SAR Measurements with cDASY6

APPLICATION NOTE

Measurement Procedure for Time Averaged SAR with cDASY6

s p e a g

How to Perform Time Averaged SAR Measurements with cDASY6

1 Introduction

Time Averaged SAR (specific absorption rate) measurements apply to devices which can control the time averaged transmitted power in real—time and therefore can control the averaged SAR over the period defined in the applicable standards. This application note shows how Time Averaged SAR measurements can be performed with cDASY6.

2 Measurement Setup

In cDASY6, a Time Averaged SAR assessment can be performed by following the procedure:

- In Project Setup / Device Settings view, specify the device dimensions.
- In Project Setup / Test Conditions view, specify the Phantom Section, Test Distance, DUT position and click on "Add Test Condition to Project".
- In Project Setup / Communication Systems view, select the communication system and test channel to be measured.
- In Project Overview view, click on the Communication System Channel line. The Time Averaged SAR settings appear at the bottom of the view. Time Averaged Scans can be anchored to the maximum location of any 2D (Fast Area, Area) or 3D (Fast Volume, Zoom) scans. The duration of the scan is specified in the "Scan Duration" field. The default value is 360 s.

Figure 1.1 shows the settings to enabled a Time Averaged Scan measurement after a Zoom Scan with a duration of 360 s.

3 Measurement Sequence

The measurement is started by clicking on the "Start" button in the main toolbar. Using the setup shown on Figure 1.1, a "Fast Scan", an "Area Scan" and a "Zoom Scan" will be performed. At the end of the Zoom Scan, the probe will be moved to interpolated maximum and a reference SAR measurement is performed. A pop-up window "Performing time averaged assessment, please enable the power monitoring feature on the device under test" will appear. Click on "OK" once the feature has been enabled. The SAR readings will be recorded during the scan duration specified in "Scan Duration". The Time Averaged Scan can be performed multiple times without repeating the Zoom Scan.

4 Post Processing

The T_x factor, defined in the IEC/IEEE 62209-1528, is calculated as:

$$T_x = \frac{\text{average of SAR readings}}{\text{reference SAR value}} \tag{1.1}$$

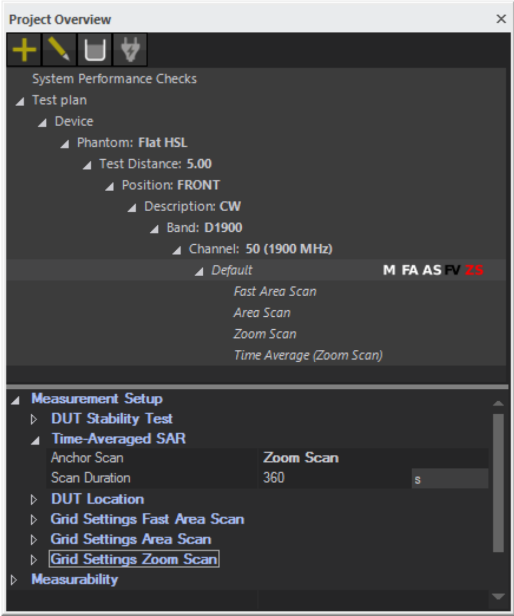


Figure 1.1: Project Overview View Showing Time Averaged SAR Settings

Once the measurement is completed, the 1g / 10g SAR displayed in the Project Overview on the Time Averaged SAR line are defined as:

$$SAR_{1g}(\text{Time Averaged}) = T_x \cdot SAR_{1g}(\text{Area / Zoom Scan}) \tag{1.2}$$

$$SAR_{10g}(\text{Time Averaged}) = T_x \cdot SAR_{10g}(\text{Area / Zoom Scan}) \tag{1.3}$$

