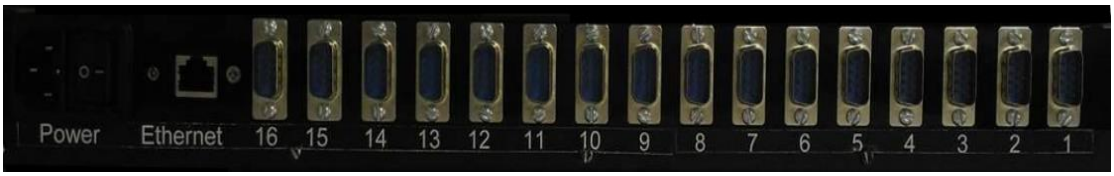
ภาคผนวก

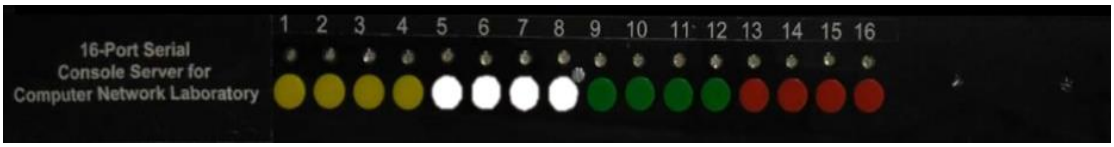
ภาคผนวก ก

ระบบจัดการอุปกรณ์ระยะไกลแบบ 16 การเชื่อมต่อสำหรับห้องปฏิบัติการเครือข่ายคอมพิวเตอร์

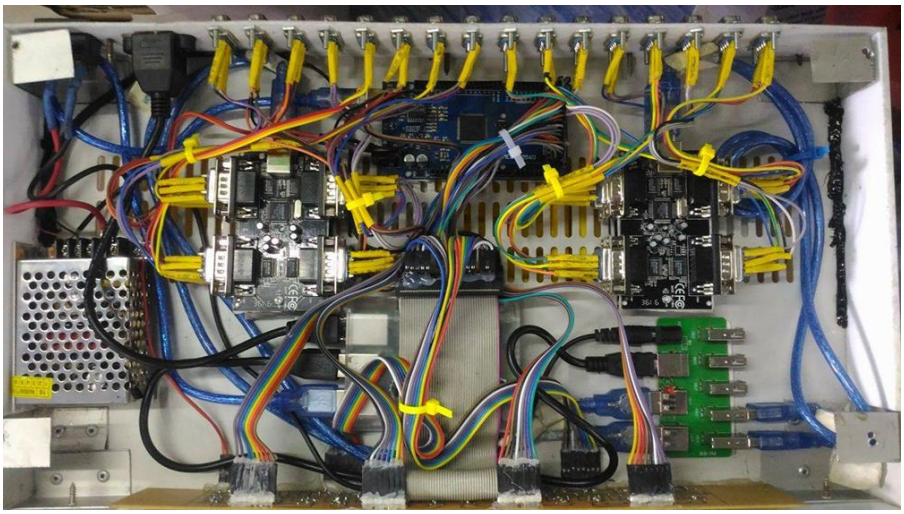
รูปที่ ก.1 เครื่องควบคุมอุปกรณ์ระยะไกลแบบหลายการเชื่อมต่อสำหรับห้องปฏิบัติการเครือข่ายคอมพิวเตอร์



