# New time stamp functionality in R3.14

David Thompson SNS

Carl Lionberger SNS

Andrew Johnson APS

# Objectives

- SNS Machine Protection System needed high resolution time stamps for logging faults
- SNS diagnostic systems, running on PC, start processing data long before the data is made available to IOC core, and can stamp data based on event link and RTDL times.
- Desirable to be able to pass time through calc records etc.
- VERY desirable not to break existing code or require modifications to existing record support routines!!!

# Why base needed to be changed

- Device support is where the time stamp can be calculated in some applications.
- Record time stamps are set by record support after device support returns the value so device support has no opportunity to override the time stamp. (Unless the process() function in every input record type is changed.)
- Available time stamps were “current time” or “event N” time depending on the timing system in use. To use the event system  meant that a new API would have to be invented for device support time stamps.
- Existing API has no support for passing time with data flow.

# The “New” recGblGetTimeStamp()


```
void epicsShareAPI recGblGetTimeStamp(void* prec) {
    struct dbCommon* pr = (struct dbCommon*)prec;
    struct link *plink = &pr->tsel;

    if(plink->type!=CONSTANT) {
        struct pv_link *ppv_link = &plink->value.pv_link;

        if(ppv_link->pvlMask&pvlOptTSELisTime) {
            long status = dbGetTimeStamp(plink,&pr->time);
            if(status)
                errlogPrintf("%s recGblGetTimeStamp dbGetTimeStamp failed\n",
                    pr->name);
            return;
        }
        dbGetLink(&(pr->tsel), DBR_SHORT,&(pr->tse),0,0);
    }
    if(pr->tse!=epicsTimeEventDeviceTime) {
        int status;
        status = epicsTimeGetEvent(&pr->time,pr->tse);
        if(status) errlogPrintf("%s recGblGetTimeStamp failed\n",pr->name);
    } }
```