WEEK 1 事前補講

補講
電子回路の基礎

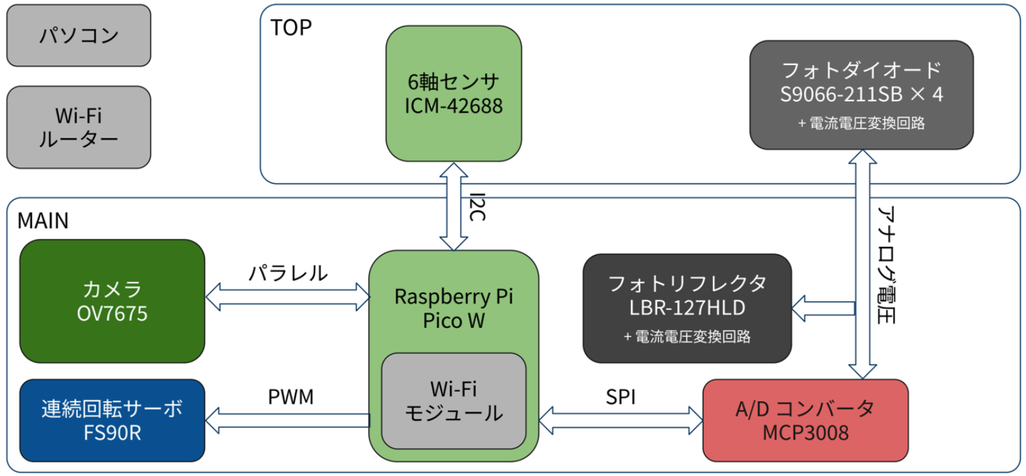
ソフトを書く前に、まず「動かす対象」であるハードウェアを知る

回路図が読める・基本部品の役割が分かる・マイコンの全体像がつかめる —— ゴールは「マイコンの役割がイメージできる」状態

0.1 今回の位置づけ

いまは「原理を理解する」フェーズの入り口

- Week 0 で共有：本ゼミのテーマは
  「根拠を持ったエンジニアリング」
- 最終ゴールは、太陽方向を検知し姿勢を制御し撮影し、画像を地上局へ送る**模擬衛星**
- Week 1 は Pico で「Lチカ」を深追いし、ソフトがハードを叩く境界を見る
- 今回の補講は、その Week 1 で**前提となるハードの土台**を先に固める



今回の最終到達目標＝模擬衛星システムの全体像

0.2 ソフトウェア開発で電子回路を学ぶ目的

Lチカは意味を知らなくても動く。だが「わかって書く」力を、今のうちに

Lチカのコードは**真似れば動いてしまう**。だが実際には**「特定のピンに電圧を出せ」とハードへ命令している**――動く理屈は物理の側にある。

1行のコード
値を書く
`gpio_put(PIN, 1)` と命じる

→

レジスタ
メモリへ書き込み
特別なメモリのビットが変わる

→

ピン
電圧が変化
0V → 3.3V へ

→

LED
点灯
物理現象として現れる

この先の姿勢制御や通信は真似だけでは動かない。一番やさしいLチカのうちに「コードが起こす物理現象まで分かって書く」型を身につける ―― 「根拠を持ったエンジニアリング」のソフトから見た入り口

0.3 今日の流れ

各部品の役割を押さえ、最後にマイコンの全体像へ

① **回路の表現**
配線図と回路図

② **電子部品**
抵抗・ダイオード・
コンデンサ・MOSFET

③ **部品の繋げ方**
ブレッドボードと配線

④ **デジタルとアナログ**
信号の2種類

⑤ **CMOSデジタル回路**
マイコンの中身と入出力

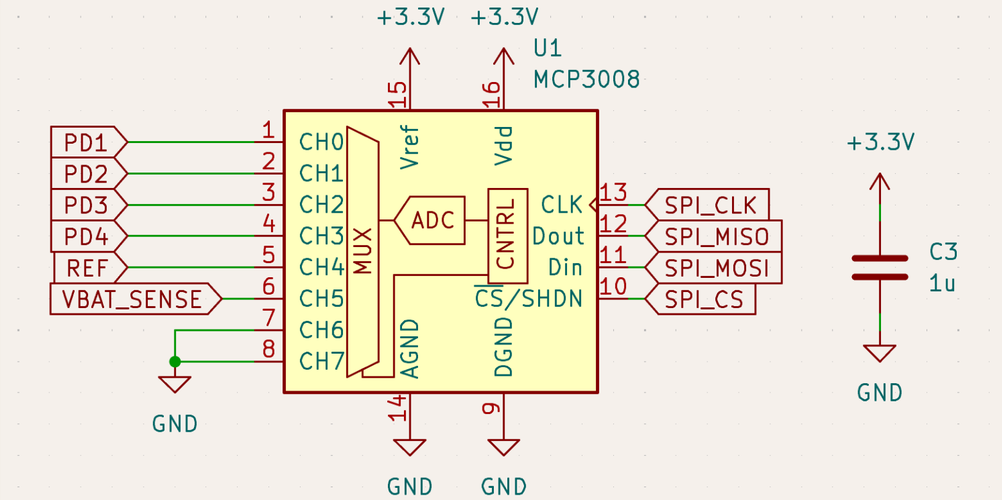
⑥ **マイコンの全体像**
ソフトとハードの接点

① → ⑥ の順に、部品・信号・マイコンへと進んでいく

SECTION 1 ／ 回路の表現方法

回路図は接続情報だけを描いた図（Schematic）

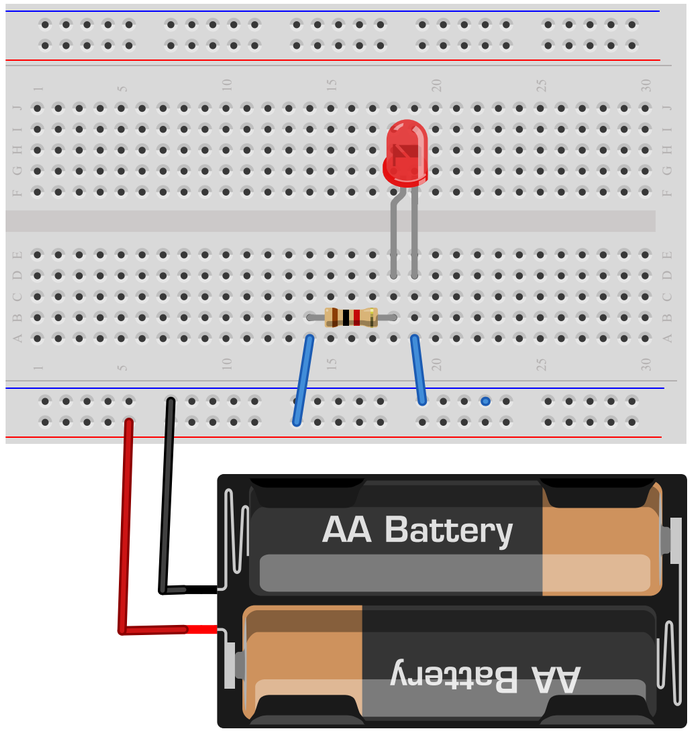
- 資料やデータシートは、すべて**回路図**で書かれている
- 現物を見なくても**「何と何が繋がっているか」**が分かる。読めないと、データシートから情報を取り出せない
- 右は模擬衛星のADC（Analog-to-Digital Converter、MCP3008）。**記号だけで全部表現**されている



実際に使うADC（Analog-to-Digital Converter、MCP3008）の回路図。+3.3V・GND・SPI・コンデンサが記号で描かれている

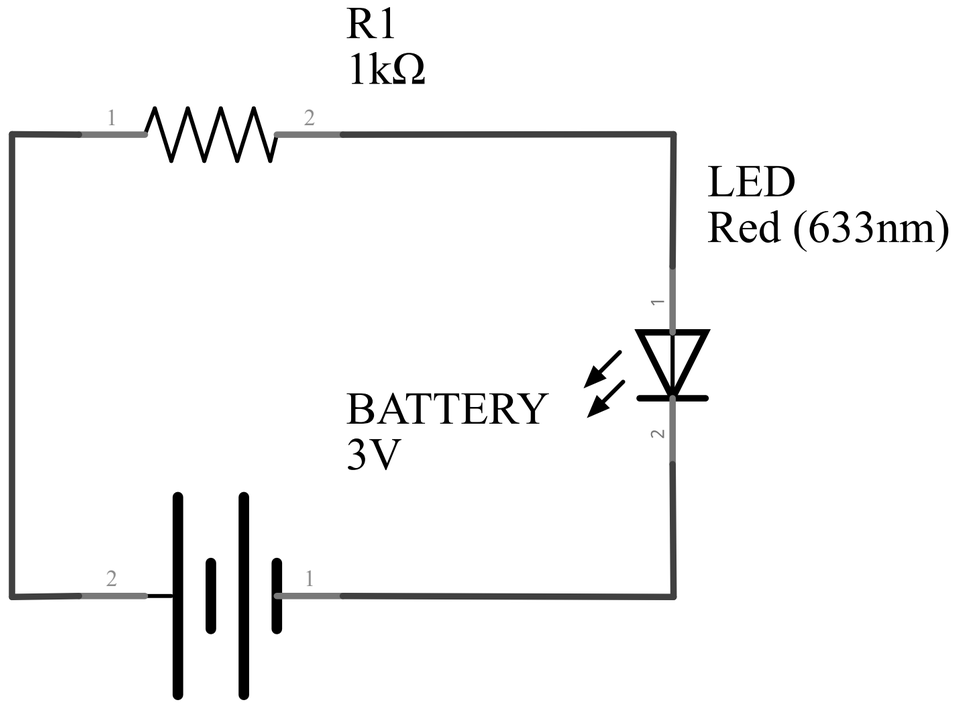
見た目（実体）と論理（回路図）は別物

実体配線図 Layout



部品の形・位置・配線の取り回しまで、現物に忠実に描く。

回路図 Schematic



記号に置き換え「電気的に何と繋がるか」だけを描く。
線の長さや配置に意味はない。

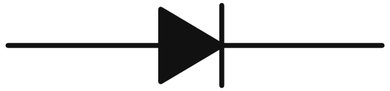
1.2 回路記号

部品を抽象化した「記号」で描く

部品ごとに決まった図記号がある

抵抗

コンデンサ



ダイオード

記号を使うから、誰が描いても・どの国でも同じ回路図として読める

同じ「抵抗」でも規格に新旧がある

新JIS（C 0617）
IEC準拠・長方形

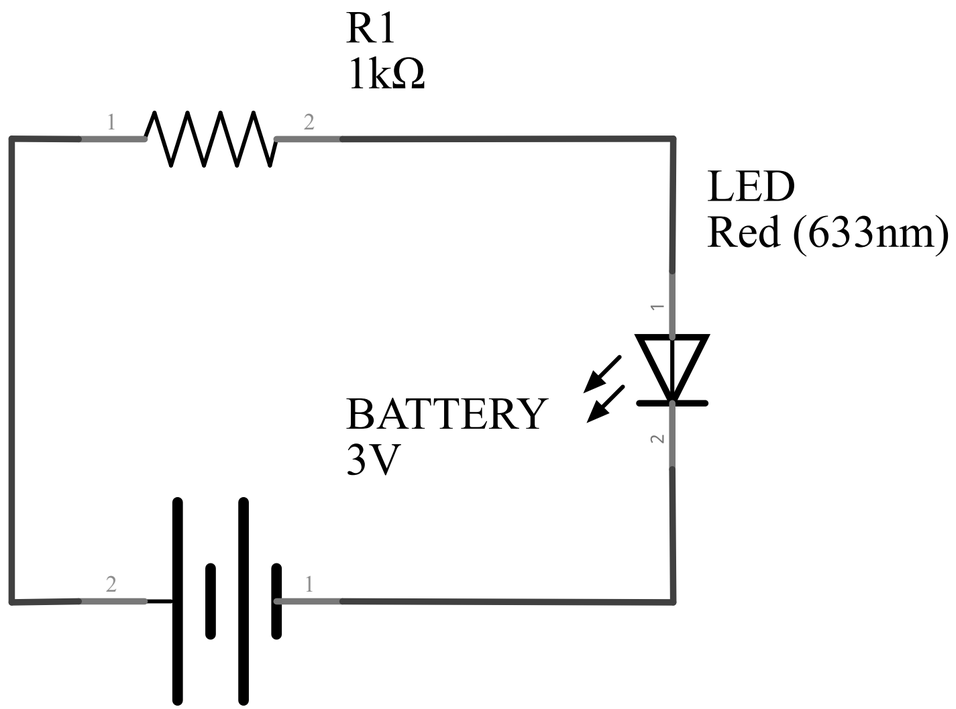
⇔

旧JIS（C 0301）
ギザギザ

記号には「方言」がある。どちらを見ても戸惑わないように

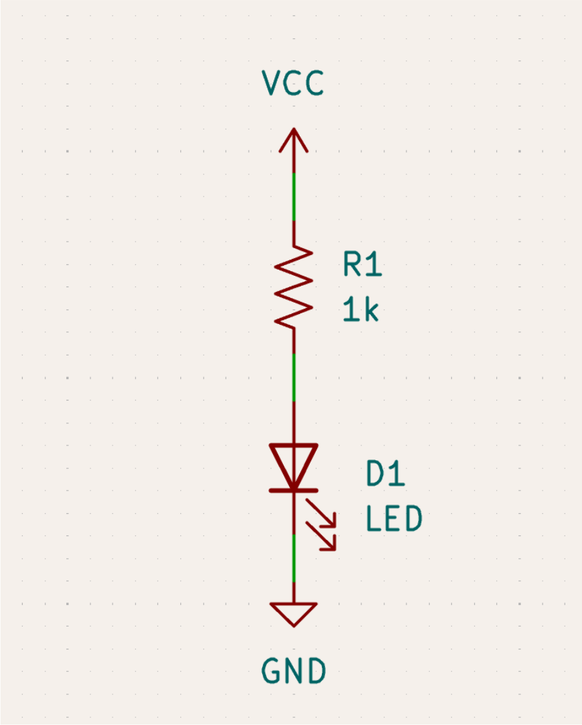
ループを VCC と GND で切り離して描く

一周のループ



電池の＋から出た電流が、抵抗とLEDを通り−へ帰る。
電流は閉じたループでしか流れない。

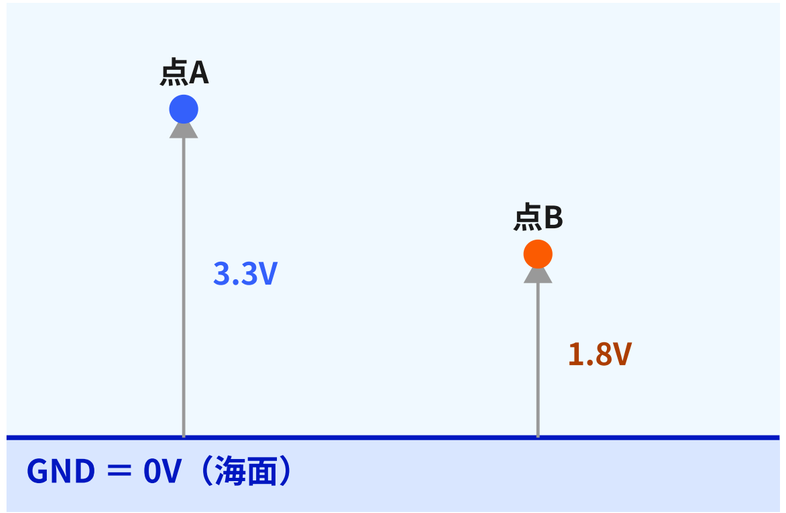
VCC / GND で分離



同じ VCC どうし・同じ GND どうしは、線で結ばなくても
繋がっているとみなす。「ここで繋がっている」という表現

電圧は「基準からの高さ」＝相対的な量

- 電圧は一点では決まらず、必ず**2点間の電位差**として決まる
- 回路では **GND を 0V の基準**と決め、そこからの高さで各点を測る
- 比喩：標高は**海面を0m**と決めた基準からの高さ。GNDがその海面、各点の電位が標高
- GNDは絶対的な0ではなく「どこを0Vとするか」で決まる
  相対的な基準である

