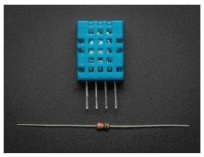
Presented here are a couple of simple add-ons to the S100-MEGA board. The units are the DHT11 temperature and humidity sensor and the DS1307 Real Time Clock, both designed to work with the Arduino UNO board. I purchased the units from Adafruit. Here are pictures from the web page:



DHT11 basic temperature-humidity sensor + extras

PRODUCT ID: 386

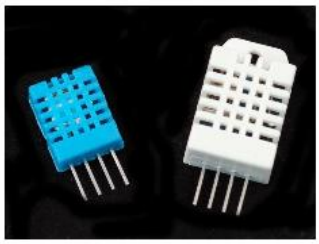
ADD TO CART $5.00 IN STOCK

Adafruit DS1307 Real Time Clock Assembled Breakout Board

PRODUCT ID: 3296

ADD TO CART $7.50 IN STOCK

The actual wring is quite simple. I have copied a section for Adafruit showing some details.



DHT11, DHT22

Overview

Connecting to a DHTxx Sensor

Using a DHTxx Sensor

DHT CircuitPython Code

Python Docs

Downloads

Featured Products

Single Page

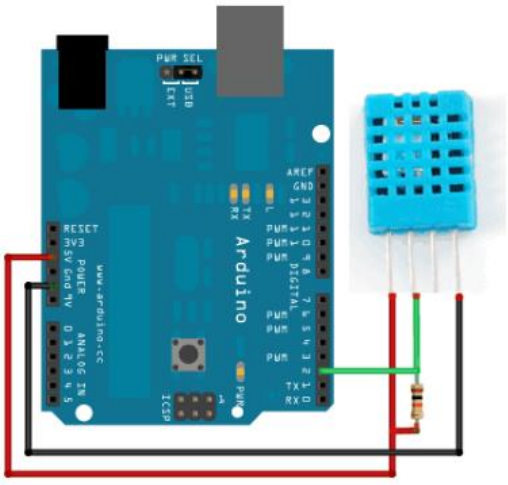
Download PDF

Likewise, it is fairly easy to connect up to the DHT sensors. They have four pins

1. **VCC** - red wire Connect to 3.3 - 5V power. Sometime 3.3V power isn't enough in which case try 5V power.
2. **Data out** - white or yellow wire
3. Not connected
4. **Ground** - black wire

Simply ignore pin 3, its not used. You will want to place a 10 Kohm resistor between VCC and the data pin, to act as a medium-strength pull up on the data line. The Arduino has built in pullups you can turn on but they're very weak, about 20-50K

This diagram shows how we will connect for the testing sketch. Connect data to pin 2, you can change it later to any pin.



