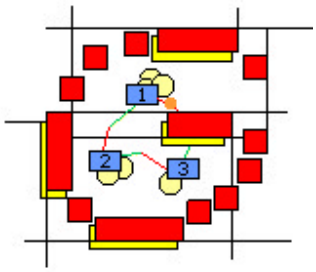

APPLICATION NOTE AN101
Sonar Sensor on PIC16F628
Version 1.0

Date: 27 May 2003

www.generexe.com

1. Introduction

This application note describes a practical sonar system, using ultrasonic sound waves to determine the distance to obstacles, using a Microchip PIC16F628 MCU.

2. Principle of Echo Location

Sonar, like radar, uses the principle of echo location. For echo location, a short pulse is sent in a specific direction (XMIT in figure 1). When the pulse hits an object, which does not absorb the pulse, it bounces back, after which the echo can be picked up by a detector-circuit (RECV in figure 1).

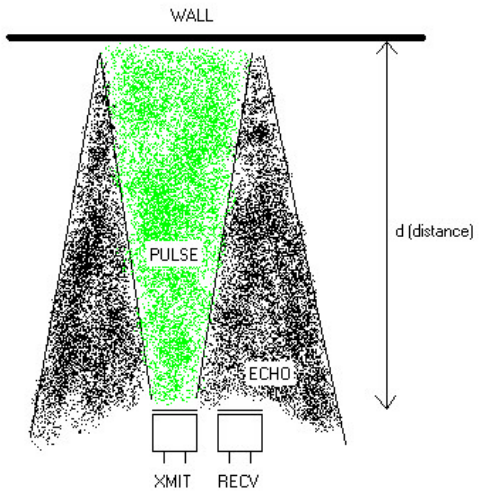

Figure 1. Principle of echo location.

By measuring the time between sending the pulse and detecting the echo, the distance to the object can be determined. I.e. sound travels at a speed of 343 meters per second through air at room temperature. By multiplying the time between pulse and echo (in seconds) with 343, you will get *twice* the distance to the object in meters (since the sound traveled the distance twice to get to the object and bounce back):

$$2d = V_{sound} \text{ X } (T_{pulse} - T_{echo})$$

where:

- V_{sound} = speed of sound-travel (343 meters/second)
- T_{pulse} = time in seconds of pulse transmission
- T_{echo} = time in seconds of echo detection
- d = distance to object onto which pulse bounces back.

3. Circuit description

Figure 2 shows a standard PIC16F628 circuit, with 4.7kOhm pull-up resistor on MCLR to provide a reset-signal and a 10mHz resonator to provide the clock-signal.

Note that the 4.7kOhm resistor can be left out if you configure the MCLR pin as an I/O pin (in the hardware configuration panel of XPad). Also

many resonators only have 2 leads, instead of 3. In this case, you will have to ground the 2 leads on OSC1 and OSC2, via 20pF capacitors.

