#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

LiquidCrystal\_I2C lcd(0x27, 16, 2); // Address, columns, rows

const int PLAYER\_PIN = 37;

const int BOT\_PIN = 31;

const int PAUSE\_PIN = 35;

const int MINUS\_PLAYER\_PIN = 39;

const int MINUS\_BOT\_PIN = 33;

const int IR\_PLAYER\_PIN = 9;

const int IR\_BOT\_PIN = 8;

const int RED\_LED\_PIN = 10;

const int GREEN\_LED\_PIN = 11;

const int BLUE\_LED\_PIN = 12;

int playerGoals = 0;

int botGoals = 0;

bool paused = false;

unsigned long lastDebounceTimePlayer = 0;

unsigned long lastDebounceTimeBot = 0;

unsigned long lastDebounceTimePause = 0;

unsigned long lastDebounceTimeMinusPlayer = 0;

unsigned long lastDebounceTimeMinusBot = 0;

unsigned long lastDebounceTimeIRPlayer = 0;

unsigned long lastDebounceTimeIRBot = 0;

const unsigned long debounceDelay = 500;

unsigned long startTime;

unsigned long elapsedTime = 0;

int displayMinute = 0;

int displaySecond = 0;

unsigned long pausedTime = 0;

bool resetHold = false;

unsigned long lastGoalTimePlayer = 0;

unsigned long lastGoalTimeBot = 0;

bool beamBrokenPlayer = false;

bool beamBrokenBot = false;

bool ledBlinking = false;

unsigned long blinkStartTime = 0;

const unsigned long blinkDuration = 2000; // 2 seconds

const unsigned long irDebounceDelay = 2000; // 200 milliseconds

// Function to update the LCD screen

void updateScreen() {

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("PLAYER TIME BOT");

char timeBuffer[6];

sprintf(timeBuffer, "%02d:%02d", displayMinute, displaySecond);

lcd.setCursor(0, 1);

lcd.print(" ");

lcd.print(playerGoals);

lcd.print(" ");

lcd.print(timeBuffer);

lcd.print(" ");

lcd.print(botGoals);

lcd.print(" ");

}

void setup() {

lcd.init();

lcd.backlight();

pinMode(PLAYER\_PIN, INPUT\_PULLUP);

pinMode(BOT\_PIN, INPUT\_PULLUP);

pinMode(PAUSE\_PIN, INPUT\_PULLUP);

pinMode(MINUS\_PLAYER\_PIN, INPUT\_PULLUP);

pinMode(MINUS\_BOT\_PIN, INPUT\_PULLUP);

pinMode(IR\_PLAYER\_PIN, INPUT\_PULLUP);

pinMode(IR\_BOT\_PIN, INPUT\_PULLUP);

pinMode(RED\_LED\_PIN, OUTPUT);

pinMode(GREEN\_LED\_PIN, OUTPUT);

pinMode(BLUE\_LED\_PIN, OUTPUT);

startTime = millis();

updateScreen();

}

void loop() {

int playerState = digitalRead(PLAYER\_PIN);

int botState = digitalRead(BOT\_PIN);

int irPlayerState = digitalRead(IR\_PLAYER\_PIN);

int irBotState = digitalRead(IR\_BOT\_PIN);

int pauseState = digitalRead(PAUSE\_PIN);

int minusPlayerState = digitalRead(MINUS\_PLAYER\_PIN);

int minusBotState = digitalRead(MINUS\_BOT\_PIN);

unsigned long currentMillis = millis();

digitalWrite(RED\_LED\_PIN, LOW);

digitalWrite(GREEN\_LED\_PIN, LOW);

digitalWrite(BLUE\_LED\_PIN, HIGH);

// Mål spiller

if ((playerState == HIGH || (irPlayerState == LOW && !beamBrokenPlayer && (currentMillis - lastGoalTimePlayer) > 2000))) {

if (irPlayerState == LOW && (currentMillis - lastDebounceTimeIRPlayer) > irDebounceDelay) {

playerGoals++;

updateScreen();

lastGoalTimePlayer = currentMillis;

beamBrokenPlayer = true;

ledBlinking = true;

blinkStartTime = currentMillis;

lastDebounceTimeIRPlayer = currentMillis;

}

} else if (irPlayerState == HIGH) {

beamBrokenPlayer = false;

}

// Mål bot

if ((botState == HIGH || (irBotState == LOW && !beamBrokenBot && (currentMillis - lastGoalTimeBot) > 2000))) {

if (irBotState == LOW && (currentMillis - lastDebounceTimeIRBot) > irDebounceDelay) {

botGoals++;

updateScreen();

lastGoalTimeBot = currentMillis;

beamBrokenBot = true;

ledBlinking = true;

blinkStartTime = currentMillis;

lastDebounceTimeIRBot = currentMillis;

}

} else if (irBotState == HIGH) {

beamBrokenBot = false;

}

// LED Blinking for goal

if (ledBlinking) {

unsigned long currentTime = millis();

if ((currentTime - blinkStartTime) < blinkDuration) {

digitalWrite(RED\_LED\_PIN, HIGH);

digitalWrite(GREEN\_LED\_PIN, LOW);

digitalWrite(BLUE\_LED\_PIN, HIGH);

delay(250);

digitalWrite(RED\_LED\_PIN, LOW);

digitalWrite(GREEN\_LED\_PIN, HIGH);

digitalWrite(BLUE\_LED\_PIN, LOW);

delay(250);

} else {

ledBlinking = false;

digitalWrite(RED\_LED\_PIN, LOW);

digitalWrite(GREEN\_LED\_PIN, LOW);

digitalWrite(BLUE\_LED\_PIN, LOW);

}

}

// Start/stop

if (pauseState == LOW && !resetHold) {

if ((millis() - lastDebounceTimePause) > 3000) {

resetHold = true;

resetGame();

}

} else if (pauseState == HIGH && !resetHold && (millis() - lastDebounceTimePause) > debounceDelay) {

paused = !paused;

if (paused) {

pausedTime = millis();

} else {

startTime += (millis() - pausedTime);

}

lastDebounceTimePause = millis();

}

if (pauseState == HIGH && resetHold) {

resetHold = false;

lastDebounceTimePause = millis();

}

// minus mål spiller

if (minusPlayerState == HIGH && (millis() - lastDebounceTimeMinusPlayer) > debounceDelay) {

if (playerGoals > 0) playerGoals--;

updateScreen();

lastDebounceTimeMinusPlayer = millis();

}

// minus mål bot

if (minusBotState == HIGH && (millis() - lastDebounceTimeMinusBot) > debounceDelay) {

if (botGoals > 0) botGoals--;

updateScreen();

lastDebounceTimeMinusBot = millis();

}

// Oppdater skjerm

if (!paused) {

elapsedTime = millis() - startTime;

displayMinute = (elapsedTime / 60000) % 60;

displaySecond = (elapsedTime / 1000) % 60;

static unsigned long lastUpdate = 0;

if (millis() - lastUpdate >= 1000) {

lastUpdate = millis();

updateScreen();

}

}

// stop etter 45 min

if (displayMinute >= 45 && !paused) {

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Game Over");

lcd.setCursor(0, 1);

lcd.print("P:" + String(playerGoals) + " B:" + String(botGoals));

while (true);

}

}

// reset

void resetGame() {

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("GAME RESET");

delay(3000);

playerGoals = 0;

botGoals = 0;

startTime = millis();

paused = false;

updateScreen();

}