ОСВОЙ МИКРОКОНТРОЛЛЕР ЗА ОДИН ДЕНЬ! (BASCOM AVR)

Можно ли за один день научиться программировать микроконтроллеры?
НЕТ! - ответите Вы и будете правы. Это удел избранных и в лучшем случае мы останавливаемся на повторении готовых изделий. Я попытаюсь опровергнуть устоявшееся мнение и доказать, что программирование МК не такое уж сложное дело. Если за день Вы и не станете профессионалом, то гарантированно сможете пройти курс ускоренного обучения с теорией, практикой и готовой конструкцией, способной укрепить Вашу веру в собственные силы.
(Я предполагаю, что Вы имеете опыт пользователя ПК и владеете паяльником, а также в состоянии отличить Вольт от Ома и микросхему от трансформатора.)

Для начала программирования, Вам необходим IBM PC с установленным WINDOWS 98, 2000, NT, XP или Vista и доступ к Интернету. Далее, необходимо приобрести микроконтроллер, собрать простейший программатор и скачать необходимый софт.

Два года назад, уже программируя компьютерную периферию, я никак не мог решиться приступить к освоению микроконтроллеров. Останавливало всё: отсутствие должного уровня знаний, необходимость приобретения дорогостоящего программатора, изучение структуры микроконтроллера, выбор среды программирования, да и самого контроллера …
Я не профессиональный программист, но по роду работы и личному желанию назрел момент перехода на новый уровень – уровень микроконтроллеров. Тогда я занялся поиском, и выбор микроконтроллера для меня оказался очевиден:

высокопроизводительные 8-разрядные RISC микроконтроллеры семейства AVR.

Отличительные особенности:

Производительность, приближающаяся к 1 MIPS/МГц
Усовершенствованная AVRa RISC архитектура
Раздельные шины памяти команд и данных, 32 регистра общего назначения
Flash ПЗУ программ, с возможностью внутрисистемного перепрограммирования и загрузки через SPI последовательный канал, 1000 циклов стирание/запись
EEPROM данных, с возможностью внутрисистемной загрузки через SPI последовательный канал, 100000 циклов стирание/запись
Блокировка режима программирования
Встроенные аналоговый компаратор, сторожевой таймер, порты SPI и UART, таймеры/счетчики
Полностью статические приборы - работают при тактовой частоте от 0 Гц до 20 МГц
Диапазон напряжений питания от 1,8 В до 6,0 В
Режимы энергосбережения: пассивный (idle) и стоповый (power down).

Более подробно о микроконтроллерах семейства AVR можно почитать на http://www.gaw.ru/html.cgi/txt/ic/Atmel/micros/avr/about.htm.
Определившись с контроллером, я начал искать инструментальные средства проектирования.
Среди огромного количества компиляторов, программаторов, ассемблеров и отладчиков, предлагаемых как корпорацией Atmel, так и сторонними производителями, основное место занимают программные продукты на основе Ассемблера и Си. Оба этих языка довольно сложны для изучения и тем более «быстрого старта». Для тех, у кого нет возможности потратить несколько месяцев (или лет!?) на их освоение, нидерландская торговая марка BASCOM-AVR http://www.mcselec.com/ предлагает программную среду разработки для микроконтроллеров фирмы Atmel, основанную на языке программирования, близком к стандартному Бейсику (QBASIC) или VB (VISUAL BASIC). Простой интерфейс, лёгкая настройка, встроенные компилятор и программатор, построчный отладчик-симулятор (с программными эмуляторами терминала, символьного ЖКИ, матрицы клавиатуры, EEPROM и SRAM, светодиодами - пинами портов, ADC и компаратором). Возможность добавлять комментарии на русском языке, автоматическое подключение необходимых библиотек, использование ассемблерных вставок, поддержка многих стандартных внешних устройств (1WIRE, I2C, LCD, SPI, SOUND, RC, RC5, MMC и SD)- позволяет рекомендовать её как для начинающих пользователей, так и более искушенных. Для написания полноценной программы необходимо потратить не так уж и много времени, тем более тем, кто знаком с BASIC не понаслышке. Моё знакомство с этой чудо – программой началось со странички http://www.cqham.ru/gbc51rs.htm. Оказалось, что всего несколько строчек кода позволяют оживить разработанное устройство, то есть, получить ожидаемый результат с минимальными временными затратами. BASCOM AVR ни в коей мере не претендует занять место Си, или тем более заменить его, это совершенно разные программные продукты, с разными задачами и сферами применения.

Возможно, кто-то после освоения BASCOM перейдёт на Си, а кто-то продолжит добиваться профессионализма, но в любом случае этот компилятор должен занять достойное место среди начинающих программистов и радиолюбителей. К моменту написания статьи в разделе downloads на фирменном сайте можно скачать версию 1.11.8.3 demo.
http://www.mcselec.com/index.php?option=com_docman&task=doc_download&gid=139&Itemid=54
Ограничение кода в демо-версии – 4 килобайта, для наших экспериментов это не так уж и мало. Среди огромного количества аппноутов на сайте программы можно найти готовый пример на любой вкус, задать вопрос (English) на форуме. Сам язык и его особенности прекрасно отражены в 750 страничном мануале. BASCOM-AVR Manual version 1.11.8.3 PDF version of the BASCOM-AVR
http://www.mcselec.com/index.php?option=com_docman&task=doc_download&gid=140&Itemid=54

Почему я так рекламирую BASCOM? Да потому, что разработать и спаять схему, собрать программатор и написать свою первую программу бегущих огней с кнопочной регулировкой скорости, выбором эффектов и т.д., мне удалось всего за один выходной день. На следующий день появилась регулировка яркости светодиодов, а через некоторое время в связке с VB6 заливка эффектов через COM порт.

