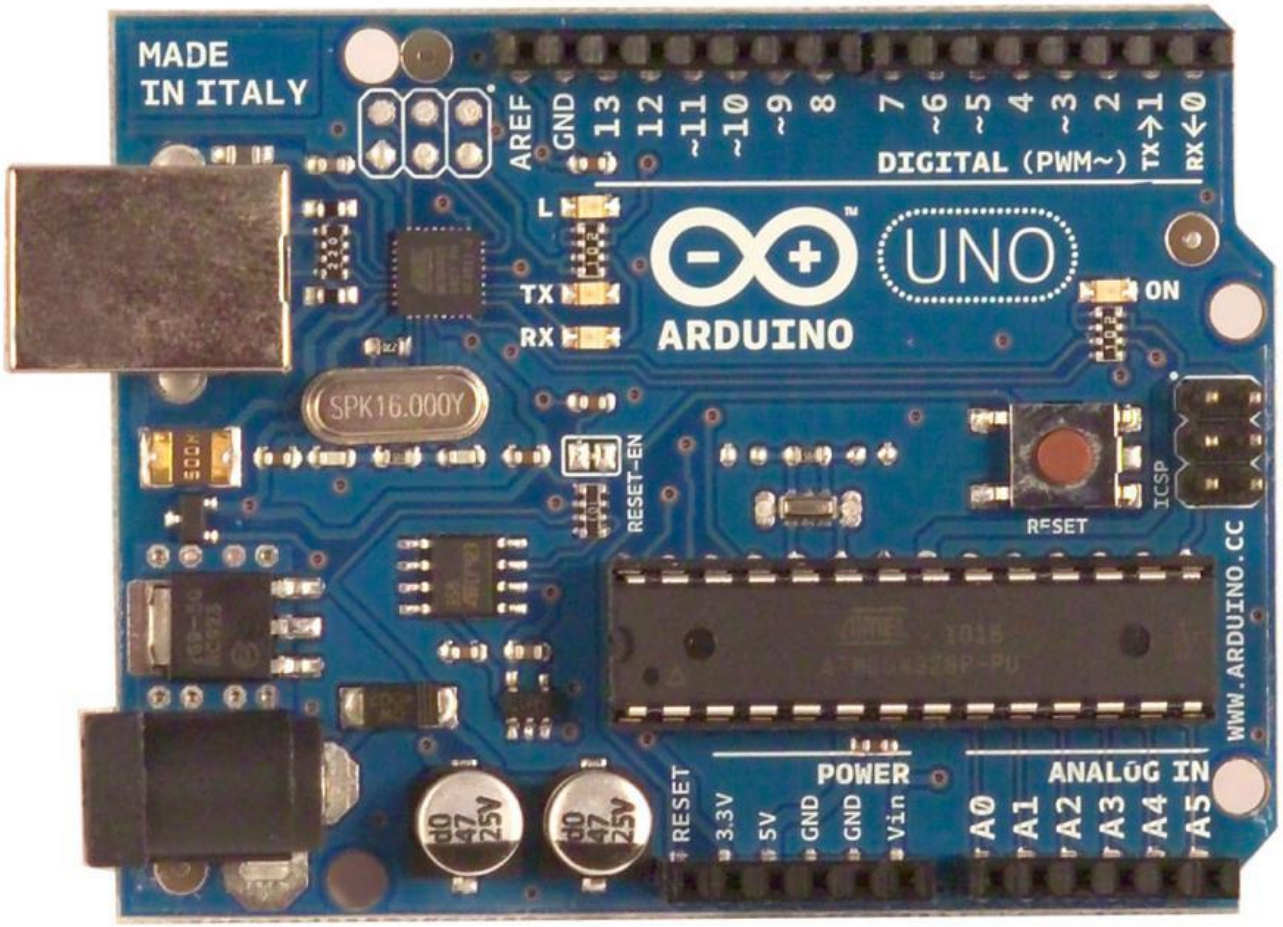
Arduinoの I/Oポート

・14本（＋6本）のデジタル入出力ポート
・6本のアナログ入力ポート



It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button.

