# KURS STM32

*Wojciech Olech*

## CZĘŚĆ VIII: SPI

# SPI - WPROWADZENIE

SPI (*Serial Peripheral Interface*) jest szybką magistralą danych, bardzo popularnie wykorzystywaną jako interfejs do transmisji dużych ilości danych z dużą szybkością.

Popularnie wykorzystywana jako główny interfejs dla wyświetlaczy, przetworników ADC i kart pamięci

# SPI - SPOSÓB DZIAŁANIA

SPI jest magistralą master-slave. Oznacza to, że master jest urządzeniem nadrzędym które może inicjować transmisję, a slave jest urządzeniem podrzędnym które pracuje pod opieką mastera.

SPI jest magistralą synchroniczną - oznacza to, że jedną z linii pomiędzy dwoma urządzenami jest linia zegarowa, taktowana przez mastera.

Do działania wymaga co najmniej czterech linii, oznaczanych jako:

- **SCLK** lub **CLK** - *Serial Clock* lub *Clock* - linia zegarowa, taktowana przez mastera
- **MOSI** - *Master Output, Slave Input* - linia danych, po której master wysyła, a slave odbiera dane
- **MISO** - *Master Input, Slave Output* - linia danych, po której slave wysyła, a master odbiera dane
- **SS** lub **CS** - *Slave Select* lub *Chip Select* - pin który określa z którym urządzeniem się komunikujemy.

Linia zegarowa i linie danych mogą być współdzielone między wszystkimi slave'ami i masterem. Każdy slave **musi** mieć własną, osobistą linię **CS**. Oznacza to że do podłączenia *n* urządzeń do mastera, potrzebujemy *n + 3* linii, z czego **CS** często jest zwykłym pinem GPIO.

# SPI - TIMING DIAGRAM

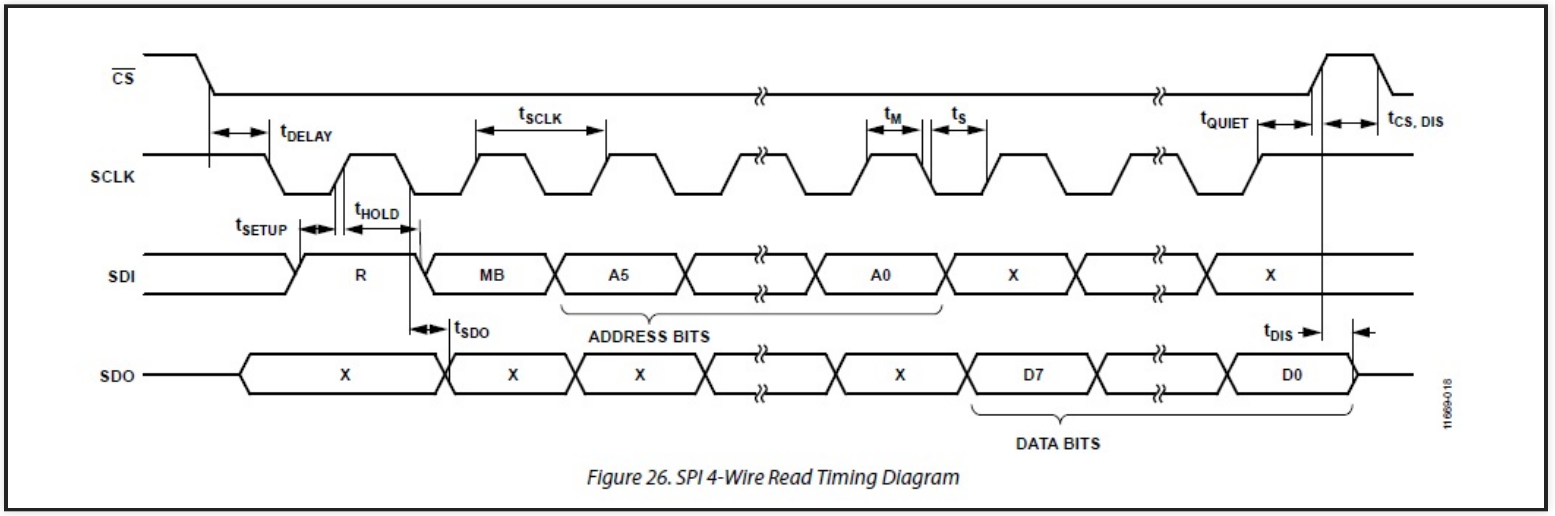


*Figure 26. SPI 4-Wire Read Timing Diagram*

Ramka SPI jest zazwyczaj 8-bitowa. To, co dokładnie oznaczają bity w ramkach zależne jest od specyfikacji protokołu komunikacyjnego między urządzeniami.

# SPI - OPCJE KONFIGURACJI

Typowe opcje które należy skonfigurować w SPI, i które muszą być takie same dla mastera i wszystkich slave'ów to:

- Szybkość zegara - większa szybkość zegara = większa przepustowość magistrali
- Polaryzacja zegara - jeśli jest *niska*, to zegar domyślnie ma stan niski (który utrzymuje się kiedy linia nie pracuje), a podczas pracy jest cyklicznie podciągany do stanu wysokiego. Jeśli jest *wysoka*, to zegar domyślnie ma stan wysoki, a podczas pracy jest cyklicznie podciągany do stanu niskiego.
- Faza zegara - jeśli jest ustawiona na 1 zbocze (*1 Edge*), to oznacza że odczyt/zmiana bitu następuje na pierwszym zboczu zegara (przy polaryzacji niskiej - zbocze narastające, przy polaryzacji wysokiej - zbocze opadające). Jeśli jest ustawiona na 2 zbocze, następuje przy drugim zboczu zegara (polaryzacja niska - zbocze opadające, polaryzacja wysoka - zbocze narastające).

