```
################################################################################################################
#############################################
#powerOnOff.py - a-gsm 2.064 power on/power off/ modem communication example utility
#COPYRIGHT (c) 2014 Dragos Iosub / R&D Software Solutions srl
#
#You are legaly entitled to use this SOFTWARE ONLY IN CONJUNCTION WITH a-gsm DEVICES USAGE.
Modifications, derivates and redistribution
#of this software must include unmodified this COPYRIGHT NOTICE. You can redistribute this
SOFTWARE and/or modify it under the terms
#of this COPYRIGHT NOTICE. Any other usage may be permited only after written notice of
Dragos Iosub / R&D Software Solutions srl.
#
#This SOFTWARE is distributed is provide "AS IS" in the hope that it will be useful, but
WITHOUT ANY WARRANTY; without even the implied
#warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
#
#Dragos Iosub, Bucharest 2014.
#http://itbrainpower.net
################################################################################################################
#############################################
#Raspberry PI - a-gsm wiring connection:
# Legal disclaimer:
# Incorrect or faulty wiring and/or connection can damage your RPi and/or your a-gsm board!
# Following directives are provided "AS IS" in the hope that it will be useful, but WITHOUT
ANY WARRANTY!
# Do the wiring on your own risk!

#name       RPi     a-gsm shield
#
#POWER a-gsm    16      D7  - power(UP/DOWN)
#RESET a-gsm    18      D6  - reset
#a-gsm STATUS   12      D5  - status
#
#serial TXD0    08      D4  - tx(rxd)
#serial RXD0    10      D3  - rx(txd)
#
#5V         02/04       5V  - on Arduino power IN connector
#GND        06/14       GND - on Arduino power IN connector
#
#IMPORTANT:
# a-gsm's POWER supply input selector must be in "use 5V pin" position
################################################################################################################
#############################################

# this utility must be runned as root (just use: sudo python powerOnOff.py)

serialSpeed = 19200#we recommend usage of 19200 bps speed. If you want to use other speed,
first set the a-gsm speed using setSerial.py

#Do not change under following line! Insteed make one copy of the file and play with!
################################################################################################################
#############################################

#definitions for a-gsm control(RPi GPIO mode)
POWER = 16
RESET = 18
STATUS = 12
```

```
i=0
buffd =""
sreadlen = 100#how many chars to read in one try over serial

import os
import serial
from time import sleep, time
import RPi.GPIO as GPIO

if not os.getuid() == 0:
    print("please use root privileges! try: \"sudo python powerOnOff.py\"")
    exit(0)

agsm = serial.Serial("/dev/ttyAMA0", serialSpeed, timeout=1)
agsm.open()

print "Hy folks. Lets check the comunication with your a-gsm"

#poweron() - power up the modem
def poweron():
    #...function code here

#poweroff() - shutdown the modem
def poweroff():
    #...function code here

#recUARTdata(endchars,to,tm)
#   read from modem - read string is loaded in global var buffd
#   looking for endchars [SUCCESS STRING,FAILURE STRING] and to - TIMEOUT
#   return 0 for SUCCESS, 1 for FAILURE, -1 for timeout
#   tm how many chars to read(maximum) in one loop from serial
def recUARTdata(endchars,to,tm):
    #...function code here
    return SuccessErrorTimeout

#sendATcommand(command, endchars,to)
#   command +"\r\n" is forwarded to modem
#   looking for endchars [SUCCESS STRING,FAILURE STRING] and to - TIMEOUT
#   return 0 for SUCCESS, 1 for FAILURE, -1 for timeout
#   modem response is loaded in global var buffd
def sendATcommand(command, endchars,to):
    global sreadlen
    agsm.write(command+"\r\n")
    return (recUARTdata(endchars,to,sreadlen))

#aGsmWRITE(command)
#   just write command to serial without CR LF
def aGsmWRITE(command):
    agsm.write(command)


#RaspberryPI hardware setup section start
GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)
try:
    GPIO.setup(STATUS, GPIO.IN)
```

```
    GPIO.setup(POWER, GPIO.OUT, initial=GPIO.LOW)
    GPIO.setup(RESET, GPIO.OUT, initial=GPIO.LOW)
except:
    GPIO.cleanup()#free GPIO
    GPIO.setup(STATUS, GPIO.IN)
    GPIO.setup(POWER, GPIO.OUT, initial=GPIO.LOW)
    GPIO.setup(RESET, GPIO.OUT, initial=GPIO.LOW)
GPIO.setwarnings(True)
#RaspberryPI hardware setup section end


#here start the main code
poweron()

while True and i<5 :
        sendATcommand("ATE1",["OK","ERROR"],5);
        print buffd
        sleep(2)
        i=i+1
agsm.close()#close serial
poweroff()#shutdown a-gsm
GPIO.cleanup()#free GPIO
```