
NobleProg

Interfejsy szeregowe w mikrokontrolerach

Grzegorz Mazur

Plan prezentacji

- Interfejsy szeregowe ogólnie
- UART
- SPI
- I2C
- OneWire
- Interfejsy specyficzne dla układów

Interfejsy szeregowe

- Interfejsy, w których poszczególne bity słowa danych są przesyłane kolejno, bit, po bicie, po tych samych liniach sygnałowych
- Tradycyjna klasyfikacja i.s.
  - Synchroniczne – z przesyłanym oddzielną linią sygnałową sygnałem synchromizującym (zegarem)
  - Asynchroniczne – nie wymagające przesyłania sygnału zegarowego
- Współcześnie wiele interfejsów przesyła dane wraz z sygnałem synchronizującym po tej samej linii – sygnał zegarowy jest zakodowany w sygnale danych (np. OneWire, WS2812)

UART

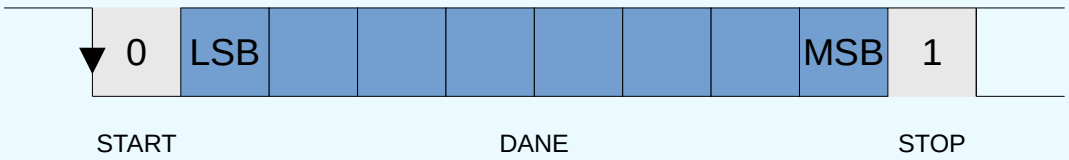
- Najprostszy interfejs szeregowy
  - W wersji minimalnej – tylko dwie linie sygnałowe TxD, RxD
- Asynchroniczny – nie wymaga transmisji przebiegu taktującego
  - Komunikujące się urządzenia muszą znać szybkość transmisji
  - Wzorce częstotliwości obu urządzeń, określające szybkość transmisji, muszą mieć precyzję nie gorszą niż 2%

UART - transmisja

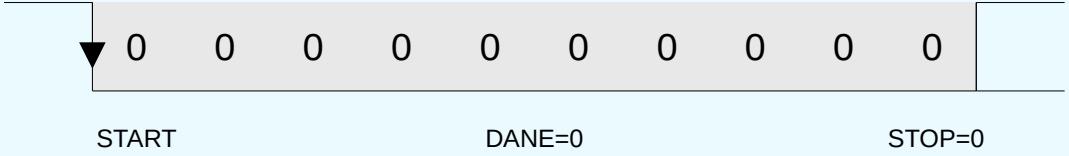
- Szybkość transmisji determinuje czas trwania bitu
- Początek transmisji sygnalizowany zmianą stanu linii – początek „bitu startu”
- Po wykryciu zbocza bitu startu odbiornik odlicza pół czasu transmisji bitu, weryfikuje ważność bitu startu, a następnie próbkuje stan linii w odstępach równych okresom bitów („w środku” bitu)

UART

Ramka danych



Symbol Break



UART – logika transmisji danych

- Stan linii:
  - Nieaktywny (SPACE) – 1
  - Aktywny (MARK) – 0
- Format ramki danych:
  - Start (1 bit o wartości 0)
  - Dane – 5..8 bitów
  - Kontrola parzystości lub znacznik – brak lub 1 bit
  - Stop – 1 lub 2 bity o wartości 1

UART – bit stop

- Bit stop ma wartość SPACE (linia nieaktywna)
- Przy ciągłej transmisji wielu ramek służy do poprawnego wykrycia bitu startu kolejnej ramki
- Wykrycie stanu MARK w miejscu bitu stop oznacza błąd (np. niezgodność szybkości nadajnika i odbiornika)

UART – stany specjalne

- Bezczynność (IDLE) – stan SPACE trwający dłużej niż określony czas (zależny od realizacji, > czasu transmisji ramki)
  - Niekiedy używany jako znacznik końca transmisji pakietu danych
- Przerwa (BREAK) – stan MARK trwający dłużej niż czas transmisji pełnej ramki (z bitami stopu)
  - Niekiedy używany jako znacznik początku