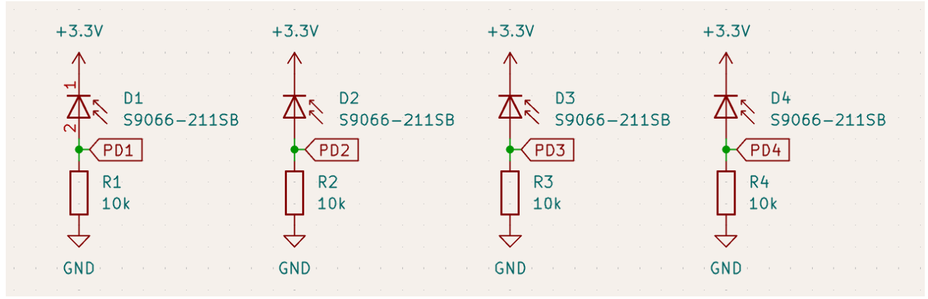


GND（海面）からの高さとして各点の電位が決まる。
基準を変えれば同じ点の数値も変わる。

ネット＝繋がりの「実体」、ラベル＝それに付ける「名前」

- **ピン**＝部品の端子、**ネット**＝ピンどうしの電気的な繋がり、**ラベル**＝そのネットに付けた名前
- ネットは名前がなくても「繋がり」として実在する。同じネットの上はどこも同じ電位
- **同じ名前のラベルどうしは、線で結ばなくても1つのネット**（＝電気的に繋がっている）とみなす
- VCC・GND もラベルの一種。信号線も同じく名前で分離して描く

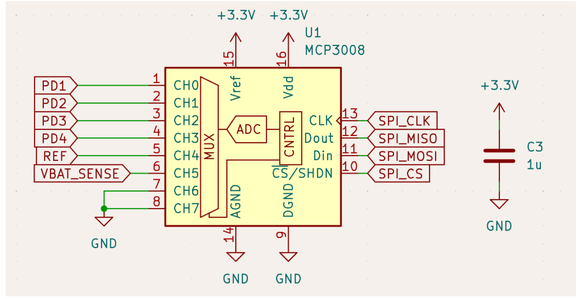
フォトダイオード（出力側）



PD1
PD2
PD3
PD4

同じネット

ADC・MCP3008（入力側）



左の出力 **PD1〜PD4** と右の入力 **PD1〜PD4** は、線で結ばれていなくても **同じラベル＝同じネット** だから繋がっている

まとめ 1.6 この章のまとめ：ここは覚えて次へ

回路図の章の核は、この3つ

①

回路図は「接続の地図」

「何と何が電気的に繋がっているか」だけを記号で描く。実体配線とは別物

②

GND ＝ 電圧の基準点（0V）

電圧は2点間の電位差。GNDを0Vと決めて、そこからの高さで測る

③

同じラベル ＝ 同じネット

同じ名前のラベル（VCC・GND・PD1…）は、線で結ばれていなくても繋がっている

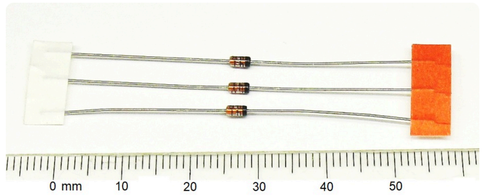
SECTION 2 ／ 電子部品

模擬衛星の基本部品を揃える

部品は「種類」ではなく**「何のために置かれているか」という用途**で覚える。この4つの役割を一言で言える状態を目指す。

抵抗

電流の流れにくさを決める

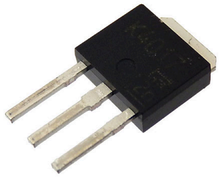


ダイオード

電流を一方向にだけ流す

コンデンサ

電気を一時的に貯める



MOSFET

電気で動くスイッチ

電流の「流れにくさ」を決める部品

- 電流が流れにくくなる＝**水道の管を細くする**イメージ
- 流れにくさの単位は**オーム［Ω］**
- 用途：電流を絞る／電圧を分ける／「基準電位の固定」（Week 2 で扱う）
- 模擬衛星でも、通信線のプルアップやセンサまわりなど**至る所で使う**

一般的な炭素皮膜抵抗

2.1.1 オームの法則

V = I × R ―― 一番よく使う式、覚える価値がある

$$V = I \times R$$

V	I	R
電圧［V］	電流［A］	抵抗［Ω］

I = V ÷ R 電流を求める

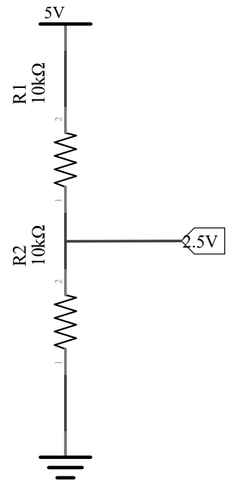
R = V ÷ I 抵抗を求める

P = V × I 消費電力［W］。発熱で壊れないかの判断に使う

直流でも交流でも、抵抗についてはこの関係が成り立つ

抵抗2つで、電源電圧を「比」で分け合う

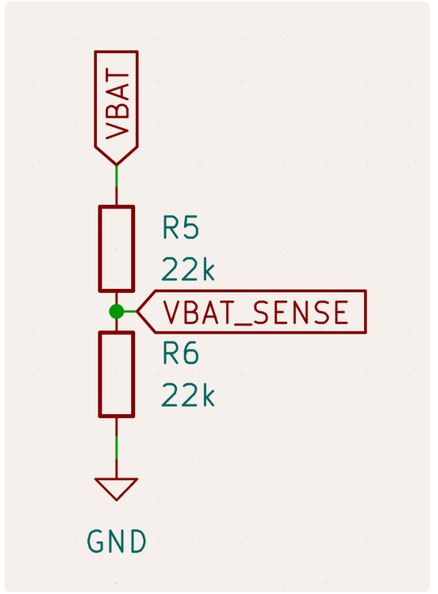
- 直列につないだ2つの抵抗には、**同じ電流 I** が流れる（枝分かれのない一本道）
- それぞれの電圧はオームの法則で決まる：`V1 = I×R1`、`V2 = I×R2`
- 中点から取り出す電圧は **`Vout = Vin × R2 ÷（R1＋R2）`**
- 例：5V を 10kΩ と 10kΩ で分けると、中点はちょうど**半分の 2.5V**
- **抵抗だけで狙った電圧を作れる**（掛け算と割り算だけで計算できる）のが要点



R1・R2 の中点から Vout を取り出す分圧回路

電池電圧を分圧してから ADC で測る

- マイコンが電圧を数値化する**ADC**には測れる上限がある（模擬衛星は3.3V系）
- 電池電圧(VBAT)はこの上限を超えることがあり、そのままでは測れない（範囲外・ピンを傷める恐れ）
- **22kΩ の抵抗2本で1:1に分圧**し、半分にした VBAT_SENSE を ADCへ渡す
- ソフトで読んだ値を**2倍**すれば元の電池電圧が分かる（分圧比が既知だから復元できる）
- 測れない大きさを、**既知の比で測れる大きさに落とす**

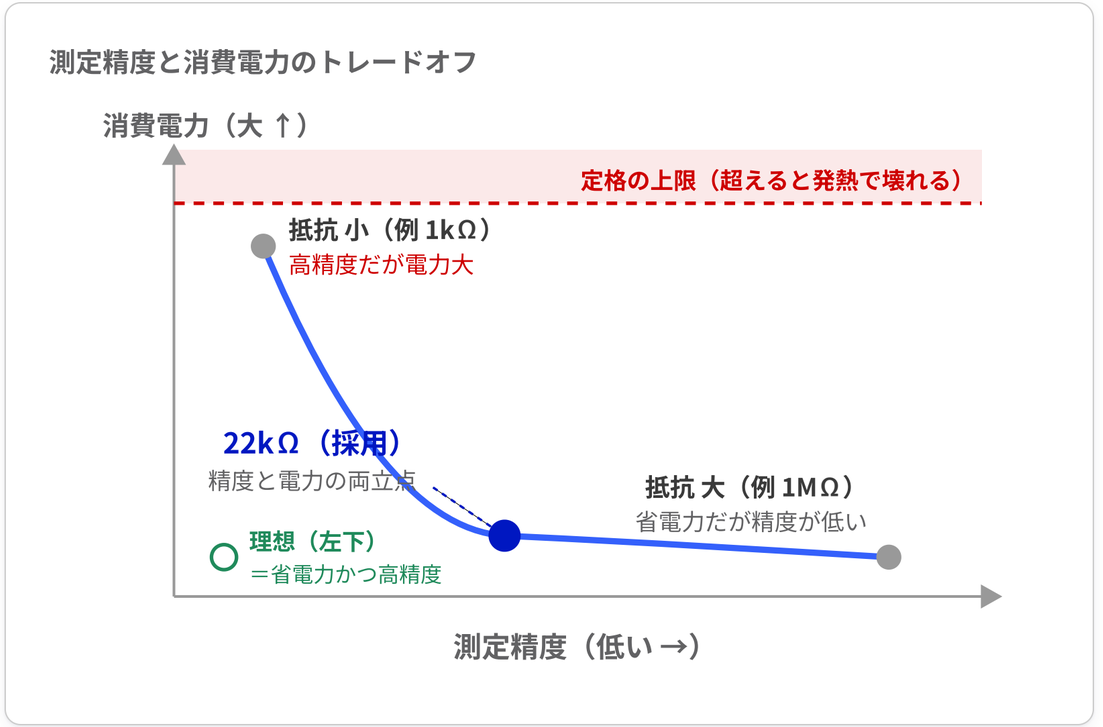


模擬衛星MAIN基板：VBAT を 22kΩ 2本で分圧し、VBAT_SENSE として ADC へ入れている

参考 2.1.4 分圧の抵抗値は「トレードオフ」で決まる

定格・消費電力・寄生成分の綱引きで「ほどよい大きさ」に

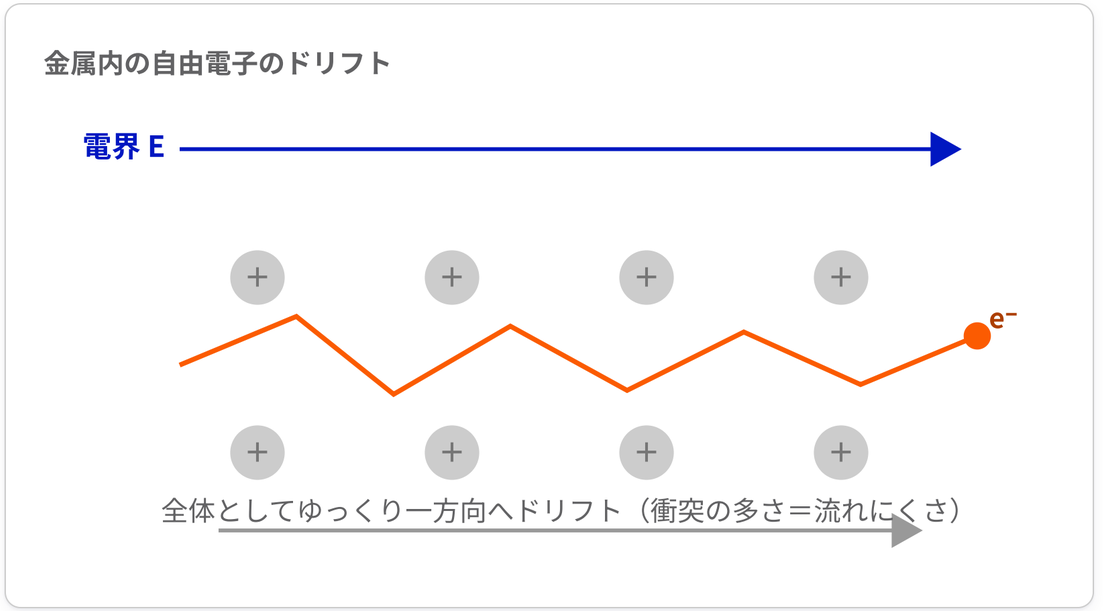
- 大前提：どの抵抗も**定格（耐えられる電力）の範囲内**で使う（P = V² ÷ R）
- 大きくしたい：分圧抵抗には常に電流が流れ続ける → 大きいほど**無駄な消費電流が減る**（電池で動く模擬衛星では効く）
- 小さくしたい：大きすぎると配線やADCの**わずかな漏れ（寄生成分・2.6）**に負けて測定値がずれる
- 綱引きの結果、模擬衛星では **22kΩ** という「ほどよい大きさ」に落ち着いている
- 「動く」だけでなく**「壊れない・狂わない」**かまで見るのが、根拠を持つ設計



