第3章：マイクロコントローラ(Arduino)の基礎

KYUSHU
UNIVERSITY

全体構成

Chapter 1
- 第1章：導入

Chapter 2
- 第2章：IoT基礎（センサー）

Chapter 3
- 第3章：マイクロコントローラ（Arduino）の基礎

Chapter 4
- 第4章：計測データ解析の基礎

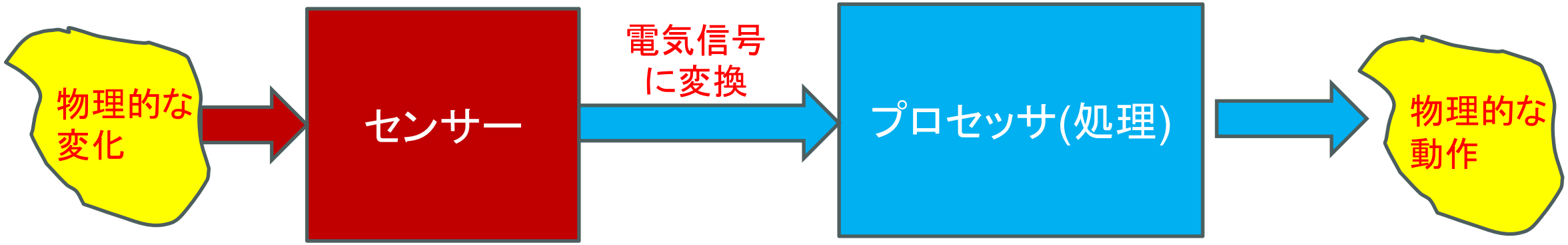
Chapter 5
- 第5章：計測データ処理(IoT)の応用

IoTシステムでは、センサーとアクチュエータの働きにより、「情報の収集とデバイスの制御」を可能にしている。

IoTシステムの基本構成：

センサー ＋ プロセッサ ＋ アクチュエータ

環境からデータを収集　　データを解釈・判断　　物理的な動作



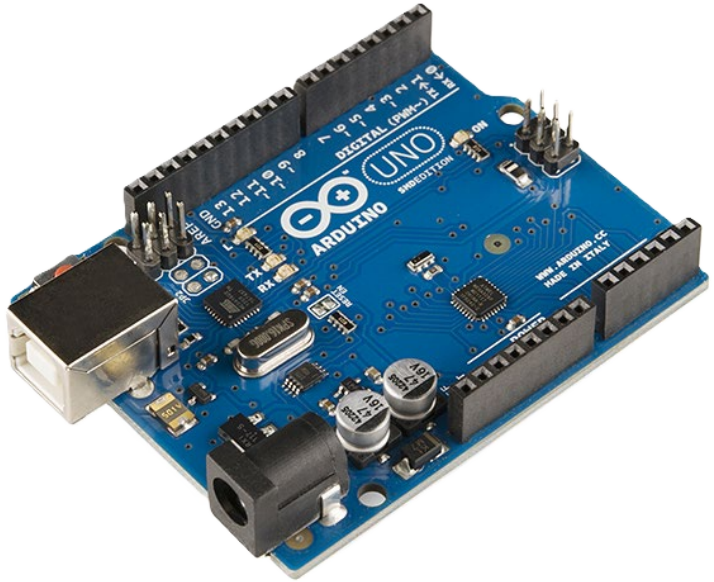
IoTでは、プロセッサ（マイクロコントローラ等の「処理装置」）が必要。

なぜ？

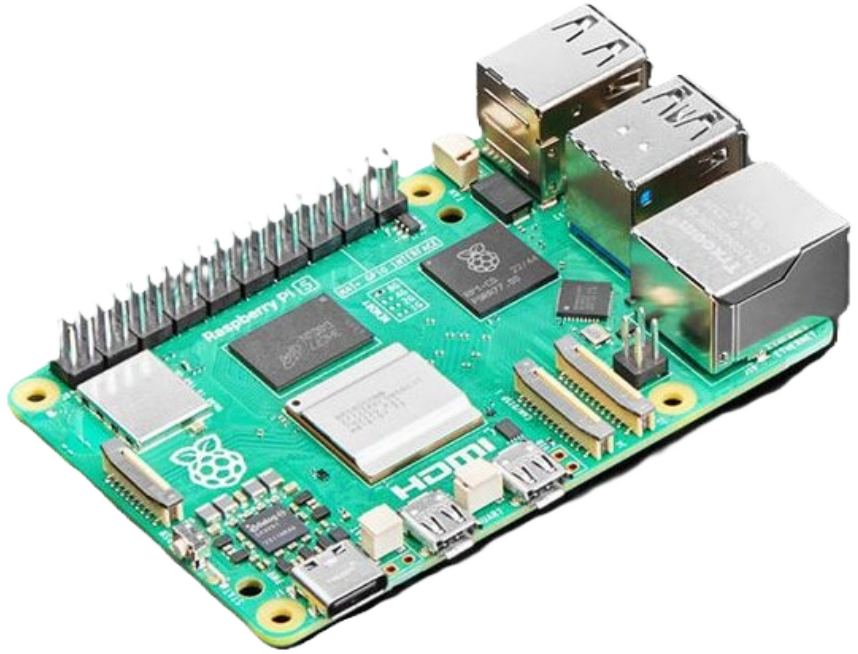
センサーで収集したデータを「処理」し、アクチュエータを「制御」するための指令塔が必要（集めたデータを活用するための処理装置）

