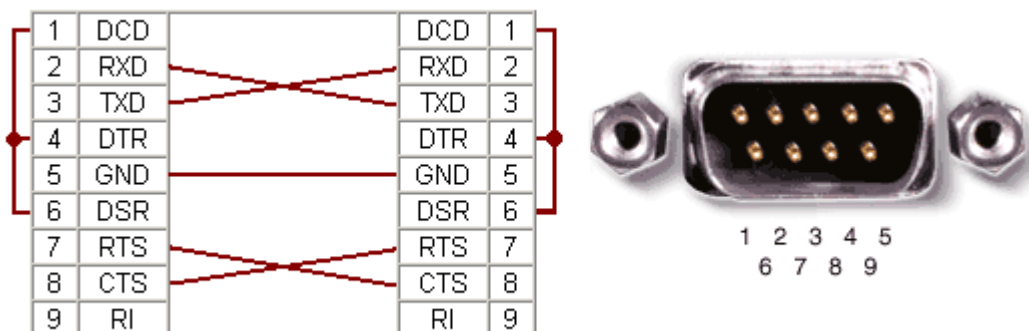
В состав программного обеспечения для программатора V5.7T входят две прошивки. Прошивка "Standard" поддерживает все микросхемы и все режимы работы программатора через интерфейсы COM и USB. Прошивка "USB", предназначена для подключения программатора только через USB, и работает, в среднем, в 2-4 раза быстрее. При подключении через COM порт прошивка "USB" с микросхемами не работает, но позволяет сменить прошивку в программаторе, загрузить в программатор файлы, проекты и списки микросхем.

USB порт.

Программатор может быть подключен к любому USB порту компьютера через стандартный кабель USB-AMBM. Для обеспечения стабильной работы с микросхемами большого объема рекомендуется использовать кабель не длиннее 1,8м с маркировкой HIGH SPEED. Программа не накладывает никаких ограничений на номер, который может иметь виртуальный COM порт. Для обеспечения максимальной скорости работы может потребоваться [настройка USB драйвера](#).

COM порт.

Для подключения к компьютеру через COM порт используется стандартный нуль-модемный кабель. Для работы используются две пары сигналов: TXD, RXD - для передачи данных, и CTS, RTS - для сигнализации о готовности.



Разводка кабеля 9 на 9 pin:

- вывод 2 - на вывод 3.
- вывод 3 - на вывод 2.
- вывод 7 - на вывод 8.
- вывод 8 - на вывод 7.
- вывод 5 - на вывод 5.
- вывод 9 - не подключать.
- выводы 1, 4 и 6 замкнуть между собой на каждом разъёме.

Дополнительная настройка порта не требуется. Штатные настройки порта игнорируются и программа сама задает все необходимые параметры. Единственный параметр, который доступен для изменения - скорость обмена, которую можно установить в меню "Программатор", "[Параметры и Тесты](#)".

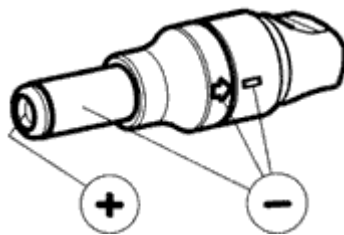
Переходник USB - COM.

После подключения переходника USB-COM к компьютеру и установки необходимых драйверов, в системе создается виртуальный COM порт, к которому может быть подключен программатор. Для многих переходников при работе в Win7, 8 и Windows Vista, возможно, потребуется обновить драйвера, которые можно скачать с сайтов производителей переходников.

Перед первым подключением переходника к компьютеру необходимо сначала установить драйвера, идущие в комплекте с переходником, после чего подключать сам переходник. На некоторых переходниках, при установке драйверов с помощью мастера установки нового оборудования, были отмечены сбои при работе на повышенных скоростях. При подключении этого же переходника после установки фирменных драйверов, подобные проблемы не наблюдались.

С переходниками, которые создают порты для подключения HID устройств, а также с переходниками от мобильных телефонов программаторы, как правило, не работают. Установленное на компьютере антивирусное программное обеспечение также может вызывать сбои при работе с микросхемами больших объемов.

Питание программатора ТРИТОН V5.7TU осуществляется от сетевого адаптера с выходным напряжением 12v и максимальным током 500mA. Программатор потребляет ток около 50mA в режиме ожидания и до 400mA в режиме программирования. Вход программаторов имеет самовосстанавливающуюся защиту по максимальному току (более 500mA) и защиту от подачи напряжения другой полярности.



Программаторы **не имеют гальванической развязки** ни по интерфейсу с компьютером, ни по цепям питания.

Запрещается подавать на вход программатора напряжение более 15v.

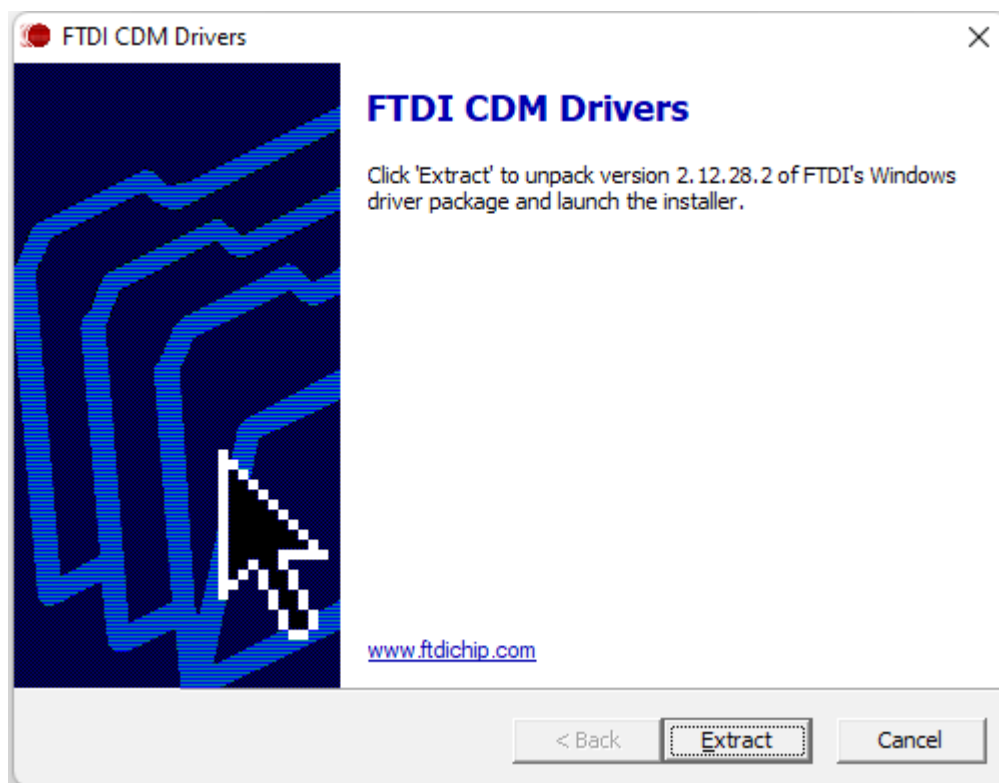
Установка драйверов и программы

Установите в компьютер компакт диск, идущий в комплекте с программатором и выполните автозапуск. Если автозапуск отключен, запустите программу **start.exe**.



Если компьютер не имеет CD привода, скачайте с сайта www.triton-prog.ru последнюю версию программного обеспечения TriSoft V5.8.xx и установочный файл USB драйвера.. Распакуйте скачанные файлы в любую папку на компьютере.

1. **Выполните установку USB драйвера.** Запустите файл **USB_Driver_Setup.exe** с CD диска или **CDMxxxxxx_Setup.exe**, скаченный с сайта и следуйте указаниям мастера установки.





Мастер установки драйверов устройств

Этот мастер поможет установить драйверы, необходимые для работы некоторых устройств.

Для продолжения нажмите кнопку "Далее".

< Назад

Далее >

Отмена

Лицензионное соглашение



Для продолжения необходимо принять лицензионное соглашение. Чтобы прочитать лицензионное соглашение, используйте полосу прокрутки или клавишу "Page Down".

IMPORTANT NOTICE: PLEASE READ CAREFULLY BEFORE INSTALLING THE RELEVANT SOFTWARE:

This licence agreement (Licence) is a legal agreement between you (Licensee or you) and Future Technology Devices International Limited of 2 Seaward Place, Centurion Business Park, Glasgow G41 1HH, Scotland (UK Company Number SC136640) (Licensor or we) for use of driver software provided by the Licensor(Software).

BY INSTALLING OR USING THIS SOFTWARE YOU AGREE TO THE

- Я принимаю это соглашение
- Я не принимаю это соглашение

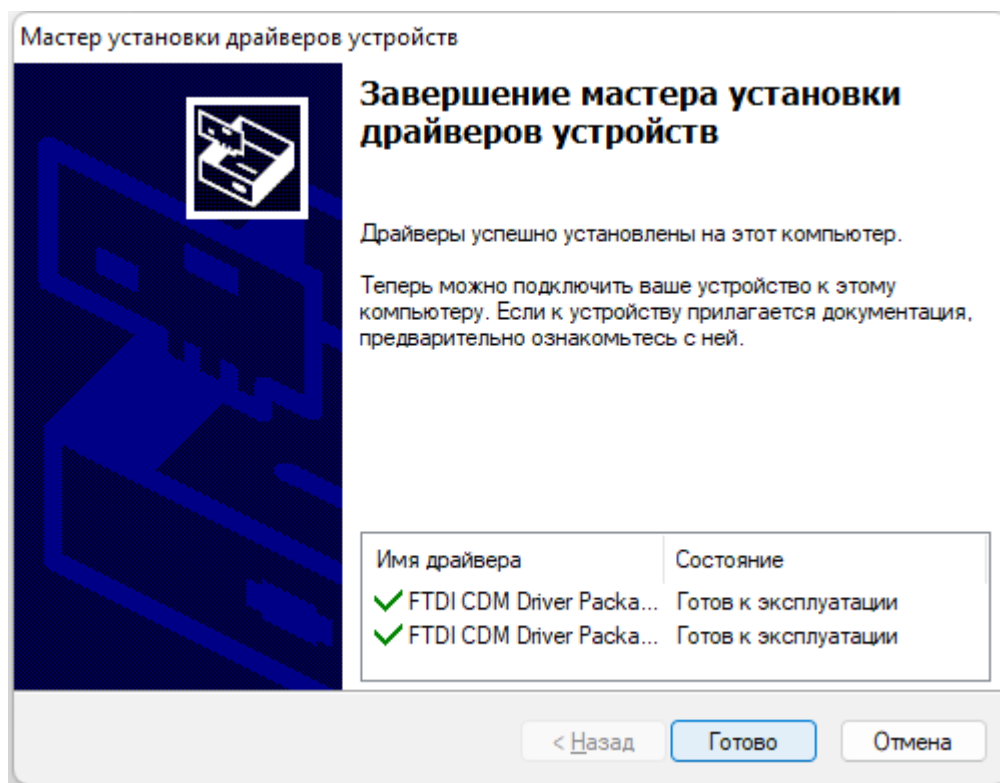
Сохранить как

Печать

< Назад

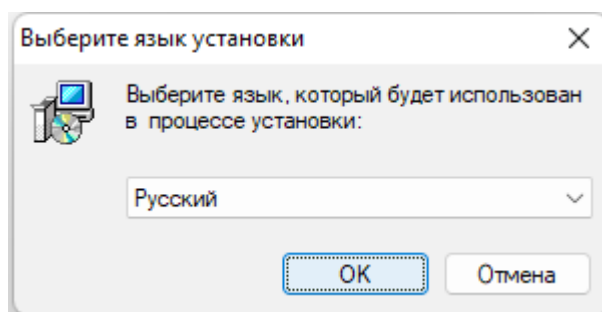
Далее >

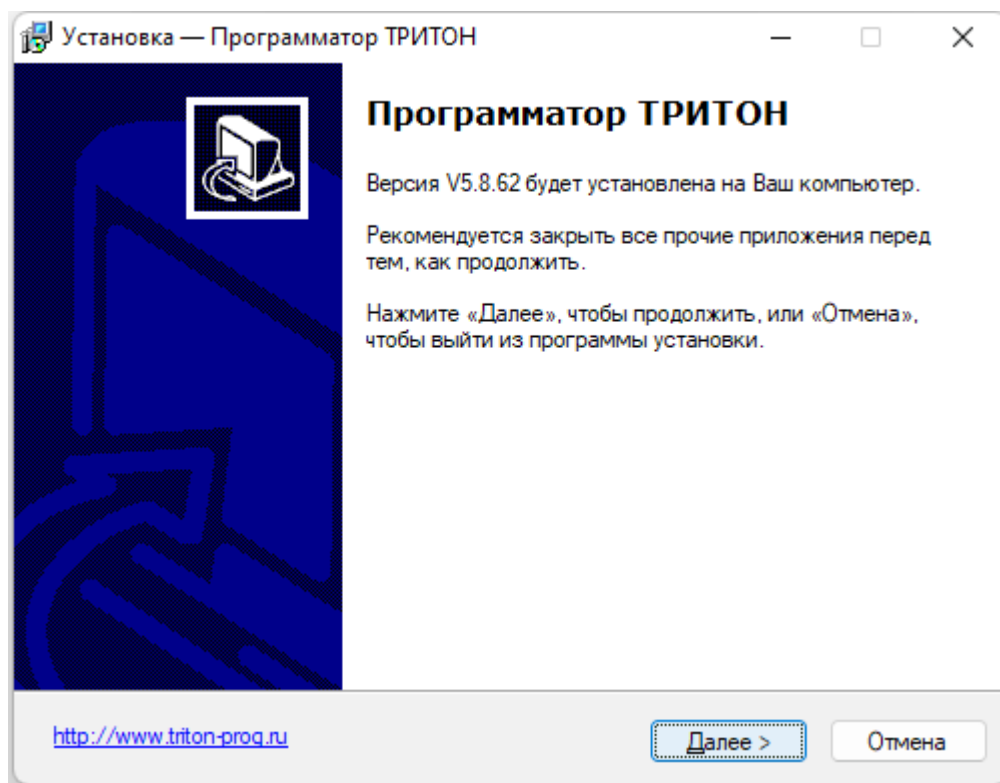
Отмена



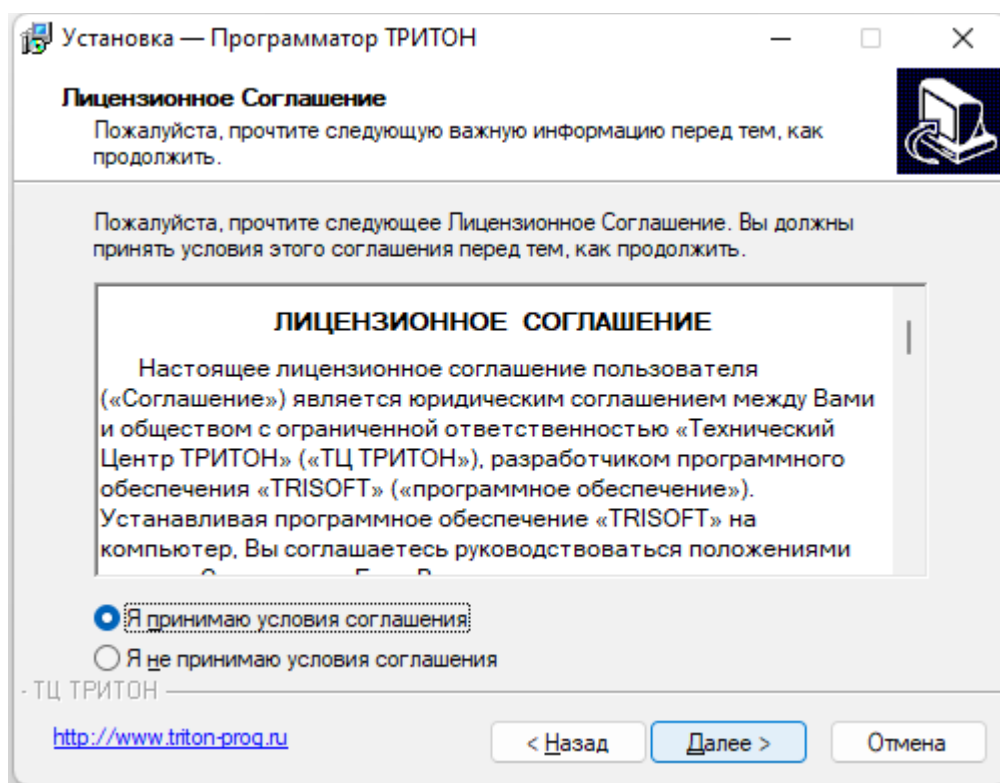
2. **Выполните установку программного обеспечения.** Запустите файл **V5_8_xx.exe** и следуйте указаниям мастера установки. При установке более новой версии программы нет необходимости деинсталлировать старую версию, но **необходимо закрыть работающую программу.**

Выберите язык, который будет использоваться мастером установки в процессе инсталляции.

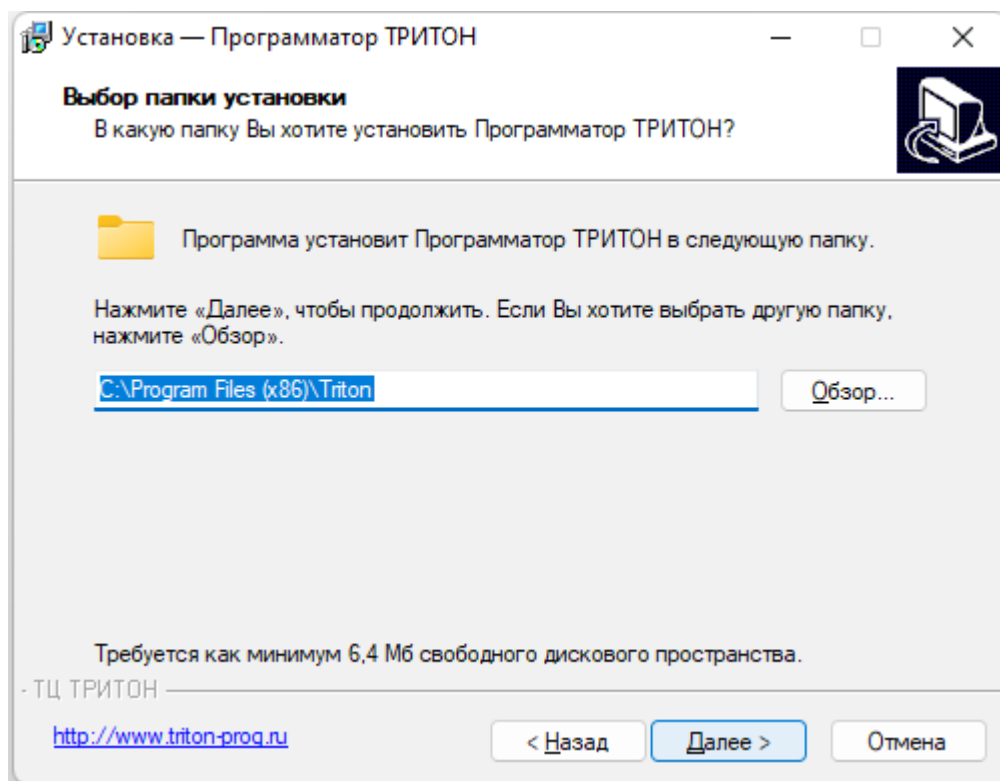




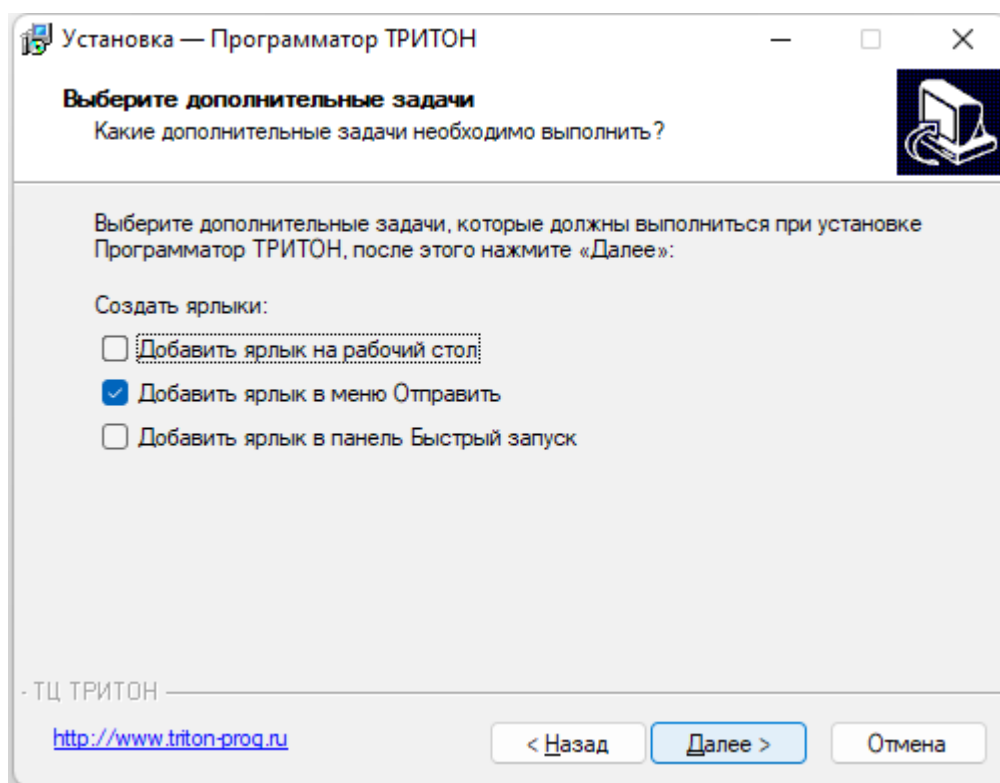
Внимательно прочитайте "Лицензионное соглашение". Оно содержит важную информацию о Ваших правах на использование программы и обязательствах фирмы "ТЦ ТРИТОН" по поддержке Программного обеспечения. Чтобы продолжить установку Вы должны принять условия данного соглашения.

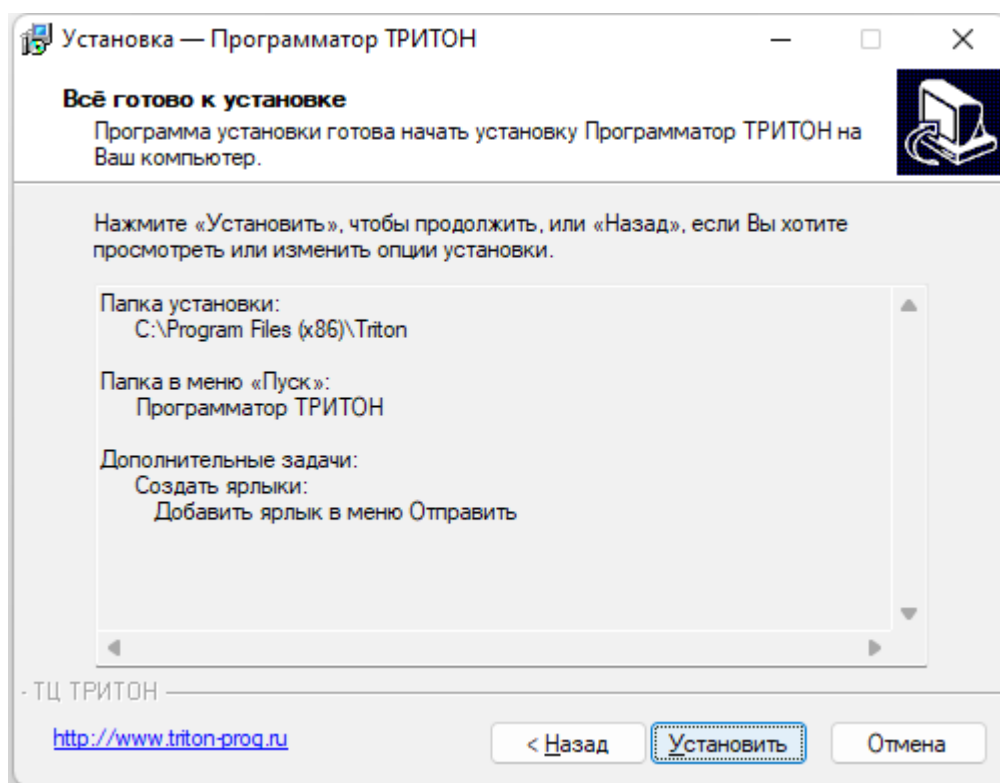


Выберите папку для установки программы. По умолчанию программа и необходимые для работы файлы будут установлены в папке "Program Files/Triton". При необходимости можно скопировать или переместить папку программы со всеми файлами в любую другую папку на компьютере.

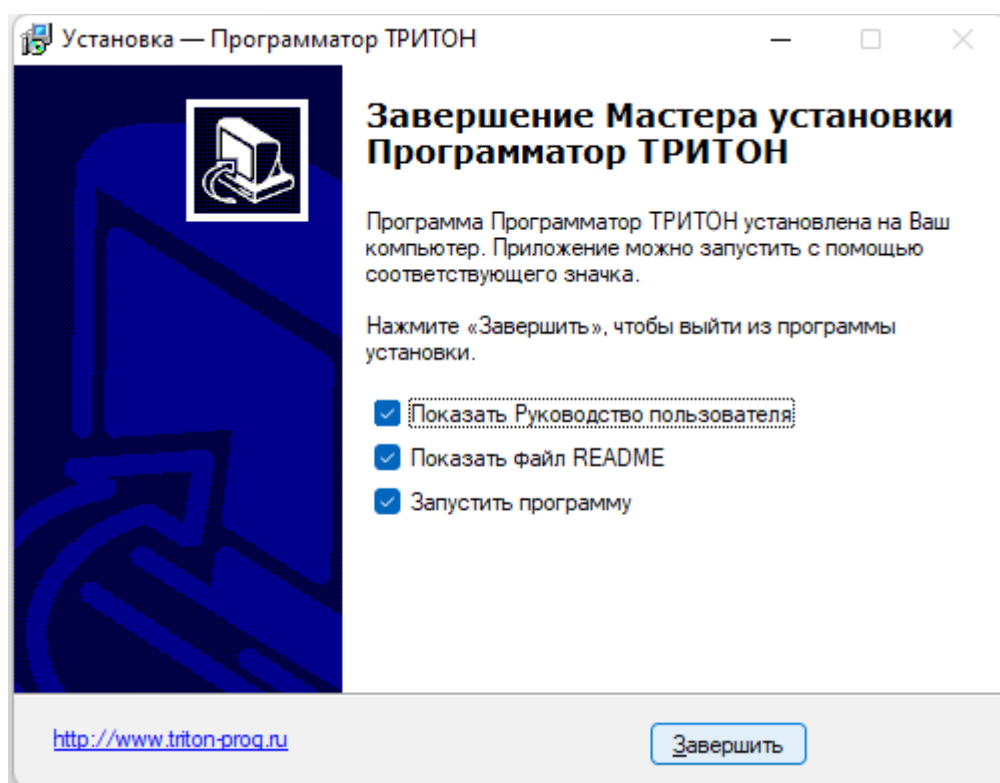


Настройте дополнительные опции инсталлятора.





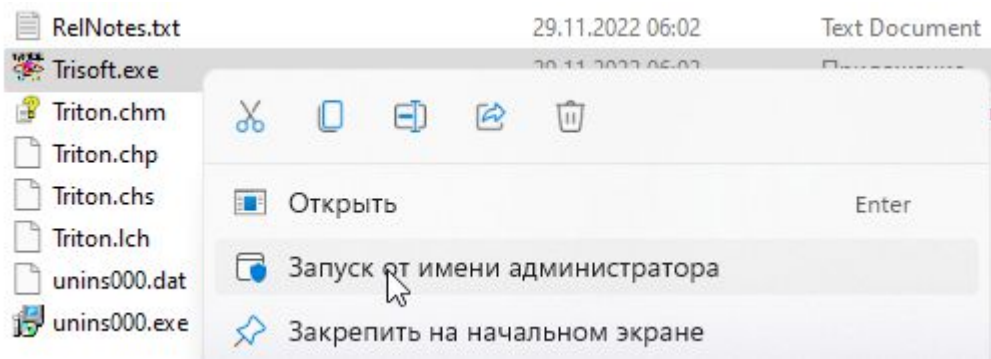
Если это первая установка программы на компьютере, то для пользователей Windows 8, 10 и 11 рекомендуется **снять** флаг **Запустить программу** и закрыть окно.



3. **Подключите программатор** к компьютеру через COM или USB кабель, затем включите питание программатора. Программатор V5.8T питается от USB, поэтому достаточно подключить USB кабель. На дисплее появится название программатора, информация о загруженной версии и через 3 секунды программатор перейдет в режим готовности.

При первом подключении программатора в USB порт, Windows обнаружит новое устройство и автоматически установит необходимый драйвер. При подключении программатора к другому USB порту, Windows определит его как новое устройство, присвоит ему другой номер и снова установит драйвер. Рекомендуется использовать один и тот же порт для подключения программатора или выполнить установку драйверов для каждого порта, куда может быть подключен программатор.

4. **Запустите программу.** Для пользователей Windows 8, 10 и 11 первый запуск программы или запуск при подключении программатора к другому USB порту, должен выполняться от имени **Администратора**.

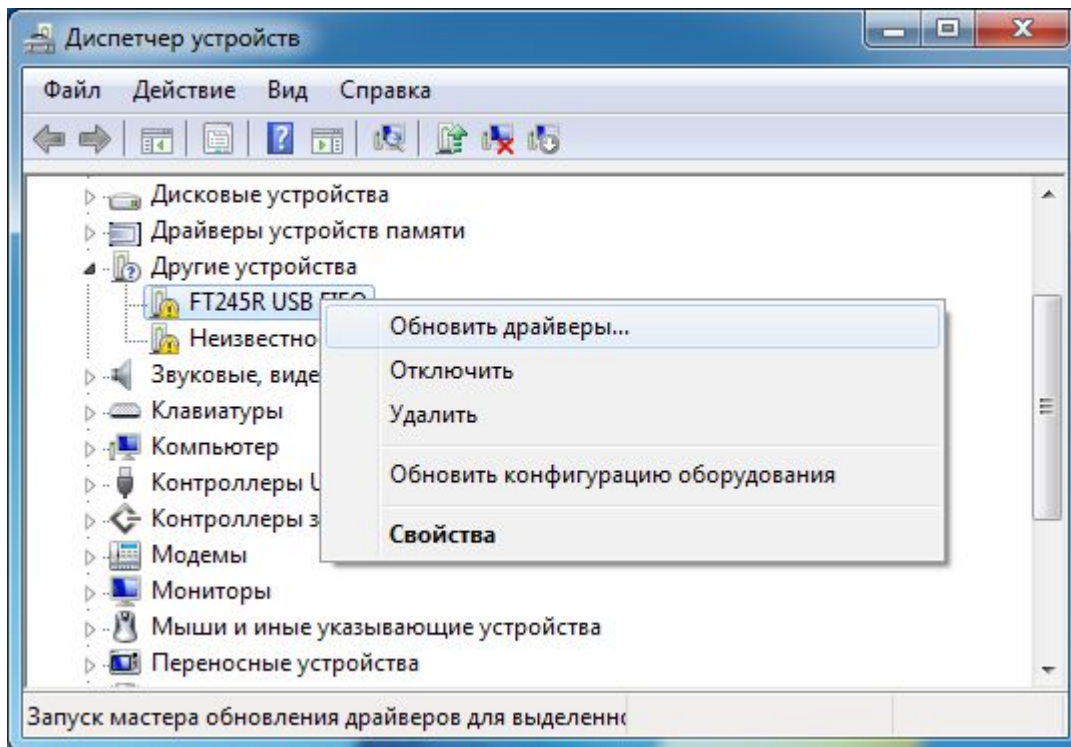


При запуске программа определит номер порта и будет использовать его по умолчанию при открытии новых окон. Если программатор не обнаружен, то программа предложит продолжить работу в демо-режиме и будет использовать номер порта, который использовался в последнем сеансе работы. Если программа до этого на компьютере не запускалась, то в этом случае будет установлен порт COM1.

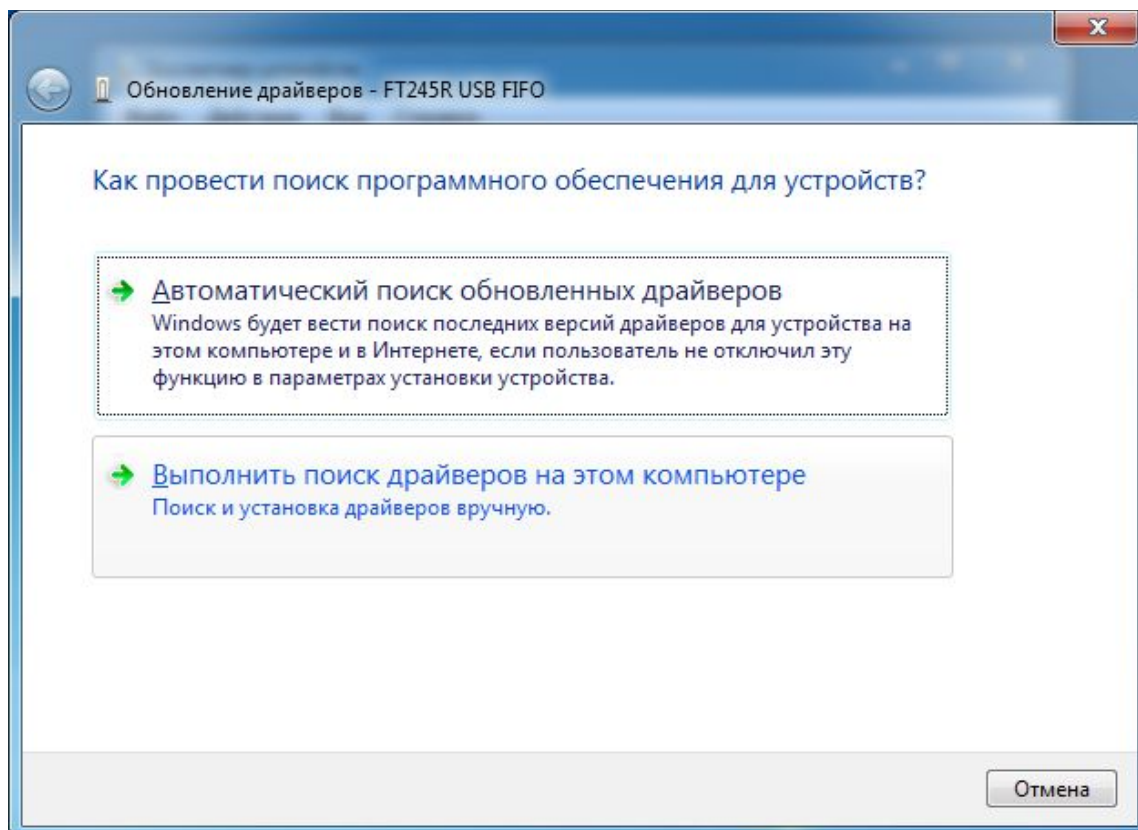
Ручная установка USB драйвера

Если драйвера не были установлены в автоматическом режиме, то нужно выполнить установку в ручном режиме. Установку драйверов и программы необходимо производить от имени **Администратора**. Предварительно необходимо отключить "**Контроль Учетных Записей**" (**УАС**) в профиле пользователя и **перезагрузить** компьютер.

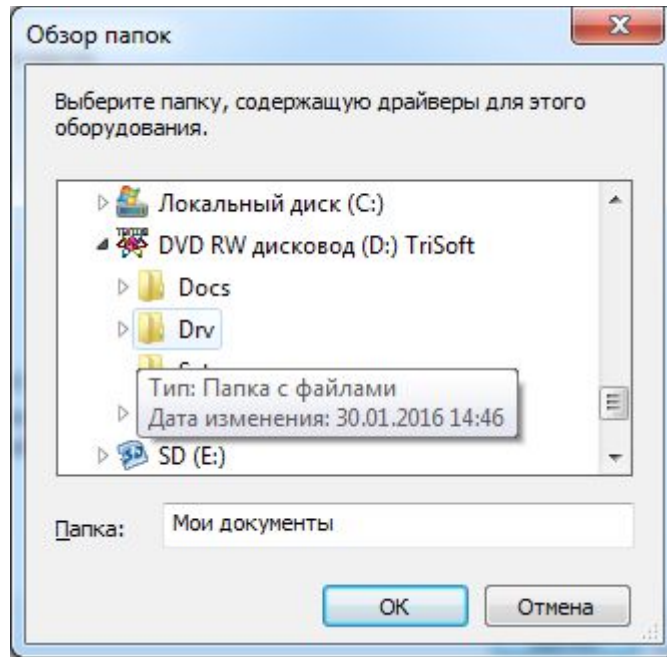
Подключите программатор к компьютер и включите питание программатора. В "Панели Управления" откройте **Диспетчер Устройств**. В разделе "Другие устройства" ("Other Devices") найдите устройство "FT245R USB FIFO", помеченное желтым восклицательным знаком, нажмите на нем правой кнопкой мыши и выберите пункт "Обновить драйверы" ("Update Driver Software").



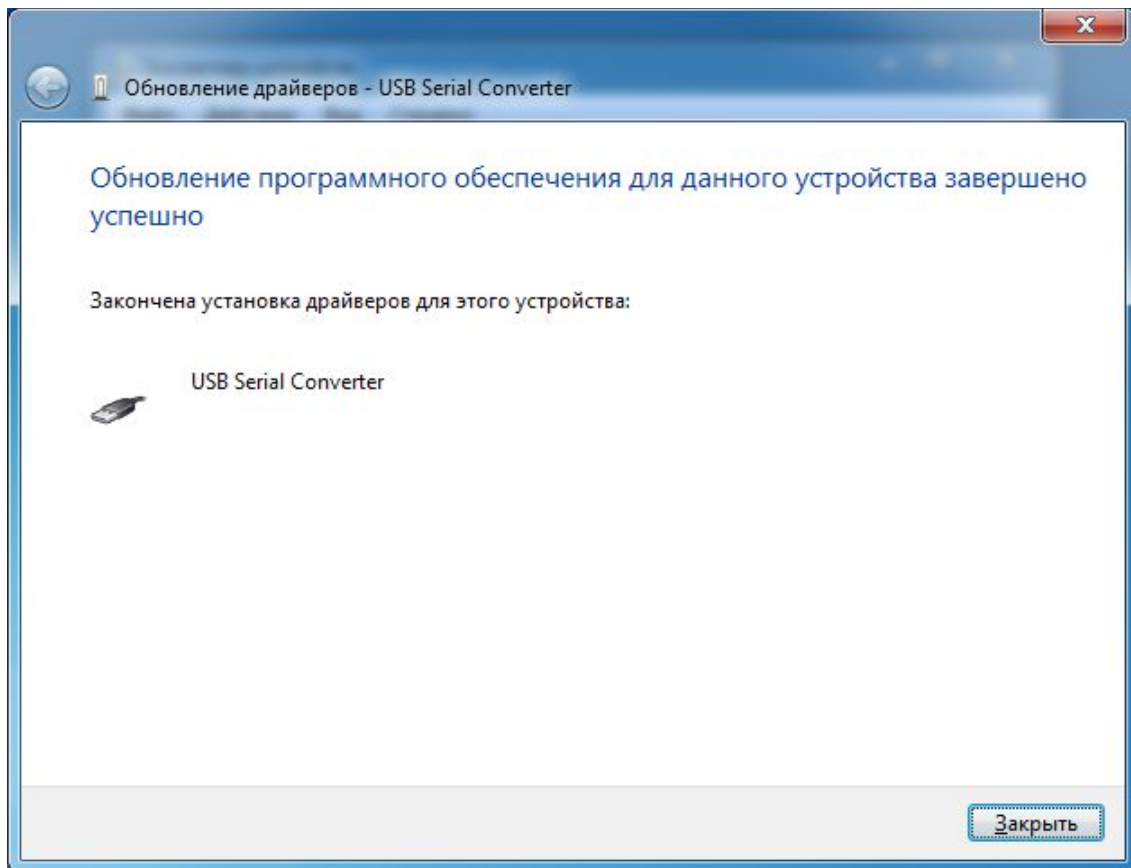
В окне "как производить поиск драйвера" выберите ручной поиск.



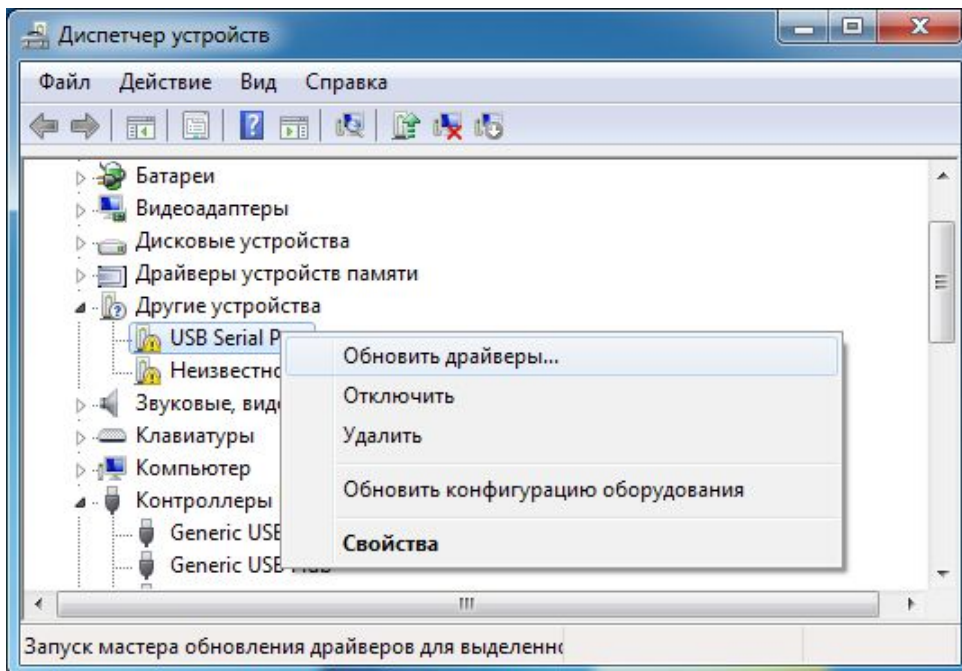
Установите компакт диск с драйверами или укажите папку, где находится драйвер и нажмите "Далее".



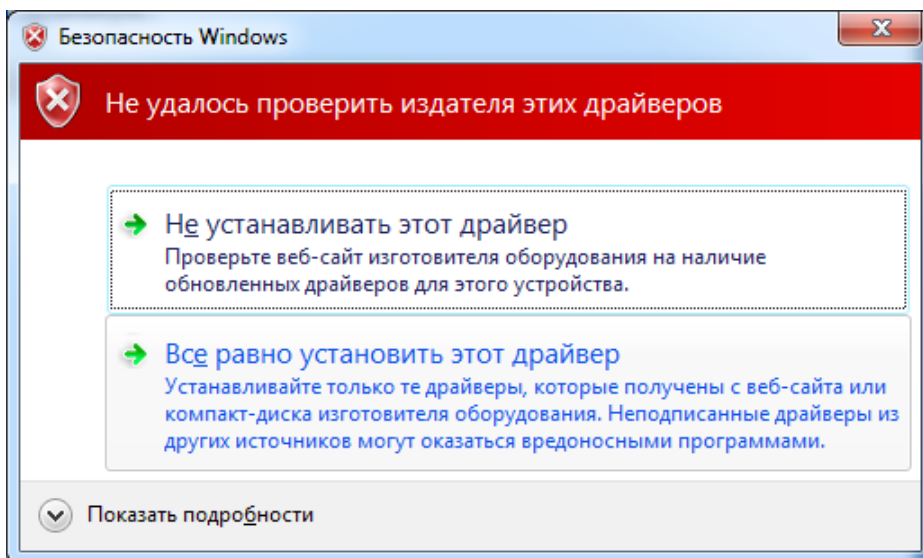
После сообщения завершения установки закройте это окно.



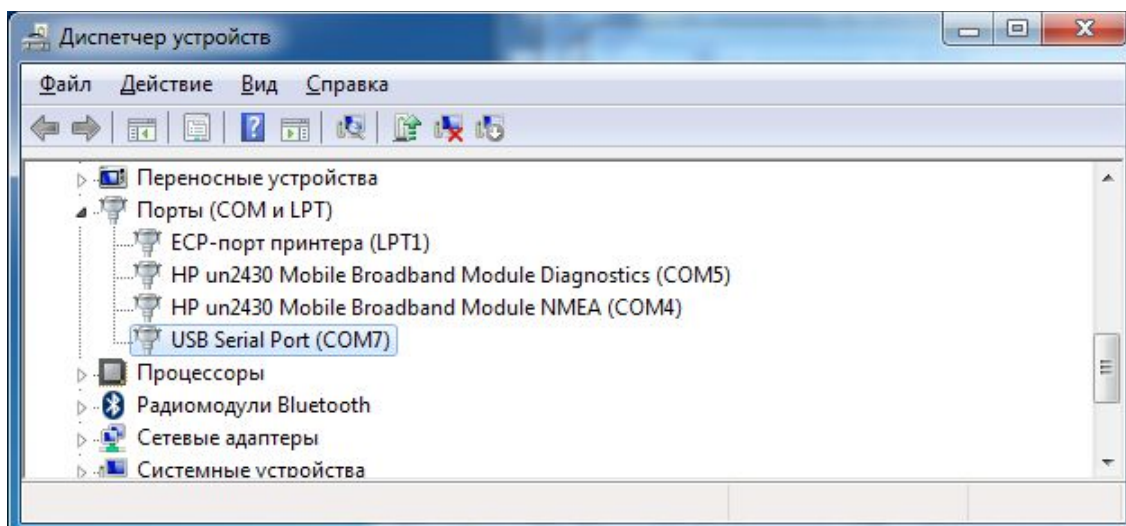
Снова вернитесь в окно **Диспетчер Устройств**. В разделе "Другие устройства" ("Other Devices") появится новое устройство "USB Serial Port", также помеченное желтым восклицательным знаком. Нажмите на нем правой кнопкой мыши, выберите пункт "Обновить драйвер" ("Update Driver Software") и повторите процедуру установки драйверов снова.



На сообщении мастера, о небезопасности программного обеспечения нажмите кнопку "Все равно установить этот драйвер".



Если все установлено правильно, то в разделе "Порты (COM & LPT)" появится новый COM порт. После завершения установки, необходимо выполнить [настройку USB драйвера](#).



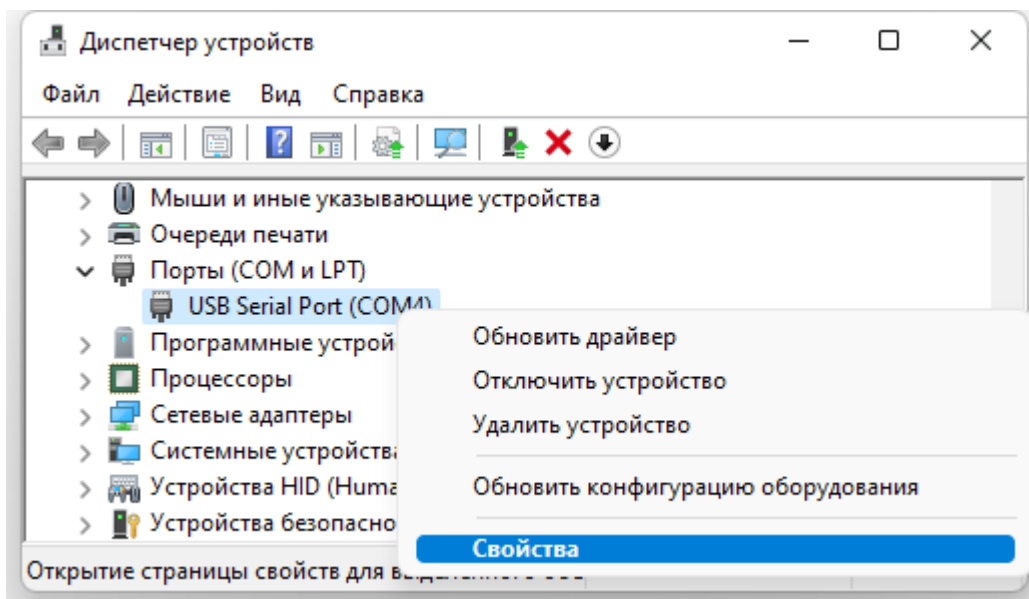
Если в процессе установки драйверов Windows сообщила об ошибке, то необходимо скачать новую версию драйверов с

сайта www.fdichip.com для своей операционной системы и снова повторить процедуру обновления драйвера. Правильно установленный драйвер видится в "Диспетчере Устройств" как "USB Serial Port (COM*)" и не должен быть помечен восклицательным знаком.

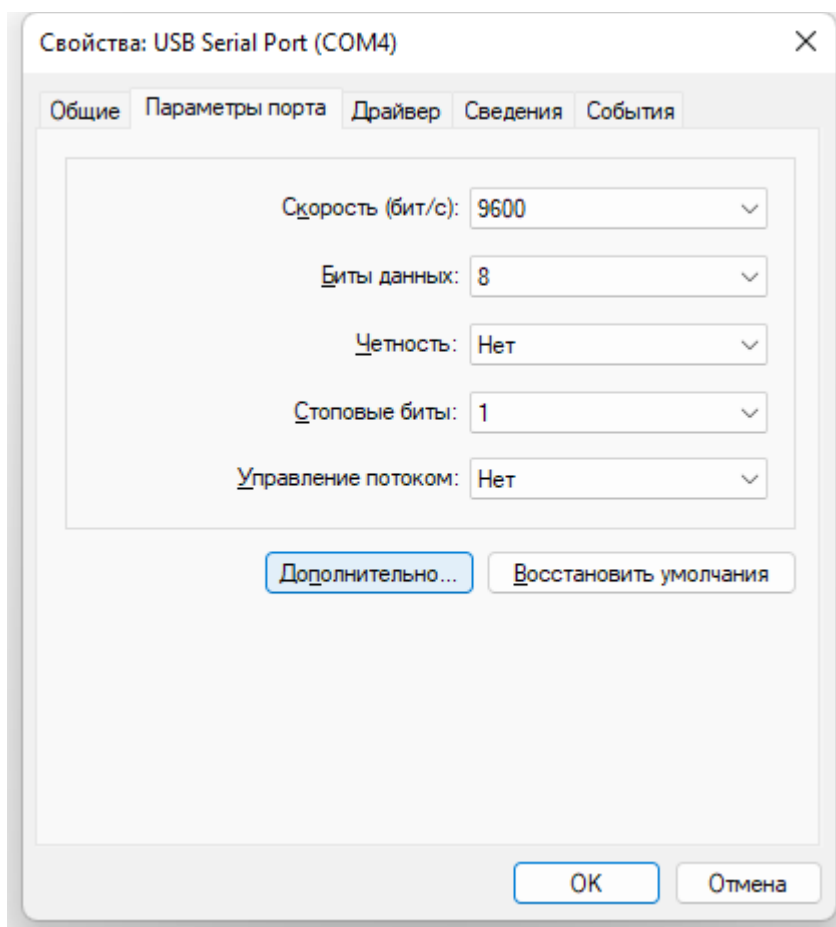
Настройка USB драйвера

Для обеспечения максимального быстродействия необходимо настроить параметры USB драйвера. Данная настройка не сильно влияет на скорость работы с микросхемами в прошивке USB. Но для программатора V5.7T с прошивкой STANDARD, этот параметр имеет ключевое значение.

Для этого в Диспетчере устройств, в разделе Порты (COM & LPT) нужно найти нужный порт, нажать на нем правой кнопкой мыши и выбрать Свойства.



Далее, выбрать закладку "Параметры порта" и нажать кнопку "Дополнительно".



Найти параметр "Время ожидания" (LATENCY TIMER) и установить его значение в единицу (по умолчанию равно 16).

Дополнительные параметры COM4

Номер COM-порта: COM4

Размер USB-пакета

Чтобы устранить проблемы с производительностью на низких скоростях передачи, попробуйте уменьшить значение.
Чтобы увеличить производительность, попробуйте увеличить значение.

Буфер приема (Байты): 4096

Буфер передачи (Байты): 4096

Дополнительные настройки для VM серии

При возникновении ошибок при приеме данных от устройства, попробуйте уменьшить значение.

Время ожидания (мсек): 1

Таймауты

Минимальное значение таймаута для чтения (мсек): 0

Минимальное значение таймаута для записи (мсек): 0

Дополнительные опции

- Обнаружение устройств Plug-and-Play
- Изменение таймаута при работе с принтером
- Отменить, если устройство выключено
- Уведомление при случайном отключении устройства
- Установить линию RTS при завершении работы
- Запретить управление модемом при старте
- Enable Selective Suspend
- Selective Suspend Idle Timeout (secs): 5

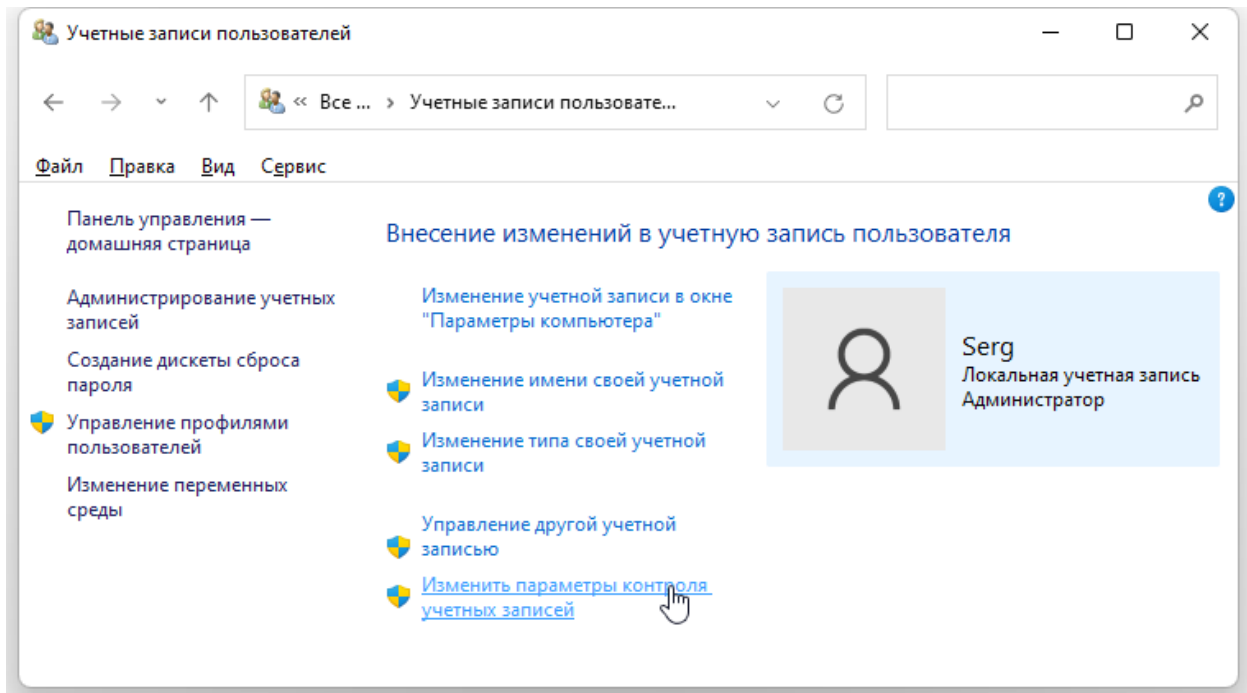
OK

Отмена

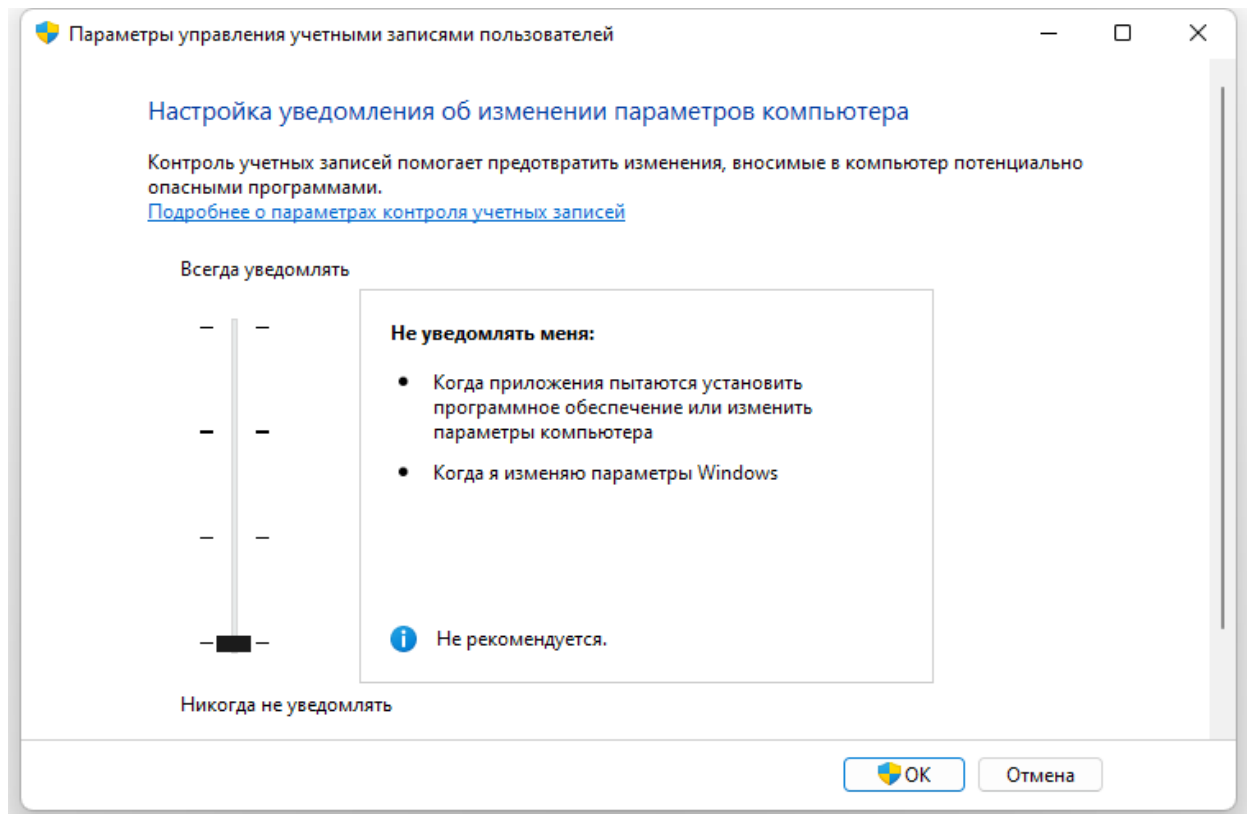
Умолчания

Устранение проблем с подключением

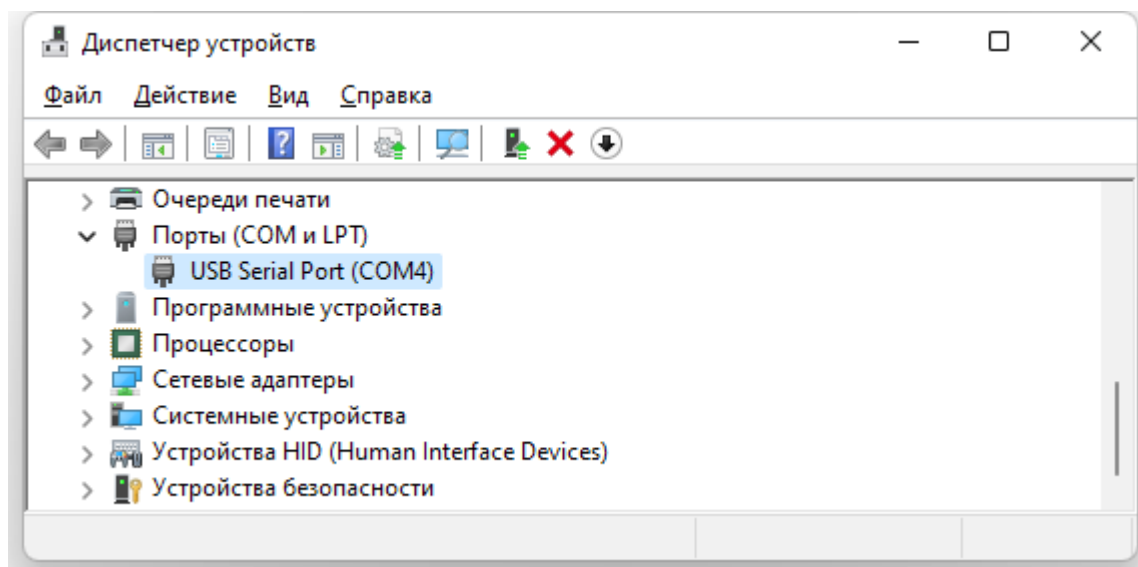
1. В **Панели управления** откройте **"Учетные записи пользователей"**. Убедитесь, что выбранный пользователь имеет права Администратора.



