|  |  |
| --- | --- |
| **Program / Project Name:** | CA5G |
| **Checklist / Template Completed by:** | TBD |
| **Date Completed:** | Click here to enter a date. |

**Tips for CA5G Development**

Revision: A.0

**Detailed Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Rev** | **Date** | **Editor(s)** | **Description of change** |
| A.0 | 18-SEP-2019 | Sang-Gu Kang | Initial Draft |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

[1 Environment 7](#_Toc35869592)

[1.1 SSH 7](#_Toc35869593)

[1.2 Manual updated 7](#_Toc35869594)

[1.3 QtCreator Setup 7](#_Toc35869595)

[1.3.1 Build 설정 11](#_Toc35869596)

[1.3.2 Run(Deploy) 설정 11](#_Toc35869597)

[1.3.3 Trouble shooting 12](#_Toc35869598)

[2 Creating new feature 12](#_Toc35869599)

[2.1 check branch 12](#_Toc35869600)

[2.1.1 .gitmodule 12](#_Toc35869601)

[2.1.2 branch 13](#_Toc35869602)

[2.2 check directory and files 13](#_Toc35869603)

[2.3 check CMakeLists.txt 14](#_Toc35869604)

[2.4 check main.cpp 16](#_Toc35869605)

[2.5 Check processes.json 16](#_Toc35869606)

[2.6 Check devload.sh 16](#_Toc35869607)

[2.7 Check submodule 17](#_Toc35869608)

[3 Add new source file 17](#_Toc35869609)

[4 Add new item 19](#_Toc35869610)

[4.1 Action 19](#_Toc35869611)

[4.2 Config 19](#_Toc35869612)

[4.2.1 confluence 등록 19](#_Toc35869613)

[4.2.2 enumLists.json 20](#_Toc35869614)

[4.2.3 Add Item in xxxItem.cpp 20](#_Toc35869615)

[4.2.4 Add function in xxxSync.cpp 20](#_Toc35869616)

[4.2.5 Add input in baseservermodule.cpp 20](#_Toc35869617)

[4.3 Result 21](#_Toc35869618)

[4.3.1 Confluence 등록 21](#_Toc35869619)

[4.3.2 header file 21](#_Toc35869620)

[4.3.3 source file 21](#_Toc35869621)

[4.4 Config Item 등록 22](#_Toc35869622)

[4.5 enumLists 추가 22](#_Toc35869623)

[4.6 Add source 23](#_Toc35869624)

[5 SCPI 24](#_Toc35869625)

[5.1 add file 24](#_Toc35869626)

[5.2 SCPI 명령 작성 24](#_Toc35869627)

[5.3 telnet을 이용한 검증 24](#_Toc35869628)

[6 CMakeLists 25](#_Toc35869629)

[7 Option활성화 비활성화 25](#_Toc35869630)

[7.1 Activation 25](#_Toc35869631)

[7.2 Deactivation 26](#_Toc35869632)

[8 Bitbucket 26](#_Toc35869633)

[8.1 pull request 26](#_Toc35869634)

[9 SW Integration Test 26](#_Toc35869635)

[9.1 Cucumber 26](#_Toc35869636)

[10 Git 28](#_Toc35869637)

[10.1 Branch 28](#_Toc35869638)

[10.1.1 Branch changing flow 28](#_Toc35869639)

[10.1.2 Branch commit 29](#_Toc35869640)

[10.2 Useful git command 30](#_Toc35869641)

[10.2.1 add 30](#_Toc35869642)

[10.2.2 branch 30](#_Toc35869643)

[10.2.3 checkout 30](#_Toc35869644)

[10.2.4 clean 30](#_Toc35869645)

[10.2.5 commit 30](#_Toc35869646)

[10.2.6 merge 30](#_Toc35869647)

[10.2.7 mergetool 30](#_Toc35869648)

[10.2.8 pull 30](#_Toc35869649)

[10.2.9 push 31](#_Toc35869650)

[10.2.10 rebase 31](#_Toc35869651)

[10.2.11 reset 31](#_Toc35869652)

[10.2.12 rm 31](#_Toc35869653)

[10.2.13 submodule 31](#_Toc35869654)

[10.2.14 status 31](#_Toc35869655)

[10.3 SSH Key Gen 32](#_Toc35869656)

[11 Debug 34](#_Toc35869657)

[11.1 gview 34](#_Toc35869658)

[11.2 ps 34](#_Toc35869659)

[11.3 qDebug() 34](#_Toc35869660)

[12 FPGA loading 36](#_Toc35869661)

[12.1 develop 36](#_Toc35869662)

[12.2 Using lfs 38](#_Toc35869663)

[12.3 System Information 39](#_Toc35869664)

[13 Package 39](#_Toc35869665)

[13.1 package for test 39](#_Toc35869666)

[13.2 Nightly\_build 41](#_Toc35869667)

[13.3 Making package 42](#_Toc35869668)

[13.4 Sharing package 45](#_Toc35869669)

[14 Temp 45](#_Toc35869670)

[15 Util 56](#_Toc35869671)

[15.1 doQt 56](#_Toc35869672)

[15.2 gitk 57](#_Toc35869673)

[15.3 subl 57](#_Toc35869674)

[15.4 gitg 57](#_Toc35869675)

[15.5 dbus-monitor 57](#_Toc35869676)

[15.6 gview 60](#_Toc35869677)

[15.7 Byond Compare 63](#_Toc35869678)

[16 Project 64](#_Toc35869679)

[17 Source Analysis 64](#_Toc35869680)

[17.1 파일 별 기능 구조 64](#_Toc35869681)

[17.1.1 baseservermodule.cpp 64](#_Toc35869682)

[17.1.2 각각의 measure 파일 64](#_Toc35869683)

[17.1.3 slot.cpp 65](#_Toc35869684)

[17.1.4 Review 65](#_Toc35869685)

[17.2 Config, Setup, Mode change 65](#_Toc35869686)

[17.3 Measure Update 65](#_Toc35869687)

[17.4 Measurement ID 65](#_Toc35869688)

[17.4.1 Action Item 65](#_Toc35869689)

[17.4.2 Config Item 66](#_Toc35869690)

[17.4.3 Result Item 66](#_Toc35869691)

[17.5 Technology 66](#_Toc35869692)

[17.6 lib 66](#_Toc35869693)

[17.6.1 dsp5GNR 66](#_Toc35869694)

[17.7 Function 66](#_Toc35869695)

# Environment

## SSH

Port: 22

User: root

Password: SiG2018

## Manual updated

\\ds\apdata$\SEO\RFTest\PLM\Public\Technical Publications

## QtCreator Setup

스크린샷이(가) 표시된 사진

자동 생성된 설명

스크린샷이(가) 표시된 사진

자동 생성된 설명  
스크린샷이(가) 표시된 사진

자동 생성된 설명

스크린샷이(가) 표시된 사진

자동 생성된 설명

스크린샷이(가) 표시된 사진

자동 생성된 설명

스크린샷이(가) 표시된 사진

자동 생성된 설명

스크린샷이(가) 표시된 사진

자동 생성된 설명

### Build 설정

스크린샷, 노트북, 컴퓨터, 실내이(가) 표시된 사진

자동 생성된 설명

### Run(Deploy) 설정

스크린샷, 컴퓨터, 노트북이(가) 표시된 사진

자동 생성된 설명

* Build 결과물을 장비에 write하기 위해 필요한 설정이다.
* Add Deploy Step버튼을 눌러 Custom Process Step을 선택한다.
* Command: /usr/bin/rsync
* Arguments: -av ./install/\* --rsh="sshpass -p SiG2018 ssh -l root" [root@10.82.26.22:/](mailto:root@10.82.26.22:/)
* Run configuration: (on Remote Device)로 되어 있는 것 선택
* Alternate executable on device: 에서 Use this command instead 체크박스 선택한 후 service.sh입력

### Trouble shooting

* 설정값이 제대로 되지 않는다면(Projects>Build에 Build directory가 제대로 잡히지 않을 때) Build directory 안의 내용을 모두 지우고, 커맨드 라인 상에서 build-ca5g-CA5G-Default 디렉토리를 날려버리고 다시 CMake Run을 실행한다.

