ParkAid

Car 1, scratches 0.

By **Terry Hinrich** (USA)

Parking in garages can sometimes be tricky as it is often difficult to know where exactly the end of the car is. A tennis ball on a string or a board on the floor can help to find the right spot, but hanging a tennis ball at the correct position or keeping the board from moving make these solutions unreliable. For this reason an ultrasonic parking assistant has been developed. "ParkAid" solves the parking problem in a simple and inexpensive way. For just about $40 (€30), enclosure and power supply included, we can be sure to park our car the way it deserves.

The name says it all

ParkAid uses an ultrasonic transmitter-receiver module, a small circuit board that transmits and receives a 40 kHz audio signal. This is well above the limit of human hearing, which is below 20 kHz for most of us, and experience has shown that this frequency is also too high for dogs. The transmitter sends a short pulse sequence; the receiver listens for pulses ("echoes") that are reflected by an object – the car – in front of the module. The module does most of the signal processing, and all it needs is a trigger pulse to launch a measurement. On its output pin it provides the received echoes. By measuring the time between the pulses and their reflections, the distance can be calculated easily. In the event that there is no object in front of the "camera", a maximum distance pulse is provided. A small microcontroller can be used to initiate the transmission and measure the time between echoes.

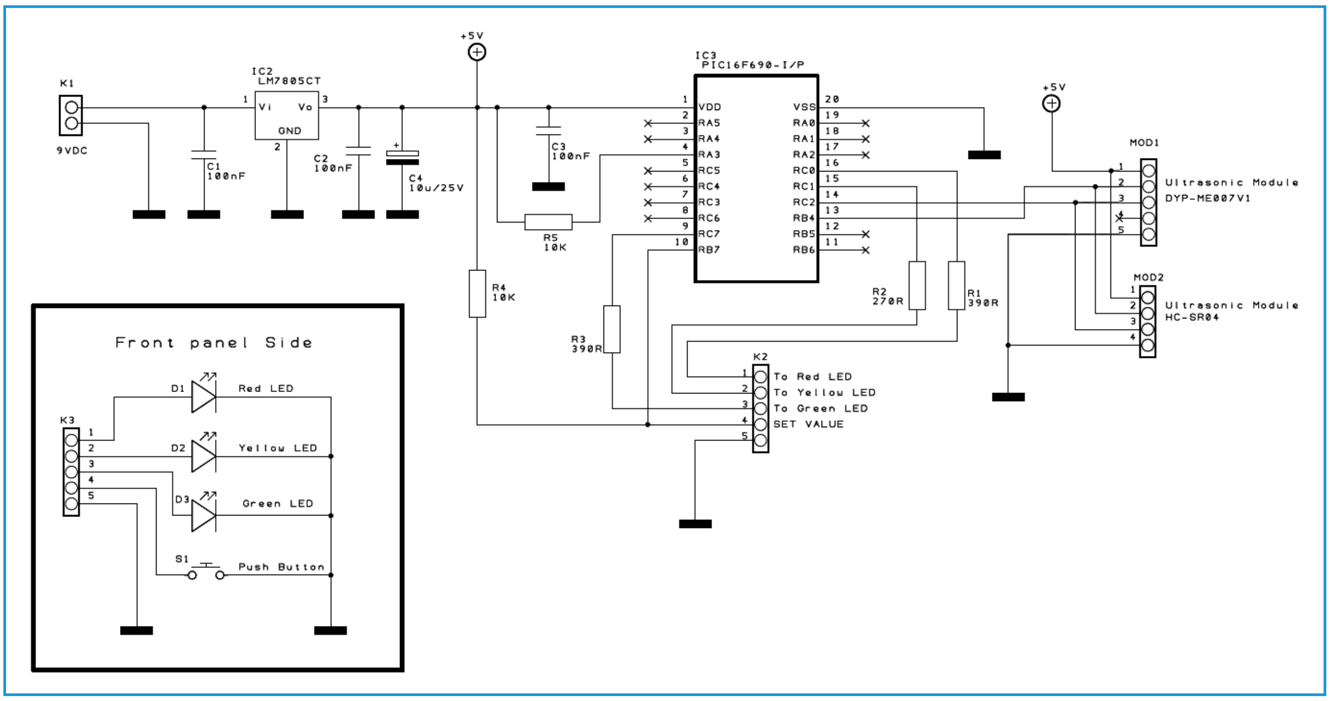


Figure 1.
Schematic of the ParkAid.

