Lua runner examples

- Configuration Example With Default and Saved Values (Modbus TCP, IEC104)
- Signal Delay with Lua
- Mathematical Operations With Lua (FW version 1.7)
- Cronjob example

Configuration Example With Default and Saved Values (Modbus TCP, IEC104)

Excel configuration

In order to control signals with Lua script an excel configuration for WCC Lite is needed. In this case three devices are required: one for Modbus TCP Master, one for IEC104-slave and one for Lua runner. Configuration example:

name	description	device_alias	enable	protocol	execution_type	error_limit	ip	id	mode	scan_rate_ms	retry_count	timeout_ms	bind_address	host	port	asdu_size	cot_size	ioa_size	nvt	swt	t1	t2	t3	time_sync	message_size	cache_size
Modbus TCP device	Modbus TCP signals	Modbus_TCP	1	Modbus TCP				1	tcp	300	3	10000	0.0.0.0		502											
IEC104 SCADA system	IEC104 SCADA signals	IEC104_SCADA	1	IEC 60870-5-104 slave									0.0.0.0		2404	2	2	3	8	12	15	10	20	1	249	100
LUA device	LUA signals	LUA	1	Lua runner	signal	0																				

In order to connect slave and master devices, master's IP address and slave's host addresses has to be specified. Modbus IP address will be the address of Wi-Fi to which the computer is connected (this can be checked on terminal window with command *ipconfig*) and host address for IEC104 slave protocol will be the IP address to which WCC Lite is connected. To reach the device via these addressed WCC Lite has to be connected to the internet.

When creating Excel configuration with Lua, there is an option **default value** for signal. This value will be set to the signal right after uploading configuration or if the script does not return any saved values.

Signals sheet:

signal_name	device_alias	signal_alias	source_device_alias	source_signal_alias	execute	default_value	threshold_units	absolute_threshold	enable	min_value	max_value	gi	log	number_type	job_todo	tag_job_todo	common_address	info_address	data_type
Result_modbus	Modbus_TCP	result	LUA	result			real	2	1				1	SIGNED16	3;0;1	3;0;1			
command LUA	LUA	command	IEC104_SCADA	Command	1	20							1						
result LUA	LUA	result											1						
Command IEC104	IEC104_SCADA	Command	LUA	command					1	2	3	1	1				1	1	50

As it is seen from example values such as **min_value** and **max_value** can be added to determine limits of a signal. This way command signal will only return results which are within this range. Otherwise command value will have negative cot with invalid, non topical or overflow attributes and new value will not be sent to result signals. As configured, until command value is sent default value will be represented for Lua command signal. For saved values to be represented a Lua script is needed.

Lua script

Lua script example for this configuration is shown below:

```
local saved = get(signals.result) --getting result signal which is equated to new variable 'saved'
local command = get(signals.command) --getting command signal which is equated to new variable 'command'

--get() function returns nill if there is no valid value

if not command then --if command is not nill
    if saved then --if signal is not nill
        publish(signals.result, saved.value) --this value is published to result signals and saved value
    end
    return 0
end


local time_diff = time_ms() - tonumber(command.time) --compares command time and real time
local is_command = time_diff < 30000 and time_diff > -30000 --if command time differs from
--real time more than 30s it will not be executed
```
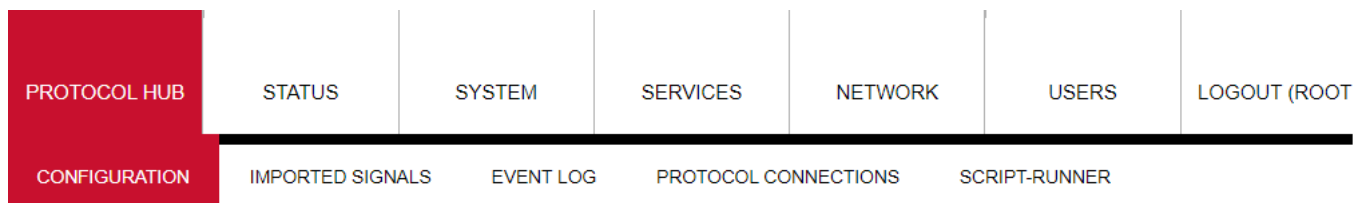
```
if string.find(command.attributes, "nt") or string.find(command.attributes,
    "iv") or string.find(command.attributes, "ov") then
  --searching if signal has negative attributes
    if is_command then --if command execution time is not exceeding the limits then
        command.attributes = "cot=7,cotn" --equates negative cot values to response signal attributes
        publish(signals.command, command) -- and publishes value to command signals and value
        if saved then --if there is saved value then
            publish(signals.result, saved.value) --restores saved value to result signals
        end
        return 0
    end
else
    if is_command then
        command.attributes = "cot=7"
        publish(signals.command, command)   --in this cycle command value is being returned as well as
    --cot7 and cot10 values in case given signal is command and has no negative attributes
        command.attributes = "cot=10"
        publish(signals.command, command) --publishes response to the command
        save(signals.result, command.value) --command value is being saved to result signal
    end
    publish(signals.result, command.value) --in this row command value is being published to result signals
end
```

