Documentare cod

Pentru realizarea unui cod coerent si lizibil programul a fost realizat in mai multe etape cheie. Este esential ca primul lucru realizat in program sa fie definirea si configurarea pini-lor led-urilor si butoanelor, folosite in proiect. Led-urile reprezinta iesirile digitale, iar butonalele intrarile este de mentionat si nevoia existentei rezistentelor de pull-up necesare pentru o buna functionare a butoanelor. Configurarea pinilor realizandu-se desigur in setup(). Led-urile folosite pentru simularea incarcarii bateriei au fost memorate intr-un vector pentru o gestionare mai usoara a acestora.

In loop() are loc partea de gestionare a cerintelor proiectului. Inainte de a realiza logica de functionare a loader-ului de incarcare este esentiala stabilirea clara a starii in care se afla in acel moment statia, acesta fiind si prima etapa din functia loop(). Starea initiala a statiei este desigur libera. Pentru a oberva modificarea starii statiei din liber in ocupat am utilizat un buton de start (START\_BUTTON). Pentru a fii siguri ca butonul a fost apasat am implemntat o logica de debouncing in care verificam daca timpul scurs de la ultima schimbare a starii depaseste intervalul de debounce- millis() - timpAnteriorDebounceStart) > intarziereDebounceStart) (millis() timpul actual, timpAnteriorDebounceStart- ultima data cand starea butonului a fost modificata, intarziereDebounceStart- timpul de debounce ce l-am ales sa fie egal cu 50 ms) -. Daca butonul a fost apasat atunci statia incepe sa incarce si vom marca acest lucru folosind o variabila de tip bool, incarcareActiva ce ii dam valoarea true. Daca statia se afla deja in procesul de incarcare, apasarea butonului de start nu va avea efect asupra statiei, procesul nu se va relua de la 0, iar incarcareActiva va ramane true.

Daca incarcarea nu este activa atunci vom folosi digitalWrite pentru a seta valoarea HIGH a pin-ului verde al led-ului RGB, statia fiind astfel libera.

Daca incarcarea este activa vom permite si oprirea fortata a incarcarii. Aceasta oprire fortata se implementeaza folosind un buton, STOP\_BUTTON. La fel ca in cazul butonului de start, am folosit debouncing pentru a verifica daca a fost cu adevarat apasat, dar de data aceasta am modificat intervalul de timp pentru care butonul trebuie sa fie apasat cu 1000ms. In cazul in care butonul de stop a fost apasat, o sa setam o variabila bool stopIncarcare cu true si trecem la finalul loaderului de led-uri.

Desigur daca nu este apasat butonul de stop si incarcarea este activa, vom implenta logica de simulare a pragurilor de incarcare a led-urilor. Initial setam ledul RGB la culoarea rosie si incepem parcurgerea vectorului (loaderLed[4]={10,9,8,7} )in care am memorat pinii led-urilor ce simuleaza incarcarea. Daca led-ul nu a mai clipit pana in acel moment se memoreaza timpul exact in care va incepe sa clipeasca pentru prima data. Am folosit o structura de decizie pentru a gestiona logica de clipire a led-urilor. Initial verificam daca a trecut suficient timp de la ultima schimbare a ledului si daca nu am ajuns inca la finalul loader-ului in cazul in care ambele conditii sunt adevarate schimbam starea led-ului pentru a avea fenomenul de clipire (utilizam un if care schimba starea din HIGH -> LOW si LOW->HIGH). Urmatoarea conditie din structura este daca am trecut peste intervalul fix de 3s, daca da atunci led-ul curent ramane aprins si trecem desigur la urmatorul led. In cazul in care am ajuns la finalul vectorului cu led-uri, reprezinta si ultima conditie din structura de decizie, atunci vom reseta variabilele, indicele vectorului trece la 0, stopIncarcare devine fals la fel si incarcare activa, led-ul RGB devine verde si cel mai important se apeleaza functia blinkFinal().

Functia blinkFinal() realizeaza comportamentul de clipire simultana de 3 ori a led-urilor din loader. In functie se foloseste delay(), pentru a nu permite inregistrarea unui nou start sau stop in timpul clipirii simultane. In functie am folosit un while() care trece led-urile din HIGH in LOW.

In cazul in care incarcarea este activa si apasam butonul de stop dupa cum am mentionat ajungem la finalul vectorului, deci intram pe ultima ramura a structurii repetitive. Am ales sa se intre pe aceasta ramura deoarece apeleaza atat functia de blinkFinal(), cat si resetarea variabilelor pentru o functionare corecta a codului la urmatoarul eveniment de activare a statiei de incarcare.