TIMEKEEPING MECHANISM AT SLS CONTROL SYSTEM

B. Kalantari, T. Korhonen, PSI, Villigen, Switzerland

Abstract

Time is one of the most important and critical parameters in a distributed control and measurement system. It is especially crucial when we need to interpret correlation of different archived process variables (PV) during the time. Advanced Light Source (APS) and Swiss Light Source (SLS) are using a very similar control system toolkit (EPICS) [1] and the same mechanism for timekeeping. Many input/output controllers (IOC) around the accelerator complex (including beamlines), run under a real-time operating system, and carry out the controls and data acquisition. Each IOC is responsible of keeping its own local time and time-stamps the local PV's but tightly synchronized with a central timing IOC. Dedicated timing hardware and network makes it possible to maintain synchronous timestamps with real-time clock. In this paper we describe the principle of this mechanism, its advantages, our experiences and further improvements.

INTRODUCTION

In many distributed systems the controlled components or subsystems require a common sense of time. This issue in accelerators is much more critical as the precise timing of synchronised actions is necessary to operate such a facility. Furthermore to study different phenomena and/or to consider their correlation a well defined mechanism to maintain the time is highly required.

To achieve these goals there should be a tight, clear and elegant coordination between hardware timing components and the software in one hand and well integration of those into the control system. Since EPICS has been specially designed to fulfil the controls requirements of experimental physics facilities (especially accelerators) this issue has been taken well into account from the very begging [2]. EPICS control system provides also means and tools to archive and retrieve the required data or PV's in the control system and to align them well in the time domain.

TIMEKIPEEING IN EPICS

In EPICS principle of timekeeping mechanism is based on a master-slave relation where there is one master and many slaves. On a network which consists many IOC's the master (timing) IOC is responsible to provide the timing information and synchronizes all the slaves. Each slave is responsible of maintaining its local time. They periodically verify their current time with the master current time and correct their local time if necessary.

Timestamp driver in EPICS core takes care of the whole mechanism. It implements both client (salve) and server (master). In an IOC whenever a record (which represents one or more PV) is processed the timestamp driver attaches the time of processing to it. The channel archiver in EPICS uses this timestamp to align the archived data in time. Each timestamp comprises two 32-bits integers showing seconds and nanoseconds passed EPICS epoch (1st of January, 1990).

TIMEKEEPING MODES

Synchronous mode

In synchronous mode the event timing hardware must exist [3]. The master timing IOC needs one event generator (EVG) and one event receiver (EVR). Each salve has to have an EVR. In synchronous mode all the IOC's will have exactly aligned timestamps.

In this mode the master IOC (by means of EVG) generates the real time clock (typically 1 KHz) so called 'tick' event and 'reset tick counter' event. All the salves and (also the master IOC) receive these events by EVR via the event system. Each EVR has a timestamp counter which is incremented upon receiving the 'tick' event. The timestamp support software (driver) knows the current time by looking to the time when the last 'reset tick counter' and the number of ticks which has elapsed since that time. Overflowing the timestamp counters is avoided by resetting them to zero upon receiving the 'reset tick counter' event. The master timing IOC also receives this event (called also sync event, typically every 10 second) whereupon the timestamp driver receives an indication by an interrupt and broadcast a timing message (here UDP) to all the slaves. The timing message includes the mater current time and some other information.

Soft mode

In the soft timestamp mode the timing hardware (EVG/EVR) does not exist. The master IOC and each slave maintain a software clock (a 1 tick watchdog at 60 Hz Rate) and the slaves ask the master for the current time periodically at sync rate (e.g. every 10 seconds).

PROBLEMS

It is clear that in the soft mode the timestamps are not precisely aligned, the time resolution is less and the correlation of the archived data is a bit difficult. So using synchronous mod eliminates all these disadvantages.

On the other hand the use of synchronous mode has also had a drawback, which is, when a problem happens to a synchronous salve timing hardware (e.g. event stream is lost) the time on this IOC stops and is frozen for ever. One of the reasons is that in the timestamp driver there is no way of switching between the soft mode and the sync mode (vxWorks tasks) for a running slave. This can be a serious problem especially in the control systems demanding high degrees of reliability like SLS. That has been the main reason not to use the synchronous mode for a long time as in most of the IOC's there PV's of interest

to be archived. There was also no external monitoring or controls on the timestamp driver and its parameters.

