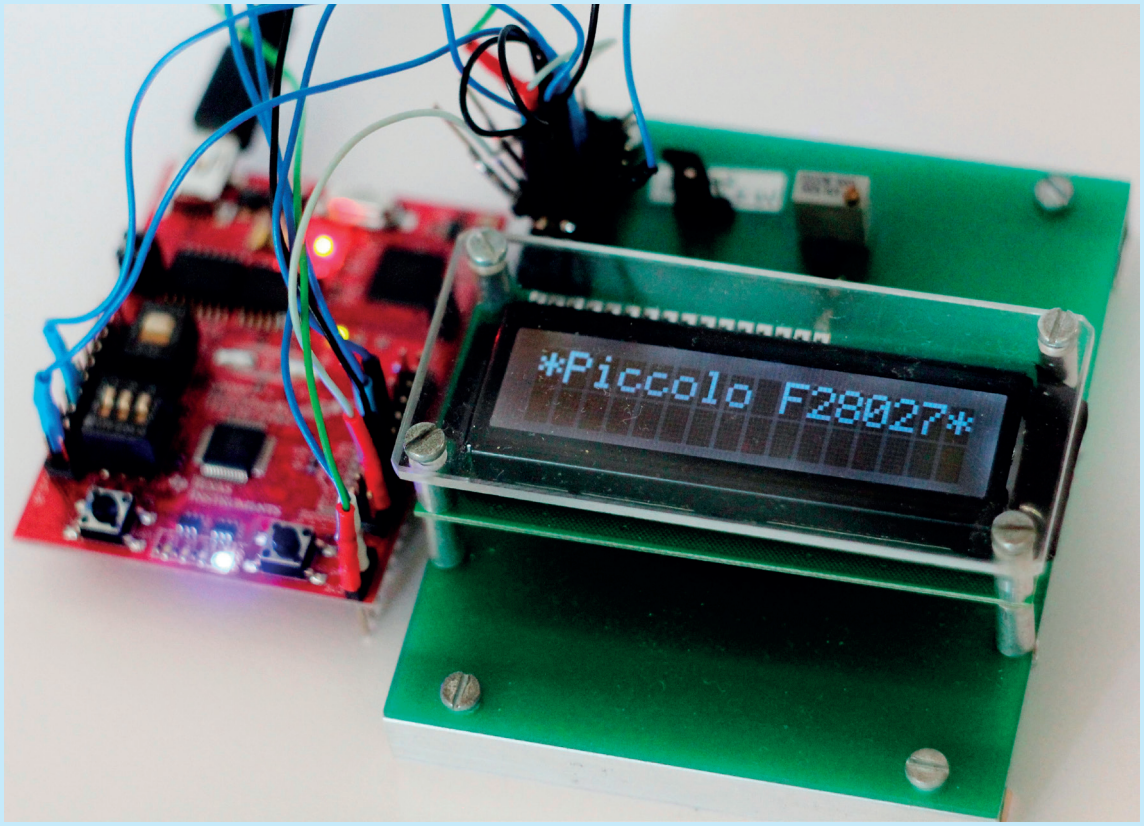


C2000 Piccolo LanuchPad (7)

Łatwa obsługa wyświetlacza LCD

Wyświetlacze alfanumeryczne LCD ze sterownikiem typu HD44780 są praktycznie standardem przemysłowym. Sterownik może pracować z równoległym interfejsem 8-bitowym lub 4-bitowym. Można też zrezygnować z odczytu stanu wyświetlacza i zastosować obsługę tylko zapisu danych. Uproszczenie to wymaga zastosowania maksymalnych czasów odstępu pomiędzy operacjami. Ale pozwala ograniczyć konieczną do sterowania wyświetlaczem liczbę wyprowadzeń procesora.

Układ procesorowy serii Piccolo F28027PT ma 22 wyprowadzenia GPIO (w obudowie 48-wyprowadzeniowej), z czego 4 są multipleksowane jako wyprowadzenia portu emulacyjnego JTAG. Ma również 13 wejść analogowych, z czego 6 może być konfigurowanych jako wyprowadzenia cyfrowe AIO2/4/6/10/12/14.

Do tworzenia w środowisku CCSv5 programów przeznaczonych dla procesorów rodziny Piccolo F2802x potrzebny jest pakiet programowy controlSUITE. Zawiera on firmware, biblioteki, opisy zestawów sprzętowych oraz projekty przykładowe dla wszystkich serii procesorów rodziny C2000.

Tekst zawiera opis następujących działań:

- Dołączenie i skonfigurowanie zestawu C2000 Piccolo LaunchPad.
- Dołączanie modułu LCD1602 do zestawu C2000 Piccolo LaunchPad.
- Uruchomienie i praca ze środowiskiem CCSv5.
- Importowanie i budowanie projektu Example_F2802xGpioSetup.
- Dołączanie pliku do projektu.
- Zmiana nazwy projektu na F2802xSpi_KAmodEXP1.

