Spis treści

[1. Wprowadzenie 2](#_Toc22918097)

[1.1. Cel pracy 2](#_Toc22918098)

[1.2. Zakres pracy i założenia 2](#_Toc22918099)

[2. Komunikacja z kartą pamięci SD 3](#_Toc22918100)

[2.1. Protokół SPI 3](#_Toc22918101)

[2.2. Standard kart pamięci SD i MMC 4](#_Toc22918102)

[2.3. Komendy odsługujące kartę 7](#_Toc22918103)

[2.4. Komunikacja z kartą 8](#_Toc22918104)

[3. Architektura projektowanych modułów 9](#_Toc22918105)

[3.1. Kontroler SPI 9](#_Toc22918106)

[3.2. Moduł sterownika kart pamięci 9](#_Toc22918107)

[3.3. Moduł testujący 9](#_Toc22918108)

[4. Symulacja i testowanie sterownika karty SD 9](#_Toc22918109)

[4.1. Symulacja kontrolera SPI 9](#_Toc22918110)

[4.2. Sposób testowania układu 9](#_Toc22918111)

[4.3. Dodatkowo użyte moduły 9](#_Toc22918112)

[4.4. Otrzymane wyniki testów 9](#_Toc22918113)

[5. Podsumowanie 9](#_Toc22918114)

# Wprowadzenie

Rozwój technologii w obecnych czasach postępuje w zaskakującym tempie. Coraz więcej urządzeń powszechnego użytku posiada zaawansowane układy cyfrowe pozwalające na wymianę danych z użytkownikiem. A dzięki nanotechnologii układy te zajmują co raz mniejszą powierzchnię co pozwala na umieszczenie więcej modułów logicznych w jednym układzie scalonym. Mimo dużych zmian oraz prędkości ich wprowadzania niektóre protokoły oraz rozwiązania nadal są istotne i aktualne.

## Cel pracy

Celem pracy jest zaprojektowanie oraz implementacja modułu sterownika obsługi kart pamięci w języku Verilog. W tym celu należy zapoznać się z protokołami komunikacyjnymi oraz standardami kart pamięci. Dokonać implementacji wybranego interfejsu oraz całego modułu dla układu FPGA. Wykonać symulację oraz opracować sposób testowania projektowanego układu dla płytki laboratoryjnej z układem MACHXO2 4000HC firmy Lattice wypożyczonej z katedry uczelni. A do samej implementacji i symulacji użyć oprogramowania Lattice Diamond 3.10 oraz Active HDL 10.5.

## Zakres pracy i założenia

Do zakresu pracy należy:

- Analiza działania protokołów komunikacyjnych dla kart SD/MMC.

- Opracowanie koncepcji działania modułu sterownika i jego implementacja w języku Verilog.

- Opracowanie środowiska testowego (symulacyjnego i sprzętowego) umożliwiającego weryfikację implementacji.

- Analiza wyników.

Na podstawie celu oraz zakresu pracy założyłem że implementacje wykonam dla protokołu SPI z wykorzystaniem karty micro SD.

# Komunikacja z kartą pamięci SD

## Protokół SPI

SPI (Serial Peripheral Interface) jest dupleksowym, szybkim, synchronicznym interfejsem szeregowym. Kiedyś używany był do łączenia urządzeń takich jak drukarka, skaner, kamera itp. do komputera osobistego PC (Personal Computer). Dziś został wyparty przez standard USB (Universal Serial Bus) ale w dalszym ciągu jest używany do komunikacji między aplikacjami używającymi monitorów, kard pamięci, czujników itp. Standard ten pozwala na łączenie jednego układu nadrzędnego (Master) z wieloma urządzeniami pobocznymi (Slave) do czego wykorzystywane są cztery linie sygnałowe.



Rysunek 1 Sposób łączenia układów poprzez interfejs SPI.

Rysunek 1 przedstawia sposób komunikacji pomiędzy modułami (nadrzędnym i podrzędnym) poprzez interfejs SPI. Obydwa układy połączone są liniami MOSI (Master-Out Slave-In), MISO (Master-In Slave-Out), SCLK (Serial Clock) i SS (Slave Select). Dane z obydwóch 8-bitowych rejestrów przesuwających (Shift register) przesyłane są pomiędzy urządzeniami w takt sygnału zegarowego SCLK. Natomiast sygnał SS pełni rolę wyboru układu podrzędnego (chip select) oraz synchronizacji startu transmisji. Dla układów z transmisją jednokierunkową tj. przetwornik analogowo cyfrowy (ADC) lub przetwornik cyfrowo-analogowy (DAC) druga linia transmisji może być pominięta. Bity danych przesyłane są od najbardziej znaczącego bitu (MSB) do najmniej znaczącego (LSB).