На западе BASCOM получил большое распространение, есть множество конструкций и форумов, а в русскоязычной части Интернета должного отражения до сих пор так и нет. Здесь я попытался собрать в одно целое разрозненную информацию, собранную из самых разных источников, чтобы можно было сделать первые шаги, остальное зависит от Вашего желания и настойчивости.

Блок питания для разрабатываемых устройств и программатора автор в течение нескольких лет использует от персонального компьютера, установив на одну из пятидюймовых заглушек предохранители, выключатели, контрольные светодиоды и удобные выходные зажимы для всех необходимых напряжений: -5В, +5В, -12В, +12В. Блок питания ПК имеет встроенную защиту от КЗ и перегрузок. Но соблюдение некоторых правил, а именно: подключение устройства к блоку питания и программатору, перепайка, переустановка микросхем в панельках и другие манипуляции, кроме измерений, необходимо проводить после отключения выключателем на заглушке соответствующего напряжения (при этом компьютер продолжает работать)! Это сбережёт Ваш компьютер и Ваши нервы. Кому покажется ненадёжным данный метод, можно порекомендовать любой источник постоянного напряжения на 5 вольт, за исключением дешёвых китайских адаптеров, которые не соответствуют ни одним нормам и в любой момент могут выдать завышенное напряжение или их чудовищные пульсации сведут на нет все попытки программирования и работы. На плате экспериментального устройства обязательно должны быть установлены блокировочные конденсаторы – электролитические и керамические в непосредственной близости от микроконтроллера, а ещё лучше – на его выводах питания.

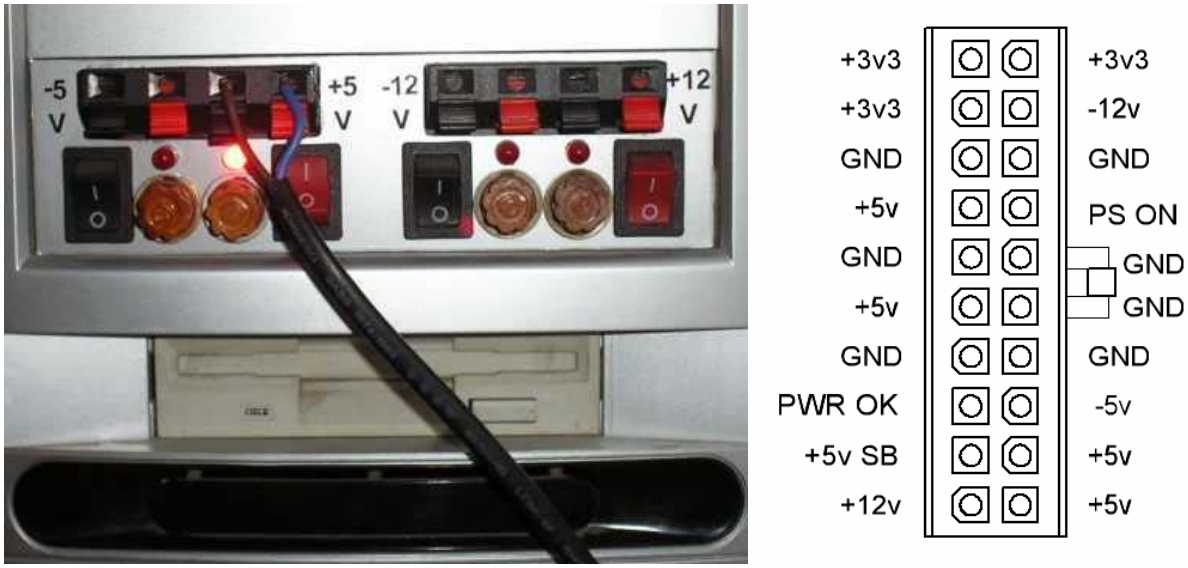

Рисунок 1 – Блок питания из IBM PC (ATX).

Среди множества существующих программаторов мы выберем простейший вариант для параллельного (LPT) порта, который можно спаять за несколько минут. Для этого необходимо в

разъёме XP1 с пластиковым корпусом DB-25M (папа) распаять четыре сопротивления R1-R4 и две перемычки - согласно схеме, а также вывести ленточный шлейф на разъём ISP XS1. Здесь сопротивления выполняют защитную функцию и уменьшают помехи импульсных сигналов. Длина соединительного кабеля должна быть как можно короче, а информационные сигналы желательно чередовать с земляными. Соединение выводов 2, 12 и 3, 11 позволяют программе обнаружить адаптер и провести его идентификацию как STK 200/300. Такое серьезное название в народе переименовали в «пять проводков».