Dodatkowe informacje:

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
- „C2000 Piccolo LanuchPad (4) – Łatwa obsługa szyny SPI", EP 05/2013
- „C2000 Piccolo LanuchPad (5) – Łatwa obsługa szyny I²C", EP 07/2013
- C2000 Piccolo LanuchPad (6) – Łatwa inicjalizacja systemowa procesora serii Piccolo F2802x ", EP 09/2013

Pliki źródłowe na CD:
Piccolo_F2802x_HD44780.c, Piccolo_F2802x_HD44780.h oraz Example_2802xGpioSetup.c (po modyfikacji)

- Dodanie obsługi wyświetlacza LCD do projektu F2802x_LCD.
- Ćwiczenie praktyczne z projektem obsługi wyświetlacza LCD ze sterownikiem typu HD44780 przy zastosowaniu układu procesorowego serii Piccolo F2802x pracującego na zestawie ewaluacyjnym *C2000 Piccolo LaunchPad*.

Celem ćwiczenia jest poznanie sposobu obsługi wyświetlacza LCD ze sterownikiem typu HD44780 przy zastosowaniu układu procesorowego serii Piccolo F2802x oraz użyciu biblioteki *driverlib* pakietu programowego controlSUITEv3 i środowiska *Code Composer Studio* v5. Zastosowano przykładowy projekt *Example_F2802xGpioSetup* z tego pakietu. Ćwiczenie jest zorganizowane tak, że działania są wykonywane w kolejnych punktach i krokach uzupełnionych o opisy.

Konfiguracja sprzętowa i programowa do wykonania ćwiczenia

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

- Środowisko *Code Composer Studio* v5.3.0 firmy Texas Instruments [1, 16]. Umożliwia tworzenie w środowisku CCSv5 programów przeznaczonych dla procesorów serii Piccolo TMS320F2802x.
- Pakiet programowy controlSUITEv3.1.2 firmy Texas Instruments [2, 16, 17]. Zawiera oprogramowanie „firmware", biblioteki, opisy zestawów sprzętowych oraz projekty przykładowe dla wszystkich serii procesorów rodziny C2000.

Platforma sprzętowa wymaga dwóch elementów:

- Zestaw ewaluacyjny *C2000 Piccolo LaunchPad* firmy Texas Instruments z układem procesorowym Piccolo TMS320F28027 firmy Texas Instruments (zawiera kabel USB-A USB-mini) [8, 15]
- Alfanumeryczny wyświetlacz LCD 2x16 ze złączem szpilkowym ze sterownikiem HD44780, np. LCD1602 z firmy Kamami [9].
- Przewody połączeniowe, standard złącza IDC (np. CAB_A firmy Kamami) [10]

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

Pliki źródłowe

Do wykonania ćwiczenia potrzebny jest kod źródłowy zawarty w dodatkowych plikach *Piccolo_F2802x_HD44780.h* oraz *Piccolo_F2802x_HD44780.c* (są zamieszczone na CD). Ponadto, zostanie zmodyfikowany kod w głównym pliku *Example_2802xGpioSetup.c* projektu przykładowego *Example_F2802xGpioSetup*.

Na stronie *http://www.kamami.pl/index.php?ukey=product&productID=135566* jest dostępny przykład dla modułu LCD w języku C dla mikrokontrolerów AVR [11].

Opisy

Dokładne omówienie budowy modułu GPIO układu procesorowego serii Piccolo F2802x jest zamieszczone w książce [13]. Dane techniczne modułu LCD z firmy Kamami są zamieszczone na stronie [9]. Opis układu sterownika HD44780 firmy Hitachi jest szeroko dostępny, np. w książce [12]. Dokładne omówienie budowy układu procesorowego serii Piccolo F2802x jest zamieszczone w książce Henryk A. Kowalski, „Procesory DSP dla praktyków", Wydawnictwo BTC, 2011 [13]. Dokładne omówienie zestawu ewaluacyjnego *C2000 Piccolo LaunchPad* jest zamieszczone w artykule Henryk A. Kowalski, „Zestaw ewaluacyjny C2000 Piccolo LaunchPad" Elektronika Praktyczna 01/13 [15]. Dokładne omówienie środowiska CCSv5 oraz pakietu controlSUITEv3 jest zamieszczone w artykule Henryk A. Kowalski„C2000 Piccolo LanuchPad (1) – Pierwszy program w środowisku programowym CCSv5", Elektronika Praktyczna 02/2013 [16].

Instalowanie i użytkowanie środowiska CCSv5.3.0 [1] oraz pakietu programowego controlSUITEv3.1.2 [2] zostało opisane w artykułach [16, 17].

Podłączenie i skonfigurowanie zestawu C2000 Piccolo LaunchPad

Po zainstalowaniu środowiska CCSv5 [1, 16] można pierwszy raz dołączyć zestaw ewaluacyjny C2000 Piccolo LaunchPad [15] kablem USB do wolnego portu USB komputera. System Windows automatycznie rozpoznaje układ. Zostaną zainstalowane sterowniki systemu Windows dla emulatora XDS100v2 [13]. Należy poczekać aż system potwierdzi, że sprzęt jest gotowy do pracy.

Do poprawnej pracy programu przykładowego wymagana jest podstawowa (standardowa) konfiguracja przełączników płytki drukowanej zestawu [15]:

- Założone zwory JP1 („3V3"), JP3 („5V") i JP2 („GND"). Oznacza to zasilanie układu procesorowego Piccolo F28027 z gniazdka USB.
- Przełącznik S1 („Boot") skonfigurowany następująco: S1.1 – do góry (ON), S1.2 – do góry, S1.3 – do góry. W praktyce oznacza to bootowanie układu procesorowego Piccolo F28027 z pamięci Flash.
- Przełącznik S4 („Serial") skonfigurowany na obu pozycjach do góry (ON). Oznacza to dołączenie portu UART układu procesorowego Piccolo F28027 do układu emulatora, a tym samym do wirtualnego portu COM na komputerze PC.

