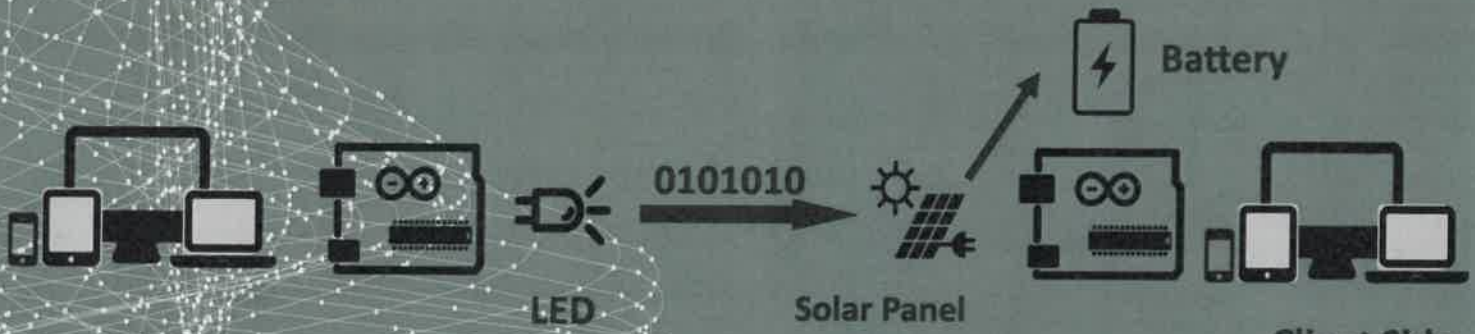
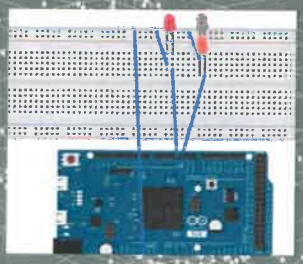
LiFi—Visible Light Communication System

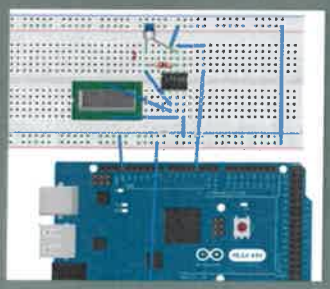
Yiren Ramon Qu

Lyndon Institute, VT

It's More Than Light







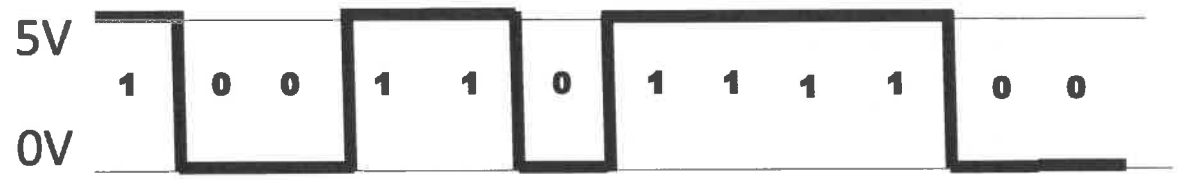
#Li-Fi #DataTransmission #Wireless #Light

Encoding the Binary Data with LED

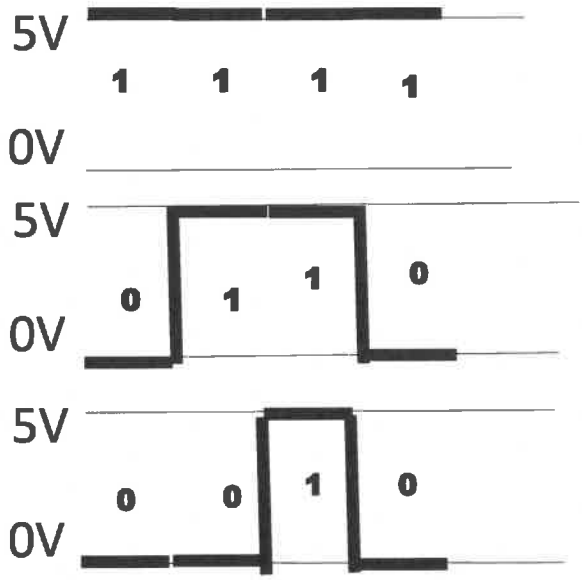
Single Channel (Baseband)

'On-Off' Encoding

Examples binary data is 100110111100



Multi Channel (Broadband)



Transmitting Images

Base64 From an Image:

/9j/4AAQSkZJRgABAQAAAQ....

Convert every character to Binary and padding 0 in the front to be 8 bits/character:

00101111 00111001 01101010....

After received the binary data, regroup to 8 bits and convert back based on ASCII code.

Chr(47) Chr(57) Chr(106) ...

