平成30年度「専修学校による地域産業中核的人材養成事業」

センシング装置を使った組込みソフトウェア教材

コネクティッド・ホームの普及に必要な中核的技術者養成事業

平成30年度「専修学校による地域産業中核的人材養成事業」

センシング装置を使った組込みソフトウェア教材

コネクティッド・ホームの普及に必要な中核的技術者養成事業

目 次

1. IoT/ コネクテッドホームとは …… 2
はじめに…… 2
1.1. コネクテッドホーム …… 3
1.2. IoT・ビッグデータ、AI からネットワーク …… 5
1.2.1. 産業界における IoT の活用例 …… 5
1.2.2. 農業における IoT の活用例 …… 6
1.2.3. 医療分野における IoT の活用例 …… 6
1.2.4. 身近な IoT の活用例 …… 7
1.3. IoT のサイクル …… 8
1.4. IoT/ コネクテッドホーム必須技術 …… 9
1.4.1. IoT/ コネクテッドホームに必要な要素と仕組み …… 9
1.4.2. ネットワーク技術の基礎知識 …… 11

2. mruby/c プログラミング …… 19
2.1. プログラミング言語「mruby/c」とは …… 19
2.1.1. mruby …… 19
2.1.2. mruby/c …… 20
2.1.3. mruby/c の簡単なプログラム …… 21
2.1.4. 基本文法 …… 22
2.2. 組み込みプログラミング 1 …… 25
2.2.1. プログラミングの開発環境構築 …… 25
2.3. 組み込みプログラミング 2 …… 32
2.3.1. デジタル出力を制御するプログラムをつくってみよう …… 32
2.3.2. PWM 出力するプログラムをつくってみよう …… 36
2.4. 組み込みプログラミング 3 …… 38
2.4.1. マルチプログラミング …… 38

3. SenStick mruby/c につて …… 44
はじめに…… 44
仕様一覧…… 44
3.1. 加速度センサについて …… 47
• 加速度センサの原理…… 47
• 加速度＋αによる行動認識…… 48
• 加速度センサを用いた乗り物分類…… 49

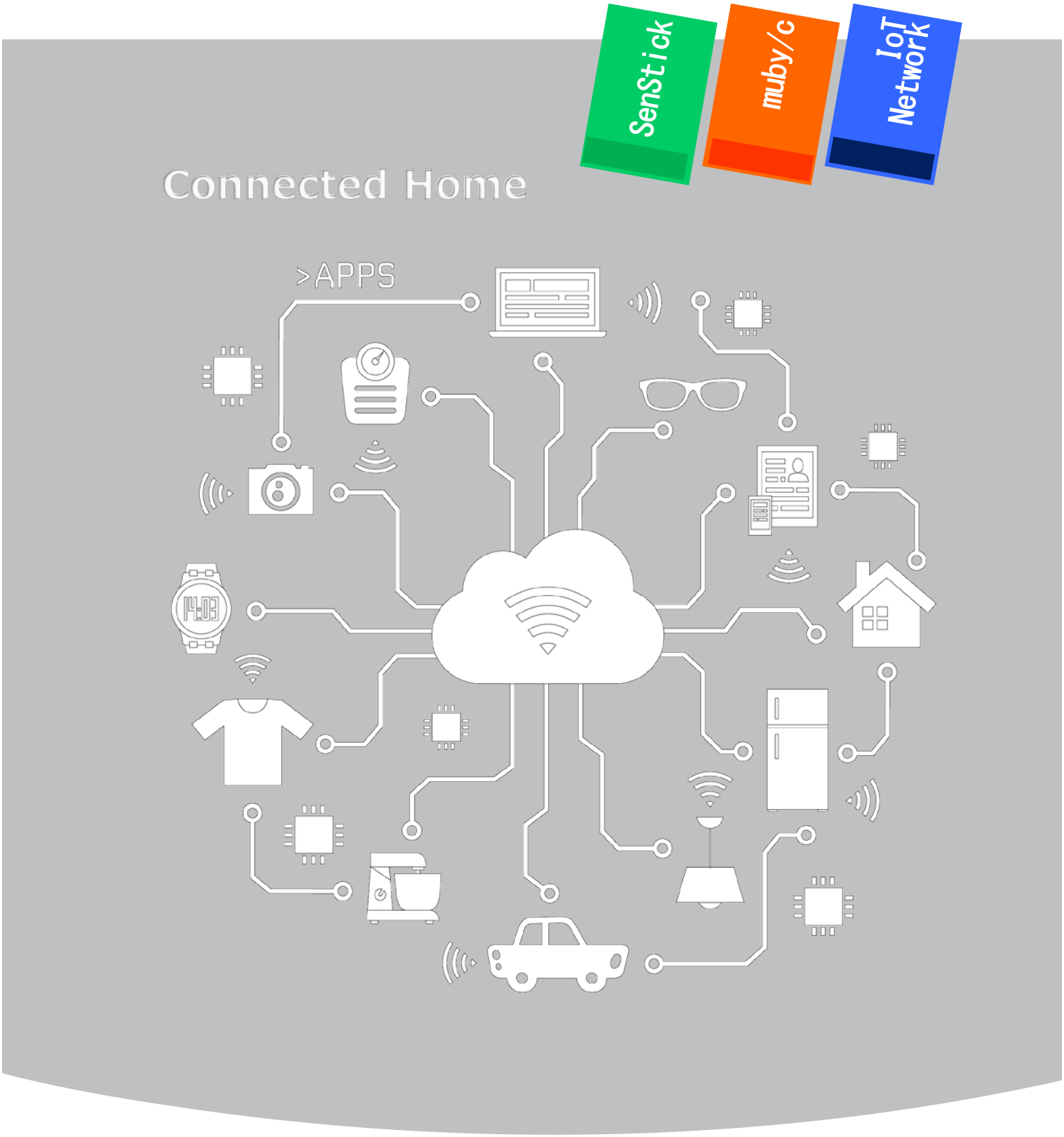
3.2. ジャイロセンサーについて …… 50
• ジャイロセンサーの原理…… 50
• 具体的な活用例…… 51
3.3. 地磁気センサについて …… 54
• 磁気センサのいろいろ…… 55
3.4. 温度センサにつて …… 55
• 温度センサの種類…… 56
• 温度センサの活用…… 57
3.5. 湿度センサについて …… 58
• センサの作動原理…… 58
3.6. 気圧 …… 59
【使用材料 - 方式から見た圧力センサ】…… 59
• ピエゾ抵抗方式の気圧センサ…… 59
3.7. 光センサについて …… 60
• 光センサの例…… 60
• 光センサの今後…… 61
3.8. UV センサについて …… 62

4. SenStick をつかった組込みプログラミング …… 63
4.1. SenStick プログラミングの流れ …… 63
4.1.1. 簡単な SenStick プログラミング …… 63

5. 演習 …… 67
5.1. 基本プログラム …… 67
5.1.1. 基本プログラムの作成と実行 …… 67
5.2. 応用プログラム …… 68
5.2.1. 応用プログラムの作成と実行 …… 69

SenStick 参考資料 …… 71

プログラミング演習参考資料……109



SenStick mruby/c で センシング技術を学ぶ

1. IoT/コネクテッドホームとは

はじめに

コンピュータの進化により、あらゆるものにコンピュータシステムが組み込まれています。 インターネットの普及により、それらが結び付き、IoT（Internet of Things）：モノのインターネットと呼ばれるようになりました。
ありとあらゆるモノがインターネットにつながることで以下のことが可能になりました。

・離れたモノを操作する遠隔制御
・離れたモノの状態を知る遠隔監視
・離れたモノ同士で任意のデータの送受信

具体的な例でいうと、
「ドアの鍵ちゃんとかけましたか？」
だれでも不安になることがありますよね？
そんな時、スマホが「ドアが開いているよ！」と教えてくれたら？
慌てて家に戻る必要はありません。遠隔操作で、ドアの施錠も可能になります。
これだけでも便利ですよね。
ところが、近年の AI の急激な進化により、IoT がさらに新たな価値を創造します。

家の中に設置したセンサから、あらゆるデータを収集し、留守であること、ペットが餌を食べていること、そして誰かが侵入したこと！など AI が正常/異常を判断し、通報が必要と判断したときには自動で映像もつけて通報してくれます。もう警備会社の機能をコネクテッドホームが自ら担ってくれるのです。

IoT+ビッグデータ＋AI が私たちの社会や生活に大きな変化をもたらすでしょう。
日本は高齢化社会を迎え、一人暮らしの高齢者は全世帯の 20%近いと言われています。
東京では 50%以上の世帯が共働きだそうです。
田舎にいる両親やお留守番の子供たちの見守り、そしてなにより自分たちの健康のチェックなど、コネクテッドホームが私たちに与えてくれるものは今までにない価値です。
これからの時代を担うみなさんがそのことを理解し学習を進めてくださることを願っています。

1.1. コネクテッドホーム

従来のホームオートメーションに「モノのインターネット(IoT)」を取り入れ、家電など様々な機器がインターネットと無線通信で接続され、それらを統合するデバイスに家庭内で命令を出すと自動制御が行われたり、スマートフォンやタブレットなどの端末によって外出先からでも遠隔操作が行えたりする家、あるいはそのシステムを表しています。

遠隔操作を行える機器は、テレビ、DVD レコーダー、ゲーム機など、ネット接続型のものはもとより、家庭用電話やプリンタ、照明、冷暖房・空調機、洗濯機、お掃除機ロボット、冷蔵庫などすでに多岐にわたり、今後も増加していく見込みです。また、音声認識を利用して様々な機器の制御を統合して家庭内で行うシステムとして、スマートスピーカーも普及が広がっています。はじめに少しご紹介しましたが、コネクテッドホームが実現する近未来はどういったものなんでしょう。
想像してみてください！

目覚ましの時間を頼んでおくと、1 時間の前から、ゆっくりと室温調整が行われ、静かにカーテンが自動であけられ、優しい音楽とアロマの香り。自然な目覚めを即してくれます。
「おはよう」と優しい声で、今日の予定や友人の誕生日プレゼンとをどうするか？など秘書のようにあなたに話かけてくれます。

就寝時間や、ヘルスケアのデータから主治医のように健康上の注意と今日の食事とトレーニングのメニューを教えてくれるかもしれません。
寝室を出るとお掃除ロボットが掃除しながら、ペットとじゃれています。
冷蔵庫に足らない食材や身の回りの品は自動で注文され、もう宅配ボックスに入っています。
家族がリビングに移動すると、今日のニュースや家族で共有する情報を教えてくれます。
今日の予定、気温や天候から、どんな服装で出かければいいかも教えてくれます。
なんだかまるで夢のような話ですね。

でも！実はここに書かれている内容は、一つ一つをとればすでに実現されている技術ばかりです。コネクテッドホームは文字通りつながることで、その一つ一つを連携し共通のプラットフォームを形成することで家の概念や生活環境を大きく変えるでしょう。

ディスカッションタイム

あなたが考えるコネクテッドホームとは？

- 今すぐにできること

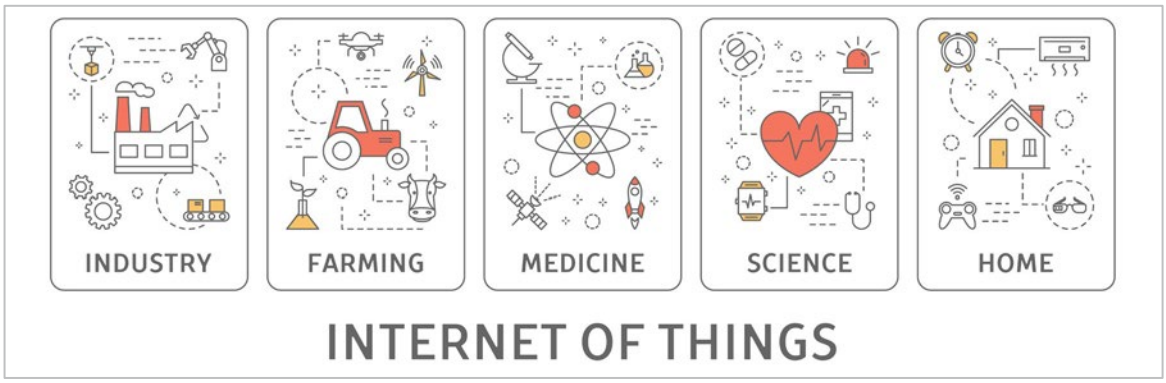
- ちょっと頑張ればできること

- 未来に実現されるだろうこと（タケコプター級）

- 未来にできたらいいなと思うこと（どこでもドア級）

1.2. IoT・ビッグデータ、AI からネットワーク

IoT がまずはどんなものか、活用例を使って紹介します。



1.2.1.産業界における IoT の活用例

● モノづくりの製造ラインの監視

多くの製造工場ではすでに産業ロボットによって製造ラインがこうちくされています。
その製造ラインを監視するのに、熱を測るセンサを使って異常に高温になっていないか？
カメラ画像や振動を測るセンサを使って故障の予兆を検知するなど、製造ラインの監視に利用されています。

● 物流（モノがどこにあるかの）サービス

都会では EC サイトで買い物をすると、一時間以内に配達してくれるというサービスがあります。出荷準備がすんだか？　サーバーにある購入データと倉庫でのピッキングデータ、出荷用のかごに入ったかをセンサで検知。これらのデータと輸送の GPS データをサーバーに集約することにより、拠点から配達場所まで MAP 情報で、いまどのあたりにいて、あと何分で到着するかをユーザーがスマホで確認することができるサービスです。

● スマートデバイスと連動する「コネクテッドカー」

「2025 年には世界で 6547 万台、日本でも 954 万台がコネクテッドカーになる」と予測しています。整備の時期を自動で通知したり、自動車の通信機能によって、自動車自身が事故を避けるようになり、自動運転が実現されるのも遠い未来ではないかもしれません。

1.2.2. 農業における IoT の活用例

● ビニールハウスの制御

温度センサにより、検知した温度があらかじめ設定している閾値を超えると、ファンがまわり窓を開ける。閾値より下がるとファンを止め、窓を閉める。こうして最適な温度に保ちます。IoT＋ビッグデータ＋AI により、安定した作物の成長を実現します。

● 水田の水位を一定に制御

稲作では水の管理に多くの時間を費やす多面、この作業を大幅に簡易化するための水位調整を遠隔化するサービスが実用化。水田に設置され水位センサの値により、水路につながる水門の開閉を、スマホやパソコンから操作でき、水位や水温までも調整できる。またこうして毎日の気温や水質などのデータの蓄積から、今後の育成や栽培に役立つデータをあつめることにもつながっている。

1.2.3. 医療分野における IoT の活用例

● 通信機能搭載型人工臓器

通信機能が埋め込まれた人工臓器、ウェアラブル型機器をつかった IoT 化によるリアルタイムモニタリングが遠隔からでも可能になり、機器の不具合や病態の急変などに即座に対応することが可能となり、需要がのびている。主に不整脈治療で必要不可欠な医療機器であるペースメーカー／植込み型除細動装置の占めるウェイトが高い。

● 遠隔医療支援 IoT システム / サービス

遠隔医療支援、服薬管理支援、遠隔看視（在宅患者見守りシステム）等のシステムおよびサービスが、今後の高齢者社会では需要増加すると見込まれる。
また使用される医療機器のモニタリング、遠隔監視など、在宅医療関係者や介護関係者の人手不足解消や、患者家族の体力的・精神的負担軽減に対応できるため、在宅医療でありながら、病院内と同じような見守りを実現可能とする。

1.2.4. 身近な IoT の活用例

● ヘルスケアデータを価値のある情報サービスへ

個人用のヘルスケア機器やスマホから得られるデータを、AI エンジンによって使用者やサービス事業者に価値ある情報へと変換してフィードバックされるサービスが急激に増加。
運動量計の履歴データより、このまま運動をつづけた場合に、血圧などの各種バイタルサインがどれくらい改善されるのかを使用者に伝えてモチベーションを継続。装着型のヘルスケア機器の情報から、熱中症などの危険を警告する！ベッドや装着型のヘルスケア機器の情報から、睡眠の質を判定し、生活習慣についてのアドバイスを行う！などの業務支援を行うことが可能です。また健康飲料メーカーとヘルスケアの機器メーカーなど異業種のコラボも生まれています。

● チケットレスのマジックバンド

世界的なテーマパークが、来場者にリストバンドを配布し、入場券・ホテルの鍵・食事・買い物といった全ての体験をリストバンド一つで行えるようにしました。支払いなどに煩わされることなく、テーマパークの世界に集中できるのです。テーマパーク側としては、リアルタイムで来場者の状況を確認し、混雑状況に応じたスタッフ配置や在庫管理に役立てています。

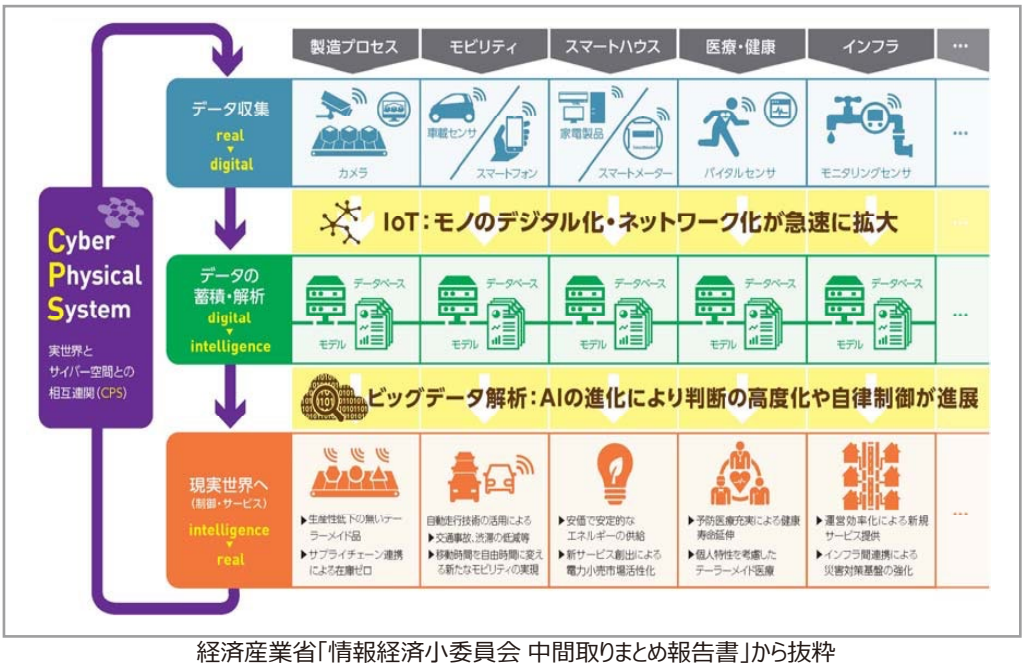
● バスのリアルタイム到着案内

インバウンドによる観光客向けサービスの拡充を目指し、観光地ではバスの停留所に大型液晶ディスプレイを設置し、バスの運行状況をリアルタイムで通知するサービスを開始されました。バス全てに通信機器を搭載し、無料の公衆 Wi-Fi などと連動して、リアルタイムの到着案内を行います。この手法は従来技術よりも少ないコストで導入できるため、他の自治体への拡大も期待されます。

● スマートスピーカーを使ってコネクテッドホームを実現

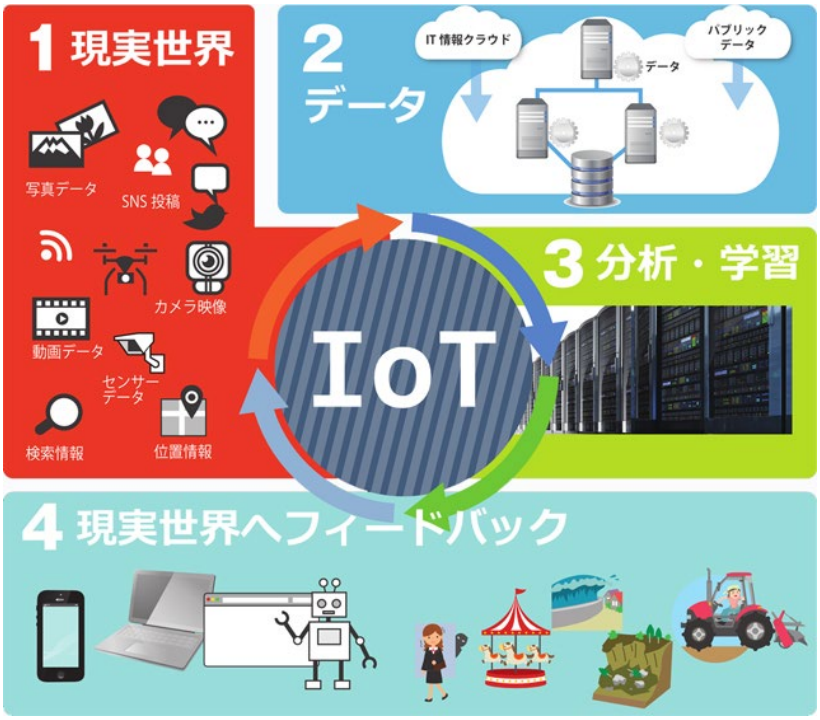
スマートスピーカーが IoT のひとつのプラットフォームになろうとしています。
スマートスピーカーに対応した、「ライト、コンセント、エアコン、スマートロック、お掃除ロボット、空気品質モニター、時計、ネットワークモニター、スマートボタン、防犯カメラ」など連動できる機器が続々と市場にでてきました。これらに対応した製品を購入することで、安価でコネクテッドホームの機能を実現できます。音声で家電を操作できるだけじゃなく、外出先からもスマホをつかって遠隔操作ができたり家の状態を確認ができたりします。

1.3. IoT のサイクル



経済産業省「情報経済小委員会 中間取りまとめ報告書」から抜粋

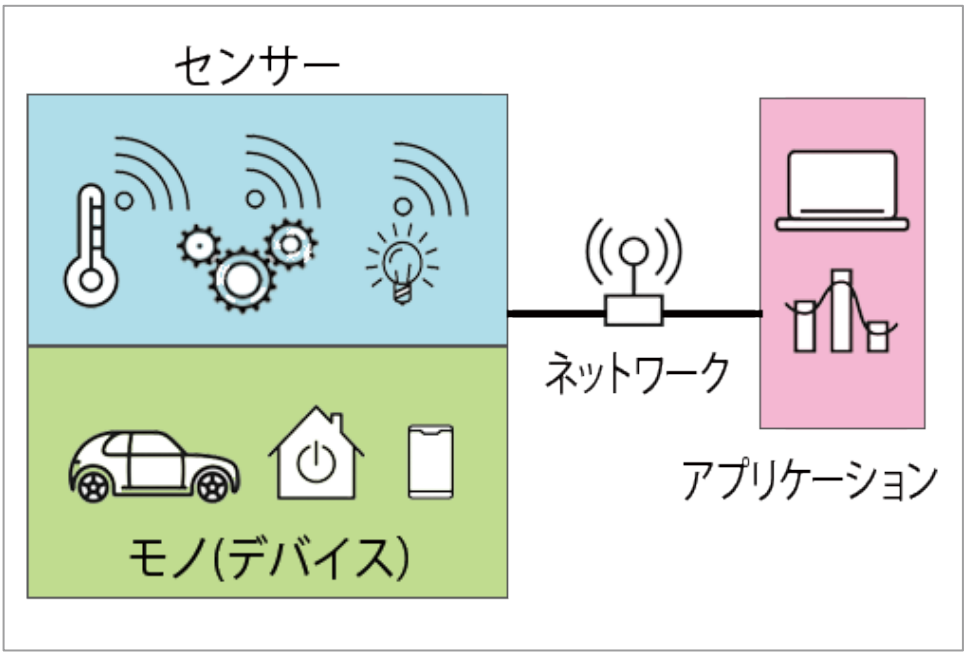
- 現実世界から収集したデータを（1）、クラウドやデータセンターに集約し（2）、集約したデータを分析・統合して（3）、分析結果を現実世界のシステムやサービスにフィードバックする（4）IoT サイクルの概念図



出展@IT(https://www.atmarkit.co.jp/ait/subtop/features/systemdesign/iot.html)

1.4. IoT/コネクテッドホーム必須技術

コネクテッドホームを実現するために必要な技術要素について学習します。



1.4.1.IoT/コネクテッドホームに必要な要素と仕組み

● モノ（デバイス）

ここで言う「モノ」は、あらゆる物体が対象となりえます。一般生活の中であれば、車や家電やスマホから、時計、等。物理的に**制御システム**や**センサ**を取り付け**ネットワークと接続**できるモノ全てが対象です。

このようにモノのインターネットにつながるデバイスを最近では、IoT 家電、スマート家電などと呼んでいます。（この場合、スマート＝賢いという意味）

● センサ

モノまた周辺環境の状態を感知し、データとして読み取ることのできる装置が、この「センサ」です。モノの存在有無や形状、位置などを感知することもできれば、重さや圧力、速度、音声、振動、温度、湿度、におい、電磁気、光など、実に様々な種類の状態を感知するセンサが存在します。

第 3 章では、SenStick をつかって具体的にどのようなデータを取得できるかを学習します。

● ネットワーク

モノの状態をセンサが感知してデータとして読み取ったら、今度はそのデータを可視化する端末（例えばパソコン、クラウド上のデータベースなど）に送る必要があります。この“送る”役割を果たすのが「ネットワーク」です。
現在では無線ネットワークを利用するのが一般的になっていますが、無線ネットワークの中でも種類がいくつかあります。データの重さ、データを送る速さ、データを送る距離、バッテリーの消費量など、実行したい内容や用途に合わせて最適なネットワークを選びます。
ネットワークについては次項にて詳しく説明します。

● アプリケーション

データの可視化や判断するための情報処理をするのが「アプリケーション」です。
収集されたデータは、まだ暗号レベルだったり量が膨大だったりするため、データの抽出・整理・解析・最適化といった処理を加え、人間が理解しやすいグラフや図にまとめるのがアプリケーションの役割です。またデータを判定し、モノに遠隔操作で状態を変更したりることもできます。
これらの情報処理を行うためには、クラウド、データベース、データサイエンスや AI、機械学習といった知識も必要になります。
ハウスメーカーが提供する HMS（見える化システム）やクラウドと家電を結びつけるスマートスピーカーなどは、これらアプリケーションが動作するプラットフォームとして期待されています。
アプリケーションの開発については、第 2 章、第 4 章で学習します。

モノの電化と電力確保の必要性

IoT を実現するには、モノの電化と災害時にも安定供給される発電・蓄電の仕組みが必要になります。近年、ハウスメーカーや自動車メーカーが住宅内エネルギ消費の最適化と快適性の両立を図る「スマートハウス」に力を入れています。これも IoT のひとつといえるでしょう。

スマートハウスの中核技術は、「HEMS（Home Energy Management System）」と呼ばれており、太陽光発電などのエネルギ機器やエアコンなどの家電、給湯器などの設備機器をネットワークで相互接続することで、住宅内のエネルギ利用状況をモニタリング（見える化）し、消費電力を抑えつつ快適に生活するためのエネルギ制御を行うシステムです。

出展 https://www.renesas.com/jp/ja/about/edge-magazine/solution/11-hems.html

1.4.2. ネットワーク技術の基礎知識

IoT に欠かせない、ネットワークについて学習しましょう。

1.4.2.1. ネットワーク技術用語解説

公衆交換電話網（PSTN）

【 Public Switched Telephone Networks 】 一般から広く加入・接続を受け付ける公衆網の一種で、街中の地中や空中に固定的に敷設されたメタル回線（銅回線）で電話局と加入者宅を結び、加入者間でアナログ伝送の音声通話を利用することができる。コンピュータで使用する場合はデジタル通信にモデムで変換。ダイアルアップ回線、ADSL 回線までインターネット接続で利用されてきました。

ただし、近年 IP 電話が従来型固定電話の契約数を逆転しました。今後も PSTN の利用増加は見込めないことから、NTT は、通信インフラをすべて IP（インターネットプロトコル）網に移行することを決定しました。それに伴い、既存の PSTN である「INS ネット 64」「INS ネット 1500」「デジタル通信モード」は、2020 年から随時廃止されることが決定しています。

IP ネットワークまたは IP 網

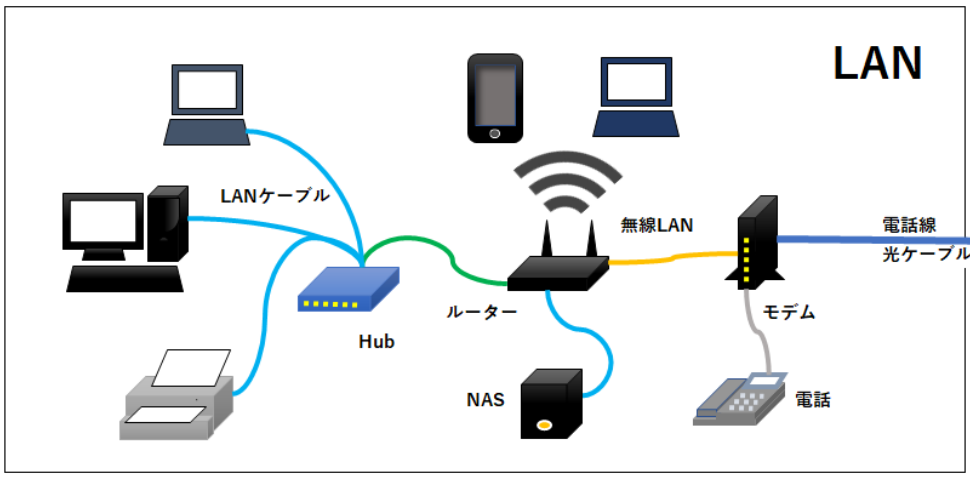
インターネット・プロトコル・スイート技術(インターネットの技術)を利用して相互接続されたコンピュータネットワークです。2024 年 1 月から切り替え開始予定。

