**Aplicație protocol MQTT**

**pe platforma FRDM KL25Z**

**PROIECT REALIZAT DE**

Perju Dragoș-Ștefan

Grupa 1306A

**Cerință**

Având la dispoziție protocolul MQTT și platforma de dezvoltare FRDM KL25Z, să se realizeze conexiunea între 2 platforme FRDM KL25Z, conectate prin USB la un PC rulând un server/broker MQTT, care prin utilizarea touchpad-ului integrat al uneia dintre platforme să controleze intensitatea luminii albastre emis de LEDul integrat de pe cealaltă platformă.

**Rezultat**

Folosind siteul de dezvoltare [mbed.com](https://developer.mbed.org/), s-a realizat o aplicație în C++ pentru platforma FRDM KL25Z ușor ajustabilă pentru situațiile în care folosim doar o platformă sau două, în format transmițător-receptor. Această ajustare se realizează prin includerea comenzilor de preprocesare #define SENDER și/sau #define RECEIVER. În ambele situații, aplicația se folosește de protocolul [MQTT](http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html) și pachetele descrise CONNECT, PUBLISH, SUBSCRIBE pentru a transmite și recepționa valori de tip float cu 2 zecimale, care să denote luminozitatea LEDului setat pe culoarea albastră. Serverul MQTT ales pentru acest proiect este [Mosquitto](http://mosquitto.org/), iar legătura TCP-Conexiune serială este realizată de programul [COMbyTCP](https://sourceforge.net/projects/combytcp/).

**Detalii implementare**

Aplicația C++ implementată lucrează pe o buclă while(1) ce conține 3 blocuri switch..case, fiecare funcționând ca un automat de stări finite, fiecare responsabil pentru conectare, publicare de mesaje și ascultare de mesaje respectiv. În orice situație, când platforma este aleasă pentru transmisie, recepție sau amândouă în același timp, la început blocurile de publicare și ascultare de mesaje sunt „dezactivate” (valorile pentru blocurile switch nu sunt corespunzătoare pentru niciun caz din bloc). Doar blocul de conectare este activ, pachetul CONNECT fiind primul ce trebuie transmis unui server MQTT, după cum dictează protocolul.

CONNECT

La început, platforma transmite un pachet CONNECT construit de o funcție de program cu același nume, cu 2 parametri. Primul parametru este timpul în secunde cerut serverului să aștepte după ultimul mesaj comunicat între cele două entități, înainte de a întrerupe conexiunea (timeout). Implicit, în aplicație este folosit o valoare de 600 secunde (10 minute) pentru acest paramaetru.

Cel de-al doilea parametru este numele clientului care și-l auto-asignează pentru a fi recunoscut în mesajele informative dispuse utilizatorului de către server și pentru distingerea efectivă a clienților – în cazul în care folosim 2 platforme FRDM KL25Z, amândouă ascultând și recepționând în același timp.

Transmisia se realizează prin comunicația serială, platforma FRDM KL25Z punând la dispoziție prin conexiunea USB, pe lângă accesul la încărcarea sa cu programe, si un port corespunzător conexiunii seriale pentru PC-ul gazdă. În program, clasa responsabilă pentru realizarea acestei conexiuni este clasa BufferedSerial, care folosește pentru comuncație un buffer implicit de 256 de octeți și întreruperi pe pinii de recepție/transmisie a portului corespunzător. Baud rate este setat la 9600.

Transmiterea pachetului CONNECT este corespunzător cu starea 0 a „automatului de stări” construit prin intermediul primului bloc switch..case din program, responsabil pentru procesul de handshake cu serverul MQTT. Imediat după transmisia acestui pachet, se trece în starea 1, unde platforma încearcă să identifice pachetul CONNACK ce ar trebui să fie transmis de către server ca răspuns. Identificarea se realizează prin funcția de program cu același nume, care verifică mai întâi dacă a fost transmis orice fel de dată prin comunicația serială, apoi încearcă identificarea acelor date ca fiind pachet CONNACK. Starea 1 apelează această funcție - în cel mai rău caz - de 3 ori, anticipând o eventuală întârziere din partea serverului, iar dacă acesta nu este prezent sau transmite un răspuns eronat de fiecare dată, se trece înapoi în starea 0.

Dacă s-a identificat un pachet CONNACK, starea pentru primul block switch..case devine -1 (se dezactivează acest bloc) și sunt activate blocurile de publicare și ascultare de mesaje, depinzând de rolul ales pentru platformă – transmițător/receptor sau amândouă.

