Sistem Inteligent de Semaforizare cu Feedback Vizual si Sonor

Introducere

Proiectul consta in implementarea unui sistem interactiv de semaforizare pentru o trecere de pietoni, controlat de un microcontroler ATmega328P.

- **Ce face:** Sistemul gestioneaza traficul dintr-o intersectie, avand un semafor complet pentru masini (Rosu, Galben, Verde) si unul pentru pietoni (Rosu, Verde). Trecerea pietonilor este conditionata de actionarea unui buton. Sistemul dispune de o “memorie” a comenzii (daca butonul este apasat in timpul de garda al masinilor, cererea este inregistrata si executata imediat ce timpul expira). In plus, tranzitiile includ o faza de degajare a intersectiei (All-Red Phase), iar sistemul ofera feedback vizual (LCD I2C) si sonor (buzzer).
- **Care este scopul lui:** Scopul principal este simularea unei situatii reale de trafic urban, punand accent pe siguranta absoluta a pietonilor si fluidizarea circulatiei rutiere prin algoritmi de temporizare predictibili.
- **Care a fost ideea de la care ati pornit:** Am dorit sa transpun o problema clasica din viata de zi cu zi (asteptarea la semafor) intr-un automat de stari (FSM - Finite State Machine) complet non-blocant. Am vrut un proiect care sa combine elegant mai multe periferice (GPIO, Timere hardware, comunicare I2C) fara a folosi functii ineficiente de asteptare.
- **De ce credeti ca este util:** La nivel didactic, proiectul consolideaza programarea bare-metal (lucrul direct cu registri, vectori de intreruperi, manipulare la nivel de bit). La nivel practic, arhitectura poate fi extinsa pentru sisteme reale de asistenta a persoanelor cu deficiente de vedere.

Descriere generală

Arhitectura sistemului se bazează pe o unitate centrală de procesare (microcontrolerul de pe placa Xplained Mini) care comunică cu mai multe module de Input/Output:

- **Modulul de Input (Cerere traversare):** Un buton tactil configurat cu rezistența de pull-up internă a microcontrolerului. Este conectat pe pinul PD2 pentru a declanșa o **întrerupere hardware externă (INT0)** pe front descrescător. Astfel, comanda este preluată instantaneu, indiferent de starea buclei principale.
- **Modulul Output Vizual 1 (Semafoare):** Format din 5 LED-uri controlate prin pinii GPIO. Automatul de stări alternează între Trafic Auto Permis și Traversare Permisă, trecând prin faze intermediare critice: Galben pentru atenționare auto, clipirea intermitentă a LED-ului verde pietonal (ultimele 5 secunde) și o stare de degajare a intersecției în care absolut toate semafoarele sunt pe Roșu.
- **Modulul Output Vizual 2 (Interfață Text):** Un display LCD 1602 conectat prin interfața I2C (pinii SDA și SCL). Acesta este controlat printr-o bibliotecă custom pentru a afișa dinamic starea

intersecției (“Pietoni: VERDE”, “Degajare...”, “Cerere inregist.”) și un cronometru invers actualizat în timp real.
- **Modulul Output Sonor:** Un buzzer activ care emite un semnal de avertizare intermitent atunci când semaforul pietonal este verde.

Funcționarea de ansamblu este dirijată de **Timer-ul 1 Hardware pe 16 biți** configurat în modul CTC (Clear Timer on Compare Match). Acesta generează o întrerupere exact la 1 secundă (`OCR1A = 15625`, prescaler 1024), care dictează ritmul automatului de stări și actualizează ecranul, asigurând o execuție 100% non-blocantă a logicii principale.

Schema Bloc a Sistemului:

Hardware Design

Proiectul se afla in stadiul de prototip hardware asamblat fizic pe breadboard, toate componentele electronice fiind cablate si interfatate cu microcontrolerul.

