3-5-2015

Low Power Real-time Video and Audio Embedded System Design for Naturalistic Bicycle Study

Janardhan Bhima Reddy Karri

University of South Florida, reddy4@mail.usf.edu

Follow this and additional works at: https://scholarcommons.usf.edu/etd

Part of the Computer Engineering Commons

Scholar Commons Citation
https://scholarcommons.usf.edu/etd/5518

This Thesis is brought to you for free and open access by the Graduate School at Scholar Commons. It has been accepted for inclusion in Graduate Theses and Dissertations by an authorized administrator of Scholar Commons. For more information, please contact scholarcommons@usf.edu.
Low Power Real-time Video and Audio Embedded System Design for
Naturalistic Bicycle Study

by

Janardhan Bhima Reddy Karri

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Computer Engineering
Department of Computer Science and Engineering
College of Engineering
University of South Florida

Co-Major Professor: Srinivas Katkoori, Ph.D.
Co-Major Professor: Pei-Sung Lin, Ph.D.
Swaroop Ghosh, Ph.D.

Date of Approval:
March 5, 2015

Keywords: Arduino, Graphical User Interface, Data Acquisition System, SPI Interface

Copyright © 2015, Janardhan Bhima Reddy Karri
DEDICATION

To my loving family for their patience and support.
ACKNOWLEDGMENTS

This work has been supported and funded by the Center for Urban Transportation and Research (CUTR) at USF and Florida Department of Transportation (FDOT). My sincere gratitude to Dr. Srinivas Katkoori for his advice and guidance. I am very thankful to CUTR and specifically to Dr. Pei-Sung Lin and Dr. Achilleas Kourtellis for their tremendous help. I would also like to thank Dr. Swaroop Ghosh for serving on my thesis committee. My friends who are also my partners in this project, Omkar Dokur and Noureddine Elmehraz should be given equal credit at this moment. Thanks to you all.
# TABLE OF CONTENTS

LIST OF TABLES .................................................................................................................. iii

LIST OF FIGURES ................................................................................................................ iv

ABSTRACT .............................................................................................................................. vi

CHAPTER 1 - INTRODUCTION AND MOTIVATION .............................................................. 1
  1.1 Naturalistic Bicycle Study ................................................................................................. 1
  1.2 Problem Description ........................................................................................................ 3
  1.3 Thesis Organization ......................................................................................................... 4
  1.4 Chapter Summary ......................................................................................................... 4

CHAPTER 2 - REAL-TIME VIDEO RECORDING SYSTEM ..................................................... 5
  2.1 Camera VC0706 ............................................................................................................. 6
  2.2 Choosing a Microcontroller ............................................................................................ 7
    2.2.1 First Version – Arduino UNO .................................................................................. 7
    2.2.2 Second Version – Arduino MEGA .......................................................................... 8
    2.2.3 Third Version – Arduino DUE .............................................................................. 10
    2.2.4 Fourth Version – MBED LPC1768 ......................................................................... 11
  2.3 Power Consumption of Proposed Video System .......................................................... 12
  2.4 Form Factor .................................................................................................................. 12
  2.5 System Cost .................................................................................................................. 13
  2.6 SD Card Timing ........................................................................................................... 14
  2.7 Chapter Summary ........................................................................................................ 15

CHAPTER 3 - DATA COLLECTION, SYNCHRONIZATION, AND VISUALIZATION .......... 16
  3.1 Collecting Sensors Data in a Time Frame ..................................................................... 16
  3.2 Synchronization Challenge .......................................................................................... 17
  3.3 Graphical User Interface (GUI) .................................................................................... 18
    3.3.1 Frontend User Interface .......................................................................................... 18
    3.3.2 Backend User Interface ......................................................................................... 20
  3.4 Chapter Summary ........................................................................................................ 20

CHAPTER 4 - SYSTEM OPERATION AND POWER MANAGEMENT ...................................... 21
  4.1 Power Management Schematic ..................................................................................... 21
  4.2 Operational Details ....................................................................................................... 22
  4.3 Complete System Flowchart ....................................................................................... 23
  4.4 Chapter Summary ....................................................................................................... 25
CHAPTER 5 - EXPERIMENTAL RESULTS .................................................................26
  5.1 Sample Trials and Data Collection ..........................................................26
  5.2 Variation of Frame Size According to Lighting Conditions ......................28
  5.3 Power Profile Graph ..............................................................................29
  5.4 Chapter Summary ..................................................................................30

CHAPTER 6 - CONCLUSION AND FUTURE WORK .............................................31
  6.1 Conclusion ............................................................................................31
  6.2 Future Version Development with Intel Edison ......................................32
    6.2.1 Scope .............................................................................................32
    6.2.2 Development Platform Options ......................................................33

REFERENCES .................................................................................................35

APPENDICES ..................................................................................................36
  Appendix A Code .......................................................................................37
  Appendix B Copyright Permissions ..............................................................38
LIST OF TABLES

Table 1 Power Consumption Details at 12V Supply Voltage..........................................................12
Table 2 System Cost Breakdown .......................................................................................................13
Table 3 SPI Interface Efficient Coding ..............................................................................................15
Table 4 Task Sequence and Time Required to Acquire a Single Frame ...........................................16
Table 5 Optimized Task Sequence .....................................................................................................17
Table 6 Sample Trials and Data Size ................................................................................................27
Table 7 Frame Size Variation with Lighting Conditions ....................................................................28
LIST OF FIGURES

