Stereofoniczny moduł sonaru
(((STEREO)))

Karol Sydor
Jan Kędzierski

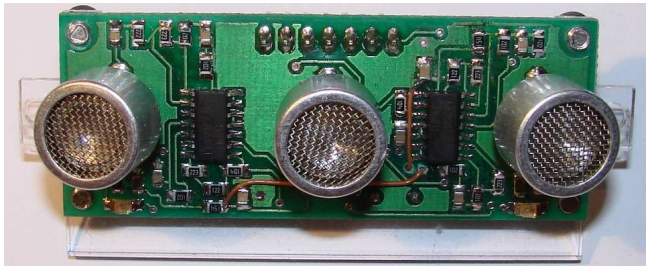
Koło Naukowe Robotyków KoNaR.
www.konar.pwr.wroc.pl

15 marca 2008

Spis treści

1 Wstęp

Stereofoniczny moduł sonaru (((STEREO))) został zaprojektowany z myślą o zastosowaniu w robotach minisumo. Jego parametry pozwalają wykorzystać go w dowolnej aplikacji, gdzie wymagany jest pomiar odległości w zakresie 5cm-1,8m. Ze względu na zastosowanie dwóch torów odbiorczych, możliwe jest rozróżnienie kierunku z jakiego powraca echo.



Rysunek 1: Moduł sonaru

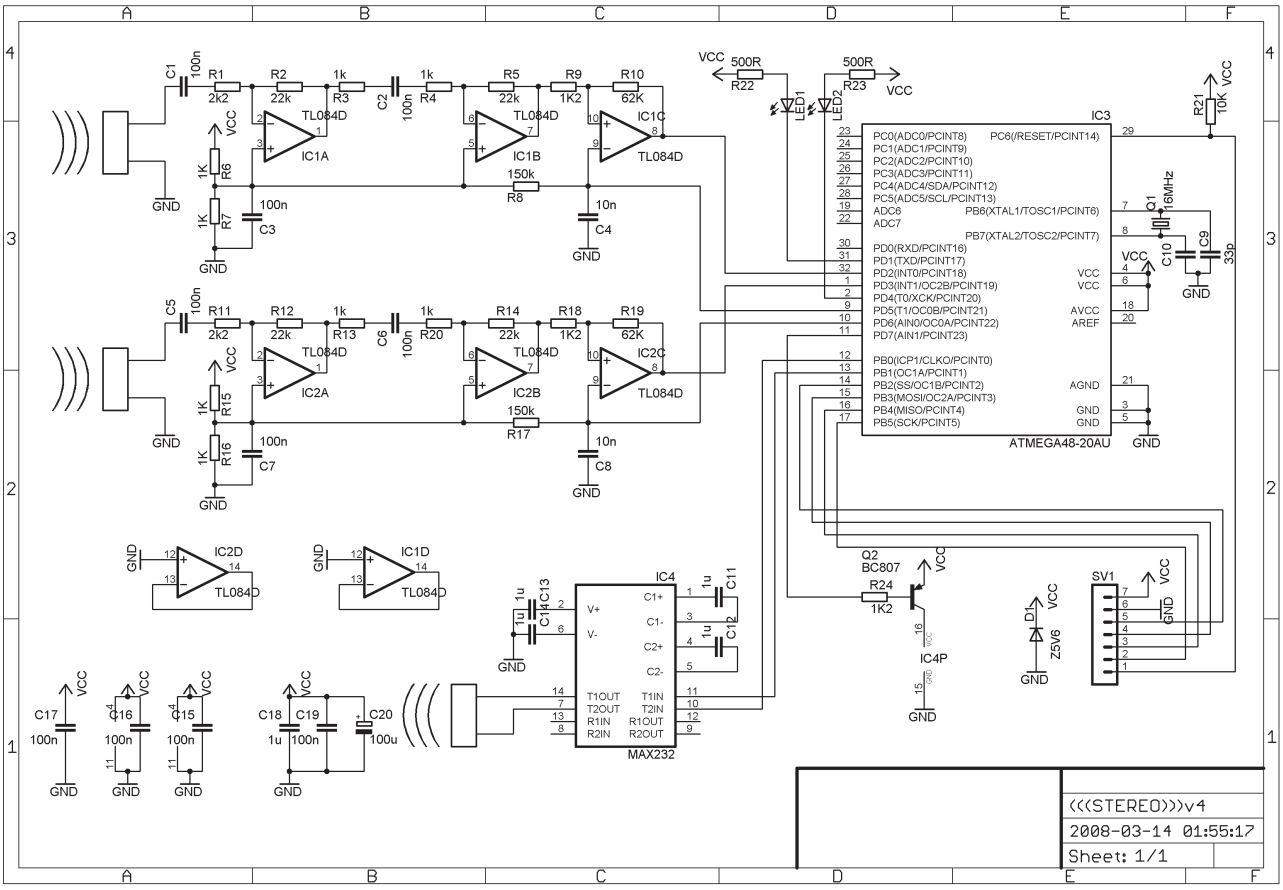
Układ stereofoniczny był rozwijany w naszym Kole od dawna. Pierwszym robotem wykorzystującym tego typu konstrukcję był "Wibrobot". Kolejna wersja sonaru została zastosowana w robocie "X". Doświadczenia z obydwu konstrukcji umożliwiły opracowanie kompletnego modułu.

