HiL-Setup with PSI5, SPI and SQUIB

-Seskion GmbH-

Content:

Version:	(1.0) 09.12.2021 – Creation
	(1.1) 28.01.2022 – minor Improvements

HiL-Setup with PSI5, SPI and SQUIB

1. Overview Seskion HiL-Setup

1.1 Seskion HiL Explanation

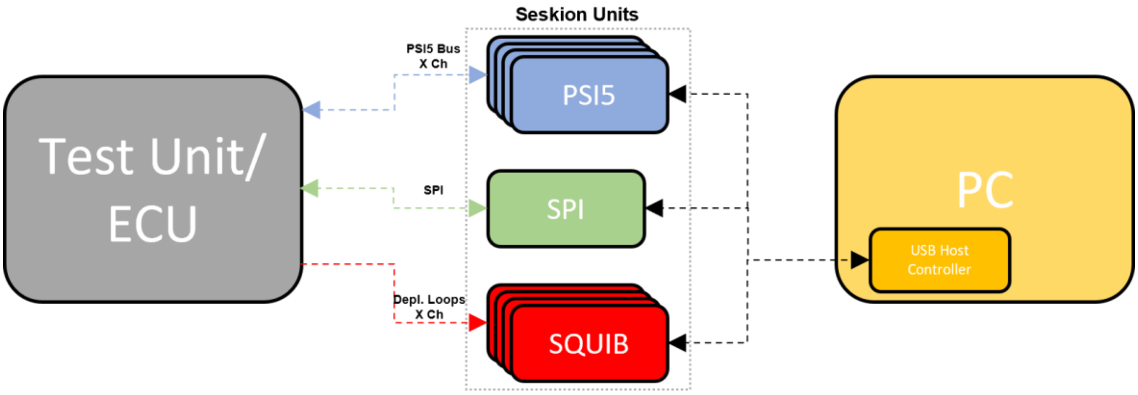
To generate a complete HIL simulation for an airbag control unit, the internal acceleration sensor signals of the control unit (SPI) and the peripheral acceleration and pressure sensor signals (PSI5) must be connected. Furthermore, a PC system connected to the simulators via a USB port/network is used for data processing.

Often up to 6 or more PSI5-Simulyzer are operated together with several SPI-Simulyzer. For coordination there is a superordinate configuration, which determines which Simulyzer is used, which signal traces are used by which Simulyzer and which detailed configurations are to be used with the bus parameters. By using the superordinate configuration, the operation of the API is very simple.

A continuous control of the sensor signals can also be made.

1.2 Possible HiL-Setup

Below you can see a HiL-Setup which is already implemented and in use at several OEMs:



1.3 FAQ

Question: I need to simulate synchronized PSI5 data on all my Test ECU's channels. Is there a way to synchronize multiple Simulyzers so that they start transmission at the same time?
Answer: All Simulyzer's have digital inputs/outputs which either generate or receive a trigger signal. On reception of this trigger signal simulation data is streamed on the PSI5 interfaces.

Question: I will have sensor data from a real-world source which I will need to "replay" in simulation with the Simulyzer tool. Is this possible?
Answer: This data is already queued via USB to the boxes, so there is no delay in between trigger signal and data stream start. The stream data could be imported from .cvs files or provided as binary data on the API interface.

Question: Multiple Simulyzers will need to be linked together in parallel to achieve simulation on all my Test ECU's channels. How many can I link together?
Answer: There is no limitation of parallel used Simulyzer's on the PC. We recommend dedicated USB host controller cards in the PC instead of USB connection on motherboards when using multiple Simulyzer's in parallel.

Question: I need to control through LabVIEW or C-APIs. Do you make all the above requirements possible with your library and API support?
Answer: Our API provide an interface to group more Simulyzers into a so called SimulyzerSystem. This system could be configured with one API call and a configuration file where the Simulyzer's of the system are defined.
One additional API call is needed to provide the simulation data to system. That's all. After receiving the external trigger signal the simulation data is streamed on the PSI5 interface.

2. Detailed view on HiL-System

Real-time system:
Provides data communication and residual bus simulation of a real operation and synchronizes the simulation system by a trigger pulse.

PSI5 Simulyzer:
Peripheral acceleration or pressure sensor signals are simulated by PSI5-Simulyzer boxes. A maximum of 6 sensors can be measured per Simulyzer. Mixed operation with real sensors is possible. Each sensor to be simulated and its signals must be described according to the real-time system.

SPI Simulyzer:
The internal acceleration sensors are simulated by SPI Simulyzer boxes. The number of boxes to be used depends on the number of sensors and the interfaces used. Mixed operation with real sensors is possible. Each sensor to be simulated in the system and its signals must be described according to the real-time system.

Triggering ECU:
Provides the ignition pulse.

PC system:
Data technical environment for sequence control and archiving/analysis of the measurement data.

2.1 Data preparation

The usual PSI5/SPI Simulyzer data export functions can be used to analyze/archive the data.

The generation of errors on protocol level is possible and must be either already included in the provided streaming data or in the provided ppf/spf files must have already been created or defined.
A detailed description can be found in the respective help system of the Simulyzer software.

The detection of the airbag deployment by the SQUIB box is possible (SQUIB replacement).
Special solutions can be realized in consultation with Seskion GmbH - e.g. feeding of position data from GPS sensor or similar.

Other sensor protocols like SENT or DSI3 are also supported.

2.2 Structure of the system - simulation data by means of binary array

The following example system is used for description:

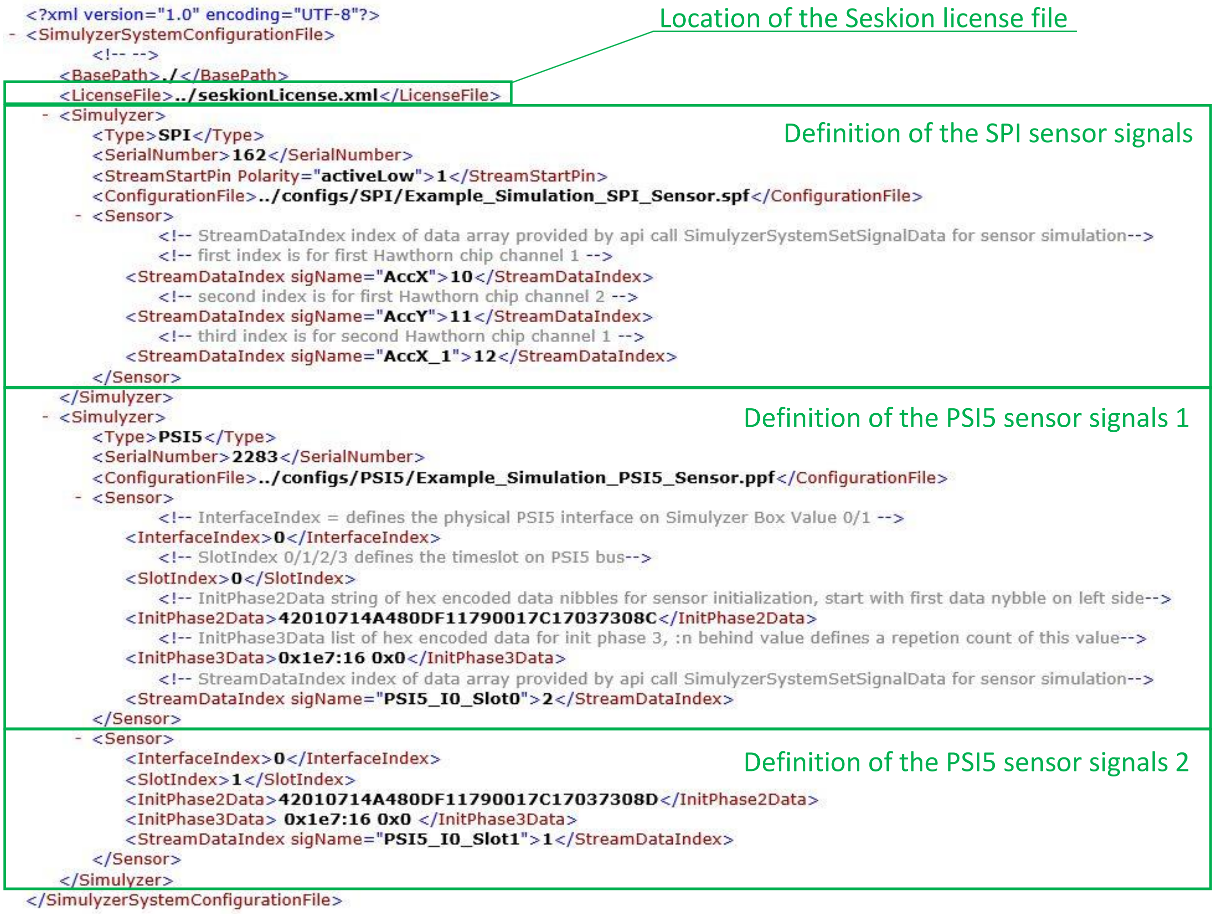
1x PSI5-Simulyzer-Box *Serial No.: 2283* *Channel 0 used* *Channel 1 not used* ***1. Signal sends in slot 0*** *Initdata:* *Phase 2: 42010714A480DF11790017C17037308C* *Phase 3: 0x1e7:16 0x0* *Real-time data signal trace:“PSI5_IO_Slot0 Index 1"* ***2. Signal sends in slot 1*** *Initdata:* *Phase 2: 42010714A480DF11790017C17037308D* *Phase 3: 0x1e7:16 0x0* *Real-time data signal trace: “PSI5_IO_Slot1 Index 1"*	***1x SPI-Simulyzer-Box*** *Serial No.: 162* *Channel 0 used* *Channel 1 not used* ***1. Sensor signal*** *Signal name AccX* *Start of data transmission with falling edge* *Real-time data signal trace:“AccX Index 10"* ***2. Sensor signal*** *Signal name AccY* *Real-time data signal trace:“AccY Index 11"* ***3. Sensor signal*** *Signal name AccX_1* *Real-time data signal trace:“AccX_1 Index 12"*

2.2.1. System configuration file for binary array

The system configuration file is a mandatory file in *.xml format for configuring the HIL system with reference to:

- Simulyzer license verification
- SPI sensor information via spf-file reference (SPI configuration data generated from SPI-Simulyzer software)
- SPI signal name definition according to simulation data.
- PSI5 sensor information via ppf-file (PSI5 configuration data generated from PSI5-Simulyzer software).
- PSI5 signal name definition according to the simulation data.

