On XAUI Synchronisation with CASPER Tools

Richard Armstrong

University of Oxford

1 The Issue

The design philosophy of the (recently re-acronymed) *Collaboration* for Astronomy Signal Processing Research (CASPER)[1] advocates a Globally Asynchronous, locally Synchronous (GALS) architecture for hardware systems. Certainly, there are many advantages to this design mode, including the possibility of a robust system that is dynamically responsive to small, non-systematic errors and the avoidance of strict system-wide synchronisation requirements.

However, there are situations when one requires a synchronous, multiple-board system. Many examples of these systems exist[2]. In such synchronous systems, high-speed, multi-gigabit (XAUI) links provide high-data-rate connectivity between boards, and in many cases these links must be synchronised with each other. This is a non-trivial task, as data are not guaranteed to arrive in alignment.

There are three classes of solutions within the CASPER framework to synchronise two or more XAUI streams from either multiple cores on the same board or multiple cores on multiple boards. Each of these solutions is a trade-off between available bandwidth and dynamic error resilience. Your application will dictate the level required.

2 Error Model

A first step in an engineering characterisation of a system is to define a representative error model for the system. One may define a model of these high-speed links which would contain a set of errors due to the following individual conditions:

- An asynchronous, nominally 156.25MHz local transmit clock is integrated onto each Multi-Gigabit Transceiver (Xilinx 'RocketIO') Core (see the Xilinx RocketIO Userguide for more information). Phase offsets between these local clocks produces errors in a range with mean 0 +-1 clock cycles between adjacent streams

[1] see the CASPER website at http://www.casper.berkeley.edu

[2] Some examples of instruments that require synchronous digital systems are the Sub-Millimetre Array (SMA), Green-Bank Ultimate Pulsar Processing Instrument (GUPPI) at NRAO, and the 2-Polarisation, All-Digital Aperture Array (2PAD)

- Temperature of the XAUI core seems to be the cause of the largest and most unpredictable errors. If FPGAs are allowed to get too hot, large, unpredictable errors are observed to occur, and are very difficult, if not impossible to correct for. The effect has been most often noted when running close-to-full FPGA designs. Certainly, all FPGAs should be *actively* cooled[3] at a minimum. If large or unusual errors are observed, we recommend more aggressive cooling as the first debug step.

- Incorrect initial synchronisation of iADC and IBOB boards due to weak clock sources or weak global synchronisation pulses (used in regular pulse synchronisation schemes, such as a 1 pulse-per-second scheme). Perhaps it should not be mentioned, but it is worth checking very carefully that all boards are receiving the correct signal levels and all SMA connections are connected with the correct torque.

- Elastic receive buffers in the Muti-Gigabit Transceiver Core compensate for the 8b10b codec operation. NRAO report error rates of up to +-4 clock cycles due to this buffering operation.

3 Synchronisation Approaches

Synchronisation is the process of aligning the XAUI streams in a receive FPGA, and compensating for alignment errors. In all synchronisation approaches, we assume that the system designer has included a scheme to synchronise all first-level boards, perhaps with a global, pulse-based synchronisation method. A 'best-practice' solution for this task is the 1PPS synchronisation scheme[4] However, we *do* assume that the boards are synchronised to begin with.

We do not either use the out-of-band (OOB) signals for synchronisation, since these have been observed to arrive out-of-step with simultaneously transmitted data.

3.1 Startup Synchronisation

The first and simplest synchronisation method assumes a constant error model. Streams are synchronised during datapath initialisation once only. If de-synchronisation occurs, the system must be reset. All data are lost after de-synchronisation. This system could be implemented automatically on the first reception of a sync pulse with 'first-in first-out' (FIFO) buffers or configured by hand using shift registers.

[3] where active cooling usually means either a fan placed directly on each FPGA heat-sink, or a rack-based fan system that will force air through hardware racks, or both

[4] A thorough description of this pulse-based synchronisation system is outside the central scope of this document. It is dealt with in detail in the 'Sync-Pulse Memo,' available on the CASPER website.

3.2 Pulse-based Synchronisation

The second method uses a single-bit system-wide 'sync-pulse' to align streams at the start of each 'frame' of data. This uses a single bit of the XAUI data word[5]. The use of the extra bit of data bandwidth is often not a serious issue since one may already be sending the sync for downstream 'start-of-frame' tagging. This method has the advantage of being able to dynamically recover from desynchronisation events. An issue is that de-synchronisation causes at least 1 frame of incorrect data. A reference design for such a system is shown in Figure 1.

