

5G UE PA R&D TESTING

Automated Solution with High Efficiency

Products:

- ▶ R&S®SMW200A
- ▶ R&S®VSE
- ▶ R&S® NGM202
- ▶ R&S®FSW
- ▶ R&S®NRP8S
- ▶ SignalCraft SC4415

Guoying Fu, Sidney Yu | 1SL365 | Version 0e | 06.2021

<http://www.rohde-schwarz.com/appnote/1SL365>



Contents

1	Overview.....	3
2	Introduction.....	4
3	5G NR Signal Generation and Analysis	5
3.1	5G NR Signal Generation.....	5
3.2	5G NR Signal Analysis	11
4	MIPI Control Interface.....	14
5	Power Servo.....	17
6	Improved PA automated testing	21
6.1	PA Modulation Measurement	21
6.2	PA Spurs Measurement	27
7	Literature	29
8	Ordering Information	30
8.1	R&S®SMW200A	30
8.2	R&S®FSW	30
8.3	R&S®VSE	30
8.4	R&S®NGM202.....	30
8.5	R&S®NRP8S	30

1 Overview

5G New Radio (NR) presents dramatic challenges to the testing of User Equipment (UE) Power Amplifier (PA). Multiple frequency bands, flexible 5G operation mode, multiple PA performance metrics and the MIPI control interface increases the potential PA testing workload significantly. Manually repeating these tests under so many different scenarios can be costly and time consuming. An automated testing solution can help the test engineer to highly improve the testing efficiency.

In this application note, it introduces an example of automated testing of 5G UE PA with R&S signal generator, signal and spectrum analyzer, vector signal explorer software, power meter along with power supply.

The entire application note is organized with following structure:

Chapter 2 outlines the challenges to 5G UE PA testing.

Chapter 3 introduces the setup for 5G signal generation and analysis. For 5G signal generation, a batch program of converting *.CSV files with IQ vectors to ARB waveform files is introduced. For 5G signal analysis, the R&S®VSE software is recommended, which enables the separation of RF signal collection and measurement to improve testing efficiency.

Chapter 4 provides the informative content about the integration of MIPI control interface for the PA ON/OFF state switching and register configuration.

Chapter 5 gives a guide on the implementation of a quick power servo, in which the techniques to speed up the PA power level adjustment are discussed.

Chapter 6 summarizes the automated testing procedure.

2 Introduction

5G NR adopts the similar OFDM technology as LTE, but in order to cover the different application fields of enhanced mobile broadband communication (eMBB), ultra-reliable ultra-low latency communication (URLLC) and massive machine-type communications (mMTC), the radio scheme of 5G NR is required to be much more flexible, for example in the definition of multiple numerologies, multiple frequency bands and Massive MIMO [1].

Nowadays most of the LTE/5G UEs are smart phones which have an advanced operating system providing office and entertainment functions. If the UEs cannot be used through a whole day and have to recharge during the day, the user experience will be impacted significantly. The main power consumption of the UEs can be categorized into the screen display, the processor, the networking and radios, and the data refreshing and storage. The PA performance dominates the overall power consumption of the networking and radios part of the UEs. The optimizing of PA is one of the key enablers of improving the UEs power consumption and battery life.

The key performance indicator of the UE PA may contain Power Added Efficiency (PAE), modulation performance such as Error Vector Magnitude (EVM) and Adjacent Channel Leakage Ratio (ACLR), and Spurs performance. Figure 2-1 illustrates a general setup of the PA testing with signal generator, signal and spectrum analyzer, power sensor and power supply.

The signal generator R&S®SMW200A can generate the standard 5G NR signal in real-time mode or ARB waveform mode. In the ARB waveform mode, the waveforms that contain the 5G NR IQ data can be prepared in advance to save testing time.

The signal and spectrum analyzer R&S®FSW measures the EVM and ACLR performance of the amplified signal after the PA.

A programmable power supply, for example R&S®NGM200 series, does not only provide the power for the PA, but also the accurate measurement of current and dissipate power. Additionally, the power meter can be used for the power servo for the PA testing.



Figure 2-1 General Setup for PA Testing

In latter chapters we will introduce improvement and enhancement to the general testing setup in Figure 2-1, such as the separation of RF signal collection and measurement, the MIPI control interface integration, and the power-servo loop, to meet the requirement of 5G UE PA.

3 5G NR Signal Generation and Analysis

3.1 5G NR Signal Generation

The R&S®SMW200A is the vector signal generator for the most demanding applications, with frequency range up to 44 GHz and IQ modulation bandwidth up to 2 GHz. The baseband section of the SMW200A is fully digital, and it contains the hardware for generating and processing I/Q signals in real-time or generating signals with an arbitrary waveform generator.

Figure 3-1 shows the real-time 5G NR uplink and downlink signal generation option SMW-K144. It supports all waveforms, channel bandwidths, modulation schemes and numerology options in line with 3GPP standard. The intuitive GUI allows configuring these and many other parameters, such as multiple bandwidth part or MIMO precoding, directly on the instrument.

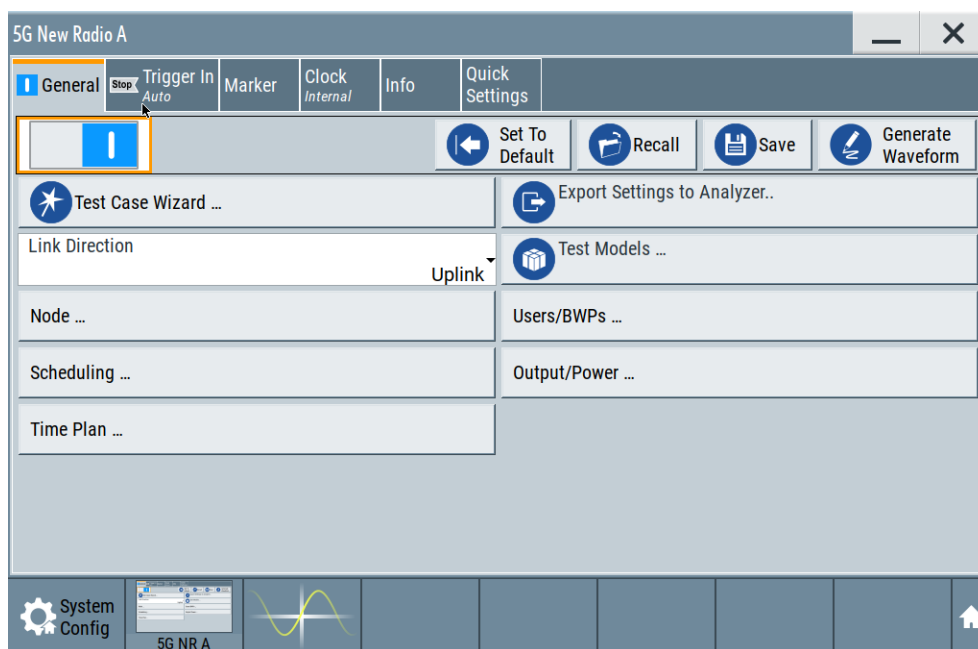


Figure 3-1 SMW-K144 5G NR signal generation option

Alternatively, SMW200A can work as an Arbitrary Waveform Generator as shown in Figure 3-2. The ARB allows the playback and output of an externally calculated modulation signals in the form of waveform files. The waveform files are used as I/Q modulation sources corresponding to different signal standards or different parameters. The waveform files for the 5G FR1 UE PA testing are characterized by the key parameters including Subcarrier Spacing, Channel Bandwidth, OFDM Type, Modulation Schemes and RB Allocation Configuration, as the example shown in Table 3-1. We can see that there can be totally thousands of waveforms with different parameter combinations.

