**Name:**
**Date:**

**PWM**

**We know by now that PWM is represented in Arduino language as a value somewhere between 0 and 255. We also know that the PWM signal is just the RedBoard turning the PWM pin ON, or HIGH, then LOW, or OFF, a bunch of times really, really fast. The thing is the computer can turn the signal HIGH and LOW a lot faster than a human can follow. So what does the PWM signal look like to us and how can we measure it? Good question. Upload the PWM Expansion Code to your Arduino, attach PWM pin # 9 to ground on your breadboard and measure the voltage between the two wires. Switch values and reload the code as necessary to fill in the table below.**

**Fill in the table below:**

| Percentage: | 0 | 10% | 20% | 30% | 40% | 50% | 60% | 70% | 80% | 90% | 100% |
|---|---|---|---|---|---|---|---|---|---|---|---|
| PWM value: | 0 | | | | | 127 | | | | | 255 |

**What is the equation you use to convert the percentage to PWM values and back again?**

**Why does the PWM value only go up to 255 if it represents 256 different values?**

**Often you will use analog sensors with your digital microprocessor. These analog sensors are even more difficult than the PWM signal. They talk to the microprocessor in 1024 little pieces!**

**Fill in the table below:**

| Percentage: | 0 | 10% | 20% | 30% | 40% | 50% | 60% | 70% | 80% | 90% | 100% |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Analog value: | 0 | | | | | 511 | | | | | 1023 |

**What is the equation you use to convert the percentage to analog values and back again?**

**Fill in the table below:**

| Percentage: | 0% | 25% | 50% | 75% | 100% |
|---|---|---|---|---|---|
| PWM value: | 0 | | | | 255 |
| Analog value: | 0 | | | | 1023 |

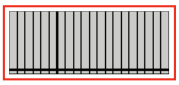
**What is the relationship between the analog and PWM value? Explain with words or math.**

**What is one advantage to representing an analog sensor's value with 1024 numbers instead of just 100?**

**Name:**
**Date:**

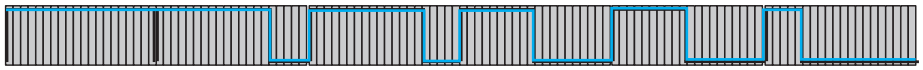
## PWM

**Below are a bunch of different PWM wave diagrams.**
**The "window" is this size:**

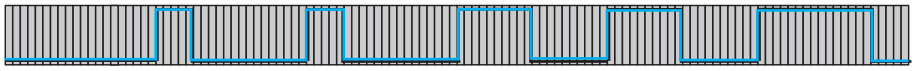


**Record below each wave diagram the starting value in the unit given, the highest value the PWM wave reaches (PWM value, 0 - 255), the lowest value the PWM wave reaches (PWM value, 0 - 255), and the ending value of the wave diagram. Finally, draw a line through the diagram showing the PWM value as it rises and falls.**

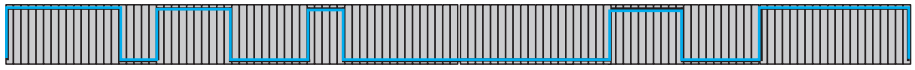
**Remember for the SIK PWM values: 100% = 255 PWM = 1.8mV**

1. 

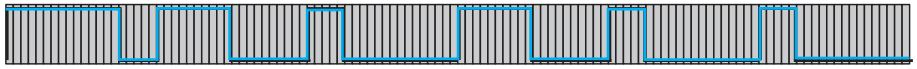
_____starting PWM _____Highest PWM _____Lowest PWM _____ending PWM

2. 

_____starting % _____Highest PWM _____Lowest PWM _____ending %

3. 

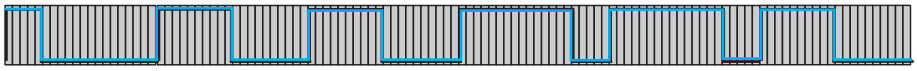
_____starting voltage _____Highest PWM _____Lowest PWM _____ending voltage

4. 

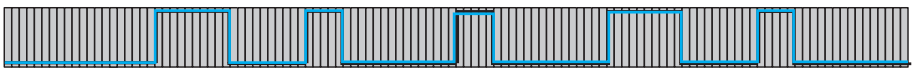
_____starting PWM _____Highest PWM _____Lowest PWM _____ending PWM

5. 

_____starting % _____Highest PWM _____Lowest PWM _____ending %

6. 

_____starting voltage _____Highest PWM _____Lowest PWM _____ending voltage

7. 