Figure 1.2 shows the results of a Time Averaged SAR scan based on a Zoom Scan. The T_x factor can be visualized at the bottom of the Project Overview window. In this specific case, $T_x = 0.5811$. The SAR results are calculated using Equations 1.4 and 1.5:

$$SAR_{1g}(\text{Time Averaged}) = 0.5811 * 1.97 = 1.14 \tag{1.4}$$

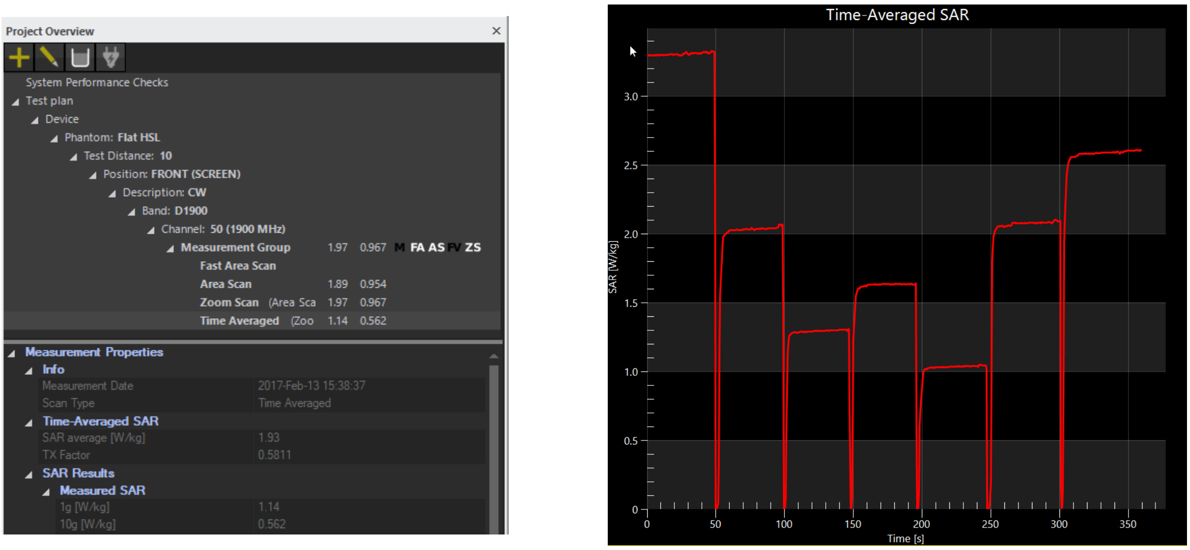
$$SAR_{10g}(\text{Time Averaged}) = 0.5811 * 0.967 = 0.562 \tag{1.5}$$

The SAR readings recorded during the Time Averaged SAR scan can be visualized by clicking on the icon on the "Time Averaged" line (Figure 1.2). More information about SAR compliance of DUTs that can monitor the transmitted power over time is available in Section 8.

5 Validation Procedure

A user may want to validate the procedure. Different procedures are possible, the most simple one is described below:

- Setup a CW measurement at the desired frequency / phantom section that contains a Fast Scan, an Area Scan and a Zoom Scan.
- In "Time-Averaged SAR" settings select "Perform After Zoom Scan" and set "Scan Duration" to 360 s.
- Start the measurement.
- The software will prompt the user to enable the power monitoring feature. Let the CW signal on for 180 s. Then switch it off.
- The measurement will be completed after 360 s. The expected TX factor is 0.5.



(a) T_x factor and SAR Results

(b) SAR Readings Visualization

Figure 1.2: Time Averaged SAR Scan Results

6 Additional Uncertainty for Time Averaged SAR

The additional uncertainty component introduced by the Time Averaged SAR feature, i.e., the uncertainty of determining the T_x factor, is determined as following:

Probe Linearity & Modulation Response Uncertainty: The probe linearity contribution to the T_x factor assessment is identical to the one used for 1g / 10g spatial peak SAR. This uncertainty is already part of the total budget and does not need to be counted twice. Therefore the weight is set to "0".

Probe Low Pass Filter Uncertainty: The low pass filter of SPEAG's SAR probes has edge rates between 0 ms and 5 ms. An error can be introduced when the power monitoring algorithm changes state. It can be expressed analytically as:

$$u_c[\%] = \frac{100}{T} \cdot \left(\int_0^{+\infty} e^{-\frac{x}{5}} dx - 1 \right) \qquad (1.6)$$

with:
T the elapsed time between the two state changes of the power monitoring feature in ms.
τ being the edge rate of the low pass filter filter Assuming that the duration of a state is at least 500 ms, the uncertainty is $u_c \leq 1\%$.

