

Министерство образования Республики Беларусь
Учреждение образования
«Белорусский государственный университет
информатики и радиоэлектроники»

Кафедра радиотехнических систем

***РАЗРАБОТКА ПРОГРАММ
ДЛЯ МИКРОКОНТРОЛЛЕРОВ ФИРМЫ MICROCHIP
В ИНТЕГРИРОВАННОЙ СРЕДЕ MPLAB IDE 7***

Методические указания
к лабораторным работам по курсу
«Микропроцессорные устройства»
для студентов радиотехнических специальностей
всех форм обучения

Минск 2008

УДК 681.3.06(075.8)
ББК 32.973.26-018.2 я 73
Р 17

С о с т а в и т е л и :

А. А. Казека, А. В. Мартинович, И. Г. Давыдов

Разработка программ для микроконтроллеров фирмы MICROCHIP
Р 17 в интегрированной среде MPLAB IDE 7 : метод. указания к лаб. работам
по курсу «Микропроцессорные устройства» для студ. радиотехнич. спец.
всех форм обуч. / сост. А. А. Казека, А. В. Мартинович, И. Г. Давыдов. –
Минск : БГУИР, 2008. – 31 с. : ил.

Рассмотрены основные функции и порядок работы в интегрированной среде
MPLAB IDE 7, предназначенной для написания и отладки программ однокристалльных
микроконтроллеров PICmicro. Материал предназначен для студентов радиотехниче-
ских специальностей всех форм обучения.

УДК 681.3.06(075.8)
ББК 32.973.26-018.2 я 73

© Казека А. А., Мартинович А. В., Давыдов И. Г.,
составление, 2008
© УО «Белорусский государственный университет
информатики и радиоэлектроники», 2008

ВВЕДЕНИЕ

Интегрированная среда разработки MPLAB IDE 7 предназначена для написания и отладки программ однокристальных микроконтроллеров PICmicro, производимых компанией Microchip.

MPLAB IDE 7 поддерживает следующие функции:

- создание и редактирование исходных текстов программы;
- сборка (link) и компилирование (compile) исходных текстов программ различными компиляторами;
- отладка кода программы с использованием симулятора или эмулятора (требуется аппаратная часть – debugger).


MPLAB IDE состоит из нескольких модулей, обеспечивающих единую среду разработки:

- менеджер проекта MPLAB – используется для создания и работы с файлами, относящимися к проекту;
- редактор MPLAB – предназначен для написания и редактирования исходного текста программы, шаблонов и файлов сценария линкера;
- симулятор MPLAB-SIM – программный симулятор моделирует выполнение программы в микроконтроллере с учетом состояния портов ввода/вывода;
- ассемблер MPASM – компилирует исходный текст программы.

Дополнительная информация по MPASM представлена в приложении и литературе [1].

1 Начало работы с MPLAB IDE 7

1.1 Запуск среды проектирования

Запустить MPLAB можно через меню программ (*Пуск -> Программы -> Microchip->MPLAB IDE v7.xx->MPLAB IDE*) или через пиктограмму «» на рабочем столе.

После запуска отобразится основное *окно программы MPLAB IDE 7* (рисунок 1.1)

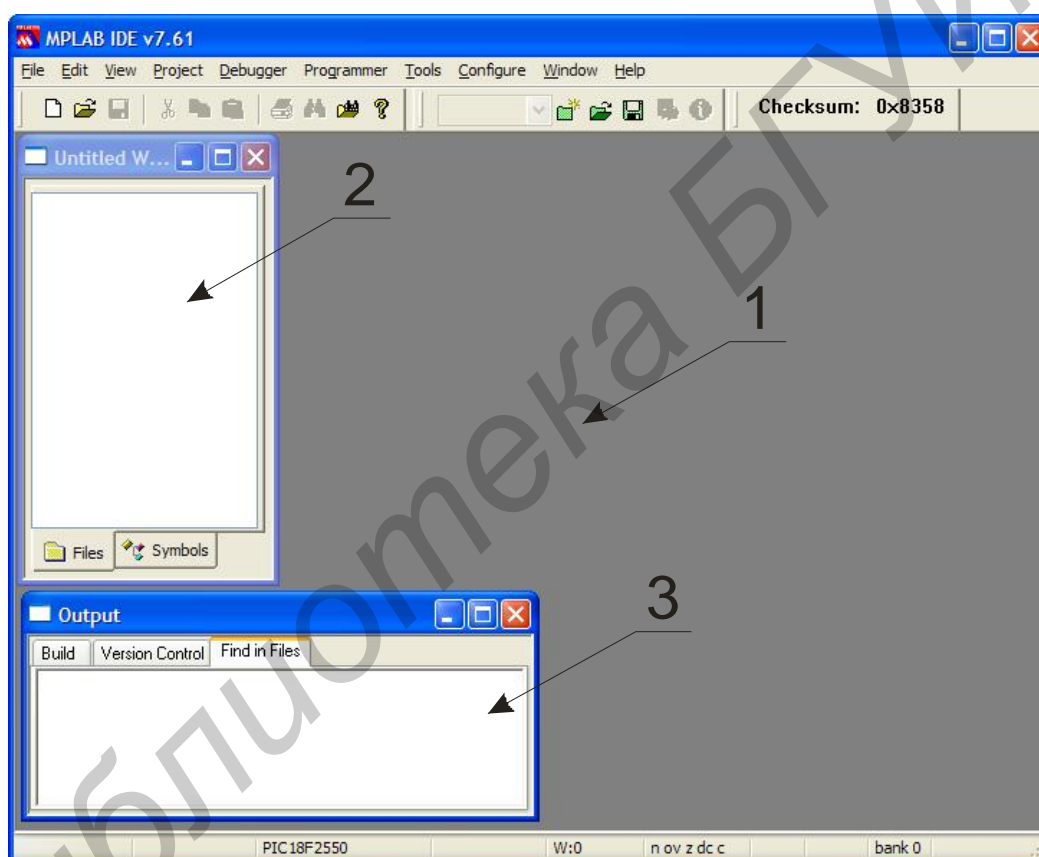


Рисунок 1.1

Рабочий стол среды содержит:

1 Рабочую область, в которой размещаются открытые окна с файлами, диалогами или другой информацией.

2 Окно «Project» (*View-> Project*), где отображается структура проекта, т.е. все участвующие при создании проекта файлы.

3 Окно «Output» (*View-> Output*) – здесь будут отображаться все данные о процессе компиляции проекта.

1.2 Создание нового проекта

Для того чтобы начать работу, нужно создать новый или открыть существующий проект.

Проект – это совокупность файлов, которые используются при создании исполняемого кода, выполняемого микроконтроллером, а также все индивидуальные настройки, такие как положения и размеры окон. Для создания проекта можно использовать мастер проектов (*Project->Project Wizard...*) либо меню «*Project->New...*» Мастер проектов предназначен для быстрого создания нового проекта.

При запуске мастера проектов (*Project->Project Wizard...*) появится окно, показанное на рисунке 1.2.



Рисунок 1.2

После нажатия кнопки «Далее» необходимо в появившемся окне (рисунок 1.3) выбрать из раскрывающегося списка модель микроконтроллера, которая будет использоваться в проекте, например PIC16F628A, и нажать «Далее».

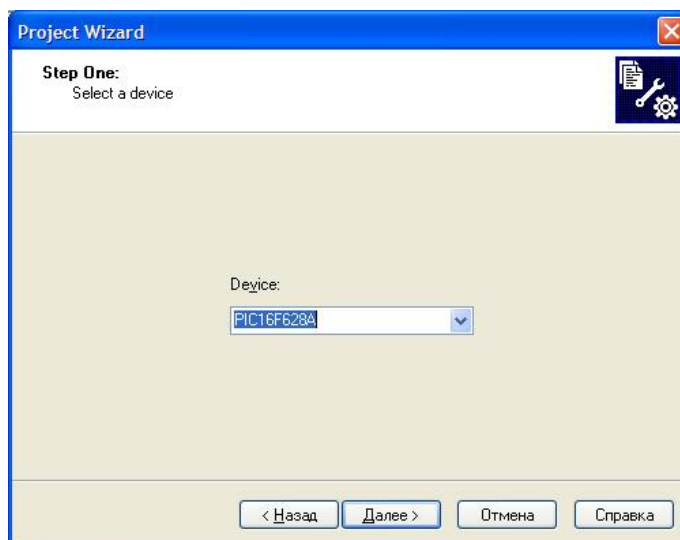


Рисунок 1.3

В следующем окне мастера (рисунок 1.4) необходимо выбрать язык программирования. Из выпадающего меню «*Active Toolsuite*» выбираем «Microchip MPASM Toolsuite» и нажимаем «Далее».