Arduinoシミュレータ

各自のデスクトップのプログラムフォルダーにArduinoシミュレータUnoArduSimがあるので、ダブルクリックして実行する。 また、UnoArduSimは
https://www.sites.google.com/site/unoardusim/services
で配布されている。（インストールせずに実行ファイルのみで実行可能）

使用上の参考サイト→arduino日本語リファレンス
http://www.musashinodenpa.com/arduino/ref/

また、文法はC++に準拠しているので、必要に応じて、調べること

C++とPythonの比較

	C++, C	Python
言語の特徴	コンパイル言語	インタープリタ言語
長所	• 実行速度が速い • 機械語の実行形式になっているので、小さなコンピュータで実行可能	• コンパイルの必要が無いため、待たずに実行できる • 多少のエラーは融通がきく • 文字列の操作が得意
短所	• 規模によってはコンパイルに時間がかかる • エラーがあるとコンパイルできない • 文字列の操作が比較的苦手	• C++に比べて遅い • 実行するコンピュータにはPythonインタープリタを実装している必要があるので、小さなコンピュータでは実行できない

マイクロコンピュータの記述言語としてはCまたはC++が使われることが多い

C++とPythonの文法構造の比較

C++

```
int x,y,z;
if (x > 500)
{
    y=x+100;
    Serial.println(y);
}
z=100;
```

- 変数の型宣言が必要
- if文の論理式は丸括弧の中に入れる
- 論理式がtrueの時に実行される実行単位は波括弧の中に入れる（インデントは動作とは無関係）

Python

```
if  x > 500:
    y=x+100
    print(y)
z=100
```

- 変数の型宣言が不要
- if文の論理式はセミコロンの前までとする
- 論理式がtrueの時に実行される実行単位はインデントされている部分とする

プログラムの基本構造

```
int ButtonPin = 3;
void setup()
{
    beginSerial(9600);
    pinMode(buttonPin, INPUT);
}
void xyz(){
  int abc;
}
void loop()
{
    xyz();  // 実行したいプログラム
}
```