Example (SimulyzerSystemConfigurationFile.xml):



Location of the Seskion license file

Definition of the SPI sensor signals

Definition of the PSI5 sensor signals 1

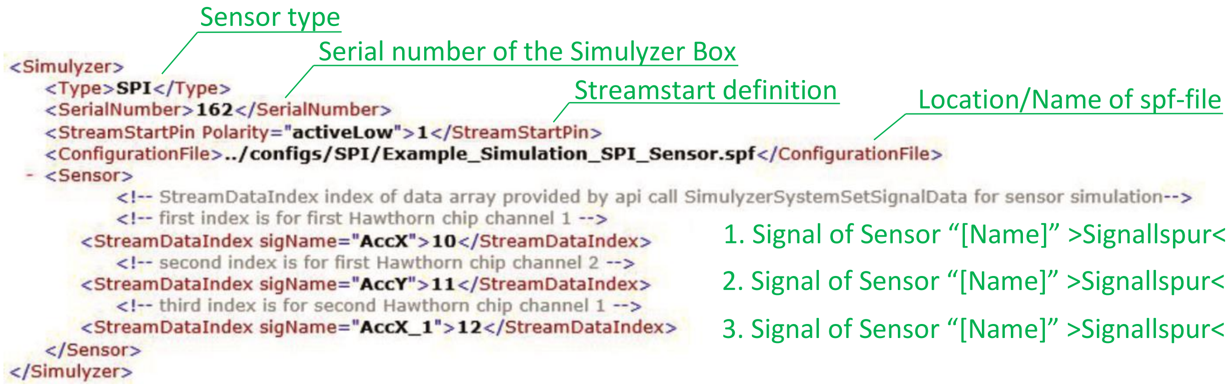
Definition of the PSI5 sensor signals 2

```
  <?xml version="1.0" encoding="UTF-8"?>
- <SimulyzerSystemConfigurationFile>
        <!-- -->
    <BasePath>./</BasePath>
    <LicenseFile>../seskionLicense.xml</LicenseFile>
  - <Simulyzer>
        <Type>SPI</Type>
        <SerialNumber>162</SerialNumber>
        <StreamStartPin Polarity="activeLow">1</StreamStartPin>
        <ConfigurationFile>../configs/SPI/Example_Simulation_SPI_Sensor.spf</ConfigurationFile>
      - <Sensor>
                <!-- StreamDataIndex index of data array provided by api call SimulyzerSystemSetSignalData for sensor simulation-->
                <!-- first index is for first Hawthorn chip channel 1 -->
            <StreamDataIndex sigName="AccX">10</StreamDataIndex>
                <!-- second index is for first Hawthorn chip channel 2 -->
            <StreamDataIndex sigName="AccY">11</StreamDataIndex>
                <!-- third index is for second Hawthorn chip channel 1 -->
            <StreamDataIndex sigName="AccX_1">12</StreamDataIndex>
        </Sensor>
    </Simulyzer>
  - <Simulyzer>
        <Type>PSI5</Type>
        <SerialNumber>2283</SerialNumber>
        <ConfigurationFile>../configs/PSI5/Example_Simulation_PSI5_Sensor.ppf</ConfigurationFile>
      - <Sensor>
                <!-- InterfaceIndex = defines the physical PSI5 interface on Simulyzer Box Value 0/1 -->
            <InterfaceIndex>0</InterfaceIndex>
                <!-- SlotIndex 0/1/2/3 defines the timeslot on PSI5 bus-->
            <SlotIndex>0</SlotIndex>
                <!-- InitPhase2Data string of hex encoded data nibbles for sensor initialization, start with first data nybble on left side-->
            <InitPhase2Data>42010714A480DF11790017C17037308C</InitPhase2Data>
                <!-- InitPhase3Data list of hex encoded data for init phase 3, :n behind value defines a repetion count of this value-->
            <InitPhase3Data>0x1e7:16 0x0</InitPhase3Data>
                <!-- StreamDataIndex index of data array provided by api call SimulyzerSystemSetSignalData for sensor simulation-->
            <StreamDataIndex sigName="PSI5_I0_Slot0">2</StreamDataIndex>
        </Sensor>
      - <Sensor>
            <InterfaceIndex>0</InterfaceIndex>
            <SlotIndex>1</SlotIndex>
            <InitPhase2Data>42010714A480DF11790017C17037308D</InitPhase2Data>
            <InitPhase3Data> 0x1e7:16 0x0 </InitPhase3Data>
            <StreamDataIndex sigName="PSI5_I0_Slot1">1</StreamDataIndex>
        </Sensor>
    </Simulyzer>
  </SimulyzerSystemConfigurationFile>
```

Definition of the SPI Sensors

For each SPI-Simulyzer box a definition must be made according to the following scheme, it does not matter if more than the sensor signals defined here are present in the real-time data or not. All undefined signals are ignored.

In the example only one SPI-Simulyzer box is used, therefore only one SPI definition block starting and ending with <Simulyzer> is defined. There must be a definition block for each Simulyzer box used.



```
<Simulyzer>
   <Type>SPI</Type>
   <SerialNumber>162</SerialNumber>
   <StreamStartPin Polarity="activeLow">1</StreamStartPin>
   <ConfigurationFile>../configs/SPI/Example_Simulation_SPI_Sensor.spf</ConfigurationFile>
 - <Sensor>
         <!-- StreamDataIndex index of data array provided by api call SimulyzerSystemSetSignalData for sensor simulation-->
         <!-- first index is for first Hawthorn chip channel 1 -->
      <StreamDataIndex sigName="AccX">10</StreamDataIndex>
         <!-- second index is for first Hawthorn chip channel 2 -->
      <StreamDataIndex sigName="AccY">11</StreamDataIndex>
         <!-- third index is for second Hawthorn chip channel 1 -->
      <StreamDataIndex sigName="AccX_1">12</StreamDataIndex>
   </Sensor>
</Simulyzer>
```

Sensor type
Serial number of the Simulyzer Box
Streamstart definition
Location/Name of spf-file
1. Signal of Sensor “[Name]” >Signallspur<
2. Signal of Sensor “[Name]” >Signallspur<
3. Signal of Sensor “[Name]” >Signallspur<

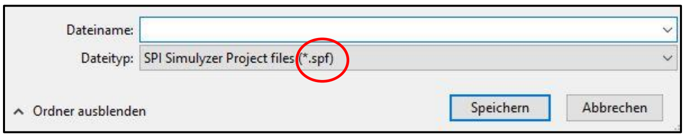
<u>Sensor type:</u> Type of sensor

<u>Serial number of the SPI-Simulyzer box</u>

<u>StreamstartPinPolarity-Definition:</u> Definition of the start pulse for data transmission used by the Simulyzer box as master for all connected Simulyzer boxes. In the example, a pulse is sent to all Simulyzer boxes at pin 1 and data transmission is started with its falling edge.
If this streamstart pulse comes from the system or automatically, no definition is required.

<u>Path/filename of the spf-file</u> - configuration file generated from the SPI-Simulyzer software. (see SPI-Simulyzer webhelp)

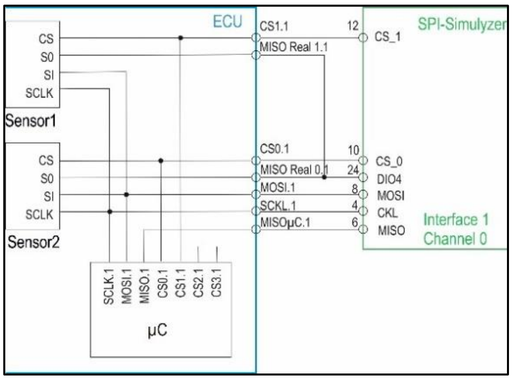
Generation of a spf-file from the Simulyzer software: Menu File - Command Save as...



Dateiname:
Dateityp: SPI Simulyzer Project files (*.spf)

<u>Signal definition</u>

One SPI-Simulyzer box can simulate a maximum of 4 SPI sensors. These 4 SPI sensors can be distributed on one or both interfaces.



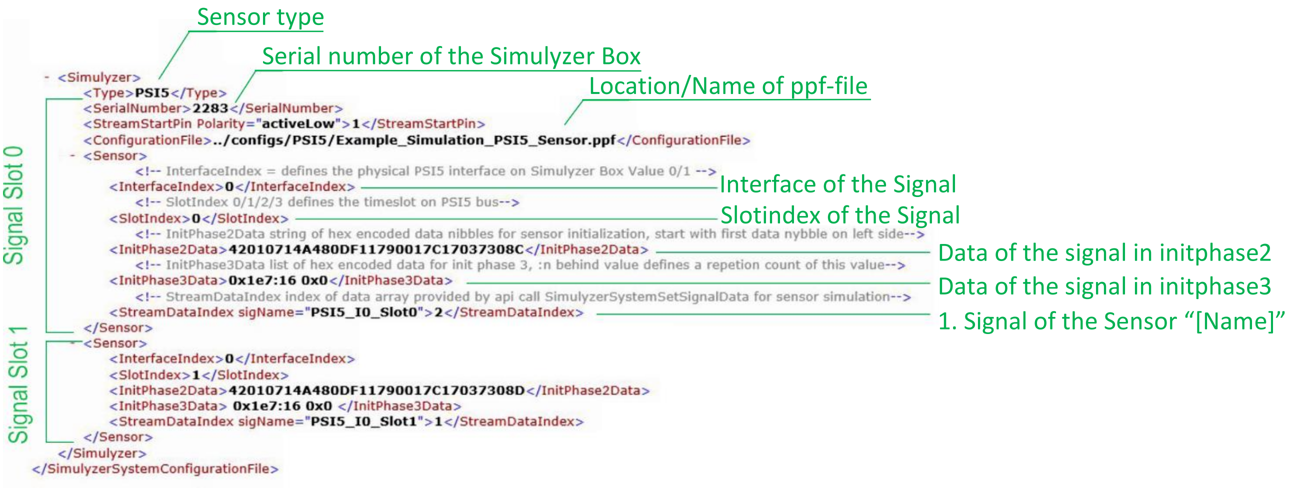
Example Wiring

The signal definition must be done according to the data of the binary array.
Each signal trace is defined by the unique defined signal name in the binary array and the corresponding stream index of the binary array.

Definition of the PSI5 sensors

For each PSI5 sensor signal a definition must be made according to the following scheme. It does not matter whether more than the sensor signals defined here are present in the real-time data or not.
All undefined signals are ignored.
In the example only one PSI5-Simulyzer box is used, therefore only one PSI5 definition block starting and ending with <Simulyzer> is defined. There must be one definition block for each Simulyzer box used.

