Banana Pi Camera module

Shenzhen SINOVOIP CO., LIMITED Copyright Statement:

More information:

Get more product and support please contact Shenzhen SINOVOIP CO.,LIMITED (www.banana-pi.com)

Attention:

Due to technical requirements components, please do not hand directly connected

Touch. Core board and development system contains static-sensitive devices. Quiet Electrical charge easily accumulate in the human body and the device can not detect possible Damage to equipment, it is recommended to take anti-static measures, it is recommended not to hand. Touch, stored in anti-static effect devices.

Product Specification:

High definition camera module for Banana Pi board via the CSI connector designed specifically for interfacing to cameras.Provide high sensitivity, low crosstalk and low noise image capture in a small and lightweight design.

Specifications

Image sensor	Omnivision 5640 CMOS image sensor in a auto-fouces modules with integral IR filter (650±10nm)
Still resolution	5 Megapixels
Active array size	2592×1944
Max frame rate	1080P 30fps@24Mhz
Picture formats	JPEG PNG YUV420 RGB888
Video formats	Raw h.264
Connection to Banana Pi	40 Pin FPC to the Camera Sensor Interface(CSI-0)
Image control functions	Automatic exposure control (AEC) Automatic white balance(AWB) Automatic black level calibration(ABLC) Automatic band filter Mirror and flip
Temp range	Operating: -30℃ to 70℃ Stable Image: 0℃ to 50℃
Lens size	1/4″
Dimension	36×32×10mm
I2C address	0x78
Weight	8g

Banana pi CSI Camera Connector

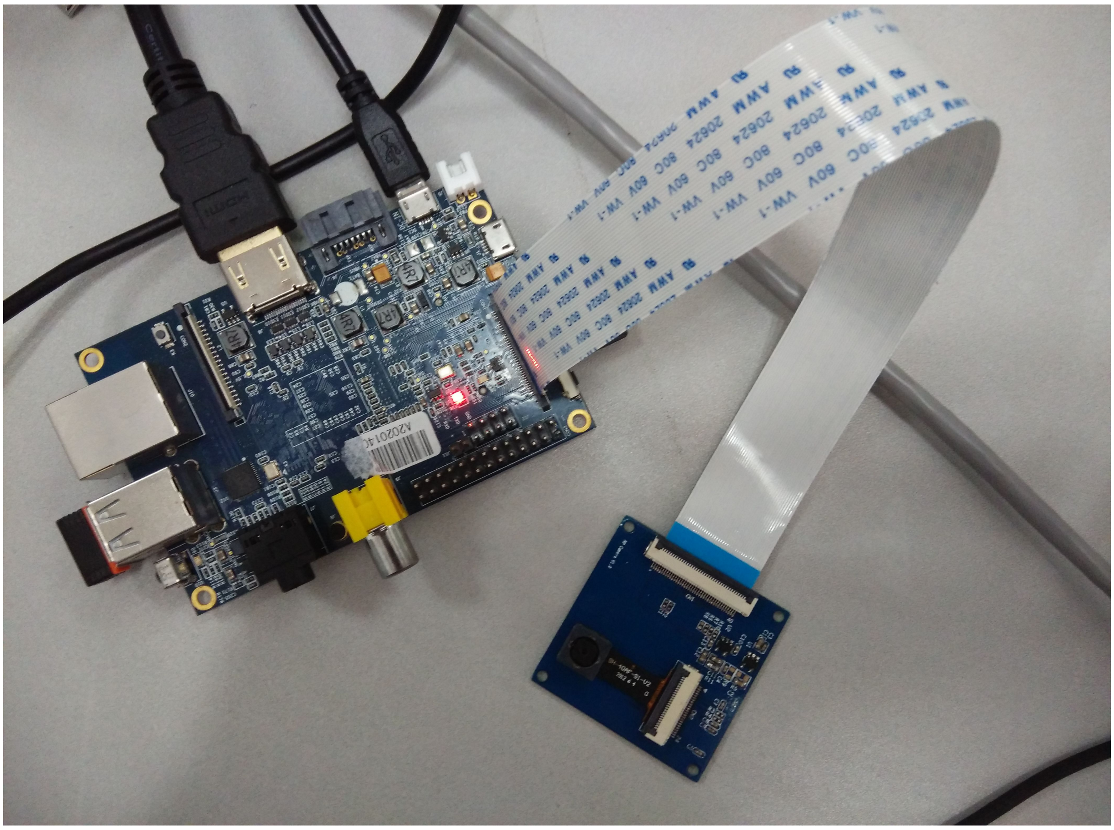
The CSI Camera Connector is a 40-pin FPC connector which can connect external camera module with proper signal pin mappings. The pin definitions of the CSI interface are shown as below. This is marked on the Banana Pi board as "CON1″.

