相机使用文档

作者	时间	版本	备注
王建阳	2020-08-29	v1.0.0	
王建阳	2020-09-17	v1.0.1	1.510g和545r1.1添加内置cpld,更改触发方式。 2.tztek_env支持zhihua相机

初始化配置相机类型

1.查看相机配置命令： cat /proc/tztek_env

```
nvidia@nvidia-desktop:~$
nvidia@nvidia-desktop:~$ cat /proc/tztek_env
#amera type test
#//sensing-9295-1920*1080-1x1
#//sensing-9295-1920*1080-1x2
#//sensing-96705-1920*1080-1x1
#//sensing-96705-1920*1080-1x2
#//sensing-96705-1280*720-1x1

[camera0]
type = sensing-96705-1920*1080-1x2
type_b = sensing-96705-1920*1080-1x2
type_c = sensing-96705-1920*1080-1x2
type_d = sensing-96705-1920*1080-1x2



//for test
[Section4]
key4: value4
key1: value4
nvidia@nvidia-desktop:~$
nvidia@nvidia-desktop:~$
```

图中，type 为相机的v1，v2配置；type_b 为相机的v3，v4配置；type_c 为相机的v5，v6配置；type_d 为相机的v7，v8配置；

sengsing相机

目前配置都是支持森云的相机，森云相机主要有两款串化芯片max96705和max9295，所以类型就以96705和9295命名。一般ISP为AR0231的相机是max96705串化，需要将类型设置为sensing-96705-1920*1080-1x2

一般ISP为imx390的相机为max9295串化，需要将类型设置为sensing-9295-1920*1080-1x2。

目前暂时由于max9295的串化还未做好，做不到随意侦测相机，需要指定相机插入个数，比如v1,v2插入两个9295相机,就需要将type = sensing-9295-1920*1080-1x2。v1只插入一个相机，就需要将type = sensing-9295-1920*1080-1x1。

如下图就是八路相机都设为sensing---9295

```
wangjianyang@DELL:~/flash_tztek_env$
wangjianyang@DELL:~/flash_tztek_env$
wangjianyang@DELL:~/flash_tztek_env$ cat tztek_env.conf
#camera type test
#//sensing-9295-1920*1080-1x1
#//sensing-9295-1920*1080-1x2
#//sensing-96705-1920*1080-1x1
#//sensing-96705-1920*1080-1x2
#//sensing-96705-1280*720-1x1
#//zhihua-96701-1x2
#//zhihua-96701-1x1

[camera0]
type = sensing-9295-1920*1080-1x2
type_b = sensing-9295-1920*1080-1x2
type_c = sensing-9295-1920*1080-1x2
type_d = sensing-9295-1920*1080-1x2


wangjianyang@DELL:~/flash_tztek_env$
```

max96705的随意侦测已经做好，不需要指定一个端口插入相机个数，不论v1,v2插入一个96705相机还是两个96705相机,只需要将type = sensing-96705-1920*1080-1x2 设置好就行。

