Didactic Tool: RC Servo Controller for Educational Robotics

Klemen Kozjek[1], **Aleš Jaklič**[1]

[1]*Faculty of Computer and Information Science, University of Ljubljana, Večna pot 113, 1000 Ljubljana*
E-mail: ales.jaklic@fri.uni-lj.si

Abstract. *We designed hardware and software for simple multichannel RC Servo Controller for smartphones and tablets running Android OS. The design is based on audio subsystem of a smartphone, pulse position modulation, and hardware decoding of the modulated audio signal. It also leverages the touch screen as GUI. The cost of the decoding hardware is in the range of few Euros and can be simply made using basic electronics tools.*

1 Introduction

Educational robotics is a popular way to provide students at various levels of knowledge with hands on experience and understanding of science, technology, engineering, and math disciplines (STEM). RC servos are important actuators in this endeavour and are usually driven programmatically by using microcontrollers to ensure precise timing requirements for the servo PWM signals.

In this paper we extend the idea to drive two RC servos by audio output of a smartphone [1], to drive up to 9 servos by using a simple electronic circuit. We avoid the use of microcontrollers in our design to alleviate steep learning curve in microcontroller programming. For completeness we first review the PWM an PPM.

2 Pulse Width Modulation (PWM) for Servos

RC servos are controlled by a series of repeating pulses of variable width [3,4]. Pulse width typically ranges from 1.0 ms to 2.0 ms and this corresponds to servo motor rotation of -45 deg to +45 deg, with 1.5 ms representing central position. The pulse width range might be extended to 0.7 ms and 2.3 ms for greater angular control from -90 deg to + 90 deg. Figure 1 shows a train of PWM pulses for servo control. The period of pulses is approximately 20 ms.

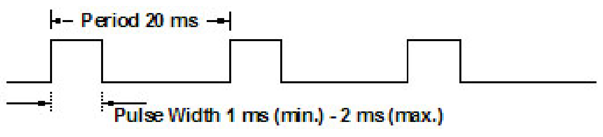


Figure 1. PWM signal for servo control

3 Pulse Position Modulation (PPM) for Radio Control

Pulse position modulation enables us to encode PWM signals of multiple servos into a single signal that can be transmitted over radio link or in our case to be transmitted over a single audio channel. Figure 2 illustrates how 5 channels PWM pulses are encoded sequentially, separated by negative pulses. Each frame of pulses is terminated by a pause or sync pulse, so that the receiver can prepare, usually resets itself, for another frame.

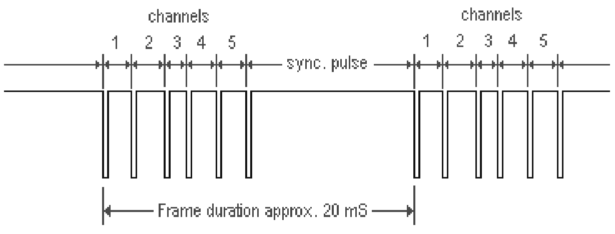


Figure 2. Pulse Position Modulation signal frame

4 Hardware Decoder for Pulse Position Modulation

We reused a classical design for decoding PPM signal that can decode up to 8 channels based on 5-stage Johnson counter 4017 circuit [2,5]. Figure 3 shows the schematics of the hardware interface.

The input signal from the audio output of a smart device is feed into connector K1. Then it is amplified by a non-inverting operational amplifier formed by resistors R1 and R2 and IC1A. An amplified signal from pin 1 of IC1 is then fed into a comparator formed by resistors R3 an R4 and IC1B, this effectively inverts the input signal. The inverted signal is used as a CLK signal for the IC2 and also drives the reset circuit composed of resistor R5, capacitor C1 and diode D1. The IC2 outputs the demodulated servo PWM pulses on outputs Q1 to Q9 connected to connectors K2 to K10. The circuit is powered by 4 AA batteries through K11.

5 Signals in the Circuit

The hardware design presents a variety of analog and digital signals suitable for observation on oscilloscope and can be used as didactic tool for oscilloscope's use.

For example a process of charging and discharging a capacitor in the reset circuit nicely illustrates how the voltage changes over time.

