

OSURC Rover 2016 Electronics: Uniboard

Revision 1 - Nick Ames <nick@fetchmodus.org>

Thursday 7th April, 2016

The Uniboard is the heart of the OSU Robotics Club's 2016 rover electrical system. It unifies last year's electronics, combining the functions of several separate devices into one. The overall goal is to push software complexity "up the stack" to the computer, where it can be dealt with more effectively. To do this, the Uniboard incorporates an FPGA (technically, a CPLD) that performs real-time functions for the computer.

Communication between the computer and FPGA happens over a USB→Serial (RS-232) link. From the computer's perspective, the Uniboard is a memory device. Peripherals are mapped to places in this memory, like peripherals in a microcontroller. To communicate with peripherals, the computer sends commands to read or write data into given memory locations.

1 Features and Connectors	2	3 Computer Interface	5
1.1 Power Input	2		
1.2 Motor Outputs	2		
1.3 Encoder Inputs	3		
1.4 Arm Connector	3		
1.5 Signal Light	3		
1.6 Expansion Connectors	4		
1.7 Xbee	4		
1.8 RC Receiver	4		
1.9 JTAG	4		
1.10 Debug	4		
1.11 USB	4		
2 Accessory Boards	5	4 Internal Peripherals	5
2.1 Arm Breakout	5		
		5 Assembly and Programming Procedure	11
		6 Design Qualification Tests	12
		6.1 Power Supply	12
		6.2 Serial Communication	12
		7 Revision History	13
		8 Assembly Log	13
		A Bill of Materials	13
		B Schematic	13

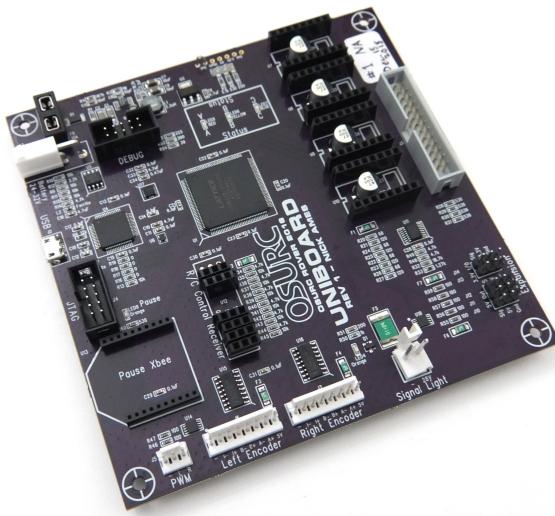


Figure 1: Uniboard without plug-in modules.

1 Features and Connectors

The Uniboard includes the following interfaces and features:

- 2x Outputs drive motors
- 2x Differential quadrature encoder inputs (with index), for left and right motors
- Switched 28V (battery voltage) output, to drive signal light
- 4x Bipolar stepper drivers, for X, Y, Z, and A (open-close) axes
- 4x Limit switch inputs, for X, Y, Z, and A (open-close) axes
- 4x Analog Inputs, for grip sensors
- 2x 3.3V GPIO pins, for expansion
- 3x 5V general-purpose outputs, for expansion
- On-board battery voltage measurement
- 3-axis accelerometer
- 3-axis gyroscope
- RC receiver socket (6 channels)
- XBee socket, for pause signal
- Battery power input
- JTAG Connector
- Debug Connector (eight signals)
- USB connection to control computer, providing a USB→Serial interface for control and JTAG programming
- 3x Status LEDs

1.1 Power Input



Connector Parts	
Board Header	Molex 441300018 (WM7513-ND)
Wire Connector	Molex 0039013028 (WM2837-ND)
Wire Pins	Molex 0457501111 (WM3279CT-ND)
Wire	18 AWG Stranded Wire

Figure 2: Power Connector

The Uniboard receives power from the battery (nominally 28V). A 10A miniature blade fuse protects against short circuits.

1.2 Motor Outputs

Three Sabertooth 2x12 motor drivers provide power to the rover's drive wheels. To control them, a signal using the Sabertooth “simple serial” protocol (at 9600 baud) is generated. The signal on the L channel is connected to the S1 input of all three saberteeth. The sabertooth DIP switches should be configured using the following settings (1-6, with 0 being off/down and 1 being on/up): 101011. This sets simple serial mode at 9600 baud.

Connector Parts		
Board Header	TE 640454-3 (A19430-ND)	
Wire Connector	TE 3-643814-3 (A31018-ND)	
Strain Relief	TE 643075-3 (A19231-ND)	
Wire	1½ twisted pairs from cat-5 cable	

Figure 3: Motor PWM Connector

Connector Parts		
Board Header	TE 640454-8 (A19435-ND)	
Wire Connector	TE 3-643814-8 (A31023-ND)	
Strain Relief	TE 643075-8 (A19236-ND)	
Wire	4 twisted pairs from cat-5 cable	

Figure 4: Motor Encoder Connectors

1.3 Encoder Inputs

To measure the rover's movement over the terrain, and to regulate its speed, a CUI AMT-113Q-V encoder is mounted on both middle wheels. (The encoder should be configured for 2048 PPR, the default out-of-the-box setting.) Quadrature signals are produced by the encoder, allowing the Uniboard to determine the direction of wheel rotation. Additionally, an index signal pulses each time the wheel makes a complete rotation. Signals are transmitted differentially to reduce the effects of noise. Power for each encoder is 5V/200mA.

