November 16, 2010

Mr. Mike Sodjerma
School of Engineering Science
Simon Fraser University
8888 University Drive, Burnaby, BC
V5A 16S

Re: Design Specification of a Music Recognition Device (The Maestro™)

Dear Mr. Sodjerma,

Please find enclosed Harmony Inc.'s design specification for a portable sheet music scanner. This device will be a prototype for a range of new handheld music recognition technologies. The goal of this project is to build a handheld device that converts sheet music into sound.

The purpose of this design specification is to unambiguously describe the technical aspects of the Maestro™ in its proof of concept phase only, with some aspects of the production models being discussed.

This document is considered high priority by some of Harmony Innovations' investors and will serve as the basis of the project that they have agreed to fund.

Please feel free to contact me personally with any questions you may have at 778-320-5346, or by email at cris.panaitiu@gmail.com

Sincerely,

Cristian Panaitiu
Chief Executive Officer
Harmony Innovations Inc.

Enclosed: Technical Specification for "The Maestro"™ music education aid

2010

Design Specification

The Maestro™

Portable Sheet Music Scanner and Player

MISSION STATEMENT

At Harmony Innovations, the future is our passion.

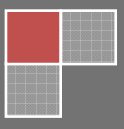
By combining musical education with the technological advances of today, Harmony Innovations strives not only to enhance the quality and accessibility of musical education for all students; but also to provide support and technology as a partner to many up and coming artists.

Technology has enormous potential to enhance our lives and this is the guiding principle behind Harmony's comprehensive approach to musical education.

Date Submitted: November 10, 2010

Submitted to:
Dr. Andrew Rawicz (ENSC 440)
Michael Sjoerdsma (ENSC 305)

Project Team:
Sean Edmond (301026670)
Nikola Cucuk (301033241)
Veronica Cojocaru (301055896)
Cris Panaitiu (301032665)



Executive Summary

Music has always been one of the crown jewels of human achievement. Not just as a form of entertainment, but as a universal language to express deeper meaning than can't be expressed in any other way.

Harmony Innovations Inc. recognizes the importance of this universal language and strives to create and develop new digital technologies to enhance the process of musical education. Designing digital electronic solutions for the enhancement of the educational experience that can be used by anyone is the guiding principle Harmony is based on and will help tomorrow's generation better understand music.

The Maesto™, a standalone device that can read and interpret sheet music is reaching maturity in the design phase.

Selecting our system peripherals has been a key component in moving the project forward. We have purchased an adequate camera and sound module based on our functional specification. The camera module provides a continuous YUV video stream that will be able to be polled by a microprocessor. The sound module is capable of buffering a MIDI file and decoding it to a standard headphone jack. These peripherals in addition to our software and memory requirements have guided us in the selection of our microprocessor.

The development board we've selected contains an Atmel 32-bit processor and 256MB of SDRAM which will be more than sufficient for our computing and memory needs. This board also supports the interfacing requirements for both the image and sound module. In addition the board has built-in LCD, and pushbuttons so there will be no need to purchase addition components for our user interface.

Development is well underway, and we have made significant headway in the isolated project components. The next design phase will be focusing on integration and testing. We are on target for completion mid December.

Table of Contents

List of Figures

List of Tables

Glossary

CMOS	Complementary Metal–Oxide–Semiconductor
CPU	Central Processing Unit
CSA	Canadian Standards Association. Non-governmental
DSP	Digital Signal Processing
IC	Integrated Circuit
I^2C	Inter-Integrated Circuit
I/O	Input/Output
IQM	Image Acquisition Module
LCD	Liquid Crystal Display
LED	Light-Emitting Diode
MIDI	Musical Instrument Digital Interface. A digital music storage standard, in use across all of the North American music industry
OCR	Optical Character Recognition
PCB	Printed Circuit Board
RGB	Red Green Blue color model
RoHS	Reduction of Hazardous Substances. Certification for environmentally friendly electronics manufacturing. RoHS compliant devices are manufactured without lead or mercury
SCCB	Serial Camera Control Bus
SD	Secure Digital Card
SMF	Standard MIDI File

SRAM	Static Random Access Memory
SPI	Serial Peripheral Interface
TRS	Tip Ring Sleeve
TWI	Two Wire Interface (Atmel's alternate name of I^2C)
USART	Universal Synchronous Asynchronous Receiver-Transmitter
USB	Universal Serial Bus
UL	Underwriter Laboratories. US Standards-setting organization
UVY	Color space in terms of one luma (Y') and two chrominance (UV) components
ZV Port	Zoom Video Port Format

1. Introduction

"Music education is the epitaph of human achievement and embodies the desire to understand that which we do not understand through that which we do" – Socrates

Throughout history, humanity has placed enormous value on the creation of music. The ancient Greeks, for example, believed that music is the 'subtle, yet complete understanding of human nature and the human existence'; devoting much time to the study and composition of music. Like the Greeks, every civilization, ancient to contemporary, has devoted enormous energies towards the development of music.

The effort to express through music is by no means exhausted today, especially with the advent of new technologies through which music can be more easily created. Musical education, however, has not benefited sufficiently from the progress made in digital music technologies; most learning is done through a process of demonstration and practice by a teacher for a student.

While a teacher should not be replaced, students can make their practice more efficient with the help of digital technology. Harmony Inc. is proud to present The Maestro™, a standalone device that will read and interpret sheet music.

The Maestro™ will consist of a simple user interface that will allow the user to configure the device. A camera will allow the user to scan and buffer the sheet music that they want to playback. Once they have finish scanning, they can play the music back. Music will be outputted to a standard headphone jack.

1.1 Scope

This document is a complete specification of all technical aspects of the Maestro™ and its incorporated technologies. It is a definitive list of all capabilities and features that will be contained within the device; to the extent that future work on this device should be guided along the outlined specification contained herein.

Some production specifications will also be discussed. References to production model specifications will be made as required for important specifications such as testing and important milestone specifications.

1.2 Intended Audience

The design teams, as well as the marketing and end customer support teams will use the specification outlined in this document as a functional framework based on which they will be able to create engineering platforms, marketing solutions, as well as customer support solutions that will fully realize the requirements of the Maestro™ project.

2. System Specification

The Maestro™ will operate with three distinct user operation mode, buffer music mode, configuration mode and playback mode.

Buffer music mode allows the user to scan in bars of standard sheet music (one bar at a time). The camera assembly is placed over the bar of music that you which to scan. Once scanning in engaged (by holding down a pushbutton), the camera assembly can be moved slowly across the page until the desired amount of music has been scanned. Scanning is disengaged by depressing the pushbutton. While scanning is engaged, an LED will light the scan surface. The resulting images will be interpreted and buffered as a MIDI file by a microprocessor.

Configuration mode will allow the user to change the key signature, clef, tempo and volume of the music. An easy to use user interface using pushbuttons, a joystick and a rotary dial will allow for easy configuration of these parameters.

Playback mode exports the MIDI file to a sound module that will make the music audible through a standard 1.8”, 3.5mm headphone jack.

3. Overall System Design

This section is meant to overview the system level design of the Maestro™. Therefore this section is limited to describing the system level design decisions. This section also provides an overview of the interfacing with subsystems.

3.1 General Design

The general design of Maestro™ proof of concept model is illustrated by Figure 1. The camera assembly will connect to the microcontroller via ribbon cables. The microcontroller, sound module and headphone jack will be mounted on a separate board and sturdy connections will be made between components.