LAN（Local Area Network）

基本的には狭いエリア＝原則として同一建物内のネットワークを意味します。

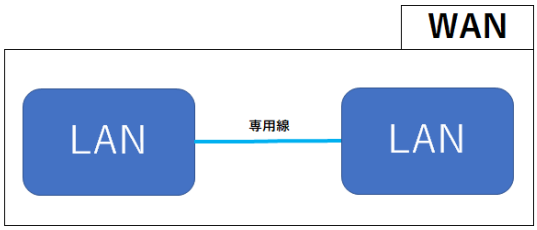
内部アドレス（プライベート IP アドレス）で情報をやり取りし、ハブやプリンタやファイルサーバの共有などを実現します。

外部回線につなぐ場合にはルーターなどのゲートウェイを経由して接続します。



WAN（Wide Area Network）

LAN では接続しきれない遠隔地の LAN 同士を結ぶネットワークです。
専用線で接続した閉じられたネットワークです。
専用線を確保できない場合、インターネット回線を疑似専用線（VPN）として使用します。



VPN

インターネット回線を疑似的に専用線として利用する技術。
契約者以外のユーザーからの接続は不可能。専用回線を用意するよりコストがかからない。

IP アドレス

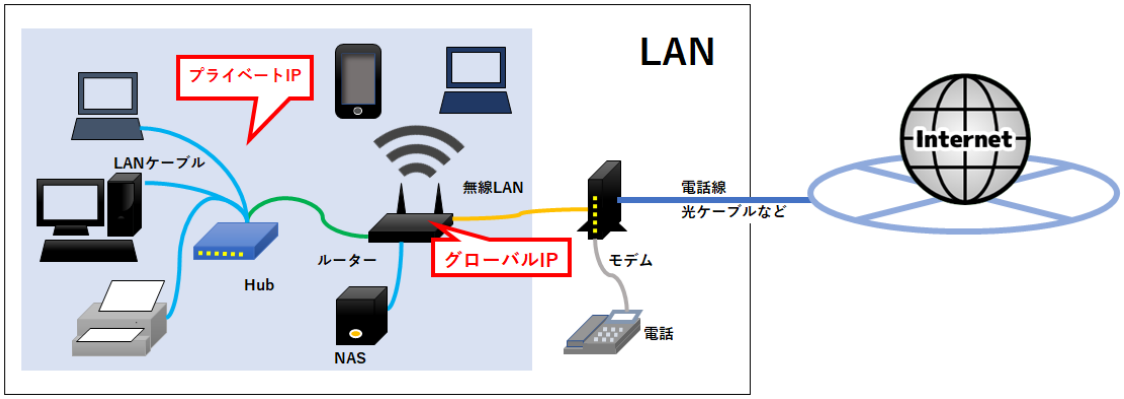
通信機器（PC など）の所在を表すもので所在がわからないと通信ができないため必ず必要です。現在は IPv4 が多く利用されている IP アドレス方式ですが、急激なインターネットの普及のため、IPv4 のアドレスが枯渇する問題が発生したため、IPv6 が新たに設定されました。

IPv4：
8 ビット（8 桁の 2 進数）×4 区切り=32 ビット
例：192.168.1.111

IPv6：
16 ビット（16 桁の 2 進数）×8 区切り=128 ビット
例：11ab:22bb:00bb:11aa:3344:1234:aabb:ccdd

グローバル IP アドレス

世界にひとつしかない IP アドレスです。インターネットに接続するには必ずグローバル IP アドレスが必要です。一般的には、**ISP(インターネットサービスプロバイダ)**が IP アドレスを多く保有し、ユーザーは ISP と契約し、ISP が保有する IP アドレスの空いているグローバル IP アドレス（リモートホスト）を貸し出されてインターネットを利用しています。



プライベート IP アドレス

LAN 上で通信可能な IP アドレスです。
インターネットで外部の回線に接続する場合は、ルーターを経由してグローバル IP アドレスを取得します。
LAN 内の IP アドレスは自由に設定が可能です。利用できる IP アドレスの範囲は世界共通。

IPv4 プライベートアドレス範囲（RFC1918）：
10.0.0.0～10.255.255.255（24 ビットブロック）
172.16.0.0～172.31.255.255（20 ビットブロック）192.168.0.0～192.168.255.255（16 ビットブロック）
IPv6 プライベートアドレス範囲：
fc00::/7 アドレスブロック=RFC4193 ユニークローカルアドレス（ULA）
fec0::/10 アドレスブロック=非推奨（RFC3879）

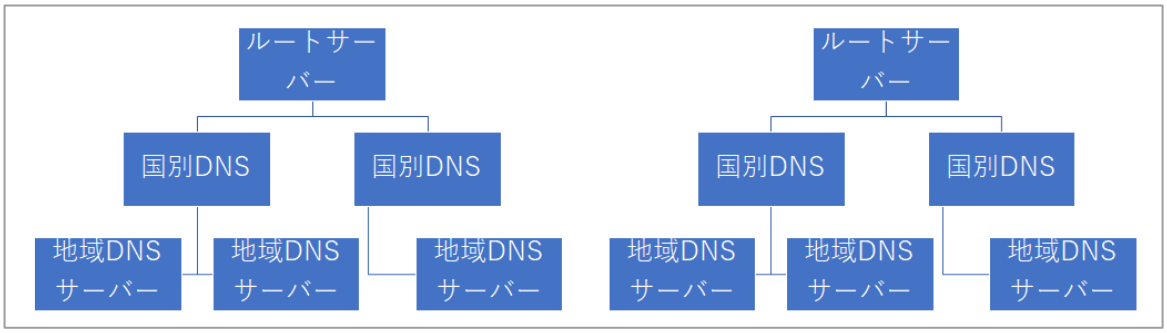
ドメインと DNS

- ドメイン

世界にひとつしかないネットワーク上の所在地情報
ネットワーク上の住所にあたる IP アドレスは数字の羅列のため扱いにくい
これを文字列で管理したもの
123.456.789.100 と ○○○○○.com は 1 対 1 で紐づいています。

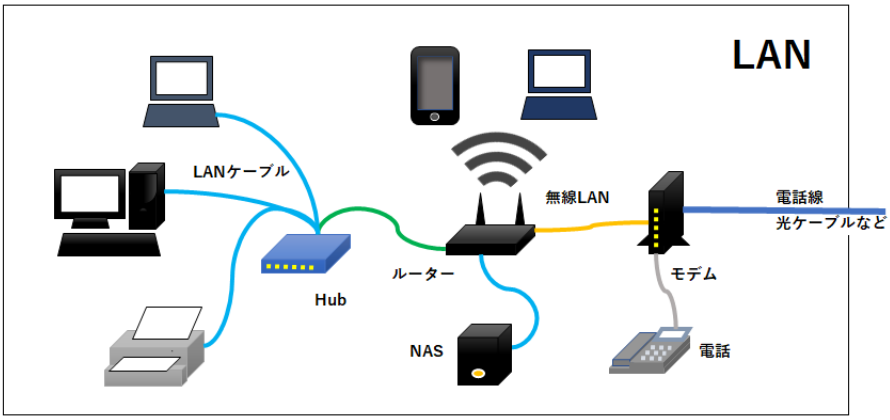
- ＤNS（Domain Name System）

IP アドレスとドメインの関連付け管理システムです。
世界に 13 のルートサーバーとその配下に複数のＤＮＳサーバーが存在しています。
階層構造で管理され、利用時は下位から問い合わせるため、上部しかその存在がわからないようになっています。



1.4.2.2.　ネットワーク機器

ＬＡＮを構成する機器の紹介



LAN ケーブル

通信を実現する伝送路（イーサネットケーブル・光ケーブルなど）です。

無線ＬＡＮの場合はケーブルの代わりに電波を利用します。

LAN ケーブルの通信速度

カテゴリの数字が大きいほど、早い機器に対応しています。

速い←　CAT7A,CAT7,CAT6A(6e),CAT6,CAT5e,CAT5　→遅い

＜注意＞回線速度が速くても古いカテゴリを使用していると、ポテンシャルを発揮することができません。また通信速度は経由するネットワーク上の一番遅い速度になりますので、ケーブルのカテゴリをよく見て使用してください。

＜注意＞LAN ケーブルは素材やカテゴリによって長さの制限があります。広域で使用する場合などには確認が必要です。

ハブ（Hub)

ＬＡＮ内で利用される複数のＬＡＮケーブルの集約装置

リピータハブ：受信データを接続された線危機に再送信

スイッチングハブ：データの宛先（送信先の機器）制御し送信先を指定

ルーター

ゲートウェイ（異なるネットワーク間で通信するための機器）の代表的な装置。

ＬＡＮ上のコンピュータが、インターネットなどに接続する際に利用する。

モデム・ターミナルアダプタ（ＴＡ）

電話回線などのアナログ信号をデジタルに変換する。

1.4.2.3. インターネットへ接続

通信回線の種類について解説します。

ＩＳＤＮ一般回線

交換機・中継回線・加入者線まで全てデジタル化された、パケット通信・回線交換データ通信にも利用できる公衆交換電話網。

ＡＤＳＬ

デジタル加入者線（DSL）の 1 つであり、ツイストペアケーブル通信線路（一般のアナログ電話回線）を使用する上り（アップリンク・受信）と下り（ダウンリンク・送信）の速度が非対称（Asymmetric）な、高速デジタル有線通信技術。

ＦＴＴＨ,

光ファイバーを伝送路として一般個人宅へ直接引き込む、アクセス系光通信の網構成方式のこと。安定した IP 電話・IP テレビ電話、光波長多重通信による多チャンネルのケーブルテレビ（デジタルテレビ放送を含む）の同時伝送等の多彩なサービスの提供が可能である。上り（≒送信）の帯域が ADSL よりも確保されているため、撮影した動画の送信や自宅サーバー営など大容量のデータをやりとりする環境に適している。

モバイル WiFi

モバイル Wi-Fi ルーターは、移動を容易にした小型の、ルーターと無線 LAN アクセスポイントの複合製品である。商用電源からの継続的な電源供給を望めない移動しながらのインターネット利用といった使途を想定しており、多くの機器で二次電池を内蔵している。

LAN 側インターフェイス

無線 LAN における接続互換性が Wi-Fi 規格にて確保されている。

WAN 側インターフェイス

モバイル WiMAX や携帯電話、PHS、他の公衆無線 LAN といった無線 WAN へ電波を用い直接ネットワーク接続するものの他に、自らは無線 WAN への接続機能は持たずに有線接続などによって外部機器（代表的には携帯電話機）経由で無線 WAN に接続するものの 2 種類に大別できる。

無線ＬＡＮ

イーサネットケーブルの代わりに電波を利用することで無線通信を実現。
一般的に WiFi と呼ばれる無線通信規格が利用されています。

Wi-Fi（Wireless Fidelity）は、「Wi-Fi Alliance」という業界団体が国際標準規格である無線 LAN 規格「IEEE 802.11」を使用していることを示すために発行している認証です。
「IEEE 802.11」に準拠した機器であれば Wi-Fi 認証をもらえます。

EEE 802.11 内の通信規格

無線 LAN の規格を定めているのが IEEE802.11 ですが、伝送規格は IEEE802.11b/11a/11g/11n/11ac、セキュリティ規格は IEEE802.11i、QoS 関連の規格は IEEE802.11e で定義しています。

規格名＼特徴	周波数帯	公称通信速度	チャンネル幅	特徴
IEEE802.11b	2.4GHz	11Mbps / 22Mbps	22MHz	高速だが干渉しやすい
IEEE802.11a	5GHz	54Mbps	20MHz	やや低速だが安定した方式
IEEE802.11g	2.4GHz	54Mbps	20MHz	a と b 企画の良いところを併せ持つ
IEEE802.11n	2.4GHz / 5GHz	65Mbps - 600Mbps	20/40MHz	技術発展により高速化
IEEE802.11ac	5GHz	290Mbps - 6.9Gbps	80/160MHz	帯域幅の拡大等でさらに高速化

無線 LAN のセキュリティ

- SSID

無線 LAN のネットワーク名。公衆 WiFi のように検索させないためにステルス機能を利用。

- 暗号化

通信傍受に対する対策。元データを暗号にして送受信する。
WEP ：古くから利用されている暗号化の企画でやや弱い
WPA：一定期間ごとに認証情報を更新するため安全
WPA2：WPA の強化版。次世代暗号化技術「AES」によりさらに安全

- MAC アドレスフィルタリング

通信機器の固有の MAC アドレスによって通信に参加する機器を指定することによりより安全にセキュアな環境を構築できます。

低消費電力・広範囲な IoT 通信を実現する「LPWAN」

LPWA(Low Power Wide Area) または LPWAN とは、なるべく消費電力を抑えて遠距離通信を実現する通信方式で、IoT の構成要素の 1 つとして近年注目されています。

LPWA (LPWAN) の体系

LPWA は、大きく「アンライセンス系」と「ライセンス系」とがあり、アンライセンス系は通信を行う時に免許は不要ですが、ライセンス系は無線局免許が必要です。
アンライセンス系 LPWA は、例えば個人や企業レベルで運用を行うことが可能ですが、ライセンス系 LPWA は従来の携帯キャリアのように総務省から包括免許の取得が必要です。

- 3G/LTE(ライセンス系), LoRaWAN, Sigfo（アンライセンス系）の利用例

	日本向け Air SIM plan-D/plan-K	グローバル向け Air SIM plan01s	グローバル向け Air SIM plan01s Low Data Volume	日本向け Air SIM plan-KM1(LTE-M)	LoRaWAN	Sigfox
スループット	最大 2Mbps	最大 2Mbps	最大 2Mbps	最大 1Mbps	最大 5.5Kbps	最大 0.1Kbps
利用可能エリア	日本	グローバル	グローバル	日本	日本	日本・欧州・アメリカ
購入可能エリア	日本	日本・アメリカ・欧州など		日本	日本	日本
消費電力	高	高	高	低 (eDRX/PSM利用時)	低	低
通信間隔	常時通信	常時通信	常時通信	常時通信 / 間欠通信 (eDRX/PSM利用時)	間欠通信	間欠通信
カバレッジ	NTT ドコモ /KDDI ネットワーク	海外キャリア ネットワーク	海外キャリア ネットワーク	KDDI ネットワーク	SORACOM LoRa Space のネットワーク範囲 または お客様設置によるプライベート ネットワーク	Sigfox ネットワーク
データサイズ	無制限	無制限	無制限	無制限	11~242 バイト / 回	12 バイト / 回
日次の通信可能回数	無制限	無制限	無制限	無制限	無制限 (ARIB 規定範囲内)	上り 140 回 / 下り 4 回
料金体系 ランニングコスト	初期費用 901 円 ~/ 回線 (NTT ドコモ) 1,500 円 / 回線 (KDDI) 基本料金 10 円 / 日 通信料金 0.2 円 ~/MB	初期費用 $5/ 回線 基本料金 $0.06/ 日 通信料金 $0.073~/MB *国によって通信料金は異なる	初期費用 $5/ 回線 基本料金 $0.4/ 月 通信料金 $0.5/MB *国によって通信料金は異なる	初期費用 1,500 円 / 回線 基本料金 100 円 / 月 通信料金 0.5 円 / KB	【共有基地局】 初期費用 24,800 円 / 台 月額利用料 9,980 円 / 台 【所有基地局】 初期費用 69,800 円 / 台 月額利用料 39,800 円 / 台 *月額利用料はゲートウェイ単位 + デバイス購入	初期費用 デバイス購入費のみ 年間利用料 1,440 円 / 年 *初年分はデバイス価格に含む
ユースケース	車両のテレマティックス・運行管理 見守り (子供・高齢者・ペット) 機器の稼働状況監視・異常検知・リモートメンテナンス サイネージ広告、監視カメラ		監視カメラや機器の 死活監視 センサーの on/off 通知		インフラ管理 (建物・道路・橋の老朽化や異常検知) 市政管理（マンホール、ゴミ箱の管理） スマート畜産・農業	

2018.07.03 時点

出展 https://soracom.jp/lpwa/

- IoT アプリケーション毎の想定データ通信量、消費電力、通信頻度、通信範囲の例

Applications	Number of message per day	Size of message	Total daily load	Battery requirement	Coverage requirement
Consumer-Wearables	10 messages/ day	20 bytes	10*200 = 200 bytes	1 - 3 years	Outdoor/indoor
Smoke detector	2 messages/day	20 bytes	2*20 = 40 bytes	5 years	Indoor
Water metering	8 messages/day	200 bytes	8*200 = 1600 bytes	15 years	Deep indoor
Vending machine	1 message/day	1 Kbytes	1*1000 = 1000 bytes	10 years (powered)	Outdoor/indoor

出展 https://soracom.jp/lpwa/

2 行目の火災報知器では 1 回 20byte、通信頻度は 1 日 2 回、4 行目の自動販売機は 1 日 1 回通信ではあるものの、1 回の通信量は 1kbyte です。火災報知器であれば LoRaWAN 等の低速通信が、自動販売機であれば LTE Cat.M1 のようにある程度スループットを確保できる方式がよい。

ここまで、IoT/コネクテッドホームと必要な技術について学習しました。
第 2 章では、アプリケーションの作成のための組み込みプログラミング言語について学習します。

2. mruby/c プログラミング

2.1. プログラミング言語「mruby/c」とは

プログラミング言語 mruby/c について学びます。以下の内容について習得できます。

- プログラム言語 Ruby の概要
- mruby/c の基本的なプログラミング文法

2.1.1.mruby

これから、プログラミング言語 mruby/c を使ってプログラムを作成していきます。mruby/c は、プログラム言語 Ruby から派生して開発された言語です。まずはじめに Ruby について説明します。
プログラム言語 Ruby は、まつもとゆきひろ氏により日本で開発された国産のプログラム言語で、1995 年に登場しました。Ruby は、可読性を重視した構文となっています。学術的には、Ruby では整数や文字列なども含めてすべてのデータ型はオブジェクトであり、純粋オブジェクト指向言語と呼ばれます。公的規格としては 2011 年 3月22日に JIS 規格（JIS X 3017）が制定され、その後 2012 年 4月1日に日本発のプログラム言語では初めて ISO/IEC 規格（ISO/IEC 30170）として承認されました。
Ruby は、Web アプリケーション（Web サーバー内で動作するアプリケーション）の開発ではポピュラーな言語となっており、多くの有名なサイトで利用されています。
このような背景の中、Ruby を組込みソフトウェア（機械などを制御するソフトウェア）に利用することを目的とした開発事業が行われ、2012 年に mruby が発表されました。mruby は、Ruby の持つ開発しやすさを持ちつつ、組込みソフトウェアで利用できる省メモリなどの用件を満たす実装となっています。現在、mruby は、以下のサイトで公開されています。

http://mruby.org/

mruby は、組込みソフトウェアの開発に利用されます。組込みソフトウェアが利用される分野は幅広く、自動車や家電、携帯電話などの身近な製品もあれば、産業用ロボットや製造装置（製造プラントなどの機器）、人工衛星などの特殊な装置もあります(1)。一般に組込みソフトウェアは、装置の中にあるマイコンボードなどで実行されます。このような装置のソフトウェア開発では、手元の開発用PC でプログラムを作成し、そのプログラムを装置内のマイコンボードへ転送して実行します。
この教材でも、PC でプログラムを作成し、そのプログラムをケーブルや無線通信を使ってマイコンボードへ転送して動かす演習を行います。

(1): 福岡市の株式会社 QPS 研究所が、Ruby を利用した小型人工衛星を開発しています。

2.1.2. mruby/c

mruby/c は、mruby をさらに小型化したプログラム言語環境です。言語仕様としては、Ruby、mruby と同じく、ISO/IEC および JIS に準拠しています。Ruby、mruby、mruby/c の違いを表で示します。

	Ruby	**mruby**	**mruby/c**
実行形式イ	インタープリタ方式	コンパイラ＋VM方式	← 同じ
実行時に必要なメモリ（プログラムにより異なる）	数 MB 程度	400KB 程度	64KB 程度
対象を想定する機器	PC やサーバー	小型マイコン Raspberry Pi など	ワンチップマイコン

mruby/c はワンチップマイコンでの動作を想定しています。現在の mruby/c では、ワンチップマイコンを制御するために必要な以下のような機能を提供しています。

- GPIO（デジタル入出力）
- PWM 出力
- ADC入力
- I2C インターフェイス

2.1.3. mruby/c の簡単なプログラム

mruby/c のプログラムを作成するために必要な内容を習得していきます。まず最初に、ワンチップマイコンの機能を使う簡単な mruby/c のプログラムを示します。

```
1: while truee
2: led 1
3: sleep_ms 500
4: led 0
5: sleep_ms 500
6: end
```

このプログラムを実行すると、マイコンボード上の LED が点滅します。このプログラムを使って、mruby/c プログラミングの動作について説明していきます。
mruby/c のプログラムは、プログラムの 1行目から順に実行されます。1行目の while true は、対応する 6行目の end までの間を繰り返し実行します。つまり、このプログラムが、2行目から 5行目までの 4行を繰り返し実行することになります。
2行目の led 1 は、ハードウェアを制御する led 命令を呼び出します。led 命令は、マイコンボード上の LED を制御する命令で、渡されたパラメータ（led のに続く値）により動作が異なります。led 1 の場合は、LED を点灯させます。

led	命令	動作内容
led	0	LEDを消灯させる
led	1	LED を点灯させる

3行目の sleep_ms 500 は、ハードウェアを制御する sleep_ms 命令を呼び出します。sleep_ms 命令は、プログラムを一定時間停止させる命令で、渡されたパラメータの時間（単位はミリ秒）の間、プログラムは停止します。sleep_ms 500 の場合は、プログラムが 500 ミリ秒間（0.5 秒間）停止します。

sleep_ms	命令	動作内容
sleep_ms	n	プログラムを n ミリ秒停止させる

2.1.4. 基本文法

while 文

while <条件> ~ end の間を、条件を満たしている間繰り返します。条件に true を指定した場合は、常に満たす条件となり、永久に繰り返すことになります。条件には、等号や不等号を用いた条件式を書きます。

次の例は、「プログラム」の部分を永久に繰り返し実行します。

```
while true
  プログラム
end
```

次の例は、「プログラム」の部分を 5 回繰り返します。

```
n = 1
while n <= 5
  プログラム
n = n + 1
end
```

if 文

if <条件> then ~ end の間を、条件を満たす場合に一度だけ実行します。また、else を使うことで、条件を満たさない場合の実行もできます。

次の例は、a の値が 5 以上の場合に「プログラム」の部分を実行します。

```
while a>=5
  プログラム
end
```

次の例は、a の値が偶数場合に「プログラム1」の部分を、奇数の場合に「プログラム2」の部分を実行します。

```
if a%2 == 0 then
  プログラム1
else
  プログラム2
End
```

繰り返し

ある決まった処理を何度か実行する場合には、繰り返しを使います。繰り返しは、while でも記述できますが、次のような繰り返しも記述できます。

- 繰り返しの回数が決まっている場合

整数に対して、times を呼び出します。例えば、「プログラム」を 5 回繰り返す場合には、次のように記述できます。

```
5.times do
  プログラム
end
```

- 繰り返しの回数が必要な場合

ある変数（ここでは変数 a）の値を 1～5 まで変えながら 5 回繰り返すような場合には、for を使います。プログラム内で、変数 a を使うことができます。

関数

ある決まったプログラムを1つにまとめて、関数を作成できます。関数を作成することで、同じようなプログラムを、きれいにまとめることができます。具体的な例を使って説明します。

つぎのようなプログラムを考えます。

```
プログラムa
プログラムb
プログラムX
プログラムa
プログラムb
プログラムY
プログラムa
プログラムb
プログラム Z
```

プログラムa と プログラムb が何度も登場しますが、その間にプログラムX、プログラムYがあるので、繰り返しは使えません。そこで次のように、def命令を使って、関数progを作成できます。これにより、同じ部分（プログラムaとプログラムb）をまとめることができます。

```
def prog
  プログラムa
  プログラムb
end
prog
プログラムX
prog
プログラムY
prog
プログラム Z
```

2.2. 組み込みプログラミング 1

mruby/c を使ったプログラムを動かしながら、IoT プログラミング作成の手順や動作について学習します。

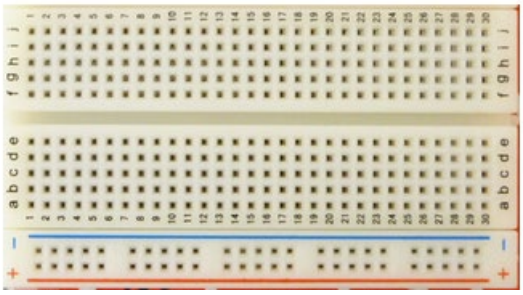
- プログラミングの開発環境構築
- mruby/c を利用した簡単なプログラム作成と実行の手順

2.2.1. プログラミングの開発環境構築

mruby/c を使ったプログラミング環境は、以下のソフトウェアとハードウェアから構成されています。

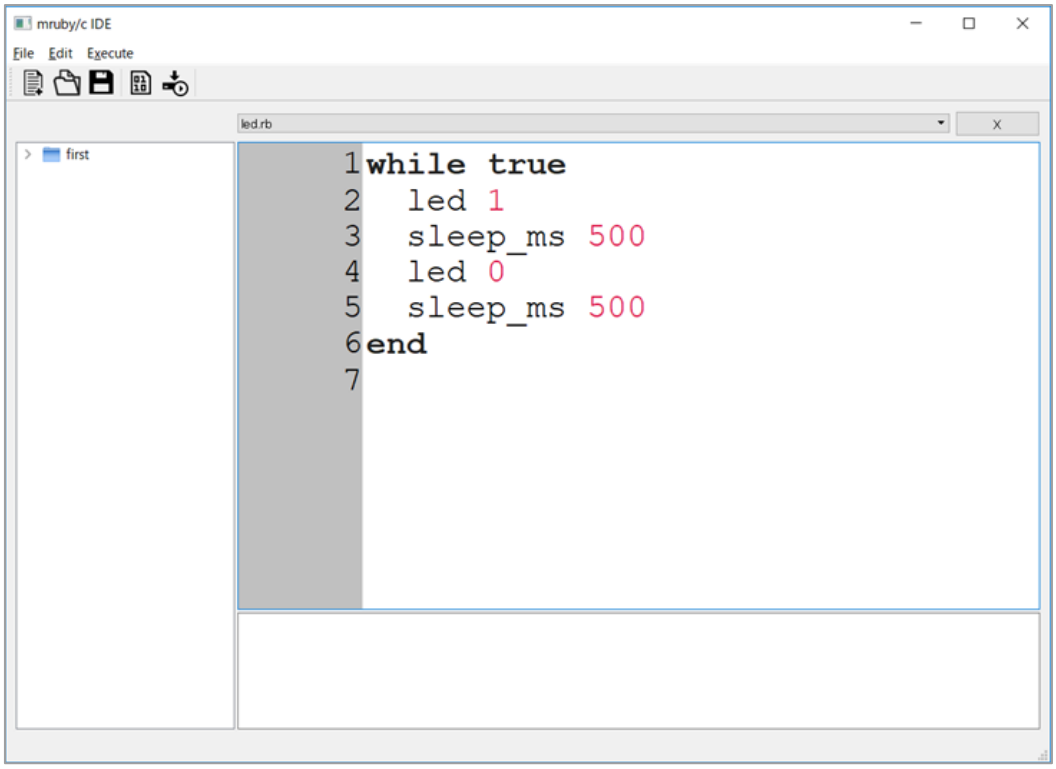
- ハードウェア

PSoC5 マイコンボードとブレッドボード



- ソフトウェア

mruby/c 統合開発環境



統合開発環境がインストールされた PC と、マイコンボードを USB ケーブルで接続して、PC で記述したプログラムをマイコンボードへ転送して実行するまでの手順を説明します。
初めて統合開発環境を利用するときは、次の「設定」を行います。すでに「設定」を済ませていれば、「簡単なプログラム作成と実行」へ進みます。

設定

統合開発環境では、PC と USB 接続されたマイコンボードを利用したプログラムの作成と実行ができます。
USB 接続のための設定を行います。

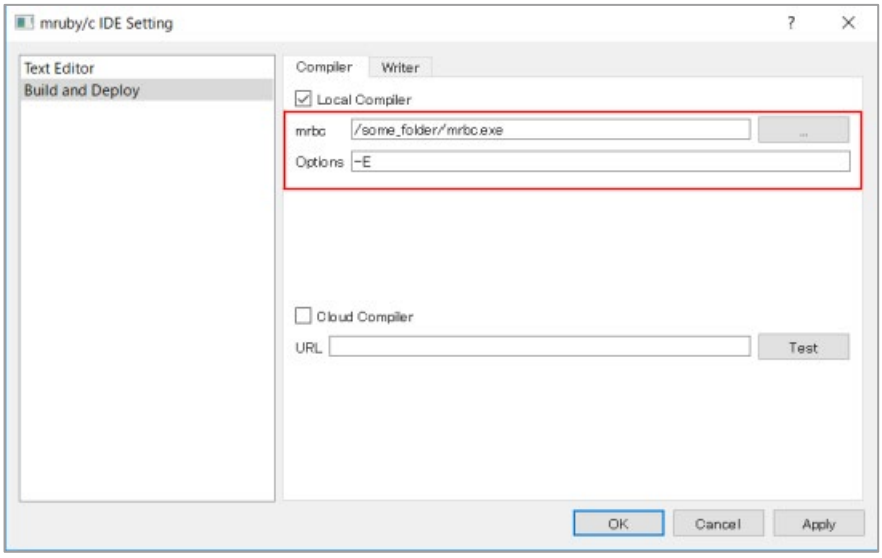
統合開発環境を起動して、以下の設定を行います。

- mruby/c コンパイラの設定
- USB 接続の設定

mruby/c コンパイラの設定

[File]-[Settings...]メニューを開き、設定のためのダイアログを呼び出します。ダイアログの左のリストから、「Build and Deploy」を選択します。この中の「Compiler」タブ内で、以下の設定をします。