Error Description	Uncert. value	Prob. Dist.	Div.	(c_i)	Std. Unc.	(v_i) v_{eff}
Probe Linearity	±4.7 %	R	$\sqrt{3}$	0	±0 %	∞
Modulation Response	±2.4 %	R	$\sqrt{3}$	0	±0 %	∞
Response Time	±1.0 %	N	1	1	±1.0 %	∞
Combined Std. Uncertainty					±1.0 %	

Table 1.1: Worst-Case uncertainty budget for T_x factor assessment

7 Total Uncertainty for Assessment of Transmitter Equipped with Integrated Power Control

The total uncertainty is computed by the root-sum-square of the standard SAR assessment uncertainty and the additional Uncertainty for Time Averaged SAR.

8 SAR Compliance Testing of DUTs with Power Monitoring Algorithm

Power monitoring algorithm controls the trasmitted power by the device over a defined time interval. They might be used to improve the user experience with smarter network connection while compliance with SAR averaged over a time window remains within the defined limits.
This section proposes a measurement procedure for DUTs that can control the time averaged transmitted power in real-time.

8.1 Measurement Procedure Validity

The purpose of the measurement procedure described in this section is provide to the end-user an insight into the possibilties offered by *cDASY6* for compliance of DUTs that can control the time averaged transmitted power in real-time. The DUT used here is a simple dipole operating at a single frequency and do not have the complexity of modern devices (band handovers...). For compliance testing, the measurement procedures described in the applicable SAR standards should be used.

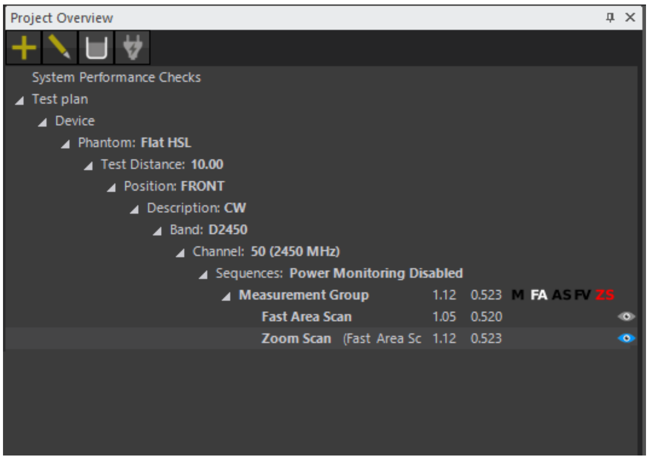
8.2 psSAR1g/10g Assessment at Compliance Power

The first step consists of measuring the psSAR1g/10g at the maximum allowed power P_{comp} when the power monitoring algorithm is switched off. In our example, we consider a limit of $1.6\,W.kg^{-1}$ for psSAR1g.
The DUT used for this purpose is a signal generator which feeds a 2450 MHz dipole. The RF level is set to 17.0 dBm. The dipole is placed below the flat section of a TWIN SAM phantom. The separation distance dipole - liquid is 10 mm.
The project defined in cDASY6 contains two scans: a Fast Area Scan is first used to find the SAR maximum location, then a Zoom Scan is performed to assess the psSAR1g/10g. The psSAR1g resulting from the Zoom Scan is $1.12\,W.kg^{-1}$, well within the defined limit.
The measurement results and SAR distribution are shown on Figure 1.3.

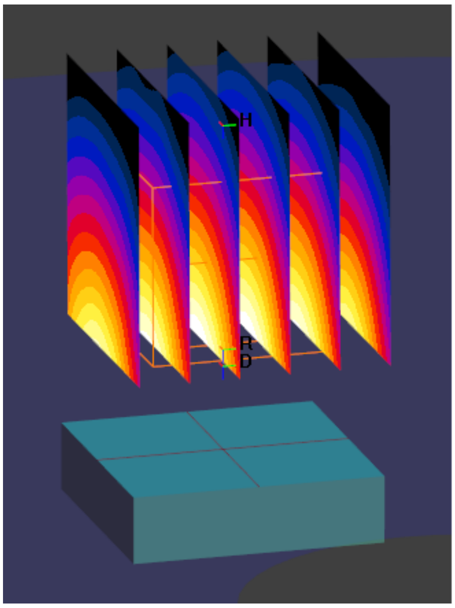
8.3 Implementation of Power Monitoring Algorithm

