RTミドルウェアツール紹介：利用可能なRTコンポーネントやツールについて

宮本 信彦
国立研究開発法人産業技術総合研究所
インテリジェントシステム研究部門

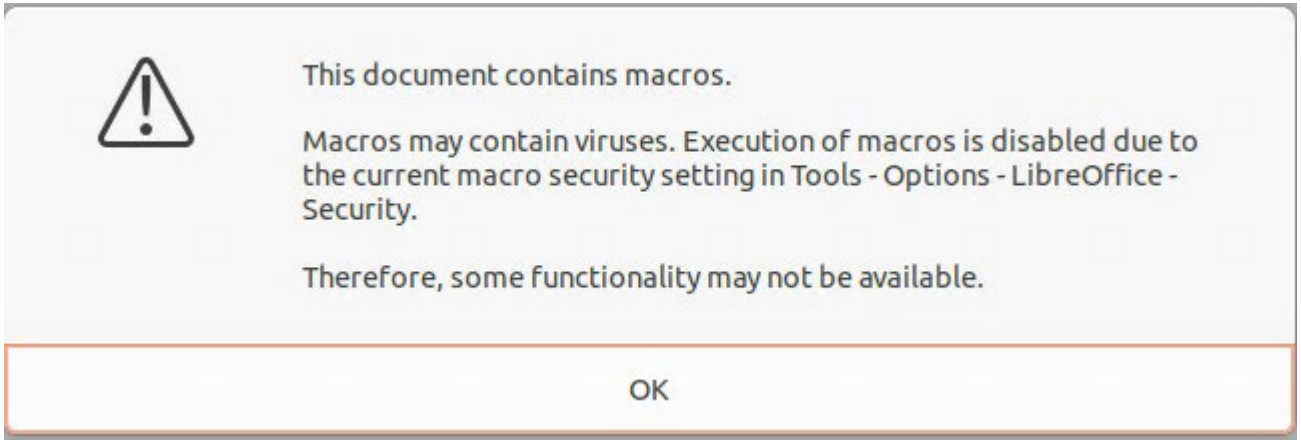
ツールのダウンロード

- 講義資料
  - https://openrtm.org/openrtm/sites/default/files/7300/2025SummerCamp-04.pdf
- Windowsの場合
  - USBメモリで配布
- Ubuntuの場合
  - インストール作業が必要のため次のスライドで説明します
- 上記URLはSlackでお知らせします

ツールのインストール(Ubuntu)

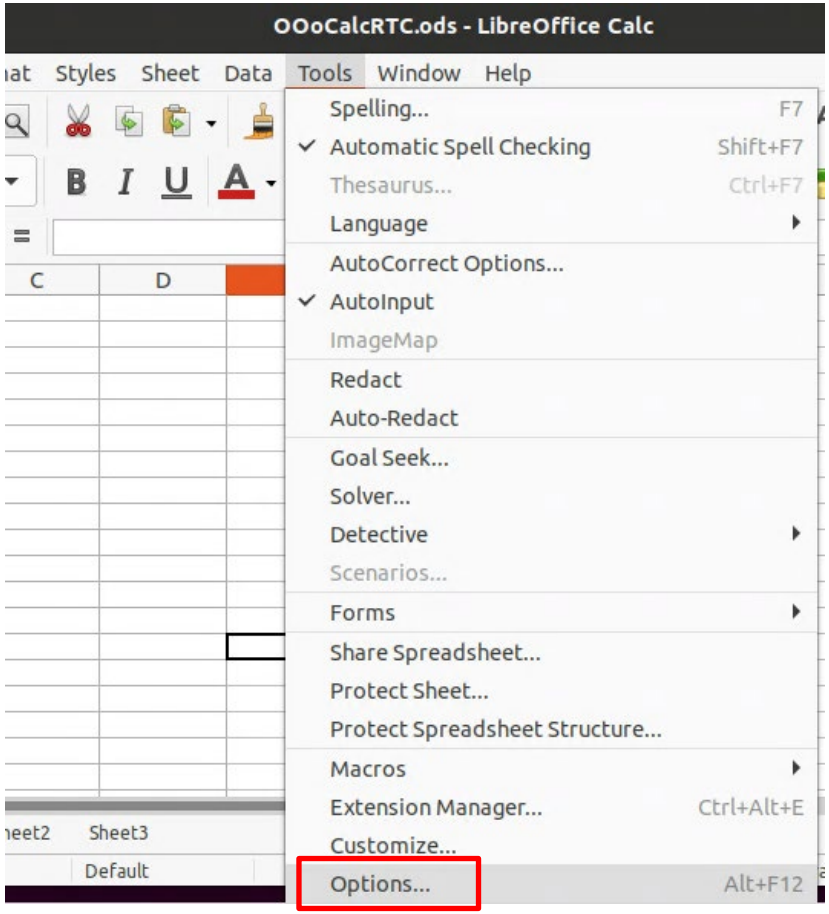
- 以下のコマンドを実行する
  - sudo apt install libreoffice-script-provider-python
  - git clone https://github.com/Nobu19800/OOoRTCs
  - cd OOoRTCs
  - sh install.sh

- OOoRTCs/OOoCalcRTC/OOoCalcRTC.odsをダブルクリックして開く
  - 以下の画面が表示されたらセキュリティの設定を変更する



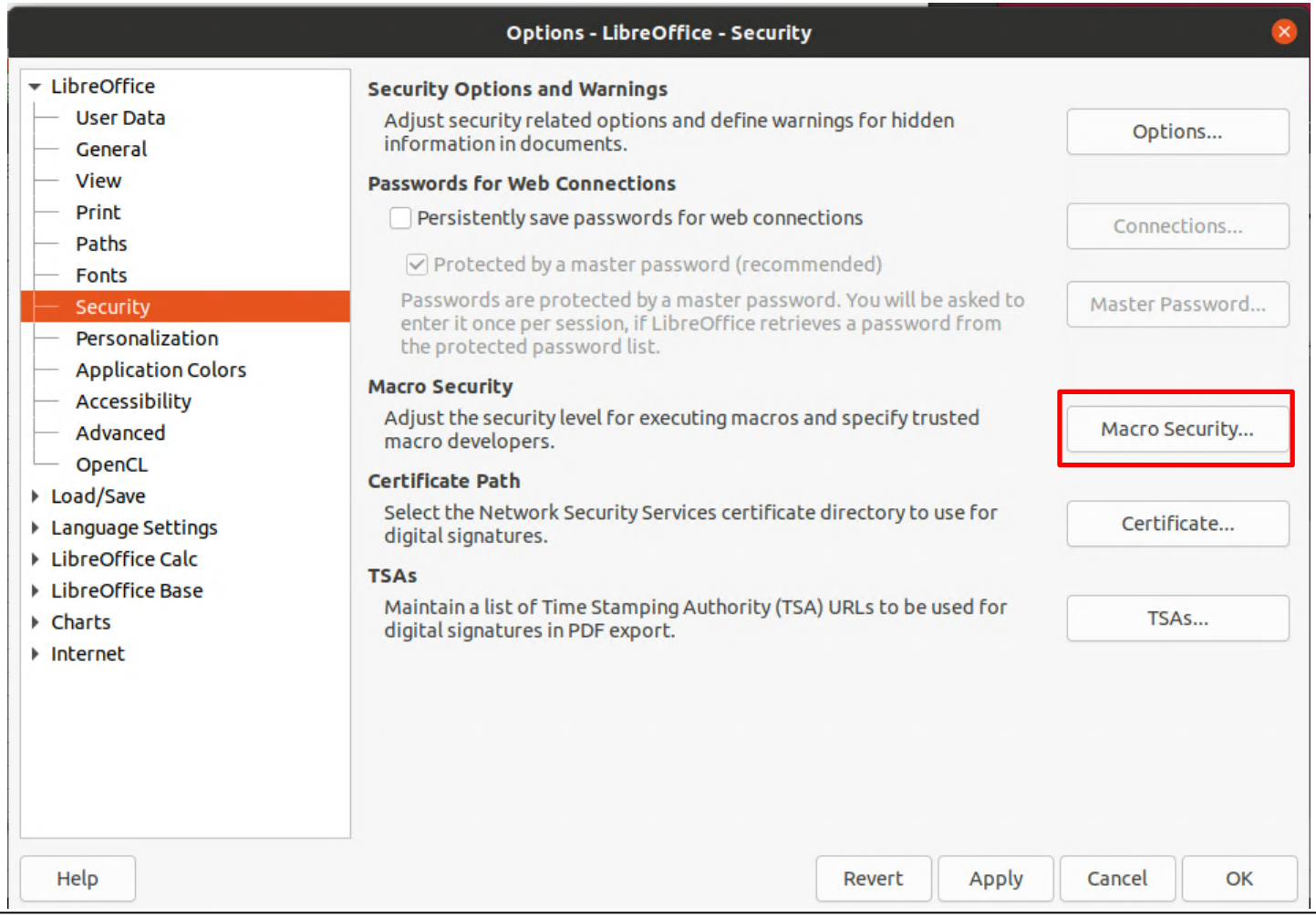
ツールのインストール(Ubuntu)

- Tools -> Options



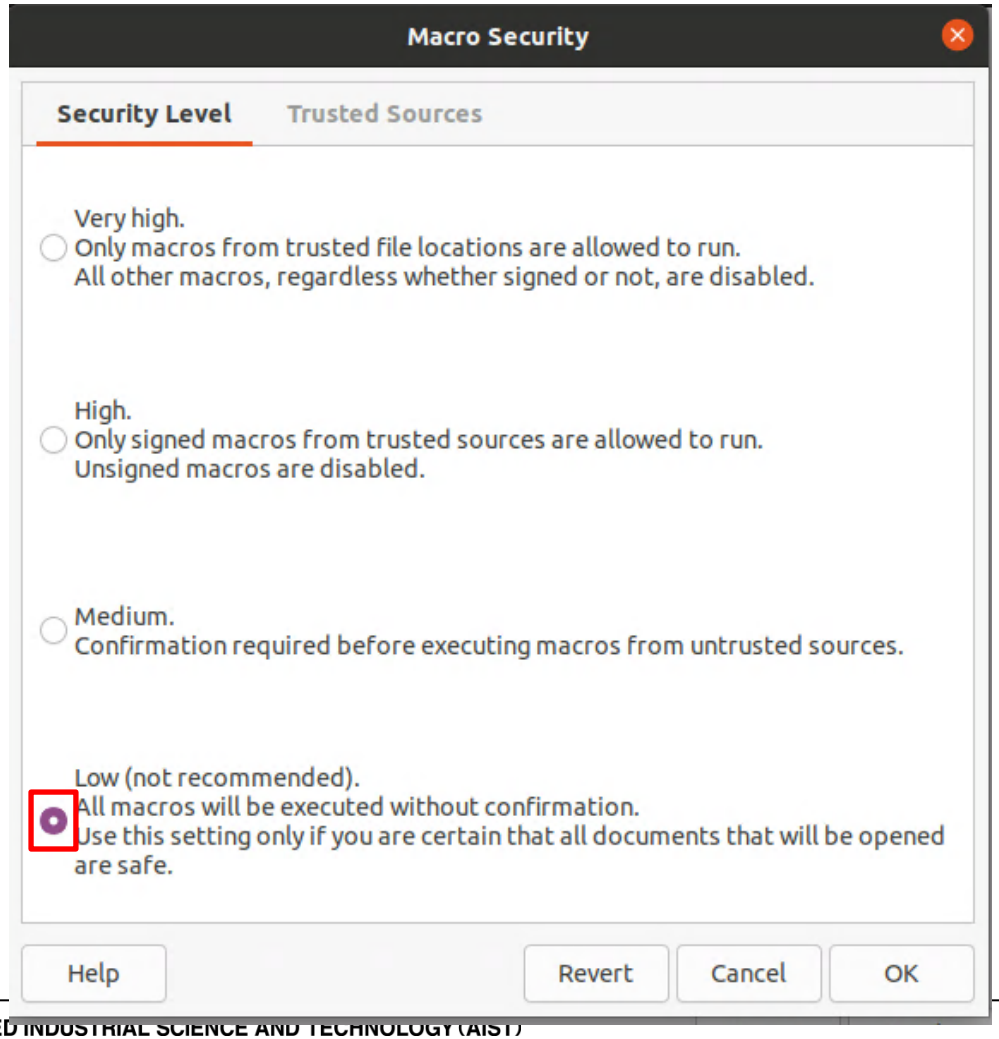
ツールのインストール(Ubuntu)

- Security -> Macro Security…



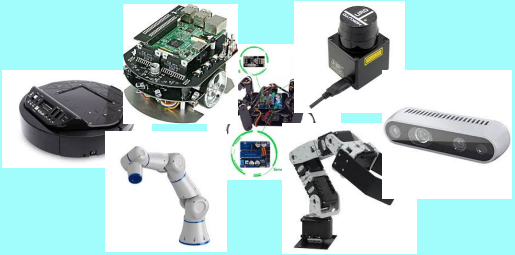
ツールのインストール(Ubuntu)

