

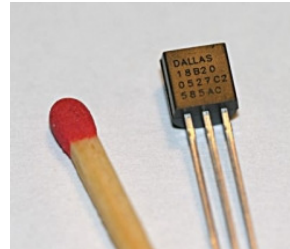
## Electronic Brick and other Versions

NOTE: There are different-appearing versions but they work the same.

This is an electronic thermometer which has high accuracy over a wide range (accurate to  $\pm 0.5^{\circ}\text{C}$  over the range of  $-10^{\circ}\text{C}$  to  $+85^{\circ}\text{C}$ ) (Workable from  $-55^{\circ}\text{C}$  to  $+125^{\circ}\text{C}$ ). You can locate these thermometer chips up to 100M away from your Arduino. Shorter cables can be just 2 wires.



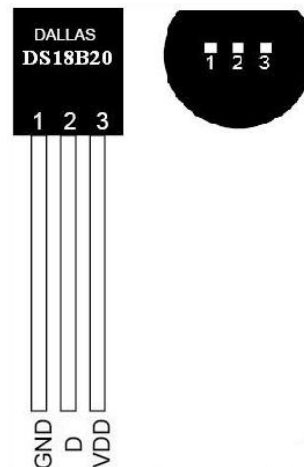
Multiple thermometers can be connected on the same wire because every one has its own internal address. This is the "1-wire" bus system ([Wikipedia](#)). Below is an example of reading up to 5 DS18B20's on a single Arduino pin. This requires that you know the internal address of each sensor. A utility sketch to find the address is also below.



Here (right) is what the individual chip sensors look like, and their pinout:

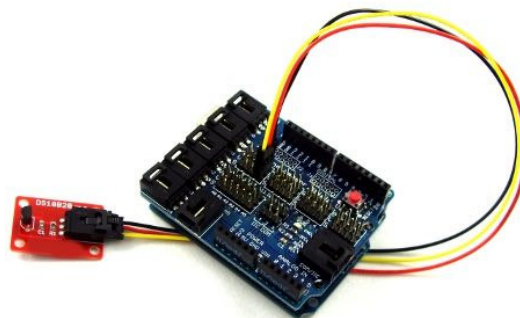
Here are links to more about the DS18B20 and other "1-Wire" chips:

- [The DS18B20 Data Sheet](#)
- [The updated "DallasTemperature" Library \(zip file\)](#)
- [The updated "1-Wire" Library \(zip file\)](#)
- [The Arduino Site: "1-Wire" information](#)
- [The Software Library and information from Miles Burton](#)
- [Schematic Diagram of this Brick](#)
- [Stainless Steel type Temperature Probe](#)
- [HOW-TO Connect several DS18B20 on same wires](#)
- [Multiple 1-Wire Buses on 1 Arduino](#)
- [MAXIM Detailed 1-Wire Protocol Presentation](#)
- [LONG Cables for DS18B20](#)
- [MAXIM Application note about Long Cables](#)



The DS18B20 Brick is the easiest way to get connected and measure temperatures. You can buy individual chips for about \$1.50 [here](#), or the Brick shown above, with a cable, for \$7.25 [here](#). Or a waterproofed version [here](#):

The easy way to get connected is to simply plug the supplied cable into a [YourDuinoRobo1](#) with built-in 3-pin connectors, or a [Sensor Shield](#) that plugs on top of an Arduino like this:



Plug your DS18B20 Brick's cable into I/O pin connector 3 on Robo1 or 2 on a sensor shield. (NOTE! Pin numbers start with Zero).

### How does this Work??

Now, let's think for a minute about what we're doing here. The earlier devices we connected

(NOTE! PIN numbers start with Zero).