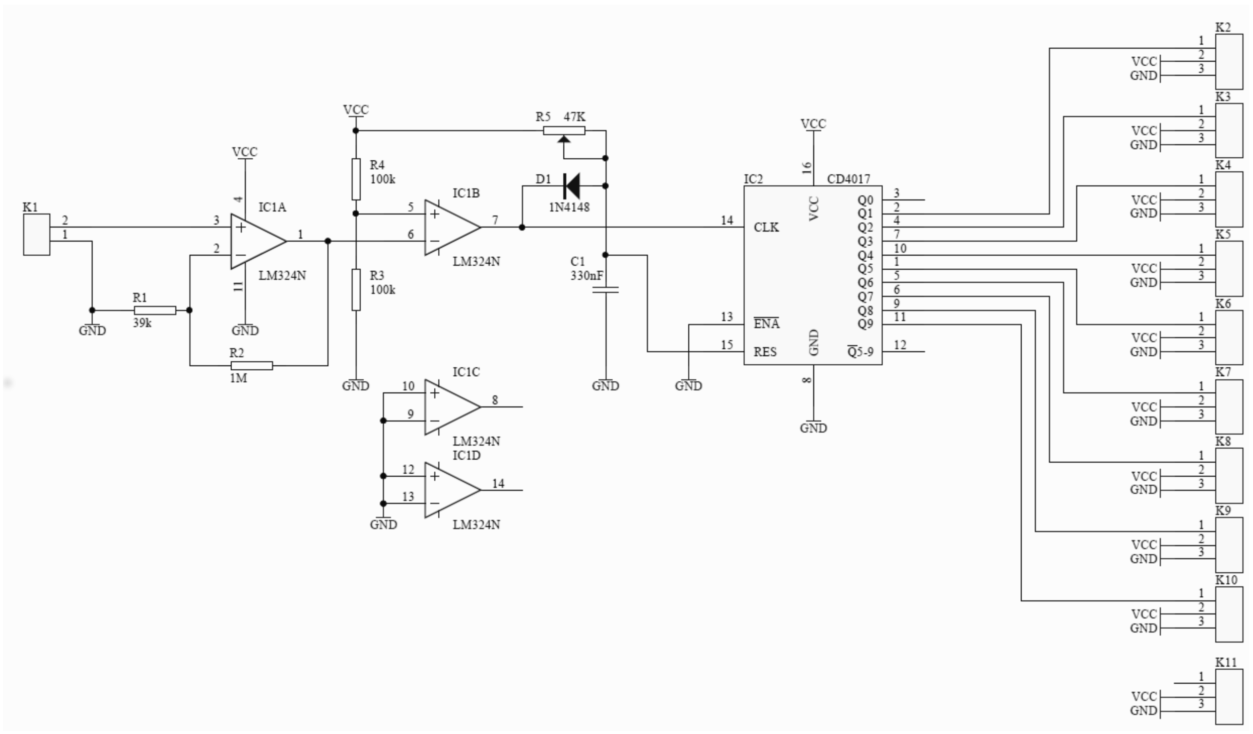


Figure 3. Schematics of the hardware interface

Figure 4 shows the oscilloscope image of input audio signal from pin 3 of IC1A (Channel 1) and the output amplified audio signal from pin 1 of IC1A (Channel 2).

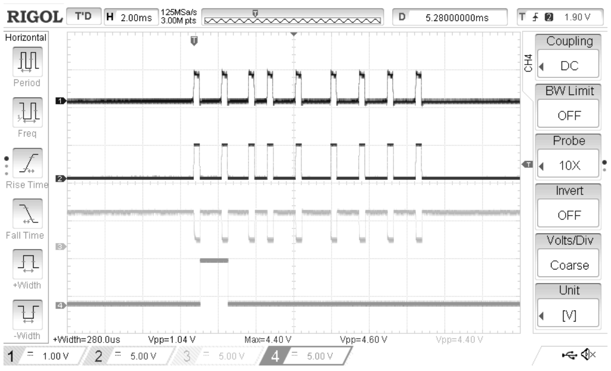


Figure 4: Signals in the circuit

The amplification A of the non inverting operational amplifier is defined as

$$A = 1 + \frac{R_2}{R_1} = 26 \qquad (1)$$

The voltage divider composed of resistors R4 an R3 sets the input voltage of pin 5 of IC1B to half the VCC = 6V that is to 3V. Thus the comparator outputs voltage close to VCC if the voltage on pin 6 of IC1B is lower than 3V, and 0V if the voltage is above 3V. This effectively inverts the amplified audio input signal (See Channel 3 of Figure 4)

To properly demodulate each frame, the counter IC2 4017 needs to be reset before each incoming frame. This is achieved during the pause between two consecutive frames. The voltage on RES pin 15 of IC2 is equal to the voltage on C1 that gets charged through R5 with maximal time constant

$$\tau = R5 * C1 = 15{,}5 \text{ ms} \qquad (2)$$

As long the pin 7 output of IC1B goes low for a brief period of time, the capacitor C1 gets discharged through diode D1 to approximately 0.6 V, and the RES signal will not go high. When the input frame is finished the pause between frames, that is the absence of negative pulses will cause the capacitor to charge and the voltage signal RES to reach sufficient level to reset the counter IC2, thus preparing the state of the counter for the next frame.