参考 2.1.5 オームの法則は「経験則」

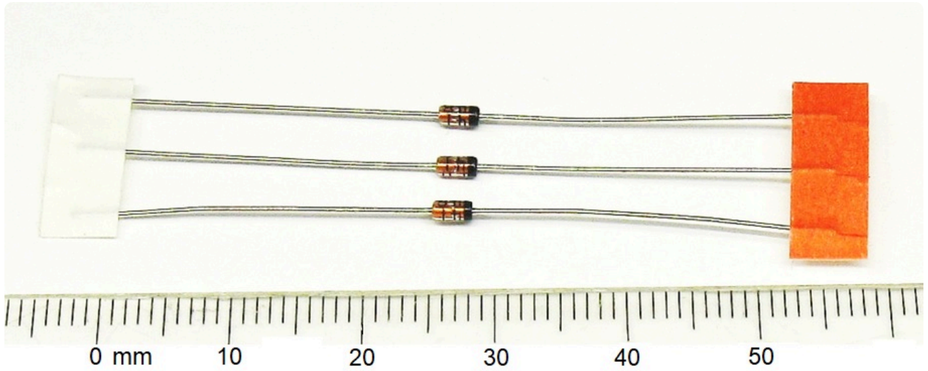
V = I × R は万能の真理ではなく近似

- 電磁気の根本原理は**マクスウェル方程式とローレンツ力**（深入りしない）
- オームの法則は、金属など**「ふつうの物質」で成り立つ経験則**
- 電界に押された電子が格子に衝突しながら進む。衝突のしやすさが抵抗（古典的なイメージ）
- **半導体（まさにLED）では成り立たない** → 電圧に比例せず、特殊な曲線を描く



電気の「逆流防止弁」

- **順方向**（決まった向き）には流すが、**逆方向**には流さない
- **極性（向き）**があり、つなぐ向きを間違えると働かない
- 用途：電源の逆接続からの保護、整流（向きをそろえる）
- 「一方向にしか通さない」ことが、この部品の存在意義



整流・スイッチング用の定番ダイオード 1N4148

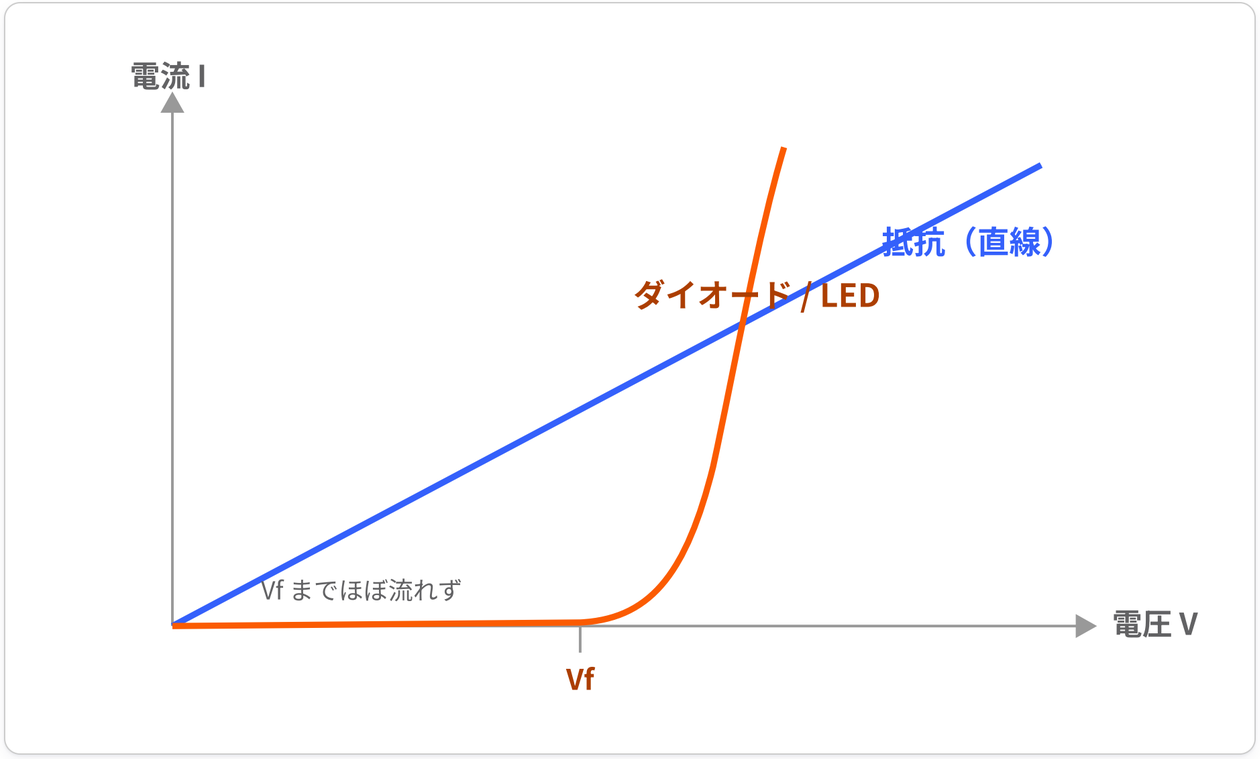
LEDは、電流を流すと光るダイオードの一種

- LED ＝ **Light Emitting Diode**。ダイオードなので当然、極性がある
- 順方向に流したときだけ光る（逆だと光らないし、流れない）
- **順方向電圧 Vf** を超えてはじめて電流が流れ始める（この Vf が、後の**電流制限抵抗の計算**で効いてくる）
- 模擬衛星のサンセンサ（フォトダイオード）もダイオードの一種

この後 2.2.3 の計算に使う赤色LED OSR5JA3Z74A（秋月電子で人気）

2.2.2 IV特性

抵抗は「直線」、ダイオードは「折れ曲がる曲線」



- **抵抗：**電圧に比例して電流が増える直線（オームの法則どおり）
- **ダイオード／LED：**Vf までほぼ流れず、超えると急に立ち上がる曲線

この**急な立ち上がり**が、抵抗で電流を制限しないと過大電流が流れる理由

2.2.3 計算例

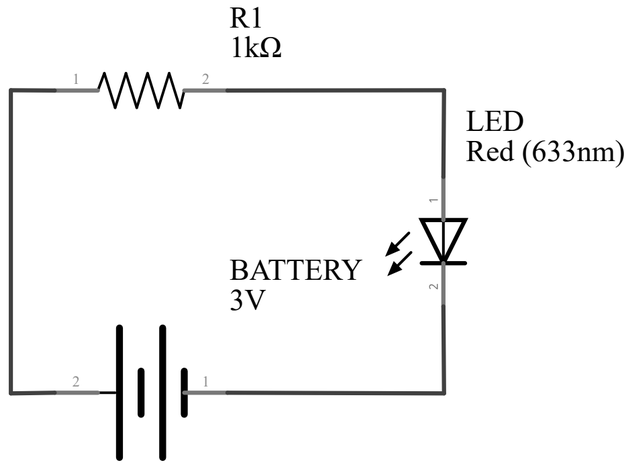
LED電流制限抵抗を、オームの法則で決める

問題：抵抗を入れずにLEDを3Vへ直結したら、どうなる？

例：赤色LED「OSR5JA3Z74A」。電源 E=3V、順方向電圧 Vf≈2.0V、流したい電流 I=1mA

抵抗の電圧　**3V − 2V = 1V**　（LEDが2V受け持ち、残りが抵抗に乗る）

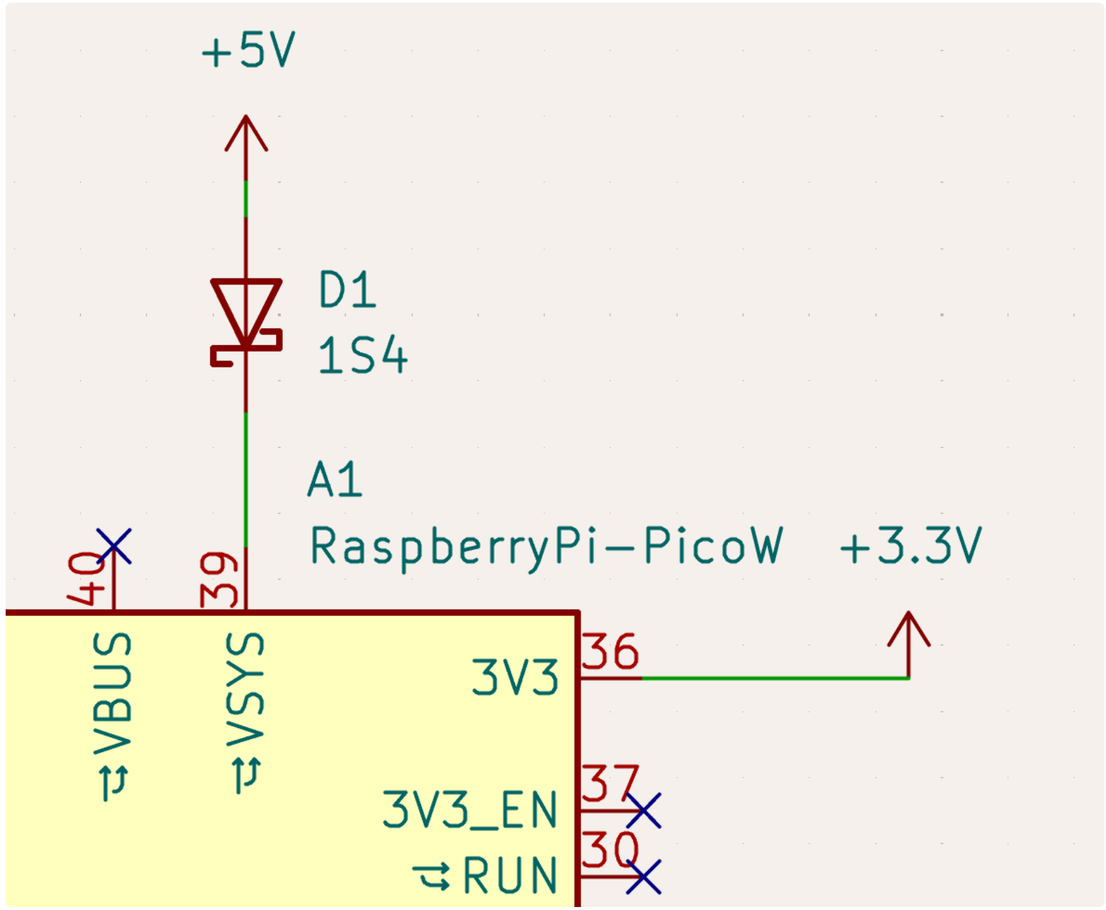
抵抗値 R　**1V ÷ 0.001A = 1kΩ**



電源・抵抗・LEDが一周のループになっている

電源の逆流を防ぐ（Picoボード上）

- 信号の一方通行だけでなく、**電源の逆流の防止**にも使える
- Pico W の電源入力では、**+5V → D1 → VSYS（電源）**へ入る
- このダイオードは、電源の逆流や競合から回路を守る役割
- 「一方向にだけ流す」性質が、そのまま**電源を守る**働きに



Raspberry Pi Pico W の電源入力回路（実際の回路図）

電気の「小さな貯水タンク」

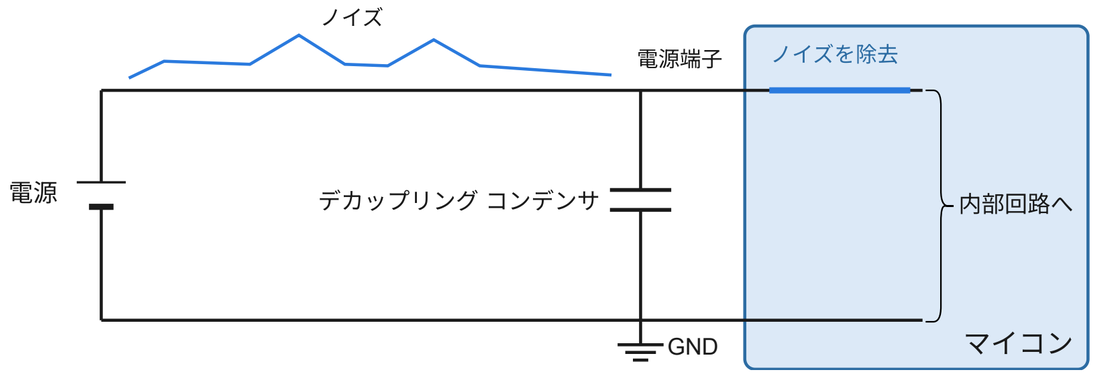
- 電荷（電気）を貯めたり放出したりできる。容量の単位は**ファラド［F］**
- 比喩：**水を貯めるタンク**。急に水が要るとき、一時的に供給できる
- 直流（一定の電圧）はある程度貯めると流れなくなるが、**変化には敏感**に応じる
- 模擬衛星では主に**電源を安定させる**ために使う

アルミ電解コンデンサ

2.3.1 コンデンサの主な用途

電圧のふらつきを吸収して回路を安定させる

- マイコンやセンサは、動作の瞬間に**消費電流が急変**する
- 電源ラインの近くにコンデンサがあると、急変を吸収して電圧を保つ
- 比喩：バケツリレーの途中に貯水槽を置くと、波が穏やかになる
- この用途のコンデンサを**デカップリングコンデンサ**と呼ぶ（ICの電源ピンのすぐ横で、電源の変動から切り離す）