1.4 Arm Connector

X Motor-A1	1	2	X Motor-B1	3	4	Y Motor-A2	5	6	Y Motor-B1	7	8
X Motor-B2	9	10	Z Motor-A2	11	12	Z Motor-B1	13	14	A Motor-A2	15	16
Y Motor-A1	1	2	Y Motor-B2	3	4	Z Motor-A1	5	6	A Motor-B2	7	8
Y Motor-B2	9	10	Z Motor-B2	11	12	A Motor-A1	13	14	Z Motor-B1	15	16
Z Motor-A1	1	2	Z Motor-B1	3	4	Z Motor-A2	5	6	Z Motor-B2	7	8
Z Motor-B2	9	10	A Motor-A1	11	12	A Motor-B1	13	14	A Motor-A2	15	16
A Motor-A1	1	2	A Motor-B2	3	4	GND	5	6	GND	7	8
A Motor-B2	9	10	5V/200mA	11	12	Limit Y	13	14	Limit Y	15	16
5V/200mA	1	2	Limit X	3	4	Limit Z	5	6	Limit A	7	8
Limit X	9	10	Limit Z	11	12	Analog 1	13	14	Analog 2	15	16
Limit Y	1	2	Analog 1	3	4	Analog 3	5	6	Analog 4	7	8
Limit A	9	10	Analog 3	11	12	Analog 4	13	14		15	16

Connector Parts		
Board Header	Assmann AWHW26G-0202-T-R (HRP26H-ND)	
Wire Connector	Assmann AWG28-26/G/300 (AE26G-5-ND)	
Wire	Assmann AWP-26-7240-T (AE11120-ND)	

Figure 5: Arm connector

To pick up samples, the rover uses a four-axis (X, Y, Z, and gripper close) cartesian arm. Each axis is moved by a bipolar stepper motor, driven by a Polulu DRV-8825 module on the Uniboard. A limit switch on the negative side of each axis provides an absolute position reference. Each limit switch input has a 190 Ohm resistance to ground to reduce noise. Switches should be wired normally closed (NC) to 5V, so that the limit switch lines receive 5V when the axis is not at the limit.

Four analog inputs (0-2.048V, 536 ohms to ground) allow the use of analog sensors (such as flex sensors) for determining when an object has been grasped.

1.5 Signal Light

An orange signal light, indicating the pause state of the robot, is mandated by the SRRCC competition rules. This port drives a signal light from the battery voltage (nominally 28V) at up to 1A. Note that the low (ground) signal is switched, not the 28V signal.



Connector Parts	
Board Header	TE 3-644615-2 (A113559-ND))
Wire Connector	TE 3-643820-2 (A31643-ND)
Strain Relief	TE 643067-2 (A19740-ND)
Wire	1 twisted pair from cat-5 cable

Figure 6: Light Connector

1.6 Expansion Connectors



Connector Parts	
Board Header	Generic 3-pin 0.1" Male Header

Figure 7: Expansion Connectors

In case new features need to be added, there are five expansion connectors on the Uniboard. Three are 5V outputs and two are 3.3 I/O. Each provides power and ground in a PWM connector pinout.

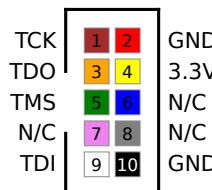
1.7 XBee

As required by the competition rules, the robot's motion can be stopped by a remote-control pause switch. TODO: Include XBee configuration info. An XBee XB24-AUI-001 transceiver receives the pause signal.

1.8 RC Receiver

During demonstrations and testing, the robot can be driven using an R/C controller. A Turnigy RX-9X8Cv2 receiver plugs into the Uniboard; channels 1-4, 7, and 8 can be used.

1.9 JTAG



Connector Parts	
Board Header	OnShore 302-S101 (ED1543-ND)

Figure 8: JTAG Connector

In case the built-in FTDI JTAG system doesn't work, a connector for an external JTAG programmer is provided. To use it, the four jumper resistors (R16-19) between the FTDI chip and JTAG lines should be removed.

1.10 Debug

To ease development of the HDL code controlling the FPGA, a debug port allows connections to nine FPGA signal lines. The 0.1" spaced pins of the connector can be easily probed with a logic analyzer.

1.11 USB

A micro USB B connector on the Uniboard connects it to the control computer. An FTDI FT2232 chip creates a USB→RS232 link for communication during normal operation. It also provides JTAG signals to program the FPGA. This configuration is compatible with Lattice Diamond, the FPGA development IDE. (On Linux systems, it may be necessary to remove the standard FTDI kernel module (with `sudo modprobe -r ftdi_sio`) to make the FTDI JTAG work.)

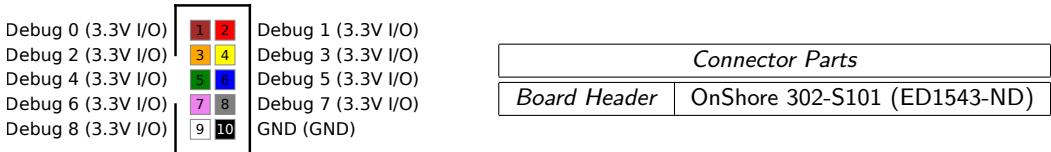


Figure 9: Debug Connector

2 Accessory Boards

2.1 Arm Breakout

To simplify the wiring between the arm connector and the Uniboard, the Arm Breakout PCB attaches to the back of the arm connector and routes the signals to a ribbon cable socket. A 26-pin ribbon cable of the appropriate length connects the Uniboard to the arm breakout.

Include photo and arm connector pinout.

3 Computer Interface

A “peripheral” is a piece of internal logic that implements a feature. Peripherals in the Uniboard are controlled through a memory-mapped interface. The control computer sends commands to read or write memory locations, telling the peripherals what to do. The format of each command is:

`<ST><peripheral><register>[data0][data1][data2][data3]<EN>`

Items in `<>` are mandatory and items in `[]` are optional in some commands. Commands are delineated with start and end bytes (0x01 and 0x17, respectively). A start byte starts a new packet and an end byte performs the memory read/write in the current packet. Verbatim 0x01, 0x17, and 0x1B values are preceded by an escape byte, 0x1B.

Each peripheral contains one or more registers. The most significant bit of the peripheral byte controls whether the command will read or write a register. If the MSB is 1, the register is read. If it is 0, it is written.

Registers can be 1-4 bytes in length. Data0, the first data byte in the command, is the least significant byte in the register. The following data bytes are more significant. If a byte in the register is not supplied by the command it will be set to 0x00.

Both read and write commands result in a reply from the Uniboard. The reply contains the registers contents. (Note: registers with read only or unused bits may not match the value written.) The peripheral and register bytes of the reply are the same as the command. While a reply is being sent, input bytes are ignored. Attempting to write a read-only register will have no effect.

4 Internal Peripherals

If not specified, the default value of writable bits is 0.

0x01: Global Control

TODO: Safety timeout?

0x00 - Global Control Register (RW, 1 Byte)

Bit	7(MSB)	6	5	4	3	2	1	0 (LSB)
Meaning					XBEE_PAUSE	FORCE_PAUSE	PAUSE	

PAUSE: Current pause state, the OR of XBEE_PAUSE and FORCE_PAUSE. 1 = paused, 0 = unpause. When paused, all robot motion is disabled.