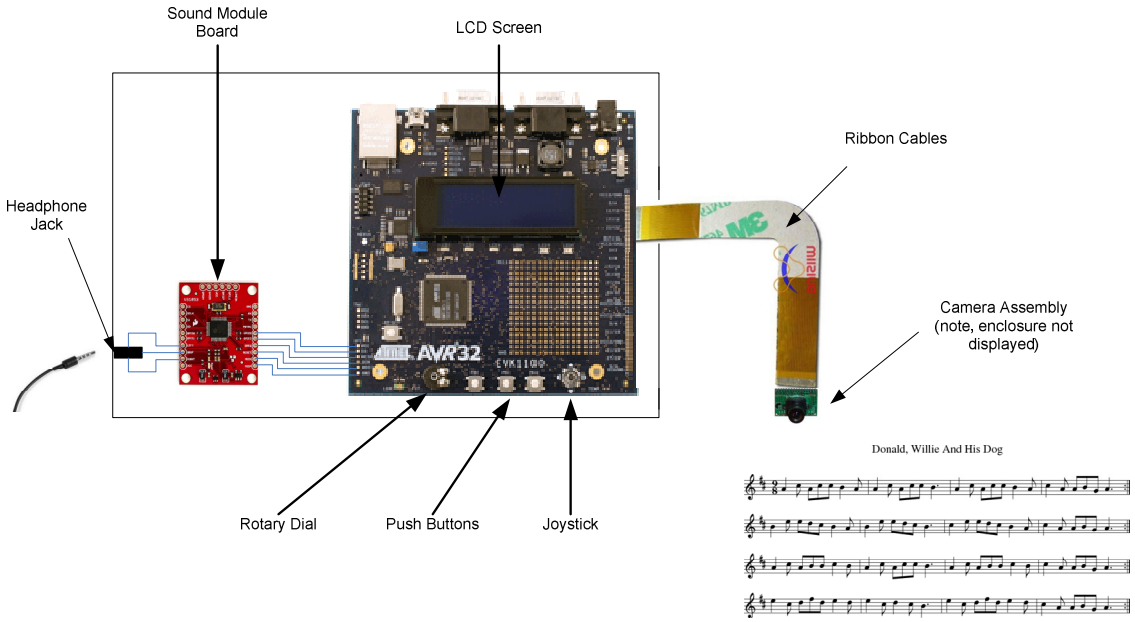


Figure 1 - Maestro Mock-up

3.2 Micro-controller

The choice of the micro-controller was crucial for Harmony solution's sheet music portable scanner. At the start of the selection process, many factors have been taken into account such as speed, number of I/O's, availability along with the development tools,

reprogrammability, supported interfaces, SRAM size and DSP processing options. Atmel AVR32 processor was short listed because it satisfies requirements and several team members had exposure to this processor family in previous projects.

EVK 1100 has been selected as the evaluation kit for this project. This board hosts the AT32UC3A512 processor. It is equipped with a rich set of peripherals, memory, and makes it easy to try the full potential of the AVR32 devices. Running at 66Mhz this CPU will be fast enough to interact with the image acquisition module and the sound module as well. Memory of 512KBs will be sufficient for image processing and c-code storing. Interfaces such as JTAG, Nexus, USART, USB 2.0, TWI, SPI SD and MMC Card Reader will be used in our project to interface different modules.

3.3 High-Level System Design

A high-level diagram of our system can be seen in Figure 2. In this diagram data flows from left to right. Note that our system is using off-the-shelf components, so signal conditioning is contained within the modules. Also note that memory requirements will exceed the on-chip memory of our microprocessor. Therefore, we will have to interface with external memory.

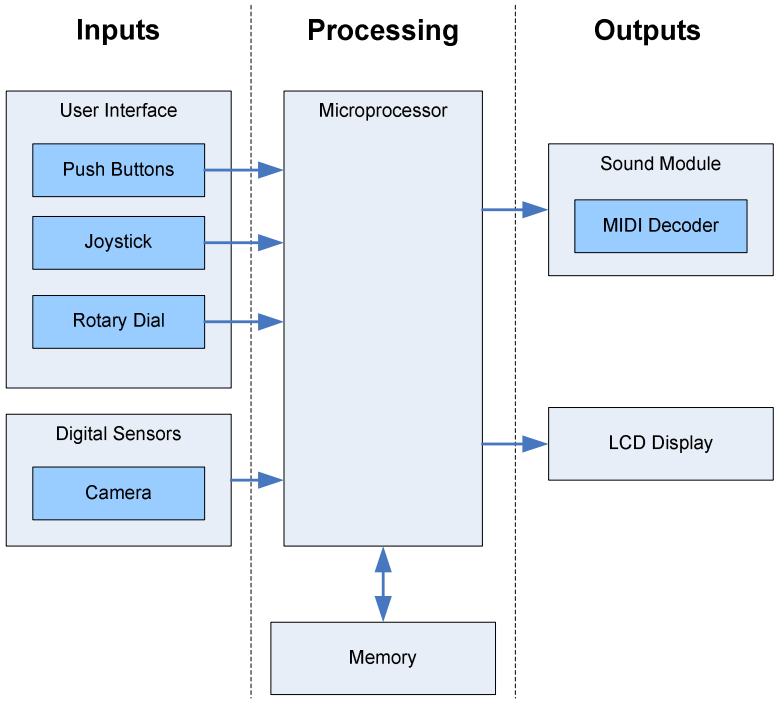


Figure 2 - High Level System Diagram

3.4 Electronics System

Three distinct electronic components had to be considered in our electronic system: the camera, sound module and microcontroller. This section describes the selection of the three components at the system level.

In designing the electronics system, it was important to choose the camera module and sound module first. The goal in these parts' selection was to select off-the-shelf components with support for standard interfaces like spi and RS-232. The microcontroller selection would then fit around the interface requirements, while fulfilling the processing requirements.

The decision was made to choose the c3188a camera module. This module contains the OV7620 camera mounted on a breakout board. This camera provides a continuous YUV video stream on a 16-bit parallel bus including HREF and VSYNC signals (the YCrCb resolution being 4:2:2). Since the requirement is for a monochromatic application, only

the Y component of the YUV stream (8 bits) is used. The camera module's programming interface uses an I^2C bus. Although interfacing with the parallel data bus will be more challenging, this camera has a significantly higher polling frequency than most other cameras.

For the sound module the proof of concept model will use the VS1053b IC mounted on a breakout board from sparkfun electronics (BOB-08954). Packaged into this device is a MIDI decoder that supports format 0 MIDI files. It also provides a stereo output for a headphone jack capable of driving a 30Ω load. The data and control interface share an SPI interface but they both have separate chip select lines. SPI protocol will be implemented using the chip's dedicated ports, but the data interface chip select will be using a GPIO port. The sound module also outputs a DREQ signal to indicate when a transfer can occur, which will also use a GPIO port on the development board.

Given the camera and sound module selection, the microcontroller needs to support I^2C, and SPI standard interfacing. We will also need 10 GPIO ports for the camera module, and 3 GPIO ports for the sound module. We will also have significant processing and storage requirements, since our project is involving image processing. Given these requirements we selected the EVK1100 development board.

The EVK1100 development board has:

- AT32UC3AO512, 32-bit microcontroller
- SPI, TWI (Atmel's version of I^2C), many other IO ports for custom interfacing
- LCD display, pushbuttons, joystick and rotary dial for user interface
- On board 3.3V voltage regulator
- 256 MB SDRAM for storage, connected to EBI interface (MT48LC16M16A2)
- Large open source community with access to drivers, etc

A block diagram of the electronics system is shown in Figure 3.