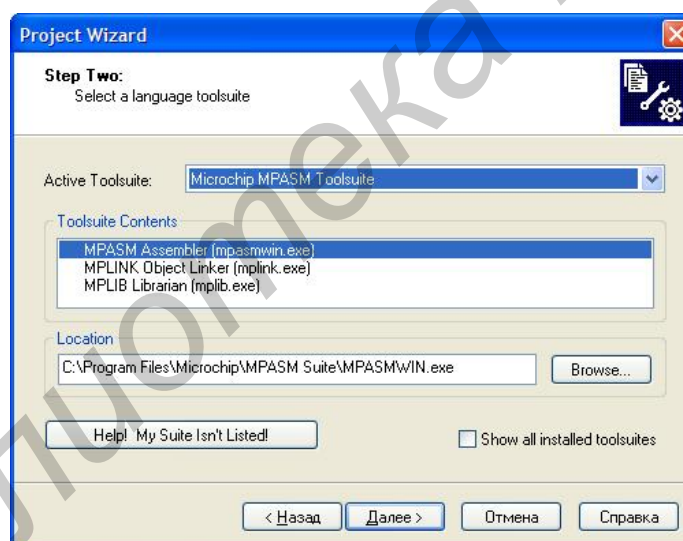


Рисунок 1.4

На следующем шаге (рисунок 1.5) необходимо ввести название проекта и определить рабочую директорию (можно с помощью кнопки «Browse»). Например, D:\Student\ViMPU\041202\MyProject01. В названии проекта, рабочей директории и пути к ней не должно быть русских символов. Нажимаем «Далее».

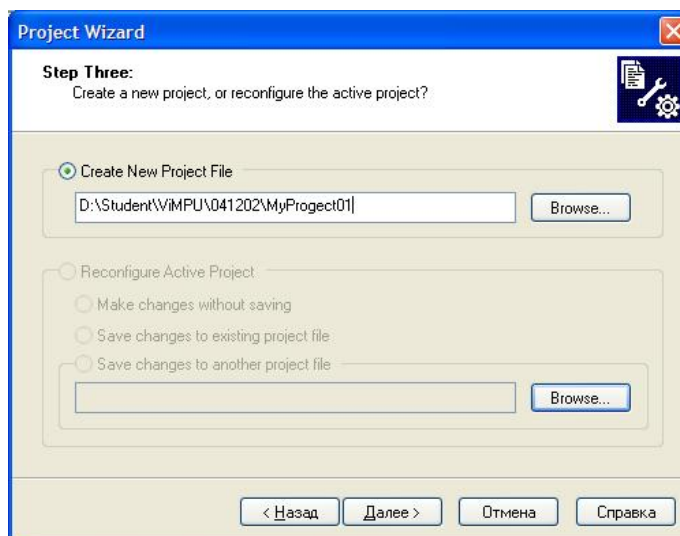


Рисунок 1.5

В следующем окне (рисунок 1.6) необходимо определить исходные файлы, которые будут использоваться в проекте (кнопка Add). Для того чтобы файлы были скопированы в рабочий каталог, необходимо выбрать пиктограмму [C]. Например, [C] C:\Program Files\Microchip\MPASM Suite\P16F628A.INC и [C] C:\Program Files\Microchip\MPASM Suite\Template\Code\16F628ATEMP.ASM. После того как все необходимые файлы добавлены, нажимаем «Далее».

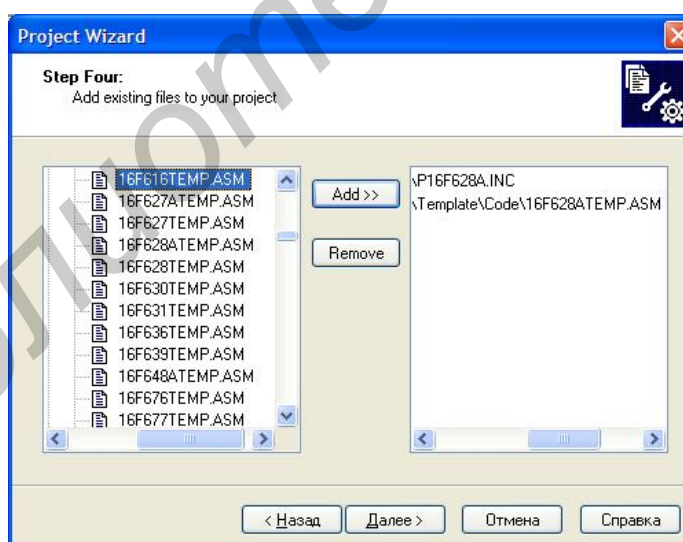


Рисунок 1.6

Последнее окно (рисунок 1.7) «Summary» отображает сведения о том, что было выбрано на предыдущих шагах. Нажимаем «Готово».

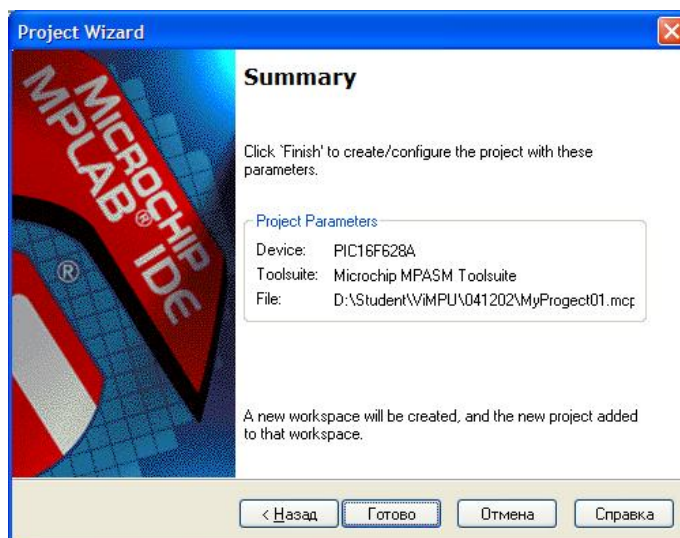


Рисунок 1.7

Рабочий стол среды MPLAB IDE 7 примет вид, показанный на рисунке 1.8.

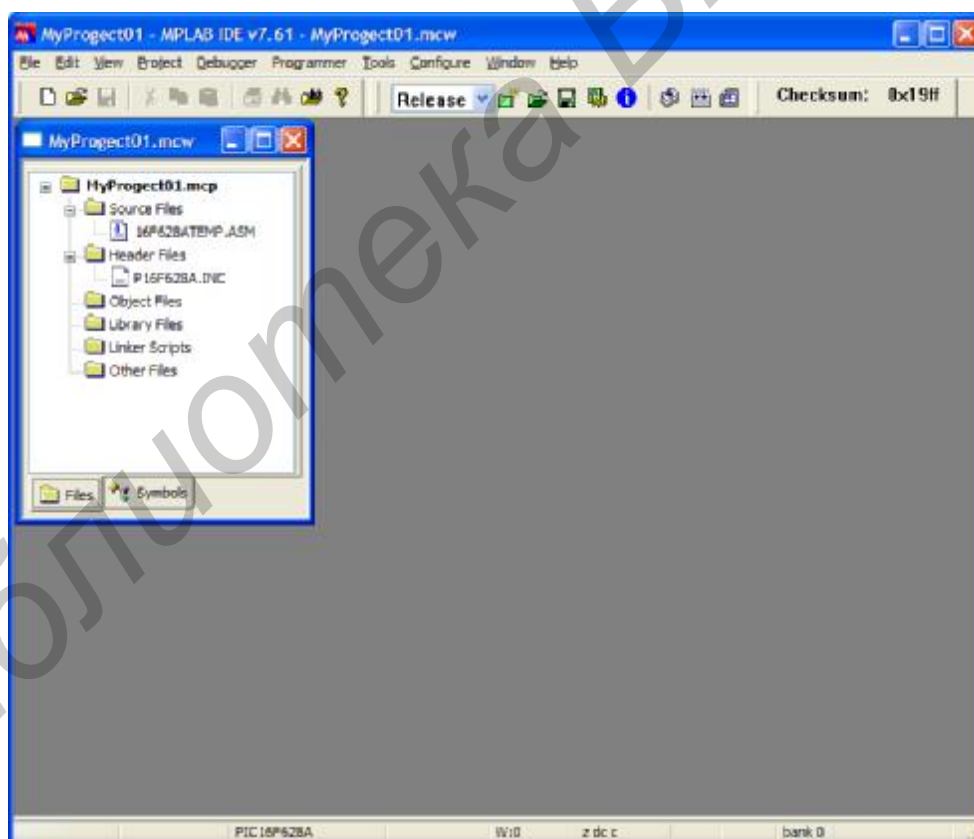


Рисунок 1.8

В окне «Project» появился файл 16F628ATEMP.ASM, который автоматически будет включен в группу «Source files» (исходные файлы). Если два раза

щелкнуть кнопкой мыши по этой записи, то файл 16F628ATEMP.ASM откроется в окне редактора.