UART – wykrywanie błędów

- Fałszywy start – stan MARK trwający krócej niż czas transmisji jednego bitu
- Zakłócenia – zmieniająca się wartość linii RxD w połowie czasu transmisji bitu
- Błąd parzystości – niewłaściwa wartość bitu parzystości
- Błąd ramki – stan MARK w czasie transmisji bitów stopu (sygnalizowany również przy BREAK)

Standardy bazujące na UART

- RS232 i podobne (EIA/TIA-562)
  - Sygnalizacja napięciowa, linia asymetryczna
- RS485, RS422
  - Linia symetryczna, dopasowana
- Pętla prądowa
- UART na poziomach logicznych TTL/CMOS

RS232/TIA-562

- Popularna, nie do końca poprawna nazwa
  - W rzeczywistości wiele standardów o podobnej charakterystyce elektrycznej, różnych złączach i zbiorach sygnałów
- Pierwotnie było to połączenie komputera z modemem
  - Zestaw sygnałów odpowiada logice modemów z lat 1970-tych

RS232

- DTE – Data Terminal Equipment – urządzenie nadrzędne; zwykle komputer
- DCE – Data Communication Equipment – urządzenie podrzędne; modem
- Obecnie zwykle łączymy dwa urządzenia DTE, co nie jest objęte standardami
  - Zmieniona funkcjonalność linii
- Interfejsy UART w mikrokontrolerach używają terminologii DTE

RS232 - sygnały

	DTE DCE	
GND		Masa sygnałowa
TXD	→	Dane nadawane przez DTE
RXD	←	Dane odbierane przez DTE
RTS	→	Gotowość DTE do nadawania
CTS	←	Gotowość DCE do odbioru danych z DTE
DTR	→	Sygnalizacja włączenia DTE
DSR	←	Sygnalizacja włączenia DCE
CD	←	DCE gotowy do transmisji danych dp DTE
RI	←	(Dzwonienie) – DCE nawiązuje połącenie

Logika sygnałów RS232

- Wszystkie linie poza RXD/TXD są zanegowane
- Poziomem aktywnym wszystkich linii jest poziom niski
- W RS232 poziomowi niskiemu odpowiada napięcie dodatnie, a wysokiemu – ujemne

Standard napięciowy RS232

- Sygnały bipolarne
- Odbiór:
  - 0 powyżej 3 V
  - 1 poniżej -3V
  - W praktyce wszystkie odbiorniki pracują z progiem ok. +2 V!
- Nadawanie – w nowszych wersjach standardu mniejsza amplituda – obecnie +5V, -5V

Synchronizacja transmisji w RS232

- W praktyce UART służy zwykle do połączenia dwóch równoprawnych urządzeń pełniących rolę DTE
- Istotną informacją jest gotowość urządzenia do odbioru danych
  - Wejście: CTS
  - Wyjście: linia RTS zmienia znaczenie na RTR (Ready To Receive)
- Linie RXD/TXD i RTR/CTS są łączone „na krzyż”

RS485

- Standard transmisji umożliwiający pracę wielu urządzeń z interfejsami UART na jednej szynie (typowo do 32)
- Linia symetryczna, różnicowa
- Możliwy zasięg:
  - 1200 m przy 100 kb/s
  - 12 m przy 10 Mb/s
- Nowe układy – do 100 Mb/s

Protokoły logiczne korzystające z RS485

- Profibus
- Fieldbus
- DMX512
- Modbus

UART w STM32

- „ułamkowe podzielniki” - umożliwiają uzyskanie dokładnej częstotliwości transmisji przy dowolnej częstotliwości zegara
- Możliwość transmisji przy użyciu DMA

Programowanie UART w STM32

- BRR – podzielnik
- CR2 – opcje dodatkowe
- CR3 – opcje dodatkowe i DMA
- Na końcu – CR1 – tryb pracy, włączanie przerwań, włączenie nadawania/odbioru, włączenie UART

```
USART1->BRR = (PCLK_FREQ + BAUD / 2) / BAUD;
USART1->CR1 = USART_CR1_TE | USART_CR1_RE | USART_CR1_UE;
```

Ciekawe opcje w CR2

- Negacja linii RX, TX
  - umożliwia współpracę z RS232 bez transceivera (konieczny rezystor szeregowy >= 47k na linii RX)
  - Ułatwia wstawienie np. transoptora
- Zamiana linii RX i TX – kiedy pomylimy się rysując schemat...

