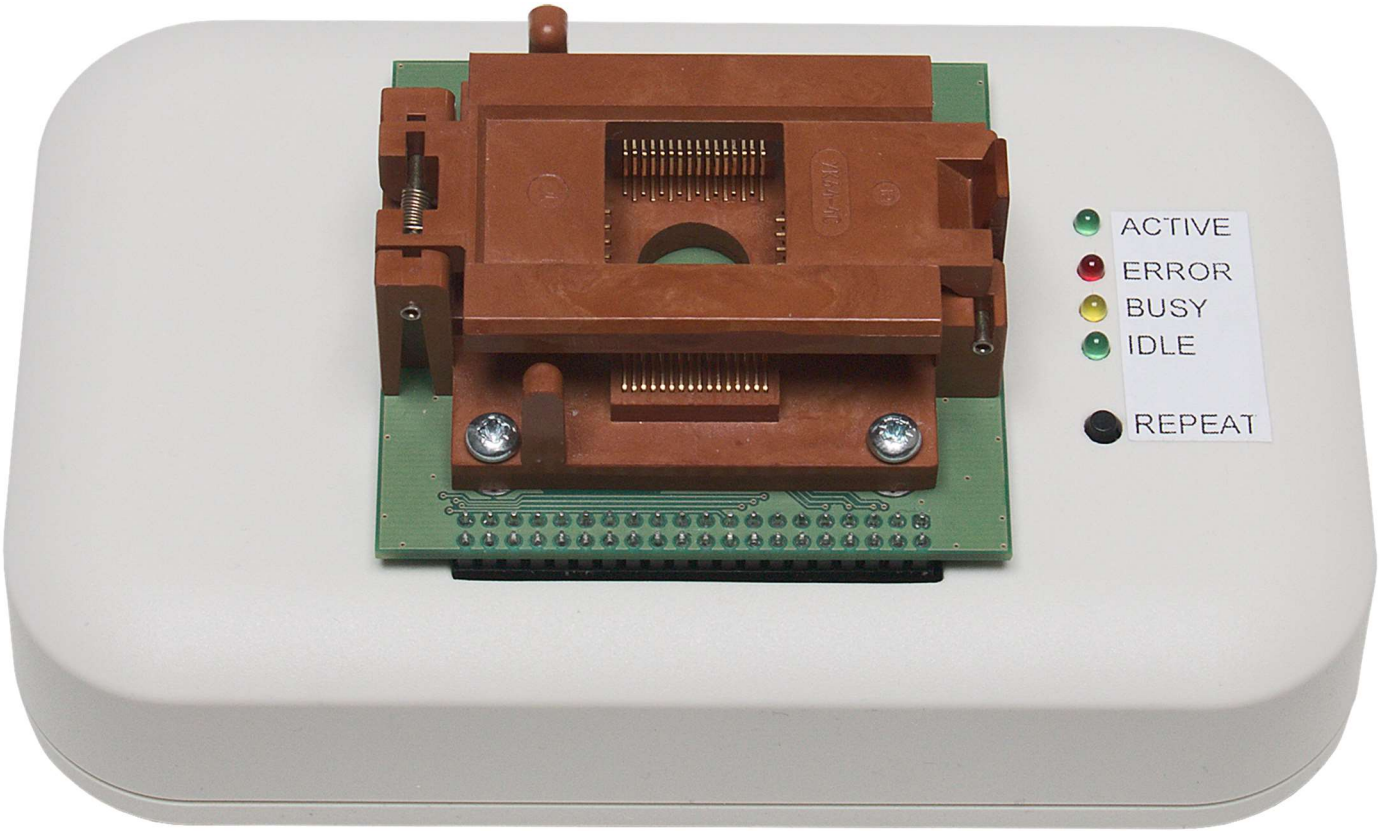
Документация для устройства

Программатор микросхем 1667РТ015/1667РТ025



Версия документации: 0.2

Оглавление

1 Основные сведения об изделии 3
2 Технические характеристики 3
3 Внешний вид устройств и назначение выводов 4
3.1 Комплектация 5
3.2 Минимальные требования к ПК 6
4 Работа с программатором 7
4.1 Пуск 8
4.2 Обновление прошивки программатора 8
4.3 Проведение входного контроля микросхемы 8
4.4 Запись данных зашивки в микросхему 9
4.5 Проведение выходного контроля микросхемы 10
4.6 Считывание тестовых строк 11
4.7 Считывание памяти микросхемы 12
4.8 Ошибки в работе программатора 12
4.9 Сохранение журнала событий (лога) программатора 13
4.10 Правила оформления данных программирования микросхем для сторонних заказчиков 14
Форматы представления информация в файле данных зашивки ПЗУ 15
Строковый шестнадцатеричный формат (she) 15
Строковый двоичный формат (sbi) 16
Текстовый шестнадцатеричный формат (the) 16
Текстовый двоичный формат (tbi) 16
Шестнадцатеричный Verilog формат (vhe) 16
Двоичный Verilog формат (vbi) 17
Формат Intel-HEX (ihe) 17
Бинарный файл (bin) 20
Особенности адресации при изменении разрядности ПЗУ 20
5 Правила и условия эксплуатации 22
6 Гарантии производителя 22
7 Свидетельство о приемке 23
8 Сведения о рекламациях 23
9 Сведения об утилизации 23
10 Лист регистрации изменений документации 24

1 Основные сведения об изделии

Программно-аппаратный комплекс предназначен для записи информации в однократно программируемые постоянные запоминающие устройства (ОППЗУ) **1667РТ015/1667РТ025** информационной ёмкостью 1/4 Мбит для использования в устройствах вычислительной техники в качестве ПЗУ с произвольным доступом и для конфигурирования программируемых логических интегральных схем (ПЛИС) как ПЗУ с последовательной выборкой. Существует возможность обновления прошивки программатора, поддерживается 8 форматов данных входных файлов и формирования разрядов Хэмминга для обнаружения и исправления одиночных ошибок в записываемых данных.

2 Технические характеристики

Интерфейс подключения программатора к ПК	USB-B
Диапазон напряжений питания программатора	12 - 18 В
Максимальный потребляемый ток программатора	500 мА
Максимальная ёмкость программируемой микросхемы	4 Мбит
Диапазон напряжения питания программируемой микросхемы	3.0 - 5.5 В
Максимальный (динамический) ток потребления программируемой микросхемы	40 мА
Максимальная частота тактирования входного/выходного сигнала программируемой микросхемы	10 МГц
Тип корпуса программируемой микросхемы	5134.64-6
Диапазон рабочих температур программатора	+10...+40 °С
Допустимый уровень влажности при эксплуатации программатора	< 80%
Масса	330 гр.
Габариты (длина х ширина х высота), мм	150 х 93 х 35

