Содержание

Согласовано 4400 ВП МО РФ И.А.Фронтов

ТСКЯ.431295.006И

Изм	Лист	№ докум.	Подп.	Дата
Разраб.		Усков		
Пров.		Шумилин		
Гл.констр		Какоулин		
Н.Контр.		Ануфриева		
Утв..		Какоулин		

Микросхема 1886ВЕ6У Инструкция по программированию

Лит	Лист	Листов
О	1	16

Копировал Формат

1 Подключение микроконтроллера

Для программирования внутренней EEPROM памяти программ микроконтроллеров 1886ВЕ6У используется последовательный интерфейс ISP (Interface Serial Program). В этом режиме программирования задействованы выводы микроконтроллера и напряжения питания, приведённые в таблице 1.

Таблица 1 Выводы ISP интерфейса

Обозначение	В режиме программирования		
	Назначение	Тип	Описание
PA2/RX1/DT1	DT	вход/выход	Последовательные данные (31)
PA3/TX1/CK1	CK	вход	Последовательный синхросигнал (32)
PA1/T0CLK	CLK	вход	Источник синхронизации микроконтроллера (30)
TEST	TEST	вход	Вход для выбора тестового режима (28)
nMCLR	nMCLR	питание	Внешний сброс (27)
U_{CC}	U_{CC}	питание	Напряжение питания (2, 15, 25, 33)
GND	GND	общий	Общий (1, 14, 24, 36, 39, 42)
AU_{CC}	AU_{CC}	питание	Напряжение питания АЦП и ЦАП (12, 46)
AGND	GND	общий	Общий АЦП и ЦАП(13, 11, 45)

2 Перевод микроконтроллера в режим последовательного программирования

Перевод микроконтроллера в режим последовательного программирования (ISP) осуществляется подачей напряжения питания U_{CC} на выводы TEST и MCLRn в следующей последовательности:

1 Подача U_{CC} на вывод TEST.

2 Подача U_{CC} на вывод nMCLR.

Время установки между событиями 1 и 2 должно быть не менее 1 микросекунды. В итоге счётчик команд будет указывать на адрес памяти программ 0xF000. В этой области памяти программ расположено загрузочное ПЗУ (Boot ROM). Этот программный код инициализирует USART/SCI для получения команд. Для выполнения программы из загрузочного ПЗУ необходимо обеспечить подачу синхросигнала на вывод PA1/TOCLK. Для того, чтобы USART/SCI был инициализирован для получения команд, необходимо выполнить следующую последовательность действий:

1 Запустить источник синхронизации микроконтроллера.

2 Ожидать 160 циклов микроконтроллера для инициализации USART/SCI загрузочным ПЗУ.

3 Послать команду по USART/SCI.

3 Команды управления программированием и специальные библиотечные функции.

В ISP режиме модуль USART/SCI сконфигурирован как синхронное ведомое устройство. Микроконтроллер ожидает команду, которая выполняется загрузочным ПЗУ, расположенным в области программной памяти 0xF000-0xF7FF. Программно-аппаратные средства ISP поддерживают 20 команд, приведённых в таблице 2. Кроме того в ПЗУ расположены библиотечные функции приведенные в таблице 3.

Таблица 2 Команды ISP

	Команда	Значение
1	СБРОС УКАЗАТЕЛЯ ПАМЯТИ ПРОГРАММ	0000 0000
2	ЗАГРУЗКА ДАННЫХ	0000 0010
3	ЧТЕНИЕ ДАННЫХ	0000 0100
4	ИНКРЕМЕНТ АДРЕСА	0000 0110
5	СТАРТ ПРОГРАММИРОВАНИЯ	0000 1000
6	ЗАГРУЗКА АДРЕСА	0000 1010
7	ЧТЕНИЕ АДРЕСА	0000 1100
8	СТОП ПРОГРАММИРОВАНИЯ	0000 1110
9	ЗАПИСЬ ДЕЛИТЕЛЯ ЧАСТОТЫ	0000 0001
10	Зарезервировано	0000 0011
11	Зарезервировано	0000 0111
12	СТИРАНИЕ СЕКТОРА	0000 1111
13	ТЕСТ СЕКТОРА НА ЧИСТОТУ	0000 0101
Расширение стандартного набора команд		
14	ЧТЕНИЕ ОЗУ+ПЕРЕФЕРИЯ	0001 0000
15	ЗАПИСЬ ОЗУ+ПЕРЕФЕРИЯ	0001 0001
16	ЗАПРОС ГОТОВНОСТИ	0001 0010
17	ПЕРЕХОД CALL	0001 0011
18	ПЕРЕХОД GOTO	0001 0100
19	ВЕРСИЯ МИКРОКОНТРОЛЛЕРА	0001 0101
20	СТЕРЕТЬ ВСЮ ПАМЯТЬ	0001 0110