Componentele folosite si rolul lor:

- **Placa ATmega328P Xplained Mini:** “Creierul” proiectului; proceseaza intrarile si controleaza iesirile.
- **5 x LED-uri (3 Auto: Rosu/Galben/Verde, 2 Pietoni: Rosu/Verde):** Simuleaza luminile intersectiei.
- **5 x Rezistoare (220 ohmi):** Protejeaza pinii placii si LED-urile prin limitarea curentului.
- **1 x Buton (Push-button):** Preia cererea de traversare a pietonului.
- **1 x Buzzer:** Asigura semnalizarea sonora a culorii verzi pentru pietoni.
- **1 x Ecran LCD 1602 cu modul I2C:** Ofera feedback vizual (starea intersectiei, contor de timp).
- **Breadboard si fire jumper:** Faciliteaza conexiunile electrice intre componente.

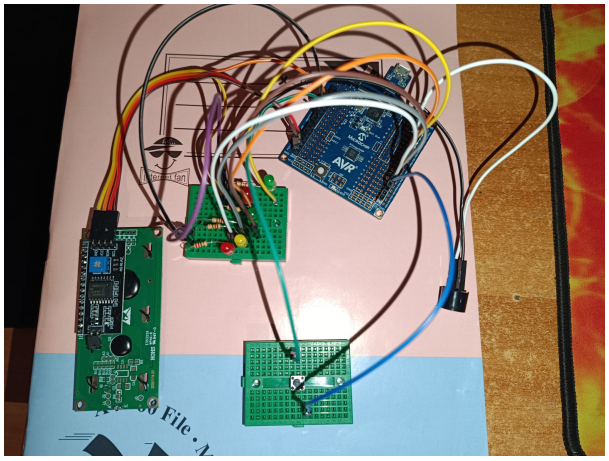
Detalierea pinilor:

- **PD2 (Buton - INPUT):** Ales obligatoriu pentru ca suporta intreruperi externe (INT0), permitand citirea instantanee a butonului fara blocarea programului.
- **PC4 si PC5 (LCD - OUTPUT):** Pinii hardware dedicati comunicarii I2C (PC4 = SDA/Date, PC5 = SCL/Clock).
- **PD3, PD4, PD5 (Semafor Auto - OUTPUT):** Grupati logic pe portul D pentru a controla masinile.
- **PB0, PB1 (Semafor Pietoni - OUTPUT) si PB2 (Buzzer):** Grupati pe portul B pentru a separa sectiunea pietonala si de avertizare sonora.

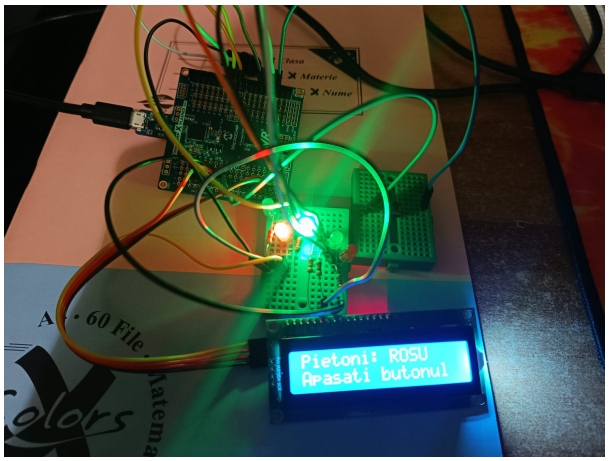
Schema electrica si explicatii:

Explicatia conexiunilor: Ecranul LCD primeste alimentare de la pinii de 5V si GND ai placii. LED-urile au anodul conectat la pinii digitali alocati (Porturile B si D), iar catodul este legat la masa (GND) prin rezistoare de 220 ohmi. Butonul foloseste rezistenta de pull-up interna a microcontrolerului, facand scurt la GND la apasare (pe pinul PD2), iar buzzer-ul este legat direct intre PB2 si masa.

Imagini cu componentele conectate si dovezi:



- **Imaginea 1:** Vedere de ansamblu a circuitului asamblat pe breadboard alaturi de placa Xplained Mini.



