# To Do list:

## Prototype-board:

* Insert new JTAG-footprint, based on Lattice programming-cable
* Select support-components for all the main components (resistors, caps, et c)
* Place all components in the optimum manor
* Calculate impedance-matching of high-speed signals (like between GPU and DRAM)
* Trace & Route

## GPL-GPU core:

* Port VHDL and Verilog code to System Verilog
* Alter code to work for Lattice FPGA

# Manufacturers:

## JLCPCB:

For the moment, the minimum clearance for multi-layer PCB when using BGA is 0.127mm spacing pad to track and 0.127mm pad to pad.

3.5 mil for multiple layers

## TRING PCB:

https://pcb.ba/Home/PCB

Located in Bosnia / Herzegovina

# Specifications:

## Specification-target:

**Bus**: PCI 2.3

**Min Memory Size**: 8MB

**Memory Type**: DDR2

**Card Type**: S-VGA

**Ports**: VGA, DVI-I, USB-C (HDMI 2.1, DP 1.4b), JTAG (pcb edge con.)

**Video Acceleration**: MPEG-1 (VCD), MPEG-2 (DVD)

**Core**: 64bit

**Effective Memory Clock**: 65 (MHz)

**Memory Bus Width**: 64bit

**Memory Bandwidth**: 250 (MB/s)

**Max Memory Clock**: 72 (MHz)

**Min Memory Clock** : 50 (MHz):

**Ramdac**: 135 (MHz)

(1.88 times faster Ramdac than actual Ram.)

The best old CRT’s can achieve speeds of up to 177 Mhz pixel-rate – the DAC needs to be able to serve analogue signals that fast.

## Speed requirements calculations:

### Memory-bandwidth

You multiply the effective frequency by the bus width to get bandwidth throughput.

#### Example:

“gtx 970 has a 256-bit memory bus and GDDR5 memory running at 1753Mhz

The memory is quad pumped so its effective speed is 7012Mhz (1753\*4).   
7012 \* 256 = 1795072 Mb/S or 224GB/s”

#### My application:

Winbond-1: 32-bit, 128 MB (1Gb), DDR2, @533 Mhz

Winbond-2: 32-bit, 512 MB (4Gb), LPDDR3, @ 933 MHz

(W63CH2MBVACE)

ISSI-1: 16-bit, 256 MB (2Gb), DDR3, @800 Mhz

Micron-2: 16-bit, 128 (1Gb), DDR3, @933 MHz

https://www.digikey.se/sv/datasheets/microntechnologyinc/micron-technology-inc-1gb\_1\_35v\_ddr3l

Frequency shown = single rate (2 Mhz example)

DDR = Double Data Rate (2 \* 2 = 4 Mhz example)

Winbond-1: 32 \* 1066 = 34112 Mb/s or 4.264 GB/s

**Winbond-2: 32 \* (933 \* 2) = 59712 Mb/s or 7.464 GB/s**

### PCI-bus bandwidth:

PCI bus bandwidths can be calculated with the following formula:

frequency \* bit width = bandwidth

#### Example:

33.33 MHz \* 32 bits => (1067 Mbit/s)/8 = 133.32 MB/s

*(divide by 8 for final MB/s)*

#### My application:

66 Mhz \* 32 bits = (2112 Mbit/s)/8

PCI 32-bit, 66 Mhz: **264 MB/s**

#### Signal-translation:

Bandwidth target for a 32-bit bus: 8.25 MB/s (per data-line)

Bandwidth formula:

Frequency \* bit width

My translation-chip: 8.8 ns / 113 Mhz

100 \* 4 = 400 Mbit/s = 50 MB/s

113 \* 16 = 1808 Mbit/s = **226 MB/s**

(6\*5 = 30 MB/s)

### DAC-speed:

The DAC needs to be running at the per-pixel frequency, which at 177Hz VGA resolution would be in the vicinity of 100MHz.

There are "video DACs" designed for this kind of use case, though other general high-speed ones would also work.

