博　士　論　文

物理的表現手法による技術・情報分野の教材開発およびその評価に関する研究

2020 年 3 月

広島大学大学院教育学研究科

吉原 和明

目次

第 1 章

序章

1.1　研究の背景

今日，情報通信ネットワークは我々になくてはならないインフラとなっている。情報通信ネットワークであるインターネットの普及率は高く，総務省によると我が国におけるインターネットの人口普及率は，2013 年以降 80 %を超えた水準を保っている [1]。また，内閣府が提唱した，我が国が目指すべき未来社会の姿として提唱した Society5.0 では，IoT(Internet of Things) であらゆる人とモノが繋がり，様々な情報が共有され，新しい価値を生み出すことにより社会的な課題を解決していくとされている [2]。Society5.0 における社会の基盤として，情報通信ネットワーク技術の発展が不可欠である。

文部科学省が発表した「Society5.0 に向けた人材育成～社会が変わる，学びが変わる～」では，アメリカや中国などに比べ，情報科学などに関する研究開発と教育が立ち遅れており，Society5.0 に向かう社会において，我が国では圧倒的に人材不足になりつつあると指摘されている [3]。現実に，平成 28 年の経済産業省による ICT 人材の最新動向と将来推計に関する調査結果では，ICT 人材は 2019 年をピークに産業人口は減少してゆき，不足規模はますます深刻になるという推計結果が出ている [4]。特に，ビッグデータや IoT，人工知能などの先端 ICT 技術と情報セキュリティの分野で人材の不足が見込まれているが，これらの分野の基盤には，情報通信ネットワークが必

要不可欠であり，情報通信ネットワークの知識を持った ICT 人材の育成が急務であると言える。

情報通信ネットワークの学習は，中学校では，技術・家庭科技術分野（以下，技術科）における「D 情報に関する技術」において，「情報通信ネットワークの構成と，情報を利用するための基本的な仕組みを理解すること」となっており，また，高等学校では，情報において「情報通信ネットワークの仕組みや構成要素，プロトコルの役割及び情報セキュリティを確保するための方法や技術について理解すること」となっている。このように，中学校や高等学校においては，情報通信ネットワークの学習は必修化されている [5][6]。これらの学習には「実践的・体験的な学習活動を通して生活に基礎的・基本的な知識及び技術を身につけさせる」と示されており，情報通信ネットワークの学習においても実践的・体験的な学習を行うことが求められている。

情報通信ネットワークの実践的・体験的な学習として，情報系の大学を中心に，ネットワーク構築の演習を行っているところも多い。多くの場合は，端末やルータなどの実際に情報通信ネットワークを構成するコンピュータ機器同士を配線してネットワークを構築する演習となっている。しかし，このようなルータの設定は中学生や高校生にとっては敷居が高いと考えられる。中学校や高等学校で利用できる情報通信ネットワークを学ぶための良い教材が皆無に等しいことが，情報通信ネットワークの学習で実験や実習を行うことを難しくしていると考えられる。

一方で，情報通信ネットワークは目に見えないところで働いており，コンピュータにおける情報通信ネットワークの設定は，通常キーボードを用いて行うことが多く，データ通信の結果は，コンピュータの画面越しでしか確認ができない。それゆえ，中学生や高校生などの初学者にとっては，情報通信ネットワークの学習は敷居が高く，理解しづらいものになっている。

1.2　本研究の概要

本研究では，情報通信ネットワークを直感的に学習するための新たな教材を開発し，教材の有効性の検証を行った。具体的には，情報通信ネットワークを学習するための教材として，(1) IP アドレスを学習するための教材の開発とその評価，(2) 情報通信ネットワークの構築を学習するためのルータ教材の開発を行った。

(1) においては，ネットワークの動作を LED の点灯により，物理的に可視化し，ボタンやダイヤルで物理的に直接操作することで，実際にネットワークの構築を通して IP アドレスの仕組みを学習するための教材を開発した。中学校 1 年生を対象に教材を用いた授業実践を行い，教材の評価を行った。授業対象者を実験群と統制群に分け，事前テスト・事後テストを行った結果，事後テストでは，実験群の方が統制群より得点が高い傾向にあり，有意差が見られたことから，教材の有効性が確認できた [7]。

(2) においては，情報通信ネットワークの仕組みを理解するためには，IP アドレスだけではなく様々な情報通信技術を理解する必要がある。IP アドレス学習教材は，2 台の端末間のみでのネットワークを扱う教材であり，複数のネットワーク間で通信を行うインターネットでは，どのように端末同士が通信しているかを学習することはできない。より現実に即したネットワークの仕組みを理解するには、複数のネットワーク間を通信するのに必要なルータの役割を学習する必要がある。

本研究では，(1) で開発した物理的可視化と物理的直接操作を取り入れた IP アドレス学習教材を拡張し，複数のネットワーク間の通信の仕組みを学習する教材としてルータ教材の開発を行った。そして，ルータ教材として有効に機能するかどうか検証実験を行った [8]。

1.3　論文の構成

本章以降の章では，本研究の具体的な内容について述べる。第 2 章では，我々が提唱する物理的可視化と物理的直接操作について述べる。第 3 章では，IP アドレス学習教材の開発とその評価について述べる。第 4 章では，ルータ教材の開発について述べる。第 5 章では，本研究の位置づけについて述べる。最後に第 6 章では本論文のまとめを行い，今後の展望について述べる。

第 2 章

物理的可視化と物理的直接操作

中学校技術科や高等学校情報科では，学習内容に生活や社会を支える現在の工業製品や情報システムに関する技術が含まれている。これらの工業製品や情報システムの多くはブラックボックスになっており，内部の動作原理や構造が見えなくなっている。それゆえ，技術そのものや，技術がどのように工業製品や情報システムに活用されているのかを理解するのが難しくなっている。また，技術を活用してものづくりを行う際，複雑な操作を要するものが多く，より学習を難しくしている。

中学生や高校生にとって理解しづらい目に見えない技術や，ものづくりに必要な複雑な操作をいかに容易にできるような教材を提供できるかが重要であると考える。そこで我々は，ものづくり教材の開発を行う際に重要なコンセプトとして，物理的可視化と物理的直接操作を提唱する。

一般的な可視化 (Visualization) は，人間が直接見ることのできない事象をディスプレイなどを用いて可視化することである。それに対し，物理的可視化とは，目に見えない事象を物理的な現象を用いて可視化することである。物理的な現象とは，例えば，目で直感的に理解できる LED のような光や，扇風機の羽のような回転する動き，または耳で直感的に理解できるブザーの音

などであり，物理的可視化は目に見えない現象を風や光のような直感的に理解しやすい現象で，1 対 1 で対応させて表現することである。図 2.1 は物理的可視化を説明したものである。

一般的な直接操作 (Direct Manipulation) は，コマンドなどを入力することで間接的に操作するのではなく，画面上のオブジェクトをマウスなどの装置を用いて直接働きながら操作することである。物理的直接操作は，複雑な操作を，物理的なものを用いて直感的に操作することである。直感的な操作とは，例えば，手でボタンを押したりダイヤルをひねったり，ペダルを足で踏んだり，また口で吹いたりする動作であり，物理的直接操作は複雑な操作を手や足を直感的に動かす操作によって 1 対 1 で対応させることである。図 2.2 は物理的直接操作を説明したものである。

物理的可視化により，理解しづらい目に見えない技術を直感的に理解することができ，さらに，物理的直接操作により，複雑な操作を直感的かつ容易に行うことができる。

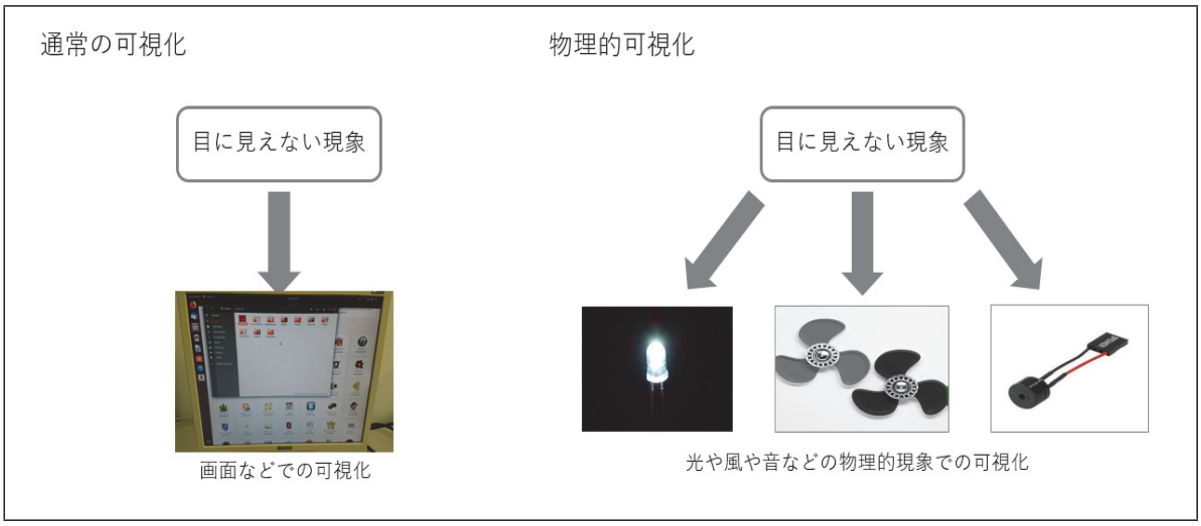

図 2.1　物理的可視化と物理的直接操作

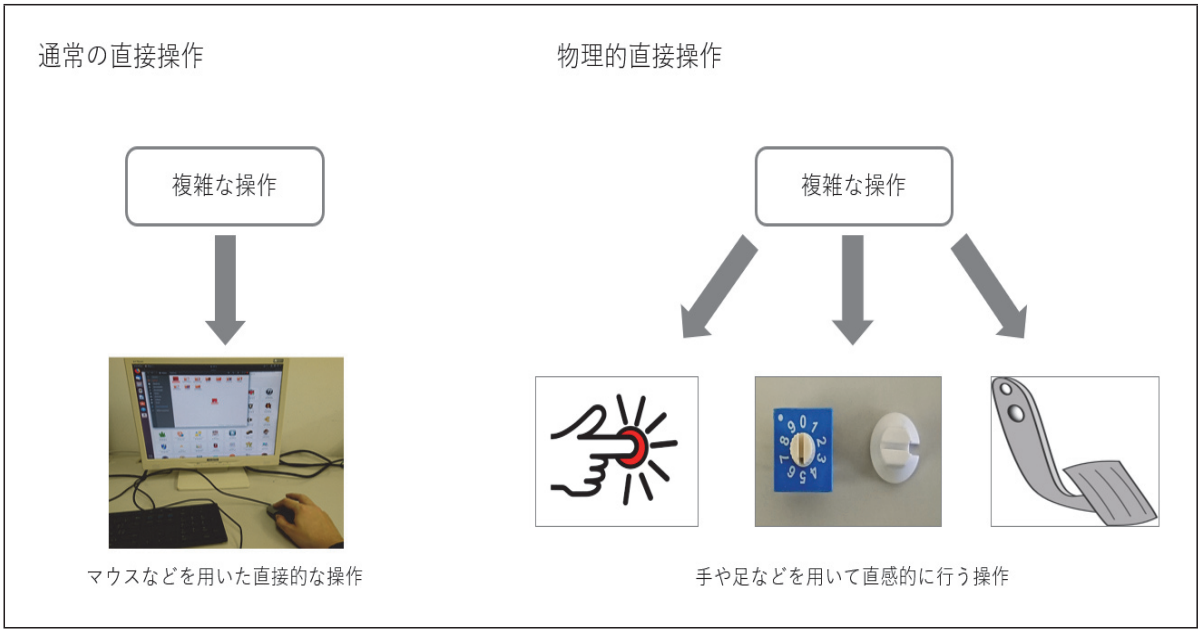

図 2.2　物理的可視化と物理的直接操作

第 3 章

IP アドレスの仕組みを学習するための教材の開発と評価

3.1　本章の概要

現行の学習指導要領では，中学校技術・家庭科技術分野（以下，技術科）および高等学校情報科において，ネットワークの仕組みについて学ぶ内容が含まれている [5][6]。具体的に，技術科では，ネットワークの学習内容として「ネットワークにおける基本的な情報利用の仕組みを知ること」と示されている。あわせて，「実践的・体験的な学習活動を通して生活に必要な基礎的・基本的な知識及び技術を身につけさせる」と示されており，ネットワークの学習においても実践的・体験的な学習を行うことが求められている。

ネットワークの実践的・体験的な学習として，情報系の大学を中心に，ネットワーク構築の演習を行っているところも多い [9][10]。しかし，多くの場合，ルータや Linux の設定を行いながら，ネットワークを構築する演習となっている。中学校や高等学校においてネットワーク構築の演習を行うことを考えると，ルータや Linux の設定は中学生や高校生にとっては敷居が高いと考えられる。我々は，中学校や高等学校で利用できる，ネットワークを学ぶための良い教材が無いことが，ネットワークの学習で実験や実習を行うこ

とを難しくしていると考えた。

一方で，社会のインフラであるネットワークは，我々が通常目にしないところで働いているシステムとなっている。そのため，ネットワークに触れることがなく，ネットワークに対する実感が乏しくなっており，ネットワークの仕組みを学ぶ際に，具体的なネットワークがイメージできず，学習を難しくしていると我々は考えた。

そこで，本研究では，前章で述べた物理的可視化と物理的直接操作を基に，ネットワーク技術の中でも重要である IP アドレスの仕組みを学習するための教材を開発し，有効性を確かめることを目的とした。

3.2　ネットワークにおける物理的可視化と物理的直接操作

技術科の学習指導要領において，技術科の目標として「ものづくりなどの実践的・体験的な学習活動を通して基礎的・基本的な知識及び技術を習得する」ことが挙げられている。例えば，江馬らは鋳造を用いたものづくり学習のための教材「銅鏡製作」を考案した [11]。このように，多くの技術科の学習内容において，ものづくりを通した実践的・体験的な活動をとりいれた研究事例がある。ネットワークの学習においても，ものづくりなどの実践的・体験的な学習活動を取り入れることが望ましい。

ネットワークにおけるものづくりとは，ネットワークの構築である。学習者はネットワークの構築を通して，ネットワークの仕組みを理解することが期待できる。しかしながら，前節で述べたように，我々は，中学校や高等学校で利用できるネットワークを学ぶための良い教材が無いことが，ネットワークのものづくりを通した実験や実習を行うことを難しくしていると考えた。

普段，我々は情報通信ネットワークはコンピュータのモニタによってネットワークの設定変更や通信の確認を行っている。図 3.1 は，WindowsOS において，コマンドプロンプト上で ifconfig コマンドを用いて IP アドレスの設定を変更したり，ping コマンドを用いて情報通信の確認を行っている画面

である。

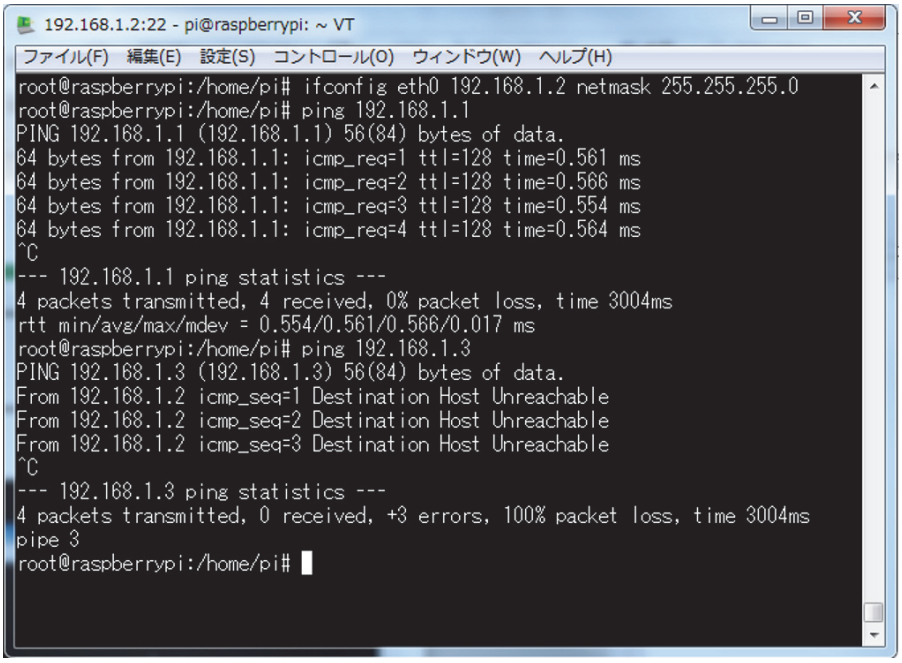

図 3.1　ネットワークの設定や確認を行っている画面

これらの操作では，「ping コマンド」や「ifconfig コマンド」のような専門の英語を用いた特殊なコマンドを，キーボードを用いて入力してネットワークの設定確認や変更をしたり，英語で表記された通信結果を読み取り，通信結果の確認を行う。ネットワークの学習教材についてコンピュータによる可視化や GUI によるマウス等を用いた直接操作（ダイレクトマニピュレーション）を利用した開発研究は多く見られる [12][13][14]。しかし，これらの学習教材は，キーボードやマウスを用い，画面上のアイコンの動きや文字でネットワークを可視化し，操作する教材である。加えて，前章で述べたように，現在のネットワークは見ることも触れることもできない。そのため，ネットワークの仕組みを初めて学習する中学生にとって，ネットワークを直感的にイメージしづらいと考えられる。

そこで，ネットワークを物理的可視化や物理的直接操作を基に教材を作成することにより，学習者が通信の確認を物理的な現象によってでき，ネットワークの構築を直感的に操作することで行うことができる。これにより学習

者はネットワークを直感的にイメージし，ネットワークを構築する実験を行うことができる。

物理的可視化や物理的直接操作に類似した情報通信ネットワーク学習教材の例として，村松らは電話網教材を開発し，技術科におけるネットワークの仕組みを学習する授業実践を行った [15]。また，現行の東京書籍の技術科教科書では，トランプを用いたルータゲームを掲載しており，ルータゲームを行うことでネットワークの仕組みについて学習することができる [16]。また，中学生のみを対象としていないが，日本科学未来館では情報が伝わる仕組みをボールの流れで可視化したインターネット物理モデルが展示してあり，ボールの流れを見ることでネットワークの仕組みを学習することができる [17]。これらの例は，ネットワークの仕組みを物理的な動きによって可視化し，操作する教材であるが，実際のネットワーク機器を用いておらず，電話網やトランプ，ボールなどでネットワークの動作を疑似的に表現している。そのため，学習者がネットワークの仕組みとして理解するには，頭の中でネットワークの仕組みに置き換える必要がある。

本研究では，物理的可視化の手段の中から LED の光を用い，また，様々な物理的直接操作の手段の中からボタンとダイヤルで物理的直接操作を行うことのできる実物のネットワークを構築できる教材の開発を行った。

次節では開発した教材について述べる。

3.3　IP アドレス学習教材

3.3.1　機能と実装

本教材の開発には，Raspberry Pi を使用した [18]。本教材の開発環境を表 3.1 に示す。

本教材の開発で用いた Raspberry Pi は OS を搭載しており，LAN ポートがあるため LAN ケーブルを配線することでネットワークを構築することができる。また，40 ピンの拡張コネクタが搭載されており，最大 27 本の汎用 I/O ポートを用いることが可能である。Raspberry Pi から物理的可

表 3.1　IP アドレス学習教材の開発環境

分類	詳細
使用機器	Raspberry Pi
OS	Raspbian
開発言語	Python
物理的可視化	LED
物理的直接操作	DIP ロータリースイッチ タクトスイッチ

視化や物理的直接操作に必要な電子部品の入出力を制御することができる。Raspbeery Pi を図 3.2 に示し，Raspberry Pi 拡張コネクタのピン配置を図 3.3 に示す。また，教材を作成するために必要な Raspberry Pi の初期設定について付録 A に記載する。

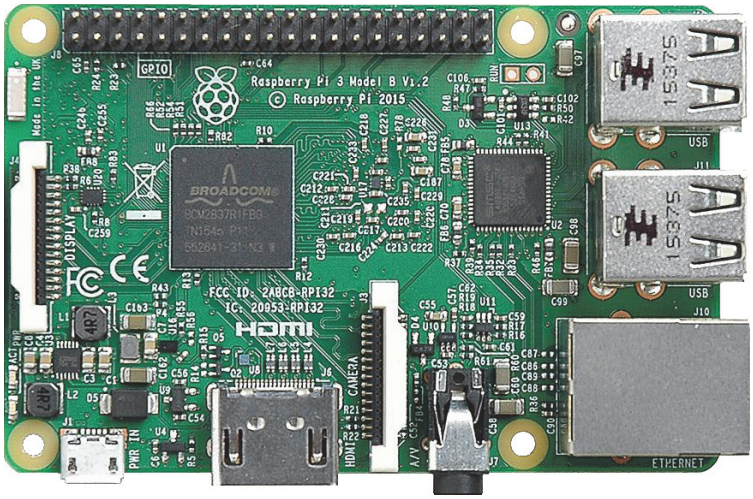

図 3.2　Raspberry Pi

Raspberry Pi の OS は Linux ベースの Raspbian であり，開発に使用したプログラム言語は，Python である。Python は豊富にライブラリを取り揃えており，本教材では Python の標準ライブラリである socket モジュールを利用した。以下に使用した関数を述べる [19]。