Uploading configuration and Lua script to WCC Lite

First Excel configuration needs to be uploaded to WCC Lite:

PROTOCOL HUB	STATUS	SYSTEM	SERVICES	NETWORK	USERS	LOGOUT (ROOT
CONFIGURATION	IMPORTED SIGNALS	EVENT LOG	PROTOCOL CONNECTIONS	SCRIPT-RUNNER		

Protocol configuration
IMPORT PROTOCOL CONFIGURATION

Here you can import Excel configuration file. Up to 1000 signals are allowed. All previous signals will be replaced.

Configuration file: [Choose File] No file chosen   [Import configuration]

PLC (IEC-61499) Boot file: [Choose File] No file chosen   [Import FBOOT file]

IEC61850 Server model file: [Choose File] No file chosen   [Import server model file]

After uploading configuration default value will be shown:

IMPORTED SIGNALS

Device	Signal	Value	Units	State	Attributes	Time
Modbus TCP device	Result_modbus					2023-08-22 16:53:40.45
IEC104 SCADA system	Command IEC104					
LUA device	command LUA	20				
LUA device	result LUA					

After uploading configuration no errors should appear and all signal should be represented on the web. To upload Lua script go to *Script-Runner*, select *upload script* and then *start:*

Connecting master and slave via Vinci software

Connecting IEC104 slave

In order to connect to WCC Lite via IEC104 protocol, select Master(Client) mode on Vinci:

Check Settings tab to match excel configuration:

Specify IP address which should match the one in Excel configuration.

Connecting ModbusTCP master:

To connect ModbusTCP Master, select Slave (Server) mode on Vinci:

Check address to match **id** in Excel configuration:

Match the IP address given in Excel configuration as well.

Executing commands

Start both master and slave simulations on Vinci. Check if both protocols are connected to WCC Lite on the web tab *Protocol Connections:*

Device	Protocol	Host	Status	Timestamp
Modbus_TCP	Modbus TCP master	192.168.67.176	Connected	2023-08-22 14:25:37
IEC104_SCADA	IEC 60870-5-104 slave	192.168.1.216	Connected	2023-08-22 14:25:11

To execute commands, open Vinci program with IEC104 master running. Here, go to *System* tab and fill in required fields such as IOA and select data type indicated in Excel configuration. First, try sending value that is outside the set range:

After selecting *execute* this value will not be showed on the web and positive cot6 and negative cot7 values will be seen on Vinci IEC104 simulation window. Positive cot6 indicates command activation and negative cot7 means that command activation confirmation was denied. The command signal value will not be represented as result signals, because it is determined in the script, that signals with negative attributes will not be published.

IMPORTED SIGNALS

Device	Signal	Value	Units	State	Attributes	Time
Modbus TCP device	Result_modbus					2023-08-25 12:32:03.43
IEC104 SCADA system	Command IEC104	2		cotn	cot=7	2023-08-25 13:04:30.72
LUA device	command LUA	2		cmd,iv	asdu=1,cot=6,ioa=1,org=1 ,qi=0,type=float	2023-08-25 13:04:30.72
LUA device	result LUA					

Now specify value which will be sent as a result and is within the given range.

	PROTOCOL HUB	STATUS	SYSTEM	SERVICES	NETWORK	USERS	LOGOUT (ROOT)	**WCC LITE**
	CONFIGURATION	IMPORTED SIGNALS	EVENT LOG	PROTOCOL CONNECTIONS	SCRIPT-RUNNER			

IMPORTED SIGNALS

Device	Signal	Value	Units	State	Attributes	Time
Modbus TCP device	Result_modbus	2				2023-08-07 12:18:52.66
IEC104 SCADA system	Command IEC104	2			cot=10	2023-08-07 12:17:57.42
LUA device	command LUA	2		cmd	asdu=1,cot=6,ioa=1,org=1 ,qi=0,type=float	2023-08-07 12:17:57.42
LUA device	result LUA	2				2023-08-07 12:18:52.66

Positive cot7 and cot10 values will be seen on the Vinci IEC104 simulation window:

