

PICKit™ 2
Программатор / Отладчик
Руководство пользователя

Глава 1. Обзор Программатора / Отладчика PIICKit 2.

1.1. Введение

Эта глава описывает свойства программатора / отладчика PIICKit 2 и меню программного обеспечения **PIICKit 2 Programmer**.

1.2. Состав комплекта PIICKit 2 (номер для заказа PG164120).

Комплект PIICKit 2 (номер для заказа PG164120) содержит следующее:

1. Программатор/отладчик PIICKit 2.
2. USB кабель
3. Диск с программным обеспечением PIICKit Starter Kit и MPLAB IDE

Комплекты PIICKit Starter Kit (номер для заказа DV164120) и PIICKit 2 Debug Express (номер для заказа DV164120) дополнительно содержат демонстрационные платы с установленным PIC микроконтроллером.

1.3. Программатор / отладчик разработчика PIICKit 2.

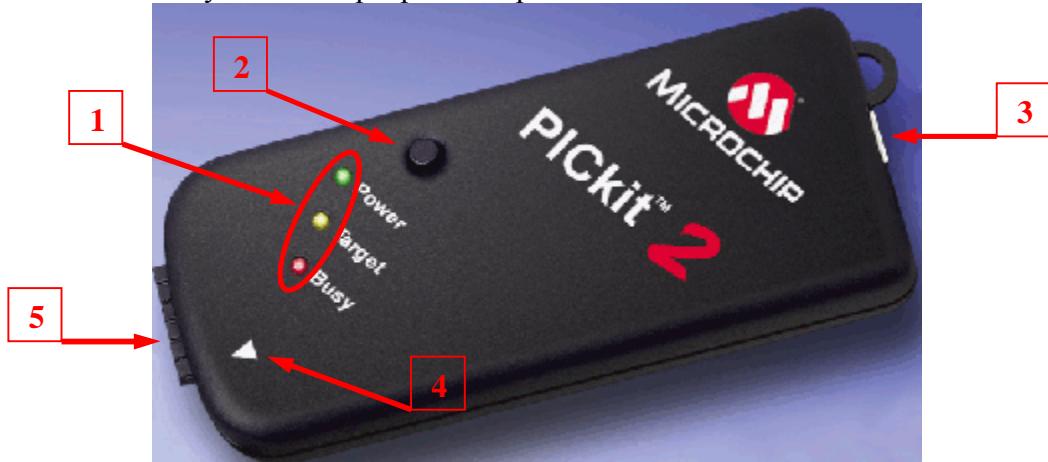
Программатор / отладчик разработчика PIICKit 2 это недорогое средство разработки, поддерживающее программирование большинства микроконтроллеров, микросхем памяти и KeeLOQ производства компании Microchip Technology Inc. Для получения полного списка поддерживаемых микросхем обратитесь к файлу README на диске PIICKit 2 Starter Kit.

Поддержка новых микросхем может быть добавлена при выходе обновлений программного обеспечения PIICKit 2. Последние версии программного обеспечения PIICKit 2 доступны на сайте компании Microchip: www.microchip.com/pickit2

PIICKit 2 так же может использоваться для внутрисхемной отладки некоторых микроконтроллеров. За подробной информацией обратитесь к главе 4 «**PIICKit 2 Debug Express**» данного руководства.

Замечание: Программатор PIICKit 2 не предназначен для промышленного программирования. Для производственных целей рекомендуется промышленный программатор MPLAB PM3 или другие программаторы, предназначенные для промышленных применений.

Рисунок 1.1. Программатор PICkit 2.



1	Светодиоды состояния
2	Кнопка
3	Разъем для подключения USB кабеля
4	Маркер первого вывода
5	Разъем для подключения программируемого устройства

1.3.1 Подключение к USB порту

PICkit 2 имеет USB разъем типа mini-B. Подключите PICkit 2 к компьютеру используя кабель из комплекта поставки.

1.3.2 Светодиоды состояния

Светодиоды состояния отображают статус программатора/отладчика PICkit 2.

1. **Power** (зеленый светодиод) показывает, что PICkit 2 подключен к USB порту.
2. **Target** (желтый светодиод) показывает, что PICkit 2 выдает питание на целевое устройство
3. **Busy** (красный светодиод) показывает, что PICkit 2 занят и выполняет такие функции как программирование, проверку и т.п.

1.3.3 Кнопка

Кнопка может быть задействована для запуска программирования целевого устройства, для этого установите галочку на пункте *Programmer>Write on PICkit Button*.

Кнопка также может использоваться для ввода PICkit 2 в загрузочный режим, в этом режиме можно обновить программное обеспечение программатора PICkit 2.

1.3.4 Разъем для подключения программируемого устройства

Программирующий разъем имеет 6 выводов для подключения целевого устройства. Назначение выводов указано на рисунке 1.2.

Для получения подробной информации о том, как использовать PICkit 2 для внутрисхемного программирования обратитесь к главе 3 «Использование внутрисхемного программирования (ICSP)» данного руководства.

Рисунок 1.2. Разъем программирования.

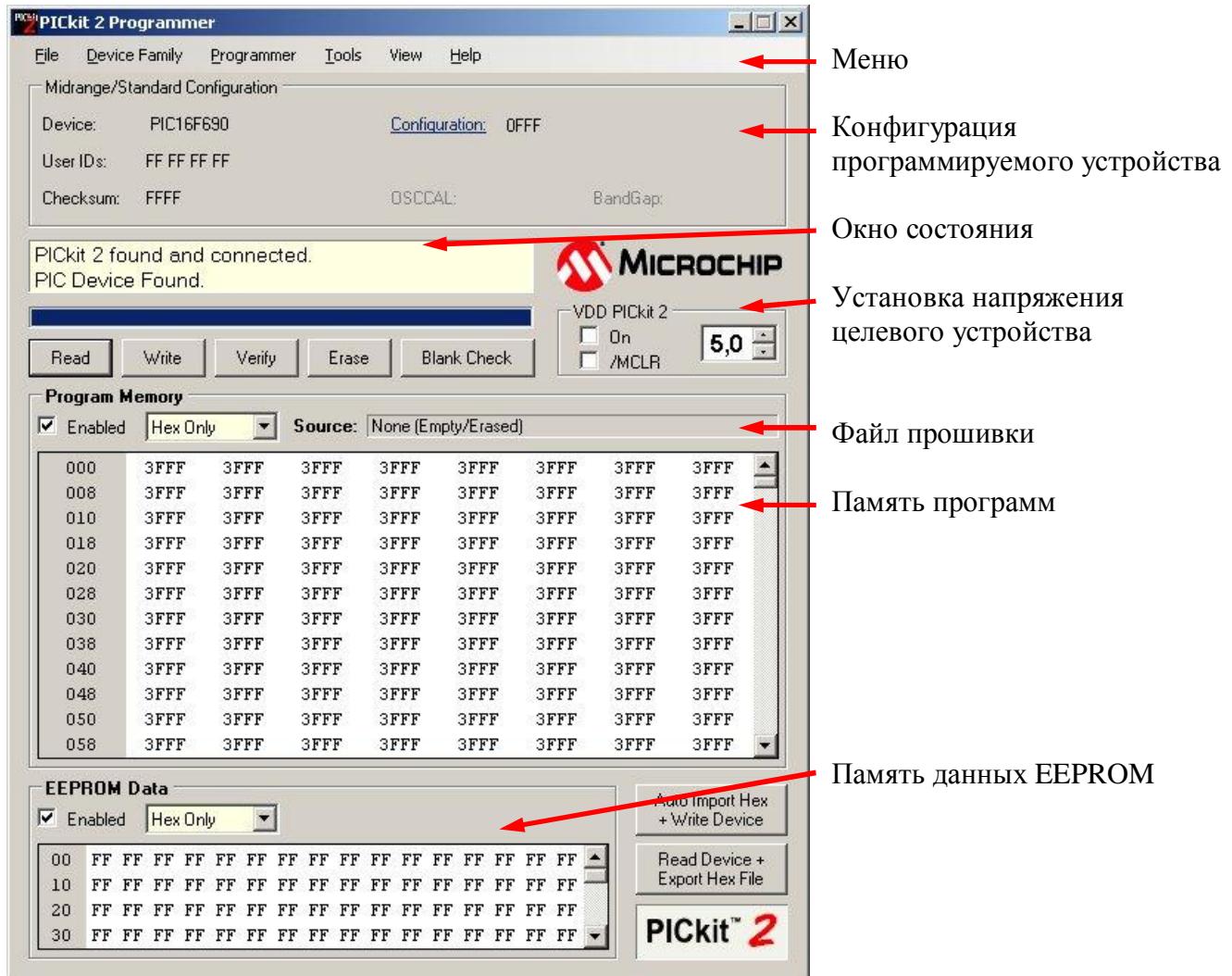


Замечание: Функции выводов программирующего разъема отличаются при программировании микросхем памяти EEPROM и микросхем KeeLOQ. Для получения подробной информации по подключению конкретной микросхемы обратитесь к файлу «PICKit 2 Programmer Readme» (меню *Help>Readme*).

1.4. Программное обеспечение PICkit 2

Программное обеспечение **PICkit 2 Programmer** позволяет программировать все поддерживаемые программатором PICkit 2 микросхемы. Интерфейс программы приведен на рисунке 1.3. Для получения подробной информации обратитесь к главе 2 «Начало работы» данного руководства.

Рисунок 1.3. Интерфейс программы PICkit 2 Programmer.



Глава 2. Начало работы

2.1. Введение

В этой главе описано, как быстро начать работу с программатором/отладчиком PICKit 2. Непосредственно работа с программатором описана в главе 3 «Использование внутрисхемного программирования (ICSP™)», внутрисхемная отладка – в главе 4 «PICkit 2 Debug Express».

- Подключение PICKit 2
- Установка программного обеспечения
- Подключение к программируемой микросхеме
- Управление питанием
- Импорт файла .hex
- Программирование
- Верификация
- Чтение содержимого памяти микроконтроллера
- Защита кода
- Стирание памяти и проверка на чистоту
- Автоматическое программирование/считывание
- Калибровка PICKit 2

2.2. Подключение PICKit 2

- Подключите ваш PICKit 2 к персональному компьютеру с помощью кабеля USB, входящего в комплект поставки
- Подключите PICKit 2 к целевой плате с помощью 6-контактного разъема
- Не подключайте программатор к целевой плате, имеющей внешнее питание, пока он не включен в работающий USB-порт
- Для подключения PICKit 2 к отладочным платам, имеющим разъем RJ-11 (как у ICD 2) используйте переходник AC164110
- При включении PICKit 2 в USB рекомендуется отключать его от целевой платы. Аналогичная рекомендация и при перезагрузке ПК

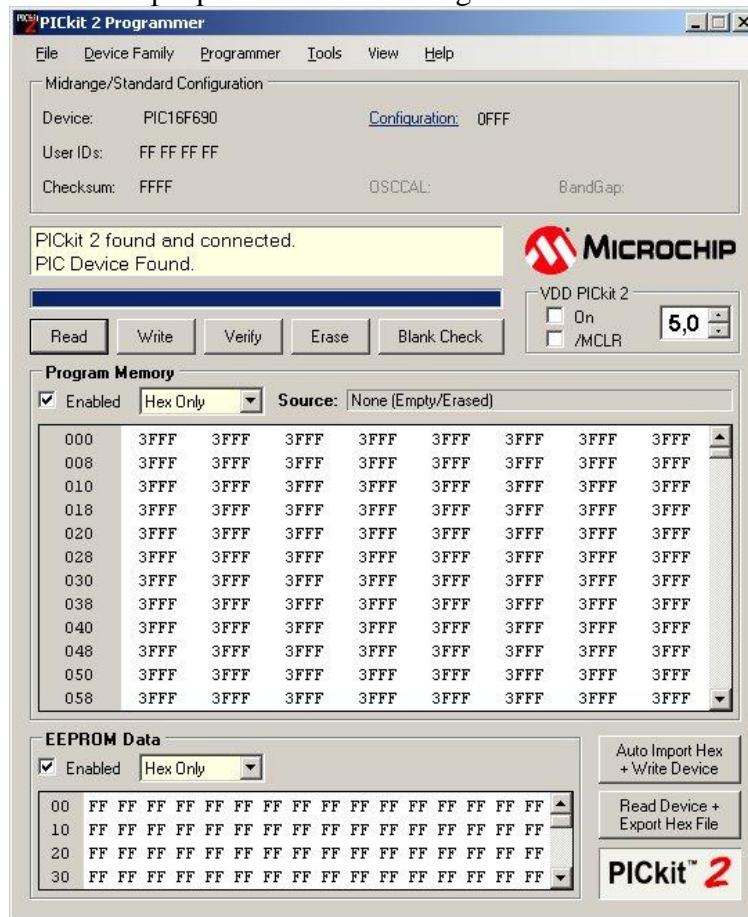
2.3. Установка программного обеспечения

Вставьте CD-ROM PICkit 2 Starter Kit в привод, произойдет автоматический запуск установочной программы. Если установщик не запуститься – откройте вручную файл PICkit_Starter_Kit_Welcome.htm

Наиболее новая версия программного обеспечения всегда доступна на сайте www.microchip.com/pickit2

После установки запустите программу PICkit 2 Programmer. Внешний вид оболочки приведен на рис. 2.1.

Рисунок 2.1. Оболочка программы PICkit 2 Programmer.



2.4. Подключение к программируемой микросхеме

PICKit 2 поддерживает программирование множества микроконтроллеров Microchip PIC и микросхем памяти EEPROM. Список поддерживаемых устройств содержится в файле *readme* на установочном CD или вызывается из меню *Help>Readme*.

При запуске программы производится автоматическое определение типа подключенного контроллера и его отображение в окне *Configuration* (рис.2.2).

