

**И.П.Матвеев**, кандидат технических наук, доцент  
кафедры АСУП Белорусского государственного  
аграрного технического университета

## **Методика изучения микроконтроллеров AVR**

В последние годы микроконтроллеры AVR приобрели большую популярность, привлекая разработчиков удобными режимами программирования, доступностью программно-аппаратных средств поддержки и широкой линейкой выпускаемых типов. Микроконтроллеры AVR представляют удобный инструмент для создания современных высокопроизводительных и экономичных встраиваемых контроллеров многоцелевого назначения. В частности, они используются в автомобильной электронике, бытовой технике, сетевых картах и материнских платах компьютеров, мобильных телефонах и т.д. [1].

Однако изучение реальных контроллеров оказывается затратной задачей, так как недостаточно только написать программу в определенной среде, необходимо с помощью программатора «прошить» процессор, т.е. записать в него разработанную программу, подключить к выходу контроллера исполнительные устройства и только тогда наглядно увидеть результат своей работы. А если что-то пошло не так, следует все повторить заново, но количество «прошивок» ограничено.

Поэтому изучение контроллеров удобнее и дешевле проводить виртуально, без паяльника или макетной платы, достаточно использовать программу **Proteus v7.7**.

Чтобы начать писать программы, нужно установить интегрированную среду разработки **AVR Studio 6**.

**AVR Studio 6** предоставляет возможность осуществлять разработку и отладку программ для микроконтроллеров AVR фирмы ATMEL, поддерживает большое количество средств программирования и отладки.

Программы пишутся на языке ассемблер (Assembler), поддерживается также язык программирования Си.

Ассемблер – это инструмент, с помощью которого создаётся программа для микроконтроллера. Ассемблер транслирует ассемблируемый исходный код программы в объектный код, который может использоваться в симуляторах или эмуляторах AVR. Также ассемблер генерирует код, который может быть непосредственно введен в программную память микроконтроллера.

При работе с ассемблером нет необходимости в непосредственном соединении с микроконтроллером.

Приведем методику работы с AVR Studio 6 и Proteus v7.7.

### Создание проекта в AVR Studio 6.

1. Запустить программу и после появления стартового окна в левом верхнем углу кликнуть New Project (рис.1).

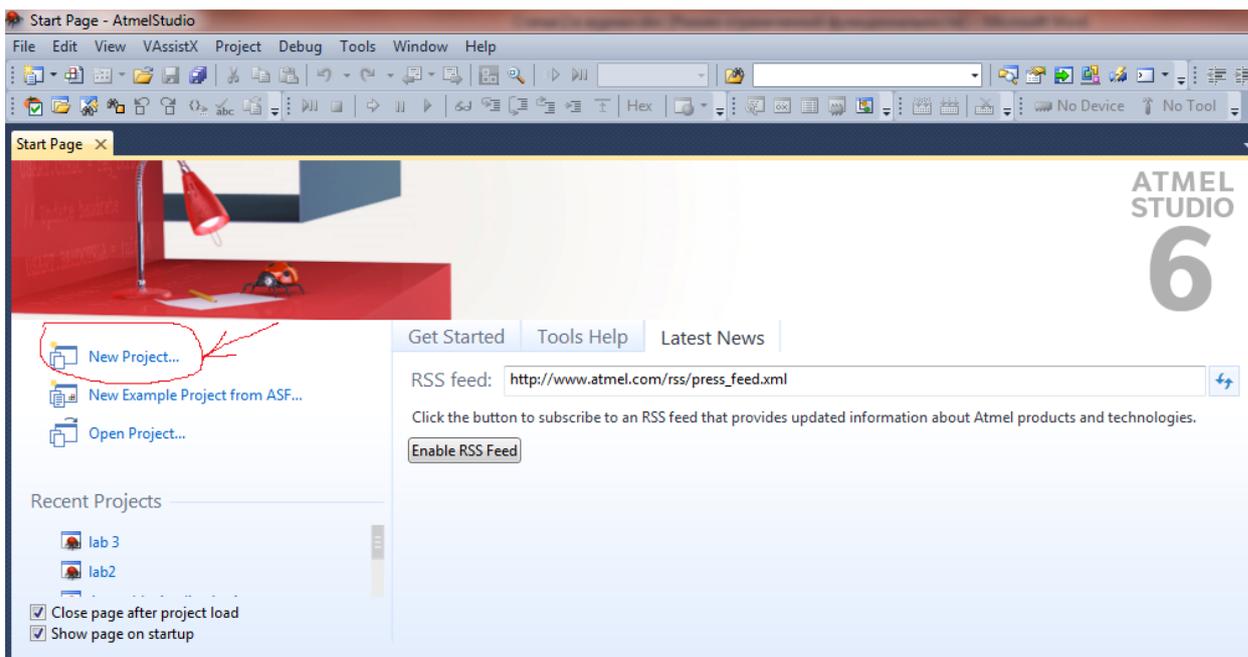


Рис.1 Стартовая страница AVR Studio 6

2. Выбрать язык программирования AVR Assembler, кликнув наверху вкладку AVR Assembler Project (рис.2).