Zestaw ewaluacyjny jest dostarczany z wpisanym do pamięci Flash układu procesorowego Piccolo F28027 programem przykładowym *Example_F2802xLaunchPad-Demo*. Program automatycznie zaczyna pracować po dołączeniu zestawu do portu USB [15].

Dołączanie modułu LCD1602 do zestawu C2000 Piccolo LaunchPad

Dołącz moduł LCD1602 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 [15].

Należy podać masę GND, zasilanie 3,3 V, dwa sygnały sterujące (Rs, E) i cztery sygnały danych (**tabela 1**).

Tabela 1. Podłączenie modułu LCD do zestawu C2000 Piccolo LaunchPad dla projektu F2802x_LCD	
Moduł LCD1602	**Płytka C2000 Piccolo LaunchPad**
1 Vss	J3.1 (+3.3V)
2 Vdd	J3.2 (GND)
3 Vo	(typowy układ regulacji)
4 Rs	J1.9 (AIO10/ADCINB2)
5 R/W	J3.3 (GND)
6 E	J1.6 (AIO4/ADCINA2)
7 D0	–
8 D1	–
9 D2	–
10 D3	–
11 D4	J6.5 (GPIO4/EPWM3A)
12 D5	J6.6 (GPIO5/EPWM3B)
13 D6	J2.8 (GPIO6/EPWM4A)
14 D7	J2.9 (GPIO7/EPWM4B)
15 LED (+)	J1.1 (+3.3V)
16 LED (–)	J5.2 (GND)
Wybór skonfigurowania wyprowadzeń układu procesorowego Piccolo F28027 został zaznaczony kolorem.	

Uruchamianie środowiska CCSv5

Po uruchomienia środowiska CCSv5 pokazywane jest okno edycyjne *Workspace Launcher* ustawiania lokalizacji foldera roboczego.

1. W oknie *Workspace* należy wpisać ścieżkę dla lokalizacji folderu (*workspace*) roboczego projektu. Można ją też wskazać przy użyciu standardowego przycisku *Browse* systemu Windows. Odznaczenie (wyłączenie) opcji *Use this as the default and do not ask again* oznacza pracę z osobnym folderem roboczym. Folder z projektem można umieścić w folderze roboczym. Ale nie odwrotnie. Przy ponownym uruchomieniu środowiska CCSv5 pokazywana jest w oknie *Workspace Launcher* ścieżka lokalizacji folderu roboczego używana przy ostatnim zamknięciu CCSv5.

W oknie *Workspace* wpisz ścieżkę i nazwę foldera roboczego.

Powinna być ona krótka i musi być zlokalizowana na dysku w miejscu, dla którego są uprawnienia dostępu (zapisu). Dla indywidualnej pracy proponowana jest ścieżka *„C:/home_dir”*. Dla tego ćwiczenia proponowana jest nazwa foldera *„/work_LCD”*. Można umieścić folder *home_dir* na innym wolumenie dyskowym z prawami dostępu.

Po kliknięciu na przycisk *OK* okna *Workspace Launcher* jest otwierane okno startowe środowiska CCSv5 (i ładowane są poszczególne elementy środowiska). Można to obserwować na pasku postępu w prawym dolnym rogu okna.

Przy uruchamianiu środowiska sprawdzana jest w sieci dostępność aktualizacji. Środowisko CCSv5 przy pierwszym uruchamianiu może pobierać sporo aktualizacji. Może to trwać dosyć długo i należy koniecznie poczekać przed rozpoczęciem dalszej pracy na zakończenie inicjalizacji środowiska i pokazanie okna *Welcome* lub *Home*. Jeśli zostały wykryte i pobrane z sieci nowe lub aktualniejsze komponenty to wyświetlane jest okno wyboru komponentów do aktualizacji. Po kliknięciu przycisku *Finish* wyświetlane jest okno informacyjne. Zainstalowanie nowych komponentów wymaga zamknięcia i ponownego uruchomienia środowiska CCSv5.

Projekty przykładowe pakietu controlSUITE

W oknie *TI Resource Explorer* perspektywy *CCS Edit* pokazywana jest strona *Welcome* (w html). Zawiera ona graficznie menu główne. Istotne informacje są zgrupowane na stronie *Home*. Można ją otworzyć po kliknięciu w oknie *TI Resource Explorer* na ikonkę Home.

Po kliknięciu na odnośnik *Examples* pokazywane jest po lewej stronie okna drzewo dokumentacji i dostępnych projektów przykładowych.

Jeśli pokazywana jest tylko jedna linia controlSUITE z gałęzią *English* to udostępnia ona tylko dokumentację pakietu. Aby dodać dostęp do przykładowych projektów należy na dole strony *Home* kliknąć na odnośnik *Configure Resource Explorer*. W oknie dialogowym *Package Configuration* trzeba kliknąć na *Add*. Następnie trzeba wskazać folder *C:\ti\controlSUITE* i kliknąć *OK*. Nazwa controlSUITE pojawi się w oknie wyboru. 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_F2802xGpioSetup

2. Dla pracy z rodziną układów procesorowych Piccolo F2802x rozwiń w oknie *TI Resource Explorer* drugą pozycję *controlSUITE*. Następnie rozwiń drzewo *controlSUITE* → *device_support* → *f2802x* → *v210* → *f2802x_examples*. Potem kliknij na nazwę wybranego projektu ***Example_F2802xGpioSetup***.