例）自動ドアに近づく ➔ 距離センサー(光センサー) ➔ プロセッサが判断 ➔ モーターを動かしてドアを開閉

IoT用途で用いられる代表的なデバイス



Arduino Uno（アルディーノ ウノ）

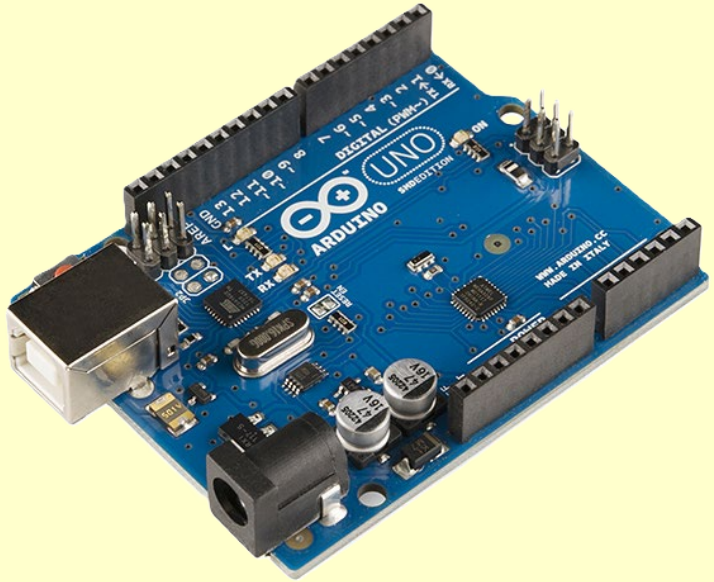


Raspberry Pi（ラズベリーパイ）

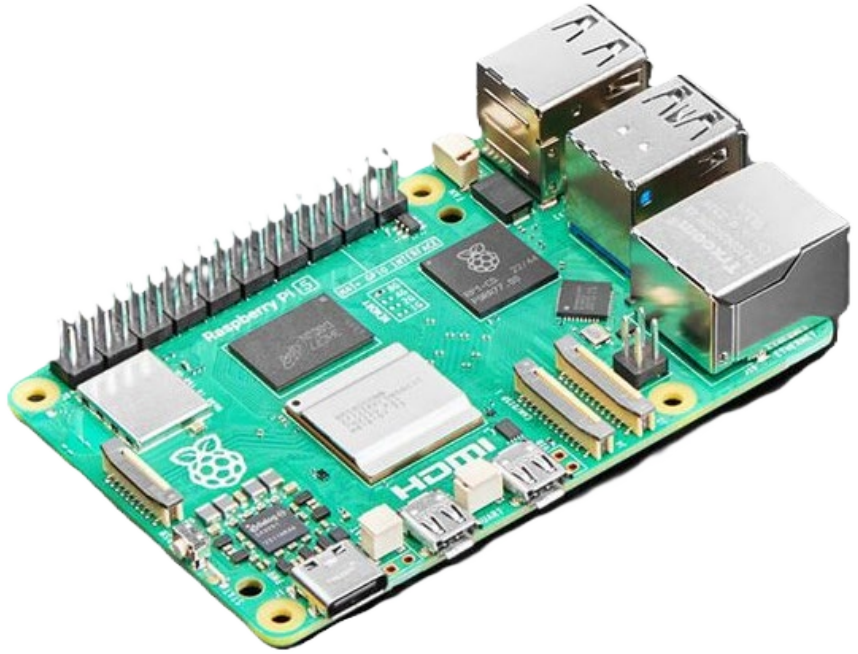
- マイクロコントローラを搭載した「Arduino Uno」や、シングルボードコンピュータの「Raspberry Pi」が代表例
- いずれもプロセッサと入出力インターフェースを備え、センサーやアクチュエータを制御可能

IoT用途で用いられる代表的なデバイス

本演習で用いるデバイス



Arduino Uno（アルディーノ ウノ）



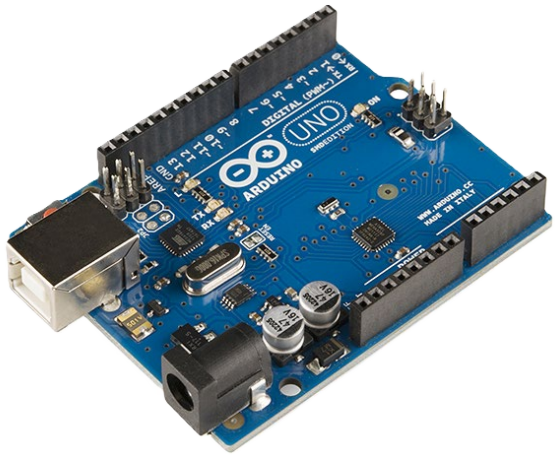
Raspberry Pi（ラズベリーパイ）

- マイクロコントローラを搭載した「Arduino Uno」や、シングルボードコンピュータの「Raspberry Pi」が代表例
- いずれもプロセッサと入出力インターフェースを備え、センサーやアクチュエータを制御可能

第3章：マイクロコントローラ（Arduino）の基礎

Arduino Uno（アルディーノ ウノ）

命令（プログラム）を
直接マイコンに入力

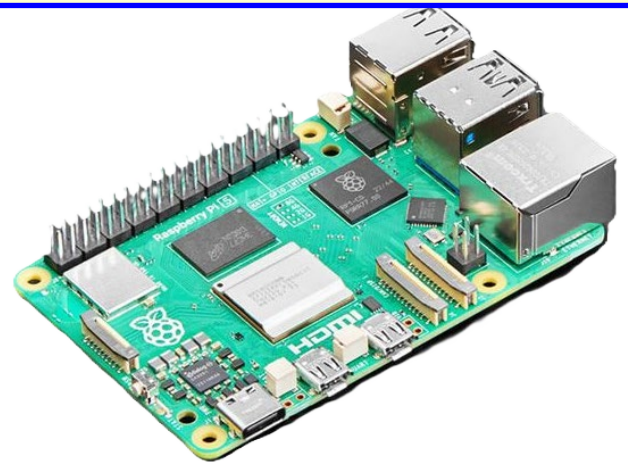


- 基板上にマイコンを搭載
- OS不要の制御用プロセッサ

Raspberry Pi（ラズベリーパイ）

プログラム（アプリ）

オペレーティングシステム（Linux等）



- 汎用プロセッサを搭載した小型計算機
- 多機能なアプリケーションを実行可能

料理に例えるなら、

- Arduino Uno は「ラーメン専用マシン」のようなもの。
- 1つのボタンを押せばラーメンが作れて、誰でも簡単に利用可能。ただし、作れるのはラーメンだけ。できることは限定的（工夫すれば、チャーハンぐらいは作れるかも。。）

- Raspberry Pi は「キッチン（厨房）」のような存在。
- 調理器具（OSの部品やツール）を揃えていて、使いこなせれば、ラーメンだけでなくいろいろな料理（多機能なアプリや制御）を作れる。ただし、道具の準備や使い方（OSの操作やツールの使い方）を学ぶ必要がある。