В строке Name указать имя проекта, используя латинский алфавит (например, Primer1).

В строке Location - путь и место хранения файлов.

Имя проекта, введенное в строке Solution name будет выводиться в меню при старте.

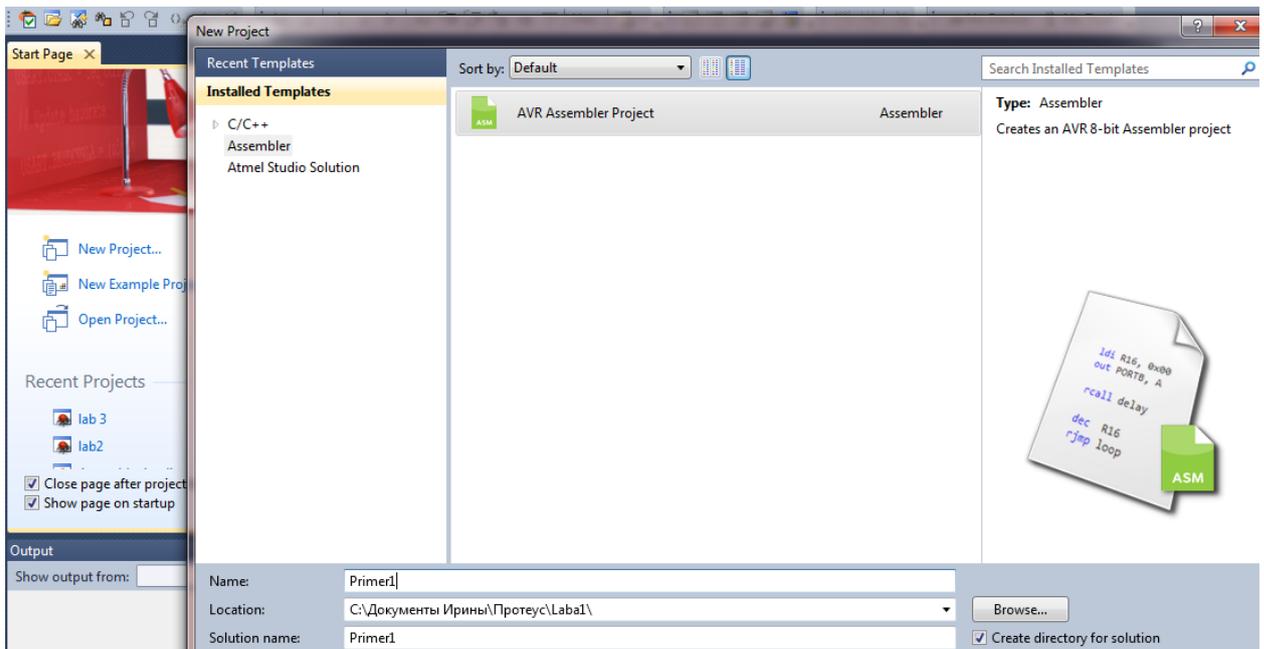


Рис.2 Выходные данные создаваемого проекта

Если что-то пошло не так при создании проекта или при открытии ранее созданного проекта можно воспользоваться стандартным путем: на панели Menu выбрать File – New или Open – Project (рис.3).

