組込み技術を利用して、血中酸素濃度測定装置を作り、体調管理を万全にしよう

※bpm：beats per minute　心拍数
※SpO2：Saturation（飽和）、Pulse（脈）、0_2（酸素）　経皮的動脈血酸素飽和度

新型コロナウィルスが猛威を振るい、外出自粛など生活にも影響がでています。でも、もし感染したら怖いです。感染の可能性がある症状が現れても、なかなか検査が受けられない現状。また、仮に感染したとしても病院では経皮的動脈血酸素飽和度の測定、体温測定、そして安静しか対処ができないとのことです。

また、4 月 28 日には厚生労働省から新型コロナウィルスの「緊急性の高い症状」について公表されました。その多くは、呼吸が困難になる、息ができない　などの症状です。

これらの症状は、動脈血酸素飽和度が低くなる、すなわち肺が正しく機能していないときに、血液中の酸素濃度が低下して呼吸困難になるなどの症状として現れます。

今回、この動脈血酸素飽和度を測定しようと試みるものです。医療用では「パルスオキシメーター」として販売されていますが、これを自作してみようと思います。組込み技術を使い、健康管理しつつ、在宅などでの余暇を活用してみませんか？

経皮的動脈血酸素飽和度（SpO2）とは？

SpO2 とは、血液中にどの程度の酸素が含まれているかを示します。SpO2 の S は　Saturation（飽和）, P は Pulse（脈）, O2 は酸素を示しています。血液中には酸素を運ぶヘモグロビンがあります。SpO2 は、血液中（動脈）の多くのヘモグロビンの何 %が酸素を運んでいるかを示しています。正常値は 96 %以上、95 %未満は呼吸不全の疑いがあり、90 %未満は在宅酸素療法の適用となります。

引用：http://chuo.kcho.jp/original/clinicallabo/shin_hai/spo2/SPO2.html

今回は、マイコンボード、心拍・血中酸素飽和度センサ、iPad を利用します。動脈血酸素飽和度を SpO2 センサのデータとしてマイコンボードで読み取り、そのデータを Bluetooth 通信で iPad に送信し、iPad で表示、監視する装置です。健康な状態であれは、98%程度でありますが、これが 95% 以下は肺に何か異常があるのではないかと思われます。医師ではないので正しい意見ではありませんが、普段から SpO2 を測定しておき、SpO2 の低下を確認することにより、肺炎がおきているという推察はできるそうです。

