**AVR. Собираем часы на микросхеме DS1307 и LCD 1602**

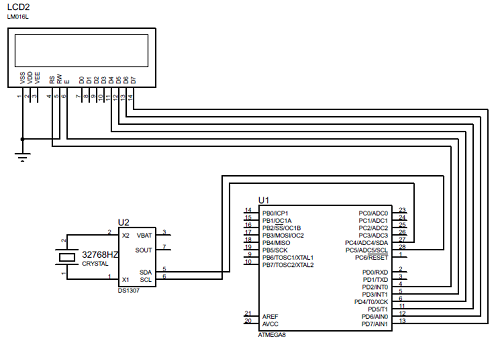
**Уро**

**Собираем часы на микросхеме DS1307 и LCD 1602**

Сегодня, набравшись уже опыта по программирования шины **I2С**, а также изучив микросхему реального времени **DS1307**, мы сможем приступить к программированию простейших часов и выводить их показания уже не в компьютер, а в жидкокристаллический дисплей. Я думаю, это будет намного наглядней и красивее, чем просматривать это в терминальной программе на ПК.

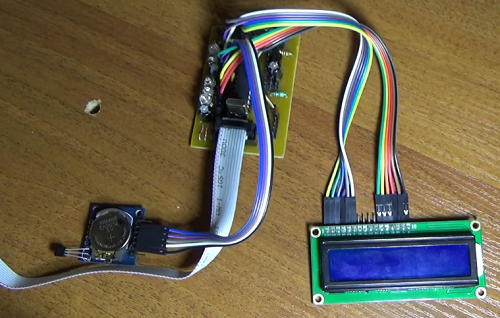
С дисплеем 16х2 мы работать также умеем, поэтому мы практически в данном занятии соединим два прошлых занятия, а также лишний раз повторим пройденный материал.

Будем считать, что модуль с микросхемой у нас уже присоединён к отладочной плате с контроллером **Atmega8A**, Присоединим к ней также модуль дисплея. Вот упрощённая схема из протеуса (нажмите на картинку для увеличения изображения)

[](https://narodstream.ru/wp-content/uploads/2016/12/Image00-8.png)

Ясное дело, что ещё здесь будет батарейка для сохранения хода часов при отключении питания, также вообще питание, контрастрость на дисплей. Но, я думаю, это понятно.

А вот так всё будет выглядеть на деле (нажмите на картинку для увеличения изображения)

[](https://narodstream.ru/wp-content/uploads/2016/12/Image01-9.png)

Дисплей подключен у нас будет подключин также в 4-битном режиме и к тем же ножкам портов, как и в [**уроке 12**](https://narodstream.ru/avr-urok-12-lcd-indikator-16x2-chast-1/).

Проект был создан с именем **MyClock1307LCD**. Все файлы, кроме главного скопированы туда и впоследствии подключены из проекта [**урока 17**](https://narodstream.ru/avr-urok-17-chasy-realnogo-vremeni-ds1307-chast-1/) **MyClock1307**.

Также к проекту подключены файлы для работы с дисплеем **lcd.c** и **lcd.h**. из проекта Test12, где мы работали с шиной USART.

Перейдём в главный модуль **MyClock1307LCD.c**. Код у нас здесь пока стандартный

#include «main.h»

//—————————————-

int main(void)

{

  while(1)

  {

  }

}

Зайдём в файл lcd.c и исправим там функцию sendchar в sencharlcd во избежания совпадения имен с другими модулями

//—————————————-

void sendchar**lcd**(unsigned char c)

Также добавим для неё прототип и ещё прототип для функции очистки дисплея в файл lcd.h

void str\_lcd (char str1[]);

**void clearlcd(void);**

**void sendcharlcd(unsigned char c);**

Теперь можно начать писать код в главном модуле. Причём почти весь его можно скопировать с предыдущих проектов.