DS1307 Real Time Clock Breakout Board Kit

ADAFRUIT PRODUCTS | BREAKOUT BOARDS / CLOCKS

Wiring It Up

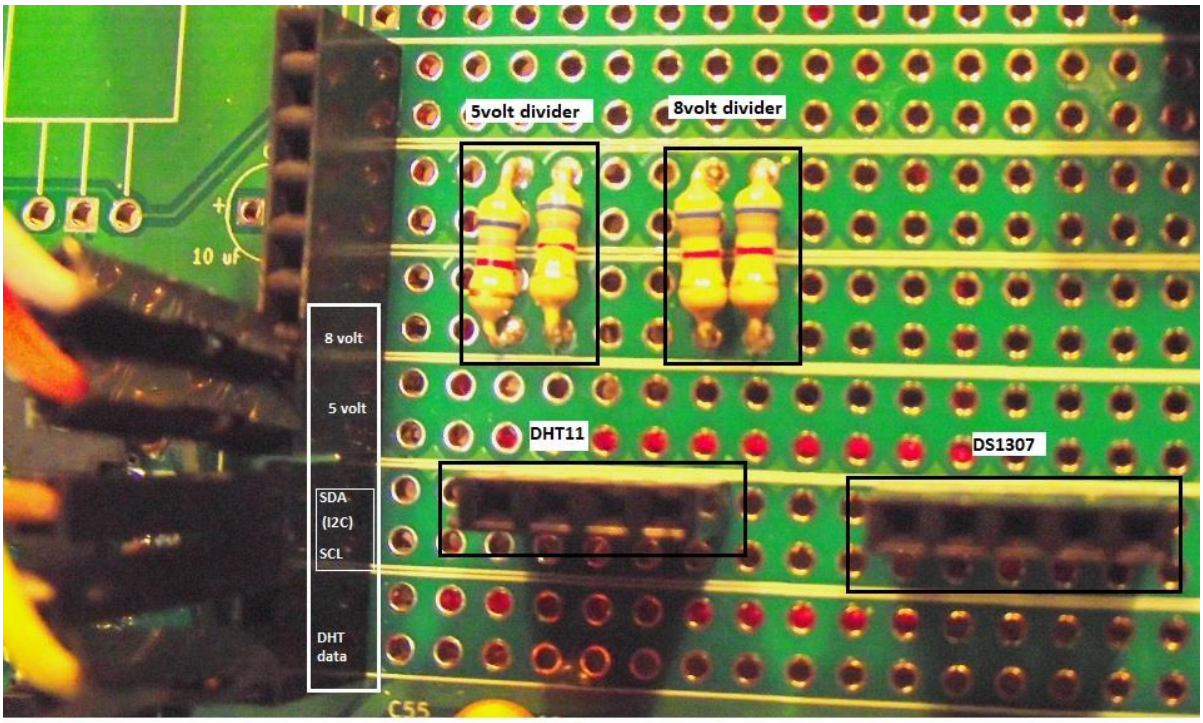
by Tyler Cooper

There are only 5 pins: **5V GND SCL SDA SQW**.

- **5V** is used to power to the RTC chip when you want to query it for the time. If there is no 5V signal, the chip goes to sleep using the coin cell for backup.
- Connect **GND** to common power/data ground
- Connect the **SCL** pin to the I2C clock **SCL** pin on your Arduino. On an UNO & '328 based Arduino, this is also known as **A5**, on a Mega it is also known as **digital 21** and on a Leonardo/Micro, **digital 3**
- Connect the **SDA** pin to the I2C data **SDA** pin on your Arduino. On an UNO & '328 based Arduino, this is also known as **A4**, on a Mega it is also known as **digital 20** and on a Leonardo/Micro, **digital 2**

SQW is the optional square-wave output you can get from the RTC if you have configured it to do so. Most people don't need or use this pin

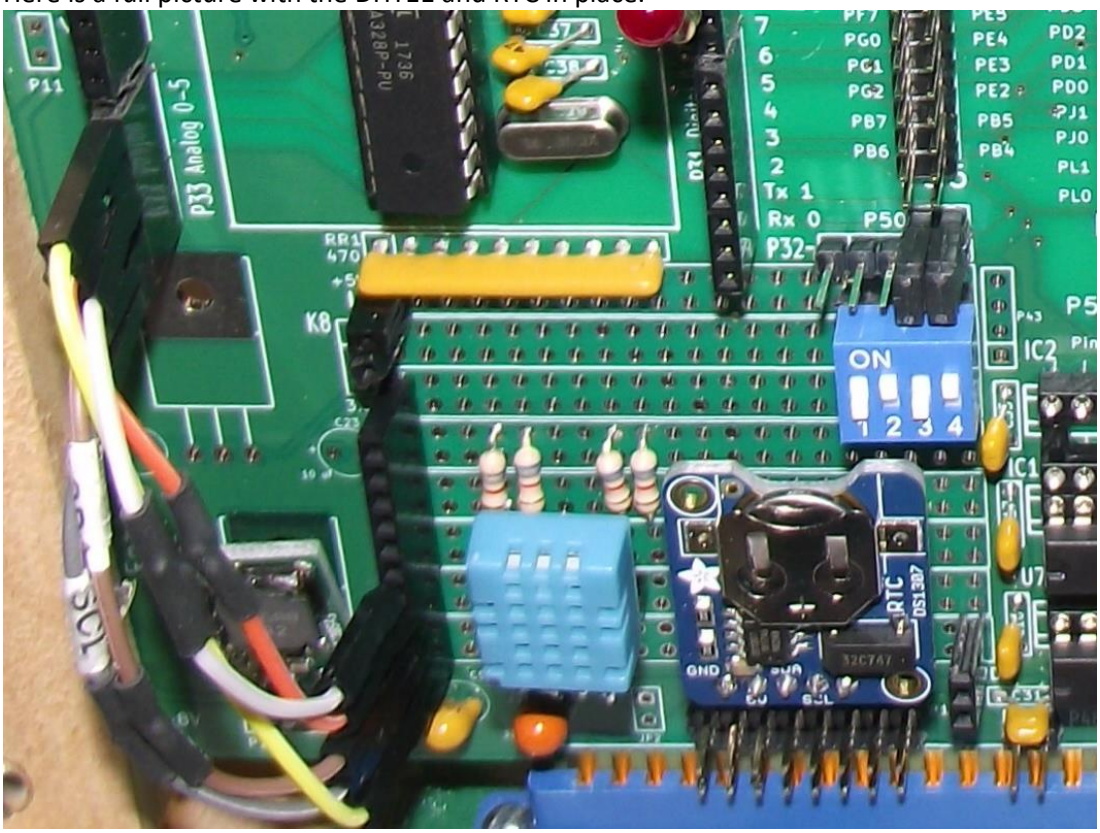
I wired up both units in the prototype area of the S100 board and added female headers for jumpers to P33 (the analog inputs of the 328). I also used headers for the DHT11 and the RTC. I did this in case one of the units went bad or if I wanted to make a change. Below is a picture of the prototype area. The set of resistors are voltage dividers that I use to acquire 5 and 8 volt board signals.