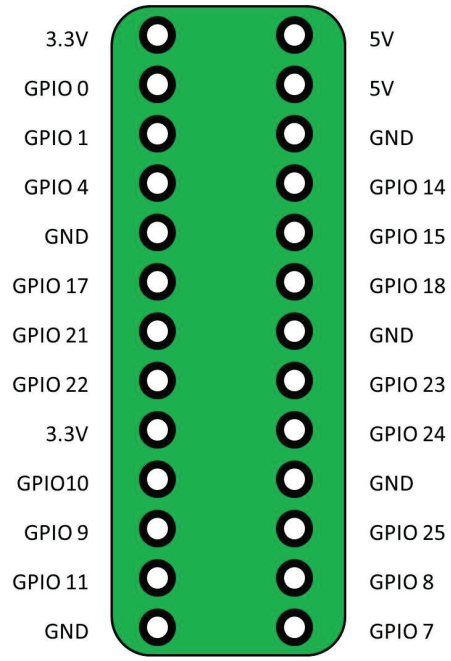

図 3.3　Raspberry Pi のピン配置

- socket.socket([family[type[,proto]]])

　　アドレスファミリ，ソケットタイプ，プロトコル番号を指定してソケットを作成する。

- socket.setdefaulttimeout()

　　新規に生成されたソケットの，デフォルトタイムアウトの値を秒数で指定する。

- socket.getprotobyname(protocolname)

　　プロトコル名を，ソケット関数の引数として指定することができる定数に変換する。

- socket.bind()

　　ソケットを制限する。今回はポート制限に用いた。

- socket.recvfrom(bufsize)

　　ソケットからデータを受信し，受信データの文字列と送信元アドレスを返す。

- socket.sendto(data,address)

ソケットにデータと接続先を指定して送信する。

- socket.close()

ソケットを閉じる。閉じられたソケットは一切操作を受け付けなくなる。

物理的可視化や物理的直接操作を行うためのインタフェース部分は，Raspberry Pi 用のユニバーサル基板で実装した。ユニバーサル基板上に回路を作成し，Raspberry Pi にユニバーサル基板を取り付け，教材を作成した。使用したユニバーサル基板を図 3.4 に示す。また，本教材で実装した回路図を図 3.5 に示す。

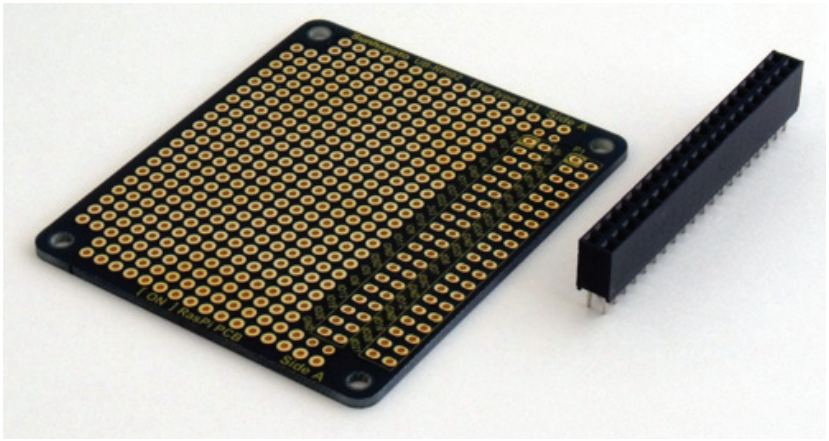

図 3.4 Raspberry Pi 用ユニバーサル基板

開発した IP アドレス学習教材は，以下の三つの機能を Python プログラムで実装している。

- ダイヤルとボタンを物理的直接操作することで自機に IP アドレスを設定する機能（config 機能）
- ping の受信を LED の点灯で物理的可視化する機能（ping 受信通知機能）
- ダイヤルとボタンを物理的直接操作することで特定の機器に ping を送信する機能（ping 送信機能）

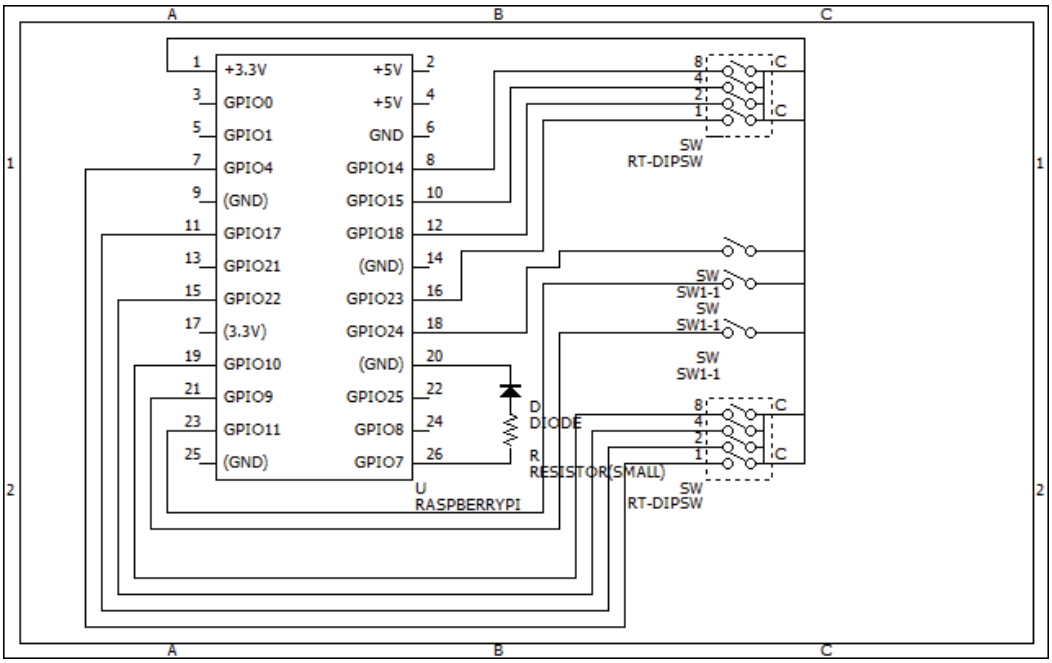

図 3.5　IP アドレス教材の回路図

IP アドレス学習教材上のダイヤルを操作し IP アドレスを指定し，それぞれの機能を実行するボタンを押すことにより，IP アドレスの設定や ping の送信を行い，LED が点灯することにより，通信の確認を行うことができる。開発した IP アドレス学習教材を図 3.6 に示す。

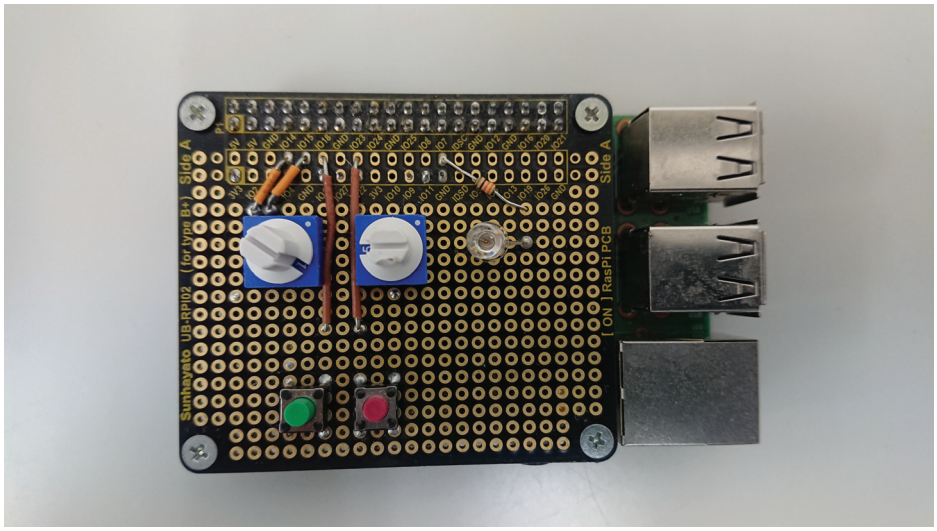

図 3.6　IP アドレス学習教材

3.3.2　config 機能

インタフェース上のダイヤルとボタンを操作することによって，自機の IP アドレスを設定できる機能である。

ダイヤルには 0 から 9 の DIP ロータリースイッチを用いた。DIP ロータリースイッチは，図 3.7 のように表面に 0 から 9 の数値が 10 進数で表記されており，右側のカバーを本体のつまみ部分に装着し，カバーのダイヤルを回すことにより，数値を指定することができる。DIP ロータリースイッチは，0 から 9 の 10 進数で指定し，学習者はキーボードを用いずに数値を指定することができる [20]。

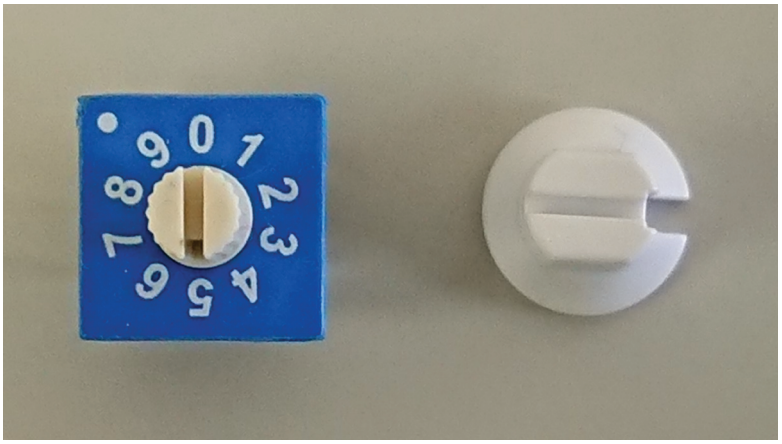

図 3.7　DIP ロータリースイッチ

DIP ロータリースイッチは C ピン 2 つと，番号 1・2・4・8 のピンがそれぞれ用意されており，シャフトを各数字のポジションに合わせることによって，それぞれのポジションに対応したピンと C ピンが接続される。以下の表 3.2 はポジションと C ピンとの接続の対応を示したものである。

IP アドレスを指定するには，ネットワークアドレスとホストアドレスの数値を入力する必要があるので，本教材では 2 つの DIP ロータリースイッチを用いた。インタフェースの左側のダイヤルがネットワーク部，右側のダイヤルがホスト部のアドレスを示している。図 3.6 の例では，左側のダイヤル

表 3.2　ピン番号と表示ポジションの対応

ピン番号	表示ポジション									
	0	1	2	3	4	5	6	7	8	9
1		●		●		●		●		●
2			●	●			●	●		
4					●	●	●	●		
8									●	●

がポジション 5，右側のダイヤルがポジション 1 に設定されており，これにより IP アドレスは 192.168.5.1/24 とプログラム内で設定される。設定される IP アドレスは変更可能である。以下に該当部分のプログラムを示す。

```
1  n=0
2  h=0
3  inputn1 = GPIO.input(4)
4  inputn2 = GPIO.input(17)
5  inputn4 = GPIO.input(22)
6  inputn8 = GPIO.input(10)
7  inputh1 = GPIO.input(14)
8  inputh2 = GPIO.input(15)
9  inputh4 = GPIO.input(18)
10 inputh8 = GPIO.input(23)
11 if inputn1 == True:
12   n = n + 1
13 if inputn2 == True:
14   n = n + 2
15 if inputn4 == True:
16   n = n + 4
17 if inputn8 == True:
18   n = n + 8
19 if inputh1 == True:
20   h = h + 1
21 if inputh2 == True:
22   h = h + 2
23 if inputh4 == True:
24   h = h + 4
25 if inputh8 == True:
```

```
26    h = h + 8
```

次に，図 3.6 下側にある左側の config ボタンを押すことにより，Python の system 関数を用いてサブシェル内で ifconfig コマンドを実行し，ダイヤルで合わせた IP アドレスを自機に設定する。設定される IP アドレスは config ボタンを押すたびに更新される。以下に該当部分のプログラムを示す。

```
1  network = (str(n))
2  host  = (str(h))
3  address = "ifconfig␣eth0␣192.168." + network + "." + host + "
       ␣netmask␣255.255.255.0"
4  os.system(str(address))
```

3.3.3 ping 受信通知機能

ping コマンドは，相手先ホストに対して ICMP エコー要求メッセージを送信し，相手先ホストから ICMP エコー応答メッセージを返送させるコマンドであり，このコマンドを使用することによって，相手先ホストと通信可能であるかを判断できる。

ping 受信通知機能は，ping を受信したときにユニバーサル基板上にある LED が点灯することによって ping が届いたことを知らせる機能である。

ping を受信したかどうかは，ICMP エコー要求メッセージの受信の有無で判断できる。本教材では Python の socket 関数を用いて，ICMP プロトコルを指定してソケットを生成し，もし受信データがあればインタフェース上の LED と対応する GPIO ピンに 1 秒間 3.3V の電圧を出力することで，ping 受信時に LED を点灯する。以下に、該当部分のプログラムを示す。

```
1  data = 0
2  sock=socket.socket(socket.AF_INET,socket.SOCK_RAW,socket.
       IPPROTO_ICMP)
3  sock.settimeout(0.1)
4  try:
5    data =sock.recv(255)
6    if data:
7      GPIO.output(25,True)
```

```
8       time.sleep(0.3)
9       GPIO.output(25,False)
10  except:
11    pass
```

3.3.4　ping 送信機能

ダイヤルを操作し，図 3.6 の右側の ping ボタンを押すことで，ダイヤルで設定された IP アドレスの機器に ping を送信する機能である。図 3.6 の例では，左側のダイヤルがポジション 5，右側のダイヤルがポジション 1 に設定されており，右側の ping ボタンを押すと，IP アドレス 192.168.5.1/24 の機器に対して ping を送信する。ping 送信には，config 機能と同様，Python の system 関数を用いてサブシェル内で，ダイヤルに設定された IP アドレスを指定して ping コマンドを実行している。以下に該当部分のプログラムを示す。

```
1  network = (str(n))
2  host = (str(h))
3  ping = "ping␣192.168." + network + "." + host + "␣-c␣1"
4  os.system(str(ping))
```

この機能と，前小節で述べた ping 受信通知機能により，ping の送信相手の教材の LED を光らせることができる。完全な IP アドレス学習教材の Python プログラムのソースコードを付録 B に記載する。

3.3.5　教材の利用方法

開発した教材は，microUSB ケーブルで給電を行う。図 3.8 のように電源が供給されると Raspberry Pi で OS が起動し，自動でログインや教材のためのプログラムが実行される。プログラムが起動すると，インタフェース上の LED が 1 秒点灯する。点灯が完了すると，教材として利用することができる。

教材には有線 NIC がついており，2 台の教材のそれぞれを，図 3.9 のよう

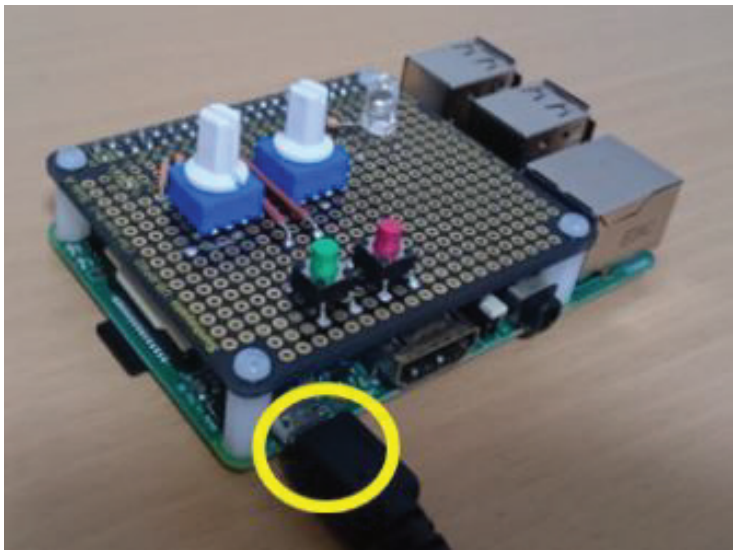

図 3.8　電源を投入した教材

に LAN ケーブルで接続することで，ダイヤルやボタンを物理的直接操作してネットワーク構築体験を行うことができる。

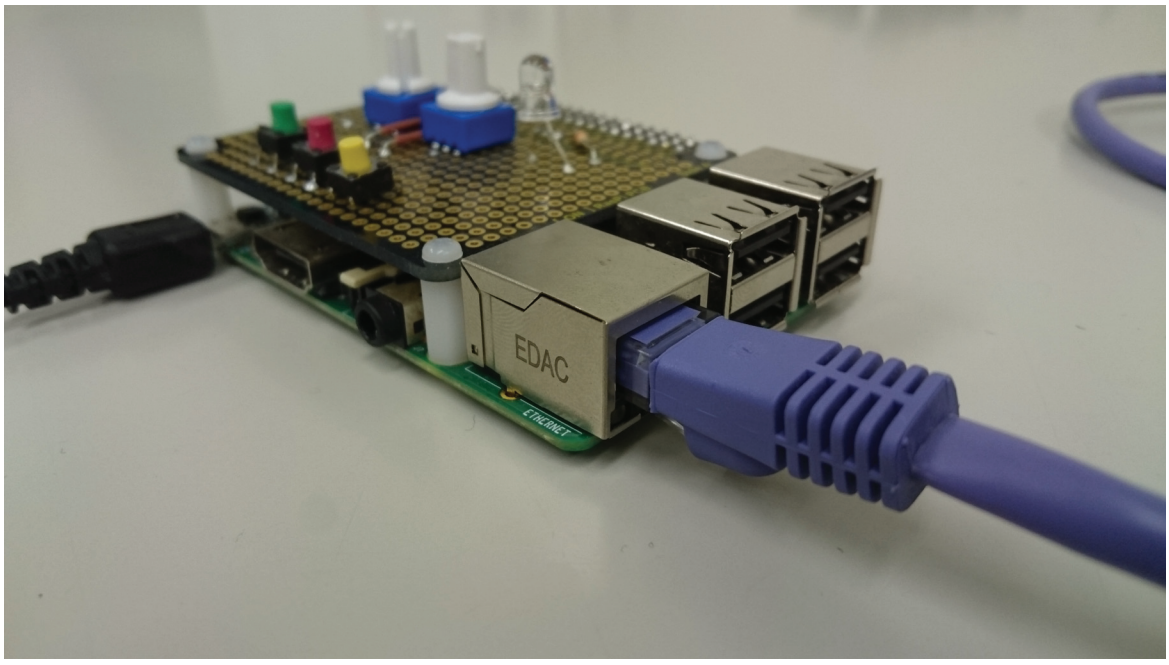

図 3.9　端末教材の LAN ケーブル接続

3.4　授業実践による教材の評価

開発したネットワーク学習教材を評価するために，教材を用いたネットワーク学習授業を設計し，授業実践を実施した。実施して得られた結果を基に教材の評価を行った。

3.4.1　実践及び評価の手続き

実践の学習内容

現在主流のネットワークであるインターネット技術について学ぶ内容には，アプリケーション層，トランスポート層，ネットワーク層，LAN(データリンク層，物理層) の各層に含まれる多岐にわたる技術が含まれる。

このなかで，IP アドレスはネットワーク層 (IP 層) における識別子であるが，単にコンピュータを識別するための記号に留まらない。他の層の一意性に関わるドメイン名やポート番号，MAC アドレスといった識別子は，それぞれ DNS や Firewall，ARP・RARP といった技術で IP アドレスに関連している。また，IP アドレスの構造に関わるネットマスクや論理演算，IP アドレスの枯渇による IPv6 の登場，そして IP アドレスによる経路制御技術など，IP アドレスは IP 層においても多くの内容と関わっている。

このように，IP アドレスは，他の層の技術や，経路制御技術などの IP 層の他の内容と関連する，インターネットの中心に位置づけられる技術であるため，本実践の学習内容としてとりあげた。

指導計画

開発した教材を用いた授業は，技術科「D 情報に関する技術」におけるネットワークの学習の 2 時間目「ネットワークの構築」として計画した。ネットワークの学習は計 2 時間で構成されており，1 時間目では，ネットワークの構成や利用方法について学習し，2 時間目では，ネットワークの仕組みを，教材を用いてネットワークの構築体験をしながら学習した。授業は情報端末

