Proiect: Stopwatch şi Countdown pe Arduino Mega

Documentaţie şi Ghid utilizare

Autor: **Tcaci Liviu**

Grupa: **30231**

Universitatea Tehnică din Cluj-Napoca
Facultatea de Automatică şi Calculatoare

16 Jan 2025

Cuprins

1 Introducere

Acest proiect își propune realizarea unui dispozitiv multifuncțional pe bază de **Arduino Mega**, care să ofere atât funcționalitatea de *cronometru (stopwatch)*, cât și un *timer (count-down)*. Dispozitivul folosește un ecran LCD de 16×2 pentru afișarea informațiilor și patru butoane: Start/Stop, Reset, Lap, plus un buzzer care semnalizează sfârșitul timpului setat.

În **modul Stopwatch**, se pot măsura intervale de timp și înregistra runde (laps), afișând dinamic timpul total. În **modul Timer**, se permite setarea unui interval (minute și secunde), iar la expirarea acestuia se afișează un mesaj și se emite un semnal sonor.

Prin combinarea celor două funcționalități, proiectul demonstrează o manieră simplă de a lucra cu intrări digitale, afișaj LCD și buzzer, integrând într-o singură platformă aplicații de măsurare și gestionare a timpului.

2 Design Hardware

2.1 Lista de componente

Proiectul **Stopwatch şi Timer pe Arduino Mega** utilizează următoarele componente hardware:

- **Arduino Mega 2560**: Placa de dezvoltare principală care gestionează toate funcționalitățile proiectului.
- **LCD 16x2**: Afișajul utilizat pentru prezentarea informațiilor privind timpul total, runde (laps) și starea curentă a dispozitivului.
- **4× Butoane pushbutton**:
  - **Start/Stop**: Controlează pornirea și oprirea cronometrelor.
  - **Reset**: Resetează valorile timpului sau comută între modurile Stopwatch și Timer prin apăsare scurtă sau lungă.
  - **Lap**: Înregistrează runde în modul Stopwatch și adaugă secunde suplimentare în modul Timer.
- **4× Rezistențe de 10kΩ**: Utilizate pentru rezistențele de pull-down la butoane, asigurând stabilitatea citirii semnalelor digitale.
- **Buzzer**: Emită semnale sonore la finalizarea unui interval de timp setat în modul Timer.
- **Cablu de conectare și breadboard**: Facilită conectarea componentelor între ele și cu placa Arduino Mega.

2.2 Conexiuni Circuit

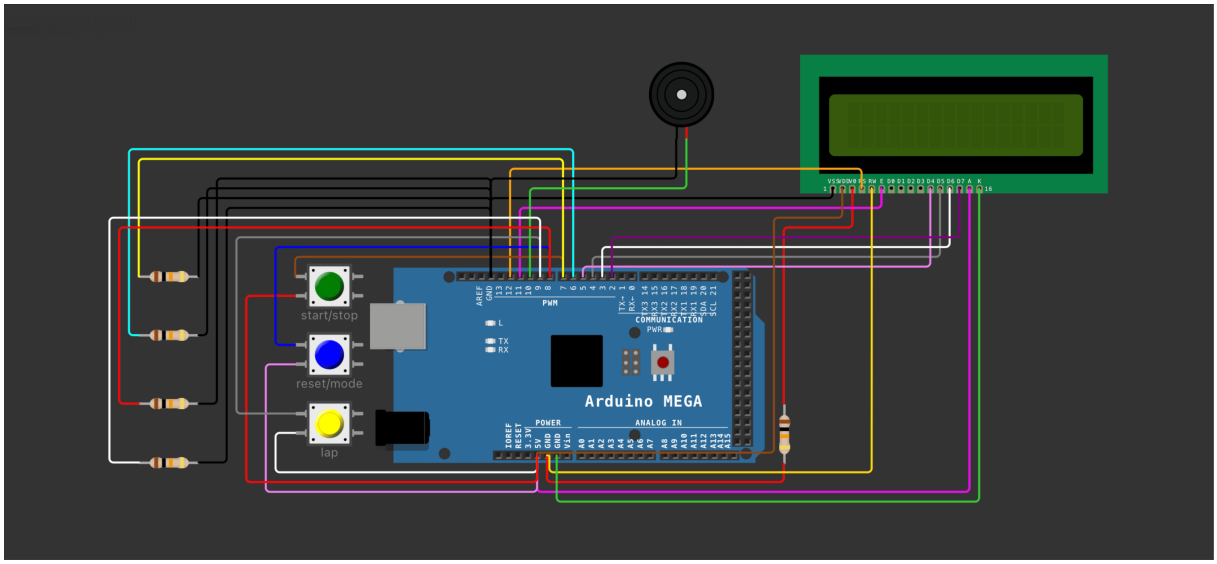

Figura 1: Schema de conexiuni pentru proiectul Stopwatch şi Timer pe Arduino Mega