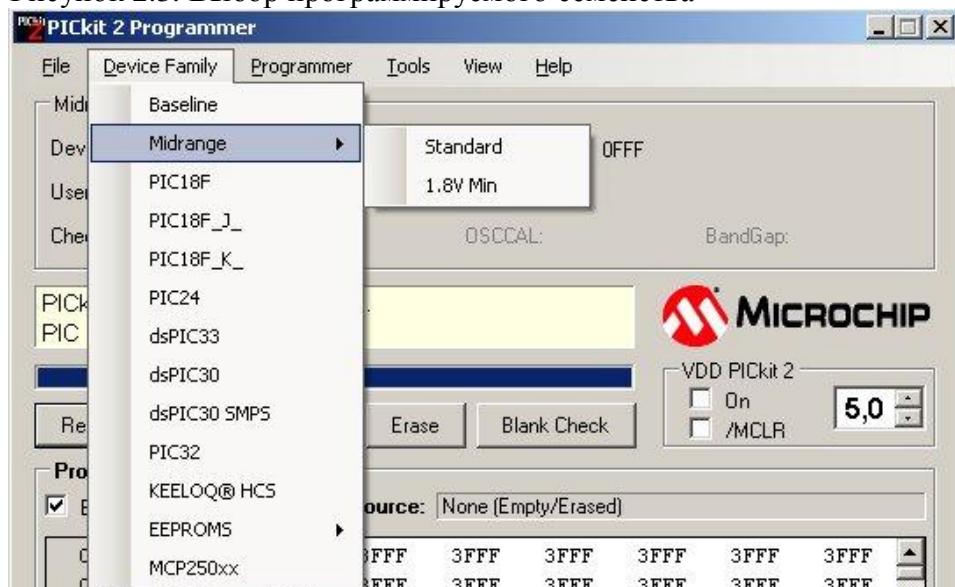
Рисунок 2.2. Определение подключенного контроллера.



Если устройство не определилось – проверьте подачу питающего напряжения (см. п.2.5) и надежность подключения к целевой плате.

Можно в любой момент выбрать нужное вам семейство, воспользовавшись меню *Device Family*, при этом PICKit 2 попытается соединиться с целевым устройством (рис.2.3).

Рисунок 2.3. Выбор программируемого семейства

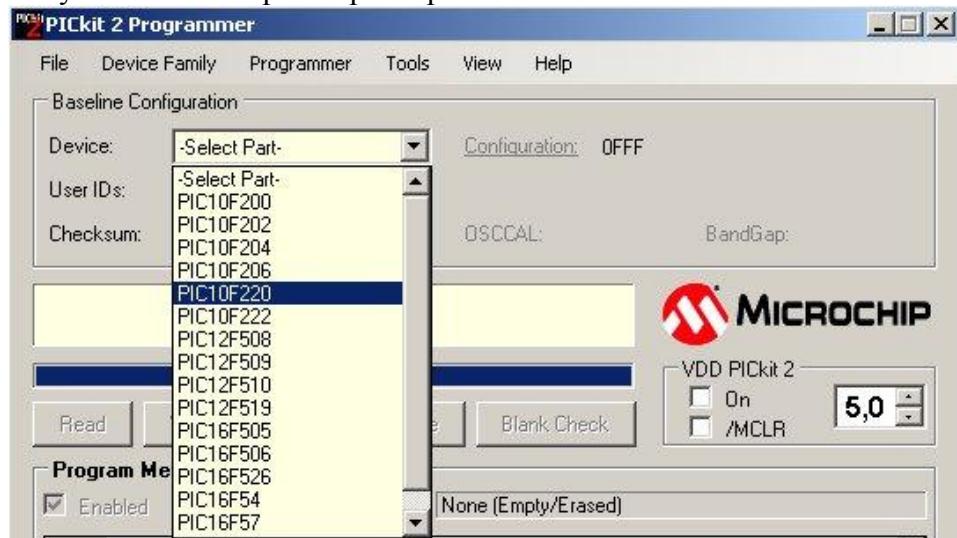


При выборе семейства Baseline, а также микросхем KEELOQ® и EEPROM, необходимо также выбрать конкретное изделие из выпадающего списка (рис.2.4), т.к. в этих микросхемах нет идентификационных битов (device ID).

Внимание

При выборе контроллера из семейства Baseline будьте внимательны – эти контроллеры на имеют идентификатора (device ID) и в случае неправильного выбора устройства возможно стирание калибровочной константы OSCCAL

Рисунок 2.4. Выбор контроллеров базового семейства.



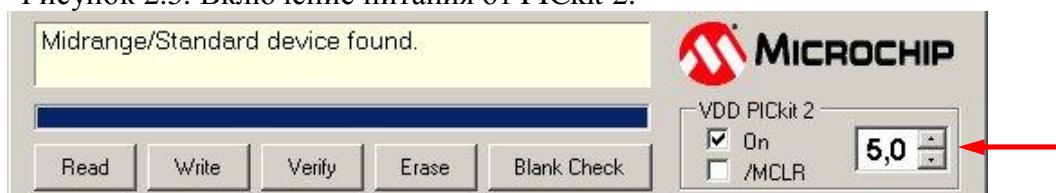
2.5. Управление питанием

При работе с программатором PICKit 2 возможны два варианта питания целевой микросхемы: от PICKit 2 и внешнее питание.

2.5.1. Питание от PICKit 2

Если используется питание от PICKit 2, отдельно подавать питание на плату не нужно, т.к. программатор измерит его и не даст подать питание через себя. Если плата не запитана, то оболочка дает возможность установить значение питающего напряжения, подаваемого с PICKit 2 (рис. 2.5).

Рисунок 2.5. Включение питания от PICkit 2.



Для непосредственной подачи напряжения выберите значение *On*.

Замечание: Если PICKit 2 не увидит внешнего напряжения питания на целевой плате, то он автоматически выдаст питание на плату при программировании, независимо от выбранного значения *On*.

В случае короткого замыкания или превышении максимального тока запитки выдается сообщение об ошибке (рис. 2.6). Потребление целевой платы не должно превышать 25 мА, при этом время нарастания питающего напряжения при включении составляет не более 500 мкс.

Внимание

Максимальный ток через порт USB ограничен значением 100 мА. В случае, если целевая плата и PICKit 2 требуют большего суммарного тока, необходимо использовать внешнее питание

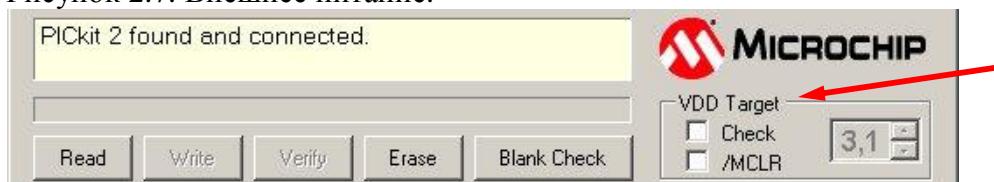
Рисунок 2.6. Ошибка V_{DD}

2.5.2. Внешнее питание

Целевая плата может питаться от собственного источника питания. PICKit 2 автоматически детектирует наличие внешнего питания и, в случае его наличия, в оболочке меняется заголовок окна с *VDD PICkit 2* на *VDD Target*, отключается возможность подачи питания и отображается значение внешнего питающего напряжения (рис. 2.7). Щелчок по галочки *Check* обновляет отображенное питающее напряжение. В случае пропадания внешнего питания оболочка переключиться в режим подачи питания от PICKit 2 (см. п.2.5.1).

Замечание: Разрешенный диапазон напряжений внешнего питания составляет 2.5..5 В

Рисунок 2.7. Внешнее питание.



2.6. Импорт .hex файлов

Для импорта файла прошивки в формате *.hex* выберите пункт меню *File>Import HEX*. В случае, если в файле прошивки отсутствуют какие-либо конфигурационные биты, оболочка выдаст предупреждение. Для правильного сохранения текущей прошивки в файл *.hex* выберите *File>Export* в меню оболочки MPLAB IDE.

Рисунок 2.8. Импорт hex файла.



2.7. Программирование микросхем

После правильного выбора семейства микросхем и импорта файла прошивки возможно программирование целевой микросхемы по кнопке *Write* (рис. 2.9).

Рисунок 2.9. Кнопка Запись.



Микросхема будет стерта и запрограммирована загруженной прошивкой.

Большая часть микроконтроллеров поддерживает режим общего стирания (Bulk Erase), доступный при минимальном напряжении питания, часть контроллеров также поддерживают блочное стирание (Row Erase). Процедура блочного стирания занимает больше времени, нежели общее стирание, но доступно при пониженных напряжениях питания. PICKit 2 автоматически переключается на блочное стирание при невозможности выполнения общего стирания. Если микроконтроллер не поддерживает блочное стирание – выдается предупреждение. Список контроллеров, поддерживающих блочное стирание, доступен в файле *readme*.

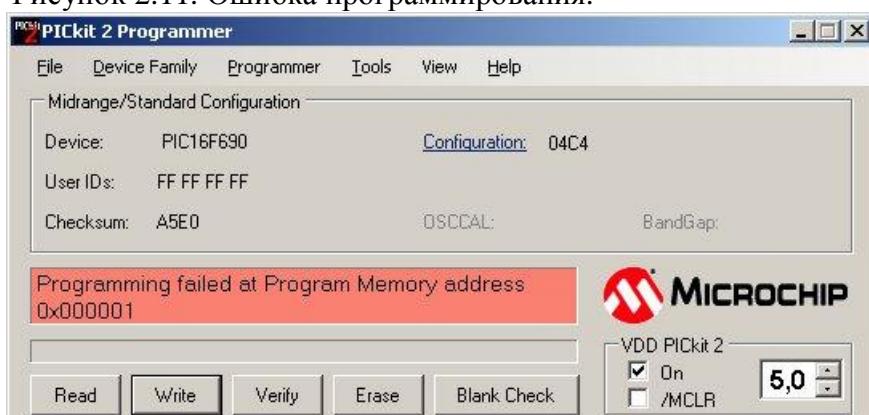
Ход выполнения процедуры программирования отображается в строке статуса. В случае, если программирование прошло успешно, строка становится зеленого цвета и на ней пишется *Programming Successful* (рис. 2.10).

Рисунок 2-10. Успешное завершение записи.



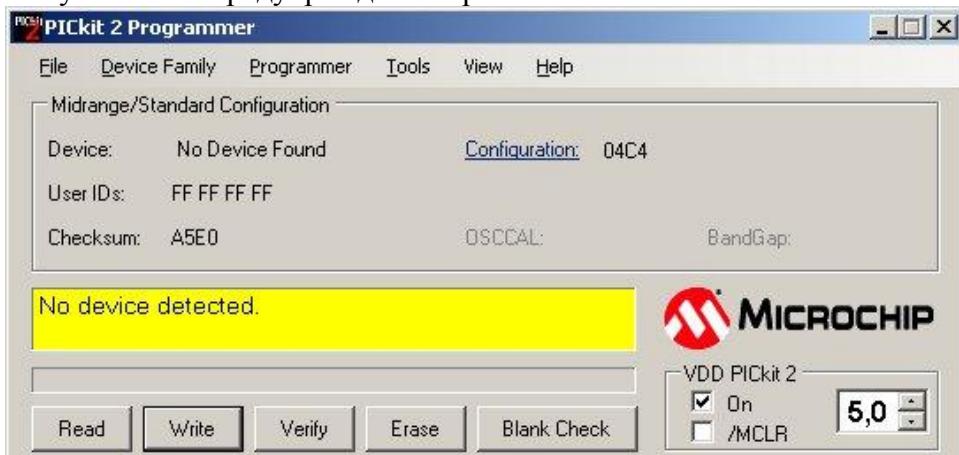
В случае ошибки строка становится красной и на ней пишется *Programming Failed* (рис. 2.11). В этом случае попробуйте повторить процедуру программирования.

Рисунок 2.11. Ошибка программирования.



В других случаях строка статуса становится желтой и на ней пишется причина предупреждения, например, нет соединения с целевым устройством (рис. 2.12).

Рисунок 2.12. Предупреждение при записи.



2.7.1. Программирование определенного раздела памяти

Если микроконтроллер имеет встроенную память EEPROM, то возможно отключение ее программирования в процессе общего программирования микросхемы. При ручном стирании будет стерта вся память.

Выбор раздела, помимо программирования, влияет аналогичным образом и на процедуры верификации и считывания.

2.7.2. Автоматическая загрузка файла прошивки

Перед каждым программированием (по нажатию кнопки *Write*) оболочка автоматически проверяет дату импортированного файла *.hex* с датой этого же файла на диске. Если на файл на диске более новый, то производится автоматический импорт этого файла.

Данная особенность позволяет автоматически использовать наиболее новую прошивку, сгенерированную MPLAB IDE, в т.ч. при режиме работы *Program on PICkit Button*, т.е. просто нажимая кнопку на корпусе программатора без переключения в окно оболочки. Этот режим работы настраивается в меню *Tools>Program on PICkit Button*.

2.8. Верификация прошивки

Функция верификации сравнивает содержимое микросхемы с импортированным .hex файлом. Производится сравнение памяти программ, EEPROM, идентификационных битов и битов конфигурации. В пункте меню *Programmer>Verify on Write* можно настроить автоматическую верификацию при программировании.

Рисунок 2.13. Кнопка верификации.



Если верификация прошла успешно, строка состояния становится зеленой и на ней появляется надпись *Device Verified*. Если нашлось несоответствие, то строка становится красной и на ней пишется область памяти, где произошло несовпадение.

2.9. Чтение прошивки

Считывание прошивки из микросхемы производится по нажатию кнопки *Read*. Содержание областей памяти отображается в соответствующих окнах. Если при программировании для микросхемы была установлена защита кода, то при считывании будут считаны нули.

Рисунок 2.14. Кнопка чтения.



2.10. Защита кода

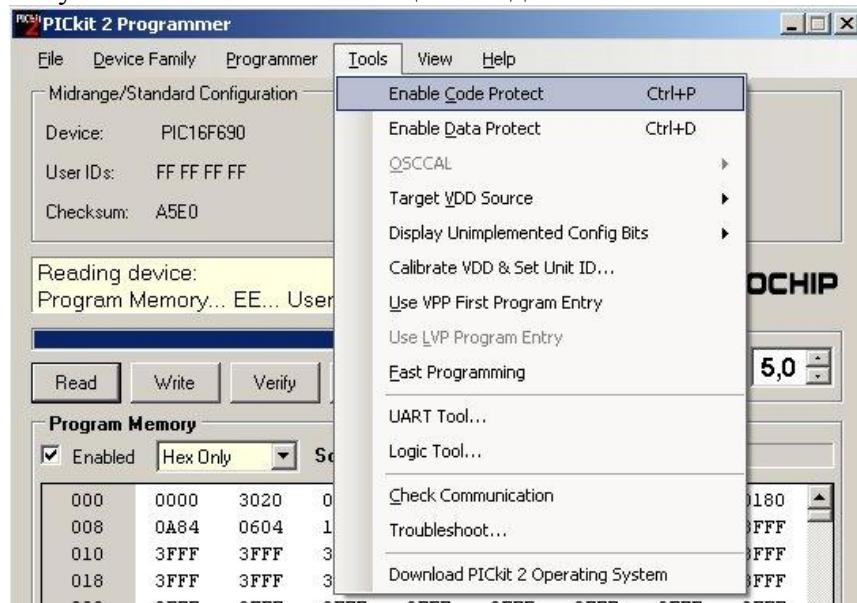
Память программ микроконтроллера и память данных EEPROM имеет защиту от считывания (защиту кода).

