UECS用ロジック開発ツール
活用マニュアル１

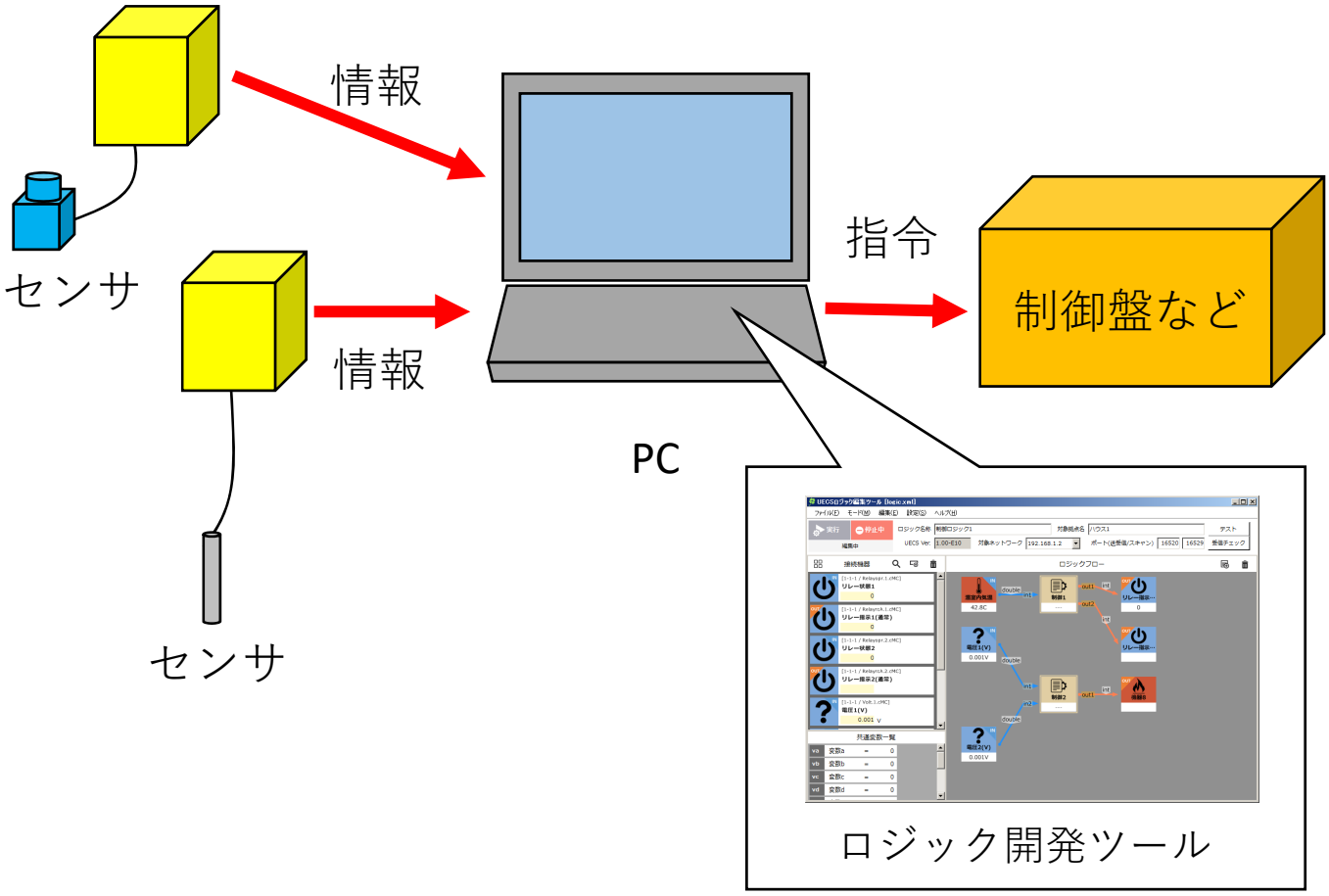
基本操作と機能概説

西日本農業研究センター
2019年3月

目次

UECS用ロジック開発ツールとは

UECS用ロジック開発ツールは、温室内に配置されたセンサから情報を受け取り、様々な条件判断を行った後、制御盤などに指令を与える、という複合環境制御のロジックを作成し、実行するためのソフトウェアです。



日本の施設園芸は作物や栽培技術に地域ごとの独自性があり、独自の環境制御手法を実現したいというニーズがあります。そこで、共通規格「UECS」を活用し、栽培の現場に近いユーザーが簡単な操作で複合環境制御のロジックを作成できることを目的にこのツールは開発されました。

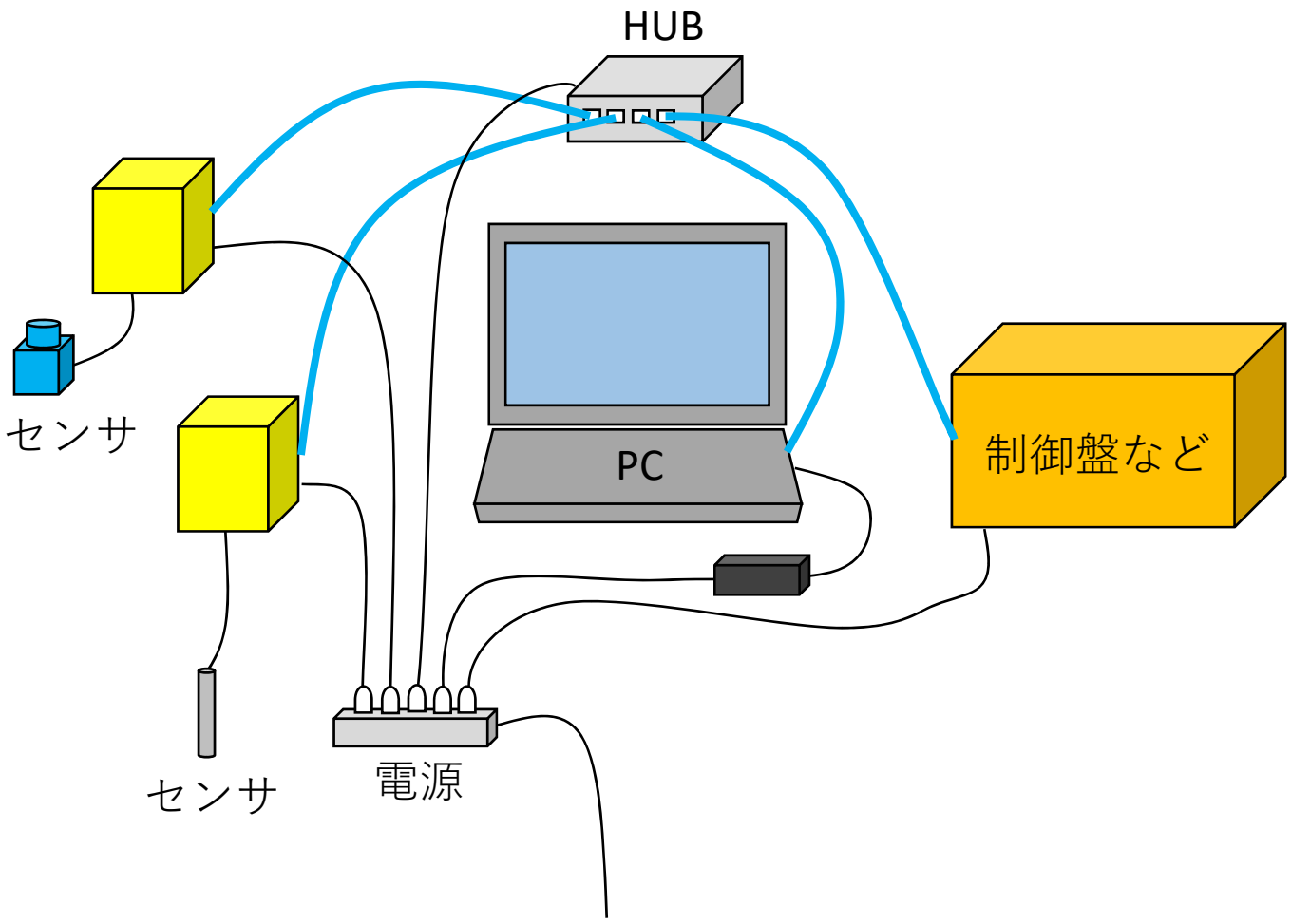
動作環境：

UECSに対応したセンサ、制御盤が必要
Windows 7以降のOSが動作するPC
メモリ 4GB
スクリプト実行中にはPCを常時稼働する必要があります。
PCのIPアドレスは固定して運用することを推奨します。
(スクリプト実行中にIPアドレスが変化すると動作が停止することがあります。)

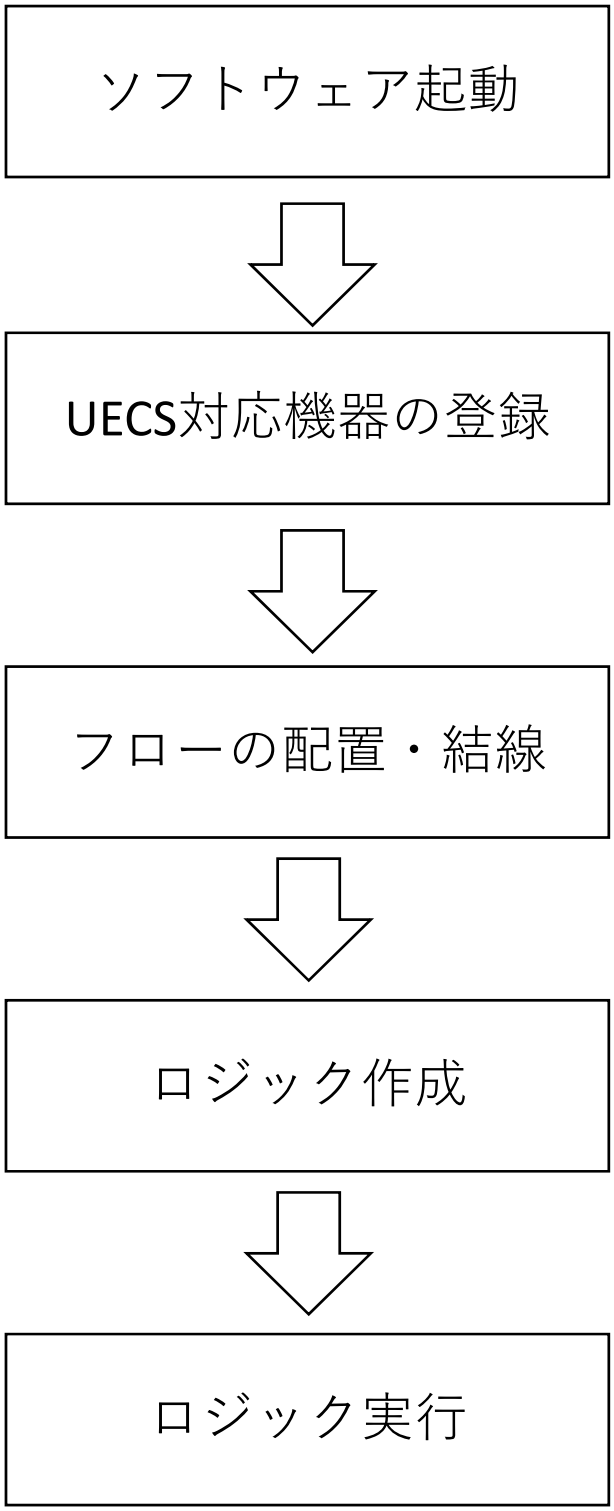
第１節　基本操作と初心者モード

下準備

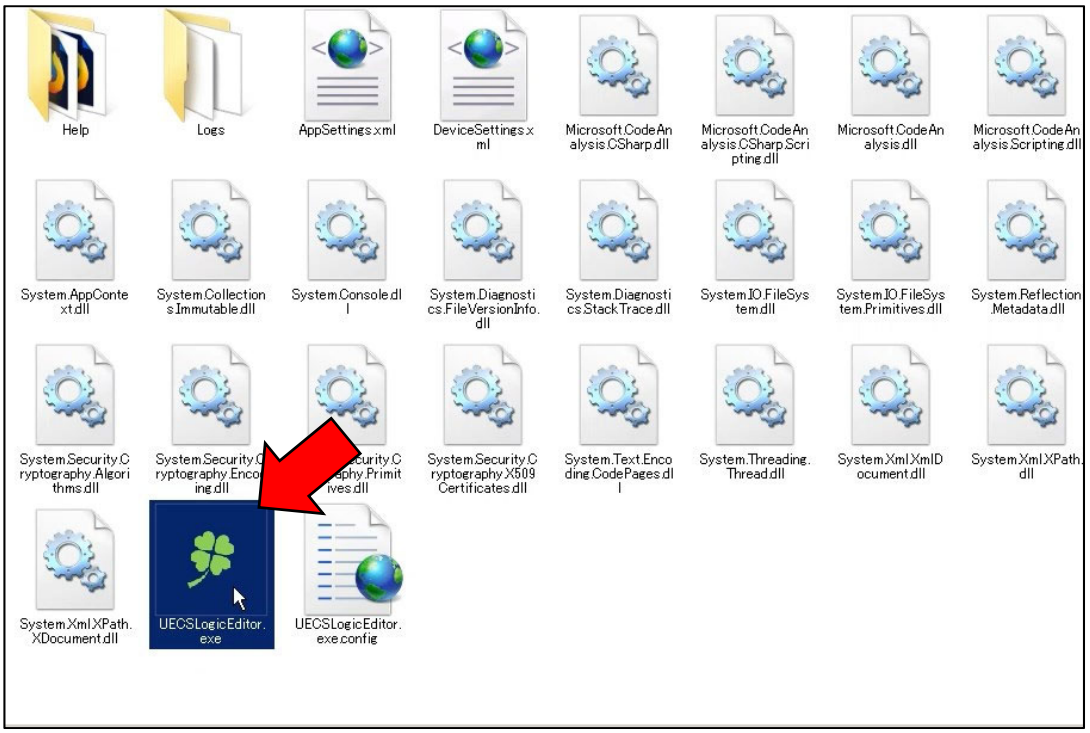
このソフトウェアを使う前にUECS機器をLANケーブルでPCと接続し、電源を入れIPアドレスやその他の設定を終わらせておく必要があります。



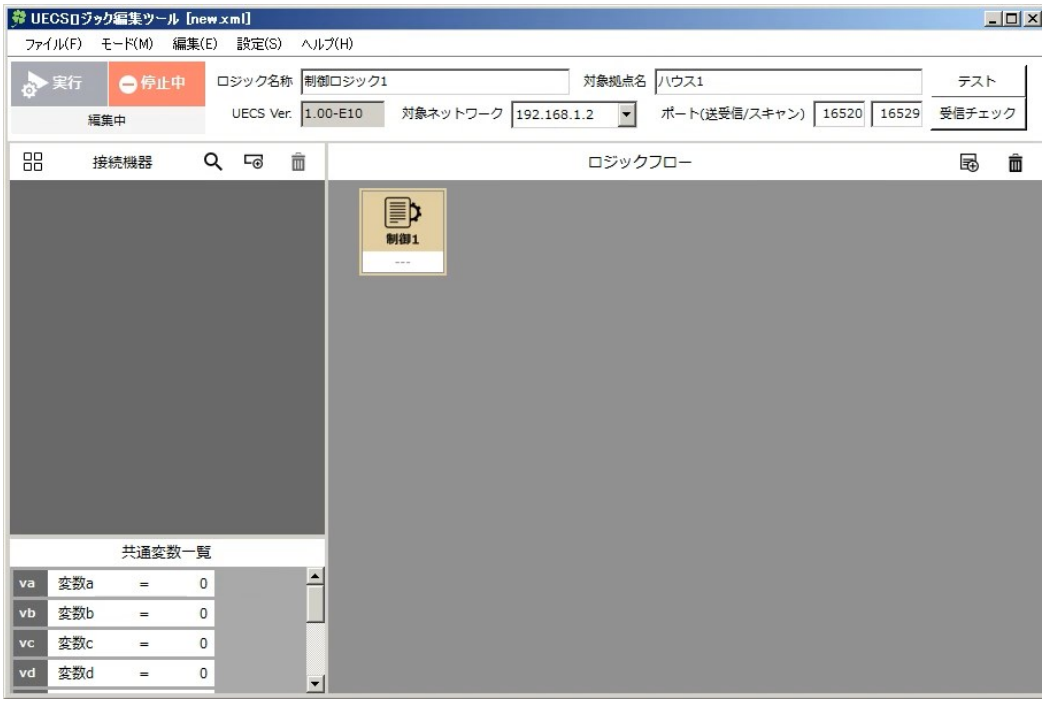
このツールは以下の手順でユーザーが制御ロジックを作成し、実行することができます。



