А что такое микроконтроллер?

Можно начать рассказ о микроконтроллерах с рассмотрения архитектуры контроллеров разных производителей. Но я не очень люблю, хотя и понимаю важность терминов, терминологию, особенно заимствованную из других областей знаний.

Можно начать рассказ о микроконтроллерах с рассказа об устройстве микропроцессоров. Но для такого рассказа недостаточно одной главы.

А можно начать с рассказа о языках программирования и программах для работы с микроконтроллерами. Без этих составляющих работа с микроконтроллерами возможна только при условии, что микроконтроллер уже запрограммирован и его нужно только впаять в готовую схему.

Но и рассказ о языках программирования, а для написания кода программ используется не один язык, с учетом особенностей разных компиляторов займет слишком много места. Да и книг и по архитектуре микроконтроллеров, и о микропроцессорах, а, тем более, по программированию написано достаточно.

Однажды, замечу, что я не программист и хочу им казаться, однажды я обнаружил, что в программе KTechlab, она существует только для Linux, можно написать программу для микроконтроллера без знания какого-либо языка программирования. И из этой же программы запрограммировать микроконтроллер, используя самодельный программатор. Мне показалось, что подобный подход самый удобный для начинающих любителей. Пока не возникнут серьезные сомнения в правильности этой точки зрения, я буду придерживаться позиции — чем проще, тем лучше. Поскольку многие любители покупают компьютер с установленной на нем операционной системой Windows, я начну рассказ, используя программу FlowCode. Эта программа работает в Linux с помощью Wine, по крайней мере версии 1.0. Одним из достоинств программы — наличие версии и для PIC, и для AVR контроллеров. Мало того, можно импортировать программы, написанные для AVR в программу для PIC контроллеров. И, думаю, все, что написано ниже для PIC контроллеров, вполне можно применить для AVR контроллеров.

Основная особенность микроконтроллеров в написании программы, без которой микроконтроллер, купленный в магазине, остается беспомощен. Можно, правда, используя программу для работы с программатором, который тоже может быть куплен в магазине, «зашить» в него готовый загружаемый файл, благо готовых решений в Интернете можно найти множество. Но при таком подходе, когда микроконтроллер перестает быть беспомощным, беспомощным оказывается радиолюбитель.

Преодолеть эту беспомощность я предлагаю при посредстве программы FlowCode.

Ни одна из программ, когда либо написанных, не начиналась с написания кода программы на каком-либо языке или без использования языка программирования. Любая программа начинается с того, что автор старается понять, что и как должна делать программа. Кто-то мысленно и быстро проделывает эту работу, кто-то без спешки проделывает это на бумаге, многие сегодня делают это за компьютером.

Если не вдаваться в детали, то результатом подобных размышлений, в том или ином виде, будет алгоритм работы программы. Для записи алгоритма давно создан специальный графический язык, подобный языку записи электрических схем. Как и электрическая схема, соединяющая элементы цепи, так и эта программная схема соединяет программные элементы. Вот почему я считаю такой подход легче должен восприниматься радиолюбителями. Они давно привыкли, что многие функции в электрических схемах

осуществляют микросхемы — на бумаге квадратики, связанные с другими квадратиками, не более того. И программа для микроконтроллера не отличается во многом от привычного вида схемы.

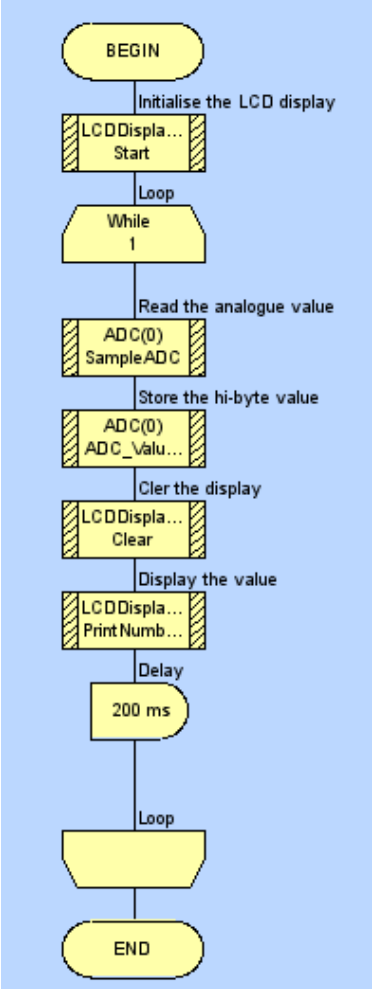

Рис. 6.1. Вид программы для микроконтроллера

Я уже делал попытки рассказать, как работать с этой программой, но впоследствии убедился, что многие вещи столь очевидные для меня, оказываются камнем преткновения для начинающих. Попробую сделать еще одну попытку, и пусть читающих этот рассказ не обидит отсутствие тех подробностей, которые они ищут (я их мог «проморгать»), и не обидит наличие тех подробностей, которые им очевидны (потерпите).

Прежде, чем перейти к рассказу о программе, я хочу сказать, что, скорее всего, за рамками этой истории окажутся такие аспекты, как работа с прерываниями, встроенными АЦП и USART, компараторами и таймерами, RF-модулями. Я уверен, что, начав освоение микроконтроллера, начинающий любитель не остановится на достигнутом и найдет, как использовать все возможности контроллеров.

Но сейчас мы будем смотреть на микроконтроллер, как на маленький компьютер с процессором внутри, огороженном устройствами, которые называются портами ввода-вывода. Когда мы включаем компьютер, мы мало думаем о том, что там внутри, и как оно устроено. Не будем этого делать и в отношении микроконтроллера. Для нас важно, что выводы портов могут по нашему желанию (это достигается при написании программы) служить цифровыми входами или выходами. На цифровые входы мы можем подать уровни

логического нуля или логической единицы, по нашему желанию (программному) на выходах появятся уровни логических нулей или логических единиц. Даже при таком, казалось бы, скромном использовании микроконтроллера можно создать много интересных схем.

Установка программы FlowCode в демо-версии сводится к обычным операциям установки любой программы в Windows, а в Linux при установленной программе Wine потребует либо точно таких же шагов, либо, скажем, запуска установки с помощью менеджера файлов Windows, это зависит от Linux дистрибутива. Если вы предполагаете использовать демо-версию, то на этом установка закончена, если используете какой-либо из вариантов регистрации, то соответствующие шаги будут описаны в приложении. Что можно увидеть при первом запуске программы?