料理に例えるなら、

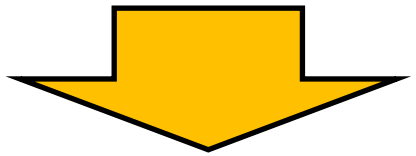
- Arduino Uno は「ラーメン専用マシン」のようなもの。
- 1つのボタンを押せばラーメンが作れて、誰でも簡単に利用可能。ただし、作れるのはラーメンだけ。できることは限定的（工夫すれば、チャーハンぐらいは作れるかも。。）

初学者も取り扱いやすい。今回の教材として選択

- Raspberry Pi は「キッチン（厨房）」のような存在。
- 調理器具（OSの部品やツール）を揃えていて、使いこなせれば、ラーメンだけでなくいろいろな料理（多機能なアプリや制御）を作れる。ただし、道具の準備や使い方（OSの操作やツールの使い方）を学ぶ必要がある。

- Arduino Uno は、センサーやアクチュエータを使った単純な制御をリアルタイムで実行可能

- ただし、その制御を実現するためには、「どのように動かすか」をプログラムとして書く必要がある。



次章から、Arduino Unoを用いて「シンプルな制御プログラムの実装」や「簡易な計測データ処理プログラム」について演習形式で学びます。

マイクロコントローラ（MCU）とは？

- 電子機器を制御するために設計された小型の集積回路
- MCUは接続されたデバイス（例：センサーやアクチュエーター）との通信や制御を行う

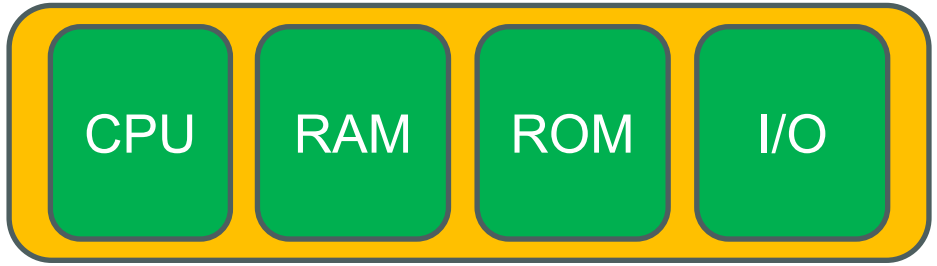
◘ 主な構成要素

✓ CPU（命令を実行する装置）
✓ メモリ（Flash, RAM）
✓ I/O ピン（GPIO, ADC, PWM）

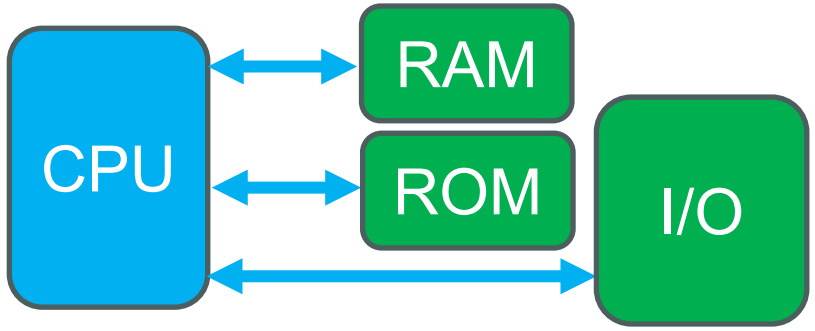
◘ マイクロコントローラ vs マイクロプロセッサ

- マイクロコントローラ：「CPU＋周辺機能」を一体型で搭載
  （例：Arduinoなど）
- マイクロプロセッサ：外部の構成部品（メモリやI/Oなど）が別途必要（例：Intelのデスクトップ用CPU+メモリ+I/O）

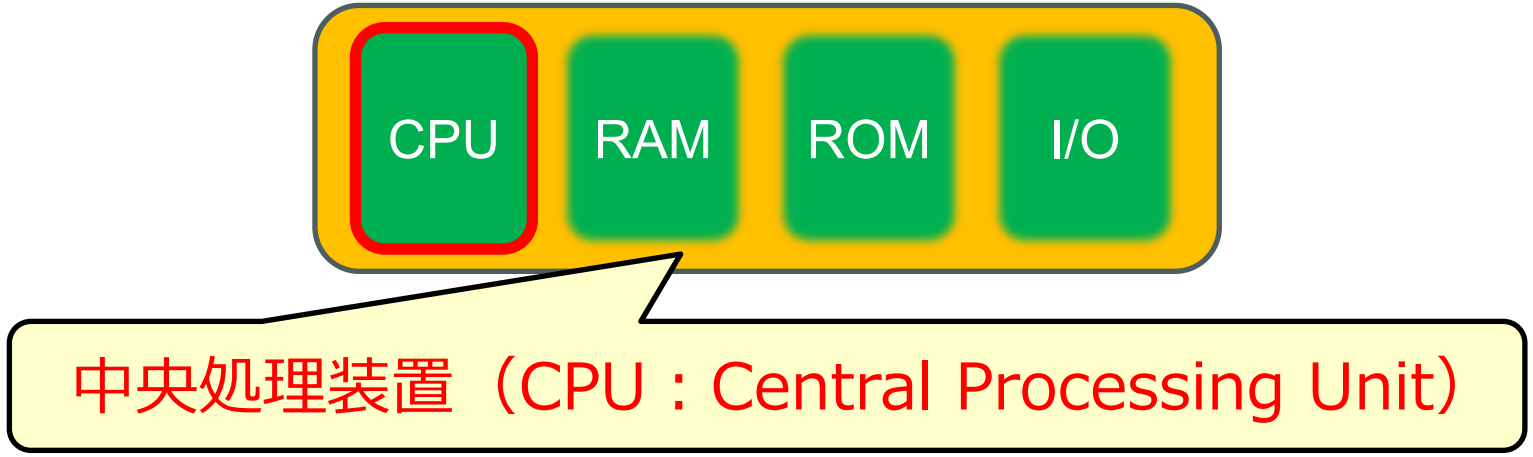
■ マイクロコントローラ
（CPU・RAM・ROM・I/Oを1チップに）



■ マイクロプロセッサ



中央処理装置（CPU）



- マイクロコントローラ部の「頭脳」
- 入出力処理、算術演算、データ転送、基本的な論理演算などの処理を実行
- 命令を1つずつ、一定のタイミング（クロック）で順番に実行する。

