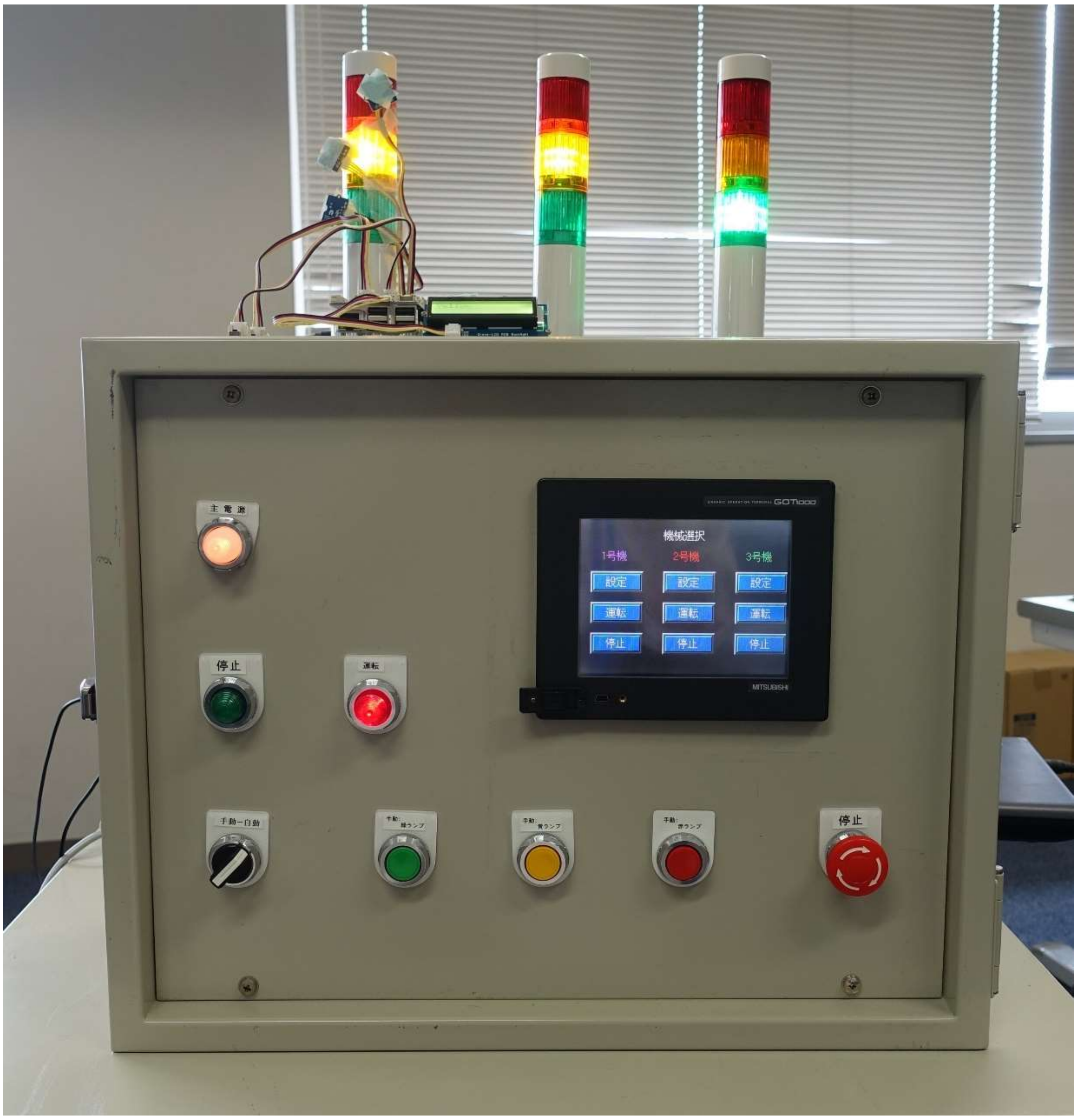
令和２年７月30日

ランプ状況取得表示システムとシステムユニットへの適用方法



富山職業能力開発促進センター
電気・電子系指導員　市川　拓実

内容

1　初めに

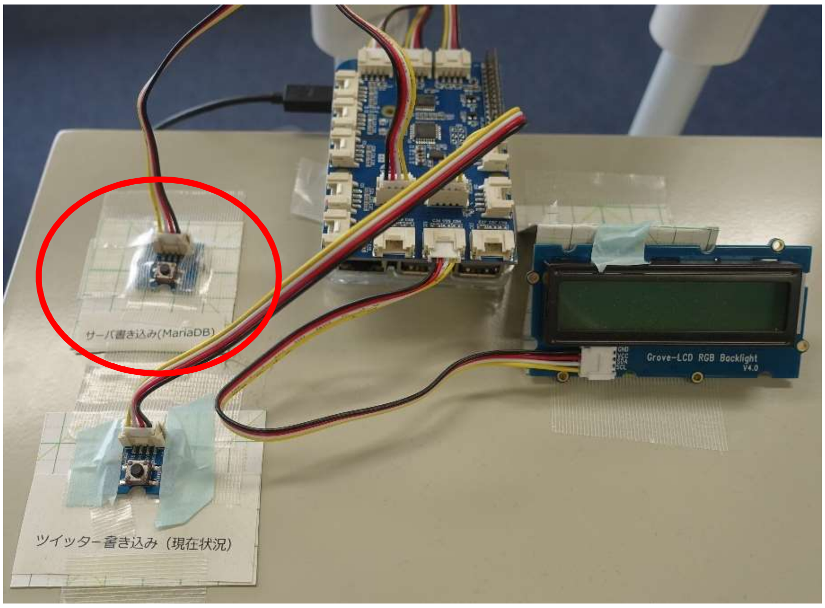
近年第４次産業革命がおこるとされており、中でも AI、IoT、ビックデータ、クラウドが注目されています。モノのインターネット・IoT(Internet of Things)では様々な「モノ」からデータを収集しようと考えられています。生産現場では古い機械のデータを集める際に「ランプの状況を取得」をしています。ランプの状況をサーバに蓄積し見える化を行い、稼働時間の改善につなげています。提示するシステムでは「ランプの状況を取得→記録→表示」といった内容を習得できるとともに、工場や生産現場での活用性を重視しております。またシステムユニットとの関連性を明示するともに離職者訓練、在職者訓練への拡大を想定しております。ユニットの詳細は基盤整備センターのホームページ(URL: https://www.tetras.uitec.jeed.or.jp/)にて検索してください。