Figure 1 Conceptual Drawing of a Bicycle Equipped with Data Acquisition System ............2
Figure 2 Interconnect of the Major Modules in the Proposed Real-Time Video Subsystem .......5
Figure 3 RadioShack VC0706 Camera Module .................................................................6
Figure 4 Arduino UNO ........................................................................................................8
Figure 5 Arduino MEGA ......................................................................................................9
Figure 6 ISR Execution in SPI Interval between Each Byte Transfer .................................9
Figure 7 Arduino DUE .........................................................................................................10
Figure 8 MBED LPC1768 ................................................................................................11
Figure 9 Graphical User Interface .....................................................................................19
Figure 10 Real-Time Data Display Using GUI .................................................................19
Figure 11 Power Management Schematic ........................................................................22
Figure 12 Complete System Flowchart ............................................................................24
Figure 13 BDAS Installed on a Bicycle ............................................................................26
Figure 14 Back Module Sample Frame ............................................................................27
Figure 15 Front Module Sample Frame ............................................................................27
Figure 16 Power Profile Graph .........................................................................................29
Figure 17 Current Consumption Graph ............................................................................30
Figure 18 Intel Edison Specifications ...............................................................................33
Figure 19 Webcam Interfacing with Intel Edison .................................................................33
Figure 20 Edison Developer Options ..................................................................................34
ABSTRACT

According to NHTSA Traffic Safety Facts [9], bicyclist deaths and injuries in 2013 are recorded as 732 and 48,000, respectively. In the State of Florida the safety of bicyclists is of particular concern as the bicycle fatality rates are nearly triple the national average. Further Florida ranks #1 on bicycle fatality rate in the nation for several years. To determine the cause of near-misses and crashes, a detailed study of bicyclist behavior and environmental conditions is needed. In a Florida Department of Transport (FDOT) funded project, USF CUTR has proposed naturalistic bicycle study based on ride data collected from 100 bicyclists for 3000 hrs. To this end, Bicycle Data Acquisition System (BDAS) is being researched and developed. The main objective of this thesis work is to design and implement low power video and audio subsystems of BDAS as specified by domain experts (USF CUTR researchers). This work also involves design of graphical user interface (Windows application) to visualize the data in a synchronized manner. Selection of appropriate hardware to capture and store data is critical as it should meet several criteria like low power consumption, low cost, and small form factor. Several Camera controllers were evaluated in terms of their performance and cost. The major challenges in this design are synchronization between collected data, storage of the video and sensor data, and design of low power embedded subsystems.
CHAPTER 1 - INTRODUCTION AND MOTIVATION

Bicycling is an increasingly popular mode of transportation in the US as it is eco-friendly, economical, and promotes healthy life. However, the occurrence rate of injuries and fatalities amongst bicyclists is very high when compared to other types of roadway vehicles. In Florida, the safety of bicyclists is of particular concern as the bicycle fatality rates were nearly triple the national average. Further, Florida ranks first in bicycle fatality rate in the nation for years [6]. Recent statistics shows that rate of bicycle accidents are growing rapidly over years. In 2013, the 732 bicyclist deaths and 48,000 injuries were recorded. In previous year of 2012, 726 bicyclist deaths and 49,000 bicyclist injuries were recorded [9]. In the time period of 2001 to 2011, the bicyclist injury rate increase by an alarming 8.9 percent [10].

It is clear from the above accident statistics, the cycling safety needs to improve. To do so naturalistic bicycle study has been proposed by USF CUTR to identify the bicyclist behaviors and environmental hazards leading to crashes and near-misses.

1.1 Naturalistic Bicycle Study

The required data for naturalistic bicycle study are real-time video, speed, date, time, and location, closeness to other vehicles, acceleration of the bike, inclination of the bike, temperature of the surrounding, and lighting measurement. An embedded system has to be designed to collect the required data. The data collected using this system helps in the analysis of the reasons and factors affecting the accidents occurred to the bicyclists. Further, this data can be used to take
countermeasures to increase the safety of the bicyclists. Figure 1 shows conceptual drawing of a bicycle equipped with Data Acquisition System.

Figure 1. Conceptual Drawing of a Bicycle Equipped with Data Acquisition System

The following are the system requirements:

- To collect synchronized data from front and rear video cameras, date and time, speed, acceleration, GPS coordinates, gyro readings, proximity distance data, and audio data.
- The system should run for 40 hours of user rides without any user intervention. This can be made possible by having a low power design that can be run continuously on a portable battery.
- The system should communicate with smart phone app at the start and end of a ride.
• The storage capacity should be large enough to accommodate and store data for not less than 40 hours.

The naturalistic bicycle study methodology has three phases – system prototype development phase, data collection phase, and data analysis phase. In the first phase, several prototypes will be designed and a robust system is developed. Next fifty (50) such systems are assembled and tested. In the second phase, 100 participants are recruited in two batches to collect 3000 person-hour data. In the third phase, the collected data is analyzed to answer the following six questions:

• What are the interactions between bicyclists and drivers making right turns at intersections (right-hook)?
• What are the interactions between drivers making left turns and oncoming bicyclists (left-hook)?
• What are the behavior, experience, and interactions of bicyclists and drivers at night?
• What are bicyclist route choices decisions with given origins and destinations?
• What is the difference of bicycling behaviors with and without formal bicycle-riding training such as Cycling Savvy?
• What are the contributing factors to bicycle crashes or close calls?

