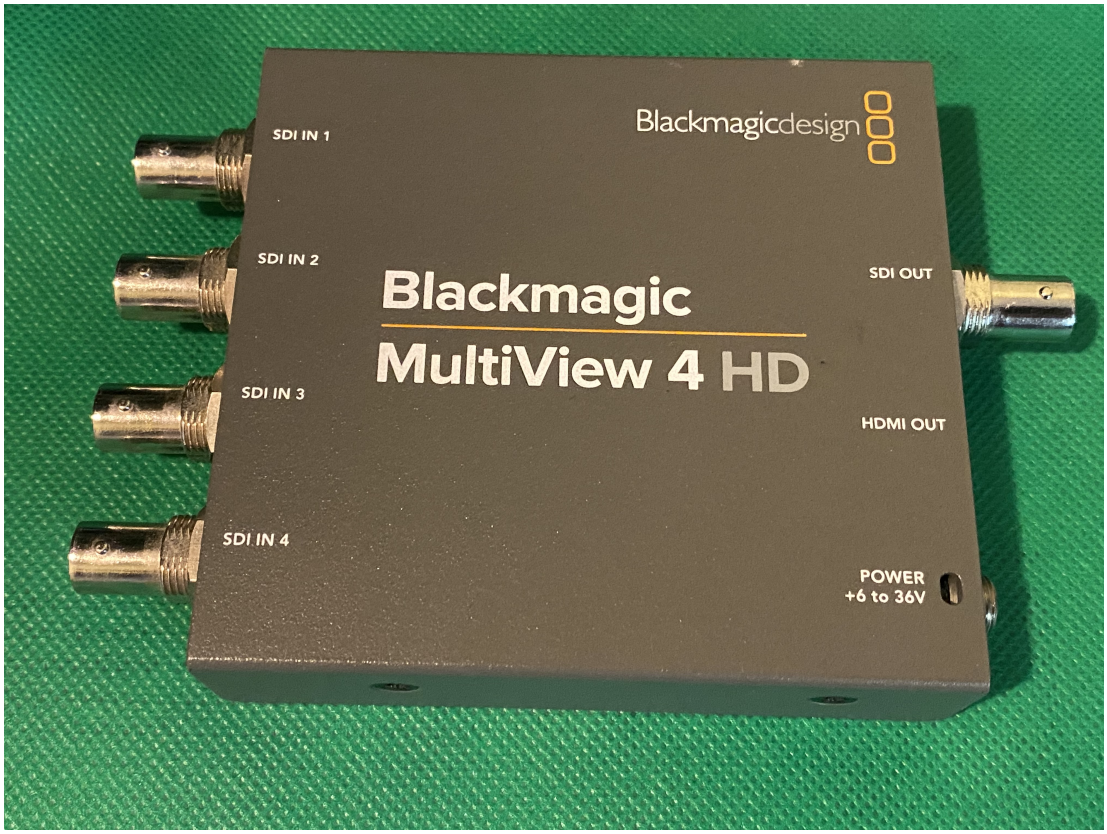
Modifying Blackmagic MicroConverters for network control

By Baz Leffler

This tutorial explains how to interface a microcontroller to Blackmagic MicroConverters. The microController used here is an Arduino but can be any microcontroller that has an ethernet port and 8 bi-directional control pins. The only controls available are those that are available as dip switches on the Blackmagic device.

In this example I am using a Blackmagic Multiviewer 4HD to replace a Teranex Multiviewer 4 that I was using as a CCU monitor with my 4 x CCU controller. My 4 x CCU controller uses the Blackmagic protocol published in their users manual.