1、起動方法

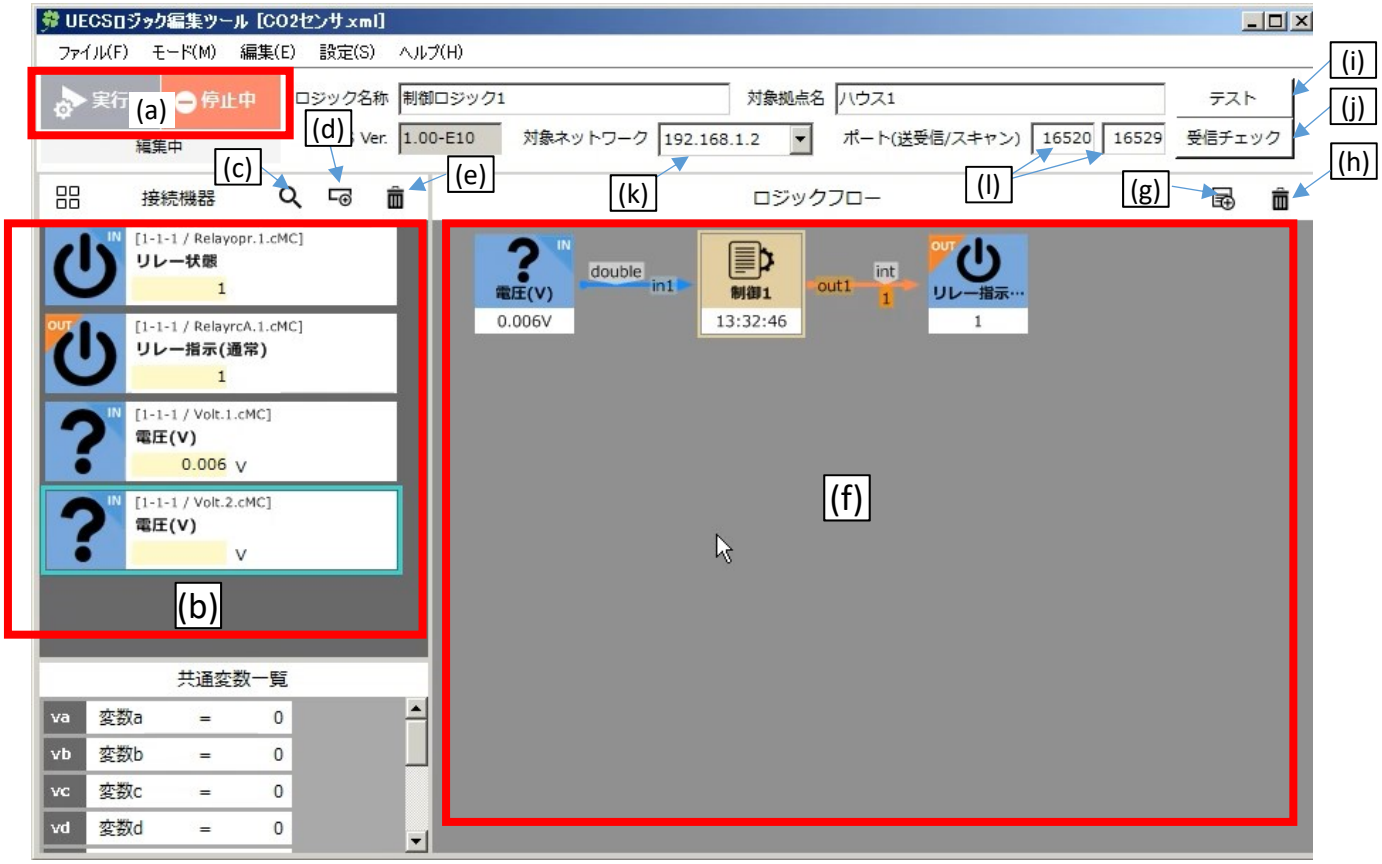


インストールフォルダの中にある四葉マークが起動用のアイコンです。



起動するとこのような画面が表示されます。

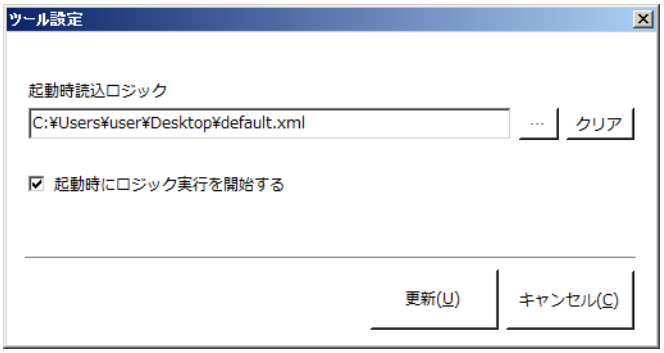
2、各部の機能



(a)実行/停止ボタン	ロジックフローを実行・停止させる
(b)接続機器リスト	UECSネットワークに接続されたセンサや制御対象のリスト
(c)接続機器の探索	UECSネットワークに接続された装置を探索する
(d)接続機器の手動追加	自動探索に対応していない機器の登録
(e)接続機器の削除	接続機器の設定を消去する
(f)ロジックフロー領域	ここにデータの流れや制御の内容を作成する
(g)スクリプト新規作成	新しい制御スクリプトのアイコンを作成する
(h)スクリプトの削除	既存の制御スクリプトのアイコンを削除する
(i)実行テストボタン	スクリプトを１回だけ実行して動作確認する
(j)受信チェックボタン	センサのデータを受信テストする、(b)のセンサの値が更新される
(k)LAN切り替え	複数のLANコネクタがあるPCではここで通信先を切り替えられる
(l)通信ポート設定	変更は不要です

ツールバーの機能

ファイル(F)	スクリプトの保存/読み込み
モード(M)	スクリプトの実行/停止
編集(E)	(b)(c)(d)(e)(g)(h)(i)(j)と同じ機能
設定(S)	右図の設定画面を表示する
ヘルプ(H)	説明書を表示

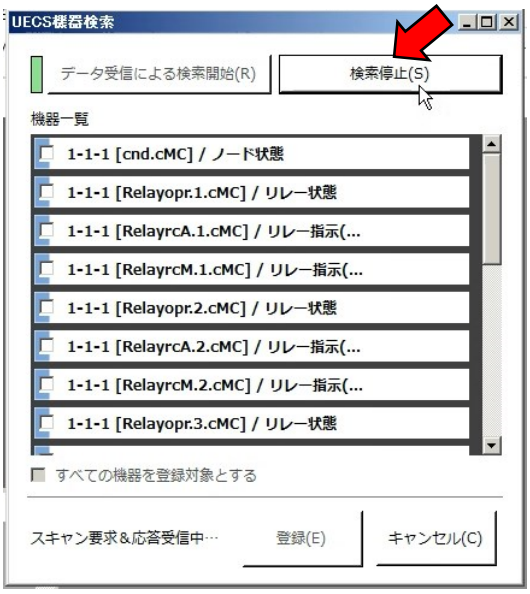


設定(S)から表示されるツール設定ではソフトウェア起動時に読み込むスクリプトファイルと、起動時のロジック自動実行の設定が可能

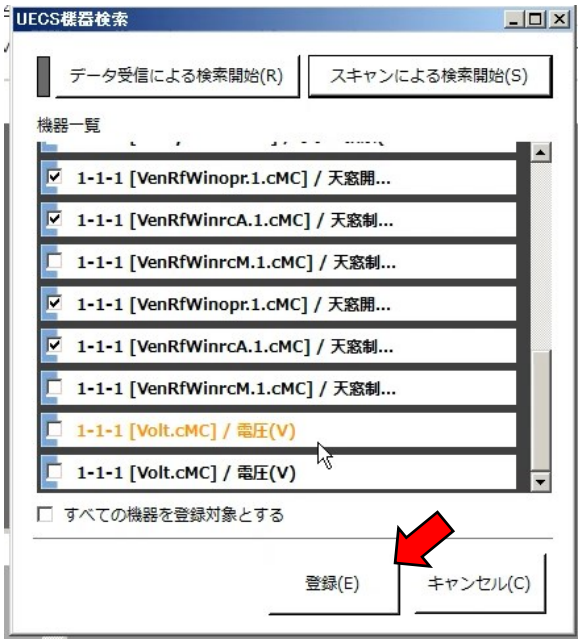
3、UECS対応機器の登録方法（自動登録）

(1)接続機器の自動探索ボタンを押す。

(2)検索画面が出るので
データを受信による検索開始(R)
スキャンによる検索開始(S)
のどちらかを押す。



(3)検出されたセンサや制御対象が表示される。十分に表示されたら検索停止を押す（停止しないと登録ボタンが押せません）。



(4)制御に使いたい項目をチェックして登録ボタンを押す。

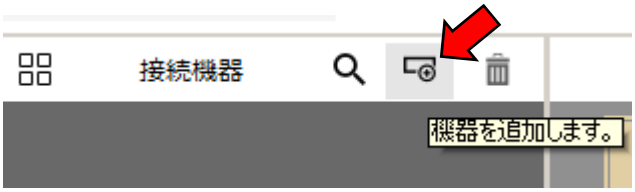
データを受信による検索開始(R)とスキャンによる検索開始(S)の違い

スキャンによる検索開始(S)は一瞬で探索が完了しますが、対応していない機器もあります。もし、スキャンによる検索開始(S)で反応がない場合、データを受信による検索開始(R)を試して下さい。

データを受信による検索開始(R)は自動探索に対応していないセンサを検出することができます。ただし、検出には最大60秒かかります。

4、UECS対応機器の登録方法（手動登録）

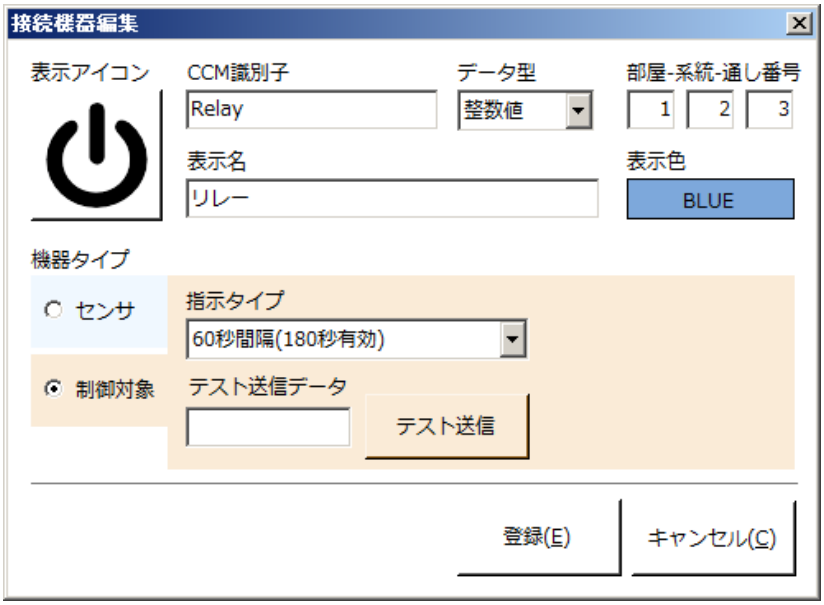
自動探索に対応していない装置や未接続の装置は手動登録する必要があります。



(1)接続機器の手動追加ボタンを押す。

(2)センサの記入例

※センサの場合、通信途絶の時に入れておく値として切断時既定値を設定することができます。



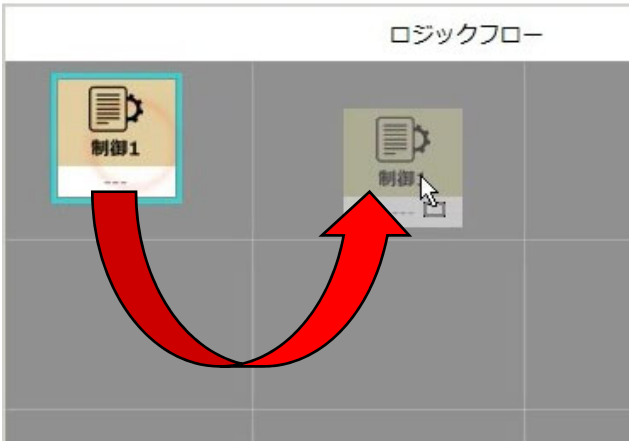
(3)制御対象(リレー)の記入例

※制御対象の場合は、値のテスト送信を行うことができます。

※一度登録した内容を修正する場合は接続機器リストのアイコンをダブルクリックすると設定画面を出せます。

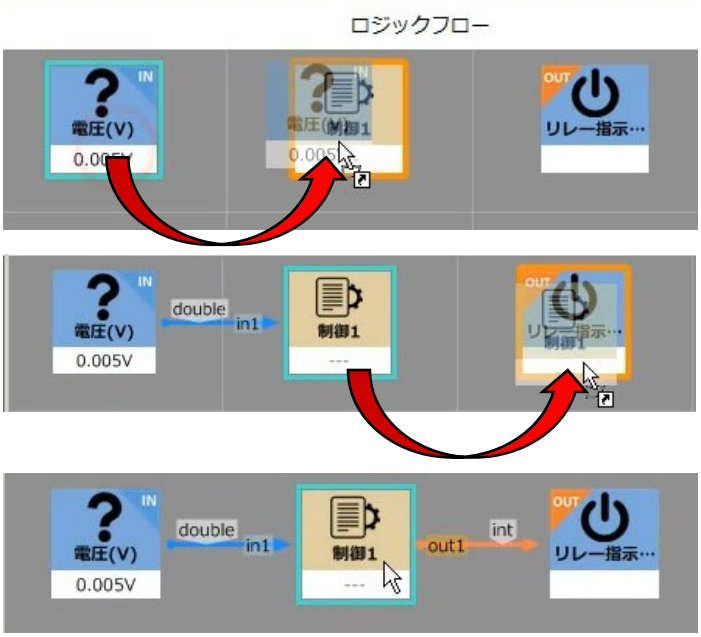
CCM識別子、部屋-系統-通し番号などの用語解説は付録のUECSの基礎知識の章を参照してください。