รูปที่ ก.2 ด้านหลังของอุปกรณ์



รูปที่ ก.3 ด้านหน้าของอุปกรณ์



รูปที่ ก.4 การวางอุปกรณ์ต่างๆของฮาร์ดแวร์

ภาคผนวก ข

โปรแกรมจัดการอุปกรณ์ระยะไกลแบบ 16 การเชื่อมต่อสำหรับห้องปฏิบัติการเครือข่ายคอมพิวเตอร์

```
# -*- coding: utf-8 -*-

#!/usr/bin/python

import thread

import time

import RPi.GPIO as GPIO

import commands

import serial

ser=serial.Serial("/dev/ttyS0",9600,timeout=1)

for i in range (1, 17):

s="led"+str(i)+" on"

time.sleep(1)

ser.write(s)

print(s)

GPIO.setmode(GPIO.BCM)

GPIO.setup(2,GPIO.IN, pull_up_down=GPIO.PUD_UP)

GPIO.setup(3,GPIO.IN, pull_up_down=GPIO.PUD_UP)

GPIO.setup(4,GPIO.IN, pull_up_down=GPIO.PUD_UP)

GPIO.setup(17,GPIO.IN, pull_up_down=GPIO.PUD_UP)

GPIO.setup(27,GPIO.IN, pull_up_down=GPIO.PUD_UP)

GPIO.setup(22,GPIO.IN, pull_up_down=GPIO.PUD_UP)

GPIO.setup(10,GPIO.IN, pull_up_down=GPIO.PUD_UP)

GPIO.setup(9,GPIO.IN, pull_up_down=GPIO.PUD_UP)

GPIO.setup(11,GPIO.IN, pull_up_down=GPIO.PUD_UP)

GPIO.setup(5,GPIO.IN, pull_up_down=GPIO.PUD_UP)

GPIO.setup(6,GPIO.IN, pull_up_down=GPIO.PUD_UP)

GPIO.setup(13,GPIO.IN, pull_up_down=GPIO.PUD_UP)
```

```
GPIO.setup(19,GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(26,GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(23,GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(24,GPIO.IN, pull_up_down=GPIO.PUD_UP)
global s1
s1 = '0'
c1 = '0'
def port_1( threadName ):
c1 = '2'
global s1
s1 ='1'
count = 0
while c1 == '2':
ser.write("led1 off")
time.sleep(1)
ser.write("led1 on")
c2,c1 = commands.getstatusoutput('sudo netstat -apn | grep :1001 | wc -l')
print c1
ser.write("led1 on")
 print("หมดเวลาการเชื่อมต่อ")
print commands.getstatusoutput('sudo iptables -D INPUT -p tcp --dport 1001 -j REJECT')
c1 = '0'
s1 ='0'
global s2
s2 = '0'
c2 = '0'
```

```
def port_2( threadName ):
c2 = '2'
global s2
s2 = '1'
count = 0
while c2 == '2':
 ser.write("led2 off")
time.sleep(1)
ser.write("led2 on")
c1,c2 = commands.getstatusoutput('sudo netstat -apn | grep :1002 | wc -l')
print c2
ser.write("led2 on")
print("หมดเวลาการเชื่อมต่อ")
print commands.getstatusoutput('sudo iptables -D OUTPUT -p tcp --sport 1002 -j REJECT')
c2 = '0'
s2 = '0'
global s3
s3 = '0'
c3 = '0'
def port_3( threadName ):
c3 = '2'
global s3
s3 ='1'
count = 0
while c3 == '2':
ser.write("led3 off")
```

```
time.sleep(1)
ser.write("led3 on")
c2,c3 = commands.getstatusoutput('sudo netstat -apn | grep :1003 | wc -l')
print c3
ser.write("led3 on")
print("หมดเวลาการเชื่อมต่อ")
print commands.getstatusoutput('sudo iptables -D OUTPUT -p tcp --sport 1003 -j REJECT')
c3 = '0'
s3 ='0'
global s4
s4 = '0'
c4 = '0'
def port_4( threadName ):
c4 = '2'
global s4
s4 ='1'
count = 0
while c4 == '2':
ser.write("led4 off")
time.sleep(1)
ser.write("led4 on")
c2,c4 = commands.getstatusoutput('sudo netstat -apn | grep :1004 | wc -l')
print c4
ser.write("led4 on")
print("หมดเวลาการเชื่อมต่อ")
print commands.getstatusoutput('sudo iptables -D OUTPUT -p tcp --sport 1004 -j REJECT')
```

```
c4 = '0'
s4 ='0'
global s5
s5 = '0'
c5 = '0'
def port_5( threadName ):
c5 = '2'
global s5
s5 ='1'
count = 0
while c5 == '2':
ser.write("led5 off")
time.sleep(1)
ser.write("led5 on")
c2,c5 = commands.getstatusoutput('sudo netstat -apn | grep :1005 | wc -l')
print c5
ser.write("led5 on")
print("หมดเวลาการเชื่อมต่อ")
print commands.getstatusoutput('sudo iptables -D OUTPUT -p tcp --sport 1005 -j REJECT')
c5 = '0'
s5 ='0'
global s6
s6 = '0'
c6 = '0'
def port_6( threadName ):
c6 = '2'
```

```
global s6
s6 ='1'
count = 0
while c6 == '2':
ser.write("led6 off")
time.sleep(1)
ser.write("led6 on")
c2,c6 = commands.getstatusoutput('sudo netstat -apn | grep :1006 | wc -l')
print c6
ser.write("led6 on")
print("หมดเวลาการเชื่อมต่อ")
print commands.getstatusoutput('sudo iptables -D OUTPUT -p tcp --sport 1006 -j REJECT')
c6 = '0'
s6 ='0'
global s7
s7 = '0'
c7 = '0'
def port_7( threadName ):
c7 = '2'
global s7
s7 ='1'
count = 0
while c7 == '2':
ser.write("led7 off")
time.sleep(1)
ser.write("led7 on")
```

```
c2,c7 = commands.getstatusoutput('sudo netstat -apn | grep :1007 | wc -l')
print c7
ser.write("led7 on")
print("หมดเวลาการเชื่อมต่อ")
print commands.getstatusoutput('sudo iptables -D OUTPUT -p tcp --sport 1007 -j REJECT')
c7 = '0'
s7 ='0'
global s8
s8 = '0'
c8 = '0'
def port_8( threadName ):
c8 = '2'
global s8
s8 ='1'
count = 0
while c8 == '2':
ser.write("led8 off")
time.sleep(1)
ser.write("led8 on")
c2,c8 = commands.getstatusoutput('sudo netstat -apn | grep :1008 | wc -l')
print c8
ser.write("led8 on")
print("หมดเวลาการเชื่อมต่อ")
print commands.getstatusoutput('sudo iptables -D OUTPUT -p tcp --sport 1008 -j REJECT')
c8 = '0'
s8 ='0'
```

```
global s9
s9 = '0'
c9 = '0'
def port_9( threadName ):
c9 = '2'
global s9
s9 ='1'
count = 0
while c9 == '2':
ser.write("led9 off")
time.sleep(1)
ser.write("led9 on")
c2,c9 = commands.getstatusoutput('sudo netstat -apn | grep :1009 | wc -l')
print c9
ser.write("led9 on")
print("หมดเวลาการเชื่อมต่อ")
print commands.getstatusoutput('sudo iptables -D OUTPUT -p tcp --sport 1009 -j REJECT')
c9 = '0'
s9 ='0'
global s10
s10 = '0'
c10 = '0'