3 Внешний вид устройств и назначение выводов

Рис. 1 : Программатор, вид сверху

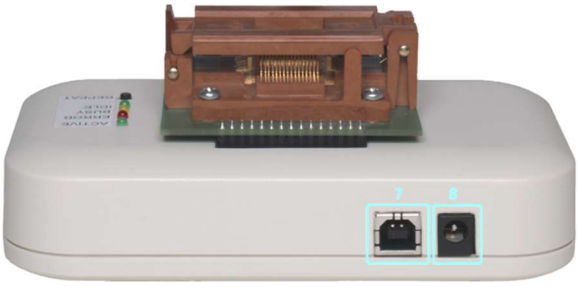


Рис. 2: Программатор с контактирующим устройством, вид сзади

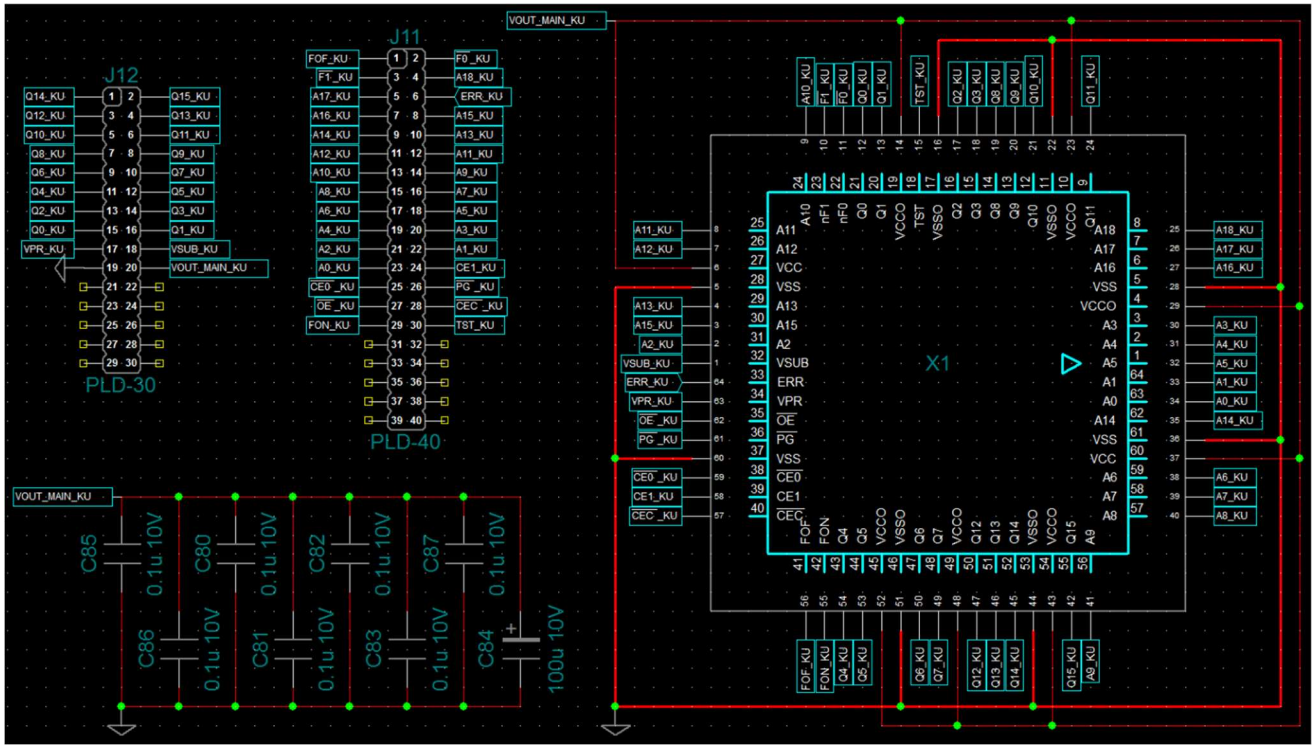


Рис. 3: Электрическая схема принципиальная контактирующего устройства

Рис. 4: Блок питания и USB кабель для программатора

Условные обозначения на рис. 1, 2:

1 - разъёмы для контактирующего устройства
2, 3, 4, 5 - светодиодные индикаторы
6 - кнопка повтора операции
7 - разъём USB-B
8 - разъём для блока питания

Назначение элементов управления:

- зелёный светодиод (рис. 1, поз. 2) предназначен для индикации работы устройства (в активном состоянии - постоянное мигание).
- красный светодиод (рис. 1, поз. 3) предназначен для индикации наличия ошибок возникших при работе программатора.
- жёлтый светодиод (рис. 1, поз. 4) предназначен для индикации наличия активных процессов в работе программатора (запись, сверка, тестирование).
- зелёный светодиод (рис. 1, поз. 5) предназначен для индикации состояния ожидания/отстуствия ошибок.
- кнопка (рис. 1, поз. 6) предназначена для быстрого запуска повторной процедуры записи и сверки данных на следующей микросхеме. Используемые для записи данные остаются от предыдущей процедуры, поэтому обязательно необходим первый ручной пуск процедуры записи данных на микросхему (см. **4.3 Запись данных в микросхему**).

3.1 Комплектация

- Программатор микросхем
- Контактирующее устройство микросхем 1667РТ015/1667РТ025
- Блок питания 15 В, 800 мА
- Кабель USB-B
- Файл программы и сопутствующие файлы на DVD накопителе для работы с программатором **Programmer.exe** (конфигурация программы portable, т.е. не требует установки)
- Файл прошивки программатора на DVD накопителе **firmware.bin**

3.2 Минимальные требования к ПК

- Операционная система Windows 10 x64
- Процессор с частотой не менее 1 ГГц
- 2 ГБ ОЗУ
- 100 МБ свободной памяти на жёстком диске

4 Работа с программатором

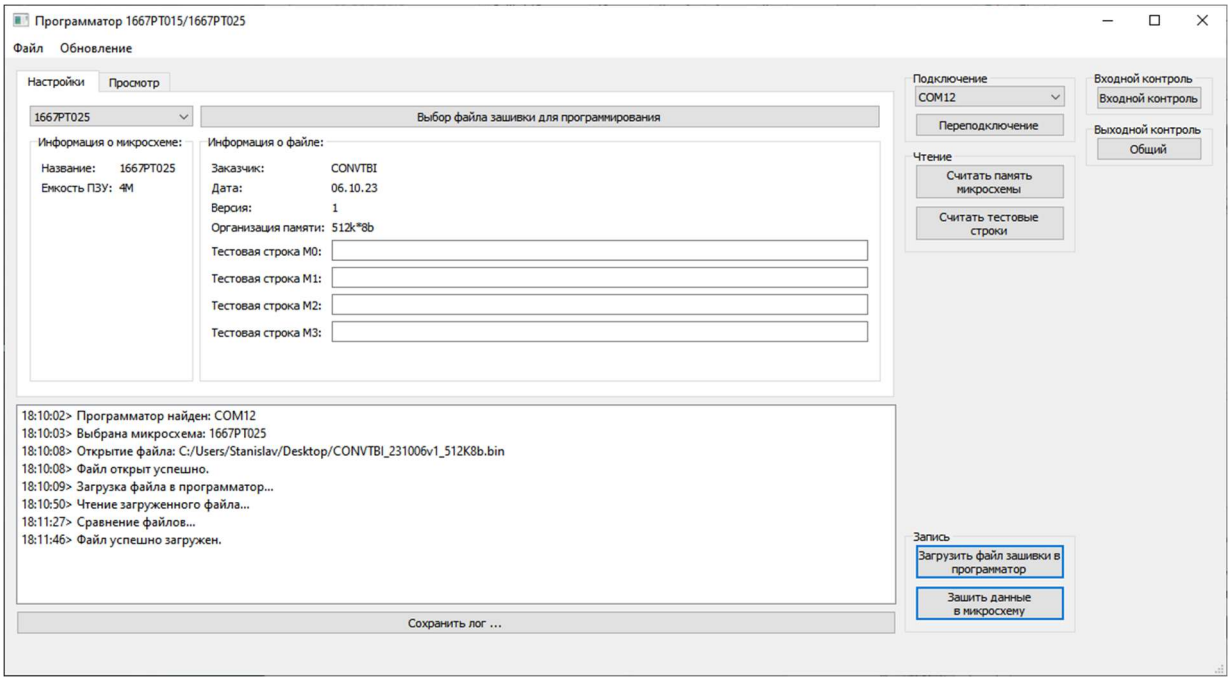


Рис. 5: Внешний вид окна программы **Программатор 1667РТ015/1667РТ025** во вкладке **«Настройки»**