ENHANCEMENTS

We have rewritten parts of the timestamp driver in the client (slave) side to enhance the functionality and solve the mentioned problems. The synchronous slave tasks automatically switches to the soft mode upon detecting an event stream loss without any jump in the time. We have also developed means (soft device support) to monitor and have some limited controls over the timestamp driver in EPICS. For example it is possible to monitor timestamp type (sync or soft) or try to switch (or force) the type via channel access.

In figure 1 a test setup has been shown. This was the hoe we redeveloped some parts of the timestamp driver and tested that. There is a master timing IOC and a slave both are configured in synchronous mode. For simplicity the real time clock (tick event) is 10 Hz. To examine the archived values precisely we have an ADC (analogue to digital converter) in slave IOC. It reads a signal with half frequency of the interrupt (60 Hz). In figure 2, the yellow part shows the archived values in synchronous mode and the pink part shows them after switching to the soft mode.

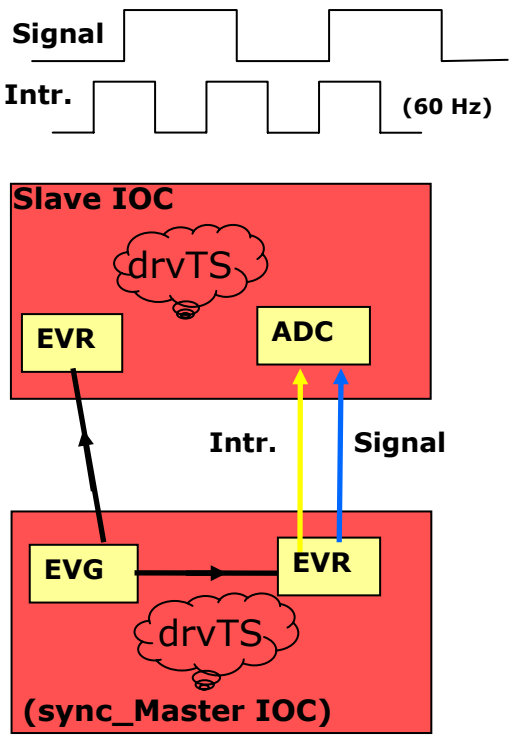

Figure 1: test setup for enhancements

The timestamps of archived values in synchronous mode are separated in 100 ms each and hitting values of the signal and of course they loose some periods because of the lower frequency than the interrupt (10 to 30 Hz). At the oval location the timing event link is taken out and the entire event stream is lost including 'tick' and 'reset tick counter' events. Then the timestamp driver switches to the soft mode with 60 Hz clock. The switching is done in one tick (1/60 s) and practically this is no jump in time. From this time on the salve continues to maintain the time in soft mode and still synchronizes with the master by periodic polling at every sync rate (e.g. 10 s).

```
Archive:     TEST-BABAK:ArchTStest []
02/04/2004 14:18:05.302486622   4.007782
02/04/2004 14:18:05.402486622   0.150301
02/04/2004 14:18:05.502486622   4.007172
02/04/2004 14:18:05.602486622   0.150301
02/04/2004 14:18:05.702486622   4.039216
02/04/2004 14:18:05.802486622   0.150607
02/04/2004 14:18:05.902486622   4.028840
02/04/2004 14:18:06.002486622   0.150607
02/04/2004 14:18:06.102486622   4.040436
02/04/2004 14:18:06.202486622   0.150301
02/04/2004 14:18:06.302486622   4.034943
02/04/2004 14:18:06.402486622   0.150301
02/04/2004 14:18:06.502486622   4.032807
02/04/2004 14:18:06.602486622   0.150912
02/04/2004 14:18:06.702486622   4.040436
02/04/2004 14:18:06.719153288   4.036164
02/04/2004 14:18:06.735819954   0.150607
02/04/2004 14:18:06.752486620   4.042267
02/04/2004 14:18:06.769153286   0.150301
02/04/2004 14:18:06.785819952   4.024262
02/04/2004 14:18:06.819153284   0.151217
02/04/2004 14:18:06.835819950   4.036164
02/04/2004 14:18:06.852486616   0.150607
02/04/2004 14:18:06.869153282   4.011749
02/04/2004 14:18:06.885819948   0.150607
```

Figure 2: archived values and mode switching

With the supports that we have added it is possible after resuming the event link to force the slave switch back to the synchronous mode. An alarm can be raised when a synchronous slave switches to soft mode.

IMPLEMENTATION AT THE SLS

The current implementation of the timekeeping in the SLS control system is shown in figure 3. The events are distributed via event system fibre optic links. The master distributes 'tick' and 'reset tick counter' events (also additional events required for the accelerator operation).