- **Imaginea 2:** Dovada functionarii hardware-ului: LCD-ul aprins si LED-urile alimentate, confirmand cablarea corecta.

Software Design

Stadiul actual al implementarii software Implementarea software este complet functionala. Codul ruleaza un automat de stari (FSM - Finite State Machine) stabil si predictibil, fiind capabil sa gestioneze tranzitiile semaforului, sa preia comenzile asincrone de la pietoni si sa afiseze timpul in timp real, fara a se bloca sau a rata instructiuni.

Motivarea alegerii bibliotecilor Pentru a reduce consumul de memorie si a maximiza performanta, proiectul utilizeaza exclusiv biblioteci minimale, evitand framework-urile greoaie:

- <avr/io.h> si <avr/interrupt.h>: Ofera acces direct si la nivel de bit asupra registrilor hardware (pini, timere, vectori de intrerupere).
- <stdio.h>: Utilizata strict pentru functia snprintf, necesara pentru formatarea dinamica a textului afisat pe LCD (combinarea textului fix cu variabilele contorului).
- “lcd_i2c.h”: O biblioteca custom, minimala, creata special pentru a pastra controlul absolut asupra protocolului TWI/I2C si a elimina overhead-ul librariilor mari din open-source.

- <util/delay.h>: Pastrata strict pentru functionarea ecranului LCD (controlerul HD44780 are nevoie de pauze microscopice intre instructiuni pentru a randa caracterele). Asteptarile sunt izolate doar in driverul ecranului, logica principala a semaforului ramanand 100% non-blocanta.

Elementul de noutate al proiectului Noutatea proiectului consta in implementarea unui sistem de semaforizare “cu memorie” si prioritate de siguranta (All-Red Phase). Fata de un semafor clasic care face doar polling pe buton, sistemul nostru retine comanda pietonului daca acesta a apasat butonul in timpul timpului de garda al masinilor. In plus, tranzitiile includ faze de degajare a intersectiei (toata lumea are rosu), reproducand un standard real de siguranta rutiera, iar driverul I2C a fost scris custom pe registri.

Justificarea utilizarii functionalitatilor din laborator

- GPIO (General Purpose I/O): Folosit pentru actionarea directa a celor 5 LED-uri si a buzzer-ului, manipuland bitii pe porturile B si D.
- Intreruperi Externe (INT0): Utilizate pe pinul PD2 pentru a capta apasarea butonului instantaneu, independent de stadiul de executie al buclei principale, garantand ca nicio apasare nu este ignorata.
- Timere (Timer1 in mod CTC): Elementul central de temporizare. Am folosit hardware-ul pentru a numara exact 1 secunda (OCR1A = 15625 la un prescaler de 1024), declansand o intrerupere ce scade contoarele. Asta a inlocuit complet functiile ineficiente _delay_ms().
- Comunicare TWI (I2C): Folosita pentru a comanda ecranul LCD folosind doar 2 pini (SDA/SCL), economisind pinii microcontrolerului pentru logica de semaforizare.

Scheletul proiectului, interactiunea si validarea

- Scheletul: Proiectul este structurat intr-o functie de initializare hardware (hardware_init()) si o bucla infinita while(1) care evalueaza o instructiune switch.
- Interactiunea: Intreruperea de timp (ISR Timer1) actualizeaza o variabila globala (state_timer). In bucla principala, automatul de stari verifica acest contor; cand ajunge la zero, modifica starea curenta (ex: din Masini_Verde in Tranzitie_Galben), actualizeaza iesirile (LED-urile) si reseteaza timer-ul.
- Validarea: A fost realizata modular. S-a testat intai clipirea LED-urilor cu delay, apoi s-a migrat la Timer1, apoi s-a validat preluarea corecta a butonului prin INT0, iar in final s-a adaugat interfata I2C. Fiecare stare a FSM-ului a fost monitorizata vizual pe LCD pentru a confirma timpii exacti.

Calibrarea elementelor de senzoristica / input Deoarece interfata de input este digitala, calibrarea “senzoristicii” s-a referit la ajustarea hardware a contrastului ecranului LCD (prin potentiometrul modulului I2C) pentru o vizibilitate optima. La nivel de semnal, s-a activat rezistenta de pull-up interna pe pinul butonului si s-a configurat intreruperea INT0 sa se declanseze doar pe front descrescator (Falling Edge), pentru a citi ferm momentul apasarii. Pentru magistrala I2C s-a realizat “calibrarea” frecventei (prescaler TWBR = 72) pentru a asigura o viteza stabila de 100kHz a ceasului SCL.

Optimizari

- Refresh conditionat al LCD-ului: A fost implementata logica prin care ecranul este sters si rescris doar daca starea sistemului sau contorul de secunde s-au schimbat. De ce: Functiile de curatare LCD consuma mult timp; scrierea lor la fiecare iteratie a buclei while(1) ar fi generat o palpaire (flicker) severa.
- Curatarea reziduurilor lasate de Bootloader: In functia de initializare am fortat TCCR1A = 0 si TCCR1B = 0. De ce: Bootloader-ul preinstalat al placii lasa timer-ele in mod PWM 8-bit. Fara aceasta

optimizare critica, Timer-ul 1 nu ar fi atins niciodata pragul necesar pentru calcularea secundelor, sistemul ramanand blocat.
- Arhitectura Non-Blocanta: Eliminarea functiilor de tip delay(). De ce: Pentru a permite sistemului sa memoreze instantaneu apasarea butonului chiar si atunci cand se afla in mijlocul unei perioade lungi de asteptare.

Rezultate Obtinute

Demo Video: YouTube - Demo Semafor Inteligent

In demonstratie se observa starea initiala (Verde pentru masini). La apasarea butonului, se declanseaza secventa de oprire trecand prin faza de degajare (Rosu pentru ambele sensuri), urmata de Verde pentru pietoni, acompaniat de semnalul sonor si de clipirea LED-ului in ultimele secunde, conform parametrilor programati.

S-a obtinut un prototip complet functional care respecta rigorile de siguranta ale unei intersectii reale. Sistemul demonstreaza o reactivitate instantanee la apasarea butonului datorita vectorului de intrerupere `INT0`. Din punct de vedere software, utilizarea Timer-ului 1 in modul CTC a eliminat complet necesitatea functiilor de tip delay din logica de control, iar scrierea unui driver I2C minimal, pe registri, a demonstrat o optimizare eficienta a resurselor microcontrolerului.

Concluzii

Proiectul a dovedit importanta planificarii arhitecturale a codului (folosirea unui Finite State Machine). Cea mai mare provocare, dar si reusita a proiectului, a fost sincronizarea fluida a trei evenimente distincte: schimbarea culorilor conform temporizarii, afisarea textului formatat dinamic pe LCD si preluarea asincrona a input-ului uman. Trecerea de la o paradigma blocanta (cu delay-uri) la una bazata pe intreruperi a ridicat semnificativ complexitatea, dar a rezultat intr-un sistem robust si stabil.

Download

- Cod sursa: GitHub Repository

Jurnal

- **Etapa 1:** Alegerea temei, achizitionarea componentelor hardware si realizarea conexiunilor pe breadboard.
- **Etapa 2:** Implementarea logicii de baza a semaforului folosind functii blocante (delay) si testarea circuitelor de LED-uri.
- **Etapa 3:** Refactorizarea codului intr-un automat de stari (FSM), implementarea Timer-ului 1 hardware (CTC mode) si conectarea intreruperii INT0 pentru citirea non-blocanta a butonului.

- **Etapa 4:** Dezvoltarea bibliotecii minimale I2C, integrarea ecranului LCD 1602 si calibrarea fazelor de siguranta (All-Red, Green Blink).

Bibliografie/Resurse

- Datasheet ATmega328P (Configurarea Timer1 si registrilor TWI)
- Laboratoarele de PM (GPIO, Timere, Intreruperi)
- Specificatiile controlerului HD44780 (pentru timpii de executie ai ecranului LCD)

Export to PDF