FORCE_PAUSE: When 1, the robot is forced into a pause state, regardless of the pause switch.

XBEE_PAUSE: The current state of the pause input from the XBee. 1 = paused, 0 = unpause.

0x01 - Battery Voltage (R, 2 bytes)

The battery voltage in millivolts.

0x02 - Uptime (R, 4 bytes)

The number of seconds the Uniboard has been powered on.

0x03 - HDL Build Number (R, 4 bytes)

A unique number indicating the HDL code version. Newer versions will have higher build numbers.

0x04 - API Version (R, 2 bytes)

The API version (corresponding with the manual revision number). The high byte is the number before the decimal point, and the low byte is the one after. For example, version 1.4 would be 0x0104.

0x02: Motor PWM Output

0x00 - Left Motor (RW, 1 byte)

The left motor pulse width (0 = 1ms, 255 = 2ms). Default value: 127.

0x01 - Right Motor (RW, 1 byte)

The left motor pulse width (0 = 1ms, 255 = 2ms). Default value: 127.

0x03: Encoders

TODO: Index directionality

0x00 - Encoder Control (RW, 1 byte)

Bit	7(MSB)	6	5	4	3	2	1	0 (LSB)
Meaning	LA	LB	LI	RA	RB	RI		RESET

RESET: Writing 1, then 0, resets both left and right encoders. While 1, the encoders will not operate.

LA: Current state of Left A pin.

LB: Current state of Left B pin.

LI: Current state of Left Index pin.

RA: Current state of Right A pin.

RB: Current state of Right B pin.

RI: Current state of Right Index pin.

0x01 - Left Encoder Index Count (R, 4 bytes)

Count of index pulses received on left channel.

0x02 - Left Encoder Pulse Count (R, 4 bytes)

Count (2's complement signed) of encoder pulses received on left channel. 2048 pulses are produced per revolution of the wheel.

0x03 - Right Encoder Index Count (R, 4 bytes)

Count of index pulses received on right channel.

0x04 - Right Encoder Pulse Count (R, 4 bytes)

Count (2's complement signed) of encoder pulses received on right channel. 2048 pulses are produced per revolution of the wheel.

0x04: Arm

MS2	MS1	MS0	Step Increment
0	0	0	1
0	0	1	$\frac{1}{2}$
0	1	0	$\frac{1}{4}$
0	1	1	$\frac{1}{8}$
1	0	0	$\frac{1}{16}$
1	0	1	$\frac{1}{32}$
1	1	0	$\frac{1}{32}$
1	1	1	$\frac{1}{32}$

Table 1: Microstep settings

0x00 - X Configuration (RW, 1 byte)

Bit	7(MSB)	6	5	4	3	2	1	0 (LSB)
Meaning	GO	EN	DIR		STEPPOL	MS2	MS1	MS0
Default	0	1	1		1	0	1	0

MS0-MS2: Microstepping mode (see table 1).

DIR: Direction output.

STEPPOL: Step polarity. If 1, steps are generated with rising edges.

EN: Stepper driver enable line. (For the DRV8825, a 0 enables the driver).

GO: If 1, steps will be generated. If the global pause signal (PAUSE) is asserted, this bit is ignored, and steps will not be generated. A change on the limit switch from unpressed to pressed sets GO to 0.

0x01 - X Status (R, 1 byte)

Bit	7(MSB)	6	5	4	3	2	1	0 (LSB)
Meaning					STEPPING	FAULT	LIMIT	

LIMIT: If 1, the limit switch is pressed. (This is inverted from the polarity of the limit switch input.) A change on the limit switch from unpressed to pressed sets GO to 0.

FAULT: Stepper driver fault line value. This line has a pull-up resistor.

STEPPING: If 1, steps are being generated, and the axis is moving. Steps are generated when STEPS is non-zero, GO is 1, and PAUSE is 0.

0x02 - X Period Factor (RW, 4 bytes)

Division factor (of 12Mhz) to produce the desired step frequency. For example, a period factor of 12 will produce a 1Mhz step frequency. Too high a frequency may result in skipped steps and unreliable operation.

0x03 - X Steps (RW, 4 bytes)

The number of steps to be generated. When set to a non-zero value, steps will be generated at the frequency specified by the period factor register. With each step, the register is decremented. The DIR bit in the configuration register sets the direction of motion.

0x10 - Y Configuration (RW, 1 byte)

Bit	7(MSB)	6	5	4	3	2	1	0 (LSB)
Meaning	GO	EN	DIR		STEPPOL	MS2	MS1	MS0
Default	0	1	1		1	0	1	0

MS0-MS2: Microstepping mode (see table 1).

DIR: Direction output.

STEPPOL: Step polarity. If 1, steps are generated with rising edges.

EN: Stepper driver enable line. (For the DRV8825, a 0 enables the driver).

GO: If 1, steps will be generated. If the global pause signal (PAUSE) is asserted, this bit is ignored, and steps will not be generated. A change on the limit switch from unpressed to pressed sets GO to 0.

0x11 - Y Status (R, 1 byte)

Bit	7(MSB)	6	5	4	3	2	1	0 (LSB)
Meaning					STEPPING	FAULT	LIMIT	

LIMIT: If 1, the limit switch is pressed. (This is inverted from the polarity of the limit switch input.) A change on the limit switch from unpressed to pressed sets GO to 0.

FAULT: Stepper driver fault line value. This line has a pull-up resistor.

STEPPING: If 1, steps are being generated, and the axis is moving. Steps are generated when Steps is non-zero, GO is 1, and PAUSE is 0.

0x12 - Y Period Factor (RW, 4 bytes)

Division factor (of 12Mhz) to produce the desired step frequency. For example, a period factor of 12 will produce a 1Mhz step frequency. Too high a frequency may result in skipped steps and unreliable operation.

0x13 - Y Steps (RW, 4 bytes)

The number of steps to be generated. When set to a non-zero value, steps will be generated at the frequency specified by the period factor register. With each step, the register is decremented. The DIR bit in the configuration register sets the direction of motion.

