Documentatie pentru Proiect Informatica Aplicata:
Osciloscop bazat pe Raspberry PICO 2-W (RP2350)

Ionescu George-Stefan, specializarea IETTI Anul II, Grupa 4LF642, Facultatea de Inginerie Electrica si Stiinta Calculatoarelor, Universitatea Transilvania din Brasov.

1. Obiectivul proiectului

Proiectul isi propune ca obiectiv crearea unei unelte ce ajuta la **vizualizarea semnalelor electrice** în funcție de timp, un osciloscop. Osciloscopul propus poate afisa semnale cu frecventa pana la 50kHz teoretic conform teoremei Nyquist-Shannon, insa scopul folosirii lui este in domeniul audio-frecventelor, adica pentru a vizualiza semnale cu frecventa pana in jur de 20kHz. Osciloscopul afișează tensiunea unui semnal pe axa verticală (Y) și timpul pe axa orizontală (X), permițând observarea formei de undă.

Proiectul propus este facut cu ajutorul unei placate RP2350 ce colecteaza datele si le transmite catre un laptop ce ruleaza un program de Python pentru vizualizarea datelor. De asemenea a mai fost folosit un RP2350 ca si generator de semnal cu frecventa reglabila.

2. Achizitia de date

Proiectul utilizeaza un convertor analog-digital (ADC) de 12 biti intern al placutei RP2350, care suporta tensiuni intre 0V si 3.3V. Acest ADC este capabil sa esantioneze date cu o viteza de pana la 500kS/s(kHz), insa in scopul vizualizarii semnalelor audio, a fost configurat pentru esantionare continua cu o frecventa de 100kS/s(kHz). Aceasta decizie contribuie la un alt factor important, anume transmisia de date insa vom acoperi efectul frecventei de esantionare asupra acesteia la momentul potrivit.

Datele sunt transferate în memorie prin intermediul controlerului DMA (direct memory access), reducând încărcarea procesorului și permițând o achiziție stabilă a semnalelor în timp real.

Datele colectate sunt tensiuni cu valori in 0 Volti si 3.3 Volti. Aceste tensiuni sunt masurate de ADC si cuantizate in format digital in valori de 12 biti (0-4095). Mai departe in logica microcontrollerului, ADC-ul umple un buffer (un vector) FIFO (first in first out, queue) din care controllerul DMA transfera datele in memorie intr-un vector care urmeaza sa fie transmis catre laptop prin USB CDC (Communications Device Class).

3. Interpretarea datelor

Workflow pentru interpretarea datelor:

1. Colectarea datelor din portul COM USB pe care este conectat modului Raspberry cu ajutorul librarieri pyserial. Se foloseste functia ser = serial.Serial(PORT, BAUDRATE, timeout=1) pentru a obtine instanta de serial care citeste din portul dorit. Mai apoi se foloseste functia ser.read(SIZE) a obiectului serial, care citeste SIZE numar de octeti.
2. Deoarece protocolul USB CDC transmite date in pachete de cate 64 de octeti, datele sunt trasmite cu un header de 2 octeti si de 62 de octeti sunt datele de ADC, adica 31 de esantioane de 16biti (62 octeti / 2 octeti = 31 esantioane). Datele sunt, deci, “culese” cate 31 de esantioane si stocate intr-un buffer cu ajutorul functiei read_992_samples(), scrisa de mine, in care citesc 992 de esantioane, 992 este 31*32, practice citesc 32 de pachete de date de la USB CDC.
3. Datele sunt afisate apoi intr-o fereastra cu ajutorul librariilor matplotlib si tinker, prima fiind pentru plotting pe axe si a doua fiind o librarie pentru toolkit-ul GUI de baza pentru Python.
4. Pentru FFT folosesc functiile de FFT incluse in libraria numpy.

Am facut de asemenea un feature pentru a opri semnalul la un anumit moment cee ace permite vizualizarea unei capture a semnalului, similar cu butonul start/stop de pe un osciloscop classic.

5. Cablaj

M-am folosit de un alt RP2350 pentru a genera un semnal PWM pentru a simula un semnal audio. Codul pentru acesta este pe GitHub, este destul de simplu deci nu voi prezenta in mare detaliu ce face. Generatorul de PWM permite controlul frecventei intre 100Hz si 20kHz printr-un potentiometru, de asemenea in acelasi mod se poate regla duty cycle-ul prin alt potentiometru.

