School of Engineering Science
Simon Fraser University
Burnaby, BC V5A 1S6

mutrixtechnology@gmail.com

March 2nd, 2009

Prof. Patrick Leung

School of Engineering Science
Simon Fraser University
Burnaby, British Columbia
V5A 1S6

The attached document describes the design specification for *DreamBox* musical entertainment system from Mutrix Technology. We aim to design and implement an innovative musical ornament with potentials to be widely applied in various situations.

The design specification gives an informative system overview, system architecture and different design aspects of the project.

Mutrix Technology consists of five motivated, hard working, and talented fifth-year engineering students: Benson Lam, Gary Heng, Winfield Zhao, Shuozhi Yang and Weiguang Mou. If you have any questions or concerns about our proposal, please feel free to contact me by phone at (604)537-9289 or by e-mail at btl2@sfu.ca, or to contact the company by email at mutrixtechnology@gmail.com.

Sincerely yours,

Benson Lam

Benson Lam

President and CEO

Mutrix Technology

Enclosure: ***Design Specification for DreamBox***

Functional Specification

for

DreamBox Musical Entertainment System

Prepare for: Patrick Leung

Steve Whitmore

Prepared by: Benson Lam

Gary Heng

Winfield Zhao

Shuozhi Yang

Weiguang Mou

Contact: mutrixtechnology@gmail.com

Date: 3/2/2009

Revision: 3.0

EXECUTIVE SUMMARY

As the digital entertainment system market grows, innovative digital home ornaments have become very popular among modern home decoration choices. Creative digital entertainment systems bring people an enjoyable and artistic lifestyle, while introducing brand-new concepts of home entertainment. Customizability also becomes one of the key features that appeal to creative users.

DreamBox Musical Entertainment System (*DreamBox*), from Mutrix Technology, aims at providing people a fashionable, creative and entertaining lifestyle. Through unlimited combinations of crystal bar patterns and LED light effects, *DreamBox* brings people the visualized musical experience. Moreover, *DreamBox* introduces great customizability to fulfill the users' pursuit of unique artistic ornament.

The design specification for *DreamBox* outlines the design of the proof-of-concept system and provides justification for the design choices. The architecture of *DreamBox* can be divided into three parts: Central Control Unit (CCU), Motor Mechanical System (MMS) and Software User Interface (SUI).

- CCU determines the proper crystal bar patterns and LED light effects according to the audio signal processing result. It is also responsible for receiving serial data from SUI and sending serial data to MMS.
- MMS receives the control commands from CCU and adjusts the crystal bars according to the commands. The sophisticated mechanical structure of MMS guarantees the precise movement of crystal bars.
- SUI offers users a friendly GUI in personal computer platform, which enables users to create their own *DreamBox* patterns for static decoration.

The selection criteria for microcontrollers, servo motors and tri-color LEDs are provided in this document. The design of user interface is also described in detail. In the end of the document, a complete system test plan, for both hardware and software, is provided as well.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF EQUATIONS

GLOSSARY

PCB	Print Circuit Board
LED	Light Emission Diode
DSP	Digital Signal Processing
ADC	Analog-to-Digital Converter
DAC	Digital-to-Analog Converter
MIPS	Mega Instruction per Second
A/D	Analog-to-Digital
PWM	Pulse-Width-Modulation
CCU	Central Control Unit
MMS	Motor Mechanical System
SUI	Software User Interface
GUI	Graphical User Interface

1. INTRODUCTION

DreamBox is a digital musical entertainment system that provides an innovative visual musical experience and creative customization ability to users. With the support of *DreamBox*, users can enjoy the presentation of various pre-designed crystal bar patterns and LED light effects along with the rhythmic music play. In addition, users are able to take advantage of the full customizability to design their own home ornament. The design specification of *DreamBox*, as proposed by Mutrix Technology, is described in this design specification.

1.1. SCOPE

The design specification for *DreamBox* specifies the design of the proof-of-concept system, and explains how the design meets the functional requirements listed in *Functional Specification for DreamBox Musical Entertainment System* [1]. This document can be divided into the following four parts:

1) Hardware and firmware design specification
2) Mechanical design specification
3) Software design specification
4) Software/hardware test plan

The specifications listed in this document will guide the future design and development of *DreamBox* Musical Entertainment System.

1.2. INTENDED AUDIENCE

The design specification is intended to be referenced by all members of Mutrix Technology. The project group leader should refer to the design specification as a design standard to examine the project progress throughout the development phase. Hardware and software design engineers should refer to the specification listed in this document as the design guide in their development. Test engineers should use this document to design and implement the testing procedures to verify the performance of the system.

2. SYSTEM SPECIFICATION

DreamBox will work in either the dynamic mode or the static mode, in which it can perform as a dynamic musical statue or static artistic ornament, respectively. The user can switch *DreamBox* between the dynamic mode and the static mode by simply using the push button switches.

In the dynamic mode, a simple but sufficient LCD GUI will be provided for various system options including system idle and light effect mute. In the static mode, a software GUI can provide full customizability support so that users can design their personal artistic home ornament.

3. OVERALL SYSTEM DESIGN

DreamBox consists of three major parts: Central Control Unit (CCU), Motor Mechanics System (MMS) and Software User Interface (SUI). CCU is responsible for audio processing, sending commands to motor controller as well as interfacing with PC. MMS guarantees the fast and precise positioning of crystal bars when control commands arrive. SUI inspires the creativity of users while providing them full control over all *DreamBox* functionalities. The following system block diagram shows the overall design of *DreamBox* musical entertainment system.

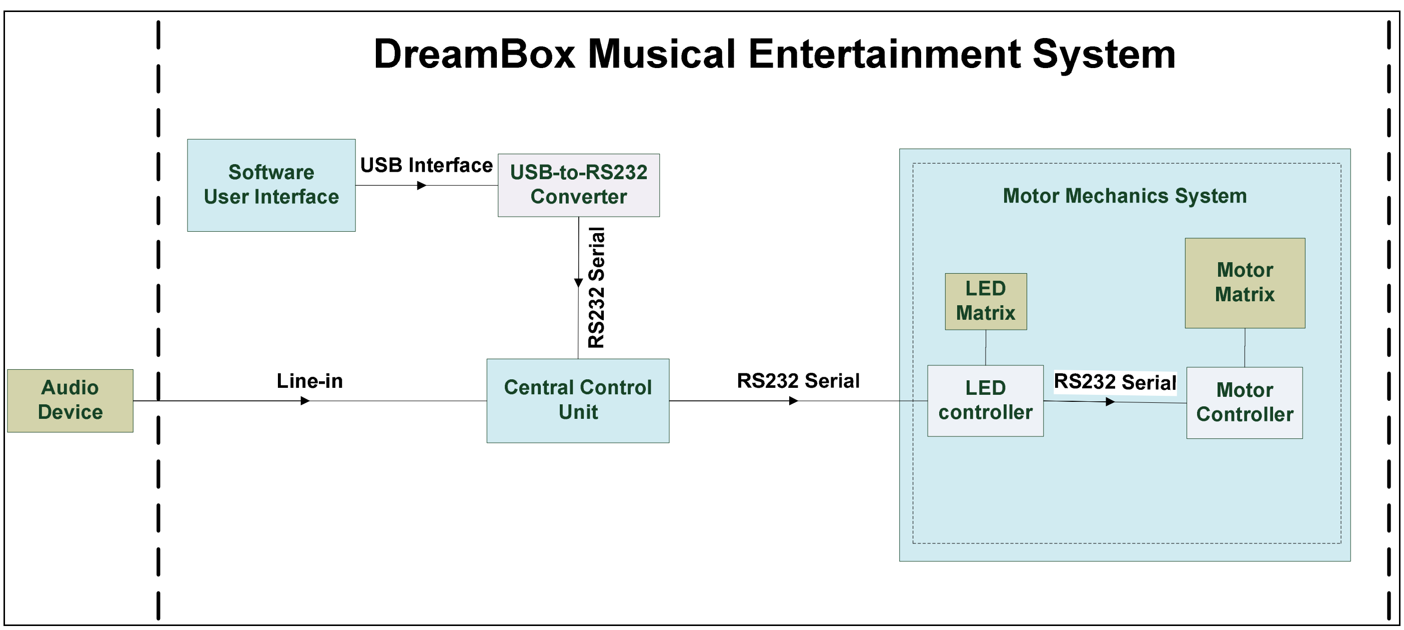

Figure 1: System Block Diagram

Depending on the mode selection, *DreamBox* uses different control command sources and control signal flow paths. The high-level operation diagram is shown below for both the static mode and the dynamic mode operations:

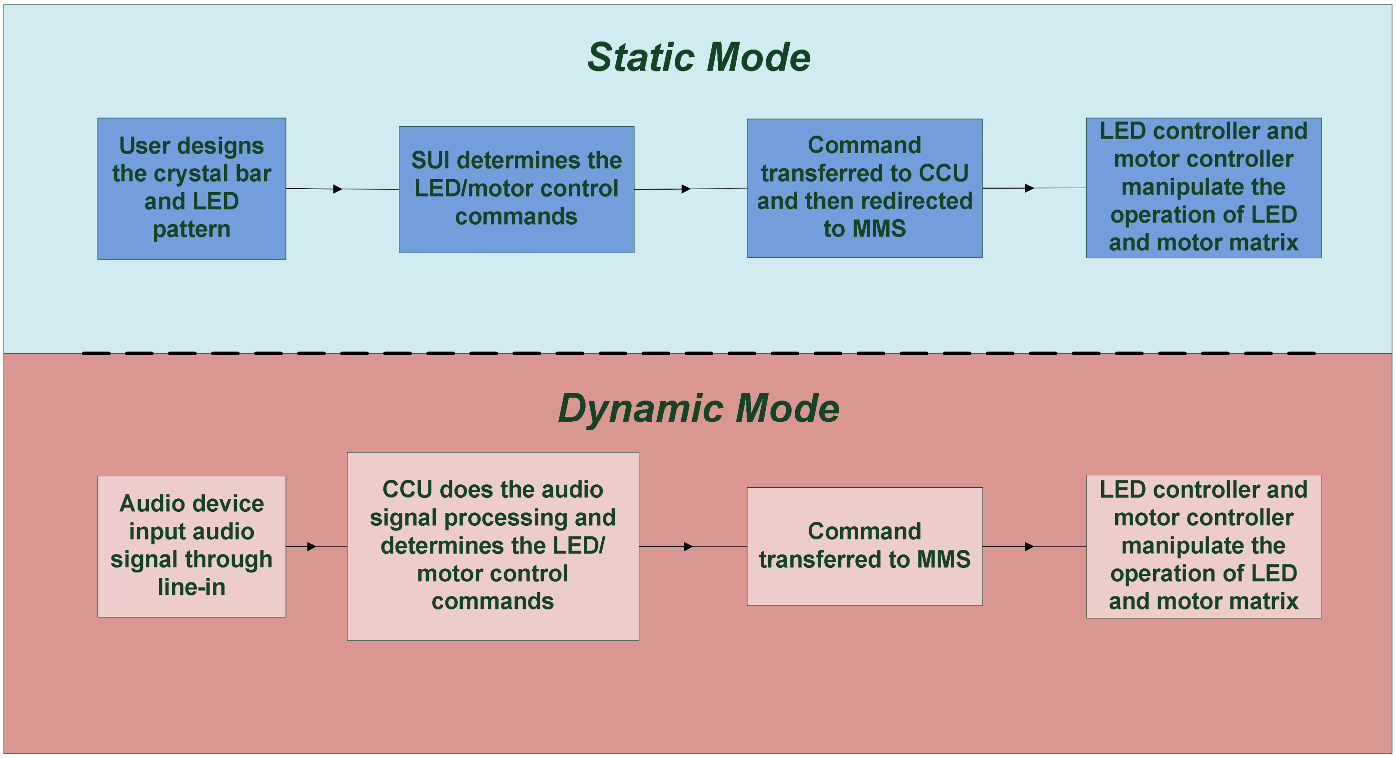

Figure 2: High-Level Operation Diagram

For the system appearance design, we choose to use crystal bars to form a 5x5 square matrix. Each crystal bar can move independently according the control commands; thus, there will be considerable number of combinations of crystal bars. With the help of LED lighting effect, the appearance of the crystal bars can be shiny and clear. The following figure shows a conceptual picture of *DreamBox*.

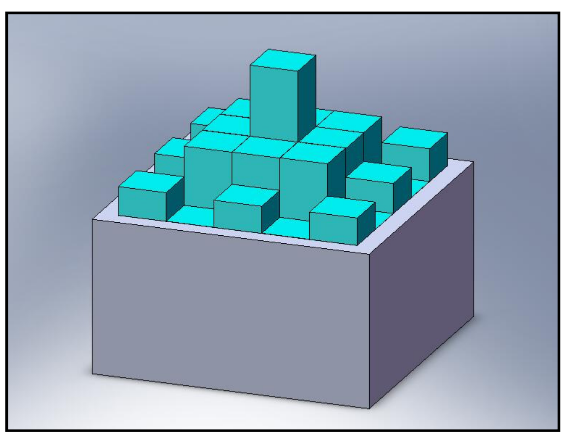

Figure 3: Conceptual Model of DreamBox

4. MICROCONTROLLER DEVELOPMENT BOARD

As depicted in Figure 1 and Figure 2, CCU of *DreamBox* takes the responsibility of audio signal processing, dynamic mode control command generation, and static mode control command delivery. Due to the requirement of digital signal processing (DSP) capability and various communication ports support, we decided to buy a microcontroller development board. Such choice significantly reduces the time and work required in project development; however, the development cost increases as a trade-off.

The development board we chose for our project is dsPICDEM GP 1.1 plus, which includes a dsPIC30F6014A microcontroller. The following figure is dsPICDEM GP 1.1 plus development board.

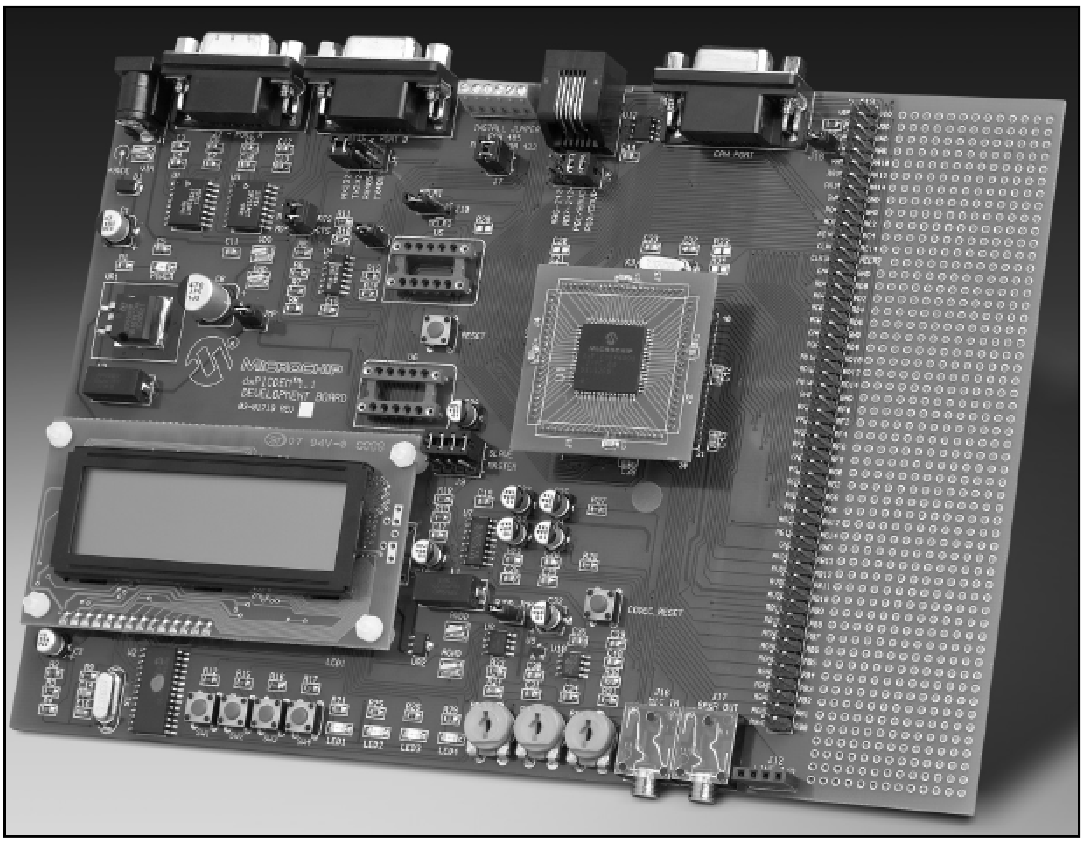

Figure 4: dsPICDEM GP 1.1 Plus Development Board

This microcontroller development board provides the following desired features [2] for our project design:

- Two RS232 communication channels
- Four push button switches
- Four red LEDs
- 122 x 32 dot addressable LCD with PIC18F242 LCD controller
- 2 x 50 prototyping header for user hardware expansion

- Prototype area for user hardware
- dsPIC30F6014A [3]:
  - 12-bit 200Ksps Analog-to-Digital Converter (ADC)
  - 4Kbytes EEPROM memory
  - Up to 30MIPs operation
  - 2 UARTs
  - 8 PWM output pins

Some desired features are explained in the following sections:

4.1.1. TWO RS232 COMMUNICATION CHANNELS

Referring to our system block diagram shown in Figure 1, the basic communication method between parts of *DreamBox* is RS232 serial connection. In order to realize the ability of both receiving serial data from SUI and sending serial data to MMS, two serial communication channels are necessary in CCU.

4.1.2. PUSH BUTTON SWITCHES

Push button switch is the only user control method when *DreamBox* is operating in the dynamic mode. *DreamBox* users use push button switch to choose the operating mode between the dynamic mode and the static mode. Also, users can access other options in dynamic mode, such as system idle and light effect mute.

4.1.3. LCD AND PIC18F242 LCD CONTROLLER

A simple GUI is realized through the 122 x 32 dot addressable LCD and PIC18F242 LCD controller. This simple GUI will display the possible options and guides the user to access the options in dynamic mode. All display content will be pre-programmed in CCU.

4.1.4. 12-BIT 200KSPS ADC

In order to have an accurate analysis on general audio signal, a sampling rate of 44.1 KHz is desired, which is the sampling rate of CD audio signal. The 12-bit 200ksps A/D supports up to 200 KHz sampling rate for the analog input signal and provides 4096 different levels of quantization.

4.1.5. 30MIPS OPERATION FREQUENCY

Sample data manipulation is needed in every sampling period, which is about 22.7 microseconds. The high operation frequency makes it possible to process the sample data without disturbing the sampling process. Fast operation is also an important aspect of real-time control system.

5. AUDIO SIGNAL PROCESSING

The audio signal processing component consists of two basic parts: analog-to-digital converter and signal processing algorithm, which correspond to hardware and firmware specification, respectively. This section of the document will describe the configuration of the audio signal processing specification in details.

5.1. ANALOG-TO-DIGITAL CONVERTER

5.1.1. PRE-PROCESSING CIRCUIT

The analog-to-digital converter (ADC) provided in dsPIC30F6014A supports various reference voltage source options, such VDD, VSS and reference voltage input pins. The standard ADC configuration provided on dsPICDEM GP 1.1 receives analog signal various between 0V to 5V; however, the audio signal is oscillating between +100mV and -100mV, as measured in the lab with multimeter. In order to ensure the proper and accurate analog-to-digital conversion, a pre-processing circuit is needed.

A voltage divider will be used to provide a reference voltage of 1.5V to the ADC reference voltage pin, and a 0.75V DC level shifter circuit will be used to boost up the oscillating audio signal to positive range. A buffer is used to ensure the constant reference voltage. The circuit is depicted below in Figure 5.