室で行い，実験群と統制群の 2 グループを設け，実験群では教材を用いた授業を実施し，統制群では教材を用いない授業を実施した。実験群と統計群の学習指導案を下記に示す。

		両群の 1 時間目の学習指導案	
時	学習内容	学習活動	
1	TCP/IP の学習 (15 分)	ワークシートを用いて，ネットワークに必要なプロトコル TCP/IP や IP アドレスについて知る。	
	IP アドレスの構造の検討 (20 分)	情報端末教室の PC の IP アドレスを調べ，発表する。調べた IP アドレスをもとに，IP アドレスの構造に関する特徴を見つけ出す。	
	IP アドレスの構造の学習 (15 分)	IP アドレスはネットワークとホストアドレスに分かれていることを知り，同一ネットワーク内では，ネットワークアドレスが同じでホストアドレスが異なることを知る。	
		2 時間目の実験群の学習指導案	2 時間目の統制群の学習指導案
時	学習内容	学習活動	学習活動
2	教材の説明 (10 分)	教材がコンピュータであることを知り，インタフェースの役割を知る。	通信できるかどうかを知るために ping コマンドがあることを知る。
	ネットワーク通信実験の概要の説明 (10 分)	二人一組で LAN を構築することを知り，前時で学習した IP アドレスの構造を基に，それぞれがどの IP アドレスに設定するかを検討する。	コマンドプロンプト上で ping コマンドの入力の仕方を知る。どの IP アドレスの機器に ping コマンドを実行すれば良いかを検討する。
	ネットワーク通信実験 (30 分)	検討した IP アドレスを指定し config ボタンで設定し，ペアの相手の IP アドレスを指定し ping ボタンを押して LED の点灯を確認する。また，様々なパターンの設定を試し，通信がうまくいくいかないを検討する。	同一ネットワーク内にある生徒用 PC や印刷機器に対し，ping コマンドを実行し，通信が行えることを確認する。また，様々な IP アドレスに ping コマンドを自由に実行し，通信ができるかどうかを調べる。

まず 1 時間目に生徒は，通信規約である TCP/IP やネットワークにおける識別子である IP アドレスの知識を学習した。その後，同一ネットワーク内にある各自 PC の IP アドレスを調べた。IP アドレスにはネットワークアドレスとホストアドレスがあることを知り，同じネットワークでは，ネットワークアドレスは同じで，ホストアドレスは一台ずつ異なるという特徴を見つけ出した。

実験群では次の 2 時間目に，調べた特徴をもとに，教材を用いたネットワーク構築体験を行った。ネットワーク構築体験では，2 人 1 組でネットワークの構築を行う。まず，電源を投入し教材を起動した。起動の LED 点灯を確認したらまず，2 台の教材同士を LAN ケーブルで繋ぐ。LAN ケーブ

ルを繋いだ後，ペアで互いの教材の IP アドレスを相談して決め，決定した IP アドレスにダイヤルを合わせ，config ボタンを押して自機に IP アドレスを設定した。その後，それぞれが相手の IP アドレスにダイヤルを合わせ，ping ボタンを押して相手の教材機器の LED が点灯することを確認した。また，ケーブルを抜いたり間違った IP アドレスを設定したりすると，通信ができないことも確認した。

統制群では，授業計画内の本教材を用いたネットワーク構築体験の部分を，情報端末室の生徒用 PC から印刷機など同一ネットワーク内にある機器に対し，コマンドプロンプトから ping コマンドを実行し通信を確認する実験にし，IP アドレスについての学習を行った。

評価の手続き

評価には，それぞれの群で事前，事後テストを実施し，実験群には教材に関するアンケートを行い，検証を行った。事前テストは授業前に実施し，事後テストは授業後，休憩時間を挟んだ後に実施した。事前テスト・事後テストの内容を図 3.10 に記載する。

テストは，①・②で 1 時間目に学習したネットワークの通信規約の問題，③・④・⑤で IP アドレスや IP アドレスの構造に関する問題，⑥・⑦・⑧でネットワークを構築するための IP アドレスの規則や，異なるネットワークと通信するためのルータの必要性を問う問題の計 8 問出題し，8 点満点で評価を行った。これらの問題は，実験群と統制群の両方で学習した内容で出題しており，差が出ないようにした。また，IP アドレスについて自由に記述する欄を設け，実験群と統制群とで比較を行った。

実験群を対象に，本実験に関する評価アンケートを実施した。アンケート用紙を図 3.11 に記す（図 3.11 には，アンケート項目の左横に番号が振ってあるが，この番号は，質問項目の説明の都合上，本論文で追加したものである）。①，②は教材を用いた実験の説明が取り組みやすかったかどうかを問う質問，③，④，⑤は物理的可視化と物理的直接操作が理解しやすいものであったかどうかを問う質問，⑥，⑦は教材を用いた実験が分かりやすいもの

であったかどうかを問う質問，⑧，⑨は実験を通してネットワークやその仕組みに興味を持ったかどうかを問う質問の計 9 項目の質問を行った。

授業実践事前・事後テスト

1. 以下の選択肢から文章内の空欄に当てはまる適当な語句の選択肢を選び答えなさい。

コンピュータどうしが通信をするためには、お互いが同じ「約束ごと」を守って通信しなければいけません。この約束ごとのことを（ ① ）といい、インターネットでは（ ② ）とよばれる約束ごとに従って情報のやり取りが行われています。

コンピュータどうしが通信するためには、通信相手のコンピュータを特定する必要があります。インターネットでは、通信相手を特定するために（ ③ ）を用い、コンピュータに割り振られ識別されています。（ ③ ）は、どのネットワークに所属しているかを表す（ ④ ）と、ネットワークの中でそれぞれのコンピュータが固有の値を持つ（ ⑤ ）から構成されています。

コンピュータどうしが通信をするには（ ⑥ ）ネットワークでなければいけません。（ ⑦ ）ネットワークにあるコンピュータと通信するには、（ ⑧ ）と呼ばれる情報の交通整理をする機器に中継をしてもらう必要があります。

① （　　）
② （　　）
③ （　　）
④ （　　）
⑤ （　　）
⑥ （　　）
⑦ （　　）
⑧ （　　）

(ア) IP アドレス (イ) ネットワークアドレス (ウ) 同じ
(エ) プロトコル (オ) ホストアドレス (カ) ルータ
(キ) 異なる (ク) TCP/IP

2. IP アドレスについて自由に述べなさい。

図 3.10　事前・事後テスト

アンケート調査表　　２８年　６月　２８日

このたびネットワーク学習教材を用いた実践授業にあたり、授業内で用いた教材のアンケート調査を実施することになりました。
様々な方からのご意見を参考にさせていただきたいので、ご協力くださいますようお願い申し上げます。

１．以下の質問事項について、あてはまる評価の□にチェックを入れていください。

質問事項	評価				
	満足	やや満足	普通	やや不満	不満
①教材に興味を持って取り組めましたか。	□	□	□	□	□
②教材の使い方はわかりやすかったですか。	□	□	□	□	□
③教材のボタンやダイヤルは扱いやすかったですか。	□	□	□	□	□
④教材の LED の点灯によってネットワークの動作が確認できましたか。	□	□	□	□	□
⑤教材の 2 つのダイヤルの役割の違いについて理解できましたか。	□	□	□	□	□
⑥教材を用いたネットワーク構築は容易でしたか。	□	□	□	□	□
⑦教材によってネットワークの仕組みを理解できましたか。	□	□	□	□	□
⑧教材を用いた体験によってネットワーク構築について興味を持ちましたか。	□	□	□	□	□
⑨教材を用いた体験によってネットワークの仕組みについて興味を持ちましたか。	□	□	□	□	□

２．教材について感想や要望があればご自由にお書きください。

これでアンケートは終わりです。ご協力ありがとうございました。
なお、この用紙は厳重に保管し、教材評価の参考にする以外の目的には使用いたしません。

図 3.11　アンケート用紙

3.4.2　結果と考察

実践の対象者と実践の様子

教材を用いた授業実践を，2016 年 6 月 28 日と 7 月 11 日に広島大学附属東雲中学校 1 年生を対象に行った。実験群は男子 17 名，女子 21 名の 38 名，

統制群は男子 20 名，女子 19 名の 39 名であった。実験群で，教材を用いたネットワーク構築体験における，教材同士を LAN ケーブルで接続する様子を図 3.12，通信の確認を LED で行っている様子を図 3.13 に記載する。

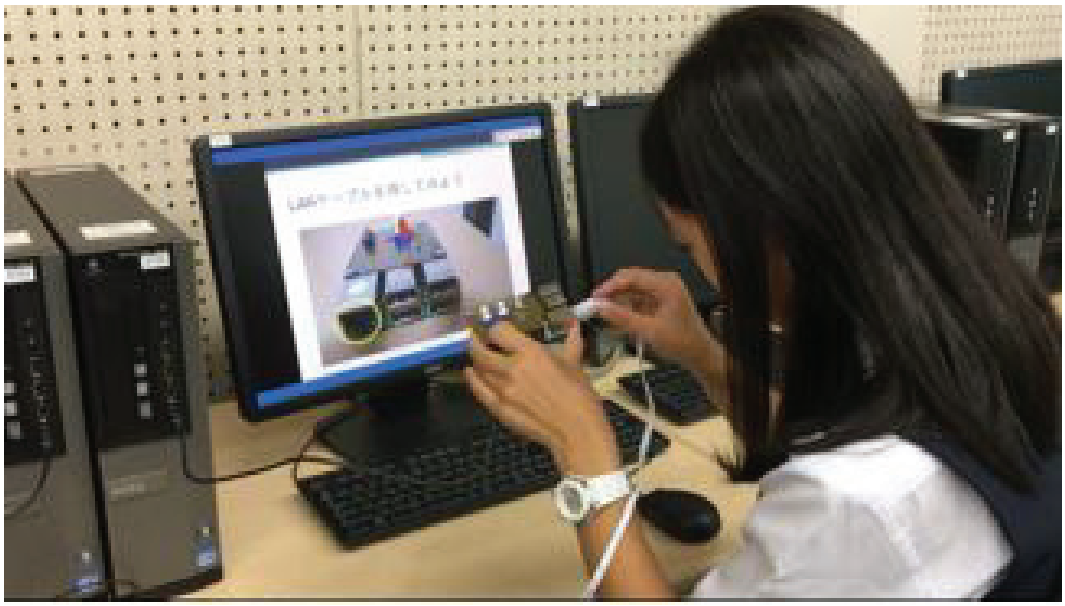

図 3.12　教材同士を LAN ケーブルで接続する様子

図 3.13　通信の確認を LED で行っている様子

事前テスト・事後テストの考察

事前テスト・事後テストの結果のそれぞれの群の平均得点と標準偏差を表 3.3 に記載する。

表 3.3　事前・事後テストの両群における結果

	事前テスト		事後テスト	
	M	SD	M	SD
実験群	3.71	2.40	7.05	1.37
統制群	3.56	2.40	5.48	2.21

学習者のテスト結果に関する 2 要因混合計画の分散分析（参加者間要因：学習者 [実験群，統制群] ×参加者内要因：テスト [事前テスト，事後テスト]）を行った。分散分析の結果では，交互作用は有意 ($F(1,75) = 6.53$, $p<.05$) にみられた。交互作用が有意にみられたので，各要因における単純主効果を検討したところ，実験群における参加者内要因の単純主効果 ($F(1,75) = 72.413$, $p<.001$) が認められ，統制群における参加者内要因の単純主効果 ($F(1,75) = 23.976$, $p<.001$) が認められた。また，事前テストにおける参加者間要因の単純主効果 ($F(1,150) = 0.1$,ns) は認められず，事後テストにおける参加者間要因の単純主効果 ($F(1,150) = 11.392$, $p<.001$) が認められ，事後テストにおいて統制群よりも実験群の方がより有意に高い結果となった。統制群よりも実験群の方が事後テストにおいてより高い得点をとったことが示され，教材が有効に機能したと考えることができる。自由記述について，記述内容を IP アドレスに関する有効な記述なし，IP アドレスの一意性についての記述，IP アドレスの構造についての記述，IP アドレスの一意性と構造についての両方の記述の 4 種類の記述に分類し，4 種類の記述の割合を図 3.14 に示す。独立性の検定を行った結果，両群と記述内容が無関係であることが棄却され（$T = 19.13 > 7.81 = \chi^2_3(0.05)$），関係性が認められた。統制群では，有効な記述なしが 44% であったのに対し，実験群では 16% と割合が低い結果となった。また，IP アドレスの一意性についての記述の割合

は，「IP アドレスの一意性についてのみ記述した生徒の割合」と「一意性・構造の両方を記述した生徒の割合」を合計して，統制群では 48% であったのに対し，実験群では 74% と高かった。また，IP アドレスの構造についての記述の割合は，「IP アドレスの構造についてのみ記述した生徒の割合」と「一意性・構造の両方を記述した生徒の割合」を合計して，統制群が 20% と低く，実験群は 68% と高かった。このように，実験群の方が IP アドレスの一意性や，IP アドレスの構造について統制群と比べて多くの生徒が述べており，実験群の方が IP アドレスについて理解が高く，教材が有効に機能したと考えられる。

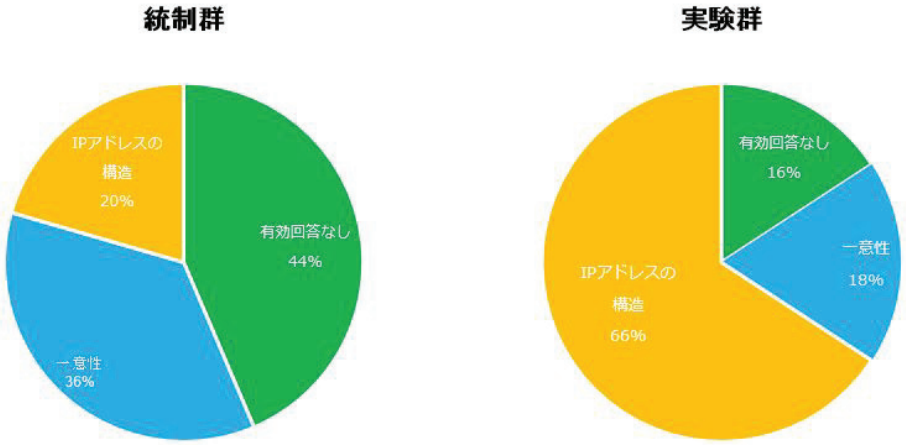

図 3.14　IP アドレスについての記述の割合

アンケート結果の考察

アンケート結果を図 3.15 に記す。①「教材に興味を持って取り組めたか」に対しては，4，5 と答えた生徒は 95% であった。②「教材の使い方はわかりやすかったか」，③「教材のボタンやダイヤルは扱いやすかったか」に対しては，4，5 と答えた生徒はそれぞれ 87%，82% であった。④「教材の LED の点灯によってネットワークの動作が確認できましたか」，⑤「教材の 2 つのダイヤルの役割の違いについて理解できましたか」に対しては，4，5 と答

えた生徒はともに 97% であった。⑥「教材を用いたネットワーク構築は容易でしたか」に対しては，4，5 と答えた生徒は 84% であった。⑦「教材によってネットワークの仕組みを理解できましたか」に対しては，4，5 と答えた生徒は 81% であった。このように，多くの生徒が物理的可視化と物理的直接操作による教材を，扱いやすい，確認しやすいものと感じているので，物理的可視化や物理的直接操作がネットワークの学習を行う教材として機能したと考えられる。⑧「教材を用いた体験によってネットワーク構築について興味を持ちましたか」，⑨「教材を用いた体験によってネットワークの仕組みについて興味を持ちましたか」に対しては，4，5 と答えた生徒がそれぞれ 73%，76% であった。このように，生徒たちの多くが，本教材を用いたネットワーク構築体験を通して，ネットワークの仕組みや構築に関して興味を持ったことが示された。以上のことから，本教材はネットワーク学習を行う教材として有効であることが確認できた。

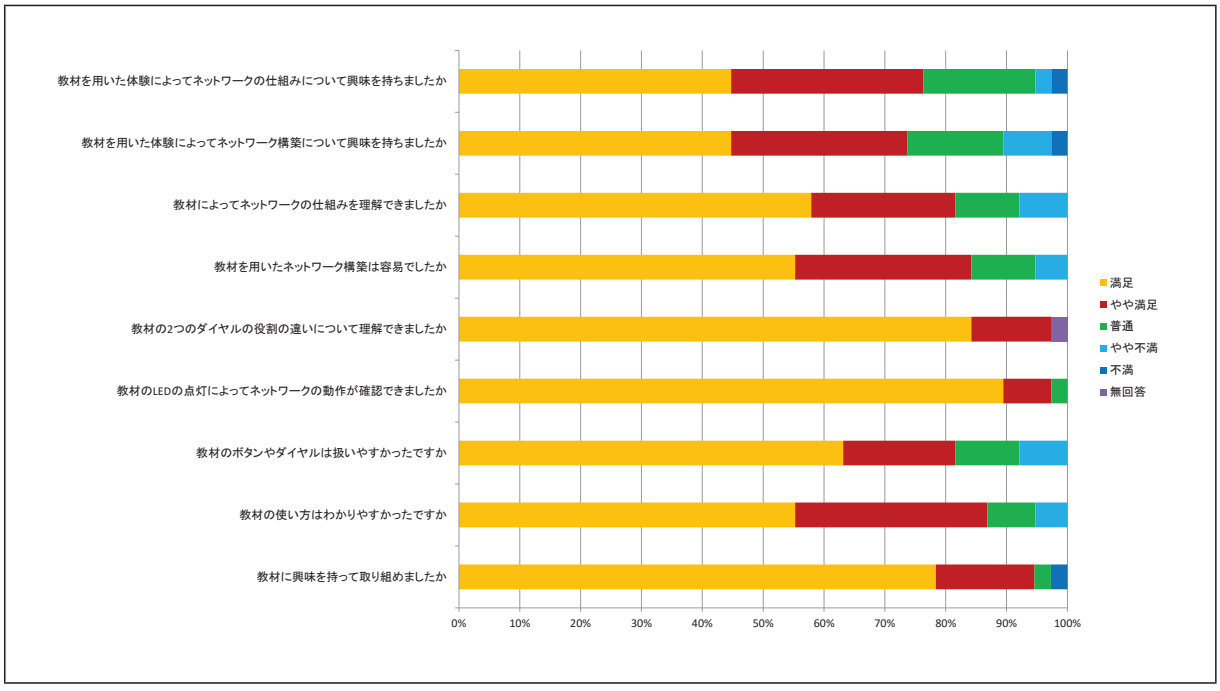

図 3.15　アンケート結果

3.5　本章のまとめ

本研究では，物理的可視化や物理的直接操作によるネットワーク学習教材を開発し，教材を用いた授業実践を実施し，事前・事後テストとアンケート結果をもとに教材の有効性を確かめた。本教材を用いることによって，現状では見ることのできないネットワーク技術を，LED を用いて物理的可視化し，ダイヤルやボタンを用いて物理的直接操作することで，直感的にネットワークの機能や動作を理解することができ，IP アドレスの仕組みを効果的に学習できることが確認できた。

第 4 章

ネットワーク構築を学習するためのルータ教材の開発

4.1　本章の概要

前章で開発した IP アドレス学習教材は，現在インターネットで用いられているプロトコル TCP/IP において重要な技術である IP アドレスについて学習する教材であった。IP アドレス学習教材を用いた実験では，同一ネットワーク内の通信における IP アドレスの設定や通信の確認を行った。しかしながら，インターネットでは，複数のネットワーク間で通信を行っており，どのように端末同士が通信しているかを学習することができない。より現実に即したネットワークの仕組みを理解するには，複数のネットワーク間を通信するのに必要なルータの役割を学習する必要がある。中学校技術科や高等学校情報科の学習指導要領においても，ルータの働きや役割を理解させるようにすると記されている。

そこで，既に開発した物理的可視化と物理的直接操作を取り入れた IP アドレス学習教材を拡張し，同様のコンセプトをもとに複数のネットワーク間の通信の仕組みを学習する教材としてルータ教材の開発を行った。そして，開発したルータ教材が教材として有効に機能するかどうか検証実験を行った。

次節では，ルータについて詳しく述べる。

4.2　ルータ

コンピュータ同士が通信を行うには，お互いが同じ通信手順に従って通信をする必要がある。この従うべき通信手順を通信プロトコルといい，国際標準化機構が策定した OSI 参照モデルやインターネットで使用されている TCP/IP などがある。これらのプロトコルは実装時，役割に応じていくつかの層が積み重なった形で構成されている。

TCP/IP では，ネットワークケーブルの物理的な仕様や隣接する機器のデータ通信方法の規格を示すネットワークインターフェース層，論理的なネットワーク構成及び経路を示すネットワーク層，通信の性質を定めたトランスポート層，アプリケーションのデータのやり取りの規格を示したアプリケーション層がある。TCP/IP を利用して通信を行うことで，世界中の離れたコンピュータ同士が一瞬で接続し，通信を行うことができる。