如下图就是八路相机都设为sensing---96705

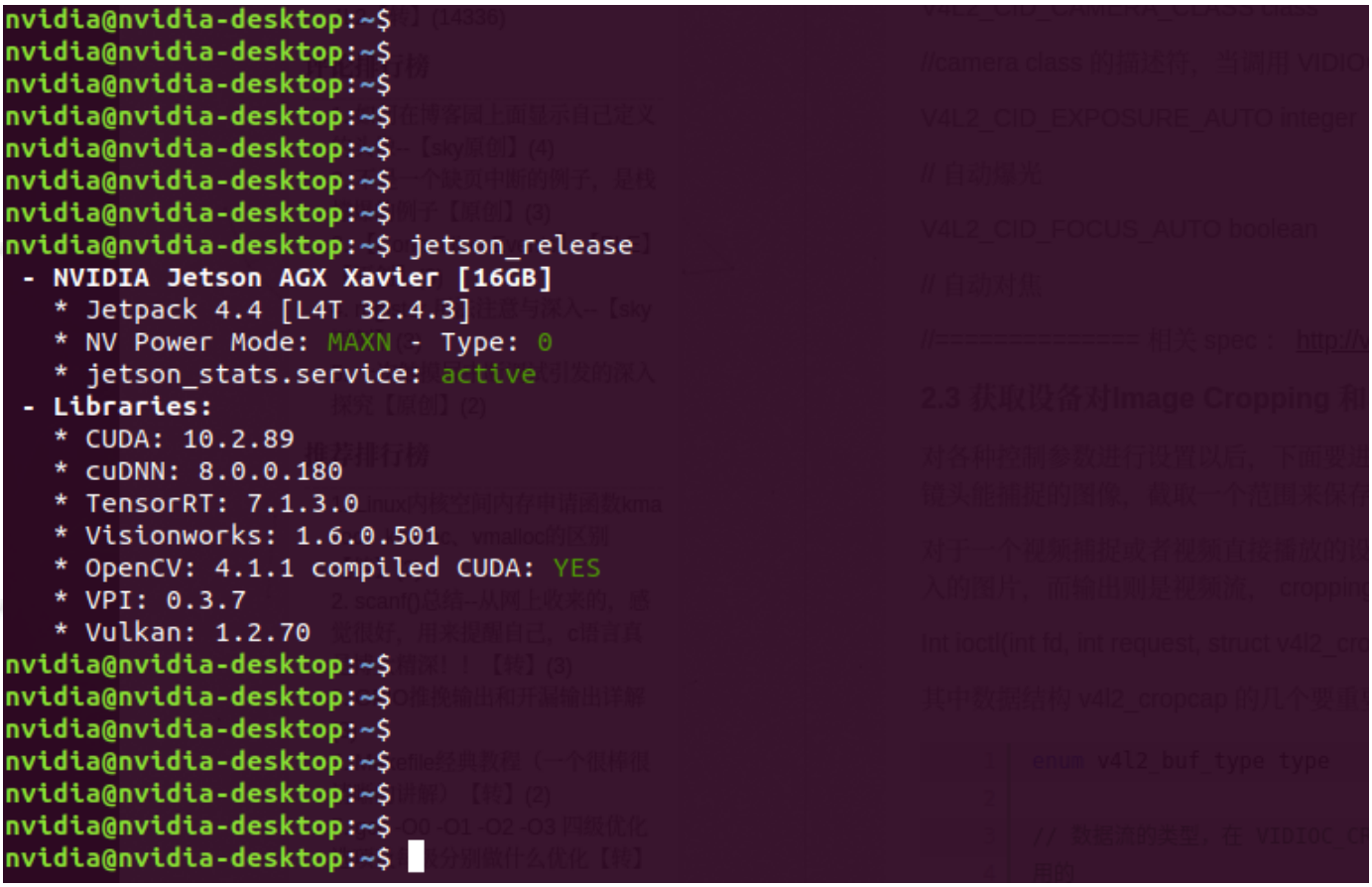
```
wangjianyang@DELL:~/flash_tztek_env$
wangjianyang@DELL:~/flash_tztek_env$
wangjianyang@DELL:~/flash_tztek_env$
wangjianyang@DELL:~/flash_tztek_env$ cat tztek_env.conf
#camera type test
#//sensing-9295-1920*1080-1x1
#//sensing-9295-1920*1080-1x2
#//sensing-96705-1920*1080-1x1
#//sensing-96705-1920*1080-1x2
#//sensing-96705-1280*720-1x1
#//zhihua-96701-1x2
#//zhihua-96701-1x1

[camera0]
type = sensing-96705-1920*1080-1x2
type_b = sensing-96705-1920*1080-1x2
type_c = sensing-96705-1920*1080-1x2
type_d = sensing-96705-1920*1080-1x2


wangjianyang@DELL:~/flash_tztek_env$
```

zhihua相机(只有jetpack 4.4可以使用此相机，硬件对应510g和545r1.1)

查看系统版本方法:在目标机器里输入　jetson_release



目前zhihua相机主要配置的是串化芯片为max96701,所以类型为96701命名。

max96701的随意侦测已经做好，不需要指定一个端口插入相机个数，不论v1,v2插入一个96701--zhihua相机还是两个96701--zhihua相机,只需要将type = zhihua-96701-1x2 设置好就行。