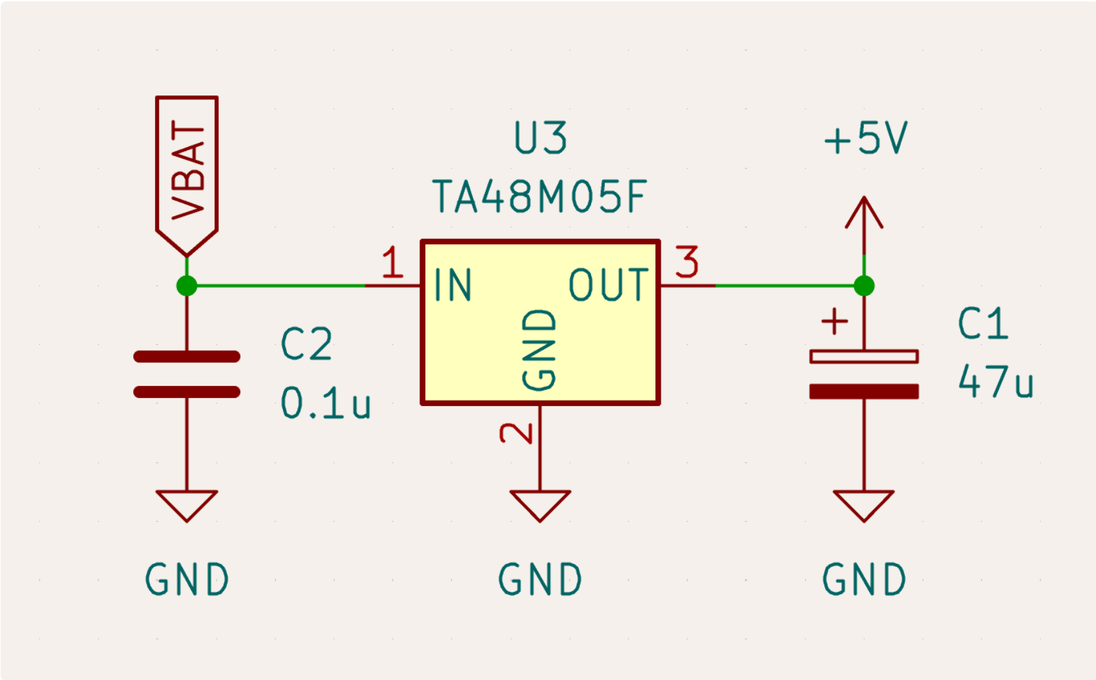


電源とGNDの間のデカップリングコンデンサが、リプルやノイズを吸収してマイコンへ安定した電源を渡す

※バイパスコンデンサは高周波ノイズをGNDへ逃がす別の役割。構造は同じで「どこで何をするか」で呼び名が変わる

LDO出力の安定化に 47μF が入っている

- 模擬衛星は **電池 → LDO**（低ドロップアウトレギュレータ）で安定した電源を作る
- そのLDOの出力段に **47μF の電解コンデンサ**を置いて出力を安定させる
- 電解コンデンサは**極性あり**なので向きに注意（逆だと壊れる。ダイオードと同じ）
- 「電源を作る所にコンデンサあり」を、実際の回路で確認できる



模擬衛星の LDO 電源回路（出力段に電解コンデンサ）

2.3.3 コンデンサの種類

いまは「2タイプある」で十分

無極性 セラミック

向きの区別なし。扱いやすい

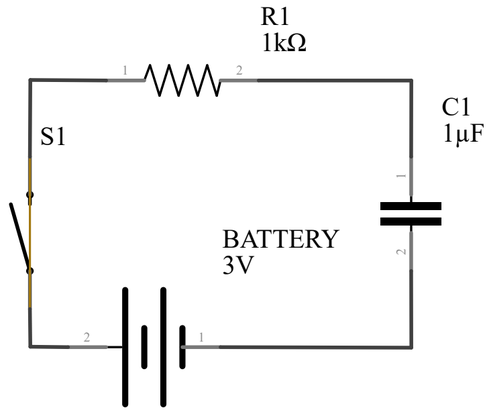
有極性 電解

逆向きにつなぐと壊れることがある。向きに注意

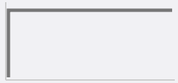
選定は模擬衛星ではハードが設計済みなので理解しなくてよい。将来選ぶ場面が来たら、データシートや定番構成を見ればよい

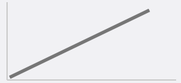
QUIZ 2.3.4 抵抗とコンデンサをつなぐとどうなる？

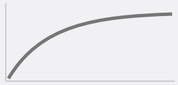
充電時、コンデンサ電圧はどんな曲線で立ち上がる？



RC回路。コンデンサは最初 0V。ヒント：空のタンクに水を入れ始めると…

A  垂直に跳ね上がる

B  直線的に上がる

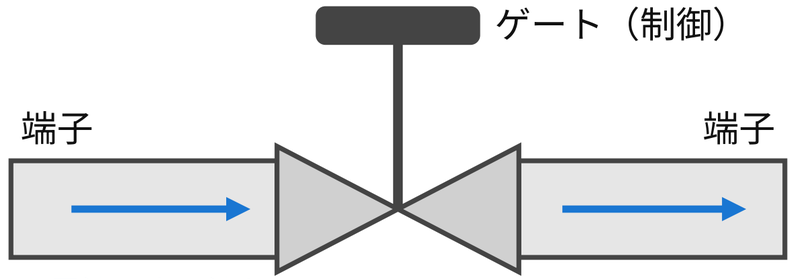
C  なだらかに飽和

D ずっと 0V のまま

2.4 MOSFET

電気信号で ON / OFF できるスイッチ

- 3本の端子。うち**1本（制御端子）の電圧**でON/OFFを切り替える
- ここで H／L を使う：デジタルは電圧を2段階——
  H（High＝電源電圧の側）と L（Low＝GNDの側）——で表す
  （詳しくは後半で）
- 制御端子を H にすると2端子間が導通、L にすると遮断（タイプにより逆もある）
- 物理スイッチと違い、マイコンの信号で**高速にON/OFF**。まずは**「電圧で操作する蛇口」**と理解すればよい



ハンドル＝ゲート（制御端子）。H で開いて電流（水）が流れ、L で閉じて止まる

2.4.1 Nch と Pch

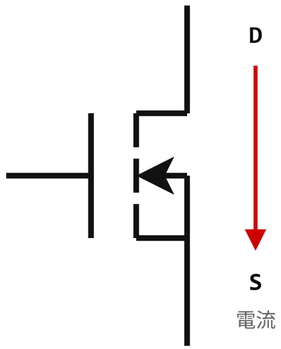
2種類あることが CMOS の鍵

Nチャンネル

H で導通

G ゲート

H/L で ON/OFF を切り替える入力

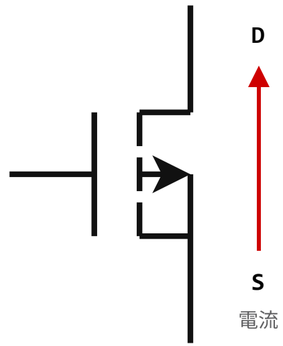


Pチャンネル

L で導通

G ゲート

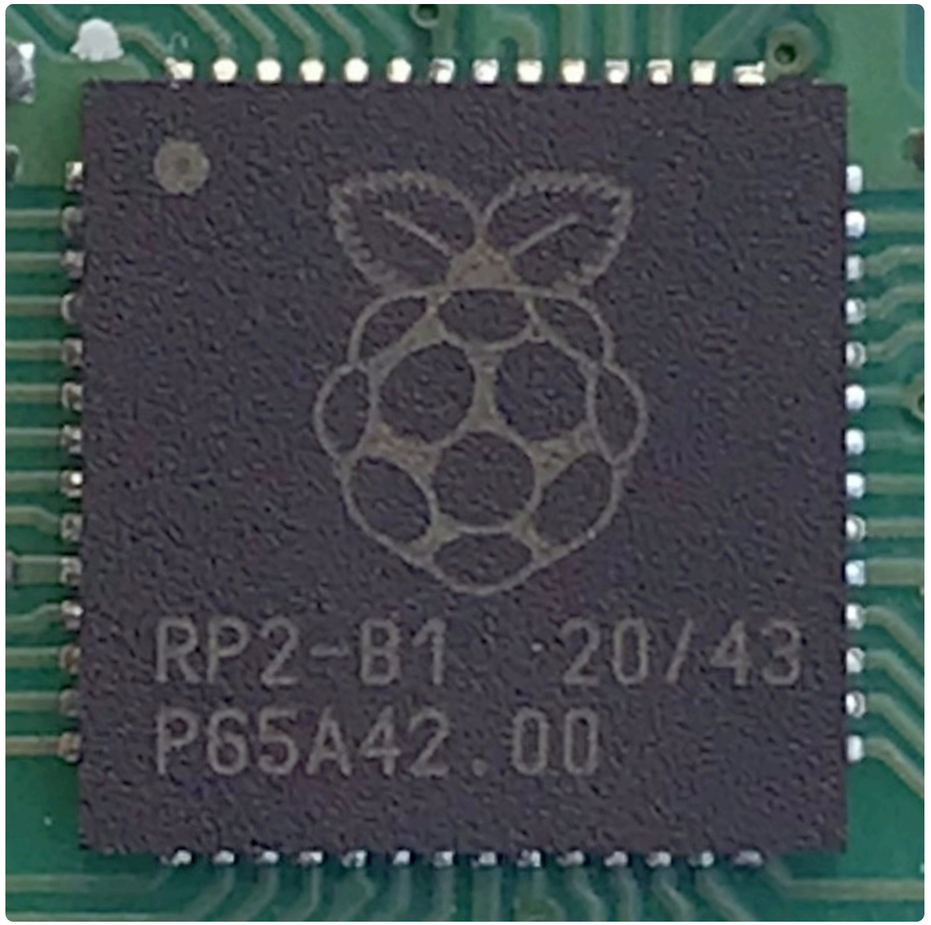
H/L で ON/OFF を切り替える入力



NchとPchは**互いに逆の動作**。この「逆の性格をもつペア」が、後で出てくる CMOS（Complementary MOS）の材料になる

基板上になく、マイコンの「中」にある

- 模擬衛星の基板に、
  単体（ディスクリート）のMOSFETは載っていない
- だが Pico の中身（RP2040）は、
  無数のMOSFETで構成された CMOS
- つまり**「意識しないが、最も大量に使っている部品」**がMOSFET
- 今日は回路記号と「スイッチ」のイメージだけ押さえれば十分



模擬衛星の頭脳 RP2040。この中身が無数のMOSFET（CMOS）

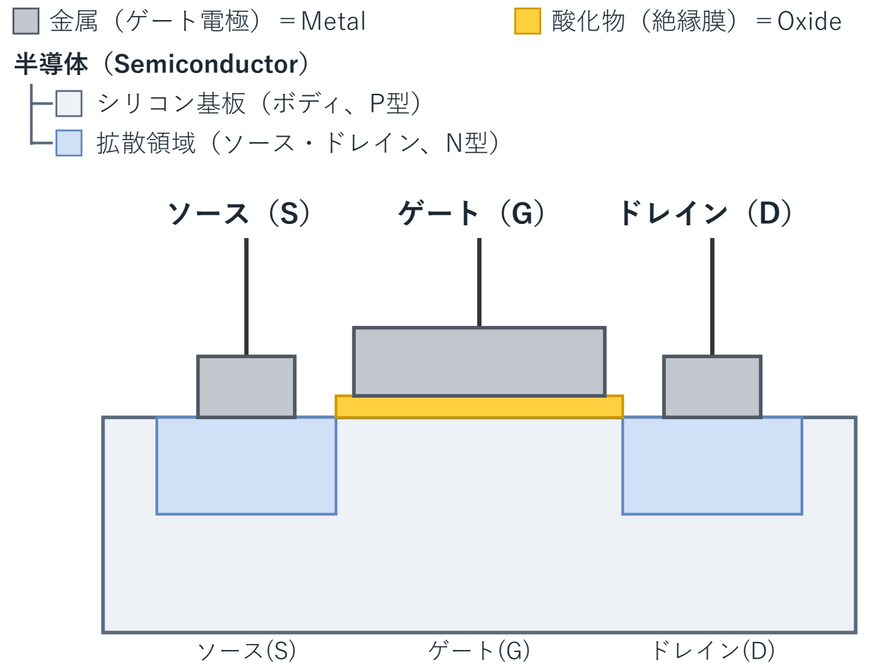
参考 2.4.3 なんでMOSFETっていうの？

名前が、構造と動かし方をそのまま表している

Metal - Oxide - Semiconductor Field-Effect Transistor

- **Metal（金属）**＝ゲート電極の層
- **Oxide（酸化物）**＝絶縁膜の層
- **Semiconductor（半導体）**＝シリコン基板
- **Field-Effect（電界効果）**＝電圧でスイッチを開け閉めする動かし方

> 略語に出会ったら、まず**元の言葉に戻して意味を調べる**。名前は中身を語っていることが多い

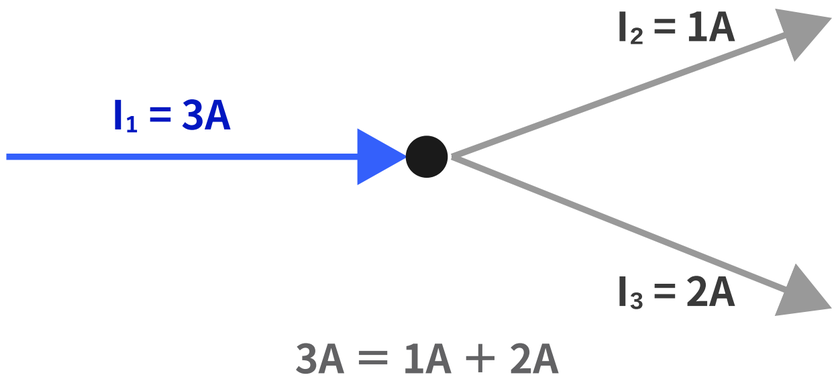


プレーナー型MOSFETの断面。3つの層が名前の前半に対応する

回路を厳密に計算する基本法則（存在だけ知ればよい）

第一法則（電流）

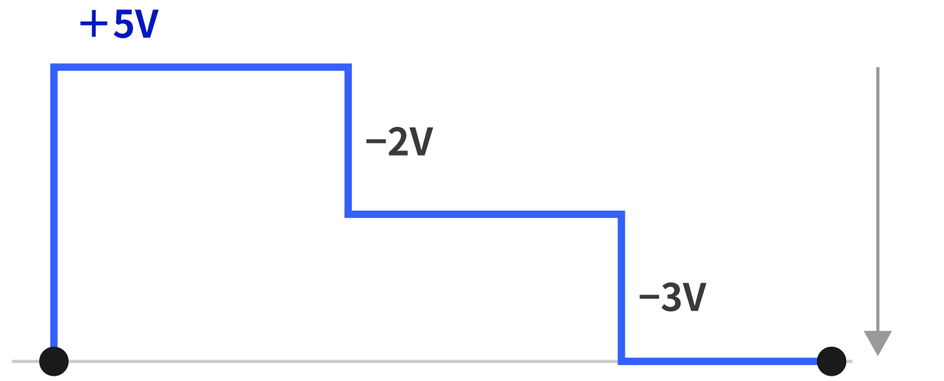
ある点に**流れ込む電流の合計**と、**流れ出る電流の合計**は等しい



$$3A = 1A + 2A$$

第二法則（電圧）

回路を一周すると、**電圧の上り下りの合計は 0**（山登りと同じ）

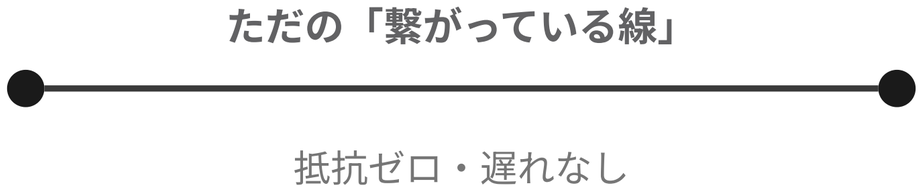


$$+5V - 2V - 3V = 0$$

模擬衛星の単純な回路では、ここまで持ち出す場面はほぼない。興味があれば：この法則はマクスウェル方程式から導ける

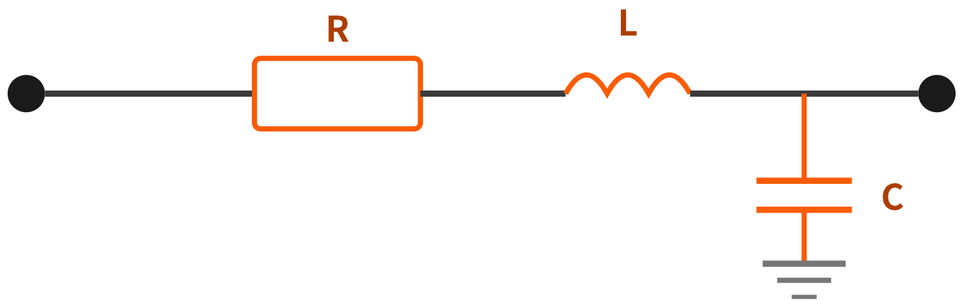
配線そのものにも、わずかな抵抗・容量が潜んでいる

理想（回路図の世界）



配線は単なる線で、電気は瞬時に、そのまま伝わる。

現実（寄生成分がある）



配線にもわずかな**抵抗**があり、配線間には**容量**が生じる。

ふだんは無視できるが、**I2C（Inter-Integrated Circuit）通信や高速な信号では効いてくる**。「回路図に載っていない要素が、現実の動作を乱すことがある」と知っておく

