C2000 Piccolo LanuchPad (4)

Łatwa obsługa szyny SPI

Moduł peryferyjny SPI układu procesorowego F2802x/3x Piccolo ma prostą budowę, jednak jego wykorzystanie sprawia zazwyczaj kłopoty. Być może przyczyną jest jakby „podwójne" włączanie układów FIFO tego modułu peryferyjnego. I niezbyt czytelny opis w dokumentacji producenta.

W artykule jest opisane ćwiczenie praktyczne z zastosowaniem biblioteki *driverlib* pakietu programowego controlSUITE oraz środowiska *Code Composer Studio* v5 i zestawu ewaluacyjnego *C2000 Piccolo LaunchPad*. Celem ćwiczenia jest poznanie i zastosowanie obsługi modułu peryferyjnego SPI układu procesorowego serii F2802x Piccolo. Zastosowano przykładowy program z projektu *Example_F2802xSpi_FFDLB* pakietu controlSUITEv3. Ćwiczenie jest zorganizowane tak, że działania są wykonywane w kolejnych punktach i krokach uzupełnionych o szczegółowe opisy.

Do wykonania ćwiczenia jest potrzebny komputer z zainstalowanym (darmowym) oprogramowaniem:

- Środowisko *Code Composer Studio* v5.3.0 firmy Texas Instruments [2, 4, 8, 9, 11]. Umożliwia tworzenie programów przeznaczonych dla procesorów serii Piccolo TMS320F2802x.
- Pakiet programowy controlSUITE v3.1.3 firmy Texas Instruments [2, 4, 8, 9, 11]. Zawiera oprogramowanie „firmware", biblioteki, opisy zestawów sprzętowych oraz projekty przykładowe dla wszystkich serii procesorów rodziny C2000.

Platforma sprzętowa obejmuje następujące elementy:

- Zestaw ewaluacyjny *C2000 Piccolo LaunchPad* firmy Texas Instruments z układem procesorowym Piccolo TMS320F28027 firmy Texas Instruments (zawiera kabel USB-A USB-mini) [1, 7].
- Moduł sprzętowy KAmodEXP1 firmy Kamami [13] z układem scalonym MCP23S08 ekspandera szyny SPI firmy Microchip [15].
- Przewody połączeniowe, standard złącza IDC (np. CAB_A firmy Kamami) [14].

Instalowanie i użytkowanie środowiska CCSv5 oraz pakietu programowego controlSUITEv3 zostało opisane w artykule [2].

W folderze *C:\home_dir* komputera zostanie utworzony nowy folder *work_SPI*. Wymagane są prawa dostępu (zapisu i modyfikacji) dla tej ścieżki dyskowej. Możliwe jest umieszczenie foldera *home_dir* na innym wolumenie dyskowym z prawami dostępu.

Opisy

Dane techniczne i parametry elektryczne modułu peryferyjnego SPI układu procesorowego serii F2802x Piccolo są zamieszczone w dokumencie Texas Instruments TMS320F28027, TMS320F28026, TMS320F28023, TMS320F28022, TMS320-F28021, TMS320F280200, Piccolo Microcontrollers, Data Sheet [7].

Opis modułu peryferyjnego SPI układu procesorowego serii F2802x Piccolo jest zamieszczony w dokumencie TMS320x2802x, 2803x Piccolo Serial Peripheral Interface (SPI) Reference Guide [12].

Dokładne omówienie budowy modułu peryferyjnego SPI układu procesorowego serii F2802x Piccolo jest zamieszczone w książce Henryk A. Kowalski, „Procesory DSP dla praktyków", BTC, Warszawa, 2011 [5].

Dokładne omówienie przykładowego projektu *Example_F2802xSpi_FFDLB* jest zamieszczone w książce Henryk A. Kowalski, „Procesory DSP w przykładach", BTC, Warszawa, 2012 [6].

Dodatkowe materiały na CD/FTP:
ftp://ep.com.pl, user: 63241, pass: 741obq51

Dotychczas w EP na temat zestawu ewaluacyjnego C2000 Piccolo LaunchPad:
- „Zestaw ewaluacyjny C2000 Piccolo LaunchPad", EP 01/2013
- „C2000 Piccolo LanuchPad (1) – Pierwszy program w środowisku programowym CCS v5", EP 02/2013
- „C2000 Piccolo LanuchPad (2) – Łatwe programowanie z pakietem controlSUITE", EP 03/2013
- „C2000 Piccolo LanuchPad (3) – Łatwe programowanie do pamięci Flash", EP 04/2013

Pliki źródłowe:
Piccolo_F2802x_KAmodEXP1.c, Piccolo_F2802x_KAmodEXP1.h oraz Example_2802xSpi_FFDLB.c (po modyfikacji)

Skonfigurowanie modułu KAmodEXP1

W module KAmodEXP1 (**fotografia 1**) został zastosowany układ scalony ekspandera MCP23S08 firmy Microchip pracujący w standardzie łącza SPI [15]. Do poprawnej pracy programu ćwiczenia wymagana jest podstawowa (standardowa) konfiguracja przełączników płytki drukowanej modułu KAmodEXP1 [13]:

- Założona zwora JP0 w pozycji „0" (JP0.1-JP0.2). Oznacza to bit adresowy A0=0.
- Założona zwora JP1 w pozycji „0" (JP1.1-JP1.2). Oznacza to bit adresowy A1=0.

Oznacza to ustawienie adresu 0x40.

Dołączanie modułu KAmodEXP1 do zestawu C2000 Piccolo LaunchPad

Dołącz moduł KAmodEXP1 do zestawu ewaluacyjnego C2000 Piccolo LaunchPad.