2.2.1 Detalii Conexiuni

- **LCD 16x2**:
  - **VSS** la GND pe Arduino Mega.
  - **VDD** la 5V pe Arduino Mega.
  - **V0** printr-o rezistență de 10kΩ la GND pentru controlul contrastului.
  - **RS** la pinul digital 12 pe Arduino.
  - **E** la pinul digital 11 pe Arduino.
  - **D4-D7** la pinii digitali 5, 4, 3 și 2 pe Arduino.
  - **A** (Anod) la 5V printr-o rezistență limitatoare de curent.
  - **K** (Catod) la GND.
- **Butoane Pushbutton**:
  - Fiecare buton are o rezistență de 10kΩ conectată între pinul său și GND pentru a asigura o citire stabilă a semnalului.
  - **Start/Stop** conectat la pinul digital 7 pe Arduino.
  - **Reset** conectat la pinul digital 8 pe Arduino.
  - **Lap** conectat la pinul digital 9 pe Arduino.
- **Buzzer**:
  - Terminalul pozitiv al buzzerului conectat la pinul digital 10 pe Arduino.
  - Terminalul negativ conectat la GND pe Arduino Mega.

2.2.2 Schema de Alimentare

Toate componentele necesită alimentare de la sursa principală de 5V a plăcii Arduino Mega. Asigură-te că toate conexiunile de GND sunt comune pentru a preveni diferențele de potențial care ar putea cauza funcționarea incorectă a circuitului.

3 Design Software

3.1 Structura Codului

Codul proiectului **Stopwatch şi Timer pe Arduino Mega** este organizat într-un singur fișier `stopwatch.ino`, care conține funcțiile principale `setup()` și `loop()`, precum și o serie de funcții auxiliare pentru gestionarea diferitelor funcționalități ale dispozitivului. Structura generală a codului este următoarea:

- `setup()`:
  - Inițializează comunicația serială pentru debugging.
  - Configurează pinii utilizați pentru buzzer, LCD și butoane.
  - Initializează ecranul LCD și afișează mesajul de pornire.
  - Atașează obiectele `Bounce` pentru debouncing-ul butoanelor.
- `loop()`:
  - Actualizează starea butoanelor utilizând biblioteca `Bounce2`.
  - Gestionează logica buzzerului dacă este activ.
  - Detectează și gestionează apăsările scurte și lungi ale butonului Reset.
  - În funcție de modul curent (`Stopwatch` sau `Timer`), execută logica specifică:
    - `handleStopwatchLogic()`
    - `handleCountdownLogic()`
  - Gestionarea afișării mesajelor de finalizare în modul Timer.

3.2 Gestionarea Funcționalităților

3.2.1 Logica Stopwatch

Funcția `handleStopwatchLogic(unsigned long currentMillis)` este responsabilă pentru gestionarea funcționalității de cronometru. Aceasta include:

- **Start/Oprire**: Detectează apăsarea butonului Start/Stop pentru a porni sau opri cronometru.
- **Înregistrarea Rondei (Lap)**: La apăsarea butonului Lap, se înregistrează timpul curent ca o rundă și se actualizează afișajul.
- **Afișarea Timpului**: Actualizează afișajul LCD cu timpul total și timpul rundei curente.
- **Blinking pentru Runda Anterioară**: După înregistrarea unei runde, timpul rundei anterioare este afișat intermitent pentru o perioadă specificată.

3.2.2 Logica Timer (CountDown)

Funcția `handleCountdownLogic(unsigned long currentMillis)` gestionează funcționalitatea de timer, care include:

- **Pornire/Pauză**: Detectează apăsarea butonului Start/Stop pentru a începe sau a pune în pauză numărătoarea inversă.
- **Setarea Timpului**: La apăsarea butonului Lap, se adaugă 10 secunde la timpul setat.
- **Decrementarea Timpului**: Dacă timerul rulează, timpul este decrementat la fiecare secundă.
- **Finalizarea Timerului**: Când timpul ajunge la zero, se activează buzzerul și se afișează mesajul `"TIME's UP!"` pe LCD pentru 5 secunde.

3.2.3 Gestionarea Butonului Reset

Funcția `handleResetButtonLogic(unsigned long currentMillis)` diferențiază între o apăsare scurtă și una lungă a butonului Reset:

- **Apăsare Lungă ($\geq$ 0.5s)**: Comută între modul Stopwatch și Timer.
- **Apăsare Scurtă ($<$ 0.5s)**: Resetează valorile cronometru sau timer, în funcție de modul curent.

