## Functional Description:

The following figure is a block diagram of the 1553 core.

Field Detect

Memory

tx\_data\_n,p

Sync Detect

rx\_data\_n,p

Manchester

Edge Detect

FSM

Counters

The core\_1553 uses an 8MHz frequency clock to sample the incoming data. The serial data input, rx\_data, is input into two shift registers, data\_sftreg, which continuously searches for edge transitions, and sync\_sftreg, which detects a sync pattern. When a valid sync pattern is detected in the 24 bit shift register, sync\_sftreg, a sync\_csw or sync\_dw flag is created. At the 8MHz clock a command and a status word is 12 ones followed by 12 zeros and a data word is 12 zeros followed by 12 ones.

After a valid sync word is detected a counter generates a data sample pulse every 8 clocks. This pulse, data\_sample, is used to sample the incoming data and also is used to keep track of the number of bits that have been received via bit\_cnt. The bit\_cnt and data\_sample are also used to register the incoming 1553 bit fields.

Additional delay is added to the incoming stream by another group of shift registers, data\_sftreg\_out. This delay allows the logic to capture the RT address that will be used to access a location in the FPGAs internal memory. The current delay is ~6 bit times. Additional counters is used to keep track of the delayed rx bits to be sent out. These counters are bitcnt and samplecnt.

In the outgoing shift register, each rx bit can be modified based on its corresponding bit field in memory. Its memory location will be specified by the registered RT address that has been captured in previous logic. If the corresponding bit field contained in memory contains a logic ‘0’, the incoming Bit is passed through. If the bit field in memory contains a logic ‘1’, the incoming Bit is inverted and passed. For example, if an incoming command word has an RT address of 13, and we want to change the T/R bit (bit 5), a logic ‘1’ would need to be written in bit 5 of memory location 13. If we want to pass what is being actually sent by the BC on subsequent command words destined for RT address 13 then bit 5 of address location 13 would need to be set to a logic ‘0’.

A signal named start\_shift is generated by the bit\_cnt and bitcnt signals when the incoming rx data has been delayed enough time to capture the RT address. The captured RT address along with the counter, bitcnt, are used to lookup the bit field in memory. If the bit field is a logic ‘1’, then the incoming bit is inverted and Manchester encoded. If the bit field is a logic ‘0’, the incoming bit is not inverted and Manchester encoded. The samplecnt is used to create the appropriate 8 clock long high to low or low to high transition.

The outgoing stream is taken from data\_sftreg\_out[12] and the encoded data is muxed back into the stream and output on txdata.

Note: Currently, the lookup table that is used to invert or pass the bits is not connected to anything and I’ve been using an input file to initialize the memory. To allow changes to memory during runtime this needs to be connected to the outside world.

## Top Level

The top level code implements two of the cores described above. One labeled BC and one labeled RT. The BC is meant to work with data from BC to RT, whereas the core labeled RT is meant to work with traffic from RT to BC. The 1553 ttl level signals RX in and TX out of the FPGA are labeled \_BC or \_RT. The pinouts for the tx and rx ttl inputs and outputs are as follows:

# inputs

NET "rxa\_p\_BC" LOC = "H15" | IOSTANDARD = LVCMOS33; # IO B19

NET "rxa\_n\_BC" LOC = "H16" | IOSTANDARD = LVCMOS33; # IO A19

NET "rxa\_p\_RT" LOC = "D14" | IOSTANDARD = LVCMOS33; # IO B28

NET "rxa\_n\_RT" LOC = "D16" | IOSTANDARD = LVCMOS33; # IO A28

# outputs

NET "txa\_p\_BC" LOC = "A2" | IOSTANDARD = LVCMOS33 | DRIVE = 12; # IO D30

NET "txa\_n\_BC" LOC = "B2" | IOSTANDARD = LVCMOS33 | DRIVE = 12; # IO D29

NET "txa\_p\_RT" LOC = "C2" | IOSTANDARD = LVCMOS33 | DRIVE = 12; # IO C27

NET "txa\_n\_RT" LOC = "C3" | IOSTANDARD = LVCMOS33 | DRIVE = 12; # IO D27

A bypass bit is included which enables the core to pass the incoming rx data through without changing the bit fields. The bypass bit for the BC->RT is connected to switch 7 and RT->BC is connected to switch 8.

Additional pinouts can be found in the top\_1553.ucf file in the core\_1553 directory.

Theoretically, as many cores as needed could be implemented inside the FPGA with the only limitation being available block RAM and slice utilization. As of now there is more than enough space.