

UECSによる複合制御実現のための アルゴリズム的実験と実装

○堀本正文¹⁾, 岡安崇史²⁾, 安武大輔²⁾, 平井康丸²⁾, 尾崎剛教³⁾, 古賀正治³⁾

1)九州大学大学院生物資源環境科学府, 〒819-0395 福岡市西区元岡 744

2)九州大学大学院農学研究院, 〒819-0395 福岡市西区元岡 744

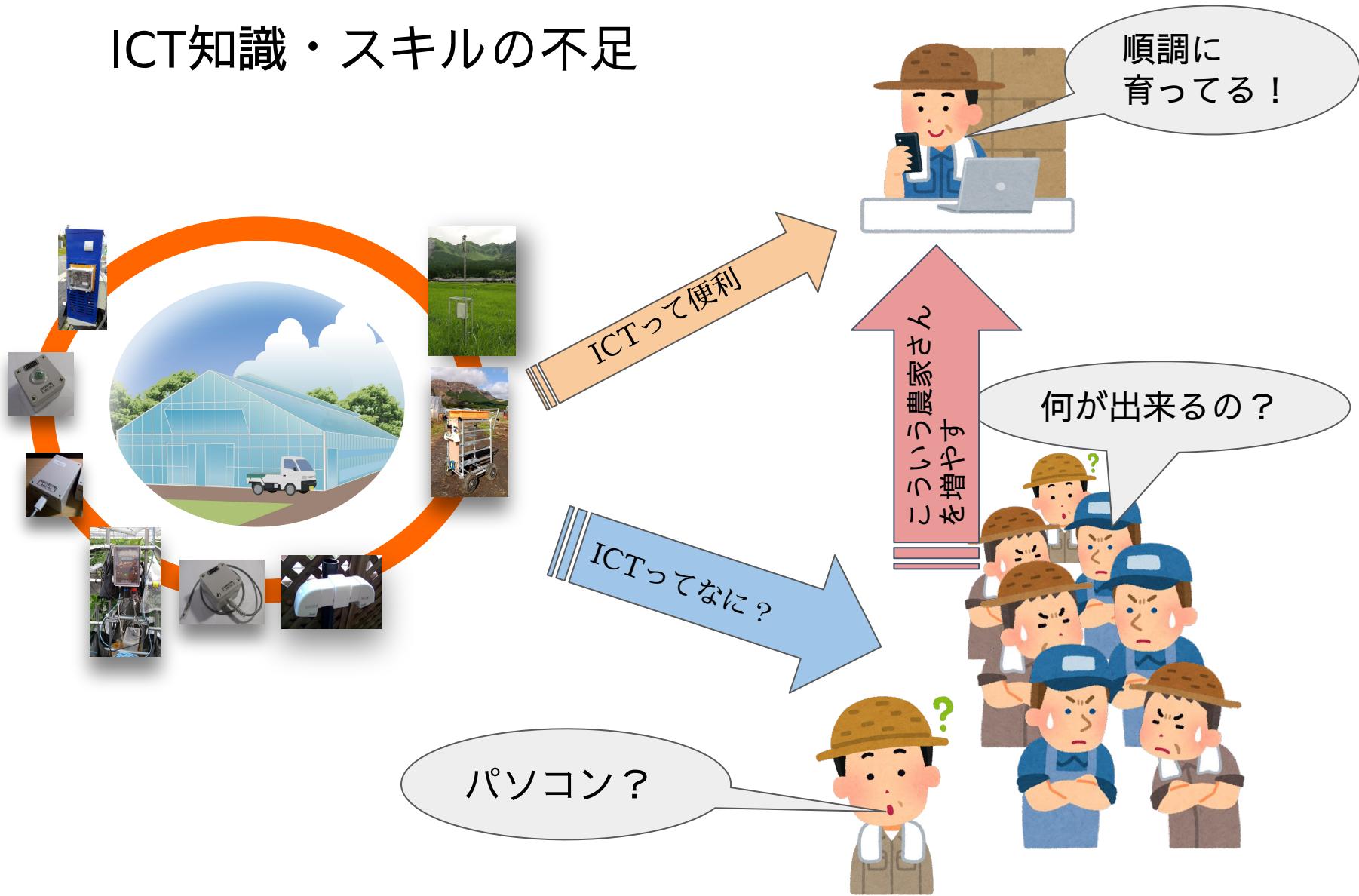
3) 株式会社 welzo 〒812-0013 福岡市博多区博多駅東 1 丁目 14-3



背景 数多くのICT関連機器類

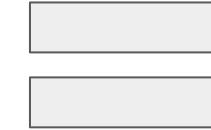
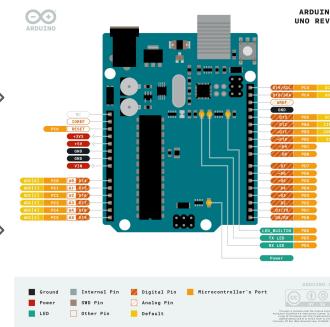
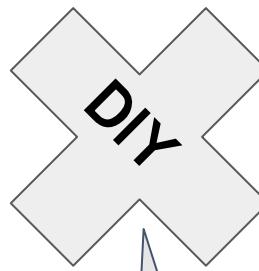