W prawym oknie zostanie wyświetlona instrukcja jak krok po kroku zbudować i uruchomić projekt.

Krok1: Importowanie projektu Example_F2802xGpioSetup do CCSv5

Krok1 umożliwia zaimportowanie wybranego projektu do CCSv5.

3. W oknie *TI Resource Explorer* kliknij na odnośnik kroku 1.

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

Projekt *Example_F2802xGpioSetup* został zaimportowany z kopiowaniem projektu i pliku ***Example_2802xGpioSetup.c*** do foldera roboczego projektu.

Kliknięcie na odnośnik kroku 2 powoduje automatyczne budowanie projektu – podobnie jak po przyciśnięciu przycisku *Build*. Powinno to spowodować zapisanie wszystkich plików ze zmianami przed rozpoczęciem budowania projektu. Tak się typowo dzieje w przypadku użycia przycisku *Build*. Czasami występują jednak kłopoty z plikami nagłówkowymi. Jednak wydaje się, że w przypadku wykonywania kroku 2 zapisywanie nie jest wykonywane.

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

Budowanie projektu ***Example_F2802xGpioSetup*** zostało zakończone poprawnie. Został utworzony wynikowy plik binarny Example_2802xGpioSetup.out (zobacz okno *Console*). Zostały jednak zgłoszone cztery

ostrzeżenia (zobacz okno *Problems*). Na razie są one nieistotne.

Krok3: Definiowanie konfiguracji sprzętowego systemu docelowego

Krok3 umożliwia zdefiniowanie konfiguracji sprzętowej systemu docelowego dla projektu. Na początku pole *Connection* pokazuje typ „none".

5. W oknie *TI Resource Explorer* kliknij na odnośnik kroku 3.

W oknie dialogowym *Debugger Configuration* rozwiń listę wyboru.

6. Wybier0z pozycję ***Texas Instruments XDS100v2 USB Emulator***. Kliknij *OK*.

W oknie *TI Resource Explorer* pole *Connection* pokazuje teraz typ *Texas Instruments XDS100v2 USB Emulator*. Zielony znaczek pokazywany jest 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_F2802xGpioSetup

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 [17].

7. W oknie *TI Resource Explorer* kliknij na odnośnik kroku 4.

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

Postęp działania środowiska CCSv5 można obserwować na pasku stanu w prawym dolnym rogu okna. Może to trwać dosyć długo i należy koniecznie poczekać przed rozpoczęciem dalszej pracy na zakończenie ładowania kodu i pokazania się okna perspektywy *CCS Debug*.

Dołączanie pliku do projektu

Skorzystanie z podglądu stanu pól bitowych rejestrów sterowania modułów peryferyjnych wymaga dołączenia do projektu pliku definicyjnego struktur modelu bezpośredniego dostępu do rejestrów. Zestawienie nazw zmiennych (struktur) udostępnianych przez plik jest podane w tab.2.12 (str. 32) dokumentacji pakietu firmware [6]. Dla modułu GPIO udostępniane są struktury rejestrów sterowania: *GpioDataRegs*, *GpioCtrlRegs* oraz *GpioIntRegs*.

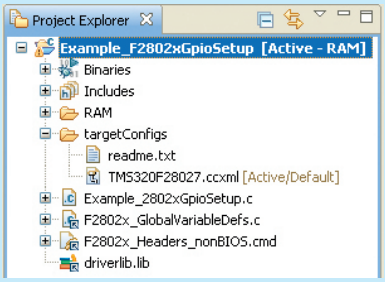
8. Przełącz się do perspektywy *CCS Edit*. W oknie *Project Explorer* kliknij prawym klawiszem myszy na linię nazwy projektu *Example_F2802xGpioSetup*. Z podręcznego menu wybierz *Add Files*.

9. Wybierz ścieżkę *C:\ti\controlSUITE\device_support\f2802x\v210\f2802x_headers\source*.

10. Zaznacz plik *F2802x_GlobalVariableDefs.c* i kliknij na *Otwórz*. W oknie *File Operation* zaznacz opcję *Link to files*. Kliknij *OK*. Plik zostanie dołączony (nie dodany) do projektu.

11. Wykonaj samo budowanie projektu (bez ponownego startowanie sesji debugowej). Kliknij na przycisk *Build*. Nie używaj przycisku *Debug*.

12. Na pytanie czy załadować plik wynikowy kodu przyciśnij przycisk *Yes*. Poczekaj na załadowanie programu. Przełącz się na perspektywę CCS Debug.



Rysunek 1. Drzewo projektu Example_F2802xGpioSetup po dołączeniu pliku F2802x_GlobalVariableDefs.c

13. Zauważ w oknie Project Explorer, że plik *F2802x_GlobalVariableDefs.c* jest dołączony (nie wstawiony) do projektu. Informuje o tym znak strzałki nałożony na ikonkę pliku (rys.1).