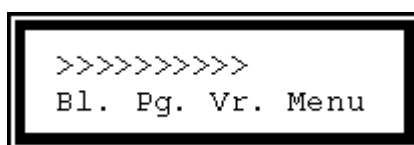
2. Для пользователей Windows 7, 8, 10 и 11 отключите "Контроль Учетных Записей" в профиле пользователя и **перезагрузите компьютер**.



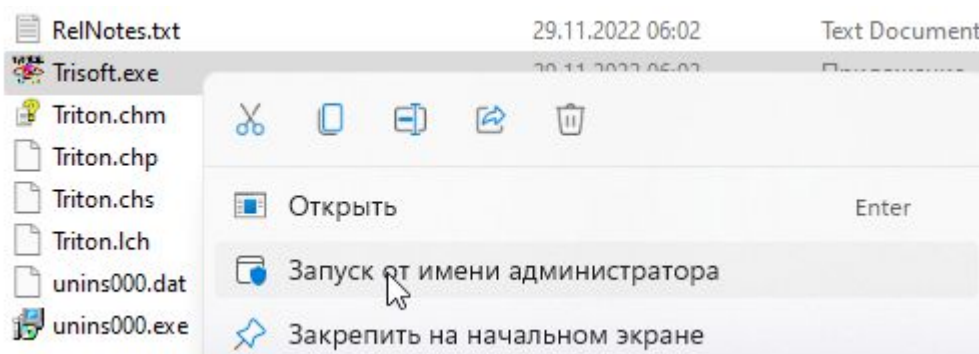
3. В **Панели управления** откройте **"Диспетчер Устройств"**, раздел **"Порты (COM и LPT)"** и проверьте, что при подключении программатора там появляется новый порт: "USB Serial Port (COM*)". Если такого устройства нет или оно помечено желтым восклицательным знаком, то выполните [установку или обновление USB драйвера](#).



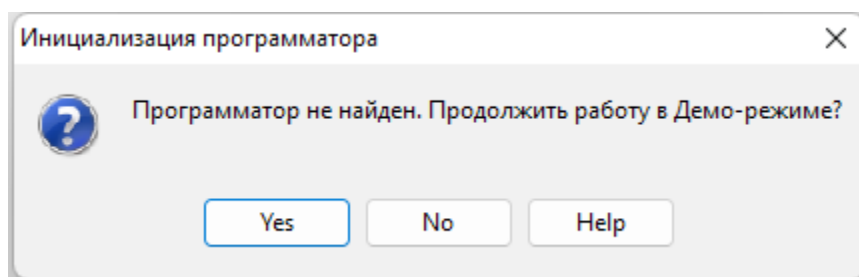
4. Проверьте, что программатор включен и находится в режиме готовности (в нижней строке написано "Bl. Pg. Vr. Menu").



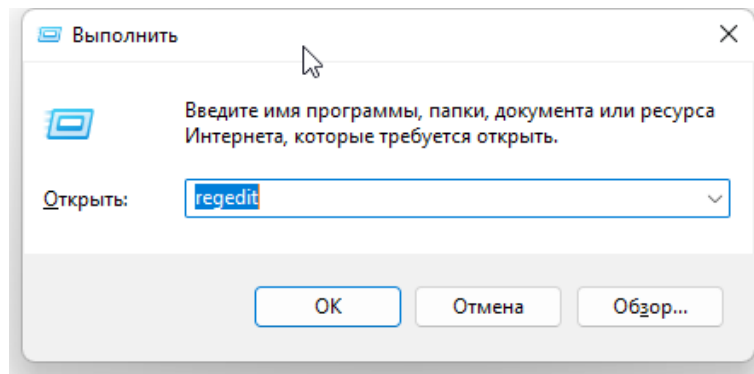
5. Запустите программу. Для пользователей Windows 8, 10 и 11 первый запуск программы или запуск при подключении программатора к другому USB порту, должен выполняться от имени **Администратора**.



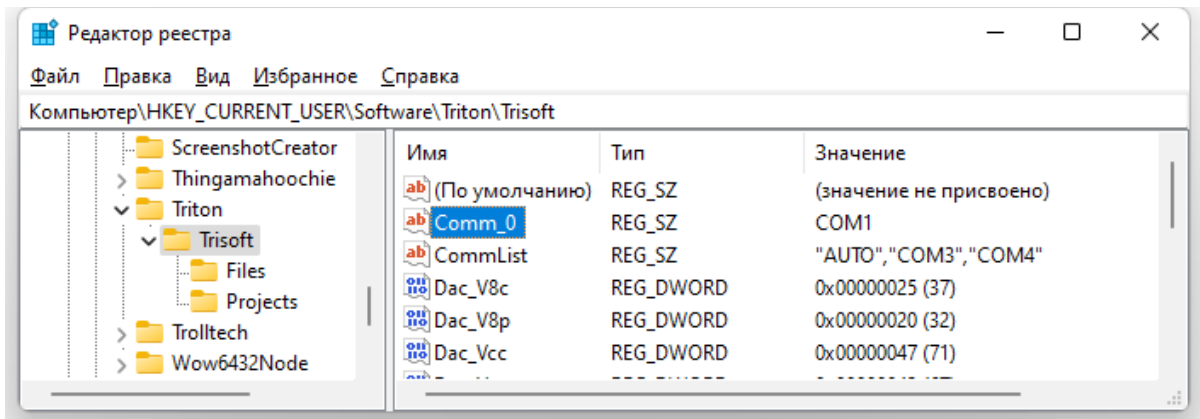
Если драйвера установлены правильно и в "Диспетчере Устройств" видится устройство "USB Serial Port (COM*)", но при запуске программы появляется сообщение "Программатор не найден" и в оболочке, в списке портов присутствует только "AUTO", и нет других портов, можно непосредственно задать номер порта, к которому подключен программатор.



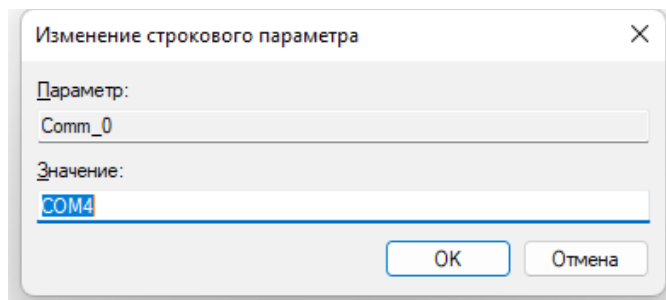
При появлении такого сообщения, выберите **НЕТ**, после чего **ЗАКРОЙТЕ ПРОГРАММУ** и откройте "Системный реестр".



Найдите ключ "HKEY_CURRENT_USER/Software/Triton/Trisoft".

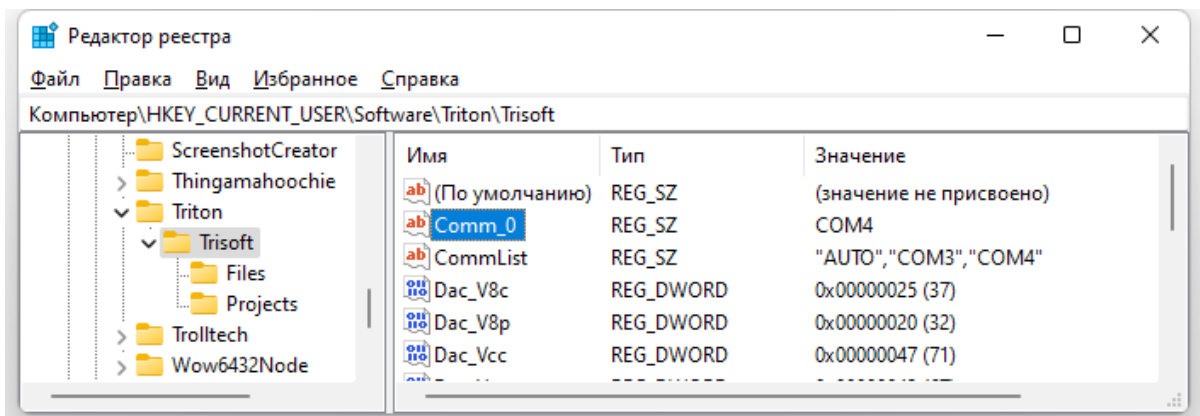


Дважды кликните мышкой по параметру "Comm_0" и в открывшемся окне введите номер порта, к которому подключен программатор. Номер порта отображается в "Диспетчере Устройств", в разделе "Порты (COM и LPT)", при подключенном программаторе. Имя порта вводится **латинскими заглавными буквами**.



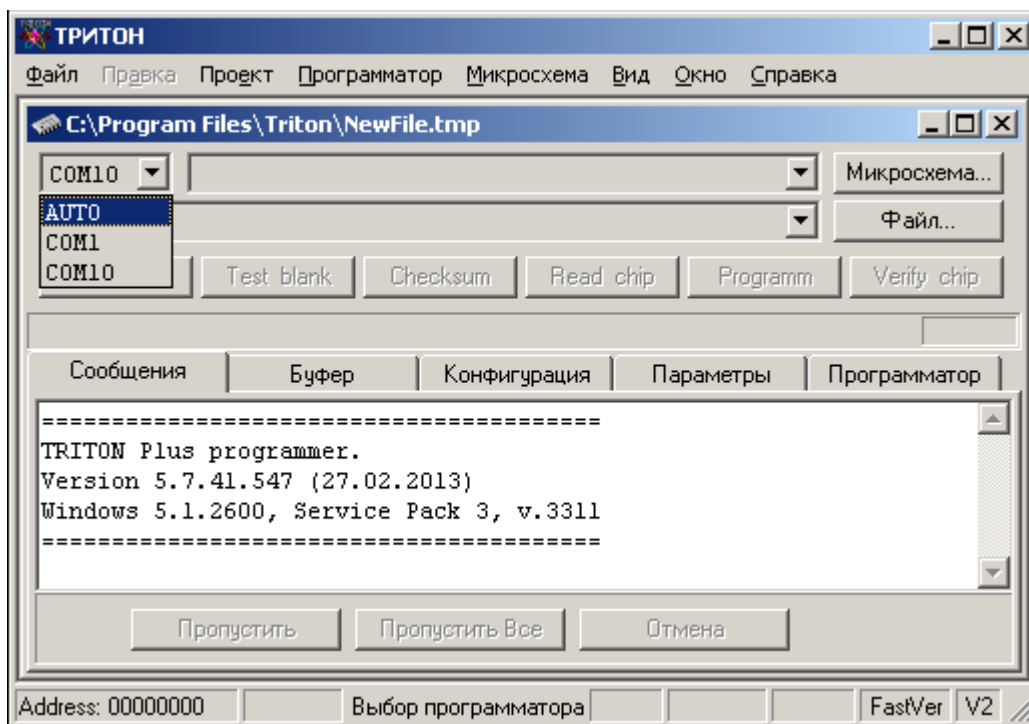
Закройте Системный реестр и снова запустите программу. Проводить автоматический поиск программатора уже не нужно.

Если планируется подключение нескольких программаторов или программатор будет подключаться к разным USB портам, то в параметре "CommList" можно ввести список портов, куда может быть подключен программатор. Не рекомендуется перечислять в этом списке все доступные комбинации, так как это может замедлить запуск программы. При запуске программа проверяет наличие программатора в порту, заданному в параметре "Comm_0" и, если он там не обнаружен, выполняет поиск по списку "CommList".



Начало работы

Если при запуске управляющей программы, программатор подключен и находится в режиме готовности, то программа определит номер порта и откроет новое окно проекта. Если программатор не найден, то программа предложит продолжить работу в демо-режиме. В демо-режиме невозможна работа с программатором и не производится сохранение настроек при закрытии программы. Выйти из демо-режима можно, подключив программатор и выбрав "AUTO" или указав номер порта, куда подключен программатор.



Ниже приведены несколько пошаговых инструкций, описывающих наиболее типовые задачи использования программатора. Выберите нужную задачу, кликните по ссылке и прочитайте подробное описание каждого процесса. Затем вернитесь обратно и посмотрите описание следующего действия.

Чтение содержимого микросхемы.

- [Выберите микросхему.](#)
- [Считайте микросхему.](#)
- [Сохраните полученный файл на компьютере.](#)

Обновление прошивки (программирование микросхемы).

- [Выберите микросхему.](#)
- Если микросхема снята с рабочего устройства, то [считайте](#) и [сохраните](#) старую прошивку.
- [Откройте файл](#), содержащий нужную прошивку.
- [Запрограммируйте](#) микросхему.

Копирование содержимого микросхемы.

- [Выберите микросхему.](#)
- [Считайте содержимое микросхемы.](#)
- [Запрограммируйте](#) новую микросхему.
- Если необходимо, [сохраните файл](#) на компьютере.

Программа TriSoft.exe

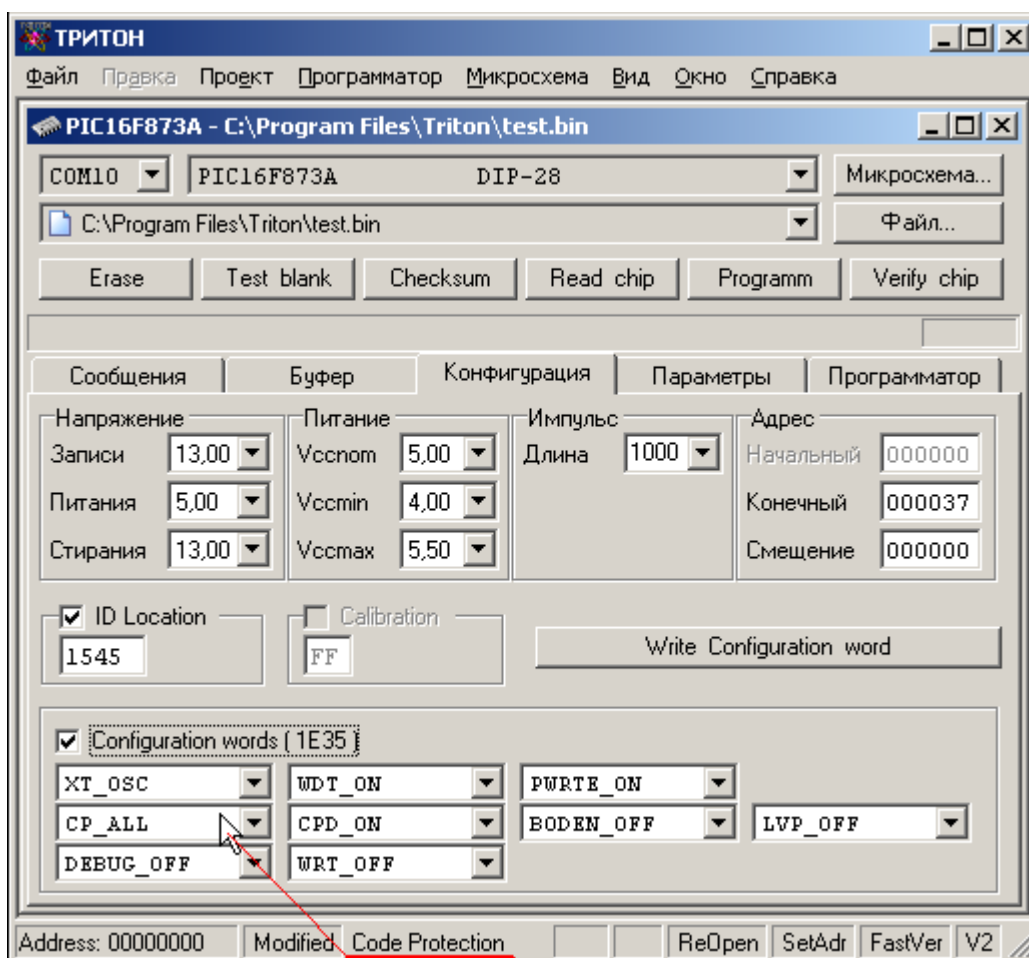
Оболочка программатора многооконная и многозадачная. Каждое окно в программе может управлять своим собственным программатором или несколько окон могут по очереди управлять одним программатором. Выбор программатора осуществляется в каждом окне из списка доступных портов.

Оболочка программатора предлагает два варианта хранения настроек. При эпизодическом использовании программатора или при работе с разными микросхемами достаточно установить флаг "[Восстанавливать установки при запуске программы](#)" и программа будет восстанавливать свое состояние, которое было в момент закрытия. Если планируется серийное программирование микросхем, с ведением статистики или когда необходимо перенести настройки с одного компьютера на другой, то лучше сохранить настройки в виде [проекта](#).

Существует два варианта запуска программы: непосредственный запуск программы (в меню "Пуск" или через ярлык программы) и запуск с помощью системных функций "Отправить" и "Открыть с помощью". Можно ассоциировать расширения нужных типов файлов (например, *.BIN, *.HEX, *.TPR, ...) с программой C:\Program Files\Triton\TriSoft.exe и открывать их двойным кликом. Также любой файл или проект можно открыть в программе, выбрав в контекстном меню команду "Отправить" в "Программатор ТРИТОН".

Тип запуска определяет логику работы программы и порядок сохранения настроек. При непосредственном запуске, программа проверяет подключение программатора и восстанавливает предыдущее состояние. При закрытии, программа обновляет данные в проектах и сохраняет все настройки. В остальных случаях, программа запускается в демо-режиме и проверка программатора не производится. В демо-режиме, при закрытии, программа не сохраняет никаких настроек. Выйти из демо-режима можно выбрав "AUTO" или указав номер порта, куда подключен программатор. Если при этом открыть проект или выбрать микросхему, то при закрытии, программа обновит данные в открытых проектах, но не сохранит текущие настройки.

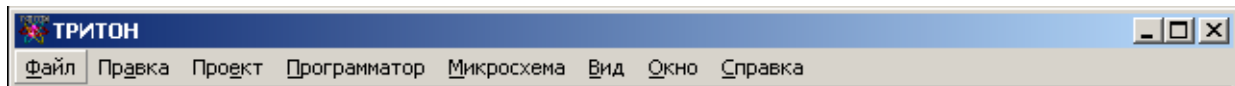
Программное обеспечение имеет встроенную контекстно-зависимую справку и развитую систему подсказок, которые позволяют быстро разобраться в программе и настроить нужные режимы работы. При перемещении указателя мыши в окне программы, в нижней части окна, в **строке состояния**, выводится краткое описание элемента, над которым находится курсор.



Для получения более подробного описания необходимо выделить интересующий элемент программы и нажать клавишу **F1**. Для вызова контекстно-зависимой справки из любого окна программы достаточно нажать клавишу **F1**. В самой справке доступен поиск информации по ключевым словам и навигация по разделам.

Меню программы

Главное меню программы.



Содержит следующие подменю:

- [Файл](#)
- [Правка](#)
- [Проект](#)
- [Программатор](#)
- [Микросхема](#)
- [Вид](#)
- [Окно](#)
- [Справка](#)

Навигация по меню может осуществляться как мышью, так и с помощью клавиатуры путем комбинации клавиш Alt + подчёркнутая буква в названии меню.

Меню Правка (Буфер)

Содержит команды, используемые для работы в шестнадцатеричном редакторе. Это меню доступно только когда активным является окно редактора. Во время работы с микросхемой редактор переводится в режим "ReadOnly" (только чтение) и команды, изменяющие содержимое буфера, работать не будут.

Многие команды этого меню продублированы на панели кнопок и в контекстном меню редактора (оно появляется при нажатии правой кнопкой мыши в окне редактора), и могут быть вызваны непосредственно с клавиатуры сочетанием клавиш. Редактор имеет стандартный набор команд для редактирования данных в ASCII или HEX форматах, в режиме вставки или замены. Кроме того, редактор поддерживает операции выделения и перемещения выделенного блока данных при помощи мыши.

Отменить (Ctrl+Z) - последовательно отменяет внесенные изменения. Поддерживается отмена всех режимов и команд. Для работы используется специальный буфер, позволяющий запомнить до 1000 команд. Если суммарный объем буфера файла и буфера отмены превышает размер свободной памяти компьютера, то отмена таких команд будет невозможна. Для отмены всех внесенных изменений можно воспользоваться командой "[Перезагрузить файл](#)" в меню "Файл".

Удалить (Del) - удаляет выделенный фрагмент. Если выделенных данных нет, то в зависимости от формата представления данных удаляется байт или слово, стоящее после курсора. Размер файла уменьшается на количество удаленных байтов.

Вырезать (Ctrl+X) - удаляет выделенный фрагмент в буфер. Размер файла уменьшается на размер выделенного блока. Для работы используется специальный буфер, доступный только внутри программы. Данные из этого буфера будут использоваться при вставке и в меню "Функция".

Копировать (Ctrl+C) - копирует выделенный фрагмент в буфер. Эта команда имеет несколько модификаций, позволяющих копировать весь выделенный фрагмент, или только четные или нечетные байты. Формат представления данных на эту команду не влияет. Для работы используется специальный буфер, доступный только внутри программы. Данные из этого буфера будут использоваться при вставке и в меню "Функция".

Вставить (Ctrl+V) - вставляет данные из буфера в позицию курсора в режиме замены или вставки (Insert). Если включен режим вставки, то объем файла увеличивается на количество вставленных байтов. Если во время вставки в редакторе был выделен блок данных, то сначала он будет удален, а затем будут вставлены данные из буфера.

Буфер обмена - Копировать (Shift+Ctrl+C) - копирует выделенный фрагмент в буфер обмена Windows (Clipboard). Если фрагмент выделен в шестнадцатеричной области редактора, то данные будут преобразованы в HEX формат и разбиты по строкам в 32 символа. Если данные выделены в текстовой области редактора, то они будут скопированы в буфер без преобразования. Формат представления данных на эту команду не влияет.

Буфер обмена - Вставить (Shift+Ctrl+V) - вставляет данные из буфера обмена Windows (Clipboard) в позицию курсора в режиме замены или вставки (Insert). Кроме текстовых форматов, программа позволяет вставлять скопированные фрагменты изображений и звуков. Если курсор установлен в текстовой области редактора, то любые данные вставляются без преобразования. Если курсор установлен в шестнадцатеричной области редактора и в буфере обмена находятся текстовые данные, то программа выполнит преобразование и вставит данные в двоичном формате. Программа позволяет преобразовывать и вставлять данные в бинарном, десятичном или шестнадцатеричном форматах.

В **бинарном** формате исходные данные могут быть с префиксом: "0b" или без него, с любыми разделителями или в виде массива бит. Формат вставляемых данных всегда = 8 бит. Если длина массива не кратна 8, то последний байт будет дополнен нулями справа. Например,

Исходный текст в буфере обмена:	Что будет вставлено в редакторе:
0b10000001, 0b11110111, 0b10000000, 0b01001111	= 81 F7 80 4F
10000001,11110111_10000000-01001111	= 81 F7 80 4F
1000000111101111000000001001	= 81 F7 80 48 - длина массива 29 бит + 3 бита нулей.

В **десятичном** формате исходные данные должны содержать только цифры и разделители. При вставке программа проверяет все числа и если все они меньше 256, то данные вставляются в 8 битном формате. Если хотя бы одно из них больше или равно 256, то все данные вставляются в 16 битном формате. Если имеются числа больше 65535, то данные будут вставлены в 32 битном формате.

Исходный текст в буфере обмена:	Что будет вставлено в редакторе:
8 96 104 255	= 08 60 68 FF
8 96 104 65535	= 08 00 60 00 68 00 FF FF (WORD: 0008 0060 0068 FFFF)
8 96 104 16777215	= 08 00 00 00 60 00 00 00 68 00 00 00 FF FF FF 00 (DWORD: 00000008 00000060 00000068 00FFFFFF)

В **шестнадцатеричном** формате исходные данные могут быть с префиксом: "0x" или "\$" или без префикса. Данные должны быть разделены любыми разделителями. 16, 24 или 32 разрядные данные размещаются, начиная с младшего байта. Если данные не содержат разделителей и длина массива больше 8 знаков, то данные будут вставлены в 8 битном формате.

Исходный текст в буфере обмена:	Что будет вставлено в редакторе:
---------------------------------	----------------------------------

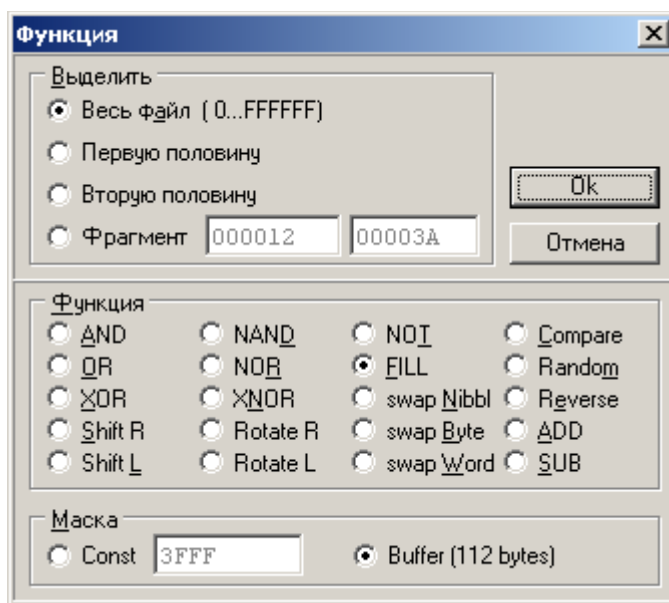
0x0, 0x00, 0x79, \$F0, \$75 = 00 00 79 F0 75
 0x0000, 0x1279, F0, 75 = 00 00 79 12 F0 75 (WORD: 0000 1279 75F0)
 00FF00C0C1C1810140 = 00 FF 00 C0 C1 C1 81 01 40

Выделить Все (Ctrl+A) - выделяет все данные в буфере.

Выделить блок - выводит окно диалога, в котором можно задать начальный и конечный адрес блока, выделить весь файл целиком или любую его половину. При вводе адресов программа учитывает формат представления данных.

Для микросхем NAND-Flash доступен специальный формат адреса: **0-1FF**, который позволяет задать номер страницы и адрес байта внутри страницы. Например, чтобы выделить две полные страницы, начиная с первой, надо ввести 1-0 и 2-FFFF. Размер страницы зависит от выбранной микросхемы. Чтобы не гадать с этим размером, можно ввести FFFF и адрес будет установлен равным размеру страницы -1.

Функция (F2) - выводит окно диалога, в котором можно задать начальный и конечный адреса блока и выбрать нужный режим работы.



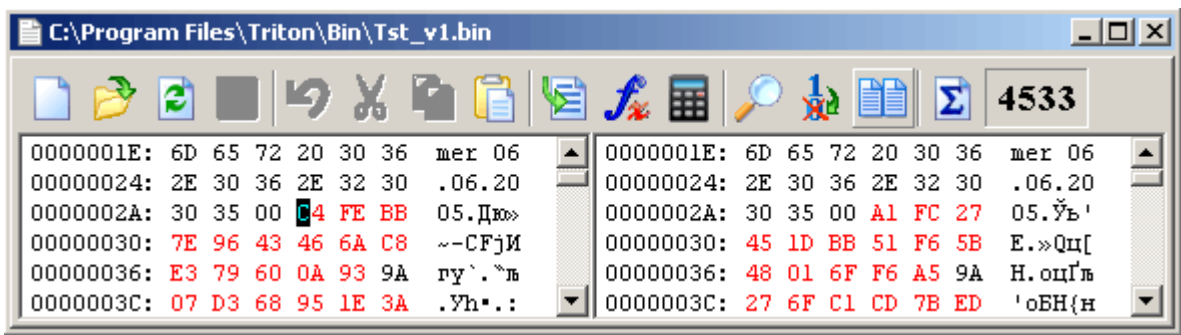
Редактор позволяет выполнить над выделенным блоком следующие функции:

- любую логическую операцию (AND, OR, XOR, NOT, NAND, NOR, XNOR);
- математические операции (ADD, SUB);
- циклический сдвиг каждого байта вправо или влево (Rotate: деление или умножение на 2);
- логический сдвиг каждого байта или всего блока вправо или влево (Shift);
- заполнение выбранным кодом или блоком данных из буфера (FILL);
- заполнение блока случайными значениями (Random);
- перестановка местами тетрад, байтов или слов (swap Nibbl, Byte, Word);
- изменение порядка бит в байте (Reverse);
- сравнение блока с заданным байтом или с данными из буфера (Compare).

В качестве маски для логических операций, сравнения или заливки может использоваться константа, размером от 1 до 4 байт или блок данных, предварительно скопированный в буфер. Размер данных, помещенных в буфер, может быть любым. Если буфер пуст, то в качестве маски будет использован введенный байт (по умолчанию это \$00). Все операции производятся только над выделенным фрагментом, и не зависят от длины данных, помещенных в буфер. При достижении конца буфера указатель данных в буфере будет перемещен в начальную позицию. Так будет продолжаться до конца выделенного фрагмента.

Сравнение. Программа позволяет выполнить сравнение данных, находящихся в буфере, с константой, размером от 1 до 4 байт, со скопированным фрагментом данных или с отдельным файлом. Для сравнения с константой или фрагментом данных, используется команда **Compare** в меню **Функция (F2)**.

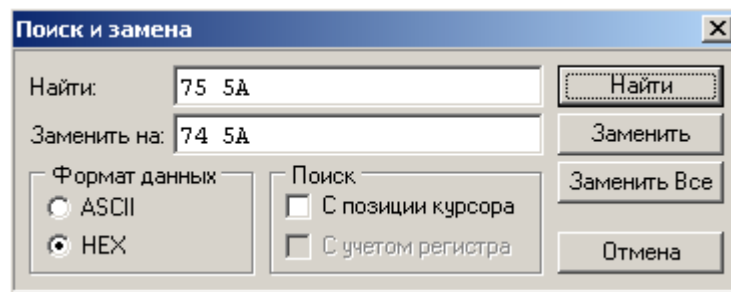
Для сравнения с файлом, используется меню **Сравнение** или кнопка **Compare**. В открывшемся окне диалога нужно выбрать файл, после чего программа создаст второе окно и выполнит сравнение данных.



После окончания сравнения, на экран выводится сообщение о результатах операции. Несовпадающие ячейки в редакторе будут выделены красным цветом. При нажатии клавиши ALT и стрелок курсора вправо (влево), курсор будет перемещаться к следующей (предыдущей) несовпадающей ячейке.

Калькулятор - выводит стандартный диалог выбора [файла калькулятора](#) и исполняет находящийся в нем скрипт. Файл калькулятора это обычный текстовый файл, содержащий C-подобный скрипт, с помощью которого можно вести диалог с пользователем и модифицировать данные в дампе памяти. Калькулятор выводится и работает в активном дампе (т.е. в том окне редактора, где находится курсор).

Поиск и замена (Ctrl+F) - обеспечивает поиск и замену введенной строки в ASCII или HEX формате. Если в этот момент в редакторе имеется выделенный фрагмент, то он будет использоваться в качестве поисковой строки. Максимальная длина поисковой строки составляет 256 байт. Переключатель "Формат данных" будет установлен в режим HEX или ASCII, в зависимости от области, в которой находится курсор.



При вводе в шестнадцатеричном формате данные форматируются автоматически. При переключении переключателя "Формат данных" данные для поиска или замены будут автоматически преобразовываться в нужный формат. При поиске строки в ASCII формате поиск может осуществляться с учетом регистра, в зависимости от флага "Поиск с учетом регистра". В зависимости от состояния флага "С позиции курсора" поиск начнется с начала файла или со следующего байта в позиции курсора. Для продолжения поиска используется клавиша **F3**.

Для замены данных необходимо заполнить обе строки. При нажатии кнопки "Заменить" заменяется первое найденное совпадение. Для следующей замены используется клавиша **F4**. При нажатии кнопки "Заменить все" заменяются все найденные совпадения, и выводится сообщение с указанием количества замен. Если размер искомой строки не совпадает с размером заменяемой, то размер файла будет изменен.

Переход (Ctrl+G) - выводит окно для ввода **адреса микросхемы**, на который необходимо переместить курсор. Если в редакторе установлен 16 битный формат данных, то при установке адреса, равного 0002h курсор будет перемещен на четвертый байт в буфере. А для 32 битного формата – на восьмой байт. Для микросхем NAND-Flash доступен специальный формат адреса: **2-FF**, который позволяет задать номер страницы и адрес байта внутри страницы.

Формат данных - позволяет выбрать 8, 16 или 32-битный формат представления данных. В момент открытия файла, формат данных устанавливается в зависимости от организации памяти выбранной микросхемы. В 16-битном формате четные и нечетные байты при выводе на экран меняются местами, образуя 16-битное слово. В 32-битном формате последовательность байт следующая: 3,2,1,0 - 7,6,5,4 ... и т.д. Эта команда используется только для отображения текста на экране и не модифицирует содержимое буфера.

Шрифт - выводит стандартный диалог для выбора шрифта. В редакторе используются только равноширинные шрифты. В Windows Vista, Win7 и Win8 не рекомендуется использовать шрифты, отличные от COURIER или FIXEDSYS. Выбранный шрифт будет использован как для отображения на экране, так и во время печати на принтере. Изменить размер и тип шрифта, который используется по умолчанию (при создании новых окон), можно в системном реестре, в ключе **"HKEY_CURRENT_USER/Software/Triton/Trisoft"**:

- "FontName" - Название шрифта, по умолчанию = 'Courier New'.
- "FontSize" - Размер шрифта, по умолчанию = 9.

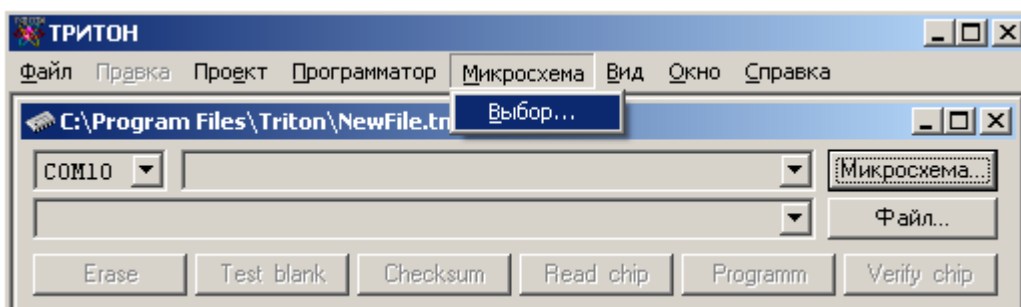
Набор символов - позволяет выбрать тип кодовой таблицы для отображения текста в редакторе. Используется только для отображения текста на экране. Не модифицирует содержимое буфера.

Меню Проект

Чтобы не выполнять при каждом запуске программы стандартные действия: выбор микросхемы, открытие файла, настройка параметров, и т.д., оболочка поддерживает работу с проектами. Проект – это совокупность микросхемы, файла прошивки и определенных настроек.

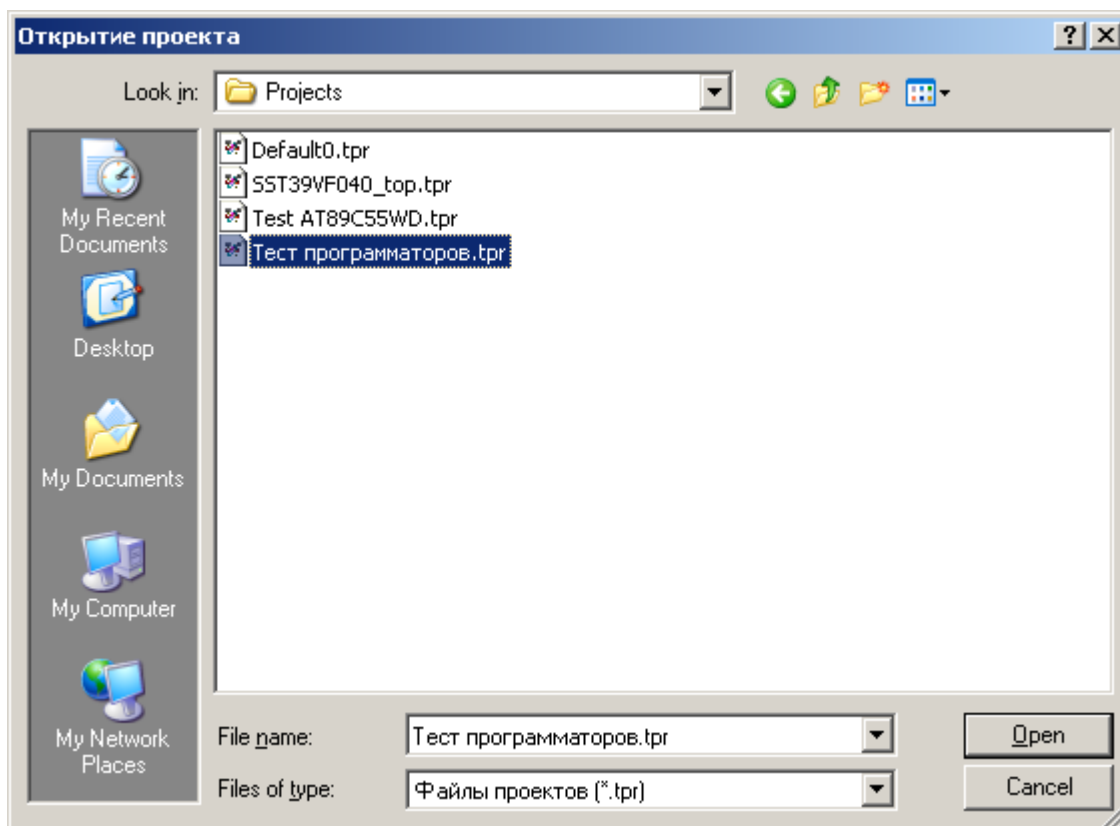
Каждое окно в программе - это проект, который может независимо от других проектов управлять своим программатором и хранить индивидуальные настройки. Одновременно может быть открыто любое число проектов. При наличии только одного программатора управление им может осуществляться последовательно из разных проектов.

Для создания нового проекта в главном меню программы кликните меню "Микросхема" - "Выбор...", [выберите нужную микросхему](#) и [откройте файл прошивки](#).



При необходимости можно настроить [режимы программирования микросхемы](#) (флаги, адреса, напряжения и т.д.), открыть файл EEPROM, разрешить доступ и настроить дополнительные параметры микросхемы. При работе в автономном режиме, желательно настроить [флаги режимов работы программатора](#). Если выполнялось чтение микросхемы или менялись данные в дампе, то сначала надо сохранить файл. После этого можно загрузить проект в программатор или сохранить его на компьютере.

Открыть проект - выводит стандартный диалог выбора проекта и открывает проект в новом окне.



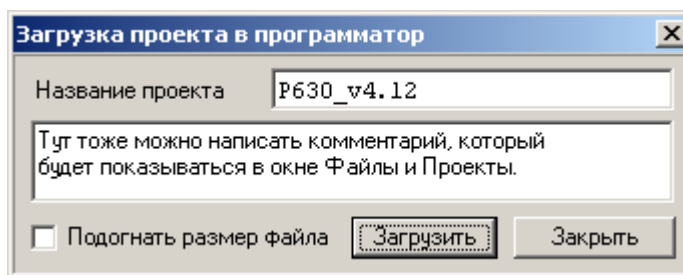
Если файлы проекта были перемещены или изменены, то программа предложит указать новый путь к каждому файлу проекта.

Если проект был создан в более ранней версии программы, то программа предложит обновить его до текущей версии.

Сохранить проект - выводит стандартный диалог сохранения файла и записывает проект на диск. В момент создания проекта в нем фиксируется дата и время создания и изменения проекта, и обнуляются счетчики микросхем. Если перед сохранением проекта проводилась запись микросхем, то данные о числе запрограммированных микросхем в текущей сессии, во вновь созданном проекте не сохраняются. Рекомендуется вначале создать и сохранить проект, а затем начинать запись микросхем.

Если в процессе работы в проект были изменены какие-либо настройки или режимы работы, то при закрытии проекта программа предложит сохранить текущие изменения.

Загрузить проект - выводит диалог для ввода имени проекта и загрузки его в программатор.



Длина имени проекта не должна быть больше 10 знаков. Русские буквы на экране программатора отображаются неправильно, поэтому их лучше не использовать. Если имя нового проекта совпадает с именем уже имеющегося в памяти проекта, то программатор выведет предупреждение. Необходимо или ввести другое имя или предварительно удалить старый проект.

Всего в энергонезависимой памяти программатора может быть сохранено до 256 проектов (по числу файлов), общим объемом до 4 Мбайт, плюс одна **оперативная конфигурация**, которая создается непосредственно на программаторе (кнопки 'Menu', 'Utl'+ 'Sav').

Для загрузки проекта в программатор необходимо выбрать микросхему и открыть файл, или выбрать уже готовый проект. При необходимости можно настроить любые параметры и включить нужные режимы работы. Также желательно настроить [флаги режимов работы программатора](#): "Стирать микросхему перед записью" и "При проверке перед записью не выводить ошибки".

Перед загрузкой программа проверяет размер свободного места, после чего загружает файл и все настройки в память программатора. Когда загружается файл проекта, программа последовательно вносит в него следующие изменения:

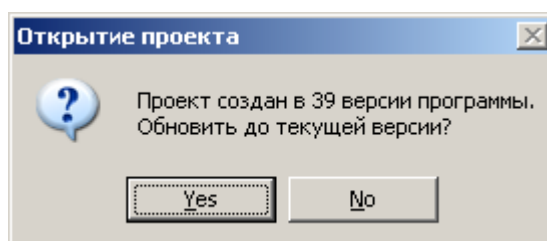
- Если значение адреса **Смещения** не равно нулю, то данные смещаются в начало файла. Соответственно, размер загружаемого файла будет уменьшен на величину адреса смещения.
- Если установлен флаг "Подогнать размер файла" и, если размер файла больше чем объем микросхемы (заданный начальным и конечным адресами), то загружена будет только часть файла, равная этому объему.
- Если разрешен доступ к EEPROM, LDR0M, OptionsROM..., то размер загружаемого файла будет установлен равным объему микросхемы (заданному начальным и конечным адресами), кратному 256 байтам для V5.4 или 512 байтам для V5.5, V5.6 и V5.7, и в конец файла будут записаны данные для этой ***ROM.

По окончании загрузки в программаторе будет выбрана только что загруженная конфигурация. Программатор готов к работе в автономном режиме. После этого, любая команда на работу с микросхемой (кроме программирования), переданная с компьютера, изменит в программаторе текущие значения флагов в выбранной конфигурации.

При работе в автономном режиме статистика запрограммированных микросхем по каждому проекту не ведется. Считается только общее количество микросхем, запрограммированных на программаторе. После завершения цикла записи и вывода результатов на дисплее будет показано текущее значение счетчика успешно запрограммированных микросхем. Включение питания программатора при нажатой кнопке 'Menu' обнулит значение этого счетчика.

Конвертор проектов

Для обеспечения совместимости с более ранними версиями в программу встроен конвертор проектов.



Конвертор активируется при открытии проекта и, при разрешении от оператора, вносит в этот проект необходимые изменения. Эти изменения будут подсвечены красным в [параметрах микросхемы](#). Поскольку изменения вносятся в уже открытый проект, это позволяет проверить работу с новой версией программы и сохранить их, или отказаться, при закрытии проекта. Можно отказаться от модификации проектов, но при этом работа с проектами будет невозможна, а при следующем запуске, программа опять запустит конвертор проектов.

Конвертор проектов обеспечивает корректное преобразование настроек только для работы с более новой версией программы.

При установке более ранних версий программы, преобразование проектов не выполняется. Также, конвертор проектов не сможет обновить проект, если данная микросхема отсутствует в списке микросхем в новой версии программы. В этих случаях проекты должны быть созданы заново.

Меню Программатор

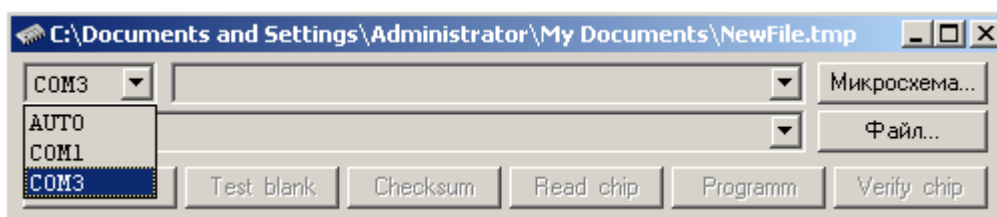
Меню "Программатор" содержит элементы:

- [Выбор COM порта](#) - выбор или смена порта и автоматический поиск программатора.
- [Обновление версии](#) - вывод окна загрузки и восстановления прошивки в программаторе.
- [Параметры и Тесты](#) - вывод окна настроек программы и тестов для проверки программатора.

Выбор COM порта

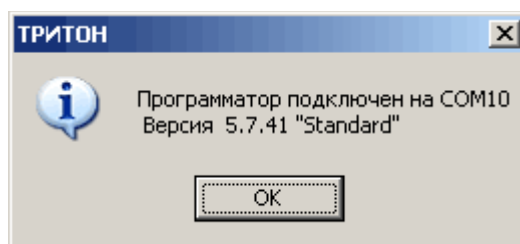
Выбор COM порта - позволяет выбрать номер коммуникационного порта или производит автоматический поиск подключенного программатора.

Это меню доступно только когда закрыты все окна проекта. При открытом окне проекта выбор порта производится с помощью выпадающего списка. При подключении нескольких программаторов это позволяет назначить каждому проекту свой программатор.



При запуске программа считывает из системного реестра список установленных на компьютере физических или виртуальных COM портов: `HKEY_LOCAL_MACHINE/Hardware/Devicemap/Serialcomm` и производит на них поиск программатора. Операционные системы Windows7, 8, 10 или 11, при включенном "Контроле Учетных Записей (UAC)", запрещают неподписанным программам доступ к этой ветке реестра, в результате чего, программа не видит подключенный программатор. Поэтому, перед первым запуском программы, необходимо отключить "Контроль Учетных Записей (UAC)" в профиле пользователя и перезагрузить компьютер. Если отключение данной функции невозможно, программа позволяет загрузить список портов, сформированных вручную. Для этого, запустите оболочку программатора, откажитесь от Демо-режима и закройте программу. Затем запустите программу **"Regedit.exe"**. В ключе `"HKEY_CURRENT_USER/Software/Triton/Trisoft"`, в параметре "CommList" введите список портов, куда может быть подключен программатор. Например: "AUTO","COM1","COM3","COM10". Номера портов можно посмотреть в "Диспетчере устройств", в разделе "Порты COM и LPT", подключая программаторы в разные USB порты. Сохраните настройки, закройте Системный реестр и запустите оболочку.

AUTO. При автоматическом поиске программатора программа обновляет список портов, затем последовательно перебирает доступные порты и на каждом выполняет следующие действия: проверяет занятость порта другими приложениями и, если порт свободен, настраивает его параметры и пытается установить связь с программатором. Если программатор обнаружен, данный порт становится активным, из программатора считывается служебная информация, после чего, выводится сообщение о номере COM порта и версии программатора.



Если программатор не найден, выводится сообщение об ошибке. Для дополнительной информации о возможных ошибках и методах их устранения смотрите раздел ["Если программатор не работает"](#).

Обновление программы

Для обновления программы, скачайте с сайта www.triton-prog.ru последнюю версию программного обеспечения TriSoft V5.8.xx, распакуйте файлы в любую папку на компьютере и выполните установку программного обеспечения.

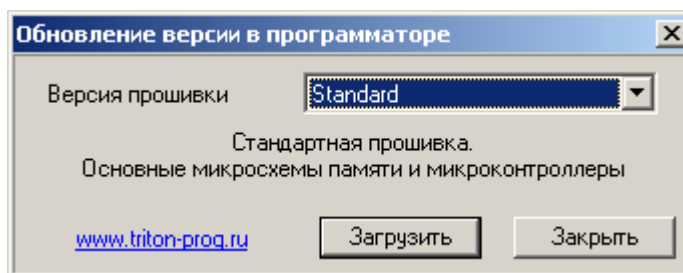
Не рекомендуется устанавливать более ранние версии программы (Software), чем версия, загруженная в программатор (Firmware). В первую очередь, это касается программаторов V5.8T, в которых обновляется не только прошивка процессора, но и прошивка в логической матрице. При установке более ранних версий, в них может не обновляться прошивка в матрице, что может привести к ошибкам в работе с микросхемами. Также, при откате на более ранние версии, в программе не работает конвертор проектов и их придется создавать заново.

Для программаторов версий V5.7T в комплект поставки входят два варианта (версии) прошивки встроенного процессора: "Standard" и "USB". Версия "Standard" работает через интерфейсы COM и USB. Версия "USB" обеспечивает более быструю работу со всеми микросхемами, но работает только через USB интерфейс. При подключении программатора через COM, версия "USB" позволяет только обновить прошивку в программаторе и обеспечивает доступ к спискам микросхем, файлам и проектам в памяти программатора.

Для программаторов V5.8T доступна только версия прошивки "USB". При обновлении этих моделей, программа сначала обновляет прошивку в процессоре, затем, при необходимости, обновляет прошивку в логической матрице.

Обновление версии - обеспечивает загрузку прошивки в программатор и автоматическое обновление списка микросхем.

При выборе этого меню программа пытается определить версию подключенного программатора и выводит окно для выбора и загрузки новой версии. В зависимости от версии программатора предлагаются различные варианты прошивок. Если программатор не был подключен в момент вывода этого окна, то предлагается загрузить версию "Standard". Из предложенного списка необходимо выбрать нужную версию прошивки и нажать кнопку "Загрузить".



Внимание: В процессе загрузки не выключайте питание программатора и не закрывайте программу. Обязательно дождитесь окончания загрузки.

Во избежание повреждения находящейся в памяти программатора информации, программа перед началом загрузки всегда проверяет версию программатора и считывает служебную информацию. В начале загрузки светодиод вспыхивает красным цветом и затем гаснет. Во время загрузки необходимо обеспечить бесперебойное питание программатора. Состояние процесса загрузки отображается на экране компьютера. Для программаторов V5.7T и V5.8T время обновления программы не превышает 4 секунд.

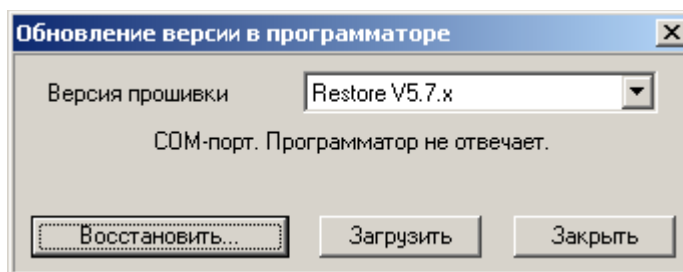
При обновлении версии в программаторах ТРИТОН+, данные в проектах не обновляются. Во избежании возможных проблем, после установки новой версии программы рекомендуется отформатировать память программатора, заново загрузить обновленные проекты и списки микросхем.

При обновлении версии также не меняются данные в оперативной конфигурации. Программа на компьютере не имеет непосредственного доступа к этой области и не может изменить находящиеся там данные. Для корректной работы этот проект также необходимо создать заново.

После завершения загрузки программатор перезапустится и загорится зеленый светодиод, сообщая о готовности к работе. При необходимости программа обновит списки микросхем, а на модели V5.8T - прошивку в логической матрице.

Если в процессе загрузки произошел сбой, в окне загрузки версии появится кнопка "Восстановить", которая обеспечит аварийную загрузку и сохранит настройки программатора. Если в этот момент закрыть программу, то настройки, хранящиеся в памяти программатора, будут потеряны.

Восстановление прошивки - обеспечивает восстановление настроек и прошивки в программаторе при сбоях во время обновления версии.



Если произошел сбой, то **НЕ ЗАКРЫВАЙТЕ ПРОГРАММУ** и четко выполните следующие инструкции:

- **Выключите** и снова **включите** питание программатора.
- Если на дисплее программатора написано "LOAD FIRMWARE", нажмите кнопку "Восстановить".
- По окончании загрузки программа сама восстановит все настройки программатора.

Если в процессе загрузки версии произошел сбой питания, и пришлось закрыть программу, то для восстановления прошивки необходимо выполнить следующие действия:

- Запустите управляющую программу и проверьте номер порта, куда подключен программатор.
- Выберите меню "Программатор/Обновление версии".
- При **выключенном** программаторе выберите версию прошивки и нажмите кнопку "Загрузить".

- Через некоторое время появится сообщение “COM-порт. Программатор не отвечает” и кнопка “Восстановить”.
- Включите питание программатора, удерживая кнопки "Vr" + "Menu", после чего нажмите кнопку “Восстановить”.
- После окончания загрузки, в меню "Программатор/Параметры и Тесты", “Калибровка”, необходимо отрегулировать напряжения питания и записи.

При первом запуске программа сохраняет в системном реестре значения калибровочных коэффициентов для формирователей напряжений, и каждый раз при выборе новой микросхемы проверяет их. Если коэффициенты, считанные из программатора, отличаются от ранее сохраненных более чем на 0,25v, то программа предложит восстановить заводские установки калибровочных коэффициентов.

Параметры и Тесты

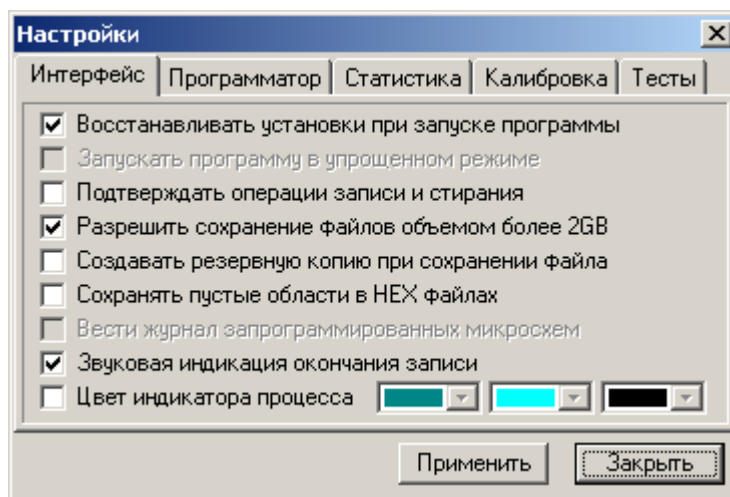
Диалог содержит пять закладок, на которых сгруппированы основные настройки и программы для проверки программатора:

- [Флаги интерфейса.](#)
- [Флаги программатора.](#)
- [Статистика.](#)
- [Калибровка программатора.](#)
- [Тесты программатора.](#)

При выборе этого меню программа проверяет наличие программатора и считывает служебную информацию. При подключении нескольких программаторов данные считываются с того программатора, окно управления которым является активным. Если программатор не подключен или занят работой с микросхемой, то некоторые функции диалога будут недоступны.

Интерфейс

Интерфейс - флаги, определяющие работу и оформление программы. Эти настройки общие для всех проектов.



Восстанавливать установки при запуске программы. При сброшенном флаге восстанавливаются только параметры главного окна программы, общие флаги и открывается новое окно проекта. При установленном флаге восстанавливается вся конфигурация, которая была в момент закрытия программы, включая все окна проектов, настройки и открытые файлы.

Существует два варианта запуска программы: непосредственный запуск программы и запуск через системные функции "Отправить" и "Открыть с помощью" (например, двойной клик по файлу проекта или при выборе контекстного меню "Отправить"). Программа сохраняет конфигурацию только при непосредственном запуске и при условии, что открыто хотя бы одно окно проекта, в котором выбрана микросхема.