Uwaga! Połączenia należy wykonywać bez włączonego zasilania, czyli przy odłączonym kablu USB. Najlepiej najpierw połączyć masę obu płytek drukowanych. Zmniejszy to niebezpieczeństwo uszkodzenia układów ze względu na ładunki elektrostatyczne. Połączenia należy wykonywać przewodami z końcówkami zgodnymi ze standardem złącza IDC [14].

Kod głównej pętli for programu

```
// infinite loop
for(;;) {
 KAmodEXP1_WriteReg(DEV_ADDR, GPIO,   0x00);
 DELAY_US(1000000); // 1000ms
 // effect 1
 for(Output = 1; Output != 0x20; Output <<= 1)
 {
    KAmodEXP1_WriteReg(DEV_ADDR, GPIO, Output);    //
    DELAY_US(700000);
    Input= KAmodEXP1_ReadReg(DEV_ADDR, GPIO);
 }

 // effect 2
 for(Output = 1; Output < 0xF; Output <<= 1)
 {
    Output |= 1;
    KAmodEXP1_WriteReg(DEV_ADDR, GPIO, Output);
    DELAY_US(700000);
 }
 // effect 3
 for(Counter = 0; Counter < 4; Counter ++)
 {
    Output = 0x55;
    KAmodEXP1_WriteReg(DEV_ADDR, GPIO, Output);
    DELAY_US(700000);
    Output = 0xAA;
    KAmodEXP1_WriteReg(DEV_ADDR, GPIO, Output);
    DELAY_US(700000);
 }
} //for(;;)
```

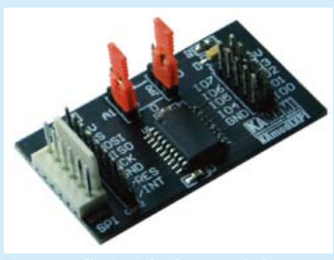
Tabela 1. Dołączenie modułu KAmodEXP1 do zestawu C2000 Piccolo LaunchPad

Płytka KAmodEXP1	Płytka C2000 Piccolo LaunchPad
Con2.6 (GND)	J3.2 (GND)
Con2.1 (V+)	J3.1 (+3V3)
Con2.5 (SCLK)	J1.7 (SPICLK/ GPIO18)
Con2.3 (MOSI)	J2.6 (SPISIMOA/ GPIO16)
Con2.4 (MISO)	J2.7 (SPISOMIA/ GPIO17)
Con2.2(SS)	J2.2 (GPIO19/SPISEA)
Con2.8 (INT)	J1.5 (GPIO34)
Con2.7 (RES)	J1.8 (AIO2/ ADCINA2)
Con3.1 (IO0)	J6.1 (GPIO0)
Con3.3 (IO1)	J6.2 (GPIO1)
Con3.5 (IO2)	J6.3 (GPIO2)
Con3.7 (IO3)	J6.4 (GPIO3)

Należy podać masę GND, zasilanie 3.3V trzy sygnały łącza SPI, dwa sygnały sterujące (SS, RES) i cztery sygnały danych (tab.1). Wybór skonfigurowania wyprowadzeń układu procesorowego F28027 Piccolo został zaznaczony kolorem.

Konfigurowanie zestawu C2000 Piccolo LaunchPad

Po zainstalowaniu środowiska CCSv5 [2] można pierwszy raz dołączyć zestaw ewaluacyjny C2000 Piccolo LaunchPad [1] kablem USB do wolnego portu USB komputera. System Windows automatycznie rozpoznaje układ. Zostaną zainstalowane sterowniki systemu Windows dla emulatora XDS100v2 [6]. Należy poczekać aż system potwierdzi, że sprzęt jest gotowy do pracy.



Fotografia 1. Płytka modułu KAmodEXP1

Do poprawnej pracy programu przykładowego jest wymagana podstawowa (standardowa) konfiguracja przełączników płytki drukowanej zestawu [1]:

- Założone zwory JP1 („3V3"), JP3 („5V") i JP2 („GND"). Oznacza to zasilanie układu procesorowego z gniazda USB.
- Przełącznik S1 („Boot") skonfigurowany następująco: S1.1 - do góry (ON), S1.2 - do góry, S1.3 - do góry. W praktyce oznacza to bootowanie z pamięci Flash.
- Przełącznik S4 („Serial") skonfigurowany w pozycji do góry (ON). Oznacza dołączenie portu UART układu procesorowego do układu emulatora, a tym samym do wirtualnego portu COM na PC.

Pliki źródłowe

Do wykonania ćwiczenia potrzebny jest kod źródłowy zawarty w dodatkowych plikach *Piccolo_F2802x_KAmodEXP1.c oraz Piccolo_F2802x_KAmodEXP1.h*. Ponadto, zostanie zmodyfikowany kod w głównym pliku *Example_2802xSpi_FFDLB.c* projektu przykładowego. Pliki są zamieszczone na załączonym CD.

Uruchomienie środowiska CCSv5

W oknie *Workspace* wpisz ścieżkę i nazwę foldera roboczego. Powinna być ona krótka i musi być zlokalizowana w miejscu, dla którego są uprawnienia dostępu (zapisu). Dla indywidualnej pracy jest proponowana ścieżka <C:/*home_dir*>. Dla tego ćwiczenia proponowana jest nazwa foldera *work_SPI*.

Po kliknięciu na przycisk *OK* okna *Workspace Launcher* otwierane jest okno startowe środowiska CCSv5 (i ładowane są poszczególne elementy środowiska). Można to obserwować na pasku postępu. Może to trwać dosyć długo i należy koniecznie poczekać na zakończenie inicjalizacji środowiska przed rozpoczęciem dalszej pracy.