Дополним программу. Для этого необходимо после переменных

```
w_temp EQU 0x7E ;
```

```
status_temp EQU 0x7F ;
```

добавить переменные:

```
COUNT EQU 0x20 ; значение счетчика
```

```
DVAR EQU 0x21 ; переменная задержки
```

```
DVAR2 EQU 0x22; переменная задержки
```

и после main вставить текст программы:

```
movlw B'00000111'
```

```
movwf CMCON
```

```
clrf PORTB; очищаем PORTB
```

```
bsf STATUS,RP0; Bank1
```

```
movlw B'00000000'
```

```
movwf TRISB; настраиваем весь PORTB на выход
```

```
bcf STATUS,RP0; Bank0
```

Init

```
clrf COUNT ; инициализация счетчика
```

IncCount

```
incf COUNT,F
```

```
movf COUNT,W; инкрементируем счетчик и
```

```
movwf PORTB ; выводим на PORTB
```

```
call Delay ; вызов подпрограммы задержки
```

```
goto IncCount ;
```

Delay ; подпрограмма задержки

```
movlw 0x40
```

```
movwf DVAR2 ; количество внешних циклов
```


DelayOuter

```
movlw 0xFF
```

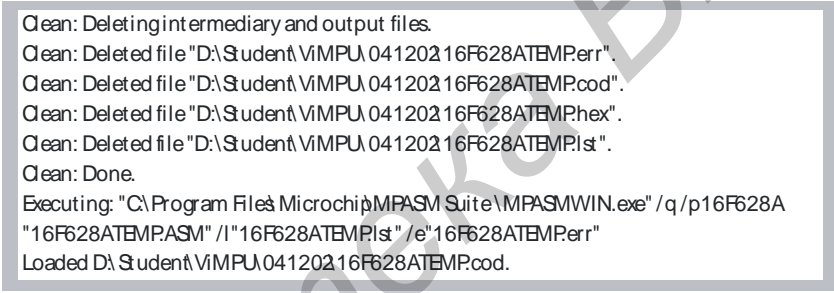
```

movwf    DVAR    ; количество внутренних циклов
DelayInner
decfsz   DVAR,F
goto     DelayInner
decfsz   DVAR2,F
goto     DelayOuter
return
goto    main

```

Теперь можно скомпилировать исходную программу. Нажимаем на пиктограмму «» на панели инструментов.

При успешной компиляции в *окне «Output»* (рисунок 1.9) отобразится следующая информация:



```

Clean: Deleting intermediary and output files.
Clean: Deleted file "D:\Student\ViMPU\04120216F628ATEMPerr".
Clean: Deleted file "D:\Student\ViMPU\04120216F628ATEMPcod".
Clean: Deleted file "D:\Student\ViMPU\04120216F628ATEMPhex".
Clean: Deleted file "D:\Student\ViMPU\04120216F628ATEMPlst".
Clean: Done.
Executing: "C:\Program Files\Microchip\MPLAB Suite\MPLABWIN.exe" /q/p16F628A
"16F628ATEMPASM" /l"16F628ATEMPlst" /e"16F628ATEMPerr"
Loaded D:\Student\ViMPU\04120216F628ATEMP.cod.

```

Рисунок 1.9

Необходимо обратить внимание на последнюю строку: «BUILD SUCCEEDED» – компиляция прошла успешно. «BUILD FAILED» – исходный код содержит ошибки.

2 Использование симулятора в среде MPLAB IDE 7

После написания исходного текста программы и компиляции проекта в MPLAB IDE 7 необходимо проверить, насколько правильно работает программа и насколько верен её алгоритм. Как правило, с первого раза новая программа может работать неправильно; также следует учитывать, что память микроконтроллера изнашивается при каждой операции программирования, поэтому удобно отлаживать исходный текст с помощью компьютерного симулятора MPLAB-SIM.

Данный симулятор моделирует выполнение программы любого типа PICmicro с учетом состояния портов ввода/вывода на скорости, которая зависит от быстродействия персонального компьютера и сложности кода программы.

Симулятор поддерживает следующие функции:

- эмуляция памяти программ;
- прерывание выполнения программы в точках остановки;
- работа по шагам;
- контроль регистров общего и специального назначения;
- измерение времени выполнения кода программы.

Все указанные функции используются в проекте MPLAB IDE 7. Символьные метки могут использоваться для указания точек остановки и трассировки, а также изменения значений регистров памяти данных.








В симуляторе предусмотрена автоматическая работа по шагам. Программа выполняется непрерывно с обновлением всех значений в регистрах после исполнения каждой инструкции.

2.1 Выбор и настройка симулятора

Чтобы активировать симулятор, необходимо выбрать пункт меню «*Debugger->Select Tool->MPLAB SIM*». После выбора в меню произойдут следующие изменения на *рабочем столе среды MPLAB IDE* (рисунок 2.1):

- 1 Появилась надпись «MPLAB-SIM» на панели состояния.
- 2 Появились дополнительные пункты в меню «*Debugger*».
- 3 Появилась панель отладки на панели инструментов.
- 4 Добавилась закладка «MPLAB-SIM» в окне «Output».

Панель отладки содержит:

-  – запуск (F9);  – пауза (F5);  – анимация;  – шаг в (F7);
-  – шаг через (F8);  – шаг из;  – сброс (F6).

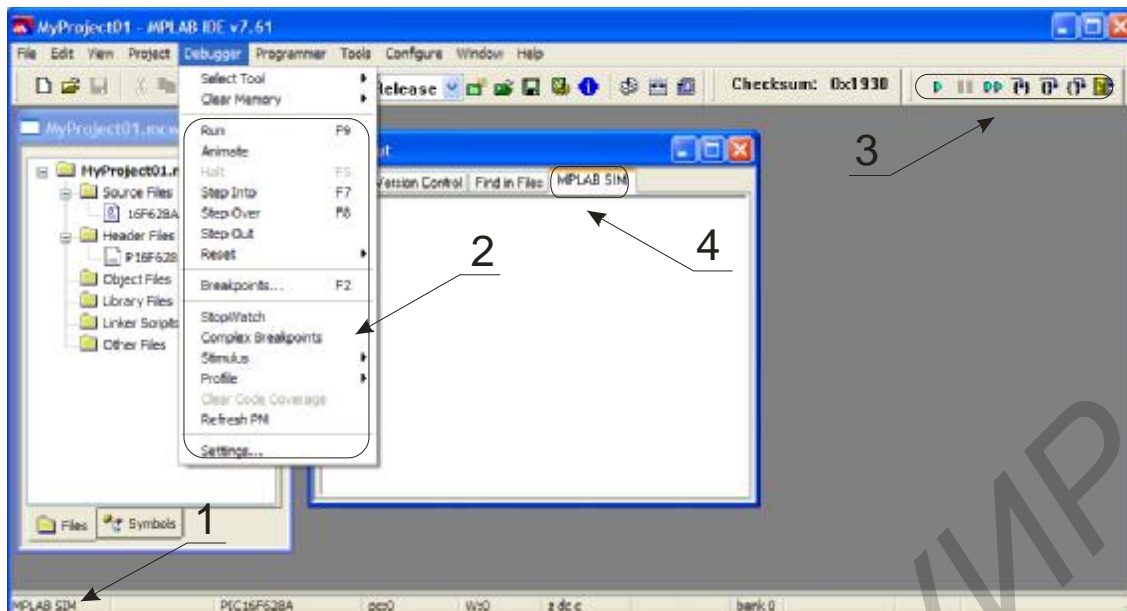


Рисунок 2.1

MPLAB-SIM рассчитывает время работы микроконтроллера исходя из его тактовой частоты, т.к. от неё зависит, сколько времени займет рабочий цикл контроллера.

