FRAME BUFFER DESIGN FOR IMAGE SENSOR ARRAY

Project Report

By

Raj Parihar

In Partial Fulfillment of the

Course ECE404 – High-Performance Microprocessor Based Systems

April 2009

University of Rochester

Rochester, New York
# TABLE OF CONTENTS

ABSTRACT ...........................................................................................................................................3

1. Introduction......................................................................................................................................4

2. Overview: Delta-Sigma based Image Sensor Array .................................................................5
   2.1 Sigma-Delta Pixel Sensor .......................................................................................................5
   2.2 Image Sensor Array ...............................................................................................................6
   2.3 Assumptions ..........................................................................................................................7

3. Frame Buffer Implementation .....................................................................................................8
   3.1 Frame Buffer .......................................................................................................................8
   3.2 Raw Sample Accumulation ...................................................................................................8
   3.3 DCT/Wavelet based Accumulation .....................................................................................9
   3.4 Objectives/ Requirement ......................................................................................................9
   3.5 High Throughput Techniques ..............................................................................................10

4. Simulations and Analysis ............................................................................................................11
   4.1 Simulation Environment ......................................................................................................11
   4.2 CACTI Timing Model ..........................................................................................................11
   4.3 Simulation Results ...............................................................................................................12
      4.3.1 Monolithic Implementation ..........................................................................................12
      4.3.2 Multiple Rank based Organization .............................................................................14
      4.3.3 Two-Level Accumulation ............................................................................................15

5. Optimizations/Suggestions ........................................................................................................19
   5.1 Proposal/ Changes in Design ..............................................................................................19

6. Results ..........................................................................................................................................21

REFERENCES ......................................................................................................................................22
ABSTRACT

Most of the state-of-art image sensor arrays generate large bit-streams of raw data which often contain very less useful information. Data collection from such sensors and processing requires high bandwidth and high processing power at the server side. In addition to that, overall power consumption which is mainly due to processing of raw data also increases.

As an alternative to traditional sensor-server based system we propose “smart sensor architecture” which pre-processes the raw data at sensor-end before transmitting it. Pre-processing can be as simple as accumulation of bit-stream or could be as complex as discrete cosine transform (DCT) – first step of JPEG compression – of raw data. This would result into reduced demand of transmission bandwidth and less processing at server end. This sort of approach would optimize the overall performance with reduced overall power consumption. However, the cost is the increased complexity of sensor array and possibly increased power consumption of every sensor.

In this report we present few image sensor architectures which pre-process the raw data captured from Sigma-Delta (SD) based image sensor of size 1024 x 1024. For high quality video streaming, 30 frames/S, the overall power consumption of image sensor array is less than 1mW. We propose various accumulator architectures for image sensor array that accumulate the bit-streams generated from image sensor at sensor end. The best design consumes the power which is of the order of sensor arrays’ power consumption.
1. INTRODUCTION

Most of the image sensors generate large bit-streams of raw data which often contain very less useful information and present significant challenges to digital signal processors (DSP) to process them. Traditional centralized processors based sensor system, where sensors only capture the psychophysical phenomena and servers do the processing of data, demands a high bandwidth and also overall power is not optimized. Smart sensor arrays which are capable of performing preprocessing on raw data before sending to the server would ensure the low bandwidth requirement and also could minimize the overall power consumption.

This report describes the design of frame buffer which could be integrated with image sensor arrays. Frame buffer stores the image frames generated from image sensors and possibly could perform some basic transformations i.e. accumulation, discrete cosine transform. We describe design and various architectures that enable direct accumulation of uncompressed image.

The primary objective is to minimize the overall power consumption of system which includes the image sensor array, frame buffer to store 1 Mega pixels data and additional support circuitry such as multiplexers, adders, buffers etc. Emphasis was on to use low power DRAM which gives five times improvement on SRAM based data array.
2. OVERVIEW: DELTA-SIGMA BASED IMAGE SENSOR

2.1 Sigma-Delta Pixel Sensor

In image sensors, photodiode works as integrator which integrates the falling light over a time period, known as integration time, and generates equivalent analog voltage. Practical lower and upper bounds on integration time were found to be in the range of 10 us to 100 ms [1].