Таблица 3 Библиотечные функции ПЗУ

	Название	Параметры	Описание
1	Стирание блока	(L/H)	Очистка содержимого старшей или младшей половины памяти программ
2	Запись блока	(L/H, 16'bDATA)	Запись старшей или младшей половины памяти программ одним числом за один цикл
3	Проверка содержимого блока	(L/H, 16'bDATA)	Проверка старшей или младшей половины памяти программ на равенство одному числу.
4	Стирание строки	(address)	Очистка содержимого строки памяти программ по адресу

	Название	Параметры	Описание
5	Проверка содержимого строки	(address, 16'bDATA)	Проверка содержимого строки памяти программ на равенство одному числу.
6	Запись слова (через регистры)	(address, 16'bDATA)	Программирование ячейки памяти программ через управление регистрами блока.
7	Чтение слова (через регистры)	(address)	Чтение ячейки памяти программ через управление регистрами блока.
8	Вычисление контрольной суммы EEPROM	Стартовый адрес, длина, начальное значение, результат.	Вычисляет контрольную сумму всей памяти EEPROM.

3.1 Сброс указателя памяти программ

Эта команда предназначена для очистки указателя адреса памяти программ. После этого указатель находиться в исходном состоянии и указывает на адрес 0x0000. Временная диаграмма команды приведена на рисунке 1.

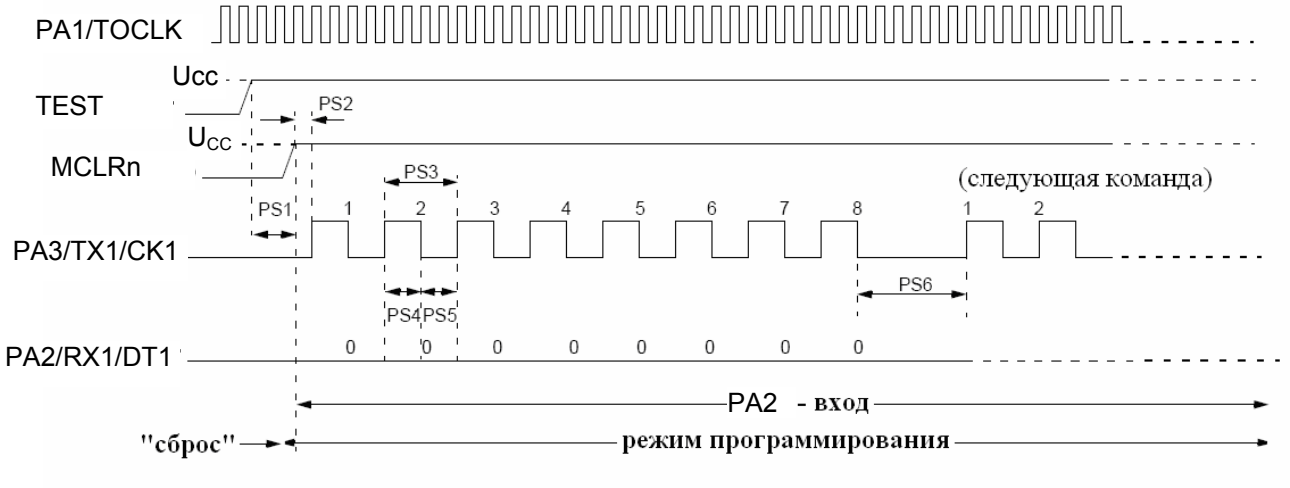

Рисунок 1 – Команда сброса указателя адреса памяти программ

3.2 Инкремент адреса

Эта команда инкрементирует указатель адреса памяти программ. Обычно эта команда следует за командами чтения или программирования. Временная диаграмма команды приведена на рисунке 2.

