Supported Test Vector Configurations

Table of contents

List of Figures

This release of Aerial cuBB currently supports the following test-vector configurations.

PUSCH

TC start	TC End	Description	TV Generated	cuPHY Pass	cuBB Pass
7201	7201	base	1	1	1
7202	7203	mcsTable	2	2	2
7204	7204	mcs	1	1	1
7205	7207	num of layers	3	3	3
7208	7208	rb0, Nrb	1	1	1
7209	7210	sym0	2	2	2
7211	7211	dmsr0	1	1	1
7212	7213	Nsym	2	2	2
7214	7214	SCID	1	1	1
7215	7215	BWP0, nBWP	1	1	1
7216	7216	RNTI	1	1	1
7217	7219	addPos	3	3	3
7220	7220	dataScId	1	1	1

7221	7222	maxLen	2	2	2
7223	7223	dmrsScId	1	1	1
7224	7224	nCdm	1	1	1
7225	7225	port0	1	1	1
7227	7227	nAnt=2	1	1	1
7228	7228	nAnt=16	1	1	0
7229	7229	slotIdx	1	1	1
7230	7232	rvIdx	3	3	3
7233	7235	FDM	3	3	3
7236	7241	CDM	6	6	6
7242	7244	rvIdx>0/BGN=1	3	3	3
7245	7245	ulGridSize=106	1	1	0
7246	7247	dmrs_par per Ueg	2	2	2
7248	7250	additional FDM	3	3	3
7251	7257	precoding	7	7	7
7258	7260	mapping type B	3	3	3

7261	7272	Flexible DMRS ports	12	12	12
7273	7273	MCS > 28	1	1	1
7274	7279	additional nCDM=1	6	6	6
7280	7283	Flexible SLIV	4	4	4
7301	7320	multi-params	20	20	18
7321	7323	LBRM	3	3	3
7324	7326	HARQ-rx	3	3	0
7327	7330	8/16 UEs	4	4	4
7331	7338	multiple layers	8	8	8
7340	7340	Multi-layers with nAnt=16	1	1	0
7401	7403	CFO	3	3	3
7404	7406	TO	3	3	3
7407	7407	RSSI	1	1	1
7408	7408	CFO w/ SDM	1	1	1
7409	7409	TO w/ SDM	1	1	1
7410	7411	CEE-TDI	2	2	2

7412	7413	rx power	2	2	2
7414	7414	TDI maxLen = 2	1	1	1
7415	7417	small/big/zero rx	3	3	3
7418	7419	additional TDI	2	2	2
7420	7426	IRC=0	7	7	7
7427	7432	SINR meas	6	6	6
7501	7516	UCI on PUSCH (w/o data)	16	16	16
7517	7530	UCI on PUSCH (w/ data)	14	14	14
7531	7531	UciOnPusch DTX	1	1	1
7532	7532	UciOnPusch CRC fail	1	1	1
7533	7534	UciOnPusch addPos	2	2	2
7551	7570	UciOnPusch (multi-params)	20	20	20
7571	7575	UCI w/ and w/o data	5	5	5
7601	7613	FR1 BW mu = 1	13	13	13
7614	7621	FR1 BW mu = 0	8	8	0
7901	7901	demo_msg3	1	1	1

7902	7902	demo_traffic_ul	1	1	1
7903	7904	UciOnPusch conformance	0	0	0
7016	7153	sweep Zc/mcs (skip 7016,7017,7024,7025,7032,7039,7045,7057)	130	130	130

PUCCH

TC Start	TC End	Description	TV Generated	cuPHY Pass	cuBB Pass
6001	6003	bases for format 0	3	3	3
6004	6010	vary single parameter for format 0	7	7	7
6011	6040	vary multiple parameters for format 0	30	30	30
6041	6046	vary slotIdx (single-UCI) for format 0	6	6	6
6047	6056	multi-UCI tests for format 0	10	10	10
6057	6061	vary slotIdx (multi-UCI) for format 0	5	5	5
6101	6103	bases for format 1	3	3	3
6104	6116	vary single parameter for format 1	13	13	13
6117	6146	vary multiple parameters for format 1	30	30	30
6147	6155	vary slotIdx (single-UCI) for format 1	9	9	9
6156	6173	multi-UCI tests for format 1	18	18	18
6175	6192	TA estimation for format 1	18	18	18
6193	6194	192 UCI groups for format 1	2	2	2
6201	6203	bases for format 2	3	3	3
6204	6219	test Nf for format 2	16	16	16
6220	6235	test Nt and freq hopping for format 2	16	16	16
6236	6236	11 info bits and 2 PRBS for format 2	1	1	1
6239	6245	different payload sizes for format 2	7	7	7