**Formula:**

*Pixel Clock = Output Frequency \* Horizontal lines \* vertical lines*

(plus margins for refresh intervals.)

177 \* 640 \* 480 = 54 374 400 (54 Mhz)

177 \* 1920 \* 1080 = 367 Mhz

(probably around 400 Mhz for refresh intervals – the maximum speed of the FPGA)

75 \* 1920 \* 1080 = 155 Mhz

120 \* 1920 \* 1080 = 249 Mhz

144 \* 1920 \* 1080 = 298 Mhz

**Alternate formula (from Analog Devices):**

*Pixel Clock Rate/Dot Rate = (Horiz Res) × (Vert Res) × (Refresh*

*Rate)/(Retrace Factor)*

Dot Rate = 1920 \* 1080 \* 60 /0.8 = 155 Mhz

**Online Video Calculator:**

http://www.csgnetwork.com/videosignalcalc.html

According to above, for **1080p @120 Hz**, the required clock frequency is **374 Mhz**

#### Built-in block in the FPGA:

JESD204A/B – ADC and DAC converter interface: 312.5 Mb/s to 3.125 Gb/s (ECP5)

# Components:

## Actives:

### FPGA

ECP5-85

### VBIOS-chip

Needed for Dual-boot: 32 Mb – rec. 64 Mb.

My chip: 128 Mb

Triple-boot possible! :D

Supports Quad-SPI (qSpi) (good, since this is supported by ECP5)

QPI -mode as well. (quad-everything, on all 4 lines)

### DRAM

*5.3. GDDRX2\_RX.ECLK.Centered* (🡨 Lattice Doc-reference)

Guidelines for connecting:

### Clocks:

“A dedicated PCLK clock pin must always be used to route an external

clock source to FPGA logic and I/O.”

F19

## Passive:

Resistors

Capacitors

Inductors

## Connectors:

JTAG

sysConfig – the spec is from Lattice – they utilize a 9-13-pin connector, the New USB Programming Cable: **HW-USBN-2B** (recommended), with 0.1 inch pitch.

(spec mentions the 3M N2510-5002-RB connector as an example)

A **4.7 kΩ** pull-down resistor is needed on the TCK signal

Supposedly there are 11 signals for configuring the FPGA.

**MSPI QUAD**

### Available signals on ECP5-85:

#### JTAG:

1. TDO
2. TDI
3. TMS
4. TCK
5. VCCIO8
6. GND

#### sysCONFIG:

(for Quad SPI)

1. PROGRAMN
2. INITN
3. IO2
4. IO3
5. CFG1 (pulled high, other cfg-pins are grounded)
6. MISO
7. MOSI
8. CSSPIN
9. DOUT
10. VCCIO8
11. GND

For full QMSPI, 6 pull-up resistors are needed.

# Features:

## Resolutions:

320x200 (16 colors)  
  
640x350 (16 colors)  
  
640x480 (2 colors)  
  
640x480 (16 colors)  
  
320x200 (256 colors)

320 x 200

320 x 240

320 x 400

320 x 480

360 x 200

360 x 240

360 x 400

360 x 480

640 x 400

640 x 480

648 X 480

800 x 600

960×720

1024 x 768

1280 x 720

1280 x 1024

1920 x 1080

## Colour modes:

2 colours (b/w)

16 colours

32 colours

256 colour-mode

16-bit colour mode

24-bit colour mode

32-bit colour mode

# Voltage/Power requirements:

Common voltages in use in board-design:

**TTL**: 2.4-5V high, 0-0.8V low

**VGA**: 0V (black), 0.7V (max brite)

(what’s the ECP5-85’s output-levels? How much can it drive?)