A power monitoring algorithm is added to the DUT defined in Section 8.2 to enhance network performance. Transmissions at power levels $> P_{comp}$ are allowed for short periods of time but the algorithm will ensure that the SAR averaged over a time window remains within the defined limits.
The power monitoring algorithm developed for this purpose is very simple. The state machine is described in Figure 1.4. The different power quantities are defined as:

- P_{comp} is the maximum power transmitted by the DUT with power monitoring algorithm off to fulfill SAR compliance requirements
- $P_{request}$ is the power level that the network requests the DUT to transmit at
- P_{max} is the maximum transmission power allowed by the power monitoring algorithm
- $P_{transmit}$ is the power the DUT is actually transmitting at. $P_{transmit} < min(P_{request}, P_{max})$ in any circumstances
- P_{safety} is the power level at which the DUT transmit to ensure SAR compliance. $P_{safety} < P_{comp}$ in any circumstances



(a) Project Overview showing psSAR results



(b) SAR distribution in 3d view

Figure 1.3: Visualization of the Measurement Results for a Power of 17.0 dBm.

In our example, P_{max} is set to 19.0 dBm, or 2 dB above P_{comp} defined in Section 8.2. The compliance is ensured by transmitting at a reduced power P_{safety} of 15.0 dBm after the DUT has been emitting at high power for some time. The radio link is therefore maintained and the user has still network access. psSAR1g values will be averaged over a time window of 100 s.
The algorithm has been implemented in Python. The source code is available in Appendix A.

8.4 Power Monitoring Algorithm Validation

The algorithm is validated with two different test sequences:

- $P_{request}$ is constant and larger than P_{max} (Sequence 1)
- $P_{request}$ has random values between 17.0 dBm and 20.0 dBm (Sequence 2)

8.4.1 Numerical Validation

The algorithm has been validated for the two test sequences described previously. The two test sequences are fed to the algorithm which provides back $P_{transmit}$. The source code of the Python script used for this purpose is available in Appendix B.
Figure 1.5 shows the variations of $P_{transmit}$, $P_{requested}$ and P_{avg} over time.The plots have been generated with the graphics library available in cDASY6 (cf. Appendix C). If Sequence 1 is applied, $P_{transmit}$ has two different levels P_{max} and P_{safety} since $P_{requested}$ is always above P_{max}. After a settling time, P_{avg} is constant and equal to P_{comp}. With Sequence 2, $P_{request}$ varies between 17.0 dBm and 20.0 dBm. As expected, $P_{transmit}$ varies between P_{safety} and P_{max}. P_{avg} never exceeds P_{comp}.

8.4.2 Validation with SAR measurements

The power monitoring algorithm is then validated with SAR measurements. The measurement setup described in Section 8.2 is used. In addition, the RF level of the signal generator is set to the output of the power monitoring algorithm $P_{transmit}$ via GPIB commands.
The project defined in cDASY6 consists of a Fast Area Scan, a Zoom Scan and a Time Averaged Scan. The Fast Area and Zoom Scans are performed with the algorithm switched off. The psSAR1g resulting from the