Projekty przykładowe pakietu controlSUITE

W oknie *TI Resource Explorer* perspektywy *CCS Edit* jest pokazywana strona *Welcome* (w html). Zawiera ona graficznie menu główne. Istotne informacje są zgrupowane na stronie *Home*. Można ją otworzyć po kliknięciu oknie *TI Resource Explorer* na ikonkę *Home*. Po kliknięciu na odnośnik *Examples* jest pokazywane po lewej stronie okna drzewo dokumentacji i dostępnych projektów przykładowych. Jeśli jest pokazywana tylko jedna linia controlSUITE z gałęzią *English* to udostępnia ona dokumentację pakietu. Aby dodać przykłady należy na stronie *Home* kliknąć na odnośnik *Configure Resource Explorer*. W oknie dialogowym *Package Configuration* trzeba kliknąć na *Add*, wskazać folder C:\ti\controlSUITE i kliknąć *OK*. Nazwa controlSUITE pojawi się w oknie wyboru i należy kliknąć *OK*. Po dłuższej chwili pojawi się w drzewie okna *TI Resource Explorer* druga linia controlSUITE zawierająca pozycje: development kits, device_support oraz libs.

Zastosowanie projektu Example_ F2802xSpi_FFDLB

Dla pracy z rodziną F2802x Piccolo rozwiń w oknie *TI Resource Explorer* drugą pozycję *controlSUITE*. Następnie należy rozwinąć drzewo *controlSUITE* → *device_support* → *f2802x* → *v210* → *f2802x_examples*. Potem trzeba kliknąć na nazwę wybranego projektu ***Example_F2802xSpi_FFDLB***. W prawym oknie zostanie wyświetlona instrukcja jak krok po kroku zbudować i uruchomić projekt.

Krok1: Importowanie projektu Example_ F2802xSpi_FFDLB do CCS

Krok1 umożliwia zaimportowanie wybranego projektu do CCSv5.

Kliknij na odnośnik kroku 1.

Po poprawnym wykonaniu importowania drzewo projektu pojawia się w oknie *Project Explorer* i zielony znaczek jest pokazywany na prawo od linii nazwy kroku.

Projekt Example_F2802xSpi_FFDLB został zaimportowany z kopiowaniem pliku *Example_2802xSpi_FFDLB.c* do foldera roboczego projektu.

Krok2: Budowanie projektu Example_F2802xSpi_FFDLB

Krok2 umożliwia wykonanie budowania wybranego projektu.

Kliknij na odnośnik kroku 2.

W oknie *Console* pokazywane są bieżące informacje o postępie budowania. W oknie *Problems* pokazywane są opisy błędów, ostrzeżeń i informacji. Po poprawnym wykonaniu budowania zielony znaczek jest pokazywany na prawo od linii nazwy kroku.

W oknie *Project Explorer* rozwiń drzewo projektu i kliknij na jego nazwę. Został zbudowany projekt w konfiguracji budowania o nazwie RAM.

Budowanie projektu *Example_F2802xSpi_FFDLB*zostało zakończone poprawnie. Został utworzony wynikowy plik binarny *Example_F2802xSpi_FFDLB*.out. Zostały jednak zgłoszone ostrzeżenia. Na razie są one nieistotne.

Krok3: Definiowanie konfiguracji sprzętowego systemu docelowego

Krok3 umożliwia zdefiniowanie konfiguracji sprzętowego systemu docelowego dla projektu. Pole *Connection* pokazuje typ „none”.

Kliknij na odnośnik kroku 3.

W oknie dialogowym *Debugger Configuration* rozwiń listę i wybierz pozycję ***Texas Instruments XDS100v2 USB Emulator***. Kliknij OK. Pole *Connection* pokazuje teraz typ *Texas Instruments XDS100v2 USB Emulator*. Zielony znaczek „check mark” jest pokazywany na prawo od linii nazwy kroku. Utworzony plik konfiguracji sprzętowej TMS320F28027.ccxml jest teraz pokazany w gałęzi */targetConfigs* drzewa projektu w oknie *Project Explorer*. Jest on ustawiony jako Active/Default (aktywny i domyślny).

Krok4: Uruchamianie sesji debugowej dla projektu Example_F2802xSpi_FFDLB

Krok4 umożliwia uruchomienie sesji debugowej dla projektu. Dotychczas praca środowiska CCSv5 nie wymagała fizycznej obecności sprzętu docelowego. Wykonanie kroku 4 wymaga wcześniejszego dołączenia zestawu ewaluacyjnego *C2000 Piccolo LaunchPad* do komputera z zainstalowanym środowiskiem CCSv5.

Kliknij na odnośnik kroku 4.

Kliknięcie na odnośnik kroku 4 powoduje automatyczne rozpoczęcie sesji debuggowej – podobnie jak po przyciśnięciu przycisku *Debug*.

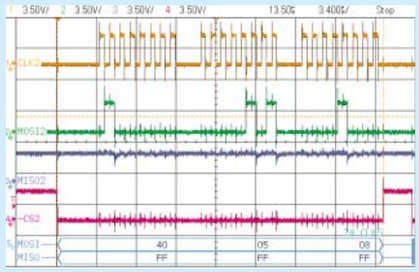
Wgląd w projekt Example_F2802xSpi_FFDLB

Zauważ, że praca programu została zatrzymana na pierwszej linii kodu funkcji *main()*.

Otwórz okno *Disassembly* z menu *View* → *Disassembly*. W tym oknie można dokładnie zobaczyć jak naprawdę pracuje układ procesorowy.

Zapoznaj się z komentarzem na początku pliku ***Example_2802xSpi_FFDLB.c.***

Krótki opis projektu przykładowego oraz założenia i wymagania sprzętowe są zamieszczone na początku głównego pliku każdego projektu przykładowego z pakietu programowego controlSUITE.