インターネットは，世界中にある LAN とよばれる狭い範囲におけるネットワーク同士が相互に接続した巨大なネットワークである。ひとつの LAN の中では端末同士が直接通信できるが，異なる離れた LAN にある端末とは通信することができない。複数の LAN を経由して通信を行うには，通信を中継するための機器であるルータが必要である。ルータは，ネットワーク層の機能を提供しており，離れたコンピュータ同士がお互いに通信し合うインターネットにおいて，必要不可欠な機器であり，情報通信ネットワークを学習するには物理的な通信媒体である物理層及びデータリンク層の端末の接続だけではなくネットワーク層のルータについての学習も必要である。

ルータは，端末と異なり二つ以上のネットワークインターフェースを持っている。異なるネットワーク間で通信を行う場合，それぞれのネットワークにある端末とルータのネットワークインターフェースを正しく LAN ケーブルで接続する必要がある。さらに，それぞれのネットワークインターフェースに正しく IP アドレスを設定することで，ルータが異なるネットワークに

データの中継を行うことができ，異なるネットワークにある端末が通信を行うことが可能になる。つまり，ルータの設定には，LAN ケーブルで配線する物理的なデータリンク層や物理層の設定から IP アドレスを付与する論理的なネットワーク層の設定を行う必要があり，それら両方を理解する必要がある。

異なるネットワーク同士の通信を行うためのルータ機能の学習に関する教材はいくつか研究開発されている。精鷹らは，画面上に描画し，設定したネットワークをシミュレーションすることにより検証することのできる教育向けネットワークシミュレータを開発した [21]。また，井口らは，自由に仮想的なルータなどの情報通信機器を画面上に配置し，ルータの設定をすることで仮想ネットワークを構築することのできるシステムを開発した [22]。これらのシステムは仮想的に機器同士を接続し，ルータの設定を行うことでネットワークを構築し，画面上で情報通信ネットワークを可視化するものであるため，ルータを用いたネットワーク層の論理的な設定などを学習することができるが，データリンク層や物理層の物理的な配線を体験的に学習することが出来ず，中学生や高校生には理解するのは難しいと考えられる。中学校技術科におけるルータの学習教材としては，本多らがルータを中心にデータの送受信を人間が模倣する体験型教材「人間ルータゲーム」を開発した [23]。 これは，人間がルータや端末を模擬的に演じており，実機を用いたデータリンク層や物理層の物理的な配線やネットワーク層の論理的なネットワークの設定などを行わないため，ルータを用いたネットワークの構築を体験的に学習することができない。

本研究では，物理的可視化と物理的直接操作によるネットワーク学習教材としてルータ教材を開発した。物理的可視化と物理的直接操作では，情報通信ネットワークを物理的な現象で可視化し，物理的な物を操作し情報通信ネットワークの論理的な設定ができるようにするものである。実際にネットワークを構築することのできるコンピュータ機器を用いて，物理的に配線し，ネットワークを構築することができ，また，ルータの役割を画面上やカードなどではなく，直感的に理解しやすい LED の光を用いて物理的に可視化を

行うことができる。情報通信ネットワークの設定をキーボード操作で行うのではなく，ダイヤルやボタンといった単純な操作で実現できるため，論理的な通信の設定を行うことができ，試行錯誤しながらネットワークの構築を行い，ネットワークインターフェース層とネットワーク層について実践的・体験的に学習することができる。

4.3　ルータ教材の概要

実践的・体験的な情報通信ネットワークの構築を行う教材には，実際の TCP/IP を用いて通信を行い，複数のネットワークインターフェースやルータの機能を持つコンピュータ機器を用いる必要がある。また，物理的可視化や物理的直接操作が行えるように，電子部品を用いて電子回路を作成できる機器であるのが望ましい。

そこで，ルータ教材には，先行研究で開発した IP アドレス学習教材と同様に，シングルボードコンピュータである Raspberry Pi で開発を行った。

ルータには複数のネットワークインターフェースが必要であるが，既に開発した IP アドレス学習教材では，ネットワークインターフェースは 1 つしかないため，複数のネットワークを構築することができない。ルータ教材では，LAN アダプタを USB ポートに接続することで，ネットワークインターフェースを 2 つ（eth0 と eth1）用意し，複数のネットワークを構築することが可能になる。

直感的に情報通信ネットワークの設定や情報通信の確認を行うため，情報通信ネットワークのデータ通信の物理的可視化のための機器には IP アドレス学習教材で用いた LED とは異なり，光の流れを表現することのできるフルカラーシリアル LED テープ（以下，LED テープ）を用い，物理的直接操作のための機器には IP アドレス学習教材と同様にダイヤルとボタンを用いた。教材の開発環境を表 4.1 に示し，開発した教材を図 4.1 に示す。

開発したルータ教材は，通常のネットワーク機器であるルータと同等の異なるネットワークにパケットを転送する機能を持ち，さらに以下のような機

表 4.1　教材の開発環境

分類	詳細
使用機器	Raspberry Pi
OS	Raspbian
開発言語	Python
物理的可視化	LED テープ
物理的直接操作	DIP ロータリースイッチ タクトスイッチ

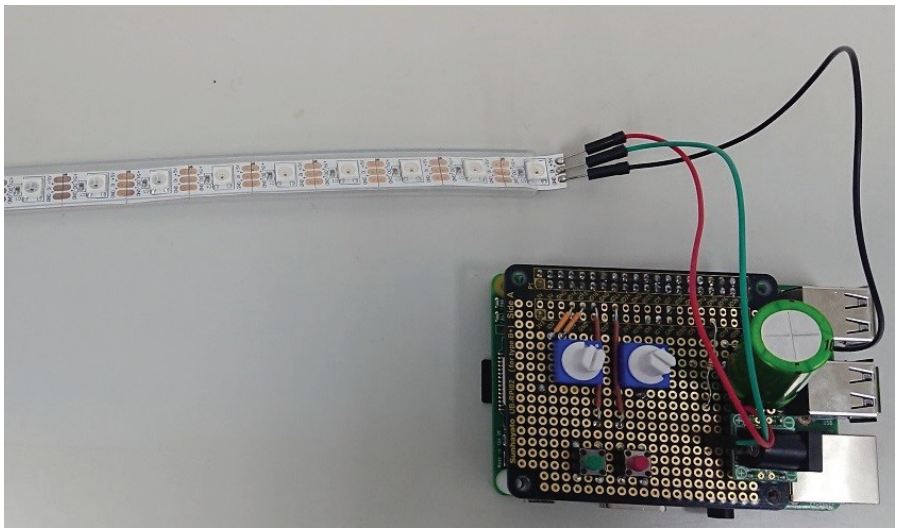

図 4.1　開発したルータ教材

能を持っている。

- DIP ロータリースイッチと ifconfig ボタンを用いて，IP アドレスを設定する機能
- DIP ロータリースイッチと traceroute ボタンを用いて，指定した IP アドレス宛に経路順に ping を送信し経路を確認する機能
- フルカラーシリアル LED を用いて，ICMP パケットを受け取ったら自機の IP アドレスを LED の群と色で表現し，流れるように光る機能

4.4　物理的直接操作および物理的可視化の実装

4.4.1　ダイヤルによる IP アドレスの設定

IP アドレスの設定は，通常のコンピュータ機器では，キーボードで専用のコマンドを用いて入力する。物理的直接操作では，専用のコマンドやキーボードを用いず，物を操作して IP アドレスを指定する。開発した教材では，IP アドレスの設定に 2 つの DIP ロータリースイッチを用いた。基板上の 2 つの DIP ロータリースイッチを操作し，IP アドレスを指定するようにした。DIP ロータリースイッチは，2 つの C ピンと 1，2，4，8 の 4 つのピンの計 6 つのピンがあり，ダイヤルを 0〜9 の各数字のポジションに合わせることで，それぞれのポジションに対応した C ピンと 4 つのピンが導通するようになっている。

IP アドレスの設定には，回路上にある ifconfig ボタンを押すことによって，ネットワーク環境の状態確認や設定のために用いる ifconfig コマンドを，コールバック関数を用いて実行する。

既に開発した IP アドレス学習教材では，自機の IP アドレスを 2 つのダイヤルを用いて指定し ifconfig ボタンを押すことで設定できる。ネットワークアドレス側（左側）のダイヤルを 1，ホストアドレス側（右側）のダイヤルを 5 にすると，指定される IP アドレスは 192.168.1.5/24 となる。IP アドレスの設定には，ネットワーク環境の状態確認や設定のために用いる ifconfig コマンドを，Phython の os モジュールを用いてプログラム上で実行する。

開発したルータ教材にある 2 つのネットワークインターフェースには，それぞれのデバイスに IP アドレスを指定する必要がある。そこで，ルータ教材の 2 つのダイヤルは，2 つあるネットワークインターフェースのホストアドレスは固定し，それぞれのネットワークアドレスをダイヤルで指定するようにした。左側のダイヤルを 1，右側のダイヤルを 5 にして ifconfig ボタンを押すと，eth0 の IP アドレスを 192.168.1.254/24 に eth1 の IP アドレスを 192.168.5.254/24 に設定する。

4.4.2　LED テープの制御

物理的可視化とは，目で直感的に理解できる LED のような光や，扇風機の羽のような回転する動きなど，物理的な物を用いる可視化である。情報通信ネットワークの物理的可視化では，コンピュータ同士の間でデータがやりとりされている様子が直感的に理解できるような物理的可視化を行う必要がある。そこで，本教材では LED テープを用いた。フルカラーシリアル LED テープは，フルカラー LED が図 4.2 のようにシリアルに連なっている LED テープである。LED テープは VCC，GND，PWM 入力信号の 3 つの端子があり，VCC と GND 端子を外部電源に繋ぎ，PWM 入力信号用の端子を Raspberry Pi の汎用入出力ピンに接続することにより，制御することができる [24]。

図 4.2　フルカラーシリアル LED テープ

GRB の順に各色 8bit で 1 モジュール当たり 24bit のデータを決められたタイミングで信号を入力することでデータを送信する。連結して使用する場合，Raspberry Pi から近い側のモジュールのデータから送信し，50s 以上 LOW が続いた場合，全モジュールでデータが確定し，LED への出力へ反映される。Raspberry Pi で使用する場合，LED テープを制御するためのライブラリ「rpi_ws281x」を利用することで，制御することができる。

LED テープは，複数の LED を同時に制御できるので，光の軌跡などの流れが表現できる。光の流れを表現することで，どのように送信先の端末までデータが流れているかを表現することができる。

4.4.3　フルカラーシリアル LED テープを用いた光の軌跡によるデータの流れの物理的可視化

経路の確認機能には traceroute コマンドの原理を利用した。

IP パケットには，パケットがループし続ける状態を防ぐために，生存時間が決められている。それぞれの IP パケットには TTL（Time to Live）が割り当てられ，パケットがルータを通過するたびに 1 ずつ減らされていき，0 になると IP データグラムが破棄される。このときルータは，ICMP 時間超過メッセージ (ICMP Time Exceeded Message) を送信側に送り，パケットが破棄されたことを知らせる。このメッセージを応用したコマンドが traceroute コマンドであり，受信ホストまでの IP パケットの生存時間を 1 から順番に増やしていき，ICMP 時間超過メッセージを受け取ることで，送信ホストから受信ホストまで，どのルータを経由したのかを表示するコマンドである。

ルータ教材および既存の IP アドレス学習教材では，基板上の traceroute ボタンを押すことで，ダイヤルで指定した IP アドレス宛に UDP パケットの TTL の値を 1 から 1 ずつ増やしながら送信して，ICMP 時間超過メッセージの送信アドレスを受け取るようにした。そして，受け取った IP アドレス宛に ping コマンドの実行を行った。ping コマンドの実行は，ifconfig コマンド同様，Python の os モジュールを用いてプログラム上で実行するようにした。以下に該当部分のプログラムを示す。

```
1  dest_addr = "192.168." + network + "." + host
2  socket.setdefaulttimeout(10)
3  icmp = socket.getprotobyname('icmp')
4  udp = socket.getprotobyname('udp')
5  ttl = 1
6  max_hops = 10
7  port = 33434
8  while True:
9    recv_socket = socket.socket(socket.AF_INET,socket.SOCK_RAW,
         icmp)
10   send socket = socket.socket(socket.AF_INET,socket.
```

```
      SOCK_DGRAM, udp)
11  send_socket.setsockopysocket.SOL_IP, socket.IP_TTL, ttl)
12  recv_socket.bind(("", port))
13  curr_addr = None
14  try:
15    send_socket.sendto("", (dest_addr, port))
16  except socket.error:
17    pass
18  try:
19    _, curr_addr = recv_socket.recvfrom(512)
20  curr_addr = curr_addr[0]
21  except socket.error:
22    pass
23  finally:
24    send_socket.close()
25    recv_socket.close()
26  if curr_addr is not None:
27    trace = "ping␣" + curr_addr + "␣-c␣1"
28    os.system(str(trace))
29    time.sleep(1)
30  ttl += 1
31  if curr_addr ==dest_addr or ttl > max_hops or curr_addr is
      None:
32    break
```

このプログラムは，受信用の icmp プロトコルのソケットと送信用の UDP プロトコルのソケットを作成し，変数 ttl の値を 1，ポート番号を 33434 と指定し，宛先のアドレスまで UDP パケットを送信する。もし，ICMP パケットが返ってくれば，返信先アドレスに再度 os.system 関数を用いてサブシェル内で ping コマンドを実行して ping を送信する。この動作を，ttl を 1 ずつ増やしながら，ICMP パケットを返信したアドレスが最初に設定した IP アドレスと一致するまで繰り返す。

また，ルータ教材および IP アドレス学習教材は ICMP パケットを受信したときに，LED テープを光の軌跡が自機の方向に流れるように制御する。受信端末教材とルータ機器の LED テープを送信側に向け，IP アドレス学習教材側がダイヤルを通信相手の端末の IP アドレスを指定し traceroute ボタンを押すと，traceroute 機能により，通った経路順に ping を送信する。そ

して，ping を受け取った端末やルータは，ルータ，端末の順に自機の方向に LED テープが光るため，送信した端末から受信した端末の方向へフルカラーシリアル LED の光の軌跡で表現することができる。これにより，どのような経路でコンピュータが目的の IP アドレスのコンピュータにデータが転送されていったのかを，LED テープの光の流れによって知ることができる。ただし，受信端末教材と送信端末教材を入れ替える場合，LED テープを逆側（送信側）に配置する必要がある。例として，IP アドレス学習教材 2 台とルータ教材 1 台を用いた場合の制御の流れを以下に示し，制御の流れを図解したものを図 4.3 に示す。

1. 図 4.3 の IP アドレス学習教材 1 のダイヤルを IP アドレス学習教材 2 宛にし，traceroute ボタンを押す
2. traceroute 機能によりルータ教材に ping を送信
3. ルータ教材の LED テープが IP アドレス学習教材 1 から自機の方向に流れるように光る
4. IP アドレス学習教材 2 に ping を送信
5. IP アドレス学習教材 2 の LED テープがルータ教材から自機の方向に流れるように光る

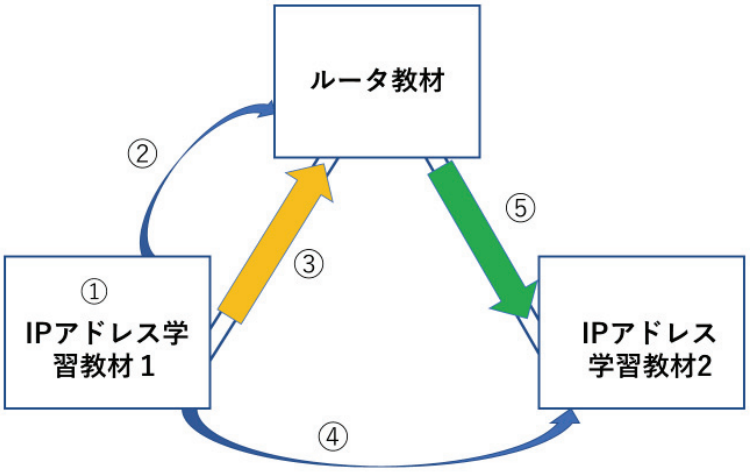

図 4.3　教材による LED テープ制御の流れ

4.4.4 LED テープの色と群の数による IP アドレスの物理的可視化

前節で述べた LED テープを用いたデータの流れの物理的可視化では，ただ光の軌跡を表現するだけではなく，光の軌跡でどの IP アドレス宛に ping が送られているかを表現する IP アドレスの物理的可視化を行った。すなわち，ping を受信した端末は自機の IP アドレスのネットワーク部を表 4.2 のように異なる色によって表現し，ホスト部を表 4.3 のように流れる LED 群の数の違いによって表現した。これにより，論理的な IP アドレスの物理的可視化を行うことができる。

表 4.2　ネットワーク部による色の違い

ネットワーク部 (/24)	色
192.168.0.*	天色
192.168.1.*	赤色
192.168.2.*	緑色
192.168.3.*	青色
192.168.4.*	黄色
192.168.5.*	紫色
192.168.6.*	水色
192.168.7.*	白色
192.168.8.*	黄緑色
192.168.9.*	桃色

表 4.3　ホスト群における LED 群の数

ホスト部 (/24)	LED 群の数
..*.0	無し
..*.1	1
..*.2	2
..*.3	3
..*.4	4
..*.5	5
..*.6	6
..*.7	7
..*.8	8
..*.9	9
..*.254	10

ping の受信を検知したとき，LED テープが IP アドレスを色と群の数で表現し，自機に向かって光の軌跡を表現するように制御する。以下に該当するプログラムの一部を記述する。

```
#colorWipereverse
def colorWipereverse(strip, color, wait_ms=10):
  for i in range(strip.numPixels(),-1,-1):
    strip.setPixelColor(i, color)
    strip.show()
    time.sleep(wait_ms/1000.0)

def colorWipereverse5(strip, color, wait_ms=10):
  for q in reversed(range(5)):
    for i in range(0,strip.numPixels(),6):
      strip.show()
      time.sleep(wait_ms/1000.0)

sock = socket.socket(socket.AF_INET,socket.SOCK_RAW,socket.
    IPPROTO_ICMP)

while True:

  data = sock.recv(255)
  time.sleep(0.28)

  if ord(data[20]) == 8:

    if ord(data[12]) == 192:
      if ord(data[13]) == 168:
        if ord(data[14]) == 0:

        elif ord(data[14]) == 1:
          if ord(data[15]) == 0:
            colorWipereverse(strip, Color(255,0,0))
            colorWipereverse(strip, Color(0,0,0))
          elif ord(data[15]) ==1:
            colorWipereverse1(strip, Color(255,0,0))
            colorWipereverse(strip, Color(0,0,0))
          elif ord(data[15]) ==2:
            colorWipereverse2(strip, Color(255,0,0))
            colorWipereverse(strip, Color(0,0,0))
          elif ord(data[15]) ==3:
```

```
38              colorWipereverse3(strip, Color(255,0,0))
39              colorWipereverse(strip, Color(0,0,0))
40            elif ord(data[15]) ==4:
41              colorWipereverse4(strip, Color(255,0,0))
42              colorWipereverse(strip, Color(0,0,0))
43            elif ord(data[15]) ==5:
44              colorWipereverse5(strip, Color(255,0,0))
45              colorWipereverse(strip, Color(0,0,0))
46            elif ord(data[15]) ==6:
47              colorWipereverse6(strip, Color(255,0,0))
48              colorWipereverse(strip, Color(0,0,0))
49            elif ord(data[15]) ==7:
50              colorWipereverse7(strip, Color(255,0,0))
51              colorWipereverse(strip, Color(0,0,0))
52            elif ord(data[15]) ==8:
53              colorWipereverse8(strip, Color(255,0,0))
54              colorWipereverse(strip, Color(0,0,0))
55            elif ord(data[15]) ==9:
56              colorWipereverse9(strip, Color(255,0,0))
57              colorWipereverse(strip, Color(0,0,0))
```

このプログラムは ICMP パケットを解析し，ping コマンドで使用されるタイプ 8 の ICMP メッセージを受信したときのみ LED テープが動作するようにした。さらに IP アドレスに応じて異なる光の色と群の数を表現するよう IP アドレスを解析し，解析結果に応じて光の色と点灯する LED 群の数が変化するようにした。

4.4.5　ルータの構築

Raspberry Pi は初期設定のままでは，ルータとして異なるネットワークにデータを転送することはできない。Raspberry Pi をルータとして用いるためには「/etc/sysctl.conf」に「net.ipv4.ip_forward=1」の記述を追加することで，IP パケットを異なるネットワークに転送するようになり，ルータとして用いることができる。

