# EP4CE10E22C8N FPGA GPU CARD for the uCOM ASSEMBLY NOTES & DEBUGGING

APRIL 2020

Contents

[EP4CE10E22C8N FPGA GPU CARD for the uCOM ASSEMBLY NOTES & DEBUGGING 1](#_Toc43986454)

[Initial Assembly & Testing Steps 3](#_Toc43986455)

[Stage 1 – Testing Power Supply 3](#_Toc43986456)

[Stage 2 – Testing Clock Output 3](#_Toc43986457)

[Stage 3 – Testing AS and U7 4](#_Toc43986458)

[Stage 4 – Test basic FPGA and audio circuit operation 4](#_Toc43986459)

[Stage 5 – Test advanced FPGA operation 4](#_Toc43986460)

[Stage 6 – Test connection to uCOM 4](#_Toc43986461)

[Stage 7 – Test video output 4](#_Toc43986462)

[Stage 8 – Test peripherals 4](#_Toc43986463)

[Faults & Debugging 5](#_Toc43986464)

[Stage 6 – Cannot read GPU RAM 6](#_Toc43986465)

[Tests 6](#_Toc43986466)

[Solution: 7](#_Toc43986467)

[Stage 7 – No video output 8](#_Toc43986468)

[Investigation: 8](#_Toc43986469)

[Solution: 8](#_Toc43986470)

[Stage 7 – Missing green levels 9](#_Toc43986471)

[(Card 2) – Stage 6 – Incorrect data read/written to GPU RAM (bit 0) 10](#_Toc43986472)

[Investigation: 10](#_Toc43986473)

[PS/2 Interface Development (on Card 3) 11](#_Toc43986474)

[PS/2 Interface – next problem 16](#_Toc43986475)

