TRAFFIC MANAGEMENT

Phase-4 :development part -2

Source code :

```
import pyfirmata  import
time comp='COM8'  board
= pyfirmata.Arduino(comp)
led_1=board.get_pin('d:2:o')
led_2=board.get_pin('d:3:o')
led_3=board.get_pin('d:4:o')
led_4=board.get_pin('d:5:o')
led_5=board.get_pin('d:6:o')
led_6=board.get_pin('d:7:o')
l1=[0,0]  def normaltime():
return 2 def
maxtime():
return 10 def
val1():     return
l1[0] def
val2():     return
l1[1] def
signalamb1():
```

```
l1[0]=1
l1[1]=0
led_2.write(0)
led_5.write(0)
led_3.write(1)
led_4.write(1)
def
signalamb2():
l1[0]=0
l1[1]=1
led_2.write(0)
led_6.write(1)
led_1.write(1)
led_5.write(0)
def signal1():
l1[0]=1 l1[1]=0
led_2.write(0)
led_5.write(0)
led_3.write(1)
led_4.write(1)
def wait():
l1[0]=0
```

```
l1[1]=0
led_2.write(1)
led_3.write(0)
led_4.write(0)
led_5.write(1)
led_1.write(0)
led_6.write(0)
def signal2():
l1[0]=0
l1[1]=1
led_2.write(0)
led_6.write(1)
led_1.write(1)
led_5.write(0)
def led_norm():
l1[0]=1
l1[1]=0 led_2.write(0)
led_5.write(0)
led_3.write(1)
led_4.write(1)
    time.sleep(5)
led_2.write(1)
```

```
led_3.write(0)
led_4.write(0)
led_5.write(1)
time.sleep(1)
l1[0]=1     l1[1]=0
led_2.write(0)
led_6.write(1)
led_1.write(1)
led_5.write(0)
time.sleep(5)
led_2.write(1)
led_1.write(0)
led_6.write(0)
led_5.write(1)
time.sleep(1) def
signalmax_1():
l1[0]=1 l1[1]=0
led_2.write(0)
led_5.write(0)
led_3.write(1)
led_4.write(1)
time.sleep(15)
```

```
led_2.write(1)
led_3.write(0)
led_4.write(0)
led_5.write(1)
time.sleep(1)
l1[0]=1     l1[1]=0
led_2.write(0)
led_6.write(1)
led_1.write(1)
led_5.write(0)
time.sleep(15)
led_2.write(1)
led_1.write(0)
led_6.write(0)
led_5.write(1)
time.sleep(1) def
signalmax_2():
l1[0]=1 l1[1]=0
led_2.write(0)
led_5.write(0)
led_3.write(1)
led_4.write(1)
```