発展 2.7 発展3：他の部品は、こうやって自分で学べる

まず全体像をざっくりつかんで体系を持つ。それが一つひとつを深く理解する土台になる

- 部品は星の数ほどある。最初にやるのは丸暗記ではなく、**「どんな部品が、どのあたりにあるか」の地図**を持つこと
- 全体像をつかむのにも、個別を詳しく理解するのにも、**AIを適切に使ってよい**
- ただしAIの答えは仮説。最後は**一次情報＝公式データシート**で裏を取る（Week 0 の原則どおり）
- AI以外の入口：部品通販サイト（秋月電子など）の**カテゴリ目次**は世の中の部品を俯瞰する地図になる／解説サイトの例：detail-infomation.com

一つひとつを詳しく理解する

気になった部品を掘り下げる（データシート・解説記事・AIへの深掘り質問）

全体像・体系をつかむ（土台）

どんな部品が、どのあたりにあるかの地図（通販カテゴリ目次・分野の概観をAIに出させる）

AI 両段で使ってよい。ただし答えは**仮説**。裏取りは一次情報（公式データシート）で

まとめ 2.8 この章のまとめ：ここは覚えて次へ

部品の章の核は、この3つ

① 4部品の役割

抵抗
電流を絞る

ダイオード
一方向にだけ流す

コンデンサ
電源を安定させる

MOSFET
電気で動くスイッチ

② オームの法則 V = I × R

分圧もLEDの電流制限も、すべてこの式から計算できる

③ LEDはダイオードの一種

Vfを超えると電流が急に立ち上がる。だから直列に抵抗を入れて電流を制限する

SECTION 3 ／ 部品の繋げ方

部品が揃ったら、次は「組み立て方」

- ここまでで基本部品（抵抗、ダイオード、コンデンサ、MOSFET）が揃った
- 実際に回路を形にするには、部品を**物理的に繋ぐ手段**が要る
- まず**「最終形（基板）」**を見てから、**「試作の手段（ブレッドボード）」**へ進む

① **回路の表現**
配線図と回路図

② **電子部品**
抵抗・ダイオード・
コンデンサ・MOSFET

③ **部品の繋げ方**
ブレッドボードと配線

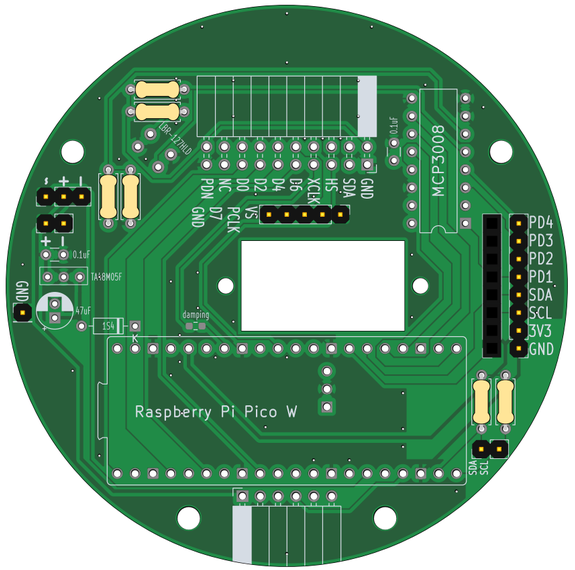
④ **デジタルとアナログ**
信号の2種類

⑤ **CMOSデジタル回路**
マイコンの中身と入出力

⑥ **マイコンの全体像**
ソフトとハードの接点

製品の回路は、配線が焼き付けられた基板の上に作る

- **PCB（Printed Circuit Board）**：銅の配線が板に印刷・固定された基板
- 部品をはんだ付けすれば、**頑丈で再現性のある**回路ができる
- 模擬衛星の本体も、複数のPCBを積み重ねた構造でできている
- しかし、PCBは**一度作ると配線を変えられない**



模擬衛星のメイン基板（MAIN）表面

いきなり本番基板を作ると、間違いの修正コストが大きい

- 設計どおりに動く保証は、作ってみるまでない（**初回から完璧はまず無い**）
- PCBは修正に時間とお金がかかるので、いきなり作るのはリスクが高い
- そこで**「何度でも配線を組み替えられる試作の場」**が欲しくなる → その答えがブレッドボード

本番基板（PCB）

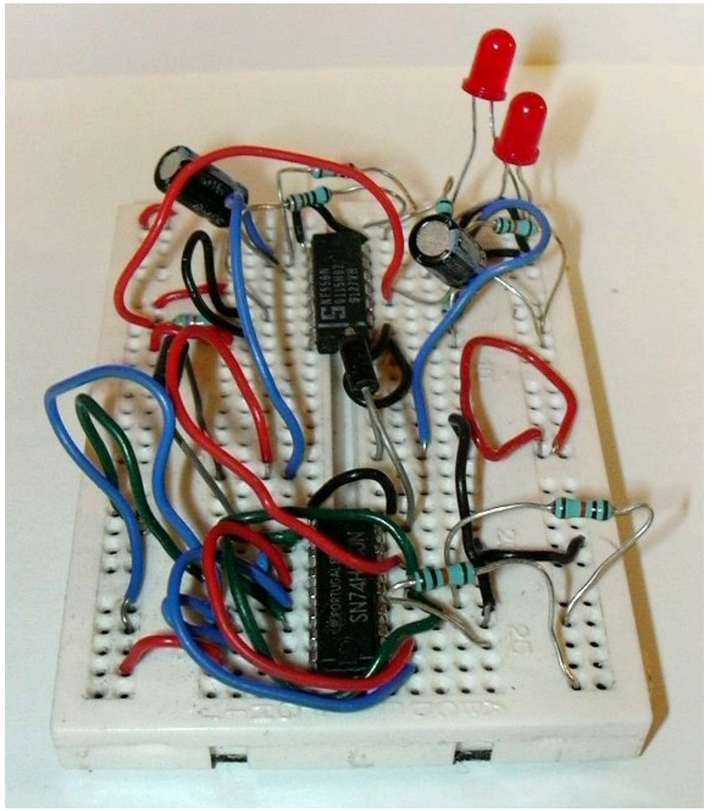
- ✕ 配線は焼き付け済みで変更できない
- ✕ 作り直しに時間とお金がかかる
- ○ 頑丈で再現性が高い（本番向き）

ブレッドボード

- ○ 何度でも挿し替えられる
- ○ はんだ付け不要。間違えても痛くない
- ✕ 接触が不安定（試作専用・3.7）

「板に部品を挿して配線する」発想から生まれた

- **語源：**その昔、パンを切る板（breadboard）に部品を留めて回路を組んだ
- **発想：**部品を挿せる板があれば、はんだ付けなしで試作できて便利
- **さらに発想：**よく使う配線が最初から板に入っていれば、もっと便利
- この**「あらかじめ入った配線」**が、現在のブレッドボードの内部構造の正体



ソルダーレスブレッドボード

余談 3.3.1 宇宙開発では試作品を「BBM」と呼ぶ

ブレッドボードは、システム工学の用語にもなっている

- システム工学では、開発初期の試作モデルを **BBM（Bread Board Model）**と呼ぶ
- 語源はまさにこのブレッドボード。形や仕上げにこだわらず、**機能が成立するか**を確かめる試作
- 模擬衛星づくりも、まず動く試作から始め、段階的に仕上げる点で同じ発想

BBM
Bread Board Model
粗い試作。機能が成立するかだけを確かめる

→

EM
Engineering Model
本番同等の設計で作り、性能を検証する

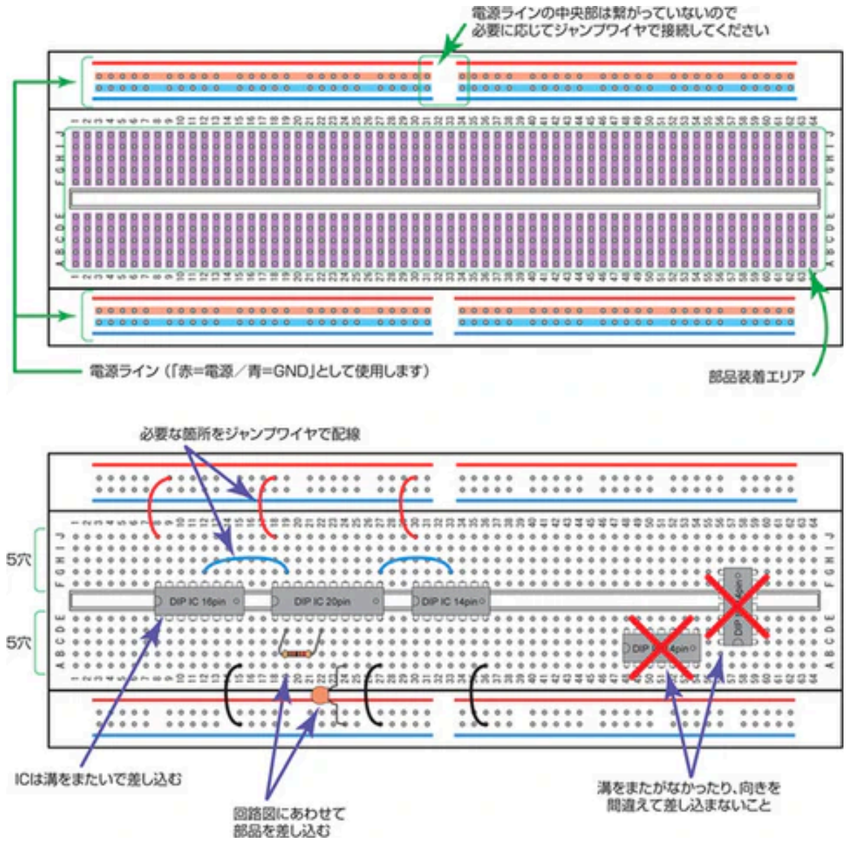
→

FM
Flight Model
実際に打ち上げる本番機

段階を踏んで、粗い試作から本番機へ近づけていく

「どの穴とどの穴が最初から繋がっているか」が全て

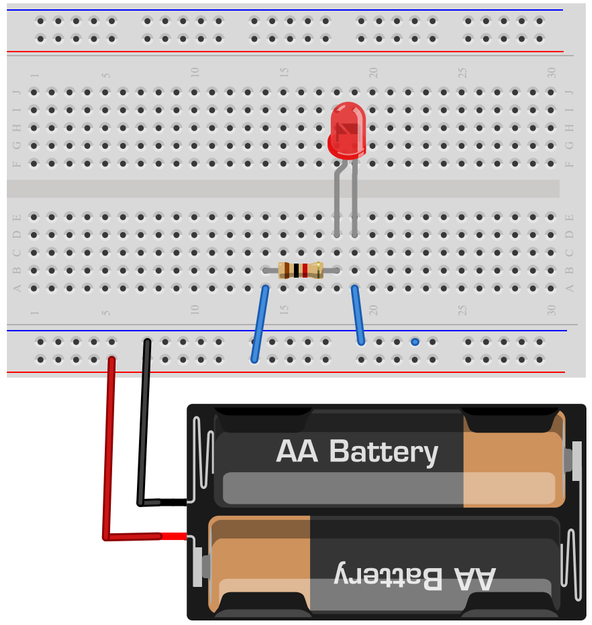
- **中央の穴：**縦方向（または横方向）の数穴が一列ごとに内部で繋がっている
- **端の長いライン：**電源（＋）とGND（−）を流すための共通レール
- 部品の足を**同じ列に挿せば、配線したことになる**
- 繋がっている列・いない列を取り違えると、**回路は意図どおり動かない**



内部の導通グループ。同じ色のまとまりが電気的に繋がっている

回路図の一周ループを、ブレッドボードの配線に翻訳する

- 電源レール（＋）→ 抵抗 → LED → GNDレール（−）の順に挿す
- LEDには**極性がある**ので、向きに注意（逆だと光らない）
- **同じ列に挿すことで「繋がった」**ことになるのを意識する
- これが、**回路図（論理）を実体配線（物理）へ戻す作業**そのもの