UART – źródła przerwań

- Wolne miejsce w buforze nadajnika – TXE
- Dane w buforze odbiornika – RXNE
- Błędy odbioru – FE, ORE, FE
  - włączane razem z RXNE
- Koniec transmisji – TC (przydatne dla RS485)
- Przerwa w odbiorze – IDLE
- … i kilka innych

UART – obsługa przerwań

- Przy włączonym RXNE konieczne sprawdzenie NF, FE i ORE
- Zgłoszenie przerwania TXE kasowane wyłącznie przez zapis danych
  - Nie można skasować go przez wyłączenie przerwania – TXEIE
  - Przerwanie TXE należy włączać wyłącznie wtedy, gdy są gotowe dane do wysłania, a wyłączać – przed zapisem ostatniej danej

UART i DMA

- Pożyteczne przy nadawaniu
- Problematyczne przy odbiorze

Dodatkowe linie UART

- -CTS (wejście) – możliwość sprzętowego blokowania nadawania
- -RTS (wyjście, w rzeczywistości jest to -RTR) – gotowość do odbioru
- DE – sprzętowe sterowanie kierunkiem bufora dla RS485 (tylko w nowszych wersjach UART – F0, L0, L4, niedostępne w F1, F4)

DMA w STM32

- Dostępny nawet w najmniejszych modelach
- 2 wersje:
  - „Stream” w F4 i F7, z ograniczeniami dostępu do AHB (porty GPIO!) w module DMA1
  - Podstawowa w pozostałych, bez ograniczeń dostępu
  - 1 lub 2 moduły, do 7 kanałów w każdym
- Podczas programowania parametrów kanał DMA musi być wyłączony (EN=0)

SPI, Microwire,

- Grupa interfejsów międzyukładowych o wspólnej ogólnej charakterystyce
  - Transmisja synchroniczna, równocześnie w obu kierunkach
  - Sygnał zegarowy SCK generowany przez układ nadrzędny
  - Dwie linie danych MOSI i MISO (Master Out, Slave In, Master In, Slave Out)
  - Sygnał wyboru/uaktywnienia urządzenia NSS (Slave Select)

SPI

- Transmisja zawsze dwukierunkowa
  - W każdym cyklu zegara Master nadaje i odbiera jeden bit
  - Slave może ignorować lub nie odbierać danych, ale Master o tym nie wie
  - Slave może nie nadawać danych, … ale Master o tym nie wie
  - Master odbiera tyle bitów, ile nadał
  - Slave jest zawsze gotowy do transmisji

SPI

- Na szynie SPI może być wiele układów podrzędnych
  - Często stosowane rozwiązanie
  - Każdy układ podrzędny ma własną, oddzielną linię NSS
- Jest możliwa konfiguracja z wieloma układami nadrzędnymi
  - Rzadko stosowana

