iNELS Design Manager - tutorial

[26] iNELS BUS and eLAN-IR

ELKO EP, s.r.o.

revised: September 2019

Contents

1 About integration of eLAN-IR to iNELS BUS

The aim of this tutorial is to introduce the way to interconnect iNELS BUS and eLAN-IR-003 using Connection Server, resp. Python script. This becomes handy especially when home automation should be controlling various appliances working with infrared remote controls. Tutorial shows an extract of the code which can be used as an example – it uses one basic command triggering eLAN-IR-003 stored command. Connection logic is straightforward – system bit connected to some physical button initiates the action – run the script.

If user omits making a connection between physical controller and system bit, then system bit can be directly used as virtual button in application. The only programming requirement would be defining and exporting of system bits.

2 Requirements

- CU3-0xM with the most up-to-dated firmware (02.9A or newer)
- eLAN-IR-003 with the most up-to-date firmware
- Connection Server with most up-to-date firmware (3.343 or newer)
- any suitable software tool for writing of script - click on highlighted links to download tool, e.g. **Visual Studio**, **Code Blocks** or **Notepad++**
- any appliance with infrared remote control, whose transmitting frequency is compatible with eLAN-IR-003
- knowledge of following tutorials:
  - ASCII communication - checking of events
  - making of export file and Connection Server configuration basics

  It is also expected that user already knows how to make a connection and assign particular action and function.
- basic knowledge of Python syntax and programming required **in case of more complex programming**
- It is also expected that user already knows how to make a connection and assign particular action and function
- basic knowledge of Linux terminal

3 Programming procedure

Example of programming procedure shows how to use some buttons to run the script stored in Connection Server. In this case, WSB3-20 will run script to turn TV on and off (using buttons *Up* and *Down*). The run of script is initiated by changing of system bit state.

3.1 eLAN-IR-003

- get to eLAN-IR-003 web interface and go through learning process according to eLAN-IR-003 manual **here**
- go to IR code section and let pop-up window with command properties - click on button **Export**:

Figure 1: eLAN-IR command settings

- then you can see URL link of eLAN-IR-003 stored command included some details about frequency:

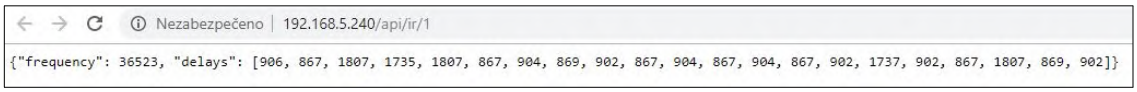

Figure 2: eLAN-IR command settings

- copy URL link for later in order to make script

3.2 CU3-0xM side

- firstly, open iDM and make sure that you are connected, i.e. FAST RUN state shows up
  - make also sure that you can be working with units and devices connected to CU3
- secondly, go to System manager and move to System bits. This example uses two system bits – first one for turning on, second one for turning off. Let us name it – e.g. bit_TV_ON and bit_TV_OFF
- thirdly, if you are not working template project and you are missing function called Digital impulse ON (of various length and without delay), then define such function. Example uses Digital Impulse ON (1 sec.long). Longer impulse means that user cannot send commands by pushing the button so fast (system waits for system bit state change from 0 to 1 again).
- fourthly, make a connecting from WSB3-20, button *Up*, leading to first system bit called TURN-ON. Follow this notation:
  - connection: **WSB3-20 Up** ⇔ system bit **TURN-ON** ;
    connection name: **ACTIVATE SYSTEM BIT – TURN ON**
    - Short press – "Digital Impulse 1 sec."
  - connection: **WSB3-20 Down** ⇔ system bit **TURN-OFF** ;
    connection name: **ACTIVATE SYSTEM BIT – TURN OFF**
    - Short press – "Digital Impulse 1 sec."