グローバル変数の定義

文の最後はセミコロン

初期設定プログラム（起動後、1度だけ実行される）

関数の定義（機能を小さな単位で分割することが望ましい）

ローカル変数の定義

実行プログラム本体（ループし続ける）

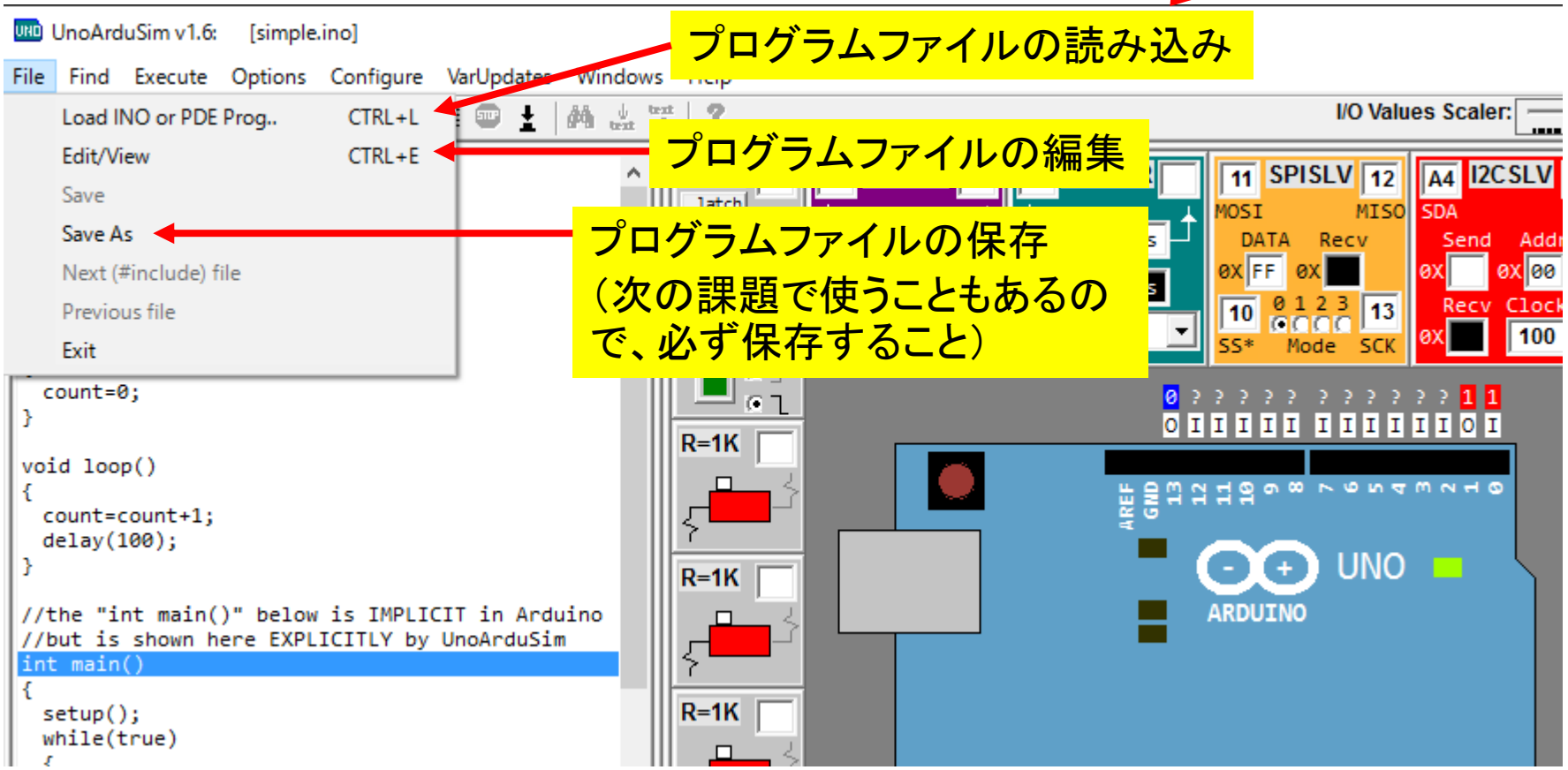
//以降はコメント