2　　ランプ状況取得表示システム動作概要

1.　PLC にてランプを稼働させます。切り替えスイッチにて手動・自動を選択し、自動の場合はタッチパネルにて設定を行います。

2.　Raspberry Pi の電源を入れ、ランプの状態を読み取っているか確認します。

3.　Raspberry Pi(Grove Pi+)のボタンを押します。



4.　確認用 LCD に「Send to Server Please wait…」と表示されます。

5.　Web ブラウザにて確認を行います。

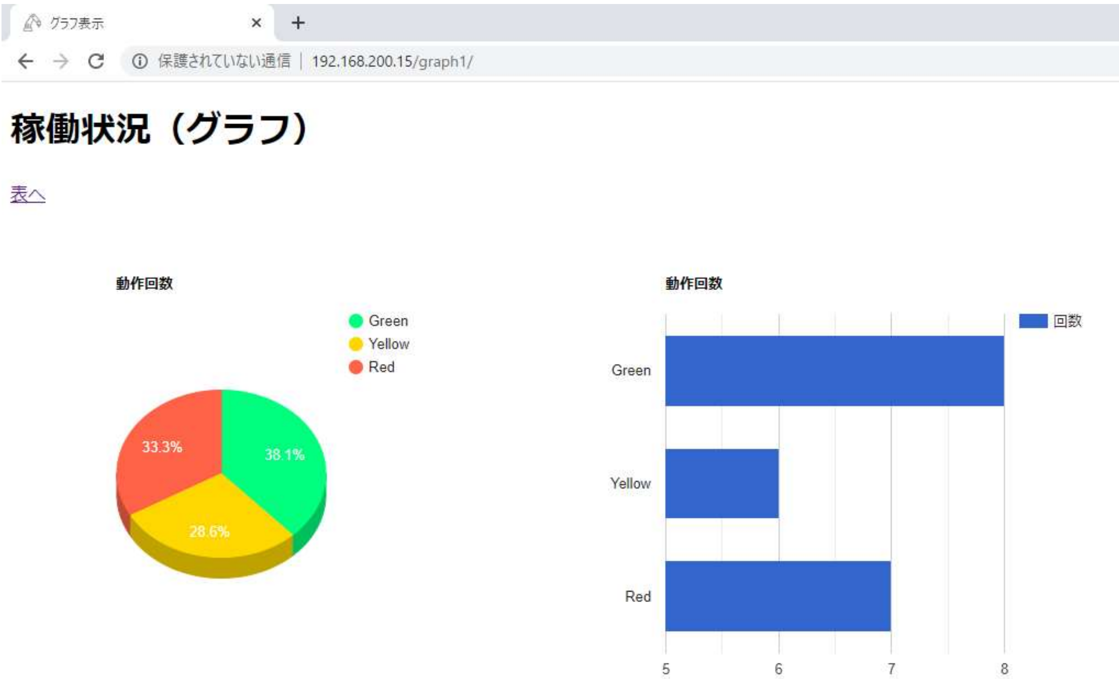
稼働状況

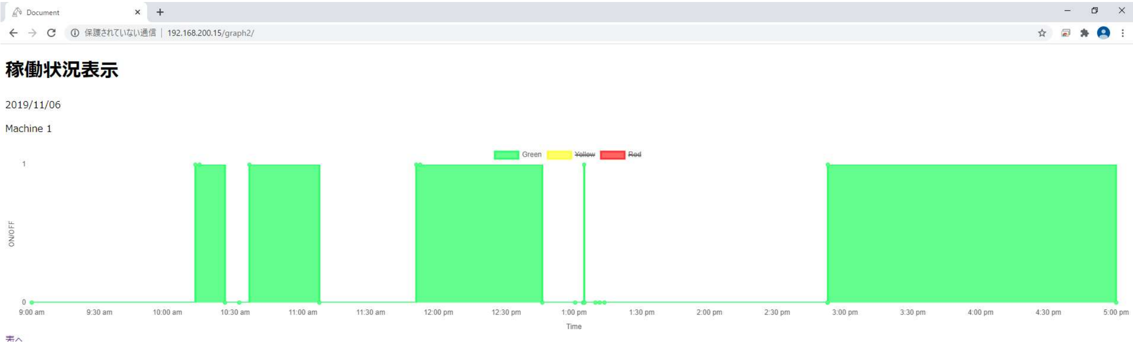
保護されていない通信 | 192.168.200.15/chart/

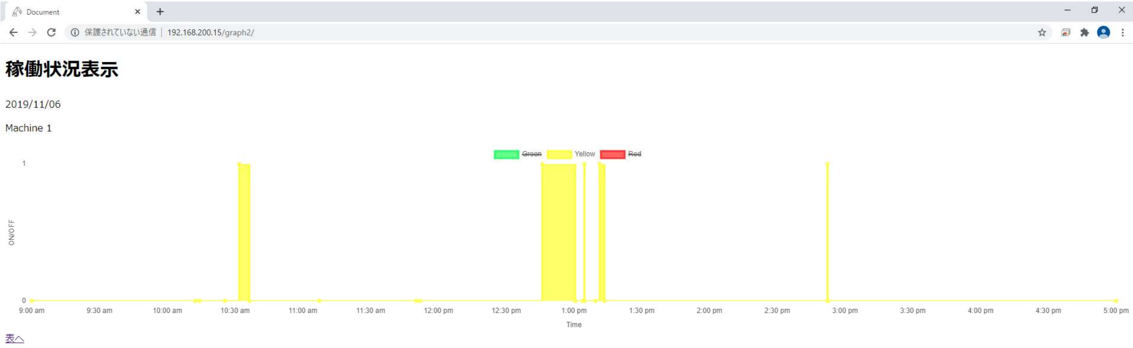
稼働状況（表）

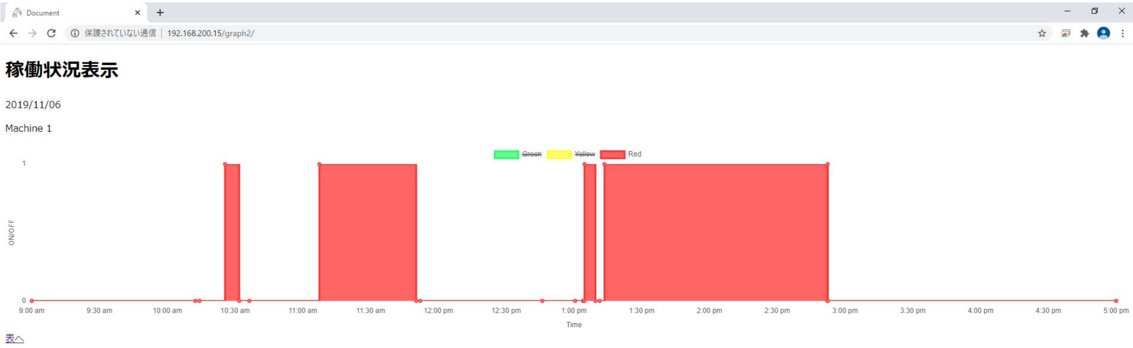
グラフ1へ グラフ2へ

時間	緑ランプ	黄ランプ	赤ランプ
2019-11-06 10:12:26	ON	OFF	OFF
2019-11-06 10:14:17	ON	OFF	OFF
2019-11-06 10:25:34	OFF	OFF	ON
2019-11-06 10:31:51	OFF	ON	OFF
2019-11-06 10:36:20	ON	OFF	OFF
2019-11-06 11:07:19	OFF	OFF	ON
2019-11-06 11:50:11	ON	OFF	OFF









プログラム等の詳しい動作内容については動作詳細にて説明します。

3　ランプ状況取得表示システムの使用を想定したユニット

適応可能なユニット

ユニット番号	ユニット名	適用箇所
IU499-X051-3	生産情報管理システム構築 1 （基本設計）	ランプ状況取得表示システムを一貫して使用可能
IU499-1052-3	生産情報管理システム構築 2 （データ収集）	
IU499-1053-3	生産情報管理システム構築 3 （データファイリング）	
IU499-1054-4	生産情報管理システム構築 4 （Web とデータベースの連携）	
IU499-1055-3	生産情報管理システム構築 5 （発表・評価）	
IU499-1056-3	生産情報管理システム構築 6 （ドキュメント作成）	
IU701-1030-1	通信サービスの種類	社会システムの概要の説明
IU704-1060-1	インターネット利用技術	インターネットの概要の説明

4　使用した機器構成

ハードウエア（ランプ情報収集用）

機器名	型番	数量(個)	金額(単価)	代用可能
パソコン	HP Compaq Pro 6300 SFF	1	50000	○
Raspberry Pi スターターキット	Raspberry Pi 3B+	1	10000	△
Grove Pi+ ベース	103010002	1	5000	△
Grove Pi+ ケーブル	ACC83057O	6	100	△
Grove Pi+ ボタン	101020003	2	250	△
Grove Pi+ センサ	101020132	3	350	△
Grove Pi+ LCD	104030001	1	1600	△

ハードウエア（ランプ表示用）

機器名	型番	数量(個)	金額(単価)	代用可能
PLC	MITSUBISHI A2US ※2014/9 生産終了	1	100000	○
タッチパネル	GT1155-QSBD ※2016/10 生産終了	1	115000	○
スイッチング・パワーサプライ	S82K-03024 ※2011/3 生産終了	1	10900	○
コントロールボックス	CS25-54A	1	27500	○
積層ランプ	LR4-302LJNW-RYG	3	10500	△
押しボタン	ABS111NG ABS111NY ABS111NR	各 1 計 3	1300	○
切り替えスイッチ	ASW201	1	900	○
非常停止用スイッチ	AVN311NR-MAU	1	4000	○
動作表示ランプ	APS122DNG APS122DNR APS122DNW	各 1 計 3	1300	○
電線	KIV 0.75 青(100m)	1	4000	○
カッチングダクト	BDR-261(1m)	1	1000	○
圧着端子	1.25Y-3.5(100 個)	1	500	○

DIN レール	IODA-50	2	200	○
電源コード	P 付-VFF 0.75SQ-3m	1	500	○
M10 ナット		6	25	○
M10×15 ボルト		6	70	○

ソフトウエア

ソフトウエア名	内容	金額	代用可能
Windows 10	パソコンの OS	17000	○
Xampp	Apache2、PHP、MariaDB、phpMyAdmin の一括インストール	Free	○
Raspbian	Raspberry Pi の OS	Free	△
GX Works2	ラダー図作成	150000	○

4.1 代用可能製品について

4.1.1 パソコン

Xampp の使用を満たしているものであれば構いません。Windows 10 が快適に動作するものであれば基準を満たしています。パソコンではなく Raspberry Pi で運用することも可能ですが Xampp ではなく各ソフトをそれぞれインストールする必要があります。インストールするソフトは「Apache2、PHP、MariaDB、phpMyAdmin」です。phpMyAdmin はブラウザ経由でデータベースを操作する予定がない場合は必要ありません。

4.1.2 Raspberry Pi

富山職業能力開発促進センターでコースを立ち上げた際、購入したのが Raspberry Pi 3B+で Grove Pi+が対応する機種であればどのバージョンの Raspberry Pi を使用しても構いません。

4.1.3 Grove Pi+

電子回路を理解しており、センサを自作または購入して Raspberry Pi の GPIO に接続できるのであれば代用可能です。Grove Pi+はボタンのほかにも湿温度センサや距離センサも用意されています。今回の光センサの調整はプログラム側で行っています。

4.1.4 PLC

入力 5 点以上、出力 11 点以上であればどのような機種でも構いません。

4.1.5 タッチパネル

積層ランプを個別と時間で制御するために使用しました。押しボタンではすべての積層ランプの緑、黄、赤が

同時に点灯するようになっています。

4.1.6　スイッチング・パワーサプライ

PLC に 24V 出力がないため使用しています。合計電流値を計算して PLC 本体から供給できるのであれば不要です。

4.1.7　コントロールボックス

様々なボックスの種類がありますが、箱に入る部品の大きさを考慮して選定してください。箱ではなく板に装着する場合は部品配置や置いた時のバランスを考えて選定してください。