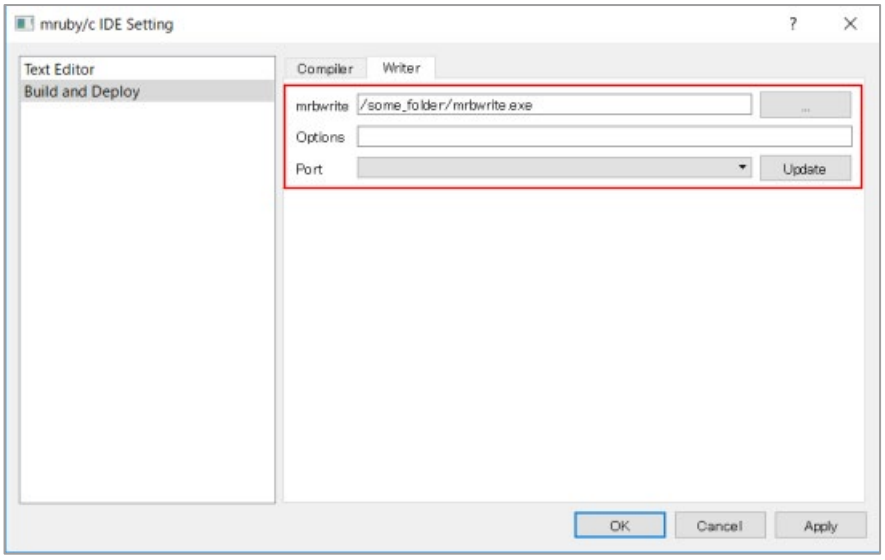
- 「mrbc」: ファイル選択のボタンをクリックして、mrbc.exe を選択します
- 「Options」: -E を記入します



USB 接続の設定

「Writer」タブ内で、以下の設定をします。

- 「mrbwrite」: ファイル選択のボタンをクリックして、mrbwrite.exe を選択します。
- 「Port」:「USB シリアルデバイス (COM**)」を選択します。COM**の数値は環境によって異なります。



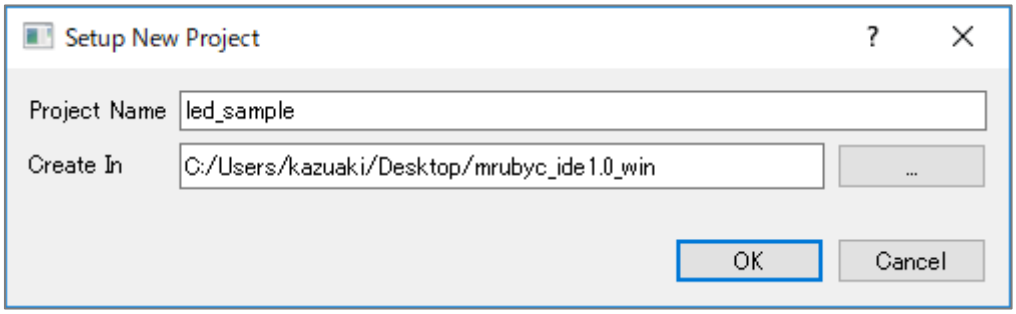
以上の設定ができたら、ダイアログを閉じます。これで設定は完了です。

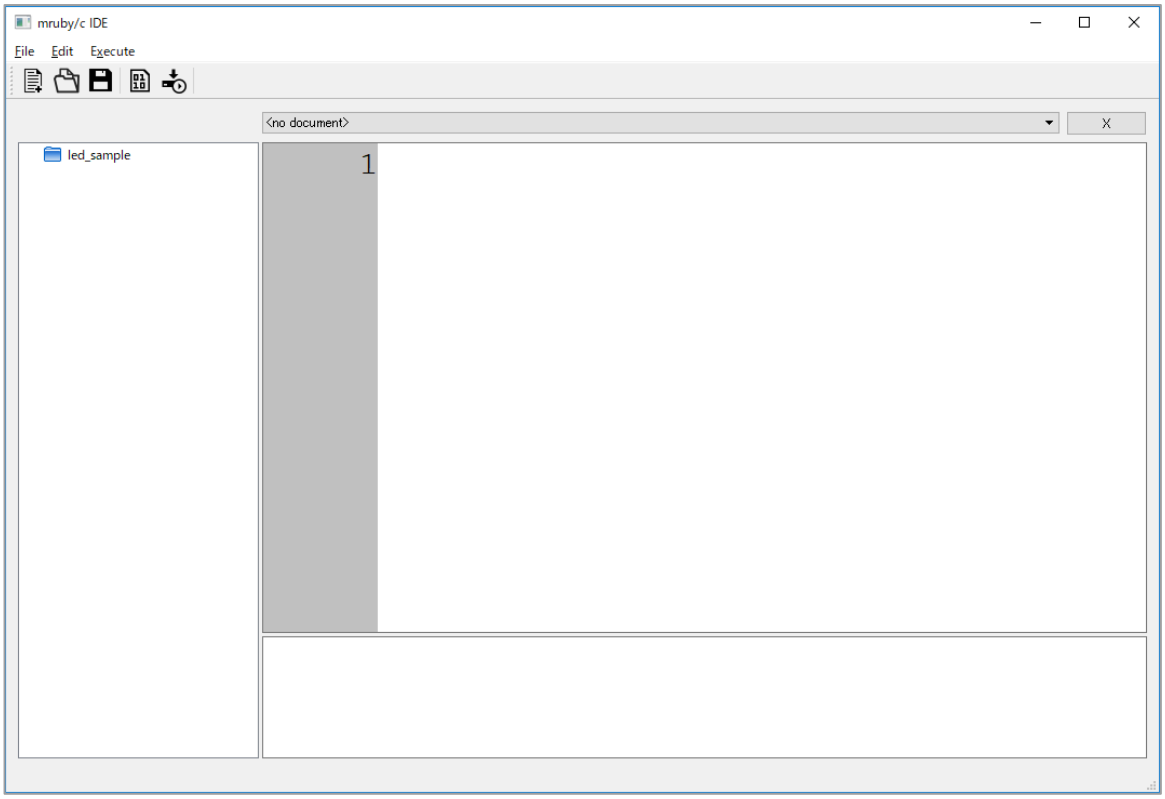
簡単なプログラム作成と実行

ここでは、第 3 回で紹介した LED を点滅させるプログラムを、マイコンボードを使って実行させてみます。実行までの手順を示していきます。

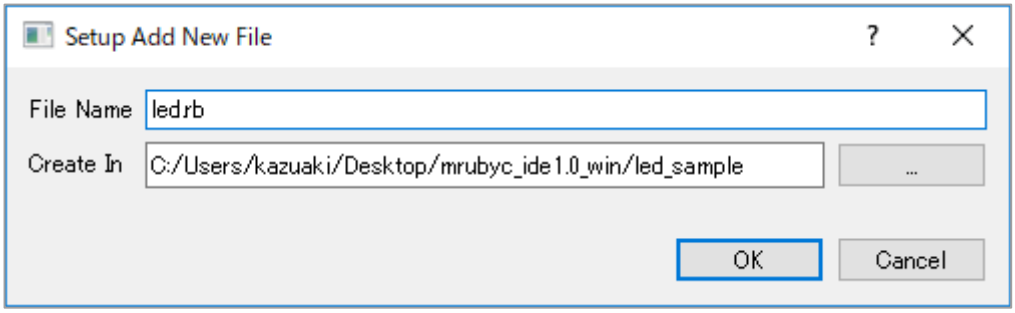
最初に統合開発環境を起動した状態です。あらかじめ「設定」を済ませておいてください。

[File]-[New]-[Project] を選択して、プロジェクト名を入力します。ここでは、「led_sample」としました。プロジェクト名には、英数字が使えます。



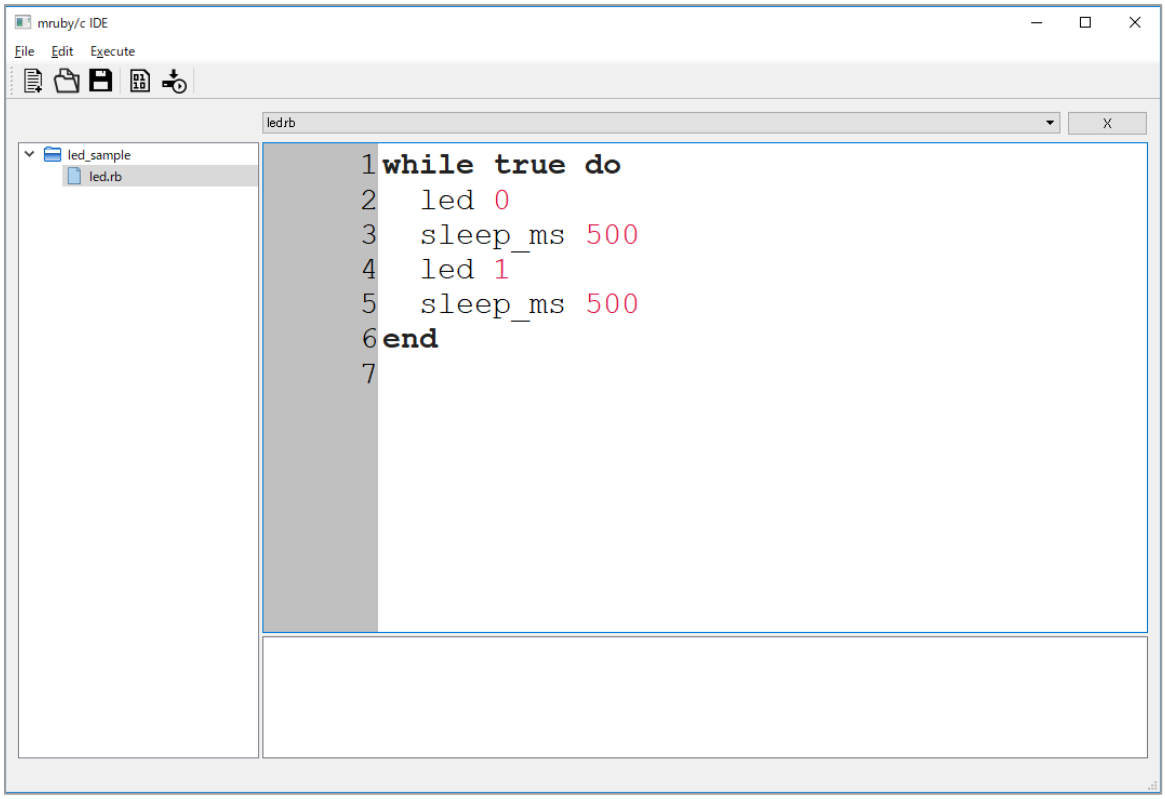


次に、mruby のファイルを追加します。[File]-[New]-[File] を選択して、ファイル名を入力します。ファイル名を「led.rb」としています。ファイル名は英数字で、ファイル拡張子として「.rb」を付けてください。

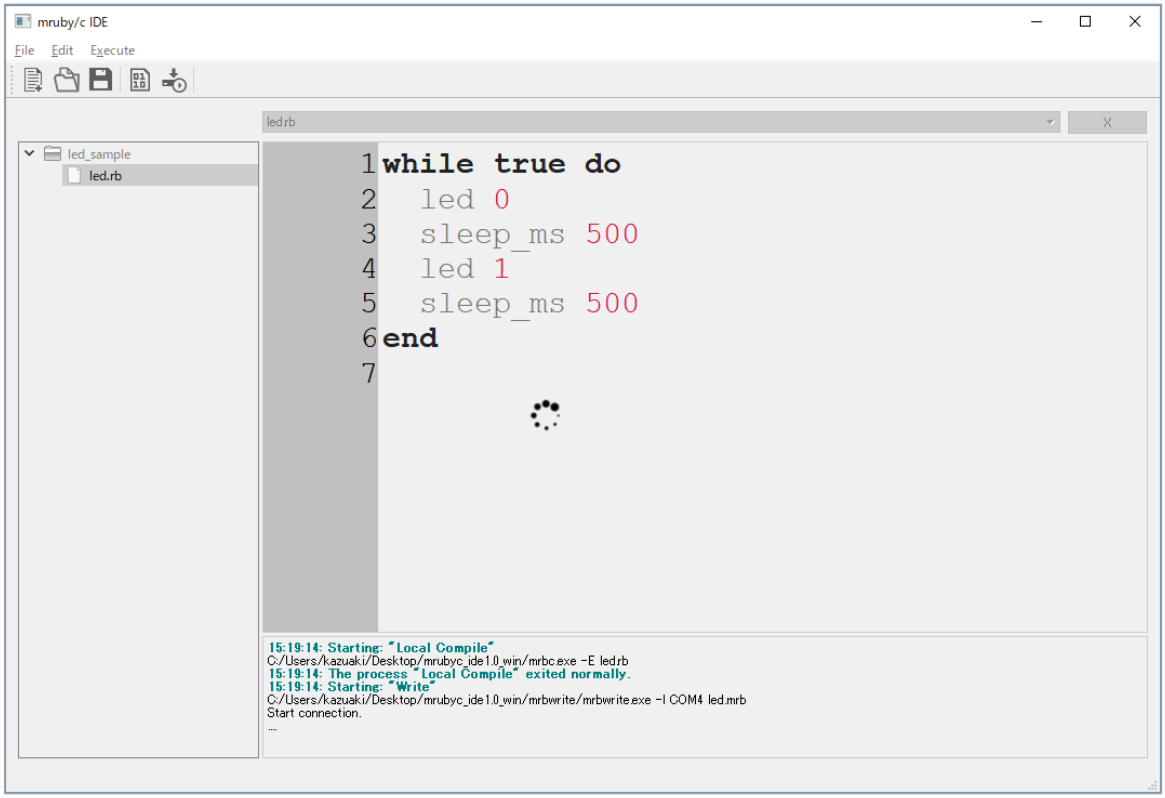


プログラムを記述します。

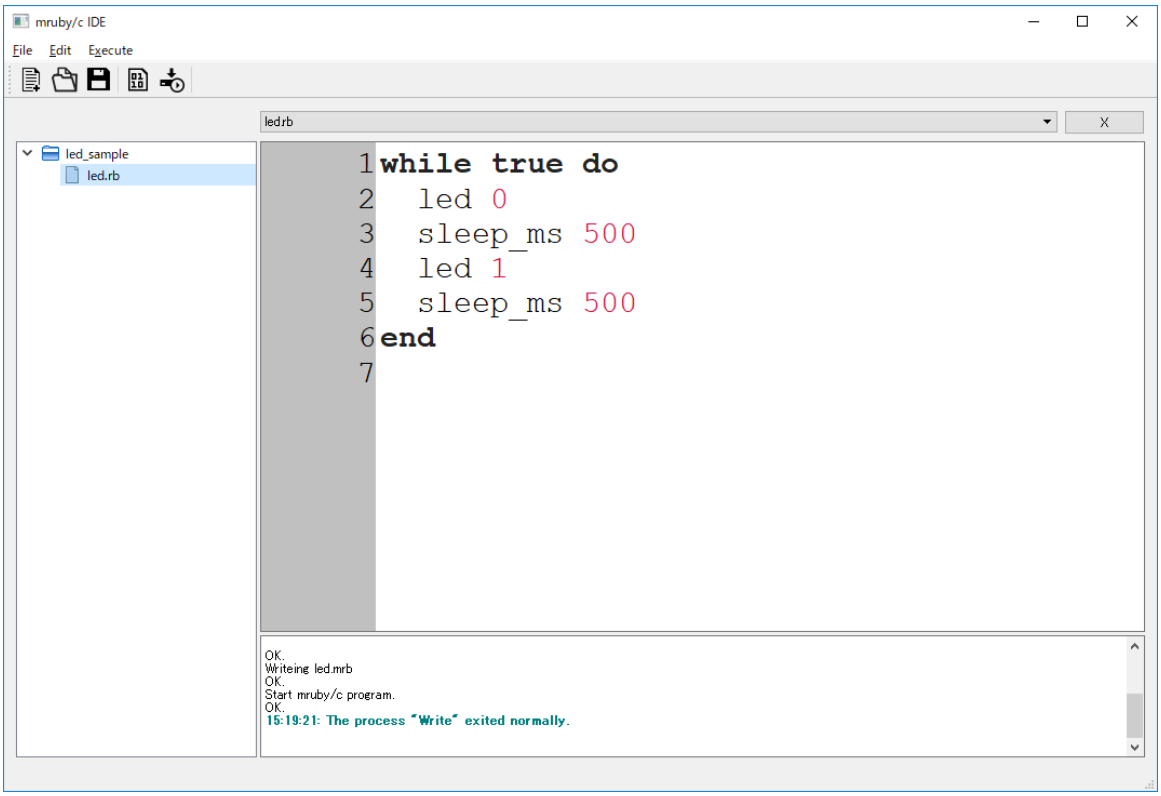
```
while true do
  led 0
  sleep_ms 500
  led 1
  sleep_ms 500
end
```

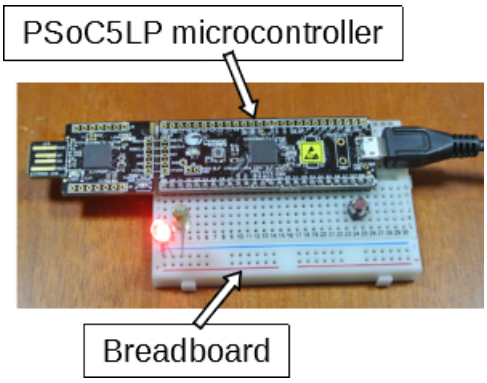


[File]-[SaveAll] で作成したプログラムを保存します。
次に、プログラムをコンパイルして、マイコンボードに転送します。[Execute]-[Write] を選択すると、下図のような転送待ち状態になるので、マイコンボードのリセット（RST）を押します。



プログラムの転送に成功すると、下図のように OK が表示されます。プログラムが正しく実行されると、マイコンボードの LED が点滅します。





2.3. 組み込みプログラミング 2

統合開発環境を使って、mruby/c のプログラムを作成していきます。以下の内容について学習します。

- デジタル出力を制御するプログラム
- PWM 出力とそのプログラム

2.3.1. デジタル出力を制御するプログラムをつくってみよう

デジタル出力

デジタル出力を制御するプログラムを作成します。

デジタル出力とは

デジタル出力とは、マイコンボードの出力電圧を制御する仕組みを使います。デジタル出力ピンの電圧を LOW（演習で使用するマイコンの場合 0V）または HIGH（5V）に設定できます。

デジタル出力に使えるのは、マイコンボード面の印刷で「3.0」～「3.7」のピンです。それぞれのピンと出力する電圧を指定できます。

output 命令	説明
output <pin番号>, <値>	<pin番号>: 0～7の値を指定します。 それぞれ、3.0～3.7に対応するピンです。 <値>: 0（LOW）または1（HIGH）を指定します。

デジタル出力を使ったプログラム

output 命令を使ったプログラムを示します。このプログラムの動作は、前回の LED（led 命令）を使ったプログラムとほぼ同じなので、容易に理解できると思います。

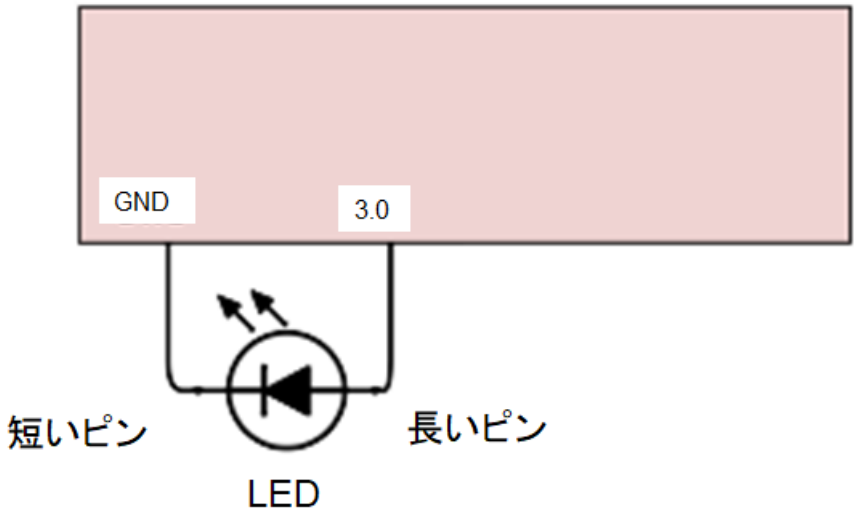
```
while true
  output 0,1
  sleep_ms 500
  output 0,0
  sleep_ms 500
end
```

このプログラムを実行すると、0.5 秒ごとにピン番号 3.0 の電圧が LOW と HIGH の値に変化します。しかし電圧の変化そのものは、目で確認できないので、正しく動作しているかを判断できないと思います。

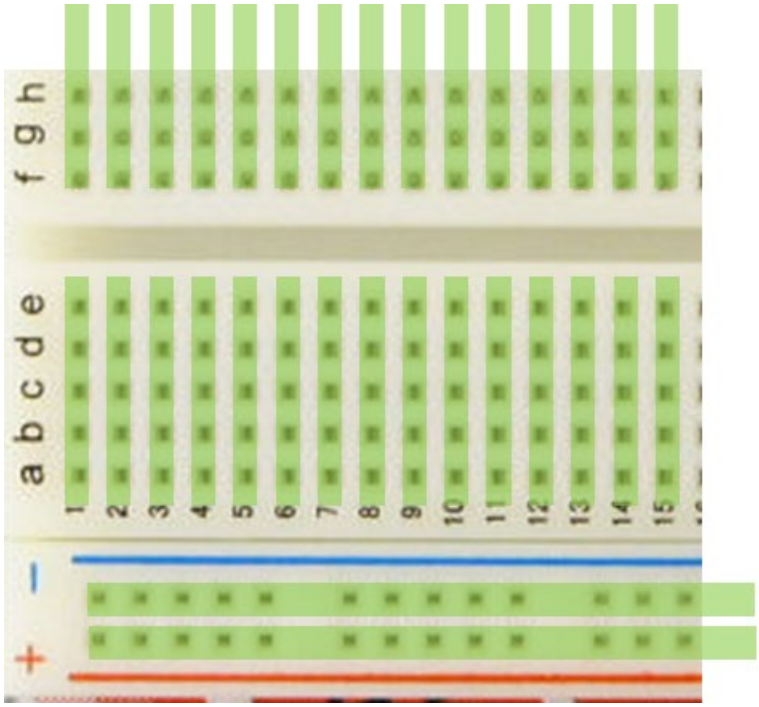
LED を使った回路

ここで、いったん USB ケーブルを外します。
電圧が変化することを確認するために、LED 部品を使います。LED は、図のように 2 本の足（ピン）の長さが異なります。LED に流す電流は、長い足から短い足に流れるように回路を作ります。

LED は、電流を流すことで光を発する半導体部品です。LED を使って、次のような回路を作成します。図の中で、GND は常に LOW となるピンです（マイコンボードにも印刷されています）。



LED の足をマイコンボードのピンに接続するために、ブレッドボードを利用します。ブレッドボードには多くの穴が並んでいますが、穴の中で図の緑色の線のように接続されています。



この接続を利用して、
部品の端子をつなぐ

最終的には、図のようにLEDを使った回路を作成できます。回路図通りに、ピン3.0からLEDの長い足に、LEDの短い足からGNDに、それぞれ接続されていることがわかります。

プログラムの実行

回路が完成したら、USBケーブルを接続します。これで、LEDが点滅し始めます。このマイコンボードでは、最後に転送されたプログラムを記憶しているので、先ほど作成したプログラムが動き始めます。

mruby/cのプログラムと、LEDの電子回路を組み合わせることで、このようなプログラムを作成できます。

演習課題

以下の動作を行うプログラムを作成してみてください。

1. LEDの点滅スピードを変える。
2. LEDを2個使って、交互に点滅させる。
3. LEDを2個使って、1つ目のLEDを3回点滅、2つ目のLEDを5回点滅させる内容を繰り返す。
   例えば、「赤LEDを3回点滅、次に、緑LEDを5回点滅」を繰り返す。

発展課題

以下の動作を行うプログラムを作成してみてください。

1. 4つのLEDを順に点滅させる。ただし、できるだけ繰り返しを使い、プログラム全体で`output`命令を1つ、または、2つ程度とする。

2.3.2. PWM 出力するプログラムをつくってみよう

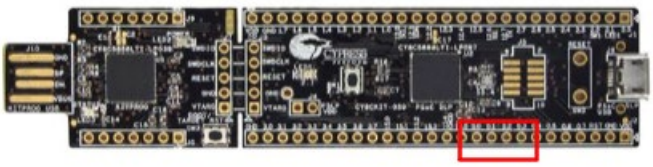
PWM 出力

PWM（Pulse Width Modulation）出力を制御するプログラムを作成します。デジタル出力では、LOWまたはHIGHの値を出力できます。したがって、デジタル出力でLEDを扱う場合は、消灯または点灯だけができます。
これに対して、LEDの明るさを変化させたい場合があります。このような場面で利用するのが、PWM出力です。PWM出力は、疑似的なアナログ出力とも言えます(1)。

(1): DACという回路を使うことでアナログ出力が可能です。多くの場合、アナログ出力を疑似的に実現するPWMで十分なことが多く、回路も簡単になるので、この演習でもPWMを使っています。

PWM とは

PWMでは、デジタル出力のLOWとHIGHを高速に切り替えることで、LOWとHIGHの時間の比率に応じた中間的な出力ができます。LOWとHIGHの時間の比率のことを、デューティ比と呼び、例えばデューティ比が0%であればLEDは消灯、100%であればLEDは点灯します。デューティ比を50%にした場合、LEDの光量が50%になることから、少し暗く点灯することになります。
PWM出力に使えるのは、マイコンボード面の印刷で「0.0」～「0.3」のピンです。それぞれのピンと出力するデューティ比を指定できます。



PWM出力ピン
0.0～0.3

output 命令	説明
pwm <pin番号>, <デューティ比>	<pin番号>: 0～3の値を指定します。それぞれ、0.0～0.3に対応するピンです。 <値>: 0（0%）～255（100%）の値でデューティ比を指定します。256段階のデューティ比を指定できます。

PWM 出力を使ったプログラム

output命令を使ったプログラムを示します。このプログラムの動作は、前回のLED（led命令）を使ったプ
ログラムとほぼ同じなので、容易に理解できると思います。

```
while true
  pwm 0, 0
  sleep_ms 500
  pwm 0, 64
  sleep_ms 500
  pwm 0, 128
  sleep_ms 500
  pwm 0, 255
  sleep_ms 500
end
```

PWM出力のための回路は、次のようになります。LEDを使った回路ですが、ピン番号が0.0となっているので注意してください。
このプログラムを実行すると、0.5秒ごとにLEDの明るさが段階的に変化することを確認できます。

演習課題

以下の動作を行うプログラムを作成してみてください。

1. LEDの明るさを5段階に変化させる。
2. 繰り返し（for）を使って、0～255までデューティ比を変化させる。
   sleep_msで指定する時間に気を付けること。forによって256回の繰り返しとなるので、1回のプログラム実行時間は、sleep_msで指定する時間の256倍となる。

2.4. 組み込みプログラミング 3

統合開発環境を使って、mruby/cのプログラムを作成していきます。
以下の内容について学習します。

- マルチプログラミング
- センサ入力を伴うプログラム

2.4.1. マルチプログラミング

マルチプログラミングとは、複数のプログラムを並行させて実行する技術です。複数のプログラムを作成しておき、それらを同時に実行させることができます。マルチプログラミング技術は、IoTプログラムを作成する時にとても役立ちます。
IoTでは、さまざまなセンサやネットワーク通信を使うことになります。センサの種類によってデータの取得タイミングが異なっていたり、ネットワーク通信においての送受信待ちの時間が発生したりします。このような異なるタイミングで動作する機能を、一つのプログラムだけで作成するのは困難です。
マルチプログラミング技術を使えば、タイミングが違う機能ごとに別のプログラムを作成しておき、それらを同時に実行させれば、全体として一つのシステムとして動作させることができます。

● mruby/c でのマルチプログラミング

mruby/cは、マルチプログラミングをサポートしています。

mruby/cでは、VM(1)がプログラムを実行します。VMは、一定時間ごとに動作させるプログラムを切り替える機能を持っていて、複数のプログラムを並行して実行させる場合は、1ミリ秒ごとにプログラムを切り替えながら実行を進めます。利用者にとっては、プログラムの切り替えは認識できないので、複数のプログラムが同時に実行されているように感じます。

(1): VM=Virtual Machine。mruby/cのプログラムは、あらかじめコンパイルを行うことで、実行時に必要なメモリを削減しています。コンパイルされたプログラムは「バイトコード」と呼ばれ、VMはバイトコードを逐次実行します。

● マルチプログラミング

ここでは、LEDを点滅させるプログラムを応用として、2つのLEDを違うタイミングで点滅させるプログラムを実行します。
[File]-[New]-[Project] を選択して、プロジェクト名を入力します。ここでは、「two_leds」としました。

次に、mruby のファイルを追加します。[File]-[New]-[File] を選択して、ファイル名を入力します。2 つのプログラムを作成するので、ファイル追加を 2 回行います。ファイル名をそれぞれ「led1.rb」、「led2.rb」としています。

