ACQ4xx Streaming Data Formats

Prepared By: Peter Milne

Date: 27 March 2014

Rev	***Date***	***Description***

Table of Contents

1 Generic Frame Word

A single 32 bit tag word is appended to each sample. Multiple tag words make up a frame. Currently, a 4-sample frame is defined:

Function	***SC***	***META2***	***META1***	***DI4***	***FrameID***
BITS	d31:d24	d23:d16	d16:d8	d7:d4	d3:d0
MASK	0xff	0xff	0xff	0xf	0xf
LABEL	SCaa	SW2aa	SW1aa	DI4 3:0	0x1
	SCbb	SW2bb	SW1bb	DI4 3:0	0x2
	SCcc	SW2cc	SW1cc	DI4 3:0	0x3
	SCdd	SC2dd	SW1dd	DI4 3:0	0x4
					0x5..0xf undefined

- FrameID : 4 bits, valid 0, 1, 2, 3 : identifies part of frame
- DI4 : 4 bits, DI4 data, full rate.
- META1: 32 bits, 8 bits per sample, sequence 0xaabbccdd
- META2: 32 bits, 8 bits per sample, sequence 0xaabbccdd
- SC : 32 bits, Sample Count, 8 bits per sample, sequence 0xaabbccdd

1.1 Modes for Meta:

MODE	***META1***	***META2***	***Description***
MODE1	SW REG 1	SW REG 2	Software inserted values
MODE2	LATCH COUNT	SW REG 2	Event Latch / SW value
MODE3	DI15-DI08	DI07:DI00	DIO input word (when fitted)
MODE4	ENCODER1	ENCODER2	Shaft encoder input

NB: there's no indication of MODE in the stream. Application software is responsible for setting mode before the stream starts. It's anticipated that STREAM_DATA_MODE will be a 2 bit field in a Site 0 reg.

NB: DI32 data? Not supported in the GFW. If full rate, full width DI data required, then then should be as part of a site-specific FIFO. That's more expandable, as it will then support multiple DI32 ports

1.2 Data Rates:

Example: ACQ420FMC-4-2000 : 4 x 2 bytes + 4 bytes tag = 12 bytes/sample.

Running at 2MSPS, data rate is 24MBytes/sec.

2 ACQ435, 32 Channel

2.1 Basic

This is the default, with channel ID in lower 8, d{7-5} : site, d{4-0} : channel

d31-d08	***d07-d00***
CH00 data 0xaabbcc	0x00
CH01 data 0xaabbcc	0x01
...	...
CH31 data 0xaabbcc	0x1f

2.2 Scratchpad

where CH25-32 are replaced by 8 32 bit values, [0] is sample count and if updated by the FPGA, [1..7] are updated from scratchpad regs. This is effective for the 24ch case and is really a special option (only when enabled).

d31-d08	***d07-d00***
CH00 data 0xaabbcc	0x00
CH01 data 0xaabbcc	0x01
...	...
CH24 data 0xaabbcc	0x17
SAMPLE COUNT 0xaabbccdd	
META1 0xaabbccdd	
...	
META7 0xaabbccdd	

2.3 Embedded bits

Extends the Basic case where a special pattern replaces the top 3 bits, could be an extra mode, or, better could simply “just happen” all the time. The embedded bits are viewed as 32 bit slices in a frame of 32 samples, where

- d7 is the Sample Count SC, a 32 bit word with d0 in the CH00 position, and d31 in the CH31 position. This is updated automatically by the FPGA at the time of sample.
- d6 is the sample count latched at time of PPS, also 32 bit in the same order.
- d5 is the contents of a “Software Embedded Word” SEW register

d31-D08	***d07***	***d06***	***d05***	***d04-d00***
24 bit adc data	***SC***	***PPS***	***SEW***	***CH ID***
CH00 data 0xaabbcc	SC.d0	PPS.d0	SEW.d0	0x00
CH01 data 0xaabbcc	SC.d1	PPS.d1	SEW.d1	0x01
CH02 data 0xaabbcc	SC.d2	PPS.d2	SEW.d2	0x02
CH03 data 0xaabbcc	SC.d3	PPS.d3	SEW.d3	0x03
…..				
CH30 data 0xaabbcc	SC.d30	PPS.d30	SEW.d30	0x1e
CH31 data 0xaabbcc	SC.d31	PPS.d31	SEW.d31	0x1f

- Software would likely handle the SEW as follows:
  d{31-25} : sequence number
  d{24-0} : embedded data.
- Each time SW updates the SEW, it bumps the sequence number. It's up to software to ensure that the SEW is not updated before the info has been sent.
- At eg 32kHz sample rate, this implies software mustn't write to the SEW more than once per 32 samples, eg at >1kHz.
- We conclude that overwrite isn't likely to be a problem.
- The decoder scans the SEW for the next sequence number, this indicates new data; other data with the same sequence number can be ignored.