4.1.8　積層ランプ

種類としては様々ありますが、360 度、光が届くタイプを選定しました。

4.1.9　押しボタン、切り替えスイッチ、緊急停止ボタン

有接点シーケンス制御[EU301-0030-1]などで使用しているもので構いません。

4.1.10　動作表示ランプ

24V で構成しているため 24V 対応のものであれば構いません。

4.2　価格を踏まえた機器構成

作成当時は元々所有していたものを流用しているので作成したときの追加購入は積層ランプとボルトとナットのみでした。流用品がなく同じ部品を集めるのには標準価格で約 40 万円してしまいます。代替品、生産終了した部品の後継品の入手のしやすさから考えた場合、以下の通りとなります。

機器名	型番と詳細	数量 (個)	金額 (単価)
Raspberry Pi スターターキット	Raspberry Pi 4B	1	10000
Grove Pi+ ベース	103010002	1	5000
Grove Pi+ ケーブル	ACC83057O	6	100
Grove Pi+ ボタン	101020003	2	250
Grove Pi+ センサ	101020132	3	350
Grove Pi+ LCD	104030001	1	1600
PLC	MITSUBISHI FX3U-16MT	1	30000
積層ランプ	LR4-302LJNW-RYG	3	10500
押しボタン	ABS111NG	各 1	1300

	ABS111NY ABS111NR	計 3	
切り替えスイッチ	ASW201	1	900
非常停止用スイッチ	AVN311NR-MAU	1	4000
動作表示ランプ	APS122DNG APS122DNR APS122DNW	各 1 計 3	1300
M10 ナット		6	25
M10×15 ボルト		6	70

PLC や有接点シーケンス制御を当該施設で実施しているのであれば Raspberry Pi、Grove Pi+、積層ランプ（合計 5 万円程度）のみを購入することにより価格を抑えることができます。制御盤に必要な工具類は記載しておりません。

4.3　プログラミング言語

今回使用している言語は以下の通りです。各種公開されているサンプルをベースに作成しています。プログラムの詳細は配布データを参照してください。

言語名	使用用途
html	ホームページを表示
JavaScript	ホームページにてグラフを表示
Python	Grove Pi+を動作
SQL	データベースにデータを書き込み
PHP	動的なホームページを表示

5　習得しておくと望ましいユニット

5.1　訓練用機材を作成するにあたり

パネルの作成を行うには制御盤を作成したことがあれば問題ありません。

ユニット番号	ユニット名	詳細
EU302-0010-1	ＰＬＣ制御（基本）	PLC の動作を習得
EU302-0020-2	ＰＬＣ制御（応用）	
EU302-0150-2	ＰＬＣ制御（プログラマブル表示器１）	タッチパネルの動作を習得
EU107-0450-1	制御盤の仕様	制御盤作成方法を習得
EU107-0440-1	制御盤製図	
EU501-1010-1	ＣＡＤ（基本操作）	図面を作成・編集を習得

5.2　機材を使用して訓練を行うにあたり

前提知識として HTML、JavaScript はある程度読み書きできれば構いません。また Raspberry Pi を使用しますので Linux の基本操作は習得する必要があります。大きくプログラムを変更するのであれば Python や PHP の書き方を習得することをお勧めします。

ユニット番号	ユニット名	詳細
IU201-X012-2	PC-UNIX 基本操作	Raspberry Pi は Linux ベースなので操作法を習得
IU201-X013-3	PC-UNIX システム管理	
IU703-1111-2	LAN 構築（基礎）	Raspberry Pi とパソコンの通信の仕方を習得
IU704-X081-1	HTML 技法（基本）	収集したデータを Web ブラウザに表示する方法を習得
IU704-X082-2	HTML 技法（ＣＧＩ）	
IU704-1112-3	Web コンテンツ作成 2(JavaScript)	
IU601-X031-2	データベース基本１	データベースに接続、登録、データの取得する手法を習得
IU601-X032-2	データベース基本２	
IU499-1021-3	DB システム構築(基本設計)	データベースを構築する手法を習得
IU499-1022-3	DB システム構築（詳細設計）	
IU499-1023-3	DB システム構築（実装）	

5.3 実例

富山職業能力開発促進センターで行っている IoT 生産システム科（正式名称：スマート生産サポート科）のカリキュラムを提示します。（2020/7/1 現在）

このシステムを使用しているユニットは「応用課題（課題名：生産支援ネットワークシステム）」です。またシステムで必要となる要素は色を付けております。Raspberry Pi は PC-UNIX とサーバ構築で使用しております。

ユニット番号	ユニット名
EU302-0010-1	ＰＬＣ制御（基本）
EU302-0020-2	ＰＬＣ制御（応用）
EU302-0330-2	ＰＬＣ制御（ネットワーク）
IU703-1111-2	ＬＡＮ構築（基礎）
IU703-0012-2	ＬＡＮの利用
EU403-X210-3	事業所内ネットワーク構築（ルーティング）
IU201-X012-2	PC-UNIX 基本操作
IU201-X013-3	PC-UNIX システム管理
IU704-1101-3	インターネットサーバ構築１（DNS）
IU704-1102-3	インターネットサーバ構築２（FTP、Web、メール）
IU703-X031-2	ネットワークシステム構築１
IU703-X032-2	ネットワークシステム構築２
EU302-0150-2	ＰＬＣ制御（プログラマブル表示器１）
EU302-0220-2	ＰＬＣ制御（プログラマブル表示器２）
EU302-0130-2	ＰＬＣ制御（位置決め１）
EU302-0070-2	PC 制御（電動機運転）
EU302-X300-2	ＰＬＣ制御（インバータ）
	応用課題（課題名：省電力生産システム構築）
IU304-1011-2	Java プログラミング 1（基本構文）
IU304-1012-3	Java プログラミング 2（クラス）
IU304-1013-3	Java プログラミング 3（先進機構）
IU304-X014-4	Java プログラミング 4（API）
IU304-X015-4	Java プログラミング 5（GUI）
IU304-X200-3	生産支援システムのための開発ツール（コレクション編）
EU307-X020-3	多機能通信端末ﾃﾞﾊﾞｲｽ制御（U/I） 1
EU307-X030-3	多機能通信端末ﾃﾞﾊﾞｲｽ制御（U/I） 2
EU307-X040-3	多機能通信端末ﾃﾞﾊﾞｲｽ制御（ﾌﾟﾗｯﾄﾌｫｰﾑﾌﾚｰﾑﾜｰｸ）

EU307-X050-3	多機能通信端末ﾃﾞﾊﾞｲｽ制御（ﾈｯﾄﾜｰｸ）
EU307-X060-3	多機能通信端末ﾃﾞﾊﾞｲｽ制御（センサ）
	応用課題（課題名：ﾀﾌﾞﾚｯﾄ端末による入出力制御実習）
IU704-X081-1	HTML 技法（基本）
IU704-X082-2	HTML 技法（CGI）
IU704-1112-3	Web コンテンツ作成２（Java Script）
EU302-0190-2	ＰＬＣ制御（センサ）
EU302-X290-2	ＰＬＣ制御（ﾌｨｰﾙﾄﾞﾈｯﾄﾜｰｸ）
	応用課題（課題名：生産支援ﾈｯﾄﾜｰｸｼｽﾃﾑ）

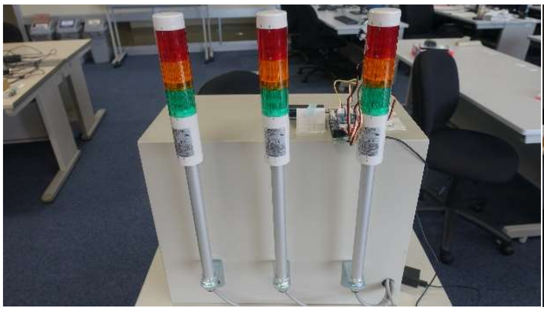
3 日間で習得できる内容は限られるため、指導員側で用意したプログラムを受講生が変更しながら作成しています。