![Block diagram of a first order, single bit sigma-delta ADC based pixel](image)

Fig 1: Block diagram of a first order, single bit sigma-delta ADC based pixel

Figure 1 depicts a block diagram of sigma-delta modulation based image pixel. The modulator contains a first order integrator and single bit comparator/quantizer with an adjustable threshold level; in practice the threshold is set at one-half of the maximum signal level. Noise may be added at the comparator input to provide dither and reduce limit cycle behavior in the modulator. The feedback DAC gain is adjustable. We assume that the modulator input signal is a DC level, which is representative of the image light intensity at the pixel location.
2.2 Image Sensor Array

Image sensor array is array of CMOS image pixels arranged in rows and columns. In our design we consider the size of sensor array is 1024 by 1024. We assume that each pixel in the row is addressable by a column select (CS) operation. Each row of the image sensor then will operate in parallel with the CS enabling the user to select a given pixel in the rows. This is shown below in figure 2.

![Image sensor array diagram](image)

Fig 2: Image sensor array: All pixels in same column operate in parallel

One way to operate the sensor is to select the columns in sequence - this is the most probable mode of operation of the sensor since it allows each pixel the maximum light integration time between read events. For example if the Over Sampling Rate (OSR) is 256 then the readout operation begins with Column 1 (C1) and proceeds to the right until the last column (CN) is reached. The column select then returns to C1 and repeats the scan. The columns are swept through OSR times in this manner. Thus each row will output a stream of single bits with the column outputs interleaved. As an example if we have 1024 columns, an OSR of 256 and 30 frames per second then the bit rate per row will be 7.86 MHz.
2.3 Assumptions

1. Image Sensor Array and Scanning

In this report it is assumed that sensor array employs a Sigma-Delta modulator based pixel. However, any type of sensor architecture can be assumed without much loss of generality. Scanning of the pixels is sequential which means the data is read from one column and all rows at a time and before moving to next.

2. DRAM Data Array Access

In most of the analysis it is assumed that one row width is equal to the width of data bus. This is hardly the case in most of the modern DRAM system. If we take this into consideration then the power figure stated in the report of over-estimated.
3. FRAME BUFFER IMPLEMENTATION

3.1 Frame Buffer

In order to store 1024 by 1024 pixels’ data we require 1 MB storage which is also called frame buffer because it stores a frame after generation of all bits from pixel array.

3.2 Raw Sample Accumulation

Here is a top level view of image accumulator architecture for raw data stream accumulation. Rows are interleaved in order to reduce the number of processing elements i.e. adders required in order to perform the accumulation. Every bit generation from sensor array would require two full blown memory operations - first read and then write to same location.

Fig 3: Image Accumulator Architecture for raw data stream accumulation
3.3 DCT/ Wavelet based Image Accumulation

The basic architecture is same and still performs the accumulation. However, the difference would lie in performing DCT before accumulating the data and storing into the frame buffer.

![Image Accumulator Architecture](image.png)

Fig 4: Image Accumulator Architecture for DCT/ Wavelet accumulation

There are two practical difficulties in this implementation. First, we need to fetch the larger data from DRAM instead of just a byte as in previous case. We will consider this in the next revision. As of now this is just a pre-mature idea.

3.4 Objectives/ Requirements

The primary objective is to minimize the overall power of system and achieve the power consumption of frame buffer of the order of image sensor array. The major source of power consumption in the frame buffer is the data array used to store the 1 Mega pixels. The high quality video streaming imaging requires 30
Frames/s. For 1 mega pixels resolution we require an aggregate bandwidth of 16 GB/s which can be calculated as following.

\[
\text{Total Bandwidth} = \text{Frames/s} \times \text{Rows} \times \text{Column} \times \text{OSR} \times 8 \times 2
\]

\[
= 30 \times 1024 \times 1024 \times 256 \times 8 \times 2
\]

\[
= 16 \text{ GB/s} \text{ (Approx)}
\]

OSR is over-sampling rate of sigma-delta pixel. This means every pixel is sampled 256 times before a byte-wide data is generated from a pixel. Multiplication by 8 signifies the fact that each byte consist of 8 bits and 2 is because of the fact that for every pixel bit two accesses, first read the old value and then write to that, are required to frame buffer. This bandwidth requirement could further increase if we decide to use a filter instead of simple accumulator (adders or counters) to accumulate the stream of 256 bits. This is because of the fact that sigma-delta sampling is immune to noise and 12-bit resolution can be achieved from over-sampling of 256. This 12-bit resolution for every pixel would result into 24 GB/s bandwidth.