Settings	Console	Statistic								
Time	Source	Message	TI	Cause	ASDU	IOA	Value	Status		RawData
09:18:16:942	192.168.73.216:2404	RSN:3 SSN:40	C_SE_NC_1 (50)	Pos. ActCon (7) (T=0 O=0)	1	1	2	Execute Default		68 12 50 00 06 00 32 01 07 00 01 00 01 00 00 00 00 00 40 00
09:18:16:993	192.168.73.216:2404	RSN:3 SSN:41	C_SE_NC_1 (50)	Pos. ActTerm (10) (T=0 O=0)	1	1	2	Execute Default		68 12 52 00 06 00 32 01 0A 00 01 00 01 00 00 00 00 00 40 00
09:18:17:004	Vinci	RSN:42								68 04 01 00 54 00
09:18:17:935	192.168.73.216:2404	RSN:3 SSN:42	C_SE_NC_1 (50)	Pos. ActCon (7) (T=0 O=0)	1	1	2	Execute Default		68 12 54 00 06 00 32 01 07 00 01 00 01 00 00 00 00 00 40 00
09:18:17:985	192.168.73.216:2404	RSN:3 SSN:43	C_SE_NC_1 (50)	Pos. ActTerm (10) (T=0 O=0)	1	1	2	Execute Default		68 12 56 00 06 00 32 01 0A 00 01 00 01 00 00 00 00 00 40 00
09:18:18:933	192.168.73.216:2404	RSN:3 SSN:44	C_SE_NC_1 (50)	Pos. ActCon (7) (T=0 O=0)	1	1	2	Execute Default		68 12 58 00 06 00 32 01 07 00 01 00 01 00 00 00 00 00 40 00
09:18:18:982	192.168.73.216:2404	RSN:3 SSN:45	C_SE_NC_1 (50)	Pos. ActTerm (10) (T=0 O=0)	1	1	2	Execute Default		68 12 5A 00 06 00 32 01 0A 00 01 00 01 00 00 00 00 00 40 00
09:18:19:942	192.168.73.216:2404	RSN:3 SSN:46	C_SE_NC_1 (50)	Pos. ActCon (7) (T=0 O=0)	1	1	2	Execute Default		68 12 5C 00 06 00 32 01 07 00 01 00 01 00 00 00 00 00 40 00
09:18:19:993	192.168.73.216:2404	RSN:3 SSN:47	C_SE_NC_1 (50)	Pos. ActTerm (10) (T=0 O=0)	1	1	2	Execute Default		68 12 5E 00 06 00 32 01 0A 00 01 00 01 00 00 00 00 00 40 00
09:18:20:936	192.168.73.216:2404	RSN:3 SSN:48	C_SE_NC_1 (50)	Pos. ActCon (7) (T=0 O=0)	1	1	2	Execute Default		68 12 60 00 06 00 32 01 07 00 01 00 01 00 00 00 00 00 40 00
09:18:21:001	192.168.73.216:2404	RSN:3 SSN:49	C_SE_NC_1 (50)	Pos. ActTerm (10) (T=0 O=0)	1	1	2	Execute Default		68 12 62 00 06 00 32 01 0A 00 01 00 01 00 00 00 00 00 40 00

This value will also be represented on Modbus TCP master Vinci simulation window:

12:21:36:324	192.168.67.133:5...	1	3	0	1		09 13 00 00 00 06 01 03 00 00 00 01
12:21:36:344	VINCI	1	3			00 02	09 13 00 00 00 05 01 03 02 00 02
12:21:36:648	192.168.67.133:5...	1	3	0	1		09 14 00 00 00 06 01 03 00 00 00 01
12:21:36:728	VINCI	1	3			00 02	09 14 00 00 00 05 01 03 02 00 02

To show what happens if the value is not within determined range after the correct value has been sent before, try executing command with smaller or larger value specified:

09:22:27:999	192.168.73.216:2404	Test Frame Confirmation								68 04 03 00 00 00
09:22:40:066	Vinci	RSN:62 SSN:3	C_SE_NC_1 (50)	Pos. Act (6) (T=0 O=1)	1	1	1	Execute Default		68 12 06 00 7C 00 32 01 06 01 01 00 01 00 00 00 00 80 3F 00
09:22:41:071	192.168.73.216:2404	RSN:4 SSN:62	C_SE_NC_1 (50)	Neg. ActCon (7) (T=0 O=0)	1	1	2	Execute Default		68 12 7C 00 08 00 32 01 47 00 01 00 01 00 00 00 00 00 40 00
09:22:42:090	192.168.73.216:2404	RSN:4 SSN:63	C_SE_NC_1 (50)	Neg. ActCon (7) (T=0 O=0)	1	1	2	Execute Default		68 12 7E 00 08 00 32 01 47 00 01 00 01 00 00 00 00 00 40 00
09:22:43:071	192.168.73.216:2404	RSN:4 SSN:64	C_SE_NC_1 (50)	Neg. ActCon (7) (T=0 O=0)	1	1	2	Execute Default		68 12 80 00 08 00 32 01 47 00 01 00 01 00 00 00 00 00 40 00
09:22:44:079	192.168.73.216:2404	RSN:4 SSN:65	C_SE_NC_1 (50)	Neg. ActCon (7) (T=0 O=0)	1	1	2	Execute Default		68 12 82 00 08 00 32 01 47 00 01 00 01 00 00 00 00 00 40 00

Again, positive cot6 and negative cot7 values is seen on Vinci window.