CSI Pin	Pin Name	GPIO
CON1 P01	LINEINL	
CON1 P02	LINEINR	
CON1 P03	VCC-CSI	
CON1 P04	ADC_X1	
CON1 P05	GND	

CON1 P06	ADC_X2	
CON1 P07	FMINL	
CON1 P08	ADC_Y1	
CON1 P09	FMINR	
CON1 P10	ADC_Y2	
CON1 P11	GND	
CON1 P12	CSI-FLASH	PH17
CON1 P13	LRADC0	
CON1 P14	TWI1-SDA	PB19
CON1 P15	LRADC1	
CON1 P16	TWI1-SCK	PB18
CON1 P17	CSI-D0	PE4
CON1 P18	CSI0-STBY-EN	PH19
CON1 P19	CSI0-D1	PE5
CON1 P20	CSI-PCLK	PE0
CON1 P21	CSI-D2	PE6
CON1 P22	CSI0-PWR-EN	PH16
CON1 P23	CSI-D3	PE7
CON1 P24	CSI0-MCLK	PE1
CON1 P25	CSI-D4	PE8
CON1 P26	CSI0-RESET#	PH14
CON1 P27	CSI-D5	PE9
CON1 P28	CSI-VSYNC	PE3
CON1 P29	CSI-D6	PE10
CON1 P30	CSI-HSYNC	PE2
CON1 P31	CSI-D7	PE11
CON1 P32	CSI1-STBY-EN	PH18
CON1 P33	RESET#	
CON1 P34	CSI1-RESET#	PH13
CON1 P35	CSI-IO0	PH11
CON1 P36	HPR	
CON1 P37	HPL	
CON1 P38	IPSOUT	
CON1 P39	GND	
CON1 P40	IPSOUT	

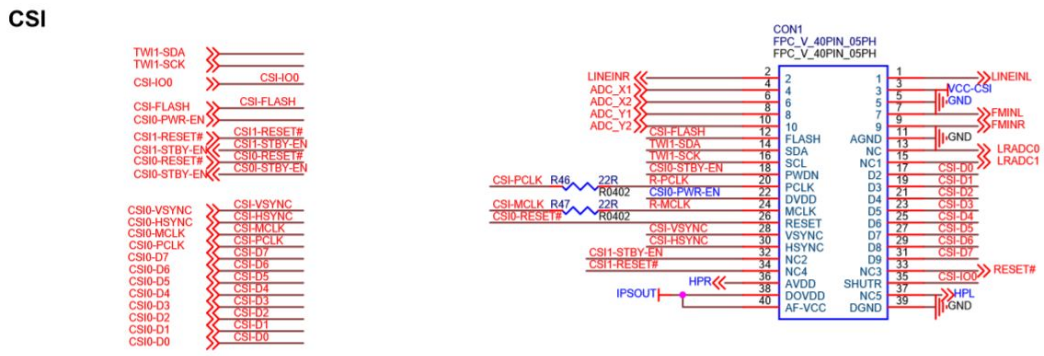
How to use:

This module can now be used under Android4.2.
First ,connect the module to CSI interface CON1 of Banana Pi. Then boot the BPI with Android4.2. Find the Camera App int the application menu and clicked.That's OK.



