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.

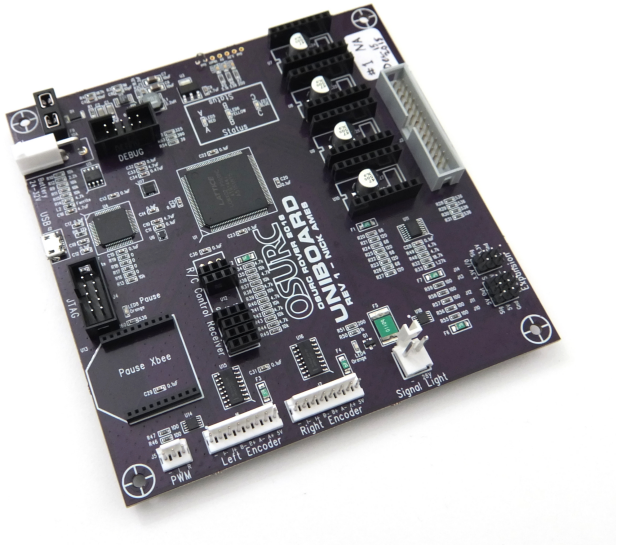


Figure 1: Uniboard without plug-in modules.

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) axises
- 4x Limit switch inputs, for X, Y, Z, and A (open-close) axises
- 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

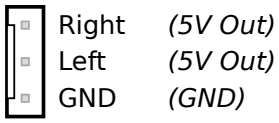
Connector Parts	
Board Header	Molex 441300018 (WM7513-ND)
Wire Connector	Molex 0039013028 (WM2837-ND)
Wire Pins	Molex 0457501111 (WM3279CT-ND)
Wire	18 AWG Stranded Wire

Figure 2: Power Connector

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

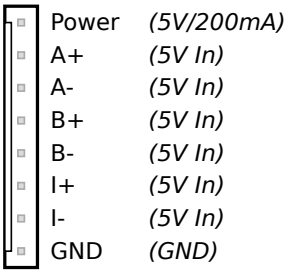
1.2 Motor Outputs

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 thee 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	$1\frac{1}{2}$ twisted pairs from cat-5 cable

Figure 3: Motor PWM Connector



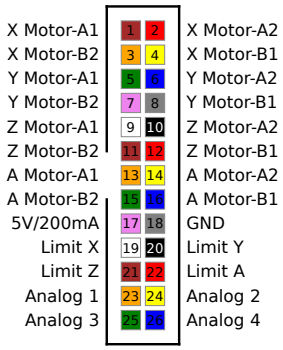
Connector Parts	
Board Header	TE 640454-8 (A19435-ND)
Wire Connector	TE 3-643814-8 (A31023-ND)
Strain Relief	TE 643075-8 (A19236-ND)
Wire	4 twisted pairs from cat-5 cable

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



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.

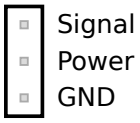
1.5 Signal Light

An orange signal light, indicating the pause state of the robot, is mandated by the SRRC 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.

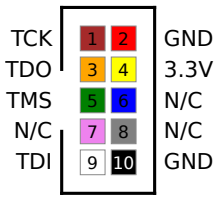
1.7 Xbee

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

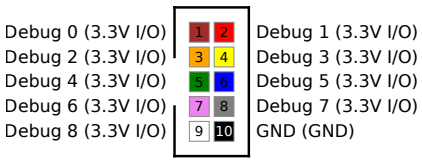
In case the built-in FTDI JTAG system doesn't work, a connector for an external JTAG programmer is provided. To use it, the four jumper resistors (R16-19) between the FTDI chip and JTAG lines should be removed.

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.)



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

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:

$$<ST><\text{peripheral}><\text{register}>[\text{data0}][\text{data1}][\text{data2}][\text{data3}]<EN>$$

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 = unpaused. 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 = unpaused.

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

0x00 - Left Encoder Control (RW, 1 byte)

Bit	7(MSB)	6	5	4	3	2	1	0 (LSB)
Meaning					A	B	I	

A: Current state of Left A pin.
B: Current state of Left B pin.
I: Current state of Left Index pin.

0x01 - Left Encoder Speed (R, 4 bytes)

Count (2's complement signed) of encoder pulses received in the last 10ms. The quadrature encoder increases the resolution by four times, and there's a 27:1 gear reduction from the motor to the wheel. To obtain RPM, multiple the this value by 60/(.01*2048*4*27).

0x10 - Right Encoder Control (RW, 1 byte)

Bit	7(MSB)	6	5	4	3	2	1	0 (LSB)
Meaning					I	B	A	

A: Current state of right A pin.
B: Current state of right B pin.
I: Current state of right Index pin.