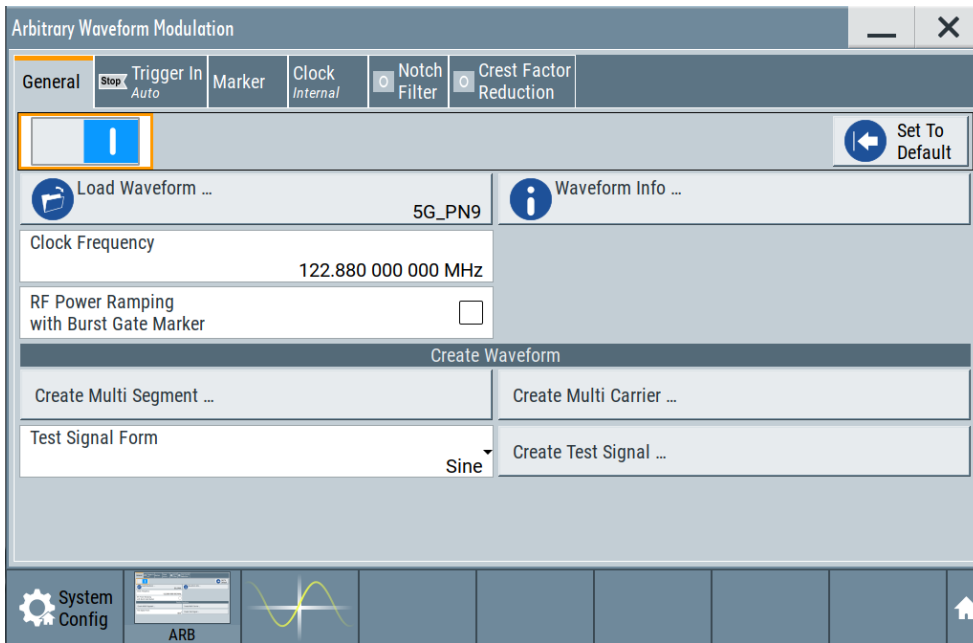


Figure 3-2 SMW200A works in ARB mode

	5G FR1 Uplink Signal Parameters
Subcarrier Spacing (kHz)	15, 30, 60
Channel Bandwidth (MHz)	5, 10, 15, 20, 25, 30, 40, 50, 60, 70, 80, 90, 100
OFDM Type	CP-OFDM, DFT-s-OFDM
Modulation	$\pi/2$ -BPSK, QPSK, 16QAM, 64QAM, 256QAM
RB Allocation Configuration (defined in 38.521)	Edge_Full_Left, Edge_Full_Right, Edge_1RB_Left, Edge_1RB_Right, Outer_Full, Inner_Full, Inner_1RB_left, Inner_1RB_Right

Table 3-1 An example of parameters of 5G FR1 Uplink Signal

As there are thousands of 5G uplink signal need to be tested, the corresponding waveforms should be prepared in advance to reduce the repeated calculation time. For the ARB mode of SMW200A, the waveforms have its own predefined format with the extension of *.WV. The 5G I/Q vectors for testing, stored for example in *.CSV files, need to be converted to *.WV waveforms. Following is an example of Matlab script for:

- ▶ Batch converting *.CSV files to waveform files within a folder;
- ▶ In case of 5G TDD signal, finding the start and end sample of the bursts to calculate the burst mean power.

The Python script for the waveforms converting:

```
import os
import csv
import math
import numpy
from datetime import date
import struct
```

```

# function of finding the start and end sample of a burst signal
def burstFinding(IQData):
    # granularity of the searching
    window = 200

    # threshold of the On->Off or Off->On sample edge
    threshold = 200

    # infinitely low value definition
    infLow = -400

    # calculate the power of the IQData
    power = []
    for IQPair in IQData:
        if abs(IQPair) == 0:
            # when the IQ are zeros, the power in dB is set to infLow
            power.append(infLow)
        else:
            power.append(10 * math.log10(abs(IQPair)))

    # find the start sample
    startFind = False
    for index in range(0, len(power), window):
        if power[index] - infLow >= threshold:
            startSample = index
            startFind = True
            break

    # find the end sample
    endFind = False
    if startFind == True:
        for index in range(startSample, len(power), window):
            if power[startSample] - power[index] >= threshold:
                endSample = index - window
                endFind = True
                break

```

```

        if endFind == False:
            endSample = index
            endFind = True

# return status is True if start and end sample are found
if startFind == True and endFind == True:
    status = True
else:
    status = False

return status, startSample, endSample

# IQData to WV converting
def mat2wvBurst(IQData, WVFileName, fSampleRate, bBurst, startSample,
endSample):
    # number of samples
    iNoOfSamples = len(IQData)

    # normalize the signal
    power = [abs(IQPair) for IQPair in IQData]
    powerMax = max(power)
    vfcSignal = [iq / powerMax for iq in IQData]

    # calculate the peak power
    fPeakPower = max([abs(x) for x in vfcSignal])
    fPeakPowerdBfs = -10 * math.log10(fPeakPower)

    # calculate the RMS power
    if bBurst == True:
        vfcSignalBurst = vfcSignal[startSample:endSample]
        fMeanPower = numpy.mean([(abs(x)) ** 2 for x in vfcSignalBurst])
    else:
        fMeanPower = numpy.mean([(abs(x)) ** 2 for x in vfcSignal])

    fRMSdBfs = -10 * math.log10(fMeanPower)

```



```

# Quantization to 16 bit
iMaxInt = 32767

viDataInterleaved = []
for iq in vfcSignal:
    viDataInterleaved.append(math.floor(iq.real * iMaxInt + 0.5))
    viDataInterleaved.append(math.floor(iq.imag * iMaxInt + 0.5))

# write waveform file
with open(WVFileName, "wb") as WVFile:
    WVFile.write("{TYPE: SMU-WV,0}".encode("ASCII"))
    WVFile.write("{COMMENT: Generated by mat2wv.m}".encode("ASCII"))
    WVFile.write("{DATE: " + str(date.today()) + "}".encode("ASCII"))
    WVFile.write("{LEVEL OFFS:" + "{:2.4f}".format(fRMSdBfs) +
":2.4f}".format(fPeakPowerdBfs)).encode("ASCII"))
    WVFile.write("{CLOCK:" + str(fSampleRate) + "}".encode("ASCII"))
    WVFile.write("{SAMPLES:" + str(iNOOfSamples) + "}".encode("ASCII"))
    WVFile.write("{WAVEFORM-" + str(4*iNOOfSamples+1) + ":
#").encode("ASCII"))
    WVFile.write(struct.pack("h"*len(viDataInterleaved),*viDataInterleaved))
    WVFile.write("}".encode("ASCII"))
    WVFile.close()

return fRMSdBfs

# the sample rate for the output WV file
fSampleRate = 491.52e6

# for TDD signal, the bBurst should be set to True to find the signal burst, and
calculate the RMS power of the burst
# otherwise, bBurst should be set to False
bBurst = True

# get current working directory
path = os.getcwd()

