

# LWM – MQTT Gateway

## 1. Opis

Projekt integruje LWM stack od firmy Atmel a Ethernet NIC Wiznet 5500 pripojený na kit s MCU Atmel ATMEGA256FR2. Výsledné zariadenie slúži brána medzi sieťami LWM a Ethernet, resp. MQTT.

## 2. Popis fungovania

Po zapnutí, zariadenie ako prvé inicializuje LWM stack a periférie MCU – SPI, Timer, Watchdog. Následne prebehne inicializácia sieťovej karty podľa zadaných nastavení (vid. Kapitola 3) a zariadenie vypíše stav karty a jej nastavenia.

Inicializáciu sieťovej karty nasleduje pokus o spojenie so vzdialeným MQTT serverom, podľa nastavení v zdrojovom kóde (vid. Kapitola 3). V prípade neúspešného pripojenia, zariadenie vypíše chybovú hlášku a reštartuje sa.

V prípade úspešného pripojenia, pokračuje zariadenie prihlásením k daným MQTT topicom pomocou MQTTSubscribe. Ak je prihlásenie úspešné, zariadenie do konzole vypíše „Subscribed “názov topicu”“.

Po prihlásení do MQTT topicu, zariadenie prejde do cyklu, v ktorom pravidelne kontroluje, či mu na LWM neprišli nové správy. V intervale `mqtt_timer` zariadenie spúšťa funkciu `MQTTYield`, pomocou ktorej dostane zariadenie priestor na spracovanie prichádzajúcich MQTT správ.

V prípade prijatia správy cez LWM, je nastavený príznak `data_ready`. Pri ďalšom prechode `while` cyklom sú data spracované do JSON formátu a odoslané na príslušný MQTT topic podľa typu dát pomocou funkcie `mqtt_pub`.

Obdobne sú spracované aj prichádzajúce MQTT konfiguračné správy, ktoré sú prijaté vo formáte JSON a sú spracované externou knižnicou.

## 3. Nastavenie

Všetky nastavenia sa nachádzajú na začiatku súboru „main.c“.

- **#define CLIENT** – nastavuje názov zariadenia pre MQTT server
- **#define USER** – nastavuje meno používateľa pre MQTT server
- **#define PASSWORD** – nastavuje heslo používateľa pre MQTT server
- **#define PUBLISH\_CONFIG\_0** – nastavuje topic pre konfiguračné správy
- **#define SUBSCRIBE\_TOPIC** – nastavuje topic na ktorom bude zariadenie odoberať správy
- **#define PUBLISH\_TOPIC** – nastavuje základný topic, ku ktorému sa pridávajú jednotlivé témy podľa zvolených senzorov
- **mqtt\_target** – IP adresa MQTT serveru
- **netInfo** – nastavenia sieťovej karty – IP adresa, Maska podsiete, MAC adresa, Gateway adresa.

Pod „// define sensors“ sa nachádzajú definície zariadení a ich senzorov vo forme poľa s názvami veličín. Pre definíciu ďalších zariadení je potrebné použiť nasledujúcu šablónu:

```
char *zariadenie[50] = {"Vel1", "Vel2", "Vel3", "Vel4", "Vel5", "Vel6", "Vel7", "Vel8"};
```

, kde „Vel*n*“ je názov veličiny. Zároveň je potrebné upraviť funkciu „create\_json“ pridaním ďalšieho switch case pre zariadenie s novou maskou.

## 4. Funkcie

- **bool appDataInd(NWK\_DataInd\_t \*ind)**
  - spracovanie prijatých dát z LWM
  - \*ind - pointer na prijaté dáta.
- **static bool appAddrInd(NWK\_DataInd\_t \*ind)**
  - získa cluster ID od zariadenia so senzormi – použité na odlíšenie jednotlivých zariadení a ich senzorov.
  - \*ind - pointer na prijaté dáta.
- **void mqtt\_pub(Client\* mqtt\_client, char \* mqtt\_topic, char \* mqtt\_msg, int mqtt\_msg\_len)**
  - odosiela správy na MQTT server
  - **char \* mqtt\_topic** – mqtt topic
  - **char \* mqtt\_msg** - mqtt správa, ktorú chceme odoslať
  - **int mqtt\_msg\_len** – dĺžka mqtt správy
- **int32\_t MQTTSubscribe(Client\* c, const char\* topicFilter, enum QoS qos, messageHandler messageHandler)**
  - prihlasuje sa k odberu správ s daným topicom
  - **const char\* topicFilter** – mqtt topic
  - **enum QoS qos** - nastavenie QoS pre MQTT
  - **messageHandler** – callback na spracovanie správ
- **static void create\_json()**
  - vytvorí struct z prijatých dát vo forme MQTT správy v JSON formáte a MQTT topicu podľa druhu prijatých dát.
- **void executeCommand(char \*command)**
  - spracuje konfiguračnú správu vo formáte JSON a nastaví podľa nej dané parametre
  - **char \*command** – prijatá správa

## 5. Použité knižnice

- m1284p\_wiz5500 [https://github.com/maxxir/m1284p\\_wiz5500](https://github.com/maxxir/m1284p_wiz5500)
- JSON Decoder for PIC® and AVR® Devices [https://github.com/MicrochipTech/json\\_decoder](https://github.com/MicrochipTech/json_decoder)