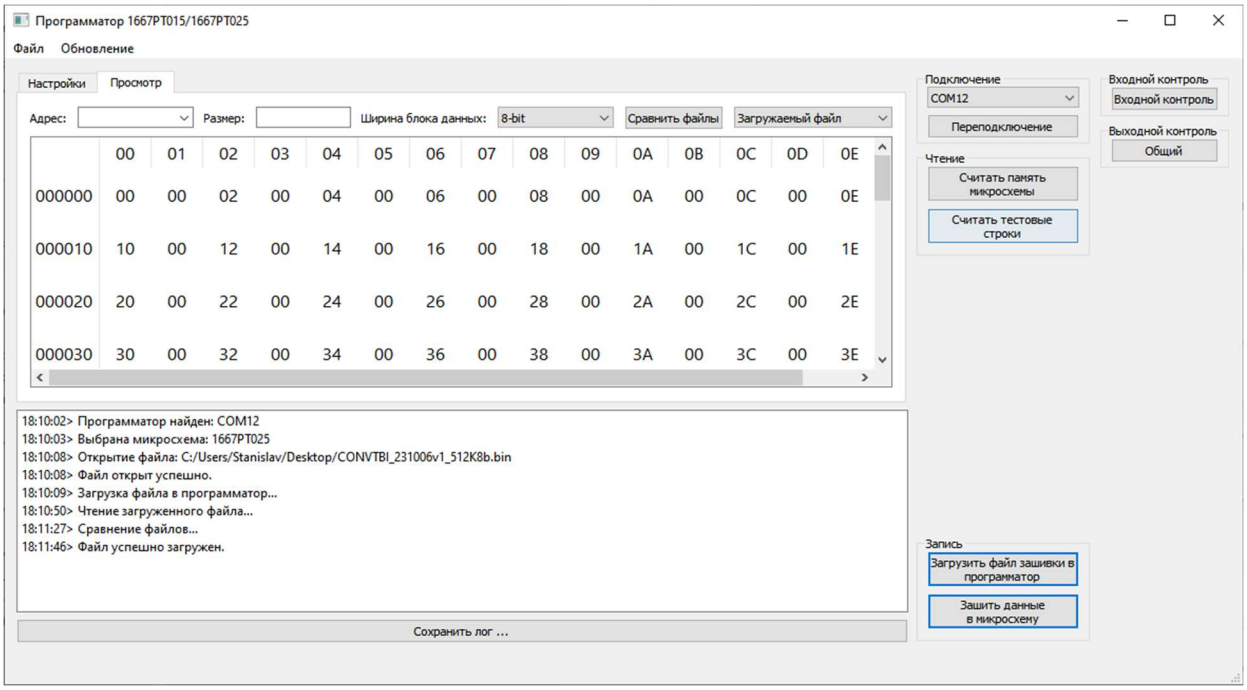


Рис. 6: Внешний вид окна программы **Программатор 1667РТ015/1667РТ025** во вкладке **«Просмотр»**

4.1 Пуск

1. Установите контактирующее устройство в разъём (рис. 1, поз. 1).
2. Подключите блок питания к программатору.
3. После подачи питания в течение ~3 сек. будет мигать зелёный светодиод (рис. 1, поз. 2) и далее устройство перейдёт в состояние ожидания (включится зелёный светодиод (рис. 1, поз. 5)).
4. Подключите кабель USB-B к программатору и ПК.
5. Запустите на ПК программу **Programmer.exe**.
6. Программатор должен автоматически определиться в операционной системе как USB устройство с виртуальным COM портом. Программа должна автоматически определить наличие программатора в системе. На случай ошибки определения (см. **4.6 Ошибки в работе программатора**) можно произвести ручное подключение, для этого в левой панели программы (рис. 5) нажмите на кнопку **«COMX»** (где **X**- число) с выпадающим списком и выберите нужный порт, соответствующий подключённому программатору. Нажмите кнопку «**Переподключение»** после чего в нижнем окне журнала событий появится запись следующего содержания:

HH.MM.SS> Программатор найден: COMX

где HH - часы, MM - минуты, SS – секунды.

4.2 Обновление прошивки программатора

Программатор поставляется с уже загруженной в него прошивкой. Для обновления / ручной прошивки выполните следующие инструкции:

1. Нажмите на кнопку **«Обновление»** в окне программы (рис. 5).
2. В выпадающем списке нажмите **«Обновить программатор»**.
3. Далее откроется окно файлового проводника, в котором нужно выбрать нужный файл прошивки формата *.bin.
4. В нижней части программы появится полоса состояния процесса обновления, дождитесь полного завершения процесса.
5. По завершении обновления в нижнем окне журнала событий появится запись следующего содержания:

HH.MM.SS> Программатор успешно обновлён

6. **Перезапустите** программу на ПК **Programmer.exe**.

4.3 Проведение входного контроля микросхемы

Перед тем, как начать процедуру записи зашивки в микросхему, настоятельно **рекомендуется провести входной контроль микросхемы**, нажав на соответствующую кнопку в интерфейсе программы (рисунок 7). Входной контроль проверяет подключение к контактирующему устройству микросхемы, а также определяет, записана ли в микросхему зашивка. Если микросхема не подключена к контактирующему устройству, или в ней уже записана зашивка, входной контроль проинформирует во всплывающем окне об этом и выдаст ошибку (рисунок 8). Входной контроль повторно проводится на этапе программирования (пункт 4.4 документации) и не позволяет записать информацию в микросхему с уже записанной информацией повторно, что **защищает от выхода из строя микросхему**.

