Katedra Inżynierii Systemów, Sygnałów i Elektroniki
Wydział Elektryczny
Zachodniopomorski Uniwersytet Technologiczny w Szczecinie

LABORATORIUM

TECHNIKA MIKROPROCESOROWA

Porty równoległe wejścia/wyścia mikrokontrolera ATmega

Opracował:
Tomasz Miłosławski

1. Cel ćwiczenia

Celem ćwiczenia jest zapoznanie się ze sposobami konfiguracji i programowania równoległych portów wejścia-wyjścia (I/O) mikrokontrolera rodziny AVR ATmega oraz ich wykorzystania w sterowaniu urządzeń wykonawczych i identyfikacji zewnętrznych sygnałów cyfrowych.

2. Charakterystyka portów równoległych AVR ATmega

Porty równoległe są podstawowymi urządzeniami wejścia/wyjścia systemów mikroprocesorowych. Przeznaczone są przykładowo do transmisji informacji w postaci równoległej, binarnego sterowania w urządzeniach przemysłowych, komunikacji wizualnej z operatorem, przyjmowania informacji z przycisków, klawiatur itp. Najczęściej porty zorganizowane są w grupy linii o długości słowa zgodnej ze słowem mikrokontrolera, np. 8 linii dla urządzeń 8-bitowych. W mikrokontrolerze AVR ATmega porty opisane są jako np.: **PORTA**, **PORTB**, **PORTC**, **PORTD**. Każda linia wybranego portu może być konfigurowana indywidualnie jako:

- wyjście (OUTPUT),

- wejście (INPUT),

- wejście z pociągnięciem do VCC (INPUT with pull-up).

Pojedyncze linie portu oznaczane są za pomocą symboli PA0-PA7, PB0-PB7, PC0-PC7, PD0- PD7.

Stan portów oraz ich konfigurację określa się za pomocą trzech rejestrów:

- DDRx - Data Direction Register x
- PORTx - Data Register x
- PINx - Input Pins Adress x

Tabela 1. Konfiguracja linii portów

DRRxn	PORTxn	KIERUNEK	Pull-up	Stan
0	0	wejście	NIE	wejście Hi-Z
0	1	wejście	TAK	Wejście podciągnięte do VCC
1	0	wyjście	NIE	stan niski
1	1	wyjście	NIE	stan wysoki

x - oznaczenie port A, B, C, D
n - numer linii portu 0, 1, 2, 3, 4, 5, 6, 7
Hi-Z - stan wysokiej impedancji

Program *Listing 1* przedstawia przykładową konfigurację portu równoległego PORTA, gdzie linie PA3..0 są skonfigurowane jako wyjścia, a linie PA7..4 jako wejścia.

Listing 1

```
#include <avr.io.h>

int main (void)
{
 DDRA = 0x0F;        // DDRA  = 0000FFFF
 PORTA = 0xC3;       // PORTA = 11000011
         .
         .
         .
         .
  while (1);
}
```

Po określeniu kierunku portu stan wyjść ustalony został następująco:
PA1..0 = 1,
PA3..2 = 0,
natomiast stan wejść:
PA5..4 - wejście Hi-Z,
PA7..6 - wejście z podciągnięciem do VCC.

Sposób w jaki można dokonać odczytu i ustawienia wybranych linii portu przestawia program *Listing 2*.

Listing 2

```
#include <avr.io.h>

int main (void)
{
  DDRA = 0x0F;                      // DDRA  = 0000FFFF
  PORTA = 0xC3;                     // PORTA = 11000011
  while (1)
       {
        if ( PINA & 0x80 )          // test stanu linii PA7
           {
            PORTA = PORTA & 0xFE;  // PA0 = 0
           }
          else
           {
            PORTA = PORTA | 0x01; // PA0 = 1
           }
       }
  while (1);
}
```

3. Ćwiczenia programowe

ZADANIE 1
Napisać program generujący przebieg prostokątny o częstotliwości 10Hz na wyjściu linii portu PB0. Stan linii portu ma być sygnalizowany za pomocą diody LED D0 na zestawie dydaktycznym ZL3AVR.

ZADANIE 2
Napisać program sterowania diodami LED D0-D7 przez PORTB zestawu dydaktycznego ZL3AVR dla uzyskania widocznego efektu świecącego punktu biegnącego w lewą stronę. Po osiągnięciu skrajnej lewej diody cykl powinien się powtarzać.

ZADANIE 3
Napisać program sterowania diodami LED D0-D7 przez PORTB zestawu dydaktycznego ZL3AVR dający efekt świecącej linijki o zwiększającej się długości w prawo od 1 diody do 8 diody. Następnie diody powinny być wygaszane od prawej do lewej strony.