- CPUが一つの命令を実行するのに要する時間は、

「1サイクルにかかる時間 × 1命令にかかるサイクル数」

したがって、一つのプログラムを実行するのに要する時間は

時間／プログラム ＝ 1サイクルにかかる時間 × 1命令にかかるサイクル数 × プログラム中の命令数

1命令あたりの処理速度

たとえば、
1命令に2サイクル、サイクル時間が1μs、命令数が100なら → 実行時間は 200μs

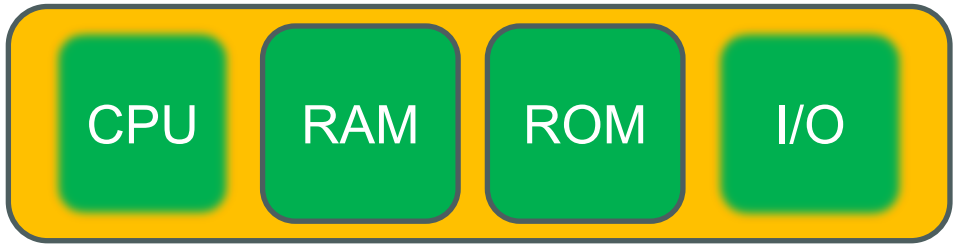
Arduinoは簡易な命令セットを用いて、1命令を1クロックサイクルで実行できるように設計されている※1。

- ハードウェアがシンプルで消費電力が少ない → 低消費電力デバイス向き
- 命令実行時間の見積もりが容易なので、リアルタイム処理に適する → リアルタイムアプリケーション向き

※1：そのような設計思想は RISC（縮小命令セットコンピュータ, Reduced Instruction Set Computer）と呼ばれる。

メモリ

- データと命令を記憶する



RAM（データメモリ: Random access memory）

- 読み書き可能
- プログラムの実行中に一時的なデータを保存するのに使用
- 例：変数、センサ値、処理中のデータなど

ROM（プログラムメモリ: Read only memory）

- 読み出し専用（書き込み不可）
- ファームウェアコードが格納される
- MCUの様々な機能（動作）を制御するプログラムを格納

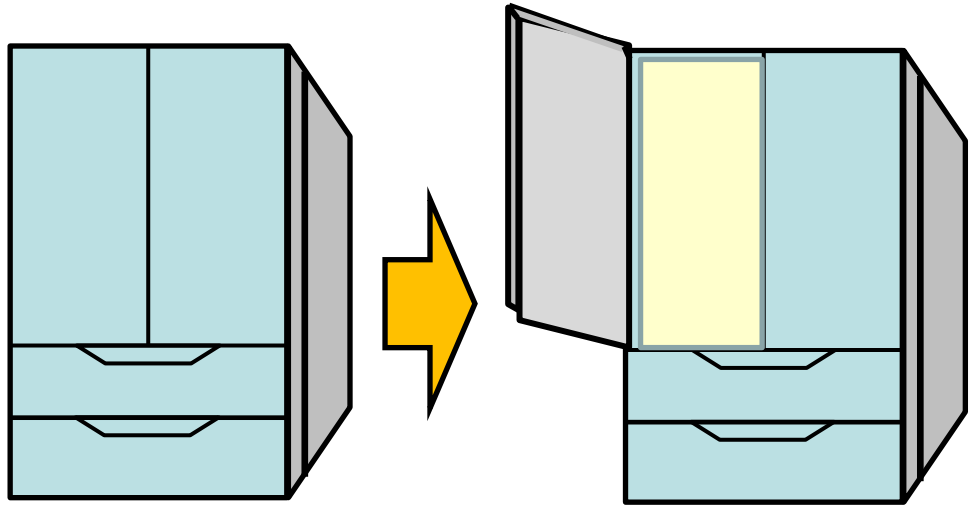
ROMとRAMの役割分担：

プログラムコードをROMに書き込み、実行時にRAMを使って一時データを処理

補足：ファームウェアとは？

マイクロコンピュータ（マイコン）の記憶装置（ROM）にあらかじめ書き込まれたプログラム（ソフトウェア）のこと。

例えば、洗濯機や電子レンジ等の家電製品では、必要なプログラムがファームウェアとして内蔵されている。買ったらすぐに利用可能！

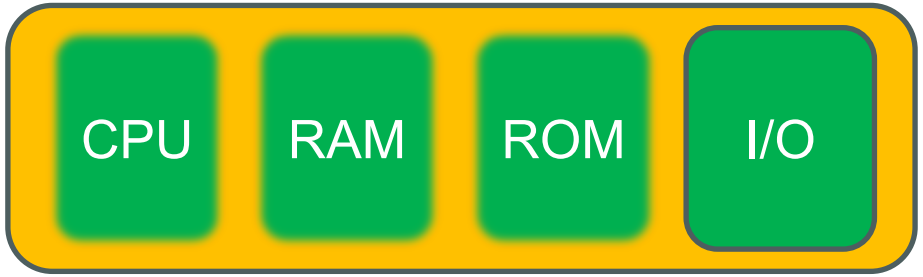


本演習の場合：「スケッチ※1」と呼ばれるユーザープログラムを実行するためのソフトウェアが、Arduino Unoの内蔵ROMにファームウェアとして書き込まれてる。
スケッチ（ユーザプログラム）を描き込むとすぐに実行できるのはそのため。

※1 スケッチ：Arduinoで記述するプログラムのこと。
　アイデアを素早く形にする「下書き」や「設計図」という意味を込めた呼び方

マイクロコントローラの入出力機能(I/O：Input-Output)とは？

- マイクロコントローラ（MCU）が外部世界とやり取りするためのインターフェース



例えば、

- 入力：「温度センサー」などから信号を受け取る（温度等を読み取る）
- 出力：「LEDを光らせる」「モーターを回す」

MCUはI/O（入出力ピン）を介してセンサーやアクチュエーターを用いて、データの読み取り（入力）や装置の制御（出力）を実行

Arduinoとは？

IoTや組み込みシステムの試作検証等に用いられるオープンソースの電子工作プラットフォーム

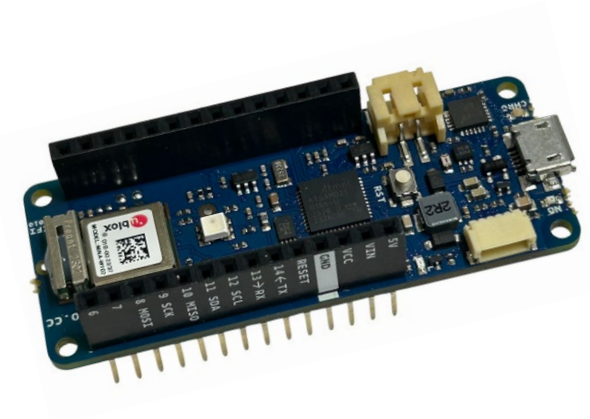
- 多くのセンサーやアクチュエーターと簡単に接続可能であり、それらを制御するプログラムを書いて実行することで動作確認も容易 → IoTシステムを簡易に試作できる。