В реальном устройстве частота микроконтроллера задаётся опорным генератором (это может быть кварцевый или керамический резонатор, внешняя или внутренняя RC-цепочка), а в «виртуальном» микроконтроллере частоту можно указать вручную, для чего необходимо выбрать пункт меню «*Debugger*» > «*Settings...*». В появившемся *окне* «*Simulator settings*» (рисунок 2.2) выберите вкладку «Osc / Trace». В поле «Processor Frequency» введите 20, а в поле «Units»

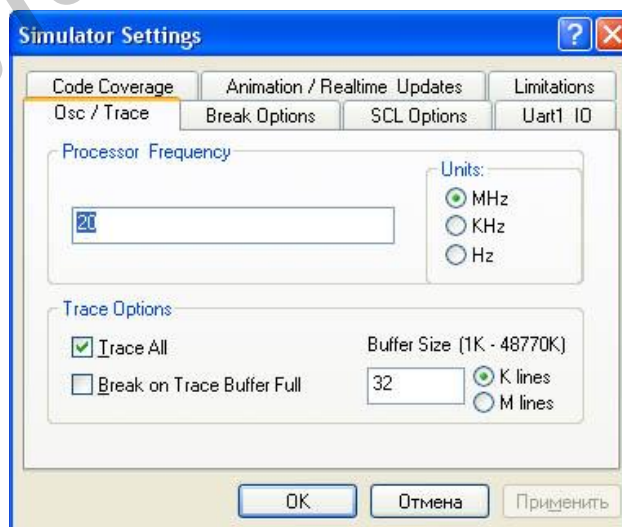


Рисунок 2.2

выберите «MHz», таким образом указав частоту в 20 МГц. Для того чтобы все изменения вступили в силу, нужно нажать кнопку «ОК».

2.2 Симуляция выполнения программы

Написанный в подразделе 1.2 программный код выполняет увеличение на единицу значение регистра COUNT с интервалом, равным задержке выполнения подпрограммы «Delay». Далее содержимое регистра COUNT сохраняется в регистре PORTB. Симулятор позволяет отслеживать все изменения регистров. Для этого:

а) выберите пункт меню *View->Watch*. Появится окно наблюдения за состоянием регистров и переменных (Watch);

б) добавьте регистр COUNT в список наблюдаемых регистров из правого раскрывающегося списка, нажмите кнопку «Add Symbol»;

в) выберите PORTB из левого раскрывающегося списка и нажмите кнопку «Add SFR».

Окно Watch примет вид, представленный на рисунке 2.3:

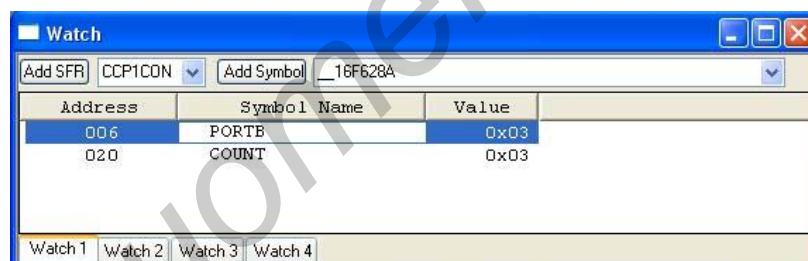







Рисунок 2.3

Для начала симуляции программы выбираем пункт меню «*Debugger>Reset>Processor Reset*» или «», после этого зеленый указатель «» появится в начале кода программы. Если нажать на иконку «» или выбрать пункт меню «*Debugger>Step Into*», то произойдет выполнение одной команды (одного шага). Чтобы запустить симуляцию в режиме анимации, нажмите на иконку «» на панели инструментов MPLAB. Остановить симуляцию можно с помощью иконки пауза «».

MPLAB-SIM позволяет рассчитывать время выполнения программы. Пусть требуется узнать время выполнения подпрограммы «Delay». Для этого установите зеленый указатель на команде вызова подпрограммы задержки (щелкните правой кнопкой мыши по команде «call Delay», выберите в открывшемся контекстном меню Set PC at Cursor). Далее установите точку остановки на команде «goto IncCount».

2.3 Точки остановки

Точки остановки (breakpoint) – это указание симулятору остановиться на выполнении какой-либо определённой команды. Есть несколько способов установить точку остановки:

- Выбрать пункт «Set Breakpoint» из контекстного меню, появляющегося при нажатии правой кнопки мыши по нужной строке программы.
- Выполнить двойной щелчок левой кнопкой мыши по строке программы.

В результате выполнения любого из указанных действий слева от строки программы появится пиктограмма «», как показано на рисунке 2.4:

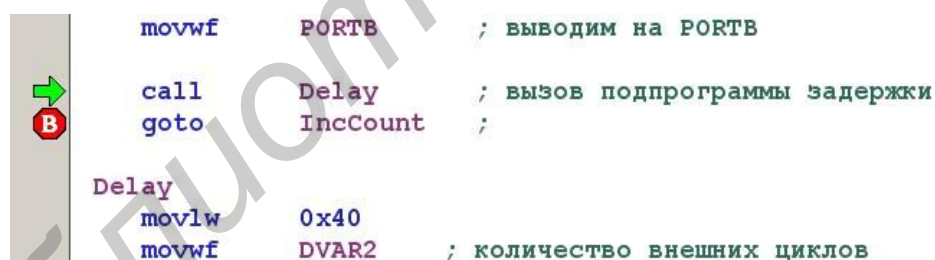


Рисунок 2.4

Убираются точки остановки теми же способами, которыми устанавливаются (правый щелчок мышью по команде -> Remove Breakpoint или двойной щелчок левой кнопкой мыши по полю слева от команды).

Для измерения временных интервалов в симуляторе MPLAB-SIM предусмотрен инструмент StopWatch. Чтобы запустить его, выберите пункт меню «Debugger->StopWatch». Появится *окно StopWatch*, показанное на рисунке 2.5.

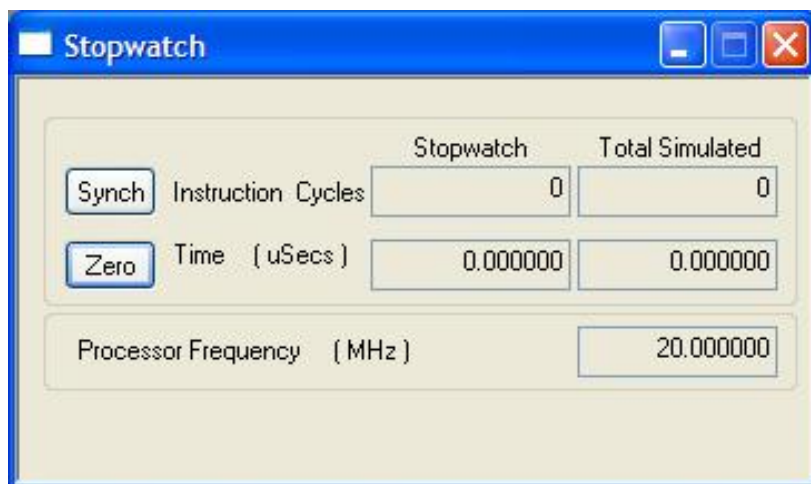


Рисунок 2.5

Запустите программу, выбрав пункт меню «*Debugger*»*Run*» или «▶». Когда симулятор остановит выполнение программы в точке останова, в окне Stopwatch в поле Time появится информация о времени, прошедшем за период выполнения, а в поле Instruction Cycles – количество выполненных циклов (в PIC16 один цикл равен четырем тактам задающего генератора). Если всё сделано правильно, в окне «Time» отобразится время 9.844 mSecs (миллисекунд). Кнопка «Zero» сбрасывает результат измерения.

Также в симуляторе есть возможность устанавливать точки останова по определенному событию или по определенной цепочке событий. Например, необходимо, чтобы выполнение программы было остановлено, когда значение счетчика COUNT равняется 10h. Для этого через пункт меню «*Debugger -> Complex Breakpoints*» откройте **окно «Simulator Complex Breakpoints»** (рисунок 2.7), выберите «Add Breakpoints» и в открывшемся **окне «Set Breakpoint»** (рисунок 2.6) перейдите на закладку «Data Memory». В поле «Address» из выпадающего списка «Symbol/Hex» выберите регистр «COUNT». В следующем поле из списка «Breakpoint Type» выберите «Read Specific Byte» и задайте значение «Specific Value», равное 10.