14. Przełącz się do perspektywy *CCS Debug*. Zauważ w oknie edytora, że praca program została zatrzymana na pierwszej linii kodu funkcji *main()*.

Wgląd w projekt Example_F2802xGpioSetup

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

16. Otwórz okno *Disassembly* z menu *View* → *Disassembly*. W tym oknie można dokładnie zobaczyć jak naprawdę pracuje układ procesorowy Piccolo F28027.

W projekcie są zamieszczone dwie funkcje Gpio_setup1 i Gpio_setup2, które realizują dwa trochę różne warianty skonfigurowania wyprowadzeń GPIO.

Zmiana nazwy projektu na F2802xSpi_KAmodEXP1

17. Przełącz się do perspektywy *CCS Edit*.

18. W oknie *Project Explorer* kliknij prawym klawiszem myszy na linię nazwy projektu *Example_F2802xGpioSetup*.

19. Z podręcznego menu wybierz z menu *Rename*.

20. Wpisz nową nazwę *F2802x_LCD* i kliknij *OK*. Zostanie również zmieniona nazwa folderu projektu w folderze roboczym projektu.

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

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

22. Przełącz się do perspektywy *CCS Debug*.

Załaduj kod wynikowy do układu procesorowego F28027 Piccolo.

23. Kliknij na przycisk *Load*. W oknie *Load Symbols* kliknij na przycisk *Browse project*.

24. Kliknij na projekt *F2802x_LCD* oraz przycisk *OK*.

25. W oknie *Load Program* kliknij na przycisk *Browse* i w folderze RAM wybierz plik *Example_F2802xGpioSetup.out* i kliknij przycisk *Otwórz* a potem *OK*.

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

Zmodyfikowanie projektu F2802x_LCD

26. Do foldera projektu *C:\home_dir\work_SPI\F2802xSpi_KAmodEXP1* dodaj plik nagłówkowy *Piccolo_F2802x_HD44780.h*.

Listing 1. Kod do wpisania przed funkcją *main()*
```
//---------------------------------------
// Display output state on LCD
//---------------------------------------
void SetOutScreen(unsigned char Out)
{
  unsigned char i;
  char  String[5];
  LCD_GoTo(0,1);  //Second line
  LCD_WriteText("Out:");  //Display string
  LCD_WriteText(Hex(Out, String));  //Display hex value
  LCD_WriteData(' ');  //Display space
  LCD_WriteBinary(Out, 8);  //Display binary value
  for(i = 0; i < 50; i ++)  //Some delay
  DELAY_US(10);  //_delay_ms(10);
}
```

Listing 2. Nowy kod konfigurowania GPIO
```
// LCD modifications
// GPIO as output
GPIO_setDirection(myGpio, GPIO_Number_4, GPIO_Direction_Output);
GPIO_setDirection(myGpio, GPIO_Number_5, GPIO_Direction_Output);
GPIO_setDirection(myGpio, GPIO_Number_6, GPIO_Direction_Output);
GPIO_setDirection(myGpio, GPIO_Number_7, GPIO_Direction_Output);

GPIO_setHigh(myGpio, GPIO_Number_4);
GPIO_setHigh(myGpio, GPIO_Number_5);
GPIO_setHigh(myGpio, GPIO_Number_6);
GPIO_setHigh(myGpio, GPIO_Number_7);

GPIO_setMode(myGpio, GPIO_Number_4, GPIO_0_Mode_GeneralPurpose);
GPIO_setMode(myGpio, GPIO_Number_5, GPIO_0_Mode_GeneralPurpose);
GPIO_setMode(myGpio, GPIO_Number_6, GPIO_0_Mode_GeneralPurpose);
GPIO_setMode(myGpio, GPIO_Number_7, GPIO_0_Mode_GeneralPurpose);

// Enable an ADCIN pins as AIO digital outputs, set them high
EALLOW;
GpioDataRegs.AIOSET.bit.AIO4 = 1;   // Set high output latch
GpioCtrlRegs.AIOMUX1.bit.AIO4  = 0;  // ADCINA4 = AIO4
GpioCtrlRegs.AIODIR.bit.AIO4 = 1;   // AIO4 = output
GpioDataRegs.AIOSET.bit.AIO10 = 1;   // Set high output latch
GpioCtrlRegs.AIOMUX1.bit.AIO10  = 0;  // ADCINB2 = AIO10
GpioCtrlRegs.AIODIR.bit.AIO10 = 1;   // AIO10 = output
EDIS;
```

