File Format - Raw Image Acquisition File (DAT) (v1.3.0)



Created by Gene Hartsell on Mar 4, 2011 9:10 PM. Last modified by Susan Azad on Jun 25, 2012 5:56 PM.

This page last changed on Mar 22, 2011

wall time

Torrent Dev Ho

Raw Image Acquisition File (DAT) Format

Experiment Files Conte File There are several File names present in each experiment: Forma prerun xxxx.dat 1.wells files Images taken before the experiment begins. (WELI beadfind_pre_xxxx.dat File Forma PH step images used to find live beads. Alignn acq_xxxx.dat **Files** (SAM) Actual sequencing images. File beadfind post xxxx.dat Forma bfmas PH step images used to find washout beads. files File **PGM Acquisition Data Format** Forma Raw Acquisition data coming off of the PGM[™] has the following format: **Image** Field Description Datatype Acquis Signature signature unsigned File int32 (DAT) Version File version unsigned int32 Forma Run L unsigned header size Size in bytes of the header **Files** int32 File data size unsigned Size in bytes of the data portion. Forma int32 Seque

Time of acquisition.

unsigned

int32

1 of 7

Files

(SFF and

Field Datatype Description unsigned Number of rows in the following images. rows int16 Number of columns in the following images. cols unsigned int16 channels unsigned Number of channels in the imaged chip. int16 interlacetype unsigned Interlace type: int16 0 = uninterlaced. 4 = compressed. frames in file unsigned Number of frames to follow this header. int16 Uncompressed frames in file unsigned Number of frames at the base frame rate. int16 sample rate (MHz) unsigned Acquisition speed at which the image was int32 taken. full scale voltage [TS:4 DACs] unsigned Max voltage for the channel A/Ds (not (mV) int16 populated). channel offset [TS:4 DACs] unsigned Current voltage for the channel A/Ds. int16 ref electrode offset unsigned Voltage of the fluid flowing over the chip. int16 frame interval unsigned Time interval between frames. int16

FAST(File Forma FAST/ File

The raw data are in big-endian format, so 2- and 4-byte values must be swapped on X86-based processors. The data are also row-major.

For each frame, the chip data are contained in the lower 14 bits, so typically mask (AND) with 0x3fff to get the raw counts. Also, there are always reference rows and columns on chip data; they are a 4 pixel-wide border around the chip. The reference pixels are tied to alternating VREF1 and VREF2.

All image files coming from the PGM should be of interlace type 4 (compressed). The compression used is simple:

Frame 1 is un-compressed and un-interlaced.

Field	Datatype	Description
timestamp (ms)	unsigned int32	Relative time from the start of acquisition to the end of this frames sampling period.
Compressed (0)	unsigned int32	If this is a previous frame subtracted frame or not. If not compressed, the data follows immediately.

Field	Datatype	Description
image data [TS:rows	unsigned	Frame data.
x cols]	int16	

Frame 2 and beyond can be compressed, but is always un-interlaced.

Field	Datatype	Description	
timestamp (ms)	unsigned int32	Relative time from the start of acquisition to the end of this frames sampling period.	
Compressed (1)	unsigned int32	If this is a previous frame subtracted frame or not. If not compressed, the data follows immediately.	
len	unsigned int32	Length of the compressed frame.	
Transitions	unsigned int32	Number of transitions from 8-bit values to 16-bit values.	
total	unsigned int32	Sum of all the pixel values after previous frame subtraction.	
sentinel	unsigned int32	Always 0xDEADBEEF.	
image data(variable length)	unsigned int8	Frame data.	

Working with the Files

The correct way to manipulate these files is by including deInterlace.cpp from the Torrent Suite analysis software, which contains the following function for parsing acquisition files:

```
void *deInterlace(
char *fname, // File name to load
void *output, // output buffer pointer. If NULL, one is allocated.
int start_frame, // first frame to return Set to 0 for all frames
int end_frame, // last frame to return Set to 0 for all frames
int mincols, // first column to return Set to 0 for all cols
int minrows, // first row to return Set to 0 for all rows
int maxcols, // last column to return Set to 0 for all columns
int maxrows); // last row to return Set to 0 for all rows
```

Transitions are handled by adding special keys:

Key	Value
KEY_8	0x4499
KEY_16	0x44BB

Example Code for Handling Compression

This code is provided for instructional purposes only. You should use the declarations in datacollect/src/experiment.h for actual parsing of DAT files.

```
int PrevFrameSubtract(int elems, int16_t *framePtr,
                       int16_t *prevFramePtr, int16_t *results)
                       int PrevFrameSubtract(int elems,
                       int16_t *framePtr, int16_t *prevFramePtr,
                       int16_t *results)
{
          int i;
          register int16_t *src1=framePtr;
          register int16_t *src2=prevFramePtr;
          int16_t dst;
          int8_t dst8;
          int state = 0;
          int8\_t *resp = ((int8\_t *)results) + 12;
          uint32_t *lenp = (uint32_t *)results;
          uint32_t len,Transitions;
          register uint32_t total=0;
          uint32_t value;
          Transitions = 0;
          int failed=0;
          int8_t * limit = ((int8_t *) results) + elems*2 - 2000;
          *(unsigned int *)resp = PLACEKEY;
          resp += 4; // add key before each frame
          for(i=0;i<elems;i++)</pre>
                     if (resp > limit)
                     {
                               failed = 1;
                               break;
                     }
                    total += *src1;
                    value = *src1;
                               dst = *src1++ - *src2++;
                               if(dst >= 128 \mid \mid dst <= -128)
                               {
                                          // 16-bit mode
                                         if (state != 16 ||
                                      ((dst >> 8) == KEY_0))
                                          {
                                                    state = 16;
```

```
*resp++ = KEY_0;
                                                    *resp++ = KEY_16_1;
                                                    Transitions++;
                                          }
                                          *resp++ = ((dst & 0xff00) >> 8);
                                          *resp++ = (dst & 0xff);
                               }
                               else
                               {
                                          // 8-bit mode
                                          if((state != 8) || (dst == KEY_0))
                                                    state = 8;
                                                    *resp++ = KEY_0;
                                                    *resp++ = KEY_8_1;
                                                    Transitions++;
                                          }
                                          dst8 = (int8_t)dst;
                                          *resp++ = dst8;
                               }
          len = (uint32_t)((uint32_t)resp - (uint32_t)results);
          len+= 3;
          len &= \sim 0x3; // quad-word allign it
          *lenp++ = len;
          *lenp++ = Transitions;
          *lenp++ = total;
          return failed;
}
```

© 2011 Life Technologies Corporation. All rights reserved.

The trademarks mentioned herein are the property of Life Technologies Corporation or their respective owners.

For Research Use Only. Not intended for any animal or human therapeutic or diagnostic use.

303 Views Categories: Tags:

Average User Rating

Your Rating:

(0 ratings)

3 Comments 0 Author comments



Pablo Cingolani May 5, 2011 12:59 AM

WARNING: There is a mistake in this documentation.

In uncompressed frame, the field "Compressed int32" does not exists.

Actions

☆ Helpful |

Like (0)



Pablo Cingolani May 5, 2011 3:00 AM

Looks like none of the DAT files version 3 are compressd. Is this correct?

Actions

☆ Helpful | Like (0)



Monkol Lek May 5, 2011 5:33 PM

I have brought this up a few weeks ago. See the comments:

http://lifetech-it.hosted.jivesoftware.com/docs/DOC-1395

Actions

☆ Helpful |

☆ Like (0)

Ion Community

Applications	Datasets	Protocols	Products	Programs & Events
Targeted Sequencing		Ion S5 Systems	Ion S5 Systems	Training & Ion Academy
Exome Sequencing		Ion Chef System	Ion Proton System	Chip Recycling
Transcriptome		Ion PGM System	Ion PGM System	
Sequencing		Ion Proton System	Ion Chef System	
Microbial Sequencing			Ion AmpliSeq Technolog	Jy
Additional Applications				
			Software	
			Ion AmpliSeq Designer	
			Torrent Suite	
			Torrent Variant Caller	
			Torrent Browser Plugin	
			Store	
			Ion Reporter	
			Product Feedback	

For Research Use Only. Not for use in diagnostic procedures.

Terms and Conditions

© 2015 Thermo Scientific Inc. All rights reserved. All trademarks are the property of Thermo Fisher Scientific and its subsidiaries unless otherwise specified.