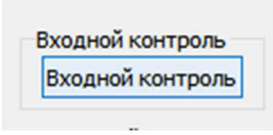


Рис. 7. Кнопка входного контроля в боковой части интерфейса программы

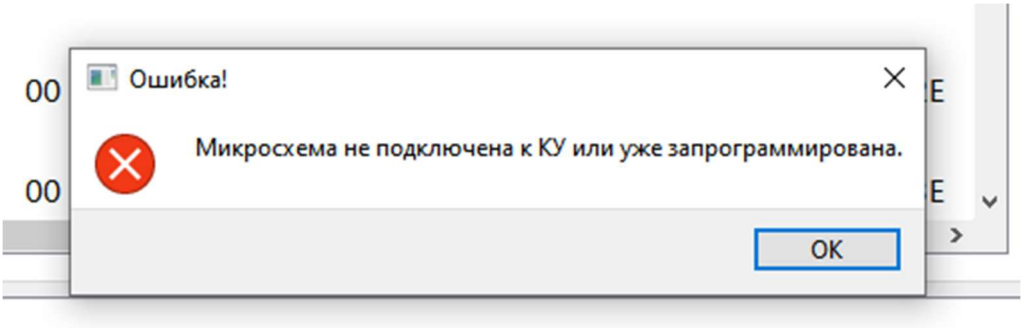


Рис. 8. Всплывающее окно об провале входного контроля микросхемы ОППЗУ

4.4 Запись данных зашивки в микросхему

Процедура записи зашивки (набор тестовых строк (M0, M1, M2, M3) со служебными данными + основные данные) предполагает предварительное перемещение данных файла зашивки из ПК в память программатора (RAM), и далее непосредственно запись хранящихся в программаторе данных в память микросхемы.

1. Установите программируемую микросхему в контактирующее устройство.
2. Находясь во вкладке **«Настройки»** программы (рис. 5), под этой надписью нажмите на кнопку **«Выбор микросхемы»** с выпадающим списком и выберите нужную микросхему.
3. В нижнем окне журнала событий появится запись следующего содержания:
   HH.MM.SS> Выбрана микросхема: 1667РТ0XX
4. Для записи содержимого файла зашивки микросхемы в память программатора нажмите на кнопку **«Выбор файла для программирования»**.
5. Далее откроется окно файлового проводника, в котором нужно выбрать соответствующий файл зашивки микросхемы формата *.she / *.sbi / *.the / *.tbi / *.vhe / *.vbi / *.ihe / *.bin. Требований к конкретному расположению этого файла в системе нет.
6. В разделе окна **«Информация о файле»** появится информация о выбранном файле зашивки (информация о заказчике, дате создания, версии, организации памяти микросхемы), в нижнем разделе окна программы (журнал событий) появятся записи следующего содержания:

   HH.MM.SS> Открытие файла: [имя файла]
   HH.MM.SS> Файл успешно открыт

7. Нажмите на кнопку **«Загрузить файл в программатор»**, которая станет активной.
8. В нижней части окна программы появится полоса состояния процесса загрузки и сверки, дождитесь полного завершения процесса. По окончании, в нижнем разделе окна программы (журнал событий) появятся записи следующего содержания:

   HH.MM.SS> Загрузка файла в программатор…
   HH.MM.SS> Чтение загруженного файла…
   HH.MM.SS> Сравнение файлов…
   HH.MM.SS> Файл успешно загружен.

9. Нажмите кнопку **«Загрузить данные в микросхему»**, которая станет активной после загрузки в RAM-память программатора данных зашивки. По окончании процесса в нижнем разделе окна программы (журнал событий) появятся записи следующего содержания:

HH.MM.SS> Установка напряжения питания…
HH.MM.SS> Выполняется входной контроль…
HH.MM.SS> Установка напряжения VPR...
HH.MM.SS> Установка напряжения VSUB...
HH.MM.SS> Выполняется программирование тестовых строк...
HH.MM.SS> Выполняется резервирование строк...
HH.MM.SS> Выполняется выходной контроль. Тест 4.1...
HH.MM.SS> Выполняется программирование основной матрицы...
HH.MM.SS> Выполняется копирование памяти микросхемы в RAM...
HH.MM.SS> Микросхема запрограммирована успешно

4.5 Проведение выходного контроля микросхемы

Проведение выходного контроля **позволяет определить наличие в микросхеме двойных ошибок** после записи (данные ошибки не исправляются самокорректирующим кодом Хемминга, поэтому считаются критическими и нарушающими целостность записанной информации) производится автоматически в процессе записи данных зашивки в микросхему. По итогам выходного контроля можно определить: корректно ли были записаны данные зашивки в микросхему. Если присутствуют двойные ошибки в микросхеме, программа проинформирует об этом (рисунок 10). Возможно проведение отдельно от записи выходного контроля микросхемы в ручном режиме по общей программе, аналогичной во время записи зашивки (кнопка «Общий» в разделе «Выходной контроль», рисунок 9, 4 выделение). Для этого необходимо сделать следующее:

1. Выбрать необходимую микросхему (1667РТ015 или 1667РТ025) в интерфейсе программы (рисунок 10, 1 выделение)
2. Загрузить файл зашивки, нажав на кнопку ***«Выбор файла зашивки для программирования»*** *(рисунок 10, 2 выделение)*
3. Нажать на кнопку ***«Загрузить файл зашивки в программатор»*** *(рисунок 10, 3 выделение)*
4. После выполнения пунктов 1-3, кнопка выходного контроля **«Общий»** станет активна. Для проведения выходного контроля по общей программе, необходимо на нее нажать.

