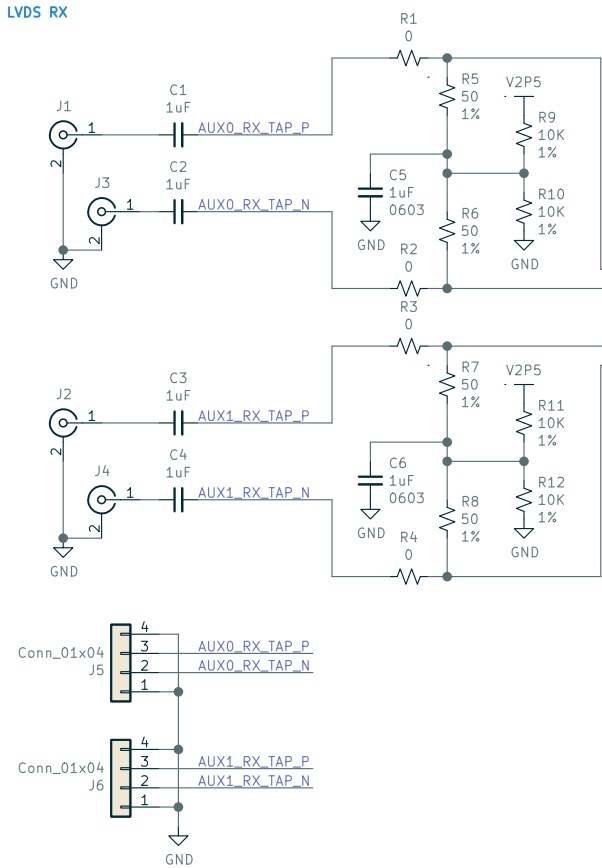


Ignition Target

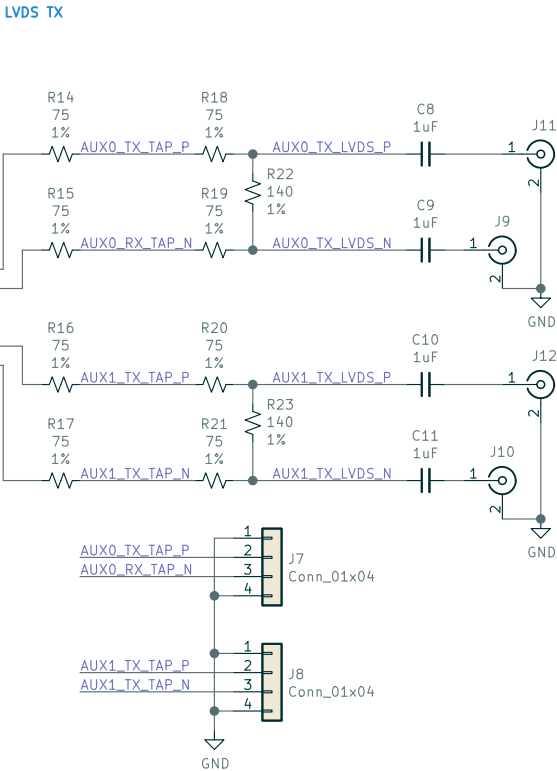
LVDS RX



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.

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:

$Z_0 = 50 \text{ ohm}$
 $V_{CCIO} = 2.5V$
 $V_{OD} = 0.35V$
 $R_{OUTPUT} = 30 \text{ ohm}$

$R_P = 2 * ((Z_0 * V_{CCIO}) / (V_{CCIO} - (2 * V_{OD})))$
 $= 2 * (165 / 1.8)$
 $= 139 \text{ ohm}$

$R_S = ((Z_0 * R_P / 2) / ((R_P / 2) - Z_0) - R_{OUTPUT})$
 $= (3472 / 19) - 30$
 $= 149 \text{ ohm}$

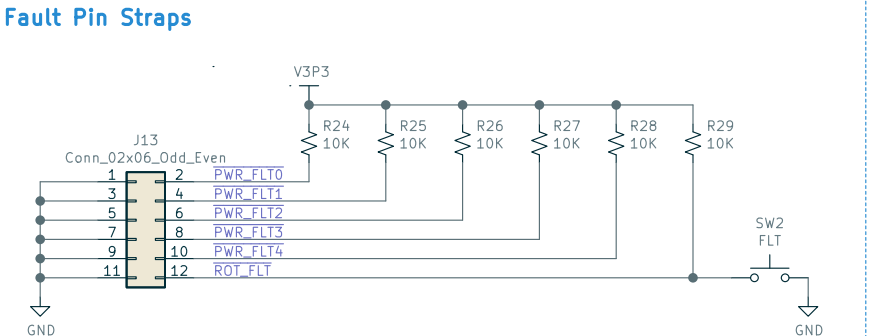
The series resistor is broken into two pieces of 75 ohm each. The intent here is that one pin of a 100 mil header/ footprint is inserted between the two resistors. If done using a tight layout this via should add minimal disruption at the edge rates of these transmitters.