如下图就是八路相机都设为zhihua---96701

```
wangjianyang@DELL:~/flash_tztek_env$
wangjianyang@DELL:~/flash_tztek_env$
wangjianyang@DELL:~/flash_tztek_env$ cat tztek_env.conf
#camera type test
#//sensing-9295-1920*1080-1x1
#//sensing-9295-1920*1080-1x2
#//sensing-96705-1920*1080-1x1
#//sensing-96705-1920*1080-1x2
#//sensing-96705-1280*720-1x1
#//zhihua-96701-1x2
#//zhihua-96701-1x1

[camera0]
type = zhihua-96701-1x2
type_b = zhihua-96701-1x2
type_c = zhihua-96701-1x2
type_d = zhihua-96701-1x2


wangjianyang@DELL:~/flash_tztek_env$
```

(ps: 智华相机一般为25fps 和 30fps ,由于这是由相机固件决定的，我们是改变不了，所以若是确定需要一个帧率，需要相机厂商支持烧写固件。智华相机的输出分辨率一般为1280*720 或者1280*960,但是如果他的相机固件没有删除ebd信息，输出分辨率就为1280*719或者1280*959, 若是您需要确定一个输出分辨率比如需要　1280*720,那就需要相机厂商支持烧录固件1280*720,删除ebd信息。)

2.烧写tztek_env

既然知道了tztek_env的作用，就需要修改并烧写tztek_env分区 ， 将flash_tztek_env.tar.gz拷贝到机器里，解压flash_tztek_env_1_1.tar.gz，tar –xpf　flash_tztek_env_1_1.tar.gz

📎flash_tztek_env_1_1.tar.gz

进入flash_tztek_env

```
nvidia@nvidia-desktop:~/flash_tztek_env$
nvidia@nvidia-desktop:~/flash_tztek_env$ ls
readme.txt  tztek_dd_tztek_env_bin.sh  tztek_env.bin  tztek_env.conf  tztek_env_gen  tztek_env_gen.txt  tztek_env.sh
nvidia@nvidia-desktop:~/flash_tztek_env$
nvidia@nvidia-desktop:~/flash_tztek_env$
nvidia@nvidia-desktop:~/flash_tztek_env$
nvidia@nvidia-desktop:~/flash_tztek_env$
nvidia@nvidia-desktop:~/flash_tztek_env$
```

里面包含readme.txt 可以查看。

```
1.给*.sh加权限，sudo chmod a+x *.sh

2.修改tztek_env.conf 后，执行./tztek_dd_tztek_env_bin.sh

8. dd完以后,重启sudo reboot,重启后，进入系统cat /proc/tztek_env 查看修改信息是否ok。
~
~
~
~
~
~
~
~
~
```

查看相机是否初始化成功

命令dmesg ｜grep 9296

```
nvidia@nvidia-desktop:~$ dmesg |grep 9296
[    3.304338] max9296_probe: platform-name is nx
[    3.304345]  max9296_probe : max9296_node is error, check node->name
[    3.304395] max9296_probe: err: no platform-name property
[    3.304413] max9296_env_init, type = sensing-96705-1920*1080-1x2, type_num = 5
[    5.803805] max9296_detect_init,1 : initial successful
[    5.967815] max9296_camera_type, 4
[    6.951795] max9296_camera_type,1 : initial failed
[    7.059792] max9296_camera_type, 4
[    8.043809] max9296_camera_type,1 : initial failed
[    8.043903] max9296_probe: err: no platform-name property
[    8.043926] max9296_env_init, type = sensing-96705-1920*1080-1x2, type_num = 5
[   10.571785] max9296_detect_init,2 : initial successful
[   10.735811] max9296_camera_type, 4
[   11.719804] max9296_camera_type,2 : initial failed
[   11.827806] max9296_camera_type, 4
[   12.811810] max9296_camera_type,2 : initial failed
[   12.811884] max9296_probe: err: no platform-name property
[   12.811930] max9296_env_init, type = sensing-96705-1920*1080-1x2, type_num = 5
[   15.371775] max9296_detect_init,3 : initial successful
[   15.459785] csi_x2_max9296_info->type = 9
[   15.459790] csi_x2_max9296_info->initialize_twice = 1
[   15.459822] max9296_info->type = 9
[   15.459831] max9296_info->camera = 3
[   15.567781] max9296_camera_type, 9
[   19.595769] max9296_camera_type,3 : initial successful
[   19.703771] max9296_camera_type, 9
[   23.699771] max9296_camera_type,3 : initial successful
[   23.699861] max9296_probe: err: no platform-name property
[   23.699882] max9296_env_init, type = sensing-96705-1920*1080-1x2, type_num = 5
[   26.283771] max9296_detect_init,4 : initial successful
[   26.339789] csi_x2_max9296_info->type = 10
[   26.339795] csi_x2_max9296_info->initialize_twice = 1
[   26.339799] max9296_info->type = 10
[   26.339808] max9296_info->camera = 4
[   26.447774] max9296_camera_type, 10
[   30.427779] max9296_camera_type,4 : initial successful
[   30.535771] max9296_camera_type, 10
[   34.483771] max9296_camera_type,4 : initial successful
```