Rysunek 2. Wpisywanie przez moduł peryferyjny SPI układu procesorowego F28027 Piccolo nowej zawartości 0x08 do rejestru konfiguracyjnego IOCON (adres 0x05) układu scalonego MCP23S08 pod adresem 0x40 (zapis) na szynie SPI

Kliknij na przycisk *Resume* na pasku narzędziowym okna *Debug*. Program zostanie uruchomiony i nic się nie dzieje.

Kliknij na przycisk *Suspend* na pasku narzędziowym okna *Debug*.

Jeśli wykonanie programu zostało zatrzymane wewnątrz funkcji *main()* to kliknij na przycisk pracy krokowej *Step Into* na pasku narzędziowym okna *Debug*. W przeciwnym przypadku przejdź do następnego punktu.

Jeśli wykonanie programu przeszło do wewnątrz wywoływanej w linii 130 funkcji z biblioteki *driverlib* to jest wyświetlany komunikat o niemożliwości uzyskania dostępu do pliku.

Problem jest spowodowany wygenerowaniem biblioteki *driverlib* w lokalizacji innej niż standardowa ścieżka pakietu controlSUITE (dokładne omówienie w [20]). Można doraźnie zaradzić problemom dostępu. Kliknij na przycisk *Locate File*. Wskaż ścieżkę C:\ti\controlSUITE\device_support\f2802x\v210\f2802x_common\source.

Kliknij na przycisk *Step Return* na pasku narzędziowym okna *Debug*.

Problem zatrzymania wykonania programu w linii 130 jest spowodowany przez niepełną i nieprawidłową inicjalizację modułu peryferyjnego SPI układu procesorowego w projekcie przykładowym. Linia kodu *SPI_enable(mySpi);* powoduje zwolnienie modułu peryferyjnego SPI ze stanu Reset i musi być ostatnią operacją przy inicjalizacji modułu peryferyjnego SPI.

Zmiana nazwy projektu F2802xSpi_KAmodEXP1

Przełącz się do perspektywy *CCS Edit*.

W oknie *Project Explorer* kliknij prawym przyciskiem myszy na linię nazwy projektu *Example_F2802xSpi_FFDLB*.

Z podręcznego menu wybierz z menu *Rename*.

Wpisz nową nazwę *F2802xSpi_KAmodEXP1* i kliknij *OK*. Zostanie również zmieniona nazwa foldera projektu w folderze roboczym projektu.

Wykonaj samo budowanie projektu (bez ponownego startowania sesji debugowej). Kliknij na przycisk *Build*. Zauważ brak pytania czy załadować plik wynikowy kodu.

Jest to spowodowane zmianą ustawienia ścieżki dostępu do pliku wynikowego. Nazwa foldera roboczego jest inna.

Przełącz się do perspektywy *CCS Debug*.

Załaduj kod wynikowy do układu procesorowego F28027 Piccolo.

Kliknij na przycisk *Load*. W oknie *Load Program* kliknij na przycisk *Browse project*.

Kliknij na projekt *F2802xSpi_KAmodEXP1* oraz przycisk *OK*.

W oknie *Load Program* kliknij na przycisk *Browse* i w folderze RAM wybierz plik *Example_F2802xSpi_FFDLB.out* i kliknij przycisk *Otwórz* a potem *OK*.

Zauważ, że nazwa pliku z kodem źródłowym oraz z wynikowym nie została zmieniona.

Inicjalizowanie modułu peryferyjnego SPI

Zmodyfikuj kod funkcji Spi_init() do postaci pokazanej w pliku *Example_2802xSpi_FFDLB.c* (po modyfikacji). Skopiuj całą zawartość kodu funkcji *Spi_init()*.

W pierwszej linii kodu funkcji Spi_init() jest włączany zegar systemowy dla modułu peryferyjnego SPI.

CLK_enableSpiaClock(myClk);

Dalej została dodana linia

SPI_reset(mySpi);

Wymusza ona programowo wejście modułu peryferyjnego SPI w stan Reset. Powoduje zainicjowanie niektórych znaczników na wartości domyślne. Przed zmianą konfiguracji jest wymagane wprowadzenie do w stan Reset. Dopiero po zakończeniu konfigurowania modułu peryferyjnego można go wyprowadzić z tego stanu, czyli uruchomić.

Gdy w trakcie uruchamiania programu zostanie wykonane polecenie Restart to nie powoduje ono zmiany ustawień rejestrów modułów peryferyjnych na domyślne. Nie robi tego również Reset programowy (patzr powyżej). Dlatego nie można zakładać, że mają one domyślną zawartość. Lepiej jest w trakcie inicjalizacji modułu peryferyjnego ustawić wszystkie istotne bity rejestrów sterujących w odpowiedni stan.

Linia

SPI_resetRxFifo(mySpi);

powoduje wyzerowanie wskaźnika bufora FIFO odbiornika i zatrzymanie go w stanie Reset.

W następnej linii

SPI_setCharLength(mySpi, SPI_CharLength_8_Bits);

została zmieniona długość słowa transmisji na 8 bitów.

Rejestr danych (i FIFO) modułu peryferyjnego SPI układu procesorowego Piccolo F2802x jest 16-to bitowy. Bity są wysuwane na szynę szeregową zaczynając od najbardziej znaczącego. Dlatego bajt nadawany trzeba przesunąć w lewo o 8 pozycji bitowych przed wstawieniem do rejestru danych. W przypadku odbioru należy wyzerować najbardziej znaczące 8 bitów słowa rejestru odbioru.

W kolejnej linii

SPI_setMode(mySpi, SPI_Mode_Master);

moduł peryferyjny ustawiany jest do pracy jako master.

Kolejne dwie linie

SPI_setClkPhase(mySpi, SPI_ClkPhase_Delayed);

SPI_setClkPolarity(mySpi, SPI_ClkPolarity_OutputRisingEdge_InputFallingEdge);

określają typ ustawień zegara modułu peryferyjnego SPI. Został zastosowany typ – zbocze narastające z opóźnieniem.