Этот миниатюрный адаптер зарекомендовал себя с наилучшей стороны и позволяет работать практически на всех модификациях компьютеров. Более продвинутый программатор с буферной микросхемой можно найти в хэлпе к BASCOMу.

Рисунок 2 – Программатор в сборе.

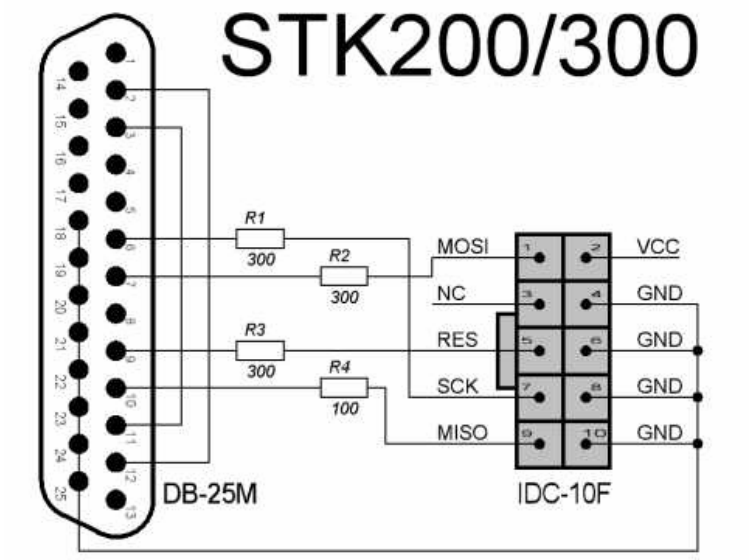

Рисунок 3 – Схема программатора.

Описание на русском языке 8-разрядного AVR-микроконтроллера с внутрисистемной программируемой флэш-памятью емкостью 128 кбайт ATMEGA128 можно найти здесь: http://www.gaw.ru/html.cgi/txt/doc/micros/avr/arh128/index.htm. Большинство материала фирменного перевода даташита ATMEGA128 можно применять и для младших моделей микроконтроллеров с архитектурой MegaAVR, сверяясь с их даташитами. Один из них, а именно ATMEGA8, мы и будем использовать в своих дальнейших экспериментах.

Рисунок 4 – ATMEGA8-16PU.

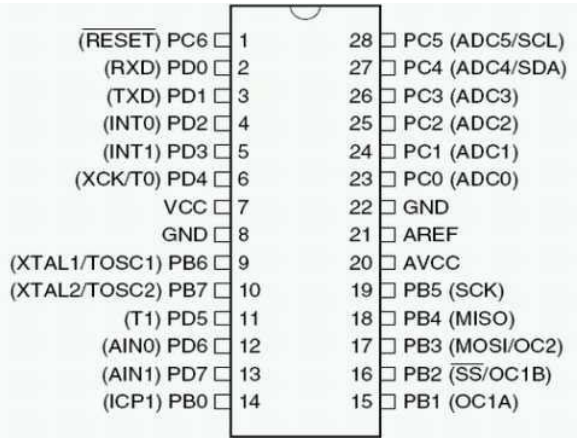

Рисунок 5 – Распиновка ATMEGA8.

Это современная модель в корпусе DIP-28 имеет функцию внутрисистемного программирования ISP (In-System Programming), поэтому для зашивки кодов программы её не требуется извлекать из платы и устанавливать в программатор. Компьютер через специальный адаптер-программатор подключается к разъёму ISP, установленному на плате конструкции. Процесс редактирования программы и прошивки кристалла происходит в считанные секунды!
В принципе, в наших экспериментах можно использовать любые МК семейства AVR, изменив схему (подключение питания и входы-выходы портов) сверяясь с даташитом, и введя незначительные изменения в коде программы. Но вначале я бы не рекомендовал этого делать, тем более что ATMEGA8 легче достать, чем например, несколько устаревшие и уже снятые с производства AT90S2313. Вводная статья по этому контроллеру есть на «Железном Феликсе» - http://www.ironfelix.ru/modules.php?name=Pages&pa=showpage&pid=111 – «Прошиваем МК с помощью BASCOM».

Помимо контроллера, нам необходимо подготовить ещё несколько деталей – конденсаторы, сопротивления, кнопочки и светодиоды. Не помешает макетная плата, хотя всю конструкцию можно собрать и на куске картона. Теперь, когда у нас подготовлены все вспомогательные устройства, следует собрать схему нашего «первенца».

Стартовым проектом будут самые обычные бегущие огни, но, тем не менее, на этой схеме мы сможем отработать начальные навыки программирования кристалла, освоим работу с портами ввода – вывода.

Давайте вначале рассмотрим назначение выводов микроконтроллера ATMEGA8:

VCC, AVCC - питание,
AREF - внутренний источник напряжения,
GND - общий,
PB0-PB7 - линии порта PB,
PC0-PC6 - линии порта PC,
PD0-PD7 - линии порта PD.