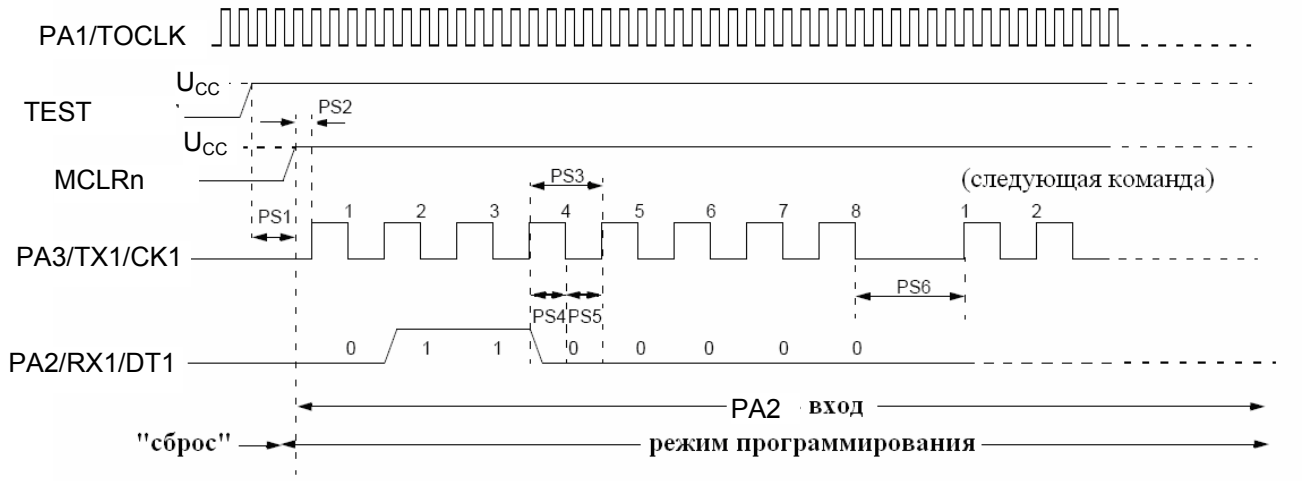

Рисунок 2 Команда инкремента адреса

3.3 Загрузка адреса

Эта команда используется для загрузки указателя адреса программ определённым 16 разрядным значением. Обычно эта команда применяется для доступа к определённому адресному пространству. Временная диаграмма команды приведена на рисунке 3.

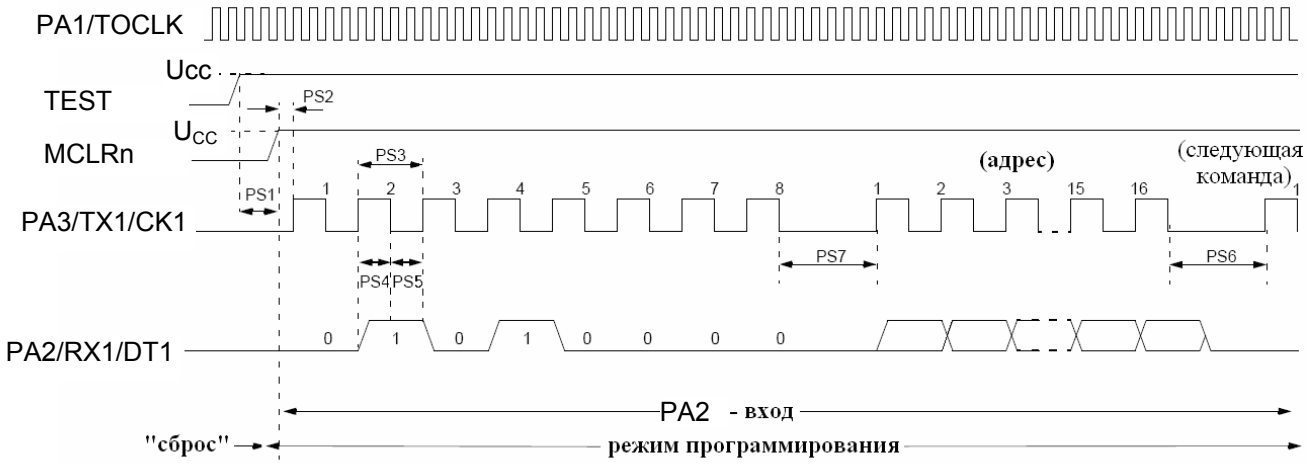

Рисунок 3 – Команда загрузки адреса памяти программ

3.4 Чтение адреса

Эта команда используется для получения текущего указателя адреса памяти программ. Временная диаграмма команды приведена на рисунке 4.

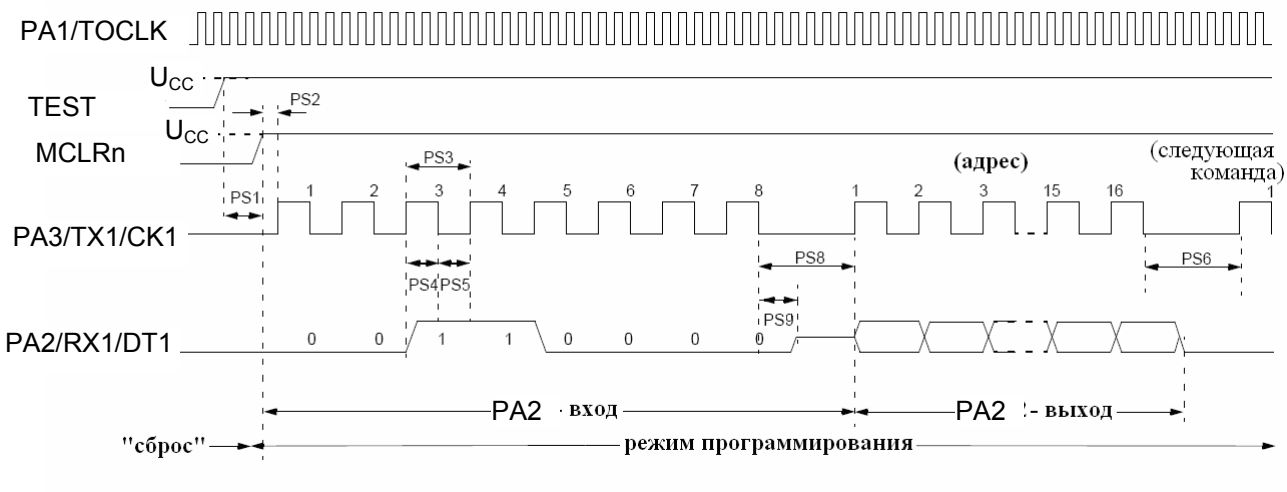

Рисунок 4 – Команда чтения адреса памяти программ

3.5 Загрузка данных

Эта команда загружает 16 разрядные данные, которые должны быть запрограммированы в память программ. Адрес памяти программ может быть модифицирован после загрузки данных. Данные не будут запрограммированы, пока не выполниться команда СТАРТ ПРОГРАММИРОВАНИЯ. Временная диаграмма команды приведена на рисунке 5.