■機材

マイコンボード：Raspberry Pi 3 Model B+

URL：https://www.raspberrypi.org/

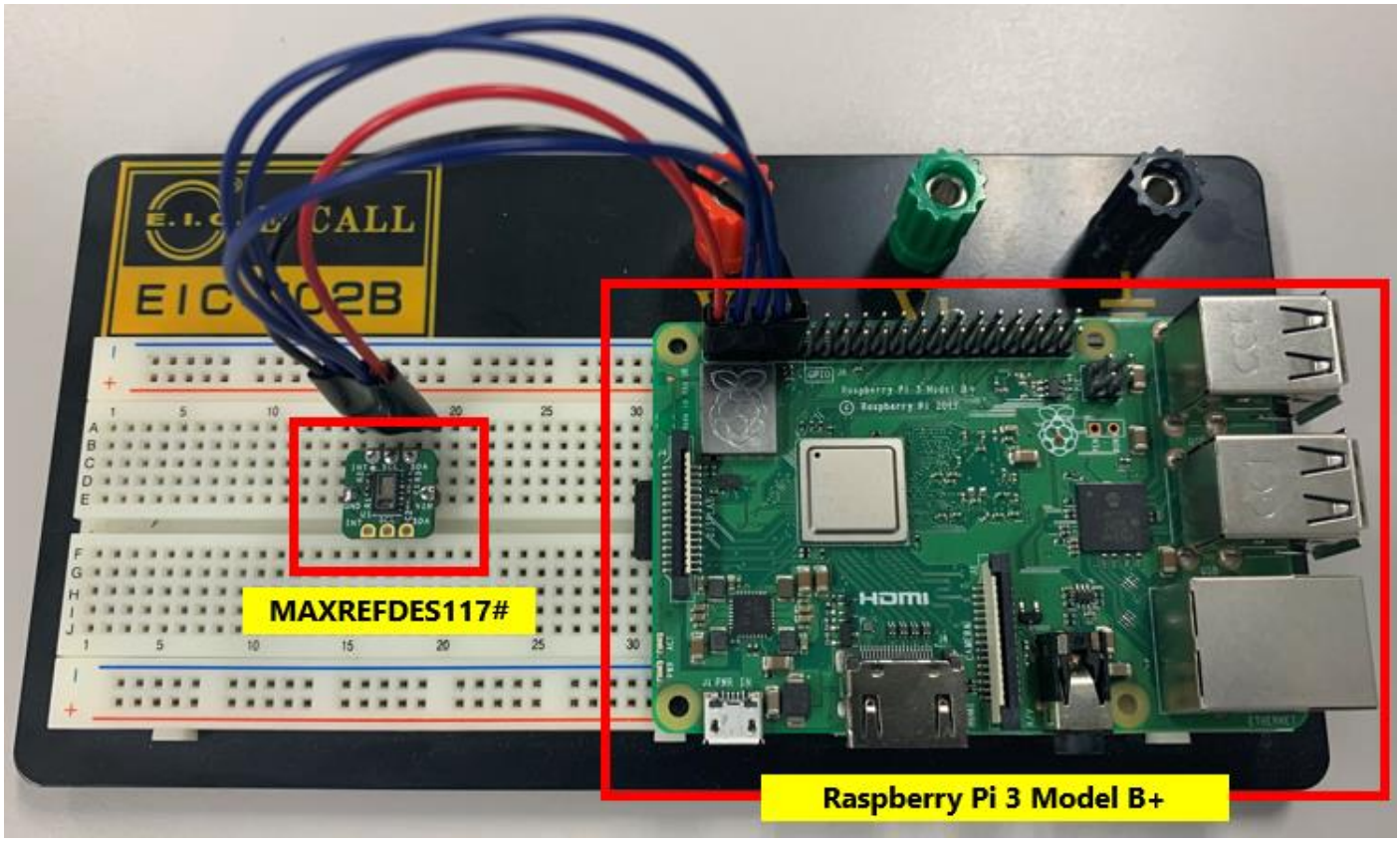
心拍・血中酸素飽和度センサ：MAX30102

今回の装置ではレベル変換（MAX14595）、電圧レギュレータ（MAX1921）がセットになった MAXREFDES117#を使用しています。

URL：https://www.maximintegrated.com/jp/design/reference-design-center/system-board/6300.html

■回路図（ピン配置）

Raspberry Pi 3 Model B+ と MAXREFDES117#を接続します。



上記ではブレッドボードを使用して接続しています。

MAXREFDES117#には VIN、SDA、SCL、INT、GND が用意されていますので、

それぞれの Pin を Raspberry Pi の

Pin01(3.3V)

Pin03(GPIO2/SDA1 I2C)

Pin05(GPIO3/SCL1 I2C)

Pin07(GPIO4/GPCLK0 1 Wire)

Pin09(GND)

と接続します。

・Raspberry Pi3 B+ GPIOのピン配置

Pin01: 3.3V	Pin02: 5V
Pin03: GPIO2/SDA1 I2C	Pin04: 5V
Pin05: GPIO3/SCL1 I2C	Pin06: Gnd
Pin07: GPIO4/GPCLK0 1 Wire	Pin08: GPIO14/UART0_TXD
Pin09: Gnd	Pin10: GPIO15/UART0_RXD
Pin11: GPIO17	Pin12: GPIO18/PWM_CLK
Pin13: GPIO27/PCM_DOUT	Pin14: Gnd
Pin15: GPIO22	Pin16: GPIO23
Pin17: 3.3V	Pin18: GPIO24
Pin19: GPIO10/SPI0_MOSI	Pin20: Gnd
Pin21: GPIO9/SPI0_MISO	Pin22: GPIO25
Pin23: GPIO11/SPI0_SCLK	Pin24: GPIO8/SPI0_CEO
Pin25: Gnd	Pin26: GPIO7/SPI0_CE1
Pin27: GPIO0/ID SD	Pin28: GPIO1/ID SC
Pin29: GPIO5	Pin30: Gnd
Pin31: GPIO6	Pin32: GPIO12
Pin33: GPIO13	Pin34: Gnd
Pin35: GPIO19	Pin36: GPIO16
Pin37: GPIO26	Pin38: GPIO20
Pin40: Gnd	Pin41: GPIO21