Dana wyjściowa jest wystawiana pół cyklu przed pierwszym zboczem narastającym zegara SPICLK oraz na zboczu opadającym następnych cykli zegara. Dana wejściowa jest zatrzaskiwana na zboczu narastającym zegara SPICLK. Wyprowadzenie SPICLK jest wyzerowane, gdy nie są wysyłane dane. Można to zaobserwować na rys. 1.

W kolejnej linii

SPI_enableTx(mySpi);

ustawiane jest zezwolenie na transmisję. Gdy moduł peryferyjnego SPI jest skonfigurowany jako master to domyślnie SPISIMOA jest ustawione w stan wysokiej impedancji. Dopiero ustawienie zezwolenia na transmisję powoduje dołączenie wyprowadzenia do modułu peryferyjnego.

W kolejnej linii

SPI_setBaudRate(mySpi, SPI_BaudRate_1_MBaud);

Ustawiana jest szybkość transmisji. Wynosi ona:

SPI Baud Rate = LSPCLK/4 , dla SPIBRR = 0, 1, 2

SPI Baud Rate = LSPCLK/(SPIBRR +1), dla SPIBRR = 3 do 127

Domyślna częstotliwość zegara systemowego SYSCLKOUT = 60MHz a szybkość zegara LSPCLK wynosi SYSCLKOUT/4. Dla zastosowanych ustawień szybkość zegara transmisji jest nieco większa niż deklarowany 1MHz (patrz rys. 1).

W kolejnej linii

SPI_setTxDelay(mySpi, 1);

jest określany odstęp (opóźnienie) pomiędzy wysyłaniem słowa z bufora FIFO nadajnika do bufora TXBUF. Opóźnienie jest definiowane jako liczba cykli zegara SPICLK modułu peryferyjnego SPI. Bardzo to poprawia czytelność obrazu transmisji w trakcie uruchamiania z użyciem oscyloskopu (patrz rys. 1).

Zakomentowana jest linia

//SPI_enableLoopBack(mySpi);

włączenia trybu pracy modułu peryferyjnego SPI z zapętleniem. Wtedy sygnał wyjściowy jest wewnętrznie dołączony do wejścia modułu peryferyjnego. Wyprowadzenia układu scalonego są odłączone. Tryb jest przydatny w trakcie testowania poprawności pracy modułu peryferyjnego.

Kolejne linie kodu konfigurują pracę FIFO nadajnika i odbiornika. W linii

SPI_enableChannels(mySpi);

jest włączana praca nadajnika i odbiornika.

W linii

SPI_enableFifoEnh(mySpi);

jest włączana praca FIFO nadajnika.

Linie

SPI_resetRxFifo(mySpi);

SPI_resetRxFifo(mySpi);

powodują wyzerowanie wskaźnika bufora FIFO nadajnika i odbiornika oraz zatrzymanie ich w stanie Reset.

Linie

SPI_enableTxFifo(mySpi);

SPI_enableRxFifo(mySpi);

powodują uruchomienie pracy FIFO nadajnika i odbiornika.

W linii

SPI_setPriority(mySpi, SPI_Priority_FreeRun);

jest ustawiana praca swobodna (free) po przejściu układu procesorowego w tryb debugowania (np. dla obsługi pułapki). Oznacza dalsze kontynuowanie pracy.

Ostatnia linia po zakończeniu konfigurowania modułu peryferyjnego

SPI_enable(mySpi);

powoduje wyprowadzenie modułu peryferyjnego SPI ze stanu RESET, czyli uruchomienie jego pracy.

Próby projektu F2802xSpi_KAmodEXP1

Usuń definicję funkcji spi_fifo_init(); na końcu pliku.

Zakomentuj linię (120) wywołania funkcji

//spi_fifo_init(); // Initialize the SPI FIFOs

W funkcji Spi_init() odkomentuj linię kodu SPI_enableLoopBack(mySpi);

Wykonaj samo budowanie projektu (bez ponownego startowanie sesji debugowania). Kliknij na przycisk *Build*. Na pytanie czy załadować plik wynikowy kodu przyciśnij przycisk *Yes*.

Przejdź do perspektywy CCS Debug.

Ustaw pułapkę w linii (138) kodu *sdata++;* Dwukliknij na linię po lewej stronie od numeru linii kodu.

Kliknij na przycisk *Resume*.

Zatrzymanie wykonania programu w tej linii oznacza poprawną pracę. Moduł peryferyjny SPI wysłał jedno słowo danych na szynę SPI i odebrał jedno słowo danych o zawartości zgodnej ze słowem wysłanym (zapętlenie wewnętrzne).

Dalsza rozbudowa projektu F2802xSpi_KAmodEXP1

Uzupełnij skonfigurowanie wyprowadzeń GPIO.

Po istniejącym kodzie konfigurowania GPIO dodaj nowy kod.

```
// KAmodEXP1 modifications
GPIO_setPullUp(myGpio, GPIO_Number_19, GPIO_PullUp_Enable);
GPIO_setQualification(myGpio, GPIO_Number_19, GPIO_Qual_Sync);
GPIO_setHigh(myGpio, GPIO_Number_19);
GPIO_setMode(myGpio, GPIO_Number_19, GPIO_19_Mode_GeneralPurpose);
GPIO_setDirection(myGpio, GPIO_Number_19, GPIO_Direction_Output);
// GPIO as monitor of MCP23S08 I/O port state
GPIO_setMode(myGpio, GPIO_Number_0, GPIO_0_Mode_GeneralPurpose);
GPIO_setMode(myGpio, GPIO_Number_1, GPIO_1_Mode_GeneralPurpose);
GPIO_setMode(myGpio, GPIO_Number_2, GPIO_2_Mode_GeneralPurpose);
GPIO_setMode(myGpio, GPIO_Number_3, GPIO_3_Mode_GeneralPurpose);
GPIO_setDirection(myGpio, GPIO_Number_0, GPIO_Direction_Input);
GPIO_setDirection(myGpio, GPIO_Number_1, GPIO_Direction_Input);
GPIO_setDirection(myGpio, GPIO_Number_2, GPIO_Direction_Input);
GPIO_setDirection(myGpio, GPIO_Number_3, GPIO_Direction_Input);
// GPIO as external Reset signal of MCP23S08
GPIO_setPullUp(myGpio, GPIO_Number_34, GPIO_PullUp_Enable);
GPIO_setHigh(myGpio, GPIO_Number_34);
GPIO_setMode(myGpio, GPIO_Number_34, GPIO_34_Mode_GeneralPurpose);
GPIO_setDirection(myGpio, GPIO_Number_34, GPIO_Direction_Output);
```