# Creating new feature

## check branch

### .gitmodule

lib 디렉토리는 변경사항이 없으므로 develop을 사용하고, 나머지 submodule은 해당 브랜치를 사용한다는 가정.



### branch

apps: 주요 작업이 해당 브랜치에서 이루어지므로 명확히 브랜치 이동이 이루어져 있어야 한다.

lib, qml, service: 서브모듈 링크를 따라가도록 해쉬를 가리키도록 되어 있다. 만약 GUI 담당자가 작업을 해서 업데이트가 되면 CMake가 돌 때, 자동으로 업데이트되도록 하는 효과가 있다.



## check directory and files



## check CMakeLists.txt






lib 디렉토리 때문에 아래 부분도 수정  


tmAnalyzer\_SOURCE 부분에 적어도 하나의 파일을 주석처리 하지 않아야 Run CMake가 이상없이 진행된다.





## check main.cpp

아래 내용을 알맞게 수정해야 함. 지금은 Run CMake 때문에 임시로 한 것임.



## Check processes.json



## Check devload.sh

./scripts/systemd/devload.sh를 수정해 주면 start가 동작한다.



## Check submodule

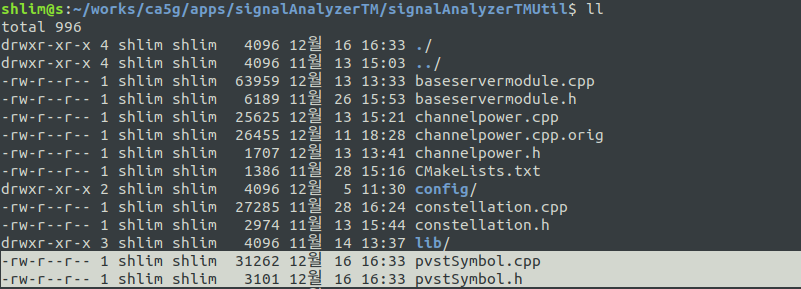
.gitmodules에 정보가 있음.

|  |
| --- |
| [submodule "elf"]  path = elf  url = https://gitlab.ds.jdsu.net/lee68914/elf-separation.git  [submodule "lib"]  path = lib  url = ssh://git@cosgit1.ds.jdsu.net:7999/ca5g/ca5g\_libs.git  branch = feature/HA-3653-tm-analyzer  [submodule "apps"]  path = apps  url = ssh://git@cosgit1.ds.jdsu.net:7999/ca5g/ca5g\_apps.git  branch = feature/HA-3653-tm-analyzer  [submodule "qml"]  path = qml  url = ssh://git@cosgit1.ds.jdsu.net:7999/ca5g/ca5g\_ui.git  branch = feature/HA-3653-tm-analyzer  [submodule "services"]  path = services  url = ssh://git@cosgit1.ds.jdsu.net:7999/ca5g/services.git  branch = feature/HA-3653-tm-analyzer |

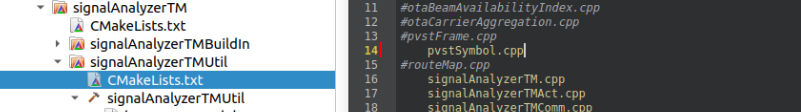
ca5g 디렉토리에서 branch를 변경하더라도 ca5g\_apps 디렉토리에는 branch 변경이 이루어지지 않는다. 따라서 branch를 일치시켜 놓고 작업하는 것이 편하다.

# Add new source file

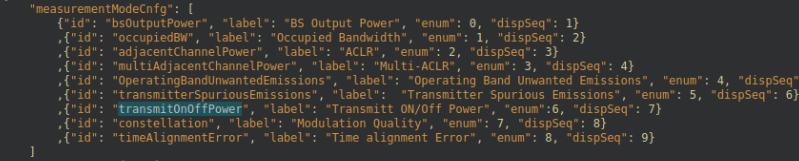
* Make or add new source



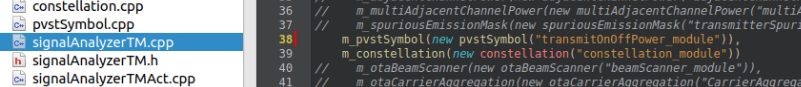
* CMakeLists.txt에 등록



* enumLists.json check



* Module 등록



* File 수정

Searching keyword: m\_<className>

signalAnalyzerTM.h  
-. Include header file: #include "pvstSymbol.h"  
-. Add smart pointer: std::unique\_ptr<pvstSymbol> m\_pvstSymbol;

signalAnalyzerTM.cpp  
-. Init module: m\_pvstSymbol(new pvstSymbol("transmitOnOffPower\_module")),  
-. Regist baseservermodule: m\_pvstSymbol.get()  
-. Create child module: addChildModule(m\_pvstSymbol.get());  
-. Check license: m\_pvstSymbol->m\_5GNRLic=Lic5G;

signalAnalyzerTMAct.cpp  
-. callActSubmodule: nRet = m\_pvstSymbol.get()->actSubmodule(nCmd,nArg, params);  
-. callGpsInfoSubmodule:   
-. Act enable debug: m\_pvstSymbol->m\_5GNRLic=TRUE;

signalAnalyzerTMComm.cpp  
-. getInsTraceComm: pRet = &m\_pvstSymbol.get()->m\_traceComm;  
-. getInsMarkerResult: pRet = &m\_pvstSymbol.get()->m\_infoMk[nSelMk];

# Add new item

## Action

todo

## Config

updateMeasure()  
 makeNRTMParamerter(): input parameter를 DSP에 전달  
 p\_5GTMModule->measureProcess(): 각각의 measure에 맞는 DSP 프로세스  
 updateMeasureData(): measureProcess()를 돌고 난 뒤 업데이트 된 result를 GUI로 전달  
 updateParameter()

### confluence 등록

todo

### enumLists.json

|  |
| --- |
| ,"selectAntennaCnfg": [  {"id": "1000", "label": "1000", "enum": 0, "dispSeq": 1}  ,{"id": "1001", "label": "1001", "enum": 1, "dispSeq": 2}  ] |

### Add Item in xxxItem.cpp

|  |
| --- |
| bool signalAnalyzerTM::initItemSet()  {  ADD\_CONFIG\_ITEM\_ACT(m\_set.m\_selectAntenna, addStringConfig, "selectAntennaCnfg", tr("Antenna Port"), "1000", &signalAnalyzerTM::syncSelectAntenna);  } |

