Copyright permission given to Dr. Brian Fraser

MG995R Servo Motor Guide

By Felix Lionardo and Ryan Ly

Parts Requirement:

- Wires
- 470-ohm resistor (2x)
- MG995R Motor Servo
- Beagle Bone Green

Table of contents:

Introduction

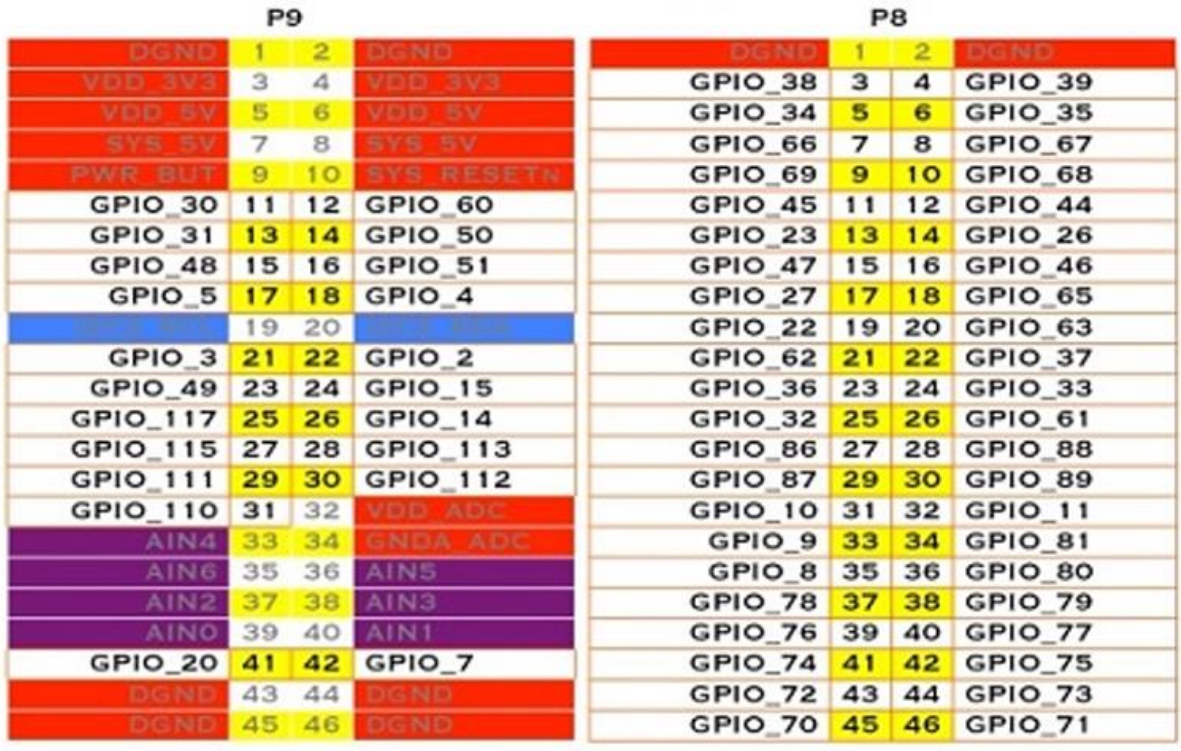
The previous servo guides from previous students are a little bit outdated so this will be an updated version of the servo guide.

The servo motor is controlled through a PWM signal, where the width of the pulse (usually ranging from 0.5 ms to 2.5 ms in a typical 20 ms period) dictates the angle to which the servo will rotate.

For more complete detail on how PWM and servo works you can go through this guide

https://opencoursehub.cs.sfu.ca/bfraser/grav-cms/cmpt433/links/files/2022-student-howtos/ServoPWMGuide_SG90_9gMicroServo.pdf

Wiring the Servo



P9

DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3
VDD_5V	5	6	VDD_5V
SYS_5V	7	8	SYS_5V
PWR_BUT	9	10	SYS_RESETN
GPIO_30	11	12	GPIO_60
GPIO_31	13	14	GPIO_50
GPIO_48	15	16	GPIO_51
GPIO_5	17	18	GPIO_4
I2C2_SCL	19	20	I2C2_SDA
GPIO_3	21	22	GPIO_2
GPIO_49	23	24	GPIO_15
GPIO_117	25	26	GPIO_14
GPIO_115	27	28	GPIO_113
GPIO_111	29	30	GPIO_112
GPIO_110	31	32	VDD_ADC
AIN4	33	34	GNDA_ADC
AIN6	35	36	AIN5
AIN2	37	38	AIN3
AIN0	39	40	AIN1
GPIO_20	41	42	GPIO_7
DGND	43	44	DGND
DGND	45	46	DGND

