EEL4924C Senior Design
Final Design Report
Fall 2021

NOTE NINJA

Benjamin Wheeler & Hannah Kirkland
(The Note Ninjas)

Contents

Introduction

Note Ninja is a music game in the spirit of Guitar Hero and Rock Band but with real musical instruments. Currently, it can interface with a MIDI keyboard and an electric guitar. The goal of Note Ninja is to be challenging and entertaining while also serving as casual practice with the instrument. It consists of a microprocessor running the main game software, an FPGA driving the game graphics, and a digital signal processor handling the guitar note detection and scorekeeping.

Features and Objectives

- MIDI Keyboard input: Allows pianists to play the game with any standard MIDI-enabled electric keyboard.
- Guitar jack input: Allows guitarists to play the game with any standard electric guitar.
- VGA output: Allows the game to be displayed on any VGA-capable monitor screen or TV.
- Scorekeeping on the LCD allows the musician to understand at a glance how well they are performing the song, which serves as feedback on their skills as they improve over time.
- Full treble clef range: The game supports all notes within the treble clef, which allows musicians to practice wide ranges on their instruments.

Competitive Products

There are existing solutions that are meant for entertainment or training, but not for both. On the entertainment side, there are music and rhythm games such as Rock Band or Guitar Hero. In these games, the player uses a toy instrument with buttons to recreate the melody of a song. On the training side, there are programs such as Melodics that are used to teach students how to play the piano, finger drums, and standard electronic drums. One thing Melodics lacks is support for the electric guitar.

Concept Selection

A Digital Signal Processor was selected for guitar note detection because it can directly interface with the raw guitar signal and has a specialized architecture which enables fast note detection via an FFT algorithm. This allows real-time note detection, which is exactly what a time-sensitive game requires.

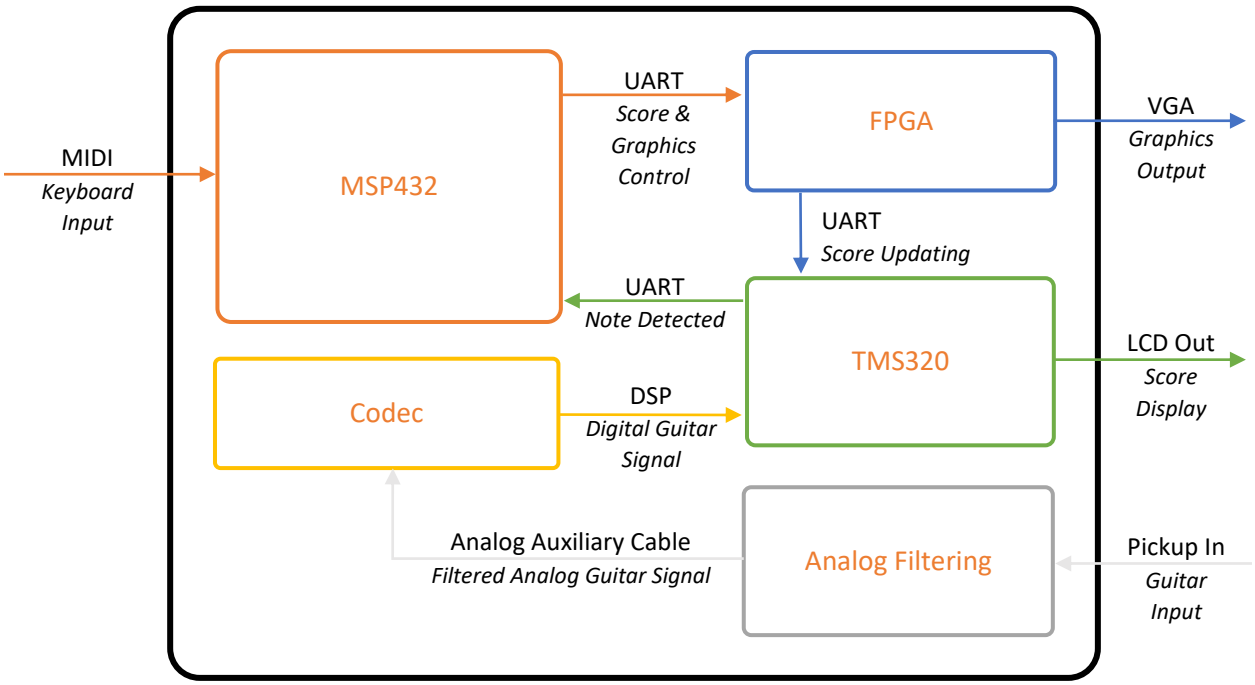
In terms of game logic, a microprocessor was chosen due to low cost and versatility with I/O and software. A real-time operating system was used on the microprocessor due to the necessity of high-performance MIDI decoding, game scorekeeping, and graphics control, which all utilized external serial modules. An RTOS gave a nicer top-level interface, better resource sharing, and more maintainable software by separating individual tasks into threads.

