Protokół CAN ,E-maks power

Spis treści

[1.Wprowadzenie 2](#_Toc12172649)

[1.1. Podział ramek 2](#_Toc12172650)

# 1.Wprowadzenie

Protokół CAN będzie używany pomiędzy następującymi płytkami:

* Interfejs Użytkownika – płytka wysyła informacje na temat położenia joysticka.
* Sterownik LED - otrzymuje informacje na temat zapalania LED.
* Interfejs Radiowy - otrzymuje dane po CAN z pozostałych płytek i wysyła je dalej droga radiową, wysyła po CAN dane otrzymane drogą radiową.

Konfiguracja protokołu:

* Ramki o standardowym polu ID (11- bit)
* Prędkość magistrali 1 Mbit/s
* Nie ma ramek RTR
  1. Podział ramek:

Każda płytka może wysłać 30 różnych ramek.

**Ramki błędów** – wysyłane jeżeli płytka wykryje błąd który może zaważyć o rzetelności wysyłanych danych.

**Ramki danych** – ramki z danymi.

Podział ramek:

1. Ze względu na rodzaj:

* 0xE[x] – ramki błędów
* 0xD[x] – ramki z danymi

Gdzie x = (0h ; 88h)

1. Ze względu na pochodzenie:

* 0x[X][0h - 23h] – ramki wysyłane przez interfejs użytkownika
* 0x[X[24h - 57h] – ramki wysyłane przez sterownik LED
* 0x[X][58h – 88h] – ramki wysyłane przez interfejs użytkownika

Gdzie X= {D,E}

Przykładowe ID ramki:

0xE40 – ramka błędu od sterownika LED

Ramki danych są wysyłane okresowo, ramki błędów w momencie pojawienia się błędów;

# Ramki przypisane do poszczególnych płytek

## 2.1 Interfejs użytkownika

Tabela 1 pokazuje ramki wysyłane przez joystick

Tabela 1

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Nazwa** | **B0** | **B1** | **B2** | **B3** | **B4** | **B5** | **B6** | **B7** |
| **0xE0** | **JOY\_ERROR\_FRAME** | **Kod[uint16]** | |  |  |  |  |  |  |
| **0xF0** | **JOY\_ADC\_READINGS** | **AXIS\_CODE**  **[uint16]** | | **AXIS\_VALUE [uint16]** | |  |  |  |  |

Kody osi tabela 2.

Tabela 2

|  |  |
| --- | --- |
| Oś/kod | oś |
| 0 | X |
| 1 | Y |

Kody błędów [tabela 3]:

Tabela 3

|  |  |
| --- | --- |
| Kod | Nazwa |
| 0 | Błąd dczytu ADC |
| 1 | Odczyt poza zakresem |
| 2 | Brak napięcia yousticka |

Wartości AXIS\_VALUE są zakodowanymi wartościami procentowymi(procent zakresu joysticka).

Kodowanie wartości na uint16:

**AXIS\_VALUE** (uint16) = **AXIS\_VALUE** (float) \* 216 / 5.5;

Dekodowanie wartości dekodowanie wartości na float:

**AXIS\_VALUE**(float) = **AXIS\_VALUE**(uint16) \* 5.5 / 216 ;