特徴：

- 簡単に使える開発環境・クラウド
- 多くのライブラリと、活発なユーザーコミュニティの支援
- センサーやアクチュエータと接続しやすい

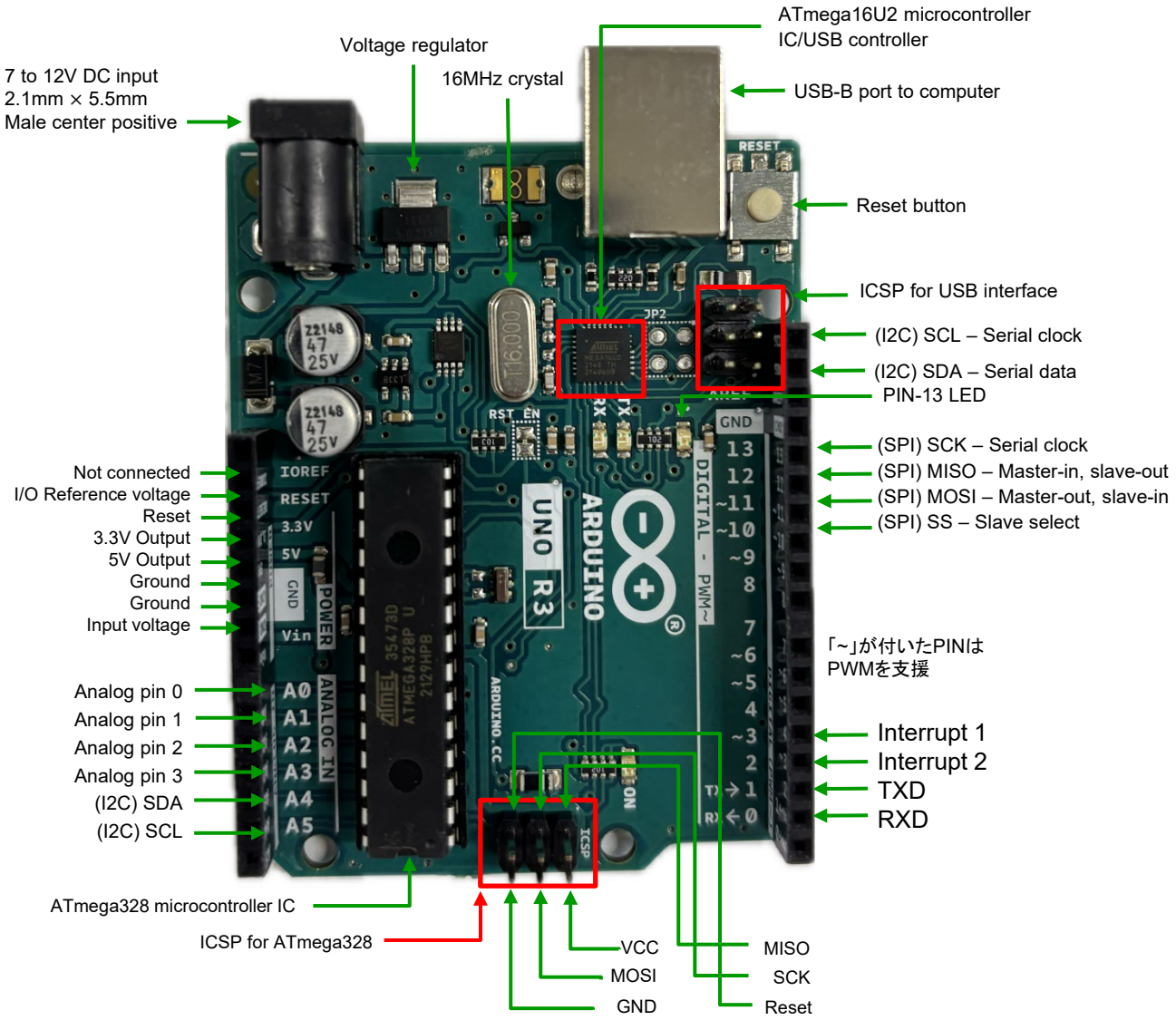
代表的なボード例：

- Arduino Uno　教育・入門用として一般的
- Arduino Nano　コンパクト
- Arduino Mega　IoT用途の拡張性が高い



Arduino Unoの構成

- デジタル I/O：ピン番号：0～13
- アナログ入力：ピン番号：A0～A5
- 電源ピン（Power Pins）
  5V、3.3V、GND などを供給
- USBポート
  プログラミングと電源供給のために使用
- リセットボタン
  電源のオン・オフと同じ効果（スケッチの再スタート）
- ATmega328P マイクロコントローラ
  Arduino Uno の「頭脳」相当の部分



