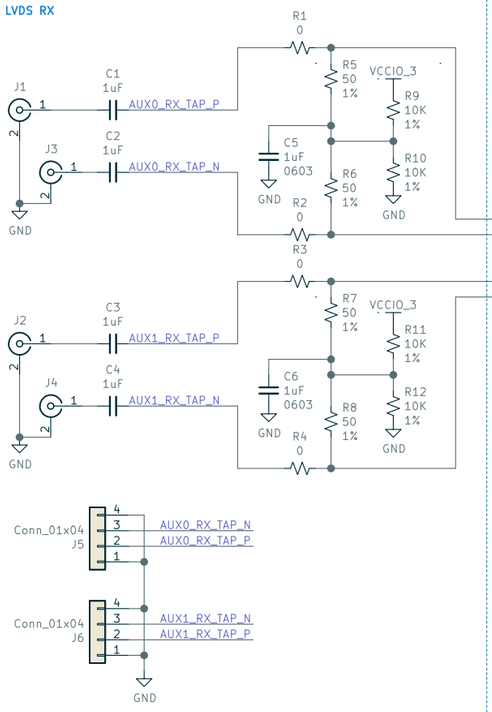
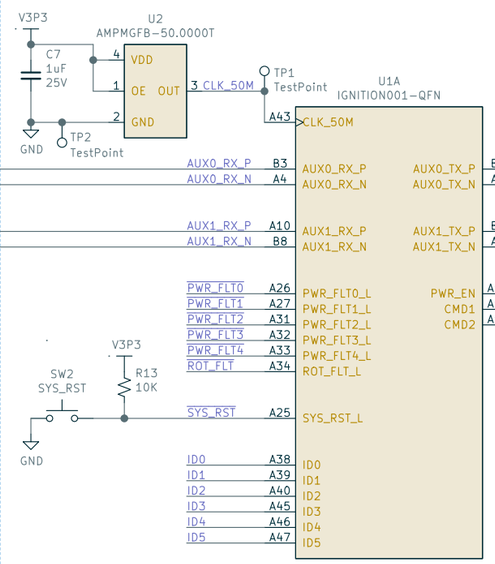
Ignition Target

LVDS RX



The LVDS transmitter pair is implemented as per Lattice FPGA-TN-1253 using PIO pin pairs in Bank 3. The iCE40 inputs expect a proper LVDS signal so the inputs are biased after the AC-coupling cap. Resistor and capacitor values can be tuned where required.

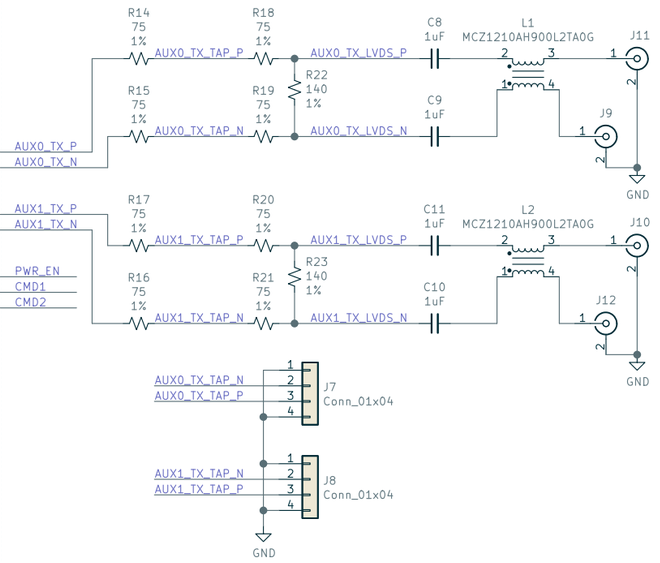
An optional header footprint/via is placed near the SMA connector to allow for alternate prototyping, using either single-ended or differential signaling. The zero ohm resistor footprint can be used for termination or slew limiting resistors if desired.