2 Montaż

Montaż należy rozpocząć klasycznie od elementów najmniejszych, na złączu i przetwornikach skończywszy. Warto zastosować podkładkę pod kwarc. Mikrokontroler należy zaprogramować po wlutowaniu. Można zastosować Atmegę48, 88 lub 168, ważne aby były w wersji umożliwiającej taktowanie zegarem 16MHz. Należy zastosować diodę zenera w obudowie MELF lub odpowiednio przyciąć diodę do montażu przewlekanego, o mocy co najmniej 1,3W. Mniejsze diody (0.5W i MINIMELF) nie zapewniają wymaganej ochrony. Przetworniki należy wlutować w odległości 4-5mm od płytki. Należy je lutować ostrożnie, starając się sam proces podgrzewania skrócić do minimum - są wrażliwe na przegrzanie.

Pin	Sygnał
1	Reset
2	SCK
3	MISO
4	MOSI
5	SS
6	GND
7	VCC

Tablica 1: Opis wyprowadzeń



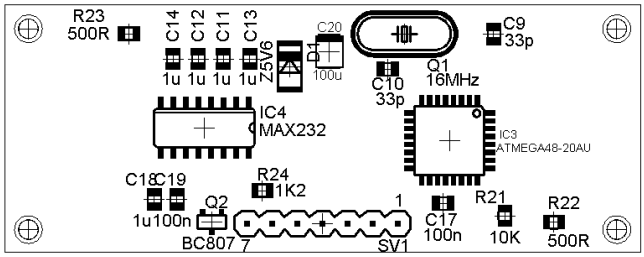
Rysunek 2: Schemat

3 Programowanie

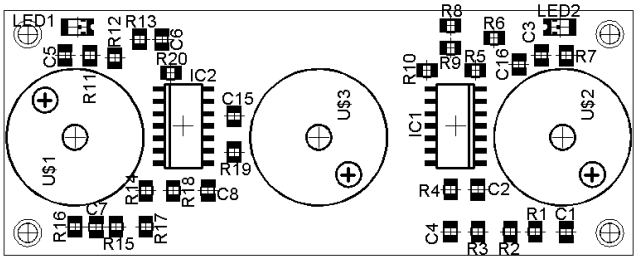
W celu zaprogramowania mikrokontrolera, należy podłączyć do złącza sonaru programator ISP, zgodnie z rozkładem wyprowadzeń pokazanym w tabeli 1. W pierwszej kolejności należy zaprogramować bity konfiguracyjne - FUSES, zgodnie z tabelą 2. Bity zaznaczone na czerwono to te których wartośco należy zmienić względem ustawień fabrycznych. Należy zachować ostrożność w czasie programowania, błędne ustawienie może trwale zablokować procesor. Następnym etapem jest wgranie zawartości pamięci flash mikrokontrolera. Pliki .hex dostępne są w archiwum na stronie KoNaR-u. Należy wgrać plik o nazwie takiej jak zastosowany układ. Po pomyślnym zaprogramowaniu, powinna zapalić się prawa dioda LED, oznaczająca że sonar jest gotowy do inicjacji.