汎用入出力ピン（GPIO：General Purpose Input Output）の基本

Arduinoの「手足」にあたる重要な部分で、外部との通信・制御に用いる。

デジタルピン

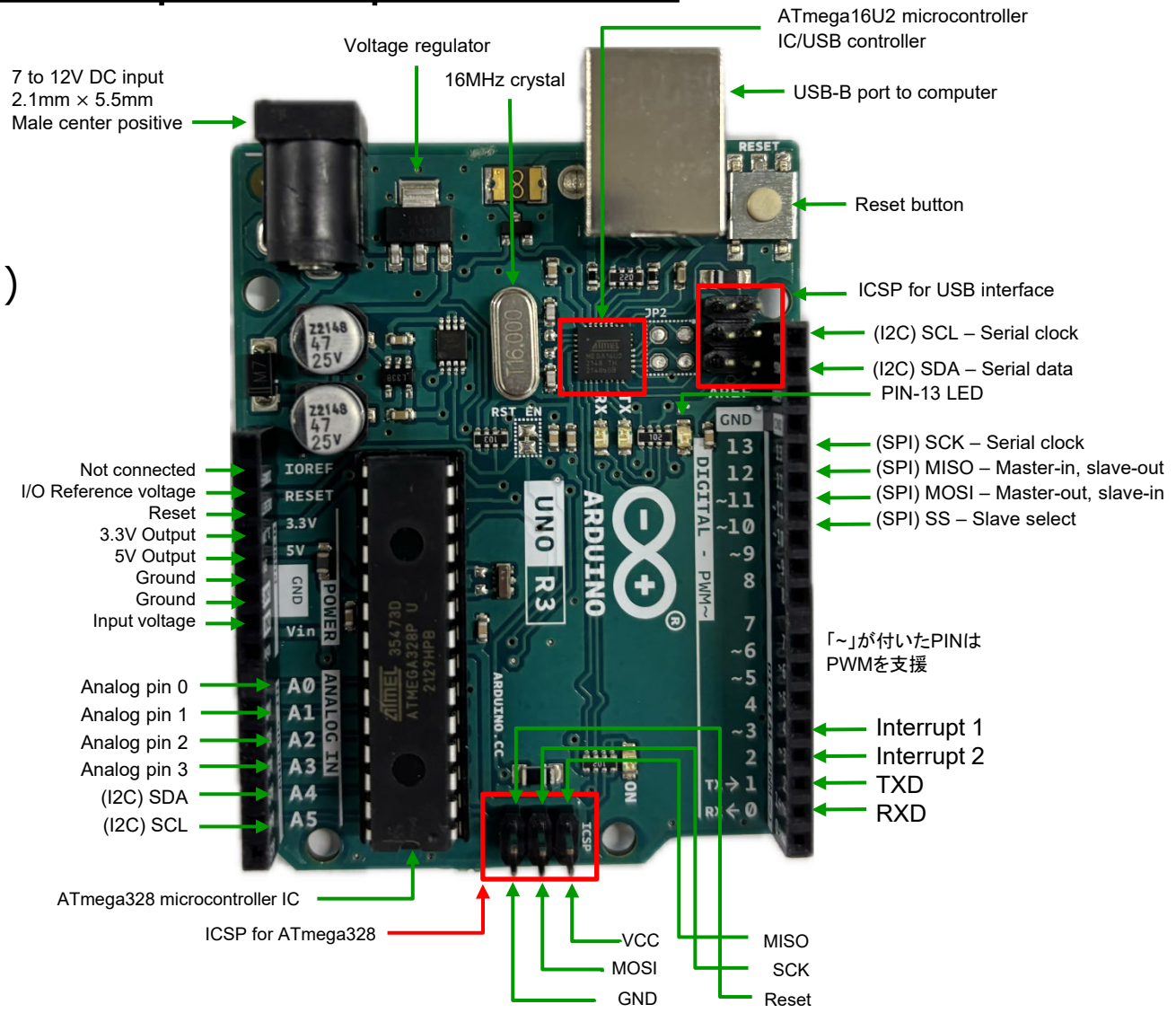
- HIGH / LOW の2値信号を読み書き可能（0または1）
- 例：LEDの点滅制御

アナログピン

- アナログ電圧の読み取りが可能
- 10ビットの分解能で、0～5V を0～1023の値として読み取る
- 例：光センサーの値の読み取り（LDR）

PWMピン

- PWM（Pulse Width Modulation：パルス幅変調）
  ディジタル情報をパルス幅に対応付けたパルス信号を出力
- 例：パルス幅に応じてLEDの明るさ調整する等



MKR WiFi 1010 ボード

※ Arduino Uno よりも高性能、IoT向けの機能を強化した上位モデル

マイクロコントローラ（SAMD21）
- 48 MHz、256 KB フラッシュメモリ、32 KB RAM

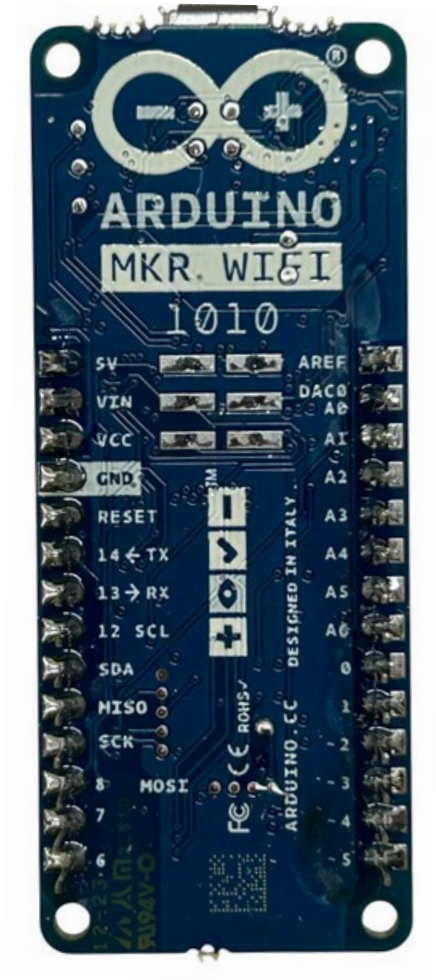
無線モジュール（u-blox NINA-W10）
- Wi-Fi 2.4 GHz、Bluetooth 4.2

デジタル I/O ピン
- 8本（D0～D7）

アナログピン
- 7本のアナログ入力（A0～A6 → 12ビットADC）
- 1本のアナログ出力（10ビットDAC）

PWM出力ピン
- 13本（D0～D8、D10、D12、D18、D19）

電源関連
- Micro-USB：プログラミングおよび5V給電用
- VINピン（5V）
- LiPoバッテリー端子（充電回路を内蔵）



MKR WiFi 1010ボード

主なIoT対応機能

Wi-Fi / Bluetooth内蔵
- 無線プロジェクトへの統合が可能

低消費電力
- バッテリー駆動IoT機器に対応するスリープモード搭載

Arduino IoTクラウド互換性
- 少ないコードでクラウド統合が可能

コンパクトな形状
- 試作やアプリケーション開発に適している

Arduino MKR WiFi 1010

第3章：マイクロコントローラ（Arduino）の基礎

項目	Arduino Uno	MKR WiFi 1010
MCU（マイコン）	ATmega328P (8ビット)	SAMD21 Cortex-M0+（32ビット）
無線通信	外付けの拡張ボードが必要	Wi-Fi / BLE（Bluetooth Low Energy）内蔵
電源管理	基本的な電源機能	LiPoバッテリー対応、低消費電力モードあり
セキュリティ	無し	ECC508 暗号化チップ搭載
コスト	低い	高い