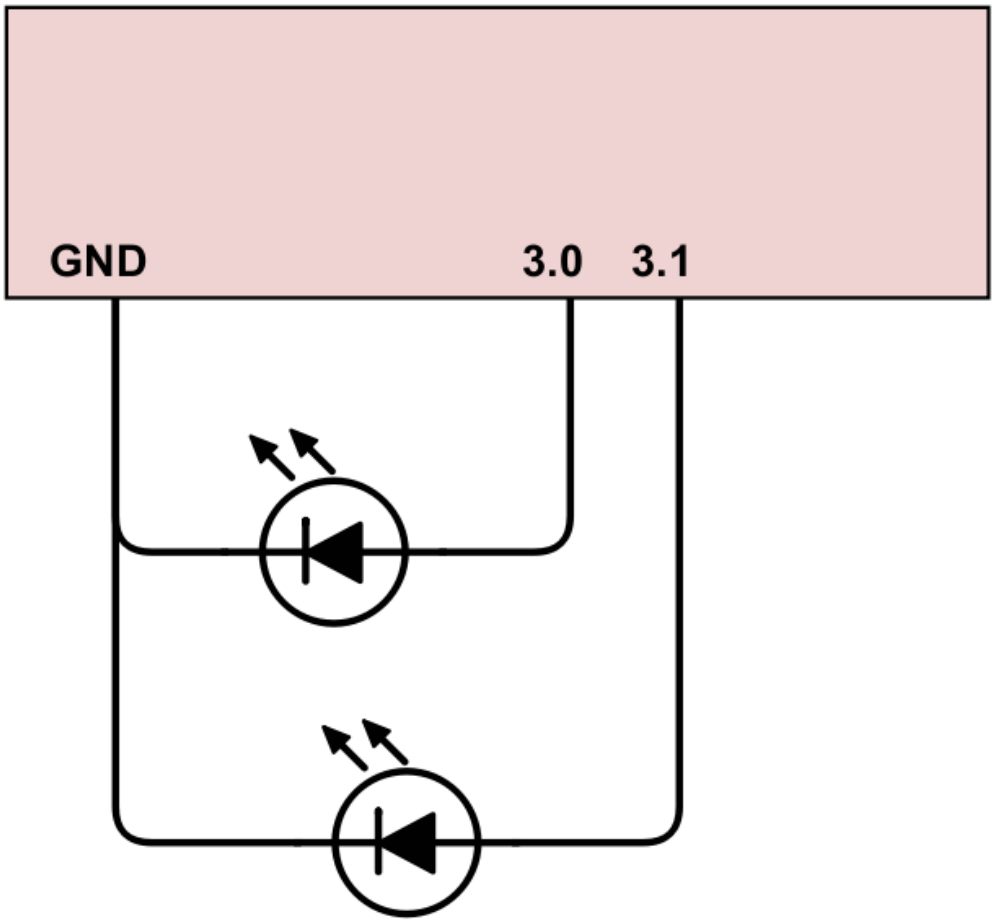
追加されたファイルは、統合開発環境の左のリストで確認できます。

それぞれのプログラムを作成します。ここでは、2つのLEDを異なるタイミングで点滅させるので、次のようなプログラムを作成します。

```
while true
  output 0,1
  sleep_ms 500
  output 0,0
  sleep_ms 500
end
```

```
while true
  output 1,1
  sleep_ms 150
  output 1,0
  sleep_ms 150
end
```

このプログラムでは、デジタル出力を使って3.0と3.1のピンの電圧を制御します。したがって、それぞれのピンにLEDを接続するような次の回路を作成します。



回路が完成したら、USBケーブルを接続してプログラムを実行します。2つのLEDが異なるタイミングで点滅し始めることを確認できます。このように、複数のプログラムを同時に実行させるプログラムを簡単に作成できます。

● 入出力を行うプログラム

マルチプログラミングは、センサを扱うプログラムを作成する際に役立ちます。ここでは、センサの代わりにスイッチを使って、スイッチの入力と、LEDの出力を組み合わせたプログラムを作成してみます。
スイッチの入力を行うための命令として、swがあります。

sw 命令	説明
Sw	スイッチが押されているならば1、押されていないならば0となります。

スイッチを使った最も簡単なプログラムは、次のようになります。このプログラムでは、スイッチを押していればLEDが点灯し、スイッチを押していなければLEDが消灯します。

```
while true
  if sw==0 then
    led 0
  else
    led 1
  end
end
```

この程度の処理であれば、1つのプログラムで十分に実現できます。より複雑な動作を考えてみます。

スイッチを押すたびに、LEDの点滅速度を変化させるプログラムを考えてみます。次のようなプログラムとなるでしょう。

```
t = 100
while true
  # スイッチのチェック
  if sw==1 then
    if t==100 then
      t = 1000
    else
      t = 100
    end
  end

  # LED点灯
  led 1
  sleep_ms t
  # LED消灯
  led 0
  sleep_ms t
end
```

最初の「スイッチのチェック」の部分で、スイッチの状態に応じて変数tの値を変化させます。このtの値は、LEDを点滅させる際の時間になります。このプログラムを実行して、動作を確認してください。

このプログラムの問題点

実行して動作を確認してみると、思ったような動作をしないという問題点に気づきます。
具体的には、「スイッチを押すタイミングが難しい」ということです。これは、スイッチの状態をチェックするsw命令が実行されるタイミングが100ミリ秒または1000ミリ秒に一度であるためです。つまり、タイミングよくスイッチを押す必要があり、これが思ったような動作とならない原因です。
この問題点を整理すると、スイッチの状態をチェックするタイミングと、LEDの点滅のタイミングが異なっていることが原因にあります。言い換えれば、スイッチの状態は人が自由に決めるのに対して、LEDの点滅のタイミングは 100ミリ秒または1000ミリ秒のタイミングであり、この違いが問題となっています。

入出力を行うプログラムの改良

スイッチをチェックするプログラムと、LEDを点滅させるプログラムを分けることで、先の問題点を解決できます。ここで問題になるのは、1つ目のスイッチの状態を、2つ目のLED点滅のプログラムに渡す方法です。

mruby/cでは、通常の変数は、一つのプログラム内だけで使えます。一方、変数に$を付けることで、異なるプログラム間で利用できる変数となります。
次のような2つのプログラムを作成します。

```
$t = 100
while true
  # スイッチのチェック
  if sw==1 then
    if $t==100 then
      $t = 1000
    else
      $t = 100
    end
  end
end
```

```
WHILE TRUE
  # LED点灯
  LED 1
  SLEEP_MS $T
  # LED消灯
  LED 0
  SLEEP_MS $T
END
```

プログラムそのものは、先のプログラムを2つに分けただけですが、2つのプログラムが並行して実行されることで、スイッチを押すタイミングと、LEDの点滅のタイミングが異なっていても問題なく動作します。

3. SenStick mruby/c につて

はじめに

SenStick は、代表的な 8 種類の MEMS センサを超小型基板に実装した世界最小クラスの BLE 搭載マルチセンシングボードおよび周辺ソフトウェアの総称です。電子工作をすることなく、購入後すぐに、IoT の本質であるデータ分析に注力できるように開発されました。iOS/Android 向けのアプリケーションや Node.js のライブラリを用いることで誰でも手軽にデータ計測が可能になります。小型かつスティック型という珍しい形状は、メガネやお箸、杖といった細長いモノや小さいモノを IoT 化するのに向いています。ボタン電池駆動ではなく、一般的な LiPo バッテリーを採用しており、1 回の充電で 24 時間以上（105mAh として）のセンシングが可能です。また、BLE チップとして、Nordic 社の nRF52 を搭載しており、Cortex-M4F による高度なデータ処理も可能です。一般的な BLE センサと異なり、大容量フラッシュメモリ（32M バイト）をボード上に実装しているため、スマートフォン無しで連続ロギングが可能なことも特徴の一つです。そして、SenStick は、回路図、ファームウェア、周辺ソフトウェア、すべて大学の研究成果として公開（https://github.com/ubi-naist/SenStick）しており、自身でファームを書き換えてオリジナルのセンサボードを開発することも可能です。

仕様一覧

搭載センサ：8 種類

モーション系：加速度・ジャイロ・地磁気

環境系：温度・湿度・気圧・光・UV

リチャージャブル

一般的な LiPo バッテリーを利用可能

用途に応じて、自由にバッテリー容量を変更可能

充電機能もボードに搭載

省電力かつ高性能 BLE チップの採用

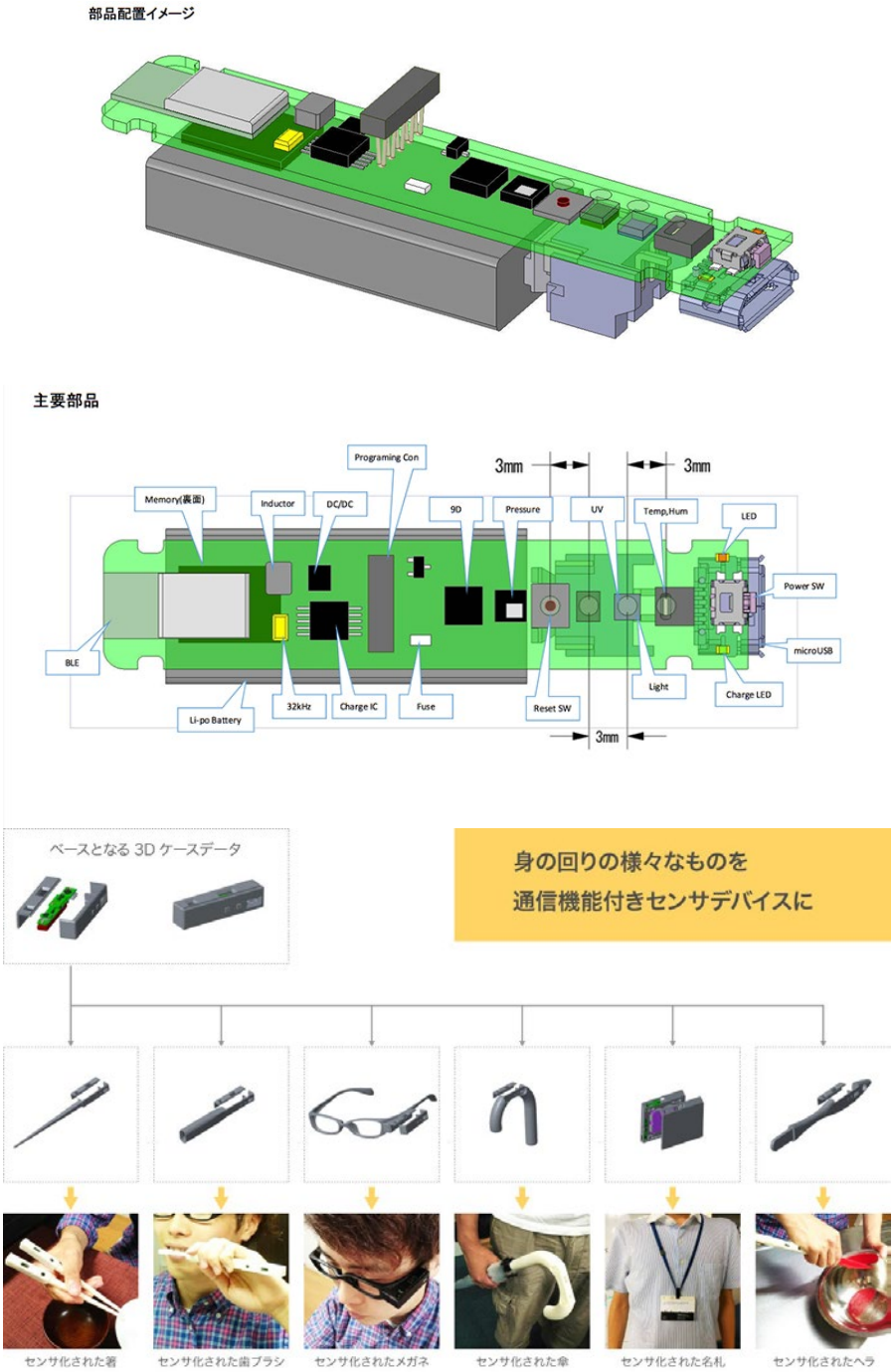
Nordic 社の nRF52 を採用

Cortex-M4F によるデータ処理が可能

周辺ソフトウェア：

iOS/Android 向けのアプリケーション
Node.js 用のライブラリ

オープン性
基板回路図、ファームウェア、3D ケースデータ、周辺ソフトウェア
をすべて公開



SenStickの開発：誰もが簡単にデータ収集

超小型
＋マルチセンサ
＋通信機能
＋記録装置
＋バッテリ

ハードウェア

あらゆるモノをセンサ化
＋
IoT教育

SenStick

マルチプラットフォーム
(iOS/Android/Mac)
＋研究用機能
＋教育用機能（今後）

ソフトウェア

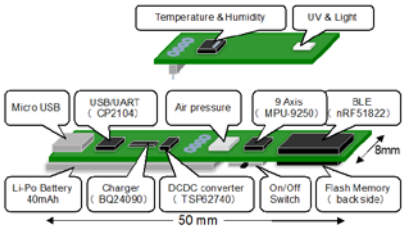
3Dデータ

ケースデータの公開
＋コミュニティ形成
＋データの共有

14

SenStick2: More smaller + More flexible

8 sensors
Acceleration・Gyro・Magnetic・Light・
Temperature・Humidity・Pressure・UV



BLE
Real-time monitoring via Smartphone
Wireless Sync

Stand-alone recording
Flash memory

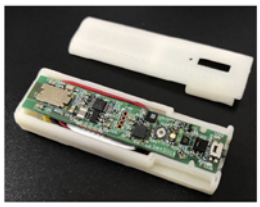
Ultra small size
3g (including a battery)
50mm(W) × 10mm(H) × 5mm(D)

Long battery life
Over **15** hours

17

SenStick3

- SenStick1とSenStick2の経験を踏まえて、いいとこ取りの形
- BLEチップをnRF52へ
  - ちょっとしたデータ処理が可能に
  - センサ間通信やメッシュネットワークの構築も
- I2Cポートを搭載
  - センサを追加できます。
  - バイブレーションを追加すればフィードバックも可能に
- 柔軟なバッテリに対応
  - コネクタ型にしたので、目的に応じて電池を選択可能



18

3.1. 加速度センサについて

加速度センサは文字通り、1 秒における速度変化（加速度）を測定するセンサのことです。測定するものの中には重力加速度も含まれるため、人の動きや振動、衝撃まで検知できます。3 軸方向（X 軸・Y 軸・Z 軸）に適応するセンサであれば水平状態を検出でき、カメラの横方向による「手ぶれ補正」などにも加速度センサの機能が応用されているのです。
加速度センサを用いる家電製品では、スマホやタブレット、ゲーム機のコントローラー、パソコンの HDD などがあります。

パソコンやビデオカメラに用いられる HDD では落下時の衝撃を検知する目的で、内部に加速度センサを搭載しているものがあります。HDD は非常に繊細なパーツのため、振動や衝撃に対してとても敏感。そのため、内部の加速度センサが落下時の重力加速度を検知し、電源を強制的にオフにして内部データを保護するのです。

出展元　https://smartdrivemagazine.jp/technology/accelerometer_gyroscope/#i

● 加速度センサの原理

加速度センサは 3 軸方向の加速度検知を目的とした慣性センサであるため、主に「重力、振動、衝撃」を測定できます。小型加速度センサの多くは MEMS（Micro Electro Mechanical System）の技術により製品化が行われており、極小のものであれば幅が 10mm 以下の商品まであるため、様々な機器に搭載可能です。

加速度センサの一般的な測定方式は、バネと重りが一体化した部品に対して、加速度が発生した時の位置を測定するといったもの。それぞれの方向に取り付けた重りが、外部からの衝撃により移動することで、その位置変化をセンサ内部で測定。後はバネ係数などを考慮した力学の方程式（フックの法則）に数値を入力することで、加速度を導き出すことができるのです。

出展元　https://smartdrivemagazine.jp/technology/accelerometer_gyroscope/#i

人

- 動作状況
- 歩行状態
- 歩数
- ゴルフのスイング

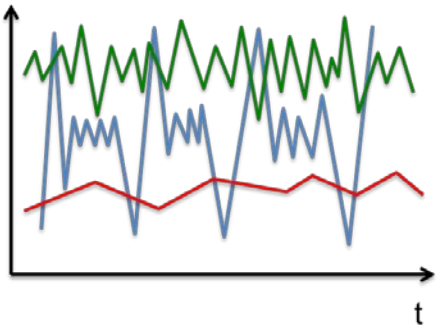
乗り物

- 運転スキル
- 乗り物判別
- 危険判定

環境

- 混雑状況推定
- 道路状態

- 行動分類
  - 停止，歩行，階段上り，階段下り，寝る，座る，立つ，走る，自転車に乗る，掃除をする，食事をする，コンピュータを操作する



信号処理
キャリブレーション，フィルタ

時間窓抽出

特徴量抽出
平均，分散，標準偏差

出展　奈良先端科学技術大学院大学情報科学研究科　ユビキタスコンピューティングシステム研究室

● 加速度＋αによる行動認識

音

加速度センサとマイクを用いて，日常行動を「歩行」「作業」「安静」に分類：95%の精度

マイクを用いて，「作業」をさらに 7 種類に分類：平均 85.9%

GPS（位置情報）

位置による行動の切り分けによく利用される

速度から，乗り物（自転車含む）か歩行かの識別

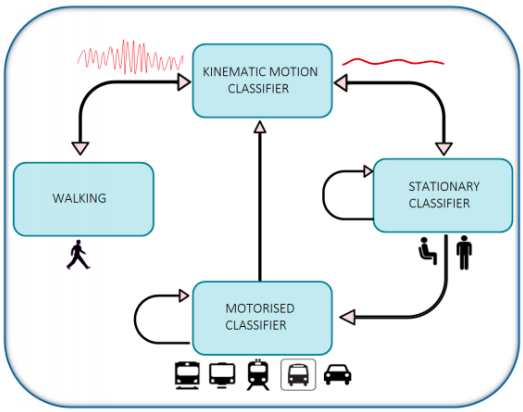
● 加速度センサを用いた乗り物分類

類似した乗り物を分類

バス，電車，地下鉄，トラム

加速度センサだけ

さまざまな特徴量に変換

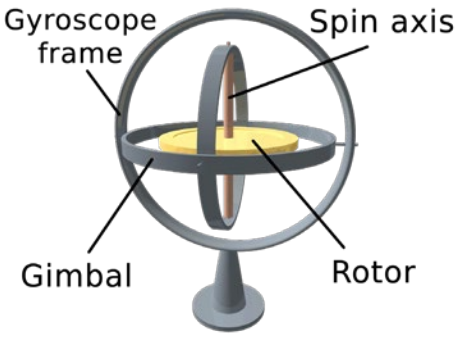


3.2. ジャイロセンサーについて

基準軸に対して 1 秒間に角度が何度変化しているかを検知するセンサのことです。

物体の回転運動を知ることができるため、加速度センサでは検知できない「回転の動き」を測定することが可能。スマホやタブレッドなどで画面を傾けると、自動で見やすい方向に位置が切り替わるのも、ジャイロセンサーの機能が応用されているからなんです。

実はジャイロセンサーはつい最近登場した技術というわけではなく、19 世紀頃から人工衛星や飛行機などに用いられていました。近年ではナノレベルで部品の小型化が進み、先にも述べたスマホやタブレット、カーナビ、腕時計などへ搭載されるようになっています。また、ジャイロセンサーは加速度センサと一緒に搭載されていることが多く、加速度を検出する 3 軸と角速度を検出する 3 軸の合計「6 軸センサ」が主流となっています。



出展元 https://smartdrivemagazine.jp/technology/accelerometer_gyroscope/#i

● ジャイロセンサーの原理

上述しましたが、ジャイロセンサーは回転や向きを検知するセンサのことで、物体の回転速度などを測定できます。細かい動きを検出できる慣性センサの一種として開発が進み、世の中の家電製品やデバイスなどにお馴染みの部品として定着してきました。

有名な活用法としてはカメラの手ぶれ補正やクルマの安全システムなど。また、ジャイロセンサーも加速度センサと同様、MEMS による繊細加工技術が用いられています。

ジャイロセンサーの一般的な測定方式は「コリオリの力（転向力）」の原理を利用して検知されます。コリオリ力とは、移動する質量に回転を加えた場合、その移動する方向に対して垂直方向に働く慣性力のことです。回転を伴う中での慣性力は、他にオイラー力や遠心力がありますが、コリオリ力もその一つとなりジャイロセンサーにおける重要な力学要素となります。

出展元 https://smartdrivemagazine.jp/technology/accelerometer_gyroscope/#i

● 具体的な活用例

いくつか例をあげながら加速度センサ、ジャイロセンサーについて述べてきましたが、もう少し具体的な活用例を紹介していきましょう。

スマホ・タブレット

日本で最も売れているモバイル製品は Apple 社の『iPhone』や『iPad』が有名ですが、こちらの製品には多種多様なセンサやが搭載されているのは周知の話。加速度センサやジャイロセンサーはもちろんのこと、輝度センサ、近接センサ、電子コンパス、GPS などが組み込まれており、まさにテクノロジーの塊です。

iPhone の加速度センサは、主に人の動きを感知することに重点を置いており、端末を動かすことにより画面の位置変更や歩数計のカウントなどが行われたりします。近年では、就寝時に加速度センサやマイクを活用して体の動きを検知することで、睡眠の質を高めるアプリまで登場。たとえば Sleep Meister というアプリでは加速度センサで人の体動を感知し、眠りの浅い（レム睡眠）時にアラームを鳴らしてくれます。

またレースゲームのアプリなどは、画面を傾けることでハンドル操作ができるものもあるため、ここでも加速度センサやジャイロセンサーの技術が活用されています。

出展元 https://smartdrivemagazine.jp/technology/accelerometer_gyroscope/#i

カーナビゲーション・システム

カーナビに搭載されている加速度センサやジャイロセンサーは、主に移動した距離やクルマの移動方向を検知するのに役立っています。カーナビのシステムは GPS やネット回線を通して現在位置を割り出すのが一般的ですが、これらの電波を受け取れないトンネル内などの場所では、搭載している各センサが補助的に機能して位置情報を導き出しているのです。

これらの機能はスマホなどのモバイル製品でも同等の役割を果たすため、カーナビを搭載していないクルマでは、地図検索アプリで代用する方が増えています。

出展元 https://smartdrivemagazine.jp/technology/accelerometer_gyroscope/#i

デジタルカメラ、ビデオカメラ

カメラによる撮影時に、人の動きによる縦横の動きや振動を各センサが検知し、いわゆる「手ぶれ」を抑える役割を果たします。まずジャイロセンサーがカメラの揺れを検知し、その揺れの中から手ぶれ動作を抽出した後、補正したものをレンズの内部機能に反映。これらの動作は高速で内部処理されるため、初心者の方でも自然な撮影を楽しむことができるのです。

出展元 https://smartdrivemagazine.jp/technology/accelerometer_gyroscope/#i

SRS エアバッグ・システム

乗員保護システムとして有名なエアバッグにも加速度センサが活用されています。衝突時による衝撃をセンサが検知した瞬間、内部に搭載されているインフレータに点火して袋状の部品にガスが送られます。この動作はまさに一瞬の出来事なので、センサの役割が人の命を左右すると言っても過言ではありません。

近年、エアバッグの不具合が問題視されましたが、こちらはガスが充填されているインフレータの構成部品が爆発時に飛散したことによる事故のため、センサによる影響ではありません。

エアバッグ・システムは過去に機械式加速度センサが用いられていましたが、半導体技術の発展により「MEMS」が登場し、現在では多くの自動車のエアバッグにMEMS 加速度センサが用いられています。先にも述べた通り、加速度センサは衝撃も検知するため、エアバッグにはなくてはならない部品として活用されています。

出展元 https://smartdrivemagazine.jp/technology/accelerometer_gyroscope/#i

人工衛星

ジャイロセンサーは、人工衛星の姿勢を検出するために採用されています。対して加速度センサは無重力空間のため、搭載されていてもあまり使用されることはありません。隕石などが衝突した際に、トラブルを検知するセンサとして役に立つかもしれませんが、宇宙飛行の状態で重きを置かれているのは、基本的にジャイロセンサーとなります。

出展元 https://smartdrivemagazine.jp/technology/accelerometer_gyroscope/#i

多様な研究分野

技術者の中には、ネットから直接購入して目的の機器に組み込む方もおり、特に人の動きや水平状態を検知するセンサは、ロボット工学の分野で盛んに活用されています。二足歩行によるロボットの開発は非常に難しいため、各センサを駆使して姿勢を維持しながら歩かせねばなりません。

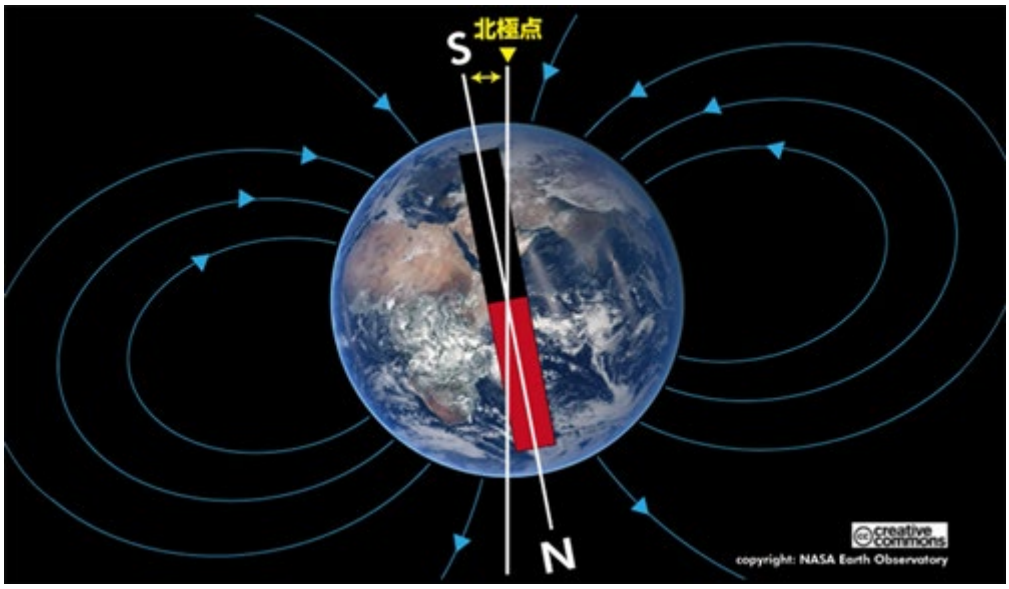
このことから、ロボット工学を専攻する学生や研究者で加速度センサやジャイロセンサーを購入する方が多く、小型のものでも 1,000 円以内で買えるため、手軽に工作できるパーツとして人気があります。
また、加速度センサは地震を検知する役割を担っており、これらの研究分野では計測器に搭載するのが一般的となっています。近年では、スマホに搭載されている加速度センサを使用して地震を計測するアプリが無料配布されており、将来の地震予測や解析に繋がる試みが行われています。

出展元 https://smartdrivemagazine.jp/technology/accelerometer_gyroscope/#i

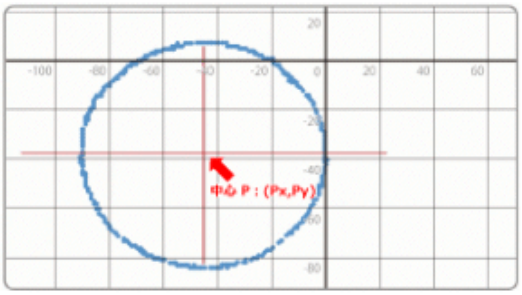
3.3. 地磁気センサについて

地球は磁気が取り巻いており、これを地磁気と呼びます。
地磁気センサはこの地球の磁力を検出するセンサで、別名「電子コンパス」とも呼ばれます。
地磁気センサはこの地磁気を検出することで、方角を知ることができます。

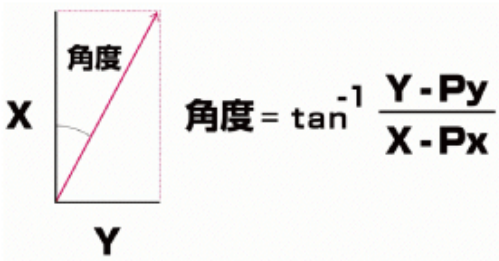
【地球をとりまく地磁気】



地磁気センサには X と Y の 2 軸タイプや Z を加えた 3 軸タイプがあり、その方向の磁力の値を計測します。
簡易コンパスなど傾きを考慮しない場合は、X と Y 軸の値のみを使用します。コンパスを傾けた場合には、地磁気センサの 3 軸の値と加速度センサと組み合わせて正しい方角に補正する必要があります。
下図は地磁気センサを水平に回転させた時の X,Y 値を分布図にしたものです。



地磁気センサ出力分布



磁北からの角度の計算式

出展元 https://www.rohm.co.jp/electronics-basics/sensors/sensor_what2

地磁気センサを水平に回転させると、周囲の磁場の影響がない理想的な場合、出力分布図の円の中心は 0 になります。
しかし実際には周囲磁場の影響で中心がズレるので、円の中心を 0 へ移動させる調整を行う必要があります。
地磁気センサで導き出せる北を磁北と呼び（北極点から少しずれています）、この磁北からの角度を上式で算出することで簡易的に方角を知ることができます。

● 磁気センサのいろいろ

磁気センサは、磁場（磁界）の大きさ・方向を測定することを目的としたセンサです。
目的に応じていろいろなセンサがありますが、以下に代表的なものをあげます。

ホールセンサ

ホール効果を用いて磁束密度を測定するセンサで磁束密度に比例した電圧を出力します。
使い勝手がよく、ドアやノート PC など物の開閉検出など主に非接触スイッチ用途に使用されます。

MR センサ

MR(Magneto Resistance)センサは、磁気抵抗効果センサともいい、個体の電気抵抗が磁界によって変化するのを利用して地場の大きさを測定するセンサです。
ホールセンサに比べ感度が高く、消費電力も小さいことから、使用頻度が多くなっている磁気センサです。電子コンパスなどの地磁気検出用途のほか、モータの回転、位置検出用途などに用いられます。

MI センサ

MI（Magneto Impedance）センサは特殊なアモルファスワイヤを使用し、その磁気インピーダンス効果を応用した次世代磁気センサです。
ホールセンサに対して 10,000 倍以上も感度に優れており、地磁気の微小な変化も高精度に測定が可能になります。
超低消費電流の方位検出（電子コンパス）用途をはじめ、屋内測位、金属異物検出など高感度の特性を活かしたアプリケーションに展開ができます。

出展元 https://www.rohm.co.jp/electronics-basics/sensors/sensor_what2

3.4. 温度センサにつて

温度センサとは、文字通り物や空気の温度を計測することが出来る機械です。
主に、接触式と非接触式の二種類のものがあります。
接触式温度センサには、熱電対、白金測温抵抗体、サーミスタ測温体、バイメタル式温度計、液体充満式温度計および水銀温度計等があり、広く使われています。
対して非接触式温度センサは、物体から発せられる赤外線を計測することで温度を測定します。