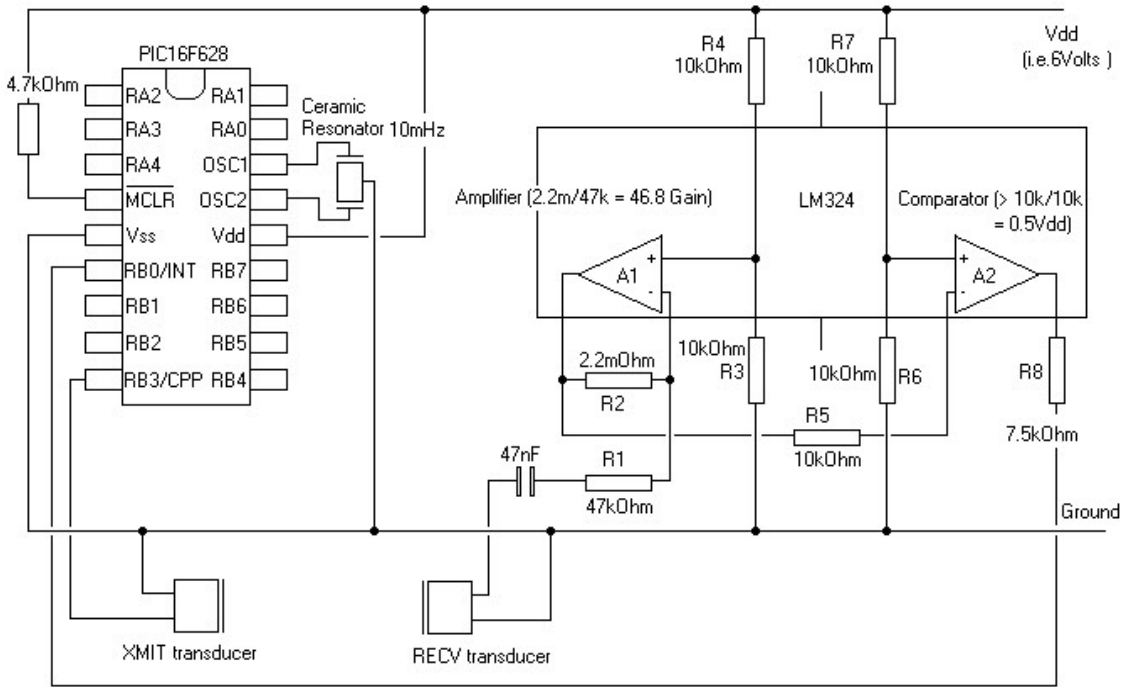

Figure 2. The sonar sensor circuit, with on the right the detector.

The CPP-pin on RB3 is used to generate a 40kHz pulse fed directly to the XMIT transducer, generating the sound wave. The CPP-module of the PIC16F628 will be put into PWM mode for this.

The external interrupt pin on RB0 is used to input the output of the detector circuit, shown on the right in figure 2. The detector circuit is built around the LM324 IC, which provides 4 low power operational amplifiers. Only 2 opamps are used in this circuit.

The tiny signals from the RECV transducer is fed into A1, which is setup as an inverting amplifier, providing a gain of –R2/R1 = 2.2mOhm/47kOhm = -46.8. Note that this gain is low enough to pass 40kHz signals, without introducing noise. The negative sign is not a problem, since the RECV transducer produces an AC-signal. Many similar circuits include a tone-decoder –i.e. a 567 IC- to filter for the 40kHz frequency. However, this really is not necessary, since the RECV transducers are naturally sensitive to signals around 40kHz only.

R3 and R4 form a voltage divider lifting the input signal up to halve of Vdd. The reason for this is that the input signal is true AC, and the circuit has a single positive power supply.

The amplified signal from A1 is fed via current-limiting resistor R5 into the negative input of a comparator, which is setup around A2. The positive input of A2 is fed with a voltage determined by the voltage divider of R6 and R7. Since

R6 and R7 are both 10kOhm, A2 will only go positive when the negative input exceeds halve of Vdd.

Note that the LM324 power-lines are not shown in figure 2, nor are any pin-numbers. The reason is that the 14-pin LM324 package is symmetrical. Each side of the LM324 contains 6 pins for 2 OpAmps opposite to each other. The pins in the middle are the Vdd and Ground lines of the IC. Depending on how you build the circuit you can use any 2 OpAmps of the LM324.

To test the circuit, I also connected a DS275 RS232 level-converter to RB1 (RX) and RB2 (TX) to use the USART of the PIC16F628. Figure 3 shows how to wire the DS275 into the circuit of figure 2.

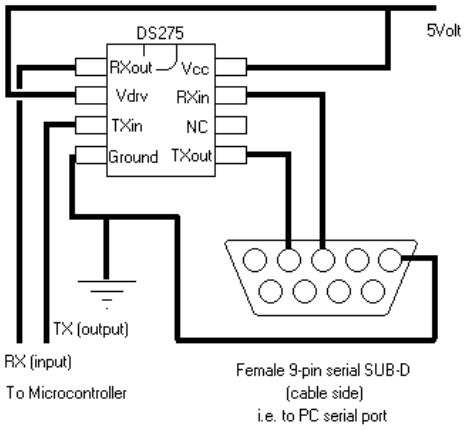

Figure 3. RS232 level-converter circuit.

4. Software Design of Sonar

Figure 4 shows the single state diagram that is used to test the sonar circuit.

The IDLE-state sets up the serial port and waits for any character. When a character has been received the KEY state is entered, which does the following:

- send “PING” back over serial port.
- reset the Ticks-variable, which is used to measure time between pulse and echo-detection.
- Start a pulse on RB3 at 40kHz and 50% duty-cycle (each pulse-period takes 25usec).
- Wait 400usec (good for 400/25 = 16 pulses).
- Stop the pulse on RB3.
- Wait another 400usec, to make sure that the transducer has stopped vibrating (and that no direct coupling from the transmitter is mistaken for a true ECHO in the detector-circuit).
- Start the timer to count Ticks until the ECHO is detected. The timer is setup for 102usec timeouts, meaning that our distance-measurement will have a resolution of $2d = 343 \times 102.10^{-6}$. Hence resolution d = 1.75 cm (3/4 inch).
- Finish the state and wait for Timer-ticks.