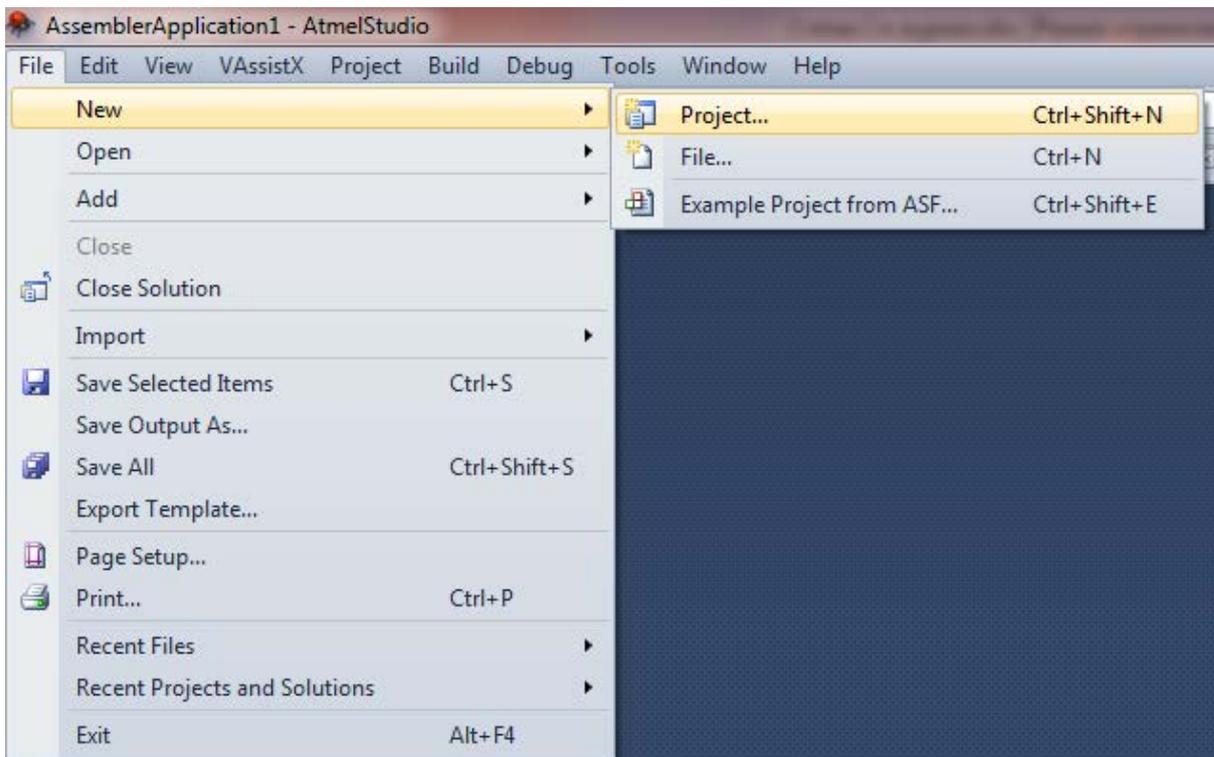


Рис.3 Создание или открытие проекта через панель Menu

После клика ОК появляется окно выбора микроконтроллера (**Device Selection**) (рис.4).

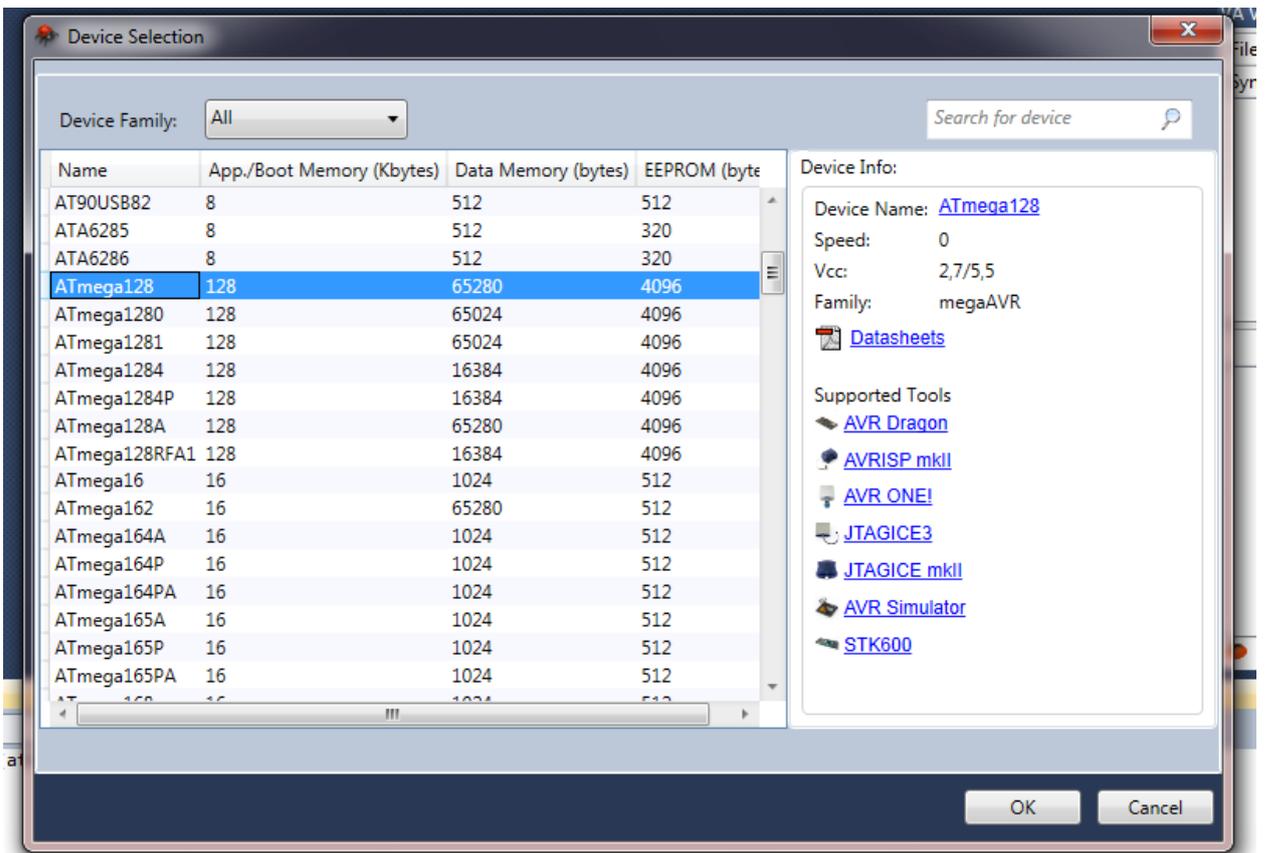


Рис.4 Окно выбора типа микроконтроллера

3. Выбрать тип микроконтроллера (например, ATmega128) [2], нажимаем ОК, после чего появляется страница редактора для написания программы на Assembler (рис.5).

