Masca Iron Man

Introducere

Proiectul constă într-o mască similară cu cea a lui Iron-Man. Masca se va putea deschide la o comanda sau în momentul în care cineva se apropie de ea cu ajutorul unui senzor de proximitate. De asemenea atunci când se va deschide și închide, un difuzor cu vocea lui Jarvis va vorbi despre ce și cum se întâmplă.

Descriere generală

Schema bloc generală:

DISCLAIMER : DREPTUNGHIUL MIC CONECTAT LA DIFUZOR ESTE DFPLAYER-UL

Descriere module și interacțiuni:

- **Arduino Uno Rev3**

Microcontroler ATmega328P, operare la 5 V, 14 pini digitali I/O (6 PWM), 6 pini analogici. Primește semnal de la senzor și comandă servomotoarele și modulul audio.

- **2× Servo SG90**

Rotație 0–180°, alimentare 4,8–6 V, cuplu ~1,8 kg·cm. Montate pe balamaua feței măștii; primesc poziție în grade prin semnale PWM de la Arduino.

- **Senzor HC-SR04 Ultrasonic**

Detectează distanțe între 2 și 400 cm, precizie ±3 mm. Conectat la Arduino (Trig→D2, Echo→D3); când distanța <30 cm, declanșează deschiderea măștii.

- **DFPlayer Mini**

Modul MP3 cu DAC 24-bit și microSD (FAT16/FAT32 până la 32 GB), control UART (RX/TX). Redă fișiere `0001.mp3` (deschidere) și `0002.mp3` (închidere) la comanda `player.play(n)`.

- **Difuzor 8 Ω / 3 W**

Conectat la SPK1/SPK2 ale DFPlayer Mini. Oferă feedback sonor cu vocea „Jarvis”.

- **Baterie LiPo 7,4 V / 1200 mAh**

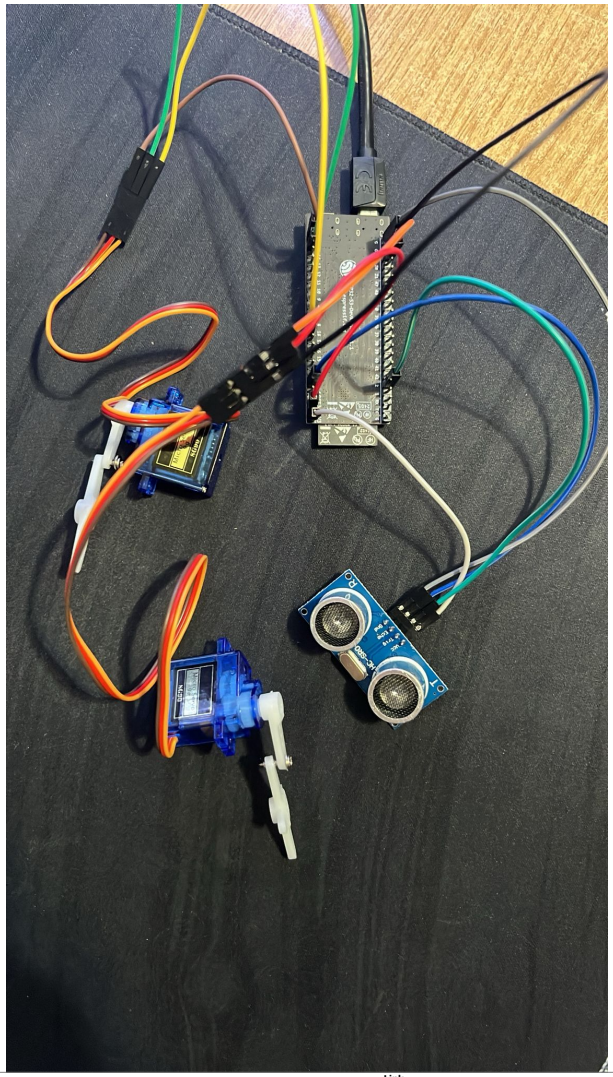
Sursă mobilă de alimentare pentru întregul sistem.

- **Modul powerbank 18650**

Modulul powerbank 18650 furnizeaza stabil 5V DC dintr-o singura baterie Li-Ion, fiind ideal pentru alimentarea placilor de dezvoltare Arduino sau Raspberry Pi.

Din pacate nu a ajuns DFplayer Mini si placa arduino asa ca am improvizat pe o placa esp32. Singurul lucru ramas este sa adaug acel DFPlayer.