## How does this Work??

Now, let's think for a minute about what we're doing here. The earlier devices we connected were simple: Switches, LEDs, Potentiometers, etc. A single component. This is different: we are connecting to a chip that has hundreds of transistors inside it and dozens of possible functions. It expects us to send it commands and it will return data as lots of signal pulses of 1's and 0's. How will we communicate with it? Fortunately, we're not in an empty room. We're in a library. On the shelves, free for us to check out, are lots of already-published Arduino functions that we can use. We will use two library check-outs to run our temperature sensor that other people have spent 100's of hours working on, and given us unlimited copies to check out. (Terry's wife designs Libraries. [Here's one she designed in China:](#) )

You will need to download and install the two libraries for One-wire devices and DallasTemperature chips and copy them to your Arduino software installation Library folder. (See above).




(each one is called a "Library" not a Book! )

For more about Libraries see our [Arduino Libraries page](#).

Then just copy and paste this test code (below) into your [Arduino IDE](#):

NOTE: If using the YourDuinoRobo1, connect to pin 3 (and change the pin number in the code below to 3).

Then:

-  Click the VERIFY button. Soon you should see "Compiling" and "Done Compiling"
-  Click the UPLOAD button. You should see "Uploading to IO Board" and "Done Uploading"
-  Now, click the SERIAL MONITOR button. A new window should pop up and scroll lines of results showing you.. Temperature!

Troubleshooting?? Go back and get the original BLINK program working first. [Check Here:](#)

```
/* YourDuino Electronic Brick Test
Temperature Sensor DS18B20
- Connect cable to Arduino Digital I/O Pin 2
terry@yourduino.com */

/*-----( Import needed libraries )-----*/
#include <OneWire.h>
#include <DallasTemperature.h>

/*-----( Declare Constants )-----*/
#define ONE_WIRE_BUS 2 /*-(Connect to Pin 2 )-*/

/*-----( Declare objects )-----*/
/* Set up a oneWire instance to communicate with any OneWire device*/
OneWire ourWire(ONE_WIRE_BUS);

/* Tell Dallas Temperature Library to use oneWire Library */
DallasTemperature sensors(&ourWire);

/*-----( Declare Variables )-----*/

void setup() /*-----( SETUP: RUNS ONCE )-----*/
{
  /*-(start serial port to see results )-*/
  delay(1000);
  Serial.begin(9600);
  Serial.println("YourDuino.com: Electronic Brick Test Program");
  Serial.println("Temperature Sensor DS18B20");
  delay(1000);

  /*-( Start up the DallasTemperature library )-*/
  sensors.begin();
}/*--(end setup )---*/

void loop() /*-----( LOOP: RUNS CONSTANTLY )-----*/
{
  Serial.println();
  Serial.print("Requesting temperature...");
  sensors.requestTemperatures(); // Send the command to get temperatures
  Serial.println("DONE");

  Serial.print("Device 1 (index 0) = ");
  Serial.print(sensors.getTempCByIndex(0));
  Serial.print("C");
}
```

```

Serial.print("Requesting temperature...");
sensors.requestTemperatures(); // Send the command to get temperatures
Serial.println("DONE");

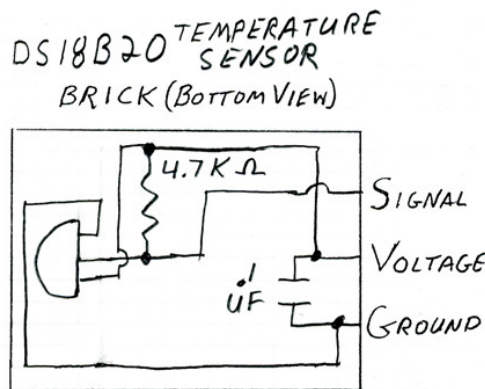
Serial.print("Device 1 (index 0) = ");
Serial.print(sensors.getTempCByIndex(0));
Serial.println(" Degrees C");
Serial.print("Device 1 (index 0) = ");
Serial.print(sensors.getTempFByIndex(0));
Serial.println(" Degrees F");

/* --(end main loop)-- */

/* ( THE END ) */

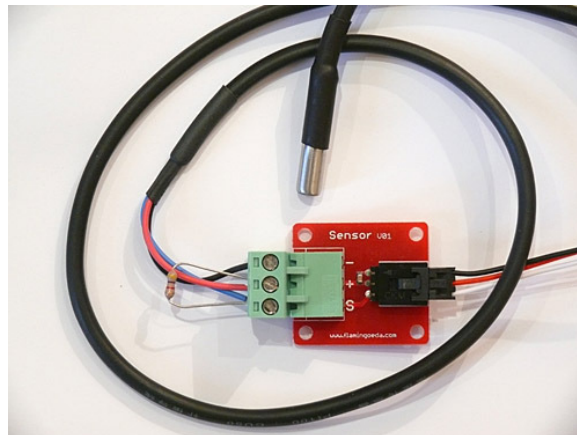
```