1.2 Problem Description

The following are the four major challenges in the design of BDAS:

1. Design of efficient algorithm for collection of data from all sensors in specific time frame.
2. Synchronization between these data is more important and storing them in a specific format as needed for GUI.

3. Design low power video recording system with minimum frame rate of 30 frames per second (fps).

4. Overcoming the writing speed latency of SDHC or SDXC cards.

In this thesis work the following problems are addressed: 1) Design of low power embedded subsystem to capture real-time video and audio data; 2) storage, synchronization, and development of graphical user interface for bicycle study data representation. 3) Power management of data acquisition system.

1.3 Thesis Organization

The rest of this thesis is organized as follows: Chapter 2 introduces different camera modules considered for the front/rear cameras of the system. We compare and contrast the camera modules in terms of dollar cost, interfacing requirements, and maximum achievable frame rates. Based on this comparison, we identify the camera controller to be used in the BDAS. Chapter 3 discusses the data collection, synchronization, and visualization. Chapter 4 presents the power management details and the system operational details. Chapter 5 reports the experimental results. Chapter 6 draws conclusion and outlines future work. Appendices A and B present the core Arduino code and copyright permissions.

1.4 Chapter Summary

In this chapter, the need for collecting naturalistic bicyclist data is explained. The design requirements and tasks to be addressed are identified. The challenges involved in the design of the system are listed. The naturalistic bicycle study methodology is briefly explained.
CHAPTER 2 - REAL-TIME VIDEO RECORDING SYSTEM

To identify bicyclist’s behavior and environmental crash hazards, we require many hours of ride data. As the system is powered by a battery, design of real-time video and audio recording system with low power is of main concern. The specifications for real-time video system are:

- The frame rate should be at least 15 frames per sec (fps).
- The resolution must be at least 640 x 480.
- The maximum power consumption is 100 mA@12V.

The proposed hardware consists of the following three modules interconnected as shown in Figure 2: a) Sam3x8e ARM microcontroller (Arduino DUE); b) JPEG camera module; and c) SDHC or SDXC card.

Figure 2. Interconnect of the Major Modules in the Proposed Real-Time Video Subsystem
2.1 Camera VC0706

Based on specifications different camera modules have been explored. Specifically, we considered: OV7670 camera module, C3088 camera module, TTL Serial JPEG camera, RadioShack camera module (VC0706) etc. Based on time constraints and video quality specifications, RadioShack camera module is chosen for video recording. The main reason is its capability for generating JPEG images while many other candidate cameras give the images in RAW format with large file sizes. The image processor in VC0706 can generate JPEG images at 640 x 480 pixel resolution. Using this module, we can achieve the required frame rate of 15 fps. It supports both SPI and UART interfaces for data collection and communication with the device respectively. Support for composite video output is also available in this module using which we can adjust the focus of the lens. All communication with camera is done using UART (frame length, resume, and stop) but actual frame data is received through SPI interface. This module works in the master mode when it uses SPI interface. Figure 3 shows the image of the VC0706 Camera Controller.

![Figure 3. RadioShack VC0706 Camera Module](image-url)
2.2 Choosing a Microcontroller

When it comes to choosing the microcontroller, the governing factor is the camera module. As mentioned in the previous section, the camera module’s SPI interface is in master mode by default. Hence the controller should be chosen in such a way that it supports SPI slave interface with a minimum RAM of 32KB.

In this project we considered AVR 8bit microcontrollers, ARM 32bit microcontrollers, and Intel Edison. To program these microcontrollers there are associated development platforms. However, some of them do not have the support to use full features of actual hardware (microcontroller). So we should choose development platform such that it supports required features we need.

The main task of microcontroller is to receive JPEG encoded frames from camera and store them in a SD card. During this transfer process the microcontroller should act as a SPI slave, while receiving frames from camera module and as a SPI master mode, while writing to SD card. The file size of each frame varies from 10KB to 30KB based on lighting and color of each frame.

While evaluating the microcontroller several obstacles were faced which rendered the microcontroller to be unusable and the need for the next version arose. In this section various versions of microcontrollers are presented along with their advantages and disadvantages.

2.2.1 First Version – Arduino UNO

In first version, Arduino UNO (Controller Specs: AVR ATmega328, 16 MHz clock, 2KB SRAM) microcontroller is used. The main focus was on the frame rate and stability of the camera module. As the camera modules suggested used is to take single images, we need to
ensure that it can be used for acquiring video in terms of takes 15 images per sec (i.e., 15 fps). After several trail-and-error experiments, we were able to get a stable implementation. Figure 4 shows an image of Arduino UNO.

![Arduino UNO](image)

**Figure 4. Arduino UNO**

The drawback with this version is when frame size is 25KB, there is a need to communicate with camera nearly 15 times to receive one complete frame because of limited 2KB RAM. Therefore, we could achieve only a maximum of 1 frame per second with this microcontroller.