TODO:

- Check TX/RX cap values againt bit rate

LVDS TX



The LVDS transmitter pair is implemented as per Lattice FPGA-TN-1253 using PIO pin pairs in Bank 3. The resistor values above were derived using the equations on p. 4 and the following assumptions:

Z0 = 50 ohm
VCCIO = 2.5V
V_OD = 0.35V
R_OUTPUT = 30 ohm

R_P = 2 * ((Z0 * VCCIO) / (VCCIO - (2 * V_OD)))
= 2 * (165 / 1.8)
= 139 ohm

R_S = ((Z0 * R_P / 2) / ((R_P / 2) - Z0) - R_OUTPUT
= (3472 / 19) - 30
= 149 ohm

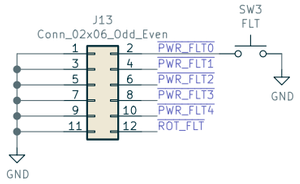
The series resistor is broken into two pieces of 75 ohm each. The intend here is that one pin of a 100 mil header/ footprint is inserted between the two resistors. If done using a tight layout this via should add minimal disruption at the edge rates of these transmitters.

Inserting the via would allow for IO pin to be reused for alternative prototyping by not fitting the second series resistor, parallel resistor and SMA connector, while using the first resistor footprint as slew limiting resistor or for series termination.

One possible application of this alternative scheme is to allow the Ignition protocol to be carried using single ended LVCMOS signaling at 3.3V between this broad and an ECP5 dev board without requiring SMA connectors for the link partner. This would simplify initial prototyping work.

The AC couping capacitors near the SMA connector are optional in case one wants to experiment. The layout should allow for some copper nearby connected to ground so experiments with a choke are possible. If not in use 0 ohm resistors should be fitted.

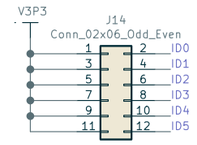
Fault Pin Straps



Fault pin straps allow setting fixed fault values. The switch attached to PWR_FLT0 provides a convenient way to cause fault interrupts during integration tests. The header allows reuse of these pins for alternative prototyping.

The FPGA should use programmable pull-up/pull-down on these poins if required.

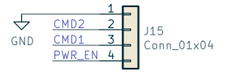
ID Straps



ID straps allow setting the desired value. The header allows reuse of these pins for alternative prototyping.

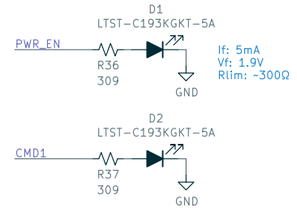
See RFD 142 for currently allocated ID values. The FPGA should use programmable pull-up/pull-down on these pins if required.

CMD Bits Header

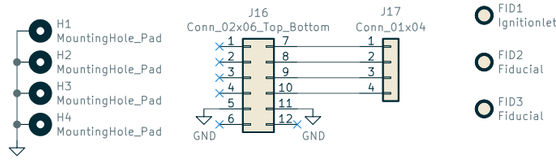


Command bits are exposed on a header for easy probing and/or alternative prototyping.

CMDx LEDs



Mechanical



The board assumes the width of two standard 0.80" PMOD boards. J16 is intended for additional mechanical stability, but is connected to J17 as a Type 1 interface, providing GPIOs for prototyping which for example can be wired to CDONE or FLASH_RESET when connected to an appropriate host board.