Следует отметить, что порты могут работать как на вход, так и на выход, причём в любой независимой комбинации, очень часто определяемой из удобства разводки печатной платы самим разработчиком. Нагрузочная способность линий порта ATMEGA8, настроенных на выход, обеспечивает 20 мА как на нагрузку, подключенную к цепи VCC, так и на нагрузку к цепи GND, т.е.

в обоих направлениях. Это довольно много, если учесть, что современные сверхяркие светодиоды рассчитаны именно на такой ток. Поэтому их можно подключать через токовые ограничительные резисторы без всяких дополнительных ключей, что мы и сделали, подключив четыре светодиода к младшим выводам порта PD0-PD3. Ещё нам необходимы две кнопки, которые запрограммируем на выбор режимов работы. В наличии осталось ещё много свободных выводов, и впоследствии их можно применять для дальнейшего усовершенствования конструкции.

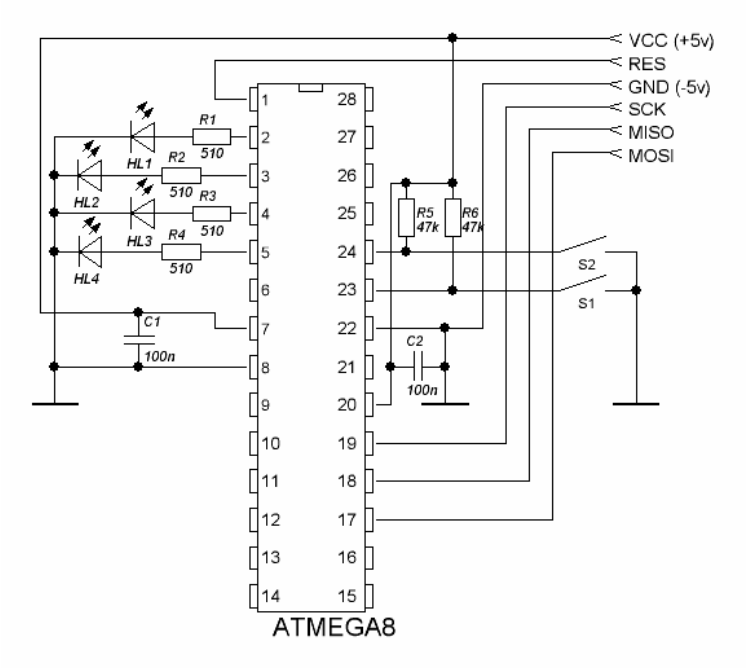

Рисунок 6 – Схема бегущих огней.

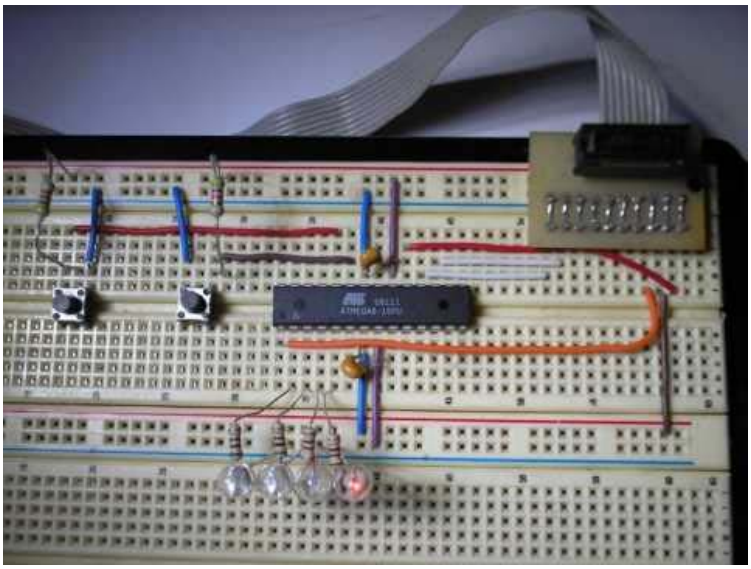

Рисунок 7 – Бегущие огни на монтажной плате.

Внимательно рассмотрев схему, мы увидим отсутствие кварцевого резонатора, присутствующего во всех МК проектах, но ошибки здесь нет. Дело в том, что синхронизация тактовых сигналов МК семейства AVR имеет множество режимов тактирования:

- от ВЧ кварцевого резонатора;
- от керамического резонатора;
- от НЧ кварцевого резонатора;
- от внутреннего RC – генератора;
- от внешней RC – цепочки;
- от внешнего импульсного генератора.

Эти режимы пользователь выбирает с помощью так называемых «фьюзов», выбирая оптимальную с точки зрения поставленной задачи синхронизацию. Следует учесть, что не все МК семейства AVR поддерживают эти режимы, это необходимо уточнять в каждом конкретном случае с помощью даташитов. В своей первой конструкции нам не важна высокая стабильность временных интервалов, поэтому мы используем внутренний перестраиваемый RC – генератор.

Заводские вставки соответствуют именно этому генератору на частоту 1МГц. Я рекомендую на первых порах пользоваться именно этим режимом, так как ошибка в установке фьюзов может привести годный контроллер в нерабочее состояние, и без специального, довольно сложного параллельного программатора, восстановить его работоспособность не представляется возможным.