def port_10( threadName ):
c10 = '2'
global s10
s10 ='1'
```

```
  count = 0
  while c10 == '2':
  ser.write("led10 off")
  time.sleep(1)
  ser.write("led10 on")
  c2,c10 = commands.getstatusoutput('sudo netstat -apn | grep :1010 | wc -l')
  print c10
   ser.write("led10 on")
  print("หมดเวลาการเชื่อมต่อ")
  print commands.getstatusoutput('sudo iptables -D OUTPUT -p tcp --sport 1010 -j REJECT')
  c10 = '0'
  s10 = '0'
 global s11
 s11 = '0'
c11 = '0'
def port_11( threadName ):
  c10 = '2'
  global s11
  s11 ='1'
  count = 0
  while c11 == '2':
  ser.write("led11 off")
  time.sleep(1)
  ser.write("led11 on")
  c2,c11 = commands.getstatusoutput('sudo netstat -apn | grep :1011 | wc -l')
   print c11
```

```
ser.write("led11 on")
 print("หมดเวลาการเชื่อมต่อ")
 print commands.getstatusoutput('sudo iptables -D OUTPUT -p tcp --sport 1011 -j REJECT')
 c11 = '0'
 s11 = '0'
 global s12
s12 = '0'
c12 = '0'
def port_12( threadName ):
 c12 = '2'
 global s12
 s12 ='1'
 count = 0
 while c12 == '2':
  ser.write("led12 off")
  time.sleep(1)
 ser.write("led12 on")
 c2,c12 = commands.getstatusoutput('sudo netstat -apn | grep :1012 | wc -l')
 print c12
 ser.write("led12 on")
 print("หมดเวลาการเชื่อมต่อ")
 print commands.getstatusoutput('sudo iptables -D OUTPUT -p tcp --sport 1012 -j REJECT')
 c12 = '0'
 s12 = '0'
 global s13
 s13 = '0'
```

```
c13 = '0'
def port_13( threadName ):
c13 = '2'
global s13
s13 ='1'
count = 0
while c13 == '2':
ser.write("led13 off")
 time.sleep(1)
ser.write("led13 on")
 c2,c13 = commands.getstatusoutput('sudo netstat -apn | grep :1013 | wc -l')
print c13
 ser.write("led13 on")
print("หมดเวลาการเชื่อมต่อ")
 print commands.getstatusoutput('sudo iptables -D OUTPUT -p tcp --sport 1013 -j REJECT')
  c13 = '0'
 s13 = '0'
global s14
s14 = '0'
c14 = '0'
def port_14( threadName ):
c14 = '2'
 global s14
s14 ='1'
 count = 0
while c14 == '2':
```

```
    ser.write("led14 off")
    time.sleep(1)
    ser.write("led14 on")
    c2,c14 = commands.getstatusoutput('sudo netstat -apn | grep :1014 | wc -l')
    print c14
    ser.write("led14 on")
   print("หมดเวลาการเชื่อมต่อ")
   print commands.getstatusoutput('sudo iptables -D OUTPUT -p tcp --sport 1014 -j REJECT')
   c14 = '0'
   s14 = '0'
  global s15
  s15 = '0'
c15 = '0'
def port_15( threadName ):
c15 = '2'
global s15
s15 ='1'
count = 0
while c15 == '2':
ser.write("led15 off")
time.sleep(1)
ser.write("led15 on")
c2,c15 = commands.getstatusoutput('sudo netstat -apn | grep :1015 | wc -l')
print c15
 ser.write("led15 on")
print("หมดเวลาการเชื่อมต่อ")
```

```
print commands.getstatusoutput('sudo iptables -D OUTPUT -p tcp --sport 1015 -j REJECT')
c15 = '0'
s15 = '0'
global s16
s16 = '0'
c16 = '0'
def port_16( threadName ):
c16 = '2'
global s16
s16 ='1'
count = 0
while c16 == '2':
    ser.write("led16 off")
    time.sleep(1)
    ser.write("led16 on")
    c2,c16 = commands.getstatusoutput('sudo netstat -apn | grep :1016 | wc -l')
    print c16
    ser.write("led16 on")
print("หมดเวลาการเชื่อมต่อ")
print commands.getstatusoutput('sudo iptables -D OUTPUT -p tcp --sport 1016 -j REJECT')
c16 = '0'
s16 = '0'
def checkled():
a = open('file','w')
for j in range (1001, 1017):
a = open("/home/pi/"+str(j)+".txt",'w')
```

```
s="sudo netstat -apn | grep :"+str(j)+" | wc -l"
time.sleep(.01)
c2,c99 = commands.getstatusoutput(s)
a.write(str(c99)+"\n")
if c99 == '2' :
stid="led"+str(j-1000)+" on"
ser.write(stid)
stid="led"+str(j-984)+" off"
ser.write(stid)
print(stid)
else :
sdep="led"+str(j-1000)+" off"
ser.write(sdep)
stid="led"+str(j-984)+" on"
ser.write(stid)
#a.close()
#else
#print(s)
def check():
state = GPIO.input(2)
f1 = open('/home/pi/1.txt')
for l1 in f1:
b1=l1
print (b1)
if state == False or b1 == '1' :
if s1 == '0' :
```

```
print s1
print commands.getstatusoutput('sudo iptables -I INPUT -p tcp --dport 1001 -j REJECT')
 try:
thread.start_new_thread( port_1, ("Thread-1",) )
except:
 print "Error: unable to start thread 1"
 a = open('/home/pi/1.txt','w')
 a.write('0')
 a.close()
 state = GPIO.input(3)
  f1 = open('/home/pi/2.txt')
   for l1 in f1:
   b1=l1
   print (b1)
   if state == False or b1 == '1':
   if s2 == '0' :
    print s2
   print commands.getstatusoutput('sudo iptables -I OUTPUT -p tcp --sport 1002 -j REJECT')
    try:
     thread.start_new_thread( port_2, ("Thread-2",) )
     except:
     print "Error: unable to start thread 2"
     a = open('/home/pi/2.txt','w')
      a.write('0')
      a.close()
     state = GPIO.input(4)
```

```
 f1 = open('/home/pi/3.txt')
 for l1 in f1:
 b1=l1
 print (b1)
 if state == False or b1 == '1':
 if s3 == '0' :
 print s3
print commands.getstatusoutput('sudo iptables -I OUTPUT -p tcp --sport 1003 -j REJECT')
try:
 thread.start_new_thread( port_3, ("Thread-3",) )
 except:
 print "Error: unable to start thread 3"
 a = open('/home/pi/3.txt','w')
 a.write('0')
 a.close()
 state = GPIO.input(17)
 f1 = open('/home/pi/4.txt')
 for l1 in f1:
 b1=l1
 print (b1)
 if state == False or b1 == '1':
 if s4 == '0' :
print s4
 print commands.getstatusoutput('sudo iptables -I OUTPUT -p tcp --sport 1004 -j REJECT')