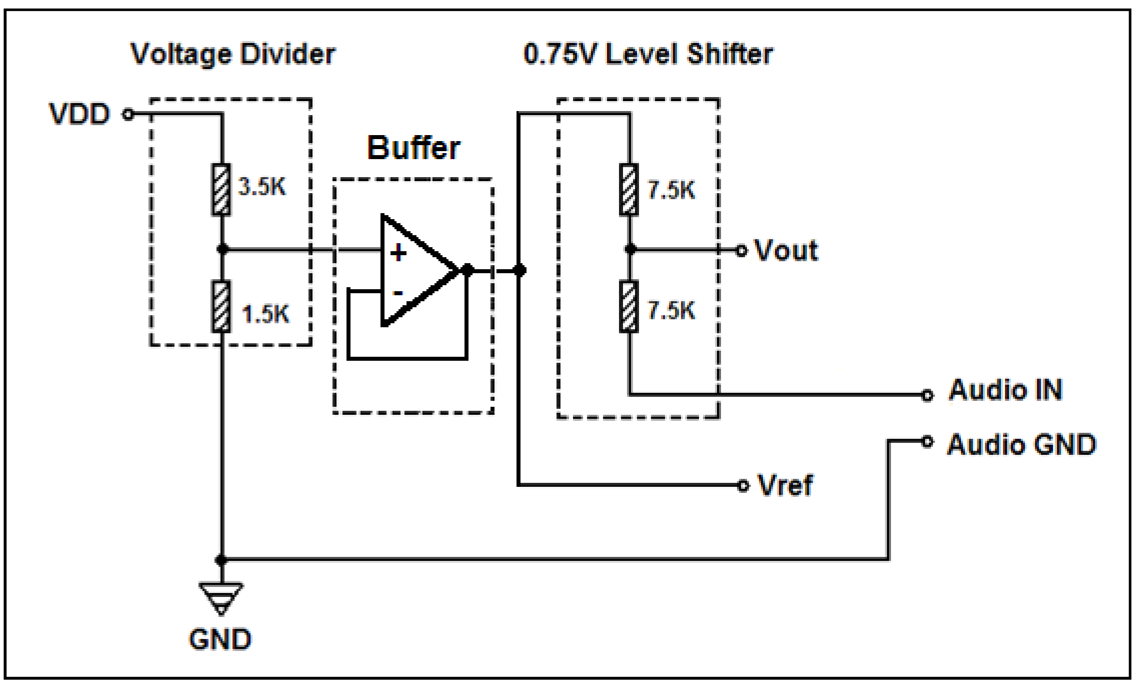

Figure 5: Audio Signal Pre-processing Circuit

As a result, the analog signal from Vout will be oscillating between 650mV and 850mV, and the sampling range will be 0V to 1.5V. The sampling precision has been increased.

5.1.2. ADC CONFIGURATION

As the datasheet states, the sampling time of the ADC in dsPIC30F6014A should be calculated according to the Instruction Cycle Time. We have configured the microcontroller to operate at its highest operation frequency, 29.4 MIPS, which is corresponding to an Instruction Cycle Time of 33.9 ns. The following equation is used to calculate the A/D clock cycle (Tad) based on Instruction Cycle Time (Tcy):

$$Tad = \frac{Tcy * (ADCS + 1)}{2} = 1.0848\ \mu s \qquad \textbf{Equation 1}$$

where ADCS is the configuration register, which is set to be 63.

In addition to the actual sampling time, the ADC needs extra 14 A/D clock cycles for the A/D process. As the desired sampling rate of the ADC is 44.1 KHz, which is corresponding to total sampling time of 22.68 μs, the total number of A/D clock cycles required is 20. Thus, the actual sampling frequency realized is about 46.1 KHz.

To avoid heavy and time consuming large data manipulation, part of the calculation will be done in the sampling period. One interrupt is generated for each sample processed by ADC, and audio pre-processing will be implemented in the period before the next interrupt.

5.2. SIGNAL PROCESSING ALGORITHM

The beat, or rhythm, of a song is a fundamental feeling of human beings. The goal of our algorithm is to translate this feeling into control signals. By using a statistical approach, we have extracted beats from a static song.

Sound waves contain different energy levels. Energy and sound are correlated (the louder the sound, the more energy involved). Therefore, by analyzing the energy level in a song, we would be able to differentiate the beats of the song. However, we need to compare the instantaneous energy with the average energy. Referring to "*Beat Detection Algorithm*" [5], if the instantaneous energy is greater than the average energy, then we have a beat!

In fact, there are many software vendors using similar rhythm/beat detection algorithm in their programs. The screen capture in Figure 5 shows a window pane from the music program, "Winamp". The left side of Figure 6 shows the left and right channels' beats whereas the right side shows the instantaneous frequency spectrum of the music.

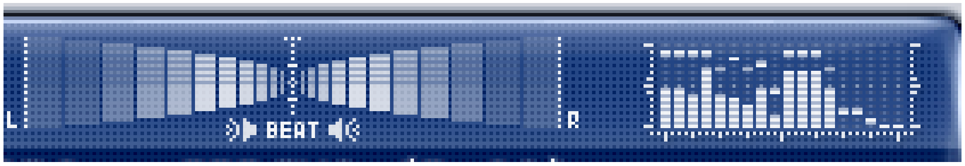

Figure 6: Frequency Spectrum and Beat from Winamp

The instantaneous energy and accumulated energy is calculated using Equation 2 below.

$$Energy = \sum_{k=i_o}^{m} a[k]^2 \qquad \textbf{Equation 2}$$

The accumulated energy is computed using the average of one-second sound wave. In our design, we have used a sampling rate of 44.1 KHz and divided our data into 30 bins with1536 samples per bin. We have imported a song and analyzed its beat by simulating the performance using MATLAB, the simulation result of which is shown in Figure 6. Each increment of the x-axis of Figure 6 indicates that one second has elapsed in time. The performance matches our expectation. The song begins with soft music followed by a sequence of melody. As a result, we have a large fluctuation of beat values at approximately the first and second minutes of the song.

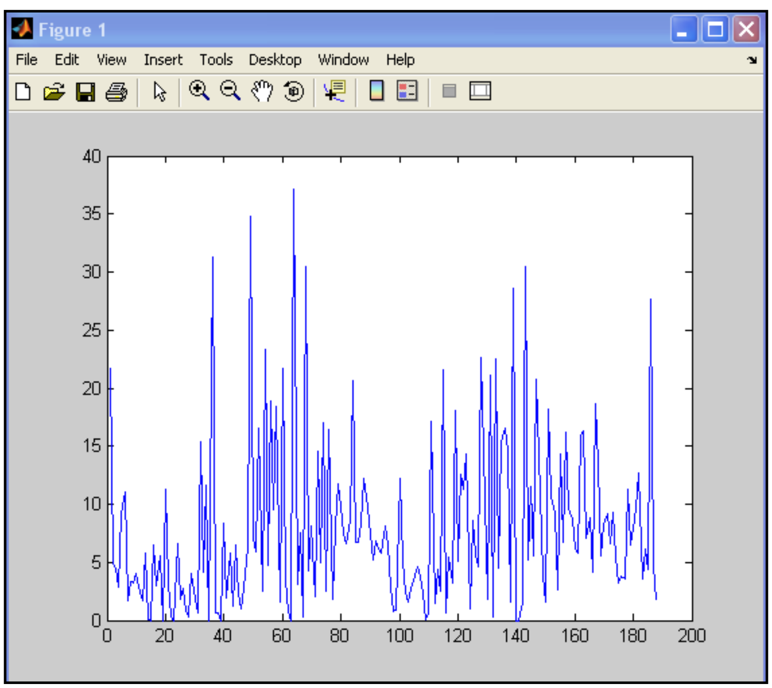

Figure 7: Rhythm/Beat Simulation

This algorithm can help us differentiate beats of a song while the song is being played. Different levels of the beat value are grouped together. Each individual beat level corresponds to a set of pre-defined crystal bar patterns and LED light effects. The flowchart of the algorithm is shown below in Figure 8.

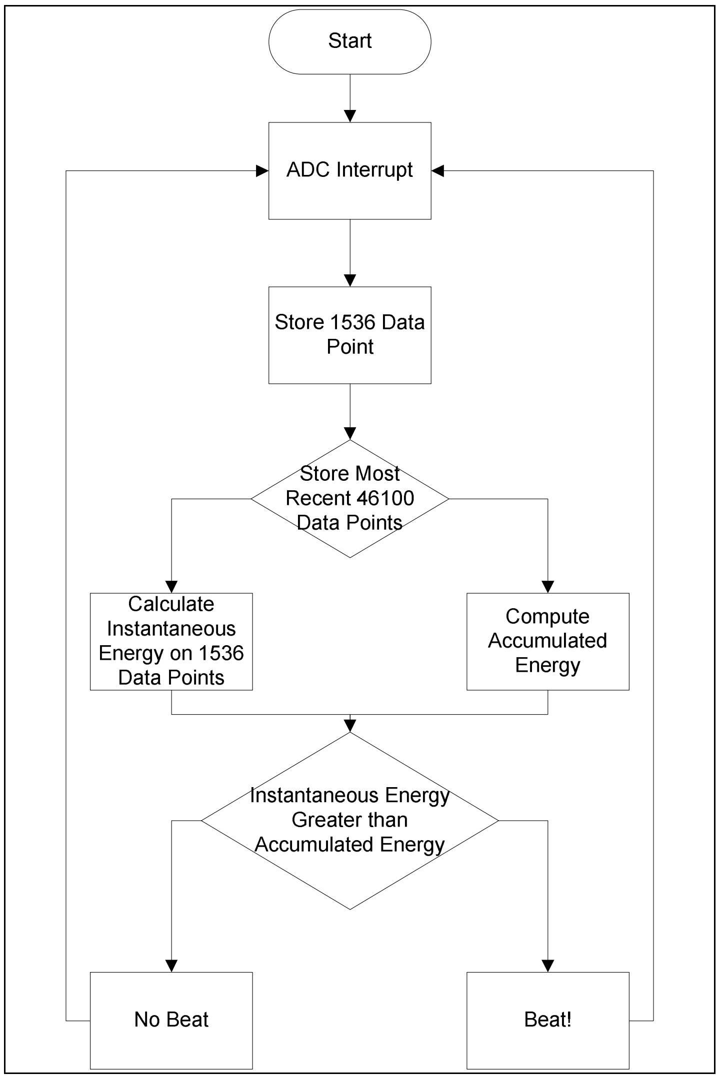

Figure 8: Rhythm/Beat Detection Algorithm Flowchart

6. USER INTERFACE

6.1. SOFTWARE USER INTERFACE

Our Software User Interface (SUI) plays the most important role during the static mode. It provides users an intuitive and interactive way to convey their creativity. The candidate programming language for building the GUI is C#. Our well-designed GUI can eliminate the need for users to learn complex command lines for controlling the amplitudes of the crystal bars and configuring the LED settings. The conceptual SUI design is shown in Figure 9. Features associated with SUI will be presented in the subsequent sections that follow.

