Health Kit

Introducere

Ideea acestui proiect a luat naștere dintr-o realitate pe care o trăim cu toții: deși spunem frecvent că sănătatea este prioritară, ritmul alert al vieții ne face să neglijăm inconștient semnalele corpului nostru. Lipsa timpului este principalul obstacol care ne desparte de un control de rutină.

Proiectul propune o soluție sub forma unui kit de monitorizare rapidă, conceput să ofere date esențiale despre starea de sănătate într-un mod eficient și la îndemână. Scopul este transformarea întregului proces costisitor, într-o sarcină simplă, rezultatele obținute din urma consultului, determinând daca e necesar sau nu un control la un specialist.

Acest kit ajută persoanele cu un program încărcat, care doresc să dețină control asupra propriei sănătăți, oferind certitudinea datelor acolo unde, până acum, existau doar presupuneri legate de oboseală sau stres.

Descriere generală

Sistemul asigură monitorizarea și stocarea datelor prin următoarele funcționalități:

Afișaj LCD: Vizualizarea instantanee a pulsului și a nivelului de oxigen.

Diagramă EKG: Generarea unei reprezentări grafice a ritmului cardiac pentru o analiză vizuală precisă.

Sistem de alertă: Un buzzer integrat care anunță sonor utilizatorul în cazul depășirii pragurilor critice, sau dacă rezultele sunt în parametrii normali

Jurnal de date: Salvarea automată a măsurătorilor pe un card microSD, împreună cu marcaje temporale, pentru a păstra un istoric medical complet.

Hardware Design

Descrierea modulelor si interactiunea hardware-software:

Placa de dezvoltare (ATmega328P): Creierul întregului sistem. Coordonează citirea senzorilor, procesează semnalele analogice și digitale, gestionează mașina de stări a buzzerului și trimite datele către ecran și interfața serială.

Modul senzor ritm cardiac (MAX30100): Masoară continuu valorile brute de reflexie a luminii în spectrul Infraroșu (IR) și Roșu (RED) pentru determinarea pulsului și a oxigenării sângelui (SpO2) prin magistrala I2C.

Modul senzor EKG (AD8232): Condiționează și amplifică semnalele bio-potențiale cardiace preluate prin electrozi, livrând o ieșire analogică curată, gata de a fi eșantionată de ADC.

Ecran LCD 1602 cu modul I2C (PCF8574): Afișează în timp real starea sistemului („EKG Pornit…”, „Evaluare Puls..”, „Puls OK!”) și oferă feedback vizual direct utilizatorului, fără a bloca pinii microcontrolerului.

Modul cititor card MicroSD: Permite stocarea locală persistentă (data logging) a eșantioanelor culese de la senzori pentru analize ulterioare.

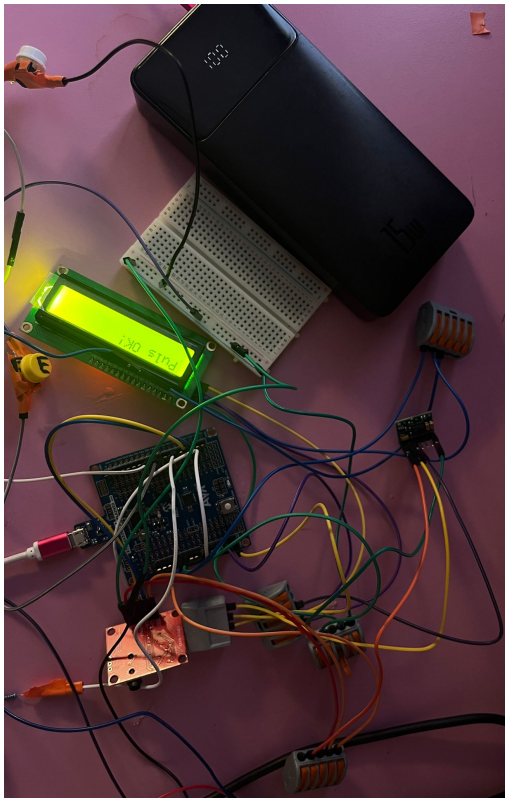
Buzzer + Tranzistor NPN: Generează alerte acustice diferențiate (melodie de succes sau sunete de alarmă pentru tahicardie/hipoxie) folosind semnale PWM de frecvență variabilă.

Butoane (2 bucăți): Interfața fizică cu utilizatorul; un buton declanșează înregistrarea eșantioanelor EKG timp de 5 secunde, iar celălalt pornește fereastra de evaluare a pulsului timp de 10 secunde.