6　　制作方法

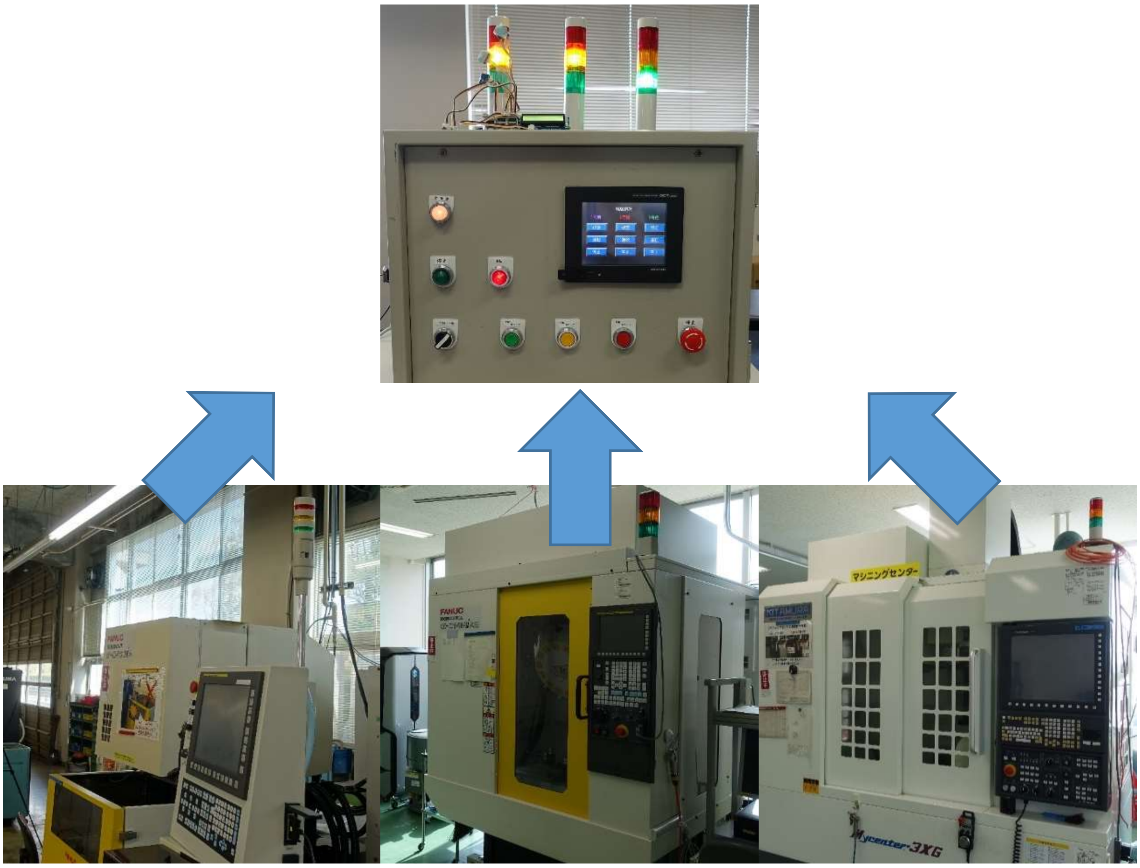
6.1　　作り方（パネル）

制御盤製作技術[B304-001-A]や制御盤設計・製作技術[B304-002-A]で作成する時に選定しているボックスを使用しています。以前、生産システム技術科で使用していたボックスを再利用しました。制御盤に関する関連知識等は制御盤製作[ES353]を参照してください。

穴位置等は Jw-CAD で図面にしております。現物写真とともに位置調整・確認をしながら作成してください。（配布データを参照してください）




ランプは 3 灯用を 3 つ使用していますが、センサを 3 つしか用意しておりませんので、3 灯をすべて収取したい場合はセンサを増やしてください。ランプ 1 灯あたりセンサ 3 個 Raspberry Pi　1 台としており、追加購入が必要です。機材のイメージとしては以下の写真の通りです。

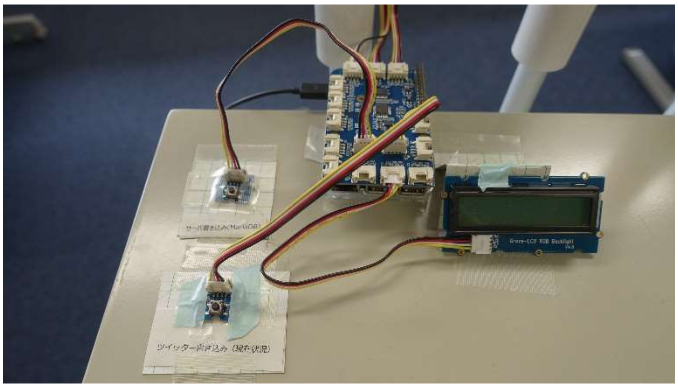


動作はパネルのボタンを押した場合は点灯のみで、タッチパネルにて連続作動・個別動作時間を規定しています。Raspberry Pi で動作するプログラムの作り方によりラダー図は変更してください。またセンサの調整時はボタンでのランプ点灯がやりやすいと思います。

6.2　作り方（Raspberry Pi）

Raspberry Pi では Grove Pi+を使用し、センサ 3 つとボタン 2 つと LCD を接続します。センサと Raspberry Pi とセンサの接続は電子回路の知識と細かい設定などを考えながらはんだ付けを行う必要があったりとしますが、Grove Pi+を使用することにより接続を簡単に行えます。接続箇所はプログラムと写真を参考にしてください。またサンプルプログラムと import 用プログラムは GitHub を参照してください。

サーバへの書き込みをボタンで行うことにより連続動作での書き込み数増大を防いでいます。現場で使用する場合はランプの点灯か消灯で書き込みを行うよう変更してください。また外部インターネットに接続できるのであれば、Twitter の API を使用することによりボタンを押すことにより Tweet するようにもできます。Grove Pi+は C#でも使用できるので Raspberry Pi にインストールする OS を Windows 10 IoT core にすることにより C#でも開発できます。



6.3　作り方（パソコン）

パソコンでは Xampp を使用して Apache2 と MariaDB と PHP をインストールしています。個別でもインストールは可能ですが、管理などの面から統合ソフトをインストールすると楽になります。またアドオンで WordPress なども扱えるため訓練対応幅も増えます。

Apache2 は Web サーバで PHP は SQL 構文を使用し、動的にホームページを表示させるために使用しています。

Raspberry Pi に機能をすべて集約して動作させることも可能ですが、多重アクセスを想定して作成しているため、パソコンにてデータの保存と表示を作成しています。また訓練コースのカリキュラムに SQL サーバ、PHP の取り扱いがないので基礎プログラムは講師側で用意しております。

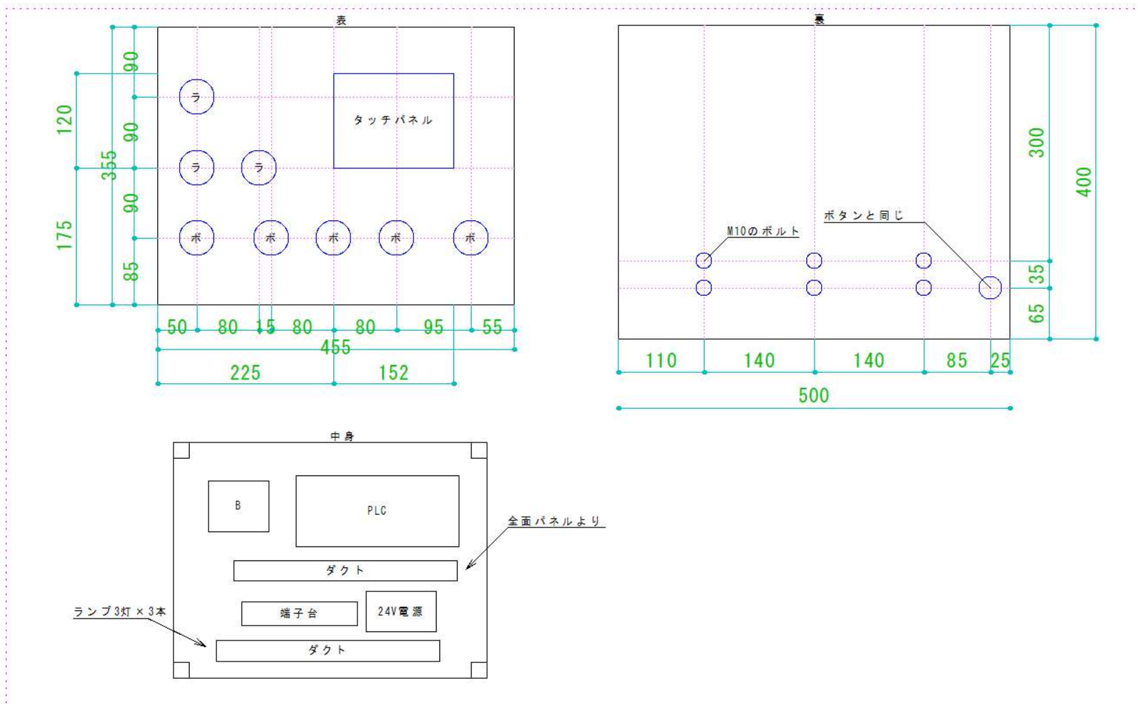
PHP と SQL を使用してデータベースに接続し、JavaScript でグラフを表示しており、グラフには Google Chart API と Chart.js を使用しております。外部インターネットに接続しない場合は表示できません。

6.4　配布データ

使用したデータを同梱しております。プログラムは Word データからコピーを行うと改行の関係で動かない場合がありますので配布データを参照してください。テキストエディタは Visual Studio Code を推奨しています。

内容	対応ソフト
パネル図面	Jw-CAD
ラダー図	GX Developer か GX Works
ホームページ PHP JavaScript HTML	テキストエディタ
ランプ状況取得 Python	テキストエディタ