(For example data see "2.2.1. System configuration" page 4).



```
- <Simulyzer>
    <Type>PSI5</Type>
    <SerialNumber>2283</SerialNumber>
    <StreamStartPin Polarity="activeLow">1</StreamStartPin>
    <ConfigurationFile>../configs/PSI5/Example_Simulation_PSI5_Sensor.ppf</ConfigurationFile>
  - <Sensor>
        <!-- InterfaceIndex = defines the physical PSI5 interface on Simulyzer Box Value 0/1 -->
      <InterfaceIndex>0</InterfaceIndex>
        <!-- SlotIndex 0/1/2/3 defines the timeslot on PSI5 bus-->
      <SlotIndex>0</SlotIndex>
        <!-- InitPhase2Data string of hex encoded data nibbles for sensor initialization, start with first data nybble on left side-->
      <InitPhase2Data>42010714A480DF11790017C17037308C</InitPhase2Data>
        <!-- InitPhase3Data list of hex encoded data for init phase 3, :n behind value defines a repetion count of this value-->
      <InitPhase3Data>0x1e7:16 0x0</InitPhase3Data>
        <!-- StreamDataIndex index of data array provided by api call SimulyzerSystemSetSignalData for sensor simulation-->
      <StreamDataIndex sigName="PSI5_IO_Slot0">2</StreamDataIndex>
    </Sensor>
  - <Sensor>
      <InterfaceIndex>0</InterfaceIndex>
      <SlotIndex>1</SlotIndex>
      <InitPhase2Data>42010714A480DF11790017C17037308D</InitPhase2Data>
      <InitPhase3Data> 0x1e7:16 0x0 </InitPhase3Data>
      <StreamDataIndex sigName="PSI5_IO_Slot1">1</StreamDataIndex>
    </Sensor>
  </Simulyzer>
</SimulyzerSystemConfigurationFile>
```

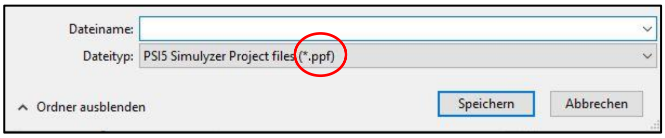
Sensor type
Serial number of the Simulyzer Box
Location/Name of ppf-file
Interface of the Signal
Slotindex of the Signal
Data of the signal in initphase2
Data of the signal in initphase3
1. Signal of the Sensor "[Name]"
Signal Slot 0
Signal Slot 1

Sensor type: Type of sensor

Serial number of the PSI5-Simulyzer box

Path/filename of the ppf-file - which can be generated from the Simulyzer software and describes the PSI5 sensor. (see PSI5-Simulyzer webhelp)

Generation of a ppf-file from the Simulyzer software:
Menu File - Command Save as...

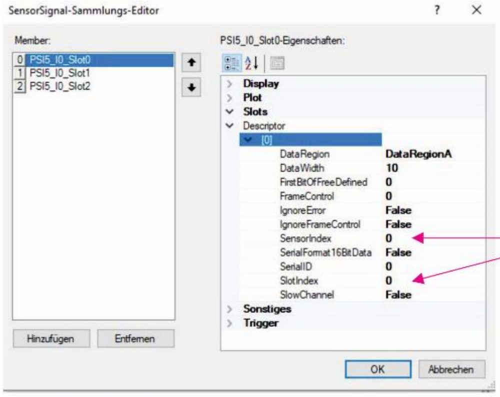


Dateiname:
Dateityp: PSI5 Simulyzer Project files (*.ppf)

Interface: Channel on which the PSI5 sensor signal sends (Channel 0 or 1).

<u>Slot index:</u> Slot index in which the PSI5 sensor sends its data.
The assignment of the time slot in which the data signal is transmitted in the PSI5 real-time system must be entered accordingly.

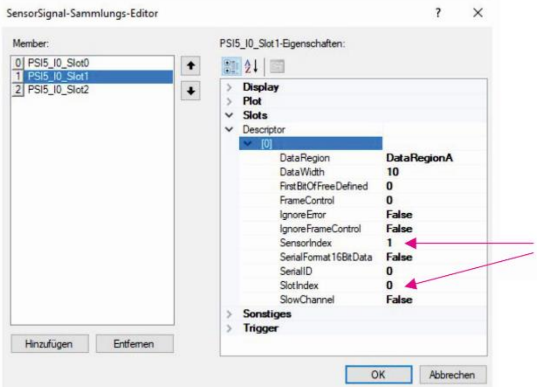
Signal 1 used in simulation
Signal 2 used in simulation
Signal 3 not used in simulation



DataRegion DataRegionA
DataWidth 10
FirstBitOfFreeDefined 0
FrameControl 0
IgnoreError False
IgnoreFrameControl False
SensorIndex 0
SerialFormat16BitData False
SerialID 0
SlotIndex 0
SlowChannel False

PSI5-Simulyzer-Software
Signal-Collection-Editor

1st signal comes from
Sensor 0 in timeslot 0

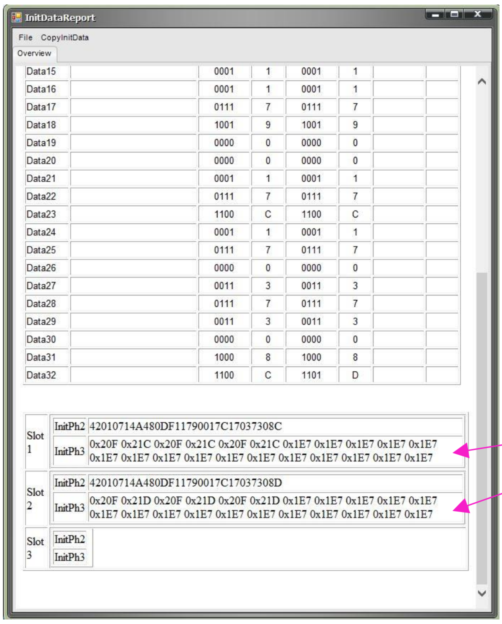


DataRegion DataRegionA
DataWidth 10
FirstBitOfFreeDefined 0
FrameControl 0
IgnoreError False
IgnoreFrameControl False
SensorIndex 1
SerialFormat16BitData False
SerialID 0
SlotIndex 0
SlowChannel False

2nd signal comes from
Sensor 1 in timeslot 0

According to this assignment, the definition must be made in the configuration file.

<u>Initphase 2/3:</u> Hexadecimal data sent by the sensor during the initialization phase.
The data can be copied and pasted in the PSI5 software via the Init Data Report.



Data15		0001	1	0001	1		
Data16		0001	1	0001	1		
Data17		0111	7	0111	7		
Data18		1001	9	1001	9		
Data19		0000	0	0000	0		
Data20		0000	0	0000	0		
Data21		0001	1	0001	1		
Data22		0111	7	0111	7		
Data23		1100	C	1100	C		
Data24		0001	1	0001	1		
Data25		0111	7	0111	7		
Data26		0000	0	0000	0		
Data27		0011	3	0011	3		
Data28		0111	7	0111	7		
Data29		0011	3	0011	3		
Data30		0000	0	0000	0		
Data31		1000	8	1000	8		
Data32		1100	C	1101	D		

Slot 1	InitPh2	42010714A480DF11790017C17037308C
	InitPh3	0x20F 0x21C 0x20F 0x21C 0x20F 0x21C 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7
Slot 2	InitPh2	42010714A480DF11790017C17037308D
	InitPh3	0x20F 0x21D 0x20F 0x21D 0x20F 0x21D 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7
Slot 3	InitPh2	
	InitPh3	

Hexadecimal values of the init data
for Slot1 and Slot2

For transfer with copy/paste

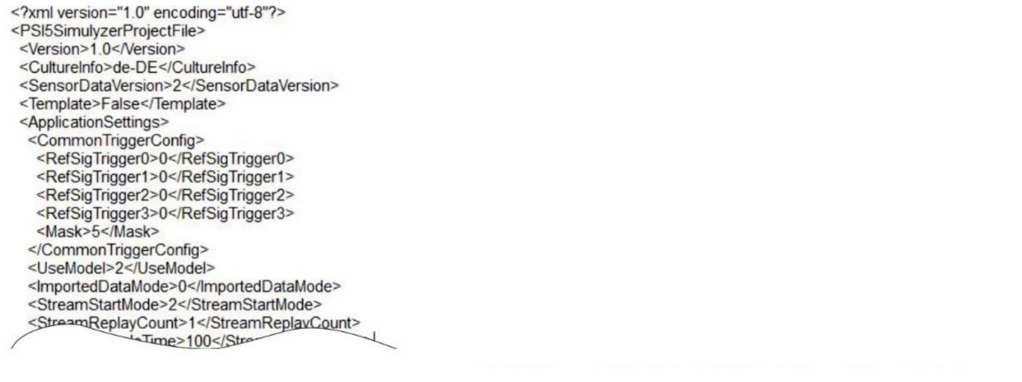
<u>Streamdataindex(signal data definition)</u>
according to the number of signals and the signal names defined in the binary data array.
Each signal track is identified by

- the unique defined signal name and a
- 1-digit signal index of the defined binary data array.

2.2.2. PPF file

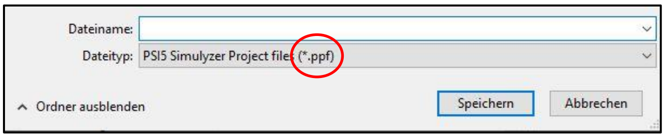
The ppf file contains all characteristic PSI5 bus data (type, timeslot definition, data length, data count and many more).
The file can be generated by the PSI5-Simulyzer software.
There must be one file per PSI5-Simulyzer box used.

Example of a ppf file:



```
<?xml version="1.0" encoding="utf-8"?>
<PSI5SimulyzerProjectFile>
 <Version>1.0</Version>
 <CultureInfo>de-DE</CultureInfo>
 <SensorDataVersion>2</SensorDataVersion>
 <Template>False</Template>
 <ApplicationSettings>
  <CommonTriggerConfig>
   <RefSigTrigger0>0</RefSigTrigger0>
   <RefSigTrigger1>0</RefSigTrigger1>
   <RefSigTrigger2>0</RefSigTrigger2>
   <RefSigTrigger3>0</RefSigTrigger3>
   <Mask>5</Mask>
  </CommonTriggerConfig>
  <UseModel>2</UseModel>
  <ImportedDataMode>0</ImportedDataMode>
  <StreamStartMode>2</StreamStartMode>
  <StreamReplayCount>1</StreamReplayCount>
      Time>100</Str
```

Generation of a ppf-file from the Simulyzer software:
Menu File - Command Save as...

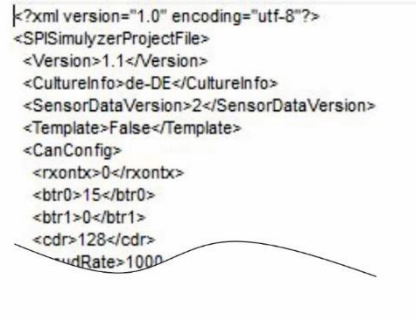


Dateiname:
Dateityp: PSI5 Simulyzer Project files (*.ppf)

2.2.3. SPF file

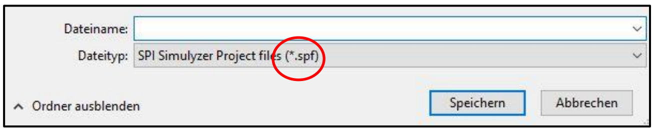
The spf file contains all characteristic PSI5 bus data (baud rate, CS allocation, data count and many more).
The file can be generated by the PSI5-Simulyzer software.
There must be one file for each SPI-Simulyzer box used.