5、フローの結線（1）



(1)ロジックフロー領域のアイコンはドラッグアンドドロップ(D&D)で移動可能です。

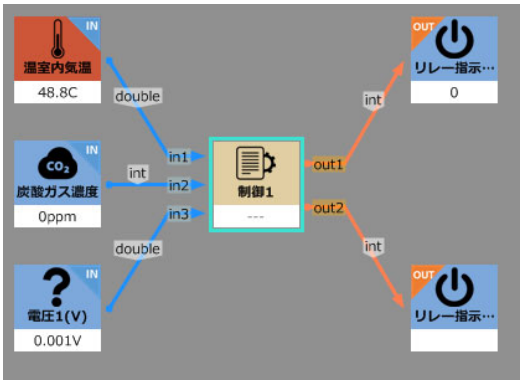
(2)接続機器リストからアイコンをD&Dしてロジックフロー領域に置けます。



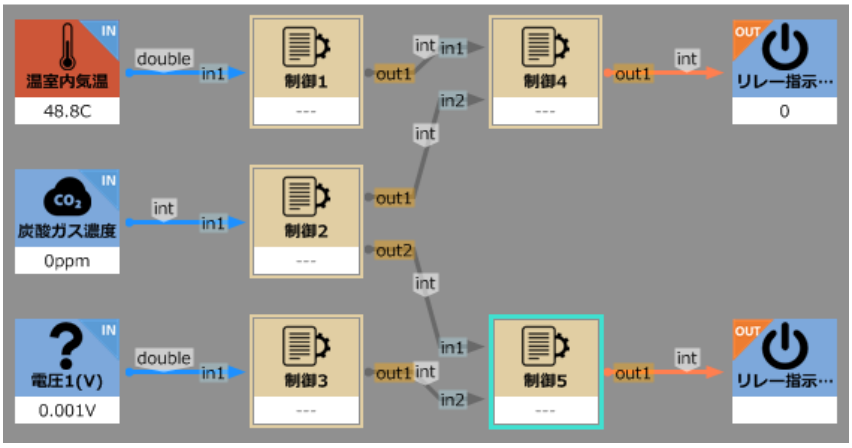
(3)ロジックフロー領域ではアイコンからアイコンへD&Dするとその間を結線することができます。

これがデータの流れを表します。

6、フローの結線（2）

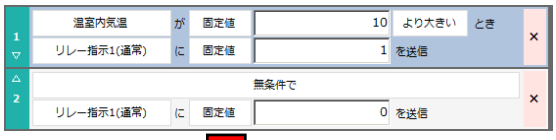


一つの制御に最大8個のセンサやリレーを繋いだり、制御同士を繋ぐこともできます。
１つの制御アイコンで全ての条件を記述するより、複数に分散したほうが融通が効きます。



フロー結線変更時の注意点と対策

初心者モードを使用する場合、一度結線した部分を消去すると、それに関連する条件判定文が消えてしまいます。これはエラー防止のための仕様ですが、つないだセンサを交換したい場合などに不便です。

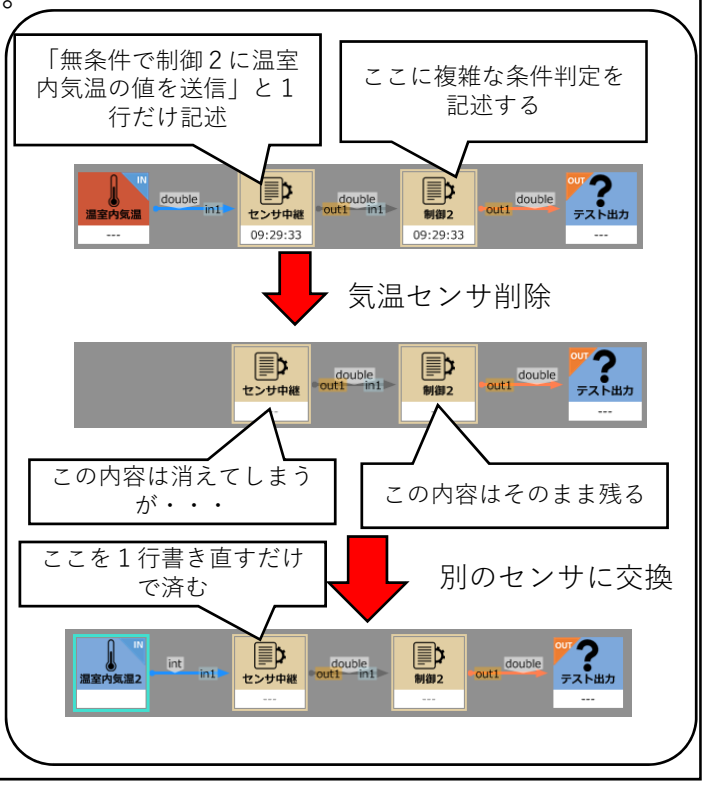


気温センサ削除後

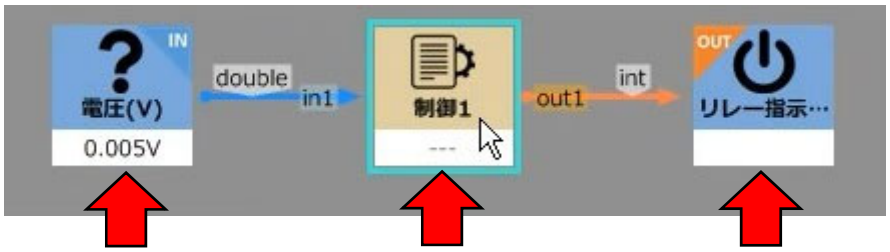
（接続されていたセンサに関連する行が消えてしまう）

そこで、右図のように中継用のアイコンを配置しておくことで影響を軽減できます。

上級者モード（通常モード）では結線を削除しても、それに関連する部分が消えることはありません。



7、初心者モードでの制御ロジックの作成



センサ　　制御　　制御対象

(1)ロジックフローの最も基本的なつなぎ方はこのようになります。

(2)「制御」のアイコンをダブルクリックすると、左の画面が表示されます。

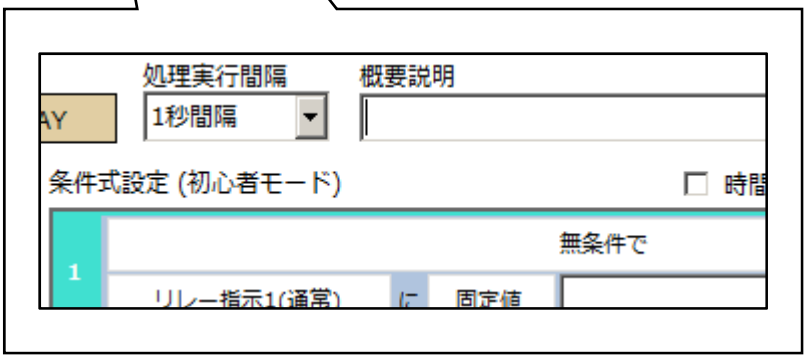
(3)選択肢を選ぶことで条件判定を作成できます。

※制御対象がリレーの場合、1を出力するとON、2を出力するとOFFになります。

条件判定文は右側のXボタンで消去、左側の上下矢印で順番の入れ替え、CTRLキーを押しながらクリックすると複数行を選択でき、コピー＆ペースト可能です。

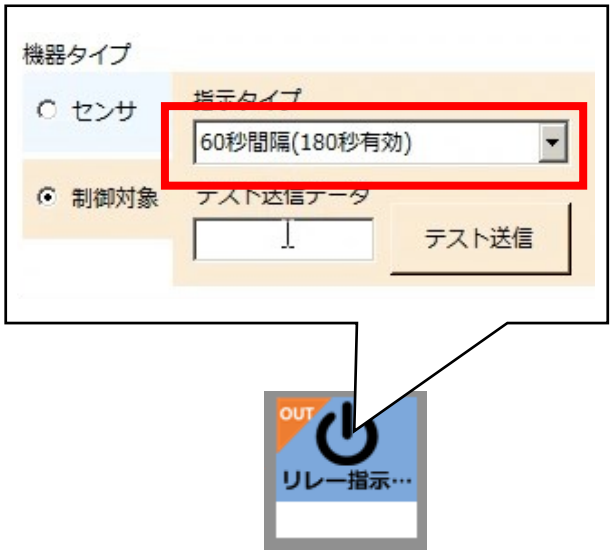
8、データの流れと制御の実行間隔

(1)このアイコン間を次々に値が送信されることで制御が成り立ちます。
センサの値は受信に成功した場合、自動的に入力されます。

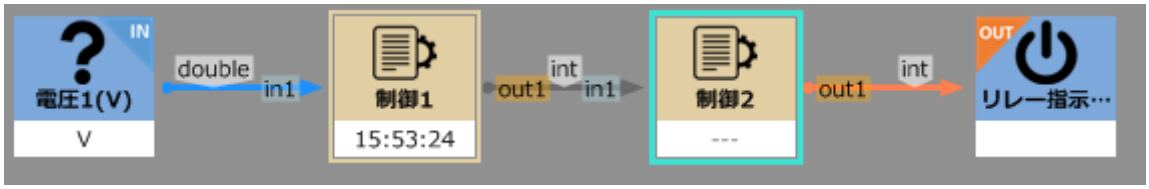


(2)制御アイコンの中に書かれた条件判定は一定時間間隔で繰り返し実行され、その結果が図のリレー指示の方に出力されます。実行間隔は「処理実行間隔」の設定から変更できます。

(3)例えば制御1の実行間隔を10秒にするとリレー指示には10秒間隔でデータが送られます。これは10秒間隔でLAN上のUECS対応機器に指令が送信されるということです。

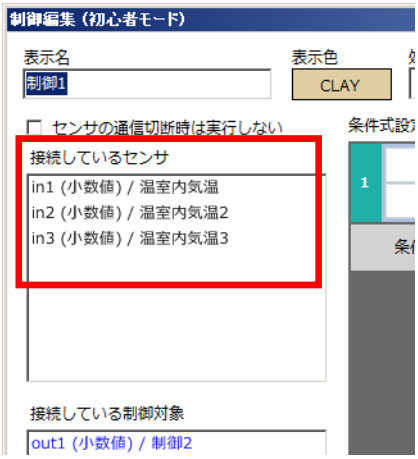


(4)ただし、制御対象にはタイムアウト時間が設定されています。例えば左の図では180秒より短い間隔で値を送らないとUECS対応機器に時間切れと見なされます。

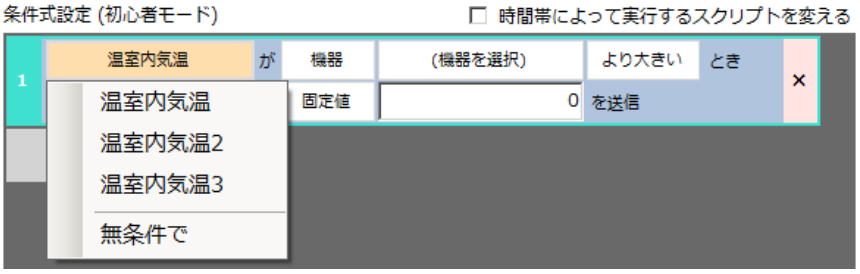


(5)このように接続されている場合、リレーユニットへの送信間隔は制御2の実行間隔で決まります。

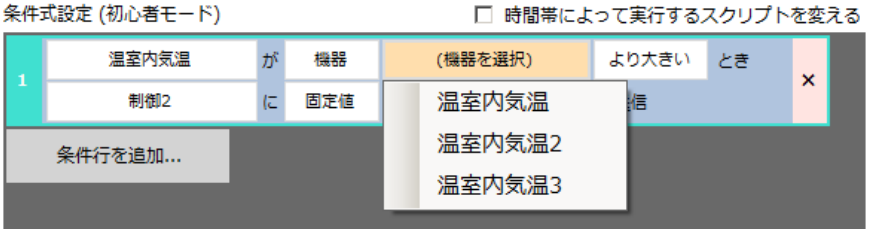
9、条件判定と値の比較



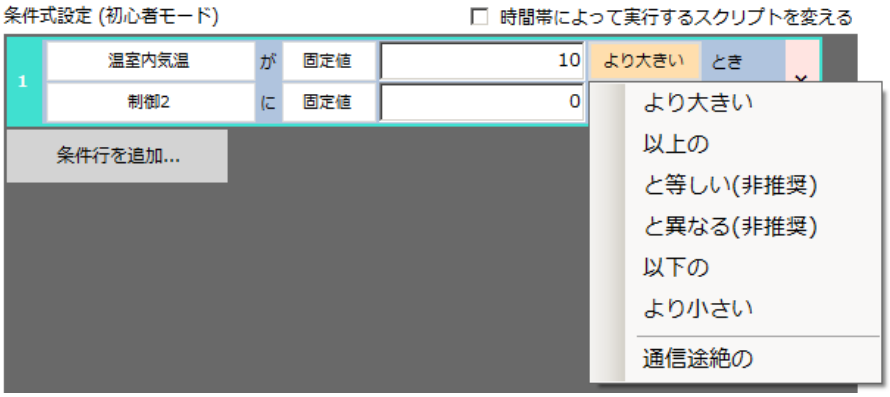
(1)制御アイコンに接続されたセンサは「入力値」として扱われ、条件判定に利用することができます。



(2) センサ値と固定値を比較できます。



(3) センサ値同士の比較もできます。



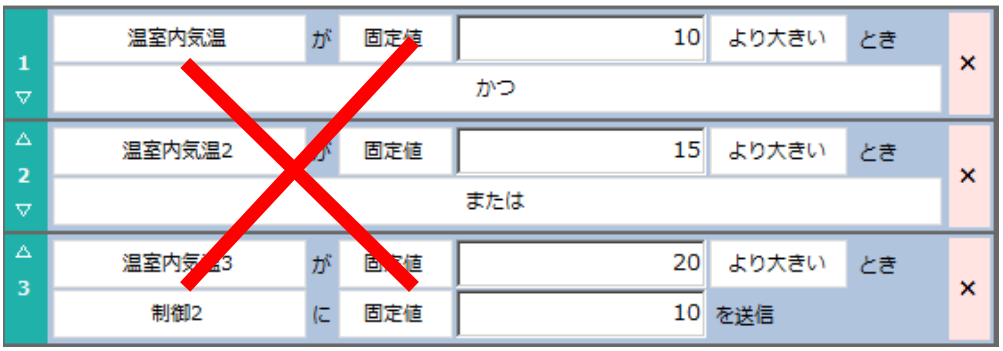
(4)設定できる比較条件は左図のようになっています。

比較条件に「非推奨」と表示される理由

この表示は扱う数値が小数の場合だけ出てきます。小数値は計算の過程で誤差が混入することがあり、厳密な「等しい」「異なる」の判定ができないことがあります。そのため、「等しい」「異なる」を使わずに大小関係で判定することを推奨します。

１０、「かつ」「または」を使った判定

(1)「かつ」「または」により複数の条件を判定に使うことができます。「かつ」を指定した場合、全条件が一致すること、「または」の場合はどれか一つの条件が一致すると反応します。



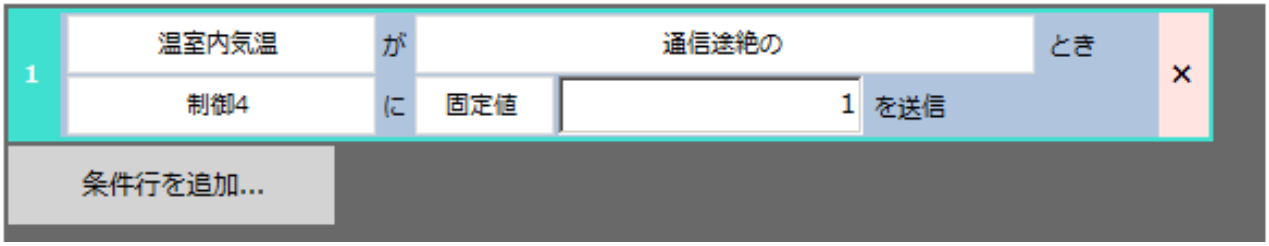
(2) ３つ以上の判定を記述する場合、「かつ」「または」を混ぜて使うことができません。全てを「かつ」あるいは「または」で統一する必要があります。

