**Heartbeat Monitoring System**

**🔧 Project Title:**

**Heartbeat Monitoring System with LCD, LED, and Buzzer Alerts**

**📝 Objective:**

To design a system that detects heartbeats using a PulseSensor, monitors the heart rate over time, and provides real-time feedback on a 16x2 I2C LCD. An LED lights up and a buzzer beeps on each beat for user assurance.

**📦 Circuit Connection:-**

**(URL:-** [**https://app.cirkitdesigner.com/project/869db244-8d52-487d-adec-2be95df55854**](https://app.cirkitdesigner.com/project/869db244-8d52-487d-adec-2be95df55854)**)**

**📦 Components Required:**

| **Component** | **Pin No. (Arduino)** | **Destination Component** | **Pin No.** | **Special Remark** |
| --- | --- | --- | --- | --- |
| Pulse Sensor | A0 | Arduino Uno | A0 | Analog pulse input |
| Pulse Sensor | VCC | Arduino Uno | 5V | Power supply |
| Pulse Sensor | GND | Arduino Uno | GND | Ground |
| I2C 16x2 LCD | SDA | Arduino Uno | A4 | I2C Data line |
| I2C 16x2 LCD | SCL | Arduino Uno | A5 | I2C Clock line |
| I2C 16x2 LCD | VCC | Arduino Uno | 5V | Power supply |
| I2C 16x2 LCD | GND | Arduino Uno | GND | Ground |
| Buzzer | D8 | Arduino Uno | D8 | Beeps on each heartbeat |
| LED (any color) | D7 | Arduino Uno | D7 | Blinks on each heartbeat |

**💻 Code Explanation:**

* Waits for **10 clean beats** → transitions to monitoring mode.
* Then counts beats for **60 seconds**, showing:
  + "Beats: N"
  + "Sec: S"
* After 60 seconds, displays BPM for 5 seconds.
* LED and buzzer give instant feedback on each detected beat.

**🧾 Libraries You Need**

* LiquidCrystal\_I2C

***Install them via the Arduino Library Manager (Sketch → Include Library → Manage Libraries).***

**🧠 Arduino Code:**

#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

LiquidCrystal\_I2C lcd(0x27, 16, 2);

const int PULSE\_PIN = A0;

const int BUZZER\_PIN = 8;

const int LED\_PIN = 7;

volatile int BPM;

volatile int Signal;

volatile int IBI = 600;

volatile bool Pulse = false;

volatile bool QS = false;

volatile bool lostPulse = false;

volatile int rate[10];

volatile unsigned long sampleCounter = 0, lastBeatTime = 0;

volatile int P = 512, T = 512, thresh = 525, amp = 100;

volatile bool firstBeat = true, secondBeat = false;

enum CounterState { WAITING, QUALIFYING, MONITORING, SHOW\_RESULT };

CounterState state = WAITING;

int consecutiveBeats = 0;

int beatsInMinute = 0;

unsigned long windowStart = 0;

unsigned long lastSecondRef = 0;

unsigned long resultShownAt = 0;

void printPadded(int value, int width) {

lcd.print(value);

int len = (value == 0) ? 1 : (int)log10(value)+1;

for (int i = len; i < width; i++) lcd.print(' ');

}

void setup() {

Serial.begin(115200);

lcd.init();

lcd.backlight();

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Place finger...");

pinMode(LED\_PIN, OUTPUT);

pinMode(BUZZER\_PIN, OUTPUT);

interruptSetup();

}

void loop() {

if (lostPulse) {

lostPulse = false;

state = WAITING;

consecutiveBeats = beatsInMinute = 0;

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Place finger...");

}

if (QS) {

QS = false;

digitalWrite(LED\_PIN, HIGH);

digitalWrite(BUZZER\_PIN, HIGH);

delay(20);

digitalWrite(LED\_PIN, LOW);

digitalWrite(BUZZER\_PIN, LOW);

switch (state) {

case WAITING:

state = QUALIFYING;

consecutiveBeats = 1;

lcd.clear();

lcd.print("Qualify: 1/10");

break;

case QUALIFYING:

consecutiveBeats++;

lcd.setCursor(9, 0);

printPadded(consecutiveBeats, 2);

if (consecutiveBeats >= 10) {

state = MONITORING;

beatsInMinute = 1;

windowStart = millis();

lastSecondRef = millis();

lcd.clear();

lcd.print("Beats: 1 ");

lcd.setCursor(0, 1);

lcd.print("Sec: 0 ");

}

break;

case MONITORING:

beatsInMinute++;

lcd.setCursor(7, 0);

printPadded(beatsInMinute, 3);

break;

default:

break;

}

}

if (state == MONITORING && millis() - lastSecondRef >= 1000UL) {

lastSecondRef += 1000UL;

int secPassed = (millis() - windowStart) / 1000;

lcd.setCursor(5, 1);

printPadded(secPassed, 2);

}

if (state == MONITORING && millis() - windowStart >= 60000UL) {

state = SHOW\_RESULT;

resultShownAt = millis();

lcd.clear();

lcd.print("Beats/min:");

lcd.setCursor(0, 1);

lcd.print(beatsInMinute);

Serial.print("Beats in one minute: ");

Serial.println(beatsInMinute);

}

if (state == SHOW\_RESULT && millis() - resultShownAt >= 5000UL) {

state = WAITING;

consecutiveBeats = beatsInMinute = 0;

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Place finger...");

}

delay(20);

}

ISR(TIMER2\_COMPA\_vect) {

cli();

Signal = analogRead(PULSE\_PIN);

sampleCounter += 2;

int N = sampleCounter - lastBeatTime;

if (Signal < thresh && N > (IBI / 5) \* 3) { if (Signal < T) T = Signal; }

if (Signal > thresh && Signal > P) { P = Signal; }

if (N > 250) {

if ((Signal > thresh) && !Pulse && (N > (IBI / 5) \* 3)) {

Pulse = true;

IBI = sampleCounter - lastBeatTime;

lastBeatTime = sampleCounter;

if (secondBeat) {

secondBeat = false;

for (int i = 0; i < 10; i++) rate[i] = IBI;

}

if (firstBeat) {

firstBeat = false;

secondBeat = true;

sei();

return;

}

long runTot = 0;

for (int i = 0; i < 9; i++) { rate[i] = rate[i + 1]; runTot += rate[i]; }

rate[9] = IBI; runTot += rate[9];

BPM = 60000 / (runTot / 10);

QS = true;

}

}

if (Signal < thresh && Pulse) {

Pulse = false;

amp = P - T;

thresh = amp / 2 + T;

P = thresh; T = thresh;

}

if (N > 2500) {

thresh = P = T = 512;

lastBeatTime = sampleCounter;

firstBeat = true; secondBeat = false;

lostPulse = true;

}

sei();

}

void interruptSetup() {

TCCR2A = 0x02;

TCCR2B = 0x06;

OCR2A = 0x7C;

TIMSK2 = 0x02;

sei();

}

**📊 Output Examples:**

**Before Pulse Detected:**

Place finger...

**While Qualifying:**

Qualify: 4/10

**Live Monitoring:**

Beats: 12

Sec: 9

**Final Result (After 60s):**

Beats/min:

78

**✅ Notes:**

* Adjust thresh, amp, and IBI if detection is unstable.
* Use a stable power supply to avoid false triggers.
* Keep finger steady for consistent readings.

Would you like a downloadable PDF version of this documentation?