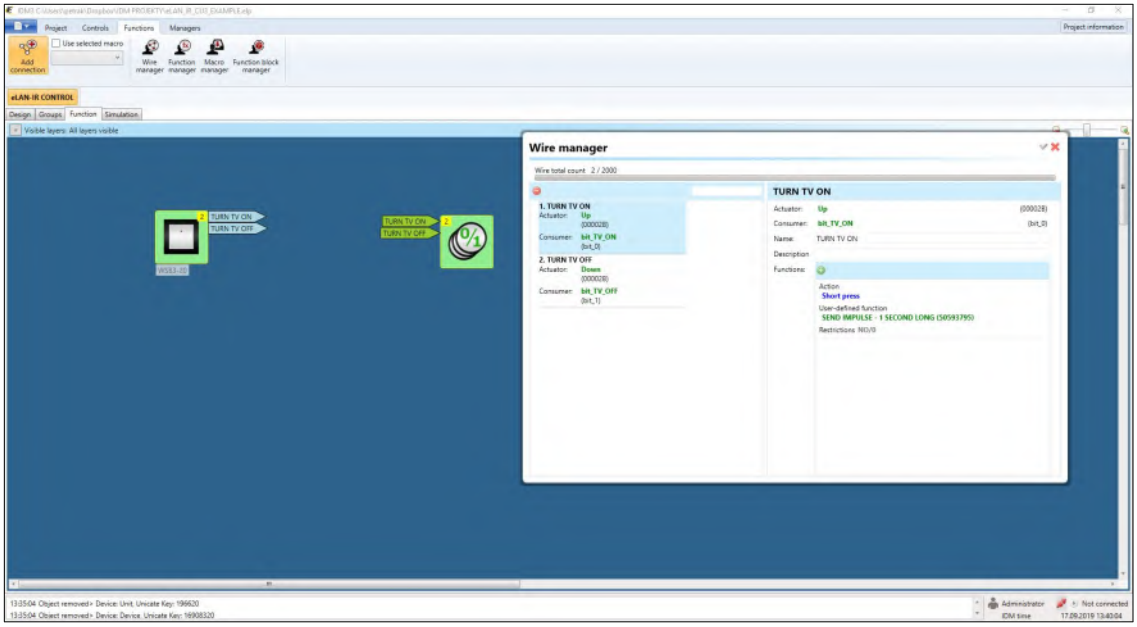

Figure 3: iDM - Wire overview

3.3 Scripting part

- example uses Notepad++ capable of showing syntactic features (code aligning, highlighting of keywords, etc.)
- firstly, open IDE (Notepad++) and create new Python script (it should have suffix .py)
- secondly, add libraries needed to call system functions, i.e.

  import os
  import sys

- thirdly, add a variable, to which you store function call URL link from eLAN-IR-003 web-interface. Function is named as *curl* and it requires additional argument *–XPUT*, whose function is send a request to get eLAN-IR-003 stored particular command. Then URL link follows – it stores information about saved IR command in eLAN. Here is the whole line:

  cmd = "curl -XPUT http://192.168.5.240/api/ir/1"

- fourthly, last line of script must contain a function, which uses previously defined variable as function argument. Function system (from library os) executes the content of argument – curl –XPUT...

  os.system(cmd)

- fifthly, the body of first script (TURN-ON.py - turn TV on) should look like this:

  import sys
  import os

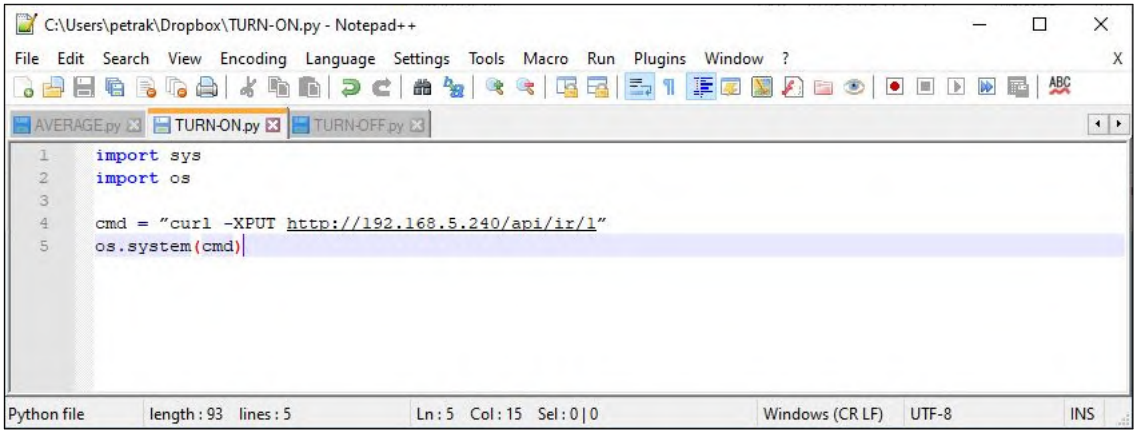

Figure 4: Notepad++ - Python script (turn TV ON)