Figure 9: Conceptual DreamBox Computer SUI

6.1.1. CONTROL MATRIX PLATFORM

The 5 x 5 matrix on SUI is purposely made to simulate the hardware platform on *DreamBox*. Each individual cell in the SUI matrix represents a crystal bar on the hardware platform. Figure 10 shows the cells in the SUI matrix and the corresponding crystal bars on the hardware platform.

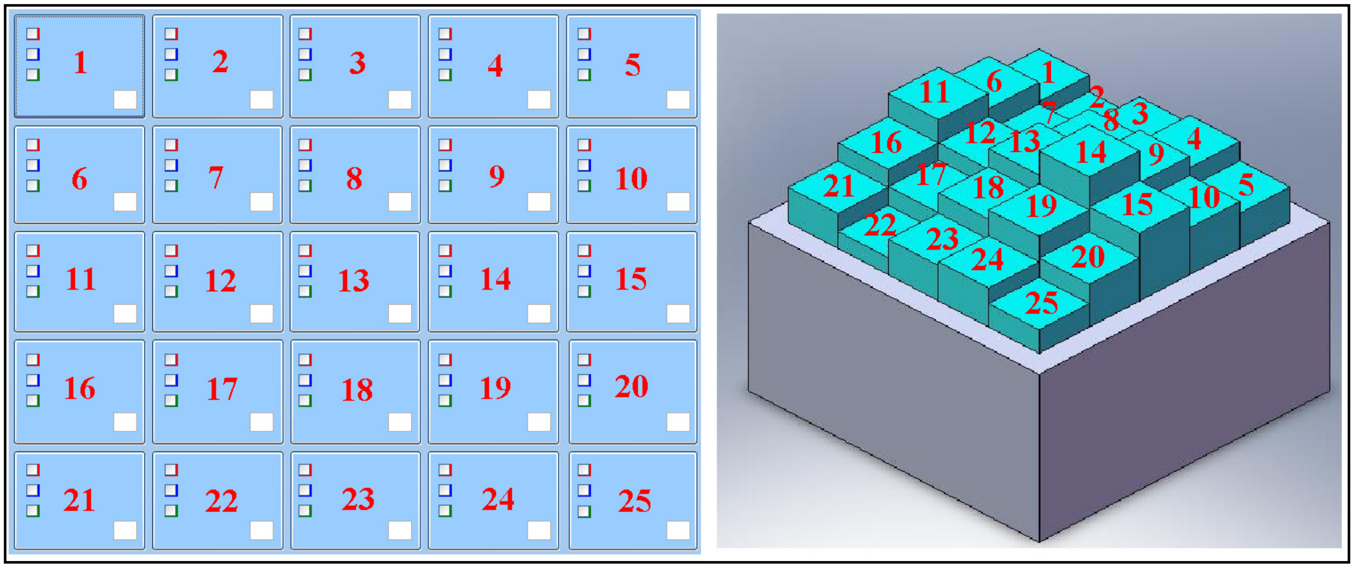

Figure 10: Cell Matrix in SUI and the Corresponding Crystal Bars

6.1.2. INDIVIDUAL CELL FEATURES

Each cell contains three checkboxes and a textbox as shown in Figure 11. The three checkboxes are designed for the users to choose which color of the tri-color LED (red, blue or green) embedded in the crystal bar should be turned on. Furthermore, the textbox is used to notify the users about the amplitude, ranging from 0% to 100%, of the crystal bar. As depicted in Figure 10, red color of the tri-color LED is chosen to be on, and the current amplitude of the crystal bar is 65% of the total height.

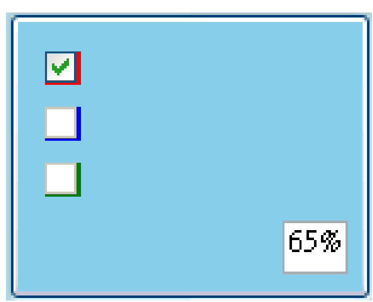

Figure 11: Color Checkboxes and Amplitude Display

The amplitude of the crystal bars can be increased and decreased by pressing the right and the left buttons of the mouse, respectively. The background color of the cell button will change to green if the crystal bar is rising, and to red if the crystal bar is falling, as shown in Figure 12.

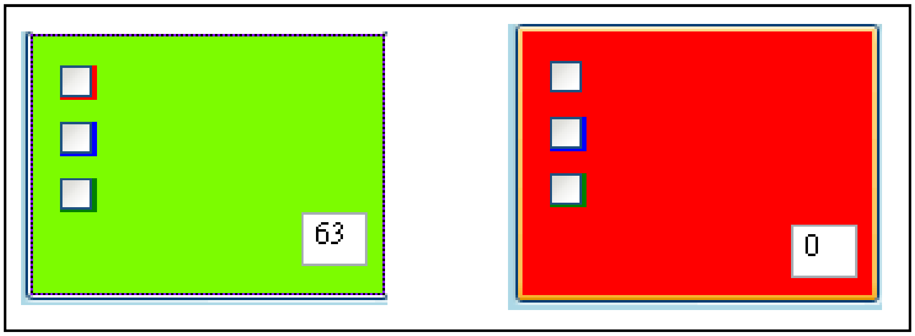

Figure 12: Background Color Indicating Increasing and Decreasing Amplitude

6.1.3. CONTROL PANEL

Control panel consists of three buttons, three checkboxes and one textbox as shown in Figure 13. The three buttons are “Clear Amplitude”, “Save” and “Load”, and their purposes are implied in their names.

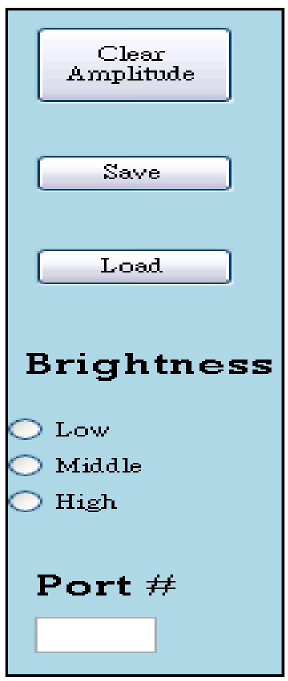

Figure 13: Control Panel

6.1.3.1. CLEAR AMPLITUDE, SAVE AND LOAD BUTTON

The “Clear Amplitude” is to set the amplitudes of all crystal bars to 0%. The “Save” and “Load” buttons are used to save/load the crystal bar amplitude and the LED configuration defined by the users.

6.1.3.2. CHECKBOX

The three checkboxes are designed to control the brightness of the LEDs. The levels of brightness are “Low”, “Middle” and “High”.

6.1.3.3. TEXTBOX

The “Port #” textbox is designed for the users to enter the COM Port that they want to use for the connection between SUI (computer) and *DreamBox*.

6.2. SIMPLE USER INTERFACE

Compared with static mode operation, a simpler user interface is necessary to provide users a proper and concise guide to use the functionalities in dynamic mode. A 122 x 32 dot addressable LCD is used for this purpose. The following figure shows a welcome message shown on the LCD when *DreamBox* is powered up.

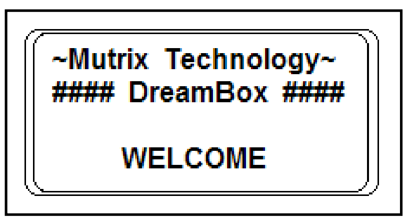

Figure 14: Welcome Message on DreamBox LCD

6.2.1. MENU DISPLAY

The options for *DreamBox* dynamic mode functionalities are displayed on the LCD screen, which is capable of display 20 characters in a row and total of 4 rows of characters. A sample menu display design is show below.

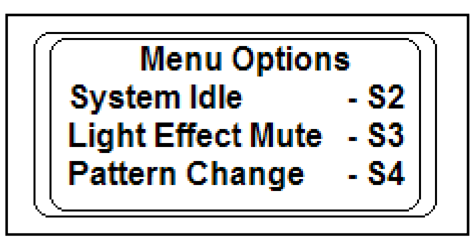

Figure 15: Menu Display

6.2.2. PUSH BUTTON SWITCH

Four push button switches help users select the options when *DreamBox* is working in dynamic mode. All options displayed on the LCD are followed by the switch labels, as shown in Figure 15 above.

6.2.3. LED MODE INDICATOR

Two on-board red LED will be used to indicate the operation mode of *DreamBox*.

7. MOTOR CONTROL SYSTEM

7.1. MOTOR CONTROL UNIT

One of the *Dreambox* application requirements is to move 25 crystal bars simultaneously. Lynxmotion SSC-32 Servo Controller can control up to 32 servo motors at the same time [6]. Lynxmotion SSC-32 Servo Controller is shown in Figure 16.

Figure 16: Lynxmotion SSC-32 Servo Controller

"The Lynxmotion SSC-32 Servo Controller has a high resolution (1uS) for accurate positioning extremely smooth moves. The motion control can be immediate response, speed controlled, timed motion, or a combination."[5] The Lynxmotion SSC-32 Servo Controller is capable of "Group Move". "It allows any combination of servos to begin and end motion at the same time, even if the servos have to move different distances."[5] Atmel ATMEGA 168-20U is the microcontroller of the Lynxmotion SSC-32 Servo Controller board. The microcontroller controls four shift registers to generate the pulse for the servo motors as shown in Figure 17.