Для защиты необходимо:

1. Импортировать файл прошивки (см. п.2.6)
2. Включить защиту кода (меню *Tools>Enable Code Protect*, рис. 2.15) и/или защиту EEPROM (меню *Tools>EnableData Protect*)
3. Запрограммировать микросхему

При обращении к защищенным областям памяти программатор считывает нули. Для снятия защиты с вашей прошивки необходимо выключить защиту кода и EEPROM и перепрограммировать микросхему.

Рисунок 2.15. Включение защиты кода.



2.11. Стирание памяти и проверка памяти микросхемы на чистоту

Функция стирания очищает содержимое всех областей памяти (память программ, EEPROM, идентификационные биты и биты конфигурации), независимо от установленных параметров программирования (см. п.2.7).

Для стирания памяти нажмите кнопку *Erase*.

Замечание: Функция стирания всегда использует режим общего стирания, который требует напряжение питания выше минимального, даже для микросхем, поддерживающих блочное стирание.

Для проверки памяти микросхемы на чистоту нажмите кнопку *Blank Check*.

Рисунок 2.16. Кнопка стирания



2.12. Автоматическое программирование/считывание

В оболочке имеются две специализированные кнопки для ускорения процедур программирования и считывания прошивок.

Рисунок 2.17. Кнопки автоматизации.



2.12.1. Автоматический импорт прошивки и программирование

Для выполнения этой операции нажмите кнопку *Auto Import Hex + Write Device*. По нажатию этой кнопки открывается диалоговое окно выбора прошивки, по умолчанию выбирается предыдущая прошивка. После подтверждения прошивки она импортируется в память и прошивается в память микросхемы. В процессе дальнейшей работы производится мониторинг загруженной прошивки (см. п.2.7.2).

При использовании этой функции остальные возможности программирования отключаются.

2.12.2. Автоматическое считывание прошивки и экспорт в .hex файл

По нажатию кнопки *Read Device + Export Hex File* производится считывание прошивки из памяти микросхемы и открытие диалогового окна сохранения файла.

2.13. Калибровка PICKit 2

Напряжение, выдаваемое программатором на целевую плату, может зависеть от конкретного экземпляра PICKit 2 и реализации порта USB в персональном компьютере. В оболочке имеется возможность калибровки этого напряжения.

Для каждого конкретного PICKit 2 можно задать собственный идентификатор (имя программатора).

2.13.1. Калибровка питающего напряжения

Калибровка позволяет увеличить точность выдаваемого напряжения и точность контроля внешнего напряжения питания. Калибровочное значение хранится в энергонезависимой памяти программатора и используется также при работе с MPLAB IDE.

Для калибровки необходим мультиметр или другой прибор для измерения напряжения. Необходимо отключить программатор от целевой платы, выбрать пункт меню *Tools>Calibrate Vdd & Set Unit ID...* и следовать указаниям мастера калибровки.

Замечание: Напряжение, выдаваемое программатором на плату, ограничено напряжением, получаемым с шины USB минус падение на диоде. Для ноутбуков это напряжение может быть 4,2 В и ниже.

2.13.2. Задание имени программатора

В процессе калибровки возможно задать уникальный идентификатор (имя) программатора. Это имя будет отображаться в строке статуса оболочки программатора (рис. 2.19) и в окне *Output* среды программирования MPLAB IDE.



Глава 3. Использование внутрисхемного программирования (ICSP™).

Введение

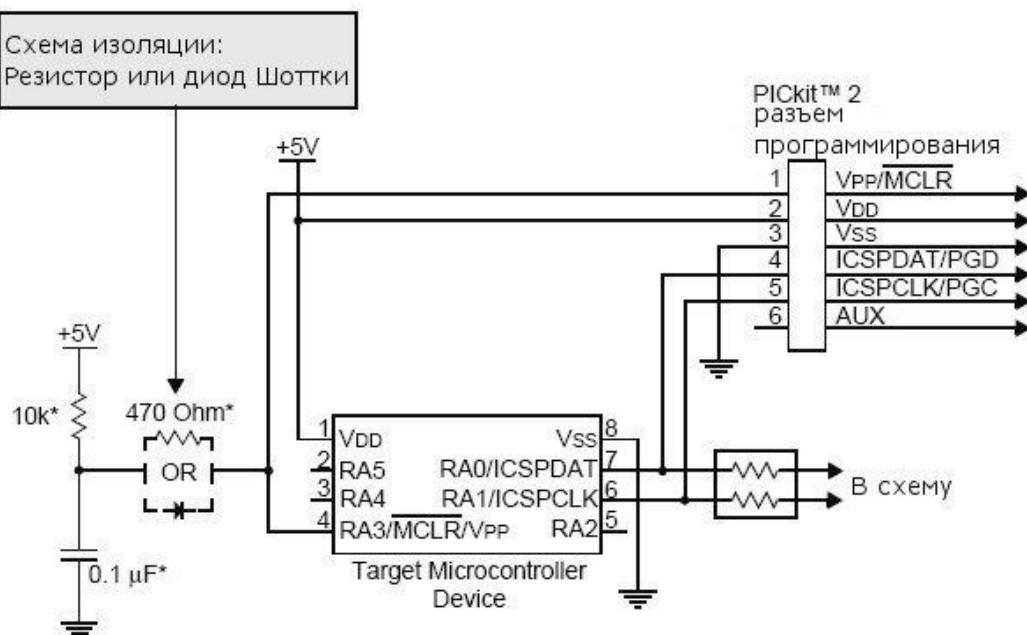
Отладчик и программатор разработчика PICkit 2 может программировать установленные в плату микроконтроллеры. Внутрисхемное программирование (In-Circuit Serial Programming – ICSP) требует пять проводов:

- V_{PP} – напряжение программирования. Когда подается это напряжение микроконтроллер входит в режим программирования.
- ICSPCLK или PGC – линия тактирования; односторонняя линия тактирования от программатора к микроконтроллеру;
- ICSPDAT или PGD – линия данных; двунаправленная линия последовательных данных, синхронна с линией тактирования.
- V_{DD} – плюс напряжение питания;
- V_{SS} – минус напряжения питания (земля).

В любом случае схема должна проектироваться так, чтобы требуемые сигналы проходили к микроконтроллеру без искажения формы. Рис. 3.1 показывает типовую схему подключения микроконтроллера при внутрисхемном программировании. Для успешного внутрисхемного программирования необходимо соблюдать меры предосторожности, которые описаны в следующих пунктах.

Замечание: Для каждого конкретного программируемого устройства пожалуйста ознакомьтесь со спецификацией на программирование, которую можно найти на сайте Microchip www.microchip.com

Рисунок 3.1. Типовая схема внутрисхемного программирования.



* Типовое значение

3.2. Изолирование вывода $V_{PP}/\overline{MCLR}/PORT$

Необходимо учесть, что напряжение программирования V_{PP} имеет типовое значение +12В. Это может предоставить некоторые проблемы в следующих случаях:

Если вывод V_{PP} используется как вывод \overline{MCLR}

Типовая рекомендованная схема включения имеет подтягивающий резистор и конденсатор. Необходимо принять меры, чтобы скорость нарастания напряжения V_{PP} не уменьшилась и превышает скорость нарастания указанную в спецификации на программирование (обычно 1 мкс).

Если в схеме используется супервизор питания или кнопка, подключенная к выводу \overline{MCLR} , то в этом случае рекомендуется чтобы они были изолированы от напряжения программирования V_{PP} с помощью диода Шоттки или ограничительный резистор как показано на рис. 3.1. Для получения дополнительной информации об использовании супервизоров питания в схемах с внутрисхемным программированием, обратитесь к инструкции по применению AN820 “System Supervisors in ICSP™ Architectures” (DS00820).

Если вывод V_{PP} используется как выход порта

Если к разрабатываемой схеме нельзя подключать выводы с напряжением программирования V_{PP} 12В, то в этом случае рекомендуется использование диода Шоттки или ограничительного резистора как показано на рис. 3.1 для защиты схемы.

3.3. Изолирование выводов ICSPCLK (PGC) и ICSPDAT (PGD).

Выводы ICSPCLK (PGC) и ICSPDAT (PGD) необходимо изолировать от схемы для предотвращения искажения сигналов программирования внешней схемой. Сигнал ICSPCLK (PGC) односторонний тактовый сигнал от программатора к программируемому устройству. Сигнал ICSPDAT (PGD) – двухсторонний сигнал данных. Если конструкция позволяет, то выделите эти выводы только для внутрисхемного программирования. Однако если требуется, чтобы эти выводы использовались в схеме, проектируйте схему так, чтобы не изменялись уровни и фронты сигналов. Изолирующая схема сильно зависит от приложения. Рис. 3.1 показывает один из возможных вариантов с применением последовательных резисторов для изоляции сигналов программирования от схемы.

3.4 Напряжение питания V_{DD} .

Во время внутрисхемного программирования необходимо чтобы программируемое устройство было запитано в соответствии со спецификацией. Обычно напряжение питания программируемого устройства соединено с напряжением питания всей схемы. Схема может получать питание от программатора PICkit 2 или иметь собственный источник питания. Необходимо соблюсти меры предосторожности, которые описаны в следующих пунктах.

3.4.1. Схема запитана от PICkit 2.

С помощью PICkit 2 можно выставлять напряжение между максимальным и минимальным значениями, которые позволяет спецификация программирования на конкретное устройство, за исключением, если минимальное напряжение не ниже 2.5В. Убедитесь что выставлено нужное напряжение для схемы до того как начнете программировать устройство или включите напряжение питания V_{DD} .

Внимание

Ток USB порта ограничивается значением 100mA. Если схема и программатор суммарно требуют больший ток, то USB порт может выключиться. Используйте внешнее питание если требуется больший ток

Замечание: Потребление схемы должно быть ограничено уровнем 25mA, когда программатор используется для питания внешней схемы. Убедитесь в том, что ваша схема не замедлят рост напряжения питания V_{DD} на время не более чем 500 мкс.

3.4.2. Схема запитана от внешнего источника питания.

PICkit 2 может использоваться с устройством, которое имеет собственный источник питания с напряжением в диапазоне от 2,5 до 5,0В.

3.4.3 Использование режима общего (Bulk) стирания.

Некоторые микросхемы используют режим общего (Bulk) стирания памяти программ, памяти данных EEPROM, слов конфигурации и идентификации. Обычно функция общего стирания памяти требует напряжения питания микроконтроллера (V_{DD}) в диапазоне от 4.5 до 5,5В (уточните в спецификации на программирование для конкретной микросхемы).

Такой диапазон напряжений может создать некоторые сложности, если конечное изделие разработано для работы в другом диапазоне напряжений питания. Для того чтобы использовать режим общего стирания памяти необходимо чтобы в схеме были предусмотрены требования к режиму общего стирания памяти и защищены все чувствительные цепи.

Если прибор имеет напряжение питания V_{DD} ниже чем требуется для режима общего стирания, то пользователь увидит сообщающее от программы до осуществления процедуры стирания памяти.

3.5 V_{SS}

«Земля» схемы должна быть подключена к «земле» программатора PIICKit 2 (V_{SS}).

3.6 Длина кабеля

Минимизируйте длину проводников линий внутрисхемного программирования от PIICKit 2 до программируемого устройства. Минимизация длины проводников необходима для сохранения величины и формы сигналов. Форма и величина сигналов будет влиять на успешное программирование устройств.

3.7 Программирование последовательной памяти EEPROM и KeeLOQ HCS кодеров/кодеков.

Назначение выводов и сигналов PIICKit 2 для программирования микросхем памяти и KeeLOQ отличается от описанных в пункте «3.1 Введение» и рисунке 3.1. Для получения подробной информации по подключению конкретной микросхемы обратитесь к файлу «PIICKit 2 Programmer Readme» (меню *Help>Readme*).

Обратите внимание, что микросхемы памяти и KeeLOQ могут не программироваться внутрисхемно. Попытки внутрисхемного программирования последовательной памяти EEPROM могут натолкнуться на ошибки программирования из-за конфликтов с другими устройствами, подключенными к последовательнойшине данных.

Глава 4. PIKit 2 Debug Express

4.1. Введение

Помимо непосредственно операции программирования программатор/отладчик PIKit 2 в комплексе с бесплатной средой разработки MPLAB IDE (www.microchip.com/mplab), поддерживает внутрисхемную отладку некоторых PIC-микроконтроллеров.

Программное обеспечение PIKit 2 Debug Express совместно со средой MPLAB IDE позволяет осуществлять пошаговое и непрерывное выполнение программы с точками останова непосредственно PIC контроллера в составе Вашего конечного устройства.

После останова процессора, содержимое регистров доступно для чтения и модификации.

За более подробной информацией об использовании среды MPLAB IDE обратитесь к следующей документации:

- MPLAB® IDE User.s Guide (DS51519)
- MPLAB® IDE Quick Start Guide (DS51281)
- MPLAB® IDE On-line Help (www.microchip.com/mplab)

4.2. Отладочный комплект PIKit 2 Debug Express

Отладочный комплект PIKit 2 Debug Express содержит:

1. Программатор/отладчик PIKit 2
2. USB-кабель
3. отладочную плату с 44-выводным PIC-контроллером*
4. 2 CD диска: PIKit 2 Starter Kit и MPLAB IDE

* помимо платы, которая входит в комплект PIKit 2 Debug Express, можно совместно с комплектом изучать и работать со следующими платами:

- Отладочная плата из комплекта Starter Kit совместно с головкой AC162061 и адаптером AC164110
- Explorer 16. обратите внимание, что на Explorer 16 неправильно помечены выводы для подключения PIKit2 (pin 1 на Explorer 16 соответствует выводу pin 6).

4.3 Подключение PICkit 2

Процедура подключения PICkit 2 описана в пункте 2.2 «Подключение PICkit 2»

Замечание: Debug Express дополнительно требует подтягивающие к земле резисторы по 4,7КОм на линиях ICSPCLK и ICSPDAT. Последние версии программаторов/отладчиков PICKit 2 имеют Красную кнопку и уже встроенные подтягивающие резисторы. На старых программаторах PICkit 2 эта кнопка черного цвета, и необходимо подключение соответствующих подтягивающих резисторов на плате.

Установите последнюю версию MPLAB IDE с прилагаемого в комплекте CD-диска или скачайте бесплатно с сайта [Microchip](#).