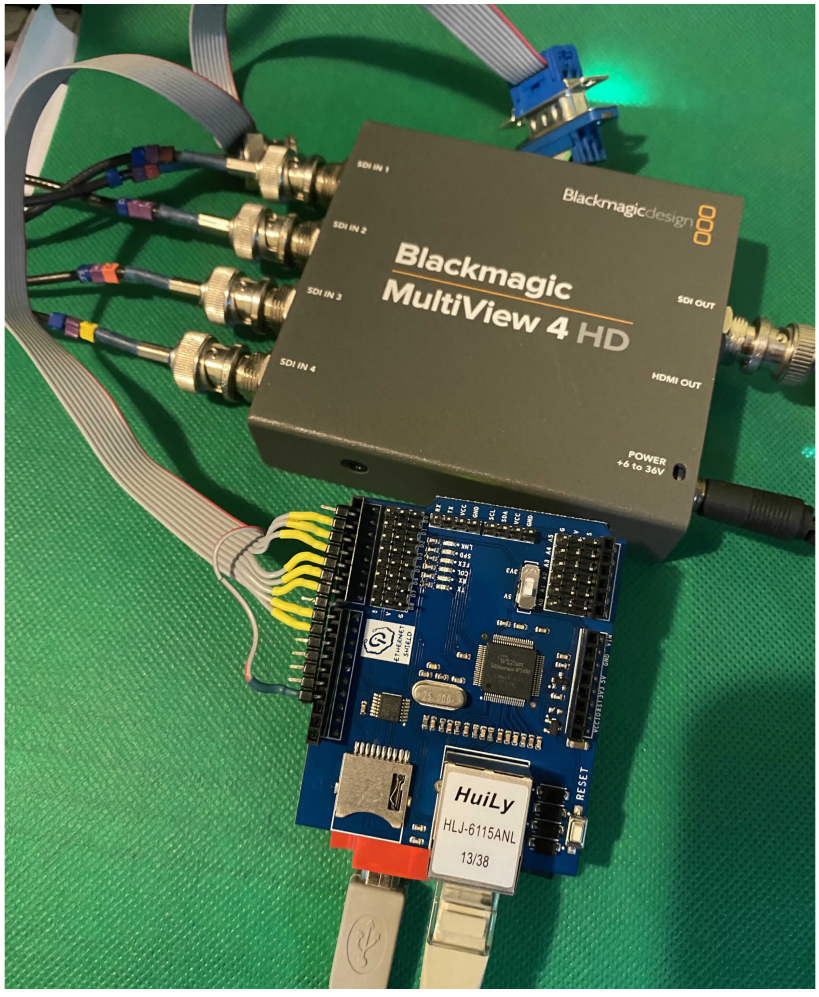
Please note this modification will void your manufacturer's warranty so I am doing it as my warranty has expired. Also note I take no responsibility if anything goes wrong.

This modification requires all dip switches to be set in the OFF position but if set to on will override the microcontroller. The Blackmagic Multiviewer 4HD will still perform normally both when connected to the Arduino and disconnected.

I have used both an Arduino UNO with W5100 ethernet shield and an Arduino Nano with a Nano V3 ethernet shield. It uses a 9 wire ribbon cable wired to a 9 pin ‘D’ connector and the pin connections are as follows:

Arduino	function	9p ‘D’	switch (solder)
Ground	ground	1	1 off position
2	3G level A/B	6	1 on position
3	Borders ON/OFF	2	2 on position
4	Labels ON/OFF	7	3 on position
5	Audio Meters ON/OFF	3	4 on position
6	SDI Tally ON/OFF	8	5 on position
7	Solo ON/OFF	4	6 on position
8	Source 1 – 4 select a	9	7 on position
9	Source 1 – 4 select b	5	8 on position