Example of an spf file:



```
<?xml version="1.0" encoding="utf-8"?>
<SPISimulyzerProjectFile>
 <Version>1.1</Version>
 <CultureInfo>de-DE</CultureInfo>
 <SensorDataVersion>2</SensorDataVersion>
 <Template>False</Template>
 <CanConfig>
  <rxontx>0</rxontx>
  <btr0>15</btr0>
  <btr1>0</btr1>
  <cdr>128</cdr>
    dRate>1000
```

Generation of a spf-file from the Simulyzer software:
Menu File - Command Save as...



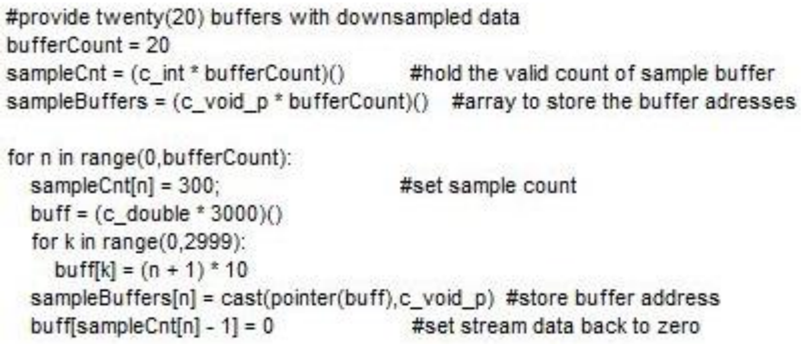
Dateiname:
Dateityp: SPI Simulyzer Project files (*.spf)

2.2.4. Program for process flow control

The program controls the entire HIL test system and regulates the complete measurement process.

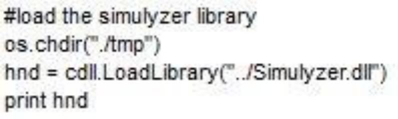
Possible programming languages: API: ANSI-C, Python, dotNET
Knowledge of the corresponding programming language is assumed.
In principle, the API contains the following structure (example Python)

1. definition of the binary array



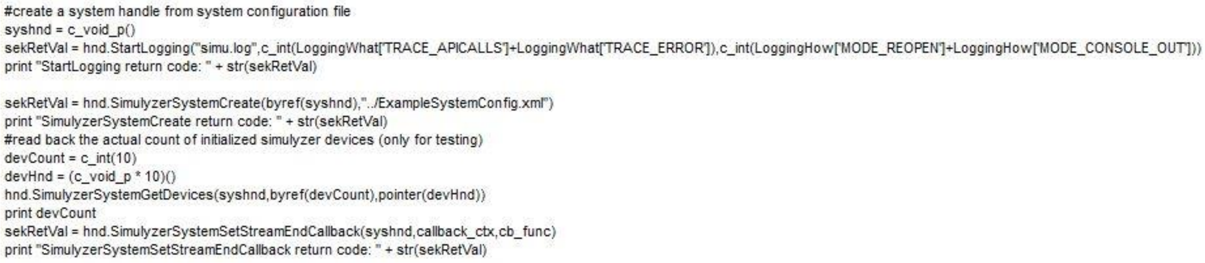
```
#provide twenty(20) buffers with downsampled data
bufferCount = 20
sampleCnt = (c_int * bufferCount)()          #hold the valid count of sample buffer
sampleBuffers = (c_void_p * bufferCount)()   #array to store the buffer adresses

for n in range(0,bufferCount):
  sampleCnt[n] = 300;                        #set sample count
  buff = (c_double * 3000)()
  for k in range(0,2999):
    buff[k] = (n + 1) * 10
  sampleBuffers[n] = cast(pointer(buff),c_void_p)  #store buffer address
  buff[sampleCnt[n] - 1] = 0                  #set stream data back to zero
```

2. loading the Simulyzer driver



```
#load the simulyzer library
os.chdir("./tmp")
hnd = cdll.LoadLibrary("../Simulyzer.dll")
print hnd
```

3. generate the measuring system and link it to the system configuration.xml

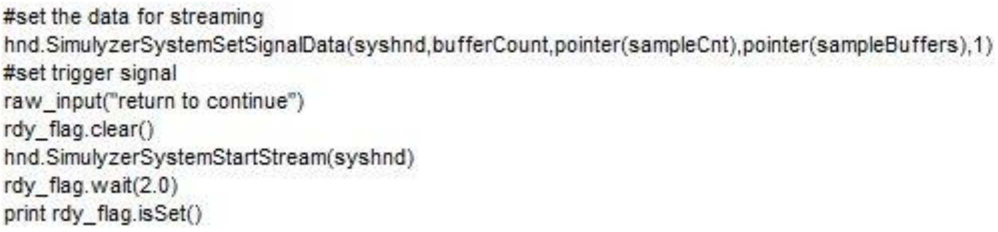


```
#create a system handle from system configuration file
syshnd = c_void_p()
sekRetVal = hnd.StartLogging("simu.log",c_int(LoggingWhat['TRACE_APICALLS']+LoggingWhat['TRACE_ERROR']),c_int(LoggingHow['MODE_REOPEN']+LoggingHow['MODE_CONSOLE_OUT']))
print "StartLogging return code: " + str(sekRetVal)

sekRetVal = hnd.SimulyzerSystemCreate(byref(syshnd),"../ExampleSystemConfig.xml")
print "SimulyzerSystemCreate return code: " + str(sekRetVal)
#read back the actual count of initialized simulyzer devices (only for testing)
devCount = c_int(10)
devHnd = (c_void_p * 10)()
hnd.SimulyzerSystemGetDevices(syshnd,byref(devCount),pointer(devHnd))
print devCount
sekRetVal = hnd.SimulyzerSystemSetStreamEndCallback(syshnd,callback_ctx,cb_func)
print "SimulyzerSystemSetStreamEndCallback return code: " + str(sekRetVal)
```

4. login - check of proper system setup and readiness of individual components.

5. start command of the measuring system/start of the trigger pulse

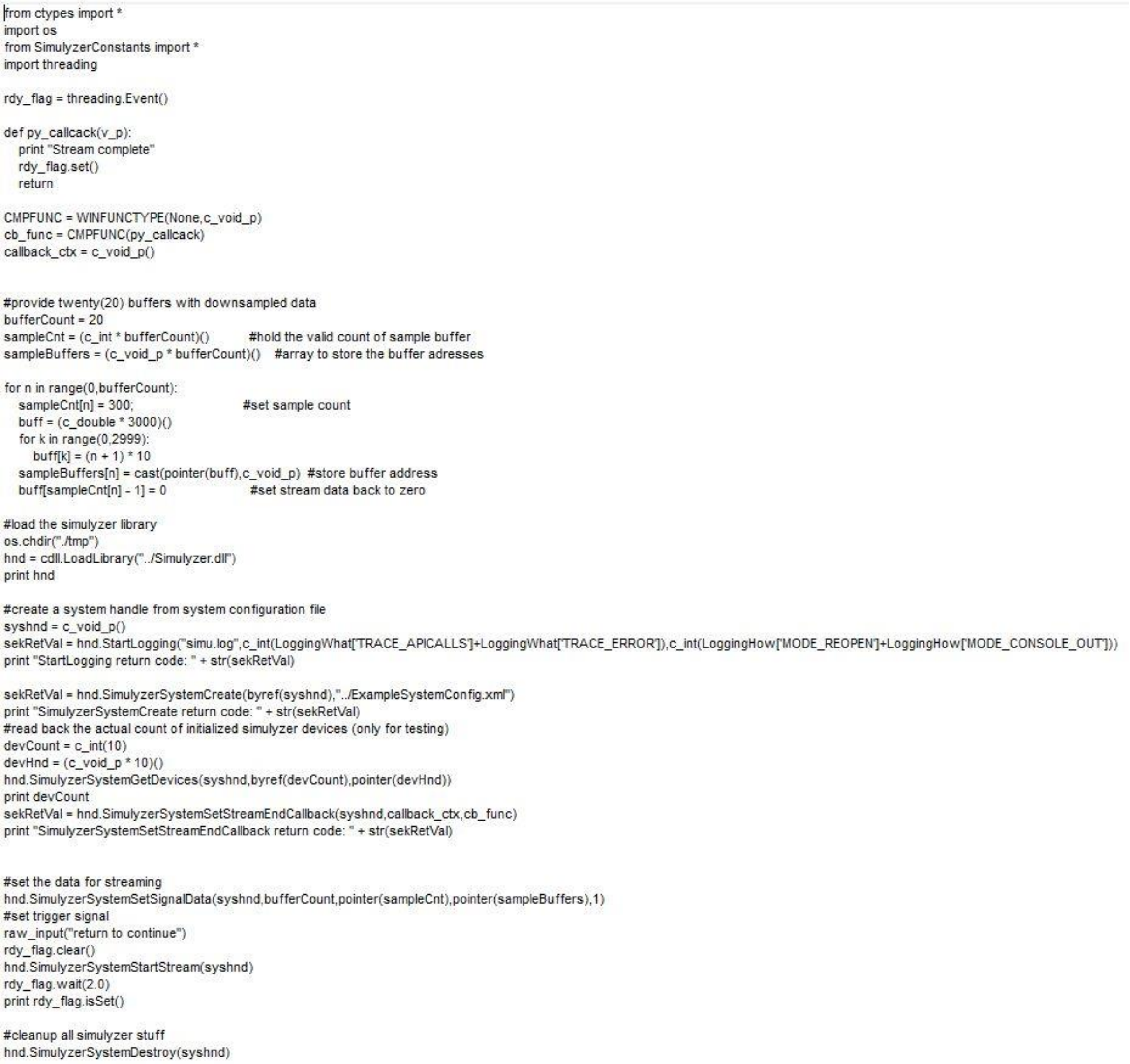
6. stop of the measuring system - stop command



```
#set the data for streaming
hnd.SimulyzerSystemSetSignalData(syshnd,bufferCount,pointer(sampleCnt),pointer(sampleBuffers),1)
#set trigger signal
raw_input("return to continue")
rdy_flag.clear()
hnd.SimulyzerSystemStartStream(syshnd)
rdy_flag.wait(2.0)
print rdy_flag.isSet()
```

The individual extension is up to the user and can be carried out without restriction in view of the system processes can be carried out.