0x11 - Right Encoder Speed (R, 4 bytes)

Count (2's complement signed) of encoder pulses received in the last 100ms.

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

0x04: Arm

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 may 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 may 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 may 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 may 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. $0 = \leq$ 1ms, $255 = \geq$ 2ms.

0x02 - Channel 2 (R, 1 byte)

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

0x03 - Channel 3 (R, 1 byte)

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

0x04 - Channel 4 (R, 1 byte)

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

0x07 - Channel 7 (R, 1 byte)

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

0x08 - Channel 8 (R, 1 byte)

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

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
5. Program Xbee modules
   (a) Download and install the Digi XCTU Xbee programming utility.
   (b) Place the Xbee S1 Pro that will be installed in the pause switch in a Sparkfun Xbee programming board. XCTU may ask to update the firmware in the module; allow it to do so.

(c) Use the XCTU application to program the Xbee with the pause_switch_xbee.xml file located in the software/xbee_config/ folder.

(d) Repeat the process with the Uniboard Xbee and the uniboard_xbee_default_unpaused.xml file.

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.

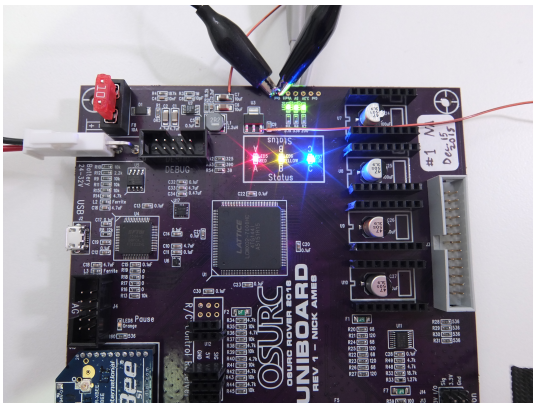


Figure 10: Power supply measurement setup.

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.

6.2 Serial Communication

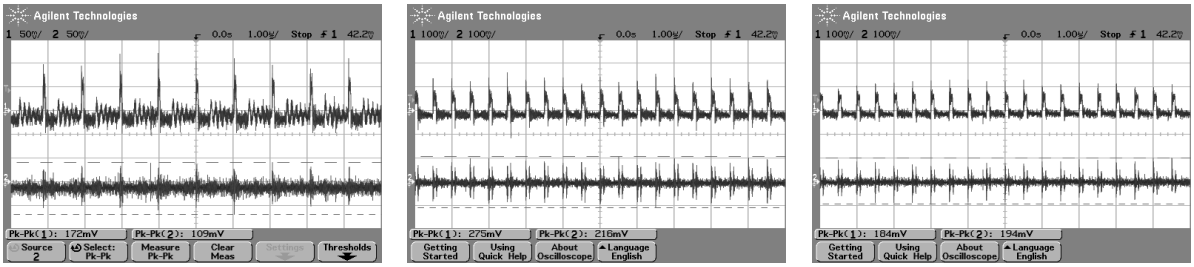
To test the electrical integrity of the FTDI->FPGA link, 100MB of random data was sent through it. The FPGA acted as a simple loop-back, connecting its TX and RX lines together. During this process, other signal lines were toggled at high frequencies. Comparing the sent and received data indicated that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



(a) No Load (b) 5V full load (c) 3.3V full load

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

- 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 the 120 ohm resistors in the arm limit inputs, to suppress stepper noise.

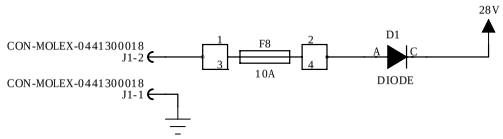
A Bill of Materials

TODO: Insert bill of materials.

