CryptoCam

(Cryptograph Camera)

Team 17 – Jihoon Lee, Sameeth Gosike, and Sang Baek Han
ECE 445 Design Review – Spring 2017
TA: Jose Sanchez Vicarte

1 Introduction

1.1 Objective

Taking the photograph using a camera has become such an easy task nowadays. As the technology improved, more people use smartphones to take the photograph rather than carrying extra device that can only take the photograph. As for the smartphone, they do have the encrypted software available freely and reliably. However, the professional camera used by photojournalists and anyone related to that field still do not have access to any type of encryption. The encryption is necessary for the professional camera so that it provides the immediate protection from unexpected damage, lost, or robbery of camera [1].

Our solution to this problem is to use the additional hardware components between camera data bus and memory storage so that when the photo is taken, the photo will immediately get encrypted and then stored. Only the authorized user who has the access to this additional hardware with the right key can see the decrypted image.

1.2 Background

Last year December, Freedom of the Press Foundation had published an open letter to the world-leading camera manufacturers such as Nikon, Sony, Canon, Olympus, and Fuji to build a camera with the encryption features for the sake of filmmakers and photojournalists. Those filmmakers and photojournalists are occasionally exposed to dangerous subjects which lead to the threat from the authoritarian governments or criminals. With their camera full of unencrypted images, they have no way to protect themselves from such risk [1].

1.3 High-Level Requirements

- Timing Requirements – For benchmarking, we did simple simulation using Arduino with ATmega2560 8-bit microprocessor [2]. For encryption process, the maximum operation took 3 addition/subtraction and 1 multiplication/division. Using this, we tested with 1920x1080 resolution image and got average time of 3 seconds. For initialization process of producing Chaotic Map, we got average time of 7 minutes. Since the microprocessor on Arduino operates at 16MHz while the microprocessor we are using operates at 200MHz, we are using this simulation result as the worst-case maximum time.
- Encryption complexity requirement – Success of the encryption relies on how different the encrypted picture is from the original data. Correlation coefficient will be used as a quantitative measure of the difference, which determines the similarity between values of two adjacent pixels. Most chaotic map schemes online attained (< 0.002) of coefficient magnitude on success. We consider this as a same requirement for a successful encryption.

- Lossless Encryption - 100% image recovery rate. We want our algorithm to be lossless, as are most image encryption algorithms are. It would be a huge disadvantage if algorithm poses a risk of damage in quality.
- Power constraint – Typical high-quality camera batteries enable taking up to 500 pictures before running out. With 3 seconds of worst case encryption time and 7 minutes (420 seconds) of mapping time, we will need about an hour for the encryption of 500 pictures. And we want our camera to be able to run for entire day (12 hours). As an attachment to camera battery, the size of 4 of 3.7v, 3200mAh battery will be suitable to go in same space and ensure good lifetime. With capacity of 12800mAh, we want to run camera for total of 15 idle hours (taking encrypting time as 3 hours of equivalent consumption in idle mode due to shifting to operating voltage), we want average power consumption to be no more than

$$P_{average} = 3.7V \times 12800mAh = 3.157W \quad .$$

2 Design

The design makes acquisition of data from camera module and utilizes data processing unit to encrypt the data. The usage of SDRAM helps the achievement of timing requirement due to its faster speed compared to other species of memories.