Here's the schematic diagram of the Brick:



And here is another form of exactly the same DS18B20 temperature sensor in a waterproof stainless steel tube. These are [Available Here](#) ). You have the same connections to voltage and ground and signal, but you have to provide your own 4.7K "pullup" resistor which can be any small type resistor. Be careful to ensure the Signal and Ground connections are correct. If they are reversed, the sensor can

get quite hot (even at such a low voltage).



Here is a photo of this sensor connected to a ["3-pin Sensor" connector](#) so that it can be connected like our regular Electronic Bricks. You can see the 4.7K pullup resistor connected between + and Signal. Sensor Connections: red (VCC), blue or yellow (DATA), black (GND)

## Read Temperatures from multiple DS18B20 sensors on 1 Arduino Pin:

You will need to know the internal address of each sensor. Farther below is a Utility Sketch that can read the address from sensors one at a time. You need to cut and paste the addresses, first to a text file and later into the example multi-sensor example sketch below.

## Multiple DS18B20 Example:

```

/* YourDuino Multiple DS18B20 Temperature Sensors on 1 wire
Connections:
*/

```

## Multiple DS18B20 Example.

```
/* YourDuino Multiple DS18B20 Temperature Sensors on 1 wire
Connections:
DS18B20 Pinout (Left to Right, pins down, flat side toward you)
- Left   = Ground
- Center = Signal (Pin 2):  (with 3.3K to 4.7K resistor to +5 or 3.3 )
- Right  = +5 or +3.3 V

Questions: terry@yourduino.com
V1.01 01/17/2013 ...based on examples from Rik Kretzinger

/*-----( Import needed libraries )-----*/
// Get 1-wire Library here: http://www.pjrc.com/teensy/td_libs_OneWire.html
#include <OneWire.h>

//Get DallasTemperature Library here:  http://milesburton.com/Main_Page?title=Dallas_Temperature_Control_Library
#include <DallasTemperature.h>

/*-----( Declare Constants and Pin Numbers )-----*/
#define ONE_WIRE_BUS_PIN 2

/*-----( Declare objects )-----*/
// Setup a oneWire instance to communicate with any OneWire devices
OneWire oneWire(ONE_WIRE_BUS_PIN);

// Pass our oneWire reference to Dallas Temperature.
DallasTemperature sensors(&oneWire);

/*-----( Declare Variables )-----*/
// Assign the addresses of your 1-Wire temp sensors.
// See the tutorial on how to obtain these addresses:
// http://www.hacktronics.com/Tutorials/arduino-1-wire-address-finder.html

DeviceAddress Probe01 = { 0x28, 0x8A, 0xB1, 0x40, 0x04, 0x00, 0x00, 0xC7 };
DeviceAddress Probe02 = { 0x28, 0xCC, 0x92, 0x40, 0x04, 0x00, 0x00, 0xB6 };
DeviceAddress Probe03 = { 0x28, 0x4D, 0x8D, 0x40, 0x04, 0x00, 0x00, 0x78 };
DeviceAddress Probe04 = { 0x28, 0x9A, 0x80, 0x40, 0x04, 0x00, 0x00, 0xD5 };
DeviceAddress Probe05 = { 0x28, 0xE1, 0xC7, 0x40, 0x04, 0x00, 0x00, 0x0D };