ICT知識・スキルの不足



DIYが実現する目標

安価、容易な入手が可能になったIoTデバイス
しかしながら、それが自在に使える人はひとり

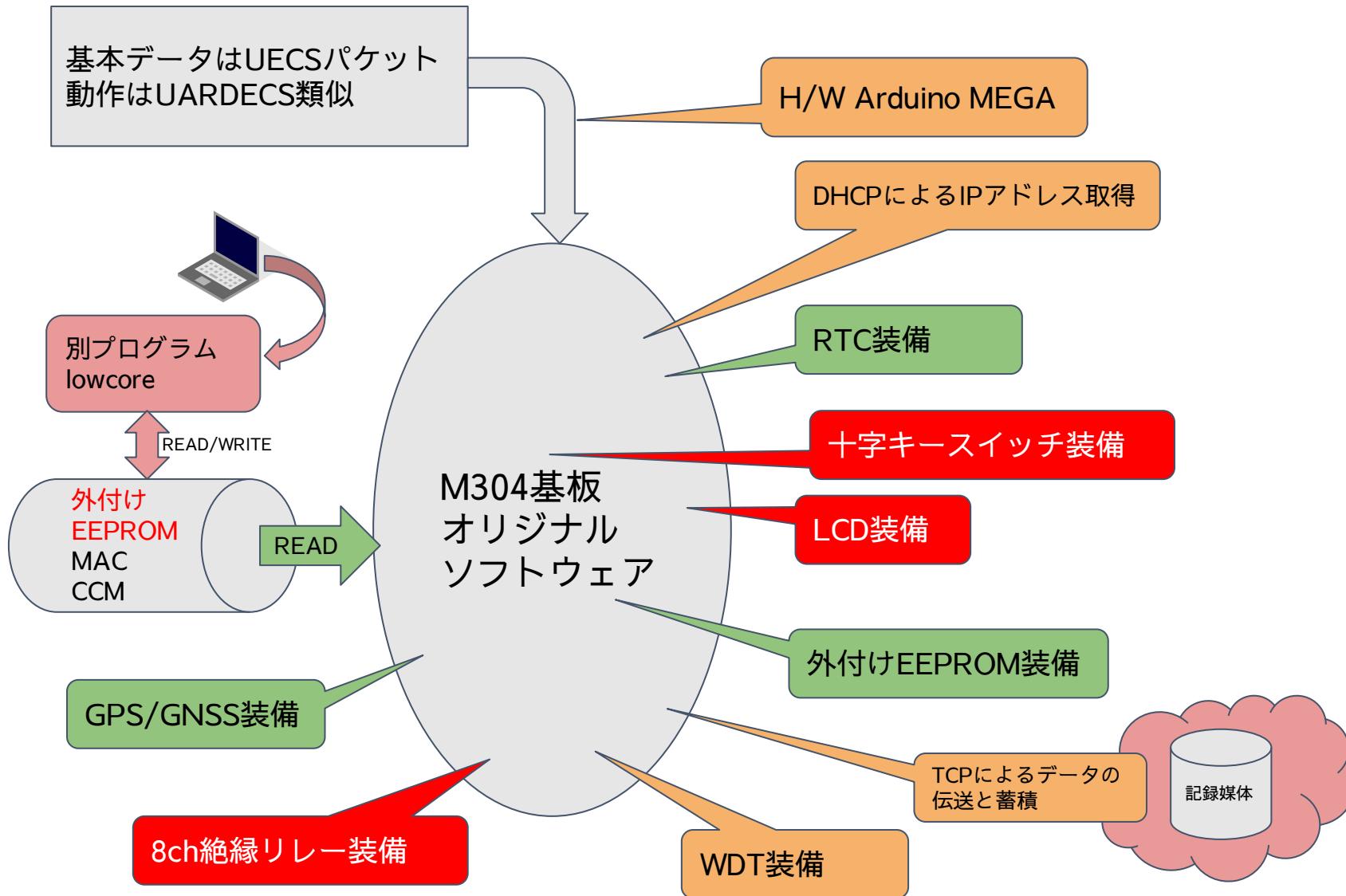


自分の作業経験から
