ECE 303

Final Project

Smart Pill Organizer

[Samuel Kelsch](mailto:kelschsd@miamioh.edu)

01/22/2022

Project Description and Purpose:

Many people who take daily medications have difficulty keeping on schedule and as a result they miss taking doses. This can be problematic and lead to worsening symptoms. To solve this, weekly pill organizers are used to keep people on track. A picture containing graphical user interface

Description automatically generated

For my project I took a pill organizer and used an Arduino to alert the user when it is time to take their medication and keep them on schedule. Two times a day, a light will shine in the compartment that it is time for. For people who may be visually impaired, or if the user is in another room, I also added an alarm that will sound when it is time as well. A display is included that can show the time as well as messages that will show when an alarm sounds.

A picture containing calendar

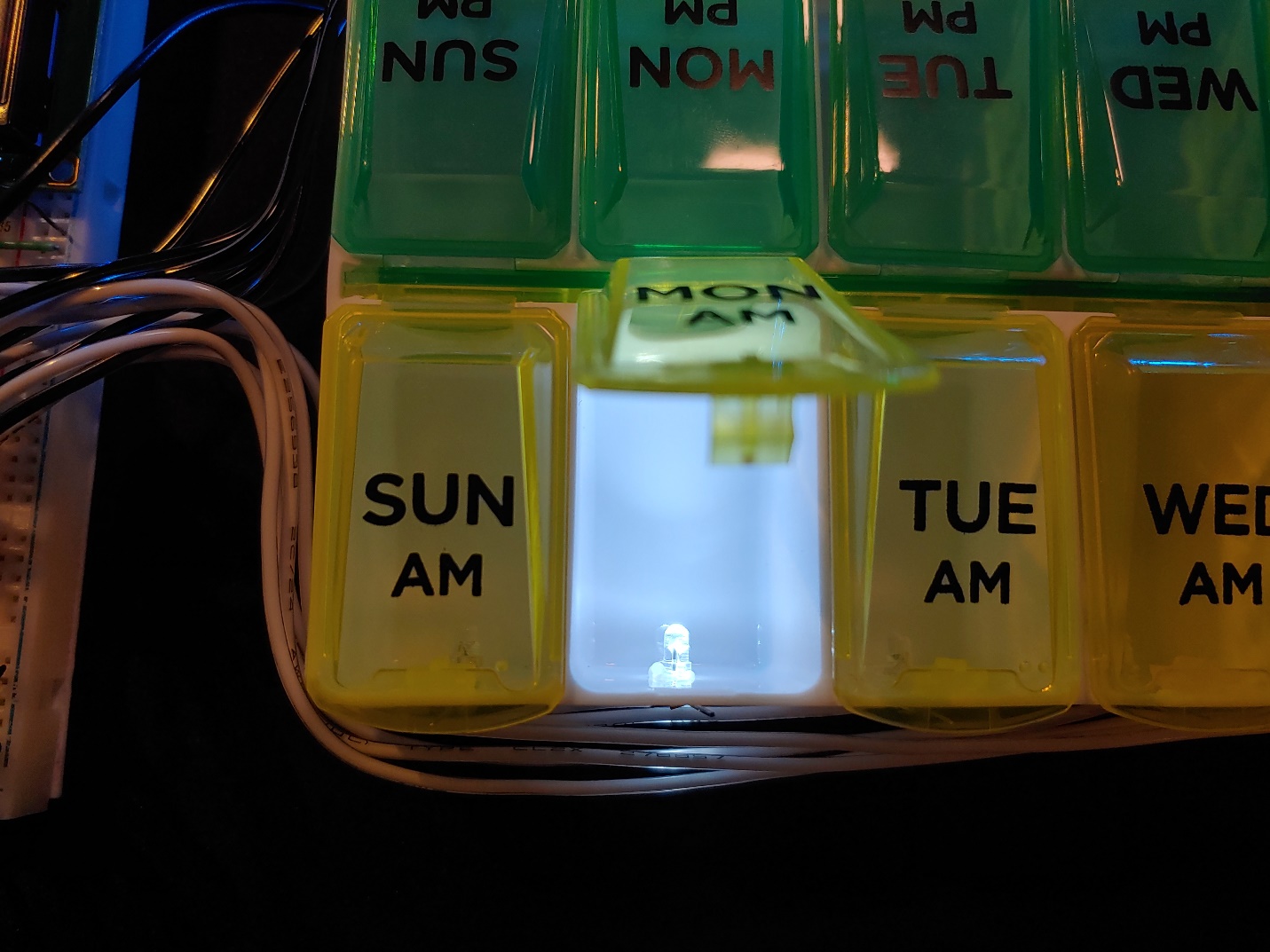
Description automatically generated

Project Details:

To make my design, I used 14 LEDs and put them through a hole in each of the compartments. I soldered all the ground connections together and used 1 resistor to ground them to the Arduino. There shouldn’t be any problems with drawing too much current as only one LED will be lit at a time.

A picture containing indoor

Description automatically generated

I added an LCD to display time and messages. This is connected to the Arduino through 4-bits of data and 2 additional pins were used to control the clock cycles. 4 more pins of the Arduino were used to communicate with 2 shift registers that address each LED individually. While these registers should be chained together, I could not get them working correctly in the time to do the project. I avoided this problem in software by addressing the 2nd register when the limit of the first one is exceeded. For the alarm, I used a piezoelectric speaker to play a short melody whenever an LED comes on. One more pin of the Arduino was used along with a push button to clear the alarm and shut off the LED. The intended use is for daily medications that are taken in the morning and evening. To keep track of the time I used the millis() function. This returns the current elapsed time in milliseconds since the program started running. While this method of keeping track of time is fairly accurate, over long time the time may drift. This could be corrected by using something like a real time clock module to track the time instead.

