Check PWM Driver at /sys/class/pwm

https://forums.developer.nvidia.com/t/how-do-i-use-pwm-on-jetson-nano/72595/5

Step 1. Check if pwm driver is already installed in your OS kernel

$ls *sys/class/pwm*

```
harry-nano@harry-desktop-nano: /sys/class
harry-nano@harry-desktop-nano:/sys/class$ ls /sys/class/pwm
pwmchip0  pwmchip4
harry-nano@harry-desktop-nano:/sys/class$
```

So in response, we have:
pwmchip0 pwmchip4

Step 2. Check the number of channels for each pwm

First go to pwm folder, then from there to pwmchip0, then do

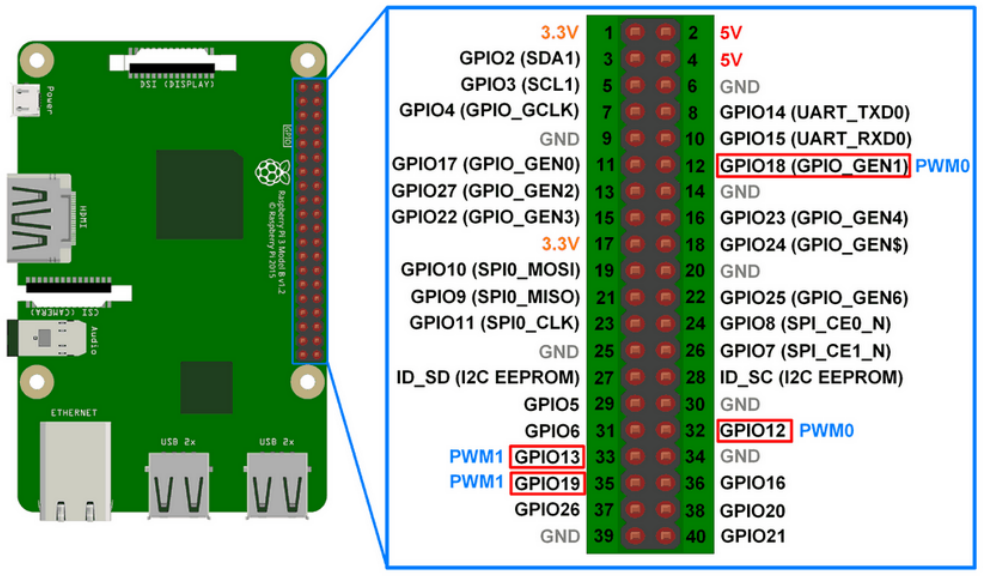
$cat npwm //to find number of channels

```
harry-nano@harry-desktop-nano: /sys/class/pwm/pwmchip0
pwmchip0  pwmchip4
harry-nano@harry-desktop-nano:/sys/class/pwm$ cd pwmchip0
harry-nano@harry-desktop-nano:/sys/class/pwm/pwmchip0$ cat npwm
4
```

If you do the same for pwmchip4, you will find 1 channel. So the total pwm channel is 5.

Step 3. Find the mapping to the connector pins, use the pi pin assignment as a reference (see below)

Pi Pin layout



Jetson Nano J41 Header Pinout for PWM

https://www.jetsonhacks.com/nvidia-jetson-nano-j41-header-pinout/

Note: I2C and UART pins are connected to hardware and should not be reassigned. By default, all other pins (except power) are assigned as GPIO. Pins labeled with other functions are recommended functions if using a different device tree.