得られる夢や希望

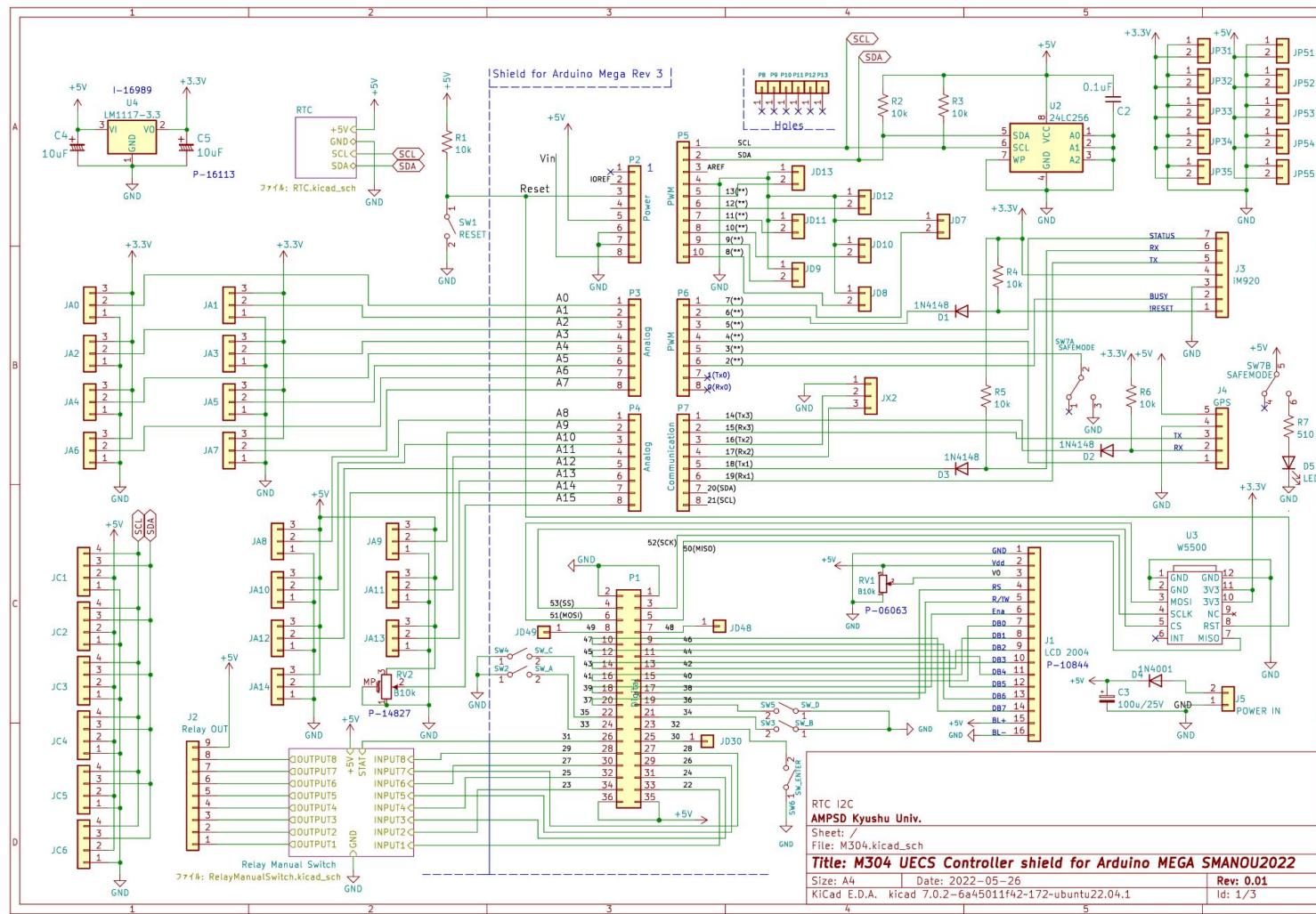
最近入手が容易になった
IoTデバイス・装置類

自らDIYをする気で勉強
原理や事例を学ぶ

結果
希望に近いシステムを構築