Справочные данные на МК ATMEGA8-16 допускают питание 4,5…5,5 вольт при частоте 0-16 МГц, поэтому можно использовать не только стационарные источники питания, но и батарейки на соответствующее напряжение. ATMEGA8 имеет в своём составе двойное питание: «цифровое» - VCC, GND и «аналоговое» - AVCC и GND. Вариант нашего включения предусматривает оперирование только с логическими уровнями «0» и «1» на входах и выходах портов, поэтому вышеуказанные пары соединяют параллельно, заблокировав их керамическими конденсаторами 0,1мкф, которые должны быть в максимальной близости от выводов МК.

Теперь необходимо соединить одноимённые выводы нашего программатора и контроллера, чтобы иметь возможность оперативно изменять прошивку. На каждой плате контроллера очень удобно установить гребёнку IDC-10M, мало ли какое усовершенствование вздумается внести через некоторое время.

Итак, подготовительные работы закончены и следует приступить к самой интересной части. Вначале установим ранее скачанную программу BASCOM-AVR Demo Version 1.11.8.3 в каталог, предлагаемый программой по умолчанию.

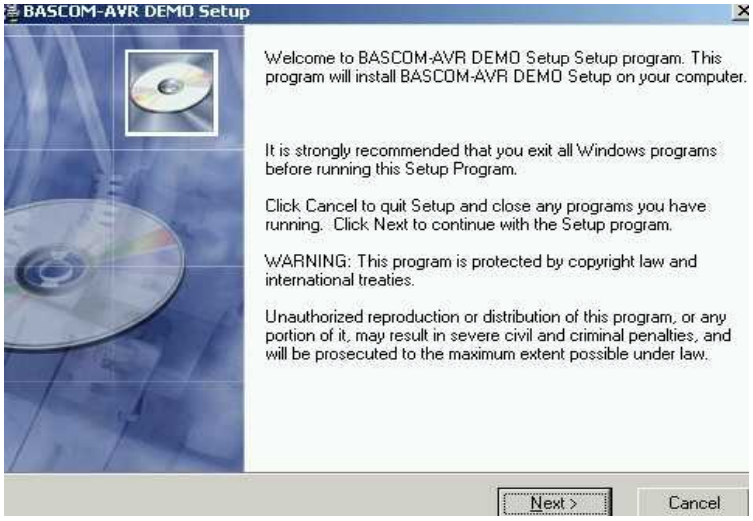

Рисунок 8 – Установка программы.

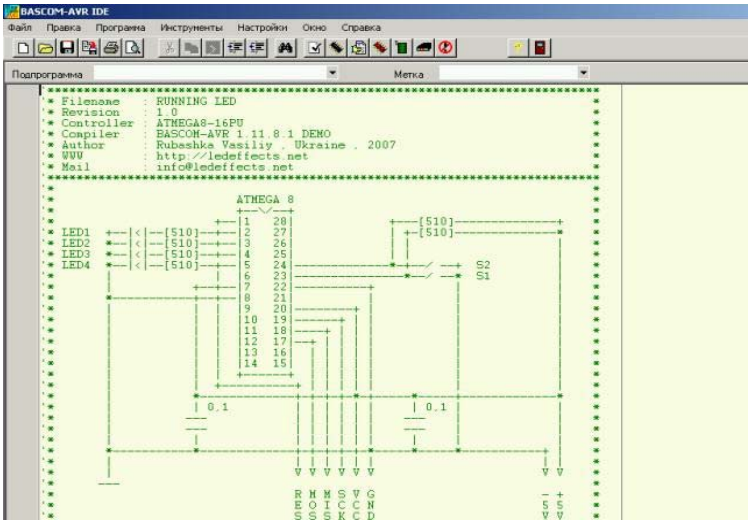

Рисунок 9 – Окно редактора BASCOM AVR.

Откроем новое окно (Create a new file, Ctrl+N) для редактирования нашей программы и введём код, используя для этого все доступные операции редактирования – отменить, восстановить, копировать, вставить и т.д. Заголовок программы следует снабдить подробными комментариями: название программы, дата создания, версия программы и версия компилятора, имя автора, по возможности структурная или принципиальная схема, выполненная элементами псевдографики, а также краткое описание предназначения программы и другую необходимую информацию. Со временем, у Вас накопится большое количество программ, и без этих сведений придётся вникать в суть программы по её коду, что очень неудобно, а порой даже невозможно. По максимуму старайтесь комментировать все элементы программы. Это облегчит жизнь не только тем, кто будет разбираться в Ваших исходниках, но и Вам самим, позволяя через определённое время свободно ориентироваться в коде программы, а не лихорадочно вспоминать, где же находиться тот или иной уникальный алгоритм. При компиляции программа пропускает все комментарии и поэтому они никоим образом не влияют на размер файла прошивки. Вообще, разработку микроконтроллерных устройств, можно условно разделить на несколько взаимосвязанных этапов:

- логический - с постановкой требуемой задачи и анализом возможных путей решения;
- физический - с определением конкретной модели МК исходя из необходимого количества линий ввода-вывода и других специфических условий, а также составление принципиальной схемы;
- программный - с составлением алгоритма, написанием кода и проверкой в симуляторе;
- практический - со сборкой, программированием МК, наладкой и проверкой в работе в самых неожиданных для прибора условиях, как со стороны окружающей среды, так и со стороны непредсказуемых действий пользователя.