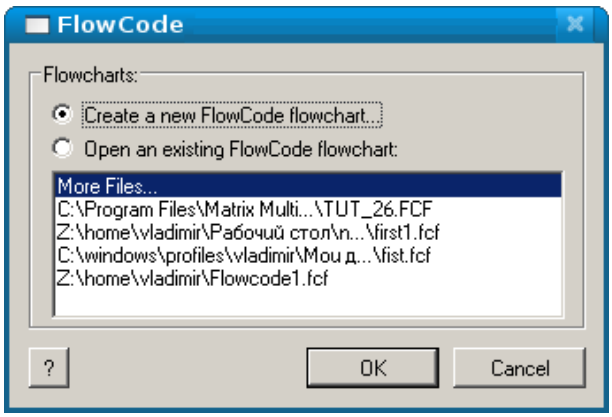

Рис.6.2. Диалоговое окно выбора проекта в FlowCode

В этом диалоговом окне можно создать новый проект (*Create a new FlowCode flowchart...*) или открыть уже существующий, выбрав его из списка ниже, или, используя опцию *Open an existing FlowCode flowchart*, открыть уже существующий проект для его модификации или использования.

Если вы выбираете новый проект, то в следующем диалоговом окне вам будет предложено выбрать модель микропроцессора из списка доступных.

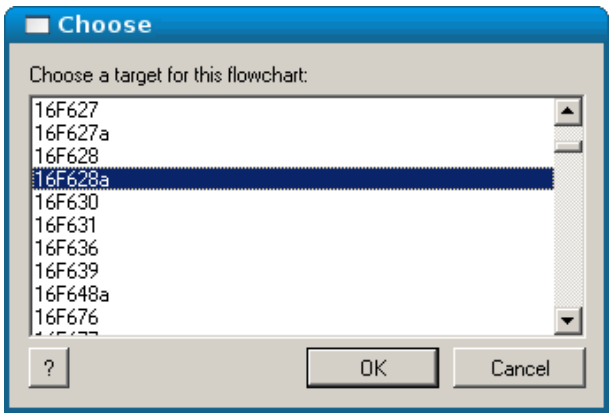

Рис. 6.3. Диалоговое окно выбора модели контроллера

Щелкнув по выбранной модели и кнопке **OK**, вы попадаете в рабочее окно программы.

Естественно, что всю центральную область занимает графический редактор, в котором будет «написана» программа. Естественно, что интерфейс программы достаточно стандартен

для всех программ Windows, и, кстати, для многих программ Linux, если вы никогда не видели этой операционной системы. Основное меню в верхней части окна, инструментальная панель работы с файлами чуть ниже и две инструментальных панели в левой части рабочего окна программы. Обе эти панели можно сместить в рабочую область окна, если вам это понравится больше, а я их сместил, чтобы иметь возможность рассказать о компонентах этих панелей. Закончив рассказ, верну их на место. В правом верхнем углу, заметьте, появляется внешний вид выбранной микросхемы с обозначением выводов, что бывает весьма полезно при работе с программой.

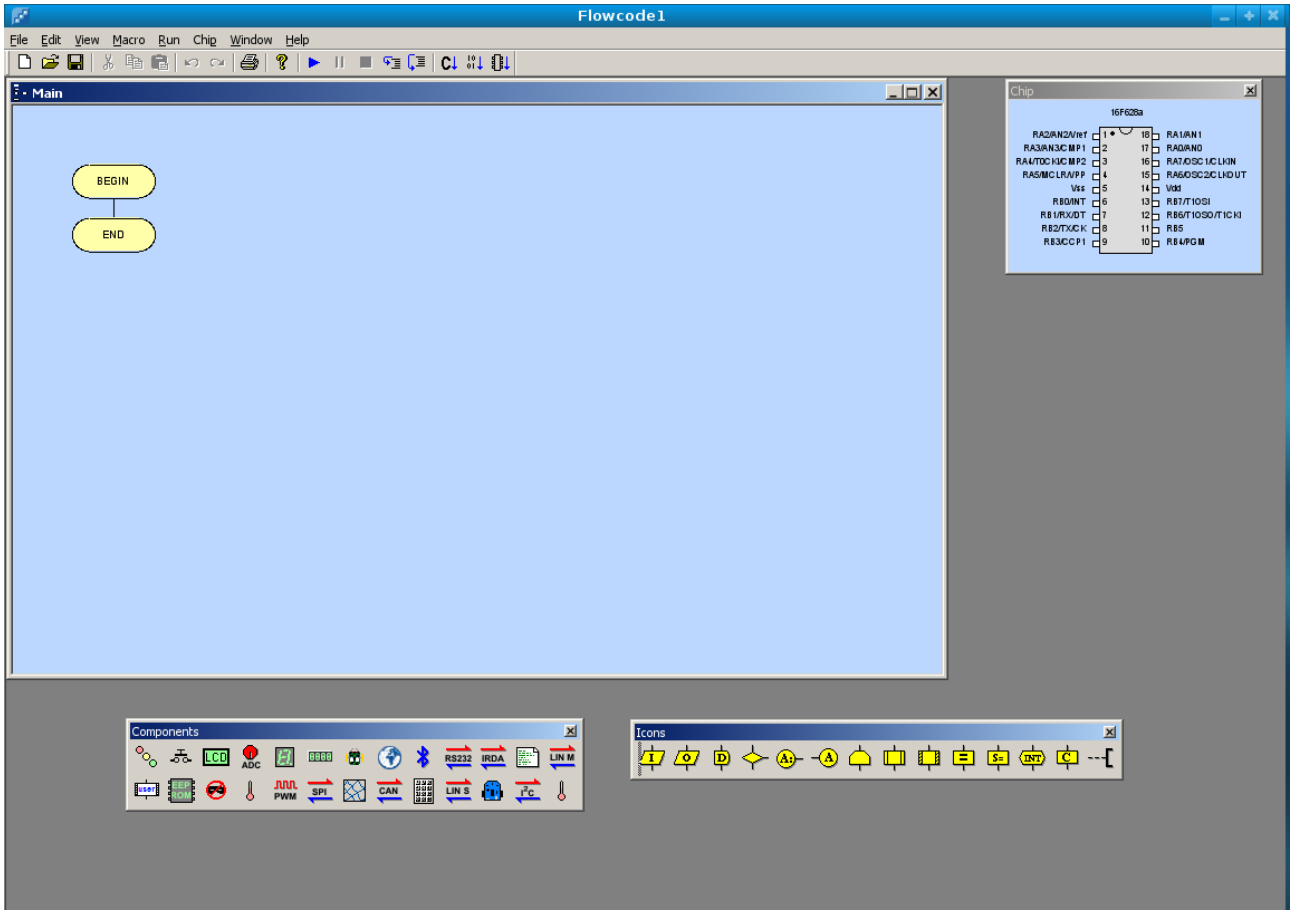

Рис. 6.4. Рабочее окно программы FlowCode

В окне графического редактора программы уже есть заготовка будущей программы: BEGIN-END. Если не ошибаюсь, в языке Pascal такая конструкция называется операторными скобками, в которую заключается и вся программа, и процедуры и функции языка. Позже подобную конструкцию мы увидим в цикле, но позже, а сейчас обратимся к основному меню:

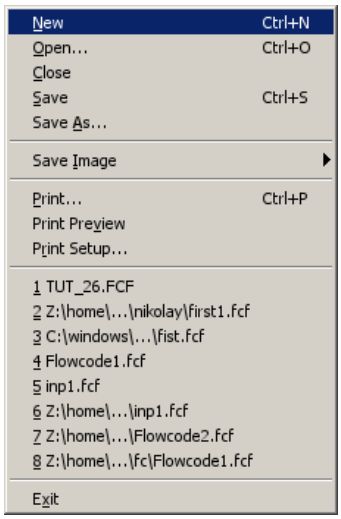

Рис. 6.5. Меню File основного меню

Стандартный набор операций с файлами: *New* — создать новый файл; *Open...* — открыть уже существующий; *Close* — закрыть файл; *Save* — сохранить файл; *Save As...* - сохранить как..., удобно на тот случай, если вы хотите сохранить файл под другим именем или в другой папке; следующий пункт выпадающего меню *Save Image*, позволяет сохранить вам программу в виде картинки в двух графических форматах, которые вы выбираете в открывающемся меню, если курсор наведен на этот пункт; пункты меню, начиная с *Print...*, относятся к выводу на печать, последний из них позволяет настроить принтер, а *Print Preview* предварительно увидеть, как будет выглядеть вывод на печать. Завершают этот список перечень недавно открывавшихся файлов и выход из программы *Exit*.

Следующий пункт основного меню *Edit* — редактирование.

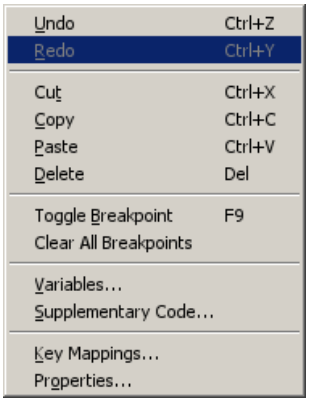

Рис. 6.6. Выпадающее меню редактирования (основное меню)

Я хочу подчеркнуть, что это вид выпадающего меню из списка основного меню, поскольку, щелкнув правой клавишей мышки в окне графического редактора, в зависимости от места, по которому вы щелкните, вы получите выпадающие меню для работы с элементами графики.

В этом меню первая половина относится к обычным операциям с объектами любого редактора: *Undo* и *Redo* — две взаимно противоположные команды по отмене последнего действия и возвращения его; *Cut* — вырезать; *Copy* — копировать; *Paste* — вставить; *Delete* — удалить. Два следующих пункта относятся к режиму отладки программы: *Toggle Breakpoint* — переключение точки останова (точка останова позволяет вам остановить

выполнение программы при отладке, чтобы, например, посмотреть значения переменных), если точка останова была включена (задана), то она выключится, и наоборот; *Clear All Breakpoints* — очистит вашу программу от всех точек останова, которые вы задавали.

Следующий пункт в меню *Variables...* открывает диалоговое окно, в котором вы можете определить переменные вашей программы, если они вам нужны. Программы всегда полны переменных, с которыми они и работают, посмотрим, как много переменных вам потребуется для написания вашей первой программы.

Следующий пункт *Supplementary Code...* (дополнительный код) позволяет при желании (или необходимости) добавить код к грфической программе, открывая окно диалога, где есть поле для объявления функции и поле для реализации этой дополнительной функции, что соответствует языковым конструкциям с разделением *Definitions and function declarations* (определение и объявление функции) и *Function implementations* (реализация функции).

В соответствующих окнах мини-редакторов можно ввести текст, который после нажатия клавиши **OK** позволит использовать эту дополнительную функцию. Возможность добавить нужные коды в программу значительно расширяет функциональность FlowCode для профессионалов, но и любителям, уже освоившимся в программе, достаточно хорошо владеющим языком программирования, это дополнение не покажется лишним.

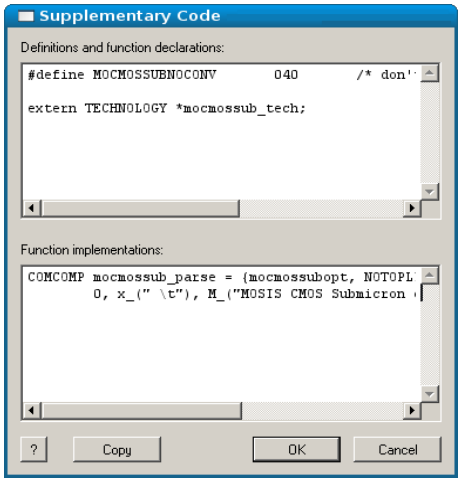

Рис. 6.7. Диалоговое окно добавления функций

Все выпадающие меню в программе «чувствительны» к происходящему в окне редактирования. И пока вы не добавите в программу клавиатуру (KeyPad) из набора дополнительных элементов, следующий пункт *Key Mappings...* (карта соответствий) не будет активен. Но с появлением клавиатуры в проекте вы можете изменить свойства этого дополнительного элемента.

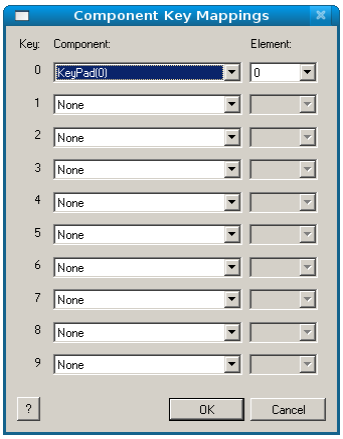

Рис. 6.8. Окно работы со свойствами клавиатуры

Аналогично поведение и последнего элемента списка из набора возможностей редактирования — *Properties...* (свойства). Чтобы «оживить» его, вам следует выделить любой элемент программы. Тогда, щелкнув по этому пункту меню, вы попадете в окно диалога свойств выбранного элемента. Вид этого диалога и предоставляемые пользователю возможности зависят от выделенного элемента. Так для элемента input (вход) можно определить его состояние, можно задать привязку к переменной и к порту, выбрать работу с одним битом или всем портом, задать маску, поставив галочку в поле *Use Masking*.