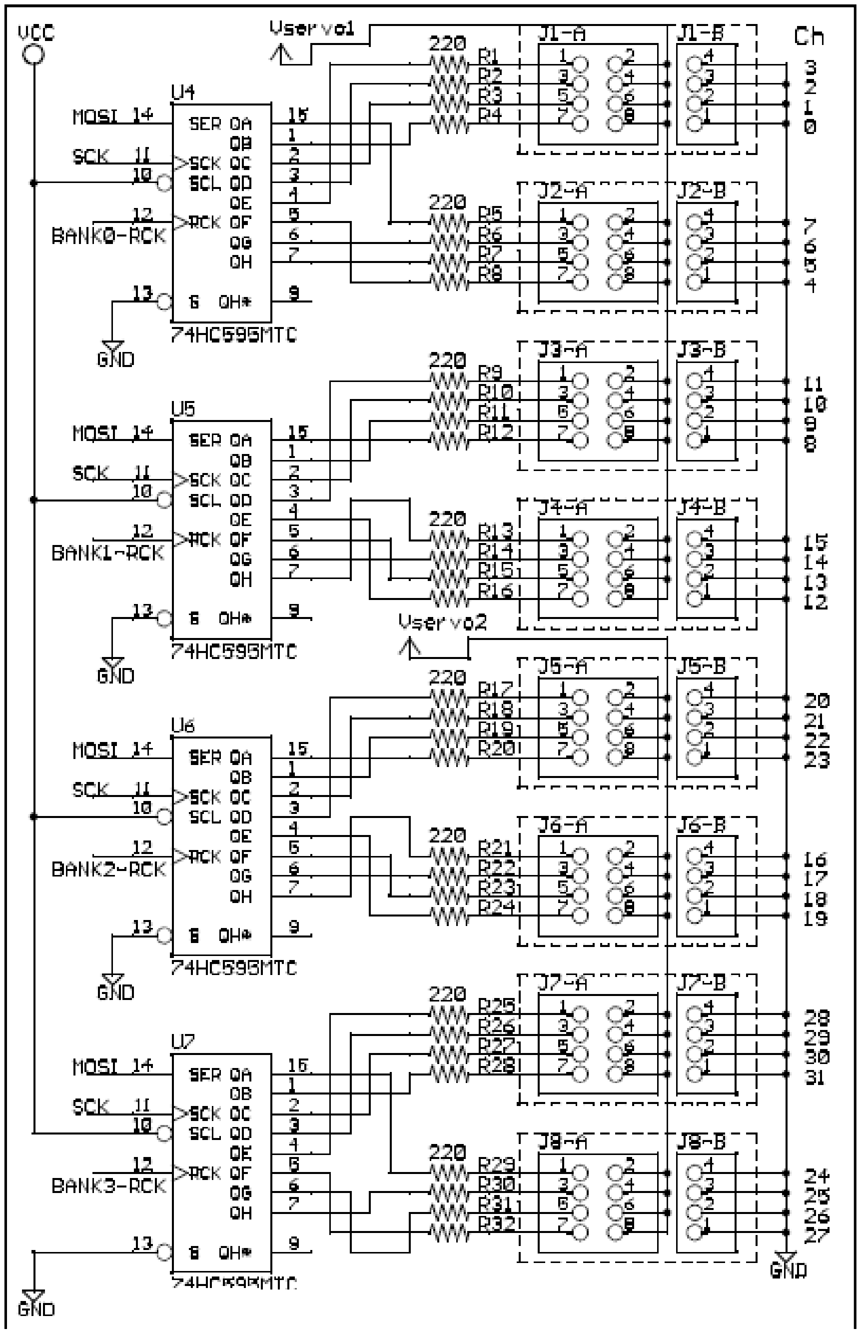

Figure 17: Circuitry of Shift Resgisters for Pulse Width Generation

The signal, MOSI, SCK and BANK1…4-RCK are controlled by Atmel ATMEGA 168-20U.

7.2. CONTROL COMMAND FORMAT

Control command is sent to Lynxmotion SSC-32 Servo Controller by serial port. Table 1 shows the servo movement control command format.

Table 1: Servo Movement Control Command Format

# <ch> P <pw> S <spd> … # <ch> P <pw> S <spd> T <time> <cr>	
<ch>	Channel number in decimal, 0-31
<pw>	Pulse width in microseconds, 500-2500
<spd>	Movement speed in uS per second for one channel (Optional)
<time>	Time in mS for the wntire move, affects all channels, 65535 max (Optional)
<cr>	Carriage return character, ASCII 13 (Required to initiate action)
<esc>	Cancel the current action, ASCII 27

For Example, "#6 P1500 S750 <cr>" will move the servo on channel 6 to position 1500 (90 degrees) at a rate of 750µS per second. "For a better understanding of the speed argument, consider that 1000µS of travel will result in around 90 degrees of rotation. A speed value of 100µS per second means the servo will take 10 seconds to move 90 degrees. Alternately, a speed value of 2000µS per second equates to 500mS (half a second) to move 90 degrees." [6] An example using time argument: "#6 P1500 T1000 <cr>". The command will move the servo on channel 6 to position 1500 (90 degrees) in 1 second.

The speed and time commands can be combined if desired. The speed for each servo will be calculated according to the following rules:

1. All channels will start and end the motion simultaneously.
2. If a speed is specified for a servo, it will not move any faster than the speed specified; however, it might move slower if the time command requires.
3. If a time is specified for the move, then the move will take at last the amount of time specified; however, it might take longer if the speed command requires. [6]

"Group Move" can be generated by cascading commands for each servo with no <cr> character in between. For example, "#5 P1600 #17 P750 T2000 <cr>" moves the servo motor in channel 5 to position 1600 and the servo motor in channel 17 to position 750 in 2 second.

7.3. POWER CONSUMPTION

Since all the servos may move simultaneously, current consumption will be an issue. Parallel power supplies from the power cord can be a solution to this issue. The number of parallel power supplies needed for the system will be determined based the result of the tests. These tests will be explained in detail in Test Plan Section.

8. MECHANICAL SYSTEM

8.1. OVERVIEW OF THE MECHANICAL SYSTEM

The crystal bars move up and down according to the music rhythm in dynamic mode or the user operations in static mode. Servo motors are used to control the movement of the crystal bars. In our design, the output of the mechanical system should be linear; however, the servo motors can only supply rotary torque. Therefore, a gear and a rack are needed to transfer the force from rotary to linear in order to drive the linear motion of the crystal bars as shown in Figure 18.

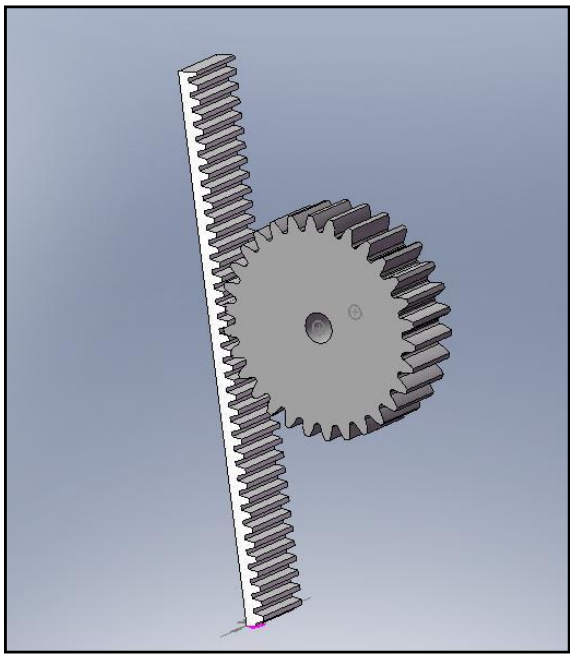

Figure 18: Gear and Rack Set Translate Rotary Movement to Linear Movement

Due to the limited space under the crystal bar, the size of the gear cannot exceed 2.5 cm in diameter. Moreover, servo motors are used to provide the rotary movement in the system, but servo motors can only a turn up to 180 degrees. As a result of the small gears and servo motors, the linear displacement is limited. Therefore, a double molded gear with different number of teeth is needed to change the turn ratio. One 180-degree turn of the gear directly driven by the servo motors can result in a 360-degree turn in the double molded gear. A 360-degree turn in the double molded gear can provide sufficient linear displacement for the crystal bar. The overall mechanical system is shown in Figure 19.

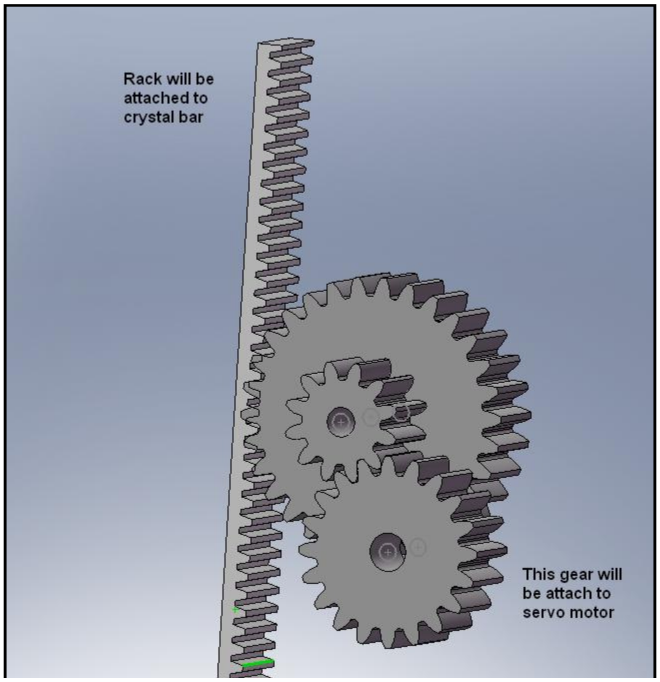

Figure 19: Overall Mechanical System

8.2 SERVO MOTORS

Motors are needed to supply the force to move the crystal bars up and down. The movement of the crystal bars requires high precision to ensure the consistent appearances; therefore, servo motors are chosen for the mechanical system. Servo motor is commonly driven by pulses, which turns from 0 to 180 degrees according the width of the input pulse, ranging from 1.5ms to 1.75ms, in a 20ms-period as shown in Figure 20.

