**AKILLI VE GÜVENLİ**

**Rasperry Pi Kullanarak Havadaki Metan Gazını Algılayarak Ölçmek**

**Giriş**

Bu proje MQ-4 kullanılarak yapılan bir tasarımıdır. Rasperry pi , MQ-4 kullanılarak eşik değerini geçtiği zaman mail yoluyla tehlike anında bildirim gönderilmesi planlanmıştır. Bu proje ile evde olduğunuz ya da olmadığınız zaman güvenliğin sağlanması amaçlanıyor.

Projede, var olan sensörümüzün eşik değerleri ile ilgili çıktıları servis yardımıyla mail atıp uyaracaktır. MQ 4 yapısı gereği analog çıkış veren bir sensördür faka raspberry pi 3 te herhangi bir analog giriş olmaması nedeniyle MCP3008 ADC kullanılması gerekmektedir. MQ 4’ün AO girişine MCP’nin channel0 bacağından gidilmiştir. GND ve VCC si ise board üzerinden alınmıştır. GND ile AO pini arasına bir direnç bağlanmıştır. ADC’nin 16 ve 15 numaralı pini board üzerinden +5 Volt ile beslenirken 14 numaralı pini ise GND olarak topraklama yapılmıştır. 13, 12, 11ve 10 numaralı pinler Raspberry Pi 3’e bağlanmıştır. (Şekil1.1 de daha net anlaşılması için fritzing çizimi verilmiştir.)

**Gerekli Donanım Bileşenleri**

**1.** 1 adet Rasperry pi

**2.** 1 adet MQ4

**3.** 1 adet MCP3008

**Gerekli Yazılım Bileşenleri**

1. Python 3

**Kullanılan Bileşenlerin Özellikleri**

1. Rasperry pi 3; üzerinde 40 adet GPIO ve ARMV8 işlemci bulunduran SD kart ile içerisine işletim sistemi yüklenebilen bir geliştirici karttır.(detaylı bil için <http://market.samm.com/raspberry-pi-3>)
2. MQ4;Bu ürün INT-EL adlı firmadan tedarik edildi.Ürün avadaki gazların oranını ölçmemize ve değişimine göre çıkış voltajının değişimini algılayıp işlem yapmamıza olanaksağlar.
3. MCP3008; Bu ürünü robotistan adlı şirketten tedarik ettik. Raspberry Pi üzerinde analog bir giriş bulunmadığı için analog veriyi dijital veriye dönüştürmek amacı ile kullandık.(https://www.robotistan.com/mcp3008-8-kanal-10-bit-aServo)

\*\*Ürünleri verilen linklerden temin edebilirsiniz.

**Şematik Çizimi**

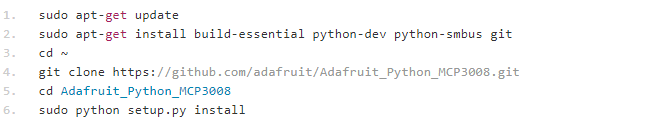
**Şekil1.1**

**Yapım Aşamaları**

**MQ4 Kurulumu**

MQ4’i kullanabilmek için öncelikli olarak MCP3008 ADC kütüphanesini indirmek gerekir.

Bunu yapabilmek için komut satırımızda sırasıyla aşağıdaki işlemler gerçekleştirilmelidir.



**Adım 1:** İlk olarak Raspberry Pi'deki Raspbian OS'yi güncellemek için aşağıdaki komutu çalıştırın:

sudo apt-get update

**2. Adım:** Daha sonra aşağıdaki komutu kullanarak "**Adafruit\_Python\_MCP3008**" Kitaplığı'nı yükleyin:

git clone https://github.com/adafruit/Adafruit\_Python\_MCP3008.git

**3. Adım:** cd komutunu kullanarak Kitaplığın içine giriniz.

cd Adafruit\_Python\_MCP3008

**4. Adım**: yükleme işleminin kurulumunu ayarlamak için aşağıdaki kodu yazıyoruz.

sudo python setup.py install

**MQ4 kodunun yüklenmesi**

**git clone git://gist.github.com/3151375.git**

komut satırından indirerek kodu kullanabilirsiniz. Kod MQ135 için hazırlanmış olup bütün MQ sensörlerinde kullanılabilmektedir.