Tabel de conectare a pinilor:

Componenta	Pini ATmega328P	Rol / Functionalitate
Modul senzor MAX30100	PC4 (SDA), PC5 (SCL)	Pinii hardware I2C/TWI pentru citirea datelor brute de puls si SpO2
Ecran LCD 1602 (I2C)	PC4 (SDA), PC5 (SCL)	Pinii hardware I2C/TWI partajati pentru afisarea starii si a mesajelor in timp real
Modul senzor EKG AD8232	PC0 (ADC0), PD2 (LO+), PD3 (LO-)	Intrare analogica pentru semnalul cardiac si intrari digitale pentru detectia detasarii electrozilor
Modul MicroSD (SPI)	PB2 (SS), PB3 (MOSI), PB4 (MISO), PB5 (SCK)	Pinii hardware SPI pentru stocarea si logarea datelor pe cardul SD
Buzzer (via tranzistor NPN)	PD5 (OC0B)	Iesire Fast PWM (Timer0) pentru generarea alertelor sonore si a melodiilor
Buton EKG	PB0	Intrare digitala cu Pull-up intern si intrerupere (PCINT0) pentru declansarea inregistrarii
Buton Puls	PB1	Intrare digitala cu Pull-up intern si intrerupere (PCINT1) pentru pornirea ferestrei de evaluare

Schema electrica:



Software Design

Mediu de dezvoltare

PlatformIO IDE (VS Code)

Motivarea alegerii bibliotecilor folosite

Petit FatFs (pff.h/.c):S-a ales această bibliotecă modulară de la ChaN deoarece este concepută special pentru microcontrolere cu resurse extrem de limitate de memorie RAM. Spre deosebire de FatFs standard, Petit FatFs utilizează un singur buffer global mic și nu alocă memorie dinamic, lăsând restul memoriei RAM a AVR-ului liberă pentru procesarea semnalelor de la senzori.

Driverul pentru MAX30100 (max30100.h/.c): Permite configurarea directă la nivel de registru a intensității LED-urilor (roșu și infraroșu) și a frecvenței de eșantionare hardware (50 Hz), asigurând o preluare curată a datelor fără a încărca procesorul.

Elementul de noutate al proiectului

Elementul de noutate constă în integrarea unui sistem de monitorizare medical dual (analogic + digital) cu feedback multisenzorial asincron. Spre deosebire de monitoarele medicale clasice care folosesc alarme sonore liniare și repetitive, acest dispozitiv folosește feedback acustic contextual gamificat (melodii Mario distincte pentru: finalizare cu succes, parametri optimi sau stări de alertă medicală).

Mai mult, designul software permite rularea simultană a redării audio, scrierii pe SD și afișării pe LCD fără ca vreuna să blocheze eșantionarea critică a senzorilor.

Justificarea utilizării funcționalităților din laborator

Laboratorul de I2C: Utilizat ca magistrală comună pentru a controla ecranul LCD și senzorul MAX30100 pe doar 2 fire (SDA/SCL), salvând pinii microcontrolerului.

Laboratorul de SPI: Folosit pentru comunicația de mare viteză cu modulul de card SD.

Laboratorul de ADC: Utilizat pentru digitizarea semnalului analogic provenit de la modulul de EKG

Laboratorul de PWM(Timer 0 și Timer 1): Timer 1 este configurat în mod CTC pentru a genera o întrerupere strictă la fiecare 1 ms, oferind baza de timp nemblocantă pentru funcția get_millis(). Timer 0 este configurat în mod Fast PWM pe pinul PD5 (OC0B). Prin modificarea dinamică a registrului OCR0A și a registrului OCR0B (care modifică factorul de umplere/volumul), am reușit să redau asincron melodiile din Super Mario direct din hardware

Laboratorul de USART: Folosit pentru trimiterea datelor în timp real către interfața grafică Teleplot de pe PC, facilitând depanarea.

Laboratorul de întreruperi: pentru cele 2 butoane care declanșează evenimentele (întrerupere externă **INT0**)

Structura proiectului, interacțiunea și validarea

Arhitectura codului este una ierarhică, structurată pe straturi distincte pentru a asigura modularitatea:

1. **Hardware Abstraction Layer:** interacționează direct cu regiștrii microcontrolerului (twi.c, spi.c, adc.c, timers.c, usart.c,)
2. **Drivere Componente:** gestionează logica componentelor periferice externe (lcd.c, max30100.c, sd.c, pff.c)
3. **Logica de Aplicație:** un automat de stări corelat direct cu interacțiunea fizică a utilizatorului