Каждый элемент программы в FlowCode имеет свой набор свойств, назначение и смысл которых яснее всего проявляются тогда, когда вы начинаете работать над своей программой. У вас обязательно появятся вопросы вида — как сделать так, чтобы в конкретном месте программы проверить состояние только одного бита порта, а не всех вводов? Например, когда вы дальше в программе хотите использовать ветвление, зависящее от состояния одного бита. Очень часто это проверка состояния «флага» (одного бита переменной или регистра, меняющего свое значение по окончании процесса или по результату операции).

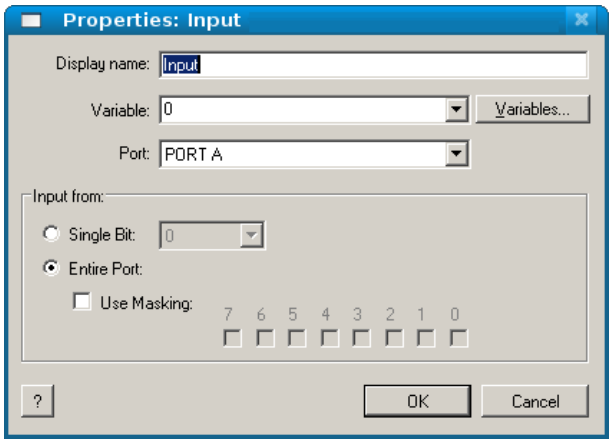

Рис. 6.9. Диалоговое окно свойств элемента программы input

Продвигаясь дальше по основному меню, вы можете открыть выпадающее меню *View* (вид).

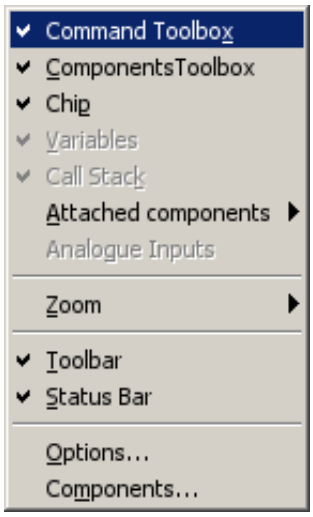

Рис. 6.10. Меню вид

«Галочки» слева показывают, какие компоненты вы желаете видеть при работе: *Command Toolbox* (инструментальное меню команд), *Components Toolbox* (инструментальное меню компонент), *Chip* (контроллер, с которым вы работаете), *Variables* (переменные, если они есть), *Call Stack* (стек вызовов подпрограмм), *Attached components* (присоединенные компоненты, если вы добавили клавиатуру или светодиоды), *Analogue Inputs* (аналоговые входы). Присоединенные компоненты появляются в виде дополнительного списка, на что указывает стрелка справа от пункта меню. Как и дополнительный список масштаба изображения *Zoom*, который вы можете выбрать, увеличить или уменьшить, привести к заполнению экрана или ширине окна редактирования.

Вы можете также включить или выключить *Toolbar* (основное инструментальное меню) и *Status Bar* (строку состояния). Вы можете изменить свойства проекта, такие как цвет рисунка и фона, шрифт, используя раздел *Options...,* или выбрать из предлагаемого списка, какие компоненты вам нужны в настоящее время для работы — *Components...* Все это позволяет вам выбрать комфортный в вашей работе вид программы: только нужные компоненты, цветовую гамму, возможность легко читать все надписи в элементах программы и т.д. Не забывайте только, что, выключив из списка компонентов что-то, вы не увидите этот компонент при следующем запуске программы, и, если сегодня он вам не нужен, то завтра может понадобится.

Следующие пункты основного меню я опишу кратко. Тому несколько причин — многое из этого вам понадобится не сразу, а, когда понадобится, вы будете разбираться в среде программирования микроконтроллеров FlowCode не хуже меня; рассказ о перечне пунктов меню к тому времени, когда вам понадобится воспользоваться чем-то, забудется, и легче самому сообразить, чем найти в тексте описания; да и не интересно начинать работу с программой, если ты о ней все уже знаешь, гораздо приятнее делать «свои маленькие открытия».

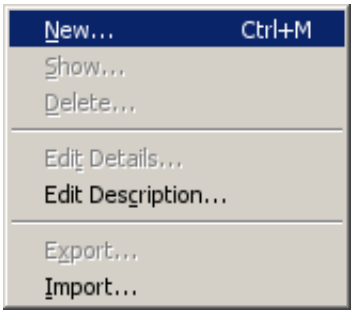

Рис. 6.11. Меню работы с макросами *Macro*

New... (новый макрос), *Show...* (показать макрос), *Delete...* (удалить), *Edit Details...* (редактировать детали), *Edit Description...* (редактировать описание, которое очень полезно делать), *Export...* (экспортировать), *Import...* (импортировать).

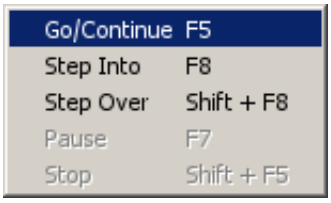

Рис. 6.12. Меню отладки *Run*

Go/Continue (запуск/продолжение), *Step Into* (шаг внутрь, например, функции), *Step Over* (шаг через, например, условие), *Pause* (пауза), *Stop* (стоп).

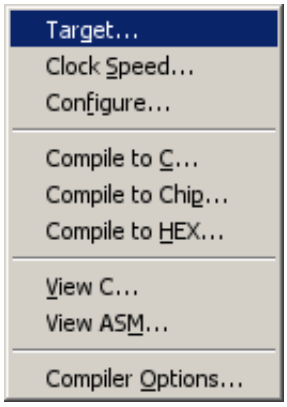

Рис. 6.13. Меню работы с контроллером Chip

Target... (выбор модели), *Clock Speed...* (частота тактового генератора, зависящая от требуемой), *Configure...* (слово конфигурации), *Compile to C...* (транслировать на Си), *Compile to Chip...* (транслировать все), *Compile to Hex...* (получить hex-файл загрузки), *View C...* (просмотр Си кода), *View ASM...* (просмотр ассемблерного кода), *Compiler Options...* (опции компиляции).

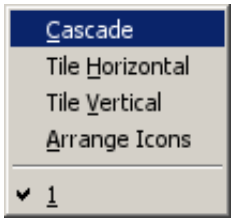

Рис. 6.14. Меню работы с окнами *Window*

Черепицей, горизонтально, вертикально, упорядочить иконки — вот, что можно сделать с окнами.