Замечание: Debug Express требует версии MPLAB IDE не ниже 7.50

4.4. Использование PICkit 2 Debug Express

4.4.1 Список поддерживаемых контроллеров

Полный список устройств, поддерживаемых PICkit 2 Debug Express, можно посмотреть в файле «Readme for PICkit 2.htm» в разделе «Readmes» директории «MPLAB IDE installation».

При выборе устройства (см. пункт 4.5. «Инструкции по применению Debug Express»), в окне «Select Device»(рис. 4-11) в разделе «Debuggers» цветом указана степень поддержки того или иного устройства:

- Красный цвет – устройство не поддерживается в настоящее время PICkit 2 Debug Express
- Желтый цвет – устройство имеет тестовую поддержку
- Зеленый цвет – полная поддержка устройства

Тестовая поддержка означает, что устройство поддерживается, но пока не прошло сертификационных тестов Microchip.

4.4.2 Зарезервированная область памяти

PICkit 2 Debug Express использует некоторые из ресурсов микроконтроллера во время отладки. Также он задействует память программ и ОЗУ во время отладки. Эти области памяти недоступны для пользователя. В MPLAB IDE зарезервированные области памяти регистров отмечаются литерой «R».

Более подробная информация об областях памяти, необходимых для внутрисхемной отладки, можно посмотреть в разделе MPLAB IDE:

Help->Topics. Информация об зарезервированных областях памяти в разделе «Resources Used By MPLAB ICD 2».

4.4.3 Использование отладочного модуля

Все контроллеры базового семейства и некоторые контроллеры среднего семейства требуют специального отладочного ICD модуля для осуществления внутрисхемной отладки. Список соответствующих каждому контроллеру отладочных модулей можно посмотреть в документе «Header Board Specification» (DS51292), на диске PICkit 2, идущем в комплекте, или на сайте www.microchip.com.

На плате отладочного модуля устанавливается специальный отладочный кристалл, аналогичный эмулируемому. На большинстве отладочных модулей расположен разъем RJ-11, используемый при отладке, и требующий дополнительный адаптер AC164110 с ICSP разъема на RJ-11 разъем. На рис. 4-1 показан пример подключения отладочного модуля AC162061 для PIC16F690 к плате DM164120-1 и использование адаптера AC164110.

Рис.4-1. Подключение PICkit2 к отладочному модулю.



Большинство контроллеров среднего семейства, семейства PIC18 и 16-разрядных PIC контроллеров не требуют отладочного модуля и могут отлаживаться напрямую внутрисхемно с помощью ICSP выводов. Например, PIC16F887, имеющийся на демонстрационной плате, входящей в комплект, может отлаживаться напрямую, простым подключением PICkit 2 (рис.4-2):

Рис.4-2. Подключение PICkit2 к отладочной плате.



PIC микроконтроллеры, которые имеют модуль внутрисхемной отладки, могут отлаживаться внутрисхемно через разъем для внутрисхемного программирования.

Показана плата DM164120-2 с микроконтроллером PIC16F887.

Отлаживаемая плата

4.4.4 Конфигурационные биты

PIC контроллеры, которые могут отлаживаться напрямую, без использования отладочного модуля, содержат так называемый DEBUG бит в слове (словах) конфигурации, запрещающий или разрешающий отладку.

Этот бит устанавливается автоматически MPLAB IDE, при использовании PICkit2 Debug Express, и не должен выставляться программно в исходном коде.

Внимание: бит /DEBUG НЕЛЬЗЯ устанавливать программно в конфигурационных настройках. Это может привести к тому, что данный бит будет выставлен неверно в момент программирования, что в свою очередь приведет к неправильному функционированию контроллера в Вашем приложении.

Большинство 16-разрядных PIC контроллеров семейства PIC24 и dsPIC33 имеют выводы для внутрисхемного программирования и отладки PGC1/EMUC1 и PGD1/EMUD1, PGC2/EMUC2 и PGD2/EMUD2 и т.д. Для программирования может быть выбран любой из портов ICSP, в то время как для отладки только один порт. Активный EMU порт задается в конфигурационных битах конкретного контроллера. Если EMU порт, к которому подключен PICkit 2, не задан, отладка будет недоступна. В диалоговом окне MPLAB IDE Configuration Bits соответствующий порт выбирается битами «Comm Channel Select».

4.4.5 Точки останова

Число точек останова, поддерживаемых PICKit 2 Debug Express, зависит от контроллера. Большинство контроллеров базового и среднего семейства поддерживают одну точку останова, некоторые контроллеры семейства PIC18 и 16-разрядные контроллеры поддерживают более одной точки.

Число точек останова для конкретного контроллера можно посмотреть в MPLAB IDE в разделе *Debugger>Breakpoints*. В диалоговом окне (рис.4-3) можно посмотреть число выставленных активных точек останова. Окно «Active Breakpoint Limit» показывает максимально возможное число точек останова для конкретного MCU. Окно «Active Breakpoint Limit» показывает сколько точек останова не использовано.

Рис.4-3.



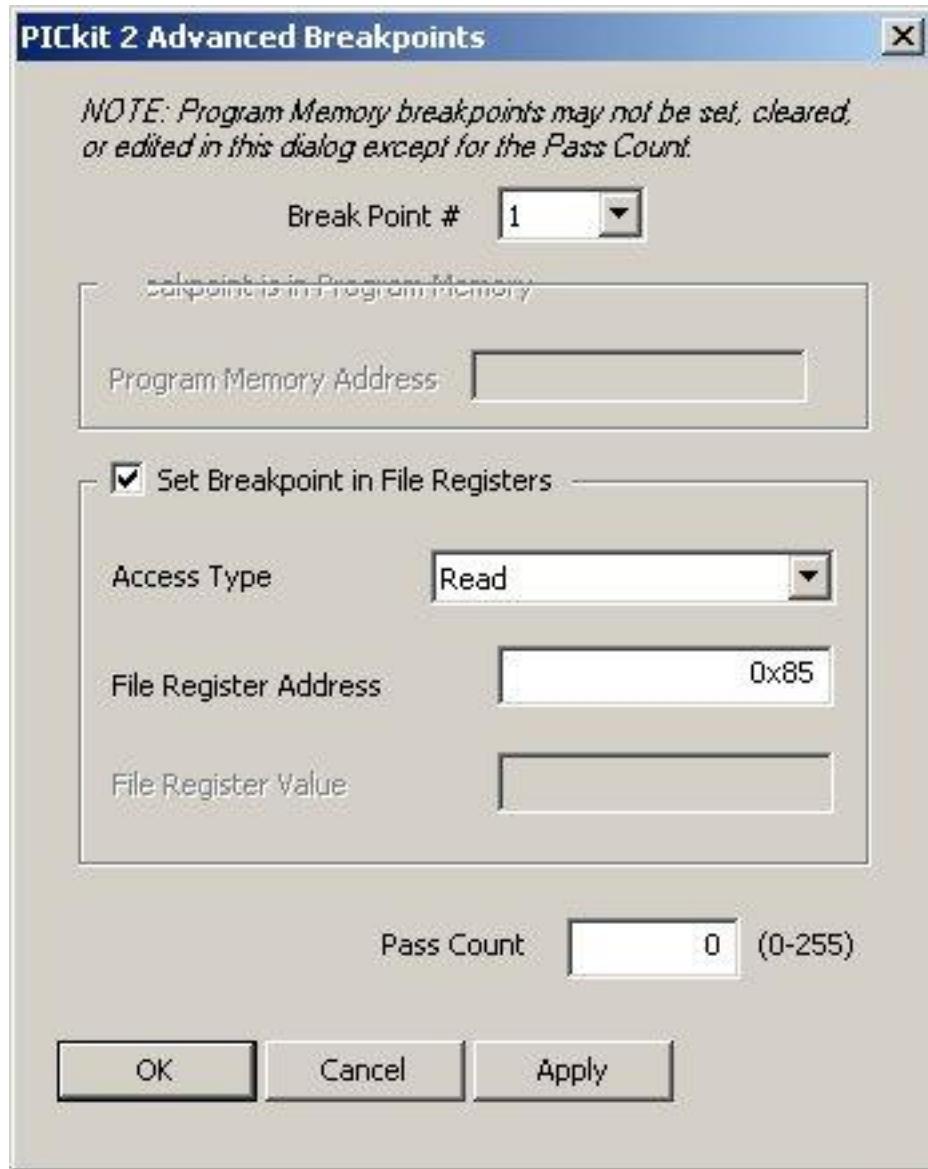
Некоторые PIC18 и 16-разрядные контроллеры также поддерживают расширенные точки останова. Расширенные точки позволяют выставлять точки останова в памяти ОЗУ, и приводят к останову программы, по факту чтения/записи в ОЗУ.

Счетчик событий (число событий до останова программы) выставляется в окне «Pass Count». Значение по умолчанию для счетчика событий равно «0», что означает останов программы при первой точке останова.

Если контроллер поддерживает расширенные точки останова, в MPLAB IDE будет доступно меню *Debugger>Advanced Breakpoints*. Если контроллер не поддерживает расширенные точки останова, это меню будет недоступно или отсутствовать.

Номер расширенной точки останова задается в меню «Break Point #» (рис.4-4).

Рис.4-4



Замечание: в диалоговом окне «Advanced Breakpoint» отображаются все расширенные точки останова, выставленные в памяти программ. Однако в данном окне нельзя выставлять/сбрасывать соответствующие точки останова, а только счетчик событий по каждой точке.
Для выставления, редактирования, очистки точек останова используйте меню *Debugger>Breakpoints* MPLAB IDE.

4.4.6 Проскальзывание

При внутрисхемной отладке РС микроконтроллеров, выполнение программы будет остановлено на инструкции, следующей непосредственно за точкой останова, а команда, на которой была выставлена точка останова будет выполнена. Это свойство называют «проскальзыванием».

Важно иметь ввиду наличие свойства проскальзывания при выставления точек останова в Вашей программе. Когда точка останова установлена на инструкции GOTO, CALL или RETURN, отладчик остановится на инструкции, на которые указывают

соответствующие команды перехода. В случае если в программе имеются две подряд следующие друг за другом инструкции CALL и прерывание установлено на первой, отлачик остановится на инструкции, на которую указывает второй CALL. Во избежание таких ситуаций, хорошим тоном считается размещение команды NOP между командами CALL, расположенными рядом.

Важно! В 16-разрядные контроллерах после прерывания будут выполнены 2 следующие инструкции.

4.4.7 Скрипты линкера

Если в Вашем проекте используются скрипты линкера, для внутрисхемной отладки вместо стандартного файла линкера необходимо использовать специальные ICD скрипты линкера, которые резервируют ресурсы, необходимые PICkit2 Debug Express для отладки.

Каждый контроллер имеет свой линкер файл, который обозначается с помощью «i» в конце имени файла.

Например:

16F877i.lkr – линкер файл внутрисхемной отладки для PIC16F877

18F4520i.lkr – линкер файл внутрисхемной отладки для PIC18F4520

4.5. Руководство пользователя Debug Express

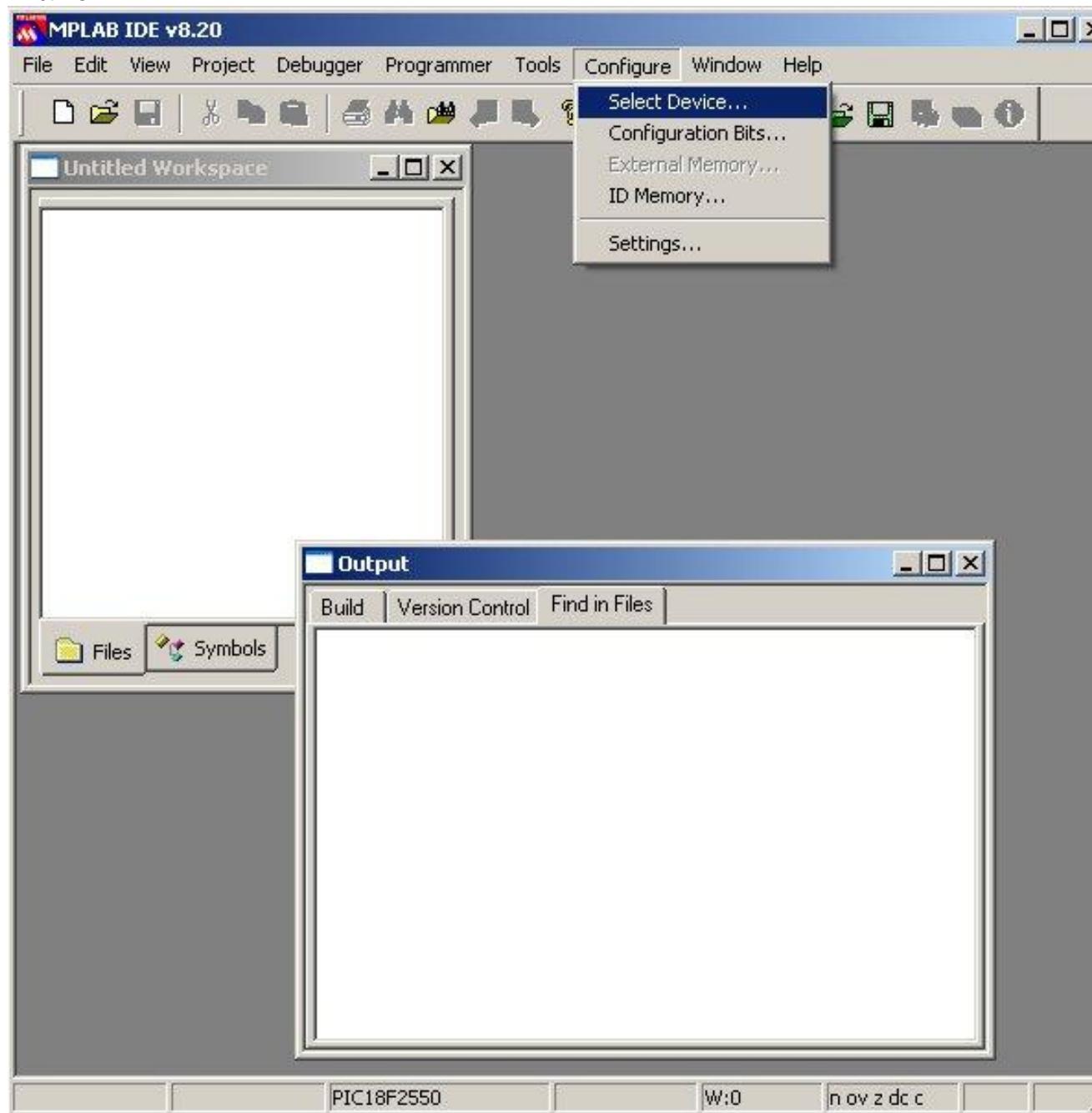
В качестве примера в этом руководстве описывается работа с демонстрационной платой DM164120-2 с контроллером PIC16F887 на борту, идущей в комплекте PICkit 2 Debug Express.

