Laboratorium Robotyki

CW-HC08 *
Programowanie mikrokontrolera MC9S08QD4 [2]

Jan Kędzierski
Marek Wnuk

Wrocław 2012

*Dokument stanowi instrukcję do ćwiczenia w ramach kursu *Sterowniki robotów*.

Przebieg ćwiczenia

Celem ćwiczenia jest ogólne zapoznanie się z 8-bitowym mikrokontrolerem z rodziny HCS08 oraz ze środowiskiem programistycznym CodeWarrior Development Studio firmy Freescale Semiconductors [1]. W ramach ćwiczenia należy wygenerować przy pomocy narzędzia Processor Expert [3] kod inicjalizujący układ, następnie uzupełnić go tak, aby układ realizował odpowiednie funkcje proponowane w niniejszej instrukcji. Ćwiczenia opisane w niniejszej instrukcji można przeprowadzić na zestawie laboratoryjnym z mikrokontrolerem MC9S08QD4 [4].

Zad1: Cykliczne przerwanie RTI, moduł obsługi klawiatury KBI

W zadaniu 1 należy utworzyć nowy projekt przy użyciu kreatora. Po utworzeniu katalogu Zad1 w menu **Plik**, należy wybrać **New Project**. Pojawi się okno kreatora, w którym należy postępować wg instrukcji:

- **Device and Connection:** wybieramy układ *MC9S08QD4* i *P&E Multilink/Cyclone Pro*,
- **Project Parameters:** wybieramy *C*, ustawiamy ścieżkę i nazwę projektu,
- **Add Additional Files:** nie dodajemy żadnych plików,
- **Processor Expert:** wybieramy *Device Initialization*,
- **C/C++ Options:** pozostawiamy bez zmian,
- **PC Lint:** wybieramy *No*.

W ćwiczeniu tym, należy skonfigurować dwa moduły. Pierwszy z nich, to cykliczne przerwanie RTI. Częstotliwość przerwania poda prowadzący. Drugi, to moduł obsługujący czteroklawiszową klawiaturę KBI. Aby móc zaprezentować działanie modułów na diodach LED, należy dodatkowo skonfigurować moduł PTA (port A).

Konfiguracja modułu PTA:

- **Settings**
  - **Port control:** wybieramy *Individual pins*,
- **Pins**
  - **Pin*x*:** konfigurujemy te linie, do których podłączono diody LED i wybieramy *Enabled*
    - **Direction:** wybieramy *Output*,
    - **Output value:** wybieramy *1*,
    - **Drive strenght:** wybieramy *High*.

Podobnie konfigurujemy PTA dla drugiej diody. Przed przystąpieniem do dalszej, części zadania można przetestować konfigurację PTA. W tym celu, w oknie **Device initlization** klikamy *Generate code* i zatwierdzamy klikając *Generate*. **Processor Expert** wygeneruje kod odpowiedzialny za poprawną inicjalizację mikrokontrolera. Po zapoznaniu się z wygenerowanym kodem, uruchamiamy *Debugger*. Jeżeli obydwie diody się zapaliły, można przejść do konfiguracji modułów RTI i KBI.

Konfiguracja modułu RTI:

- **Settings**
  - **Clock settings:**
    - **Clock select:** pozostawiamy *Internal oscillator*,
    - **Prescaler:** częstotliwość przerwania wybieramy zgodnie z zaleceniem prowadzącego,
- **Interrupts:**
  - **Real-Time Interrupt:** wybieramy *Enabled*.

Klikamy *Generate code* i zatwierdzamy klikając *Generate*. Konfiguracja RTI jest już zakończona. Naturalnie, aby móc oglądać efekt działania cyklicznego przerwania, należy w pliku źródłowym odszukać funkcję wywoływaną przez przerwanie i zapisać instrukcje, które np. będą zaświecały i gasiły dowolną diodę LED. W funkcji tej potwierdzamy obsłużone przerwanie przy pomocy instrukcji:

```
SRTISC|=(0x40); //ustaw 1 (RTIACK) w rejestrze SRTISC
```

UWAGA: Device initialization nie generuje kodu obsługującego przerwanie. W każdej funkcji wywoływanej przez dane przerwanie, należy „ręcznie" zgasić flagę. W przeciwnym wypadku program nigdy nie opuści przerwania.