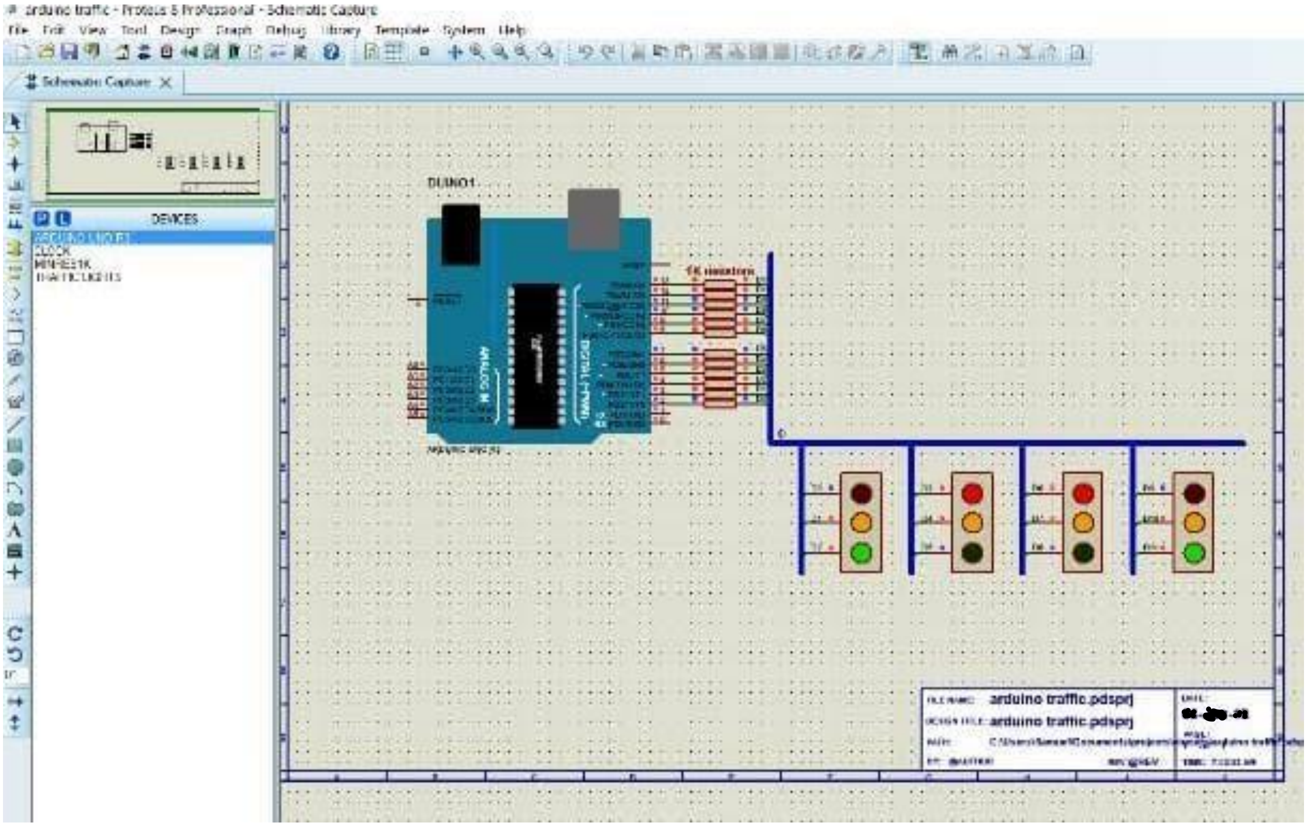
```
time.sleep(15)
led_2.write(1)
led_3.write(0)
led_4.write(0)
led_5.write(1)
time.sleep(1)
l1[0]=1     l1[1]=0
led_2.write(0)
led_6.write(1)
led_1.write(1)
led_5.write(0)
time.sleep(15)
led_2.write(1)
led_1.write(0)
led_6.write(0)
led_5.write(1)
time.sleep(1)
   {
import cv2 from signals import
* import time cascade_src =
'cars.xml' video_src =
'dataset/test2.mp4' video_src1 =
'dataset/test3.mp4' l=[0,0] cap =
cv2.VideoCapture("dataset/test2.mp
4") cap1
=
cv2.VideoCapture("dataset/test3.mp
4")
car_cascade =
cv2.CascadeClassifier(cascade_src)
ret, img = cap.read() ret1,img1 =
cap1.read() num=0 def returnval():
return l[:] def fun1(time1):
now=time.time()     timer = 0
while timer<=time1:         ret, img
= cap.read()         ret1,img1 =
cap1.read()         if (type(img) ==
type(None)):             break
if(type(img1)==type(None)):
            break         gray =
cv2.cvtColor(img,
cv2.COLOR_BGR2GRAY)
        gray1 = cv2.cvtColor(img1,
cv2.COLOR_BGR2GRAY)
        #cv2.imshow('video', gray)         cars =
car_cascade.detectMultiScale(gray,
1.1, 1)
        cars1 =
car_cascade.detectMultiScale(gray1,
1.1, 1)         for (x,y,w,h)
in cars:

cv2.rectangle(img,(x,y),(x+w,y+h),(
0,0,255),2)         for
(x,y,w,h) in cars1:

cv2.rectangle(img1,(x,y),(x+w,y+h),
(0,0,255),2)
cv2.imshow('video', img)
cv2.imshow('video1', img1)
if(val1()==0):
l[0]=len(cars)
if(val2()==0):
l[1]=len(cars1)         if
cv2.waitKey(33) == 27:
break         end = time.time()
timer = round(end-now)
print(timer)
cv2.destroyAllWindows()
```

```
        #return num
}
```

Source code output diagram


arduino traffic - Proteus 8 Professional - Schematic Capture
File Edit View Tool Design Graph Debug Library Template System Help
Schematic Capture
DEVICES
CLOCK
MINRES1K
TRAFFICLIGHTS
DUINO1
1K resistors
ANALOG IN
DIGITAL (PWM)
arduino traffic.pdsprj
arduino traffic.pdsprj