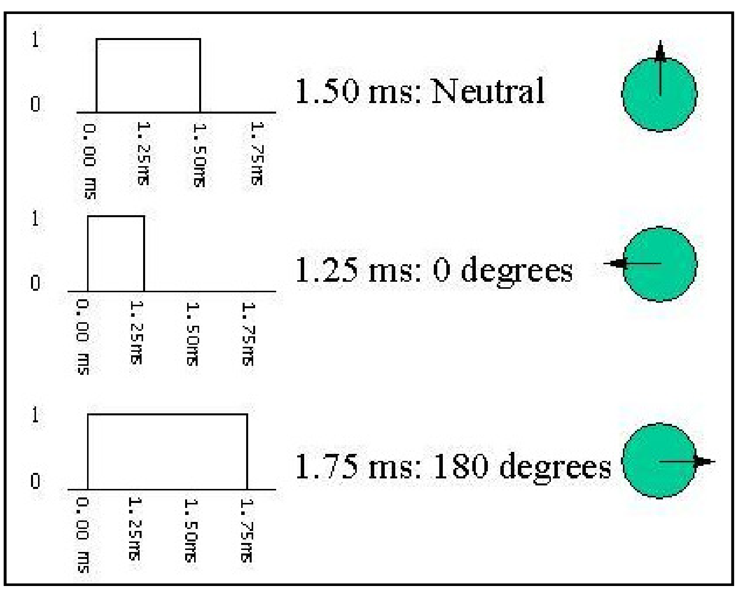

Figure 20: Turning Angle of a Servo Motor Depending on Input Pulse Width [7]

In our application, TowerPro SG-50 Micro Servo is used for the mechanical system. The servo motor is shown in Figure 21.

Figure 21: TowerPro SG-50 Micro Servo Motor [8]

The size and the speed of this servo motor are the main reasons why this type of servo motors is chosen for our mechanical system. The size of TowerPro SG-50 is 21.5mmx11.7mmx25.1mm; and the speed of TowerPro SG-50 is 0.3sec/60degree (4.8V). It is small enough to fit in the space under the crystal bar, and fast enough to create real time movement with the music.

8.3. CRYSTAL BAR

The material to build the crystal bars must be light because the stall torque of TowerPro SG-50 is 0.6kg/cm. Plastic glass was chosen to build the crystal bars. The size of the crystal bar will be 27mmx27mmx90mm as shown in Figure 22.

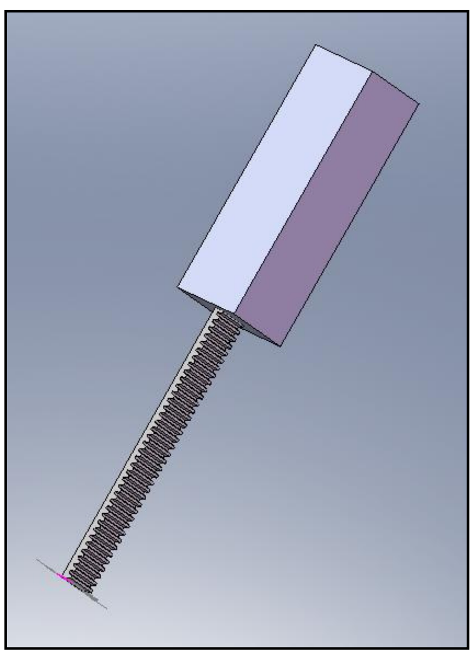

Figure 22: Crystal Bar

8.4. GEARS

The mechanical system needs one standard gear, one double molded gear and one rack as shown in Figure 23 to Figure 25 respectively.

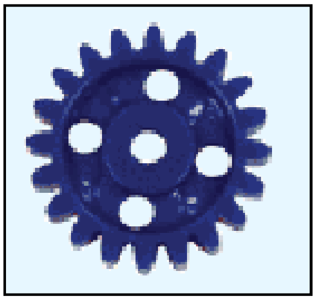

Figure 23: Standard Gear Connected to Servo Motor

Figure 24: Molded Gear for Changing Turn Ratio

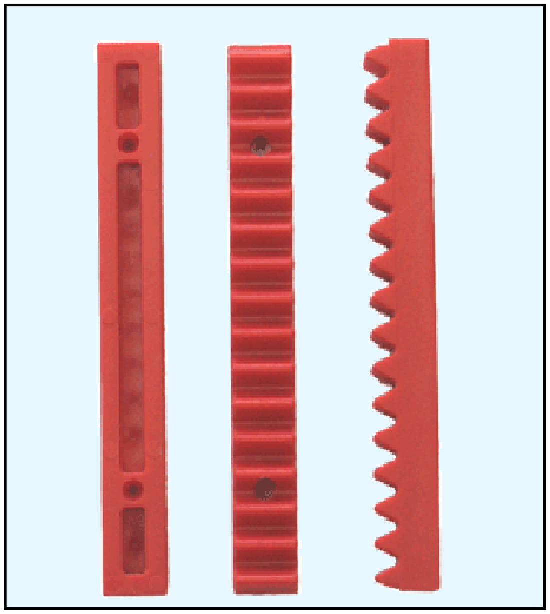

Figure 25: Gear Rack for Translation of Rotary Movement to Linear Movement

The standard gear has 20 teeth and will be mounted on the servo motor. The small gear of the molded gear has 10 teeth, and the large gear of the molded gear has 30 teeth. The standard gear will match with the small gear of the molded gear, and the large gear of the molded gear will match with the gear rack. A 180-degree turn of the standard gear will result in a 360-degree turn of the molded gear. The diameter of the large gear of the mould gear is 2.5cm. Therefore; the circumference of the large gear will be:

$$L = \pi d = \pi * 2.5cm = 7.85cm$$ **Equation 3**

The linear displacement of the gear rack will equal to the circumference of the large gear of the molded gear; therefore, the maximum displacement of the crystal bars will be 7.85cm within the design expectation.

9. LED LIGHTING SYSTEM

LED lighting system provides colorful light effects for *DreamBox*. The follow picture shows an example of LED light effects in a 5x5 square matrix.

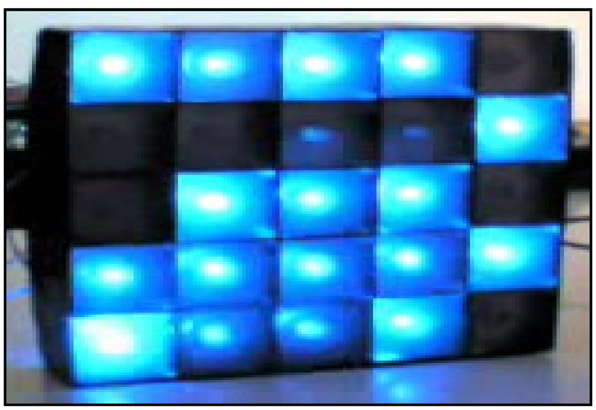

Figure 26: LED Light Effect

There are 25 tri-color LEDs in our system and each tri-color LED has 4 pins: red, green, blue and common ground. The figure below shows the picture and specification of the tri-color LED we chose. By setting different color pins to high voltage level and common pin to low voltage level, we can get different lighting colors.