Konfiguracja modułu KBI:

- **Settings**
  - **Triggering sensitivity:** wybieramy *edge*,
- **Pins**
  - **Pin*x*:** konfigurujemy tę linię, do której podłączono klawisz i wybieramy *Enabled*,
    - **Pull resistor:** wybieramy *pull up*,
- **Interrupts:**
  - **Keyboard request:**
    - **Keyboard request interrupt:** wybieramy *Enabled*.

Ponownie klikamy *Generate code* i zatwierdzamy klikając *Generate*. Konfiguracja KBI jest już zakończona. W pliku źródłowym, należy odszukać funkcję wywoływaną przez przerwanie i zapisać instrukcje, które np. będą zaświecały i gasiły dowolną diodę LED. W funkcji tej, potwierdzamy obsłużone przerwanie przy pomocy instrukcji:

```
KBISC|=(0x04); //ustaw 1 (KBACK) w rejestrze KBISC
```

Do oceny należy pozostawić projekt z komentarzami wyjaśniającymi powyższą konfigurację wszystkich modułów. Komentarze powinny znajdować się na początku każdej funkcji, której one dotyczą.

Zad2: Funkcje czasowo-licznikowe TPMx, przetwornik analogowo cyfrowy ATD

W zadaniu 2 przy pomocy kreatora można utworzyć na nowo projekt lub skopiować dotychczasowy do utworzonego katalogu Zad2. W ćwiczeniu tym należy uruchomić kolejne dwa moduły. Pierwszy z nich konfigurujemy tak, aby pracował jako modulator PWM o częstotliwości i wypełnieniu zadanym przez prowadzącego. Efekt pracy modulatora należy zwizualizować na diodzie LED. Drugi moduł konfigurujemy tak, aby pracował jako 8 - bitowy przetwornik A/C na kanale, do którego podłączono potencjometr. Wartość zmierzoną należy odpowiednio przeskalować i zadać w postaci wypełnienia na wcześniej skonfigurowany kanał PWM. W rezultacie uzyskamy regulację jasności świecenia diody LED przy pomocy potencjometru.

Konfiguracja modułu TPM1:

- **Settings**
  - **Clock settings:**
    - **Clock source select:** pozostawiamy *Fixed system clock*,
    - **Prescaler:** wybieramy na podstawie zadanej częstotliwości,
    - **Modulo counter:** wpisujemy na podstawie zadanej częstotliwości,
  - **Aligned:** pozostawiamy na *left*,
  - **Channels:** ustawiamy *1*,
    - **Channel:**
      - **Capture/compare device:** wybieramy *TPM11*,
      - **Settings:**
      - »**Mode:** wybieramy *PWM*,
      - »» **PWM output action:** wybieramy *Clear output on compare*,
      - »» **Channel compare capture:** wpisujemy na podstawie zadanego wypełnienia,
      - **Pin:**
      - »**Channel pin:** wybieramy *PTA1_KBI1P1_TPM1CH1_ADC1P1*.

Przed przystąpieniem do dalszej części zadania, można przetestować konfigurację TPM1.

Konfiguracja modułu ATD:

- **Settings**
  - **Clock settings:**
    - **Input clock select:** pozostawiamy *BusClk*,
    - **Prescaler:** pozostawiamy *1*,
  - **Conversion mode:** wybieramy *Continous conversion*,
  - **Result data format:** pozostawiamy *8-bit right*,
- **Pins**
  - **ADC Input Pins:** klikamy *1*,
    - **Pin:** wybieramy pin, do którego podłączono potencjometr,
- **Initialization**
  - **Initial channel select:** wybieramy kanał, do którego podłączono potencjometr.

Zmierzoną przy pomocy przetwornika A/C wartość podajemy jako wypełnienie sygnału PWM. W tym celu należy np. w programie głównym użyć następujących instrukcji:

```
TPMC1VH=0; // TPMC1V jest 16 bitowy (dwa bajty L i H)
TPMC1VL=ADCRL;
```

Do oceny należy pozostawić projekt z komentarzami wyjaśniającymi powyższą konfigurację wszystkich modułów. Komentarze powinny znajdować się na początku każdej funkcji, której one dotyczą.

Literatura

[1] *www.freescale.com*, Freescale Semiconductors

[2] *MC9S08QD4 Datasheet*, Rev. 4, 9/2008, Freescale Semiconductors

[3] *CodeWarrior Development Studio for Microcontrollers, Quick Start*, Freescale Semiconductors

[4] Kędzierski J. Wnuk M. *Opis płytki demo z mikrokontrolerem MC9S08QD4*, Wrocław 2012.