UNIVERSITATEA TEHNICĂ „Gheorghe Asachi” din IAȘI
FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DOMENIUL: Calculatoare și Tehnologia Informației
SPECIALIZAREA: Calculatoare Încorporate

SISTEM DE MONITORIZARE ȘI CONTROL MULTI-NOD
INPUT 2: POTENȚIOMETRU

Student:
Constantin-Alexandru ARHIP
Daniel BUDU
Cornelia-Ștefana HRISCU

Iași, 2026

Cuprins

Capitolul 1. Introducere

1.1. Descriere

Acest proiect implementează un sistem de control distribuit utilizând protocolul de comunicație industrială **CAN (Controller Area Network)**. Obiectivul principal este transferul de date de la nodurile de intrare (senzori) către nodurile de execuție (actuatori), sub supravegherea unui nod dedicat de monitorizare. Întregul sistem este construit pe platforma **NXP FRDM-MCXN947**.

1.1.1. Arhitectura Sistemului și Distribuția Echipelor

Sistemul este compus din 5 noduri, fiecare fiind gestionat de o echipă specifică:

1. **Potențiometru (Input):** achiziția analogică (ADC) și transmiterea poziției către rețea pentru controlul unghiular sau de intensitate.
2. **Senzor de Temperatura/Umiditatem (Input):** monitorizarea condițiilor de mediu și partajarea acestor parametri pe magistrală.
3. **Monitor:** interceptarea pachetelor. Acesta nu modifică datele, ci analizează traficul pentru validarea comunicării între celelalte noduri.
4. **Servomotor (Output):** Interpretează datele primite (de la potențiometru si senzor) și generează semnalul PWM necesar poziționării mecanice.
5. **LED RGB (Output):** Oferă feedback vizual pe baza datelor de la senzori.

1.1.2. Schema Logică a Transmisiei

Figura 1.1 ilustrează modul în care datele circulă de la cele două surse de intrare către restul componentelor sistemului.

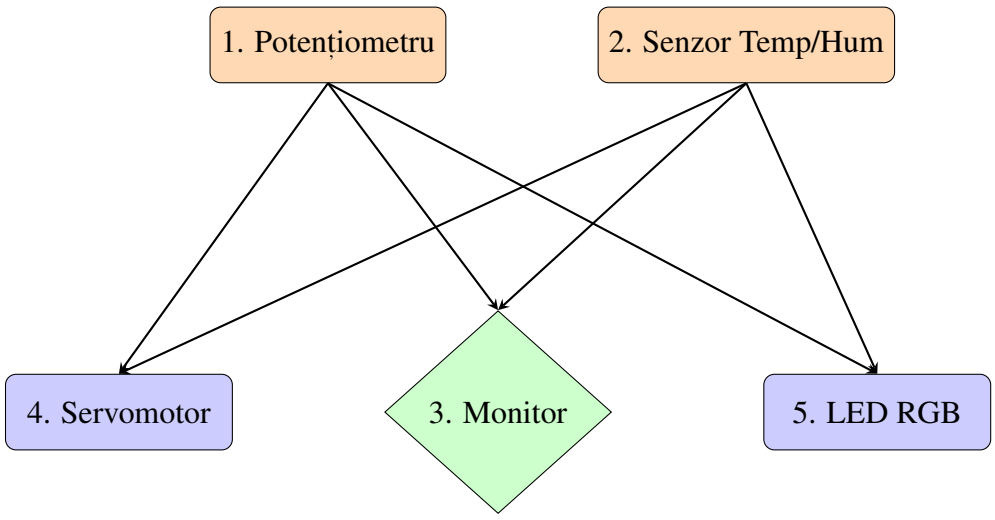

Figura 1.1. Schema de distribuție

1.2. Placa de dezvoltare

Placa de dezvoltare **FRDM-MCXN947** este o platformă compactă și scalabilă, concepută pentru evaluarea microcontrolerelor din seria **MCX N94x**. Aceasta este ideală pentru aplicații de tip Edge Computing, automatizări industriale și Internet of Things (IoT).