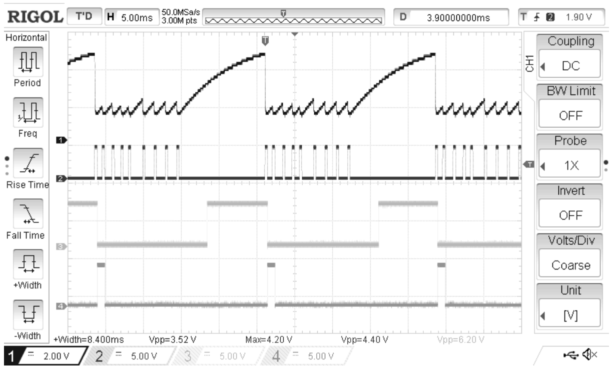


Figure 5: Signals in the circuit: Channel 1 – RES of IC2, Channel 2 – pin 1 of IC1A, Channel 3 – output Q0 of IC2, Channel 4 – output Q1 of IC2

Since the threshold for RES signal can vary from IC to IC, we use the trimmer potentiometer R5 and an oscilloscope to adjusted precise timing for the counter 4017 reset. As the counter IC2 is reset, the output Q0 goes high. Figure 6 shows how the PPM signal on oscilloscope's Channel 1 gets decoded for the first three servo channels.

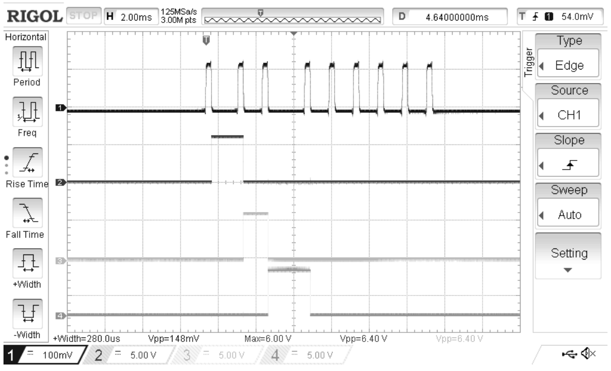


Figure 6: Signals in the circuit: Channel 1 – input audio signal, Channel 2 – output Q1 of IC2 – servo channel 1, Channel 3 – output Q2 of IC2 servo channel 2, Channel 4 – output Q3 of IC2 servo channel 3

6 Hardware Construction

For circuit prototype we designed a single sided PCB shown in Figure 7 and Figure 8 with standard through hole components and ICs mounted on sockets to allow easy construction in a classroom.

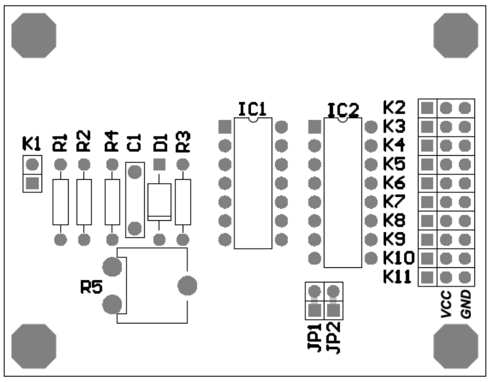


Figure 7: Component side of the PCB

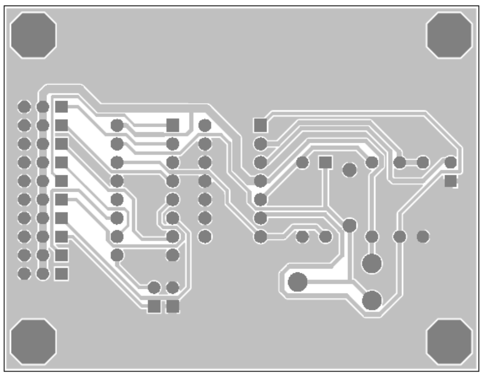


Figure 8: Bottom (copper) side of the PCB

Figure 9 displays the hardware in operation.