● 温度センサの種類

上記でも述べたとおり、温度センサには接触式と非接触式の二種類のものがあります。

接触式温度センサの中でも多く使われているのが気温を測る温度計です。
公的な気象観測に用いられる温度計は、ガラス製温度計・金属製温度計・電気式温度計があります。
ガラス製温度計は皆さんにも馴染み深いものではないでしょうか。
ガラス製の毛細管の中に、水銀や赤く着色したアルコールなどが入っており、
気温変化による体積の変化で温度を測定します。公式な気温の観測などには水銀温度計が使われます。
金属式温度計は温度を測定する感部にバイメタル（2 種類の金属を組み合わせたもの）が使われます。
気温の変化による変形で温度を測定します。構造が簡単で安価なので家庭用に広く普及しています。
電気式温度計は温度を測定する感部に白金を使用しています。
この白金の温度による電気抵抗の変化によって温度を測定します。
代表的な白金測温抵抗体は、Pt100、または、Pt1000　を使用されています。
自動計測、遠隔計測に適するため、気象庁などの多くの機関で使用されています。

接触式温度センサの測定方法を想像するのは難くありません。
なにせ空気や物体と直接触れ合っているからです。
それでは非接触式温度センサはどのように温度を測定しているのでしょうか。
まず熱を持っている物体からは必ず赤外線というものが発せられています。
もちろん温度が上がることにより、比例して赤外線量も増えていきます。
この赤外線量を測定することで物体の温度を測っているのです。
代表的なものはやはりサーモグラフィです。サーモグラフィは物体の表面から発せられる赤外線を測定し、表面温度に色を割り当てて視覚的に分かりやすくしたもの

です。
非接触式なので、動いているものや危険で近づけないものでもリアルタイムで測定できることがメリットとして挙げられます。
また接触式では接触している部分のみでしか測定することが出来ませんが、
非接触式では広い範囲を相対的にリアルタイムで測定できることも大きなメリットとして挙げられます

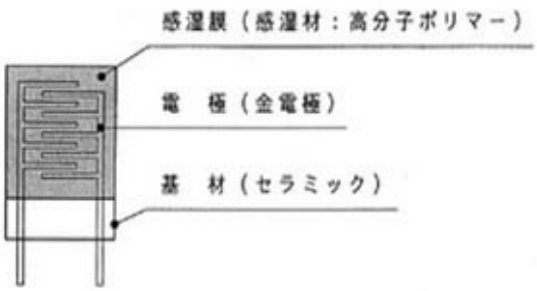
● 温度センサの活用

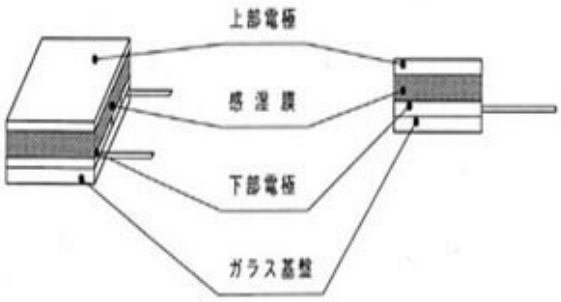
　気温は寒暖の違いとして人々の日々の活動において直接関係し、動植物の成育と関係が深いので、日常生活や各種産業などにおける指標として利用されています。

出展元 https://fieldpro.jp/word/word-temper/

3.5. 湿度センサについて

● センサの作動原理





湿度センサを大きく分けると高分子抵抗式（左）と高分子静電容量式（右）の２種類に分類されます。
現在、世界的に主流なのは高分子静電容量式になっていますが、シェアは少しだけ少ないですが高分子抵抗式にも優れた特性があります。
 共に湿分を吸湿、脱湿する感湿材を使用し、電極間を橋渡ししている構造になっています。
その電気信号を電気抵抗で捉えるのが高分子抵抗式湿度センサ。
その電気信号を電気容量で捉えるのが高分子静電容量式湿度センサなのです。

【電気抵抗式】

・構造が簡単で、大量生産ができ、また、比較的安価である。
・電気抵抗を測定するためにノイズに強い。その為にセンサ部を
小型化する事ができ、センサケーブルも細くすることが可能。
・原理上１０～２０％以下が測定が出来ない。

【静電容量式】

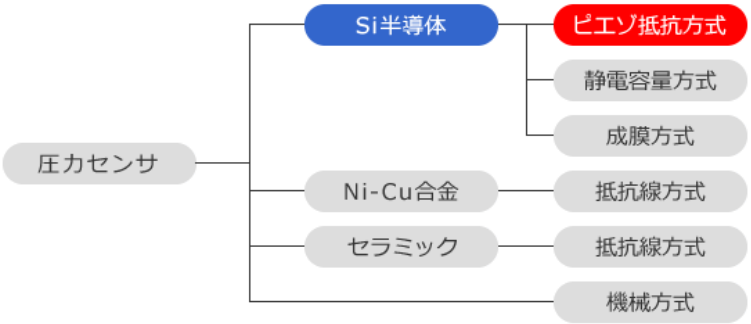
・応答速度が早い。
・高温・低温度域での使用が可能。
・低湿度測定に優れる。
・静電容量を測定するためノイズに弱い。したがってシールド性が必要で
ケーブルなども太い。

出展元 https://www.daiichi-kagaku.co.jp/blog/labo/?p=603

3.6. 気圧

気圧センサとは大気の圧力を検知するセンサです。
圧力センサには、測定する圧力値などにより、下図のような様々な材料や方式のセンサが存在します。
この圧力センサのうち、大気圧を検知（気圧検知を目的）するセンサを一般的に気圧センサと呼んでいます。

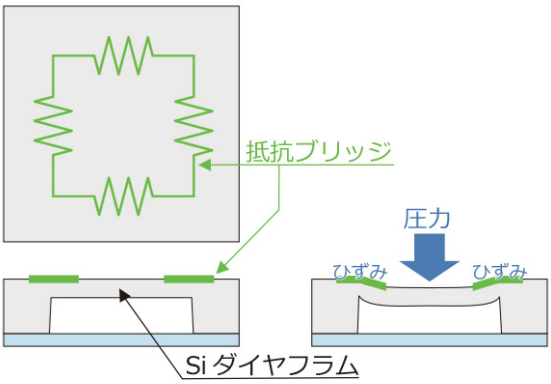
【使用材料-方式から見た圧力センサ】



気圧センサの代表的なものとして、シリコン（Si）半導体を使用したピエゾ抵抗方式のものがあげられます。

● ピエゾ抵抗方式の気圧センサ

ピエゾ抵抗方式の圧力センサは、Si 単結晶板をダイヤフラム（受圧素子）とし、その表面に不純物を拡散させ抵抗ブリッジ回路を形成、圧力が加わった時の抵抗ブリッジのひずみを抵抗値変化として検出し、圧力（気圧）を算出します。



出展元 https://www.rohm.co.jp/electronics-basics/sensors/sensor_what4

3.7. 光センサについて

光センサとは、光を検知・感知するものであり、照射される光が変化することにより、それが電気的信号に変換されるものを指します。

つまり、人間が視覚でとらえる情報を集めるものが、光センサなのです。
人間の目の機能に近いものとして、リニアイメージセンサー、CCD カメラ、赤外線カメラなどが挙げられます。これに対し、望遠鏡や、赤外線センサなどは、人間の視覚機能を拡張したものであるといえます。

● 光の性質と光センサ

光は振動を繰り返しており、波長を持っています。さらに、光は波長域ごとに特色のある性質を持ち、多種多様な用途で用いられます。

特に IoT とはモノ単体であり、仕組みであり、より広い考え方も含めた概念 紫外線、可視光、赤外線の 3 領域で大きく性質が異なります。

どの波長域の光を感知したいかで、センサは異なります。
光センサにとって、検出器（ディテクタ）はとても重要な要素となるでしょう。
すべての波長域の光を高感度にセンシング出来る単一素子の検出器はありません。
そのため、どの波長域の光を用いた IoT の仕組みをつくりたいかによって、選ぶべきセンサは変わります。

● 光センサの例

高度な光センサ

デジタルカメラが挙げられます。これは、光センサを用いるとても身近な例といえるでしょう。
CMOS や CCD といった撮像素子を用いて、光を電気信号に変換しデータを得ます。
さらに、まだ民生用では馴染みがあまりありませんが、産業用の高度なセンシング技術として、物体の形状や圧力の変化などを検知するものもあります。
また、光の「反射」や「透過」等の性質を用いて、光を照射し、その反射光を測定して対象物の特性を分析することも可能です。
特に透過性の高い近赤外線を用いることで、対象物にダメージを与えず、その内部を調べる（非破壊検査）ようなことも可能です。それらは近赤外分光センサとも呼ばれております。

単純な光センサ

光が当たったことを「ゼロかイチか」で判定するだけのものです。

代表的なものとして、照射された光の波長や、波長ごとの強度をデータとして取得できる分光センサなどがあります。

私たちの身近にある例としては「人感センサ」が挙げられるでしょう

自動ドアが開く、公衆トイレの手をかざすと水が流れる、といった私たちの生活に欠かせないものに、こうした光センサが組み込まれています。

これらは、光の遮断によって人が通ることを感知しているのです。

● 光センサの今後

目に見えない光（赤外線、紫外線）に関する高度なセンサは、産業用途で重要なデバイスとして用いられてきましたが、今後はそれが IoT との関係上、ますます活用が進んでいくものと思われます。

これに対し、可視光を検出する光センサは我々の生活に身近なものとなっています。

例えば、オフィスを IoT 化し、ここに光センサを導入するといった活用法が考えられます。このセンサによって、日中にオフィスに入る太陽光の明るさと、オフィスの電気の明るさを測定します。

これらのデータを分析することで、オフィスを適切な明るさに保ち、電気を節約するといったことも可能でしょう。

光センサの IoT への導入は、まだ始まったばかりです。

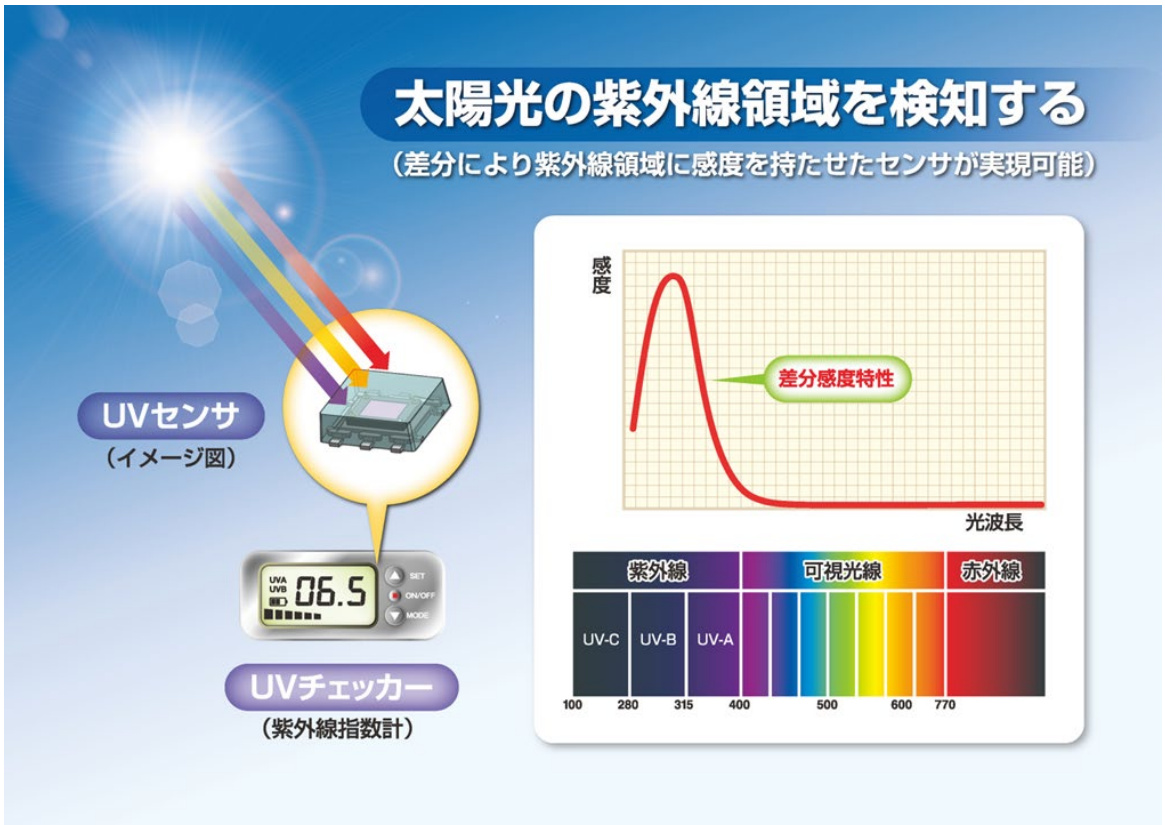
それだけに今後、さまざまな用途での活躍が期待されます。

出展元 https://www.klv.co.jp/iot/iot-optical-sensor.html

3.8. UV センサについて

UV センサは、化粧品の「UV カット」などで有名なように、紫外線（Ultra Violet）を計測する装置だ。化学変化を使ったセンサと異なり、電気信号として紫外線の量を計れるのが特徴。

出展元 https://www.itmedia.co.jp/mobile/0307/09/n_esec.html



出展 https://www.sii.co.jp/jp/news/release/2017/03/27/12987/

4. SenStick をつかった組込みプログラミング

SenStick で、mruby/c プログラムを実行させることにより、さまざまな動作を実現できます。以下の内容について学習します。

- SenStick プログラミングの流れ
- 簡単な SenStick プログラミング

4.1. SenStick プログラミングの流れ

SenStick で mruby/c で記述したプログラムを実行する手順は、次のようになります。第 4 回目からのマイコンボードを使ったプログラミングとほぼ同じ流れなのですが、統合開発環境の PC と SenStick は BLE 通信により接続されます。マイコンボードのプログラミングの流れに、BLE に関する設定が加わることになります。

- PC 上の統合開発環境で mruby/c プログラムを作成する
- 作成したプログラムを SenStick へ転送する。
  この際、BLE デバイスの設定が必要です。
- SenStick でプログラムが実行される

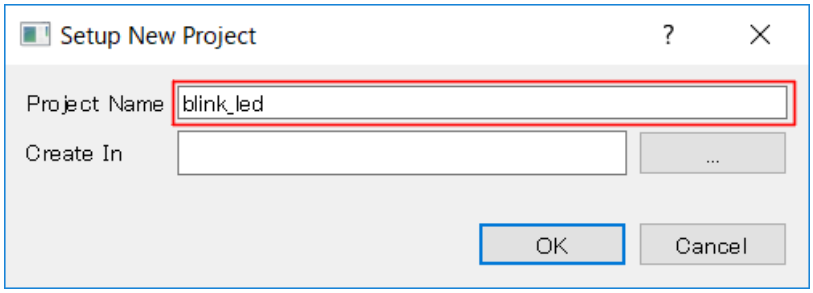
（注）現時点では、BLE を介して SenStick へプログラムを転送する際の設定項目は未確定です。

4.1.1.簡単な SenStick プログラミング

SenStick を使ったプログラム開発の習得のため、SenStick 上の LED を点滅させるプログラムを作成する手順について説明します。
ここでも、第 3 回で紹介した LED を点滅させるプログラムと同じ動作をするプログラムを作成していきます。

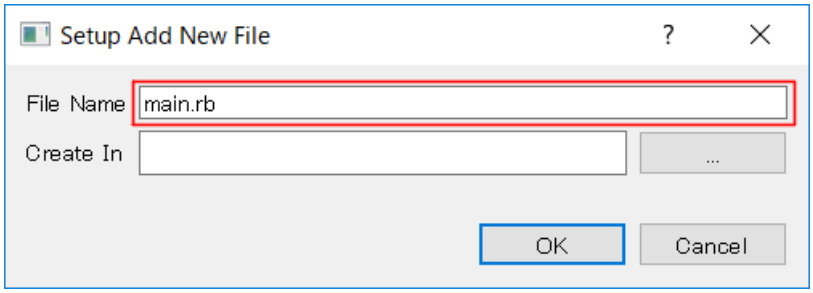
手順 1: プロジェクトの新規作成

開発環境で、[File]-[New]-[Project]メニューを選択し、作成するプロジェクト名を入力します。この例では、blink_led というプロジェクト名を指定しました。



手順 2: プログラムファイルの新規作成

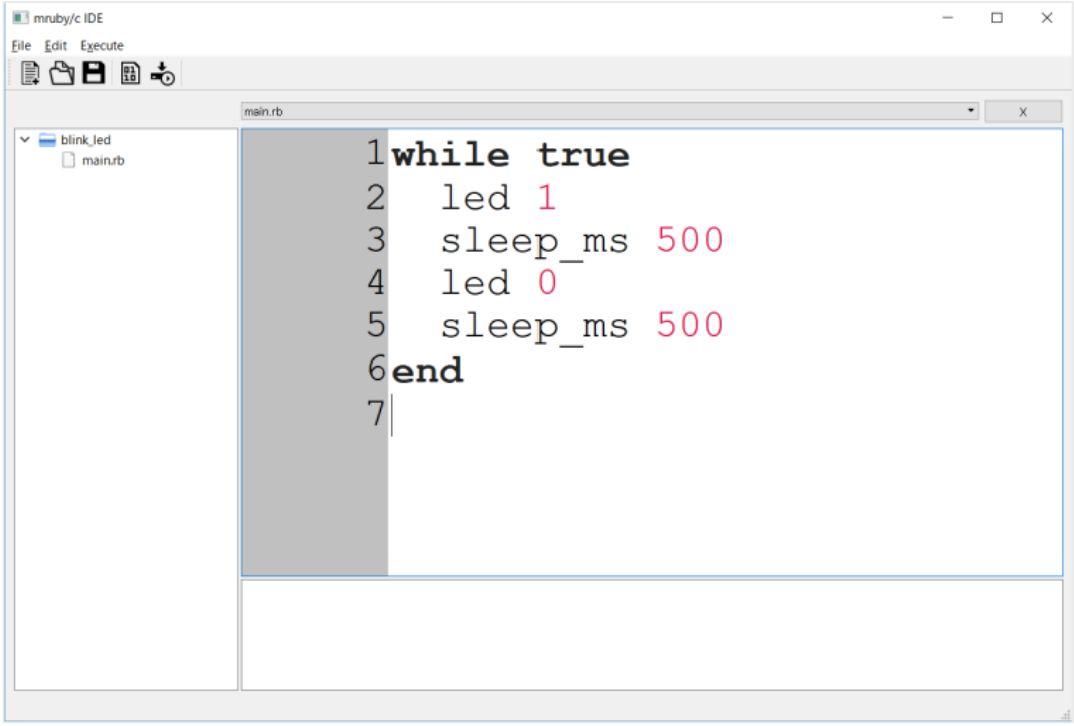
開発環境で、[File]-[New]-[File]メニューを選択し、作成するファイル名を入力します。この例では、`main.rb` というファイル名を指定しました。ファイル名の拡張子は必ず `.rb` とします。



手順 3: プログラムの作成

開発環境のエディタ部分にmruby/cプログラムを入力します。LEDを点滅させるため、次のプログラムを入力します。入力前に、図のようにプログラムファイルが選択されていることを確認します。プログラムファイルをダブルクリックすることで、そのプログラムが選択されます。

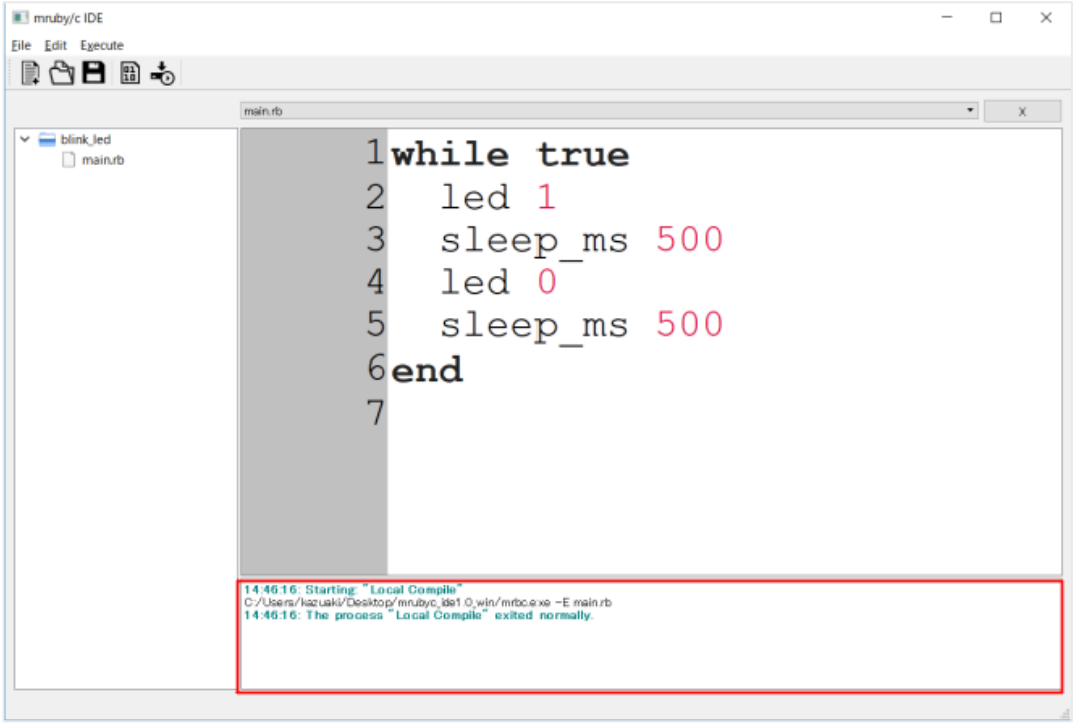
```
while true
  led 1
  sleep_ms 500
  led 0
  sleep_ms 500
end
```



手順 4: コンパイル

[Execute]-[Compile]メニューを選択することで、mrubyプログラムをコンパイルします。

エラーなくコンパイルされた場合は、図のようなメッセージが表示されます。



手順 5: バイトコード転送

[Execute]-[Write]メニューを選択して、コンパイルされたプログラム（バイトコード）をSenStickに転送します。

バイトコードが転送されると、mrubyプログラムが実行されます。SenStickに搭載されているLEDが点滅します。

今回作成したプログラムは、マイコンボードのプログラムと全く同じです。このように、mruby/cでは、プログラムを実行するmruby/c VMがハードウェアの違いを吸収して、異なるハードウェアであっても同じプログラムを動作させることができます。

5. 演習

SenStick の持つセンサを使ったプログラム演習を行います。以下の内容について学習します。

- 温度センサを使ったプログラム

5.1. 基本プログラム

ここで作成するプログラムは、温度センサの値を BLE で送信し、タブレット等で値を確認できるようにします。 SenStcik 内のプログラムは、以下の動作を行います。

- 温度センサからの値の取得
  センサからの値を 1 秒ごとに取得します。
- BLE 通信でデータを送信する
  センサの値を取得したら、直ちに BLE 通信で送信します。

BLE で送信されたデータは、タブレットのアプリで値を確認できます。

5.1.1.基本プログラムの作成と実行

手順 1: プロジェクト作成

開発環境で、[File]-[New]-[Project]メニューを選択し、作成するプロジェクト名を入力します。この例では、temp というプロジェクト名を指定しました。
続いて、[File]-[New]-[File]メニューを選択し、作成するファイル名を入力します。この例では、main.rb というファイル名を指定しました。

手順 2: プログラムの作成

開発環境のエディタ部分に mruby/c プログラムを入力します。最初に説明した動作は、以下のようなプログラムになります。

```
sensor = Temperature.new
ble = BLE.new
while true
  t = sensor.getdata
  b.send t
  sleep_ms 1000
end
```

このプログラムでは、プログラムの先頭で、sensor と ble という変数を作成しています。これらは、それぞれセンサにアクセスするための機能（オブジェクト）、BLE 通信のための機能（オブジェクト）となります。

`Temperature` と `BLE` はクラスと呼ばれ、クラスに対して `.new` という命令を実行することで、それぞれの機能を利用できるようになります。このように、クラスに対して命令を送って、それらの機能を利用するプログラミング手法を、「オブジェクト指向」といいます。
`.getdata` は、センサなどの機能を利用して、その値を取得します。`.send` は、通信などの機能を利用してデータを送信します。このように機能を利用するための命令を共通化しておくことで、異なるセンサや通信などの機能を使うときにも、同じようなプログラムを書くことができます（つまり、プログラムが分かりやすくなります）。

手順 3: 実行

[Execute]-[Write]メニューを選択して、プログラム（バイトコード）をSenStickに転送します。バイトコードが転送されると、プログラムが実行されます。
SneStickのプログラム実行中に、タブレットでSenStickから送信されたデータを確認できます。

5.2. 応用プログラム

すでに、1秒間隔で温度センサの値をBLEに送信するプログラムを作成しました。
次に、このプログラムを少しだけ変更したいと思います。具体的には、次のような動作をするプログラムを作成します。

- 温度センサから、1秒ごとに値を取得する
- 「前回の温度と異なる場合に」BLE通信でデータを送信する

温度センサの取得はそれほど大きな電力を必要としません。一方、BLE通信は大きな電力を必要とします。
1秒ごとにBLE通信を行うと、大きな電力を消費し、結果的に電池で動作する時間が短くなります。
温度はそれほど変化しないため、温度が変化したときのみにデータを送信すると、省電力を実現できます。

5.2.1. 応用プログラムの作成と実行

開発環境のエディタ部分に mruby/c プログラムを入力します。

```
sensor = Temperature.new
ble = BLE.new
t0 = 999
while true
  t = sensor.getdata
  if t!=t0 then
    b.send t
    t0 = t
  end
  sleep_ms 1000
end
```

新たに、変数 t0 を使っています。この変数には、「前回の温度」が格納されます。前回との違いがあれば、BLE 通信でデータを送信します。BLE 通信で送信したデータが、新しい「前回の温度」となるわけです。
なお、最初の「前回の値」は存在しませんが、1 回目に取得した温度データを BLE で必ず送信するように、温度としてあり得ない値（999）を設定しています。
[Execute]-[Write]メニューを選択して、プログラム（バイトコード）を SenStick に転送します。
SneStick のプログラム実行中に、タブレットで SenStick から送信されたデータを確認できます。SenStick を温めるなど、温度を変化させて動作を確認します。

SenStick 参考資料

mruby/c 評価ボード
利用説明書

2018/8/20 rev 1.2
しまねソフト研究開発センター

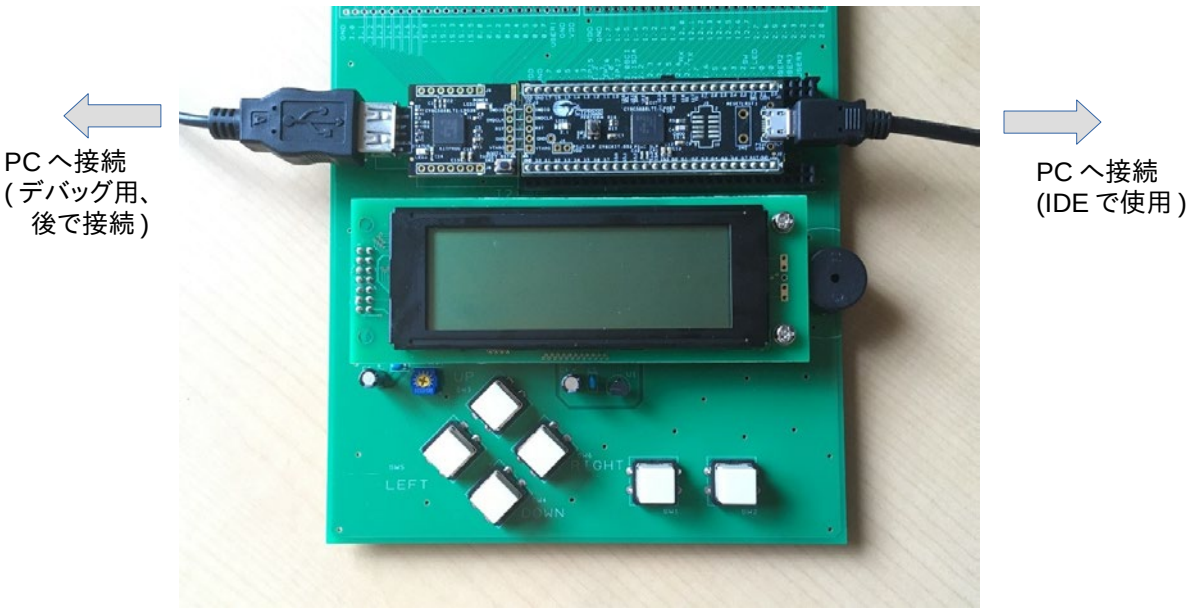
ピンアサイン

ポート名	タイプ	デバイス割り当て	（参考） CY8CKIT-059 上のデバイス PSoC5 内蔵デバイス
P0-0	D-in	SW1	
P0-1	D-in	SW2	
P0-2	D-in	SW3 UP	C12 1u
P0-3	D-in	SW4 DOWN	C13 1u
P0-4	D-in	SW5 LEFT	C9 1u
P0-5	D-in	SW6 RIGHT	
P0-6			
P0-7			
P1-0		(N/C)	PROG_SWDIO
P1-1		(N/C)	PROG_SWDCLK
P1-2			
P1-3		(N/C)	P_SWO
P1-4			P_TDI JTAG[0] tdi
P1-5			JTAG[0] ntrst
P1-6			
P1-7			
P2-0			
P2-1	D-out	--------	LED1
P2-2	D-in	--------	SW1
P2-3			
P2-4			
P2-5			
P2-6			
P2-7			
P3-0	A-in	U1 ThermoSensor TI LM60BIZ	
P3-1			
P3-2			C7 1u
P3-3			
P3-4			
P3-5			
P3-6		(SP1+)	
P3-7		(SP1-)	
P12-0		I2C SCL	I2C[1] scl
P12-1		I2C SDA	I2C[1] sda
P12-2			SIO
P12-3			SIO
P12-4		UART RxD	I2C[0] scl
P12-5		UART TxD	I2C[0] sda
P12-6	D-in	--------	SIO RX (プログラマ経由ホストシリアル)
P12-7	D-out	--------	SIO TX (プログラマ経由ホストシリアル)
P15-0	D-out	LCD DB4 (SC2004CS)	
P15-1	D-out	LCD DB5	
P15-2	D-out	LCD DB6	C42 22p
P15-3	D-out	LCD DB7	C41 22p
P15-4	D-out	LCD E	C4 2200p
P15-5	D-out	LCD RS	
P15-6	USB	--------	USBIO dp (オンボード USB マイクロコネクタ用)
P15-7	USB	--------	USBIO dm (オンボード USB マイクロコネクタ用)

mruby/c IDE の利用

提供された機能のみを使ってアプリケーションを構築する方法です。
統合開発環境、IDE を利用します。
あらかじめファームが書き込まれた本体を使用します。ファームを壊してしまった等の場合は、後述する方法でファームを書き込みます。

