```
//    This program is part of the microstill project.
//    For details and instructions visit <http://www.microstill.net>.
//
//    This program is free software: you can redistribute it and/or modify
//    it under the terms of the GNU General Public License as published by
//    the Free Software Foundation, either version 3 of the License, or
//    (at your option) any later version.
//
//    This program is distributed in the hope that it will be useful,
//    but WITHOUT ANY WARRANTY; without even the implied warranty of
//    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
//    GNU General Public License for more details.
//
//    You should have received a copy of the GNU General Public License
//    along with this program.  If not, see <http://www.gnu.org/licenses/>.

#include <OneWire.h>
#include<DallasTemperature.h>

// Data wire is plugged into port 8 on the Arduino
#define ONE_WIRE_BUS 8
#define TEMPERATURE_PRECISION 9

// Setup a oneWire instance to communicate with any OneWire devices (not just
Maxim/Dallas temperature ICs)
OneWire oneWire(ONE_WIRE_BUS);

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

// arrays to hold device addresses
DeviceAddress thermometer;

void setup(void)
{
  // start serial port
 Serial.begin(9600);

  // Start up the library
 sensors.begin();

  // locate devices on the bus
 Serial.print("Locating devices...");
 Serial.print("Found ");
 Serial.print(sensors.getDeviceCount(), DEC);
 Serial.println(" devices.");

 if (!sensors.getAddress(thermometer, 0)) Serial.println("Unable to find address
for Device 0");
```

```
  // show the addresses we found on the bus
  Serial.println("Device 0 Address: ");
 printAddress2(thermometer);
  Serial.println();


}

// function to print a device address
void printAddress(DeviceAddress deviceAddress)
{
  for (uint8_t i = 0; i < 8; i++)
  {
    // zero pad the address if necessary
    if (deviceAddress[i] < 16) Serial.print("0");
   Serial.print(deviceAddress[i], HEX);
  }
}

// function to print a device address
void printAddress2(DeviceAddress deviceAddress)
{
  Serial.print("dsaddrH[0] = 0x");
  if (deviceAddress[0] < 16) Serial.print("0");
 Serial.print(deviceAddress[0], HEX);
 Serial.println(";");
  Serial.print("dsaddrH[1] = 0x");
  if (deviceAddress[1] < 16) Serial.print("0");
 Serial.print(deviceAddress[1], HEX);
 Serial.println(";");
  Serial.print("dsaddrH[2] = 0x");
  if (deviceAddress[2] < 16) Serial.print("0");
 Serial.print(deviceAddress[2], HEX);
 Serial.println(";");
  Serial.print("dsaddrH[3] = 0x");
  if (deviceAddress[3] < 16) Serial.print("0");
 Serial.print(deviceAddress[3], HEX);
 Serial.println(";");
  Serial.print("dsaddrH[4] = 0x");
  if (deviceAddress[4] < 16) Serial.print("0");
 Serial.print(deviceAddress[4], HEX);
 Serial.println(";");
  Serial.print("dsaddrH[5] = 0x");
  if (deviceAddress[5] < 16) Serial.print("0");
 Serial.print(deviceAddress[5], HEX);
 Serial.println(";");
  Serial.print("dsaddrH[6] = 0x");
  if (deviceAddress[6] < 16) Serial.print("0");
```

```
 Serial.print(deviceAddress[6], HEX);
 Serial.println(";");
  Serial.print("dsaddrH[7] = 0x");
  if (deviceAddress[7] < 16) Serial.print("0");
 Serial.print(deviceAddress[7], HEX);
 Serial.println(";");
}


void loop(void)
{

}
```