### 2.2.2 Second Version – Arduino MEGA

In second version, Arduino MEGA (Controller Specs: AVR ATmega1280, 16MHz clock, 8KB SRAM) microcontroller is used. The larger RAM helps reduce the number of communication cycles between the microcontroller and the camera. With Arduino MEGA we were able to receive each frame in 2 or 3 cycles. This translated to a frame rate of 5 fps, which still fell short of the required 15 fps. Figure 5 shows the image of Arduino MEGA.
The drawback with this version is high ISR execution time. In SPI master mode the camera module can transfer data at the speeds of 1 MHz to 13.5 MHz. The interrupt service routine (ISR) of the controller reads the SPI data register and saves it to the RAM. Execution time of ISR must be less than interval between each byte transfer. The microcontroller can attain a successful read and write operation only at speed of 1 MHz. Any higher rates from the camera resulted in losing bytes due to over-writing on the SPI register. Hence there is a need for a controller that can execute the ISR in less clock cycles. Figure 6 shows ISR execution in SPI interval between each byte transfer.
2.2.3 Third Version – Arduino DUE

In the third and final current version Arduino DUE (Controller Specs: 32-bit ARM core, Atmel SAM3X8E ARM Cortex-M3, 96 KB SRAM (two banks: 64KB and 32KB)) microcontroller is used.

In this version we have enough RAM to receive a complete image at a Speed of 6 MHz and write to SD card. As this microcontroller has 96 MHz clock ISR routines executes much faster and thus we have overcome the drawback of the previous microcontroller. Larger RAM and faster clock speed of Arduino DUE resulted in frame rate of 15 fps. Figure 7 shows the image of Arduino DUE Prototype Board.

![Arduino DUE Prototype Board](image)

Figure 7. Arduino DUE

The drawbacks with this version are: a) when assembling 50 systems, unique ID should be assigned for each system but there is No EEPROM in this microcontroller. Therefore, assigning a unique ID to each system is a challenge. This problem is solved by using the unique ID of Bluetooth as System ID; b) Even though this architecture supports high level features such as
image processing, SD High speed interface, they cannot be leveraged as the development platform is not fully mature.

### 2.2.4 Fourth Version – MBED LPC1768

After the Atmel SAM3X8E controller, MBED LPC1768 (Controller Specs: NXP LPC1768, 32-bit ARM Cortex-M3 core, 96 MHz clock, 512 KB FLASH, 32 KB RAM) is chosen.

In this microcontroller we have two SPI interfaces: one can be used in slave mode and the other in master mode. So a frame from camera can be received simultaneously while writing another frame to SD card. MBED RTOS is easy to implement and takes less time. The read and write operations can be run in two concurrent threads. Figure 8 shows the image of MBED LPC1768 Prototype Board.

![Figure 8. MBED LPC1768](image-url)
The drawback with this version is that the software development platform MBED does not support SPI burst read even though hardware NXP LPC1768 supports it. Therefore, the chip select should be pulsed for every byte which cannot be set from the camera module side. As a result after each byte the chip-select has to be made high and to continue reading again the chip-select has to be made low by the camera module.

2.3 Power Consumption of Proposed Video System

Table 2 shows power consumption details of proposed video system in standby state and active state.

<table>
<thead>
<tr>
<th>Component</th>
<th>Idle or Standby Current (mA)</th>
<th>Active State Current (mA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camera Module</td>
<td>30</td>
<td>40</td>
</tr>
<tr>
<td>Sam3x8e</td>
<td>50</td>
<td>100</td>
</tr>
<tr>
<td>SD card</td>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td>Total</td>
<td>85</td>
<td>160</td>
</tr>
</tbody>
</table>

2.4 Form Factor

Currently, the system uses lead acid battery, which is low cost, so form factor of this system is not compact. By using Lithium ion battery, form factor of the system can be reduced by three times but increases the cost of the system (by approximately 70 USD). So, proper tradeoff analysis between cost and size has to be made in selection of the battery.
2.5 System Cost

Table 1 shows cost of all Back module components, Power management components, and Front module components.

Table 2 System Cost Breakdown

<table>
<thead>
<tr>
<th>Back Module Components</th>
<th>Cost (USD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arduino Due Board</td>
<td>40</td>
</tr>
<tr>
<td>GPS</td>
<td>33</td>
</tr>
<tr>
<td>SD card 32GB</td>
<td>15</td>
</tr>
<tr>
<td>MPU-6050 6DOF 3 Axis Gyroscope +Accelerometer</td>
<td>4.2</td>
</tr>
<tr>
<td>3 Ultrasonic HC-SR04 sensors</td>
<td>4.8</td>
</tr>
<tr>
<td>Radio Shack Camera Module</td>
<td>25</td>
</tr>
<tr>
<td>12V 10AH battery</td>
<td>23</td>
</tr>
<tr>
<td>CR1220 Low Drain 3V lithium Battery(for GPS)</td>
<td>1</td>
</tr>
<tr>
<td>PCB estimation</td>
<td>20</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Power Management Components</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>MOSFT Load Switch-Si1869DH</td>
<td>1.3</td>
</tr>
<tr>
<td>ADXL362 Accelerometer breakout board</td>
<td>14</td>
</tr>
<tr>
<td>3V Lithium CR2032 Battery</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Front Module components</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Arduino Due Board</td>
<td>40</td>
</tr>
<tr>
<td>SD card 32GB</td>
<td>15</td>
</tr>
<tr>
<td>Ultrasonic HC-SR04 sensors</td>
<td>4.8</td>
</tr>
<tr>
<td>Radio Shack Camera Module</td>
<td>25</td>
</tr>
<tr>
<td>MEMS Microphone Breakout</td>
<td>10</td>
</tr>
<tr>
<td>PCB estimation</td>
<td>20</td>
</tr>
</tbody>
</table>

| Total Cost                                                  | 297        |
2.6 SD Card Timing

Communication with SD cards can be done through SPI and SD Bus protocols.