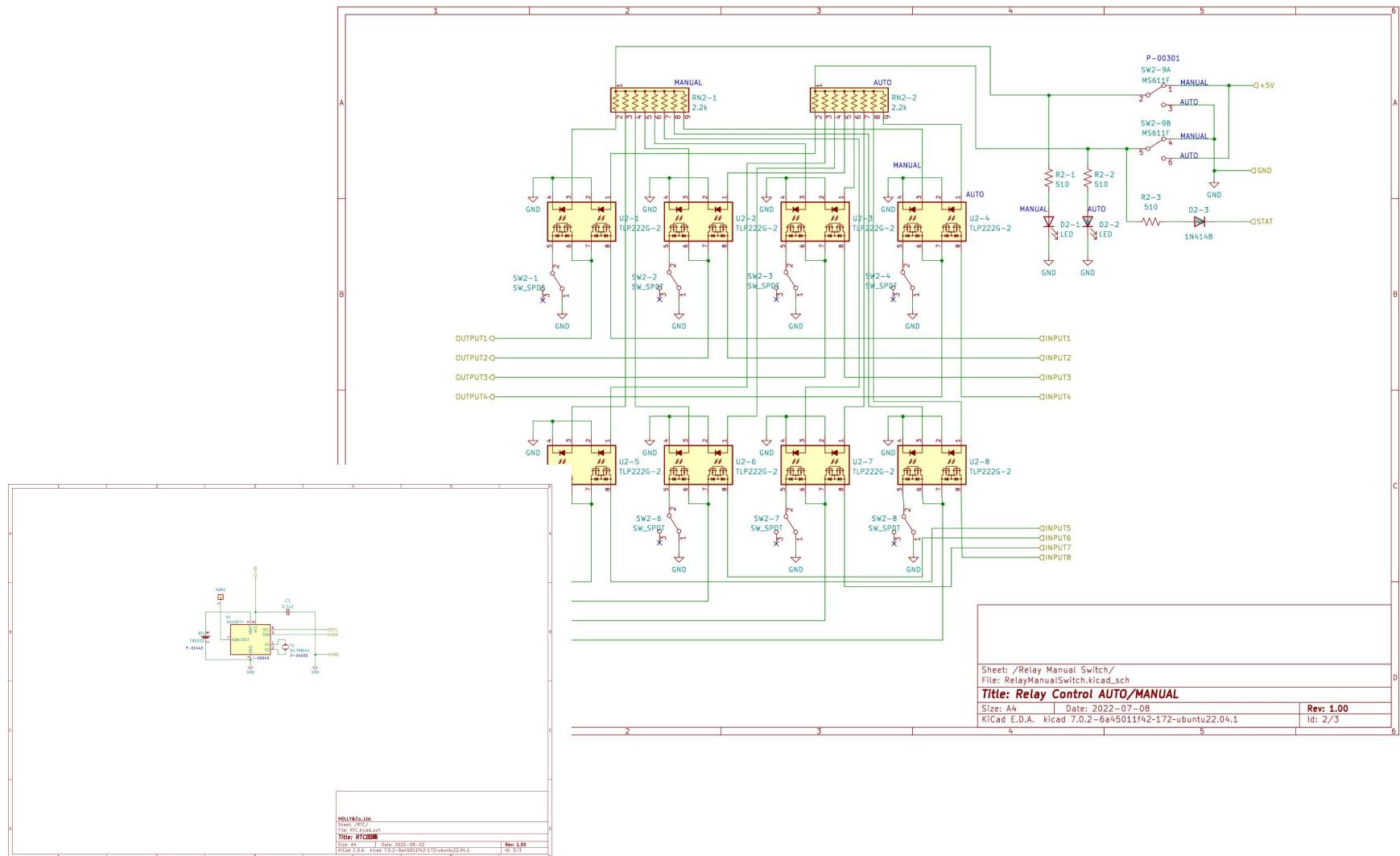


回路図



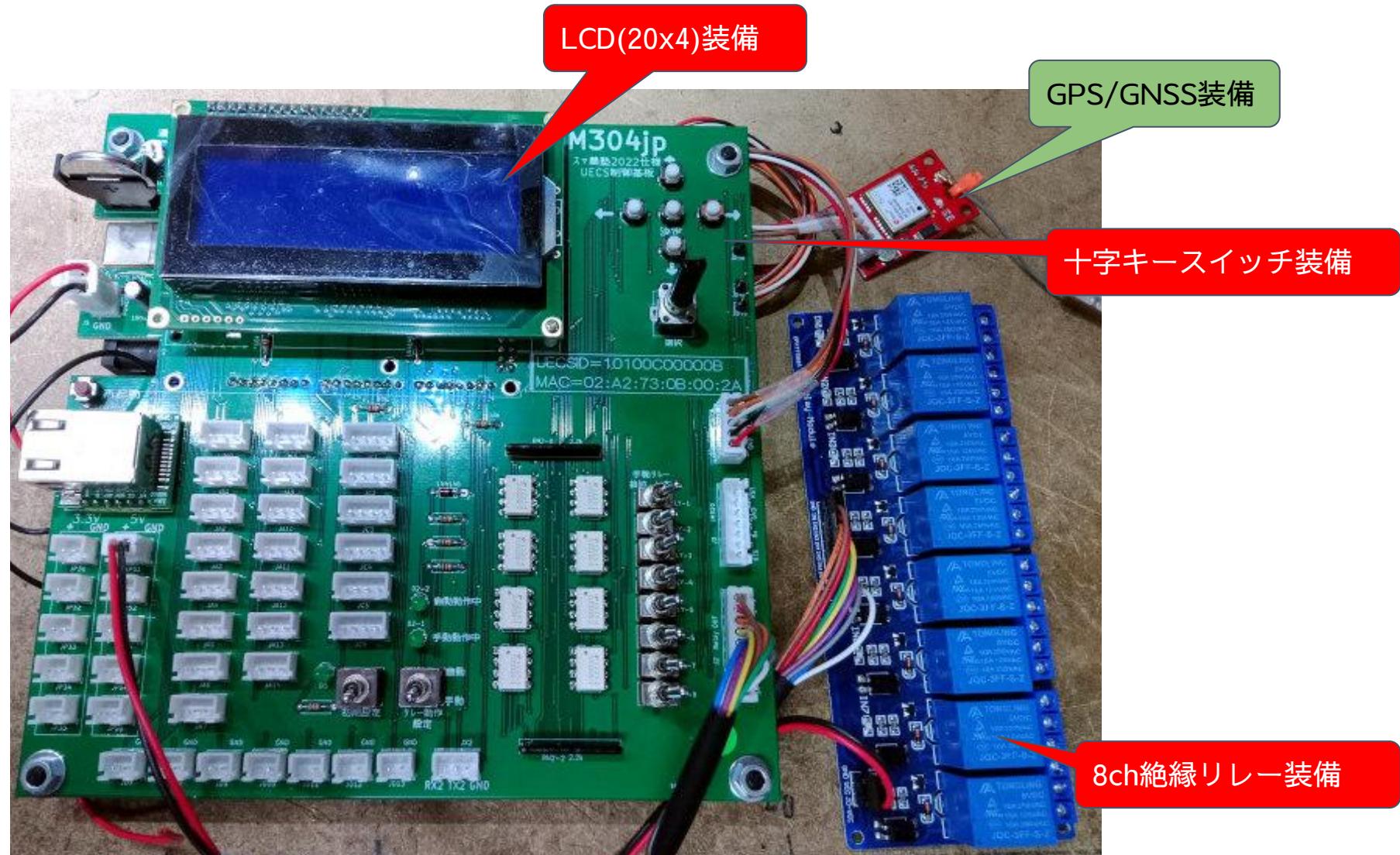
回路図は前述のKiCADで描かれたもので、希望者にはCAMOUTデータを含めて提供可能である。