より高度な条件判定を行いたい場合、あるいは演算結果などを判定に使いたい場合は上級者モード（通常モード）でif文を使ってより複雑な判定を記述することができます。

１１、条件判定の優先順位と通信途絶の扱い

(1)記述した条件判定文は上ほど優先順位が高くなります。
この図では、気温が30より大きいという条件が最初に判定され、次に20より大きい、10より大きいという条件が判定されます。この順番で最初に一致したものが制御４に渡される値になります。

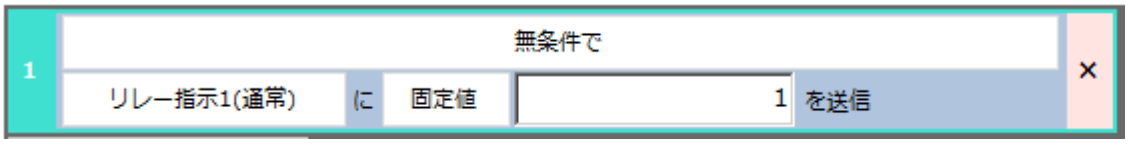
最後に無条件で0を送信という行がありますが、この行を記述せずに、最後の10より大きいという条件にまで引っかからなかった場合は制御４に何の値も送信されません。



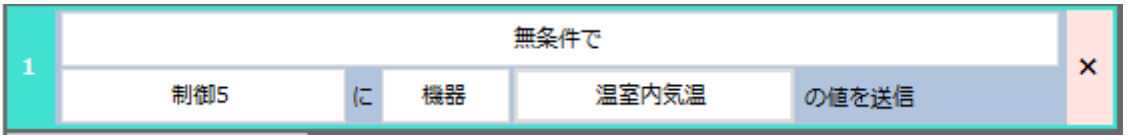
(2)アイコンが何の値も受け取れなかったときの事を「通信途絶」と言います。センサの故障によって通信が途絶することもあるかもしれません。「通信途絶」は条件判定で検出することができます。通信途絶を考慮してスクリプトを設計してください。

初心者モードでは通信途絶状態のセンサ値に対して「等しい」「異なる」「大きい」「小さい」などの判定をすると全て「不一致」となります。

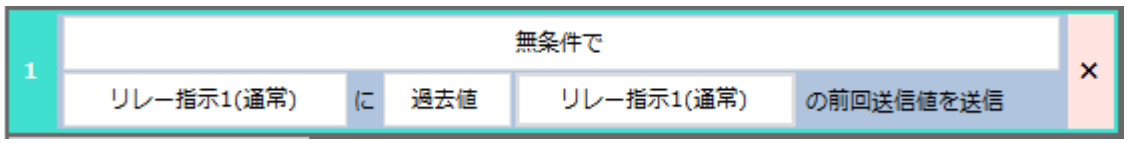
１２、値の出力



(1)「固定値を送信」とすると、単純に固定値を送信し続けます。（出力値の種類に小数が設定されている場合のみ小数値の入力が可能です）



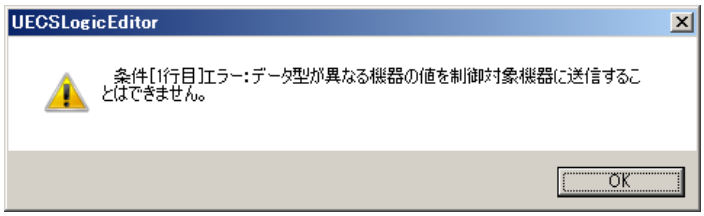
(2)「機器○ ○の値を送信 」とすると、入力側に入ってきた値をそのまま出力値にします。

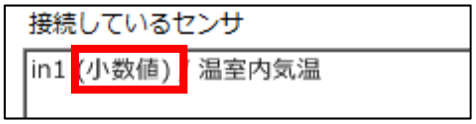


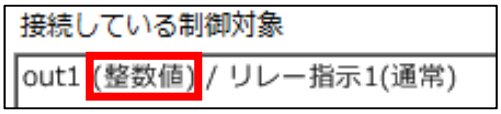
(3)「過去値○ ○の前回送信値を送信 」とすると、一つ前に条件判定が実行されたときの出力値をもう一度送信します。

初心者モードで出力側に指定できる値の種類は上記３種類のみですが、より複雑な計算結果を出力したい場合は、モード切替により上級者モードを使ってください。上級者モードではC#に準拠した四則演算の他、数学関数(System.Mathクラス)が利用できます。

(2)を使って制御ロジックを作成すると下のようなエラーが出ることがあります。

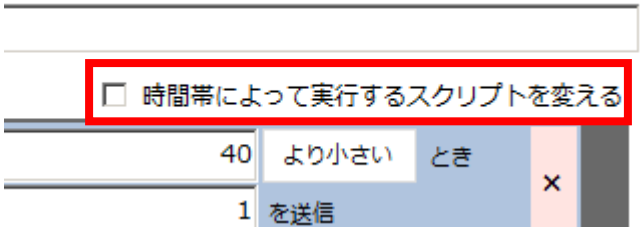




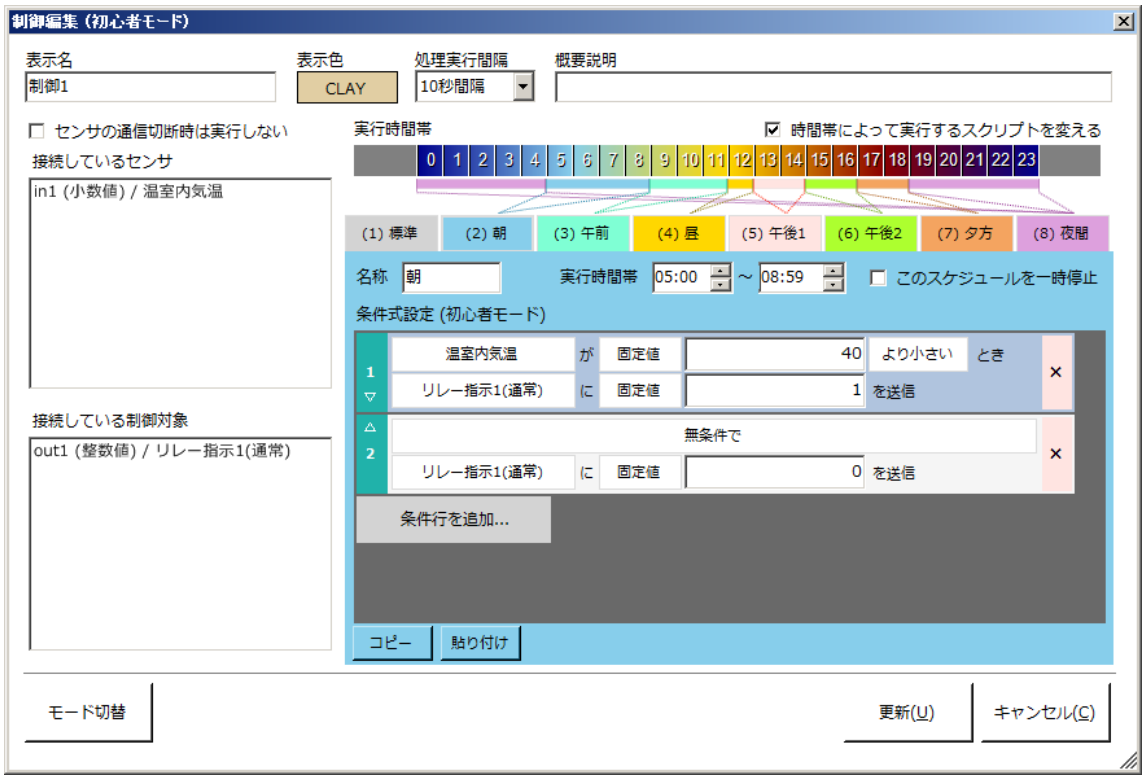


これは入力側と出力側の整数値と小数値の設定が食い違っているためです。
出力側が「制御」アイコンの場合に限り、制御編集画面左下の「接続している制御対象」をクリックすると整数/小数の区別を変更できます。
出力側が「接続機器」の場合はアイコンをクリックすると設定を変更できますが、整数値しか受け付けない装置に小数値を出力すると不具合の原因になります。

１３、スケジュール機能



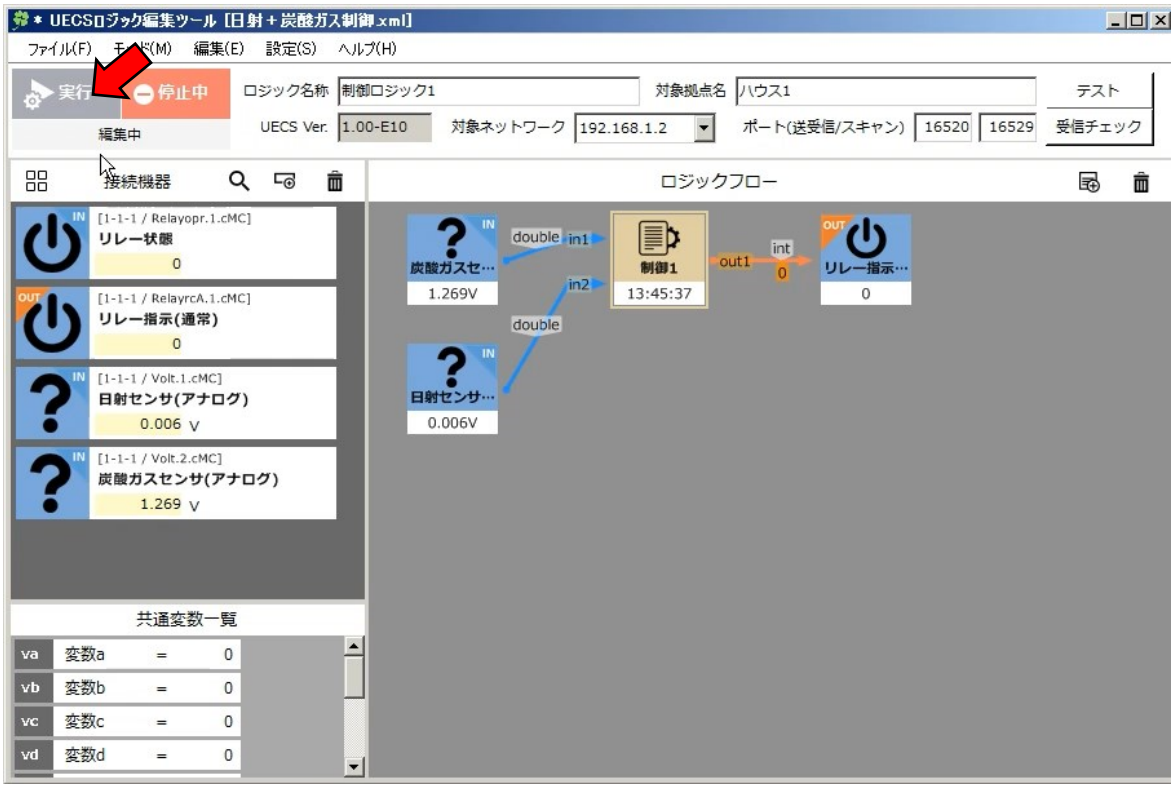
(1)制御編集画面で「時間帯によって実行するスクリプトを変える」をチェックするとスケジュール機能が使えます。



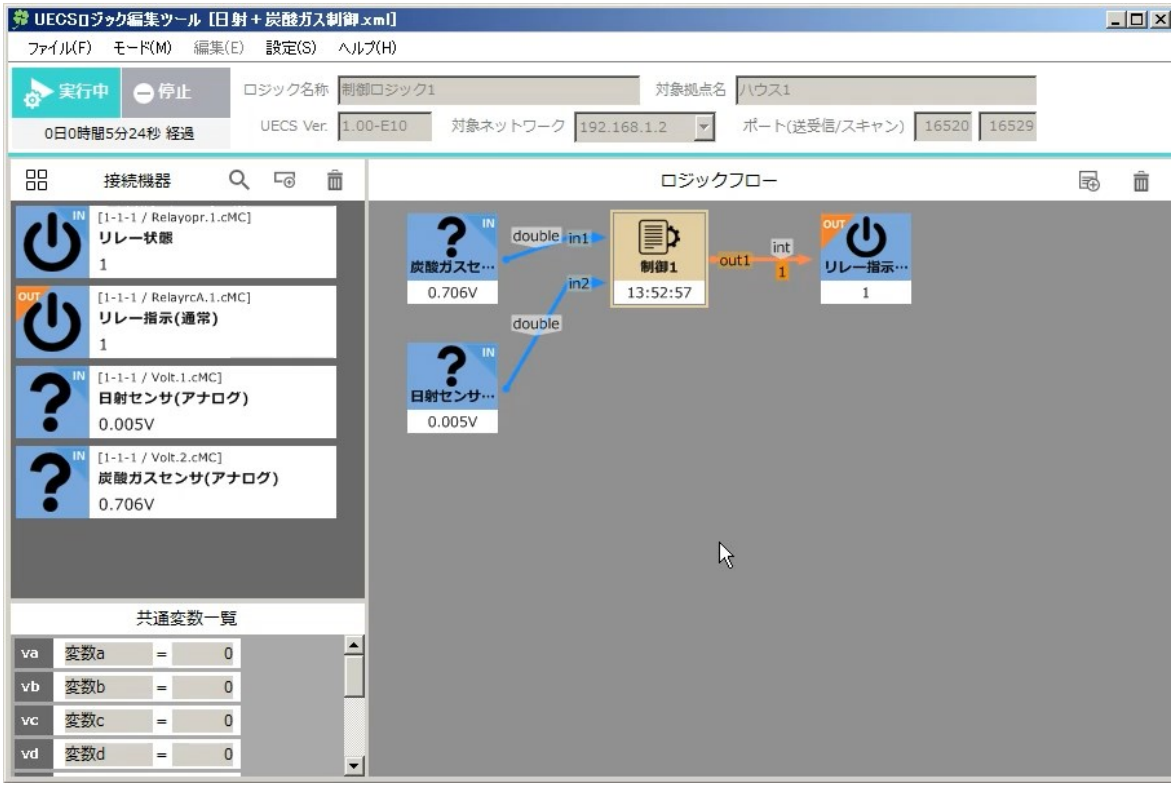
(2)時間帯ごとに最大8種類のスクリプトを設定できます。

他のタブにスクリプトをコピーする場合、CTRLキーを押しながらスクリプトをクリックすると複数まとめて選択できます。その状態で下のコピー/貼り付けボタンを押すとまとめてコピー・貼り付けができます。