# Initial Assembly & Testing Steps

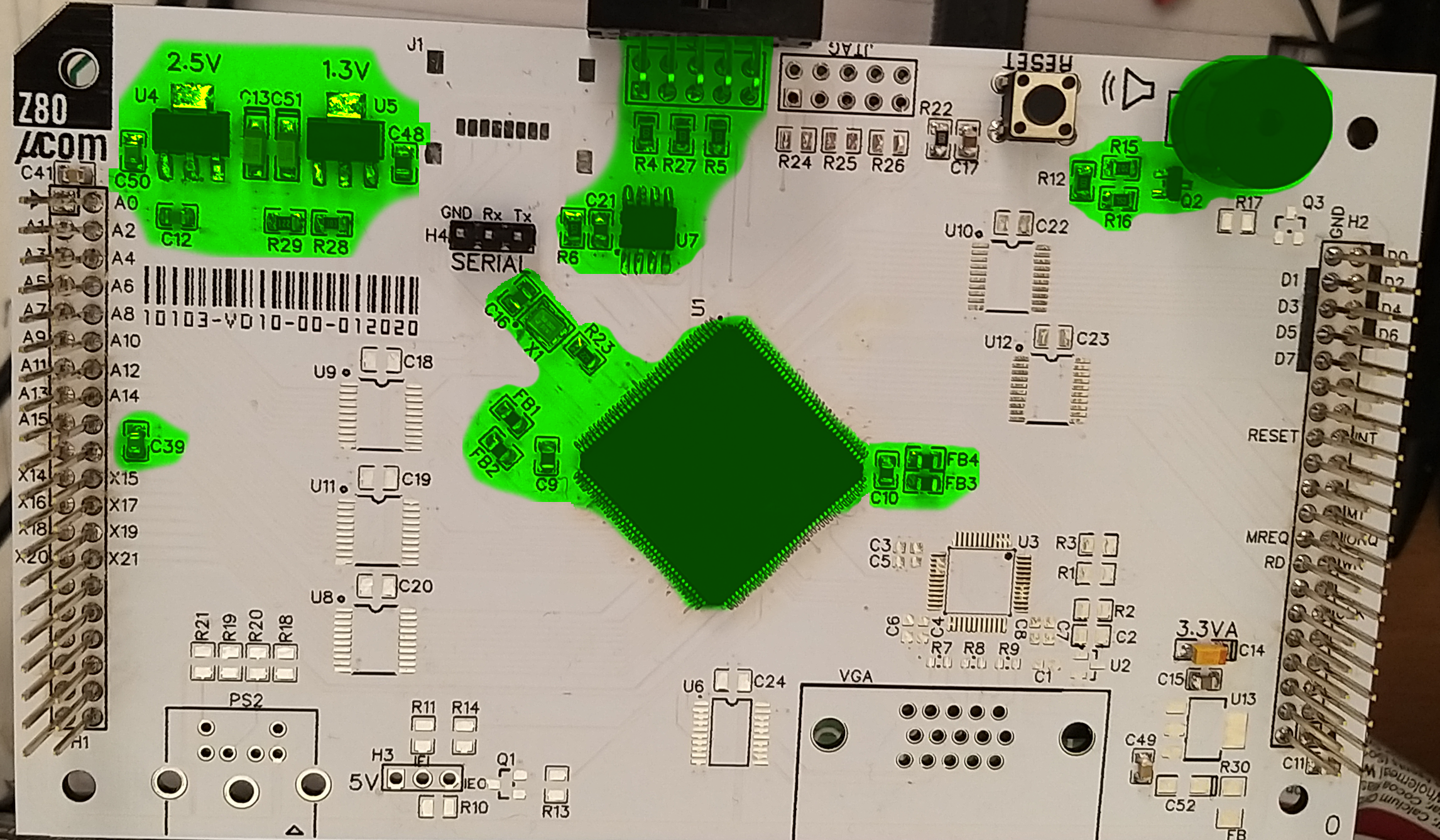


Figure - Showing essential components for initial testing (all capacitors on the underside are required)

Figure 1 above shows the key components required on the card to allow initial basic testing of the FPGA, including C39 (the 3.3V capacitor), 2.5V and 1.3V regulators and supporting components, critical FPGA chip and supporting components (clock source and power smoothing, including all capacitors on the underside), including the EPCS16 (U7) and AS programming connector and pullups. **NOT SHOWN ARE THE DECOUPLING CAPS UNDERNEATH THE FPGA – THESE ARE ALL REQUIRED AS WELL.**

The above setup allows testing of the FPGA via the AS programming port to implement basic HDL setups that facilitate audio output via the speaker. These test the EPCS16 ROM, AS port hardware and crucially the power supply and clock source (especially important as the clock is identified as a possible weak spot in the design, being difficult to solder properly after the FPGA itself).

R23 should NOT be installed for initial testing.

## Stage 1 – Testing Power Supply

Whilst connected to a uCom stack, test the top tab of the power regulators for 2.5V and 1.3V output, and 3.3V at C39 and various places on the card (decoupling caps for the 245s).

## Stage 2 – Testing Clock Output

Construct and/or use a clock divider to reduce the expected 50MHz clock signal from X1 down to a measurable clock speed – a single 7474 FF can drop the clock rate down to 12.5MHz, which is just readable by a cheap USB logic analyser (although the measured clock speed is not accurate in the slightest, it can still show a clock output vs no clock output from a faulty X1).

If clock signal is okay, fit R23.

## Stage 3 – Testing AS and U7

Connect the USB Blaster to the AS (and JTAG if fitted) port and use Quartus to read the contents of the EPCS16 ROM, erase, check for empty, etc

## Stage 4 – Test basic FPGA and audio circuit operation

Upload a simple HDL project (AUDIO\_TEST) to the FPGA. Disconnect and power-up the card on a uCOM stack. Audio output should heard from the speaker.

## Stage 5 – Test advanced FPGA operation

Fit serial header H4 and upload the full GPU HDL project to the card. Connect a TTL-USB (FT232) adaptor to the serial header (remembering to reverse Tx/Rx connections) and power up the uCOM stack using a DC jack power supply capable of supplying 9-12V at 1A or more. The USB connection alone is not able to supply enough current to the uCOM to power the GPU.

Load up the RS232\_debugger (in GPU/tools) and connect to the GPU – if all is well, you should see the memory of the GPU. Test operation of the uCOM itself to make sure it’s working normally.

## Stage 6 – Test connection to uCOM

Fit the five 74LVC245s and their decoupling caps to the board. Power-on the uCOM. Map bank $40 to Area 3 (for example) in the DMI, then use MEMX to inspect Area 3 ($C000 onwards) and compare values there with what the RS232\_debugger shows. Try changing values using POKE to test the Z80’s ability to write data to the GPU RAM and confirm changes via the RS232\_debugger.

#### For Revision 2 boards:

Additionally, test any IO-controlled peripherals, such as the LED on IO30 (usually port 0xF0).

## Stage 7 – Test video output

Assemble the VGA output section of the board and connect to a monitor. Ensure the uCOM has GPU drivers installed in the Bootstrap, DMI and CP/M. Turn on and check for video output on the monitor.

## Stage 8 – Test peripherals

Attach any other peripherals (like PS2 port) to the card and test as necessary.

# Faults & Debugging

## Stage 6 – Cannot read GPU RAM

The GPU RAM is not appearing in the memory space at all. The memory banks relating to the third chip socket are all showing $7E – or empty – values, where they should show GPU RAM contents and $FF, up to the bank\_id bytes at the top of the 16K bank.

*I’ve tested the system header pins for shorts and bad connections from the system headers to the 245 level converters, and from the converters to the FPGA, but appears there are no bad connections or shorts.*

*Have tested voltage on the DIR pin of the EA converter – reads 3.3V as expected. Lower addresses (A0-A7) appear to be working as the IO command ‘OUT’ is working on address 242 to turn sound output on and off.*

Output from Bootstrap hardware testing below, showing three-byte values with bank number and first two bytes of bank\_id address for each bank.