回路図



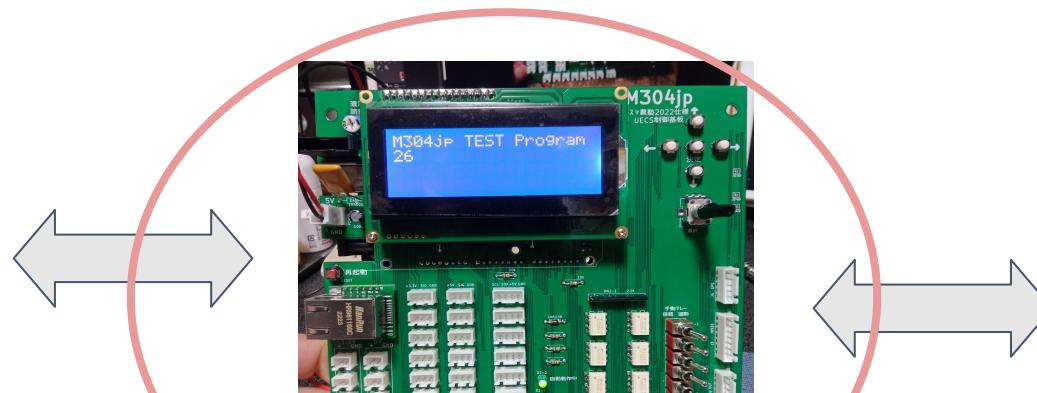
回路図は前述のKiCADで描かれたもので、希望者にはCAMOUTデータを含めて提供可能である。

M304完成図

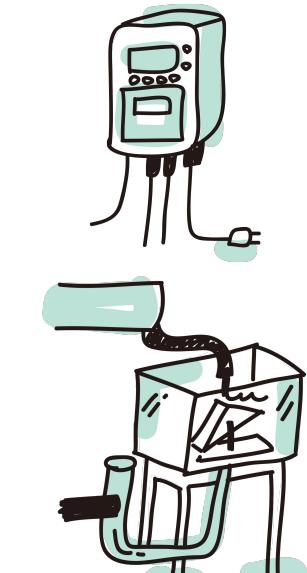


UECS環境観測装置との連携

- 時刻と気象環境データをUECS受電して制御を行う



M304
この部分のソフトウェアを対象



TB2C2
排液量計測

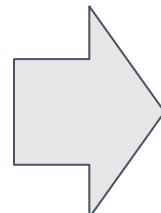
希望するタイミングでの灌水のために

〇〇時〇〇分～△△時△△分までは、〇〇分間隔で△△分水を撒く
気温が〇〇度以上で、湿度が〇〇%未満のときに灌水する。などなど。

このような単純な設定を簡単に行う方法が必要



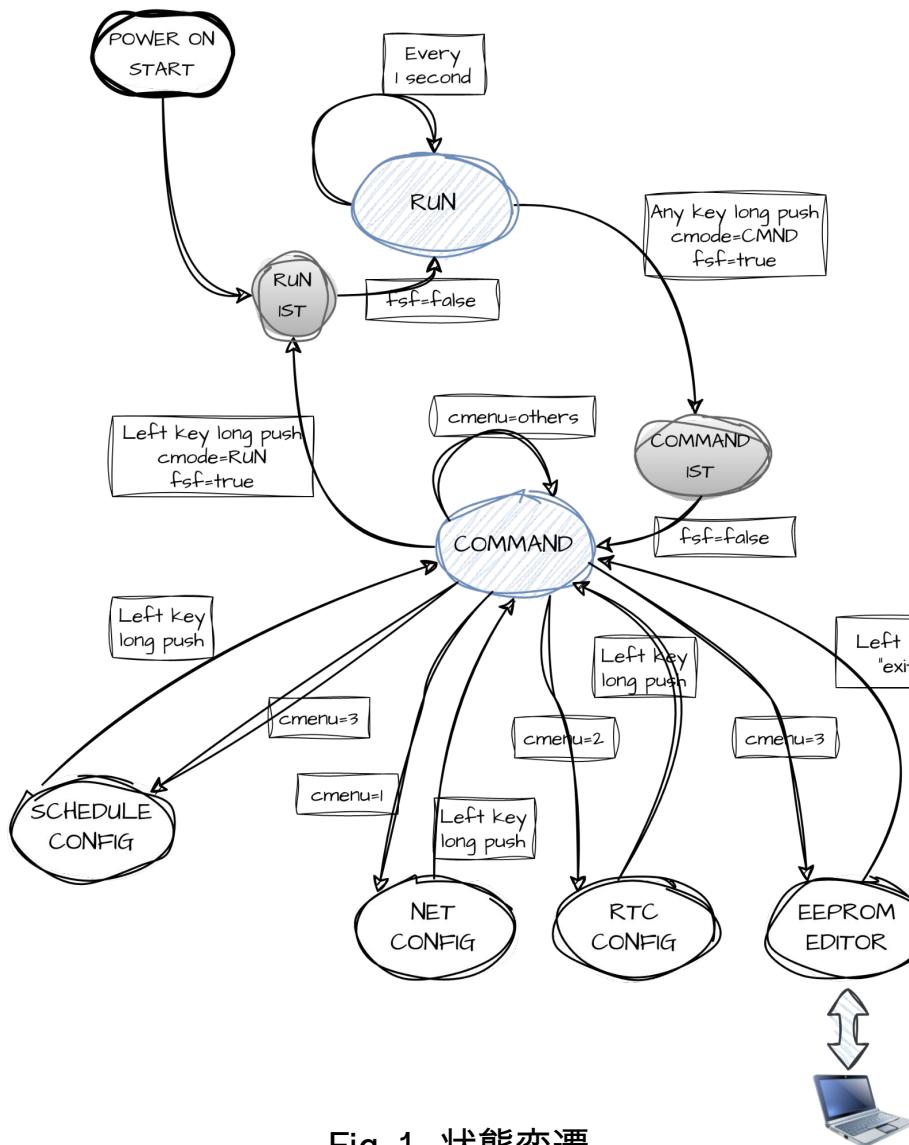
左の十字スイッチとボリュームでLCDを見ながら簡単入力が出来るように、画面・入力インターフェースを開発