Example complete API (Python)



```
from ctypes import *
import os
from SimulyzerConstants import *
import threading

rdy_flag = threading.Event()

def py_callcack(v_p):
   print "Stream complete"
   rdy_flag.set()
   return

CMPFUNC = WINFUNCTYPE(None,c_void_p)
cb_func = CMPFUNC(py_callcack)
callback_ctx = c_void_p()


#provide twenty(20) buffers with downsampled data
bufferCount = 20
sampleCnt = (c_int * bufferCount)()          #hold the valid count of sample buffer
sampleBuffers = (c_void_p * bufferCount)()   #array to store the buffer adresses

for n in range(0,bufferCount):
   sampleCnt[n] = 300;                        #set sample count
   buff = (c_double * 3000)()
   for k in range(0,2999):
      buff[k] = (n + 1) * 10
   sampleBuffers[n] = cast(pointer(buff),c_void_p)  #store buffer address
   buff[sampleCnt[n] - 1] = 0                  #set stream data back to zero

#load the simulyzer library
os.chdir("./tmp")
hnd = cdll.LoadLibrary("../Simulyzer.dll")
print hnd

#create a system handle from system configuration file
syshnd = c_void_p()
sekRetVal = hnd.StartLogging("simu.log",c_int(LoggingWhat['TRACE_APICALLS']+LoggingWhat['TRACE_ERROR']),c_int(LoggingHow['MODE_REOPEN']+LoggingHow['MODE_CONSOLE_OUT']))
print "StartLogging return code: " + str(sekRetVal)

sekRetVal = hnd.SimulyzerSystemCreate(byref(syshnd),"../ExampleSystemConfig.xml")
print "SimulyzerSystemCreate return code: " + str(sekRetVal)
#read back the actual count of initialized simulyzer devices (only for testing)
devCount = c_int(10)
devHnd = (c_void_p * 10)()
hnd.SimulyzerSystemGetDevices(syshnd,byref(devCount),pointer(devHnd))
print devCount
sekRetVal = hnd.SimulyzerSystemSetStreamEndCallback(syshnd,callback_ctx,cb_func)
print "SimulyzerSystemSetStreamEndCallback return code: " + str(sekRetVal)


#set the data for streaming
hnd.SimulyzerSystemSetSignalData(syshnd,bufferCount,pointer(sampleCnt),pointer(sampleBuffers),1)
#set trigger signal
raw_input("return to continue")
rdy_flag.clear()
hnd.SimulyzerSystemStartStream(syshnd)
rdy_flag.wait(2.0)
print rdy_flag.isSet()

#cleanup all simulyzer stuff
hnd.SimulyzerSystemDestroy(syshnd)
```

2.3. Structure of the system - simulation data via CSV file

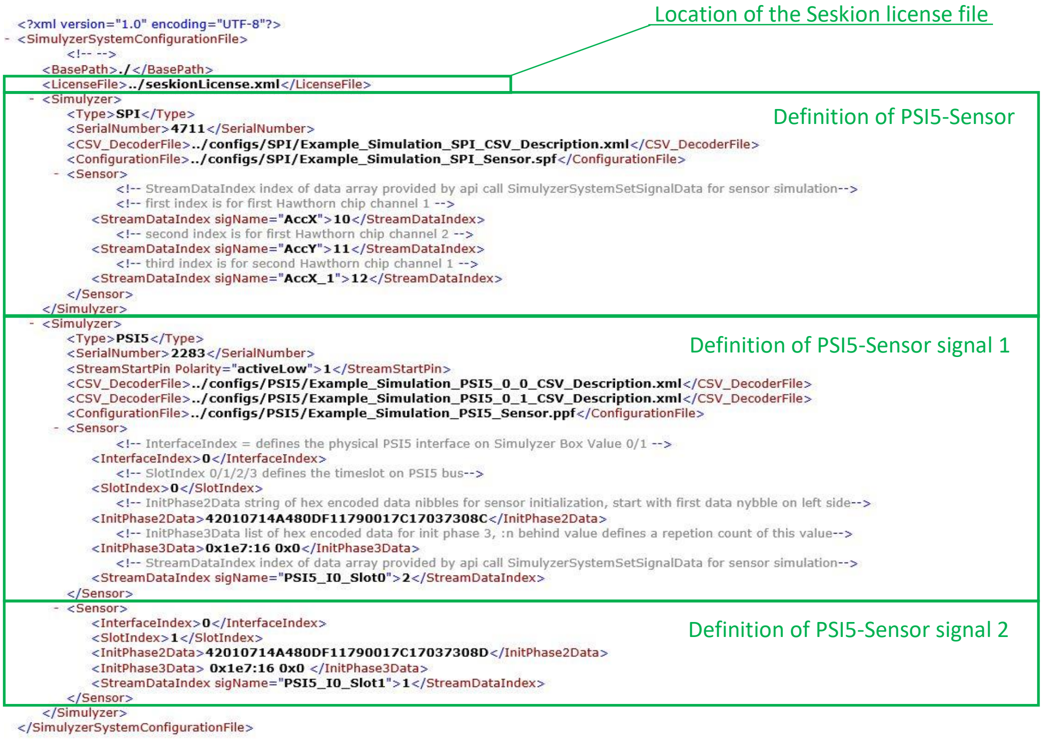
The following example system is used for description:

1x PSI5-Simulyzer-Box	***1x SPI-Simulyzer-Box***
Serial No.: 2283 *Channel 0 used* *Channel 1 not used* ***1. Signal sends in slot 0*** *Initdaten:* *Phase 2: 42010714A480DF11790017C17037308C* *Phase 3: 0x1e7:16 0x0* *Real-time data signal trace:"PSI5_IO_Slot0 Index 1"* ***2. Signal sends in slot 1*** *Initdaten:* *Phase 2: 42010714A480DF11790017C17037308D* *Phase 3: 0x1e7:16 0x0* *Real-time data signal trace: "PSI5_IO_Slot1 Index 1"*	*Serial No.: 162* *Channel 0 used* *Channel 1 not used* ***1. Sensor signal*** *Signal name AccX* *Start of data transmission with falling edge* *Real-time data signal trace:"AccX Index 10"* ***2. Sensor signal*** *Signal name AccY* *Real-time data signal trace:"AccY Index 11"* ***3. Sensor signal*** *Signal name AccX_1* *Real-time data signal trace:"AccX_1 Index 12"*

2.3.1. System configuration file for CSV data

Mandatory file in *xml format to configure the HIL test system regarding:

- Simulyzer license verification
- Decoder file reference SPI signals for data preparation
- SPI sensor information via spf-file reference (SPI configuration data generated from SPI-Simulyzer software)
- SPI signal name definition according to simulation data.
- Reference to - PSI5 bus *csv file
- PSI5 sensor information via ppf-file (PSI5 configuration data generated from PSI5-Simulyzer software)
- Decoder file reference PSI5 signals for data preparation
- PSI5 sensor information via ppf-file (PSI5 configuration data generated from PSI5-Simulyzer software)
- PSI5 signal name definition according to simulation data.


Location of the Seskion license file
Definition of PSI5-Sensor
Definition of PSI5-Sensor signal 1
Definition of PSI5-Sensor signal 2

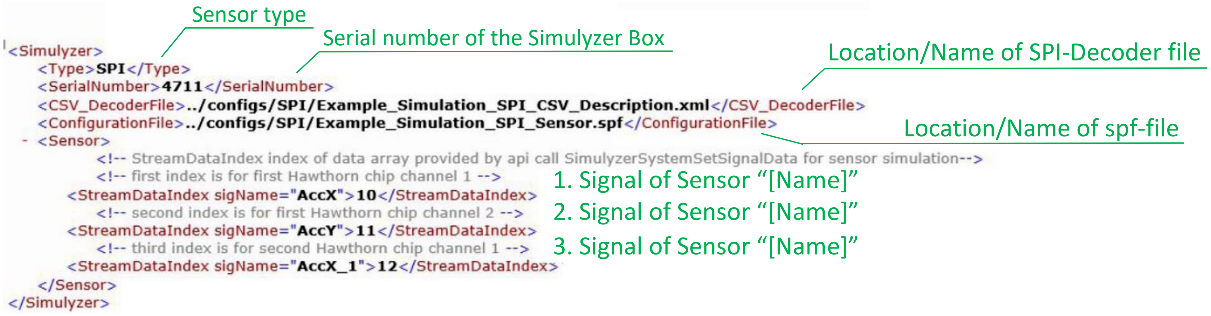
```
  <?xml version="1.0" encoding="UTF-8"?>
- <SimulyzerSystemConfigurationFile>
        <!-- -->
     <BasePath>./</BasePath>
     <LicenseFile>../seskionLicense.xml</LicenseFile>
   - <Simulyzer>
        <Type>SPI</Type>
        <SerialNumber>4711</SerialNumber>
        <CSV_DecoderFile>../configs/SPI/Example_Simulation_SPI_CSV_Description.xml</CSV_DecoderFile>
        <ConfigurationFile>../configs/SPI/Example_Simulation_SPI_Sensor.spf</ConfigurationFile>
      - <Sensor>
              <!-- StreamDataIndex index of data array provided by api call SimulyzerSystemSetSignalData for sensor simulation-->
              <!-- first index is for first Hawthorn chip channel 1 -->
           <StreamDataIndex sigName="AccX">10</StreamDataIndex>
              <!-- second index is for first Hawthorn chip channel 2 -->
           <StreamDataIndex sigName="AccY">11</StreamDataIndex>
              <!-- third index is for second Hawthorn chip channel 1 -->
           <StreamDataIndex sigName="AccX_1">12</StreamDataIndex>
        </Sensor>
     </Simulyzer>
   - <Simulyzer>
        <Type>PSI5</Type>
        <SerialNumber>2283</SerialNumber>
        <StreamStartPin Polarity="activeLow">1</StreamStartPin>
        <CSV_DecoderFile>../configs/PSI5/Example_Simulation_PSI5_0_0_CSV_Description.xml</CSV_DecoderFile>
        <CSV_DecoderFile>../configs/PSI5/Example_Simulation_PSI5_0_1_CSV_Description.xml</CSV_DecoderFile>
        <ConfigurationFile>../configs/PSI5/Example_Simulation_PSI5_Sensor.ppf</ConfigurationFile>
      - <Sensor>
              <!-- InterfaceIndex = defines the physical PSI5 interface on Simulyzer Box Value 0/1 -->
           <InterfaceIndex>0</InterfaceIndex>
              <!-- SlotIndex 0/1/2/3 defines the timeslot on PSI5 bus-->
           <SlotIndex>0</SlotIndex>
              <!-- InitPhase2Data string of hex encoded data nibbles for sensor initialization, start with first data nybble on left side-->
           <InitPhase2Data>42010714A480DF11790017C17037308C</InitPhase2Data>
              <!-- InitPhase3Data list of hex encoded data for init phase 3, :n behind value defines a repetion count of this value-->
           <InitPhase3Data>0x1e7:16 0x0</InitPhase3Data>
              <!-- StreamDataIndex index of data array provided by api call SimulyzerSystemSetSignalData for sensor simulation-->
           <StreamDataIndex sigName="PSI5_IO_Slot0">2</StreamDataIndex>
        </Sensor>
      - <Sensor>
           <InterfaceIndex>0</InterfaceIndex>
           <SlotIndex>1</SlotIndex>
           <InitPhase2Data>42010714A480DF11790017C17037308D</InitPhase2Data>
           <InitPhase3Data> 0x1e7:16 0x0 </InitPhase3Data>
           <StreamDataIndex sigName="PSI5_IO_Slot1">1</StreamDataIndex>
        </Sensor>
     </Simulyzer>
  </SimulyzerSystemConfigurationFile>
```

Definition of the SPI sensors

For each SPI sensor signal a definition must be made according to the following scheme. It does not matter whether more than the sensor signals defined here are present in the simulation data or not.