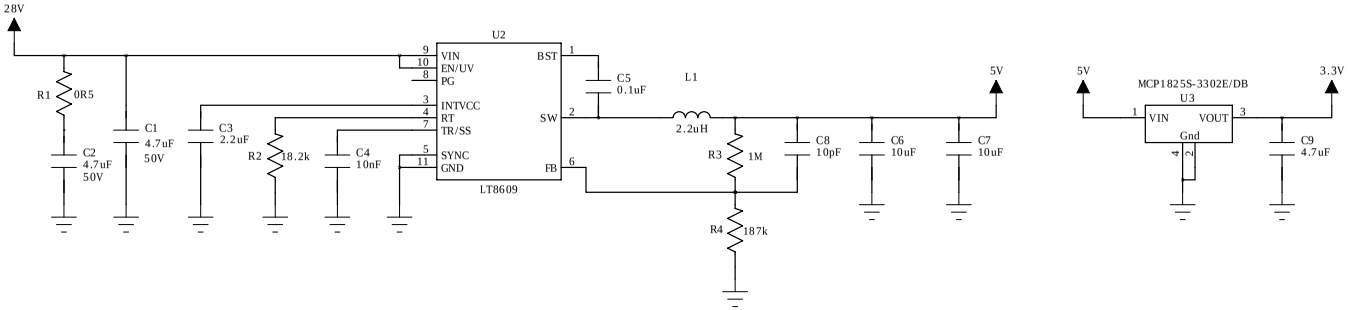
B Schematic

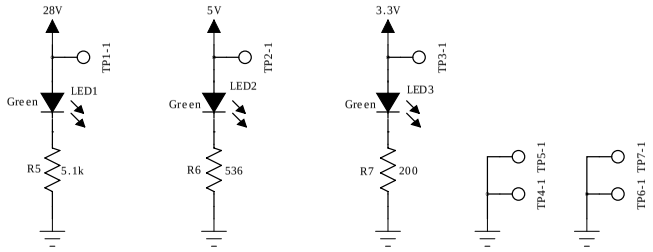
REVISION RECORD			
LTR	ECO NO:	APPROVED:	DATE:

Battery Input
24-32V

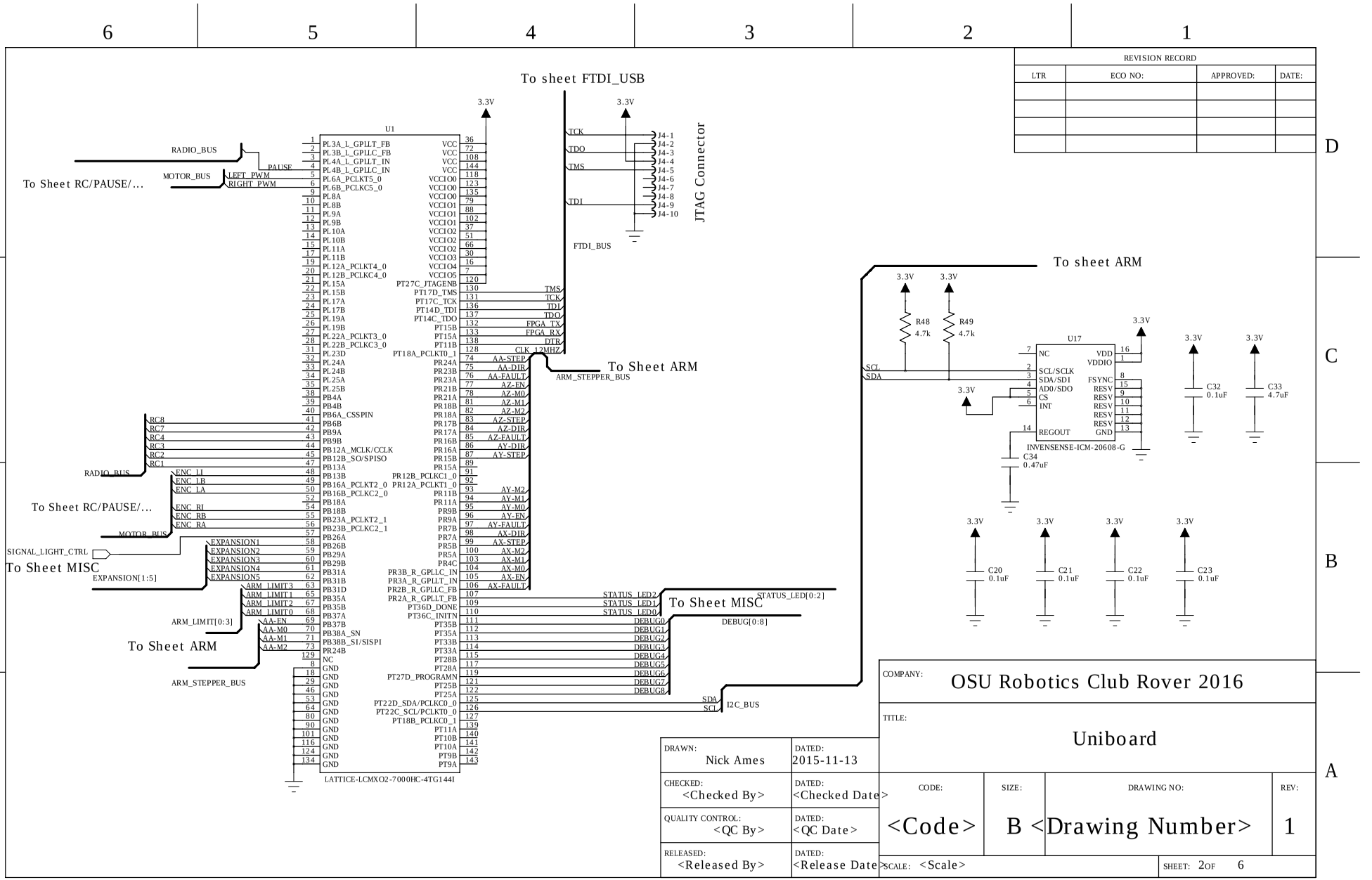


The actual battery voltage varies from 24-32V.

