Time Sync Protocol

Commands and protocol definition

Revision history

Date	Revision	Author	Description
2016/09/13	1	Daniele Comotti	First draft
2016/09/21	2	Daniele Comotti	Added examples and viewer changes
2016/10/13	3	Daniele Comotti	Updated document with clock drift command
2016/10/14	4	Daniele Comotti	Added command to set the clock drift

1 Introduction

This document is a brief description about the proposed implementation of the time sync method for the BioMX systems. The method aims to synchronize a platform with a central node (e.g. computer) by means of a set of commands (protocol). The parameters which need to be set are:

- Clock drift (or clock rate), a factor indicating the relative difference between the rate of increment of the platform clock and a real clock. This value doesn't change over time (under ideal conditions).
- Clock offset, the time shift between the platform clock and a real clock. This value changes each time the platform is shut down and need to be set again (with a different value) as soon as it starts up again.

2 Proposed protocol

The proposed method is based on two different procedures used at two different moments:

- Clock drift - this value can be set after the manufacturing of a platform is complete and the system is under test for check up purpose. During this very procedure, the clock drift is estimated and stored within the platform for later usage. This procedure could take 1 minute in order to get accurate results.
- Each time before (or after, in case real-time data are not needed) an experiment, the clock offset (with respect to the most recent start-up) is estimated.

The time sync requires the platform to be operated in a specific mode. To enter such mode, the following command shall be issued:

?	!	T	I	M	E	S	Y	N	C	!	?

Once the command is received, the platform switches to the time sync mode where each command is represented by 2 characters only, in order to minimize any transmission delay:

- To compute the clock drift, the timestamp of the platform must be retrieved in the fastest way available. To get it, issue the following command:

!	!

  The platform will respond with the following ASCII characters sequence:

C	D	tB4	tB3	tB2	tB1

  where tBi is the i-th byte of the 32 bits unsigned integer value representing the timestamp. Once at least two timestamps have been retrieved, the clock drift can be estimated as described in [1].
- As described in [1], the estimation of the clock offset needs two timestamp values. This can be done by issuing the following command at the platform:

?	?

As soon as the command is received, the platform computes the sum of two timestamp values each one separated from the other by an arbitrary amount of time (5 ms), and send it back in forms of the following ASCII characters sequence:

C	O	tB4	tB3	tB2	tB1

where tBi is the i-th byte of the cumulative value calculated by the platform, represented as a 32 bits unsigned integer. This value can be used to estimated the offset, as described in [1].

3 Usage

NOTE: Before using the firmware integrating time sync commands, erase the log memory of the Muse platform.

3.1 Clock Drift

The clock drift is supposed to be set only once. This can be done via the Viewer application, in the "Time Sync" panel from the "Settings" window. To read out the value, press on the "Readout" button. Please note that after the very first firmware upgrade, the clock drift value should be equal to 1. To compute a new value for the clock drift, press on the "Compute" button. The process estimating the clock drift lasts about 1 minute, after which the user is informed about the outcome.

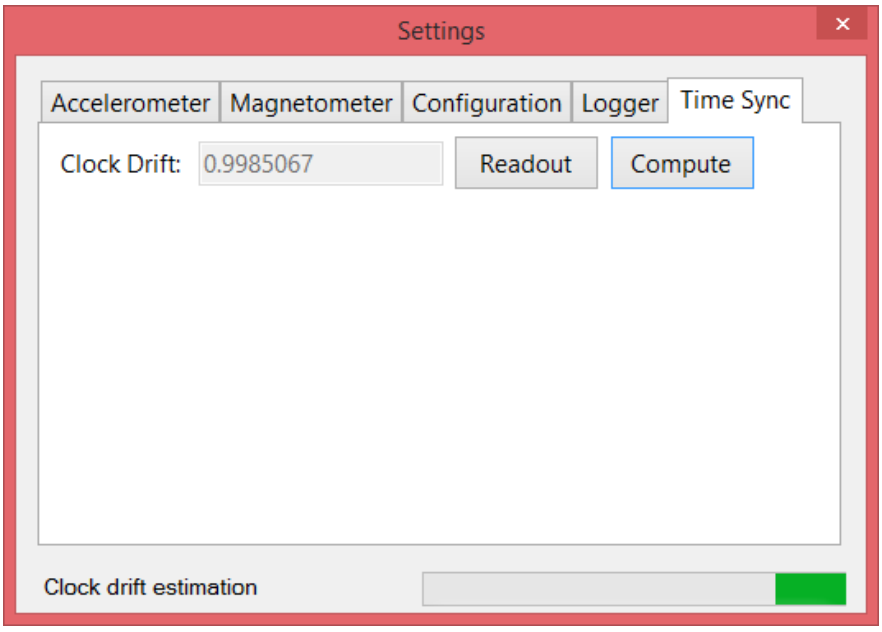

FIGURE 1: SETTINGS WINDOW WHILE PROCESSING THE CLOCK DRIFT.

Once the clock drift is computed, close the window and confirm to store the value permanently.