P8

DGND	1	2	DGND
GPIO_38	3	4	GPIO_39
GPIO_34	5	6	GPIO_35
GPIO_66	7	8	GPIO_67
GPIO_69	9	10	GPIO_68
GPIO_45	11	12	GPIO_44
GPIO_23	13	14	GPIO_26
GPIO_47	15	16	GPIO_46
GPIO_27	17	18	GPIO_65
GPIO_22	19	20	GPIO_63
GPIO_62	21	22	GPIO_37
GPIO_36	23	24	GPIO_33
GPIO_32	25	26	GPIO_61
GPIO_86	27	28	GPIO_88
GPIO_87	29	30	GPIO_89
GPIO_10	31	32	GPIO_11
GPIO_9	33	34	GPIO_81
GPIO_8	35	36	GPIO_80
GPIO_78	37	38	GPIO_79
GPIO_76	39	40	GPIO_77
GPIO_74	41	42	GPIO_75
GPIO_72	43	44	GPIO_73
GPIO_70	45	46	GPIO_71

3 Wires of servo:

- Red = Power
- Brown = Ground
- Yellow = PWM signal

The servo requires 4.8 – 6 volts in order for it to run. So, you want to connect the red wires into either p9-7 or p9-8.

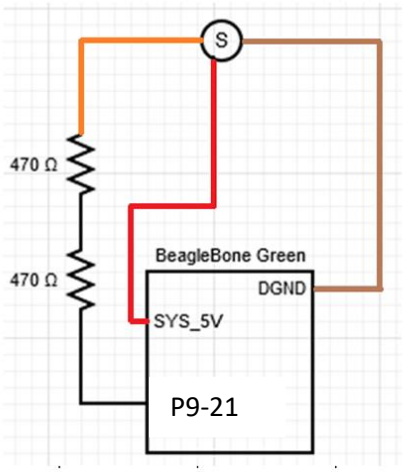
Connect the yellow wires into one of these pins.

Zen Cape Use	PWM Channel	BBB Pin	Linux Path[1]	Notes
Buzzer	PWM-0A	P9-22	/dev/bone/pwm/**0/a/**	These two share a PWM timer: Period must be the same; duty cycle is independently.
unused	PWM-0B	P9-21	/dev/bone/pwm/**0/b**	
Blue LED	PWM-1A	P9-14	/dev/bone/pwm/**1/a**	These two share a PWM timer.
Red LED	PWM-1B	P9-16	/dev/bone/pwm/**1/b/**	
Green LED	PWM-2A	P8-19	/dev/bone/pwm/**2/a**	These two share a PWM timer.
unused	PWM-2B	P8-13	/dev/bone/pwm/**2/b**	

For the sake of the guide, we are going to be connecting the yellow wires into p9-21. You will also need to have 2 470-ohm resistors in between the yellow wires of the servo and the p9-21 pin to lower the voltage that comes into the pin.

The brown wires can connect to any ground pins on the zen cape (DGND)

This is how the wiring will look like:



Controlling the Servo

Once it is connected properly, to control the servo we need to go into the bbg and cd to the file that controls the pwm.

First you need to configure the p9-21 to be pwm

BBG: sudo config-pin p9_21 pwm

Then go into the file that controls the pwm

BBG: cd /dev/bone/pwm/0/b

BBG: ls

capture duty_cycle enable period polarity power uevent

Then you want to enable it.

BBG: echo 1 > enable

Then you want to set the period to 20.000.000.

BBG: echo 20000000 > period

Then to move the servo you want to set the duty_cycle in the range of 500.000 to 2.500.000. setting the duty_cycle to 500.000 will move the servo to one end and 2.500.000 will move it to the other end. 1.500.000 will be the neutral position

BBG: echo 500000 > duty_cycle

BBG: echo 1500000 > duty_cycle

BBG: echo 2500000 > duty_cycle

You can refer to this link for more details about controlling PWM:

https://opencoursehub.cs.sfu.ca/bfraser/grav-cms/cmpt433/guides/files/PWMGuide.pdf

Troubleshooting

- If your servo does not move double check the wiring. Make sure that the power comes from SYS_5V and NOT VDD_5V
- If you are unable to set the period of the pwm. Make sure the period you are trying to set is not less than the current duty_cycle. It is because period cannot be less than the duty_cycle.