 try:
thread.start_new_thread( port_4, ("Thread-4",) )
```

```
except:
print "Error: unable to start thread 4"
a = open('/home/pi/4.txt','w')
a.write('0')
 a.close()
state = GPIO.input(27)
f1 = open('/home/pi/5.txt')
for l1 in f1:
b1=l1
print (b1)
if state == False or b1 == '1':
if s5 == '0' :
print s5
print commands.getstatusoutput('sudo iptables -I OUTPUT -p tcp --sport 1005 -j REJECT')
try:
thread.start_new_thread( port_5, ("Thread-5",) )
except:
print "Error: unable to start thread 5"
a = open('/home/pi/5.txt','w')
a.write('0')
a.close()
state = GPIO.input(22)
f1 = open('/home/pi/6.txt')
for l1 in f1:
b1=l1
print (b1)
```

```
if state == False or b1 == '1':
if s6 == '0' :
print s6
print commands.getstatusoutput('sudo iptables -I OUTPUT -p tcp --sport 1006 -j REJECT')
try:
thread.start_new_thread( port_6, ("Thread-6",) )
except:
print "Error: unable to start thread 6"
a = open('/home/pi/6.txt','w')
a.write('0')
a.close()
state = GPIO.input(10)
f1 = open('/home/pi/7.txt')
for l1 in f1:
 b1=l1
 print (b1)
 if state == False or b1 == '1':
 if s7 == '0' :
  print s7
  print commands.getstatusoutput('sudo iptables -I OUTPUT -p tcp --sport 1007 -j REJECT')
  try:
  thread.start_new_thread( port_7, ("Thread-7",) )
  except:
  print "Error: unable to start thread 7"
  a = open('/home/pi/7.txt','w')
  a.write('0')
```

```
a.close()
state = GPIO.input(9)
f1 = open('/home/pi/8.txt')
for l1 in f1:
 b1=l1
print (b1)
if state == False or b1 == '1':
if s8 == '0' :
print s8
print commands.getstatusoutput('sudo iptables -I OUTPUT -p tcp --sport 1008 -j REJECT')
try:
thread.start_new_thread( port_8, ("Thread-8",) )
except:
print "Error: unable to start thread 8"
a = open('/home/pi/8.txt','w')
a.write('0')
a.close()
state = GPIO.input(11)
f1 = open('/home/pi/9.txt')
for l1 in f1:
b1=l1
print (b1)
 if state == False or b1 == '1':
if s9 == '0' :
print s9
print commands.getstatusoutput('sudo iptables -I OUTPUT -p tcp --sport 1009 -j REJECT')
```

```
 try:
thread.start_new_thread( port_9, ("Thread-9",) )
except:
print "Error: unable to start thread 9"
a = open('/home/pi/9.txt','w')
 a.write('0')
a.close()
 state = GPIO.input(5)
 f1 = open('/home/pi/10.txt')
  for l1 in f1:
b1=l1
  print (b1)
   if state == False or b1 == '1':
  if s10 == '0' :
   print s10
   print commands.getstatusoutput('sudo iptables -I OUTPUT -p tcp --sport 1010 -j REJECT')
   try:
   thread.start_new_thread( port_10, ("Thread-10",) )
    except:
    print "Error: unable to start thread 10"
   a = open('/home/pi/10.txt','w')
    a.write('0')
    a.close()
   state = GPIO.input(6)
   f1 = open('/home/pi/11.txt')
   for l1 in f1:
```

```
 b1=l1
  print (b1)
 if state == False or b1 == '1':
if s11 == '0' :
 print s11
print commands.getstatusoutput('sudo iptables -I OUTPUT -p tcp --sport 1011 -j REJECT')
try:
thread.start_new_thread( port_11, ("Thread-11",) )
except:
 print "Error: unable to start thread 11"
 a = open('/home/pi/11.txt','w')
 a.write('0')
 a.close()
 state = GPIO.input(13)
 f1 = open('/home/pi/12.txt')
 for l1 in f1:
 b1=l1
print (b1)
 if state == False or b1 == '1':
if s12 == '0' :
  print s12
 print commands.getstatusoutput('sudo iptables -I OUTPUT -p tcp --sport 1012 -j REJECT')
 try:
 thread.start_new_thread( port_12, ("Thread-12",) )
  except:
 print "Error: unable to start thread 12"
```

```
a = open('/home/pi/12.txt','w')
a.write('0')
a.close()
state = GPIO.input(19)
f1 = open('/home/pi/13.txt')
for l1 in f1:
b1=l1
print (b1)
if state == False or b1 == '1':
if s13 == '0' :
 print s13
print commands.getstatusoutput('sudo iptables -I OUTPUT -p tcp --sport 1013 -j REJECT')
try:
thread.start_new_thread( port_13, ("Thread-13",) )
except:
print "Error: unable to start thread 13"
a = open('/home/pi/13.txt','w')
 a.write('0')
a.close()
 state = GPIO.input(26)
f1 = open('/home/pi/14.txt')
for l1 in f1:
b1=l1
print (b1)
if state == False or b1 == '1':
if s14 == '0' :
```

```
print s14
print commands.getstatusoutput('sudo iptables -I OUTPUT -p tcp --sport 1014 -j REJECT')
try:
thread.start_new_thread( port_14, ("Thread-14",) )
except:
print "Error: unable to start thread 14"
a = open('/home/pi/14.txt','w')
a.write('0')
 a.close()
  state = GPIO.input(23)
 f1 = open('/home/pi/15.txt')
  for l1 in f1:
  b1=l1
  print (b1)
  if state == False or b1 == '1':
 if s15 == '0' :
 print s15
 print commands.getstatusoutput('sudo iptables -I OUTPUT -p tcp --sport 1015 -j REJECT')
  try:
  thread.start_new_thread( port_15, ("Thread-15",) )
  except:
  print "Error: unable to start thread 15"
  a = open('/home/pi/15.txt','w')
   a.write('0')
   a.close()
  state = GPIO.input(24)
```

```
f1 = open('/home/pi/16.txt')
for l1 in f1:
b1=l1
print (b1)
if state == False or b1 == '1':
if s16 == '0' :
print s16
print commands.getstatusoutput('sudo iptables -I OUTPUT -p tcp --sport 1016 -j REJECT')
try:
thread.start_new_thread( port_16, ("Thread-16",) )
except:
print "Error: unable to start thread 16"
 a = open('/home/pi/16.txt','w')
a.write('0')
 a.close()
 while 1:
 pass
check()
checkled()
```