4.5.1 Выбор контроллера

Для выбора контроллера в MPLAB IDE:

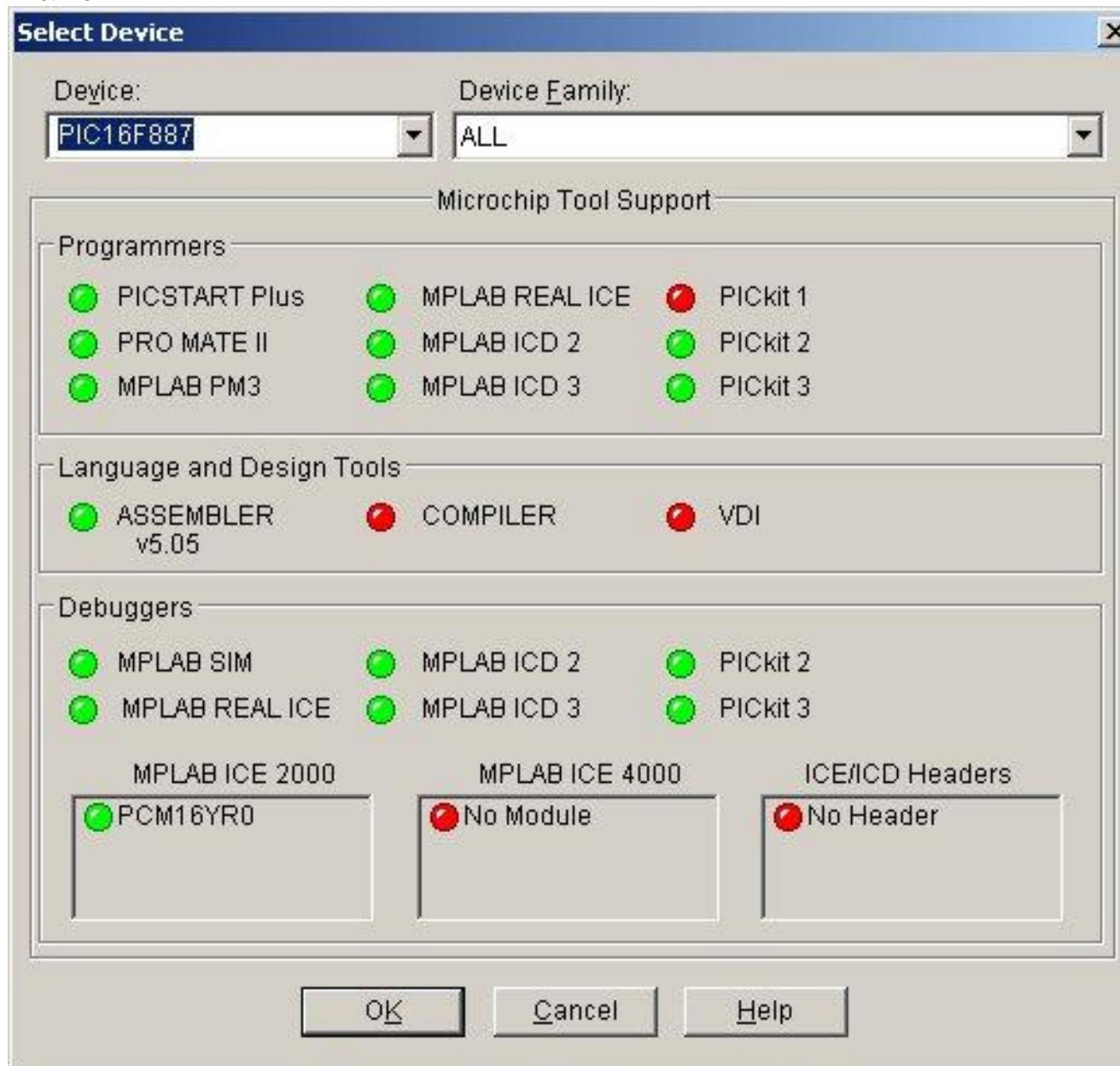
1. Запустите MPLAB IDE
2. Войдите в меню *Configure>Select Device* (рис. 4-5)

Рис.4-5



3. В окне *Select Device* в ниспадающем меню *Device* (рис. 4-6) выберите контроллер, с которым работаете. В данном примере это PIC16F887.

Рис.4-6

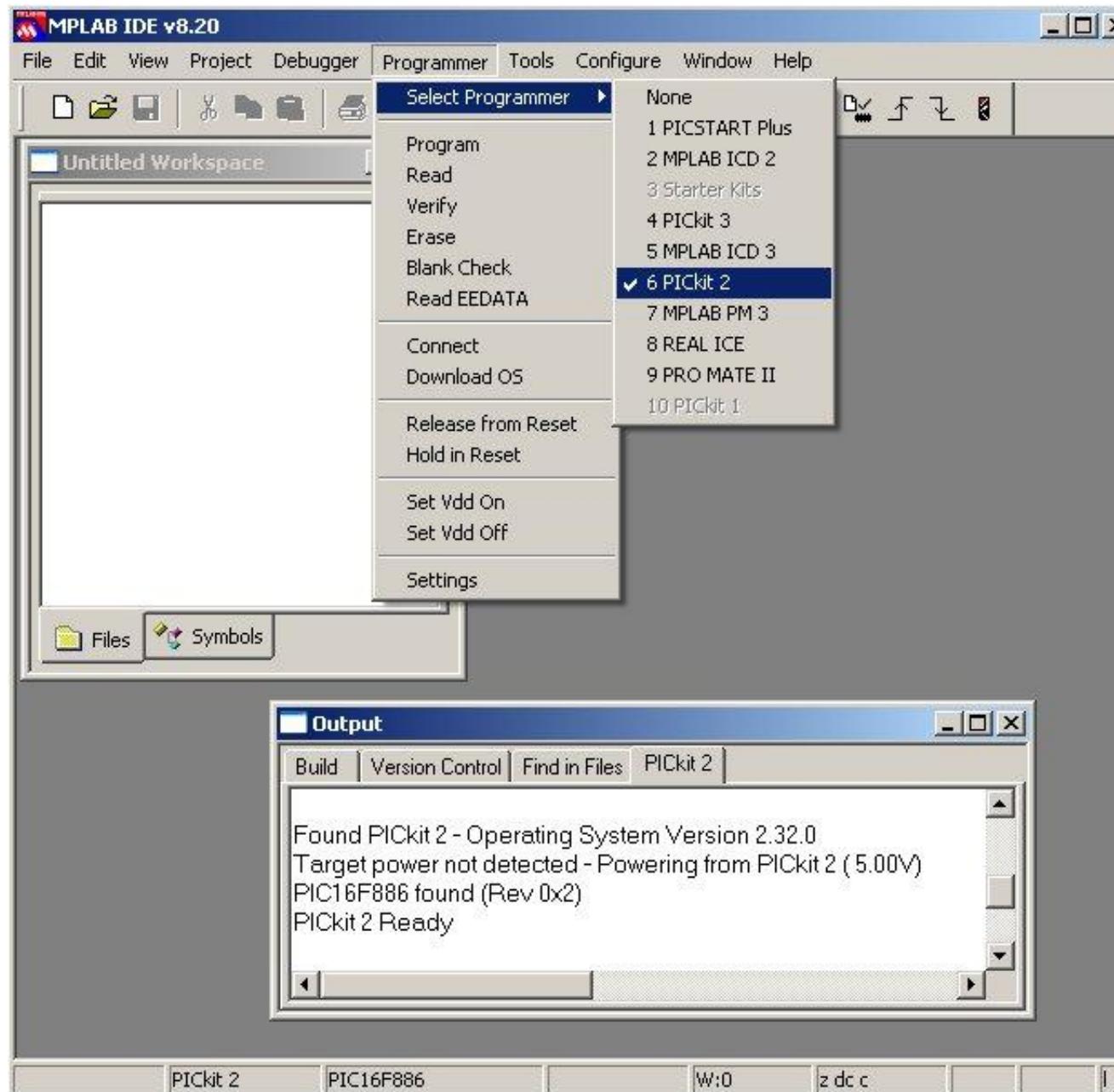


4. Нажмите «OK»

4.5.2 Выбор PICkit 2, в качестве средства отладки

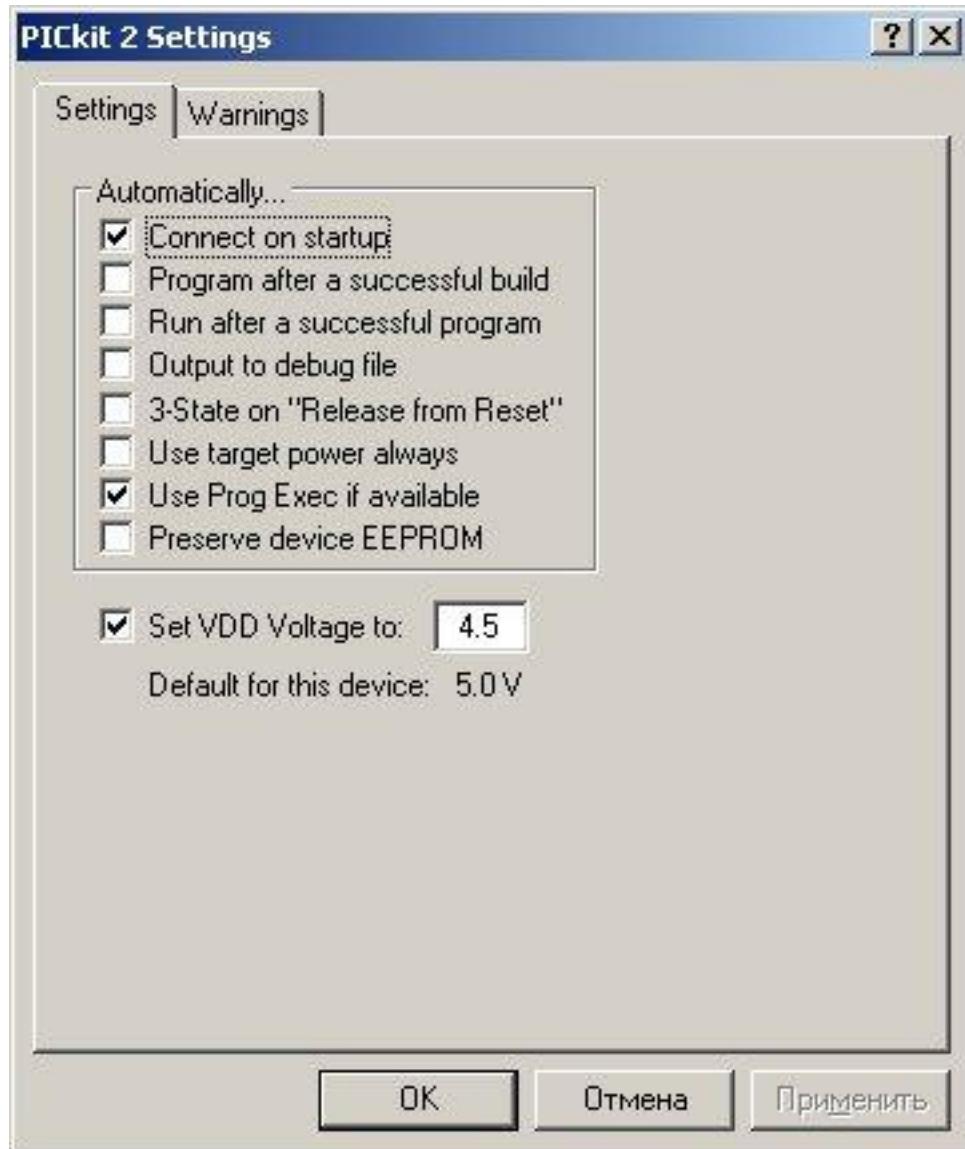
1. В меню *Debugger>Select Tool>PICkit 2* выберете PICkit2 в качестве отладочного средства (рис.4-7), в строке состояния и меню появятся дополнительные пункты и информация в соответствие с выбранным отладчиком.

Рис.4-7



2. В меню *Debugger>Settings* поставьте галочку напротив *Connect on Startup* - активирование функции автоподключения (рис.4-8)

Рис.4-8



3. Если PIKit 2 не определяется автоматически после его выбора в качестве отладочного средства, выберите *Connect now* в меню *Debugger>Connect*. Статус подключения будет отображаться в окне *Output window*.

4.5.3 Создание нового проекта в MPLAB IDE

Для создания нового проекта в MPLAB IDE используйте Project Wizard.