**FPGA**: 0.8V, 1.1V, 2.5V, (1.8, 3.3 V tolerant I/O) / 212 mA, 26 mA, 0.5 mA, 11 mA (250 mA)

(vref= 0.8V)

**DRAM**:

~~2.5V, 1.2V (I/O) / 142 mA~~

~~(AS4C256M16D4)~~

1.8V, 1.2V / 75 mA

(W63CH2MBVACE)

**VBIOS**: 1.8V / 30 mA

(IS25WP128F)

**Sig.Tran**: / 2 uA, 100 uA, 100 uA, 900 uA, 900 uA, 2uA, 20 uA, 20 uA, 50 uA (2094 uA = 2.1 mA)

(x4 = 8.4 mA)

PCI\_a: 3.3V

PCI\_b: 5V

3.3V -> 0.8V, 1.1, 1.2, 1.8, 2.5 V

5V -> 0.8V, 1.1, 1.2, 1.8, 2.5 V

Total Amp:

250 + 75 + 30 + 8,4 mA = **363,4 mA**

Total Watt: **1,8 W**

## Common voltages in use in board-design:

**TTL**:

2.4-5V high, 0-0.8V low

**VGA**:

0V (black), 0.7V (max brite)

*(what’s the ECP5-85’s output-levels? How much can it drive?)*

ICCHTX-OP VCCHTX, Output Buffer Current (Per Channel) = 13 mA

Output levels are bound to be between 0.8V ~ 2.5V

U=R\*I => U=75 ohm \* 13 mA = 0.975V *(too high for VGA! Output-resistors needed)*

**DVI:**

LVDS Out chan.= 3.3V *(2.8V bias + 0.5V peak to peak)*

SDA & SCL = 3.3V – 5V

HPD = 2+ V

## VRM-needs:

Shielded Inductors needed (make sure they don’t have a loud hum)

----------

For 0.8V:

R1-1= 5 Ohm

R2-1 = 10K Ohm

(special case)

--------------

For 1.1V:

R1-2 =

R2-2=

For 1.2V:

R1-3 =

R2-3 =

For 2.5V:

R1-4 =

R2-4 =

### Voltages:

0.8V, 1.1V, 1.2V, 2.5V,

### Amperage:

The PCI-bus can supply the following:

+3.3V/7.6A,

+5V/5A,

+12V/500mA,

-12V/100mA.

# Analogue signals

Separate routing as much as possible, carefully connect the ground-lines, to the same ground-plane, utilizing both coils and capacitors.

May be necessary to use a **Coupled Inductor** for the analogue-digital ground meeting.

Instructions for my DAC does not include inductors, but it may be prudent to use a Ferrite Bead on VAA.

Utilize VIA-fencing.

https://electronics.stackexchange.com/questions/128637/how-should-i-connect-agnd-and-dgnd

https://www.analog.com/en/analog-dialogue/articles/staying-well-grounded.html

# Notes:

**Hi-speed** I/O is generally used for **Clocks**, **Resets** and other truly global signals.

Configuration-mode – Slave is the most flexible, aim for that.

## Potential Issues

Clock domain crossing

Synchronization of the PCI interface

Latency

Delays between main memory and the GPU – BIG bottleneck!

*(Developers need to work around this big limitation, by making sure to overlap transfers with computation, doing preemptive transfers and other techniques to mask the transfer overhead.)*

Dram – VREF must always be less than all other supply voltages!

## Dev-Diary

Replaced DDR4 as it is not compatible with ECP5.

New footprint and symbols for the new LPDDR3-mem module

Changed all schematic power-signals from labels into global symbols.

Changed schematic layout of Power-sheet.

Replaced Inductors to increase margins from zero, on SMPS-outputs.

--

Added Pull-up and pull-down resistors for JTAG and sysConfig connections.

Added pinheaders for FPGA-Configuration. (replacing edge-connector)

Combined the DVI+VGA and USB-C hierarchical sheets into one.

Assigned a dedicated sheet for the DAC.

Added 75 Ohm terminal-resistors to output of DAC.

Placed filtering-capacitors for DAC VAA.

Connected VGA ID-bits and DVI DDC together.