More information：

Schematic diagram：



Example and Test Code:

This Camera driver uses v4l2 to control the device. So when we want to test it, we can open and init the device like this.

```
static void open_device (void)
{
    struct stat st;
     if (-1 == stat (dev_name, &st)) {
    fprintf (stderr, "Cannot identify '%s': %d, %s/n",dev_name, errno, strerror (errno));
    exit (EXIT_FAILURE);
    }
    if (!S_ISCHR (st.st_mode)) {
    fprintf (stderr, "%s is no device/n", dev_name);
    exit (EXIT_FAILURE);
    }

    //open framebuffer
        fbfd = open("/dev/fb0", O_RDWR);
        if (fbfd==-1) {
            printf("Error: cannot open framebuffer device./n");
            exit (EXIT_FAILURE);
        }
```

```
    //open camera
    fd = open (dev_name, O_RDWR| O_NONBLOCK, 0);

    if (-1 == fd) {
    fprintf (stderr, "Cannot open '%s': %d, %s/n",dev_name, errno, strerror (errno));
    exit (EXIT_FAILURE);
    }
}


static void init_device (void)
{
    struct v4l2_capability cap;
    struct v4l2_cropcap cropcap;
    struct v4l2_crop crop;
    struct v4l2_format fmt;
    unsigned int min;


    // Get fixed screen information
      if (-1==xioctl(fbfd, FBIOGET_FSCREENINFO, &finfo)) {
            printf("Error reading fixed information./n");
            exit (EXIT_FAILURE);
        }

        // Get variable screen information
    if (-1==xioctl(fbfd, FBIOGET_VSCREENINFO, &vinfo)) {
            printf("Error reading variable information./n");
            exit (EXIT_FAILURE);
      }
    screensize = vinfo.xres * vinfo.yres * vinfo.bits_per_pixel / 8;


    if (-1 == xioctl (fd, VIDIOC_QUERYCAP, ∩)) {
        if (EINVAL == errno) {
            fprintf (stderr, "%s is no V4L2 device/n",dev_name);
            exit (EXIT_FAILURE);
        } else {
            errno_exit ("VIDIOC_QUERYCAP");
          }
    }
```

```
    if (!(cap.capabilities & V4L2_CAP_VIDEO_CAPTURE)) {
        fprintf (stderr, "%s is no video capture device/n",dev_name);
        exit (EXIT_FAILURE);
    }
    if (!(cap.capabilities & V4L2_CAP_STREAMING)) {
        fprintf (stderr, "%s does not support streaming i/o/n",dev_name);
        exit (EXIT_FAILURE);
    }

    CLEAR (cropcap);

    cropcap.type = V4L2_BUF_TYPE_VIDEO_CAPTURE;

    if (0 == xioctl (fd, VIDIOC_CROPCAP, &cropcap)) {
        crop.type = V4L2_BUF_TYPE_VIDEO_CAPTURE;
        crop.c = cropcap.defrect;

        if (-1 == xioctl (fd, VIDIOC_S_CROP, &crop)) {
            switch (errno) {
            case EINVAL:
            break;
            default:
            break;
            }
        }
    }else {     }

    CLEAR (fmt);

    fmt.type = V4L2_BUF_TYPE_VIDEO_CAPTURE;
    fmt.fmt.pix.width = 640;
    fmt.fmt.pix.height = 480;
    fmt.fmt.pix.pixelformat = V4L2_PIX_FMT_YUYV;
    fmt.fmt.pix.field = V4L2_FIELD_INTERLACED;

if (-1 == xioctl (fd, VIDIOC_S_FMT, &fmt))
        errno_exit ("VIDIOC_S_FMT");

    init_mmap ();

}
```