Все этапы представлены в порядке их очерёдности, но могут быть и исключения. В нашем случае уже составлена и собрана принципиальная схема, поэтому исходя из этого, нужно придумать алгоритм работы и написать программу. Мы остановимся на таком простом алгоритме работы наших бегущих огней:

1) при включении все светодиоды мигнут для идентификации работоспособности МК;
2) при нажатии кнопки 1 бегущий огонь влево;
3) при нажатии кнопки 2 бегущий огонь вправо;
4) при нажатии обеих кнопок – мигание всех светодиодов.

Вот что у нас получилось:

```
'*******************************************************************************
'* Filename    : RUNNING LED                                                   *
'* Controller  : ATMEGA8-16PU                                                  *
'* Compiler    : BASCOM-AVR 1.11.8.1 DEMO                                      *
'* Author      : Rubashka Vasiliy , Ukraine , 2007                             *
'* WWW / Mail  : http://ledeffects.net / info@ledeffects.net                   *
'*******************************************************************************
'*                         ATMEGA 8                                            *
'*                         +--\/--+                                            *
'*                      +--|1   28|             +---[47к]--------------+       *
'* LED1  +--|<|--[510]--+--|2   27|             | +-[47к]--------------*       *
'* LED2  *--|<|--[510]--+--|3   26|             | |                     |       *
'* LED3  *--|<|--[510]--+--|4   25|             | |                     |       *
'* LED4  *--|<|--[510]--+--|5   24|-------------*-+--/ --+  S2         |       *
'*       |              |  |6   23|---------------*--/ --*  S1         |       *
'*       |           +--+--|7   22|----------+           |             |       *
'*       *-----------+--+--|8   21|          |           |             |       *
'*       |           |  |  |9   20|--------+ |           |             |       *
'*       |           |  |  |10  19|------+ | |           |             |       *
'*       |           |  |  |11  18|----+ | | |           |             |       *
'*       |           |  |  |12  17|--+ | | | |           |             |       *
'*       |           |  |  |13  16|  | | | | |           |             |       *
'*       |           |  |  |14  15|  | | | | |           |             |       *
'*       |           |  |  +------+  | | | | |           |             |       *
'*       |           |  +-----------+ | | | | |           |             |       *
'*       |           *----------------+-+-+-+-*-+-----*-----+-------------*       *
'*       |           | 0,1            | | | | | |     | 0,1 |             |       *
'*       |          ---               | | | | | |    ---    |             |       *
'*       |          ---               | | | | | |    ---    |             |       *
'*       |           |                | | | | | |     |     |             |       *
'*       *-----------*----------------+-+-+-+-+-*-----*-----*-----------+ |       *
'*       |                            | | | | | |                       | |       *
'*      ---                           V V V V V V                       V V       *
'*                                    R M M S V G                       - +       *
'*                                    E O I C C N                       5 5       *
'*                                    S S S K C D                       V V       *
'*                                      I O                                     *
'*******************************************************************************
```

```
$regfile = "m8def.dat"                          'установка модели микроконтроллера
$crystal = 1000000                              'частота генератора 1МГц
Config Portd.0 = Output                         'настройка пина d0 на выход (LED1)
Config Portd.1 = Output                         'настройка пина d1 на выход (LED2)
Config Portd.2 = Output                         'настройка пина d2 на выход (LED3)
Config Portd.3 = Output                         'настройка пина d3 на выход (LED4)
Config Portc.0 = Input                          'настройка пина c0 на вход (S1)
Config Portc.1 = Input                          'настройка пина c1 на вход (S2)
Dim Effect As Byte
Dim Temp As Byte
Temp = 300
'инициализация светодиодов
Portd = 15                                      '@@@@'
Waitms Temp
Portd = 0                                       'OOOO'
Waitms Temp
'-------------------------------------------------------------------------------
Do                                              'начало цикла
'опрос кнопок и присвоение переменной Effect соответствующего значения
If Pinc.0 = 0 Then Effect = 1
If Pinc.1 = 0 Then Effect = 2
If Pinc.0 = 0 And Pinc.1 = 0 Then Effect = 3
'бегущий огонь вправо, если Effect = 1
If Effect = 1 Then
Portd = 1                                       '@OOO'
Waitms Temp
Portd = 2                                       'O@OO'
Waitms Temp
Portd = 4                                       'OO@O'
Waitms Temp
Portd = 8                                       'OOO@'
Waitms Temp
End If
'бегущий огонь влево, если Effect = 2
If Effect = 2 Then
Portd = 8                                       'OOO@'
Waitms Temp
Portd = 4                                       'OO@O'
Waitms Temp
Portd = 2                                       'O@OO'
Waitms Temp
Portd = 1                                       '@OOO'
Waitms Temp
End If
'мигание, если Effect = 3
If Effect = 3 Then
Portd = 0                                       'OOOO'
Waitms Temp
Portd = 15                                      '@@@@'
Waitms Temp
End If
Loop                                            'конец цикла
'-------------------------------------------------------------------------------
```

Вышеприведённый листинг необходимо скопировать и вставить в новый проект.