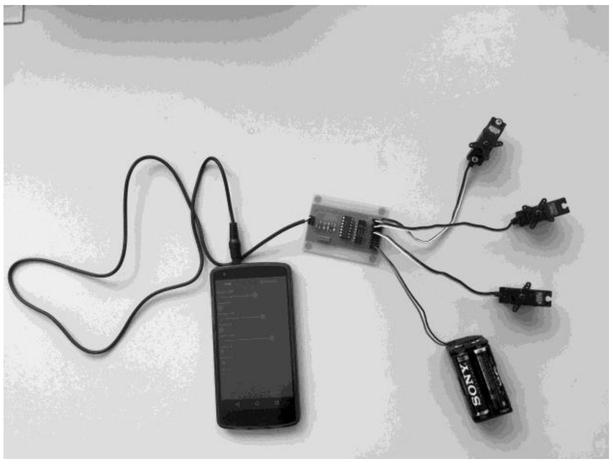
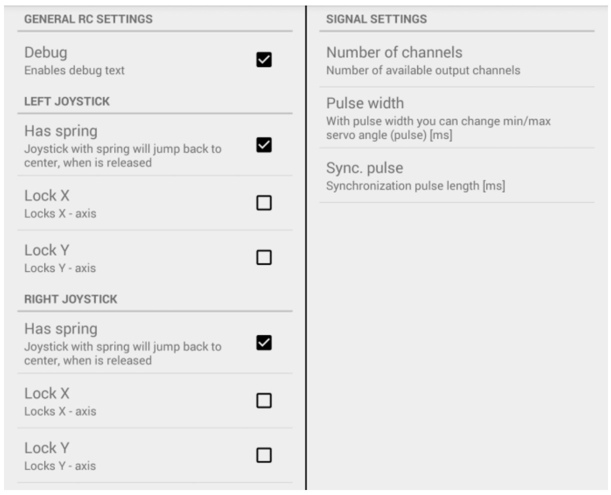


Figure 9: PPM RC Servo Controller hardware in operation

7 Android App for Smartphone/Tablet

To control the circuit which is shown in Figure 3 we developed a simple Android application which is able to control up to 9 servo motors simultaneously. To generate an audio output signal we use AudioTrack API that allows us to generate and play audio signal easily in real time. In our application we set AudioTrack to operate in stream mode, which allows us to feed its buffer constantly with generated data. Sampling frequency is set to 44100 Hz, for presentation of audio data we use a 16 bit format, because API ensures that the format is supported by devices and audio channel is configured as mono output [6]. The size of buffer is set to the size required for a single frame, which is calculated according to the configuration defined by the user or the application itself. The configuration that user can alter, includes number of active output channels, pulse width for each servo motor, synchronization pulse length and inversion of PPM signal. Some of the settings are shown on Figure 10.



GENERAL RC SETTINGS

Debug ☑
Enables debug text

LEFT JOYSTICK

Has spring ☑
Joystick with spring will jump back to center, when is released

Lock X ☐
Locks X - axis

Lock Y ☐
Locks Y - axis

RIGHT JOYSTICK

Has spring ☑
Joystick with spring will jump back to center, when is released

Lock X ☐
Locks X - axis

Lock Y ☐
Locks Y - axis

SIGNAL SETTINGS

Number of channels
Number of available output channels

Pulse width
With pulse width you can change min/max servo angle (pulse) [ms]

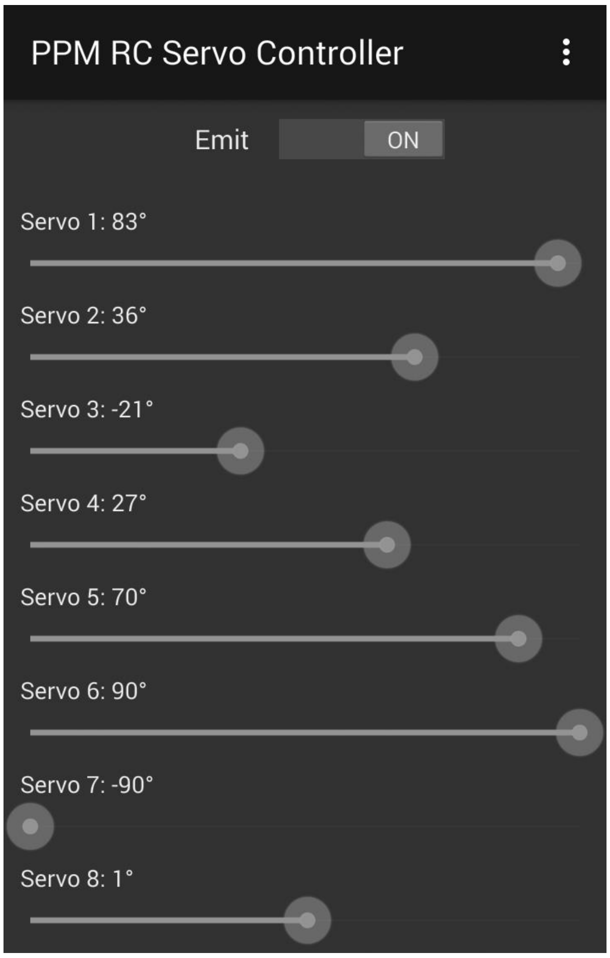
Sync. pulse
Synchronization pulse length [ms]