Inserting the via would allow for ID 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 board and an ECP5 dev board without requiring SMA connectors for the link partner. This would simplify initial prototyping work.

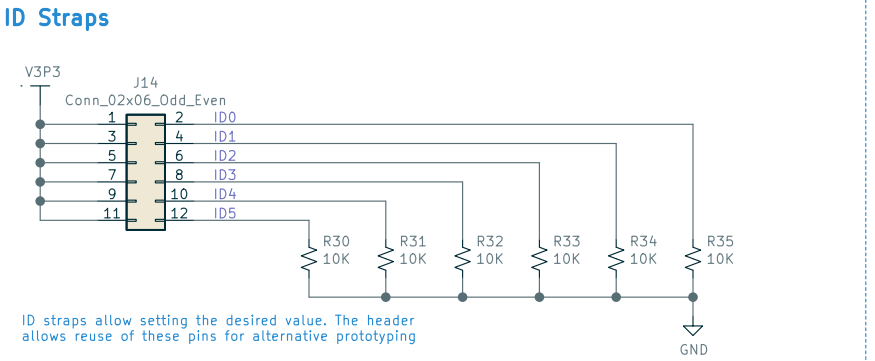
The AC coupling capacitors near the SMA connector are optional in case one wants to experiment. The layout should allow for some copper nearby connected to ground so experiments with a choke are possible. If not in use 0 ohm resistors should be fitted.

Fault Pin Straps



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

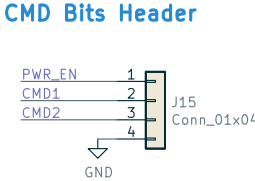
ID Straps



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

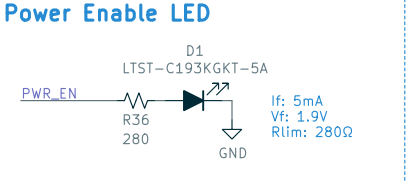
See RFD 142 for currently allocated ID values.

CMD Bits Header



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

Power Enable LED



Sheet: Power

Sheet: Config

Power

Programming & Configuration

File: ignitionlet-power.sch

File: ignitionlet-config.sch

- TODO:
- Check TX/RX cap values against bit rate
 - Add TPs, mounting holes, fiducials, logo, P/N, S/N "parts"