Sprawdzić czy nie został w trakcie zmian wprowadzony błąd kodu. Zbuduj projekt. W perspektywie *CCS Edit* kliknij na przycisk *Build*. Nie ładuj kodu.

Do foldera projektu *C:\home_dir\work_SPI\F2802xSpi_KAmodEXP1* dodaj (skopiuj) plik nagłówkowy *Piccolo_F2802x_KAmodEXP1.h* oraz plik kodu *Piccolo_F2802x_KAmodEXP1.c*.

Sprawdź w perspektywie *CCS Edit* czy oba pliki zostały pokazane w drzewie projektu w oknie *Project Explorer*.

Sprawdzić czy nie został w trakcie zmian wprowadzony błąd kodu. Zbuduj projekt. W perspektywie *CCS Edit* kliknij na przycisk *Build*. Nie ładuj kodu.

Bez pośrednio po linii kodu *spi_init();* wstaw nowy kod

```
DELAY_US(3);
// KAmodEXP1 modifications
KAmodEXP1_Reset();
DELAY_US(1);
KAmodEXP1_Init();
KAmodEXP1_WriteReg(DEV_ADDR, IOCON, 0x08);
DELAY_US(1);
KAmodEXP1_WriteReg(DEV_ADDR, IODIR, 0x00);
DELAY_US(1);
KAmodEXP1_WriteReg(DEV_ADDR, GPIO, 0x00);
DELAY_US(1);
```

Bezpośrednio za blokiem instrukcji *include* na początku pliku wstaw nowy fragment

```
// KAmodEXP1 modifications
#include “KAmodEXP1.h”
#define DEV_ADDR 0 // Hardware addres of KAmodEXP1 (JP0=0 JP1=0)
//#define DEV_ADDR 1
//#define DEV_ADDR 2
//#define DEV_ADDR 3
```

Na początku funkcji main() dodaj blok deklaracji

```
// KAmodEXP1 modifications
Uint16 Counter;
Uint16 Output = 1, Input;
```

Zbuduj projekt. W perspektywie *CCS Edit* kliknij na przycisk *Build*. Nie ładuj kodu.

Sprawdzić czy nie został w trakcie zmian wprowadzony błąd kodu

Wymień kod całej głównej pętli for(;;). Wstaw w to miejsce nowy kod (ramka)

Uwaga! Ważne. W funkcji *Spi_init()* zakomentuj linię kodu *SPI_enableLoopBack(mySpi);*

Zbuduj projekt. Kliknij na przycisk *Build*. Na pytanie czy załadować plik wynikowy kodu przyciśnij przycisk *Yes*. Zostały zgłoszone nowe ostrzeżenia. Na razie są one nieistotne.

Zapoznaj się z kodem funkcji w pliku *F2802xSpi_KAmodEXP1.c* oraz definicjami w pliku *F2802xSpi_KAmodEXP1.h*.

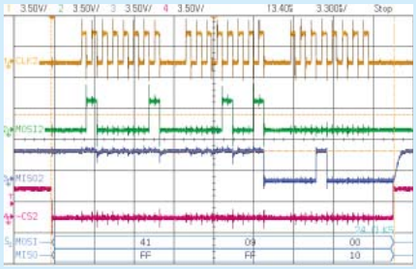
Wpis danych do modułu KAmodEXP1

Funkcja

Uint16 KAmodEXP2_WriteReg(Uint16 idx, Uint16 RegAddr, Uint16 RegValue)

realizuje wpis danych do wskazanego rejestru układu scalonego MCP23S08 firmy Microchip zastosowanego w module KAmodEXP1 firmy Kamami.

Układ scalony MCP23S08 jest urządzeniem slave na szynie SPI. Jego adres slave zawiera pięć bitów stałych oraz dwa bity adresowe definiowane przez wyprowa-



Rysunek 3. Odczytywanie przez moduł peryferyjny SPI układu procesorowego F28027 Piccolo zawartości (0x01) z do rejestru portu GPIO (adres 0x09) układu scalonego MCP23S08 pod adresem 0x40 (odczyt) na szynie SPI

dzenia. Bit najmniej znaczący całego bajtu adresowego definiuje typ operacji: 0 – oznacza zapis , 1- oznacza odczyt danych.

Parametr *idx* podaje adres układu scalonego MCP23S08 na szynie SPI. Adres ten jest definiowany poziomem logicznym wejść A0, A1 układu scalonego. Tak zdefiniowany zmienny numer jest przesuwany w lewo o jeden bit i konkatenowany ze stałą częścią adresu na szynie wynoszącą 0x40. Na koniec jest uzupełniany bitem typu operacji zapisu (0).

Parametr *RefAddr* podaje adres rejestru wewnętrznego układu scalonego MCP23S08. Jest on numerem od 0x00 do 0x0A. Parametr *RegValue* przekazuje wartość bajtu do wpisania do rejestru.