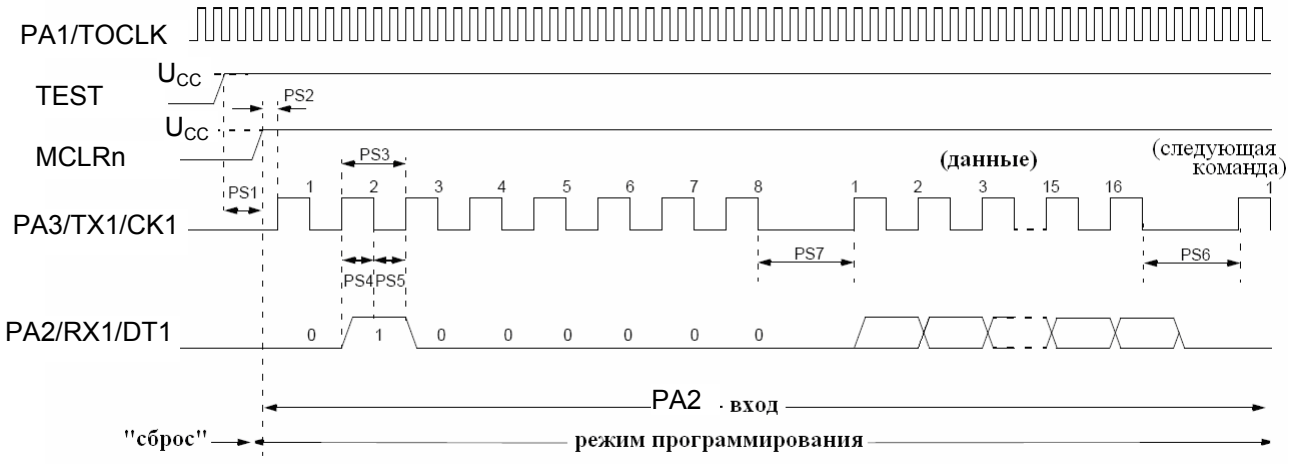

Рисунок 5 – Команда загрузки данных

3.6 Чтение данных

Эта команда читает данные, расположенные по адресу на который ссылается указатель адреса памяти программ. Обычно эта процедура используется в процедуре контроля правильности запрограммированных данных. Временная диаграмма команды приведена на рисунке 6.

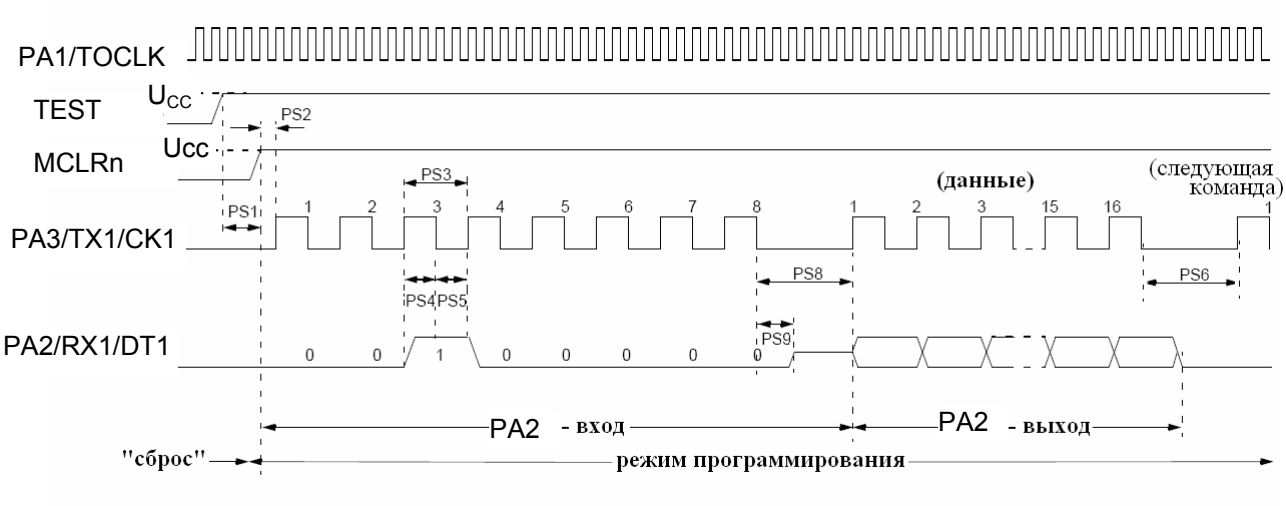

Рисунок 6 – Команда чтения данных

3.7 Старт программирования

Эта команда используется для программирования текущих данных (посланных последней командой ЗАГРУЗКА ДАННЫХ) по адресу памяти программ на который ссылается указатель адреса. Время программирования данных 200 микросекунд. После прохождения этого времени любая команда может быть послана и выполнена. Временная диаграмма команды приведена на рисунке 7.