1. *SPI Bus protocol:* Most of microcontrollers support SPI interface. This is easy to implement but the maximum write speed is only 500 KB/s even for class 10 10MB/S cards. Using SPI interface, the protocol treats all class SD cards as class 0 SD cards even if it is ultra-SD cards (9 MB/s).

2. *SD Bus protocol:* Only a few microcontrollers support this mode. It is 4 bit SD mode. It supports full speed of SD cards (e.g., class 6 with 6 MB/s transfer speed, class 10 with 10 MB/s speed). To use this mode, the microcontroller must support this feature. In some microcontrollers, it referred to as HSMCI or SD controller mode.

The nominal time taken to transfer a 25 KB image in the Arduino environment is up to 250 ms. This time must be reduced as much as possible to achieve a higher frame rate required in the real-time environment. The transfer time can be minimized by reducing the file system layers and/or by efficient coding. These two optimizing methods are briefly described below.

The file system layers that can be dropped are

1. *SD Volume:* This layer supports FAT16 and FAT32 partitions.

2. *SD File:* File access functions are handled by this layer. Some of the file access functions handled in this layer are open(), close(), read(), write(), remove(), and sync(). Access to the root directory and subdirectories is also supported by this layer.
Table 2 shows an example of the optimizing technique for efficient coding. A simple change in the position of a line in the code can save 1 micro second per byte. When transferring an image with size of 25 KB, this technique can save as much as 25 ms.

Table 3 SPI Interface Efficient Coding

<table>
<thead>
<tr>
<th>for (i = 0; i &lt; 512; i++)</th>
<th>SPDR = 0xFF;</th>
</tr>
</thead>
<tbody>
<tr>
<td>{ SPDR = 0xFF;</td>
<td>for (i = 0; i &lt; 511; i++)</td>
</tr>
<tr>
<td>while(!((SPSR&amp;(1 &lt;&lt; SPIF))));</td>
<td>{ while(!((SPSR&amp;(1&lt;&lt;SPIF))));</td>
</tr>
<tr>
<td>array[i] = SPDR;</td>
<td>array[i] = SPDR;</td>
</tr>
<tr>
<td>}</td>
<td>SPDR = 0xFF;</td>
</tr>
<tr>
<td></td>
<td>}</td>
</tr>
<tr>
<td>2.75μs pause between each SPI read</td>
<td>1.75μs pause between each SPI read</td>
</tr>
</tbody>
</table>

2.7 Chapter Summary

In this chapter, different micro-controller and camera modules have been evaluated. Their drawbacks and benefits are discussed explaining the reason for selecting this particular microcontroller to meet the design requirements. Form factor, power consumption, and cost attributed have been estimated.
Synchronization between the collected data and its visualization are the main challenges in the naturalistic bicycle study. Therefore, a graphical user interface is designed for data visualization. This chapter discusses how the tasks are optimized with time and data is stored.

### 3.1 Collecting Sensors Data in a Time Frame

The primary time constraints are: capturing video at 12 frames per second from front Camera module; capturing video at 10 frames per second from back Camera module; measuring distance with each proximity sensor, 4 times per second; measuring acceleration and inclination 4 times per second, and measuring GPS coordinates once a second.

In Arduino DUE microcontroller we are continuously waiting 20-40ms for each frame to read SPI receive register. So approximately 20-40%, of time we cannot collect any sensor data. However, collecting all other sensors data in a timely manner is more important. Table shows breakdown of the time required for different tasks involved in collecting a frame data, assuming frame size to be 25 KB.

<table>
<thead>
<tr>
<th>Task</th>
<th>Sequence and Time Required to Acquire a Single Frame</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resume</td>
<td>Stop</td>
</tr>
<tr>
<td>30</td>
<td>2</td>
</tr>
</tbody>
</table>
By optimizing the task sequence, the time taken for the write operation can be reduced. By combining the Resume operation with SD card writing, 30 ms can be saved.

Table 5 Optimized Task Sequence

<table>
<thead>
<tr>
<th>Stop</th>
<th>Get Framelength</th>
<th>Get Frame Through SPI</th>
<th>Resume &amp; Writing to SD card</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>3</td>
<td>40ms@6MHz</td>
<td>40 ms</td>
<td>85 ms</td>
</tr>
</tbody>
</table>

In this system all non-visual sensor data is collected between capturing of frames. The time taken is 300 ms for every 4 frames. The sensor data is collected from accelerometer, gyro, and proximity sensor and this takes 50 ms making a total of 350 ms for 4 frames. This cycle repeats after every 350 ms.

