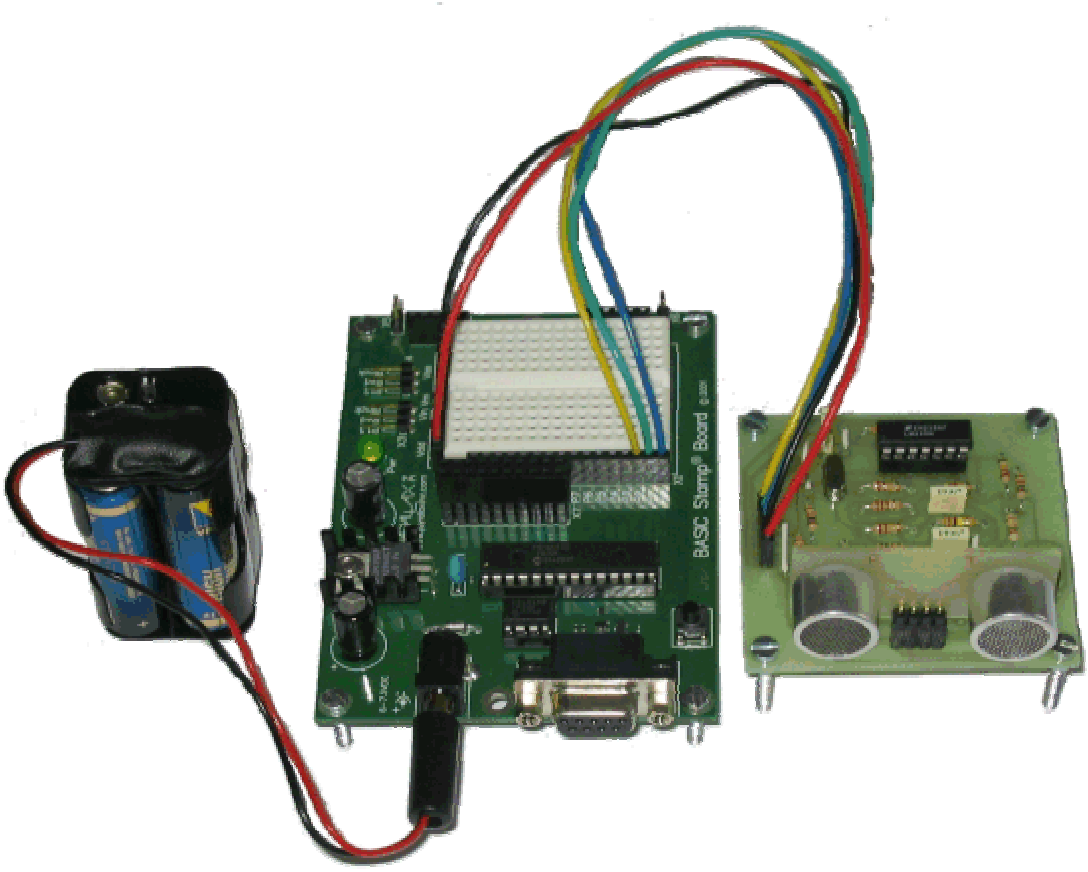
Progetto per la realizzazione
di un sonar

Realizzato da Adriano Gandolfo

Home-page: http://adrirobot.too.it/

Progetto originario all'Home page

http://www.reconnsworld.com/bs2sonar.html



Elenco componenti:

R1	10kΩ -¼W
R2	47kΩ -¼W
R3	22kΩ -¼W
R4	100kΩ -¼W
R5-	10kΩ -¼W
R6	10kΩ -¼W
R7	47kΩ -¼W
R8	22kΩ -¼W
R9	100kΩ -¼W
R10	18kΩ -¼W
R11	10kΩ -¼W
R12	220kΩ -¼W
R13	220kΩ -¼W
R14	100kΩ -¼W
R15	2,2kΩ -¼W
R16	220kΩ -¼W
R17	220kΩ -¼W
R18	100kΩ -¼W
R19	10kΩ -¼W
R20	1kΩ -¼W
C1	0,1µF poliestere
C2	0,1µF poliestere
C3	0,1µF poliestere
IC1	LM358N
IC2	LM339N
JP1	Strip 1x5 femmina
JP2/5	Strip 1X4 femmina
JP3/4	Strip 1X4 maschio 90°
Q1	quarzo da 40 kHz
RCVR1	capsula ricevente per ultrasuoni 40 KHz
XMTR1	capsula trasmittente per ultrasuono 40 KHz

```
'{$STAMP BS2}
' Programma: test_sonar.bs2 (sonar a corto raggio per BS2)
' Questo programma in cooperazione con un una circuiteria trasmittente/ricevente
' sui 40 kHz crea un'economico sonar a corto raggio (10"-250 mm)
' per controllo collisione su piccoli robot.
' Il programma si occupa di far emettere un breve treno d'impulsi a 40kHz (ping)
' e si mette in attesa del ritorno dell'eco.
' L'istruzione del BS2 RCtime valuta il l'intervallo di tempo tra l'emissione del segnale
' e l'arrivo dell'eco.
' I risultati migliori sono compresi nel range tra 2 a 7" (tra i 58 e le 225 unità).
'========CONSTANTI
nSmp  CON  5         ' Number of samples (NOT LESS THAN 5).
maxSmp  CON  nSmp-1  ' Array index # of last sample.
maxSrt  CON  nSmp-2  ' Maximum index # to sort.
s1  CON  nSmp/2-2    ' 1st sorted sample to include in average.
s2  CON  nSmp/2-1    ' 2nd "      "      "  "       "  "
s3  CON  nSmp/2      ' 3rd "      "      "  "       "  "
s4  CON  nSmp/2+1    ' 4th "      "      "  "       "  "
s5  CON  nSmp/2+2    ' 5th "      "      "  "       "  "
ping  CON  0         ' Contatto libero (0) su Scheda madre - Output per attivazione segnale.
pingLen  CON  200    ' Durata del segnale 2-us units.
compRC  CON  2       ' Contatto libero (2) su Scheda madre - Output per settare circuito RC del
comparatore
rcvr  CON  1         ' Contatto libero (1) su Scheda madre - Input da ricevitore/comparatore 40kHz
'========VARIABILI
echTime   VAR  Word
smp       VAR  Byte(nSmp)
index     VAR  Byte
swapTmp   VAR  Byte
swap      VAR  Bit
'========PROGRAMMA
HIGH ping
again:
  GOSUB sonar
  DEBUG "Tempo Echo (0-255 unità): ", DEC echTime,CR
GOTO again
'========SONAR SUBROUTINE
sonar:
FOR index = 0 TO maxSmp
  HIGH compRC
  PAUSE 1
  INPUT compRC
  PULSOUT ping,pingLen
  RCTIME rcvr,0,echTime
  smp(index) = echTime/2 MAX 255
NEXT
sort:
  swap = 0
FOR index = 0 TO maxSrt
  IF smp(index) >= smp((index+1)) THEN noSwap
    swapTmp = smp(index)
    smp(index) = smp(index+1)
    smp(index+1) = swapTmp
    swap = 1
noSwap:
NEXT
IF swap = 1 THEN sort
echTime = smp(s1)+smp(s2)+smp(s3)+smp(s4)+smp(s5)/5 MAX 255
RETURN
```

```
' Program: BS2SONAR.bs2 ( short-range sonar using BS2)
' This program, in cooperation with the 40kHz send/receive circuitry
' creates an inexpensive short-range (10") sonar
' system suitable for collision avoidance in small robots. The program
' instructs the circuit to emit a short burst of 40kHz sound (ping)
' and waits to hear a return echo. The BS2 RCtime instruction times
' the ping-to-echo interval to the nearest 2us. Raw sonar data
' is quite noisy, so the sonar subroutine actually takes several
' samples (5 or more; set by the nSmp constant below). It scales
' the samples to byte size (0-255 units of 4us each), stores
' them in an array, and sorts them from high to low. The program
' then averages the five middle samples. The result is a reading
' that's quite stable (assuming a fixed distance from the sonar
' to the object). Best accuracy is in the range of 2 to 7 inches
' (about 58 to 225 units).
'========CONSTANTS
nSmp    con     10              ' Number of samples (NOT LESS THAN 5).
maxSmp  con     nSmp-1          ' Array index # of last sample.
maxSrt  con     nSmp-2          ' Maximum index # to sort.
s1      con     nSmp/2-2        ' 1st sorted sample to include in average.
s2      con     nSmp/2-1        ' 2nd "       "       "  "       "  "
s3      con     nSmp/2          ' 3rd "       "       "  "       "  "