- Security LevelをLowに設定する
- 設定後、Calcは再起動する



これからどうやってロボットシステムを開発するのか？

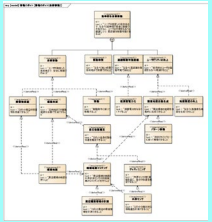
ハードウェア、既存のRTCの動作確認

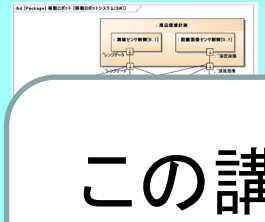


- RTCの導入方法、使用方法の調査
- RTCのインストール
- RTCの入出力の確認

設計

要求図　内部ブロック図

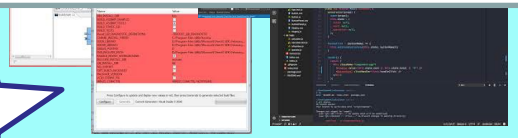




- システムの要
- 使用するハ
- 再利用するR

昨日説明があった

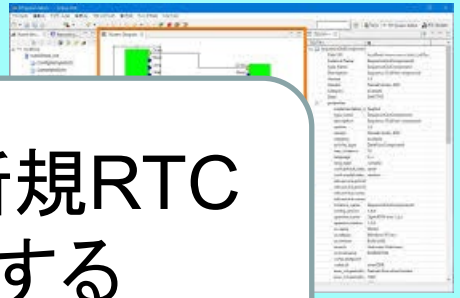
RTC Builderの利用方法は講習会で説明済み



- RTCの詳細な仕様を決める
- コーディング、ビルド
- RTCの動作確認、デバッグ

RT System Editorの利用方法は講習会で説明済み

ムの構築、作確認



- RT System Editor
- rtshell
- 動作確認

rtshellについては前の講義で説明があった

この講義ではRTCの動作確認、新規RTCの開発で有用なツールを紹介する

紹介するツール

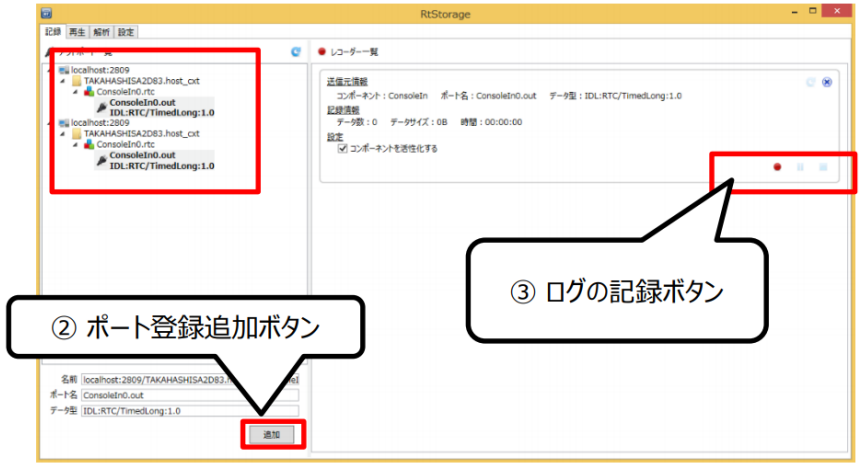
- **RtStorage**
  - 元々セックでOpenRTM.NETを開発していた人が作ったツール
  - OutPortから出力されたデータをファイルに記録
  - 保存したデータを、RTコンポーネントのInPortに対して再生

- **表計算ソフトとRTCを連携させるためのツール**
  - Excel、LibreOfficeCalcのセルの値をデータポートから入出力する
  - 新規に開発したRTCの動作確認が簡単にできる
    - OutPortから出力された値の確認
    - InPortに任意の値を入力したときの挙動
  - 既存のRTCについても、動作がよく分からない場合に確認できる

RtStorage

RtStorageの概要

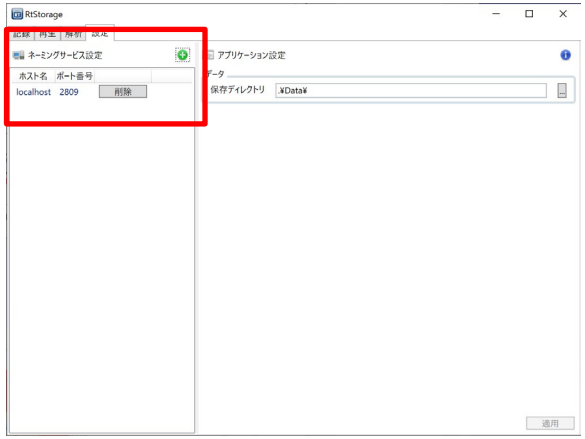
- できる事
  - OutPortから出力されたデータをファイルに記録
  - 保存したデータを、RTコンポーネントのInPortに対して再生
  - 上記の保存、再生がGUIで簡単にできる



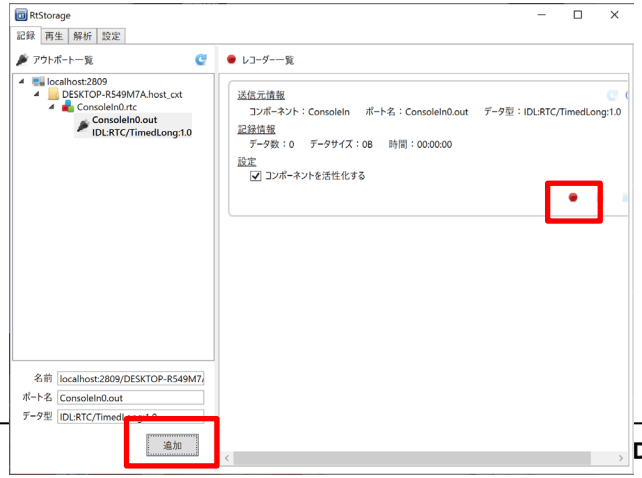
- インストール手順
  - 以下からRtStorage-ver.0.3.1.zipをダウンロードして展開
    - https://github.com/zoetrope/RtStorage/releases/tag/ver.0.3.1
  - RtStorage.slnをVisualStudioで開いてソリューションをビルド
  - RtStorage¥bin¥ReleaseのRtStorage.exeを実行すると起動する