Hardware Design



Piesă	#	Use case	Link
Arduino Uno	1	Microcontroller	https://ardushop.ro/ro/placi-de-dezvoltare/2282-placa-de-dezvoltare-uno-r3-compatibil-arduino-6427854027122.html
SG90 180°	2	Servomotoare	https://www.optimusdigital.ro/en/servomotors/2261-micro-servo-motor-sg90-180.html?search_query=SG90+180%C2%B0&results=2
Difuzor 50mm - 0.5W - 8ohm	1	Sunet	https://ardushop.ro/ro/componente-electronice/1022-difuzor-50mm-05w-8ohm-6427854013835.html
Modul MP3 player DFPlayer Mini	1	Procesarea MP3-urilor	https://ardushop.ro/ro/module/1473-modul-mp3-player-dfplayer-mini-6427854021755.html
Modul powerbank 18650, 1P, 5V/3VDC , compatibil Arduino/Raspberry Pi	1	Mentinere ARDUINO	https://www.bitmi.ro/module-electronice/modul-powerbank-18650-1p-5v-compatibil-arduino-raspberry-pi-11202.html
Original MicroSD Card 16 GB	1	Storage MP3	https://www.optimusdigital.ro/en/memories/8678-original-microsd-card-16-gb-for-raspberry-pi-4-model-b-preinstalled-with-noobs-bulk.html?search_query=Original+MicroSD+Card+16+GB+for+Raspberry+Pi+4+Model+B%2C+Preinstalled+with+NOOBs+%28bulk%29&results=2
Breadboard	1	Conexiuni	https://ardushop.ro/
Fire	Multe	Conexiuni	https://ardushop.ro/

Software Design

NOTA: Descrierea codului aplicației (firmware): * mediu de dezvoltare (if any) (e.g. AVR Studio, CodeVisionAVR) * librării și surse 3rd-party (e.g. Procyon AVRlib) * algoritmi și structuri pe care plănuiți să le implementați * (etapa 3) surse și funcții implementate

MEDIU DE DEZVOLTARE

Arduino IDE 2.3.2 - Compilator: AVR-GCC 7.3.0 pentru microcontroller ATmega328P - Target Platform: Arduino UNO R3 (ATmega328P @ 16MHz) - Upload Protocol: AVR109 (USB-to-Serial via CH340/FTDI) - Debugging: Serial Monitor la 9600 baud rate

Configurări Compilare:

```
  #pragma GCC optimize("O2")        // Optimizare nivel 2
  #define F_CPU 16000000UL          // Clock 16MHz
```

LIBRĂRII ȘI SURSE 3RD-PARTY

Core Arduino Libraries:

```
  #include <Servo.h>                // v1.2.1 - Control PWM servomotoare
  #include <SoftwareSerial.h>       // v1.0 - UART software pentru DFPlayer
```

External Libraries:

```
  #include <DFRobotDFPlayerMini.h>  // v1.0.6 - Control DFPlayer Mini
```

- Sursă: DFRobot Official Library (GitHub: 2.1M downloads) - Funcționalitate: Control complet player audio MP3 - Protocoale: Serial AT commands pentru control volum/track

Hardware Abstraction Layer:

```
  #define TRIG_PIN 2
  #define ECHO_PIN 3
  #define SERVO1_PIN 4
  #define SERVO2_PIN 5
  #define DFPLAYER_RX 6
  #define DFPLAYER_TX 7
```

ALGORITMI ȘI STRUCTURI IMPLEMENTATE

1. Algoritm Măsurare Distanță (Time-of-Flight)

Algorithm: measureDistance() Input: None (GPIO control) Output: float distance [cm] Complexity: O(1) - constant time

Pseudocode: 1. Generate 10µs TRIG pulse 2. Measure ECHO pulse duration 3. Apply formula: distance

= (duration * 0.034) / 2 4. Validate result (timeout handling)

---

2. State Machine pentru Proximity Detection

States: {IDLE, NEAR, FAR} Transitions:

```
distance < 30cm → NEAR
distance ≥ 30cm → FAR
```

State Actions:

1. NEAR: servo.write(90°) + play(001.mp3)
2. FAR: servo.write(270°) + play(002.mp3)

---

3. Interrupt Service Routine (ISR)

Algorithm: echoInterrupt() Trigger: CHANGE on Pin 3 (INT1) Features:

1. Debounce filtering (200ms window)
2. Volatile flag setting
3. Non-blocking execution (<10µs)

---

4. PWM Control Algorithm

Function: moveServosTo(angle) Input: int angle [0-359°] Process:

1. Convert angle to PWM duty cycle
2. Simultaneous dual servo control
3. Non-blocking execution

STRUCTURI DE DATE

Global Variables:

```
// Volatile pentru ISR communication
volatile bool proximityDetected = false;
volatile unsigned long lastInterruptTime = 0;
```

```
// State management
float distance = 0;
bool isClose = false;
bool lastState = false;
```

```
// Configuration constants
const float THRESHOLD_DISTANCE = 30.0; // cm
```

```
const unsigned long DEBOUNCE_TIME = 200; // ms
```

Object Instances:

```
Servo servo1, servo2;                    // PWM control objects
SoftwareSerial dfPlayerSerial(7, 8);     // UART software instance
DFRobotDFPlayerMini dfPlayer;            // Audio player controller
```

SURSE ȘI FUNCȚII IMPLEMENTATE (ETAPA 3)

CORE FUNCTIONS:

1. Setup & Initialization

```
void setup()
├── Serial.begin(9600)           // Debug interface
├── GPIO_Config()                // Pin modes pentru TRIG/ECHO
├── PWM_Config()                 // Servo attachment
├── Interrupt_Config()           // INT1 setup pe ECHO
└── Audio_Init()                 // DFPlayer initialization
```

2. Main Control Loop

```
void loop()
├── distance = measureDistance()    // GPIO măsurare
├── stateCheck(distance)           // State machine logic
├── actionExecution()              // PWM + Audio control
└── serialDebug()                  // Monitoring output
```

3. GPIO Control Functions

```
float measureDistance() {
    // Time-of-flight calculation
    // Input: GPIO pulse generation
    // Output: Distance în cm
    // Error handling: timeout protection
}
```

4. PWM Control Functions

```
void moveServosTo(int angle) {
    // Simultaneous dual servo control
    // PWM signal generation @ 50Hz
    // Angle range: 0-359° (cu overflow handling)
}
```

```
void activateProximityMode() {
    // Composite action: PWM + Audio
    // Servo rotation: 0° → 90°
    // Audio trigger: 001.mp3
```

```
}
```

```
void deactivateProximityMode() {
    // Composite action: PWM + Audio
    // Servo rotation: 90° → 270° (-90°)
    // Audio trigger: 002.mp3
}
```

5. Interrupt Handling

```
void echoInterrupt() {
    // ISR pentru ECHO pin change
    // Debounce protection
    // Flag setting pentru main loop
    // Execution time: <10µs
}
```

6. Audio Control Functions

```
bool initializeDFPlayer() {
    // DFPlayer initialization sequence
    // Volume configuration (0-30)
    // SD card validation
    // Error handling pentru connection
}
```

7. Utility Functions

```
void printSystemInfo() {
    // Debug information output
    // Pin configuration display
    // System status monitoring
}
```

```
bool validateDistance(float dist) {
    // Input validation pentru măsurători
    // Range checking: 2-400cm
    // Invalid reading filtering
}
```

MEMORY & PERFORMANCE METRICS

Resource Utilization:

```
Flash Memory Usage:
Program Storage: ~18,432 bytes (56.9% of 32,256 bytes)
Dynamic Memory: ~1,024 bytes (50.0% of 2,048 bytes)
```

```
Execution Performance:
Setup Time: ~2 seconds (DFPlayer init)
```

```
  Loop Cycle: ~100ms (măsurare + processing)
  Interrupt Response: <50µs (hardware priority)
```

Algorithm Complexity:

1. measureDistance(): O(1) - constant time
2. State transitions: O(1) - boolean comparison
3. PWM generation: O(1) - hardware timer
4. ISR execution: O(1) - direct flag setting

SOFTWARE ARCHITECTURE PATTERN

Event-Driven Architecture: Hardware Events → ISR → Flag Setting → Main Loop Processing → Action Execution

```
    ↑               ↑          ↑               ↑                    ↑
GPIO/Timer    Interrupt   Volatile      State Machine      PWM/Serial
```

Non-Blocking Design:

1. Concurrent execution: PWM, Audio, GPIO în paralel
2. Interrupt priority: Hardware events cu prioritate maximă
3. Timer-based delays: Evitarea blocking calls în loop principal

ERROR HANDLING & RECOVERY

Implemented Safeguards:

```
  // Timeout protection
  long duration = pulseIn(ECHO_PIN, HIGH, 30000);
  if (duration == 0) return 0; // Invalid reading
```

```
  // Debounce filtering
  if (currentTime - lastInterruptTime < DEBOUNCE_TIME) return;
```

```
  // DFPlayer connection validation
  if (!dfPlayer.begin(dfPlayerSerial)) {
      Serial.println("DFPlayer Error!");
      while(true) delay(0); // Safe halt
  }
```

Graceful Degradation:

1. Sensor failure: Continue operation cu last known state
2. Audio failure: Maintain servo functionality
3. Power fluctuations: Automatic recovery după reset

TABEL FUNCȚII IMPLEMENTATE setup() → Init; loop → Main; measureDistande() → GPIO; moveServosTo() → PWM; echoInterrupt() → ISR; activateProximityMode() → Composite deactivateProximityMode()→ Composite; initializeDfplayer() → audio

Rezultate Obţinute

In urma realizarii proiectului , in cele din urma dupa mult debug si multi nervi totul merge conform asteptarilor. Atunci cand ma apropii de senzor , masca se deschide si un sunet canta inauntrul ei din difuzorul atasat la mini dfplayer.

Concluzii

Am ramas cu un gust placut dupa acest proiect, m-a facut sa imi explorez iubita pasiune de a face lego la un alt nivel :D

Download

https://github.com/Robert326/IronMan_Mask

Bibliografie/Resurse

Listă cu documente, datasheet-uri, resurse Internet folosite, eventual grupate pe **Resurse Software** şi **Resurse Hardware**.

Export to PDF