ただし、
インターフェースプログラムは少々
難しいので他の方法も考えます。

UDP/Backdoor方式

システムの大まかな流れ



動作状態の変遷を Fig-1.に示します。

SCHEDULE CONFIG

灌水時間を設定するコマンド

NET CONFIG

ネットワーク設定を行うコマンド

RTC CONFIG

RTCの設定を行うコマンド

EEPROM EDITOR

EEPROMの内容を変更するコマンド

RUN

灌水動作実施モード

少し細かい説明

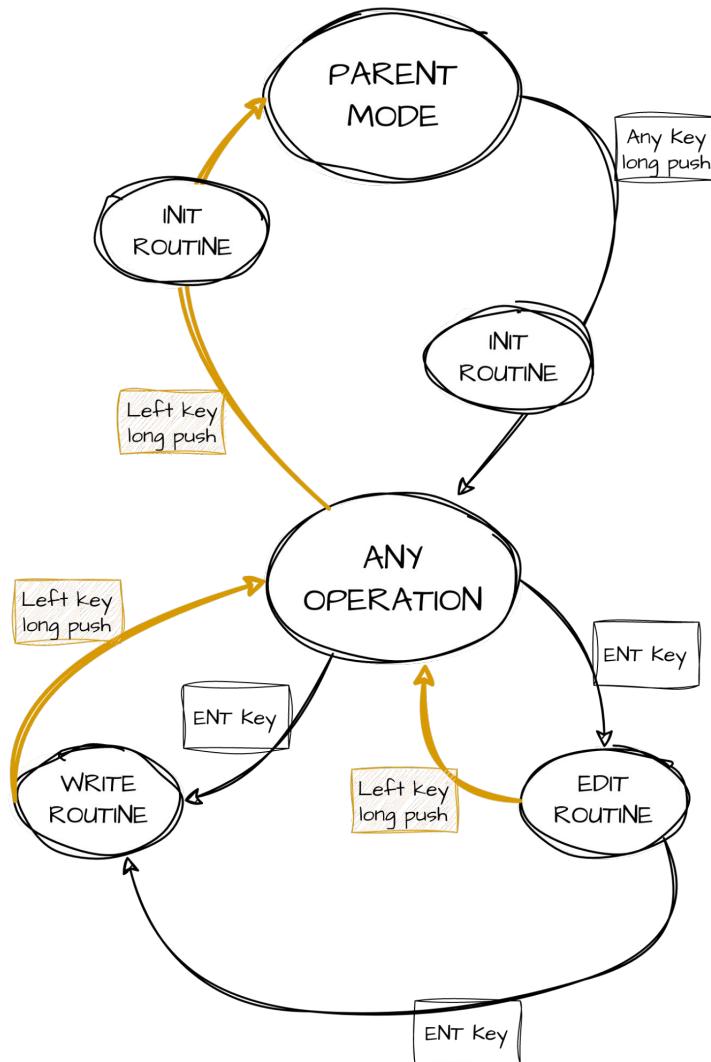


Fig-2. どのように状態遷移するのか

ボタン操作方法のポリシー

統一感を持たせることで次なるメリットがあります。

1. 操作説明が直感的に出来るようになります
状況に寄っては説明書が割愛できます。
2. プログラムの作成が標準化しやすく、バグが入りにくくなります。

「ENTキー」は機能を進めるときに使い、
「LEFTキー」は長押しでメイン画面に戻るときに使います。

タイマー設定内容

データ番号00～99

開始時刻

終了時刻

何分毎に

何分間灌水するか



スイッチングするリレーを指定

```
+-----+  
Set Timer  
02 12:00 16:00 20-01  
RLY:01101000 OK=ENT  
0:BREAK 1:MAKE  
+-----+
```

タイマー設定内容

タイマー設定内容

外つけEEPROMの1000Hから

16バイト/1レコード

総レコード数 100レコード

データ番号02なので
1020H番地から

```
% atdump 10
AT24C256 EEPROM DUMP
ADDR | 00 01 02 03 04 05 06 07 08 0A 0B 0C 0D 0E 0F
+-----+
1000 | FF |
1010 | FF |
1020 | 0C 00 10 00 14 01 FF FF FF FF FF FF FF FF FF | 3C C0 |
1030 | FF |
1040 | FF |
1050 | FF |
1060 | FF |
1070 | FF |
1080 | FF |
1090 | FF |
10A0 | FF |
10B0 | FF |
10C0 | FF |
10D0 | FF |
10E0 | FF |
10F0 | FF |
%
```