### Add function in xxxSync.cpp

|  |
| --- |
| bool signalAnalyzerTM::syncSelectAntenna()  {  int nSel = m\_eItem->getOrderfromValue("selectAntennaCnfg", m\_set.m\_selectAntenna->getValue());  m\_set.m\_selectAntenna->commit();  return true;  } |

### Add input in baseservermodule.cpp

|  |
| --- |
| void BaseServerModule::makeNRTMParameter(int nMeasureMode,EngTMInput \*inputNRTMData,EngTMOutput \*outputNRTMData)  {  if(getEnumNumber("externalOffsetModeCnfg") == eON)  {  }  } |

## Result

### Confluence 등록

todo

### header file

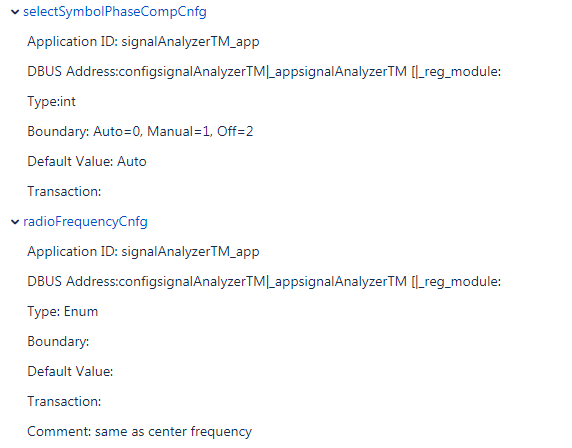
DoubleElfItem\* m\_pdschDmRsPwrRslt = nullptr;

### source file

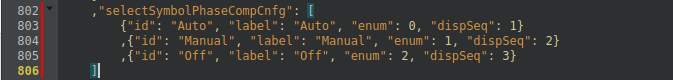
-. constructor  
m\_pdschDmRsPwrRslt = (**addDoubleResult**("timeAlignmentErrorPDSCHDMRSPowerRslt", tr("PDSCH DM-RS Power"), 0)[0]);

void timeAlignmentErrorMimo::**genResult**(CsvVariantData \*m\_varData)  
{  
 m\_varData->addRow(QList<QVariant>()  
 << QVariant("PDSCH DM-RS Power:")  
 << QVariant(m\_pdschDmRsPwrRslt->getValue())  
 );  
}

## Config Item 등록

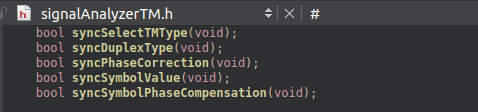


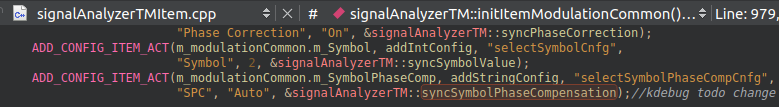
## enumLists 추가

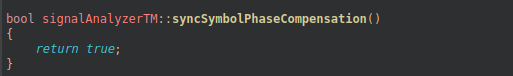


## Add source

signalAnalyzerTMComm.h  


signalAnalyzerTM.h  


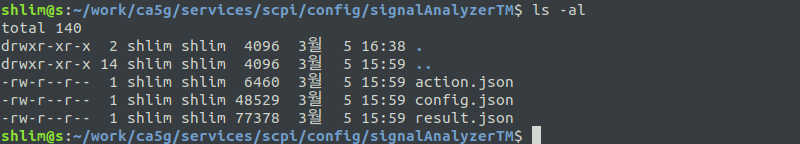
signalAnalyzerTMItem.cpp  


signalAnalyzerTMSync.cpp  


# SCPI

## add file

해당 위치(ca5g/services/scpi/config/.)에 파일을 만들어 추가한다. CMake Run을 하면 Qt에서도 파일을 볼 수 있다.





## SCPI 명령 작성

명령 작성 후 reboot을 해야 적용된다.

## telnet을 이용한 검증

port는 5600

## scpi debugging

SSH에서 scpi 명령을 치면 상태를 볼 수 있다.

|  |
| --- |
| root@tb5800-a0291d:/# scpi  Connnecting remote control service : "com.viavisolutions.platform.RemoteControl"  Connnected remote control service : "com.viavisolutions.platform.RemoteControl"  MMAP Creating gpio DevBlock  MMAP Creating gpioMux DevBlock  MMAP Creating fpgaPci DevBlock  Started Scpi Register execute  "/base/config/scpi/config/realtimeAnalyzer/result.json"  "/base/config/scpi/config/realtimeAnalyzer/action.json"  "/base/config/scpi/config/realtimeAnalyzer/config.json"  SCPI Error2(command has numeric index(#), but elfId does not) : :config:spectrumAnalyzerBuildIn\_app:spectrumAnalyzerBuildIn\_reg\_module:modeMultiSegmentLineCnfg:  SCPI Error2(command has numeric index(#), but elfId does not) : :config:spectrumAnalyzerBuildIn\_app:spectrumAnalyzerBuildIn\_reg\_module:numberOfLineCnfg:  SCPI Error1(Duplicated command) : REALtime:MARKer#:TYPE  "/base/config/scpi/config/signalAnalyzer5GNR/result.json"  SCPI Error2(Duplicated command) : NR5G:BAI:PCI?  "/base/config/scpi/config/signalAnalyzer5GNR/action.json" |

개별 scpi 명령의 동작 여부를 알기 위해 다음과 같이 할 수 있다.

|  |
| --- |
| root@tb5800-a0291d:~# scpiClient NRTM:TAEMimo:ANTenna1:TIMoffset? |

# CMakeLists

qml등의 sub module을 원하는 브랜치에서 가져와 사용(컴파일)할 수 있기를 원할 때 아래와 같이 막는다.

|  |
| --- |
| execute\_process(COMMAND git submodule init  WORKING\_DIRECTORY ${CMAKE\_SOURCE\_DIR})  #execute\_process(COMMAND git submodule update --recursive --remote  # WORKING\_DIRECTORY ${CMAKE\_SOURCE\_DIR}) |

그리고 나서 브랜치를 바꾼 후 컴파일하면 된다.

# Option활성화 비활성화

SSH 접속 후 직접 입력. 타 부서 및 문서 노출 절대 안됨. 명령 입력 후 장비 재부팅해야 적용됨.

## Activation

enable\_options.sh F001O

## Deactivation

disable\_options.sh F001O

# Bitbucket

## pull request

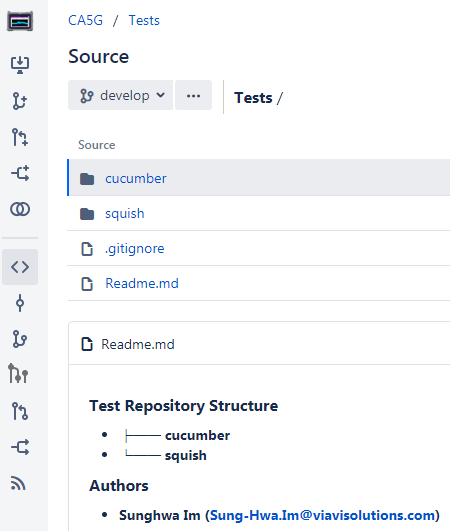
local에서 push하면 메시지에 remote 주소가 나온다. 링크를 열면 bitbucket에 연결되어 create pull request 화면이 나타난다.