```

```

# CSV files as input
CSVFileNames = ["{}\\{}".format(path, x) for x in os.listdir() if
x.endswith(".csv") or x.endswith(".CSV")]

# log file
logFile = open("log.txt", "w")
logFile.write("Convert begin\n")

# flag indicating if the converting is finished successfully
convertSuccess = True

# converting CSV one by one
for CSVFileName in CSVFileNames:
    # read CSV data
    with open(CSVFileName) as CSVFile:
        CSVData = list(csv.reader(CSVFile))
        # transfer to complex IQ
        IQData = [float(CSVLine[0]) + 1j * float(CSVLine[1]) for CSVLine in
CSVData]

    # find the burst start and end sample
    if bBurst == True:
        status, startSample, endSample = burstFinding(IQData)
    else:
        status = True

    # generate WV file
    if status == True:
        # set the output WV file with the same name of the CSV file
        WVFileName = CSVFileName[0:-4] + ".wv"

        # IQData to WV
        logFile.write("Start to convert {}\n".format(CSVFileName))
        rms = mat2wvBurst(IQData, WVFileName, fSampleRate, bBurst, startSample,
endSample)
        logFile.write("Crest factor = {}\n".format(rms))
        logFile.write("Convert finish\n")
    else:

```

```

        logFile.write("Convert failed with {}\n".format(CSVFileName))

        convertSuccess = False

# to note if converting succeeds or fails
if convertSuccess == True:
    logFile.write("Success to convert files\n")
else:
    logFile.write("Failure happens during converting\n")

logFile.close()

```

3.2 5G NR Signal Analysis

The R&S®FSW is a high-performance signal and spectrum analyzer, with frequency range up to 85 GHz and IQ analysis bandwidth up to 8.3 GHz. The Tx signal of 5G UE can be analyzed by the FSW-K145 5G NR UL measurements option, as shown in Figure 3-3. The analysis result of EVM/ACLR of the 5G NR signal amplified by the PA helps to evaluate the performance of the PA.

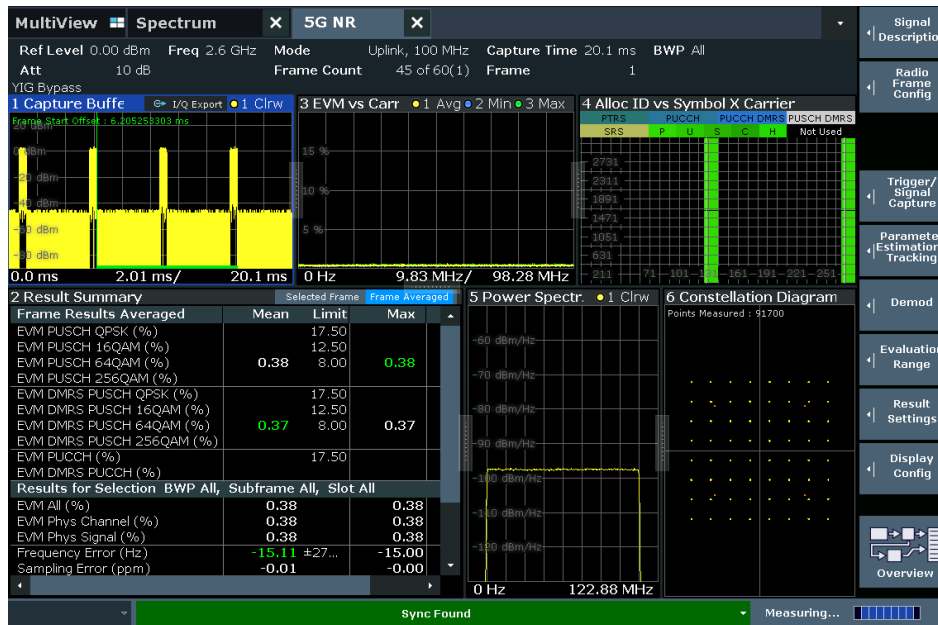


Figure 3-3 Analysis of a 5G NR uplink signal with FSW-K145

The R&S®VSE vector signal explorer software brings the power of R&S®FSW signal processing to PC [2]. It enables the remote control of R&S signal and spectrum analyzers or oscilloscopes to capture I/Qs and analyzing the I/Qs in the meantime.

In addition, the R&S®VSE also supports the off-line analysis. It is an innovative way of measurement by separating the RF signal collection and measurement. At the first step, the I/Qs are captured and stored on PC. At the second step, the stored I/Qs can be analyzed repeatedly by the R&S®VSE at any time.

The VSE-K144 is the specified option for 5G NR uplink and downlink analysis. Each signal subframe is analyzed, and a wide range of measurement results is provided, including EVM. The looking and operation of VSE-K144 on desktop is very similar to the FSW-K144/K145 on the instrument, as shown in Figure 3-4.

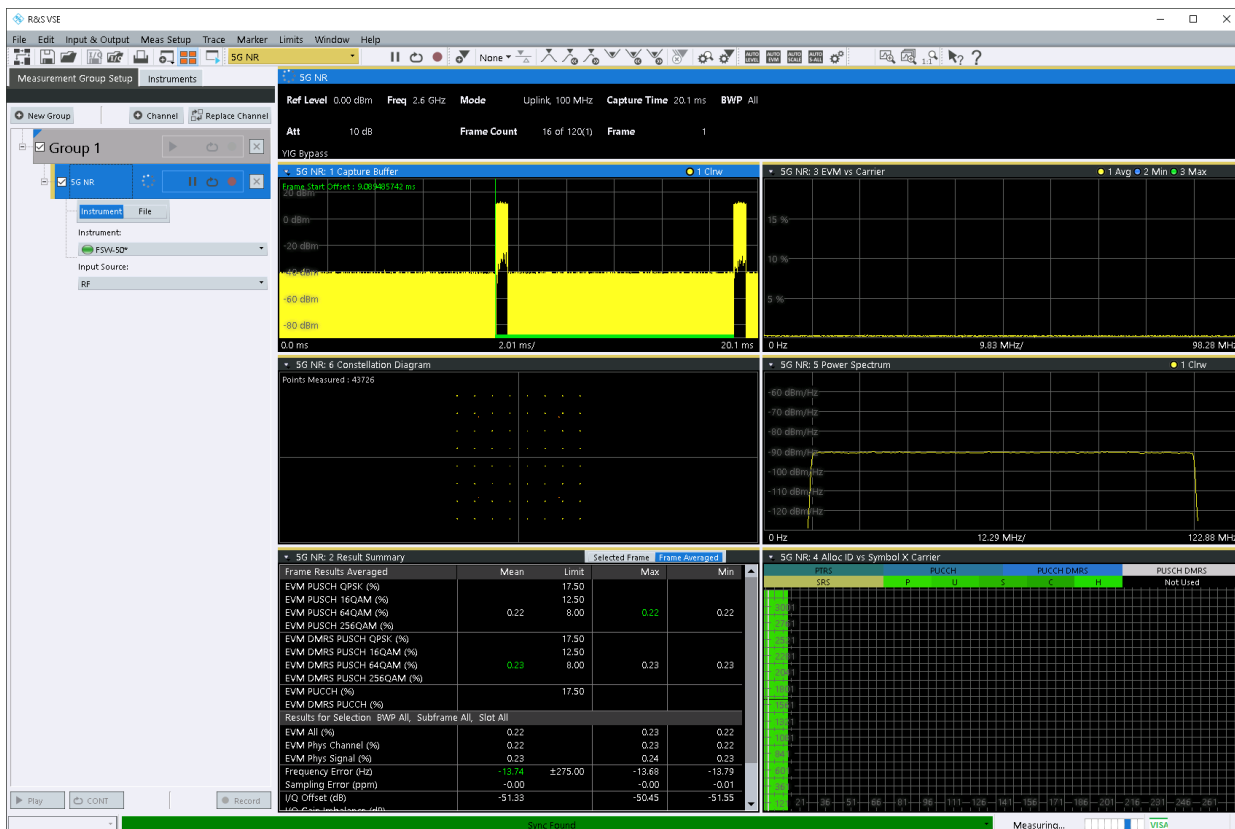


Figure 3-4 Analysis of a 5G NR uplink signal with VSE-K144

Traditionally, the signal capture and signal analysis are both executed on the instrument in a sequential method, as shown in Figure 3-5. By contrast, with R&S®VSE, the signal analysis can be executed on PC, and the instrument is only in charge of the signal capture, as shown in Figure 3-6. The 5G UE PA testing benefits from this kind of separation of RF signal collection and measurement method.