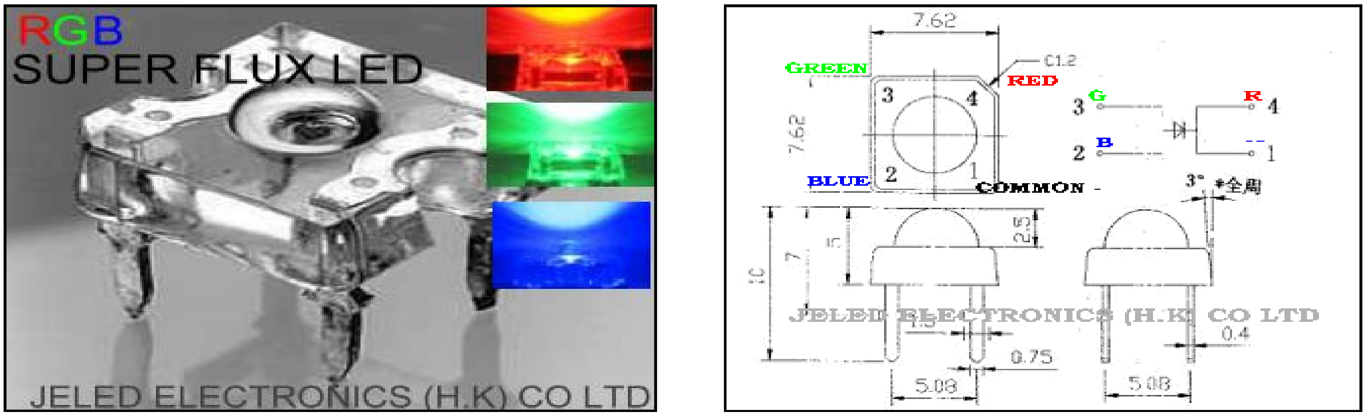

Figure 27: Tri-color RGB LED

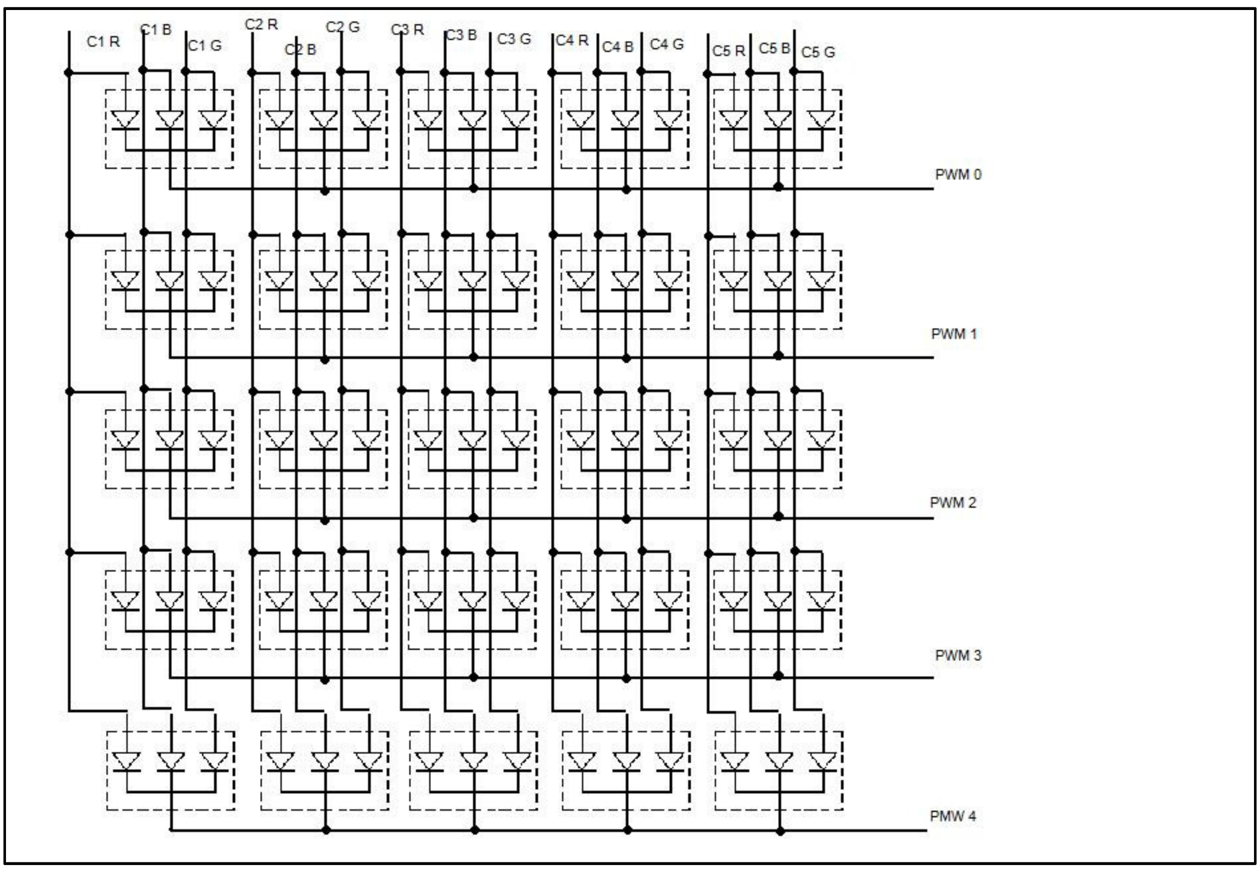

Figure 28: LED Control System Schematic

As the above figure shows, we put 25 tri-color LEDs in a 5X5 matrix. To indicate each LED, we use 15 general output pins (C1_R to C5_G) on the board for column and color selection. 5 pulse-width-modulated (PWM) output pins (PWM_0 to PWM_4) are required for our system in order to select rows and control LED brightness. LED will light up if the column and color selection pin is high and row selection PWM pin is low. To control the brightness of LED, we can simply adjust PWM's duty cycle. If the PWM pin is driven high longer, LED will be dimmer. For the waveform shows below, the first waveform will generate brighter light.

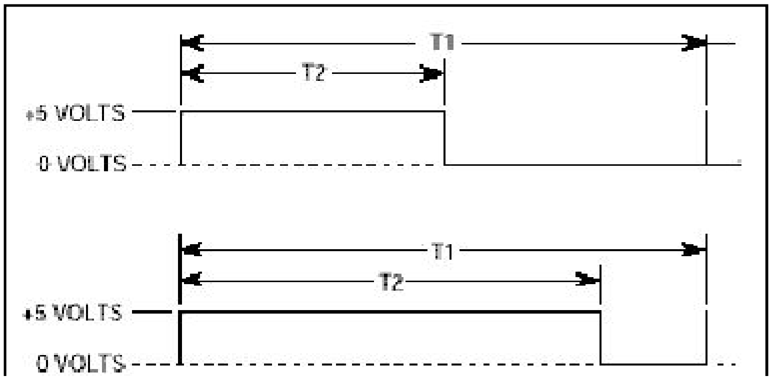

Figure 29: PWM Duty Cycle

Initially, we set all 15 column and color pins to be high, and set PWM duty cycle to be 100%. Then, the LED control system will receive control commands in the order of row, column, color and brightness. The system will use those four variables to control the LED. The following table shows the format of the LED control command.

Table 2: LED Control Command Format

$ <row> <coln> <clor> <bri> <cr>	
<row>	Row number in decimal, 0-5
<coln>	Column number in decimal, 0-5
<clor>	Color of the LED
<bri>	Brightness of the LED
<cr>	Carriage return character, ASCII 13 (Required to initiate action)

10. COMMUNICATION

CCU is the center processor unit of the entire system. It needs to communication with the computer as well as the motor control unit of MMS. CCU can support up to two UARTs to control RS232 serial ports, which will be the main communication method with the computer and the motor control unit. The schematic of the UART connection is shown in Figure 30.

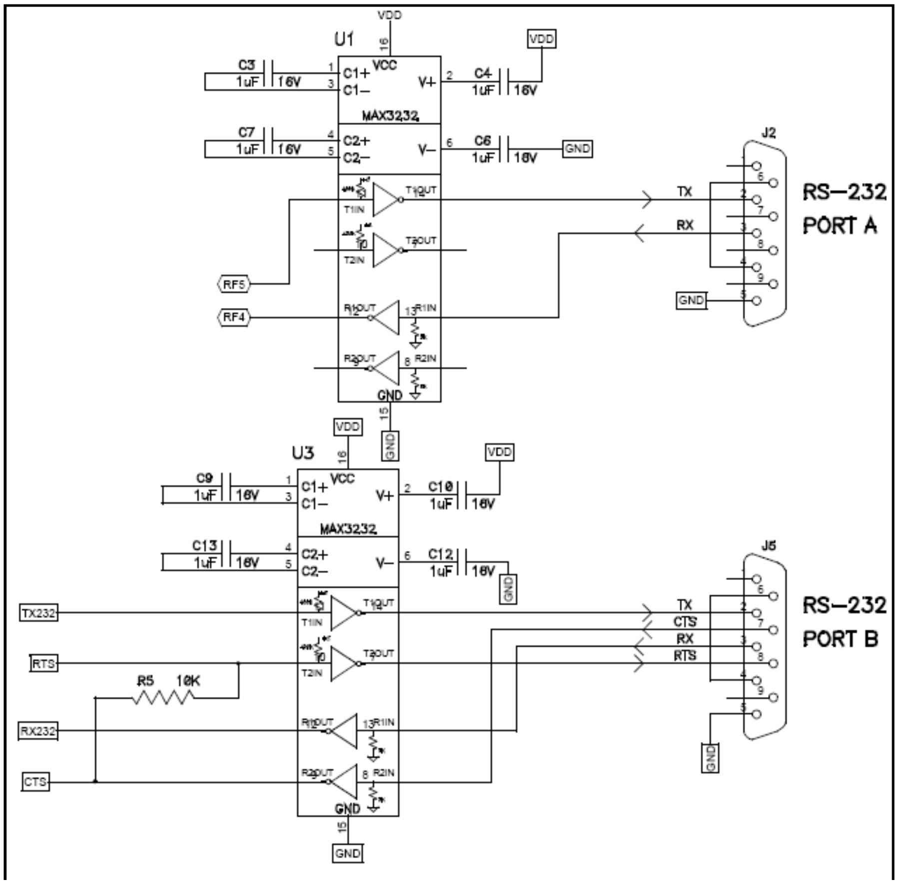

Figure 30: Schematic of UART Connection

10.1. COMMUNICATION BETWEEN COMPUTER AND CENTER CONTROL UNIT

UART 1 is set up for the communication between CCU and the computer. To increase the connection flexibility, a RS232-to-USB adaptor is connected between UART1 to computer. The communication between CCU and the computer happens when the system is working in static mode. Motor control commands and LED control commands will be sent from the computer to CCU. CCU only buffers and processes the LED control commands, and directly passes the motor control commands to the motor control unit. Two types of commands are differentiated by unique command headers. Motor control command starts with <#> character; on the other hand, LED control command starts with <$> character. Both types of commands end with <cr> character. The program flowchart is shown in Figure 31.