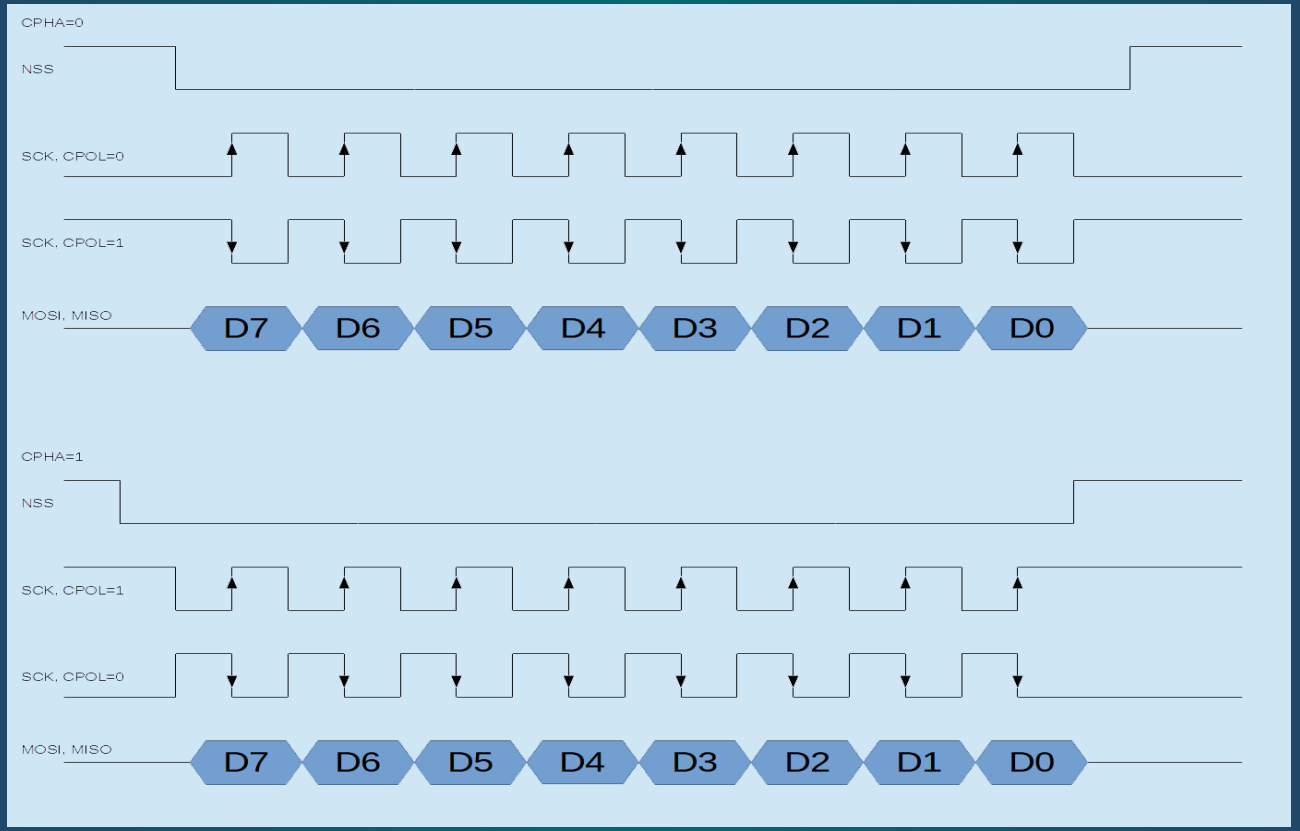
SPI – wybór układu podrzędnego

- Uaktywnienie linii NSS oznacza wybór układu i początek transmisji ramki
- Niektóre układy Slave przy konfiguracji z jednym Slave mogą pracować jako stale aktywne (NSS stale aktywne lub brak linii NSS)
  - Wtedy początek ramki jest sygnalizowany przez określony wzorzec danych na linii MOSI, np. jedynka po serii zer

Parametry CPOL i CPHA

- CPOL – poziom linii SCK w stanie bezczynności (polaryzacja zegara)
- CPHA – faza zegara
  - 0 – dane próbkowane pierwszym zboczem i kolejnymi nieparzystymi
  - 1 – dane próbkowane drugim zboczem i kolejnymi parzystymi
- Większość układów peryferyjnych może działać przy CPOL=0 i CPHA=0 albo CPOL=1 i CPHA=1

SPI


CPHA=0
NSS
SCK, CPOL=0
SCK, CPOL=1
MOSI, MISO
D7
D6
D5
D4
D3
D2
D1
D0
CPHA=1
NSS
SCK, CPOL=1
SCK, CPOL=0
MOSI, MISO
D7
D6
D5
D4
D3
D2
D1
D0