Firstly, as R&S®VSE can be running on a powerful PC with high performance processor and memory, the time cost for the signal analysis on PC could be highly reduced. The benefit could be more obvious when there are thousands of waveforms to test.

Secondly, the separation of RF signal collection and measurement enables the measurement in parallel method. As shown in Figure 3-5, the traditional measurement is executed in sequential method. The entire measurement routine is split to loops. In each loop, the instrument captures the IQ and analyzes the IQ, so IQ analysis is blocked until the IQ is available. As shown in Figure 3-6, since the IQ capture and signal analysis are separated, they can be scheduled parallelly to increase the overall measurement throughput in parallel method.

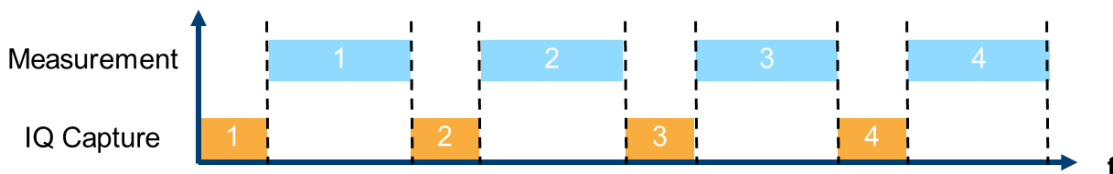


Figure 3-5 Signal capture and signal analysis on instrument in sequential method

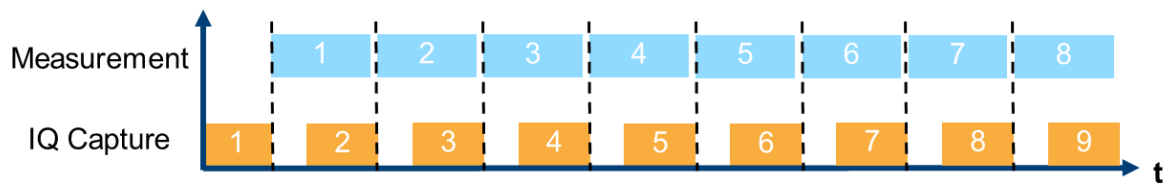


Figure 3-6 Signal capture on instrument and signal analysis on PC separately

Furthermore, with R&S®VSE the users can analyze and investigate the captured signal over and over again, change parameters and settings, examine the signal in depth and troubleshoot a wide range of the signals.

4 MIPI Control Interface

The MIPI RF Front End Control Interface (MIPI RFFE) is the world's de facto standard interface for control of RFFE subsystems including up to 19 components per bus instance, for example PA, LNA, antenna tuner, filter and switch [3].

For 5G NR TDD signal, the PA should be turned OFF when not in use to save power, for example during the Rx slot of the UE. As the PA is turned ON/OFF via the MIPI control interface, the transient state effects may influence the UE Tx performance, and the dynamic EVM needs to be measured to evaluate the performance degradation.

In the setup, the SignalCraft Serial Bus Controller SC4415 [4], as shown in Figure 4-1, is in charge of the MIPI RFFE control signal generation. The SC4415 is featured with:

- ▶ USB-to-Serial Bus Controller Provides Deterministic Serial Communications
- ▶ Supports MIPI-I3C (w/ Legacy I2C), MIPI-RFFE(v2.1), and SPI
- ▶ Software-Selectable Clock Rates (mode dependent), Supporting 50 kHz to 52 MHz in 1 kHz Steps
- ▶ Several Triggering Modes
- ▶ 8 General Purpose I/O (GPIO)
- ▶ Designed for Manufacturing Environments
- ▶ Field Upgradeable
- ▶ Wide Operating Temperature Range (0 °C to 70 °C)



Figure 4-1 SignalCraft SC4415

SC4415 has two IO pins for MIPI signal, one for clock and the other one for data, and is able to work in a triggering mode. The USB interface provides the power supply and programming interface for SC4415. The marker signal out of SMW200A is used as the trigger signal for SC4415. SC4415 works as the MIPI master to control the MIPI slaves such as the PA under test. When the rising edge of the trigger signal comes, the SC4415 will drive the clock and data pins with the preloaded MIPI commands in the buffer. The MIPI commands provides the functionality of the PA ON/OFF state switch and the read/write access to the registers of the PA.

Figure 4-2 shows the typical "PA ON" and "PA OFF" MIPI commands. The Clock signal is the serial clock signal driven by SC4415, and the Data signal is the bidirectional data line used to exchange data between SC4415 and the PA.

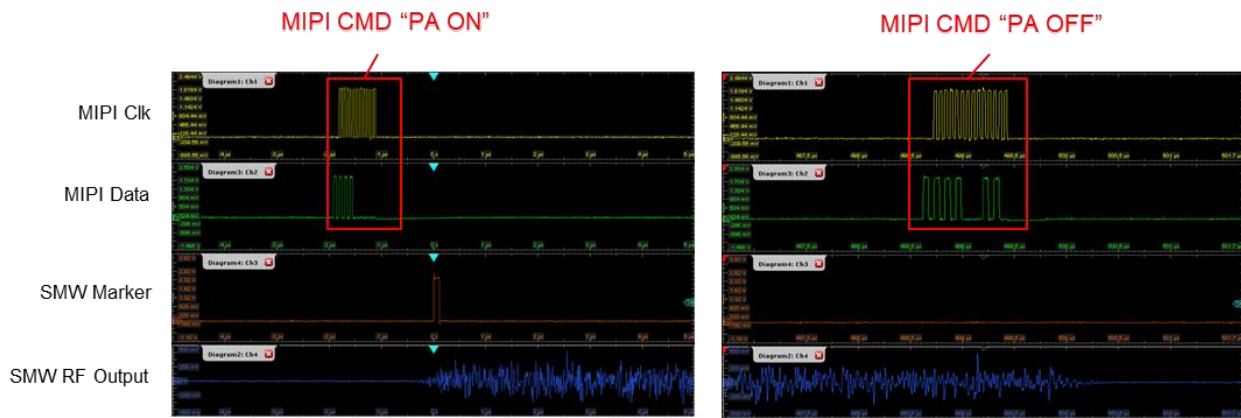


Figure 4-2 An example of MIPI "PA ON" and "PA OFF" commands

SC4415 shall be programmed following the flow chart shown in Figure 4-3. SC4415 works in Buffer Mode which allows for commands to be stored into command queues until a trigger signal is received. Doing so enables commands to be sequenced and executed in a deterministic fashion.