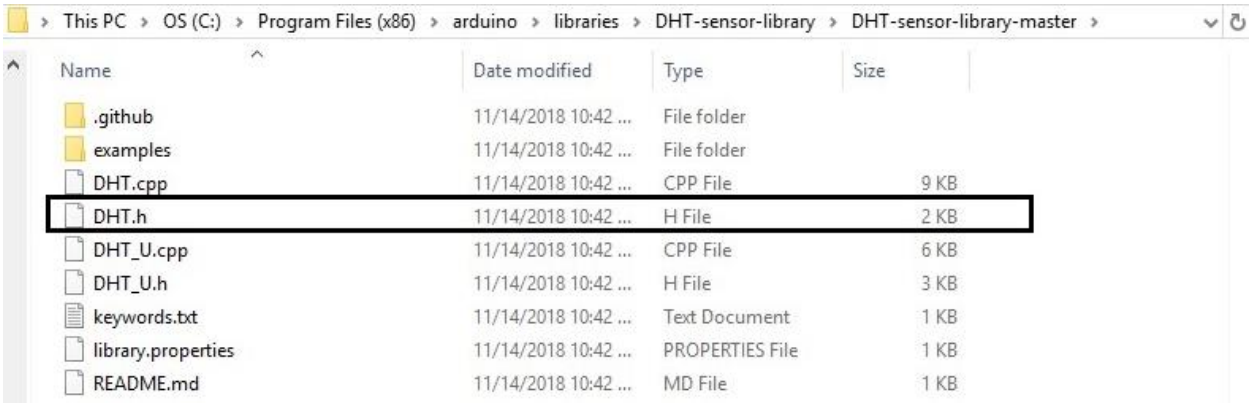
The jumpers are connected as follows:

	P33 Analog 0-5
DHT11 data (pin2)	- A0
5 volt signal	- A1
8 volt signal	- A2
DS1307 SDA (pin4)	- A4
DS1307 SCL(pin3)	- A5

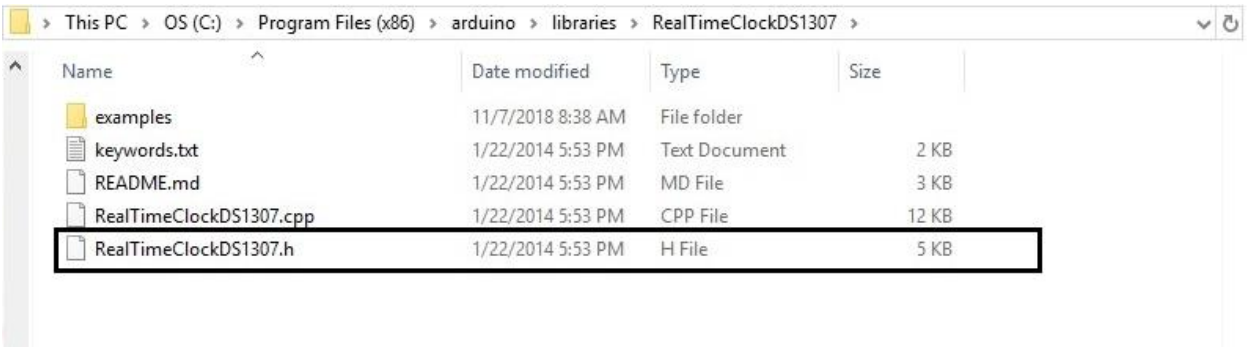
Here is a full picture with the DHT11 and RTC in place.



