Specyfikacja spi\_exe\_unit\_1

Imię: Konrad

Nazwisko: Krupski

Numer albumu: 310729

# Opis sygnałów

## wejścia

**i\_rst**

Sygnał resetu modułu spi\_exe\_unit

**i\_sclk**

Sygnał zegara modułu spi\_exe\_unit, który zapewnia odpowiednie sterowania slave

**i\_mosi**

MASTER OUT SLAVE IN. Jednobitowy, sygnał z danymi pochodzącymi od mastera

**i\_cs**

CHIP SELECT.

## wyjścia

**o\_miso**

Jednobitowy sygnał, którym przesyłane są dane wyjściowe z slave

## Parametry

**NUM**

Parametr pomocniczy, który służy do określenia liczby bitów, na których pracuje moduł watchdog

**N**

Parametr lokalny, służący do określenia liczby bitów danych, na których ma pracować exe\_unit

**BITS**

Parametr, służący do określenia liczby bitów danych, na których ma pracować slave

**STATES\_NUM**

Parametr lokalny, służący do określenia liczby stanów automatu zastosowanego w spi\_exe\_unit

**STATE\_READY**

Stan oznaczający gotowość slave do otrzymania danych od mastera. Dodatkowo w tym stanie do modułu watchdog wpisywana jest odpowiednia ilość cykli pobudzenia

**STATE\_LOAD**

Stan, w którym dane z zmiennej s\_wyniki są przypisywane do odpowiednich zmiennych pomocniczych, które następnie przypisywane są do rejestrów. Dodatkowo w tym stanie blokowany jest zapis nowej ilości cykli do układu watchdog, co pozwala na jego odpowiednią pracę.

**STATE\_FLAGS**

Stan, w którym przypisywane są wartości wyjść modułu exe\_unit do odpowiednich zmiennych pomocniczych w celu późniejszego zapisania ich do rejestrów.

**STATE\_SEND**

Stan, w którym wartości zapisane w rejestrach są podawane na równoległe wyjście slave’a.

## Opis sygnałów pomocniczych

S\_transfer – sygnał pozwalający na pracę shifter

S\_bit – sygnał przechowujący pojedynczy bit, który jest podawany na wyjście slave

S\_bit\_next - sygnał przechowujący kolejną wartość s\_bit

Reg\_argA – rejestr sygnału i\_argA modułu exe\_unit , przechowujący wartość zmiennej, na której moduł exe\_unit będzie wykonywał operacje.

Reg\_argB - rejestr sygnału i\_argB modułu exe\_unit, przechowujący wartość zmiennej, na której moduł exe\_unit będzie wykonywał operacje.

Reg\_oper - rejestr sygnału i\_oper modułu exe\_unit, przechowujący numer operacji, którą ten moduł ma wykonać

Reg\_results - rejestr sygnału o\_results modułu exe\_unit, przechowujący wynik operacji wykonanej przez ten moduł

Reg\_flags - rejestr flag modułu exe\_unit, w którym przechowywane są wartości poszczególnych flag zwracanych przez ten moduł

s\_argA\_next – sygnał przechowujący kolejną wartość sygnału reg\_argA

s\_argB\_next - sygnał przechowujący kolejną wartość sygnału reg\_argB

s\_oper\_next - sygnał przechowujący kolejną wartość sygnału reg\_oper

s\_results\_next - sygnał przechowujący kolejną wartość sygnału s\_results

s\_flags\_next - sygnał przechowujący kolejną wartość sygnału s\_flags

s\_results – sygnał przechowująca wartość, która ma zostać zapisana do rejestru reg\_results

s\_flags – sygnał przechowująca wartość, która ma zostać zapisana do rejestru reg\_flags

s\_wyniki – sygnał przechowujący ciąg 20 bitów otrzymanych z mastera, które są przypisywane do odpowiednich sygnałów pomocniczych. Wyjście równoległe z shifter’a.

s\_watchdog\_we – zezwolenie na wpis do modułu watchdog

s\_wrt – sygnał pozwalający do wpis równoległy/szeregowy do shifter’a

s\_data – sygnał służący do zapisywania wyników rejestrów i argumentów

s\_state – sygnał przechowujący wartość mówiącą o aktualnym stanie slave

s\_state\_next – sygnał przechowujący wartość kolejnego stanu slave.

## Instancjonowane moduły

### shifter

Moduł jest odpowiedzialny za przesuwanie bitów otrzymanych z master’a. Slave otzymuje kolejno pojedyncze bity od master’a i dzięki temu modułowi są one pojedynczo wsuwane i przypisywane do sygnału pomocniczego s\_wyniki.

### watchdog

Moduł, które zadaniem jest pobudzenie slave’a do pracy na 20-bitowym ciągu. Watchdog ma 20 cykli, dzięki którym możliwe jest zapisanie danych do odpowiednich rejestrów i wykonanie na nich odpowiednich operacji.

### exe\_unit

Moduł, którego zdaniem jest wykonanie operacji na danych otrzymanych z master’a. Przyjmuje on odpowiednio argument i\_argA, i\_argB, i\_oper oraz zwraca wyniki w postaci o\_result, o\_flags. Jego szczegółowa specyfikacja jest opisana w dokumentacji projektu 1.

# Algorytm pracy spi\_exe\_unit\_1

W pierwszym stanie pracy spi\_exe\_unit\_1 zapisywana jest odpowiednia ilość cykli do modułu watchdog sterującym tą jednostką. Dodatkowo ustawiane są odpowiednie wartości sygnałów zezwalające do transfer danych oraz zmianę stanu do stanu numer 2.

W kolejnym stanie pracy slave’a blokowany jest zapis nowej ilości cykli do modułu watchdog. Dodatkowo do zmiennych pomocniczych s\_argA\_next, s\_argB\_next, s\_oper\_next, s\_results\_next, s\_flags\_next, przypisywane są dane otrzymane z master’a. Stan ten jest powtarzany do momenty, gdy watchdog zgłosi koniec pracy poprzez sygnał s\_inter.

W kolejnym stanie pracy slave’a, przypisywane są do odpowiednich zmiennych pomocniczych s\_flags\_next, s\_results, wyniki obliczeń zwrócone przez moduł exe\_unit.

Ostatnim stanem pracy slave’a jest stan, w którym otrzymane wartości wyników oraz danych wejściowych są przypisywane do wyjścia równoległego i cała informacja jest zwracana do jednostki master.

## Dane podane do testowania

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Lp. | I\_argA | I\_argB | I\_oper | I\_result | I\_flags |
| 1 | 0100 | 0001 | 0000 | 0101 | 0000 |
| 2 | 1010 | 0101 | 0010 | 1111 | 0001 |
| 3 | 1100 | 0011 | 0001 | 0000 | 0000 |
| 4 | 1000 | 0000 | 0111 | 0011 | 0000 |
| 5 | 0010 | 0010 | 0011 | 1000 | 0110 |
| 6 | 0010 | 1000 | 0100 | 0000 | 0000 |
| 7 | 1010 | 0000 | 0101 | 1111 | 0001 |
| 8 | 0001 | 0101 | 0110 | 0101 | 0000 |
| 9 | 1101 | 0010 | 1011 | 1011 | 0100 |
| 10 | 1101 | 0010 | 1000 | 1010 | 0000 |

Dane uzyskane przy pracy spi\_exe\_unit zostały przetworzone poprawnie.