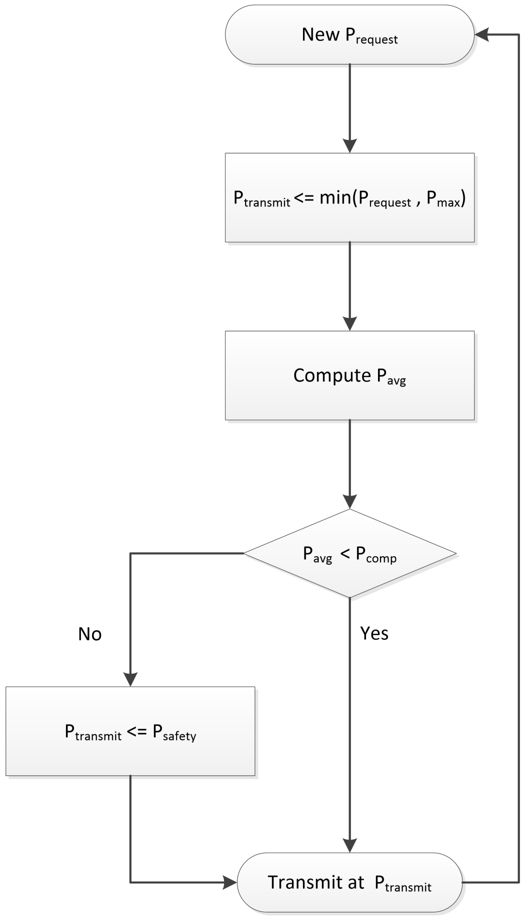


Figure 1.4: State Machine of the Power Monitoring Algorithm

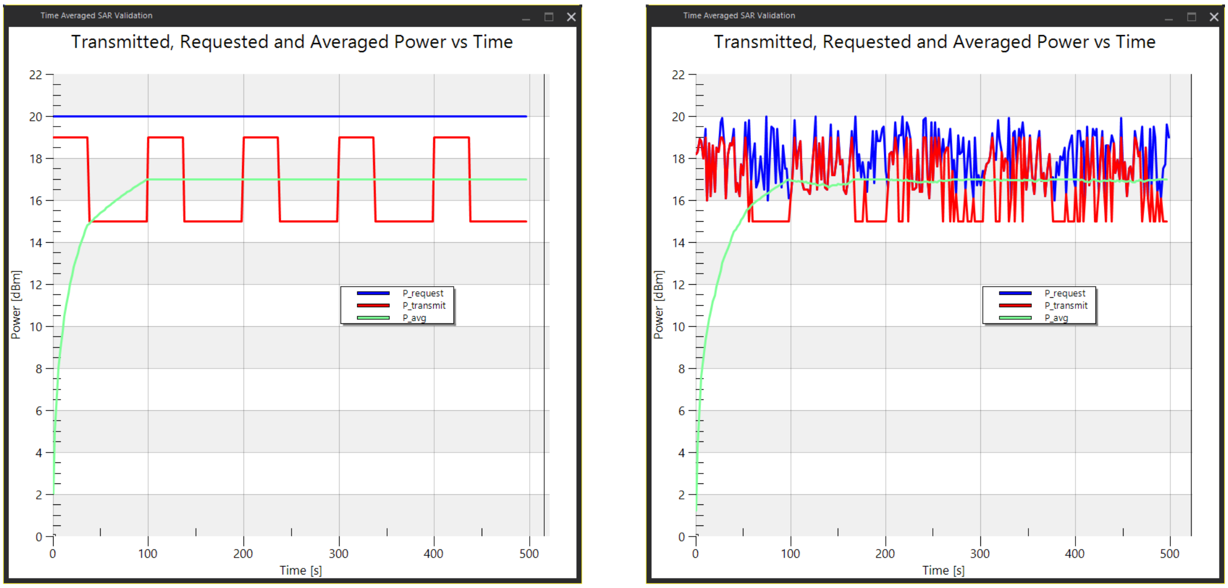
Zoom Scan is $1.12 W.kg^{-1}$. The deviation with the reference measurement performed in Section 8.2 is less than 1% which shows the excellent repeatability of the system.
The power monitoring algorithm is switched on and the two test sequences are successively applied to the power monitoring algorithm. For each test sequence, a separate Time Averaged Scan is performed.
The results are shown in Figure 1.6. The displayed SAR levels have been normalized to equivalent psSAR1g values with:

$$SAR_{displayed}(t) = \frac{SAR_{measured}(t)}{SAR_{max\,algo\,off}} * psSAR1g_{comp} \tag{1.7}$$

$SAR_{measured}(t)$ is the point measured SAR at the time t
$SAR_{algo\,off}$ is the point SAR measured with the power monitoring algorithm disabled
$psSAR1g_{comp}$ is the peak spatial averaged SAR over 1g of tissue measured with the power monitoring algorithm disabled
The measured SAR levels have the same shape than $P_{transmit}$ in Figure 1.5. SAR_{avg} never exceeds SAR_{comp}.
For Sequence 1, a TX factor of 1 is expected since $P_{requested}$ is always higher than P_{max}. The measured TX factor is 0.974, less than 3% off the target.
The algorithm behaves as expected and the compliance of the DUT has been demonstrated.

