

```
#####
#####
#readSMS.py - a-gsm 2.064 list/read sms example utility
#COPYRIGHT (c) 2014 Dragos Iosub / R&D Software Solutions srl
#
#You are legally 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 permitted 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 readSMS.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
usePoweringControl = 1#change it to 0 if you do not want to control powerUP/powerDown the
a-gsm board. In this case, please be sure the a-gsm board is powered UP(the a-gsm green led
lights continuous) before run this utility

#Do not change under following line! Instead 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

noSMS = 0#SMS stored count
totSMS = 0#total SMS capacity
SMSmessage = [ "", "", "", "" ]

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

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

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

print "Hy folks. Lets read one SMS from your a-gsm shield"

#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)

#listSMS()
# find stored SMS number(==>noSMS) and total SMS number (==>totSMS)
def listSMS():
    global noSMS
    global totSMS
    #...function code here
    print("noSMS: "+str(noSMS))
    print("totSMS: "+str(totSMS))

#readSMS(SMSindex)
# sweet little baby... read the SMS found at SMSindex
# extract and store the SMS content in global var SMSmessage
# SMSmessage[0] type(REC) READ/UNREAD
# SMSmessage[1] sender number
# SMSmessage[2] SMS date and time
# SMSmessage[3] SMS content(message)
def readSMS(SMSindex):
    global sreadlen
    global SMSmessage
    #...function code here

#setupMODEMforSMSusage()
# ...and Deedee said: "Ooooooo, what does this button do?"
def setupMODEMforSMSusage():
    #...function code here

#RaspberryPI hardware setup section start
if usePoweringControl==1:
    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

if usePoweringControl==1:
    poweron()
```

```
setupMODEMforSMSusage()

res = listSMS()
if res==0:
    print "list SMS executed with succes"

readSMS(1)

print "SMS type: "+SMSmessage[0]
print "Sender no.: "+SMSmessage[1]
print "Date and time local: "+SMSmessage[2]+" ...divide to 4 the last 2 digits to reach UTC
standard offset(in hours). This is dependent on your NMO's behavior!"
print "Date and time UTC: "+ str(SMSmessage[2])[0:-3]
print "...and the message content:\r\n"+SMSmessage[3]
print ""

agsm.close()#close serial

sleep(5)

if usePoweringControl==1:
    poweroff()#shutdown a-gsm

GPIO.cleanup()#free GPIO
```