4 Obsługa

Sonar komunikuje się z jednostką nadrzędną za pomocą interfejsu SPI. Maksymalna częstotliwość zegara SPI to 2MHz. Sonar posiada wejście RESET, jego



Rysunek 3: Rozmieszczenie elementów - tył



Rysunek 4: Rozmieszczenie elementów - przód

podłączenie nie jest konieczne. Wyprowadzenie oznaczone jako SS to pin SlaveSelect, występujący częściej pod nazwą ChipSelect. Jest aktywowany stanem niskim. Interfejs należy ustawić w trybie MODE 1: CPHA = 1, CPOL = 0. Bit MSB powinien być transmitowany jako pierwszy. Format transmisji pojedynczego bajtu obrazuje 5. Każda ramka wysyłana do sonaru powinna się składać z 4 bajtów.
Parametry sonaru

-

W przypadku kiedy sonar nic nie widzi zwraca wartość 20000. Od momentu wysłania komendy pomiaru, do jego zakończenia należy odczekać co najmniej 12,5ms.

Po podłączeniu sonaru do zasilania należy odczekać co najmniej 100ms. Po tym czasie zaświeci się jedna dioda LED. W tym trybie sonar czeka na inicjację. Polega ona na wysłaniu do sonaru czterech bajtów o wartości 0xF0. Sonar odpowiada 3 bajtami bez znaczenia, oraz czwartym o wartości oznaczającej wersję firmware (10 oznacza wersję 1.0 itp.) Po poprawnej inicjacji zaświecą się obie diody LED. Sonar na każdą ramkę odpowiada 4 bajtami będącymi wynikiem ostatniego pomiaru, w formacie Hi(Lewy) Low(Lewy) Hi(Prawy) Low(Prawy). Gdzie Lewy i Prawy to 16 bitowe wartości pomiaru. Hi(Lewy) to górne 8 bajtów, Low(Lewy) to dolne 8 bajtów tworzące 16 bitową zmienną. Odległość można wyliczyć ze wzoru 1

$$D = W * 0,09mm. \tag{1}$$

FUSE	Wartość
SELFPRGEN	1
BOTSZ(1:0)	00
BOTRST	1
RSTDISBL	1
DWEN	1
SPIEN	0
WDTON	1
EESAVE	1
BODLEVEL(2:0)	101
CKDIV8	1
SUT(1:0)	10
CKSEL(3:0)	1111

Tablica 2: Konfiguracja fusebit

Gdzie D to odległość w mm. W to zmierzona wartość. W przypadku kiedy sonar nie widzi przeszkody w swoim zasięgu zwraca wartość 20000. Po wydaniu komendy pomiaru należy odczekać maksymalnie 12,5ms na wyniki pomiaru.

4.1 Firmware w wersji 1.0

Aby wykonać pomiar należy wysłać do sonaru 3 bajty o dowolnej wartości. Ostatni bajt określa liczbę strzałek i może przyjmować wartości 5 i 8. Sonar rozpoznaje kierunek dopiero dla odległości większych od 15cm. Oprogramowanie było przystosowane do sonarów w wersji V3 bez przeróbek i nie jest udostępniane.