RtStorageの使い方

1. RtStorage.exeを実行する
2. ネームサーバーを登録する

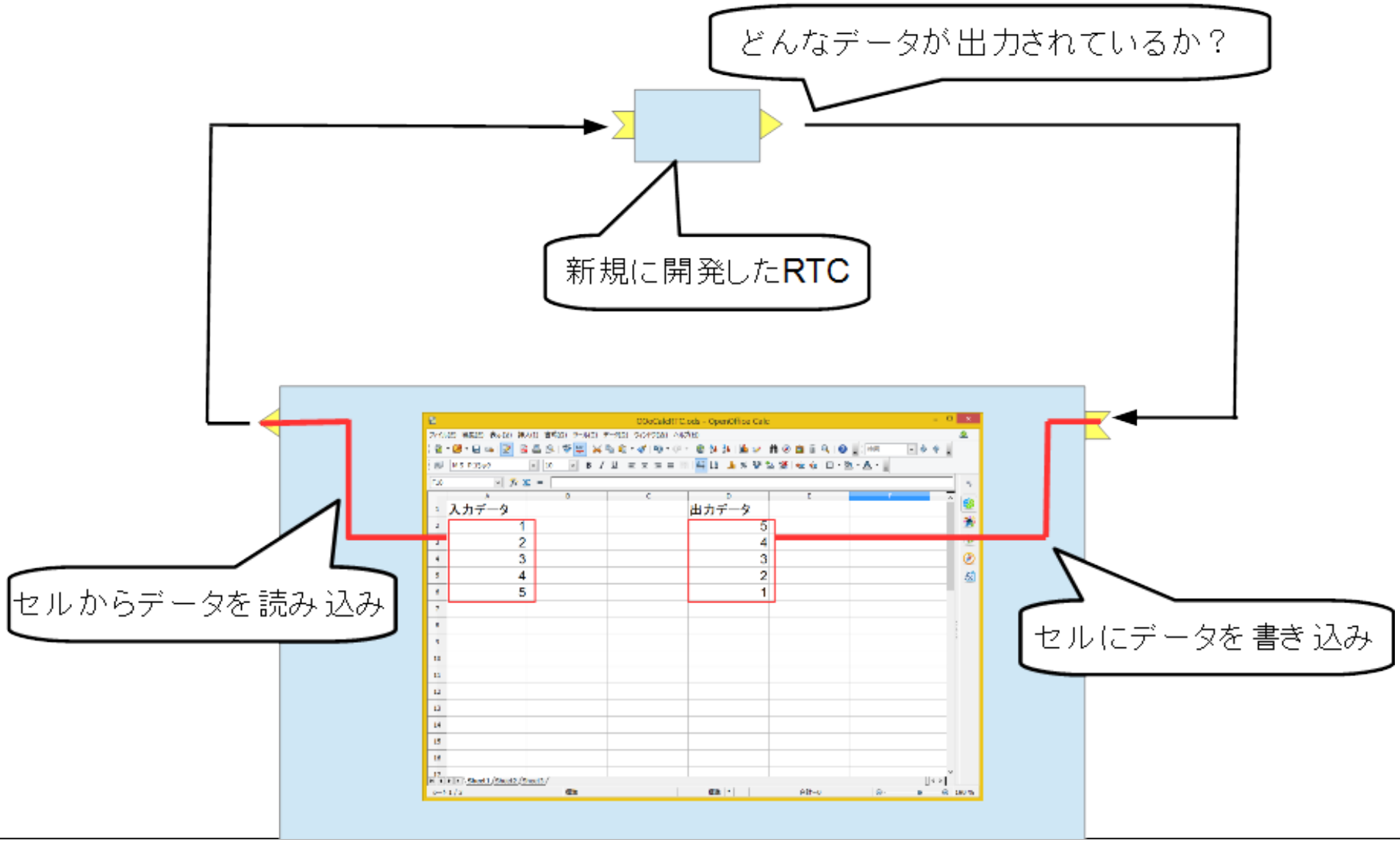


3. ポートを登録追加、記録の開始

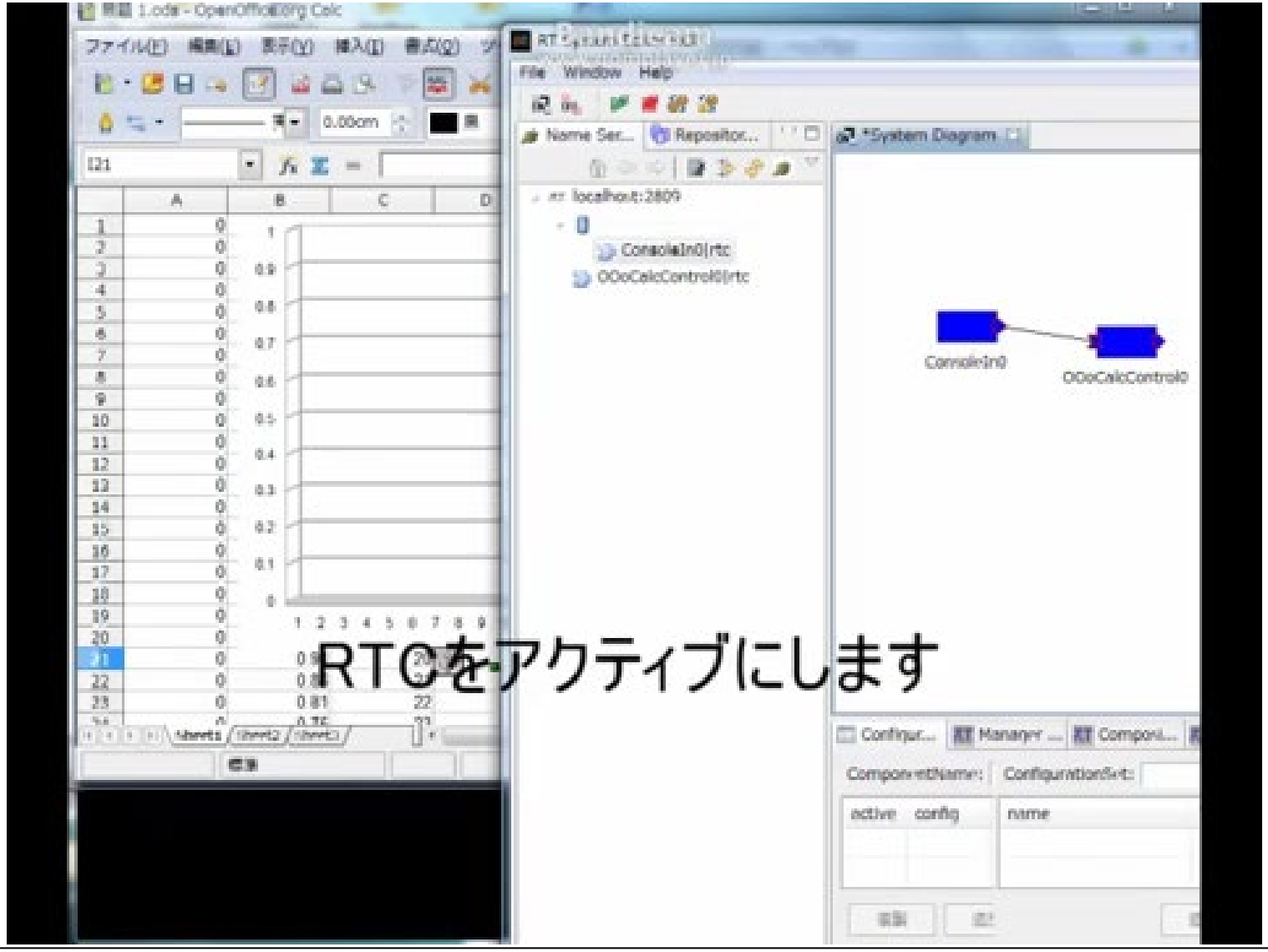


表計算ソフトとRTCの連携

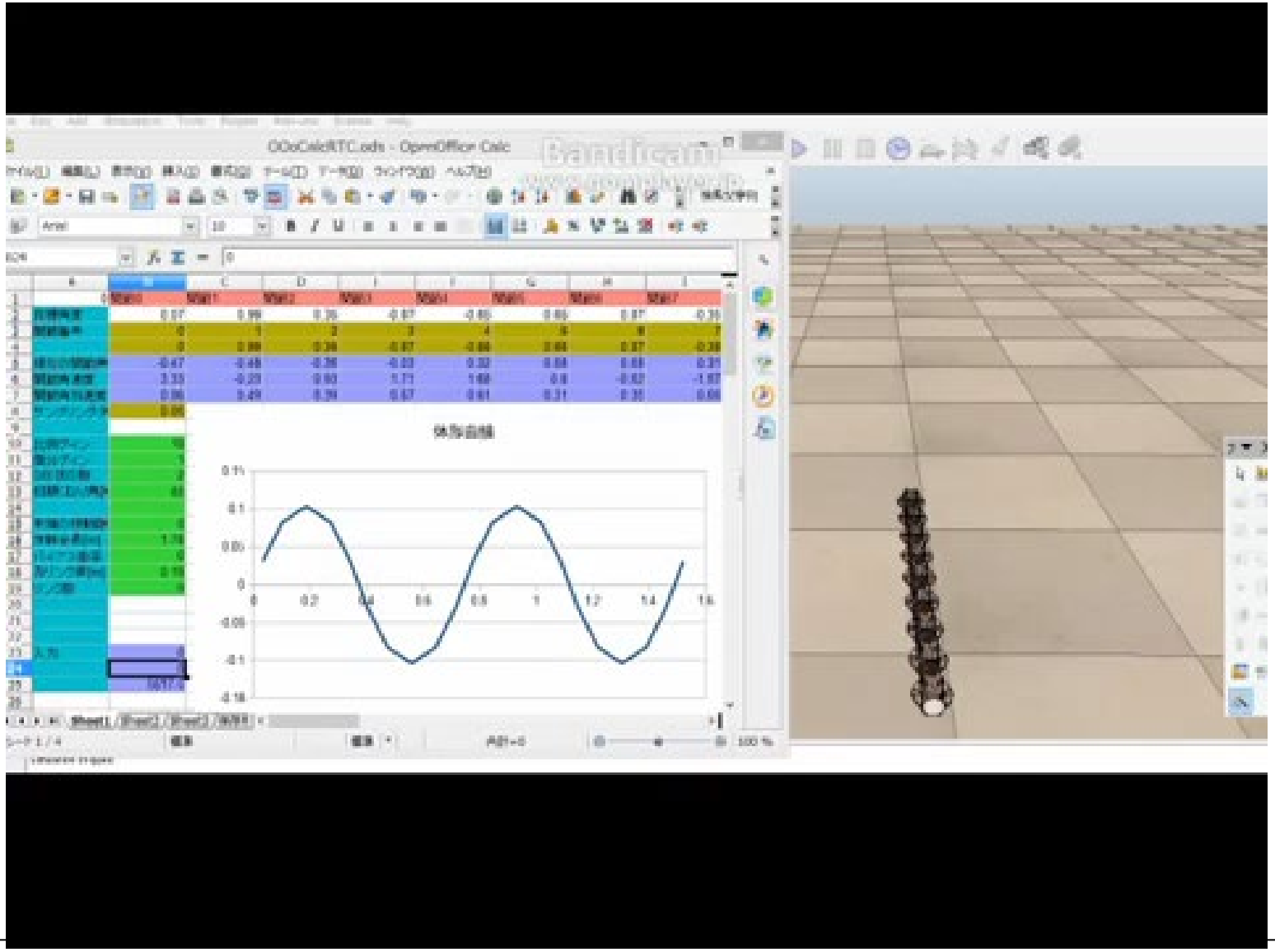
表計算ソフトによるデータ入出力



デモ動画

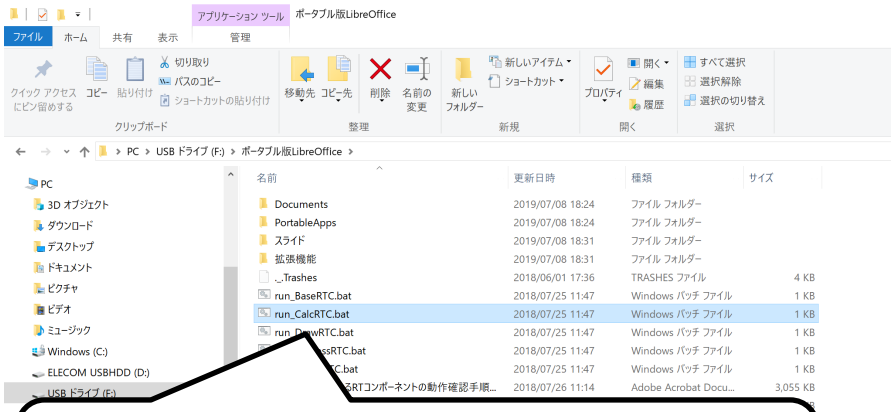

RTCをアクティブにします
File Window Help
System Diagram
localhost:2809
Console1n0|rtc
OOoCalcControl0|rtc
Console1n0
OOoCalcControl0
active
config
name

デモ動画

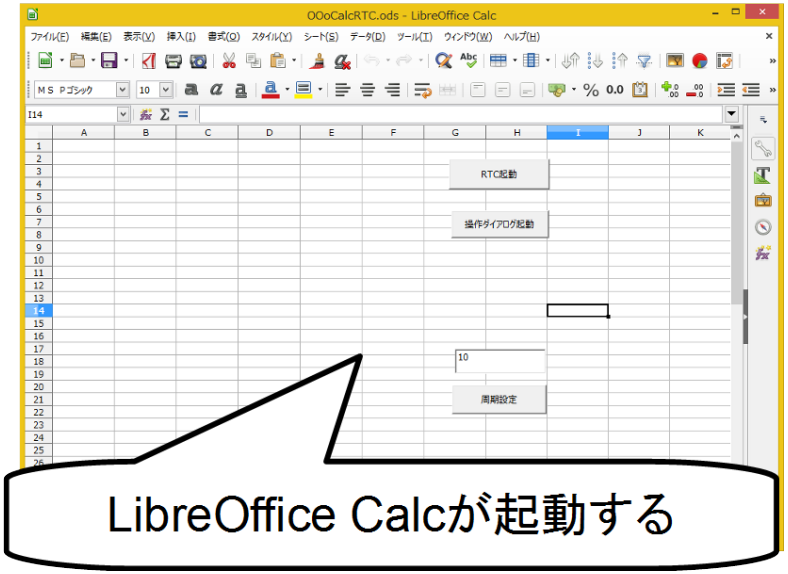


ポータブル版LibreOffice対応RTC

- 配布ファイルに以下のソフトウェアを同梱(Windows)
  - ポータブル版LibreOffice
  - OpenRTM-aist-Python
  - OpenOffice用RTコンポーネント
- 起動手順(Windows)



ポータブル版LibreOffice¥run_CalcRTC.bat
をダブルクリック

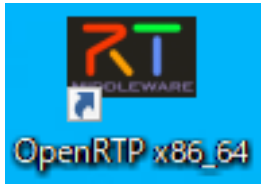


LibreOffice Calcが起動する

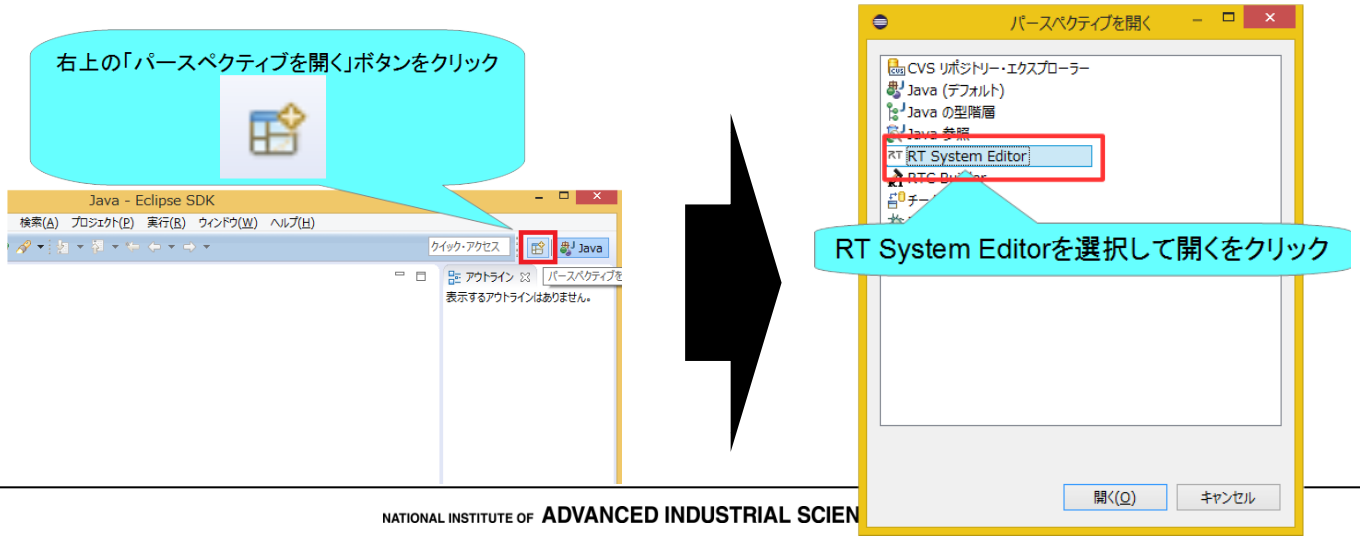
- 起動手順(Ubuntu)
  - OOoRTCs/OOoCalcRTC/OOoCalcRTC.odsをダブルクリック

事前準備

- OpenRTPを起動
  - Windows
    - デスクトップのショートカットをダブルクリックする

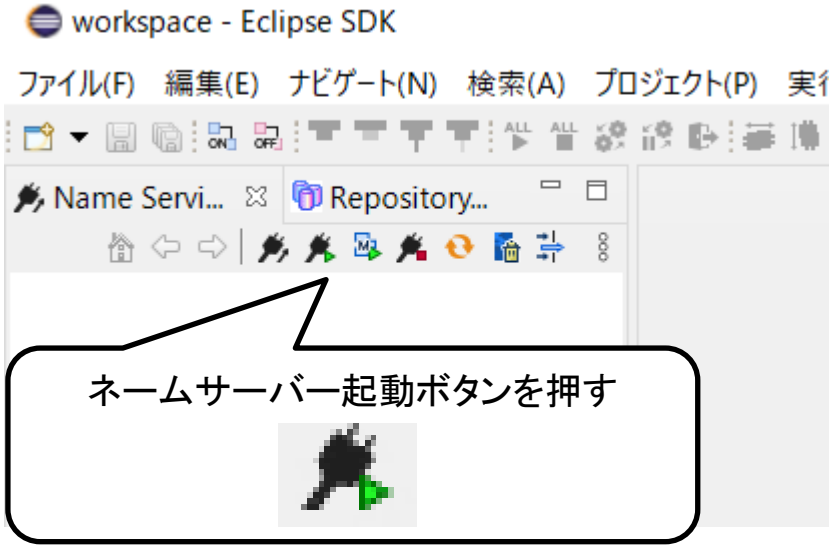


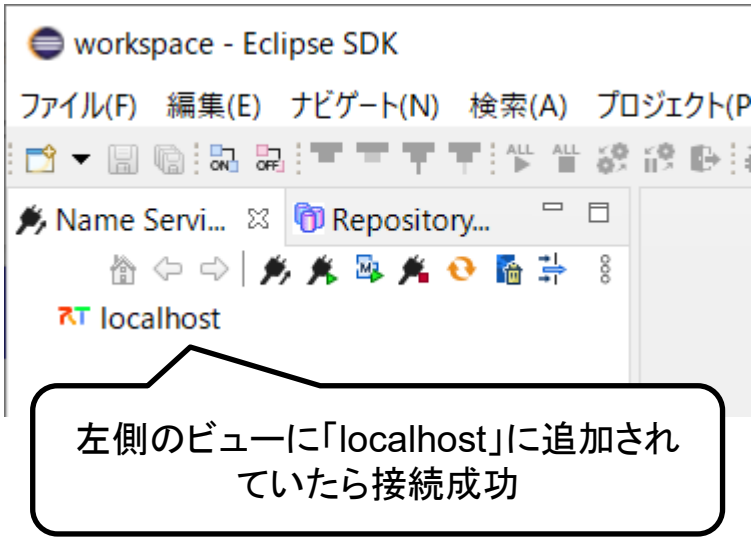
  - Ubuntu
    - 以下のコマンドを実行
      - openrtp2



事前準備

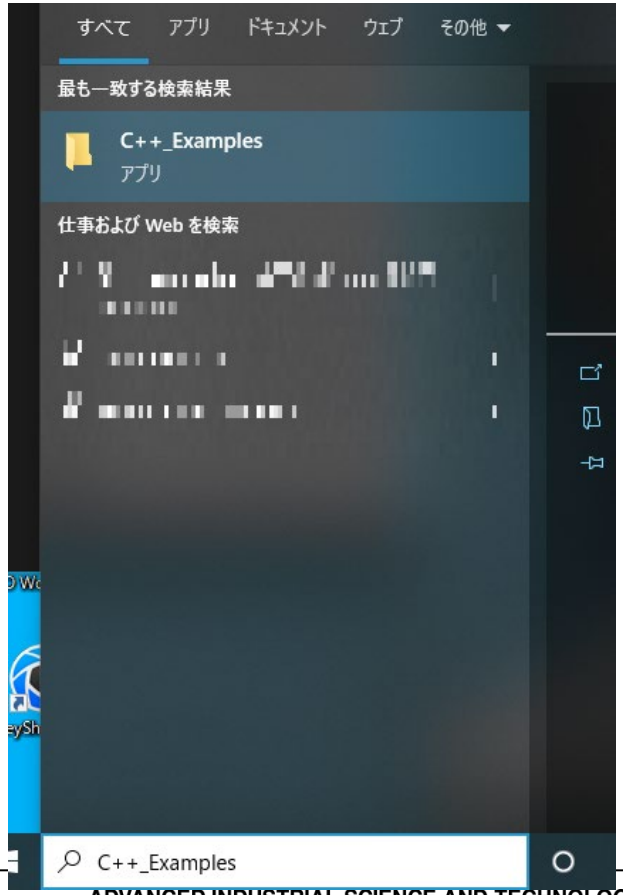
- ネームサーバーを起動





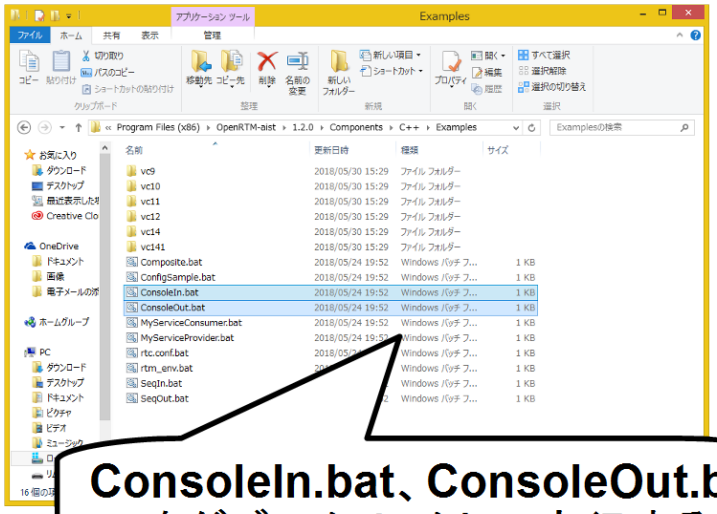
動作確認用のRTCを起動

- ConsoleIn、ConsoleOutのサンプルコンポーネントを起動する
  - Windows 10
    - 左下の「ここに入力して検索」にC++_Examplesと入力して、表示されたC++_Examplesをクリック

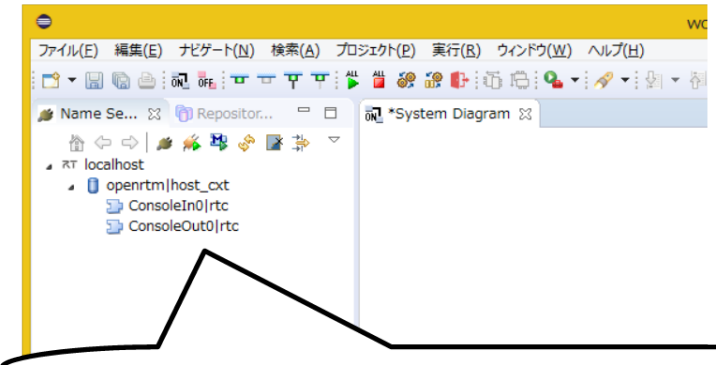


動作確認用のRTCを起動

- ConsoleIn、ConsoleOutのサンプルコンポーネントを起動する
  - Windows 10
    - 左下の「ここに入力して検索」にC++_Examplesと入力して、表示されたC++_Examplesをクリック



ConsoleIn.bat、ConsoleOut.bat
をダブルクリックして実行する



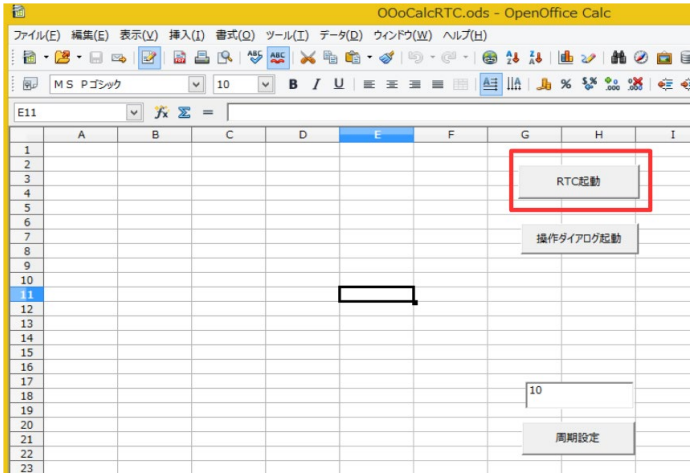
ネームサービスビューにConsoleIn、ConsoleOutが
表示されていれば起動成功

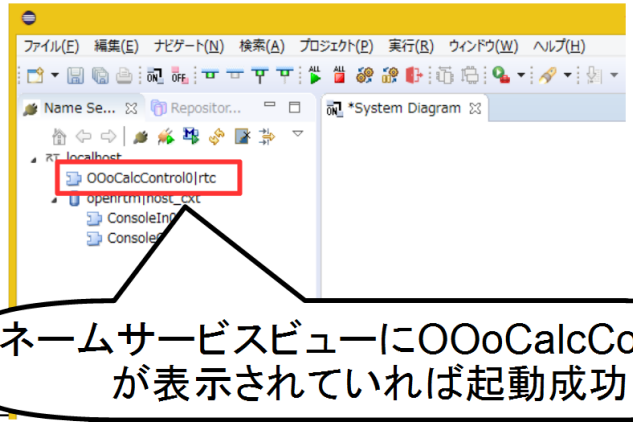
動作確認用のRTCを起動

- ConsoleIn、ConsoleOutのサンプルコンポーネントを起動する
  - Ubuntu
    - 以下のコマンドを実行
      - /usr/share/openrtm-2.0/components/c++/examples/ConsoleInComp
      - /usr/share/openrtm-2.0/components/c++/examples/ConsoleOutComp

Calc用RTCを起動

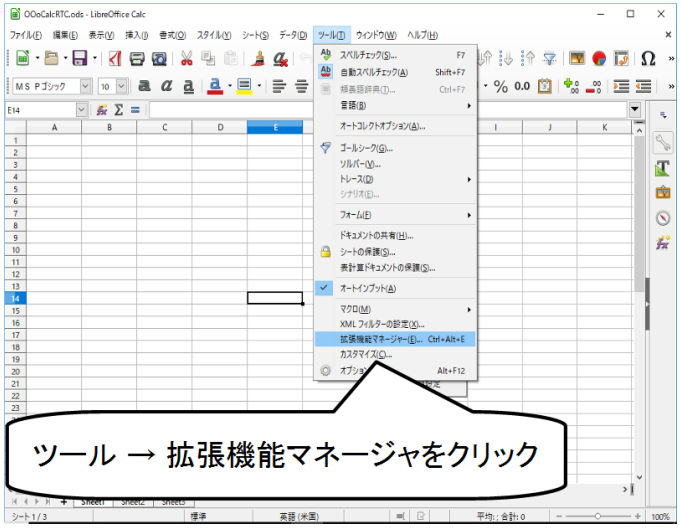
- LibreOffice Calcの「RTC起動」ボタンをクリックする

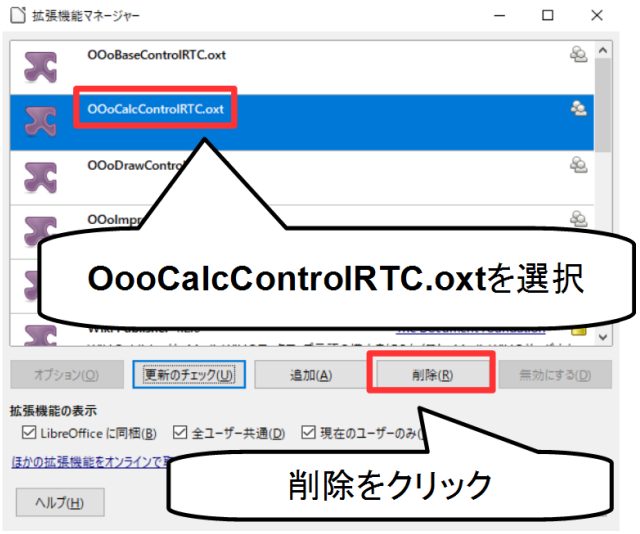


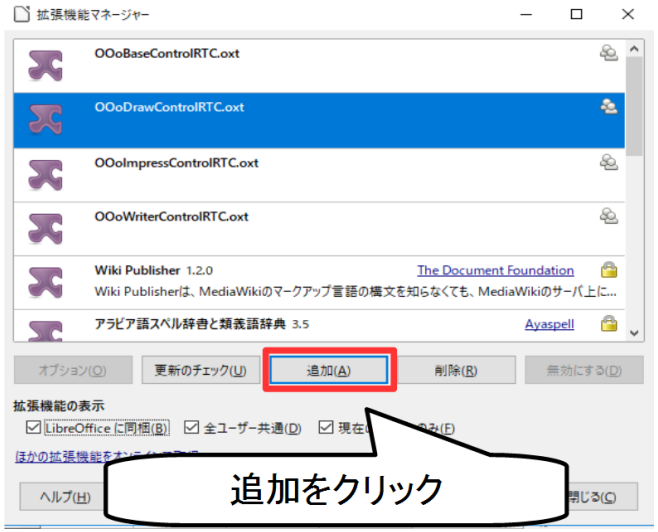


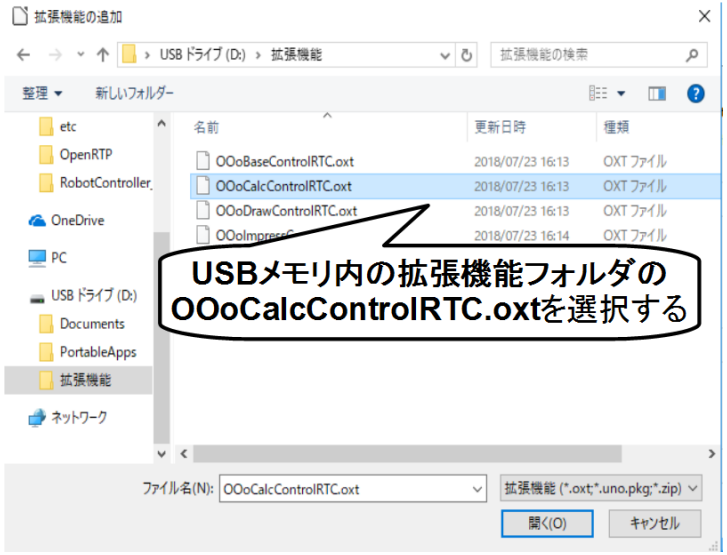
ネームサービスビューにOOoCalcControl0が表示されていれば起動成功

起動に失敗する場合


ツール → 拡張機能マネージャをクリック

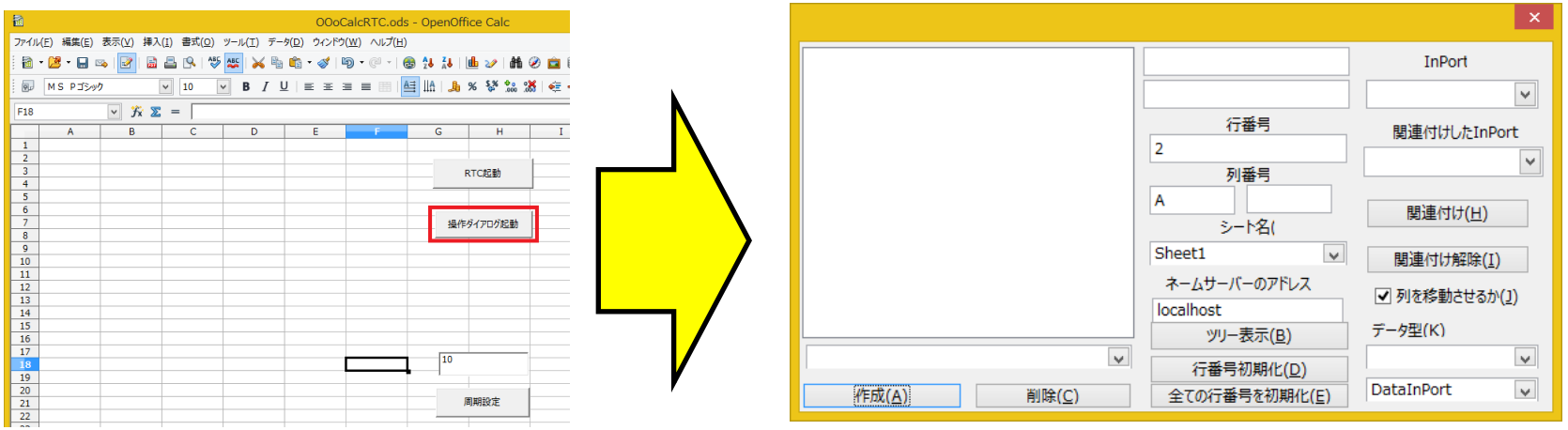

拡張機能マネージャー
OOoBaseControlRTC.oxt
OOoCalcControlRTC.oxt
OOoCalcControlRTC.oxtを選択
オプション(O)
更新のチェック(U)
追加(A)
削除(R)
無効にする(D)
拡張機能の表示
LibreOffice に同梱(B)
全ユーザー共通(D)
現在のユーザーのみ
削除をクリック
ヘルプ(H)


拡張機能マネージャー
OOoBaseControlRTC.oxt
OOoDrawControlRTC.oxt
OOoImpressControlRTC.oxt
OOoWriterControlRTC.oxt
Wiki Publisher 1.2.0
The Document Foundation
Wiki Publisherは、MediaWikiのマークアップ言語の構文を知らなくても、MediaWikiのサーバ上に...
アラビア語スペル辞書と類義語辞典 3.5
Ayaspell
オプション(O)
更新のチェック(U)
追加(A)
削除(R)
無効にする(D)
拡張機能の表示
LibreOffice に同梱(B)
全ユーザー共通(D)
ヘルプ(H)
追加をクリック

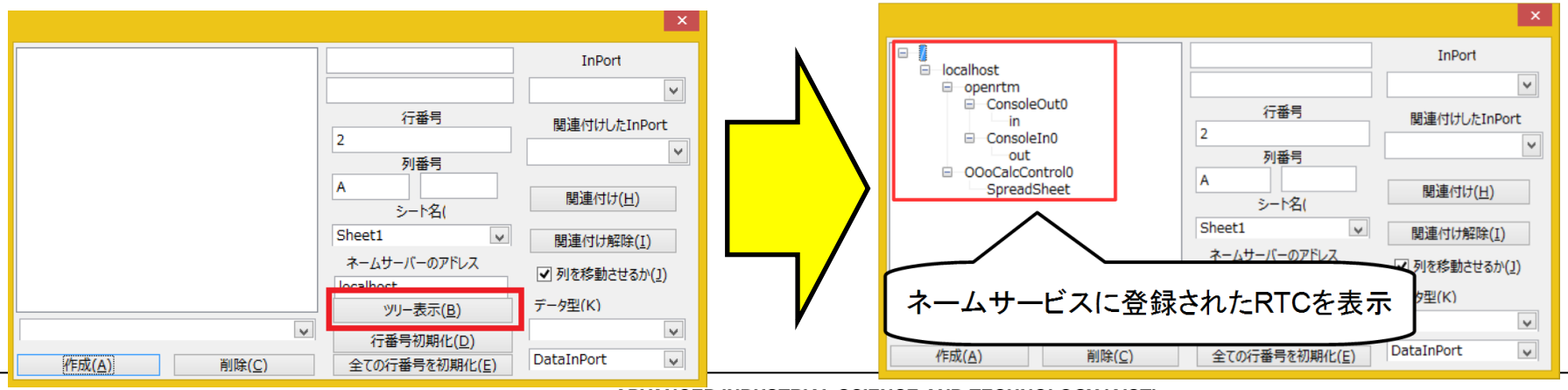

拡張機能の追加
USB ドライブ (D:) › 拡張機能
拡張機能の検索
整理
新しいフォルダー
etc
OpenRTP
RobotController
OneDrive
PC
USB ドライブ (D:)
Documents
PortableApps
拡張機能
ネットワーク
名前
更新日時
種類
OOoBaseControlRTC.oxt 2018/07/23 16:13 OXT ファイル
OOoCalcControlRTC.oxt 2018/07/23 16:13 OXT ファイル
OOoDrawControlRTC.oxt 2018/07/23 16:13 OXT ファイル
2018/07/23 16:14 OXT ファイル
USBメモリ内の拡張機能フォルダの
OOoCalcControlRTC.oxtを選択する
ファイル名(N): OOoCalcControlRTC.oxt
拡張機能 (*.oxt;*.uno.pkg;*.zip)
開く(O)
キャンセル

OutPortの接続

- LibreOffice Calcの「操作ダイアログ起動」ボタンをクリックする

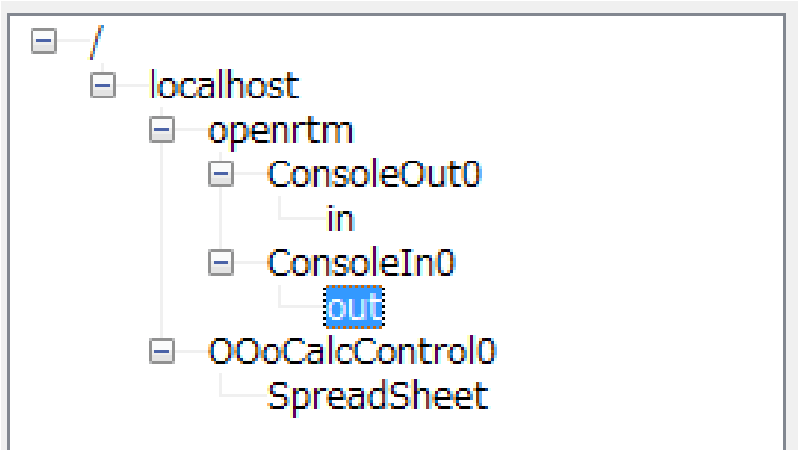


- 「ツリー表示」ボタンをクリックする

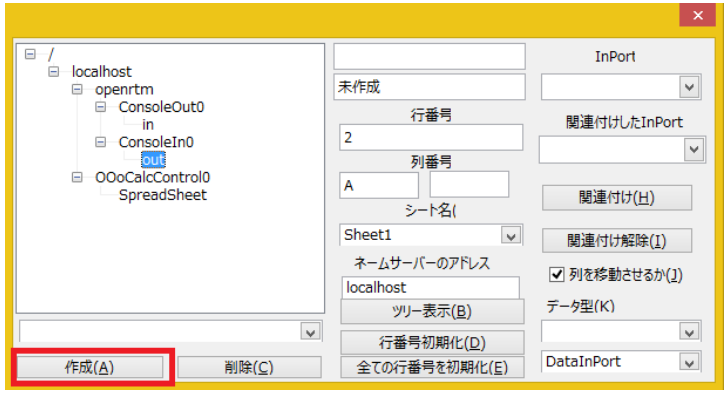


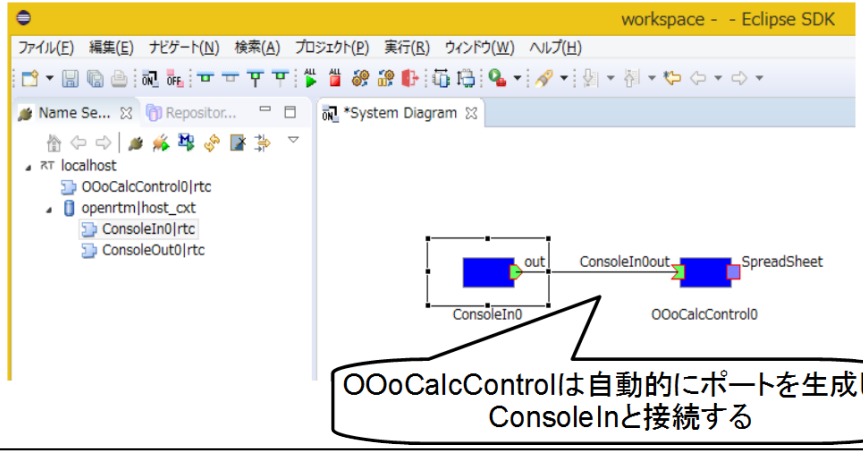
OutPortの接続

- ツリーからConsoleInのoutを選択



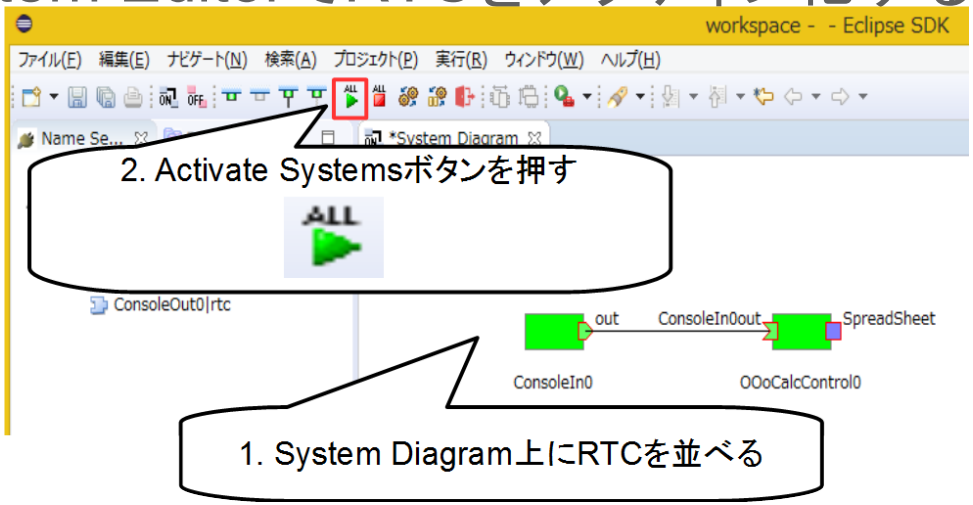
- 「作成」ボタンをクリックする



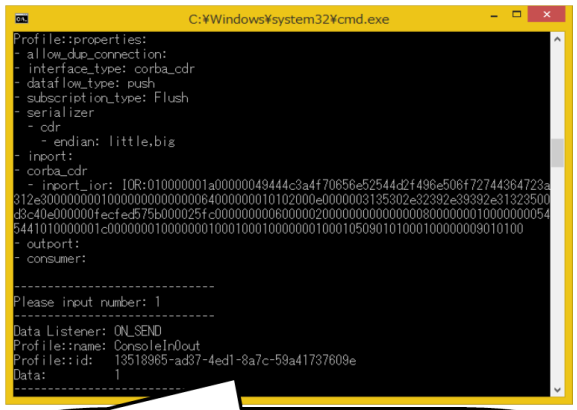


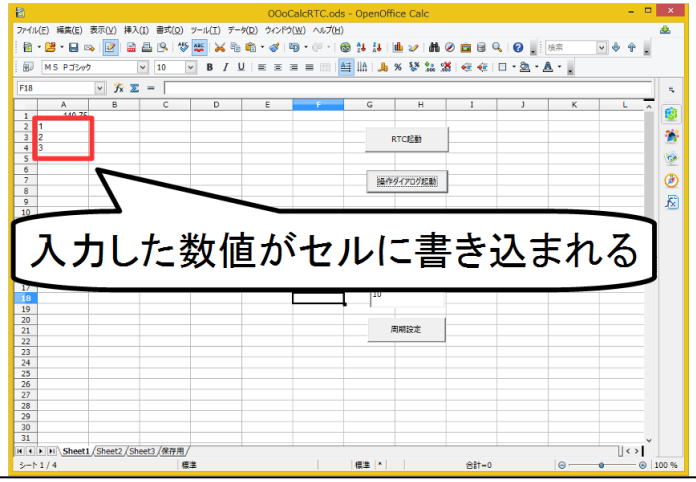
OutPortの動作確認

- RT System EditorでRTCをアクティブ化する



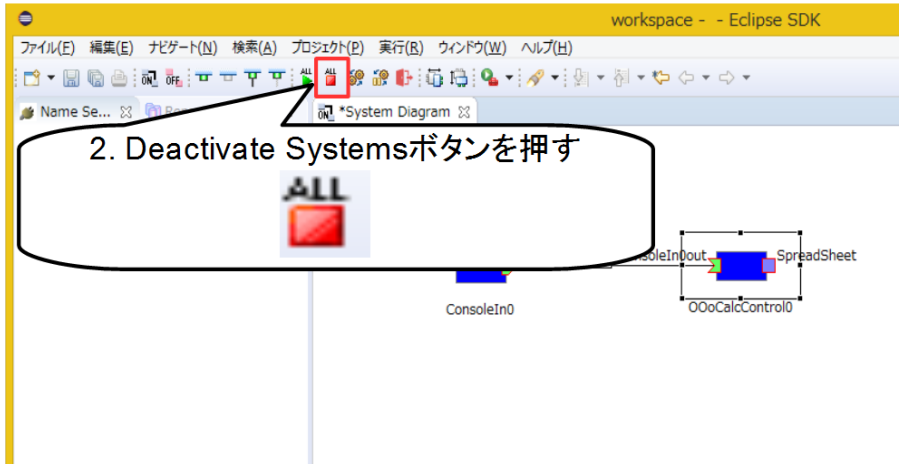
- ConsoleInのウィンドウに数値を入力する





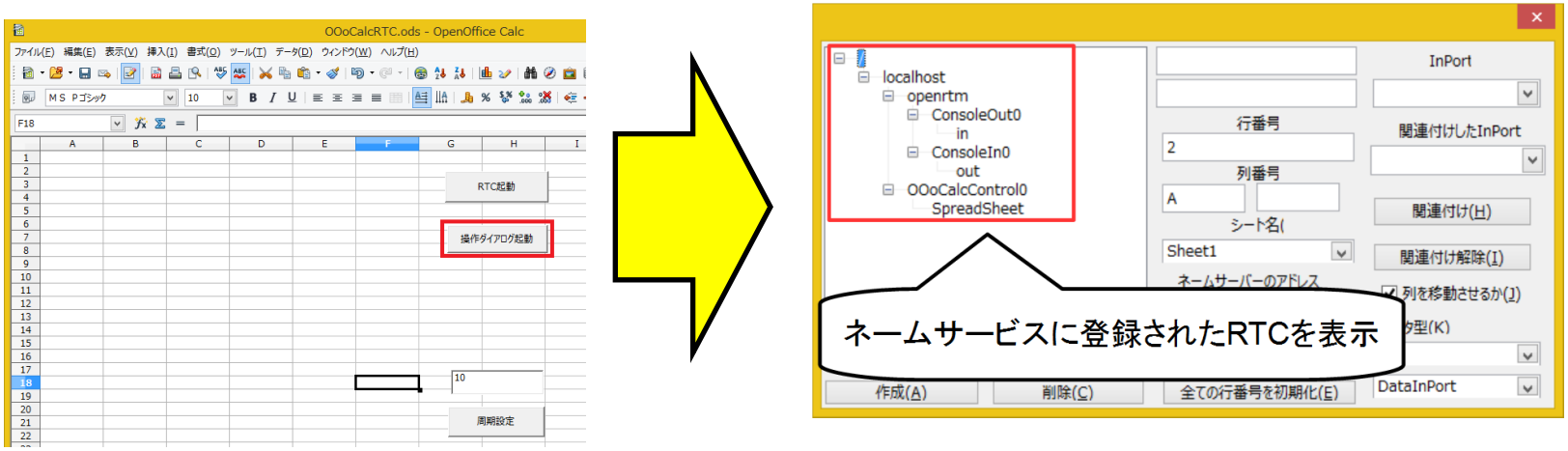
OutPortの動作確認

- 一旦、RTCを非アクティブ化する

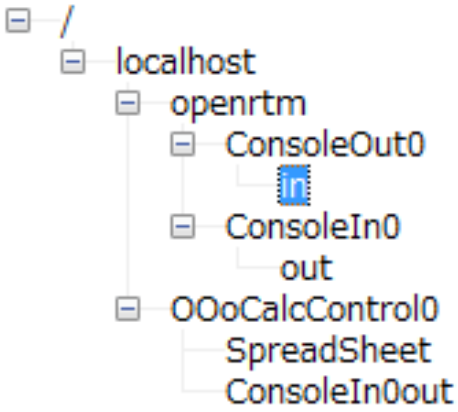


InPortの接続

- LibreOffice Calcの「操作ダイアログ起動」ボタンをクリック後、「ツリー表示」ボタンをクリック

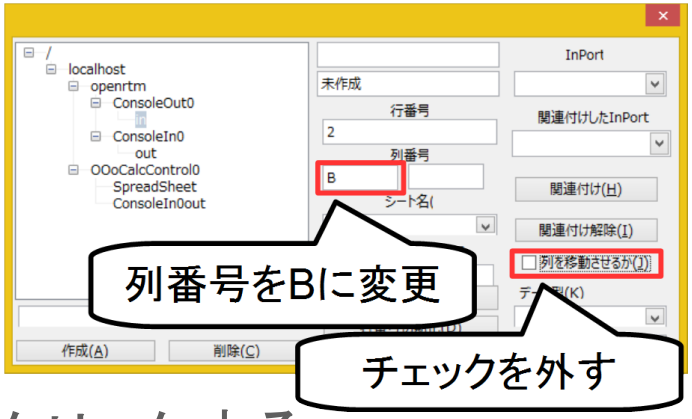


- ツリーからConsoleOutのinを選択

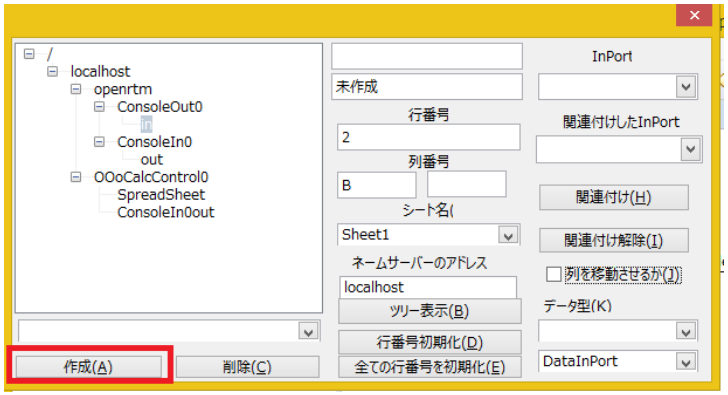


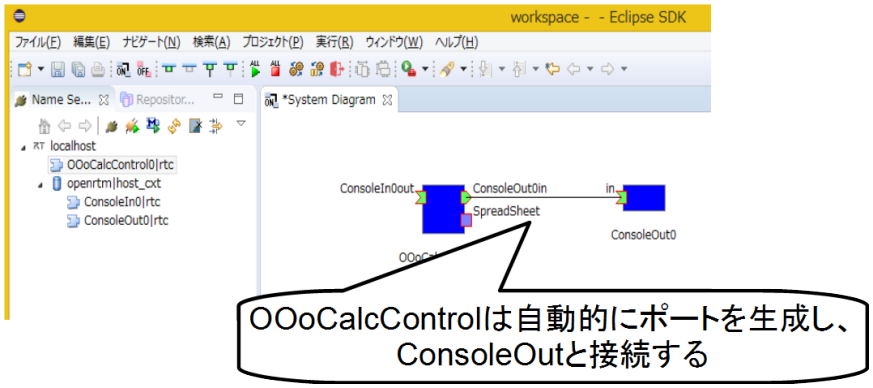
InPortの接続

- 列番号をBに設定
- 「列を移動させるか」のチェックを外す
  - ※2回クリックしないとチェックが外れない



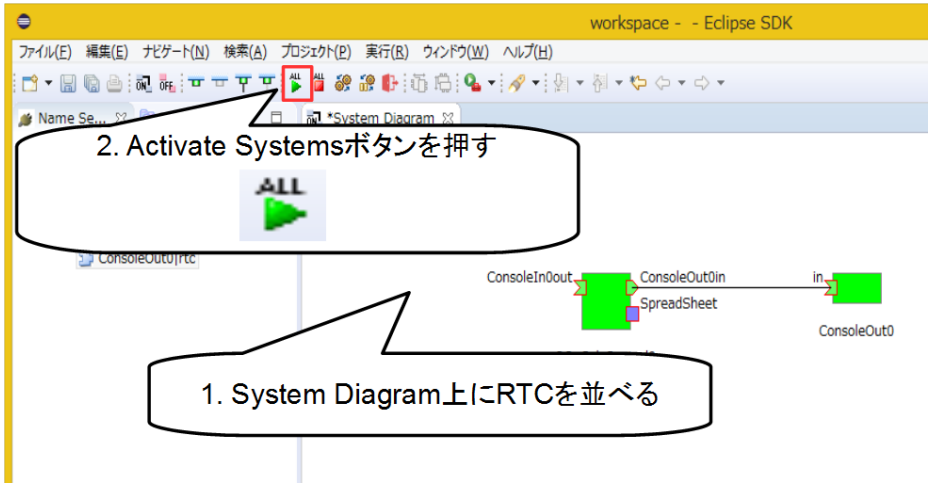
- 「作成」ボタンをクリックする



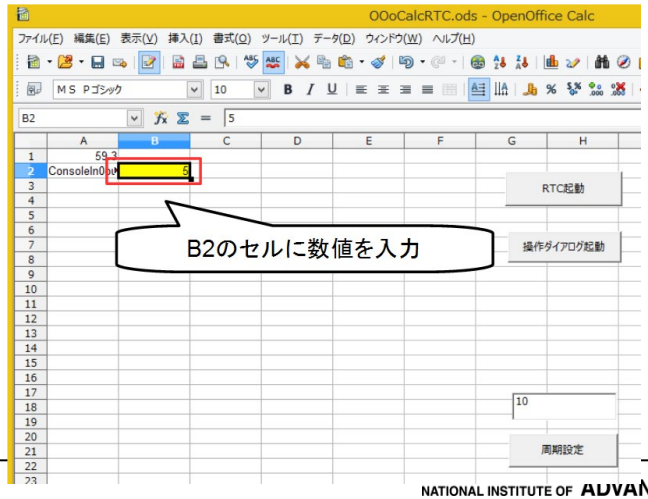


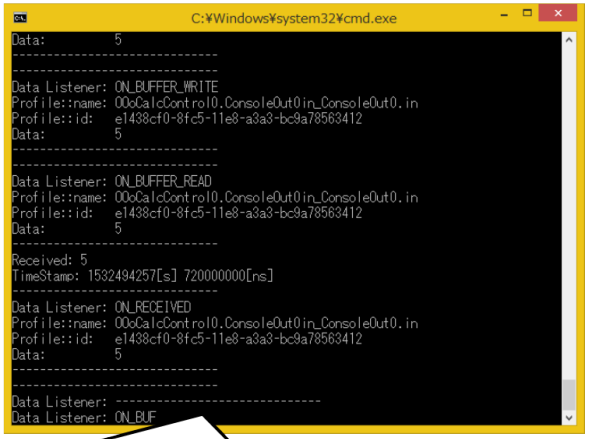
InPortの動作確認

- RT System EditorでRTCをアクティブ化する



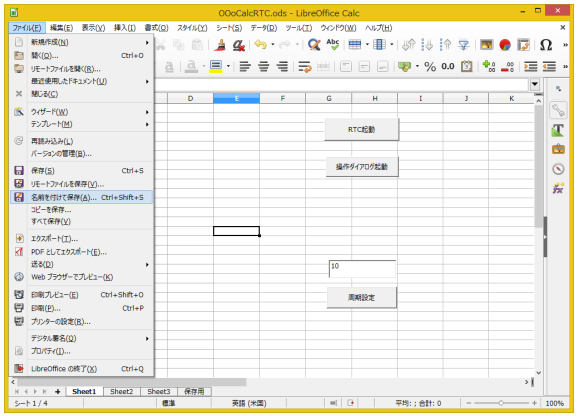
- B2のセルに数値を入力する



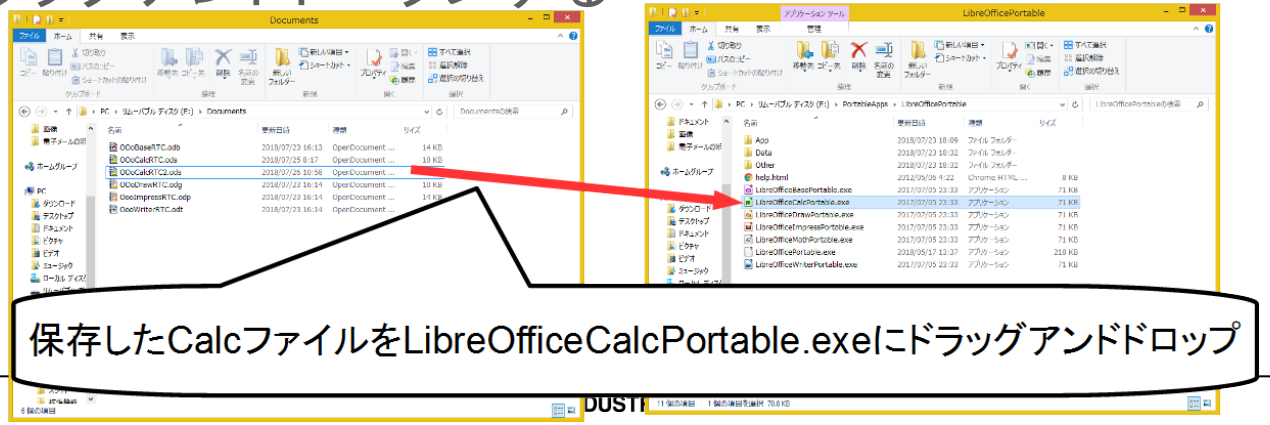


保存

- Calcファイル(.ods)を名前を付けて保存する
  - 接続したポートの情報などはCalcファイル(.ods)に保存される

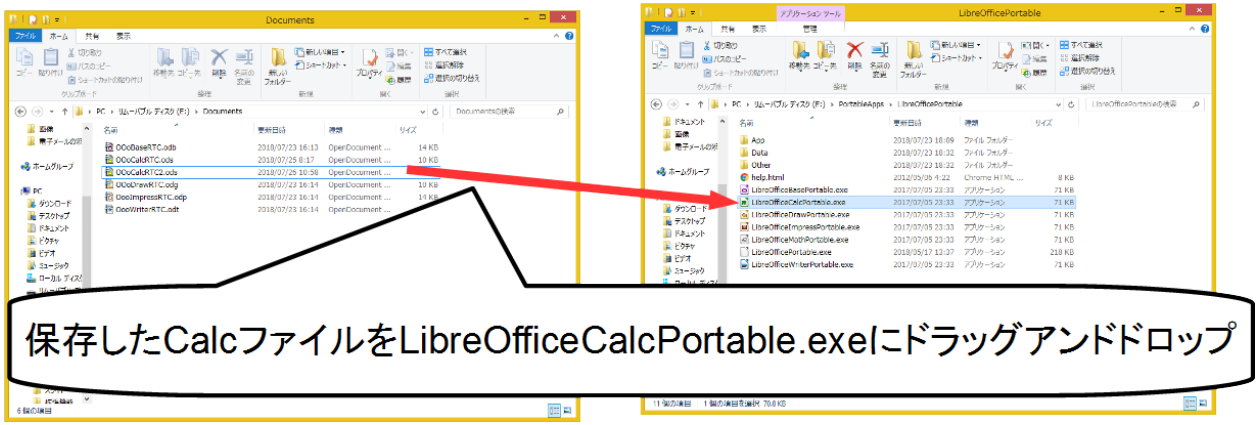


- 保存したCalcファイルをUSBメモリ内の PortableApps¥LibreOfficePortable¥**LibreOfficeCalcPortable.exe** にドラッグアンドドロップする

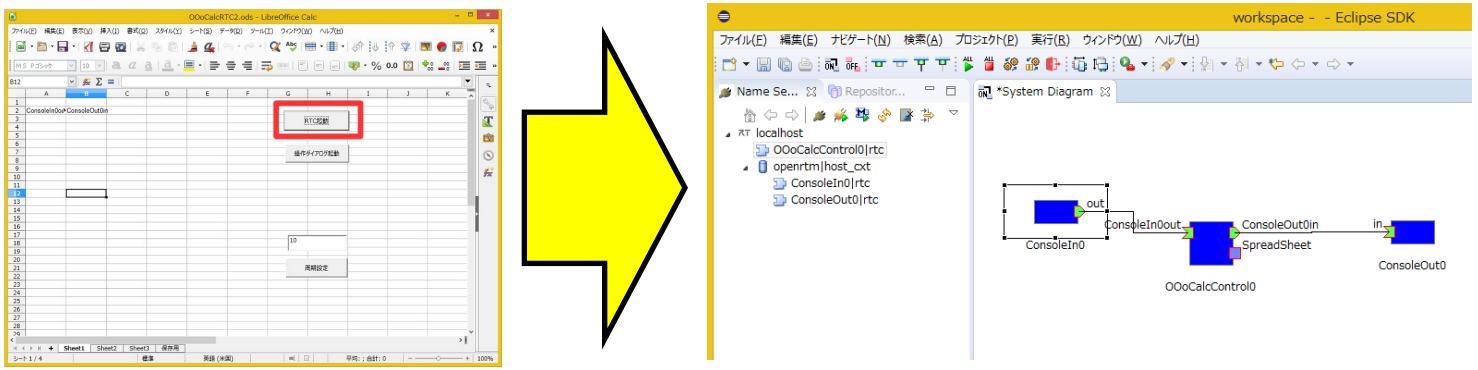


復元

- 保存したCalcファイルをUSBメモリ内の PortableApps¥LibreOfficePortable¥**LibreOfficeCalcPortable.exe** にドラッグアンドドロップする



- 「RTC起動」ボタンを押すと、ポートに自動接続して状態を復元する



まとめ

RTCの動作確認、テストで有用なツールを紹介した

- **RtStorage**
  - 出力データの保存と再生が簡単
  - 一度保存したデータを再利用できるため、例えばセンサRTCのデータを保存しておいて再生することで、InPort側のRTCのテストだけを実行できる(テストでセンサRTCを実行する必要がない)
  - rtshellでも同じことはできるので、好きな方を使ってください

- **表計算ソフトとRTCを連携させるためのツール**
  - 指定のデータの入力が簡単
  - 出力データが視覚的に分かりやすい
  - グラフ表示も併用するなど様々な使い方ができる

Lua版RTミドルウェア
(OpenRTM Lua)

OpenRTM Lua

- プログラミング言語Luaに対応したRTミドルウェア
  - RTCを**Lua**、もしくは**MoonScript**、**LuneScript**で開発可能にする

Lua

```lua
ConsoleIn.new = function(manager)
        local obj = {}
        -- RTObjectをメタオブジェクトに設定する
        setmetatable(obj, {__index=openrtm.RTObject.new(manager)})
        -- データ格納変数
        obj._d_out = openrtm.RTCUtil.instantiateDataType("::RTC::TimedLong")
        -- アウトポート生成
        obj._outOut = openrtm.OutPort.new("out",obj._d_out,"::RTC::TimedLong")

        -- 初期化時のコールバック関数
        -- @return リターンコード
        function obj:onInitialize()
                -- ポート追加
                self:addOutPort("out",self._outOut)

                return self._ReturnCode_t.RTC_OK
        end
        -- アクティブ状態の時の実行関数
        -- @param ec_id 実行コンテキストのID
        -- @return リターンコード
        function obj:onExecute(ec_id)
                io.write("Please input number: ")
                local data = tonumber(io.read())
                -- 出力データ格納
                self._d_out.data = data
                -- 出力データにタイムスタンプ設定
                openrtm.OutPort.setTimestamp(self._d_out)
                -- データ書き込み
                self._outOut:write()
                return self._ReturnCode_t.RTC_OK
        end

        return obj
end
```

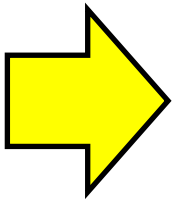
MoonScript

```moonscript
class ConsoleIn extends openrtm_ms.RTObject
        -- コンストラクタ
        -- @param manager マネージャ
        new: (manager) =>
                super manager
                -- データ格納変数
                self._d_out = openrtm_ms.RTCUtil.instantiateDataType("::RTC::TimedLong")
                -- アウトポート生成
                self._outOut = openrtm_ms.OutPort("out",self._d_out,"::RTC::TimedLong")

        -- 初期化時のコールバック関数
        -- @return リターンコード
        onInitialize: =>
                -- ポート追加
                @addOutPort("out",self._outOut)

                return self._ReturnCode_t.RTC_OK


        -- アクティブ状態の時の実行関数
        -- @param ec_id 実行コンテキストのID
        -- @return リターンコード
        onExecute: (ec_id) =>
                io.write("Please input number: ")
                data = tonumber(io.read())
                -- 出力データ格納
                self._d_out.data = data
                -- 出力データにタイムスタンプ設定
                openrtm_ms.setTimestamp(self._d_out)
                -- データ書き込み
                self._outOut\write()
                return self._ReturnCode_t.RTC_OK
```

OpenRTM Lua

- プログラミング言語Luaに対応したRTミドルウェア
  - Luaは軽量なプログラミング言語であり、RTC実行環境一式で**2MB程度**
    Lua(2MB)>>>>C++(8MB)>Python(10MB以上)>>>Java(笑)
  - **スクリプト言語**であるため、Pythonと同様に効率的な開発が可能
  - LuaJITによる**高速な動作**
    - Luaはゲーム開発で主に使用されるプログラミング言語のため、10000体×onExecute関数1000回のキャラクターの当たり判定にかかる時間を計測

言語	結果[s]
Lua	4.3324
LuaJIT	1.2459
C++	0.6142
Python	6.4802



Pythonを圧倒し、C++に匹敵する性能

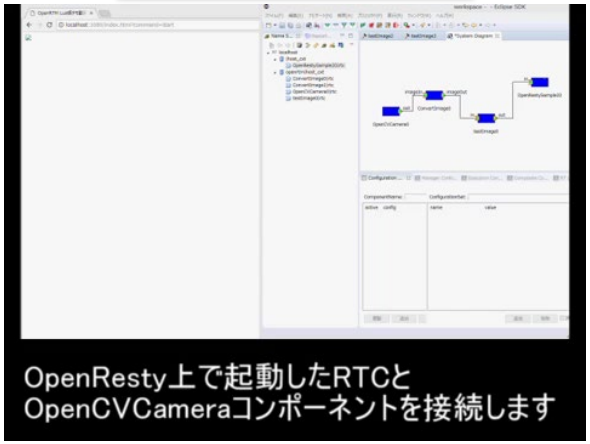
OpenRTM Lua

- プログラミング言語Luaに対応したRTミドルウェア
  - 他のソフトウェアへの組み込み

ゲームエミュレータ

ロボットシミュレータ

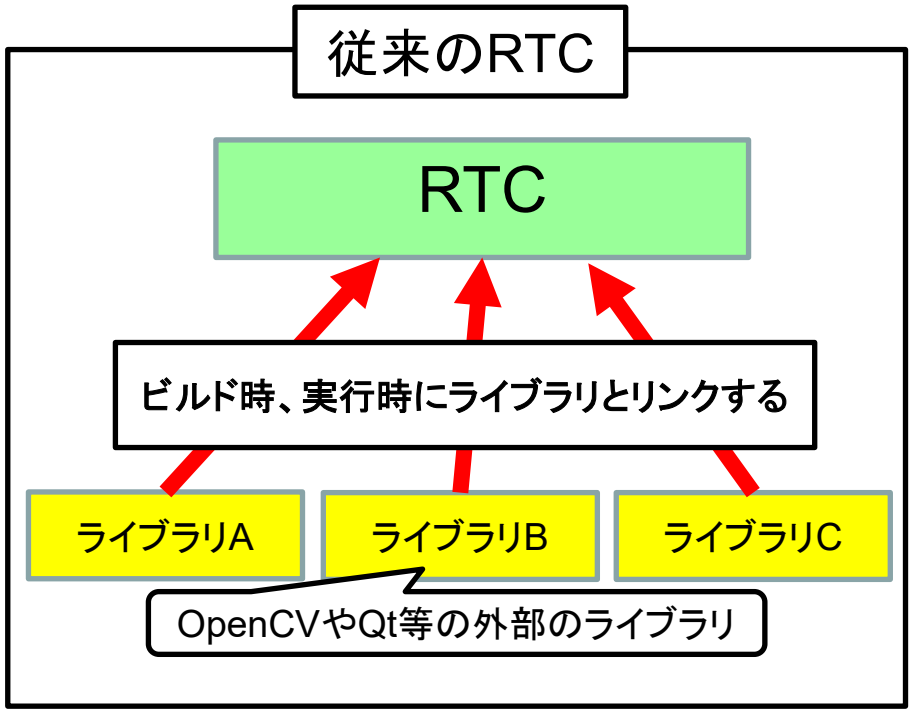
物理シミュレータ

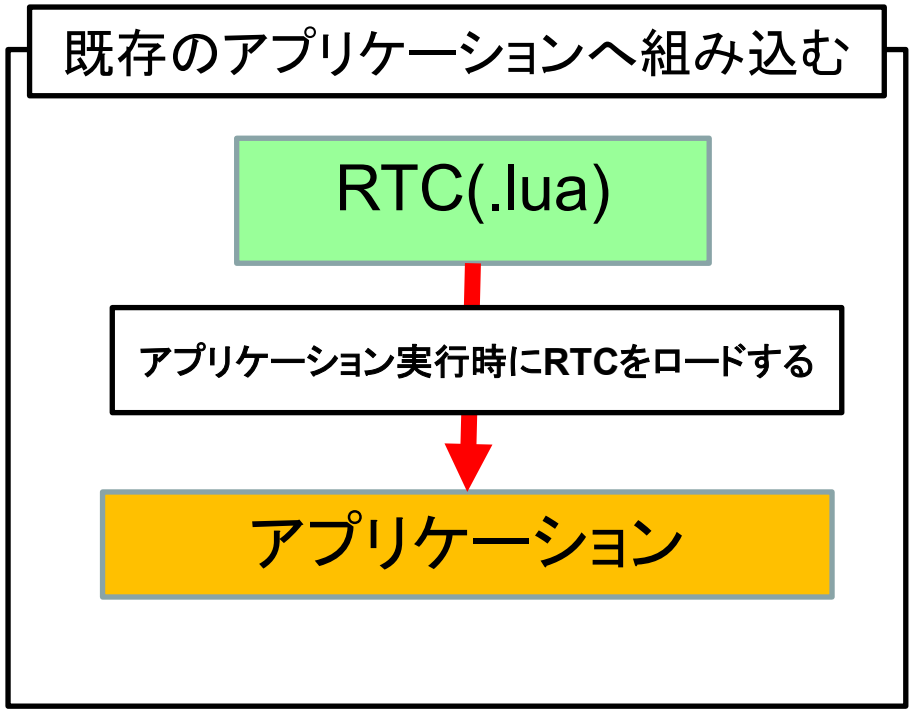


WEBアプリサーバー

既存のアプリケーションへの組み込み

- OpenRTM-Luaは既存のアプリケーションへ組み込むことを得意としている

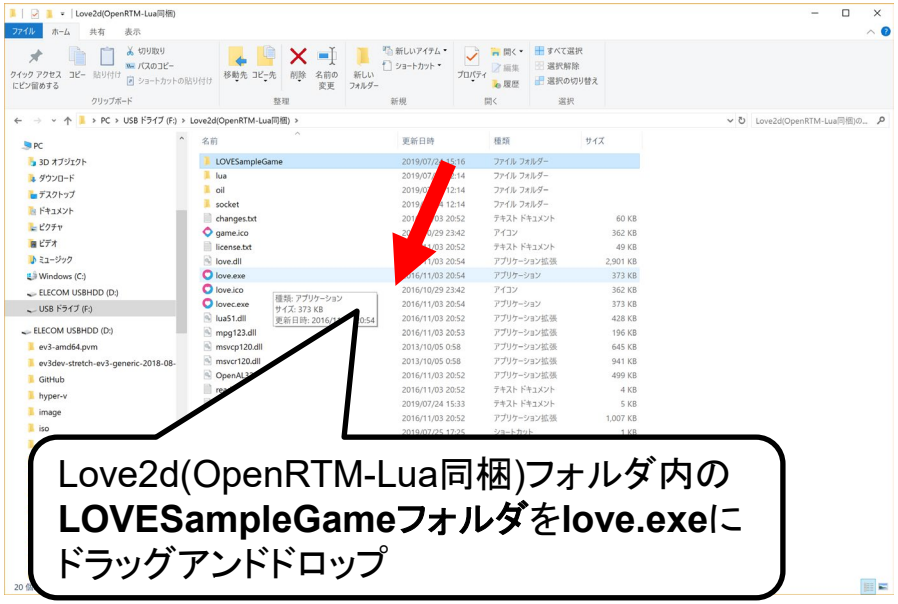


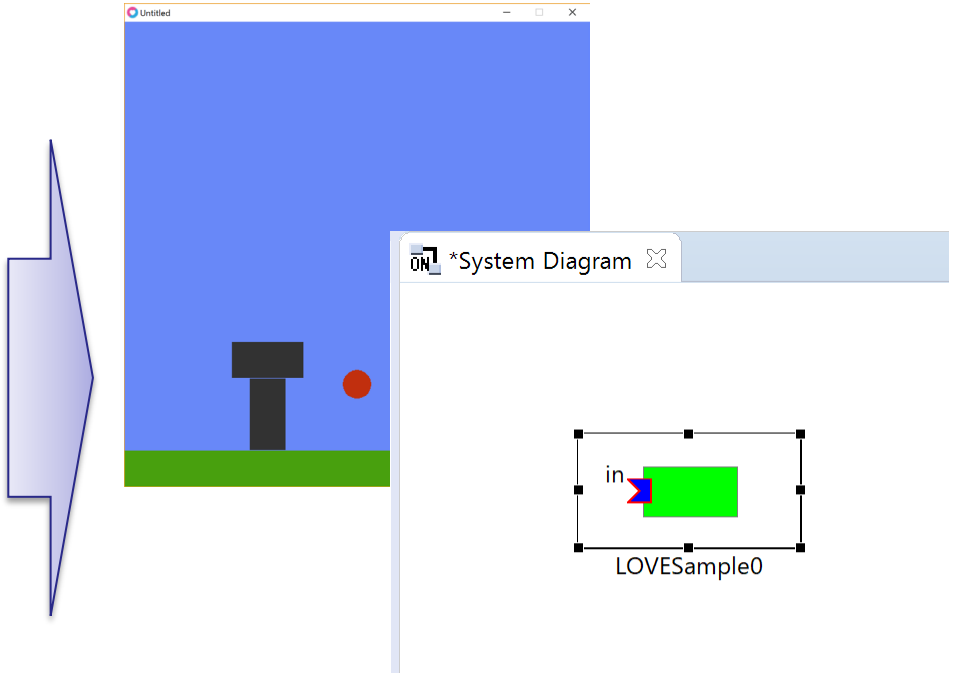


- V-REP等は自分でビルドすると大変だが、Luaにより容易に機能を追加可能になっている

2Dゲームエンジン LÖVEでの応用例

- 配布のUSBメモリに以下のソフトウェアを同梱
  - LÖVE(ゲームエンジン)
  - OpenRTM-Lua
  - LÖVE用RTCのサンプル

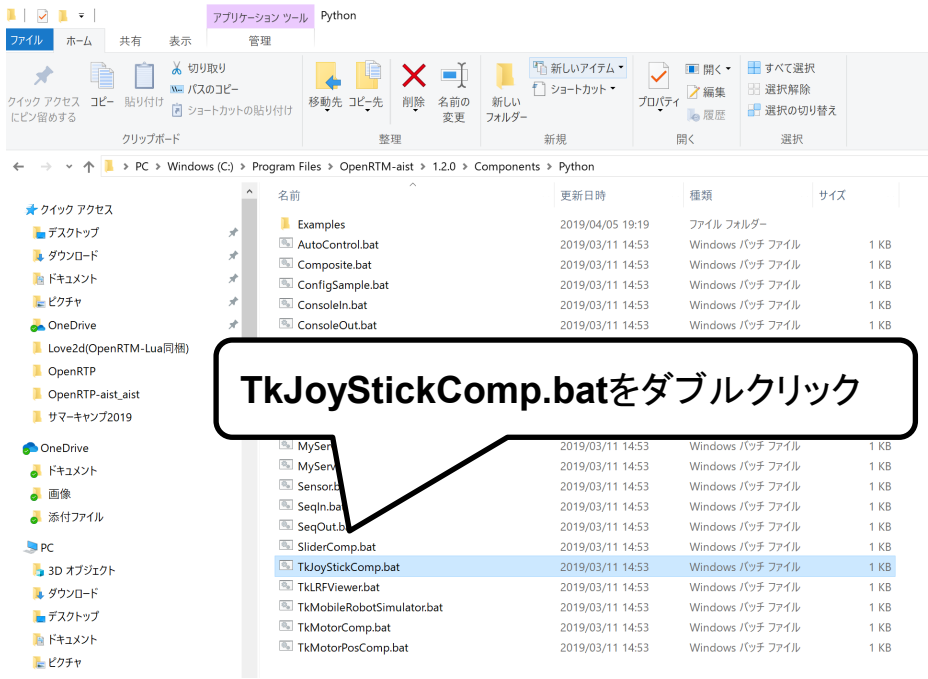




2Dゲームエンジン LÖVEでの応用例

- ジョイスティックコンポーネント(OpenRTM-aist Python版サンプル)を起動して接続

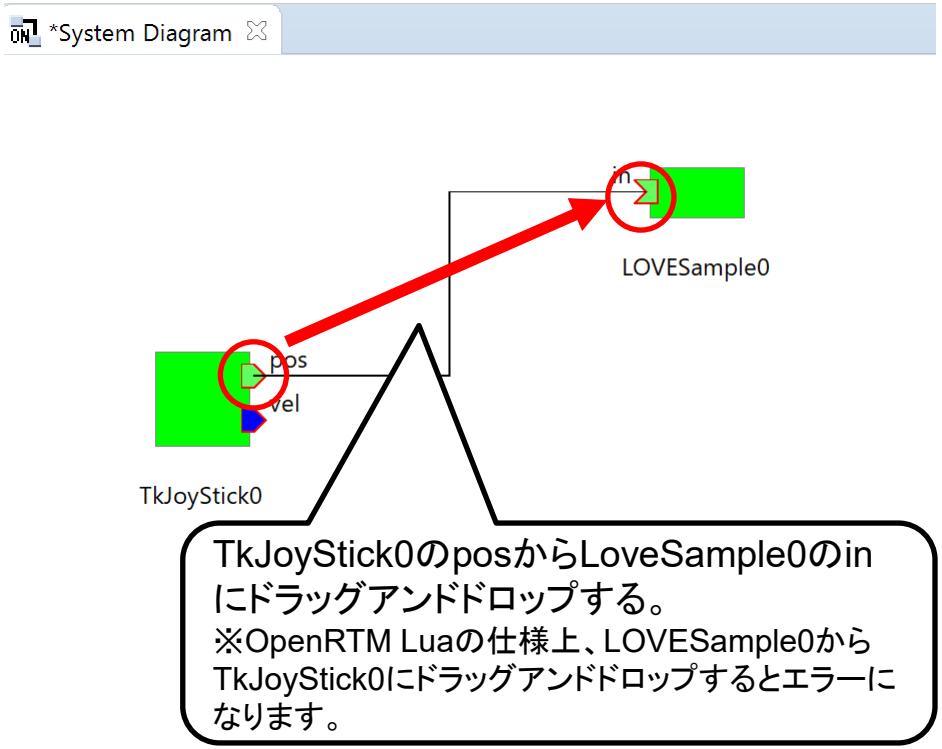
Windows 10の場合、左下の「ここに入力して検索」に**Python_Examples**と入力して検索して実行

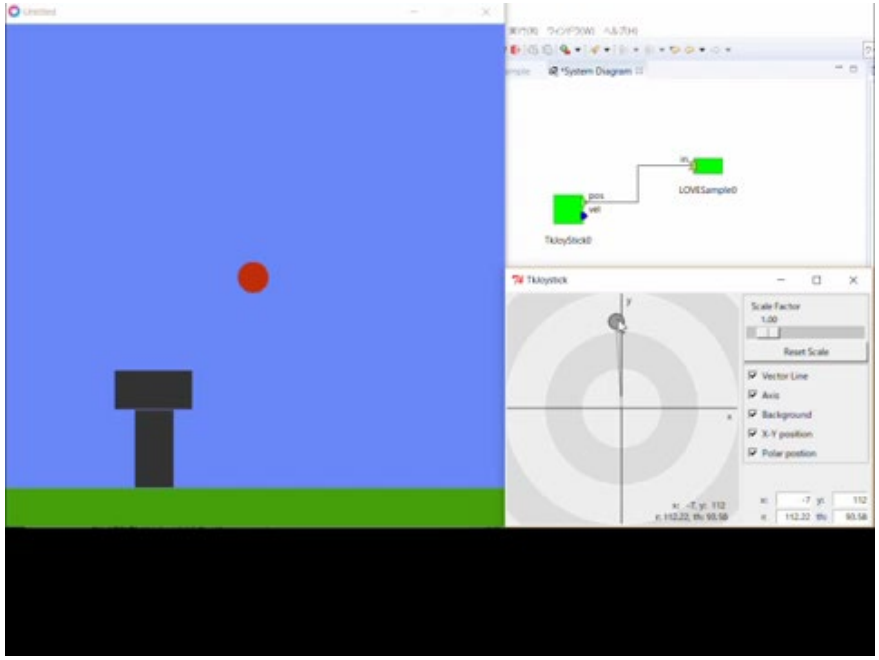


TkJoyStickComp.batをダブルクリック

2Dゲームエンジン LÖVEでの応用例

- ジョイスティックコンポーネント(OpenRTM-aist Python版サンプル)を起動して接続

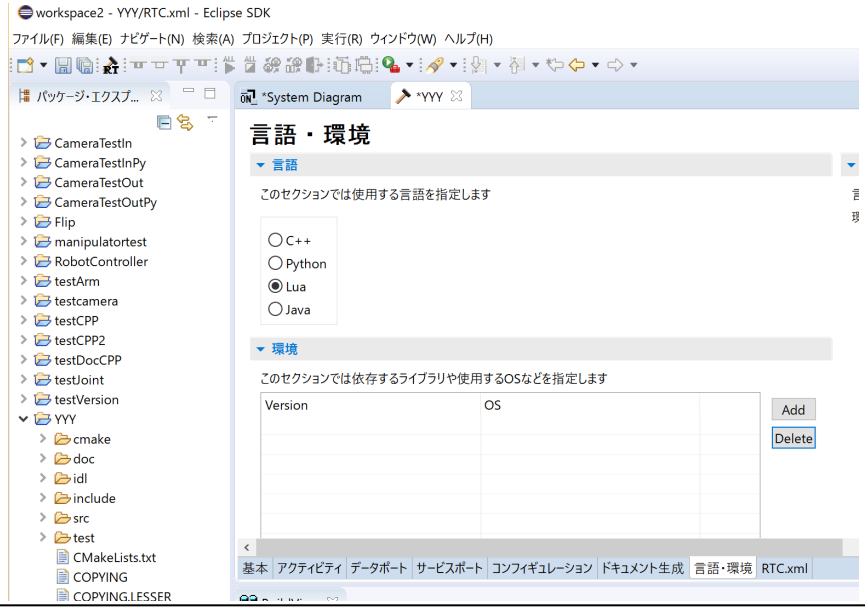




Luaスクリプティング機能をサポートするアプリケーション(今回はゲームエンジン)とOpenRTM-aistの連携が可能

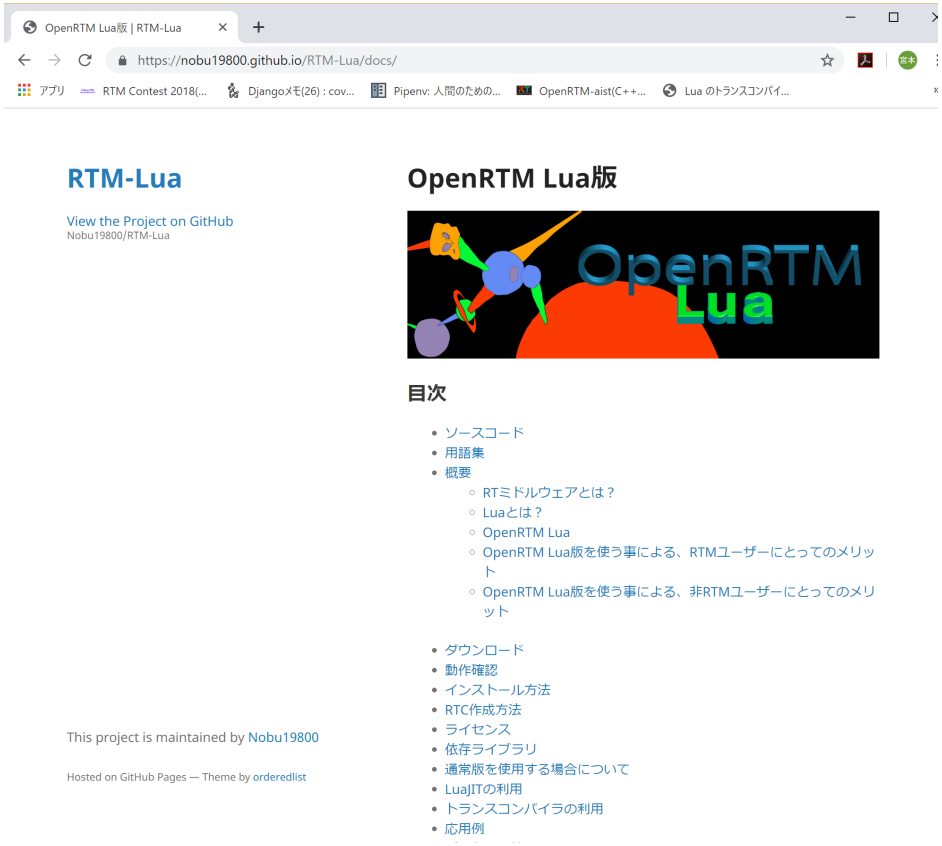
LÖVE用RTCの作成手順

- 以下のファイルが必要
  - LOVESampleGameフォルダ(名前は任意)
    - **main.lua**
      - LÖVEがロードするスクリプトファイル
      - LÖVEがmain.luaの以下の関数を呼び出す
        - » load関数：main.luaのロード時に呼び出し
        - » draw関数：描画更新時に呼び出し
        - » update関数：フレーム更新時に呼び出し
      - 以下のLOVESample.luaをロードしてRTCを起動する
    - **LOVESample.lua** (名前は任意)
      - RTCの実装
      - RTC Builderで生成できる
      - onExecute関数等を実装する



OpenRTM Lua

- 詳細な説明は以下に記載
  - 「OpenRTM Lua」で検索すると出る



https://nobu19800.github.io/RTM-Lua/docs/