How to build affordable Panoramic Camera product with Bubblegum96

Bo Dong <bdong@ucrobotics.com>
● Brief introduction
● The goal of this project
● Software - Hardware Acceleration
● Hardware - Open hardware MIPI Mezzanine Board
● Performance Benchmark
Brief Introduction

Who am I?

- Bo Dong
- bdong@ucrobotics.com
- Software Engineer
- Community Manager
- Maker
Brief Introduction

Who is uCRobotics?

uCRobotics is a Hi-tech company which focuses on Intelligent Platform, System Integration, and the development of Embedded System, etc.

Intelligent Platform

Solution

OpenSource

uCRobotics is the manufacturer of Bubblegum-96 which is the 4th intelligence development platform based on the Linaro Standard.

3D cloud printing solution, Robotics solution, big data solution

Contribution in open source projects
Brief Introduction

What is Bubblegum96?

- Optimized Actions S900 quad-core ARM Cortex-A53 64-bit SoC up to 1.8GHz
- Embedded Imagination PowerVR™ G6230 GPU support OpenGL ES 3.1, OpenGL 3.2, OpenCL 1.2 EP
- Integrated HDMI with Ultra 4K output
- 802.11b/g/n WiFi, Bluetooth 4.0
- USB 3.0 (Type A) x1 & USB 2.0 (Type A) x1 & Micro USB 2.0 x 1
- Extend Header(x40 pin) +1.8V, +5V, SYS_DCIN, GND, UART, I2C, SPI, PCM, GPIO x12
- Extend Header(x60 pin) SDIO, MIPI_DSI, MIPI_CSI, I2C, USB 2.0
- Support Ubuntu Core & Android 5.1 Lollipop & Debian & Gentoo & OpenSUSE & Remix OS
- High performance, low latency and low cost
Goal of the project

- **Insta 360**: $239.92
- **Ricoh Theta V**: $396.99
- **Rylo 360**: $497.89
Goal of the project

Bubblegum96
$89

Camera Set
~ $30

< $120
Software

Base System

- Based on Debian 8 (Jessie)
- Imagination DDK support OpenCL
- OpenCV
- Linaro overlay
- OV5640 Driver
Software

Camera Driver Architecture

**Userspace**
- Applications
  - API (open, ioctl, close...)

**Kernel Space**
- V4L2
  - v4l2_file_operations
- soc-camera
  - soc_camera_host
  - soc_camera_device
- ISP driver
  - v4l2_subdev
- Module driver
  - v4l2_subdev

**Hardware**
- ISP
- OV5640
Software

**OV5640 Sensor Initialization Timing**

![Diagram of OV5640 sensor initialization timing](image)

- set_gpio_level(&spinfo->gpio_rear, GPIO_HIGH);
- set_gpio_level(&spinfo->gpio_rear_reset, GPIO_LOW);
- mdelay(100);
- set_gpio_level(&spinfo->gpio_rear, GPIO_LOW);
- mdelay(100);
- set_gpio_level(&spinfo->gpio_rear_reset, GPIO_HIGH);
- mdelay(500);

**Note:**
- \( t_0 \geq 0 \text{ms}, \text{delay from DOVDD stable to AVDD stable, it is recommended to power up AVDD shortly after DOVDD has been powered up} \)
- \( t_1 \geq 0 \text{ms}, \text{delay from XVCLK off to AVDD off} \)
- \( t_2 \geq 5 \text{ms}, \text{delay from AVDD stable to sensor power up stable, PWDN can be pulled low after this point, XVCLK can be turned on after power on} \)
- \( t_3 \geq 1 \text{ms}, \text{delay from sensor power up stable to RESETB pull up} \)
- \( t_4 \geq 20 \text{ms}, \text{delay from RESETB pull high to SCCB initialization} \)
- \( t_5 > 0 \text{ms}, \text{delay from AVDD off to DOVDD off} \)
- \( t_6 > 0 \text{ms}, \text{delay from RESETB pull low to AVDD off} \)
Software

Infrastructure

Application
Algorithm
QP Middleware
OpenCL
Debian

GPU Acceleration

General Interface
QP/C++ (Quantum Platform in C++) is a lightweight, open source software framework/RTOS for building reactive real-time embedded applications as systems of cooperating, event-driven active objects (actors). The QP/C++ framework is a member of a larger QP family consisting of QP/C, QP/C++, and QP-nano frameworks, which are all strictly quality controlled, thoroughly documented, and commercially licensable.
Software

QP State Machine

State Sequence for [Wakeup] and [Pause]
Software

GPU Acceleration

PowerVR Rogue G6230

- Compute Core: 64Core
- Freq: 528MHz
- Shared Memory: 4Kbyte
- Support: OpenGL ES 3.1, OpenGL 3.2, OpenCL 1.2 EP
Software

OpenCL Acceleration

Discover and initialize the platform
clGetPlatformIDs()

Discover and initialize the devices
clGetDeviceIDs()

Create a context
clCreateContext()

Create device buffers
clCreateBuffer()

Write host data to device buffers
clEnqueueWriteBuffer()

Create and compile the program

Create the kernel
clCreateKernel()

Set the kernel arguments
clSetKernelArg()

Configure the work-item structure

Enqueue the kernel for execution
clEnqueueNDRangerKernel()

Read the output buffer back to the host
clEnqueueReadBuffer()

Release OpenCL resources

Create a command queue
clCreateCommnadQueue()
1. Correct Fisheye distortion by spherical projection.
2. Perform the inverse mapping.
Software

Panoramic Stiching

Ref: http://www.360facil.com/eng/360-degree-photo-shot-fisheye-two-shots.php
Hardware

Dual MIPI Cam Schematic
Hardware

Three reversions of PCB
Hardware

Raw capture from module
Hardware

Calibration

Fisheye camera model

{opencv_fisheye}

https://docs.opencv.org/trunk/db/d58/group__calib3d__fisheye.html
Performance Benchmark

Benchmark

GPU

CPU

8X
Thanks!

Bo Dong <bdong@ucrobotics.com>
Bubblegum96 <96boards@ucrobotics.com>