1.3. Arhitectură Microcontroler (MCU)

Nucleul plăcii este reprezentat de MCU-ul **MCX N947**, care dispune de următoarele caracteristici:

- **Dual-Core:** 2x Arm® Cortex®-M33 rulând la frecvențe de până la 150 MHz.
- **Accelerator AI/ML:** Include unitatea de procesare neurală **eIQ® Neutron NPU** pentru sarcini de inteligență artificială la nivel hardware.
- **Securitate:** Subsistem de securitate **EdgeLock® Secure Enclave** (Core Profile).
- **Memorie:** Până la 2 MB de memorie Flash și 512 KB de memorie SRAM cu paritate/ECC.

1.4. Caracteristici Hardware și Periferice

Placa FRDM-MCXN947 integrează o gamă largă de componente pentru prototipare rapidă:

- **Conectivitate Ethernet:** Port RJ45 integrat pentru aplicații de rețea.
- **Interfețe USB:** Port USB Type-C High-Speed pentru date și alimentare.
- **Expansiune:** Compatibilitate cu shield-uri **Arduino Uno R3** și conectori **MikroBus™** pentru senzori și actuatori.
- **Debug Integrat:** On-board **MCU-Link** debugger bazat pe CMSIS-DAP, eliminând necesitatea unui programator extern.
- **Interfață Audio:** Suport pentru ieșiri audio de tip MQS (Medium Quality Sound).

1.5. Ecosistem de Dezvoltare

Suportul software este asigurat prin **MCUXpresso Developer Experience**, oferind:

1. **IDE:** MCUXpresso IDE, VS Code (cu extensii NXP) sau IAR/Keil.
2. **SDK:** Drivere periferice optimizate, stive de protocoale și exemple de cod (RTC, PWM, Ethernet).
3. **Configurare:** Instrumente grafice pentru rutarea pinilor și configurarea ceasurilor (Pins, Clocks, Peripherals Tool).

1.6. Implementare Tehnică

Pentru realizarea comunicării, s-au utilizat următoarele resurse hardware ale microcontrolerului MCX N947:

- **FlexCAN:** Configurarea baud-rate-ului și a filtrelor de acceptare (Hardware Acceptance Filters).
- **GPIO & PWM:** Pentru controlul direct al servomotorului și al LED-ului.
- **ADC:** Pentru conversia semnalului analogic de la potențiometru.