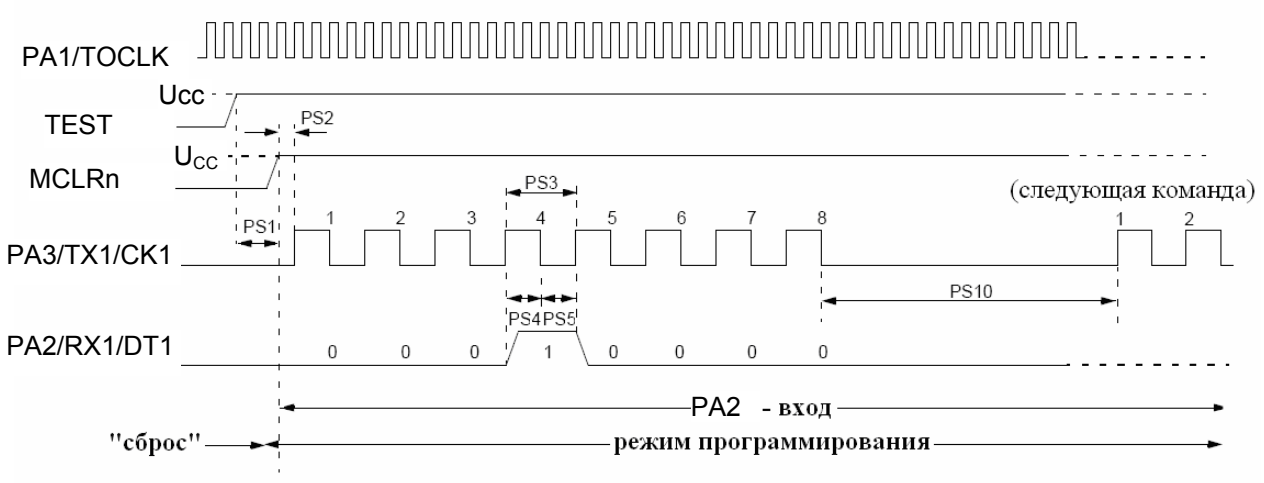

Рисунок 7 –Команда программирования

3.8 Запись делителя частоты

Эта команда предназначена для записи коэффициента деления частоты генератора в 8 разрядный регистр микроконтроллера (EEDIV) с целью выработки импульса записи и стирания памяти программ длительностью 4 мс. Делитель передается в первом байте 16 разрядных данных. Правило выбора коэффициента деления: **K = 4 мс / 651*Tosc**

Временная диаграмма команды приведена на рисунке 8.

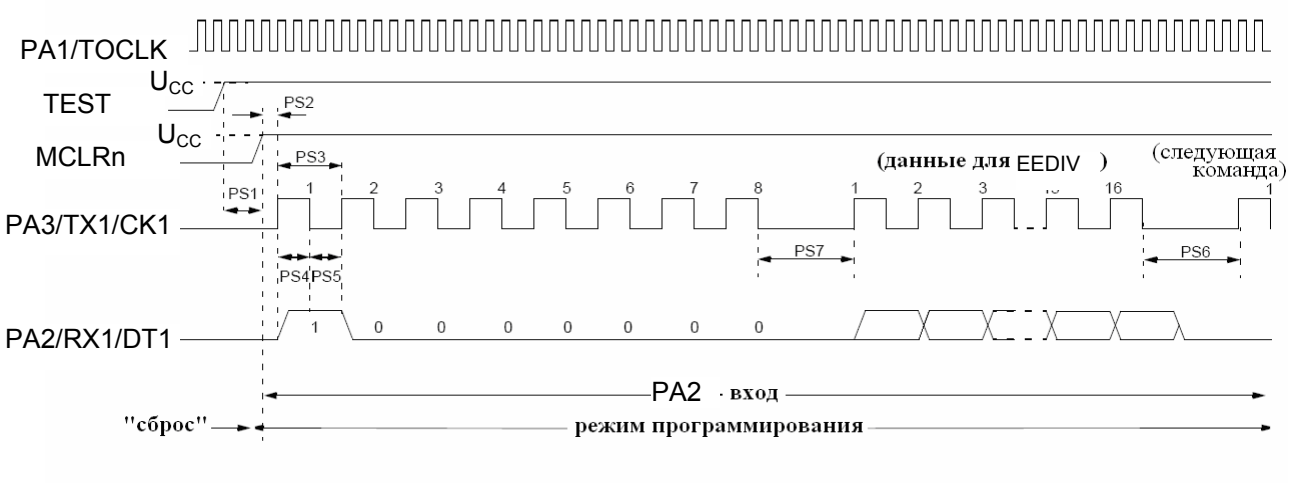

Рисунок 8 – Команда записи делителя частоты

3.9 Стирание сектора

Эта команда предназначена для запуска процедуры стирания одного из 2 секторов EEPROM-памяти. Стираемый сектор выбирается 16 разрядным адресом начала сектора, передаваемым в команде:
1 сектор – 0x0000;
2 сектор – 0x0800;
Первым передается старший байт начального адреса стираемого сектора.
Временная диаграмма команды приведена на рисунке 9.