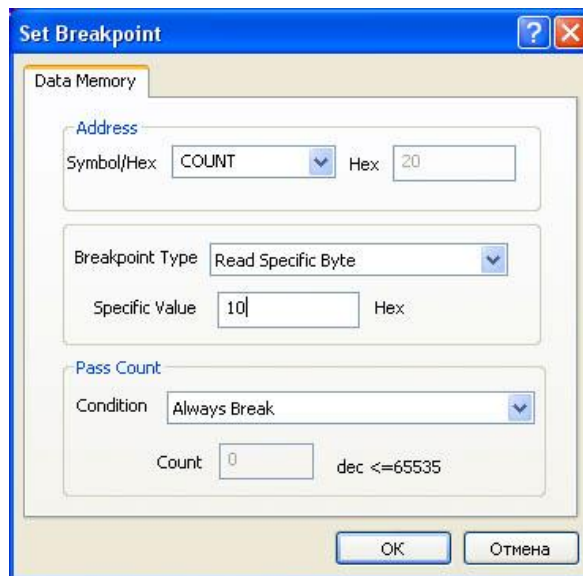


Рисунок 2.6

Теперь симулятор будет прерывать выполнение программы каждый раз, когда значение регистра «COUNT» будет равно 10h. Если необходимо изменить параметры прерывания выполнения программы, *в окне «Simulator Complex Breakpoints»* нажмите правой кнопкой мыши по строке «Data...» и из выпадающего списка выберите пункт «Edit», как показано на рисунке 2.7. Также с помощью этого меню можно разрешить или запретить использование такого типа точки остановки.

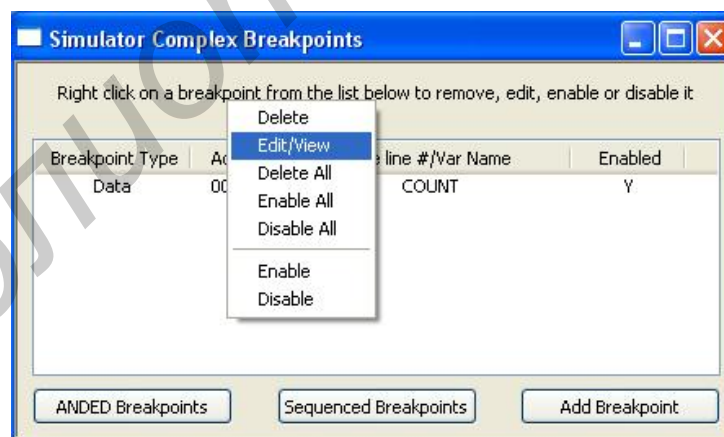


Рисунок 2.7

2.4 Точки трассировки

Дополнительным средством контроля над ходом выполнения программы являются точки трассировки.

Симулятор MPLAB-SIM поддерживает буфер трассировки, в котором сохраняются данные точек трассировки. Допускается запись в буфер при переполнении с удалением более старых значений (если не выбран параметр остановки программы при переполнении буфера).

Чтобы разрешить использование буфера трассировки, необходимо выбрать пункт меню «Debugger->Settings...». В окне «Simulator settings» выберите закладку «Osc / Trace» (см. рисунок 2.2). В поле «Trace Options» поставьте галочку возле параметра «Trace All» и нажмите «ОК». После этого в меню «View-> Simulator Trace» станет доступно *окно трассировки «Trace»* (рисунок 2.8).

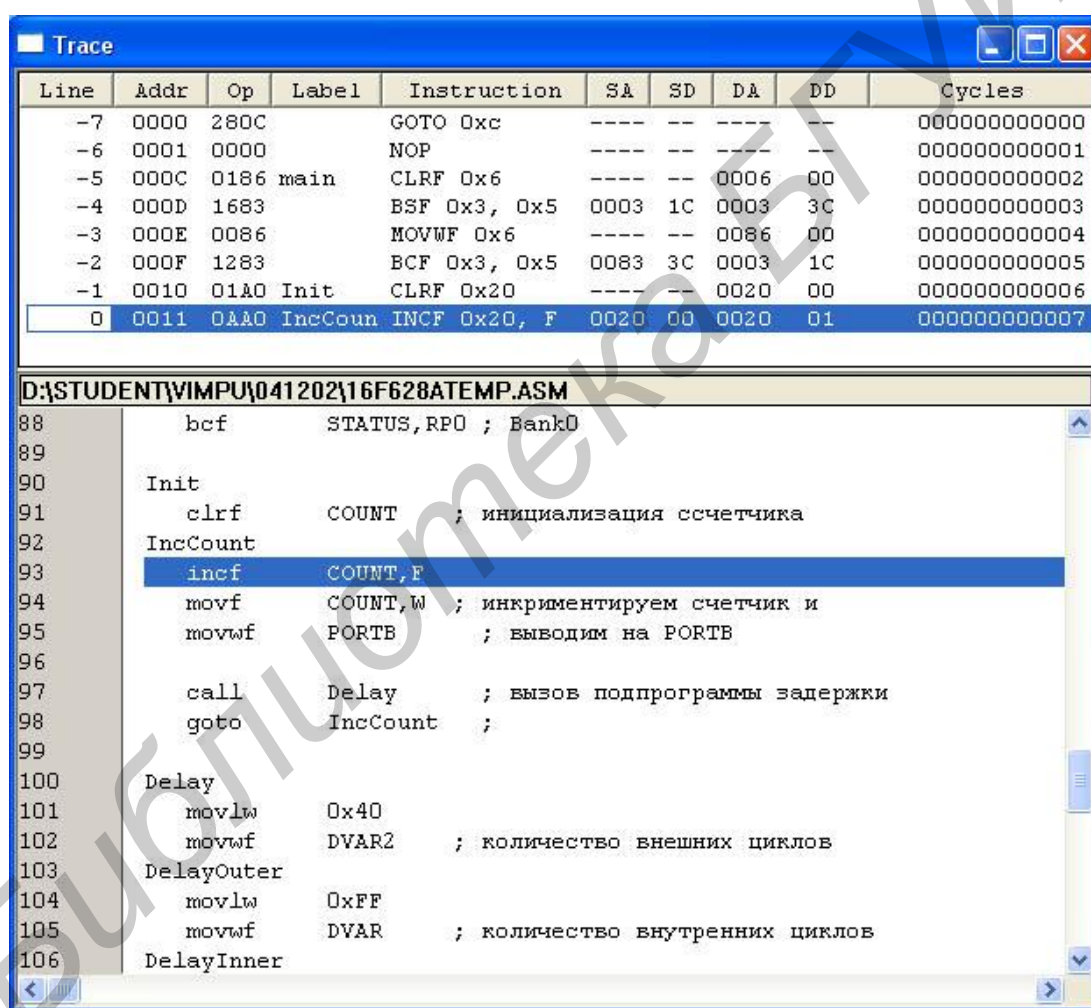


Рисунок 2.8

Запустите выполнение программы и остановите. Окно трассировки примет вид, как на рисунке 2.8. Оно будет содержать следующую информацию:

- **Line** (Line Number) – позиция относительно пункта остановки;
- **Addr** (Address) – адрес в памяти программ;

- **Op** (Opcode) – код команды;
- **Label** (Label) – метка переходов в программе;
- **Instruction** (Instruction) – мнемоника команды;
- **SA** (Source Data Address) – адрес исходных данных;
- **SD** (Source Data Value) – значение исходных данных;
- **DA** (Destination Data Address) – адрес данных назначения;
- **DD** (Destination Data Value) – значение данных назначения;
- **Cycles/Time** (Time Stamp) – интервал времени от сброса в циклах или в секундах.

Использование фильтров трассировки «Filter in trace» или «Filter out trace» позволяет добавлять или исключать точки трассировки из буфера трассировки.

Для того чтобы установить фильтр трассировки, необходимо нажать правой кнопкой мыши по нужной строке программы и из контекстного меню выбрать «Filter in trace» или «Filter out trace».

Слева от строки программы появится пиктограмма «» (Filter in trace) или «» (Filter out trace).

