Beliciu Andrei – 421 F

Fratu-Halunga Theodor-Corneliu – 421 F

Cod echipă – F01

**PCLP3 - Proiect Python**

**Generarea de semnale cu ajutorul microcontroller-ului ESP32**

1. **Introducere**

În cadrul acestui proiect, s-a dorit realizarea vizualizării unui semnal cu ajutorul microcontroller-ului ESP32. Programul este gândit să afișeze pe un osciloscop și să furnizeze unui buzzer piezoelectric pasiv mai multe semnale generate folosind biblioteca Numpy pe un calculator, apoi introduse în codul Python care rulează pe ESP32. Schimbarea semnalului afișat se face ciclic prin apăsarea unui buton.

Pentru a reuși acest lucru, mai întâi s-a generat un semnal pe calculator cu ajutorul programului signal\_list\_generator/main.py și a bibliotecilor „numpy” și „matplotlib.pyplot” . Pentru a genera un semnal este nevoie de a stabili care sunt parametrii acestuia, apoi crearea unui vector de timp, generarea semnalului și vizualizarea acestuia. De exemplu, pentru generarea unui semnal sinusoidal, primul pas este de a atribui valori frecvenței și a intervalului de eșantionare care este dependent de programul care rulează pe ESP32 și care prin măsurători s-a dedus a fi 44us, apoi crearea vectorului de timp folosind instrucțiunea „np.arange(0, T, Ts)”. După acest pas cu ajutorul instrucțiunii „x\_sin = 0.5 \* np.sin(2 \* np.pi \* f \* t) + 0.5 ” din funcția „ def sinusoidal\_wave(T, Ts)" s-a obținut generarea și in cele din urmă s-a vizualizat cu „mplot.plot(t, x\_sin)” și „mplot.show()”.

Pentru a continua, în programul signal\_list\_generator/main.py s-a creat o metodă „def esp\_list(self) ” care afișează în consolă o listă/array care conține valorile eșantioanelor semnalului și numărul acestora. Aceste date sunt exportate în ESP, care folosește MicroPython, și se rulează programul scris în Python pentru microcontroller. După accea se măsoară semnalul transmis de pe pinul cu numărul 25, cu ajutorul unui osciloscop și se mai adaugă un buzzer pentru a putea fi observată comportarea audio a acestuia.

1. **Programele și echipamentele necesare**
2. O placă ESP32 (în cazul de față am folosit un ESP32-WROOM DevKitC V4)
3. Instalarea Python3
4. Instalarea unui IDE care suportă limbajul Python (Visual Studio Code, PyCharm, etc.)
5. Instalarea modulelor NumPy și Matplotlib.Pyplot într-un virtual environment.
6. Instalarea unui IDE care suportă utilizarea MicroPython cu un ESP32 (uPyCraft sau Thonny IDE, în cazul de față am folosit uPyCraft)
7. Firmware-ul binar pentru ESP32, de pe site-ul <https://micropython.org/download/ESP32_GENERIC/> sau /Proiect\_MicroPython/ESP32\_GENERIC-IDF3-20210202-v1.14.bin din repozitorul GitHub
8. Un cablu de USB-MicroUSB cu suport transfer date
9. Un osciloscop
10. Un modul piezo-buzzer pasiv (MH-FMD în cazul de față), un buton, un rezistor de 1k și cabluri de conexiune
11. **Pași urmați pentru utilizarea programului**
12. Se descarcă și instalează Python3
13. Se descarcă și se rulează uPyCraft (în repozitorul de pe GitHub /Proiect\_MicroPython/uPyCraft\_V1.1.exe)
14. Se descarcă fișierul Firmware binar pentru ESP32 (în repozitorul GitHub /Proiect\_MicroPython/ESP32\_GENERIC-IDF3-20210202-v1.14.bin)
15. Se conectează ESP32 la calculator și se determină portul COM la care este conectat (în Windows, în Device Manager > Ports(COM & LPT))
16. În uPyCraft, se selectează portul găsit în meniul de sus, Tools > Serial > COM…

A screenshot of a computer

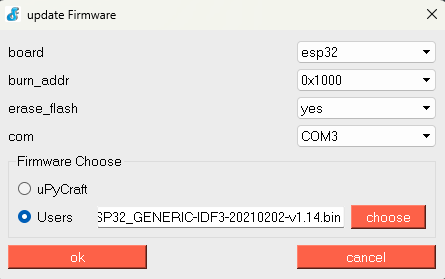
Description automatically generated

1. Se selectează placa ESP32 în meniul Tools > board > esp32

A screenshot of a computer

Description automatically generated

1. Se instalează bootloader-ul de MicroPython pe ESP32 folosind meniul Tools > BurnFirmware, și selectând în fereastra nou apărută placa esp32, adresa de scriere (0x1000), ștergerea memoriei flash setată ”yes”, portul COM al plăcii folosit anterior, iar în secțiunea Firmware Choose se folosește opțiunea Users și se caută fișierul Firmware binar descărcat anterior



1. Se realizează pe o placă de test montajul de mai jos. Modulul de buzzer pasiv MH-FMD se alimentează la 5V și la masă, iar portul de I/O se conectează la portul 25 al ESP32. Portul 35 al ESP32 se conectează printr-un rezistor de pull-down la masă și la buton, care se mai leagă la portul de 3,3V al ESP32.

A computer screen shot of a circuit board

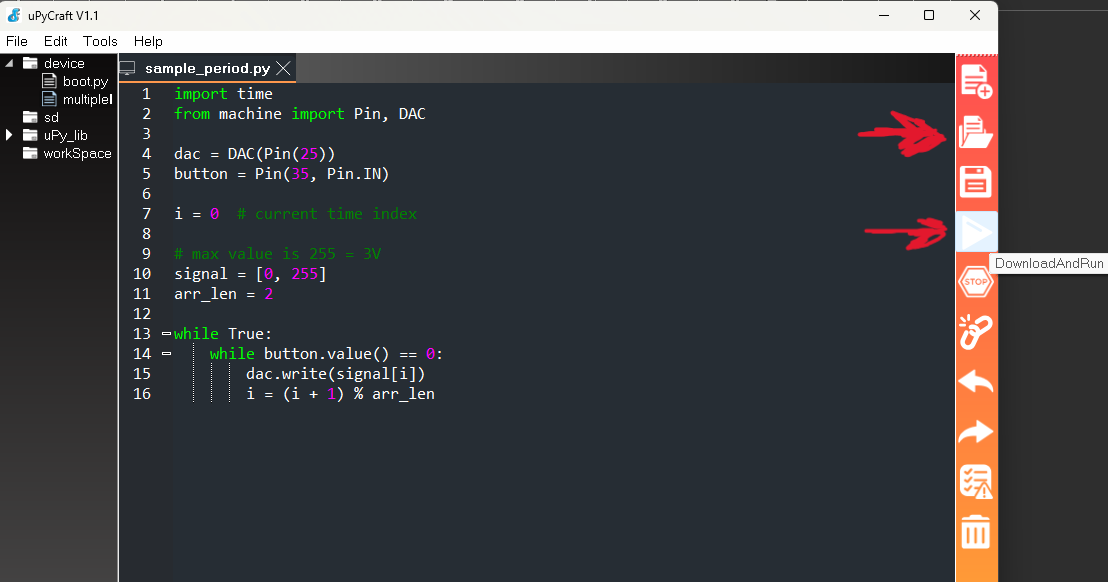
Description automatically generated

1. Se inițializează conexiunea cu ESP32 apăsând pe butonul Connect din meniul din dreapta în uPyCraft

A screenshot of a computer program

Description automatically generated

1. Se deschide în uPyCraft fișierul din repozitor /Proiect\_MicroPython/Python\_Scripts/ sample\_period.py, apoi se apasă butonul din meniul din dreapta în uPyCraft DownloadAndRun



1. Se determină cu ajutorul unui osciloscop perioada de eșantionare. În cazul nostru, perioada de eșantionare a fost de 44us.

A screen shot of a computer

Description automatically generated

1. Se rulează programul din repozitor /Proiect\_MicroPython/Python\_Scripts/ signal\_list\_generator/main.py și se copiază listele afișate în variabilele din /Proiect\_MicroPython/Python\_Scripts/multiple\_signals.py. Conțiuntul acestui se deschide în uPyCraft, se salvează și se apasă butonul DownloadAndRun
2. **Atribuțiile membrilor**

|  |  |  |
| --- | --- | --- |
| Nr. Crt. | Nume, Prenume, Seria, Grupa | Contribuții |
| 1. | Fratu-Halunga Theodor-Corneliu  (421F) | * Realizarea programului ce rulează pe ESP32 cu MicroPython * Realizarea clasei wrapper Signal din signal\_list\_generator * Implementarea repozitorului de pe GitHub |
| 2. | Beliciu Andrei  (421F) | * Realizarea funcțiilor din signal\_list\_generator pentru generarea diferitelor tipuri de semnale * Documentația |

1. **Bibliografie**

[1] <https://www.youtube.com/playlist?list=PLruzZCuhcsGO_kN1JXV0_YdzJ6ObZB7i1>

[2][https://how2electronics.com/esp32-micropython-upycraft-getting- started/#:~:text=There%20are%20only%20a%20few,the%20code%20in%20ESP32%20board](https://how2electronics.com/esp32-micropython-upycraft-getting-%20%20%20started/#:~:text=There%20are%20only%20a%20few,the%20code%20in%20ESP32%20board)

[3] <https://docs.micropython.org/en/latest/esp32/quickref.html>

[4] <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/hw-reference/esp32/get-started-devkitc.html>

[5] <https://numpy.org/doc/stable/reference/index.html#reference>

[6] <https://www.digitalocean.com/community/tutorials/python-convert-numpy-array-to-list>

[7] <https://www.geeksforgeeks.org/matplotlib-pyplot-plot-function-in-python/>

[8] <https://arduinointro.com/articles/projects/adding-sounds-to-arduino-using-the-mh-fmd-piezo-buzzer-module#google_vignette>

[9] <https://xn--llions-yua.jutge.org/upc-python-cookbook/signal-processing/signals.html>