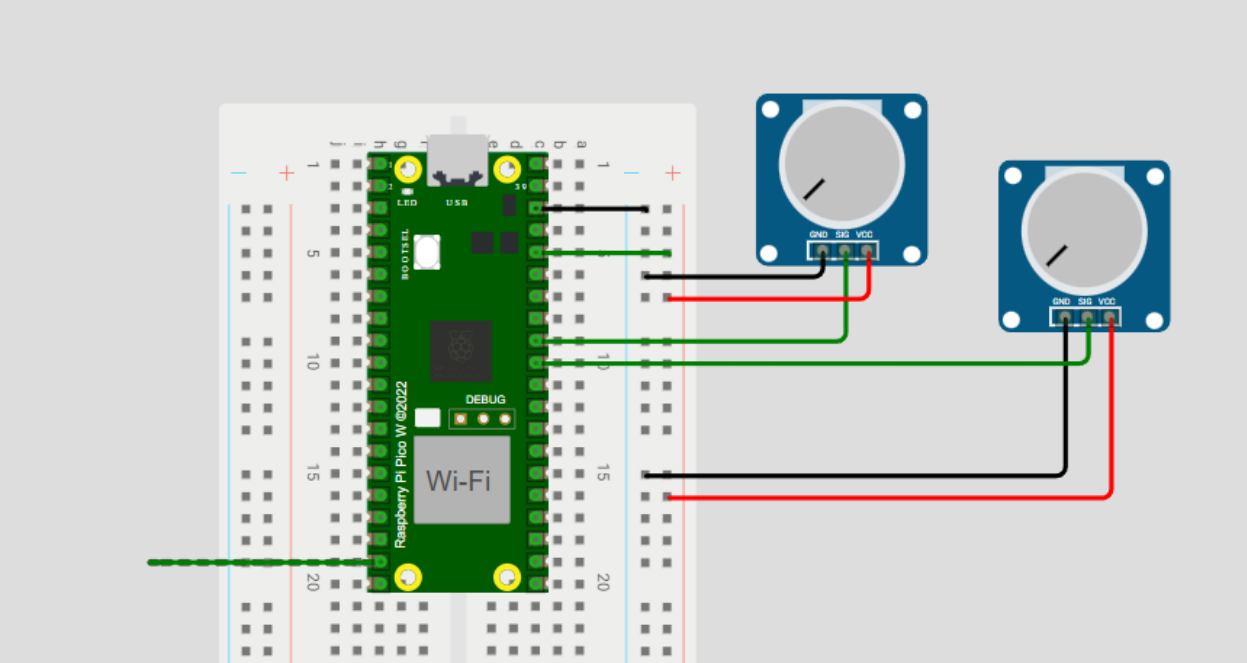

Fig.1 Cablajul pentru generatorul de PWM.

Cablajul pentru placuta care achizitiona date a fost mai simplu. La acesta doar am conectat un fir intre pin-ul GP26 (ADC0 IN) al placutei ce achizitiona date la pin-ul GP14 al placutei generatoare, pe care l-am setat ca iesire pentru semnalul PWM. De asemenea amandoua placutele aveau un fir care le conecta la masa comuna de pe rail-ul albastru al breadboard-ului (GND-GND).