	PROTOCOL HUB	STATUS	SYSTEM	SERVICES	NETWORK	USERS	LOGOUT (ROOT)	**WCC LITE**
	CONFIGURATION	IMPORTED SIGNALS	EVENT LOG	PROTOCOL CONNECTIONS	SCRIPT-RUNNER			

IMPORTED SIGNALS

Device	Signal	Value	Units	State	Attributes	Time
Modbus TCP device	Result_modbus	2				2023-08-07 12:23:07.22
IEC104 SCADA system	Command IEC104	2		cotn	cot=7	2023-08-07 12:22:40.21
LUA device	command LUA	2		cmd,iv	asdu=1,cot=6,ioa=1,org=1 ,qi=0,type=float	2023-08-07 12:22:40.21
LUA device	result LUA	2				2023-08-07 12:23:07.22

As seen on WCC Lite web window, command signals have negative attributes and result signals have the same value as before, because it was saved by Lua script. This example shows that Lua runner can be used to save certain values to signal. For example after the restart saved value could be seen on command signal to determine minimum or maximum value, last value or typical value. This solution could be useful for protecting important data even after reboot or connection faults.

Configuration --> Download

Lua script --> Download

Signal Delay with Lua

For delaying the response of command execution, Lua runner could be used as one of the solutions. In this example IEC104 and Modbus TCP are used. IEC104 protocol sends the command to Modbus and Lua signals and the results are represented as two separate signals. To create Excel configuration for WCC Lite in this case device sheet should look like this:

name	device_alias	enable	protocol	execution_type	error_limit	ip	id	mode	scan_rate_ms	retry_count	timeout_ms	bind_address	host	port	asdu_size	cot_size	ioa_size	rwt	swt	t1	t2	t3	time_sync	message_size	cache_size
Modbus TCP device	Modbus_TCP	1	Modbus TCP				1	tcp	300	3	10000	0.0.0.0		502											
IEC104 SCADA system	IEC104_SCADA	1	IEC 60870-5-104 slave									0.0.0.0		2404	2	2	3	8	12	15	10	20	1	249	100
LUA device	LUA	1	Lua runner	signal	0																				

In the fields marked red, for Modbus TCP enter IP address of Wi-Fi connected to computer and for IEC104 enter IP address of WCC Lite. Map the signals as shown below:

signal_name	device_alias	signal_alias	source_device_alias	source_signal_alias	execute	enable	gi	log	number_type	job_todo	tag_job_todo	common_address	info_address	data_type
Result_modbus	Modbus_TCP	result	LUA	result		1		1	SIGNED16	3;0;1	3;0;1			
command LUA	LUA	command	IEC104_SCADA	Command	1			1						
result LUA	LUA	result						1						
Command IEC104	IEC104_SCADA	Command	LUA	command		1	1	1				1	1	50

IEC104 SCADA will send command which will then go to Lua signal. Lua signal will send the response back to IEC104 SCADA and to Modbus TCP result signal.

For delaying signal response, Lua script could be written in many ways, however the most simple and effective one is to determine a wait time before publishing signal values to result signals. So in this case Lua script will look like this:

```
local saved = get(signals.result)--getting result signal which is equated to new variable 'saved'
local command = get(signals.command) --getting command signal which is equated to new variable 'command'

if not command then
    if saved then
        publish(signals.result, saved.value)--this value is published to result signals and saved value
    end
    return 0
end

if string.find(command.attributes, "nt") or string.find(command.attributes, "iv") or
string.find(command.attributes, "ov") then
    --searching if signal has negative attributes
    if command then
        command.attributes = "cot=7,cotn"--equates negative cot values to response signal attributes
        publish(signals.command, command)-- and publishes value to command signals and value
        if saved then
            publish(signals.result, saved.value)--restores saved value to result signals
        end
        return 0
    end
else
    if command then
        command.attributes = "cot=7"
        publish(signals.command, command)--in this cycle command value is being returned as well as
    --cot7 and cot10 values in case given signal is command type and has no negative attributes
        command.attributes = "cot=10"
        publish(signals.command, command) --publishes response to the command
        save(signals.result, command.value)--command value is being saved to result signal
    end
    local sleepTime = 30
    sleep(sleepTime) --before publishing values to result signals script waits 30s
    publish(signals.result, command.value)--in this row values are being published to result signals
```

```
        end
```

After entering values to empty Excel configuration fields, upload the configuration to WCC Lite (it should upload without any errors):

Upload Lua script to script runner and press start. After this,**Status** should show *Running* and script process number will appear.