All undefined signals are ignored.

In the example only one SPI-Simulyzer box is used, therefore only one SPI definition block starting and ending with <Simulyzer> is defined. There must be a definition block for each Simulyzer box used.



```
<Simulyzer>
    <Type>SPI</Type>
    <SerialNumber>4711</SerialNumber>
    <CSV_DecoderFile>../configs/SPI/Example_Simulation_SPI_CSV_Description.xml</CSV_DecoderFile>
    <ConfigurationFile>../configs/SPI/Example_Simulation_SPI_Sensor.spf</ConfigurationFile>
  - <Sensor>
            <!-- StreamDataIndex index of data array provided by api call SimulyzerSystemSetSignalData for sensor simulation-->
            <!-- first index is for first Hawthorn chip channel 1 -->
        <StreamDataIndex sigName="AccX">10</StreamDataIndex>
            <!-- second index is for first Hawthorn chip channel 2 -->
        <StreamDataIndex sigName="AccY">11</StreamDataIndex>
            <!-- third index is for second Hawthorn chip channel 1 -->
        <StreamDataIndex sigName="AccX_1">12</StreamDataIndex>
    </Sensor>
</Simulyzer>
```

Sensor type
Serial number of the Simulyzer Box
Location/Name of SPI-Decoder file
Location/Name of spf-file
1. Signal of Sensor “[Name]”
2. Signal of Sensor “[Name]”
3. Signal of Sensor “[Name]”

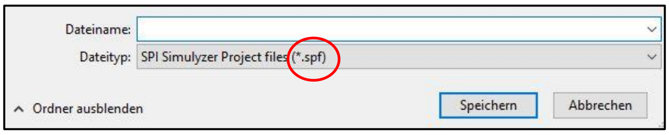
Sensor type: Type of sensor

Serial number of the SPI-Simulyzer box.

Path/filename SPI decoder csv file

Path/filename of the spf-file - Configuration file generated from the SPI-Simulyzer software. (see SPI-Simulyzer webhelp).

Creation of an spf-file from the Simulyzer software:
Menu File - Command Save as...



Dateiname:

Dateityp: SPI Simulyzer Project files (*.spf)

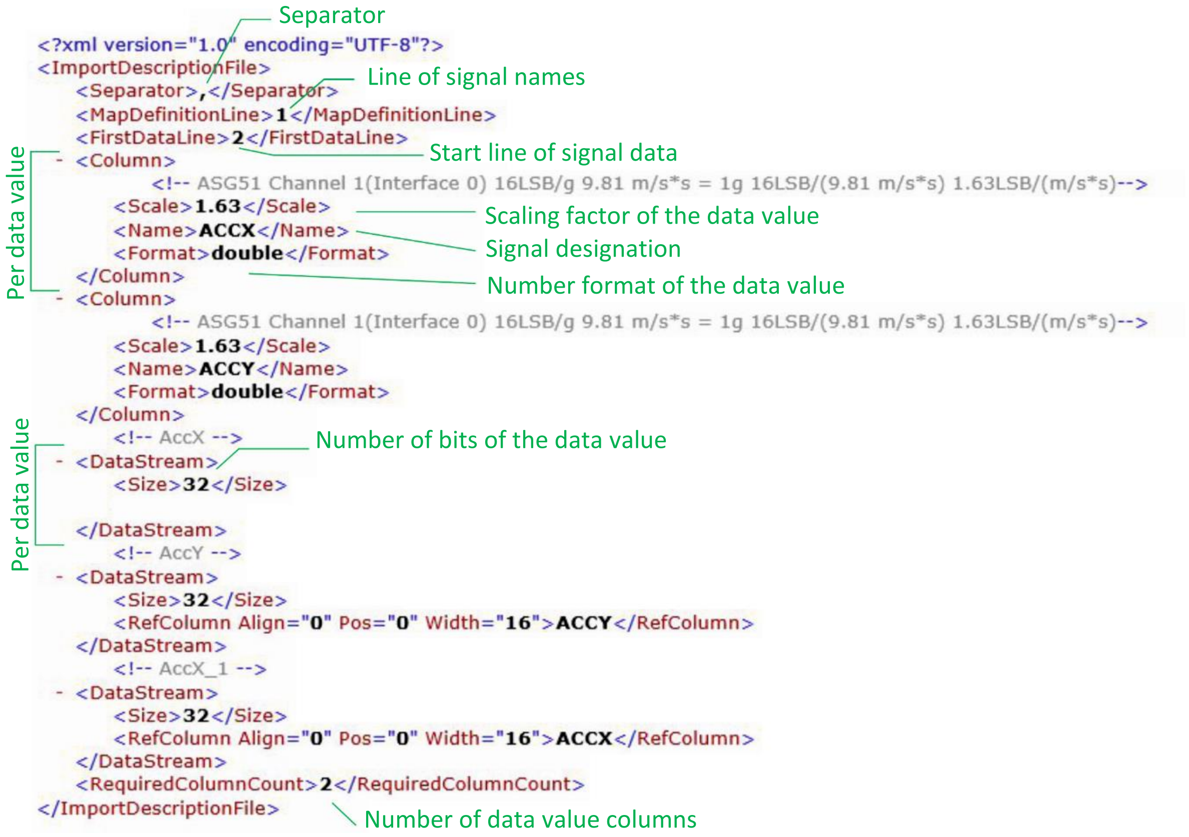
Signal definition

according to the number of signals of the SPI sensor and the signal names defined in the real-time data.csv.
Each signal track is identified by

- the unique, defined signal name and a
- 2-digit signal index.
  1.index = channel 0 or 1 (interface 1 or 2)
  2.Index = Chipselect line of the signal.

SPI Decoderfile

The SPI Decoderfile references and defines the SPI signals from the simulation data file.

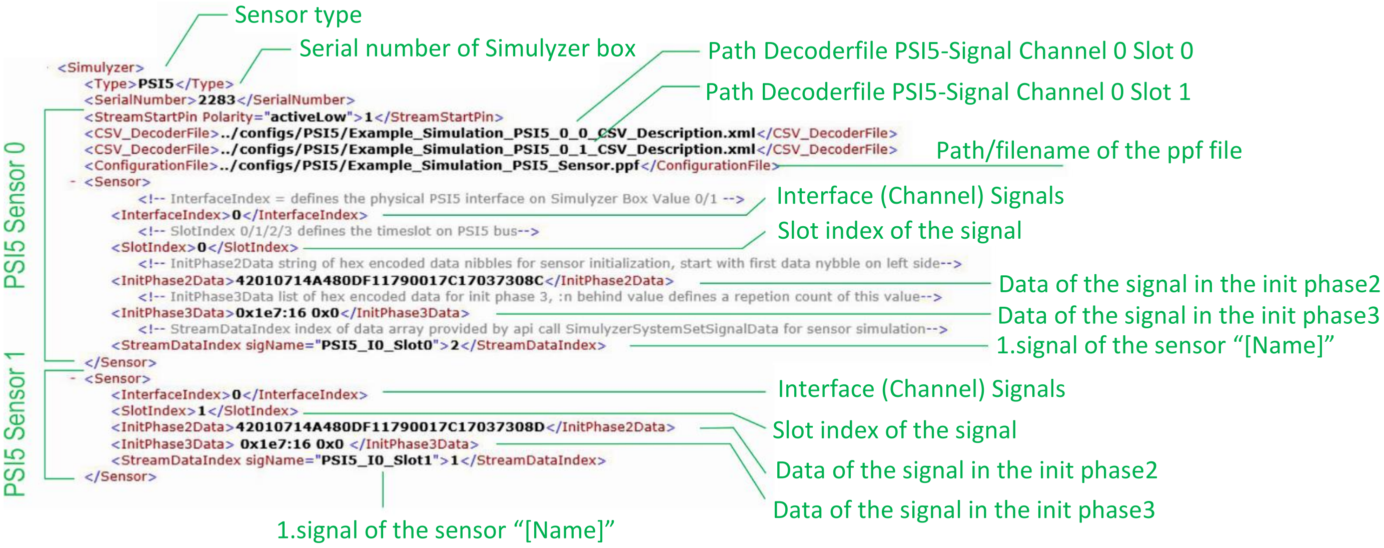


Separator
Line of signal names
Start line of signal data
Scaling factor of the data value
Signal designation
Number format of the data value
Per data value
Number of bits of the data value
Per data value
Number of data value columns

```
<?xml version="1.0" encoding="UTF-8"?>
<ImportDescriptionFile>
    <Separator>,</Separator>
    <MapDefinitionLine>1</MapDefinitionLine>
    <FirstDataLine>2</FirstDataLine>
  - <Column>
        <!-- ASG51 Channel 1(Interface 0) 16LSB/g 9.81 m/s*s = 1g 16LSB/(9.81 m/s*s) 1.63LSB/(m/s*s)-->
      <Scale>1.63</Scale>
      <Name>ACCX</Name>
      <Format>double</Format>
    </Column>
  - <Column>
        <!-- ASG51 Channel 1(Interface 0) 16LSB/g 9.81 m/s*s = 1g 16LSB/(9.81 m/s*s) 1.63LSB/(m/s*s)-->
      <Scale>1.63</Scale>
      <Name>ACCY</Name>
      <Format>double</Format>
    </Column>
      <!-- AccX -->
  - <DataStream>
      <Size>32</Size>

    </DataStream>
      <!-- AccY -->
  - <DataStream>
      <Size>32</Size>
      <RefColumn Align="0" Pos="0" Width="16">ACCY</RefColumn>
    </DataStream>
      <!-- AccX_1 -->
  - <DataStream>
      <Size>32</Size>
      <RefColumn Align="0" Pos="0" Width="16">ACCX</RefColumn>
    </DataStream>
    <RequiredColumnCount>2</RequiredColumnCount>
</ImportDescriptionFile>
```

Definition of the PSI5 sensors

For each PSI5 sensor signal, a definition must be made according to the following scheme.
(Example data see "2.3 System configuration file for CSV data" page 11)