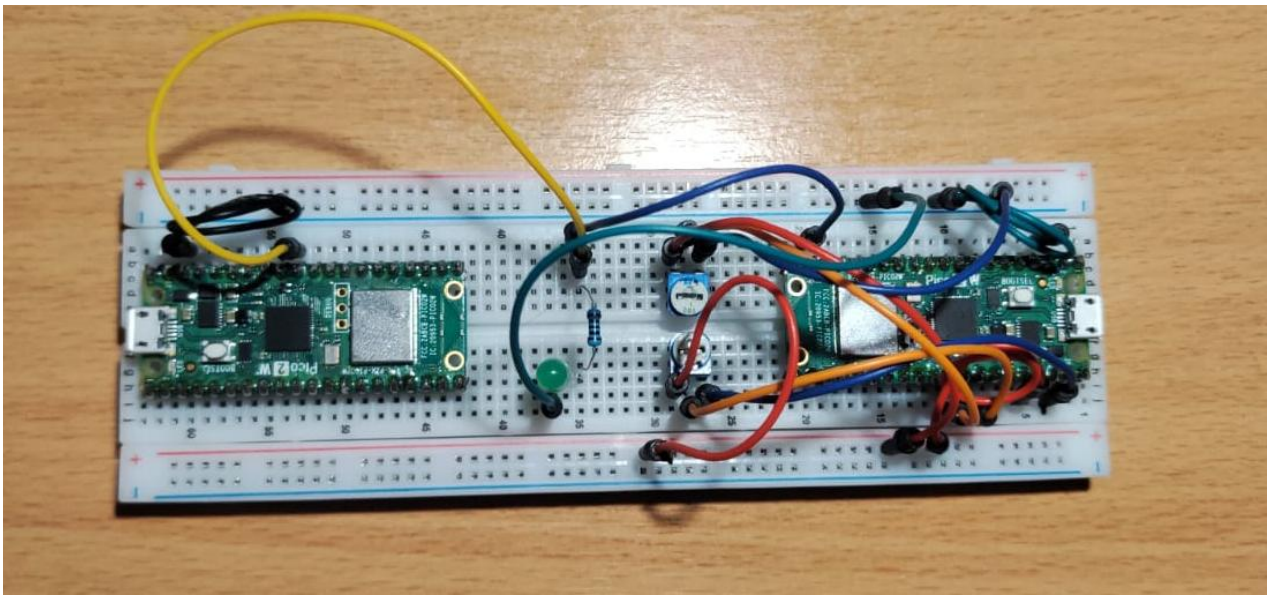

Fig. 2 Cablajul final (STANGA – PICO achizitie date; DREAPTA – PICO generator PWM).

Amandoua placutele erau conectate prin USB la laptop, cea de achizitie trimitand datele prin USB CDC si cea generatoare de PWM trimitand prin serial cu printf() parametrii activi ai semnalului PWM (duty, freq, wrap, set point, divider, adc1, adc2).

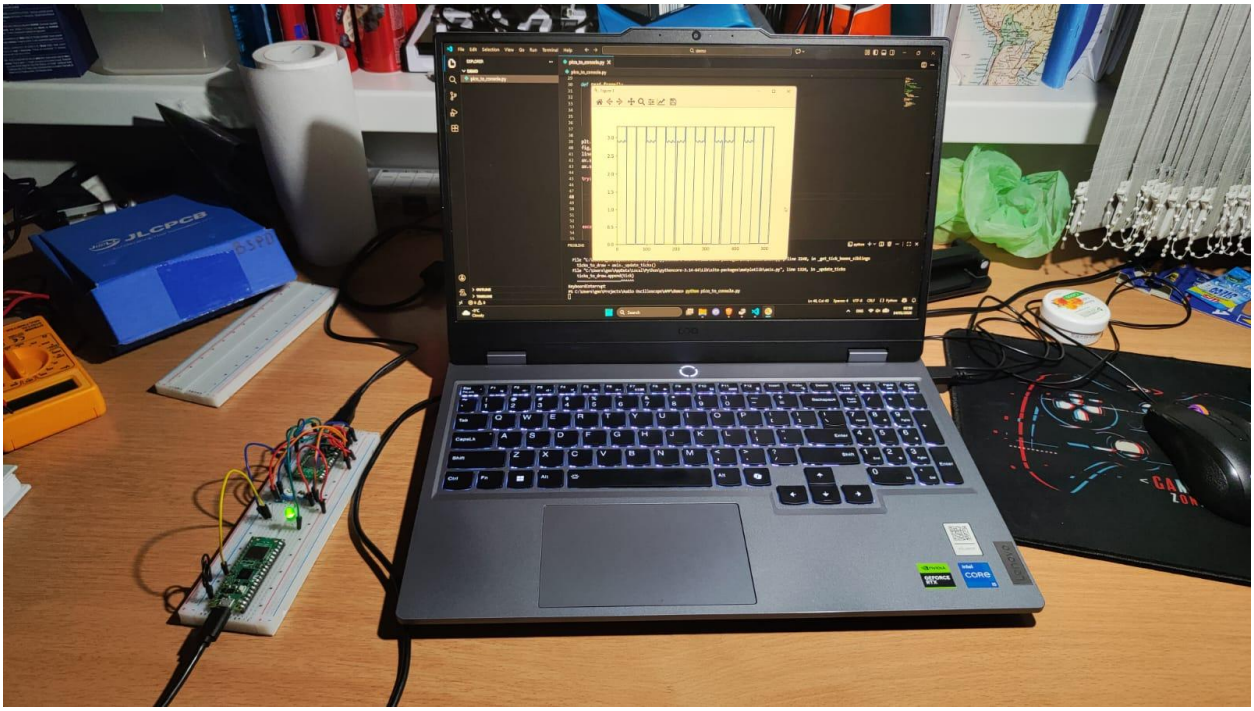

Fig. 3 Setup complet. Ecranul arata initial un semnal eronat datorita transmiterii USB CDC, ulterior reparat.