Use pin 32 for PWM

Sysfs GPIO	Name	Pin	Pin	Name	Sysfs GPIO
	3.3 VDC *Power*	1	2	**5.0 VDC** *Power*	
	I2C_2_SDA *I2C Bus 1*	3	4	**5.0 VDC** *Power*	
	I2C_2_SCL *I2C Bus 1*	5	6	**GND**	
gpio216	**AUDIO_MCLK**	7	8	**UART_2_TX** */dev/ttyTHS1*	
	GND	9	10	**UART_2_RX** */dev/ttyTHS1*	
gpio50	**UART_2_RTS**	11	12	**I2S_4_SCLK**	gpio79
gpio14	**SPI_2_SCK**	13	14	**GND**	
gpio194	**LCD_TE**	15	16	**SPI_2_CS1**	gpio232
	3.3 VDC *Power*	17	18	**SPI_2_CS0**	gpio15
gpio16	**SPI_1_MOSI**	19	20	**GND**	
gpio17	**SPI_1_MISO**	21	22	**SPI_2_MISO**	gpio13
gpio18	**SPI_1_SCK**	23	24	**SPI_1_CS0**	gpio19
	GND	25	26	**SPI_1_CS1**	gpio20
	I2C_1_SDA *I2C Bus 0*	27	28	**I2C_1_SCL** *I2C Bus 0*	
gpio149	**CAM_AF_EN**	29	30	**GND**	
gpio200	**GPIO_PZ0**	31	32	**LCD_BL_PWM**	gpio168
gpio38	**GPIO_PE6**	33	34	**GND**	
gpio76	**I2S_4_LRCK**	35	36	**UART_2_CTS**	gpio51
gpio12	**SPI_2_MOSI**	37	38	**I2S_4_SDIN**	gpio77
	GND	39	40	**I2S_4_SDOUT**	gpio78

pin 12 for gpio79

pin 12 for gpio78

Harry Li, Ph.D.

Configuration of Pins with jetson-io.py

Step 1. Fix bugs from the distribution

$sudo find /opt/nvidia/jetson-io/ -mindepth 1 -maxdepth 1 -type d -exec touch {}/__init__.py \;

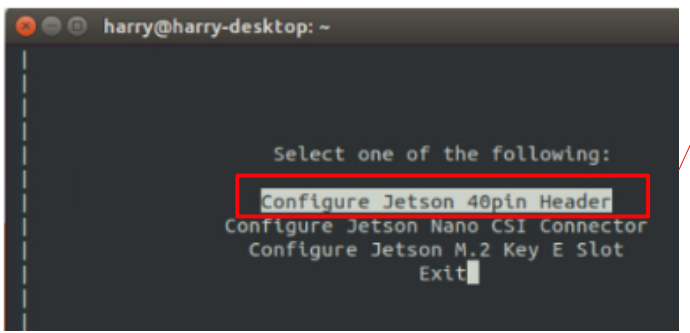
$sudo /opt/nvidia/jetson-io/config-by-pin.py -p 5

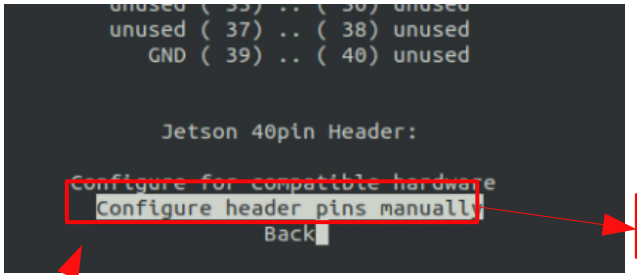
```
harry@harry-desktop:~$ sudo /opt/nvidia/jetson-io/config-by-pin.py -p 5
Traceback (most recent call last):
  File "/opt/nvidia/jetson-io/config-by-pin.py", line 84, in <module>
    main()
  File "/opt/nvidia/jetson-io/config-by-pin.py", line 39, in main
    jetson = board.Board()
  File "/opt/nvidia/jetson-io/Jetson/board.py", line 229, in __init__
    self.dtb = _board_get_dtb(self.compat, self.model, dtbdir)
  File "/opt/nvidia/jetson-io/Jetson/board.py", line 114, in _board_get_dtb
    raise RuntimeError("No DTB found for %s!" % model)
RuntimeError: No DTB found for NVIDIA Jetson Nano Developer Kit!
```

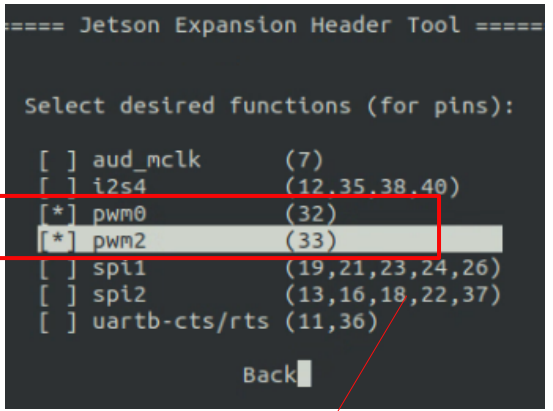
$sudo mkdir -p /boot/dtb
$ ls /boot/*.dtb | xargs -I{} sudo ln -s {} /boot/dtb/

