Star Controller

Table des matières

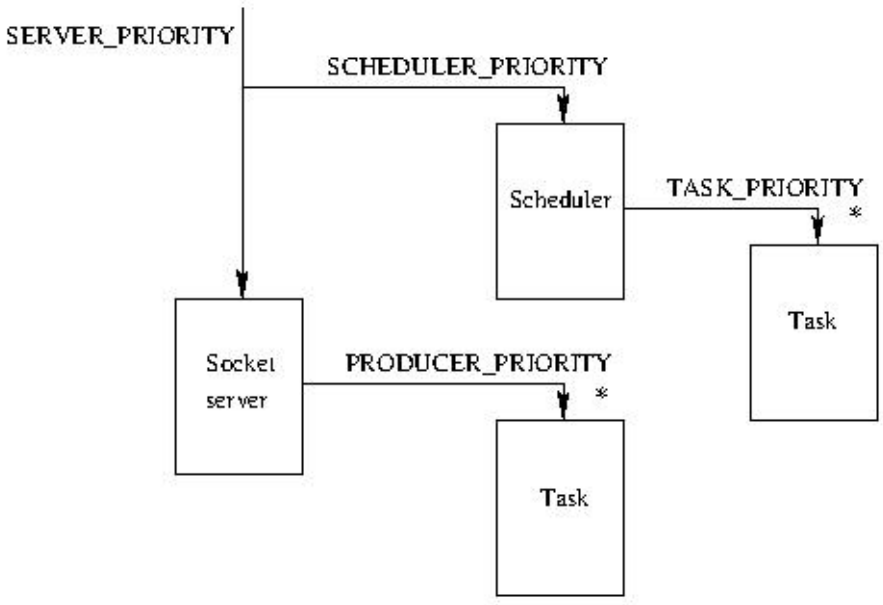
1 Introduction

Ce document décrit la gestion des pixels de la caméra.
Les pixels peuvent être éteind

- suite à la détection d'un voltage trop élevé
- ou en prévention.

Il faut par ailleurs rallumer les pixels éteints.

2 Code HESS1



Dans HESS1, Le code utilitaire de gestion des pixels est appellé au niveau du scheduleur du serveur de socket et de l'acquisition.

- *Sdaq/Server/Daqd/VERSION_0.00/src/acq_camera.c* :
  This process is real time acquisition of the HESS telescope camera on interrupt from the IO register board

```
void *thread_inbuf (void *arg) {
 /* block all signals */
 while(1) {
  sigwait(&ensemble, &num);
  if(sig_monitoring == SIG_CAMERA_HTMON) {
   headp = (Event_Header_t *) &cbuff[inbuff];
   Do_Slc ((Event_Header_t *) &cbuff[inbuff],...)

   //get present time
   if(!DrawerConfig) {
    drawer_ptr = (unsigned short *) &cbuff[inbuff] + header_size;
    while(i < headp->data_size)
     DrawerSurvey(drawer_ptr + i, papat_desc, 0xf, 0xf, 0x3b, time_of_endread.tv_sec);
}}}}

int main(int argc, char *argv[])
{
 tcam_desc = open("/dev/camera", O_RDWR);
 common_def.des_camera = tcam_desc; // pass camera-device desc

 status = pthread_create(&thr_inbuf, &thread_attr, thread_inbuf, NULL);
 ...
 status = pthread_join(thr_inbuf, &thread_result);
 exit(0);
}
```

- *Sdaq/Server/Slcd/VERSION_3.00/Consumer/Task_DATA_Consumer.c* :
  Scan FIFO_Buffer and search for slow control or DAQ event Check header and tailer of the block, block identificator and corresponding size.
  Store data in DATA_Buffer to be read end send by other consumers

```
void Task_DATA_Consumer(void *mytask){...
do{...
 /* Store locale GPS/CPU time stamp */
 locate_time_sec = *(int *)(pDATA+10);

 /* check online star monitoring prediction if PM are not in transition state
 SurveyStars(locate_time_sec, slcd.pattern);

 /* Loop on all 4xFIFO */
 for(ii=0; ii<4; ii++)
  /* If high voltage is on then check the behaviour of photo multipliers */
  err = DrawerSurvey(pDATA_save,...)

} while (task->_status & TASK_STATUS_CREATED);
pthread_exit(0);
}
```

