Pupscan Imaging Requirements

Firmware features

The firmware will do the image capture and **FIFO queuing** into
- the RAM memory (if we use a microprocessor),
- the Flash memory (if microcontroller with too few RAM)

The firmware should perform :
- Auto-focus
- Auto-white-balance (can be disabled).

The **white balance** should be either:
- automatically handled by the Firmware (JPG format case)
- disabled : will be post processed in the Sofware (RAW format case)

The **focus** should:
- use a well known minimizing algorithm (starting converging from lowest focal distance)
- be faster than 500ms

The firmware **capture command** should have these three parameters :
- **exposure** duration : {duration},
- image **format** : {[JPG, quality value]} or {RAW}
- image **resolution** : {[width, height]}

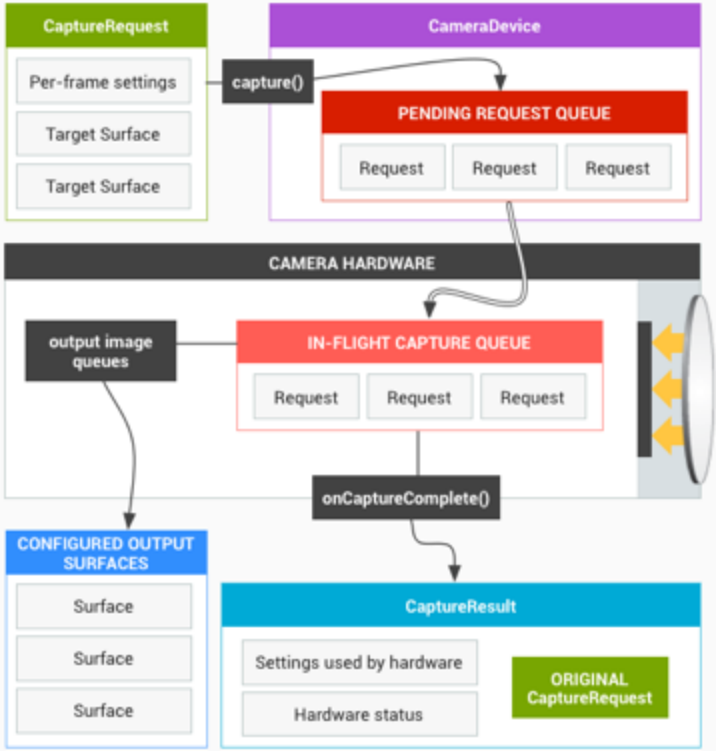
The **firmware** will **read** capture requests from a queue and **write** images in another queue

The **software** will **write** capture requests to a queue and **read** images from another queue