0x20 - Z Configuration (RW, 1 byte)

Bit	7(MSB)	6	5	4	3	2	1	0 (LSB)
Meaning	GO	EN	DIR		STEPPOL	MS2	MS1	MS0
Default	0	1	1		1	0	1	0

MS0-MS2: Microstepping mode (see table 1).

DIR: Direction output.

STEPPOL: Step polarity. If 1, steps are generated with rising edges.

EN: Stepper driver enable line. (For the DRV8825, a 0 enables the driver).

GO: If 1, steps will be generated. If the global pause signal (PAUSE) is asserted, this bit is ignored, and steps will not be generated. A change on the limit switch from unpressed to pressed sets GO to 0.

0x21 - Z Status (R, 1 byte)

Bit	7(MSB)	6	5	4	3	2	1	0 (LSB)
Meaning					STEPPING	FAULT	LIMIT	

LIMIT: If 1, the limit switch is pressed. (This is inverted from the polarity of the limit switch input.) A change on the limit switch from unpressed to pressed sets GO to 0.

FAULT: Stepper driver fault line value. This line has a pull-up resistor.

STEPPING: If 1, steps are being generated, and the axis is moving. Steps are generated when Steps is non-zero, GO is 1, and PAUSE is 0.

0x22 - Z Period Factor (RW, 4 bytes)

Division factor (of 12Mhz) to produce the desired step frequency. For example, a period factor of 12 will produce a 1Mhz step frequency. Too high a frequency may result in skipped steps and unreliable operation.

0x23 - Z Steps (RW, 4 bytes)

The number of steps to be generated. When set to a non-zero value, steps will be generated at the frequency specified by the period factor register. With each step, the register is decremented. The DIR bit in the configuration register sets the direction of motion.

0x30 - A Configuration (RW, 1 byte)

Bit	7(MSB)	6	5	4	3	2	1	0 (LSB)
Meaning	GO	EN	DIR		STEPPOL	MS2	MS1	MS0
Default	0	1	1		1	0	1	0

MS0-MS2: Microstepping mode (see table 1).

DIR: Direction output.

STEPPOL: Step polarity. If 1, steps are generated with rising edges.

EN: Stepper driver enable line. (For the DRV8825, a 0 enables the driver).

GO: If 1, steps will be generated. If the global pause signal (PAUSE) is asserted, this bit is ignored, and steps will not be generated. A change on the limit switch from unpressed to pressed sets GO to 0.

0x31 - A Status (R, 1 byte)

Bit	7(MSB)	6	5	4	3	2	1	0 (LSB)
Meaning					STEPPING	FAULT	LIMIT	

LIMIT: If 1, the limit switch is pressed. (This is inverted from the polarity of the limit switch input.) A change on the limit switch from unpressed to pressed sets GO to 0.

FAULT: Stepper driver fault line value. This line has a pull-up resistor.

STEPPING: If 1, steps are being generated, and the axis is moving. Steps are generated when Steps is non-zero, GO is 1, and PAUSE is 0.

0x32 - A Period Factor (RW, 4 bytes)

Division factor (of 12Mhz) to produce the desired step frequency. For example, a period factor of 12 will produce a 1Mhz step frequency. Too high a frequency may result in skipped steps and unreliable operation.

0x33 - A Steps (RW, 4 bytes)

The number of steps to be generated. When set to a non-zero value, steps will be generated at the frequency specified by the period factor register. With each step, the register is decremented. The DIR bit in the configuration register sets the direction of motion.

0x0A - Analog 1 (R, 2 bytes)

Analog channel 1 voltage, in millivolts.

0x0B - Analog 2 (R, 2 bytes)

Analog channel 2 voltage, in millivolts.

0x0C - Analog 3 (R, 2 bytes)

Analog channel 3 voltage, in millivolts.

0x0D - Analog 4 (R, 2 bytes)

Analog channel 4 voltage, in millivolts.

0x05: GPIO

0x00 - GPIO Direction (RW, 1 byte)

Bit	7(MSB)	6	5	4	3	2	1	0 (LSB)
Meaning			DIR5	DIR4				

DIR4: Direction of expansion GPIO 4. 1 = output, 0 = input.

DIR5: Direction of expansion GPIO 5. 1 = output, 0 = input.

0x01 - GPIO Output (RW, 1 byte)

Bit	7(MSB)	6	5	4	3	2	1	0 (LSB)
Meaning			DIR5	DIR4	DIR3	DIR2	DIR1	

DIR1: Output of expansion GPIO 1, if configured as output.

DIR2: Output of expansion GPIO 2, if configured as output.

DIR3: Output of expansion GPIO 3, if configured as output.

DIR4: Output of expansion GPIO 4, if configured as output.

DIR5: Output of expansion GPIO 5, if configured as output.

0x02 - GPIO Input (R, 1 byte)

Bit	7(MSB)	6	5	4	3	2	1	0 (LSB)
Meaning			DIR5	DIR4	DIR3	DIR2	DIR1	

DIR1: Current state of expansion GPIO 1, regardless of input/output setting.

DIR2: Current state of expansion GPIO 2, regardless of input/output setting.

DIR3: Current state of expansion GPIO 3, regardless of input/output setting.

DIR4: Current state of expansion GPIO 4, regardless of input/output setting.

DIR5: Current state of expansion GPIO 5, regardless of input/output setting.

0x06: IMU

TODO: Update rate, units, range, integration?

0x00 - Lock (RW, 1 byte)

Bit	7(MSB)	6	5	4	3	2	1	0 (LSB)
Meaning							LOCK	
Default							0	

LOCK: If 0, the registers are updated with new data when it's available. If 1, the current value is locked in the registers. This ensures that reading from all the registers returns data from the same sample.

0x01 - X Acceleration (R, 2 bytes)

0x02 - Y Acceleration (R, 2 bytes)

0x03 - Z Acceleration (R, 2 bytes)

0x04 - X Angular Rate (R, 2 bytes)

0x05 - Y Angular Rate (R, 2 bytes)

0x06 - Z Angular Rate (R, 2 bytes)

0x07: RC Receiver

0x00 - RC Status (R, 1 byte)

Bit	7(MSB)	6	5	4	3	2	1	0 (LSB)
Meaning	VALID8	VALID7			VALID4	VALID3	VALID2	VALID1

VALID1: If 1, the value in the Channel 1 is valid.
VALID2: If 1, the value in the Channel 2 is valid.
VALID3: If 1, the value in the Channel 3 is valid.
VALID4: If 1, the value in the Channel 4 is valid.
VALID7: If 1, the value in the Channel 7 is valid.
VALID8: If 1, the value in the Channel 8 is valid.