Запускать программу в упрощенном режиме. При установленном флаге недоступны некоторые функции программы: работа с несколькими программаторами, расширенные настройки, скрипты, редактирование параметров и расширение списка микросхем. Предназначена для начинающих пользователей.

Подтверждать операции записи и стирания. Включение безопасного режима. Каждая операция, которая может изменить содержимое памяти микросхемы будет требовать дополнительного подтверждения. При установленном флаге недоступна работа с несколькими программаторами.

Разрешить сохранение файлов объемом более 2GB. При установленном флаге, при чтении микросхем, объемом 2 и более ГБайт, будет создан единый файл. Не рекомендуется устанавливать этот флаг для дисков с FAT32, так как максимальный размер файла не может превышать 4 ГБайта. При сброшенном флаге, программа создаст группу файлов с расширениями *.bin, *.bin1, *.bin2,... Размер этих файлов зависит от объема микросхемы и наличия свободной памяти, и не превышает 1 ГБайт.

Сохранять пустые области в HEX файлах. При установленном флаге, в шестнадцатеричных файлах сохраняются все данные из дампа. При сброшенном флаге, для уменьшения размера файла, программа записывает только те строки,

которые содержат данные. Пустые строки, содержимое которых равно \$FF или \$00, в зависимости от выбранной микросхемы, в файл не записываются.

Создавать резервную копию при сохранении файла. При установленном флаге, когда выбрана команда 'Сохранить', к исходному файлу добавляется расширение '.bak', (например, 'test.bin.bak'), а измененные данные записываются в новый файл под старым именем. При сохранении файлов в текстовых форматах (Intel HEX, Motorola, Tektronix...), чтобы открыть резервную копию, ее надо предварительно переименовать.

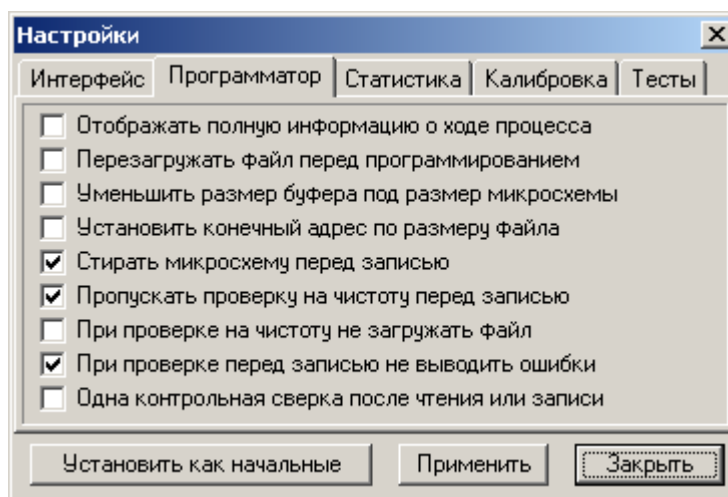
Вести журнал запрограммированных микросхем. Разрешает сохранять в проекте дополнительную детализированную информацию о сессиях проекта (дату и время открытия проекта, и количество запрограммированных микросхем).

Звуковая индикация окончания записи. Используется в режиме тиражирования при окончании записи микросхемы. Если микросхема запрограммирована без ошибок, выводится стандартный звук Windows 'Восклицание (Asterisk)'. При записи с ошибками выводится звук 'Критической ошибки (Critical Stop)'. Настроить звуковые события можно с помощью стандартных средств Windows: 'Control Panel / Sounds...'.
Цвет индикатора процесса. Позволяет выбрать цвет индикатора процесса при работе с микросхемой. Эти же цвета используются в окне загрузки файла в память программатора. Для смены цветов в окне обновления прошивки, а также для изменения цвета текста в индикаторе процесса необходимо в системном реестре изменить значение параметра "HKEY_CURRENT_USER/Software/Triton/Trisoft/Flags_Main". Значение по умолчанию \$B9E60038. Старший байт – конечный и начальный цвет в окнах обновления версии и загрузке теста. Второй байт – конечный и начальный цвет индикатора процесса при работе с микросхемой. Старший знак третьего байта – цвет текста в индикаторе процесса.

На этой же закладке можно установить скорость работы COM порта. При работе программатора через USB, этот параметр, как и другие настройки COM порта игнорируются программой.

Программатор

Программатор - флаги режимов работы программатора. Эти флаги можно настроить индивидуально для каждого проекта. При работе с несколькими проектами, в окне показывается и меняются состояния флагов активного проекта.



Отображать полную информацию о ходе процесса. Когда флаг установлен, на закладке "Сообщения" выводится более детальная информация о ходе процесса. При обращении в техподдержку, анализ этой информации позволяет смоделировать возникшую ситуацию, определить причину ошибки и дать соответствующие рекомендации.

Перезагружать файл перед программированием. Все внесенные изменения будут утеряны. Если файл для записи в микросхему создается в результате работы внешнего компилятора, то установка этого флага гарантирует запись в микросхему самой последней версии файла. В момент загрузки файла, данные о дополнительных режимах работы (например, конфигурационное слово) обновляются только в том случае, если доступ к ним разрешен флагами, установленными на панели управления. Состояние этого флага отображается в строке статуса программы (поле 'Reopen').

Уменьшить размер буфера под размер микросхемы. При открытии файлов большого объема, считывает не весь файл, а только часть, равную объему выбранной микросхемы. Если размер микросхемы превышает размер файла, то считывается весь файл, а оставшаяся часть при программировании микросхемы будет заполнена кодом \$FF.

Установить конечный адрес по размеру файла. Установка данного флага позволяет автоматически устанавливать (уменьшать) конечный адрес микросхемы в момент открытия файла. При перезагрузке файла перед программированием конечный адрес будет устанавливаться в соответствии с размером файла. Если размер файла превышает размер микросхемы, то изменение адреса не производится. Состояние этого флага отображается в строке статуса программы (поле 'SetAdr'). Данная функция учитывает организацию микросхем со страничной записью и конечный адрес увеличивается до конца страницы. При частичном стирании таких микросхем необходимо учитывать, что размер блока для стирания может в несколько раз превышать размер страницы для записи.

Стирать микросхему перед записью. Позволяет уменьшить общее время программирования. Во время стирания программатор контролирует флаг готовности и флаг ошибки. Если во время стирания ошибок не было, то проверка на

чистоту не выполняется, а сразу начинается запись микросхемы. Если в микросхеме такого флага нет, или он был установлен, то программатор выполняет проверку на чистоту. Файл при этом не загружается. Если флаг сброшен, то микросхема сначала проверяется на чистоту или возможность записи, после чего начинается программирование. Если запись невозможна, программа предложит стереть микросхему.

Пропускать проверку на чистоту перед записью. В режиме программирования пропускает проверку микросхемы на чистоту. Установка этого флага отключает проверку для всех вновь выбираемых микросхем. Включить или отключить этот режим для выбранной микросхемы можно на закладке "Параметры". Состояние этого флага отображается в строке статуса программы (поле "NoTest").

При проверке на чистоту не загружать файл. Когда флаг установлен, микросхема проверяется только на чистоту, а не на возможность записи. Позволяет сократить общее время программирования. Обычно флаг устанавливается при программировании новых микросхем или для микросхем с электрическим стиранием.

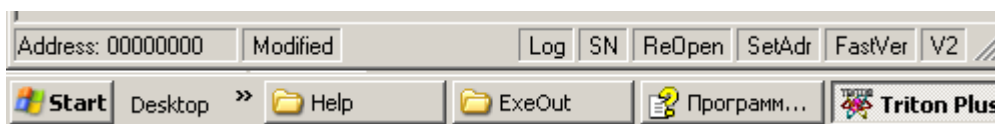
При проверке перед записью не выводить ошибки. Когда флаг установлен, то при проверке микросхемы перед программированием, сообщение об ошибках будет выводиться только в случае, когда запись микросхемы будет невозможна. Если флаг сброшен, то будут выведены все ошибки, и перед программированием, будет выдан запрос на подтверждение начала записи.

Для микросхем в которых нет режима стирания, установка данного флага пропускает проверку на чистоту и сразу переводит программатор в режим записи.

Одна контрольная сверка после чтения или записи. При установленном флаге, независимо от напряжений Vcsm1n и Vcsm1x, выполняет только одну контрольную сверку: после чтения микросхемы или при сверке – при номинальном напряжении питания (Vcsm0n), а после записи – при максимальном напряжении питания (Vcsm1x). Состояние этого флага отображается в строке статуса программы (поле 'V1/V2'). В автономном режиме этот флаг не работает.

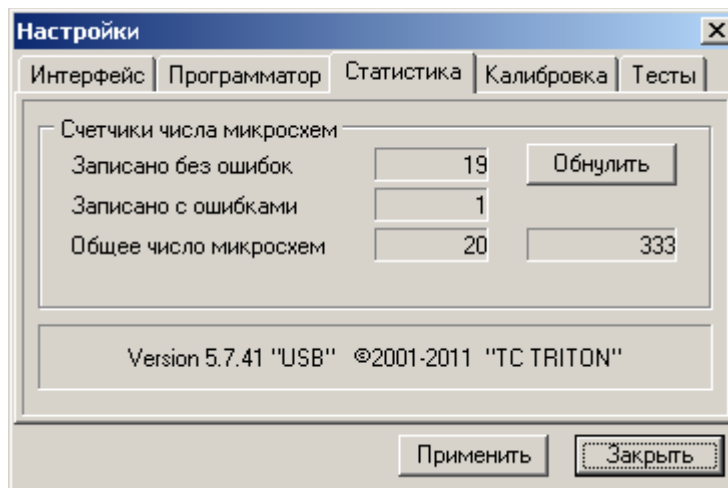
Установить как начальные. Устанавливает выбранные значения флагов программатора и рабочую папку активного проекта в качестве начальных, при создании новых проектов.

Некоторые флаги, влияющие на записываемые в микросхему данные, продублированы в строке состояния и могут быть оперативно изменены двойным кликом мыши.



Статистика

Статистика - общая статистика запрограммированных микросхем.



На этой вкладке находятся счетчики числа микросхем и кнопка для сброса показаний счетчиков. Счетчики реализованы во внутренней памяти программатора и ведут подсчет числа запрограммированных микросхем.

Счетчики в программаторе не инкрементируются, если:

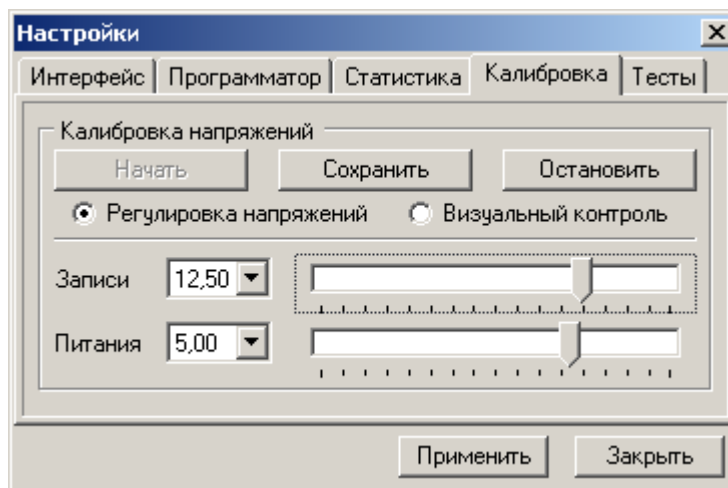
- микросхема имеет дефекты кристалла;
- сработала защита программатора от перегрузок;
- возник сбой во время передачи данных;
- команда отменена пользователем.

Калибровка

Калибровка - проверка ЦАПов и калибровка напряжений. Все программаторы поставляются в откалиброванном и проверенном состоянии. **Менять эти настройки НЕ РЕКОМЕНДУЕТСЯ.** Программное обеспечение постоянно следит за

значениями калибровочных коэффициентов и, в случае сбоя, позволяет восстановить их в заводское состояние.

Внимание: Перед началом теста удалите микросхему из панельки программатора!



Перед началом теста проверьте отсутствие микросхемы в панельке программатора и нажмите кнопку "Начать". Загорится красный светодиод, и на выводах панельки появятся указанные напряжения. По окончании теста нажмите кнопку "Сохранить" для записи настроек в программатор, или кнопку "Остановить" для прекращения теста.

Тест **Регулировка напряжений** позволяет проверить работу умножителей напряжения, цифро-аналоговых преобразователей, схему компенсации напряжения, а также, при необходимости, откалибровать напряжения питания и программирования. Калибровку рекомендуется выполнять при указанных напряжениях питания и записи, и проверять правильность установки во всем диапазоне рабочих напряжений. Все измерения должны проводиться цифровым вольтметром с классом точности не ниже 0,5. Напряжение программирования измеряется между 20(GND) и 17 (или 11) выводами панельки на резисторе 1кОм. Напряжение питания измеряется между 21(GND) и 24 (или 30) выводами панельки на резисторе 1кОм. Из выпадающего списка выбирается нужное значение напряжения и подстраивается с помощью движков. Перемещение между элементами управления осуществляется клавишами "Tab" и "Shift+Tab". Изменение и регулировку напряжений лучше проводить клавишами управления курсором. Для грубой регулировки используются клавиши "PageUp", "PageDown". Сохранение калибровочных значений производится только при установке напряжения записи и питания равными 12,5v и 5,5v.

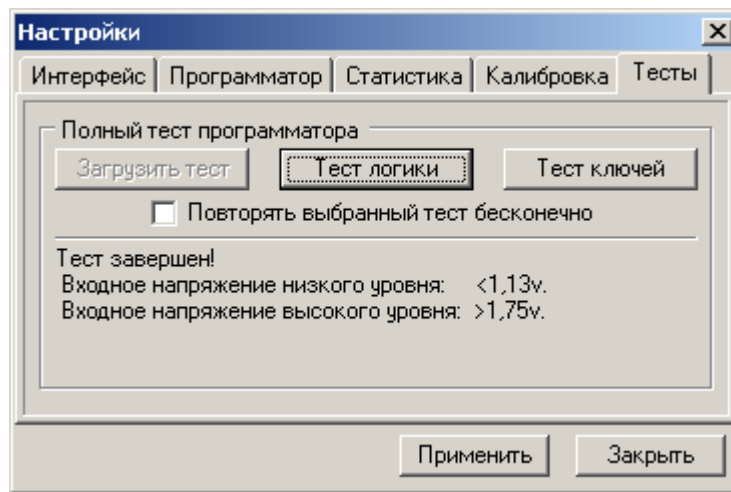
Программатор обеспечивает диапазон установки напряжения питания от 2v до 6,75v и напряжения программирования от 2v до 16v с шагом 0,25v (при установленном флаге "Повышенная точность установки напряжений" - шаг = 0,125v). Правильно откалиброванный программатор обеспечивает точность установки и стабилизацию напряжений питания и программирования не хуже +/-100mv во всем диапазоне напряжений, при токах нагрузки 0-80mA(для Vpp) и 5-80mA(для Vcc).

Тест **Визуальный контроль** позволяет с помощью осциллографа визуально проверить работу формирователей фронтов и логических уровней. В этом режиме на всех выводах панельки формируются прямоугольные импульсы с амплитудой, указанной в выпадающем списке "Питание". Период импульсов - около 10mks.

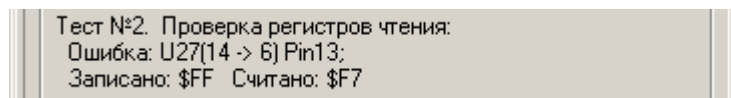
Тесты

Тесты - позволяют самостоятельно выполнить проверку программатора и в случае ошибки, определить неисправный вывод и поврежденный элемент на плате. Успешное завершение этих тестов и правильная калибровка гарантируют 100% исправность программатора.

Внимание: Перед началом теста удалите микросхему из панельки программатора!



Всего для проверки программатора используются два теста: "Тест логики" и "Тест ключей". Тест логики проверяет работу всего программатора и позволяет выявить до 90% всех неисправностей. Тест ключей дополнительно проверяет работу ключей под нагрузкой. Тесты могут быть выполнены однократно или повторяться циклически. Для непрерывного повтора необходимо нажать кнопку нужного теста и затем установить флаг "Повторять выбранный тест бесконечно". Каждый тест включает в себя несколько последовательных проверок. При возникновении ошибки работа теста прерывается и выводится сообщение об ошибке, которое включает: номер и название теста, номер цикла, номера выводов панельки и маркировку микросхем или ключей, которые вызвали ошибку.



Все тесты самонастраиваемые. В начале каждого цикла сначала определяется порог срабатывания, после чего начинается сам тест. Чувствительность тестов очень высока. Даже касание рукой контактов панельки во время теста может вызвать ошибку. В связи с этим допускаются нерегулярные сбои тестов, которые не оказывают влияния на качество записи микросхем.

Тест логики

В ходе каждого цикла сначала проверяется шина данных процессора, далее последовательно проверяются регистры чтения и записи, работа цифро-аналоговых преобразователей, затем измеряются напряжения логических уровней и в конце, проводятся тесты "Бегущий ноль" и "Бегущая единица". Погрешность измерений составляет 0,125mv. При успешном окончании теста выводится сообщение "Тест Ок" и предельные значения логических уровней. Эти напряжения должны быть в пределах 1,25-1,75v при откалиброванных напряжениях питания и программирования.

Тест ключей

Для работы этого теста необходимо в панельку программатора установить специальную заглушку. В этой заглушке все выводы панельки подключаются через резисторы ~100ом к земле (например, корпус разъема COM порта).

В этом тесте сначала проверяется наличие заглушки, затем тестируются ключи питания и программирования, измеряется максимальное время срабатывания, и в конце, проводится проверка на замыкания. При успешном окончании теста выводится сообщение "Тест Ок" и максимальные значения времени срабатывания. Как правило, это время не должно превышать ~6 mks.

Меню Микросхема

Это меню содержит команды, которые необходимы для работы с микросхемой. В основном тут дублируются кнопки, находящиеся в окне проекта.

- [Выбор...](#) – создает новый проект и открывает окно для выбора микросхемы.
- [Серийный номер](#) – показывает диалог для установки параметров серийных номеров.
- [Управление блоками](#) – показывает диалог для выбора алгоритма обработки дефектных блоков.
- [Коррекция ошибок](#) – показывает диалог для установки параметров ECC для NAND-Flash.

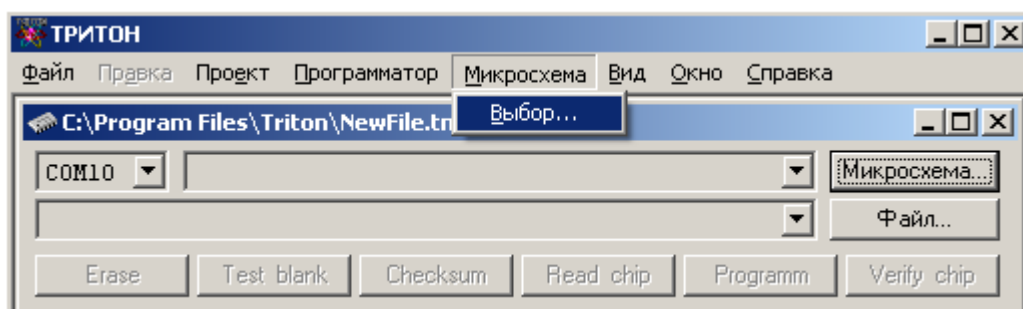
Команды, непосредственной работы с микросхемой.

- [ERASE](#) – электрическое стирание микросхемы.
- [PROGRAMM](#) – программирование микросхемы.
- [TEST BLANK](#) – проверка на чистоту и возможность записи.
- [CHECKSUM](#) – подсчет контрольной суммы микросхемы.
- [READ CHIP](#) – чтение микросхемы.
- [VERIFY CHIP](#) – сверка микросхемы и с содержимым буфера.
- [EEPROM](#) – команды для работы со встроенной памятью данных.

Выполнение любой команды (кроме Стирания) можно безопасно прервать в любой момент. Для этого, справа от прогресс-бара есть поле счетчика. Если кликнуть по нему во время работы с микросхемой - прервется цикл, если в режиме ожидания, то обнулится сам счетчик.

Выбор...

Сразу после запуска программы, доступна только команда "Выбор". Эта команда создает [новый проект](#), используя стандартные установки и открывает окно для [выбора микросхемы](#). После выбора микросхемы будут доступны кнопки режимов работы и все элементы окна. Внешний вид окна и набор команд может меняться в зависимости от типа микросхемы.



Серийный номер

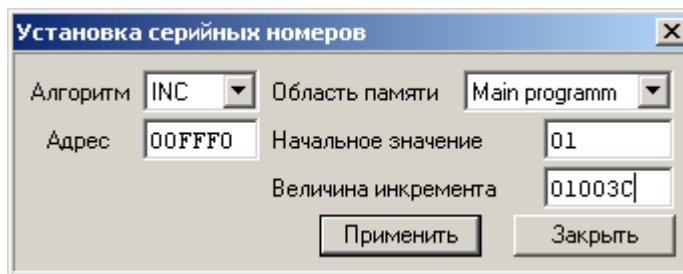
Программное обеспечение позволяет непосредственно перед записью микросхемы изменить в исходном файле, в EEPROM или в параметрах микросхемы один или несколько байт, по заданному алгоритму. Это может быть увеличение или уменьшение заданного номера на какую-то величину, произвольная функция, описанная в файле калькулятора или набор значений, считанный из текстового файла.

Когда разрешена запись серийных номеров, состояние этого флага показывается в строке состояния. Для предотвращения случайной записи это режим отключается при выборе новой микросхемы. Параметры серийного номера и флаг разрешения записи сохраняются только вместе с проектом или при установленном флаге "[Восстанавливать установки при запуске программы](#)".

Серийный номер загружается в буфер непосредственно перед записью микросхемы, и меняется только при успешном завершении всего цикла. Если в процессе записи микросхемы возникли ошибки, то при записи следующей микросхемы серийный номер не изменится. Такой механизм позволяет между операциями записи выполнять любые другие команды, например, дополнительную верификацию.

Запись серийных номеров в автономном режиме не поддерживается.

В программе реализовано три алгоритма установки серийных номеров: "INC", "FILE" и "CALC", которые могут менять данные в одном из трех дампов: в основной прошивке (Main programm), в области EEPROM data или в параметрах микросхемы (Chip Parametrs).



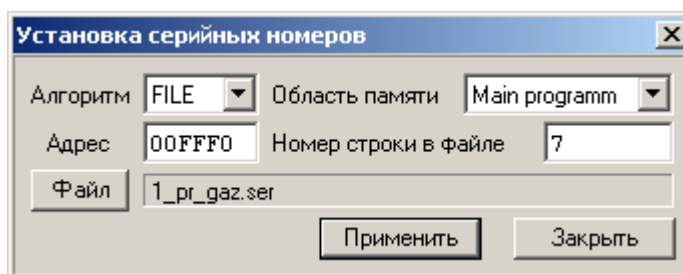
Алгоритм INC – серийный номер имеет фиксированную длину (от одного до четырех байт) и увеличивается на конкретное значение.

Начальный адрес – может иметь любое значение. Если установленный адрес превышает размер буфера, то последний будет увеличен до нужного размера и заполнен кодом FF. Исключение составляет буфер с параметрами микросхемы. Там адрес не должен превышать \$17F.

Величина инкремента – может иметь любое значение. Если значение инкремента будет равно \$FFFFFFFF, то начнется обратный отсчет.

Начальное значение – устанавливает начальное значение серийного номера. Длина серийного номера (количество байт) определяется исходя из длины начального значения (по числу знаков). Если начальное значение равно 1 или 01 (один или два знака), то длина серийного номера будет равна 1 байту, если 001 или 0001 (три или четыре знака), то 2 байтам и т.д.

Например, адрес равен \$00FFFF0, начальное значение равно \$01003C, инкремент равен 01. В первую микросхему, начиная с адреса \$00FFFF0, будут записаны три байта \$3C, \$00, \$01. Во вторую – \$3D, \$00, \$01. В третью – \$3E, \$00, \$01 и т.д. Если значение инкремента будет равно \$FFFFFF, то начнется обратный отсчет (\$3C, \$3B, \$3A...).



Алгоритм FILE – перед каждой записью значение серийного номера считывается из файла. Длина каждого номера вычисляется исходя из длины считанной строки. Комментарии и пустые строки игнорируются. При достижении конца файла запись будет остановлена и выведено сообщение: "Серийный номер. Достигнут конец файла".

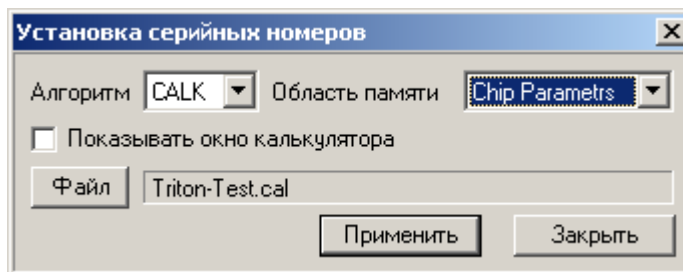
Пример файла серийных номеров. В файле допускаются заглавные и строчные буквы, длина строки – до 8 знаков (4 байта). Строка комментариев должна начинаться с ";".

```

;=====
; Файл серийных номеров
;=====
00
1
2
1e
;это комментарий, следующая - пустая строка.
1f
6754
55
56
ffEE6857
59
5A
5B
;=====

```

В этом примере в ячейку \$00FFFF0, начиная с 7 строки, будут последовательно загружены значения: 1E, 1F, 54 (в \$00FFF1 – 67), далее в \$00FFFF0 – 55, 56, 57 (в \$00FFF1 – 68, в \$00FFF2 – EE, в \$00FFF3 – FF), и снова в \$00FFFF0 – 59, 5A, 5B. При очередной записи содержимое ячеек не изменится, а будет выведено сообщение о конце файла.



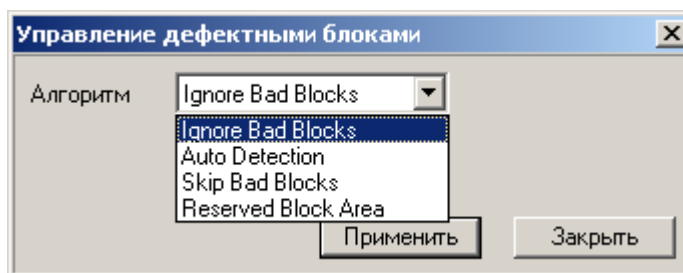
Алгоритм CALK – перед каждой записью значение серийного номера вычисляется по алгоритму, заданному в файле [калькулятора](#). Программа может показать диалоговое окно, в котором пользователь может ввести нужные значения или выполнить находящийся в файле скрипт в фоновом режиме.

Управление блоками

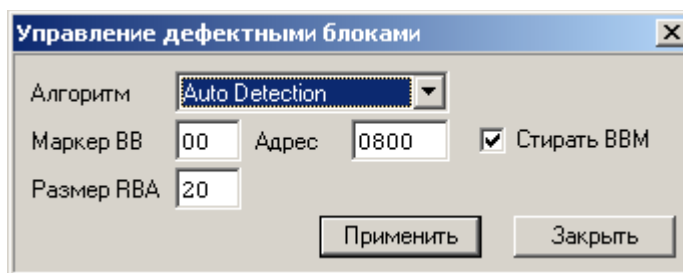
Это меню доступно только для микросхем NAND-Flash при подключенном программаторе ТРИТОН+ V5.7T или V5.8T.

Во многих, даже новых, микросхемах NAND-Flash могут присутствовать нерабочие области, именуемые дефектными (плохими) блоками. Кроме того, такие области могут появляться в микросхеме во время работы, в результате износа или сбоя. Поэтому в микросхемах NAND-Flash используются специальные алгоритмы обработки дефектных блоков.

Программное обеспечение в процессе работы с микросхемами NAND-Flash позволяет управлять дефектными блоками: пропускать их или подменять из резервной области. При записи микросхем, программа может изменять дампы, перенося данные, попадающие в дефектный блок, в резервную область, обеспечивая при этом автоматический перерасчет ECC в измененных блоках. Все операции производятся непосредственно во время работы с микросхемой, так называемая "подготовка файла" не требуется.



Алгоритм – позволяет выбрать алгоритм обработки дефектных блоков. В режиме **Ignore Bad Blocks** программа читает и пишет всю микросхему целиком, игнорируя маркеры дефектных блоков. [Коррекция ошибок](#) (ECC) при этом может быть включена или выключена. В режиме **Auto Detection** программа проверяет содержимое микросхемы и файла, определяет наличие дефектных блоков и сама выбирает нужный алгоритм обработки. В этом режиме ECC должен быть обязательно включен. Режимы **Skip Bad Blocks** и **Reserved Block Area (RBA)** позволяют вручную установить нужный алгоритм и настроить параметры, но программа все равно проведет анализ дампа и выдаст сообщение об ошибке, если данные не соответствуют выбранному алгоритму. Программа не позволит записать микросхему в режиме пропуска плохих блоков, если в файле используется режим замены. И наоборот, нельзя использовать режим замены плохих блоков, если файл не содержит эти области.



Маркер BB - задает значение маркера дефектного блока. В большинстве микросхем, рабочий блок маркируется как \$FF, любое другое значение обозначает дефектный блок. В некоторых случаях, рабочий блок может маркироваться кодом \$F0. Если в программе значение маркера = \$FF, то любой блок в микросхеме, имеющий другое значение, будет считаться дефектным. Если значение маркера = \$F0, то блоки, маркированные кодом \$FF и \$F0 будут считаться рабочими, все остальные - дефектными.

Адрес - задает адрес размещения маркера дефектного блока, начиная с начала страницы. В большинстве случаев, согласно документации, для микросхем с объемом страницы 2112 байт или больше, для этого используется нулевой байт в дополнительной (спаре) области, в нулевой или первой странице плохого блока. Для микросхем с объемом страницы 528 байт, как правило, это 5 байт в спаре области. Для 16-битных микросхем с размером страницы 264 слова, это нулевой байт в спаре области. Но во многих реальных устройствах, адрес размещения маркера дефектных блоков зависит от организации логических страниц внутри сектора и типа применяемой ECC. Поэтому для корректной работы с микросхемой должен быть включен автоматический режим [коррекции ошибок](#). При этом, когда программа определяет алгоритм ECC, она

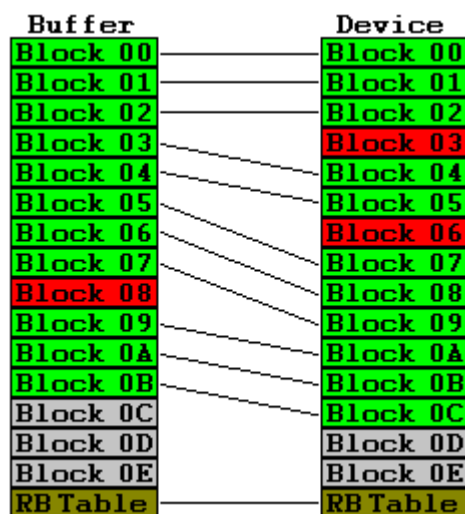
устанавливает и адрес маркера дефектного блока.

Стирать BVM. При установленном флаге, в режиме стирания микросхемы программа стирает маркеры дефектных блоков. В микросхеме может быть два типа плохих блоков. Первый тип это аппаратный или заводской. При стирании такого блока, он не стирается, и микросхема всегда выдает сообщение об ошибке. Второй - программный. Его создает устройство при сбоях или ошибках во время работы. Установка этого флага позволяет очистить микросхему от таких блоков.

Размер RBA - задает максимальное количество блоков в резервной области. Во многих микросхемах последние несколько блоков зарезервированы для хранения служебной информации. Это могут быть таблицы дефектных блоков или подменные блоки в алгоритме RBA. В режиме пропуска плохих блоков, когда происходит сдвиг данных, установка этого параметра обеспечивает запись этих блоков по тем же адресам, где они находились в оригинальной микросхеме. Параметр устанавливается автоматически, в зависимости от объема выбранной микросхемы.

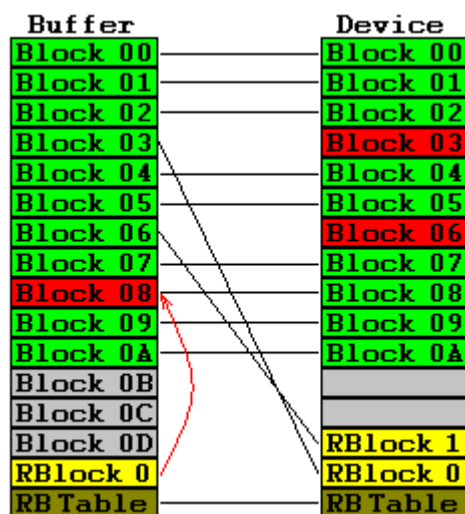
Skip Bad Blocks - пропуск плохих блоков. Во время работы с микросхемой программа анализирует содержимое файла и маркеры плохих блоков в микросхеме, и пропускает дефектные блоки. Таблицы рабочих и дефектных блоков, при их наличии, определяются и обрабатываются автоматически.

Последние блоки, количество которых задано в параметре **Размер RBA**, записываются в конец микросхемы без сдвигов. Соответственно, наличие плохих блоков в этой области недопустимо. Если дефектный блок попадает в эту область, то необходимо уменьшить размер RBA, так чтобы дефектный блок оказался за пределами резервной области.



Reserved Block Area - замена блоков из резервной области. Этот режим может быть выбран только автоматически, режиме Auto Detection. Программа поддерживает несколько вариантов, используемых в реальных устройствах, например, в телевизорах SAMSUNG и LG.

Перед началом работы программа анализирует файл и, если он содержит плохие блоки, то переносит эти блоки из резервной области в их реальное место. Если во время работы в микросхеме будут обнаружены дефектные блоки, то содержимое этих блоков будет перенесено в резервную область. При этом программа автоматически вносит изменения в таблицу плохих блоков и выполняет перерасчет ECC.



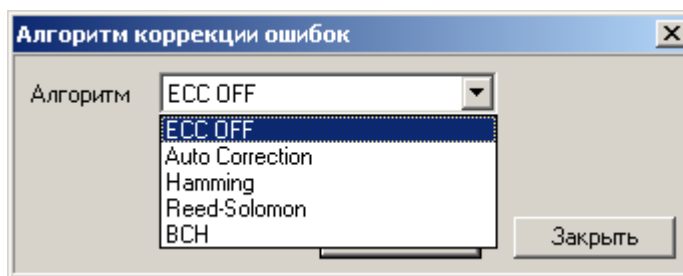
Адрес размещения резервной области, тип и адрес таблицы резервных блоков, устанавливаются автоматически, в зависимости от выбранного файла и микросхемы.

Коррекция ошибок

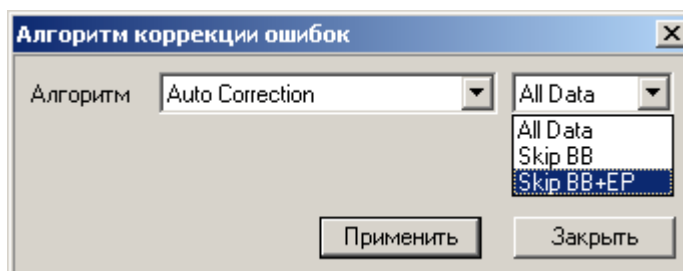
Это меню доступно только для микросхем NAND-Flash при подключенном программаторе ТРИТОН+ V5.7Т или V5.8Т.

Технология, по которой изготовлены микросхемы NAND-Flash, не может гарантировать отсутствие ошибок в процессе работы с микросхемой. Для исправления таких ошибок используются корректирующие коды. Обычно, в микросхемах NAND-Flash используются коды Хемминга (Hamming), Рида-Соломона (Reed-Solomon) и BCH (BCH). Код Хемминга, самый простой, позволяет исправлять одну битовую ошибку в блоке 256 или 512 байт. Код Рида-Соломона корректирует до 4 поврежденных байт в 512-байтном блоке. Коды BCH позволяют исправить до 64 ошибок в блоках от 256 до 4096 байт. Недостатком кода Хемминга является невозможность коррекции ошибки за пределами информационного блока, например в самом корректирующем коде, тогда как коды Рида-Соломона и BCH исправляют ошибки в любом месте страницы.

Программное обеспечение позволяет в процессе работы с микросхемой корректировать данные, принимаемые из программатора или загружаемые в него. При чтении микросхемы, программа вычисляет корректирующий код страницы и сравнивает его с кодом, считанным из микросхемы. Если коды отличаются, программа вычисляет адрес ошибки и исправляет ее. Это позволяет исправлять ошибки, которые могут возникать в микросхемах NAND-Flash, до того, как эти данные будут обработаны основной программой. При записи, программа проверяет записываемые данные, автоматически исправляя найденные ошибки, что позволяет писать в микросхему любой файл, без предварительной подготовки.

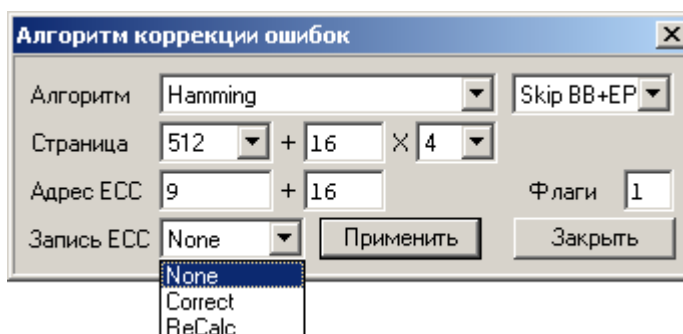


Алгоритм – позволяет выбрать алгоритм коррекции ошибок. Программа поддерживает основные коды коррекции ошибок: Хемминга, BCH и Рида-Соломона, а также предлагает режим Автокоррекции, для тех, кто не понимает о чем идет речь. В этом режиме программа пытается определить тип ECC, используемый в микросхеме, на основе имеющейся базы данных. Если алгоритм ECC определен, то программа выполняет чтение и запись микросхемы с проверкой и коррекцией ошибок. Если микросхема не содержит ECC или использует неизвестный алгоритм, или данные сильно повреждены, то программа выключает режим коррекции, и читает или пишет данные "как есть".



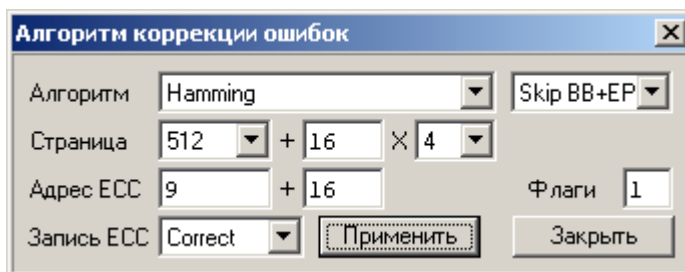
В режиме Автокоррекции пользователю доступна только одна опция, позволяющая выбрать, как программа будет обрабатывать массив данных: весь файл (All Data), с пропуском дефектных блоков (Skip BB) или с пропуском дефектных блоков и пустых страниц (Skip BB + EP). Необходимо отметить, что размещение маркера дефектного блока не всегда совпадает с тем, что написано в документации. Это касается не только сторонних разработчиков, но также и самих фирм, которые придумали эту спецификацию. В любом случае, алгоритмы, работающие в программе, достаточно умные и действуют по принципу, что лучше пропустить ошибку, чем повредить данные.

В режиме Автокоррекции действуют следующие правила. При **чтении** микросхемы, программа проверяет и исправляет выявленные ошибки в считанном блоке данных, после чего загружает его в буфер или сохраняет в файл. При **записи** микросхемы, программа сначала проверяет и исправляет выявленные ошибки в блоке данных, после чего передает его в программатор. Это позволяет исправлять ошибки в файлах считанных на других программаторах.

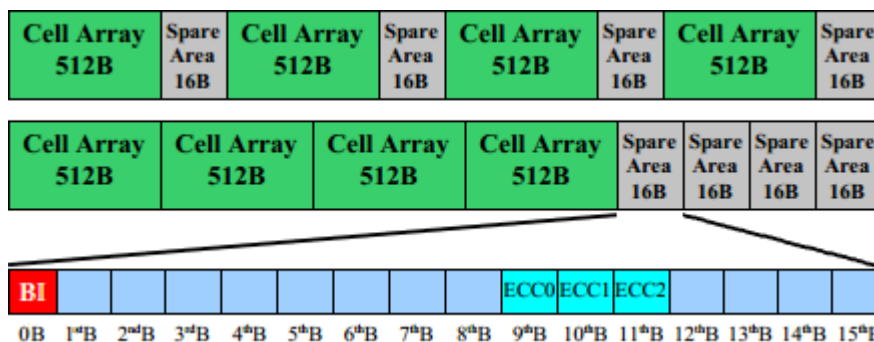


В остальных режимах пользователь может выбрать порядок обработки ECC при записи микросхемы, в параметре **Запись ECC**. Режим **"None"** - выбранный файл или данные из буфера пишутся в микросхему "как есть" без изменений. В режиме **"Correct"**, также как при Автокоррекции, программа проверяет и исправляет ошибки перед записью данных в микросхему. В режиме **"ReCalc"** программа производит перерасчет ECC, в соответствии с выбранными настройками, после чего передает данные в программатор для записи микросхемы. Использовать этот режим нужно очень осторожно! Надо четко знать какой алгоритм и какие параметры ECC используются в устройстве, в противном случае устройство работать не будет.

При ручной настройке алгоритмов необходимо установить ВСЕ параметры, представленные в диалоге. Для этого считайте несколько блоков из микросхемы и посмотрите структуру данных в редакторе. Зная размер страницы и размер Spare области в микросхеме, посмотрев несколько непустых страниц можно определить, сколько байт отведено под ECC, по каким адресам они расположены и на каком расстоянии друг от друга. Количество байт ECC четко связано с размером страницы, типом алгоритма и количеством корректируемых ошибок.



Страница – задает размер логической страницы 256, 512, 1024 или 2048 байт, для которой будет рассчитываться ECC. Далее указывается размер Spare области, для каждой логической страницы. И третий параметр, это число логических страниц в физическом секторе микросхемы. Например, для микросхемы NAND с размером страницы равным 2112 байт (2048 + 64), наиболее часто используют следующие схемы разметки: **512 + 16 X 1** для первого и **512 + 16 X 4** для второго рисунка.



```

000007D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000007E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000007F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000800: FF FF FF FF FF FF FF FF FF 3C FF 3F FF FF FF FF яяяяяяяяяя?яяяяяя
00000810: FF FF FF FF FF FF FF FF FF C0 CC FF FF FF FF FF яяяяяяяяяяАМяяяяяя
00000820: FF FF FF FF FF FF FF FF FF A9 96 69 FF FF FF FF яяяяяяяяяя@-ияяяяя
00000830: FF FF FF FF FF FF FF FF FF 3C 0F F0 FF FF FF FF яяяяяяяяяя>яяяяяя
00000840: 03 00 11 24 02 00 12 3C 60 83 13 3C 00 08 73 26 ...$.<`т.<...e&
00000850: 21 20 20 02 00 0C 05 24 B8 00 D8 0C 00 00 00 00 ! ....$»..Ш.....

```

Адрес ECC задает начальный адрес первого байта ECC внутри логической Spare области. В данном примере, это девятый адрес. Плюс 16 байт - это **Инкремент ECC**, который задает смещение для следующего кода ECC. Для алгоритмов БЧХ и Рида-Соломона можно указать количество корректируемых ошибок и полином. Параметр **Флаги** определяет алгоритм обработки данных и порядок вывода ECC в Spare области. Прежде чем ввести новые данные в параметры микросхемы, программа проверяет их корректность и при необходимости выводит сообщение об ошибке.

Команда ERASE

Доступно только для микросхем имеющих режим электрического стирания. Алгоритм стирания выбирается автоматически, в зависимости от типа выбранной микросхемы, и не может быть изменен. Если микросхема имеет несколько различных алгоритмов стирания, то выбирается наиболее быстрый режим или обеспечивающий более полное стирание.

При установленном флаге **"Подтверждать операции записи и стирания"**, перед стиранием выводится запрос на подтверждение режима, и затем, производится стирание микросхемы. После стирания производится проверка микросхемы на чистоту (без загрузки файла) при минимальном напряжении питания и выводится сообщение о результатах работы. Для некоторых типов микросхем во время стирания программатор контролирует содержимое микросхемы. В этом случае последующая проверка микросхемы на чистоту не выполняется.

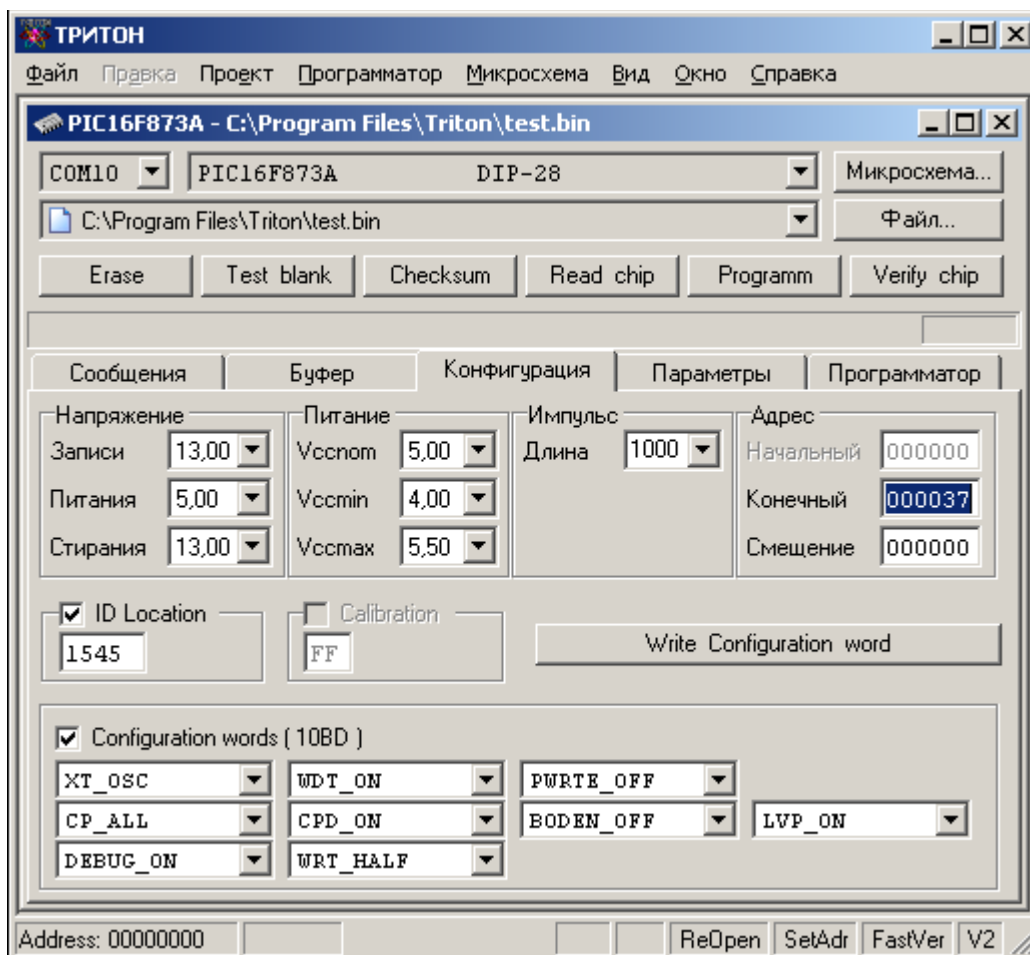
Время стирания устанавливается согласно фирменной спецификации и для некоторых микросхем может достигать 600

секунд.

Некоторые микросхемы, в основном это PIC контроллеры, во время стирания сохраняют содержимое EEPROM, или имеют в конфигурационном слове неиспользуемые ячейки, которые читаются как нули. В результате, после стирания такой микросхемы, при последующей проверке будет выведено сообщение об ошибке. Можно игнорировать такие ошибки, нажав кнопку "Пропустить" или снять флаг, разрешающий доступ к этой области. Для более подробной информации об особенностях стирания конкретных типов микросхем смотрите раздел "[Алгоритмы работы](#)".

Команда PROGRAMM

Перед записью, на закладке "Конфигурация" проверьте и, при необходимости, настройте конфигурацию микросхемы (Fuses, Config, Биты защиты...), и установите флаги, разрешающие доступ к этим областям. Если необходимо записать EEPROM, на закладке "Буфер" надо установить флаг доступа к EEPROM. Если микросхема имеет режим электрического стирания, то установите флаг "[Стирать микросхему перед записью](#)". Менять значения напряжений питания, записи или стирания, а также параметры импульса записи, без необходимости не рекомендуется.



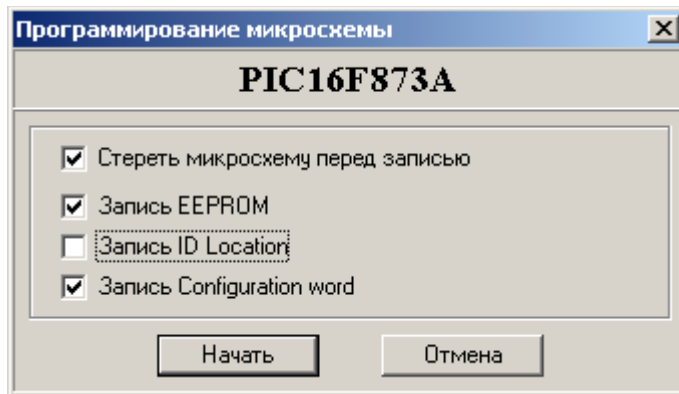
Для сокращения времени записи, можно установить флаг "[Одна контрольная сверка после чтения или записи](#)" (V1/V2). Если размер данных в файле прошивки меньше, чем размер микросхемы, то можно установить флаг "[Установить конечный адрес по размеру файла](#)" (SetAdr). Значения этих флагов отображаются в правой половине строки состояния и их можно менять двойным кликом мыши.

На закладке "Параметры" посмотрите, как устанавливается микросхема в панельку программатора. При использовании переходной панели проверьте ее название и убедитесь, что она соответствует типу, указанному при выборе микросхемы.



Правильно установите микросхему в панельку программатора и нажмите кнопку "**PROGRAMM**". Если установлен флаг

"Подтверждать операции записи и стирания", то программа выведет запрос о начале цикла записи. Если микросхема имеет режим стирания, EEPROM или другие дополнительные области, которые нужно записать, то можно разрешить к ним доступ, непосредственно из этого окна. Флаги режимов работы в этом окне имеют более высокий приоритет, чем одноименные флаги в окне проекта.

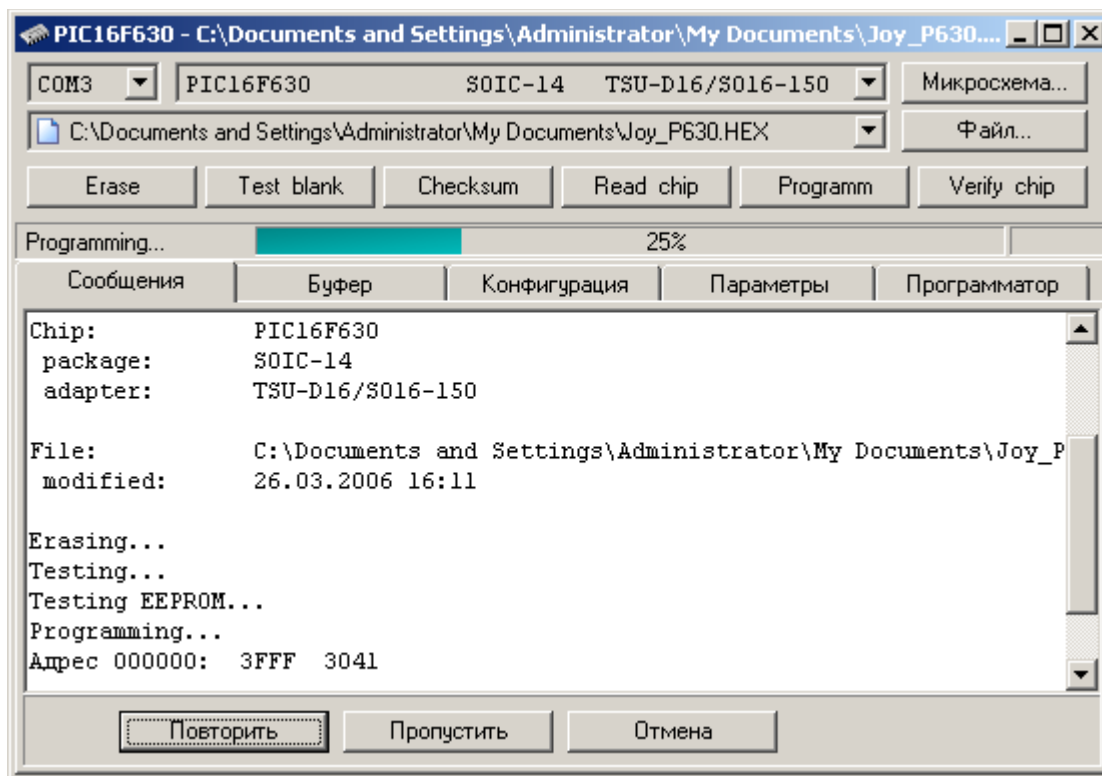


Программа проверит версию программатора, загрузит в него текущие настройки и данные из файла. Программатор проверит правильность установки микросхемы (для некоторых микросхем эта функция недоступна), сформирует необходимые напряжения, сигналы и обеспечит работу с микросхемой. Компьютер в процессе работы только подгружает данные и выводит на экран ход процесса, результаты работы и сообщения об ошибках. Благодаря такой схеме обеспечивается высокая скорость работы, безопасность микросхемы и точное соблюдение всех временных параметров в алгоритме записи.

Стандартный цикл программирования включает несколько этапов:

- стирание микросхемы, если установлен флаг и микросхема имеет этот режим;
- проверка на чистоту или на возможность записи (после стирания – только проверка на чистоту);
- программирование основной памяти микросхемы;
- запись дополнительных областей памяти (EEPROM, Fuses, и т.д.);
- контроль качества записи (сверка) при минимальном и максимальном напряжении питания;
- установка защиты, запись конфигурационного слова.

Программа обеспечивает независимое управление несколькими программаторами в фоновом режиме. Поэтому, **сообщения об ошибках также выводятся в фоновом режиме**. При возникновении ошибки на любом из этапов, процесс работы с микросхемой прерывается, на закладке "**Сообщения**" выводится сообщение об ошибке, и программа **ждет решение пользователя**. Диалог с пользователем ведется в каждом окне, независимо от других окон.



В зависимости от режима, программа предлагает продолжить работу или прервать цикл программирования. При прерывании цикла, если ошибка возникла при записи микросхемы, то дальнейшая запись, верификация и установка защиты не проводятся. Если ошибка возникла на этапе верификации, то не производится установка защиты.

После окончания процесса записи выводится сообщение об итогах работы, контрольная сумма микросхемы и обновляются счетчики микросхем.

При необходимости, цикл чтения можно безопасно прервать. Для этого, справа от прогресс-бара есть поле счетчика. Если кликнуть по нему во время работы с микросхемой, то программатор прервет цикл и перейдет в режим готовности.

Более подробно режимы записи каждой микросхемы описаны в разделе "[Особенности работы с микросхемами](#)", а возможные ошибки в "[Описании сообщений программы](#)".

Команда TEST BLANK

Проверка микросхемы на чистоту или возможность записи производится при минимальном значении напряжения питания ($V_{cc\ min}$). Это гарантирует, что предыдущие данные надежно стерты и микросхема готова к программированию. Если флаг "[При проверке на чистоту не загружать файл](#)" сброшен, то подгружается файл и микросхема проверяется на возможность записи.

В случае успешного завершения теста выводится сообщение "TEST OK". Если микросхема не чистая, но ее содержимое можно перепрограммировать новыми данными из файла, то выводится сообщение "CHIP NOT BLANK" (микросхема не чистая). Если микросхему без стирания переписать нельзя, то сообщение - "WRITE IMPOSSIBLE" (дальнейшая запись не возможна).

Команда CHECKSUM

В программаторе реализованы два алгоритма подсчета контрольной суммы, в зависимости от типа выбранной микросхемы. Контрольная сумма для микросхем с 8-битной шиной данных подсчитывается путем сложения всех байтов с переносом в старший байт. Для микросхем с 16-битной шиной данных контрольная сумма есть сумма всех 16-битных адресов без учета переносов. Если микросхема имеет дополнительные режимы работы, и доступ к ним разрешен, то контрольная сумма подсчитывается с учетом этих режимов.

Микросхема в этом режиме работает при номинальном напряжении питания ($V_{cc\ nom}$).