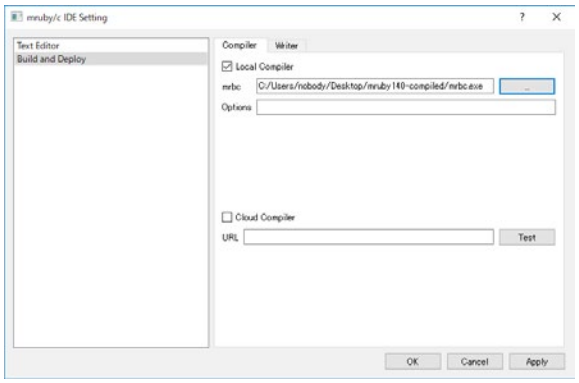
接続

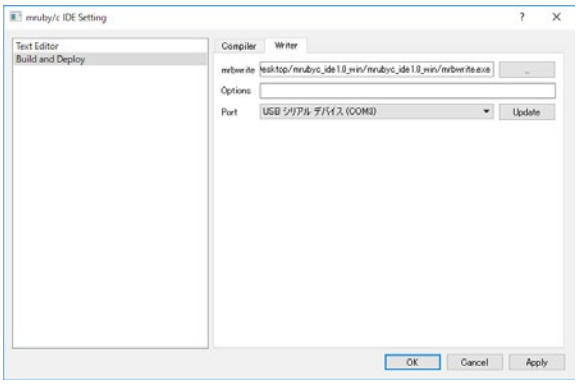


IDE 起動と設定

Windows の例で説明しますが、Mac や UNIX も同等です。

1. mrubyc-ide.exe を実行します。
2. File > Settings を選び、設定ダイアログを表示します。
3. 左ペインの Build and Deploy をクリックします。
4. 右ペインの Compiler タブで、LocalCompiler にチェックが入っていることを確認します
5. mrbc 欄の[...]ボタンをクリックし、mrbc.exe のパスを指定します。
6. 右ペインの Writer タブに移り、同様に mrbwrite.exe のパスを指定します。
7. Port 欄に、基板右サイドのマイクロ USB 端子側の認識ポートを指定します。



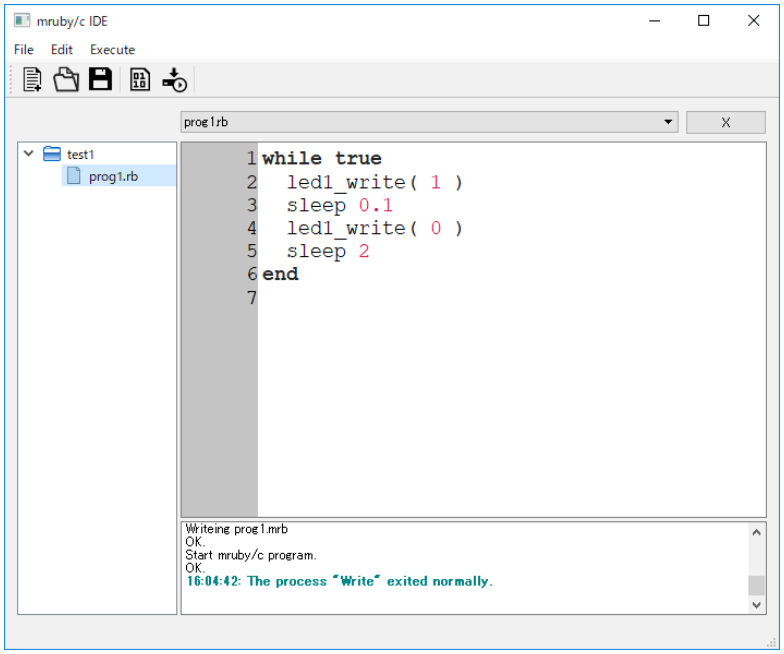


デバグ用コンソールの起動（任意）

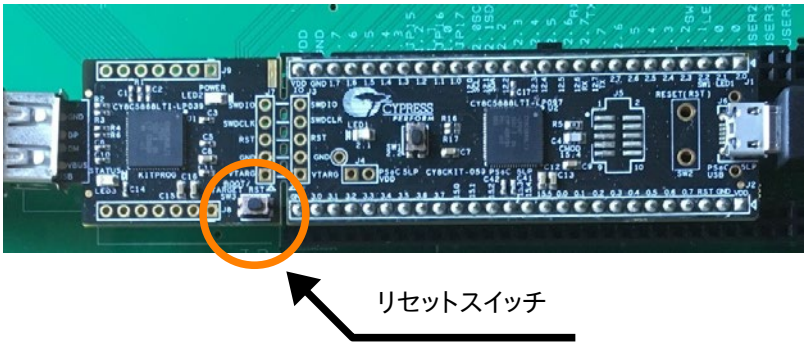
基板左サイドの USB 端子から、デバッグ用メッセージが表示されますので、開発時は接続しておくと便利です。
任意のターミナルソフトを利用して、ボーレート 57600bps で接続します。

プログラムの編集と書き込み

1. File > New > Project を選び、Setup New Project ダイアログを表示します。
2. 任意のプロジェクト名と、保存場所を指定します。
3. File > New > File を選び、Setup Add New File ダイアログを表示します。
4. 任意のファイル名を指定します。
5. 右上のプログラムペインで、Ruby プログラムを編集します。



6. 編集が終わったら、Execute > Write を選びます。
7. プログラムの書き込み待ち状態になるので、基板上のリセットスイッチを押下します。
8. プログラムが書き込まれ、動作開始します。



利用できる機能　mruby/c IDE 編

オンボードスイッチ

マイコンドータ基板上のプッシュスイッチです。SW1 のシルクがあります。
負論理で、0（押された）、1（押されていない）が、返ります。

```
sw1 = sw1_read()
```

LED

マイコンドータ基板上の青色 LED です。LED1 のシルクがあります。
正論理で、1（点灯）、0（消灯）です。

```
led1_write(1)
```

キャラクタ LCD 表示器

4 行×20 文字の LCD パネルです。
アルファベットと数字、記号などが表示できます。詳しくは、HD44780 のデータシート等を参照してください。

```
lcd_location( row, column )
lcd_putc( char_code )
lcd_puts( "String" )
lcd_clear()
```

キーパッド

メイン基板上の 6 個のタクトスイッチです。
ビットマップで全てのスイッチの押下状態を同時に確認できます。
負論理で、ビットは基板上シルクの SWx を参照してください。

```
key = keypad_read()
```

温度センサー（A/D 変換）

メイン基板上に、温度センサー LM60 が搭載されており、温度の測定ができます。
LM60 は、アナログ値を出力しますので、A/D 変換を行います。

A/D コンバータ仕様

- 12 ビット
- 0V - 2.048V
- 100000sps

```
adc_start_convert()                # A/D 変換開始
adc_wait_end_conversion()          # A/D 変換終了待ち
val = adc_get_result16()           # 値読み込み(温度変換前、整数値)
```

PWM

メイン基板上に、圧電スピーカーが搭載されており、PWM を使って音を出すことができます。

PWM 仕様

- カウンタ 16bit
- クロック 120kHz

```
pwm_write_period( n )    # n = 0〜65535
pwm_write_compare( n )
```

I2C

I2C インターフェースを扱うことができます。
参照：https://github.com/mrubyc/dev/tree/master/PSoC5LP

I2C 仕様

- 5V
- 100kbps
- マスタモードのみ
- ピンアサイン 12.0 SCL　12.1 SDA

```
i2c = I2C.new()

# write
i2c.write( i2c_address, device_register, data1, data2, ... )
i2c.write( i2c_address, device_register, "String")

# read
s = i2c.read( i2c_address, device_register, length )
```

UART

シリアルインターフェースを扱うことができます。
PSoC5LP 開発環境の仕様により、ボーレートの変更がソフトウェアでできません。標準で、19200bps 固定です。
変更するためには、サイプレス社の開発環境 PSoC Creator を使って、画面上で書き換える必要があります。
現在の仕様では、読み込み（get, read）時にデータが揃っていない場合、ブロックせずに nil を返します。

UART 仕様

- 5V
- 19200bps
- ピンアサイン　12.5 TxD　12.4 RxD

```
uart = UART.new()

# String read / write
s = uart.gets()
uart.puts("STRING\r\n")

# Binary read / write
s = uart.read(n)    # read n bytes.
uart.write("STRING\r\n")

# Flush buffer
uart.clear_tx_buffer()
uart.clear_rx_buffer()
```

Mutex

複数プログラムの同期用に、Mutex を使う事ができます。

```
$mutex1 = Mutex.new
$mutex1.lock()
$mutex1.unlock()
$mutex1.try_lock()
```

その他のメソッド

mruby/c ランタイムスケジューラには、以下のメソッドを用意しています。

```
sleep( n )        # 実行一時停止。秒単位。1 以下（0.1 等）も可能。
sleep_ms( n )     # 実行一時停止。ミリ秒単位。
relinquish()      # 他のタスクに実行を譲る。
```

ファームの書き込み　改造

ファームウェアがなんらかの原因で壊れてしまったとき、または、UART 数やボーレート変更など、改造が必要となったときに、ファームウェアの書き込みが必要になります。

手順

1. サイプレスのホームページから、PSoC Creator をダウンロードし、インストールします。
2. ITOC のホームページから、ファームウェアをダウンロードし、展開します。
3. ファームウェアを PSoC Creator で開きます。
4. 必要に応じて、回路図等を変更します。
5. メニューから、Build > Build Design 01 を選び、コンパイルを行います。
6. ターゲットボードの左側の USB（カードエッジ側）を PC と接続します。
7. メニューから Debug > Program を選び、書き込みます。

SenStick解説書

～基本的な使い方から機械学習を使ったデータ分析まで～

はじめに 3

SenStickとは 3
SenStickに搭載されたセンサ 3
SenStickの周辺ソフトウェア 3

SenStickの使い方（iOSアプリ版） 4
SenStickの充電と基本操作 4
iOSアプリケーションのインストール 4
iOSアプリの基本的な使い方 4
接続 4
センサの選択 4
センサの設定（取得間隔など） 4
センシング開始とセンシング終了 4
データの吸出し（同期） 4
iTunesを使ったCSVファイルの取得 4

加速度センサを使ったデータ分析の例 4
加速度とは 4
データ収集してみよう：筋トレ（剣道でもOK） 4
データを前処理しよう 4
動作ごとのデータの切り出し 4
アノテーション（ラベル付け） 4
Windowサイズと特徴量計算 4
機械学習をやってみよう 4
Google Colaboratory 4
Python 4
モデル構築（Random Forest, ANNとか） 4
評価（Precision/Recall/F-value） 4
まとめ

1. はじめに

SenStickとは、2014年から著者が開発を開始し、2016年11月に株式会社Matildeより発売された超小型のマルチセンシングボードである。超小型基板の上に、多数のセンサが搭載されており、誰でも手軽にモノのセンシングができるようになっている。

この基板は、著者の想いに応えていただいた複数の企業、そして奈良先端科学技術大学院大学の学生たちによる協力がなければ実現不可能なものであった。いち教員のプロジェクトだったため、開発に2年以上の歳月を要したが、著者自身を含めて、ユビキタスコンピューティング領域の研究者や企業にとって、とても魅力的なプラットフォームが完成したと自負している。

本書では、これまで開発経緯やコンセプトをまとめるとともに、基本的な使い方、そして応用まで、この2年間、学生たちとともに開発してきたSenStickについてまとめる。

2. SenStickとは

SenStickは、わずか数グラムのボード上に、加速度、地磁気、ジャイロ、温度、湿度、気圧、照度、UVという8つの代表的なセンサが搭載されている。メインチップとしては、Nordic社のnRF52を搭載しており、BLEによるデータ通信とcortex-M4Fによるデータ処理が可能となっている。最も特徴的な部分としては、32MBのフラッシュメモリを搭載している点である。通常、BLE機器はスマートフォンに接続して使うことを想定しており、内部記憶容量が少ない。1時間に1回気温を記録する用途であれば、容量の小ささは問題とならないが、行動認識のために加速度を50Hzで計測するような用途の場合、これが問題となる。また、BLEは通信速度が遅いため、8種類のセンサデータを50Hzでスマートフォンに送ることは困難である。これらの問題を解決するために、SenStickはBLE機器でありながら、大きなメモリを搭載している。その結果、8種類のセンサを50Hzで計測してもすべてのデータを記録することが可能である。

2.1 SenStickに搭載されたセンサ

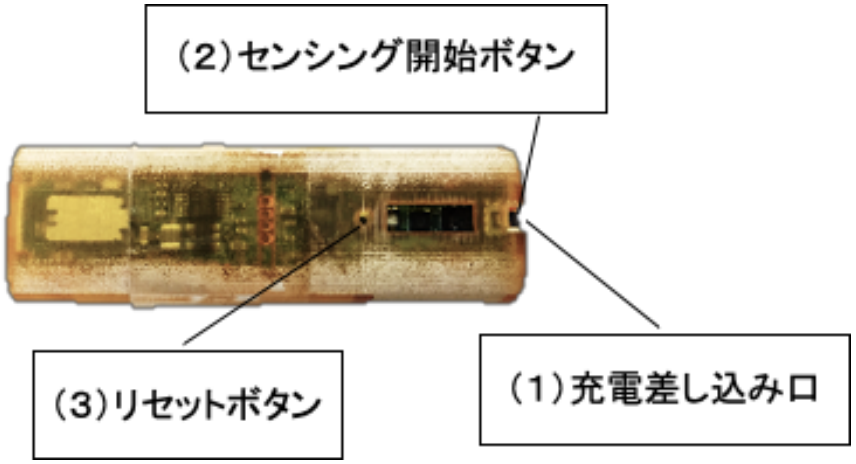
2.2 SenStickの周辺ソフトウェア

3. SenStickの使い方（iOSアプリ版）

ここでは、iOSアプリの利用を前提としたSenStickの使い方を解説する。

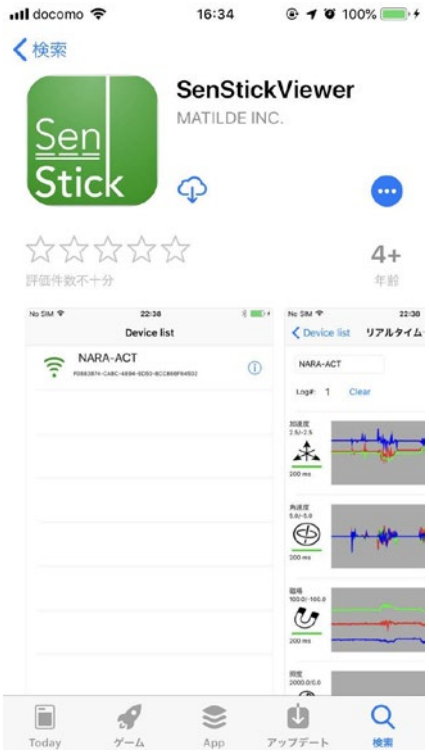
3.1 SenStickの充電と基本操作

１：充電差し込み口（充電はMicro-Bケーブルで行う）
２：センシング開始ボタン
３：リセットボタン

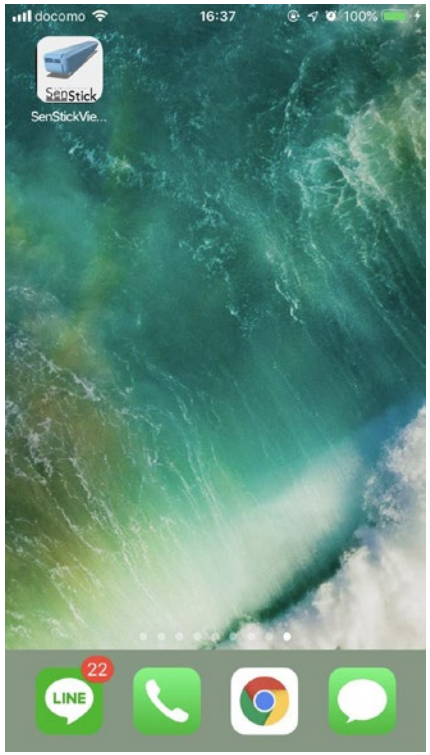


3.2 iOSアプリケーションのインストール

3.2.1 : App storeからインストール方法：



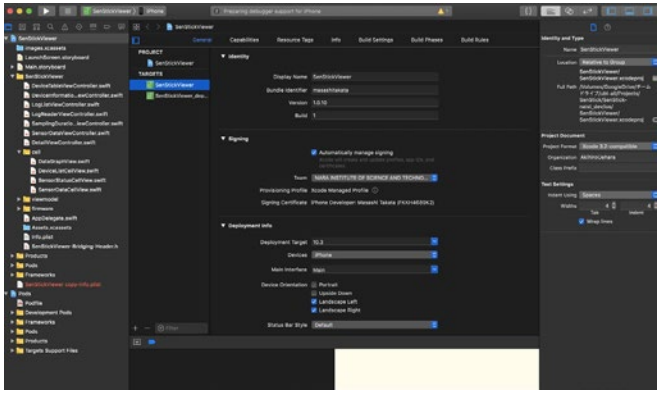
1：App storeで「SenStickViewer」で検索



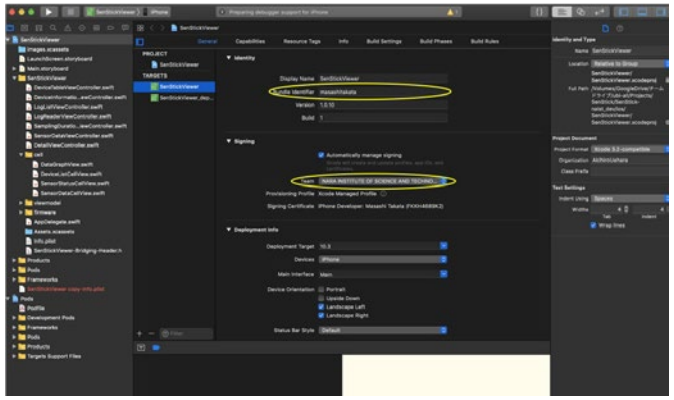
２：インストール完了

3.2.1 ソースコードからインストール方法

1：SenStick-masterをチームドライブから自分のPCにダウンロードする
（保存先：チームドライブ/ubi-all/Projects/SenStick/SenStick-master）
2：ターミナルでダウンロードした場所まで移動する
$ cd SenStick-master/ios
3：podをインストールする
$ pod install
4：Xcodeでインストールする

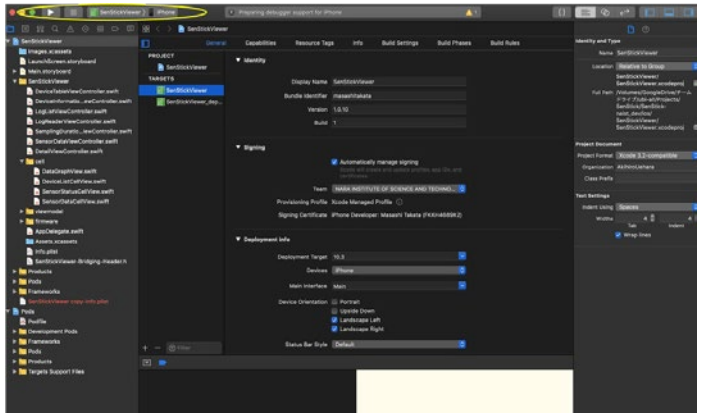


１：SenStickViewer.xcworkspaceをXcodeで開く

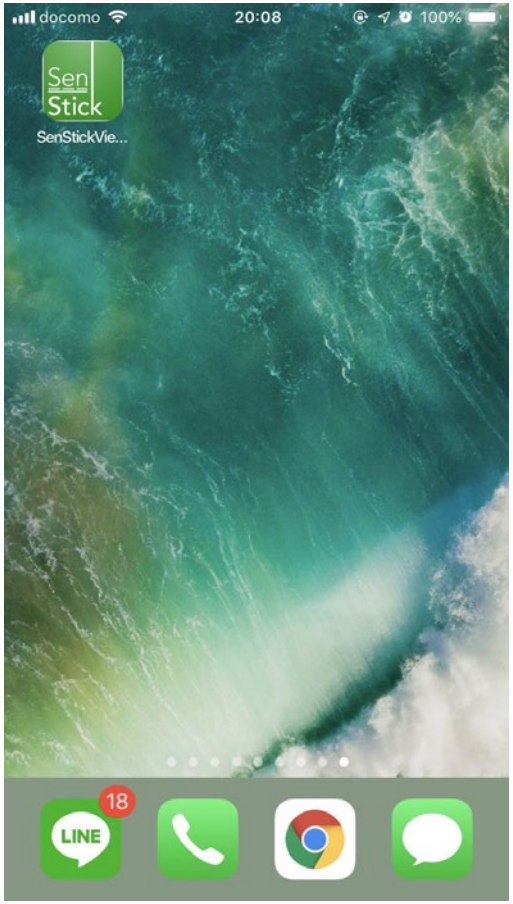


２：Bundle IdentifierとTeamを各自の設定に変更する

3.2.1 ソースコードからインストール方法



３：MacとiPhoneを繋ぎ，ダウンロード先を自分のiPhoneに設定しRunボタン（▶）を押す

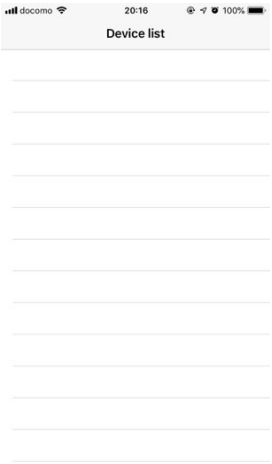


４：インストール完了

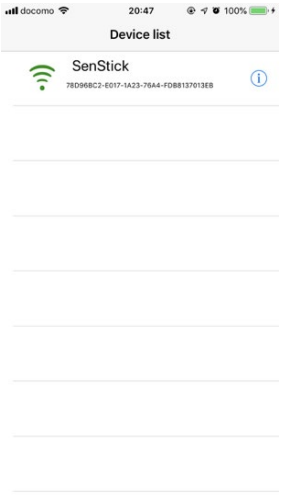
3.3 iOSアプリの基本的な使い方

3.3.1：接続

スマホでインストールしたアプリを立ち上げる
SenStickとスマホがBLE通信可能な距離であれば，下スワイプをするとスクショのように通信可能なSenStickが表示される（スマホのBLE通信はONにしておく）



１：アプリ立ち上げ画面



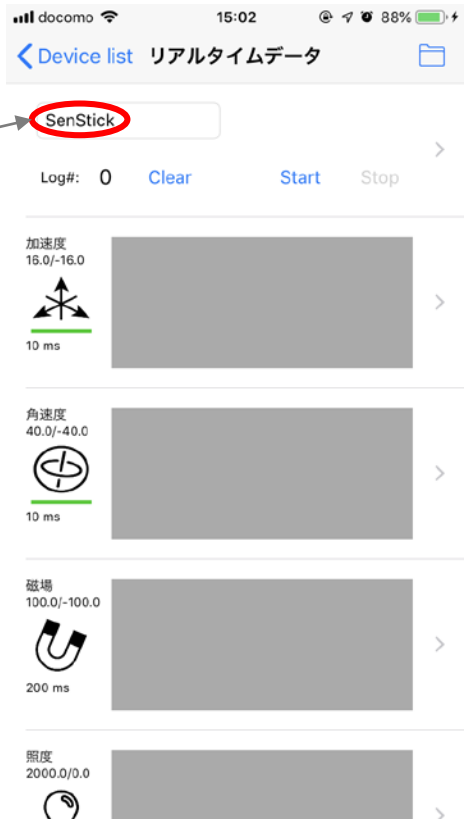
２：通信可能なSenStickが表示

３：接続後の画面

3.3.2 センサの選択

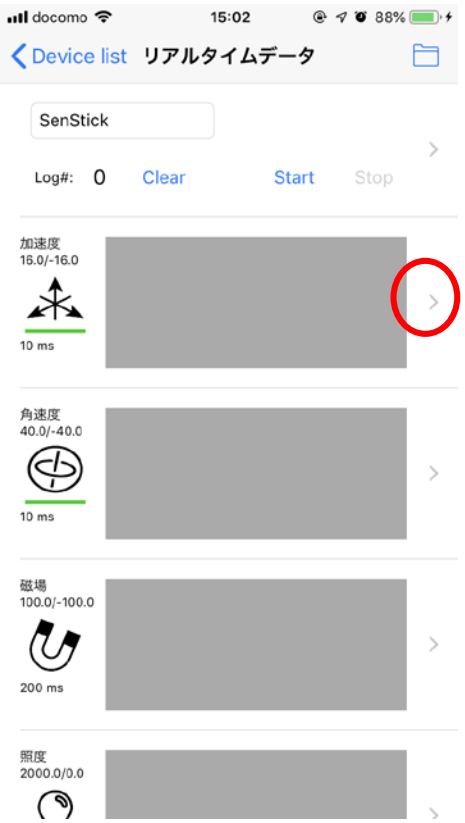
・センサの名前は変更可能

・取得したいセンサのアイコンをタップして選択



４：加速度，角速度を選択後の画面

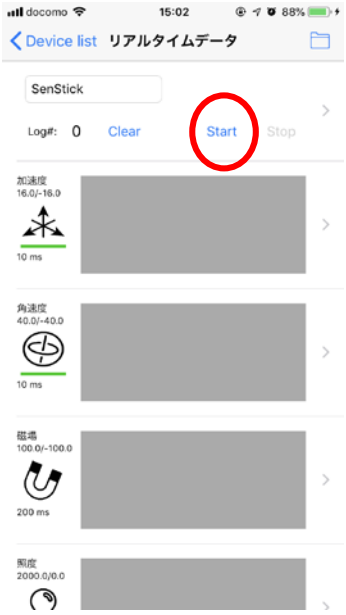
3.3.3 センサの設定(取得間隔など)



１：矢印ボタンを押す

２：設定画面

3.3.4 センシング開始とセンシング終了



1 ：Startボタンを押す

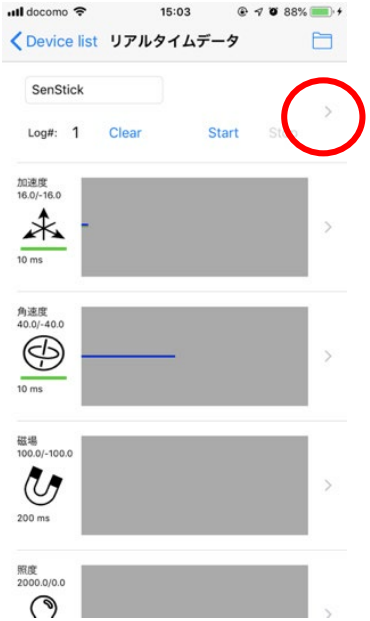
2 ：センシング開始

3 ：Stopボタンを押す

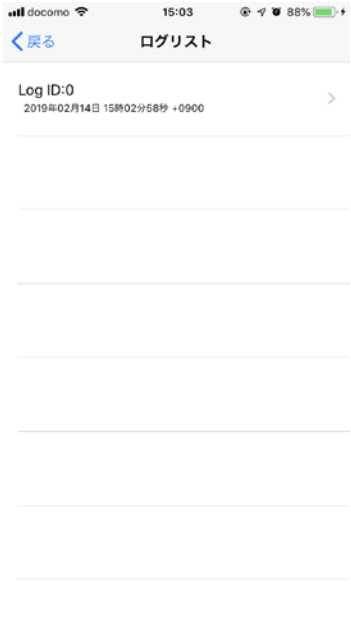
4 ：センシング終了

3.3.5 データの吸い出し(同期)

１：Logが増える

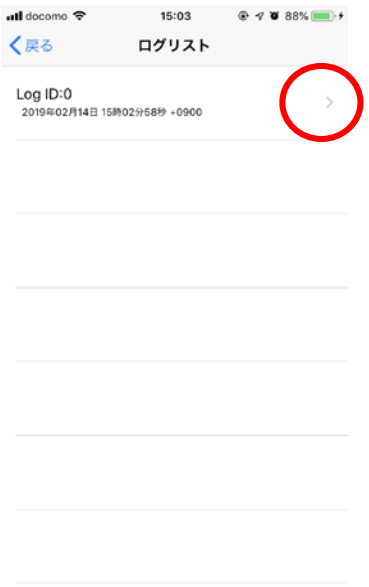


２：矢印ボタンを押す

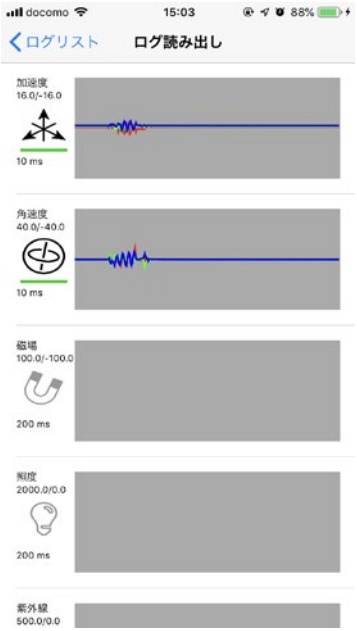


３：ログリスト画面

3.3.5 データの吸い出し(同期)



4：矢印ボタンを押す



5：データ吸い出し

6：Clearボタンを押して，
SenStickからLogを削除

一連のデータ収集操作の動画

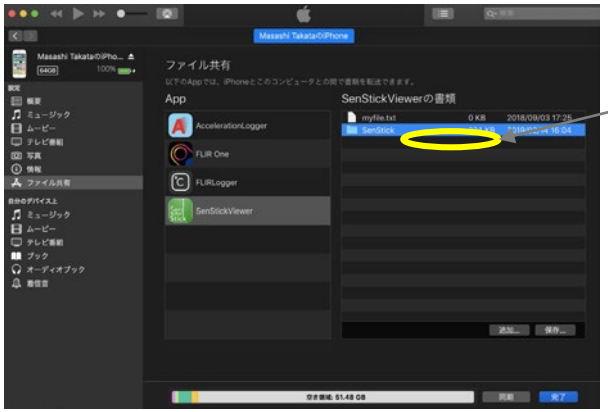
3.3.6 iTunesを使ったCSVファイルの取得

1 ：iTunesを立ち上げる

2 ：iPhoneアイコンを選択

3 ：ファイル共有を選択

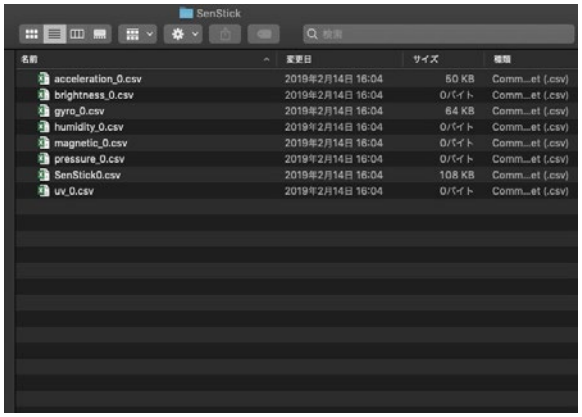
4 ：SenStickViewerを選択



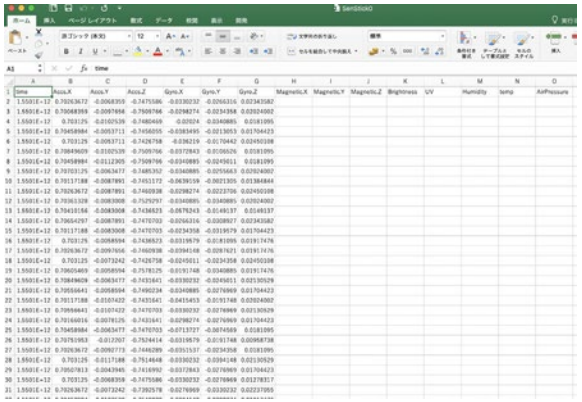
フォルダの名前
は各自決めた
SenStickの名前

5：取り込むセンサデータフォルダを選択し，
ドラックアンドドロップで自分のPCに移動させる

3.3.6 iTunesを使ったCSVファイルの取得



フォルダの中身



csvファイルの中身

ファイル形式
・センサの種類_ログ番号.csv
・例：acceleration_0.csvは,加速度の値が入ったログ0のcsvファイル

・SenStick0.csvは全てのセンサ値が格納されたcsvファイル
SenStickは各自決めた名前なので，アプリ上で変更するとこのファイル名も変わる.