0x01 - Channel 1 (R, 1 byte)

Pulse width on RC channel 1. $0 = \leq 1\text{ms}$, $255 = \geq 2\text{ms}$.

0x02 - Channel 2 (R, 1 byte)

Pulse width on RC channel 2. $0 = \leq 1\text{ms}$, $255 = \geq 2\text{ms}$.

0x03 - Channel 3 (R, 1 byte)

Pulse width on RC channel 3. $0 = \leq 1\text{ms}$, $255 = \geq 2\text{ms}$.

0x04 - Channel 4 (R, 1 byte)

Pulse width on RC channel 4. $0 = \leq 1\text{ms}$, $255 = \geq 2\text{ms}$.

0x07 - Channel 7 (R, 1 byte)

Pulse width on RC channel 7. $0 = \leq 1\text{ms}$, $255 = \geq 2\text{ms}$.

0x08 - Channel 8 (R, 1 byte)

Pulse width on RC channel 8. $0 = \leq 1\text{ms}$, $255 = \geq 2\text{ms}$.

5 Assembly and Programming Procedure

1. Assemble top-side components using solder paste and a reflow oven.
2. Assemble bottom-side components using a soldering iron.
Note: The TVS diodes originally specified in the design are no longer available. I substituted a near-equivalent in the Dec. 2015 Digikey order. These have an opposite pinout to the originals.
3. Assemble connectors using a soldering iron.
4. Power-up and Test:
 - (a) Apply 24V using a current-limited supply set to 50mA max.
 - (b) Check Vbatt, 5V, and 3.3V voltages. All three green LEDs should be lit.
 - (c) Check Vbatt, 5V, and 3.3V for oscillation using an oscilloscope.
 - (d) Connect Uniboard to computer with a micro-USB cable. FTDI device should appear.
 - (e) Program FTDI device with Uniboard configuration file (containing the Lattice eval board config with the suspend on DBUS7 option set) using the FT_Prog application. The file is in the software folder.
 - (f) Program FPGA using Lattice diamond. (On Linux systems, it may be necessary to remove the standard FTDI kernel module (with `sudo modprobe -r ftdi_sio`) to make the FTDI JTAG work.)
 - (g) Test each peripheral

6 Design Qualification Tests

These tests verify that the design is correct and can work reliably. They do not need to be performed on each assembled board. The results here were gathered using an HP E3611A power supply (providing V_{batt} at 28V/500mA max.), an Agilent 54642D oscilloscope (with Agilent 10073C probe on channel 1 and Agilent N2863A probe on channel 2), and a cheap Chinese multimeter.

6.1 Power Supply

Power supply measurements were carried out with the FPGA outputting 12Mhz, 1Mhz, 100kHz, and 10kHz signals on the stepper control lines in order to create an absolutely worst-case scenario in terms of switching noise. Voltages were measured at the voltage test points near the green LEDs. See fig. 10 for measurement setup.

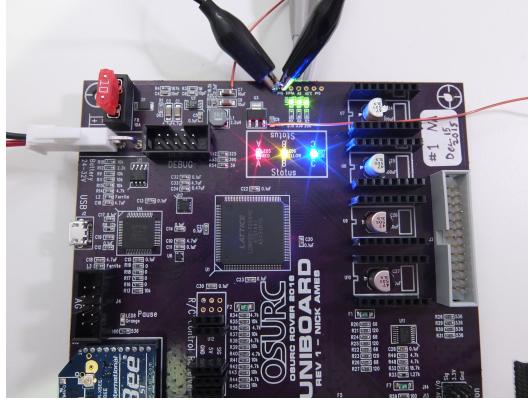


Figure 10: Power supply measurement setup.

Load	Current (mA)	Voltage (V)	28V Vbatt Current (mA)
0	0	4.97	30
Half	500	4.96	130
Full	1000	4.95	220

(a) 5V Rail

Load	Current (mA)	Voltage (V)
0	0	3.26
Half	150	3.26
Full	300	3.26

(b) 3.3V Rail

Table 2: Power rail voltages at loads.
Each was measured with the other rail unloaded.

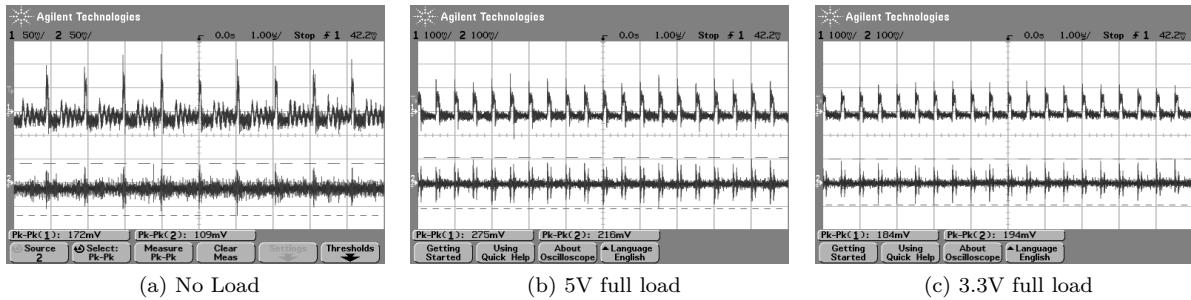


Figure 11: Power supply noise. Ch. 1 is 5V rail, ch. 2 is 3.3V rail.

6.2 Serial Communication

To test the electrical integrity of the FTDI->FPGA link, 100MB of random data was sent through it. The FPGA acted as a simple loop-back, connecting its TX and RX lines together. During this process, other signal lines were

toggled at high frequencies. Comparing the sent and received data indicated that there was no corruption.

While developing the HDL code, a test program read and wrote the motor PWM and RC receiver registers 170,000 times without any issues.

TODO: More tests.

7 Revision History

Both this document and the Uniboard design share the same revision number.

- 1.0: Initial version.

8 Assembly Log

#1 (Revision 1)

Assembled Dec. 15, 2015 by Nick Ames. Tested/Initialized Dec. 19th, 2015 by Nick Ames.