One inch, 2.54 cm, is a good resolution for our system. A pulse moving at the speed of sound would travel out to an object one inch away and back in 148 µs. So, after initiating a transmission, the program counts 148 µs intervals to determine how far away the reflecting object is. This method turned out to be quite accurate. It should be noted that the transmitted signal has a cone shape that becomes wider as it travels away from the transmitter. At some point it may bounce off from an unintended object, possibly limiting the maximum range of 3 m (about 9 ft). However, this doesn't seem to be a problem in practice, as the car is always detected at a sufficient distance. Both the author and his wife were able to park the car at the same spot each time.
For this project, it would be helpful to first show how "ParkAid" is supposed to be used, so we will make a *tour* through the "display" (the LEDs), the microcontroller and the firmware before explaining the circuit itself. The latter is depicted in **Figure 1**, and may be downloaded as a DesignSpark PCB project from the Elektor.LABS page of this project [1].

Green, yellow and red: a universal language

This project has three LEDs to indicate the relative distance. The green LED signals that the car is too far away from the unit. The yellow LED shows the position to stop the vehicle, and the red LED indicates that the car is too close and needs to back up a bit.

There are actually seven ranges displayed by these three LEDs:

- Solid green is on as you enter the garage and indicates that you may continue advancing.
- The green LED starts flashing when you are within 12 inches of your target.
- Solid green/yellow occurs one inch away from the target position. We should prepare to stop.
- Solid yellow means that you have reached the target position. This is actually a one-inch wide zone and is easy to hit.
- When both the yellow and red LEDs light up you are moving away from the target position. Please back up slightly.
- Solid red indicates that you have left the target zone and that you must back up.
- Flashing red indicates that you are much too close to the unit. Back up now or you will destroy the system.

This use of lights makes it simple to leave your vehicle in the same spot every time. It is easily done in software, but would be very cumbersome to accomplish in hardware alone. The ranges can be changed in seconds and are stored in the microcontroller's EEPROM.

After powering up the unit, all three LEDs light up briefly to indicate that they are working correctly. Power can be removed and re-applied at any time and the system will begin to perform immediately.

Figure 2 shows the finished system of the author with the three LEDs sticking through the top. The two large round things below the LEDs are the ultrasonic transmitter and receiver. On the left side is the push-button and on the right is the wall transformer connection. The transformer itself appears above the unit.

A PIC is in charge here

The microcontroller used is a PIC16F690 [2] from Microchip. This is a 20-pin chip and has more features than are needed for this application. The author chose it because he had received it for free when he ordered a PICKit 2 programmer from Microchip. Decisions sometimes can be easy. The PIC16F690 is fast, has plenty of I/O pins, enough memory, on-chip EEPROM, and is also inexpensive as it is available for under $2 (about €1.50) from several suppliers. Those who prefer other PICs or microcontrollers from other manufacturers should have no problem making appropriate software changes.

The PIC microcontroller runs from its internal 8 MHz clock, so no external timing components are needed. A single interrupt routine is used to flash the LEDs at a quarter second rate. All other timing is taken care of using wait loops. Debouncing the push button is also handled by the software.

Your wish is my command

To write the code, the HiTech PICC C Lite compiler [3] has been used. This is a free version that lacks the optimization options of the professional version. For a hobbyist with limited funds, this is a fair tradeoff. The source code is straight-forward enough to port to other languages as well. The source code is available on-line, via the Elektor.LABS page of this project [1]. If you want to see how the code works or would like to make changes to it, you're welcome to. We just kindly ask you to give credit for the original. In case you don't have the compiler but you do have a programmer, the HEX file is available too.