### 3.5 High Throughput Techniques

There are various ways to achieve high bandwidth in DRAM systems. One can use a single bank with wider data ports. Alternatively, many banks can be used in parallel which doesn’t enforce the rows to close before the next accesses. In addition to that, we can pipeline the operations such as while we write to one bank we can read from other bank. Larger page size or row size would certainly increase the locality and also the bandwidth. Some of these would result into the increased power consumption. The idea is to maximize the bandwidth while keeping the power consumption low. Maximized bandwidth can be traded-off with power in order to make the design even more power efficient.
4. SIMULATIONS AND ANALYSIS

4.1 Simulation Environment

All the simulations are done using web-based version of CACTI v5.3. CACTI is a cache and memory arrays’ power and access time estimation tool developed by HP lab. It takes the size of data array and target technology as input and estimates the overall power consumption, including dynamic and leakage power, along with access time and random cycle time. Simulation results presented in this report are targeted for 90 nm technology. In 90 nm technology the power supply for SRAM array is 1.2 V whereas for DRAM is 0.9 V. Specific technology cells which were chosen are following:

1. ITRS – HP: High performance SRAM based cells
   - To achieve low latency at the cost of increased power consumption.
2. ITRS – LOP: SRAM based cells optimized for power
   - To achieve low power at the cost of reduced speed
3. COMM – DRAM: Commodity embedded DRAM
   - To achieve the ultra low power at the cost of reduced speed
   - These are specific DRAM based cells which use trench capacitor to store the data

4.2 CACTI Timing Model

For any data array the CACTI model specifies the following timings.

- Random Access Time (RAT)
  
  This is the time taken to get the data from storage cell to output of the sense amplifier – into row buffers. If a particular row is activated this is the total time required to access the data as well.
- **Random Cycle Time (RCT)**
  This much time is needed to activate any row in the same bank. This includes the pre-charging of next row and closing of previous row.

- **Multibank Interleave Cycle Time (MICT)**
  This much time is required to activate any row in other bank. This is lower than Random Cycle Time in most of the cases and doesn’t require closing the row in previous bank.

From the above descriptions it is clear that the minimum time required to access the data is RAT which is the case if the row is open. RAT with RCT is required in the case where data is in the same bank and not in the row buffer. The case where data is present in other bank is RAT plus MICT. In the simulation, we compute the maximum operating frequency as the reciprocal of total access time. However the access time is little over-estimated because in general the row could have much longer so for subsequent we just require RAT.

### 4.3 Simulation Results

In this section we present the simulation results of various implementations. The results presented in this section are for a data array of size 1 MB which provides the aggregate bandwidth of 16 GB/s.

#### 4.3.1 Monolithic Implementation

1. **SRAM based Data Array**
   SRAM provides good speed although it suffers from high power consumption. To bolster our assumption, that DRAM is right choice, we simulated the design using SRAM and found out that for same bandwidth requirement SRAM based implementation is *five times* more power hungry than DRAM.
Fig 5: Power consumption SRAM data array with different data-width

The minimum power consumption comes for a 1 MB SRAM array which has a single read-write port of width 1024. The order of power consumption for monolithic SRAM based frame buffer is 1 W for 16 GB/s bandwidth.

2. DRAM based

As an alternative to SRAM we simulated the power figures for DRAM based monolithic implementation. Although the speed of DRAM is not as good as SRAM still DRAM array with wider data ports achieves the same bandwidth.

Fig 6: Power consumption of DRAM data array with different data-width
A DRAM of 1 MB with data width of 2048 gives best power consumption which is 5x lower than SRAM based data array of same size for same bandwidth.

4.3.2 Multiple Rank based Organization
As we have seen in section 4.3.1 that the overall power of frame buffer is two orders higher than the image sensor power. We further try to optimize the power by combining the multiple DRAM arrays with smaller data width into multiple rank organizations. A rank is group of devices which share the common address and data buses. Devices from different ranks can be accessed in concurrent manner. The organization looks as depicted in figure below.

Figure 7: Two-Rank implementation of frame buffer
With two-rank organization the overall bandwidth requirement of 16 GB/s is reduced to 8 GB/s because while the one access is still in progress to rank 1 the other one case still be imitated to rank 2 in parallel. Multiple devices in same rank also increase the data width of data array without increasing the data width of individual device. This sort of implementation gives 2x improvements over monolithic implementation.

![Bar chart showing total power consumption for different array sizes](image)

Fig. 8: Power consumption of DRAM based Two-Rank organization for 16 GB/s

The array size implies the size of one particular array which is grouped together with similar multiple arrays to constitute the total storage requirement of 1MB. For 16 GB/s and 1024 data width/ rank the simulation shows that if 16 64-KB arrays with a data width of 128 are combined in two ranks would give the best power figure which close to 120 mW.

4.3.3 Two-Level Accumulation

While two-rank implementation reduces the power consumption still it is not of the order of image sensor’s power consumption. In order to reduce the power consumption further we try to accumulate the raw data, generated
from pixels, in distributed manner. While the first-level of accumulation and frame buffers (buckets) provides the high bandwidth the second level of frame buffer provides the capacity.