実行部は波括弧の中に入れる

UnoArduSimの使い方

プログラムファイルの読み込み・編集・保存

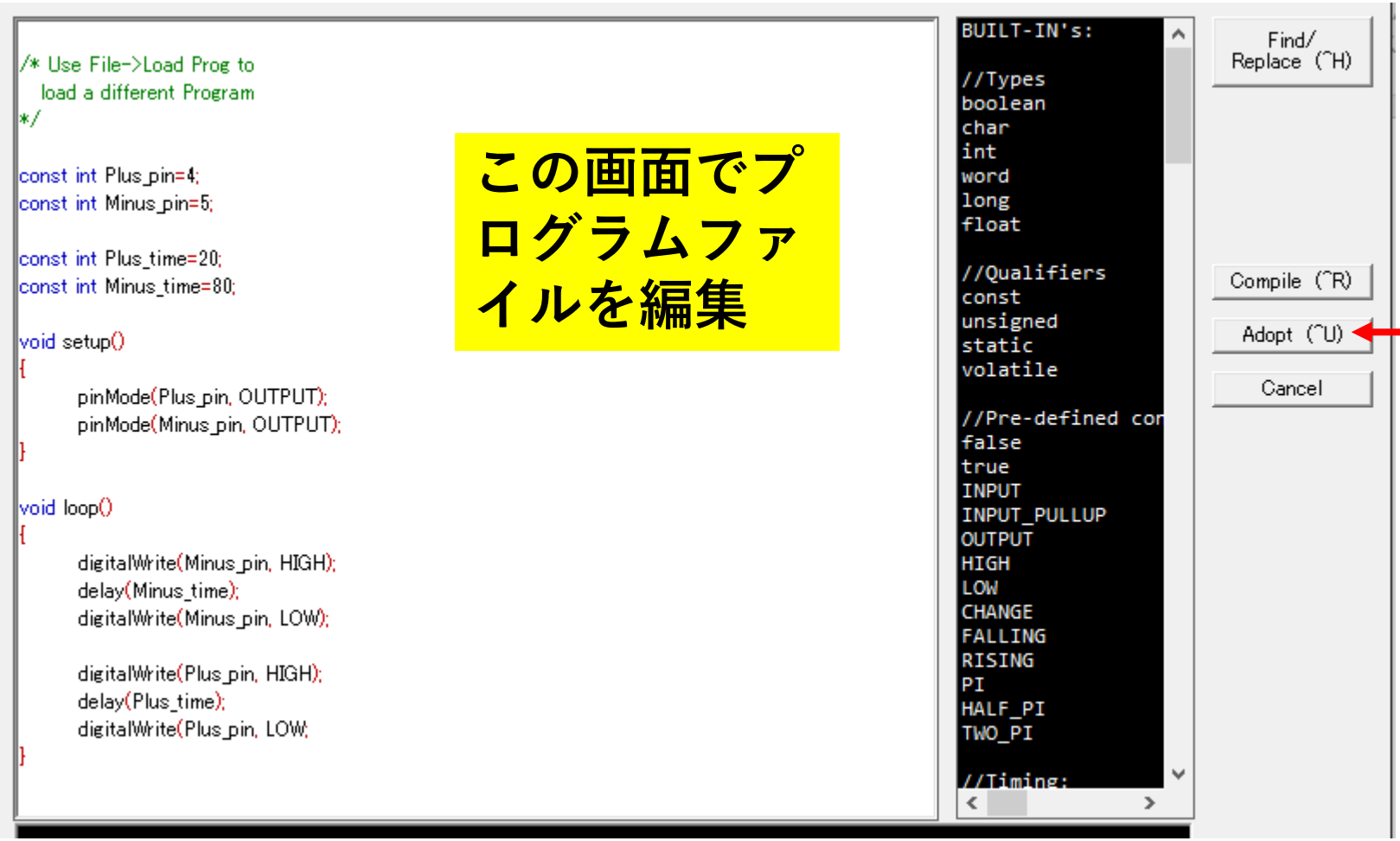
ファイルを開くのにはダブルクリックではなく、これを使うこと。（ダブルクリックではArduino IDEが立ち上がる）



