

利用シーン: リモートI/O制御

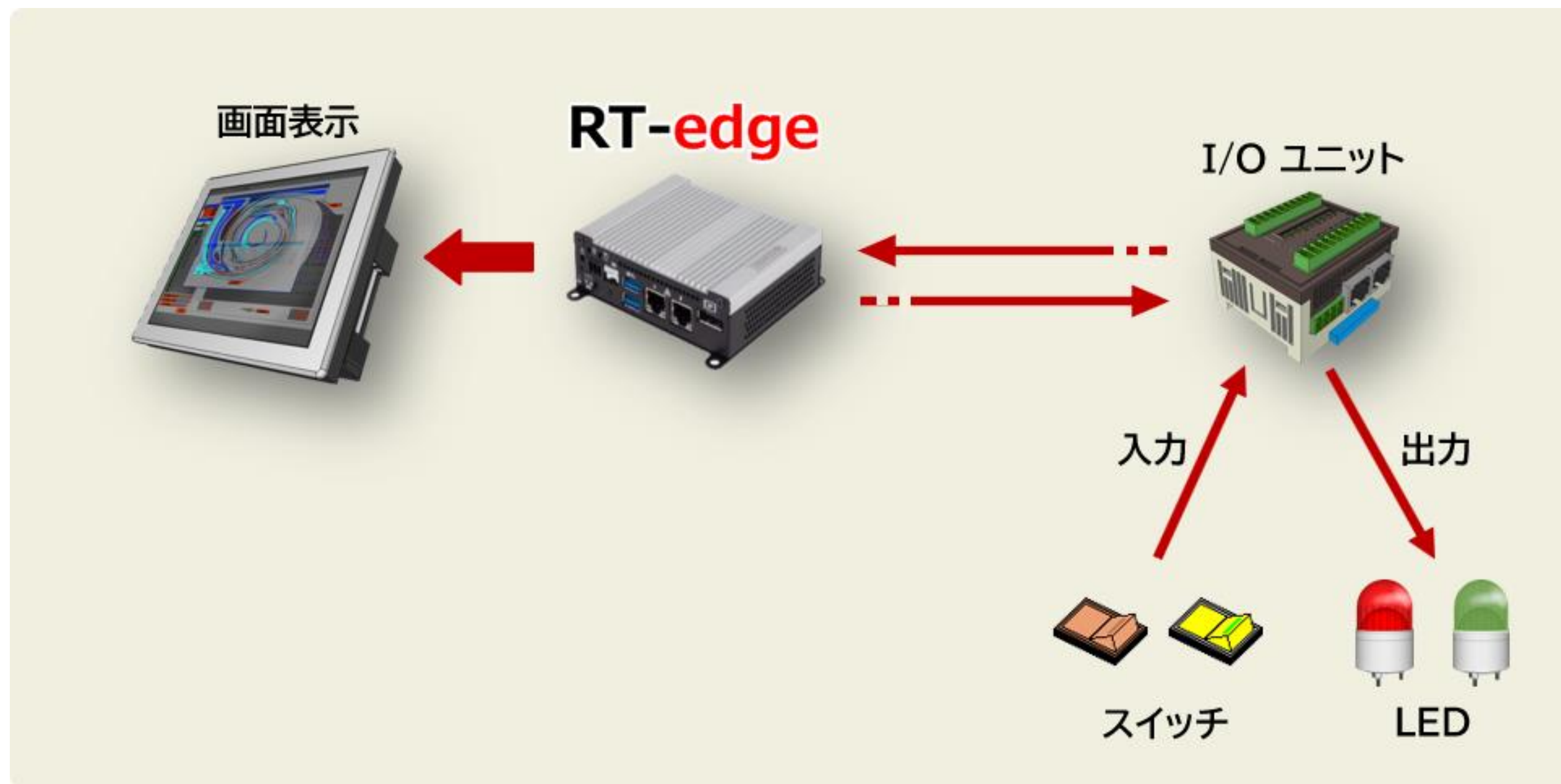
2つのスイッチと画面のボタン操作から、2つのLED制御する構成です。

ソフトウェアPLCで2つのスイッチ操作と画面のボタン操作を受け取り、

2つのLEDに対して、点灯/消灯/フリッカー動作のいずれかを指令します。

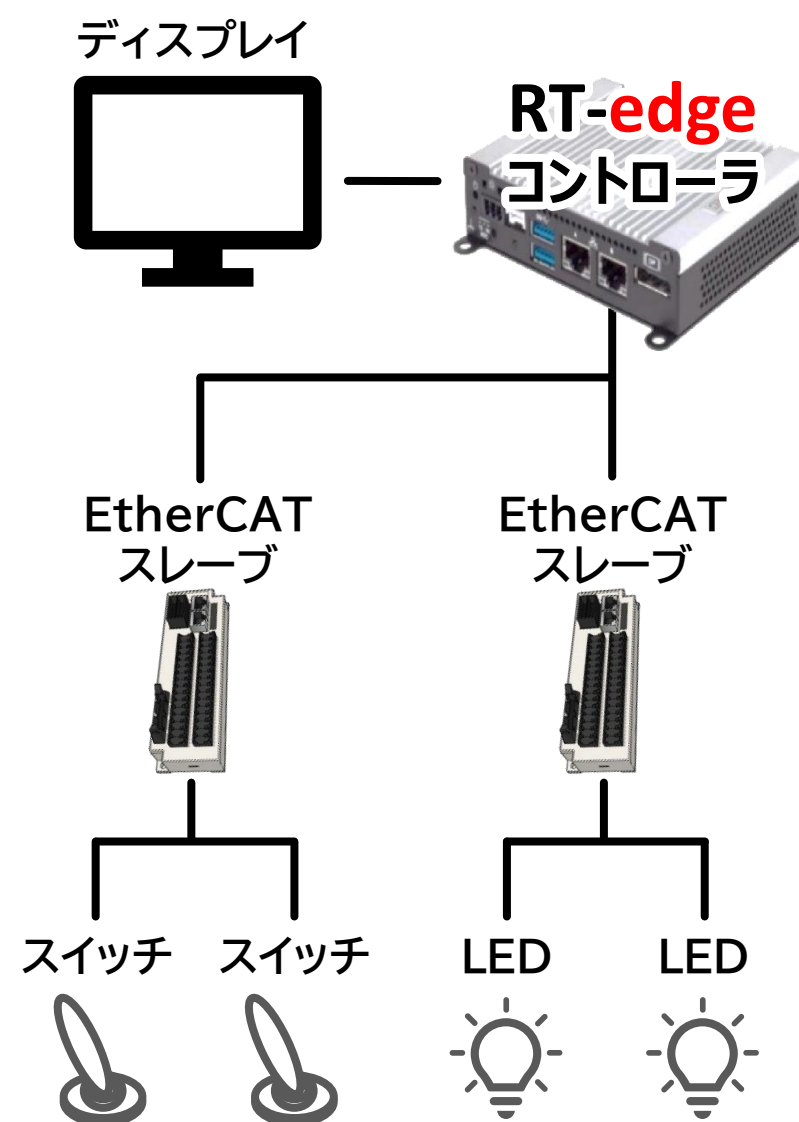
実現したいこと

1. 2つのスイッチが ON にされた時、1つ目のLED を点灯します
2. 画面のボタンが押された時、2つ目のLED を点滅します



実体配線図

1. I/O をソフトウェアPLC で制御し、I/O状態をディスプレイに表示します
2. I/O に EtherCATスレーブのデジタル入力ユニットに、2つのスイッチを繋ぎます
3. I/O に EtherCATスレーブのデジタル出力ユニットに、2つのLEDを繋ぎます