Rspberry Pi3		MAXREFDES117#
Pin01	⇔	VIN
Pin03	⇔	SDA
Pin05	⇔	SCL
Pin07	⇔	INT
Pin09	⇔	GND

接続するピンを間違えるとセンサが故障する原因にもなりますので、接続するピンを間違えないように十分な注意が必要です。

■ソフトウェア処理の流れ

心拍・血中酸素飽和度センサと Raspberry Pi は I2C 通信で制御します。I2C 通信を利用し、心拍・血中酸素飽和度センサのデータを Raspberry Pi で取得します。

I2C 通信とはフィリップス社で開発されたシリアルバスのことで、シリアルデータ（SDA）とシリアルクロック（SCL）の 2 本のバスで構成されています。

引用：https://ja.wikipedia.org/wiki/I2C

I2C 通信にはマスタとスレーブがあり今回はマスタ（Raspberry Pi）、スレーブ（心拍・血中酸素飽和度センサ）の関係で動作させます。

Raspberry Pi で I2C 通信を行うためにはモジュールの有効化を行う必要があります。以下の手順で I2C 通信設定を行い、有効化します。

1.以下のコマンドを実行する。

sudo raspi-config

2.「5 Interfacing Options」を選択する。

3.「P5 I2C」を選択する。

4.以下のメッセージが表示されるので「Yes」を選択する。

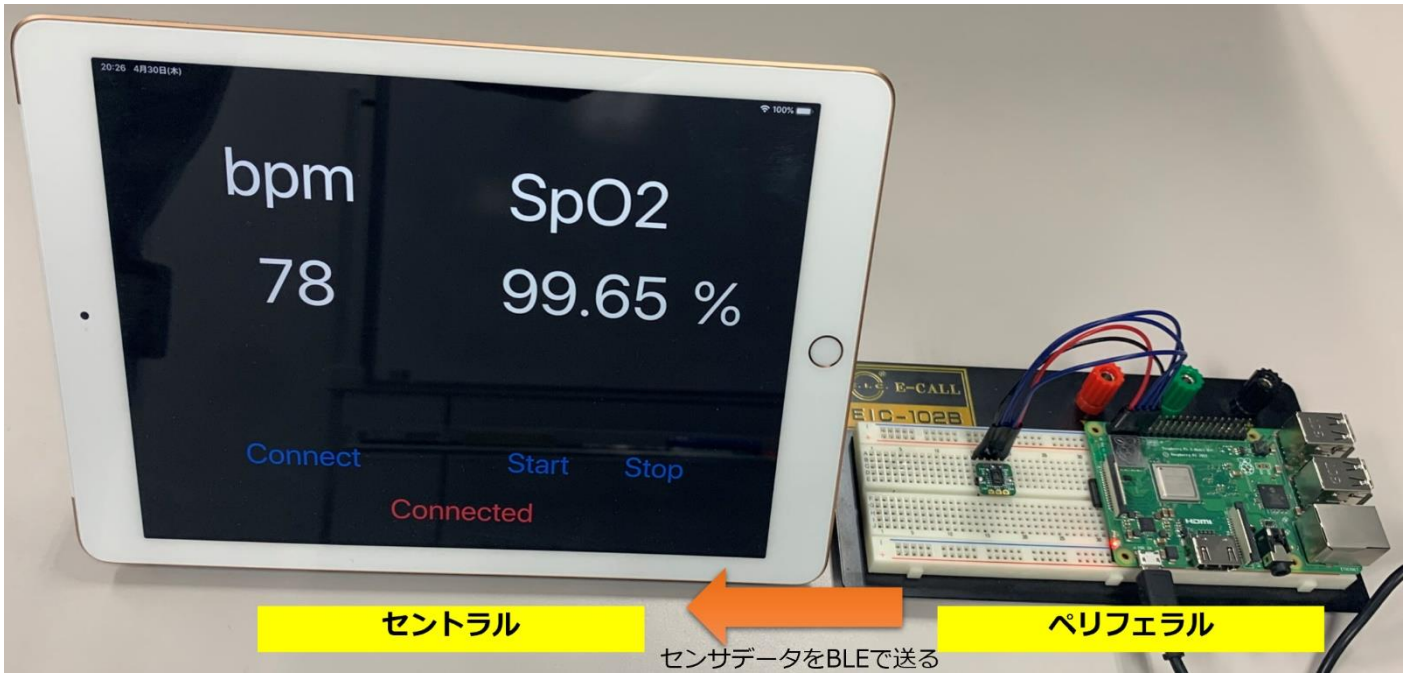
Would you like the ARM I2C interface to be enabled?