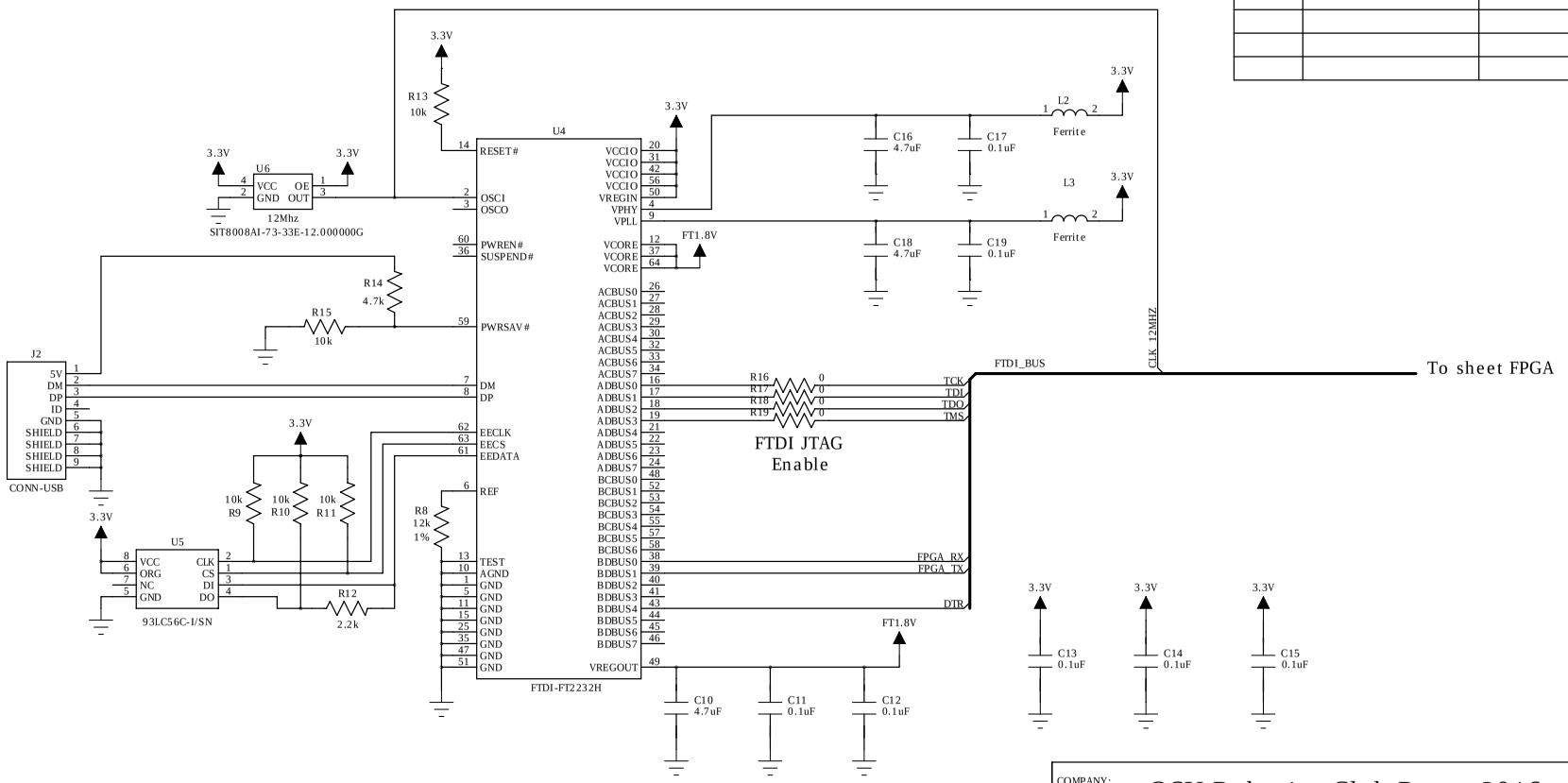




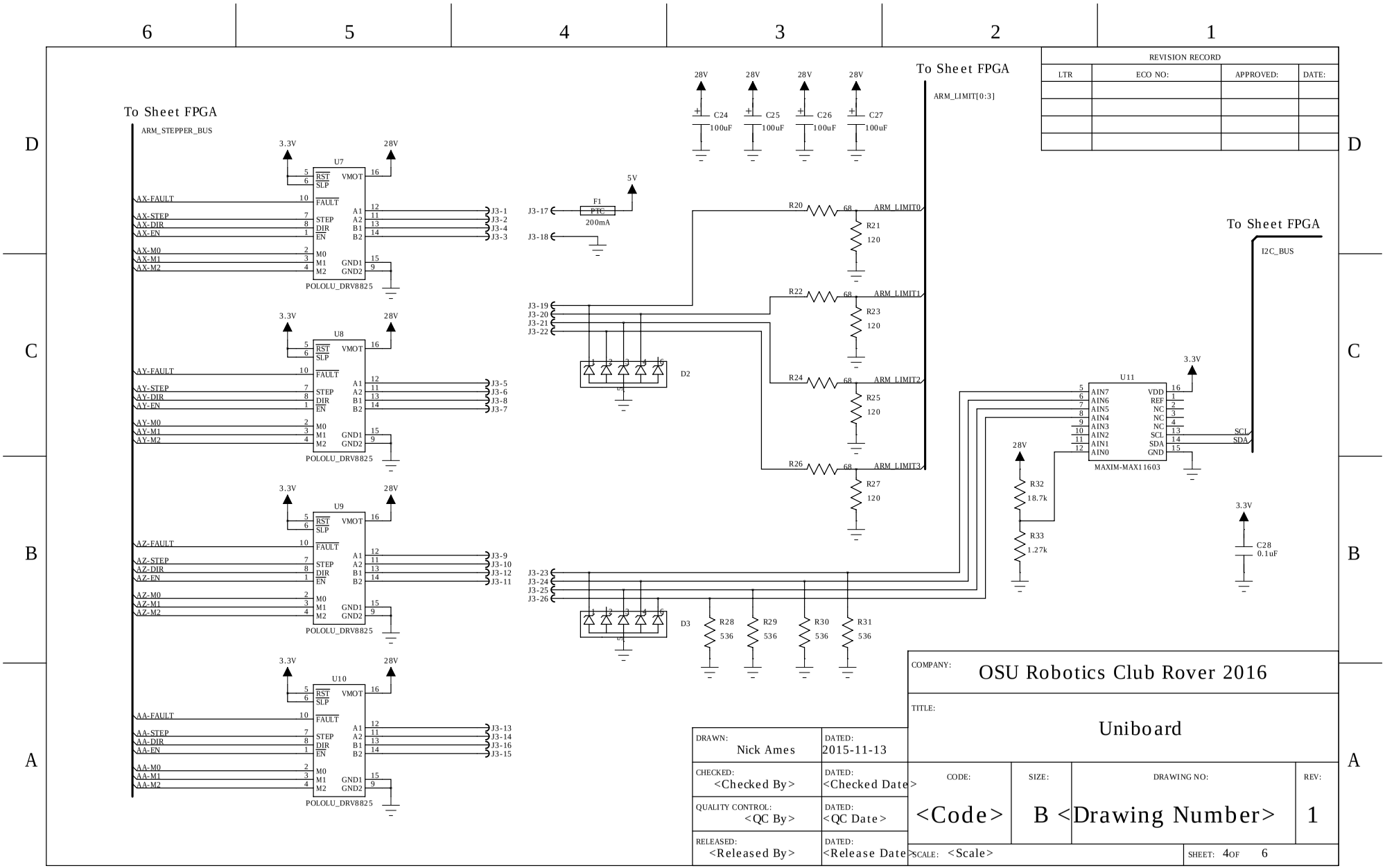
DRAWN: Nick Ames	DATED: 2015-11-13	COMPANY: OSU Robotics Club Rover 2016	
CHECKED: <Checked By>	DATED: <Checked Date>	TITLE: Uniboard	
QUALITY CONTROL: <QC By>	DATED: <QC Date>	CODE: <Code>	SIZE: B
RELEASED: <Released By>	DATED: <Release Date>	DRAWING NO: <Drawing Number>	REV: 1
		SCALE: <Scale>	SHEET: 1 OF 6