**Python Kodu**

1. **from** email.mime.multipart **import** MIMEMultipart
2. **from** email.mime.text **import** MIMEText
3. **import** smtplib
5. **import** time
6. **import** os
7. **import** RPi.GPIO as GPIO
8. **import** datetime
9. **import** sys
10. **from** time **import** sleep
12. mode=GPIO.getmode()
14. GPIO.setwarnings(False)
15. GPIO.setmode(GPIO.BCM)
17. DEBUG = 1
19. msg = MIMEMultipart()
20. msg['From'] = "göndericimaili"
21. msg['To'] = "alıcımaili"
22. msg['Subject']= "Acil durum"
23. password = 'gönderimailşifresi'
25. body = "Evinizde doğalgaz kaçağı olabilir lütfen süvarileri çağırın!"
26. msg.attach(MIMEText(body, 'html'))
27. **print**(msg)
29. # MCP'nin 0-7 pin arası kullanılabilir. Onları aktif hale getiriyoruz.
30. **def** readadc(adcnum, clockpin, mosipin, misopin, cspin):
31. **if** ((adcnum > 7) **or** (adcnum < 0)):
32. **return** -1
33. GPIO.output(cspin, True)
35. GPIO.output(clockpin, False)  # baslangic sıfırlama
36. GPIO.output(cspin, False)     # CS sıfırlama
38. commandout = adcnum
39. commandout |= 0x18  # baslangic biti
40. commandout <<= 3    # sadece 5 bit kullanmalıyız
41. **for** i **in** range(5):
42. **if** (commandout & 0x80):
43. GPIO.output(mosipin, True)
44. **else**:
45. GPIO.output(mosipin, False)
46. commandout <<= 1
47. GPIO.output(clockpin, True)
48. GPIO.output(clockpin, False)
50. adcout = 0
51. # ADC bitlerini oku
52. **for** i **in** range(12):
53. GPIO.output(clockpin, True)
54. GPIO.output(clockpin, False)
55. adcout <<= 1
56. **if** (GPIO.input(misopin)):
57. adcout |= 0x1
59. GPIO.output(cspin, True)
61. adcout >>= 1       # İlk biti null yap
62. **return** adcout
64. # Bağlantılı olan pinler, istenildiği gibi ayarlanabilir.
65. # ADC için SPI portları
66. SPICLK = 18
67. SPIMISO = 23
68. SPIMOSI = 24
69. SPICS = 25
71. # GPIO pinleri için setupları kurmak için yazılması gerekenler
72. GPIO.setup(SPIMOSI, GPIO.OUT)
73. GPIO.setup(SPIMISO, GPIO.IN)
74. GPIO.setup(SPICLK, GPIO.OUT)
75. GPIO.setup(SPICS, GPIO.OUT)
77. # MQ-135 adc bağlantısı
78. MQ135\_adc = 0;
80. last\_read = 0       # Sensöre gelen son veri
81. tolerance = 5       # Sadece değişim
82. # MQ-135 değeri 5'ten fazla değişimini algıla


86. **while** True:
87. # Sensörde değişim olamdığı durumda
88. trim\_pot\_changed = False
90. # Analog veriyi oku
91. trim\_pot = readadc(MQ135\_adc, SPICLK, SPIMOSI, SPIMISO, SPICS)
92. #Son okumanın değişimi
93. pot\_adjust = abs(trim\_pot - last\_read)
95. **if** DEBUG:
96. **print** ("Sensor\_verisi:", trim\_pot)
97. **print** ("Sensor\_girdisi:", pot\_adjust)
98. **print** ("Son\_okunan\_deger", last\_read)
100. **if** ( pot\_adjust > tolerance ):
101. trim\_pot\_changed = True
103. **if** DEBUG:
104. **print** ("trim\_pot\_changed", trim\_pot\_changed)
106. **if** ( trim\_pot\_changed ):
107. set\_volume = trim\_pot / 10.24           # 0-1024 arası değeri yüzde bir ifade olarak çevirmek için yapılmalı
108. set\_volume = round(set\_volume)          # Ondalık değer
109. set\_volume = int(set\_volume)            # Tam sayı
111. **print** ('Volume = {volume}%' .format(volume = set\_volume))
112. set\_vol\_cmd = ('sudo amixer cset numid=1 -- {volume}% > /dev/null' .format(volume = set\_volume))
113. os.system(set\_vol\_cmd)  # Düzey ayarı
115. **if** DEBUG:
116. **print** ("set\_volume", set\_volume)
117. **print** ("tri\_pot\_changed", set\_volume)
119. # MQ-135 den alınan veriyi bir sonraki döngü için kaydet
120. last\_read = trim\_pot
121. # 5 saniyede bir okumaları tekrar et ! Standart prosedürde daha uzun sürede okunması gerekebilir!
122. time.sleep(5)
124. **if**  set\_volume > 80:
125. server = smtplib.SMTP("smtp-mail.outlook.com", 587)
126. server.starttls()
127. server.login(msg['From'], password)
128. server.sendmail(msg['From'], msg['To'], msg.as\_string())
129. server.quit()

**Nasıl Kullanılır**

Bu projeyi kullanmak için belirtilen yerlerden veya kendi istediğiniz yerlerden malzemeleri temin ediniz. Daha sonrasında fritzing adlı programda sizin için hazırladığım bağlantı ayarlamalarını sağlayınız(Şekil1.1) . Şimdi komut satırında mq-4 adlı bir klasör oluşturunuz. Bunun içinse:

mkdir mq-4

yazmanız yeterlidir. ardından

cd mq-4

yazarak bu klasörün içine gidiniz. Burada

nano mq-4m.py

adlı bir dosya açınız. Dosyanın içine aşağıda paylaşılan kodu yazıp Ctrl+X kombinasyonunu uygulayıp kaydetmek için y tuluna basıp enter tuşuna basmlaısınız.

Sonra aşağıdaki komutu kullanarak Python kodunu çalıştırın:

python mq-4.py

**Öneriler**

Bu cihaz ileride birden fazla sensör ile maden ocaklarında da kullanılabilir.