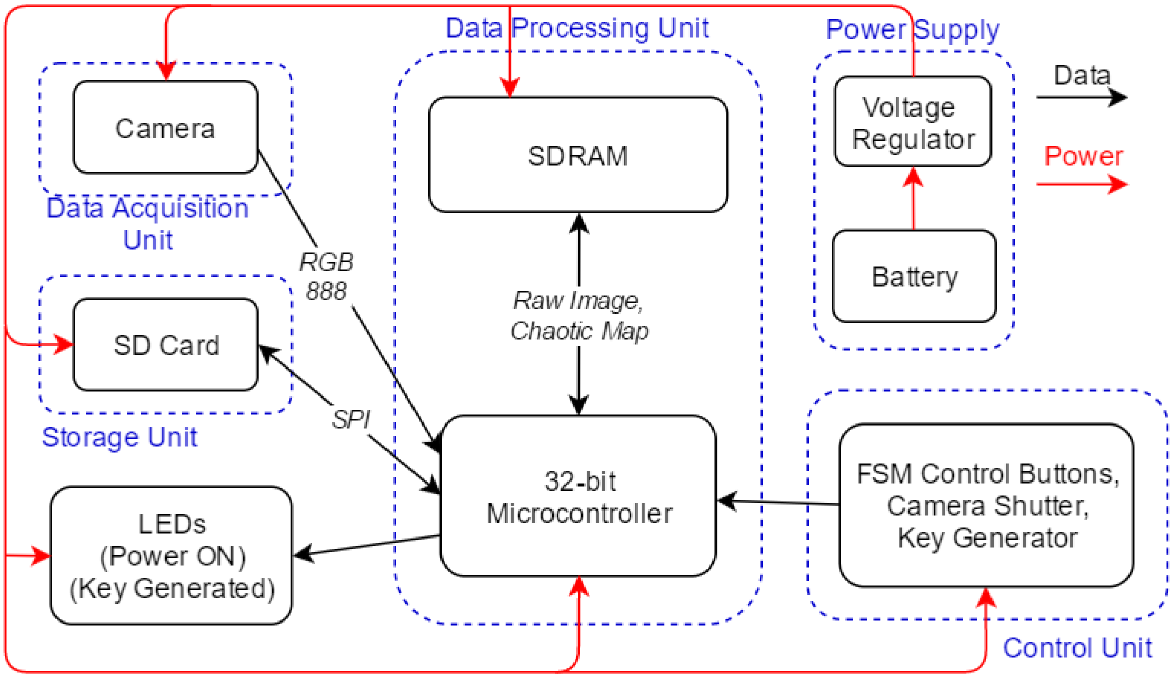


Figure 1 – Block Diagram

2.1 Data acquisition Unit

Data acquisition unit is composed of a single camera, which uses light sensitive receptor array to acquire optical input in RGB888 format. RGB888 will be sent to microprocessor then stored in SDRAM. Raw image will be encrypted with Chaotic Map. The resolution will be 1920x1080 to meet the typical resolution of DSLR camera [4].

2.2 Data Processing Unit

SDRAM: **AS4C64M32MD1-5BIN (Alliance Memory)**

- Big memory capacity (256MB), enough to store 32 images without flushing memory
- Low dynamic power consumption (Max Supply voltage: 1.9V, Maximum current:190mA)
- Fast operating frequency (200Mhz) resulting in low memory access time (5ns). It even excels many of its SRAM counterparts.
- It is cheap (a bit over $10) and takes only one chip to store multiple pictures. Some SRAM counterparts can provide faster access time (0.45ns). But their price with same memory capacity (256MB) goes beyond few thousands).

Microcontroller: **TMS320F2837xD Dual-Core Delfino (Texas Instruemnt)**

- Low operating voltage (1.2V) helps achieving power requirement
- High Maximum operating frequency (200Mhz) helps achieving timing requirement.
- Good amount of Flash memory (1MB) will allow us to store additional algorithms.
- High temperature tolerance (max 125 degrees Celcius)
- Cons: Expensive price ($36~40). PIC processors with similar performance costs around 10 dollars.