s4      con     nSmp/2+1        ' 4th "       "       "  "       "  "
s5      con     nSmp/2+2        ' 5th "       "       "  "       "  "
ping    con     X               ' !!!!! FREE STAMP PIN '0' !!!! Output to activate pinger.
pingLen con     200             ' Duration of ping in 2-us units.
compRC  con     X               ' !!!!! FREE STAMP PIN '1' !!!!Output to set comparator RC circuit.
rcvr    con     X               ' !!!!! FREE STAMP PIN '2' !!!! Input from 40kHz receiver/comparator.
'========VARIABLES
echTime var     word            ' Time to echo return.
smp     var     byte(nSmp)      ' Storage for multiple readings.
index   var     byte            ' Counter for sampling.
swapTmp var     byte            ' Temporary storage for swapping.
swap    var     bit             ' Flag to indicate whether sort is done.
'========PROGRAM
' The "again" loop takes sonar ranges continuously and displays
' them on the PC's debug screen.
high ping               ' Turn pinger off initially.
again:                  ' Loop.
  gosub sonar           ' Take the sonar reading
  debug "Echo time (0-255 units): ", dec echTime,cr ' Display it.
goto again              ' Repeat endlessly.
'========SONAR SUBROUTINE
' It takes only five instructions to get a quick sonar snapshot of the
' distance to the closest sonar-reflective object. However, you can get
' better, more consistent results by taking several sonar readings,
' discarding the highest and lowest ones and averaging the middle.
' This routine takes the number of samples specified by the constant
' nSmp, sorts them, and averages the middle. Each reading takes
' only a few milliseconds (owing to the sonar's short range).
sonar:
for index = 0 to maxSmp             ' Take nSmp samples.
  high compRC                       ' Raise C2 to +5 volts.
  pause 1                           ' Allow time for C2 to reach +5V.
  input compRC                      ' Disconnect pin from C2.
  pulsout ping,pingLen              ' Send a short 40kHz pulse.
  rctime rcvr,0,echTime             ' Wait for echo; save time to echTime.
  smp(index) = echTime/2 max 255    ' Save to array smp() as byte (0-255).
next                                ' Get another sample.
' At this point, there are nSmp sonar samples stored in the bytes of
' the smp() array. One way to discard the lowest and highest samples
' is to sort the array so that the lowest index values contain the
' largest numbers. The code starting with "sort" does this using
' a technique called "bubble sort." The idea is simple--compare
' adjacent bytes in the array, for instance smp(0) and smp(1).
' If the value stored in smp(0) is greater than or equal to that in
' smp(1), do nothing. Otherwise, swap the values so that smp(0)
' gets the contents of smp(1), and vice versa. Keep doing this
' with each pair of values in the array. The larger values in the
' array will migrate toward the lower index values--they rise
' like soda bubbles. Repeated passes through the array will
' completely sort it. The routine is done when it makes a loop
' through the array without swapping any pairs.
sort:
  swap = 0                                      ' Clear flag that indicates swap.
for index = 0 to maxSrt                         ' For each cell of the array...
  if smp(index) >= smp((index+1)) then noSwap   ' Move larger values up.
    swapTmp = smp(index)                        ' ..by swapping them.
    smp(index) = smp(index+1)
    smp(index+1) = swapTmp
    swap = 1                ' Set bit if swap occurred.
noSwap:
next                        ' Check out next cell of the array.
if swap = 1 then sort       ' Keep sorting until no more swaps.
' The line below just averages particular cells of the array. If you
' use my values of the constants s1 through s5, it averages readings
' from the middle of the range. By assigning other values to s1-
' s5, you can alter this.
echTime = smp(s1)+smp(s2)+smp(s3)+smp(s4)+smp(s5)/5 max 255
return                      ' Done: return to program.
```