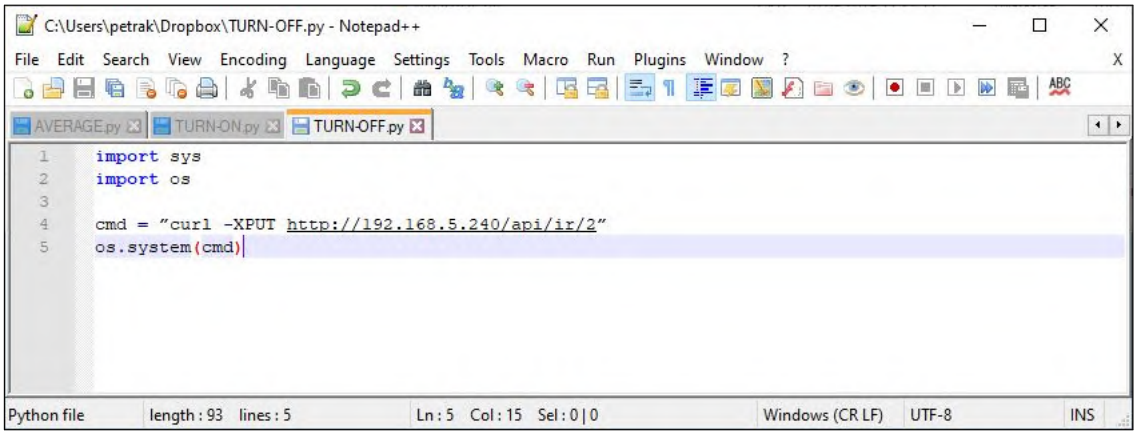

Figure 5: Notepad++ - Python script (turn TV OFF)

cmd = "curl -XPUT http://192.168.5.240/api/ir/1"
os.system(cmd)

- sixthly, the body of second script (TURN-OFF.py - turn TV off) should look like this:

  import sys
  import os

  cmd = "curl -XPUT http://192.168.5.240/api/ir/2"
  os.system(cmd)

- if you have managed to complete all this, then please script and transfer the file to Connection Server. For instance, application **WinSCP** allows to make connection between NTFS and EXT3 (or 4). Remember the location of script.

- if you need to make sure that script is really working, then open SSH connection between your computer and Connection Server (e.g. via **PuTTY** in order to try out the script. If your Connection Server contains Python library and you are already logged in via Terminal (SSH), then call the script using command python. More precise example:

python TURN-ON.py
or
python TURN-OFF.py

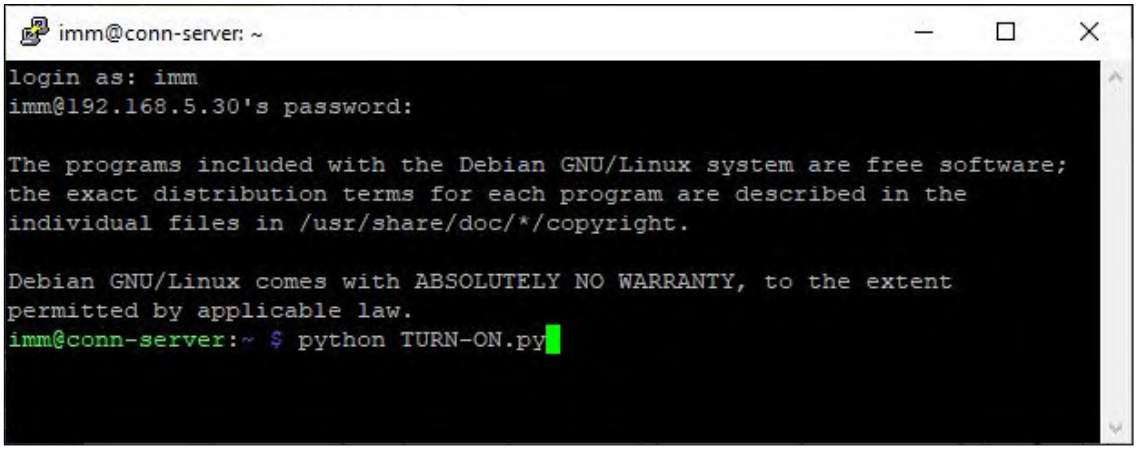

Figure 6: PuTTY - testing in Terminal via SSH

- if IR lens of eLAN-IR-003 points toward to IR sensor of appliance and script does not contain typo, then running of script results in successful action

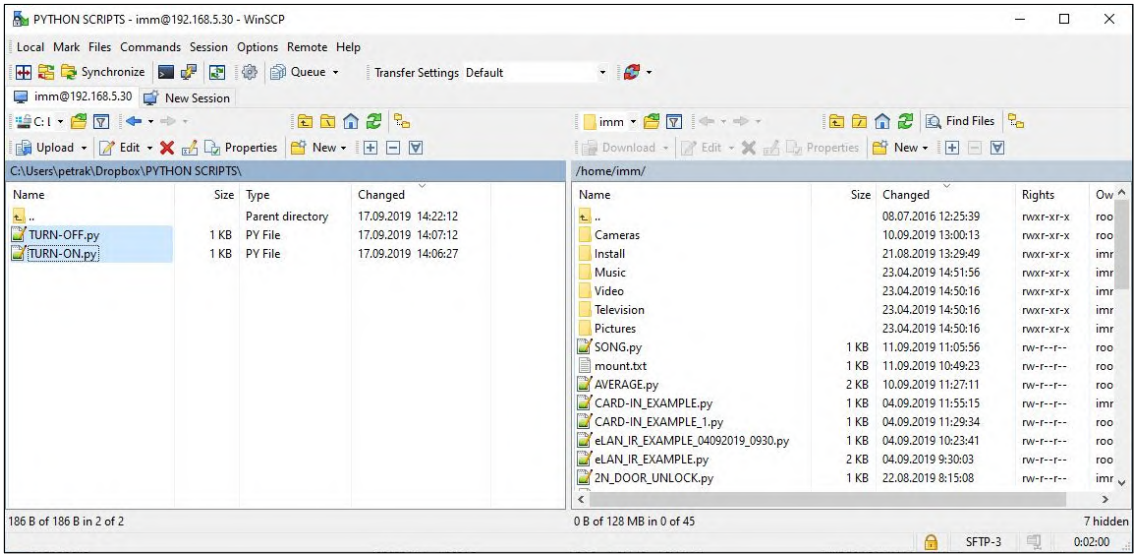

Figure 7: WinSCP - Copying of Python scripts to Connection Server

3.4 Connection Server part

- assuming that you have a working Connection Server with uploaded export file in it, go to tab *Eventscript*
- firstly, fill in all required fields, i.e.