The event system hardware is connected in start-like via fibre optic links. The master IOC distributes events (by EVG) to each existing EVR. All the IOC's are connected to the local network (Ethernet) of the control system. The all timing information and synchronisation messages including the current time or synchronisation queries are done through UDP (User Datagram Protocol) on the local net (or subnet).

The master timing IOC gets its time upon reboot from the NTP (Network Time Protocol) server on the subnet.

When there is no master on the subnet all the IOC's are soft slaves (or direct salve) and synchronising themselves the NTP server at their sync rates. It is clear that the implementation anyway would be a combination of the soft and synchronous slaves (as shown in figure 3) as any IOC may not have timing hardware (EVR).

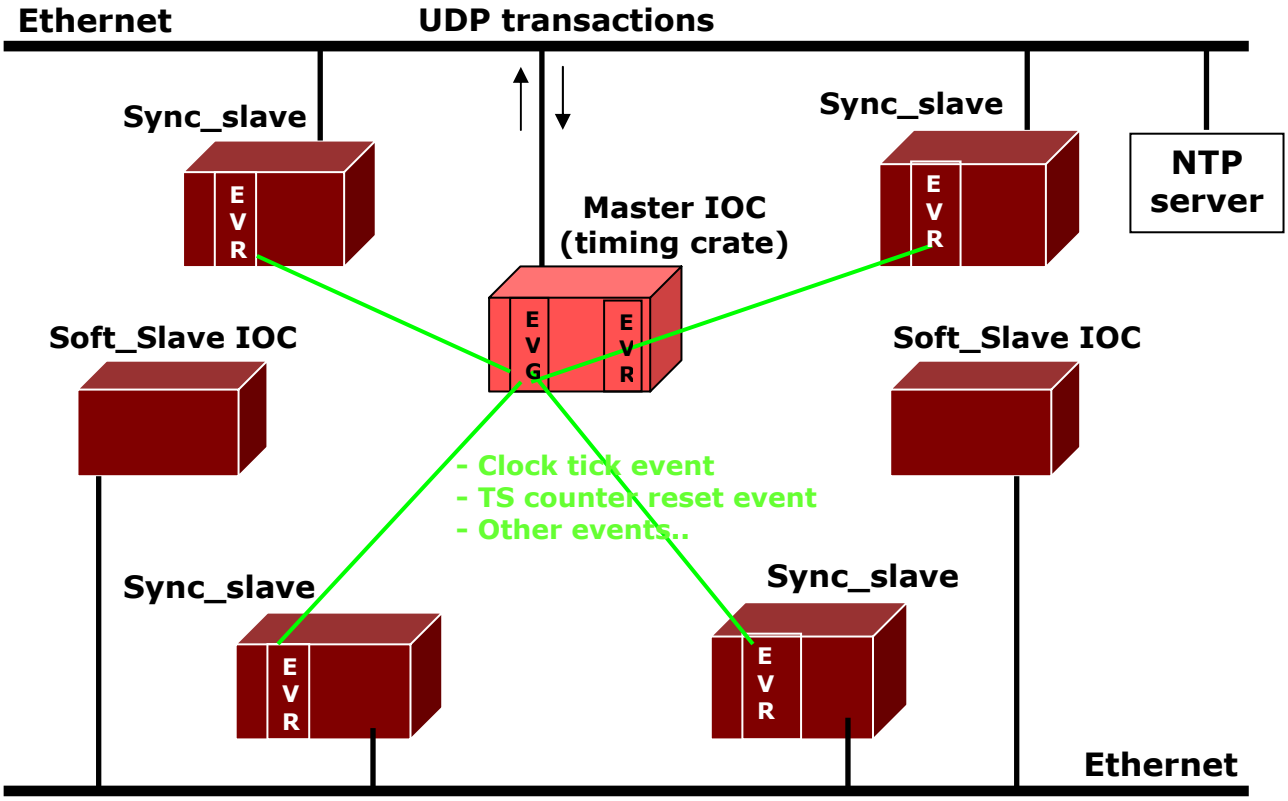

Figure 3: SLS implementation for timekeeping

FUTURE PLANS

In near future we are going to re design the timestamp software support in a way that uses the NTP protocols for correcting the time. At the moment the master sync to the NTP server is done in ever five weeks (at each shutdown) which causes a skew in time with outside application (outside of the control system).

REFERENCES

[1] http://www.aps.anl.gov/epics/
[2] http://www.aps.anl.gov/asd/controls/hideos/GTS.html
[3] http://www.sls.psi.ch/controls/help/howto/globalEventV2.pdf