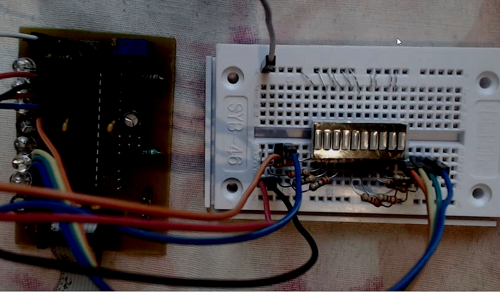
**AVR Урок 6. Бегущие огни**

**Урок 6**

**Бегущие огни**

Сегодня мы немного усовершенствуем наш проект, заодно и повторим битовые сдвиги, и не только повторим, а и увидим их смысл в деле. Мы применим данные сдвиги для того, чтобы наши светодиоды, находящиеся в матрице, мигали один за другим поочерёдно, за счёт чего наша схемка приобретёт ещё более живой вид.

Для этого нам потребуется уже не один светодиод. У меня на этот счёт имеется светодиодная планка или матрица. Я её поместил в беспаечную макетную плату, катоды всех светодиодов соединил вместе и подключил к общему проводу, а аноды каждый через токоограничивающий резистор подключил к соответствующим ножкам порта D. Вот так это всё выглядит (нажмите на картинку для увеличения изображения)

[](https://narodstream.ru/wp-content/uploads/2016/11/Image00_1041.png)

Поэтому, как обычно, по старой доброй традиции мы запускаем **Atmel Studio**, создаём в ней проект, выбрав тот же самый микроконтроллер **Atmega8a**, назовём проект **Test03**. Таким же образом в качестве отладчика выберем simulator, и также, чтобы сэкономить наше драгоценное время, скопируем весь код из файла main.c прошлого занятия.

Начнём писать код. Сначала мы в функции **main()** создадим целочисленную короткую беззнаковую переменную

int main(void)

{

**unsigned char i;**

Порт также оставляем на выход, и сразу на данном порте включим нулевую ножку в 1

DDRD = 0xFF;

PORTD = 0b0000000**1**;

А в бесконечном цикле мы создадим цикл другого типа — типа **for**. Данный цикл уже является конечным и работает он следующим образом



Данный цикл немного сложнее и условие в скобках здесь уже состоит из трёх частей, но я думаю, мы разберёмся. Мы ещё не с таким впоследствии разберёмся. Применим цикл данного типа в нашем коде:

while(1)

{

**for(i=0;i<=7;i++)**

**{**

     \_delay\_ms(500);

**}**

 }

В данном цикле у нас будет пока только задержка, остальной код мы уберём. То есть тело нашего цикла будет у нас выполняться до тех пор, пока переменная **i** у нас не достигнет значения, большего или равного **7**. То есть получится, что наше тело будет выполняться ровно 8 раз, затем мы выйдем из данного цикла и благодаря бесконечному циклу заново в него войдём и наш восьмикратный процесс повторится сначала.

А вот теперь сдвиг. Вставим его до задержки

**PORTD = (1<<i);**

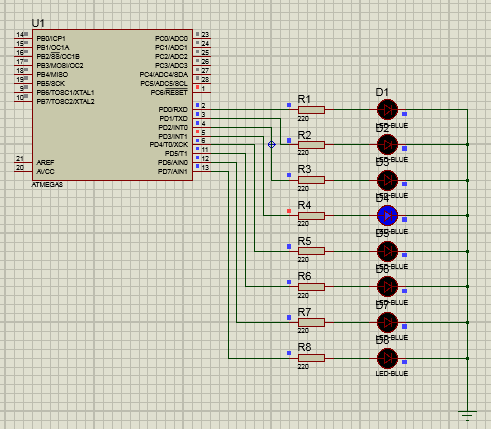
\_delay\_ms(500);

Как мы видим, данный сдвиг мы применяем к регистру, отвечающему за состояния порта **D**, и в нём мы будем сдвигать единичку влево на величину нашей переменной **i**, а так как данная переменная с каждым циклом увеличивается на 1 (или *инкрементируется*), то, соответственно, наша единичка постепенно раз в полсекунды будет двигиться влево, также как и лапки порта, за которые отвечает каждый бит нашего регистра. И тем самым мы и получим эффект бегущего огня.

Давайте соберём наш проект. И, также как и на прошлом занятии скопируем файл протеуса с прошлого занятия и переименуем его в **Test03**. Откроем его, заменим файл прошивки в свойствах контроллера.

Также добавим ещё 7 светодиодов и 7 резисторов, так как показано на схеме. Можно применять операцию копирования. Как это делается, показано в видеоуроке.

Запустим проект в протеусе и увидим, что наши светодиоды мигают поочерёдно, создавая впечатление эффекта бегущего огня



Теперь прошьём настоящий контроллер и увидим уже результат на практике. Это, конечно, намного интереснее, чем в протеусе.

На следующем занятии мы уже поработаем с ножкой порта не на выход, а на вход. Так как мы уже знаем, что порт — это такая шина, которая умеет работать в разных направлениях. Мы подключим кнопку и попытаемся отследить её состояние. То есть наша задача — узнать в какой-то определённый момент, нажата наша кнопка или отжата. Так что будет интересно.