また，これだけではコンピュータがどのコンピュータにデータを転送すべきかが分からない。ルータや端末は経路表を参考にし，転送先のコンピュー

タを判断しデータを転送する経路制御を行っている。本教材では，複雑な経路を設定する必要がないため，経路の確定に一定の時間のかかる動的経路制御ではなく，静的に経路制御することにした。IP アドレスを ifconfig ボタンで設定する際に，IP アドレスのネットワーク部を参照し，静的に経路表を変更するようにプログラムした。IP アドレス学習教材 (ホスト教材) とルータ教材の Python プログラムのソースコードを付録 B にそれぞれ記載する。また，ルータ教材の回路図を図 4.4 に示す。

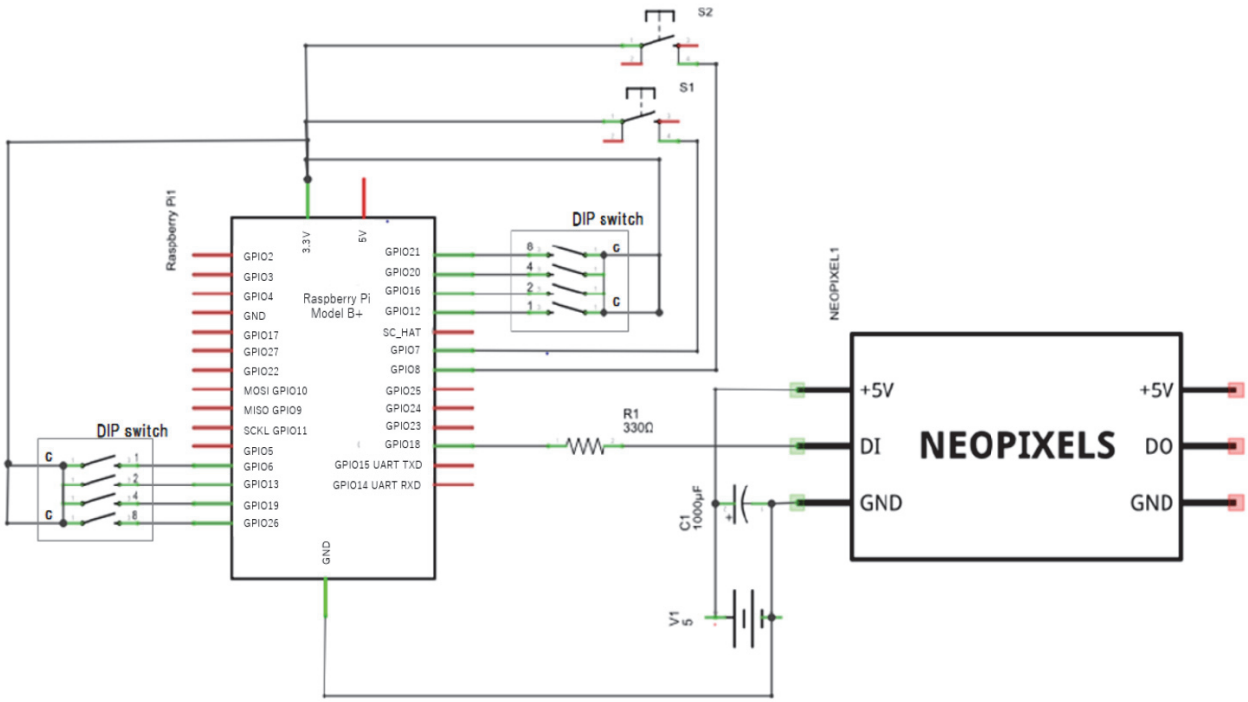

図 4.4　ルータ教材の回路図

4.5　ルータ教材を用いた実験

開発した教材を用いて大学院生を対象に予備的な実験を行い，教材の教育的効果の検証を行った。

4.5.1　実験の概要

実験は端末教材 2 台とルータ教材 1 台を用いて行った。計画した実験の流れは以下のとおりである。

1. 異なる 2 つのネットワークでルータ教材を経由して端末同士が通信するために物理的及び論理的にどのように設定をするか構成図を決定する。
2. 決定したネットワーク図をもとに教材同士を LAN ケーブルで物理的に接続し，送信端末側にルータの LED テープを配置する。
3. IP アドレスの論理的な設定を各教材のボタンとダイヤルで設定し，traceroute ボタンで LED テープの光の軌跡で通信を見ることで，物理的及び論理的に正しく設定できていることを確認する。
4. 間違った LAN ケーブルの配線や，論理的な設定をすると traceroute ボタンを押しても LED が光らないことを確認する。

(1) において想定されるネットワークの簡易的な構成図は図 5 のようになる。また，実験に用いる 2 台の端末教材は同じ教材であり，ifconfig ボタンで正しく設定を行えば入れ替わっても問題なく動作する。

4.5.2　実験結果

実験は情報通信ネットワークについて学習済みである大学院生 1 名を対象に行い，実験後に教材を用いた実験に関してインタビューを行い，教材としての機能の評価を行った。

実験は，まず，教材の詳細を説明し，ネットワーク構築のために必要なネットワーク構成図を考えさせた（前節で記述した実験の流れの (1)）。そして，考えた構成図をもとに LAN ケーブルを用いて物理的配線を行なわせた（実験の流れ（2））。それから，考えた構成図をもとにダイヤルとボタンを用いた論理的な設定を正しく行わせ，traceroute ボタンを押すとフルカラーシ

リアル LED テープが送信側の端末から受信側の端末に流れるように光ることで，通信の物理的可視化による確認を行わせた（実験の流れ（3））。その後，間違った設定を行って確認する実験を行わせた（実験の流れ（4））。被験者は，LAN ケーブルの物理的配線を入れ替えて traceroute ボタンを押しても LED は光らないことを確認し，さらに，論理的な設定を間違った例にした場合など，何度かネットワーク構築実験を行った。ネットワーク構築実験の様子を図 4.5，traceroute ボタンを押し，LED テープが光る様子を図 4.6 に示す。

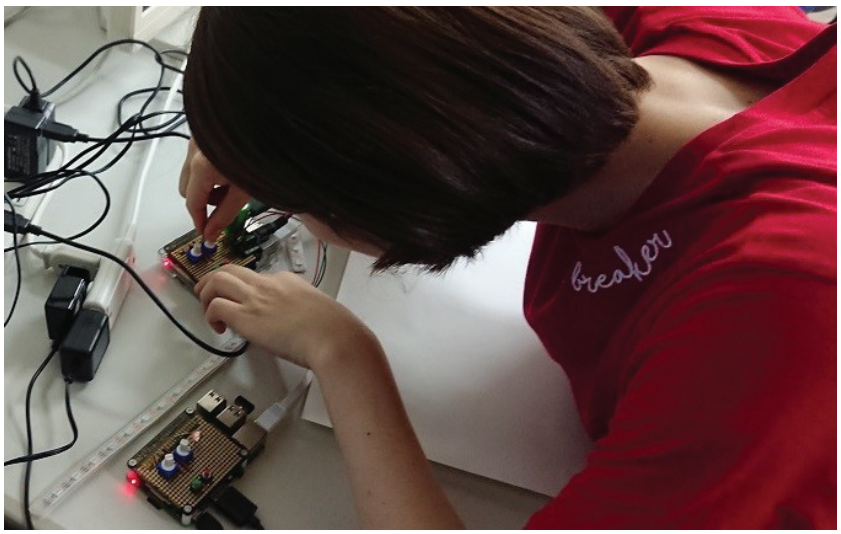

図 4.5　実験の様子

実験後，実験に参加した大学院生に教材に関してインタビューを行った。インタビューの結果を以下に記述する。

実験全体に関しての質問では，「ネットワークの構築を容易に行うにすることができ，興味を持って実験に取り組めた」との回答を得た。ネットワークの構築の難易度に関しての質問では，「物理的な配線も簡単に行うことができた。ダイヤルやボタンの意味さえ分かれば IP アドレスを簡単に設定することができた。」との回答を得た。LED テープによる物理的可視化に関しての質問では，「従来では，スクリーンを見て通信ができているか確認を行うが，LED テープを見るだけで，直感的に通信ができていることが確認できるの

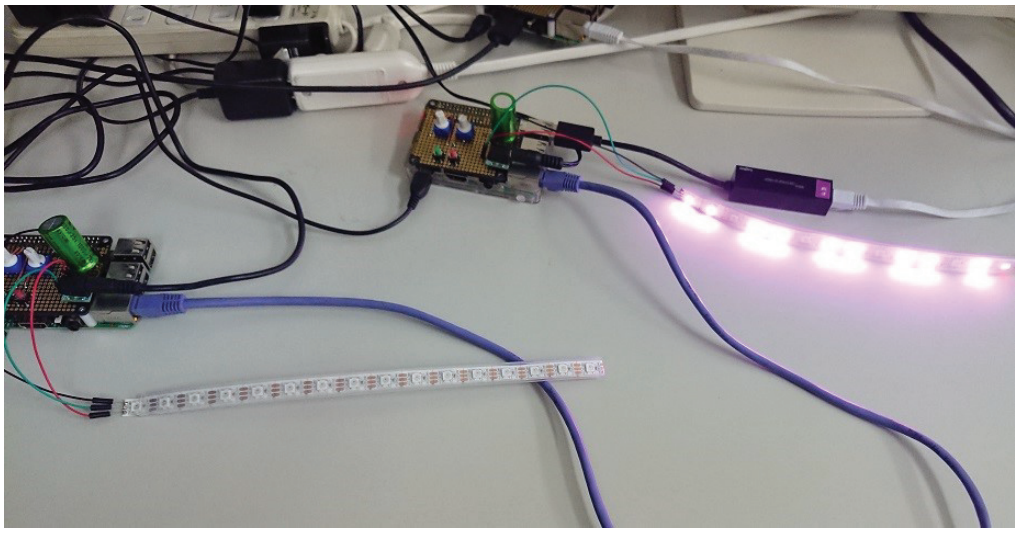

図 4.6　LED テープが光る様子

が良かった。論理的なネットワーク構築の可視化においては，色によるネットワークの違いは分かりやすかったが，ホストアドレスによる群による可視化は直感的には理解しづらかった。」との回答を得た。ダイヤルによる論理的なアドレス設定に関する質問では，「ifconfig や ping などのコマンドを知らなくてもボタンを押すだけで通信の設定や確認が行えることが良かった。また，従来の IP アドレスの設定とは異なり，0〜9 までの 10 通りのみでシンプルにネットワークアドレスとホストアドレスを指定することができ，設定しやすかった。また，玩具で遊ぶような感覚で設定できるのも良かった。」との回答を得た。実験を通して情報通信ネットワークや TCP/IP に興味を持てたかどうかについての質問では，「様々なネットワークの構築が何度も容易に行えるので，このような設定ならどうなるだろうと様々な考察をしながら興味を持って進んで実験して確認することができた。」との回答を得た。

これらの実験インタビュー結果により，本研究で開発したルータ教材が，LED テープの光の軌跡による物理的可視化やダイヤルやボタンを用いた物理的直接操作によって直感的に情報通信ネットワークを理解できる教材として有効に機能することが確認できた。しかしながら，ホストアドレスの LED 群による物理的可視化などが理解しづらいなどの問題点も明らかになった。

4.6　本章のまとめ

本研究では，物理的可視化や物理的直接操作による情報通信ネットワークを学習するためのルータ教材を開発し，教材の機能を確認した。本教材は，現状では見ることのできない情報通信ネットワークを LED テープの光の軌跡で物理的に可視化し，ダイヤルとボタンを用いた操作で IP アドレスの設定を何度も行い確認することができる。教材を用いた実験を通して，直感的に情報通信ネットワークの機能や動作を理解することができ，情報通信ネットワークを実践的・体験的に学習することが確認できた。　今後は，明らかになった問題点を解決するために教材の改良を行い，本教材を用いた授業案を考案し，中学校や高等学校で授業実践を行うことで，本教材の学習の有効性を評価する必要がある。

第 5 章

本研究の位置

5.1　小型デバイスを用いた教材の開発の視点から

教育工学では，様々なハードウェアやソフトウェアを用いた教材・教具の開発研究が行われている。そのような中，小型デバイスを用いた教育支援システムの研究が盛んに行われている [25][26][27][28][29]。小型デバイスは，情報処理演習室で用いるようなコンピュータより安価で入手でき，持ち運びが可能なため学校教育での利用もしやすい。しかしながら，小型デバイス機器を利用した教育システムやツールの多くは，ソフトウェア環境を構築するために用いたり，他のデバイスとの通信用に用いたりするものが多く，小型デバイスのみで完結する教育システムやツールは多くない。

それに対して本研究で開発した教材は，Raspberry Pi と Raspberry Pi 用の基板で一体となって構成されている。。そのため，実験にあたり，マウスやキーボード，マウスなどの周辺機器を準備する必要がなく，開発した教材と通信を行うために必要な LAN ケーブルのみで教材を用いた実験を行うことができる。

5.2　情報通信ネットワークの学習教材の視点から

情報通信ネットワークの学習のための教材開発研究は多く行われている [12][13][14][15]。その中の多くは，汎用コンピュータを用いた教育システムが多く，通信は仮想化され，モニタ上で通信の様子を可視化し，IP アドレスの設定などは CUI もしくは GUI 上でキーボードやマウスを用いて行われる。中学生や高校生などの，コンピュータの操作に不慣れな学習者にとっては，これらの操作は敷居が高いと考えられる。

本研究で開発した教材の IP アドレスの設定や通信の確認には，より直感的に操作しやすいダイヤルやボタンを用いるという新しい手法を選択した。これにより，ダイヤルを回す操作と，ボタンを押すという単純な操作のみで IP アドレスの設定や通信の確認を行うことができる。

5.3　技術・情報教育の視点から

中学校技術・家庭科技術分野の学習指導要領において，技術分野の目標として「ものづくり」などの技術に関する実践的・体験的な活動を通して学習することが望ましいとされている。情報通信ネットワークにおける「ものづくり」とは，ネットワークの構築である。

本研究で開発した教材は，IP アドレスの学習やルータの役割を，ネットワークの構築を行うことが学習することができ，情報通信ネットワークの「ものづくり」を通して学習することが可能である。

5.4　情報通信ネットワーク技術者の視点から

総務省が発表している平成 29 年版情報通信白書によると，情報通信職種の従業者は 5800 人程度と低い割合となっている [30]。ICT 人材は，システムエンジニアやプログラマを中心に情報通信関連の人手不足が生じており，今後はセキュリティ関連の人材不足が深刻化する見通しである。

本研究で開発した教材は，中等教育を対象にした情報通信ネットワークを学習する教材であり，情報通信ネットワーク技術者を対象にしたものではない。しかしながら，第 1 章で論じた IP アドレス学習教材の実践評価において，情報通信ネットワークを構築する実験を通して情報通信ネットワークに興味を持った生徒も多くみられた。キャリア選択を行う前の中等教育段階で情報通信ネットワークに興味を持つことで，進路選択として情報通信職種を選択する生徒の増加が見込まれ，ICT 人材不足の解消に寄与することが期待できると考えられる。

5.5　教育手法の視点から

学習における教育手法は様々な手段が取られている。特に技術・情報分野では，学習内容がブラックボックス化されており，学習者にとって学習内容が目に見えず，理解が難しいものが多い。その中で，可視化やシンプルな操作による手段が多く取られている [31][32][33]。

しかしながら，我々が提唱している物理的可視化や物理的直接操作などの物理的手法では，ネットワークの動きやはたらきを物理的な現象を用いて表現したり，“ダイヤル” を回したり “ボタン” を押すという単純な動作で情報通信ネットワークの設定や疎通の確認を実行するものである。これらの手法は今まで考案されておらず，物理的手法で表現することで直感的に理解することができる。これらの表現手法は，情報通信ネットワークだけではなく，技術・情報分野におけるブラックボックス化された技術など多くの学習内容で重要な手法であると考えられる。

5.6　インタフェースの視点から

コンピュータと人間との情報をやり取りするためのユーザインタフェースでは，多くのシステムやコンピュータを用いた教材では，GUI が用いられている。GUI は，汎用性に優れており，マウスやキーボードなどの入力機器を用いディスプレイにグラフィカルな出力を提示することで様々な情報を取り

扱うことができる。しかしながら，特定の情報のみをグラフィカルユーザインタフェースで取り扱う場合，不必要な情報が多く，入力に必要となる労力や出力結果を理解するのにかかる労力が多くかかる可能性が高い。

本研究で提唱している物理的可視化や物理的直接操作を用いた物理的インタフェースでは，入力にボタンやダイヤルを用い，ボタンを「押す」動作やダイヤルを「回す」といった単純な動作のみで操作を実行することができ，出力結果の確認には LED の光を用い，光を「見る」という単純な動作で確認することができる。物理的インタフェースを用いることで，入力に必要となる労力や出力結果を理解するのにかかる労力を抑えることができ，使い方の学習にかかる労力が非常に少なく直感的に扱うことができ，効率的に学習をすすめることができる。

第 6 章

終章

6.1　本論文のまとめ

本論文では「物理的表現手法による技術・情報分野の教材開発およびその評価に関する研究」をテーマとして，物理的可視化と物理的直接操作をコンセプトに技術・情報分野における情報通信ネットワークを学習するための教材開発の研究として（1）IP アドレス学習教材の開発とその評価と，（2）ネットワーク構築を学習するためのルータ教材の開発を行った。

本論文の第 3 章では，IP アドレスを学習するための教材の開発とその評価について述べた。従来の情報通信ネットワークの学習教材では，汎用コンピュータによるモニタを用いた可視化や，キーボードやマウスの操作による設定が中心であり，中等教育においては敷居の高いものであった。本研究で開発した IP アドレス学習教材は，通信の確認に LED を用い，IP アドレスの設定や通信の実行にダイヤルやボタンを用いることで，直感的に操作をし通信の確認を行うことができる。IP アドレス学習教材により IP アドレスを効果的に学習することができ，ネットワークを構築する実験を行うことができる。中学校 1 年生を対象に教材を用いた授業実践を行い，教材の評価を行った。授業対象者を実験群と統制群に分け，事前テスト・事後テストを行った結果，事後テストでは，実験群の方が統制群より得点が高い傾向にあり，有意差が見られたことから，教材の有効性が確認できた。

本論文の第 4 章では，第 2 章で開発した IP アドレス学習教材を発展させ，ネットワーク構築を学習するためのルータ教材を開発した。ルータ教材は第 2 章で開発した IP アドレス学習教材をルータ化し，ルータを介した通信の流れをフルカラーシリアル LED テープを用いて物理的可視化することにより実現した。この教材により，異なるネットワーク間の通信に必要なルータの役割を学習することができ，複数のネットワークを構築する実験を行うことができる。大学院生 1 名を対象に予備的な実践実験を行い，実験後に教材を用いた実験に関してインタビューを行い，教材としての機能の評価を行った。結果として，直感的にルータの動作や役割を理解することができ，情報通信ネットワークの構築を実践的・体験的に学習することが確認できた。

6.2　今後の展望

本論文で述べた開発した 2 つの教材は， IP アドレスの仕組みを学習する教材とルータの役割を学習する教材である。

しかしながら，ネットワークを支える技術は，IP アドレスやルータだけにとどまらず，多岐にわたっており，例えば経路制御など，様々な技術が利用されており，情報通信ネットワークを学習するためには，多くの内容を学習する必要がある。今後は，今までの学習内容に加え，様々な物理的表現手法を用いた情報通信ネットワークに関する学習教材を開発し，情報通信ネットワークの技術をより体系的に学習できるような学習モデルへ発展させて行くことが課題である。

また，物理的可視化や物理的直接操作は，第 2 章で述べたように，様々な技術科や情報科の内容において教材作成のコンセプトとして有用であると考えられる。そして，技術科や情報科では，多くの分野にわたる学習内容が含まれている。今後は，ネットワークだけではなく，他の学習内容においても物理的可視化と物理的直接操作のコンセプトを基に教材開発をしていき，教材を評価することによって物理的可視化や物理的直接操作の有用性を検証していく必要がある。

謝辞

本研究を進めるにあたり，多くの御指導を頂きました広島大学大学院教育学研究科技術・情報教育学講座 渡辺健次 教授に厚く御礼申し上げます。また，本論文をご精読頂き，多くの有用な助言をいただきました広島大学大学院教育学研究科技術・情報教育学講座 田中秀幸 教授，藤中透 教授，長松正康 教授に厚く御礼申し上げます。そして，多くの有用な助言をいただきました近畿大学理工学部情報学科 井口信和 教授に厚く御礼申し上げます。第 3 章における授業実践に協力していただいた広島大学附属東雲中学校 堤健人 教諭に深く御礼申し上げます。そして，第 4 章の教材開発において本研究に関して多くの知識や示唆を頂いた広島大学教育学部第二類文化教育系技術・情報系コース（現，広島市役所）入田雄基 氏に御礼申し上げます。また，実験に協力し，開発した教材の評価をしていただいた広島大学大学院教育学研究科博士課程前期教科教育学専攻技術・情報教育学専修 石川有彩 氏に御礼申し上げます。