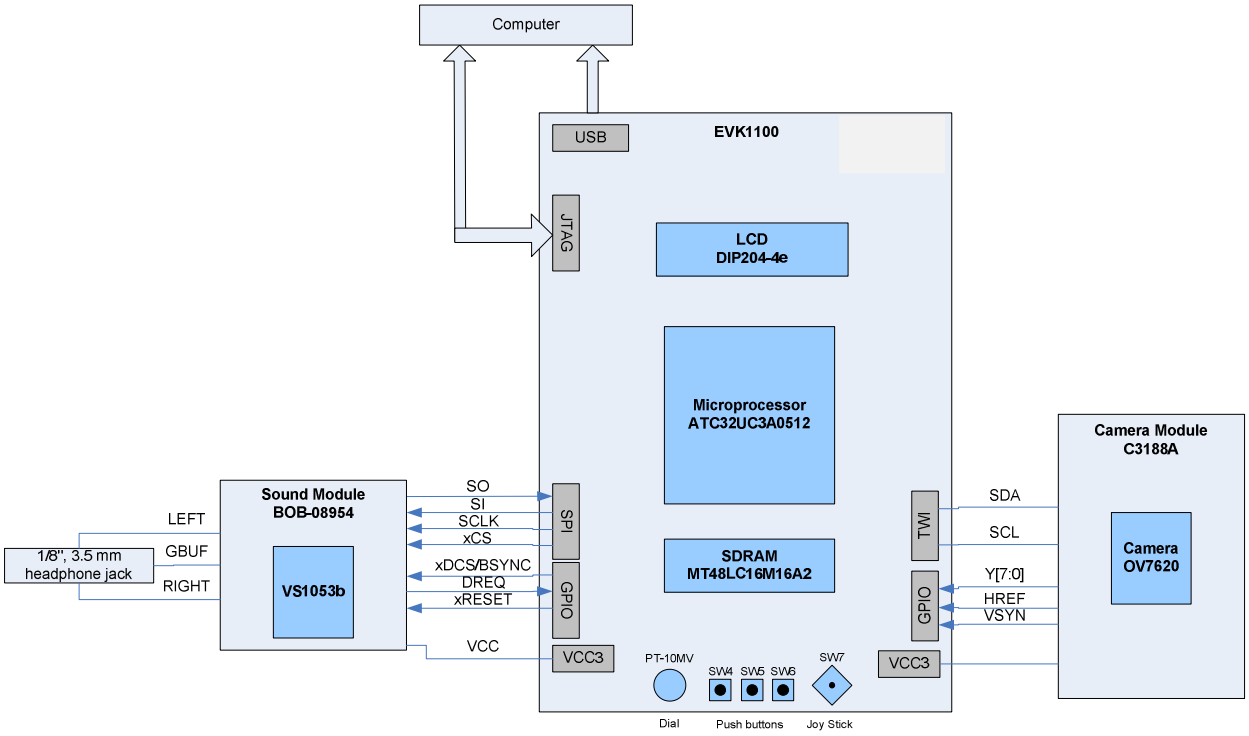


Figure 3 - Electronics System Block Diagram

3.5 Power Considerations

The EVK1100 board can be powered over USB or with a separate 8-20V DC power supply plugged into the board's power socket. We would like our device to run without a connection to a computer, so we will be using the power socket with a power supply. We will use either a CSA compliant power supply, or a 9V battery.

The board contains a 3.3V voltage regulator and broadcasts this voltage as VCC3 across the board at all external board interfaces. Since all voltage regulation circuitry is contained on the EVK1100 board, we will only have to connect the supply voltage for our camera and sound module up to the VCC3 supply voltage on the board.

The sound module operates with a 3.3V supply voltage. Under typical operating condition, the digital components will draw no more than 15mA. At full volume during music playing, the headphone jack will draw no more than 60mA.

The camera module operates with a 5V supply voltage, drawing less than 120mA during active power. Initial tests have shown that the camera module works adequately off the nominal voltage (at 3.3V). Our preference is to continue using this device with a 3.3 supply voltage. However, we may have to find a way to use the regulated 5V supply voltage from the board. The LED included in the camera assembly will draw roughly 20mA.

The max current rating for the 3.3V supply on the EVK1100 board is 0.8A (for all on board and external peripherals). Most board peripherals draw current in the order of µA. Therefore, the board will be more than capable of powering all of our electronic devices.

4. Image Acquisition Module

The module chosen for Image Acquisition of The Maestro™ project is C3188A. This module interfaces using a digital port as its output. It is configurable using internal registers which are accessed using the Serial Camera Control Bus interface.

The camera on the C3188A module is the CMOS image sensor OV7620 [1] from Omnivision [2]. OV7620 is a highly integrated high resolution (640x480) Interlaced / Progressive Scan CMOS digital color / black&white video camera chip. The digital video port supports 60Hz YCrCb 4:2:2 16Bit / 8 Bit format, ZV Port output format, RGB raw data 16Bit/8Bit output format and CCIR601/CCIR656 format. This project will use the 16-Bit ZV digital port output format as its interface. Output data can be expressed in different formats, and with different type of channels (RGB, UVY). In this project the UVY format will be used, of which 8 bits represent the intensity (Y channel) of one pixel, the other 8 bits represent the U and V channels. Only the monochrome Y component is acquired since a black and white picture is sufficient for the OCR software. The information is sent continuously and is synchronized by the HREF, VSYNC and PCLK clocks as seen in Figure 4.

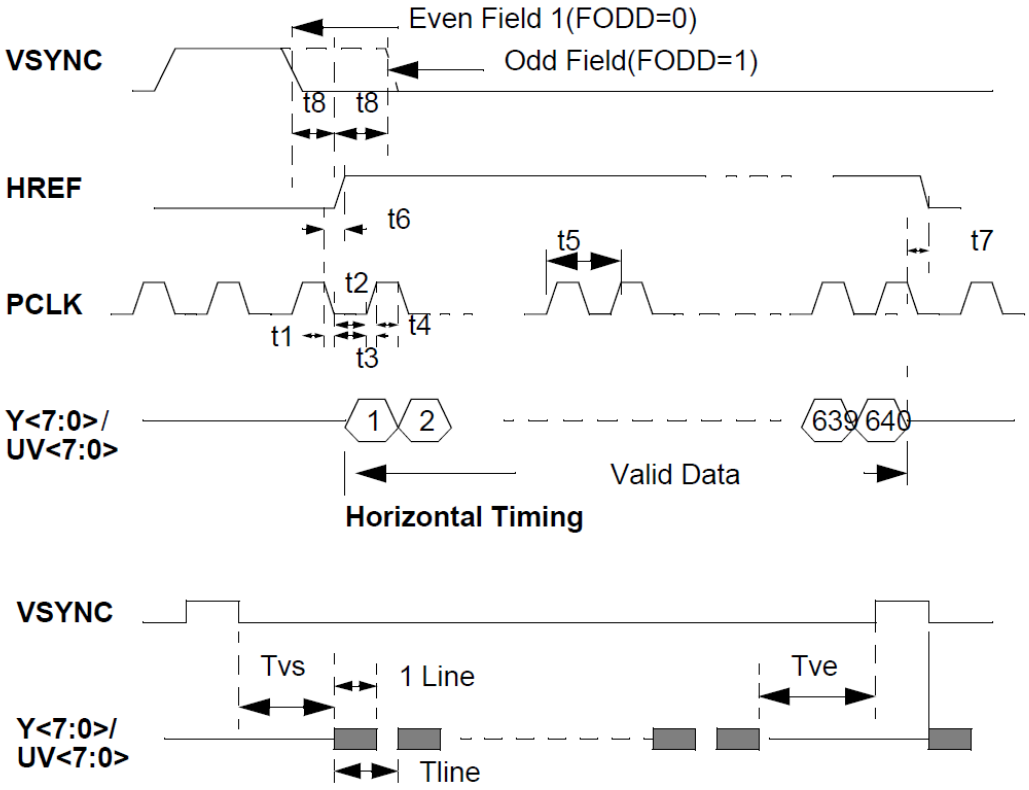


Figure 4 - ZV Port Timing

The falling edge of the VSYNC signal indicates a new frame, while the HREF signal indicates the validity of the information from the horizontal synchronization point of view. The PCLK is the data acquiring clock signal and it is valid only when the HREF is set. The default frequency of the PCLK is 13.5MHz which is too fast for our processor (30frames per second). PCLK frequency can be changed by modifying the default value of register address 0x11. This change allows us to read images from the camera directly with the microcontroller without the use of additional hardware.