000200 01AA3A 022BF8 03CAAC 040000 05EAA6 06A8BE 07A883

Socket 1

RAM

08889B 09FEA3 0AEAAC 0B0ACE 0CEAA2 0DAA2A 0EA8E3 0F20AA

10BCCA 112826 12AF03 13AA8A 14A18B 153AA0 16A882 17CAE2

18A2EA 196BBA 1AABAB 1BAA82 1CCABA 1DAAAE 1EAC1A 1FA3B2

207E7E 217E7E 227E7E 237E7E 247E7E 257E7E 267E7E 277E7E

Socket 2

EMPTY

287E7E 297E7E 2A7E7E 2B7E7E 2C7E7E 2D7E7E 2E7E7E 2F7E7E

307E7E 317E7E 327E7E 337E7E 347E7E 357E7E 367E7E 377E7E

387E7E 397E7E 3A7E7E 3B7E7E 3C7E7E 3D7E7E 3E7E7E 3F7E7E

407E7E 417E7E 427E7E 437E7E 447E7E 457E7E 467E7E 477E7E

Socket 3

GPU

487E7E 497E7E 4A7E7E 4B7E7E 4C7E7E 4D7E7E 4E7E7E 4F7E7E

507E7E 517E7E 527E7E 537E7E 547E7E 557E7E 567E7E 577E7E

587E7E 597E7E 5A7E7E 5B7E7E 5C7E7E 5D7E7E 5E7E7E 5F7E7E

600100 610403 620200 630303 640700 650503 66FFFF 670000

Socket 4

ROM

680100 690403 6A0200 6B0303 6C0700 6D0503 6EFFFF 6F0000

700100 710403 720200 730303 740700 750503 76FFFF 770000

780100 790403 7A0200 7B0303 7C0700 7D0503 7EFFFF 7F0000

--- Banks below mirror those above (no expansion fitted) ---

### Tests

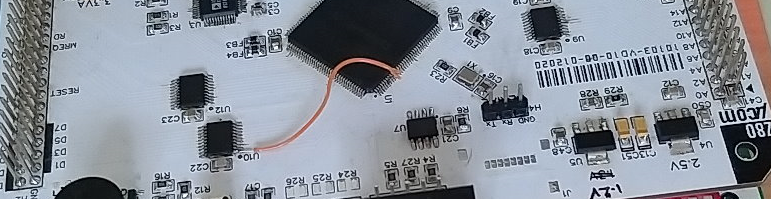
1. Write to the GPU RAM and check for changes using the RS232\_debugger to check if the GPU is reading the Z80’s address bus correctly.  
     
   *This test shows that the Z80 is able to write values to the GPU RAM. It appears that reading values is the problem and that there are no issues with the GPU reading the address bus, as the Z80 is able to write values 0xAA and 0x55 (amongst others) to the GPU RAM at various addresses.*
2. Test for timing issues with the z80\_bridge in the GPU. Set up a readable IO port in the GPU that returns a known value (e.g. 0x55 and 0xAA on a subsequent read, alternating).  
     
   *Have changed the PS2 Data IO to return 0xAA on any reads – however the Z80 is still only readying bad data (0x7E).*
3. Focus testing on U10 (data level converter). DIR pin needs to go HIGH to allow data to pass from the FPGA to the Z80 bus and is a possible point of failure for the problems experienced so far, especially if shorted to ground with a bad solder joint:
   1. Check soldering around U10 – **OK**
   2. Check continuity for OE and DIR lines on U10 to FPGA – **OK**
   3. Check no shorts on OE/DIR lines on U10 – **OK**
   4. Double-check schematic pins vs Quartus pinout – **OK**Have found a problem in the HDL design in Quartus. Z80\_bridge.sv has the direction of the data 245 reversed, as in the prototype the Z80 was on the A-side and the FPGA was on the B-side, but on the PCB, this is reversed. Have corrected this but have since had problems getting the uCOM to boot (even without the GPU on the stack).  
      Perhaps the FPGA is damaged and the 245\_DIR IO (pin 143) is unable to drive 3.3V to reverse the direction of the data 245. This would explain why the uCOM was able to operate with the FPGA’s operation of the data 245 reversed, as the data 245 was locked in a permanent A<B direction, so no data could be output by the FPGA. This needs to be verified with a logic analyser to see if the 245\_DIR line ever goes HIGH during a Z80 RD operation.
   5. Verify OE/DIR are valid IO pins on the EP4CE10 – **OK**
   6. Double-check OE/DIR aren’t reversed – **OK**
   7. Use logic analyser to verify good HIGH level on DIR & OE for data output – **FAIL**  
        
      I am unable to attach the logic analyser to the 245 or FPGA to get a stable connection and record activity on the line. Will have to use the multimeter (inaccurate) or oscilloscope (which will have its own issues with stability).  
        