画面イメージ

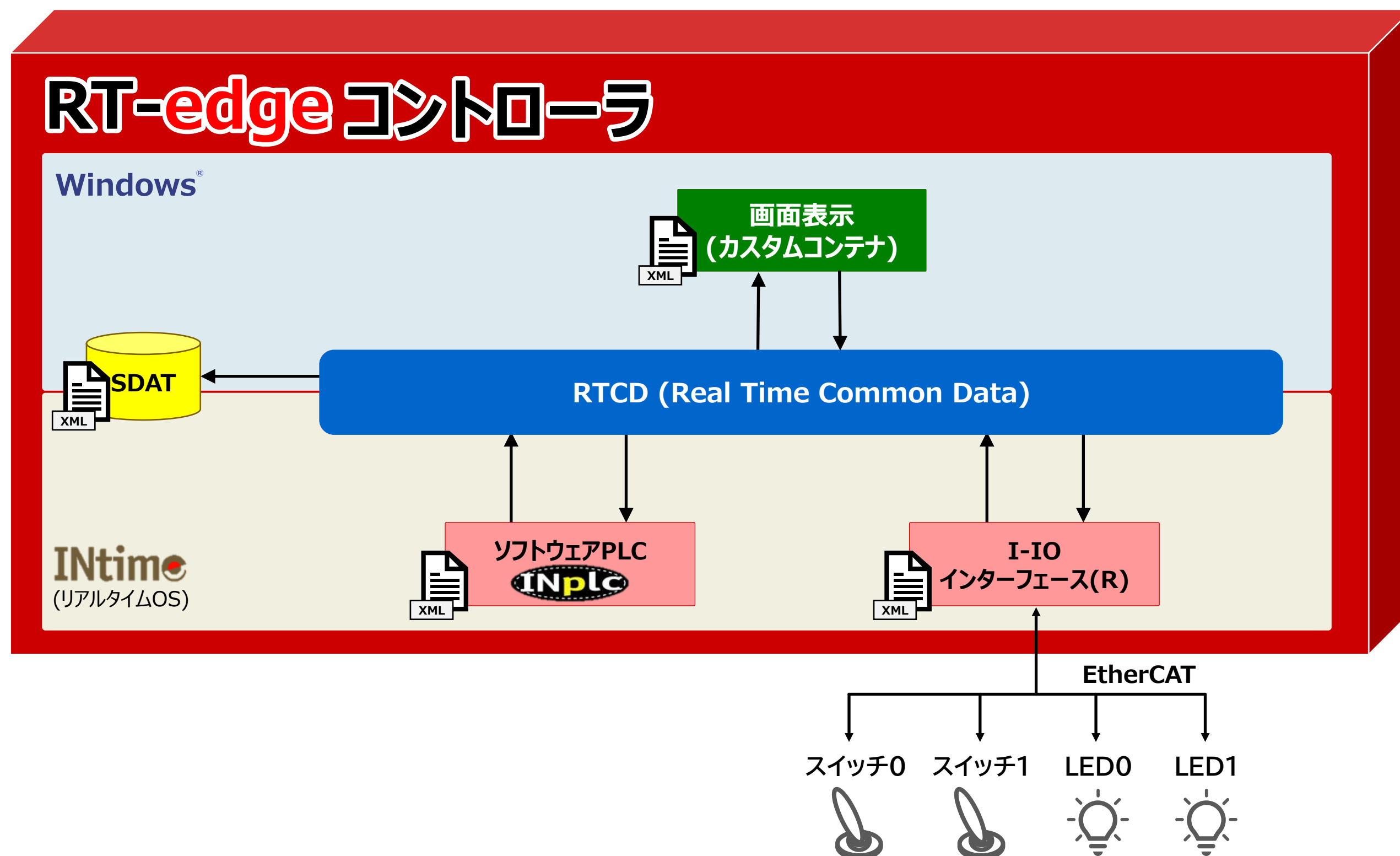
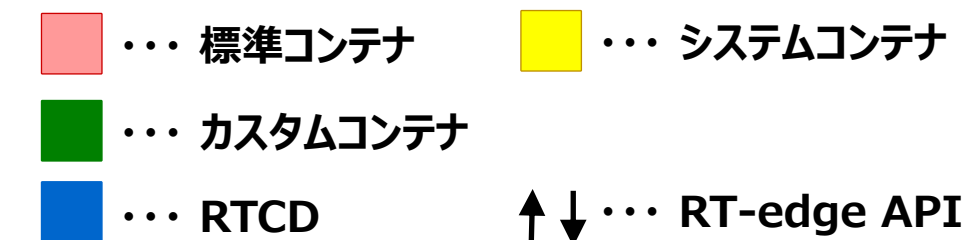
1. ディスプレイに1画面を表示します
2. ①に、スイッチ 2 つのON/OFF状態を示すオブジェクトを配置します
3. ②に、LED 0 の点灯/消灯状態を示すオブジェクトを配置します
4. ③に、画面からLED 1 のフリッカー点灯/消灯を操作するボタンを配置します
5. ④に、LED 1 の点灯/消灯状態を示すオブジェクトを配置します
6. ③のボタンだけ画面から操作可能になります