CPOL - polaryzacja zegara
CPHA - faza zegara

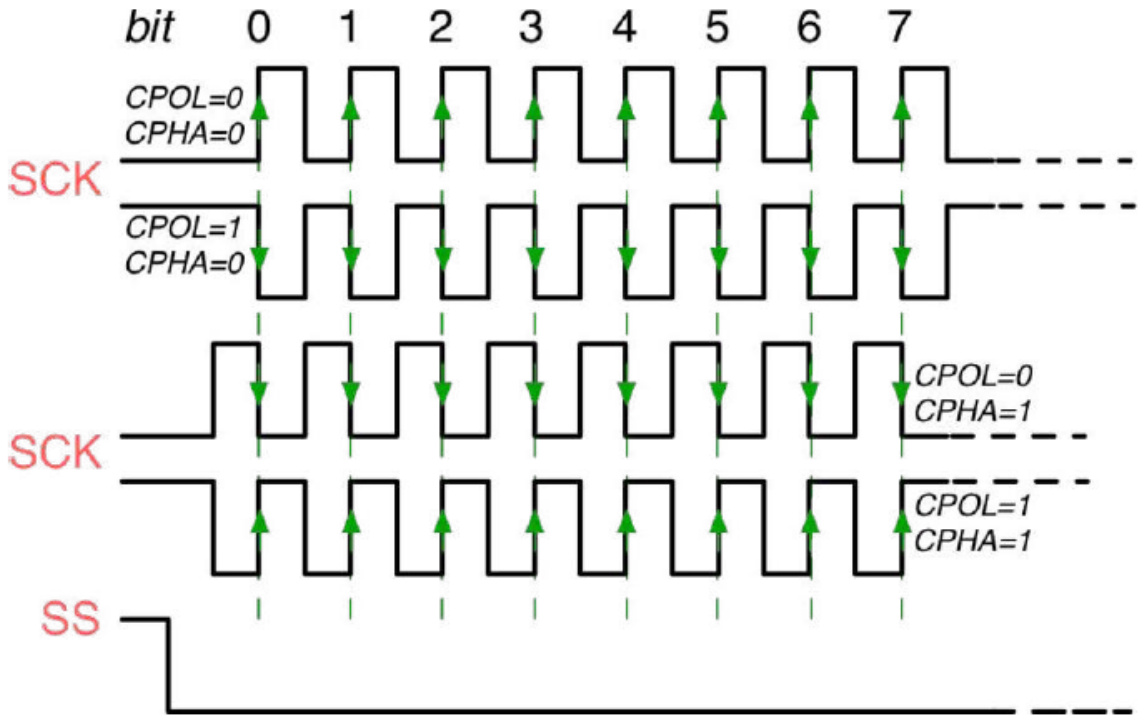


Figure 4: The SPI timing diagram according CPOL and CPHA settings

*Źródło: Mastering STM32*

# TRYBY PRACY SPI W MIKROKONTROLERACH STM32

SPI może pracować w kilku trybach, które różnią się możliwościami komunikacji i ilością linii.

Standardowym trybem jest full-duplex, w którym wymagane są 4 linie i transmisja może odbywać się jednocześnie w obie strony.

Oprócz tego, istnieje tryb half-duplex który pozwala na komunikację z użyciem 3 linii (zegar, SS i jedna linia danych) w obie strony, ale nie ma możliwości wysyłania i odbierania danych po obu stronach w tym samym czasie.

Najbardziej ograniczonymi trybami są transmit/receive only, ponieważ umożliwiają tylko wysyłanie/odbiór danych po 3 liniach.

## HARDWARE NSS

STMy wspierają tryb *hardware NSS*. NSS to pin chip select, można skonfigurować go w trybie hardware co spowoduje że automatycznie podczas komunikacji będzie on ustawiany w odpowiedni tryb. Uniemożliwi to jednak używanie pinu chip select jako dowolnego pinu GPIO, więc nie nadaje się w sytuacjach gdzie hardware'owy pin NSS jest już zajęty.

Posiada on dwa tryby: *input* i *output*. W trybie *output*, nasze urządzenie w trybie master może działać tylko z jednym slavem. W trybie input, NSS działa jako standardowy chip select kiedy wybieramy nasze urządzenie jako slave, ale żeby pracować w trybie master musimy ręcznie wysterować ten pin, więc nie blokuje nam opcji multi-mastera (wielu slave'ów podłączonych do naszego mastera)

# KOMUNIKACJA Z UŻYCIEM SPI

SPI, podobnie jak UART, pozwala na trzy tryby pracy: polling, na przerwaniach i z użyciem DMA. Działają one tak samo, jak w przypadku UARTa.

Istnieją trzy funkcje każdego typu jakie możemy wykorzystać do komunikacji:

- **HAL_SPI_Transmit** - wysyłanie danych
- **HAL_SPI_Receive** - odbiór danych
- **HAL_SPI_TransmitReceive** - jednoczesny odbiór i wysyłanie danych, możliwy tylko w trybie full-duplex

Oprócz tego, SPI ma standardowe callbacki. Wszystkie funkcje są dostępne w nagłówku `stm32f4xx_hal_spi.h`

# DODATKOWE MATERIAŁY:

SPI w STM32F7:
https://www.st.com/content/ccc/resource/training/technical/product_training/group0/3e/ee/cd/b7/84/4b/45/ee/ST