6301	6310	bases for format 3	10	10	10
6311	6313	multi-UCI tests for format 3	3	3	3
6314	6324	tests with freqHop enabled for format 3	11	11	11
6325	6335	tests with freqHop disabled for format 3	11	11	11
6336	6346	tests with add'l DMRS postion, freqHop enabled for format 3	11	11	11
6347	6357	tests with add'l DMRS postion, freqHop disabled for format 3	11	11	11
6358	6364	different payload sizes for format 3	7	7	7
6365	6373	24-UCI tests for format 3	9	9	9
6501	6513	sweep different bandwidth for format 0, mu = 1	13	13	13
6514	6526	sweep different bandwidth for format 1, mu = 1	13	13	13
6527	6539	sweep different bandwidth for format 2, mu = 1	13	13	13
6540	6552	sweep different bandwidth for format 3, mu = 1	13	13	13
6553	6560	sweep different bandwidth for format 0, mu = 0	8	8	0
6561	6568	sweep different bandwidth for format 1, mu = 0	8	8	0
6569	6576	sweep different bandwidth for format 2, mu = 0	8	8	0
6577	6584	sweep different bandwidth for format 3, mu = 0	8	8	0
6585	6586	rx power for format 0	2	2	2
6587	6588	rx power for format 1	2	2	2
6589	6590	rx power for format 2	2	2	2
6591	6592	rx power for format 3	2	2	2

6593	6595	very small/very big/forcRxZero rx power for format 0	3	3	3
6596	6598	very small/very big/forcRxZero rx power for format 1	3	3	3
6599	6601	very small/very big/forcRxZero rx power for format 2	3	3	3
6602	6605	very small/very big/forcRxZero rx power for format 3	4	4	4
6801	6802	perf TV F08	2	2	2
6803	6804	perf TV F14	2	2	2

PRACH

TC Start	TC End	Description	TV Generated	cuPHY Pass	cuBB Pass
5001	5001	base	1	1	1
5002	5002	format 0	1	1	0
5003	5003	rootIdx	1	1	1
5004	5004	zoneIdx	1	1	1
5005	5005	prmbIdx	1	1	1
5006	5006	Nant	1	1	0
5007	5007	N_nc	1	1	1
5008	5008	delay	1	1	1
5009	5009	SNR	1	1	1
5010	5010	CFO	1	1	1
5011	5011	2-UE	1	1	1
5012	5012	4-UE	1	1	1
5013	5013	4FDM/16UE	1	1	1
5014	5018	rx power	5	5	5
5101	5101	FDD,mu=0,B4,nAnt=2	1	1	0

5102	5102	FDD,mu=1,B4,nAnt=4	1	1	1
5103	5103	TDD,mu=0,B4,nAnt=8	1	1	0
5104	5104	TDD,mu=1,B4,nAnt=16	1	1	0
5105	5105	FDD,mu=0,F0,nAnt=16	1	1	0
5106	5106	FDD,mu=1,F0,nAnt=8	1	1	0
5107	5107	TDD,mu=0,F0,nAnt=4	1	1	0
5108	5108	TDD,mu=1,F0,nAnt=2	1	1	0
5201	5213	FR1 BW mu = 1	13	13	13
5214	5221	FR1 BW mu = 0	8	8	0
5801	5802	perf TV F08	2	2	2
5803	5804	perf TV F14	2	2	0
5901	5901	demo_msg1	1	1	1
5911	5914	comformance TC	4	4	1

PDSCH

TC Start	TC End	Description	TV Generated	cuPHY Pass	cuBB Pass
3201	3201	base	1	1	1
3202	3203	mcsTable	2	2	2
3204	3204	mcs	1	1	1
3205	3207	num of layers	3	3	3
3208	3208	rb0, Nrb	1	1	1

3209	3210	sym0	2	2	2
3211	3211	dmrs0	1	1	1
3212	3213	Nsym	2	2	2
3214	3214	SCID	1	1	1
3215	3215	BWP0, nBWP	1	1	1
3216	3216	RNTI	1	1	1
3217	3219	addPos	3	3	3
3220	3220	dataScId	1	1	1
3221	3222	maxLen	2	2	2
3223	3223	dmrsScId	1	1	1
3224	3224	nCdm	1	1	1
3225	3225	port0	1	1	1
3226	3228	nAnt	3	3	3
3229	3229	slotIdx	1	1	1
3230	3232	rvIdx	3	3	3
3233	3235	FDM	3	3	3

3236	3241	SDM/SCID	6	6	6
3242	3244	rvIdx>0/BGN=1	3	3	3
3245	3245	dlGridSize=106	1	1	0
3246	3247	dmrs_par per Ueg	2	2	2
3248	3254	precoding	7	7	7
3255	3257	mapping type B	3	3	3
3258	3260	mixed precoding	3	3	3
3261	3261	refPoint	1	1	1
3262	3262	TxPower	1	1	1
3263	3263	modComp	1	0	0
3264	3264	precoding (mixed nPorts)	1	1	1
3265	3265	TxPower with 2 UEs	1	1	1
3266	3267	different rv	2	2	2
3268	3269	multi-layer	2	2	2
3271	3276	nCDM = 1	6	6	6
3321	3322	LBRM	2	2	2