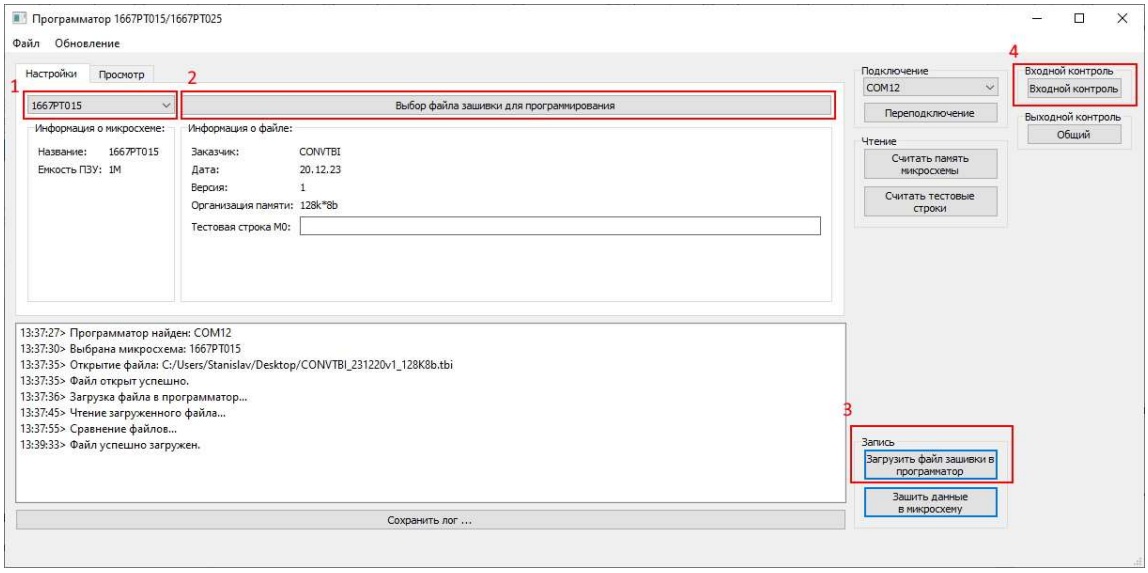


Рис. 9. Иллюстрация последовательности действий проведения выходного контроля

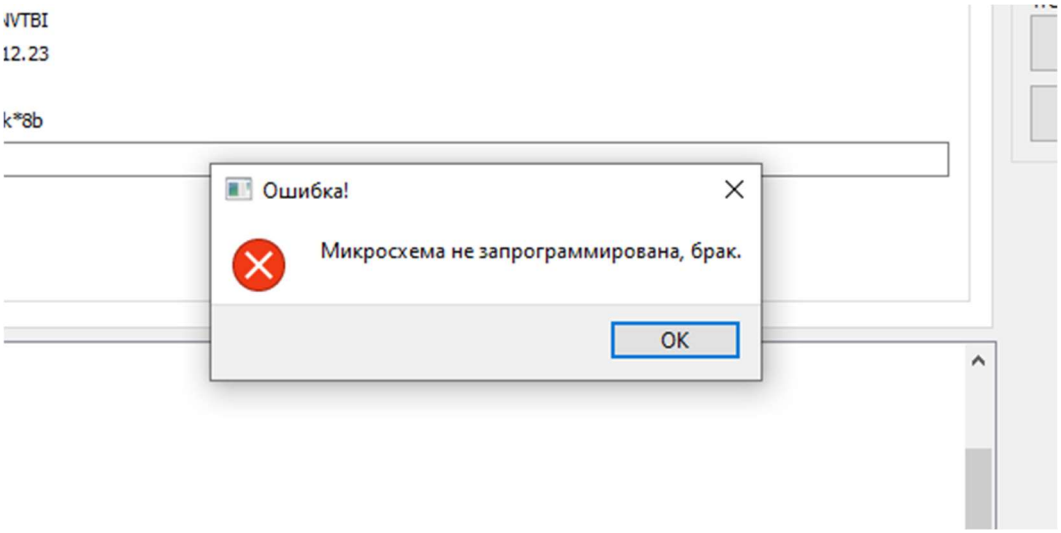


Рис. 10. Информирование о провале выходного контроля

4.6 Считывание тестовых строк

В программаторе реализована возможность считывания содержимого тестовых строк микросхемы, которые могут содержать служебную информацию (дату программирования, номер зашивки, наличие инвертирования данных и т.д.).

1. Выполнив 1, 2, и 3 пункты инструкции **4.3 Запись данных в микросхему** нажмите на кнопку **«Считать тестовые строки»** окна программы (рис. 5).
2. В нижнем разделе окна программы (журнал событий) появятся записи следующего содержания:

HH.MM.SS> Установка напряжения питания...
HH.MM.SS> Установка напряжения VPR...
HH.MM.SS> Считывание тестовых строк с микросхемы…
HH.MM.SS> Тестовая строка M0:
HH.MM.SS> Тестовая строка M1:
HH.MM.SS> Тестовая строка M2:

HH.MM.SS> Тестовая строка М3:

Содержимое тестовых строк отобразится в соответствующих полях раздела окна программы **«Информация о файле»** во вкладке **«Настройки»**.

4.7 Считывание памяти микросхемы

В программаторе реализована возможность считывания содержимого памяти микросхемы для сверки основных записываемых данных и реального содержимого памяти микросхемы, а также для изучения содержимого памяти сторонней микросхемы.

1. Выполнив 1, 2, и 3 пункты инструкции **«4.3 Запись данных в микросхемы»**, в главном окне программы (рис. 5) нажмите на кнопку **«Считать память микросхемы»**.
2. По окончании процесса в нижнем разделе окна программы (журнал событий) появятся записи следующего содержания:

HH.MM.SS> Выполняется копирование памяти микросхемы в RAM...
HH.MM.SS> Считывание памяти RAM...
HH.MM.SS> Файл считан успешно.

3. После процедуры считывания можно просмотреть полученное содержимое памяти микросхемы перейдя во вкладку **«Просмотр»** программы (рис. 6). В ней можно настроить разрядность представляемых данных (длинна слова 8, 16, 32 бит), и также выбрать содержимое для просмотра (нажмите кнопку **«Память микросхемы»** / **«Загружаемый файл»** и выбираете нужный вариант). Также можно сравнить содержимое памяти микросхемы и записываемого файла (нажмите кнопку **«Сравнить файлы»**).

4.8 Ошибки в работе программатора

При возникновении следующих ошибок, возникающих в процессе работы (включается красный светодиод (рис. 1, поз. 3), появляются записи в окне журнала событий или в отдельных открывающихся предупредительных окнах):

- Ошибка чтения состояния программатора!
- Ошибка обновления программатора!
- Ошибка при программировании.
- Ошибка при выходном контроле. Резервирование невозможно.
- Ошибка при резервировании.
- Ошибка записи тестовых строк.
- Ошибка чтения.
- Системная ошибка.
- Микросхема не подключена к КУ или уже запрограммирована.

произведите следующие действия:

- проверьте надёжность подключения блока питания и кабеля USB
- проверьте качество контакта между контактным устройством и программатором
- проверьте качество контакта между контактным устройством и микросхемой
- проверьте правильность установки микросхемы

- при подозрении на неправильную работу внутренней программы устройства попробуйте повторно обновить прошивку программатора (см. **4.2 Обновление прошивки программатора**)
- удостоверьтесь, что зашиваемая микросхема предварительно не была зашита

4.9 Сохранение журнала событий (лога) программатора

Основные вехи и этапы работы программатора записываются в журнал событий в нижней части интерфейса программы. При желании, лог работы программатора можно сохранить в отдельный файл на ПК. Для этого необходимо нажать на кнопку **«Сохранить лог…»**, откроется отдельное окно с выбором директории для сохранения файла и в котором необходимо вписать имя сохраняемого файла.

4.10 Правила оформления данных программирования микросхем для сторонних заказчиков.

Информация по кодировке данных ПЗУ предоставляется заказчиком как отдельный файл в электронном виде на машиночитаемом носителе. Файл данных зашивки является приложением к карте заказа, не должен содержать кириллических символов и должен иметь имя в виде:

<Заказчик>_<Дата>v<Версия>_<Инф. организация ПЗУ>.<обозн-е формата>,

где **<Инф. организация ПЗУ>** записывается в виде:

<кол-во слов в тысячах>K<длина слова, бит>b

или

<кол-во слов в миллионах>M<длина слова, бит>b

Примеры имени файла данных зашивки:

```
mikron_150830v1_256K16b.vhe
niime_161104v2_1M8b.bin
```

Для записи информации в файл данных зашивки могут быть использованы 8 различных форматов записи, из которых 7 являются текстовыми и 1 является бинарным.

Таблица 1, форматы записи данных ПЗУ:

№	Описание	Обозначение
1	Шестнадцатеричный строковый формат	she
2	Двоичный строковый формат	sbi
3	Шестнадцатеричный текстовый формат	the
4	Двоичный текстовый формат	tbi
5	Шестнадцатеричный Verilog формат	vhe
6	Двоичный Verilog формат	vbi
7	Формат Intel-HEX	ihe
8	Данные в бинарном формате	bin

Каждая запись данных в файле данных зашивки представляет собой слово, длина которого соответствует разрядности зашивки ПЗУ (8, 16 или 32 бит). Для любого из используемых форматов, запись как данных, так и адреса (для форматов с указанием адреса) производится слева направо от старшего к младшему разряду.