Transmission Protocols

The protocol I used for this system is the 'on-off' protocol, similar to UDP (User Datagram Protocol) which transmits data one-way. The data packet (the smallest unit of a transmission) is 15 bits. A series of data contains a header and multiple data packets. The header gives the number of the packets in one series.

The procedures of transmitting the pictures:

Use Python to read the image -> Convert to Base64 data -> Turn every character to binary (8 bit/character) -> Regroup to 540 bits/ packet (The limit of one-time send to Arduino) -> Regroup to 15 bit/packet -> To Receiver side.

Receive binary data -> check the size -> convert back to characters -> Convert back to base64 and save as Image.

```
'''
Li-Fi VIsible Light Communication System. Yiren Ramon Qu,
Lyndon Institute, VT
MIT License
Copyright (c) 2017 Yiren Qu

Li-Fi Transmitter Side
'''
import serial
import binascii
import sys
import random
from tqdm import tqdm
import base64
import time

def pic(s):
    '''
    :param s: The image link
    Function reads the imagefile and converts it to Base-64
string list.
    :return: Base-64 string file.
    '''
    with open(s,"rb") as imageFile:
        strr = base64.b64encode(imageFile.read())
        print(str(strr)[2:-1])
        return str(strr)[2:-1]
def bin_str(s):
    '''
    :param s: a character
    :return: The ASCII number in binary with padding 0s to make
 them uniform.
    '''
    data = ""
    for i in s:
        temp = str(bin(ord(i)))[2:]
        data +=temp.rjust(8,'0')
    return data

ser = serial.Serial('COM4',115200,timeout=.1) #Begin the Serial
 connection
print("#####################################################\
```

```
n"
      "##--------------------START-------------------------##\
n"
      "#############################################################\
n"
      "##-------------------Emitter-------------------------##\
n"
      "##---------------------Side-------------------------##\
n"
      "##----------------Start Sending-- ------------------##\
n"
      "##-------------------------------------------------##\
n"
      "#############################################################\
n"
      )

img_link = "C:/Users/Ramon Qu/Desktop/vlc/untitled/exp2.jpg"
test = ["Ramon Qu",
        "Lyndon Institute",
        ":-)",
        "Happy",
        "Today is May 1st",
        "Happy Monday",
        "Welcome to the fair"]

while(not "DONE".encode() in ser.readline()):
    pass
flag = False
while True:
    a = input()
    '''
    input 's', 't' or 'd' to select the mode you would like to
use.
    's'  -  Transmit the image (Set the link before running it)
    't'  -  Text Mode. You can input the text and send to the
other side
    'd'  -  Auto Text Mode.  THere are test string list. Every
time, the function will randomly pick one and send to the other
 side.
    '''
    if (a=="s"):
```

```
        print(
            "##-------------Transmitting Image Mode
--------------##\n"

"##-------------------------------------------------##\n"

"|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||\n"
            "
VVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV\n"

"                                                             \n"
        )
        pbar = tqdm(total=100)    # Progress bar function.
        t0 = time.clock()         # Initial the clock
        data = bin_str(pic(img_link))
        temp = len(data)
        # Parse the data to 540 bits/ group
        data = [ data[i:i+540] for i in range(0, len(data), 540
) ]
        # Send the number of packets to emitter controller.
        ser.write(bytes("{0:b}".format(temp//8).rjust(15,'0')+"
\n",encoding="ascii"))
        print(temp//8)
        ser.flush()
        print(len(data))
        # Wait until the emitter confirming the message has
been sent
        while(not "DONE".encode() in ser.readline()):
            pass
        # Send the packets. 540 bits/ group. The emitter will
regroup them into 15 bits/ packet
        for i in range(len(data)):
            ser.write(bytes(data[i]+"\n",encoding="ascii"))
            ser.flush()
            pbar.update(1/len(data)*100
            while(not "DONE".encode() in ser.readline()):
                    pass
        print(bin_str(pic(img_link)))
        print(time.clock()-t0)
    if(a=="t"):
        print(
            "##-------------Transmitting Text Mode
```

```
---------------##\n"

"##-------------------------------------------------##\n"
              "##  Please Type in what you would like to transmit
  ##\n"

"||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||\n"
              "
vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv\n"

"                                                            \n"
        )
        count = 0
        while(1):
            data = input()
            print("No."+str(count)+" -- Raw Data: ")
            # Serial write a start signal  character
            ser.write(bytes("00010\n",encoding="ascii"))
            ser.flush()
            while(not "DONE".encode() in ser.readline()):
                    pass
            #data is the input. It turns very character to 8
bit binary string and send to the emitter controller
            for i in data:
                ser.write(bytes(str(bin(ord(i))).rjust(8,'0')+"
\n",encoding="ascii"))
                ser.flush()
                print(bytes(str(bin(ord(i))).rjust(8,'0'),
encoding="ascii"),end="")
                while(not "DONE".encode() in ser.readline()):
                    pass
            #Send the end signal character
            ser.write(bytes("00011\n",encoding="ascii"))
            ser.flush()
            print()
            count+=1
            while(not "DONE".encode() in ser.readline()):