本論文の第 3 章は，日本産業技術教育学会誌第 60 巻第 2 号に掲載された「物理的可視化と物理的直接操作による IP アドレスの仕組みを学習するための教材の開発と評価」の内容を詳しく述べたものである。

参考文献

[1] 総務省：通信利用動向調査，http://www.soumu.go.jp/johotsusintokei/statistics/data/180525_1.pdf（2020 年 1 月 7 日アクセス）

[2] 内閣府：Society5.0, <https://www8.cao.go.jp/cstp/society5_0/index.html（2020 年 1 月 7 日アクセス）

[3] 文部科学省：Socity5.0 に向けた人材育成～社会が変わる、学びが変わる～,http://www.mext.go.jp/component/a_menu/other/detail/__icsFiles/afieldfile/2018/06/06/1405844_002.pdf（2020 年 1 月 7 日アクセス）

[4] 経済産業省：IT 人材の最新動向と将来推計に関する調査結果,https://www.meti.go.jp/policy/it_policy/jinzai/27FY/ITjinzai_report_summary.pdf(2020 年 1 月 7 日アクセス)

[5] 文部科学省 (2018)：中学校学習指導要領（平成 29 年告示）解説技術・家庭編，開隆堂出版

[6] 文部科学省 (2019)：高等学校学習指導要領（平成 30 年告示）解説情報編，開隆堂出版

[7] 吉原和明・井口信和・渡辺健次：物理的可視化と物理的直接操作による IP アドレスの仕組みを学習するための教材の開発と評価，日本産業技術教育学会誌第 60 巻第 2 号，pp.73-80(2018)

[8] 吉原和明：物理的可視化と物理的直接操作によるネットワーク構築を学習するためのルータ教材の開発，広島大学大学院教育学研究科紀要第二

部（文化教育開発関連領域）第 68 号，2019 年 12 月採択済

[9] 筑波大学 : 筑波大学シラバス「2019 年度 3 年次実験コンピュータネットワーク実験」，http://www.u.tsukuba.ac.jp/~kimura.shigetomo.gb/netexp/netexp.html（2020 年 1 月 7 日アクセス）

[10] 佐賀大学 : 佐賀大学シラバス「情報ネットワーク実験」，http://syllabus.sc.admin.saga-u.ac.jp/（2020 年 1 月 7 日アクセス）

[11] 江馬諭・木村泰樹 : ものづくり学習のための教材「銅鏡製作」について，日本産業技術教育学会誌, 第 47 巻，第 1 号，pp.31-37 (2005)

[12] 川西千晶・今井慈郎：ネットワーク学習支援のための経路制御可視化アプリケーション開発，電子情報通信学会技術研究報告，第 110 巻，第 453 号，pp. 181-186 (2011)

[13] 立岩佑一郎・安田孝美・横井茂樹：仮想環境ソフトウェアに基づくネットワーク処理可視化教育システムの開発，情報処理学会研究報告コンピュータと教育，pp.7-14 (2012)

[14] 荒井正之・田村尚也・渡辺博芳・他 : TCP/IP プロトコル学習ツールの開発と評価，情報処理学会論文誌，第 44 巻，第 12 号，pp.3242-3251（2003）

[15] 村松浩幸・本多満正・坂口謙一・他：中学校技術科でのネットワーク学習における電話網の教材化，日本教育工学会論文誌，第 28 巻，suppl. 号，pp.237-240(2005)

[16] 田口浩継・他：新しい技術・家庭技術分野：東京書籍，p.205(2016)

[17] 日本科学未来館 : インターネット物理モデル，http://www.miraikan.jst.go.jp/exhibition/future/information/internet.html（2020 年 1 月 7 日アクセス）

[18] RASPBERRY PI FOUNDATION : Raspberry Pi，https://www.raspberrypi.org/（2020 年 1 月 7 日アクセス）

[19] pythonJapan「プロセス間通信とネットワーク」，http://docs.python.jp/2/library/socket.html（2020 年 1 月 7 日アクセス)

[20] 秋月電子通商「DIP Rotary Switch_10」, http://akizukidenshi.com/

download/DIP%20Rotary%20Switch_10.pdf(2020 年 1 月 7 日アクセス)

[21] 精鷹幹人・木村昌史：教育向けネットワークシミュレータの開発，情報処理学会 65 回全国大会，pp.273-274(2003)

[22] 井口信和：仮想ルータを活用したネットワーク構築演習支援システムの開発，情報処理学会論文誌第 52 巻第 3 号，pp.1412-1423(2011)

[23] 本多満正・水谷浩紀・菅家久貴：中学校技術科における情報通信ネットワークの仕組み教材化とその評価，日本教材学会教材学研究 19，pp.195-202(2008)

[24] SWITCHSCIENCE「フルカラーシリアル LED テープ」，https://www.switch-science.com/catalog/1399/（2020 年 1 月 7 日アクセス）

[25] 柴田幸司・花田一麿・飯野真弘・武美里・赤塚優磨：Linux マイコンを用いた VPN による小型センサ情報遠隔監視システムの開発と教育への応用，電子情報通信学会技術研究報告 IEICE 信学技報第 114 巻第 441 号，pp.63-68(2015)

[26] 吉田智子・中村亮太・松浦敏雄：「プログラムによる計測と制御」を学ぶための学習環境の開発と教育実践～LilyPad Arduino シミュレータ機能付 PEN を利用して～，情報処理学会研究報告コンピュータと教育，pp.1-10(2015)

[27] 鴻池泰元・中西通雄：IchigoJam 用ビジュアルブロックプログラミング環境の開発とプログラミング体験教室の実践，情報処理学会研究報告コンピュータと教育，pp.1-8(2018)

[28] 細合晋太郎・石田繁巳・亀井靖高・鵜林尚靖・福田晃細：自律走行ロボットを用いた IoT 開発 PBL に向けた教材開発，組込みシステムシンポジウム 2015 論文集，pp.40-45(2015)

[29] 竹田慎・曽根直人：センササーバと ScratchX による計測システムの開発，鳴門教育大学情報教育ジャーナル第 14 巻，pp.41-47(2017)

[30] 総務省：平成 29 年版情報通信白書 (2017)

[31] 内海能亜・吉田昌史：塑性加工を学ぶ教材としての小型プレス機械の開発，工学教育第 60 巻第 4 号，pp.109-113(2012)

[32] 青井中央人・伊藤陽介・谷陽子：電波を題材とする技術教育の開発と評価，日本産業技術教育学会誌第 55 巻第 3 号，pp.199-206(2013)

[33] 今井慈郎・森藤義雄・堀幸雄・林敏浩・井面仁志・白木渡：計算機内部の構造・動作を可視化する計算機シミュレータの評価と強調学習環境での利用効果，電子情報通信学会技術研究報告 ET 教育工学第 109 巻第 335 号，pp.215-220(2009)

付録 A

Raspberry Pi のインストール・設定

A.1　Raspberry Pi のインストール

ここでは，Raspberry Pi に OS をインストールする方法を説明する。Raspberry Pi は microSD メモリーカードから OS を読み込んで起動するので，microSD メモリーカードに OS のイメージファイルを書き込む必要がある。書き込む PC の環境は Windows10 とする。

準備するものとして OS Raspbian のイメージファイルと，イメージファイルを書き込むソフトウェアである “Win32 DiskImager” が必要となる。

1. Win32 DiskImager を起動。
2. ImageFile“jessie” を選択。
3. 書き込むデバイスに SD ファイルを選択。
4.「Write」を選択してイメージファイルを OS に書き込む。

書き込んだ SD カードを Raspberry Pi に差し込んで起動すればよい。

A.2　Raspberry Pi の初期設定

ここでは，Raspbian の初期設定の方法を説明する。

初めて起動する際は，HDMI ポートとディスプレイを，また，USB ポートとキーボードを接続しておく必要がある。

起動するとコンフィギュレーション画面が出るので，キーボードの上下の矢印キーで選択し，Enter キーで決定し，各種メニューの設定を行う。初期設定では，SD メモリーカード領域を拡張する設定と，タイムゾーンとキーボードの設定，SSH 接続を可能な状態にする設定を行う。

1. Raspberry Pi マークのアイコンをクリック→ Preference → Raspberry Pi Configuration を選択，Raspi-config を起動。
2. system タブを開き，「Expand Filesystem」をチェックする。
3. Localisation タブを開き, Locale は「Language:ja, Country:JP, Character Set:UTF-8」を選択，Set Timezone は「Japan」を選択，Set Keyboard は，日本語キーボードの場合は Japanese を選択。
4. Interfaces タブを開き，SSH 欄の「Enabele」にチェック。
5. すべて設定し終えたら「OK」→「Yes」で再起動する。

再起動すると，ログイン画面が出てくるので，ユーザ名 “pi” パスワード “raspberry” を入力しシェルを起動させ，以下のコマンドを打ち root パスワードを設定する。

root パスワードの設定

```
sudo passwd root
```

これで root になることができる。

A.3　端末・ルータの設定

A.3.1　IP アドレスの設定

Raspberry Pi は初期設定では IP アドレスは DHCP で設定してあるので，本教材を利用するにあたって静的に設定する必要がある。

IP アドレスを静的に設定するには，/etc/dhcpcd.conf を以下の内容を追記する必要がある。

/etc/dhcpcd.conf に追記

```
interface eth0
static ip_ddress=192.168.1.2/24
static routers=192.168.1.1
static domain_name_servers=192.168.1.1
```

上記の場合は，IP アドレスを 192.168.1.2/24 と設定した。ルータや DNS サーバの設定は各自の環境に合わせて変更する必要がある。また，SSH で Raspberry Pi にログインすることも考えて，設定した IP アドレスなどの設定情報は，Raspberry Pi 以外の記憶媒体に保存しておくとよい。

ルータの場合，各インターフェースで同様の設定をする必要がある。その場合，それぞれの IP アドレスが同一ネットワークにならないように注意が必要である。

A.3.2　ルータの設定

Raspberry Pi をルータにするには，/etc/sysctl.conf を以下のように書き換える必要がある。

- “net.ipv4.ip_forward=1” のコメントアウトを外す。

これにより，Raspberry Pi が異なるネットワークに IP パケットを転送することができる。

A.3.3　dhcpcd サービスの停止

IP アドレス学習教材やルータ教材は静的な経路制御を行うため，初期状態で起動している dhcpcd サービスを停止しておく必要がある。停止するためには，ターミナル上で下記のコマンドを入力しておく。

dhcpcd サービスの停止

```
sudo systemctl stop dhcpcd
```

付録 B

プログラム・ソースコード

第 3 章で述べた IP アドレス学習教材（ホスト教材）の Python プログラムのソースコードを記載する。

```
1  #! /usr/bin/env python
2
3  import RPi.GPIO as GPIO
4  import time
5  import os
6  import socket
7
8
9  GPIO.setmode(GPIO.BCM)
10
11 GPIO.setup(4,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
12 GPIO.setup(17,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
13 GPIO.setup(22,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
14 GPIO.setup(10,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
15 GPIO.setup(9,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
16 GPIO.setup(11,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
17 GPIO.setup(14,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
18 GPIO.setup(15,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
19 GPIO.setup(18,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
20 GPIO.setup(23,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
21 GPIO.setup(24,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
22 GPIO.setup(25,GPIO.OUT)
23
24 def number():
```

```
25    n = 0
26    h = 0
27    inputn1 = GPIO.input(4)
28    inputn2 = GPIO.input(17)
29    inputn4 = GPIO.input(22)
30    inputn8 = GPIO.input(10)
31    inputh1 = GPIO.input(14)
32    inputh2 = GPIO.input(15)
33    inputh4 = GPIO.input(18)
34    inputh8 = GPIO.input(23)
35    if inputn1 == True:
36      n = n + 1
37    if inputn2 == True:
38      n = n + 2
39    if inputn4 == True:
40      n = n + 4
41    if inputn8 == True:
42      n = n + 8
43    if inputh1 == True:
44      h = h + 1
45    if inputn2 == True:
46      h = h + 2
47    if inputn4 == True:
48      h = h + 4
49    if inputn8 == True:
50      h = h + 8
51    return [n, h]
52
53
54  while True:
55    input1 = GPIO.input(9)
56    input2 = GPIO.input(11)
57    input3 = GPIO.input(24)
58    if input1 == True:
59      network = (str(n))
60      host    = (str(h))
61      ping = "ping␣192.168." + network + "." + host + "␣-c␣1"
62      os.system(str(ping))
63      while input1 == True:
64        input1 = GPIO.input(9)
65
66
```

```
67    elif input2 == True:
68      n ,h = def number()
69      network = (str(n))
70      host    = (str(h))
71      address ="ifconfig␣eth0␣192.168." + network + "." + host
            + "␣netmask␣255.255.255.0"
72      os.system(str(address))
73      while input2 == True:
74        input2 = GPIO.input(11)
75
76
77    else:
78      data = 0
79      sock =socket .socket(socket.AF_INET, socket.SOCK_RAW,
            socket.IPPROTO_ICMP)
80      sock.settimeout(0.1)
81      try:
82        data =sock.recv(255)
83        if data:
84          GPIO.output(25,True)
85          time.sleep(0.3)
86          GPIO.output(25,False)
87      except:
88        pass
```