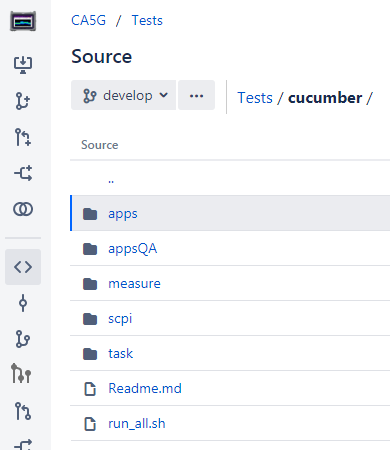
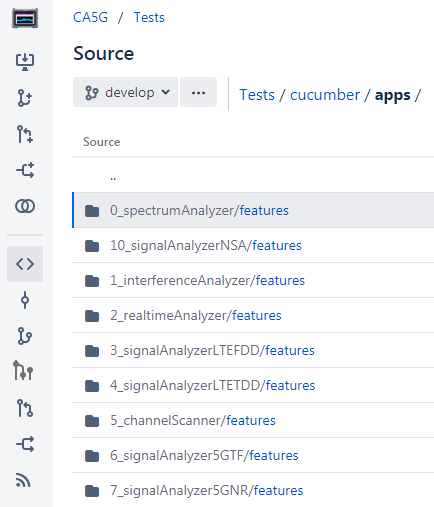
|  |
| --- |
| shlim@s:~/work/ca5g/apps$ **git push**  Counting objects: 27, done.  Delta compression using up to 4 threads.  Compressing objects: 100% (27/27), done.  Writing objects: 100% (27/27), 4.72 KiB | 2.36 MiB/s, done.  Total 27 (delta 21), reused 0 (delta 0)  remote:  remote: Create pull request for feature/nrtm-analyzer:  remote: https://cosgit1.ds.jdsu.net/projects/CA5G/repos/ca5g\_apps/pull-requests?create&sourceBranch=refs/heads/feature/nrtm-analyzer  remote:  To ssh://cosgit1.ds.jdsu.net:7999/ca5g/ca5g\_apps.git  5f8637ff..1fa0d776 feature/nrtm-analyzer -> feature/nrtm-analyzer |

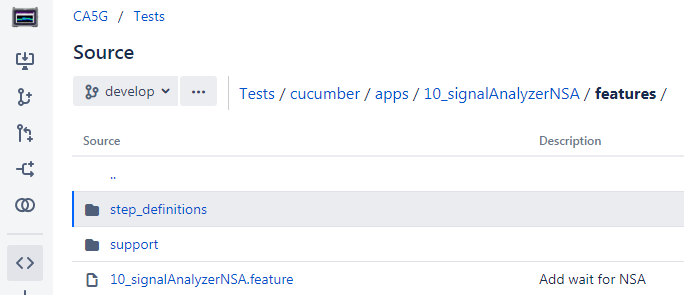
# SW Integration Test

## Cucumber

아래의 위치에서 xxx.feature만 수정해서 사용하면 된다.



# Git

## Branch

### Branch changing flow

브랜치를 변경한 경우에는 부모 프로젝트와 자식 프로젝트 간 submodule을 초기화 시켜주어야 한다.

* Git clone  
  shlim@s:~/workspace/practice$ git clone ssh://git@cosgit1.ds.jdsu.net:7999/ca5g/ca5g.git
* Checkout  
  shlim@s:~/workspace/practice/ca5g$ git checkout feature/HA-3653-tm-analyzer
* Submodule deinit  
  
* Submodule init  
  
* doQt 실행  
  open CMakeLists.txt
* Run CMake  
  Build > Run CMake
* Build, Run 설정 체크  
  -. 좌측 Projects 버튼 선택  
  -. Build 항목 선택 후 1.3.1 Build 설정 참조  
  -. Run 항목 선택 후 1.3.2 Run(Deploy)설정 참조
* Build All  
  Build > Build All
* Deploy All  
  Build > Deploy All
* Run
* Warning message  
  Warning: The unit file, source configuration file or drop-ins of scpi.service changed on disk. Run 'systemctl daemon-reload' to reload units.와 같은 warning 메시지가 발생하면 장비에 systemctl daemon-reload 명령을 타이핑 해 주거나 장비를 다시 시작하면 된다.

### Branch commit

하위 디렉토리의 변경된 내용을 먼저 commit 하고 상위 ca5g 디렉토리로 와서 다시 commit한다???

## Useful git command

### add

git add manifest/processes.json scripts/system/devload.sh

git add dspTM/

### branch

git branch

### checkout

git checkout feature/HA-3653-tm-analyzer

git checkout develop

git checkout origin/feature/HA-3653-tm-analyzer

git checkout 26e67aba

git checkout <file name>

### clean

git clean -f

### commit

git commit -am “Moved dspTM to apps repo”