ブレッドボード上に組んだLED点灯回路（実体配線）

抵抗なしで電源とGNDが直結すると、過大電流が流れる

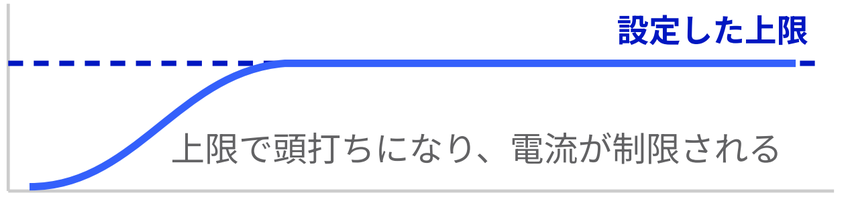
危険：ショート

- ショート＝抵抗なしで＋と−が繋がること
- 電池によっては瞬間的に**数百A**を流す能力があり、発熱・発火の危険

対策

- 配線前に繋がりを確認する
- **電源を入れる前に見直す**
- **安定化電源を使う**

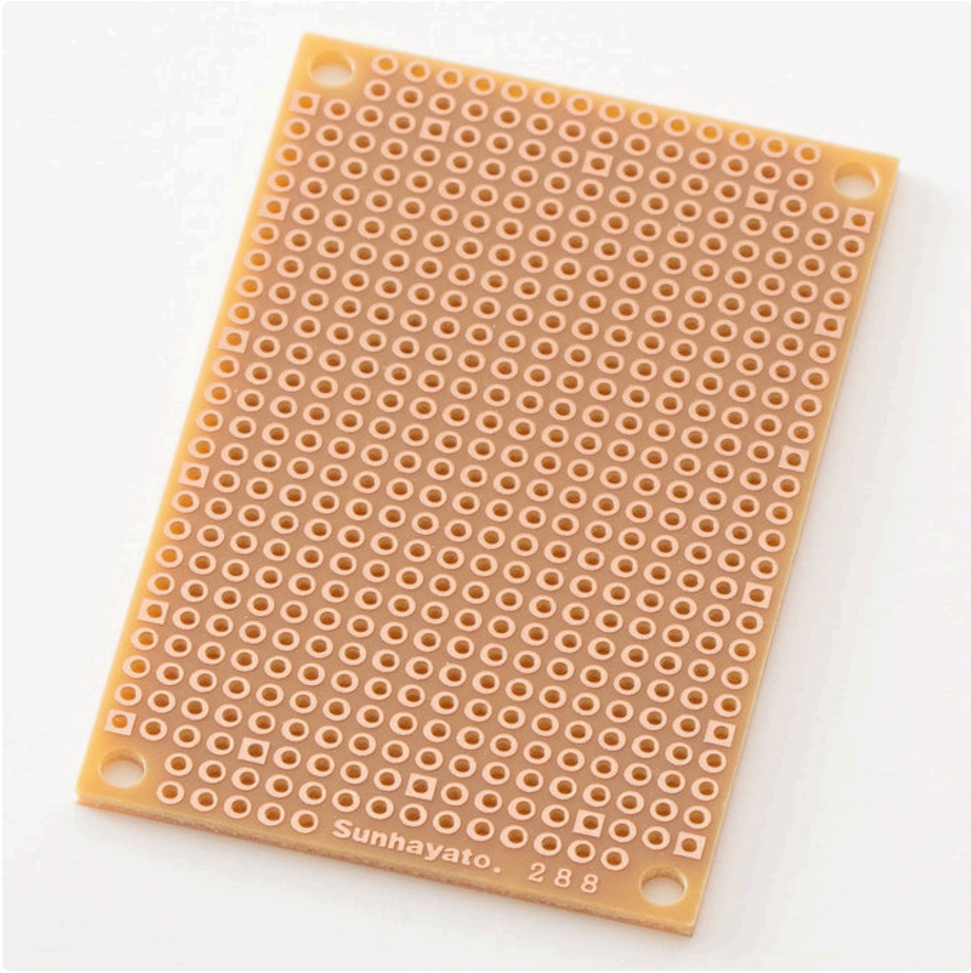
安定化電源の電流制限



電流の上限を設定でき、ショート時も電流を制限してくれる

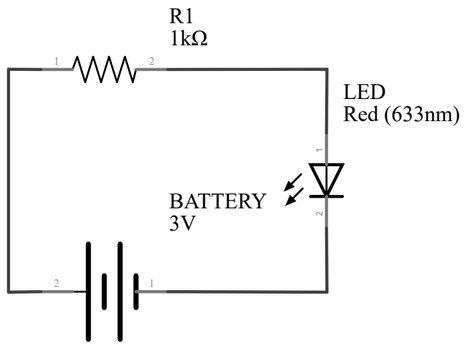
試作専用。動いても「次も動く」とは限らない

- 構造上、**接触が不安定**で、持ち運ぶと接触不良になりやすい
- 接触抵抗や配線間の容量が大きく、**高速信号や大電流には向かない**
- 長く挿しっぱなしだとバネが弱り、保持力が落ちる
- 本番で確実に動かすなら、最終的には
  PCBやユニバーサル基板へ起こす



ユニバーサル基板の例：サンハヤト ICB-288

この回路図のとおりに繋がっているのは、A〜Cのどれ？

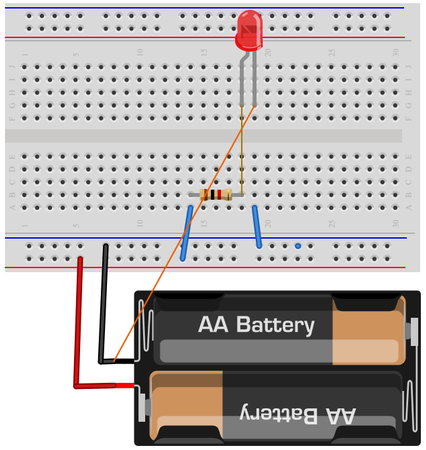


問題の回路図

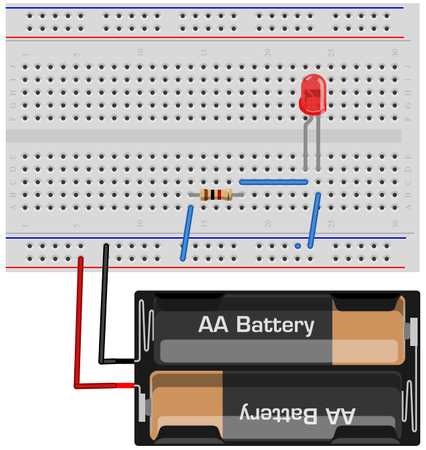
チェックの観点

- 中央の溝の左右は繋がっていない
- 同じ列は内部で繋がる
- 部品の両足を同じ列に挿すと短絡する

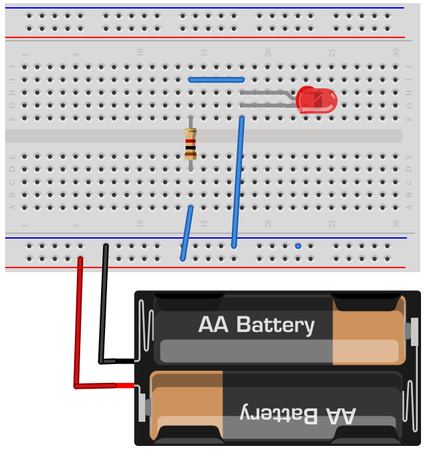
A



B



C



まとめ 3.9 この章のまとめ：ここは覚えて次へ

組み立ての章の核は、この3つ

①

内部の導通が全て

同じ列は繋がり、中央の溝で左右は切れている

②

＋と−の直結はショート

過大電流が流れて危険。電源を入れる前に
必ず配線を見直す

③

あくまで試作専用

動いても「次も動く」とは限らない。確実に
動かすなら基板へ起こす

SECTION 4 ／ デジタルとアナログ

信号には2種類ある：アナログとデジタル

- ここまでは**「電流をどう流すか」**という物理の話だった
- ここからは**「電気で情報をどう表すか」**という信号の話に移る
- マイコンやセンサが扱う信号を理解するための、**共通言語**を入れる

① **回路の表現**
配線図と回路図

② **電子部品**
抵抗・ダイオード・
コンデンサ・MOSFET

③ **部品の繋げ方**
ブレッドボードと配線

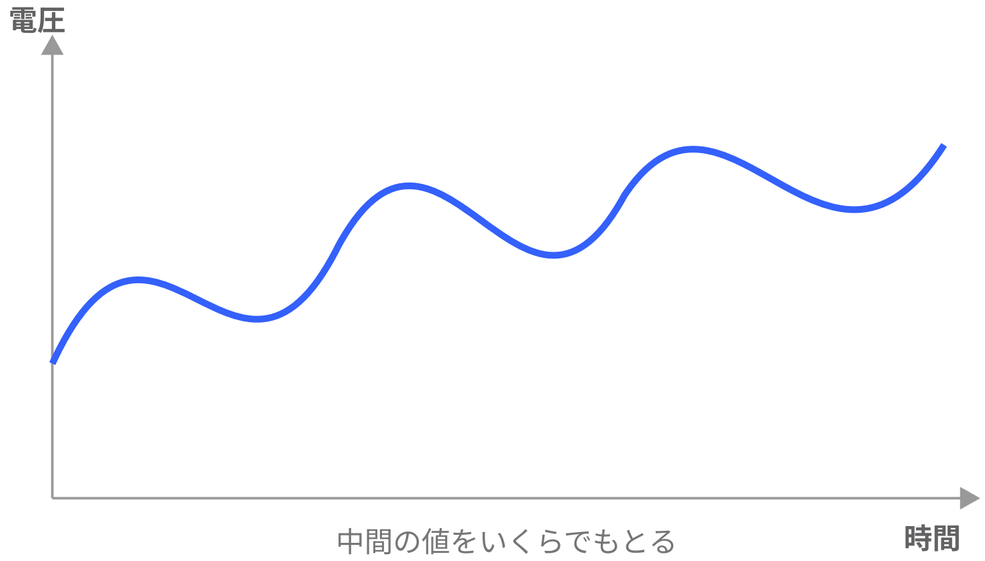
④ **デジタルとアナログ**
信号の2種類

⑤ **CMOSデジタル回路**
マイコンの中身と入出力

⑥ **マイコンの全体像**
ソフトとハードの接点

滑らかに連続して変化する量

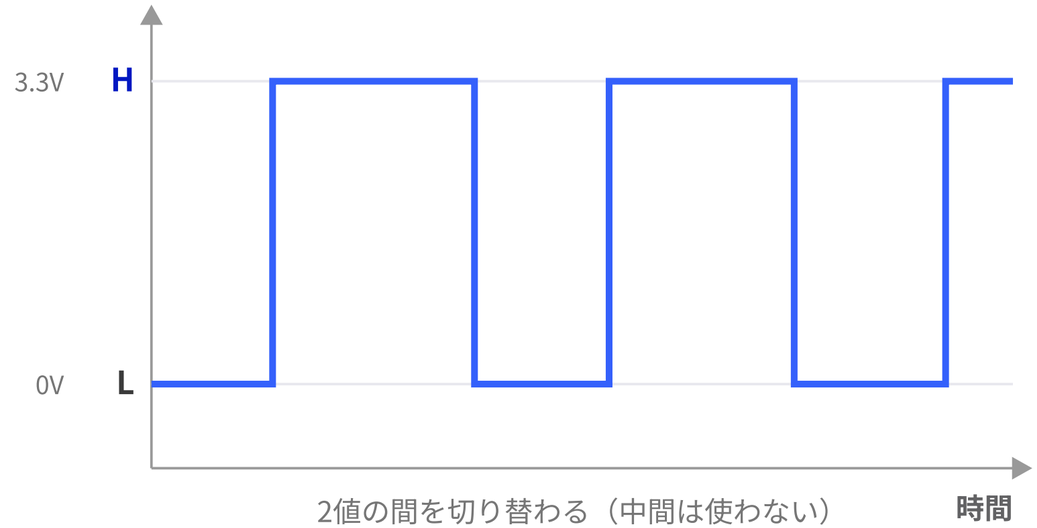
- 例：気温、音の大きさ、明るさ。値はなだらかに変化し、**中間値が無限にある**
- 模擬衛星のフォトダイオード（光センサ）が拾う「明るさ」もアナログ量
- 情報が豊かだが、**ノイズが乗るとどれが本来の値か分からなくなる**
- 例：本来 2.00V のはずが、ノイズで 2.02V や 1.98V に揺れてしまう



連続して滑らかに変化するアナログ波形

H と L の2値しかとらない

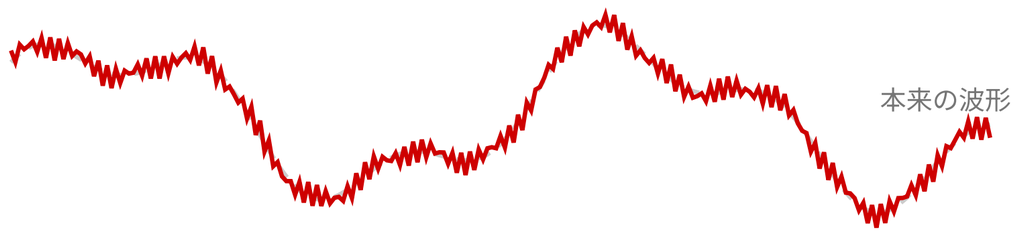
- 中間を捨て、電圧を**「高い＝H＝1」「低い＝L＝0」**の2段階だけで扱う
- 例：3.3V系なら、**3.3V付近を H、0V付近を L** とみなす
- 多少ノイズで電圧が揺れても、**「高いか低いか」の判定は揺るがない**
- マイコンの中は、すべてこの H と L の組み合わせでできている



H と L を行き来する矩形波

ノイズに強く、情報を正確に保てる

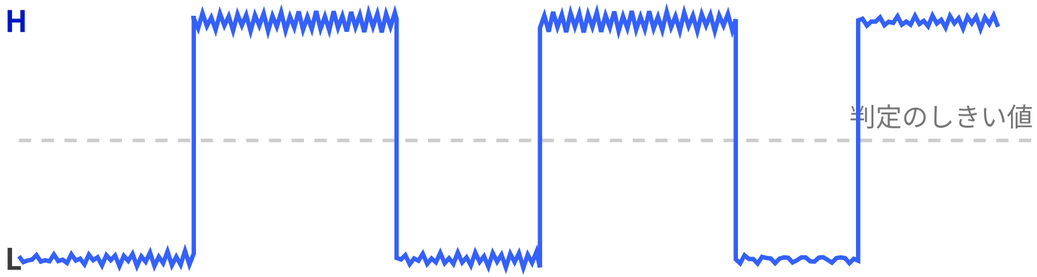
アナログ ＋ ノイズ



元の値がどれか分からなくなる

中間値がノイズで埋もれると復元できない。

デジタル ＋ ノイズ



しきい値の上か下かで、H/L を復元できる

HかLかだけなので、少々の揺れでは値が変わらない。

近年は省電力化のため、H の電圧が **5V → 3.3V → さらに低く**と下がってきている。だからこそ**「どの電圧をHとみなすか」は機器ごとに決まっている**（要データシート確認）