UnoArduSimの使い方（編集・コンパイル）

ファイルメニューでEdit/Viewを選ぶと以下のwindowが開く。

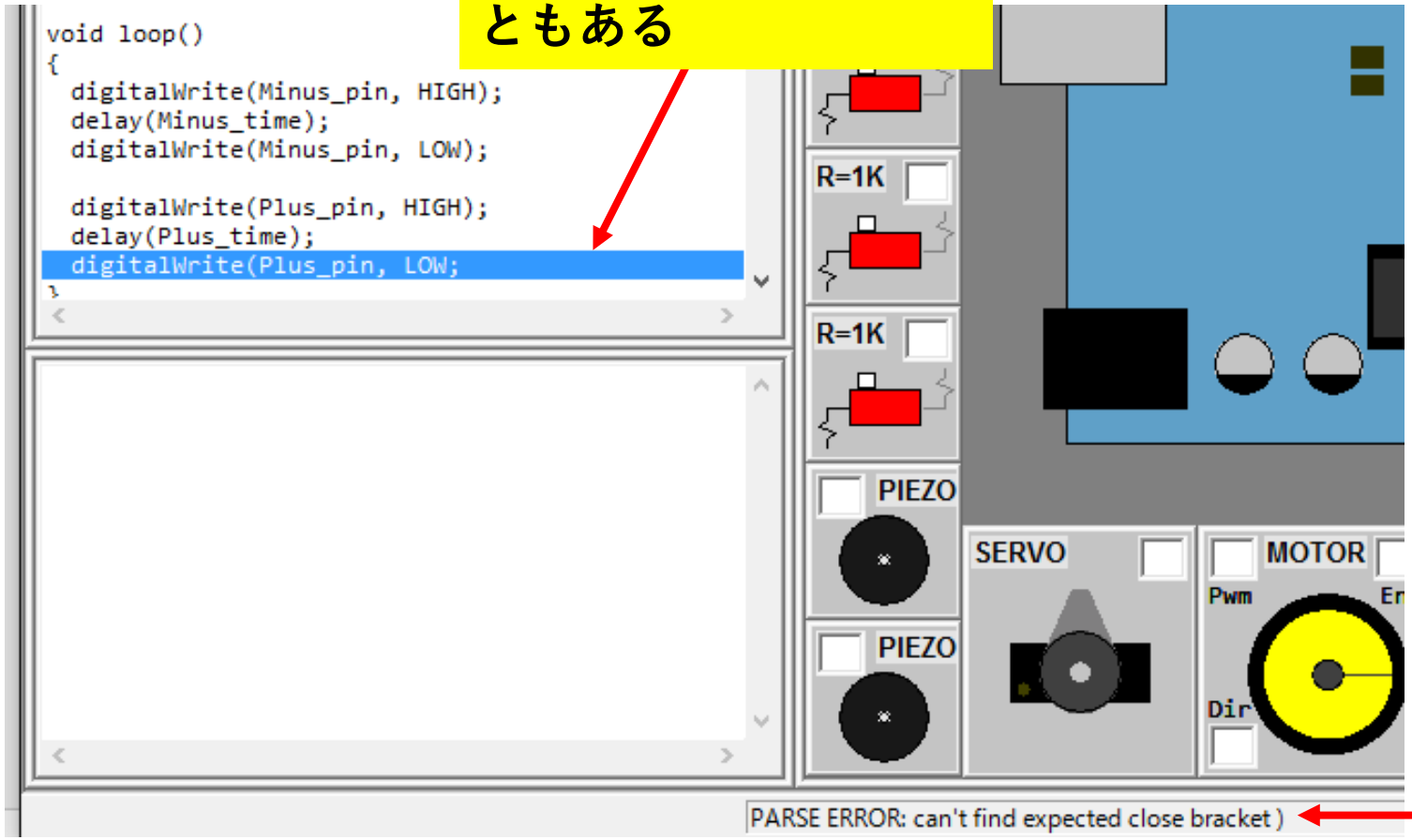
プログラムファイルの編集画面およびコンパイル



UnoArduSimの使い方（エラーの表示）

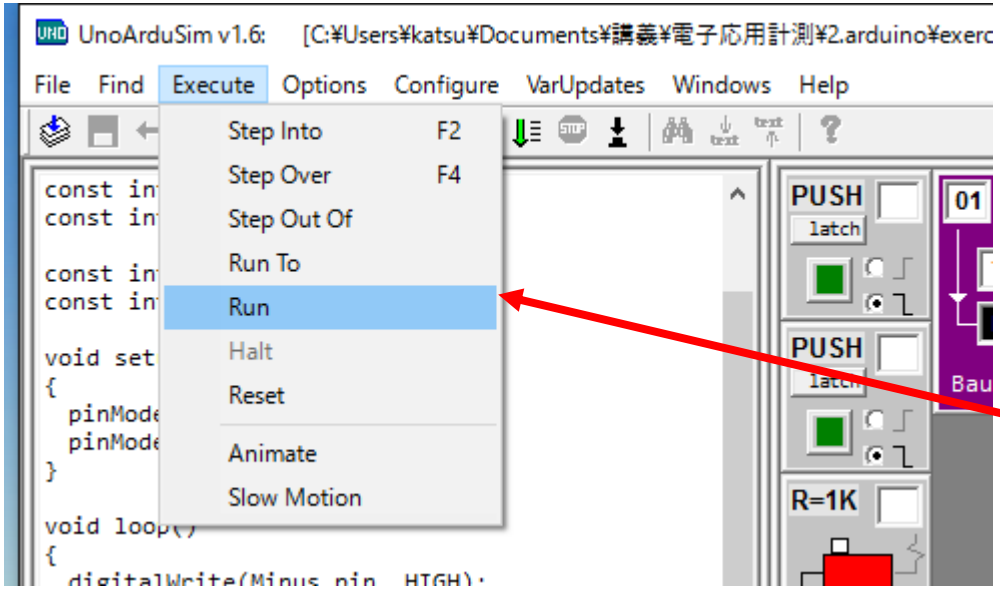
“Adopt”を押しても、エラーがあると実行できない

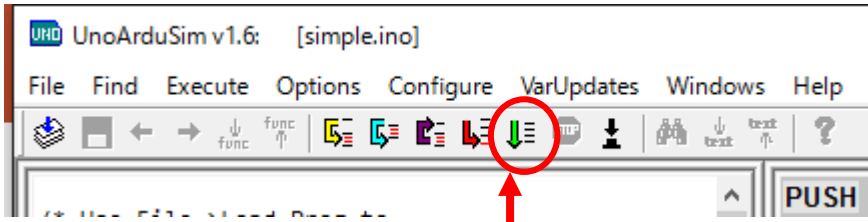