git commit -a

### merge

git merge --abort

git merge develop

### mergetool

git mergetool

### pull

git pull origin feature/HA-3653-tm-analyzer

git pull

### push

git push

git push origin feature/HA-3653-tm-analyzer

git push origin origin/feature/HA-3653-tm-analyzer

### rebase

git rebase --continue

git rebase --abort

git rebase develop

git rebase --help

git rebase origin/develop

### reset

git reset --hard origin/feature/HA-3653-tm-analyzer

git reset --hard

### rm

git rm -rf dspTM/

git rm dsp5GNR/consts.h

git rm -r dsp5GNR/ -f

### submodule

git submodule update --recursive --remote

git submodule update --recursive --remote -f

git submodule init

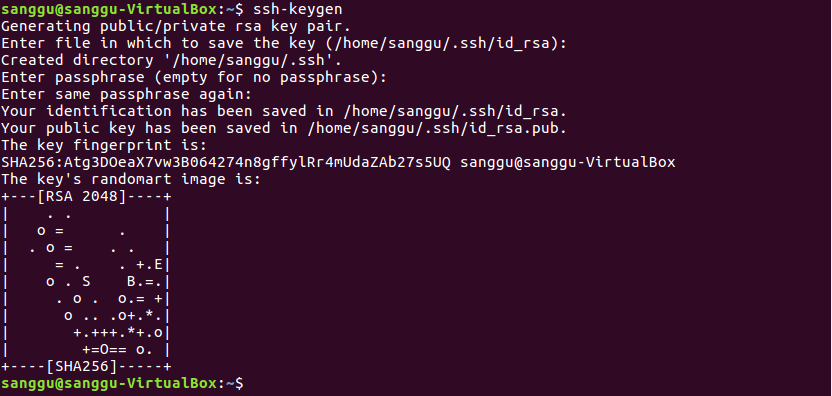
git submodule deinit -f --all

### status

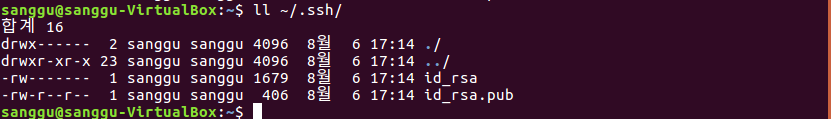
git status

## SSH Key Gen

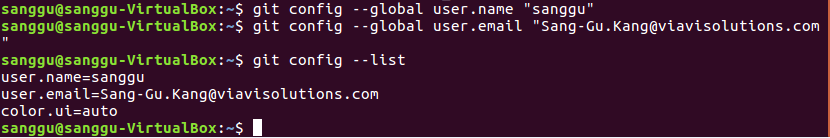
-. Key gen



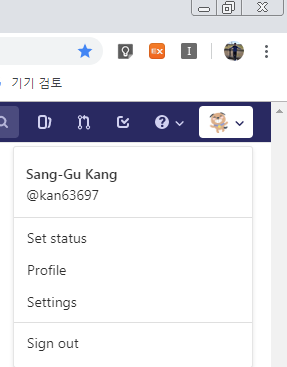
-. Confirm

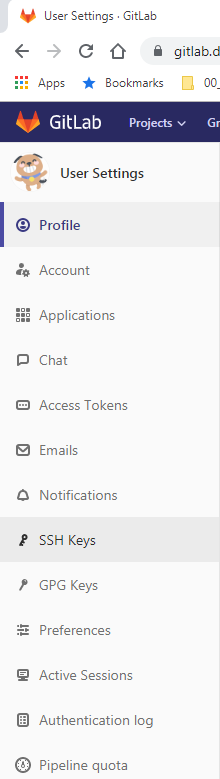


-. Global setup



-. Git Host Add SSH Key





Copy from ~/.ssh/id\_rsa.pub

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDsq/SDJbENre+JeaJK75LxgWWIVd/S6ZTlBziJnfqRKGZxJnIGJqIEI4mIzAmPhYiukaUH/TcZ8lUhwGiE2FpD1mBSkIauMF7/oK0EI67Iun8wyHLgr73/g64SGsnGBIUcqYU/b5ALH2OKm1S6EJpKMFPXSZ2caGq9G7ScoI/08wSNnJg+McdcnYy388vY5EZ1p85L44bzbStZ8DOlGEEAcv5wZ5reO44UfWxHHmlGFTTwbsNHwPLQRR+hJYFGIfQQyBrRQph8FYB96Qoen2mG4fRUmU3n1yk7X4FzFCJE60cuZV9ivD1xe2sVFWUqYJT47RpiB71ZvANbDismVhMt sanggu@sanggu-VirtualBox

Fingerprint: 7a:4b:63:3c:42:f7:e7:ac:8b:4d:fe:92:e0:8d:7d:e0

# Debug

## gview

SSH에서 해당 명령을 입력하여 코드에서의 printf 메시지를 본다.

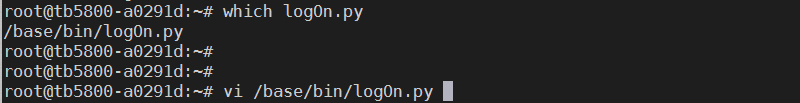
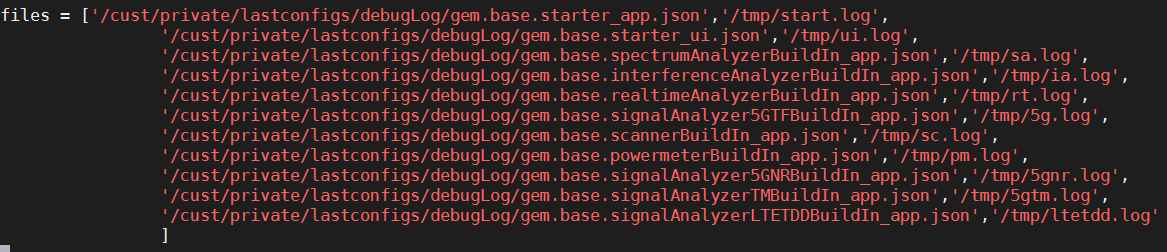
|  |
| --- |
| gview -f $(tty) -o gem.base.signalAnalyzerTMBuildIn\_app |

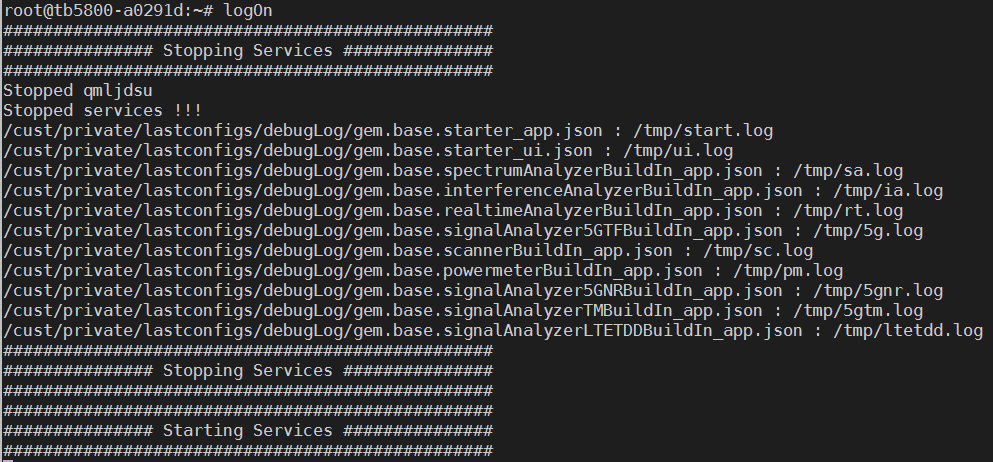
## ps

SSH에서 해당 명령을 입력하여 process가 제대로 도는지 확인한다.

|  |
| --- |
| ps -ax |
| 3625 ? Sl 0:02 /base/bin/signalAnalyzerTMBuildIn\_app |

## qDebug()

logOn.py 파일 수정(추가)  
  


SSH에서 logOn을 한번 입력해 준다. 그러면 /tmp/ 디렉토리에 추가한 ltetdd.log 파일이 생긴다.  


SSH에서 tail -f /tmp/5gtm.log 명령을 주면 qdebug 메시지가 보인다.



# FPGA loading

## develop

* DSP bin 파일을 sftp로 ca5g에 밀어 넣는다. Drag and drop to device.  
    
  -. Using MobaXterm  
  
* protoFlash <bank number> <bin file name>   
  This is for TM(Version: 9333, Bank 8, 5GNR+nrTM+LTE)  
  protoFlash 7 ./9333\_HetNet\_top\_v4.bin  
    
  
* protoFpgaConfig <bank number>  
  protoFpgaConfig 7  
  
* check address and compare with bin file  
    
  0x2457 convert to decimal using calculator  
  0x2457 == 9333(version is same. Ok)

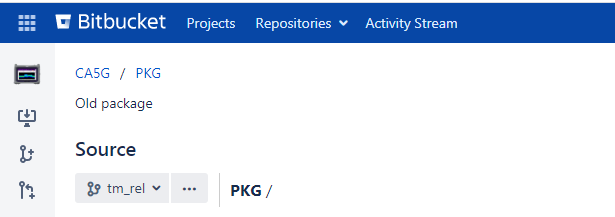
## Using lfs

* lfs가 설치되어 있지 않다면 설치  
  
* pull  
  
* bin 파일 commit  
    
    
  