次に，第 4 章で述べたルータ学習教材の Python プログラムのソースコードを 2 つ記載する。

まずは，ホスト用の教材のプログラムのソースコードを記載する。

```
1  #! /usr/bin/env/ python
2
3  import RPi.GPIO as GPIO
4  import time
5  import sys
6  import os
7  import socket
8  from neopixel import *
9
10 GPIO.setmode(GPIO.BCM)
11
12 GPIO.setup(4,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
13 GPIO.setup(17,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
```

```
14 GPIO.setup(22,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
15 GPIO.setup(10,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
16 GPIO.setup(9,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
17 GPIO.setup(11,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
18 GPIO.setup(14,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
19 GPIO.setup(15,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
20 GPIO.setup(18,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
21 GPIO.setup(23,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
22 GPIO.setup(24,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
23 GPIO.setup(27,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
24
25 LED_COUNT       = 16
26 LED_GPIO        = 13
27 LED_CHANNEL     = 1
28 LED_FREQ_HZ     = 800000
29 LED_DMA         = 10
30 LED_BRIGHTNESS  = 150
31 LED_INVERT      = False
32 network1        = None
33
34 if __name__ == '__main__':
35   strip = Adafruit_NeoPixel(LED_COUNT, LED_GPIO, LED_FREQ_HZ,
         LED_DMA, LED_INVERT, LED_BRIGHTNESS,LED_CHANNEL)
36   strip.begin()
37   print "LEDready"
38
39 def number():
40   n = 0
41   h = 0
42   inputn1 = GPIO.input(4)
43   inputn2 = GPIO.input(14)
44   inputn4 = GPIO.input(15)
45   inputn8 = GPIO.input(17)
46   inputh1 = GPIO.input(9)
47   inputh2 = GPIO.input(27)
48   inputh4 = GPIO.input(22)
49   inputh8 = GPIO.input(11)
50   if inputn1 == True:
51     n = n + 1
52   if inputn2 == True:
53     n = n + 2
54   if inputn4 == True:
```

```
55      n = n + 4
56    if inputn8 == True:
57      n = n + 8
58    if inputh1 == True:
59      h = h + 1
60    if inputh2 == True:
61      h = h + 2
62    if inputh4 == True:
63      h = h + 4
64    if inputh8 == True:
65      h = h + 8
66    return [n, h]
67
68  def colorWipe(strip, color, wait_ms=10):
69    for i  in range(strip.numPixels()):
70      strip.setPixelColor(i, color)
71      strip.show()
72      time.sleep(wait_ms/1000.0)
73
74  def colorWipe1(strip, color, wait_ms=10):
75    for q in range(1):
76      for i  in range(0,strip.numPixels(),2):
77        strip.setPixelColor(i+q, color)
78        strip.show()
79        time.sleep(wait_ms/1000,0)
80
81  def colorWipe2(strip, color, wait_ms=10):
82    for q in range(2):
83      for i  in range(0,strip.numPixels(),3):
84        strip.setPixelColor(i+q, color)
85        strip.show()
86        time.sleep(wait_ms/1000,0)
87
88  def colorWipe3(strip, color, wait_ms=10):
89    for q in range(3):
90      for i  in range(0,strip.numPixels(),4):
91        strip.setPixelColor(i+q, color)
92        strip.show()
93        time.sleep(wait_ms/1000,0)
94
95  def colorWipe4(strip, color, wait_ms=10):
96    for q in range(4):
```

```
97      for i  in range(0,strip.numPixels(),5):
98        strip.setPixelColor(i+q, color)
99        strip.show()
100       time.sleep(wait_ms/1000,0)
101
102 def colorWipe5(strip, color, wait_ms=10):
103   for q in range(5):
104     for i  in range(0,strip.numPixels(),6):
105       strip.setPixelColor(i+q, color)
106       strip.show()
107       time.sleep(wait_ms/1000,0)
108
109 def colorWipe6(strip, color, wait_ms=10):
110   for q in range(6):
111     for i  in range(0,strip.numPixels(),7):
112       strip.setPixelColor(i+q, color)
113       strip.show()
114       time.sleep(wait_ms/1000,0)
115
116 def colorWipe7(strip, color, wait_ms=10):
117   for q in range(7):
118     for i  in range(0,strip.numPixels(),8):
119       strip.setPixelColor(i+q, color)
120       strip.show()
121       time.sleep(wait_ms/1000,0)
122
123 def colorWipe9(strip, color, wait_ms=10):
124   for q in range(9):
125     for i  in range(0,strip.numPixels(),10):
126       strip.setPixelColor(i+q, color)
127       strip.show()
128       time.sleep(wait_ms/1000,0)
129
130 def colorWipereverse(strip, color, wait_ms=10):
131   for i  in range(strip.numPixels(),-1,-1):
132     strip.setPixelColor(i, color)
133     strip.show()
134     time.sleep(wait_ms/1000.0)
135
136 def colorWipereverse1(strip, color, wait_ms=10):
137   for q in reversed(range(1)):
138     for i in range(0,strip.numPixels(),2):
```

```
139       strip.setPixelColor(i+q, color)
140       strip.show()
141       time.sleep(wait_ms/1000.0)
142
143 def colorWipereverse2(strip, color, wait_ms=10):
144   for q in reversed(range(2)):
145     for i in range(0,strip.numPixels(),3):
146       strip.setPixelColor(i+q, color)
147       strip.show()
148       time.sleep(wait_ms/1000.0)
149
150 def colorWipereverse3(strip, color, wait_ms=10):
151   for q in reversed(range(3)):
152     for i in range(0,strip.numPixels(),4):
153       strip.setPixelColor(i+q, color)
154       strip.show()
155       time.sleep(wait_ms/1000.0)
156
157 def colorWipereverse4(strip, color, wait_ms=10):
158   for q in reversed(range(4)):
159     for i in range(0,strip.numPixels(),5):
160       strip.setPixelColor(i+q, color)
161       strip.show()
162       time.sleep(wait_ms/1000.0)
163
164 def colorWipereverse5(strip, color, wait_ms=10):
165   for q in reversed(range(5)):
166     for i in range(0,strip.numPixels(),6):
167       strip.setPixelColor(i+q, color)
168       strip.show()
169       time.sleep(wait_ms/1000.0)
170
171 def colorWipereverse6(strip, color, wait_ms=10):
172   for q in reversed(range(6)):
173     for i in range(0,strip.numPixels(),7):
174       strip.setPixelColor(i+q, color)
175       strip.show()
176       time.sleep(wait_ms/1000.0)
177
178 def colorWipereverse7(strip, color, wait_ms=10):
179   for q in reversed(range(7)):
180     for i in range(0,strip.numPixels(),8):
```

```
181       strip.setPixelColor(i+q, color)
182       strip.show()
183       time.sleep(wait_ms/1000.0)
184
185 def colorWipereverse8(strip, color, wait_ms=10):
186   for q in reversed(range(8)):
187     for i in range(0,strip.numPixels(),9):
188       strip.setPixelColor(i+q, color)
189       strip.show()
190       time.sleep(wait_ms/1000.0)
191
192 def colorWipereverse9(strip, color, wait_ms=10):
193   for q in reversed(range(9)):
194     for i in range(0,strip.numPixels(),10):
195       strip.setPixelColor(i+q, color)
196       strip.show()
197       time.sleep(wait_ms/1000.0)
198
199 while True:
200   input1 = GPIO.input(23)
201   input2 = GPIO.input(18)
202   if input2 == True:
203     n ,h = number()
204     network1 = (str(n))
205     host1    = (str(h))
206     address  = "ifconfig␣eth0␣192.168." + network1 + "." +
          host1 + "␣netmask␣255.255.255.0"
207     routeadd = "route␣add␣default␣gw␣192.168." + network1 + "
          .254"
208     print address
209     os.system(str(address))
210     time.sleep(2)
211     print routeadd
212     os.system(str(routeadd))
213     while input2 == True:
214       input2 = GPIO.input(18)
215
216   if input1 == True:
217     n ,h = number()
218     network = (str(n))
219     host    = (str(h))
220     dest_addr = "192.168." + network + "." + host
```

```
221     print dest_addr
222     socket.setdefaulttimeout(10)
223     icmp = socket.getprotobyname('icmp')
224     udp =  socket.getprotobyname('udp')
225     ttl = 1
226     max_hops = 4
227     port = 33434
228     while True:
229      recv_socket = socket.socket(socket.AF_INET,socket.
            SOCK_RAW, icmp)
230      send_socket = socket.socket(socket.AF_INET,socket.
            SOCK_DGRAM, udp)
231      send_socket.setsockopt(socket.SOL_IP, socket.IP_TTL, ttl
            )
232      recv_socket.bind(("",port))
233      try:
234        send_socket.sendto("",(dest_addr, port))
235      except socket.error:
236        pass
237      curr_addr = None
238      try:
239        _, curr_addr = recv_socket.recvfrom(512)
240        curr_addr = curr_addr[0]
241      except socket.error:
242        pass
243      finally:
244        send_socket.close()
245        recv_socket.close()
246      if curr_addr is not None:
247        trace = "ping␣" + curr_addr + "␣-c␣1"
248        os.system(str(trace))
249        time.sleep(1)
250      ttl +=1
251      print curr_addr
252      print dest_addr
253      if curr_addr == dest_addr or ttl > max_hops or curr_addr
             is None:
254        time.sleep(0.3)
255        break
256   else:
257     data = 0
```

```
258     sock = socket.socket(socket.AF_INET,socket.SOCK_RAW,
            socket.IPPROTO_ICMP)
259     sock.settimeout(0.1)
260     try:
261       data = sock.recv(255)
262       if data:
263         time.sleep(0.28)
264         if ord(data[20]) == 8:
265           if ord(data[12]) == 192:
266             if ord(data[13]) == 168:
267               if ord(data[14]) == 0:
268                 if ord(data[15]) == 0:
269                   colorWipereverse(strip, Color(0,85,255))
270                   colorWipereverse(strip, Color(0,0,0))
271                 elif ord(data[15]) == 1:
272                   colorWipereverse1(strip, Color(0,85,255))
273                   colorWipereverse(strip, Color(0,0,0))
274                 elif ord(data[15]) == 2:
275                   colorWipereverse2(strip, Color(0,85,255))
276                   colorWipereverse(strip, Color(0,0,0))
277                 elif ord(data[15]) == 3:
278                   colorWipereverse3(strip, Color(0,0,255))
279                   colorWipereverse(strip, Color(0,0,0))
280                 elif ord(data[15]) == 4:
281                   colorWipereverse4(strip, Color(0,85,255))
282                   colorWipereverse(strip, Color(0,0,0))
283                 elif ord(data[15]) == 5:
284                   colorWipereverse5(strip, Color(0,85,255))
285                   colorWipereverse(strip, Color(0,0,0))
286                 elif ord(data[15]) == 6:
287                   colorWipereverse6(strip, Color(0,85,255))
288                   colorWipereverse(strip, Color(0,0,0))
289                 elif ord(data[15]) == 7:
290                   colorWipereverse7(strip, Color(0,85,255))
291                   colorWipereverse(strip, Color(0,0,0))
292                 elif ord(data[15]) == 8:
293                   colorWipereverse8(strip, Color(0,85,255))
294                   colorWipereverse(strip, Color(0,0,0))
295                 elif ord(data[15]) == 9:
296                   colorWipereverse9(strip, Color(0,85,255))
297                   colorWipereverse(strip, Color(0,0,0))
298
```

```
299                 elif ord(data[14]) == 1:
300                   if ord(data[15]) == 0:
301                     colorWipereverse(strip, Color(255,0,0))
302                     colorWipereverse(strip, Color(0,0,0))
303                   elif ord(data[15]) == 1:
304                     colorWipereverse1(strip, Color(255,0,0))
305                     colorWipereverse(strip, Color(0,0,0))
306                   elif ord(data[15]) == 2:
307                     colorWipereverse2(strip, Color(255,0,0))
308                     colorWipereverse(strip, Color(0,0,0))
309                   elif ord(data[15]) == 2:
310                     colorWipereverse2(strip, Color(255,0,0))
311                     colorWipereverse(strip, Color(0,0,0))
312                   elif ord(data[15]) == 3:
313                     colorWipereverse3(strip, Color(255,0,0))
314                     colorWipereverse(strip, Color(0,0,0))
315                   elif ord(data[15]) == 4:
316                     colorWipereverse4(strip, Color(255,0,0))
317                     colorWipereverse(strip, Color(0,0,0))
318                   elif ord(data[15]) == 5:
319                     colorWipereverse5(strip, Color(255,0,0))
320                     colorWipereverse(strip, Color(0,0,0))
321                   elif ord(data[15]) == 6:
322                     colorWipereverse6(strip, Color(255,0,0))
323                     colorWipereverse(strip, Color(0,0,0))
324                   elif ord(data[15]) == 7:
325                     colorWipereverse7(strip, Color(255,0,0))
326                     colorWipereverse(strip, Color(0,0,0))
327                   elif ord(data[15]) == 8:
328                     colorWipereverse8(strip, Color(255,0,0))
329                     colorWipereverse(strip, Color(0,0,0))
330                   elif ord(data[15]) == 9:
331                     colorWipereverse9(strip, Color(255,0,0))
332                     colorWipereverse(strip, Color(0,0,0))
333
334                 elif ord(data[14]) == 2:
335                   if ord(data[15]) == 0:
336                     colorWipereverse(strip, Color(0,255,0))
337                     colorWipereverse(strip, Color(0,0,0))
338                   elif ord(data[15]) == 1:
339                     colorWipereverse1(strip, Color(0,255,0))
340                     colorWipereverse(strip, Color(0,0,0))
```

```
341                 elif ord(data[15]) == 2:
342                   colorWipereverse2(strip, Color(0,255,0))
343                   colorWipereverse(strip, Color(0,0,0))
344                 elif ord(data[15]) == 3:
345                   colorWipereverse3(strip, Color(0,255,0))
346                   colorWipereverse(strip, Color(0,0,0))
347                 elif ord(data[15]) == 4:
348                   colorWipereverse4(strip, Color(0,255,0))
349                   colorWipereverse(strip, Color(0,0,0))
350                 elif ord(data[15]) == 5:
351                   colorWipereverse5(strip, Color(0,255,0))
352                   colorWipereverse(strip, Color(0,0,0))
353                 elif ord(data[15]) == 6:
354                   colorWipereverse6(strip, Color(0,255,0))
355                   colorWipereverse(strip, Color(0,0,0))
356                 elif ord(data[15]) == 7:
357                   colorWipereverse7(strip, Color(0,255,0))
358                   colorWipereverse(strip, Color(0,0,0))
359                 elif ord(data[15]) == 8:
360                   colorWipereverse8(strip, Color(0,255,0))
361                   colorWipereverse(strip, Color(0,0,0))
362                 elif ord(data[15]) == 9:
363                   colorWipereverse9(strip, Color(0,255,0))
364                   colorWipereverse(strip, Color(0,0,0))
365
366               elif ord(data[14]) == 3:
367                 if ord(data[15]) == 0:
368                   colorWipereverse(strip, Color(0,0,255))
369                   colorWipereverse(strip, Color(0,0,0))
370                 elif ord(data[15]) == 1:
371                   colorWipereverse1(strip, Color(0,0,255))
372                   colorWipereverse(strip, Color(0,0,0))
373                 elif ord(data[15]) == 2:
374                   colorWipereverse2(strip, Color(0,0,255))
375                   colorWipereverse(strip, Color(0,0,0))
376                 elif ord(data[15]) == 3:
377                   colorWipereverse3(strip, Color(0,0,255))
378                   colorWipereverse(strip, Color(0,0,0))
379                 elif ord(data[15]) == 4:
380                   colorWipereverse4(strip, Color(0,0,255))
381                   colorWipereverse(strip, Color(0,0,0))
382                 elif ord(data[15]) == 5:
```

```
383                     colorWipereverse5(strip, Color(0,0,255))
384                     colorWipereverse(strip, Color(0,0,0))
385                   elif ord(data[15]) == 6:
386                     colorWipereverse6(strip, Color(0,0,255))
387                     colorWipereverse(strip, Color(0,0,0))
388                   elif ord(data[15]) == 7:
389                     colorWipereverse7(strip, Color(0,0,255))
390                     colorWipereverse(strip, Color(0,0,0))
391                   elif ord(data[15]) == 8:
392                     colorWipereverse8(strip, Color(0,0,255))
393                     colorWipereverse(strip, Color(0,0,0))
394                   elif ord(data[15]) == 9:
395                     colorWipereverse9(strip, Color(0,0,255))
396                     colorWipereverse(strip, Color(0,0,0))
397
398                 elif ord(data[14]) == 4:
399                   if ord(data[15]) == 0:
400                     colorWipereverse(strip, Color(255,255,0))
401                     colorWipereverse(strip, Color(0,0,0))
402                   elif ord(data[15]) == 1:
403                     colorWipereverse1(strip, Color(255,255,0))
404                     colorWipereverse(strip, Color(0,0,0))
405                   elif ord(data[15]) == 2:
406                     colorWipereverse2(strip, Color(255,255,0))
407                     colorWipereverse(strip, Color(0,0,0))
408                   elif ord(data[15]) == 3:
409                     colorWipereverse3(strip, Color(255,255,0))
410                     colorWipereverse(strip, Color(0,0,0))
411                   elif ord(data[15]) == 4:
412                     colorWipereverse4(strip, Color(255,255,0))
413                     colorWipereverse(strip, Color(0,0,0))
414                   elif ord(data[15]) == 5:
415                     colorWipereverse5(strip, Color(255,255,0))
416                     colorWipereverse(strip, Color(0,0,0))
417                   elif ord(data[15]) == 6:
418                     colorWipereverse6(strip, Color(255,255,0))
419                     colorWipereverse(strip, Color(0,0,0))
420                   elif ord(data[15]) == 7:
421                     colorWipereverse7(strip, Color(255,255,0))
422                     colorWipereverse(strip, Color(0,0,0))
423                   elif ord(data[15]) == 8:
424                     colorWipereverse8(strip, Color(255,255,0))
```

```
425                     colorWipereverse(strip, Color(0,0,0))
426                   elif ord(data[15]) == 9:
427                     colorWipereverse9(strip, Color(255,255,0))
428                     colorWipereverse(strip, Color(0,0,0))
429
430                 elif ord(data[14]) == 5:
431                   if ord(data[15]) == 0:
432                     colorWipereverse(strip, Color(255,0,255))
433                     colorWipereverse(strip, Color(0,0,0))
434                   elif ord(data[15]) == 1:
435                     colorWipereverse1(strip, Color(255,0,255))
436                     colorWipereverse(strip, Color(0,0,0))
437                   elif ord(data[15]) == 2:
438                     colorWipereverse2(strip, Color(255,0,255))
439                     colorWipereverse(strip, Color(0,0,0))
440                   elif ord(data[15]) == 3:
441                     colorWipereverse3(strip, Color(255,0,255))
442                     colorWipereverse(strip, Color(0,0,0))
443                   elif ord(data[15]) == 4:
444                     colorWipereverse4(strip, Color(255,0,255))
445                     colorWipereverse(strip, Color(0,0,0))
446                   elif ord(data[15]) == 5:
447                     colorWipereverse5(strip, Color(255,0,255))
448                     colorWipereverse(strip, Color(0,0,0))
449                   elif ord(data[15]) == 6:
450                     colorWipereverse6(strip, Color(255,0,255))
451                     colorWipereverse(strip, Color(0,0,0))
452                   elif ord(data[15]) == 7:
453                     colorWipereverse7(strip, Color(255,0,255))
454                     colorWipereverse(strip, Color(0,0,0))
455                   elif ord(data[15]) == 8:
456                     colorWipereverse8(strip, Color(255,0,255))
457                     colorWipereverse(strip, Color(0,0,0))
458                   elif ord(data[15]) == 9:
459                     colorWipereverse9(strip, Color(255,0,255))
460                     colorWipereverse(strip, Color(0,0,0))
461
462                 elif ord(data[14]) == 6:
463                   if ord(data[15]) == 0:
464                     colorWipereverse(strip, Color(0,255,255))
465                     colorWipereverse(strip, Color(0,0,0))
466                   elif ord(data[15]) == 1:
```

```
467                     colorWipereverse1(strip, Color(0,255,255))
468                     colorWipereverse(strip, Color(0,0,0))
469                   elif ord(data[15]) == 2:
470                     colorWipereverse2(strip, Color(0,255,255))
471                     colorWipereverse(strip, Color(0,0,0))
472                   elif ord(data[15]) == 3:
473                     colorWipereverse3(strip, Color(0,255,255))
474                     colorWipereverse(strip, Color(0,0,0))
475                   elif ord(data[15]) == 4:
476                     colorWipereverse4(strip, Color(0,255,255))
477                     colorWipereverse(strip, Color(0,0,0))
478                   elif ord(data[15]) == 5:
479                     colorWipereverse5(strip, Color(0,255,255))
480                     colorWipereverse(strip, Color(0,0,0))
481                   elif ord(data[15]) == 6:
482                     colorWipereverse6(strip, Color(0,255,255))
483                     colorWipereverse(strip, Color(0,0,0))
484                   elif ord(data[15]) == 7:
485                     colorWipereverse7(strip, Color(0,255,255))
486                     colorWipereverse(strip, Color(0,0,0))
487                   elif ord(data[15]) == 8:
488                     colorWipereverse8(strip, Color(0,255,255))
489                     colorWipereverse(strip, Color(0,0,0))
490                   elif ord(data[15]) == 9:
491                     colorWipereverse9(strip, Color(0,255,255))
492                     colorWipereverse(strip, Color(0,0,0))
493
494                 elif ord(data[14]) == 7:
495                   if ord(data[15]) == 0:
496                     colorWipereverse(strip, Color(255,255,255))
497                     colorWipereverse(strip, Color(0,0,0))
498                   elif ord(data[15]) == 1:
499                     colorWipereverse1(strip, Color(255,255,255)
                          )
500                     colorWipereverse(strip, Color(0,0,0))
501                   elif ord(data[15]) == 2:
502                     colorWipereverse2(strip, Color(255,255,255)
                          )
503                     colorWipereverse(strip, Color(0,0,0))
504                   elif ord(data[15]) == 3:
505                     colorWipereverse3(strip, Color(255,255,255)
                          )
```

```
506                     colorWipereverse(strip, Color(0,0,0))
507                   elif ord(data[15]) == 4:
508                     colorWipereverse4(strip, Color(255,255,255)
                            )
509                     colorWipereverse(strip, Color(0,0,0))
510                   elif ord(data[15]) == 5:
511                     colorWipereverse5(strip, Color(255,255,255)
                            )
512                     colorWipereverse(strip, Color(0,0,0))
513                   elif ord(data[15]) == 6:
514                     colorWipereverse6(strip, Color(255,255,255)
                            )
515                     colorWipereverse(strip, Color(0,0,0))
516                   elif ord(data[15]) == 7:
517                     colorWipereverse7(strip, Color(255,255,255)
                            )
518                     colorWipereverse(strip, Color(0,0,0))
519                   elif ord(data[15]) == 8:
520                     colorWipereverse8(strip, Color(255,255,255)
                            )
521                     colorWipereverse(strip, Color(0,0,0))
522                   elif ord(data[15]) == 9:
523                     colorWipereverse9(strip, Color(255,255,255)
                            )
524                     colorWipereverse(strip, Color(0,0,0))
525
526                 elif ord(data[14]) == 8:
527                   if ord(data[15]) == 0:
528                     colorWipereverse(strip, Color(85,255,70))
529                     colorWipereverse(strip, Color(0,0,0))
530                   elif ord(data[15]) == 1:
531                     colorWipereverse1(strip, Color(85,255,70))
532                     colorWipereverse(strip, Color(0,0,0))
533                   elif ord(data[15]) == 2:
534                     colorWipereverse2(strip, Color(85,255,70))
535                     colorWipereverse(strip, Color(0,0,0))
536                   elif ord(data[15]) == 3:
537                     colorWipereverse3(strip, Color(85,255,70))
538                     colorWipereverse(strip, Color(0,0,0))
539                   elif ord(data[15]) == 4:
540                     colorWipereverse4(strip, Color(85,255,70))
541                     colorWipereverse(strip, Color(0,0,0))
```

```
542                     elif ord(data[15]) == 5:
543                       colorWipereverse5(strip, Color(85,255,70))
544                       colorWipereverse(strip, Color(0,0,0))
545                     elif ord(data[15]) == 6:
546                       colorWipereverse6(strip, Color(85,255,70))
547                       colorWipereverse(strip, Color(0,0,0))
548                     elif ord(data[15]) == 7:
549                       colorWipereverse7(strip, Color(85,255,70))
550                       colorWipereverse(strip, Color(0,0,0))
551                     elif ord(data[15]) == 8:
552                       colorWipereverse8(strip, Color(85,255,70))
553                       colorWipereverse(strip, Color(0,0,0))
554                     elif ord(data[15]) == 9:
555                       colorWipereverse9(strip, Color(85,255,70))
556                       colorWipereverse(strip, Color(0,0,0))
557
558                   elif ord(data[14]) == 9:
559                     if ord(data[15]) == 0:
560                       colorWipereverse(strip, Color(255,70,85))
561                       colorWipereverse(strip, Color(0,0,0))
562                     elif ord(data[15]) == 1:
563                       colorWipereverse1(strip, Color(255,70,85))
564                       colorWipereverse(strip, Color(0,0,0))
565                     elif ord(data[15]) == 2:
566                       colorWipereverse2(strip, Color(255,70,85))
567                       colorWipereverse(strip, Color(0,0,0))
568                     elif ord(data[15]) == 3:
569                       colorWipereverse3(strip, Color(255,70,85))
570                       colorWipereverse(strip, Color(0,0,0))
571                     elif ord(data[15]) == 4:
572                       colorWipereverse4(strip, Color(255,70,85))
573                       colorWipereverse(strip, Color(0,0,0))
574                     elif ord(data[15]) == 5:
575                       colorWipereverse5(strip, Color(255,70,85))
576                       colorWipereverse(strip, Color(0,0,0))
577                     elif ord(data[15]) == 6:
578                       colorWipereverse6(strip, Color(255,70,85))
579                       colorWipereverse(strip, Color(0,0,0))
580                     elif ord(data[15]) == 7:
581                       colorWipereverse7(strip, Color(255,70,85))
582                       colorWipereverse(strip, Color(0,0,0))
583                     elif ord(data[15]) == 8:
```

```
584                     colorWipereverse8(strip, Color(255,70,85))
585                     colorWipereverse(strip, Color(0,0,0))
586                   elif ord(data[15]) == 9:
587                     colorWipereverse1(strip, Color(255,70,85))
588                     colorWipereverse(strip, Color(0,0,0))
589       except:
590         pass
```