Data processing unit is centered on a 32-bit microcontroller. A 256MB SDRAM is used to store pixel data and the variables that are needed to process the whole pixel data. One image will be 6MB as shown from the calculation below.

$$Image\ Size = \frac{1920\ pixels \times 1080\ pixels \times 3\ byts\ per\ pixel}{1024 \times 1024\ bytes\ per\ Megabytes} \approx 5.93MB$$

The data processing unit will implement an image encryption based on chaotic logistical maps. Traditional encryption methods such as AES, IDEA, DES, and RSA are computationally complex and slow [5]. Chaotic map based encryption will be better suited to professional camera use in which high speeds, close to real-time are needed. In addition, the microcontrollers that are used in this design will be able to handle such algorithms in terms of speed and computational power. The level of security of these algorithms is not a concern as they have been proven to have high success rates in numerous studies.

Chaotic map will be generated using 80-bit Chaotic key when the Key Generator button is pressed or when the camera turns off. Chaotic map will be stored in SDRAM even in sleep-mode. Whenever the raw image is taken from the camera module, the raw image will be stored in SDRAM. Encryption will be applied on the raw image once the camera module stops sending the raw image data.

The requirement for this unit is to dissipate less than 2.2W of power at fully functional state. (maximum dynamic power) to satisfy power requirement. In addition the SDRAM access time must maximally be around 5ns in order to operate at the microcontroller's operating frequency of 200MHz.

2.2.1 Chaos Map based Image Encryption Algorithm

The algorithm used will start with an 80-bit pseudo random generated key, divided into ten session keys consisting of 8 bits each:

$$K = K_1K_2 \dots K_{10}\ (ASCII)$$

Two logistic chaotic maps are employed based on the following:

$$X_{n+1} = 3.999X_n(1 - X_n)$$

$$Y_{n+1} = 3.999Y_n(1 - Y_n)$$

In order to calculate the initial condition, X_0, three blocks of session keys are chosen i.e. $K_4K_5K_6$ and converted into a bit stream:

$$B_1 = K_{41}K_{42}..K_{48}K_{51} \dots K_{58}K_{61} \dots K_{68}$$

So then the value X_{01} is calculated as:

$$X_{01} = (K_{41} * 2^0 + K_{42} * 2^1 + \dots + K_{48} * 2^7$$

$$+ K_{51} * 2^8 + K_{52} * 2^9 + \dots + K_{58} * 2^{15}$$

$$^+ K_{61} * 2^{16} + K_{62} * 2^{17} + \dots + K_{68} * 2^{23}) / 2^{24}$$

Further, X_{02} can be calculated as:

$$X_{02} = \sum_{i=13}^{18} \frac{(k_i)}{96}$$

in which the k_i's are part of the key in hexadecimal mode. Finally X_0 can be calculated as:

$$X_0 = X_{01} + X_{02}$$

A 24 digit long real number sequence is generated using the initial condition, X_0, and the chaotic map formula $f_1, f_2 \dots f_{24}$. However only values in the range [0.1,0.9] are retained, and the rest are discarded. The real number sequence is converted into an integer number sequence such that,

$$P_k = int(\frac{23*(f_k - 0.1)}{0.8}) + 1, \text{k} = 1,2 \dots 24$$

To calculate the initial condition Y_0, three blocks of session keys are chosen i.e. $K_1K_2K_3$ and converted to a bit stream similar to the above B_2. Y_{01} is computed as:

$$Y_{01} = \frac{B_2}{2^{24}}$$

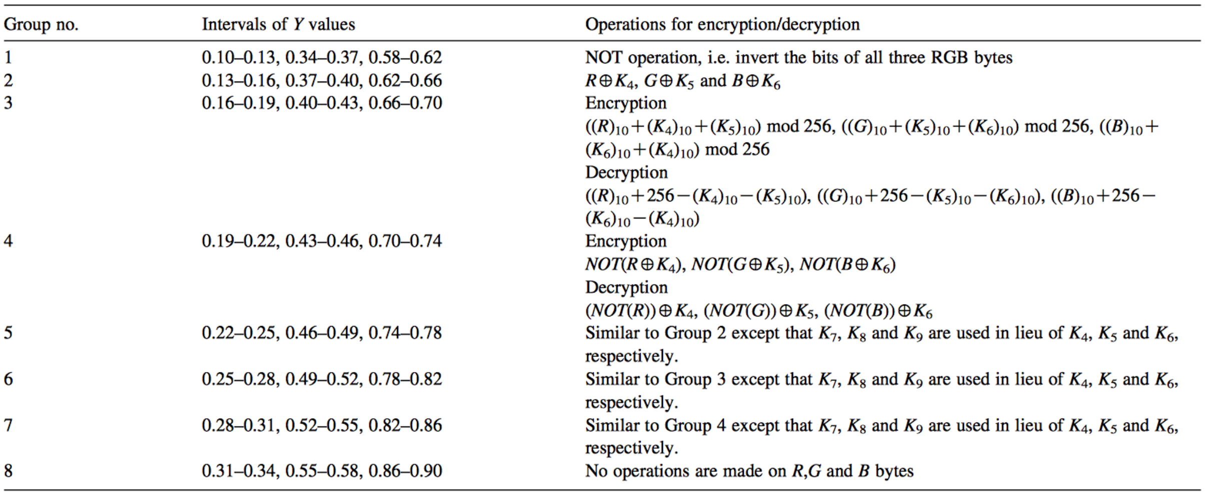
Further, Y_{02} is calculated as

$$Y_{02} = \frac{\sum_{k=1}^{24} B_2[P_k] * 2^{k-1}}{2^{24}}$$

Note $B_2[P_k]$ is derefercing the P_kth bit in the binary sequence B_2. Y_0 is calculated as

$$Y_0 = Y_{01} + Y_{02}$$

The final chaos map used for the pixel manipulation is calculated by the initial condition Y_0 and the formula mentioned above, iterating for as many pixels that exist in the image. A pixel's RGB values are read in (three consecutive bytes) and the value of it's corresponding Y_n will determine the operation done on the pixel, according to the following table:



Group no.	Intervals of Y values	Operations for encryption/decryption
1	0.10–0.13, 0.34–0.37, 0.58–0.62	NOT operation, i.e. invert the bits of all three RGB bytes
2	0.13–0.16, 0.37–0.40, 0.62–0.66	$R \oplus K_4$, $G \oplus K_5$ and $B \oplus K_6$
3	0.16–0.19, 0.40–0.43, 0.66–0.70	Encryption $((R)_{10}+(K_4)_{10}+(K_5)_{10})$ mod 256, $((G)_{10}+(K_5)_{10}+(K_6)_{10})$ mod 256, $((B)_{10}+(K_6)_{10}+(K_4)_{10})$ mod 256 Decryption $((R)_{10}+256-(K_4)_{10}-(K_5)_{10})$, $((G)_{10}+256-(K_5)_{10}-(K_6)_{10})$, $((B)_{10}+256-(K_6)_{10}-(K_4)_{10})$
4	0.19–0.22, 0.43–0.46, 0.70–0.74	Encryption $NOT(R \oplus K_4)$, $NOT(G \oplus K_5)$, $NOT(B \oplus K_6)$ Decryption $(NOT(R)) \oplus K_4$, $(NOT(G)) \oplus K_5$, $(NOT(B)) \oplus K_6$
5	0.22–0.25, 0.46–0.49, 0.74–0.78	Similar to Group 2 except that K_7, K_8 and K_9 are used in lieu of K_4, K_5 and K_6, respectively.
6	0.25–0.28, 0.49–0.52, 0.78–0.82	Similar to Group 3 except that K_7, K_8 and K_9 are used in lieu of K_4, K_5 and K_6, respectively.
7	0.28–0.31, 0.52–0.55, 0.82–0.86	Similar to Group 4 except that K_7, K_8 and K_9 are used in lieu of K_4, K_5 and K_6, respectively.
8	0.31–0.34, 0.55–0.58, 0.86–0.90	No operations are made on R,G and B bytes

Figure 2 – Comparison Table

Possible microcontroller optimizations can include processing several pixels at a time in a parallel fashion, as well as converting the Y chaos map values into integers. Doing so will make the comparative operations when determining the pixel's operation group less computationally complex if the same was done on a floating point number.