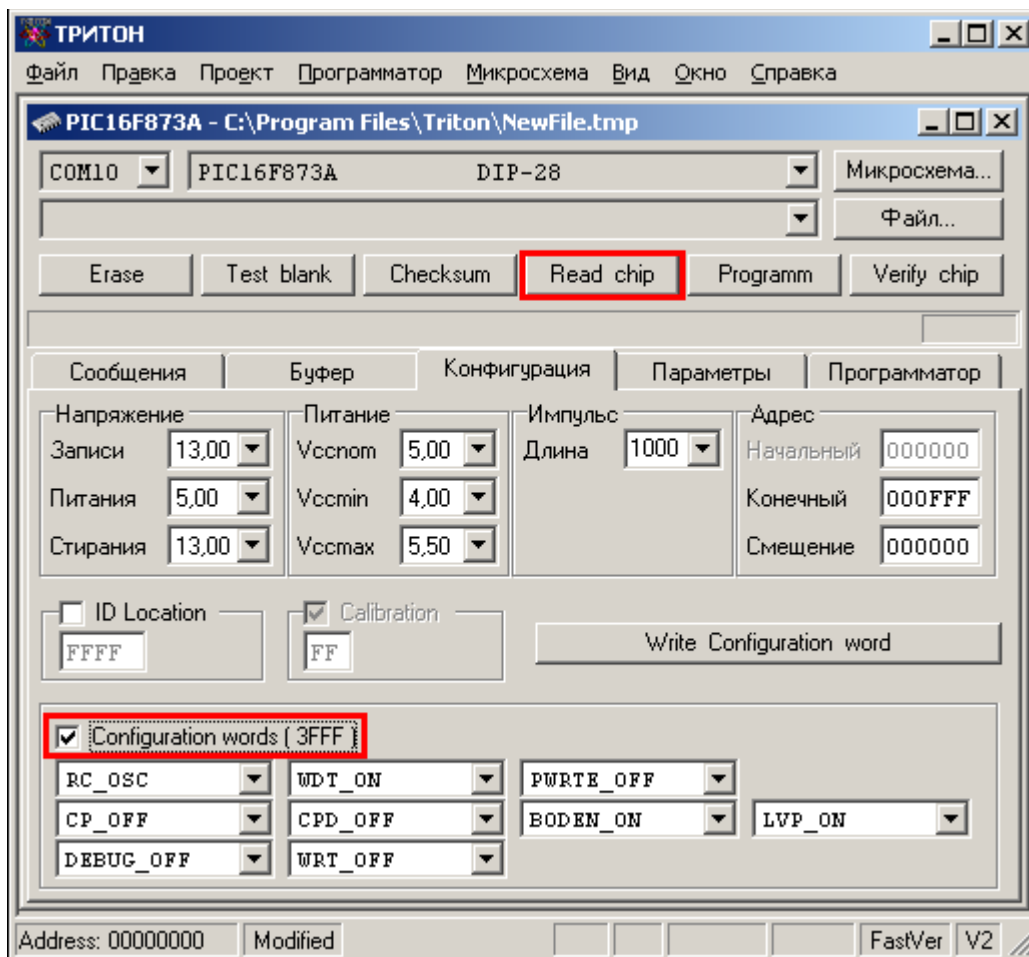
Для PIC-контроллеров контрольная сумма подсчитывается согласно спецификации фирмы MICROCHIP.

Контрольная сумма, находящаяся на закладке "[Буфер](#)", не имеет никакого отношения к контрольной сумме микросхемы. Они могут совпадать, если размер файла равен объему микросхемы и микросхема не имеет дополнительных областей, к которым разрешен доступ.

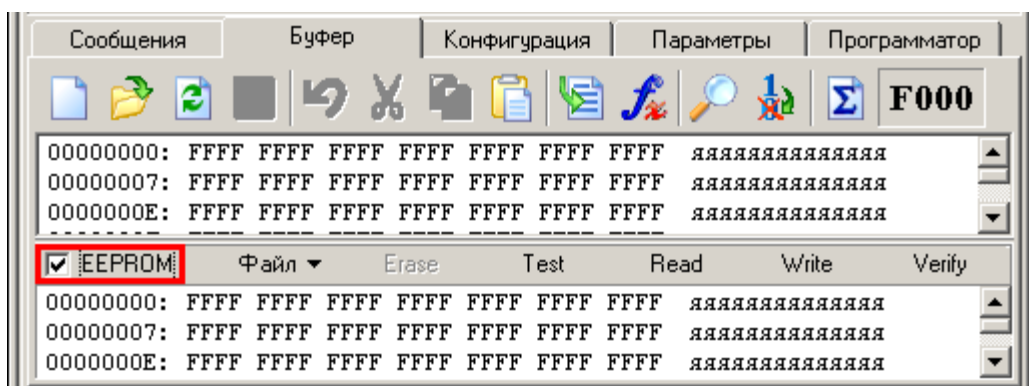
Команда READ CHIP

После выбора микросхемы ее название и параметры будут показаны в окне проекта. Все напряжения, адреса и другие параметры устанавливаются в заводское состояние, и менять их без необходимости не рекомендуется.

Чтение содержимого микросхемы осуществляется командой "**READ CHIP**". Если в микросхеме имеются дополнительные области, такие как EEPROM, Fuses, Биты защиты, и их необходимо прочитать или записать, то надо установить соответствующие флаги.



В данном примере, чтобы считать конфигурационное слово надо установить флаг "Configuration word", а для чтения памяти данных (EEPROM), на закладке "Буфер" надо установить флаг доступа к EEPROM.

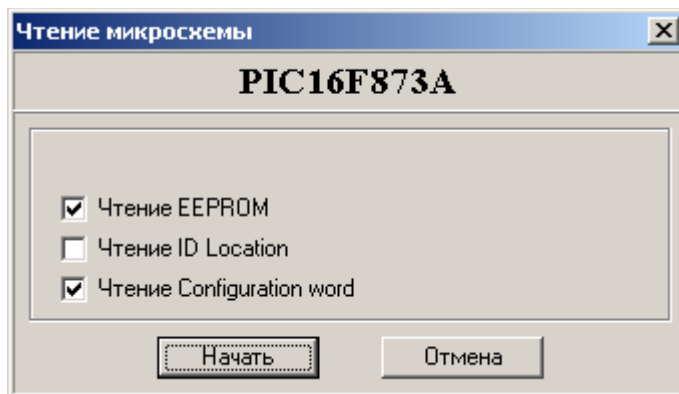


На закладке "Параметры" посмотрите, как устанавливается микросхема в панельку программатора. При использовании переходной панельки проверьте ее название и убедитесь, что она соответствует типу, указанному при выборе микросхемы.



Когда все необходимые режимы установлены, для чтения содержимого микросхемы нажмите кнопку "READ CHIP". Программа откроет закладку "Буфер", зарезервирует в памяти компьютера необходимую область и заполнит ее значением "FF". Если установлен флаг "[Подтверждать операции записи и стирания](#)", то программа выведет запрос на начало чтения. Если микросхема имеет EEPROM или другие дополнительные области, которые можно прочитать, то можно разрешить или запретить к ним доступ, непосредственно из этого окна. Флаги режимов работы в этом окне имеют более высокий

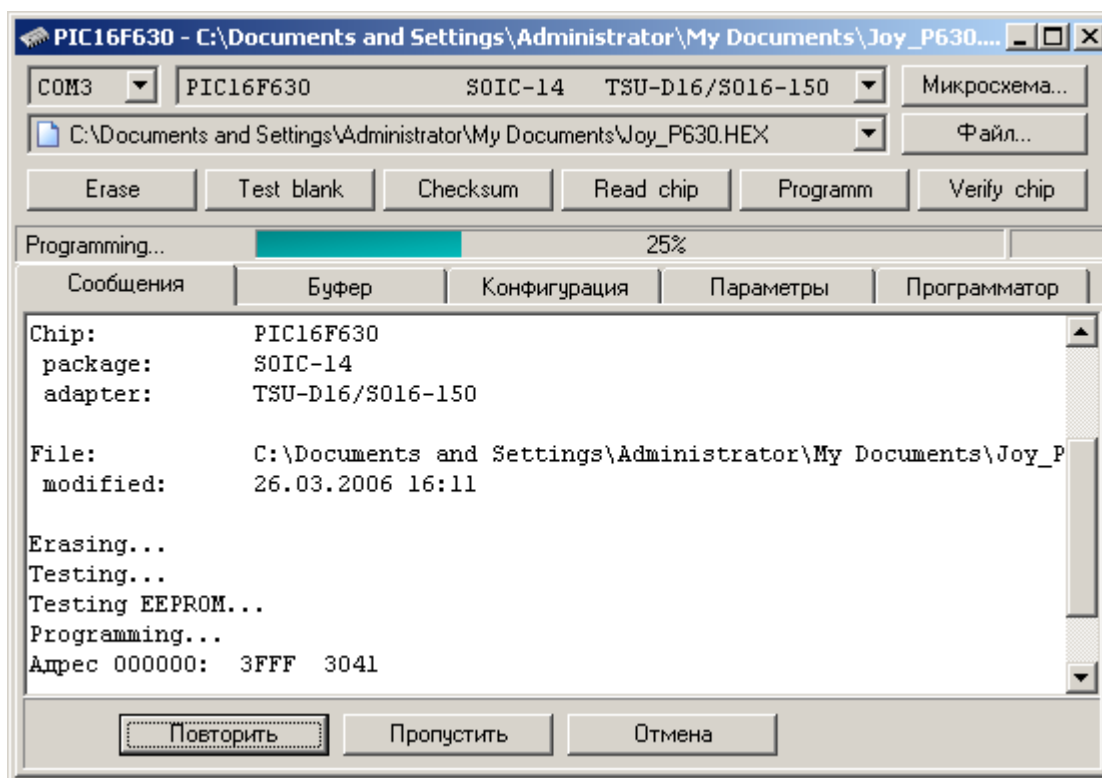
приоритет, чем одноименные флаги на панели управления.



Перед каждым циклом работы программа проверяет версию программатора и загружает в него текущие настройки. Программатор проверяет правильность установки микросхемы (для некоторых микросхем эта функция недоступна), подает питание на микросхему и начинает считывать данные, поблочно передавая их в компьютер.

Основная память микросхемы читается всегда, дополнительные области читаются и обрабатываются программой, только при условии, что к ним разрешен доступ. После окончания цикла чтения всегда производится контрольная сверка содержимого микросхемы и считанных данных, и подсчитывается контрольная сумма, с учетом установленных режимов работы. По умолчанию, контрольная сверка производится сначала при минимальном, а затем при максимальном напряжении питания. Если установить флаг "[Одна контрольная сверка после чтения или записи](#)", или сделать одинаковыми значения минимального и максимального напряжений питания, то контрольная сверка будет проведена только один раз.

Программа обеспечивает независимое управление несколькими программаторами в фоновом режиме. Поэтому, в отличие от старых версий и других программаторов, **сообщения об ошибках также выводятся в фоновом режиме**. При возникновении ошибки, программа открывает закладку "[Сообщения](#)", выводит информацию об ошибке и **ждет решение пользователя**. Диалог с пользователем ведется в каждом окне, независимо от других окон.



При необходимости, цикл чтения можно безопасно прервать. Для этого, справа от прогресс-бара есть поле счетчика. Если кликнуть по нему во время работы с микросхемой, то программатор прервет цикл и перейдет в режим готовности.

Более подробно режимы чтения каждой микросхемы описаны в разделе "[Особенности работы с микросхемами](#)", а возможные ошибки в "[Описании сообщений программы](#)".

Команда VERIFY CHIP

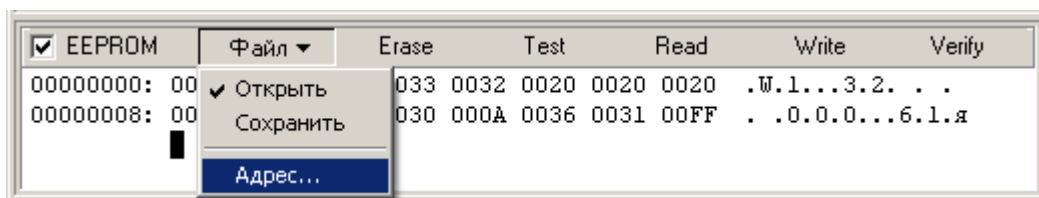
Этот режим позволяет дополнительно проконтролировать правильность чтения данных или корректность записи микросхемы, путем побитовой сверки содержимого микросхемы с содержимым файла, и подсчитать контрольную сумму микросхемы. Проверка выполняется дважды: при минимальном (Vcc min) напряжении питания, затем, если ошибок не

обнаружено, при максимальном (Vcc max) напряжении питания.

Если минимальное напряжение питания равно максимальному, то повторная сверка не проводится. Во время работы с компьютером, если установлен флаг "[Одна контрольная сверка после чтения или записи](#)", то независимо от установленных напряжений, производится только одна контрольная сверка при номинальном напряжении питания. При работе в автономном режиме состояние этого флага игнорируется, и количество проходов определяется значениями напряжение питания (Vcc min и Vcc max).

Команды для работы с EEPROM

Команды для работы только со встроенной памятью данных. В зависимости от настроек EEPROM в дополнительных параметрах микросхемы данные для сверки и записи будут взяты из основного файла, начиная с заданного адреса, или из отдельного файла.



Стирание. Эта команда доступна только для микросхем, в которых есть режим отдельного стирания EEPROM. По окончании стирания производится проверка EEPROM на чистоту. Этот режим аналогичен полному стиранию микросхемы за исключением того, что доступ осуществляется только к EEPROM.

Проверка. Эта команда проверяет EEPROM на чистоту. Если флаг "[При проверке на чистоту не загружать файл](#)" сброшен, то подгружается файл и EEPROM проверяется на возможность записи. Проверка производится при минимальном напряжении питания.

Чтение. Эта команда обеспечивает чтение EEPROM в буфер. После чтения будет произведена контрольная сверка. Сверка выполняется один раз при номинальном напряжении питания.

Запись. Эта команда обеспечивает программирование EEPROM и последующую проверку качества записи. При записи EEPROM используются те же настройки напряжений, что и при программировании основной памяти микросхемы. Контрольная сверка выполняется один раз при номинальном напряжении питания.

Сверка. Эта команда обеспечивает сверку содержимого EEPROM и данных из памяти компьютера. Сверка повторяется дважды, при минимальном и максимальном напряжениях питания.

Программаторы ТРИТОН+ при работе в автономном режиме не поддерживают отдельные файлы для EEPROM памяти данных.

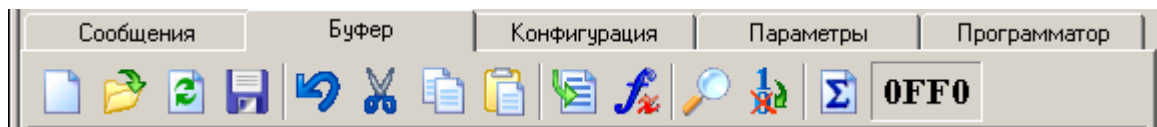
Для автономных программаторов ТРИТОН+ данные для записи EEPROM должны находиться в основном файле, в следующем блоке, сразу после последнего записываемого адреса. Размер блока для программатора V5.4(T) равен 256 байт, для V5.5(T), V5.6(T), V5.7T и V5.8T = 512 байт. Как правило, размер микросхемы всегда кратен 256 байтам, и когда пишется вся микросхема, EEPROM всегда начинается с нового блока. Например, если необходимо записать только небольшую часть микросхемы, первые 100 байт (начальный адрес равен \$000000, конечный \$000063), то данные для записи EEPROM должны быть размещены, начиная с адреса \$000100 для V5.4(T) или \$000200 для V5.5(T), V5.6(T), V5.7T и V5.8T.

Во время загрузки [проекта](#) в память программатора, программа сама размещает данные EEPROM по нужному адресу.

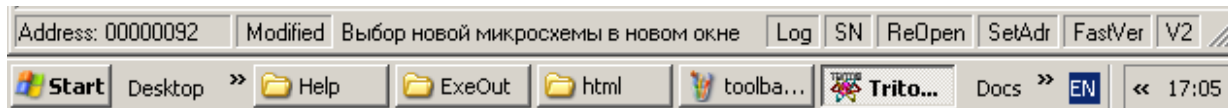
Меню Вид

Меню "Вид" содержит команды для настройки внешнего вида программы.

Панель инструментов – показывает на экране или скрывает панель с кнопками команд во всех окнах шестнадцатеричного редактора.



Строка состояния – показывает внизу основного окна программы строку статуса, в которой выводятся краткие описания элементов управления, состояния некоторых флагов и другая информация.



Демо-режим – переводит программу в демонстрационный режим. В этом режиме, при закрытии программа не сохраняет никаких настроек. Это полезно, когда надо поработать с другими микросхемами, но при этом не хотелось бы менять текущую конфигурацию программы.

Язык – переключает язык интерфейса программы. Программа позволяет подключить любое число языковых файлов (*.lng). Для этого надо перевести основной файл сообщений "L_rus.lng" на нужный язык, в секции [LngFile] ввести название языка Language=xxxxxxx и положить новый файл в корневую папку программы. Новый язык будет доступен после перезагрузки программы. Список языков будет отсортирован по именам lng-файлов.

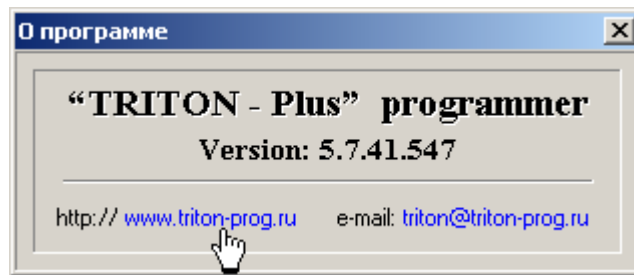
Меню Окно

Меню "Окно" содержит стандартные команды для настройки положения окон внутри программы и смены активного окна. Команды размещения работают только с развернутыми окнами. Например, в программе открыто несколько окон. Чтобы развернуть два из них на весь экран, необходимо минимизировать не нужные окна и затем выбрать команду размещения окон.

Меню Справка

Руководство пользователя – показывает на экране это руководство.

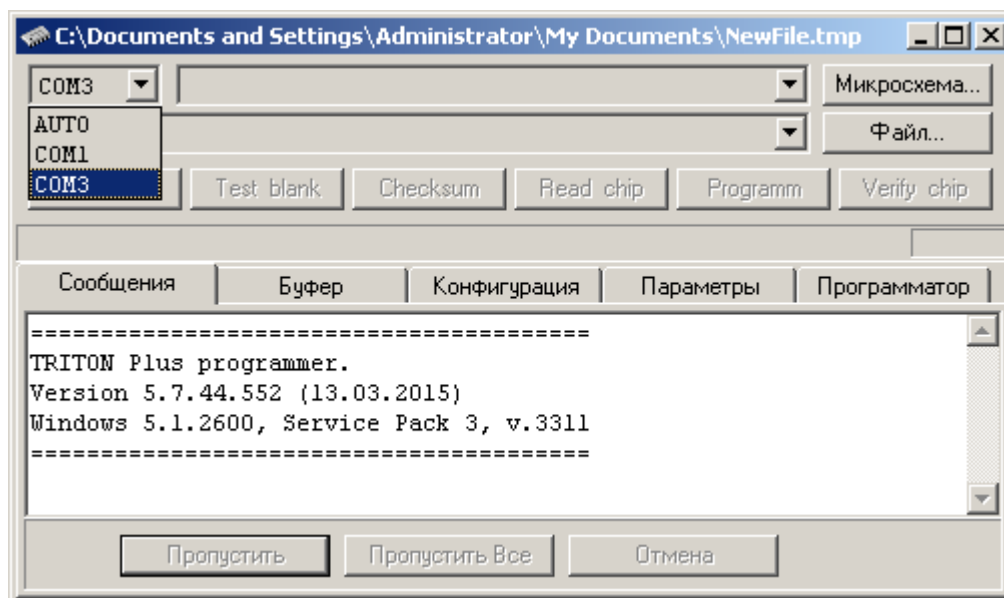
О программе – показывает на экране диалог с номером версии программы и ссылками в Интернет и службу поддержки.



Окно проекта

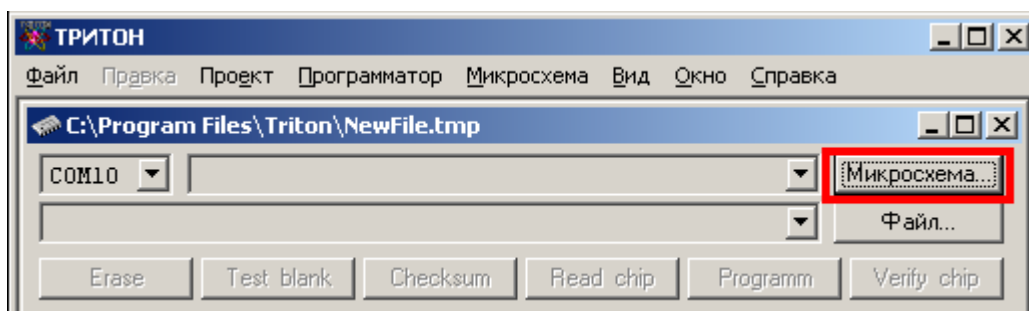
Оболочка программатора многооконная и каждое окно может управлять своим собственным программатором или несколько окон могут по очереди управлять одним программатором. Количество окон, также как количество программаторов может быть любым. Управление всеми программаторами осуществляется независимо друг от друга и ведется в фоновом режиме. Программа не накладывает ограничений на тип подключения программатора или на режим его работы. Например, один программатор, подключенный через COM порт, может читать одну микросхему. В то время как на другом программаторе, подключенным через USB, можно программировать микросхемы другого типа. При этом в оболочке можно редактировать файл или отлаживать скрипт.

Выбор программатора осуществляется в каждом окне из списка доступных портов.



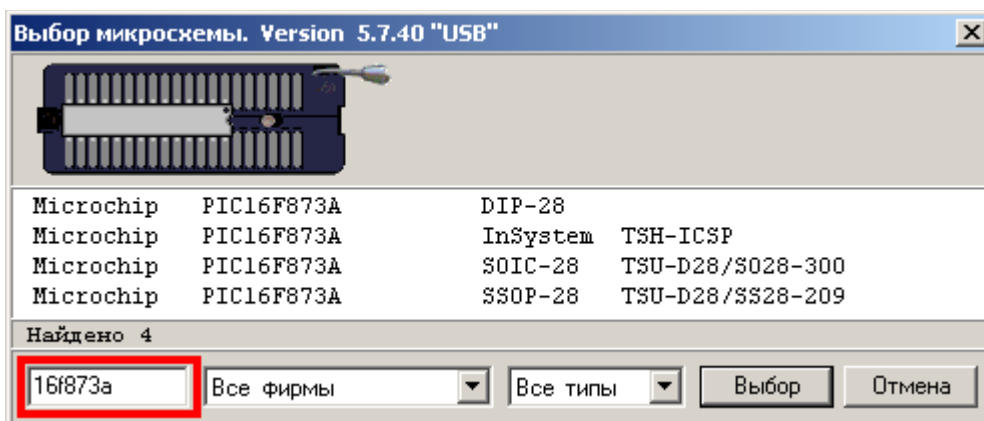
Выбор микросхемы

Для выбора микросхемы в окне проекта нажмите кнопку с надписью "Микросхема" или сразу после открытия окна, нажмите "Enter".



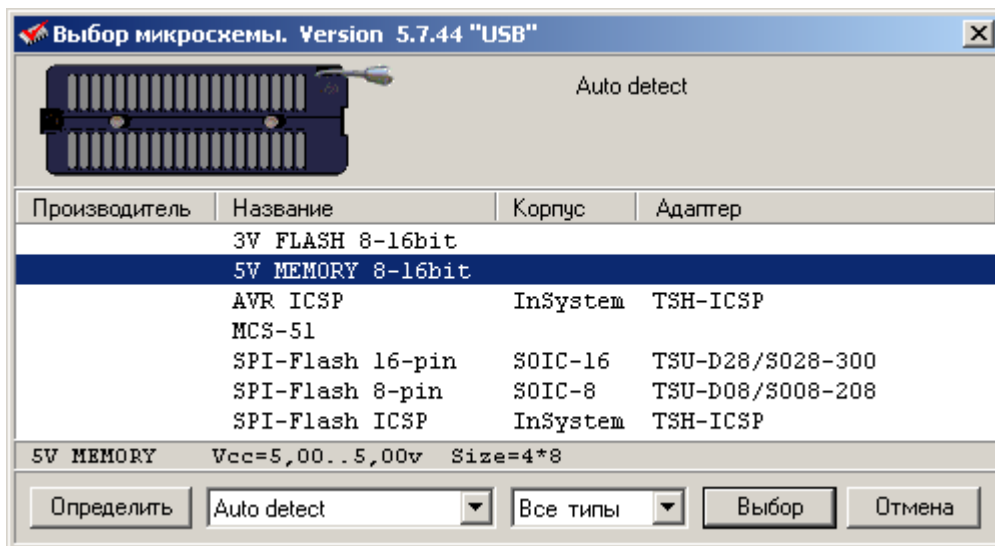
Откроеется окно "Выбор микросхемы", в котором будут представлены все микросхемы поддерживаемые программатором. Список может быть отсортирован по **типам** микросхем или по **фирмам** производителям. Дополнительно, список микросхем конкретной фирмы может быть отсортирован по типам микросхем.

Для удобства выбора, имеется возможность **поиска по нескольким символам**, идущим подряд в названии микросхемы. Если сразу после вывода окна набрать на клавиатуре несколько символов, то программа отфильтрует список и покажет все микросхемы, в названии которых встречается введенная комбинация букв и цифр.

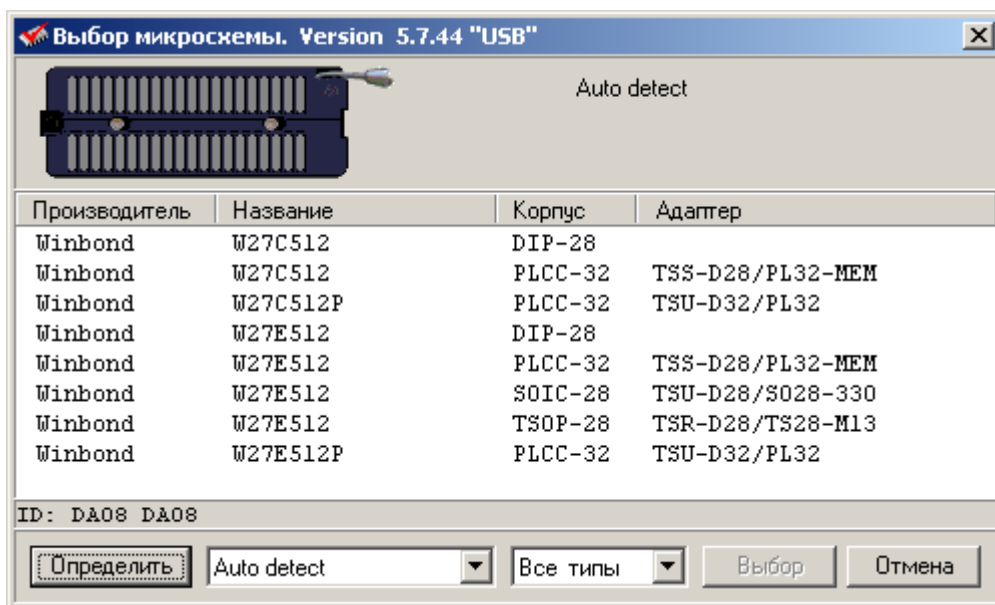


Автоматическое определение типа микросхемы. Для микросхем стандартной памяти (EPROM, FLASH) с параллельным доступом, SPI и NAND-Flash, микроконтроллеров AVR и MCS доступен режим автоматического определения. Необходимо отметить, что не все микросхемы поддерживают режим чтения сигнатуры, кроме того, существует несколько алгоритмов автоопределения, часть из которых пока не реализована. Алгоритмы чтения сигнатуры реализованы в программаторе с помощью скриптов, что позволяет считывать сигнатуры практически любых микросхем. Найти эти файлы можно в папке "Scripts", в директории, куда установлена программа. Посмотреть или изменить параметры можно просто выбрав нужную запись. Чтобы выбранный скрипт работал как обычная микросхема, нужно в параметрах изменить номер алгоритма на \$1F.

Для автоматического определения установите микросхему в панельку программатора, выберите группу "Auto Detect" в списке производителей, затем выберите нужный тип микросхемы и адаптера, и нажмите кнопку "Определить". Программа запомнит выбранный алгоритм и будет использовать его при последующих командах. Сменить алгоритм автоопределения можно путем повторного выбора группы "Auto Detect" в списке производителей.



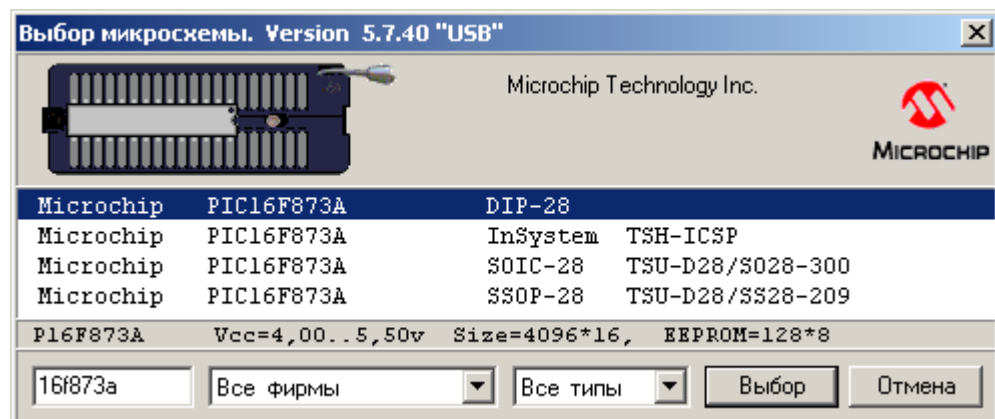
Если микросхема содержит коды сигнатуры, и эти коды зарегистрированы в базе данных программатора, то в окне будет показан список микросхем, содержащих эти коды.



Для каждой микросхемы указывается тип корпуса и соответствующая ему переходная панелька или адаптер. Обратите внимание на то, что одна и та же микросхема может выпускаться в разных корпусах или для одного и того же корпуса могут быть использованы разные панельки. Поэтому, всегда **внимательно выбирайте нужную микросхему** и используйте только тот адаптер, название которого указано в программе.

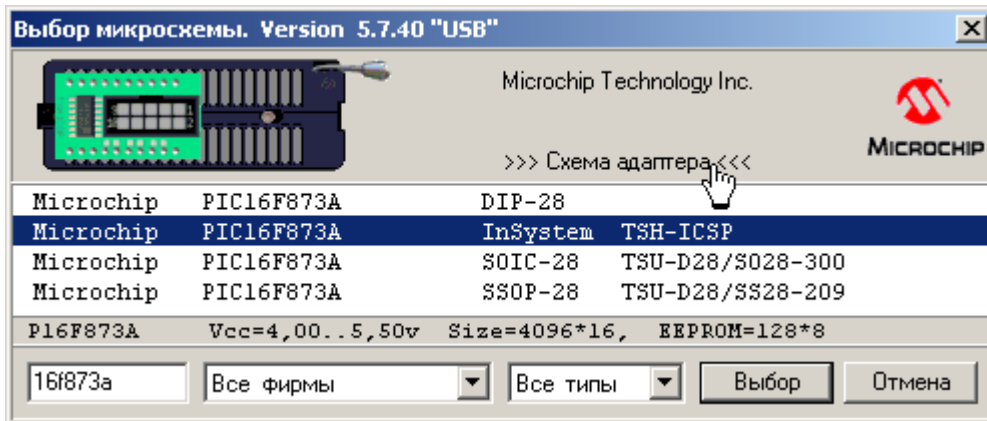
Выделив нужную микросхему, программа покажет следующую информацию о микросхеме:

- Положение микросхемы в панельке программатора;
- Название, логотип фирмы-изготовителя и ссылку на его сайт;
- Сокращенное название микросхемы, напряжение питания, размер и организацию основной (и дополнительной) памяти.



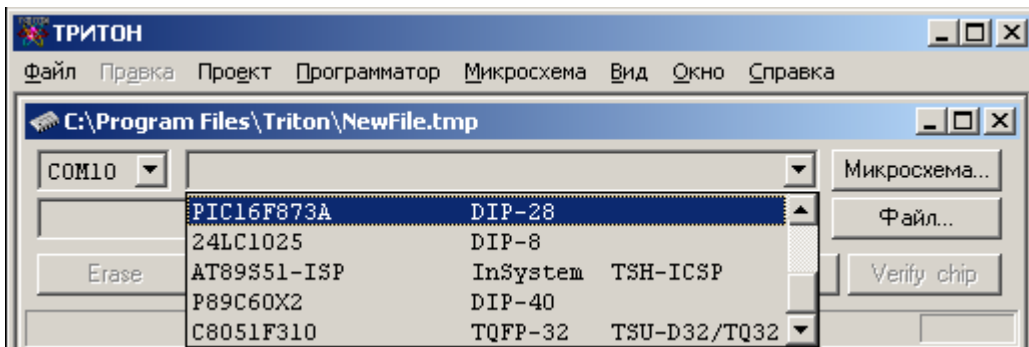
Для микросхем, которые используют специальные адаптеры, можно посмотреть схемы разводки. Если кликнуть мышкой

по надписи "Схема адаптера", то в новом окне будет показана принципиальная схема этого адаптера. Принципиальные схемы всех адаптеров находятся в папке "Panels" в директории, куда инсталлирована оболочка программатора. Для стандартных и универсальных адаптеров схемы не приводятся.



Чтобы выбрать микросхему дважды кликните мышкой по ее названию или выделите нужную строку и нажмите кнопку "Выбор". Закроется диалог выбора, а название и параметры микросхемы будут показаны в окне проекта.

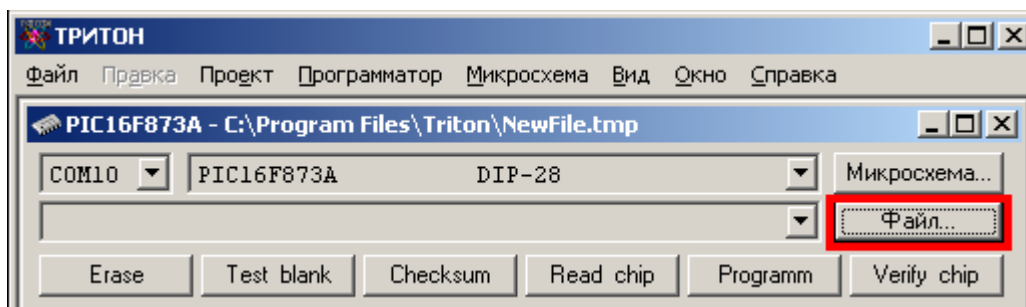
Если Вы уже работали с этой микросхемой, то для ускорения выбора, можно выбрать микросхему из списка недавно использовавшихся микросхем. Список сортируется по времени. Последняя выбранная микросхема помещается в начало списка.



Для удаления позиции из этого списка необходимо открыть список, выбрать микросхему, которую нужно удалить и нажать "Delete".

Выбор файла

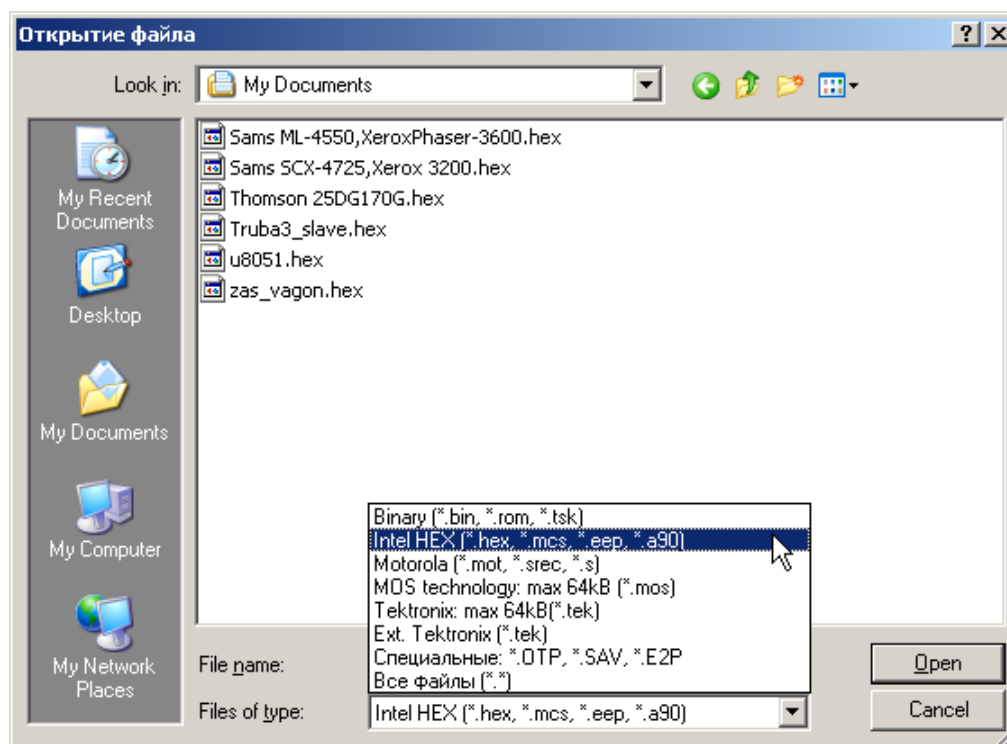
Чтобы открыть файл для записи в микросхему нажмите кнопку "Файл" в окне проекта.



Откроется стандартный диалог выбора файла. Можно посмотреть все файлы в папке или отфильтровать список по типу. Файлы в шестнадцатеричных форматах (Intel, Motorola, Tektronix...), при открытии автоматически преобразуются в двоичные. Если расширение файла не совпадает с зарегистрированными расширениями или его формат не соответствует спецификации, то файл будет открыт как двоичный.

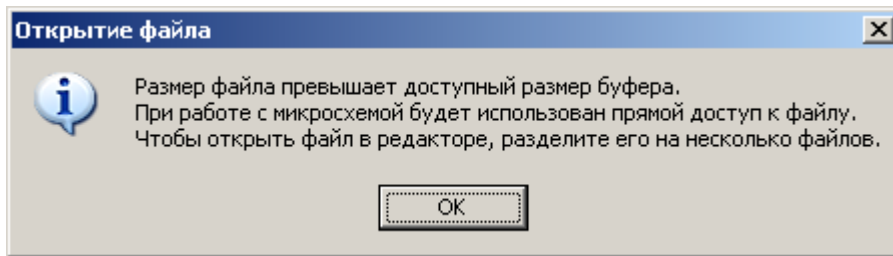
Программное обеспечение поддерживает следующие форматы файлов:

- Двоичный формат (объем файла не ограничен);
- Intel Hexadecimal object file format (8- и 16-bit);
- Intel Extended HEX file format (32-bit);
- Motorola S-Record hexadecimal file format;
- Tektronix Hexadecimal file format;
- Tektronix Extended Hexadecimal file format;
- MOS Technologies Hexadecimal file format;
- Сжатые файлы в формате *.ZIP;
- Специальные форматы (только чтение): Holtek OTP, Ангстрем SAV, E2P.



Найдите нужный файл на диске и нажмите кнопку "Открыть". Программа считывает содержимое файла и покажет его в окне редактора. В зависимости от организации памяти выбранной микросхемы, файл будет открыт в 8, 16 или 32 битном формате. Формат представления данных в редакторе можно сменить, выбрав меню "Правка/Формат данных". Изменение формата не изменяет содержимое буфера. При работе с файлом программа не запрещает доступ к нему из других программ. В момент открытия файла, в памяти компьютера создается буфер, куда считываются данные. В дальнейшем программа работает только с этим буфером. Максимальный объем буфера определяется доступной памятью компьютера, но не может превышать 2 Гбайт.

Если размер файла превышает размер свободной памяти в компьютере или размер файла превышает 2 Гбайта, то программа загружает в буфер часть файла, размером 128Мбайт, в режиме только для чтения (ReadOnly). Для работы с микросхемой программа будет использовать режим прямого доступа к файлу, минуя буфер.



Если при открытии файла в папке находятся несколько файлов с одинаковым именем и расширениями *.bin, *.bin1, *.bin2, ..., то программа откроет их как группу файлов. При этом в буфер будет загружен файл с расширением *.bin, а при работе с микросхемой будет использован режим прямого доступа к файлу и данные в буфере будут проигнорированы.

Файлы в шестнадцатеричных форматах (Intel, Motorola, Tektronix...) при открытии автоматически преобразуются в двоичные. Поскольку эти файлы могут иметь пустые области, то программа предварительно заливает весь буфер константой, после чего загружает данные из файла. Значение константы (\$FF или \$00) определяется выбранной микросхемой и может быть изменено в блоке параметров микросхемы (ячейка \$BC - Test Blank Mask_L).

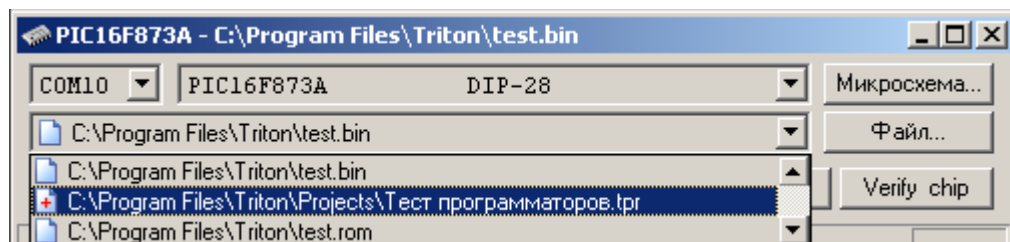
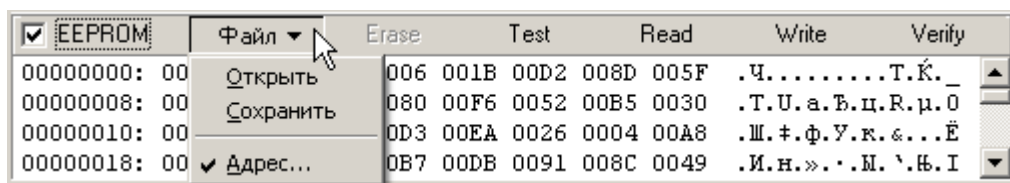
При сохранении шестнадцатеричных файлов (сочетание клавиш "Ctrl+S") они будут сохранены в оригинальном формате. Чтобы сохранить файл в другом формате, выберите в меню "Файл" пункт "Сохранить как..." и смените тип файла. В настройках программы есть флаг "[Сохранять пустые области в HEX файлах](#)", при установке которого в файл записывается полный образ дампа в нужном формате. При сброшенном флаге, пустые области данных (\$FF или \$00) в файле не сохраняются.

При открытии специальных файлов (*.E2P, *.SAV или *.OTP), если выбрана соответствующая микросхема, программа преобразует эти файлы и считывает из них только данные для программирования. Вся служебная информация игнорируется. При сохранении изменений в таких файлах в них сохраняются только данные для записи, т.е. создается обычный двоичный файл.

Программа позволяет изменять расширения для зарегистрированных типов файлов. Например, чтобы добавить поддержку шестнадцатеричных файлов с расширением "m51", нужно открыть в текстовом редакторе, типа "Блокнот", файл "L_rus.lng" или "L_eng.lng" (файлы находятся в папке, куда инсталлировалась программа), найти строку, которая начинается на "170=" и добавить в конец строки запись ".m51". В этой строке хранятся расширения файлов, которые программа будет пытаться конвертировать в момент открытия. Затем добавить запись ";*m51" в конец строки "181=Intel HEX..." и перезапустить программу. Для двоичных файлов нужно добавить требуемую запись только в конец строки "180=Binary...".

Для PIC контроллеров из файла считывается информация о конфигурационном слове, ID Locations, EEPROM данных и калибровочных значениях. Если открываемый файл содержит данную информацию, то соответствующий флаг доступа будет установлен.

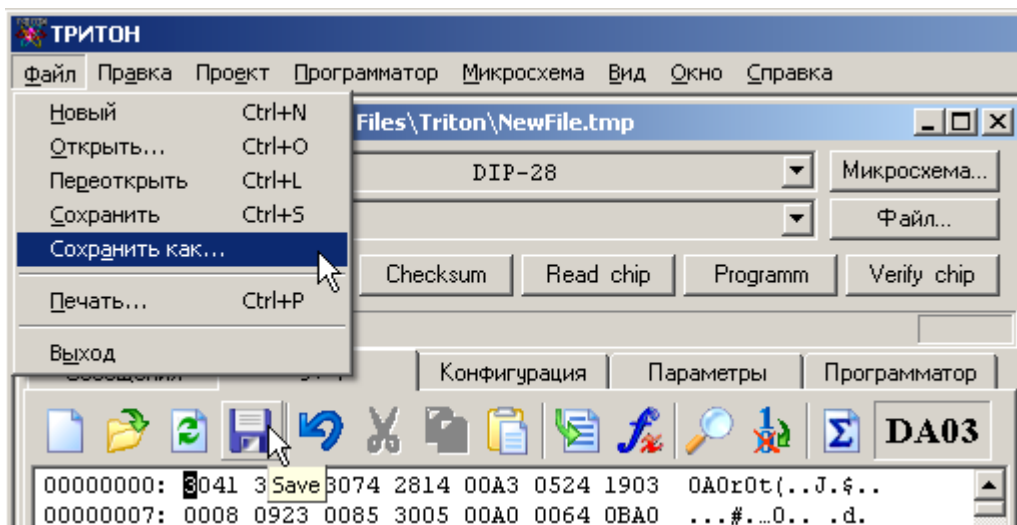
Для других микросхем, имеющих EEPROM память данных, эти данные будут считаны из основного файла или могут быть загружены из отдельного файла. В основном файле эти данные должны располагаться сразу после основной памяти микросхемы. В принципе, начальный адрес данных для дополнительной памяти может быть установлен любым, в пределах основного файла. При необходимости, на панели управления параметрами микросхемы можно оперативно поменять любые из этих установок и включить нужные режимы работы.



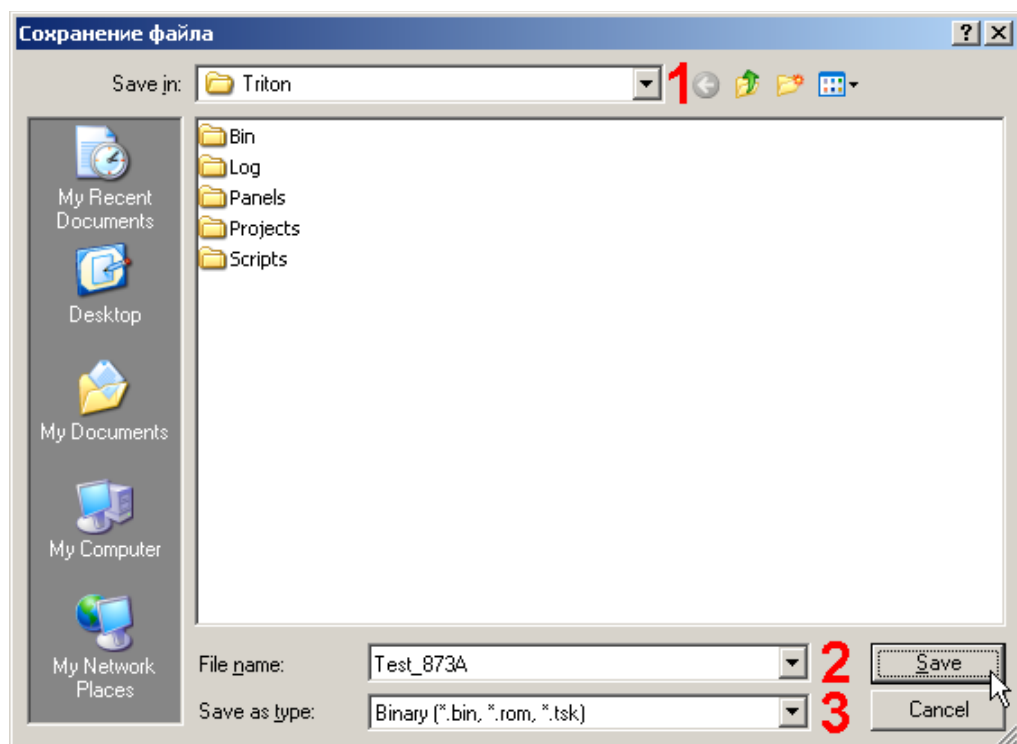
Файлы проектов имеют расширение ".TPR" и помечаются иконкой со знаком "+". При выборе проекта данное окно закрывается и будет создано новое, в котором и будет открыт выбранный проект.

Сохранение файла

Чтобы сохранить файл нажмите кнопку "Save" на панели кнопок в окне проекта или выберите в меню "Файл" пункт "Сохранить как...".



Откроется стандартный диалог сохранения файла. Выберите папку (1) куда нужно сохранить файл, затем введите имя файла (2) и выберите формат (3), в котором должен быть сохранен файл. При вводе имени файла расширение вводить не обязательно, при его отсутствии программа сама добавит расширение, соответствующее выбранному формату. Нажмите кнопку "Сохранить" (Save) для записи файла на диск.



Программное обеспечение поддерживает сохранение файлов в следующих форматах:

- Двоичный формат (объем файла не ограничен);
- Intel Hexadecimal object file format (8- и 16-bit);
- Intel Extended HEX file format (32-bit);
- Motorola S-Record hexadecimal file format;
- Tektronix Hexadecimal file format;
- Tektronix Extended Hexadecimal file format (ограничен до 256 Мбайт);
- MOS Technologies Hexadecimal file format (ограничен до 256 Мбайт);
- Сжатые файлы в формате *.ZIP.

При сохранении файла, кроме файлов с расширением "*.TMP", его имя добавляется с список активных файлов. Этот список сортируется по алфавиту и содержит до 50 имен файлов и проектов, с которыми программа работала в последнее время. Чтобы удалить запись из этого списка, нужно открыть список, выделить запись и нажать на клавиатуре кнопку "Delete".

Для PIC контроллеров, в соответствии со спецификацией Microchip, в файл всегда добавляется информация о конфигурационном слове, ID Locations, EEPROM данных и калибровочных значениях.

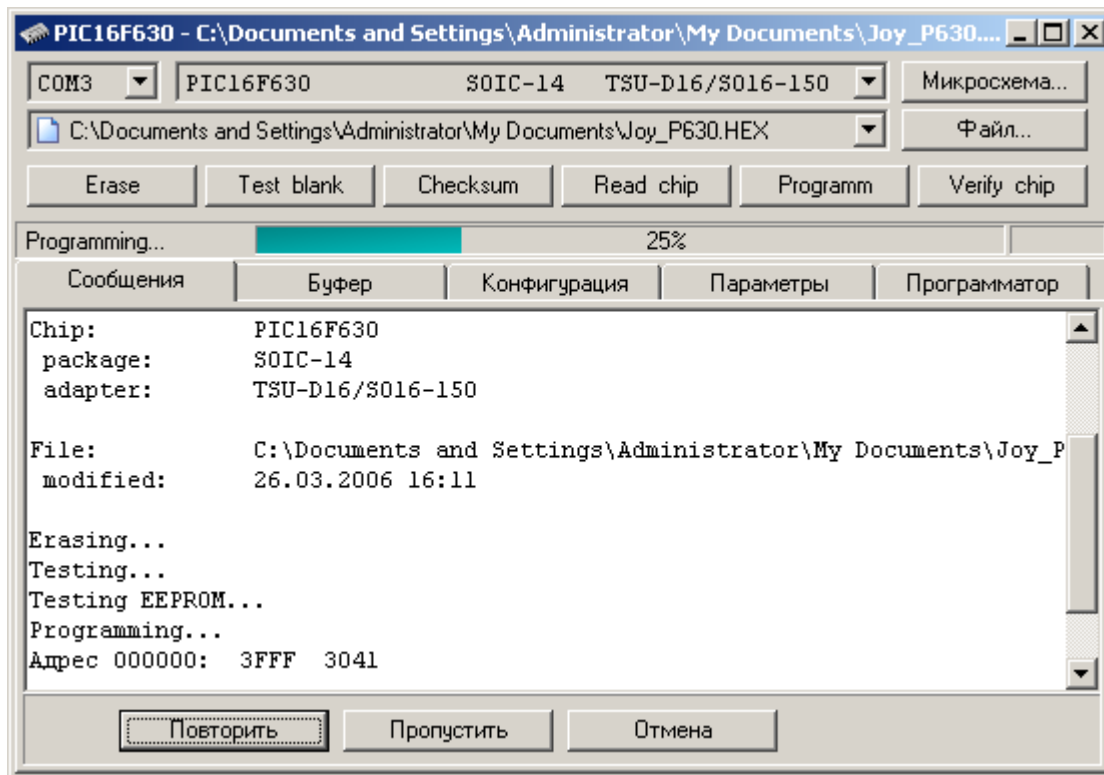
Для других микросхем, имеющих EEPROM память данных, она записывается в файл сразу после основной памяти микросхемы.

Информация о Fuses и Lock bits для AVR и других микроконтроллеров в файле не сохраняется. Сохранить эти данные можно только в виде [проекта](#).

Закладка Сообщения

Выводит различную информацию, сообщения об ошибках и обеспечивает диалог с пользователем во время работы с микросхемой.

Поскольку программа работает с программатором фоновом режиме, то и сообщения об ошибках выводятся в фоновом режиме, не мешая работе с другими программаторами. При ошибке программа открывает закладку "Сообщения", выводит сообщение об ошибке и **ждет решение пользователя**. Сообщение об ошибке содержит адрес ошибки, содержимое микросхемы и данные из файла. Адреса и данные выводятся в шестнадцатеричном формате. Для микросхем с 8-битной шиной данных старший байт данных всегда равен нулю.



Названия кнопок могут меняться, в зависимости от режима работы. Ошибки на этапе программирования:

- **Повторить** - повторить запись ячейки, в которой произошла ошибка. Если повторная запись прошла успешно, то эта ошибка не засчитывается, продолжается программирование, а при достижении конца микросхемы начинается верификация и другие режимы работы.
- **Пропустить** - пропустить этот адрес и продолжить запись до появления следующей ошибки или до конца микросхемы, после чего цикл программирования будет прерван.
- **Отмена** - остановить процесс записи и вывести информацию об итогах работы. При этом все последующие режимы работы не проводятся.

Ошибки в режимах верификации и проверки на чистоту:

- **Пропустить** - пропустить этот адрес и продолжить работу до появления следующей ошибки или до конца микросхемы.
- **Пропустить все** - продолжить работу до конца микросхемы и посчитать общее количество ошибок. По окончании проверки будет выведено сообщение о возможности продолжать работу.
- **Отмена** - прервать выполнение режима и вывести информацию об итогах работы. В этом случае проверка основной памяти прекращается, но программатор дополнительно может проверить EEPROM, конфигурационное слово и состояние битов защиты.

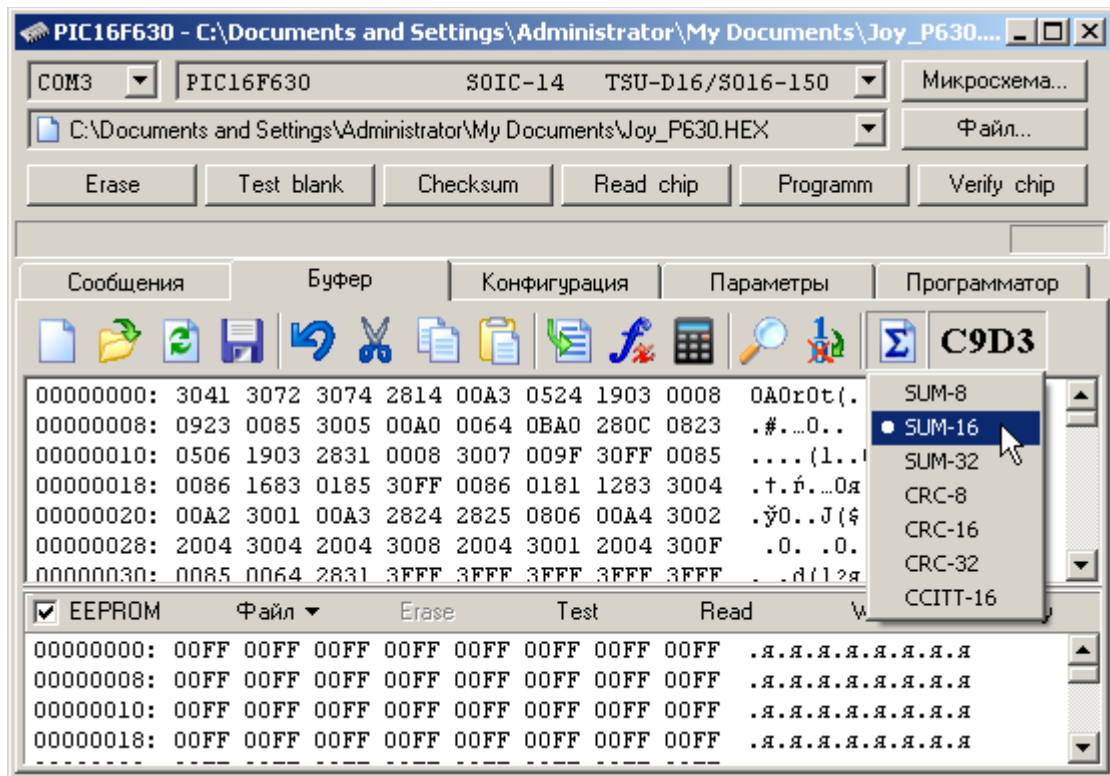
В момент вывода сообщения об ошибке микросхема в панельке программатора находится в неактивном состоянии, но под напряжением. Извлечение в этот момент микросхемы из панельки может повредить находящуюся в ней информацию.

Большинство современных микросхем имеют встроенный механизм записи. Для ускорения работы программатор контролирует только флаги микросхемы: флаг окончания записи и, если есть, флаг ошибки. При записи такой микросхемы, если она устанавливает флаг ошибки или превышает время ожидания, то цикл программирования прерывается с сообщением об ошибке. Верификация, запись EEPROM, запись конфигурационного слова или установка защиты после этого уже не проводятся.

Для микросхем, у которых контролируется только флаг готовности, возможные ошибки могут проявиться только на этапе верификации и, если они обнаружены, то запись конфигурационного слова или установка защиты также не производится.

Закладка Буфер

Это окно предоставляет доступ к буферу который будет использоваться для работы с микросхемой. При наличии в микросхеме дополнительной области памяти, здесь отображается второй буфер и команды для работы с этой областью. Размер каждого буфера зависит от наличия свободной оперативной памяти компьютера, но не может превышать 2 ГБайта.