Последний пункт основного меню *Help*, на тот случай, если есть желание или необходимость обратиться к руководству по работе с программой.

В нижней части окна (справа) на рис. 6.4, а при запуске программы левая инструментальная панель — это панель команд.

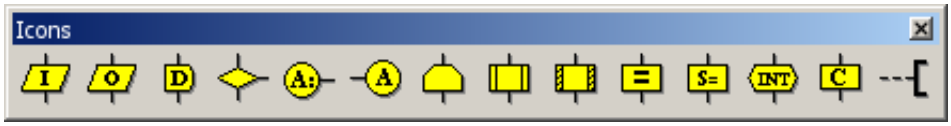

Рис. 6.15. Инструментальная панель команд

Перечень представленных команд (слева-направо на рисунке, сверху-вниз при запуске):

Input (ввод), *Output* (вывод), *Delay* (пауза), *Decision* (ветвление), *Connection Point* (две точки соединения), *Loop* (цикл), *Macro* (макрос), *Component Macro* (компонент макроса), *Calculation* (вычисление), *String Manipulation* (строковые операции), *Interrupt* (прерывание), *C Code* (блок кода на языке Си), *Comment* (комментарий).

Рис. 6.16. Инструментальная панель добавочных компонентов

Компоненты (слева-направо):

LEDs (светодиоды), *Switches* (переключатели), *LCDDisplay* (жидкокристаллический дисплей), *ADC* (АЦП, если есть порт АЦП), *LED7Seg1* (семисегментный индикатор), LED7Seg4 (блок из 4х семисегментных индикаторов), *Buggy* (компонент игрушки), далее несколько стандартных интерфейсов *TCP_IP, Bluetooth, RS232, IrDA, AddDefines* (добавить определения), *LinMaster* (ведущий в локальной сети), *Custom* (заказной компонент), *EEPROM* (перепрограммируемая память), *Alarm* (охранное устройство), *Thermometer* (термометр), *PWM* (широтно-импульсный модулятор), *SPI* (последовательный внешний интерфейс), *WebServer* (web-сервер), *CAN* (сеть абонентского доступа), *KeyPad* (клавиатура), *LinSlave* (ведомый в локальной сети), *FormulaFlowCode* (компонент игры), *I2C* (шина связи между ИС).

Использование программы FlowCode в профессиональном плане помогает экономить время на кодирование программы для микроконтроллера. Графическое построение

программы в FlowCode соответствует написанию алгоритма.

Для начинающих использование FlowCode полезно в двух проявлениях: можно, не зная языка программирования, создать свою программу для микроконтроллера, загрузить ее с помощью соответствующей программы в контроллер, получить работающее устройство и почувствовать, что ты можешь работать с микроконтроллерами; второе, что дает использование FlowCode — получение модулей, написанных, например, на языке Си, для использования в своей программе, которая будет отлаживаться и транслироваться, скажем, в MPLAB.

Если использовать программатор, который работает с FlowCode, то «прошивать» микроконтроллер можно непосредственно из FlowCode, как, впрочем, и из MPLAB, но это важнее для профессионалов, чем для любителей.

Итак. Программа FlowCode. Можно использовать демо-версию, но тогда будет ограничен выбор микроконтроллеров и будет появляться напоминание, что это демо-версия. Можно использовать «лекарство». В любом случае программу следует установить на компьютер. После запуска появляется окно диалога программы (на фоне основного окна):

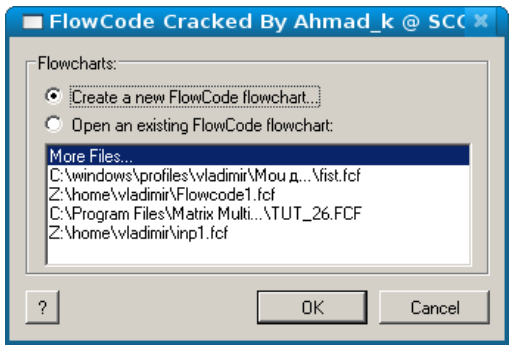

Рис. 6.17. Создание нового проекта

В этом диалоге можно создать новый проект (Create a new FlowCode flowchart), или открыть уже существующий (Open an existing FlowCode flowchart) или щелкнуть по существующему проекту в окне ниже. Создадим новый проект с названием first. Первое, что будет предложено в этом случае — выбрать микроконтроллер (16f628a). После чего появится окно редактора проекта:

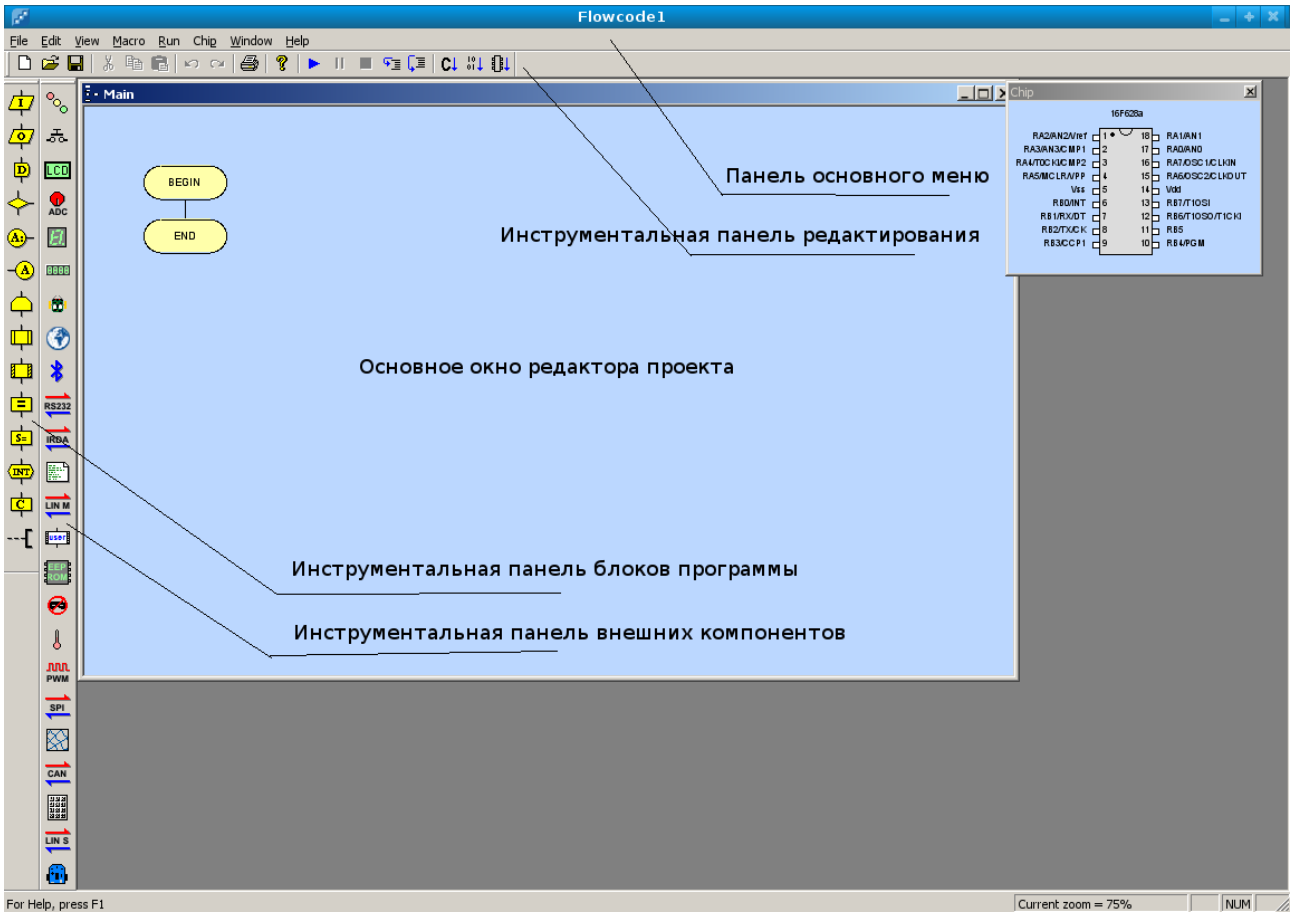

Рис. 6.18. Редактор и панели FlowCode

Посмотрим, как создавались программы «зажечь светодиод» и «помигать светодиодом» в этом редакторе. Расположение блоков программы на инструментальной панели блоков программы (сверху-вниз): ввод (input), вывод (output), пауза (delay), ветвление (decision), точка соединения (connection point), вторая точка соединения (парная), цикл (loop) и т.д. Расположение внешних компонентов на инструментальной панели внешних компонентов (сверху вниз): светодиоды (LEDs), переключатели (switches), дисплей (LCDDisplay), аналого-цифровой преобразователь (ADC), семи-сегментный идикатор (LED7Seg1) и т.д.

Первое, что нам нужно, это установить младший выход порта A в высокое состояние. Для этого нажмем левой клавишей мышки на вывод (out) инструментальной панели блоков программы и перетащим (не отпуская клавиши) блок к линии программы между блоками BEGIN и END.

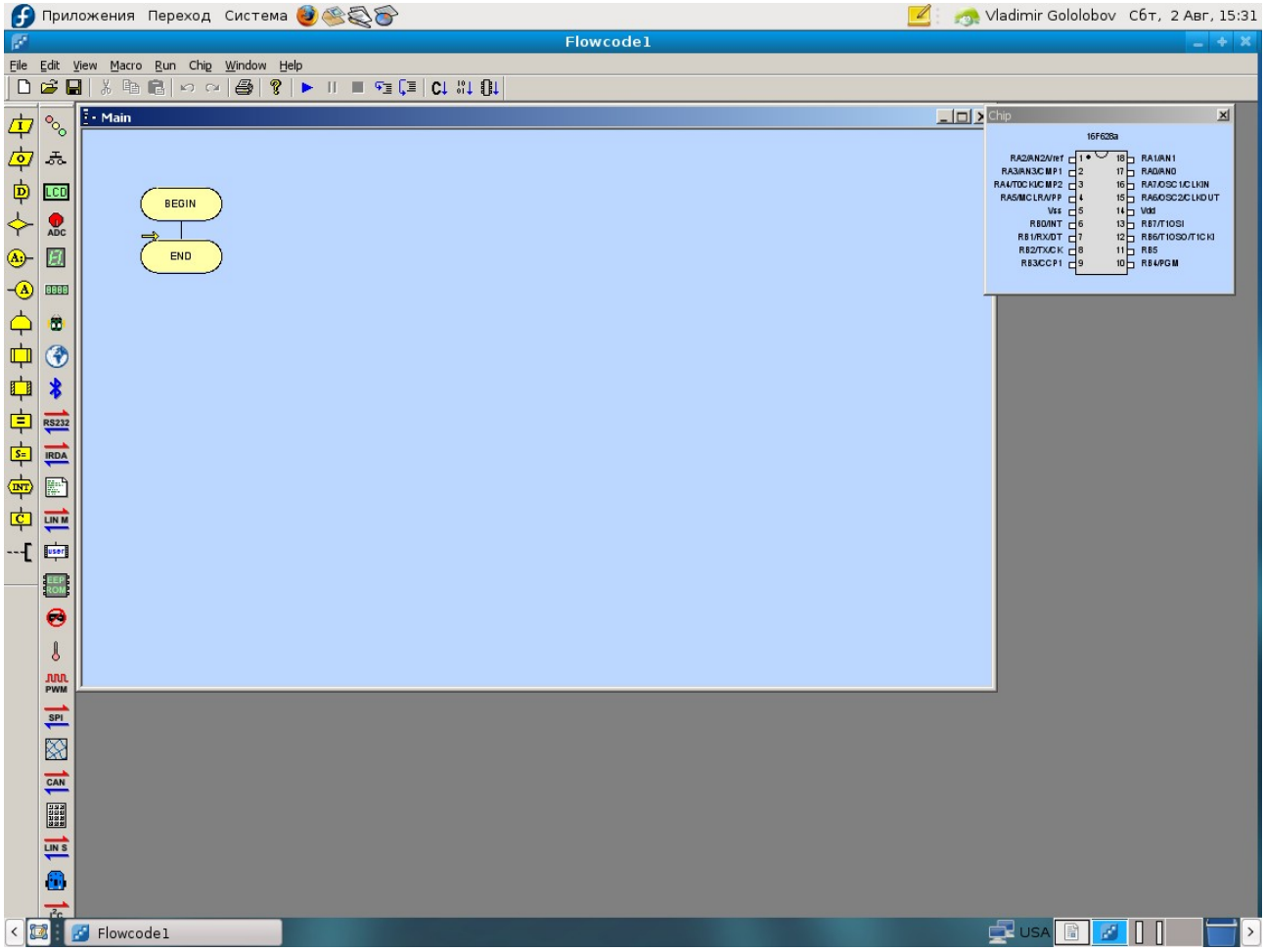

Рис. 6.19. Добавление элемента программы

После того, как мы отпустим клавишу, вид программы изменится.