何分毎に

何分間灌水するか

複合条件の合致
フラグ

開始時刻

終了時刻

スイッチングするリレーを指定
次ページにて説明

RUNモードでは毎秒このテーブルを参照して合致
した条件でリレーの動作を行います。

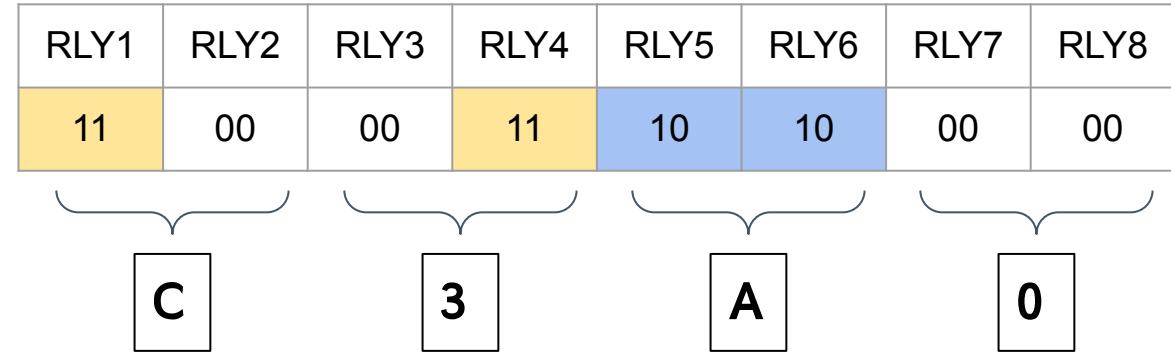
結果・まとめ



対象リレーの設定

8chのリレー 8bitsあれば十分と考えるが、2bits/chを使う
下の2bitsの組み合わせが8ch分=16bitsで構成する

bits	意味
00	DON'T CARE
01	N/A
10	BREAK
11	MAKE



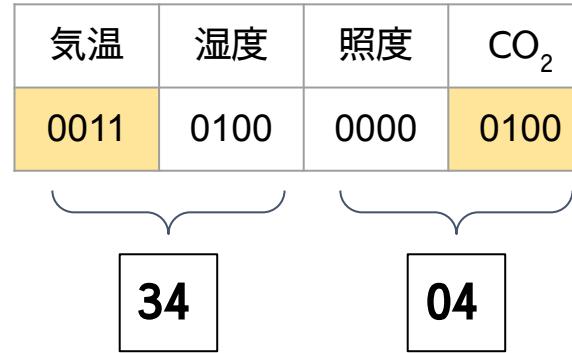
上の設定は、
対象条件に合致したら

RLY1	MAKE
RLY4	MAKE
RLY5	BREAK
RLY6	BREAK
それ以外は	変化なし

複合条件フラグの設定

8bytesを用意する 8bytes X 8bits = 64bits
条件数は4 ⇒ 3bits 複合要因数は4 ⇒ 12bits ← 必要bits数
3bits長は、計算し難いので4bitsにする。MSB側に拡張する。

bits	意味
0000	DON'T CARE
0001	N/A
0010	N/A
0011	HIGH
0100	LOW
0101	P-EDGE
0110	N-EDGE
0111	ALWAY MATCH



上の設定は、対象計測ノード群からのUECS電文の内容が

- 気温が設定値よりも高温
- 湿度が設定値よりも低い
- CO₂濃度が設定値よりも低い

上記の条件にANDで合致した場合に前述のリレー動作を行う。

結果

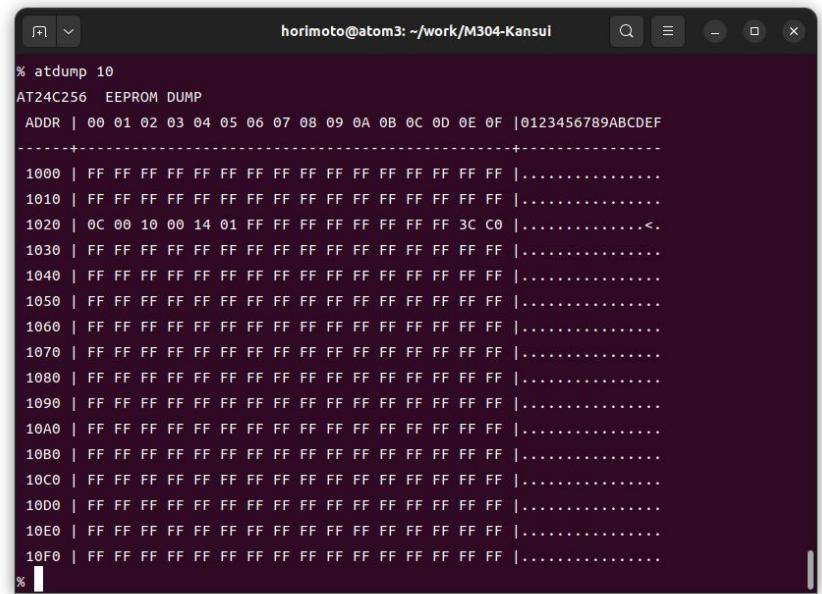
プログラミングコードがたくさん必要

```
//CO2
showValueCO2 = U_ccmList[CCMID_CO2].value;
if(setONOFFAUTO_CO2==2 && U_ccmList[CCMID_CO2].validity ) {
    if (U_ccmList[CCMID_CO2].value < 300) {
        U_ccmList[CCMID_CndCO2].value=1;
        Serial.println("CO2-Under 300");
        co2_gen(1);
        sidewindow(1);
    } else if (U_ccmList[CCMID_CO2].value > 700) {
        U_ccmList[CCMID_CndCO2].value=2;
        Serial.println("CO2-Over 700");
        co2_gen(0);
        sidewindow(2);
    } else {
        U_ccmList[CCMID_CndCO2].value=4;
        Serial.println("CO2-STOP");
        co2_gen(0);
        sidewindow(0);
    }
} else {
    if (setONOFFAUTO_CO2==0) {
        co2_gen(0);
        U_ccmList[CCMID_CndCO2].value=0;
    } else if (setONOFFAUTO_CO2==1) {
        co2_gen(1);
        U_ccmList[CCMID_CndCO2].value=1;
    }
}
showValueStatusCO2 = U_ccmList[CCMID_CndCO2].value;
```