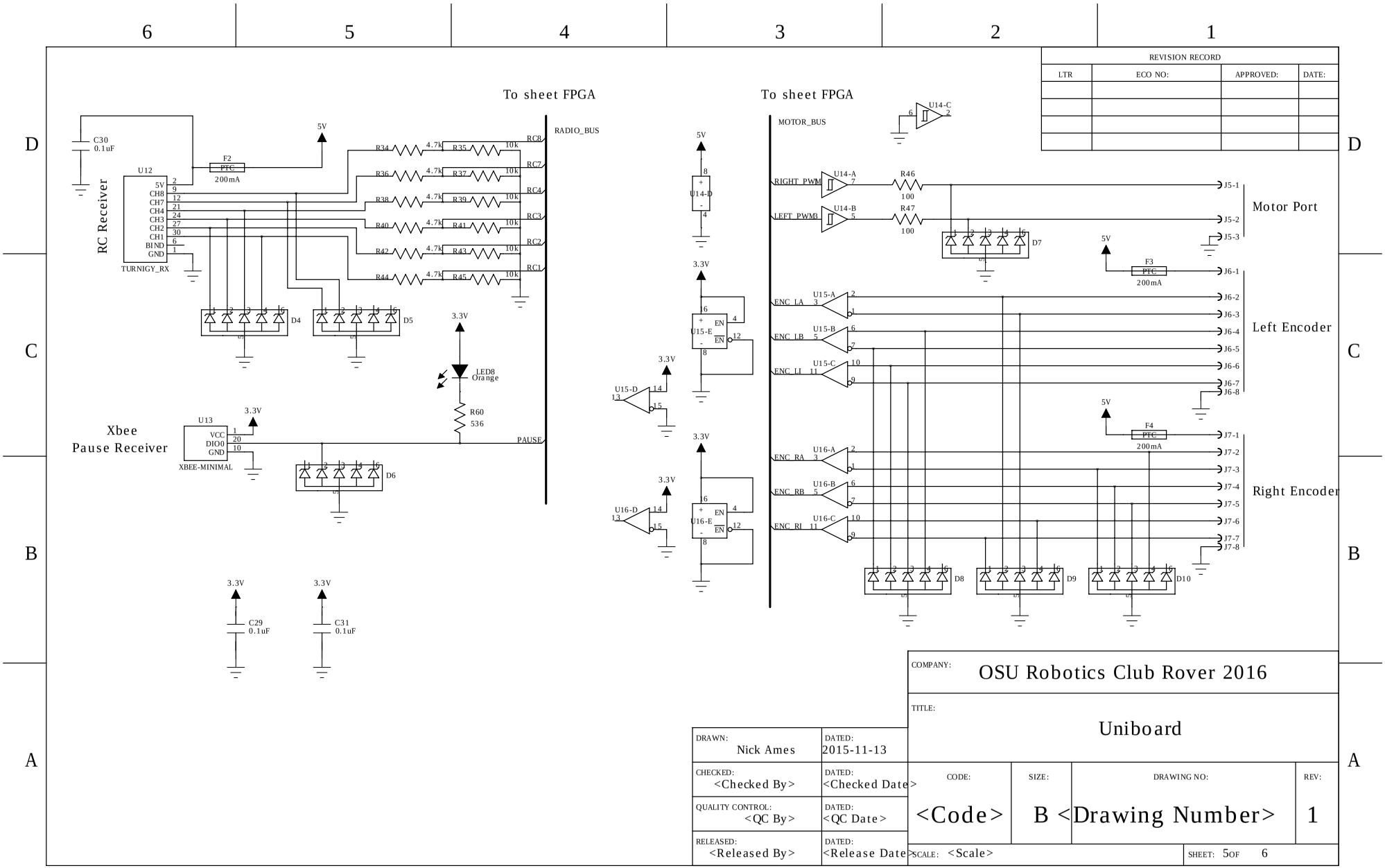

REVISION RECORD
LTR
ECO NO:
APPROVED:
DATE:
To sheet FTDI_USB
To sheet ARM
To Sheet RC/PAUSE/...
To Sheet MISC
To Sheet ARM
JTAG Connector
FTDI_BUS
ARM_STEPPER_BUS
I2C_BUS
DEBUG[0:8]
STATUS_LED[0:2]
EXPANSION[1:5]
ARM_LIMIT[0:3]
SIGNAL_LIGHT_CTRL
RADIO_BUS
MOTOR_BUS
U1
LATTICE-LCMXO2-7000HC-4TG144I
U17
INVENSENSE-ICM-20608-G
R48 4.7k
R49 4.7k
C32 0.1uF
C33 4.7uF
C34 0.47uF
C20 0.1uF
C21 0.1uF
C22 0.1uF
C23 0.1uF
3.3V
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>
COMPANY: OSU Robotics Club Rover 2016
TITLE: Uniboard
CODE: <Code>
SIZE: B
DRAWING NO: <Drawing Number>
REV: 1
SCALE: <Scale>
SHEET: 2 OF 6