ภาคผนวก ค

โปรแกรมควบคุมสถานะการเชื่อมต่อโดยแสดงผ่าน หลอดแอลอีดี (LED)

```
char inData[100]; // Allocate some space for the string
char inChar=-1; // Where to store the character read
byte index = 0; // Index into array; where to store the character
intledPin[]={22,24,26,28,30,32,34,36,38,40,42,44,46,48,50,52,23,25,27,29,31,33,35,37,39,41,43,
45,47,49,51,53};
void setup() {
Serial.begin(9600);
Serial.write("Power On");
for (int i=0;i<34;i++)
{
pinMode(ledPin[i], OUTPUT);
}
}
char Comp(char* This) {
while (Serial.available() > 0) // Don't read unless
{
if(index < 34) // One less than the size of the array
{
inChar = Serial.read(); // Read a character
inData[index] = inChar; // Store it
index++; // Increment where to write next
inData[index] = '\0'; // Null terminate the string
}
}
if (strcmp(inData,This)  == 0) {
for (int i=0;i<34;i++) {
```

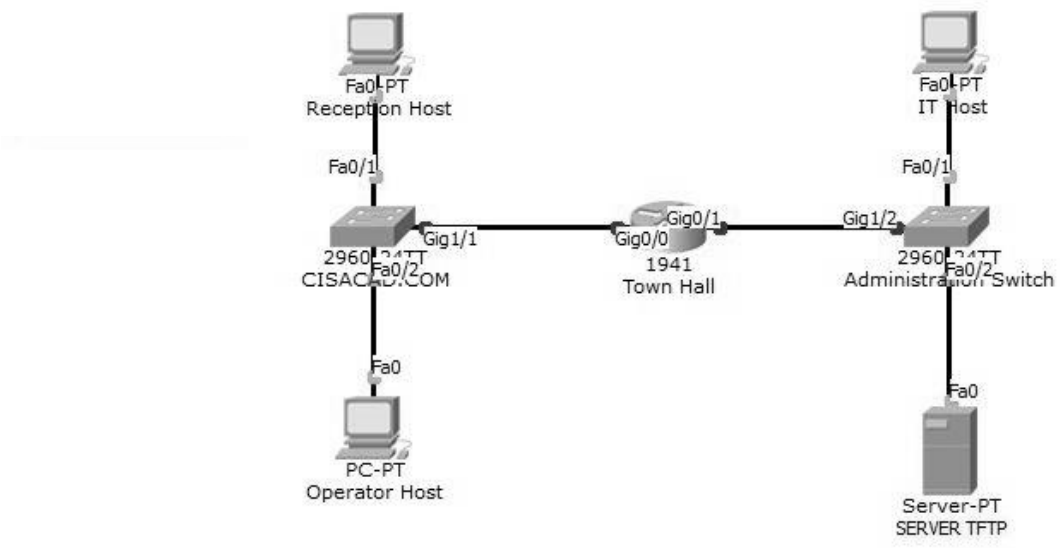
```
inData[i]=0;
}
index=0;
return(0);
}
else {
return(1);
}
}
void loop()
{
   if (Comp("led1 on")==0)     digitalWrite(ledPin[0], HIGH);
   if (Comp("led1 off")==0)    digitalWrite(ledPin[0], LOW);
   if (Comp("led2 on")==0)     digitalWrite(ledPin[1], HIGH);
   if (Comp("led2 off")==0)    digitalWrite(ledPin[1], LOW);
   if (Comp("led3 on")==0)     digitalWrite(ledPin[2], HIGH);
   if (Comp("led3 off")==0)    digitalWrite(ledPin[2], LOW);
   if (Comp("led4 on")==0)     digitalWrite(ledPin[3], HIGH);
   if (Comp("led4 off")==0)    digitalWrite(ledPin[3], LOW);
   if (Comp("led5 on")==0)     digitalWrite(ledPin[4], HIGH);
   if (Comp("led5 off")==0)    digitalWrite(ledPin[4], LOW);
   if (Comp("led6 on")==0)     digitalWrite(ledPin[5], HIGH);
   if (Comp("led6 off")==0)    digitalWrite(ledPin[5], LOW);
   if (Comp("led7 on")==0)     digitalWrite(ledPin[6], HIGH);
   if (Comp("led7 off")==0)    digitalWrite(ledPin[6], LOW);
   if (Comp("led8 on")==0)     digitalWrite(ledPin[7], HIGH);
```

```
if (Comp("led8 off")==0)    digitalWrite(ledPin[7], LOW);
if (Comp("led9 on")==0)     digitalWrite(ledPin[8], HIGH);
if (Comp("led9 off")==0)    digitalWrite(ledPin[8], LOW);
if (Comp("led10 on")==0)     digitalWrite(ledPin[9], HIGH);
if (Comp("led10 off")==0)    digitalWrite(ledPin[9], LOW);
if (Comp("led11 on")==0)     digitalWrite(ledPin[10], HIGH);
if (Comp("led11 off")==0)    digitalWrite(ledPin[10], LOW);
if (Comp("led12 on")==0)     digitalWrite(ledPin[11], HIGH);
if (Comp("led12 off")==0)    digitalWrite(ledPin[11], LOW);
if (Comp("led13 on")==0)     digitalWrite(ledPin[12], HIGH);
if (Comp("led13 off")==0)    digitalWrite(ledPin[12], LOW);
if (Comp("led14 on")==0)     digitalWrite(ledPin[13], HIGH);
if (Comp("led14 off")==0)    digitalWrite(ledPin[13], LOW);
if (Comp("led15 on")==0)     digitalWrite(ledPin[14], HIGH);
if (Comp("led15 off")==0)    digitalWrite(ledPin[14], LOW);
if (Comp("led16 on")==0)     digitalWrite(ledPin[15], HIGH);
if (Comp("led16 off")==0)    digitalWrite(ledPin[15], LOW);
if (Comp("led17 on")==0)     digitalWrite(ledPin[16], HIGH);
if (Comp("led17 off")==0)    digitalWrite(ledPin[16], LOW);
if (Comp("led18 on")==0)     digitalWrite(ledPin[17], HIGH);
if (Comp("led18 off")==0)    digitalWrite(ledPin[17], LOW);
if (Comp("led19 on")==0)     digitalWrite(ledPin[18], HIGH);