Sensor type
Serial number of Simulyzer box
Path Decoderfile PSI5-Signal Channel 0 Slot 0
Path Decoderfile PSI5-Signal Channel 0 Slot 1
Path/filename of the ppf file
Interface (Channel) Signals
Slot index of the signal
Data of the signal in the init phase2
Data of the signal in the init phase3
1.signal of the sensor "[Name]"
PSI5 Sensor 0
PSI5 Sensor 1
Interface (Channel) Signals
Slot index of the signal
Data of the signal in the init phase2
Data of the signal in the init phase3
1.signal of the sensor "[Name]"

```
- <Simulyzer>
    <Type>PSI5</Type>
    <SerialNumber>2283</SerialNumber>
    <StreamStartPin Polarity="activeLow">1</StreamStartPin>
    <CSV_DecoderFile>../configs/PSI5/Example_Simulation_PSI5_0_0_CSV_Description.xml</CSV_DecoderFile>
    <CSV_DecoderFile>../configs/PSI5/Example_Simulation_PSI5_0_1_CSV_Description.xml</CSV_DecoderFile>
    <ConfigurationFile>../configs/PSI5/Example_Simulation_PSI5_Sensor.ppf</ConfigurationFile>
  - <Sensor>
        <!-- InterfaceIndex = defines the physical PSI5 interface on Simulyzer Box Value 0/1 -->
      <InterfaceIndex>0</InterfaceIndex>
        <!-- SlotIndex 0/1/2/3 defines the timeslot on PSI5 bus-->
      <SlotIndex>0</SlotIndex>
        <!-- InitPhase2Data string of hex encoded data nibbles for sensor initialization, start with first data nybble on left side-->
      <InitPhase2Data>42010714A480DF11790017C17037308C</InitPhase2Data>
        <!-- InitPhase3Data list of hex encoded data for init phase 3, :n behind value defines a repetion count of this value-->
      <InitPhase3Data>0x1e7:16 0x0</InitPhase3Data>
        <!-- StreamDataIndex index of data array provided by api call SimulyzerSystemSetSignalData for sensor simulation-->
      <StreamDataIndex sigName="PSI5_I0_Slot0">2</StreamDataIndex>
    </Sensor>
  - <Sensor>
      <InterfaceIndex>0</InterfaceIndex>
      <SlotIndex>1</SlotIndex>
      <InitPhase2Data>42010714A480DF11790017C17037308D</InitPhase2Data>
      <InitPhase3Data> 0x1e7:16 0x0 </InitPhase3Data>
      <StreamDataIndex sigName="PSI5_I0_Slot1">1</StreamDataIndex>
    </Sensor>
```

Sensor type: Type of sensor

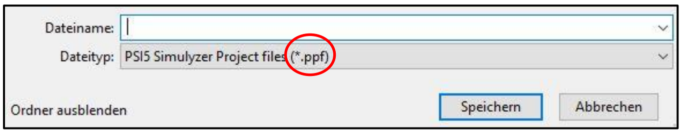
Serial number of the PSI5-Simulyzer box.

StreamstartPinPolarity-Definition: Definition of the start pulse for data transmission which is used by the Simulyzer box as master for all connected Simulyzer boxes. In the example, a pulse is sent to all Simulyzer boxes at pin 1 and data transmission is started with its falling edge.

If this stream start pulse comes from the system or automatically, no definition is required.

Path/filename of the ppf-file which can be generated from the Simulyzer software and which describes the PSI5 sensor. (see PSI5-Simulyzer webhelp)

Creation of a ppf-file from the Simulyzer-Software: Menu File - Command Save as...



Dateiname: |

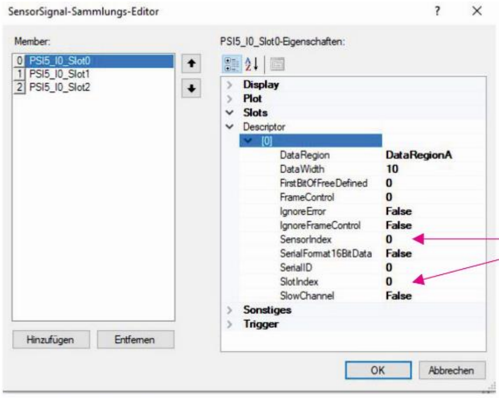
Dateityp: PSI5 Simulyzer Project files (*.ppf)

Interface: Channel on which the PSI5 sensor signal is sent (channel 0 or 1).

Slot index: Slot index in which the PSI5 sensor signal is sent. The assignment of the time slot in which the data signal is sent is done in the PSI5 simulation system and must be entered accordingly.

PSI5-Simulyzer-Software
Signal-Collection-Editor

Signal 1 used in simulation
Signal 2 used in simulation
Signal 3 not used in simulation



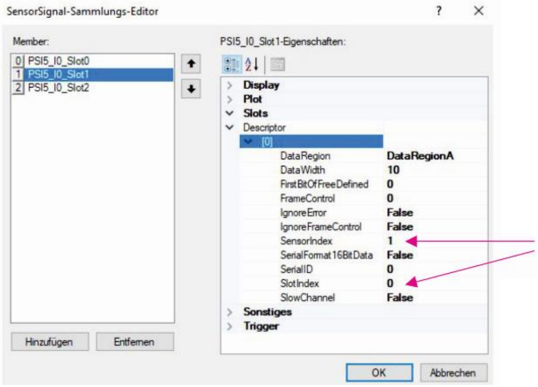
SensorSignal-Sammlungs-Editor

Member:
0 PSI5_I0_Slot0
1 PSI5_I0_Slot1
2 PSI5_I0_Slot2

PSI5_I0_Slot0-Eigenschaften:

DataRegion	DataRegionA
DataWidth	10
FirstBitOfFreeDefined	0
FrameControl	0
IgnoreError	False
IgnoreFrameControl	False
SensorIndex	0
SerialFormat16BitData	False
SerialID	0
SlotIndex	0
SlowChannel	False

1st signal comes from
Sensor 0 in timeslot 0



SensorSignal-Sammlungs-Editor

Member:
0 PSI5_I0_Slot0
1 PSI5_I0_Slot1
2 PSI5_I0_Slot2

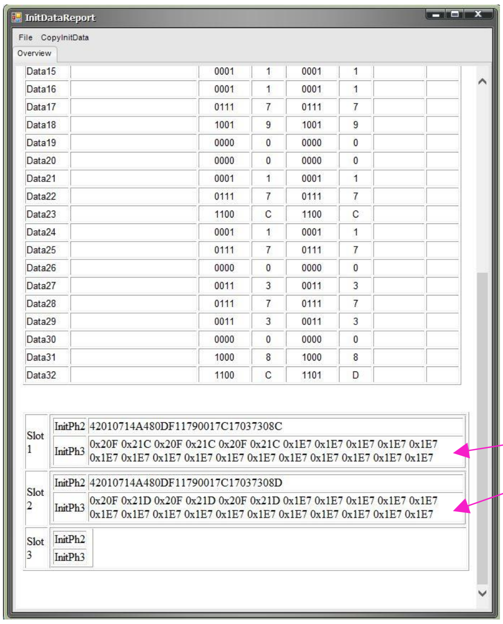
PSI5_I0_Slot1-Eigenschaften:

DataRegion	DataRegionA
DataWidth	10
FirstBitOfFreeDefined	0
FrameControl	0
IgnoreError	False
IgnoreFrameControl	False
SensorIndex	1
SerialFormat16BitData	False
SerialID	0
SlotIndex	0
SlowChannel	False

2nd signal comes from
Sensor 1 in timeslot 0

According to this assignment, the definition must be made in the configuration file.

Initphase 2/3: Hexadecimal data sent by the sensor during the initialization phase.
The data can be copied and pasted in the PSI5 software via the Init Data Report.



Data15		0001	1	0001	1		
Data16		0001	1	0001	1		
Data17		0111	7	0111	7		
Data18		1001	9	1001	9		
Data19		0000	0	0000	0		
Data20		0000	0	0000	0		
Data21		0001	1	0001	1		
Data22		0111	7	0111	7		
Data23		1100	C	1100	C		
Data24		0001	1	0001	1		
Data25		0111	7	0111	7		
Data26		0000	0	0000	0		
Data27		0011	3	0011	3		
Data28		0111	7	0111	7		
Data29		0011	3	0011	3		
Data30		0000	0	0000	0		
Data31		1000	8	1000	8		
Data32		1100	C	1101	D		

Slot 1	InitPh2	42010714A480DF11790017C17037308C
	InitPh3	0x20F 0x21C 0x20F 0x21C 0x20F 0x21C 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7
Slot 2	InitPh2	42010714A480DF11790017C17037308D
	InitPh3	0x20F 0x21D 0x20F 0x21D 0x20F 0x21D 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7 0x1E7
Slot 3	InitPh2	
	InitPh3	

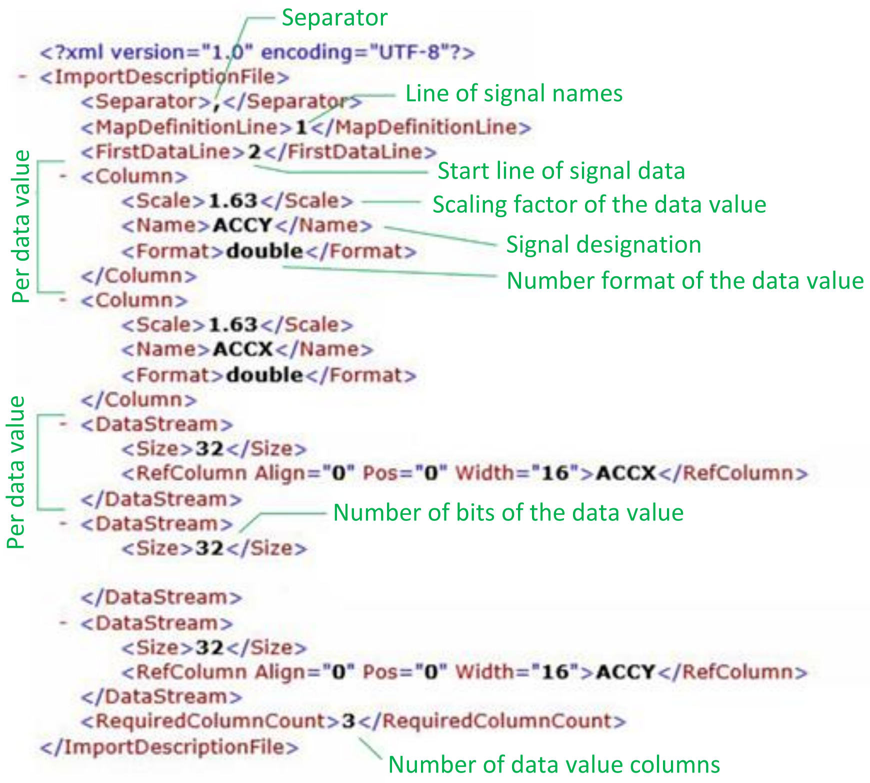
Hexadecimal values of the init data for Slot1 and Slot2

For transfer with copy/paste

Streamdataindex(signal data definition)
according to the number of signals and the signal names defined in the CSV.
Each signal track is identified by

- the unique defined signal name and a
- 1-digit signal index

PSI5-Decoderfile
The PSI5 decoder file references and defines the PSI5 signals from the simulation data file.