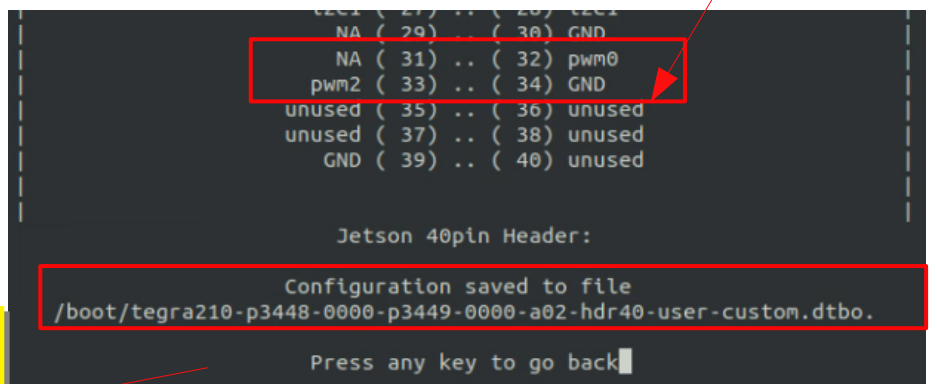
Step 2. Run jetson-io.py to configure pins

$sudo /opt/nvidia/jetson-io/jetson-io.py









Be sure to choose save and reboot to reboot the system

Command Line PWM Testing

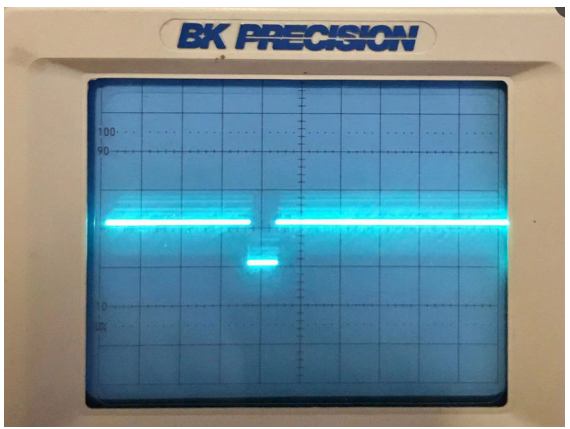
https://forums.developer.nvidia.com/t/how-do-i-use-pwm-on-jetson-nano/72595/5

Once you have properly configure the jetson-io as described in the previous slide, now you can use command line input (CLI) to test PWM port. You can use pin 32 as pwm0.

```
cd /sys/class/pwm/pwmchip0
echo 0 > export
sleep 1
cd pwm0
echo 5000000 > period
echo 2500000 > duty_cycle
echo 1 > enable
```