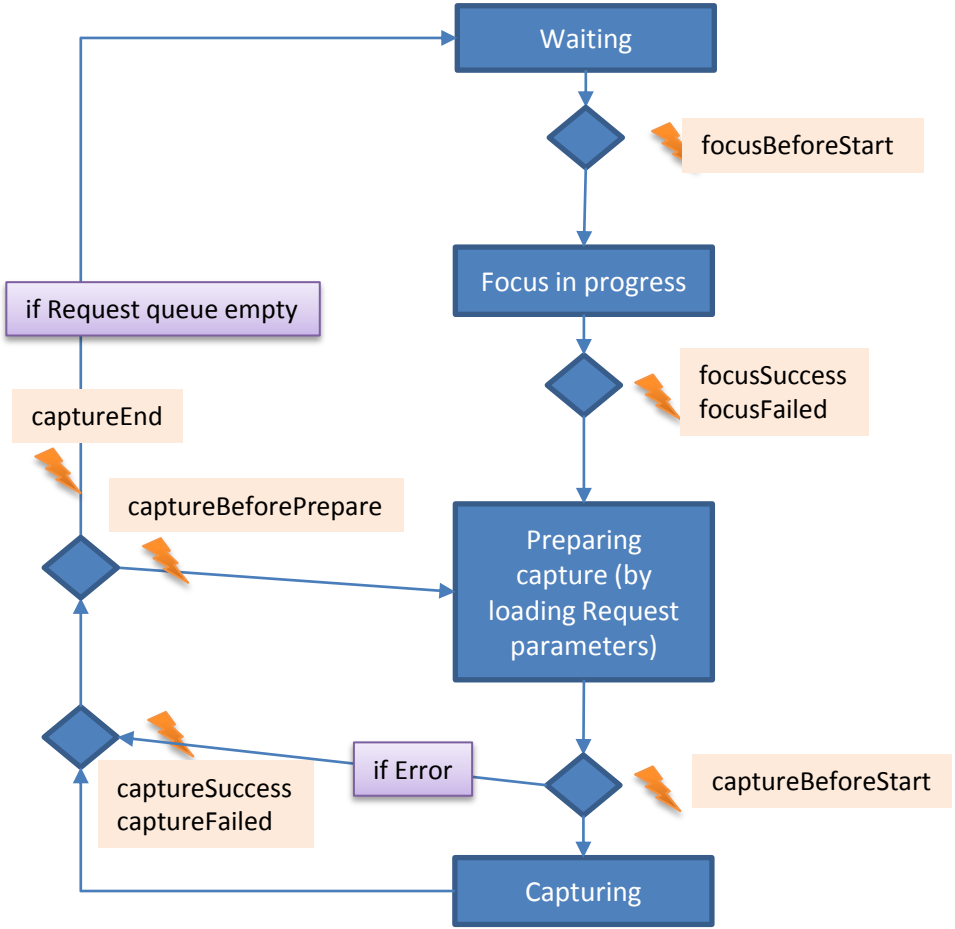
For the Firmware/Software interface we REALLY WANT to take our inspiration from the Android Camera2 API (see next slides). See my Proposal inspired by this API.

Android Camera2 API (on top of Hal3)

https://source.android.com/devices/camera/camera3.html
https://source.android.com/devices/camera/camera3_3Amodes.html



Firmware/Software interface proposal




```
void capture(List<Request>, CaptureCallback)
interface Request{
                    Resolution,
                    ExposureDuration,
                    ImageFormat
}
interface CaptureCallback{
                    focusBeforeStart(){
                                        //... turn on flash...
                    }
                    focusSuccess(){
                                        //... user feedback ? ...
                    }
                    focusFailed(){
                                        //... something wrong...
                    }
                    captureBeforePrepare(Request) {
                                        //... cancel the request?
                    }
                    captureBeforeStart(Request){
                                         //... turn off laser ...
                    }
                    captureSuccess(Request, Response, Image){
                                        //... do something with Image and meta data
                    }
                    captureFailed(Request, Response, Error){
                                        //... handle the error
                    }
}
```

Firmware/Software interface proposal

Request Queue
FIFO queue for capture requests

Camera
Settings (per Request)
- exposure
- resolution
- format (RAW, JPG)

Trigger events
- focusBeforeStart
- focusSuccess
- focusFailed
- captureBeforePrepare
- captureBeforeStart
- captureSuccess
- captureFailed

Image Queue
FIFO queue for image content together with original request and metadata
Discard first when overflow

Flash
Driven by software

Laser
Driver by software

The **firmware** is consuming the Request Queue. A new Request is loaded, the camera is configured according to the request. The Capture process is started. Some event are thrown so the software can deal with Flash and Laser. The captured Image is stored into the Image Queue so the software can read it later.
The **software** is producing the request into the RequestQueue. He is handling the events using the Callback object. He is responsible of tuning the Laser and Flash intensity. When the image is captured, the software is processing it.