6.4.1　CAD 図面



6.4.2 PLC 入出力使用接点

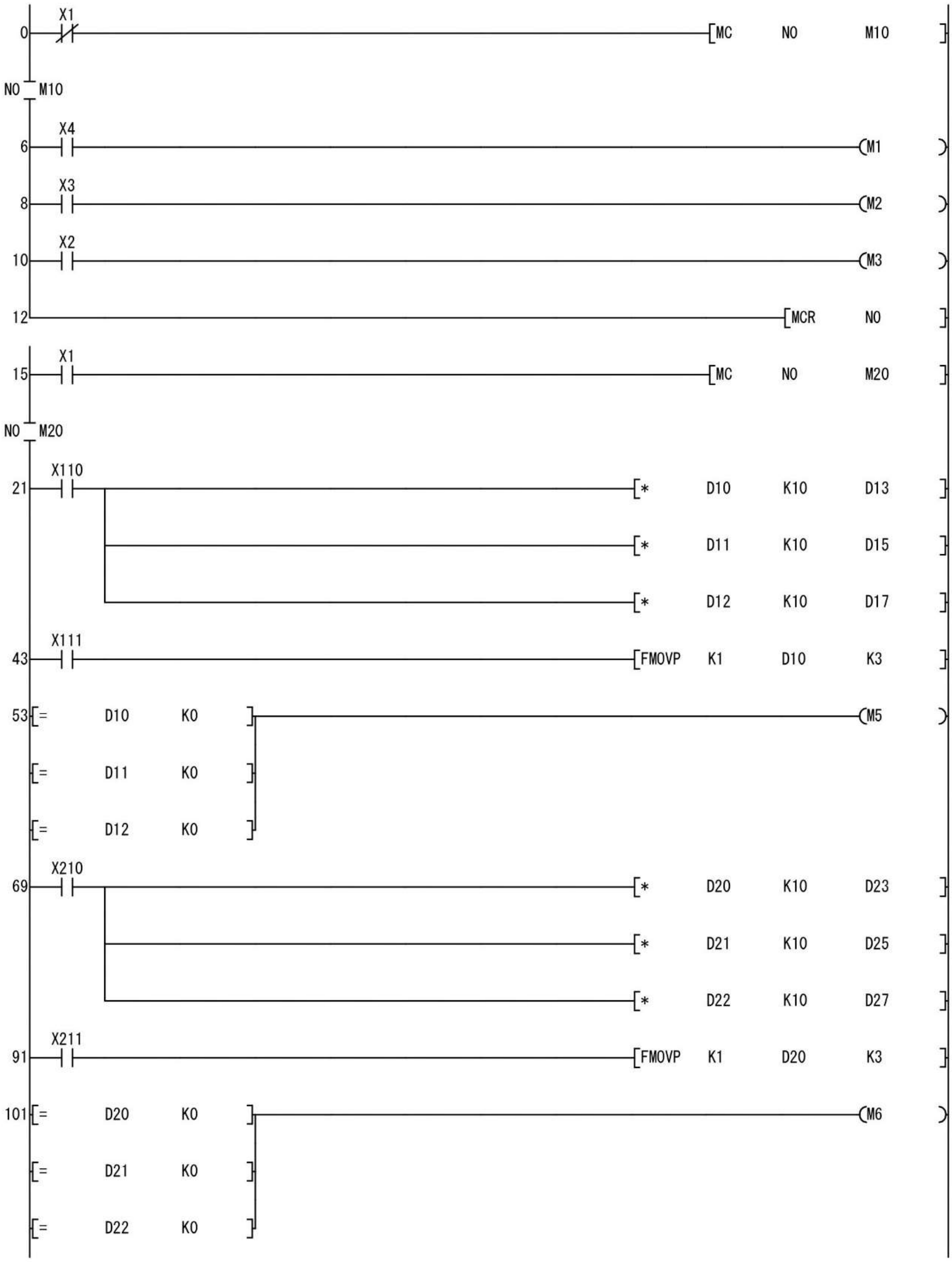
入力

接点番号	概要
X0	緊急停止ボタン
X1	手動・自動切換え
X2	手動　緑ボタン
X3	手動　黄ボタン
X4	手動　赤ボタン

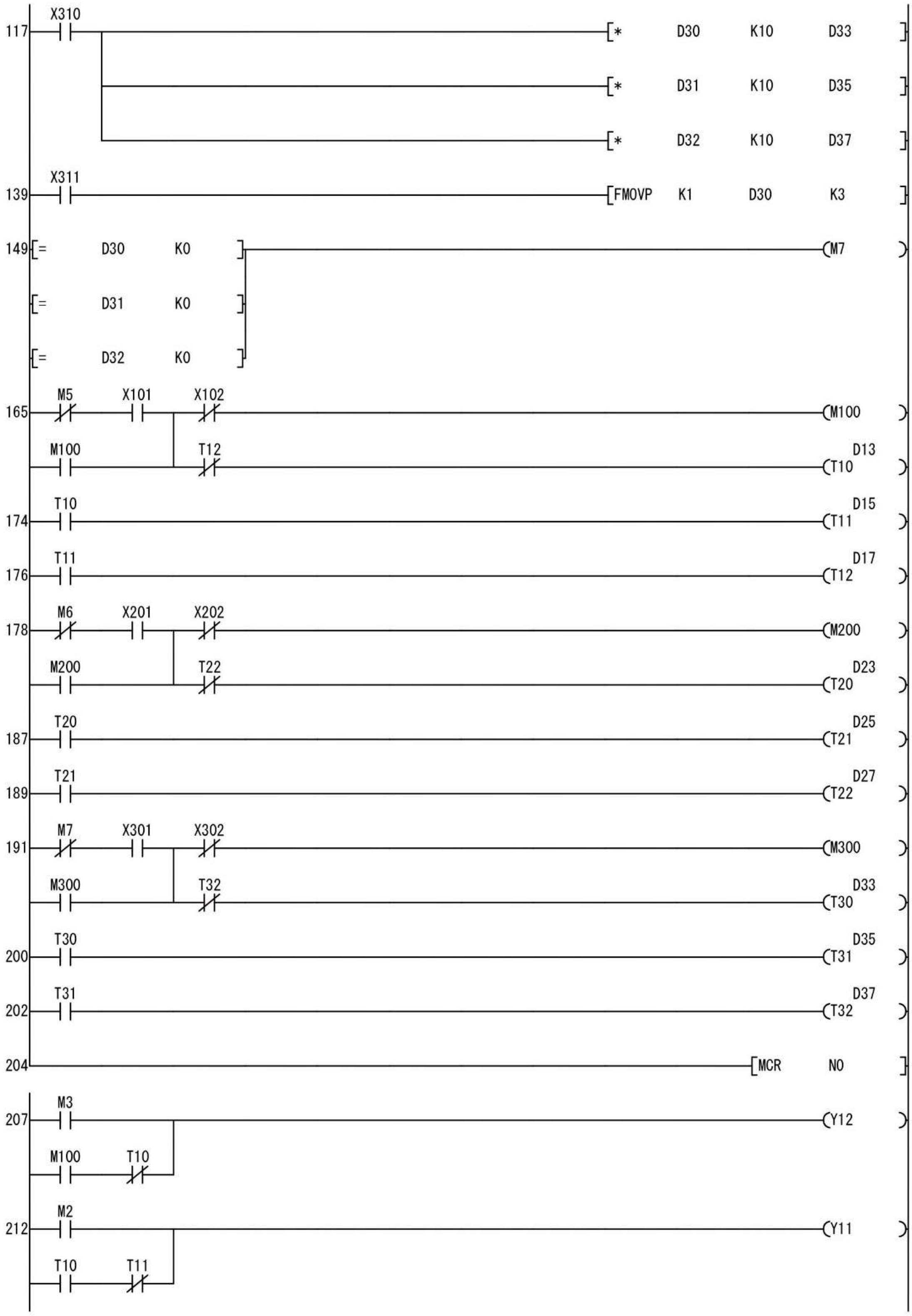
出力

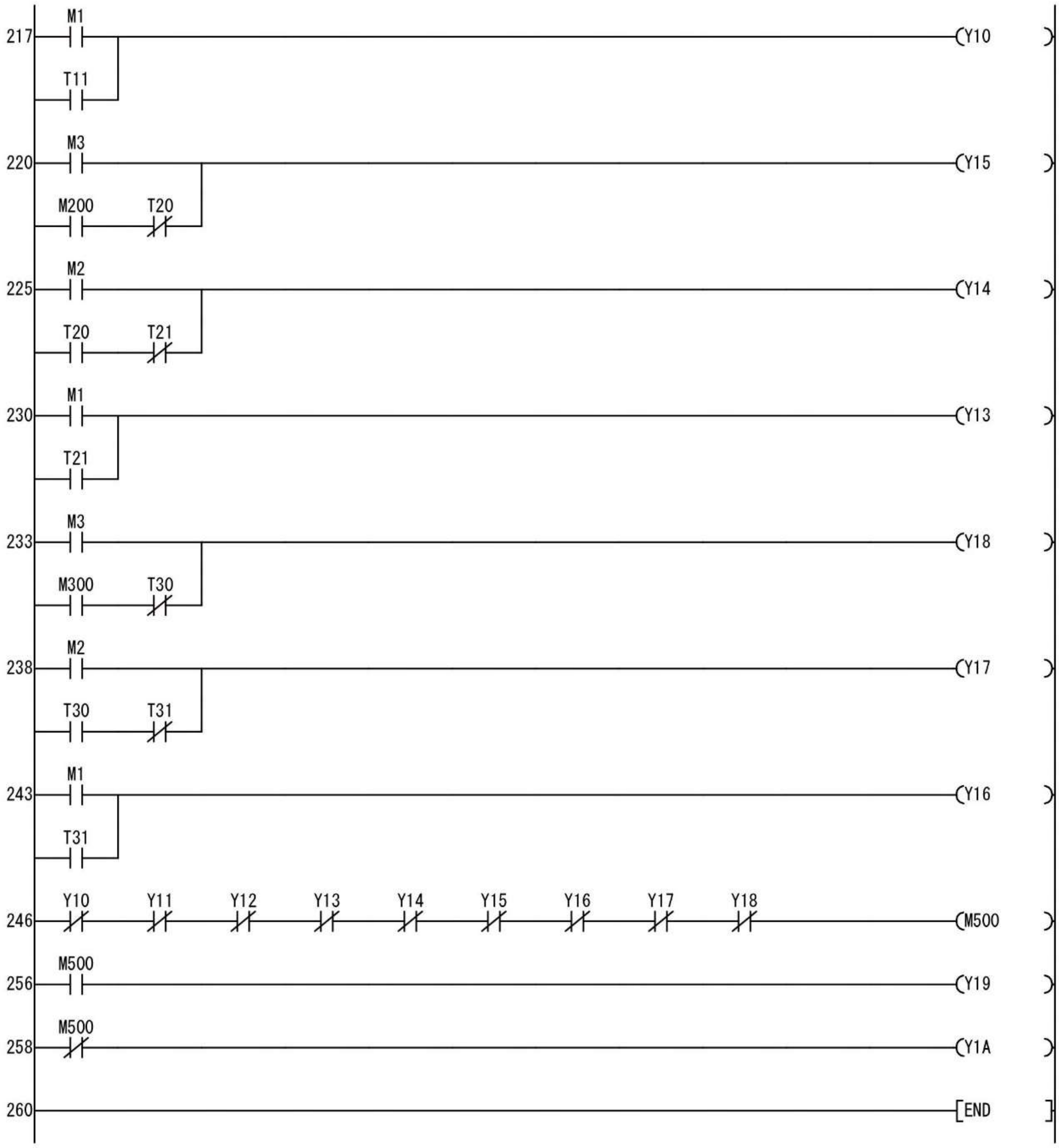
接点番号	概要
Y10	左ランプ　赤
Y11	左ランプ　黄
Y12	左ランプ　緑
Y13	中ランプ　赤
Y14	中ランプ　黄
Y15	中ランプ　緑
Y16	右ランプ　赤
Y17	右ランプ　黄
Y18	右ランプ　緑
Y19	停止ランプ
Y1A	運転ランプ