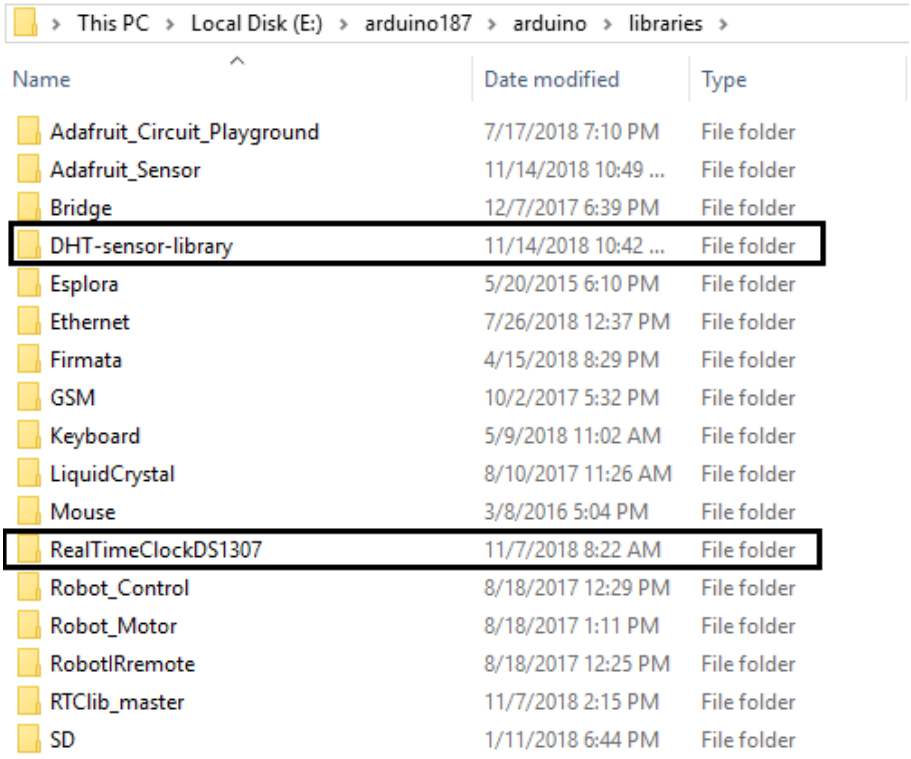
Completing the first part of the S100 Mega board also required downloading the IDE and associated libraries. It is important to do that since the program that I will be describing uses many of the header files designed for the S100 MEGA board. The directory path for the DHT11 should look something like this:



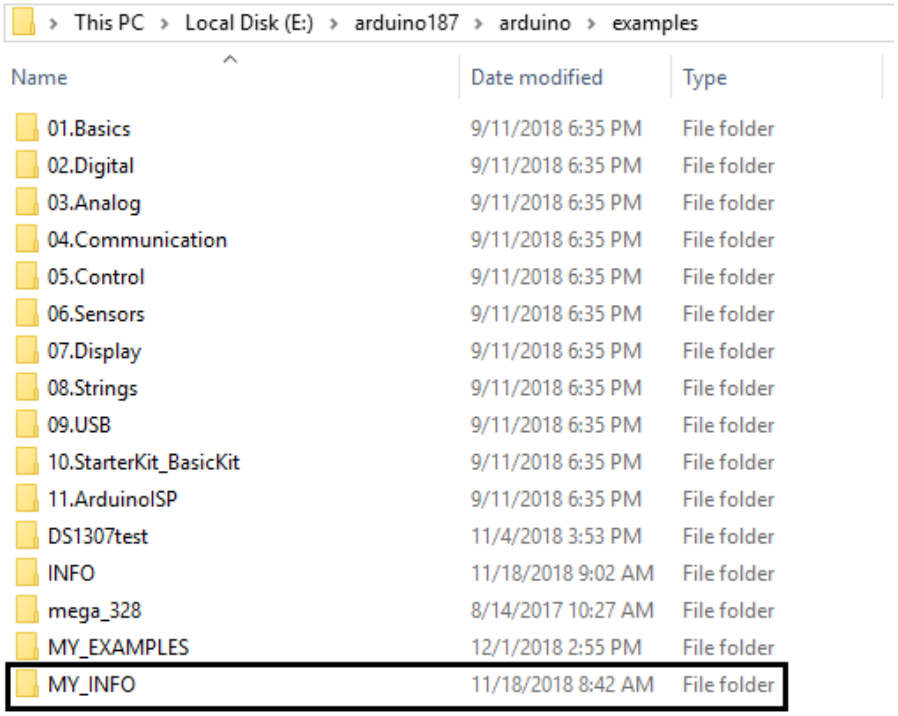
The required library is easily downloaded from Adafruit. I have highlighted the header file that is needed for my program. Similarly for the DS1307:



The directory for the IDE should include the highlighted libraries:



The following web page gives detailed instructions for the DS1307 and DHT:
https://www.instructables.com/id/Arduino-Real-Time-Clock-DS1307
Download the library "RealTimeClockDS1307" and save it in directory path with the Arduino IDE.
https://github.com/adafruit/DHT-sensor-library
Download the library “DHT-sensor-library” and save it in the directory path with the Arduino IDE.
Once these libraries have been downloaded and unzipped then copy the sketch (program) MY_INFO.ino from below. The program needs to be in its own folder named MY_INFO in the examples folder. Here is what the EXAMPLES directory looks like for my setup:



I constructed the sketch “MY_INFO.ino” by pulling out sections from the example programs that I downloaded and modifying some of the pin assignments to match the S100 MEGA board. As shown below the initial portion of the program defines and includes needed header files.

```
/*
  MY_INFO.ino  11/28/2018  rev 2 using the MEGA 328 controller
  Using DS1307 RTC from Adafruit
  now with temp and humid from DHT11 from Adafruit
  set for A1 5volt and A2 8volt
  switch (sw1) set for break which will display the voltages when program is running
  LCD Display  on Arduino 328-2560 S100 CPU Board
  Use sections of RTC_RD_LCD.ino for the reading and writing to the LCD (downloaded from web)
  Use the main code from the example LCD_328.ino
  added offset -1.5 for DHT11 to match other readings

*/

// Libraries

#include <Wire.h>                       // contains headers for 2 wire (I2C)  communication
#include <LiquidCrystal.h>              // contains headers for the control of the LCD
#include <EEPROM.h>
#include <SD.h>
#include <SPI.h>                        // required for Arduino >= 1.5.4
#include <DHT.h>                        // download from Aeduino and keep in library
#include <RealTimeClockDS1307.h>        // download from Arduino or Adafruit keep in library
                                        // for rtc readout on serial
#define DHTPIN A0                       // input used for DHT11
#define DHTTYPE DHT11                   // we are using the DHT11 sensor
#define ANALOG_INPUT 59                 // Analog A5 (pin1 P33) to Switches on the
                                        // S100_MEGA board (if MEGA 328 is used)
#define Display_Clock_Every_N_Seconds 1 // RTC updated every second
```

Note that the pin assignments have to be changed from Arduino UNO examples that were downloaded. The following code comes from John Monahan’s examples for the S100 MEGA board.

```
// Needed in the LCD libraries to define pin setup with the Arduino
// the pin settings are for the S100_MEGA, not the Arduino UNO
// CPU pins 14, 15, 6, 11, 12, 13  (PB0, PB1, PD7, PD6, PD5, PD4 of CPU)
// PL7, PL2, PL3, PL4, PL5, PL6 on schematic with P51 jumpers

LiquidCrystal lcd(8, 9, 7, 6, 5, 4);

// Pin that will receive clock pulse from RTC
int clockPin = 0;

// Time and date variables
byte second;
byte minute;
byte hour;
byte day;
byte date;
byte month;
byte year;

DHT dht(DHTPIN, DHTTYPE);     // need this for DHT11 choice

int sensorPin = A1;           // select the input pin for the voltage measurement of system 5V
int sensorPin2 = A2;          // select the input pin for the voltage measurement of system 8V
int sensorValue = 0;          // variable to store the value coming from the sensorPin1
int sensorValue2 = 0;         // variable to store the value coming from the sensorPin2
int AnalogIn;                 // measure switch voltage
```

The basic flow of the program follows the typical format used by the Arduino system with a "void setup ()" section followed by a "void loop()" section. I also added a break command to jump out and read out the voltages using sw1 and described for the S100 MEGA board.

Here is a picture of the system. The MEGA board is running the MY_INFO program continuously, even when the Z80 SBC boots up the CP/M.