Примечание – В MPLAB-SIM можно использовать или «Filter in trace», или «Filter out trace», т.к. при выборе одного из параметров второй будет недоступен.

2.5 Использование стимулов

Во время симуляции выполнения программы MPLAB-SIM позволяет имитировать с помощью так называемых «стимулов» внешние сигналы на портах ввода/вывода, а также изменять данные регистров специального (SFR) и общего (GPR) назначения.

Существуют два основных типа стимулов:

- Асинхронный стимул – непосредственное управление состоянием портов ввода/вывода.
- Синхронный стимул – определенная последовательность изменения

сигналов на портах ввода/вывода или последовательное изменение значения регистров SFR и GPR.

Для создания синхронных и асинхронных стимулов откройте **окно диалога «Stimulus»**, которое представлено на рисунке 2.9, с помощью меню *Debugger-> Stimulus ->New Workbook.*

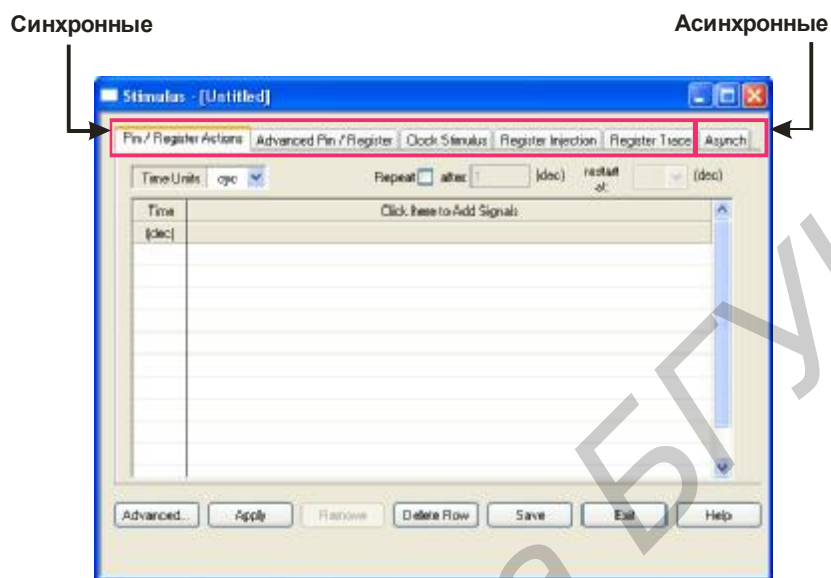


Рисунок 2.9

2.5.1 Асинхронный стимул

Асинхронный стимул используется для моделирования логического состояния порта ввода/вывода, настроенного на вход (устанавливаются значения +5В или 0В). В диалоговом окне асинхронных стимулов «Asynch» выбирается в поле «Pin/SFR» вывод порта микроконтроллера (МК). В поле «Action» выставляется уровень входного сигнала на выбранном выводе МК. Нажатие соответствующей кнопки «Fire» приведет к изменению состояния порта RB1,

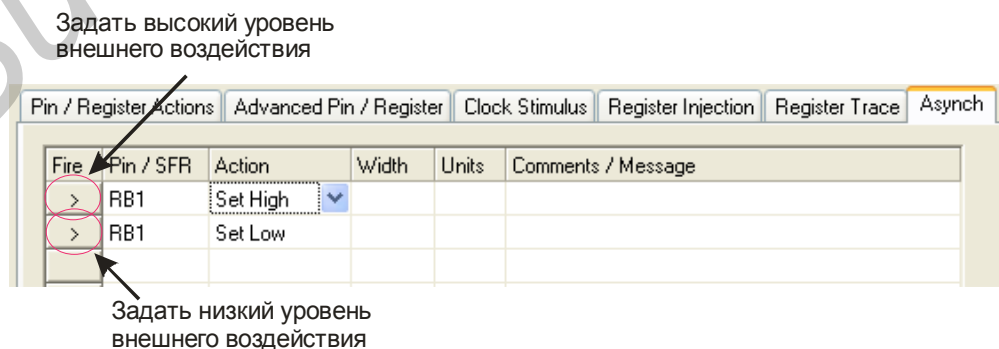


Рисунок 2.10

и в окне «Output» на закладке «MPLAB-SIM» (см. рисунок 2.1) появится надпись «SIM-N0001 Note: Asynchronous Stimulus Set Low RB1 fired».

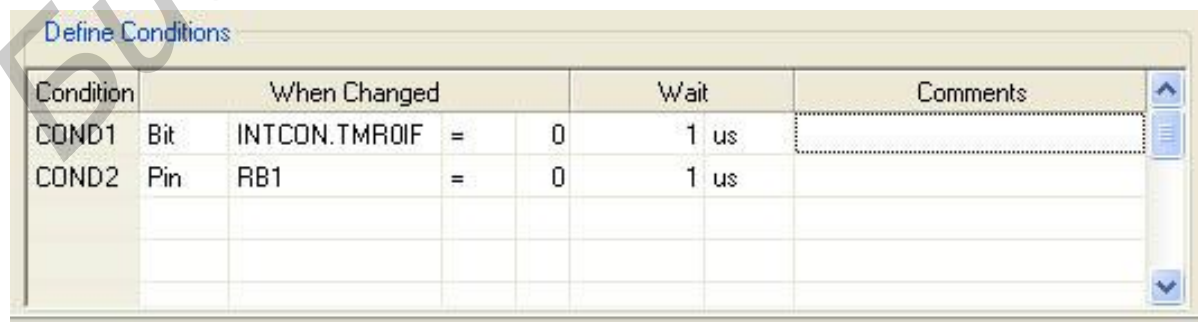
2.5.2 Синхронный стимул

Для того чтобы использовать синхронный стимул, откройте закладку «Pin/Register Actions» в окне «Stimulus». Выберите из выпадающего списка «Time Units» интервал времени срабатывания стимула в микросекундах «us».

Нажмите левой кнопкой мыши по надписи «Click here to Add Signals», в появившемся окне «Add/Remove Pin/Registers» выберите в поле «Available Signals» необходимый вывод, например RB1. Далее нажмите кнопку «Add» и «OK». В появившемся столбце «RB1» задаются значение нуля или единицы, которые соответствуют низкому или высокому уровню на порте RB1 микроконтроллера (рисунок 2.10).

В столбце «Time» задаются интервалы времени от момента нажатия кнопки «Apply» в окне «Stimulus», например 100, 200,..., 900, 1000 us. При выполнении симуляции программы каждый интервал времени будет отображаться в окне «Output» на закладке «MPLAB SIM». Нажмите кнопки «Apply» и «Save», чтобы применить и сохранить параметры воздействия.

На закладке «Advanced Pin/Register» можно задавать изменения сигналов на портах ввода/вывода или изменение значения регистров SFR и GPR по определенному событию. В поле «Define Conditions» задаются условия, в соответствии с которыми будут происходить изменения воздействия (рисунок 2.11).



Condition	When Changed	Wait	Comments
COND1	Bit INTCON.TMR0IF = 0	1 us	
COND2	Pin RB1 = 0	1 us	

Рисунок 2.11

Оно содержит:

- «Condition» – имя события;
- «When Changed» – условие события;
- «Wait» – задержка перед реагированием стимула, если условие истинно.

ТИННО.

Enable	Condition	Type	Re-Arm Delay	RBO	Click here to Add Signals
<input checked="" type="checkbox"/>	COND1	Cont	1 us	1	
<input checked="" type="checkbox"/>	COND2	1x		0	
<input type="checkbox"/>					
<input type="checkbox"/>					
<input type="checkbox"/>					

Рисунок 2.12

Поле «Define Triggers» (рисунок 2.12) позволяет настраивать реакцию стимула на событие и содержит следующие столбцы:

- «Enable» – разрешение или запрещение реагирования стимула;
- «Condition» – имя события;
- «Type» – тип воздействия («Count» – многократное, «1x» – однократное);
- «Re-Arm Delay» – задержка до начала проверки следующего события.

2.5.3 Файл стимула

Сохранение и загрузка файла сценария стимулов осуществляется при помощи окна «Advanced SCL Operations», которое вызывается при нажатии кнопки «Advanced...» в окне «Stimulus». Создайте файл, нажав кнопку «Generate SCL File», и сохраните его в директории с программой. Кнопки «Attach» и «Detach» подключают и отключают сохраненный файл сценария.