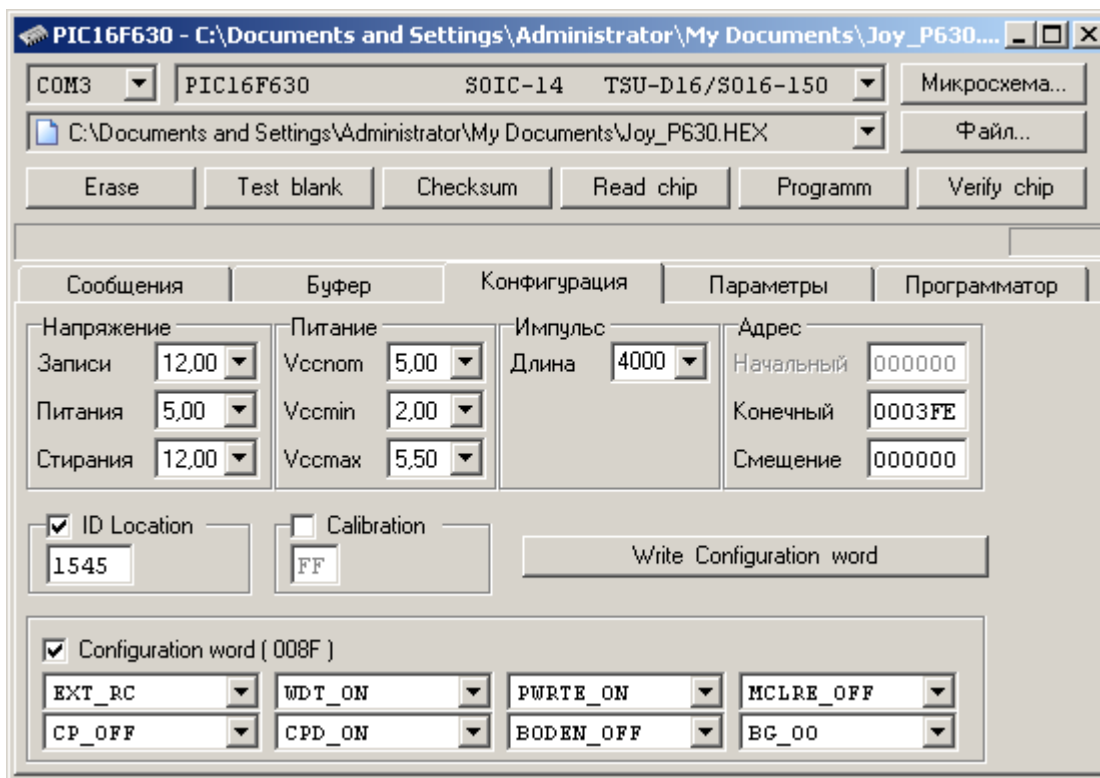
Кнопки, расположенные на панели управления, дублируют команды, находящиеся в меню [Правка](#).

CLEAR - кнопка очистить (левая кнопка на панели управления) резервирует в памяти компьютера буфер, равный объему микросхемы и заполняет его кодом FF.

Кнопка **SUM** имеет выпадающее меню, которое позволяет выбрать алгоритм, и производит подсчет контрольной суммы всего буфера или выделенного сегмента. При открытии файла программа подсчитывает и отображает контрольную сумму основного буфера, исходя из организации микросхемы. Так, для 8-битных микросхем контрольная сумма подсчитывается путем сложения всех байт, с переносом в старший байт. Для 16-битных микросхем контрольная сумма подсчитывается путем сложения всех слов, без учета переноса. Контрольная сумма, показанная в этом окне, это контрольная сумма буфера, которая может не совпадать с контрольной суммой микросхемы, полученной в результате чтения или записи. Это связано с тем, что при подсчете контрольной суммы микросхемы, программа учитывает контрольные суммы дополнительных областей микросхемы, таких как EEPROM, FUSE и LOCK bits, к которым разрешен доступ.

Закладка Конфигурация

После выбора микросхемы программа настраивает внешний вид панели управления и устанавливает все параметры программирования в соответствии с требованиями фирмы-производителя. Программатор обеспечивает высокую точность установки и хорошую стабильность всех напряжений, что гарантирует качественную запись микросхемы без дополнительных регулировок.



Программное обеспечение позволяет задать разные напряжения для разных режимов работы, таких как проверка перед записью, программирование, контрольные сверки. Во время работы с микросхемой эти напряжения будут подаваться на микросхему, обеспечивая качественную запись.

Для сокращения времени записи программа позволяет работать только с частью микросхемы, заданной начальным и конечным адресом. Если установлен флаг "[Устанавливать конечный адрес по размеру файла](#)", то перед записью программа сама скорректирует конечный адрес микросхемы, в соответствии с объемом загруженного файла.

Дополнительные параметры – это присущие только данной микросхеме настройки конфигурационного слова, битов защиты и т.д., которые могут быть загружены из исходного файла или установлены вручную. При выборе микросхемы или сбросе настроек все дополнительные параметры сбрасываются в заводское состояние, и доступ к ним запрещается. Исключения составляют PIC контроллеры, для которых, устанавливается доступ к конфигурационному слову, если оно прописано в исходном файле. Чтобы программатор во время работы с микросхемой, мог работать с EEPROM, шифровальной таблицей, Fuses, битами защиты и т.д., необходимо установить соответствующий флажок.

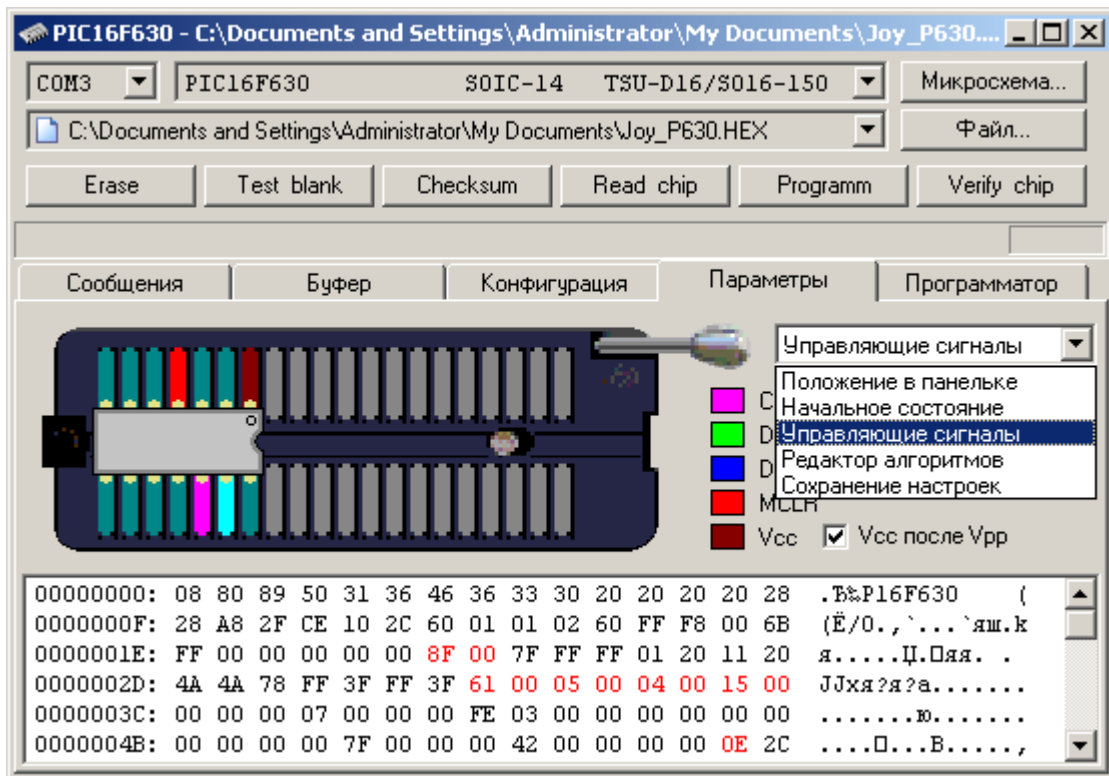
На панели управления работает контекстное меню (нажатие правой кнопки мыши), которое позволяет задать диапазон напряжений во время сверки, вернуть начальные значения адресов или всех параметров, считать из программатора блок параметров микросхемы.

При изменении настроек они сразу же вносятся в блок параметров микросхемы. Для элементов управления, которые поддерживают ввод с клавиатуры (адреса и длительность импульса записи), новое значение вводится в блок параметров после нажатия "Enter" или при выборе другого элемента или кнопки режима работы. Перемещение между элементами панели управления осуществляется с помощью клавиш "Enter", "Tab" и "Shift+Tab".

Чтобы не перегружать внешний вид программы обилием настроек, некоторые параметры никогда не отображаются на панели управления, но могут быть изменены при непосредственном редактировании блока [параметров микросхемы](#). Как правило, изменение этих параметров не требуется для нормальной работы с микросхемами, но иногда, это может пригодиться. Например, для того чтобы считать или перезаписать резервные копии калибровочных коэффициентов, изменить задержку при включении питания, перевести PIC в режим низковольтного программирования, изменить коды команд записи, переназначить сигналы на другие выходы панельки программатора...

Закладка Параметры

Это окно - "кухня разработчика", которое предоставляет доступ ко всем параметрам микросхемы. Здесь собраны все инструменты, необходимые для добавления новых микросхем. Те, кто мало знаком с программированием, тут могут посмотреть положение микросхемы в панельке, размещение адресных и управляющих сигналов, описание выводов для переходника TSH-ICSP и разъема ICP-CONN.

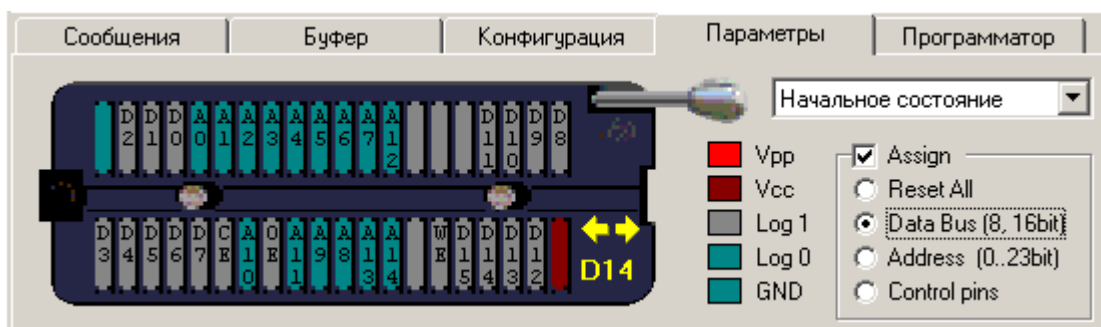


При перемещении курсора в дампе памяти, программа выводит в строке состояния описание каждого параметра, которое зависит от типа выбранной микросхемы. При перемещении указателя мыши над панелькой программатора, программа выводит номера выводов микросхемы, адреса сигналов и ключей программатора. Менять расположение сигналов можно путем непосредственной правки блока параметров или путем перетаскивания или назначения сигналов с помощью мыши. Измененные ячейки подсвечиваются красным цветом. Поскольку в этом окне осуществляется прямой доступ к параметрам микросхемы, то нет необходимости сохранять настройки перед запуском команд работы с микросхемой.

Положение в панельке - в основном, информирует о положении микросхемы в панельке программатора. Позволяет включить или отключить проверку контактов на Шине Данных. Для переходника TSH-ICSP и разъема ICP-CONN выводит расположение сигналов на разъемах. Для микросхем с последовательным доступом, позволяет перетащить микросхему в другое место на панельке, что может пригодиться в случае повреждения одного из ключей программатора.



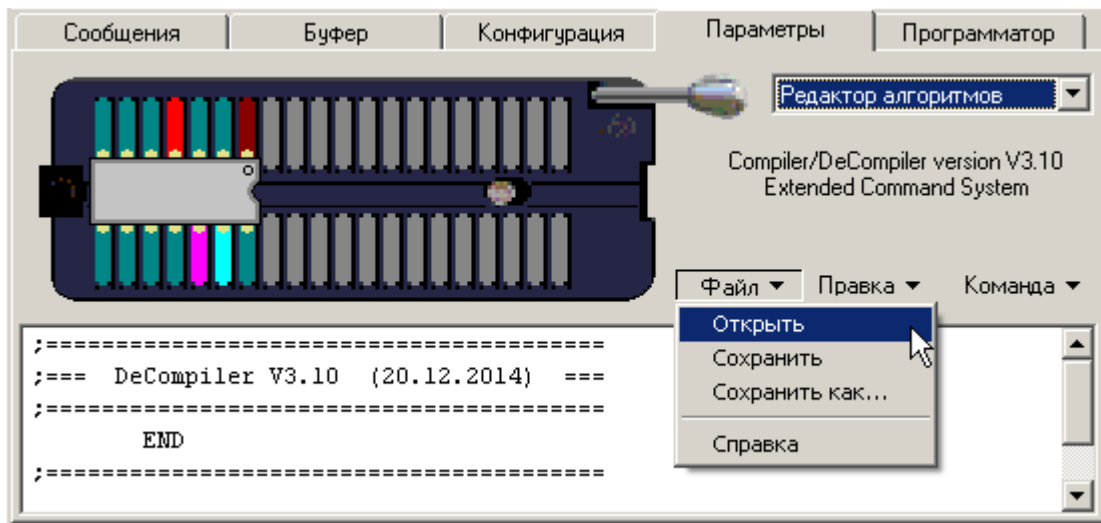
Начальное состояние. - показывает состояние выводов панельки программатора, обеспечивающее вход микросхемы в режим программирования. В этом состоянии задаются выводы питания, земли, начальные состояния адресных и управляющих сигналов. При установке состояния выводов необходимо учитывать, чтобы микросхема в момент подачи питания, находилась в неактивном состоянии. За это отвечают маски логических уровней - ячейки \$1A..\$1E. Ячейки \$1F..\$23 обеспечивают подачу напряжений питания и, если необходимо, напряжения программирования.



Управляющие сигналы - позволяет задать номера выводов, на которых программатор будет формировать сигналы для микросхем с последовательным доступом. Здесь же находится выпадающее меню, которое позволяет задать скорость тактирования. Для большинства микросхем это ячейки \$2C..\$2F. Необходимо учитывать, что "плясать" сигнал будет, исходя из уровня 0 или 1, заданного в начальном состоянии.



Редактор алгоритмов - обычный текстовый редактор, который позволяет написать свой собственный алгоритм работы с микросхемой. При выборе новой микросхемы файл скрипта сбрасывается и, когда открывается окно редактора, программа запускает встроенный дизассемблер, который пытается декомпилировать скрипт. Если микросхема использует один из штатных алгоритмов работы, то в окне редактора будет только команда END. Доступа к этим алгоритмам нет. Если микросхема работает через скрипт, то программа покажет декомпилированный скрипт, который можно подправить и заново скомпилировать. Необходимо отметить, что в скриптах есть секретная область, которая недоступна обычным пользователям и данные в этой области не подлежат декомпиляции. Соответственно, перекомпилировать такой файл обычному пользователю будет невозможно. В комплект поставки программатора входят несколько десятков скриптов, описывающих работу с конкретными микросхемами.



Меню текстового редактора содержит стандартные команды открытия и сохранения файлов, команды копирования и вставки текста, поиска и замены данных. Программа предлагает сохранять файлы скриптов в специальной папке SCRITPS, но при использовании проектов, размещение этих файлов не ограничивается. При перемещении курсора по тексту или при вводе команд с клавиатуры, программа выводит в строке состояния описание команды на которой находится курсор. Вызов справки из этого окна позволяет получить быстрый доступ к описанию всей [системы команд](#). Чтобы скомпилировать скрипт используется кнопка "Компилировать".

Сохранение настроек - сохраняет непосильный труд пользователя в виде новой микросхемы в списках.



Применить настройки. Особого смысла в этой команде нет, поскольку все настройки и так уже находятся в блоке параметров. Но нажатие этой кнопки позволяет сбросить подсветку изменений и начать якобы с нового листа.

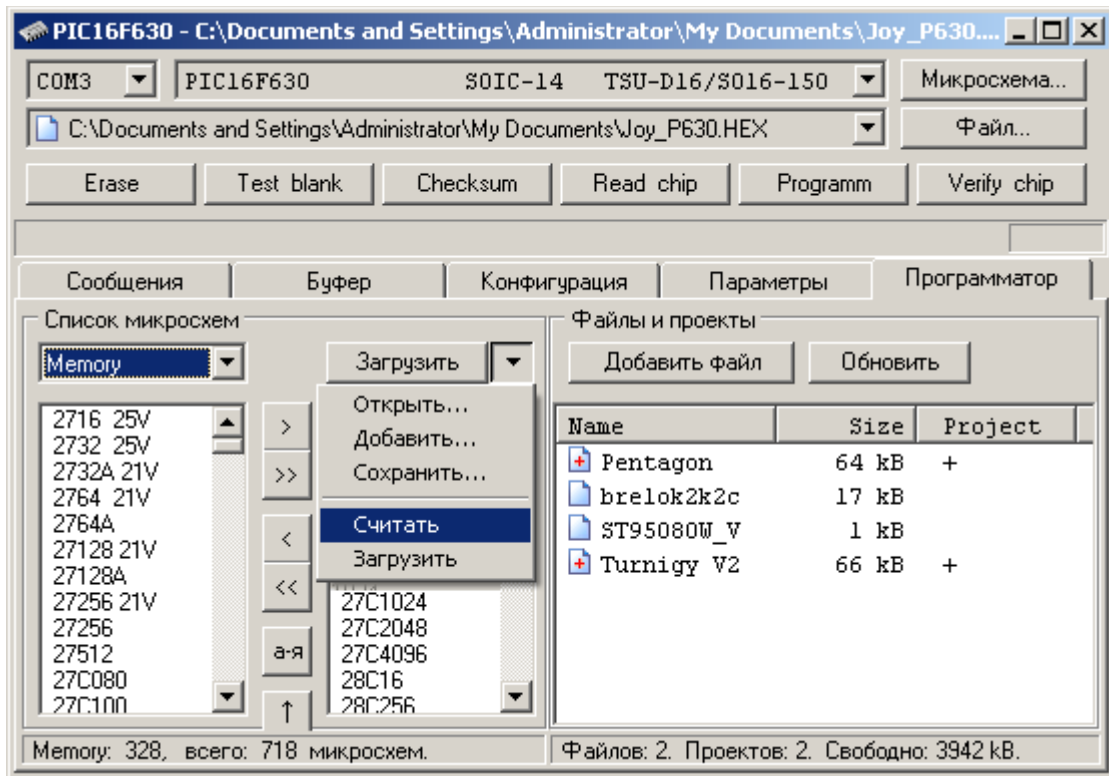
Добавить в список. Выводит диалог для сохранения файла и добавляет новую микросхему в список пользователя.

Вернуть начальные - отменяет все внесенные изменения и возвращает параметры микросхемы в начальное состояние.

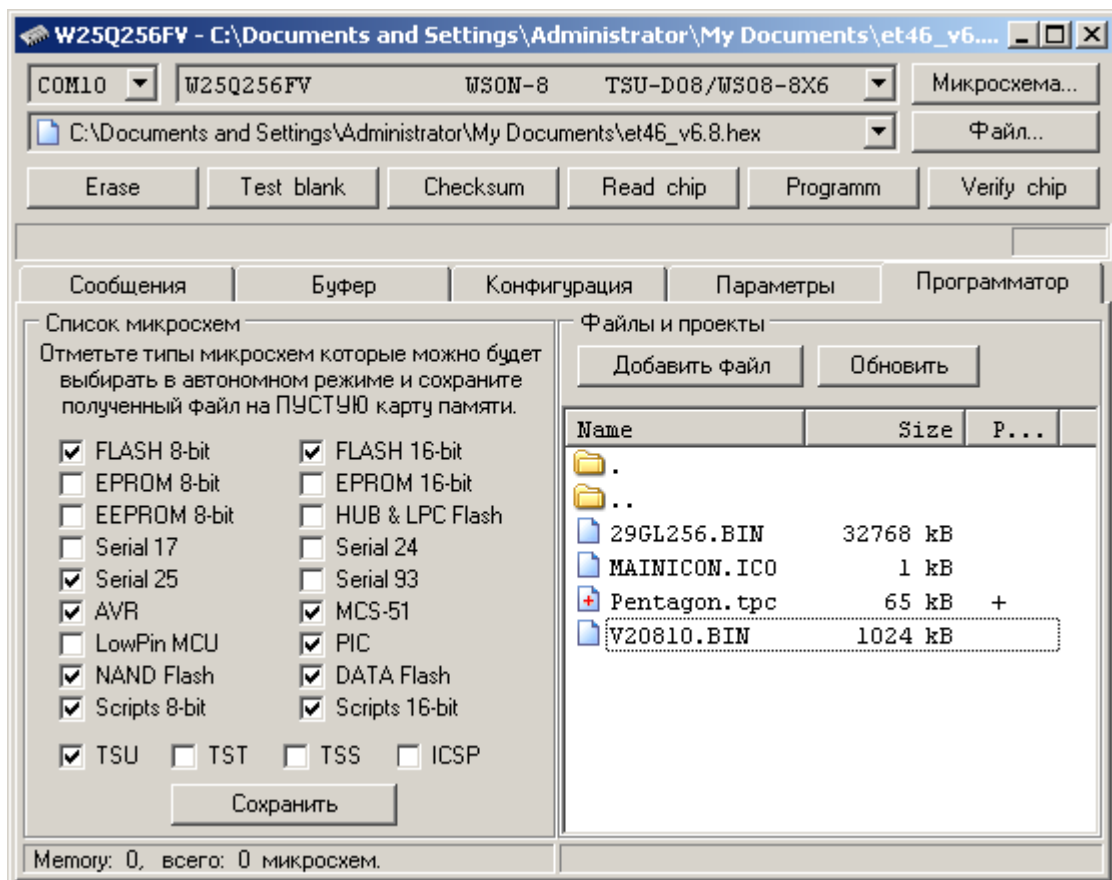
Закладка Программатор

Обеспечивает доступ к внутренней памяти программатора для управления списком микросхем и списком загруженных файлов и проектов. Вид этого окна и режимы работы зависят от модели подключенного программатора.

Для программатора ТРИТОН V5.7TU



Для программатора ТРИТОН V5.8TU



Список микросхем. - формирует и загружает в программатор списки микросхем для работы в автономном режиме.

V5.7TU. В память программатора можно загрузить данные о 1018 микросхемах, которые будут доступны для выбора в

автономном режиме. Кроме того, еще до 256 микросхем могут быть сохранены в памяти программатора в виде [проектов](#). Микросхемы использующие скрипты, могут быть загружены в память программатора только в виде проектов.

Для экономии места в памяти программатора файл микросхем сделан переменной длины. Минимальный объем этого файла равен 1кБ (10 микросхем + 6 названий групп), максимальный – 64кБ (1018 микросхем + 6 групп). Для ускорения поиска и выбора микросхемы список микросхем в программаторе разбит на шесть групп, по типам. Каждая из групп может содержать любое число микросхем или быть пустой. Первые пять групп могут содержать микросхемы только одного типа: это стандартная память EPROM или FLASH; AVR; PIC контроллеры; микросхемы памяти с последовательным доступом и микроконтроллеры MCS-51. Шестая группа - FAVORITES может содержать любые микросхемы, даже уже включенные в другие группы.

Программное обеспечение позволяет считать список микросхем из памяти программатора и сохранить в виде файла на компьютере. А также: открыть файл списка микросхем, объединить несколько списков, отсортировать их по алфавиту, добавить или удалить микросхемы, вручную задать порядок следования микросхем. Новый список микросхем может быть загружен в программатор, взамен старого, или сохранен в виде файла на компьютере.

При редактировании списка микросхем программа не накладывает ограничений на максимальное количество микросхем, но при загрузке такого списка в программатор он будет обрезан по максимальному числу микросхем (1018). Например, если количество микросхем в первых группах превосходит установленный лимит, то все последующие группы будут пустыми.

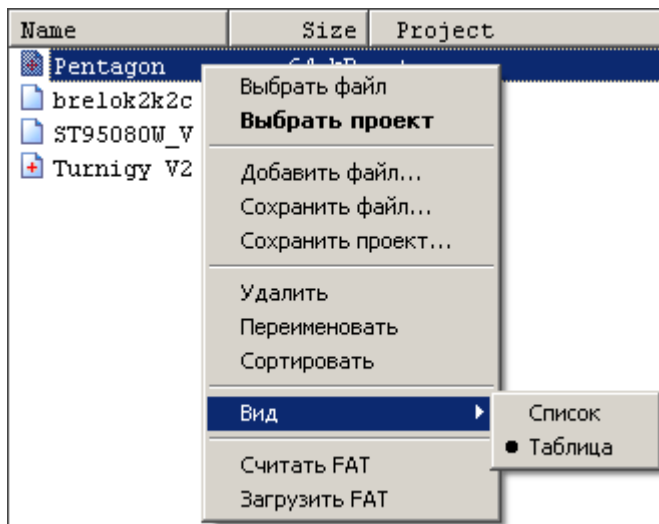
V5.8TU. Поскольку программатор использует карты micro SD, то нет никаких ограничений по количеству и типу микросхем в этом списке. Для ускорения поиска и выбора микросхемы, программа предлагает загрузить только нужные типы микросхем и соответствующие им типы переходных панелей и адаптеров. Список микросхем будет сгруппирован по фирмам.

По команде "Сохранить" надо записать полученный файл CHPLIST.CHC в корневой каталог на пустую карту micro SD, после чего установить карту в программатор.

Файлы и проекты. - обеспечивает управление файлами в памяти программатора.

V5.7TU. В зависимости от модели программатора, в его памяти может быть сохранено до 256 файлов, общим объемом 960кБ (V5.4), 1984кБ (V5.5) или 4032кБ (V5.6 и V5.7). Каждому файлу может быть присвоено собственное имя, длиной не более 10 знаков. Каждый файл может содержать данные о микросхеме, т.е. быть проектом.

V5.8TU. Количество файлов и проектов определяется размером карты micro SD. Программное обеспечение позволяет просматривать содержимое карты, установленной в программаторе, выбирать файлы и проекты для работы в автономном режиме, считывать и сохранять их на компьютере. При работе на компьютере, программа поддерживает структуру каталогов, длинные имена и национальные кодировки. При работе в автономном режиме, программатор поддерживает структуру каталогов, но не поддерживает длинные имена и национальные кодировки. Для корректного отображения на дисплее программатора, имена файлов и проектов не должны превышать 8 знаков и содержать русские буквы.



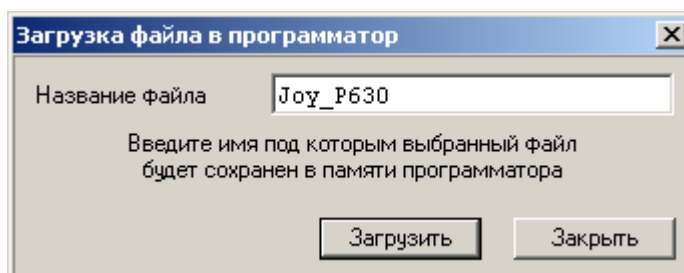
В этом окне имеется контекстное меню, которое позволяет считать или загрузить список файлов и проектов, отсортировать список по алфавиту или задать порядок следования вручную. Кроме того, можно установить (выбрать) нужный файл или проект на программаторе, переименовать, удалить или сохранить его в компьютере.

На компьютере проекты помечаются иконкой с красным знаком плюс. Двойной клик по названию проекта выбирает и устанавливает в программаторе указанный проект. Двойной клик мышкой по названию файла - выбирает файл. При работе в автономном режиме программатор не показывает различия между файлами и проектами. Если на программаторе выбрать файл или проект, через меню "Select: Fil", то программатор только откроет файл и проигнорирует информацию о проекте. Если выбрать проект, через меню "Select: Prj", то программатор загрузит параметры микросхемы и откроет файл. Если через меню "Select: Prj" выбрать файл, то будет открыт только сам файл.

Список может быть отображен в виде обычного списка или в виде таблицы, с указанием размера файла. Клик мышкой по заголовку таблицы позволяет отсортировать список по алфавиту или по объему, а заодно и рассчитать размер свободного места в памяти программатора. Перетаскивание мышкой записей, позволяет задать любую последовательность имен в списке. Чтобы переименовать файл или проект, нужно выделить нужную запись, кликнуть мышкой по ее названию и ввести

новое название. После этих операций необходимо загрузить обновленный список в программатор ("Загрузить FAT").

Добавить файл - выводит окно диалога для загрузки файла в память программатора. Диалог позволяет выбрать файл, присвоить ему имя, которое будет идентифицировать его в списке файлов программатора, и загрузить в память программатора. Перед загрузкой программа преобразует файл в двоичный формат и проверяет объем свободного места в памяти программатора.



При работе в автономном режиме в программаторе действуют следующие ограничения:

- Длина имени файла не должна превышать 10 знаков. Русские буквы на экране программатора отображаются неправильно, поэтому их лучше не использовать.
- Данные для записи в исходном файле должны начинаться с нулевого адреса. В автономном режиме программатор не поддерживает установку начального адреса данных в файле.
- Для микросхем с EEPROM, данные для записи в эту область памяти должны находиться в основном файле программы, в следующем блоке, сразу после последнего записываемого адреса. Например, если начальный адрес равен 000000, конечный 00035F, то данные для записи EEPROM должны быть размещены, начиная с адреса 000400.
- Для PIC контроллеров информация о конфигурационном слове, размещенная в исходном файле, программатором не поддерживается. Конфигурационное слово в программаторе может быть загружено и сохранено только вместе с проектом.

Загружать отдельные файлы в память программатора лучше, когда ведется работа с обычными микросхемами памяти или с микроконтроллерами, которые имеют простой механизм защиты. Если планируется автономная работа с микроконтроллерами, имеющими дополнительные области памяти (такие как EEPROM, Fuses и т.д.), то необходимо использовать команду "[Загрузить проект](#)".

Меню "Удалить" - удаляет файл или проект и обновляет список в программаторе. Время удаления зависит от объема файла, и для 256кБ составляет около 8 секунд.

Если по оценке программы, программатор имеет достаточно свободного места, а при чтении микросхемы или загрузке файла выводится сообщение о недостатке памяти, то необходимо сохранить на компьютере нужные файлы и отформатировать FLASH-диск (нажать все четыре клавиши и включить питание программатора). После чего снова загрузить список микросхем, необходимые файлы и проекты.

Форматирование памяти программатора удаляет все списки микросхем.

Настройка программы

Данная глава носит технический характер и предназначена для подготовленных пользователей. Это подразумевает, что вы знакомы с основами программирования, имеете представление об ассемблере и основах языка си.

Программное обеспечение позволяет не только менять параметры существующих микросхем, но и дает возможность добавлять совершенно новые микросхемы. Для этого в программе имеется редактор скриптов, а в программатор встроен виртуальный процессор, способный исполнять загруженный скрипт. Для обеспечения дружественного интерфейса с пользователем, параметры микросхем настраиваются с помощью конфигурационных файлов. А файл прошивки можно менять с помощью файлов-калькуляторов.

Блок параметров микросхемы

Блок параметров содержит всю информацию, необходимую программатору для работы с микросхемой и загружается в программатор при каждой команде с компьютера или хранится в загруженном проекте. Все параметры, которые могут потребоваться для настройки конфигурации микросхемы или режимов работы с ней, выведены на закладку [Конфигурация](#) и их можно менять. Изменение специальных параметров, таких как описание выводов микросхемы или алгоритм работы можно менять только на закладке [Параметры](#) микросхемы.

Необдуманное изменение параметров может привести к неправильной работе программатора и даже повредить микросхему.

Ниже описываются параметры, общие для всех микросхем. Назначение некоторых байт может меняться, в зависимости от выбранной микросхемы. Описание параметров каждой микросхемы выводится в виде подсказки при перемещении курсора или при клике мышкой в окне параметров.

\$00..\$02	-	Сигнатура микросхемы (3 байта)
\$03..\$0D	-	Название микросхемы (11 байт)
\$0E	+	Напряжение питания во время записи
\$0F	+	Номинальное напряжение питания во время чтения
\$10	+	Длительность импульса записи (объём блока)
\$11	!	Байт флагов микросхемы
\$12	x	Алгоритм программирования или стирания
\$13	+	Минимальное напряжение питания
\$14	+	Максимальное напряжение питания
\$15	+	Напряжение программирования
\$16	+	Количество импульсов записи (Размер страницы)
\$17	!	Кол-во бит защиты (конфигурационных слов)
\$18	+	Длительность задержки при включении питания
\$19	+	Напряжение стирания
\$1A..\$1E	!	Маски логических уровней (5 байт)
\$1F..\$23	!	Маски VPP и VCC (5 байт)
\$24..\$25	+	Конфигурационное слово микросхемы
\$26..\$2F	!	Индивидуальные параметры микросхемы
\$30..\$3F	!	Индивидуальные параметры микросхемы
\$40..\$42	+	Начальный адрес микросхемы
\$43..\$45	+	Конечный адрес микросхемы
\$46..\$48	+	Адрес данных буфере (Смещение)
\$49..\$4B	x	Зарезервировано для дальнейшего использования
\$4C..\$4E	+	Начальный адрес EEPROM
\$4F..\$51	+	Конечный адрес EEPROM
\$52..\$54	+	Начальный адрес EEPROM в буфере
\$55	x	Адрес регистра DAT_WR
\$56	+	Начальный адрес EEPROM в ОЗУ (Ext Byte)
\$57	x	Адрес регистра DAT_CK
\$58	!	Флаги режимов работы – устанавливаются вначале каждой команде
\$59	x	Номер версии, в которой был создан проект
\$5A	x	Номер используемой панели
\$5B..\$5E	!	Индивидуальные параметры микросхемы
\$5F	!	Флаги программатора
\$60..\$6F	x	Параметры файла (используется в автономном режиме)
\$70..\$77	x	Служебные регистры (используется при работе с микросхемой)

\$78..\$7F	!	Адреса сигналов управления (только для программатора V5.8T)
\$80..\$9F	!	Адреса размещения шины данных D15..D0 при записи и чтении
\$A0..\$B7	!	Адреса размещения шины адреса A23..A0
\$B8..\$BA	x	Максимальный адрес микросхемы
\$BB	!	Флаги пропуска режимов работы
\$BC	x	Test Blank Mask_L
\$BD	x	Test Blank Mask_H
\$BE..\$BF	x	Зарезервировано для дальнейшего использования
\$C0..\$17F	x	Скрипт, описывающий работу с микросхемой
	-	Этот параметр не влияет на работу с микросхемой
	+	Можно менять, часть этих параметров вынесена на панель управления
	!	Можно менять, определяет режим работы с микросхемой
	x	Менять этот параметр ЗАПРЕЩЕНО! Можно повредить микросхему

Формат файла микросхем

Параметры всех микросхем содержатся в двух файлах: TRITON.CHS и TRITON.CHP. Эти файлы поставляются и обновляются с каждой версией программы и содержат информацию обо всех поддерживаемых микросхемах. Для самостоятельного расширения списка микросхем или для хранения микросхем с измененными параметрами, используются файлы *.CHS, помещаемые в папку SCRIPTS.

При добавлении нового файла *.CHS новые микросхемы будут доступны только при следующем запуске программы.

Файлы *.CHS - это обычные бинарные файлы, которые содержат полный [блок параметров микросхемы](#). Файлы, создаваемые пользователем, могут иметь любое имя и содержать параметры одной или более микросхем. В автономном режиме микросхемы из этих файлов могут работать только в виде проектов. В параметрах каждой микросхемы по адресу \$59 должен находиться номер версии, в шестнадцатеричном формате, которой должен совпадать с номером версии ПО, а по адресу \$5A - номер используемого адаптера.

Файл TRITON.CHP содержит блок параметров микросхемы, только в упакованном виде. Этот файл используется при формировании списков микросхем для автономного режима на программаторах V5.7T. В последующих версиях программы планируется отказаться от этого файла и хранить настройки микросхем для автономного режима в виде файлов *.CHS.

В файлах *.CHS и *.CHP существуют два типа записей: название фирмы-производителя или группы микросхем и собственно данные о микросхеме. Размер каждой записи в файле *.CHS равен 384 байта, а в файле *.CHP = 128 байт. Первый байт в названии группы должен быть \$FF, длина названия не более 30 знаков. Первый байт в параметрах микросхемы, в большинстве случаев, это код фирмы-производителя, может иметь любое значение кроме \$FF. При запуске программы, в процессе загрузки, если считывается код \$FF, то создается новая закладка с данным именем, в которую начинают записываться названия микросхем. Количество наименований фирм в этих файлах, также как и число микросхем в каждой фирме может быть любым.

При запуске сначала загружается содержимое файла TRITON.CHS, после чего программа пытается загрузить файлы *.CHS из папки SCRIPTS. Проверка на совпадения имен микросхем не производится и, в случае совпадения, в окне выбора микросхем будут две микросхемы с одинаковыми именами. При этом программа позволяет выбрать каждую из этих микросхем. Однако, при последующем выборе такой микросхемы из списка открытых микросхем, в результате поиска будет выбрана та микросхема, которая расположена ближе к началу файла. Поэтому, во избежание возможной путаницы, лучше всегда сохранять микросхему с уникальным именем.

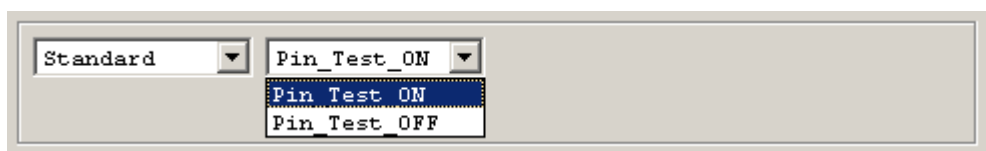
Формат файла конфигурации

Внешний вид закладки [Конфигурация](#) может быть настроен с помощью конфигурационных файлов *.CFG. В наследство от старой версии программы осталось несколько конфигурационных файлов которые имеют разный формат. Это файлы "Avg.cfg" и "Flash.cfg". Описание их можно посмотреть в самих файлах. Остальные файлы имеют одинаковый формат, который описан ниже. При необходимости эти файлы могут быть отредактированы пользователем, но необходимо учитывать, что при обновлении версии ПО эти файлы будут заменены новыми.

Для самостоятельной настройки конфигурации микросхем используются файлы с расширением ".CFG", помещаемые в папку SCRIPTS. Название каждого файла должно точно совпадать с именем выбранной микросхемы. Каждый файл должен содержать две секции. Обязательная секция [CHIPS], в которой указано название микросхемы и ссылка на секцию параметров. Секция параметров может иметь любое название и содержит список элементов управления, позицию размещения на экране, описание элемента на английском и/или русском языке, адрес в блоке параметров и сами значения.

Пример файла конфигурации для микросхемы W27C512P и результат на экране.

```
=====
;
[CHIPS]
W27C512P=W512
;
=====
[W512]
L00=Algo Select|Выбор алгоритма
_Script=121F
_Standard=1201
L01=Check DataBus|Проверка шины данных
_Pin_Test_ON=BB00
_Pin_Test_OFF=BB80
;
=====
```



Полный список элементов управления:

[Wxx](#) - Флаг режима работы (ширина=220) Стоит галочка =1; сброшена =0.

[Cxx](#) - Флажок (ширина=110). Стоит галочка =0; сброшена =1.

[Sxx](#) - Флажок (ширина= 55). Стоит галочка =0; сброшена =1.

[Gxx](#) - Группа из 8 флажков "Sxx", расположенных в одном байте.

[Lxx](#) - Список из нескольких параметров (8, 16, 24, 32 бит).

[Exx](#) - Текстовое поле-набор шестнадцатеричных значений(max 16 байт).

[Axx](#) - Текстовое поле-ввод адреса (max 4 байта, обратный порядок).

[Txx](#) - Текстовое сообщение (ширина=от "xx" до конца окна. Max=440).

[Bxx](#) - Кнопка. Используется в режиме LOCK. Устанавливает \$58.7=0

[Uxx](#) - Кнопка. Используется в режиме LOCK. Устанавливает \$58.7=1

xx - положение в окне (DEC. 0..3-первая строка, 4..7-вторая,...).

+x - вывод в последнюю строку (0..3-текущая строка, 4-следующая).

[REE](#) - Название: EEPROM, LDR0M, Encryption, Config Zone...

[RTO](#) - Таймаут для режимов чтения/записи при работе через USB (2..9сек).

[REC](#) - Задаёт алгоритм коррекции считываемых из микросхемы данных.

[RLI](#) - Название библиотеки для управления программатором через MPSSE.

[Rss](#) - Названия выводов для разъема ISP-CONN и переходника TSH-ICSP.

ss - код вывода [CS, WR, RD, CK, VP, IO].

[RFs](#) - Управление размещением конфигурационной области в блоке параметров.

s - Начальный адрес, размер и значения [A, S, V].

[Nss](#) - Отключает кнопки режимов работы, которых нет в микросхеме.

ss - название режима [TB, CS, RD, PG, VR].

[Fzz](#) - Для алгоритмов \$1C..\$1F - вывод штатной панели конфигурации.

zz - номер алгоритма (HEX).

[Hzz](#) - Подсказка (описание параметра) в блоке параметров микросхемы.

zz - адрес ячейки в блоке параметров (\$00..\$BF).

[Mzz](#) - Маска конфигурации (исп., когда часть битов идут записанными)

zz - начальный адрес маски (HEX) в блоке параметров микросхемы.

[P0x](#) - Выбор OTP страницы. Элемент выводится на закладке Буфер.

x - число страниц.

Dssx- Загрузка данных из файла в параметры микросхемы.

ss - код источника и приемника [F-основной файл, E-eeprom, P-параметры].

x - L - код Конфигурации и битов защиты; F - код Fuses, ID_Loc...

x - десятичное число.

z - шестнадцатеричное число.

s - буквенный параметр.

Элемент управления задается **заглавной буквой** и сразу после неё следует двузначный десятичный номер позиции на экране (xx), затем знак равно и описание параметра, которое будет выводиться в строке состояния при наведении курсора на этот элемент. Позиция на экране рассчитывается построчно, начиная с нуля, по 4 элемента строке (00..03-первая строка, 04..07-вторая,...). Исключение составляют короткий флажок "Sxx" и "Gxx", которые выводятся по 8 элементов в строке (00..07-первая строка, 08..15-вторая,...).

+x - выводит элемент или текстовое сообщение на "x/4" строк ниже самого последнего элемента. Значения +0..+3 - 1,2,3 и 4 позиция в текущей (последней) строке. +4 - следующая строка. Чтобы не допустить наложения, данная эта команда должна быть последней в списке или как дополнительный параметр.

В приведенных ниже примерах имеются комментарии. В рабочих файлах комментарии должны находиться только с начала строки. Комментарии в конце строки параметров недопустимы. Ни один параметр внутри секции не должен повторяться.

Wxx - флаг режима работы (ширина=220, два элемента в строке). Стоит галочка =1; сброшена =0. При установленном флаге разрешается доступ к таким режимам работы, как FUSE и LOCK bits. Данный элемент должен указывать только на \$58 ячейку в блоке параметров и модифицировать только 2 и 3 биты в этой ячейке (=5804, =5808). Кроме того, этот элемент может вывести в заданном порядке шестнадцатеричные значения до 10 ячеек из блока параметров.

```
-----  
W00=Enable Read/Write FUSE bits|Разрешение чтения/записи FUSE bits  
_FUSE =5804  
W02=Enable Read/Write LOCK bits|Разрешение чтения/записи LOCK bits  
_LOCK (%.2x%.2x)=58082524 ;показывает содержимое ячеек 24 и 25.  
-----
```

Sxx - флажок для управления одним или несколькими битами в ячейке (ширина элемента=110). Стоит галочка - выбранные биты сбрасываются в ноль. Сброшена, биты устанавливаются в единицу.

Sxx - флажок для управления одним или несколькими битами в ячейке (ширина элемента=55). Позволяет вывести до 8 элементов в строке, для модификации каждого бита в байте. Стоит галочка - выбранный бит сбрасывается в ноль. Сброшена, бит устанавливается в единицу.

Gxx - группа из 8 флажков, для управления каждым битом в одной ячейке (ширина элемента=55). Стоит галочка - выбранный бит сбрасывается в ноль. Сброшена, бит устанавливается в единицу.

```
-----  
C04=English Hint.|Подсказка на Русском.  
_Flags_1=2583 ;25-адрес ячейки; 83-маска битов.  
S10=English Hint.|Подсказка на Русском.  
_Bit_0 =2601  
S11=English Hint.|Подсказка на Русском.  
_Bit_1 =2602  
G16=24:B7,B6,B5,B4,,,B0|Описание Бита 7,Бит 6,Бит 5,Бит 4  
;16 - позиция первого флага на экране.  
;24 - адрес ячейки.  
;Vx - название бита для вывода на экран (нет названия - нет элемента на экране)  
;После "|" может идти необязательное описание каждого бита.  
-----
```

Lxx - список из нескольких параметров. Может модифицировать расположенные подряд от 1 до 4 байт в блоке параметров, начиная с указанного адреса. Адрес должен указывать на младший байт. Размещение данных в блоке параметров: младший, средний, старший... Длина списка может быть любой, каждый параметр должен иметь уникальное название.

```
-----  
L06=English Hint.|Подсказка на Русском.  
_LP_OSC =243FFC ;3F - старший байт, FC - младший байт  
_XT_OSC =243FFD ;По адресу 24 находится младший байт  
_RC_OSC =243FFE ;По адресу 25 находится старший байт  
_HS_OSC =243FFF ;неиспользуемые биты должны быть =1  
-----
```

P0x - Выбор страницы EEPROM на закладке Буфер. Позволяет изменить до 4 байт, для установки начального и конечного адресов страницы. Формат команда аналогичен команде "Lxx". Отличие в том, что число задает не позицию на экране, а число страниц в списке.

```
-----  
P04=Select a page to work with|Выбор страницы для работы  
_Page 0 =4D00FF0000  
_Page 1 =4D10FF0010  
_Page 2 =4D20FF0020  
_Page 3 =4D30FF0030  
-----
```

Exx - текстовое поле - набор шестнадцатеричных значений. Позволяет модифицировать до 16 байт в блоке параметров, расположенные подряд, начиная с указанного адреса. Первый байт в редактора соответствует заданному адресу, второй - адрес+1... Если число байт в поле более 7, то ширина элемента увеличивается вправо.

```
-----  
E08=English Hint.|Подсказка на Русском.  
_X01 =3008 ;30-адрес ячейки, 08-число байт  
-----
```

Axx - текстовое поле для ввода адреса в шестнадцатеричном формате (max 4 байта). В отличии от предыдущего элемента, адрес располагается в блоке параметров в обратном порядке (мл.ср.ст).

```
-----  
A09=English Hint.|Подсказка на Русском.  
_Y01 =4303 ;43-адрес ячейки, 03-число байт (мл.ср.ст)  
-----
```

Txx - выводит на экран текстовое сообщение. Ширина элемента зависит от длины текста, начинается с позиции "xx" и продолжается до конца экрана. Высота элемента зависит от длины сообщения. Для длинных сообщений программа формирует многострочный текст, в зависимости от ширины окна. Для принудительного перевода строки используется символ "~".

```
-----  
T12=Text message.~New string.|Текстовое сообщение.~Новая строка.  
-----
```

Vxx - кнопка. Используется для запуска режима LOCK. Устанавливает \$58.7=0. Отдельная команда для установки защиты.

Uxx - кнопка. Используется для запуска альтернативного режима LOCK. Устанавливает \$58.7=1. Может использоваться для записи FUSES или снятия защиты. Для этого скрипт или штатная подпрограмма должна иметь два алгоритма и уметь анализировать бит 7 в ячейке \$58. Команды **JWL** или **JWH**.

```
-----  
B10=Protect  
U11=UnProtect  
-----
```

REE или **R00** - выводит название для второго редактора на закладке [Буфер](#) (EEPROM, LDROM, Encryption, Config Zone...).

```
-----  
REE=EEPROM|ЕЕПРОМ  
-----
```

RTO - задает таймаут для режимов чтения и записи при подключении программатора через USB. По умолчанию таймаут = 3 сек. Диапазон допустимых значений от 2 до 9 секунд.

```
-----  
RTO=5  
-----
```

REC - задает алгоритм коррекции считываемых из микросхемы данных. Параметр находится в стадии разработки.

```
-----  
REC=80  
-----
```

RLL - задает имя подгружаемой библиотеки. Библиотека должна находиться в папке, где находится основная программа. Только для программатора V5.8TU.

```
-----  
RLL=USER.DLL  
-----
```

Rss - позволяет задать названия выводов для ISP-CONN или переходника TSH-ICSP и названия управляющих сигналов на закладке [Параметры](#). "ss" - строковый параметр, состоящий из заглавных букв. Допустимые значения: **CS, WR, RD, CK, VP, IO**.

RCS=RES
RWR=MOSI
RRD=MISO
RCK=SCK

RFs - управляет размещением нестандартной конфигурационной области в блоке параметров. Под конфигурационные данные в параметрах отведены две области: \$24..\$2B и \$30..\$3D. Данная команда позволяет использовать еще одну область с адресами \$60..\$6C.

- RFA - Начальный адрес конфигурационной области в блоке параметров = \$60.
- RFS - Размер конфигурационной области в блоке параметров [\$01..\$0C].
- RFV - Значения конфигурации, которые будут установлены при выборе микросхемы.

RFA=60
RFS=04
RFV=010203FF

Nss - Отключает в программе кнопки режимов работы, которых нет в микросхеме.

NTB=No Test Blank
NCS=No CheckSum
NRD=No Read
NPG=No Programm
NVR=No Verify

Fzz - для скриптовых алгоритмов \$1C, \$1D, \$1E и \$1F, выводит штатную панель конфигурации. "zz" - номер алгоритма (HEX). Например, для микросхемы AVR, поддерживаемой через скрипты, чтобы не писать отдельный файл конфигурации, можно вызвать штатные настройки, указав "F0A".

Допустимые номера алгоритмов:

- F03..F05: - Flash 8-16 bit;
- F15, F16: - Flash 8-16 bit;
- F07: - Serial EEPROM 25;
- F09: - Serial EEPROM 93;
- F0A: - AVR;
- F0E, F0F: - Pic16xxx;
- F11: - PIC18, PIC24, PIC30, dsPIC;

F0A=Вывод конфигурации AVR

Hzz - подсказка (описание параметра) в блоке параметров микросхемы. "zz" - шестнадцатеричный адрес ячейки в блоке параметров (\$00..\$BF).

H24=Configuration word (Low)|Конфигурационное слово, младший байт.
H2C=Адрес сигнала RESET
H2D=Адрес сигнала MOSI
H2E=Адрес сигнала MISO
H2F=Адрес сигнала SCK

Mzz - маска конфигурации. Не оконный элемент. Сделана специально для работы с PIC контроллерами, когда часть битов в конфигурационном слове идут записанными. "zz" - начальный адрес маски (HEX) в блоке параметров микросхемы.

M24=BF3F ;24-адрес ячейки(HEX); BF-первый байт; 3F-второй...

Dssx - загрузка (копирование) данных из файла в параметры микросхемы. Спецификации некоторых микросхем описывают формат размещения конфигурационных данных в исходном файле. Данная команда позволяет считать эти данные из файла и разместить их в заданные ячейки в блоке параметров микросхемы. При сохранении файла программа

выполняет обратное действие и записывает данные из этих областей по указанным адресам в файл. На самом деле, данная команда более универсальна и позволяет скопировать блок данных любой длины внутри или между исходным файлом, EEPROM и параметрами микросхемы. Если количество байт или заданный адрес превышают размер указанного буфера, то данная команда не выполняется.

Формат команды: Dss*=xxxxxxx,yyyyyy,zzzzzzz

- "ss" - первый символ - код источника, второй - код приемника [F-основной файл, E-еергом, P-параметры].
- "*" - любая буква или цифра, обеспечивающая уникальность параметра внутри секции.
- "xx" - адрес начала данных в источнике (HEX).
- "yy" - адрес начала данных в приемнике (HEX).
- "zz" - количество байт для копирования (HEX).

DFPL - загрузка конфигурационного слова и установка флага доступа к CFG Words (ячейка \$58 бит 3).

DFPF - загрузка FUSEs или ID_Locations и установка соответствующего флага (ячейка \$58 бит 2).

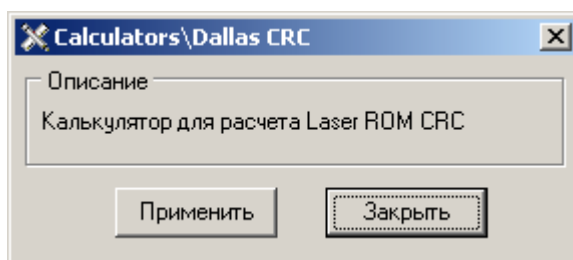
DFE - загрузка EEPROM и установка флага доступа к EEPROM (ячейка \$58 бит 1), если не выбран отдельный файл. В этой команде количество байт для копирования может быть равно нулю. В этом случае, размер загружаемого блока будет установлен равным размеру EEPROM области выбранной микросхемы.

Команды **DFE**, **DFPL** и **DFPF** после выбора микросхемы и открытия файла, считывают из файла все данные и устанавливают соответствующие флаги, при условии, что указанный адрес находится внутри файла. При перезагрузке файла (например, перед циклом программирования), загружаются только те данные, доступ к которым разрешен флагами, установленными в окне проекта. Аналогично, при сохранении файла будут записаны только разрешенные области.

```
-----  
;Загрузка конфигурационного слова, ID-location и EEPROM для PIC16Fxxx  
DFPL=400E,24,2 ;адрес в файле, адрес ячейки, количество байт  
DFPF=00004000,34,8  
DFE=4200,0,200 ;адрес в файле, адрес EEPROM, длина(256 слов)  
-----
```

Формат файла калькулятора

Шестнадцатеричный код, представленный в редакторе, не всегда удобен для выполнения расчетов, пакетной обработки данных или правки считанного дампа под конкретное оборудование. Калькулятор позволяет выполнить над этими данными определенные действия и вывести их в виде стандартных элементов управления Windows. В диалоговом окне пользователь может изменить нужные ему параметры, после чего калькулятор обработает и сохранит их в исходном файле. Если вмешательство пользователя не требуется, то программа позволяет запускать калькулятор в фоновом режиме и автоматически вносить нужные изменения перед записью каждой микросхемы.



Файл калькулятора представляет собой обычный текстовый файл, содержащий C-подобный скрипт.

```
[Calculators\Dallas CRC]
Size(280,100);
Form{
  Group("Описание",5,5,270,50)
  {
    Label(lb1,"",5,20);
  }
}
OnShow{
  lb1 = Калькулятор для расчета 8-битной CRC;
}
OnApply{
  Var0 = 0;
  @Var0 = @Var0 + 1;
  if (@Var0 = 0){@0x1 = @0x1 + 1}
  CRC = 0;
  while (Var0 < 0x07) {
    inp = CRC ^ @Var0;
    CRC = 0;
    if((inp & 0x1) != 0) { CRC = CRC ^ 0x5e; }
    if((inp & 0x2) != 0) { CRC = CRC ^ 0xbc; }
    if((inp & 0x4) != 0) { CRC = CRC ^ 0x61; }
    if((inp & 0x8) != 0) { CRC = CRC ^ 0xc2; }
    if((inp & 0x10) != 0) { CRC = CRC ^ 0x9d; }
    if((inp & 0x20) != 0) { CRC = CRC ^ 0x23; }
    if((inp & 0x40) != 0) { CRC = CRC ^ 0x46; }
    if((inp & 0x80) != 0) { CRC = CRC ^ 0x8c; }
    Var0 = Var0 + 1;
  }
  lb1 = "Сумма = " + #b.crc;
  @Var0 = CRC;
}
```

Для совместимости с другими проектами, первая строка файла должна содержать название файла, заключенное в квадратные скобки. Название должно начинаться с начала строки, без ведущих или ведомых пробелов. Между квадратными скобками и первым и последним символом названия также не должно быть пробелов.

Далее указывается размер клиентской области окна: **Size(280,100)**; (ширина, высота), в которой могут находиться элементы управления, описываемые в секции "Form".

Секция **FORM** содержит список элементов диалогового окна. Допустимо использовать только указанные названия компонентов. Регистр букв не важен. Другие компоненты не поддерживаются.

```
GROUP ("Caption", Left, Top, Width, Height) {...}
LABEL (Name, "Caption", Left, Top);
CHECKBOX (Name, "Caption", Left, Top);
COMBOBOX (Name, "Caption", Left, Top, Width, "Items0", "Items1", ...);
DIGIT (Name, "Caption", Left, Top, Width);
HEXDIGIT (Name, "Caption", Left, Top, Width, MaxLength);
PICTURE (Name, "FileName.bmp", Left, Top, 0, 0);
```

Элементы представляют собой обычные оконные компоненты Windows и не нуждаются в особом описании.

- Name - уникальный идентификатор, позволяющей обращаться к элементу из других секций.
- "Caption" - выводимый в элементе текст.
- Left, Top - координаты левого верхнего угла элемента.
- Width, Height - ширина и высота элемента.
- "Items*" - содержимое списка элемента COMBOBOX.
- MaxLength - количество символов выводимое в шестнадцатеричном редакторе.

Секция **OnShow** содержит код, который будет исполняться в момент вывода диалога на экран. Секция **OnApply** содержит код, который будет исполняться при нажатии пользователем кнопки Ok. Содержимое этих секций представляет собой сильно упрощенный синтаксис языка си. В отличие от других проектов, программа не накладывает ограничения на имена переменных и на использование в выражениях оконных переменных, не привязывает тип переменной к ее имени, понимает отрицательный результат операции, позволяет выполнять операции непосредственно с дампом памяти и идентификаторами компонентов. Также имеет расширенные команды форматирования строки.

Компилятор поддерживает только два оператора:

- **IF** - if(True){выполняется блок А}else{нет, блок Б}.
- **WHILE** - while(True){повторяется блок}.

Доступны следующие арифметические и логические операторы:

+ - * / % & | ^ << >>

Операции сравнения:

= != <= >= < >

Приоритет операций задается только скобками. Оператор "@" позволяет обратиться к содержимому буфера.

```
Var1 = 12; //Данные можно вводить в десятичном
Var2 = 0x0B; // или в шестнадцатеричном формате
adr1 = @0x0010; //Прямое обращение к дампу памяти
@Var1 = @Var2 + adr1 - 1; //Косвенное, через адрес, содержащийся в Var1
lb1 = Var1; //Стандартный вывод данных на экран.
lb1 = Var1 + " = " + #b.@Var1; //А можно и так. (в других проектах работать не
будет)
...
lb1 = "Текст"; //Оформление текстовых сообщений:
lb1.bold = 1; // жирный шрифт
lb1.color = 0xFF0000; // цвет шрифта
...
lb1 = #b.D1 + " = " + #i.D1; //Команды форматирования строки:
lb1 = #b.D1 + " = " + #c.D1; // i=256; c="я";
lb1 = "WORD = " + #h.D1; // b=FF; h=00FF; d=000000FF;
```

В дополнение к "штатным" функциям тритоновский компилятор поддерживает вызов скоростных подпрограмм для работы с блоками данных.