Conclusion:

Overall, I am happy with the result. I especially enjoy how the wiring to the LEDs turned out. I used a soldering iron to melt a in each compartment that was just the right size for the LEDs I was using. I also used hot glue to hold everything in place to make it robust. To improve the design, I would also add a multiplexer to interface the register to the LEDs. This was the LEDs could be addressed as 1-14 instead of powers of 2.

Arduino Code:

#include<NoDelay.h>

#include<LiquidCrystal.h>

// Create an LCD object. Parameters: (RS, E, Data Lines)

LiquidCrystal lcd = LiquidCrystal(2,3, 4,5,6,7);

int stClk = 8; // RCLK (12)

int srClk = 12; // SRCLK (11)

int dataPin = 11; // SER (14)

int dataPin2 = 10; // SER (14)

int button1 = 9; // Reset push button

int button1State = 0;

char output[100];

int LED = 0; // Which LED should be illuminated

bool hasNotification = 0;

int alarm1 = 9; // Hour of time to give notification

int alarm2 = 18;

void setup(){

Serial.begin(9600);

// Parameters: (cols, rows)

lcd.begin(16,2);

lcd.clear();

lcd.setCursor(0, 0);

pinMode(stClk, OUTPUT);

pinMode(srClk, OUTPUT);

pinMode(dataPin, OUTPUT);

pinMode(dataPin2, OUTPUT);

pinMode(button1, INPUT);

digitalWrite(stClk, LOW); // Sets register to recieve data

shiftOut(dataPin, srClk, LSBFIRST, 0);

shiftOut(dataPin2, srClk, LSBFIRST, 0);

digitalWrite(stClk, HIGH); // Ends communications with register

delay(500);

}

// Time conversions

unsigned long day = 86400000; // 86400000 milliseconds in a day

unsigned long hour = 3600000; // 3600000 milliseconds in an hour

unsigned long minute = 60000; // 60000 milliseconds in a minute

unsigned long second = 1000; // 1000 milliseconds in a second

int startDay = 0;

int startHour = 8;

int startMin = 59;

int startSec = 55;

unsigned long startTime = day \* startDay + hour \* startHour + minute \* startMin + second \* startSec; // Time operation begins in ms

unsigned int currDay = 0;

unsigned int currHour = 0;

unsigned int currMin = 0;

unsigned int currSec = 0;

unsigned int currMil = 0;

unsigned long lastUpdate = 0;

int melody[] = {1000,1500,2000,2500,3000}; // An array of frequencies

void loop(){

button1State = digitalRead(button1);

// What to do when reset button is pushed

if (button1State == HIGH) {

lcd.clear();

LED = 0; // Turns off LEDs

digitalWrite(stClk, LOW); // Sets register to recieve data

shiftOut(dataPin, srClk, LSBFIRST, 0); // Outputs a value of 0 to registers

shiftOut(dataPin2, srClk, LSBFIRST, 0);

digitalWrite(stClk, HIGH); // Ends communications with register

}

if (millis() - lastUpdate > 1000) { // Checks if 1 second has elapsed

lastUpdate = millis() + startTime;

currSec = (lastUpdate/second)%60; // Converts miliseconds to other time units

currMin = (lastUpdate/minute)%60;

currHour = (lastUpdate/hour)%24;

currDay = (lastUpdate/day)%30;

printTime();

//test();

if ((currSec == 0) && (currMin == 0) && (currHour == alarm1)) { // Checks if time = alarm time

LED = (1 << (currDay \* 2));

lcd.setCursor(0, 1);

lcd.print("Time for AM RX");

lcd.setCursor(0, 0);

notify();

}

if ((currSec == 0) && (currMin == 0) && (currHour == alarm2)){

LED = (2 << (currDay \* 2));

lcd.setCursor(0, 1); // Displays secondary message

lcd.print("Time for PM RX");

lcd.setCursor(0, 0);

notify();

}

}

}

// A method to test and cycle through all LEDs

void test(){

while(true){

Serial.print("Day: ");

Serial.println(currDay);

LED = (1 << (currDay \* 2));

notify();

delay(2000);

LED = (2 << (currDay \* 2));

notify();

delay(2000);

currDay += 1;

}

}

// Method to alert user an alarm is triggered

void notify(){

if (LED < 256){ // Uses first register

digitalWrite(stClk, LOW); // Sets register to recieve data

shiftOut(dataPin, srClk, LSBFIRST, LED);

digitalWrite(stClk, HIGH); // Ends communications with register

}

else if(LED >= 256){ // Uses second register

digitalWrite(stClk, LOW); // Sets register to recieve data

shiftOut(dataPin, srClk, LSBFIRST, 0);

shiftOut(dataPin2, srClk, LSBFIRST, (LED >> 8)); // Shifts data by 8 bits to other register

digitalWrite(stClk, HIGH); // Ends communications with register

}

// Sounds alarm

for (int thisNote = 0; thisNote < (sizeof(melody) / sizeof(melody[0])) ; thisNote++){ // Loops for each note in melody[]

tone(13, melody[thisNote], 500); // Plays the note at the position in melody[] for 100ms

delay(500); // Waits for 100ms for the note to finish

}

}

void printTime(){

delay(50);

lcd.setCursor(0,0);

sprintf(output, "day:%02d %02d:%02d:%02d",currDay, currHour, currMin, currSec); // Formats time output to LCD

lcd.print(output);

}