# References:

**Project VGA**

<http://wacco.mveas.com/index.php?entry=22>

<https://wiki.osdev.org/VGA_Hardware#Overview>

<https://www.youtube.com/watch?time_continue=102&v=EHePto95qoE&feature=emb_logo>

**GPL-GPU**

<https://github.com/asicguy/gplgpu>

## Video on the PC

**VGA Voltage Levels**

http://microvga.com/faq/electrical/what-are-vga-voltage-levels

**Video Display Signals and the MAX9406 DP-HDMI/DVI Level Shifter—Part II**

https://www.maximintegrated.com/en/design/technical-documents/app-notes/4/4313.html

*“The signal levels for LVDS channels are 0.5VP-P with a DC bias of 2.8V on both leads of each differential pair. The signal levels for the SDA and SCL pins allow levels up to 5V. The logic-high level on the hot-plug pin is greater than 2V.”*

**DVI waveforms**

https://www.tek.com/datasheet/dvi-compliance-test-solution-0

***Differential Signalling***

https://www.microcontrollertips.com/what-is-differential-signaling-faq/

## VIA Fencing

https://en.wikipedia.org/wiki/Via\_fence

*“Via fences too close to the line being guarded can degrade the isolation otherwise achievable. In stripline, a* ***rule*** *of thumb is to* ***place the fences at least four times*** *the trace to ground-plane distance* ***away*** *from* ***the line*** *being guarded.”*

**PCI-bus interface card**

<http://markmuzzin.blogspot.com/2011/10/pci-bus-interface-card.html>

<https://pinouts.ru/Slots/PCI_pinout.shtml>

http://www.interfacebus.com/Design\_PCI\_Pinout.html

https://allpinouts.org/pinouts/connectors/buses/pci/

**Video Timings Calculator**

<https://tomverbeure.github.io/video_timings_calculator>

<https://en.wikipedia.org/wiki/Coordinated_Video_Timings>

**Frequency-choice for Voltage-switch**

<https://www.allaboutcircuits.com/technical-articles/how-to-choose-the-frequency-of-your-switching-regulator/>

**JLCPCB fpga board**

https://www.eevblog.com/forum/fpga/custom-spartan-7-board-for-beginners/

**Hot Plug Detection (DDC & AUX-dp)**

https://www.datapro.net/techinfo/hot\_plug\_detection.html

## Schematic Layout rules

https://www.youtube.com/watch?v=R\_Ud-FxUw0g

Ground always down

Never have lines crossing your symbols

All signals should be drawn going from left -> right

Don’t show footprint-names on IC-package symbols

## Symbiflow – Open Source FPGA – Trellis (ECP5)

https://symbiflow.readthedocs.io/en/latest/prjtrellis/docs/

https://github.com/SymbiFlow

<https://symbiflow.readthedocs.io/en/latest/symbiflow-arch-defs/docs/source/getting-started.html>

### IRC

<https://webchat.freenode.net/#symbiflow>

### Suggested contribution:

[SV-Tests](https://github.com/SymbiFlow/sv-tests)

YOSYS

Syntax 8.3

Jump statements 12.8

Array locator methods 7.12.1

FX68K m68k core fx68k

<https://antmicro.com/blog/2019/11/systemverilog-test-suite/>

## JLCPCB design rules

<https://jlcpcb.com/capabilities/Capabilities>

**GitHub**

https://github.com/Doomn00b/Open-SVGA/blob/master/README.md

## Feedback (for improvements)

<https://www.twitch.tv/videos/730153613>

<https://www.twitch.tv/videos/754641831>

## Lattice FPGA’s:

<https://www.digikey.se/en/articles/fundamentals-of-fpgas-part-2-getting-started-with-lattice-semiconductor-fpgas>

**Summarizing the Lattice ECP5**

https://mindchasers.com/dev/pi-lattice-ecp5

### Configuring / Programming FPGA’s

<https://allaboutfpga.com/fpga-configuration-tutorial/>