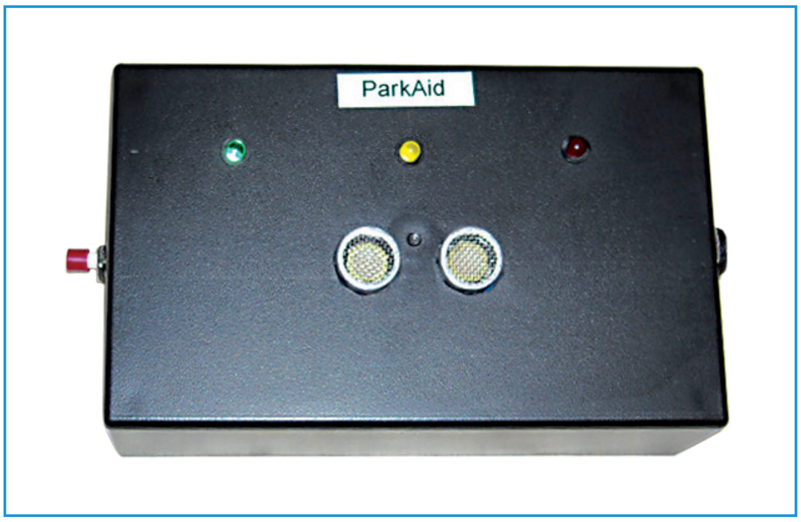


Figure 2.
The finished unit of the author, looking great in its enclosure.

The source code is formatted in a slightly unusual way. The main function is at the end of the file. It invokes the functions that appear above it, and by doing things this way, the functions prototypes can be omitted. Also note how the functions are grouped by functional purpose with alphanumeric prefixes so that they all appear in alphabetical order, making them easy to find. This problem is not as big with small programs as it is with large ones, but it is still helpful.

Ultrasonic module: the eyes of the board

For his prototype the author had used a TS601P01 ultrasonic module. It has a 3-pin interface: 5V power, ground and a bi-directional data pin. However, when we at Elektor.LABS started building our prototype we found that the TS601P01 was rather hard to get. Searching the internet we came up with two other modules that seem to be more common: the DYP-ME007 and the HC-SR04. These modules both use a 4-pin interface even though the first has five pins. The MOD1 connector shown in Figure 1 is intended for the DYP-ME007 module, the MOD2 connector is for the HC-SR04. Technical details for both modules are available in their datasheets, which may be downloaded from the Elektor. LABS project page [1].

The software has been adapted to work with a 4-pin ultrasonic module.

Power

A 7805 voltage regulator without a heat sink provides 5 V, the only voltage needed for this unit. With one LED on a current of about 15 mA was measured, a value of about 28 mA was observed with all three LEDs on. This is easily handled by a 78L05 if preferred. Because of the low current consumption and the desire to have the unit always available, a power switch has not been included. To supply permanent power, a 9 V wall transformer was chosen, but any transformer providing from 9 V to 15 V DC should work just as well.

Circuit board(s)

For his prototype, the author used a Radio Shack model 276-170 printed circuit board. It is a single-sided pre-drilled board of about 2 by 6 inches (5 x 15 cm) suitable for ICs and point-to-point wiring. **Figure 3** shows his complete setup before closing the case. The circuit board is fixed to the removable bottom of the case with 15 mm spacers and the LEDs are on the top face.
We at Elektor designed a set of two PCBs (**Figure 4**), making it even easier to replicate the system. The ultrasonic sensor board, the microcontroller, the voltage regulator and most of the components are located on the main board, whereas the LEDs and the push button are placed on the daughter board. Like the schematic, the DesignSpark PCB files of the board may be downloaded from Elektor. LABS [1].