ЛИТЕРАТУРА

1 PIC16F62X. Однокристалльные 8-разрядные FLASH CMOS микроконтроллеры компании Microchip technology incorporated [Электронный ресурс]: пер. с англ. – М. : ООО «Микрочип», 2001. – 148 с. Режим доступа: [http:// www.microchip.ru](http://www.microchip.ru)

2 Бурак, А. И. Интегрированная среда MPLAB IDE разработки программ для микроконтроллеров PICmicro фирмы Microchip : метод. пособие к лабораторным работам по курсу «Цифровые и микропроцессорные устройства» / А. И. Бурак, В. Н. Левкович. – Минск : БГУИР, 2003. – 31 с.

3 Левкович, В. Н. Конструирование программ на Ассемблере для микроконтроллеров семейства PICmicro: учеб. пособие по курсу «Цифровые и микропроцессорные устройства» для студ. спец. 39 01 01 «Радиотехника» и 39 01 02 «Радиоэлектронные системы» всех форм обуч. / В. Н. Левкович. – Минск : БГУИР, 2004. – 80 с.

ПРИЛОЖЕНИЕ

Директивы MРasm

ДИРЕКТИВА	ОПИСАНИЕ	ПРИМЕР
<i>Директивы управления</i>		
CONSTANT	Определение символьной константы	constant cnt=255
#DEFINE	Определение текстовой последовательности для замены	#define snd portsnd, 1
END	Конец блока программы	end
EQU	Определение константы	temp equ 0xF0
ERROR	Сообщение об ошибке	error "error line"
ERROR LEVEL	Установка типа сообщений об ошибках в файле листинга и файле ошибок	errorlevel 1, -202
INCLUDE	Вставить другой файл источника	include <addmain.asm>
LIST	Определение формата (тип микроконтроллера, количество символов в строке, табуляция и др.)	list p=16f628A, f=INHX32, r=DEC
MESSG	Создать пользовательское сообщение	messg "see here!"
NOLIST	Запретить вывод	nolist
ORG	Установить начальный адрес программы	org 0x100
PAGE	Вставить страницу в файл листинга	page
PROCESSOR	Установить тип микроконтроллера	processor 16F628A
RADIX	Установить систему счисления по умолчанию для выражения данных	radix dec
SET	Определение константы. Аналогична EQU, но впоследствии можно переопределить	temp set b'00110011'
SPACE	Вставить пустые строки в файл листинга	space 3
SUBTITLE	Вставить второй заголовок в файл листинга	subtitle "Main Project"
TITLE	Вставить заголовок в файл листинга	title "Project Of PIC"
#UNDEFINE	Удаление определенной текстовой последовательности	#undefine snd
VARIABLE	Определение символьной переменной	variable temp=0xF0
<i>Условия</i>		
ELSE	Начало блока альтернативного условия (IF)	else
ENDIF	Завершение блока условия	endif
ENDW	Завершение цикла ПОКА	endw
IF	Начало блока условия	if version == 100
IFDEF	Выполнить, если определено	ifdef testing
IFNDEF	Выполнить, если не определено	ifndef testing
WHILE	Цикл ПОКА	while i < count
<i>Данные</i>		
CBLOCK	Определение блока констант	cblock 0x20
__CONFIG	Описание бит конфигурации микроконтроллера	__config H'FFFF'
DATA	Создание числовых и текстовых данных	txt data "please", 0x30
DB	Определение байта данных	temp db 0xFF
DE	Определение данных в EEPROM	temp de 0xF0, 0xF1
DT	Определение таблицы	temp dt "text", 0, 0x30
DW	Определение слова (2 байта) данных	temp dw 39, "text"
ENDC	Окончание блока констант	endc
FILL	Заполнение области константой	fill 0x1009, 5
__IDLOCS	Определение ID	__idlocs H'FFEE'
RES	Резервирование памяти	buffer res 64
<i>Макросы</i>		
ENDM	Окончание макроса	endm
EXITM	Выход из макроса	exitm
EXPAND	Полный текст макроса в файле листинга	expand
LOCAL	Определение локальной переменной в макросе	local leng, tmp
MACRO	Определение макроса	out_sym macro temp

Арифметические операторы MASM

Оператор	Описание	Пример
\$	Текущий счетчик программы	goto \$ + 3
(Левая скобка	1 + (d * 4)
)	Правая скобка	(leght + 1) * 255
!	Операция «НЕ» (логическая инверсия)	if ! (a - b)
~	Инверсия	flags = ~ flags
-	Отрицательное число (вторая инверсия)	- 1 * leght
high	Выделить старший байт слова	movlw high llasid
low	Выделить младший байт слова	movlw low (llasid + .2551)
*	Умножение	a = c * b
/	Деление	a = b / c
%	Модуль	leght = totall % 16
+	Сложение	tot_len = leght * 8 + 1
-	Вычитание	Entry_Son = (Tot - 1) / 8
<<	Сдвиг влево	val = flags << 1
>>	Сдвиг вправо	val = flags >> 1
>=	Больше либо равно	if ent >= num
>	Больше	if ent > num
<	Меньше	if ent < num
<=	Меньше либо равно	if ent <= num
==	Равно	if ent == num
!=	Не равно	if ent != num
&	Поразрядное «И»	flags = flags & err_bit
^	Поразрядное «ИСКЛЮЧАЮЩЕЕ ИЛИ»	flags = flags ^ err_bit
	Поразрядное «ВКЛЮЧАЮЩЕЕ ИЛИ»	flags = flags err_bit
&&	Логическое «И»	if (len == 512) && (b == c)
	Логическое «ИЛИ»	if (len == 512) (b == c)
=	Присвоить значение	entry_index = 0
+=	Присвоить значение	entry_index += 1
-=	Присвоить значение	entry_index -= 1
*=	Присвоить значение	entry_index *= leght
/=	Делить и присвоить значение	entry_index /= leght
%=	Модуль и присвоить значение	entry_index %= 8
<<=	Сдвиг влево и присвоить значение	entry_index << 3
>>=	Сдвиг вправо и присвоить значение	entry_index >> 4
&=	«И» и присвоить значение	entry_index &= err_flags
=	«ВКЛЮЧАЮЩЕЕ ИЛИ» и присвоить значение	entry_index = err_flags
^=	«ИСКЛЮЧАЮЩЕЕ ИЛИ» и присвоить значение	entry_index ^= err_flags
++	Увеличить на 1 (инкремент)	i ++
--	Уменьшить на 1 (декремент)	i --

Формат представления чисел

ФОРМАТ	СИНТАКСИС	ПРИМЕР
Десятичный	D'число' .число	D'100' .100
Шестнадцатеричный	H'число' 0xчисло	H'f9' 0xAF00
Восьмеричный	O'число'	O'777'
Двоичный	B'число'	B'11110000'
Символьный	'символ' A'символ'	'C' A'C'

Стандартные расширения для файлов MPLAB

РАСШИРЕНИЕ	НАЗНАЧЕНИЕ ФАЙЛА
*.ASM	Исходный файл на ассемблере
*.C	Исходный файл на C
*.CFG	Файл конфигурации
*.COD	Содержит символьную информацию и объектный код
*.CSV	Файл с записью трассировки (только для MPLAB-ICE 2000)
*.DAT	Файл данных симулятора
*.ERR	Файл обнаруженных ошибок, генерируется ассемблером или C при компиляции
*.H	Добавленный файл на C
*.HEX	Файл с машинными кодами в HEX формате для PIC микроконтроллеров
*.HLP	Файл помощи
*.INC	Добавленный файл на ассемблере
*.INI	Конфигурация MPLAB и установленного языка программирования
*.KEY	Файл схемы кнопок MPLAB
*.LKR	Файл сценария компоновки MPLINK
*.LST	Абсолютный листинг, генерируется ассемблером или C при компиляции
*.MTC	Файл конфигурации языка программирования
*.PJT	Файл содержит главную информацию о проекте
*.REG	Файл, описывающий модификацию регистров при отладке
*.STI	Файл, описывающий входные сигналы на входах микроконтроллера
*.TB	Файл трассировки, точек остановки
*.TBR	Файл панели инструментов
*.TPL	Временный файл
*.TRC	Файл записи трассировки
*.TXT	Файл записи трассировки (только MPLAB-ICE 2000)
*.WAT	Файл окна просмотра

Словарь терминов

Alpha character

набор символов, который включает в себя только буквы латинского алфавита: a, b,... z, A, B,... Z.