PUBLISH

În continuare, dacă programul a fost compliat cu activarea celui de-al doilea bloc switch..case, prin folosirea comenzii de preprocesare #define SENDER, se intră în cazul 0 al acestuia – care este singurul caz din acest bloc. S-a păstrat blocul switch..case cu un singur caz pentru a menține consistența în scrierea codului.

Cazul 0 se ocupă cu citirea poziției atingerii pe touchpad și transmiterea valorii corespunzătoare în format float cu 2 zecimale. Bineînțeles, totul este transmis, recepționat și interpretat în formatul string, dar informația în spate reprezintă un număr real în intervalul [0,1], scalată la lungimea fizică a touchpadului.

Biblioteca care se ocupă cu touchpadul și obținerea acestei valori va returna, prin funcția corespunzătoare, o valoare de 0 atunci când touchpadul nu este atins. Știind acest lucru, cazul 0 va transmite un pachet PUBLISH doar dacă valoarea care se vrea pusă este diferită de valoarea veche obținută ultima dată când a fost transmis un pachet PUBLISH și dacă această valoare este diferită de 0. Astfel se transmit pachete doar când este nevoie - lucru cerut de natura protocolului MQTT, fiind un protocol lightweight de minimă comunicație.

Funcția ce împachetează și transmite pachetul PUBLISH cere, ca parametri, topicul pe care va transmite mesajul și mesajul în sine. Nu există restricții pentru dimensiunea acestor parametri exprimați în șiruri de caractere. Topicul folosit de program este „LED”. Dorim posibilitatea de a alege un topic diferit decât cel prestabilit pentru a avea 2 platforme FRDM KL25Z cu rolul de transmițător și receptor în același timp, dar pe canale/topicuri diferite pentru a avea comunicație bidirecțională bine diferențiată – protocolul permițând subscrierea și publicarea de mesaje pe același topic fără a evita transmiterea mesajelor care au fost primite de la o sursă înapoi la acea sursă, dacă ea este subscrisă topicului.

SUBSCRIBE

Ultimul bloc switch..case prevăzut în program are rolul de a stabili subscrierea la topicul folosit pentru cazul în care platforma este aleasă pentru recepționarea valorilor transmise de blocul switch..case anterior de pe altă platformă sau de pe aceeași platformă, după caz. Astfel, acest proces de transmitere a pachetului SUBSCRIBE și a recepționării pachetului SUBACK este pentru a asigura că serverul MQTT, când primește un mesaj nou în legătură cu topicul de interes (în acest caz topicul „LED”), va înainta acel mesaj clienților care au realizat acest proces de subscriere pe topicul respectiv.

Similar primului bloc switch..case, se va transmite pachetul SUBSCRIBE în cazul 0 a declarației switch, după care la următoarea trecere în bucla while(1) se va intra în cazul 1, unde se apelează de 3 ori – în cel mai rău caz – funcția de program ce se ocupă cu recepționarea a pachetului SUBACK. Dacă această funcție rezultă în eșec de fiecare dată, se reia cazul 0. Dacă recepționarea se face cu succes, se trece la cazul 2, care se ocupă cu recepționarea de pachete PUBLISH din partea serverului, folosind funcția de program read\_PUBLISH.

Ne dorim primirea unui pachet SUBACK de la server deoarece, în construcția pachetului SUBSCRIBE, setăm jumătatea a doua din primului octet pe 0x2 (0010), semnalând astfel prin setarea nivelului de QoS la 1 că așteptăm un răspuns la pachetul transmis de noi SUBSCRIBE. Observăm că ceea ce s-a făcut nu este descris în documentația protocolului MQTT, jumătatea a doua a primului octet din pachetul SUBSCRIBE fiind descris a fi rezervat. Însă, această setare s-a făcut în urma faptului că serverul Mosquitto nu primea altfel pachetele SUBSCRIBE transmise de noi, fără acest nivel QoS setat – care este descris la alte pachete precum PUBLISH în documentația protocolului.

Necesitatea acestei setări s-a luat în considerare după folosirea programului de urmărit pachete de rețea, [Wireshark](https://www.wireshark.org/), în combinație cu un utilitar ce permite monitorizarea interfeței de loopback [RawCap](http://www.netresec.com/?page=RawCap), pe conexiunea și subscrierea pe un topic realizat între serverul Mosquitto și un program client MQTT ales, denumit [MQTTLens](https://chrome.google.com/webstore/detail/mqttlens/hemojaaeigabkbcookmlgmdigohjobjm). Se poate vedea fișierul pcap rezultat în anexa acestei documentații.