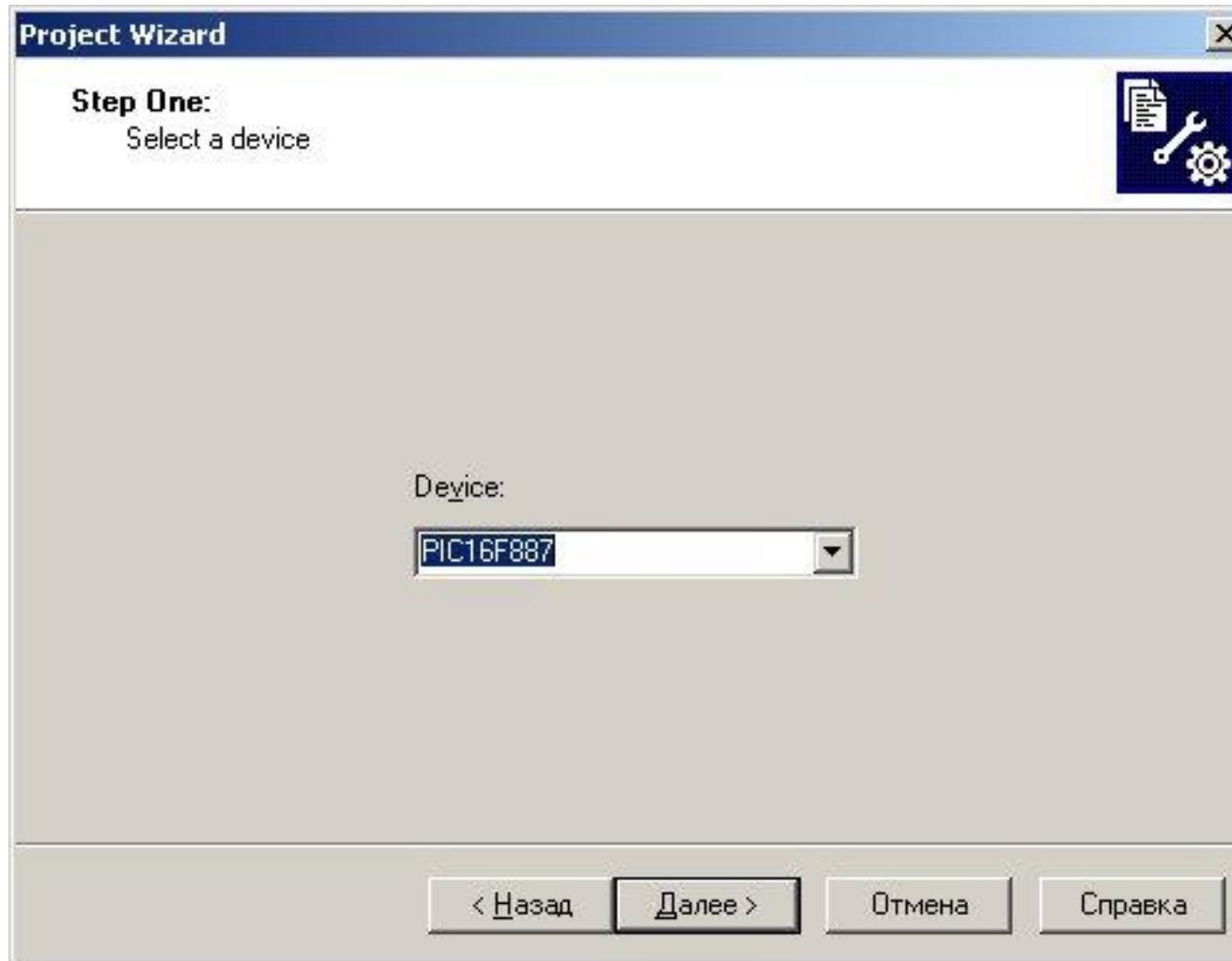
1. Выберите *Project>Project Wizard* для создания нового проекта. В появившемся окне *Project Wizard* нажмите *Next*.

Рис.4-9



2. Выберете контроллер в ниспадающем списке (рис.4-10). В нашем примере: PIC16F887.

Рис.4-10



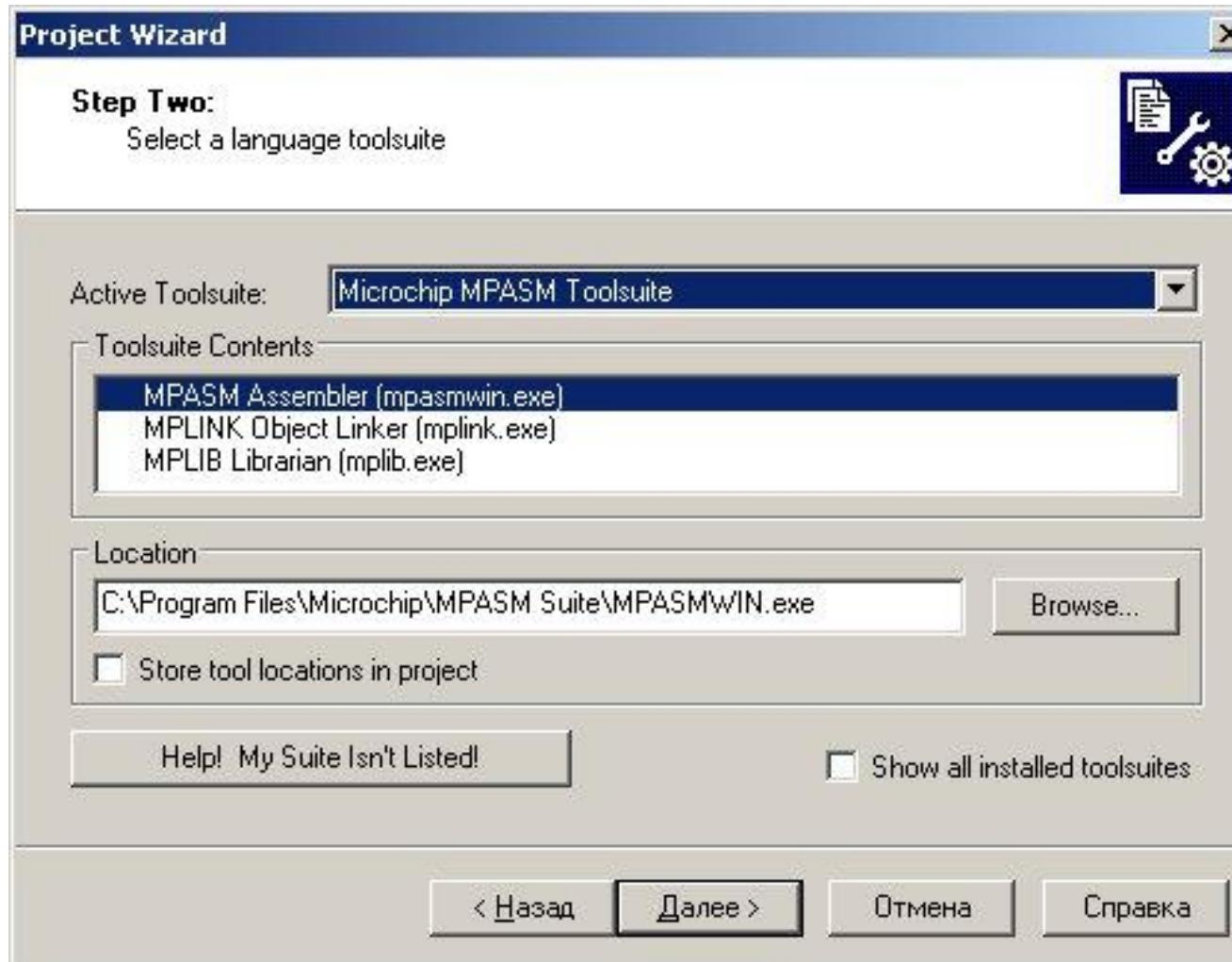
3. В данном проекте будем использовать MPASM ассемблер (рис.4-11).

Убедитесь в правильности указания пути по умолчанию:

- для MPASM ассемблера путь должен быть указан к файлу mpasmwin.exe
- для MPLINK линкера путь должен быть указан к файлу mplink.exe
- для MPLIB библиотеки путь должен быть указан к файлу mplib.exe

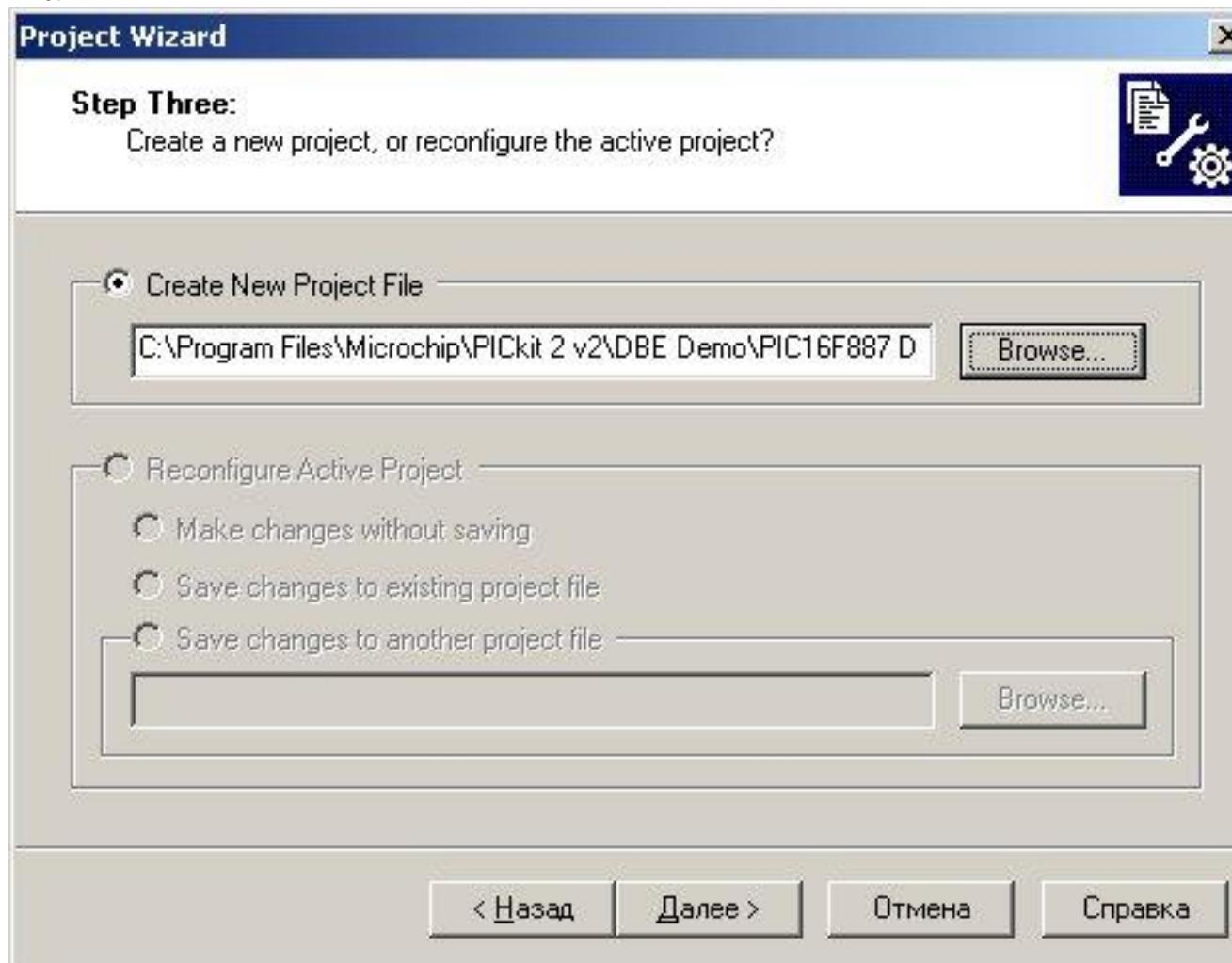
Нажмите *Next*

Рис.4-11



4. Укажите путь для вновь создаваемого проекта и назовите его (рис.4-12). В нашем случае: *C:\Program Files\Microchip\PICkit 2 v2\DBE*, название *PIC16F887 Debug Demo*.

Рис.4-12



5. Добавьте Файл Вашего проекта (рис.4-13).

Замечание: в другие файлы можно будет добавить позже.

Литера «A» означает, что MPLAB IDE может определять, должен ли быть путь абсолютным или косвенным к файлу проекта. Более подробная информация об этом в разделе помощи к MPLAB IDE.

Нажмите *Next*

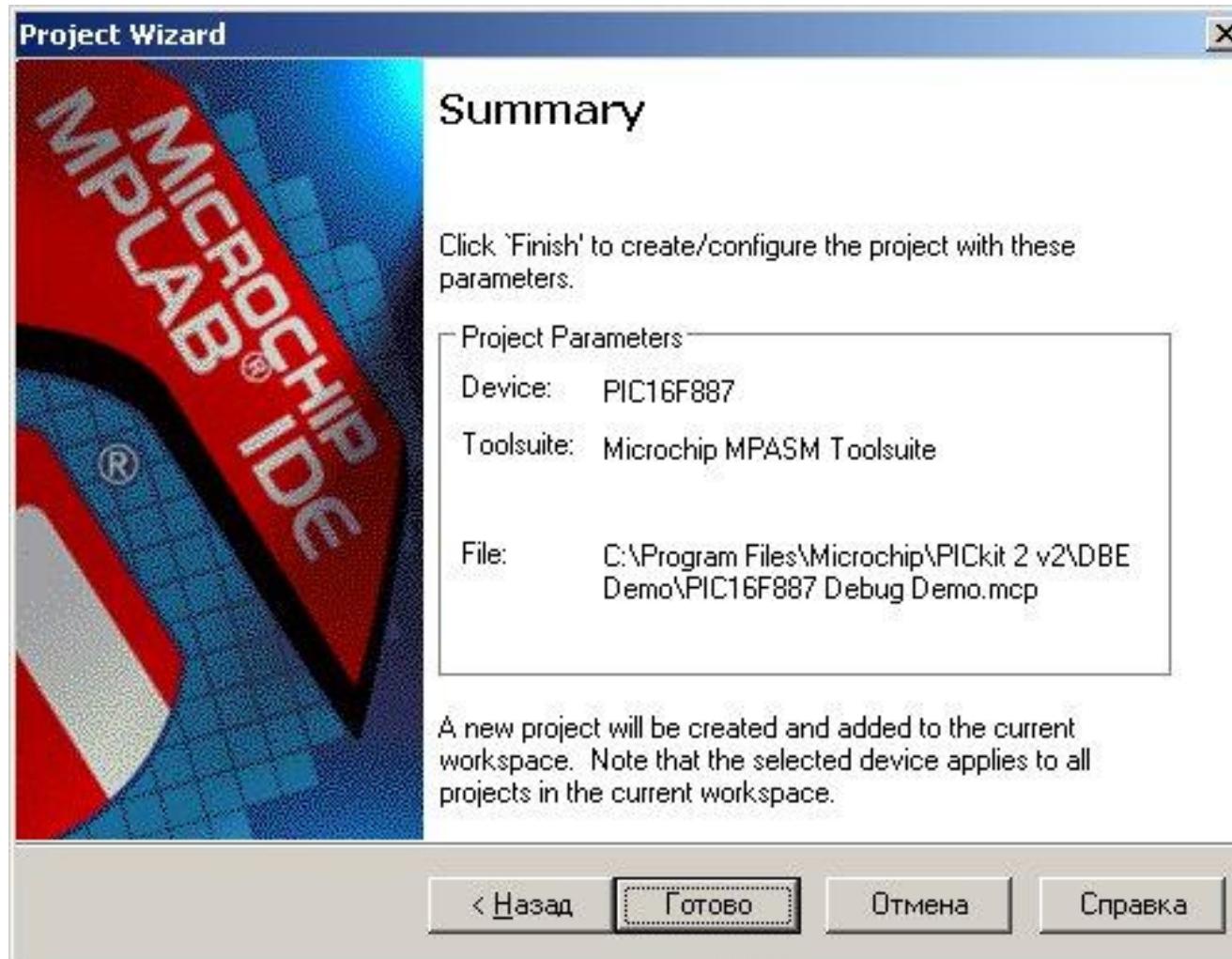
Замечание: для проектов, содержащих более одного файла необходимо добавить еще и файл скрипта линкера.