Kompletna sekwencja wpisu danych do modułu KAmodEXP1 ma szynie SPI jest pokazana na **rysunek 2**.

W funkcji *KAmodEXP2_WriteReg* najpierw jest ustawiany aktywny (niski) poziom sygnału /CS. Jest to realizowane poprzez ustawienie poziomu wyjściowego wyprowadzenie GPIO skonfigurowane jako wyjście.

Do FIFO nadajnika (rejestr SPIDAT) jest wpisywany bajt adresu 0x40 (zapis) układu scalonego MCP23S08 .

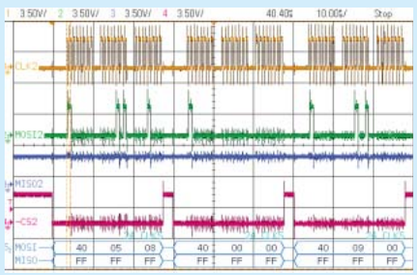
Dane do nadawania muszą być dosunięte lewostronnie, ponieważ rejestr danych (i FIFO) modułu peryferyjnego SPI układu procesorowego Piccolo F2802x jest 16-to bitowy. Bity są wysuwane na szynę szeregową zaczynając od najbardziej znaczącego.

Do FIFO nadajnika jest wpisywany bajt adresu rejestru konfiguracyjnego IOCON (adres 0x05) układu.

Następnie do FIFO nadajnika jest wpisywany bajt danych (0x08).

Po wpisaniu danych do FIFO nadajnika moduł peryferyjny SPI układu procesorowego Piccolo F2802x zaczyna automatycznie wystawiać sygnał zegarowy i wysyłać 8 bitów danych na szynę SPI. Następne dane (jeśli są) oczekują w FIFO nadajnika. Po czasie odstęp (opóźnienia) zdefiniowanym w module SPI jest wysyłane następne słowo danych.

Gdy wszystkie trzy bajty danych zostały wysłane to w FIFO odbiornika czekają trzy słowa. Zawierają one przypadkową zawartość.



Rysunek 4. Transmisja konfigurowania rejestru konfiguracyjnego IOCON (adres 0x05), rejestru kierunku IODIR (adres 0x00) oraz rejestru portu GPIO (adres 0x09) układu scalonego MCP23S08 pod adresem 0x40 (zapis) na szynie SPI.

Program oczekuje na ustawienie znacznika liczby odebranych słów na 3.

Wtedy jest ustawiany nieaktywny (wysoki) poziom sygnału /CS i jest wykonywane kasowanie ustawienia znaczników FIFO nadajnika i odbiornika.

Odczyt danych z modułu KAmodEXP1

Funkcja *Uint16 KAmodEXP1_ReadReg(Uint16 idx, Uint16 RegAddr)* realizuje wpis odczyt ze wskazanego rejestru układu scalonego. Parametr *idx* podaje adres układu scalonego MCP23S08 na szynie SPI. Adres ten jest definiowany poziomem logicznym wejść A0, A1 układu scalonego. Tak zdefiniowany zmienny numer jest przesuwany w lewo o jeden bit i konkatenowany ze stałą częścią adresu na szynie wynoszącą 0x40. Na koniec jest uzupełniany bitem typu operacji odczytu (1).

Parametr *RefAddr* podaje adres rejestru wewnętrznego układu scalonego MCP23S08. Jest on numerem od 0x00 do 0x0A.

Kompletna sekwencja odczytu danych z modułu KAmodEXP1 ma szynie SPI jest pokazana na **rysunek 3**.

W funkcji *KAmodEXP2_ReadReg* najpierw jest ustawiany aktywny (niski) poziom sygnału /CS. Jest to realizowane poprzez ustawienie poziomu wyjściowego wyprowadzenie GPIO skonfigurowane jako wyjście.

Do FIFO nadajnika (rejestr SPIDAT) wpisywany jest bajt adresu 0x41 (odczyt) układu scalonego MCP23S08. Dane do nadawania muszą być dosunięte lewostronnie.

Do FIFO nadajnika jest wpisywany bajt adresu rejestru portu GPIO (adres 0x09) układu MCP23S08.

Następnie do FIFO nadajnika wpisywany jest bajt danych o dowolnej zawartości (0x00).

Po wpisaniu danych do FIFO nadajnika moduł peryferyjny SPI układu procesorowego Piccolo F2802x zaczyna automatycznie wystawiać sygnał zegarowy i wysyłać 8 bitów danych na szynę SPI. Następne dane (jeśli są) oczekują w FIFO nadajnika. Po czasie odstęp (opóźnienia) zdefiniowanym w module SPI jest wysyłane następne słowo danych.

Gdy wszystkie trzy bajty danych zostały wysłane to w FIFO odbiornika czekają trzy słowa. Dwa pierwsze odczytane słowa zawierają przypadkową zawartość. Ostatnie odczytane z FIFO odbiornika słowo 16-to bitowe zawiera przysłaną przez układ MCP23S08 zwartość 8-bitowego rejestru portu. Najbardziej znaczące 8 bitów zawiera przypadkową zawartość. Dlatego należy wyzerować najbardziej znaczące 8 bitów.

Program oczekuje na ustawienie znacznika liczby odebranych słów na 3.

Wtedy wykonywane są trzy odczyty danych z FIFO odbiornika.

Ustawiany jest nieaktywny (wysoki) poziom sygnału /CS.

Następnie wykonywane jest kasowanie ustawienia znaczników FIFO nadajnika i odbiornika.

Inicjalizacja scalonego ekspandera MCP23S08

Pełna sekwencja inicjalizacji scalonego ekspandera MCP23S08 dla szyny SPI wykonywana w projekcie *F2802xSpi_KAmodEXP1* jest pokazana na **rysunek 4**.