Arduinoの統合開発環境

Arduinoの統合開発環境
（IDE：Integrated Development Environment）

本演習では、IDEを用いて、ユーザプログラム（スケッチ※1）の作成を行います。

詳細は次章以降で説明

※1 スケッチ：Arduinoで記述するプログラムのこと。
アイデアを素早く形にする「下書き」や「設計図」という意味を込めた呼び方

Arduinoの統合開発環境

Arduinoの統合開発環境
（IDE：Integrated Development Environment）

スケッチコードの構成

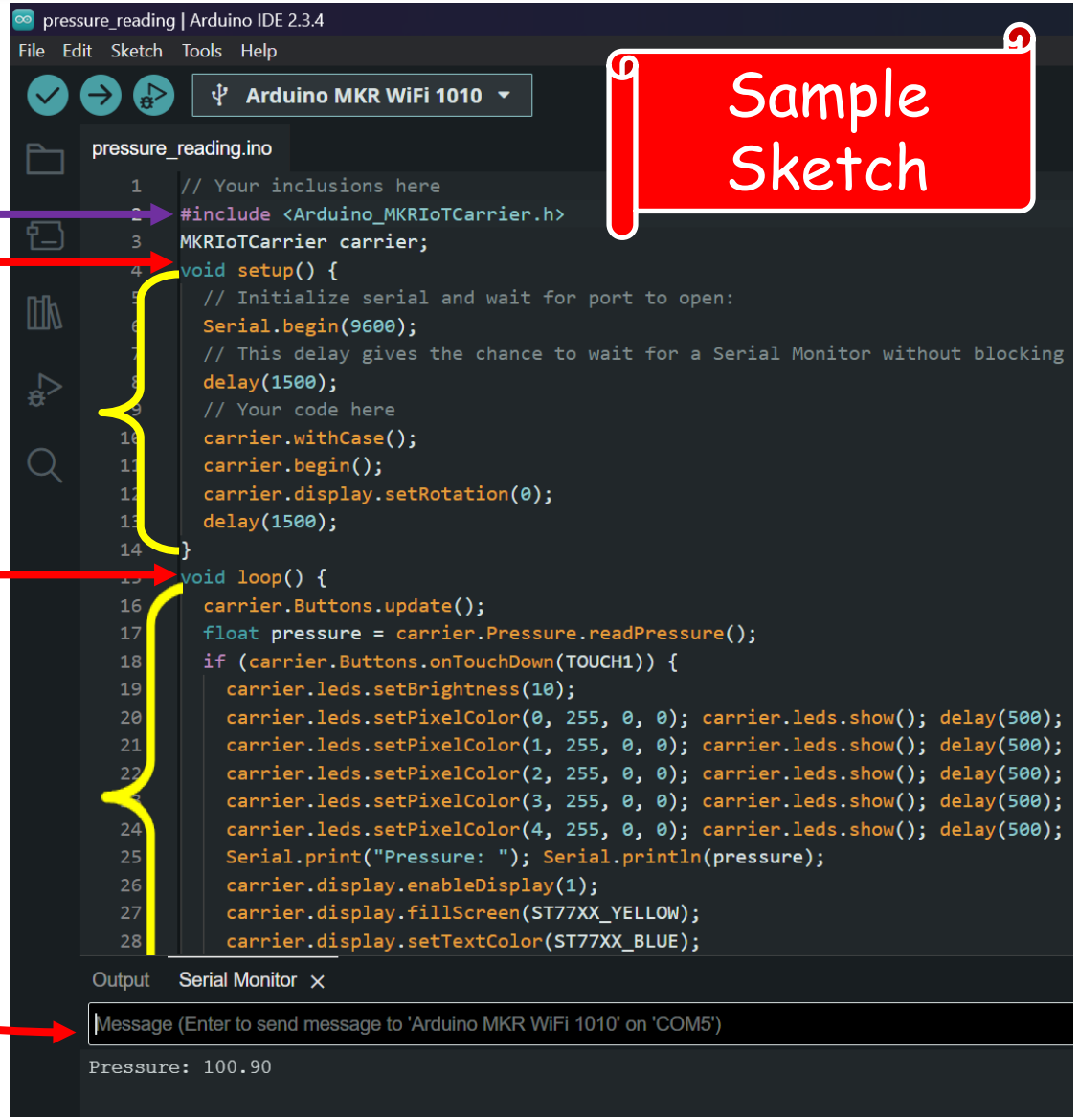
```
void setup() { /* Runs once */ }
void loop() { /* Runs forever */ }
```

ライブラリ：

センサーやアクチュエーター制御用の事前定義されたコード群
例: DHT.h (温度・湿度センサー用), Servo.h (サーボモーター用)