3.2 Synchronization Challenge

As frame size varies from 10 KB to 32 KB the frame rate of both modules (front and back) is not constant. The file size of each frame changes because of varying lighting conditions and scenes. The front camera module has variable frame rate of 12-16 fps. The back Camera module has variable frame rate of 10-14 fps. Therefore, synchronization between both the two modules and all other sensors data is required.

Initially when the system initializes, time & date information is received from GPS. A new session is named according to date and time. GPS has data logging ability which can log data up to 30 hours. So we will be collecting data from GPS only at end of the session.

Microcontroller uses inbuilt timer to measure the time it takes to capture every 100 frames and stores all these timing information in a file which is later used for synchronization. The
microcontroller captures all sensors data for every 4 frames and stores them with respect to frame number.

During synchronization display, a video is made with constant frame rate of 10 fps and while playing the video, the play rate is changed according to the time taken to capture current 100 frames. And all other sensors data is displayed according to frame number.

3.3 Graphical User Interface (GUI)

To identify the behaviors of riders and environmental crash hazards, we require an interface (Application) to display the data in synchronized manner. This application consists of two parts: 1) Frontend user interface which is visible to user; and 2) Backend is database where the actual data resides.

3.3.1 Frontend User Interface

The frontend user interface application for naturalistic bicycle study is developed using Microsoft’s Visual Studio framework. To display maps, we have integrated Google maps using javascript and jquery. In frontend interface, when user selects particular session or video from local database or from server, then all related data (graphs, maps, video, etc.) must be populated on web form. Data synchronization is performed using the date and time stamp which is stored along with all sensors data. Figures 9 and 10 show the sample screenshots of GUI.
Figure 9. Graphical User Interface

Figure 10. Real-Time Data Display Using GUI
There is one more application user interface that copies all data from SD card, post-processes it and stores them in database in required format for quick access. Some of the post process tasks are: 1) Creating a video out of image sequence; 2) Adding audio to the video and 3) Converting GPS locus parser data into time, date, and geo-coordinates.

3.3.2 Backend User Interface

The backend part of application consists of database SQL and its connectivity. Microsoft’s ODBC connectivity tools are used for database connectivity. All data is stored in the database in tabular form except videos. For video data, links for video source are stored and the associated video is stored on hard disk inappropriate video format.

3.4 Chapter Summary

In this chapter, we discussed how various sensor data is captured in one time frame. Further, we briefly discussed data synchronization challenges. Lastly, we presented the implementation details of data visualization framework consisting of frontend (GUI) and backend (database).
CHAPTER 4 – SYSTEM OPERATION AND POWER MANAGEMENT

For the system to gather data about the naturalistic behavior of the bicyclist, it is important that he/she be unaware of the data acquisition system. Therefore, the bicyclist should not be required to turn on/off the system. The power consumption of complete (front and back module) system is estimated to be 3.6W. As the system is powered by a portable battery (10Ah), in order to save power, the system should automatically shut off in idle mode.

A power management system is developed that can automatically power on when there motion is detected for certain period of time. It also monitors the idle condition to power off the system.

4.1 Power Management Schematic

Figure 11 shows schematic of the proposed Power management with three components.

- Accelerometer (ADXL362)
- Small coin cell battery (2V)
- Switch with on/off input (Si1869DH).
4.2 Operational Details

Ultra-low Power Accelerometer ADXL362 is used to detect the presence of a motion. This accelerometer can be configured in such a way that it generates INT1 (output logic high) and maintains this logic even in absence of motion until INT1 is cleared by microcontroller (Arduino Due). Therefore, there will be no abnormal power shutdown when there is no motion.

The power supply from main battery to Arduino due is controlled by a switch (Si1869DH). The on/off logic for switch is provided by ADXL362 through INT1. The BDAS has other accelerometer that is used for measuring acceleration and gyro readings, therefore microcontrollers scans the last 30 seconds reading and decides whether the system is idle or not and clears the interrupt if it idle. This scanning of reading is done only once for every 30 seconds. Once the microcontroller decides it is idle, interrupt is cleared by microcontroller.
only after finishing all other tasks (such as closing file, end of session Bluetooth communication) etc.

4.3 Complete System Flowchart

Bicycle Data acquisition system is powered on only when there is a motion detection of 5 seconds, which is done with the help of power management as discussed in above section. Figure 12 shows the complete system flowchart. Starting with initialization, in this phase all Serial, SPI, I2C interfaces, and sensors are initialized. Sensors functionality is checked in the next phase and if all the sensors work then it moves to next block. If there is any error with the sensors then a complete debug is done and the information is sent to the smart phone through Bluetooth. Therefore, when the sensors work correctly, the system proceeds to the following sequential tasks: (a) Getting time from the GPS and forwarding it to the front module as the session folder (trip) is named according to the date and time; (b) Creates a log file for current session (trip) which can be used for debugging purposes; and (c) Starts GPS data logging; and (d) Camera initial setting (compression ratio, SPI baud rate, Resolution).

