```
'''
 Q. What is PWM?
 A. A Pulse Width Modulation (PWM) Signal is a method for generating an analog
signal using a digital source. A PWM signal consists of two main components
that define its behavior: a duty cycle and a frequency. The duty cycle
describes the amount of time the signal is in a high (on) state as a percentage
of the total time of it takes to complete one cycle. The frequency determines
how fast the PWM completes a cycle (i.e. 1000 Hz would be 1000 cycles per
second), and therefore how fast it switches between high and low states. By
cycling a digital signal off and on at a fast enough rate, and with a certain
duty cycle, the output will appear to behave like a constant voltage analog
signal when providing power to devices.

Example: To create a 3V signal given a digital source that can be either high
(on) at 5V or low (off) at 0V, you can use PWM with a duty cycle of 60% which
outputs 5V 60% of the time. If the digital signal is cycled fast enough, then
the voltage seen at the output appears to be the average voltage. If the
digital low is 0V (which is usually the case) then the average voltage can be
calculated by taking the digital high voltage multiplied by the duty cycle, or
5V x 0.6 = 3V. Selecting a duty cycle of 80% would yield 4V, 20% would yield
1V, and so on.

PWM signals are used for a wide variety of control applications. Their main use
is for controlling DC motors but it can also be used to control valves, pumps,
hydraulics, and other mechanical parts.
'''
;-------------------------------------------------------------
import Adafruit_BBIO.GPIO as GPIO
import Adafruit_BBIO.PWM as PWM
import time

#These four Pins are used for taking input from user
pwm_pins = ['P9_7', 'P8_8', 'P8_9', 'P8_10']

#Initialization: Set each pin as input
for i in range(len(pwm_pins)):
     GPIO.setup(pwm_pins[i], GPIO.IN)

#This array defines the value to be used as duty cycle on giving
specific input
duty_cycle = [20,90,50,0]

#One of the 8 ports for PWM, see the image in FAQ's.docx
PWM.start("P8_13")


while True:
     if (GPIO.input("P8_8")==0):
          PWM.set_duty_cycle("P8_13", duty_cycle[0])
     elif (GPIO.input("P8_7")==0):
          PWM.set_duty_cycle("P8_13", duty_cycle[1])
     elif (GPIO.input("P8_10")==0):
          PWM.set_duty_cycle("P8_13", duty_cycle[2])
     elif (GPIO.input("P8_9")==0):
          PWM.set_duty_cycle("P8_13", duty_cycle[3])
```

Output :

```
alka@comp-alka:~$ sudo su
[sudo] password for alka:
root@comp-alka:/home/alka# minicom -s


Debian GNU/Linux 7 beaglebone ttyGS0

default username:password is [debian:temppwd]

Support/FAQ: http://elinux.org/Beagleboard:BeagleBoneBlack_Debian

The IP Address for usb0 is: 192.168.7.2
beaglebone login: debian
Password:
Last login: Wed Apr 23 20:20:55 UTC 2014 on ttyGS0
Linux beaglebone 3.8.13-bone47 #1 SMP Fri Apr 11 01:36:09 UTC 2014 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
debian@beaglebone:~$su

root@beaglebone:~$
root@beaglebone:$ python pwm.py
^Z

[1]+  Stopped                 python pwm.py
```

```
root@beaglebone:$ cpufreq-info

cpufrequtils 008: cpufreq-info (C) Dominik Brodowski 2004-2009
Report errors and bugs to cpufreq@vger.kernel.org, please.
analyzing CPU 0:
  driver: generic_cpu0
  CPUs which run at the same hardware frequency: 0
  CPUs which need to have their frequency coordinated by software: 0
  maximum transition latency: 300 us.
  hardware limits: 300 MHz - 1000 MHz
  available frequency steps: 300 MHz, 600 MHz, 800 MHz, 1000 MHz
  available cpufreq governors: conservative, ondemand, userspace, powersave,pee
  current policy: frequency should be within 300 MHz and 1000 MHz.
                  The governor "ondemand" may decide which speed to use
                  within this range.
  current CPU frequency is 300 MHz.
  cpufreq stats: 300 MHz:58.43%, 600 MHz:0.00%, 800 MHz:0.00%, 1000
MHz:41.57% )
```

```
root@beaglebone:$ sudo cpufreq-set -f 600Mhz


root@beaglebone:$ cpufreq-info
cpufrequtils 008: cpufreq-info (C) Dominik Brodowski 2004-2009
Report errors and bugs to cpufreq@vger.kernel.org, please.
analyzing CPU 0:
  driver: generic_cpu0
  CPUs which run at the same hardware frequency: 0
  CPUs which need to have their frequency coordinated by software: 0
  maximum transition latency: 300 us.
  hardware limits: 300 MHz - 1000 MHz
  available frequency steps: 300 MHz, 600 MHz, 800 MHz, 1000 MHz
  available cpufreq governors: conservative, ondemand, userspace, powersave,pee
 current policy: frequency should be within 300 MHz and 1000 MHz.
                      The governor "userspace" may decide which speed to use
                 within this range.
  current CPU frequency is 600 MHz.
  cpufreq stats: 300 MHz:64.74%, 600 MHz:2.07%, 800 MHz:0.00%, 1000
MHz:33.19% )
```