Configuration registers can be accessed by the SCCB interface which is identical to the I2C bus. There are 80 registers in total which can be configured to fine-tune the image quality and the camera digital interface. Properties such as: PCLK period, image resolution, mirror image, windowing and color gains are just some of the configuration registers. Although the data-sheet of the C3188A camera says that the interfaced used to configure the registers is I2C, in the communication protocol of the camera OV7620 it is SCCB [4]. The only difference

is the ACK bit which in the SCCB protocol is treated as a don't care condition. Below is the I2C or the TWI interface protocol timing diagram (Figure 5).

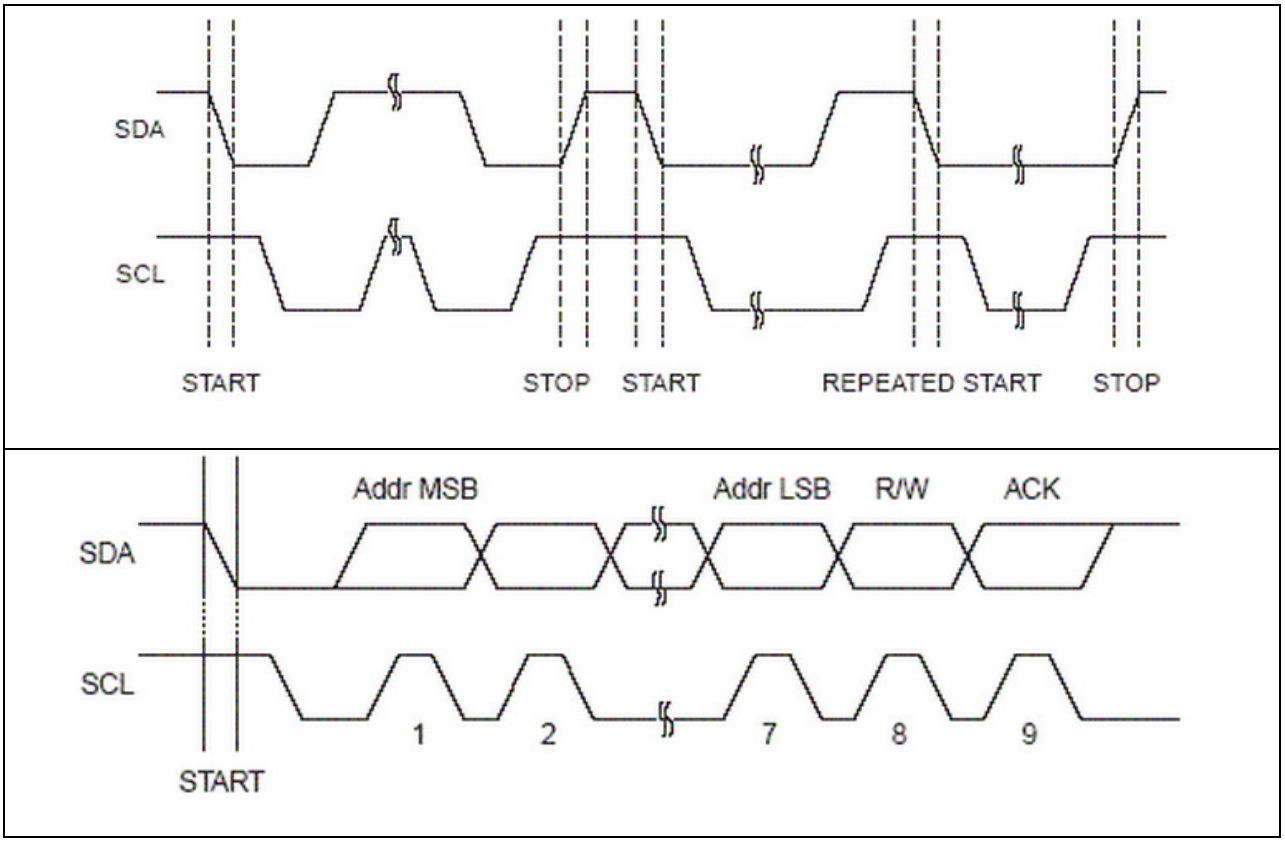


Figure 5 - TWI Interface Timing

The I2C bus is a communication protocol developed by Philips. In this protocol two pins are used, one is the clock and the other is the data. This protocol has a Master-Slave architecture. In our case the master is the AVR and the slave the C3188A camera. Registers of the camera can be read or written by the AVR. In the writing operation the master puts in the bus the writing address of the device, and after that puts the address of the register it wants to write to, and finally the byte it wants to write to in the register. The reading operation is similar: first the master puts the writing address of the device it wants to write to in the bus, after that the register address it wants to read from, and then the device

reading address. Finally the slave puts in the bus the data requested. The I2C communication was the most difficult part of the project, because of two main reasons. As it was said in section 2.1 [2], the C3188A camera implements the SCCB protocol that it is almost the same as I2C. To solve the protocol mismatch issue the TWI (Two Wire Interface) present in the AVR32 processor was used, since it is a synchronous bus as the I2C. Modifying AVR32 API's written in the assemble language was managed to interface the micro-controller with the camera module's SCCB . The result showed that the writing worked, but not the reading. After some investigations with the oscilloscope the problem was detected and solved - it was a timing related.

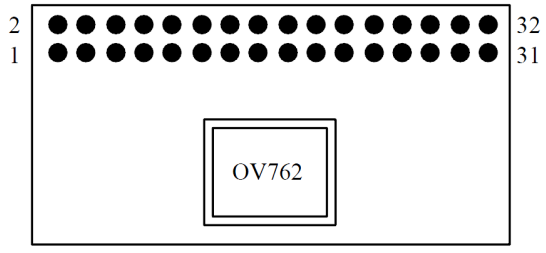
Camera module pins are interfaced with the EVK1100 board as shown in the table below (Table 4.1). The camera operation voltage is 5V, pins 20 and 22 are VCC and are connected to the 5V output on the EVK1100 board. Pin 31 is GND and pins 21 and 15 are AGND (analog ground) and all of them are connected to the common ground. Pins PWDN and RST are connected to ground, so all the resets of the camera are performed by software. The bus Y0-Y7 of the camera is connected to the ports PA0-PA7. PCLK is connected to PA16, HREF to PA17 and VSYNC to PA18. The bus UV is not connected because it is not used. SDA and SCL (I2C bus) from the camera are connected to SDA and SCL on the micro-controller, because these pins are connected to the TWI hardware that is used to implement the I2C protocol. VTO pin that is the Video Analog Output and it is not used in this project.

Table 1 - C188A pin connection interface with EVK1100

EVK1100	C3188A	Description
Y0-Y7	PA03-PA10	Y component of the YUV color model
Pin 20 & 22	VCC 5V	VCC
Pin 33 GND	GND	GND
Pin 21& 15 AGND	GND	Analog ground, GND
PWDN	GND	Power down pin

RST	GND	Reset pin
PCLK	PA16	Data valid CLK
HREF	PA17	Horizontal sync CLK
VSYNC	PA18	Vertical sync CLK
SDA	SDA	Serial data TWI signal
SCL	SCL	Serial CLK TWI signal
VTO	OPEN	Video output (not used)
FODD	OPEN	Odd Field flag (Not used)
EXCLK	OPEN	External Clock input (not used)

Camera module pins layout is outlined in Figure 6.