１４、制御ロジックの実行



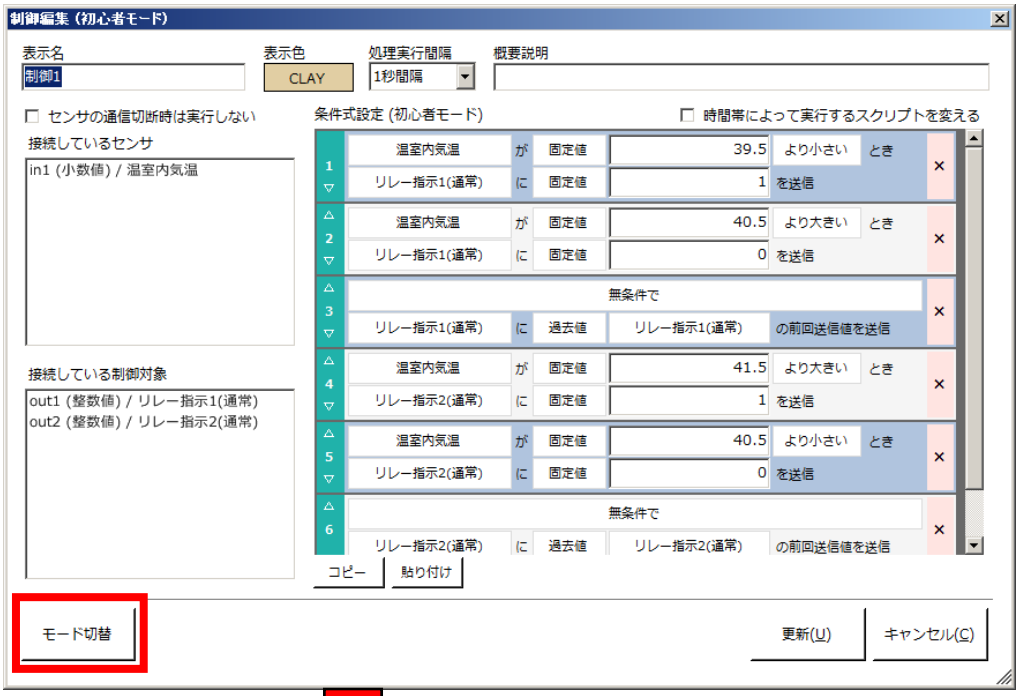
(1)実行ボタンを押すと、作成したロジックフローが実行されます。



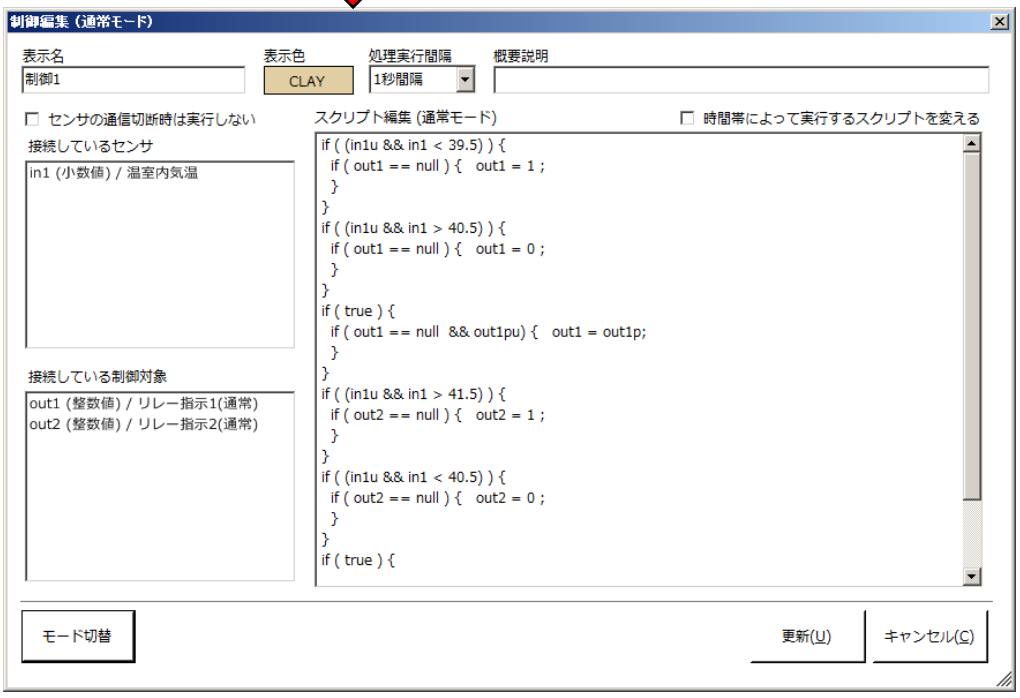
(2)実行中はリアルタイムに送受信されている値が表示されます。

第２節　上級者モード

1、モード切替



上級者モード（通常モード）切り替え



制御編集画面でモード切り替えボタンを押すと初心者モードから上級者モードに切り替わります。初心者モードのスクリプトは上級者モードのスクリプトに自動変換されます。
この変換を行う前にファイル名を変えて保存しバックアップを取ることをおすすめします。上級者のスクリプトを初心者モード用には変換できません。

２、基本仕様

上級者モードのスクリプトはC#に類似の文法になっています。ここではごく一部だけ解説します。

(1)記法

```
a=1; //変数に代入
a=a+1; //aに1足した結果をaに代入
a=Sin(angle/180*3.141592); //組み込み関数呼び出し
return; //意図してスクリプトを終える
```

```
//ダブルスラッシュは1行コメント
/*
複数行のコメントはこの記号で括る
*/
```

一つの処理ごとにセミコロン";"を後ろにつけて区切ります。

(2)一時的な変数を作る

int x;	整数の変数xを作る
int x=0;	整数変数xを作って0を代入
double y=0.5;	小数変数yを作って0.5を代入

消えない変数が必要な場合、特殊変数一覧に記載された共通変数（グローバル変数）を使ってください。

intは整数値のみ代入可能、doubleは小数を代入できる変数です。
これらは一時的な変数でスクリプトの実行が終わると中身が消えます。

(3)四則演算

x + y	x と y を足す	x=1+2;
x - y	x から y を引く	x=y-3;
x * y	x と y を掛ける	x=3*y;

x / y	x を y で割る	double x=9.0/2.0; //xは4.5になる
		int y=9/2; //yは４になる(小数位切り捨て)

x % y	x を y で割った余り	double x=9.0%2.0; //xは4.5になる
		int y=9%2; //yは1になる

-x	xの符号を反転する	y=10; x=-y; //xは-10になる
x++	xに1を足す	x=0; x++; //xは1になる
x--	xから1を引く	x=0; x--; //xは-1になる

計算の優先順位を指定するのに括弧( )を使うことができます。

(4)小数値と整数値の相互変換

```
double x;
int y;
y=(int)x; //変数の型を変換
```

小数と整数が食い違う変数はそのままでは代入できません。計算式に( )で括った変数型を記述するとその部分だけ一時的に別の型に変換できます。(型キャスト)

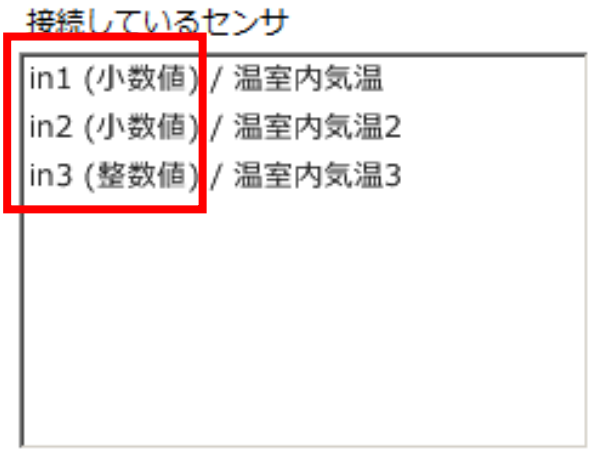
```
y=(double)( a+b );    //計算後の型変換
```

( ) を使うと、計算が終わった後の数値を型変換する、などの指定ができます。

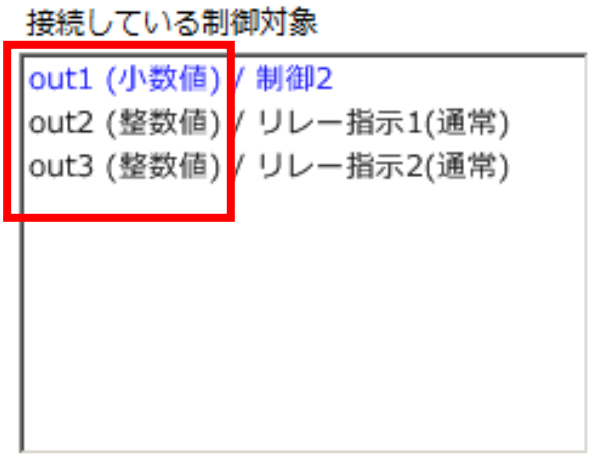
小数→整数の変換では小数位は切り捨てられます
整数→小数 の変換では僅かな誤差が生じることがあります。

基本文法はC#のものですが、ループ命令(for,whileなど)が実装されていない点が異なります。

3、入力値、出力値の扱い



制御編集画面の右上に表示されているのが入力値を受け取る変数名はin?(?は数字)という名前になっています。この図では、
in1　温室内気温
in2　温室内気温2
in3　温室内気温3
が代入された状態でスクリプトが実行されます。



制御編集画面の右下に表示されているのが出力用変数名で、out?（?は数字)という名前になっています。この図では、
out1　制御2アイコンに出力する値
out2　リレー指示1
out3　リレー指示2
に接続されています。

```
//入力をそのまま出力に
out1=in1;
//型変換して出力(out2とin2の型が食い違う場合)
out2=(int)in2;
//計算して出力
out3=in3+10;
```

out?という変数に何かの数値を代入すると、その値を出力するという意味になります。
（代入を複数回行った場合は最後に代入した値が反映されます）
もし、何の値も代入せずにスクリプトを終えると、何も送信しないという意味(通信途絶)になります。

> out?に代入した値は次にスクリプトが実行されたときには消えています。もし、過去に出力した値はout?pという変数を参照してください。

４、特殊変数一覧

(1)in?、out?を含め、あらかじめ役割の決まっている変数を示します。

変数名	説明	使用例
in1 ～ in8	制御に入力として接続した機器、制御の値を格納した変数。	1番目のセンサ(入力機器)として接続した機器の値が28.0を超えた場合に、out1へ1を送信する **if (in1 > 28.0) {out1 = 1;** }
in1u ～ in8u	制御に入力として接続した機器の通信状況を格納した変数。機器の受信値が有効ならtrue、通信が途絶しているならfalseが入る。	1番目のセンサ(入力機器)として接続した機器が通信途絶している場合、out1へ0を送信する **if (in1u == false) {out1 = 0;}**
out1 ～ out8	制御に出力として接続した機器、制御へ送信する値をセットするための変数。制御の実行開始時にnullにリセットされ、スクリプト中でセットしなかった場合は送信自体を行わない。	1番目の制御対象機器として接続した機器に、値[1]を送信する **out1 = 1;**
out1p ～ out8p	制御に出力として接続した機器(out1～8)に対し、前回セットした値を格納した変数。	1番目の制御対象機器として接続した機器に、前回セットした値を再度送信する。 **out1 = out1p;**
out1pu ～ out8pu	制御に出力として接続した機器(out1～8)に対し、前回送信が実行されたかどうかを格納した変数。	1番目の制御対象機器として接続した機器に対し、前回値を送信した場合は同じ値を再送する。 **if (out1pu == true) {out1 = out1p;}**
count	各制御が実行された回数を格納した変数。ロジック実行の開始時に0にリセットされる。	実行2回に1回(実行回数÷2の余りが0の場合)、out1へ1を送信する **if(count % 2 == 0) {out1 = 1;}**
va ～ vj	共通変数。スクリプト中で任意の小数値をセットすることができる。どの制御アイコンからも同じ変数を参照・操作可能。ロジック開始前に、共通変数一覧で初期値を手動入力できる。	**va = 10.0;** **out1 = ToInt(vb);**

(2)特殊変数の応用

```
if(count % 3 ==0)
{
//3回に1度実行
}
```

３回に１回だけ実行される処理を記述する。

countは2147483647までカウントアップした後オーバーフローして負数になりますが、1秒間隔で実行した場合、オーバーフローまで約68年かかります。

5、if文による条件判定の書き方(1)

(1)比較演算子の使い方

演算子	意味
>	より大きい
>=	以上
<	より小さい
<=	以下
==	等しい
!=	等しくない

if(比較演算)
{
一致した場合の処理;
}
という書き方になります。

「等しい」の表現が”==”ダブルイコールであることに注意してください。

```
if ( in1 > 10 ) {
out1 = 0 ;
}
```

in1が10より大きければout1を0にする。

```
if ( in1 == 10 ) {
out1 = 1 ;
}
```

in1が10ならばout1を1にする。

```
if ( in1 >= 10 ) {
out1 = 2 ; out2 = 3 ;
}
```

in1が10以上ならばout1を2、out2を3にする。

```
if ( in1 != (in2+10) ) {
out3 = 4 ;
}
```

in1がin2に10を足した値と等しくなければout3を4にする。

(2)論理演算子の使い方

演算子	意味
&&	「かつ」AND
||	「または」OR
!	「否定」NOT

```
if ( in1 > 10 && in1<30) {
out1 = 0 ;
}
```

in1が10より大きく、かつin1が30より小さければout1を0にする。

```
if ( in1 > 10 || in2>10) {
out1 = 1 ;
}
```

in1が10より大きいまたはin2が10より大きければout1を1にする。

```
if (!( in1 > 10)) {
out1 = 2 ;
}
```

in1が10より大きくなければout1を2にする。

```
if ( (in1 > 10 && in2>10) || in3<20) {
out1 = 100 ;
}
```

in1とin2が両方とも10より大きい、またはin3が20より小さければout1を100にする。

```
if ( (in1 > 10 && in1<20) || (in2>20 && in2<30)) {
out2 = 200 ;
}
```

in1が10～20の範囲内、またはin2が20～30の範囲内ならばout2を200にする。

括弧を使うことで判定の順番を決めることができます。

6、if文による条件判定の書き方(2)

(1)else文

```
if ( in1 >10 ) {
  out1 = 1 ;
 }
else
{
  out1 = 0 ;
}
```

in1が10より大きい場合、out1を1に、
そうでない場合、out1を0にする。

判定から漏れた場合にどうするかを
記述できます。

(2)else if文

```
if ( in1 >30 ) {
  out1 = 3 ;
 }
else if (in1>20)
{
  out1 = 2 ;
}
else if (in1>10)
{
  out1 = 1 ;
}
```

in1が30より大きい場合、out1を3に。
そうでない場合、in1が20より大きいときはout1を2に、
そうでない場合、in1が10より大きいときはout1を1にする。

最初の判定から漏れたら次の判定、それから漏れたら次の判定へと、判定をいくつも記述できます。

elseの見落としに注意

(A)

```
if ( in1 >30 ) {
  out1 = 3 ;
 }
else if (in1>20)
{
  out1 = 2 ;
}
else if (in1>10)
{
  out1 = 1 ;
}
```

(B)

```
if ( in1 >30 ) {
  out1 = 3 ;
 }
if (in1>20)
{
  out1 = 2 ;
}
if (in1>10)
{
  out1 = 1 ;
}
```

(A)と(B)では全く意味が変わります。

(A)では最初に一致したif文で代入されたout1が最終結果になります。
(B)では全部のif文が実行され、最後に一致した時に代入されたout1が最終結果になります。