Это вполне работоспособная программа, но она не лишена недостатков. Самый главный – это неудачный и даже можно сказать неправильный опрос кнопок. Как известно, кнопкам свойственен дребезг контактов, и чтобы не было многократных непредсказуемых переключений, с этим приходится бороться. В нашем случае одна кнопка выполняет одно действие – выбор одного эффекта, поэтому дребезгом мы можем пренебречь. Если вы решите усовершенствовать программу, например, одной кнопкой переключать по кольцу программы, а второй регулировать скорость, без подавления дребезга контактов не обойтись. В BASCOM есть встроенная функция – DEBOUNCE, которая производит периодическую проверку состояния битового порта, и там уже программно устранён эффект дребезга.

Я настоятельно рекомендую пользоваться хэлпом к программе, в нём всё очень хорошо расписано и можно найти ответы на множество возникающих вопросов.

Второй нюанс с кнопками – их опрос происходит вначале цикла, поэтому время реакции на нажатие напрямую зависит от длины выбранного эффекта. Как это устранить, я предлагаю подумать Вам в качестве домашнего задания.

Пожалуй, пора от скучной теории переходить к практике. У нас есть листинг программы, который понятен (или не очень) нам, но совсем непонятен микроконтроллеру. Давайте сделаем всё наоборот (Compile program F7)! Исходники программы компилируются в машинные коды микроконтроллера. Теперь мы ничего не понимаем, глядя на абракадабру шестнадцатеричных цифр (немного забежав наперёд, можно посмотреть в окне программатора), зато их очень любит наш друг

контроллер. При удачной компиляции на секунду выведется окно с сообщением о процентах используемой памяти, если же обнаружатся ошибки, внизу редактора появится нумерованный построчный список, где их следует искать.

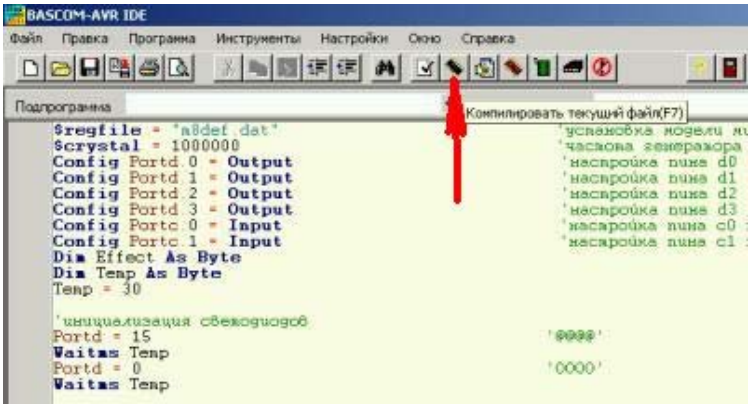

Рисунок 10 – Компилируем программу.

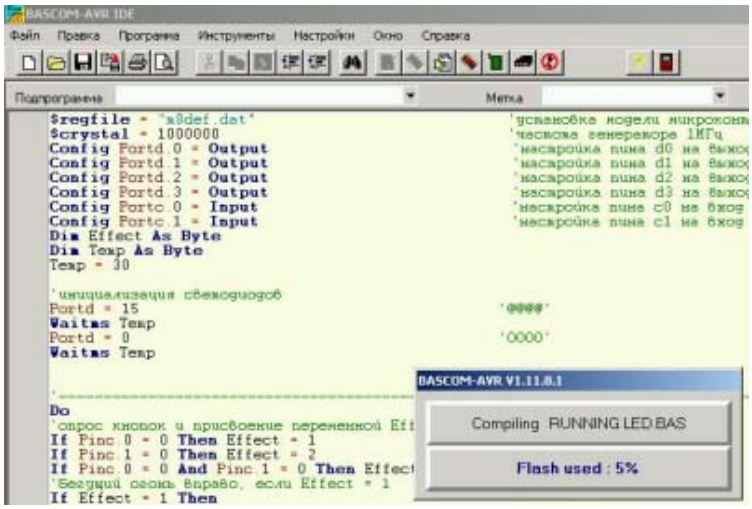

Рисунок 11 – Компиляция успешна!

Наша программа занимает всего 5 процентов памяти, но она уже работает! 95 оставшихся процентов так и просят, чтобы их заполнили. Что можно предложить в нашем случае? Увеличение числа каналов за счёт использования дополнительных портов, увеличение количества эффектов, введение регулировки скорости, автоматический – ручной режим работы и т. д. Огромное поле деятельности, причём ничего не надо перепаивать, изменил или добавил код, перекомпилировал, запрограммировал и любуешься результатом.

Теперь необходимо выполнить последний, завершающий шаг – программирование контроллера. В меню настроек выберем программатор и порт, к которому он подключен(Options-Programmer), как на рисунке.