Pin Description

1~8	Y0~Y7	Digital output Y Bus.
9	PWDN	Power down mode
10	RST	Reset
11	SDA	I^2C Serial data
12	FODD	Odd Field flag
13	SCL	I^2C Serial clock input
14	HREF	Horizontal window reference output
15	AGND	Analog Ground
16	VSYN	Vertical Sync output
17	AGND	Analog Ground
18	PCLK	Pixel clock output
19	EXCLK	External clock input (need to remove crystal)
20	VCC	Power Supply 5VDC
21	AGND	Analog Ground
22	VCC	Power Supply 5VDC
23~30	UV0-UV7	Digital output UV bus.
31	GND	Common ground
32	VTO	Video Analog Output (75Ω monochrome)

Figure 6 - C188A Pin Layout

The image acquisition module will be enclosed in a box and internally illuminated for better image capturing conditions. This box will be a simple wooden rectangular enclosure whose main purpose is to keep the camera normal to the sheet music paper eliminating tilt issues and containing internal LED emitting light.

5. Sound Module

The part selected for sound generation is the VS1053b IC mounted on a breakout board from sparkfun electronics (BOB-08954). This chip contains a MIDI decoder, allowing us to load a format 0 MIDI file into the device for playback. The chip also provides a stereo earphone driver capable of driving a 30Ω load. This section is provides an overview of the sound module operation in our system.

5.1 Interfacing

The device employs two separate interfaces, the serial data interface and the serial control interface. The serial data interface allows a music file to be loaded into the device. The serial control interface allows for register configurations. It is recommended that we operate the device in VS1002 native mode (default mode of operation), so both of these interfaces will an SPI interface with shared clock, data in and data out lines. However, the data and control interface will have their own chip select line.

Implementing these SPI interfaces on the EVK1100 development board will by using the existing SPI drivers in AVR32 Studio.

The minimum period of SCLK is 4 internal clock cycles (CLKI). The default internal clock frequency is 12.288MHz, so the maximum frequency we can clock SCLK is 3.07MHz.

Both interfaces make use of the DREQ line. This line is used to indicate when data can be written into the device. If DREQ is high, the data interface can receive at least 32 bytes of data or the control interface can receive a new read or write command.

The breakout board provides a standard TRS, audio output (LEFT, RIGHT, GBUF signals). These will be connected to a 3.5mm, 1/8” headphone jack.

A summary of the port names and their connection to the development board or headphone jack is listed in Table 2.

Table 2 - Sound Module Port Assignments

VS1053b Port Name	BOB-08954 Port Name	EVK1100/Jack Port Name	Description
SO	SO	MISO (SPI)	Serial data out
SI	SI	MISI (SPI)	Serial data in
SCLK	SCLK	SCK (SPI)	Serial clock
xCS	CS	SS (SPI)	Control interface chip select (active low)
xDCS	BSYNC	GPIO (TBD)	Data interface chip select (active low)
DREQ	DREQ	GPIO (TBD)	Data request
xRESET	RESET	GPIO (TBD)	Reset (active low)
LEFT	LEFT	LEFT	Left channel, connect to left on headphone jack
RIGHT	RIGHT	LEFT	Right channel, connect to right on headphone jack
GBUF	GBUF	LEFT	Ground buffer, connect to ground on headphone jack

5.1.1 Serial Data Interface

The serial data interface connects to a 2048 byte FIFO that can be loaded with a song file. Loading the song is a matter of sending each byte of the song consecutively over the serial interface (MSB to LSB). Transfers over the serial data interface require setting the xDCS line low.

In order to maintain byte synchronization it is recommended that data is sent in chunks with the xDCS line going high in-between data chunks. For our device we will send data in 32 byte chunks.

5.1.2 Serial Control Interface

The serial control interface allows for configuration of the vs1053b registers. Writing to or reading from a register occurs using the 32 clock cycle transfer shown in Figure and Figure respectively.

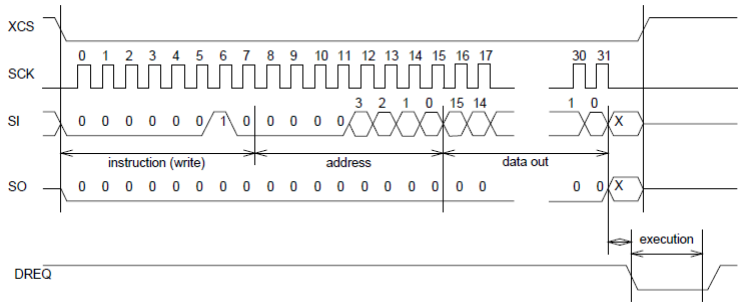


Figure 7 - SCI Write Operation

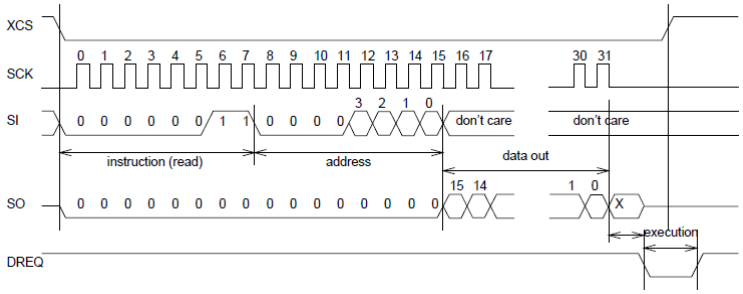


Figure 8 - SCI Read Operation

All SCI registers and their reset value are outlined in Table 3. Our device will be using all default configurations of vs1053b. Our user interface will allow access to change the Volume Control register (reg 0xB). A special sequence of reading to and writing to these registers is also required when playing a song (see section 5.2 Playing A File).

Table 3 - SCI Registers

Reg	Type	Reset	Time[1]	Abbrev[bits]	Description
0x0	rw	0x4800	80 CLKI[4]	MODE	Mode control
0x1	rw	0x000C[3]	80 CLKI	STATUS	Status of VS1053b
0x2	rw	0	80 CLKI	BASS	Built-in bass/treble control
0x3	rw	0	1200 XTALI[5]	CLOCKF	Clock freq + multiplier
0x4	rw	0	100 CLKI	DECODE_TIME	Decode time in seconds
0x5	rw	0	450 CLKI[2]	AUDATA	Misc. audio data
0x6	rw	0	100 CLKI	WRAM	RAM write/read
0x7	rw	0	100 CLKI	WRAMADDR	Base address for RAM write/read
0x8	r	0	80 CLKI	HDAT0	Stream header data 0
0x9	r	0	80 CLKI	HDAT1	Stream header data 1
0xA	rw	0	210 CLKI[2]	AIADDR	Start address of application
0xB	rw	0	80 CLKI	VOL	Volume control
0xC	rw	0	80 CLKI[2]	AICTRL0	Application control register 0
0xD	rw	0	80 CLKI[2]	AICTRL1	Application control register 1
0xE	rw	0	80 CLKI[2]	AICTRL2	Application control register 2
0xF	rw	0	80 CLKI[2]	AICTRL3	Application control register 3

5.2 Playing A File

When the Maestro in switched to Playback Mode, a complete format 0 MIDI file is available in SDRAM of the current buffered music. To play this file with vs1053b the following steps need to be performed:

1) Send the music file over the SDI interface
2) Read endFillByte from address 0x1e06 (SCI interface)
3) Send at least 2052 bytes of endFillByte over SDI interface
4) Set SCI_MODE (reg 0x0) bit SM_CANCEL
5) Send at least 32 bytes of enFillByte over SDI interface
6) Read SCI_MODE. If SM_CANCEL is still set, repeat step 5. Software reset if it doesn't clear
7) Song has been sent. Return to 1 to play another song

The vs1053b will decode the MIDI file and sound will be available through the headphone jack.