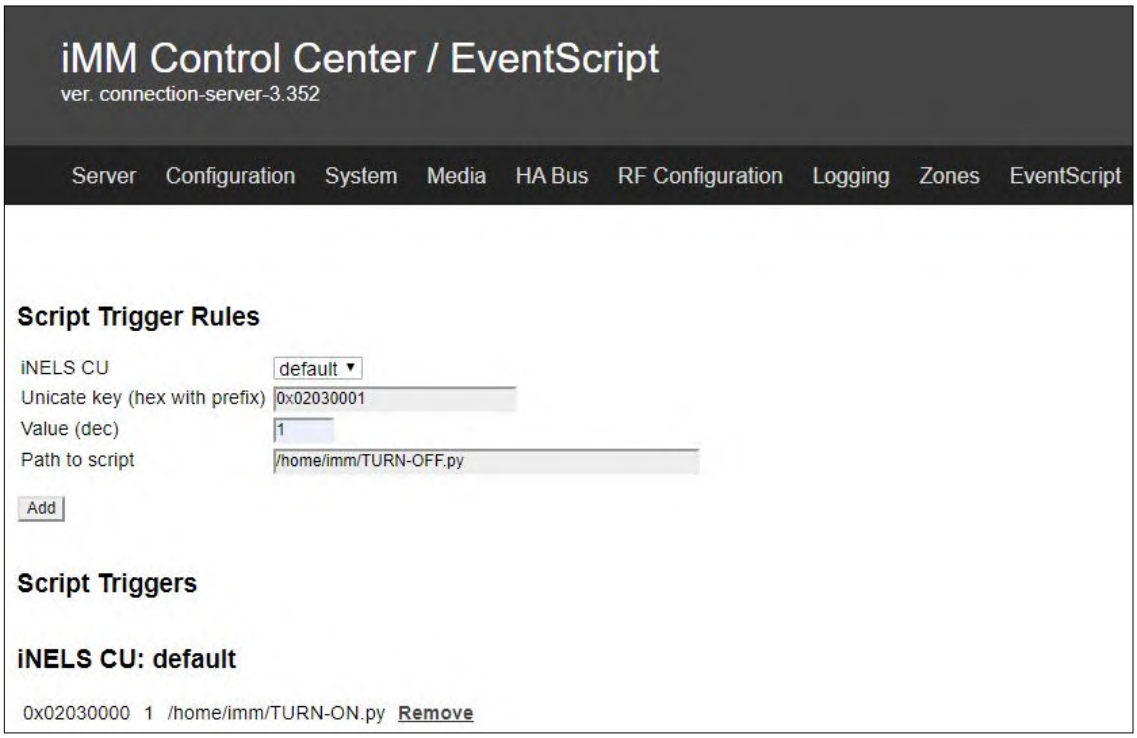

Figure 8: Connection Server web-interface - EventScript tab settings

  - *iNELS CU* – leave default or select particular one
  - *Unicate key (hex with prefix)* – add hexadecimal address of system bit from export file
  - *Value (dec)* – both script will take "1" as argument (i.e. system bit turned ON)
  - *Path to script* – provide the link to script location, e.g. */home/imm/TURN-ON.py*

- secondly – optional: to run script from mobile/smartphone application you can add a corresponding icon to some room. Choose icon type Scene and provide the link of script location with system bit. You can also choose icon type on/off and assign particular system bit.