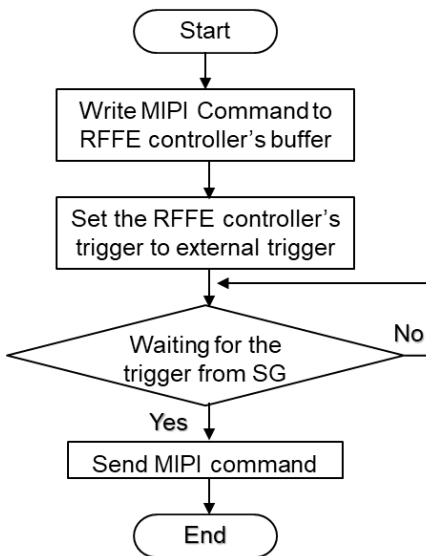


Figure 4-3 Flow chart of the program to generate the MIPI control signal

Several latencies must be taken into considerations in the program to ensure the ON period of the PA fully covers the burst of the input 5G NR TDD signal, as shown in Figure 4-4. One latency is the MIPI signal generation latency of SC4415 and the other one is the intrinsic ON/OFF status switching latency of the PA under test.

The inter-command delay of SC4415 can be tightly controlled by inserting Delay commands. The additional delay, specified in terms of clock cycles, helps to ensure the required command timing can be achieved. The clock rate of SC4415 is up to 52 MHz, with which the delay can be accurately tuned with a fine time resolution.

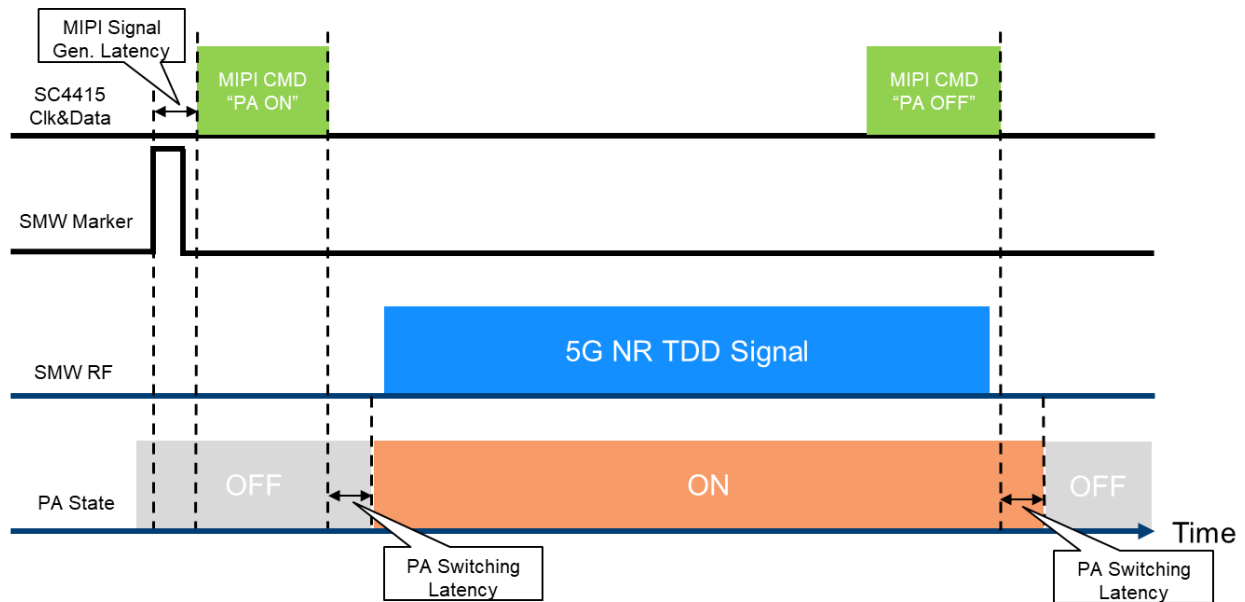


Figure 4-4 Latencies in the generation of MIPI control signal

When the command timing is not correctly achieved, it may happen that the PA is ON later than the starting of the input 5G TDD signal burst. This issue can be indicated by the EVM vs Symbol measurement within the 5G NR signal analysis application. As shown in Figure 4-5, the first several symbols of the frame may show abnormal EVM results.

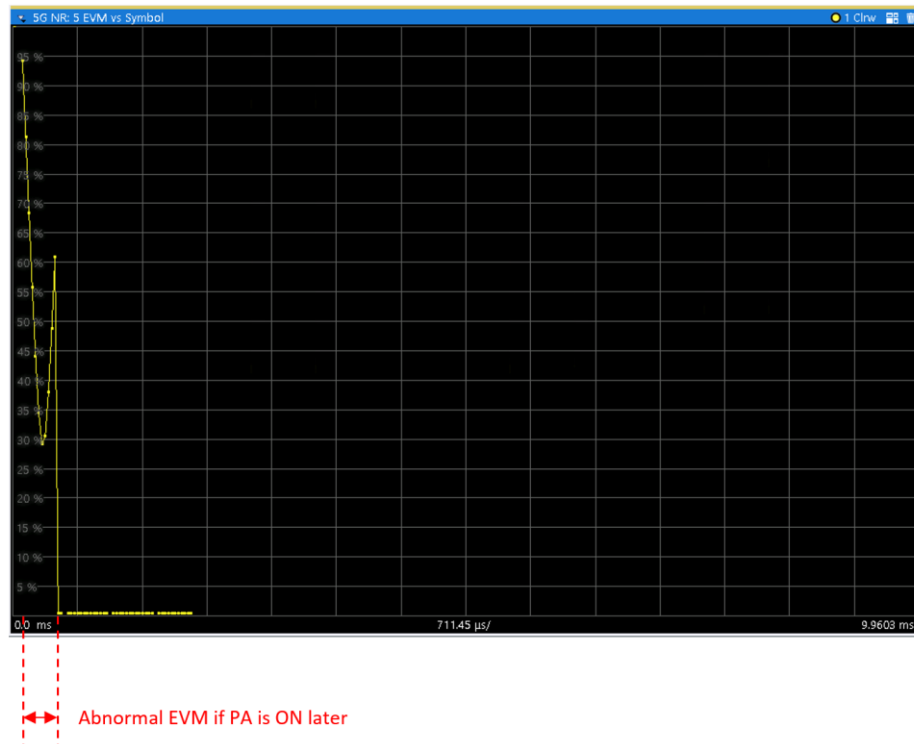


Figure 4-5 Example of EVM vs Symbol measurement result

5 Power Servo

The ACLR and EVM performance of the PA are very sensitive to the rated level, so before any PA measurements can be proceeded, the PA output power must be adjusted to a target level. This level adjustment procedure is also known as power servo.

The power servo implements the level adjustment algorithm in an iterative loop as shown in Figure 5-1. The signal out of PA is coupled to a power sensor which measures the PA output power. The program calculates the offset between the measured PA output power and the target PA output power, then adjusts the output level of the signal generator according to the offset. When the offset is within a defined limit range, the target PA output power is treated as achieved. Normally this offset is required to be smaller than 0.05 dB ~ 0.1 dB, so using a power sensor is necessary according to the power measurement uncertainty.

Usually the signal generator needs a relatively long time for output leveling. To speed up the power servo, the complete output leveling of the signal generator should be avoided. At the beginning of the algorithm, the initial output level and the attenuator of the signal generator are dedicatedly selected, so when trying to compensate the offset between measured and target power level, only the digital attenuator of the signal generator needs to be adjusted which is much quicker.