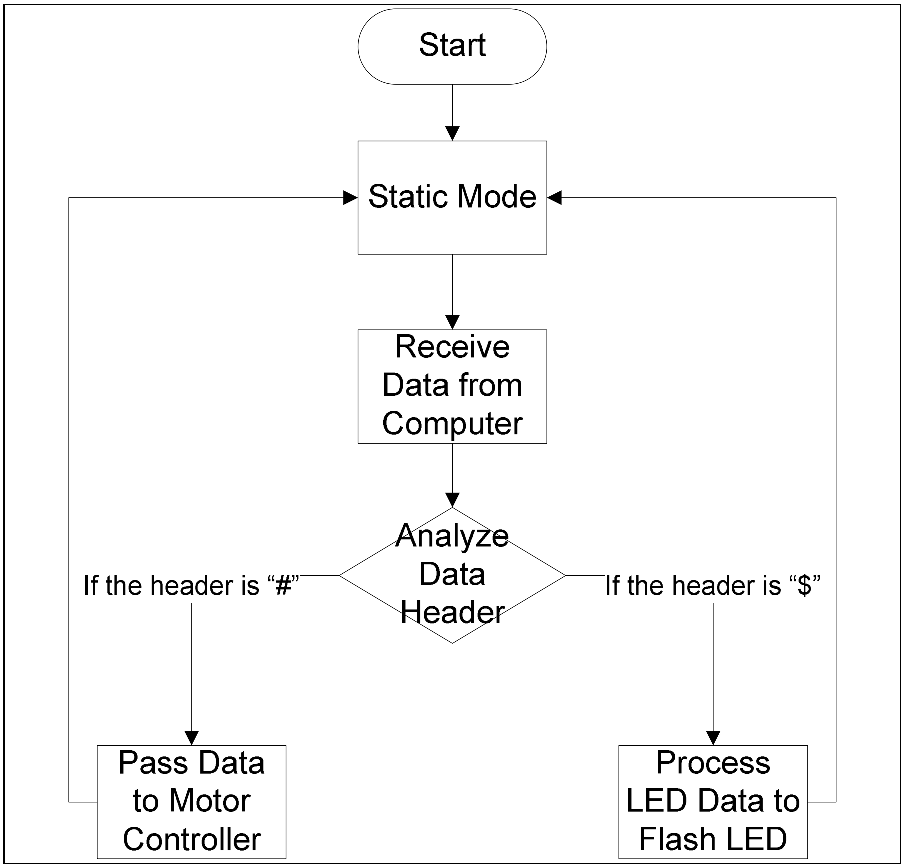

Figure 31: Program Flowchart for Communication Protocol between CCU and Computer

The error rate of transmission is considerably small in such a short transfer distance. Moreover, one or two commands with error will not make great impact on the overall performance of the system. Error is possibly not visible to the user and can be easily ignored.

10.2. COMMUNICATION BETWEEN CENTER CONTROL UNIT AND MOTOR CONTROL UNIT

UART 2 is set up for the communication between CCU and the motor control unit of MMS. The communication between CCU and the motor controlled unit happens in both dynamic mode and static mode. In static mode, motor control commands will be passed from computer to the motor controlled unit as mentioned in the previous section; in dynamic mode, CCU will send motor control commands to the motor control unit based on the rhythm of the music. Since audio processing will be implemented in real time, the motor control commands will be continuously sent out. The program flowchart is shown in Figure 32.

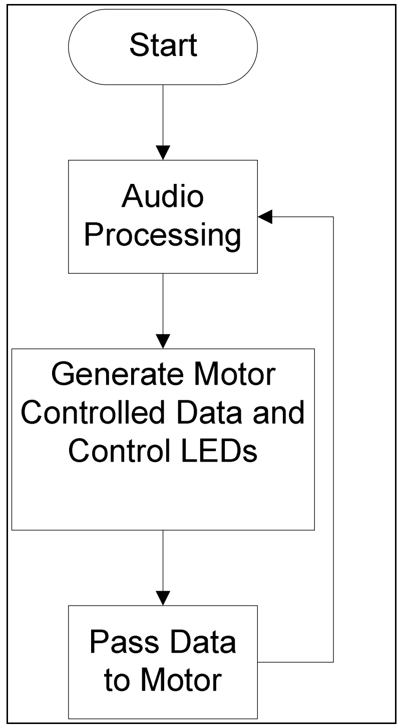

Figure 32: Program Flowchart for Communication Protocol between CCU and Motor Control Unit

Commands will be continuously updated, so the error checking mechanism is not needed. Error of previous command will be overridden by the next command.

11. MEMORY ORGANIZATION

Pre-designed crystal bar patterns and LED light effects will be stored in the on-chip memory. Based on the format of the motor control command and the LED control command, the total number of bytes required by a set of motor and LED commands for one cell of *DreamBox* is 15, thus, to the memory space needed for a set of commands for the entire matrix is 375 bytes.

The following figure shows the user memory allocation of dsPIC30F6014A microcontroller.

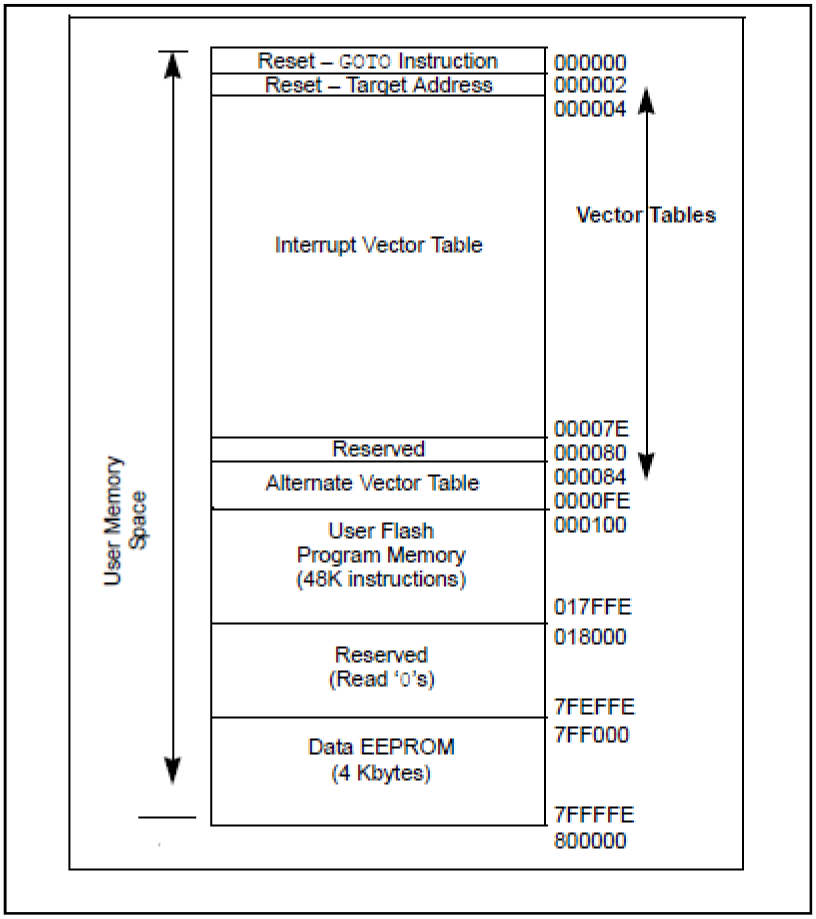

Figure 33: User Memory Allocation of dsPIC30F6014A Microcontroller

Starting from 0x7FF000, the microcontroller provides 4 Kbytes of EEPROM memory for data storage purpose. Up to 10 frames of control commands can be stored in EEPROM. In our design estimation, each set of crystal bar pattern and LED lighting sequence requires 5 frames of control commands. Thus, for the prototype of the system, we will design two set of *DreamBox* operation sequence to demonstrate the concept of visualized musical experience.

12. SYSTEM TEST PLAN

This section will describe the overall system test plan for *DreamBox* musical entertainment system. The product is in the prototyping stage; therefore, a set of tests must be performed to ensure that the design has met the specification of each component. The tests will be carried out in each stage throughout the development cycle. The test plan can be divided into three subsets of test: software, hardware and mechanics. Each subset of tests works closely together. For example, in order to test LED circuitry design, test software must be written to control the LED circuit.

The procedure of each test will be listed out in the test and verification plan document. After each test had been performed, test data need to be collected and compared with the specification of the components. At the final stage of the development cycle, the prototype will be tested by potential users to obtain feedback about the appearance and the functionalities of *DreamBox* musical entertainment system.

The technical details of each test will be explained in the design specification. A brief description of each test will be list below:

Software Test:

1. Audio Filter Test
   - Frequency Display of the Audio
2. LED Control Software Test:
   - LED Light up Sequence
   - LED Color Control
3. Motor Control Software Test:
   - Speed of Motor Control
   - Turning Angle Control
4. Computer Interface and Communication Test:
   - Data Flow from Computer to the Processers

Hardware Test:

1. Connectivity of Circuitry
   - LED Circuitry Connection
   - Communication Connection of each Processors
2. Power Consumption of each PCB
   - Servo Control PCB
   - Audio Process PCB
   - LED Control PCB

Mechanical Test:

1. Servo Motor Movement Test:
   - Turning Speed vs Different Load
   - Maximum Load
   - Start-up Current
   - Current Consumption vs Different Load
2. Gear Movement Test:
   - Speed of the Movement of Crystal Bar vs Speed of the Motor
   - Verification of Ratio of the Gears
   - Maximum Load of the Gears
3. Stress Test:
   - Stability of the Overall Mechanical System

13. CONCLUSION

The design for our *DreamBox Musical Entertainment System* prototype is defined throughout this document, including the Central Control Unit (CCU), Motor Mechanical System (MMS) and the Software User Interface (SUI). Design for each module is described, and an informative test plan is included. The project is progressing well and a functional prototype is anticipated to be built by April, 2009.

14. REFERENCE

[1]. Mutrix Technology, "Functional Specification for *DreamBox* Musical Entertainment System", Simon Fraser University, Burnaby, BC, Canada, February 2009.

[2]. *dsPICDEM GP 1.1 Development Board User Guide*, page 67. Microchip: 2006.

[3]. Microchip.com, dsPIC30F6014A description, 2008. [Online]. Available: http://www.microchip.com/wwwproducts/Devices.aspx?dDocName=en024766. [Accessed: February 27, 2009]

[4]. Frederic Patin, "Beat Detection Algorithm", Feb 21, 2003. [Online]. Available: www.flipcode.com/misc/BeatDetectionAlgorithms.pdf. [Accessed: February 28, 2009]

[5]. Informantion, *Servo Motor Controller*. [Online]. Available: http://www.lynxmotion.com/Product.aspx?productID=395&CategoryID=52. [Accessed: February 28, 2009]

[6]. *SSC-32 Manual*. [Online]. Available: http://www.lynxmotion.com/images/html/build136.htm. [Accessed: February 28, 2009]

[7]. *Wha's a servo*. [Online]. Available: http://www.seattlerobotics.org/guide/servos.html. [Accessed: February 26, 2009]

[8]. *Towerpro SG50 SG 50 5g Micro Mini Servo*. [Online]. Available: https://www.toysonics.com/towerpro-sg50-sg-50-5g-micro-mini-servo.html. [Accessed: February 26, 2009]