if (Comp("led19 off")==0)    digitalWrite(ledPin[18], LOW);
if (Comp("led20 on")==0)     digitalWrite(ledPin[19], HIGH);
if (Comp("led20 off")==0)    digitalWrite(ledPin[19], LOW);
if (Comp("led21 on")==0)     digitalWrite(ledPin[20], HIGH);
```

```
    if (Comp("led21 off")==0)    digitalWrite(ledPin[20], LOW);
    if (Comp("led22 on")==0)     digitalWrite(ledPin[21], HIGH);
    if (Comp("led22 off")==0)    digitalWrite(ledPin[21], LOW);
    if (Comp("led23 on")==0)     digitalWrite(ledPin[22], HIGH);
    if (Comp("led23 off")==0)    digitalWrite(ledPin[22], LOW);
    if (Comp("led24 on")==0)     digitalWrite(ledPin[23], HIGH);
    if (Comp("led24 off")==0)    digitalWrite(ledPin[23], LOW);
    if (Comp("led25 on")==0)     digitalWrite(ledPin[24], HIGH);
    if (Comp("led25 off")==0)    digitalWrite(ledPin[24], LOW);
    if (Comp("led26 on")==0)     digitalWrite(ledPin[25], HIGH);
    if (Comp("led26 off")==0)    digitalWrite(ledPin[25], LOW);
    if (Comp("led27 on")==0)     digitalWrite(ledPin[26], HIGH);
    if (Comp("led27 off")==0)    digitalWrite(ledPin[26], LOW);
    if (Comp("led28 on")==0)     digitalWrite(ledPin[27], HIGH);
    if (Comp("led28 off")==0)    digitalWrite(ledPin[27], LOW);
    if (Comp("led29 on")==0)     digitalWrite(ledPin[28], HIGH);
    if (Comp("led29 off")==0)    digitalWrite(ledPin[28], LOW);
    if (Comp("led30 on")==0)     digitalWrite(ledPin[29], HIGH);
    if (Comp("led30 off")==0)    digitalWrite(ledPin[29], LOW);
    if (Comp("led31 on")==0)     digitalWrite(ledPin[30], HIGH);
    if (Comp("led31 off")==0)    digitalWrite(ledPin[30], LOW);
    if (Comp("led32 on")==0)     digitalWrite(ledPin[31], HIGH);
    if (Comp("led32 off")==0)    digitalWrite(ledPin[31], LOW);
}
```

ภาคผนวก ง

แบบทดสอบการปฏิบัติ CCNA 1 Final Exam Cisco



รูปที่ ง.1 แผนภาพเน็ตเวิร์คของการปฏิบัติการเครือข่ายคอมพิวเตอร์ CCNA1

Step 1:

Design an IPv4 addressing scheme and complete the Addressing Table based on the following requirements. Use the table above to help you organize your work.

a. Subnet the 192.168.1.0/24 network to provide 30 host addresses per subnet while wasting the fewest addresses.

b. Assign the fourth subnet to the IT Department LAN.

c. Assign the last network host address (the highest) in this subnet to the G0/0 interface on Town Hall.

d. Starting with the fifth subnet, subnet the network again so that the new subnets will provide 14 host addresses per subnet while wasting the fewest addresses.

e. Assign the second of these new 14-host subnets to the Administration LAN.

f. Assign the last network host address (the highest) in the Administration LAN subnet to the G0/1 interface of the Town Hall router.

g. Assign the second to the last address (the second highest) in this subnet to the VLAN 1 interface of the Administration Switch.

h. Configure addresses on the hosts using any of the remaining addresses in their respective subnets.

Step 2: Configure the Town Hall Router.

a. Configure the Town Hall router with all initial configurations that you have learned in the course so far:

• Configure the router hostname: Middle

• Protect device configurations from unauthorized access with the encrypted password.

