TIMEKEEPING MECHANISM AT SLS/APS CONTROL SYSTEM

B. Kalantari, T. Korhonen

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 correlations of different archived process variables (PV). The Advanced Photon Source (APS) and the 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 for keeping its own local time and time-stamps the local PVs, tightly synchronized with a central timing IOC. Dedicated timing hardware and network makes it possible to maintain synchronous timestamps with a real-time clock.

INTRODUCTION

In many distributed systems the controlled components or subsystems require a set of synchronized clocks. This issue is highly critical for accelerators 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 time is required.

To achieve these goals there should be a tight, clear and elegant integration between hardware timing components, the software and 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 beginning [2]. The 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.

TIMEKEEPING IN EPICS

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

The Timestamp driver in the EPICS core takes care of the whole mechanism. It implements both client (slave) and server (master) sides. In an IOC whenever a record (which represents one or more PVs) 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 since the beginning of the 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 slave has to have an EVR. In synchronous mode all the IOC's will have exactly aligned times.

In this mode the master IOC (by means of EVG) generates the real time clock (typically 1 KHz), the so-called 'tick' event, and 'reset tick counter' event. All the slaves (as well as the master) receive these events from 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 at the time of the last 'reset tick counter' and the number of ticks which have 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 seconds) whereupon the timestamp driver receives an indication by an interrupt and broadcasts a timing message (here UDP) to all the slaves. The timing message includes the master 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 precise and the correlation of the archived data is a bit difficult. Using the synchronous mode eliminates all these disadvantages.

On the other hand the use of the synchronous mode has also had a drawback: when a problem happens to a synchronous slave timing hardware (e.g. an event stream is lost) the time on this IOC stops and is frozen forever. 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 degree 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 are 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 switch 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 some limited control over the timestamp driver in EPICS. For example, it is possible to monitor the timestamp type (sync or soft) or try to switch (or force) the type via the channel access.

By setting up a test stand we redeveloped some parts of the timestamp driver and tested that. A master timing IOC and a slave both were configured in synchronous mode. For simplicity the real time clock (tick event) was chosen at 10 Hz. To examine the archived values precisely we had an ADC (analogue to digital converter) in a slave IOC. It read a signal with half the frequency of the interrupt (60 Hz). The time stamps of the archived values in synchronous mode were separated by 100 ms. Some periods are lost because the sampling frequency is lower than the signal. At some stage the frequency timing event link was taken out and the entire event stream was lost including 'tick' and 'reset tick counter' events. Then the timestamp driver switches to the soft mode with the 60 Hz clock. The switching is done in one tick (1/60 s). From this time on the slave continued to maintain the time in soft mode and still was synchronized with the master by periodic polling at the sync rate (e.g. 10 s).

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

IMPLEMENTATION AT THE SLS

The events are distributed via the 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 a star-like configuration via dedicated fibre optic links. The master timing IOC distributes events (by EVG) to each existing EVR. All the IOC's are connected to the local network (Ethernet) of the control system. All the timing information and synchronisation messages including the current time or synchronisation queries are provided 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 slave) and are 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 some of the IOCs may not have the timing hardware (EVR).

The redeveloped timestamp driver is used by operating subsystems at the SLS and has proved to be stable and reliable.

FUTURE PLANS

In the near future we are going to redesign 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 every five weeks (at each shutdown) which causes a drift in time with respect to external applications (external to 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