センサ数、条件数に応じて
プログラムを変更する事が必要

本
方
式
を
用
い
る
と

プログラミングコードは決まった比較演算
のみで条件パラメータだけで動作が決まる



```
horimoto@atom3: ~/work/M304-Kansui
% atddump 10
AT24C256 EEPROM DUMP
ADDR | 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F | 0123456789ABCDEF
+-----+
1000 | FF | .....
1010 | FF | .....
1020 | 0C 00 10 00 14 01 FF | .....<
1030 | FF | .....
1040 | FF | .....
1050 | FF | .....
1060 | FF | .....
1070 | FF | .....
1080 | FF | .....
1090 | FF | .....
10A0 | FF | .....
10B0 | FF | .....
10C0 | FF | .....
10D0 | FF | .....
10E0 | FF | .....
10F0 | FF | .....
```

新しいセンサ、条件が出てこなければ
プログラミングはほとんど不要

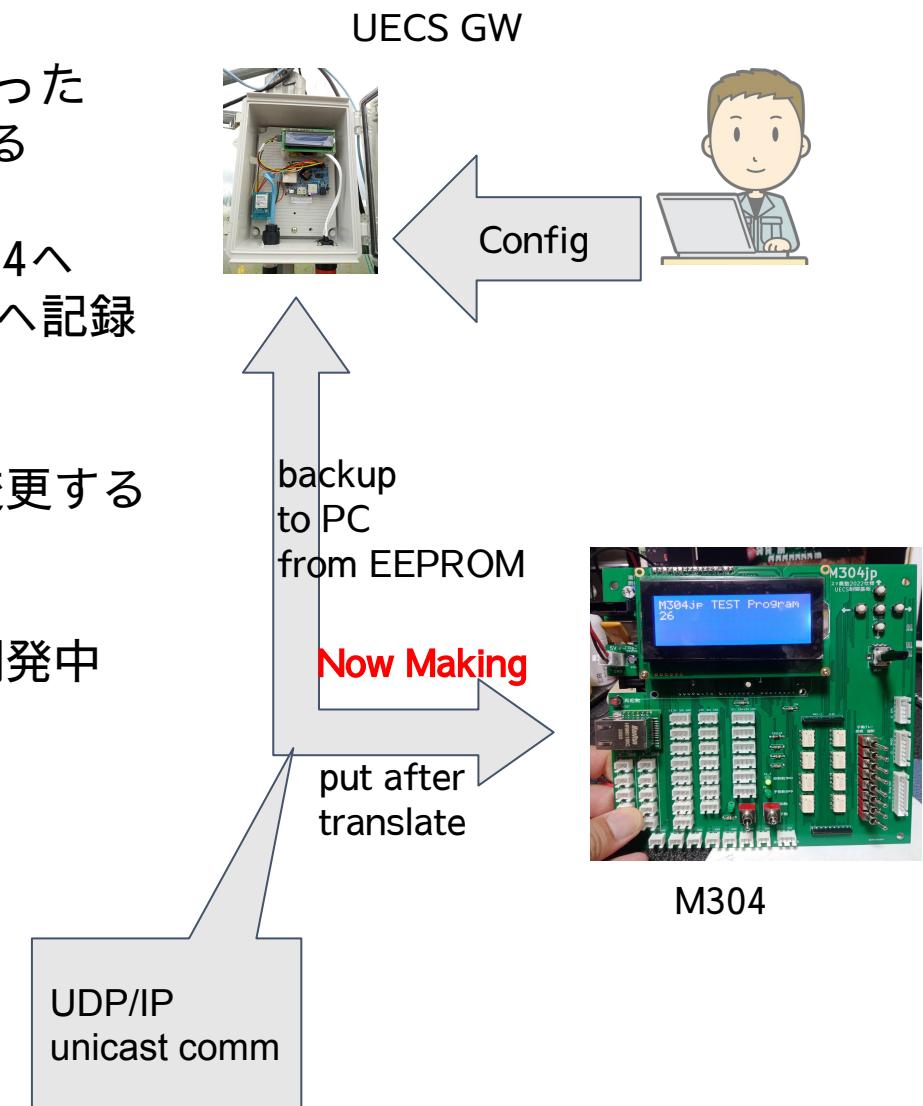
このメモリテーブルを作成するプログ
ラムが別途必要になる
容易に出来るUIの開発も平行して行う
必要がある

既存のUECSノードの設定に用いる
Web I/Fではなく、パワフルな機能を持った
UECS GWによりメモリマップを生成する

生成されたメモリマップイメージはM304へ
UDPを使って転送されM304のEEPROMへ記録
される

プログラムを作り直すこと無く動作を変更する
ことが出来る

現在、それらに関係するプログラムを開発中
大きく3ブロックに分かれる



完