* lfs install  
  
* git push  
  
* check  
  

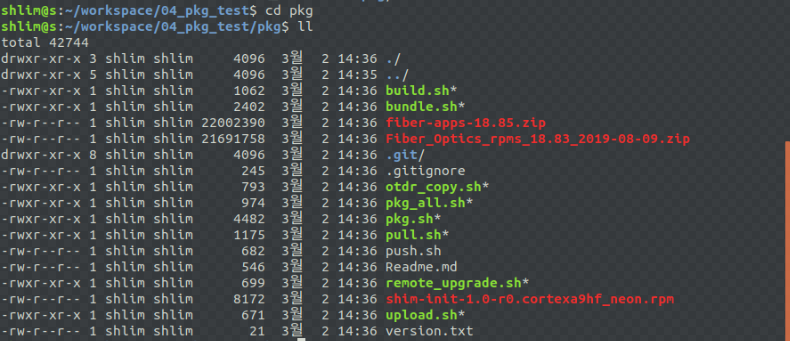
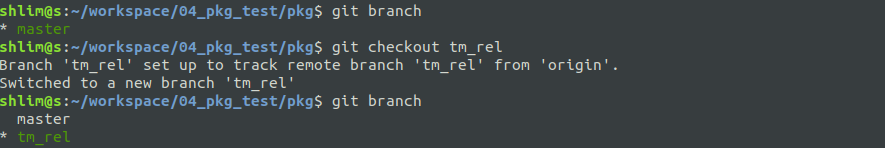
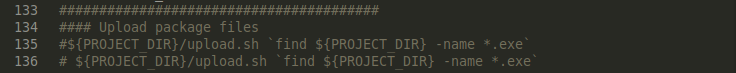
## System Information

# Package

## package for test

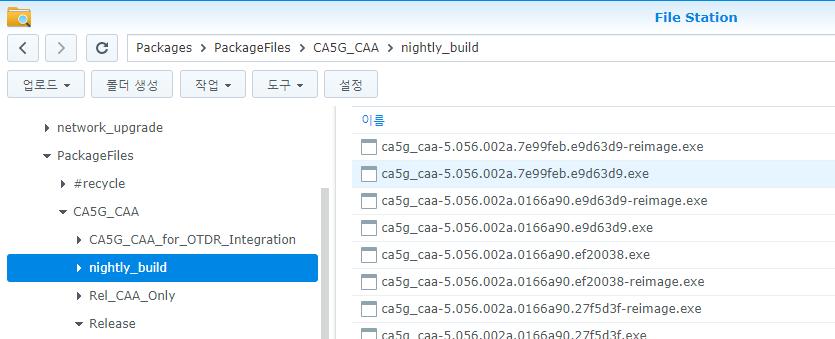


NOTE: 원격 저장소에 테스트용 fw를 위한 브랜치가 PKG에 만들어져 있다고 가정한다. 이 브랜치는 TM analyzer만을 위한 script 등을 포함하고 있고 필요에 따라 수정해서 쓰도록 하기 위해 만들어졌다.

* clone PKG  
  shlim@s:~/work$ git clone ssh://git@cosgit1.ds.jdsu.net:7999/ca5g/pkg.git PKG
* pkg directory  
  
* checkout TM release branch  
  
* version.txt 수정  
  
* pkg.sh 수정  
  REL\_BRANCH(ca5g and caa)  
    
    
  cmake -DBUILD\_PLATFORM  
    
    
  commit, push 주석처리  
    
    
  exe package upload 주석처리  
  
* pkg.sh 스크립트 실행  
  shlim@s:~/workspace/04\_pkg\_test/pkg$ ./pkg.sh  
  …  
  Files read from disk: 1  
  Archive size: 615085639 bytes (587 MiB)  
  Everything is Ok
* exe 파일 확인  
  성공하면 upgrade-exe 디렉토리가 생성되고 그 안에 exe 파일이 만들어진다. 이 파일을 윈도우에서 실행시키면 다운로드 파일이 만들어진다.
* push(나중에 버전이 어디까지 진행되었는지 확인 가능)
* confluence에서 Team>Patch/HotFixRelease에 해당 내용 업데이트

## Nightly\_build

매일 build해서 develop의 image를 저장해 놓는다.



## Making package

-. git lfs pull (???)



-. Check fpga file (???)



-. git clone pkg

mkdir pkg

git clone ssh://git@cosgit1.ds.jdsu.net:7999/ca5g/pkg.git pkg

cd pkg

git checkout tm\_rel

git pull

-. Version.txt 수정



-. Script 실행

./pkg.sh

-. Check release file

Upgrade-exe/\*.exe

-. window에서 실행시키면 파일이 만들어짐



NOTE: pkg.sh 수정

cmake .. -DWITH\_CAA=true -DCMAKE\_C\_FLAGS="-s" -DBUILD\_PLATFORM=ca5g

REL\_BRANCH 수정: REL\_BRANCH=feature/HA-3653-tm-analyzer

caa해당 브랜치 정의 : REL\_BRANCH=release/rel\_055

commit & push 지우기

exe 패키지 업로드 : ${PROJECT\_DIR}/upload.sh

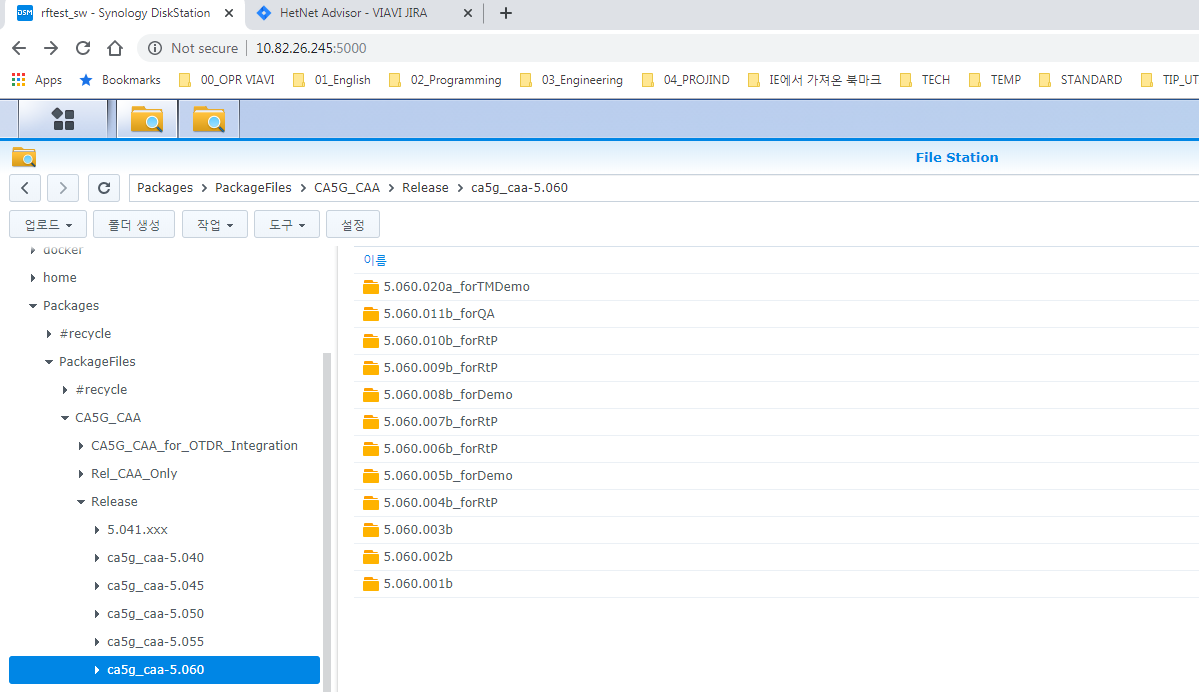






## Sharing package

-. 서버에 올려서 공유한다.



