Smart Pocket Calculator cu Feedback Audio

Introducere

Proiectul constă într-un Calculator de Buzunar Smart cu Feedback Audio. Acesta rezolvă operații matematice de bază (+, -, *, /) și, spre deosebire de un calculator standard implementat pe breadboard, oferă o confirmare sonoră instantanee.

Scopul proiectului este de a crea un dispozitiv fiabil și interactiv, eliminând incertitudinea apăsărilor pe tastatură (fenomenul de ghosting sau taste neînregistrate). **Ideea de la care am pornit** a fost necesitatea de a avea certitudinea preluării corecte a input-ului fizic fără a fi nevoie să privim constant ecranul, inspirată de interfețele de la casele de marcat sau bancomate. **Utilitate:** Este un proiect extrem de util pentru consolidarea noțiunilor de debouncing, lucrul cu mașini de stări pentru parsarea input-ului și integrarea eficientă a mai multor periferice externe folosind protocoale diferite.

Descriere generală

Sistemul utilizează microcontrolerul ca unitate centrală de procesare și integrează 3 periferice externe (cerință de regulament), folosind conceptele a cel puțin 3 laboratoare (GPIO, Timere/PWM, I2C).

Sistemul este alcătuit din următoarele module:

- **Modul Input (Tastatură matricială 4×4):** Permite introducerea cifrelor și a operatorilor. Folosește pini GPIO pentru scanarea liniilor și coloanelor.
- **Modul Output Vizual (Ecran LCD 1602 cu I2C):** Afișează expresia introdusă și rezultatul. Utilizarea interfeței I2C minimizează numărul de conexiuni, crescând fiabilitatea hardware-ului.
- **Modul Output Audio (Buzzer Pasiv):** Generat folosind semnale PWM / Timere. Oferă un “beep” scurt la o apăsare corectă și un ton lung/diferit la o acțiune invalidă (ex. împărțire la zero).

Interacțiunea modulelor: La apăsarea unei taste pe matricea 4×4, microcontrolerul execută o rutină de debouncing. Dacă acțiunea este validată, se declanșează un semnal PWM către buzzer pentru feedback sonor scurt și se actualizează datele pe LCD prin magistrala I2C. Logica internă (o mașină de stări software) memorează primul număr, operatorul și al doilea număr, executând calculul la apăsarea tastei ”=”.

Hardware Design

Designul hardware a fost gândit pentru a fi simplu de implementat și robust, respectând cerința de a avea un cablaj curat și un aspect ordonat, fără fire redundante.

Listă de piese:

- 1 x Microcontroler
- 1 x Tastatură matricială 4×4
- 1 x Ecran LCD 1602
- 1 x Modul adaptor I2C pentru LCD (PCF8574)
- 1 x Buzzer pasiv
- Fire de conexiune, breadboard / placă de test

Conexiuni principale (Schemă bloc simplificată):

- **Tastatura 4×4:** Conectată la un port GPIO (ex. Portul C) pentru a citi starea butoanelor.
- **LCD I2C:** Conectat la pinii SDA și SCL ai microcontrolerului, plus alimentare (VCC, GND). Tehnologia I2C ne scapă de cablarea paralelă complexă de 8 fire.
- **Buzzer:** Conectat la un pin cu capabilitate de PWM (ex. pin de output de la Timer) pentru a genera independent tonurile audio.

https://ocw.cs.pub.ro/courses/_media/pm/prj2026/florin.stancu/schematicspirofinal.png?w=120&h=66&t=1778957577&tok=222ffb
https://ocw.cs.pub.ro/courses/_media/pm/prj2026/florin.stancu/img20260517182510.jpg?w=90&h=120&t=1779034121&tok=0a469c

Software Design

Descrierea Firmware-ului Mediu de dezvoltare Firmware-ul este scris în C++ și compilat cu toolchain-ul avr-g++ prin intermediul PlatformIO (extensie Visual Studio Code). Platforma este atmelavr, board-ul uno, framework-ul arduino. Flag-urile de compilare sunt -Os (optimizare pentru dimensiune) și -Wall (toate avertismentele). Codul sursă se află într-un singur fișier: src/main.cpp.

Librării externe Wire.h — driver I2C (TWI), inclus în Arduino Core, folosit pentru comunicarea cu LCD-ul. LiquidCrystal_I2C.h (Marco Schwartz, v1.1.4) — control LCD 1602 prin backpack-ul PCF8574 la adresa I2C 0x27. Librăria Keypad.h nu a fost utilizată. Scanarea matricei 4×4 este implementată manual prin GPIO, pentru a demonstra înțelegerea multiplexării la nivel hardware. De asemenea, funcția tone() nu a fost utilizată — generarea sunetului se face prin configurarea directă a registrelor Timer1.

Descrierea logicii Firmware-ul este organizat în jurul unei mașini de stări finite (FSM) cu 5 stări: introducere operand 1, selectare operator, introducere operand 2, afișare rezultat și eroare.

Tastatura 4×4 este scanată manual: pinii de rând (D2–D5) sunt configurați ca OUTPUT, iar pinii de coloană (D6–D9) ca INPUT_PULLUP. La fiecare iterație a buclei principale, un rând este coborât la LOW și se citesc coloanele. O coloană LOW indică o tastă apăsată. Debounce-ul se face prin comparație cu millis() — o apăsare este acceptată doar dacă au trecut cel puțin 200 ms de la precedenta.

Fiecare tastă validă este rutată prin funcția processKey(), care folosește un switch pe starea curentă a FSM-ului. Cifrele sunt acumulate într-un buffer de tip string (maxim 8 cifre per operand). Operatorii (+, −, ×, ÷) declanșează tranziția către starea de așteptare a celui de-al doilea operand. Tasta = lansează calculul, iar tasta C resetează FSM-ul din orice stare.

Calculul se face în virgulă mobilă (float) pentru a suporta rezultate fracționare (ex: 7 ÷ 2 = 3.5). Împărțirea la zero este detectată explicit înainte de execuție și produce o tranziție în starea de eroare, cu mesaj pe LCD și beep de avertizare.

Buzzerul pasiv (D10) este controlat prin Timer1 în modul CTC 12 (TOP = ICR1, prescaler 8). Pinul OC1B este comutat automat la fiecare compare-match, generând o undă dreptunghiulară cu duty cycle 50%. Frecvența se obține din formula TOP = 1 000 000 / f − 1. Un beep scurt la 2000 Hz confirmă fiecare apăsare validă; un beep lung la 500 Hz semnalează o acțiune invalidă. Durata tonului este gestionată non-blocking — buzzerUpdate() verifică la fiecare iterație loop() dacă a expirat.

LCD-ul I2C afișează expresia curentă pe prima linie și un hint contextual sau rezultatul pe a doua linie. Rezultatele întregi sunt afișate fără zecimale, iar cele fracționare cu 4 zecimale.

Firmware-ul nu apelează delay() niciodată. Toată logica temporală folosește millis(), astfel încât microcontroller-ul rămâne responsiv în permanență.

Module software principale Scanare tastatură 4×4 (GPIO manual + debounce) Mașină de stări finite (FSM) cu 5 stări Motor aritmetic (adunare, scădere, înmulțire, împărțire, detecție div/0) Generare ton PWM prin registre Timer1 (CTC mode 12) Control LCD 1602 prin I2C (PCF8574) Monitorizare serială (debug la 9600 baud) smartcalculator.zip

Rezultate Obţinute

Care au fost rezultatele obţinute în urma realizării proiectului vostru.

Concluzii

Download

O arhivă (sau mai multe dacă este cazul) cu fişierele obţinute în urma realizării proiectului: surse, scheme, etc. Un fişier README, un ChangeLog, un script de compilare şi copiere automată pe uC crează întotdeauna o impresie bună .

Fişierele se încarcă pe wiki folosind facilitatea **Add Images or other files**. Namespace-ul în care se încarcă fişierele este de tipul **:pm:prj20??:c?** sau **:pm:prj20??:c?:nume_student** (dacă este cazul). **Exemplu:** Dumitru Alin, 331CC → **:pm:prj2009:cc:dumitru_alin**.

Jurnal

Puteți avea și o secțiune de jurnal în care să poată urmări asistentul de proiect progresul proiectului.

Bibliografie/Resurse

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

Export to PDF