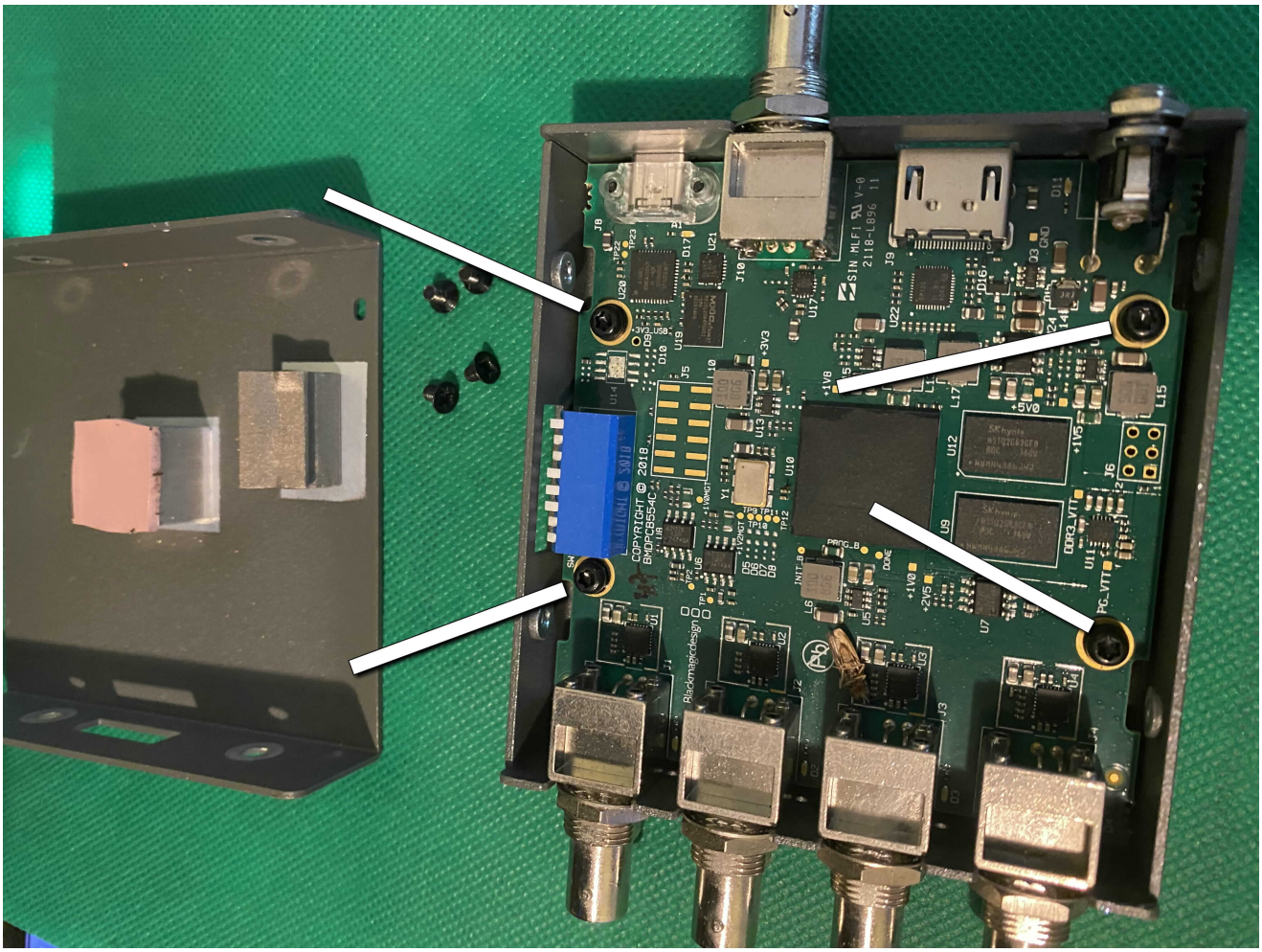
The following is how to add the 9 pin ‘D’ connector to a Blackmagic MicroConverter. Once completed you will have the following:



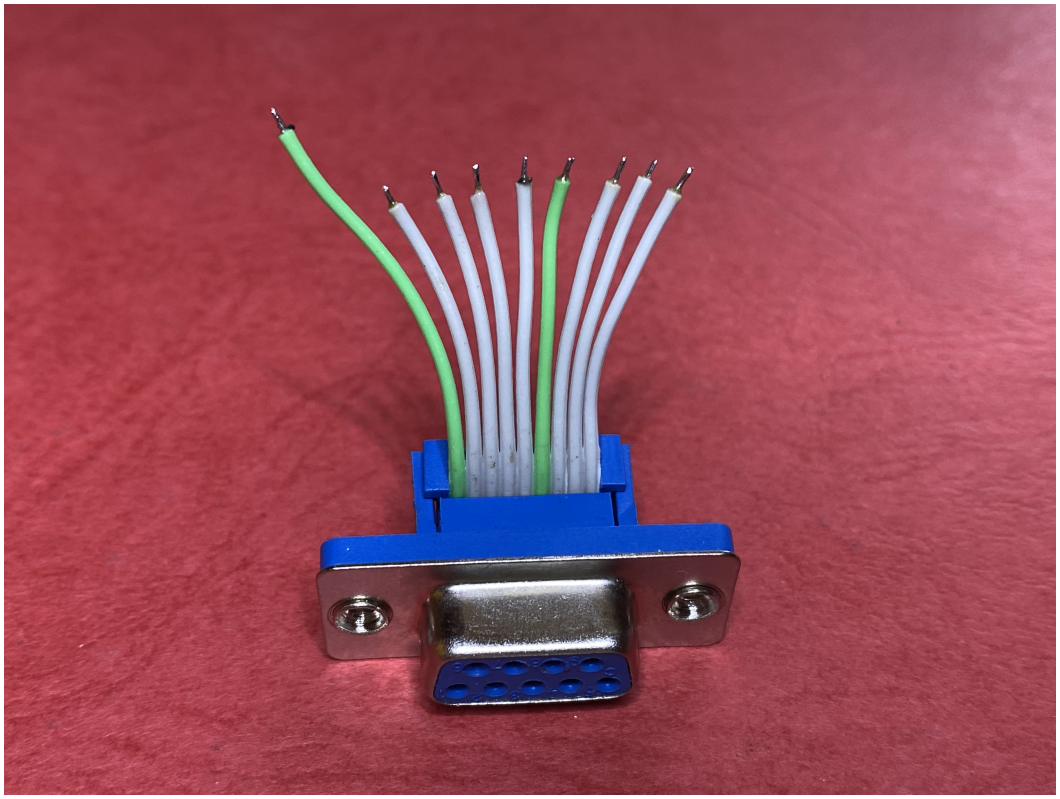
At the bottom is the Arduino Uno and W5100 ethernet shield with a USB connection for power/programming, an ethernet connection and a breakout cable for the control functions. At the top is the ‘floating’ 9 pin ‘D’ connector attached to the Blackmagic Multiviewer 4HD.