6. Software Design

In addition to creating the software drivers for the camera and sound module, will we also have to write software to interpret the sheet music and also to create a MIDI file from this interpreted music. This section provides an overview to our music notation recognition software, and how we will create standard MIDI files.

6.1 Music Recognition Software

Interpreting the scanned sheet music is instrumental in our system design. There are many existing image recognition techniques that were initially investigated. For example, using an algorithm to detect the location of ellipses could be used to pin point the location of all notes. However, running existing ellipse detection algorithms took about 30 seconds for a single 160x80 pixel image. Furthermore, finding the location of the notes doesn't give us the timing information associated with them. Given our limited computing resources, we needed to find a recognition technique that reduces the order of our operations.

Our proposed solution is to make use of a "black histogram" for the captured images. For every image that is acquired by the camera a histogram will be created. This histogram will count the number of black pixels in each vertical line of the image as shown in Figure 7. The features of the image are clearly visible in the histogram, and all recognition techniques on this histogram will be reduced to one dimension. However, this technique does rely on correct camera orientation. We plan to use a guide for the camera to ensure correct orientation, but may add software to correct for rotation error so that a guide isn't necessary.

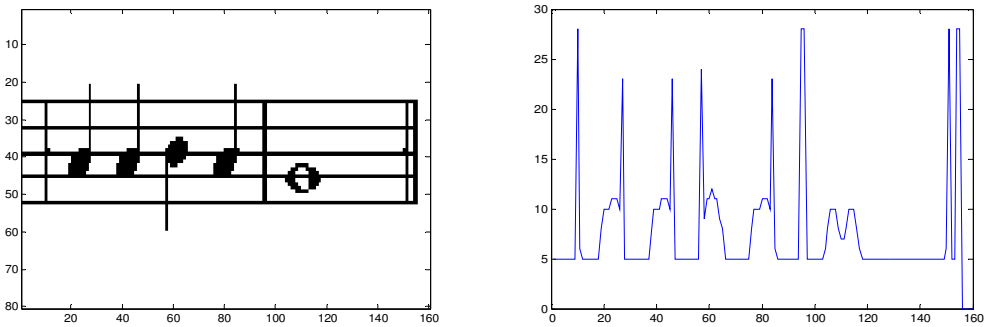


Figure 7 - Sample Image and Associate "black histogram"

Since the camera will be acquiring images at a constant rate, there will be an indeterminate amount of overlap between images. Therefore, the software will need to determine the point of overlap between adjacent images. This is possible by overlapping the histograms at an initial point and summing the error between histograms. The histogram can then be shifted and the error can be recomputed for the different point of overlap. If this error is normalize (by dividing it by the amount of overlap), the point of least error is the point of overlap. An example of the algorithm is shown in Figure 8. This shows that there are about 115 pixels of overlap between the first and second image, which is indeed the case on visual inspection.

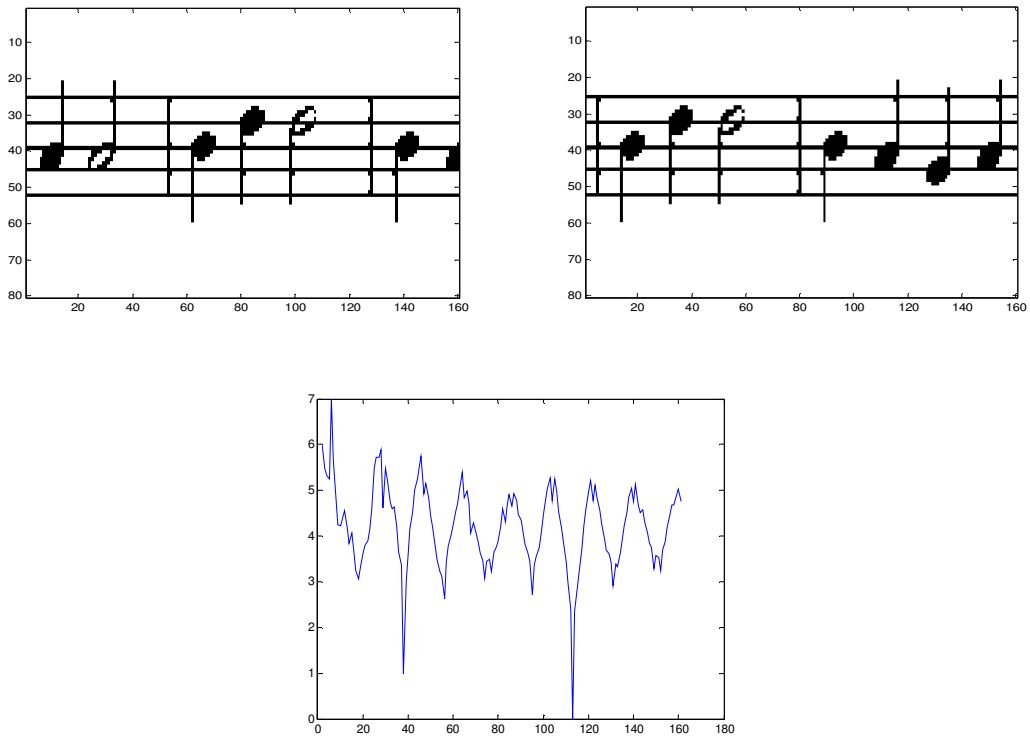


Figure 8 - Overlap Algorithm Example

Now that the point of overlap had been determined, the music features need to be isolated and recognized. Although it's possible that some features are cut off on the boundary of images, the entire feature will be contained in the adjacent image. The isolation of features will need to take this into consideration, ensure not to only count features in the overlap region once.

Once the features are isolated, the feature needs to be determined. We have opted to use a method of histogram matching to recognize features. Since each feature has a characteristic histogram, it can be compared to a library of reference histograms to see which one matches best. The reference histograms will be created with higher resolution images. A constant scale factor will have to be applied to "scale up" the scanned image histogram to the reference histogram. The comparison that has the least error is then determined to be the feature.

This histogram matching algorithm has been prototyped in Matlab and is showing excellent results. However, some corner cases will have to be considered, such as accidental and dotted notes. The reference library currently has 35 features in it. In order to avoid comparing the image histogram to every reference histogram, we will want to find way of reducing the search set (such as the width of the histogram, or the number of peaks in the histogram).

Once the feature has been determined, the pitch will have to be determined (if the feature is a note). This is a simple matter of detecting the location of the note elipses in the histogram. Then a vertical line at this location on the image will determine where the note sits on the staff bars.

6.2 MIDI File Generation

MIDI is the ideal choice for representing sheet music in a format that a sound card can understand and playback, however there are different types of MIDI file and different modes and the appropriate design choices need to be made.

The first choice was whether our device will implement a MIDI interface, i.e. streaming the MIDI commands to the sound card in real time, or store the data into a Standard MIDI File (SMF). The SMF file was chosen since it best fit the standard use case for the Maestro™ in which a user would scan a piece of music once and play it back multiple times while learning it. Storing the MIDI information as a file allows for the music to be played back multiple times as the student is learning it without needing to scan every time as well, it allows for the tempo to be varied as the user needs. The user may want to listen to the music at normal speed the first time then listen to it at a slower tempo as they are learning it and playing along. When they get better at it, they may even want to

challenge themselves by playing along to the music at a faster tempo – this is an ideal way to build up your technique, especially when learning scales or sequences.