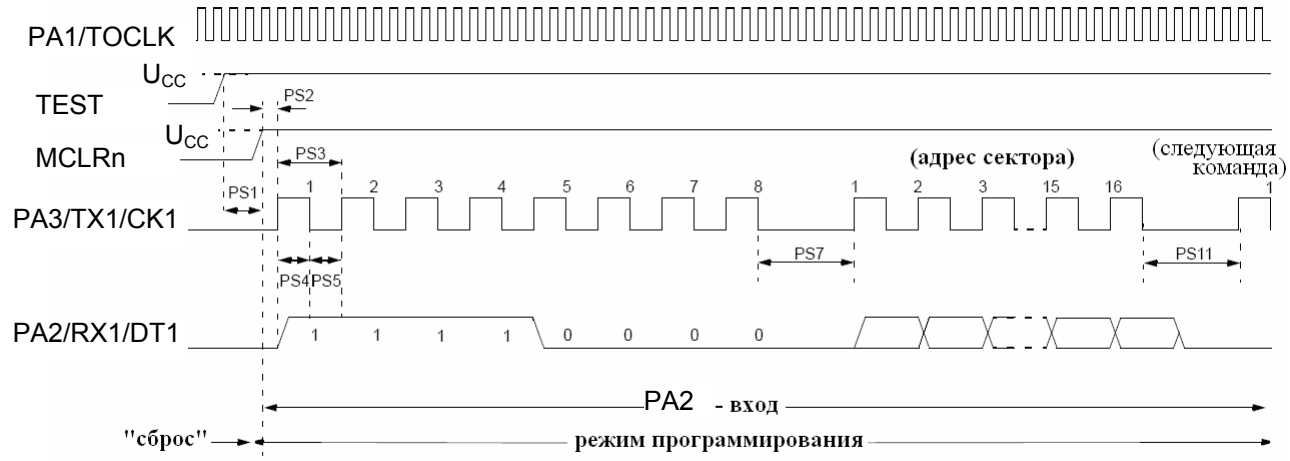

Рисунок 9 – Команда стирания сектора

3.10 Тест сектора на чистоту

Эта процедура проводиться после операции стирание сектора, для проверки чистоты сектора. Проверяемый сектор выбирается 16 разрядным адресом начала сектора, передаваемым в команде:
1 сектор – 0x0000;
2 сектор – 0x0800;
В случае, если все ячейки памяти в адресном пространстве проверяемого сектора содержат значение 0x0000, сектор считается очищенным и из канала USART/SCI

читаются 16 разрядные данные 0xFFFF, в противном случае читаются 0x0000. Временная диаграмма команды приведена на рисунке 10.

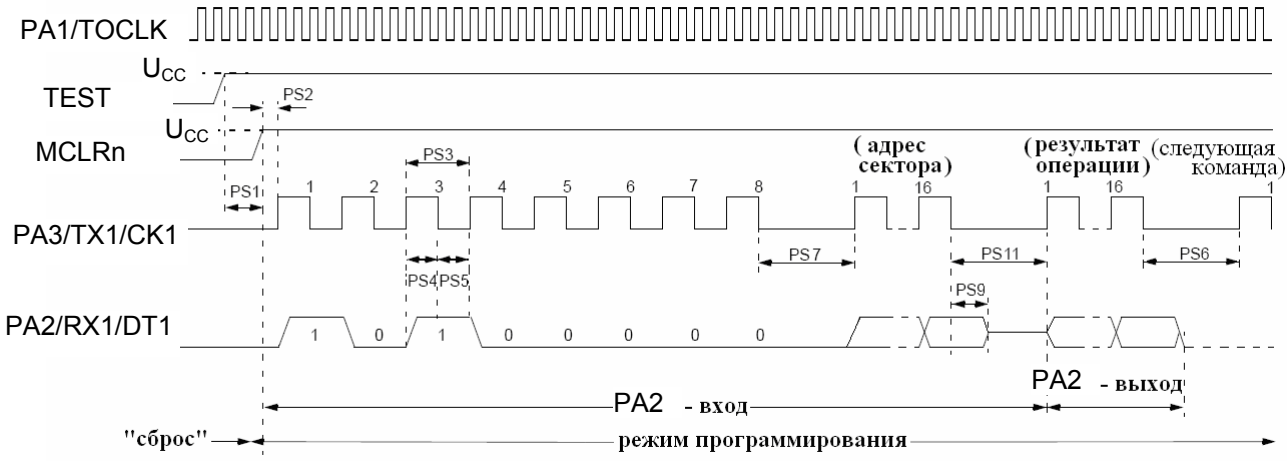

Рисунок 10 – Команда тест сектора на чистоту

3.11 Функция ЧТЕНИЕ ОЗУ+ПЕРЕФЕРИЯ

Команда функции состоит из 4 частей:

1. Запись команды чтения ОЗУ (0x10).
2. Запись банка ОЗУ (регистр BSR).
3. Запись адреса ОЗУ.
4. Чтение полученных данных DATA.