To add the modification to the Blackmagic Multiviewer 4HD you will need to take the cover off by removing the 4 countersunk black screws (goodbye warranty).

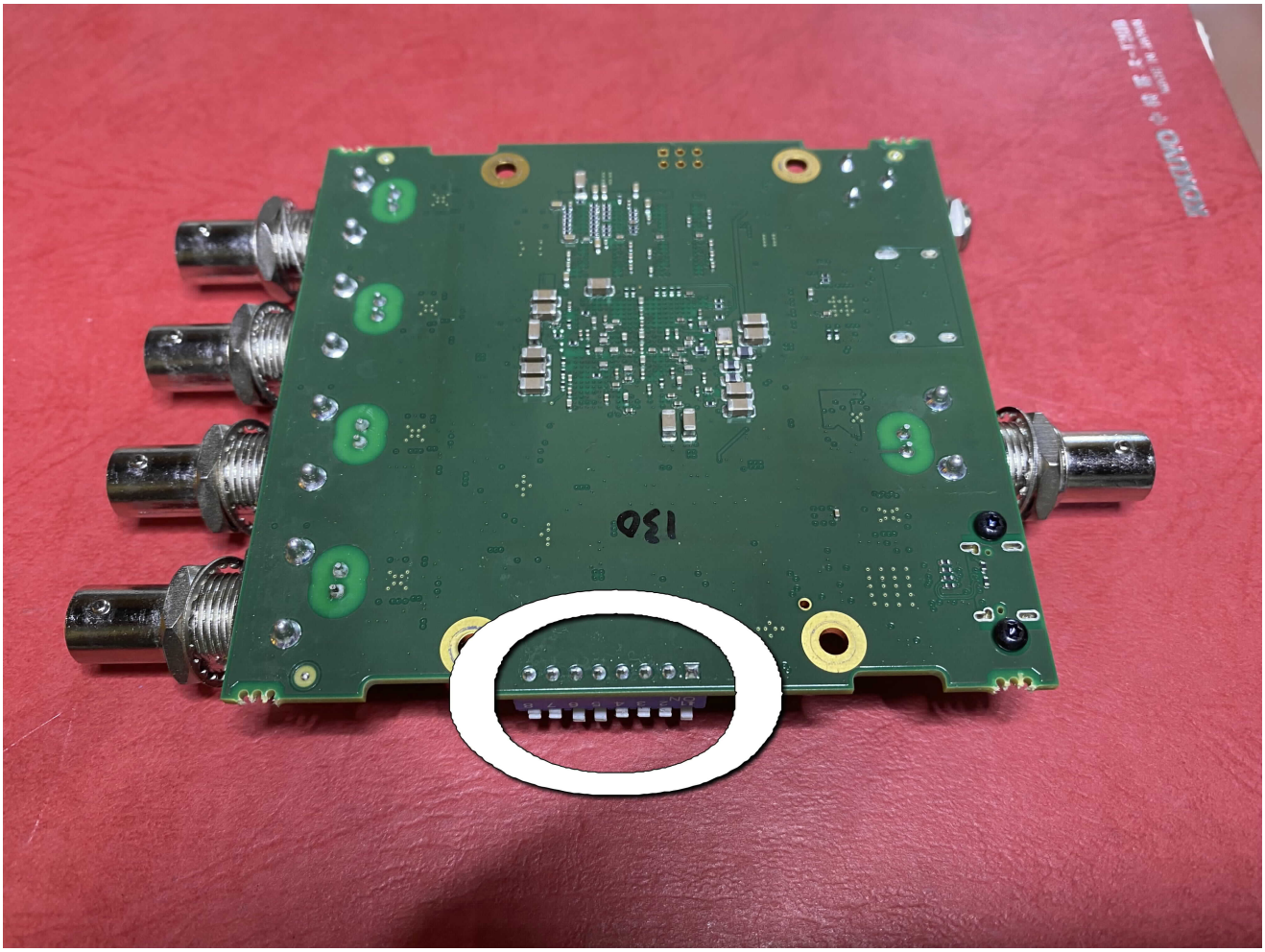
Next remove these 4 star screws marked in white:



Make an open ended 9 pin cable as shown here:

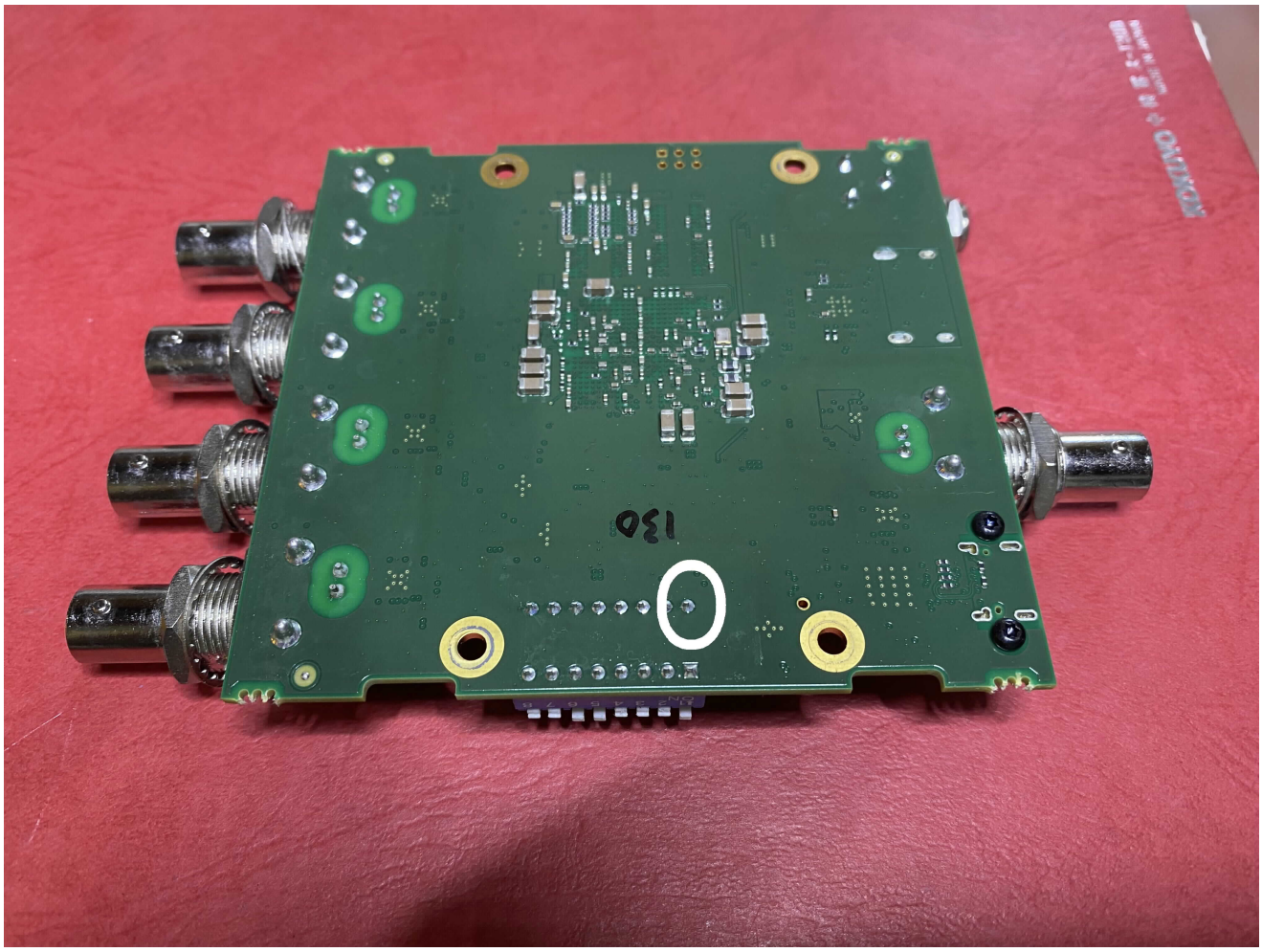


Note the 1st cable is connected to pin 1 and is about 8mm longer.

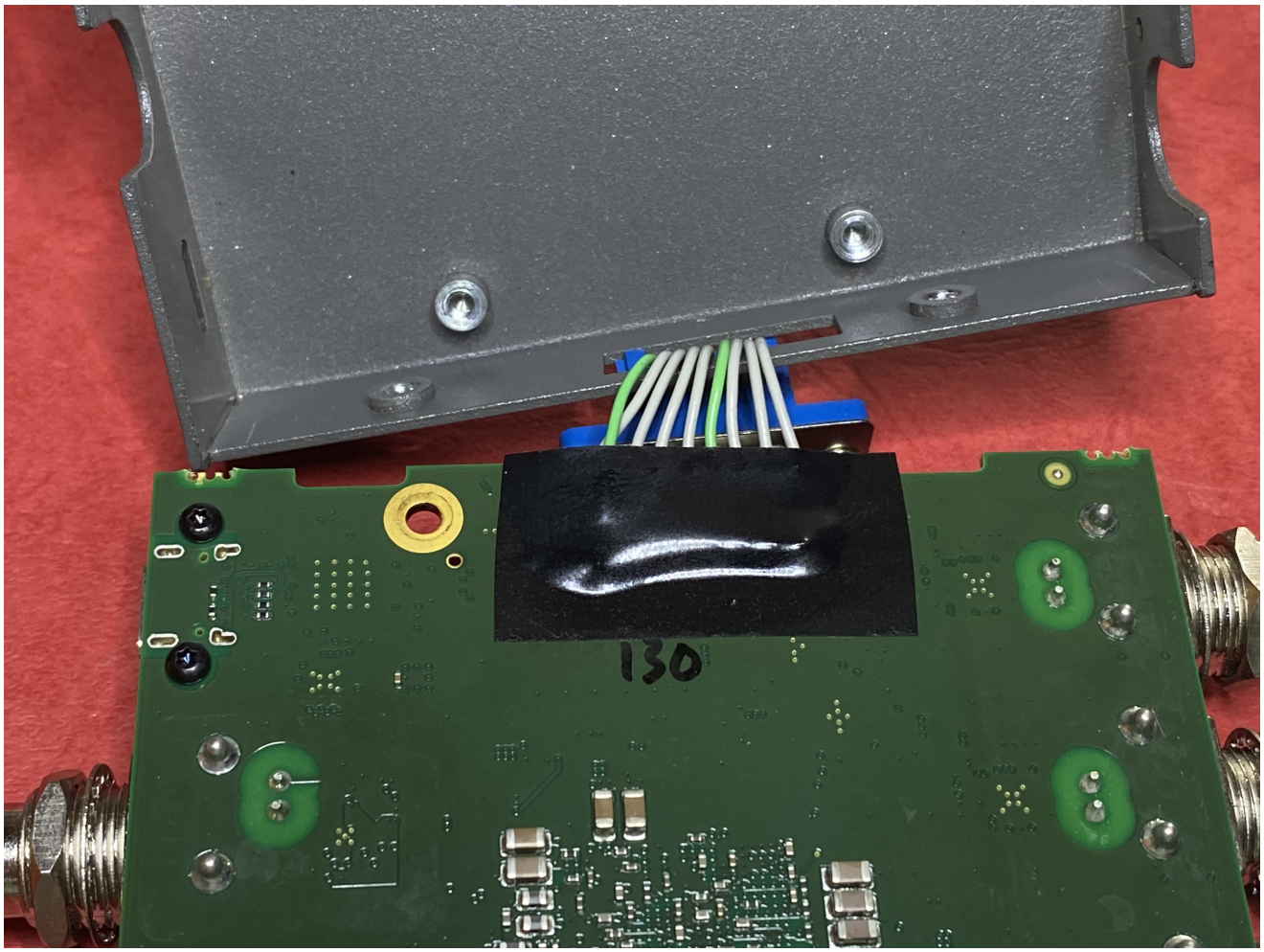


Now thread the ribbon thru the cover and solder the wires to the following switch pins as marked above:

9p #6 = Multiview switch 1 9p #2 = Multiview switch 2 9p #7 = Multiview switch 3 9p #3 = Multiview switch 4 9p #8 = Multiview switch 5 9p #4 = Multiview switch 6 9p #9 = Multiview switch 7 9p #5 = Multiview switch 8



Next solder the longer wire to the pin as circled above.

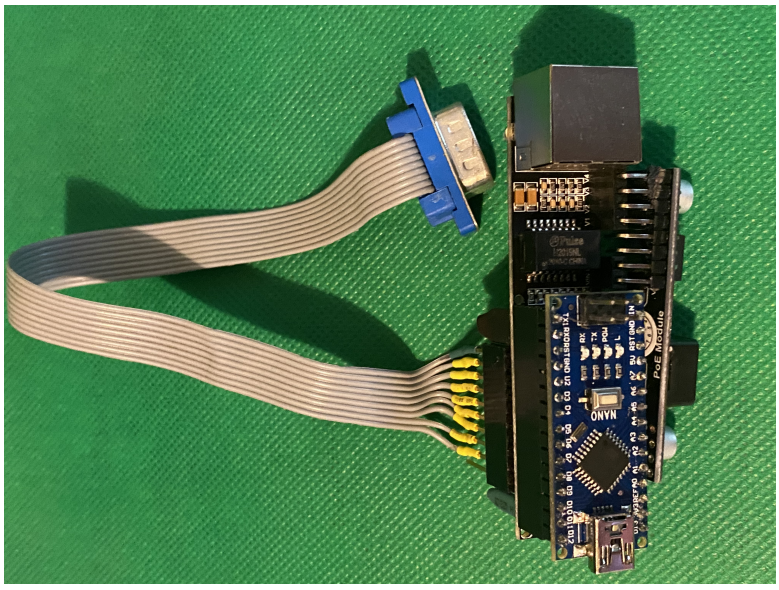


Run some electrical insulation tape across the soldered wires. Note how the ribbon is threaded thru the cover.

You can now reassemble the Blackmagic Multiviewer 4HD back to normal. You will notice where the ribbon cable exits the box it is hard up against the circuit board which acts as a clamp to prevent it from being strained but you may like to add your own strain relief where necessary. You should end up with the following:

Hopefully you should have retained access to the dip switches for stand alone operation.

Here is the Arduino Nano with the Nano V3 ethernet shield version.



Note that this unit also has a POE (power over ethernet) module attached.

To write your own code all you need are the associated libraries and the following defines:

```
#include <SPI.h>

// **** W5100 *****

#include <Ethernet.h>
#include <EthernetUdp.h>

// ***** W5500 *****

//#include <Ethernet3.h>
//#include <EthernetUdp3.h>
#include <EEPROM.h>
#include <Wire.h>
                                                    // 9p #1 = ground
#define LEVELAB          2                          // 9p #6 = Multiview switch 1
#define BORDERS          3                          // 9p #2 = Multiview switch 2
#define LABELS           4                          // 9p #7 = Multiview switch 3
#define AUDIOMETERS      5                          // 9p #3 = Multiview switch 4
#define SDITALLY         6                          // 9p #8 = Multiview switch 5
#define SOLO             7                          // 9p #4 = Multiview switch 6
#define AUDIOA           8                          // 9p #9 = Multiview switch 7
#define AUDIOB           9                          // 9p #5 = Multiview switch 8

void setup()
{   // set hold up on all pins
digitalWrite(LEVELAB, HIGH);                        // Multiview switch 1
digitalWrite(BORDERS, HIGH);                        // Multiview switch 2
digitalWrite(LABELS, HIGH);                         // Multiview switch 3
digitalWrite(AUDIOMETERS, HIGH);                    // Multiview switch 4
digitalWrite(SDITALLY, HIGH);                       // Multiview switch 5
digitalWrite(SOLO, HIGH);                           // Multiview switch 6
digitalWrite(AUDIOA, HIGH);                         // Multiview switch 7
digitalWrite(AUDIOB, HIGH);                         // Multiview switch 8

. . . . . .

// Menu items

Serial.println(F("press:"));
Serial.println(F("\t0 = Quad Split"));
```

```
Serial.println(F("\t1 = Solo 1"));
Serial.println(F("\t2 = Solo 2"));
Serial.println(F("\t3 = Solo 3"));
Serial.println(F("\t4 = Solo 4"));
Serial.println(F("\tt = Toggle Tally"));
Serial.println(F("\tm = Toggle Audio Meters"));
Serial.println(F("\tm = Toggle Labels"));
Serial.println(F("\tb = Toggle borders"));
Serial.println(F("\tA = Level A"));
Serial.println(F("\tB = Level B"));void loop()

{
. . . . . .

if (Host.available())
{
 SetPortDirection(OUTPUT);                          // set to output command
 switch (Host.read())
 {
 case '0':                                  // Quad Split
   digitalWrite(SOLO, HIGH);                // 9p #4 = Multiview switch 6
   break;

 case '1':                                  // Solo 1
   digitalWrite(SOLO, LOW);                 // 9p #4 = Multiview switch 6
   digitalWrite(AUDIOA, HIGH);              // 9p #9 = Multiview switch 7
   digitalWrite(AUDIOB, HIGH);               // 9p #5 = Multiview switch 8
   break;

case '2':                                   // Solo 2
   digitalWrite(SOLO, LOW);                 // 9p #4 = Multiview switch 6
   digitalWrite(AUDIOA, LOW);               // 9p #9 = Multiview switch 7
   digitalWrite(AUDIOB, HIGH);              // 9p #5 = Multiview switch 8
   break;

 case '3':                                  // Solo 3
    digitalWrite(SOLO, LOW);                // 9p #4 = Multiview switch 6
    digitalWrite(AUDIOA, HIGH);             // 9p #9 = Multiview switch 7
    digitalWrite(AUDIOB, LOW);              // 9p #5 = Multiview switch 8
    break;

  case '4':                                 // Solo 4
     digitalWrite(SOLO, LOW);               // 9p #4 = Multiview switch 6
     digitalWrite(AUDIOA, LOW);             // 9p #9 = Multiview switch 7
```

```
      digitalWrite(AUDIOB, LOW);                  // 9p #5 = Multiview switch 8
      break;

    case 't':                                     // Toggle Tally (Multiview switch 5)
       digitalRead(SDITALLY) ? digitalWrite(SDITALLY, LOW) : digitalWrite(SDITALLY, HIGH);
       break;

    case 'm':                                      // Toggle Audio  (Multiview switch 4)
       digitalRead(AUDIOMETERS) ? digitalWrite(AUDIOMETERS, LOW) :
                                                        digitalWrite(AUDIOMETERS, HIGH);

        break;

    case 'l':                                     // Toggle Labels (Multiview switch 3)
        digitalRead(LABELS) ? digitalWrite(LABELS, LOW) : digitalWrite(LABELS, HIGH);
        break;

      case 'b':                                    // Toggle borders (Multiview switch 2)
       digitalRead(BORDERS) ? digitalWrite(BORDERS, LOW) : digitalWrite(BORDERS, HIGH);
       break;

     case 'A':                                    // Level A (Multiview switch 1)
       digitalWrite(LEVELAB, LOW);
       break;

      case 'B':                                    // Level B (Multiview switch 1)
        digitalWrite(LEVELAB, HIGH);
        break;

     }


void SetPortDirection(boolean direction)

{
  pinMode(LEVELAB, direction);              // Multiview switch 1
  pinMode(BORDERS, direction);              // Multiview switch 2
  pinMode(LABELS, direction);               // Multiview switch 3
  pinMode(AUDIOMETERS, direction);      // Multiview switch 4
  pinMode(SDITALLY, direction);             // Multiview switch 5
  pinMode(SOLO, direction);                 // Multiview switch 6
  pinMode(AUDIOA, direction);               // Multiview switch 7
  pinMode(AUDIOB, direction);               // Multiview switch 8
  return;
}
```

As soon as I get time I will make the source code available including an user ready HEX file. With the source code you will be able to re configure it to work with all other types of dip switch controllable converters that use ‘switch ground’ configurations. You will also be able to do a HTML etc version. Please make any source code improvements available to all users.

Also keep a lookout for any updates I may have.