2.3 Control unit

Control unit is responsible for controlling FSM state between key acquisition phase, picturing phase, encryption phase, and reset phase. An additional shutter button will be used to signal the data processing unit to acquire data from acquisition unit.

When Key Generator button is pressed, new Chaotic key of 80 bits will be randomly generated. Using this new Chaotic key, Chaotic map will be generated and stored in partially assigned space of SDRAM. It will remain there until the Key Generator button is pressed again. While the map is processing (indicated by LED), the other buttons will be disabled until the process ends.

When the power button is pressed to turn off the camera, raw image in SDRAM will be force-erased to prevent the chance of hacking into SDRAM to retrieve the non-encrypted images. New

Chaotic map will be generated from new key and SDRAM will remained power on to keep this new Chaotic map.

Control unit should be able to de-bounce abrupt fluctuations in signals from buttons. Otherwise, processor will be loaded with more instructions to run than needed, affecting our power requirement.

2.4 Storage Unit

The Storage unit is responsible for the storage of the encrypted images after they are processed by data processing unit. SD card will store 3 files per pictures: encrypted image, encrypted chaotic key, and public key. User will need to remember the private key to decrypt the image.

The storage capacity will be on the lower side (8GB) since the main goal of the project does not include storing a large number of pictures. However, in order to read/write to the memory device from and to the microcontroller, a separate SD module will have to be included in between. This will be much more successful than trying to create a protocol in the microcontroller.

2.5 Power Supply Unit

Dynamic power consumption from microcontroller and SDRAM will be the main power drain, but datasheets have little information about internal load capacitance. Utilizing information given (operating voltage, maximum current),

These figures include power dissipated in voltage regulator.

$$P_{max}(Microprocessor, operating) = 3.7V \times 440mA = 1.628W$$

$$P_{max}(Microprocessor, idle) = 3.7V \times 210mA = 0.777W$$

$$P_{max}(SDRAM, operating) = 3.7V \times 190mA = 0.703W$$

$$P_{max}(SDRAM, idel) = 3.7V \times 70mA = 0.259W$$

$$P_{max}(SD\ card, operating) = 3.7V \times 100mA = 0.370W$$

We will use 4 3.7V, 3200mAh Li-ion battery for power supply due to its high capacity compared to other battery types. The voltage regulator will adjust voltage levels to each components. We are going to use TPA7A87 (Texas instrument) linear regulator due to its high current output capacity.

2.6 Requirements and Verification

<u>Power Supply [15 points]</u>

Requirements	Verification
Voltage regulator must be able to deliver different supply voltage to different components • Microcontroller: 1.2V ± 0.1V at 325mA • SDRAM: 1.7V ~ 1.95V at 190mA • SDcard: 2.7~3.6V at 100mA	Probe voltage from VDD and GND pins of each components using multimeter over time. If any value outside of allowed range is detected, it is flawed.

Li-ion Battery: 3.7V ± 10% voltage supply at 0.730A.	Probe both voltage and current over the Li-ion battery with multimeter and oscilloscope. Detect fluctuation outside of defined range.

Microcontroller[25points]

Requirements	**Verification**
Must be able to calculate chaotic map in less than 2 minutes with no accuracy tolerance.	Implement the same software on computer, and export the data in SD card to computer. Check data validity through comparing, and estimate the time difference through number of clock ticks * frequency.
Must be able to encrypt 1920*1080 image in 3 seconds with < 1% mismatch.	Implement the same encryption on computer, and export the data in SD card to computer. Check data validity through comparing, and estimate the time difference through number of clock ticks * processor frequency

SDRAM [15 points]

Requirements	**Verification**
Must be able to communicate with microcontroller at 200Mhz with data loss of < 1%.	After an attempt to write one image to SDRAM through microcontroller, turn off the power on the module and connect SDRAM to Arduino and its clock to receive image. Compare the received image with original image to check validity.