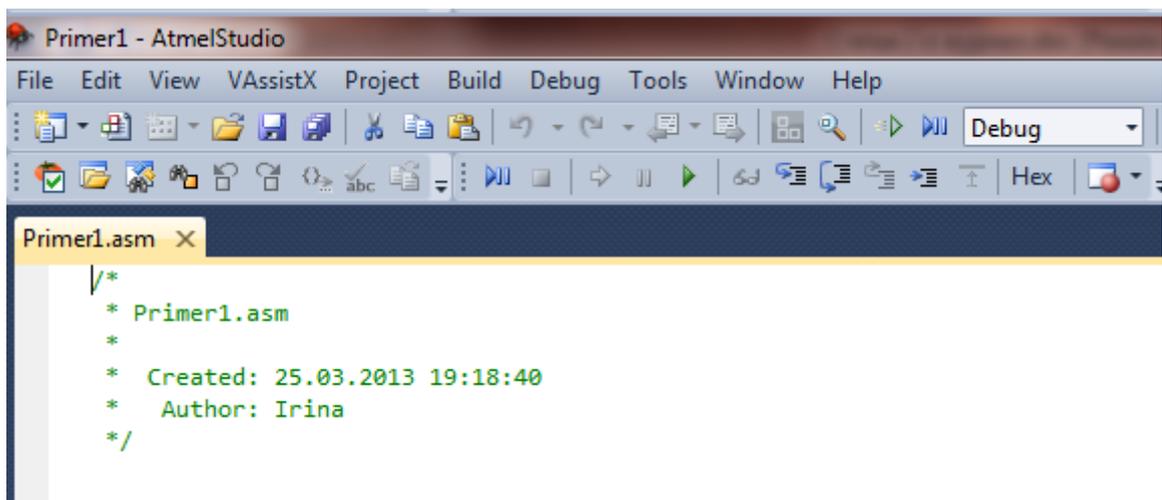


Рис.5 Страница редактора для написания программы на Assembler

4. Написать программу. Для этого необходимо предварительно изучить наборы команд Assembler и знать постановку задачи, т.е. что мы хотим получить на выходе микроконтроллера. Нам нужно чтобы микроконтроллер принял информацию, обработал по заданному алгоритму и

выдал результат в понятной для нас форме. В простейшем случае, чтобы увидеть результат работы микроконтроллера, к его выходным портам подключают светодиоды, которые должны загораться в соответствии с алгоритмом [3].

Пример текста программы приведён ниже. В этой программе через порты В и D контроллера ATmega128 устанавливается набор заданных сигналов (11001100), т.е. производится включение соответствующих диодов.

```
/*
 * Primer1.asm
 *
 * Created: 25.03.2013 19:18:40
 * Author: Irina
 */
.def temp=r16 ; директива
;=====
; Начало программы
.cseg ; директива
.org 0 ; начало первой строки программы
rjmp Start ; относительный переход к метке Start
; =====
Start:
ser temp; устанавливает все биты регистра temp в 1
out DDRB,temp; переводит все биты
out DDRD,temp; порта В и D на вывод
clr temp; обнуляет регистр temp (устанавливает все биты регистра temp
в 0)
out PortB,temp; отключает подтягивающие резисторы
out PortD,temp; портов В и D
Cicle:
ldi temp,0b11001100; включает светодиоды
out PortB, temp; порта В
rjmp Cicle; Возвращаемся к метке Cicle, зацикливаемся
```

5. Провести компиляцию программы, используя кнопку Debug – Start

Without Debugging или кнопку , как указано на рис.6. Суть работы компилятора заключается в переводе письменных символов, понятных для человека, в машинный код (в код нулей и единиц) и создание нового файла с расширением .hex.