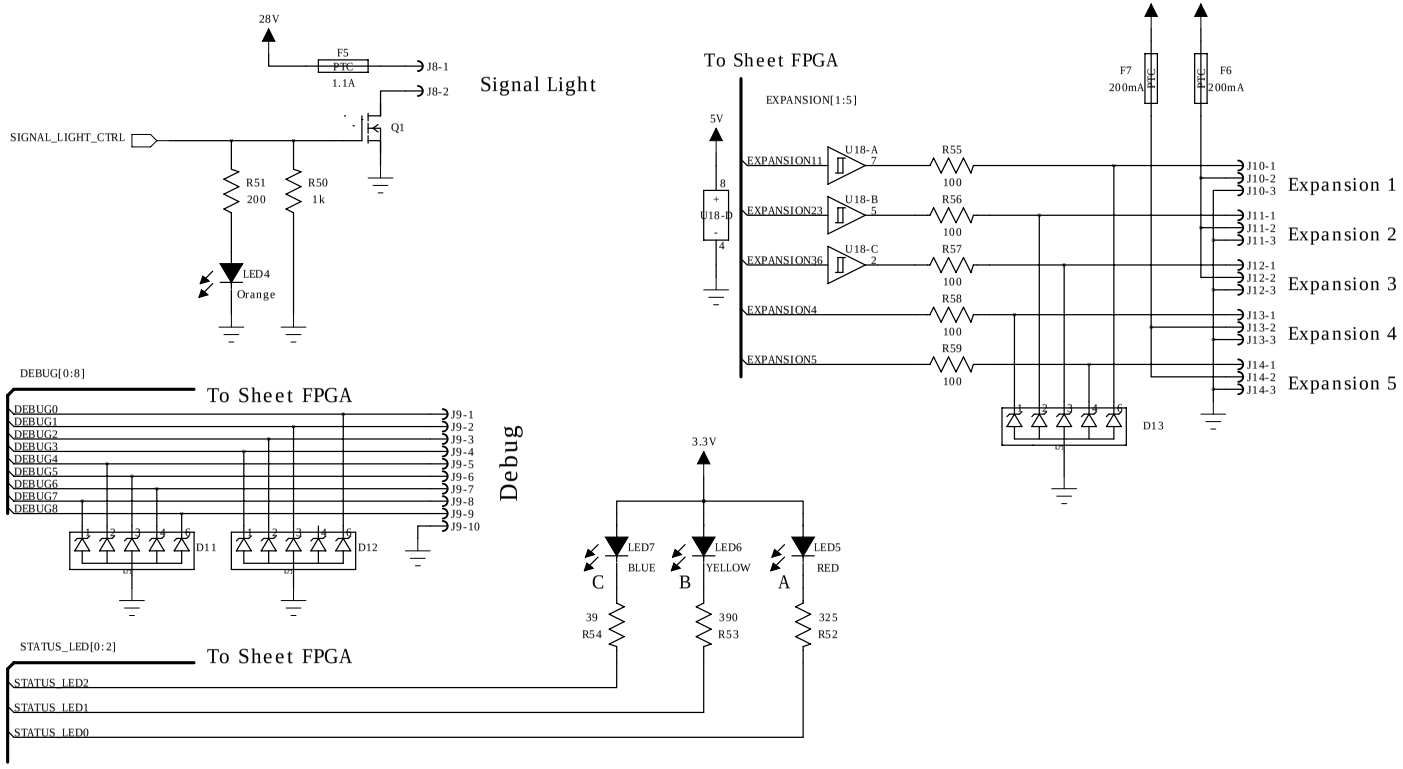
REVISION RECORD			
LTR	ECO NO:	APPROVED:	DATE:



DRAWN: Nick Ames	DATED: 2015-11-13	COMPANY: OSU Robotics Club Rover 2016	
CHECKED: <Checked By>	DATED: <Checked Date>	TITLE: Uniboard	
QUALITY CONTROL: <QC By>	DATED: <QC Date>	CODE: <Code>	SIZE: B
RELEASED: <Released By>	DATED: <Release Date>	DRAWING NO: <Drawing Number>	REV: 1
		SCALE: <Scale>	SHEET: 3 OF 6