Sheet: /
File: Ignitionlet.sch

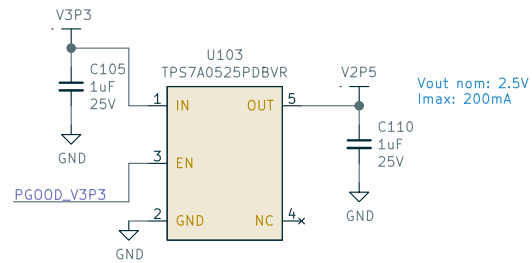
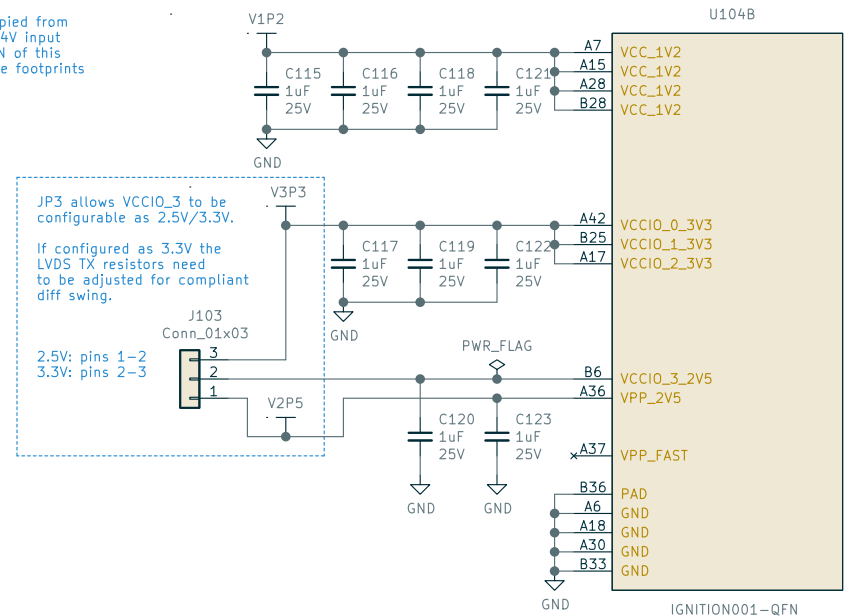
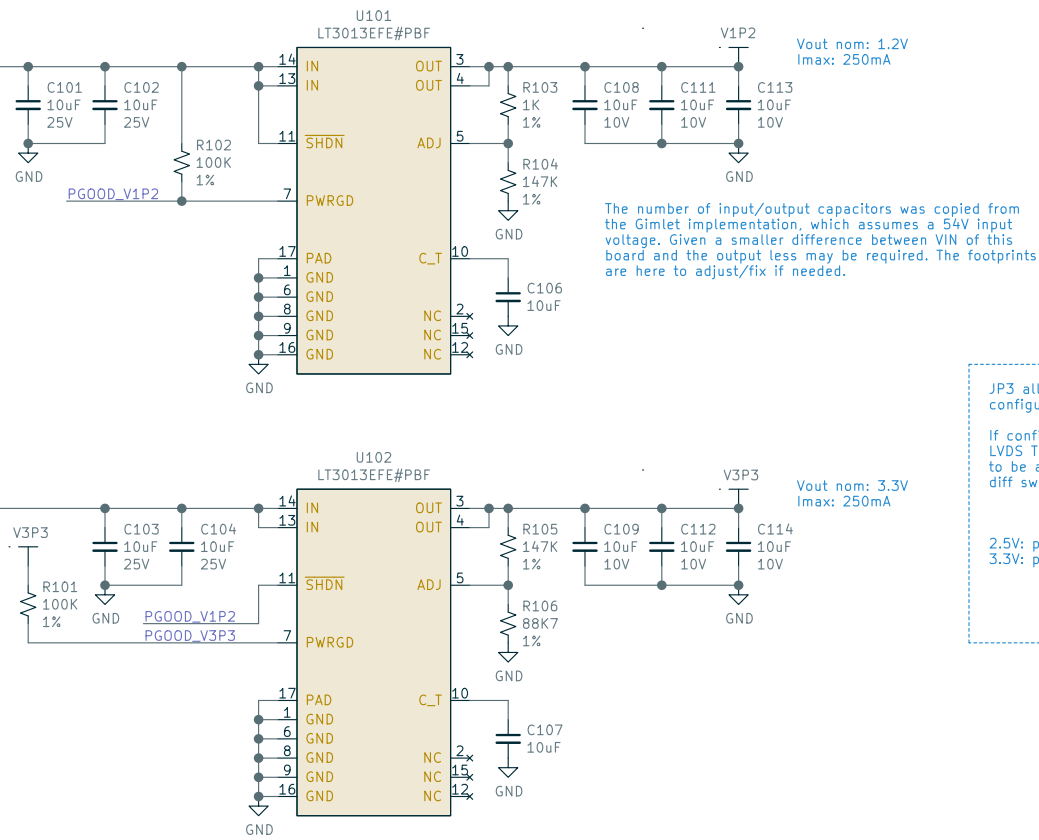
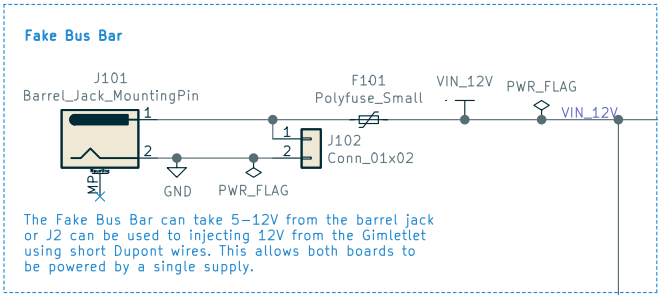
Title: Ignition Target

Size: A3
KiCad E.D.A. kicad (5.1.10-1-10_14)

Date: 2021-06-21

Rev: 1

Id: 1/3



Sheet: /Power/
File: Ignitionlet-power.sch

Title: Power

Size: A3 Date: 2021-06-21
KiCad E.D.A. kicad (5.1.10-1-10_14)

Rev: 1
Id: 2/3

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 SPL_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 SPL_SS and drive SPL_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 SPL_RESET and SPL_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.