I2C 通信で心拍・血中酸素飽和度センサ（MAX30102）のデータを取得するためにはサンプルのソースコードが公開されていますので、サンプルのソースコードを利用しデータを取得します。

続いて Bluetooth 通信を使い Raspberry Pi で取得した心拍と血中酸素飽和度のデータを iPad へ送り表示します。本システムでは Bluetooth Low Energy（BLE）で通信します。BLE とは Bluetooth の Version4.0 から追加になった低消費電力の通信モードです。

引用：https://ja.wikipedia.org/wiki/Bluetooth_Low_Energy

Raspberry Pi と iPad では BLE 通信が可能であるため、BLE 通信を使いデータの送受信を行います。BLE 通信では、データが提供される側をセントラル、データを提供する側をペリフェラルと呼びます。今回は iPad 側をセントラル、心拍・血中酸素飽和度センサと接続した Raspberry Pi をペリフェラルとし、BLE 通信をします。



BLE 通信の流れとしては以下になります。

(1) ペリフェラル側にセントラルから実行して欲しい機能を実行する。
(2) ペリフェラル側に実装する機能はキャラクタリスティック（特性）からなるサービスという単位で実装する。
(3) セントラルはペリフェラルがアドバタイズをしてくれることで端末を発見する。
(4) セントラルはペリフェラルを発見したら、そのサービスを見つけ、更にサービスのキャラクタリスティックを見つけ、書き込みや読み込みをする。

セントラル（iPad）側の処理

① ペリフェラルを検索する。
② ペリフェラルに接続する。
③ サービスを見つける。
④ キャラクタリスティックを見つける。
⑤ キャラクタリスティックに書き込みをする。

ペリフェラル（Raspberry Pi）側の処理

1．キャラクタリスティックを作成する。

２．キャラクタリスティックに書き込み、読み込みがあったときにどのような処理をするかを実装する。
３．サービスを作成し、キャラクタリスティックを追加する。
４．アドバタイズを実行する。

となります。

本システムでは上記のようになりますが、他に Raspberry Pi へ温度センサを取り付けることで、体温も同時に計測するということも可能です。その場合は体温値というキャラクタリスティックを実装すれば BLE 通信で iPad へデータを送ることが可能になります。

本システムの開発では Raspberry Pi 側は Python、iPad 側は Swift で開発しました。BLE 通信を実現するために Raspberry Pi では Python 版の BLE ライブラリである pybleno、iPad 側では Core Bluetooth フレームワークが用意されていますのでそれらを使用します。BLE の処理の流れを理解しライブラリを使用すれば BLE 通信を行うことが可能です。

■まとめ
皆さんも組込み技術でできるコロナ対策に Try してみませんか？このシステムを完成させ自宅で健康管理を行ってください。

※当社規定ではレビュー実施前のためソースコードの公開はできません。
本システムにご興味がある方は Web サイトのお問合せより個別にご連絡ください。