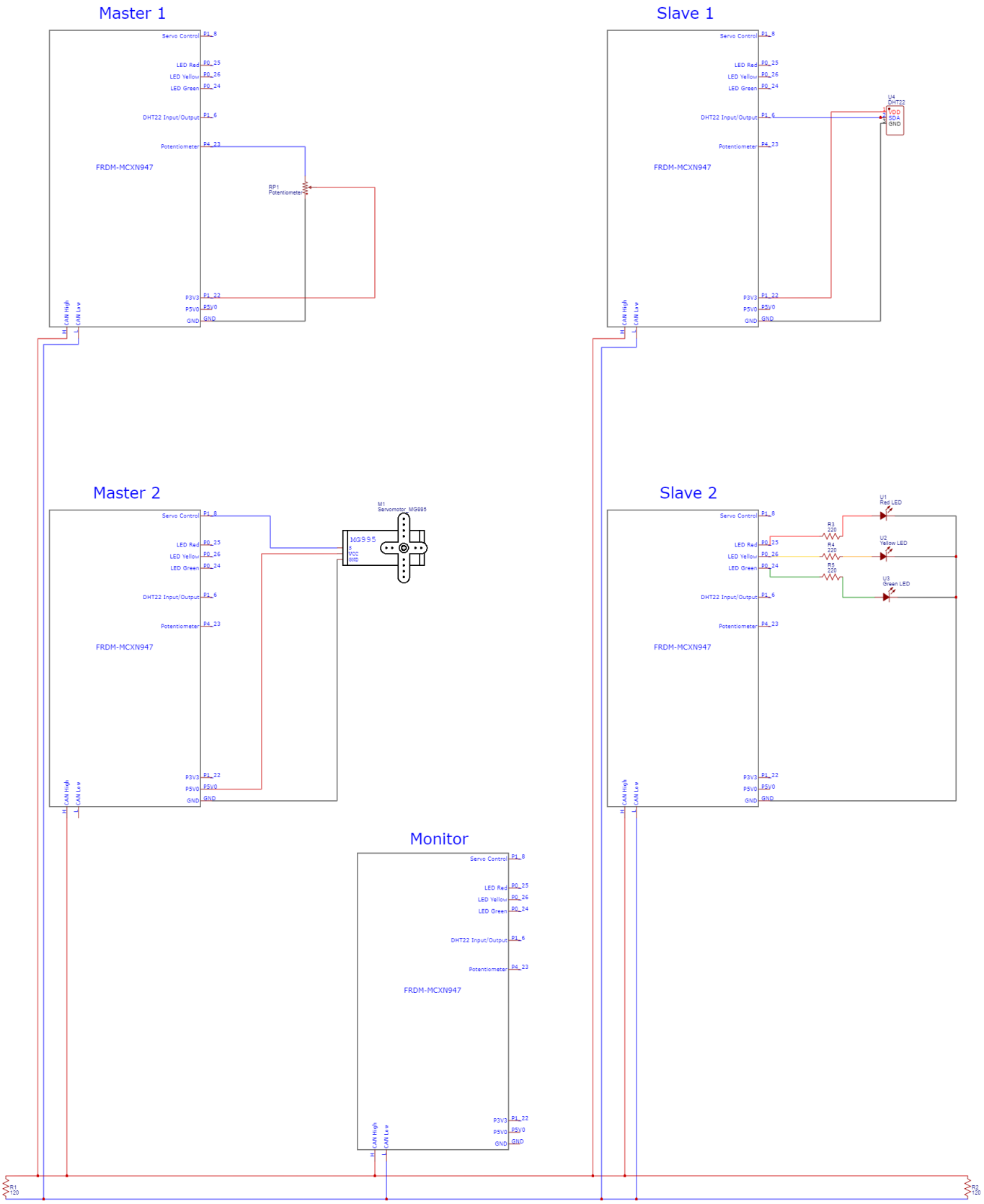

Figura 1.2. Schema Arhitecturii Sistemului

Capitolul 2. Implementarea Controlului pentru Potențiometru și Integrarea CAN

2.1. Arhitectura Hardware și Interfațarea

2.1.1. Descrierea Componentei: Potențiometrul

Potențiometrul utilizat în cadrul acestui proiect este un senzor analogic rezistiv de tip rotativ, configurat ca un divizor de tensiune variabil. Dispozitivul este alimentat la o tensiune stabilizată de $3.3V$ furnizată de placa de dezvoltare, iar cursorul central (wiper) este conectat la pinul de intrare analogică **P4_23** (canalul ADC corespunzător). Această configurație permite transformarea mișcării mecanice într-o variație de tensiune liniară, interpretată ulterior de unitatea de conversie analog-digitală (LPADC).