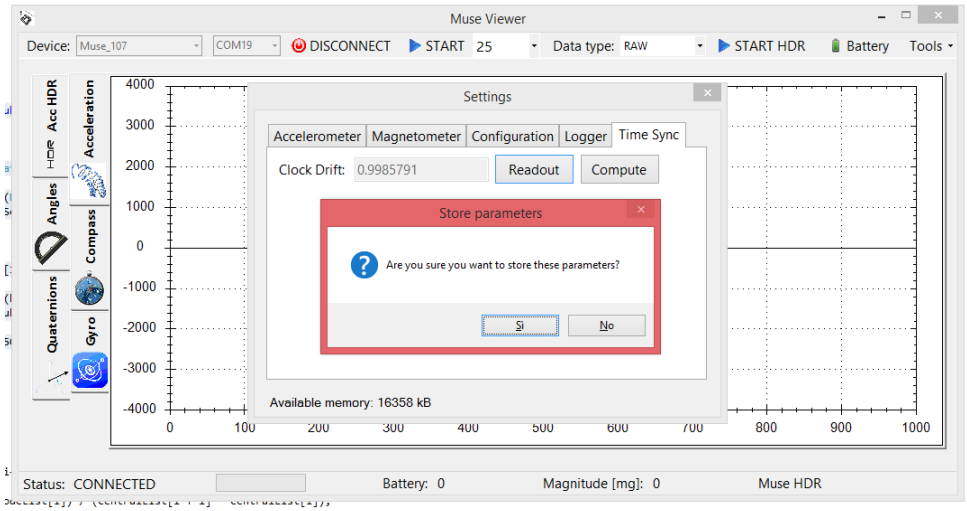

FIGURE 2: CONFIRMATION WINDOW TO STORE THE CLOCK DRIFT.

To estimate the clock drift within a different software, use the following guideline code to get the timestamps and refer to [1] for equations:

```
comPort.WriteLine("!!"); //Sends the command to retrieve a timestamp
while (comPort.BytesToRead <6) ;

    byte[] buffer = new byte[6];
    comPort.Read(buffer, 0, 6); //Readout the timestamp
    if (buffer[0] == 'C' && buffer[1] == 'D')
    {
       //Decode the timestamp
        uint t = decodeIntData(buffer[2], buffer[3], buffer[4], buffer[5]);

    }
    System.Threading.Thread.Sleep(5000);

//Iterate this process at least 2 times in such a way that the clock drift can
be computed
```

To retrieve the clock drift, issue the following command (the platform must NOT be in time sync mode):

?	!	G	E	T	D	R	I	F	T	!	?

The platform should reply with the following ASCII sequence:

C	D	dB4	dB3	dB2	dB1

where "CD" is the header, and dBi is the i-th byte of the floating point data representing the clock drift. A bitwise casting should be done to achieve the value correctly.

To set the clock drift, once it has been computed, issue the following command:

?	!	S	E	T	D	R	I	F	T	!	?	dB4	dB3	dB2	dB1

where dB4, dB2, dB2 and dB1 are the 4 bytes building up the floating point number representing the clock drift.

3.2 Clock Offset

The clock offset must be computed each time the system is used on after shut down. To do so, use the following guideline code and refer to equations shown in [1]:

```
//Read the clock drift and store it in "a"
```

```
                    float a=ReadClockDrift(); //Abstract function dealing with clock drift readout

                    comPort.WriteLine("?!TIMESYNC!?");
                    System.Threading.Thread.Sleep(200);

                   DateTime T1 = HighResolutionDateTime.UtcNow; //Save first TS on computer side
                   comPort.WriteLine("??"); //Clock offset command
                   while (comPort.BytesToRead < 2) ; //Wait for the confirmation
                   DateTime T4 = HighResolutionDateTime.UtcNow; //Sample the confirmation TS
                   while (comPort.BytesToRead < 6) ; //Wait for the cumulative TS packet
                   byte[] buffer = new byte[6];
                   comPort.Read(buffer, 0, 6); //Read the cumulative TS packet
                   if (buffer[0] == 'C' && buffer[1] == 'O')
                   {
                       uint TN=decodeIntData(buffer[2], buffer[3], buffer[4], buffer[5])); //Save
                          the cumulative TS from the node side
                       uint TC=(T1.Subtract(new DateTime(1970, 1, 1, 0, 0, 0,
DateTimeKind.Utc)).TotalMilliseconds + T4.Subtract(new DateTime(1970, 1, 1, 0, 0, 0,
DateTimeKind.Utc)).TotalMilliseconds); //Save the cumulative timestamp in epoch format (ms
       resolution) from the central side

                    float offset= ((float)TN)/a/2-((float)TC)/2; //clock offset computation

                   }
           comPort.WriteLine("?!"); //Exit Time Sync mode
```

The clock offset is typically a negative number.

Once the clock offset is computed, it can be used to achieve the correct timestamp starting from the timestamp of the platform:

$$t_{adj} = \frac{t_{rx}}{a} - b$$

where *a* is the clock drift and *b* is the clock offset.

4 References

[1] 221e, Wireless Sensor Network Synchronization, Feasibility study for application within a network of Muse platforms.