Rysunek 2 Wykorzystanie interfejsu SPI z wieloma modułami Slave.

Rysunek 2 przedstawia przykładowe podłączenie trzech układów podrzędnych. Zapis lub odczyt następuje po ustawieniu wyjścia układu SPI Master na stan niski. W interfejsie SPI przesunięcie i zatrzaśnięcie się danych podczas zapisu bądź odczytu odbywa się na przeciwnych zboczach zegara. W związku z tym wyróżniamy cztery różne tryby pracy które wynikają z konfiguracji interfejsu SPI po stronie układu nadrzędnego. Tryby te przedstawia tabela 1.

|  |  |
| --- | --- |
| Tryby pracy SPI | Przebiegi czasowe |
| Tryb 0  Pozytywne zbocze  Zatrzask i przesunięcie  (CPHA=0, CPOL=0) |  |
| Tryb 1  Pozytywne zbocze  Zatrzask i przesunięcie |  |
| Tryb 2  Negatywne zbocze  Zatrzask i przesunięcie |  |
| Tryb 3  Negatywne zbocze  Zatrzask i przesunięcie |  |

Tabela 1 Cztery tryby pracy interfejsu SPI.

## Standard kart pamięci SD i MMC

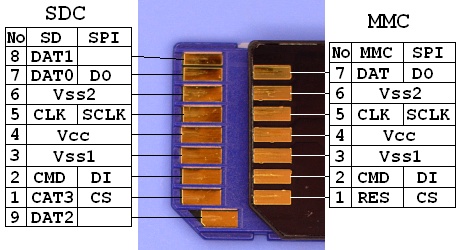
Do najbardziej popularnych standardów kart pamięci należą karty SD (Secure Digital) oraz karty MMC (Multi Media Card). Karta SD została opracowana jako zgodna z większością kart MMC dlatego sprzęt zgodny z kartami SD może w większości przypadków korzystać z MMC. Dostępne są również wersje o zmniejszonych rozmiarach, takie jak RS-MMC, miniSD i microSD, z taką samą funkcjonalnością. Karty te mają wbudowaną pamięć flash i mikrokontroler. Sterowanie tej karty pamięci tj. kasowanie, odczyt, zapis, sterowanie błędami i równoważenie zużycia jest wykonywane bezpośrednio na niej. Dane są przesyłane między kartą pamięci a kontrolerem hosta jako bloki danych w jednostkach 512 bajtów, dlatego karty te można postrzegać jako dyski twarde z punktu widzenia warstw wyższego poziomu (systemu operacyjnego).

Schemat blokowy karty SD pokazano na rys. 3. Składa się ona z 9-stykowego interfejsu (Interface Driver), wewnętrznego układu taktującego (Internal Clock), kontrolera karty (Card Interface Controller), wbudowanych rejestrów karty (OCR, CID, RCA, DSR, CSD, SCR), interfejsu rdzenia pamięci (Memory Card Interface) oraz rdzenia pamięci (Memory Core). 9-pinowy interfejs pozwala na wymianę danych między podłączonym systemem a kontrolerem karty. Kontroler może odczytywać lub zapisywać dane z lub do rdzenia pamięci za pomocą interfejsu rdzenia pamięci. Rejestry wewnętrzne przechowują aktualny stan karty. Kontroler odpowiada na dwa typy żądań użytkowników: kontrolę i dane. Żądania kontrolne konfigurują operację sterownika i umożliwiają dostęp do rejestrów. Natomiast żądania danych służą do odczytu danych lub zapisywania danych w rdzeniu pamięci.

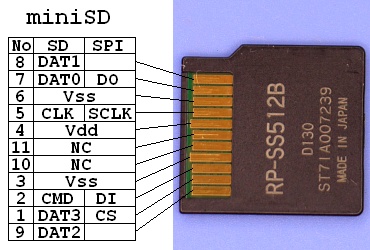


Rysunek 3 Schemat blokowy karty SD.