Функцию инициализации порта возьмём из урока по дисплею

#include «main.h»

//—————————————-

**void port\_ini(void)**

**{**

**PORTD=0x00;**

**DDRD=0xFF;**

**}**

//—————————————-

Оттуда же возьмём в main() код инициализации порта и дисплея, а заодно его сразу и очистим

int main(void)

{

**port\_ini(); //Инициализируем порты**

**LCD\_ini(); //Инициализируем дисплей**

**clearlcd(); //Очистим дисплей**

Из главного модуля проекта из урока по часовой микросхемы возьмём инициализацию I2C

port\_ini(); //Инициализируем порты

**I2C\_Init(); //Инициализируем шину I2C**

LCD\_ini(); //Инициализируем дисплей

Оттуда же возьмём глобальные переменные для хранения показаний регистров

#include «main.h»

//—————————————-

**unsigned char sec,min,hour,day,date,month,year;**

//—————————————-

Вполне нам может также пригодиться закомментированный код установки времени, вдруг придется устанавливать, так как кнопки мы пока не заводим

clearlcd(); //Очистим дисплей

**//Устанавливаем время**

**// I2C\_StartCondition();**

**// I2C\_SendByte(0b11010000);**

**// I2C\_SendByte(0);//Переходим на 0x00**

**// I2C\_SendByte(RTC\_ConvertFromBinDec(0)); //секунды**

**// I2C\_SendByte(RTC\_ConvertFromBinDec(31)); //минуты**

**// I2C\_SendByte(RTC\_ConvertFromBinDec(20)); //часы**

**// I2C\_SendByte(RTC\_ConvertFromBinDec(5)); //день недели**

**// I2C\_SendByte(RTC\_ConvertFromBinDec(29)); //дата**

**// I2C\_SendByte(RTC\_ConvertFromBinDec(1)); //месяц**

**// I2C\_SendByte(RTC\_ConvertFromBinDec(16)); //год**

**// I2C\_StopCondition();**

while(1)

Также возьмём пока всё полностью и из бесконечного цикла

**while(1)**

После преобразований данных в бесконечном цикле добавим функцию установки курсора дисплея в начальное положение

date = RTC\_ConvertFromDec(date); //Преобразуем в десятичный формат

**setpos(0,0); //Ставим курсор на начало координат**

Заменим в коде **USART\_Transmit** на **sendcharlcd**пока там, где работаем с датой ,не со временем, а также перед показаниями года добавим «20», чтобы год смотрелся посолидней

setpos(0,0); //Ставим курсор на начало координат

**sendcharlcd(date/10+0x30);//Преобразуем число в код числа**

**sendcharlcd(date%10+0x30);//Преобразуем число в код числа**

**sendcharlcd('.');**

**sendcharlcd(month/10+0x30);//Преобразуем число в код числа**

**sendcharlcd(month%10+0x30);//Преобразуем число в код числа**

**sendcharlcd('.');**

**sendcharlcd('2');**

**sendcharlcd('0');**

**sendcharlcd(year/10+0x30);//Преобразуем число в код числа**

**sendcharlcd(year%10+0x30);//Преобразуем число в код числа**

**sendcharlcd(' ');**

**sendcharlcd('-');**

**sendcharlcd(day+0x30);//Преобразуем число в код числа**

**sendcharlcd('-');**

Так как мы с показаниями времени не влезем на 1 строку, поэтому переведём перед этим строку, а затем также исправим везде вызов функции на **sendcharlcd**, также уберём перевод строки и возврат каретки.

setpos(0,1); //Ставим курсор на начало координат

**sendcharlcd(hour/10+0x30);//Преобразуем число в код числа**

**sendcharlcd(hour%10+0x30);//Преобразуем число в код числа**

**sendcharlcd(':');**

**sendcharlcd(min/10+0x30);//Преобразуем число в код числа**

**sendcharlcd(min%10+0x30);//Преобразуем число в код числа**

**sendcharlcd(':');**

**sendcharlcd(sec/10+0x30);//Преобразуем число в код числа**

**sendcharlcd(sec%10+0x30);//Преобразуем число в код числа**

Попробуем собрать код и прошить контроллер.

Наши часы отлично ходят!