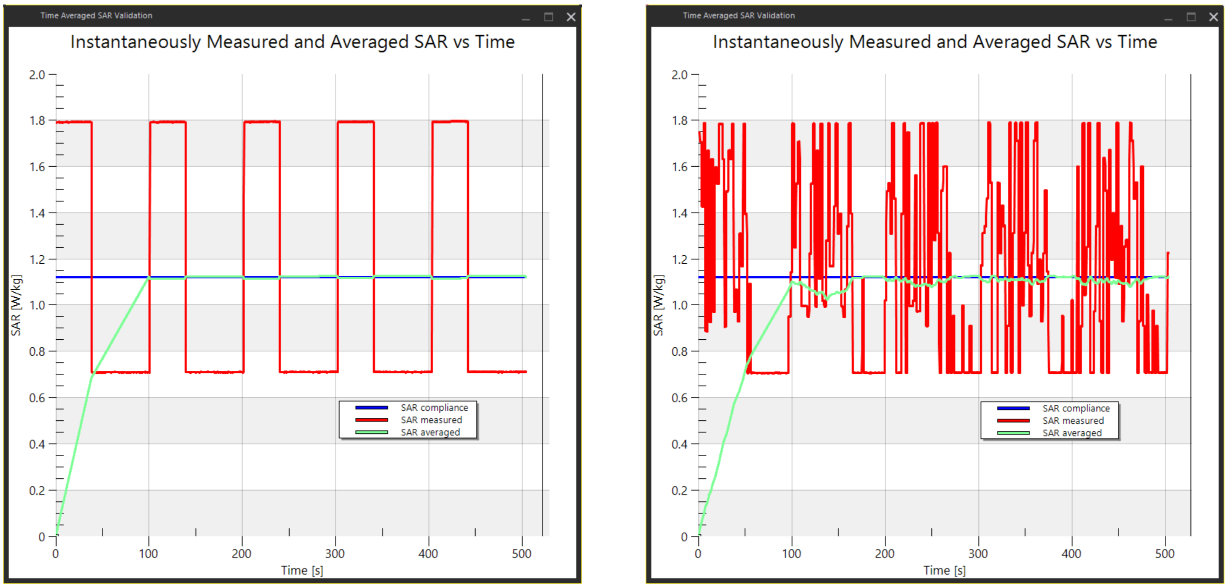
9 Conclusion

This new cDASY6 feature supports the user to measure the Time Averaged SAR for devices that monitor and control the time averaged transmitted power in real—time over the period defined in the applicable standards. It is a very flexible and easy to apply implementation.



(a) Sequence 1 (b) Sequence 2

Figure 1.5: Numerical Validation of the Power Monitoring Algorithm



(a) Sequence 1 (b) Sequence 2

Figure 1.6: Power Monitoring Algorithm Validation with SAR Measurements

Appendices

A Python Implementation of Power Monitoring Algorithm

```python
# -*- coding: utf-8 -*-

import math
import numpy

class PowerMonitoringAlgorithm:

    def __init__(
        self,
        p_comp, # compliance power
        p_max, # maximum power
        p_safety, # safety power to ensure compliance
        t_avg, # SAR averaging window
        t_refresh # refresh rate of power setting on the DUT
    ):
        # Convert all power levels in linear
        self._p_comp_lin = 10**(p_comp/10)
        self._p_max_lin = 10**(p_max/10)
        self._p_safety_lin = 10**(p_safety/10)
        self._t_avg = t_avg
        self._t_refresh = t_refresh
        # Store the previous relevant power settings
        self._p_memory = int(self._t_avg / self._t_refresh) * [self._p_safety_lin]
        # Store the current index in self._p_memory
        self._p_index = 0

    # Return the power at which the DUT can transmit based on the power
    # requested by the network to ensure SAR compliance
    # p_request is the power requested by the network at a given time
    def get_p_transmit(self, p_request):
        p_request_lin = 10**(p_request/10)
        p_request_lin = min([self._p_max_lin, p_request_lin])
        p_transmit_lin = 0
        self._p_memory[self._p_index] = p_request_lin
        p_avg = self.get_avg_power()
        if p_avg < self._p_comp_lin:
            p_transmit_lin = p_request_lin
        else:
            p_transmit_lin = self._p_safety_lin
        self._p_memory[self._p_index] = p_transmit_lin
        self._p_index = (self._p_index + 1) % len(self._p_memory)
        return 10*math.log10(p_transmit_lin)

    def get_avg_power(self):
        return sum(self._p_memory) / len(self._p_memory)
```