Пример:

	Запись:	Порядок бит в записи:
запись данных (BIN)	01011010	Q7 Q6 Q5 Q4 Q3 Q2 Q1 Q0
запись данных (HEX)	3AC7	Q15 Q14 Q13 Q12 Q11 Q10 Q9 Q8 Q7 Q6 Q5 Q4 Q3 Q2 Q1 Q0
запись адреса (HEX)	5E	A7 A6 A5 A4 A3 A2 A1 A0

Все из используемых текстовых форматов являются регистронезависимыми, т.е. записи «A0B4» и «a0b4» считаются идентичными.

Также, все текстовые форматы могут быть записаны с использованием как **DOS-формата** переноса строк (каждая строка оканчивается парой непечатаемых символов **CR**+**LF**), так и **Unix-формата** переноса строк (каждая строка оканчивается только непечатаемым символом **LF**). Однако при этом в пределах одного файла должен использоваться только один из вариантов.

Ни один из используемых текстовых форматов не должен содержать пустых строк (следующих последовательно друг за другом символов переноса строки).

Для всех используемых форматов должна производиться полная запись данных и адреса (для форматов с указанием адреса), т.е. при записи всегда указываются старшие значимые нули. При записи данных обязательно использование всех адресов. Если поле данных для какого-либо адреса содержит «0», оно все равно должно быть записано в файл.

Пример:

Значение поля данных (HEX)	Запись в файле для HEX 16 бит	Запись в файле для BIN 8 бит
A	000A	00001010
10	0010	00010000
1	0001	00000001
0	0000	00000000

Таблица 2, диапазон адресов для ПЗУ:

Обозначение микросхемы и её организация памяти		Диапазон допустимых адресов	
		HEX	DEC
1662РЕ1Т	(128К×32) бит	0 ÷ 1FFFF	0 ÷ 131 071
	(256К×16) бит	0 ÷ 3FFFF	0 ÷ 262 143
	(512К×8) бит	0 ÷ 7FFFF	0 ÷ 524 287
1662РЕ2Т	(256К×32) бит	0 ÷ 3FFFF	0 ÷ 262 143
	(512К×16) бит	0 ÷ 7FFFF	0 ÷ 524 287
	(1М×8) бит	0 ÷ FFFFF	0 ÷ 1 048 575
1662РЕ7У	(256К×16) бит	0 ÷ 3FFFF	0 ÷ 262 143
	(512К×8) бит	0 ÷ 7FFFF	0 ÷ 524 287

Форматы представления информация в файле данных зашивки ПЗУ

Строковый шестнадцатеричный формат (she)

Шестнадцатеричное представление данных в текстовом виде, начиная с нулевого адреса без контрольных сумм. Каждая запись данных производится через пробел без переносов строк. Каждая пара 16-ричных цифр соответствует 1 байту слова данных. Для разрядностей 8, 16 и 32 бит в каждой записи должно быть соответственно 1, 2 и 4 пары 16-ричных цифр. Количество записей должно соответствовать числу адресов для данной зашивки.

Пример содержимого файла в формате **she** (разрядность 8 бит):

00 FFFFFFFFFFFF 00 34 AC 63 40 00 00 00 00 00 07 01 00 …

Строковый двоичный формат (sbi)

Двоичное представление данных в текстовом виде, начиная с нулевого адреса без контрольных сумм. Каждая запись данных производится через пробел без переносов строк. Каждая цифра соответствует 1 биту слова данных. Для разрядностей 8, 16 и 32 бит в каждой записи должно быть соответственно 8, 16 и 32 двоичных цифры. Количество записей должно соответствовать числу адресов для данной зашивки.

Пример содержимого файла в формате **sbi** (разрядность 8 бит):

00000000 00110110 10110100 11111111 01100110 11000000 00110100 …

Текстовый шестнадцатеричный формат (the)

Шестнадцатеричное представление данных в текстовом виде, начиная с нулевого адреса без контрольных сумм. Каждая запись данных представляет собой отдельную строку, оканчивающуюся символом/символами переноса строки. Каждая пара 16-ричных цифр соответствует 1 байту слова данных. Для разрядностей 8, 16 и 32 бит в каждой строке должно быть соответственно 1, 2 и 4 пары 16-ричных цифр. Число строк в файле должно соответствовать числу адресов для данной зашивки.

Пример содержимого файла в формате **the** (разрядность 16 бит):

00FF
FFFF
FF00
34AC
6340
0000

Текстовый двоичный формат (tbi)

Двоичное представление данных в текстовом виде, начиная с нулевого адреса без контрольных сумм. Каждая запись данных представляет собой отдельную строку, оканчивающуюся символом/символами переноса строки. Каждая цифра соответствует 1 биту слова данных. Для разрядностей 8, 16 и 32 бит в каждой строке должно быть соответственно 8, 16 и 32 двоичных цифры. Число строк в файле должно соответствовать числу адресов для данной зашивки.

Пример содержимого файла в формате **tbi** (разрядность 8 бит):

00000000
11111111
00001101
11100110
00000000
00001011

Шестнадцатеричный Verilog формат (vhe)

Шестнадцатеричное представление данных в текстовом виде с указанием адреса без контрольных сумм. Каждая запись представляет собой ASCII-строку, состоящую из полей адреса и данных, разделенных символом пробела. Строка должна начинаться с символа «@», затем без пробела должен быть записан адрес в шестнадцатеричном формате. После адреса через пробел записывается поле данных в шестнадцатеричном формате. Оканчиваться строка должна символом/символами переноса строки. Количество символов в поле адреса для всех строк должно быть одинаковым и не может быть меньше числа символов в максимальном для данной зашивки

адресе. В поле данных для разрядностей 8, 16 и 32 бит должно быть соответственно 1, 2 и 4 пары 16-ричных цифр. Число строк в файле должно соответствовать числу адресов для данной зашивки.

Пример содержимого файла в формате **vhe** (для информационной организации (256К×16) бит):

```
@00000 00FF
@00001 FFFF
@00F00 FF00
@00F01 34AC
@00F02 6340
@00F03 0000
@00F04 073A
```

Двоичный Verilog формат (vbi)

Двоичное представление данных в текстовом виде с указанием адреса без контрольных сумм. Каждая запись представляет собой ASCII-строку, состоящую из полей адреса и данных, разделенных символом пробела. Строка должна начинаться с символа «@», за ним без пробела должен быть записан адрес в шестнадцатеричном формате. После адреса через пробел записывается поле данных в двоичном формате. Оканчиваться строка должна символом/символами переноса строки. Количество символов в поле адреса для всех строк должно быть одинаковым и не может быть меньше числа символов в максимальном для данной зашивки адресе. В поле данных для разрядностей 8, 16 и 32 бит должно быть соответственно 8, 16 и 32 двоичных цифры. Число строк в файле должно соответствовать числу адресов для данной зашивки.

