FRAME

Captured frames are read out over usb in muliples of 1024 bytes. Each 'frame' of data contains a HEADER, CONFIG, FRAME DATA, and enought FILLER data to fill the last USB packet. The size of the FRAME DATA is given by FRAMESIZE contained in the HEADER. Since the usb transfers are send in multiples of 1024 bytes the MAGIC fields in the HEADER can be used by drivers and applications to synchronize USB reads to the HEADER and subsequent FRAME DATA. FILLER is used to padd the FRAMEDATA to the next 1024 byte boundary. All data is transfered in **BIG ENDIAN** format.

Basic Frame Layout

Word idx	# of bytes	
0	20	**HEADER**
10	108	**HEADER PADDING**
64	128	**CONFIG DATA**
128	768	**CONFIG FILLER**
512	N	**FRAME DATA**
	-	**FRAME FILLER**

HEADER

#0 / #1 MAGIC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MAGICH															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAGICL															

MAGICH Slways set to 0xDDDD
MAGICL Slways set to 0xDDDD

#2 / #3 TEMP

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
?															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
?															

? * there should be temperature encoded here somehow. Todo- fugure out how temp is coded
? * for temperature monitoring...

#4 / #5 ETS DELAY

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
?															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
?															

? ETS delay
?

#6 / #7 ETS MIN

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
?															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
?															

? ETS min
?

#8 / #9 ETS MAX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
?															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
?															

? ETS max
?

#10 - #127 HEADER PADDING

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-															

These bytes are reserved

CONFIG DATA

The configuration data is sent in the header of every frame. It gives context for the FRAME DATA

#0 ADC_control_reg_addr

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRA[0:15]															

CRA[0:15] This value should always be set to 0x000D. No idea why...

#1 ADC_control_reg_data

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								**CRD[0:7]**							

CRD[0:7] Allows setting the mode. No idea what the different modes do though... you probably just want to set it to normal

BIN	HEX	Mode
00000000	00	NORMAL
00000010	02	TEST

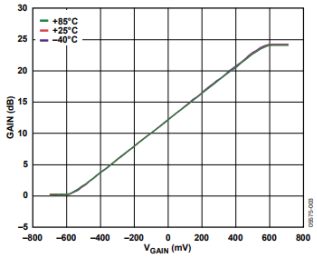
#2 / #3 **VGAINA / VGAINB**

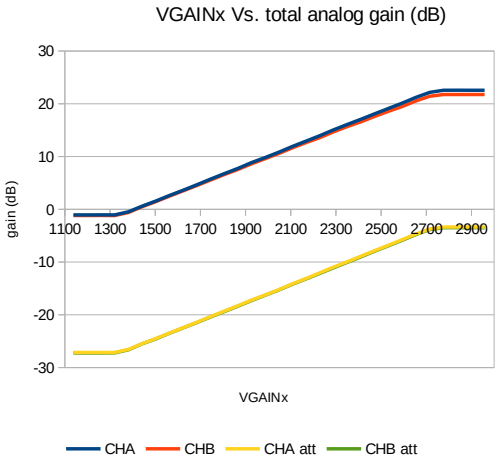
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-				**VGAINx[0:11]**											

VGAINx[0:11] VGAINx is used to configure the desired gain of the analog input stage for each channel. Internally a voltage between -1.8 and 1.8 volts is produced which corresponds to a VGAIN value of 0 and 4095 respectively. This voltage is fed to the variable gain amplifier wich in turn provides an analog gain between approximately 0db and 24db. The VGA itself is referenced to 0v at approximately 12db. This in combination with the ATT bit provides the total attenuation/amplification per channel. These values and graphs are approximate. Callibration is required.

VGAINx	HEX	GAIN (DB)
0	0000	0
…	…	…
1365	0555	0
3413	0D55	24
…	…	…
4095	0FFF	24

*aproximate





#4 / #5 **OFFSETA / OFFSETB**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-				**OFFSETx[0:15]**											

OFFSETx[0:15] Ground reference for measurments. This is a 12 bit twos complement signed value in the range of -2048 to 2047. The Ground reference is produced by a DAC and fed to the Vcom of the differential input amplifier. Positive values move the sampled values down. Negative values move samples values up.