Listing 3. Kod do wpisania po ciele funkcji *main()*
```
// Configure GPIO 0-3 as outputs
GPIO_setMode(myGpio, GPIO_Number_0, GPIO_0_Mode_GeneralPurpose);
GPIO_setMode(myGpio, GPIO_Number_1, GPIO_0_Mode_GeneralPurpose);
GPIO_setMode(myGpio, GPIO_Number_2, GPIO_0_Mode_GeneralPurpose);
    GPIO_setMode(myGpio, GPIO_Number_3, GPIO_0_Mode_GeneralPurpose);

GPIO_setDirection(myGpio, GPIO_Number_0, GPIO_Direction_Output);
GPIO_setDirection(myGpio, GPIO_Number_1, GPIO_Direction_Output);
GPIO_setDirection(myGpio, GPIO_Number_2, GPIO_Direction_Output);
GPIO_setDirection(myGpio, GPIO_Number_3, GPIO_Direction_Output);

GPIO_setMode(myGpio, GPIO_Number_12, GPIO_12_Mode_GeneralPurpose);
GPIO_setDirection(myGpio, GPIO_Number_12, GPIO_Direction_Input);
    GPIO_setPullUp(myGpio, GPIO_Number_12, GPIO_PullUp_Disable);

//Scan the LEDs until the pushbutton is pressed
while(GPIO_getData(myGpio, GPIO_Number_12) != 1)
{
  GPIO_setHigh(myGpio, GPIO_Number_0);
  GPIO_setHigh(myGpio, GPIO_Number_1);
  GPIO_setHigh(myGpio, GPIO_Number_2);
  GPIO_setLow(myGpio, GPIO_Number_3);
  DELAY_US(300000);       //

  GPIO_setHigh(myGpio, GPIO_Number_0);
  GPIO_setHigh(myGpio, GPIO_Number_1);
  GPIO_setLow(myGpio, GPIO_Number_2);
  GPIO_setHigh(myGpio, GPIO_Number_3);
  DELAY_US(300000);

  GPIO_setHigh(myGpio, GPIO_Number_0);
  GPIO_setLow(myGpio, GPIO_Number_1);
  GPIO_setHigh(myGpio, GPIO_Number_2);
  GPIO_setHigh(myGpio, GPIO_Number_3);
  DELAY_US(300000);

  GPIO_setLow(myGpio, GPIO_Number_0);
  GPIO_setHigh(myGpio, GPIO_Number_1);
  GPIO_setHigh(myGpio, GPIO_Number_2);
  GPIO_setHigh(myGpio, GPIO_Number_3);
  DELAY_US(300000);

  GPIO_setHigh(myGpio, GPIO_Number_0);
  GPIO_setHigh(myGpio, GPIO_Number_1);
  GPIO_setHigh(myGpio, GPIO_Number_2);
  GPIO_setHigh(myGpio, GPIO_Number_3);
  DELAY_US(300000);

  GPIO_setLow(myGpio, GPIO_Number_0);
  GPIO_setHigh(myGpio, GPIO_Number_1);
  GPIO_setHigh(myGpio, GPIO_Number_2);
  GPIO_setHigh(myGpio, GPIO_Number_3);
  DELAY_US(300000);
```

27. Do foldera projektu dodaj plik kodu *Piccolo_F2802x_HD44780.c*.

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

29. Bezpośrednio za blokiem instrukcji *include* na początku pliku wstaw nowy fragment
```
// LCD modifications
#include "Piccolo_F2802x_HD44780.h"
```

30. Przed kodem funkcji *main()* wstaw nowy kod pokazany na **listingu 1**.

31. Na początku funkcji *main()* dodaj blok deklaracji
```
Uint16 Counter;
Uint16 Output, Input;
Uint16 DelayTime;
```

Wyprowadzenia cyfrowe i analogowe układu procesorowego Piccolo F2802x

Wyświetlacz LCD jest typowo dołączany do układu procesorowego realizującego rozbudowaną funkcjonalność. Zestaw ewaluacyjny *C2000 Piccolo LaunchPad* udostępnia tylko ograniczoną liczbę wyprowadzeń cyfrowych układu procesorowego Piccolo F28027. Co więcej typowo są one już zastosowane ze względu na przypisane do nich funkcje modułów peryferyjnych (np. UART, SPI itd.).

Należy zauważyć dodatkowe 6 wyprowadzeń analogowych, które można konfigurować jako wyprowadzenia cyfrowe AIO2/4/6/10/12/14. Dokładne przypisanie sygnałów układu procesorowego do złącz rozszerzeń zestawu ewaluacyjnego *C2000 Piccolo LaunchPad* jest pokazane w tekście [15].

Pomimo sporej liczby funkcji biblioteka *driverlib* nie zapewnia obsługi wszystkich działań dostarczanych przez moduły GPIO. Brakuje obsługi konfigurowania wyprowadzeń analogowych układów procesorowych serii Piccolo F2802x. Dlatego ich obsługa wymaga zastosowania model bezpośredniego dostępu do rejestrów z dołączonym plikiem F2802x_GlobalVariableDefs.c.

W projekcie są zamieszczone dwie funkcje Gpio_setup1 i Gpio_setup2, które realizują dwa trochę inne warianty skonfigurowania wyprowadzeń GPIO.

Zmień skonfigurowanie wyprowadzeń GPIO.

32. Skasuj kod wywołania funkcji konfigurowania GPIO
```
#if EXAMPLE1
 // This example is a basic pinout
  Gpio_setup1();
#endif  // - EXAMPLE1
#if EXAMPLE2
 // This example is a communications pinout
```

```
    Gpio_setup2();
  #endif   // - EXAM-
PLE2
```

33. Skasuj kod funkcji Gpio_setup1 oraz Gpio_setup2.

34. W to miejsce wstaw nowy kod konfigurowania GPIO pokazany na **listingu 2**.

35. Na końcu funkcji *main()* wstaw nowy kod pokazany na **listingu 3**.

36. Wykonaj samo budowanie projektu (bez ponownego startowanie sesji debugowej). Kliknij na przycisk *Build*. Nie używaj przycisku *Debug*.