Adjusting the target position

What if you bought a new car or if you want to change the ranges or target position? A simple procedure allows you to do this. It is probable that you have to execute it when you install the system because the default values are unlikely to be the desired ones.

Pressing S1 puts the unit in program mode. The red and yellow LEDs flash to indicate that you are setting the target position. You should have parked your vehicle already at the desired position prior to this. If S1 is pressed twice within one second, the "near" target position will be stored (yellow LED). This also automatically adjusts the width of the red and green ranges so no other change is necessary. To acknowledge that the new target value was saved, the red and yellow LEDs flash three times. You can now adjust the "far" position. If you do not wish to change the target position press the switch once to proceed to the "far" adjustment point.

The "far" position is the position where the green LED switches from always on to flashing. Setting the target position automatically changes the "far" position, by default about 4 inches (10 cm) from the target position. Should you wish to change this point, you can move your vehicle further away from the unit and then push the button twice within a

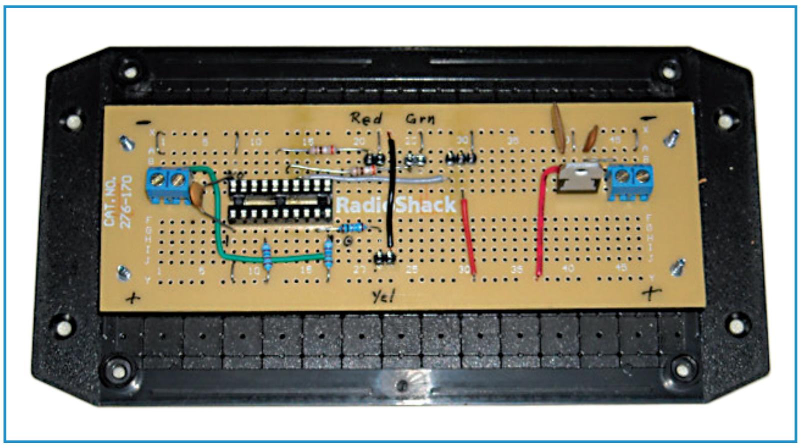


Figure 3.
The complete prototype of the author before closing the lid. The Elektor design is slightly different.