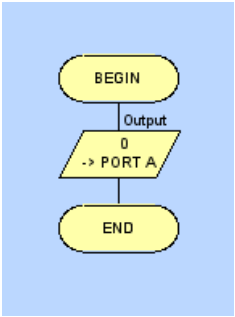

Рис. 6.20. Вид программы после первого шага

Двойным щелчком по элементу Output, появившемуся между BEGIN и END, мы открываем окно диалога, где можем изменить состояние порта A:

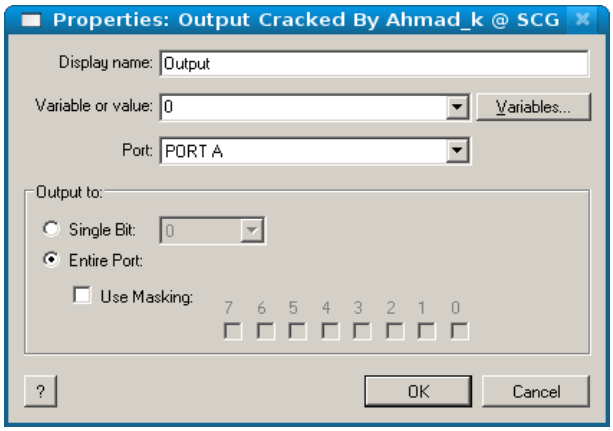

Рис. 6.21. Задание значения на выходе порта A

В окне с надписью Variable or value вместо «0» впишем «1». Этим мы устанавливаем младший вывод порта A в высокое состояние. Далее добавим паузу (delay), которую берем с инструментальной панели блоков программы, как мы это сделали с блоком output.

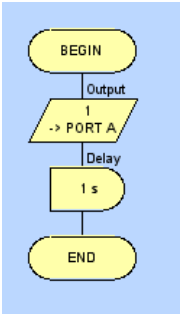

Рис. 6.22. Добавление паузы в проект

Двойной щелчок по новому блоку открывает диалоговое окно его свойств, где мы просто поменяем опцию «миллисекунд» на «секунд». Повторяя добавление еще одного блока output и еще одной паузы delay, мы приводим программу к окончательному виду:

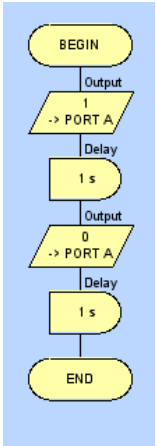

Рис. 6.23. Завершение работы над фрагментом программы

Написание программы для микроконтроллера закончено. Теперь можно, используя основное меню (File-Save As...), сохранить файл, затем, используя раздел Chip основного

меню:

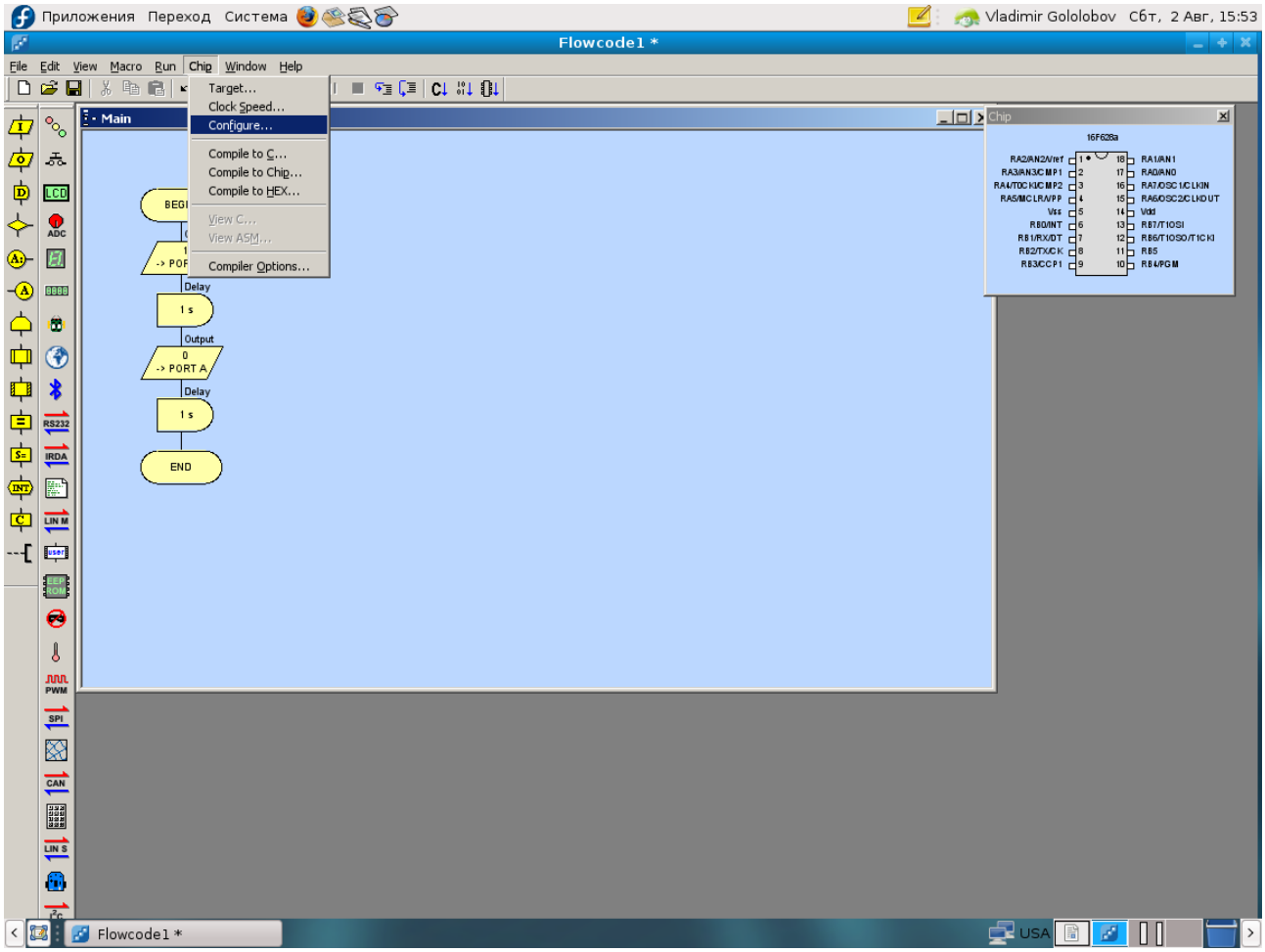

Рис. 6.24. Задание слова конфигурации контроллера

задать слово конфигурации контроллера (для этого служит PPP), оттранслировать на Си (Compile to C...) или получить загружаемый hex-файл (Compile to HEX...). Последний готов для «прошивки» микроконтроллера.