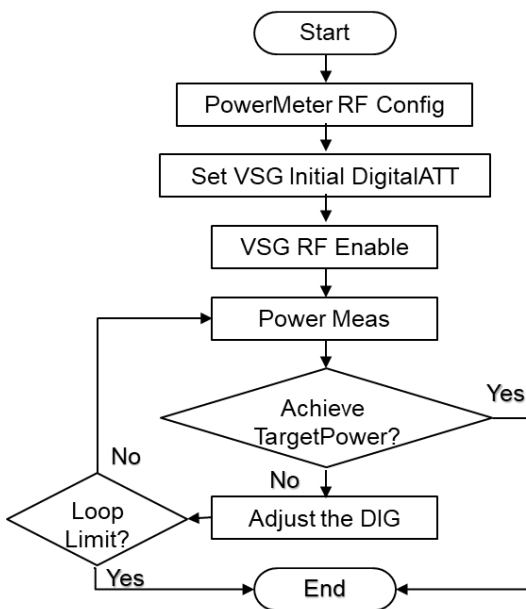


Figure 5-1 Flow chart of power servo

Python program for power servo:

```
import pyvisa
import time
import datetime

"""Connect to the instrument on IP"""
rm = pyvisa.ResourceManager()
smw = rm.open_resource("TCPIP0::192.168.1.29::hislip0::INSTR")
smw.timeout = 100000
```

```

rm = pyvisa.ResourceManager()
nrp = rm.open_resource("TCPIP0::192.168.1.40::hislip0::INSTR")
nrp.timeout = 10000
nrp.write("*IDN?")
read_string = nrp.read()
print(read_string)

"""Set the test parameter"""
power_goal = 0 #dBm #target of output power
freq = 2600
sg_loss = 0.6 #dB #path loss between SG and PA device
nrp_loss = 30.6 #dB #path loss between PA device and PM
level = 0 #dBm
pre_gain = 27 #dB #the estimated gain of PA
dig_att = level - (power_goal - pre_gain + sg_loss) # the actual output power of
SG is (level-dig_att) dBm

"""Setup SG"""
smw.write("SYST:PRES")
smw.write("*OPC?")
read_string = smw.read()
smw.write("SOUR1:BB:NR5G:SETT:TMOD:DL 'NR-FR1-TM1_1__FDD_100MHz_30kHz'")
smw.write("SOUR1:BB:NR5G:NODE:CELL0:PCFR %f" % (freq*1000000))
smw.write("SOUR1:BB:NR5G:OUTP:POW:MODE PSDC")
smw.write("SOUR1:BB:NR5G:OUTP:POW:RSBW 98.3e6")
smw.write("SOUR1:BB:NR5G:TRIG:OUTP:MODE SUBF")
smw.write("SOUR1:BB:NR5G:STAT 1")
smw.write("*OPC?")
read_string = smw.read()
smw.write("SOUR1:BB:IMP:OPT:MODE QHT")
smw.write("SOUR1:FREQ:CW %f" % (freq*1000000))
smw.write("SOUR1:POW:POW %f" % level)
smw.write("SOUR1:POW:ATT:DIG %f" % dig_att) # set digital att
smw.write(":OUTP1:STAT 1")
smw.write("*OPC?")
read_string = smw.read()

```

```

"""Setup Power Sensor"""
nrp.write("*RST")
nrp.write("SENS:FUNC 'POW:AVG'")
nrp.write("SENS:POW:AVG:APER 0.001")
nrp.write("SENSe:AVERAge:COUNT:AUTO OFF")
nrp.write("SENSe:AVERAge:COUNT 1")
nrp.write("SENSe:AVERAge:TControl REPEAT")
nrp.write("SENS:CORR:OFFS %.2f" % nrp_loss)
nrp.write("SENS:CORR:OFFS:STAT ON")
nrp.write("UNIT:POW DBM")
nrp.write("INIT:CONT OFF")
nrp.write("FREQ %f" % (freq*1000000))
nrp.write("TRIG:SOUR EXT2")
nrp.write("SYST:ERR:ALL?")
read_string = nrp.read()
print(read_string)
nrp.write("INIT:IMM")
nrp.write("*OPC?")
read_string = nrp.read()
nrp.write("FETC?")
power_read = nrp.read()
power = round(float(power_read), 2)

```

```

"""Loop to get the target power"""
nrp_power = []
loop_time = []
sg_time = []
pm_time = []
while True:
    pm_start = time.time()
    nrp.write("INIT:IMM")
    nrp.write("*OPC?")
    read_string = nrp.read()
    nrp.write("FETC?")
    power_read = nrp.read()
    # power = "%.2f" % float(power_read)

```

```

power = round(float(power_read), 2)
# print(power)
nrp_power.append(power)
pm_stop = time.time()
pm_time.append(round((pm_stop - pm_start), 8))

if abs(power_goal - power) < 0.1:
    # exit()
    break

sg_start = time.time()
dig_att = dig_att - (power_goal - power)
smw.write("SOUR1:POW:ATT:DIG %f" % dig_att)
smw.write("*OPC?")
read_string = smw.read()
sg_stop = time.time()
sg_time.append(round((sg_stop - sg_start), 8))
loop_time.append(round((sg_stop - pm_start), 8))

```

6 Improved PA automated testing

The typical entire UE PA testing procedure can be divided into three phases.

The first phase is the optimal PA mode register searching which is time costing.

The PA can be configured to different working state via modifying the register values. For an optimum performance, the working state shall be selected according to the PAE/EVM/ACLR measurement result. Several factors may influence the measurement result including different PA power classes, 5G configuration and 5G frequency bands. For example, a PA with a 4-bit register defines 16 different working state. It has to be tested 16 times by modifying the register value via MIPI interface, to obtain the optimal register value for the target scenario.

The combinations of different PA state registers, PA classes, waveforms of different 5G configuration modes and 5G frequency bands define a huge number of measurement loops. Usually it may take several days to complete the entire PA testing occupying labor resources and instruments. The test engineers urgently need a PA automated testing solution to reduce the time for test, and save energy from the tedious, repetitive and error-prone measurement work.

In the second phase, with the optimal PA mode register configuration, the PAE/EVM/ACLR performance of the PA are evaluated with all different 5G signal waveforms.

For 5G TDD signal, the PA can be turned ON/OFF via the MIPI control interface to save energy. The EVM measurement of the PA with the operation of power ON/OFF is called dynamic EVM measurement, which indicates the performance degradation of the PA due to the transient response effect.

Similarly, for 5G TDD signal, the voltage and current of the PA must be measured during the period when the PA is turned ON, which usually is in microsecond range. The R&S®NGM200 power supplies offer the high-speed FastLog functionality to record voltage and current readings versus time. Available sample rates for fast logging are 100, 1000, 10000, 50000, 250000 and 500000 values per second. The data can be stored on an external USB storage device or can be transferred to an external PC via the remote-control interface.

In the last phase, with the optimal PA mode register configuration, the Spurs measurement is proceeded with several typical 5G signal waveforms.

6.1 PA Modulation Measurement

A complete 5G UE PA testing setup for PAE/EVM/ACLR measurement is proposed in Figure 6-1. The test efficiency is highly improved benefitting from the system-level automated testing program, signal generation from waveforms, separation of RF signal collection and measurement, integration of MIPI control interface and the optimized power servo implementation.

The instrument configuration of the 5G UE PA automated testing setup consists of signal generator, signal and spectrum analyzer, power sensor, power supply and MIPI serial bus controller SC4415.