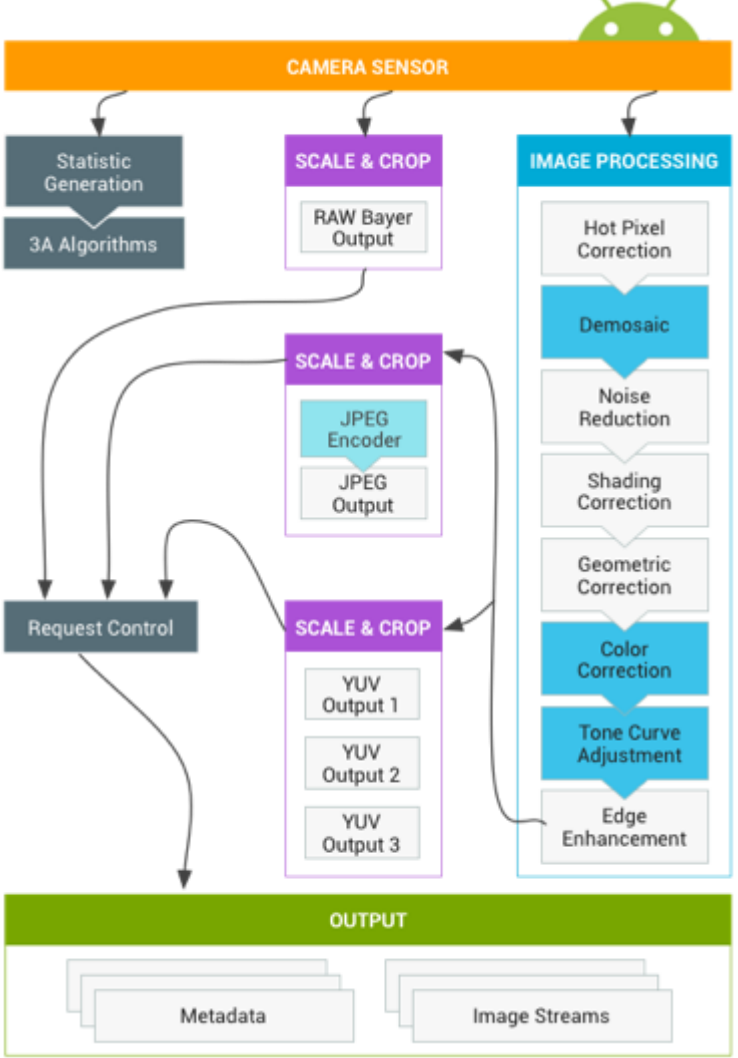
Optical & Imaging Requirements

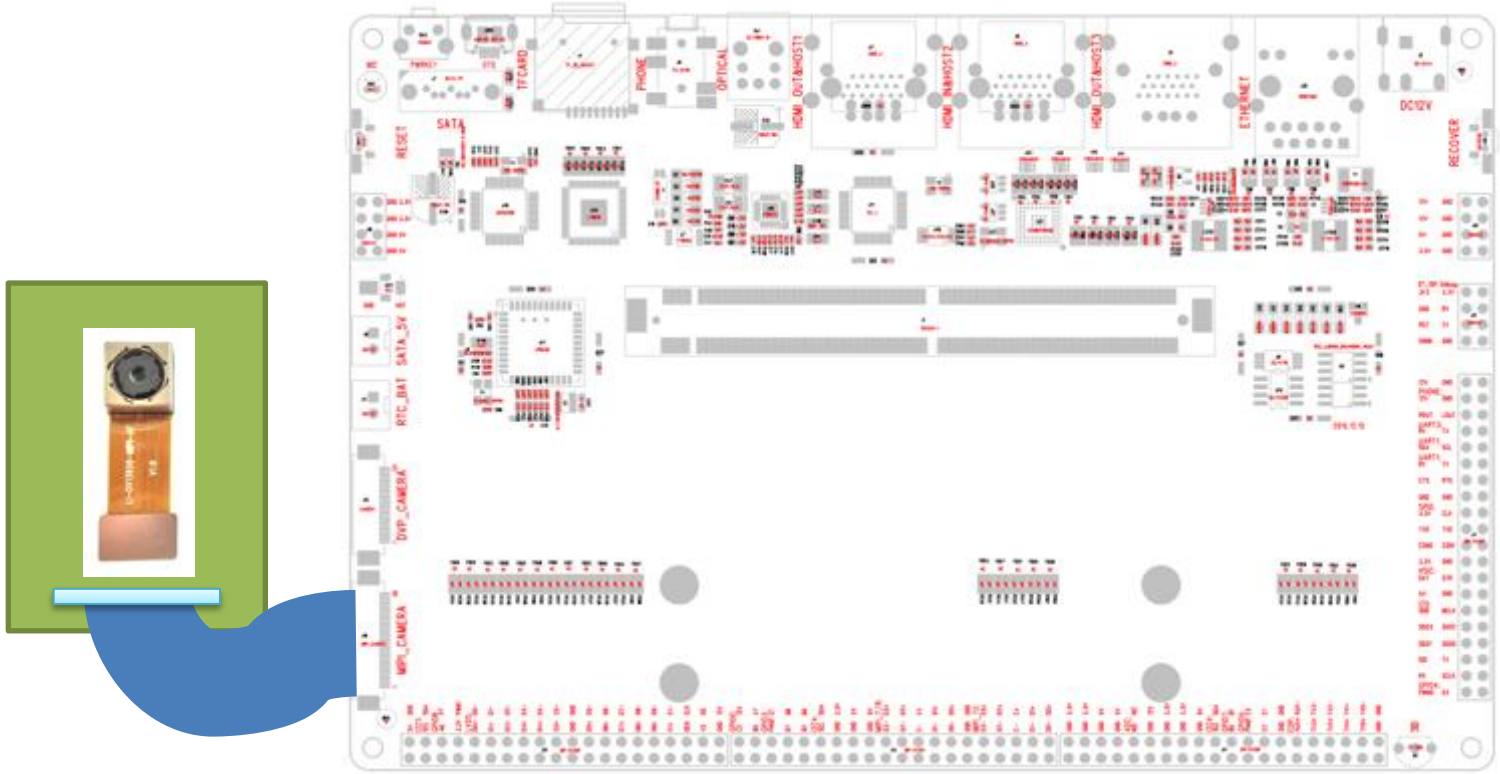
1. Auto-crop (software)
The required function is to use the laser window to indicate where to crop the image
2. Auto-rotate (software)
The required function is to use the top hat button is to indicate page orientation.
3. Auto-compress (software if RAW, firmware if JPG)
The required function is to use the distance sensor to optimize image size and compression
4. Anti-reflection(optical or software)
The required function is to use some filters (such as polarizing filters) to avoid having a ghost image of the flash itself.
5. Anti-burn lighting(optical or software HDR-like)
The required function is to get a very homogeneous light over the captured area.
6.Curve fixing TBD
This is a hard but key point, the required function is to get a good looking image for curved papers.
7.Resolution
The required function is to be able to capture any size of document for A3 to A6 (nice to have A8). When capture a A3 paper, Arial size 6 letter is readable.
8.Sharpness control (software)
The required function is to use near-realtime (<500ms) sharpness analysis to check the validity of the captured image. If not good, the image is rejected and the device is notifying the user by vibrating.

Processor features

Our application is document scanning :
•small distance capture < 50cm
•well defined lighting (we expect small distance flash to lead to stable homogeneous lighting, quite independent from ambiant lights)
•highest available resolution
•low frame rate (except maybe if we need HDR : same scene with different exposures)
•no real time display

We want to shorten the time to market so the most image processing ready to go would be our best choice :
<u>3A units for converging algorithms</u>
•auto focus processing unit
•auto exposure processing unit
•auto white processing unit
<u>Post processing</u>
•hot pixel correction
•demosaic
•noise reduction
•shading correction
•geometric correction
•color correction
•tone curve ajustment
•edge enhancement
•jpeg compression




TFCARD
PHONE
OPTICAL
HDMI_OUT&HOST1
HDMI_IN&HOST2
HDMI_OUT&HOST3
ETHERNET
DC12V
RECOVER
RESET
SATA
SATA_5V
RTC_BAT
DVP_CAMERA
MIPI_CAMERA