3.12 Функция ЗАПИСЬ ОЗУ+ПЕРЕФЕРИЯ

Команда функции состоит из 4 частей:

1. Запись команды записи ОЗУ (0x11).
2. Запись банка ОЗУ (регистр BSR).
3. Запись адреса ОЗУ.
4. Запись данных DATA.

3.13 Функция ЗАПРОС ГОТОВНОСТИ

Команда функции состоит из 2 частей:

1. Запись команды Запрос Готовности (0x12).
2. Чтение данных DATA. Если принятое значение отличается от 0x55, то микроконтроллер не готов и синхронизация не восстановлена (ожидаемое значение при неготовности микроконтроллера 0xFF). После получения значения равного 0x55 микроконтроллер готов к обработке следующих команд.

3.14 Функция ПЕРЕХОД CALL

Команда функции состоит из 3 частей:

1. Запись команды Переход CALL (0x13).
2. Запись младшей половины адреса.
3. Запись старшей половины адреса.

После выполнения команды для восстановления синхронизации используется команда Запрос Готовности.

3.15 Функция ПЕРЕХОД GOTO

Команда функции состоит из 3 частей:

1. Запись команды Переход GOTO (0x14).
2. Запись младшей половины адреса.
3. Запись старшей половины адреса.

После выполнения команды для восстановления синхронизации используется команда Запрос Готовности.

3.16 Функция ВЕРСИЯ МИКРОКОНТРОЛЛЕРА

Команда функции состоит из 2 частей:

1. Запись команды Версия Микроконтроллера (0x15).
2. Чтение данных DATA.

3.17 Функция СТЕРЕТЬ ВСЮ ПАМЯТЬ

Команда функции состоит из 1 части:

1. Запись команды Стереть Всю Память (0x16).

После выполнения команды для восстановления синхронизации используется команда Запрос Готовности.

4 Механизм восстановления синхронизации

При выполнении команды Переход CALL или GOTO микропроцессорное ядро может уйти на подпрограммы не обеспечивающие обработку запросов по USART. В этом случае при приеме USART новых команд, ядро их не будет обрабатывать и они будут накапливаться в очереди USART. При переполнении очереди будет выставлен флаг OVERFLOW. И дальнейшие принимаемые данные будут игнорироваться. После того как будет выполнена подпрограмма, на которую был осуществлен переход CALL или GOTO, микроконтроллер вернется в обработчик USART. При этом он должен дождаться появления флага OVERFLOW. После чего очистить очередь, считывая ранее полученные команды и игнорируя их (всего будет три команды в очереди). После очистки очереди дождаться команды запроса готовности и вернуть значение результата выполнения подпрограммы, на которую был осуществлен переход CALL иди GOTO.

5 Специальные библиотечные функции ПЗУ

Набор подпрограмм работы с EEPROM
Функции используют адреса ОЗУ 4 банка, 0xF0….0xFF и WREG.

Вызов функций реализован таблицей в коде памяти ROM. Для выполнения функций в режиме программирования необходимо с помощью команд функции ЗАПИСЬ ОЗУ+ПЕРЕФЕРИЯ задать входные параметры, а затем с помощью функции CALL вызвать саму функцию. В рабочем режиме сначала программой выполняемой из EEPORM задаются входные параметры, затем осуществляется переход с помощью CALL на саму функцию. При работе в рабочем режиме по изменению содержимого EEPROM памяти программ необходимо заблокировать все прерывания.

0xF200 goto BLOCK_CLR
0xF201 goto BLOCK_WR
0xF202 goto BLOCK_VF
0xF203 goto ROW_CLR
0xF204 goto ROW_VF
0xF205 goto WORD_WR
0xF206 goto WORD_RD
0xF207 goto CRC_32

№	Название	Параметры	Описание
1	Стирание блока BLOCK_CLR	(L/H)	Очистка содержимого старшей или младшей половины памяти программ
2	Запись блока BLOCK_WR	(L/H, 16'bDATA)	Запись старшей или младшей половины памяти программ одним числом за один цикл
3	Проверка содержимого блока BLOCK_VF	(L/H, 16'bDATA)	Проверка старшей или младшей половины памяти программ на равенство одному числу.
4	Стирание строки ROW_CLR	(address)	Очистка содержимого строки памяти программ по адресу
5	Проверка содержимого строки ROW_VF	(address, 16'bDATA)	Проверка содержимого строки памяти программ на равенство одному числу.
6	Запись слова (через регистры) WORD_WR	(address, 16'bDATA)	Программирование ячейки памяти программ через управление регистрами блока.
7	Чтение слова (через регистры) WORD_RD	(address)	Чтение ячейки памяти программ через управление регистрами блока.
8	Вычисление контрольной суммы EEPROM CRC_32	Стартовый адрес, длина, начальное значение, результат.	Вычисляет контрольную сумму всей памяти EEPROM.