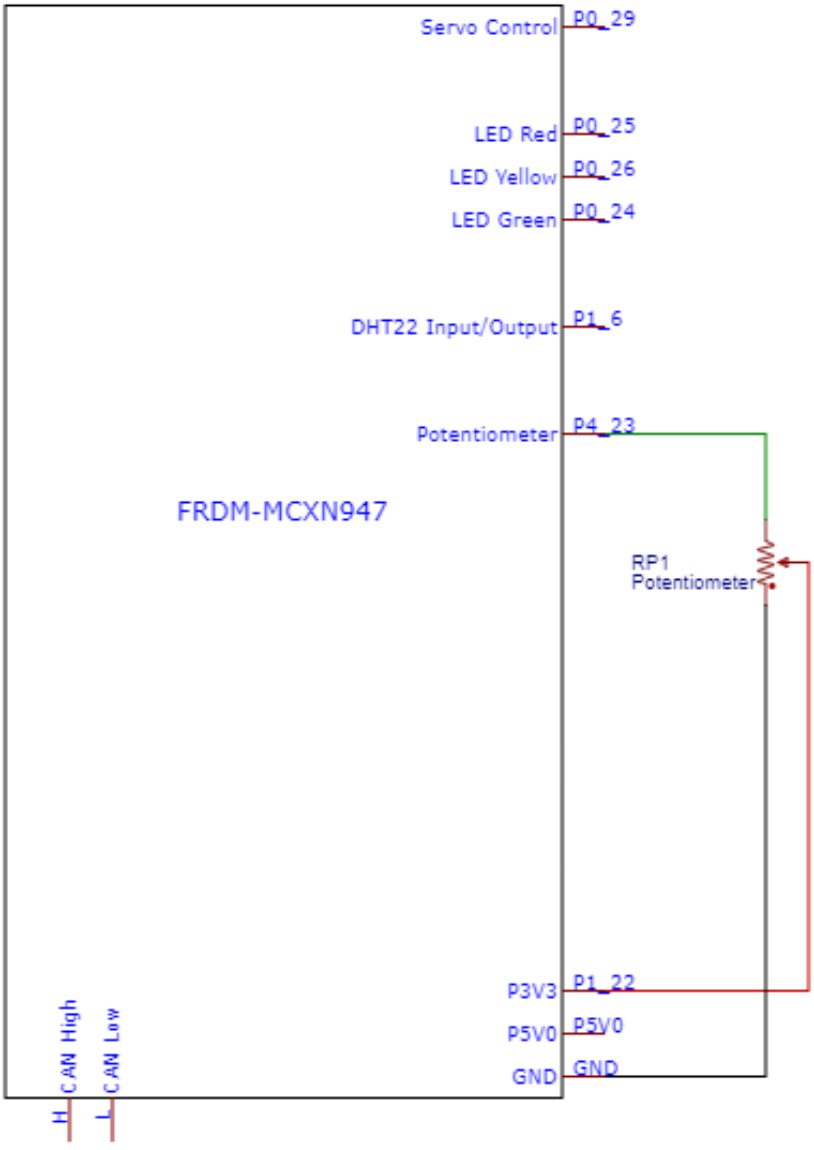

Figura 2.1. Schema Arhitecturii Potențiometrului și conexiunea la MCXN947

2.1.2. Configurarea Perifericului LPADC (Low-Power ADC)

Pentru a asigura o achiziție de date de înaltă rezoluție cu un consum energetic redus, perifericul LPADC a fost configurat utilizând structurile de tip `lpadc_config_t`. Parametrii critici setați sunt:

- **Power Level:** Bitul `PWRSEL` din registrul de configurare este setat pe `kLPADC_PowerLevelAlt4`, permițând o frecvență de eșantionare ridicată și stabilitate în regim tranzitoriu.
- **Referință și Rezoluție:** S-a optat pentru utilizarea referinței interne `VREFH`, eliminând fluctuațiile cauzate de zgomotul de pe linia de alimentare. Rezoluția este fixată la **12 biți** (`kLPADC_ConversionResolutionHigh`), rezultând într-un interval digital între 0 și 4095.
- **Calibrare Automată:** Procedura `LPADC_DoAutoCalibration` este apelată la fiecare pornire a sistemului pentru a compensa derivele termice și erorile de offset ale circuitului intern de eșantionare-menținere (*Sample-and-Hold*).
- **Hardware Average:** S-a activat acumularea hardware a 8 eșantioane per conversie pentru a reduce zgomotul de înaltă frecvență (*jitter*) fără a încărca procesorul.

2.2. Arhitectura Software și Gestiunea Evenimentelor

2.2.1. Sincronizarea prin SCTimer și Mașina de Stări

Implementarea se bazează pe o arhitectură orientată pe evenimente (*event-driven*). Se utilizează perifericul **SCTimer** (State Configurable Timer) configurat ca un numărător pe 32 de biți. Acesta este programat să genereze o întrerupere periodică (*Match Interrupt*) la un interval de 2000*ms*.