Define as in Hz

Output high defined as in Hz



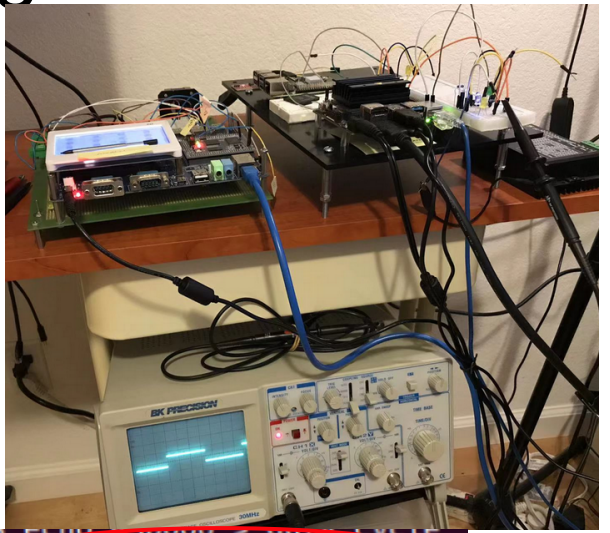

```
harry@harry-desktop: /sys/class/pwm/pwmchip0/pwm0
y@harry-desktop:~$ cd /sys/class/pwm/pwmchip0
y@harry-desktop:/sys/class/pwm/pwmchip0$ echo 0 > export
y@harry-desktop:/sys/class/pwm/pwmchip0$ sleep 1
y@harry-desktop:/sys/class/pwm/pwmchip0$ cd pwm0
y@harry-desktop:/sys/class/pwm/pwmchip0/pwm0$ echo 5000000 > period
y@harry-desktop:/sys/class/pwm/pwmchip0/pwm0$ echo 2500000 > duty_cycle
y@harry-desktop:/sys/class/pwm/pwmchip0/pwm0$ echo 1 > enable
y@harry-desktop:/sys/class/pwm/pwmchip0/pwm0$ echo 1 > enable
y@harry-desktop:/sys/class/pwm/pwmchip0/pwm0$ echo 1000000 > duty_cycle
y@harry-desktop:/sys/class/pwm/pwmchip0/pwm0$ echo 500000 > duty_cycle
y@harry-desktop:/sys/class/pwm/pwmchip0/pwm0$ echo 250000 > duty_cycle
y@harry-desktop:/sys/class/pwm/pwmchip0/pwm0$ echo 450000 > duty_cycle
y@harry-desktop:/sys/class/pwm/pwmchip0/pwm0$ echo 4500000 > duty_cycle
```

Harry Li, Ph.D.

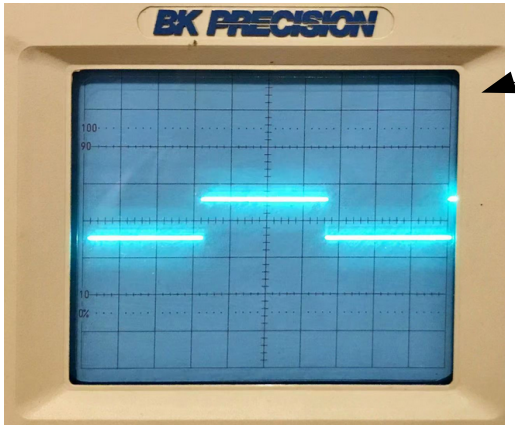
Command Line PWM Testing Result

https://forums.developer.nvidia.com/t/how-do-i-use-pwm-on-jetson-nano/72595/5

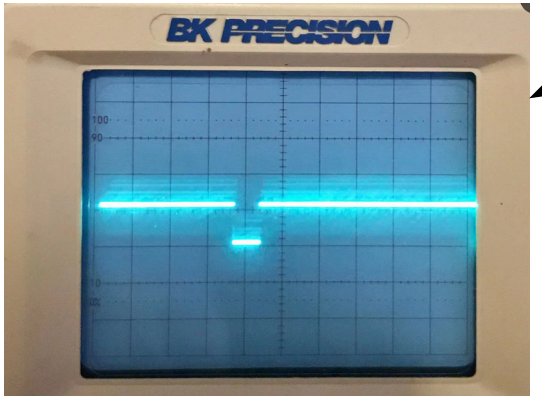
The testing environment setup, with the prototype board, and a oscilloscope



```
harry@harry-desktop:/sys/class/pwm/pwmchip0/pwm0$ echo 1000000 > duty_cycle
harry@harry-desktop:/sys/class/pwm/pwmchip0/pwm0$ echo 500000 > duty_cycle
harry@harry-desktop:/sys/class/pwm/pwmchip0/pwm0$ echo 250000 > duty_cycle
```