Algorithm [25 points]

Requirements	**Verification**
Encrypted image displays less than 0.002 correlation coefficient.	Export the image to the computer, and run MATLAB functions that shows the correlation coefficiency data.
Decrypted image matches the original image with less than 1% mismatch	Export the image to PC, and run comparison program ('diff' in Linux).

Status LEDs & Buttons [20 points]

Requirements	**Verification**
LEDs must accurately show state transition in FSM . (no tolerance applicable)	Write a separate program to communicate with Arduino to trigger state transition before using button. When single transition succeeds, trigger multiple state transition routine.
Buttons should be debounced to register only a single input per press	Probe the microcontroller's pin connected to the buttons and display them in oscilloscope.
Buttons must be able to trigger their corresponding FSM transition	Write a separate routine that store stream of stream transition log onto SD card for test purpose. Plug SD card to PC and verify the sequence.

2.7 Software Flow Charts

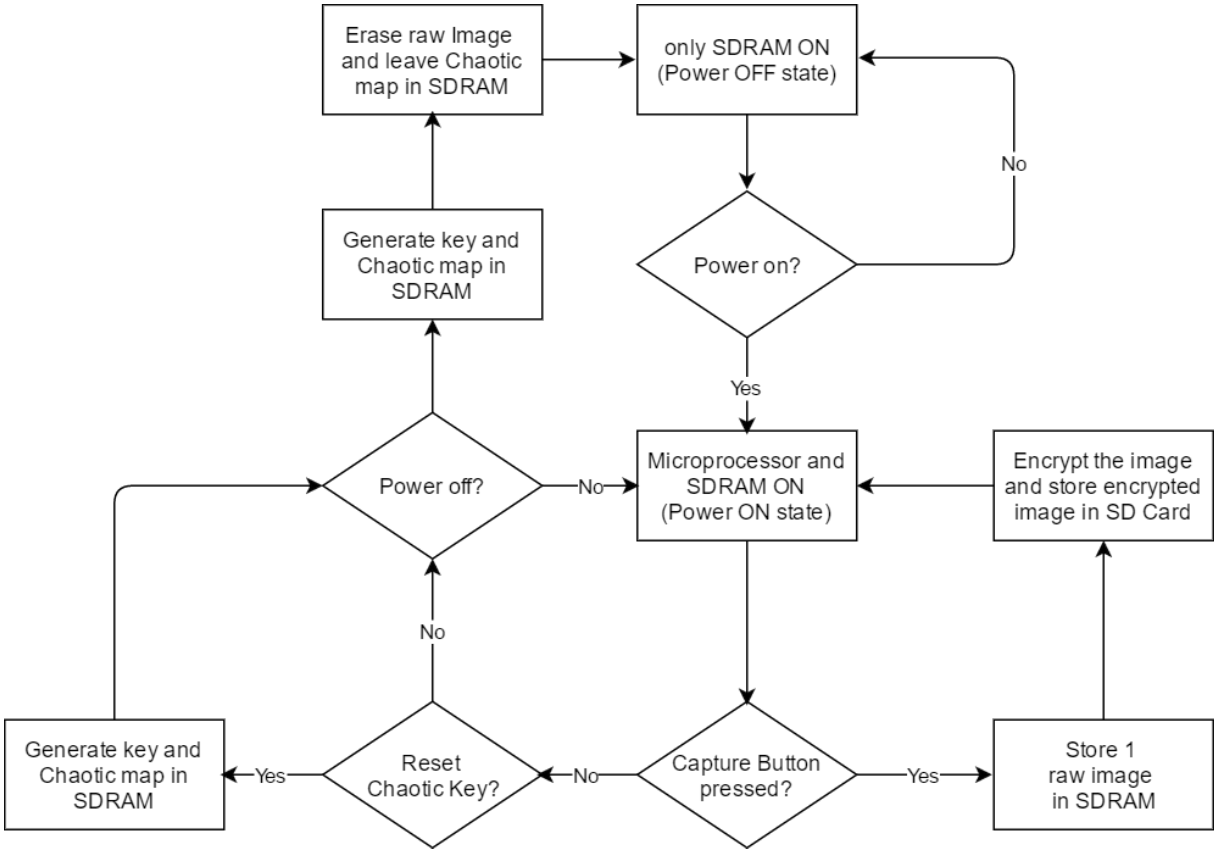

Erase raw Image and leave Chaotic map in SDRAM
only SDRAM ON (Power OFF state)
No
Power on?
Yes
Generate key and Chaotic map in SDRAM
Power off?
No
Microprocessor and SDRAM ON (Power ON state)
Encrypt the image and store encrypted image in SD Card
No
Generate key and Chaotic map in SDRAM
Yes
Reset Chaotic Key?
No
Capture Button pressed?
Yes
Store 1 raw image in SDRAM

Figure 3 – System Flowchart

2.8 Circuit Schematic

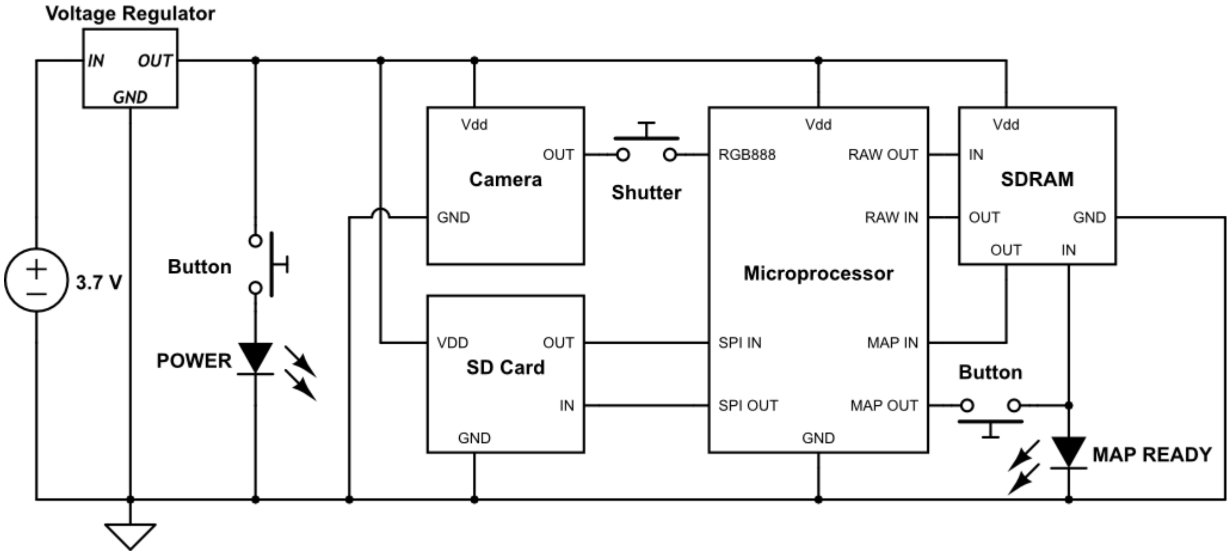


Figure 4 – Overall Circuit Schematic

2.9 Risk Analysis

The biggest risk lies in the interface between data acquisition unit and data processing unit. While input clock to the processors and camera can be synchronized, the output pixel clock of camera may not be precisely synchronized with the input clock. We will be scaling input clock inside camera so that it runs slower than the input clock, and using multiple-cycle instructions to get data.