Figure 10: Setting screens: Left - RC Controller settings, Right - N-Channels settings

In our application we have currently two implemented modules, which can control circuit: N-Channels and RC Controller.

7.1 N-Channels

N-Channels operates as a sort of a test bed and allows user to configure and test the settings, such as: number of channels, pulse width and synchronization pulse length. The setting that changes pulse width enables user to control servo motor which does not use standard pulse width from 1.0 ms to 2.0 ms, but they need longer pulse length (i.e. from 0.7 ms to 2.3 ms) to cover full workspace. Figure 11 shows user interface, simple to manage by using the multi-touch sliders which change angles of servo motors. For the offset angle we use a scale in the range of -90 degrees to 90 degrees. Default angle is set to 0 degrees as is same as pulse with 1.5 ms width.


PPM RC Servo Controller

Emit ON

Servo 1: 83°

Servo 2: 36°

Servo 3: -21°

Servo 4: 27°

Servo 5: 70°

Servo 6: 90°

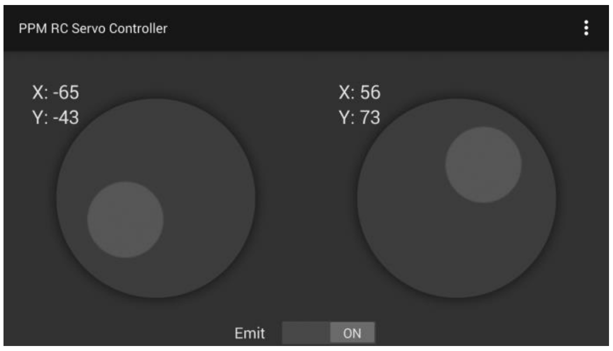
Servo 7: -90°

Servo 8: 1°

Figure 11: User interface with sliders for N-Channels module configured to work with eight servo motors

7.2 RC Controller

RC Controller module provides interface similar to RC controller. It contains two joysticks as it is shown in Figure 12, which can be configured. Each joystick is connected to two output channels and each axis manipulates one servo, so together they use four servo motors. The module also provides an option that can reduce the number of active output channels by locking axes as it is shown in Figure 10.


PPM RC Servo Controller

X: -65
Y: -43

X: 56
Y: 73

Emit ON

Figure 12: User interface with two joysticks for RC Controller module

To calibrate joysticks with servo motors we can help us with N-Channels module and its settings for pulse width and synchronization pulse length, because this two settings are shared between modules.

8 Conclusions and Future Work

We designed a simple hardware interface to control up to 9 RC servos by audio output of smartphones and tablets. Only standard through hole electronics components and single sided PCB were used to allow for easy construction of the interface, which can serve as an excellent beginners electronics project. We avoided the use of microcontrollers, which might lead to a more flexible design but at a cost of steep learning curve for learning how to program a microcontroller. The controller allows for cheap implementation of robotics arms used in didactic [7] or artistic projects [8].

The interface can be expanded to drive up to 18 servos by simple replication of the circuit for the second audio channel of a stereo audio output. We plan to port the PPM RC Servo Controller Android application to iOS.

References

[1] Smartphone servo, http://makezine.com/projects/make-34/smartphone-servo/
[2] Proportional Radio control, http://sm0vpo.altervista.org/use/rc-prop.htm
[3] Servo Control, http://en.wikipedia.org/wiki/Servo_control
[4] Servo control interface in detail, https://www.pololu.com/blog/17/servo-control-interface-in-detail
[5] Model Radio Control History, http://myweb.tiscali.co.uk/norcimradiocontrol/Radio6.htm
[6] Android AudioTrack documentation, http://developer.android.com/reference/android/media/AudioTrack.html
[7] Robotic arm with 7 servos http://www.thingiverse.com/thing:2433
[8] P. Tresset and F. F. Leymarie. Portrait drawing by Paul the robot. Computers Graphics, 37(5):348 - 363, 2013.