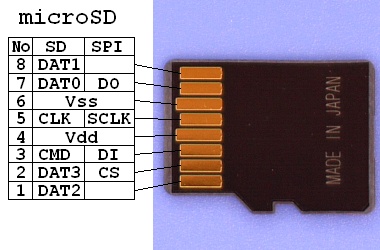
Karta SD posiada 9 pól stykowych czyli o 2 piny więcej niż karta MMC co widać na rysunku 4. Trzy z nich to styki przeznaczone dla zasilania dlatego liczba efektywnych pinów zarezerwowanych do komunikacji z kartą wynosi sześć dla karty SD oraz cztery dla karty MMC. Z tego powodu transfer danych pomiędzy hostem a kartą odbywa się za pomocą synchronicznego interfejsu szeregowego. Zakres napięcia zasilania przechowywany jest w rejestrze OCR który powinien być odczytany i potwierdzony podczas inicjalizacji karty. Jednak napięcie zasilania może być również ustawione na stałe w zakresie 3,0 V do 3,3 V bez potwierdzenia ponieważ wszystkie karty SD i MMC działają w przedziale 2,7 V do 3,6 V. Pobór prądu przez kartę pamięci podczas operacji zapisu może osiągnąć maksymalnie 100 mA dlatego nie należy podłączać większego napięcia niż dopuszczalne (np. 5V) gdyż spowoduje to uszkodzenie karty a układ zasilania karty powinien dostarczać maksymalnie 100mA.



Rysunek 4 Opis wyprowadzeń styków dla karty SD po lewej stronie oraz dla karty MMC po prawej.



Rysunek 5 Opis wyprowadzeń styków dla karty miniSD.



Rysunek 6 Opis wyprowadzeń styków dla karty microSD.

## Komendy odsługujące kartę

Komunikacja z kartą SD odbywa się poprzez wysyłanie do niej odpowiednich komend i otrzymywaniu odpowiedzi. Karta może pracować w dwóch trybach: SD oraz SPI. Domyślnym trybem pracy jest tryb SD dlatego aby można było nawiązać komunikację z kartą w trybie SPI należy ją ustawić w ten tryb. Prawidłowa komenda obsługująca kartę SD składa się z 48 bitów i jest przedstawiona na rysunku 7.



Rysunek 7 Format 48-bitowej komendy.

Pierwsze dwa najbardziej znaczące bity to bity startu które ostawione są na wartość 01 dla każdej komendy. Następnym polem jest 6-bitowy numer komendy (Command Number) który określa rodzaj wykonywanej operacji na karcie SD. Dalej znajduje się 32-bitowy argument z wartością oraz 7-bitowe pole CRC (Cyclic Redundancy Chceck) oraz bit stopu ustawiony na wartość 1 dla każdej operacji. Wartość CRC jest używana dla weryfikacji integralności otrzymanej odpowiedzi. Domyślnie karta SD ignoruje bity CRC dla większości komend (za wyjątkiem komendy CMD8) chyba że użytkownik zażąda sprawdzenia bitów CRC po każdej wiadomości.



Rysunek 8 Format podstawowej 8-bitowej odpowiedzi na każdą komendę w trybie SPI.



Rysunek 9 Format 40-bitowej odpowiedzi.