Fragment din main.c - Structura automatului de stări (LCD & Logica Generală):

switch (current_lcd_state) {

```
  case LCD_STATE_START:
      if (state_changed) {
```

```
            lcd_clear();
            lcd_set_cursor(0, 0);
            lcd_print("Alege modulul:");
            lcd_set_cursor(1, 0);
            lcd_print("E-EKG | P-Puls");
            state_changed = false;
        }
        break;
```

```
    case LCD_STATE_EKG_RUNNING:
        // Execută eșantionarea ADC și scrierea pe SD timp de 5 secunde
        if (current_ms - recording_start_time > EKG_DURATION_MS) {
            opreste_inregistrare(&offset_ekg);
            count_ekg++;
            current_lcd_state = LCD_STATE_EKG_DONE;
            state_changed = true;
        }
        break;
```

```
}
```

Mecanismul de interacțiune prin butoane

Tranziția dintre stări este complet asincronă, fiind declanșată de întreruperile externe de tip schimbare de pin *(PCINT0)* pe pinii *PB0 și PB1.* Când sistemul se află în starea inițială *(LCD_STATE_START)*, utilizatorul alege modul de funcționare prin apăsarea unui buton fizic.

Apăsarea butonului:

- **de pe PB0:** generează o întrerupere care comută automat sistemul în starea de preluare a datelor pentru EKG (LCD_STATE_EKG_RUNNING) și pornește scrierea în fișierul ekg.csv.
- **de pe PB1:** comută sistemul în starea de *monitorizare a pulsului* (LCD_STATE_PULSE_WAIT), pornind fereastra de eșantionare a senzorului optic.

```
ISR(PCINT0_vect) {
```

```
    button_event = 1;
```

```
}
```

```
void init_button_interrupts(void) {
```

```
    DDRB &= ~((1 << PB0) | (1 << PB1)); // Setare pini ca intrare
    PORTB |= (1 << PB0) | (1 << PB1);   // Activare rezistențe pull-up interne
    PCICR |= (1 << PCIE0);              // Activare întreruperi pentru grupul
PCINT0
    PCMSK0 |= (1 << PCINT0) | (1 << PCINT1); // Mascare pini specifici
```

}

Un *algoritm de debounce software* de 40 ms rulează în interiorul buclei pentru a elimina zgomotele electrice la apăsare, asigurând o comutare curată a stărilor.

if (debounce_active && (current_ms - debounce_start >= 40)) {

```
  uint8_t pin_state = PINB;
  uint8_t btn_ekg = (pin_state & (1 << PB0)) ? 1 : 0;

  if (current_lcd_state == LCD_STATE_START && btn_ekg == 0) {
      current_lcd_state = LCD_STATE_EKG_RUNNING;
      incepe_inregistrare("ekg.csv", offset_ekg);
      recording_start_time = current_ms;
      state_changed = true;
  }
  debounce_active = false;
```

}

Interacțiunea generală din bucla principală se bazează pe *cooperare non-blocantă*:

- **run_sensor_task()** funcție care actualizează datele senzorului optic la fiecare 20 ms
- **run_lcd_state_machine()** funcție care *gestionează ecranele* și *scrie pe cardul SD* doar la finalizarea ferestrei de eșantionare
- **funcția update_buzzer()** funcție care este apelată la fiecare iterație pentru a schimba dinamic starea pinului audio în mod asincron, generând melodiile fără a îngheța execuția codului

Modul de validare

- **TWI/I2C:** s-a realizat prin rularea rutinei *twi_discover()* la inițializare. Aceasta scanează întreaga magistrală și confirmă prin terminalul serial USART prezența adaptorului LCD la adresa 0x27 și a senzorului MAX30100 la adresa 0x57.
- **stocarea**: S-a confirmat prin mutarea fizică a cardului SD într-un calculator după efectuarea testelor. S-a verificat integritatea structurală a fișierului ekg.csv și a fișierului puls.txt (care reține istoricul ultimelor 4 măsurători de puls sub formă de text aliniat).
- **asincronismului** S-a verificat experimental, atât auditiv, cât și vizual. În momentul în care sistemul scrie un sector întreg pe cardul SD, melodia redată în fundal de către buzzer nu agață, nu se întrerupe și nu produce sacadări, demonstrând eficiența programării asincrone bazate pe timere

Calibrarea elementelor de senzoristica

Senzorul Ekg: rutina de citire verifică în permanență *pinii hardware de alertă* (“Leads Off”). Dacă un electrod se dezlipește de pe corp, modulul de condiționare a semnalului simte asta imediat și trimite un mesaj text de avertizare pe interfața serială către calculator

Senzorul MAX30100:

Pentru a funcționa corect, senzorul a fost reglat pe două părți: hardware și software.