例えばin1が35の時、out1の最終値は(A)では3に対し、(B)では1になります。

これはバグの原因になりやすいので注意してください。

7、特殊な数値の表現

条件判定では特殊な数値の表現をすることがあります。

true	真を示す、0ではない値
false	偽を示す、0のこと
null	変数に有効な値が入っていない (通信途絶状態)

```
if ( in1u ==true ) {
  out1 = 1 ;
 }
```

in1の値が有効な場合out1を1にする。

```
if ( in1u ) {
  out1 = 1 ;
 }
```

上と同じくin1の値が有効な場合out1を1にする。

```
if ( in1u ==false) {
  out1 = null ;
 }
```

in1が通信途絶したらout1を意図的に通信途絶にする。

```
if (! in1u) {
  out1 = null ;
 }
```

上と同じくin1が通信途絶したらout1を意図的に通信途絶にする。

```
if ( out1 ==null ) {
  out1 = 0 ;
 }
```

out1にまだ何も代入されていない場合、out1を0にする。

in1は通信途絶時にもnullになりません（切断時規定値か0が代入されます）in1uを使って通信途絶判定を行ってください。

if文では比較演算子を書かなかった場合、括弧の中の数値が0(偽)か0でない値(真)かの判定が行われ、0ではない数値が与えられた時に一致判定が作動します。

８、組み込み数学関数

使用可能な関数一覧を示します。

関数	説明
Abs(v)	vの絶対値を返す
Sqrt(v)	vの平方根を返す
Pow(v1, v2)	v1のv2乗を返す
Sign(v)	符号を取得する
Round(v)	四捨五入する
Ceiling(v)	小数点以下切り上げる
Floor(v)	小数点以下切り捨て (負数では小さな整数側に変換)
Truncate(v)	小数点以下切り捨て (負数では大きな整数側に変換)
Min(v1, v2)	小さい方を返す
Max(v1, v2)	大きい方を返す
Sin(double r)	サイン
Cos(double r)	コサイン
Tan(double r)	タンジェント
Asin(double r)	アークサイン
Acos(double r)	アークコサイン
Atan(double r)	アークタンジェント 使用可能範囲　-π/2 < atan(r) <π/2
Atan2(double y, double x)	アークタンジェント 使用可能範囲　-π < atan2(x,y) <π

三角関数の入力は全て小数値で単位はラジアンです。

９、日付時刻処理

System.DateTimeクラスライブラリ相当の機能が使用可能です。
DateTime.Nowで取得可能なのはPCの時計の情報になります。

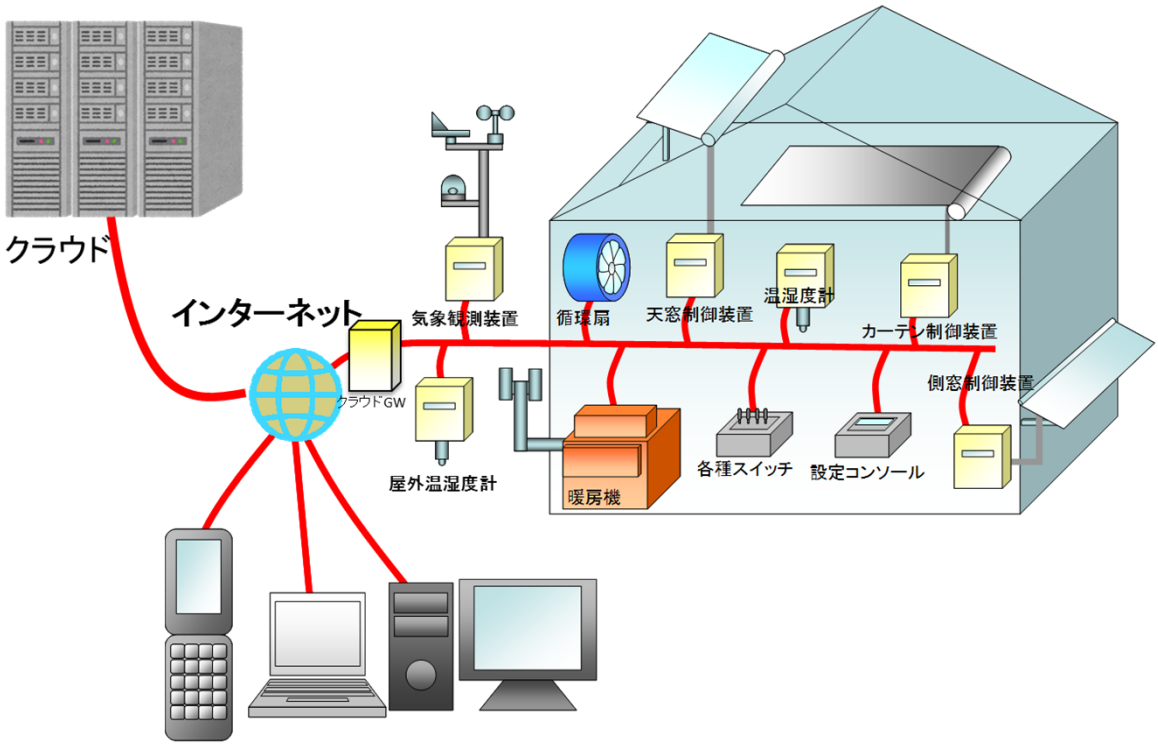
例文

DateTime dt = DateTime.Now;	//日付時刻の取得開始
int year=dt.Year;	//西暦年を取得する
int month=dt.Month;	//月を取得する
int day=dt.Day;	//日を取得する
int hour = dt.Hour;	//時を取得する
int minute = dt.Minute;	//分を取得する
int second = dt.Second;	//秒を取得する
int doy=dt.DayOfYear;	//1月1日からの経過日数

DateTime dt = DateTime.Now;は最初に実行する必要があります。

付録１　UECSの基礎知識

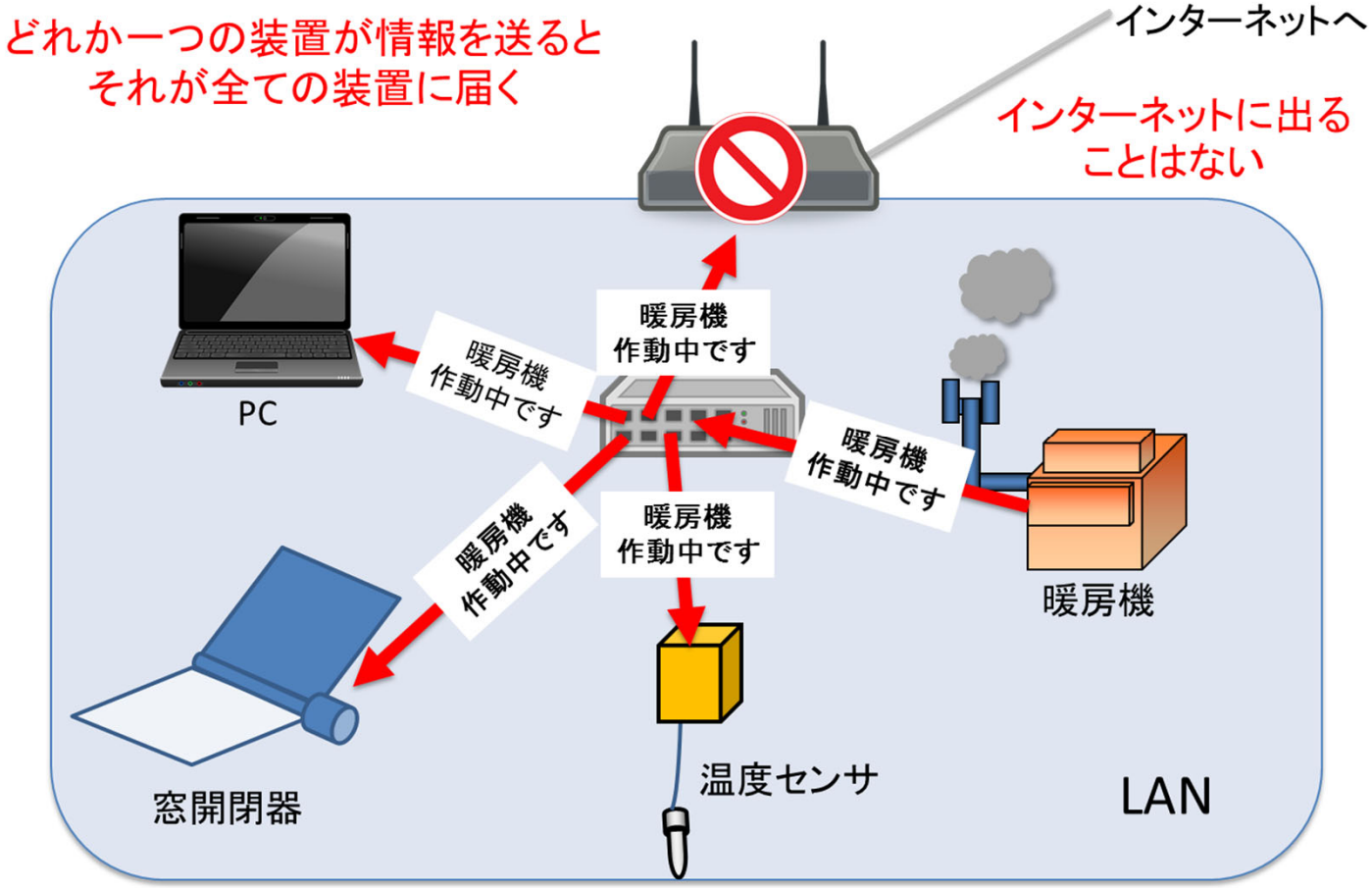
UECSとは

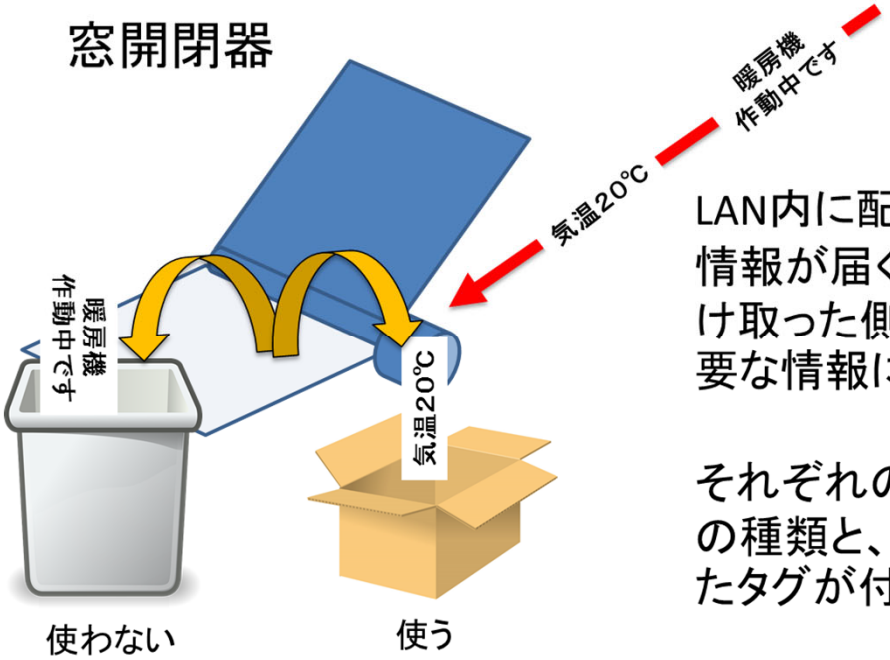


UECSとは施設園芸用の通信の共通規格です。温室内の機器をLANで接続してネットワークを形成し、互いに情報伝達を行うことで制御を行います。UECSに対応していればメーカーの異なる機器同士でも共通の方法で通信できます。

※インターネットとクラウドは無くてもUECSを動かすことは可能です。

UECSの通信の特徴



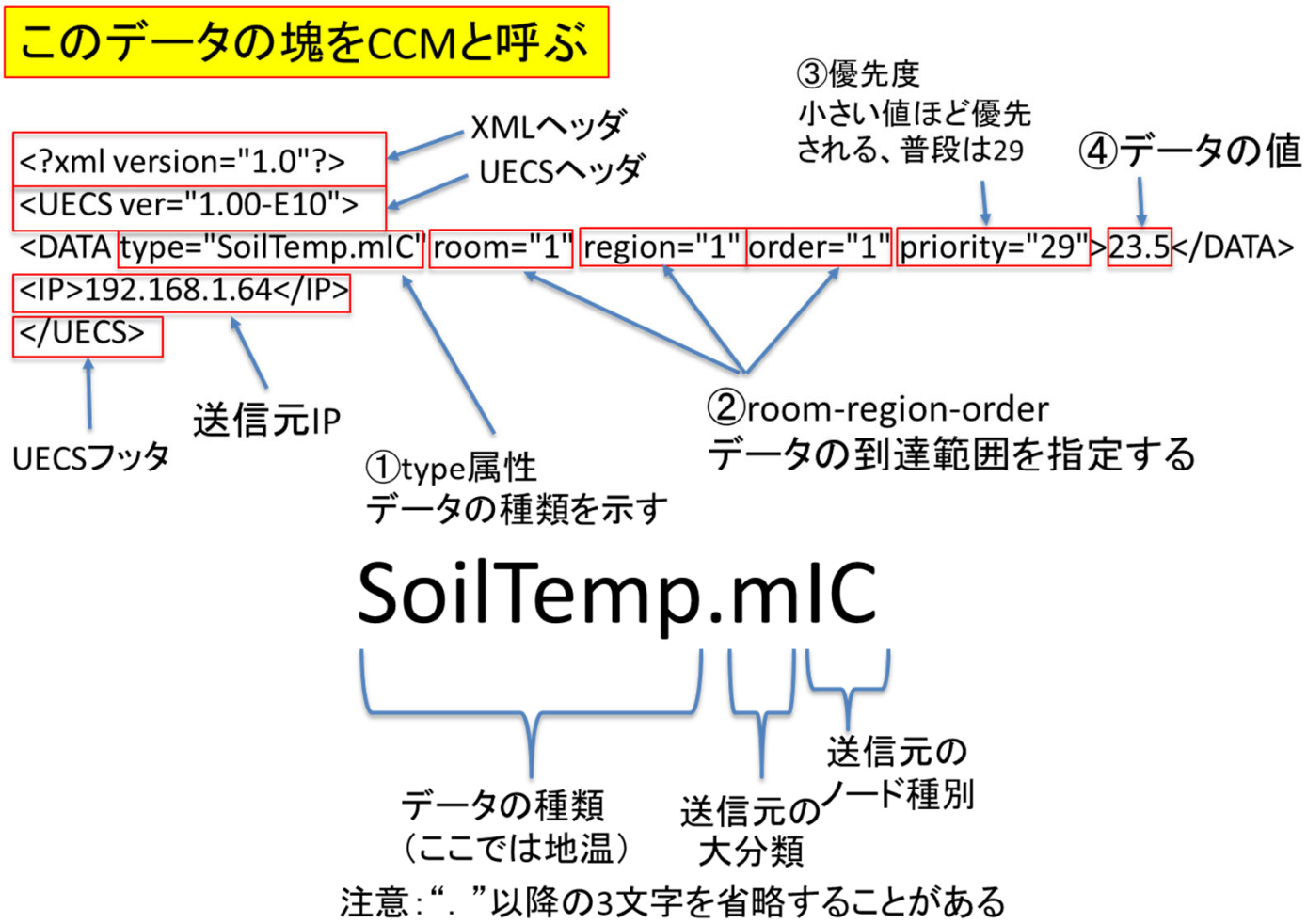


LAN内に配信されたあらゆる情報が届くので、情報を受け取った側が使う情報と不要な情報に仕分ける。

それぞれの情報には、情報の種類と、届ける範囲を示したタグが付いている。

※インターネット上のクラウドと通信するにはLAN内にクラウド中継機を設置する必要がある。

実際に装置の間でやり取りされる情報の例



注意：“．”以降の3文字を省略することがある

情報種類	CCM名（type属性値）	単位	精度（最低値）	通信間隔（最低値）
1．時間				
時間（時：分：秒）	Time	hhmmss	整数	1分おき
日付	Date	yyMMdd	整数	1分おき
2．センサ				
温室内気温	InAirTemp	℃	下1桁	10秒おき
温室内湿度	InAirHumid	%	整数	10秒おき
温室内CO2	InAirCO2	ppm	整数	10秒おき
温室内日射	InRadiation	kW/m²	整数	10秒おき
屋外気温	WAirTemp	℃	下1桁	10秒おき
屋外湿度	WAirHumid	%	整数	10秒おき
屋外CO2	WAirCO2	ppm	整数	10秒おき
屋外日射	WRadiation	kW/m²	整数	10秒おき
屋外風速	WWindSpeed	m/s	整数	10秒おき
屋外風向	WWindDir16		整数	10秒おき
屋外降雨	WRainfall		整数	10秒おき