シリアルモニター（Serial Monitor）

デバッグやデータの可視化に使用
例：センサーの値の表示など

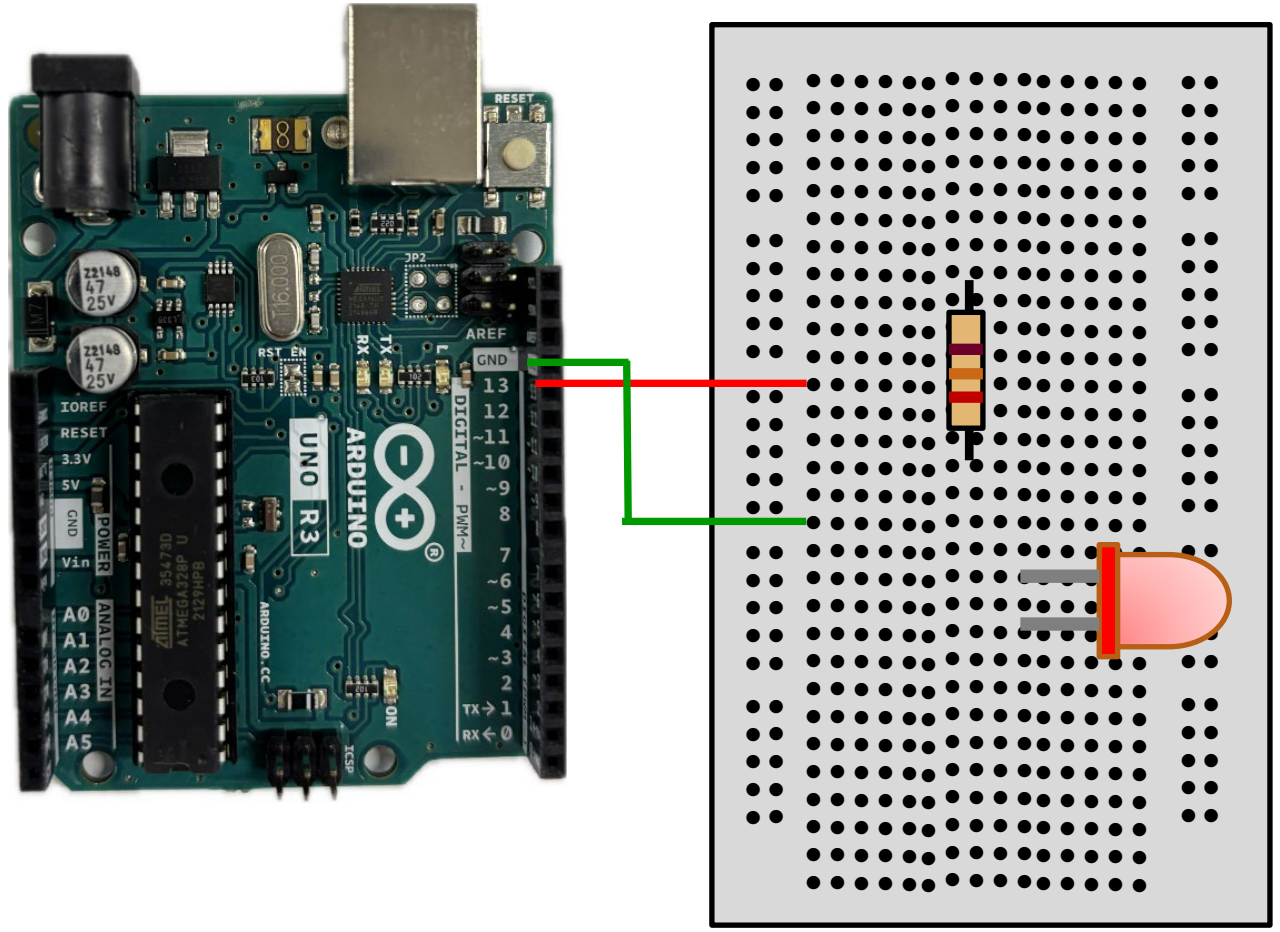


スケッチコードの例① LED点滅

1. LED を 13番ピンに接続（220Ωの抵抗を介して）
2. コードをアップロード

```
void setup() { pinMode(13, OUTPUT); }
void loop() {
  digitalWrite(13, HIGH); delay(1000);
  digitalWrite(13, LOW); delay(1000);
}
```

LEDが交互に点灯・消灯を繰り返す
（1秒間点灯し、次の1秒間は消灯）



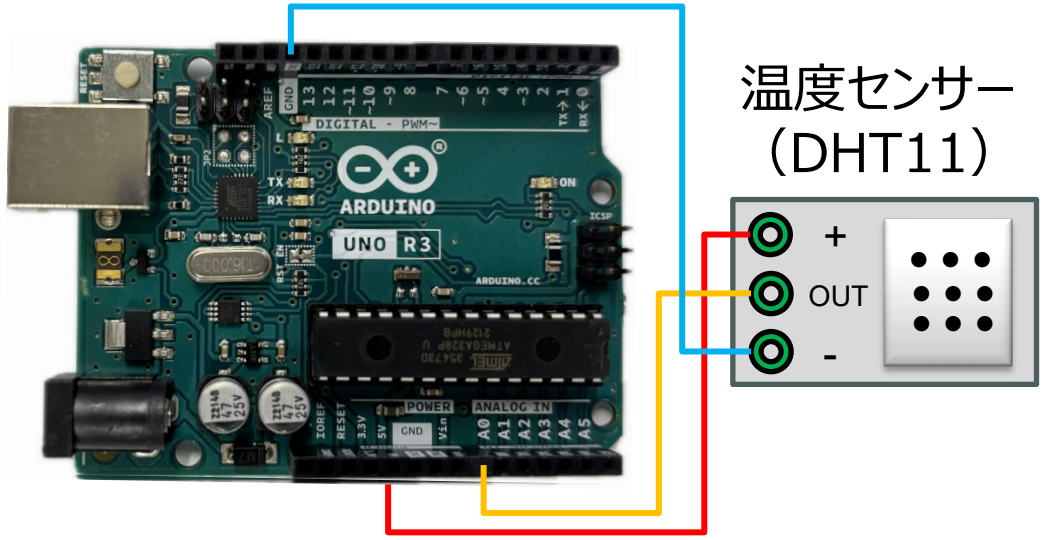
スケッチコードの例② 温度センサーとの接続

例：温度センサー DHT11 を使う場合

1. DHT11 をピン2に接続
2. DHTライブラリをインストール
3. コードをアップロード

```
#include <DHT.h>
DHT dht(2, DHT11);
void setup() { Serial.begin(9600); dht.begin(); }
void loop() {
  float temp = dht.readTemperature();
  Serial.print("Temp: "); Serial.println(temp);
  delay(2000);
}
```

DHT11はデジタルセンサー
➔ アナログピンではなく、
デジタルピンに接続する



1. デバッグ用にシリアル通信を初期化し、センサーモジュールをスタート
2. 温度を読み取り、2秒ごとにシリアルに記録・表示

まとめ

本章では、Arduino Unoを中心としたマイクロコントローラボードの特徴、CPU・メモリ・I/Oなどの基本構成について述べた。

４章以降では、具体的なプログラムの作成等について学習する。