Рис.4-13.



6. Если Вы все указали правильно нажмите *Finish*, иначе можно вернуться к предыдущим шагам по созданию проекта *Back*. (рис.4-14).

Рис.4-14

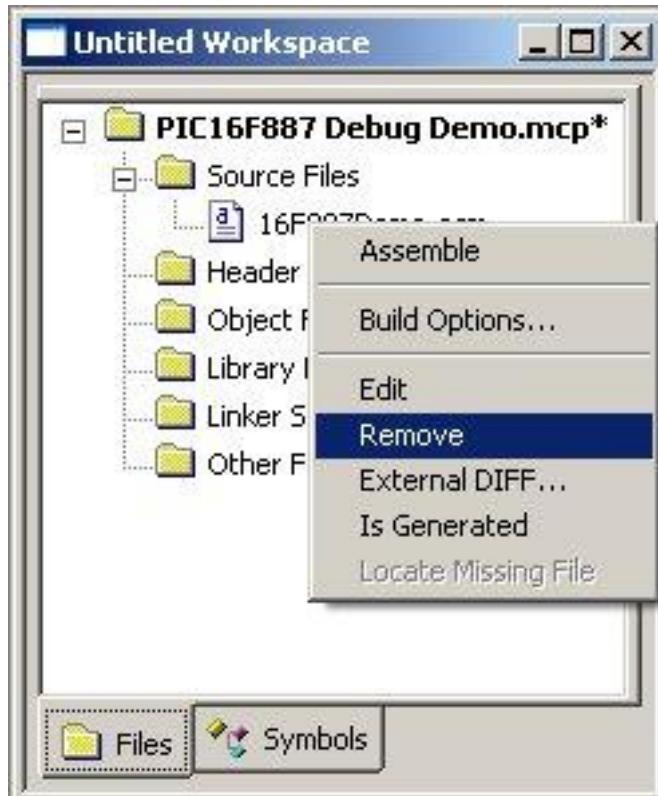


4.5.4 Просмотр проекта

После создания проекта в рабочей области MPLAB IDE появится окно проекта *PROJECT WINDOW*. (рис 4-15). Если оно не открыто, можно открыть его с помощью *View>Project*.

С помощью этого окна можно добавлять или удалять файлы проекта (правая кнопка мыши).

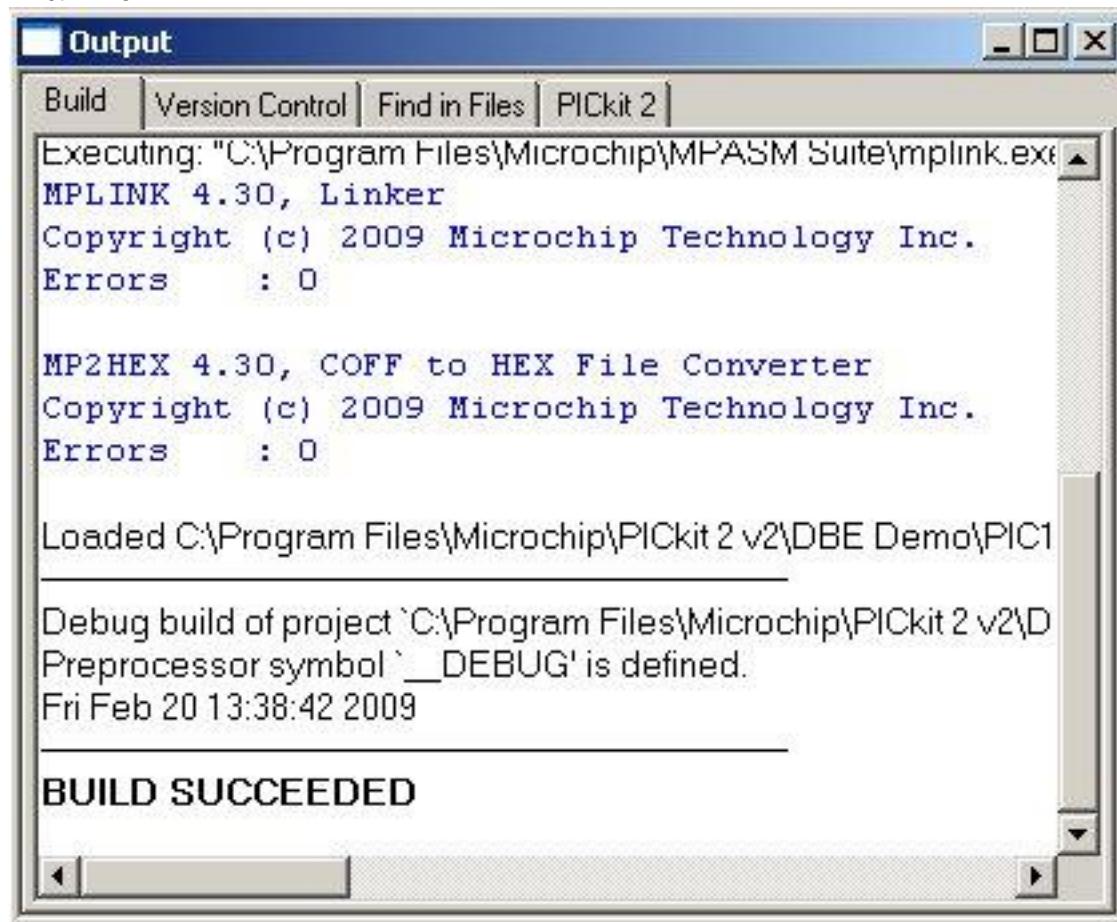
Рис.4-15



4.5.5 Создание hex-файла

Чтобы запрограммировать контроллер необходимо скомпилировать проект и получить hex-файл. Для этого выберите в меню *Project>Build All* или *Build All* во всплывающем меню при нажатии правой клавишей мыши по иконке проекта. MPASM-ассемблер создаст hex-файл с тем же названием, что и исходный asm-файл. В окне *Output* на вкладке *Build* можно просмотреть текущее действие, выполняемое ассемблером.

Рис. 4-16



4.5.6 Проверка значений битов конфигурации

Биты конфигурации запрограммированного контроллера устанавливаются в соответствие с директивами `_CONFIG` программы. После компиляции проекта их значения можно просмотреть в окне *Configure>Configuration Bits*.

Рис. 4-17

The screenshot shows the 'Configuration Bits' window. The window title is 'Configuration Bits'. A checked checkbox says 'Configuration Bits set in code.' Below is a table of configuration bits:

Address	Value	Category	Setting
2007	20E4	Oscillator	Internal RC No Clock
		Watchdog Timer	Off
		Power Up Timer	On
		Master Clear Enable	/MCLR is external
		Code Protect	Off
		Data EE Read Protect	Off
		Brown Out Detect	BOD and SBORN disabled
		Internal External Switch Over	MC Disabled
		Monitor Clock Fail-safe	Disabled
		Low Voltage Program	Disabled
2008	3EFF	Brown Out Reset Sel Bit	Brown out at 2.1V
		Self Write Enable	No protection

Для выполнения ознакомительной работы с набором *PICkit2 Debug Express* следует установить следующие биты конфигурации:

Config1:

- Oscillator . Internal RC No Clock
- Watchdog Timer . Off
- Power-Up Timer . On
- Master Clear Enable . MCLR is external
- Code-Protect . Off
- Data EE Protect . Off
- Brown-Out Detect . BOD and SBOREN Disabled
- Internal-External Switch Over Mode . Disabled
- Monitor Clock Fail-safe . Disabled
- Low-Voltage Program . Disabled

Config 2:

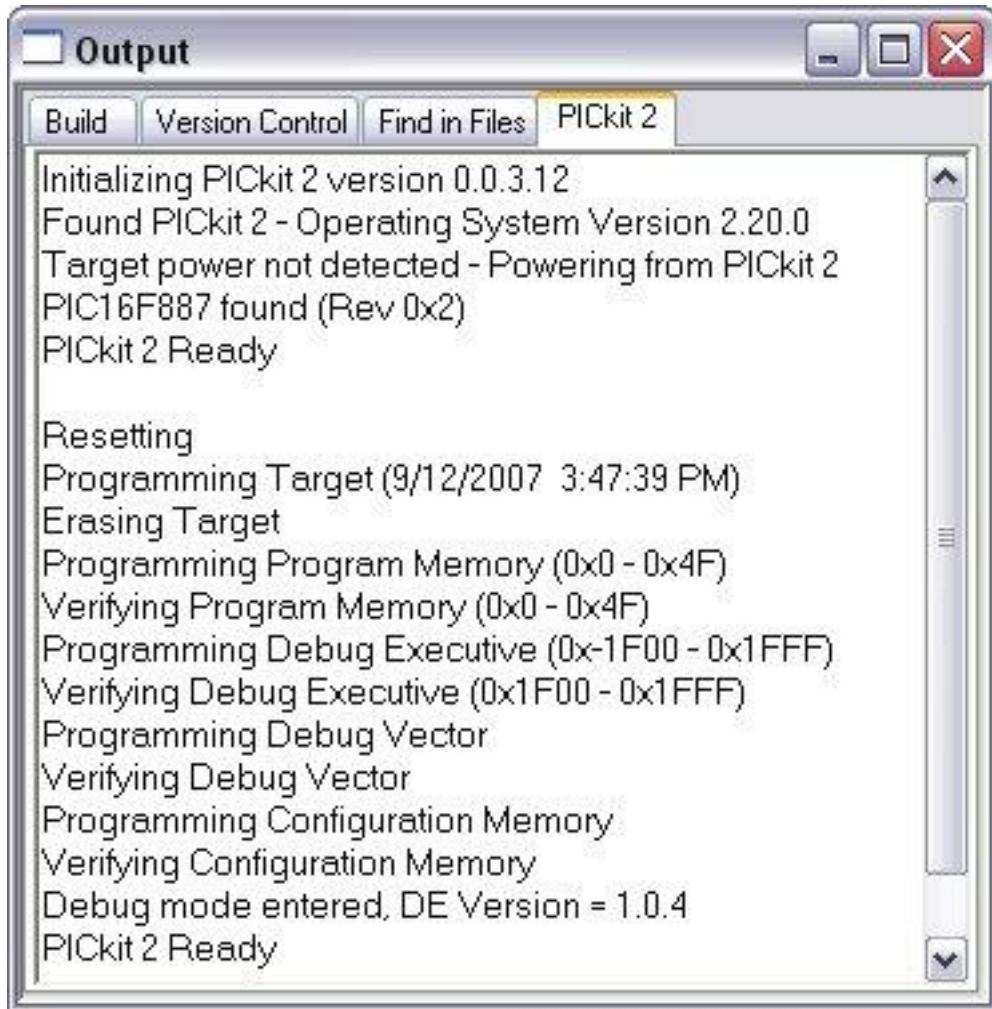
- Self Write Enable . No Protection
- Master Brown-out Reset Sel Bit . Brown-out at 2.1V

4.5.7 Загрузка кода программы для отладки

Чтобы запрограммировать микроконтроллер PIC16F887, установленный на плате 44-pin Demo Board, выберите *Debugger>Program*

Программирование займет несколько секунд. Во время программирования на вкладке *PICkit2* в окне *Output* отображается текущее выполняемое действие. По окончании программирования, в диалоговое окно примет вид схожий с Рис. 4-18

Рис. 4-18



4.5.8 Начало отладки демонстрационного проекта на базе PIC16F887

Выполнение кода программы возможно в режиме реального времени (*Run*) и пошагово (*Step Into*, *Step Over*, *Step Out*, *Animate*). Выполнение кода в реальном времени начинается при выборе *Run*, останов происходит при нажатии *Halt*, либо по достижении установленной точки останова. После этого можно начать пошаговое исполнение. Для быстрого доступа к вышеперечисленным операциям отладки соответствующие кнопки вынесены в отдельную панель инструментов MPLAB IDE.

Debugger Menu	Toolbar Buttons
Run	
Halt	
Animate	
Step Into	
Step Over	
Step Out	
Reset	

Для начала выполнения кода демонстрационного проекта:

1. Откройте файл 16F887Demo.asm, выполнив двойной щелчок на его иконке в окне проекта, либо выберите файл через диалог *File>Open*
2. Выберите *Debugger>Run* или нажмите кнопку *Run* на панели инструментов
3. Покрутите потенциометр RP1 на демоплате и наблюдайте за светодиодной индикацией

Если все предыдущие действия были выполнены правильно, то частота включения/выключения светодиодов будет зависеть от положения потенциометра. Для демонстрации возможностей отладки в исходном коде преднамеренно допущена ошибка, но об этом в следующем пункте 4.5.9.

4. Для остановки выполнения программы выберите *Debugger>Halt*, либо щелкните *Halt* на панели инструментов
5. Для сброса контроллера нажмите *Debugger>Reset>Processor Reset*. При этом стрелка-указатель исполняемой строки исчезнет.

4.5.9 Отладка демонстрационного проекта PIC16F887

Корректной работе препятствуют следующие ошибки:

1. Значения АЦП не записываются должным образом
2. АЦП не разрешен или не готов к преобразованию
3. Опечатка в исходном коде

Для выявления первой ошибки установите точку останова в месте записи старшего байта результата АЦП:

1. Установите указатель мыши на строчку *movwf Delay+1* файла 16F877Demo.asm как показано на Рис. 4-19.
Выполнение программы будет приостановлено, когда АЦП завершит преобразование.
2. Установите точку останова, выполнив двойной щелчок на нужной строке кода программы, либо выбрав *Set Breakpoint* во всплывающем меню по нажатию правой кнопки мыши. Индикатором установленной точки останова является буква «В» в красном восьмиугольнике напротив выбранной строки (Рис. 4-19).

Рис. 4-19

```

MPLAB IDE Editor
16F887Demo.asm

banksel    Display
movlw      0x80
movwf      Display
clrf       Direction
clrf       LookingFor ; Looking for a 0 on the button
MainLoop:
    movf      Display,w ; Copy the display to the LEDs
    movwf      PORTD
    nop        ; wait total of 5uS for A2D amp to settle and capacitors to charge
    nop        ; wait 1uS
    nop        ; wait 1uS
    nop        ; wait 1uS
    nop        ; wait 1uS
    bsf       ADCONO,GO_DONE ; start conversion
    btfsc     ADCONO,GO_DONE ; this bit will change to zero when the conversion is complete
    goto     $-1
    movf      ADRESH,w ; Copy the A2D result to the delay loop
    movwf      Delay+1

A2DDelayLoop:
    incfsz   Delay,f ; Waste time.
    goto     A2DDelayLoop ; The Inner loop takes 3 instructions per loop * 2!
    incfsz   Delay+1,f ; The outer loop takes an additional 3 instructions
    goto     A2DDelayLoop ; (768+3) * 256 = 197376 instructions / 1M instructions

```

3. Выберите Debugger>Run или щелкните Run для старта программы. Останов произойдет, когда программа исполнит код соответствующий строчке, на которой была установлена точка останова.