void setup()    /***** SETUP: RUNS ONCE *****/
{
    // start serial port to show results
    Serial.begin(9600);
    Serial.print("Initializing Temperature Control Library Version ");
    Serial.println(DALLASTEMLIBVERSION);

    // Initialize the Temperature measurement library
    sensors.begin();

    // set the resolution to 10 bit (Can be 9 to 12 bits .. lower is faster)
    sensors.setResolution(Probe01, 10);
    sensors.setResolution(Probe02, 10);
    sensors.setResolution(Probe03, 10);
    sensors.setResolution(Probe04, 10);
    sensors.setResolution(Probe05, 10);
}

//--(end setup )---

void loop()    /***** LOOP: RUNS CONSTANTLY *****/
{
    delay(1000);
    Serial.println();
    Serial.print("Number of Devices found on bus = ");
    Serial.println(sensors.getDeviceCount());
    Serial.print("Getting temperatures... ");
    Serial.println();

    // Command all devices on bus to read temperature
    sensors.requestTemperatures();

    Serial.print("Probe 01 temperature is:  ");
    printTemperature(Probe01);
    Serial.println();

    Serial.print("Probe 02 temperature is:  ");
    printTemperature(Probe02);
    Serial.println();

    Serial.print("Probe 03 temperature is:  ");
    printTemperature(Probe03);
    Serial.println();

    Serial.print("Probe 04 temperature is:  ");
    printTemperature(Probe04);
    Serial.println();

    Serial.print("Probe 05 temperature is:  ");
    printTemperature(Probe05);
    Serial.println();
}
```

```

Serial.print("Probe 05 temperature is:  ");
printTemperature(Probe05);
Serial.println();

} //--(end main loop )--

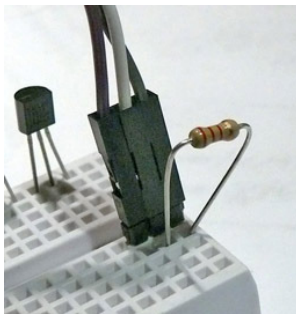
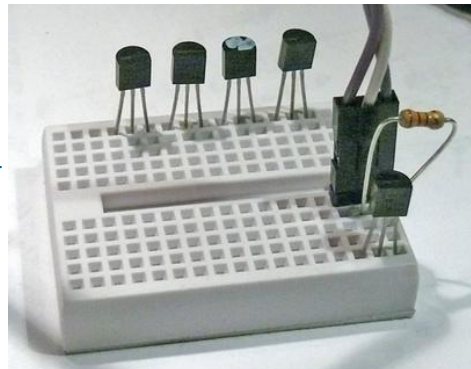
/*-----( Declare User-written Functions )-----*/
void printTemperature(DeviceAddress deviceAddress)
{
float tempC = sensors.getTempC(deviceAddress);

if (tempC == -127.00)
{
Serial.print("Error getting temperature  ");
}
else
{
Serial.print("C: ");
Serial.print(tempC);
Serial.print(" F: ");
Serial.print(DallasTemperature::toFahrenheit(tempC));
}
}
} // End printTemperature
//***** ( THE END ) *****

```

## Read individual DS18B20 Internal Addresses:

Here's a way to connect multiple DS18B20's to get their addresses and test them: A 3-wire cable connects Arduino +5V, Ground and Pin 2 to a small breadboard. The breadboard can "Store" DS18B20's you are working on, and 1 or more can be connected for test.



Here's a closeup of the connections: Left=Ground, Center=Data, Right=+5V, 3300 to 4700 ohm resistor from +5V to Data.

Find your sensors and put them in the top "Storage" row of the breadboard. Then put them one at a time in the active 3 columns on the lower right. Cut/Paste the results into your text file.

Next, cut and paste the sketch below into an Arduino IDE blank window. Save it. Upload it. Then open the Serial Monitor window.