SPI w STM32

- Dwie wersje
  - Uboższa 8/16 bit – F2, F4
  - Bogatsza 4..16 bit – F0, L0, L4...
- Uwaga – przy danych <= 8 b:
  - Dostępy do rejestru DR muszą być jawnie 8-bitowe z atrybutem volatile (domyślnie są volatile 16-bitowe)
  - Musi być ustawiony bit FRXTH w rejestrze CR2

SPI w STM32

- Częstotliwość transmisji – podzielniki tylko 2^n w stosunku do zegara szyny peryferiala
  - Jeżeli niezbędna jest konkretna wartość częstotliwości – należy odpowiednio dobrać częstotliwość taktowania µC!
- SSM, SSI – ważne bity w rejestrze CR1
- Warto używać DMA

SPI - inicjowanie

- CR2 – opcje, w tym:
  - Długość ramki danych
  - FRXTH
  - DMA
- CR1
  - Dzielnik częstotliwości, włączenie - SPE
- Zmiana ustawień w CR1, CR2 możliwa tylko przy wyzerowanym SPE

I2C (TWI)

- Interfejs międzyukładowy, synchroniczny
- 2 linie
  - SCL – zegar
  - SDA - dane
- Szyna z możliwością adresowania – może łączyć wiele układów
- Transmisja zwykle znacznie wolniejsza niż w SPI

I2C - komunikacja

- Linie SCL i SDA są typu „otwarty dren”
- Niezbędne rezystory podciągające 2..10k
- Podczas transmisji danych stan SDA zmienia się przy nieaktywnym (wysokim) SCL
- Zmiana stanu SDA przy niskim SCL oznacza:
  - Z 1 na 0 – START
  - Z 0 na 1 – STOP

I2C - komunikacja

- Transmisję taktuje układ nadrzędny
- Układ podrzędny może spowolnić transmisję, „przytrzymując” linię SCL w stanie niskim

I2C

- Ramki 8-bitowe tworzące pakiety
- Po każdej ramce następuje bit potwierdzenia, nadawany przez odbiorcę
  - 0 – ACK
  - 1 – NACK – brak odpowiedzi, niegotowość lub błąd

I2C – komunikacja

- Pakiet rozpoczyna się od symbolu START
- Pierwszy bajt pakietu jest nadawany przez MASTER i zawiera adres i bit określający kierunek transmisji następnych bajtów

I2C

- Kolejne bajty pakietu są transmitowane w kierunku określonym przez pierwszy bajt
- Zmiana kierunku wymaga ponownego rozpoczęcia pakietu
- Transmisja kończy się symbolem specjalnym STOP

I2C

- Zazwyczaj transakcja I2C składa się z jednej lub dwóch faz: zapisu lub zapisu i odczytu
  - W fazie zapisu podaje się np. polecenie i adres wewnątrz układu potrzebne do późniejszego odczytu, wtedy po fazie zapisu nie ma STOP
- <START><addr+dir><dane><START><addr+dir><dane><STOP>

I2C w STM32

- Złożony peryferial, skomplikowane oprogramowanie
  - Wiele stanów, złożone sekwencje czynności
  - Specjalna obsługa transmisji dwóch ostatnich bajtów