      Simple assembly program to repeatedly read from address 0000 in GPU RAM (need to map bank 0x40 to Area 3 first) – run from address 0x6000:  
      2A 00 C0 7E C3 03 60

I would expect the voltage on the multimeter to increase to at least 1V or perhaps more, given the high frequency of reads being performed meaning the DATA\_DIR line should be HIGH for a significant portion of time.  
Reading 69mV on DATA\_DIR pin at 245 U10 in normal state (not reading GPU RAM).  
*Reading the same during the test program above.*

**Therefore, it appears to be a bad IO on the FPGA preventing DATA\_DIR going HIGH.**

### Solution:

Connect a free IO (nearest is pin 7) on the FPGA to the 245\_DIR pin on U10 and cut the trace to pin 143 on the FPGA. Set up the new 245\_DIR pin in Quartus and recompile the HDL to test the alternate IO. **SUCCESS – STAGE 6 PASSED**

****

## Stage 7 – No video output

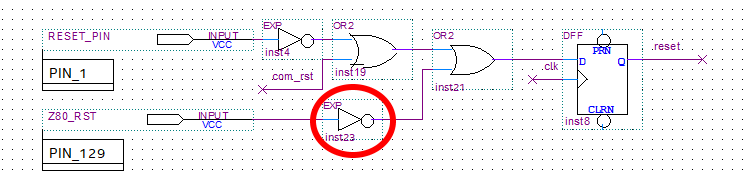
Video stage added to board. Value approximations for R1 (510R instead of 530R) and R3 (83K instead of 100K). Soldering U3 was awkward – issues with solder not getting hot enough (on discrete components in places too) I think due to lack of flux in solder. No video output at all during testing under Bootstrap with GPU driver. In fact, even without any GPU drivers there should still be the default FPGA output, yet there is nothing.

### Investigation:

1. Check HDL in Quartus for errors/incorrect pin allocations.  
   No blank output assigned in HDL or Quartus pin manager. Output added from vid\_de\_out, ANDed with an IO port output so BLANK should default to HIGH, but is switchable from the uCOM via IO port 243 (0 – off, ~0 – on). *Still no video output.*
2. Check *vde* line from HDL for correct level and check HDL for correct output **– OK**
3. Check for good 3.3V from U13 **– OK (3.24V)**
4. Check solder connections for dry joints, shorts or pins not soldered at all on U3 **– OK**
5. Check U6 for good 5V supply **– OK (4.7V)**
6. Check *pixel\_clk* output from FPGA **– OK**
7. Check VSYNC and HSYNC output from FPGA with logic analyser. – FAIL.No hs or vs signals detected from FPGA. Preliminary testing on a couple of BLUE colour channel pins couldn’t pick anything up, either.

### Solution:

The issue wasn’t a hardware (PCB or component) problem – it was the HDL used in the FPGA. I’d made a silly mistake when I originally added the Z80 RESET line to the z80\_bridge and connected it up to the existing RESET line via an OR-gate. I’d forgotten to include an inverter – the Z80’s RESET line is active LOW, but the RESET line on the video card is active HIGH, so effectively the video circuit was held in reset all the time the Z80 wasn’t, and no image was getting output.

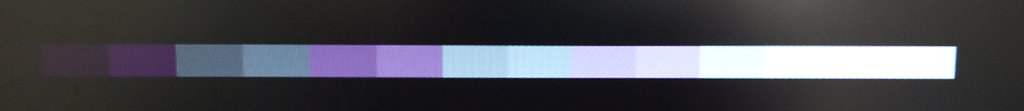


After adding the inverter and uploading the corrected HDL to the FPGA, I got a good quality image straight away. A couple of months wasted thanks to a silly omission on my part…

Also, regarding issues soldering U3 – this is likely due to the way the GND’d pins are connected together, forming good heat-sinking which made soldering them harder. I’ve updated the PCB design (for version 3 at least) to make them easier to solder.

