JAGA16 Data format v.3

Dec 26, 2016

1. Layman's version

Each packet comes with a header related to data transmission info followed by actual neural data. The header has 10 values (each value has 2bytes so total 20 bytes of header info values). The header includes timestamps, version number, channel number, sampling rate, etc. Neural data (Data block) is-2 bytes of data per channel x number of channels, repeated N times, where N=43 for 16 channels, 86 for 8 channels, 125 for 4 channels, 250 for 2 channels, 500 for 1 channel.

1. In order to see the header info in the file 'year-date-time-jaga_samples_(py).dat' in MATLAB, you can type the below.

```
infile = '2016-10-27_08-05-50_jaga.dat';
fid = fopen(infile, 'r');
timestamp = fread(fid, [1], 'double', 'ieee-le')
date = datestr(timestamp / 86400 + datenum(1970,1,1)) % MATLAB native timestamp
version = fread(fid, [1], 'uint8')
channels = fread(fid, [1], 'uint8')
diagnostic_word = fread(fid, [1], 'uint16')
mode_word = fread(fid, [1], 'uint16')
sampling_rate =fread(fid, [1], 'uint16', 'ieee-le')
sample_count = fread(fid, [1], 'uint32', 'ieee-le')
samples = fread(fid, [32], 'uint16','ieee-le')  % Read the first 32 samples
fclose(fid);
```

2. The below bulleted list shows an example of the numeric values in the header data structure.

- 1478057491.223793 (timestamp, 8 byte IEEE little-endian double, seconds since UNIX epoch)
- In the code above, we also produce the value 'date' which converts this floating point value to a MATLAB-native timestamp
- 3 (format version) – 1 byte
- 16 (No. of Channels) – 1 byte
- 43 (Diagnostic word) – 2 bytes uint16 little-endian (see below for decoding). In this example, this indicates that 43 samples were in the backlog on the JAGA16 device, which is fairly normal (1 packet in the buffer backlog).

- 12299 (Mode word) – 2 bytes uint16 little-endian. In this example, bit 13==1 (Diagnostic word is backlog in sample count), and bit12==1 (Bottom 8 bits of Mode word indicates lost packets since previous report). Bottom 8 bits == 11 indicating 11 lost packets since last report.
- 1000 (Sampling Rate=Samples/sec) – 2 bytes uint16 little-endian
- 1,742,489 (uint32 little-endian, Elapsed samples.)

After this header, Data block starts. The data block consists of N sample sets (where each sample set is 2 bytes x 16 channels)

- **Sample set:** 56049, ..., value (2 byte uint16 little-endian for each value). 56049 is the value for channel 1, followed by values for channels 2, ..16
- This sample set repeats N times

The number of sample sets in the data block depends on the number of channels, as follows

- N=43 when 16 channels
- N=125 when 4 channels
- N=250 when 2 channels
- N=500 when 1 channel

After the data blocks, there is a TTL block (if TTL is enabled, otherwise this block does not exist)

After the data blocks (and optional TTL block), the next packet is recorded, starting with the header as above.

3.The output below shows the hexadecimal representation of the same example data shown in 2., that are written in numeric values. You can obtain this hexadecimal output on a Mac OS/Linux command line by typing `$ xxd filename`

We have highlighted particular components of the data stream in the same colors as shown in the numerical representation in 2. above.

```
Byte Offset: Hexadecimal values
16 byte location
0000000: a052 ce84 5706 d641 0310 2b00 0b30 e803
0000010: 9996 1a00 f1da ffc5 14db 9fd4 36da 70c4
0000020: 96d8 b2ce 7add cbcc 7fd0 88db e1cc ccd6
0000030: e9c0 8bc1 35d0 8bc0 adce d1cb 8ccd dcbe
0000040: e6cb 02c8 21d2 cac6 dbc4 e3d2 7dc0 b3cd
0000050: 08ad 7eaf 26c1 e7b6 3cbc 72bf 64bb 93b6
0000060: 05b9 13be 24c0 0fbd 5bb2 f1c6 5eac f8c1
0000070: 6f90 3796 59af 14ab f3a5 a1b0 81a5 d5ac
0000080: e2a1 61b2 74a9 d9b1 b49b 3bb9 1d94 c7b4
```