あらかじめ定義されたtype属性値の例(一部)
※定義されていないセンサ等はtypeを自作しても良い。

room-region-orderの意味

room ：部屋番号　　温室単位で値を付ける
region：系統番号　　温室の内部を複数に分割する時に値を変える
order ：通し番号　　同じものが複数ある場合に値を変える

0を指定すると特別な意味(ワイルドカード)を持つ
room,region　　:整数0-255の範囲で設定可能
order　　:整数0-30000の範囲で設定可能
以上はあくまでも推奨される使い方で、どのように割り振るかは、使う人が最初に決める

送信側のルール

データの送信間隔には次のような種類がある
（1）常に一定時間間隔で送信するもの
　　1秒間隔：　A_1S_0
　　10秒間隔：A_10S_0
　　60秒間隔：A_1M_0
（2）常に一定時間間隔または値が変化したときに送信するもの
　　1秒間隔＋変化時：　　A_1S_1
　　10秒間隔＋変化時：　A_10S_1
　　60秒間隔＋変化時：　A_1M_1
（3）必要なときだけ一定時間間隔で送信するもの
　　1秒間隔：　S_1S_0
　　60秒間隔：S_1M_0
（4）要求があったときだけ送信するもの
　　要求があったときだけ送信：　B_0
　　要求＋変化時に送信：　　B_1

一定時間間隔で送信されるCCMは、送信間隔の3倍の時間経過しても届かないと通信途絶（タイムアウト）と見なされる。

受信側のデータ仕分けルール

（1）欲しいtypeが書かれていないデータは捨てる

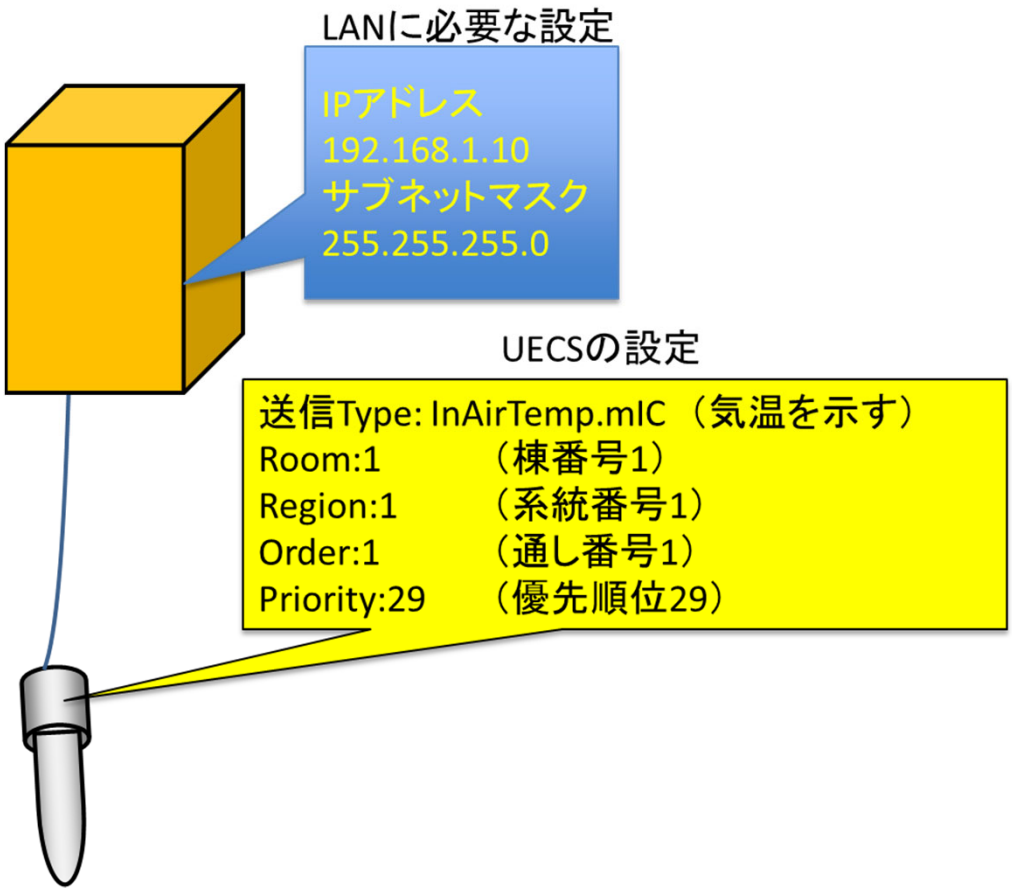
（2）room-region-orderの3つの値が完全一致したデータだけを受け取る

（3）room-region-orderが完全一致したデータが複数の場所から送信されている場合、priorityの値が小さいデータを受け取る

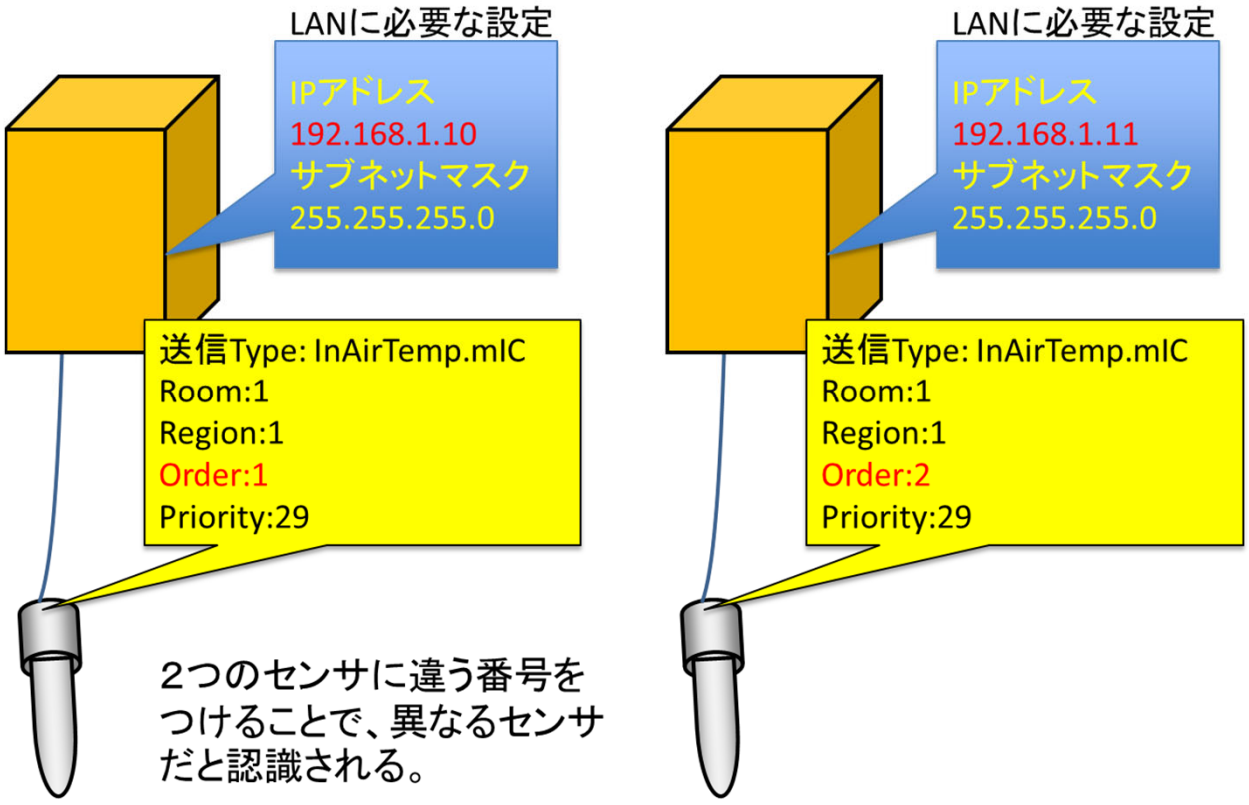
（4）room-region-orderの値が0（ワイルドカード）の部分は受信側の設定値がどんな値でも一致とみなす