エラーを起こしている文が強調されることもある



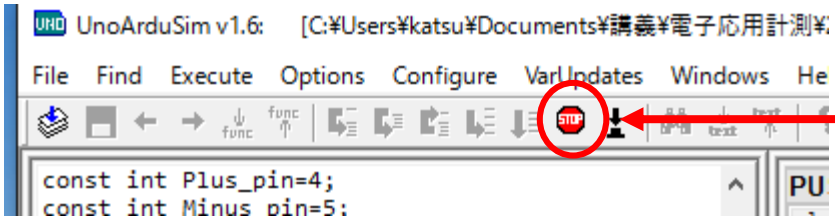
文法エラー時(PARSE ERROR)には、ここにエラーの内容が表示される

UnoArduSimの使い方（実行方法）





プルダウンメニューから”Run”を選ぶか（左）、実行アイコン（上）を選んで実行させる。

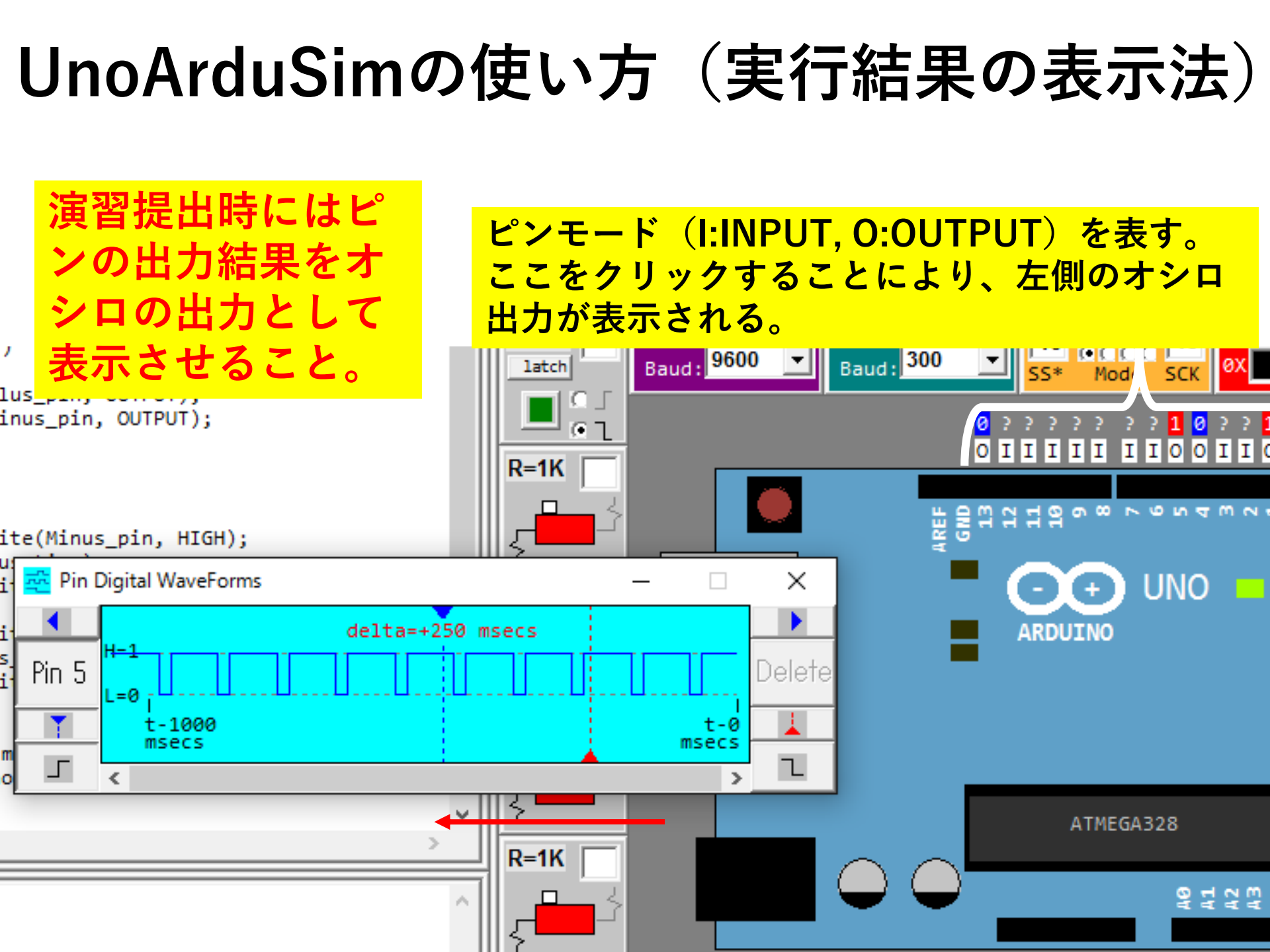


途中で実行をストップするときにはストップアイコンを押す

UnoArduSimの使い方（実行結果の表示法）