Open Vinci as IEC104 master, enter IP address of WCC Lite and start communication. Then open another Vinci window and connect Modbus TCP master – select Modbus TCP slave in Vinci and enter the same IP address as set in Excel configuration for Modbus device. With both communications running check **Protocol connections** on WCC Lite web interface, it should show *connected.* From IEC104 Vinci window go to *System* tab. Select command determined in the Excel configuration (50), IOA (1) and value (eg. 2).

Execute the command and check *Imported signals:*

Device	Signal	Value	Units	State	Attributes	Time
Modbus TCP device	Result_modbus	0				2023-08-10 11:10:03.32
IEC104 SCADA system	Command IEC104	2		cmd	asdu=1,cot =6,ioa=1,or g=1,ql=0,ty pe=float	2023-08-10 11:35:40.18
LUA device	command LUA	2		cmd	asdu=1,cot =6,ioa=1,or g=1,ql=0,ty pe=float	2023-08-10 11:35:40.18
LUA device	result LUA	3				2023-08-10 10:04:14.94

After 30s result signals will now have the same value:

Device	Signal	Value	Units	State	Attributes	Time
Modbus TCP device	Result_modbus	2				2023-08-10 11:36:10.52
IEC104 SCADA system	Command IEC104	2		cmd	asdu=1,cot =6,ioa=1,or g=1,ql=0,ty pe=float	2023-08-10 11:35:40.18
LUA device	command LUA	2		cmd	asdu=1,cot =6,ioa=1,or g=1,ql=0,ty pe=float	2023-08-10 11:35:40.18
LUA device	result LUA	2				2023-08-10 11:36:10.20

Signal delay could be used as a tool to synchronize signals so all the values are received at the same time. It can also be used to schedule commands or tasks when delay is required. Since Lua is one of the faster programming languages, it is the most effective instrument to be used is such matters.

Configuration --> Download

Lua script --> Download

Mathematical Operations With Lua (FW version 1.7)

Mathematical operations can be applied to Lua signals as in any other protocol. This can be done by configuring WCC Lite according to solution needed. To create an example which would test multiple mathematical operations, Excel configuration and Lua script is required. Device sheet should look similar to this:

name	device_alias	enable	protocol	execution_type	error_limit	ip	id	mode	scan_rate_ms	retry_count	timeout_ms	bind_address	host	port	asdu_size	cot_size	ioa_size	rwt	swt	t1	t2	t3	time_sync	message_size	cache_size
Modbus TCP device	Modbus_TCP	1	Modbus TCP				1	tcp	300	3	10000	0.0.0.0		502											
IEC104 SCADA system	IEC104_SCADA	1	IEC 60870-5-104 slave									0.0.0.0		2404	2	2	3	8	12	15	10	20	1	249	100
LUA device	LUA0	1	Lua runner	signal	0																				
LUA device	LUA1	1	Lua runner	signal	0																				
LUA device	LUA2	1	Lua runner	signal	0																				
LUA device	LUA3	1	Lua runner	signal	0																				
LUA device	LUA4	1	Lua runner	signal	0																				
LUA device	LUA5	1	Lua runner	signal	0																				
LUA device	LUA6	1	Lua runner	signal	0																				
LUA device	LUA7	1	Lua runner	signal	0																				
LUA device	LUA8	1	Lua runner	signal	0																				
LUA device	LUA9	1	Lua runner	signal	0																				
LUA device	LUA10	1	Lua runner	signal	0																				
LUA device	LUA11	1	Lua runner	signal	0																				
LUA device	LUA12	1	Lua runner	signal	0																				

Each Lua device is created to send result values to Modbus TCP signals with different mathematical functions applied. This way the same Lua script can be reused and is more optimal since signal alias for each device can stay the same. There are many other solutions but this one allows to observe results more clearly.

In the field "ip" for Modbus TCP master, enter IP address of Wi-Fi connection for computer in use. In the field "host" for IEC104 slave protocol enter IP address of WCC Lite device.

