# Cylon Voi

# Robotalk from an ATtiny microcontroller

Gert Baars, g.baars13@chello.nl

**Changing your voice into that of a robot is a task that is perfectly suitable for a microcontroller. This circuit shows how this can be performed by a simple setup built around a small ATtiny45 microcontroller.**

A voice changer that emulates a so-called Cylon voice can be implemented with the help of a small microcontroller. Those of you who have watched the (recent or older) TV series of Battlestar Galactica will immediately know what we're talking about. For the non-SF fans amongst you, this is a metallic sounding robot voice.
The circuit can be used as a gadget, but was originally designed to show how a simple digital circuit could be used for audio processing.

## Hardware

The hardware (**Figure 1**) consists of a pre-amplifier built round opamp IC2, which has a gain of about 70. This is sufficient to amplify microphone signals. R1 and K1 can be used to supply the microphone with a DC voltage, which is required by electret types.
The pre-amplifier feeds the amplified signal to the ADC input of the ATtiny45 controller, which then processes it.
Since the clock for the controller doesn't have to be very accurate the internal RC oscillator has been used as the clock generator. It has an output with a frequency of about 8 MHz,

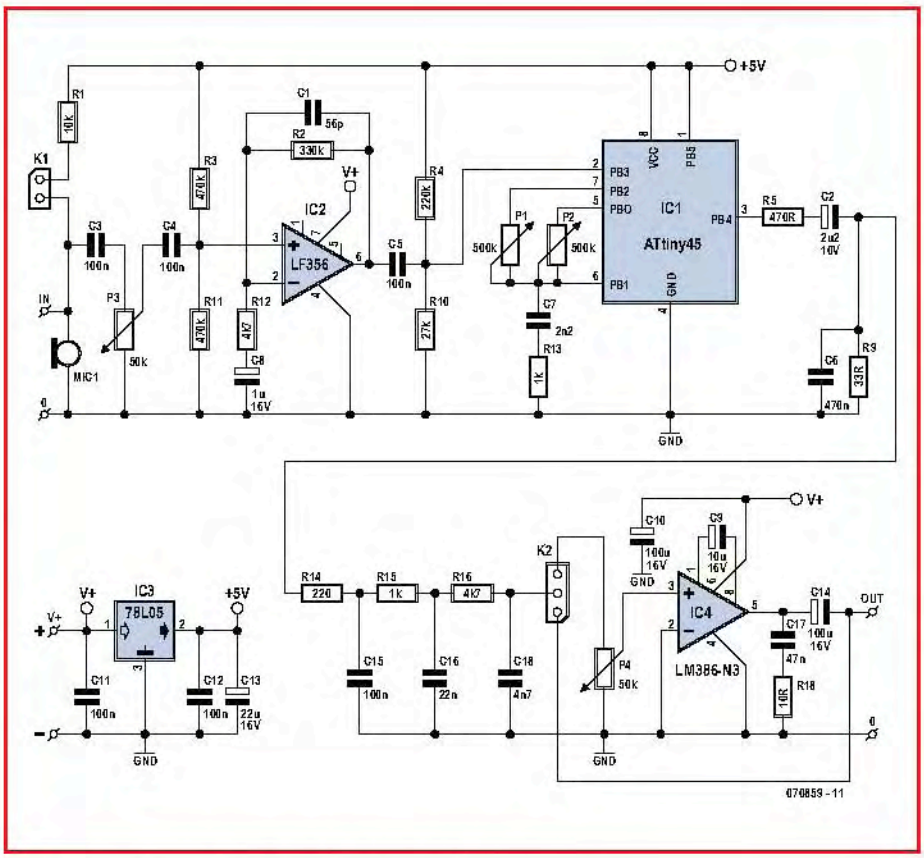
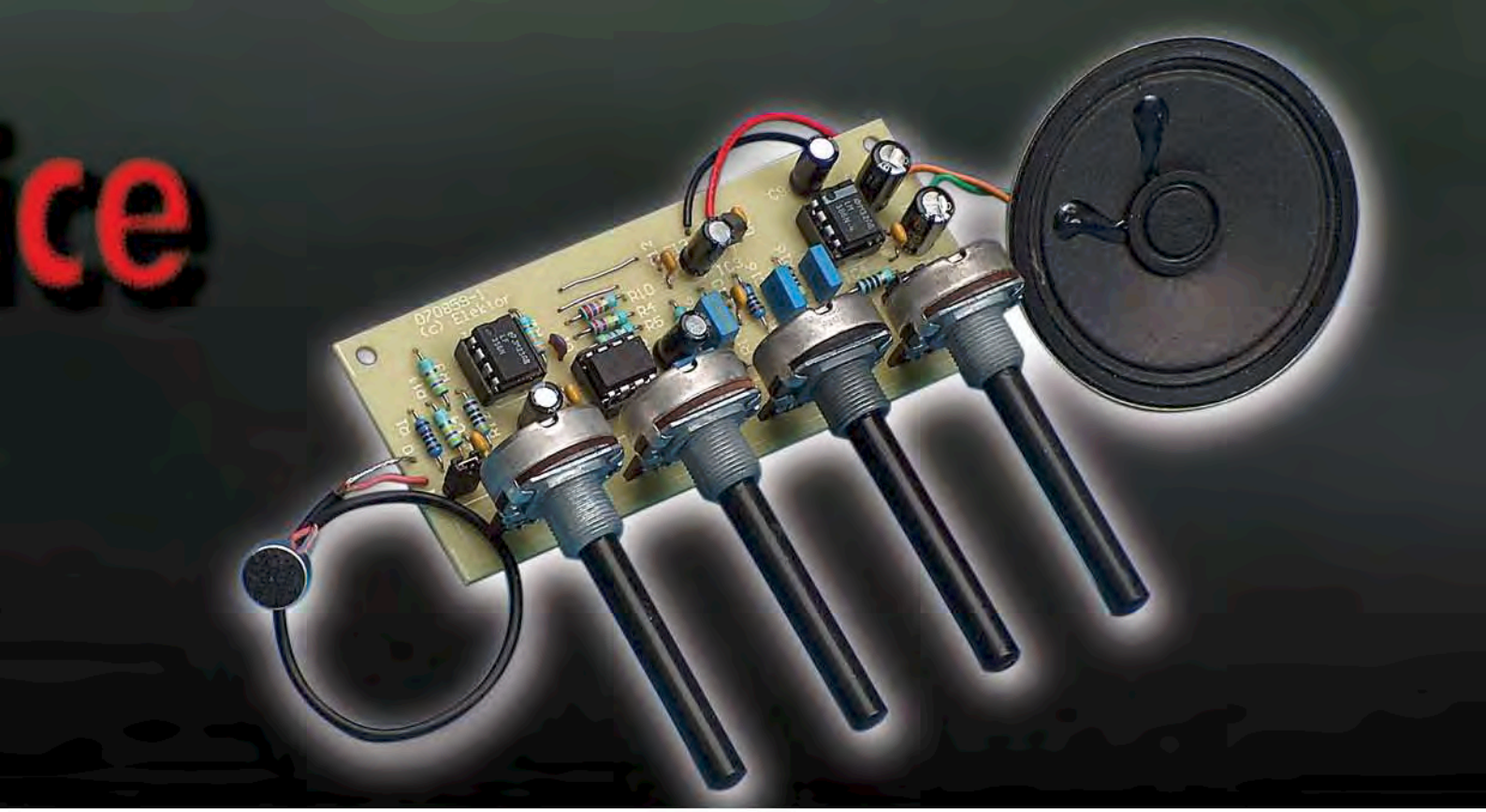