演習提出時にはピンの出力結果をオシロの出力として表示させること。

ピンモード（I:INPUT, O:OUTPUT）を表す。ここをクリックすることにより、左側のオシロ出力が表示される。

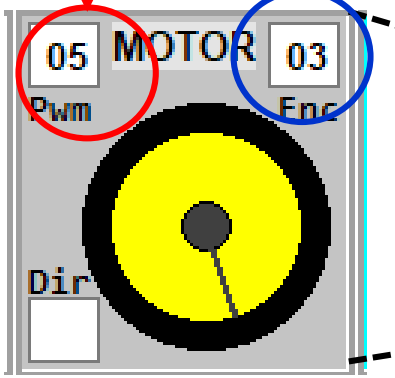


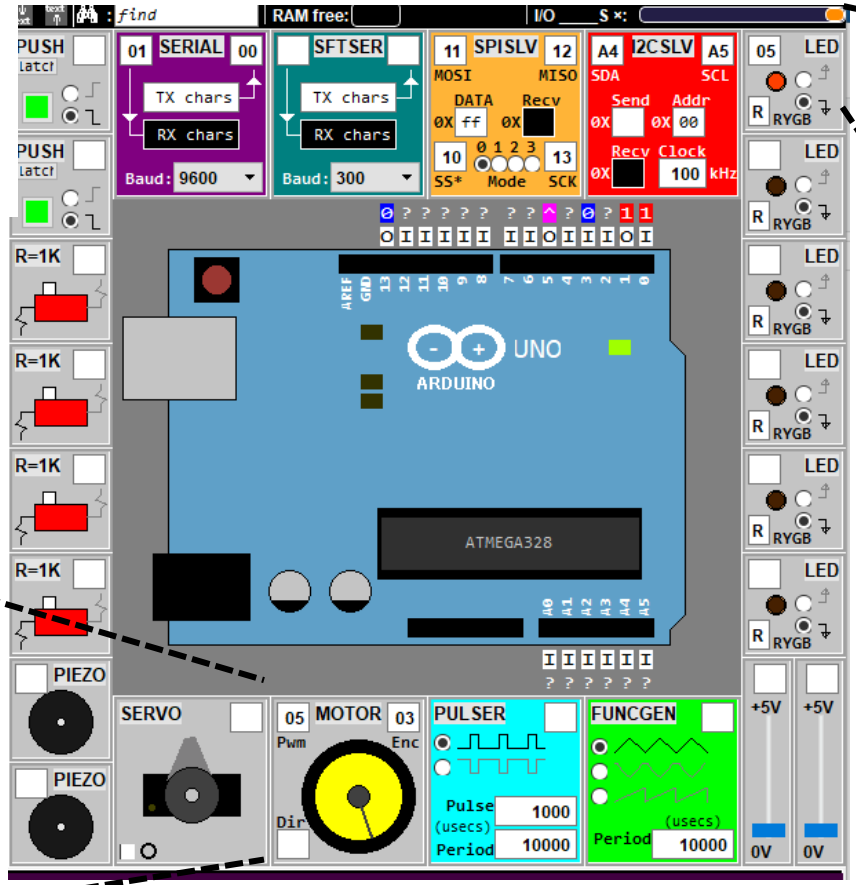
モーター、LEDの接続方法

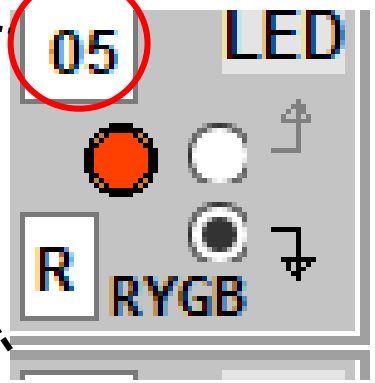
Wheel Encoder:ここから１回転あたり８パルスの出力が出てくるので、適当なピンに接続し、その信号を見る