• Secure all of the ways to access the router using methods covered in the course and labs.

• Newly-entered passwords must have a minimum length of 10 characters.

• Prevent all passwords from being viewed in clear text in device configuration files.

• Configure the router to only accept in-band management connections over the protocol that is more secure than Telnet, as was done in the labs. Use the value 1024 for encryption key strength.

• Configure user authentication for in-band management connections.

b. Configure the two Gigabit Ethernet interfaces using the IPv4 addressing values you calculated and the IPv6 values provided in the addressing table.

• Reconfigure the link local addresses as was practiced in the labs. The IPv6 link-local Interface ID should match the IPv6 unicast Interface ID as is practiced in the labs.

• Document the interfaces in the configuration file.

Step 3: Configure the Administration Switch.

Configure Administration Switch for remote management.

Step 4: Configure and Verify Host Addressing.

a. Use the IPv4 addressing from Step 1 and the IPv6 addressing values provided in the addressing table to configure all host PCs with the correct addressing.

b. Use the router interface link-local addresses as the IPv6 default gateways on the hosts.

c. All hosts should be able to ping each other over IPv4.

Step 5: Backup the Configuration of the Town Hall Router to TFTP.

a. Complete the configuration of the TFTP server using the IPv4 addressing values from Step 1 and the values in the addressing table.

b. Backup the running configuration of Town Hall to the TFTP Server. Use the default file name.

Answer

Router>

Router>enable

Router#configure terminal

Router(config)#interface g0/0

Router(config-if)#ip address 192.168.1.126 255.255.255.224

Router(config-if)#description IT Department LAN

Router(config-if)#no shutdown

Router(config-if)#exit

Router(config)#interface g0/1

Router(config-if)#ip address 192.168.1.158 255.255.255.240

Router(config-if)#description Administration LAN

Router(config-if)#no shutdown

Router(config-if)#exit

Router(config)#ipv6 unicast-routing

Router(config)#interface g0/0

Router(config-if)#ipv6 address 2001:db8:acad:A::1/64

Router(config-if)#ipv6 address FE80::1 link-local

Router(config-if)#no shutdown

Router(config-if)#exit

Router(config)#interface g0/1

```
Router(config-if)#ipv6 address 2001:db8:acad:B::1/64
Router(config-if)#ipv6 address FE80::1 link-local
Router(config-if)#no shutdown
Router(config-if)#exit
Router(config)#
Router(config)#hostname Middle
Middle(config)#Enable secret class12345
Middle(config)#line console 0
Middle(config-line)#password ccna5.net2014
Middle(config-line)#login
Middle(config-line)#exit
Middle(config)#line vty 0 15
Middle(config-line)#password ccna5.net2014
Middle(config-line)#login
Middle(config-line)#exit
Middle(config)#line aux 0
Middle(config-line)#password ccna5.net2014
Middle(config-line)#login
Middle(config-line)#exit
Middle(config)#
Middle(config)#Banner motd “Authorized Access Only”
Middle(config)#security password min-length 10
Middle(config)#service password-encryption
Middle(config)#ip domain-name ccna5.net
Middle(config)#username cisco secret ccna5.net2014
Middle(config)#crypto key generate rsa
```

The name for the keys will be: Middle.cisco.local

Choose the size of the key modulus in the range of 360 to 2048 for your

General Purpose Keys. Choosing a key modulus greater than 512 may take

a few minutes.

How many bits in the modulus [512]: 1024

% Generating 1024 bit RSA keys, keys will be non-exportable…[OK]

Middle(config)#line vty 0 15

Middle(config-line)#login local

Middle(config-line)#transport input ssh

Middle(config-line)#exit

Middle(config)#

Middle# copy running-config startup-config

Administration Switch

ip default-gateway 192.168.1.158

Reception Host

IPv4

IP address: 192.168.1.97

Mask: 255.255.255.224

default gateway: 192.168.1.126

IPv6

IPv6 address: 2001:DB8:ACAD:A::FF/64

default gateway: FE80::1

Operator Host

IPv4

IP address: 192.168.1.98

Mask: 255.255.255.224

default gateway: 192.168.1.126

IPv6

IPv6 address: 2001:DB8:ACAD:A::15/64

default gateway: FE80::1

_____

IT Host

IPv4

IP address: 192.168.1.145

Mask: 255.255.255.240

default gateway: 192.168.1.158

IPv6

IPv6 address: 2001:DB8:ACAD:B::FF/64

default gateway: FE80::1

_____

SERVER TFTP

IPv4

IP address: 192.168.1.146

Mask: 255.255.255.240

default gateway: 192.168.1.158

IPv6

IPv6 address: 2001:DB8:ACAD:B::15/64

default gateway: FE80::1

_____

Backup the Configuration of the Town Hall Router to TFTP.

Middle#copy running-config tftp

Address or name of remote host []? 192.168.1.146

Destination filename [Router-confg]? [Press Enter]

ภาคผนวก จ

แบบประเมินความพึงพอใจในการใช้งานระบบควบคุมอุปกรณ์ระยะไกลแบบหลายการเชื่อมต่อสำหรับห้องปฏิบัติการเครือข่ายคอมพิวเตอร์

แบบประเมินผลวิจัย

เรื่อง ความพึงพอใจในการใช้งานระบบควบคุมอุปกรณ์ระยะไกลแบบหลายการเชื่อมต่อสำหรับห้องปฏิบัติการเครือข่ายคอมพิวเตอร์

คำชี้แจง แบบประเมินนี้ใช้เพื่อประเมินผลการปฏิบัติการเครือข่ายคอมพิวเตอร์โดยใช้หลักสูตร Cisco Academy โดยแบ่งเนื้อหาการประเมิน 3 ส่วน คือ