You should see this:

```

Found '1-Wire' device with address:
0x28, 0x8A, 0xB1, 0x40, 0x04, 0x00, 0x00, 0xC7
Done

```

That address is unique to that DS18B20.. not another the same anywhere! Open a text file and cut/paste the address to save it. I'd label it Sensor1 and put that sensor back in the first "storage" position. Do the same for the rest of your sensors. The sensors are hard to physically mark. "Sharpie" Paint Markers can work. Or an electric engraver works OK. Tape? Now cut/paste the "Multiple DS18B20 Example" Sketch above. Save. Edit it to put the addresses of your sensors in the lines that look like this:

```
DeviceAddress Probe01 = { 0x28, 0x8A, 0xB1, 0x40, 0x04, 0x00, 0x00, 0xC7 };
```

Save, Upload. Plug your first DS18B20 in. Then open the Serial Monitor window. You should see this:

DeviceAddress Probe01 = { 0x28, 0x8A, 0xB1, 0x40, 0x04, 0x00, 0x00, 0xC7 };

Save, Upload. Plug your first DS18B20 in. Then open the Serial Monitor window. You should see this:

Getting temperatures...

Probe 01 temperature is: C: 26.75 F: 80.15

Probe 02 temperature is: Error getting temperature

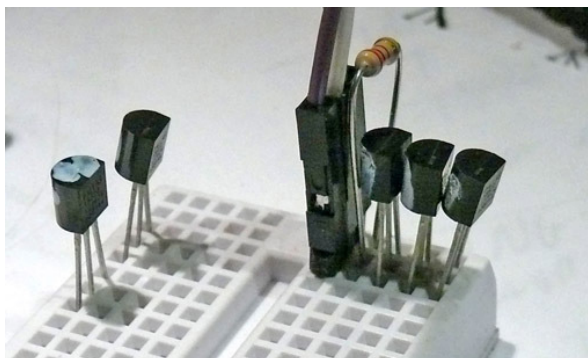
Probe 03 temperature is: Error getting temperature

Probe 04 temperature is: Error getting temperature

Probe 05 temperature is: Error getting temperature

Now try plugging in 2 or 3 sensors at once, like this:

Now more sensors should show temperature readings. Wait about 5 minutes for them to stabilize and they should all be within 0.5 Degrees C.



## Test Sketch to read DS18B20 addresses:

(Cut and paste from Serial Monitor)

```
/* YourDuino Example: Find Address of a DS18B20 Temperature Sensor
Cut and paste the address to a text file for later use.
V1.1 01/17/2013
Questions: terry@yourduino.com

Connections:
DS18B20 Pinout (Left to Right, pins down, flat side toward you)
- Left   = Ground
- Center = Signal (Pin 2): (with 3.3K to 4.7K resistor to +5 or 3.3 )
- Right  = +5 or +3.3 V
This sketch looks for 1-wire devices and prints their addresses (serial number)
to the Serial Monitor in a format that is useful in Arduino sketches.
Based on example at:
http://www.hacktronics.com/Tutorials/arduino-1-wire-address-finder.html
*/

/*-----( Import needed libraries )-----*/
#include <OneWire.h>

/*-----( Declare Constants and Pin Numbers )-----*/
#define SENSOR_PIN 2 // Any pin 2 to 12 (not 13) and A0 to A5

/*-----( Declare objects )-----*/
OneWire ourBus(SENSOR_PIN); // Create a 1-wire object

void setup() /****** SETUP: RUNS ONCE *****/
{
  Serial.begin(9600);

  discoverOneWireDevices(); // Everything happens here!
}

void loop() /****** LOOP: RUNS CONSTANTLY *****/
{
  // Nothing happening here
}

/*-----( Declare User-written Functions )-----*/
void discoverOneWireDevices(void) {
  byte i;
  byte present = 0;
  byte data[12];
  byte addr[8];

  Serial.print("Looking for 1-Wire devices...\n\r");// "\n\r" is NewLine
  while(ourBus.search(addr)) {
    Serial.print("\n\r\n\rFound '1-Wire' device with address:\n\r");
    for( i = 0; i < 8; i++) {
      Serial.print("0x");
      Serial.print(addr[i], HEX);
      if(i < 7) Serial.print(" ");
    }
    Serial.println();
  }
}
```