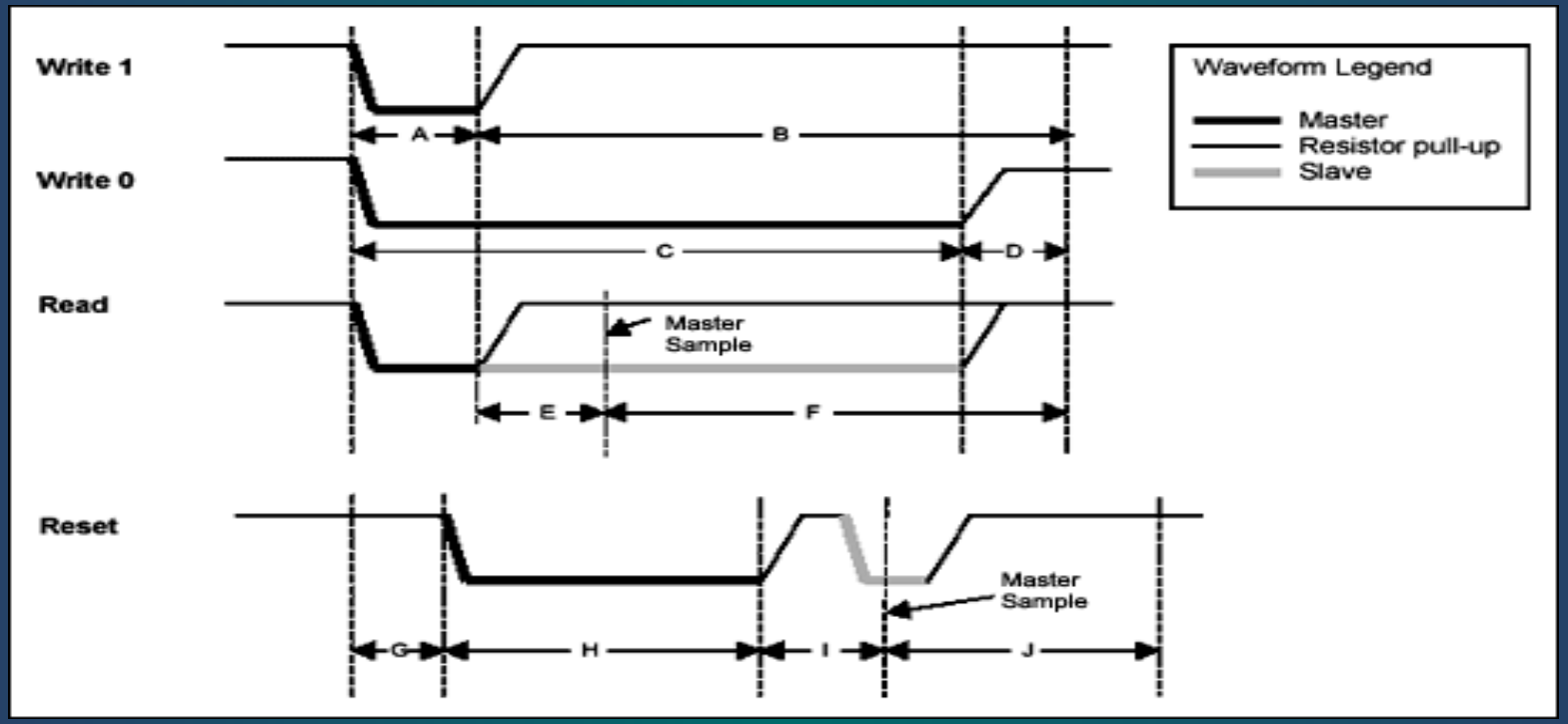
Oprogramowanie I2C

- Konstrukcja oprogramowania wykracza poza ramy czasowe kursu
- Użyjemy biblioteki HAL
- Pożyteczne funkcje: MemoryWrite, MemoryRead – zapewniają współpracę z większością układów I2C
- Funkcje HAL STM32 dostępne w 3 wersjach:
  - Podstawowe – blokujące, synchroniczne
  - Z przyrostkiem _IT – asynchroniczne z użyciem przerwań, zalecane do użycia przy I2C
  - Z przyrostkiem _DMA – z użyciem przerwań i DMA

OneWire

- Standard firmy Dallas (obecnie Maxim)
- Dwukierunkowa transmisja po jednej linii
- Opcjonalne zasilanie układów z linii danych
- Możliwe połączenie wielu układów na jednej linii
- Układy mają nadane fabrycznie unikatowe adresy 64-bitowe
- Możliwe wykrycie adresów wszystkich układów na szynie