Now the execution moves on to the next phase where the frames are captured. For every fourth frame, data is captured from distance, accelerometer and gyroscope sensors. This data is directly saved in the RAM, and for every 1024 frames the collected data from distance and accelerometer sensors is transferred to SD card. And also, the system checks whether the bicycle is in movement or ideal. If it is ideal then the system moves to the shutdown phase where the following tasks are done: (a) Stop GPS data logging and transfers the logged data from GPS to SD card; and (b) log file for current session (trip) is updated. Finally, the system communicates with accelerometer in power management to turn off the power.
Figure 12. Complete System Flowchart
4.4 Chapter Summary

In this chapter, the power management and its operational details are discussed. The complete system working flow chart is explained in detail.
CHAPTER 5 - EXPERIMENTAL RESULTS

Audio and video systems were continuously tested for 48 hours to test the stability. The results obtained from the test are accurate. Figure 13 shows BDAS installed on the bicycle. The front and back module are powered by a 12 Volts 10 Ampere-hour battery. Both modules communicate with each other through UART for synchronization of data.

![BDAS Installed on a Bicycle](image)

Figure 13. BDAS Installed on a Bicycle

5.1 Sample Trials and Data Collection

Table 6 shows the duration of sample trials and size of the data. Trails 1 and 2 are conducted on USF Campus with the BDAS installed on a bicycle. The video is acquired at 15 fps with
resolution of 640 x 480. Trail 1 is a short ride that resulted in approximately 116 MB data. Trial 2 is a longer ride for 27 minutes that resulted in 555 MB data. In order to test the stability of video and audio systems, BDAS system was run for 30 hours continuously. Approximately 30 GB of data was generated at the rate of 1 GB/hour. We verified the video and audio randomly and found that the system worked as expected.

Table 6 Sample Trials and Data Size

<table>
<thead>
<tr>
<th>tests</th>
<th>Time</th>
<th>Video</th>
<th>Audio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trial 1</td>
<td>374 seconds</td>
<td>112MB</td>
<td>3.8MB</td>
</tr>
<tr>
<td>Trial 2</td>
<td>27.61 Minutes</td>
<td>533MB</td>
<td>20.2 MB</td>
</tr>
<tr>
<td>Trial 3</td>
<td>30 hours (continuously without power management)</td>
<td>29 GB</td>
<td>1.009 GB</td>
</tr>
</tbody>
</table>

Figures 14 and 15 shows sample snapshots of back and front modules respectively.

Figure 14. Back Module Sample Frame

Figure 15. Front Module Sample Frame
5.2 Variation of Frame Size According to Lighting Conditions

The frame size varies according to lighting conditions and the color of the image. Table 7 shows the variation of frame size with varying lighting conditions. The largest file size occurs under the best lighting conditions.

Table 7 Framesize Variation with Lighting Conditions

<table>
<thead>
<tr>
<th>Frame size</th>
<th>Lighting condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.5KB</td>
<td>Poor</td>
</tr>
<tr>
<td>18KB</td>
<td>Fair</td>
</tr>
<tr>
<td>25.5KB</td>
<td>Good</td>
</tr>
</tbody>
</table>
5.3 Power Profile Graph

The system was run continuously for 48 hours and the current consumption of system was analyzed. A 1 Ω resistor was connected between the battery output and power supply of BDAS. Figure 16 shows the source voltage as function of time. We can observe that voltage drops from 12V to 10V over 48 hours. Figure 17 shows the current profile.

We have used MCR-4V, a multi-channel voltage data logger for power profiling the system. Immediate on-the-spot checking of continually-changing data is possible with this data logger.

![Power Profile Graph](image)

Figure 16. Power Profile Graph
5.4 Chapter Summary

In this chapter experimental results for video and audio system are presented. Power profile and storage results are also presented.
CHAPTER 6 - CONCLUSION AND FUTURE WORK

6.1 Conclusion
This work presented hardware design details to collect real-time video and audio data as well as software implementation to analyze the collected data. Each of the sensors was tested independently. Then, they were added one by one to the BDAS and consistency in the data collection and storage was achieved. Finally, this data was collected in a synchronized manner. The power management system powers on the BDAS when there is motion and shuts the power down when no motion accordingly. The current system meets the design requirements of naturalistic bicycle study and all the challenges described above have been met. Currently, the system uses lead acid battery, which is low cost, but increases the form factor of the system. Lithium ion battery can be used which reduces the form factor of the system but increases the cost of the system. So, there is a tradeoff between cost and size in the selection of the battery. Moreover, the graphical user interface developed using Microsoft Visual Studio provides better tool to organize analyze and represent the data. The same system can also be used for motorcycle study. The Intel Edison can be the potential tool for further development and the upside in the usage of that specific module can be the IOT connectivity, HD video synchronization, etc. Cycling data can be used to implement new protective measures/traffic rules to increase the safety of the bicyclists.
6.2 Future Version Development with Intel Edison

6.2.1 Scope

The BDAS developed now is just a beginning and there a lot of options and parameters that can be incorporated to make it better. The latest technology can help in developing a system that may even live stream the videos along with the related sensor data. The current model has some drawbacks:

- Any version update cannot be made in the current system because the total system is developed according to the requirements of camera.
- All sensors are chosen according to development platform interface support and microcontroller interface support.

Intel Edison is chosen for future version because we can capture HD video from USB webcams save it to SD card. USB Video Class devices can stream video functionality on the Universal Serial Bus. There are more than 200 UVC compactable webcams models available in market. So this provides a better scope for camera module selection in terms of resolution, zoom distance, form factor, and cost.