Figure 1. The voice changer consists of a pre-amp, a microcontroller and a power amp.



which is doubled to 16 MHz by an internal PLL. This method ensures that two inputs remain free on the 8-pin controller for connecting the two potentiometers. These are used to set the required 'Cylon' effect.

If two other ADC channels were used for reading the two potentiometers it would mean that the audio input would be interrupted by the internal MUX whilst reading those inputs, which would be detrimental to the audio quality. It is therefore preferable to read the potentiometers by timing the charging of an external capacitor.

The 'Cylon' software reads in the audio signal with a sampling frequency of about 10 kHz. According to the Nyquist Sampling Theorem the sampling frequency has to be at least twice that of the highest frequency in the input signal. From this it follows that the maximum input frequency is 5 kHz. As the 'Cylon' circuit is meant for speech signals, this is more than sufficient.

The Nyquist Theorem isn't that difficult to understand if we first consider what sampling does in the frequency domain. During the sampling period the input signal can be thought of as multiplied by '1' and during the remaining period by '0'. This is in principle the same as 100% AM modulation with a square wave, which results in an AM spectrum as shown in **Figures 2a and 2b**. From these you can clearly see that overlapping occurs when the sampling frequency is less than twice the highest input frequency, which leads to distortion. This goes to show how important a good input filter is before the ADC. This so-called anti-aliasing filter blocks components that are higher in frequency than half the sampling frequency. Since the 'Cylon' circuit usually operates with speech signals, which are mostly below 3 kHz, there is no need for an elaborate filter when sampling at 10 kHz.

After reading each sample the controller software carries out certain processes to obtain the 'Cylon' effect. The result is then fed to the PWM output of the controller. This generates a square wave with a variable duty cycle at output PB4 of the ATtiny45. After integrating this through an RC filter (R14 to R16, C15, C16, C18) the audio signal reappears.

This signal is then fed to audio amplifier IC4, which makes the sound audible via a loudspeaker. Setting the jumper on header K2 to the other position bypasses the power stage, so that the signal can be fed to the line input of, for example, a computer.

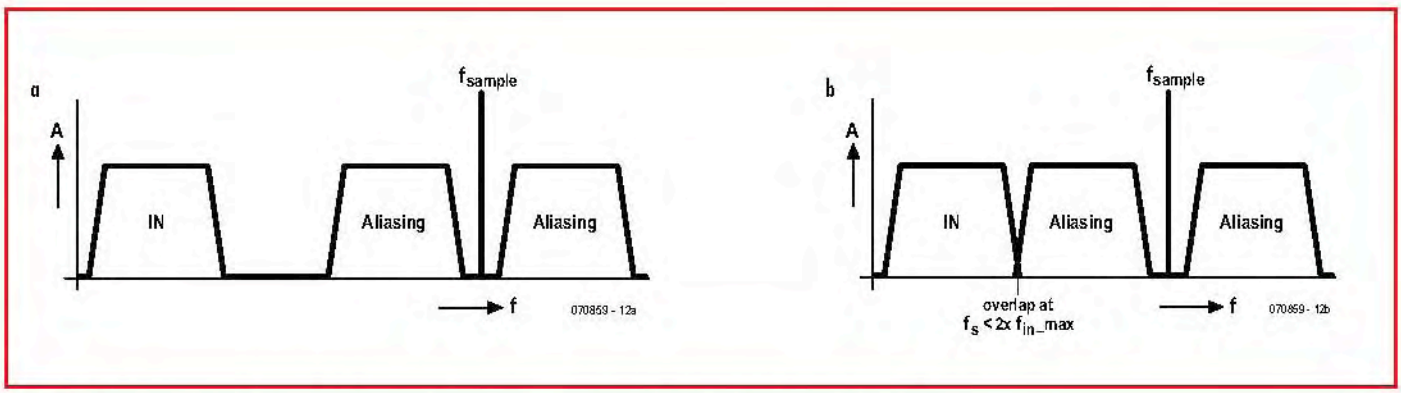



Figure 2. This shows that overlapping occurs when the sampling frequency is less than twice the highest component of the input signal.

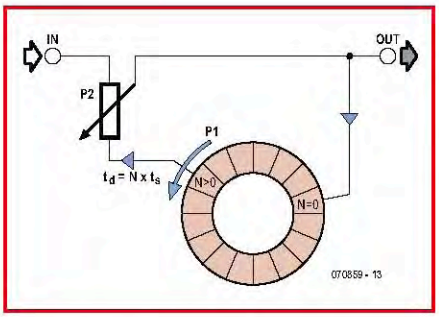



Figure 3. Graphical representation of the FIFO register, configured here as a ring-buffer.

## Software

The software has to carry out a number of tasks. The main one is the reading, processing and outputting of the audio signal, but there are also two potentiometers that have to be read. As the ADC is fully occupied with reading the audio signal, an internal counter and comparator are used to read in the potentiometer values. First of all, C1 is discharged. Then an internal counter is started and C1 is charged via the potentiometer to a certain voltage, when the counter is stopped. The resulting counter value is then dependant on the resistance of the potentiometer.

To put it simply, the 'Cylon' effect is obtained by mixing a delayed version of the input signal with the direct input signal, and delaying the resulting signal again, and so on.

The same principle can be used to obtain an echo effect, but then the delay of the input signal needs to be at least 100 ms, whereas in the 'Cylon' version the delay is much smaller. The internal FIFO buffer used for obtaining the delay is 200 bytes long, so with a sampling frequency of about 10 kHz the maximum delay can only be 200/10 kHz = 20 ms.