Sheet: Power

Power

File: ignitionlet-power.sch

Sheet: Config

Programming & Configuration

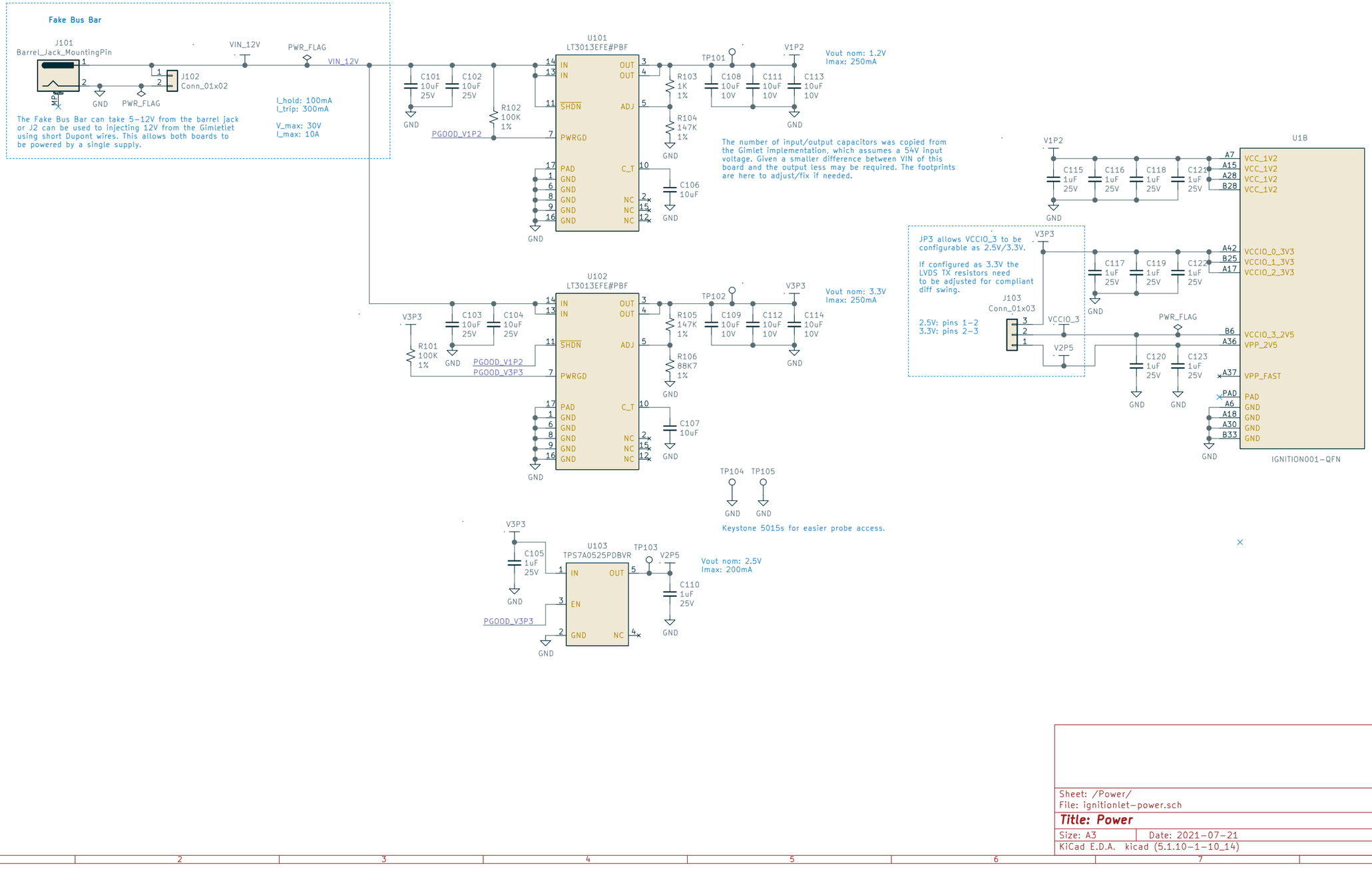
File: ignitionlet-config.sch

Oxide Computer Co.

Sheet: /
File: ignitionlet.sch

Title: Ignition Target

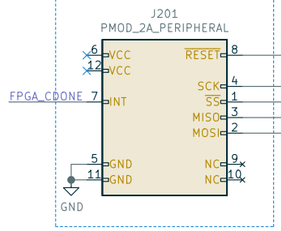
Size: A3	Date: 2021-07-21	**Rev: 2**
KiCad E.D.A. kicad (5.1.10-1-10_14)		Id: 1/3



Sheet: /Power/		
File: ignitionlet-power.sch		
Title: Power		
Size: A3	Date: 2021-07-21	**Rev: 2**
KiCad E.D.A. kicad (5.1.10-1-10_14)		Id: 2/3

PMOD 2A Peripheral (optional)

Intended to allow a Gimletlet to program either SPI flash or the FPGA.

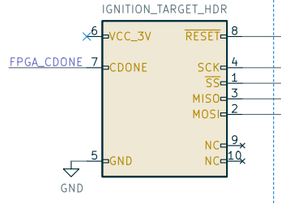


Ignition Program/Debug Header

Intended to mate with Adafruit FT232H + adapter to allow software compatibility with both Lattice and open source programming tools.

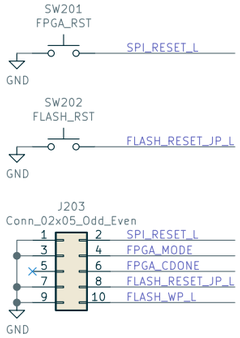
The pinout is compatible with a straight PMOD 2A connector to allow for alternative adapters if desired.

