

Laboratorium: Systemy Wbudowane w Układach Sterowania

**Ćwiczenie nr 6**

TEMAT: Przerwania w systemie ARM7.

**1. Przykład 1**

**Konfiguracja systemu do obsługi przerwań pochodzących od kontrolera PIOA w reakcji na zmianę stanu linii PA7 i PA14 (odchylenie joysticka w lewo i w prawo)**

W celu wykorzystania przerwań pochodzących od kontrolera PIO należy:

1. Napisać odpowiednią **procedurę obsługi przerwań**.
2. Skonfigurować kontroler PIO w taki sposób, aby generował żądanie obsługi przerwań w reakcji na zmianę stanu na wskazanych liniach.
3. Skonfigurować kontroler AIC w celu właściwej reakcji na zgłaszane żądanie obsługi przerwania.

**Ad1.**

```
#define LEFT_STICK (((AT91C_BASE_PIOA->PIO_PDSR) & AT91C_PIO_PA7)==0)

static volatile unsigned int status_IRQ, zmienna_1, zmienna_2;

__irq void pioIsr(void)
{
    //kopiujemy zawartosc rejestru PIO_ISR do zmiennej status
    status_IRQ = AT91C_BASE_PIOA->PIO_ISR;
    //teraz flagi w PIO_ISR sa juz skasowane

    //.....
    //kod uzytkownika

    //przechodzimy do analizy zawartosci zmiennej status
    //przykladowo:

    if (status_IRQ & (1<<7)) //jesli PA7 wywolalo przerwanie...
        //(dowolna zmiana stanu na linii PA7 = wcisniecie lub puszczenie klawisza)
        {
            //kod dla zmiany PA7
            if (LEFT_STICK) //dodatkowo sprawdzamy, czy aktualny stan to wcisniety klawisz? (czyli
                zdarzenie, które wystapilo to wcisniecie)
            {
                zmienna_1--;
            }
        }
    if (status_IRQ & (1<<14)) //jesli PA14 wywolalo przerwanie... (dowolna zmiana stanu na linii PA8 =
        wcisniecie lub puszczenie klawisza)
    {
        //kod dla PA8...
        zmienna_1++;
    }
    //.....
}
```

```

        //tak zakonczamy przerwanie
        AT91F_AIC_AcknowledgeIt(AT91C_BASE_AIC);
    }

```

## Ad2.

```

//***** konfiguracja przerwan zgłaszanych od jednostki PIOA *****
//*****
//przerwanie od PIOA - bedzie zgłaszane przy zmianie stanu wybranych wejsc (wychylenie joysticka w lewo
lub w prawo) - czyli zmianie zawartosci rejestru kontrolera AT91C_BASE_PIOA->PIO_PDSR

```

```

AT91C_BASE_PIOA->PIO_IDR = 0xffffffff; //na wszelki wypadek wyłączenie zezwolen przerwan od wszystkich
linii PIOA (gdyby przypadkowo byly wzczesniej zezwolone)

```

```

AT91C_BASE_PIOA->PIO_IER = (1<<7) | (1<<14); //przerwania beda generowane w reakcji na zmianie stanu linii
PA7 i PA14

```

```

//PA7 = left, PA8 = down, PA9 = UP, PA14 = PRAWO, AP15 = CLICK

```

## Ad3.

```

//*****
//*****

```

```

const unsigned char PIO_IRQ_PRIORITY = 5;

```

```

//przerwanie zostanie "podlaczone" do kontrolera AIC za pomoca pierwszego sposobu, tj. z wykorzystaniem
procedur biblioteki "lib_AT91SAM7X256"
//funkcja biblioteki "lib_AT91SAM7X256" podlaczajaca procedure obslugi przerwania do systemu przerwan AIC

```

```

AT91F_AIC_ConfigureIt ( AT91C_BASE_AIC , AT91C_ID_PIOA ,    PIO_IRQ_PRIORITY ,
                        AT91C_AIC_SRCTYPE_INT_HIGH_LEVEL , (void*) pioIsr);

```

```

status_IRQ = AT91C_BASE_PIOA->PIO_ISR; //odczyt statusu PIOA - spowoduje to skasowanie ew. zgloszonych
wczesniej przerwan

```

```

AT91F_AIC_AcknowledgeIt(AT91C_BASE_AIC); //potwierdzenie ew. wzczesniej zgloszonych przerwan w AIC

```

```

AT91F_AIC_EnableIt (AT91C_BASE_AIC, AT91C_ID_PIOA); //zezwozenie na generacje przerwan od calego PIOA

```

```

//*****
//*****

```

## 1. Przykład 2

**Konfiguracja systemu do obsługi przerwań pochodzących od timera TC0. Timer zostanie skonfigurowany w ten sposób aby zgłaszał żądanie obsługi przerwania z częstotliwością 100 Hz.**

W celu realizacji postawionego zadania należy:

1. Napisać odpowiednią **procedurę obsługi przerwań**.
2. Skonfigurować timer TC) w taki sposób, aby generował żądanie obsługi przerwań **w reakcji osiągnięcia wartości wskazywanej rejestrem TC\_RC**.  
Timer TC0 liczy impulsy wejściowe (pochodzące z zegara o wybranej częstotliwości. Zliczanie następuje od wartości zero w górę. Wpływ na szybkość uzyskania wartości TC\_RC przez timer ma: a) częstotliwość zliczanych impulsów oraz b) wartość zapisana w rejestrze TC\_RC.
3. Skonfigurować kontroler AIC w celu właściwej reakcji na zgłaszane żądanie obsługi przerwania.

### Ad1.

```
// Timer0 ISR
__irq void tim0_isr (void) {

    volatile int dummy = AT91C_BASE_TC0->TC_SR;          // Interrupt Ack - odczytanie rejestru statusu kasuje
    flage zgloszenia komparacji timera
    AT91C_BASE_AIC->AIC_ICCR = (1 << AT91C_ID_TC0); // Interrupt Clear Command Register - skasowanie flagi
    zgloszenia przerwania w AIC

    //.....
    //tu mozemy wstawic swoje instrukcje, np.
        zmienna_2++;
    //.....

    //tak zakonczamy przerwanie
        *AT91C_AIC_EOICR = 0;                                // End of Interrupt
}
```

### Ad2.

```
//*****
//***** Konfiguracja timera TC0 oraz przerwan zgłaszanych przy doliczeniu do wartosci TC_RC
//*****

    AT91F_PMC_EnablePeriphClock(AT91C_BASE_PMC, 1 << AT91C_ID_TC0); // Enable Clock for TIM0
    AT91C_BASE_TC0->TC_CCR      = AT91C_TC_CLKEN | AT91C_TC_SWTRG; //clk enable oraz Software trigger
    (softwareowy restart licznika)
    AT91C_BASE_TC0->TC_CMR      = 2 | AT91C_TC_CPCTRG; //2 - sterowany przez Timer_CLOCK3 = Master Clock
    MCK/32 ;
    //czyli czestotliwosc zegara napedzajacego timer = 1.4975MHz (wynika to z wartosci 18.432MHz
    (czestotliwosc kwarcu) / 32

    //AT91C_TC_CPCTRG = compare trigger enable
    AT91C_BASE_TC0->TC_RC      = 14975; // period is 10ms - 14975 cykli po 0.6677 us kazdy = 10000us

    AT91C_BASE_TC0->TC_IER      = AT91C_TC_CPCS; //zgłaszanie przerwan Timera w reakcji na wykrycie stanu
    COMPARE
    //czyli w reakcji na zrownanie sie wartosci licznika z rejestrem TC_RC
```

### Ad3.

```
//*****
//przerwanie zostanie "podlaczone" do kontrolera AIC za pomoca drugiego sposobu, tj. BEZ wykorzystania
    procedur biblioteki "lib_AT91SAM7X256"

// Konfiguracja przerwan zgłaszanych przez timer: 1. tryb pracy, 2. ustawienie wektora przerwania oraz 3.
    zezwolenie na przerwanie
    // TIM0 Interrupt: Mode and Vector with Lowest Priority and Enable
    //wszystko to ustawia sie w AIC (Advanced Interrupt Controller)

    AT91C_BASE_AIC->AIC_SMR[AT91C_ID_TC0] = AT91C_AIC_SRCTYPE_INT_POSITIVE_EDGE | AT91C_AIC_PRIOR_HIGHEST;
    //tryb przerwania - wewnetrzne, wyzwalane zboczem (od pojawienia sie stanu przepelnienia), poziom
    priorytetu

    AT91C_BASE_AIC->AIC_SVR[AT91C_ID_TC0] = (unsigned long) tim0_isr;
    //wektor przerwania TC0 wskazuje teraz na NASZA procedure obslugi 'tim0_isr'

    AT91C_BASE_AIC->AIC_IECR = (1 << AT91C_ID_TC0);
    //zezwozenie na przerwanie od Timera TC0 (przerwanie zgłaszane jest gdy timer doliczy do wartosci
    compare)

//*****
```

### **3. Program ćwiczenia**

1. Utworzyć program według przykładu 1. Wgrać do sterownika i uruchomić. Sprawdzić poprawność działania.
2. Poprawić kod programu w ten sposób aby zwiększanie wartość „zmienne\_1” przy zmianie joysticka w prawo następowało tylko przy odchyleniu (a nie przy puszczeniu) dźwigni.
3. Utworzyć program według przykładu 2. Wgrać do sterownika i uruchomić. Sprawdzić poprawność działania.
4. Przerobić program z zadania 3 w ten sposób, aby przerwania zgłaszały się 1000 razy na sekundę.
5. Odpowiedzieć na pytanie (analizując dokumentację procesora oraz sprawdzając doświadczalnie) jaką najniższą częstotliwość przerwań może generować timer TC0? W sprawozdaniu należy przedstawić obliczenia (wykonane na podstawie analizy dokumentacji procesora) oraz wynik uzyskany na rzeczywistym stanowisku.
6. Na podstawie programu z zadania 3 oraz programami z poprzednich zajęć napisać program odliczający czas w postaci hh:mm:ss.

**W rozwiązaniu należy podać:**

1. Treść zadań
2. Kod zadań wraz z obszernymi komentarzami
3. Wnioski i odpowiedzi na postawione pytania