Medical Technology

Session 2
Physiological Monitoring

28 September 2017

Researchers leading this session:
Joanna Brunker
Tom Else
Calum Williams

County Upper School
The Bury Trust

This hand-out belongs to:

Examples of physiological monitoring

Physiological monitoring involves measuring certain properties of the body. These include:

- Blood flow
- Air flow
- Blood pressure
- Lung volume
- Temperature
- Blood oxygen level
- Muscle movement
- Gait
- Heart rate

There are many medical technologies available for monitoring different properties of the body.
Three examples are shown below:

Doppler ultrasound

for detecting
blood flow

Plethysmography

for detecting
lung volume

Pulse oximeter

for detecting
pulse rate and
blood oxygen

Reading a photoresistor

Introduction:

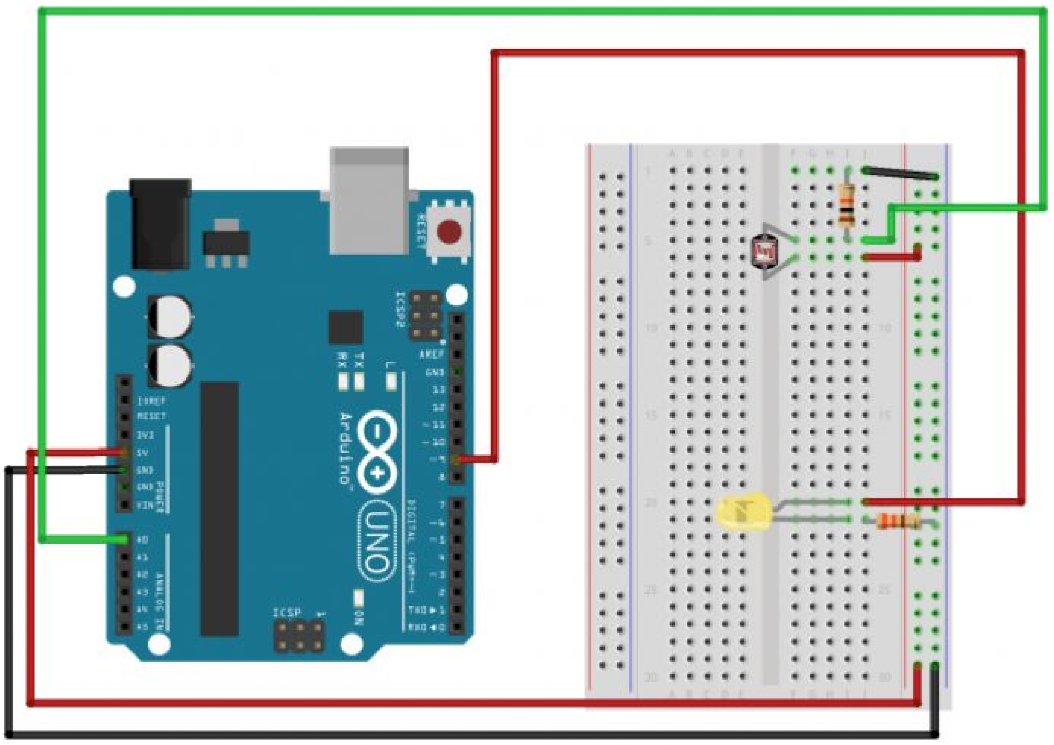
In this circuit, you'll be using a photoresistor, which changes resistance based on how much light the sensor receives. Since the Arduino Uno R3 can't directly interpret resistance (rather, it reads voltage), we need to use a **voltage divider** to use our photoresistor. This voltage divider will output a high voltage when it is getting a lot of light and a low voltage when little or no light is present.

Parts needed:

You will need the following parts:

- 1x Breadboard
- 1x Arduino Uno
- 1x LED
- 1x 560Ω Resistor (green-blue-brown)
- 6x Jumper Wires
- 1x Photoresistor
- 1x 10k Resistor (brown-black-orange)

Use the diagram below to help you connect your circuit:



Open Up the Arduino IDE software on your computer.
Coding in the Arduino language will control your circuit.