Najpierw do rejestru konfiguracyjnego IOCON (adres 0x05) układu jest wpisywany bajt danych (0x08). Są to ustawienia zgodne z domyślnymi po operacji Reset.

Następnie do rejestru kierunku IODIR (adres 0x00) układu jest wpisywany bajt danych (0x00). Oznacza to ustawienie wszystkich bitów portu układu MCP23S08 jako wyjścia.

Na koniec do rejestru portu GPIO (adres 0x09) jest wpisywana nowa zawartość 0x00.

Uruchamianie projektu F2802xSpi_KAmodEXP1

Przełącz się do perspektywy *CCS Debug*. Kliknij na przycisk *Resume*.

Program zostanie uruchomiony. Obserwuj diody D5-D2 na zestawie ewaluacyjnym C2000 Piccolo LaunchPad.

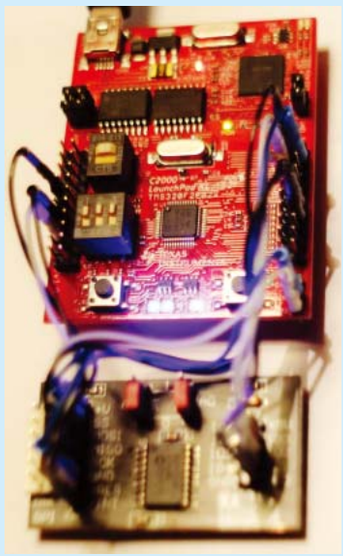
Diody D5/4/3/2 są dołączone do wyjść bramek cyfrowych typu „open-drain" dołączonych do wyprowadzeń GPIO3/2/1/0 układu procesorowego. Diody świecą dla niskiego poziomu logicznego na wyprowadzeniu. Do skonfigurowanych jako wejścia wyprowadzeń GPIO3/2/1/0 układu procesorowego dołączone są wyjścia bitów IO3/2/1/0 portu układu scalonego MCP23S08 (**tabela 1**). Tak więc diody na zestawie ewaluacyjnym C2000 Piccolo pokazują stan bitów portu ekspandera szyny.

Diody prezentują trzy wzory świecenia:

Efekt1: Do portu wpisywane są zera (świecą). Następnie jedynka jest wpisywana na pozycję zerową i przesuwana w lewo.

Efekt2: Jedynki są wsuwane do bufora od pozycji zerowej w lewo.

Efekt3: Na przemian jest wyświetlany wzór: świeci-nie świeci oraz jego zanegowanie.



Fotografia 5. Zestaw C2000 Piccolo LaunchPad z dołączonym modułem KAmodEXP1.

Możliwości rozbudowy projektu F2802xSpi_KAmodEXP1

Projekt realizuje z odpytywaniem obsługę zależności czasowych pracy modułu peryferyjnego SPI układu procesorowego F28027 Piccolo. Jest to bardzo przydatny sposób podczas prezentacji działania modułu peryferyjnego SPI i debugowania jego pracy.

Projekt w obecnej wersji realizuje minimalną obsługę funkcji udostępnianych przez moduł KAmodEXP1 z układem scalonego ekspandera MCP23S08. Można wykorzystać zdolność zgłaszania przez układ ekspandera MCP23S08 zmian stanu wejść portu. Na jego wyprowadzeniu INT pojawia się wtedy poziom niski. Jego stan można odczytywać z użyciem wyprowadzenia GPIO34 układu procesorowego F28027 Piccolo.

Na **fotografii 5** pokazano zestaw C2000 Piccolo LaunchPad z dołączonym modułem KAmodEXP1.

Bibliografia

[1] Henryk A. Kowalski, „Zestaw ewaluacyjny C2000 Piccolo LaunchPad", EP 01/2013

[2] Henryk A. Kowalski, Henryk A. Kowalski, „C2000 Piccolo LanuchPad – pierwszy program w środowisku programowym Code Composer Studio v5" EP/02/2013.

[3] Code Composer Studio, strona produktu *http://www.ti.com/ccs*

[4] controlSUITE, strona pobierania *http://www.ti.com/tool/controlsuite*

[5] Henryk A. Kowalski, Procesory DSP dla praktyków, BTC, Warszawa, 2011 *http://ii.pw.edu.pl/kowalski/dsp/book/*

[6] Henryk A. Kowalski, Procesory DSP w przykładach, BTC, Warszawa, 2012 *http://ii.pw.edu.pl/kowalski/dsp/book/*

[7] TMS320F28027, TMS320F28026, TMS320F28023, TMS320F28022, TMS320-F28021, TMS320F280200, Piccolo Microcontrollers, Data Sheet, SPRS523I, 31 Jul 2012

[8] F2802x Firmware Development Package USER'S GUIDE v. 210 [f2802x-FRM-EX-UG.pdf], pakiet controlSUITE

[9] F2802x Peripheral Driver Library USER'S GUIDE v. 210 [f2802x-DRL-UG.pdf], pakiet controlSUITE

[10] TMS320x2802x/TMS320F2802xx Piccolo System Control and Interrupts Reference Guide (Rev. C) 29 Oct 2009, [SPRUFN3C]

[11] controlSUITE Getting Started Guide (Rev. B), SPRUGU2B , 09 June 2011

[12] TMS320x2802x, 2803x Piccolo Serial Peripheral Interface (SPI) Reference Guide [SPRUG71B]

[13] KAmodEXP1, Adresowalny ekspander GPIO z interfejsem SPI www.kamami.pl

[14] CAB_A, Przewody połączeniowe, *www.kamami.pl*

[15] MCP23008/MCP23S08, 8-Bit I/O Expander with Serial Interface, Data Sheet (08/24/2007) *www.microchip.com*

Henryk A. Kowalski
kowalski@ii.pw.edu.pl