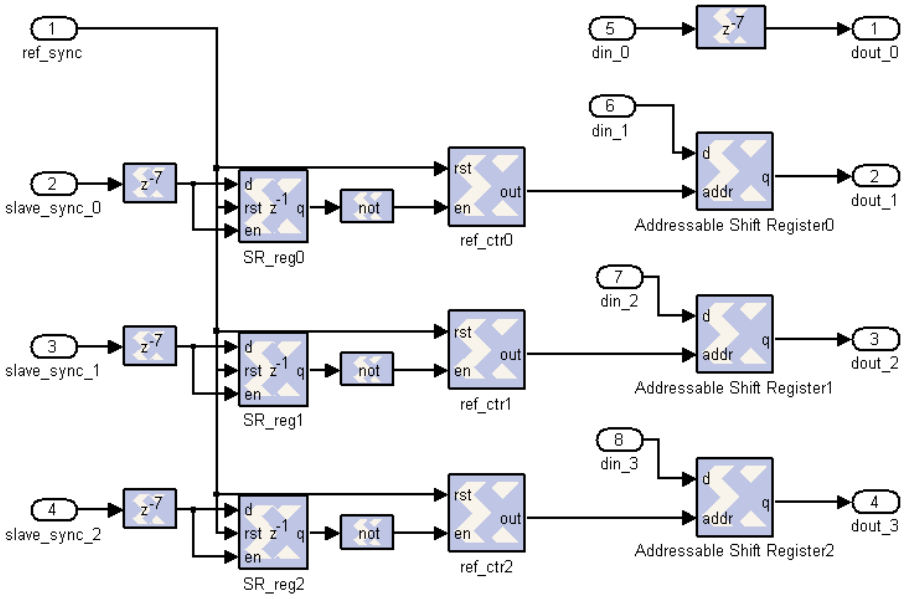

Figure 1: A reference design for a sync-pulse-based synchronisation scheme. A reference sync pulse starts a counter for each other stream. The other sync pulses are delayed by an amount that is at least as long as the maximum expected error and these stop the counter when they finally arrive. This counter amount then forms the address of an addressable shift register through which the data are delayed. The reference data are finally delayed by the same amount as the other streams.

3.3 Tag-based Synchronisation

The third method uses an extra 3-4 bits of the data width, and provides precise synchronisation on each data packet sent, without data loss in de-

[5]which may be 32- or 64-bits, depending on mode of operation

synchronisation events. A 3- or 4-bit clock is sent with the data word, which allows each stream to be precisely aligned on the receive FPGA.

Due to its guaranteed error resilience, this is the preferred solution, and is the solution implemented for the 2PAD instrument. Engineers working on the GUPPI project will soon release a detailed description of synchronisation in their system which includes this method. A reference design for this type of system is shown in Figure 2.

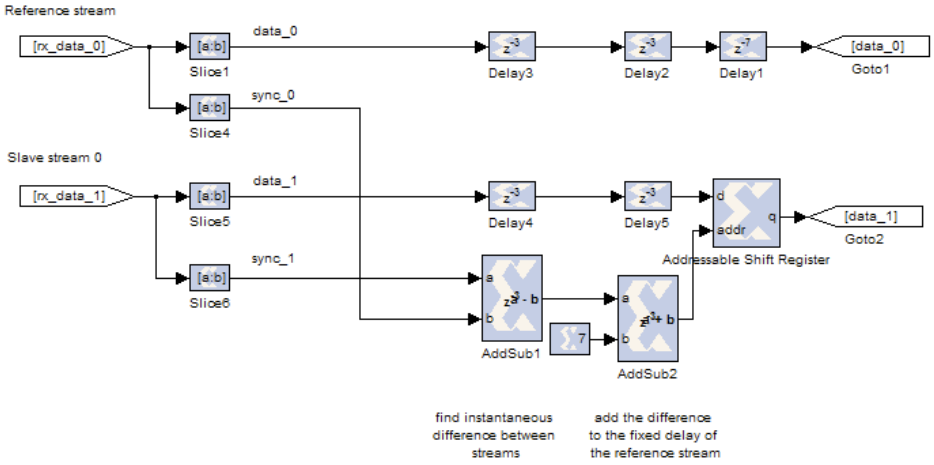

Figure 2: A reference design for a tag-based synchronisation scheme.