6.4.3　ラダー図



0 X1 MC N0 M10
N0 M10
6 X4 M1
8 X3 M2
10 X2 M3
12 MCR N0
15 X1 MC N0 M20
N0 M20
21 X110 * D10 K10 D13
* D11 K10 D15
* D12 K10 D17
43 X111 FMOVP K1 D10 K3
53 = D10 K0 M5
= D11 K0
= D12 K0
69 X210 * D20 K10 D23
* D21 K10 D25
* D22 K10 D27
91 X211 FMOVP K1 D20 K3
101 = D20 K0 M6
= D21 K0
= D22 K0


117 X310 [* D30 K10 D33]
[* D31 K10 D35]
[* D32 K10 D37]
139 X311 [FMOVP K1 D30 K3]
149 [= D30 K0] (M7)
[= D31 K0]
[= D32 K0]
165 M5 X101 X102 (M100)
M100 T12 D13 (T10)
174 T10 D15 (T11)
176 T11 D17 (T12)
178 M6 X201 X202 (M200)
M200 T22 D23 (T20)
187 T20 D25 (T21)
189 T21 D27 (T22)
191 M7 X301 X302 (M300)
M300 T32 D33 (T30)
200 T30 D35 (T31)
202 T31 D37 (T32)
204 [MCR N0]
207 M3 (Y12)
M100 T10
212 M2 (Y11)
T10 T11


217 M1 T11 Y10
220 M3 M200 T20 Y15
225 M2 T20 T21 Y14
230 M1 T21 Y13
233 M3 M300 T30 Y18
238 M2 T30 T31 Y17
243 M1 T31 Y16
246 Y10 Y11 Y12 Y13 Y14 Y15 Y16 Y17 Y18 M500
256 M500 Y19
258 M500 Y1A
260 END

6.4.4　タッチパネル表示画面

6.4.5 PHP プログラム(リスト出力)

```
<!DOCTYPE html>
<html>

<head>
    <meta charset="UTF-8">
    <title> 稼働状況</title>
</head>

<body>

    <h1>稼働状況（表）</h1>

    <A href="http://192.168.200.15/graph1/">グラフ１へ</A>
    <A href="http://192.168.200.15/graph2/">グラフ２へ</A>
    <br>
    <table border="2">
        <tr>
            <td>時間</td>
            <td>緑ランプ</td>
            <td>黄ランプ</td>
            <td>赤ランプ</td>
        </tr>

        <?php

        try {

            $dsn = 'mysql:dbname=lamp_status;host=localhost;charset=utf8';
            $user = 'root';
            $password = '';
            $dbh = new PDO($dsn, $user, $password);
            $dbh->setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);

            $sql = 'SELECT * FROM lamp_status WHERE 1';
```

```
            $stmt = $dbh->prepare($sql);
            $stmt->execute();

            $dbh = null;

            while (true) {
                $rec = $stmt->fetch(PDO::FETCH_ASSOC);
                if ($rec == false) {
                    break;
                }
        ?>
                <tr>
                    <td><?php print $rec['time']; ?></td>
                    <td><?php print $rec['Green']; ?></td>
                    <td><?php print $rec['Yellow']; ?></td>
                    <td><?php print $rec['Red']; ?></td>
                </tr>
        <?php
            }
        } catch (Exception $e) {
            print 'ただいま障害により大変ご迷惑をお掛けしております。';
            exit();
        }

        ?>

</body>

</html>
```

6.4.6 PHP プログラム(グラフ 1 出力)

```
<!DOCTYPE html>
<html>

<head>
  <meta charset="UTF-8">
  <title> グラフ表示</title>
</head>

<body>

  <h1>稼働状況（グラフ）</h1>
  <div>
    <A href="http://192.168.200.15/chart/">表へ</A>
  </div>
  <?php

  $dsn = 'mysql:dbname=lamp_status;host=localhost;charset=utf8';
  $user = 'root';
  $password = '';
  $dbh = new PDO($dsn, $user, $password);
  $dbh->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

  $sql = 'SELECT COUNT(*) FROM lamp_status WHERE Green = "ON"';
  $stmt1 = $dbh->query($sql);
  $gr = $stmt1->fetchColumn();

  $sql = 'SELECT COUNT(*) FROM lamp_status WHERE Yellow = "ON"';
  $stmt2 = $dbh->query($sql);
  $yw = $stmt2->fetchColumn();

  $sql = 'SELECT COUNT(*) FROM lamp_status WHERE Red = "ON"';
  $stmt3 = $dbh->query($sql);
  $rd = $stmt3->fetchColumn();

  $dbh = null;
```

```
?>

<script type="text/javascript" src="http://www.google.com/jsapi"></script>
<script type="text/javascript">
  var stmt1 = '<?php echo $gr; ?>';
  var stmt2 = '<?php echo $yw; ?>';
  var stmt3 = '<?php echo $rd; ?>';
  stmt1 = parseInt(stmt1);
  stmt2 = parseInt(stmt2);
  stmt3 = parseInt(stmt3);
  google.load("visualization", "1", {
    packages: ["corechart"]
  });
  google.setOnLoadCallback(drawChartSamplePie);

  function drawChartSamplePie() {
    var data = google.visualization.arrayToDataTable([
      ['ランプ', '回数'],
      ['Green', stmt1],
      ['Yellow', stmt2],
      ['Red', stmt3],
    ]);

    var options = {
      title: '動作回数',
      width: 500,
      height: 500,
      slices: {
        0: {
          color: 'springgreen'
        },
        1: {
          color: 'gold'
        },
        2: {
```

```
              color: 'tomato'
            }
          },
          is3D: true
        };

        var chart = new
google.visualization.PieChart(document.getElementById('sample_pie'));
        chart.draw(data, options);
        var chart = new
google.visualization.BarChart(document.getElementById('sample_bar'));
        chart.draw(data, options)
      }
  </script>
  <div id="sample_pie" style="float:left;"></div>
  <div id="sample_bar" style="float:left;"></div>

</body>

</html>
```

6.4.7 html プログラム(グラフ 2 出力)

```
<!DOCTYPE html>
<html lang="jp">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <script type="text/javascript"
src="https://cdnjs.cloudflare.com/ajax/libs/moment.js/2.18.1/moment.js"></scrip
t>
    <script type="text/javascript"
src="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.9.3/Chart.min.js"></script>
    <title>Document</title>
</head>
```

```
<body>
    <h1>稼働状況表示</h1>
    <?php include("script/Connect.php") ?>

    <p>Machine 1</p>
    <canvas id="chartJs"  width="100" height="50"></canvas>
    <script type="text/javascript">
        var sampleStr1 = JSON.parse('<?php echo $php_json_time; ?>');
        var sampleStr2 = JSON.parse('<?php echo $php_json_Green; ?>');
        var sampleStr3 = JSON.parse('<?php echo $php_json_Yellow; ?>');
        var sampleStr4 = JSON.parse('<?php echo $php_json_Red; ?>');
    </script>
    <script type="text/javascript" src="script/chartJs.js"></script>
    <div>
        <A href="http://192.168.200.15/chart/">表へ</A>
    </div>

</body>

</html>
```

6.4.8 JavaScript プログラム(グラフ 2 出力)

```
var ctx = document.getElementById('chartJs').getContext('2d');
var chart = new Chart(ctx, {
    type: 'line',
    data: {
        labels: sampleStr1,
        datasets: [{
                label: "Green",
                backgroundColor: 'rgb(101,255,140)',
                borderColor: 'rgb(50,255,101)',
                data: sampleStr2,
                steppedLine: true
            }, {
                label: "Yellow",
                backgroundColor: 'rgb(   255,255,101)',
```

```
                borderColor: 'rgb(255,255,50)',
                data: sampleStr3,
                steppedLine: true
            }, {
                label: "Red",
                backgroundColor: 'rgb(255,101,101)',
                borderColor: 'rgb(255,50,50)',
                data: sampleStr4,
                steppedLine: true
            }

        ]
    },
    options: {
        scales: {
            xAxes: [{
                type: 'time',
                scaleLabel: {
                    display: true,
                    labelString: 'Time'
                },
                gridLines: {
                    display: false
                },
                time: {
                    unit: 'minute',
                    unitStepSize: 30,
                    format: "HH:mm:ss",
                }
            }],
            yAxes: [{
                scaleLabel: {
                    display: true,
                    labelString: 'ON/OFF',

                },
```

```
                gridLines: {
                    display: false
                },
                ticks: {
                    min: 0,
                    max: 1,
                    stepSize: 1
                },
            }]
        }
    }
});
```