Go to the webpage:
http://ardx.org/src/circ/CIRC09-code.txt
Copy and paste the text into an empty Arduino sketch.

What You Should See

You should see the LED grow brighter or dimmer in accordance with how much light your photoresistor is reading. If it isn't working, make sure you have assembled the circuit correctly and verified and uploaded the code to your board or see the troubleshooting section.

Real World Application

Some street lamps as well as solar walkway lights use photoresistors to detect the absence of the sun and turn on the lights.

Troubleshooting

LED Remains Dark
This is a mistake we continue to make time and time again, if only they could make an LED that worked both ways. Pull it up and give it a twist.

It Isn't Responding to Changes in Light
Given that the spacing of the wires on the photoresistor is not standard, it is easy to misplace it. Double check it's in the right place.

Still Not Quite Working
You may be in a room which is either too bright or dark. Try turning the lights on or off to see if this helps. Or if you have a flashlight near by give that a try.

Making it better

Reverse the response:

Perhaps you would like the opposite response. Don't worry we can easily reverse this response just change:

```
    analogWrite(ledPin, lightLevel);
```

to

```
    analogWrite(ledPin, 255 - lightLevel);
```

Upload and watch the response change!

Night light:

Rather than controlling the brightness of the LED in response to light, let's instead turn it on or off based on a threshold value. Change the loop() code with:

```
    void loop(){
        int threshold = 300;
        if(analogRead(lightPin) > threshold){
        digitalWrite(ledPin, HIGH);
        }else{
        digitalWrite(ledPin, LOW);
        }
    }
```

Reading a temperature sensor

Introduction:

A temperature sensor is exactly what it sounds like – a sensor used to measure ambient temperature. This particular sensor has three pins – a positive, a ground, and a signal. This is a linear temperature sensor. A change in temperature of one degree centigrade is equal to a change of 10 millivolts at the sensor output.

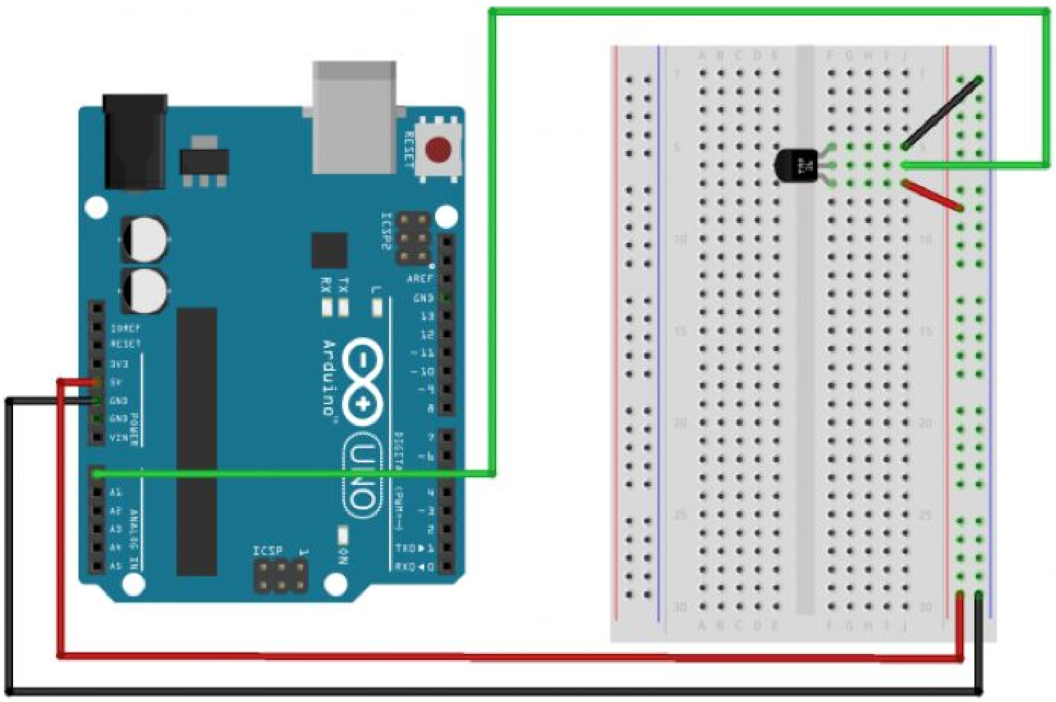
The TMP36 sensor has a nominal 750 mV at 25°C (about room temperature). In this circuit, you'll learn how to integrate the temperature sensor with your Arduino Uno R3, and use the Arduino IDE's serial monitor to display the temperature.