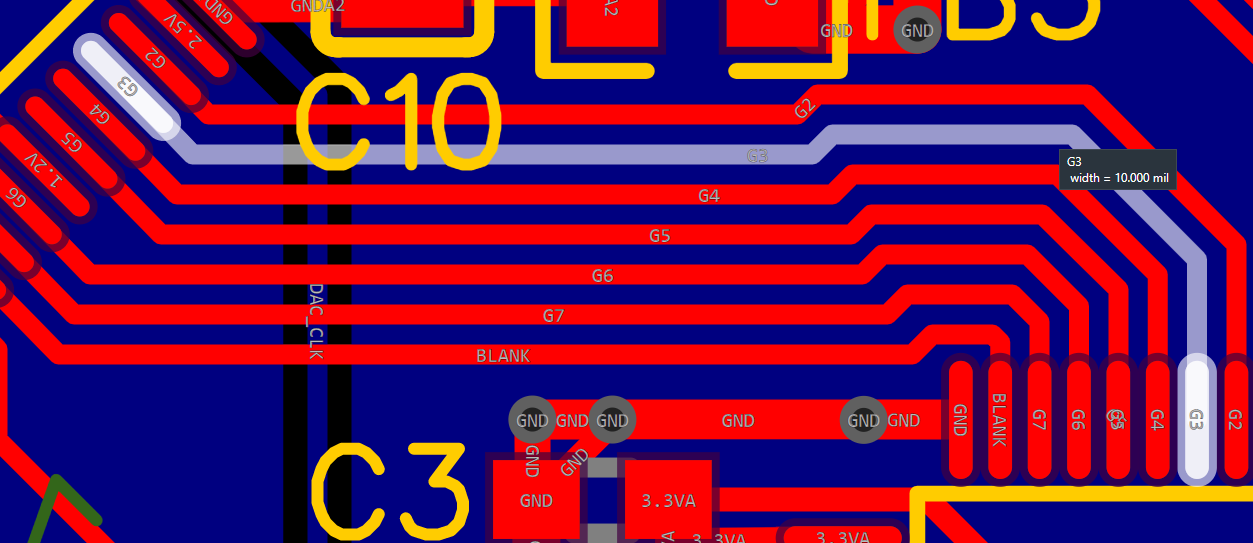
## Stage 7 – Missing green levels

Now the card is working and outputting video, I’ve tested the output by producing a greyscale image to see if all the colour channel components are working correctly. It would appear that the Green channel has a problem with bit 1:



There are two black squares off the picture to the left, equating to colours 0 and 1. As the grey scale increments up to 15 at the far right, you can see colours 2 & 3, 6 & 7, 10 & 11 are affected by a lack of green. Colours 14 & 15 are similarly affected, but it’s not noticeable in the image.

This would correlate with G3 not going HIGH when needed – which could either be a bad solder connection or another faulty IO on the FPGA. Physical inspection of the trace and connections on the PCB is required.



No fault can be found with the soldering or continuity between U1 and U3, so it would appear to be another dead IO. It’s not critical and is a fault with this board only, so I’m not going to attempt a fix.

## (Card 2) – Stage 6 – Incorrect data read/written to GPU RAM (bit 0)

uCOM can read and write to the GPU RAM, but it appears bit 0 is being dropped in both directions. The RS232\_debugger is able to write odd values, but all values written or read by the uCOM are even (bit 0 not set).

### Investigation:

1. Check solder joints on D0 line, specifically U10 pin 2 and U1 pin 143, as well as continuity checks (including between U10 pin 18 and the system header). **OK**

## PS/2 Interface Development (on Card 3)

*So, it should be an easy matter to create an IO interface in the z80\_bridge, based off the memory interface but using IORQ instead of MREQ and thus provide an IO port or two for the PS/2 interface, right?*

IO writes are easy enough – I have had these working for a while now and they are integral to testing the GPU; switching LEDs on and off and enabling the speaker etc.

However, IO reads seem to be a different problem entirely. I can’t seem to get a consistently reliable value returned from the **z80\_bridge**. A simple polling of the PS/2 data IO port (*240 / 0xF0*) is producing the following output (note: *the HDL is set up to always return ‘U’, or 0x55, no matter what*):

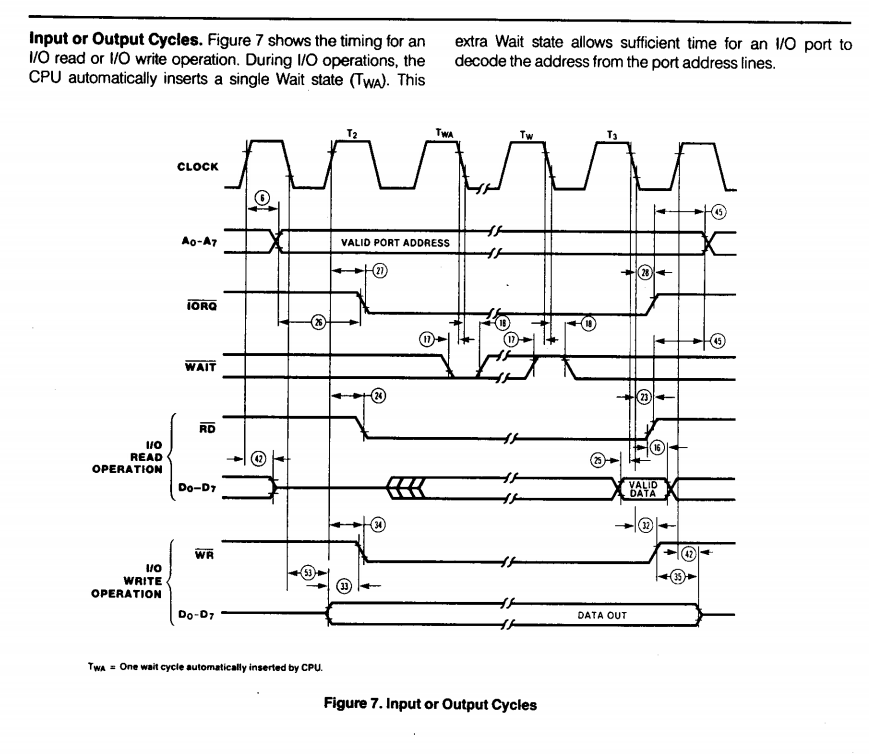
UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUxUUUUUUxUUUxUUUUUUUUUUUUUUUxUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUxUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUxUUUxUUUxUUUxUUUxUUUxUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUxUUUxUUUxUUUUUUUxUUUxUUUUUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUxUUUxUUUxUUUxUUUxUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUUUUUxUUUUUUUxUUUxUUUxUUUxUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUxUUUxUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUUUUUUUUUxUUUUUUUxUUUUUUUUUUUUUUUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUUUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUxUUUUUUUxUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUxUUUxUUUxUUUUUUUUUUUxUUUxUUUUUUUxUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUxUUUUUUUxUUUxUUUUUUUUUUUUUUUUUUUxUUUUUUUUUUxUUUxUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUUUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUxUUUxUUUxUUUxUUUxUUUUUUxUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUxUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUxUUUUUUUxUUUUUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUxUUUUUUUUUUxUUUxUUUUUUUxUUUxUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUUUUUxUUUUUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUxUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUU

Each one of those *x*’s is a problem. [Counting them](https://www.browserling.com/tools/letter-frequency) all gives:

U:11494, X: 1447 = 12.6% error rate

Looks like it could be a timing issue when the data is being put onto the Z80’s bus, but testing doesn’t seem to back that up. Note; I’m getting similar results using the polled-IO as part of the serial port reading function (shown above) and also from direct IO requests in the DMI (using the IN command). I can only run about 8 IN’s at a time using the chained-command feature though, so *x*’s pop up less frequently. The upshot is that I’m fairly sure it’s not a code problem on the Z80’s side.

Let’s have a look at the Z80’s IO cycle:



2

1

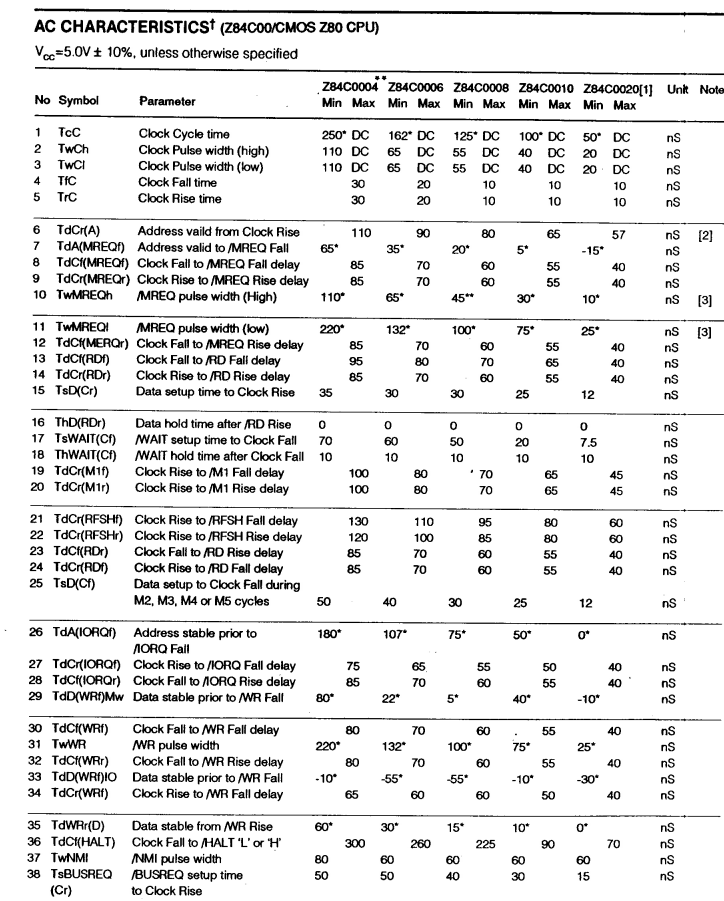
3

One Z80 clock cycle at 8 MHz is 125ns, so each vertical division in Figure 7 denotes 125ns. Running at 50 MHz, the FPGA has 20ns per clock cycle and can squeeze in at least 6 clocks for one of the Z80’s.

