Project 3 Individual Report

LanChau LeTran

**Project Description:**

For this project, I wrote the functionality that plays Tetris (mode 2) on the Simon Board, documented our work, and wrote the team report. All the calculations that we did as a group for the music notes are in the group report.

For Tetris, I got each individual notes required and coded the function to play the right node.

I documented our code and went through each functionality to write up the report.

These are the following functions that I contributed for the group portion:

void playSong2() {

// Measure 1.

displayNote(4);

playNote(notes5[4], 2, 1);

displayNote(11);

playNote(notes4[11], 1, 1);

displayNote(0);

playNote(notes5[0], 1, 1);

displayNote(2);

playNote(notes5[2], 2, 1);

displayNote(0);

playNote(notes5[0], 1, 1);

displayNote(11);

playNote(notes4[11], 1, 1);

// Measure 2.

displayNote(9);

playNote(notes4[9], 2, 1);

displayNote(9);

playNote(notes4[9], 1, 1);

displayNote(0);

playNote(notes5[0], 1, 1);

displayNote(4);

playNote(notes5[4], 2, 1);

displayNote(2);

playNote(notes5[2], 1, 1);

displayNote(0);

playNote(notes5[0], 1, 1);

// Measure 3.

displayNote(11);

playNote(notes4[11], 2, 1);

playNote(0, 1, 1);

displayNote(0);

playNote(notes5[0], 1, 1);

displayNote(2);

playNote(notes5[2], 2, 1);

displayNote(4);

playNote(notes5[4], 2, 1);

// Measure 4.

displayNote(0);

playNote(notes5[0], 2, 1);

displayNote(9);

playNote(notes4[9], 2, 1);

displayNote(9);

playNote(notes4[9], 3, 1);

// Only Delay the Next Play of the song if We are not switching modes.

if(!flag)

timerover(SEC);

}

/\*

\* Discription: Does one Cycle of playing a note at a Frequency. Repeat to create a audiable sound.

\* Requires: A Preload Value for a note that is less than 16 bit and is already set to 1/2 Duty.

\* Returns: None.

\*/

void holdNote(unsigned int note) {

// Make sure speaker is off.

SPEAKER = 0;

// Load the note value up.

TH1 = -note >> 8;

TL1 = -note;

TR1 = 1;

// Turn On SPEAKER.

SPEAKER = 1;

while(TF1 == 0);

// Turn Off SPEAKER.

SPEAKER = 0;

TR1 = 0;

TF1 = 0;

}

/\*

\* Discription: Uses Timer 0 AND Timer 1 to play a note.

\*\*\* assumes that both timers are not in use.

\*\*\* note is preload for square wave for that frequency of that note.

\*\*\* type is the type of note: 1 = eighth note, 2 = quarter note, 3 = half note, 4 = whole note.

\* Requires: A Note Preload Value that is less than 16 bit and is already set to 1/2 Duty.

\*\*\* The type of note to be playing and the tempo to be set at.

\* Returns: None.

\*/

void playNote(unsigned int note, unsigned char type, bit tempo) {

// Looping over causes imprecise timing due to loop nature.

unsigned char loop = 225;

// Set the tempo.

unsigned int timerLoad = (tempo == 0 ? beats[type] : beats[type + 4]);

// Run the pause.

for(; loop > 0; loop--) {

TH0 = -timerLoad >> 8;

TL0 = -timerLoad;

TR0 = 1;

// Run while timer goes, speaker may slightly change.

while(TF0 == 0) {

// Check to see if we are playing a rest ( note = 0) or if the mode change flag has been set.

if(note > 0 && !flag)

holdNote(note);

// If the mode change flag has been set, leave the playNote Function.

if(flag)

return;

}

TR0 = 0;

TF0 = 0;

}

// Articulation Pause.

timerover(beats[0]);

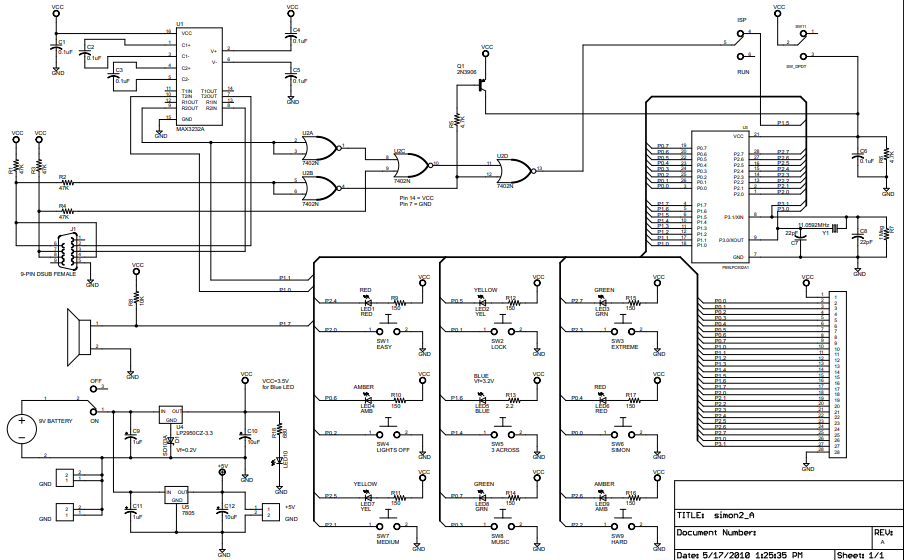
}

**Architecture (in-depth in the team report):**

I used timer 0 and timer 1 in order to play each of the notes. HoldNote sends a single square wave using timer 1, and PlayNote loops HoldNote for as long as the node should last using timer 0. The preload values for each of these timers are calculated in the team report.

**Simon Board:**

P2.2: Used to switch between modes.



**Individual functionality (description and architecture):**

While in mode 1 or mode 2, the program will serially transmit the note and display the current musical note in terminal.

/\*

\* Discription: Display Function to display a note being played via UART to Terminal.

\* Requires: index of the note being played. valid if between 0 - 11.

\* Returns: None.

\*/

void displayNote(unsigned char note) {

display(notes[note], 2);

turnOnLED(note); // This is alan’s function

}

/\*

\* Discription: Sends a string via Serial Communication via UART to be displayed on terminal.

\* Requires: Array of characters to be sent and size of that array.

\* Returns: None.

\*/

void display(unsigned char text[], unsigned char size) {

unsigned char k = 0;

// Loop through each character in the array.

if(flag) return;

for(;k < size; k++) {

// Transmit the character.

uart\_transmit(text[k]);

}

}

In each of the song’s function where they are played, I call displayNote passing corresponding note that is being played. DisplayNote then calls the display function that serially transmit the corresponding letter to be display.

No pins are used in this individual part.