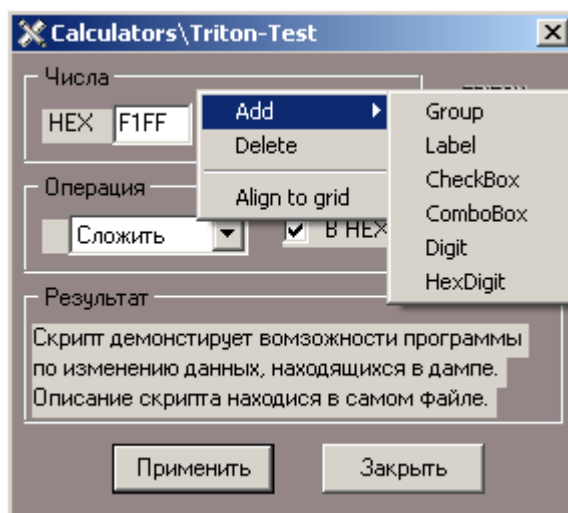
```
BlockCopy(Begin, Length, Dest);
BlockMask(Begin, Length, Mode, Mask);
BlockDelete(Begin, Length);
BlockInsert(Begin, Length, Fill);
```

В качестве параметров могут выступать как константы, так и переменные, включая имена оконных компонентов.

- Begin - Начальный адрес блока.
- Length - Длина блока.
- Dest - Адрес назначения.
- Fill - Байт для заливки (0x00..0xFF).
- Mask - Маска для выполнения логических операций или заливки. Длина от 1 до 4 байт.
- Mode - Режим работы (0..19):
 - 0 - AND. Логическое И.
 - 1 - OR. Логическое ИЛИ.
 - 2 - XOR. Логическое исключающее ИЛИ.
 - 3 - SHR. Сдвиг всего блока на 1 бит вправо. 0 бит предыдущего байта записывается в 7 бит следующего.
 - 4 - SHL. Сдвиг всего блока на 1 бит влево. 7 бит следующего байта записывается в 0 бит предыдущего.
 - 5 - NAND. Логическое И-НЕ.
 - 6 - NOR. Логическое ИЛИ-НЕ.
 - 7 - XNOR. Логическое исключающее ИЛИ-НЕ.
 - 8 - ROTR. Циклический сдвиг каждого байта вправо.
 - 9 - ROTL. Циклический сдвиг каждого байта влево.
 - 10 - NOT. Иверсия данных.
 - 11 - FILL. Циклическая заливка блока значениями "Mask".
 - 12 - SWAP NIBBLE. Перестановка местами тетрад в каждом байте.
 - 13 - SWAP BYTE. Перестановка местами четных и нечетных байт в блоке.

- 14 - SWAP WORD. Перестановка местами 16-битных слов в блоке.
- 15 - COMPARE. Сравнение блока со значениями "Mask".
- 16 - RANDOM. Заполнение блока случайными значениями.
- 17 - REVERSE. Изменение порядка бит в байте. 0 -> 7, 1 -> 6, ..., 7 -> 0.
- 18 - ADD. Сложение.
- 19 - SUB. Вычитание.

Чтобы упростить процесс разработки, в программе имеется визуальный конструктор оконных форм калькулятора. Вход в режим редактирования и выход из него находятся в системном меню окна калькулятора. Редактор позволяет добавлять и удалять компоненты, выравнивать, менять положение и размеры, редактировать текст. При выходе из режима редактирования, данные сохраняются в текстовом виде в Буфере обмена для последующей вставки в файл калькулятора.



Для добавления компонента, установите курсор в нужное место, нажмите правую кнопку мыши и выберите нужный компонент из списка. Перемещение компонента осуществляется с помощью мыши. Для изменения размера используется правый нижний угол компонента. Клик левой кнопкой мышки на компоненте, позволяет изменить в нем текст. Для удаления текста используется клавиша BACKSPACE. Клик правой кнопкой мыши позволяет удалить, выровнять по сетке или добавить новый компонент.

Компонент GROUP является контейнером для размещенных в нем компонентов. При выравнивании или удалении этого компонента, данное действие будет применено ко всем находящимся в нем компонентам. Программа поддерживает многоуровневые вложения компонента GROUP.

Виртуальный процессор

В дополнение к стандартным алгоритмам программирования, программатор имеет виртуальный процессор, позволяющий пользователю самостоятельно написать произвольный алгоритм работы с микросхемой. На практике это означает, что программатор может работать с любой микросхемой, которая проходит по электрическим характеристикам и "попадает" в ключи (Vcc и Vpp) в панельке программатора. Система команд разработана с учетом особенностей программирования микросхем, и позволяет писать простой и эффективный код, обеспечивающий высокую скорость работы с микросхемой.

Исходный код программы это обычный текстовый файл, который содержит последовательность команд, описывающих алгоритм чтения или записи конкретной микросхемы. С помощью встроенного в оболочку компилятора этот файл проверяется на наличие ошибок, компилируется и загружается по определенному адресу в блок параметров микросхемы. В начале каждого цикла компьютер передает в программатор код режима работы и загружает блок параметров микросхемы. В соответствии с заданными параметрами, программатор устанавливает конфигурацию выводов панельки и начинает считывать и исполнять команды загруженного скрипта. После завершения цикла программатор отключает питание микросхемы и передает в компьютер результаты работы.

Скрипт представляет собой сильно упрощенный специализированный микроассемблер, заточенный под ПО программатора и ориентированный на работу с микросхемами. Кроме простейших арифметико-логических операций, операций проверки условий или управления выводами панельки, система команд содержит команды чтения/записи байта, слова или блока данных в последовательном или параллельном режимах, команды обработки считанных данных, а также комбинированные команды, выполняющие несколько операций.

Под скрипт в параметрах микросхемы отведено 192 байта. Это может показаться ничтожно мало, но сейчас в программаторе более 300 скриптов и пока этого объема вполне хватает. Вот два примера скрипта, читающих содержимое микросхемы W25Qxx:

```
=====
READ: BGS ;Включить питание
      FLP ;Открыть файл
      CLR CS
      LDI $03 ;Команда: READ_Chip
      SWL
      LDS ADR_H ;Адрес
      SWL
      LDS ADR_M ;Адрес
      SWL
      LDS ADR_L ;Адрес
      SWL
L01: SRL ;Чтение байта
      CPB
      JNA L01 ;Следующий адрес
      SET CS
      END ;Завершить цикл
=====
или
=====
READ: BGR ;Включить питание
      FLP ;Открыть файл
      SPI_ADR $03 ;Установить Команду и Адрес
      SPI_RDA ;Считать содержимое
      END ;Завершить цикл
=====
```

Скрипт позволяет обратиться к первым 128 ячейкам в блоке параметров (\$00..\$7F), где находятся основные параметры микросхемы. Начиная с адреса \$80 размещаются шины адреса и данных и сам скрипт. Доступ из скрипта к этой области закрыт. В памяти программатора блок параметров микросхемы расположен по адресу \$80. Поэтому, в командах адрес ячейки необходимо увеличивать на \$80. Например, для проверки считанной сигнатуры, расположенной в блоке параметров по адресу \$00..\$01, используются команды **JRN \$80,Label** и **JRN \$81,Label**.

Описание. В каждой строке должна находиться только одна команда. В начале строки может находиться метка, за ней через разделитель должна следовать команда и далее один или два операнда, в зависимости от команды. Регистр значения не имеет. В качестве **разделителей** можно использовать один или несколько пробелов, знак табуляции или запятую. Используемый компилятор не имеет препроцессора, поэтому какие-либо операции над операндами недопустимы.

Комментарий - ';' может находиться в любом месте строки. Все, что стоит после него компилятором игнорируется. Другие типы комментариев, такие как '/*', '{}', не поддерживаются.

Метка должна начинаться с начала строки. Знак ':' в конце метки должен присутствовать обязательно. После метки должен быть разделитель: пробел или знак табуляции.

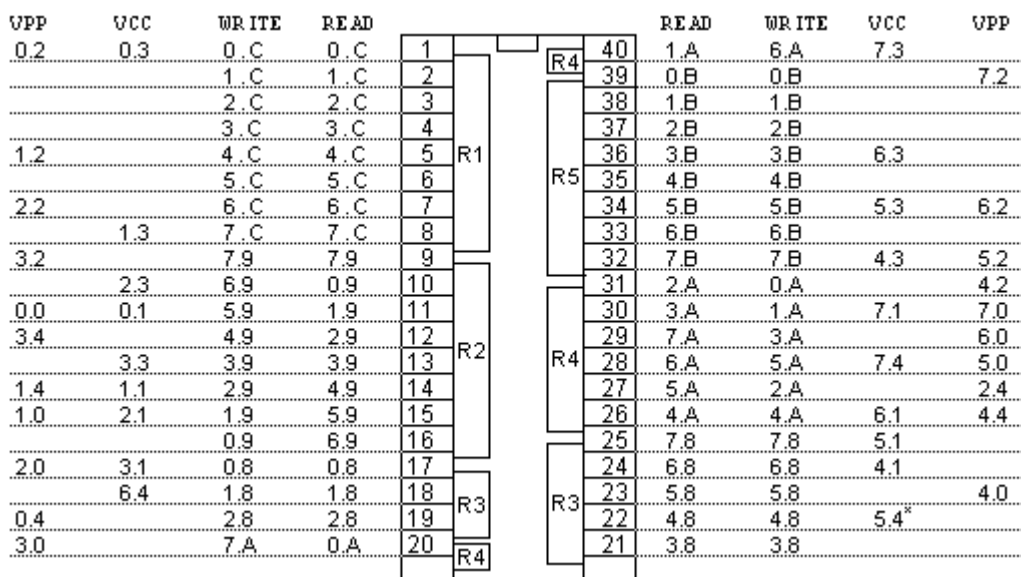
Команда не должна находиться в начале строки. Если метки нет, то перед командой должен быть разделитель. После

команды, через разделители, могут присутствовать один или два операнда.

Операнд. В качестве операндов используются обычные числа, зарезервированные или назначенные константы. Все числа должны записываться только в **шестнадцатеричном** формате, начинаться с символа '\$' и содержать 2 знака. Запись '\$3' - недопустима, правильная запись - '\$03'.

Константы. Для лучшей читаемости скрипта вместо числовых значений можно использовать зарезервированные или назначенные константы. Чтобы назначить константу, в начале скрипта необходимо ввести что-то типа: **WE=\$4B'**

Константа	Описание
ADR_L, ADR_M, ADR_H	Счетчик адреса микросхемы. Автоматически инициализируется в начале каждого цикла. Для инкремента и проверки окончания цикла используются команды JNA (JumpIfNextAddr) и JLA (JumpIfLastAddr).
FIN_L, FIN_M, FIN_H	Конечный адрес микросхемы. Используется при проверке конца цикла. Мало когда нужен, но вдруг потребуется...
DATAL, DATAH	Регистры временного хранения данных, считанных из буфера командами LDB и LDW .
BYTEL, BYTEH	Регистры временного хранения данных, считанных из микросхемы.
TIMER	Регистр таймера. Период инкремента ~3msec. Максимальное значение TIMER ~380msec. Сброс таймера - CLT . Проверка таймера - JTO .
FLAGM, FLAGW, FLAGS, FLAGE	Флаги микросхемы, флаги режимов работы, флаги программатора, флаги ошибок. Описание этих флагов есть в описании блока параметров микросхемы. Менять значения этих флагов, кроме флага ошибок, во время работы, не рекомендуется.
REG_1.. REG_5	Адреса регистров [\$18,\$19,\$1A,\$1B,\$1C], обеспечивающих установку логических уровней и чтение состояния выводов панельки. REG_1=\$1C, REG_2=\$19, REG_3=\$18, REG_4=\$1A, REG_5=\$1B. Для программатора V5.8T: REG_6=\$1D. Привязка регистров к выводам панельки показана на рисунке.
VCP_1.. VCP_5	Адреса регистров [\$10,\$11,\$12,\$13,\$14], обеспечивающих подачу Vpp и Vcc на выводы панельки. Привязка регистров к выводам панельки показана на рисунке.
PULSE	Ячейка блока параметров (\$10). Длительность импульса записи.
NLOCK	Ячейка блока параметров (\$17). Количество бит защиты.
CFG_L, CFG_H	Ячейки блока параметров (\$24, \$25). Конфигурационное слово микросхемы.
REG_CS, REG_DWR, REG_DRD, REG_CLK	Ячейки блока параметров (\$2C..\$2F). Используются для подпрограмм последовательного ввода-вывода. Адреса регистров: CS, Dat_WR, Dat_RD, CLK.



* Vcc на 22 выводе есть только в программаторах V5.7T

Вначале создается описание параметров микросхемы. Лучше подобрать ближайший аналог микросхемы и изменить необходимые параметры. Напряжения, адреса, расположение выводов, все это задается в параметрах микросхемы и затем используется в скрипте для работы. Система команд позволяет писать скрипт не привязанный к данной микросхеме, т.е.

один и тот же скрипт может работать с группой микросхем, отличающихся напряжением питания, объемом памяти и разной цоколевкой. Чтобы перевести любую микросхему со штатного алгоритма на скрипт, надо в параметрах микросхемы изменить ячейку \$12 и загрузить или написать собственно алгоритм работы. Значение \$1E позволяет работать с микросхемами с 16-битной организацией, значение \$1F - для микросхем с 8-битной организацией.

Для правильной работы скрипта необходимо задать **начальное состояние** выводов микросхемы (Log 0/1, GND, VCC), которое будет подано на микросхему в момент включения питания и которое обеспечивает вход микросхемы в режим программирования. Чтобы не повредить находящуюся в микросхеме информацию во время включения или выключения питания, микросхема должна находиться в неактивном состоянии. Для микросхем с параллельным доступом тут же можно назначить шину адреса и данных. Для микросхем с последовательным доступом, линии ввода-вывода назначаются на вкладке **Управляющие сигналы**.

Для работы с любой микросхемой используются всего четыре режима работы: чтение, запись, стирание и установка защиты. Соответственно, в каждом скрипте обязательно должны присутствовать метки **READ**, **PROG**, **ERAS** и **LOCK**. В начале каждого цикла командами **BGR** или **BGW** необходимо подать питание на микросхему. При необходимости, можно выполнить проверку готовности или считать ID микросхемы. Затем для режимов **READ** и **PROG** открыть файл командой **FLP**.

В режиме **READ** данные считываются с микросхемы и обрабатываются, в зависимости от выбранного режима работы (записываются в буфер, сравниваются с данными из буфера, проверяются на чистоту и участвуют в подсчете контрольной суммы). Для чтения 8 или 16 битных микросхем с параллельной шиной данных используются команды **PRB** и **PRW**. Для чтения микросхем с последовательной шиной - **SRL** и **SRR**. Обработка данных производится командами **CPB** (Compare Byte) и **CPW** (Compare Word). Для обработки конфигурации микросхем используются команды **CCB**, **CCW** и **CCP**. Для организации цикла используются команды проверки адреса: **JNA Label** и **JLA Label**.

В режиме **PROG** данные считываются из буфера (команды **LDB** - Load Byte и **LDW** - Load Word) и записываются в микросхему. Для микросхем с параллельной шиной данных, для установки данных используются команды **PWB** и **PWW**, после чего формируется импульс записи. Для микросхем с последовательной шиной - **SWL** и **SWR**. Цикл организуется командами **JNA Label** и **JLA Label**.

В режиме **ERAS** производится стирание микросхемы, если в микросхеме есть такой режим. Если такого режима работы нет, то после метки **ERAS** должна следовать команда END.

Режим **LOCK** обеспечивает установку или снятие защиты, запись Fuses или конфигурационных регистров. Если режим не используется, то после метки **LOCK** должна следовать команда END.

Пример скрипта для работы с W27C512 в корпусе PLCC-32 через переходник TSU-D32/PL32:

```
=====
;===  W27C512P  PLCC-32
;=====
.CE =$2A ;Адреса выводов
.OE =$3A
;=====
ERAS: BGE ;Включить питание
      SVP $52 ;Подать Vpp на A9
      SVP $60 ;Подать Vpp на OE
      PLS CE,$E4 ;Pulse CE,100ms
      SET $60 ;Выключить Vpp на OE
      SET $52 ;Выключить Vpp на A9
      END ;Завершить цикл
;=====
READ: BGR ;Включить питание
      FLP ;Открыть файл
      CLR CE
      CLR OE
L01: A16 ;Адрес 16-bit
      PRB ;Чтение байта
      CPB ;Обработка данных
      JNA L01 ;Следующий адрес
      SET OE
      SET CE
      END
;=====
PROG: BGW ;Включить питание
      FLP ;Открыть файл
      SVP $60 ;Подать Vpp на OE
L02: A16 ;Адрес 16-bit
      LDB ;Байт из буфера
      PWB ;Запись байта
      PLS CE,$80 ;Pulse
```

```
JNA L02 ;Следующий адрес  
SET $60 ;Выключить Vpp  
END
```

```
;=====  
LOCK: END  
;=====
```

Система команд

Арифметические и логические команды.

Команда	Описание	Результат
LDI \$XX	Загружает константу \$XX в ACC.	ACC=\$XX
ADD \$XX	Добавляет константу \$XX к ACC. Флаг переноса игнорируется.	ACC=ACC + \$XX
SUB \$XX	Вычитает константу \$XX из ACC. Флаг переноса игнорируется.	ACC=ACC - \$XX
AND \$XX	Операция AND между ACC и константой \$XX.	ACC=ACC & \$XX
OR \$XX	Операция OR между ACC и константой \$XX.	ACC=ACC \$XX
EOR \$XX	Операция XOR между ACC и константой \$XX.	ACC=ACC # \$XX
INC	Инкремент ACC.	ACC=ACC + 1
DEC	Декремент ACC.	ACC=ACC - 1
LSL	Сдвиг ACC влево. Флаг переноса игнорируется.	ACC=ACC << 1
LSR	Сдвиг ACC вправо. Флаг переноса игнорируется.	ACC=ACC >> 1
COM	Инверсия ACC.	ACC=!ACC
SWAP	Перестановка тетрад в ACC.	ACC=swap ACC
LDS @RR	Загружает ACC из ячейки @RR в блоке параметров. @RR=[\$80..\$FF]. Блок параметров в памяти программатора располагается с адреса \$80.	ACC=@RR
STS @RR	Записывает содержимое ACC в ячейку @RR блока параметров.	@RR=ACC
CBR @RR,\$XX	Выполняет операцию AND между содержимым регистра @RR и константой \$XX.	ACC=\$XX
SBR @RR,\$XX	Выполняет операцию OR между содержимым регистра @RR и константой \$XX.	ACC=\$XX
CBRA @RR	Выполняет операцию AND между содержимым регистра @RR и ACC.	ACC=ACC
SBRA @RR	Выполняет операцию OR между содержимым регистра @RR и ACC.	ACC=ACC
CLT	Сброс таймера TIMER . Период инкремента ~3msec. Максимальное значение ~380msec. Проверка таймера - JTO .	ACC=ACC
LDC \$XX	Загружает константу \$XX в счетчик циклов. Декремент счетчика и проверка нуля - JCZ и JCN .	ACC=ACC
LDR @RR	Загружает содержимое регистра \$RR в счетчик циклов. Декремент счетчика и проверка нуля - JCZ и JCN .	ACC=ACC

Команды проверки условий и переходы.

Команда	Описание	Результат
JAZ LABEL	Jump if ACC Zero - Проверяет ACC и переходит на LABEL, если равен 0.	ACC=ACC
JAN LABEL	Jump if ACC not Zero - Проверяет ACC и переходит на LABEL, если не равен 0.	ACC=ACC
JNA LABEL	Jump if Next Address - Проверяет конечный микросхемы, если не равен, то переход на LABEL. Инкрементирует счетчик адреса и выводит его значение на дисплей программатора. Значение ACC не меняется или равно \$00, если был вывод на дисплей.	ACC=??
JLA LABEL	Jump if Last Address - Проверяет конечный микросхемы, если равен, то переход на LABEL. Инкрементирует счетчик адреса и выводит его значение на дисплей программатора. Значение ACC не меняется или равно \$00, если был вывод на дисплей.	ACC=??

JCZ LABEL	Jump if Count Zero - Уменьшает счетчик циклов на единицу и переходит на LABEL, если равен 0.	ACC=ACC
JCN LABEL	Jump if Count not Zero - Уменьшает счетчик циклов на единицу и переходит на LABEL, если не равен 0.	ACC=ACC
JPL LABEL	Jump if Pin(@ACC) Low - Проверяет вывод панельки, заданный в ACC. Если вывод в нуле, переход на адрес LABEL. На выходе ACC = состоянию выводов панельки REG_x.	ACC=@REG_x
JPH LABEL	Jump if Pin(@ACC) High - Проверяет вывод панельки, заданный в ACC. Если вывод в единице, переход на адрес LABEL. На выходе ACC = состоянию выводов панельки REG_x.	ACC=@REG_x
JWL \$XX,LABEL	Jump if bits \$XX in FLAGW is Zero - Проверяет биты \$XX в регистре FLAGW и переходит на LABEL, если результат равен 0. Используется для проверки, какие дополнительные режимы работы надо выполнить (чтение/запись EEPROM, FUSES, LOCK bits)	ACC=ACC
JWH \$XX,LABEL	Jump if bits \$XX in FLAGW is not Zero - Проверяет биты \$XX в регистре FLAGW и переходит на LABEL, если результат не равен 0.	ACC=ACC
JRE @RR,LABEL	Jump if byte in Register @RR is Equal ACC - Сравнивает содержимое регистра @RR и ACC, и переходит на LABEL, если совпадает.	ACC=ACC
JRN @RR,LABEL	Jump if byte in Register @RR is not Equal ACC - Сравнивает содержимое регистра @RR и ACC, и переходит на LABEL, если не совпадает.	ACC=ACC
JRL @RR,LABEL	Jump if byte in Register @RR is Lower ACC - Сравнивает содержимое регистра @RR и ACC, и переходит на LABEL, если @RR < ACC.	ACC=ACC
JRH @RR,LABEL	Jump if byte in Register @RR is Higher ACC - Сравнивает содержимое регистра @RR и ACC, и переходит на LABEL, если @RR > ACC.	ACC=ACC
JTO LABEL	Jump if TimeOut - Если флаг TimeOut установлен (max =380msec.), переходит на адрес LABEL.	ACC=ACC
JCP LABEL	Jump if Compare - Сравнивает считанный из микросхемы байт (в режиме PROG или LOCK) с байтами из буфера, ранее считанными командами LDB или LDW и переходит на LABEL, если равно.	ACC=ACC
JMP LABEL	Команда перехода на адрес LABEL.	ACC=ACC
CALL LABEL	Вызывает подпрограмму по адресу LABEL. Допускаются вложенные вызовы.	ACC=ACC
RET	Выход из подпрограммы.	ACC=ACC
EXT \$XX	Вызывает подпрограмму \$XX из внешней библиотеки. В зависимости от режима работы (READ или PROG) вызывается соответствующая функция из библиотеки: EXT \$40 - FT_MPSSE_Erase EXT \$41 - FT_MPSSE_PreRead1 или FT_MPSSE_PreProg1 EXT \$42 - FT_MPSSE_PreRead2 или FT_MPSSE_PreProg2 EXT \$43 - FT_MPSSE_PreRead3 или FT_MPSSE_PreProg3 EXT \$44 - FT_MPSSE_ReadPrgm или FT_MPSSE_ProgPrgm EXT \$45 - FT_MPSSE_ReadData или FT_MPSSE_ProgData EXT \$46 - FT_MPSSE_ReadUser или FT_MPSSE_ProgUser EXT \$47 - FT_MPSSE_ReadLock или FT_MPSSE_ProgLock	ACC=?
END	Завершает работу скрипта и передает управление основной программе. Обязательная инструкция в конце каждого цикла.	

Управление выводами панельки.

Команда	Описание	Результат
SVP \$PP	Подает на вывод \$PP напряжение записи или питания. Перед выполнением команды вывод может находиться в нуле.	ACC=@REG_x
CLR \$PP	Устанавливает на выводе \$PP ноль или подает напряжения записи или питания (при этом вывод должен быть заранее установлен в единицу).	ACC=@REG_x

SET \$PP	Устанавливает на выводе \$PP единицу или отключает напряжения записи (питания).	ACC=@REG_x
XOR \$PP	Инвертирует текущее состояния вывода \$PP (логические 0/1) или включает/выключает напряжения (в случае подачи питания, вывод должен быть =1).	ACC=@REG_x
PLS \$PP,\$XX	Формирует на выводе \$PP импульс длительностью \$XX. Где, \$XX = \$00..7F - Длительность = \$XX * 1mks = 1..127mks; = \$81..FF - Длительность = (\$XX-\$80) * 1msec = 1..127msec; = \$80 - Длительность импульса определяется значением ячейки \$10 в блоке параметров = 10mks..12,8msec.	ACC=??
OSC \$PP,\$XX	Формирует на выводе \$PP пакет из \$XX импульсов с частотой около 1,5МГц.	ACC=@REG_x
SVPA	Аналогична SVP , но адрес вывода должен быть предварительно задан в ACC.	ACC=@REG_x
CLRA	Аналогична CLR , но адрес вывода должен быть предварительно задан в ACC.	ACC=@REG_x
SETA	Аналогична SET , но адрес вывода должен быть предварительно задан в ACC.	ACC=@REG_x
XORA	Аналогична XOR , но адрес вывода должен быть предварительно задан в ACC.	ACC=@REG_x
PLSA	Аналогична PLS , но адрес вывода должен быть предварительно задан в ACC. Длительность импульса определяется значением ячейки \$5B в блоке параметров: = \$00..7F - Длительность = \$XX * 1mks = 1..127mks; = \$81..FF - Длительность = (\$XX-\$80) * 1msec = 1..127msec; = \$80 - Длительность импульса определяется значением ячейки \$10 в блоке параметров = 10mks..12,8msec.	ACC=??
PLSC	Формирует импульс на выводе, адрес которого задан в ячейке \$2C (REG_CS). Длительность импульса определяется значением ячейки \$5B в блоке параметров: = \$00..7F - Длительность = \$XX * 1mks = 1..127mks; = \$81..FF - Длительность = (\$XX-\$80) * 1msec = 1..127msec; = \$80 - Длительность импульса определяется значением ячейки \$10 в блоке параметров = 10mks..12,8msec.	ACC=??
IN @REG	Загружает ACC из порта (выводы панельки). REG = [\$18,\$19,\$1A,\$1B,\$1C]. См. рисунок. Можно использовать зарезервированные имена: REG_1=\$1C, REG_2=\$19, REG_3=\$18, REG_4=\$1A, REG_5=\$1B.	ACC=@REG_x
OUT @REG	Записывает содержимое ACC в порт. REG=[Logic=\$18,\$19,\$1A,\$1B,\$1C; Vpp/Vcc=\$10,\$11,\$12,\$13,\$14]. Перед записью в регистры "Vpp/Vcc" (включение напряжений) необходимо установить соответствующий вывод панельки в состояние единицы. Для безопасной подачи напряжений лучше пользоваться командой SVP \$PIN . Например, OUT REG_1,\$02 установит вывод 2 в единицу, а выходы 1,3,4,5,6,7,8 в ноль. На выходе значение ACC=!ACC. OUT \$10,\$02 - подаст Vpp на 15 вывод панельки и отключит выходы 11,17,20,23,28,29,30. Значение ACC не меняется.	@REG_x=ACC, ACC=??

Подпрограммы ввода-вывода данных.

Команда	Описание	Результат
SRL	Serial Read Left - Последовательный, тактируемый ввод (чтение) 8 бит данных в ACC, старшим битом вперед. Ячейки \$2D, \$2E и \$2F в блоке параметров микросхемы содержат адреса выводов панельки для сигналов: Data_WR, Data_RD и CLK. Ячейка \$5B - задает длительность импульса и паузы (\$00..FF = 1..255mks). Частота тактирования CLK = 1 / (2 * Tclk). Например, для Tclk = \$05, частота CLK ~ 100кГц.	ACC=RD_Byte
SRR	Serial Read Right - Последовательный, тактируемый ввод (чтение) 8 бит данных в ACC, младшим битом вперед. Ячейки \$2D, \$2E и \$2F в блоке параметров микросхемы содержат адреса выводов панельки для сигналов: Data_WR, Data_RD и CLK. Ячейка \$5B - задает длительность импульса и паузы (\$00..FF = 1..255mks).	ACC=RD_Byte
SRX \$\$\$	Serial Read - Чтение данных в ACC в последовательном режиме. Аналогична	ACC=RD_Data

	двум предыдущим командам, дополнительно можно задавать направление сдвига и количество бит. Например, $SS=17$: бит4 =0-сдвиг влево; =1-сдвиг вправо; 7-восемь бит данных.	
SWL	Serial Write Left - Последовательный, тактируемый вывод (запись) ACC, старшим битом вперед. Ячейки \$2D, \$2E и \$2F в блоке параметров микросхемы содержат адреса выводов панельки для сигналов: Data_WR, Data_RD и CLK. Ячейка \$5B - задает длительность импульса и паузы ($\$00..FF = 1..255mks$).	ACC=RD_Byte
SWR	Serial Write Righth - Последовательный, тактируемый вывод (запись) ACC, младшим битом вперед. Ячейки \$2D, \$2E и \$2F в блоке параметров микросхемы содержат адреса выводов панельки для сигналов: Data_WR, Data_RD и CLK. Ячейка \$5B - задает длительность импульса и паузы ($\$00..FF = 1..255mks$).	ACC=RD_Byte
SWLC \$XX	Serial Write Left - Последовательный, тактируемый вывод (запись) константы \$XX, старшим битом вперед. Ячейки \$2D, \$2E и \$2F в блоке параметров микросхемы содержат адреса выводов панельки для сигналов: Data_WR, Data_RD и CLK. Ячейка \$5B - задает длительность импульса и паузы ($\$00..FF = 1..255mks$).	ACC=RD_Byte
SWRC \$XX	Serial Write Righth - Последовательный, тактируемый вывод (запись) константы \$XX, младшим битом вперед. Ячейки \$2D, \$2E и \$2F в блоке параметров микросхемы содержат адреса выводов панельки для сигналов: Data_WR, Data_RD и CLK. Ячейка \$5B - задает длительность импульса и паузы ($\$00..FF = 1..255mks$).	ACC=RD_Byte
SWA \$SS	Serial Write - Последовательный, тактируемый вывод (запись) содержимого ACC. Можно задавать направление сдвига и количество бит. Например, $SS=03$: бит4 =0-сдвиг влево; =1-сдвиг вправо; 3-четыре бита данных. Ячейка \$5B - задает длительность импульса и паузы ($\$00..FF = 1..255mks$).	ACC=RD_Data
SWX \$XX,\$SS	Serial Write - Последовательный, тактируемый вывод (запись) константы \$XX. Можно задавать направление сдвига и количество бит. Например, $SS=03$: бит4 =0-сдвиг влево; =1-сдвиг вправо; 3-четыре бита данных. Ячейка \$5B - задает длительность импульса и паузы ($\$00..FF = 1..255mks$).	ACC=RD_Data
PRB	Parallel Read Byte - Чтение байта в ACC для микросхем памяти с параллельной 8-битной шиной данных. Ячейки \$98..\$9F в блоке параметров микросхемы содержат адреса выводов панельки для шины данных D7..D0.	ACC=Byte
PRW	Parallel Read Word - Чтение слова для микросхем памяти с параллельной 16-битной шиной данных. Ячейки \$90..\$9F в блоке параметров микросхемы содержат адреса выводов панельки для шины данных D15..D0.	ACC=LowByte
PRX \$AA	Parallel Read Data - Чтение байта в ACC по "адресу" ($AA=\$10$ -старший байт данных; \$18-младший байт данных).	ACC=Byte
PWB	Parallel Write Byte - Запись содержимого ACC для микросхем памяти с параллельной 8-битной шиной данных. Ячейки \$88..\$8F в блоке параметров микросхемы содержат адреса выводов панельки для шины данных D7..D0.	ACC=\$00
PWW	Parallel Write Word - Запись слова для микросхем памяти с параллельной 16-битной шиной данных. Ячейки \$80..\$8F в блоке параметров микросхемы содержат адреса выводов панельки для шины данных D15..D0.	ACC=\$00
PWBC \$XX	Parallel Write Byte - Запись константы \$XX для микросхем памяти с параллельной 8-битной шиной данных. Ячейки \$88..\$8F в блоке параметров микросхемы содержат адреса выводов панельки для шины данных D7..D0.	ACC=\$00
PWA \$AA	Parallel Write Data - Запись содержимого ACC по "адресу" ($AA=\$00$ -старший байт данных; \$08-младший байт данных; \$20,\$28,\$30 - старший, средний и младший байт адреса).	ACC=\$00
PWX \$XX,\$AA	Parallel Write Data - Запись байта \$XX по "адресу" ($AA=\00 -старший байт данных; \$08-младший байт данных; \$20,\$28,\$30 - старший, средний и младший байт адреса).	ACC=\$00
PZW	Parallel Z-state Word - Перевод 16-битной Шины Данных в третье состояние. Ячейки \$80..\$8F в блоке параметров микросхемы содержат адреса выводов панельки для шины данных D15..D0.	ACC=\$00

Комбинированные подпрограммы.

Команда	Описание	Результат
BGR	Подготавливает программатор к режиму Чтения. Устанавливает Начальное состояние выводов панельки, подает напряжения и обрабатывает Задержку при включении питания. Для снижения шумов напряжение Vpp устанавливается на 0.5v больше, чем Vcc.	ACC=??
BGS	Аналогична режиму BGR . Во время включения питания защита по току отключена. В этом режиме можно использовать ключи Vpp для питания микросхемы (Vcc=Vpp).	ACC=??
BGW	Подготавливает программатор к режиму Записи. Vpp устанавливается согласно заданному напряжению записи. В остальном аналогична режиму BGR .	ACC=??
BGE	Подготавливает программатор к режиму Стирания. Vpp устанавливается согласно заданному напряжению стирания. В остальном аналогична режиму BGR .	ACC=??
FLP	При работе с компьютером подготавливает буфер для работы с компьютером. В автономном режиме открывает файл в памяти программатора. Обязательная команда в режимах READ и PROG. Не использовать в режимах ERAS и LOCK.	ACC=??
FLE	Аналогична FLP , только используется, когда необходим доступ к EEPROM.	ACC=??
LDB	Считывает очередной байт из буфера для последующей записи в микросхему. При необходимости подгружает новый блок данных. Считанный байт помещается в регистр DATAL и ACC. Используется только в режиме PROG.	ACC=DATAL
LDW	Считывает 2 байта из буфера для последующей записи в микросхему. При необходимости подгружает новый блок данных. Считанные байты помещаются в регистры DATAL и DATAH. Сначала идет младший байт, затем старший. Используется только в режиме PROG.	ACC=DATAL
CPB	Сравнивает считанный из микросхемы байт с байтом из буфера. При необходимости подгружает новый блок данных и выводит сообщения об ошибках. Используется только в режиме READ. В зависимости от режима работы байт будет проверен на чистоту (= \$FF) и возможность перезаписи, записан в буфер или сравнен с байтом из буфера.	ACC=??
CPW	Сравнивает считанное из микросхемы слово с 2 байтами из буфера. Аналогична команде CPB . Используется только в режиме READ.	ACC=??
CHL \$XX	Проверяет установлены ли в ACC биты \$XX и устанавливает флаг LOCK в регистре FLAGS. Используется перед командами CCB или CCW . Не меняет содержимое ACC.	ACC=ACC
CCB	Сравнивает считанный из микросхемы байт с младшим байтом конфигурационного слова (\$24). Используется в режимах READ и LOCK.	ACC=??
CCP	Сравнивает считанный из микросхемы байт с байтом из блока параметров, на который указывает указатель, заданный командой SPA . Используется в режимах READ и LOCK.	ACC=??
CCW	Сравнивает считанное из микросхемы слово с 2 байтами конфигурационного слова (\$24,\$25). Используется в режимах READ и LOCK.	ACC=??
SPA \$XX	Устанавливает указатель данных на адрес \$XX (\$80..\$FF) в блоке параметров. Используется только после обработки основной и дополнительной памяти, для обработки FUSES или LOCK bits командой CCP . Команда SPA \$00 восстанавливает предыдущее значение указателя.	ACC=\$XX
CLA	Устанавливает счетчик адреса микросхемы в начальное состояние. Счетчик автоматически устанавливается в командах BGR , BGW , BGS и BGE .	ACC=ACC
CLE	Устанавливает счетчик адреса EEPROM в начальное состояние. Счетчик автоматически устанавливается командой FLE .	ACC=ACC
A08	Для микросхем памяти с параллельной шиной данных и адреса, устанавливает 8 битный адрес. Ячейки \$B0..\$B7 в блоке параметров	ACC=\$00

	микросхемы содержат адреса выводов панельки (A7..A0).	
A16	Для микросхем памяти с параллельной шиной данных и адреса, устанавливает 16 битный адрес. Ячейки \$A8..\$B7 в блоке параметров микросхемы содержат адреса выводов панельки (A15..A0).	ACC=\$00
A24	Для микросхем памяти с параллельной шиной данных и адреса, устанавливает 24 битный адрес. Ячейки \$A0..\$B7 в блоке параметров микросхемы содержат адреса выводов панельки (A23..A0).	ACC=\$00
DAC \$XX	Устанавливает значение ЦАП-а VPP равным \$XX*0.125v.	ACC=??
DACA	Аналогична DAC , но значение ЦАП-а должно быть предварительно задано в ACC.	ACC=??
DLS \$XX	Формирует задержку длительностью \$XX. Где, \$XX =\$01..7F - Длительность = \$XX * 1,00mks = 1..127mks; =\$81..FF - Длительность = (\$XX-\$80)*1msec = 1..127msec; =\$00 - Минимальная задержка, зависит от тактовой частоты программатора. V5.8T - 0.62mks; V5.xT - 0.9mks; V5.x - 1.8mks. =\$80 - Длительность задержки определяется значением ячейки \$10 в блоке параметров = 10mks..12,8msec.	ACC=??
DLL \$XX	Задержка длительностью T = \$XX*10msec.	ACC=\$00
ALG \$XX	Запуск подпрограммы, находящейся в памяти программатора. Позволяет выполнить штатный алгоритм работы с микросхемой. Номер алгоритма задан в ячейке \$12. Запрещены вызовы алгоритмов \$1C, \$1D, \$1E и \$1F. Вызов несоответствующего алгоритма работы приведет к ошибкам в работе и может повредить микросхему. \$XX может принимать значения: =AAAA A000 - ERAS - режим стирания. =AAAA A010 - LOCK - режим установки защиты. =AAAA A100 - PROG - режим записи. =AAAA A110 - READ - режим чтения. Где, Axxx - номер алгоритма *8. Например: W27C512 (алг.= \$01): Запись = \$0C, Чтение = \$0E, Стирание = \$08. P12F508A (алг.= \$0F): Запись = \$7C, Чтение = \$7E, Стирание = \$78, Запись CFG = \$7A. i28F800C3B(алг.= \$16): Запись = \$B4, Чтение = \$B6, Стирание = \$B0.	ACC=??
ERR	Выводит диалог с сообщением об ошибке. Используется только в режиме PROG.	ACC=??
OFF	Выключение питания, сброс всех выводов панельки в состояние OFF и задержка 20ms.	ACC=\$00

Команды для работы с определенным типом микросхем.

Команда	Описание	Результат
SPI flash		
SPI_CMD \$XX	Передача однобайтной команды: Low_CS + Command(\$XX). Это должна быть первая команда, которая передается в микросхему, после включения питания.	ACC=??
SPI_CWR \$XX	Передача однобайтной команды: Low_CS + Command(\$XX) + High_CS.	ACC=??
SPI_CRD \$XX	Однобайтная команда чтения: Low_CS + Command(\$XX) + Read_Byte + High_CS.	ACC=RD_Byte
SPI_ADR \$XX	Передача команды и установка адреса: Low_CS + Command(\$XX) + Address(1..4 bytes). Количество байт адреса: Par[\$11].4 =1 - второй байт адреса; Par[\$11].5 =1 - третий байт адреса; Par[\$11].6 =1 - четвертый байт адреса (всегда = 0).	ACC=??
SPI_ADF \$XX	Передача команды и установка адреса в формате AT45*: Low_CS + Command(\$XX) + Address(2 bytes)+\$00.	ACC=??
SPI_RDP	Чтение страницы + High_CS. Для микросхем со страничной организацией. Размер страницы = Par[\$10]*256+Par[\$16].	ACC=??
SPI_RDA	Чтение ВСЕЙ микросхемы с обработкой адреса + High_CS.	ACC=??

SPI_WRP	Загрузка страницы + High_CS. Для микросхем со страничной организацией. Размер страницы = Par[\$10]*256+Par[\$16].	ACC=??
SPI_WRA	Загрузка страницы с обработкой адреса + High_CS. Размер страницы = Par[\$10]*256+Par[\$16].	ACC=??
NAND Flash		
NAN_RDP	Чтение страницы + High_CS. Размер страницы = Par[\$10]*256+Par[\$16].	ACC=??
NAN_WRP	Загрузка страницы. Размер страницы = Par[\$10]*256+Par[\$16].	ACC=??
NAN_ADR \$XX	Передача команды и установка адреса: Low_CS + Command(\$XX) + Address(3..5 bytes). Количество байт адреса: Par[\$11].5 =1 - двухбайтный адрес PAGE; Par[\$11].6 =1 - трехбайтный адрес ROW/BLOCK.	ACC=??
NAN_CBS \$XX	Передача команды и ожидание готовности: Low_CS + Command(\$XX) + Busy_Pin (ячейка \$2E).	ACC=??
NAN_CST \$XX	Передача команды и чтение статус-регистра: Low_CS + Command (\$XX) + Status + High_CS.	ACC=Status
MCS-51		
MCS_48T	Задержка 48 pulses on XTAL.	ACC=??
MCS_DEL	Задержка ~24msec. Pulses on XTAL.	ACC=??
MCS_ADR	Установка адреса(2 bytes) + 48T.	ACC=??
MCS_ARD	Установка адреса(2 bytes) + 48T + Read_Byte.	ACC=RD_Byte
MCS_ADW	Установка данных(@R2) + установка адреса(2 bytes) + 48T.	ACC=??
MCS_MOD \$XX	Установка режима работы \$XX.	ACC=??
MCS_MRD \$XX	Установка режима работы \$XX + Read_Data.	ACC=??
MCS_PLS \$XX	Импульс записи ALE/PROG (ячейка \$2C). Длительность = \$XX. Где, \$XX = \$00..7F - Длительность = \$XX * 10 mks = 10..1270mks; = \$81..FF - Длительность = (\$XX-\$80) * 2 msec = 2..514msec; = \$80 - Длительность задержки определяется значением ячейки \$10 в блоке параметров.	ACC=??
MCS_BSY	Проверка готовности BUSY_pin (ячейка \$2E)	ACC=??
PIC16		
P16_CRD \$XX	Передача команды (\$XX) и чтение слова из микросхемы.	ACC=LowByte
P16_CWR \$XX	Передача команды (\$XX) и загрузка слова в микросхему.	ACC=??

Описание ошибок и сообщений программы

Ниже дан ответ на два самых частых вопроса, которые задают техподдержке:

В управляющей программе есть ошибка! - В УПРАВЛЯЮЩЕЙ ПРОГРАММЕ ОШИБОК НЕТ. В программном обеспечении практически нет алгоритмов, которые написаны для одной конкретной микросхемы. Все алгоритмы и скрипты пишутся для группы или семейства микросхем и проверяются на реальных микросхемах.

Программатор ничего не пишет, ни читает! - изучите **РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ**, техническую документацию на микросхему и научитесь работать с программатором. Все программаторы перед продажей тестируются, настраиваются и проверяются в работе с конкретными микросхемами. На исправном программаторе, при правильном подключении исправной микросхемы, все заявленные микросхемы работают согласно технической документации.

Ниже приведен полный список сообщений, которые может выдавать управляющая программа в процессе работы с микросхемами или при инициализации программатора. Найдите нужное сообщение, нажмите на ссылку и прочитайте описание ошибки, причину ее появления и возможные способы устранения. Дополнительно посмотрите раздел "[Особенности работы с микросхемами](#)".

Запуск программы, подключение программатора, коммуникационные ошибки:

- [Программатор не найден. Запустить программу в демо-режиме.](#)
- [Программа не видит список портов на компьютере.](#)
- [СОМ-порт недоступен или занят.](#)
- [СОМ-порт. Некорректные параметры.](#)
- [СОМ-порт. Программатор не отвечает.](#)
- [СОМ-порт. Ошибка инициализации программатора.](#)
- [Прием \(Передача\) данных. Ошибка контрольной суммы.](#)

Ошибки при работе с микросхемами:

- [Ошибка установки микросхемы.](#)
- [Дефект или другой тип микросхемы.](#)
- [Сработала защита по питанию.](#)
- [Time Out. Программатор не отвечает.](#)
- [Мало время ожидания.](#)
- [Программатор читает одни нули или FF, не стирает или не пишет.](#)
- [Программатор плохо читает, стирает или программирует микросхемы.](#)
- [Возникают ошибки при верификации микросхемы.](#)
- [Ошибка записи микросхемы, число ошибок ноль.](#)
- [При обращении к микросхеме программатор перезапускается.](#)
- [Не получается считать или записать микросхему в устройстве.](#)

Ошибки при обновлении программы:

- [Файл прошивки не найден.](#)
- [Программа не может определить версию программатора.](#)
- [Версия файла не соответствует версии программы.](#)
- [Self Test...Fail / Load Firmware.](#)

Другие неисправности, которые могут возникнуть при эксплуатации программатора:

- [В автономном режиме недоступны списки микросхем.](#)
- [Загрузка файла в программатор. Недостаточно памяти.](#)
- [Программатор не включается.](#)
- [Программатор не определяется программой.](#)

Запуск программы, подключение программатора, коммуникационные ошибки:

Программатор не найден. Запустить программу в демо-режиме?

- Проверьте правильность [установки USB драйверов](#) на компьютере.
- Запустите программу **от имени Администратора**.
- При работе в Windows 7, 8, 10 и 11 отключите Контроль учетных записей (UAC) в профиле пользователя и перезагрузите компьютер.
- Перед запуском программы подключите программатор, включите питание и **ждите готовности** программатора (должен загореться зеленый светодиод и в нижней строке появится сообщение "Bl. Pg. Vr. Menu").
- Проверьте или замените интерфейсный кабель, подключите программатор к другому порту и снова запустите программу.
- Смотрите раздел: [Устранение проблем с подключением программатора](#).

Управляющая программа не видит список портов на компьютере - выбранный профиль пользователя не имеет нужных полномочий для доступа к ресурсам компьютера.

- Запустите программу от имени **Администратора**.
- Смотрите раздел: [Устранение проблем с подключением программатора](#).

COM-порт недоступен или занят – программа не может открыть COM или USB -порт.

- Выбранный порт уже используется другой программой или устройством. Подключите программатор к другому порту или закройте другие программы. Попробуйте перезагрузить компьютер.
- При предыдущем запуске программатор был подключен в другой порт или через переходник USB–COM, который в этот момент не установлен. Проведите автоматический поиск программатора или перезапустите программу.
- Нормальное выполнение программы было прервано в момент, когда шла работа с программатором. В этом случае выключите и снова включите питание программатора, повторите последнюю команду или перезапустите программу.
- При работе в Windows 7, 8, 10 или 11 запустите программу от имени **Администратора** и отключите контроль учетных записей (UAC) в профиле пользователя.
- При подключении через USB или при использовании переходников USB-COM под Windows 7, 8, 10 или 11, установите драйвера, адаптированные для работы в этих операционных системах.

COM-порт. Некорректные параметры – какой-то из параметров COM-порта имеет недопустимое значение.

- Перезагрузите компьютер. Если ошибка появляется регулярно, сообщите об этом в службу поддержки.
- При использовании переходника USB – COM скорость обмена может быть равной 230400 или 460800 бод. Некоторые переходники, а также стандартный COM-порт такие скорости не поддерживает. Установите скорость 115200 бод или меньше.

COM-порт. Программатор не отвечает – программатор не отвечает на запросы компьютера.

- Установите правильный номер COM-порта или проведите автоматический поиск программатора.
- Проверьте подключение, интерфейсный кабель и переподключите питание программатора.
- Программатор должен находиться в режиме готовности, т.е. в нижней строке должны быть выведены режимы работы с микросхемой "Bl. Pg. Vr. Menu". Только в этом режиме программатор отвечает на запросы компьютера.
- При работе в Windows 7, 8, 10 или 11 запустите программу от имени **Администратора** и отключите контроль учетных записей (UAC) в профиле пользователя.

COM-порт. Ошибка инициализации программатора – программатор не может связаться с компьютером на новой скорости.

- Попробуйте изменить скорость работы COM порта
- Подключите программатор к другому порту.

Прием (Передача) данных. Ошибка контрольной суммы – искажен или потерян байт во время приема или передачи блока данных.

- При подключении через USB, попробуйте использовать более короткий кабель или кабель с маркировкой HIGH-SPEED.
- При подключении через COM порт, проверьте разводку и исправность [интерфейсного кабеля](#), попробуйте изменить скорость работы COM порта.
- Возможно, другой процесс или программа имеющие более высокий приоритет прерывают работу программы. При чтении больших объемов, такие ошибки часто вызывает антивирусное ПО.
- Проверьте, не перезапускается ли программатор во время работы с микросхемой. Обычно, это происходит при внутрисхемном программировании, когда устройство потребляет слишком большой ток, что вызывает срабатывание защиты по входу.

Ошибки при работе с микросхемами:

Ошибка установки микросхемы - сообщение программы проверки контактов шины данных (pin-tester) перед началом работы с микросхемой. Данная проверка выполняется перед каждым циклом, до подачи напряжения на микросхему.

- Отсутствие контакта по одному или нескольким выводам шины данных. Наиболее частая ошибка, связанная с окислением выводов микросхемы или плохой очисткой паяных микросхем.
- Выбор микросхемы другого типа или производителя.
- Неправильная установка микросхемы в панельке. Использование несоответствующей переходной панельки или адаптера.
- Для старых микросхем EPROM, выполненных по технологии NMOS: аппаратная реализация ключей программатора не

позволяет тестировать контакт выводов у этих микросхем. Отключить проверку контактов можно на закладке [Параметры](#) установив флаг "Отключить проверку ШД".

Дефект или другой тип микросхемы - сообщение программы проверки сигнатуры микросхемы или при отсутствии отклика микросхемы. Проверка выполняется в начале каждого цикла, сразу после подачи напряжения на микросхему.

- Выбор микросхемы другого типа или производителя.
- Неправильная установка микросхемы в панельке. Использование несоответствующей переходной панельки или адаптера.
- Работа с неисправной микросхемой.
- Работа с микроконтроллером в котором установлена защита от чтения.
- Для микросхем AVR в режиме ISP, эта ошибка может возникать при работе с низкочастотным кварцами. Изменить скорость тактирования можно в [Управляющих сигналах](#) на закладке [Параметры](#).

Сработала защита по питанию – превышение максимально допустимого тока для формирователей напряжения питания или программирования.

- При программировании микросхемы в панельке программатора такое сообщение может возникнуть в следующих случаях:
 - при неправильной установке микросхемы или установке неисправной микросхемы.
 - при использовании несоответствующей переходной панельки или адаптера.
 - при сбое в протоколе обмена с компьютером. Попробуйте более короткий кабель или кабель с маркировкой HIGH-SPEED. Подключите программатор к другому USB порту или другому компьютеру.
 - при повреждении ключей программатора. Чтобы проверить исправность ключей удалите микросхему из панельки и [протестируйте программатор](#).
 - при перезаписи микроконтроллера с рабочей программой, когда используется внутренний генератор и время запуска микроконтроллера меньше, чем установленная длительность задержки при включении питания. В этом случае необходимо **уменьшить** значение в ячейке \$18 в [блоке параметров](#) микросхемы.
- При программировании микросхемы в составе устройства такое сообщение может появляться, если устройство потребляет ток более 50-80mA и (или) содержит большую емкость по цепи питания, что и вызывает срабатывание защиты в момент подачи напряжений. В первом случае необходимо отключить вывод питания микросхемы от остальной платы и подключить питание с программатора к этому выводу. Во втором случае, необходимо **увеличить** длительность задержки при включении питания, в течении которой защита будет отключена. Номер этой ячейки в блоке параметров микросхемы = \$18, значение устанавливается из расчета 1ms на ~5-10mkF.
- При внутрисхемном программировании защита может срабатывать во время конфликта между сигналами программатора и работающего в устройстве процессора. Как правило, остановить процессор можно сигналом RESET (для этого можно использовать свободный вывод разъема ICSP, подав на него логический ноль или единицу) или остановкой кварца.
- При внутрисхемном программировании PIC-контроллеров это сообщение может возникнуть, если вывод MCLR микросхемы подключен к VCC через резистор менее 10кОм. Рекомендуемое значение этого резистора не менее 20кОм.

Time Out. Программатор не отвечает или **Мало время ожидания** – программатор не отвечает в течение заданного таймаута.

- Если на программаторе горит красный светодиод, то программатор ожидает окончания внутреннего цикла записи или стирания микросхемы (для некоторых микросхем памяти длительность цикла стирания может достигать 3.5 минут). Необходимо дождаться завершения цикла и, если был включен режим программирования, то повторить запись без предварительного стирания микросхемы.
- Некоторые микросхемы памяти, в которых установлена защита от перезаписи, могут зависать в режиме стирания. Необходимо отключить защиту и повторить режим стирания.
- Сбой в протоколе обмена (потеря байта). Если при чтении или записи, на дисплее программатора перестает обновляться информация о ходе процесса, то попробуйте использовать более короткий кабель или кабель с маркировкой HIGH-SPEED. Для программатора V5.8T, можно попробовать изменить системную частоту или время выборки.
- Если светодиод на мгновение гаснет, а затем загорается зеленым, то в программаторе сработала защита по входу питания. Может проявляться при внутрисхемной записи при неправильном подключении или слишком больших значениях Задержки при включении питания.

Программатор читает одни нули или FF, не стирает или не пишет - тут многое зависит от типа микросхемы. Рекомендуется изучить раздел [особенности работы с микросхемами](#) и фирменную документацию на микросхему.

- Неправильная установка микросхемы в панельку или неисправность самой микросхемы.
- Неправильный выбор переходной панельки или отсутствие контакта.
- При внутрисхемном программировании: неправильное подключение или шунтирование сигналов программатора внутри устройства.
- В микросхеме может быть установлена защита. Существуют два типа защит:
 - **Защита от чтения.** Устанавливается в микроконтроллерах для защиты программы от несанкционированного доступа. При установленной защите, в большинстве случаев, содержимое микросхемы читается нулями или FF. Отключить защиту, чтобы считать содержимое микросхемы, нельзя. Доступен только режим стирания с потерей

записанной внутри программы. Запись без стирания также не работает.

- **Защита от перезаписи.** Устанавливается в микросхемы памяти и предназначена для защиты данных от случайного стирания или записи. При этом содержимое микросхемы читается всегда, но не работает стирание или запись. В зависимости от типа микросхемы, программатор может отключать защиту автоматически или оператор должен отключить защиту отдельной командой. В некоторых микросхемах защита устанавливается однократно и изменить содержимое таких микросхем уже невозможно.

Программатор плохо читает, стирает или программирует микросхемы.

- При использовании переходных панелек, проверьте правильность распайки и надежность контактов.
- Проверьте состояние выводов микросхемы. Они не должны быть гнутыми или обломанными, на них не должно быть остатков припоя, клея, окислов...
- Загрузите и установите самую последнюю версию программного обеспечения. Посмотрите раздел [особенности работы с микросхемами](#) и фирменную документацию по программированию на соответствующую микросхему.
- [Протестируйте программатор](#), проверьте [калибровку напряжений](#) питания и записи.

Возникают ошибки при верификации микросхемы.

- Проверьте [калибровку напряжений](#) питания и записи.
- Если включены две контрольные сверки после чтения или записи и ошибка появляется при проверке до 50%, необходимо **увеличить** значение Vcc min, если ошибки появляются после 50%, то необходимо **уменьшить** значение Vcc max.
- Можно включить режим одной контрольной сверки или сделать напряжения Vcc max и Vcc min, равными Vcc nom.
- Дополнительно посмотрите раздел [особенности работы с микросхемами](#).

Ошибка записи микросхемы, число ошибок ноль – ошибка записи микросхемы, ошибка записи конфигурационного слова или установки битов защиты.

- При записи FLASH памяти 28- или 29-серий микросхема установила флаг ошибки. Можно попробовать изменить напряжения питания микросхемы во время записи, установить блокировочную емкость (0,1..10mkF) на выводы питания микросхемы или переключить микросхему на универсальный алгоритм записи (ячейка \$11, старший бит =1. Например, значение в \$11 ячейке =\$6D - стандартный алгоритм записи, =\$ED - универсальный).
- При программировании PIC контроллеров это сообщение появляется, когда считанное из микросхемы конфигурационное слово не совпадает с исходным. Связано это с тем, что у многих PIC-ов неиспользуемые биты в конфигурационном слове читаются нулями, а большинство компиляторов устанавливает их в единицы. В этом случае в оболочке программатора необходимо изменить значение любого бита конфигурации и вернуть его в прежнее состояние.

При обращении к микросхеме программатор перезапускается.

- Может проявляться при внутрисхемной записи при неправильном подключении или слишком больших значениях задержки при включении питания.
- Поврежден ключ в программаторе. Чтобы проверить исправность ключей удалите микросхему из панельки и [протестируйте программатор](#). Если какой-либо из тестов постоянно сообщает об одной и той же ошибке, то обратитесь в [службу поддержки](#).