3 Cost and Schedule

3.1 Cost Analysis

<u>LABOR</u>: We will assume to work 4hrs per day for every 5 days of week. We got 8 weeks till the actual demo. We will assume that we are working 160 hrs for this project. The minimum wages of \$7.25 will be used for the calculation.

$$Reasonable\ Salary = \$7.25\ per\ hrs \times 2.5 \times 160hrs = \$2900$$

Since we got 3 members in our group, total labor is \$8700.

<u>PARTS:</u>

SDRAM: AS4C64M32MD1-5BIN x 1	\$13.40
Microcontroller: TMS320F2837xD Dual-Core Delfino x 1	\$24.69

<u>TOTAL:</u> \$8738.09

3.2 Schedule

DATE	TASKS
Week of 2/27	Buy and gather all the materials we need and get use to the microprocessor.
Week of 3/6	Design PCB to connect all the parts. Start assembling the parts.
Week of 3/13	Finish assembling all the parts. Program the algorithm for microprocessor.
Week of 3/20 (Spring break)	Simulate the algorithm on Arduino. Simulate the algorithm on microprocessor.
Week of 3/27	Prepare mock-up
Week of 4/3	Integrate prototype.
Week of 4/10	Refine Prototype and test integrated system.

4 Ethics and Safety

There are potential ethical and safety issues regarding to our project. One ethical issue is whether the photo taken does deserve to be encrypted and remained unknown for non-authorized user. As for example, secret photography is the most concern as it can be used for stalking, paparazzi, and hidden camera inside the room. If ever those photographers got caught, then it will be hard to get the evidence out of encrypted image when the access key is broken or purposely damaged.

One way to solve the above issue is to provide the alternative method to decrypt the image. But, at the same time, it raises the issue on the reliability of our encryption and decryption system. The users expect that our decryption is only allowed with one way so they can safely take the photos without being exposed to the risk of getting caught. If there is some way to recover the encrypted image and used as an evidence, then there is no point of using our camera. We will positively assume that this project will be used by modest users, who accept the responsibility as mentioned in #1 of the IEEE Code of Ethics, “to accept responsibility in making decisions consistent with the safety, health, and welfare of the public, and to disclose promptly factors that might endanger the public or the environment” [6].

Potential safety concern we have is Li-ion battery. Since we are constantly providing power for SDRAM even in sleep-mode, the battery might get overheated if the camera is stored in a bag or heated place. We do not expect the battery to explode or fail at room temperature. If ever stored in a bag, it would be recommened for user to not keep it for a long time inside a bag. Also, we would not recommend user to carry this camera to a hot place or stay in a hot weather for a long time.

References

[1] A. Greenberg, "150 Filmmakers Ask Nikon and Canon to Sell Encrypted Cameras," in WIRED, WIRED, 2016. [Online]. Available: https://www.wired.com/2016/12/200-filmmakers-ask-nikon-canon-sell-encrypted-cameras/. Accessed: Feb. 6, 2017.

[2] "ATmega2560," in MICROCHIP. [Online]. Available: http://www.microchip.com/wwwproducts/en/ATmega2560. Accessed: Feb. 24, 2017.

[3] "Nikon D5200," in Nikon. [Online]. Available: http://www.nikonusa.com/en/nikon-products/product/dslr-cameras/d5200.html. Accessed: Feb. 25, 2017.

[4] "Nikon D750," in Nikon, 2017. [Online]. Available: http://www.nikonusa.com/en/nikon-products/product/dslr-cameras/d750.html. Accessed: Feb. 21, 2017.

[5] N. K. Pareek, V. Patidar, and K. K. Sud, "Image encryption using chaotic logistic map," Image and Vision Computing, vol. 24, no. 9, pp. 926–934, Sep. 2006.

[6] "IEEE IEEE code of ethics," in IEEE.org, 2017. [Online]. Available: http://www.ieee.org/about/corporate/governance/p7-8.html. Accessed: Feb. 8, 2017.