3.3 Funcții Auxiliare

3.3.1 FormatTime și FormatCountdown

Aceste funcții transformă valorile numerice ale timpului în șiruri de caractere formate corespunzător pentru afișare pe LCD.

```
String formatTime(unsigned long msVal) {
    unsigned long hours = msVal / 3600000;
    unsigned long minutes = (msVal / 60000) % 60;
    unsigned long seconds = (msVal / 1000) % 60;
    unsigned long milliseconds = (msVal % 1000) / 100;

    String formattedTime;
    formattedTime += String(hours) + ":";
    if (minutes < 10) formattedTime += "0";
    formattedTime += String(minutes) + ":";
    if (seconds < 10) formattedTime += "0";
    formattedTime += String(seconds) + ".";
    formattedTime += String(milliseconds);

    return formattedTime;
}

String formatCountdown(unsigned int mm, unsigned int ss) {
    String result;
    if (mm < 10) result += "0";
    result += String(mm) + ":";
    if (ss < 10) result += "0";
    result += String(ss);
    return result;
}
```

Figura 2: Funcțiile `formatTime` și `formatCountdown` pentru formatarea timpului în modul Stopwatch și Timer.

3.3.2 handleBuzzer

Funcția `handleBuzzer(unsigned long currentMillis)` gestionează activarea și dezactivarea buzzerului atunci când este necesar.

```
void handleBuzzer(unsigned long currentMillis) {
    if (buzzerActive) {
        // Activăm buzzerul
        digitalWrite(buzzerPin, HIGH);
        // Verificăm dacă timpul de beep a trecut
        if ((currentMillis - buzzerStartTime) >= timeIsUpBlinkDuration) {
            buzzerActive = false;
            digitalWrite(buzzerPin, LOW);
            Serial.println("Buzzer OFF => 5s beep done.");
        }
    }
}
```

Figura 3: Funcția `handleBuzzer` pentru activarea și dezactivarea buzzerului.

3.3.3 startNewLap și prepareNewLap

Aceste funcții gestionează înregistrarea și inițializarea rundelor (laps) în modul Stopwatch.

```
void prepareNewLap() {
    lapCount++;
    int lapIndex = lapCount % maxLaps;
    laps[lapIndex].number = lapCount % maxLaps;
    laps[lapIndex].time = totalElapsedTime;
}

void startNewLap() {
    int lapIndex = lapCount % maxLaps;
    laps[lapIndex].number = lapCount % maxLaps;
    laps[lapIndex].time   = totalElapsedTime;

    lcd.setCursor(0, 0);
    lcd.print("Lap" + String(laps[lapIndex].number)
              + ": " + formatTime(0) + "    ");

    lapStartTime = millis();
    Serial.print("startNewLap => Lap ");
    Serial.println(laps[lapIndex].number);
}
```

Figura 4: Funcțiile `prepareNewLap` și `startNewLap` pentru gestionarea rundelor în modul Stopwatch.

3.4 Gestionarea Debouncing-ului Butoanelor

Pentru a asigura că semnalele de la butoane sunt stabile și nu sunt afectate de zgomotul electric (*bouncing*), proiectul utilizează biblioteca `Bounce2`. Obiectele `Bounce` sunt inițializate pentru fiecare buton și actualizate la fiecare ciclu al funcției `loop()`.

```
Bounce debouncerStartStop = Bounce();
Bounce debouncerReset     = Bounce();
Bounce debouncerLap       = Bounce();

void setup() {
    // ... (alte inițializări)
    debouncerStartStop.attach(buttonStartStopPin);
    debouncerStartStop.interval(25);

    debouncerReset.attach(buttonResetPin);
    debouncerReset.interval(25);

    debouncerLap.attach(buttonLapPin);
    debouncerLap.interval(25);
}

void loop() {
    debouncerStartStop.update();
    debouncerReset.update();
    debouncerLap.update();

    // ... (restul codului)
}
```

Figura 5: Configurarea și utilizarea obiectelor `Bounce` pentru debouncing-ul butoanelor.

3.5 Fluxul Principal al Programului

1. Inițializare:
   - Setarea pinilor și inițializarea componentelor hardware în `setup()`.
   - Afișarea mesajelor de pornire pe LCD.
2. Loop Principal:
   - Actualizarea stării butoanelor pentru a detecta apăsările.
   - Gestionarea buzzerului dacă este activ.
   - Detectarea și gestionarea apăsărilor butonului Reset.
   - Executarea logicii specifice modulului curent (Stopwatch sau Timer).
   - Gestionarea afișării mesajelor de finalizare în modul Timer.

3.6 Managementul Modulului Stopwatch

Logica pentru modul Stopwatch include gestionarea cronometrelor, înregistrarea rundei (lap), afișarea timpului și manipularea butoanelor Start/Stop și Lap. Fiecare rundă este stocată într-un array de structuri `Lap`, care conține numărul rundei și timpul la care a fost înregistrată.