An FPGA was selected for graphics control due to its I/O (the selected development board has a VGA port on it), its relatively easy ability to send VGA signal to a screen, as well as its large I/O capabilities (for serial communications with other devices running the game).

These selections could possibly have all been replaced with a more general-purpose computer, such as a Raspberry Pi, but selecting embedded-oriented devices such as DSPs, MCUs, and FPGAs allows for major cost reduction in the final product.

Project Architecture

In addition to the chart below, a 5VDC barrel plug power supply was used for the MSP432 and USB power was used for the TMS320 and FPGA.



Hardware Selection

For the guitar note detection, a DSP was needed to run an FFT on the signal from the guitar pickup in real time. After this limitation, the TMS320 was a good choice as Hannah Kirkland was already familiar with the chip from the class Real Time Digital Signal Processing. In addition, we already had a development board for the TMS320 which would enable us to develop software faster and more efficiently.

Another DSP we considered was the PIC DSP. One other option we found later that could have been advantageous to use is the ATSAMD51, as this DSP also is supported by MicroPython.

Running the overall game requires a few tasks: Serial communication for midi keyboards, guitar note information, graphics control, and scorekeeping, as well as logic to control all of them. An MSP432 was selected for these tasks due to its sufficient I/O capability (4 UARTs) as well as its ability to run a real-time operating system which creates more efficient control over each task.

Finally, a DE10-Lite FPGA development board was selected for the FPGA due to its large I/O breakout for serial I/O, as well as a VGA header to connect to a screen for graphics.

Software Selection

As previously stated, many tasks must be completed by the microprocessor for the game to meet all of the feature requirements. It must be able to decode an incoming MIDI UART signal, read an incoming guitar note UART signal, output a high-throughput graphics control UART signal, and control an LCD via yet another UART signal. Each one of these tasks is complicated and requires a lot of CPU time. It would be difficult to program a microprocessor without any frameworks to allow each task to run in real-time,

with asynchronous events, without either unmaintainable/complicated code, or features that do not work to spec.

An RTOS, on the other hand, allows the microcontroller to run tasks in a round-robin fashion, ensuring each one gets adequate CPU time to perform their duties. Secondly, inter-process communication structures such as FIFOs make it trivial to (for example) collect incoming UART data from a MIDI keyboard, which can be read whenever the MIDI decode process needs the next byte of information. Finally, the thread concept allows for more isolated code, which improves maintainability. That combined with the asynchronous nature of each of the tasks makes an RTOS the perfect tool for the project.