2. Engineering Version

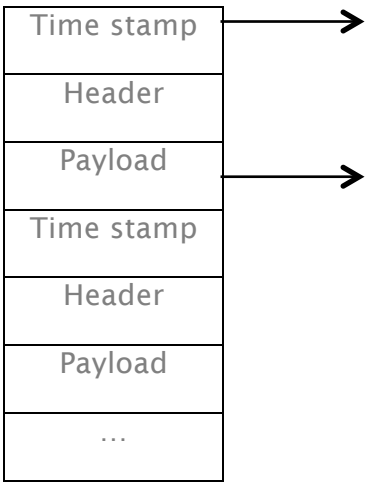
The JAGA device produces data asa stream of UDP packets, and sends it to a receiving computer. The receiving computer then stores that data to disk, with some additional timestamp information per received packet.

The timestamp must be generated by the capturing software, based upon time of receipt of the packet, with some logic for removing packet jitter. The packet sample counter will be far more linear than the network packet receive time, so the counter should be used to linearly extrapolate the receive times to a more stable start time for the capture. This logic is implemented in the Python software provided with the JAGA16 device, so custom software implementations can use this as a reference implementation. The timestamps recorded in the file can then be used to synchronize the captured data with other data sources *captured on the same computer* (since clocks can vary significantly between computers). For longer continuous capture times, the linear extrapolation should be repeated periodically depending on the system clock drift rate, since the system clock and JAGA16 device's clock ticks will drift apart over time. The JAGA16 device's clock accuracy is rated at ±20ppm, so you could expect a drift of approximately ±20 milliseconds per 1000 seconds, or ±72ms per hour of capture time.

We here document the on-disk format of the data, to permit further decoding and processing by third-party software.

2.1.Description

The data file is organized as follows:



The specific data is:

- **Timestamp** (8 bytes): IEEE little-endian double-precision floating point number, indicating the number of seconds since 1970-01-01 00:00:00 UTC (Unix Epoch), to microsecond precision.
- **Header** (12 bytes):
    - **Format** (2 bytes, unsigned 16-bit integer, little-endian): the data format in this file, currently always = 3. Future file formats will use a different version number.

    - **Channels** (1 bytes, unsigned 8-bit integer): the number of channels being captured, should be one of 16, 8, 4, 2 or 1.
    - **Diagnostic word** (2 bytes, unsigned 16-bit integer, little-endian): Meaning depends on the value of the Mode word, see Section 2.2 Data structures for details.
    - **Mode word** (2 bytes, unsigned 16-bit integer, little-endian): Bitwise flags indicating various modes and the meaning of the Diagnostic word. See Section 2.2. Data structures for details.
    - **Samples per second** (2 bytes, unsigned 16-bit integer, little-endian): the number of sample sets (there is 1 sample per channel in a sample set) captured in 1 second. This rate varies depending on the specific configuration of your device. You should use this value as the sampling rate in FFT functions and the like.
    - **Elapsed samples** (4 bytes, unsigned 32-bit integer, little-endian): the number of complete seconds that have elapsed since the beginning of capture (when the JAGA device was turned on). This value starts at 0 and is incremented by 1 every time the "sequence number" field (see below) reaches the value of "Samples per second" (see above).
- **Sample data** (2 bytes x Channels x Samples per packet) There are "Samples per packet" sample sets per packet (see the table below for details).
    - **sample set** (2 bytes x Channels). A sample set consists of one sample for each channel, as captured at a particular time. [1]
        - **sample** (2 bytes, unsigned 16-bit integer, little-endian). A sample is an integer value as captured by an analog-digital converter [2] in the range 0-65535, with a grounded channel yielding a value near the middle of the range.[3]

The "Samples per packet" value that determines how many sample sets are in each packet is set depending on the number of channels being captured:

Channels	Samples per packet
1	500
2	250
4	125
8	86
16	43

2.2 Data structures

File format

The file format documented here is as produced by the capture.py program (JAGA Python v2.0 software package), or `jaga_ephys.m` in in MATLAB v2.0 software package.