In **Figure 3** you can see a graphical representation of this method. The wheel 'turns' anti-clockwise with one revolution every 20 ms and represents the FIFO buffer, configured as a ring-buffer. The input signal is mixed with the delayed output signal (shown here using a potentiometer), creating a new output signal that is fed back again to the start of the ring-buffer.

The same effect can also be achieved mechanically with the use of magnetic tape and separate record and play heads, with the latter having a variable position to set the delay time.

The software is in effect a simulation of this method, where the memory is the equivalent of the magnetic tape, a store instruction is the record head, and a load instruction is the play head.

The amount of delay is set using P1. The position of P1 is used by the software to set the position of the tap in the ring-buffer. P2 is used to set the strength of the delayed feedback signal. With a larger feedback signal the decay becomes less resulting in a stronger effect.

P1 is used to set the tap to positions between 1 and 200 in the ring-buffer. The delay is therefore variable from 100 µs to 20 ms. Since the delayed signal is fed back in a loop the damped resonances will become noticeable when the damping is small enough. This is the so-called 'Cylon' effect. The frequency of these resonances is 1/delay-time and can therefore be set between 50 Hz and 10 kHz. A setting near 250 Hz sounds about the same as the original 'Cylon' robots.

## Construction

**Figure 4** shows the board layout for the voice changer. Populating the board should be simplicity itself, with all components clearly laid out. The controller can be obtained ready-programmed from Elektor Shop as item **070859-41**. If you prefer, you could program it yourself (software 070859-11, but make sure you use the correct settings for programming the chip, as shown in the inset).

If you use an electret microphone you'll need to place a jumper across K1. The jumper on K2 selects the type of output signal (pins 2-3 uses the output amplifier, pins 1-2 for a line output signal).

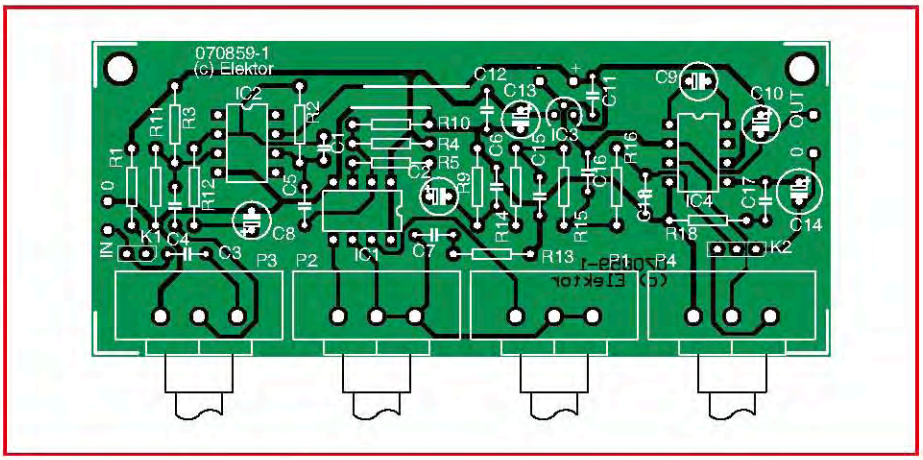



Figure 4. A small PCB has been designed for the voice changer, which makes the construction very easy.

## COMPONENTS LIST

### Resistors
R18 = 10Ω
R9 = 33Ω
R14 = 220Ω
R5 = 470Ω
R13,R15 = 1kΩ
R12,R16 = 4kΩ7
R1 = 10kΩ
R10 = 27kΩ
R4 = 220kΩ
R2 = 330kΩ
R3, R11 = 470kΩ
P1,P2 = 500kΩ
P3,P4 = 50kΩ

### Capacitors
C1 = 56pF
C2 = 2µF2 25V
C18 = 4nF7
C17 = 47nF
C6 = 470nF
C3,C4,C5,C11,C12,C15 = 100nF
C1 = 1µF 25V
C9 = 10µF 25V
C10,C14 = 100µF 25V
C13 = 22µF 25V
C7 = 2nF2
C16 = 22nF