- Different TVS diodes substituted (originally specified ones no longer available). These have a reversed pin-out from the originals.
- 47uF capacitors from IEEE store substituted for 100uF ones (they didn't get ordered due to an error on the BOM).
- Status LED C (Blue) resistor substituted to reduce brightness.
- RC receiver footprint was slightly incorrect, so I didn't populate the power or bind channel connectors. Bits of wire were used to transfer power to the ch. 8 connector.
- Added 4.7uF capacitors across 120 ohm resistors in the arm limit inputs, to suppress stepper noise.

A Bill of Materials

TODO: Insert bill of materials.

B Schematic

6
5
4
3
2
1

D

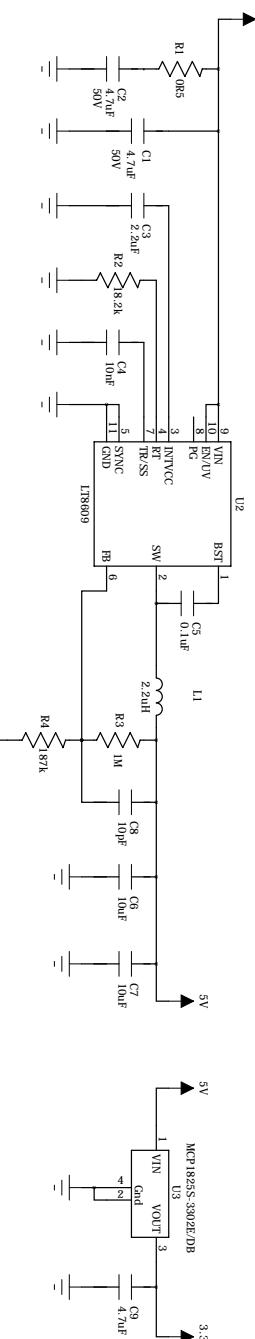
Battery Input
24-32V

CON-MOLEX-0441300018
CON-MOLEX-0441300018
J1-1 C
J1-2 C
F8 10A
D1 DIODE

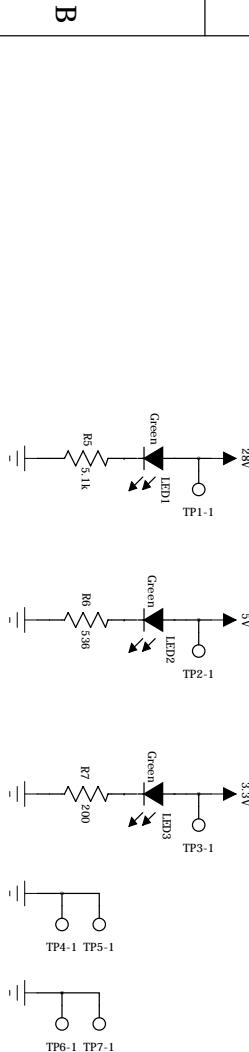
The actual battery voltage varies from 24-32V.

REVISON RECORD			
LTR	ECO NO.	APPROVED	DATE

C



C



B

X1
MECH. HOLE 3MM
X2
MECH. HOLE 3MM
X3
MECH. HOLE 3MM
X4
MECH. HOLE 3MM

COMPANY:
OSU Robotics Club Rover 2016
TITLE:
Uniboard

DRAWN:
Nick Ames

DATED:
2015-11-13

TITLE:

<Code> B <Drawing Number> 1

CHECKED:
<Checked By>
CODE:
<Checked Date>

QUALITY CONTROL:
<QC By>
CODE:
<QC Date>

RELEASED:
<Released By>
CODE:
<Release Date>

SCALE: <Scale>

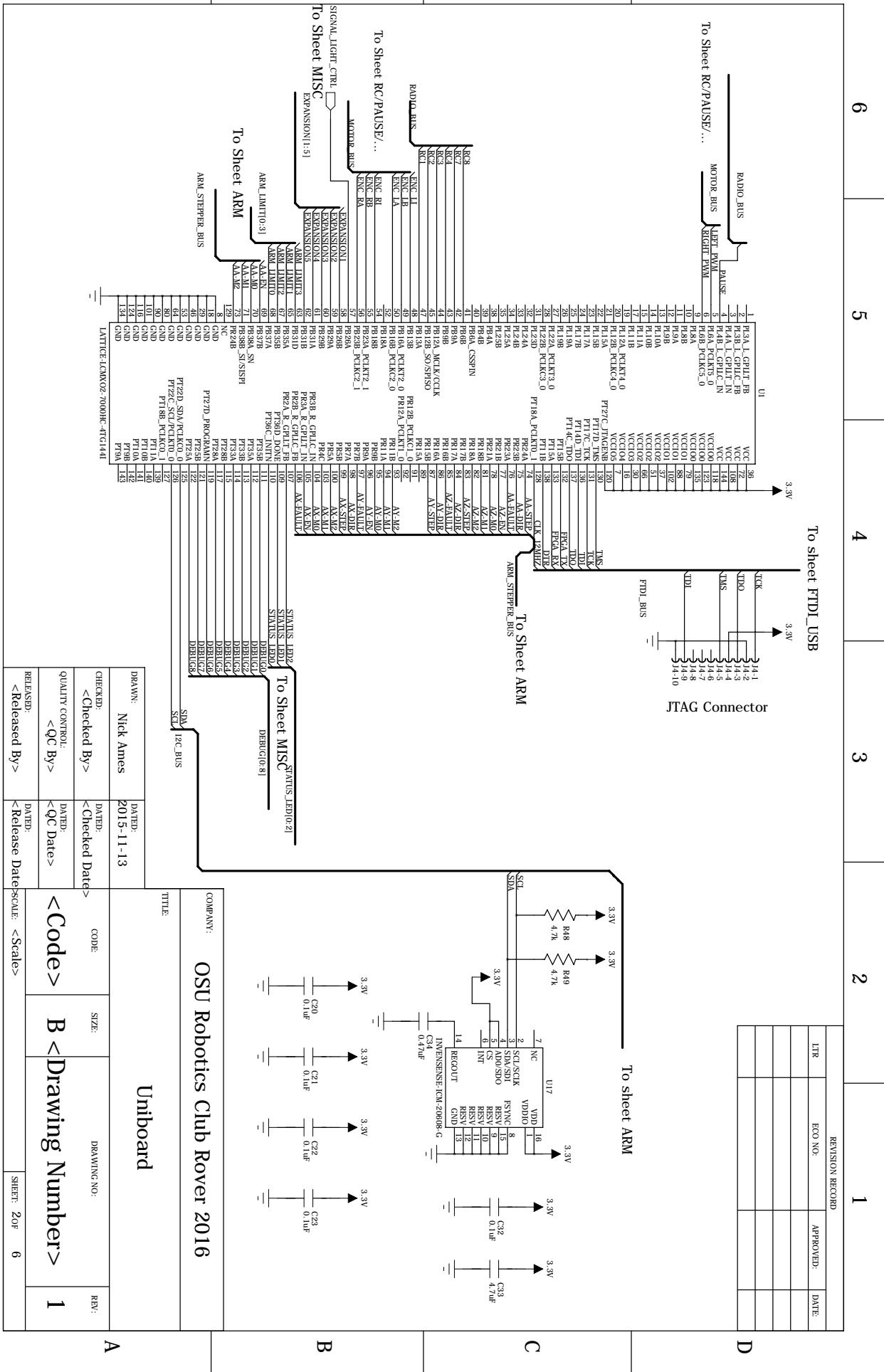
REV: 6

SHEET: 1 of 6

A

A

6 | 5 | 4 | 3 | 2 | 1



6

5

4

3

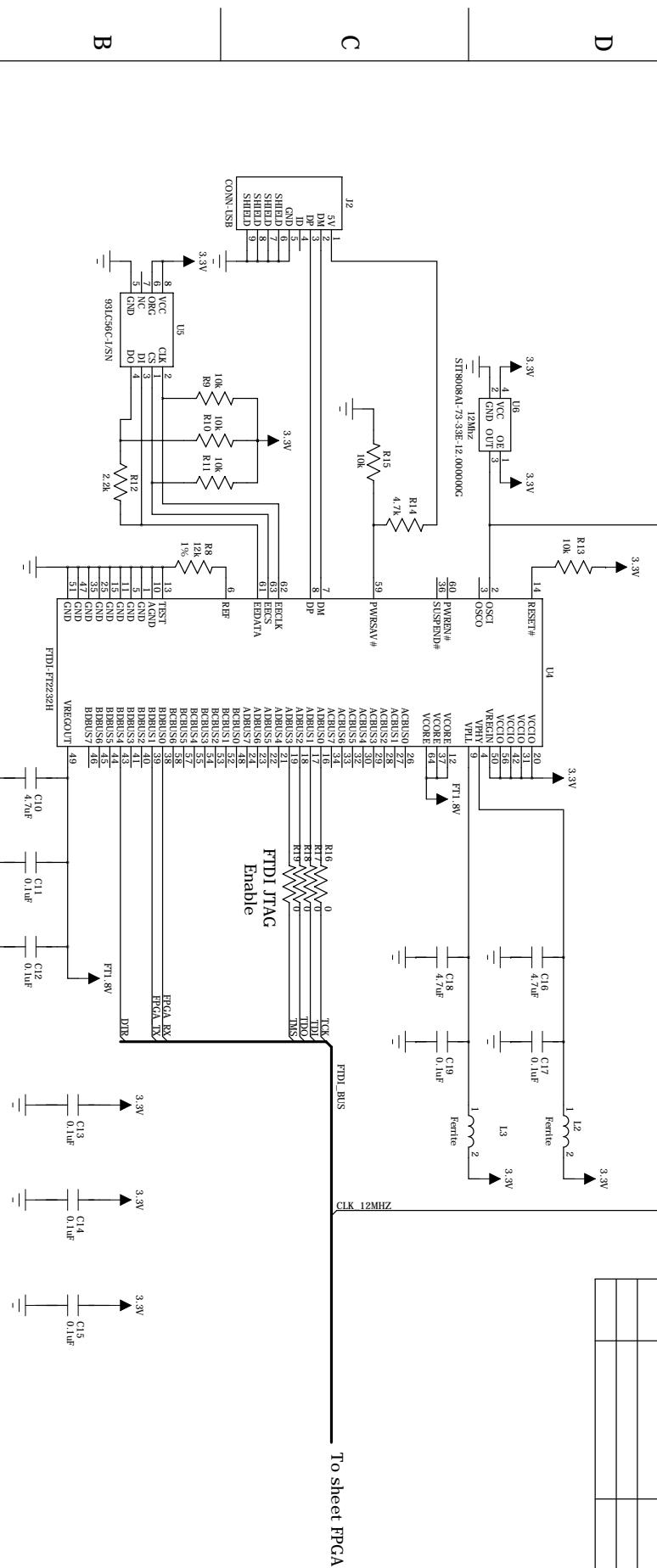
2

1

D

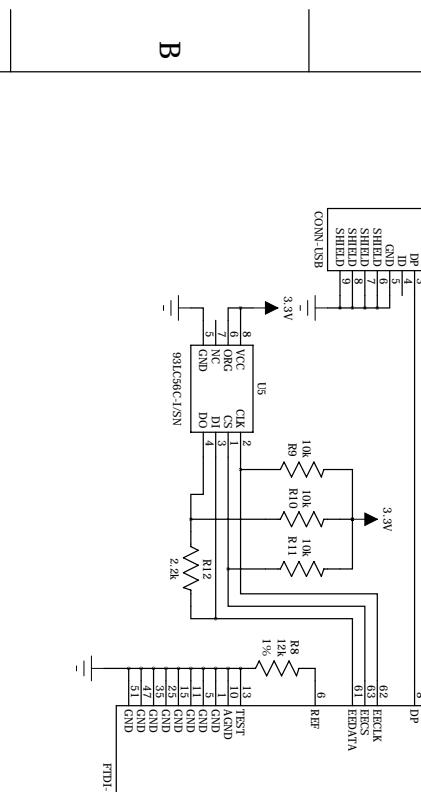
REVISON RECORD			
LTR	ECO NO.	APPROVED	DATE