Ошибки при обновлении программы:

Файл прошивки не найден. Это сообщение появляется при попытке установить старую версию программы на новую модель программатора (например, на программатор V5.8T нельзя установить версию ПО V5.7.49). Также может возникать из-за ошибки чтения параметров программатора перед загрузкой прошивки. В этом случае:

- Запустите программу **от имени Администратора**.
- Проверьте наличие файлов прошивок в папке BIN.
- Сравните дату и версию файлов прошивок с датой и версией программы.
- Переподключите программатор к компьютеру и повторите команду.
- Перезапустите программу, переподключите программатор и повторите команду.

Программа не может определить версию программатора. Такая ошибка может возникнуть при повреждении внутреннего EEPROM в программаторе. Необходимо включить питание программатора, удерживая нажатыми кнопки 'Vg.' и 'Menu', и [восстановить прошивку](#) в программаторе.

Версия файла не соответствует версии программы - загружаемая прошивка подготовлена для другой версии программы. Как правило, возникает при неправильной установке новой версии программы, когда мастер установки не может заменить некоторые файлы, например, запущенную программу. Необходимо заново переустановить программу. Для заказных или платных прошивок необходимо установить версию программы, для которой они подготовлены.

Self Test...Fail / Load Firmware - повреждена программа в памяти программатора.

- Попробуйте [восстановить прошивку](#) в программаторе.
- Установите предыдущую версию управляющей программы и попробуйте на ней восстановить прошивку программатора.
- Если во время восстановления программы, загрузка проходит, но после программатор опять выводит это сообщение, то обратитесь в [службу поддержки](#).

Другие неисправности, которые могут возникнуть при эксплуатации программатора:

В автономном режиме недоступны списки микросхем. Необходимо открыть стандартный или сформировать свой собственный [список микросхем](#) и загрузить его в память программатора.

Загрузка файла или проекта в программатор. Недостаточно памяти. Освободить память в программаторах V5.7T можно:

- Удалив ненужные [файлы и проекты](#).
- Сократив [список микросхем](#).
- Отформатировав [память программатора](#).
- В памяти программатора V5.7T может быть сохранено не более 256 файлов, общим объемом 4096 КБайт. Из них 4 КБайта отведено под список файлов (FAT) и от 2 до 128 КБайт под список микросхем. В памяти программатора можно сохранить один файл объемом 2 МБайта, или 3 файла по 1 МБайту или 7 файлов по 512 КБайт... Файл, объемом 4 МБайта сохранить в памяти программатора нельзя.

Программатор не включается - при включении питания дисплей не работает, светодиод не горит.

- Неисправен блок питания, поврежден кабель питания или не та полярность. См. раздел [Питание программатора V5.7TU](#) или [V5.8TU](#).
- Вследствие сильного удара, поврежден кварцевый резонатор в программаторе. Разберите программатор и попробуйте подключить любой кварц с частотой 4-12МГц, если загорится светодиод или появится информация на дисплее, то замените кварц или обратитесь в [службу поддержки](#)
- Повреждены ключи в программаторе. Проверьте ток, потребляемый программатором в режиме ожидания. Если ток превышает 50mA, обратитесь в [службу поддержки](#).

Программатор не определяется программой. После запуска сторонних программ, например, от программатора RT809F, программатор ТРИТОН может перестать видиться в программе. Это может проявляться только на программаторах V5.7TM и V5.7TU.

Симптомы ошибки:

- При запуске оболочки программатор не обнаружен, но Диспетчере устройств и в списке портов есть нужный COM*-порт.
- При включении питания и переходе в ждущий режим программатор не реагирует на кнопки и не отвечает на запросы программы.
- При удержании кнопок в момент включения питания программатор переходит в заданные режимы.

Если ВСЕ эти признаки присутствуют, то необходимо восстановить настройки FT245RL.

1. Скачайте и запустите программу [MPROG.EXE](#).
2. Подключите программатор к компу и включите питание.
3. В Диспетчере устройств должен появиться USB SERIAL PORT (COM*).
4. В программе MPROG выберите меню DEVICE -> SCAN. В нижнем окне программы появится строчка: Number Of Programmed Devices = 1
5. В программе MPROG выберите меню FILE -> OPEN. В открывшемся окне выберите файл FT245RL.EPT.
6. В программе MPROG выберите меню DEVICE -> PROGRAM. В нижнем окне программы появится строчка: Programmed Serial Number : XXXXXXXX.
7. Выключите питание программатора и закройте программу MPROG.EXE.
8. Включите питание программатора и запустите оболочку программатора.

Особенности работы с микросхемами

Представленная в этом разделе информация не заменяет фирменную документацию, а только описывает особенности работы микросхемы на программаторах ТРИТОН.

Микросхемы памяти с параллельным доступом (8- и 16-бит):

- [Микросхемы памяти EPROM 27x](#)
- [Микросхемы памяти EEPROM 28x](#)
- [Микросхемы памяти FLASH](#)
- [Микросхемы памяти FWH и LPC-Flash](#)
- [Микросхемы статического ОЗУ](#)
- [Микросхемы памяти NAND-Flash](#)

Микросхемы памяти с последовательным доступом (Serial EEPROM):

- [Микросхемы Serial DATA-Flash](#)
- [Микросхемы SEEPROM 17x](#)
- [Микросхемы SEEPROM 24x](#)
- [Микросхемы SEEPROM 25x](#)
- [Микросхемы SEEPROM 93x](#)

Микроконтроллеры:

- [Микроконтроллеры AVR](#)
- [Микроконтроллеры Holtek](#)
- [Микроконтроллеры MSC-51](#)
- [Микроконтроллеры Philips P89LPC9xx](#)
- [Микроконтроллеры Winbond](#)
- [Микросхемы Pic10xxx, Pic12xxx, Pic16C5x](#)
- [Микросхемы Pic12F6xx, Pic16Cxxx, Pic16Fxxx](#)
- [Микросхемы Pic17C](#)
- [Микросхемы Pic18C, Pic18F, Pic24, dsPIC](#)

Для работы с микросхемами в корпусах, отличных от DIP, могут быть использованы различные переходные панельки и адаптеры. При записи большого количества микросхем необходимо использовать только панельки с нулевым усилением.

Схемы разводки специальных переходников можно посмотреть в окне выбора микросхемы, кликнув мышкой по названию "Схема адаптера".

Программаторы V5.8TU могут работать как со специальными, так и с универсальными панельками без потери скорости. Возможно использование адаптеров от других программаторов, но для этого надо переназначить сигналы на другие выводы панельки программатора, в соответствии со схемой этого адаптера.

Внутрисхемное программирование

Все модели программаторов ТРИТОН позволяют программировать микросхемы непосредственно в устройствах пользователя. Для этого не только микросхема, но и само устройство должны поддерживать режим внутрисхемного программирования. В устройстве должно быть предусмотрено подключение программатора. Поскольку для программирования многих микросхем используются напряжения, значительно превышающие напряжение питания, устройство должно выдерживать эти напряжения. Подключаемое устройство не должно оказывать шунтирующего влияния на сигналы программатора. Как правило, все эти требования подробно описаны в фирменных спецификациях по программированию на каждую микросхему.

Общие замечания и рекомендации:

- Перед подключением разъема для внутрисхемного программирования, **ОБЯЗАТЕЛЬНО СОЕДИНИТЬ** общий вывод платы или корпус устройства с корпусом компьютера или программатора!!! В первую очередь это касается устройств работающих от внешних импульсных источников питания. При отсутствии надежного контакта или при неправильном заземлении, разность потенциалов будет приложена через выводы панельки к ключам программатора. Как правило, такие программаторы уже не ремонтпригодны.
- Формирователь напряжения питания в программаторе обеспечивает ток не более 80mA. При внутрисхемном программировании, с учетом емкостей по питанию, ток потребляемый программируемым устройством не должен превышать 50mA. Если устройство потребляет больший ток, то необходимо использовать внешний источник питания. При этом подавать питание одновременно этого источника и программатора НЕЛЬЗЯ.
- Программатор формирует уровни сигналов в соответствии с заданными в программе значениями напряжения питания. Поэтому, при питании устройства от внешнего источника питания, необходимо установить значения напряжений (V_{cc} min, max, nom), формируемые программатором, в соответствии с напряжением этого источника.
- Программируемое устройство может иметь большие емкости в цепях питания. Программатор позволяет работать с емкостями по питанию до 500mkF, при условии, что устройство потребляет ток не более 20..30mA. Для этого в параметрах микросхемы необходимо изменить ячейку \$18 - длительность задержки при включении питания. Этот параметр определяет время, в течение которого защита в программаторе находится в выключенном состоянии, обеспечивая заряд емкости максимальным током. Значение устанавливается из расчета 1ms на ~5-10mkF.
- Программатор формирует "землю" и логический ноль с помощью полевых транзисторов с очень малым сопротивлением канала. Поэтому выводы микросхемы, которые используются при программировании, не должны подключаться напрямую к цепям питания или выходам других микросхем, которые находятся в активном состоянии. Выходы этих микросхем должны быть отключены переключками на время программирования или переведены в третье состояние с помощью дополнительных сигналов.

V5.7T. Программирование микросхем внутрисхемно может осуществляться двумя способами: непосредственно сигналами с панельки программатора или с помощью специального переходника [TSH-ICSP](#).

V5.8T. Внутрисхемное программирование осуществляется только через встроенный разъем **ISP-CONN**, который имеет защиту статики и аппаратную "землю". Панелька TSH-ICSP на этом программаторе не работает. Подключаться напрямую к выводам панельки программатора ЗАПРЕЩЕНО. В случае ошибки подключения и повреждения ключей программатора ремонт такого программатора осуществляться не будет.

Программирование сигналами с панельки программатора (только для V5.7T)

Для этого нужно просто соединить выводы микросхемы с соответствующими выводами панельки программатора и, если микросхема имеет несколько алгоритмов программирования, выбрать режим ICSP. Посмотреть на каких выводах программатор сформирует необходимые сигналы, можно в документации на данную микросхему или на закладке "Параметры".



Возможные проблемы. Ключи программатора, которые формируют логические сигналы и обеспечивают чтение данных с микросхемы, построены по схеме с открытым коллектором, в нагрузку которого стоит резистор 10k. Поэтому, если в устройстве пользователя к выводам DATA или CLOCK подключена какая-либо низкоомная нагрузка, то возможное шунтирование сигналов программатора и как следствие сбой в работе. В этом случае, непосредственно на разъем для внутрисхемного программирования, между VCC и сигналом DATA (или CLOCK) можно установить дополнительный резистор номиналом 0,5..1k, который будет подтягивать уровень сигнала и увеличивать нагрузочную способность. Для устойчивой работы необходимо отношение сопротивления нагрузки к сопротивлению в открытом коллекторе не менее, чем 5:1.

Программирование через переходник TSH-ICSP (только для V5.7T)

В отличие от выводов панельки программатора, которые построены по схеме с ОК, логические выходы переходника TSH-ICSP буферизированы, т.е. выполнены по схеме push-pull и обеспечивают ток нагрузки в нуле и единице до 15-20mA. Остальные сигналы: питание, земля и напряжение программирования передаются на выходы переходника без изменений. Необходимо отметить, что ток нагрузки логических выходов идет за счет источника питания в программаторе, максимальный ток которого не должен превышать 80mA.



Переходник для внутрисхемного программирования TSH-ICSP поддерживает технологию универсальных алгоритмов и может быть частично переконфигурирован пользователем:

выводы с 1 по 4	Vcc, Vpp, GND, логический вход/выход с ОК.
выводы 5, 7, 9	буферизированные логические входы/выходы, вход/выход с ОК или GND.
выводы 6, 8	Vpp, GND, логический вход/выход с ОК.
вывод 10	Vcc, GND, логический вход/выход с ОК.

Для увеличения нагрузочной способности выходов с ОК на выводах 6, 8 и 10 на плате переходника предусмотрена возможность установки дополнительных подтягивающих резисторов номиналом 0,5..1k.

Если расположение сигналов на выходах панельки не совпадает с цоколевкой разъема на плате, то программа позволяет изменить конфигурацию выводов. Для этого в положениях "Начальное состояние" и "Управляющие сигналы" надо мышкой перетащить сигналы на требуемые выводы, после чего сохранить настройки. Чтобы после переназначения сигналов, переходник работал корректно, менять конфигурацию надо в двух местах:

- в "Начальном состоянии", где задается состояние выводов, которое будет подано на микросхему в момент включения питания, и которое обеспечивает вход микросхемы в режим программирования.
- и в положении "Управляющие сигналы", где задаются номера выводов на которых программатор будет формировать логические сигналы.

Программа учитывает схмотехнику переходника и программатора, и подсвечивает неправильно сконфигурированные сигналы желтым цветом. Кроме того, при перетаскивании управляющих сигналов, программа может исправлять некоторые ошибки.

Программирование через встроенный разъем ISP-CONN (только для V5.8T)

Основной причиной повреждения ключей в старых моделях программаторов, было подключение незаземленных устройств, поэтому в программаторах V5.8T был сделан специальный разъем для внутрисхемного программирования. В этом разъеме крайние выводы GND соединены с "землей" программатора, а остальные выводы имеют защиту от статического электричества.

Цоколевка разъема ISP-CONN меняется в зависимости от выбранной микросхемы. Посмотреть на каких выводах программатор сформирует необходимые сигналы, можно на закладке "Параметры". Цвет проводов шлейфа, идущего в комплекте, не имеет никакого отношения к цветам сигналов в оболочке программатора.



Как и для переходника TSH-ICSP программа позволяет изменить конфигурацию выводов. Для этого в положениях "Начальное состояние" и "Управляющие сигналы" надо мышкой перетащить сигналы на требуемые выводы. Сигналы перетаскиваются только на панельке программатора и отображаются на разъеме.

Возможные проблемы. Программатор не всегда может проверить подключение микросхемы на длинном шлейфе, поэтому при появлении сообщения "Ошибка установки микросхемы", необходимо установить флаг "Отключить проверку ШД". На микросхемах с SPI интерфейсом (25 серия), возможны ошибки при работе на частотах 33 или 50МГц, поэтому рекомендуется снизить тактовую частоту до 16МГц.

Микросхемы памяти EPROM 27х

Выбор микросхемы.

Для большинства этих микросхем работает автоматическое определение типа микросхемы. Для этого нужно установить микросхему в панельку, выбрать фирму "Auto detect" и нажать кнопку "Определить". Часть микросхем в этом режиме не работают. Это: микросхемы типа 2716, 2732 и 2764; многие микросхемы выпуска начала 90-х г.; микросхемы с 16-битной организацией. Выбор таких микросхем должен осуществляться вручную.

Особенности работы.

Чтение. Для микросхем EPROM с 8-битной организацией доступны три алгоритма чтения. При выборе микросхемы, по умолчанию включается алгоритм FAST, оптимизированный и обеспечивающий максимально возможную скорость работы. Алгоритм SOFT работает несколько медленнее, но при этом значительно снижает уровень шумов источника питания в программаторе. Он устанавливается для старых микросхем, а также для некоторых 3v микросхем, которые читаются с ошибками в алгоритме FAST. Переключить эти режимы можно в параметрах микросхемы, ячейка \$18 (длительность задержки): четное значение - режим FAST, нечетное - SOFT. Также для всех микросхем доступен универсальный алгоритм (не привязанный к архитектуре программатора), который позволяет работать практически с любыми микросхемами. В этом режиме цоколевка микросхемы задается программно, непосредственно в блоке параметров микросхемы. Программное обеспечение позволяет переназначить VCE сигналы шин адреса, данных и управления на произвольные выводы панельки программатора. Переключение со стандартного на универсальный алгоритмом производится автоматически, при переназначении любых сигналов, или в параметрах микросхемы, ячейка \$11 (байт флагов микросхемы): 7бит =0 - стандартный алгоритм, 7бит =1 - универсальный алгоритм. Обратное переключение возможно только в случае когда цоколевка микросхемы соответствует стандартной разводке для микросхем этого типа.

Запись. При выборе микросхемы программатор устанавливает алгоритм и электрические параметры в соответствии с фирменной спецификацией. Сервис программатора позволяет изменять любые режимы программирования и выбрать один из четырех алгоритмов программирования:

- **Normal** – стандартный алгоритм: запись и последующий контроль ячейки памяти. Если данные не совпадают, то производится до 25 попыток записи ячейки, после чего выводится сообщение об ошибке. Если данные совпали, то производится дополнительная запись ячейки (закрепление данных, в ячейку добавляется 3*N импульсов).
- **Quick** – быстрый алгоритм: запись и последующий контроль ячейки памяти. Как и в алгоритме "Normal" производится до 25 попыток записи. Закрепляющая запись ячейки не производится.
- **Rapid** – скоростной алгоритм: запись в каждую ячейку по одному импульсу заданной длительности, после чего проверка всей микросхемы и если данные в ячейке не совпадают, производится запись и последующий контроль ячейки памяти (до 25 попыток записи на ячейку).
- **MTP** – самый быстрый алгоритм записи: запись в каждую ячейку по одному импульсу. Рекомендуется для микросхем 37SF* фирмы Silicon Storage Technology.

Для ускорения процесса записи микросхем в меню "Программатор/Параметры и тесты" можно установить флаги "Устанавливать конечный адрес по размеру файла", "Пропускать проверку на чистоту перед записью" и "Одна контрольная сверка после чтения или записи".

Особенности работы в автономном режиме.

Для этих микросхем в программаторе предусмотрен режим автоматического определения типа микросхемы. Для этого нужно установить микросхему в панельку программатора, войти в режим выбора микросхемы и нажать кнопку 'Mem'. Если микросхема включена в [Список микросхем программатора](#), то ее название появится на дисплее. Если нет, то программатор предложит выбрать микросхему из списка.

Переходные панельки.

При работе с 8-битными микросхемами в корпусах PLCC32. Необходимо учитывать, что переходные панельки для этих микросхем могут быть двух типов: DIP28 – PLCC32 или DIP32 – PLCC32. Первый тип панельки используется для работы с микросхемами, объем памяти которых 64КБ и меньше. Это микросхемы 27х64, 128, 256 и 512. Для микросхем 27х010, 020, 040 и 080 необходимо использовать переходник DIP32 – PLCC32.

Возможные ошибки и методы их устранения.

Для старых микросхем, с плавающими битами. Если при чтении микросхемы, на этапе верификации появляются ошибки, то необходимо снизить напряжение питания (V_{ccnom}), установить V_{ccmin} и V_{ccmax} равными V_{ccnom} , и повторить чтение.

Временные параметры записи микросхем ориентированы на работу с современными микросхемами, и при работе со старыми версиями микросхем возможны ошибки на этапе программирования. Если при нажатии кнопки "Повторить" программатор показывает ошибку по другому адресу, то в этом случае желательно установить алгоритм "Normal" или увеличить длительность или количество импульсов записи.

Если запись микросхемы проходит нормально, а на этапе верификации хаотично возникают ошибки, можно немного снизить максимальное напряжение питания (V_{ccmax}).

Микросхемы памяти EEPROM 28x

Выбор микросхемы.

Большинство микросхем этого типа не содержат идентификационных кодов, поэтому автоматическое определение для них не возможно. Выбор таких микросхем должен осуществляться только вручную.

Особенности работы.

Чтение. Для микросхем EEPROM с 8-битной организацией доступны три алгоритма чтения. При выборе микросхемы, по умолчанию включается алгоритм FAST, оптимизированный и обеспечивающий максимально возможную скорость работы. Алгоритм SOFT работает несколько медленнее, но при этом значительно снижает уровень шумов источника питания в программаторе. Его можно включить, если микросхема читается с ошибками в алгоритме FAST. Переключить эти режимы можно в параметрах микросхемы, ячейка \$18 (длительность задержки): четное значение - режим FAST, нечетное - SOFT. Также для этих микросхем доступен универсальный алгоритм (не привязанный к архитектуре программатора), который позволяет работать практически с любыми микросхемами. В этом режиме цоколевка микросхемы задается программно, непосредственно в блоке параметров микросхемы. Программное обеспечение позволяет переназначить ВСЕ сигналы шин адреса, данных и управления на произвольные выходы панельки программатора. Переключение со стандартного на универсальный алгоритмом производится автоматически, при переназначении любых сигналов, или в параметрах микросхемы, ячейка \$11 (байт флагов микросхемы): 7бит =0 - стандартный алгоритм, 7бит =1 - универсальный алгоритм. Обратное переключение возможно только в случае когда цоколевка микросхемы соответствует стандартной разводке для микросхем этого типа.

Запись. Часть микросхем 28C*, 29C* серий имеет два режима записи. Программатор обеспечивает запись как в обычном режиме, так и в режиме "Software Data Protection" (для этого необходимо установить этот флаг). Во время стирания микросхемы сначала отключается режим "Software Data Protection", после чего стирается вся микросхема. При программировании микросхем с побайтовой записью программатор проверяет флаг окончания записи и байт данных, записанный в микросхему. При программировании микросхем со страничной записью программатор проверяет только флаг окончания записи. Если цикл записи байта или страницы превышает максимально допустимое время, то процесс программирования прерывается и выводится сообщение об ошибке.

29C* – размер загружаемых данных или начальный и конечный адреса микросхемы должны быть кратны размеру страницы. Изменять размер страницы не рекомендуется. Перезапись части страницы этими микросхемами не поддерживается.

В режимах чтения контроль установленной защиты на сектора или Boot block не производится. Если в микросхеме установлены биты защиты на сектора, то режим стирания не работает, но по команде стирания отключается режим "Software Data Protection". Содержимое защищенного сектора изменить нельзя. При этом возможна перезапись остальной памяти микросхемы без предварительного стирания.

Последовательность работы программатора в режиме записи:

- Стирание и проверка микросхемы на чистоту;
- Программирование основной памяти;
- Контроль качества записи;
- Установка защиты (если разрешена) на Boot block.

Для ускорения процесса записи микросхем в меню "Программатор/Параметры и тесты" можно установить флаги "Устанавливать конечный адрес по размеру файла", "Пропускать проверку на чистоту перед записью" и "Одна контрольная сверка после чтения или записи".

Особенности работы в автономном режиме.

Для этих микросхем в программаторе предусмотрен режим автоматического определения типа микросхемы. Для этого нужно установить микросхему в панельку программатора, войти в режим выбора микросхемы и нажать кнопку 'Mem'. Если микросхема включена в [Список микросхем программатора](#), то ее название появится на дисплее. Если нет, то программатор предложит выбрать микросхему из списка.

Программаторы ТРИТОН+ в автономном режиме позволяют разрешить или запретить установку защиты (кнопки 'Menu', 'Utl'+ 'Lck') и дублировать микросхемы с установленной защитой. В этом случае будет установлена защита, на нулевой сектор или boot block.

Микросхемы памяти FLASH

Выбор микросхемы.

Для большинства этих микросхем возможно автоматическое определение типа. Для этого устанавливаем микросхему в панельку, выбираем фирму "Auto detect" и нажимаем кнопку "Определить". Коды некоторых микросхем совпадают, например: AT49F002 и AT49F002N или SST39LF* и SST39VF*. Поэтому если программатор не может определить их тип, или определяет не правильно, то выбор таких микросхем должен осуществляться вручную.

Особенности работы.

Чтение. Для 8-битных микросхем доступны два алгоритма чтения. При выборе микросхемы, по умолчанию включается алгоритм FAST, оптимизированный и обеспечивающий максимально возможную скорость работы. Алгоритм SOFT работает несколько медленнее, но при этом значительно снижает уровень шумов источника питания в программаторе. Его можно включить, если микросхема читается с ошибками в алгоритме FAST. Переключить эти режимы можно в параметрах микросхемы, ячейка \$18 (длительность задержки): четное значение - режим FAST, нечетное - SOFT. Для 16-битных микросхем предусмотрен только режим чтения FAST.

Также для всех микросхем доступен универсальный алгоритм (не привязанный к архитектуре программатора), который позволяет работать практически с любыми микросхемами. В этом режиме цоколевка микросхемы задается программно, непосредственно в блоке параметров микросхемы. Программное обеспечение позволяет переназначить ВСЕ сигналы шин адреса, данных и управления на произвольные выходы панельки программатора. Переключение со стандартного на универсальный алгоритм производится автоматически, при переназначении любых сигналов, или в параметрах микросхемы, ячейка \$11 (байт флагов микросхемы): 7бит =0 - стандартный алгоритм, 7бит =1 - универсальный алгоритм. Обратное переключение возможно только в случае когда цоколевка микросхемы соответствует стандартной разводке для микросхем этого типа.

V5.8T. Ошибки чтения можно устранить, увеличив время выборки или ее множитель на закладке Параметры - Управляющие сигналы. Либо установить флаг "Отключить HardMode".

В режимах чтения контроль установленной защиты на сектора или Boot block не производится. Контроль этих флагов осуществляется только в момент установки или снятия защиты.

Стирание. Для микросхем 29 серии доступно как стирание всей микросхемы (код \$10), так и стирание отдельного сектора (код \$30). Для этого начальный адрес микросхемы нужно установить равным адресу сектора, который нужно стереть, и в параметрах микросхемы (ячейка \$2F) изменить код команды стирания с \$10 на \$30. Если в микросхеме имеются сектора, на которые установлена защита, то команда стирания всей микросхемы (код \$10) может зависать или не работать. При стирании по секторам (код \$30), будут стерты незащищенные сектора, а содержимое защищенных секторов не изменится. Для временного отключения защиты необходимо установить флаг "Temporary Sector Unprotect" выполнить процесс стирания или записи. После окончания цикла защита будет восстановлена в исходное состояние, которое было до начала работы. Для снятия защиты со всех секторов микросхемы можно воспользоваться кнопкой "UnProtect" на панели управления. Если в микросхеме установлен режим "Защиты Паролем", то снятие защиты с секторов не поддерживается. Соответственно, стирание и запись защищенных секторов невозможны.

Для микросхем 28Fxxx выбор алгоритмов записи или стирания не доступен. Если микросхема имеет несколько алгоритмов стирания, то выбирается наиболее быстрый режим или обеспечивающий более полное стирание. При записи также выбирается наиболее быстрый алгоритм.

Запись. Последовательность работы программатора в режиме записи:

- Стирание и проверка микросхемы на чистоту;
- Программирование основной памяти;
- Контроль качества записи;
- Установка защиты (если разрешена) на Boot block или выбранный сектор.

Для ускорения процесса записи микросхем в меню "Программатор/Параметры и тесты" можно установить флаги "Устанавливать конечный адрес по размеру файла", "Пропускать проверку на чистоту перед записью" и "Одна контрольная сверка после чтения или записи".

При программировании микросхем с побайтовой записью программатор проверяет флаг окончания записи и байт данных, записанный в микросхему. При программировании микросхем со страничной записью программатор проверяет только флаг окончания записи. Если цикл записи байта или страницы превышает максимально допустимое время, то процесс программирования прерывается и выводится сообщение об ошибке.

При установленном флаге "Temporary Sector Unprotect" (временное отключение защиты) на микросхему подается дополнительное напряжение, которое отключает защиту и позволяет перезаписать микросхему, после чего защита возвращается в исходное состояние. Если этот флаг сброшен, а в микросхеме установлена защита, то при записи защищенного блока программатор выведет сообщение об ошибке. Содержимое этого блока при этом не меняется.

Когда для микросхемы 29xxx серии выбран универсальный алгоритм, то в режиме "UnProtect" работает только альтернативный алгоритм снятия защиты (Low Voltage Algorithm). Высоковольтный режим снятия защиты в этом алгоритме не поддерживается.

Особенности работы в автономном режиме.

Для этих микросхем в программаторе предусмотрен режим автоматического определения типа микросхемы. Для этого нужно установить микросхему в панельку программатора, войти в режим выбора микросхемы и нажать кнопку 'Mem'. Если микросхема включена в [Список микросхем программатора](#), то ее название появится на дисплее. Если нет, то программатор предложит выбрать микросхему из списка.

Программаторы ТРИТОН+ в автономном режиме позволяют разрешить или запретить установку защиты (кнопки '**Menu**', '**Uti**'+'**Lck**') и дублировать микросхемы с установленной защитой. В этом случае будет установлена защита, на нулевой сектор или boot block.

Микросхемы памяти FWH и LPC-Flash

Выбор микросхемы.

Для микросхем этого типа автоматическое определение типа микросхемы не работает. Попытка считать сигнатуру микросхемы может привести к порче микросхемы. Выбор таких микросхем должен осуществляться вручную.

Особенности работы.

SST49LF003. Необходимо учитывать, что адресное пространство этой микросхемы начинается с \$020000, а не с нуля. Выбор этой микросхемы на программаторе в автономном режиме недоступен. В автономном режиме эта микросхема может работать только в виде проекта.

Для ускорения процесса записи микросхем в меню "Программатор/Параметры и тесты" можно установить флаги "Пропускать проверку на чистоту перед записью" и "Одна контрольная сверка после чтения или записи".

Возможные ошибки и методы их устранения.

Для некоторых микросхем (i82802xx) на этапе верификации, при повышенном напряжении питания (3,5 - 3,6v), могут хаотично возникать ошибки. Для их устранения можно немного снизить максимальное напряжение питания.

Микросхемы статического ОЗУ

Особенности работы.

Программатор обеспечивает работу только с энергонезависимыми микросхемами статического ОЗУ фирм DALLAS, THOMSON или их аналогами.

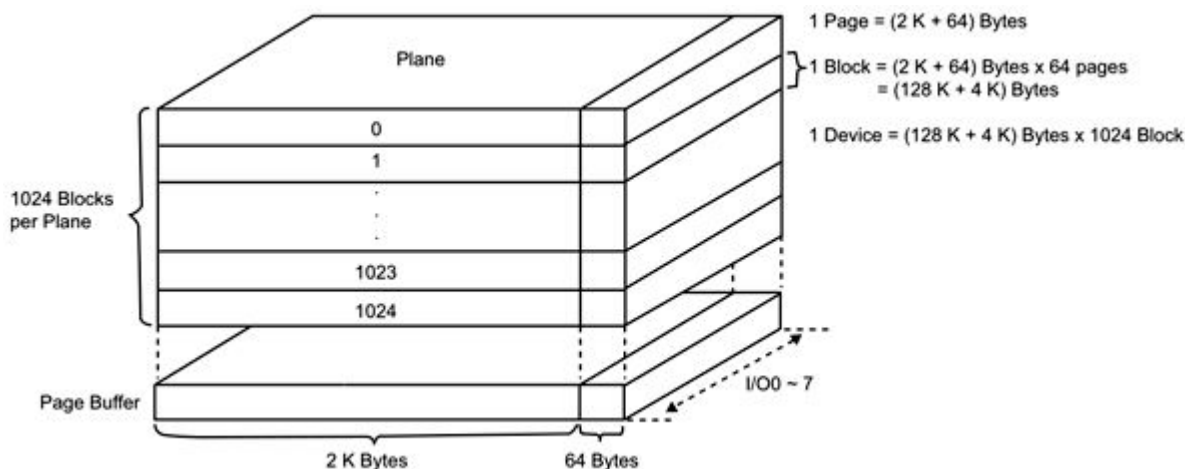
Работа с микросхемами ОЗУ с самодельной подпиткой, также как и сохранность находящейся в них информации не гарантируется.

При чтении микросхем со встроенным таймером, типа M48Txxx, могут возникать ошибки, связанные с работой этого таймера. В этом случае нужно изменить конечный адрес микросхемы, таким образом, чтобы программатор не обращался к этим регистрам.

Для работы с 36-выводными микросхемами (bq4016, bq4017, DS1265, DS1270, M48T513, M48Z2M1) на 36 вывод микросхемы необходимо подать питание с 40 вывода панельки программатора.

Микросхемы NAND-Flash

Работа с микросхемами NAND-Flash несколько отличается от работы с обычными микросхемами памяти. Это связано с особенностями самих микросхем. Данные микросхемы имеют очень большой объем памяти при относительно небольших размерах кристалла, что приводит к невысокой надежности целостности данных и нестандартным режимам чтения и записи. На рисунке приведена типовая организация микросхемы NAND-Flash.



Память микросхемы разделена на страницы (pages). Размер страницы зависит от микросхемы. Страница - это минимальный размер данных, который может быть считан или записан в микросхему. В каждой странице, кроме основной области находится небольшая дополнительная (spare) область, в которой может храниться служебная информация: коды коррекции ошибок, маркеры дефектных блоков, номера страниц и т.д. 64 или 128 страниц объединяются в блок (block). Это минимальная единица при стирании. Чтобы изменить в памяти микросхемы один байт, надо считать и записать страницу, но чтобы перезаписать страницу, надо стереть целый блок. Таким образом, чтобы изменить в микросхеме один байт, надо считать, стереть и записать целый блок. Даже, если содержимое страницы можно переписать без стирания, делать это настоятельно не рекомендуется. Повторная запись одной и той же страницы или запись в микросхему пустой страницы, может привести к появлению ошибок в этой странице.

Самой плохой особенностью NAND-Flash является наличие дефектных ячеек памяти или целых секторов, которые могут быть в микросхеме уже на этапе изготовления и могут размножаться в процессе эксплуатации. Причиной появления ошибок в микросхемах могут быть сбои в устройстве при стирании или записи данных, появление дефектных ячеек, вызванное старением или износом микросхемы.

На форумах есть немало примеров, когда один и тот же дамп, тупо записанный в разные микросхемы, приводит к противоположным результатам: в одном случае устройство работает, в другом нет. Здесь все зависит от того, попадают ли нужные данные в дефектный блок и корректно ли программатор пишет микросхему. Если микросхема не содержит дефектных блоков или дефектный блок попадает в пустое место в прошивке, то, как правило, устройство работает нормально. В противном случае устройство не запустится или будет работать с ошибками. Такой же результат будет, если записать микросхему в режиме пропуска плохих блоков, а в устройстве используется алгоритм RBA. Другой пример, когда в прошивке требуется изменить один или несколько байт. Запись проходит без ошибок, но устройство работает не правильно. В этом случае виноваты алгоритмы коррекции ошибок (ECC - Error Correction Code), используемые в устройстве. Если изменить дамп или карту дефектных блоков и не перерасчитать ECC в этих страницах, то устройство воспримет внесенные изменения как ошибки и исправит их.

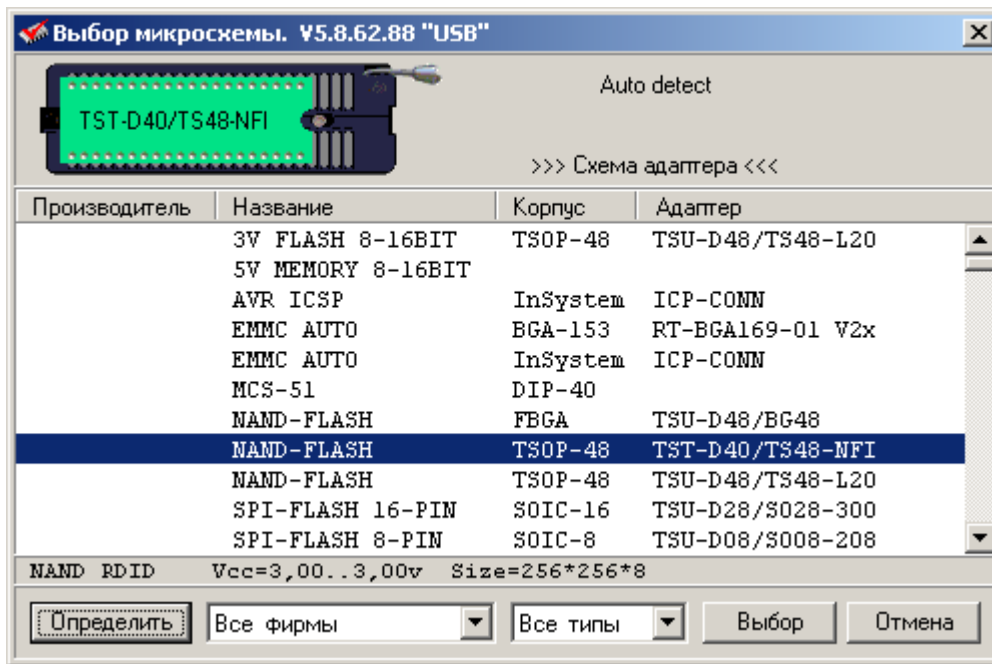
Таким образом, чтобы программатор правильно работал с микросхемами NAND-Flash, он должен не только обеспечивать высокую скорость работы и поддерживать "обработку плохих блоков", но, в первую очередь, **программатор должен поддерживать алгоритмы коррекции ошибок, используемые в устройстве**, а во-вторых, **поддерживать таблицы и схемы обработки дефектных блоков, используемые в устройстве**. Без этих опций, работа любого программатора с микросхемами NAND-Flash, это потеря.

В отличие от программаторов "ведущих" разработчиков, программаторы Тритон знают и поддерживают реальные схемы обработки дефектных блоков, алгоритмы коррекции ошибок, маркеры и таблицы плохих блоков, используемые в реальных устройствах. Предварительная подготовка дампа не требуется. Все операции с данными производятся программой "на лету" во время чтения и записи микросхемы.

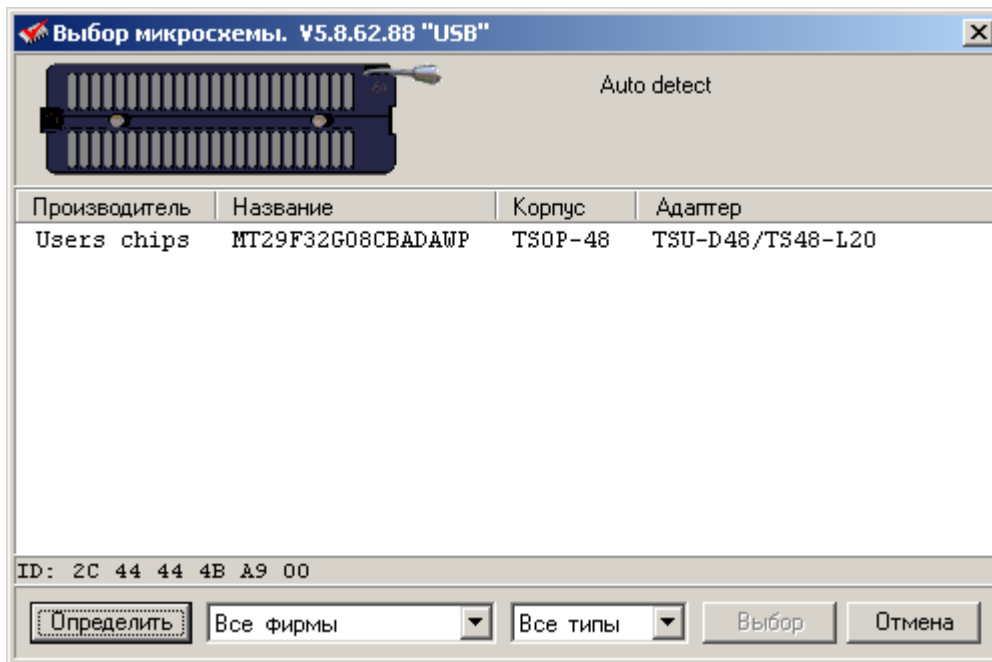
Работу с микросхемами NAND-Flash поддерживают программаторы Тритон V5.7TU и V5.8TU при подключении через USB. Работа с микросхемами NAND-Flash в автономном режиме или через COM порт отключена из-за низкой скорости. Рекомендуется использовать программатор V5.8TU, не только как более современный и скоростной, но и потому, что эта модель имеет гораздо больше возможностей и настроек для обеспечения стабильной работы с б/у микросхемами. В отличие от программатора V5.7TU, в котором процессор программно формирует управляющие сигналы, в программаторе V5.8TU работа с микросхемами организована на аппаратном уровне, без участия процессора.

Выбор микросхемы.

Для микросхем NAND-Flash в программе доступно автоматическое определение (чтение ID) и считывание блока Параметров микросхемы. Запуск скрипта чтения выполняется при нажатии кнопки "Определить" в окне выбора микросхемы. Для корректной работы скрипта должен быть выбран соответствующий адаптер.



Если считанный ID отсутствует в базе микросхем, программа производит анализ блока Параметров по стандарту ONFI, настраивает все параметры и добавляет ее в список поддерживаемых микросхем. После перезапуска программы микросхема будет находиться в разделе "Users Chips".



Чтение микросхемы.

Все ошибки, возникающие при чтении микросхемы, можно разделить на три типа. Определив тип ошибки и следуя изложенным ниже рекомендациям, можно добиться стабильной работы программатора с любыми микросхемами NAND-Flash.

```

Reading...
Verifying...
Адрес 000314-1F24: 007F 00FF
Адрес 00033E-2153: 00BC 00FC
Адрес 00034F-0437: 0020 0000
Ошибка чтения. Число ошибок - 3 T=11,01s
    
```

Битовые ошибки (или одиночные), называются так потому, что в ячейке, как правило, искажен только один бит. В микросхемах таких ошибок относительно не много. Как правило, они будут расположены по разным адресам, в разных страницах. При чтении такой микросхемы, ошибка может проявляться по одному и тому же адресу или пропадать, при изменении напряжения питания. Со временем, в некоторых микросхемах, вследствие износа памяти, количество этих ошибок может увеличиваться даже в выключенном состоянии. Наличие таких ошибок не влияет на работоспособность микросхемы в устройстве, если их число в каждой странице не превышает корректирующую способность кода.

Для исправления таких ошибок в устройстве используются специальные алгоритмы коррекции ошибок (ECC - Error Correction Code). Наибольшее распространение получили три алгоритма:

- Код Хэмминга (Hamming Code) - позволяет исправлять одну битовую ошибку в блоке 256 или 512 байт.
- Код Рида-Соломона (Reed-Solomon Code) - корректирует до 4 поврежденных байт в 512-байтном блоке.
- Коды BCH (BCH Code) - позволяют исправить до 64 ошибок в блоках от 256 до 4096 байт.

Программное обеспечение поддерживает все три алгоритма ECC, а также позволяет задать размер и организацию логической страницы, число корректируемых ошибок, расположение корректирующих кодов на странице, порождающий полином и алгоритмы обработки данных. Подробнее о настройках и работе ECC написано в разделе [Коррекция ошибок](#).

Для не особо подготовленных пользователей, в программе имеется режим автоматического определения алгоритма ECC - **Auto Correction**. В этом режиме, на основе имеющейся базы данных, программа проверяет считываемые или записываемые данные и подбирает алгоритм коррекции. На практике, программа определяет порядка 90% всех алгоритмов, используемых в телевизорах, ресиверах, навигаторах, магнитолах и другом оборудовании.

```
Verifying...
Адрес 00B600-0000: 0000 0046
Адрес 00B600-0001: 0000 0053
Адрес 00B600-0002: 0000 0052
Адрес 00B600-0003: 0000 005F
Адрес 00B600-0004: 0000 0053
Адрес 00B600-0005: 0000 0054
```

Блочные ошибки начинаются с начала страницы, идут подряд по всем адресам, во всех страницах в пределах блока. Число ошибок зависит от размеров блока и может достигать нескольких десятков и сотен тысяч. Дефектные блоки в современных микросхемах, как правило, заполнены нулями, не стираются и читаются стабильно. Но нередко попадаются микросхемы с "приобретенными" дефектными блоками с "плавающим" содержимым, которое каждый раз считывается по-разному. Читать содержимое таких микросхем без ошибок невозможно. Добиться более-менее стабильного чтения с меньшим количеством ошибок, можно подбором напряжения питания (меняются одновременно все напряжения питания и повторяется режим чтения). При стирании и последующей записи "приобретенные" дефектные блоки в большинстве случаев восстанавливаются.

Для работы с дефектными блоками в реальных устройствах используются два алгоритма:

- Пропуск плохих блоков (Skip Bad Blocks).
- Использование резервной области (RBA - Reserved Block Area).

Программное обеспечение поддерживает три схемы RBA, используемые в телевизорах LG (2 схемы) и SAMSUNG (1 схема). А также алгоритм пропуска дефектных блоков с различными таблицами рабочих и дефектных блоков. Выбор этих схем и таблиц производится автоматически на основе анализа считываемых данных или записываемого файла. В большинстве случаев, достаточно в меню [Управление блоками](#) включить режим **Auto Detection**.

```
Verifying...
Адрес 001A18-0E55: 00D3 0000
Адрес 001A18-0E56: 0054 00D3
Адрес 001A18-0E57: 0012 0054
Адрес 001A18-0E58: 00FC 0012
Адрес 001A18-0E59: 0057 00FC
Адрес 001A18-0E5A: 0027 0057
Адрес 001A18-0E5B: 0013 0027
```

Сдвиговые ошибки идут подряд, могут начинаться с любого адреса и не пересекают пределы страницы. Данные сдвинуты на 1-2 байта в одну или в другую сторону. Сдвиговые ошибки проявляются только при чтении данных с микросхемы. В режиме записи их не бывает. Об этих ошибках мало кто говорит и пишет, но они встречаются на разных микросхемах и на многих программаторах. Большинство производителей предлагают аппаратные доработки адаптеров для устранения этих ошибок. В программаторах Тритон, в зависимости от модели, предлагаются два варианта:

- Для V5.8TU. На закладке "Параметрах" установите флаг "Отключить Hard Mode" и повторите чтение.
- Для V5.7TU - подбор напряжений. Пропустите все ошибки, чтобы увидеть общее количество ошибок. После этого уменьшите напряжения питания (Vcspot, Vcstip и Vcstax) и повторите чтение. Если количество ошибок уменьшилось, снова уменьшите все напряжения и повторите чтение. Если чисто ошибок увеличивается, то увеличьте напряжения. К

сожалению, иногда попадают микросхемы, которые считать на программаторах V5.7TU не получается. В этом случае могут помочь следующие рекомендации:

- Попробуйте увеличить блокировочную емкость по питанию микросхемы. Припаять дополнительный конденсатор (0,1..4mkF) можно на переходную панельку, там снизу есть место.
- Попробуйте подтянуть сигнал RE (5 вывод панельки) к VCC (40 вывод панельки) через резистор 1kOhm.

Особенности работы.

При работе с микросхемами NAND-Flash, для обеспечения максимальной скорости работы, компьютер должен иметь достаточный объем свободной оперативной памяти. При недостатке памяти программа предложит сохранить считываемые данные напрямую в файл. Дополнительно, программа может сохранять данные в сжатый файл (в ZIP формате) или читать из него при записи микросхемы. Поскольку микросхемы NAND-Flash имеют большие объемы, то процессы чтения и записи могут продолжаться достаточно долго. Так, чтение 128-МБайтной микросхемы K9F1G08 на программаторе V5.7TU при работе через USB составляет порядка 6 минут. Более новый программатор V5.8TU прочитает эту микросхему менее чем за 20 секунд.

Для микросхем NAND-Flash в окнах адресов (начальный, конечный, смещение) на закладке "Параметры" выводится не физический адрес, а HOMEР страницы. Порядковый номер байта внутри страницы недоступен. Размер страницы зависит от типа микросхемы и может быть равен 264, 528 или 2112 байт. Этот размер устанавливается по умолчанию при выборе микросхемы. Программатор позволяет читать полностью все данные страницы или только основные (256, 512 или 2048 байт), пропустив служебную область. Для этого в параметрах микросхемы в ячейке \$16 (число дополнительных байт), нужно установить значение \$00. Программировать микросхемы с такими установками не рекомендуется, так как содержимое дополнительной области будет не определено.

Для удобства работы, при просмотре дампа памяти в программе, служебная (spare) область выделяется синим цветом. Клавиши курсора вверх или вниз, при нажатой клавише CTRL перемещают курсор на следующую страницу, а клавиши Page Up и Page Dn, при нажатом CTRL перемещают курсор на следующий блок.

Так как, согласно документации, микросхемы NAND-Flash могут иметь дефектные сектора, то в процессе записи или стирания контролируется только флаг окончания внутреннего цикла, что позволяет запрограммировать или стереть всю микросхему или заданную область. Состояние флага ошибки запоминается в программаторе и обрабатывается в конце каждого цикла. Так, если процесс стирания прошел без ошибок, то программатор пропускает проверку на чистоту и сразу переходит к записи. Если в процессе стирания возникла ошибка или в микросхеме имеются дефектные блоки, то после стирания выполняется проверка на чистоту. При установленном флаге "Пропустить Test Blank" проверка на чистоту не выполняется.

Рекомендуется следующий порядок работы с микросхемами NAND-Flash.

- После выбора микросхемы, не меняя настроек, запускаем режим чтения.
- Если при верификации появились ошибки, определяем характер этих ошибок.
- Если это сдвиговые ошибки, то меняем параметры и повторяем чтение, добиваясь стабильной работы..
- Если это битовые ошибки или микросхема считалась без ошибок, то включаем [Коррекцию ошибок](#) и [Управление блоками](#) в режим AUTO и повторяем чтение.
- Если программа определила алгоритм ECC, то читаем и пишем NAND, как обычную микросхему.
- Если скорость чтения резко упала, что может говорить о незнакомом алгоритме ECC, то отключаем [Коррекцию ошибок](#).

Чтобы добавить новый алгоритм ECC, выложите архив со считанным дампом в облако (например, на Яндекс-диск) и пришлите ссылку в техподдержку. Не забудьте указать название микросхемы, с которой был считан этот дамп.

Микросхемы Serial DATA-Flash

Выбор микросхемы.

Автоматическое определение для этих микросхем в программаторе не реализовано. Выбор таких микросхем должен осуществляться вручную.

Особенности работы.

Работа с микросхемами DATA-Flash ведется с использованием технологии универсальных алгоритмов. При необходимости, в параметрах микросхемы можно поменять коды команд для работы с микросхемой, это ячейки \$26..\$29. При записи микросхем ATMEL (AT45*) используется режим с предварительным стиранием страницы, что позволяет переписать часть микросхемы без общего стирания.

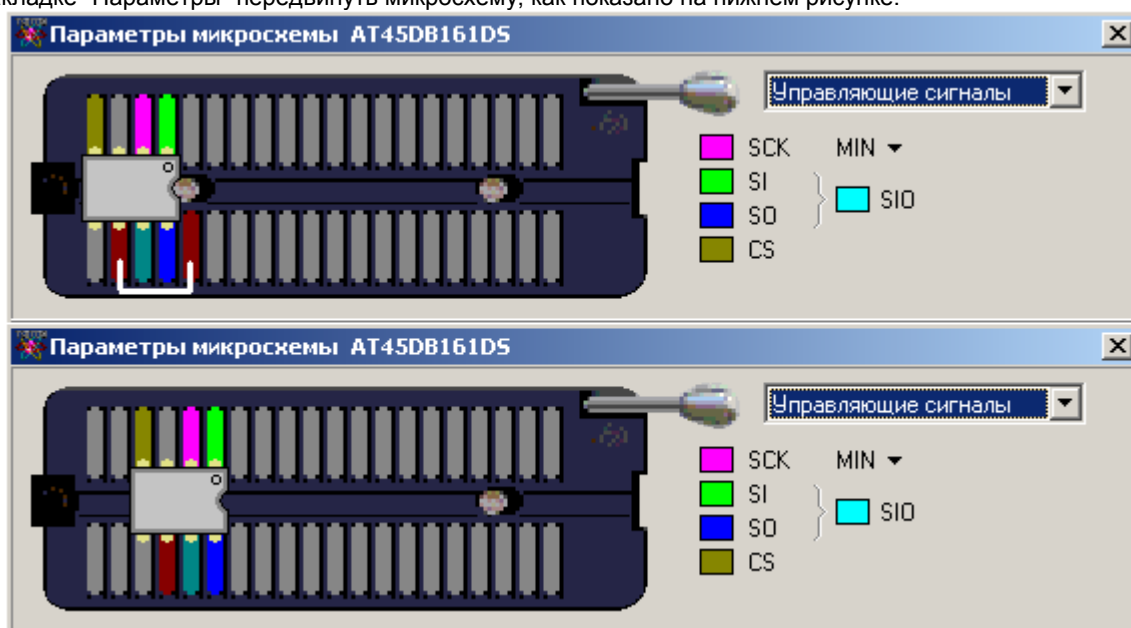
Для микросхем DATA-Flash в окнах адресов (начальный, конечный) на панели управления параметрами микросхемы и в окне сообщений об ошибке выводится не физический адрес, а HOME сектора. Размер сектора зависит от типа микросхемы и может быть равен 264, 528 или 1056 байт. Порядковый номер байта внутри сектора недоступен.

Программатор позволяет считать полностью все данные сектора или только основные (256, 512 или 1024 байта), пропустив служебную область. Для этого в параметрах микросхемы в ячейке \$16(число дополнительных байт), нужно установить значение \$00. Программировать микросхемы с такими установками не рекомендуется, так как содержимое дополнительной области будет не определено.

Возможные ошибки и методы их устранения.

AT45DBxxx 8-pin корпус. В на старых моделях программаторов, кроме V5.8TU и V5.7T, на 22 выводе панельки отсутствует ключ, который подает питание на эту микросхему. Для работы с такими микросхемами необходимо:

- или соединить 22 и 25 выводы панельки;
- или на закладке "Параметры" передвинуть микросхему, как показано на нижнем рисунке.



Микросхемы SEEPROM 17x

Выбор микросхемы.

Автоматическое определение для этих микросхем в программаторе не реализовано. Выбор таких микросхем должен осуществляться вручную.

Особенности работы.

Последовательность работы программатора в режимах чтения (проверка на чистоту, подсчет контрольной суммы, чтение, контрольная сверка):

- Чтение основной памяти.
- Проверка полярности сигнала "Reset" не выполняется.

Последовательность работы программатора в режиме записи микросхемы:

- Проверка микросхемы на чистоту;
- Запись основной памяти;
- Контроль качества записи;
- Установка полярности сигнала "Reset".

Переходные панельки.

Благодаря технологии универсальных алгоритмов, для работы с микросхемами могут быть использованы любые переходные панельки. По умолчанию используются Универсальные (с разводкой pin-to-pin) переходные панельки. При использовании других переходников, программное обеспечение позволяет переназначить сигналы в соответствии с разводкой используемого переходника.

Микросхемы SEEPROM 24x

Выбор микросхемы.

Микросхемы этой серии не содержат идентификационных кодов, поэтому автоматическое определение для них не возможно. Выбор таких микросхем должен осуществляться только вручную.

Особенности работы.

Микросхемы этой серии не имеют отдельного режима стирания. Встроенный в микросхему механизм записи предварительно стирает старые данные и затем программирует новые. Чтобы очистить (стереть) такие микросхемы нужно создать файл, состоящий из одного или нескольких байтов \$FF, и записать его в микросхему.

Перед записью программатор проверяет микросхему на чистоту или на возможность записи (в зависимости от флага "[При проверке на чистоту не загружать файл](#)") и, если микросхема уже содержит какую-либо информацию, выводит сообщение об ошибке и **ждет решение оператора** о начале записи или отмене операции. При установленном флаге "[Пропускать проверку на чистоту перед записью](#)", программатор пропускает цикл проверки и сразу начинает программирование микросхемы.

Микросхемы этой серии имеют страничную запись. Объем страницы установлен для каждой микросхемы согласно документации и может быть изменен на закладке "[Конфигурация](#)". Микросхемы разных производителей могут иметь разный размер страницы, поэтому, если при записи аналога возникают ошибки, то надо уменьшить размер страницы.

Переходные панельки.

Благодаря технологии универсальных алгоритмов, для работы с микросхемами могут быть использованы любые переходные панельки. По умолчанию используются Универсальные (с разводкой pin-to-pin) переходные панельки. При использовании других переходников, программное обеспечение позволяет переназначить сигналы в соответствии с разводкой используемого переходника.

Микросхемы SEEPROM 25x

Выбор микросхемы.

В программе доступно автоматическое определение (чтение ID) микросхемы. Запуск скрипта чтения выполняется при нажатии кнопки "Определить"; в окне выбора микросхемы. Для корректной работы должен быть выбран соответствующий скрипт для 8- или 16- выводного корпуса.

Старые микросхемы малых объемов, выполненные по технологии EEPROM, не содержат идентификационных кодов, поэтому выбор таких микросхем должен осуществляться вручную. Также, в этих микросхемах, часто нет режима стирания, поэтому кнопка "Стирание" недоступна. Запись новой информации в этих микросхемах осуществляется поверх старой.

Особенности работы.

Последовательность работы программатора в режимах чтения (проверка на чистоту, подсчет контрольной суммы, чтение, контрольная сверка):

- Чтение основной памяти;
- Чтение выбранной страницы OTP (если разрешен доступ);
- Чтение Status Register, если разрешен доступ (при проверке на чистоту не обрабатываются).

Последовательность работы программатора в режиме записи микросхемы:

- Стирание и проверка микросхемы на чистоту (если разрешен доступ к этим режимам);
- Программирование основной памяти;
- Программирование выбранной страницы OTP (если разрешен доступ);
- Контроль качества записи основной памяти и выбранной страницы OTP;
- Запись Status и Config Register.

В современных микросхемах SPI-Flash, кроме основной памяти, часто имеется до 4 дополнительных страниц, объемом 256, 512 или 1024 байта. В окне выбора микросхемы программа показывает название, организацию и размер дополнительной области для каждой микросхемы. Если адреса размещения этих страниц идут подряд, то программатор читает и пишет все страницы единым блоком. Но часто, адреса идут с разрывом и программатор обрабатывает каждую страницу отдельно. Выбор страницы доступен на закладке "Буфер", в дополнительной области данных. Для некоторых микросхем эта область однократно-программируемая и в программе кнопка стирания будет неактивна. Если в микросхеме не установлена защита от перезаписи этой области, то программатор может изменить в OTP области значение 1 в 0, но не наоборот.

Также, большинство микросхем SPI-Flash содержат один или более Status или Config регистров, которые позволяют защитить содержимое различных областей памяти и задать конфигурацию микросхемы. При стирании и записи микросхемы, защита основной памяти отключается и эти биты сбрасываются в ноль, независимо от состояния флагов на закладке Конфигурация. Если в микросхеме имеется два типа Status регистра (Volatile и Nonvolatile), то защита отключается через Volatile регистр, а значение Nonvolatile регистра при этом не меняется.

Часто, часть битов в этих регистрах является однократно-программируемыми, т.е. после записи изменить их значение невозможно. В программе, в конце описания таких битов, выводится слово (OTP). Программа позволяет считать и записать все регистры в процессе чтения и записи всей микросхемы. Для микросхем, которые содержат режим ASP (Advanced sector protection), в режиме чтения программатор читает BCE регистры, но при записи всей микросхемы записывает только перезаписываемые регистры. OTP регистры программируются отдельной кнопкой "Write OTP Registers".