6
5
4
3
2
1
D
C
B
A
REVISION RECORD
LTR
ECO NO:
APPROVED:
DATE:
To Sheet FPGA
ARM_STEPPER_BUS
AX-FAULT
AX-STEP
AX-DIR
AX-EN
AX-M0
AX-M1
AX-M2
AY-FAULT
AY-STEP
AY-DIR
AY-EN
AY-M0
AY-M1
AY-M2
AZ-FAULT
AZ-STEP
AZ-DIR
AZ-EN
AZ-M0
AZ-M1
AZ-M2
AA-FAULT
AA-STEP
AA-DIR
AA-EN
AA-M0
AA-M1
AA-M2
3.3V
28V
U7
U8
U9
U10
RST
SLP
VMOT
FAULT
STEP
DIR
EN
M0
M1
M2
GND1
GND2
A1
A2
B1
B2
POLOLU_DRV8825
J3-1
J3-2
J3-3
J3-4
J3-5
J3-6
J3-7
J3-8
J3-9
J3-10
J3-11
J3-12
J3-13
J3-14
J3-15
J3-16
To Sheet FPGA
ARM_LIMIT[0:3]
C24
C25
C26
C27
100uF
5V
F1
PTC
200mA
J3-17
J3-18
J3-19
J3-20
J3-21
J3-22
J3-23
J3-24
J3-25
J3-26
D2
D3
R20
R21
R22
R23
R24
R25
R26
R27
68
120
ARM_LIMIT0
ARM_LIMIT1
ARM_LIMIT2
ARM_LIMIT3
R28
R29
R30
R31
536
To Sheet FPGA
I2C_BUS
U11
MAXIM-MAX11603
AIN7
AIN6
AIN5
AIN4
AIN3
AIN2
AIN1
AIN0
VDD
REF
NC
SCL
SDA
GND
R32
18.7k
R33
1.27k
C28
0.1uF
DRAWN:
Nick Ames
DATED:
2015-11-13
CHECKED:
<Checked By>
<Checked Date>
QUALITY CONTROL:
<QC By>
<QC Date>
RELEASED:
<Released By>
<Release Date>
COMPANY:
OSU Robotics Club Rover 2016
TITLE:
Uniboard
CODE:
<Code>
SIZE:
B
DRAWING NO:
<Drawing Number>
REV:
1
SCALE: <Scale>
SHEET: 4 OF 6


6
5
4
3
2
1
D
C
B
A
REVISION RECORD
LTR
ECO NO:
APPROVED:
DATE:
To sheet FPGA
To sheet FPGA
RC Receiver
TURNIGY_RX
U12
C30
0.1uF
F2
PTC
200mA
5V
RADIO_BUS
R34 4.7k R35 10k RC8
R36 4.7k R37 10k RC7
R38 4.7k R39 10k RC4
R40 4.7k R41 10k RC3
R42 4.7k R43 10k RC2
R44 4.7k R45 10k RC1
D4
D5
3.3V
LED8
Orange
R60
536
PAUSE
Xbee
Pause Receiver
U13
XBEE-MINIMAL
D6
C29
0.1uF
C31
0.1uF
MOTOR_BUS
U14-C
U14-D
U14-A
U14-B
RIGHT_PWM
LEFT_PWM
R46
100
R47
100
D7
F3
PTC
200mA
Motor Port
J5-1
J5-2
J5-3
J6-1
J6-2
J6-3
J6-4
J6-5
J6-6
J6-7
J6-8
U15-A
U15-B
U15-C
U15-D
U15-E
ENC_LA
ENC_LB
ENC_LI
Left Encoder
F4
PTC
200mA
J7-1
J7-2
J7-3
J7-4
J7-5
J7-6
J7-7
J7-8
U16-A
U16-B
U16-C
U16-D
U16-E
ENC_RA
ENC_RB
ENC_RI
Right Encoder
D8
D9
D10
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>
COMPANY: OSU Robotics Club Rover 2016
TITLE: Uniboard
CODE: <Code>
SIZE: B
DRAWING NO: <Drawing Number>
REV: 1
SCALE: <Scale>
SHEET: 5 OF 6



REVISION RECORD			
LTR	ECO NO:	APPROVED:	DATE:

DRAWN: Nick Ames	DATED: 2015-11-13	COMPANY: OSU Robotics Club Rover 2016	
CHECKED: <Checked By>	DATED: <Checked Date>	TITLE: Uniboard	
QUALITY CONTROL: <QC By>	DATED: <QC Date>	CODE: <Code>	SIZE: B
RELEASED: <Released By>	DATED: <Release Date>	DRAWING NO: <Drawing Number>	REV: 1
		SCALE: <Scale>	SHEET: 6 OF 6