Lab 2: General MIDI Explorer (GME) with Record/Playback

Associated lectures:
Lab assignment and demonstration, microcontroller basics

Lab objectives:
Gain experience working in lab (logic analyzer, oscilloscope, multimeter, power supply, breadboard circuits)
Gain experience with microcontroller basics (configuration, timers, interrupts, EEPROM).
Gain experience with microcontroller I/O (digital I/O, analog inputs, serial communication).

1. Introduction

In this lab you will build and program a MIDI (Musical Instrument Digital Interface) controller, the GME (General MIDI Explorer). A MIDI controller is a device which sends MIDI messages to a synthesizer. In this lab, the synthesizer is a program running on the PC.

The MIDI standard was established by a consortium of manufacturers of electronic musical instruments over 20 years ago. The companies Sequential Circuits, Roland Corporation and Oberheim Electronics were major players in establishing MIDI as an international standard. General MIDI (GM) is an extension of the original MIDI specification (MIDI 1.0). Whereas 1983’s MIDI 1.0 did not tie certain musical instrument sounds to certain MIDI codes, GM did just that. Established in 1991, GM assigned 128 musical instrument sounds (timbres) their own 7-bit codes. SoundBlaster-compatible soundcards the world over support GM.

The GME that you will create will allow the user to "explore" the sounds available in General MIDI. Specifically, the GME will record General MIDI notes coming into the device and will modify and replay the notes on command.

2. Project Description

The user will send MIDI notes into the GME from the computer. When in recording mode, the GME will store to EEPROM the sequence of notes that are received and the timing of the notes. The stored notes remain in memory until a new recording is made. When the user switches to playback, the notes will be played back from memory with approximately the same inter-note timing as in recording. If the modify switch is on when in playback mode, the notes that are played back will be modified as described below. If the playback and record switches are both off, the GME is idle.

In modify mode, students will manipulate the playback of notes by changing the amount of light falling on two optical sensors. The on-chip ADC converts an analog voltage from a photo sensor circuit into a digital value that modifies the MIDI message that will be played back. The first analog input should control the speed of playback. Groups are free to decide how to use the second analog input for controlling the playback in other ways (for example, it could modify the instrument/note/velocity or the number of times each note is repeated during playback). Other modifications are fine too, so explore this and have fun with it. Credit for the second analog input will given as long as the students can demonstrate and explain how it is being used.

3. Design Specification

The system will be built around the ATmega32 AVR microcontroller from Atmel. You will program the device in C using the AVR Studio Integrated Development Environment (IDE). You will use the following AVR on-chip subsystems: Timer1 (a 16-bit timer) with interrupts, EEPROM, Parallel I/O Ports, 10-bit Analog to Digital Converter (ADC), and the USART (Universal Synchronous/Asynchronous Receiver/Transmitter) Serial Port. The most important document you will need to
refer to is the 300+ page ATmega32 datasheet. The GME can be based on a loop program structure with timing based on counting of 16-bit Timer1 overflows.

Input/Output signals on the ATmega32 AVR microcontroller:

*Play Mode – 1 Bit Input Switch* – Turns off and on Playback Mode. The play mode switch is required to play back any stored MIDI information in the order in which it was received. It is required that it plays both the note on and note off in quick succession (around 1ms, can vary by preference). The timing between the start of notes during playback must approximately match the timing of the notes during recording.

*Modify Mode – 1 Bit Input Switch* – Turns off and on the playback modifications that are controlled by the analog inputs when the system is in play mode. Switch has no effect when in record mode.

*Record Mode – 1 Bit Input Switch* – Turns off and on Record Mode. The recording mode switch is required to record MIDI notes being sent by the computer and store it in the AVR’s EEPROM data. The recording mode should keep count of how many notes have been stored and should stop recording when it reaches its maximum storage capabilities.

*Digital outputs* – LED bar shows the last note (i.e. the value of second MIDI byte) recorded or played. Therefore, the LED bar should be changing as the music is being recorded or played back.

*Analog input from optical sensors* – analog voltage from a simple voltage divider circuit as shown in slides. Each of the two analog input signals will be connected to one of the eight available Analog to Digital Converter (ADC) inputs on the AVR, say ADC0 (pin 40), ADC1 (pin 39).

*MIDI Input – 1 Bit Input Line* – MIDI Information. The MIDI input sends the notes into the AVR to store into the AVR’s EEPROM memory. The MIDI input will be received via the USART serial port (pin 14, RXD/PD0). The messages will only consist of Note On and Note Off messages.

*MIDI Output – 1 Bit Output Line* - The MIDI messages will be sent out of the AVR via the USART serial port (pin 15, TXD/PD1).

4. Project Demonstration and Report

At the start of the demo the oscilloscope should be connected to display the USART output, and your programmer should be set up to program the board. MIDI cables should be connected properly, and MIDI-OX should be open and ready to display incoming/outgoing messages from/to your board and to the PC soundcard. Code should be included as a separate file with your report at time of submission. The grading rubric is given in the lab assessment document, and it provides (in italics) the exact sequence of steps that will be used to evaluate each grade component. Include a copy of the lab2 assessment form in your Moodle submission to show your understanding of how you’ve done on the lab. A second copy of the assessment will be completed at the demo and used for grading.

The report should describe your design in a couple of paragraphs (e.g. how do you orchestrate play/record timing? what do your analog inputs do?) The writing should be of good quality, but concise. The report must include the following parts.

- Annotated logic analyzer printouts/screenshots indicating that you analyzed the results and showing whether the design functions correctly or not.
- A brief explanation of the debugging and any problems encountered.
- Describe the contributions of each group member.
- Your report should answer the following questions
  - Given your scheme for measuring timing between notes, what is the maximum imprecision in timing interval between two notes? In other words, if A is the time between the start of two notes during recording, and B is the actual time between when you play the notes back, what is the maximum difference between A and B? Keep in mind that both recording and playback may introduce imprecision.
  - Given your choice of how to store the MIDI notes in EEPROM, what is the largest number of MIDI notes that can be stored and played back?
  - Based on the readings from the ADC and the photocell properties as shown in slides, try to estimate the illuminance of the room under normal conditions and with the sensor covered. Show your calculations.