Această abordare este superioară funcțiilor `delay()` clasice deoarece:

1. Permite intrarea procesorului în stări de consum redus (*Sleep Mode*) între citiri.
2. Garantează un timp de eșantionare determinist, esențial pentru stabilitatea controlului în rețeaua CAN.
3. Handler-ul de întrerupere `SCT0_IRQHandler` execută doar setarea unui flag (`g_triggerSensorRead`), menținând latența sistemului la un nivel minim.

Mașina de stări a aplicației (`app_state_t`) gestionează fluxul logic prin următoarele tranziții:

- **STATE_LISTEN:** Nodul se află în stare pasivă, procesând cadrele primite de la alte noduri (ex: mesaje de sincronizare sau interogări).
- **STATE_SEND:** Activată de flag-ul timer-ului sau de un eveniment extern (buton), această stare preia datele din buffer și apelează driverul FlexCAN.
- **STATE_WAIT:** Stare de „gardă” (back-off) de 500*ms* utilizată pentru a preveni saturarea magistralei în cazul unor transmisii repetate și pentru a permite arbitrarea prioritară pentru alte noduri.

2.2.2. Sistemul de Buffer FIFO și Prioritizarea Mesajelor

Deoarece viteza de transmisie pe magistrala CAN poate varia în funcție de trafic, s-a implementat o coadă circulară (FIFO) care acționează ca un strat de abstractizare între producerea datelor și trimiterea lor fizică.

- **Evenimente Analogi:** Valoarea potențiometrului este introdusă în FIFO cu ID-ul `TX_MSG_ID_TIMER`.

- **Evenimente Digitale:** Apăsarea butoanelor **SW2** și **SW3** declanșează întreruperi de tip GPIO care introduc instantaneu în FIFO valoarea brută eșantionată care este mapată în procente. Acest lucru demonstrează natura multi-sursă a sistemului.

2.3. Procesarea Datelor și Transmisia CAN-FD

2.3.1. Algoritmul de Scalare și Histerezis Software

Pentru a evita fenomenul de *chatter* (transmisii inutile cauzate de oscilația ultimului bit ADC), am implementat un algoritm de filtrare bazat pe pași de granularitate de 5%. Valoarea brută eșantionată este mai întâi mapată în procente folosind constanta `g_LpadcFullRange`.

Calculul filtrării este descris de formula:

$$P_{filtered} = \left\lfloor \frac{P_{raw} + 2.5}{5} \right\rfloor \cdot 5 \tag{2.1}$$

Această logică asigură că actuatorii (LED-urile sau servomotoarele din rețea) primesc comenzi stabile, prelungind durata de viață a componentelor mecanice.

2.3.2. Integrarea CAN-FD (Flexible Data-rate)

Protocolul utilizat este CAN-FD, configurat cu o rată de transfer de date de **2 Mbps** și o rată nominală (arbitrare) de **500 kbps**.

- **Non-Blocking I/O:** Transmisia se realizează prin `FLEXCAN_TransferFDSendNonBlocking`, permițând codului să continue procesarea în timp ce perifericul se ocupă de serializarea biților pe cablu.
- **Callback Function:** La finalizarea transmisiei, funcția de tip callback (`flexcan_callback`) confirmă succesul operațiunii prin setarea flag-ului `txComplete`.

2.3.3. Mecanisme de Reziliență și Handshake

Sistemul a fost proiectat pentru a funcționa într-un mediu nesigur.

Funcția `FLEXCAN_PHY_Config_Safe` este critică: la pornire, nodul încearcă să trimită un mesaj de „bună ziua” (*handshake*). Dacă nu există un alt nod care să trimită semnalul de confirmare (**ACK**), software-ul nu rămâne blocat într-o buclă infinită datorită unui contor de tip *timeout*.

Dacă `timeout` ajunge la zero, funcția `FLEXCAN_TransferFDAbortSend` este apelată pentru a curăța registrele transmițătorului și a permite sistemului să funcționeze în regim local, marcând eroarea de comunicare în log-urile de sistem. Această abordare garantează robustețea necesară aplicațiilor de tip industrial.