- **La nivel hardware**, am setat curentul prin ambele LED-uri la valoarea de 17.4 mA. Acest nivel este perfect pentru a trece prin deget fără probleme, dar fără să fie atât de puternic încât să „orbească” fotodioda (senzorul de lumină).
- **La nivel software,** semnalul brut primit de la senzor conține mult zgomot de fond din cauza luminii din cameră și a țesutului fix al degetului. Pentru a curăța datele, am aplicat un filtru software care scade constant această componentă de curent continuu (DC). Semnalul util rămas (componenta AC) reprezintă doar variația sângelui din vene și este trimis mai departe către algoritmul care detectează bătăile inimii.
- **Stabilizarea rezultatelor prin mediere**: Deoarece bătăile inimii pot fi neregulate sau degetul se mai poate mișca ușor, calculul brut al ritmului cardiac ar fluctua haotic pe ecran. Pentru a rezolva asta, codul folosește un sistem de **mediere glisantă.** Sistemul numără bătăile într-o fereastră scurtă de aproximativ 1.3 secunde pentru a afla un puls instantaneu, după care salvează ultimele 6 valori obținute într-un vector (bpm_samples). Valoarea finală trimisă către ecran și cardul SD este o medie matematică a acestor eșantioane stocate. Această metodă netezește fluctuațiile bruște și oferă un afișaj stabil și real al BPM-ului.

Optimizări realizate

Optimizarea Scrierii Consecutive pe SD (opreste_inregistrare din main.c)

- Petit FatFs are limitarea nativă de a rotunji în jos pointerul de scriere la începutul sectorului curent la apeluri repetate de deschidere/închidere fișier, cauzând suprascrierea datelor anterioare
- am implementat *alinierea forțată* a offset-ului la următorul sector de 512 bytes: **offset = (fs_global.fptr + 511) & 0xFFFFFE00**. Acest lucru forțează scrierile consecutive să sară peste restul sectorului umplut cu spații de către disk_writep, permițând salvarea tuturor măsurătorilor succesive.

void opreste_inregistrare(DWORD* offset) {

```
  WORD bytes_written;
  pf_write(0, 0, &bytes_written);
  if (offset != NULL) {
      // rotunjire offset-ul la urmatorul multiplu de 512 bytes (sector
hardware)
      *offset = (fs_global.fptr + 511) & 0xFFFFFE00;
  }
```

}

Managementul erorilor I2C

- Protocolul I2C se poate bloca din cauza zgomotului parazit generat de firele lungi sau de buzzer
- Toate buclele *while din twi.c* au fost dotate cu un *contor de timeout (TWI_TIMEOUT_COUNT)*. Dacă magistrala se blochează, se ridică flag-ul **twi_error_flag,** iar funcția *handle_twi_recovery()* din loop-ul principal reinițializează automat perifericele hardware și senzorii, asigurând reziliența totală

a dispozitivului

void twi_start(void) {

```
  TWCR = (1 << TWINT) | (1 << TWEN) | (1 << TWSTA);
  uint16_t timeout = TWI_TIMEOUT_COUNT;
  while (!(TWCR & (1 << TWINT)) && --timeout);
  if (timeout == 0) twi_error_flag = 1;
```

}

Trecerea dinamică a vitezei SPI:

- Cardurile SD nu pot fi inițializate la frecvențe mari, dar odată pornite, viteza mică încetinește execuția programului
- Inițializarea se face la *fosc/16*. Imediat ce cardul confirmă starea activă în disk_initialize(), registrele *SPCR și SPSR* sunt modificate din mers pentru a comuta SPI-ul pe viteza maximă suportată (fosc/2 sau fosc/4), scurtând timpul blocant de scriere pe disk