ここにPWM出力のピン番号を入力

ここにLEDに接続するピン番号を入力

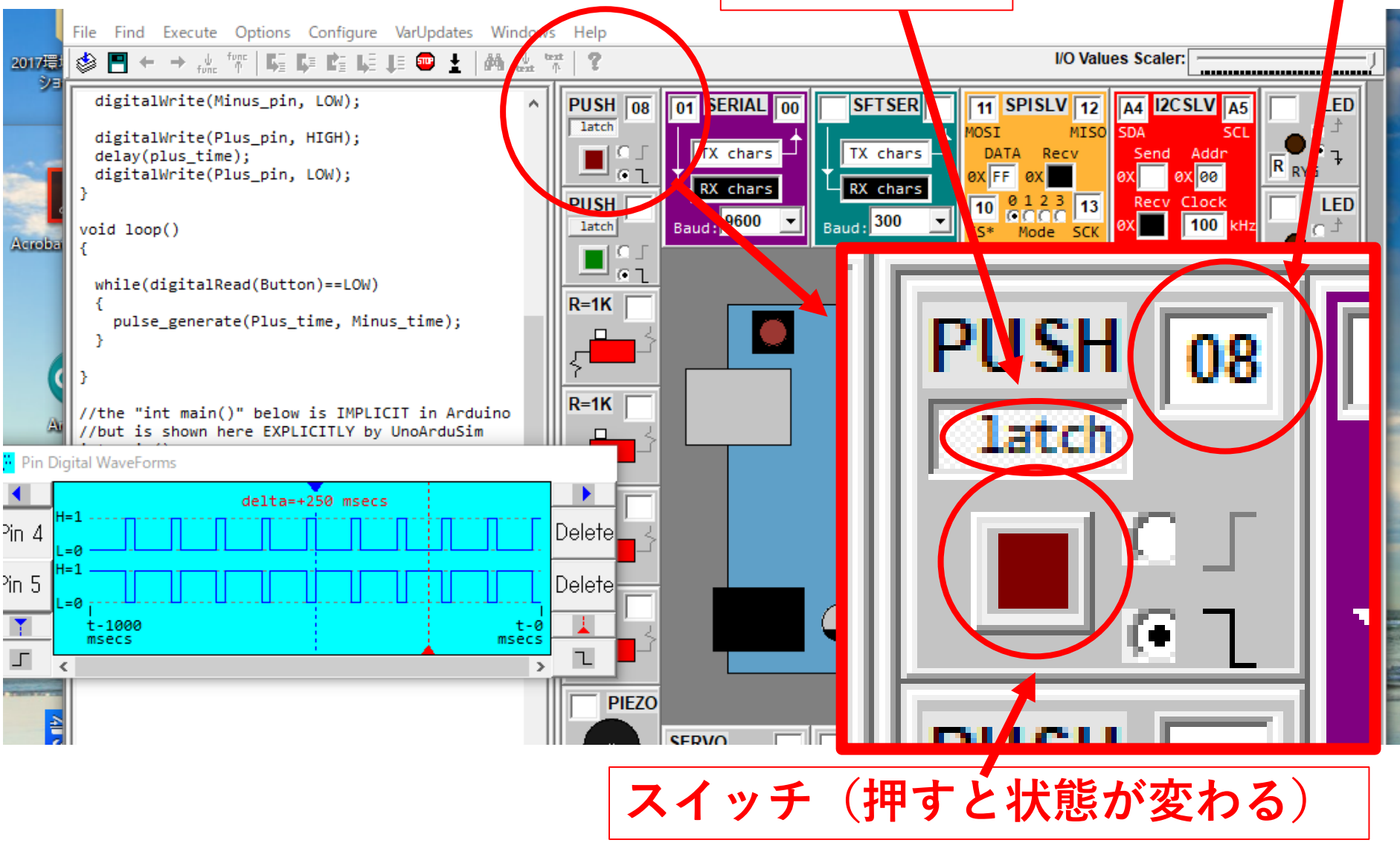






PWMピンの出力を観測し、マウスのホイールを使って時間軸を拡大し、デューティー比が変化していることを確かめる。

スイッチの使い方


スイッチの種類
このスイッチの接続ピン番号
スイッチ（押すと状態が変わる）
File Find Execute Options Configure VarUpdates Windows Help
I/O Values Scaler:
digitalWrite(Minus_pin, LOW);
digitalWrite(Plus_pin, HIGH);
delay(plus_time);
digitalWrite(Plus_pin, LOW);
}
void loop()
{
while(digitalRead(Button)==LOW)
{
pulse_generate(Plus_time, Minus_time);
}
}
//the "int main()" below is IMPLICIT in Arduino
//but is shown here EXPLICITLY by UnoArduSim
PUSH 08 latch
SERIAL 00
TX chars
RX chars
Baud: 9600
SFTSER
TX chars
RX chars
Baud: 300
11 SPISLV 12
MOSI MISO
DATA Recv
0X FF
10 13
S* Mode SCK
A4 I2CSLV A5
SDA SCL
Send Addr
Recv Clock
100 kHz
LED
R=1K
R=1K
PIEZO
SERVO
Pin Digital WaveForms
delta=+250 msecs
H=1
L=0
H=1
L=0
Pin 4
Pin 5
t-1000 msecs
t-0 msecs
Delete
Delete
PUSH
08
latch

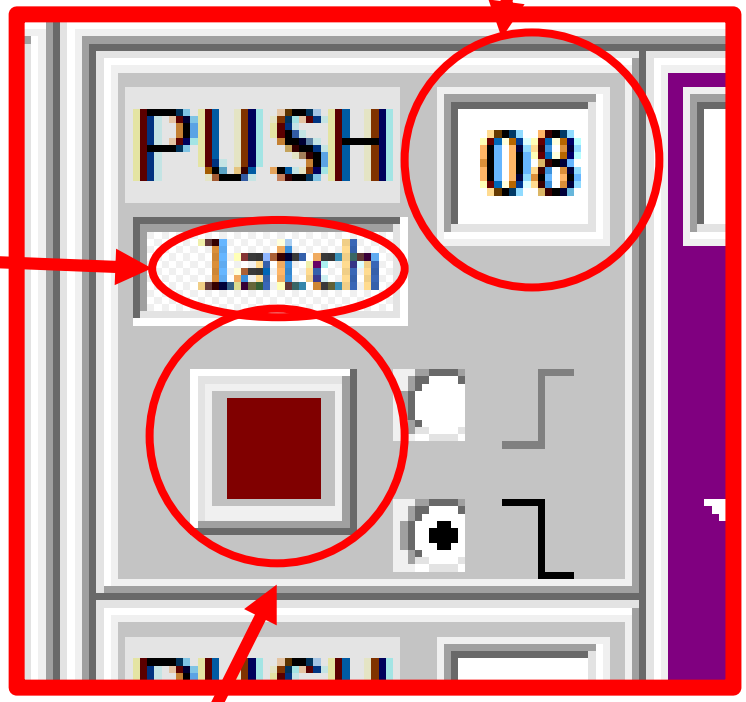
Pushスイッチの設定

このスイッチの接続ピン番号（キーボードから入力）

スイッチの種類
Latch:押し込むことにより状態を変えられる

latchなし：押しているときだけOn（LOW)になる

latchあり：押す毎にon(LOW)とoff(HIGH)が切り替わる



スイッチ（押すと状態が変わる）