Arduino Breathalyser

Although we call this one a breathalyser, the sensor we're using is not considered accurate enough to give useful readings. Still, when everyone at the office heard we had an alcohol sensor, we had to have a go at building one. We've gone with a compact form factor- a Duinotech Nano and a Linker 4-digit 7-segment display. It's all just free-wired, but should fit into a small enclosure if you want to make something more permanent.

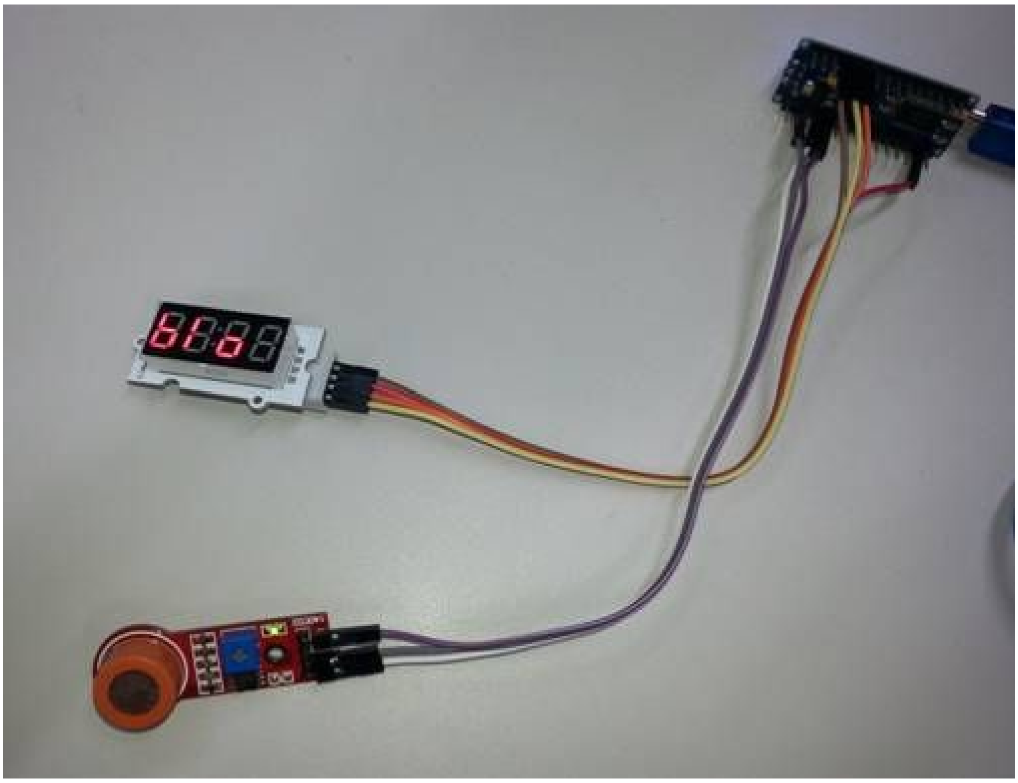
Components

XC4414 duinotech Nano

XC4540 Alcohol Sensor Module

XC4569 Linker 4-digit 7-segment Display

WC6026 Socket-socket Jumper Leads



Connections

Wire up as per the picture above, details in the table below. Note that the two modules each need a ground connection. We've used the ground connection near the digital pins for the display, and the one near the analog pins for the alcohol sensor. Even though the display module is designed to take a linker cable, it works well with the jumper leads too.

Duinotech Nano	Display	Alcohol Sensor	Function
5V		VCC	Power
GND	GND	GND	Ground
3.3V	VCC		Power
D2	DIO		Data to display
D3	CLK		Clock signal for display
A7		A0	Analog signal for Alcohol Sensor

Arduino Sketch

The Linker 4-digit 7-segment display needs a library to be able to work, and this can be downloaded from https://github.com/avishorp/TM1637. This library has some handy functions that make it very easy to run the display. Install the library, open the IDE, then choose Nano (328) as the board, and upload the sketch.

The sketch operates works by cycling through a few different modes. The first mode displays 'INI', and an icon that cycles. During this stage, the sketch is waiting for the reading on the alcohol sensor to stabilise. The sensor will typically require being powered for up to 24 hours on first use to evaporate the manufacturing residues on the internals of the sensor, and then up to a minute for subsequent uses.

After this, the sketch moves into ready mode, where it displays 'BLO'. When the sketch detects a sudden change in the reading due to being breathed on, it starts a counter icon to allow the air around the sensor to stabilise before making its calculation.

The final phase is where the Breathalyser displays the measured reading. It does this for 10 seconds, showing ‘BA:’, following by the last two digits of the reading, eg 0.05% would be displayed as ‘BA:05’. The 10 second delay also allows time for the sensor to recover back to a resting state. After this, the sketch will move to the ‘INI’ stage in case the sensor hasn’t fully stabilised, and the cycle continues.

Improvements

Although the sensor is nowhere near precise and repeatable enough to be used as a reliable Breathalyser (readings also vary with temperature and humidity, for example), it would be interesting to compare the readings to that taken from a commercial breathalyser.

Another option would be to turn the project into a portable unit, perhaps by adding a case and batteries. A 9V battery would probably only run for an hour or so, because the alcohol sensor uses quite a bit if power, so you could use a PH9206 6xAA Battery holder, and feed that into the VIN (+) and GND (-) pins on the Nano.

Code

```
#include <TM1637Display.h>

// Module connection pins
#define CLK 2
#define DIO 3
#define MQPIN A7

TM1637Display display(CLK, DIO);
int n = 0;
const uint8_t MSG_BLOW[] = {124, 48, 92, 0};    // blow message
const uint8_t MSG_INIT[] = {6, 55, 48, 0};      // ini message
const uint8_t MSG_BA[] = {127, 247, 0, 0};      // BA: message
uint8_t data[] = { 0xff, 0xff, 0xff, 0xff };    // blank message for animations

void setup()
{
   display.setBrightness(0x0f);
   display.setSegments(MSG_INIT);
   stabilise();
}
```

```
void loop()
{
    display.setSegments(MSG_BLOW);          //ready, show blow message

    int a = analogRead(MQPIN);
    int b = a;

    // wait for a large change
    while (abs(a-b) < 10)
    {
        b = a;
        a = analogRead(MQPIN);
        delay(300);
    }

    // wait a bit
    for (int i = 1; i < 64; i = i<<1)
    {
        data[0] = i - 1;
        display.setSegments(data, 1, 3);    // filling circle pattern
        delay(500);
    }

    n = analogRead(MQPIN) - b;              // get reading above stable background

    //avoid invalid values near zero
    if (n < 0)
    {
        n = 0;
    }

    n = n / 10;                             // scale
    display.setSegments(MSG_BA, 2, 0);      // BA:
    display.showNumberDec(n, 1, 2, 2);      // display analog value
    delay(10000);                           // show reading, and let sensor stabilise
    stabilise();
}
```

```
void stabilise()
{
   data[0] = 1;
   int a;
   int b = 0;
   a=analogRead(MQPIN);
   display.setSegments(MSG_INIT);
   delay(500);

   // wait for analog input to settle
   while (abs(a-b) > 0)
   {
       b = a;
       a = analogRead(MQPIN);
       data[0] = data[0]<<1;          // cycle pattern so it looks like its doing something
       if (data[0] > 32)
       {
           data[0] = 1;
       }
       display.setSegments(data, 1, 3);
       delay(500);
   }
}
```

Links

Source:

https://www.jaycar.com.au/diy-arduino-breathalyser

Code:

https://github.com/ZennonDamant/arduino/blob/master/Breathalyser.ino