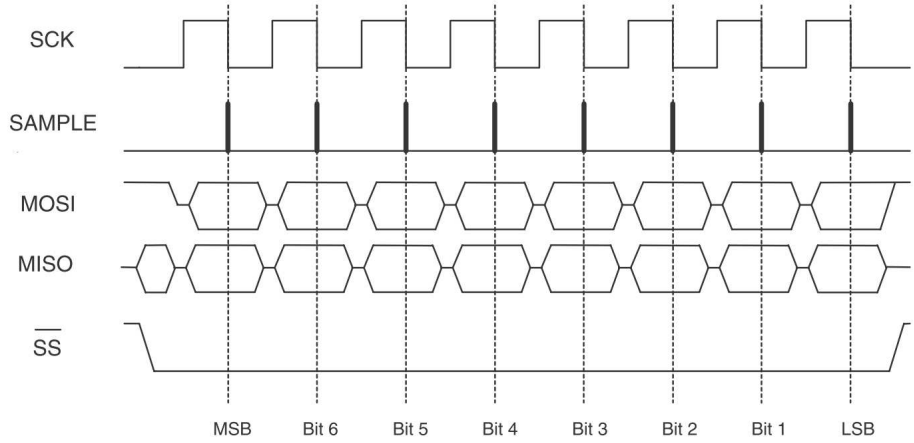
4.2 Firmware w wersji 1.1

Aby wykonać pomiar należy wysłać do sonaru 4 bajty o dowolnej wartości. Liczba strzałek została ustawiona na optymalną wartość. Wprowadzono obsługę dynamicznego progowania, co znacznie poprawiło zasięg sonaru. Sonar rozpoznaje kierunek od odległości 5cm. Jest to aktualnie udostępniana wersja.

5 Parametry sonaru

Parametry sonaru dla wersji oprogramowania 1.1 (parametry wyznaczono dla przetworników $\phi = 12mm$.)

- Interfejs SPI
- Zasilanie 5V, max. 50mA
- Zasięg 5cm-1.8m, poniżej 5cm sonar nie zwraca prawidłowej odległości, ale widzi przeszkodę.
- Czas pomiaru do 12,5ms, częstotliwość maksymalna 80pomiarów/s
- Wymiary 64x25mm

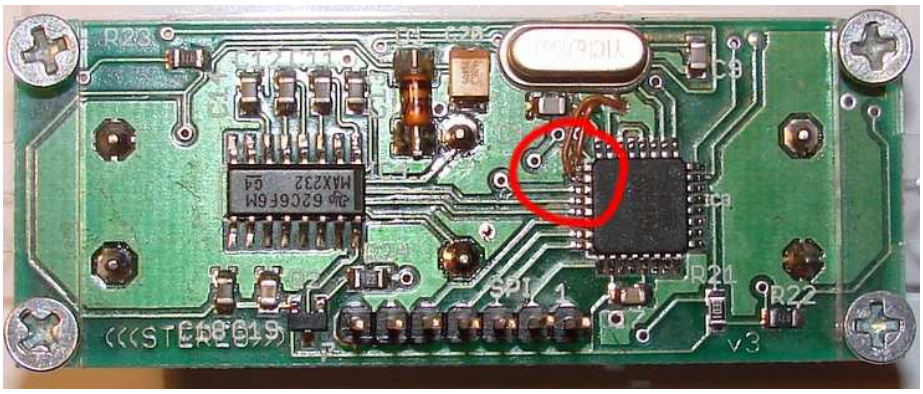


Rysunek 5: Transmisja pojedynczego bajtu

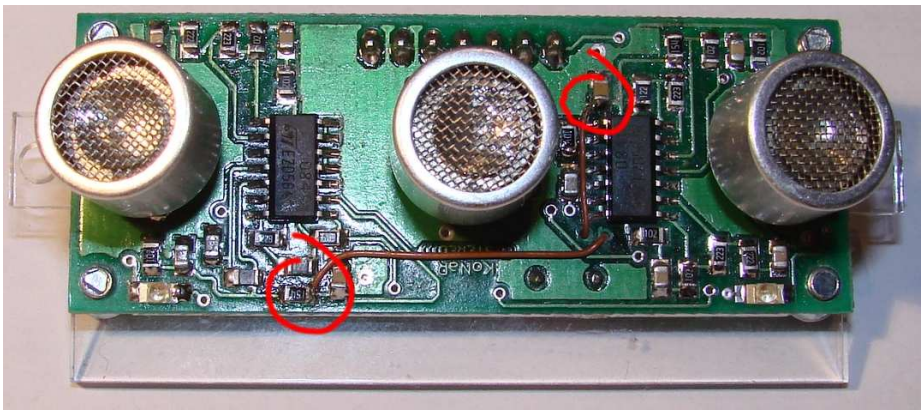
Zasięg sonaru jest ograniczany programowo. Większość producentów uzyskuje podobne wyniki dla przetworników $\phi = 16mm$, które charakteryzują się znacznie większym wzmocnieniem.