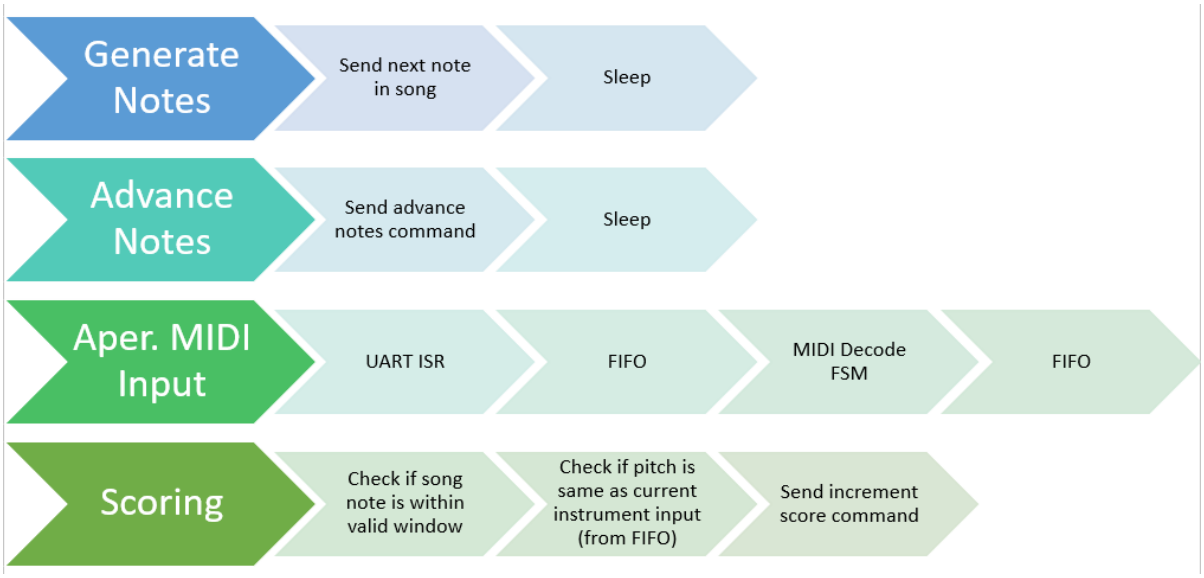
Design Procedure

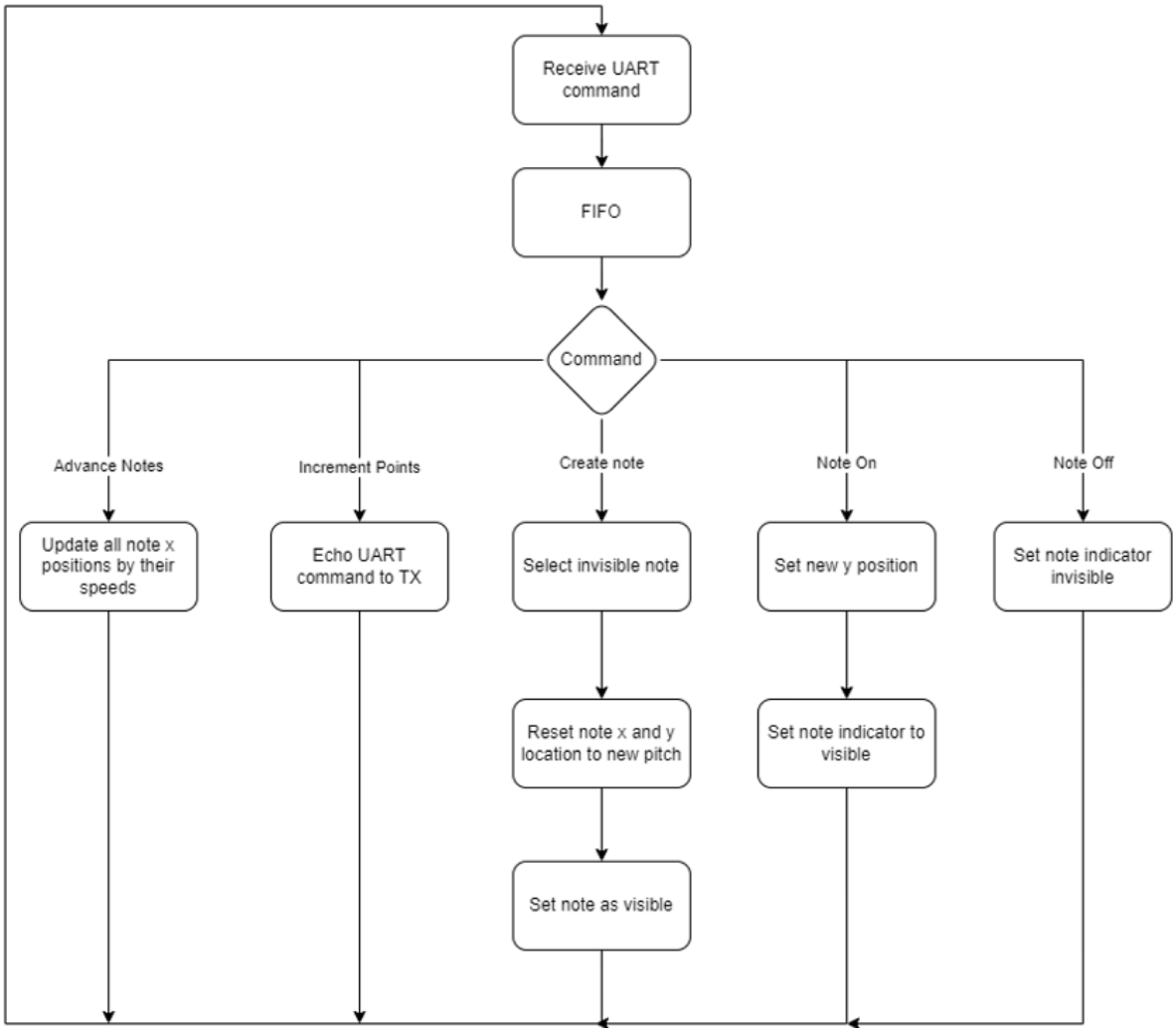
Since each software task is isolated, they can be completed in any order. The MIDI decode task involves understanding the MIDI specification and creating a finite state machine that can decode the signals into useful bytes. The Graphics control threads simply send UART commands to the FPGA at controlled time intervals. Finally, the logic for scoring involves ensuring a note that is eligible to award points to the musicians is being played by the musicians in a valid window of time.