37. Na pytanie czy załadować plik wynikowy kodu przyciśnij przycisk *Yes*). Poczekaj na załadowanie programu. Przełącz się na perspektywę CCS Debug.

38. Kliknij na przycisk *Resume* na pasku narzędziowym okna *Debug*.

Świecą się po kolei cztery diody LED od lewej do prawej – wskazując na prawo na przycisk S3. Jest to cyklicznie powtarzane. Wzór jest podobny jak podczas pracy programu przykładowego wpisanego do pamięci Flash procesora. Jest jednak wykonywany znacznie szybciej. Pozwala to rozróżnić pomiędzy pracą programu własnego z pamięci RAM oraz pracą programu własnego płytki zestawu.

Listing 3. c.d.

```
  GPIO_setHigh(myGpio, GPIO_Number_0);
  GPIO_setHigh(myGpio, GPIO_Number_1);
  GPIO_setHigh(myGpio, GPIO_Number_2);
  GPIO_setHigh(myGpio, GPIO_Number_3);
  DELAY_US(300000);

  GPIO_setLow(myGpio, GPIO_Number_0);
  GPIO_setHigh(myGpio, GPIO_Number_1);
  GPIO_setHigh(myGpio, GPIO_Number_2);
  GPIO_setHigh(myGpio, GPIO_Number_3);
  DELAY_US(300000);
}

for(DelayTime = 0; DelayTime < 50; DelayTime ++)  // Some delay
DELAY_US(30000); //delay_ms(30);
LCD_GoTo(0,1);      // Clear second line
LCD_WriteText("                ");

// infinite loop
for(;;) {
  Output =0;
  Input = GPIO_getPortData(myGpio, GPIO_Port_A);
  GPIO_setPortData(myGpio, GPIO_Port_A, (Input & 0xFFF0)|(Output & 0x000F) );
  SetOutScreen(Output);
  DELAY_US(2000000);

  //------------
  // effect 1
  //------------
  for(Output = 1; Output != 0x20; Output <<= 1)
  {
    GPIO_setPortData(myGpio, GPIO_Port_A, (Input & 0xFFF0)|(Output & 0x000F) );
    SetOutScreen(Output);
    DELAY_US(1000000);  // 700ms
  }

  //------------
  // effect 2
  //------------
  for(Output = 1; Output < 0xF; Output <<= 1)
  {
     Output |= 1;
     GPIO_setPortData(myGpio, GPIO_Port_A, (Input & 0xFFF0)|(Output & 0x000F) );
     SetOutScreen(Output);
     DELAY_US(1000000);                                    // 700ms
  }

  //------------
  // effect 3
  //------------
  for(Counter = 0; Counter < 4; Counter ++)
  {
    Output = 0x55;
    GPIO_setPortData(myGpio, GPIO_Port_A, (Input & 0xFFF0)|(Output & 0x000F) );
    SetOutScreen(Output);
    DELAY_US(1000000);  // 700ms
    Output = 0xAA;
    GPIO_setPortData(myGpio, GPIO_Port_A, (Input & 0xFFF0)|(Output & 0x000F) );
    SetOutScreen(Output);
    DELAY_US(1000000);  // 700ms
  }
}    //for(;;)
```

Dodanie obsługi wyświetlacza LCD do projektu F2802x_LCD

39. W funkcji *main()*, bezpośrednio przed pętlą *while* wstaw nowy kod pokazany na listingu 4.

40. W głównej pętli *for* bezpośrednio po każdej linii wywołania funkcji *GPIO_setPortData* wstaw nową linię kodu (5 razy)

```
SetOutScreen(Output);
```

41. Wykonaj samo budowanie projektu (bez ponownego startowanie sesji debugowej). Kliknij na przycisk *Build*. Nie używaj przycisku *Debug*.

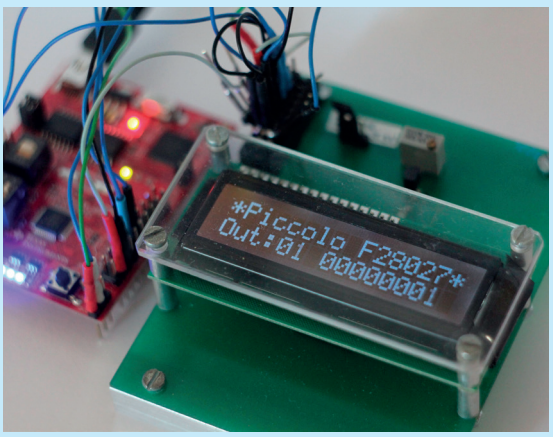
42. Na pytanie czy załadować plik wynikowy kodu przyciśnij przycisk *Yes*. Poczekaj na załadowanie programu. Przełącz się na perspektywę CCS Debug.