图中max9296_camera_type,1 ： initial failed 就表示v1,v2 初始化失败相机，相机失败原因可能，1.插入相机的型号与tztek_env中配置不符。2.v1,v2未插入相机

max9296_camera_type,3 ： initial successful 就表示v4,v5初始化相机成功。

相机的图像显示

1.no_show 显示查看帧率

在tztek_test脚本里的 test/camera_no_show/camera_no_show_fps_v1.2/中，

📎tztek_test_510_545_v1.2_07_20.tar.gz

若查看v1相机则执行./camera0_no_show_v1.2_1920_1080.sh 或者./camera0_no_show_v1.2_1280_720.sh

1920*1080还是1280*720 这个是由相机决定，根据你所使用的相机，可以修改宽与高。

```
nvidia@nvidia-desktop:~/tztek_test_510_v1.2_0520/test/camera_no_show/camera_no_show_fps_v1.2$
nvidia@nvidia-desktop:~/tztek_test_510_v1.2_0520/test/camera_no_show/camera_no_show_fps_v1.2$ ./camera5_no_show_v1.2_1920_1080.sh
<<<<<<<<<<<<<<<<<<<<<<<< 22.11 fps
<<<<<<<<<<<<<<<<<<<<<<<< 22.05 fps
<<<<<<<<<<<<<<<<<<<<<<<< 22.03 fps
<<<<<<<<<<<<<<<<<<<<<<<< 22.02 fps
<<<<<<<<<<<<<<<<<<<<<<<< 22.02 fps
<<<<<<<<<<<<<<<<<<<<<<<< 22.01 fps
nvidia@nvidia-desktop:~/tztek_test_510_v1.2_0520/test/camera_no_show/camera_no_show_fps_v1.2$
```

修改宽与高：vim camera0_no_show_v1.2_1920_1080.sh

```
#!/bin/bash

v4l2-ctl --set-fmt-video=width=1920,height=1080 --stream-mmap --stream-count=10000000000 -d /dev/video0 #--stream-to=camera_0_1280_720.yuv
~
~
```

修改width，height。

2.show cuda 查看图像

在tztek_test脚本里的 test/camera_show_cuda中，若查看v1相机则执行./camera0_show_cuda，若查看v2相机则执行./camera1_show_cuda

```
nvidia@nvidia-desktop:~/tztek_test_510_v1.2_0520/test/camera_show_cuda$ ls
camera0_show_cuda      camera2_show_cuda      camera4_show_cuda      camera6_show_cuda      make_camera0_show_cuda.sh  make_camera4_show_cuda.sh  README.TXT
camera0_show_cuda.cpp  camera2_show_cuda.cpp  camera4_show_cuda.cpp  camera6_show_cuda.cpp  make_camera1_show_cuda.sh  make_camera5_show_cuda.sh  yuyv2rgb.cu
camera1_show_cuda      camera3_show_cuda      camera5_show_cuda      camera7_show_cuda      make_camera2_show_cuda.sh  make_camera6_show_cuda.sh  yuyv2rgb.cuh
camera1_show_cuda.cpp  camera3_show_cuda.cpp  camera5_show_cuda.cpp  camera7_show_cuda.cpp  make_camera3_show_cuda.sh  make_camera7_show_cuda.sh
nvidia@nvidia-desktop:~/tztek_test_510_v1.2_0520/test/camera_show_cuda$
nvidia@nvidia-desktop:~/tztek_test_510_v1.2_0520/test/camera_show_cuda$
nvidia@nvidia-desktop:~/tztek_test_510_v1.2_0520/test/camera_show_cuda$
nvidia@nvidia-desktop:~/tztek_test_510_v1.2_0520/test/camera_show_cuda$
```