Замечание: В зависимости от команды и контроллера может иметь место эффект «проскальзывания» (skidding), т.е. останов происходит на несколько команд ниже команды на которой была поставлена точка останова:
Incfsz Delay,f

4. Наведите курсор мыши на *ADRESH* на строку выше точки останова и увидите значение регистра (Рис. 4-20)

Рис. 4-20

```

16F887Demo.asm

        bsf      ADCONO, GO_DONE ; start conversion
        btfsc   ADCONO, GO_DONE ; this bit will change to zero when the conversion
        goto    $-1
        movf    ADRESH, w       ; Copy the A2D result to the delay loop
        movwf   Delay+1
        ADRESH = 0x00

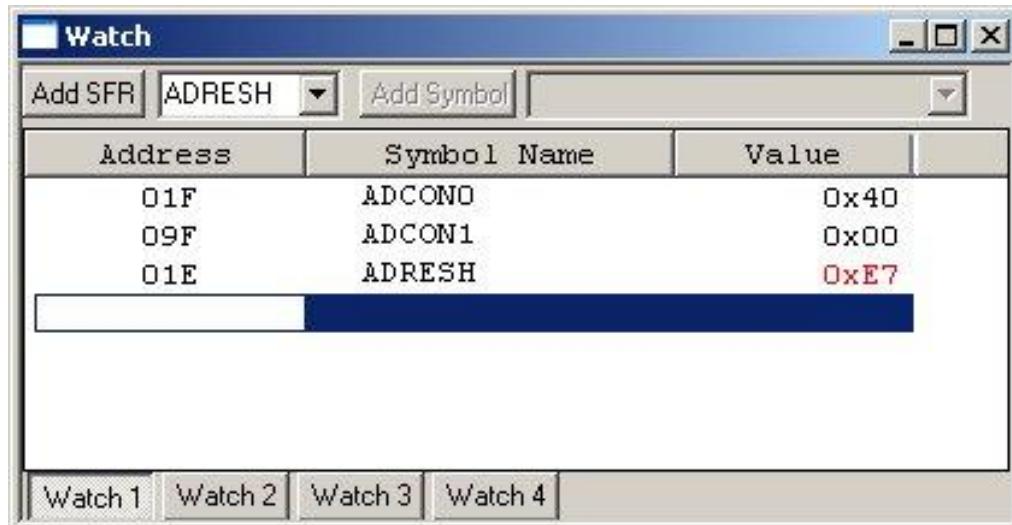
A2DDelayLoop:
        incfsz  Delay,f       ; Waste time.
        goto    A2DDelayLoop  ; The Inner loop takes 3 instructions per loop * 256 = 768
        incfsz  Delay+1,f     ; The outer loop takes and additional 3 instructions
        goto    A2DDelayLoop  ; (768+3) * 256 = 197376 instructions / 1M instructions = 197.376ms
        ; call it a two-tenths of a second.

        movlw   .13             ; Delay another 10mS plus whatever was above

```

- Измените положение потенциометра (RP1) и продолжите выполнение программы (*Debug>Run*). Программа прокрутится полный и цикл и остановится в том же месте.
- Проверьте значение *ADRESH* и убедитесь, что оно не изменилось. Это значит, что АЦП не работает. Инициализация и настройка АЦП производится в начале программы
- Выберите *Debugger>Reset* для сброса программы
- Выберите *View>Watch*, чтобы открыть окно Watch. Оно позволяет просматривать значения регистров в реальном режиме времени при выполнении кода программы. (Рис. 4-21)

Рис. 4-21



Замечание: При отладке PICkit2 Debug Express не рекомендуется использовать *View>File Registers* и *View>Special Function Registers*. Т.к. это приведет к необходимости обновления всех переменных (регистров) на каждой итерации обмена отладочной информацией с контроллером, что значительно снизит скорость отладки. В окне Watch следует указать только интересующие Вас переменные (регистры).

- В первом «выпадающем» списке выберите *ADCON0* и нажмите кнопку Add SFR, чтобы добавить регистр в Watch. Аналогично добавьте *ADCON1* и *ADRESH*. (Рис. 4-22)

Рис. 4-22

```

MPLAB IDE Editor
16F887Demo.asm

        movwf    ANSEL
        movlw    0x00          ; remaining pins are all digital
        movwf    ANSELH
SetupA2D:
        banksel  ADCON1      ; address Register Page 1
        movlw    0x00          ; A2D Left-Justified, references are VDD and VSS
        movwf    ADCON1
        banksel  ADCONO      ; address Register Page 0
        movlw    0x41          ; configure A2D for Fosc/8, Channel 0 (RA0), and t
InitRegisters:
        banksel  Display
        movlw    0x80
        movwf    Display

```

- Снова запустите исполнение программы (*Run*). Программа опять остановится в точке останова.
- Взгляните на значения регистров *ADCON0* и *ADCON1* в окне Watch. Значение *ADCON0* 0x40 (б'01000000'). Но это некорректное значение. Согласно даташиту на PIC16F882/883/884/886/887 (DS41291), младший бит должен быть установлен в 1 (б'01000001') для включения модуля АЦП. Для исправления данной ошибки замените:
movlw 0x40
на
movlw 0x41, как показано на Рис. 4-23

Рис. 4-23

Add SFR	ADRESH	Add Symbol
Address	Symbol Name	Value
01F	ADCON0	0x41
09F	ADCON1	0x00
01E	ADRESH	0xF7

Watch 1 Watch 2 Watch 3 Watch 4

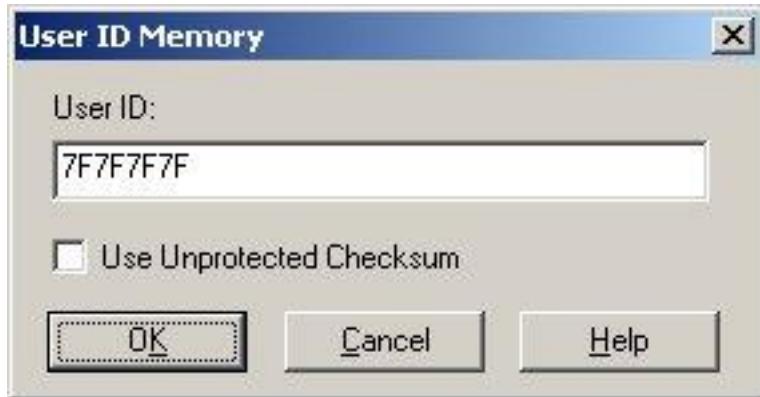
12. Выберите *File>Save*, чтобы сохранить изменения
13. Для того, чтобы изменения вступили в силу, необходимо перекомпилировать проект (*Project>Build All*) и перепрограммировать контроллер (*Debugger>Program*).
14. Снова запустите исполнение программы (*Run*). Программа опять остановится в точке останова.
15. Взгляните на значения регистров в окне Watch. Значение *ADCON0* 0x41 ('01000001') (Рис. 4-24)
16. Снимите установленную ранее точку останова (двойной щелчок мышью на соответствующей строке)
17. Снова запустите исполнение программы (*Run*). Теперь изменение положения потенциометра отображается светодиодами. Программа работает корректно! Данная программа содержала всего одну ошибку, реальный же код может содержать гораздо больше. Отладочные возможности PICkit2 помогут Вам быстро обнаружить и исправить все ошибки.

4.5.10 Программирование отлаженной программой

Когда программа проверена и работает верно, необходимо запрограммировать контроллер для автономной работы в конечном устройстве. На данной стадии ресурсы зарезервированные для работы ICD доступны для использования программой. Для программирования выполните следующие действия:

1. Отключите PICkit2 как отладчик – *Debugger>Select Tool>None*
2. Выберите PICkit2 в качестве программатора – *Programmer>Select Programmer*.
3. Дополнительно можете установить *ID Configure>ID Memory* (Рис. 4-24)

Рис. 4-24



4. Установите параметры программирования *Programmer>Settings* на вкладке *Program*
5. Выберите *Programmer>Program*

Глава 5. Решение возникающих проблем

5.1 Введение

Эта глава содержит вопросы и ответы на наиболее часто встречающиеся проблемы связанные с использованием программатора-отладчика PICkit 2. Так же дается информация по ошибкам возникающим в среде разработки MPLAB IDE.

5.2. Часто задаваемые вопросы

Устройство не опознано

Вопрос:

Почему я получаю сообщение «устройство не найдено» (No Device Found)?

Ответ:

Проверьте что используемое устройство поддерживается и что микроконтроллер подключен к PICkit 2 в соответствии с главой 3 «Использование внутрисхемного программирования.»

Проверьте что контроллеры PIC18FXXJXX, PIC24X и dsPIC33F имеют конденсатор надлежащей емкости на выводе VDDCORE/VCAP в соответствии с документацией на используемый микроконтроллер.

Проверьте что подключенное устройство соответствует выбранному семейству (индцируется в окне Статуса). При необходимости выберите правильное семейство в меню выбора семейства (Device Family).

См. также вопрос по ошибкам, возникающим при программировании слова конфигурации.

Превышение лимита по току

Вопрос:

Почему я получаю сообщение об ошибке □ Превышение лимит Тока USB хаба (USB Hub Current Limit Exceeded) в Microsoft® Windows®?

Ответ:

Проверьте что ваша схема потребляет ток не более чем 25mA.

Драйвер Microsoft Windows

Вопрос:

После подключения PICkit 2 к USB порту Windows 98 спрашивает об установке драйвера?

Ответ:

PICkit 2 использует драйвер, имеющийся в Windows 98 SE. Когда Windows 98 SE запрашивает драйвер, выберите «Поиск наиболее подходящего драйвера» (Search for the best driver for your device) и нажмите Далее (Next). Выберите пункт Microsoft Windows Update и нажмите Далее (Next). Windows автоматически установит подходящий драйвер. Не используйте драйвер для MPLAB ICD 2 USB.

Проверка (Verify) и Чтение (Read) возвращает одни нули

Вопрос:

Почему при выборе Чтения или Проверки кода окно Памяти Программ содержит одни нули? Что я делаю неправильно?

Ответ:

Скорее всего считываемое устройство имеет установленные биты защиты кода. Убедитесь что в битах конфигурации не указано защищать память программ.

Microsoft Windows 95/98/NT

Вопрос:

Могу я использовать PICkit2 под Windows 95/98/NT?

Ответ:

Нет. Эти операционные системы не поддерживают USB в нужном объеме или не имеют совместимых драйверов.

Ошибки напряжений VDD и/или VPP (VDD/VPP Errors)

Вопрос:

Почему я получаю сообщения VDD Error или VPP Error ?

Ответ:

Эти ошибки показывают, что PICkit 2 не может выставить напряжения VDD или VPP. Проверьте вашу схему на короткие замыкания, на потребление тока (допускается не более 25mA) и проверьте что контроллер подключен к PICkit 2 в соответствии с главой 3 «Использование внутрисхемного программирования.» Убедитесь что конденсатор по питанию в схеме не задерживает установление напряжение питания VDD на время дольше чем 500 мксек.

Ошибка программирования

Вопрос:

Почему я могу запрограммировать некоторые микроконтроллеры, а другие нет?

Ответ:

Некоторые микроконтроллеры могут быть сконфигурированы для использования с Низковольтным Программированием (Low-Voltage Programming), висящий в «воздухе» вывод PGM может ловить помехи. Используйте подтягивающий к земле резистор, подключенный к выводу PGM.

Некоторые контроллеры Среднего семейства, такие как семейства PIC16F72/73/74/76/77 и PIC16F737/747/767/777 при программировании требуют напряжение питания не ниже чем +4,75V. Напряжение USB порта может варьироваться в зависимости от компьютера, и возможно что PICkit 2 не может обеспечить напряжение питания +4,75V. В этом случае программируйте такие контроллеры с использованием внешнего источника на 5,0V.

Некоторые контроллеры PIC18F требуют увеличить величину блокировочного конденсатора по цепи питания до 10мкФ.

PIC18FXXJXX, PIC24X и dsPIC30F/33F требуют 4.7мкФ на выводе VDDCORE/VCAP для правильной работоспособности. Если используется внутренний регулятор напряжения для VDDCORE, убедитесь что вывод ENVREG подключен к напряжению питания VDD.

Ошибка Windows: Неопознанное устройство USB (Unrecognized USB Device)

Вопрос:

Почему я получаю ошибку «Неизвестное устройство» при подключении PICkit 2 к USB?

Ответ:

Эта ошибка может возникать если PICkit 2 подключается к USB вместе с подключенными отлаживаемой платой. Когда подключаете PICkit 2 к компьютеру или перезагружаете компьютер, убедитесь что к PICkit 2 не подключена программируемая или отлаживаемая плата. Подобная ошибка так же может возникать при подключении PICkit 2 к некоторым USB хабам. Попробуйте подключить PICkit 2 напрямую к USB порту компьютера.

Ошибка «Не найден PICkit2» (PICkit 2 Not Found)

Вопрос:

Я подключил мой PICkit 2 к USB, но программное обеспечение PICkit 2 Programmer сообщает что PICkit 2 не обнаружен. В чем дело?

Ответ:

Пожалуйста ознакомьтесь с ответом на вопрос «Ошибка Windows: Неопознанное устройство USB».

Программное обеспечение PICkit 2 Programmer не отвечает на команды управления.

Вопрос:

Почему программное обеспечение PICkit 2 Programmer заблокировалось?

Ответ:

Скорее всего программа работает. Во время выполнения процедуры программирования интерфейс программного обеспечения PICkit 2 Programmer не активен. До тех пор пока программирование не закончится Windows OS может остановить обращение к ПО PICkit 2. После выполнения процедуры программирования Windows OS начнет обновление окна PICkit 2. Для контроллеров с большим объемом программирования может длиться несколько минут. Для 8-и разрядных контроллеров время ожидания может достигать до 2- минут, для 16-и разрядных – до 5-и.

Существует несколько чип-сет контроллеров USB которые могут вызывать подобные проблемы (обычно в ноутбуках). Данная «проблема» может решиться применением USB хаба. Рекомендуются хабы с внешним источником питания.