Функция стирания блока BLOCK_CLR.
Адрес начала = 0xF200
Параметры:

Номер блока
ОЗУ(4 банк, 0xF0)
0x00 – первый блок
0x01 – второй блок

Функция записи блока BLOCK_WR.
Адрес начала = 0xF201
Параметры:

Номер блока
ОЗУ(4 банк, 0xF0)
0x00 – первый блок
0x01 – второй блок

Записываемое значение
ОЗУ(4 банк, 0xF2 (старший), 0xF3 (младший))
2 байта

Функция проверки блока BLOCK_VF.
Адрес начала = 0xF202
Параметры:

Номер блока
ОЗУ(4 банк, 0xF0)
0x00 – первый блок
0x01 – второй блок

Ожидаемое значение
ОЗУ(4 банк, 0xF2 (старший), 0xF3 (младший))
2 байта

Адрес первой ошибки
ОЗУ(4 банк, 0xF4 (старший), 0xF5 (младший))
2 байта

Результат выполнения
ОЗУ(4 банк, 0xF6)
0x00 – все ячейки блока равны ожидаемому
0x01 – выявлена ошибка по адресу первой ошибки

Функция стирания строки ROW_CLR.
Адрес начала = 0xF203
Параметры:

Адрес строки
ОЗУ(4 банк, 0xF0 (старший), 0xF1(младший))
4 младший бита 0xF1 – не имеет значения

Функция проверки строки ROW_VF.
Адрес начала = 0xF204
Параметры:

Адрес строки
ОЗУ(4 банк, 0xF0 (старший), 0xF1(младший))
4 младший бита 0xF1 – не имеет значения

Ожидаемое значение
ОЗУ(4 банк, 0xF2 (старший), 0xF3 (младший))
2 байта

Адрес первой ошибки
ОЗУ(4 банк, 0xF4 (старший), 0xF5 (младший))
2 байта

Результат выполнения
ОЗУ(4 банк, 0xF6)
0x00 – все ячейки блока равны ожидаемому
0x01 – выявлена ошибка по адресу первой ошибки

Функция записи слова WORD_WR.
Адрес начала = 0xF205
Параметры:

Адрес слова
ОЗУ(4 банк, 0xF0 (старший), 0xF1(младший))

Записываемое значение
ОЗУ(4 банк, 0xF2 (старший), 0xF3 (младший))
2 байта

Функция чтения слова WORD_RD.
Адрес начала = 0xF206
Параметры:

Адрес слова
ОЗУ(4 банк, 0xF0 (старший), 0xF1(младший))

Полученное значение
ОЗУ(4 банк, 0xF2 (старший), 0xF3 (младший))
2 байта

Функция расчета CRC CRC32.
Адрес начала = 0xF207
Параметры:

Адрес начала
ОЗУ(4 банк, 0xF0 (старший), 0xF1(младший))

Адрес окончания
ОЗУ(4 банк, 0xF2 (старший), 0xF3(младший))

Начальное значение/ Результат
ОЗУ(4 банк, 0xF4 (старший), 0xF5, 0xF6 , 0xF7 (младший))
4 байта

6 Электрические характеристики для последовательного режима программирования

Параметр	Характеристика	Мин.	Тип.	Макс.	Единицы измерен.
U_{CC}	Напряжение питания	4,5	5	5,5	В
f_C	Входная частота на **PA1**	-	-	40	МГц
T_{OSC}	Период тактовой частоты f_C	-	$1/f_C$	-	
T_{CY}	Такт инструкции микроконтроллера	-	$4/f_C$	-	
PS1	Время между установкой $\mathbf{U_{cc}}$ на выводах **TEST** и **nMCLR**	1	-	-	мкс
PS2	Время инициализации интерфейса	160	-	-	T_{CY}
PS3	Период частоты	1	-	-	T_{CY}
PS4	Время установки данных	15	-	-	нс
PS5	Время удержания данных	15	-	-	
PS6	Задержка между последним спадом сигнала CK и первым фронтом CK следующей команды	160	-	-	T_{CY}
PS7	Задержка между последним спадом сигнала CK передачи байта команды и первым фронтом CK записываемого слова данных	160	-	-	T_{CY}
PS8	Задержка между последним спадом сигнала CK передачи байта команды и первым фронтом CK читаемого слова данных	160	-	-	T_{CY}
PS9	Время перенастройки вывода данных с выхода на вход	1	-	-	T_{CY}
PS10	Минимальное время программирования	4	-	-	мс
PS11	Время выполнения процедуры стирания сектора	4	-	-	мс

Лист регистрации изменений

Изм	Номера листов (страниц)				Всего листов (страниц) в докум.	№ докум.	Входящий № сопроводительного докум. и дата	Подп.	Дата
	измененных	замененных	новых	аннулированных					

Подп. и дата
Инв. № дубл.
Взам. инв. №
Подп. и дата
Инв. № подл.

Изм	Лист	№ докум.	Подп.	Дата	ТСКЯ.431295.006И	Лист
						16