1. JTAG Control of the Threshold Voltages

The Mezzanine card contains a Xilinx CPLD configured as a JTAG slave device. This device provides a way to set the ASD threshold voltages using the standard JTAG protocol.

1.1 JTAG Instructions

Currently there are only three types of instructions the JTAG slave understands. The CPLD knows how to bypass itself, identify itself and talk to the DAC thus setting the threshold voltages. The instruction register is four bits wide so there is room to expand the functionality of this JTAG slave if needed.

Instruction	*Bit Pattern*	*Comment*
BYPASS	0000, 1111	
IDCODE *	0010	Identifies CPLD
USERCODE *	0011	Identifies firmware version
LOADDAC	0100	Puts DAC in scan-path

* Not implemented at this time

1.1.1 BYPASS

The BYPASS instruction places the single-bit BYPASS register in the scan path.

1.1.2 IDCODE

The IDCODE instruction places the 32-bit device identification register in the scan path. This register is used to identify the device as the CPLD.

1.1.3 USERCODE

The USERCODE instruction places the 32-bit firmware revision register in the scan path. This register is used to identify what version of the firmware is currently loaded in the CPLD.

1.1.4 LOADDAC

The LOADDAC instruction places the DAC serial interface in the scan path. The serial information consists of a 16-bit data word. The 16-bits are shifted through the interface as a user-defined JTAG register. This operation will set the threshold voltages.

1.2 DAC Bit Definition

The 16 DAC register bits are defined as follows:

- DAC[15:8] DAC Control Byte
- DAC[7:0] DAC Data Byte

Control Bit	*Name*	*State*	*Operation*
DAC15	UB1	X	Unassigned Bit 1
DAC14	UB2	X	Unassigned Bit 2
DAC13	UB3	X	Unassigned Bit 3
DAC12	C2	0 *	Power-Up Mode
		1	Power-Down Mode
DAC11	C1	0	DAC Register Load Operation Disabled
		1 *	DAC Register Load Operation Enabled
DAC10	C0	0	DAC Register Updated on ~CS's Rising Edge
		1 *	DAC Register Updated on ~LDAC's Falling Edge
DAC9	A1	X	Don't Care
DAC8	A0	0	Do not address DAC A
		1 *	Address DAC A

*Recommended command bit pattern.

As noted, the recommended DAC Control bit pattern is XXX0 11X1. In using this recommended bit pattern the DAC register will be updated on the falling edge of TCK in the UPDATE-DR JTAG state.

1.3 Setting the Threshold Voltages (an example)

The purpose of the DAC is to set the ASD threshold voltages. The threshold voltages are set to Vdd/2 plus or minus up to100 mV depending on the DAC data byte. The one DAC voltage controls the offset from Vdd/2 the thresholds will be set to. Threshold voltage 1 (Vth1) is set to Vdd/2 plus the offset and Vth2 is set to Vdd/2 minus the offset. When the DAC is loaded with the data byte 0x00 the offset is - 100 mV. When the DAC is loaded with the data byte 0xff the offset is + 100 mV. The offset is a linear operation with respect to the DAC data byte. Therefore, at the DAC 1/2 setting the offset = 0 and Vth1 = Vth2. As an example, if we want to set the threshold voltages Vth1 = 1.75 V and Vth2 = 1.55 V (Vdd = 3.3V) the data byte would be 0xFF. The sequence of events to set the DAC to this level would be:

1. Go to the Shift-IR state

2. Shift in the LOADDAC instruction bits (0100). The instruction is loaded in with the LSB first. The final bit is shifted in upon leaving the Shift-IR state.

3. Go to the Shift-DR state. The DAC register is now in the scan path.

4. Shift in the DAC Data Register word (XXX0 11X1 1111 1111). Although it is not the JTAG convention, the DAC forces us to shift the MSB in first. The Control byte is followed by the data byte, in this case 0xFF. The final bit (the LSB of the data byte) is shifted in upon leaving the Shift-DR state.

5. The DAC output will be updated upon the falling edge of the TCK while in the Update DR state

A detailed description of the DAC can be found in the Maxim 550A data sheet.

The mezzanine card also has a temperature sensor that is read out using the JTAG interface. As soon as this functionality has been finalized this document will be updated.