# Temp

주요 작업은 /base/

장비의 저장 장치는 /user/cellAdvisor5G/Internal

Save/Load등 할 때 임시 파일 위치 /tmp

/tmp/logOn

Journalct기

X86에서 시작할 때는 start.sh

/base

/base/bin

service.sh or service.sh start는 켤 때, service.sh stop은 끌 때

/base/bin/qmljdsu

/base/manifest process.json 새로 measure를 만들면 이곳에 등록

Confluence에 UI와 협의한 Item들을 등록해야 함(서로 일치하게 하기 위해서)



Debug할 때는 gview나 dbus-monitor를 사용함. Gview는 개발할 때 유용하고, dbus-monitor는 UI와 App 중 문제가 어디 있는지 진단할 때 유용.








































Kit(CA5G)에서 실행할 수 있도록






# Util

## doQt

Qt 실행할 때는 doQt 명령으로 실행.

## gitk

git log와 유사함.

## subl

notepad와 비슷한 tool.

## gitg

## dbus-monitor

dbus를 모니터한다.

**root@tb5800-a0291d:~# dbus-monitor**

signal time=1581618907.582107 sender=org.freedesktop.DBus -> destination=:1.693 serial=2 path=/org/freedesktop/DBus; interface=org.freedesktop.DBus; member=NameAcquired  
 string ":1.693"  
signal time=1581618907.582782 sender=org.freedesktop.DBus -> destination=:1.693 serial=4 path=/org/freedesktop/DBus; interface=org.freedesktop.DBus; member=NameLost  
 string ":1.693"  
signal time=1581618907.608141 sender=:1.666 -> destination=(null destination) serial=6770 path=/; interface=jdsu.elf; member=SigResultChangedNotification  
 array [  
 struct {  
 string ":result:spectrumAnalyzerBuildIn\_app:spectrumTuned\_module:traceDataYRslt:doubleARRAY:"  
 variant array [  
 double -50.4281  
 double -49.7094  
 double -48.6156  
 double -48.3969  
 double -48.6976

중간 생략

double 0  
 double 0  
 ]  
 int32 1048579  
 }  
 ]

중간 생략

signal time=1581618909.696288 sender=:1.666 -> destination=(null destination) serial=6791 path=/; interface=jdsu.elf; member=SigResultChangedNotification  
 array [  
 struct {  
 string ":result:spectrumAnalyzerBuildIn\_app:spectrumTuned\_module:traceDataYRslt:doubleARRAY:"  
 variant array [  
 double -60.0726  
 double -54.2523  
 double -51.5258

중간생략

double 0  
 double 0  
^C

**root@tb5800-a0291d:~# dbus-monitor | grep result**string ":result:spectrumAnalyzerBuildIn\_app:spectrumTuned\_module:traceDataYRslt:doubleARRAY:"  
string ":result:spectrumAnalyzerBuildIn\_app:spectrumTuned\_module:traceDataYRslt:doubleARRAY:"  
string ":result:spectrumAnalyzerBuildIn\_app:spectrumTuned\_module:traceDataYRslt:doubleARRAY:"  
string ":result:spectrumAnalyzerBuildIn\_app:spectrumTuned\_module:traceDataYRslt:doubleARRAY:"  
string ":result:spectrumAnalyzerBuildIn\_app:spectrumTuned\_module:traceDataYRslt:doubleARRAY:"  
string ":result:spectrumAnalyzerBuildIn\_app:spectrumTuned\_module:traceDataYRslt:doubleARRAY:"  
string ":result:spectrumAnalyzerBuildIn\_app:spectrumTuned\_module:traceDataYRslt:doubleARRAY:"  
^C

**root@tb5800-a0291d:~# dbus-monitor | grep config**  
string ":config:spectrumAnalyzerBuildIn\_app:spectrumAnalyzerBuildIn\_reg\_module:stepFrequencyCnfg:"  
string ":config:spectrumAnalyzerBuildIn\_app:spectrumAnalyzerBuildIn\_reg\_module:displayTypeFrequencyCnfg:"  
string ":config:spectrumAnalyzerBuildIn\_app:spectrumAnalyzerBuildIn\_reg\_module:centerFrequencyCnfg:"  
string ":config:spectrumAnalyzerBuildIn\_app:spectrumAnalyzerBuildIn\_reg\_module:centerFrequencyCnfg:"  
string ":config:spectrumAnalyzerBuildIn\_app:spectrumAnalyzerBuildIn\_reg\_module:stopFrequencyCnfg:"  
string ":config:spectrumAnalyzerBuildIn\_app:spectrumAnalyzerBuildIn\_reg\_module:startFrequencyCnfg:"  
string ":config:spectrumAnalyzerBuildIn\_app:spectrumAnalyzerBuildIn\_reg\_module:channelNumberCnfg:"  
string ":config:spectrumAnalyzerBuildIn\_app:spectrumAnalyzerBuildIn\_reg\_module:resolutionBandwidthCnfg:"  
string ":config:spectrumAnalyzerBuildIn\_app:spectrumAnalyzerBuildIn\_reg\_module:timeSweepCnfg:"  
string ":config:spectrumAnalyzerBuildIn\_app:spectrumAnalyzerBuildIn\_reg\_module:currentMinTimeSweepCnfg:"  
^C

NOTE: config, action의 경우 명령을 먼저 실행한 후 device에 원하는 동작을 시킨다.

**root@tb5800-a0291d:~# dbus-monitor | grep action**  
string ":action:spectrumAnalyzerBuildIn\_app:spectrumAnalyzerBuildIn\_reg\_module:stop:"  
string ":action:spectrumAnalyzerBuildIn\_app:spectrumAnalyzerBuildIn\_reg\_module:stop:"  
string ":action:spectrumAnalyzerBuildIn\_app:spectrumAnalyzerBuildIn\_reg\_module:stop:"  
string ":action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:start:"  
string ":action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:start:"  
string ":action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:traceClearAllActn:"  
string ":action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:start:"  
string ":action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:traceClearAllActn:"  
string ":action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:traceClearAllActn:"  
string ":action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:stop:"  
string ":action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:stop:"  
string ":action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:stop:"  
string ":action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:start:"  
string ":action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:start:"  
string ":action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:start:"  
^C

**root@tb5800-a0291d:~# dbus-monitor | grep config -a**

**root@tb5800-a0291d:~# dbus-monitor | grep config -a1**

**root@tb5800-a0291d:~# dbus-monitor | grep config -a2**

**root@tb5800-a0291d:~# dbus-monitor | grep config -a5**

**root@tb5800-a0291d:~# dbus-monitor | grep config -b**

NOTE: 정보의 양을 조절하는 옵션. a means after, b means before

## gview

**root@tb5800-a0291d:~# gview**

Select application by service name:

1. gem.base.basemgr  
2. gem.base.RFoCPRIBuildIn\_app  
3. gem.base.caaHwService  
4. gem.base.cableAntennaAnalyzer\_app  
5. gem.base.cableAntennaAnalyzer\_ui  
6. gem.base.interferenceAnalyzerBuildIn\_app  
7. gem.base.powermeterBuildIn\_app  
8. gem.base.realtimeAnalyzerBuildIn\_app  
9. gem.base.scannerBuildIn\_app  
10. gem.base.signalAnalyzer5GNRBuildIn\_app  
11. gem.base.signalAnalyzer5GTFBuildIn\_app  
12. gem.base.signalAnalyzerLTEFDDBuildIn\_app  
13. gem.base.signalAnalyzerLTETDDBuildIn\_app  
14. gem.base.signalAnalyzerTMBuildIn\_app  
15. gem.base.spectrumAnalyzerBuildIn\_app  
16. gem.base.starter\_app  
17. gem.base.starter\_ui

Enter application choice (1-17, enter 'q' to exit): **14**

Select module:

1. :result:signalAnalyzerTMBuildIn\_app:adjacentChannelPower\_module:manifest:  
2. :result:signalAnalyzerTMBuildIn\_app:bsOutputPower\_module:manifest:  
3. :result:signalAnalyzerTMBuildIn\_app:constellation\_module:manifest:  
4. :result:signalAnalyzerTMBuildIn\_app:multiAdjacentChannelPower\_module:manifest:  
5. :result:signalAnalyzerTMBuildIn\_app:occupiedBW\_module:manifest:  
6. :result:signalAnalyzerTMBuildIn\_app:operatingBandUnwantedEmissions\_module:manifest:  
7. :result:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:manifest:  
8. :result:signalAnalyzerTMBuildIn\_app:timeAlignmentError\_module:manifest:  
9. :result:signalAnalyzerTMBuildIn\_app:transmitOnOffPower\_module:manifest:  
10. :result:signalAnalyzerTMBuildIn\_app:transmitterSpuriousEmissions\_module:manifest:

Enter module choice (1-10, enter 'q' to exit): **7**

Select action for  
service=gem.base.signalAnalyzerTMBuildIn\_app  
module=:result:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:

1. streamstdout  
2. streamstderr  
3. debug log settings  
4. streamlog  
5. dumpconfigs  
6. dumpresults  
7. dumpmanifest  
8. action  
9. setconfig  
10. setresult  
11. getconfig  
12. getresult  
13. showenv  
14. showpid  
15. showcmd  
16. showcwd  
17. raisehome  
18. dumpprocs  
19. Choose another application / module context

Enter action choice (1-19, enter 'q' to exit): **8**

Select ELF action ID:

1. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:applyconfig:  
2. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:autoScaleActn:  
3. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:captureActn:  
4. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:clear\_uservisible\_log:  
5. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:dumpactions:  
6. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:dumpconfigs:  
7. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:dumplog:  
8. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:dumpresults:  
9. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:enableDebugActn:  
10. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:get\_uservisible\_log:  
11. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:halt:  
12. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:loadResultActn:  
13. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:loadconfig:  
14. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:logStartActn:  
15. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:logStopActn:  
16. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:markerAllOffActn:  
17. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:minSearchActn:  
18. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:moveToCenterActn:  
19. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:moveToStartActn:  
20. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:moveToStopActn:  
21. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:nextPeakActn:  
22. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:nextPeakLeftActn:  
23. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:nextPeakRightActn:  
24. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:peakSearchActn:  
25. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:presetActn:  
26. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:presetActnWOMeas:  
27. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:removeconfig:  
28. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:resetMeasureActn:  
29. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:saveConfResultActn:  
30. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:saveIQDataActn:  
31. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:saveIQDataScpiActn:  
32. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:saveResultCsvActn:  
33. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:saveconfig:  
34. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:setCalculateTrace05:  
35. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:setCalculateTrace06:  
36. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:start:  
37. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:startSA:  
38. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:stop:  
39. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:stopSA:  
40. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:streamlog:  
41. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:sweepOnceActn:  
42. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:traceClearAllActn:  
43. :action:signalAnalyzerTMBuildIn\_app:signalAnalyzerTMBuildIn\_reg\_module:updateParameterActn:  
44. Go back to previous menu

Enter ELF action ID choice (1-44, enter 'q' to exit): q

## Byond Compare

ca5g$ **bcompare** apps/signalAnalyzer5GNR/ apps\_old/signalAnalyzer5GNR/

# Project

DSP library는 ca5g\_libs에 위치한다.

JIRA issue resolve하면 QA 팀으로 assign할 것.

# Source Analysis

## 파일 별 기능 구조

### baseservermodule.cpp

* **makeNRTMParameter**(int nMeasureMode,EngTMInput \*inputNRTMData,EngTMOutput \*outputNRTMData)
* **makeNRTMParameterCa**(int nMeasureMode,EngTMInput \*inputNRTMData,EngTMOutput \*outputNRTMData,int nIndex)  
  input parameter를 DSP에 전달.
* **updateParameter**(int nMeasureMode,EngTMInput \*inputNRTMData,EngTMOutput \*outputNRTMData)  
  어플리케이션 자체에서 처리해야 할 일들을 처리???

### 각각의 measure 파일

* updateMeasure(void)  
  makeNRTMParameter(eMEAS\_TAE\_MIMO, inputNRTMData, outputNRTMData); 호출  
  p\_5GTMModule->measureProcess(eMEAS\_TAE\_MIMO, inputNRTMData, outputNRTMData); 호출  
  if(measureProcess가ok) -> updateMeasuredData(outputNRTMData);, updateParameter(eMEAS\_TAE\_MIMO,inputNRTMData,outputNRTMData); 호출  
  checkLimit();호출
* **updateMeasuredData**(EngTMOutput \*outputNRTMData)  
  measureProcess를 실행한 후 그 결과를 GUI에 전달하기 위해 result item 을 set한다.  
  내부적으로 처리해야 하는 기능을 이곳에 넣는다(예, time offset 차이 계산. peak 계산).
* **checkLimit**()  
  pass/fail을 판단하기 위한 기능.

### slot.cpp

* slotUpdateMeasure()  
  일정 시간마다 호출되며 주로 eACTCMD\_UPDATE가 호출되도록 한다. 그 외 auto scale이나 spurious range를 체크한다.

### Review

int signalAnalyzerTM::callActSubmodule(int nMeasMode, int nCmd, int nArg, std::string params)

## Config, Setup, Mode change



## Measure Update



## Measurement ID

### Action Item

ADD\_ACT()

### Config Item

### Result Item

### Useful Function

#### bool signalAnalyzerTM::initItemProperty()

단위 표시(keypad, 등등)

## Technology

Slot number와 sub frame number는 PvsT(Symbol)에 쓰는 입력값이지만 TM에는 필요 없다.

## Function

|  |  |  |  |
| --- | --- | --- | --- |
|  | function | description | remark |
| A |  |  |  |
| B |  |  |  |
| C |  |  |  |
| D |  |  |  |
| E |  |  |  |
| F |  |  |  |
| G |  |  |  |
| H |  |  |  |
| I |  |  |  |
| J |  |  |  |
| K |  |  |  |
| L |  |  |  |
| M |  |  |  |
| N |  |  |  |
| O |  |  |  |
| P |  |  |  |
| Q |  |  |  |
| R |  |  |  |
| S | sendMsgWarning  syncmeasurementMode  setPropertyMHz | Warning popup  …  주파수 입력창 단위 표시 |  |
| T |  |  |  |
| U |  |  |  |
| V |  |  |  |
| W |  |  |  |
| X |  |  |  |
| Y |  |  |  |
| Z |  |  |  |