B Python Test Bench for Numerical Validation of Power Monitoring Algorithm

```python
# -*- coding: utf-8 -*-

import math
import numpy
import random

import PowerMonitoringAlgorithm

# Power levels definition in dBm
p_comp = 17.
p_max = 19.
p_requested = 20.
p_safety = 15.
# Averaging window in s
t_avg = 100.
# Refresh rate of the power setting
t_refresh = 2.

p_algo = PowerMonitoringAlgorithm.PowerMonitoringAlgorithm(
    p_comp,
    p_max,
    p_safety,
    t_avg,
    t_refresh
)

# Trasmitted power in dBm
p_transmit = []
# Averaged power in dBm
p_avg = []

# Test Sequences
seq_length = 250
p_request = [p_requested for i in range(seq_length)] # sequence 1
p_request = [random.randint(160, 210) / 10. for i in range(seq_length)] # sequence 2

def algorithm_test():
    nb_samples_avg = int(t_avg / t_refresh)
    idx = 0
    p_avg_mem = nb_samples_avg * [0]
    for i in range(seq_length):
        p_current = p_algo.get_p_transmit(p_request[i])
        p_transmit.append(p_current)
        p_avg_mem[idx] = 10**(p_current/10)
        idx = (idx + 1) % nb_samples_avg
        p_avg.append(10*math.log10(sum(p_avg_mem) / len(p_avg_mem)))
    return [p_request, p_transmit, p_avg, t_refresh]
```

C Visualization of Power Monitoring Algorithm Outputs

```
# -*- coding: utf-8 -*-

import numpy

from XCore import *
from XCoreUI import *
from XPlotLib import *

import TestBench

[p_request, p_transmit, p_avg, t_refresh] = TestBench.algorithm_test()

# Get MainFrame, create a floating view and set the style
MyFrame = GetApp().Frame
MainView = FloatingView()
MainView.HasCloseButton = True
MainView.Sizeable = True
MainView.ViewSize = Size(800,800)

# Add the MainView to the frame
MainView = MyFrame.AddView("Time Averaged SAR Validation", MainView)

Figure = Figure()
Figure.FigureName="Time Averaged SAR Validation"

# Set the Figure settings
FigureData = Figure.FigureData
FigureSettings = FigureData.FigureSettings
FigureSettings.MainTitle = "Transmitted, Requested and Averaged Power vs Time"
FigureSettings.SubTitle = " "
FigureSettings.ShowCursor = True
FigureSettings.CursorMode = CursorMode.Vertical

# Plot the 3 Powers
Figure.AddPlot(
   numpy.arange(0, (len(p_request)-1)*t_refresh, t_refresh),
   numpy.array(p_request),
   "P_request"
)
Figure.AddPlot(
   numpy.arange(0, (len(p_transmit)-1)*t_refresh, t_refresh),
   numpy.array(p_transmit),
   "P_transmit"
)
Figure.AddPlot(
   numpy.arange(0, (len(p_avg)-1)*t_refresh, t_refresh),
   numpy.array(p_avg),
   "P_avg"
)

# Set the axis parameters
AxisSettings = FigureData.AxisSettings
AxisSettings.RangeModeX = RangeMode.Auto
AxisSettings.RangeModeY = RangeMode.MinMax
AxisSettings.YMin = 0
AxisSettings.YMax = 22
AxisSettings.xLabel = "Time [s]"
AxisSettings.yLabel = "Power [dBm]"
```

```
LegendSettings = FigureData.LegendSettings
LegendSettings.Visible = True


def OnClose():
   print "Pressed close"
   MainView.Lose()

# Connect callback to the signal
MainView.OnClose.Connect(OnClose)

MainView.AddView("Figure", Figure.View)
MainView.Visible = True
```