Capitolul 3. Concluzii și Direcții de Dezvoltare

3.1. Concluzii

Proiectul a demonstrat succesul implementării unui sistem distribuit de monitorizare și control bazat pe arhitectura **FRDM-MCXN947** și protocolul de comunicare **CAN**. Integrarea nodului de input pentru potențiometru a evidențiat capacitățile avansate ale perifericului **LPADC**, oferind o precizie ridicată și o stabilitate a datelor prin utilizarea algoritmilor de filtrare software.

Principalele realizări ale proiectului includ:

- **Eficiența Conversiei:** Utilizarea rezoluției de 12 biți și a funcțiilor de auto-calibrare a permis obținerea unor date analogice fidele, eliminând erorile de offset hardware.
- **Stabilitatea Sistemului:** Implementarea filtrului de tip „snapping” la intervale de 5% a redus semnificativ jitter-ul pe magistrala CAN, asigurând o mișcare fluidă a servomotorului și un control stabil al culorilor LED-ului RGB.
- **Robustezul Comunicării:** Protocolul CAN a asigurat schimbul de date între cele 5 noduri fără pierderi de informație, demonstrând viabilitatea sistemului într-un context de automatizare industrială.

În concluzie, utilizarea ecosistemului NXP MCX a facilitat dezvoltarea rapidă a unei rețele de senzori și actuatori, oferind un echilibru optim între consumul redus de energie și puterea de procesare necesară.

3.2. Direcții de Dezvoltare Viitoare

Deși sistemul actual este funcțional și stabil, acesta poate fi extins și optimizat prin următoarele direcții:

3.2.1. Optimizarea Consumului de Energie

O dezvoltare ulterioară ar putea presupune utilizarea modurilor de *Deep Sleep* ale microcontrolerului MCXN947. Nodul potențiometru ar putea fi configurat să „trezească” sistemul doar atunci când detectează o schimbare a tensiunii analogice peste un anumit prag (*Wake-on-ADC*), reducând astfel consumul mediu.

3.2.2. Implementarea unei Interfețe Grafice (HMI)

Adăugarea unui ecran tactil (TFT/OLED) pe nodul de monitorizare ar permite vizualizarea grafică a evoluției temperaturii (DHT22) sau a poziției potențiometrului în timp real, oferind o experiență de utilizare superioară.

3.2.3. Extinderea Magistralei CAN

Sistemul poate fi extins prin adăugarea de noi noduri, cum ar fi senzori de proximitate sau module de comunicație wireless (Wi-Fi/Bluetooth) pentru a crea o punte între rețeaua CAN locală și o platformă de tip **IoT Cloud**.

3.2.4. Algoritmi de Control Avansați

Pentru nodul servomotor, se poate implementa un control de tip **PID** (Proporțional-Integral-Derivativ). Acesta ar permite o poziționare mult mai precisă și o eliminare completă a oscilațiilor la schimbările bruște de setpoint trimise de potențiometru.

Bibliografie

[1] NXP Semiconductors, *FRDM-MCXN947 Board User Guide*, Document Number: FRDMMCXN947UG, disponibil la adresa oficială NXP.

[2] NXP Semiconductors, *MCX N Series Microcontrollers Reference Manual*, rev. 1, 2024, oferind detalii despre perifericele LPADC, FlexCAN și SCTimer utilizate în proiect.

[3] NXP Semiconductors, *MCUXpresso SDK API Reference Manual*, capitolul *LPADC Driver* și *FlexCAN Driver*, pentru implementarea funcțiilor de calibrare și transfer non-blocant.

[4] International Organization for Standardization (ISO), *ISO 11898-1:2015 Road vehicles — Controller area network (CAN) — Part 1: Data link layer and physical signalling*, utilizat pentru standardizarea protocolului CAN-FD.

[5] NXP Semiconductors, *Application Note: Configuring LPADC for High Precision Measurements on MCX Series*, oferind baze pentru implementarea algoritmului de eșantionare și filtrare digitală.