Signals for these devices should be mapped in example to this:

signal_name	device_alias	signal_alias	source_device_alias	source_signal_alias	execute	enable	multiply	add	bit_select	min_value	max_value	absolute_threshold	threshold_units	suppression_values	suppression_time_ms	gi	log	number_type	job_todo	tag_job_todo	common_address	info_address	data_type
Result_modbus0	Modbus_TCP	result0	LUA0	result		1											1	SIGNED16	3;0;1	3;0;1			
command LUA	LUA0	command	IEC104_SCADA	Command0	1												1						
result LUA	LUA0	result															1						
Command IEC104.0	IEC104_SCADA	Command0	LUA0	command		1		5								1	1				1	1	50
Result_modbus1	Modbus_TCP	result1	LUA1	result		1											1	SIGNED16	3;1;1	3;1;1			
command LUA	LUA1	command	IEC104_SCADA	Command1	1												1						
result LUA	LUA1	result															1						
Command IEC104.1	IEC104_SCADA	Command1	LUA1	command		1		-5								1	1				1	2	50
Result_modbus2	Modbus_TCP	result2	LUA2	result		1											1	SIGNED16	3;2;1	3;2;1			
command LUA	LUA2	command	IEC104_SCADA	Command2	1												1						
result LUA	LUA2	result															1						
Command IEC104.2	IEC104_SCADA	Command2	LUA2	command		1	5									1	1				1	3	50
Result_modbus3	Modbus_TCP	result3	LUA3	result		1											1	SIGNED16	3;3;1	3;3;1			
command LUA	LUA3	command	IEC104_SCADA	Command3	1												1						
result LUA	LUA3	result															1						
Command IEC104.3	IEC104_SCADA	Command3	LUA3	command		1			1							1	1				1	4	50
Result_modbus4	Modbus_TCP	result4	LUA4	result		1											1	SIGNED16	3;4;1	3;4;1			
command LUA	LUA4	command	IEC104_SCADA	Command4	1												1						
result LUA	LUA4	result															1						
Command IEC104.4	IEC104_SCADA	Command4	LUA4	command		1	6		2							1	1				1	5	50
Result_modbus5	Modbus_TCP	result5	LUA5	result		1											1	SIGNED16	3;5;1	3;5;1			
command LUA	LUA5	command	IEC104_SCADA	Command5	1												1						
result LUA	LUA5	result															1						
Command IEC104.5	IEC104_SCADA	Command5	LUA5	command		1		2	3							1	1				1	6	50
Result_modbus6	Modbus_TCP	result6	LUA6	result		1											1	SIGNED16	3;6;1	3;6;1			
command LUA	LUA6	command	IEC104_SCADA	Command6	1												1						
result LUA	LUA6	result															1						
Command IEC104.6	IEC104_SCADA	Command6	LUA6	command		1	5	34	4							1	1				1	7	50
Result_modbus7	Modbus_TCP	result7	LUA7	result		1											1	SIGNED16	3;7;1	3;7;1			
command LUA	LUA7	command	IEC104_SCADA	Command7	1												1						
result LUA	LUA7	result															1						
Command IEC104.7	IEC104_SCADA	Command7	LUA7	command		1				2	20					1	1				1	8	50

Each Lua device has command and result signals. Command received from IEC104 protocol is sent to Lua command signal and then this signal sends back a response for IEC104 protocol. If the response does not have negative cot

attributes, value is then sent to Lua result signal which sends value to Modbus TCP result signal. Mathematical operations are applied to IEC104 protocol signals since it is the one sending the commands.

As mentioned before Lua script for each Lua device is going to be unchanged and should look like this:

```
local saved = get(signals.result) --getting result signal which is equated to new variable 'saved'
local command = get(signals.command) --getting command signal which is equated to new variable 'command'

--get() function returns nill if there is no valid value

if not command then --if command is not nill
    if saved then --if signal is not nill
        publish(signals.result, saved.value) --this value is published to result signals and saved value
    end
    return 0
end


local time_diff = time_ms() - tonumber(command.time) --compares command time and real time
local is_command = time_diff < 30000 and time_diff > -30000 --if command time differs from
--real time more than 30s it will not be executed


if string.find(command.attributes, "nt") or string.find(command.attributes,
    "iv") or string.find(command.attributes, "ov") then
  --searching if signal has negative attributes
    if is_command then --if command execution time is not exceeding the limits then
        command.attributes = "cot=7,cotn" --equates negative cot values to response signal attributes
        publish(signals.command, command) -- and publishes value to command signals and value
        if saved then --if there is saved value then
            publish(signals.result, saved.value) --restores saved value to result signals
        end
        return 0
    end
else
    if is_command then
        command.attributes = "cot=7"
        publish(signals.command, command)    --in this cycle command value is being returned as well as
    --cot7 and cot10 values in case given signal is command and has no negative attributes
        command.attributes = "cot=10"
        publish(signals.command, command) --publishes response to the command
        save(signals.result, command.value) --command value is being saved to result signal
    end
    publish(signals.result, command.value) --in this row command value is being published to result signals
end
```
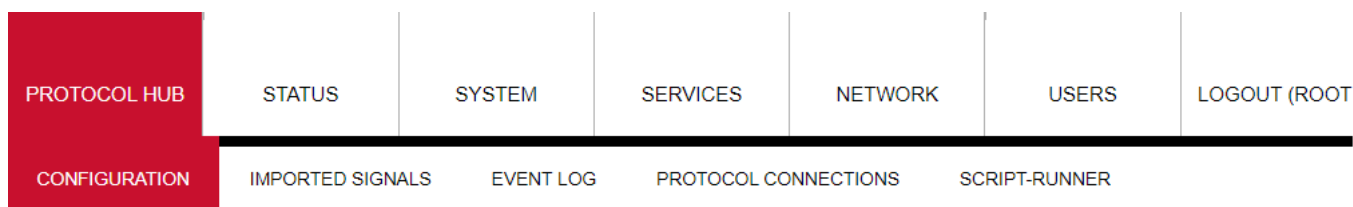
Upload Excel configuration to WCC Lite:

After uploading configuration no errors should appear and all signal should be represented on the web. To upload Lua script go to *Script-Runner*, select *upload script* and then *start* (for each Lua device):

Open Vinci as IEC104 master, enter IP address of WCC Lite and start communication. Then open another Vinci window and connect Modbus TCP master – select Modbus TCP slave in Vinci and enter the same IP address as set in Excel configuration for Modbus device. With both communications running check **Protocol connections** on WCC Lite web interface, it should show *Connected*. From IEC104 Vinci window go to System tab. Select command determined in the Excel configuration (50), IOA (different for each signal) and value. After executing command, each signal (IEC104 command, Lua command, Lua result and Modbus result) will have the same value, which now will be with math applied. For example, command with IOA=1 and value 1 is being executed. In Excel configuration for this signal **add** column has a value of 5, which means that this value is going to be added to the value sent and the result will be 6.

There can be multiple mathematical operations for one signal. For example add, multiply, bit select etc. If that is the case, math will be applied in typical order (eg. first bit select, then multiply, then add). More detailed explanation about mathematical operations in Excel configuration can be found here: Optional parameters for signals

A user can also apply mathematical condition for the signal value. For example minimum or maximum value, threshold, suppression time for specific value etc. Minimum and maximum values can be applied to set the range of the signal, if the value is smaller or larger signal state will show invalid or overflow. Thresholds can be used in many ways. It can be a specific value or a percentage. If the signal value passes set threshold it will be represented on imported signals window. Threshold works by comparing old value with new value and then applying the condition of either representing the value or suppressing it, depending on the value change. Suppression value and suppression time is best used together, because suppression time determines how long the specific value should be suppressed. There could be multiple values set for suppression. In Excel configuration those values should be separated by comma.

Mathematical operations combined with Lua script is useful for may cases. They can be used for filtering data, converting units, applying specific mathematical logic or other solutions.

Configuration --> Download

Lua script --> Download

Cronjob example

Lua device can be configured using cron time expression. This way script execution can be scheduled or executed at certain times. To create such solution, device sheet of Excel configuration should look like this:

name	description	device_alias	enable	protocol	execution_type	execution_parameter	error_limit	ip	id	mode	scan_rate_ms	retry_count	timeout_ms	bind_address	host	port	asdu_size	cot_size	ioa_size	rvt	swt	t1	t2	t3	time_sync	message_size	cache_size
Modbus TCP device	Modbus TCP signals	Modbus_TCP	1	Modbus TCP					1	tcp	300	3	10000	0.0.0.0		502											
IEC104 SCADA system	IEC104 SCADA signals	IEC104_SCADA	1	IEC 60870-5-104 slave										0.0.0.0		2404	2	2	3	8	12	15	10	20	1	249	100
LUA device	LUA signals	LUA	1	Lua runner	date	*/30 * * * * *	0																				

As seen in this configuration, execution type for Lua device is date and execution parameter is in cron time expression. In this case script will be executed every 30s starting from a mm:30 or a mm:00 mark. There are a lot of online cron expression parsers or generators to convert this expression to a more understandable form: https://crontab.cronhub.io/.

A Cron expression must have six variables, for instance, the code "0 * * * *" will not suffice because it contains only five variables. To rectify this, add a "0" to the beginning of the code: "0 0 * * * *". Similarly, the code "0 20-23,0-4,11 * * *" which is displayed as correct on the website, to achieve the effect of every hour from 08:00 PM to 11:59 PM, 12:00 AM to 04:59 AM, and 11:00 AM, it must be adjusted as follows: "0 0 20-23,0-4,11 * * *".

To complete Excel configuration fill out the red fields with correct parameters. For "ip" field enter IP address of Wi-Fi connection that is connected to computer. This can be checked by entering command "ipconfig" on terminal window. For the "host" field enter IP address of WCC Lite device.

