Alicja Kapiszka
Paweł Adamski
Informatyka N1, gr. 30C

Sprawozdanie nr 5

Transmisja szeregowa 8051

Wstęp

Podobnie jak na poprzednich zajęciach, wykorzystano mikrokontroler 8051 i zestaw uruchomieniowy ZL2MCS51. Przyciski podłączono do portu P3, a wyświetlacze 7-segmentowe do portu P0. Wykorzystano oprogramowanie MIDE-51 i Flip.

Kolejnym używanym na laboratoriach elementem mikrokontrolera 8051 był układ transmisji szeregowej, zapewniający komunikację pomiędzy mikrokontrolerem a światem zewnętrznym. Układ ten wykorzystuje dwie linie portu P3 (nr 0 i 1), noszące nazwy TxD (transmitted data) i RxD (received data). Przy wykonaniu tego ćwiczenia spięto ze sobą wyprowadzenia tych linii (wyprowadzenia nr 1 i 2) na złączu JP9 (dane przesyłane były wewnętrznie). Konfiguracja układu transmisji szeregowej – m.in. wybór trybu pracy – jest możliwa poprzez rejestr SCON. W rejestrze SCON ustawiono bit REN na 1, przez co aktywowano możliwość odbioru danych (która domyślnie jest wyłączna). Możliwy jest też wybór czterech trybów pracy, które wybiera się przez ustawienie bitów SM0 i SM1 tego rejestru. Na niniejszych laboratoriach wykorzystany został tryb 2 – transmisja synchroniczna dziewięciobitowa. W trybie tym do transmisji danych używany jest rejestr SBUF, do którego zapisuje się dane, a następnie odczytuje. Transmisja następuje z określoną prędkością 1/64 lub 1/32 zegara, którym odpowiadają odpowiednio wartości 0 i 1 bitu SMOD w rejestrze PCON. Wartości te ustawia się poprzez nakładanie masek, w naszym przypadku ustawiono wartość 0 poprzez operację: PCON = PCON & 0x7F. Do obsługi układu transmisji szeregowej wykorzystano przerwanie (nr 4), w którym zaimplementowano odbiór danych. We wspomnianym rejestrze SCON znajdują się również bity TI i RI będące znacznikami przerwania odpowiednio nadajnika i odbiornika. Są one ustawiane automatycznie po wysłaniu / odebraniu ostatniego bitu danych. W poniższym ćwiczeniu wykorzystany został bit RI, który zerowano po odczytaniu wszystkich danych z bufora (na koniec procedury przerwania).

Na laboratoriach należało wykorzystać programy z poprzednich zajęć. Należało zapewnić możliwość ustawiania za pomocą dwóch przycisków (nr 6 i 7) liczb na wyświetlaczu (dwóch cyfr na dwóch wyświetlaczach – nr 3 i 4), a następnie przy użyciu przycisku nr 5 należało umożliwić przesłanie tych cyfr za pomocą układu transmisji szeregowej na wyświetlacze nr 1 i 2. W celu minimalizacji przesyłanych danych, obie nadawane cyfry należało umieścić w jednym bajcie danych, a po odebraniu – ponownie rozdzielić.

Kod programu:

```
#include "8051.h"

#define TH0_RELOAD 0xF7
#define TL0_RELOAD 0x00
#define TIK 1
int wysw = 1;
int serialUpdated = 0;
int serialData = 0;

void timer0_init(void) {
    TH0 = TH0_RELOAD;
    TL0 = TL0_RELOAD;
    TMOD = TMOD | 0x01;
    TR0 = 1;
    ET0 = 1;
}

// Przerwanie dot. timera
void timer_isr (void) __interrupt (1) __using (0) {
    static int count=0;
    TH0 = TH0_RELOAD;
    TL0 = TL0_RELOAD;
    count++;
    if (count==TIK) {
        count=0;
        P2_7=!P2_7;
        wysw *= 2;
        if(wysw == 16) wysw = 1;
    }
}

//Przerwanie dot. odbioru danych
void nazwa_procedury (void) __interrupt (4) __using (0) {
    if (RI==1) {
        serialData = SBUF;
        serialUpdated = 1;
        RI = 0;
    }
}

// Procedura konfiguracji układu transmisji szeregowej
void serial_init(void) {
    SM0 = 1; // ustawienie trybu pracy nr 2
    SM1 = 0;
    TI = 1; // pozwolenie na rozpoczęcie nadawania w pętli głównej
    RI = 0;
    ES = 1; // dopuszczenie przerwań od układu transmisji szeregowej
    PCON = PCON & 0x7F; // ustawienie zegara na 1/64
}

char nums[10] = {0x3F, 0x06, 0x5B, 0x4F, 0x66, 0x6D, 0x7D, 0x07, 0x7F,
0x6F};

void main(void) {
    int num0 = 0;
    int num1 = 0;
    int num2 = 0;
    int num3 = 0;
```

```
int stateS0 = 1;
int stateS1 = 2;
int stateS2 = 3;
int state = stateS0;
int btn_current = 0xFF;
int btn_before = 0xFF;
int btn_zero = 0xFF;
int time = 0;
int maxTime = 0x0F;
int update = 0xFF;
int skip = 0;

EA = 0; // zablokowanie przerwań
serial_init(); // przygotowanie układu transmisji szeregowej
EA = 1; // odblokowanie przerwań
REN = 1; // zezwolenie na odbiór danych

EA = 0; // zablokowanie przerwań
timer0_init(); // przygotowanie układu Timer0
EA = 1; // odblokowanie przerwań
P2=0xFF; // wygaszenie wszystkich diod

while(1) {
    P2_6 = P3_6; // obsługa przycisku i diody
    if(!skip)
    {
        if(state == stateS0)
        {
            btn_current = P3;
            if(btn_current != btn_zero)
               skip = 1;
            for(time = 0; time < maxTime; time++);
            if(P3 == btn_current)
               state = stateS1;
        }else{
            if(state == stateS1){
                if(btn_current != P3)
                    state = stateS2;
            }else{
                state = stateS0;
                update = btn_current;
                btn_current = 0xFF;
            }
        }
    }else{
        skip++;
        if(skip >=maxTime)
           skip = 0;
    }
    if(!(update & 0x20)){
        update = 0xFF; // nadawanie
        //przygotowanie danych:
        serialData = 0;
        serialData = num3 << 4;
        serialData |= num2;
        SBUF = serialData;
   }
```

```
        if(serialUpdated){
            serialUpdated = 0;
            num0 = 0;
            num0 = serialData & 0x0F
            num1 = 0;
            num1 = (serialData >> 4) & 0x0F;
        }
        if(!(update & 0x40)){
            update = 0xFF;
            num2++;
            if(num2>9)
                num2=0;
        }
        if(!(update & 0x80)){
            update = 0xFF;
            num3++;
            if(num3>9)
                num3=0;
        }

        P1 = wysw;
        switch(P1){
            case 1:
                P0=nums[num0];
            break;

            case 2:
                P0=nums[num1];
            break;

            case 4:
                P0=nums[num2];
            break;

            case 8:
                P0=nums[num3];
            break;
        }

    }
}
```