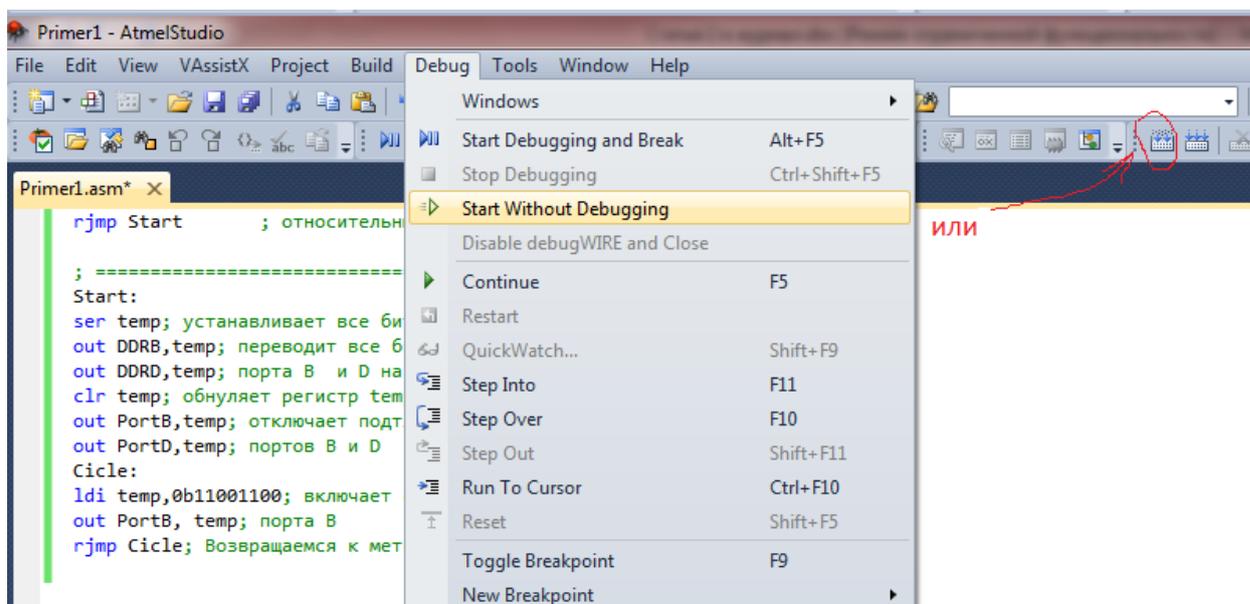


Рис.6 Проведение компиляции программы

В окне Output появятся информация о проведенной компиляции, в конце должна быть надпись Build succeeded, которая подтверждает удачную сборку .hex файла.

Все файлы можно посмотреть там, где было указано в строке Location. В папке Debug, которая находится в папке, указанной в строке Location, будет находится скомпилированный .hex файл, который необходим для прошивки реального микроконтроллера или для симуляции работы микроконтроллера в программе Proteus v7.7.

Proteus (by Labcenter Electronics) - симулятор принципиальных электронных схем. С помощью него можно проверить работу спроектированной электрической схемы. Proteus содержит большую библиотеку электронных компонентов [4].

В Proteus наряду с редактором электронных схем (ISIS) включен графический редактор печатных плат (ARES), т.е. при необходимости возможно развести печатную плату в соответствии с разработанной электронной схемой и создать реальное устройство.

### **Создание проекта в Proteus v7.7.**

1. Открыть предварительно установленную программу Proteus v7.7.

2. Собрать виртуальную электронную схему, которая в данном проекте включает:

- микроконтроллер ATmega128, программу для которого создали в AVR Studio 6;
- восемь светодиодов, с помощью которых можно увидеть результат работы микроконтроллера;
- восемь токоограничивающих резисторов;
- восемь кнопок, с помощью которых имеется возможность управлять вручную горением светодиодов.

3. Выбрать элементы виртуальной электронной схемы.

Это можно сделать двумя способами (рис.7 и рис.8).