SenStick0.csvの中身
A列にunixtime，B列にX軸の加速度，C列にY軸の加速度，，，
と格納されている.

4 加速度センサを使ったデータ分析の例

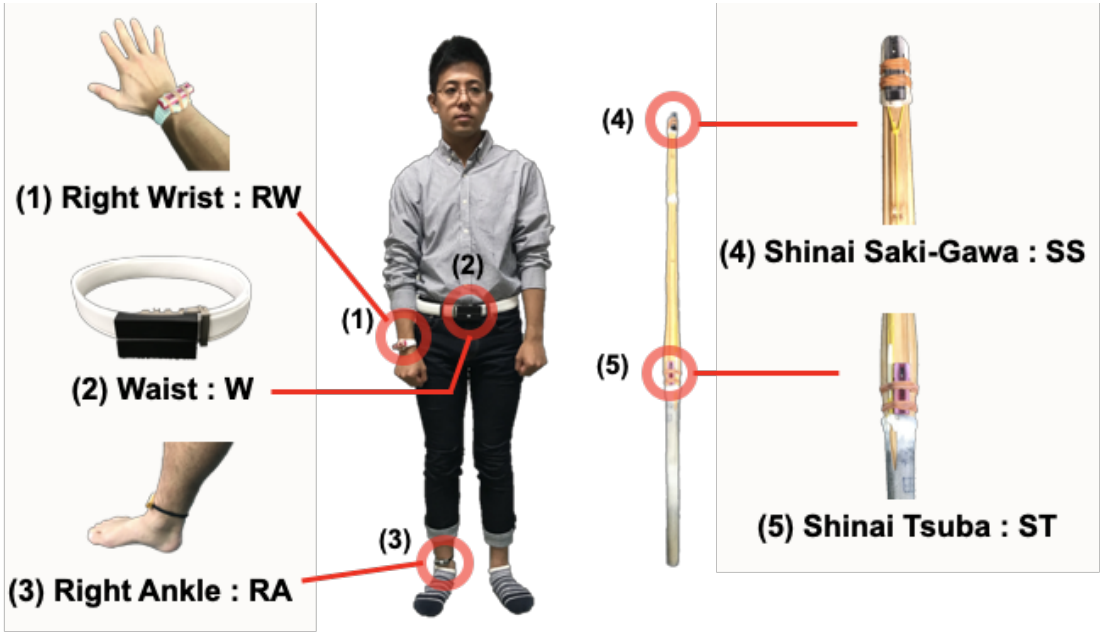
4.1 加速度とは

加速度とは何でしょう？速度は，位置の変化率，すなわち単位時間あたりにどの程度位置（変位）が変化しているか観察した量が速度です．当たり前ですが位置がどんどんずれると速度を持つことになり，位置が変わらないと速度がゼロです．感覚的にもこの物理量はわかりやすいと思います．加速度は，さらにこの速度の単位時間あたりの変化率を指します．同じように速度が変化すると（増加したり，減少したりすると），加速度が生じます． 速度の単位は単位時間あたりの位置の変化量ですから[m/s]です．ですから，加速度の単位はさらにもう一度時間で割るので[m/s^2]です．しかし，これは一般的な科学で用いられる SI単位で，加速度センサは通常重力加速度[g]を単位としています．つまり，地球が物体を引張る引力を基準にした加速度で，1[g]はおおよそ9.8[m/s^2]です．

引用先：https://www.sports-sensing.com/support/motionmesurement/motion_biomechanics/basis_accsensor.html#prettyPhoto

4.2 データ収集してみよう

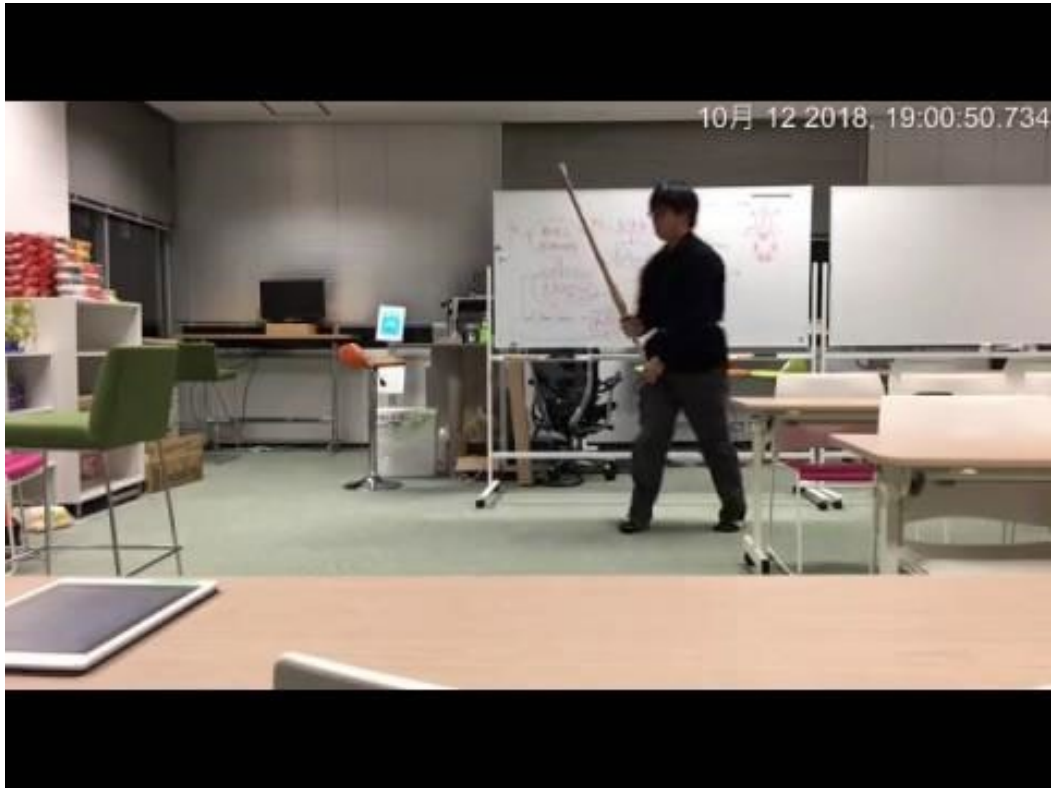
剣道競技における，面，突き，胴，小手を行なった際の加速度とジャイロ（角速度）をセンシングしてみた



装着箇所

4.2 データ収集してみよう

実験中の動画



- 動作前に一定時間，静止状態をとってから行動開始することで波形を確認したときにわかりやすい
- 実験中の動画は録画しておく(時刻がわかるように)
- 録画に便利なiosアプリ:https://itunes.apple.com/us/app/timestamp-camera-basic/id840110184?mt=8)

TimestampCamera(無料)

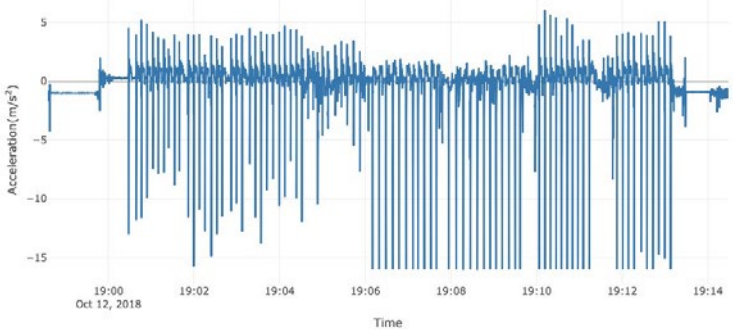
4.3 データ前処理をしてみよう

4.3.1 動作ごとのデータの切り出し

- 基本は収集したcsvファイルから録画したビデオの時刻を元に動作を切り出していく.
- ただし，長時間のデータ収集やSenStickの電池残量が少ないとロギングにズレが生じる可能性があるため，その動作前後の波形を見て確認する.
- 波形表示は，Google Colaboratory（ローカルにiPythonとかをいれなくても同様のことができるプラットフォーム）を用いる.

Google Colaboratoryのリンク
https://colab.research.google.com/drive/1bAVyXWmjeeQIPxT6AjDBs7pKpxDZIuVz

(対応箇所：ファイル読み込み～波形表示)



波形を表示の出力結果

4.3.2 アノテーション(ラベルづけ)

1. 録画したビデオを元に，各動作の開始時刻と終了時刻を確認して記録する.
2. 記録した時刻付近の波形を見て正しいか再度確認する.

   ※長時間のデータ収集やSenStickの電池残量が少ないとロギングにズレが生じる可能性があるため

   （対応箇所：ファイルを読み込み～波形を表示まで）

1. 波形で対象行動を確認できたら，その開始時刻におけるindexをgsheetに記録していく.

   例：https://docs.google.com/spreadsheets/d/1ca02nNHJAdn7LAhm3miAYvP6pUTpROBvyfvh52KBaiU/edit?usp=sharing

   (対応箇所：時刻記録）

1. 記録したgsheetを元に，アノテーションしていく

   (対応箇所：アノテーション）

	0	1	2	3	4	5	6	7	8	9	...	71	72	73	74	75	76	77	78	79	80
0	Label_Trial	1_1	1_2	1_3	1_4	1_5	1_6	1_7	1_8	1_9	...	8_1	8_2	8_3	8_4	8_5	8_6	8_7	8_8	8_9	8_10
1	Right_Wrist	103	114	121	129	137	145	153	160	168	...	794	802	810	818	828	836	845	853	862	870
2	Waist	102	112	119	127	135	143	151	157	166	...	784	791	800	807	818	826	835	842	851	859
3	Right_Ankle	101	112	119	127	135	143	151	157	166	...	785	793	801	809	819	827	836	844	852	860
4	Shinai_Mochite	102	112	119	127	135	143	151	157	166	...	786	794	802	810	820	828	837	845	853	861
5	Shinai_Sentan	101	111	118	126	134	142	150	156	165	...	786	794	802	810	820	828	836	844	853	861

6 rows × 81 columns

indexを記録したgsheetを読み込みの出力結果

4.3.3 動作ごとのデータの切り出し

- 装着位置＋ラベルごとにデータを切り出す

（対応箇所：動作ごとのデータの切り出し）

4.3.4 Windowサイズと特徴量計算

- Windowサイズは3秒に設定
- 特徴量は「最大値, 最小値, 平均, 標準偏差, 合計値, 範囲(最大値と最小値の差), 分散, 平均絶対偏差, 四分位範囲, 尖度, 二乗平均平方根」を抽出

（対応箇所：Windowサイズと特徴量計算）

4.4 機械学習をやってみよう

4.4.1 モデル構築

- モデル構築には「sklearn」を使用
- アルゴリズムは「Support Vector Machine」．「Random Forest」，「K-Nearest Neighbor」，「Artificial Neural Network」，「Decision Tree」，「Extremely Randomized Trees」，「Logistic Regression」，「Naive Bayes」の計８つを使用

（対応箇所：機械学習をやってみよう）

4.4.2 評価

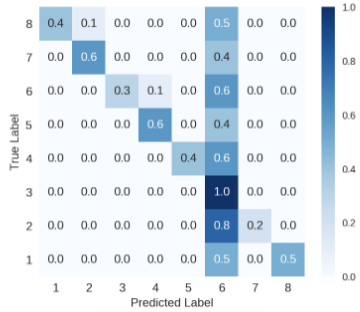
- 今回の評価には試行間での交差検定で求める

  （例：１～９セットを訓練データ，１０セット目をテストデータにする）
- 混同行列による結果の可視化を行う．

[ ]

SVMで学習した結果

	recall	precision	f1-score	support
1	0.4	1.000000	0.571429	10
2	0.6	0.857143	0.705882	10
3	0.3	1.000000	0.461538	10
4	0.6	0.857143	0.705882	10
5	0.4	1.000000	0.571429	10
6	1.0	0.208333	0.344828	10
7	0.2	1.000000	0.333333	10
8	0.5	1.000000	0.666667	10
micro avg	0.5	0.500000	0.500000	80
macro avg	0.5	0.865327	0.545123	80
weighted avg	0.5	0.865327	0.545123	80

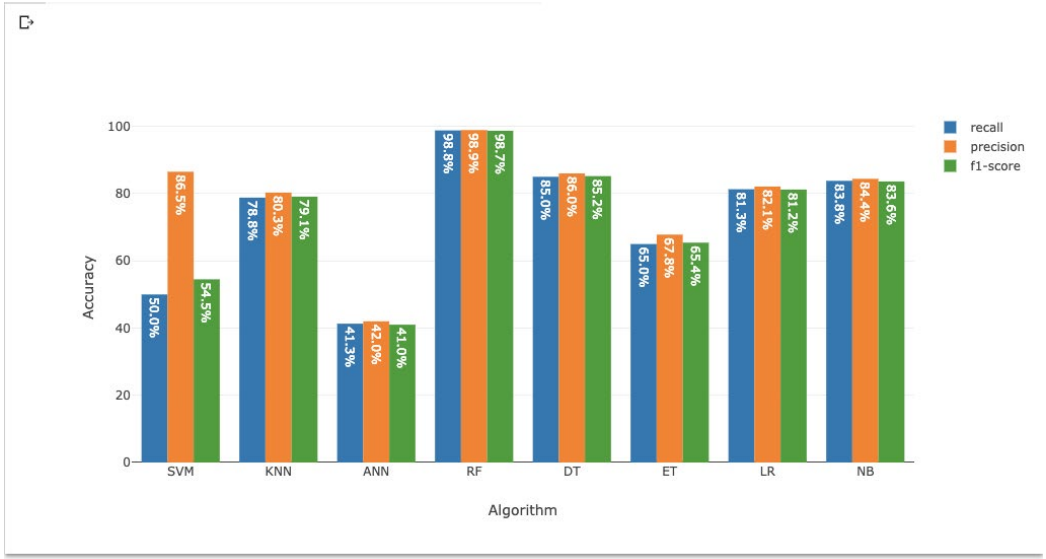


SVMで学習の出力結果

4.5 結果の可視化

- 各アルゴリズムで算出した，recall，precision，f1-scoreの結果をそれぞれ可視化する．

（対応箇所：結果の可視化）



結果を可視化させるの出力結果

Tips（やってしまいがちなミスと対処）

- SenStickに関すること
  - センサ名称の変更方法
    - 複数使用する際は，同名称は避ける
    - 日本語は避ける(データを読み込む際にバグが発生する可能性があるため)
    - 自分が把握しやすい名前にす(どこに装着されていたものかなど)
  - 接続がうまくいかない場合
    - SenStickの充電を確認する
    - 直接SenStickのセンシング開始ボタンを押してみて，再度繋がるか確認する
    - リセットボタンを3秒間くらい押してみて，再度繋がるか確認する
    - アプリを再インストールしてみる

- 機械学習に関すること
  - 性能がやたらといいときに疑う（自分のデータを混ぜてしまっている？）
    - 訓練データに何が用いられているか，テストデータも同様に確認する
    - modelを再度設定する
  - 特徴量が多すぎるときのTips
    - 主成分分析で次元を削減してみる
    - 特徴量ランキングの結果から，上位のものだけをしようしてみる
    - Grid Searchをしてみる

プログラミング演習参考資料

IoT プログラミング演習の資料

演習で使用するプログラム言語 Ruby の簡単な説明、演習の際に参考になる機能やプログラム例です。

mruby 文法

多くの場合、mruby のプログラムは以下のように「while true～end」を使って書く。

```
一度だけ実行する処理を書く
while true
  繰り返し実行する処理を書く
end
```

簡単なプログラムの例。led と sleep_ms の機能を使っている。「while true ～ end」の間を繰り返し実行する。処理は 1 行ずつ順に実行される。
このプログラムでは、led 1 で LED を点灯し、500 ミリ秒（0.5 秒）処理を止め、led 0 で LED を消灯、500 ミリ秒の処理を止める動作を繰り返す。0.5 秒間隔で LED が点滅する。

```
while true
  led 1
  sleep_ms 500
  led 0
  sleep_ms 500
end
```

変数が使える。変数は英数字で、値を代入することで変数として利用できる。次の例は、on と off の変数にそれぞれ 1 と 0 を代入している。この変数 on と off の値を、プログラム中で使っている。
変数への代入は、プログラムを実行する最初に一度だけ実行すればよいので、プログラムの先頭で行っている。

```
on = 1
off = 0
while true
  led on
  sleep_ms 500
  led off
  sleep_ms 500
end
```

if により、条件を判定して、処理を変えることができる。「if 条件 then 処理 end」と書く。
次のプログラムは、次第に点滅の速度が速くなる動作をする。
最初、t は 1000 であるが、点滅ごとに 100 減らしていく。sleep_ms の時間として使うので、0 以下の値になった場合は、1000 に戻している。

```
t = 1000
while true
  led 1
  sleep_ms t
  led 0
  sleep_ms t
  t = t - 100
  if t<=0 then
    t = 1000
  end
end
```

mruby が持つ機能

処理名	内容	使い方の例
sleep_ms <t>	一定時間プログラムを停止する <t>に時間をミリ秒単位で指定する	sleep_ms 1000
led <value>	基板上の青色 LED を制御する <value>が 0: LED が OFF <value>が 1: LED が OFF	led 1

sw()	基板上のスイッチの状態を得る
	スイッチが押されていると 1
	スイッチが押されていないと 0
		if sw()==1 then
		  led 1
		else
		  led 0
		end

output <pin>,<value>	ピンの電圧を制御する <pin>はピン番号で 0~7 を指定し、それぞれ P3.0～P3.7 を意味する <value>が 0: 低い電圧 <value>が 1: 高い電圧	output 3,1
tone <freq>	P15.0 に音の信号を出す <freq>に周波数を指定する。0 を指定すると音の信号が止まる	tone 440

pwm <pin>, <duty>	PWM 信号を出力する <pin>はピン番号で 0～3 を指定し、それぞれ P0.0～P0.3 を意味する <duty>はデューティ比で、0~255 を指定する	pwm 0, 128

プログラム例

ボード上の青色 LED を点滅させる。

```
while true
  led 1
  sleep_ms 500
  led 0
  sleep_ms 500
end
```

スイッチの入力に対して、ボード上の青色 LED を点灯させる。

```
while true
  if sw()==1 then
    led 1
  else
    led 0
  end
  sleep_ms 10
end
```

PWM により LED の明るさを変化させる。LED は P0.0 に取り付ける。

```
while true
  pwm 0, 0
  sleep_ms 500
  pwm 0, 64
  sleep_ms 500
  pwm 0, 128
  sleep_ms 500
  pwm 0, 255
  sleep_ms 500
end
```

３つの LED を順に点灯させる。LED は P3.0～P3.2 に取り付ける。

```
i = 0
while true
  output i, 0
  i = i + 1
  if i>=3 then
    i = 0
  end
  output i, 1
  sleep_ms 500
end
```

音（ラ、A4）を鳴らす。圧電スピーカを P15.0 に取り付ける。

```
while true
  tone 440
  sleep_ms 1000
  tone 0
  sleep_ms 1000
end
```

参考情報

今回使用するのは、米国 Cypress 社の PSoC5LP というマイクロプロセッサです。

CPU	ARM Cortex-M3, 63MHz
ROM	256KB
RAM	64KB
I/O	60 GPIO
Digital Unit	PWM, I2C, CAN
Analog Unit	ADC, DAC, OpAmp
消費電力	15mA（最大）、2μA（スリープ） 300nA（停止）

IoTプログラミング
演習

IoTプログラミング演習

- プログラムで電気回路を制御する

- 演習の内容
  - 電気回路の説明
  - プログラム作成と実行

電気回路の説明

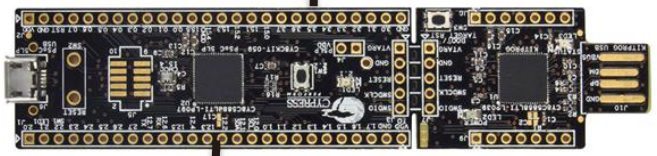
構成

統合開発環境（IDE）で
プログラムを開発する

USBケーブル

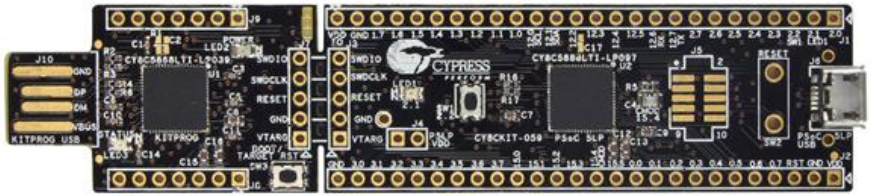
ブレッドボード

マイコン

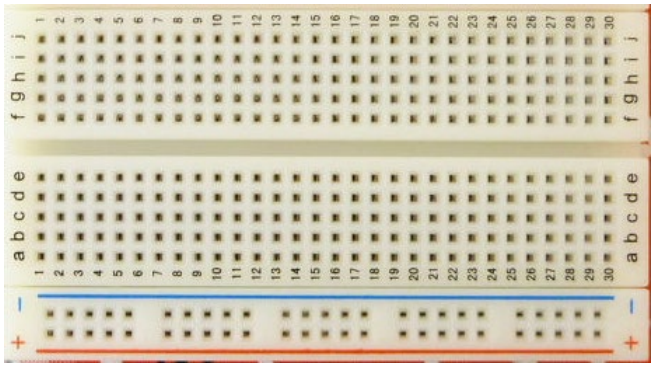


LED

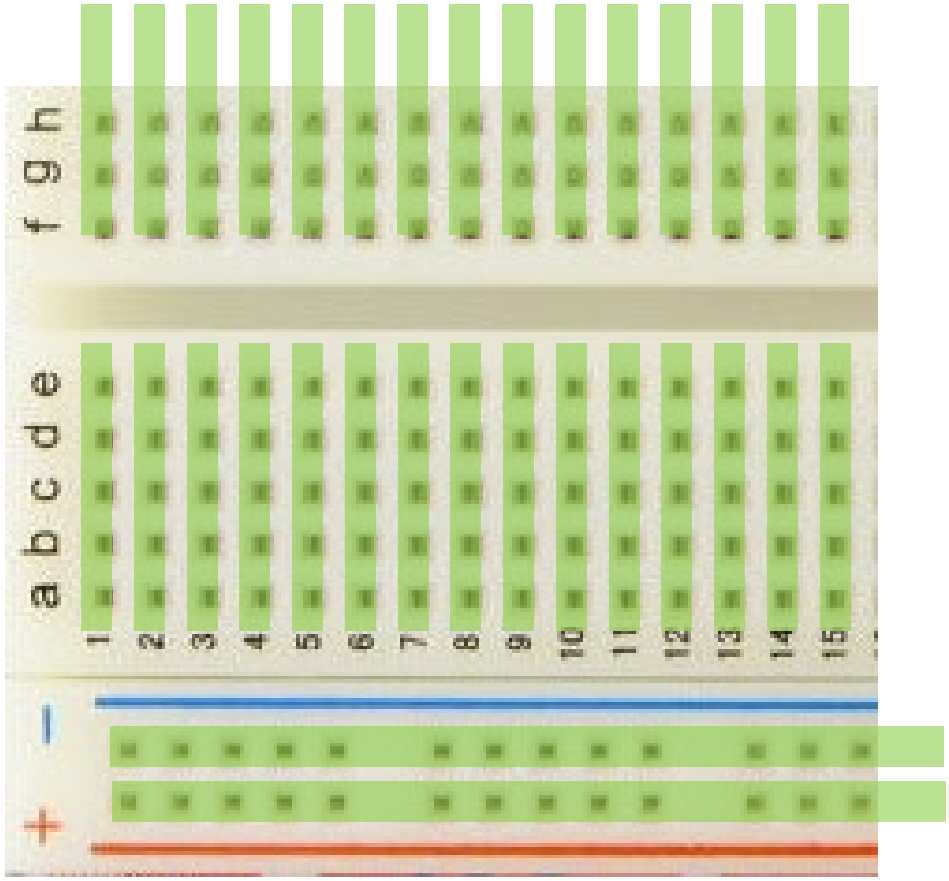
マイコンとブレッドボード



ピン番号を確認しておく



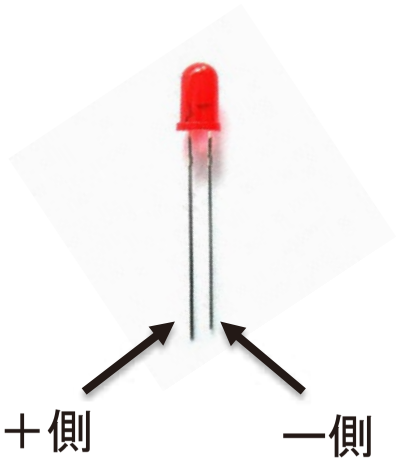
このように接続されている



この接続を利用して、
部品の端子をつなぐ

LED

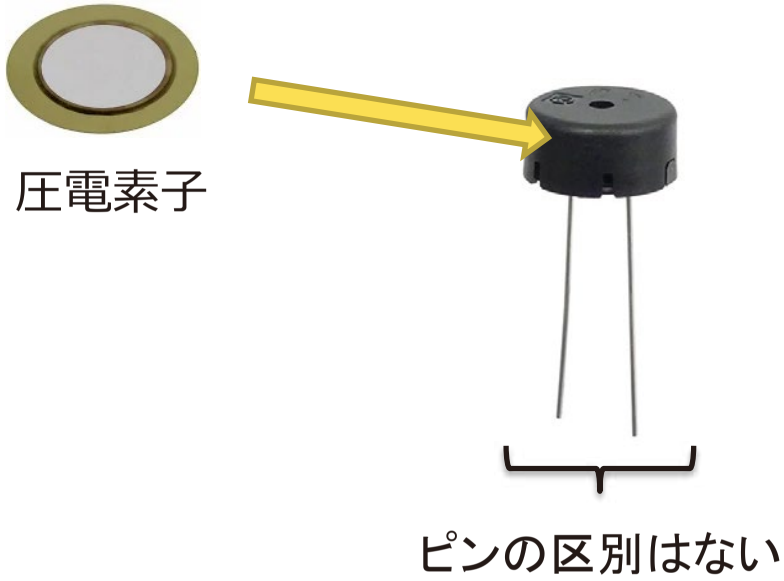
- 電流を流すと光を発する



注意： 電池などの電源に直接接続してはいけない

圧電素子

- 電圧を加えると厚さが変化する
- ピエゾ素子とも呼ばれる



マイコンでできること

- ピンの電圧変化させる
  - ただし電圧は LOW または HIGH のみ
  - 今回のマイコンでは、
    LOWは 0V
    HIGHは 3.3V