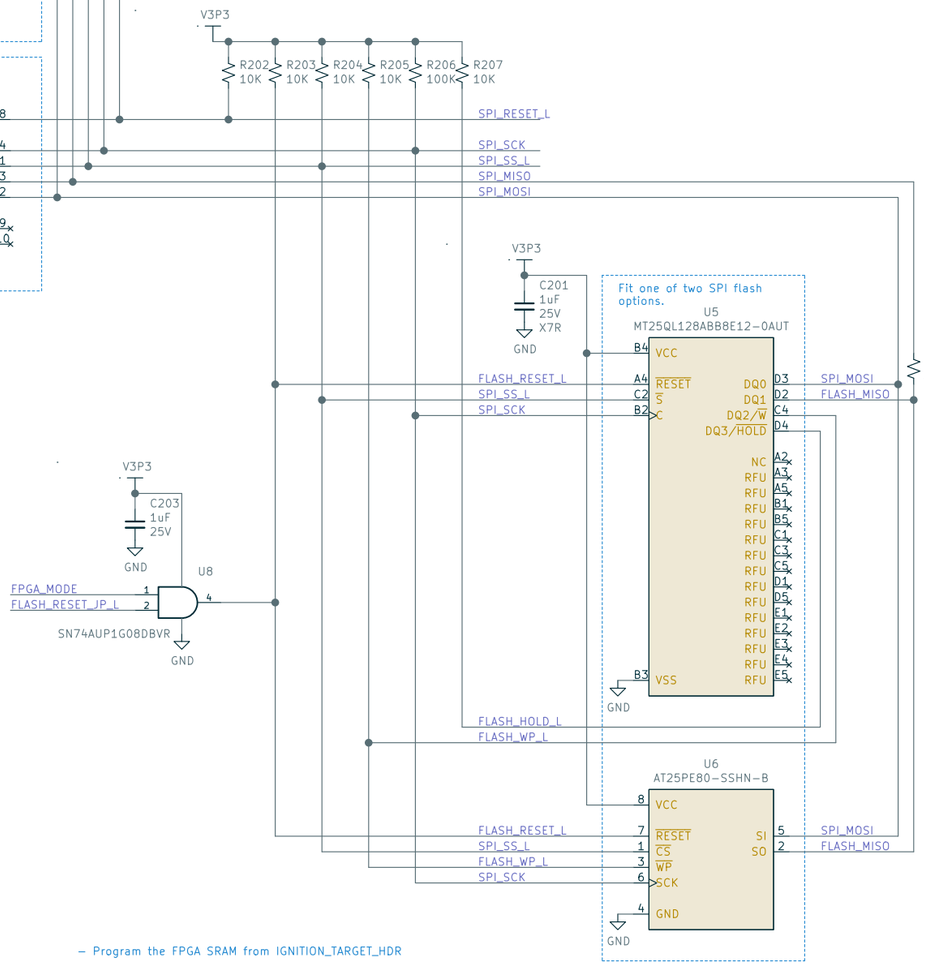
Whatever connects to this header is expected to bring slew limiting resistors.