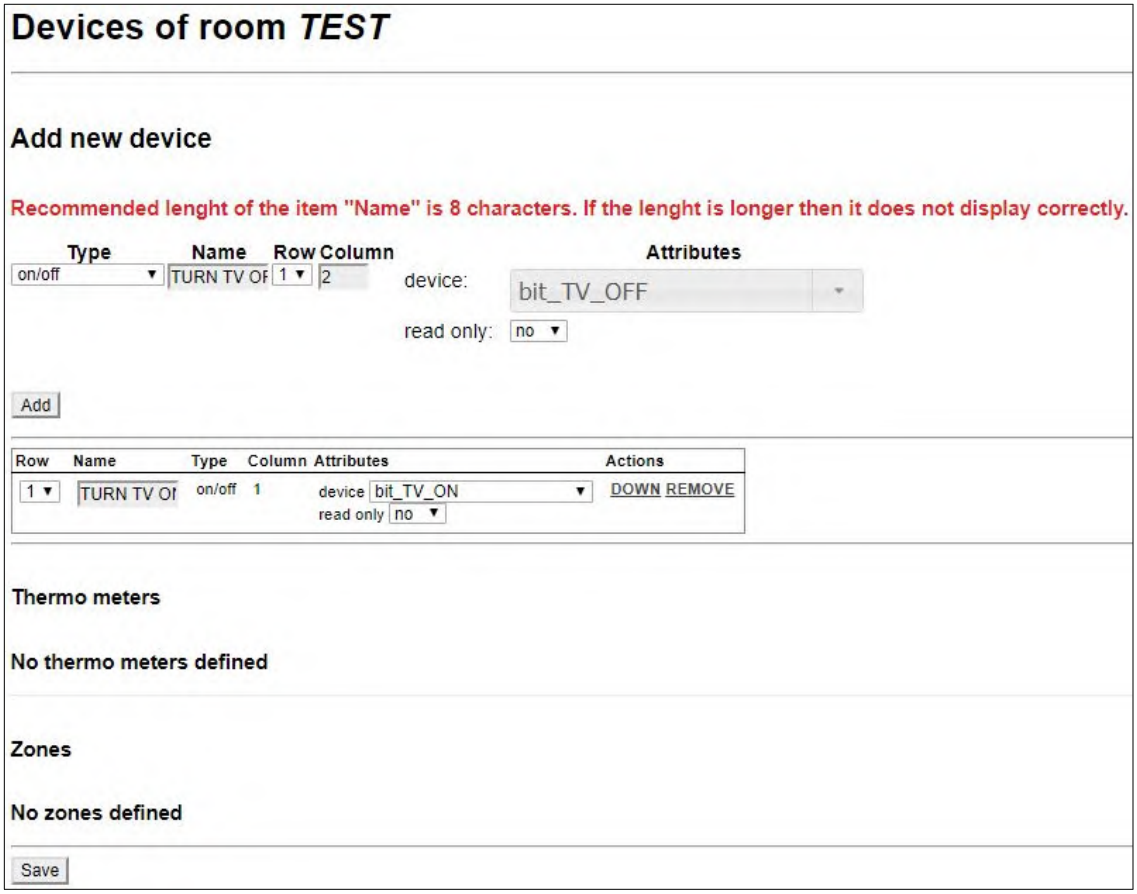

Figure 9: iDM - event overview

4 Troubleshooting

- if scripted remote control does not work properly, watch checked events on CU3-0xM carefully. Make sure that *DIGITAL_OUT_SwitchON* and *DIGITAL_OUT_SwitchOff* are checked.

- in some cases, if calling the script by pushing button is still not working, it is recommended to check running daemons in Connection Server (*IP-ADDRESS-CONNECTION-SERVER:9001* ... default login *imm / imm123*).

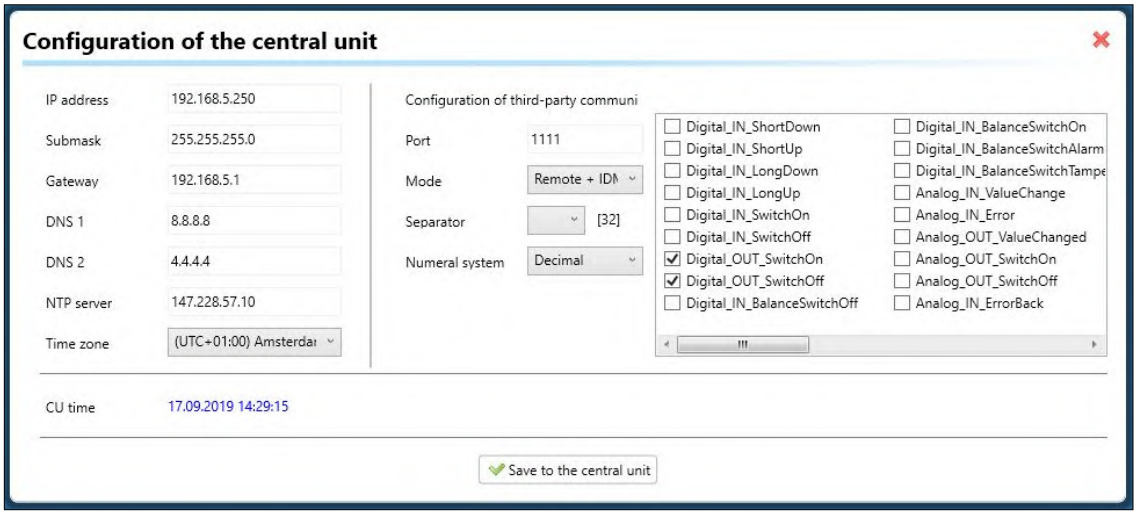

Figure 10: Connection Server - room settings