Signals sheet should look similar to this:

signal_name	device_alias	signal_alias	source_device_alias	source_signal_alias	execute	enable	gi	log	number_type	job_todo	tag_job_todo	common_address	info_address	data_type
Result_modbus	Modbus_TCP	result	LUA	result		1		1	FLOAT	3;0;1	3;0;1			
command LUA	LUA	command	IEC104_SCADA	Command				1						
result LUA	LUA	result						1						
Command IEC104	IEC104_SCADA	Command	LUA	command	1	1	1					1	1	50

IEC104 SCADA will send command which will then go to Lua signal. Lua signal will send the response back to IEC104 SCADA and to Modbus TCP result signal.

Lua script in this case will be very simple. To show how Lua with cronjob can be used in real life, a calculation for kilowatts per hour has been added:

```
local kW = tonumber(get_value(signals.command))--function "get_value" will get value from iec104 command
--without any attributes this value is still in string form so funtion "tonumber" will convert it to number
--a new variable is creted which is now equal to command value
value = value or 0 --new variable is created. It has to be equal to itself or to 0 if the script is running
--for the forst time
value = (kW * 30/3600) + value --formula for calculating kilowatts. command value from iec104 is multiplied by
time.
--Since the script is being executed every 30s, this time needs to be converted to hours. an old value is being
--added to new value, this way result value that is being published will grow every 30s
publish(signals.result, value)--publishes result to result signals for lua and modbus TCP.
--Since the script is being executed every 30s, those values will be refreshed every 30s as well
```

To test the functionality of this script upload Excel configuration to WCC Lite (it should upload without any errors):

Upload Lua script to script runner and press start. After this,**Status** should show *Running* and script process number will appear:

Open Vinci as IEC104 master, enter IP address of WCC Lite and start communication. Then open another Vinci window and connect Modbus TCP master – select Modbus TCP slave in Vinci and enter the same IP address as set in Excel configuration for Modbus device. With both communications running check **Protocol connections** on WCC Lite web interface, it should show *connected*. From IEC104 Vinci window go to *System* tab. Select command determined in the Excel configuration (50), IOA (1) and value (for example 3600).

Tags | System | Channel

Commands

Read | Test

Clock synchronization

Send | ☐ IV | ☐ SM | ☐ SB

☑ PC time | 2023-08-17 09:14:03 ☐▼

Custom Command

Type: C_SE_NC_1(50) ▼ | IOA: 1

Value: 3600 | QU/QL: 0

Cause: Activation ▼ | SBO delay: 0

Select | Execute

Execute the command and check *Imported signals:*

IMPORTED SIGNALS

Device	Signal	Value	Units	State	Attributes	Time
Modbus TCP device	Result_modbus	0				2023-08-17 10:19:00.86
IEC104 SCADA system	Command IEC104	3600		cmd	asdu=1,cot=6,ioa=1,org=1 ,ql=0,type=float	2023-08-17 10:19:05.20
LUA device	command LUA	3600		cmd	asdu=1,cot=6,ioa=1,org=1 ,ql=0,type=float	2023-08-17 10:19:05.20
LUA device	result LUA					

On a 30s mark result signals will have calculated value:

IMPORTED SIGNALS

Device	Signal	Value	Units	State	Attributes	Time
Modbus TCP device	Result_modbus	30				2023-08-17 10:19:30.35
IEC104 SCADA system	Command IEC104	3600		cmd	asdu=1,cot=6,ioa=1,org=1 ,ql=0,type=float	2023-08-17 10:19:05.20
LUA device	command LUA	3600		cmd	asdu=1,cot=6,ioa=1,org=1 ,ql=0,type=float	2023-08-17 10:19:05.20
LUA device	result LUA	30				2023-08-17 10:19:30.30

After another 30s new value will be added to old value and result signals will be updated accordingly:

IMPORTED SIGNALS

Device	Signal	Value	Units	State	Attributes	Time
Modbus TCP device	Result_modbus	60				2023-08-17 10:20:00.36
IEC104 SCADA system	Command IEC104	3600		cmd	asdu=1,cot=6,ioa=1,org=1 ,ql=0,type=float	2023-08-17 10:19:05.20
LUA device	command LUA	3600		cmd	asdu=1,cot=6,ioa=1,org=1 ,ql=0,type=float	2023-08-17 10:19:05.20
LUA device	result LUA	60				2023-08-17 10:20:00.39

Cronjob with Lua can be used mainly to schedule tasks. It can also be used as a way to filter, monitor or control the data. As seen in this example, Lua script can help calculate certain parameters which will then can be sorted using cron time expression.

Configuration --> Download

Lua script --> Download