内部構成図

1. RT-edge の使用コンテナ

- ① 画面表示 (カスタムコンテナ)
- ② ソフトウェアPLC
- ③ I-I/Oインターフェース(R)
- ④ SDAT



使用タグ情報

No.	タグ名	説明
1	SWITCH_0	スイッチ0 の値(ONの時 1、 OFFの時 0)
2	SWITCH_1	スイッチ1 の値(ONの時 1、 OFFの時 0)
3	LED_0	LED0 の値(ONの時 1、 OFFの時 0)
4	LED_1	LED1 の値(ONの時 1、 OFFの時 0)
5	DISP_BUTTON	画面の「点灯ボタン」(ONの時 1、 OFFの時 0)
6	EDGESYSTEM.CurrentTime	システム時刻

挙動

1. 画面表示 (カスタムコンテナ)

- ① RTCDから、SWITCH_0, SWITCH_1, LED_0, LED_1 タグの値を取得し、画面へ表示します
- ② 画面の「点灯ボタン」の状態を、RTCDにある DISP_BUTTONタグへ、値をセットします

2. ソフトウェアPLC

- ① RTCDから、SWITCH_0, SWITCH_1, DISP_BUTTON タグの値を取得します
- ② SWITCH_0 と SWITCH_1 の値の両方が 1 の時、LED_0タグ へ 1 をセットします
- ③ SWITCH_0 と SWITCH_1 の値のどちらかが 0 の時、LED_0タグ へ 0 をセットします
- ④ DISP_BUTTON の値が 1 の時、LED点滅動作を開始し、LED_1 タグへ 0 と 1 を一定時間毎に交互にセットします
- ⑤ DISP_BUTTON の値が 0 の時、LED点滅動作を停止し、LED_1 タグへ 0 をセットします

3. I-I/Oインターフェース

- ① RTCDから、LED_0 タグの値を取得し、EtherCATスレーブの LED0 へ出力します
- ② RTCDから、LED_1 タグの値を取得し、EtherCATスレーブの LED1 へ出力します
- ③ EtherCATスレーブから スイッチ0 の接点状態を取得し、RTCD の SWITCH_0 タグへセットします
- ④ EtherCATスレーブから スイッチ1 の接点状態を取得し、RTCD の SWITCH_1 タグへセットします

4. データベース(DB)

- ① スイッチ0, スイッチ1, LED0, LED1, 点灯ボタン, システム時刻 のデータをDBへ書き込みます

タグ読み書き処理(一部抜粋)

1. 画面表示では、タグを読み書きする処理をコーディングしています。
2. タグの読み込みには、RT-edge API の EgTagRead を使用しています
3. タグの書き込みには、RT-edhe API の EgTagWrite を使用しています

```
// Tag: SWITCH_0 の値取得
if (null != strTagSw0)
{
    if (0 == EgTEMP.EGAPI.EgTagRead(strTagSw0, ref objValue, false))
    { // 0: 成功か?
        rdoSwitch_0.Checked = (bool)objValue;
    }
    else
    {
        rdoSwitch_0.Checked = false;
    }
}

// Tag: SWITCH_1 の値取得
if (null != strTagSw1)
{
    if (0 == EgTEMP.EGAPI.EgTagRead(strTagSw1, ref objValue, false))
    { // 0: 成功か?
        rdoSwitch_1.Checked = (bool)objValue;
    }
    else
    {
        rdoSwitch_1.Checked = false;
    }
}

// Tag: LED_0 の値取得
if (null != strTagLed0)
{
    if (0 == EgTEMP.EGAPI.EgTagRead(strTagLed0, ref objValue, false))
    { // 0: 成功か?
        rdoLight_0.Checked = (bool)objValue;
    }
    else
    {
        rdoLight_0.Checked = false;
    }
}
```


XML内容(一部抜粋)

1.<?xml version="1.0" encoding="utf-8"?>

2.<RTedge xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">

3.<Tags>

4.<Tag Name="SERVICE.EgINplc.Mode_MArea" Type="3" Size="1" Value="1" Comment=""/>