- *Sdaq/Server/Slcd/VERSION_3.00/Util/Warning.c*

```
int SwitchOffPixel(int drawer, int pixel, int fifo, int pattern, char *reason){
 Config[drawer].cntrl_ht_16.HV_status &= ~(0x1<<pixel);
 ierr = cPCI_FIFO_Write_Block(fifo, (short *)&Config[drawer].cntrl_ht_16, pattern);
}

int SwitchOnPixel(int drawer, int pixel, int fifo, int pattern, char *reason){
 Config[drawer].cntrl_ht_16.HV_status |= 0x1<<pixel;
 ierr = cPCI_FIFO_Write_Block(fifo, (short *)&Config[drawer].cntrl_ht_16, pattern);
}

int SurveyStars(unsigned long long int time, unsigned int pattern){
 while(1){
  if (!ioctl(common_def.des_camera, SURVEY_STAR_CHECK, &star)) break;
  drawer_num = DrawerId2Number(star.drawer);

  switch(star.InOrOut){
  case 1: " Star enter (drawer 0x%X : HT=0x%X)"
  SwitchOffPixel(...)
  case 0: " Star leave (drawer 0x%X : HT=0x%X)"
  SwitchOffPixel(...)
  }

  ioctl(common_def.des_camera, SURVEY_STAR_REMOVE, star.id);
}}

int DrawerSurvey(short *pDATA,...){
 /* Check limit on HTI value */
 SwitchOffPixel(drawer_num, pixel, fifo, pattern, message);
 /* Check HT value send for this drawer */
 SwitchOffPixel(drawer_num, pixel, fifo, pattern, message);
 /* Check noise intensity - HTI vs HTV */
 SwitchOffPixel(drawer_num, pixel, fifo, pattern, message);
}
```

- *Sdaq/Server/Slcd/VERSION_3.00/src/SURVEYControl.c* :
  Control sun, moon and star

```
int SURVEYControl(char *Buffer){
 if (!Look4Key(local_buffer,"SUN",3))...
 else if (!Look4Key(local_buffer,"MOON",4))...
 else if (!Look4Key(local_buffer,"STAR",4))...
 else if (!Look4Key(local_buffer, "CLEAR", 5))...
 else if (!Look4Key(local_buffer,"DUMP",4))...
}
```

- *Sdaq/Driver/tcamera/drv_camera.c* :
  Linux driver for the HESS telescope camera this is a logical driver, no hardware is directly involved

```
case SURVEY_SET_SUN:       Run.survey.sun  = *(Solar_t *)ioctl_param;        break;
case SURVEY_SET_MOON:      Run.survey.moon = *(Solar_t *)ioctl_param;        break;
case SURVEY_STAR_ADD:
  if (!(star = kmalloc(sizeof(struct star_t), GFP_KERNEL))) return(ENOMEM);
  *star = *(Star_t *)ioctl_param;
  /* gestion de liste ... */
  Run.survey.nbStars++;
  return(star->id);
case SURVEY_STAR_REMOVE:
case SURVEY_STAR_CHECK:
case SURVEY_GET_STAR_ID:
case SURVEY_GET_STAR_NUMBER:
case SURVEY_GET_SUN:
case SURVEY_GET_MOON:
case SURVEY_GET_NUMBER_OF_STARS:
```

Question : Où se trouve le code du star controller ?

3 Extinction préventive

Grammaire donnée par le controler :

```
/* star survey */
  else if (!Look4Key(local_buffer,"STAR",4))
    {
      local_buffer +=5;
      if (sscanf(local_buffer,"IN %2c%02d %Ld",drawer,&pixel,&val) == 3)
      ...
      else if (sscanf(local_buffer,"OUT %2c%02d %Ld",drawer,&pixel,&val) == 3)
```

On simulera le Star Controller par un fichier à l'aide de la commande date.
Question : s'agit-il d'autre chose que du controleur ?

4 Voltage trop élevé

```
int DrawerSurvey(short *pDATA,...)
{
   ...
   SwitchOffPixel(...);
}
```

Question : à implémenter dans *Task_Drawer_Monitoring.c* ?

5 Ré-allumage des pixels

Il faut élaborer la liste des pixels éteints.
Tableau des pixels :

```
typedef struct
{
  ...
  Drawer_HV_16_t           cntrl_ht_16;
  ...
} Slc_config_16;

SEXTERN Slc_config_16 Config[__NUMBER_OF_DRAWERS];
```

Réaction :

- 4 à 6 pixels étends : étoiles (pas grave)
- 6 à 120 pixels étends : étoile filante (pas grave)
- $>$ 120 : voiture ou éclairs (on stop l'acquisition)

question : à implémenter dans une nouvelle tâche du squeduler ?