Но, прежде, чем программировать контроллер, займемся отладкой программы. Используем светодиоды, щелкнув по иконке светодиодов на панели внешних компонентов. Инструментальная панель редактирования изменяет свой вид и на ней появляется кнопка с иконкой, как у плеера для проигрывания. Нажав на эту кнопку, мы можем увидеть, что светодиод A0 зажигается и гаснет.

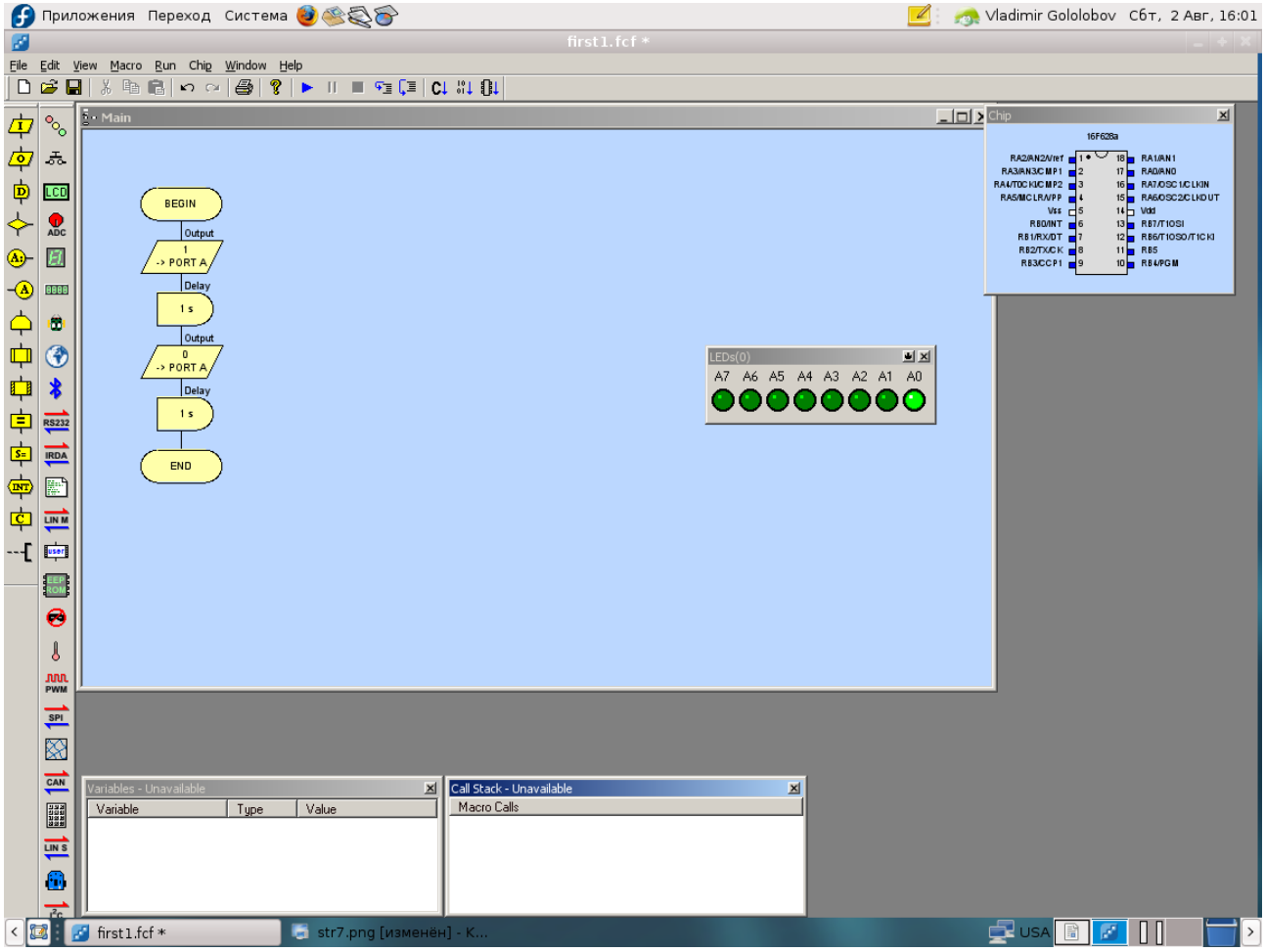

Рис. 6.25. Первая проверка

Следует иметь в виду, что в данный момент (по умолчанию) тактовая частота процессора 20 МГц, и если мы загрузим в контроллер hex-файл, а контроллер будет работать с внутренним тактовым генератором (частота 4 МГц), то 1 секунда изменит свою длительность.

Посмотрим, как нужно изменить программу, чтобы осуществить программу «мигающий светодиод». Перенесем в программу цикл (блок программы Loop с инструментальной панели программных блоков).

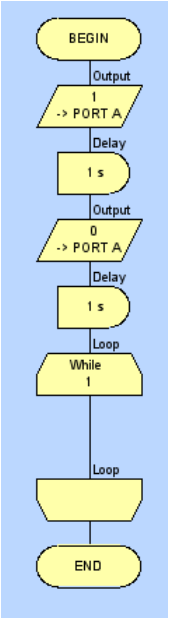

Рис. 6.26. Модификация программы, первый шаг

Затем выделим, используя нажатую левую клавишу мышки, нашу прежнюю программу, и перетащим с помощью левой клавиши мышки внутрь цикла while.

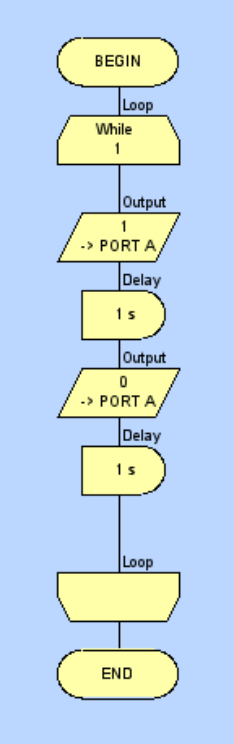

Рис. 6.27. Окончательное решение

Если теперь включить наш «проигрыватель» клавишей ►, то светодиод A0 будет мигать непрерывно, пока мы не нажмем клавишу ■ (Stop). А, если мы изменим свойства второго блока output нашей программы, записывая вместо «0» двойку (2), то...

Наличие на инструментальной панели внешних компонентов, таких как дисплей,

клавиатура, интерфейс RS232 и т.д., заставляет предполагать, что без знания языков программирования можно создать весьма и весьма сложные программы.