The SMF is also a much simpler implementation than sending the MIDI commands to the sound card in real time. As described in the previous section, image processing will take a lot of the processor's power, thus the image is scanned and processed and after that the corresponding notes are written to the SMF. In this way there is no need to handle complex RTL while performing image processing.

The SMF standard specifies three types of MIDI files:

- Format 0: contains a single track
- Format 1: contains multiple tracks that are intended to be played back simultaneously
- Format 2: contains multiple tracks that do not necessarily have to be played back simultaneously

Format 2 is rarely used and Format 1 is useful for songs that have multiple instruments playing simultaneously, which is not the case for this project. Format 0 was chosen because it is the simplest of the three (providing easy integration and robustness in the algorithm) and supports everything that is needed for the Maestro™ device.

MIDI files consist of two "chunks":

- the header chunk – contains information about the file itself such as the format the number of tracks and the resolution
- the track chunk – contains information about the track such as the length, in bytes, of the track, as well as the MIDI "events" that constitute the body of the file

Events:

Events are the body of the music file – they specify what sound should be played at any given time as well as what effects or instruments should be used and how the sound should be generated. The most basic events are the "note on" and "note off" events which specify when a note should sound on and off. There are many events that are

specific to particular equipment, manufacturers or for generating particular effects. Since the MIDI file that we write is meant for playback from our device, there is no need to include any of the manufacturer-specific events and for the prototype version the events supported will be restricted to the basic "note on" and "note off" events that comprise the bread and butter of the MIDI sequence.

Resolution:

The resolution refers to the amount of units or "ticks" that each music beat is composed of. For example, in the header, the value of 96 ticks per quarter note can be specified. Thus any event that goes on for 96 ticks would be treated as a quarter note.

Take this sequence for example:

Note on: A; note off after 96 ticks

Note on: G; note off after 192 ticks

If the resolution is set to the aforementioned 96 ticks per quarter note, this would produce a quarter note A followed by a half-note G.

Changing the resolution allows for easy change of the tempo that the music plays at. Looking again at the example sequence, if the resolution is set to 48 ticks per quarter note, it would produce a half note A followed by a full note G.

For this device, the resolution doesn't need to be very high as the music generated is meant to teach students and not meant to closely simulate the sounds of instruments for integrating into a performance. As such, the resolution used will be 120 ticks per quarter note – this allows plenty of flexibility including the ability to speed up or slow down the tempo up to 5 times the original speed.

File Generation:

The MIDI file is generated by writing out the header to memory then writing out MIDI events in the body as they are identified by the image processing algorithm. As previously discussed, the image processing algorithm will perform error checking for each bar of music scanned, so the MIDI will be generated and written to memory one bar at a time after each bar has been processed. As well, the function that writes the MIDI information

into the appropriate memory location updates the number of bytes in the track length. After all the bars have been processed, the length, in bytes, of the MIDI file is checked to make sure that the value is correct, as to not cause any errors for the sound card interface.

The proof-of-concept model will only support one MIDI file in memory at a time, so if when another set of images is scanned, the resulting MIDI file will overwrite the existing file.

7. User Interface

The prototype of the Maestro™ will use the LCD screen and the buttons on the board itself for the user interface. Obviously for the actual product this will need to be fine-tuned and other considerations will be needed for such things as ease-of-use and ergonomics. This section will focus on explaining the user interface that will be implemented for the proof-of-concept model.

The user interface will be coded in C and will run on the processor along with the other software. It will interface with the components described below as per the descriptions of those components given in previous sections.

7.1 Components Used

LCD screen

The screen will display the menu as well as the action currently being performed. The main menu for the proof-of-concept model will consist of 3 options as shown

```
Main Menu:
Play
Scan
Change tempo
Set time signature
Set key signature
```

Arrows on the screen will indicate in which direction you can scroll with the joystick. The current menu option will be highlighted.

Pushbuttons

There are 3 pushbuttons on the board as shown in Figure 1. Their functions, in order from left to right are: SELECT, MAIN MENU, SCAN

SELECT – chooses the current menu item

MAIN MENU – returns to the main menu

SCAN – starts/stops the scanning if in scan mode

Rotary dial

The rotary dial will be used to control the volume for the music playback in the standard way: clockwise increases the volume, counterclockwise decreases it.

Joystick

The joystick will be used to navigate the menu: the up and down directions will move the selection up or down respectively. The right and left buttons will, on certain menu options, be used to change the settings such as choosing the key signature, time signature or changing the tempo.

7.2 Using the Maestro™

Navigating the Menu and Changing Settings

1. Push the MAIN MENU button to go into the main menu if not already there
2. Use the joystick to navigate the menu up and down
3. Push SELECT to choose an option
4. If the "change tempo" option is selected, the user can then use the joystick left or right to decrease or increase the tempo respectively
5. If "set time signature" or "set key signature" are selected, use the joystick left or right to navigate to the desired signature and press SELECT to choose it

Scanning

1. Place the device over the section of music that should be scanned according to the guiding enclosure
2. Navigate to the "scan" option in the main menu and press SELECT
3. If the time and/or the key signature settings have not been set, the user will be prompted to do so
4. If there is already a saved MIDI file the user will be reminded that scanning new music will overwrite the previously saved file

5. Push SCAN to start scanning; the LCD screen should confirm that the device is scanning
6. Move the device slowly in a straight line over the bars that need to be scanned
7. If the edge of the paper was reached and a jump needs to be made, push the SCAN button to pause the scan, move the device to a new position and push the SCAN button again to resume the scan
8. Push the SELECT button to stop scanning and proceed to image processing
9. At this point the last images scanned will be processed and the MIDI file will be finalized.

   NOTE: if during the scan, the software has detected any error in the images scanned that prevents proper recognition of the music, an error message will be displayed and the user will need to re-start the scan

Playback

1. Navigate to the "play" option in the menu
2. Push SELECT to start the playback
3. Push SELECT again to stop the playback if desired
4. If a tempo change is desired, navigate to the tempo settings in the Main Menu, make the desired change and play the file again

7.3 Additional Considerations for the production model

Usability

Currently the proof-of-concept model only supports one saved file in memory at a time. For the actual device, this may need to be increased in order for the user to scan multiple parts of a music sequence and play them bask so that they may alternate practicing them without re-scanning.

The production model will also support fast-forwarding or rewinding the track since the user may want to scan a longer section of music, but focus their practice on a smaller sub-section.

Ergonomics

The enclosure of the production model will need to be designed with user comfort in mind, i.e. the buttons need to be easily accessible, the device needs to be comfortable to hold while scanning and the interface needs to be easy to use.

8. System Test Plan

To ensure that the device is operating in accordance to the functional specifications, the prototype will undergo comprehensive testing to make sure that the goals have been achieved and that the design can proceed with working towards the production model. To ensure thoroughness and to ease integration, each component and module will be designed and tested individually before integration into the system. Once integrated, these same tests will be performed to make sure that the module still works as expected. Plus, there will be additional tests to ensure that the device itself is operating according to specifications.

8.1 Unit Tests

This section outlines the tests that will be performed on each of the modules individually before they will be integrated into the system, as well as after integration.