if (ty) {

```
  SPCR &= ~(1 << SPR0);
  SPSR |= (1 << SPI2X);
```

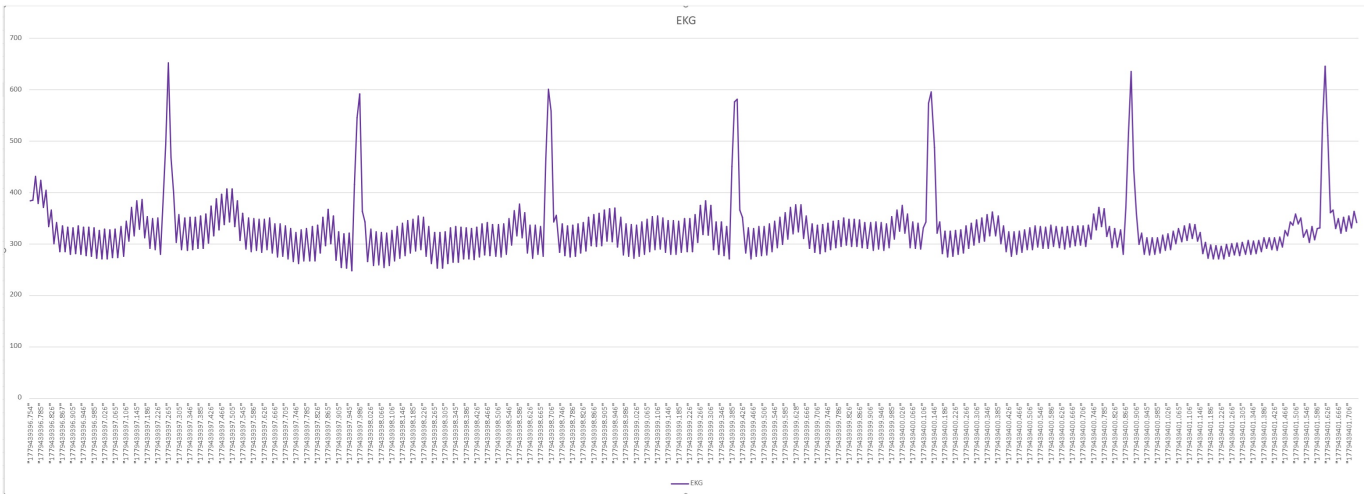
}

Rezultate Obţinute

Monitorizarea activității electrice cardiace (Modul EKG)

Pentru modulul EKG, sistemul oferă o *flexibilitate dublă* în vizualizarea și analiza rezultatelor, extrem de utilă în scenariile medicale reale:

1. **Vizualizare în timp real (Live Streaming):** Datele citite de ADC-ul intern sunt transmise instant prin USART către utilitarul Teleplot de pe PC. Acest mod permite urmărirea dinamică a undei cardiace direct pe ecran în timpul măsurătorii, facilitând o diagnosticare rapidă.
2. **Stocare și analiză ulterioară (Offline Post-Processing):** Simultan, aceleași perechi de date (timp_ms, valoare_adc) sunt salvate pe cardul SD în fișierul *ekg.csv.*



Graficul de mai sus reprezintă waveform-ul extras prin importarea fișierului ekg.csv în Excel. Pe axa orizontală avem timpul exprimat în milisecunde, iar pe axa verticală amplitudinea digitizată a semnalului cardiac.

```
1   0,612
2   56,556
3   66,645
4   76,253
5   86,568
6   96,233
7   106,564
8   116,239
9   126,555
10  136,243
11  146,556
12  156,252
13  166,560
14  176,249
15  186,574
16  196,255
17  206,574
18  216,254
19  226,588
20  236,266
21  246,587
22  256,269
23  266,592
24  276,272
25  286,595
26  296,263
27  306,581
28  316,225
29  326,550
30  336,213
```

Mai sus este un fragment din fișierul ekg.csv (format timp_ms, valoare_adc)

Monitorizarea indicilor vitali (Modul Puls)

Sesiunea de testare a modulului optic (MAX30100) s-a validat prin analiza fișierului text generat consecutiv pe cardul SD. În urma a 4 determinări succesive, structura datelor salvate în puls.txt arată astfel:

```
BPM: 0 | SpO2: 0% | ALERTA
----------

BPM: 67 | SpO2: 90% | OK
----------

BPM: 90 | SpO2: 90% | OK
----------

BPM: 67 | SpO2: 91% | OK
----------
```

Se observă că sistemul stochează corect și consecutiv toate înregistrările, datorită algoritmului de aliniere forțată la sectorul de 512 bytes implementat la oprirea înregistrării. Prima înregistrare simulează semnalarea aferentă datelor care nu sunt în pragurile stabilite, pe când următoarele trei arată metrici arată măsurătorile unui pacient stabil

Link demo cu funcționalitatea

https://www.youtube.com/watch?v=m5Z1DsHCiDg

Download

Codul sursă al proiectului este disponibil în repository-ul GitHub:

- https://github.com/Adzinaa/medkit_PM

Bibliografie/Resurse

https://www.analog.com/media/en/technical-documentation/data-sheets/max30100.pdf

https://www.analog.com/media/en/technical-documentation/data-sheets/ad8232.pdf

https://github.com/robsoncouto/arduino-songs/blob/master/supermariobros/supermariobros.ino