1.รายละเอียดของผู้ที่ทำการประเมิน

2.ประเมินความพึงพอใจจากการปฏิบัติการเครือข่ายคอมพิวเตอร์ใช้โดยเครื่องควบคุมอุปกรณ์ระยะไกลแบบหลาย

การเชื่อมต่อสำหรับห้องปฏิบัติการเครือข่ายคอมพิวเตอร์

3.ข้อเสนอแนะทั่วไป

ส่วนที่ 1 รายละเอียดผู้ทำการประเมิน

วิธีการประเมิน ให้ผู้ประเมินทำเครื่องเครื่องหมาย ✓ ในช่อง ☐ ตามความเป็นจริง

1.เพศ ☐ ชาย ☐ หญิง

2.สถานภาพผู้ตอบแบบประเมิน ☐นักศึกษา ☐ อาจารย์ ☐ อื่นๆ โปรดระบุ.....

3.ระดับการศึกษา ☐ ป.ตรี ☐ ป.โท ☐ ป.เอก ☐ อื่นๆ โปรดระบุ.....

4.ระดับชั้นปีการศึกษา ☐ปี 1 ☐ปี 2 ☐ ปี 3 ☐ปี 4 ☐อื่นๆ โปรดระบุ.....

5.คณะวิชาที่ศึกษา ☐ สาขาวิชาวิทยาการคอมพิวเตอร์☐สาขาวิชาวิศวกรรมคอมพิวเตอร์

☐ สาชาวิชาเทคโนโลยีสารสนเทศ☐ สาขาวิชาคอมพิวเตอร์ธุรกิจ

☐ อื่นๆ โปรดระบุ.....

6.ระดับการศึกษาหลักสูตร

☐ CCNA1 ☐ CCNA2 ☐ CCNA3 ☐ CCNA4

☐ CCNA Security ☐ อื่นๆ โปรดระบุ.....

2.ประเมินความพึงพอใจจากการปฏิบัติการเครือข่ายคอมพิวเตอร์ใช้โดยเครื่องควบคุมอุปกรณ์ระยะไกลแบบหลายการเชื่อมต่อสำหรับห้องปฏิบัติการเครือข่ายคอมพิวเตอร์

วิธีการประเมิน เขียนเครื่องหมาย ✓ ลงในช่องระดับความพึงพอใจตามความเป็นจริงโดยแบ่งระดับการประเมินออกเป็น 5 ระดับความพึงพอใจ

ระดับคะแนน 5 หมายถึง พึงพอใจมากที่สุด

ระดับคะแนน 4 หมายถึง พึงพอใจมาก

ระดับคะแนน 3 หมายถึง พึงพอใจปานกลาง

ระดับคะแนน 2 หมายถึง พึงพอใจน้อย

ระดับคะแนน 1 หมายถึง พึงพอใจน้อยที่สุด

ข้อ	หัวข้อ	ระดับความพึงพอใจ				
		1	2	3	4	5
1.	ความสะดวกของขั้นตอนใน**“การจัดเตรียมอุปกรณ์ก่อน”**ปฏิบัติการเครือข่ายคอมพิวเตอร์โดยเครื่องควบคุมอุปกรณ์ระยะไกลแบบหลายการเชื่อมต่อสำหรับห้องปฏิบัติการเครือข่ายคอมพิวเตอร์					
2.	ความสะดวกใน**“การใช้งานอุปกรณ์ระหว่าง”**ปฏิบัติการเครือข่ายคอมพิวเตอร์โดยเครื่องควบคุมอุปกรณ์ระยะไกลแบบหลายการเชื่อมต่อสำหรับห้องปฏิบัติการเครือข่ายคอมพิวเตอร์					
3.	ความสะดวกของขั้นตอนใน**“การจัดเก็บอุปกรณ์หลัง”**ปฏิบัติการเครือข่ายคอมพิวเตอร์โดยเครื่องควบคุมอุปกรณ์ระยะไกลแบบหลายการเชื่อมต่อสำหรับห้องปฏิบัติการเครือข่ายคอมพิวเตอร์					
4.	**“ความถูกต้อง”**ในการทำงานของเครื่องควบคุมอุปกรณ์ระยะไกลแบบหลายการเชื่อมต่อสำหรับห้องปฏิบัติการเครือข่ายคอมพิวเตอร์					

5.	**“ความเหมาะสม”**ในการทำงานของเครื่องควบคุมอุปกรณ์ระยะไกลแบบหลายการเชื่อมต่อสำหรับห้องปฏิบัติการเครือข่ายคอมพิวเตอร์					
6.	**“ความสวยงามและทันสมัย”**ในการทำงานของเครื่องควบคุมอุปกรณ์ระยะไกลแบบหลายการเชื่อมต่อสำหรับห้องปฏิบัติการเครือข่ายคอมพิวเตอร์					
7.	สนับสนุนการเรียนรู้การปฏิบัติการเครือข่ายคอมพิวเตอร์**“แบบบุคคล”**โดยเครื่องควบคุมอุปกรณ์ระยะไกลแบบหลายการเชื่อมต่อสำหรับห้องปฏิบัติการเครือข่ายคอมพิวเตอร์					
8.	สนับสนุนการเรียนรู้การปฏิบัติการเครือข่ายคอมพิวเตอร์**“แบบกลุ่ม”**โดยเครื่องควบคุมอุปกรณ์ระยะไกลแบบหลายการเชื่อมต่อสำหรับห้องปฏิบัติการเครือข่ายคอมพิวเตอร์					

ส่วนที่ 3 ข้อเสนอแนะทั่วไป

ข้อคิดเห็น/ข้อเสนอแนะเพิ่มเติม

.....

.....

.....

.....

.....

.....

.....