June 30, 2004

Basic Stamp Sonar

Just so you know, the following project is from the book: **Programming and Customizing the Basic Stamp Computer.** Click here, to read more about this title.

Note: This circuit requires the Basic Stamp 2, and 3 I/O ports of that Basic Stamp.

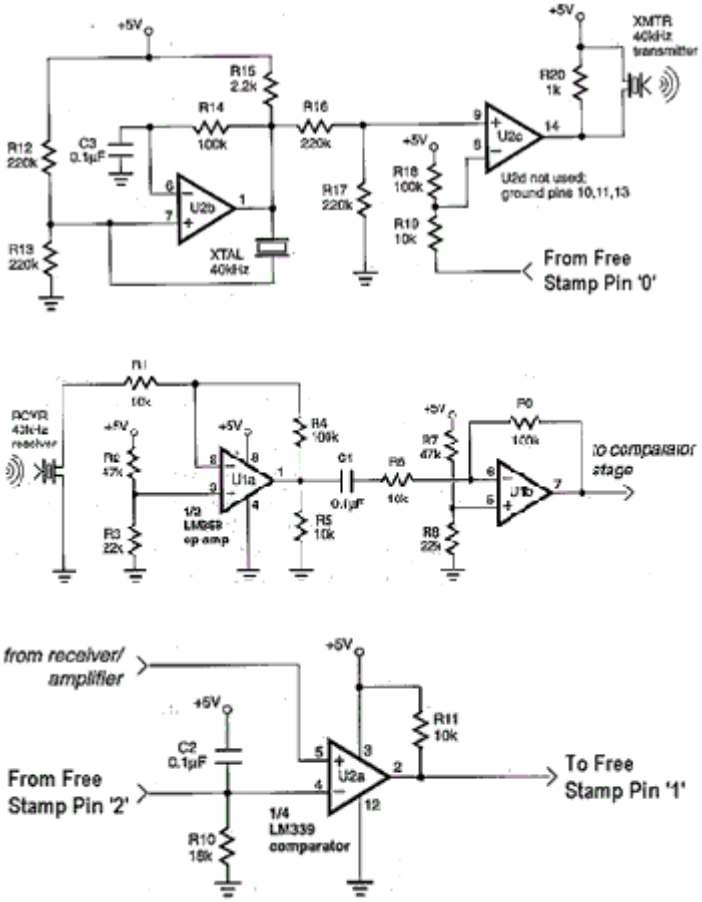
This project is best suited for use as a sensor on a robot. This circuit will actually allow you to measure distances from an object. It measures in a 'unit' and it's range is about 10 inches. As the receiver and transmitter, you will use an ultrasonic transmitter and receiver pair tuned to 40khz (see parts list below.) The circuit is very easy to build, and you can build it on whatever you have laying around. I assembled it onto a Radio Shack pre-etched circuit board - and it only took up about an inch and a half square.

parts:

Resistors (1/4 watt, 10% or better)		
R1, R5, R6, R11, R19, R21	10K	Jameco: 29911 DCkits: CF25-10K
R2, R7	47K	Jameco: 31149 DCkits: CF25-47K
R3, R8	22K	Jameco: 30453 DCkits: CF25-22K
R4, R9, R14, R18	100K	Jameco: 29997 DCkits: CF25-100K
R10	18K	Jameco: ------- DCkits: CF25-18K
R12, R13, R16, R17	220K	Jameco: 30525 DCkits: CF25-220K
R20	1k	Jameco: 29663 DCkits: CF25-1K
Other Components:		
C1, C4	0.1uf ceramic cap 50WVDC	Jameco: 151116 DCkits: 21ET100
RCVR, XMTR	40khz ultrasonic receiver and transmitter	Jameco: 136653 DCkits: --------
U1	LM358AN dual op amp	Jameco: 120862 DCkits: --------
U2	LM339N quad comparator	Jameco: 23851 DCkits: LM339
XTAL	40khz quartz crystal	Digikey: SE3316-ND