An IO operation flows as below:

1. IO\_DATA\_RQ goes HIGH all the time IORQ and RD are LOW and the Z80 is addressing the PS2\_CHAR port (240 / 0xF0). IO\_DATA\_ST is one-shot triggered on the rising edge of IO\_DATA\_RQ. This sets 245\_DATA’s direction to OUT (towards Z80), enables its output and sets the FPGA’s bidir data pins to output. It also triggers IO\_245\_DLY.
2. IO\_DATA\_RL is one-shot triggered when IO\_DATA\_RQ is HIGH and IO\_245\_DLY[4] is HIGH. This is 80ns (4 x 20ns) after IO\_DATA\_ST triggers. This releases data onto the bus.
3. IO\_DATA\_EX one-shot triggers on the falling edge of IO\_DATA\_RQ, which goes LOW with whichever of IORQ or RD goes HIGH first, signalling the end of the IO cycle. The 245\_DATA direction is set back to IN (toward FPGA), the bidir data pins are set to input and the PS2\_CHAR register is cleared to null.

The Z80 samples the data bus somewhere around 2.5 clock cycles into the IO cycle, equating to anything from 275-325ns after IO\_DATA\_RQ goes HIGH.



So, in theory at least, I could increase the delay on IO\_DATA\_RL triggering from 4 clock cycles (80ns) to 13 or more (260+ns) to release the data after the Z80 has inserted a WAIT state with no detrimental effects. There is no requirement to hold the data on the bus after IO\_DATA\_EX has triggered as the Z80 should have sampled the data bus before this point.

Dropping the delay to 2 clock cycles (40ns) produces the following error rate:

U: 12227, X: 714 = 6% error rate

Baseline delay of 4 clock cycles (80ns) produces the following error rate:

U:11494, X: 1447 = 13% error rate

**Raising the delay to 6 clock cycles (120ns) produces the following error rate:**

**U: 12291, X: 650 = 5% error rate**

Raising the delay to 7 clock cycles (140ns) produces the following error rate:

U: 11454, X: 1487 = 13% error rate!

Raising the delay to 8 clock cycles (160ns) produces the following error rate:

U: 12215, X: 726 = 6% error rate

Raising the delay to 10 clock cycles (200ns) produces the following error rate:

U: 11602, X: 1339 = 11% error rate

Raising the delay to 13 clock cycles (260ns) produces the following error rate:

U: 11532, X: 1409 = 12% error rate

So, it seems from the rough statistical analysis above that a delay of 6 clock cycles (120ns) before releasing the data on to the Z80’s data bus provides the minimum error rate, but not zero. I’m going to have to delve a little deeper into this issue; time to hit the simulations.

Initial simulation results:

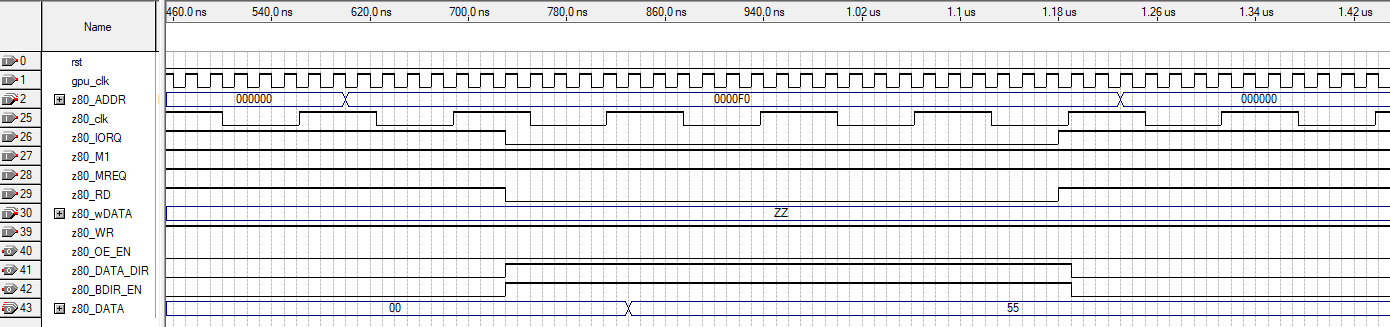
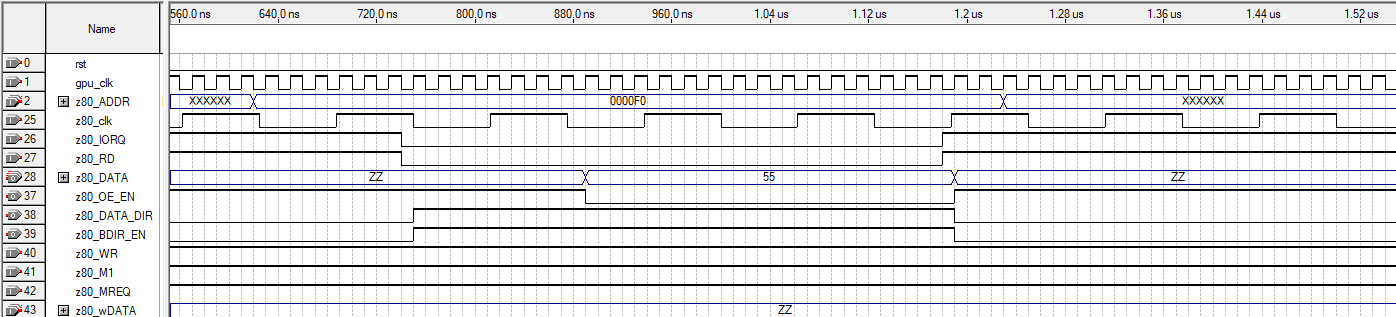