5.<Tag Name="SERVICE.EgINplc.Mode_IArea" Type="3" Size="1" Value="2" Comment=""/>

6.<Tag Name="SERVICE.EgINplc.AutoRun" Type="1" Size="1" Value="1" Comment=""/>

7.<Tag Name="SERVICE.EgINplc.Cycle" Type="7" Size="1" Value="1" Comment=""/>

8.<Tag Name="SERVICE.EgINplc.InPriority" Type="3" Size="1" Value="181" Comment=""/>

9.<Tag Name="SERVICE.EgINplc.OutPriority" Type="3" Size="1" Value="179" Comment=""/>

10.<Tag Name="SERVICE.EgINplc.TrigPriority" Type="3" Size="1" Value="175" Comment=""/>

11.<Tag Name="SERVICE.EgINplc.PlcChkCycle" Type="7" Size="1" Value="100" Comment=""/>

12.<Tag Name="SERVICE.EgINplc.PlcChkPriority" Type="3" Size="1" Value="196" Comment=""/>

13.<Tag Name="SERVICE.EgINplc.IAreaPriority" Type="3" Size="1" Value="177" Comment=""/>

14.<Tag Name="SERVICE.EgINplc.IDataPriority" Type="3" Size="1" Value="178" Comment=""/>

15.<Tag Name="SERVICE.EgINplc.QAreaPriority" Type="3" Size="1" Value="176" Comment=""/>

16.<Tag Name="SWITCH_0" Type="1" Size="1" Address="%IX0.0" Comment="Iエリアに割り付けるスイッチ0入力情報" />

17.<Tag Name="SWITCH_1" Type="1" Size="1" Address="%IX0.1" Comment="Iエリアに割り付けるスイッチ1入力情報" />

18.<Tag Name="LED_0" Type="1" Size="1" Address="%QX0.0" Comment="Qエリアに割り付けるLED0点灯情報" />

19.<Tag Name="LED_1" Type="1" Size="1" Address="%QX0.1" Comment="Qエリアに割り付けるLED1点灯情報" />

20.<Tag Name="DISP_BUTTON" Type="1" Size="1" Address="%IX8.0" Comment="Iエリアに割り付ける画面ボタン情報" />

21.</Tags>

22.<Services>

23.<Service Name="EgINplc">

24.<TagRefs_IN Comment="ユーザー タグと Qエリア を紐づける定義部分">

25.<TagRef Name="LED_0" Comment="Qエリアに割り付けるLED0点灯情報" />

26.<TagRef Name="LED_1" Comment="Qエリアに割り付けるLED1点灯情報" />

27.</TagRefs_IN>

28.<TagRefs_OUT Comment="ユーザー タグと Iエリア を紐づける定義部分">

29.<TagRef Name="SWITCH_0" Comment="Iエリアに割り付けるスイッチ0入力情報" />

30.<TagRef Name="SWITCH_1" Comment="Iエリアに割り付けるスイッチ1入力情報" />

31.<TagRef Name="DISP_BUTTON" Comment="Iエリアに割り付ける画面ボタン情報" />

32.</TagRefs_OUT>

33.</Service>

34.</Services>

35.</RTedge>

コンテナ設定情報:
コンテナで予約しているタグで、コンテナによって様々な設定が行えます

タグ登録:
今回使用するタグを登録しています
ここでは、ソフトウェアPLC のI/Oアドレスとの割り付けを、Addressプロパティで定義しています。

タグの入出力方向設定:
今回使用するタグと、データの方法を定義しています。
ここでは、ソフトウェアPLC のI/Oの内、Outputと紐づけたタグ(LED_0, LED_1)をRT-edge の共有メモリ: RTCDへ入力することから、TagRefs_IN へ定義しています。

タグの入出力方向設定:
今回使用するタグと、データの方法を定義しています。
ここでは、ソフトウェアPLC のI/Oの内、Inputと紐づけたタグ(SWITCH_0, SWITCH_1, DISP_BUTTON)をRT-edge の共有メモリ: RTCDへ出力することから、TagRefs_OUTへ定義しています。

2023/6/14

株式会社マイクロネット 篠崎 勝利

9