The file contains one timestamp/payload pair as documented below.

- IEEE Little-endian double precision number of seconds since epoch, to microsecond precision (8 bytes)
- Wire format packet payload as documented below.

Wire format

All data is little-endian.

```
typedef struct tlm_packet_s
{
 tlm_data_header_t header;
 tlm_data_t samples[SAMPLES_PER_PACKET];
#if TTL
 uint8 ttl_data[(SAMPLES_PER_PACKET / 8) + 1];  // bit-packed TTL values.
#endif
} tlm_packet_t;

// TLM Format 3
typedef struct tlm_data_header_s
{
 uint16 format;              // For this version, format = 3.
 uint16 channels;
 uint16 diagnostic_word;
 uint16 mode_word;
  // Bit 15: Is 1 if TTL info present at end of packet
  // Bit 14: Is 1 if packet is CRC packet (Deprecated)
  // Bit 13: Is 1 if packet contains backlog value in sample counts. (value is in 16-bit
  //     diagnostic word)
  // Bit 12: Is 1 if bottom 8 bits indicates # of discarded packets since previous report
  //    (i.e. that could not be sent due to network traffic)
  // Bits 8-11 Unassigned.
  // Bits 0-7: If format = 0, this is bits-per-sample (always 16). If format > 0, value is
  //    indicated by bits above
 uint16 samples_per_second;
 uint32 elapsed_samples;  // Since capture began. Starts at 0 and
} tlm_data_header_t;

typedef struct tlm_data_s
{
 uint16 sample_data[CHANNELS];
} tlm_data_t;
```

2.3. Define Values for JAGA device

#define values

The #define values are set according to the specific configuration of the device. Find your configuration below. samples_per_second can vary depending on the specific configuration, and is recorded in each packet header as shown above.

16 Channels

#define CHANNELS 16

#define SAMPLES_PER_PACKET 43

8 Channels

#define CHANNELS 8

#define SAMPLES_PER_PACKET 86

4 Channels

#define CHANNELS 4

#define SAMPLES_PER_PACKET 125

2 Channels

#define CHANNELS 2

#define SAMPLES_PER_PACKET 250

1 Channel

#define CHANNELS 1

#define SAMPLES_PER_PACKET 500

2.4. TTL Data

JAGA can accept TTL inputs from an external device. For JAGA devices designed to accept and record TTL signals, the packet format is modified. To indicate that the packet carries TTL data, the most significant bit of the 'bits_per_sample' field is set to 1 (so the integer uint16 value would be 32768 + 16 = 32784).

The value of the TTL pin at each sample time is recorded in a bit-packed field at the end of the packet (after all samples). Because the chip used on the JAGA device is inherently 16-bit, the length of the TTL field is padded to a 16-bit boundary.

So:

- 16 channels: 43 samples → 48 bits (6 bytes)
- 8 channels: 86 samples → 96 bits (12 bytes)
- 4 channels: 125 samples → 128 bits (16 bytes)
- 2 channels: 250 samples → 256 bits (32 bytes)
- 1 channel: 500 samples → 512 bits (64 bytes)

In contrast to the rest of the packet data, the TTL data is big-endian, i.e. TTL values are packed from left to right into the TTL data field.

---

[1]The time between samples for a particular channel is exactly the sampling rate (error <0.01%). The time between samples of successive channels (e.g. channel 1 and channel 2 in the same sample set) depends upon exact device configuration, so if such information is needed please contact Jinga-hi.

[2]Analog Devices AD7980

[3]The exact value for a grounded channel varies slightly by channel and device, since all amplifiers are slightly different and there is a separate amplifier for each channel in the JAGA device. If the exact sample value is important, then ground all channels and record for a while, using the average value of each channel as = 0V.