そして，ルータ教材のプログラムのソースコードを記載する。

```
#! /usr/bin/env/ python

import RPi.GPIO as GPIO
import time
import os
import socket
from neopixel import *

GPIO.setmode(GPIO.BCM)

GPIO.setup(4,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
GPIO.setup(17,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
GPIO.setup(22,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
GPIO.setup(10,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
GPIO.setup(9,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
GPIO.setup(11,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
GPIO.setup(14,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
GPIO.setup(15,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
GPIO.setup(18,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
GPIO.setup(23,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
GPIO.setup(24,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
GPIO.setup(27,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)

LED_COUNT       = 16
LED_GPIO        = 13
LED_CHANNEL     = 1
LED_FREQ_HZ     = 800000
LED_DMA         = 10
LED_BRIGHTNESS  = 150
LED_INVERT      = False

if __name__ == '__main__':
  strip = Adafruit_NeoPixel(LED_COUNT, LED_GPIO, LED_FREQ_HZ,
       LED_DMA, LED_INVERT, LED_BRIGHTNESS,LED_CHANNEL)
  strip.begin()
  print "LED"

def number():
  n = 0
  h = 0
  inputn1 = GPIO.input(4)
```

```
41    inputn2 = GPIO.input(14)
42    inputn4 = GPIO.input(15)
43    inputn8 = GPIO.input(17)
44    inputh1 = GPIO.input(9)
45    inputh2 = GPIO.input(27)
46    inputh4 = GPIO.input(22)
47    inputh8 = GPIO.input(11)
48    if inputn1 == True:
49      n = n + 1
50    if inputn2 == True:
51      n = n + 2
52    if inputn4 == True:
53      n = n + 4
54    if inputn8 == True:
55      n = n + 8
56    if inputh1 == True:
57      h = h + 1
58    if inputh2 == True:
59      h = h + 2
60    if inputh4 == True:
61      h = h + 4
62    if inputh8 == True:
63      h = h + 8
64    return [n, h]
65
66  def colorWipereverse(strip, color, wait_ms=10):
67    for i  in range(strip.numPixels(),-1,-1):
68      strip.setPixelColor(i, color)
69      strip.show()
70      time.sleep(wait_ms/1000.0)
71
72  def colorWipereverse1(strip, color, wait_ms=10):
73    for q in reversed(range(1)):
74      for i in range(0,strip.numPixels(),2):
75        strip.setPixelColor(i+q, color)
76        strip.show()
77        time.sleep(wait_ms/1000.0)
78
79  def colorWipereverse2(strip, color, wait_ms=10):
80    for q in reversed(range(2)):
81      for i in range(0,strip.numPixels(),3):
82        strip.setPixelColor(i+q, color)
```

```
83        strip.show()
84        time.sleep(wait_ms/1000.0)
85
86  def colorWipereverse3(strip, color, wait_ms=10):
87    for q in reversed(range(3)):
88      for i in range(0,strip.numPixels(),4):
89        strip.setPixelColor(i+q, color)
90        strip.show()
91        time.sleep(wait_ms/1000.0)
92
93  def colorWipereverse4(strip, color, wait_ms=10):
94    for q in reversed(range(4)):
95      for i in range(0,strip.numPixels(),5):
96        strip.setPixelColor(i+q, color)
97        strip.show()
98        time.sleep(wait_ms/1000.0)
99
100 def colorWipereverse5(strip, color, wait_ms=10):
101   for q in reversed(range(5)):
102     for i in range(0,strip.numPixels(),6):
103       strip.setPixelColor(i+q, color)
104       strip.show()
105       time.sleep(wait_ms/1000.0)
106
107 def colorWipereverse6(strip, color, wait_ms=10):
108   for q in reversed(range(6)):
109     for i in range(0,strip.numPixels(),7):
110       strip.setPixelColor(i+q, color)
111       strip.show()
112       time.sleep(wait_ms/1000.0)
113
114 def colorWipereverse7(strip, color, wait_ms=10):
115   for q in reversed(range(7)):
116     for i in range(0,strip.numPixels(),8):
117       strip.setPixelColor(i+q, color)
118       strip.show()
119       time.sleep(wait_ms/1000.0)
120
121 def colorWipereverse8(strip, color, wait_ms=10):
122   for q in reversed(range(8)):
123     for i in range(0,strip.numPixels(),9):
124       strip.setPixelColor(i+q, color)
```

```
125       strip.show()
126       time.sleep(wait_ms/1000.0)
127
128 def colorWipereverse9(strip, color, wait_ms=10):
129   for q in reversed(range(9)):
130     for i in range(0,strip.numPixels(),10):
131       strip.setPixelColor(i+q, color)
132       strip.show()
133       time.sleep(wait_ms/1000.0)
134
135 while True:
136   input1 = GPIO.input(23)
137   input2 = GPIO.input(18)
138   if input2 == True:
139     n ,h = number()
140     network = (str(n))
141     address1 = "ifconfig␣eth0␣192.168." + network + ".254␣␣
          netmask␣255.255.255.0"
142     print address1
143     os.system(str(address1))
144     while input2 == True:
145       input2 = GPIO.input(18)
146
147   if input1 == True:
148     n ,h = number()
149     host    = (str(h))
150     address2 = "ifconfig␣eth1␣192.168." + host + ".254␣␣
          netmask␣255.255.255.0"
151     print address2
152     os.system(str(address2))
153     while input1 == True:
154       input1 = GPIO.input(23)
155   else:
156     data = 0
157     sock = socket.socket(socket.AF_INET,socket.SOCK_RAW,
          socket.IPPROTO_ICMP)
158     sock.settimeout(0.1)
159     try:
160       data = sock.recv(255)
161       if data:
162         if ord(data[20]) == 8:
163           if ord(data[12]) == 192:
```

```
164             if ord(data[13]) == 168:
165               if ord(data[14]) == 0:
166                 if ord(data[15]) == 0:
167                   colorWipereverse(strip, Color(0,85,255))
168                   colorWipereverse(strip, Color(0,0,0))
169                 elif ord(data[15]) == 1:
170                   colorWipereverse1(strip, Color(0,85,255))
171                   colorWipereverse(strip, Color(0,0,0))
172                 elif ord(data[15]) == 2:
173                   colorWipereverse2(strip, Color(0,85,255))
174                   colorWipereverse(strip, Color(0,0,0))
175                 elif ord(data[15]) == 3:
176                   colorWipereverse3(strip, Color(0,0,255))
177                   colorWipereverse(strip, Color(0,0,0))
178                 elif ord(data[15]) == 4:
179                   colorWipereverse4(strip, Color(0,85,255))
180                   colorWipereverse(strip, Color(0,0,0))
181                 elif ord(data[15]) == 5:
182                   colorWipereverse5(strip, Color(0,85,255))
183                   colorWipereverse(strip, Color(0,0,0))
184                 elif ord(data[15]) == 6:
185                   colorWipereverse6(strip, Color(0,85,255))
186                   colorWipereverse(strip, Color(0,0,0))
187                 elif ord(data[15]) == 7:
188                   colorWipereverse7(strip, Color(0,85,255))
189                   colorWipereverse(strip, Color(0,0,0))
190                 elif ord(data[15]) == 8:
191                   colorWipereverse8(strip, Color(0,85,255))
192                   colorWipereverse(strip, Color(0,0,0))
193                 elif ord(data[15]) == 9:
194                   colorWipereverse9(strip, Color(0,85,255))
195                   colorWipereverse(strip, Color(0,0,0))
196
197               elif ord(data[14]) == 1:
198                 if ord(data[15]) == 0:
199                   colorWipereverse(strip, Color(255,0,0))
200                   colorWipereverse(strip, Color(0,0,0))
201                 elif ord(data[15]) == 1:
202                   colorWipereverse1(strip, Color(255,0,0))
203                   colorWipereverse(strip, Color(0,0,0))
204                 elif ord(data[15]) == 2:
205                   colorWipereverse2(strip, Color(255,0,0))
```

```
206                     colorWipereverse(strip, Color(0,0,0))
207                   elif ord(data[15]) == 2:
208                     colorWipereverse2(strip, Color(255,0,0))
209                     colorWipereverse(strip, Color(0,0,0))
210                   elif ord(data[15]) == 3:
211                     colorWipereverse3(strip, Color(255,0,0))
212                     colorWipereverse(strip, Color(0,0,0))
213                   elif ord(data[15]) == 4:
214                     colorWipereverse4(strip, Color(255,0,0))
215                     colorWipereverse(strip, Color(0,0,0))
216                   elif ord(data[15]) == 5:
217                     colorWipereverse5(strip, Color(255,0,0))
218                     colorWipereverse(strip, Color(0,0,0))
219                   elif ord(data[15]) == 6:
220                     colorWipereverse6(strip, Color(255,0,0))
221                     colorWipereverse(strip, Color(0,0,0))
222                   elif ord(data[15]) == 7:
223                     colorWipereverse7(strip, Color(255,0,0))
224                     colorWipereverse(strip, Color(0,0,0))
225                   elif ord(data[15]) == 8:
226                     colorWipereverse8(strip, Color(255,0,0))
227                     colorWipereverse(strip, Color(0,0,0))
228                   elif ord(data[15]) == 9:
229                     colorWipereverse9(strip, Color(255,0,0))
230                     colorWipereverse(strip, Color(0,0,0))
231
232                 elif ord(data[14]) == 2:
233                   if ord(data[15]) == 0:
234                     colorWipereverse(strip, Color(0,255,0))
235                     colorWipereverse(strip, Color(0,0,0))
236                   elif ord(data[15]) == 1:
237                     colorWipereverse1(strip, Color(0,255,0))
238                     colorWipereverse(strip, Color(0,0,0))
239                   elif ord(data[15]) == 2:
240                     colorWipereverse2(strip, Color(0,255,0))
241                     colorWipereverse(strip, Color(0,0,0))
242                   elif ord(data[15]) == 3:
243                     colorWipereverse3(strip, Color(0,255,0))
244                     colorWipereverse(strip, Color(0,0,0))
245                   elif ord(data[15]) == 4:
246                     colorWipereverse4(strip, Color(0,255,0))
247                     colorWipereverse(strip, Color(0,0,0))
```

```
248                     elif ord(data[15]) == 5:
249                       colorWipereverse5(strip, Color(0,255,0))
250                       colorWipereverse(strip, Color(0,0,0))
251                     elif ord(data[15]) == 6:
252                       colorWipereverse6(strip, Color(0,255,0))
253                       colorWipereverse(strip, Color(0,0,0))
254                     elif ord(data[15]) == 7:
255                       colorWipereverse7(strip, Color(0,255,0))
256                       colorWipereverse(strip, Color(0,0,0))
257                     elif ord(data[15]) == 8:
258                       colorWipereverse8(strip, Color(0,255,0))
259                       colorWipereverse(strip, Color(0,0,0))
260                     elif ord(data[15]) == 9:
261                       colorWipereverse9(strip, Color(0,255,0))
262                       colorWipereverse(strip, Color(0,0,0))
263
264                   elif ord(data[14]) == 3:
265                     if ord(data[15]) == 0:
266                       colorWipereverse(strip, Color(0,0,255))
267                       colorWipereverse(strip, Color(0,0,0))
268                     elif ord(data[15]) == 1:
269                       colorWipereverse1(strip, Color(0,0,255))
270                       colorWipereverse(strip, Color(0,0,0))
271                     elif ord(data[15]) == 2:
272                       colorWipereverse2(strip, Color(0,0,255))
273                       colorWipereverse(strip, Color(0,0,0))
274                     elif ord(data[15]) == 3:
275                       colorWipereverse3(strip, Color(0,0,255))
276                       colorWipereverse(strip, Color(0,0,0))
277                     elif ord(data[15]) == 4:
278                       colorWipereverse4(strip, Color(0,0,255))
279                       colorWipereverse(strip, Color(0,0,0))
280                     elif ord(data[15]) == 5:
281                       colorWipereverse5(strip, Color(0,0,255))
282                       colorWipereverse(strip, Color(0,0,0))
283                     elif ord(data[15]) == 6:
284                       colorWipereverse6(strip, Color(0,0,255))
285                       colorWipereverse(strip, Color(0,0,0))
286                     elif ord(data[15]) == 7:
287                       colorWipereverse7(strip, Color(0,0,255))
288                       colorWipereverse(strip, Color(0,0,0))
289                     elif ord(data[15]) == 8:
```

```
290                     colorWipereverse8(strip, Color(0,0,255))
291                     colorWipereverse(strip, Color(0,0,0))
292                   elif ord(data[15]) == 9:
293                     colorWipereverse9(strip, Color(0,0,255))
294                     colorWipereverse(strip, Color(0,0,0))
295
296                 elif ord(data[14]) == 4:
297                   if ord(data[15]) == 0:
298                     colorWipereverse(strip, Color(255,255,0))
299                     colorWipereverse(strip, Color(0,0,0))
300                   elif ord(data[15]) == 1:
301                     colorWipereverse1(strip, Color(255,255,0))
302                     colorWipereverse(strip, Color(0,0,0))
303                   elif ord(data[15]) == 2:
304                     colorWipereverse2(strip, Color(255,255,0))
305                     colorWipereverse(strip, Color(0,0,0))
306                   elif ord(data[15]) == 3:
307                     colorWipereverse3(strip, Color(255,255,0))
308                     colorWipereverse(strip, Color(0,0,0))
309                   elif ord(data[15]) == 4:
310                     colorWipereverse4(strip, Color(255,255,0))
311                     colorWipereverse(strip, Color(0,0,0))
312                   elif ord(data[15]) == 5:
313                     colorWipereverse5(strip, Color(255,255,0))
314                     colorWipereverse(strip, Color(0,0,0))
315                   elif ord(data[15]) == 6:
316                     colorWipereverse6(strip, Color(255,255,0))
317                     colorWipereverse(strip, Color(0,0,0))
318                   elif ord(data[15]) == 7:
319                     colorWipereverse7(strip, Color(255,255,0))
320                     colorWipereverse(strip, Color(0,0,0))
321                   elif ord(data[15]) == 8:
322                     colorWipereverse8(strip, Color(255,255,0))
323                     colorWipereverse(strip, Color(0,0,0))
324                   elif ord(data[15]) == 9:
325                     colorWipereverse9(strip, Color(255,255,0))
326                     colorWipereverse(strip, Color(0,0,0))
327
328                 elif ord(data[14]) == 5:
329                   if ord(data[15]) == 0:
330                     colorWipereverse(strip, Color(255,0,255))
331                     colorWipereverse(strip, Color(0,0,0))
```

```
332                     elif ord(data[15]) == 1:
333                       colorWipereverse1(strip, Color(255,0,255))
334                       colorWipereverse(strip, Color(0,0,0))
335                     elif ord(data[15]) == 2:
336                       colorWipereverse2(strip, Color(255,0,255))
337                       colorWipereverse(strip, Color(0,0,0))
338                     elif ord(data[15]) == 3:
339                       colorWipereverse3(strip, Color(255,0,255))
340                       colorWipereverse(strip, Color(0,0,0))
341                     elif ord(data[15]) == 4:
342                       colorWipereverse4(strip, Color(255,0,255))
343                       colorWipereverse(strip, Color(0,0,0))
344                     elif ord(data[15]) == 5:
345                       colorWipereverse5(strip, Color(255,0,255))
346                       colorWipereverse(strip, Color(0,0,0))
347                     elif ord(data[15]) == 6:
348                       colorWipereverse6(strip, Color(255,0,255))
349                       colorWipereverse(strip, Color(0,0,0))
350                     elif ord(data[15]) == 7:
351                       colorWipereverse7(strip, Color(255,0,255))
352                       colorWipereverse(strip, Color(0,0,0))
353                     elif ord(data[15]) == 8:
354                       colorWipereverse8(strip, Color(255,0,255))
355                       colorWipereverse(strip, Color(0,0,0))
356                     elif ord(data[15]) == 9:
357                       colorWipereverse9(strip, Color(255,0,255))
358                       colorWipereverse(strip, Color(0,0,0))
359
360                   elif ord(data[14]) == 6:
361                     if ord(data[15]) == 0:
362                       colorWipereverse(strip, Color(0,255,255))
363                       colorWipereverse(strip, Color(0,0,0))
364                     elif ord(data[15]) == 1:
365                       colorWipereverse1(strip, Color(0,255,255))
366                       colorWipereverse(strip, Color(0,0,0))
367                     elif ord(data[15]) == 2:
368                       colorWipereverse2(strip, Color(0,255,255))
369                       colorWipereverse(strip, Color(0,0,0))
370                     elif ord(data[15]) == 3:
371                       colorWipereverse3(strip, Color(0,255,255))
372                       colorWipereverse(strip, Color(0,0,0))
373                     elif ord(data[15]) == 4:
```

```
374                     colorWipereverse4(strip, Color(0,255,255))
375                     colorWipereverse(strip, Color(0,0,0))
376                   elif ord(data[15]) == 5:
377                     colorWipereverse5(strip, Color(0,255,255))
378                     colorWipereverse(strip, Color(0,0,0))
379                   elif ord(data[15]) == 6:
380                     colorWipereverse6(strip, Color(0,255,255))
381                     colorWipereverse(strip, Color(0,0,0))
382                   elif ord(data[15]) == 7:
383                     colorWipereverse7(strip, Color(0,255,255))
384                     colorWipereverse(strip, Color(0,0,0))
385                   elif ord(data[15]) == 8:
386                     colorWipereverse8(strip, Color(0,255,255))
387                     colorWipereverse(strip, Color(0,0,0))
388                   elif ord(data[15]) == 9:
389                     colorWipereverse9(strip, Color(0,255,255))
390                     colorWipereverse(strip, Color(0,0,0))
391
392                 elif ord(data[14]) == 7:
393                   if ord(data[15]) == 0:
394                     colorWipereverse(strip, Color(255,255,255))
395                     colorWipereverse(strip, Color(0,0,0))
396                   elif ord(data[15]) == 1:
397                     colorWipereverse1(strip, Color(255,255,255)
                          )
398                     colorWipereverse(strip, Color(0,0,0))
399                   elif ord(data[15]) == 2:
400                     colorWipereverse2(strip, Color(255,255,255)
                          )
401                     colorWipereverse(strip, Color(0,0,0))
402                   elif ord(data[15]) == 3:
403                     colorWipereverse3(strip, Color(255,255,255)
                          )
404                     colorWipereverse(strip, Color(0,0,0))
405                   elif ord(data[15]) == 4:
406                     colorWipereverse4(strip, Color(255,255,255)
                          )
407                     colorWipereverse(strip, Color(0,0,0))
408                   elif ord(data[15]) == 5:
409                     colorWipereverse5(strip, Color(255,255,255)
                          )
410                     colorWipereverse(strip, Color(0,0,0))
```

```
411                     elif ord(data[15]) == 6:
412                       colorWipereverse6(strip, Color(255,255,255)
                              )
413                       colorWipereverse(strip, Color(0,0,0))
414                     elif ord(data[15]) == 7:
415                       colorWipereverse7(strip, Color(255,255,255)
                              )
416                       colorWipereverse(strip, Color(0,0,0))
417                     elif ord(data[15]) == 8:
418                       colorWipereverse8(strip, Color(255,255,255)
                              )
419                       colorWipereverse(strip, Color(0,0,0))
420                     elif ord(data[15]) == 9:
421                       colorWipereverse9(strip, Color(255,255,255)
                              )
422                       colorWipereverse(strip, Color(0,0,0))
423
424                   elif ord(data[14]) == 8:
425                     if ord(data[15]) == 0:
426                       colorWipereverse(strip, Color(85,255,70))
427                       colorWipereverse(strip, Color(0,0,0))
428                     elif ord(data[15]) == 1:
429                       colorWipereverse1(strip, Color(85,255,70))
430                       colorWipereverse(strip, Color(0,0,0))
431                     elif ord(data[15]) == 2:
432                       colorWipereverse2(strip, Color(85,255,70))
433                       colorWipereverse(strip, Color(0,0,0))
434                     elif ord(data[15]) == 3:
435                       colorWipereverse3(strip, Color(85,255,70))
436                       colorWipereverse(strip, Color(0,0,0))
437                     elif ord(data[15]) == 4:
438                       colorWipereverse4(strip, Color(85,255,70))
439                       colorWipereverse(strip, Color(0,0,0))
440                     elif ord(data[15]) == 5:
441                       colorWipereverse5(strip, Color(85,255,70))
442                       colorWipereverse(strip, Color(0,0,0))
443                     elif ord(data[15]) == 6:
444                       colorWipereverse6(strip, Color(85,255,70))
445                       colorWipereverse(strip, Color(0,0,0))
446                     elif ord(data[15]) == 7:
447                       colorWipereverse7(strip, Color(85,255,70))
448                       colorWipereverse(strip, Color(0,0,0))
```

```
449                 elif ord(data[15]) == 8:
450                   colorWipereverse8(strip, Color(85,255,70))
451                   colorWipereverse(strip, Color(0,0,0))
452                 elif ord(data[15]) == 9:
453                   colorWipereverse9(strip, Color(85,255,70))
454                   colorWipereverse(strip, Color(0,0,0))
455
456               elif ord(data[14]) == 9:
457                 if ord(data[15]) == 0:
458                   colorWipereverse(strip, Color(255,70,85))
459                   colorWipereverse(strip, Color(0,0,0))
460                 elif ord(data[15]) == 1:
461                   colorWipereverse1(strip, Color(255,70,85))
462                   colorWipereverse(strip, Color(0,0,0))
463                 elif ord(data[15]) == 2:
464                   colorWipereverse2(strip, Color(255,70,85))
465                   colorWipereverse(strip, Color(0,0,0))
466                 elif ord(data[15]) == 3:
467                   colorWipereverse3(strip, Color(255,70,85))
468                   colorWipereverse(strip, Color(0,0,0))
469                 elif ord(data[15]) == 4:
470                   colorWipereverse4(strip, Color(255,70,85))
471                   colorWipereverse(strip, Color(0,0,0))
472                 elif ord(data[15]) == 5:
473                   colorWipereverse5(strip, Color(255,70,85))
474                   colorWipereverse(strip, Color(0,0,0))
475                 elif ord(data[15]) == 6:
476                   colorWipereverse6(strip, Color(255,70,85))
477                   colorWipereverse(strip, Color(0,0,0))
478                 elif ord(data[15]) == 7:
479                   colorWipereverse7(strip, Color(255,70,85))
480                   colorWipereverse(strip, Color(0,0,0))
481                 elif ord(data[15]) == 8:
482                   colorWipereverse8(strip, Color(255,70,85))
483                   colorWipereverse(strip, Color(0,0,0))
484                 elif ord(data[15]) == 9:
485                   colorWipereverse1(strip, Color(255,70,85))
486                   colorWipereverse(strip, Color(0,0,0))
487     except:
488       pass
```