Micro-controller

- Ensure that the processor can load and run code (done through the debugger interface – run a simple piece of code to ensure correct operation of the component)
- Verify that the memory interface works as expected (done by writing non-trivial values to locations in memory and reading them back to confirm that read/write works)

Sound Module

- Verify that the sound module can interface with the memory by playing a test MIDI file loaded into the expected memory address through the debugger

Camera and Image Processing Module

- Verify that the camera can be used to acquire images (make sure that an image can be read by the processor from the camera module)
- Ensure that the setting for the camera are appropriate for obtaining the image for the application (use the debugger to view the image that the processor obtained from the camera interface, extract it to a bitmap image and run the image processing algorithm on

Software

The software for the image recognition has been prototyped in Matlab. As such, the software testing needs to be done twice: once to test the validity of the algorithms used in Matlab, then again when the code is ported to C and runs on the processor. Test cases include:

- Verify that that algorithm works on manually-generated test images (split a line of music into "snapshots" and run these snapshots through the algorithm)
- Verify that the algorithm works with different partitioning of the music line into "snapshots" paying close attention to corner cases such as too much overlap or not enough overlap between the partitions and ensure that the algorithm performs as expected each time
- Simulate noise, lack of light, etc, in the test images to ensure that the algorithm meets expectations
- Verify that the algorithm works on actual images taken from the camera module
- Once the code has been ported and runs on the microprocessor ensure that proper operation is maintained (also need to ensure that interrupts, memory interface, etc. interact properly)

8.2 System Tests

Once the system is integrated, it is important to ensure that the device operates as expected in normal conditions as well as handles corner cases in a deterministic manner.

Table 4 - System Testcases

Case	Input, Conditions, and States	Expected Results
User Interface Navigation	User navigates through the user interface and changes settings	Settings should be changed according to the user inputs, all possible settings must be able to be selected

Normal scanning and playback (simple sheet music test)	User scans a line of simple sheet music then plays it back	Interface should work to allow the user to scan the music, once scanning is complete the confirmation that the music has been properly converted to MIDI should appear and the MIDI should play back as expected
Normal scanning and playback (comprehensive sheet music test)	User scans a line of sheet music containing all the different symbols that the device in its prototype stage can support, as described in the functional specification	Interface should work to allow the user to scan the music, once scanning is complete the confirmation that the music has been properly converted to MIDI should appear and the MIDI should play back as expected
Scanning and playback more than one line of music	User scans more than one line of sheet music	Interface should work to allow the user to scan the music, once scanning is complete the confirmation that the music has been properly converted to MIDI should appear and the MIDI should play back as expected
Corner-case scanning (sheet music is slightly distorted, but still within the limit of detection)	User scans sheet music that has some distortion in it such as dirt, slightly, crumpled paper or slightly tilted	Interface should work to allow the user to scan the music, once scanning is complete the confirmation that the music has been properly converted to MIDI should appear and the MIDI should play back as expected
Corner-case scanning (sheet music is distorted or contains symbols not within the capability of the prototype to identify)	User scans sheet music that has a great deal of distortion in it	Interface should work to allow the user to scan the music, but once an unidentifiable section is processes, the device should indicate that an error has occurred and the music cannot be identified

8.3 Tests for the Production Model - Standards

The Maestro™ will be designed under the strictest compliance with consumer electronics standards. Some of these standards will be mandated by federal law, especially in the most important markets, which will be North America upon introduction of the device.

Some voluntary standards will also be adhered to for environmental protection and other goodwill agreements that make the Maestro™ a responsible choice both for the environment, the user, and because there is value in the technology. These standards will apply only to the final production model and are not a primary concern for the proof of concept model.

8.3.1 Electronic Standards

The electronic standards that the Maestro™ will have to comply with are set out by the CSA and UL. These standards are mandatory and regulate both electronic safety to prevent unintended interference with other devices. They include

Electromagnetic fields (EMF) for consumer electronics is minimized in order to limit interference between proximate devices. These waves also interfere with communications hardware and are thus strictly regulated. The production model will be designed along the guidelines set forth in the CSA and UL standards for low-EMF operation firstly by using low current levels of operation (mA range of current consumption), and also incorporate reactive-balancing capacitors in its circuits)

8.3.2 Environmental Standards

Environmental standards are another important part of the Maestro™'s design. RoHS is a standard for environmentally friendly products that contain no mercury, lead or zinc so that, despite improper disposal, will not release harmful substances into the environment.

The plastic case for the Maestro™ will be made out of plastics that are compliant with ASTM standard D5511-02 that describes anaerobically biodegradable plastics. These plastics will degrade in landfill conditions.

8.3.3 Safety Standards

The electronic standards that the Maestro™ will have to comply with are set out by the CSA and UL. These standards are mandatory and regulate both electronic safety to

prevent unintended interference with other devices. CSA standards have not been fully researched at the time of the writing of this document as the standards have not been purchased, but standards regarding control electronics are being researched, such as CSA C22.2 NO 142-M1987. For testing purposes, CSA C22.2 NO 0.3-01 will be looked into.

Electromagnetic fields (EMF) for consumer electronics is minimized in order to limit interference between proximate devices. These waves also interfere with communications hardware and are thus strictly regulated. The production model will be designed along the guidelines set forth in the CSA and UL standards for low-EMF operation firstly by using low current levels of operation (mA range of current consumption), and also incorporate reactive-balancing capacitors in its circuits.

Standard CSA C22.2 NO 8-M1986 will be used as a starting point for research into EMF shielding. CSA C22.2 NO 1-04 describe standards related to audio equipment.

8.3.4 Usability Standards

Compliance with usability standards in North America are not mandated by any federal regulation in North America. Standards regarding usability and ergonomic design are subject to goodwill adherence to agreements and conventions by the manufacturer.

Production of the Maestro™ will be in full compliance with these standards in order to ensure ease of use and convenience for the user. Conventions such as soft corners and soft colors will be adhered to so as to ensure the device is easy to use for young users as well.

The colors on a device can affect the mood of a user, with sharp contrasts creating an image of modernity and sophistication for the user. An ergonomic enclosure which fits naturally in the user's hand, with buttons with the thumb able to control the device easily makes the device able to handle. These standards will be inspired by other devices on the market, and extensive research will be conducted into usability factors. For the proof of concept model, however, this will not be a primary concern.

Such standards are currently being scrutinized to determine which will reach the final production model.

9. Conclusion

Throughout this document the design specifications for the Maestro™ were discussed in detail so that the engineers may take them and use them to construct the prototype of the device. The document focuses mostly on the prototype model, as this will have to be completed and demonstrated to work before the production model can be designed. However, some specifications for the production model were discussed, in particular the ones that refer to standards that the product must meet and thus must be considered early on the design stages. The document explains the choices behind the chosen hardware and the implementation and how these choices are made in accordance to the functional spec as well as the company's goal to make the Maestro™ an invaluable device for music students.

References

[1] Omnivision Inc, "Advanced Information Preliminary OV7620" Quasar Electronics, July 10 2001, [online], http://www.quasarelectronics.com/kit-files/omnivision/ov7620_ov7120_v1.2whole.pdf [Accessed: November 5, 2010]

[2] Omnivision Inc, "c3188a 1/3" Camera Module with Digital Output" Quasar Electronics, July 10 2001, [online], http://www.quasarelectronics.com/kit-files/camera-module/d-c3188a.pdf [Accessed: November 5, 2010]

[3] Micron, "MT48LC32M8A2 Datasheet" Data Sheet Catalog, August 20 2006, [online], http://www.datasheetcatalog.org/datasheet/micron/MT48LC32M8A2.pdf [accessed: November 1 2010]

[4] VLSI Solutions, "VS1053b – Ogg Vorbis/MP3/AAV/WMA/FLAC/MIDI AUDIO CODEC CIRCUIT" VLSI Solutions, January 10 2007, [online]m http://www.vlsi.fi/fileadmin/datasheets/vlsi/vs1053.pdf [accessed: November 5, 2010]

[5] Joseph Rothstein, MIDI a comprehensive introduction 2nd edition, Madison, Wisconsin : A-R Editions, Inc, 1995

[6] CSA Standards, "CSA – Electrical/Electronics Collection" CSA Standards, September 2010, [online], http://canada.ihs.com/collections/csa/78b.jsp [accessed: November 7, 2010]