![Diagram of two-level accumulation implementation of frame buffer]

Figure 9: Two-level accumulation implementation of frame buffer

The first-level buckets are of smaller size and are accessed more often then the second level. While one bucket from the first level engages with image sensor and accumulator hardware the second one tries to refill the data to and from the second level. The second level of implementation could either pair of bank based of multiple bank based implementation as described in section 4.3.2.

1. First Level Buffer (Buckets)

The first step is to choose the size and data-width of first level buffers which minimize the power consumption.
Simulation results show that a 4-KB of data array with 512-bit wide data port minimizes the overall power for first level. It is approx 14 mW.

2. Second Level Buffer

For second level buffer multiple organizations can be used such as multiple-rank organization. Here we try to implement a pair of banks which would minimize the overall power of second level.

Fig 10: Power consumption of first-level buffer in two-level accumulation.

Fig 11: Power consumption of second-level buffer in two-level accumulation.
Simulation result show that 4 256-KB DRAM buffers with each having a data-width of 2048 minimizes the overall power consumption which is 26.6 mW. This way the total power consumption of two-level frame buffer is approx 40mW. This is the best so far. The reason for such low power consumption is that the accesses to second-level array are less frequent. Another reason is that while the data is accessed from one bank the other bank goes to sleep mode and contribute only leakage power as oppose to other implementations where multiple banks contributed to dynamic power because most of them were activated most of the time.
5 OPTIMIZATIONS/ SUGGESTIONS

5.1 Proposal/ Changes in Design

One of the complications in the sigma-delta pixel design which we are studying is that the comparator and DAC are shared among the complete row of pixels in following manner.

![Image Sensor with shared sigma-delta ADC among pixels](image)

Fig 12: Image Sensor with shared sigma-delta ADC among pixels

This kind of architecture severely limits the integration of memory (small buffers) close to pixels due to the fact that readout circuitry is shared among pixels and the bit-stream would be interleaved. This also restricts to perform readout in parallel fashion. In order to minimize the power for whole system we propose to perform accumulation of n-bits data near to pixels itself in following manner.

![Block Diagram of DPS with individual digital readout](image)

Fig 13: Block Diagram of DPS with individual digital readout

This kind of architecture would require individual ADC and DAC for each pixel. However, this would allow the integration of small piece of memory (up to a byte) in the proximity of image pixel.

These small buffers can be implemented using 3-T DRAM cell because

1. While 3-T DRAM cell is as fast as 6-T SRAM it consumes less than half of the power as compare to SRAM.
2. It is implemented without any capacitor as oppose to 1-T DRAM.
3. READ operation is non-destructive (so no pre-charging is required) in 3-T DRAM as oppose to 1-T DRAM.

2-bit memory (now 6 additional transistors per pixel) would allow the reading rate to reduce by 4 which would also cut down the power by 4 times. Modified block diagram of complete system looks as following.

Two proposed changes are as following:
- Individual ADC for each pixel
- 2-bit 3-T based buffer for each pixels

![Diagram](image)

**Fig 14**: Modification proposed in image accumulator architecture

The 3-T and Comp/DAC block can be shared among 2 to 4 pixels but sharing them among the whole row doesn’t seem practical if we want to reduce the power by implementing a level of accumulation locally.
6 RESULTS

We discussed four possible implementation of frame buffer for image sensor. Multiple-Rank organization and Two-Level Accumulation are orthogonal which can be incorporated in any implementation. The proposal described in section-5 is also orthogonal. The power saving factor is $2^n$ for n-bit pixel memory implementation. To compare the various implementations we summarize the results presented in section-4.

<table>
<thead>
<tr>
<th>Frame Buffers Implementation</th>
<th>Power Consumption (mW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monolithic: SRAM</td>
<td>847</td>
</tr>
<tr>
<td>Monolithic: DRAM</td>
<td>162</td>
</tr>
<tr>
<td>Multi-Rank DRAM</td>
<td>119</td>
</tr>
<tr>
<td>Two-Level DRAM</td>
<td>40</td>
</tr>
<tr>
<td>Pixel-Level 1T DRAM</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 1: Comparison: Various implementation of frame buffer for 16 GB/s BW

![Various Frame Buffers](image_url)

Fig15: Comparison: Various implementation of frame buffer for a 16 GB/s BW
REFERENCES


[2] “0.88nw/pixel, 99.6 dB linear dynamic range fully digital Image sensor employing a pixel level Sigma-Delta ADC”, Z. Ignjatovic, Mark F. Bocko