默认宽与高都是1920*1080，若需要修改，则vim camera0_show_cuda.cpp ，修改宏定义 PIXL_WIDTH 和 PIXL_HEIGHT

```
#include "yuyv2rgb.cuh"


static int index_get = 0;
static int index_pro = 0;

#define CLEAR(x) memset(&(x), 0, sizeof(x))
#define BUFFER_LENGHT           4

#define PIXL_WIDTH              1920
#define PIXL_HEIGHT             1080
```

修改完之后需要编译，执行 ./make_camera0_show_cuda.sh。

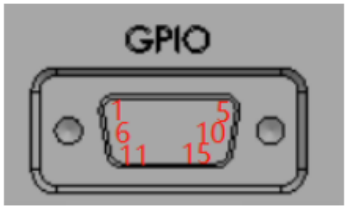
相机的触发功能

1 .由于不同的相机可能有不同的出图，有的是自动出图，有的是需要触发信号才能出图。不同的机器类型给相机的触发方式不一样，下面分别描述XAVIER-GEAC91和NX-GEAC90的触发方式。

XAVIER-GEAC91

1.GEAC91为外部触发，通过外部预留的GPIO接口，接受外部的触发信号，下面是GPIO引脚定义

GPIO PIN 定义



接口名称	引脚序号	接口信号定义	接口说明
Serial(MULTI)	1	GND	GND
	2	GPIO_INPUT1	GPIO_INPUT1
	3	GPIO_INPUT2	GPIO_INPUT2
	4	GPIO_INPUT3	相机触发
	5	GPIO_INPUT4	相机触发
	6	GPIO_INPUT5	相机触发
	7	GPIO_INPUT6	相机触发
	8	GND	GND
	9	GPIO_OUTPUT1	GPIO_OUTPUT1

	10	GPIO_OUTPUT2	GPIO_OUTPUT2
	11	GPIO_INPUT7	相机触发
	12	GPIO_INPUT8	相机触发
	13	GPIO_INPUT9	相机触发
	14	GPIO_INPUT10	相机触发
	15	GND	GND

其中4号pin脚GPIO_INPUT3为v1，v2两路相机的触发，6号pin脚GPIO_INPUT5为v3,v4两路相机触发，11号pin脚GPIO_INPUT7为v5,v6两路相机触发，13号pin脚GPIO_INPUT9为v7,v8两路相机触发。

（附：一般imx390的相机可能会有是需要触发才出图，它需要的触发信号为30hz，方波，高电平3.3v，低电平0v，高电平3.33us, 如果高电平的精度不够，这个相机有概率会是黑屏）

XAVIER-GEAC91(510g)

📎trig_new_510g.tar.gz

1.510g内置cpld可通过串口发命令使cpld给出触发信号

进入　trig_new

```
wangjianyang@DELL:~/trig_new$
wangjianyang@DELL:~/trig_new$
wangjianyang@DELL:~/trig_new$ ls
510g_all_10_fps  510g_all_20_fps  510g_all_25_fps  510g_all_30_fps  include  lib  Makefile  obj  source
wangjianyang@DELL:~/trig_new$
wangjianyang@DELL:~/trig_new$
wangjianyang@DELL:~/trig_new$
wangjianyang@DELL:~/trig_new$
```