Запускаем программатор, и если нет ошибок, он должен автоматически определить тип вашего микроконтроллера и вывести в рабочее окно скомпилированные машинные коды. Нажимаем на заветную кнопочку (auto program chip – ПУСК) и через несколько секунд ваши светодиоды радостно мигнут, приветствуя с первой победой разума над железом! Понажимайте кнопочки на плате бегущих огней, проверьте работу схемы в разных режимах, подумайте, как её можно улучшить.

Теперь Вы наверняка сможете повторить эту конструкцию, усовершенствовать её, внести необходимые изменения и дополнения, а также разработать совершенно новую, уникальную, необходимую для Ваших собственных нужд.

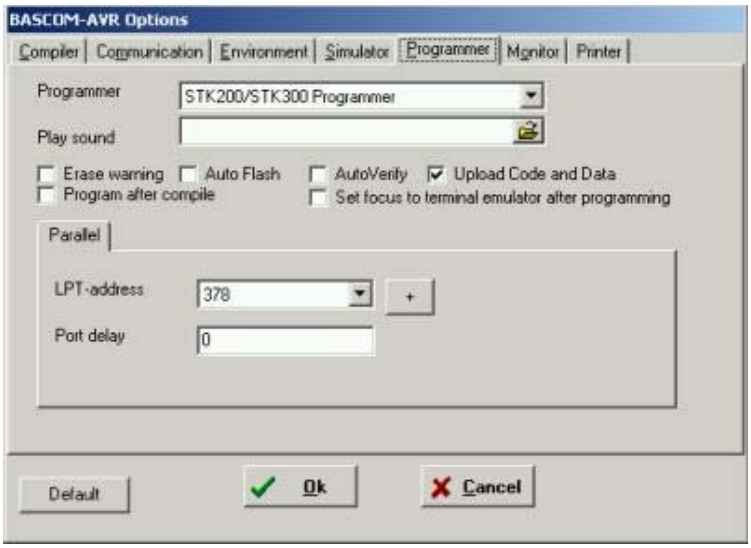

Рисунок 12 – Настройки программатора.

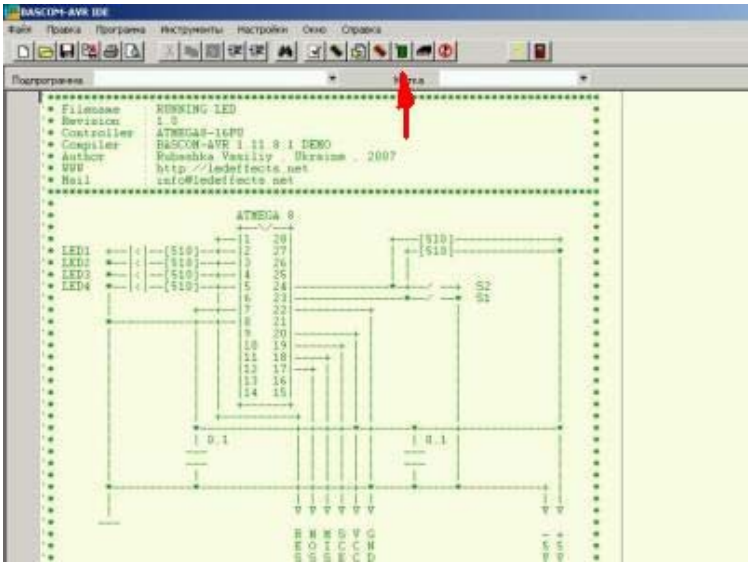

Рисунок 13 – Запускаем программатор.

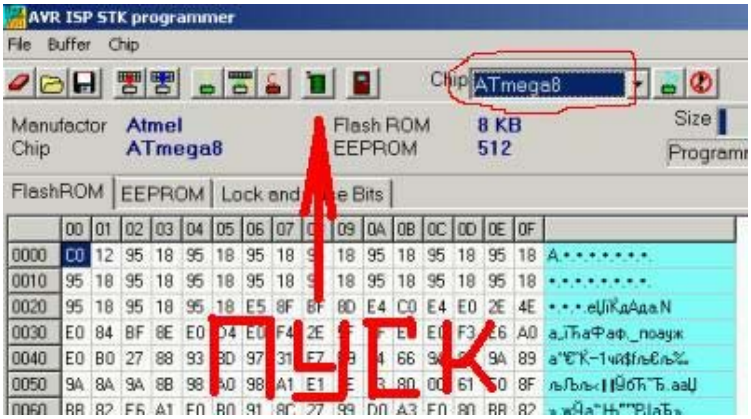

Рисунок 14 – Программируем контроллер!

Я надеюсь, что у Вас всё получилось, а если нет – милости прошу в форум «Программирование ATMEL в BASCOM» - http://bascomavr.3bb.ru и на страничку сайта «Светодиодные динамические эффекты – BASCOM» - http://ledeffects.net/pages/?id=12. Много полезного Вы сможете почерпнуть из http://www.mcselec.com/index.php?option=com_content&task=category&sectionid=7&id=79&Itemid=57 примеров – application notes.

P.S. Я, как и любой человек, могу ошибаться, за что заранее приношу извинения. Если у Вас есть замечания или предложения, прошу писать на info@ledeffects.net.