                    pass
    if (a=="d"):
        print(
            "##----------Auto Transmitting Text Mode
------------##\n"
```

```
"##------------------------------------------------##\n"

"||||||||||||||||||||||||||||||||||||||||||||||||||||\n"
             "
VVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV\n"

"                                                    "
        )
        last = -1
        count = 0
        while (1):
            #Random Select one
            temp = random.randint(0, len(test) - 1)
            while(temp ==last):
                temp = random.randint(0, len(test) - 1)
            last = temp
            data = test[temp]
            # The same method used in the text mode.
            print("No."+str(count)+"--->   "+str(data))
            count+=1
            ser.write(bytes("00010\n", encoding="ascii"))
            ser.flush()
            while (not "DONE".encode() in ser.readline()):
                pass
            for i in data:
                ser.write(bytes(str(bin(ord(i))).rjust(8, '0')
+ "\n", encoding="ascii"))
                ser.flush()
                while (not "DONE".encode() in ser.readline()):
                    pass

            ser.write(bytes("00011\n", encoding="ascii"))
            ser.flush()
            while (not "DONE".encode() in ser.readline()):
                pass
            time.sleep(1)
```

```
'''
Li-Fi VIsible Light Communication System. Yiren Ramon Qu,
Lyndon Institute, VT
MIT License
Copyright (c) 2017 Yiren Qu

Li-Fi Receiver Side
'''
import serial
import binascii
import sys
import base64
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
from tqdm import tqdm

ser = serial.Serial('COM3', 115200, timeout=.1) # Start the
serial connectiion


print("############################################################\
n"
      "##--------------------START-------------------------##\
n"
      "############################################################\
n"
      "##--------------------Receiver----------------------##\
n"
      "##---------------------Side-------------------------##\
n"
      "##----------------Start Receiving -----------------##\
n"
      "##-----------------------------------------------------##\
n"
      "############################################################\
n"
      )
a = input()
'''
Input 's' or 't'
's'  - Receving the image
't'  -  Receving the text
```

```
'''
if (a == "s"):
    print(
      "##-------------Receiving Image Mode-------------##\n"
      "##------------------------------------------------##\
n"
      "||||||||||||||||||||||||||||||||||||||||||||||||||\
n"
      "VVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV\
n"
      "                                                  \
n"
     )
    flag = False
    while True:
        a = ser.readline()
        data = a
        if (len(a) > 0):
            pbar = tqdm(total=100)
            if (not flag):
                n = int(a, 2)
                # n is the number of packets need to be
received
                print(n)
                pic = ""
                picdata = ""
                count = 0
                while len(pic) < n * 8:
                    # Receive the packet and save back to a
complete binary string
                    a = ser.readline()
                    if (len(str(a)) > 15):
                        pic += str(a)[2:17]
                        pbar.update(15 / n / 8 * 100)
                    elif (len(str(a)) > 3):
                        pic += str(a)[2:-5]
                        pbar.update((len(str(a)) - 7) / n / 8 *
 100)
                print(pic)
                i = 0
                #Parse the image to 8 bits/group and convert
back to character
```

```
                    data = [pic[i:i + 8] for i in range(0, len(pic)
, 8)]
                    print(len(data))
                    for i in data:
                        data_part = i
                        x = int(data_part, 2)
                        message = chr(x)
                        picdata += message
                    print(picdata)
                    #The string is a base-64 string representing an
 image
                    picdata = base64.standard_b64decode(picdata)
                    #Save this image and open it with plot library
                    fh = open("C:/Users/Ramon Qu/Desktop/vlc/pypic/
img/imageToSave.jpg", "wb")
                    fh.write(picdata)
                    fh.close()
                    img = mpimg.imread('C:/Users/Ramon Qu/Desktop/
vlc/pypic/img/imageToSave.jpg')
                    plt.imshow(img)
                    plt.show()
if a=="t" :
    print(
        "##-------------Receiving Text Mode---------------##\n
"

"##-------------------------------------------------##\n"

"||||||||||||||||||||||||||||||||||||||||||||||||||||||||\n"
        "
VVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV\n"

"                                                        \n"
    )
    count = 0
    while(1):
        a = ser.readline()
        data = a
        if(str(data)[2:-5]=="00010"):
            #Detect whether received a start signal binary
         print("----------------> No."+(str(count)),end="")
         while(1):
```

```
                a = ser.readline()
                data = a
                if(str(data)[2:-5]=="00011"):
                    #If detecting the end signal, exit and wait
a new start.
                    print()
                    count+=1
                    break
                if(len(data)>3):
                    #If it is a character, print this character
on the same line.
                    print(chr(int(data,2)),end="")
                    sys.stdout.flush()
```