6.4.9 PHP プログラム(グラフ 2 出力のデータ取得部分)

```
<?php
$dsn = 'mysql:dbname=lamp_status;host=localhost;charset=utf8';
$user = 'root';
$password = '';
$dbh = new PDO($dsn, $user, $password);
$dbh->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
$sql = "SELECT * FROM lamp_status WHERE time BETWEEN '2019/11/06
09:00:00' AND '2019/11/06 17:00:00'";
$stmt1 = $dbh->query($sql);
$stmt1->execute();
$dbh = null;

echo "2019/11/06";

$php_array_time = [];
$php_array_Green = [];
$php_array_Yellow = [];
$php_array_Red = [];
$lampColor = [$php_array_Green, $php_array_Yellow, $php_array_Red];
$accessLabel = ['Green','Yellow','Red'];
array_push($php_array_time, "9:00:00");
```

```
for ($i = 0; $i < count($lampColor); $i++) {
    array_push($lampColor[$i], "0");
}

while (true) {
    $rec = $stmt1->fetch(PDO::FETCH_ASSOC);
    if ($rec == false) {
        break;
    }
    $replace = str_replace('2019-11-06', '', $rec['time']);
    array_push($php_array_time, $replace);
    for($i = 0;$i <count($accessLabel);$i++){
        if (strcmp($rec[$accessLabel[$i]], "ON") == 0) {
            array_push($lampColor[$i], str_replace('ON', '1',
$rec[$accessLabel[$i]]));
        } else {
            array_push($lampColor[$i], str_replace('OFF', '0',
$rec[$accessLabel[$i]]));
        }
    }

}

array_push($php_array_time, "17:00:00");
for ($i = 0; $i < count($lampColor); $i++) {
    array_push($lampColor[$i], "0");
}

$php_json_time = json_encode($php_array_time);
$php_json_Green = json_encode($lampColor[0]);
$php_json_Yellow = json_encode($lampColor[1]);
$php_json_Red = json_encode($lampColor[2]);
```

6.4.10 Python プログラム

```
# 展示品用プログラム
# 自身で用意したサーバか twitter にデータ送信

# -----import 設定-----
import mysql.connector #SQL サーバ接続用
import datetime # 現在時間の取得
import grovepi #grove pi の出力定義
from time import sleep #時間定義（1 秒ずつ）
from twython import Twython #Twitter への書き込み
from grove_rgb_lcd import * #grove pi lcd 使用定義

# -----入出力設定-----
light_sensor1 = 2
light_sensor2 = 1
light_sensor3 = 0
button1 = 5
button2 = 6

grovepi.pinMode(light_sensor1,"INPUT")
grovepi.pinMode(light_sensor2,"INPUT")
grovepi.pinMode(light_sensor3,"INPUT")
grovepi.pinMode(button1,"INPUT")
grovepi.pinMode(button2,"INPUT")

# -----twitter 設定-----
CONSUMER_KEY ='your get key'
CONSUMER_SECRET =' your get key '
ACCESS_KEY =' your get key '
ACCESS_SECRET =' your get key '
api =
Twython(CONSUMER_KEY,CONSUMER_SECRET,ACCESS_KEY,ACCESS_SECRET)

# ----LCD 初期表示-----
setText("LED Check¥nStart")
setRGB(255,255,255)
```

```
sleep(3)

# -----状態取得-----
while True:
    try:
        sensor_value1 = grovepi.analogRead(light_sensor1)
        sensor_value2 = grovepi.analogRead(light_sensor2)
        sensor_value3 = grovepi.analogRead(light_sensor3)

        if  sensor_value1 < 700:
            Green = "OFF"
        else:
            Green = "ON"
            setText("Green")
            setRGB(0,255,0)

        if sensor_value2 < 700:
            Yellow = "OFF"
        else:
            Yellow = "ON"
            setText("Yellow")
            setRGB(255,255,0)

        if sensor_value3 < 700:
            Red = "OFF"
        else:
            Red = "ON"
            setText("Red")
            setRGB(255,0,0)

        sleep(.5)

        if Red == "OFF" and Green == "OFF" and Yellow == "OFF":
            setText("STOP")
            setRGB(255,255,255)
```

```
        #twitter 書き込み
        button_status1 = grovepi.digitalRead(button1)
        if button_status1:
            dt_now = datetime.datetime.now().isoformat(' ', 'seconds')#ボタンが押され
た時間
            out_str = dt_now + "    Green = " + Green + "      Yellow = " + Yellow +
"     Red = " + Red
            print ("twitter: " + out_str)
            api.update_status(status = out_str)
            setText("Send to Twitter¥nPleace wait...")
            sleep(3)

        #サーバ書き込み
        button_status2 = grovepi.digitalRead(button2)
        if button_status2:
            dt_now = datetime.datetime.now().isoformat(' ', 'seconds')#ボタンが押され
た時間
            setText("Send to Server¥nPleace wait...")
            db=mysql.connector.connect(host="192.168.200.15",
user="userpython1", password="password", database="lamp_status")
            cursor=db.cursor()
            add_user = ("insert into lamp_status "
                        "(time, Green, Yellow, Red) "
                        "values (%s, %s, %s, %s)")
            data_user1 = (dt_now, Green, Yellow, Red)
            cursor.execute(add_user, data_user1)# レコード追加
            db.commit()# コミット
            query = ("select * from lamp_status")# クエリ選択
            cursor.execute(query) # レコード取得
            #ターミナル表示
            for (time, Green, Yellow, Red) in cursor:
                print("time:{},  Green:{},  Yellow:{},Red:{}".format(time,  Green,
Yellow, Red))
            cursor.close()#データ破棄
            db.close() #切断
            sleep(3)
```

```
    except IOError:
        print ("Error")
        setText("Error")
```

6.4.11 Raspberry Pi 自動起動プログラム(service 登録)

lamp_system.service

```
[Unit]
Description=do something

[Service]
ExecStart=/usr/bin/python3 /opt/systemd/lamp_system.py

[Install]
WantedBy=multi-user.target
```

service 有効化

```
sudo systemctl enable lamp_system.service
```

7　　動作詳細

7.1　　手順と解説

1.　PLC の起動

　PLC の電源を入れ、タッチパネルにてランプを連続動作させせます。秒数は 3 秒ごとの設定にします。Pyhton のプログラム周期が 0.5 秒ほどの更新となっているため 1 秒以上の設定をお勧めします。（Python プログラム sleep(.5)の部分）

2.　Raspberry Pi の起動と書き込み

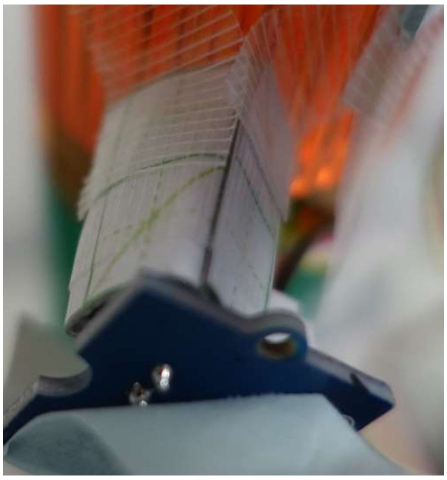
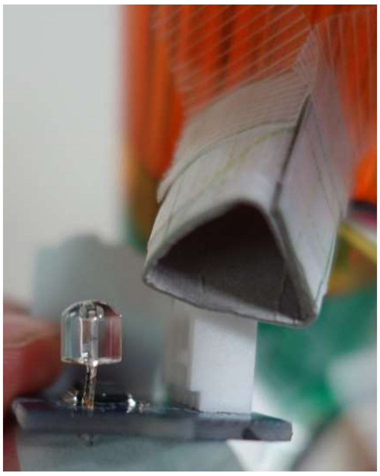
　Raspberry Pi を起動し LCD に「LED Check Start」と表示されたらプログラムが自動起動しています。service での自動起動のため操作はいりませんが、起動しているか確認する場合は Raspberry Pi の LX Terminal にて

```
sudo systemctl status lamp_system.service
```

を入力し、確認を行います。

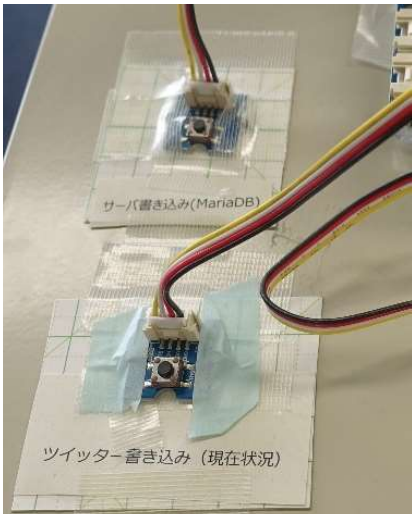
　ランプが光っている色に応じて LCD に「Green」「Yellow」「Red」どの色にも反応していない場合や、ランプが光っていない場合は「STOP」と表示されます。