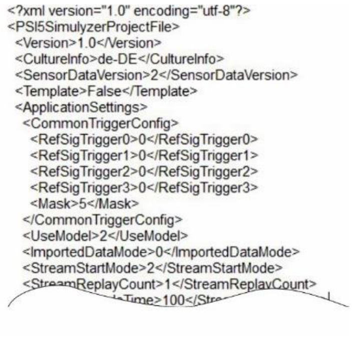
```
  <?xml version="1.0" encoding="UTF-8"?>
- <ImportDescriptionFile>
     <Separator>,</Separator>
     <MapDefinitionLine>1</MapDefinitionLine>
     <FirstDataLine>2</FirstDataLine>
   - <Column>
        <Scale>1.63</Scale>
        <Name>ACCY</Name>
        <Format>double</Format>
     </Column>
   - <Column>
        <Scale>1.63</Scale>
        <Name>ACCX</Name>
        <Format>double</Format>
     </Column>
   - <DataStream>
        <Size>32</Size>
        <RefColumn Align="0" Pos="0" Width="16">ACCX</RefColumn>
     </DataStream>
   - <DataStream>
        <Size>32</Size>

     </DataStream>
   - <DataStream>
        <Size>32</Size>
        <RefColumn Align="0" Pos="0" Width="16">ACCY</RefColumn>
     </DataStream>
     <RequiredColumnCount>3</RequiredColumnCount>
  </ImportDescriptionFile>
```

Separator
Line of signal names
Start line of signal data
Scaling factor of the data value
Signal designation
Number format of the data value
Per data value
Per data value
Number of bits of the data value
Number of data value columns

2.3.2. PPF file

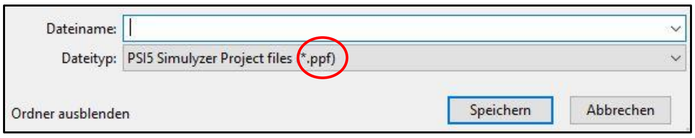
The ppf file contains all characteristic data (type, timeslot definition, data length, data count and many more).
The file can be generated by the PSI5-Simulyzer software.
There must be one file per PSI5-Simulyzer box used.

Example of a ppf file:



```
<?xml version="1.0" encoding="utf-8"?>
<PSI5SimulyzerProjectFile>
 <Version>1.0</Version>
 <CultureInfo>de-DE</CultureInfo>
 <SensorDataVersion>2</SensorDataVersion>
 <Template>False</Template>
 <ApplicationSettings>
  <CommonTriggerConfig>
   <RefSigTrigger0>0</RefSigTrigger0>
   <RefSigTrigger1>0</RefSigTrigger1>
   <RefSigTrigger2>0</RefSigTrigger2>
   <RefSigTrigger3>0</RefSigTrigger3>
   <Mask>5</Mask>
  </CommonTriggerConfig>
  <UseModel>2</UseModel>
  <ImportedDataMode>0</ImportedDataMode>
  <StreamStartMode>2</StreamStartMode>
  <StreamReplayCount>1</StreamReplayCount>
         Time>100</Str
```

Creating a ppf-file from the Simulyzer software:
Menu File - Command Save as...



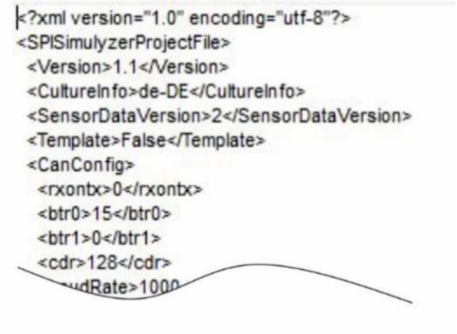
Dateiname:

Dateityp: PSI5 Simulyzer Project files (*.ppf)

2.3.3. SPF file

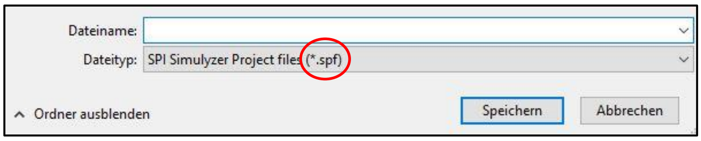
The spf file contains all characteristic PSI5 bus data (baud rate, CS allocation, data count and many more).
The file can be generated by the PSI5-Simulyzer software.
There must be one file per SPI-Simulyzer box used.

Example of spf file:



```
<?xml version="1.0" encoding="utf-8"?>
<SPISimulyzerProjectFile>
 <Version>1.1</Version>
 <CultureInfo>de-DE</CultureInfo>
 <SensorDataVersion>2</SensorDataVersion>
 <Template>False</Template>
 <CanConfig>
  <rxontx>0</rxontx>
  <btr0>15</btr0>
  <btr1>0</btr1>
  <cdr>128</cdr>
    udRate>1000
```

Creation of an spf-file from the Simulyzer software: Menu File - Command Save as...



Dateiname:

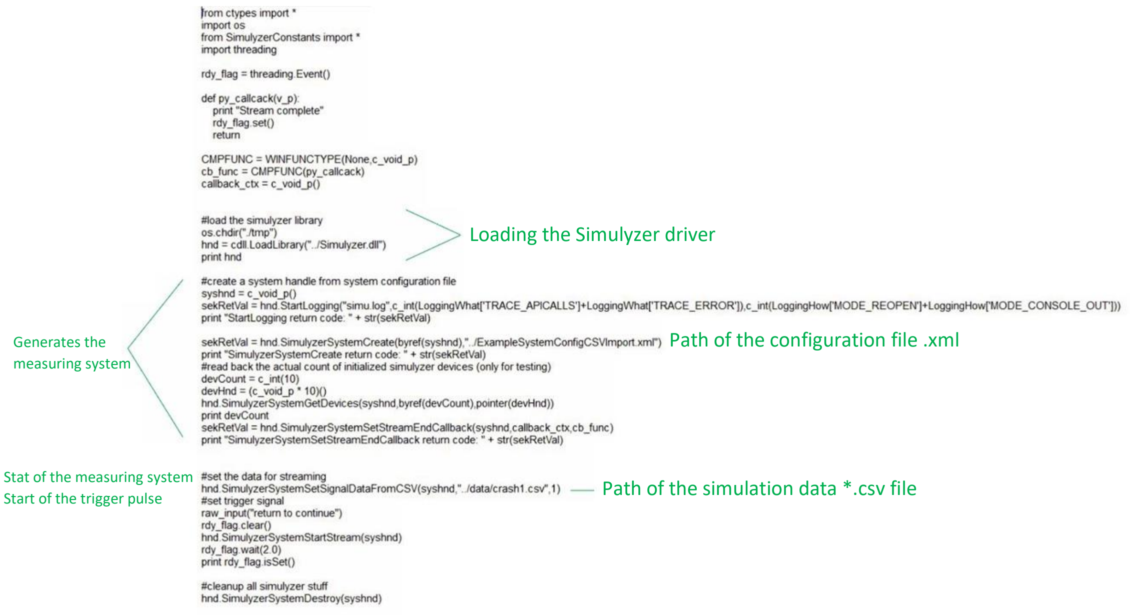
Dateityp: SPI Simulyzer Project files (*.spf)

2.3.4. Program for process flow control

The program controls the entire HIL test system and regulates the complete measurement process.

Possible programming languages: API: ANSI-C, Python, dotNET
Knowledge of the corresponding programming language is required.
In principle, the API contains the following structure (example Python)

1. loading of the Simulyzer driver
2. generation of the measuring system and linkage with the system configuration.xml
3. login - check of proper system configuration and readiness of individual components
4. start command of the measuring system/start of the trigger pulse
5. stop of the measuring system - stop command



```
from ctypes import *
import os
from SimulyzerConstants import *
import threading

rdy_flag = threading.Event()

def py_callcack(v_p):
   print "Stream complete"
   rdy_flag.set()
   return

CMPFUNC = WINFUNCTYPE(None,c_void_p)
cb_func = CMPFUNC(py_callcack)
callback_ctx = c_void_p()


#load the simulyzer library
os.chdir("./tmp")
hnd = cdll.LoadLibrary("../Simulyzer.dll")
print hnd

#create a system handle from system configuration file
syshnd = c_void_p()
sekRetVal = hnd.StartLogging("simu.log",c_int(LoggingWhat['TRACE_APICALLS']+LoggingWhat['TRACE_ERROR']),c_int(LoggingHow['MODE_REOPEN']+LoggingHow['MODE_CONSOLE_OUT']))
print "StartLogging return code: " + str(sekRetVal)

sekRetVal = hnd.SimulyzerSystemCreate(byref(syshnd),"../ExampleSystemConfigCSVImport.xml")
print "SimulyzerSystemCreate return code: " + str(sekRetVal)
#read back the actual count of initialized simulyzer devices (only for testing)
devCount = c_int(10)
devHnd = (c_void_p * 10)()
hnd.SimulyzerSystemGetDevices(syshnd,byref(devCount),pointer(devHnd))
print devCount
sekRetVal = hnd.SimulyzerSystemSetStreamEndCallback(syshnd,callback_ctx,cb_func)
print "SimulyzerSystemSetStreamEndCallback return code: " + str(sekRetVal)


#set the data for streaming
hnd.SimulyzerSystemSetSignalDataFromCSV(syshnd,"../data/crash1.csv",1)
#set trigger signal
raw_input("return to continue")
rdy_flag.clear()
hnd.SimulyzerSystemStartStream(syshnd)
rdy_flag.wait(2.0)
print rdy_flag.isSet()

#cleanup all simulyzer stuff
hnd.SimulyzerSystemDestroy(syshnd)
```

Loading the Simulyzer driver

Generates the measuring system

Path of the configuration file .xml

Stat of the measuring system
Start of the trigger pulse

Path of the simulation data *.csv file

The individual expansion is up to the user and can be carried out without restriction in view of the system processes.

Further sources of information and tutorials

Seskion GmbH
Karlsruher Straße 11/1
D-70771 Leinfelden-Echterdingen
Telefon: +49 (711) 990 58 14
Fax: +49 (711) 990 58 27
E-Mail: info@seskion.de
URL: http://www.seskion.de