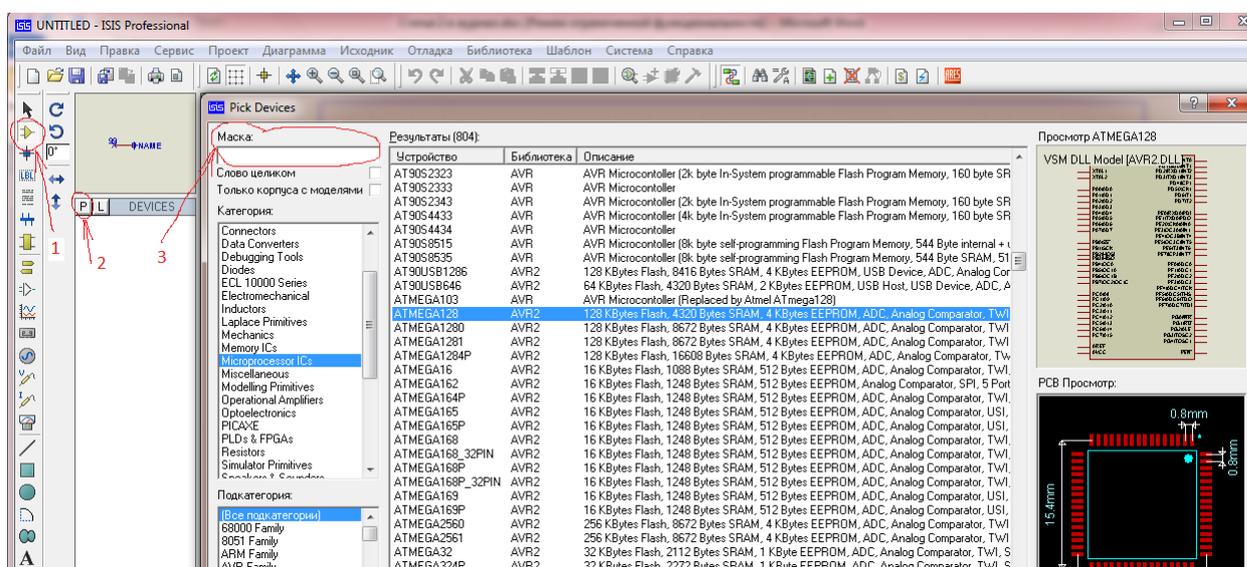


Рис.7 Выбор элементов виртуальной схемы (1 способ)

Нажимаем на кнопку  на панели инструментов слева (метка 1 на рис.7), затем на кнопку **P** (метка 2 на рис.7) слева от надписи DEVICES. Откроется окно, в котором необходимо выбрать нужный нам элемент. Например, **Microprocessor ICs — AVR Family — ATMEGA128**, а можно написать нужное название в строке поиска сверху (метка 3 на рис.7).

Второй способ выбора элементов – через панель **Menu — Библиотека — Выбрать устройство/ Символ** (рис.8).

После нажатия кнопки Ввод, выбранные элементы появятся в списке DEVICES на левой панели (рис.8).

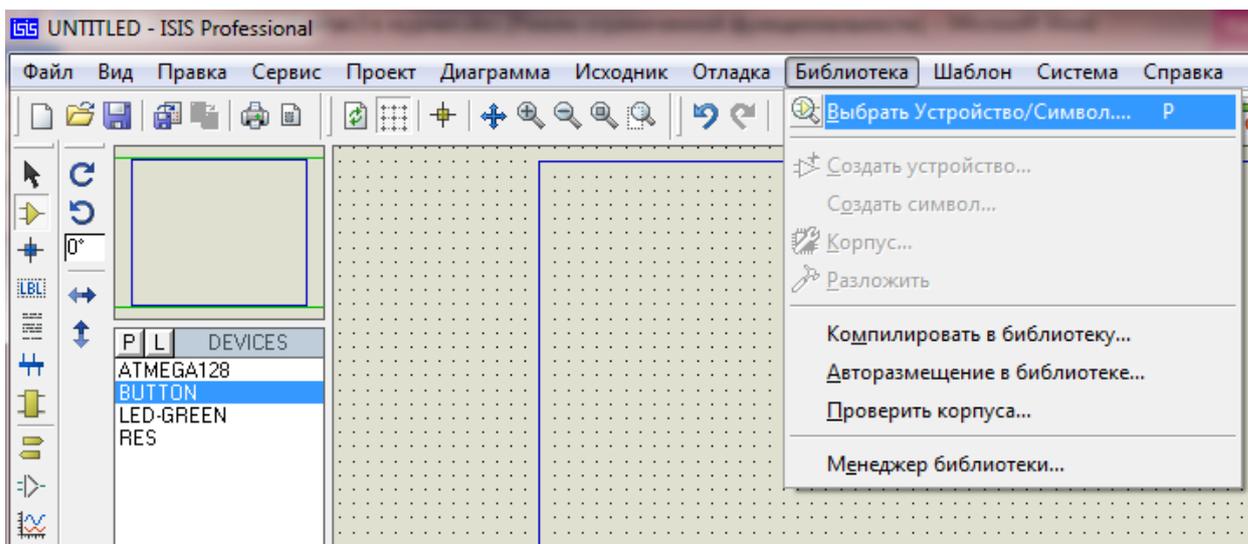
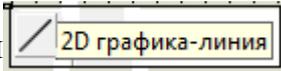


Рис.8 Выбор элементов виртуальной схемы (2 способ)

Аналогично осуществляется выбор других элементов схемы: резистор (**RES**), светодиод (**LED-GREEN**) и кнопка (**BUTTON**). Светодиоды могут быть выбраны различного свечения – зеленого (green), как в нашем случае, красного (red) или желтого (yellow) (рис.9).

#### 4. Объединить элементы в схему.

Из списка DEVICES выбираем микроконтроллер и помещаем его в рабочую область. Потом добавляем аналогичным образом 8 светодиодов и 8 резисторов и 8 кнопок. Резисторы имеют сопротивление по умолчанию 10 кОм, а нам нужно 300 Ом. Для изменения сопротивления щёлкнем по резистору двойным щелчком и в открывшемся окне найдем поле «Resistance» и внесем туда число 300.

Элементы схемы на монтажном поле размещаем таким образом, чтобы с одной стороны, минимизировать связи между элементами, а с другой – исключить их пересечение, т.е. в нашем случае, входные кнопки подключаются к порту D микроконтроллера, а светодиоды с ограничивающими резисторами к порту В. Соединение элементов осуществляется с помощью кнопки  на левой панели.