#6 **CTRL**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								**ETS**	**ADCINT**	**ACA**	**ACB**	**CNGA**	**GNDB**	**ATTA**	**ATTB**

ETS ??
ADCINT Set to 1 to connect Channel 1 to both ADC inputs (and disconnect the analog input channel 2)
ACA Set to 1 to enable DC coupling
ACB Set to 1 to enable DC coupling
CNGA Set to 1 to ground channel A
GNDB Set to 1 to ground channel B
ATTA Set to 1 to attenuate channel A
ATTB Set to 1 to attenuate channel B

ATTx		Rt		Rb	Vgain	db
1	R10	47,00	R9	953,00	0,0470	-26,5580
0	R16	953,00	R15	49,90	0,9502	-0,4433

* approximate

#7 **TRIGGER_MODE**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-													**REARM**	**TMOD1**	

TMOD[0:1] Trigger mode selection bits

BIN	HEX	Mode	
00	0000	Auto	Automatically switch between Normal and continuous mode
01	0001	Normal	A capture is made whenever the trigger event occurs
10	0002	Single	The capture is only triggered once.
11	0003	Continuous	Capture is triggered immediatly after the previous one. Other trigger settings are ignored

REARM Rearms the trigger system. Set this bit to 1 to rearm the trigger in single mode.

#8 **TRIGGER_SOURCE**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														**TSRC**	

TSRC[0:2] Selects the source of the trigger event

BIN	HEX	Source
000	0000	CH-A
001	0001	CH-B
010	0002	AWG-1
011	0003	AWG-2
100	0004	External
xxx	xxxx	Reserved

#9 **TRIGGER_SLOPE**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														**TSLP**	

TSLP[0:1] Selects the slope of the trigger event

BIN	HEX	Direction
00	0000	Rising
01	0001	Falling
10	0002	Both
xx	xxxx	Reserved

#10 **TRIGGER LEVEL**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-					**TLVL**										

TLVL[0:10]
Sets the trigger level for the trigger event. A trigger event will occur in the associated trigger mode when the sampled value passes the trigger level

#11 **TRIGGERHYST**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-					**THYST**										

THYST[0:10] Sets the hysteresis for the trigger event. This value can be between 0 and TLVL. The lower value of the trigger is given by TLVL-THYST. The upper value is given by TLVL. Min 0. Max 1023-TLVL

#12 **PRETRIGGER**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TPRE															

TPRE[0:15] Sets the number of samples before the trigger in steps of 1024

#13 **TIMEBASE**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-											**TBASE**				

TBASE[0:4] Sets the timebase. This is effectively the sample rate

BIN	HEX	Timebase
00000	0000	2ns
00001	0001	4ns
00010	0002	8ns
00011	0003	20ns
00100	0004	40ns
00101	0005	80ns
00110	0006	200ns
00111	0007	400ns
01000	0008	800ns
01001	0009	2us
01010	000A	4us
01011	000B	8us
01100	000C	20us
01101	000D	40us
01110	000E	80us
01111	000F	200us
10000	0010	400us
10001	0011	800us
10010	0012	2ms
10011	0013	4ms
10100	0014	8ms
10101	0015	20ms
xxxxx	xxxx	---
11111	001F	4ns-ETS

#14 / #15 **HOLDOFF**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HOLDOFFH															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HOLDOFFL															

Have yet to figgure out exactly what this means... i think it is a blanking of the trigger. But i don't know if it is before or after or … whatever

#16 / #17 FRAMESIZE

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FRAMESIZE															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRAMESIZE															

FRAMESIZE Gives the number of samples in the body of the frame. The size of the SAMPLE DATA in bytes is FRAMESIZE*4 rounded up to the next multiple of 1024

#18 AWG1-CNF0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-							**AWG1EN**	-				**AGW1TYPE**			

AWG1EN Set this bit to 1 to enable the arbitrary waveform generator

AGW1TYPE Select the type of waveform

BIN	HEX	Mode
0000	0000	reserved
0001	0001	custom
0010	0002	sin
0011	0003	cos
0100	0004	triangle
0101	0005	saw
0110	0006	square
0111	0007	delta
1000	0008	dc*
1001	0009	noise
xxxx	xxxx	reserved

