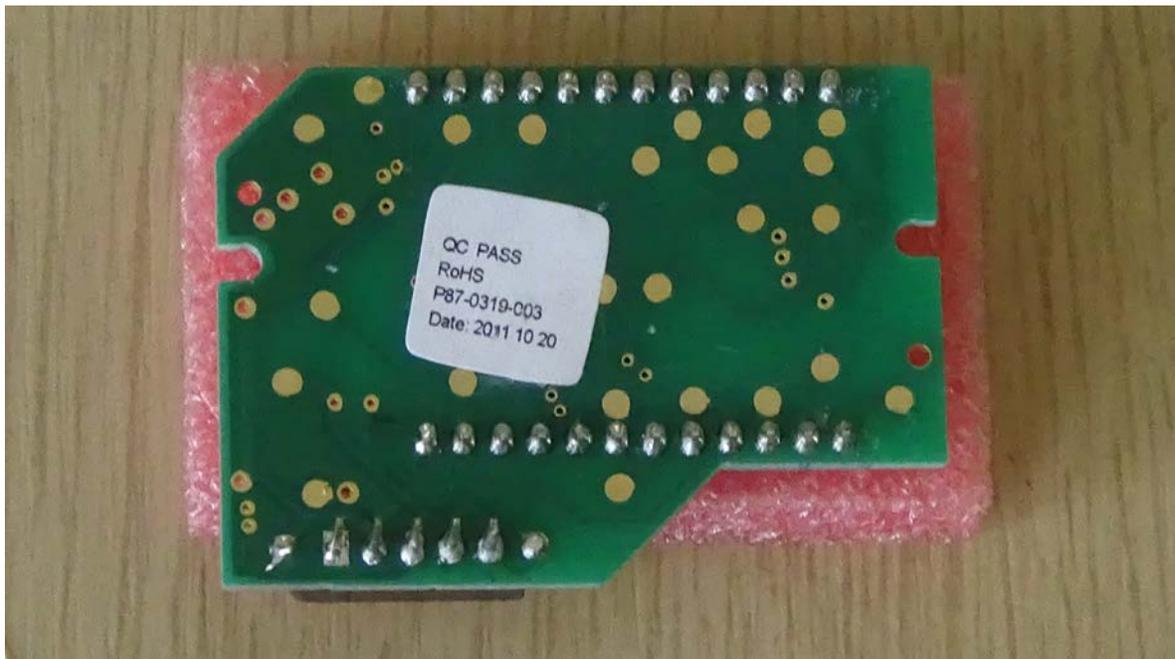
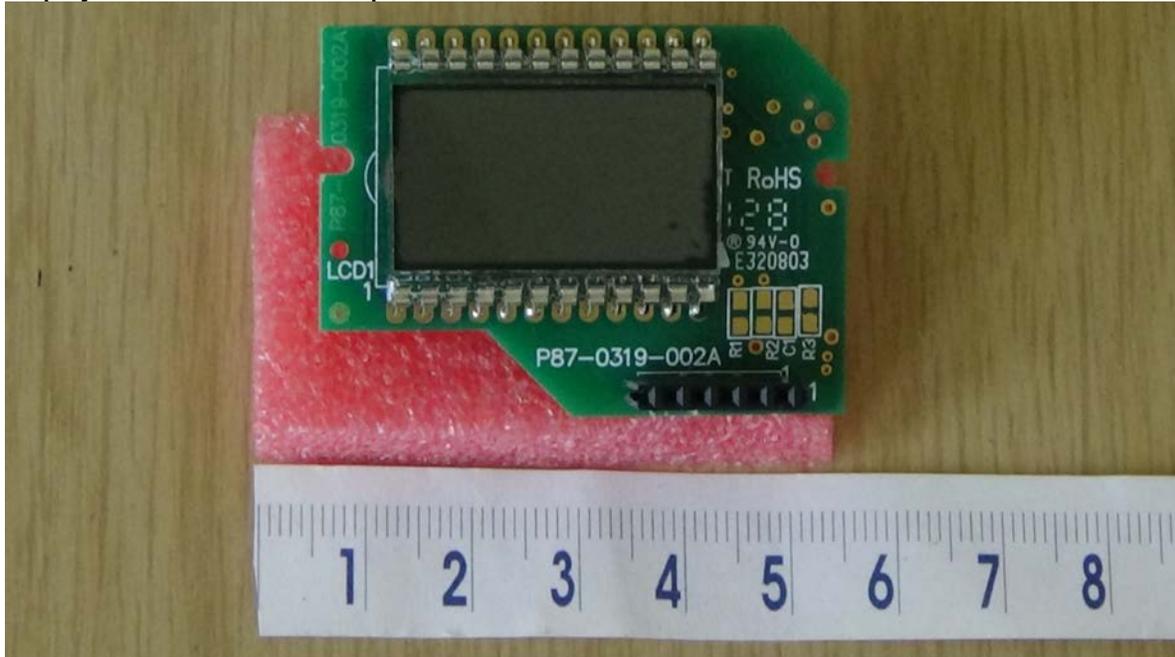


## Salvage display from end-of-life CO alarm

Carbon monoxide alarms were mandated for new construction in 2011, with a sensor life expectancy of seven years. The two First Alert units I bought about that time have now failed as predicted.

I took them apart and didn't find too many parts for robots, but the 3-digit, 7-segment display was mounted on a separate breakout board.



It had only four wires for an I2C interface, soldered into the standard 0.1" thru-holes on one edge. (Actually five wires: SDA, SCL, GND, +5V, +5V.) I soldered headers for breadboard use.

I kept one unit powered with a 9-V battery in order to read the I2C address and command bytes, and to figure out the bit mapping to the segments. (The unit was constantly displaying "End"..)

I was able to drive the I2C lines directly from an Arduino with no external pull-ups. I used a simple routine with the Wire library to display hex and decimal numbers along with the two decimal points.

The alarms were Model CO614. This may also be true for other First Alert models. I'll be happy to answer questions. This Arduino code is was tested on a UNO:

```
/*
  This tests 3-digit LCDs from old First Alert Carbon Monoxide alarms (CO614).
  The I2C communication is from master UNO to slave LCD only. No ACK is done.
  All possible decimal combinations of three digits are tested.
*/

#include <Wire.h>

byte zero = 0xEE, one = 0x28, two = 0xCD, three = 0x6D, four = 0x2B;
byte five = 0x67, six = 0xE7, seven = 0x2C, eight = 0xEF, nine = 0x2F;
byte dec[10] = {zero, one, two, three, four, five, six, seven, eight, nine};
byte x = 0;

void setup()
{
  Wire.begin(); // join i2c bus (address optional for master)
}

void loop()
{
  Wire.beginTransmission(0x3E); // transmit to device 0x3E
  Wire.write(0xE0);
  Wire.write(0xCD);
  Wire.write(0x00);
  Wire.write(dec[x]);
  Wire.write(dec[x + 1]);
  Wire.write(dec[x + 2]);
  Wire.endTransmission(); // stop transmitting
  x = x + 1;
  delay(1000);
  if (x >= 8)
    x = 0;
}
```

John F. Davis