現実はアナログ、マイコンの中はデジタル。繋ぐのが ADC と DAC

- **ADC（アナログ→デジタル変換）**：連続的な電圧を、数値（H/Lの並び）に変換する
- **DAC（デジタル→アナログ変換）**：数値を、連続的な電圧に戻す（ADCの逆向き）
- 模擬衛星では、フォトダイオードの明るさを **ADC（MCP3008）** で読み取っている

まとめ 4.5 この章のまとめ：ここは覚えて次へ

信号の章の核は、この3つ

①

2種類の信号

アナログ（連続）とデジタル（H/Lの2値）。
デジタルはノイズに強い

②

橋渡しは ADC と DAC

現実の量はアナログ、マイコンの中身はデジタル。
ADCがアナログ→デジタル、DACがその逆

③

Hの電圧は機器ごと

模擬衛星は3.3V系。つなぐ前にデータシートで確認する

SECTION 5 ／ CMOSデジタル回路

H/L から論理を作る CMOS

- ここまでで**「MOSFET（スイッチ）」**と**「H/L（デジタル）」**が揃った
- この2つを掛け合わせると、**計算する回路＝デジタル回路**ができる
- この章は短い：CMOSの正体を一目見て、それが**マイコンの中身と入出力の正体**だと知れば十分

① **回路の表現**
配線図と回路図

② **電子部品**
抵抗・ダイオード・
コンデンサ・MOSFET

③ **部品の繋げ方**
ブレッドボードと配線

④ **デジタルとアナログ**
信号の2種類

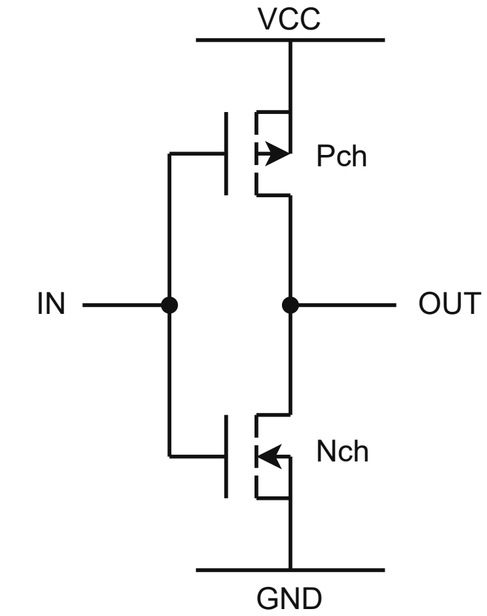
⑤ **CMOSデジタル回路**
マイコンの中身と入出力

⑥ **マイコンの全体像**
ソフトとハードの接点

逆の性格のMOSFETを組み合わせて、H/Lを反転・演算する

- **CMOS**：Nch と Pch を組み合わせた回路方式（今のデジタル半導体の標準）
- 例：入力 H を入れると出力 L、入力 L を入れると出力 H（**NOT＝反転**）
- これらを大量に組み合わせると、**足し算や記憶**など複雑な処理ができる
- ここでは**「H/Lを操作する素子が作れる」**ことだけ分かればよい

入力		出力	入力		出力
H	⟶	**L**	**L**	⟶	**H**



CMOSインバータ。入力Hで出力L、入力Lで出力Hに反転する

単純なスイッチを大量に集めると、やがて計算になる

1個

H が来たら L を返す

それだけの単純な動作

→

数十〜数百個

足し算ができる

論理を組み合わせた演算回路

→

数千万〜数億個

計算・記憶・判断

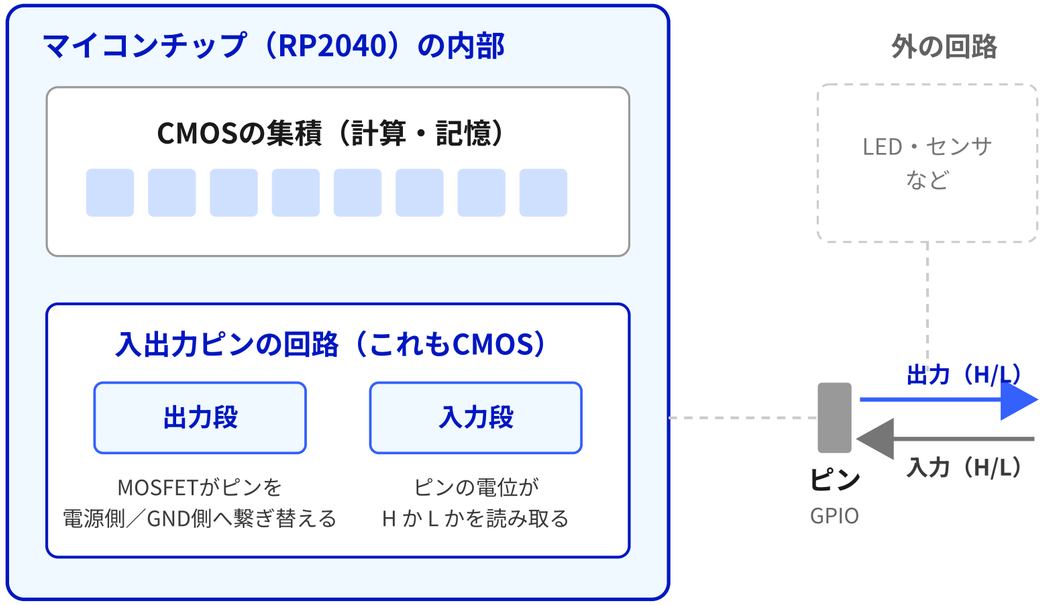
これがマイコンのチップの中身

- 1つ1つは「HならL」程度の単純な動作である
- だが膨大に組み合わせると、**四則演算も、記憶も、判断も**できるようになる
- **「魔法」ではなく、単純なスイッチの集積が計算を生んでいる**

チップの中身だけでなく、外の回路との接点も CMOS

- マイコンの中身は、無数のCMOSの集積だった
- 外の回路と繋がる**入出力ピン（GPIO）も、同じCMOS回路**でできている
- **出力：**中のMOSFETがピンを電源側（H）かGND側（L）に繋ぎ替えて、電圧を外へ出す
- **入力：**ピンの電位が H か L かを読み取る

入出力ピンを正しく使うための電気的な作法（Floating・プルアップなど）は、**Week 2 の通信**で根拠から扱う



チップ内部の計算回路も、外部との接点である入出力ピンも、同じCMOSでできている

まとめ 5.4 この章のまとめ：ここは覚えて次へ

CMOSの章の核は、この3つ

① **CMOS ＝ H/Lの論理を作る**

逆の性格をもつNchとPchのMOSFETをペアにした回路（インバータはHとLを反転する）

② **マイコン ＝ CMOSの集積**

数千万〜数億個が集まってできている。計算は魔法ではなくスイッチの集積

③ **入出力ピンもCMOS**

外の回路との接点である入出力ピンも、同じCMOS回路でできている

入出力ピンの電気的な作法（Floating・プルアップ／プルダウンなど）は、**Week 2 の通信**で根拠から扱う

SECTION 6 ／ マイコンの全体像

電気回路をソフトで操る存在

- 部品（抵抗、ダイオード、コンデンサ、MOSFET）と、信号（H/L）が揃った
- 残る問題は**「これらをどうやってソフトウェアから動かすのか」**
- その接点に立つのが**マイコン**。今日の最後のテーマ

① **回路の表現**
配線図と回路図

② **電子部品**
抵抗・ダイオード・
コンデンサ・MOSFET

③ **部品の繋げ方**
ブレッドボードと配線

④ **デジタルとアナログ**
信号の2種類

⑤ **CMOSデジタル回路**
マイコンの中身と入出力

⑥ **マイコンの全体像**
ソフトとハードの接点

「ソフトでハードを動かしたい」という要求から生まれた

- **我々がやりたいこと：**センサを読み、計算し、モーターやLEDを制御する
- それを毎回、専用の論理回路で作るのは大変で、変更もきかない
- そこで**「プログラムで振る舞いを変えられる小さなコンピュータ」**を回路に置く
- これがマイコン。**ソフトの命令を、ピンの電圧という物理に翻訳**してくれる

コンピュータの機能を1つのチップに収めたもの

- **マイコン（マイクロコントローラ）**：CPU、メモリ、入出力を1チップに統合
- 特に**入出力の機能（ペリフェラル）が豊富**なのが特徴
- 模擬衛星の頭脳 **Raspberry Pi Pico W** も、このマイコンの一種

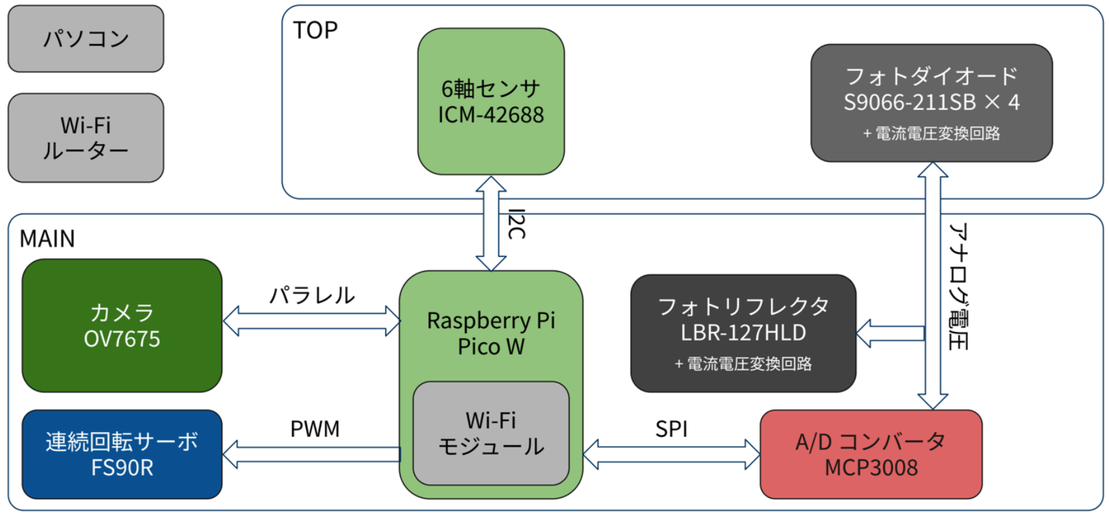
※コンピュータの内部の仕組みや動作原理は **Week 1 の講義**で詳しく扱う

パソコンでは別々の部品が、マイコンでは1チップに収まっている

Pico W は各サブシステムを束ねる司令塔（C&DH）

- **C&DH（データ処理系）**：Pico W が全体の司令塔
- **ADCS（姿勢制御系）**：光センサやジャイロを読み、モーターを制御する
- **Payload（ミッション系）**：カメラを制御し、画像を取得する
- **COMM（通信系）**：Wi-Fi で地上局とデータをやり取りする

