BMA400:

* Tryb pracy: low power – po wykryciu ruchu automatycznie przechodzi do normal.
* Zasada działania: czujnik reaguje na ruch w jakiejkolwiek osi, trzeba dostosować, żeby nie zbierał małych drgań, jak np. wiatr zawieje. Jak wykryje ruch, to wybudza STM
* Dane są przesyłane przez I2C do STM32

STM32:

* Tryb uśpienia: STOP – wybudzanie przerwaniem z czujnika. Można potem spróbować ustawić tryb SLEEP, bo to się odbywa na pinie, który zamiast jako EXTI0 (przerwanie), może być skonfigurowany jako SYS\_WKUP (chyba tak się nazywał – on może ze SLEEP wybudzić), tylko trzeba kod zmodyfikować odrobinę, żeby nie było tego fragmentu z ustawieniem flagi w przerwaniu.
* Przeliczone procenty są przesyłane na bieżąco do ESP przez UART.
* Uwaga: w kodzie w pliku z przerwaniami (stmf4xx\_it.c – jakoś tak) trzeba po każdej zapisanej konfiguracji w GUI usuwać wygenerowaną automatycznie linię. Ona powinna wywoływać przerwanie w main.c, ale tego nie robi, ręczne wywoływanie napisane dalej już działa.
* Trzeba np. analizatorem stanów logicznych sprawdzić czy po wybudzaniu się nie powoduje jakichś skoków na PA5 (EN\_IO) i ew. skorygować w programie, bo to by bardzo dużo energii pobierało.

ESP8266:

* On ma być zawsze wyłączony. Będzie aktywowany przez STM32 na pinie EN\_ESP w momencie wysyłania danych.
* Przesyła procent otwarcia do drugiego ESP przez ESP-NOW.

Zasada działania:

* STM wybudzany ze SLEEP przy wykryciu ruchu.
* Co STM robi zanim
* ESP wysyła do STM maksymalny czas otwarcia i zamknięcia.
* BMA400 sprawdza w którą stronę porusza się brama.
* Procent otwarcia będzie wysyłany do ESP co np. 2 s. ESP będzie tylko wtedy uruchamiany (min. 200 ms na inicjalizację).