D



C

C



B

B

COMPANY: OSU Robotics Club Rover 2016
TITLE: Uniboard

A

A

DRAWN:	Nick Ames	DATED:	2015-11-13
CHECKED:	<Checked By>	DATED:	<Checked Date>
QUALITY CONTROL:	<QC By>	DATED:	<QC Date>
RELEASED:	<Released By>	DATED:	<Release Date>
SCALE:	<Scale>	DRAWING NO.:	1
REV.:		SHEET:	3 of 6

6

5

4

3

2

1

REVISON RECORD			
LTR	ECO NO.	APPROVED	DATE

D

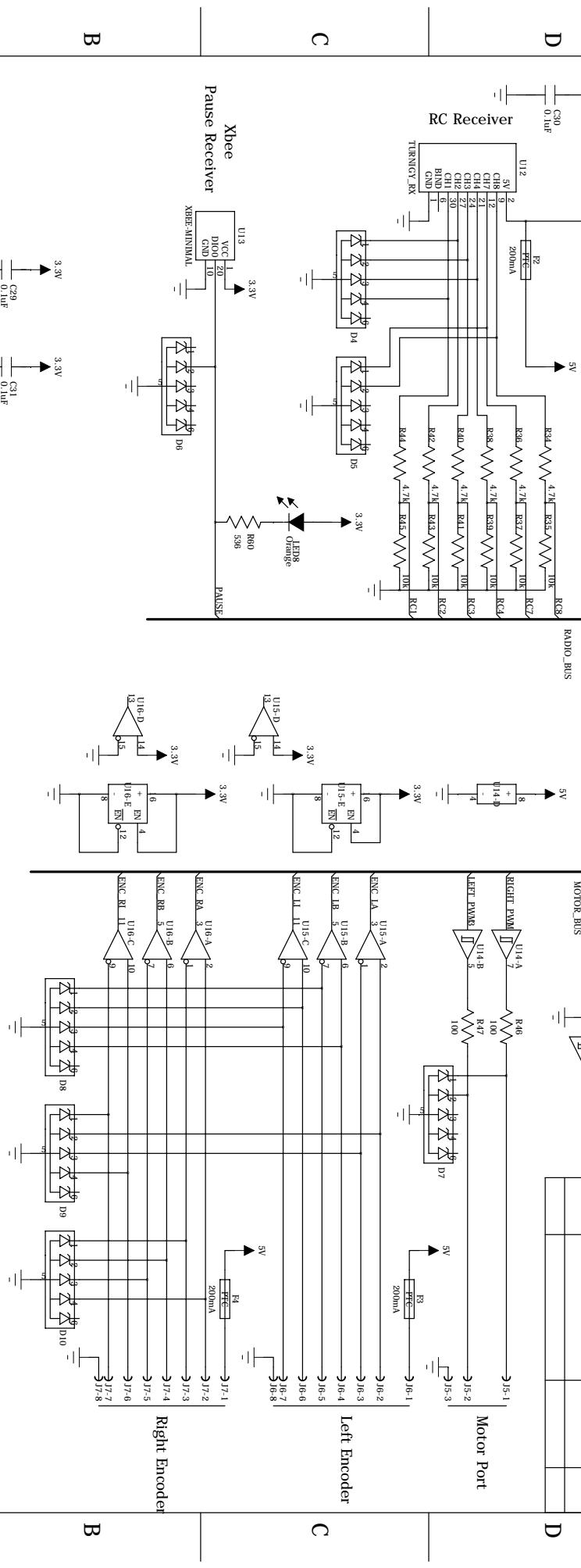
C

D

To sheet FPGA

To sheet FPGA

D



COMPANY: OSU Robotics Club Rover 2016
TITLE: Uniboard

A

A

DRAWN:	Nick Ames	DATED:	2015-11-13
CHECKED:	<Checked By>	DATED:	<Checked Date>
QUALITY CONTROL:	<QC By>	DATED:	<QC Date>
RELEASED:	<Released By>	DATED:	<Release Date>
SCALE:	<Scale>	DRAWING NO.:	1
REV.:		SHEET:	5 of 6

6

5

4

3

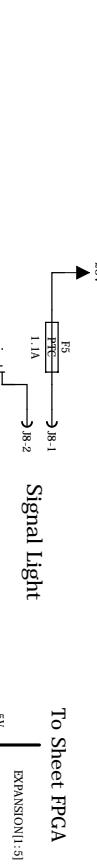
2

1

REVISON RECORD			
LTR	ECO NO.	APPROVED	DATE

D

D



To Sheet FPGA
EXPANSION[1:5]

3.3V → 5V → F7 → 200mA → R6 → 500mA → Expansion 1

3.3V → 5V → J10-1 → J10-2 → R55 → Expansion 2

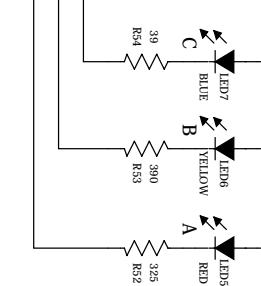
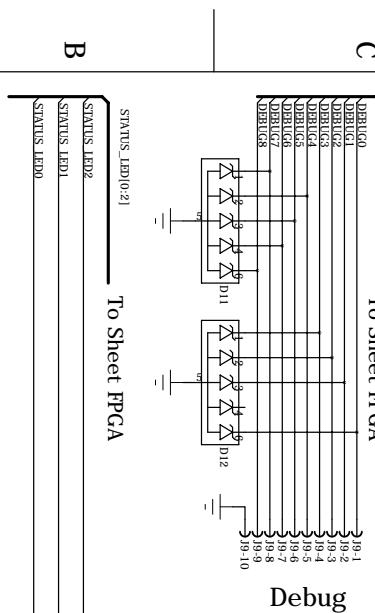
3.3V → 5V → J11-1 → J11-2 → R56 → Expansion 3

3.3V → 5V → J12-1 → J12-2 → R57 → Expansion 4

3.3V → 5V → J13-1 → J13-2 → R58 → Expansion 5

C

C



B

B

A

A

COMPANY:		OSU Robotics Club Rover 2016	
TITLE:			
DRAWN:		DATED:	
Nick Ames	2015-11-13		
CHECKED:			
<Checked By>	<Checked Date>	CODE:	SIZE:
QUALITY CONTROL:			
<QC By>	<QC Date>	DRAWING NO.:	REV:
RELEASED:			
<Released By>	<Release Date>	<Scale>	SHEET: 6 of 6