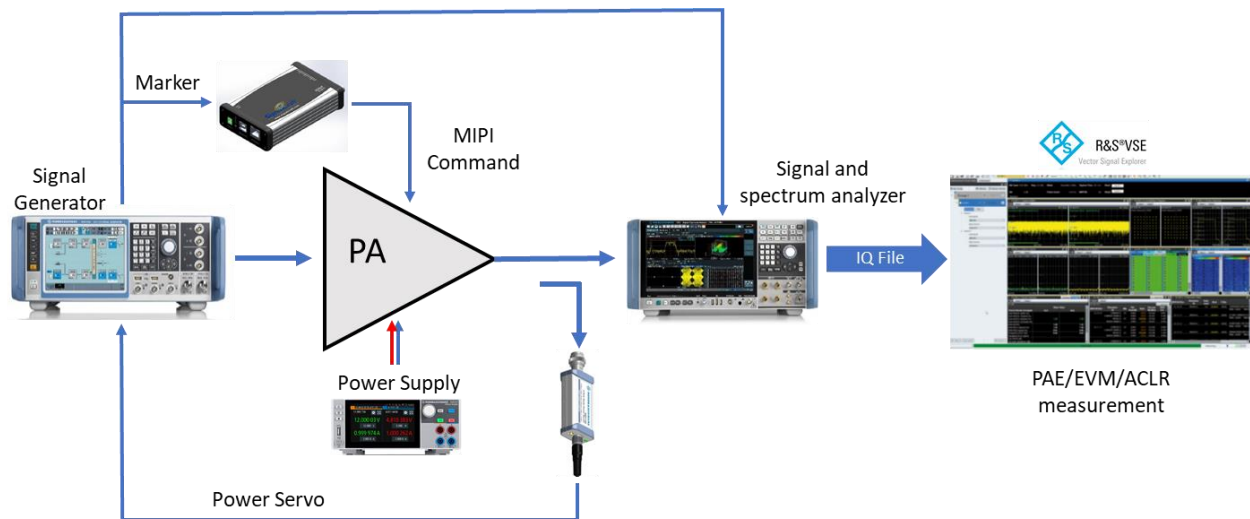


Figure 6-1 Setup for PA modulation measurement

A system-level python program remote controls the instruments, and manages the automated measurement routines. The flow chart of the program is shown in Figure 6-2. One measurement loop is made up of seven steps:

- ▶ Instruments Initialization
- ▶ MIPI Control Interface Configuration
- ▶ 5G Signal Generation
- ▶ Power Servo
- ▶ PA Current Measurement for PAE calculation
- ▶ IQ Capture
- ▶ 5G IQ Signal Analysis

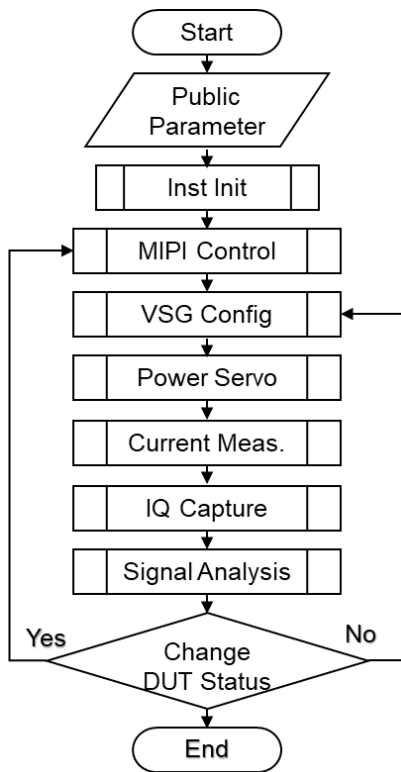


Figure 6-2 Flow chart of PA modulation test

The python program for I/Q capturing:

```

def InstIqCaptureInit(self, iq_capture_input):

    # Hz * Second
    self.target_file_size = int(8 * int(iq_capture_input['samplingrateHz']) *
float(iq_capture_input['captureTimeSec']))

    self._fsw.write("*RST;*CLS;*OPC?")
    status = self._fsw.read()
    self._fsw.write("INST:SEL IQ;:INIT:CONT OFF")

    # set center frequency
    cmd = "FREQ:CEN {0}".format(iq_capture_input['centerFrequencyHz'])
    self._fsw.write(cmd)

    # set ATT
    cmd = "INP:ATT {0}".format(iq_capture_input['inputAttdB'])
    self._fsw.write(cmd)
  
```

```

cmd = "DISP:TRAC:Y:RLEV:OFFS {}".format(iq_capture_input['offsetdB'])
self._fsw.write(cmd)

cmd = "DISP:WIND:TRAC:Y:RLEV {}".format(iq_capture_input['refLeveldBm'])
self._fsw.write(cmd)

# set sample rate
cmd = "TRAC:IQ:SRAT {}".format(iq_capture_input['samplingrateHz'])
self._fsw.write(cmd)

# set sample time
cmd = "SENS:SWE:TIME {}".format(iq_capture_input['captureTimeSec'])
self._fsw.write(cmd)

cmd = "TRIG:SOUR {}".format(iq_capture_input['triggerSource'])
self._fsw.write(cmd)

self._fsw.write("FORM REAL,32")
self._fsw.write("TRAC:IQ:DATA:FORM IQP")
self._fsw.write("TRAC:IQ:RLEN?")
self._fsw.write("TRAC:IQ:SRAT?")
self._fsw.write("FREQ:CEN?")

def InstIqCapture(self, capture_input):

    try:
        __iqw_file_name = capture_input["iqw"]
    except KeyError as e:
        print(f'bad object info:{e}')

    # set center frequency
    cmd = "FREQ:CEN {}".format(capture_input['frequency'])
    self._fsw.write(cmd)

    # set sample rate
    cmd = "TRAC:IQ:SRAT {}".format(capture_input['samplingrate'])

```



```

self._fsw.write(cmd)

# tStart = time.time()

self._fsw.write("INIT:IMM;*WAI;;TRAC:IQ:DATA:MEM?")

# eg-> #7
iq_buffer_head = str(self._fsw.read_bytes(2).decode('utf-8'))
if iq_buffer_head[0] == '#':
    iq_buffer_len = int(iq_buffer_head[1])
    iq_buffer_size =
int(str(self._fsw.read_bytes(iq_buffer_len).decode('utf-8')))
    iq_data = self._fsw.read_bytes(iq_buffer_size,
chunk_size=iq_buffer_size, break_on_termchar=False)
else:
    print("Read IQ buffer fail")

fo = open(__iqw_file_name, "wb")
fo.write(iq_data)
fo.close()
out_info = "OK"

return out_info, iq_data

```

The python program for 5G signal analysis in VSE:

```

# *****
# Function: Vse5GNRAnalsyis
# Input:      *args      -> Reserve
#            **Kwargs -> status(ON | OFF)
# *****

def Vse5GNRAnalsyis(self, *args, **kwargs):
    self._status = "OK"
    _waveform = ""
    _frequency_MHz = 0.0
    _samplingRate_Hz = 0.0
    _bandwidth_Hz = 0.0
    _channel = 1
    _read_res = ""

    try:
        if kwargs:
            if kwargs["waveform"] is not None:
                _waveform = kwargs["waveform"]

            if kwargs["bandwidth_MHz"] is not None:
                _bandwidth_Hz = kwargs["bandwidth_MHz"]*1000000

            if kwargs["samplingRate_MHz"] is not None:
                _samplingRate_Hz = kwargs["samplingRate_MHz"]*1000000

            if kwargs["frequency_MHz"] is not None:
                _frequency_MHz = kwargs["frequency_MHz"]

        self.InstWrite(":CONF:NR5G:LDIR UL")
        self.InstWrite(":CONF:NR5G:UL:CC1:FRAM1:BWP0:CSL 20")
        self.InstWrite("CONF:UL:CC1:FRAM1:BWP0:SLOT0:ALL0:MOD QAM256")

        for slot in range(1, 20):
            self.InstWrite(f":CONF:NR5G:UL:CC1:FRAM1:BWP0:SLOT{slot:d}:ATYP
UNUS")

        self.InstWrite(":CONF:UL:CC:RFUC:STAT OFF")