OneWire – komunikacja



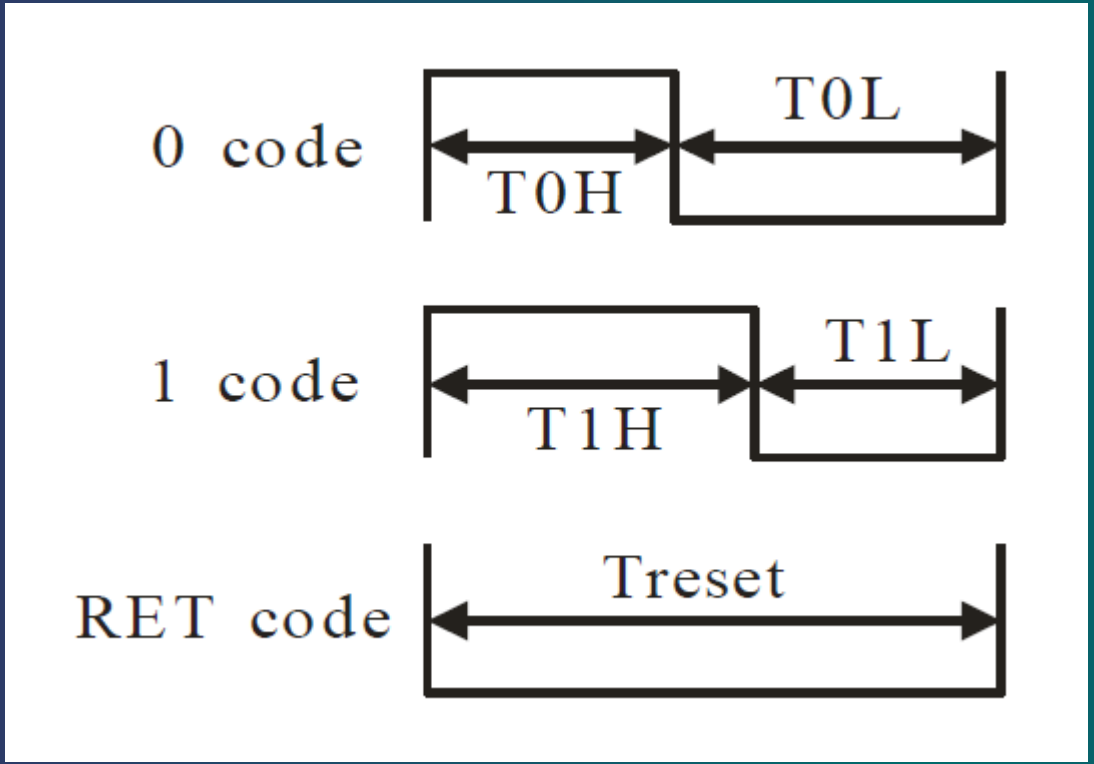
OneWire - realizacja

- Brak obsługi sprzętowej w mikrokontrolerach z powodów licencyjnych
- Możliwe realizacje:
  - Całkowicie programowa z opóźnieniami
  - Timer wielokanałowy, przerwania
    - 2 lub 3 przerwania na bit
  - UART – zwarte RX i TX, 1 bit na ramkę, konieczna zmiana szybkości dla RESET

OneWire - oprogramowanie

- Dwu- lub trójpoziomowy automat
  - Poziom bitu
  - Poziom transakcji
  - Poziom logiczny – sekwencja transakcji

WS2812 – komunikacja



WS2812 – realizacje na STM32

- SPI
  - Bit transmitowany do WS2812 kodowany na trzech bitach SPI
  - Wymagana częstotliwość 2.4 MHz +- 10%
- UART
  - 3 bity kodowane w ramce UART z 7 bitami danych (+ START i STOP – łącznie 9 bitów nadawanych w ramce)
  - Wymagana negacja linii TXD – stanem nieaktywnym musi być 0
- Timer
  - Tryb PWM, ładowanie wypełnienia przez DMA
  - Każdy bit kodowany słowem 16-bitowym (wypełnienie PWM) – duża zajętość pamięci