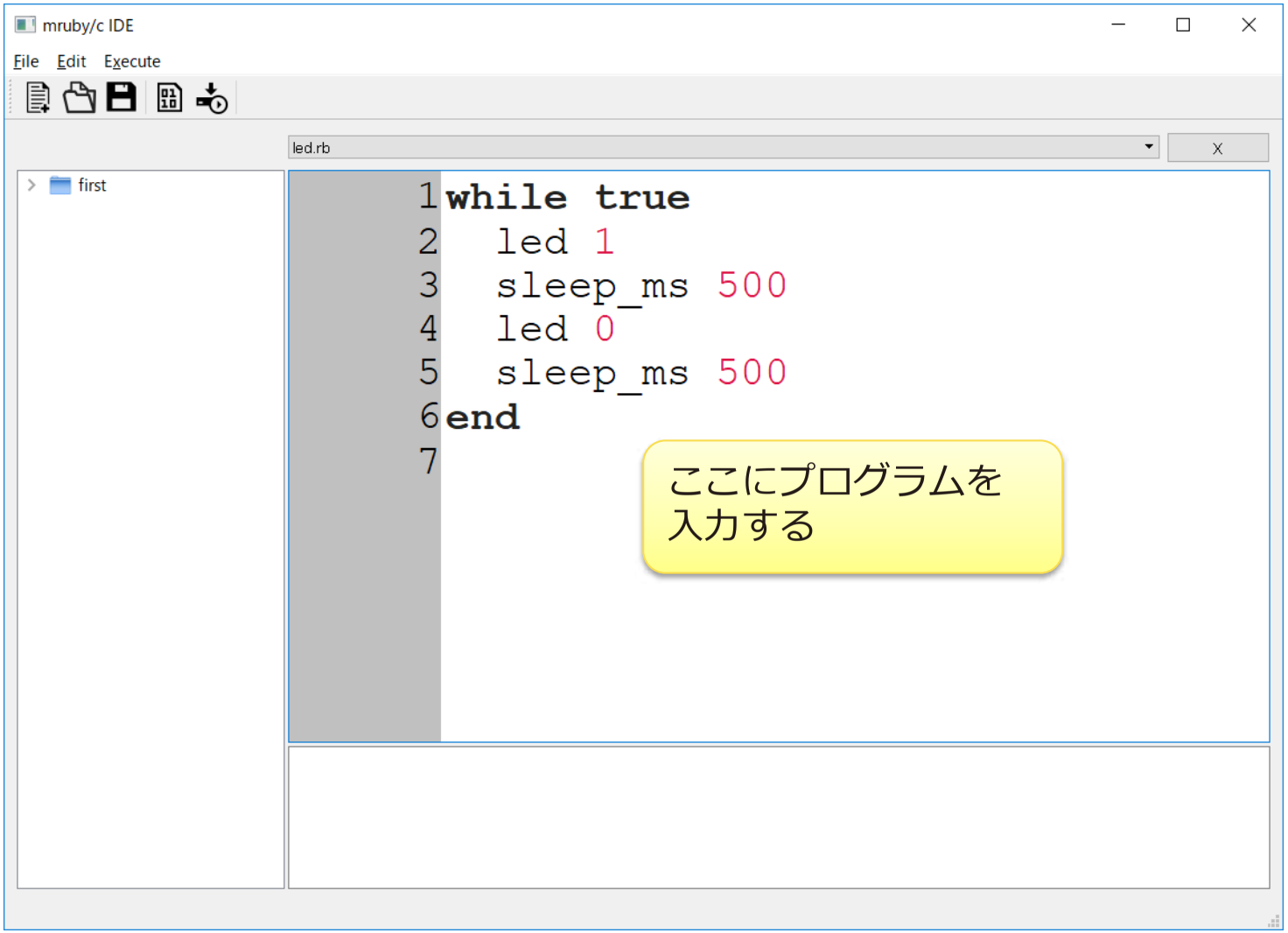
演習 1

簡単なプログラム

- LEDを0.5秒間隔で点滅させる

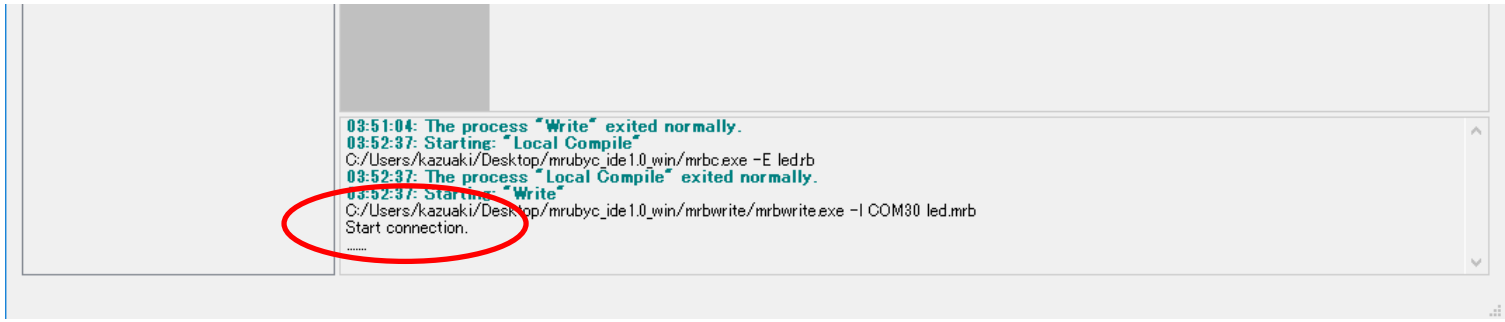
- プロジェクトの作成（File->New->Project）
  - プロジェクト名の例： first
- ファイルの作成（File->New->File）
  - ファイル名の例： led.rb
  - 末尾は「.rb」とする

プログラム

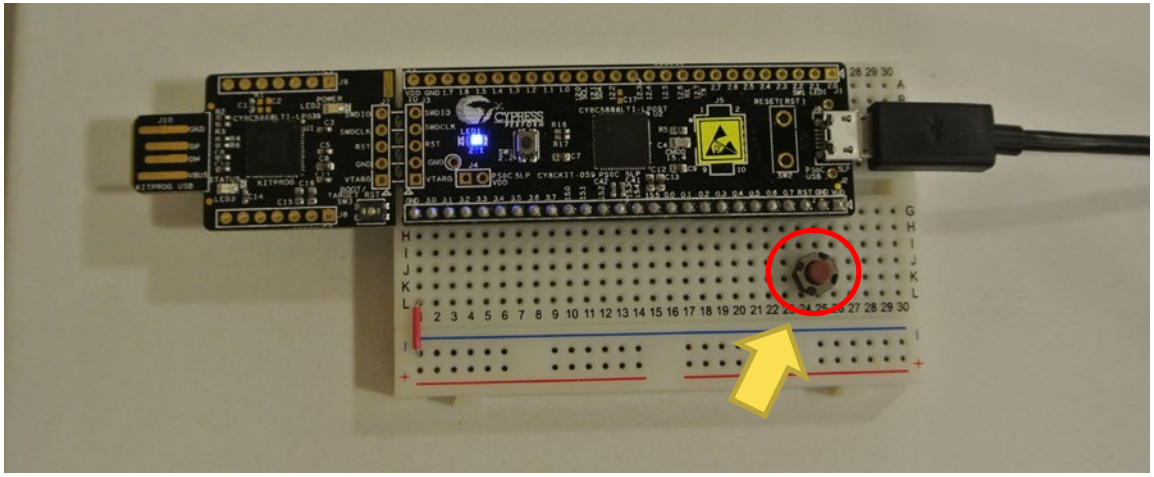

mruby/c IDE
File Edit Execute
led.rb
first
while true
led 1
sleep_ms 500
led 0
sleep_ms 500
end
ここにプログラムを
入力する

プログラムの実行

- プログラムのコンパイルと転送
  Execute->Write
- 「Start connection」が表示されるので、リセットボタンを押す



リセットボタン



プログラムの説明

```
while true
  led 1
  sleep_ms 500
  led 0
  sleep_ms 500
end
```

「while true」
do～endの間を繰り返す
（上から順に1行ずつ実行される）

led 1：LEDを点灯する
led 0：LEDを消灯する

sleep_ms 500
500ミリ秒停止する

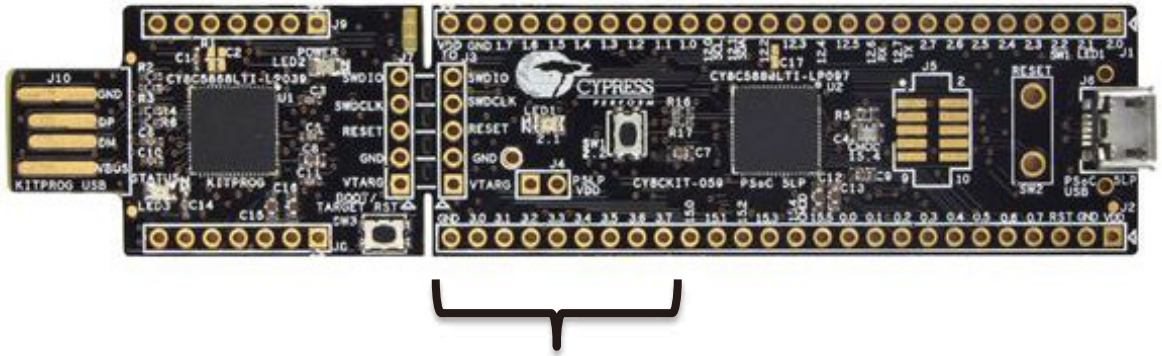
演習

- LEDの点滅速度を変えてみる

演習 2

GPIOを使う

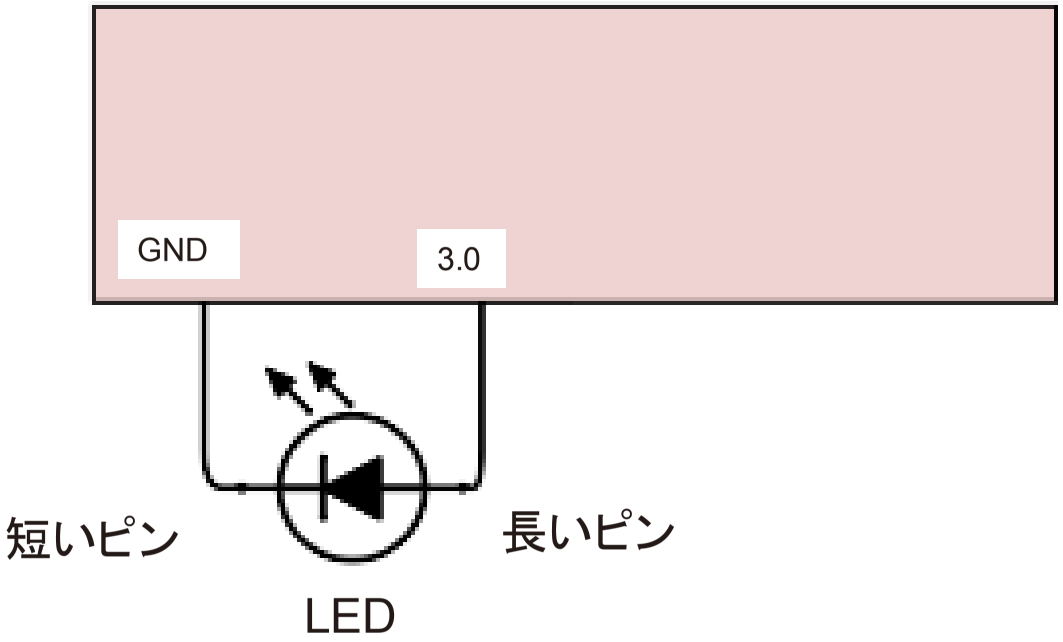
- GPIO（General Purpose Input Output）
  - 「output <pin>,<value>」
    ピン番号 3.<pin> の電圧を制御する
    <pin>: 0～7
    <value>: 0（低い電圧） または 1（高い電圧）



LEDを接続する

- outputでピン「3.0」の電圧を設定できる
  - 電流は電圧（電位）が高いところから低いところへ流れる
  - LEDに電流が流れると、光を出す

回路図



プログラム

- ledの代わりに、outputを使う
- output <ピン番号>, <値>

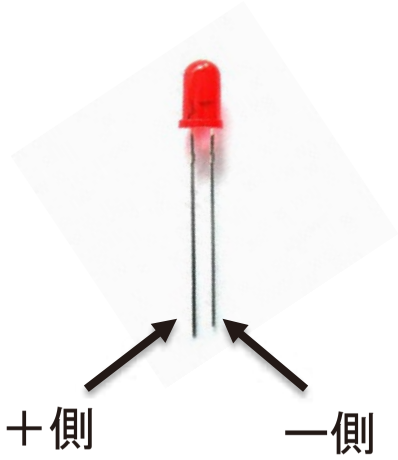
```
while true
  output 0,1
  sleep_ms 500
  output 0,0
  sleep_ms 500
end
```

ピン番号 3.0 を HIGH にする

ピン番号 3.0 を LOW にする

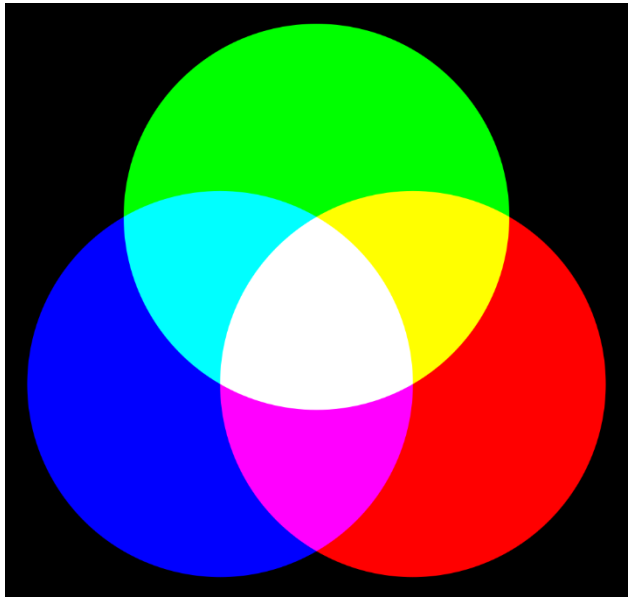
演習（１）

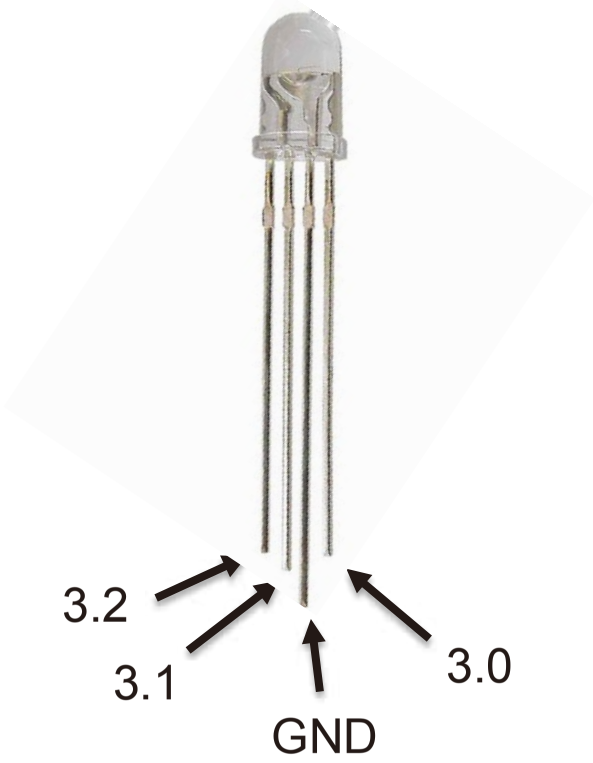
- ３つのLEDを順に点灯させる
  - 点灯させかた、時間などは自由に



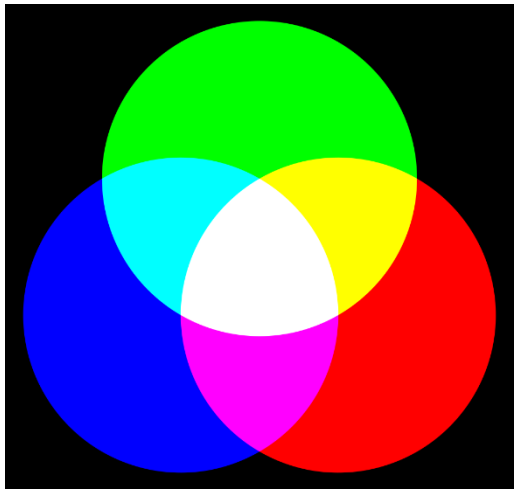
演習（2）

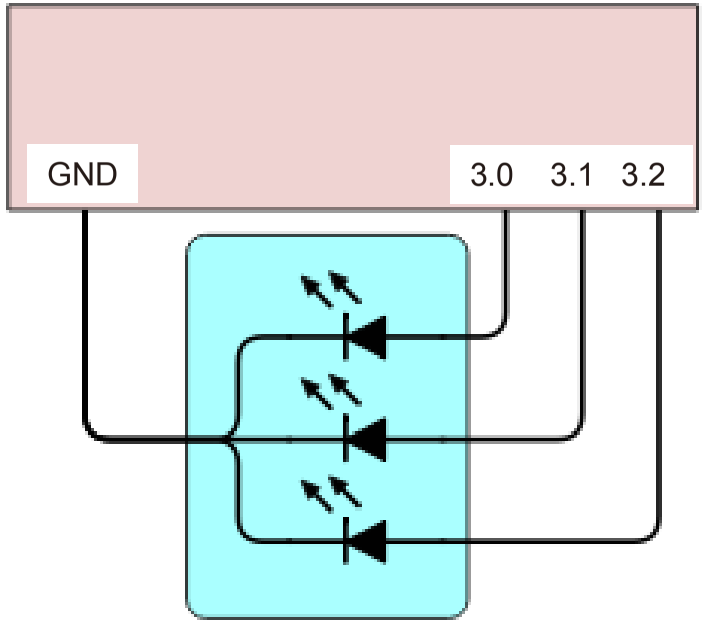
- カラーLEDを点灯させる
- 赤、青、緑の組み合わせ





回路図



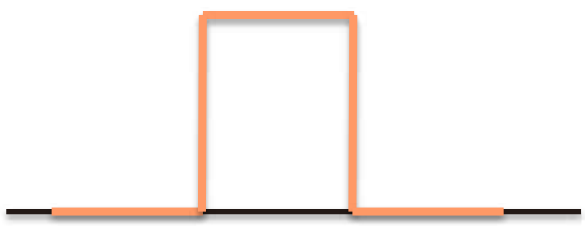


カラーLED

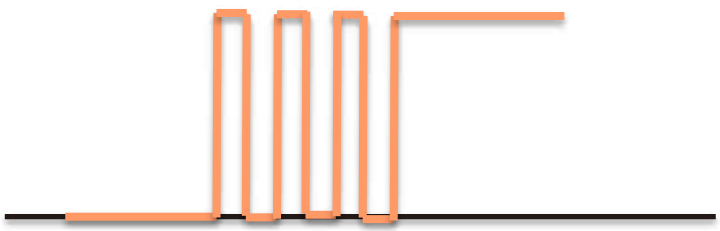
演習 3

PWM

- Pulse Width Modulation
  - GPIOのHIGHとLOWの間の電圧を出力する



outputの出力



PWMの出力

HIGHとLOWを高速に切り替えることで
中間の電圧を出力する

PWMを使ってみる

- PWMを出力する
  - 「pwm <pin>,<value>」
    ピン番号 0.<pin> の電圧を制御する
    <pin>: 0～4
    <value>: 0～255の値でLOW～HIGHの電圧を出力する

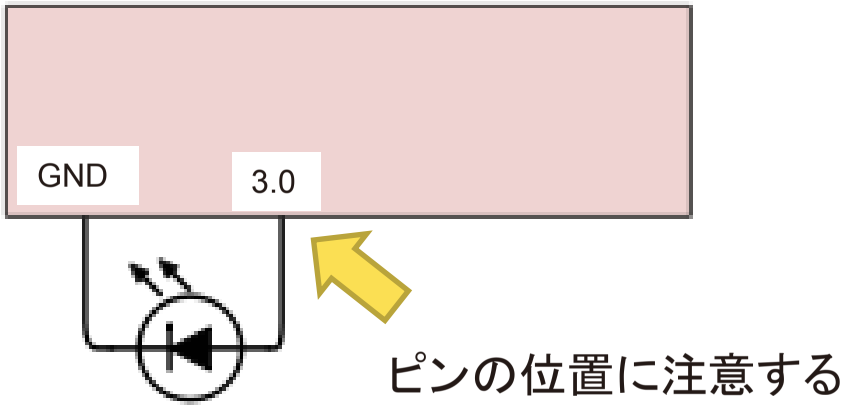
PWMピン
0.0 ～ 0.3

プログラム

- pwmを使う
- pwm <ピン番号>, <値>

```
while true
  pwm 0, 0
  sleep_ms 500
  pwm 0, 50
  sleep_ms 500
  pwm 0, 100
  sleep_ms 500
  pwm 0, 255
  sleep_ms 500
end
```

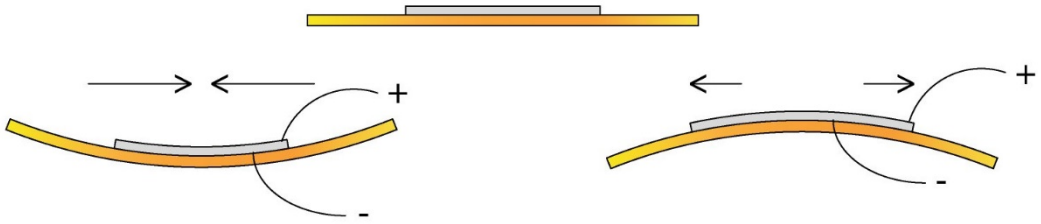
ピン番号 0.0 を使う
出力は 0→50→100→255と変化する


GND
3.0
ピンの位置に注意する

演習 4

音を出す

- 圧電スピーカ（ピエゾ素子）は
  電圧を加えると形が変化する
  ↓
- 周囲の空気の体積が変化する
  ↓
- 音（空気の振動）



プログラム

- 圧電スピーカを 15.0 と ー に接続する
- 「tone <周波数>」

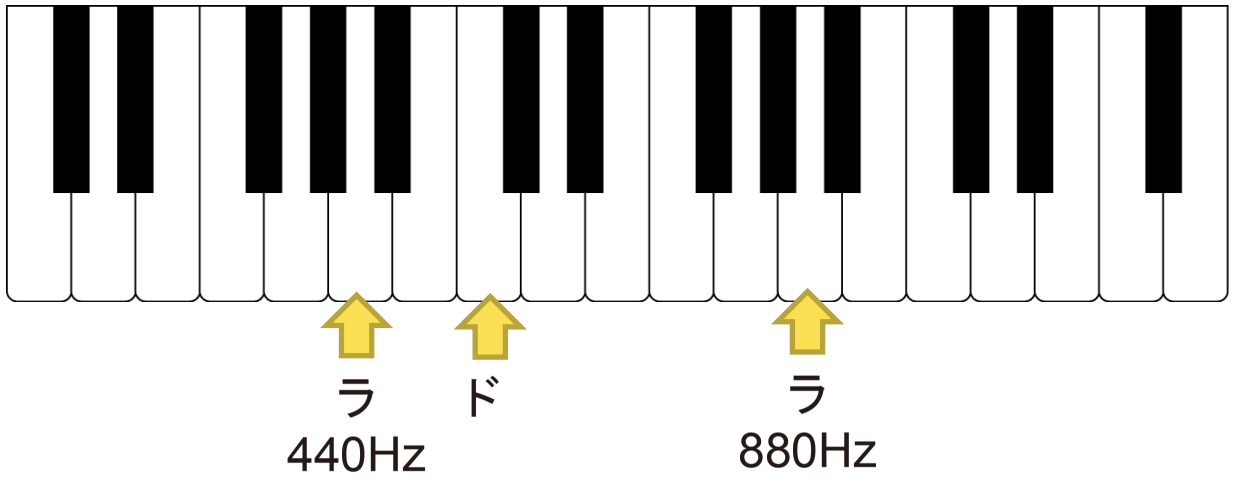
```
while true
  tone 440
  sleep_ms 700
  tone 0
  sleep_ms 300
end
```

440Hz＝ラの音

0Hz＝無音

音の計算

- １オクターブで周波数は２倍になる
- 音階は「等比数列」になっている
  （白鍵・黒鍵関係なく）
- ラの音が 440Hz



計算方法

- 12個の音階で周波数が2倍になる
- 隣り合う音階の公比を r とする

$$440r^{12} = 880$$

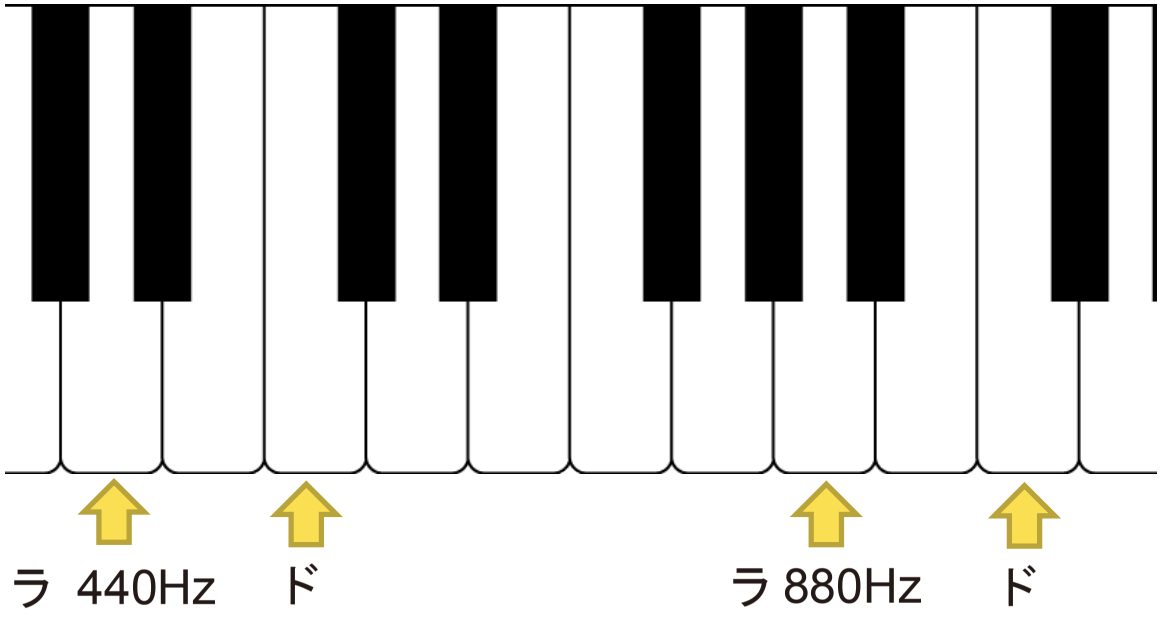
- r を求める

$$r^{12} = 2$$

$$r = \sqrt[12]{2} \cong 1.0595$$

演習（3）

- 「ド・レ・ミ」を繰り返し演奏する
- 参考：音と音の間に少し休符を入れると聞きやすい



$$r \cong 1.0595$$

平成 30 年度「専修学校による地域産業中核的人材養成事業」
コネクティッド・ホームの普及に必要な中核的技術者養成事業

■実施委員会

◎	伊東　和幸	大阪工業技術専門学校　副校長
	佐々木　章	東京テクニカルカレッジ
	村岡　好久	名古屋工学院専門学校 講師／一般社団法人 TukurouneMono 振興協会代表理事
	荒川　豊	奈良先端科学技術大学院大学　情報科学研究科　准教授
	田中　和明	九州工業大学　情報工学研究院　機械情報工学研究系　准教授
	草野　賢次	パナソニック株式会社　エコソリューションズ社技術本部 Ｒ＆Ｄ企画室オープンイノベーション企画部　部長
	阪本 権一郎	株式会社ケイ・アイ・エス　代表取締役
	甲斐 俊亘	株式会社三興社　総務部　課長
	岡山　保美	株式会社ユニバーサル・サポート・システムズ 取締役
	吉岡　正勝	株式会社日本教育ネットワークコンソシアム
	小幡　忠信	一般社団法人 Ruby ビジネス推進協議会　理事長
	木村 貞基	公益社団法人日本建築家協会近畿支部　副支部長
	飯塚　正成	一般社団法人全国専門学校情報教育協会　専務理事

■調査・開発委員会

◎	堀部　達夫	大阪工業技術専門学校 ロボット・機械学科　学科長
	松田　財秀	大阪工業技術専門学校 設備環境デザイン学科　教員
	岡山　保美	株式会社ユニバーサル・サポート・システムズ 取締役
	村岡　好久	名古屋工学院専門学校 講師／一般社団法人 TukurouneMono 振興協会代表理事
	吉岡　正勝	株式会社日本教育ネットワークコンソシアム
	高畑　道子	一般社団法人 Ruby ビジネス推進協議会　副理事長
	吉岡　信吾	一般社団法人 Ruby ビジネス推進協議会　理事
	阪本 権一郎	株式会社ケイ・アイ・エス　代表取締役
	甲斐 俊亘	株式会社三興社　総務部　課長

■評価委員会

◎	飯塚　正成	一般社団法人全国専門学校情報教育協会　専務理事
	木村 貞基	公益社団法人日本建築家協会近畿支部　副支部長
	有馬　貴博	経済産業省近畿経済産業局地域経済部次世代産業・情報政策課

平成 30 年度「専修学校による地域産業中核的人材養成事業」
コネクティッド・ホームの普及に必要な中核的技術者養成事業

センシング装置を使った組込みソフトウェア教材

平成 31 年 3 月

学校法人福田学園　大阪工業技術専門学校
〒530-0043　大阪府大阪市北区天満 1-9-27
TEL　06-6352-0093　　FAX　06-6352-5995

●本書の内容を無断で転記、掲載することは禁じます。