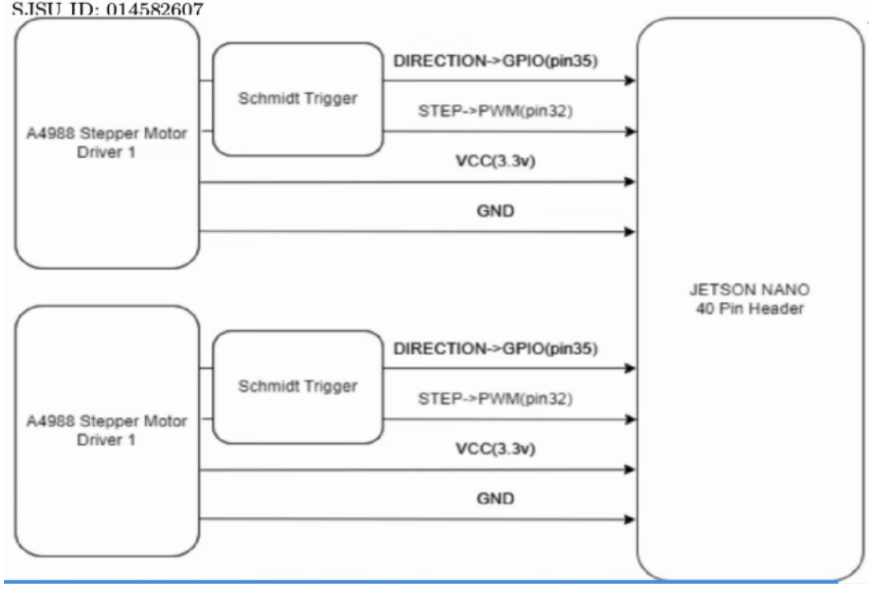


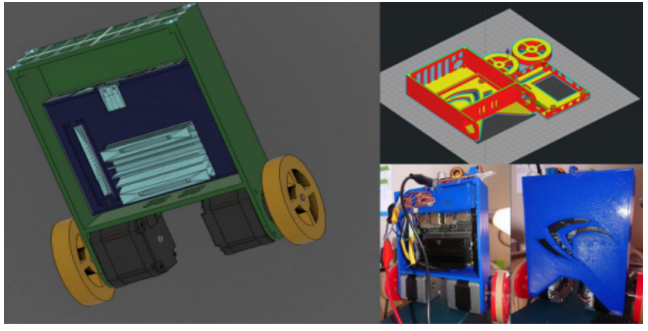

```
harry@harry-desktop:/sys/class/pwm/pwmchip0/pwm0$ echo 4500000 > duty_cycle
```

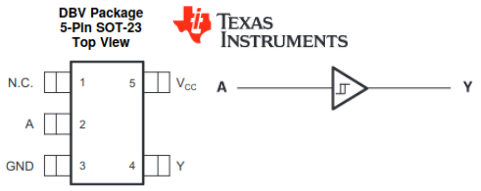


NANO PWM with Schmidt Trigger and BNO055

Stewart Lach
Harry Li CMPE-244
SJSU ID: 014582607







SN74LVC1G17-Q1 Single Schmitt-Trigger Buffer

3 Description

This single Schmitt-trigger buffer is designed for 1.65-V to 5.5-V V_{CC} operation.

The SN74LVC1G17-Q1 device contains one buffer and performs the Boolean function Y = A.

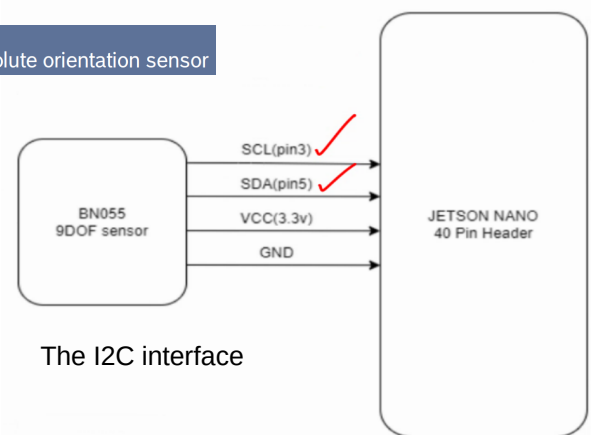
The CMOS device has high output drive while maintaining low static power dissipation over a broad V_{CC} operating range.

It can produce 24 mA of drive current at 3.3 V making it Ideal for driving multiple outputs and good for high speed applications up to 100 MHz. The inputs are 5.5 V tolerant allowing it to translate down to VCC.