*To use the generator in dc mode. The value must be set to 0 and the offset used to set the desired output voltage

#19 AWG1-CNF1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
?	?	?	?	**AWG1VOL**											

AWG1VOL arbitrary waveform generator amplitude. Unsigned

#20 AWG1-OFFSET

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-				**AWG1OFFSET**											

AWG1OFFSET arbitrary waveform generator offset. Twos complement. Signed

R = 100
IFS = 32 * 1,25 * 4700
Vo1 = R * IFS * AWG1OFFSET / 4096
Vo1 = R * IFS * (4096 – AWG1OFFSET) / 4096
Vout = 2,4 * (Vo2 -Vo1)

#21 / #22 AWG1-DELTA

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DELTAH															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DELTAL															

Have yet to figure out exactly how this works.

#23 AWG1-DUTY

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-					**AWG1DUTY**										

AWG1DUTY Duty cycle of the square wave in steps of 1/2048 %

#24 - #29

AWG2 Identical to AWG1

#30 / #31 DIGITAL PATTERN 1

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-															

Have not documented this yet

#32 DIGITAL MASK 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-															

Have not documented this yet

#33 / #34 DIGITAL PATTERN 2

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-															

Have not documented this yet

#35 DIGITAL MASK 2

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-															

Have not documented this yet

#36 / #37 DIGITAL PATTERN 3

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-															

Have not documented this yet

#38 DIGITAL MASK 3

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-															

Have not documented this yet

#39 / #40 DIGITAL PATTERN 4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-															

Have not documented this yet

#41 DIGITAL MASK 4

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-															

Have not documented this yet

#42 DELAY MAX COUNT 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-															

Have not documented this yet

#43 DELAY MAX COUNT 2

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-															

Have not documented this yet

#44 DELAY MAX COUNT 3

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-															

Have not documented this yet

#45 DELAY MAX COUNT 4

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-															

Have not documented this yet

#46 dt_StageStart; dtSerial; dtSerialCh stage at which capture starts! (0, 1, 2, 3); parallel/serial mode; seria dt_StageStart=bit9-bit8; dtSerial=bit4; dtSerialCh=bit3-bit0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-						STAGE START					SERIAL	CHANNEL			

Have not documented this yet

#47 DIGITAL IO VOLTAGE

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								**WIPER**							

WIPER This sets the digital voltage for both inputs and outputs. put formula here for approximating output voltage of the vcc_dig DAC
VCC_DIG = …. Volts

#48 DIGITAL DIRECTION

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														**DIR1**	**DIR0**

DIR0 Not sure but most likely 1 means output and 0 means input. DIR0 sets ios 0-5 and DIR1 sets io 6-11... probably
DIR1

#49 **DIGITAL OUTPUT WORD**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-															

?

#50 **DIGITAL OUTPUT MASK**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-															

?

#51/#52 **DIGITAL SAMPLING CLOCK PRESCALER**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRESCALERH															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRESCALERL															

#53 **ADC SAMPLE AVERAGING**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-						**CHA**	**CHB**	-							

CHA Set to 1 to enable moving average of channel A How big is the window?
CHB Set to 1 to enable moving average of channel B

#54 - #63

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-															

These bytes are reserved.

#64 - #127 **#192**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CFILLERH															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CFILLERL															

CFILLERH Always set to 0x0000
CFILLERL Always set to 0xFFFF

SAMPLE DATA

The Sample data is a block of FRAMESIZE samples. Each sample contains the A channel, B channel, and digital values encoded in two 16bit words.

#0 to FRAMESIZE*2

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SAMPLE A										**SAMPLE B**					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SAMPLE B				**D11**	**D10**	**D9**	**D8**	**D7**	**D6**	**D5**	**D4**	**D3**	**D2**	**D1**	**D0**

SAMPLE A Channel A sample value
SAMPLE B Channel B sample value
D0-D11 Digital input sample value

FILLER

USB transfers are always made in chunks of 1024 bytes. So the last chunk in the SAMPLE data is padded out with FILLER data

 # 0 to next boudary of 1024 bytes

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FFILLERH															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FFILLERL															

FFILLERH Always 0x0000
FFILLERL Always 0x0000