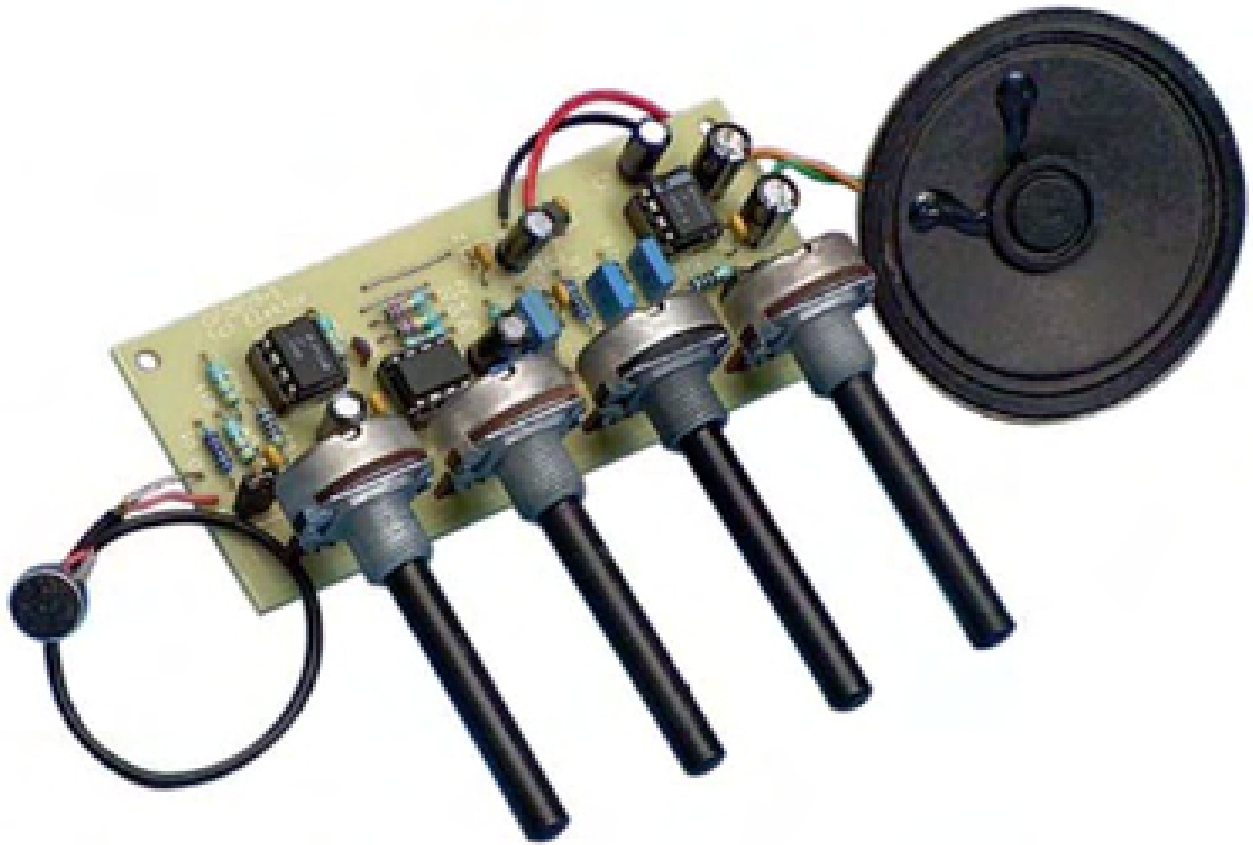
### Semiconductors
IC1 = ATtiny45, programmed, Elektor Shop # 070859-41
IC2 = LF356
IC3 = 78L05
IC4 = LM386-N3

### Miscellaneous
K1 = 2-way SIL pinheader
K2 = 3-way SIL pinheader
MIC1 = electret microphone
PCB, ref. 070859-1, see www.elektor.com
Project software, 070859-11.zip, free download from www.elektor.com
PCB layout, free download from www.elektor.com

The circuit doesn't require much current, which depends mainly on the output amplifier (between 25 mA and 150 mA). You could therefore us a 9 V battery as long as you don't use the circuit for long periods. When you don't use the output amplifier stage the current consumption drops to about 25 mA.

When you first switch it on it is best to set the effect to minimum. When you hear an undistorted audio signal you know that the circuit functions properly, so you can then increase the amount of effect. Too much effect eventually results in no audio at all (as you can work out from Figure 3). This can be set precisely using P1, giving a good effect whilst keeping the speech clearly understandable.

(070669-I)



## Settings for programming the controller

**Fuses:**
- Brown-out detection disabled: BODLEVEL=111
- PLL clock: CKSEL=0001, SUT=11

## P1 and P2 settings

- P1: Smaller value → higher frequency.
- P2: Larger value → bigger effect.