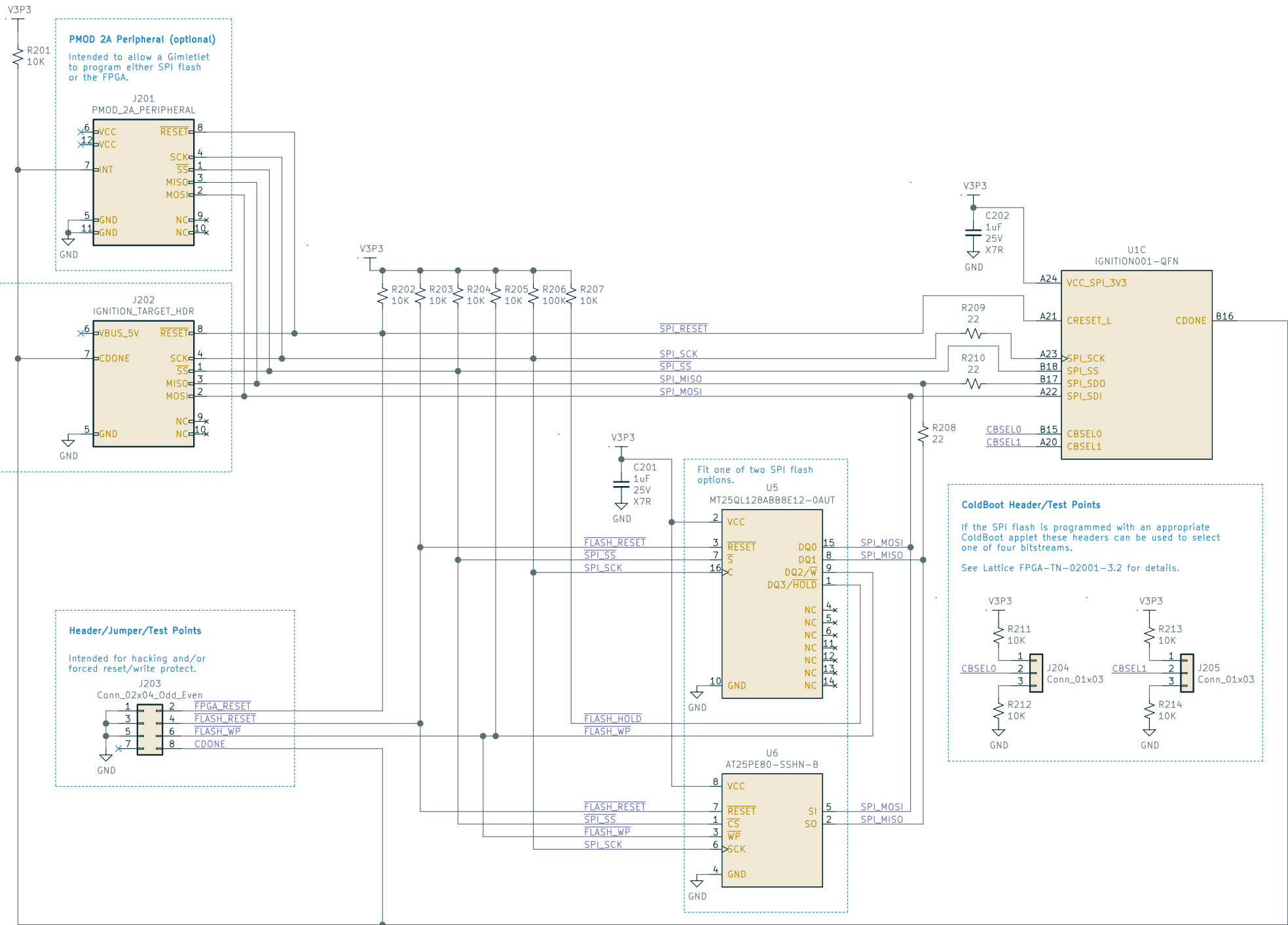
- Program the FPGA SRAM from IGNITION_TARGET_HDR

The FTDI programmer can program the FPGA SRAM (or NVCM) directly if the user installs a jumper on the FLASH_RESET signal. This will cause the SPI flash to remain in reset and ignore any SPI traffic. The programmer then asserts SPL_SS while toggling SPL_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: 2021-06-21

KiCad E.D.A. kicad (5.1.10-1-10_14)

Rev: 1

Id: 3/3