The schematics are below. Make sure you connect everything correctly, then after that is done move onto the basic stamp code below.

Click on the thumbnails to enlarge the schematics.



Basic Stamp Code:

What will it do?
It will open a debug window in the basic stamp compiller, and display distance readings (in 'units') from the circuit.

Please note that the 'Free Stamp Pin' in the schematics refer to the basic stamp i/o port that you connect that part of the circuit to. You will need to look out for the the three (!!!! FREE STAMP PIN ' ' !!!!)'s in the code, and change the X's that accompany them to the correct i/o port number. Download the source code by

1.) Right-click on the link and select 'Save Target As..' (Windows)

2.) Click on the link, and hold it until a menu pops up, and select 'Save Target As...' (Mac)

bs2sonar.txt
OR
bs2sonar.bs2

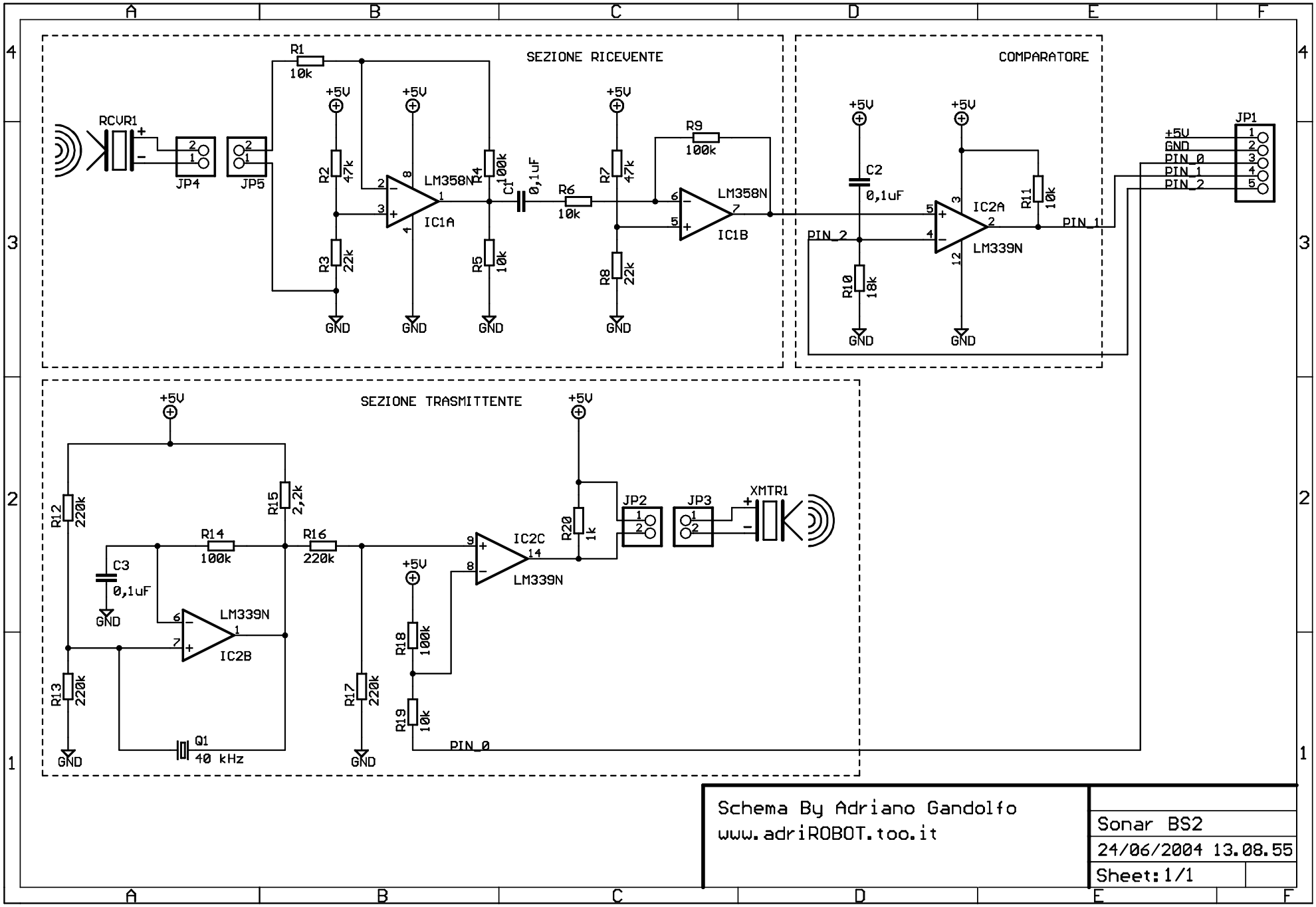
Refer to the SmallBot Programs page to see how to implement this circuit as an ultrasonic sensor for your robot.