Figure - Initial simulation setup showing IO RD cycle

Figure 2 above is an *initial* setup for the simulation. I’m making that point because I’m learning how to use simulations – it’s not particularly straightforward so I’m doubting their accuracy and use. They’re also generated in Quartus II v9.1, because of licensing issues with the simulation software in Quartus Prime, so the z80\_bridge.sv module is being tested in isolation.

Now I can see it in a visual form, I’m not overly happy with the DATA 245 being enabled all the time. So, I’ve done some tweaking and made it so that the DATA 245 is only enabled when it actually *has to be*. The outcome of these tweaks are below:



Key points of difference:

* Z80\_DATA bidirectional bus is high-impedance when not in use
* Z80\_OE\_EN is OFF when the data bus is not in use (in either direction)

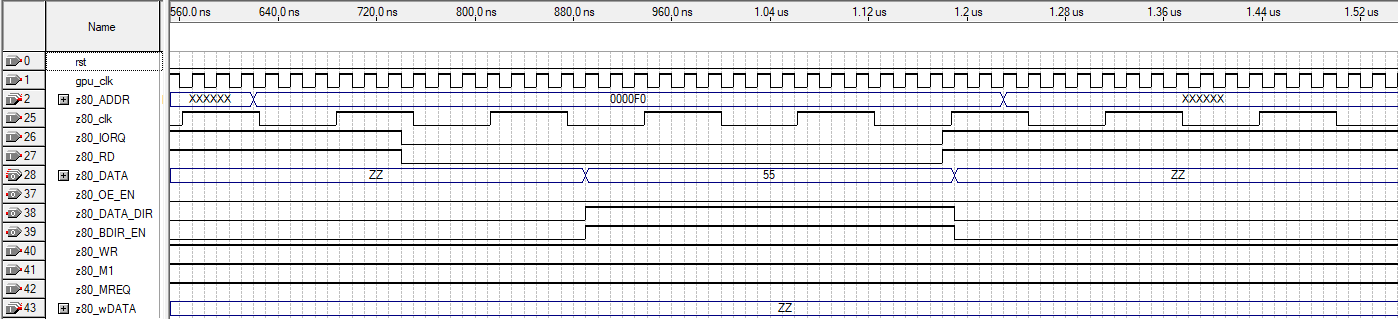
And the results of this tweak? A drop in error rate, but now I’m getting square brackets as well as x’s:

U: 12743, ]: 115, x: 83 = 1.6% error rate

But memory access to the GPU RAM is broken, so I’m going to have to review the GPU RAM RD/WR routines in z80\_bridge. For the moment however, rolling back the change and having the DATA 245 enabled all the time seems to have fixed the RAM RD/WR issue, with the following change to the statistics:

U: 12134, x: 727 = 6% error rate

It appears that the DATA 245 being permanently enabled is having an appreciable effect on the error rate, so I need to get to grips with why having the DATA 245 momentarily enabled is causing a problem with memory ops. The next change tightens the direction control on the bidirectional data pins from the FPGA:



DATA\_DIR and BDIR\_EN are only active whilst data is ready to be put on the data bus, instead of before where they were active for a while before this (a short time after IORQ and RD dropped).

\**drumroll*\* Aaaaaand the results are in:

**U: 12861 – 0% error rate!**

## PS/2 Interface – next problem

I can now get the keyboard to print stuff to the DMI. Problem is, it’s mostly gobbledygook:

qAAAqAAAA@w@AA@AwAAAAAeAAAAeAAAAAr@@@@@r@@@@@t@@@@@@t@@@@@ssAAAAsAAAAA

Above is the result of typing, “*QWERTS”*. So, it seems the old problems we were having with the Verilog PS/2 modules have now resurfaced. Some more investigation into debouncing and perhaps even trying to find a better PS/2 interface is required. I’m not sure why capital A’s and @ signs have made it into the text either.

Using an alternative PS/2 interface module ([provided by Digikey](https://www.digikey.com/eewiki/pages/viewpage.action?pageId=28279002)) gives this with default settings:

qqqq wwwwww999eeeeeRRRRTTTTSSSS

The fix was a simple one – the clock source to the PS2 module was actually running at 125 MHz instead of 50 MHz, so correcting that resolved nearly all the problems. There’s still some occasional repetition of keys, but increasing the debounce counter from 8 to 9 bits (effectively doubling the debounce time to 10µs), stops this and provides reliable keyboard input to the µCOM.