The FPGA procedure is also relatively isolated, since its primary task is to display several squares to the screen. UART commands are received by a UART entity, and a controller decodes and executes the command. Each square can be designed with registers preserving their locations, which the controller may update as new commands come in. Finally, the logic to send proper display signals can be written by understanding and implementing a standard VGA interface in RTL.

For the guitar note detection, an amplifier was designed to filter out low and high frequency noise and amplify the signal to a voltage level the codec could detect. The codec was set up to communicate via DSP with the TMS320. For the software on the TMS320, a 1024 size FFT was set up. This, along with an 8kHz sampling rate on the codec, gave us a 7.8Hz resolution and a max frequency of 4kHz. This resolution and range was more than enough for note detection on an electric guitar. A buffer is filled up with the data points from the codec, the FFT is run, and then a series of if statements check to see if the signal has sufficient amplitude to indicate a note is played. Whenever a note is played, it is broadcast to the microcontroller via UART. Whenever the DSP receives a score update message via UART, the score on the LCD is increased. The LCD is controlled via UART.

Flowcharts & Diagrams





Design Changes

A major design change was made to the guitar note detection part of Note Ninja. The original TMS320 stopped functioning, and due to the chip shortages and design requirements, a new PCB was designed to hold the TMS320 launchpad, a codec breakout board, and analog filtering and amplification circuitry.

A minor design change was made to save time. In our original design, Benjamin Wheeler's board would output a basic synthesized sound corresponding to the input from the MIDI keyboard. His original board broke, either due to ESD or a manufacturing defect, so when resoldering another board, the components for this functionality were let out (DAC, audio amp, piezo buzzer). This allowed him to put more time into software development, which was more important for the objectives of his part of the project since he is studying computer engineering.