6 Przeróbka płytki v3 do wersji v4

Sonar oparty o płytkę w wersji v4 ma znacznie lepsze parametry. Aby przerobić płytkę z wersji v3 należy wykonać dwa dodatkowe połączenia. Proponujemy wykonanie połączeń za pomocą kynaru, będzie to wymagało wywiercenia dwóch dodatkowych otworów w płytce. Sposób modyfikacji pokazano na 7 i 6. Przewody należy dołączyć do 9 i 10 wyprowadzenia mikrokontrolera, kolejność nie ma znaczenia. Podczas wiercenia otworów należy uważać żeby nie przeciąć żadnej ścieżki. Można wiercić otwory w płaszczyźnie masy. Po wykonaniu opisanej modyfikacji płytka v3 jest elektrycznie zgodna z wesją v4.



Rysunek 6: Modyfikacja - widok z tyłu



Rysunek 7: Modyfikacja - widok z przodu

Lista elementów

Element Wartość Obudowa Uwagi

C1 100n C0805
C2 100n C0805
C3 100n C0805
C4 10n C0805
C5 100n C0805
C6 100n C0805
C7 100n C0805
C8 10n C0805
C9 33p C0805
C10 33p C0805
C11 1u C0805
C12 1u C0805
C13 1u C0805
C14 1u C0805
C15 100n C0805
C16 100n C0805
C17 100n C0805
C18 1u C0805
C19 100n C0805
C20 100u SMC B
D1 Z5V6 MELF Dioda zenera 5v6
IC1 TL084D SO14
IC2 TL084D SO14
IC3 ATMEGA48-16AU TQFP32-08
IC4 MAX232ECWE SO16
LED1 CHIPLED 1206 kolor niebieski
LED2 CHIPLED 1206 kolor niebieski
Q1 16MHz HC49/S niski z podkładką
Q2 BC807 SOT23 BEC
R1 2k2 R0805
R2 22k R0805
R3 1k R0805
R4 1k R0805
R5 22k R0805
R6 1K R0805
R7 1K R0805
R8 150k R0805
R9 1K2 R0805
R10 62K R0805
R11 2k2 R0805
R12 22k R0805
R13 1k R0805
R14 22k R0805
R15 1K R0805
R16 1K R0805
R17 150k R0805

R18 1K2 R0805
R19 62K R0805
R20 1k R0805
R21 10K R0805
R22 500R R0805
R23 500R R0805
R24 1K2 R0805
SV4 MA06-1 goldpin
U$1 40R F/CM12P odbiornik 12mm 40KHz
U$2 40R F/CM12P odbiornik 12mm 40KHz
U$3 40T F/CM12P nadajnik 12mm 40KHz