The screenshot shows a software configuration window for SEEPROM 25x chips. It is divided into two main sections: "Status and Configuration Registers" and "Configuration Registers (OTP)".

Status and Configuration Registers:

- There are four checkboxes for SRWD, BP2, BP1, and BP0, each with a corresponding "Write Status Reg." button.
- Below these are two rows of labels: "Lower Die" and "Upper Die", each with its own set of SRWD, BP2, BP1, and BP0 checkboxes.

Configuration Registers (OTP):

- There are several checkboxes for TBPRO, BPNV, TBPAR, QUAD, AL, QA, IO3R, RL3, RL2, RL1, RL0, BC, O2h, 20h, F0h, OI3, OI2, OI1, WE, WL1, and WL0.
- Below these are three input fields: "Data Learning Reg:" with the value "F0", "Password Register:" with the value "0123456789ABCDEF", and "ASP Register:" with a dropdown menu showing "PWDMLB".

Возможные ошибки и методы их устранения.

Некоторые микросхемы для стабильной работы требуют обязательную установку блокировочной емкости (электролит,

тантал, керамика) номиналом от 1мкF до 4,7мкF на выводы питания микросхемы. На программаторах V5.7 установка конденсаторов обязательна для микросхем фирм EON EN25xxx, MACRONIX MX25Lxxxx и SPANSION S25FSxxx. Эти микросхемы могут нормально работать в режиме чтения, но глючить при стирании или записи. На программаторах V5.8T установка конденсаторов обязательна для 1.8V SPANSION S25FSxxx и S70FSxxx во всех режимах.

На программаторах V5.8T, если микросхема читается с ошибками, то можно снизить тактовую частоту или повысить напряжение питания. В программе на закладке Параметры, в Управляющих сигналах можно выбрать частоту от 2 до 50МГц.

Микросхемы SEEPROM 93x

Выбор микросхемы.

Микросхемы этой серии не содержат идентификационных кодов, поэтому автоматическое определение для них не возможно. Выбор таких микросхем должен осуществляться только вручную.

Особенности работы.

Последовательность работы программатора в режимах чтения (проверка на чистоту, подсчет контрольной суммы, чтение, контрольная сверка):

- Чтение основной памяти.

Последовательность работы программатора в режиме записи микросхемы:

- Стирание и проверка микросхемы на чистоту;
- Запись основной памяти;
- Контроль качества записи;
- Запись Protect register (только для 93CSx).

Переходные панельки.

Благодаря технологии универсальных алгоритмов, для работы с микросхемами могут быть использованы любые переходные панельки. По умолчанию используются Универсальные (с разводкой pin-to-pin) переходные панельки. При использовании других переходников, программное обеспечение позволяет переназначить сигналы в соответствии с разводкой используемого переходника.

Микроконтроллеры AVR

Выбор микросхемы.

Автоматическое определение для этих микросхем в программаторе не реализовано. Выбор таких микросхем должен осуществляться вручную.

Особенности работы.

При работе с AVR в программаторе используется параллельный (12V) или последовательный (ISP) режим программирования. Выбор режима осуществляется пользователем. По умолчанию, при выборе микросхемы устанавливается параллельный режим программирования. Для работы в режиме ISP, если микросхема находится в панельке программатора, то она должна быть сконфигурирована для работы от внутреннего генератора. Если микросхема не имеет внутреннего генератора, то программатор допускает подачу на вход XTAL тактовой частоты от внешнего источника.

Последовательность работы программатора в режимах чтения (проверка на чистоту, подсчет контрольной суммы, чтение, контрольная сверка):

- Чтение основной памяти;
- Чтение EEPROM data;
- Чтение Fuse bits (при проверке на чистоту Fuse bits не обрабатываются);
- Чтение Lock bits, контроль установленных битов защиты.

Последовательность работы программатора в режиме записи микросхемы:

- Стирание и проверка микросхемы на чистоту (Fuse bits не обрабатываются);
- Запись основной памяти;
- Запись Fuse bits (установка бита RSTDISBL или смена типа генератора при ISP может заблокировать дальнейшую работу с микросхемой);
- Запись EEPROM data;
- Контроль качества записи (Lock bits не обрабатываются);
- Запись Lock bits - установка и контроль защиты.

Доступ к EEPROM возможен как при работе со всей микросхемой, так и с помощью отдельных команд чтения/записи. При работе с EEPROM возможно использование, как отдельного файла, так и одного общего файла для программы и для данных. В одном общем файле EEPROM данные должны располагаться, сразу после последнего адреса памяти программ. Так как каждый адрес микросхемы в файле занимает два байта, то для микросхемы AT90S1200 с конечным адресом, равным \$01FF, адрес начала EEPROM данных будет равен \$0400.

В режимах чтения Fuse и Lock bits считываются и обрабатываются отдельно, в зависимости от установленных флагов. При выводе сообщений на экран для индикации Fuse и Lock bits используются следующие адреса:

- Ext_Fuse_bits – \$03FFFF;
- High_Fuse_bits – \$02FFFF;
- Low_Fuse_bits – \$01FFFF;
- Lock_bits – \$00FFFF;

При чтении микросхем программатор постоянно контролирует состояние всех битов защиты и, если хотя бы один из них установлен, то выводится сообщение "Chip is Locked".

На закладке "Конфигурация" в поле "Fuse bits" установленная галочка означает запрограммированный (установленный в ноль) Fuse-бит. Дополнительно, в скобках показывается шестнадцатеричное значение всех Fuses.

Особенности работы в автономном режиме.

При выборе микросхемы на программаторе, разрешается доступ к EEPROM и FUSE bits. Флаг доступа к Lock bits сбрасывается. Все Fuse bits устанавливаются в начальное состояние, в соответствии с фирменной документацией.

Программаторы ТРИТОН+ в автономном режиме позволяют считать незащищенную микросхему, разрешить установку битов защиты (кнопки 'Menu', 'Utl'+ 'Lck') и дублировать микросхемы с установленной защитой. В этом случае устанавливаются ВСЕ биты защиты, которые есть в микросхеме.

Микроконтроллеры Holtek

Выбор микросхемы.

Выбор этих микросхем осуществляется только вручную.

Внимание: Микроконтроллеры Holtek устанавливаются в панельку программатора в крайнее правое положение, ключом вправо.

16, 18 и 20-выводные микросхемы при этом сдвигаются на два вывода влево, первый вывод микросхемы устанавливается в третий вывод панельки.

Особенности работы.

При работе с MCU Holtek необходимо учитывать, что чистое состояние этих микросхем – нули, а запрограммированный бит – единица. При установленном бите защиты память микросхемы считывается как \$FF. Чтение и запись Option ROM возможна и при установленном бите защиты.

В программе не предусмотрена модификация битов конфигурации в Options ROM. Эти данные загружаются из исходного "*.otr" файла, и могут быть впоследствии изменены непосредственно в редакторе. Оригинальный "*.otr" файл кроме данных для записи содержит и служебную информацию. Поэтому при открытии таких файлов действуют следующие правила: если выбран микроконтроллер HOLTEK, то из файла извлекается вся необходимая информация, в остальных случаях данный файл открывается как обычный двоичный файл, т.е. без преобразования.

Последовательность работы программатора в режимах чтения (проверка на чистоту, подсчет контрольной суммы, чтение, контрольная сверка):

- Чтение основной памяти;
- Чтение Options ROM.

Последовательность работы программатора в режиме записи микросхемы:

- Проверка всей микросхемы на чистоту;
- Запись основной памяти;
- Контроль качества записи памяти программ;
- Запись Options ROM.

Особенности работы в автономном режиме.

Программаторы ТРИТОН+ в автономном режиме позволяют только дублировать микросхемы, и не поддерживают модификацию бита защиты для микроконтроллеров HOLTEK. Все необходимые установки должны быть выполнены на компьютере, после чего загружены в программатор в виде проекта.

Микроконтроллеры MSC-51

Выбор микросхемы.

Для этих микросхем доступно автоматическое определение типа. Для этого в окне выбора микросхемы нужно выбрать фирму "Auto detect" и нажать кнопку "Определить". Многие современные микроконтроллеры имеют различные алгоритмы чтения сигнатуры, существенно отличающиеся от стандартного, поэтому они должны выбираться вручную. Не определяются следующие типы микросхем:

- ATMEL: AT89C51Rx2, 55WD, S51, S52...;
- ATMEL: старые AT89C51 с установленными битами защиты;
- PHILIPS: P89C5x;

Особенности работы.

Последовательность работы программатора в режимах чтения (проверка на чистоту, подсчет контрольной суммы, чтение, контрольная сверка):

- Чтение основной памяти;
- Чтение, если возможно, Fuse bits (при проверке на чистоту не обрабатываются);
- Проверка, если возможно, установленных битов защиты.

Последовательность работы программатора в режиме записи микросхемы:

- Стирание микросхемы, если возможно;
- Проверка микросхемы на чистоту;
- Запись основной памяти;
- Запись, если возможно, Fuse bits;
- Контроль качества записи;
- Программирование, если есть, шифровальной таблицы;
- Установка битов защиты.

При установленных битах защиты содержимое микросхемы считывается как \$FF. Для некоторых современных микросхем (ATMEL, PHILIPS) возможен контроль состояния битов защиты и, если хотя бы один из них установлен, то выводится сообщение "Chip is Locked".

87C5x: Запись шифровальной таблицы в процессе программирования всей микросхемы возможна только вместе с записью битов защиты. В тоже время возможна только установка битов защиты. Кроме того возможна отдельная запись шифровальной таблицы или установка битов защиты. Для сверки содержимого микросхемы и исходным файлом, при запрограммированной шифровальной таблице можно воспользоваться функциями шестнадцатеричного редактора. Необходимо выделить всю шифровальную таблицу, скопировать ее, выделить необходимую часть исходного файла, в меню Функция выбрать команду XNOR, а в качестве маски указать "Buffer" и нажать "Ok". Теперь можно произвести контрольную сверку.

AT89S5x, AT89C55WD: Эти микросхемы для стирания и установки защиты требуют напряжение питания 6,5v. В программаторе это напряжение формируется из напряжения питания микросхемы во время записи (Vccswg) + 1,5v. Таким образом, изменяя значение напряжения Vccswg можно регулировать напряжение питания микросхемы в этих режимах.

AT89S8252: Доступ к EEPROM осуществляется также как и к основной памяти программ. При выборе микросхемы конечный адрес устанавливается равным \$0027FF, разрешая тем самым доступ к EEPROM.

AT89S8253: При необходимости работать с USER ROW (чтение-запись) необходимо установить конечный адрес = 00003F, затем в параметрах микросхемы в ячейках \$28, \$29 изменить значения с \$F3 и \$F7 на \$F2 и FB. Сохранить изменения. Пользоваться командами READ CHIP и PROGRAMM.

AT89Sx051: При необходимости работать с USER ROW (чтение-запись) необходимо установить конечный адрес = 00001F, затем в параметрах микросхемы изменить значения в ячейках \$28(\$0C на \$08) и \$29(\$0E на \$00). Сохранить изменения. Пользоваться командами READ CHIP и PROGRAMM.

P89C51Rx+: Эти микросхемы имеют в конфигурационной области два байта Boot Vector byte и Status byte. Если значение Status byte =0, то после рестарта процессора программа начинается с адреса 0. Если значение Status byte >0, то после рестарта программа перейдет на адрес, заданный в Boot Vector byte. Адрес перехода кратен 256 байтам и Boot Vector byte задает старший байт этого адреса.

P89C5xX2: Эти микросхемы имеют конфигурационный бит, установка которого переводит микросхему в 6-тактный режим. При стирании микросхемы этот бит сбрасывается и включает 12-тактный режим.

P89C6xX2, P89C51Rx2: Эти микросхемы дополнительно имеют следующие возможности: 6-тактный режим, Status Byte и Boot Vector. При стирании, значения Status Byte и Boot Vector не меняются, микросхема переводится в 12-тактный режим. При переключении микросхемы в 6-clk режим производится запись Status Byte в 0, тем самым, разрешая выполнение пользовательской программы. Модификация Boot Vector в данной версии программы не поддерживается.

Особенности работы в автономном режиме.

Программаторы ТРИТОН+ в автономном режиме позволяют считать незащищенную микросхему, разрешить установку битов защиты (кнопки 'Menu', 'Utl'+ 'Lck') и дублировать микросхемы с установленной защитой. В этом случае устанавливаются ВСЕ биты защиты, которые есть в микросхеме.

Переходные панельки.

Для работы с микросхемами в корпусах, отличных от DIP, могут быть использованы стандартные переходные панельки для микроконтроллеров MCS-51 от любых программаторов.

Микроконтроллеры Philips P89LPC9xx

Выбор микросхемы.

Микросхемы этой серии не содержат идентификационных кодов, поэтому выбор этих микросхем осуществляется только вручную.

Особенности работы.

Для обеспечения максимальной секретности микросхемы этого семейства не имеют режимов чтения, поэтому кнопки чтения микросхемы и проверки на чистоту не работают. Для контроля записанной информации фирма Philips рекомендует использовать встроенные команды для подсчета CRC всей микросхемы или конкретного блока. При верификации программатор подсчитывает контрольную сумму загружаемого файла и сверяет ее с контрольной суммой всей микросхемы. После записи микросхем P89LPC9xx на экран выводятся младшие 16-бит от 32-битной контрольной суммы, подсчитанной согласно спецификации Philips.

Последовательность работы программатора в режиме записи микросхемы (используется режим последовательного внутрисхемного программирования – ISP):

- Стирание микросхемы (контролируется только флаг готовности и флаги ошибок);
- Запись основной памяти;
- Подсчет и сравнение CRC файла и всей микросхемы;
- Запись конфигурационной области.

Для работы с P89LPC906..908 необходимо на панельке программатора соединить между собой выводы 22 и 25. (для всех моделей программаторов, кроме V5.7T и V5.8T)

Микроконтроллеры Winbond

Выбор микросхемы.

Микросхемы этой серии не содержат идентификационных кодов, поэтому автоматическое определение для них не возможно. Выбор таких микросхем должен осуществляться только вручную.

Особенности работы.

При установленных битах защиты содержимое микросхемы и Security Register считывается как \$FF. Контроль состояния бита защиты в этих микросхемах недоступен.

Последовательность работы программатора в режимах чтения (проверка на чистоту, подсчет контрольной суммы, чтение, контрольная сверка):

- Чтение основной памяти;
- Чтение LD ROM (только для W78E58B и W78E516);
- Чтение Security Register (если установлена защита, считывается FFh).

Последовательность работы программатора в режиме записи микросхемы:

- Стирание и проверка всей микросхемы на чистоту;
- Запись основной памяти;
- Запись LD ROM (только для W78E58B и W78E516);
- Контроль качества записи (Security Register не обрабатывается);
- Запись 2 байт Seed_0 и Seed_1 (кроме W78E58B и W78E516);
- Запись Security Register.

Доступ к LD ROM возможен как при работе со всей микросхемой, так и с помощью отдельных команд чтения/записи. Данные для LD ROM могут находиться как в отдельном файле, так и в основном файле программы, начиная с адреса \$10000 (по умолчанию). При чтении микросхемы и LD ROM данных всегда создается общий файл.

Запись 2 байт Seed_0 и Seed_1 для микросхем W7x* производится всегда перед записью Security Register. Для отключения шифрования считываемых данных значения Seed_0 и Seed_1 должны быть равны \$FF. Если производится повторная запись в Security Register в котором уже установлен флаг "Lock Seed" и значения Seed_0 и Seed_1 отличаются от \$FF, то программа выведет сообщение об ошибке.

Особенности работы в автономном режиме.

Программаторы ТРИТОН+ в автономном режиме позволяют разрешить или запретить запись Security Register (кнопки 'Menu', 'Util'+ 'Lck'). При этом значении Security Register должно быть сформировано на компьютере. При выборе микросхемы на программаторе значение Security Register устанавливается в состояние \$FF.

Переходные панельки.

Для работы с микросхемами в корпусах, отличных от DIP, могут быть использованы стандартные переходные панельки для микроконтроллеров MCS-51 от любых программаторов.

Микросхемы Pic10xxx, Pic12xxx, Pic16C5x

Выбор микросхемы.

Автоматическое определение для этих микросхем в программаторе не реализовано. Выбор таких микросхем должен осуществляться вручную.

Особенности работы.

- Pic10F2xx;
- Pic12C508/509, Pic12CE518/519;
- Pic12F508/509/510;
- Pic16C5x;
- Pic16C505;
- Pic16F5x;
- Pic16F505/506.

Последовательность работы программатора в режимах чтения (проверка на чистоту, подсчет контрольной суммы, чтение, контрольная сверка):

- Чтение конфигурационного слова и проверка битов защиты;
- Чтение основной памяти;
- Чтение калибровочной области, если установлен флаг "Calibration";
- Чтение ID Locations;
- Если разрешен доступ, то чтение Backup OSCCAL bits.

Последовательность работы программатора в режиме записи микросхемы:

- Стирание микросхемы и восстановление калибровочных коэффициентов;
- Проверка всей микросхемы на чистоту;
- Запись основной памяти;
- Запись калибровочной области, если установлен флаг "Calibration";
- Запись ID Locations (и, если разрешено записи Backup OSCCAL bits);
- Контроль качества записи (конфигурационное слово не обрабатывается);
- Запись конфигурационного слова.

При выборе микросхемы флаг доступа к "Calibration data" сбрасывается, и обработка этих данных производится программатором автоматически. В режиме стирания программатор считывает калибровочную константу, стирает микросхему и затем записывает считанную константу. При завершении цикла значение константы выводится в окне сообщений. Если флаг "Calibration" установлен, то в режиме стирания чтение калибровочной константы не выполняется и производится полное стирание микросхемы, а в последующем цикле программирования в микросхему будут записаны данные, заданные пользователем или загруженные из файла при его открытии. При этом в окне сообщений выводится значение константы, с пометкой "user value".

Чтобы получить доступ к Backup OSCCAL bits для чтения или записи необходимо в параметрах микросхемы в ячейке \$17 (Кол-во слов конфигурации) установить значение \$02 и разрешить доступ к ID_Locations. Ячейки для хранения Backup OSCCAL bits в блоке параметров микросхемы = \$3C и \$3D.

Pic10F2xx, Pic12F5xx, Pic16F5xx: Область ID_Locations может быть запрограммирована только один раз. После чего в обычном режиме будет стираться только память программ и конфигурационное слово. ID_Locations стираться не будут. Для стирания всей микросхемы необходимо установить флаг "Calibration data" и дать команду стирания. Будут стерты вся память программ, конфигурационное слово, калибровочные данные, ID_Locations и Backup OSCCAL bits.

Особенности работы в автономном режиме.

При выборе микросхемы на программаторе флаг доступа к "Calibration data" сбрасывается, и обработка этих данных производится программатором автоматически.

В автономном режиме программатор не может считать конфигурационное слово или ID Locations из отдельно загруженного файла. Для тиражирования микросхем в автономном режиме все настройки должны быть выполнены на компьютере, и загружены в программатор в виде проекта.

Программаторы ТРИТОН+ в автономном режиме позволяют считать незащищенную микросхему, разрешить установку битов защиты и дублировать микросхемы с установленной защитой. В этом случае устанавливаются ВСЕ биты защиты, которые есть в микросхеме. Для этого выбирается микросхема и разрешается доступ к конфигурационному слову (кнопки 'Menu', 'Utl'+ 'Lck'). После чего производится чтение микросхемы. Затем повторно разрешаем доступ к конфигурационному слову (кнопки 'Menu', 'Utl'+ 'Lck') и тем самым, устанавливаем биты защиты. Теперь можно запрограммировать микросхемы.

Переходные панельки.

Для работы с микросхемами в корпусах, отличных от DIP, могут быть использованы стандартные переходные панельки от любых программаторов.

При использовании специальных переходников программное обеспечение позволяет переназначить сигналы на панельке

программатора в соответствии с разводкой используемого переходника.

Возможные ошибки и методы их устранения.

При записи конфигурационного слова для Pic12C5x (16C505), которое считано из файла, и имеет установленные биты защиты, может выводиться сообщение об ошибке (“Ошибка записи микросхемы, Число ошибок - 0”). При этом запись будет произведена правильно. Это связано с тем, что при установленной защите старшие биты конфигурационного слова считываются нулями, а MPLAB, почему-то, упорно формирует их как единицы. Во избежание таких сообщений рекомендуется создавать код прошивки с выключенными битами защиты, и уже в оболочке программатора разрешать защиту.

Программатором не поддерживается для старых Pic16C5x (при установленных битах защиты!!!) следующие режимы работы:

- дозапись незащищенного сегмента памяти программ;
- правильный подсчет контрольной суммы (согласно спецификации Microchip);

Микросхемы Pic12F6xx, Pic16Cxxx, Pic16Fxxx

Выбор микросхемы.

Автоматическое определение для этих микросхем в программаторе не реализовано. Выбор таких микросхем должен осуществляться вручную.

Особенности работы.

•

Последовательность работы программатора в режимах чтения (проверка на чистоту, подсчет контрольной суммы, чтение, контрольная сверка):

- Чтение основной памяти;
- Чтение EEPROM data;
- Чтение ID Locations;
- Чтение конфигурационного слова и проверка битов защиты.

Последовательность работы программатора в режиме записи микросхемы:

- Стирание микросхемы и восстановление калибровочных коэффициентов;
- Проверка всей микросхемы на чистоту;
- Запись основной памяти;
- Запись ID Locations;
- Запись EEPROM data;
- Контроль качества записи. Проверяется память программ, ID Locations и EEPROM. Конфигурационное слово не обрабатывается;
- Запись конфигурационного слова.

Pic12F629, Pic12F675, Pic16F630, Pic16F676. Восстановление калибровочных коэффициентов производится в одном цикле вместе со стиранием. Это сложная процедура, которая требует отключения питания и повторного входа в режим программирования, поэтому не исключена возможность потери заводских установок. Для уменьшения вероятности ошибки, рекомендуется не использовать отдельную команду стирания, а стирать микросхему перед записью во время программирования. При этом, если установлен флаг "Вести журнал запрограммированных микросхем" программа считывает из микросхемы значение Calibration Data и сохраняет их в одноименном *.log файле, после чего производит стирание микросхемы. При этом, в случае сбоя, есть возможность восстановить калибровочные значения.

При выборе микросхемы флаг доступа к "Calibration data" сбрасывается, и обработка этих данных самостоятельно производится программатором. Если флаг "Calibration data" установлен, то в микросхему будут записаны данные, заданные пользователем или загруженные из файла при его открытии.

Pic16F6xx, Pic16F8xx. Большинство этих микросхем поддерживают два режима записи памяти программ: просто запись ячейки с внешним управлением длительностью (Begin Programming и End Programming), и запись с предварительным стиранием ячейки (Begin Erase Programming). Для некоторых микросхем использование команд с внешним управлением длительностью записи может дать четырехкратный выигрыш по времени. Программное обеспечение позволяет менять коды этих команд, а также менять код команды стирания. В блоке параметров микросхем это первые три ячейки. Для информации о кодах этих команд и длительности цикла записи необходимо смотреть спецификацию по программированию на конкретную микросхему.

Доступ к EEPROM возможен как при работе со всей микросхемой, так и с помощью отдельных команд чтения/записи. При работе с EEPROM возможно использование, как отдельного файла, так и одного общего файла для программы и для данных. EEPROM данные в общем файле должны располагаться, начиная с адреса \$4200.

Некоторые микросхемы имеют два или три конфигурационных слова для хранения калибровочных данных. Чтобы получить к ним доступ для чтения или записи необходимо в параметрах микросхемы в ячейке \$17 (Кол-во слов конфигурации) установить значение \$02 или \$03. Ячейки для хранения этих настроек: \$30..31 для адреса \$2008, и \$32..33 для \$2009. Если открываемый файл содержит калибровочные данные для второго и третьего конфигурационного слова, то они будут записаны в указанные ячейки в блоке параметров микросхемы. При необходимости подстройки этих значений можно в файл P16.CFG добавить описания и параметры дополнительных FUSES, которые после повторного выбора микросхемы будут выведены на панель управления. Описание формата файла P16.CFG приведено в заголовке этого файла.

В конфигурационном слове многих микросхем имеются неиспользуемые биты, которые всегда читаются как нули. Однако MPLAB всегда формирует эти биты как единицы. Соответственно, при записи может возникать ошибка ("Ошибка записи микросхемы Число ошибок - 0"). Во избежание таких ошибок рекомендуется создавать код прошивки с выключенными битами защиты, и уже в оболочке программатора разрешать защиту.

Особенности работы в автономном режиме.

В автономном режиме программатор:

- не может считать конфигурационное слово или ID Locations из отдельно загруженного файла.
- не поддерживает отдельные файлы для EEPROM памяти данных.

Pic12F629/F675, Pic16F630/F676. При выборе микросхемы на программаторе флаг доступа к "Calibration data" сбрасывается, и обработка этих данных самостоятельно производится программатором.

Для тиражирования микросхем в автономном режиме все настройки должны быть выполнены на компьютере, и загружены в программатор в виде проекта.

Программаторы ТРИТОН+ в автономном режиме позволяют считать незащищенную микросхему, разрешить установку битов защиты (кнопки 'Menu', 'Utl'+ 'Lck') и дублировать микросхемы с установленной защитой. В этом случае устанавливаются ВСЕ биты защиты, которые есть в микросхеме. Для этого выбирается микросхема и разрешается доступ к конфигурационному слову (кнопки 'Menu', 'Utl'+ 'Lck'). После чего производится чтение микросхемы. Затем повторно разрешаем доступ к конфигурационному слову (кнопки 'Menu', 'Utl'+ 'Lck') и тем самым устанавливаем биты защиты. Теперь можно программировать микросхемы.

Переходные панельки.

Для работы с микросхемами в корпусах, отличных от DIP, могут быть использованы стандартные переходные панельки от любых программаторов. При использовании специальных переходников программное обеспечение позволяет переназначить сигналы на панельке программатора в соответствии с разводкой используемого переходника.

Для работы с микросхемами PIC16C92x используется специальный переходник TST-D40/PL68-PIC. Разводку переходника можно посмотреть в окне выбора микросхемы.

Возможные ошибки и методы их устранения.

Pic12F6xx, Pic16Fxxx. Во время стирания микросхемы, если не установлены биты защиты, то область EEPROM не стирается. При этом если разрешен доступ к EEPROM, программатор выведет сообщение об ошибке и предложит стереть микросхему снова или начать запись. Повторное стирание ничего не изменит, поэтому надо просто нажать кнопку "Начать запись".

Pic12F629/F675, Pic16F630/F676. В конфигурационном слове этих микросхем имеется несколько неиспользуемых битов, которые всегда читаются как нули. Соответственно при стирании микросхемы и проверке на чистоту возникает ошибка ("Микросхема не стирается или не чистая, Ошибка по адресу \$002007"). Нужно просто игнорировать это сообщение или использовать команду "PROGRAMM" с установленным флагом "Стирать микросхему перед записью".

Pic16F87,88 и Pic16F7x7. При подсчете контрольной суммы не маскируется последний байт CFG word, в соответствии с документацией MICROCHIP.

Pic16C61,62,64,65,71,73,74,C84. При установленных битах защиты программатором не поддерживаются следующие режимы работы:

- дозапись незащищенного сегмента памяти программ;
- правильный подсчет контрольной суммы (согласно спецификации Microchip);
- только для Pic16C61,71,C84 при записи конфигурационного слова с установленным битом защиты, неправильно работает контроль записи и выводится сообщение об ошибке, хотя сами данные записываются правильно (не работает режим Scrambled).

Микросхемы Pic17

Выбор микросхемы.

Микросхемы этой серии не содержат идентификационных кодов, поэтому автоматическое определение для них не возможно. Выбор таких микросхем должен осуществляться только вручную.

Особенности работы.

Последовательность работы программатора в режимах чтения (проверка на чистоту, подсчет контрольной суммы, чтение, контрольная сверка):

- Чтение основной памяти;
- Чтение конфигурационного слова.

Последовательность работы программатора в режиме записи микросхемы:

- Проверка всей микросхемы на чистоту;
- Запись основной памяти;
- Контроль качества записи;
- Запись конфигурационного слова.

Согласно спецификации фирмы "Microchip" информация о конфигурационном слове в HEX файле должна находиться, начиная с адреса \$1FC00. При чтении микросхемы информация о конфигурационном слове помещается, начиная с адреса \$1FC00.

Переходные панельки.

Для работы с микросхемами PIC17C75х используются специальные переходники с DIP40 на PLCC68 или PLCC84. Разводку этих переходников можно посмотреть в окне выбора микросхемы.

Микросхемы Pic18C, Pic18F, Pic24, dsPIC

Выбор микросхемы.

Автоматическое определение для этих микросхем в программаторе не реализовано. Выбор таких микросхем должен осуществляться вручную.

Особенности работы.

Последовательность работы программатора в режимах чтения (проверка на чистоту, подсчет контрольной суммы, чтение, контрольная сверка):

- Чтение основной памяти;
- Чтение EEPROM data (только для Pic18Fxxx);
- Чтение ID Locations;
- Чтение конфигурационной области.

Последовательность работы программатора в режиме записи микросхемы:

- Стирание и проверка всей микросхемы на чистоту;
- Запись основной памяти;
- Запись ID Locations;
- Запись EEPROM data (только для Pic18Fxxx);
- Контроль качества записи. Проверяется память программ, ID Locations и EEPROM. Область конфигурационного слова не обрабатывается;
- Запись конфигурационной области.

Pic18Cxxx. Согласно спецификации фирмы “Microchip” информация о конфигурационном слове в HEX файле должна находиться, начиная с адреса \$1FC00. При чтении микросхемы информация о конфигурационном слове помещается, начиная с адреса \$1FC00. Данные об ID Location в файле не сохраняются.

Pic18F44J10 и Pic18F45J10. При работе с микросхемами, между выводом 6 V_{сар} и землей (выводы 12 и 31) должен быть установлен конденсатор 4,7...10mkF.

Особенности работы в автономном режиме.

Для работы на программаторах ТРИТОН+ в автономном режиме все необходимые настройки должны быть выполнены на компьютере и загружены в программатор в виде проекта. При разрешении или запрете защиты (кнопки ‘Menu’, ‘Utl’+‘Lck’) в автономном режиме, разрешается или запрещается работа со всей областью конфигурации. Модификация битов защиты не производится.

Переходные панельки.

Для работы с микросхемами в корпусах, отличных от DIP, могут быть использованы стандартные переходные панельки от любых программаторов. При использовании специальных переходников программное обеспечение позволяет переназначить сигналы на панельке программатора в соответствии с разводкой используемого переходника. Для работы с микросхемами в корпусах TQFP-64 и TQFP-80 используются специальные переходники, разводку которых можно посмотреть в окне выбора микросхемы.

Для работы с Pic24FJ* в корпусе TQFP-64 необходимо доработать адаптер TSH-D40/TQ64-05:

- на подставке снизу отрезать дорожки, идущие к выводам 56 и 57;
- вывод 56 соединить с выводом 28 DIP-40;
- вывод 57 соединить с выводом 30 DIP-40;
- вывод 11 соединить с выводом 7 DIP-40 (доработка не обязательна);
- вывод 12 соединить с выводом 8 DIP-40 (доработка не обязательна).

Возможные ошибки и методы их устранения.

Pic18Fxxx. При записи конфигурационного слова с установленным битом WRTC (бит защиты конфигурации) в микросхеме блокируется дальнейшая запись и программатор выводит сообщение об ошибке. Чтобы избежать этого сначала программируются все биты конфигурационного слова с выключенным битом WRTC, после чего производится повторная запись, но уже с включенным битом WRTC.

При стирании микросхемы выдается сообщение “Ошибка стирания микросхемы. Адрес: 300000 0000 0000. Число ошибок - 14”. Конфигурационная область этих микросхем содержит неиспользуемые биты, которые всегда читаются как нули. Естественно, когда после стирания идет проверка на чистоту, то программатор сообщает об ошибке. Нужно просто игнорировать это сообщение или пользоваться флагом “Стирать микросхему перед записью”.

Программатор не стирает микросхемы, у которых установлены биты защиты. Это проявляется только на микросхемах PIC18Fxxxx, изготовленных по новой технологии, и при работе в панельке программатора. При внутрисхемной записи таких проблем обнаружено не было. Единственный способ решения этой проблемы – это установка блокировочной емкости (от 20nF до 10mkF) на выводы питания программируемой микросхемы.

Автономный режим

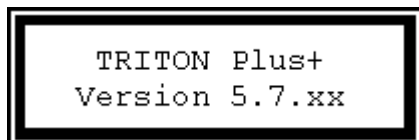
Программаторы Тритон+ могут работать без компьютера в автономном режиме. Автономный режим позволяет проводить все базовые операции: выбор микросхемы, файла или проекта, чтение микросхемы в память программатора, стирание, проверку на чистоту, программирование и сверку содержимого микросхемы и файла.

Управление ведется с помощью 4-кнопочной клавиатуры. Данные отображаются на LCD дисплее. В программаторах V5.7T для хранения данных используется встроенная память, организованная по типу Flash-диска. Для удобства работы в автономном режиме необходимо предварительно записать проект. Пользователь может добавить до 256 проектов, общим объемом до 4 МБайт.

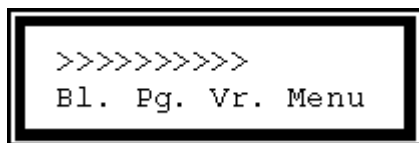
В программаторах V5.8T для хранения данных используются карты micro SD, отформатированные в FAT16 или FAT32, объемом до 32 ГБайт. Для использования SD карты в программаторе, предварительно, на компьютере на нее необходимо записать нужные файлы и проекты. Для выбора микросхем в автономном режиме, в корень карты необходимо записать файл **CHIPLIST.CHS**, который можно создать в оболочке программатора. В автономном режиме программатор поддерживает структуру каталогов, но не поддерживает длинные имена и национальные кодировки.



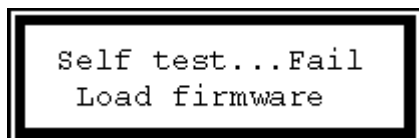
Подготовка к работе



Включите питание программатора. Через некоторое время (не более 0,5 сек) на экране появится сообщение и начнется тестирование программатора. Если проверка завершена успешно, во второй строке появится информация о версии программатора, и через 3 секунды программатор перейдет в режим ожидания.



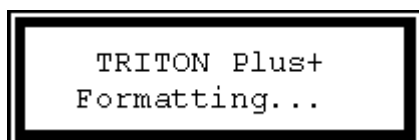
В режиме ожидания на экран выводится тип выбранной микросхемы, разрешается запуск команд и доступ к меню программатора. Только в этом режиме программатор будет отвечать на запросы компьютера. При нажатии и удержании кнопки 'Menu' на экран выводится имя файла и дополнительные три режима работы.



Если тест закончился неудачно, на экране появится сообщение об ошибке. Необходимо выключить питание программатора, подключить к компьютеру, и восстановить прошивку в программаторе (меню 'Программатор \ Обновление версии'). По окончании загрузки необходимо откалибровать напряжения питания и записи. Перед повторным включением программатора необходима пауза не менее

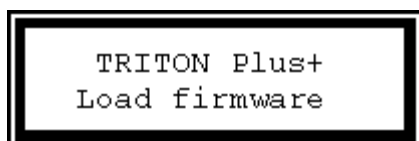
10 секунд, для сброса внутренних флагов процессора.

При включении питания с нажатыми кнопками, программатор позволяет запускать встроенные утилиты.



Форматирование Flash-диска. Только для программаторов V5.7T. Если включить питание программатора, удерживая все четыре кнопки, то будет произведено форматирование Flash-диска. Время форматирования зависит от объема Flash-диска и длится не более одной минуты. При этом из памяти программатора будут удалены все файлы, проекты и списки микросхем.

Восстановить файлы или проекты после форматирования невозможно. Если планируется выбор микросхем в автономном режиме, то [список микросхем](#) должен быть загружен из управляющей программы. Если планируется автономная работа только с проектами, то список можно не загружать или загрузить пустой список.

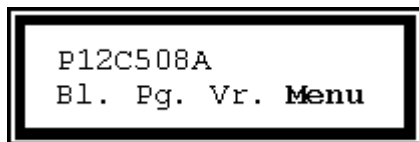


Восстановления прошивки. Включение питания с нажатыми кнопками 'Menu' и 'Vr.' - переводят программатор в режим [восстановления прошивки](#). Этот режим может потребоваться после неудачного обновления прошивки, когда программатор не реагирует на запросы компьютера.

Счетчик микросхем. После цикла записи на дисплей выводится количество успешно запрограммированных микросхем. Включение питания при нажатой кнопке 'Menu' обнуляет этот счетчик. В программаторе реализован только один счетчик, который подсчитывает общее число микросхем. Отдельная статистика по каждому проекту в автономном режиме невозможна.

Оперативная конфигурация. Программатор позволяет сохранить текущие настройки или открытый проект как оперативную конфигурацию и загружать ее **после** включения питания или рестарта. Загрузка производится при удержании кнопки 'Bl.' в момент, когда на экран выводится информация о версии программатора.

Назначение кнопок



Практически во всех режимах работы программатора в нижней строке экрана выводятся подсказки, определяющие назначение соответствующих клавиш на программаторе. Короткое нажатие запускает выбранный режим работы. Нажатие и удержание клавиши более 1,5сек. приводит к автоповтору клавиши, что очень удобно при просмотре списков.

В режиме ожидания кратковременное нажатие (до 1,5 сек) четвертой кнопки 'Menu', обеспечивает ввод команды (например: вход в меню). Более длительное нажатие - ничего не делает и, после отпускания, переводит программатор обратно в выбранный режим. При нажатой кнопке 'Menu' на экран выводится имя активного файла и разрешается доступ к дополнительным режимам работы.

Во время работы с микросхемами, или в меню, при выборе микросхем, файлов или конфигураций, кнопки выполняют следующие функции (слева на право):

- отмена, прекращение работы, переход в режим ожидания;
- пропуск ошибок при работе с м/сх, предыдущая позиция в списке;
- пропуск ошибок при работе с м/сх, следующая позиция в списке;
- подтверждение, повтор записи, пропуск ошибки, выбор позиции из списка.
- одновременное нажатие всех четырех кнопок приводит к сбросу программатора (т.н. горячий рестарт).

Выбор микросхемы и файла

```
Select:
Chp Fil Prj Utl
```

В начале необходимо выбрать микросхему. Для этого кратковременно нажмите кнопку 'Menu', затем кнопку 'Chp' - выбор микросхемы. Программатор предложит выбрать группу микросхем. В программаторе для ускорения поиска все микросхемы сгруппированы в шесть групп по типам (Memory, MCS-51, PIC, AVR, Serial EEPROM и Favorites). При этом возможна прокрутка всего списка.

```
Select type:
Mem Mcs Pic Esc
```

Первые три кнопки соответственно выбирают стандартные микросхемы памяти, однокристалки MCS-51 или PIC контроллеры. Эти же клавиши при нажатой четвертой кнопке 'Menu' выбирают AVR, Serial EEPROM или Favorites - список, сформированный пользователем.

```
Select chip:
>P12C508A      Ok
```

После выбора нужной группы, производится поиск микросхемы в списке. Левая кнопка - отмена режима и выход, две средние - прокрутка списка (для этих клавиш работает автоповтор), правая - выбор микросхемы.

Если предстоит работа со стандартной микросхемой памяти (EPROM или FLASH), то можно установить ее в панельку программатора и нажать кнопку 'Mem'. При этом программатор сам определит тип микросхемы, при условии, что микросхема поддерживает этот режим и она есть в списках программатора.

При выборе микросхемы, программатор разрешает доступ к дополнительным областям микросхемы, такие как EEPROM, FUSES..., но запрещает доступ битам защиты. При необходимости чтения или записи этих битов, если микросхема поддерживает такой режим, его необходимо включить (кнопки 'Menu', 'Utl'+Lck'). Внимание, для AVR и PIC контроллеров данный режим дополнительно модифицирует параметры микросхемы, устанавливая в конфигурационном слове все биты защиты.

Все прошивки хранятся в памяти программатора в виде двоичных файлов. Каждый файл может быть проектом и содержать данные о выбранной микросхеме, включая все настройки и индивидуальный алгоритм программирования. Файл может быть загружен с компьютера или будет создан при чтении микросхемы. В программаторе можно сохранить до 256 файлов, общим объемом до 4096кБайт. Из них 4кБайта отведено под список файлов (FAT) и от 2 до 128кБайт под список микросхем.

Чтобы выбрать файл из списка необходимо кратковременно нажать кнопку 'Menu', затем кнопку 'Fil' - выбор файла. Для открытия проекта - кнопку 'Menu', затем кнопку 'Prj' - выбор проекта. При просмотре списка файлов или проектов назначение кнопок такое же, как при выборе микросхемы. При выборе файла или проекта программатор всегда показывает один и тот же список. Когда открыт список файлов, то независимо от того, содержит ли файл проект или нет, данные проекта игнорируются, и открывается только файл. Когда открыт список проектов и, если файл содержит проект, то эти настройки будут загружены в память программатора, после чего будет открыт файл проекта. Если выбранный файл не содержит проект, то настройки и файл останутся прежними.

В автономном режиме программатор не поддерживает установку начального адреса данных в исходном файле (смещение), отдельные файлы для EEPROM данных и не может считать параметры конфигурационного слова из файла. Для корректной работы с такими микросхемами рекомендуется загружать данные в виде проекта, при этом программа на компьютере сама вносит все необходимые изменения.

Работа с микросхемой

Все режимы работы с микросхемой, кроме режима чтения, начинают работать сразу, в момент нажатия клавиши, без дополнительных запросов подтверждения. По окончании режима выводится сообщение о результатах работы и, если все прошло без ошибок, то через три секунды программатор сам переходит в режим ожидания. В режиме программирования дополнительно показывается количество успешно запрограммированных микросхем. Если в процессе работы были ошибки, то для перевода программатора в режим ожидания необходимо нажать на любую клавишу.

```
P12C508A
Bl. Pg. Vr. Menu
```

В режиме ожидания первые три клавиши (слева на право) 'Bl.', 'Pg.' и 'Vr.' запускают один из трех основных режимов работы с микросхемой:

- проверку на чистоту или на возможность записи;
- полный цикл записи микросхемы;
- сверку содержимого микросхемы с выбранным файлом (или с байтом \$FF, если

файл не выбран).

```
File: MyProgram
Rd. Er. Sm. Menu
```

Нажатие этих же клавиш при нажатой кнопке 'Menu' запускает дополнительные три режима работы:

- чтение микросхемы в память программатора;
- стирание микросхемы и проверку на чистоту;
- подсчет контрольной суммы микросхемы.

```
P12C508A
Read chip? N/Y
```

Чтобы считать микросхему, необходимо нажать кнопки 'Menu'+ 'Rd'. Программатор запросит подтверждение на чтение файла. Первые три кнопки - отказ, четвертая кнопка - запуск режима чтения. Если объем микросхемы превышает объем свободного места в памяти программатора, чтение не начнется, и будет выведено сообщение о нехватке места.

```
P12C508A
000000 0FFF 0C12
```

При возникновении ошибки во время работы с микросхемой, на экран выводится адрес ошибки, данные, считанные из микросхемы и соответствующие данные из файла. Нажатие первой (левой) кнопки прекращает работу и переводит программатор в режим ожидания. Во всех режимах, кроме программирования, вторая и третья кнопки пропускают все ошибки, а кнопка 'Menu' продолжает работу

программатора, до следующей ошибки. В режиме записи - средние две кнопки пропускают ошибку, а кнопка 'Menu' повторяет запись ошибочного адреса.

Меню программатора

Программное обеспечение на компьютере позволяет пользователю редактировать [список файлов и проектов](#), сохраненных в памяти программатора и формировать [списки микросхем](#), исключая заведомо не нужные позиции.

```
Select:
Chp Fil Prj Ut1
```

В автономном режиме меню программатора позволяет выбрать:

- шесть списков микросхем (до 1018 записей);
- список файлов (до 256 записей);
- список проектов (до 256 записей);

и утилиты (при нажатой кнопке 'Ut1'):

- разрешение или запрет на установку защиты;
- удаление файла или проекта;
- создание оперативной конфигурации.

```
Select:
Lck Del Sav
```

Программатор позволяет считать не защищенную микросхему и произвести последующую запись с защитой (например, AVR, PIC или MCS-51). Для этого после выбора микросхемы нужно разрешить доступ к защите (кнопки 'Menu', 'Ut1'+ 'Lck'). Затем считать микросхему и повторно разрешить доступ к защите (кнопки 'Menu', 'Ut1'+ 'Lck').

При удалении файла (кнопки 'Menu', 'Ut1'+ 'Del') происходит стирание секторов на диске, а в списке файлов сам файл помечается как удаленный (стерт первый символ в имени). Такой файл показывается в списке файлов, но его нельзя выбрать. При записи на диск нового файла он будет записан на место первого удаленного файла. Когда команда на удаление файла приходит с компьютера удаляется не только сам файл, но и обновляется список файлов в программаторе.

Текущие настройки можно сохранить как оперативную конфигурацию (кнопки 'Menu', 'Ut1'+ 'Sav'), и загружать ее при включении программатора, удерживая кнопку 'Bl.' в момент, когда на экран выводится информация о версии программатора.

Лицензионное соглашение

Настоящее лицензионное соглашение пользователя ("Соглашение") является юридическим соглашением между Вами и обществом с ограниченной ответственностью "Технический Центр ТРИТОН" ("ТЦ ТРИТОН"), разработчиком программного обеспечения "TRISOFT" ("программное обеспечение"). Устанавливая программное обеспечение "TRISOFT" на компьютер, Вы соглашаетесь руководствоваться положениями данного Соглашения. Если Вы не согласны с условиями настоящего Соглашения, "ТЦ ТРИТОН" отказывается от предоставления Вам лицензии на пользование своим программным обеспечением. В таком случае вы не имеете права устанавливать, копировать или иным способом использовать данное программное обеспечение.

1. Авторские права

Все права собственности и авторские права на программное обеспечение, а также любые копии программного обеспечения являются собственностью фирмы "ТЦ ТРИТОН". Программное обеспечение защищено законами об авторских правах, международными соглашениями об авторских правах, а также другими законами и соглашениями об интеллектуальной собственности. Данное программное обеспечение не продается, а предоставляется в пользование по лицензии.

2. Предоставление лицензии

"ТЦ ТРИТОН" предоставляет Вам лицензию с ограниченными правами пользования продуктом, без права продажи его третьим лицам или сдачу в аренду. Согласно этой лицензии Вы можете использовать программное обеспечение отдельно или совместно с программаторами "ТРИТОН" и "ТРИТОН+" пятой версии, к которым оно прилагается.

3. Описание требований, прав и ограничений

- Запрещается продавать или сдавать в аренду третьим лицам данное программное обеспечение.
- Запрещается наносить ущерб или нарушать права собственности, авторские права "ТЦ ТРИТОН" и/или третьих лиц посредством использования данного программного обеспечения.
- Запрещается вскрывать технологию, модифицировать, декомпилировать и дизассемблировать программное обеспечение в целом или его составные части.
- Все материалы, доступные посредством программного обеспечения, являются собственностью соответствующих владельцев этих материалов, и если конкретно не указано противное, настоящее Соглашение не дает Вам прав на использование этих материалов.
- Запрещается копировать или воспроизводить любые компоненты программного обеспечения.
- Не разрешается уступать все свои права или их часть по настоящему соглашению третьим лицам без их согласия со всеми пунктами данной лицензии.

4. Отказ от предоставления гарантий на программное обеспечение

Вы определенно соглашаетесь и признаете, что будете пользоваться программным обеспечением на свой страх и риск. Программное обеспечение "TRISOFT" предоставляется на условиях "КАК ЕСТЬ" и БЕЗ КАКИХ-ЛИБО ГАРАНТИЙ. Фирма "ТЦ ТРИТОН" определенно отказывается от предоставления любых гарантий, явно выраженных или подразумеваемых, включая (но не ограничиваясь только ими): ПОДРАЗУМЕВАЕМЫЕ ГАРАНТИИ НЕНАРУШЕНИЯ, ПРИГОДНОСТИ ДЛЯ ПРОДАЖИ И ПРИМЕНИМОСТИ ДЛЯ ОПРЕДЕЛЕННОЙ ЦЕЛИ, ВОЗМОЖНОСТИ ИСПОЛЬЗОВАНИЯ ВСЕХ ФУНКЦИЙ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ, И ДОСТУПНОСТИ ВСЕХ ПРОДУКТОВ И УСЛУГ, НЕОБХОДИМЫХ ДЛЯ ИСПОЛЬЗОВАНИЯ ВСЕХ ФУНКЦИЙ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ. "ТЦ ТРИТОН" НЕ ГАРАНТИРУЕТ, ЧТО ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ БУДЕТ ВЫПОЛНЯТЬ СВОИ ФУНКЦИИ НА ОЖИДАЕМОМ ВАМИ УРОВНЕ, А ТАКЖЕ НЕ ГАРАНТИРУЕТ, ЧТО РАБОТА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ БУДЕТ БЕСПЕРЕБОЙНОЙ И БЕЗОШИБОЧНОЙ И, ЧТО ДЕФЕКТЫ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ БУДУТ ИСПРАВЛЕНЫ. ФИРМА "ТЦ ТРИТОН" НЕ ПРЕДОСТАВЛЯЕТ НИКАКИХ ГАРАНТИЙ ОТНОСИТЕЛЬНО ТОЧНОСТИ, ПРАВИЛЬНОСТИ ИЛИ НАДЕЖНОСТИ ИСПОЛЬЗОВАНИЯ И РЕЗУЛЬТАТОВ, ПОЛУЧЕННЫХ ПОСРЕДСТВОМ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ. Устная или письменная информация или консультация, полученная от "ТЦ ТРИТОН" или уполномоченного лица, не является гарантией.

5. Ограниченная ответственность

ФИРМА "ТЦ ТРИТОН" НЕ НЕСЕТ ОТВЕТСТВЕННОСТИ НИ ЗА КАКОЙ ПРЯМОЙ, НЕПРЯМОЙ, СЛУЧАЙНЫЙ, ИЛИ ЛЮБОЙ ДРУГОЙ УЩЕРБ ИЛИ ПОЛОМКУ, А ТАКЖЕ ЗА НАРУШЕНИЕ ЛЮБЫХ ВЫРАЖЕННЫХ ИЛИ ПОДРАЗУМЕВАЕМЫХ ГАРАНТИЙ, НАРУШЕНИЕ КОНТРАКТА, УБЫТКИ, ВЫЗВАННЫЕ НЕБРЕЖНОСТЬЮ, И ЗА НАРУШЕНИЕ ЛЮБЫХ ИНЫХ ЮРИДИЧЕСКИХ ПОЛОЖЕНИЙ, ИМЕЮЩИХ ОТНОШЕНИЕ К ПРОГРАММНОМУ ОБЕСПЕЧЕНИЮ. Такой ущерб включает (но не ограничивается перечисленным) УТРАТУ ПРИБЫЛИ, УТРАТУ ДОХОДА, УТРАТУ ИНФОРМАЦИИ, ПРОСТОЙ ОБОРУДОВАНИЯ И ПОТЕРЯННОЕ ВРЕМЯ ПОЛЬЗОВАТЕЛЯ, даже если фирма "ТЦ ТРИТОН" была заранее уведомлена о возможности такого ущерба.

6. Ущерб, являющийся результатом ваших действий

Обязательства, требования, поломки, издержки, и денежные убытки, возникшие в результате Вашего пользования программным обеспечением или в результате нарушения Вами положений настоящего Соглашения, являются Вашей ответственностью. Вы обязуетесь не привлекать фирму "ТЦ ТРИТОН" к ответственности за вышеуказанное.

7. Действующее законодательство

НАСТОЯЩИМ ВЫ ОТКАЗЫВАЕТЕСЬ ОТ ВСЕХ ПРАВ НА СУДЕБНЫЕ РАЗБИРАТЕЛЬСТВА ПО ПОВОДУ ЛЮБОЙ ТЯЖБЫ, КОТОРАЯ БЫ ВОЗНИКЛА ИЗ КАКИХ-ЛИБО ПОЛОЖЕНИЙ НАСТОЯЩЕГО СОГЛАШЕНИЯ, ИЛИ ЖЕ ИМЕЛА К НЕМУ ОТНОШЕНИЕ. Вы согласны соблюдать все местные, национальные и международные законы и положения, которые относятся к программному обеспечению. Вдобавок к положениям, изложенным в настоящем Соглашении, "ТЦ ТРИТОН" может предпринять необходимые меры для защиты своих авторских прав на программное обеспечение.

8. Расторжение соглашения

Вы можете расторгнуть настоящее Соглашение в любое время, прекратив пользоваться программным обеспечением. В случае расторжения настоящего Соглашения, Вы должны удалить со своего компьютера программное обеспечение и уничтожить все копии программного обеспечения. Положения, описанные в настоящем Соглашении остаются действительными и после расторжения настоящего Соглашения.

9. Положение о частях соглашения

Если какая-либо из частей настоящего Соглашения признается недействительной или неосуществимой, остальные части продолжают быть действительными в соответствии с действующим законодательством.

10. Общие положения

Настоящее Соглашение является полным соглашением между сторонами об использовании программного обеспечения "TRISOFT" и заменяет все предыдущие или настоящие отношения между сторонами касательно такого использования. Настоящее Соглашение будет являться действительным для санкционированных приемников и правопреемников участвующих сторон. Отказ от прав, нарушение прав, или несоблюдение каких-либо положений настоящего Соглашения любой из участвующих сторон не исключает ее ответственности и не ограничивает таковой, а также не имеет никакого влияния на обязанности этой стороны впоследствии в строгости соблюдать все другие положения. Никакие изменения, внесенные в это Соглашение, не являются действительными, если не подтверждены подписями обеих сторон.

Гарантийные обязательства

ООО "Технический Центр ТРИТОН" (Изготовитель) обеспечивает все программаторы гарантийным обслуживанием в течение **трех лет**, с момента покупки. Данная гарантия распространяется только на сам программатор и его составные части, исключая панельку. На дополнительное оборудование, поставляемое в комплекте: блок питания, интерфейсные кабели и носители информации, распространяется гарантия сроком на три месяца. Нормативный срок службы программатора составляет 5 лет.

1. В течение гарантийного срока Изготовитель гарантирует отсутствие в программаторе производственных дефектов и обязуется выполнить гарантийный ремонт изделия, если такие дефекты обнаружатся в процессе нормальной эксплуатации.

2. Изделие и программное обеспечение поставляются **как есть**, на условиях [Лицензионного соглашения](#). Изготовитель оставляет за собой право вносить в изделие или программное обеспечение любые изменения без уведомления покупателя. Если изделие устарело или больше не выпускается, замена его на новое не производится.

3. Гарантийные обязательства не распространяются на:

- периодическое обслуживание, ремонт или замену составных частей или аксессуаров;
- применение неоригинальных панелек, адаптеров или других аксессуаров;
- любые механические повреждения, в том числе износ панельки программатора;
- дефекты, возникшие в результате любой транспортировки изделия;
- дефекты, возникшие в результате любой модификации сделанной покупателем;
- дефекты, возникшие в результате неправильного или неаккуратного использования, а также в случае форс-мажорных обстоятельств (включая, но не ограничиваясь этим) несчастные случаи, стихийные бедствия, пожары, аварии, попадания насекомых, инородных жидкостей, химических или других веществ, облучения, затопления, вибрации, неправильной вентиляции, воздействия высокой или низкой температуры, колебаний напряжения, электростатических разрядов, включая неправильное заземление, и иные виды внешнего воздействия или влияния.

4. Изготовитель не несет ответственность за любой ущерб или убытки, связанные с использованием программатора или запрограммированных на нем микросхем, включая экономические и нематериальные потери, утрату прибыли, утрату дохода, потерю данных, простой оборудования и потерянное время пользователя, а также прямые, косвенные, случайные или вытекающие как следствие потери или убытки.

Перед обращением в службу поддержки внимательно изучите данное РУКОВОДСТВО:

- Найдите [описание ошибки](#) и точно выполните указанные там рекомендации.
- В разделе "[Особенности работы с микросхемами](#)" прочитайте как работать с данной микросхемой.
- Проверьте работу с самой последней [версией программного обеспечения](#).
- Если используется переходная панелька, проверьте ее название и исправность выводов панельки.
- При использовании чужих панелек и адаптеров дополнительно проверьте правильность разводки.

Техподдержка **НЕ ОТВЕЧАЕТ** на вопросы, которые подробно описаны в данном Руководстве.

При обращении в службу поддержки: triton@triton-prog.ru

- Укажите модель программатора, версию программы и название панельки.
- Приложите к сообщению полный лог-файл работы с микросхемой.
- Не надо присылать документацию на микросхемы, файлы прошивок и перехваты экранов.

Для упрощения оформления запроса в службу технической поддержки, в оболочку программатора встроен генератор отчетов, записывающий в текстовый файл текущие настройки программы, режимы работы и сообщения об ошибках. Специалист в техподдержке, с помощью этого файла, симулирует данную ситуацию, проверит работу программатора с заведомо исправной микросхемой и предложит конкретные действия по решению проблемы.

Чтобы предоставить более подробную информацию, несколько раз повторите операцию, на которой возникли ошибки. Посмотрите какие ошибки, выдает программа, по каким адресам. Для этого в окне сообщения об ошибке несколько раз нажмите кнопку "Пропустить". Если проблема связана с чтением микросхемы, то несколько раз проведите контрольную сверку и подсчитайте контрольную сумму. Если ошибка при стирании микросхемы, то проверьте микросхему на чистоту. При ошибках записи - проведите несколько контрольных сверок при разных напряжениях.

После чего отправьте полученный файл в службу технической поддержки. Файлы отчетов находятся в папке "Log", которую можно найти в директории, куда инсталлирована оболочка программатора. Не надо комментировать свои действия - они уже расписаны в лог-файле. Лучше опишите, с какой микросхемой работаете (новая, паянная или б/у), какая панелька используется, ее название, кто производитель или где покупалась. При внутрисхемном программировании обязательно укажите, как подключается микросхема, что подключено на плате к выводам, которые используются для программирования.

Помните, чем полнее и подробнее будет присланная информация, тем точнее и качественней будет ответ.

[ООО "ТЦ ТРИТОН"](#)