_____starting PWM _____Highest PWM _____Lowest PWM _____ending PWM

**These PWM windows are nicely divided into ten sections so you can count out the windows and figure out the PWM values and percentages easier. Real PWM signals are not so convenient and are usually represented by one of the following with no diagram: the PWM value, a percentage, or a voltage value.**

**PWM**

**We know by now that PWM is represented in Arduino language as a value somewhere between 0 and 255. We also know that the PWM signal is just the Arduino turning the PWM pin ON, or HIGH, then LOW, or OFF, a bunch of times really, really fast. The thing is the computer can turn the signal HIGH and LOW a lot faster than a human can follow. So what does the PWM signal look like to us and how can we measure it? Good question. Upload the PWMExpansion Code to your Arduino, attach PWM pin # 9 to ground on your breadboard and measure the voltage between the two wires. Switch values and reload the code as necessary to fill in the table below.**

| Percentage: | 0% | 10% | 25% | 33% | 50% | 66% | 75% | 80% | 90% | 100% |
|---|---|---|---|---|---|---|---|---|---|---|
| PWM value: | 0 | | 63.75 | | | | 191.25 | | | |
| Analog Value | 0 | | | | 511.5 | | | | | 1023 |
| Voltage (mV) | 0 | | | | | | | | | |

**One of the easiest ways to find a correlation between numbers is to look at a lower set of values and compare them to a larger set of values with a common denominator. Look at all the values for 10% and then look at all the values for 100%. If there is a pattern it should be easier to see it with these two set of values simply because they are different by a factor of 10, or one decimal place.**

**Using the table above create an equation that you can use to change PWM or Percentage into an analog value. Don't forget that zero is a value in PWM and analog. Write that equation below:**

**_____=analog value**

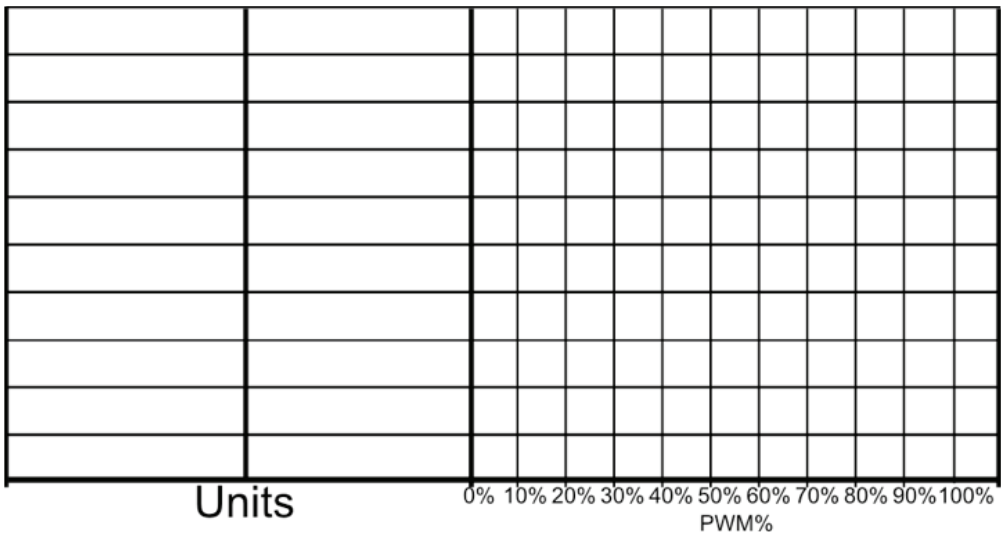
**Using the table above create an equation that you can use to change a PWM or Percentage value into a voltage value. Take into account any difference in units that may make the math weird. Write that equation below:**

**_____=voltage (mV)**

**Now, you should be able to mash these two equation together into one equation that you can use to convert from analog to voltage. Think this is tough? Lucky we're not using a temperature sensor and making you also convert Fahrenheit and Celsius and resistance values! Write your equation below.**

**_____**

**Using your equations fill in the line graph below for the analog and voltage variables above. Label your graph wherever you think necessary. Use a different color for the analog and voltage lines.**



**Name:**
**Date:**

## PWM

**Write below these different values whether they are a PWM value, an analog sensor value, a percentage or a voltage. Then match each value below with the PWM wave diagram that best represents them by drawing a line from the value to the diagram.**

**50%**

_____

**1.8 mV**

_____

**analogWrite(pin, 64)**

_____

**analogRead(pin)= 768**

_____

**511**

_____

**194 out of 256**

_____

**.45mV**

_____

**100 out of 100**

_____