Załącznik 1:
Przykład obsługi sonaru dla mikrokontrolera AVR

Kody źródłowe plików przeznaczone są dla mikrokontrolera rodziny AVR - Atmega16 taktowanego zegarem 8MHz. Kody można dostosować dla dowolnego mikrokontrolera AVR wyposażonego w sprzętowy interfejs SPI. Należy przestrzegać prędkości taktowania interfejsu SPI i wyprowadzenia sterującego pinem SS. Sonar należy podłączyć do interfejsu SPI, pin SS sonaru należy podłączyć do pinu SS Atmegi16 - PortB.4

```
#define Nop 0xF0
#define cbi(add,bit) ((add) &= ~(1 << bit));
#define sbi(add,bit) ((add) |= (1 << bit));
#define Liczba_bajtow_inicjujacych 4
#define Najdluzsza_ramka 4

void Inicjalizacja_SPI_Master(void); //konfiguruje  interfejs
unsigned char Inicjuj_sonar(void);  //inicjuje  sonar, wykonac po konfiguracji  interfejsu
void Zrob_Pomiar(unsigned char Liczba_strzalek, unsigned int *Lewy,unsigned int *Prawy);
//przed dokonaniem pomiaru wykonac inicjacje

/*Biblioteka  odpowiedzialna za komunikacje po SPI
Wykorzystuje sprzetowy interfejs  SPI, oraz uruchamia przerwania*/



#include "SPI.h"
#include <avr/io.h>       //nazwy portow rejestrow itp.
#include <avr/interrupt.h> //przerwania



static volatile unsigned char Licznik_bajtow = 0;
static unsigned char Dane_T[Najdluzsza_ramka];
static unsigned char Dane_R[Najdluzsza_ramka];
static volatile unsigned char Ukonczono_nadawanie;


// obsluga  wektora przerwania
ISR(SPI_STC_vect){
        if (--Licznik_bajtow > 0){
                Dane_R[Licznik_bajtow]=SPDR; //odczytujemy co przyszlo
                //_delay_ms(1);
                SPDR = Dane_T[Licznik_bajtow-1];//wysylamy
        }
        else{                                   //jesli  wyslano wszystko
                Dane_R[0]=SPDR;                 //odczytujemy co przyszlo
                Ukonczono_nadawanie = 1;        //ustawiamy flage
                SPCR &= ~(1 << SPIE);           //blokujemy przerwania SPI
                sbi(PORTB, 4);                  //SS w stan wysoki
        }

}
```

```
void Inicjalizacja_SPI_Master(void){
        sbi(DDRB, 7);  // SCK jako wyjscie
        sbi(PORTB,5);
        sbi(DDRB, 5);  // MOSI i
        sbi(PORTB, 4); // Ustawiamy SS (chipselect)
        sbi(DDRB, 4);  // i ustawiamy jako wyjscie (SS skonfigurowane jako wejscie moze namieszac)
        SPCR = (1<<MSTR)|(1<<SPE)|(1<<SPR0)|(1<<SPR1)|(1<<CPHA);
}   /*wpisujemy do rejestru konfiguracyjnego
        MSTR=1 - SPI w trybie master
        SPE =1 - SPI Enable
        SPR0 i 1 to konfiguracja  dzielnika , tu fosc/128*/



//wysyła 4 bajty FF inicjujące sonar, zwraca wersję oprogramowania sonaru, lub 0 gdy niepodlaczony
unsigned char Inicjuj_sonar(void){
        Licznik_bajtow = Liczba_bajtow_inicjujacych;
        Dane_T[0]=Nop;
        Dane_T[1]=Nop;
        Dane_T[2]=Nop;
        Dane_T[3]=Nop;
        SPCR |= (1<<SPIE); //Przerwanie od SPI mozliwe
        cbi(PORTB, 4);
        Ukonczono_nadawanie = 0; //zerujemy flage
        SPDR = Dane_T[0];       //rozpoczynamy transmisje (wpisanie do rejestru uruchamia nadawanie)
        sei ();                 //globalne zezwolenie na przerwania
        do{}                                    //petla czekajka
        while(Ukonczono_nadawanie < 1);
        return Dane_R[0];       //zwracamy to co dostalim od sonaru w ostatnim bajcie
}

void Zrob_Pomiar(unsigned char Liczba_strzalek, unsigned int *Lewy,unsigned int *Prawy){
        Licznik_bajtow = Liczba_bajtow_inicjujacych;
        Dane_T[0]=Liczba_strzalek;
        Dane_T[1]=Nop;
        Dane_T[2]=Nop;
        Dane_T[3]=Nop;
        SPCR |= (1<<SPIE); //Przerwanie od SPI mozliwe
        cbi(PORTB, 4);      //SS aktywny
        Ukonczono_nadawanie = 0; //zerujemy flage
        SPDR = Dane_T[3];       //rozpoczynamy transmisje (wpisanie do rejestru uruchamia nadawanie)
        sei ();                 //globalne zezwolenie na przerwania
        do{}                                    //petla czekajka
        while(Ukonczono_nadawanie < 1);
        *Lewy = Dane_R[3]<<8;
        *Lewy |= Dane_R[2];
        *Prawy = Dane_R[1]<<8;
        *Prawy |= Dane_R[0];
}
```