Bill of Materials (BOM)

Comment	Description	Footprint	Quantity	Price
Cap	Capacitor	6-0805_M	28	$2.80
BYP Cap	Capacitor	6-0805_M	2	$0.20
P1 LED	Typical INFRARED GaAs LED	LED-0	1	$0.75
Diode	Default Diode	DIODE-0.4	2	$1.00
Hole 3mm		Hole 3mm	4	$0.00
SDS-50J		CUI_SDS-50J	2	$3.82
PWR2.5	Low Voltage Power Supply Connector	Power 2.5 fixed	1	
3.5mm jack GM		3.5mm jack GM	2	$1.60
LQM2HPN4R7MG0L	Inductor with Inductance: 4.7uH Tol. +/-20%, Package: 1008 _2520_	INDC2520X100N	1	$0.09
Speaker	Loudspeaker	PIN2	1	$1.99
Res1	Resistor	AXIAL-0.3	22	$2.20
RPot	Potentiometer	VR5	1	$0.46
RST-BTN	DIGIKEY SW400-ND	SW PB	1	$0.18
LM3940IT-3.3	1-A, low-dropout voltage regulator for 5-V to 3.3-V conversion 3-TO-220 -40 to 125	TO254P1054X470X1955-3	1	$2.00
LT1661		DIP-8	1	$5.08
MSP432P401RIPZ	SimpleLink ultra-low-power 32-bit Arm Cortex-M4F MCU With	QFP50P1600X1600X160-100N	1	$20.21

	Precision ADC, 256KB Flash and 64KB RAM 100-LQFP -40 to 85			
LM386N-1	Low-Voltage Audio Power Amplifier	DIP-8	1	$1.50
Optoisolator1	4-Pin Phototransistor Optocoupler	DIP-4	2	$1.14
Jumper	Jumper Wire	RAD-0.2	1	$0.10
NE5532P	Op-amp	DIP-8	2	$1.52
Headers-40POS	Headers	0.100” Pins	4	$3.64
LAUNCHXL-F28379D	C2000 Launchpad	Evaluation Board	1	$40.55
TLV320AIC23B	Codec	28-TSSOP	1	$8.49

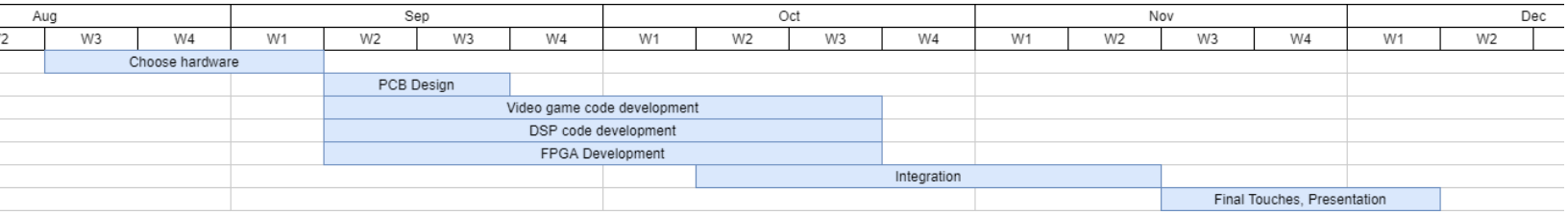
Work Division and Collaboration Assessment

The work was mainly divided into two sections: guitar note detection and game control logic. Hannah Kirkland was responsible for detecting what note was being played on the electric guitar and making that information available to the MSP432. In addition, she was responsible for designing and redesigning her own PCB board and ensuring all components and software contained on it were functional.

Benjamin Wheeler was responsible for the main game logic on the MSP432 as well as the graphics control on the FPGA. He was also responsible for designing his own PCB board and ensuring all the software on the MSP432 and FPGA was functional.

As Benjamin Wheeler is studying computer engineering, he provided software help and support; and as Hannah Kirkland is studying electrical engineering, she provided hardware help and support. In this way, we were able to achieve a project with good digital design as well as good signal processing and analog design.

Gantt Chart/Project Schedule



Software Repository Links

Software, RTL: https://github.com/benjamin051000/Note-Ninja