Катоды светодиодов потребуется подключить к земле. Чтобы получить вывод «Земля» нужно щёлкнуть на кнопке  и в списке выбрать «GROUND». В результате получаем готовый макет проекта.

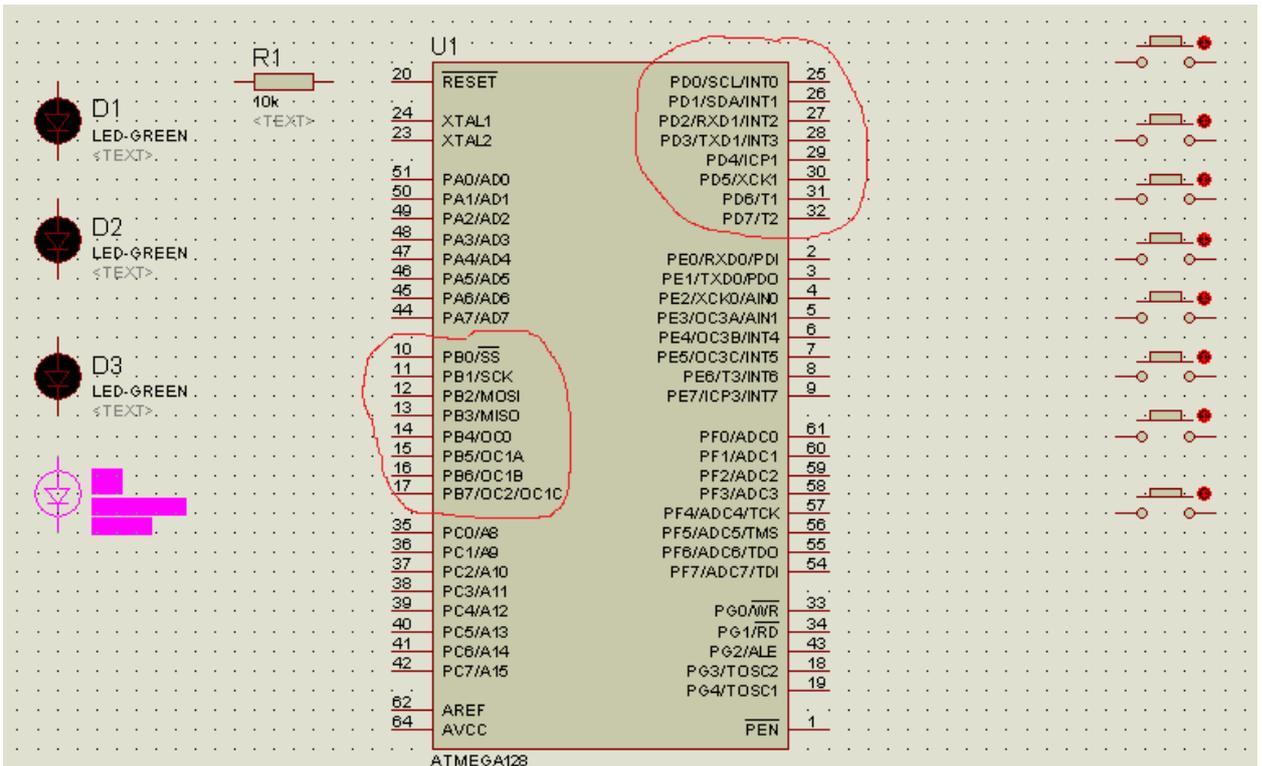


Рис.9 Размещение элементов схемы

## 5. Проверить работу собранной схемы.

В схеме кликаем на изображение контроллера и вводим путь, где находится .hex файл (рис.10), нажимаем ОК. Затем запускаем эмуляцию программы, нажав на кнопку **Старт**, и наблюдаем работу схемы в соответствии с написанной программой для микроконтроллера (рис.11).