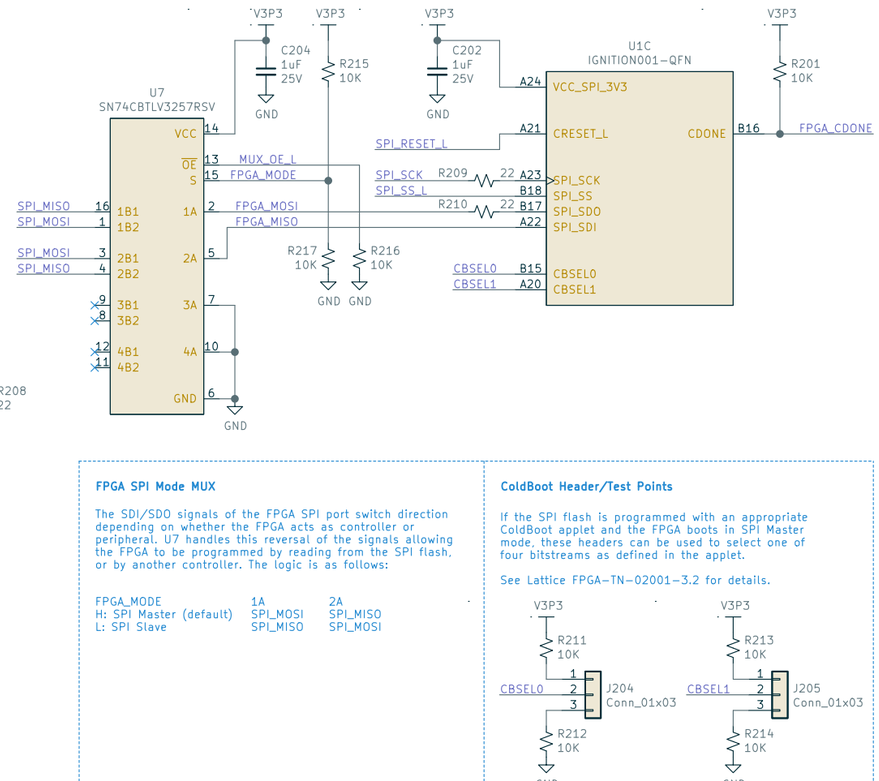


Header/Jumpers/Test Points

Intended for hacking and/or forced reset/write protect.







FPGA SPI Mode MUX

The SDI/SDO signals of the FPGA SPI port switch direction depending on whether the FPGA acts as controller or peripheral. U7 handles this reversal of the signals allowing the FPGA to be programmed by reading from the SPI flash, or by another controller. The logic is as follows:

FPGA_MODE	1A	2A
H: SPI Master (default)	SPI_MOSI	SPI_MISO
L: SPI Slave	SPI_MISO	SPI_MOSI

ColdBoot Header/Test Points

If the SPI flash is programmed with an appropriate ColdBoot applet and the FPGA boots in SPI Master mode, these headers can be used to select one of four bitstreams as defined in the applet.

See Lattice FPGA-TN-02001-3.2 for details.

Operating Modes

– FPGA as SPI Master (default)

The default operating mode for this board is with the FPGA acting as SPI master. Without anything driving SPI_SS this signal is pulled high. On init (after PoR or asserting CRESET) the FPGA will sample this pin. With the pin pulled high it will resume its init sequence as SPI master. Consequently it will then assert SPI_SS and drive SPI_SCK, allowing it to read a bitstream from SPI flash and enter the user application.

– Program the SPI flash from IGNITION_TARGET_HDR

The second mode is to program the SPI flash via the IGNITION_TARGET_HDR using an FTDI USB programmer. The programmer will assert both SPI_RESET and SPI_SS, causing the FPGA to go/stay in reset while selecting the SPI flash as peripheral. The programmer then writes to SPI flash as normal.

– Program the FPGA SRAM from IGNITION_TARGET_HDR

The FTDI programmer can program the FPGA SRAM (or NVCM) directly if the user installs a jumper on the FPGA_MODE signal. This will cause the SPI flash to remain in reset and ignore any SPI traffic while mux beteen the header and FPGA SPI port will switch the SDO/SDI lines appropriately. The programmer then asserts SPI_SS while toggling SPI_RESET. This causes the FPGA to (re-)initialize, sampling the SS pin, and initialize as SPI peripheral instead of master when it finds this signal low. The programmer then writes to the FPGA as per Lattice FPGA-TN-02001-3.2.

Holding the flash in reset can potentially be automated by connecting FLASH_RESET to an additional GPIO pin on the programming adapter and modifying the (open source) programming software to assert this pin during the programming cycle. This may already be supported in software, but is not tested.

– Program the FPGA SRAM from PMOD_2A

In the same way as stated above the PMOD header can be used to program the FPGA SRAM from an attached Gimletlet using software running on the SP. This mode assumes the SPI flash is disabled during programming, either using the described FLASH_RESET jumper, or by connecting the FLASH_RESET signal to an SP GPIO using a Dupont wire, allowing flash reset to happen under software control.

Sheet: /Config/
File: ignitionlet-config.sch

Title: Programming & Configuration

Size: A3	Date: 2021-07-21	**Rev: 2**
KiCad E.D.A. kicad (5.1.10-1-10_14)		Id: 3/3