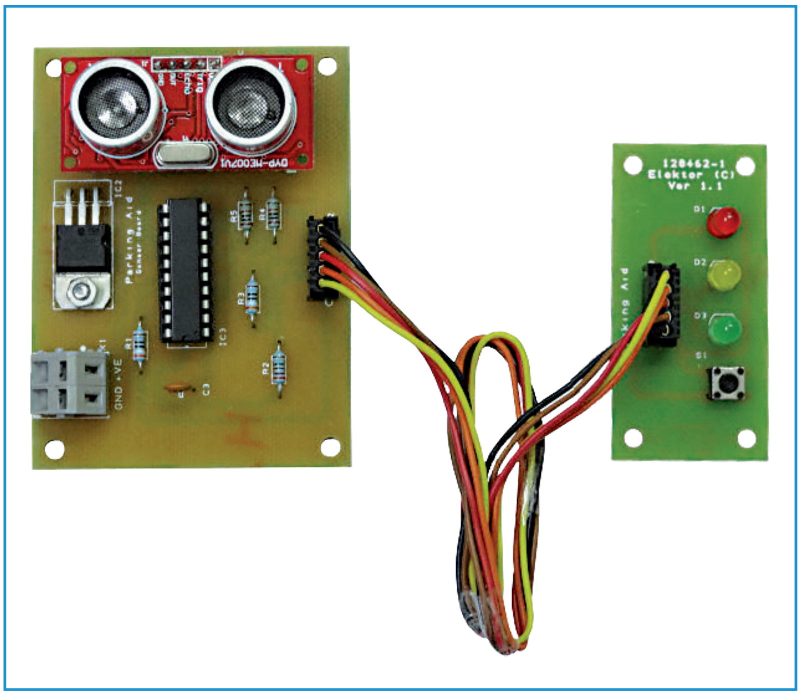


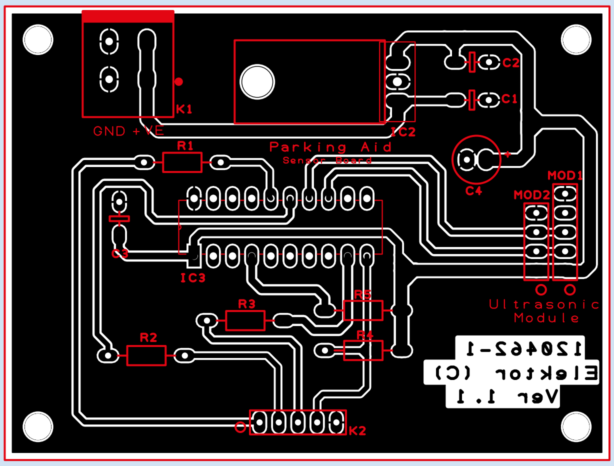
Figure 4.
Assembled circuit boards redesigned at the Elektor Lab. The split design makes it easy to mount the indicator LEDs in a visible place, away from the ultrasonic sensor.

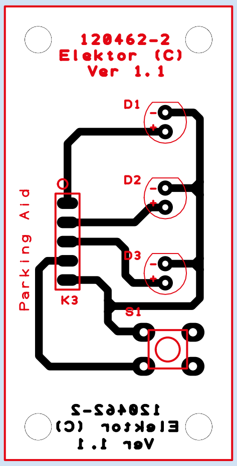
COMPONENT LIST

Resistors
R1,R3 = 390Ω, 0.25W, 5%
R2 = 270Ω, 0.25W, 5%
R4,R5 = 10K, 0.25W, 5%

Capacitors
C1,C2,C3 = 100nF, ceramic
C4 = 10µF, 25V, electrolytic, radial

Semiconductors
D1 = LED, 5mm, red
D2 = LED, 5mm, yellow
D3 = LED, 5mm, green
(IC1 not implemented)
IC2 = 7805 (Fairchild Semiconductor LM7805CT)
IC3 = PIC16F690 (Microchip Technology)





Miscellaneous
K1 = 2 pin PCB terminal block, 5mm (Multicomp MC000046)
K2,K3 = 5 pin header connector, 2.54mm
S1 = tactile switch, 1.2cm X 1.2cm (Multicomp MCDTS2-1N)
MOD1 = 4 pin Sil connector, 2.54mm
MOD2 = 5 pin Sil connector, 2.54mm
Ultrasonic sensor board, DYP-ME007 or HC-SR04 (see text and [1])

second to save this new distance.

When a new point is set, it is stored immediately. The distance is also recalculated so you have to be sure not to be standing in front of the unit at that moment. When you have gone through the two program steps, the unit automatically returns to normal mode.

The “near” and “far” target distances are saved in the EEPROM so that they can survive power cycles.

What comes next?

The nice thing of playing with ultrasonic module is that there is always room for new ideas, as it is easy to come up with new applications.

A possible improvement might be to make the unit work from batteries. Power consumption is pretty low, but this would be an interesting exercise, as the unit is supposed to be always on. Therefore, it must sleep most of the time and wake up periodically to see if the distance has changed.

In any case, now we should be able to park our car in the garage without any problem. And if some day there’s a new scratch in the car... this time we have no excuses!

(120462)

Internet Links

[1] www.elektor-labs.com/120462
[2] www.microchip.com/wwwproducts/Devices.aspx?dDocName=en023112
[3] www.htsoft.com

About the author

Terry Hinrich has recently retired with an interest in electronics as a hobby. He holds a degree in Mathematics and his nearly 50-year career was spent in various aspects of computer software. He has written programs in many languages for mainframe and medium-size computers during that time and he uses Delphi (Pascal based) on his home computer. However, “ParkAid” is his first foray into the very small processors. This project has been very practical for him and provided a great learning experience.