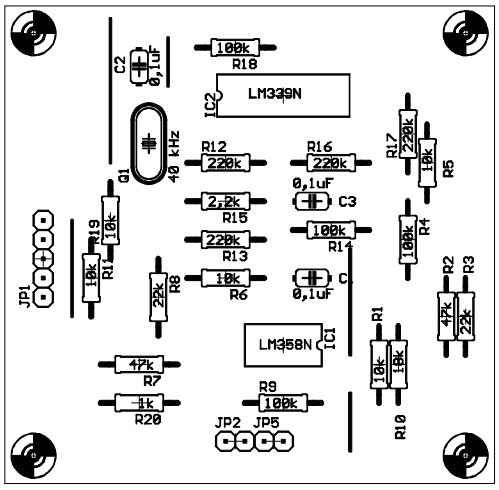
Related Stuff:

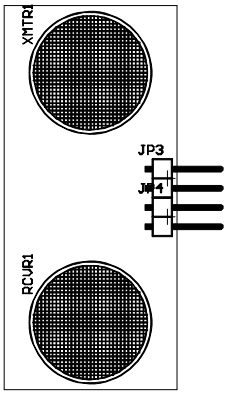
Robotics - Check out all the Robotics content reconnsworld.com has to offer. Plans, Pictures, Programs, and more!

Robotics and EE Webring - Browse the ring and find more great Robotics and Electrical Engineering sites. Or join the ring and increase your traffic.

Reconn's World Store - Offering you the best titles in robotics and electronics books and magazines, Reconn's World strives to make the online shopping experience as easy, and comprehensive as possible.


SEZIONE RICEVENTE
COMPARATORE
SEZIONE TRASMITTENTE
RCVR1
JP4
JP5
R1
10k
R2
47k
R3
22k
R4
100k
C1
0,1uF
R5
10k
R6
10k
R7
47k
R8
22k
R9
100k
LM358N
IC1A
IC1B
C2
0,1uF
R10
18k
R11
10k
IC2A
LM339N
PIN_1
PIN_2
JP1
+5V
GND
PIN_0
PIN_1
PIN_2
R12
220k
R13
220k
R14
100k
R15
2,2k
R16
220k
R17
220k
R18
100k
R19
10k
R20
1k
C3
0,1uF
IC2B
IC2C
LM339N
Q1
40 kHz
JP2
JP3
XMTR1
Schema By Adriano Gandolfo
www.adriROBOT.too.it
Sonar BS2
24/06/2004 13.08.55
Sheet: 1/1





WWW.adriROBOT.too.it by Adriano Gandolfo	
Date: 24/06/2004 13.12.28	
TITLE: Sonar BS2	Sheet: