[APNOTE03]

アラームデバイスのプッシュボタンでスクリプト起動

ABS-9000 DeviceServer
APNOTE03 Rev A.1.0
2008/10/1

オールブルーシステム (All Blue System)
ウェブページ: www.allbluesystem.com
コンタクト:contact@allbluesystem.com

1　　イントロダクション

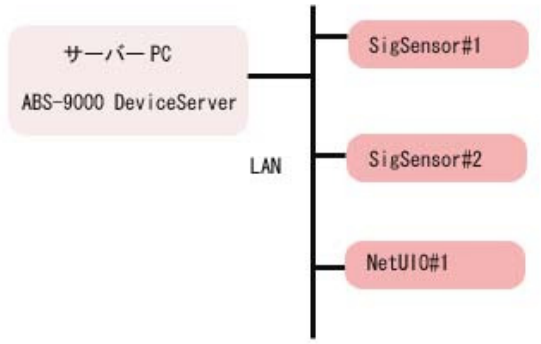
アラームデバイスのプッシュボタンを、決められたパターンで連続で押すことによって、DeviceServer のスクリプトを起動する例について説明します。

このドキュメントで説明するシステムでは、ボタンのパターンと対応して起動するスクリプトは３種類までですが、イベントハンドラを変更することで様々にカスタマイズ可能です。

2　　必要な機材・リソース

必要なシステムやデバイス等	説明
ABS-9000 DeviceServerの動作しているPC	スタンダードライセンスもしくはエンハンスライセンスが必要になります。
SigSensorデバイス NetUIOデバイス	プッシュボタンが実装されたSigSensor または、NetUIOデバイスが最低１台必要になります。アラームデバイスを複数同時に接続しても使用できます。

3　　システム構成図



4　　システム動作概要

- アラームデバイス(SigSensor,NetUIO) 上でユーザーがプッシュボタンを押すと、ALARM_BUTTON_PUSH イベントが DeviceServer で発生します。
- ALARM_BUTTON_PUSHイベントハンドラで、予め設定された順序通りにボタンが押されているかをチェックします。もし、設定された順序に一致した場合はスクリプトを起動します。ボタンを押した順序はアラームデバイス毎に区別され、３種類までのパターンと、対応して起動するスクリプトを登録できます。

5 設定手順

5.1 デバイス設定

SigSensor デバイスまたは、NetUIOデバイスを LAN 上に設置します。

DeviceServer のアラーム管理プログラムで、設置したアラームデバイスをDeviceServer に登録します。

また、アラーム管理プログラムで、アラームデバイス(SigSensorまたは NetUIO) の下記の項目の詳細設定行います。

アラーム管理プログラムで設定するデバイスの詳細機能設定	
設定が必要な項目	**設定内容**
IPアドレス	デバイスのIP アドレスを設定する
IPネットワークマスク	デバイスを設置した LAN の環境に合わせる
デフォルトゲートウェイアドレス	デバイスを設置した LAN の環境に合わせる
イベント送信先ホストアドレス	DeviceServer の動作しているPC の IPアドレスを設定する
送信先ポート番号	27102
ボタンを押したときに、サーバーに CSVIFパケット送信	チェックを付ける

デバイス登録と詳細機能設定の方法については “DeviceServerユーザーマニュアル”を参照してください。また、SigSensor、NetUIO デバイスの詳しい使用方法については、”SigSensor_NetUIOユーザーマニュアル” を参照してください。

5.2 スクリプト・イベントハンドラ設定

5.2.1 ボタンを押したときに実行するスクリプトを作成

ボタンを押したときに起動するスクリプトを３種類作成します。スクリプト名やスクリプト内に記述する内容は自由に決められますが、ここではサンプルとして以下の３つのスクリプトを作成します。(SAMPLE1,SAMPLE2,SAMPLE3)

SAMPLE1スクリプト。ファイル名(SAMPLE1.lua) で DeviceServerのスクリプトフォルダに保管します。

```
File_id = "SAMPLE1"
log_msg("start..",file_id)
for key,val in pairs(g_params) do
    log_msg(string.format("g_params[%s] = %s",key,val),file_id)
end
```

SAMPLE2スクリプト。ファイル名(SAMPLE2.lua) で DeviceServerのスクリプトフォルダに保管します。

```
File_id = "SAMPLE2"
log_msg("start..",file_id)
for key,val in pairs(g_params) do
```

```
    log_msg(string.format("g_params[%s] = %s",key,val),file_id)
end
```

SAMPLE3スクリプト。ファイル名(SAMPLE3.lua) で DeviceServerのスクリプトフォルダに保管します。

```
File_id = "SAMPLE3"
log_msg("start..",file_id)
for key,val in pairs(g_params) do
    log_msg(string.format("g_params[%s] = %s",key,val),file_id)
end
```

5.2.2 ALARM_BUTTON_PUSH イベントハンドラの作成

アラームデバイスのボタンを押したときに起動される、ALARM_BUTTON_PUSH イベントハンドラを下記の様に記述します。ファイル名(ALARM_BUTTON_PUSH.lua) で DeviceServerのスクリプトフォルダに保管します。

イベントハンドラ中の、ボタンシーケンスパターンと起動スクリプト名の設定部分は、適宜環境に合わせて修正してください。

```
    start_seq_code** = xxxxxxx
    script_name** = xxxxxxx
```

```
file_id = "ALARM_BUTTON_PUSH"
-----------------------------------------------------------------
-- 同時に押されたボタンリストを作成する
-----------------------------------------------------------------
push_list = {}
for key,val in pairs(g_params) do
  if string.match(key,"^SW_") and (val == "ON") then
     table.insert(push_list,key)
   end
end
log_msg(g_params["DeviceName"] .. " " .. table.concat(push_list,","),file_id)

-----------------------------------------------------------------
-- キーの押された順番を記録して、start_seq_code#n に一致した場合は
-- スクリプト script_name#n を実行する。スクリプトパラメータに
-- DeviceName が渡されて、ボタンを押したデバイス名が渡される。
-----------------------------------------------------------------
start_seq_code1 = "ABCABCAA"
```

```
start_seq_code2 = "AABB"
start_seq_code3 = "AACC"

script_name1 = "sample1"
script_name2 = "sample2"
script_name3 = "sample3"

--------------------------------------------------------------------------
-- デバイス毎に現在のキーシーケンスを記録するための shared_data の key 値
--------------------------------------------------------------------------
key_name1 = "KEY_SEQ_STORE1" .. ":" .. g_params["DeviceName"]
key_name2 = "KEY_SEQ_STORE2" .. ":" .. g_params["DeviceName"]
key_name3 = "KEY_SEQ_STORE3" .. ":" .. g_params["DeviceName"]

--------------------------------------------------------------------------
-- デバイス毎に最後にボタンを押した時刻を記録するための shared_data の key 値
--------------------------------------------------------------------------
key_timestamp = "KEY_TIME_STORE" .. ":" .. g_params["DeviceName"]

--------------------------------------------------------------------------
-- 3 秒以上間隔が空いてキーが押された場合は過去のキーシーケンスクリア
--------------------------------------------------------------------------
stat,prev_timestamp = get_shared_data(key_timestamp)
if not stat then error() end
local current_time = os.time(os.date "*t")
if (prev_timestamp ~= "") then
   local prev_time = tonumber(prev_timestamp)
   if (os.difftime(current_time,prev_time) > 2) then
     if not set_shared_data(key_name1,"") then error() end
     if not set_shared_data(key_name2,"") then error() end
     if not set_shared_data(key_name3,"") then error() end
   end
end
if not set_shared_data(key_timestamp,tostring(current_time)) then error() end

--------------------------------------------------------------------------
-- キーが２つ以上同時に押された時は、キーシーケンスクリア
--------------------------------------------------------------------------
if (#push_list > 1) then
```

```
  set_shared_data(key_name1,"")
  set_shared_data(key_name2,"")
  set_shared_data(key_name3,"")
end


--------------------------------------------------------------------------
-- キーが一つだけ押された時にキーシーケンスを調べる
--------------------------------------------------------------------------
if (#push_list == 1) then
  -------------------------
  -- start_seq_code1 check
  -------------------------
  stat,val = get_shared_data(key_name1)
  if (stat) then
    local new_seq1 = val .. string.gsub(push_list[1],"^SW_(%w+)","%1")
    if (string.match(start_seq_code1,"^" .. new_seq1)) then
      if (start_seq_code1 == new_seq1) then
        if not script_exec(script_name1,"DeviceName",g_params["DeviceName"]) then error() end
        if not set_shared_data(key_name1,"") then error() end
      else
        if not set_shared_data(key_name1,new_seq1) then error() end
      end
    else
      set_shared_data(key_name1,"")
    end
  end
  -------------------------
  -- start_seq_code2 check
  -------------------------
  stat,val = get_shared_data(key_name2)
  if (stat) then
    local new_seq2 = val .. string.gsub(push_list[1],"^SW_(%w+)","%1")
    if (string.match(start_seq_code2,"^" .. new_seq2)) then
      if (start_seq_code2 == new_seq2) then
        if not script_exec(script_name2,"DeviceName",g_params["DeviceName"]) then error() end
        if not set_shared_data(key_name2,"") then error() end
      else
        if not set_shared_data(key_name2,new_seq2) then error() end
      end
```

```
    else
      set_shared_data(key_name2,"")
    end
  end
  -------------------------
  -- start_seq_code3 check
  -------------------------
  stat,val = get_shared_data(key_name3)
  if (stat) then
    local new_seq3 = val .. string.gsub(push_list[1],"^SW_(%w+)","%1")
    if (string.match(start_seq_code3,"^" .. new_seq3)) then
      if (start_seq_code3 == new_seq3) then
        if not script_exec(script_name3,"DeviceName",g_params["DeviceName"]) then error() end
        if not set_shared_data(key_name3,"") then error() end
      else
        if not set_shared_data(key_name3,new_seq3) then error() end
      end
    else
      set_shared_data(key_name3,"")
    end
  end
end
```

⚠ 注意

スクリプト中に日本語を記述するときは、スクリプトファイルを UTF-8N 形式で保存してください。Shift_JISや UTF-8 BOM付き形式などで保存すると、DeviceServer でエラーが発生します。Windows付属のワードパッドやメモ帳ではこの形式で保存できませんので、別途 UTF-8N 形式で保存可能なエディタソフト（*1）を使用してください。

（*1）TeraPad 等のソフトウエアがよく使用されています。

6 備考

DeviceServer の動作するPC の処理性能が低い場合や、ネットワーク速度が低下している場合は、イベントハンドラの処理が、連続したボタンシーケンスに追従できない場合があります。この場合は、連続してボタンを押す間隔を少し空けてください。

7 このドキュメントについて

7.1 著作権および登録商標

7.2 連絡先

オールブルーシステム (All Blue System)

ウェブページ http://www.allbluesystem.com

メール contact@allbluesystem.com

7.3 このドキュメントの使用について

このドキュメントは、ABS-9000 DeviceServer の一般的な使用方法と応用例について解説してあります。お客様の個別の問題について、このドキュメントに記載された内容を実際のシステムに利用するときには、ここに記載されている以外にも考慮する事柄がありますので、ご注意ください。特に安全性やセキュリティ、長期間にわたる運用を想定してシステムを構築する必要があります。

オールブルーシステムでは ABS-9000 DeviceServer の使用や、このドキュメントに記載された内容を使用することによって、お客様及び第三者に損害を与えないことを保証しません。 ABS-9000 DeviceServer を使用したシステムを構築するときは、お客様の責任の下で、システムの構築と運用が行われるものとします。