3.7 Managementul Modulului Timer

Logica pentru modul Timer permite setarea unui interval de timp (minute și secunde), pornirea și oprirea numărătorii inversă și semnalizarea finalului intervalului prin afișarea unui mesaj

și activarea buzzerului. Utilizatorul poate adăuga 10 secunde la timpul setat prin apăsarea butonului Lap.

3.8 Gestionarea Modului și Resetarea

Butonul Reset are două funcționalități diferite, în funcție de durata apăsării:

- **Apăsare Lungă ($\geq$ 0.5s)**: Comută între modul Stopwatch și Timer.
- **Apăsare Scurtă ($<$ 0.5s)**: Resetează valorile cronometru sau timer.

Această distincție este realizată prin măsurarea duratei între începutul și sfârșitul apăsării butonului Reset.

3.9 Exemplu de Aplicație

Pentru a ilustra funcționarea codului, să considerăm următorul scenariu:

1. Utilizatorul pornește dispozitivul și vede mesajul `"Press Start"` pe LCD.
2. Apasă butonul Start/Stop pentru a începe cronometrajul. LCD afișează timpul curent și așteaptă comenzi.
3. După un anumit interval, utilizatorul apasă butonul Lap pentru a înregistra o rundă. LCD afișează timpul rundei și continuă cronometrajul.
4. Utilizatorul apasă din nou butonul Start/Stop pentru a opri cronometrajul. LCD afișează timpul total înregistrat.
5. În modul Timer, utilizatorul setează un interval și pornește numărătoarea inversă. La expirare, LCD afișează `"TIME's UP!"` și buzzerul emite un semnal sonor pentru 5 secunde.

4 Ghid de Utilizare

Această secțiune explică modul de utilizare al dispozitivului în cele două moduri disponibile: **Stopwatch** și **Timer**.

4.1 Modul Stopwatch

- **Start/Oprire**: Apăsați butonul **Start/Stop** pentru a porni cronometrul. Apăsați din nou pentru a-l pune în pauză.
- **Înregistrare Lap**: În timpul funcționării, apăsați butonul **Lap** pentru a înregistra o nouă rundă.
- **Resetare**: Apăsați rapid butonul **Reset** pentru a reseta toate valorile la zero.

4.2 Modul Timer

- **Activare Timer**: Țineți apăsat butonul **Reset** timp de 1–2s pentru a comuta în modul Timer.
- **Adăugare Secunde**: Apăsați butonul **Lap** pentru a adăuga 10 secunde la intervalul de timp setat.
- **Start/Pauză**: Apăsați butonul **Start/Stop** pentru a începe sau a pune pe pauză numărătoarea inversă.
- **Finalul Timpului**: Când timpul ajunge la `00:00`, pe ecranul LCD apare mesajul intermitent **"TIME's UP!"**, iar buzzer-ul emite un semnal sonor timp de 5s.

5 Exemple de Utilizare

Pentru a ilustra funcționalitățile dispozitivului, vom prezenta două scenarii tipice de utilizare, câte unul pentru fiecare mod: **Stopwatch** și **Timer**.

5.1 Scenariu 1: Măsurarea unui Timp Scurt cu Modul Stopwatch

Un utilizator dorește să cronometreze timpul necesar pentru a finaliza un traseu scurt de alergare:

1. Utilizatorul pornește dispozitivul și vede pe ecran mesajul `"Press Start"`.
2. Apasă butonul **Start/Stop** pentru a începe cronometrul. Ecranul afișează timpul curent în format `H:MM:SS.s`.
3. După finalizarea traseului, utilizatorul apasă butonul **Start/Stop** pentru a opri cronometrul. Ecranul arată timpul total înregistrat, de exemplu: `"Time:  0:00:47.5"`.
4. Dacă utilizatorul dorește să reseteze dispozitivul, apasă rapid butonul **Reset**, iar valorile sunt resetate la zero.

5.2 Scenariu 2: Setarea unui Countdown de 30 de Secunde în Modul Timer

Un utilizator dorește să seteze un timer pentru 30 de secunde pentru a cronometră timpul unei pauze scurte:

1. Utilizatorul intră în modul **Timer** ținând apăsat butonul **Reset** timp de 1–2s. Pe ecran apare mesajul `"Timer Ready"`.
2. Apasă butonul **Lap** de trei ori pentru a adăuga 30 de secunde (câte 10 secunde per apăsare).
3. Apasă butonul **Start/Stop** pentru a începe numărătoarea inversă. Ecranul afișează timpul rămas, de exemplu: `"T: 00:30"`.
4. După ce timpul ajunge la `"00:00"`, ecranul arată intermitent mesajul `"TIME's UP!"`, iar buzzer-ul emite un semnal sonor timp de 5 secunde.
5. Pentru a reseta timerul, utilizatorul apasă rapid butonul **Reset**. Ecranul revine la starea inițială.