BNO055
Intelligent 9-axis absolute orientation sensor

Quaternion
Linear Acceleration
Rotation
Gravity
Robust Heading

Quaternion, Euler angles, Rotation vector, Linear acceleration, Gravity, Heading, 3 sensors in one device an advanced triaxial 16bit gyroscope, a versatile,leading edge triaxial 14bit accelerometer and a full performance geomagnetic sensor



The I2C interface

Command Line PWM vs. Command Line GPIO

Example: PWM

```
cd /sys/class/pwm/pwmchip0
echo 0 > export
sleep 1
cd pwm0
echo 5000000 > period
echo 2500000 > duty_cycle
echo 1 > enable
```

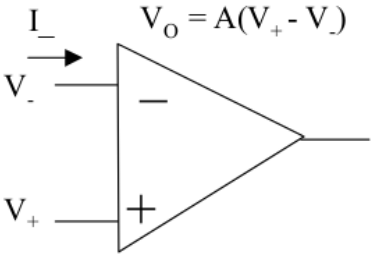
Example: GPIO

```
$echo 79 > /sys/class/gpio/export
$ echo out > /sys/class/gpio/gpio79/direction
$echo 1 >  /sys/class/gpio/gpio79/value
$echo 0 >  /sys/class/gpio/gpio79/value
$echo 79 /sys/class/gpio/unexport
$cat /sys /kernel /debug/gpio
```

```
$echo 168 > /sys/class/pwm/pwmchip0/export
$ echo out > /sys/class/gpio/gpio79/direction
$echo 1 >  /sys/class/gpio/gpio79/value
$echo 0 >  /sys/class/gpio/gpio79/value
$echo 79 /sys/class/gpio/unexport
$cat /sys /kernel /debug/gpio
```

OpAmp Device As a Buffering Stage

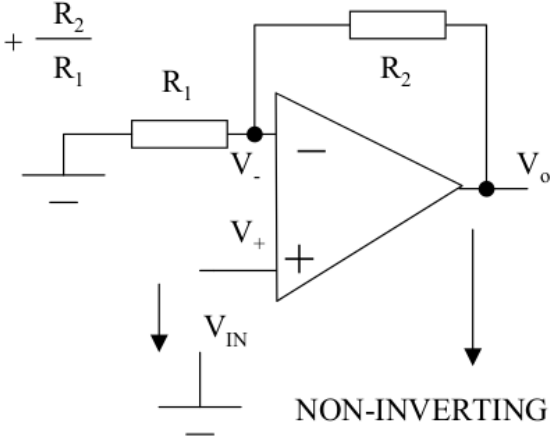
Both Analog and Digital Circuit



(1) To protect the previous stage's output signal, which is the input to the next stage, while sampling/connecting the signal to its next stage logic circuit. (2) Unit gain non-inverting OpAmp configuration is an excellent choice.

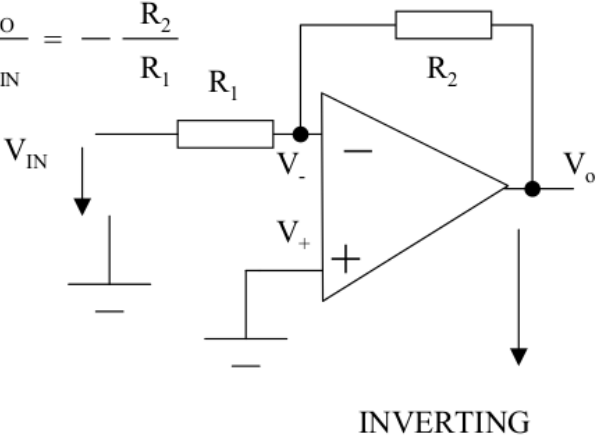
Ideal OpAmp Properties: (1) very large gain, A>>M; (2) draws very little current, $I_- \sim 0$, e.g., very high impedance; (3) $V_O = A(V_+ - V_-)$ is finite range, which leads to $V_+ = V_-$.