```

```

        self.InstWrite("NR5G:FRAM:COUN:AUTO OFF")

        self.InstWrite(f"INST:BLOC:CHAN:FILE:IQW '{_waveform}', {_bandwidth_Hz},
{_samplingRate_Hz}, IQP;*WAI")

self.InstWrite("FETC:SUMM:POW?;:FETC:CC:FRAM:SUMM:EVM?;:FETC:SUMM:FERR?;:FETCh:S
UMMary:FERRor:MAXimum?;:FETCh:SUMMary:FERRor:MINimum?")

        _read_res = self.InstRead()
    except KeyError:
        self._status = "G001"
    except pyvisa.errors.VisaIOError:
        self._status = "G003"

    return self._status

def Inst5GNRMeas(self,dict_5gnr_input):

    _wavaform = dict_5gnr_input["wavaform"]
    self.InstWrite(":CONF:NR5G:LDIR UL")
    self.InstWrite(":CONF:NR5G:UL:CC1:FRAM1:BWP0:CSL 20")
    self.InstWrite("CONF:UL:CC1:FRAM1:BWP0:SLOT0:ALL0:MOD QAM256")

    for slot in range (1,20):
        self.InstWrite(f":CONF:NR5G:UL:CC1:FRAM1:BWP0:SLOT{slot:d}:ATYP UNUS")

    self.InstWrite("NR5G:FRAM:COUN:AUTO OFF")

    self.InstWrite(f"INST:BLOC:CHAN:FILE:IQW '{_wavaform}', 300000000.000000,
375000000.000000, IQP;*WAI1")

```

6.2 PA Spurs Measurement

For PA Spurs measurement, the setup is additionally equipped with a tunable band-pass filter and a tunable band-stop filter, as shown in Figure 6-3.

Usually the 5G UE PA supports multiple frequency bands, so a group of band-pass filters is placed after the signal generator to improve the purity of the generated 5G signal as the input of the PA. The proper band-pass filter shall be selected within the group to match the frequency band.

The signal and spectrum analyzer FSW is the best choice for Spurs measurement, which offers superior displayed average noise level, dynamic range and measurement speed. To further mitigate the limitation to Spurs measurement from the dynamic range, a group of band-stop filters shall be placed before the signal

and spectrum analyzer. According to the frequency band, the proper band-stop filter is selected by switch. As the main TX signal output of the PA is filtered out, spurious emissions generated by the PA with low levels can be recognized by further improving a proper displayed average noise level of the signal and spectrum analyzer with the preamplifier.

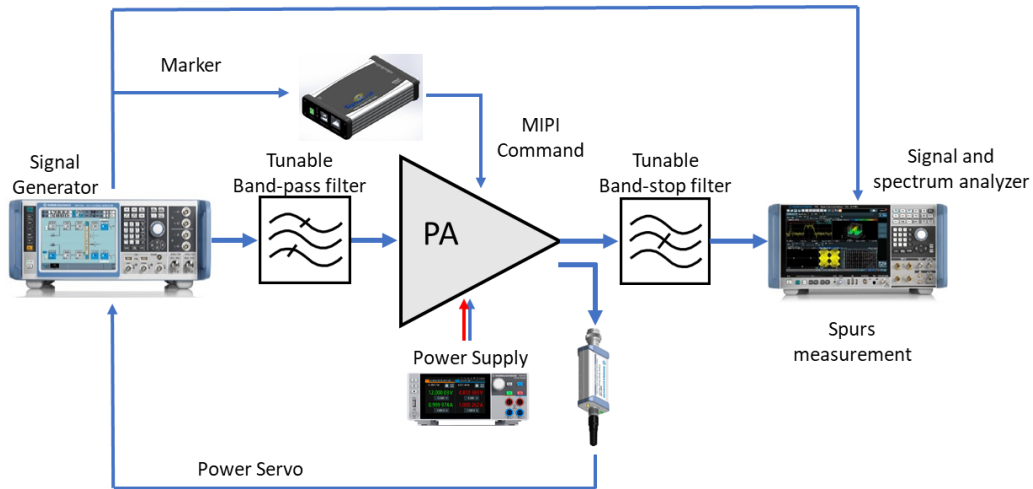


Figure 6-3 Setup for PA spurious measurement

7 Literature

- [1] Rohde & Schwarz, "5G NR eBook," [Online]. Available: <https://www.rohde-schwarz.com/5G-ebook>.
- [2] Rohde & Schwarz, "R&S®VSE Vector Signal Explorer Software - Product Brochure," 2021. [Online]. Available: <https://www.rohde-schwarz.com/au/brochure-datasheet/vse/>.
- [3] MIPI Alliance, "MIPI RF Front-End Control Interface," 2021. [Online]. Available: <https://www.mipi.org/specifications/rf-front-end>.
- [4] SIGNAL CRAFT TECHNOLOGIES, "SC4415 - Serial Bus Controller (MIPI-I3C, MIPI-RFFE, and SPI)," [Online]. Available: <https://www.signalcraft.com/products/test-measurement/serial-bus-controllers/sc4415/>.

8 Ordering Information

8.1 R&S®SMW200A

Designation	Type	Order No.
Vector signal generator	R&S®SMW200A	1412.0000.02
Frequency options, RF path A, 100 kHz to 7.5 GHz	R&S®SMW-B1007	1428.7700.02
Baseband main module option	R&S®SMW-B9	1413.7350.02
Baseband main module option	R&S®SMW-B13XT	1413.8005.02
5G New Radio	R&S®SMW-K144	1414.4990.02

8.2 R&S®FSW

Designation	Type	Order No.
Signal and Spectrum Analyzer, 2 Hz to 26.5 GHz	R&S®FSW26	1331.5003.26
2000 MHz Analysis Bandwidth	R&S®FSW-B2001	1331.6916.14
RF Preamplifier	R&S®FSW-B24	1313.0832.26
3GPP 5G-NR UL Measurements	R&S®FSW-K145	1338.3612.02

8.3 R&S®VSE

Designation	Type	Order No.
Vector Signal Explorer Base Software, Basic Edition	R&S®VSE	1345.1011.06
License dongle	R&S®FSPC	1310.0002.03
3GPP 5G NR downlink and uplink measurement application	R&S®FSW-K144	1309.9574.06

8.4 R&S®NGM202

Designation	Type	Order No.
Two-channel power supply	R&S®NGM202	3638.4472.03

8.5 R&S®NRP8S

Designation	Type	Order No.
100 pW to 200 mW, 10 MHz to 8 GHz	R&S®NRP8s	1419.0006.02
USB interface cable, length: 0.75 m	R&S®NRP-ZKU	1419.0658.02

Rohde & Schwarz

The Rohde & Schwarz electronics group offers innovative solutions in the following business fields: test and measurement, broadcast and media, secure communications, cybersecurity, monitoring and network testing. Founded more than 80 years ago, the independent company which is headquartered in Munich, Germany, has an extensive sales and service network with locations in more than 70 countries.

www.rohde-schwarz.com

Certified Quality Management

ISO 9001

Rohde & Schwarz training

www.training.rohde-schwarz.com

Rohde & Schwarz customer support

www.rohde-schwarz.com/support