Alphanumeric

набор символов, который включает в себя набор alpha символов и цифры: 0, 1,... 9.

Assemble (ассемблирование)

операция, производимая ассемблером (assembler).

Assembler (ассемблер)

инструмент языка программирования, который переводит текстовый файл-источник с расширением .asm в исполняемый машинный код.

Breakpoint

точка остановки, адрес, устанавливаемый пользователем, где выполнение программы останавливается.

Build

функция перекомпилирования всех исходных файлов рабочего проекта.

C

язык программирования высокого уровня для PIC микроконтроллеров.

Calibration memory

специальный регистр или регистры для калибровки внутреннего RC-генератора.

Compile (компилирование)

операция, производимая компилятором.

Compiler (компилятор)

инструмент языка программирования, переводящий исходный файл в исполняемый машинный код.

Configuration bits (биты конфигурации)

специальные биты, устанавливаемые при программировании микроконтроллера, определяющие рабочую конфигурацию. В исходном файле необходимо определять, используя директиву `__config`, при использовании симулятора или эмулятора необходимо установить в меню *Options>Development Mode*.

EEPROM

электрически стираемая и программируемая память только для чтения (Electrically Erasable Programmable Read Only Memory). Особый тип памяти, доступный из основной программы микроконтроллера для чтения и записи. Содержимое EEPROM сохраняется при выключении напряжения питания.

Extended Microcontroller Mode

режим расширенного микроконтроллера, только для PIC17CXXX и PIC18CXXX кристаллов, при котором используется как внутренняя память программы, так и внешняя. При превышении объема внутренней памяти автоматически происходит переход на внешнюю память.

External RAM

внешняя память для чтения/записи, возможна только для PIC17CXXX и PIC18CXXX кристаллов.

Hex Code

стандартный файл hex формата, содержащий выполняемые инструкции, результат ассемблирования или компилирования исходных файлов. Hex Code может быть конвертирован в Object Code (объектный код).

Hex File

ASCII файл, содержащий шестнадцатеричный адрес и содержимое (hex code). Файлы этого формата являются исходными файлами программаторов.

ICD

внутрисхемный отладчик (In-Circuit Debugger).

ICE

внутрисхемный эмулятор (In-Circuit Emulator).

IDE

интегрированная среда разработки (Integrated Development Environment). MPLAB IDE включает в себя: компилятор, ассемблер, менеджер проекта, редактор, отладчик, симулятор и другие инструменты.

Identifier

функция или имя переменной.

Librarian

библиотекарь – инструментальный язык для создания и перемещения библиотек.

Library

библиотека – коллекция перемещаемых объектных модулей.

Link

функция, выполняемая линкером (linker).

Linker

инструментальный язык, комбинирующий объектные файлы (Object Files) и библиотеки для создания выполняемого кода.

Linker Script Files

командный файл MPLINK с расширением .lkr, определяющий опции связывания и описывающий возможную память.

Listing File

листинг-файл – текстовый файл, показывающий машинный код, сгенерированный для каждого C оператора или ассемблерной инструкции, а также MPASM директивы и макросы.

Machine Code

выполняемый машинный код.

Macro

при ассемблировании в исходном тексте имя макроса заменяется на определенные ранее инструкции ассемблера (участок кода). Начало макроса соответствует директиве macro и оканчивается директивой endm.

Macro Directives

директивы, контролирующие выполнение макроса и распределение данных внутри тела макроса.

Make Project

команда для перекомпилирования только тех файлов-источников, которые были изменены после последнего компилирования.

Microcontroller Mode

одна из возможных конфигураций памяти программы только для PIC17CXXX и PIC18CXXX кристаллов. В этом режиме используется только внутренняя память, расположенная на одном кристалле с микроконтроллером.

Microprocessor Mode

одна из возможных конфигураций памяти программы только для PIC17CXXX и PIC18CXXX кристаллов. В этом режиме используется только внешняя память программы, внутренняя память отключена. Адресное пространство составляет 64 Кбайта.

Node

компонент проекта MPLAB.

NOP

нет операции.

Object Code

объектный код, получаемый путем ассемблирования или компилирования исходного файла. Этот перемещаемый код может быть загружен в MPLINK для создания выполняемого кода. Объектный код содержится в объектном файле.

Object File

модуль, который содержит перемещаемый объектный код или данные и ссылки на внешний код или данные.

Off-Chip Memory

память

Opcodes

Operational Codes. Мнемоника, которая при ассемблировании или компиляции порождает код.

Pod

внешнее устройство эмулятора, содержащее память, таймеры, логику прерываний.

Power-on-Reset Emulation

программный случайный процесс записи случайных значений в регистры общего назначения (RAM).

Program Counter

регистр, содержащий адрес текущей выполняемой команды (инструкции).

Program Memory

область памяти микроконтроллера, где хранится код программы (команды, инструкции). В эту область загружается при симулировании или эмуляции код текущей программы.

Programmer

программатор, устройство для записи программ в микросхемы, такие, как микроконтроллеры, микросхемы памяти и т. п.

Project

совокупность исходных файлов и инструкций по компилированию или ассемблированию для одного приложения.

RAM

память с произвольным доступом (Random access memory), память данных.

ROM

память только для чтения (Read only memory), память программ.

SFR, Special Function Registers

регистры специального назначения (Special Function Registers), ПОН.

Simulator

программа моделирования работы микроконтроллера.

Simulator Stimulus

описание внешних входных сигналов. Возможность изменять состояние или задавать периодический или произвольный сигнал.

Source

источник кода, обычно текстовый файл, содержащий инструкции ассемблера или код на C.

Source Code - Assembly

источник кода, содержащий инструкции микроконтроллера, директивы ассемблера и макросы, который будет переведен в машинный код при ассемблировании.

Source Code - C

программа, написанная на языке высокого уровня C, которая будет переведена в машинный код при компилировании.

Stack

стек, список вызываемых подпрограмм. При выполнении call или при прерывании в стек заносится текущий адрес программы, а при команде return и возвращении из прерывания адрес из стека переносится в счетчик программы.

Stopwatch

счетчик, измеряющий выполненные циклы.

Tool Bar

колонка или полоса кнопок-иконок, по которым можно щелкать мышью для выполнения функций MPLAB.

Watchdog Timer

таймер микроконтроллера, сбрасывающий его при переполнении. Разрешить или запретить таймер возможно установкой соответствующих битов конфигурации микроконтроллера при программировании. Выбрать и изменить предделитель и его коэффициент можно в программе.

Watch Window

окно просмотра содержит список переменных и регистров, которые можно просматривать и модифицировать при отладке.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	3
1 Начало работы с MPLAB IDE 7.....	4
1.1 Запуск среды проектирования	4
1.2 Создание нового проекта.....	5
2 Использование симулятора в среде MPLAB IDE 7	10
2.1 Выбор и настройка симулятора	11
2.2 Симуляция выполнения программы.....	13
2.3 Точки остановки	14
2.4 Точки трассировки	16
2.5 Использование стимулов	18
2.5.1 Асинхронный стимул.....	19
2.5.2 Синхронный стимул	20
2.5.3 Файл стимула	21
ЛИТЕРАТУРА.....	22
ПРИЛОЖЕНИЕ	23

Учебное издание

**РАЗРАБОТКА ПРОГРАММ
ДЛЯ МИКРОКОНТРОЛЛЕРОВ ФИРМЫ MICROCHIP
В ИНТЕГРИРОВАННОЙ СРЕДЕ MPLAB IDE 7**

Методические указания
к лабораторным работам по курсу
«Микропроцессорные устройства»
для студентов радиотехнических специальностей
всех форм обучения

Составители:

Казека Александр Анатольевич
Мартинovich Алексей Васильевич
Давыдов Игорь Геннадьевич

Редактор Т. П. Андрейченко
Корректор Е. Н. Батурчик

Подписано в печать 06.03.2008.
Гарнитура «Таймс».
Уч.-изд. л. 1,6.

Формат 60×84 1/16.
Печать ризографическая.
Тираж 120 экз.

Бумага офсетная.
Усл. печ. л. 1,98.
Заказ 6.

Издатель и полиграфическое исполнение: Учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники»
ЛИ №02330/0056964 от 01.04.2004. ЛП №02330/0131666 от 30.04.2004.
220013, Минск, П. Бровки, 6