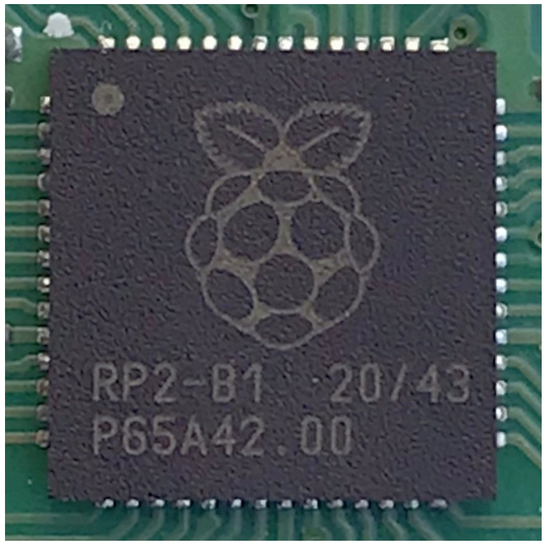
> これら全てを、**Pico W というマイコンがソフトウェアで束ねる**



模擬衛星システム構成図

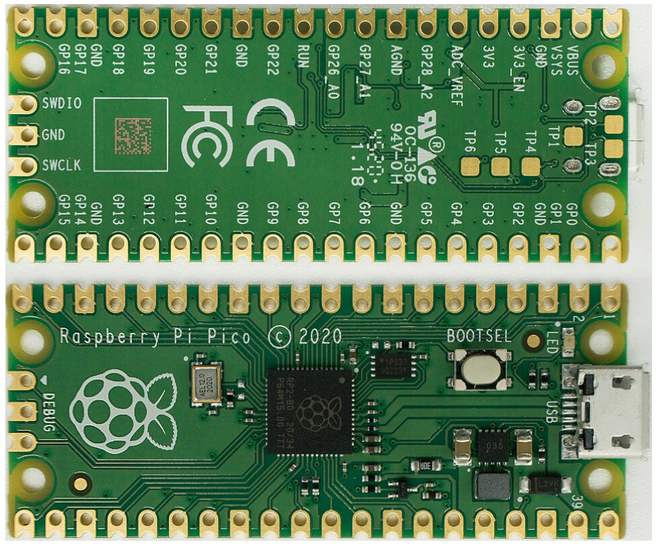
「チップ」はCPUそのもの、「ボード」はそれを使いやすくした基板

マイコンチップ単体 RP2040



演算の本体。これ単体では電源もUSBもなく、使いにくい。
例えるなら **CPU**。

開発用ボード Raspberry Pi Pico



チップに電源回路、USB、ピンを足したもの。
例えるなら **CPUを載せたマザーボード一式**。

試作はボードで行い、製品化するときはチップ単体を自分の基板に載せる

知りたい情報によって、開くべきデータシートが変わる

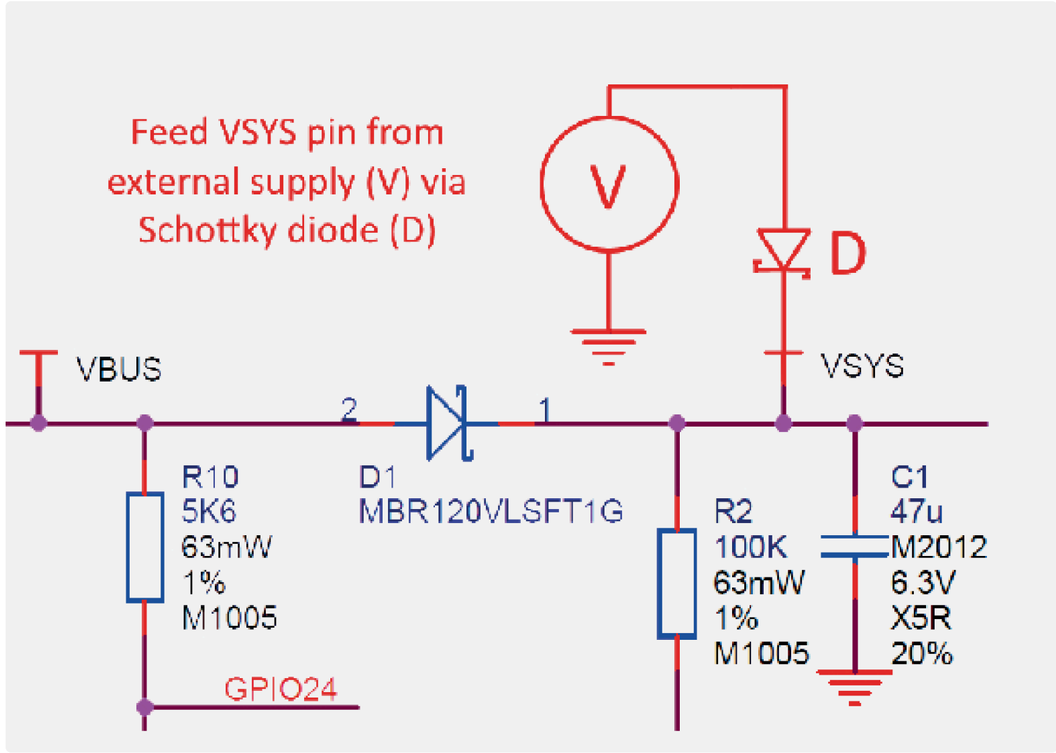
チップのデータシート

チップの内部仕様（Week 1 で読む）

ボードのデータシート

どのピンがどこに出ているか、電源の仕様など

- **スコープの正しいデータシートを開けること**が、これからの最初の関門
- 間違ったほうを開くと、いくら探しても情報は見つからない



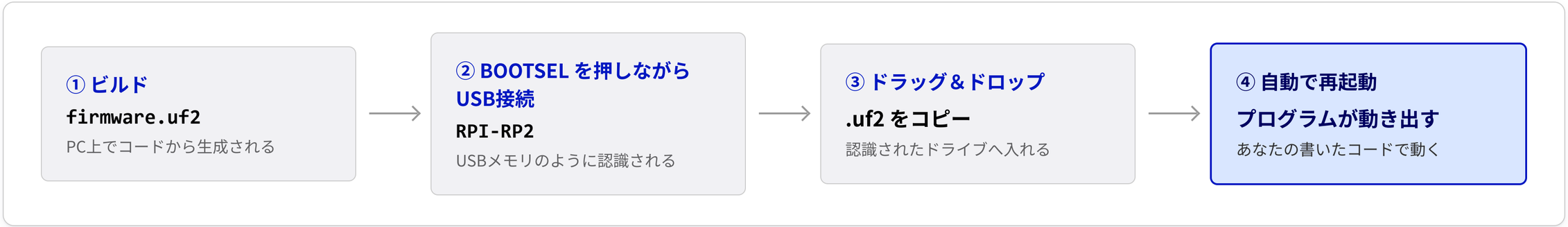
ボードのデータシートに載る情報の例（Pico の電源まわり）

Lチカの物理的な配線をもう一度確認する

- **用意するもの：**Pico Wボード、ブレッドボード、LED、抵抗、ジャンパワイヤ
- **配線の流れ：**Pico WのGPIOピン（例えばGPIO 0番）からジャンパワイヤで引き出す
- ブレッドボード上で抵抗を挟み、LEDの**アノード（長い足）**に繋ぐ
- LEDの**カソード（短い足）**から、Pico WのGNDピンへ戻す
- これで**「ソフトから電圧を変えられるピン」**を使った一周のループ（回路）が完成する

GPIO 0番ピンから抵抗とLEDにつないだ回路図

ビルドしたソフトをハードウェアに送り込む手順



- コードを書いてビルドすると、**.uf2** という拡張子のファイルができる
- Pico W の **BOOTSEL ボタンを押しながら**PCにUSB接続すると、USBメモリのように認識される
- この一連の作業が**「マイコンへの書き込み（フラッシュ）」**である

Week 0 で動かした Lチカのコードを、今の知識で見る

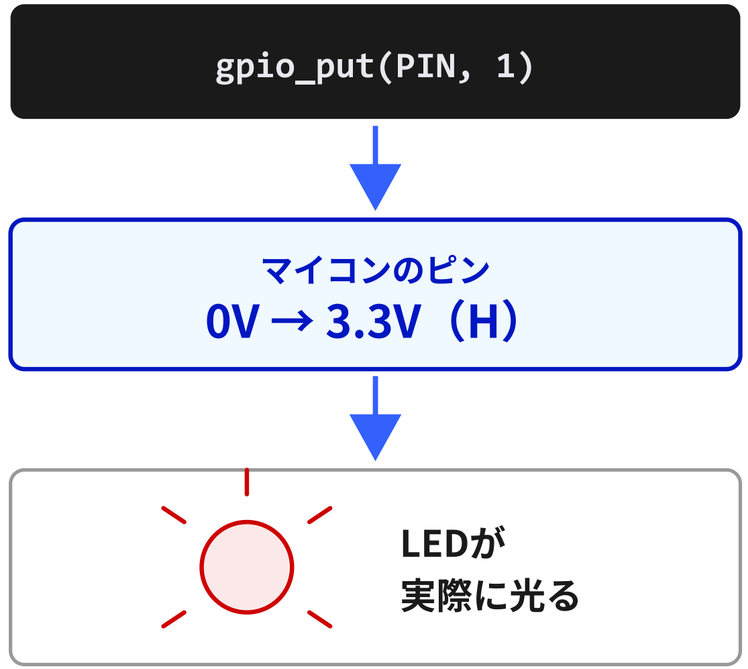
```
// Lチカのコード（抜粋）
gpio_init(LED_PIN);
gpio_set_dir(LED_PIN, GPIO_OUT);
while (true) {
    gpio_put(LED_PIN, 1);
    sleep_ms(250);
    gpio_put(LED_PIN, 0);
    sleep_ms(250);
}
```

- あのコードは「魔法」ではなく、マイコンに**物理的な電圧（H/L）を切り替えさせる命令**
- 各行が、**どのピンを、出力にして、HかLにするか**を1つずつ指示している
- この**「ソフトから物理ピンの電圧を操る」**ことが、すべての制御の基礎になる

ここではコードの意味だけざっくりつかむ。レジスタの仕組みは **Week 1**、C言語の構文は**次の補講**で扱う

それぞれの行が「マイコンへのどんな指示」になっているか

`gpio_init(LED_PIN);`	このピンを使います、という準備
`gpio_set_dir(LED_PIN, GPIO_OUT);`	このピンを「出力」にします、という設定
`while (true) { ... }`	この中をずっと繰り返します（無限ループ）
`gpio_put(LED_PIN, 1);`	電圧を**「高い（H＝1）」**に → LEDが点く
`sleep_ms(250);`	250ミリ秒だけ、そのままの状態で待ちます
`gpio_put(LED_PIN, 0);`	電圧を**「低い（L＝0）」**に → LEDが消える



C言語の1行が、物理的な電圧の変化とLEDの点灯に直結する

まとめ 6.10 この章のまとめ：ここは覚えて次へ

マイコンの章の核は、この4つ

① **マイコン＝通訳**

ソフトとハードの通訳。CPU・メモリ・入出力を1チップに収めたもの

② **コード＝電圧の指示**

Lチカのコードは魔法ではなく、ピンの電圧（H/L）を切り替える物理的な指示

③ **チップとボードは別物**

だからデータシートも2種類あり、正しいスコープのほうを開く

④ マイコンの**内部構造や動作原理は Week 1** で扱う

7. まとめ：今日の内容

部品と信号を揃え、マイコンという接点まで来た

回路図	物理の一周ループを、VCC/GND の記号で抽象化した地図
部品	抵抗（電流制限）、ダイオード（一方向）、コンデンサ（安定化）、MOSFET（スイッチ）
信号	アナログ（連続）とデジタル（H/L）。橋渡しは ADC と DAC
CMOS	単純なスイッチの集積が計算を生む。マイコンの中身も入出力もCMOS
マイコン	CPU・メモリ・入出力が1チップ。ソフトとハードの通訳

① **回路の表現**
配線図と回路図

② **電子部品**
抵抗・ダイオード・
コンデンサ・MOSFET

③ **部品の繋げ方**
ブレッドボードと配線

④ **デジタルとアナログ**
信号の2種類

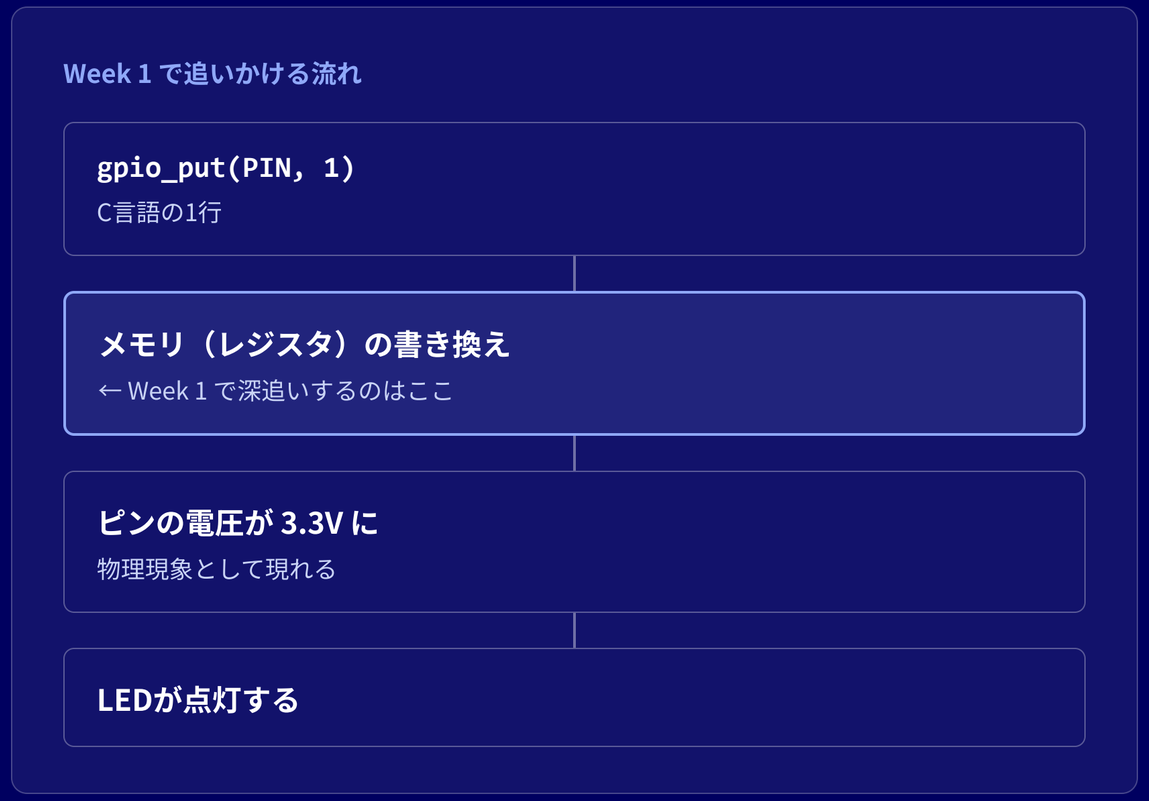
⑤ **CMOSデジタル回路**
マイコンの中身と入出力

⑥ **マイコンの全体像**
ソフトとハードの接点

NEXT ／ WEEK 1

Lチカを深追いし、ソフトとハードの境界を見にいく

- **次回やること：**Pico を題材に、Lチカのコードが最終的にメモリ（レジスタ）を叩く瞬間を追う
- 今日の**「コード＝ピンの電圧を切り替える指示」「データシートのスコープ」**が前提になる
- 今日の宿題は、**6章で見た blink のコードを実際に Pico で動かすこと**（次のスライド）
- 分からないことは**「キャッチアップの会」**で遠慮なく質問する



宿題 7.2 宿題：blink のコードを動かしてみる

6章で解説した blink を、実際に Pico で動かす

① 配線する（6.6）

GPIO → 抵抗 → LED → GND

LEDのアノード（長い足）とカソード（短い足）の向きに注意

→

② 書き込む（6.7）

.uf2 をドラッグ＆ドロップ

BOOTSEL を押しながらUSB接続して現れたドライブへ

→

③ 点滅を確認（6.8・6.9）

LEDが250msごとに点滅

コードの1行がピンの電圧を切り替えている

提出

点滅している様子を**写真か短い動画で Discord に投稿**する

動かないときの切り分け

- LEDの向き
- 抵抗が入っているか
- 列の導通（中央の溝）
- GNDに戻っているか
- 書き込みの成否

動かなくても投稿してよい。切り分けの過程が Week 1 以降の力になる

付録

画像クレジット（出典）

`resistor.jpg`	Haragayato（cropped by hidro）／Wikimedia Commons "Metal film resistor.jpg"／CC BY-SA 3.0
`cap_electrolytic.jpg`	oomlout／Wikimedia Commons "47uf Electrolytic Capacitor.jpg"／CC BY-SA 2.0
`breadboard.jpg`	LukeSurl／Wikimedia Commons "Breadboard.JPG"／CC BY-SA 3.0
`breadboard_internal.png`	（ネットからダウンロードする汎用イラスト。取得時に作者・出典・ライセンスを確認し、ここに明記する）
`pico_board.jpg`	Laserlicht／Wikimedia Commons "Raspberry pi pico.jpg"／CC BY-SA 4.0
`mosfet_2sk4017.jpg`	秋月電子通商 商品ページ（NchパワーMOSFET 2SK4017(Q)／通販コード107597）の商品画像／© 秋月電子通商（出典明記のうえ教材で使用）
`diode_1n4148.jpg`	Vonvon／Wikimedia Commons "Diode 1n4148.jpg"／CC BY-SA 3.0
`cap_ceramic.jpg`	Xofc／Wikimedia Commons "CeramicCapacitor-220nF.jpg"／CC BY-SA 4.0
`universal_board.jpg`	サンハヤト株式会社 ユニバーサル基板 ICB-288 商品ページの商品画像／© サンハヤト株式会社（出典明記のうえ教材で使用）
`rp2040_chip.jpg`	Thomas Glau／Wikimedia Commons "RP2040 Microcontroller (cropped)"／CC BY-SA 4.0
`led_osr5ja3z74a.jpg`	秋月電子通商 商品ページ（OSR5JA3Z74A／通販コード111577）の商品画像／© 秋月電子通商（出典明記のうえ教材で使用）