Parts needed:

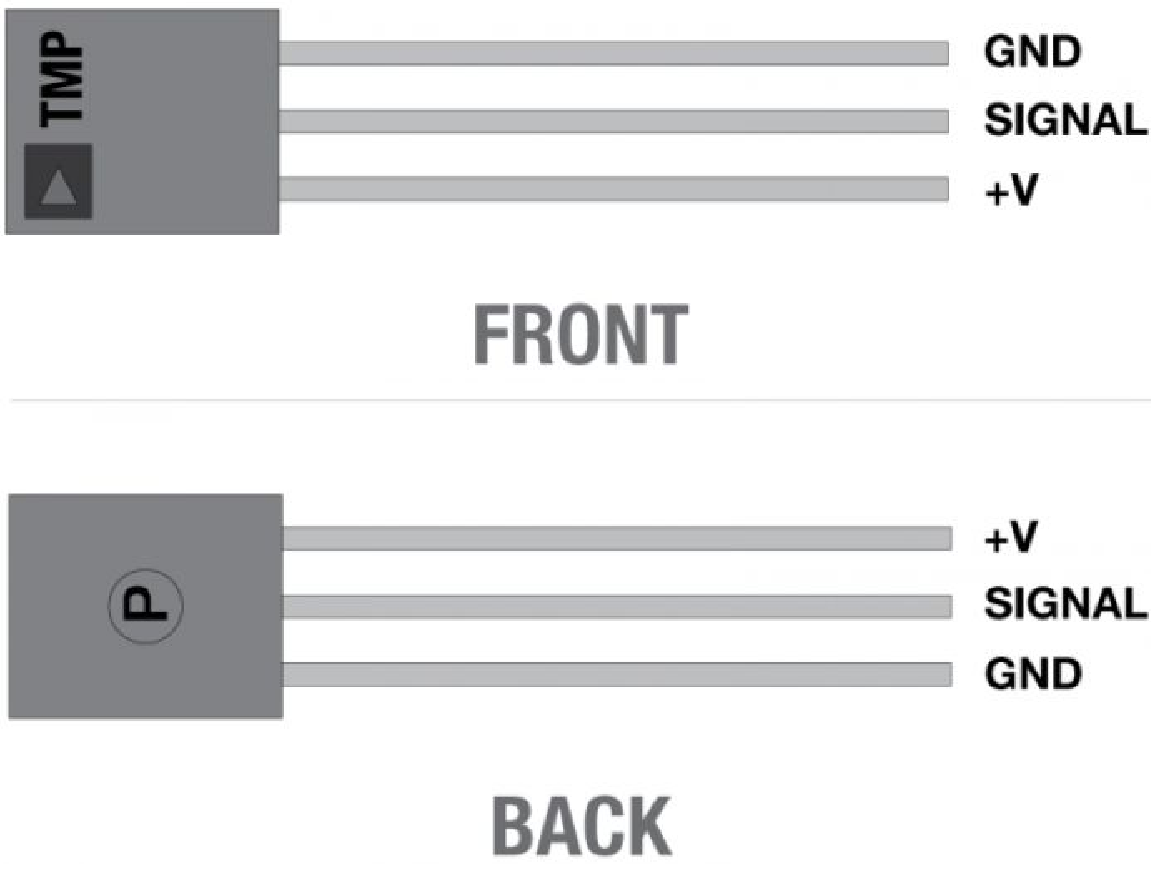
You will need the following parts:

- 1x Breadboard
- 1x Arduino Uno
- 5x Jumper Wires
- 1x Temperature Sensor

Use the diagram below to help you connect your circuit:



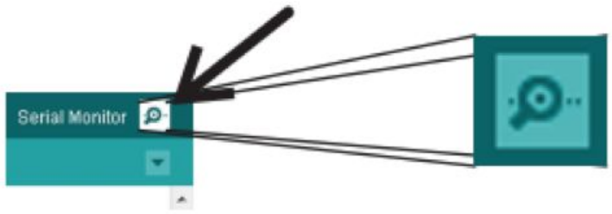
Please note: The temperature sensor can only be connected to a circuit in one direction. See below for the pin outs of the temperature sensor - TMP36



Open Up the Arduino IDE software on your computer.
Coding in the Arduino language will control your circuit.

Go to the webpage:
http://ardx.org/src/circ/CIRC10-code.txt
Copy and paste the text into an empty Arduino sketch.

One extra note, this circuit uses the Arduino IDE's serial monitor. To open this, first upload the program then click the button which looks like a magnifying glass or press (ctrl + shift + m)



Code to note

```
Serial.begin(9600);
```

Before using the serial monitor, you must call Serial.begin() to initialize it. 9600 is the "baud rate", or communications speed. When two devices are communicating with each other, both must be set to the same speed.

```
Serial.print()
```

will print everything on the same line.

```
Serial.println()
```

will move to the next line. By using both of these commands together, you can create easy-to-read printouts of text and data.

What You Should See

You should be able to read the temperature your temperature sensor is detecting on the serial monitor in the Arduino IDE. If it isn't working, make sure you have assembled the circuit correctly and verified and uploaded the code to your board or see the troubleshooting section.

Real World Application

Building climate control systems use a temperature sensor to monitor and maintain their settings.

Troubleshooting

Nothing Seems to Happen

This program has no outward indication it is working. To see the results you must open the Arduino IDE's serial monitor (instructions on previous page).

Gibberish is Displayed

This happens because the serial monitor is receiving data at a different speed than expected. To fix this, click the pull-down box that reads "*** baud" and change it to "9600 baud".

Temperature Value is Unchanging

Try pinching the sensor with your fingers to heat it up or pressing a bag of ice against it to cool it down.

Making it better

Outputting voltage:

This is a simple matter of changing one line. Our sensor outputs 10mv per degree centigrade so to get voltage we simply display the result of getVoltage().

```
    delete the line temperature = (temperature - .5) * 100;
```

Outputting degrees Fahrenheit:

Again this is a simple change requiring only maths. To go from degrees C to degrees F we use the formula:

```
    ( F = C * 1.8) + 32 )
```

add the line

```
    temperature = (((temperature - .5) * 100)*1.8) + 32;
```

before Serial.println(temperature);

More informative output:

Let's add a message to the serial output to make what is appearing in the Serial Monitor more informative. To do this first revert to the original code then change:

```
    Serial.println(temperature);
```

To

```
    Serial.print(temperature);
    Serial.println(" degrees centigrade");
```

The change to the first line means when we next output it will appear on the same line, then we add the informative text and a new line.

Changing the serial speed:

If you ever wish to output a lot of data over the serial line time is of the essence. We are currently transmitting at 9600 baud but much faster speeds are possible. To change this change the line:

```
    Serial.begin(9600);
```

To

```
    Serial.begin(115200);
```

Upload the sketch turn on the serial monitor, then change the speed from 9600 baud to 115200 baud in the pull down menu. You are now transmitting data 12 times faster.

Notes