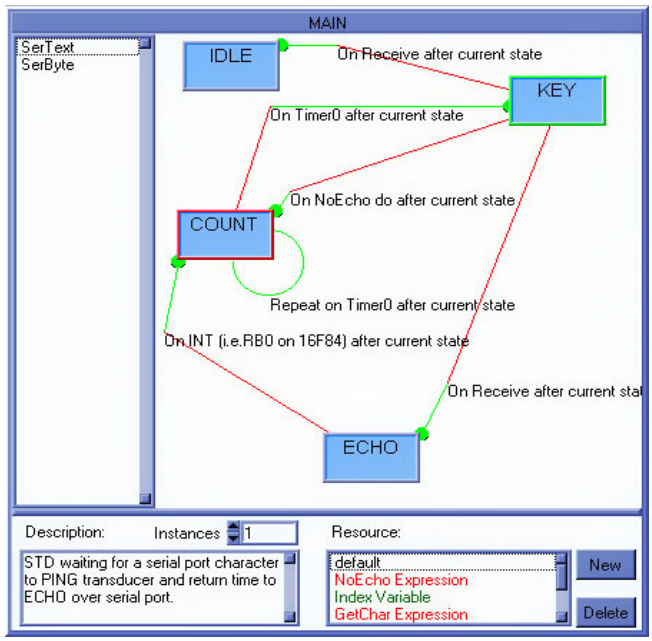

Figure 4. State Diagram for Sonar circuit.

Now every 102usec, the COUNT state is re-activated, which just increases the

Ticks-variable by 1 (indicating an extra 1.75cm to the object). Besides re-activating the COUNT state, the State Diagram can go into the ECHO-state when an ECHO was detected on the RB0 pin or –when the Ticks-variable exceeds the value of 100- go back to the KEY state.

The latter transition is tested by the NoEcho expression and –when true- assumes that no ECHO will come altogether; because no objects are close enough to generate a sufficiently strong ECHO passing the detection-circuit. When this happens NoEcho will put the software back into the KEY state, immediately retrying the PING.

When an ECHO was detected the ECHO state is entered, which stops the timer, transmits a newline character on the RS232 port, followed by a hexadecimal string-representation of the value in the Ticks-variable! I.e. a value of 11 (hexadecimal) would indicate that the object in front of the sensor was 17(decimal) X 1.75cm = 29.75cm (approx. 1 foot) away.

The XPad-model can be downloaded for free from http://www.generexe.com.

5. Conclusions and improvement

The lower detection-limit in the software is determined by the minimum delay between the PULSE and that an ECHO can be detected. This is 400usec (in the KEY state) + 102usec (to pass through the COUNT state at least once) = 502 usec. The sound may have traveled approx. 5 X 1.75 cm = 8.75 cm. Since this is 2-way, we are talking about a minimum detection-distance of almost 4.37 cm. Strictly spoken the Ticks-variable should be initialized with 4 in the KEY-state to account for the 400usec we are waiting before starting the timer.

The NoEcho timeout check of 100 determines the higher limit. 100 ticks would mean 175 cm (almost 6 feet). Increasing the 100 in the NoEcho expression, will allow for detection of objects further away. However, it will also take longer to restart a PULSE in case no objects are nearby enough to result in a sufficient ECHO.

When used for some kind of robotic vision application, the range of the shown circuit is perfect. To detect objects at longer distances, you can simply increase the range of the circuit, by increasing power provided to the XMIT transducer. These things can take up to 20V. (In the shown circuit the transducer is only fed with 5V signals directly from the CPP-pin of the PIC16F628). The easiest way to do this is via a transistor, which switches a larger power supply and/or combined with a transformer, jacking up the voltage. Make sure that any components you put in here are able to follow the 40kHz signal, generated by the CPP-pin!

Note that increased transmitter power, also increases sensitivity to close by smaller objects. As can be seen from figure 1, the sound-‘cone’ coming back from a wall is pretty large; smaller objects give much smaller echoes.

You can download a copy of XPad from http://www.generexe.com. Low cost compiler modules (USD40 or less) can be downloaded separately from the same site. Limited compiler modules can be tried for free.