În cazul 2 a blocului switch..case responsabil pentru primirea mesajelor, se consideră că platforma este subscrisă topicului de interes, implicit ales a fi „LED”, pe serverul MQTT, și apoi se iau în considerare 2 variabile de tip float care să desemneze intensitatea luminii albastre de pe LEDul multicolor integrat pe platforma FRDM KL25Z. Variabila readValue este setată inițial pe -1.0 și reprezintă variabila în care funcția read\_PUBLISH va seta valoarea de return. Valoarea de -1.0 este corespunzătoare cazului în care funcția read\_PUBLISH nu a primit nicio dată nouă pe serială pe care să o interpreteze sau dacă a primit un pachet care nu este de tip PUBLISH (tipul 3).

Variabila ledValue este setată inițial, înainte de toate blocurile switch..case, a fi de valoare 0.85 – o valoare potrivită pentru intensitatea luminii LEDului (unde 1 este stins și 0 este aprins pe intensitate maximă). Dacă funcția read\_PUBLISH nu returnează o valoare diferită de -1.0, această variabilă va fi folosită pentru setarea pentru prima dată a LEDului pe culoarea albastră la prima trecere în acest caz al blocului.

Funcția read\_PUBLISH extrage, dintr-un pachet PUBLISH primit, topicul și mesajul. Această funcție este creată special pentru aplicația prevăzută de acest proiect, în sensul în care caută pachete PUBLISH dimensionate exact pe o valoare de PUBLISHSIZE = 11, potrivit pentru un astfel de pachet să țină un număr real pe două zecimale și să fie destinată unui topic cu numele format din 3 litere. De asemenea, deși se extrage topicul, nu se verifică dacă acesta este același cu cel la care platforma s-a subscris. Funcția la urmă convertește mesajul extras într-un format float și returnează valoarea.

**Concluzie**

Acestea sunt etapele prin care o platformă sau două (sau chiar mai multe) de dezvoltare FRDM KL25Z trec pentru ca să își aprindă cu o anumită intensitate LEDurile pe culoare albastră, intensitate dictată doar de o singură altă platformă conectată la același server MQTT. Bineînțeles că se pot observa multe îmbunătățiri ce pot fi aduse chiar și acestei simple aplicații. Una dintre aceste îmbunătățiri sunt robustețea necesară funcției read\_PUBLISH, care poate evita a avea necesar o dimensionare fixă a pachetului PUBLISH așteptat, precum și o verificare a topicului primit - a fi la fel cu cel la care platforma s-a subscris.

Cea de-a doua îmbunătățire este implementarea transmisiei/recepției de pachete PINGREQ și PINGRESP, pentru a împiedica deconectarea eventuală realizată de server din cauza inactivității. De asemenea, odată ce o platformă s-a conectat, nu mai identifică dacă cumva conexiunea a fost întreruptă pentru a încerca reconectarea.

Deși programul este, în teorie, capabil să lucreze și bidirectional (touchpadul de pe fiecare două platforme să controleze lumina albastră de pe cealaltă platformă) – trebuie făcut anumite modificări și compilate diferite versiuni ale programului pentru fiecare platformă (topicuri diferite, PUBLISHSIZE eventual diferit datorită numelui topicului ș.a.m.d.) și, de asemenea, nu s-a testat pe deplin această facilitate.

Două aspecte ce nu au fost menționate în decursul acestei documentații sunt aprinderea LEDului pe culori diferite pe parcursul conectării, subscrierii și publicării de mesaje și o întârziere impusă pentru fiecare etapă menționată, pentru a nu supraaglomera serverul MQTT. Perechile de culori și etapele corespunzătoare sunt:

* galben – transmitere CONNECT;
* roșu – așteptare CONNACK;
* verde (la transmițător) – gata pentru PUBLISH\*;
* verde (la receptor) – transmitere SUBSCRIBE;
* portocaliu – se așteaptă SUBACK;
* albastru – gata pentru primire pachete PUBLISH.

Se menționează că o pereche culoare-etapă nu se mai folosește când doar o platformă este folosită pentru transmisie și recepție în același timp (cea marcată cu steluță).

Întârzierea impusă este presetată a fi de 50 ms și „împrăștiată” prin program, astfel încât, în cel mai rău caz, procesorul stă suspendat o perioadă maximă de 2000 ms.

*Proiectul a fost realizat de Dragoș-Ștefan Perju, student în anul 3, grupa 1306A, la Facultatea de Automatică și Calculatoare în cadrul Universității Tehnice „Gheorghe Asachi” Iași, sub îndrumarea domnului profesor dr. Nicolae-Alexandru Botezatu, pentru disciplina Rețele de Calculatoare.*