$$\frac{V_O}{V_{IN}} = 1 + \frac{R_2}{R_1}$$



NON-INVERTING

$$\frac{V_O}{V_{IN}} = -\frac{R_2}{R_1}$$

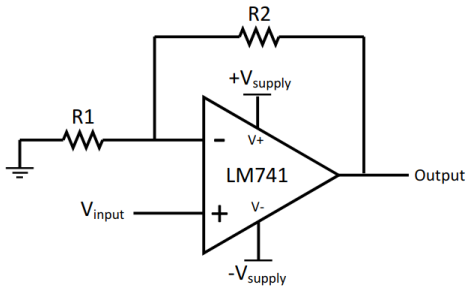


INVERTING

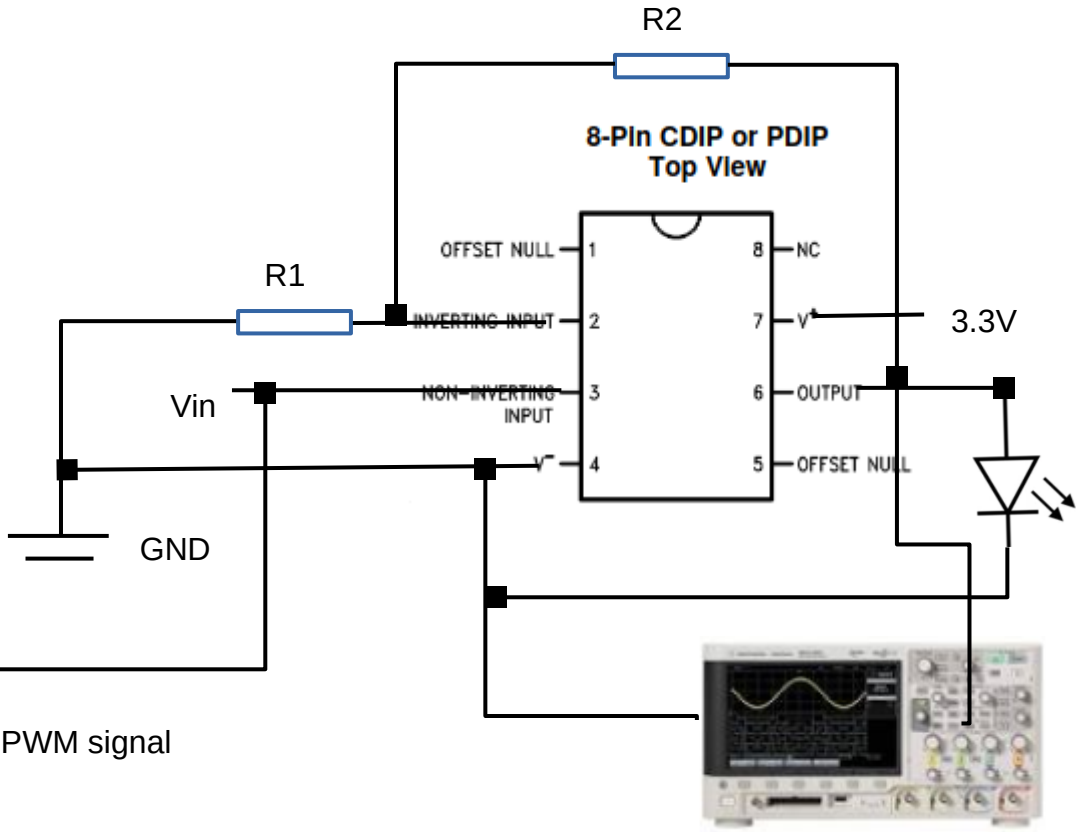
Hardware Buffer For Interface PWM

1. Gain A = 1 + R2/R1

2. Set -V_supply = 0



	GND	25	26	SPI_1_CS1	gpio20
	I2C_1_SDA *I2C Bus 0*	27	28	I2C_1_SCL *I2C Bus 0*	
gpio149	CAM_AF_EN	29	30	GND	
gpio200	GPIO_PZ0	31	32	LCD_BL_PWM	gpio168
gpio38	GPIO_PE6	33	34	GND	
gpio76	I2S_4_LRCK	35	36	UART_2_CTS	gpio51
gpio12	SPI_2_MOSI	37	38	I2S_4_SDIN	gpio77
	GND	39	40	I2S_4_SDOUT	gpio78



Harry Li, Ph.D.

NANO for Stepper Motor Drive