（ワイルドカードは優先順位が低くroom-region-orderの3つの値が完全一致したCCMが他の場所から送られている場合はそちらが優先される。）

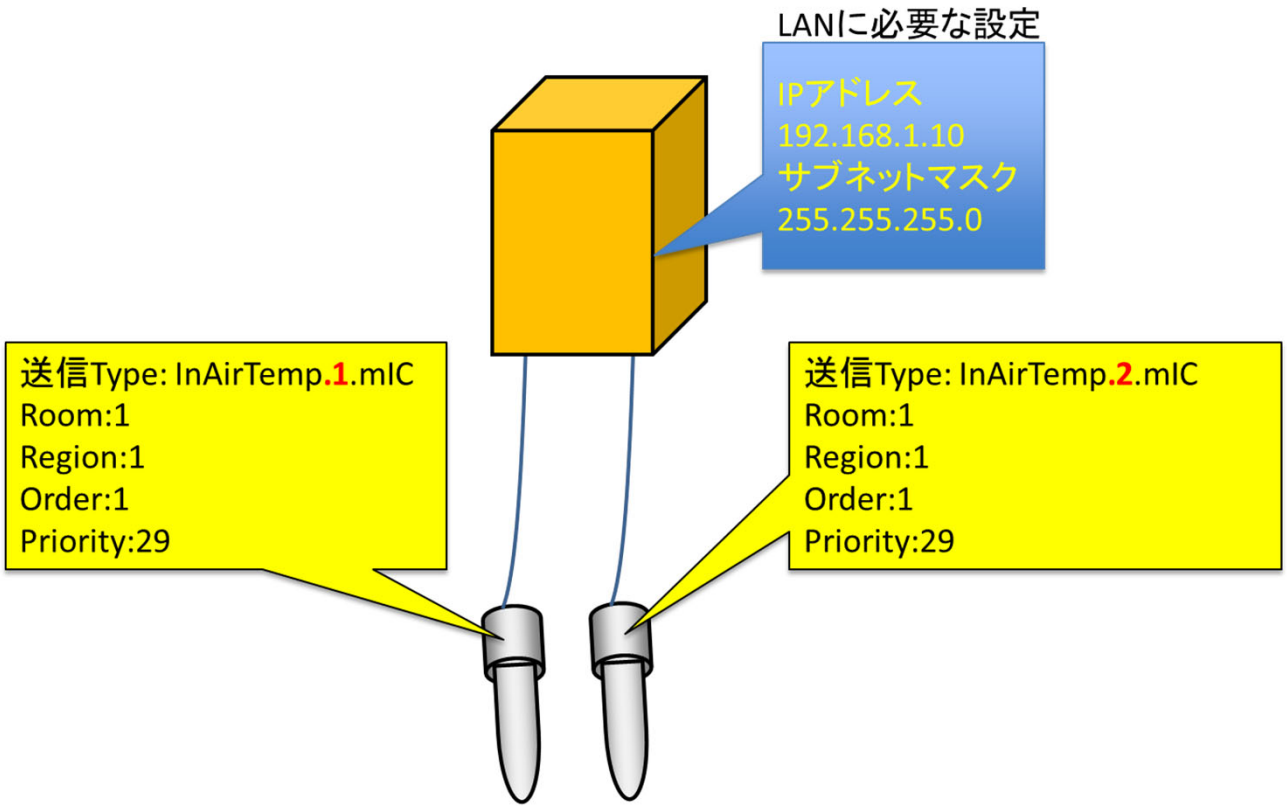
センサにIDを振る



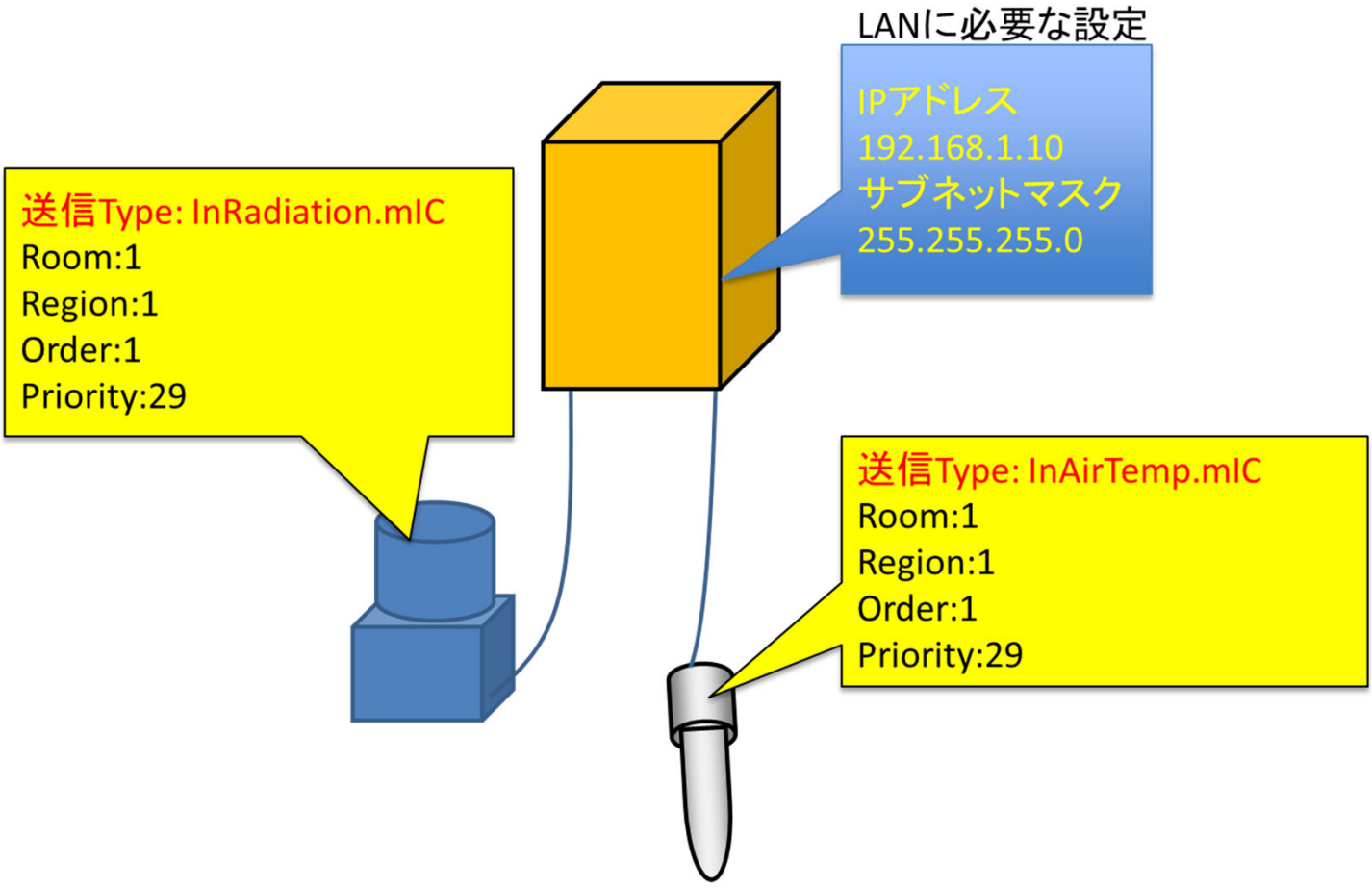
複数の温度センサがある場合



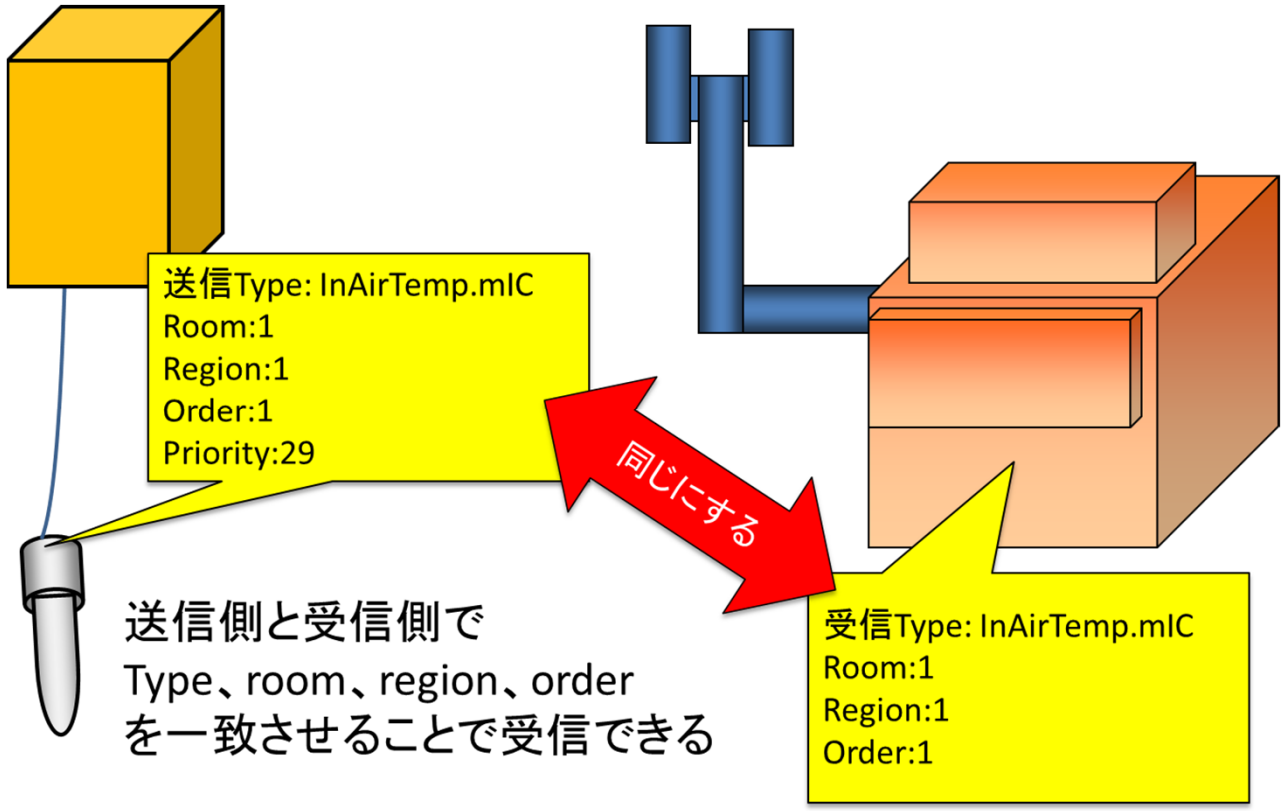
１つの装置に同じセンサが複数ある場合

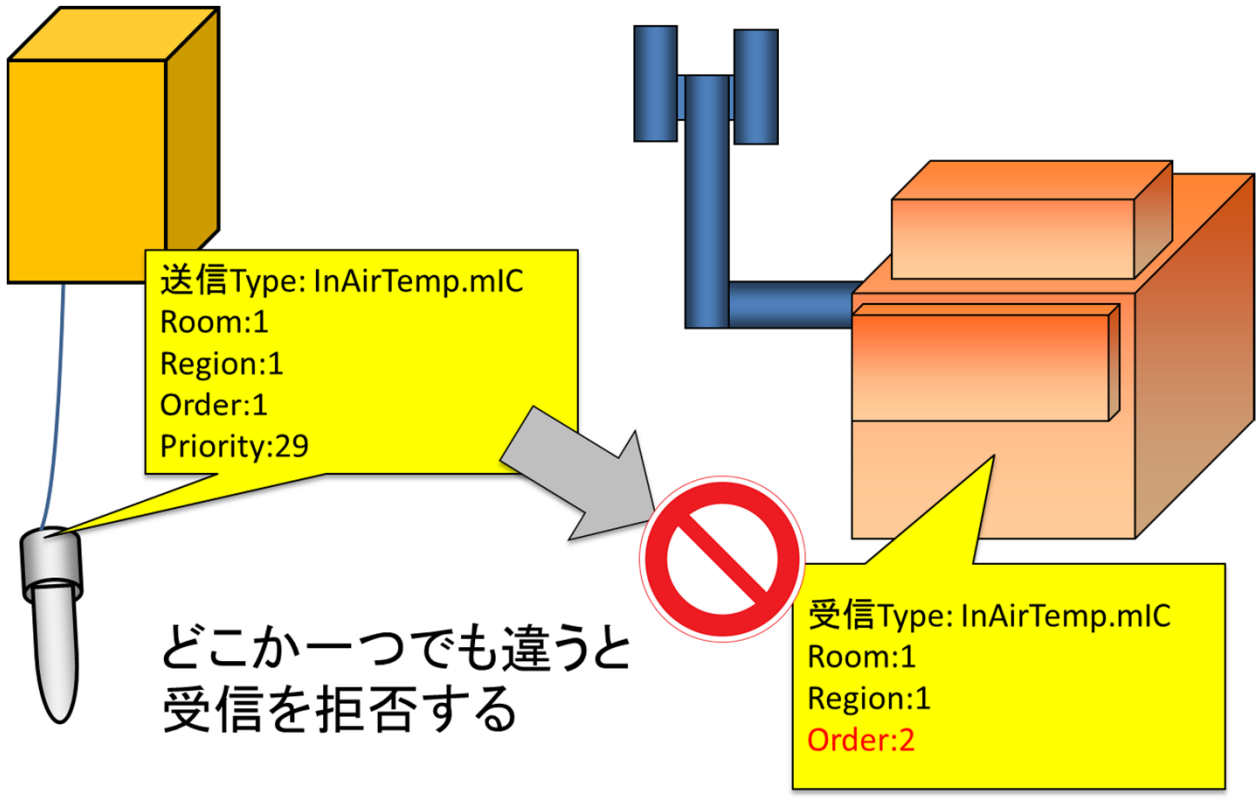


１つの装置に２つの違うセンサがある場合

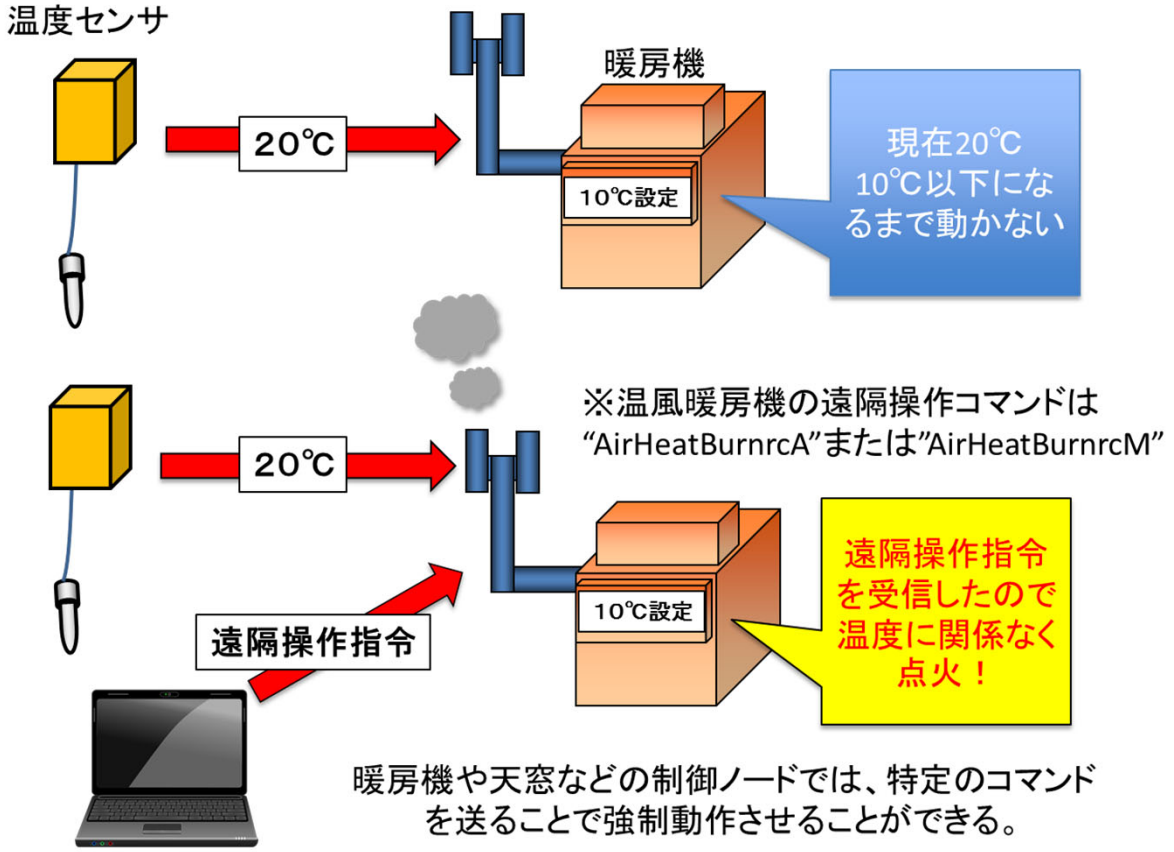


情報の送信と受信（気温センサと暖房機）

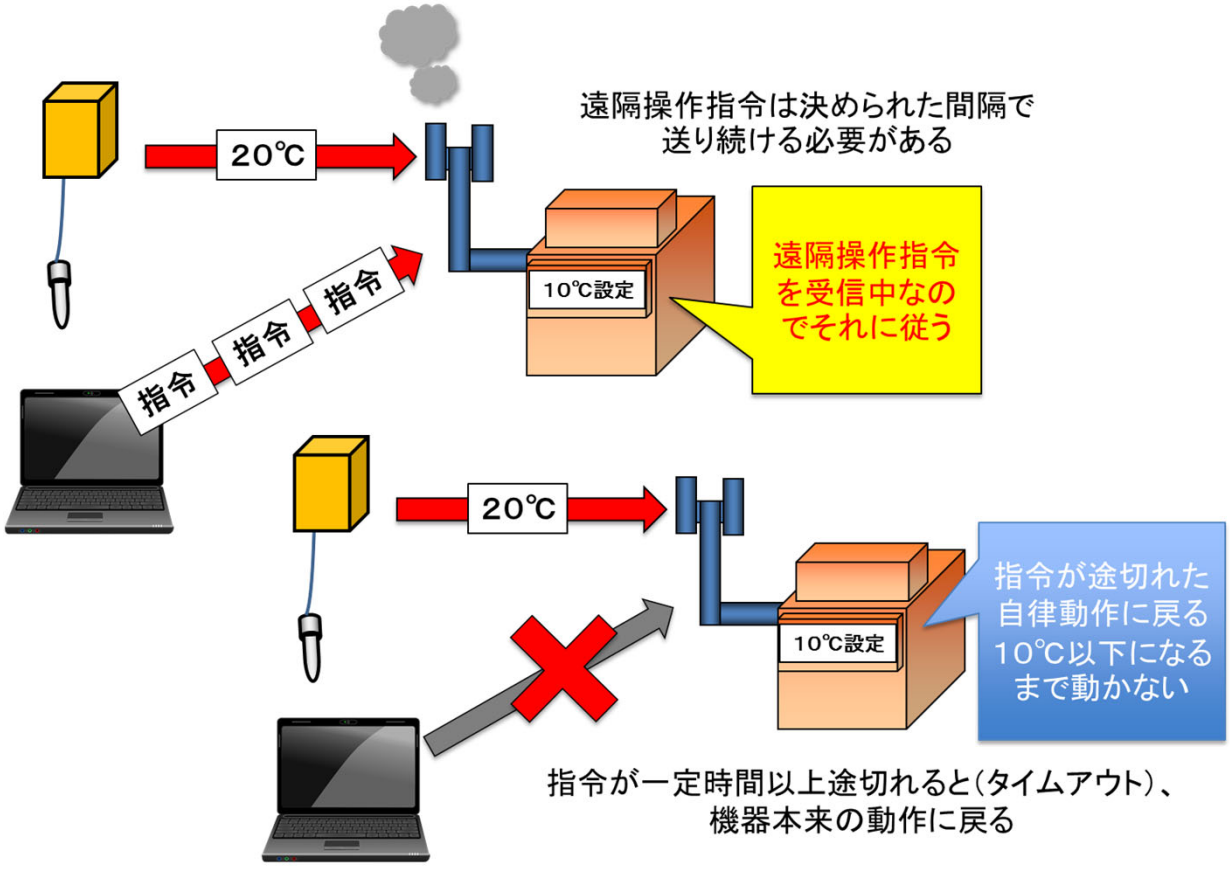




遠隔操作指令



遠隔操作指令のタイムアウト



付録２　Windows10の強制的な再起動への対策

UECS用ロジック開発ツールは常時稼働するPCで動かす必要がありますが、Windows10の標準機能ではアップデート時の再起動をユーザーが停止することができなくなりました。PCの再起動が発生するとロジック開発ツールが停止してしまう恐れがあるので、再起動のタイミングをユーザーが制御する必要があります。

2019年1月現在 西村誠一氏よりWinUpdateSettingsというフリーソフトが公開されています。

ホームページ
http://www.asahi-net.or.jp/~tz2s-nsmr/

説明・ダウンロードサイト
http://www.asahi-net.or.jp/~tz2s-nsmr/WinUpdateSettings.html

このソフトウェアを使うとWindows10のアップデートによる再起動を抑制することができます。詳細な使い方は説明サイトを参照してください。

MEMO

UECS用ロジック開発ツール活用マニュアル１
基本操作と機能概説
2019年3月

編集・発行
国立研究開発法人農業・食品産業技術総合研究機構
西日本農業研究センター企画部産学連携室
〒721- 8514 広島県福山市西深津町6-12-1
Tel.084-923-5385 Fax.084-923-5215

農研機構ホームページ
http://www.naro.affrc.go.jp/

本マニュアルに掲載した内容の一部は、
生研支援センター
「革新的技術開発・緊急展開事業」（うち地域戦略プロジェクト）
「UECSプラットホームで日本型施設園芸が活きるスマート農業の実現」の
支援を受けて行っています。