光の強さを取得しているので色では判断していません。またセンサが基盤からすこし出ているため厚紙で筒を作りランプにテープで貼り付けています。光の強さは Python プログラムの「sensor_value < 700」の部分の数値で制御しています。そのままでは近接するランプや自然光で反応してしまいます。

Raspberry Pi に接続されたボタンを押すことによりサーバに SQL 構文を送り、書き込みを行います。接続は「db=mysql.connector.connect(host="192.168.200.15",user="userpython1",password="password", database="lamp_status")」で行っています。用意したサーバの内容に書き換えて使用してください。

Twitter に書き込む場合は Twitter のアカウントを作成し、API の利用申請(英語での審査)を行い API キーとトークンを取得してください。



3.　サーバの起動とデータの確認

Xampp から Apache と MySQL を起動します。常時起動でも構いません。ボタンを押した後、Web ブラウザにて書き込まれているか確認を行います。データベースには時間と各ランプの ON か OFF が書き込まれています。稼働状況（表）ではすべてのデータを表示しています。再読み込みをすると最終行に新しいデータが表示されます。

グラフ 1 は Google Chart API を使用して円グラフとバーチャートを表示しています。サーバ接続に PHP を使用し、グラフ表示には JavaScript を使用しています。すべてのデータから各ランプの ON の個数をカウントし表示に使用しています。PHP と JavaScript では変数の使用方法が違うため「var stmt1 = '<?php echo $gr; ?>';」の部分で受け渡しを行っています。

時間	緑ランプ	黄ランプ	赤ランプ
2019-11-06 10:12:26	ON	OFF	OFF
2019-11-06 10:14:17	ON	OFF	OFF
2019-11-06 10:25:34	OFF	OFF	ON
2019-11-06 10:31:51	OFF	ON	OFF
2019-11-06 10:36:20	ON	OFF	OFF
2019-11-06 11:07:19	OFF	OFF	ON
2019-11-06 11:50:11	ON	OFF	OFF

グラフ 2 では Chart.js を使用して ON と OFF の状態を表示させています。html と JavaScript と PHP を使用していますが別々のファイルにしています。接続部分はグラフ 1 と同様サーバに接続していますがサンプルとして 2019/11/06 分のデータを表示させるように取得しています。JavaScript は Chart.js の内容を記述しています。PHP から JavaScript へデータを受け渡す際に JSON(JavaScript Object Notation)を使用しています。

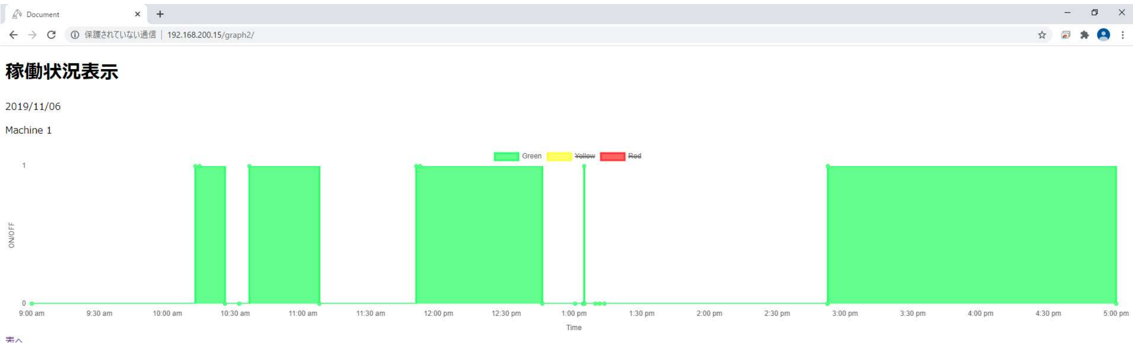
```
$php_json_time = json_encode($php_array_time);
$php_json_Green = json_encode($lampColor[0]);
$php_json_Yellow = json_encode($lampColor[1]);
$php_json_Red = json_encode($lampColor[2]);
```

```
var sampleStr1 = JSON.parse('<?php echo $php_json_time; ?>');
var sampleStr2 = JSON.parse('<?php echo $php_json_Green; ?>');
var sampleStr3 = JSON.parse('<?php echo $php_json_Yellow; ?>');
var sampleStr4 = JSON.parse('<?php echo $php_json_Red; ?>');
```

データベースに ON と OFF で書き込んでいるため、Chart.js の縦軸に対応するよう「str_replace」にて ON は 1、OFF は 0 に変換しています。

```
if (strcmp($rec[$accessLabel[$i]], "ON") == 0) {
                array_push($lampColor[$i], str_replace('ON', '1',
$rec[$accessLabel[$i]]));
          } else {
                array_push($lampColor[$i], str_replace('OFF', '0',
$rec[$accessLabel[$i]]));
```

表示は一斉に各色表示されますが、ラベルを選択することにより表示される内容を制限できます。



7.2　適用方法

7.2.1　ハードウエア

Grove Pi+で接続しているセンサのポートは変更できません。アナログ入力ポートを 3 つすべて使っているためです。LCD は I2C で接続しているため対応ポートなら、どの接続場所でも可能です。ボタンはデジタル入出力であればどのポートでも接続可能です。接続先を変更した場合 Python プログラムの「どの位置を変更しないといけないか」という展開になります。

PLC の動きについては各自で判断し動作させるように展開します。収集する時間や状態に合わせて動作を選択してください。

ランプからの光と自然光などの外的要因との関係をプログラムの数値を変更しながら展開してください。ハード側ソフト側の両方の調整を行う必要があります。

7.2.2　ソフトウエア

Xampp からインストールしデータベースを作成するところから始める場合は時間を要すると思います。説明からデータベース作成まで約 6 時間程度を見込んでおります。対象ユニットや使用できる時間を考えながら展開してください。

Raspberry Pi と Grove Pi+の接続は指導員側が行ったほうがトラブルは少ないと思われます。GitHub

からデータをダウンロードとサンプル起動して確認を行いますが、バージョンが上がると参照場所が変わる可能性があるので今回提示しているプログラムが作動するか確認を行い、その後を提示し展開する方法がいいと思われます。初めから作り動作させるまでに 2 週間ほどかかりました。

プログラムの作成、変更は下記の表を参考に展開してください。共通項目としてサーバの IP アドレス、ID、パスワードは変更、統一してください。今回使用している ID とパスワードはデフォルトの状態で使用しています。専用アカウントを作成して構築したほうがセキュリティは強固になります。

プログラム名	展開内容
PHP(リスト出力)	● CSS などで装飾を施す ● 日付選択で表示
PHP(グラフ 1 出力)	● Google Chart API を参照し、表示したい内容にあったチャートに変更 ● 時間軸で集計し表示
html(グラフ 2 出力) JavaScript(グラフ 2 出力) PHP(グラフ 2 出力のデータ取得部分)	● 日付固定であるため選択フォーム等の設置 ● 1 つのランプ状態表示ではなく 3 つ分の表示 ● 日付対比や曜日対比での表示
Python プログラム	● ボタンではなくランプの状態で書き込み

8　最後に

プログラムは環境や取得したい情報に応じて変更してください。サーバとの接続に使用する ID とパスワードに関しては訓練やデモ作成時は全権限を持っているものを使用したほうがいい場合もありますが、都度適切な設定範囲を決めて使用してください。

IoT を行う上ではオーダーメイドの要素が強く汎用性に欠ける部分が見受けられます。ランプの情報を取得し活用を図るのは多くの工場でも採用されてきており仕組みとしては汎用となってきています。情報の取得や蓄積、表示は作成者によって内容が異なるので今回提示した方法もオーダーメイドに近い感覚で作成しています。粗削りな部分があるとは思いますが、各作成者の基本的プログラムとなり、活用いただければと思います。

8.1　参考 URL

基盤整備センター	https://www.tetras.uitec.jeed.or.jp/
Raspberry Pi	https://www.raspberrypi.org/
Grove Pi+	https://www.dexterindustries.com/grovepi/
GitHub(Grove Pi+)	https://github.com/DexterInd/GrovePi
Xampp	https://www.apachefriends.org/jp/index.html
Apache2	https://httpd.apache.org/
PHP	https://www.php.net/
MariaDB	https://mariadb.org/
phpMyAdmin	https://www.phpmyadmin.net/
Google Chart API	https://developers.google.com/chart
Chart.js	https://www.chartjs.org/
momentjs	https://momentjs.com/

Twitter アカウント	https://twitter.com/ManufactureIot

8.2　参考：富山職業能力開発促進センターで作成したシステム

今回紹介したシステム以外にも様々なアプローチで作成したシステムがあります。参考までに画像を添付します。ご興味がありましたらご連絡いただけると幸いです。

作成システム一覧

基本モデル

ランプ監視システム

ランプの状態をサーバーに書き込みます。

製作者：指導員　市川

システムを見る

展示品

空き状況リアルタイム表示

座席に座っているかをリアルタイム監視しています。
展示品ですのでぬいぐるみで代用しています。
ID:webiopi/PASS:raspberry

製作者：指導員　市川

システムを見る

出退勤管理システム

出退勤の状況を判断します。
また打刻も確認できます。

製作者：令和1年度10･11月生

システムを見る

発表資料

FAライン監視システム

Raspberry Piのカメラを用いて遠隔での監視を行います。
またモニターだけではなく録画もできます。

製作者：令和1年度10･11月生

システムを見る

発表資料

図書館システム

QRコードを使い、貸し借りの状況を判断します。

製作者：令和1年度10･11月生

システムを見る

発表資料

アプリインストール

鋭意製作中

製作者：

システムを見る

発表資料