Measuring more bicycle-related variables such as reaction times and braking times can be implemented in further versions. Figure 18 shows Intel Edison controller specs.
<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor</td>
<td>22 nm Intel® SoC that includes a dual-core, dual-threaded Intel® Atom™ CPU at 500 MHz and a 32-bit Intel® Quark™ microcontroller at 100 MHz</td>
</tr>
<tr>
<td>RAM</td>
<td>1 GB LPDDR3 POP memory (2 channel 32 bits @ 800 MT/sec)</td>
</tr>
<tr>
<td>Internal storage</td>
<td>4 GB eMMC (v4.51 spec)</td>
</tr>
<tr>
<td>Power</td>
<td>T1 S/N8024 power management IC</td>
</tr>
<tr>
<td>Wireless</td>
<td>Dual-band (2.4 and 5 GHz) IEEE 802.11a/b/g/n</td>
</tr>
<tr>
<td>Bluetooth*</td>
<td>BT 4.0 + 2.1 EDR</td>
</tr>
<tr>
<td>Antenna</td>
<td>Dual-band onboard chip antenna or u.FL for external antenna</td>
</tr>
<tr>
<td>Connector</td>
<td>70-pin Hirose DF40 Series (1.5, 2.0, or 3.0 mm stack height)</td>
</tr>
<tr>
<td>Size</td>
<td>35.5 × 25.0 × 3.9 mm maximum (to be verified)</td>
</tr>
<tr>
<td>Power input</td>
<td>3.15 to 4.5 V</td>
</tr>
<tr>
<td>I/O</td>
<td>40 general purpose GPIO which can be configured as:</td>
</tr>
<tr>
<td></td>
<td>• SD card: 1 interface</td>
</tr>
<tr>
<td></td>
<td>• UART: 2 controllers (one full flow control, one Rx/Tx)</td>
</tr>
<tr>
<td></td>
<td>• I²C: 2 controllers</td>
</tr>
<tr>
<td></td>
<td>• SPI: 1 controller with 2 chip selects</td>
</tr>
<tr>
<td></td>
<td>• I2S: 1 controller</td>
</tr>
<tr>
<td></td>
<td>• GPIO: Additional 14 (with 4 capable of PWM)</td>
</tr>
<tr>
<td>USB 2.0</td>
<td>1 OTG controller</td>
</tr>
<tr>
<td>Clocks</td>
<td>19.2 MHz, 32 kHz</td>
</tr>
</tbody>
</table>

Figure 18. Intel Edison Specifications (source: [7])

![Image of Intel Edison with webcam](image_url)

Figure 19. Webcam Interfacing with Intel Edison

### 6.2.2 Development Platform Options

We have more Platform development options for Intel Edison because it uses Yocto Linux. Custom Linux-based systems for embedded products can be created using Yocto project regardless of the hardware architecture.
Figure 20. Edison Developer Options (source:[7])
REFERENCES


APPENDICES
Appendix A Code

This appendix section contains code structure used in the project.

```c
void setup()
{
  Initialize();   //initialize Serial ports 1,2,3 and I2C lines.
  accelgyro_initialization(); //Accelrometer+Gyroscope initialization.
  camera_initialization();    // sets SPI Baudrate & compression ratio
  SPI_MasterInit();          // SPI is set to mastermode
  GPS_initialization();      // Get time from GPS & start datalogging
  sd_initialization();       // SDCARD initialization
  AllsensorsFunctional();    // checks sensors are working or not
  if (runsystem == false)
    systemshutdown();
  create_subfolder_session();
}

Void loop()
{
  // capture 4 frame frames & and sensor readings w.r.t. every 4th frame
  capture_frame();          // capture frame.
  capture_frame();          // capture frame.
  capture_frame();          // capture frame.
  capture_frame();          // capture frame.
  ultrasonicL();           // Measure Left Distance sensor reading & save to ram.
  ultrasonicR();           // Measure Right Distance sensor reading & save to ram.
  ultrasonicS();           // Measure Front Distance sensor reading & save to ram.
  AccelGyro();             // Measure Acceleration & gyro sensor readings & save to ram.
  Sub_Frame_Number++;
  if (Frame_Number % 1024 == 0)
  {
    Timeupdate();        // Timeupdate called because new subdirectories have to be named
                         // according to time.
    ram_to_sd();         // Transfers last 1024 frames sensor readings from ram to
                         // sdcard. All reading are stored w.r.t to framenum.
    if (checkforidle() == true)  // checks for idle condition after every 1024 frames.
    {
      Dump_GPS_Data();     // if idle receive gps logged data.
      Updatelogfile();     // At End of session (trip), logfile is updated.
      ble();               // Communicate with Bluetooth at end of trip.
      systemshutdown();    // complete system shutdown.
    }
    create_subfolder_session(); // A new sub directory is created for every 1024
                                 // frames in current session (trip) directory.
  }
}
```
Appendix B Copyright Permissions

Figure 4, Figure 5, Figure 7, are Arduino prototype board images. Below figure shows copyright permission for using these images in my work.

Copyright Notice

Editorial contents of the arduino.cc website, such as texts and photos, are released as Creative Commons Attribution ShareAlike 3.0.

This means you can use them on your own derived works, in part or completely, as long as you also adopt the same license. You find the complete text of the license here.