Na początku pliku *Piccolo_F2802x_HD44780.h* zdefiniowana jest obsługa wyprowadzeń AIO4/AIO10 przy zastosowaniu modelu bezpośredniego dostępu do rejestrów. Stan wyprowadzeń jest indywidualnie ustawiany na poziom niski lub wysoki.

```
#define LCD_RS_COMMAND     GpioDataRegs.AIOCLEAR.bit.AIO10 = 1;//Clear to low output latch
```

Listing 4. Kod do wpisania przed pętlą *while()* w funkcji *main()*

```
// LCD modifications
LCD_Initialize();
LCD_Clear();
LCD_GoTo(0,0);  // Welcome screen
LCD_WriteText("*Piccolo F28027*");
LCD_GoTo(0,1);
LCD_WriteText("    Example");
```



Fotografia 2. Widok wyświetlanych wzorców bitowych

```
#define LCD_RS_DATA  GpioDataRegs.AIOSET.
bit.AIO10 = 1;  //Set high output latch
```

Obsługa wyprowadzeń GPIO4-GPIO7 jest realizowana przy zastosowaniu modelu drajwerów programowych. Stan wyprowadzeń jest indywidualnie ustawiany na poziom niski lub wysoki.

```
#define  DB4_HIGH  GPIO_setHigh(myGpio,
GPIO_Number_4);
#define DB4_LOW GPIO_setLow(myGpio, GPIO_
Number_4);
```

Reszta program obsługi sterownika HD44780 jest zrealizowana w standardowy sposób, z indywidualną obróbką bitów.

43. Kliknij na przycisk *Resume* na pasku narzędziowym okna *Debug*.

Zauważ napis **Piccolo F28027** wyświetlony w pierwszej linii wyświetlacza LCD (zdjęcie 1).

44. Przyciśnij i przytrzymaj na przycisk S3 (prawy).

Program przechodzi do wykonania dalszej części kodu. Obserwuj trzy wzory świecenia czterech diod LED płytki zestawu ewaluacyjnego *C2000 Piccolo LaunchPad*. W drugiej linii wyświetlacza LCD są pokazywane w formacie Hex i bitowym wzorce bitowe wystawiane przez program na wyprowadzenia GPIO0-3 (zdjęcie 1).

Podsumowanie ćwiczenia z projektem F2802x_LCD

Zaprezentowane w artykule postępowanie pozwala na poznanie sposobu obsługi wyświetlacza LCD ze sterownikiem typu HD44780 przy zastosowaniu układu procesorowego serii Piccolo F2802x oraz użyciu biblioteki *driverlib* pakietu programowego controlSUITEv3 i środowiska *Code Composer Studio* v5. Uruchamianie przykładowych programów pakietu programowego controlSUITEv3 umożliwia poznanie sposobów programowania układów procesorowych Piccolo F2802x.

Przedstawione postępowanie pokazuje typowy sposób działania projektu dla większości instalacji środowiska programowego. Jednak mogą występować różnice zachowania się środowiska dla instalacji na różnych komputerach. Pliki projektu po wykonaniu ćwiczenia (zmienione/dodane w trakcie pracy) są zamieszczone na CD.

Henryk A. Kowalski
kowalski@ii.pw.edu.pl
fotografie: Nathalie Manilow

Bibliografia

[1] Code Composer Studio, strona produktu http://www.ti.com/ccs
[2] controlSUITE Getting Started Guide (Rev. B), SPRUGU2B , 09 June 2011
[3] TMS320F28027, TMS320F28026, TMS320F28023, TMS320F28022, TMS320-F28021, TMS320F280200, Piccolo Microcontrollers, Data Sheet, SPRS523I, 31 Jul 2012
[4] TMS320F28027, TMS320F28026, TMS320F28023, TMS320F28022, TMS320-F28021, TMS320F280200, Piccolo MCU, Silicon Errata, SPRZ292J, 31 Jan 2012
[5] TMS320x2802x Piccolo System Control and Interrupts, SPRUFN3C, 29 Oct 2009
[6] F2802x Firmware Development Package USER'S GUIDE v. 210 [f2802x-FRM-EX-UG.pdf], pakiet controlSUITE
[7] F2802x Peripheral Driver Library USER'S GUIDE v. 210 [f2802x-DRL-UG.pdf], pakiet controlSUITE
[8] LAUNCHXL-F28027 C2000 Piccolo LaunchPad Experimenter Kit, User's Guide, SPRUHH2, 25 Jul 2012
[9] Alfanumeryczny wyświetlacz LCD1602, Kamami, http://www.kamami.pl/index.php?ukey=product&productID=22147
[10] CAB_A, Przewody połączeniowe, www.kamami.pl
[11] KAmodEXP1, Adresowalny ekspander GPIO z interfejsem SPI, Kamami, http://www.kamami.pl/index.php?ukey=product&productID=135566
[12] Rafał Baranowski, Wyświetlacze graficzne i alfanumeryczne w systemach mikroprocesorowych, Wydawnictwo BTC, 2008
[13] Henryk A. Kowalski, Procesory DSP dla praktyków, Wydawnictwo BTC, 2011 http://ii.pw.edu.pl/kowalski/dsp/book/
[14] Henryk A. Kowalski, Procesory DSP w przykładach, Wydawnictwo BTC, 2012 http://ii.pw.edu.pl/kowalski/dsp/book/
[15] Henryk A. Kowalski,"Zestaw ewaluacyjny C2000 Piccolo LaunchPad", Elektronika Praktyczna 01/2013
[16] Henryk A. Kowalski„C2000 Piccolo LanuchPad (1) – Pierwszy program w środowisku programowym CCSv5", Elektronika Praktyczna 02/2013
[17] Henryk A. Kowalski, „C2000 Piccolo LanuchPad (2) – Łatwe programowanie z pakietem controlSUITE", Elektronika Praktyczna 03/2013