NetMedia

Basic Express
Application Note

Controlling Servos with BasicX

Introduction

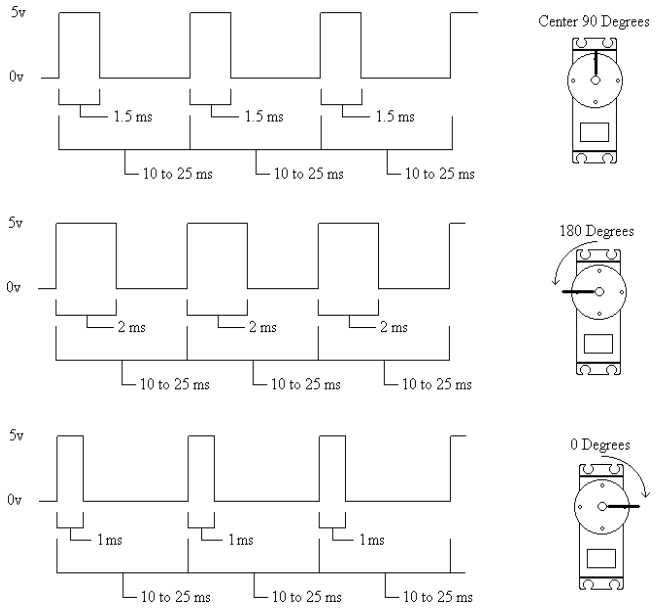
This application note describes both hardware and software methods for interfacing R/C type hobby servos to the BasicX chip.

R/C hobby servos are quickly becoming one of the most popular means for computer controlled motion applications. Hobby servos were originally designed and sold for the purpose of actuating the control surfaces of remotely controlled models (airplanes, cars, boats etc.).

Hardware interface

Servos require electrical power and a position signal to operate. Servos typically require 4.8 VDC to 6.5 VDC power. Current requirements are highly variable. You may need to refer to the servo manufacturer's specifications for more precise voltage and current requirements.

The position signal usually consists of a 1 ms to 2 ms high-going pulse repeated at a 50 Hz refresh rate. An example of this signal and the servo's resulting movements can be seen below:



The pulse width range can vary depending on the make and model of servo. Some servos have a larger movement range and may require pulses as small as 0.5 ms or as large as 2.2 ms to achieve full range of movement. But be careful not to exceed a safe range -- depending on the servo, if a pulse is too long or too short, you may drive the servo against an internal stop, stall the motor and damage gears.

The optimum pulse rate can vary somewhat. If your refresh rate is either too slow or too fast, the servo may produce a loud buzzing sound. If this happens, try turning the output horn -- if the noise gets louder, the refresh rate is too slow. Otherwise it's too fast.

Servo connection

Figure 1 shows a suggested method of interfacing a servo to the BasicX chip. The red and black servo leads are usually for power (positive and ground, respectively). The white (or sometimes yellow) wire is the signal wire.

Notice that the servo is powered from a separate battery in this example. Servos typically draw far more power that the BasicX development board can supply. It is also important to note that both the ground from the battery pack and the BasicX are tied together -- otherwise the servo may not operate properly.

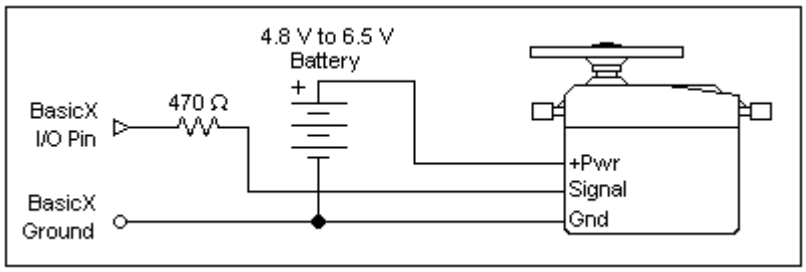


Figure 1

Example Code

By using procedure PulseOut, the BasicX can generate the required servo position signals. Depending on how the program is written, 10 or more servos can be run from one BasicX chip. Example code for a single servo:

```
Sub Main()

' This program will move the servo to its middle position.

    Const ServoPin As Byte = 16

    Do
        ' Generate a high-going 1.5 ms pulse.
        Call PulseOut(ServoPin, 0.0015, 1)

        ' This is to produce a pulse rate of about 50 Hz.
        Call Delay(0.02)
    Loop

End Sub
```

Warning -- if you use PulseOut to generate servo signals, the real time clock can lose time unless pulse widths are smaller than the real time clock period (about 1.94 ms).

Example program

An example program called Servo.bas is provided as a separate file. The program exercises a servo by moving it back and forth through its range of motion.

2.00.A