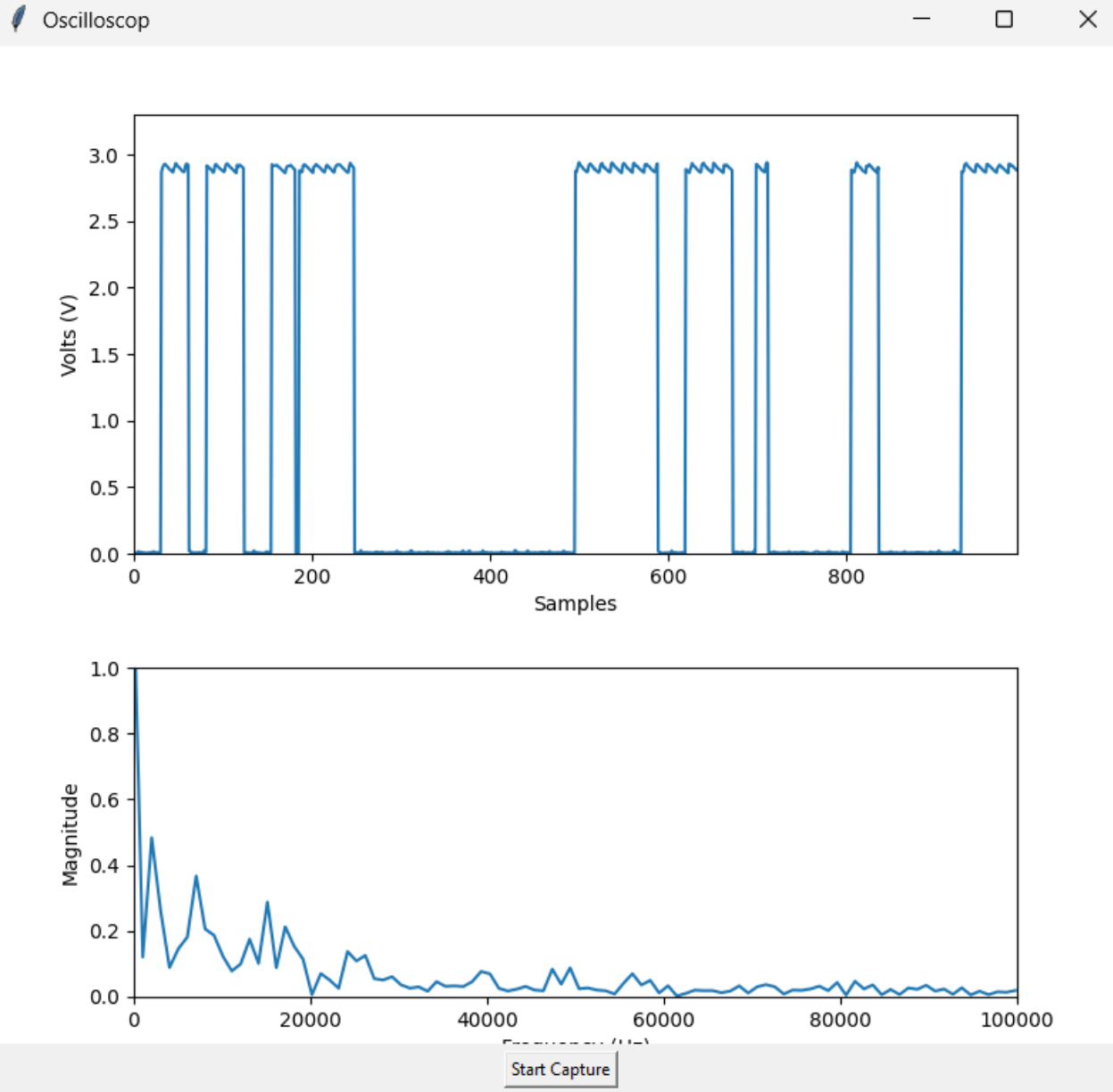

Fig. 4 Demonstratie cu un semnal PWM @ 200Hz.

6. Improvements/Revizii

As vrea sa rezolv problemele ce tin de sincronizarea transmisiei de date/afisare, deoarece semnalul nu pare sa fie esantionat continuu, ci pare ca ar fi receptionat si afisat “pe bucati”. De asemenea adaugarea unei joje ce modifica scala de timp pentru vizualizare pe o perioada mai mica ar fi utila in cazul frecventelor mai inalte.

7. Cod

Este urcat pe GitHub pe repository-ul https://github.com/i-onescu/Audio-Oscilloscope. Codul de PICO este scris in C si codul de desktop in python.