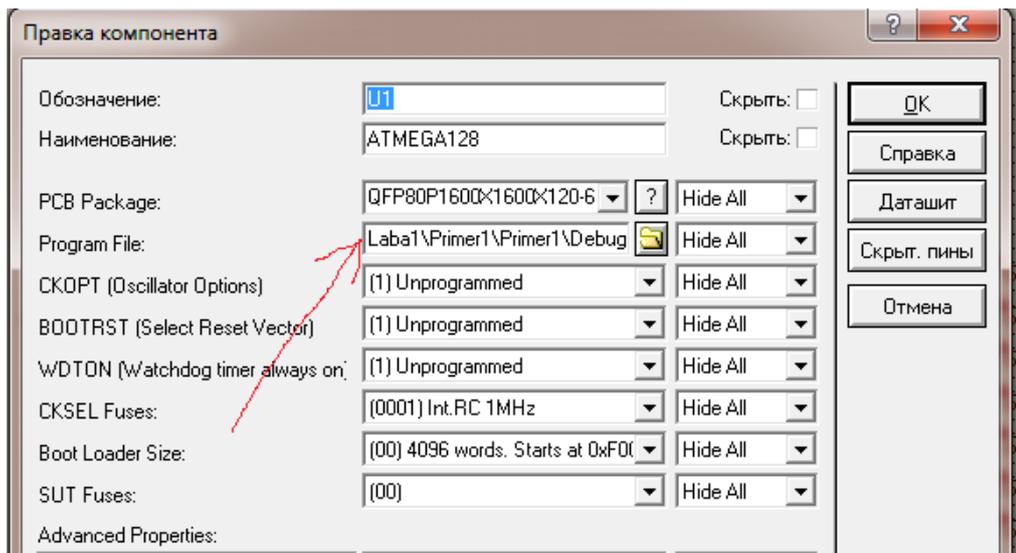


Рис.10 Выбор пути к .hex файлу микроконтроллера

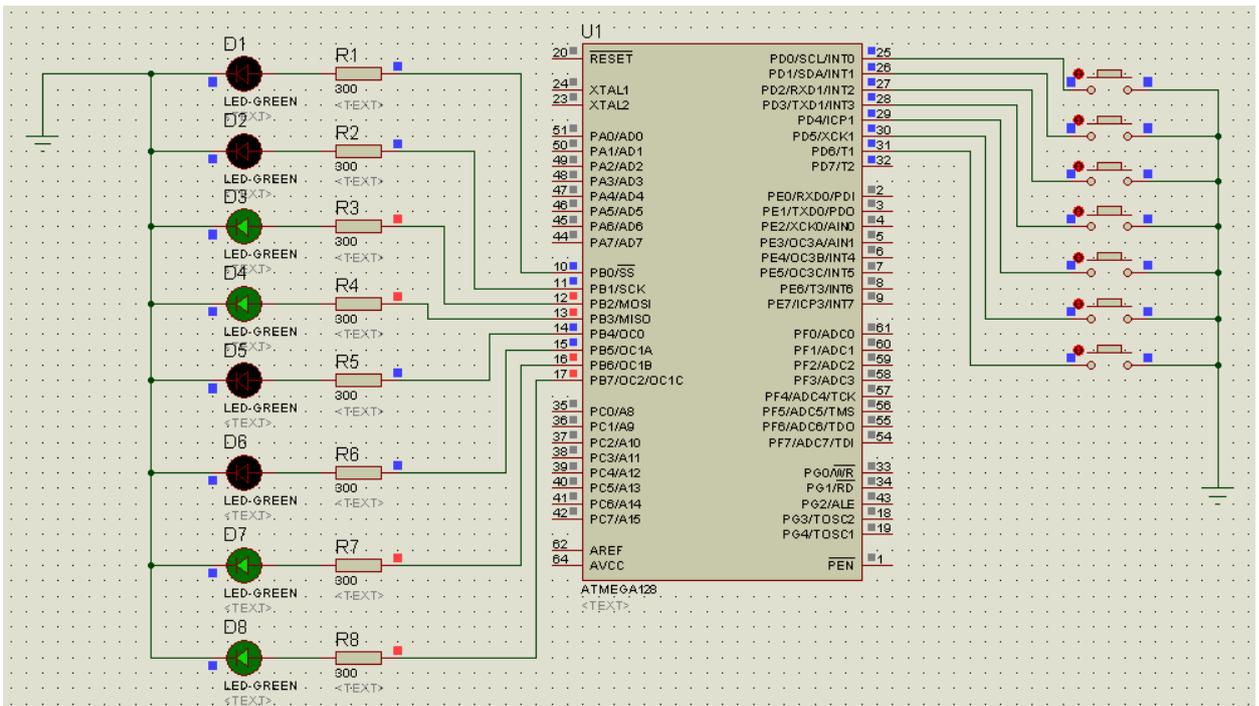


Рис.11 Работающий макет проекта

Таким образом, используя интегрированную среду AVR Studio 6 и программу Proteus v7.7., появляется возможность достаточно легко, с наименьшими материальными и временными затратами (что особенно важно в учебных условиях), изучить микроконтроллеры AVR фирмы ATMEL.

### Литература

1. Джон Мортон. Микроконтроллеры AVR. Вводный курс. – М.: Издательский дом Додэка-XXI, 2006. – 272 с.
2. Евстифеев А.В. Микроконтроллеры AVR семейства Mega. Руководство пользователя. – М.: Издательский дом «Додэка-XXI», 2007. – 592 с.
3. Баранов В.Н. Применение микроконтроллеров AVR: схемы, алгоритмы, программы. – М.: Издательский дом «Додэка-XXI», 2004. – 288с.
4. Программирование в AVR Studio 5 с самого начала: <http://datagor.ru/microcontrollers/1787-programirovanie-v-avrstudio-5-s-nulya.html>.