需要10hz触发则执行 sudo ./510g_all_10fps，25,20,30fps依次类推

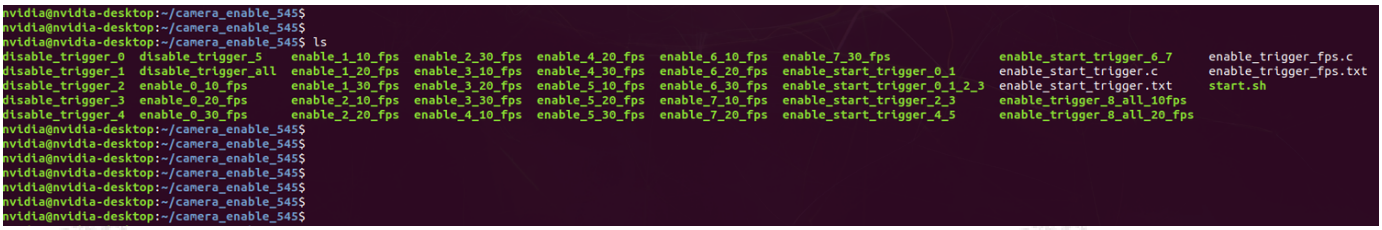
```
wangjianyang@DELL:~/trig_new$
wangjianyang@DELL:~/trig_new$ sudo ./510g_all_10_fps
```

NX-GEAC90

1.NX-GEAC90有内部触发，遇到需要触发的相机可使用脚本使能触发信号。

📎camera_enable_545_2020_07_28.tar.gz

进入camera_enable_545



先执行./start.sh　给/dev/tztek_i2c_cpld_enable_trigger加权限

若需要给v1,v2 30fps的帧率， 则需要执行./enable_0_30_fps，./enable_1_30fps 再执行./enable_start_trigger_0_1。以此类推　v3,v4 则./enable_2_30_fps，./enable_3_30fps 再执行./enable_start_trigger_2_3。

2.enable_trigger_fps.c 和 enable_start_trigger.c可以修改设置

enable_trigger_fps.c 可以设置每一路相机的帧率参数

vim enable_trigger_fps.c

```c
int main(void)
{
        int fd, ret;
        struct ioctl_data time[] = {
                {0, 0},
                {1, 0},
                {2, 0},
                {3, 0},
                {4, 0},
                {5, 0},
                {6, 0},
                {7, 0},
        };

        fd = open("/dev/tztek_i2c_cpld_enable_trigger", O_RDWR);
        if(fd < 0)
        {
                printf("can not open file\n");
                return 0;
        }

        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_0_30_FPS, &time[0]);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_1_30_FPS, &time[1]);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_2_30_FPS, &time[2]);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_3_30_FPS, &time[3]);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_4_30_FPS, &time[4]);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_5_30_FPS, &time[5]);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_6_30_FPS, &time[6]);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_7_30_FPS, &time[7]);


        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_0_10_FPS, &time[0]);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_1_10_FPS, &time[1]);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_2_10_FPS, &time[2]);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_3_10_FPS, &time[3]);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_4_10_FPS, &time[4]);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_5_10_FPS, &time[5]);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_6_10_FPS, &time[6]);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_7_10_FPS, &time[7]);


        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_0_20_FPS, &time[0]);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_1_20_FPS, &time[1]);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_2_20_FPS, &time[2]);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_3_20_FPS, &time[3]);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_4_20_FPS, &time[4]);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_5_20_FPS, &time[5]);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_6_20_FPS, &time[6]);
        ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_7_20_FPS, &time[7]);

        return 0;
}
```

如果要设置20fps ，则把ioctl 20fps的打开，再使用gcc 编译即可。struct ioctl_data 结构体，包含id 和 time ，id是指第几路相机，time 表示这一路相机触发的偏移量，比如id = 0，time = 0，则表示v1触发偏移量为0。