Пример содержимого файла в формате **vbi** (для информационной организации (512К×8) бит):

```
@00000 00000000
@00001 11111111
@00F00 00001101
@00F01 01101101
@00F02 11100110
@00F03 00000000
@00F04 00001011
```

Формат Intel-HEX (ihe)

Формат соответствует спецификации HEX8: «Hexadimal Object File Format Specification, Revision A January 6, 1988» с некоторыми ограничениями. Число строк в файле формата **ihe** всегда больше, чем количество адресов для выбранной зашивки ПЗУ. Каждая запись в файле представляет собой ASCII-строку, состоящую из нескольких пар 16-ричных цифр. Каждая строка должна начинаться с символа двоеточия и оканчиваться символом/символами переноса строки. Запись состоит из пяти полей следующего формата:

:LLAAAATTDD…CC

Каждая группа букв (LL, AAAA, TT и т.д.) представляет собой отдельное поле. Каждая буква – отдельную 16-ричную цифру (4 бита). Каждое поле состоит как минимум из пары 16-ричных цифр (1 байт). Ниже представлена расшифровка полей записи:

:	Указатель на начало записи. Каждая запись в файле должна начинаться с данного символа
LL	Поле длины – показывает количество байт данных (DD) в записи
AAAA	Поле адреса – представляет начальный адрес записи
TT	Поле типа (из всех типов записи, описанных в спецификации, в файле данных о зашивке ПЗУ разрешено использование только трех типов: 00 – запись данных, 01 – код конца файла и 04 – запись расширенного линейного адреса)
DD	Поле данных. Запись может содержать несколько байт данных. Количество байт должно соответствовать полю LL
CC	Поле контрольной суммы для строки. Контрольная сумма вычисляется путем сложения значений всех байт записи (пар 16-ричных цифр) по модулю 256 (0x100), с последующим переводом в дополнительный формат (Пример: для записи «:010E3200F6**C9**» сумма вычисляется как $0x01 + NOT_8(0x01 + 0x0E + 0x32 + 0x00 + 0xF6) = 0x01 + NOT_8(0x37) =$ $= 0x01 + NOT_8(0b00110111) = 0x01 + 0b11001000 = 0b11001001 =$ **0xC9**, где NOT_8 – побитовая инверсия для модуля 256)

Данные, записываемые в ПЗУ, имеют тип записи 00 (код записи данных). Строка с данным кодом записывается как:

:01AAAA00**DD**CC (для разрядности 8 бит)
:02AAAA00**DDDD**CC (для разрядности 16 бит)
:04AAAA00**DDDDDDDD**CC (для разрядности 32 бит)

где:
01, 02, 04 - поле длины слова (указывает разрядность ПЗУ в байтах),
AAAA - поле адреса,
00 - поле типа с кодом записи данных,
DD - данные, записываемые в ПЗУ (1, 2 или 4 байта в зависимости от разрядности),
CC - контрольная сумма для строки.

В связи с ограничением длины поля адреса в 2 байта, использование только записей с типом 00 не может охватить весь диапазон адресов зашивки ПЗУ. Для полного охвата адресного пространства ПЗУ в файл должны быть включены строки с кодом записи расширенного адреса (04). Использование записи с данным кодом типа, позволяет указать 2 старших байта для 4-байтного адресного пространства. По умолчанию, старшие байты адреса установлены в «0». Если в файле присутствует строка с данным кодом, то для всех последующих строк с кодом 00, устанавливается смещение старших байт адреса, указанное в данной строке. При каждой последующей встрече строки с кодом 04, устанавливается новое смещение.

Запись расширенного адреса имеет вид:

:02000004**UUUU**CC

,где:
02 - поле длины слова (всегда 2 байта),
0000 - поле адреса (всегда 0),
04 - поле типа с кодом записи данных,
UUUU - 2 старших байта адреса,
CC - контрольная сумма для строки.

Помимо указанных типов записи, файл в формате **ihe** должен иметь запись конца файла (код типа - 01). Запись данного типа всегда должна быть последней записью в файле и может присутствовать лишь в единичном экземпляре. Запись всегда имеет следующий вид:

:00000001FF

Пример содержимого файла в формате **ihe** (для информационной организации (1M×8) бит):

Запись	Расшифровка записи
:010000**00**1E**E1**	по адресу **[00000000]** пишутся данные (код 00) **[1E]** Контрольная сумма вычисляется как 0x1 + NOT$_8$(0x01 + 0x00 + 0x00 + 0x00 + 0x1E) = 0xE1
:010A30**00**D5**F0**	по адресу **[00000A30]** пишутся данные (код 00) **[D5]** Контрольная сумма вычисляется как 0x1 + NOT$_8$(0x01 + 0x0A + 0x30 + 0x00 + 0xD5) = 0xF0
:020000**04**000C**EE**	устанавливается смещение старших байт адреса **[000Cxxxx]** для последующих записей. Контрольная сумма вычисляется как 0x1 + NOT$_8$(0x02 + 0x00 + 0x00 + 0x04 + 0x00 + 0x0C) = 0xEE
:010A31**00**D4**F0**	по адресу **[000C0A31]** пишутся данные (код 00) **[D4]** Контрольная сумма вычисляется как 0x1 + NOT$_8$(0x01 + 0x0A + 0x31 + 0x00 + 0xD4) = 0xF0
:010A30**00**C5**00**	по адресу **[000C0A30]** пишутся данные (код 00) **[C5]** Контрольная сумма вычисляется как 0x1 + NOT$_8$(0x01 + 0x0A + 0x30 + 0x00 + 0xC5) = 0x00
:020000**04**0007**F3**	устанавливается новое смещение старших байт адреса **[0007xxxx]**. Контрольная сумма вычисляется как 0x1 + NOT$_8$(0x02 + 0x00 + 0x00 + 0x04 + 0x00 + 0x07) = 0xF3

Пример содержимого файла в формате **ihe** (для информационной организации (512K×16) бит):

Запись	Расшифровка записи
:02030A**00**1E47**8C**	по адресу **[0000030A]** пишутся данные (код 00) **[1E47]** Контрольная сумма вычисляется как 0x1 + NOT$_8$(0x02 + 0x03 + 0x0A + 0x00 + 0x1E + 0x47) = 0x8C
:020000**04**0003**F7**	устанавливается смещение старших байт адреса **[0003xxxx]** для последующих записей. Контрольная сумма вычисляется как 0x1 + NOT$_8$(0x02 + 0x00 + 0x00 + 0x04 + 0x00 + 0x03) = 0xF7
:0203FE**00**0000**FD**	по адресу **[000303FE]** пишутся данные (код 00) **[0000]** Контрольная сумма вычисляется как 0x1 + NOT$_8$(0x02 + 0x03 + 0xFE + 0x00 + 0x00 + 0x00) = 0xFD

Пример содержимого файла в формате **ihe** (для информационной организации (256К×32) бит):

Запись	Расшифровка записи
:020000**04**0002**F8**	устанавливается смещение старших байт адреса **[0002xxxx]** для последующих записей. Контрольная сумма вычисляется как 0x1 + NOT_8(0x02 + 0x00 + 0x00 + 0x04 + 0x00 + 0x02) = 0xF8
:040100**00**0E3A4701**03**	по адресу **[00020100]** пишутся данные (код 00) **[0E3A4701]** Контрольная сумма вычисляется как 0x1 + NOT_8(0x04 + 0x01 + 0x00 + 0x00 + 0xE3 + 0xA4 + 0x70 + + 0x01) = 0x03

Бинарный файл (bin)

Формат обычного двоичного файла. Данные в файле представлены побайтово, начиная с нулевого адреса, без символов разделения и указания адреса. Разрядность для данного формата не может быть определена из содержимого файла, в связи с чем, указание верной разрядности в имени файла с данными о зашивке ПЗУ является обязательным. Размер файла для выбранной информационной емкости является фиксированным (в отличие от остальных форматов) и не зависит от разрядности:

- 524 288 байт для зашивки ПЗУ 4М (1662РЕ1Т, 1662РЕ7У)
- 1 048 576 байт для зашивки ПЗУ 8М (1662РЕ2Т)

Пример содержимого файла в формате **bin**:

Адрес	Данные, записанные в файле (HEX)	Печатные ASCII символы
00000000	57 C6 E7 56 88 C3 E7 56 55 78 04 50 A9 2E 1D 0E	WÆçVˆÃçVUx P©.
00000010	EF F9 2B F1 F1 F8 25 95 EE DF B7 37 97 EE F8 62	ïù+ññø%•îß·7—îøb
00000020	1D AF 4A 18 5F 8E 59 A5 3B BE 54 BB C6 F0 BC 98	¯J _ŽY¥;¾T»Æð¼˜
00000030	9B AD BE 3C 19 73 B3 8B 4A 77 7C 3D 25 D7 79 C7	›-¾< s³‹Jw|=%×yÇ
00000040	37 BE F1 3D 77 7C B5 BD 1A FB 00 C7 D7 5B D4 9C	7¾ñ=w|µ½ û Ç×[Ôœ
00000050	3D 03 B3 9B 5C E7 1D 5F CF D1 6F DE F1 8D 11 DC	= ³›\ç _ÏÑoÞñ Ü
00000060	BB FE CD FA 0F F5 17 27 C0 82 7F 67 B5 BA F5 9B	»þÍú õ 'À, gµºõ›
00000070	F9 0F 1A F7 1B F8 EF 68 88 FA 86 9B F9 9B D2 31	ù ÷ øïhˆú†›ù›Ò1

Особенности адресации при изменении разрядности ПЗУ

При необходимости, внутри файла данных о зашивке ПЗУ может быть использована разрядность, отличная от используемой в рабочем ПЗУ. Так, одно слово для 32-битной зашивки может быть записано в файле как два 16-битных или четыре 8-битных слова, и наоборот. При каждом уменьшении разрядности вдвое, к адресу добавляется 1 старший адресный бит.

Таблица 3, используемые адресные биты для разных типов зашивки:

Разрядность / Ёмкость	8 бит	16 бит	32 бита
4M	A18 ÷ A0	A17 ÷ A0	A16 ÷ A0
8M	A19 ÷ A0	A18 ÷ A0	A17 ÷ A0

Таблица 4, соответствие добавочных адресных бит частям 32-битного слова:

Биты / Разрядность	Q31,Q30,Q29,Q28, Q27,Q26,Q25,Q24	Q23,Q22,Q21,Q20, Q19,Q18,Q17,Q16	Q15,Q14,Q13,Q12, Q11,Q10,Q9,Q8	Q7,Q6,Q5,Q4, Q3,Q2,Q1,Q0
32 бита (+0 бит адреса)	--			
16 бит (+1 бит адреса)	-1		-0	
8 бит (+2 бита адреса)	11	10	01	00

Для записи слова в файл данных о зашивке с меньшей разрядностью, одно слово просто делят на два или четыре, в зависимости от величины уменьшения. К адресу при этом добавляется один или два старших бита в соответствии с табл. 4. Для записи слова в файл с большей разрядностью, два или четыре слова, в зависимости от величины увеличения, записывают как одно слово. При этом один или два старших бита адреса убираются, а порядок объединения слов определяется согласно табл. 4.

Пример 1:

Два 16-битных слова (с адресами для информационной емкости 8М):
1011011101101101 (Adr: 0b**0**111101101111101010)
0110101101100101 (Adr: 0b**1**111101101111101010),
могут быть записаны как одно 32-битное слово:
01101011011001011011011101101101 (Adr: 0b111101101111101010).

Пример 2:

Одно 32-битное слово (с адресом для информационной емкости 4М):
11011011000101100111010101011000 (Adr: 0b01011100011010011),
может быть записано как четыре 8-битных слова:
11011011 (Adr: 0b**11**01011100011010011)
00010110 (Adr: 0b**10**01011100011010011)
01110101 (Adr: 0b**01**01011100011010011)
01011000 (Adr: 0b**00**01011100011010011).

5 Правила и условия эксплуатации

После получения программатора микросхем 1667РТ0ХХ (далее сокращенно именуемое «устройство») внимательно осмотрите устройство и прилагающиеся к нему компоненты, на наличие видимых механических повреждений, вызванных транспортировкой. В случае обнаружения оных сообщите об этом поставщику. После подключения контактных устройств, кабелей (питание, USB), перед включением устройства убедитесь в отсутствии посторонних предметов / объектов снаружи и внутри данного устройства, способных вызвать короткое замыкание или иное нарушение работоспособности изделия.

Подключение устройства к ПК производится при помощи USB кабеля. Алгоритм работы указан в данном руководстве. Подача на устройство напряжения питания величиной больше/меньше заявленной в электрических характеристиках, несоблюдение полярности при подключении блока питания, попадание влаги внутрь данного устройства или его работа вне диапазона указанных в данном документе рабочих температур может привести к неработоспособности либо поломке устройства.

6 Гарантии производителя

Изготовитель гарантирует соответствие устройства требованиям технической документации при соблюдении условий эксплуатации, хранения и транспортирования.

Гарантийный срок эксплуатации устройства - 12 месяцев со дня продажи.

Изготовитель не несет ответственности в случае причинения косвенного или прямого ущерба вследствие эксплуатации и/или неисправности устройства.

Изготовитель гарантирует безвозмездный ремонт устройства в течение указанного срока при соблюдении потребителем условий эксплуатации, транспортирования и хранения.

Гарантийный ремонт не производится в следующих случаях:

- при нарушении положений, приведенных в руководстве пользователя
- если неисправность возникла вследствие попадания в него посторонних предметов, веществ, жидкостей и т.п., а также вследствие неправильного монтажа, эксплуатации, транспортировки или хранения
- при обнаружении механических повреждений изделия, следов ремонта или внесения конструктивных изменений
- если нормальная работа устройства может быть восстановлена его надлежащей настройкой или регулировкой

7 Свидетельство о приемке

Устройство заводской № _____ соответствует паспортным данным и признано годным к эксплуатации.

Дата изготовления: "___" _____ 20___ г.

Дата продажи: "___" _____ 20___ г.

8 Сведения о рекламациях

В случае отказа в работе устройства в период гарантийного срока необходимо акт о неисправности, устройство и паспорт направить по адресу производителя:

Адрес:
Телефон:
e-mail:
Web:

9 Сведения об утилизации

По окончании срока эксплуатации устройство подлежит утилизации в установленном порядке. При утилизации не должны выделяться в почву, воздух или воду вредные для окружающей среды вещества.

10 Лист регистрации изменений документации

Версия документации	Добавление / Обновление / Удаление содержимого
0.2	Добавлены пункты 4.3 «Проведение входного контроля микросхемы», 4.5 «Проведение выходного контроля микросхемы», 4.9 «Сохранение журнала событий (лога) программатора». Обновлены рисунки 5, 6. Обновлен пункт 3.2 «Минимальные требования к ПК». Обновлен пункт 4.2 «Обновление прошивки программатора».