Po każdej wysłanej komendzie karta wysyła odpowiedź która może być 8-bitowa (rysunek 8) lub 40-bitowa (rysunek 9). Odpowiedź zawsze rozpoczyna się od zera następnie znajdują się znaczniki flag oznaczające: błąd podanego parametru (parameter error), błąd podanego adresu (address error), kasowanie błędu sekwencji (erase sequence error), błąd sumy kontrolnej CRC (CRC error), niepoprawna komenda (illegal command), kasowanie resetu (erase reset), karta w stanie bezczynności (in idle state). W przypadku odpowiedzi 40-bitowej otrzymujemy dodatkowe pola tj. 4-bitowe pole z wersją (Version), 18-bitowe zarezerwowane pole (Reserved), 4-bitowe pole określające napięcie karty (Voltage) oraz 8-bitowe pole kontrolne (Check Pattern).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nazwa komendy** | **Argument** | **Odpowiedź** | **Dane** | **Nazwa skrótowa** | **Opis** |
| CMD0 | None(0) | R1 | Nie | GO\_IDLE\_STATE | Programowy reset karty. |
| CMD1 | None(0) | R1 | Nie | SEND\_OP\_COND | Inicjacja procesu inicjalizacji. |
| ACMD41  (\*) | \*\* | R1 | Nie | APP\_SEND\_OP\_COND | Inicjacja procesu inicjalizacji (tylko dla SD) |
| CMD8 | \*\*\* | R7 | Nie | SEND\_IF\_COND | Sprawdzanie zakresu napięć (tylko dla SD V2) |
| CMD9 | None(0) | R1 | Tak | SEND\_CSD | Odczyt rejestru CSD. |
| CMD10 | None(0) | R1 | Tak | SEND\_CID | Odczyt rejestru CID. |
| CMD12 | None(0) | R1b | Nie | STOP\_TRANSMISSION | Stop odczytu danych. |
| CMD16 | Block  length[31:0] | R1 | Nie | SET\_BLOCKLEN | Zmiana odczytu lub zapisu długości bloku. |
| CMD17 | Address[31:0] | R1 | Tak | READ\_SINGLE\_BLOCK | Odczyt bloku. |
| CMD18 | Address[31:0] | R1 | Tak | READ\_MULTIPLE\_BLOCK | Odczyt wielu bloków. |
| CMD23 | Number of  blocks[15:0] | R1 | Nie | SET\_BLOCK\_COUNT | Definiowanie liczby bloków do transmisji przy następnej komendzie odczytu lub zapisu w trybie wielu bloków (tylko dla standardu MMC). |
| ACMD23  (\*) | Number of  blocks[22:0] | R1 | Nie | SET\_WR\_BLOCK\_ERASE\_COUNT | Definiowanie liczby bloków do skasowania przy następnej komendzie zapisu w trybie multi blokowości (tylko dla SD). |
| CMD24 | Address[31:0] | R1 | Tak | WRITE\_BLOCK | Zapis jednego bloku pamięci. |
| CMD25 | Address[31:0] | R1 | Tak | WRITE\_MULTIPLE\_BLOCK | Zapis wielu bloków. |
| CMD55  (\*) | None(0) | R1 | Nie | APP\_CMD | Ładowanie komendy ACMD<n> |
| CMD58 | None(0) | R3 | Nie | READ\_OCR | Odczyt rejestru OCR. |
| \*:ACMD<n> means a command sequence of CMD55-CMD<n>.  \*\*: Rsv(0)[31], HCS[30], Rsv(0)[29:0]  \*\*\*: Rsv(0)[31:12], Supply Voltage(1)[11:8], Check Pattern(0xAA)[7:0] | | | | | |

Tabela 2 Lista komend obsługiwanych przez kartę pamięci.

## Komunikacja z kartą

Wysyłanie polecenia do karty SD odbywa się gdy linia MOSI jest ustawiona w stan wysoki co oznacza że żadna wiadomość nie została wysłana. Proces wysyłania wiadomości rozpoczyna się od umieszczenia najbardziej znaczącego bitu MSB (most-significant bit) na wyjściu MOSI przy jednoczesnym przełączaniu sygnału zegarowego SCLK z poziomu niskiego na wysoki oraz z wysokiego na niski. Powtórzenie tej procedury aż do najmniej znaczącego bitu LSB (least significant bit) pozwala na przesłanie zawartości całego rejestru do karty SD która składa się na całą komendę.

Aby zapewnić poprawne działanie karty SD, zegar taktujący SCLK powinien zawierać się w przedziale 100 – 400 kHz a karta musi być ustawiona w trybie SPI. Żeby ją ustawić w ten tryb linie MOSI oraz CS muszą być ustawione w stan logicznej jedynki przynajmniej przez 74 takty zegarowe. Po tym czasie należy ustawić CS na zero oraz do karty wysłać komendę resetu (CMD0).

Gdy tylko karta SD otrzyma komendę rozpoczyna ją procesować. Dla poprawnej odpowiedzi na polecenie karta SD wymaga przełączania sygnału SCLK przez co najmniej 8 taktów i obserwacji wejścia MISO. W tym czasie linia MOSI znajduje się z wysokim poziomie a CS w poziomie niskim. Długość odpowiedzi wynosi zwykle 8-bitów lub 40-bitów w zależności od użytej komendy. Detekcja otrzymywanej wiadomości jest możliwa dzięki temu że każda otrzymywana wiadomość rozpoczyna się od zera. Natomiast gdy karta nie wysyła odpowiedzi linia MISO jest ustawiona jako logiczna jedynka. Warto zauważyć że odpowiedź na każdą komendę jest wysyłane przez kartę kilka cykli SCLK później. Jeśli oczekiwana odpowiedź nie zostanie odebrana w ciągu 16 cykli zegara po wysłaniu polecenia resetu, polecenie to należy wysłać ponownie.

# Architektura projektowanych modułów

## Kontroler SPI

## Moduł sterownika kart pamięci

## Moduł testujący

# Symulacja i testowanie sterownika karty SD

## Symulacja kontrolera SPI

## Sposób testowania układu

## Dodatkowo użyte moduły

## Otrzymane wyniki testów

# Podsumowanie