enable_start_trigger.c 可以设置使能触发的路数

vim enable_start_trigger.c

```
#define TZTEK_CPLD_CAMERA_TRIGGER_1_STOP 0Xf7
#define TZTEK_CPLD_CAMERA_TRIGGER_2_STOP 0Xf8
#define TZTEK_CPLD_CAMERA_TRIGGER_3_STOP 0Xf9
#define TZTEK_CPLD_CAMERA_TRIGGER_4_STOP 0Xfa
#define TZTEK_CPLD_CAMERA_TRIGGER_5_STOP 0Xfb
#define TZTEK_CPLD_CAMERA_TRIGGER_6_STOP 0Xfc
#define TZTEK_CPLD_CAMERA_TRIGGER_7_STOP 0Xfd


struct ioctl_data_start
{
    unsigned int id[8];
};

int main(void)
{
        int fd, ret;
        struct ioctl_data_start start_trigger_id;
        start_trigger_id.id[0] =1;
        start_trigger_id.id[1] =1;
        start_trigger_id.id[2] =1;
        start_trigger_id.id[3] =1;
        start_trigger_id.id[4] =0;
        start_trigger_id.id[5] =0;
        start_trigger_id.id[6] =0;
        start_trigger_id.id[7] =0;

        fd = open("/dev/tztek_i2c_cpld_enable_trigger", O_RDWR);
        if(fd < 0)
        {
                printf("can not open file\n");
                return 0;
        }
        ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_multiple_START, &start_trigger_id);

        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_0_STOP, NULL);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_1_STOP, NULL);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_2_STOP, NULL);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_3_STOP, NULL);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_4_STOP, NULL);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_5_STOP, NULL);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_6_STOP, NULL);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_7_STOP, NULL);
        //ret = ioctl(fd, TZTEK_CPLD_CAMERA_TRIGGER_ALL_STOP, NULL);
        return 0;
}
```

如图中，想使能哪一路，只需要将那一路的id 置 1 即可。比如想要使能v1触发则置start_trigger_id[0] =1;

使能v2触发则置　start_trigger_id[1] =1;以此类推，设置好之后，gcc 编译即可。

NX-GEAC90(545r1.1)

📎trig_new_545_r1_1.tar.gz

545r1.1的cpld 协议改变不再用i2c,而使用串口，保持一致性

进入trig_new

```
wangjianyang@DELL:~/trig_new$
wangjianyang@DELL:~/trig_new$
wangjianyang@DELL:~/trig_new$ ls
545_r1_1_trigger_all_10_fps  545_r1_1_trigger_all_25_fps  include  Makefile  source
545_r1_1_trigger_all_20_fps  545_r1_1_trigger_all_30_fps  lib      obj
wangjianyang@DELL:~/trig_new$
wangjianyang@DELL:~/trig_new$
wangjianyang@DELL:~/trig_new$
```

如果需要10fps,则执行sudo ./545_r1_1_trigger_all_10_fps, 25,20,30fps依次类推。

```
wangjianyang@DELL:~/trig_new$
wangjianyang@DELL:~/trig_new$
wangjianyang@DELL:~/trig_new$
wangjianyang@DELL:~/trig_new$
wangjianyang@DELL:~/trig_new$ sudo ./545_r1_1_trigger_all_10_fps
```

相机多种工具显示图像

1.ffplay

ffplay 是通用的普通多媒体播放器，可播放实时的图像。

ffplay_ffmpeg.tar.gz

解压ffplay_ffmpeg.tar.gz，tar –xpf ffplay_ffmpeg.tar.gz 。

```
ffmpeg  ffplay  ffprobe  install.sh  libSDL2.so
```

执行sudo ./install.sh （拷贝库文件）

播放/dev/video0 ，则执行sudo ./ffplay /dev/video0

(若提示出现 没有libSDL2-2.0.so.0，则sudo mv /usr/lib/libSDL2.so /usr/lib/libSDL2-2.0.so.0 修改名字)

2.python-opencv

python_opencv_camera.tar.gz

解压python_opencv_camera.tar.gz ， tar –xpf python_opencv_camera.tar.gz 。

```
camera_test.py
```

解压内容为camera_test.py。

获取图像则，执行sudo python camera_test.py

默认播放video0，若要修改为其他通道，比如修改为 video1 可vim camera_test.py

```
import cv2

cap = cv2.VideoCapture(0)
while(1):
    # get a frame
    ret, frame = cap.read()
    # show a frame
    cv2.imshow("capture", frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
~
~
```

修改VideoCapture(0) 为 VideoCapture(1)。

退出播放，则按q键退出。