3323	3333	RE map from CSI-RS	11	11	11
3334	3336	8/16 UEs (SU-MIMO)	3	3	3
3337	3337	16 UEs (MU-MIMO)	1	1	1
3401	3413	FR1 BW mu = 1	13	13	13
3414	3421	FR1 BW mu = 0	8	8	0
3901	3901	demo_coreset0	1	1	1
3902	3902	demo_msg2	1	1	1
3903	3903	demo_msg4	1	1	1
3904	3904	demo_traffic_dl	1	1	1
3001	3015	multi-params	15	15	15
3016	3154	sweep Zc/mcs (3016,3017,3024,3025,3032,3039,3045,3057 are skipped)	131	131	131

PDCCH

TC Start	TC End	Description	TV Generated	cuPHY Pass	cuBB Pass
2001	2001	base	1	1	1
2002	2002	slotIdx	1	1	1
2003	2003	nBWP	1	1	1
2004	2004	BPW0	1	1	1
2005	2005	sym0	1	1	1

2006	2007	Nsym	2	2	2
2008	2009	crstIdx	2	2	2
2010	2010	intl	1	1	1
2011	2012	nBndl	2	2	2
2013	2014	nIntl	2	2	2
2015	2015	nShift	1	1	1
2016	2016	isCSS	1	1	1
2017	2017	rnti	1	1	1
2018	2018	scrbId	1	1	1
2019	2019	scrbRnti	1	1	1
2020	2022	aggrL	3	3	3
2023	2023	dbQam	1	1	1
2024	2024	dbDmrs	1	1	1
2025	2025	Npayload	1	1	1
2026	2027	crstMap	2	2	2
2028	2028	nDCI	1	1	1
2029	2029	Npayload	1	1	1
2030	2030	aggrL	1	1	1
2031	2031	precoding	1	1	1
2032	2032	modComp	1	0	0
2033	2033	multi-PDCCH	1	1	1
2101	2112	multi-params	12	12	12
2201	2213	FR1 BW mu = 1	13	13	13
2214	2221	FR1 BW mu = 0	8	8	0
2801	2802	perf TV F14	2	2	2
2803	2804	perf TV F08	2	2	2
2805	2806	perf TV F09	2	2	2

2901	2901	demo_msg2	1	1	1
2902	2902	demo_msg4	1	1	1
2903	2903	demo_coreset0	1	1	1
2904	2904	demo_traffic_dl	1	1	1
2905	2905	demo_msg5	1	1	1

SS Block

TC Start	TC End	Description	TV Generated	cuPHY Pass	cuBB Pass
1001	1001	base	1	1	1
1002	1002	mu = 0	1	1	0
1003	1003	N_CELL_ID	1	1	1
1004	1004	n_hf = 1	1	1	1
1005	1005	L_max = 4	1	1	1
1006	1006	k_SSB	1	1	1
1007	1007	offsetPointA	1	1	1
1008	1008	SFN	1	1	1
1009	1009	blockIdx	1	1	1
1010	1010	precoding	1	1	1
1011	1011	betaPss	1	1	1
1101	1101	mu=0, 1SSB	1	1	0
1102	1102	mu=1, 1SSB	1	1	1
1103	1103	mu=1, 2SSB	1	1	0
1104	1104	mu=1, 2SSB	1	1	1
1202	1213	FR1 BW, mu = 1	12	12	12
1214	1221	FR1 BW, mu = 0	8	8	0
1801	1801	Perf TV	1	1	1
1901	1901	demo_ssb	1	1	1

1902	1902	for CP pipeline	1	1	1

CSI-RS

TC Start	TC End	Description	TV Generated	cuPHY Pass	cuBB Pass
4001	4004	nPorts = 1	4	4	4
4005	4007	nPorts = 2	3	3	3
4008	4009	nPorts = 4	2	2	2
4010	4012	nPorts = 8	3	3	3
4013	4038	nPorts > 8, row > 8	26	26	0
4039	4039	RB0	1	1	1
4040	4040	nRB	1	1	1
4041	4041	sym0	1	1	1
4042	4042	sym1	1	1	0
4043	4043	nID	1	1	1
4044	4044	power control	1	1	1
4045	4050	freqDomainAllocation	6	6	5
4051	4051	idxSlot	1	1	1
4052	4054	batching	3	3	3
4055	4055	small gird size	1	1	0
4056	4056	TRS	1	1	1
4057	4057	precoding	1	1	1
4058	4058	modComp	1	0	0
4059	4060	16/32 CSIRS PDUs	2	2	2
4101	4103	multiple parameters	3	3	3
4201	4213	FR1 BW mu = 1	13	13	13
4214	4221	FR1 BW mu = 0	8	8	0
4801	4801	perf TV F08	1	1	1

4802	4802	perf TV F09	1	1	1
4803	4803	perf TV F14	1	1	0

SRS

TC Start	TC End	Description	TV Generated	cuPHY Pass	cuBB Pass
8001	8001	base	1	1	1
8002	8002	rnti	1	1	1
8003	8003	Nap=2	1	1	1
8004	8004	Nap=4	1	1	1
8005	8005	nSym=2	1	1	1
8006	8006	nSym=4	1	1	1
8007	8007	Nrep=2	1	1	1
8008	8008	Nrep=4	1	1	1
8009	8009	sym0	1	1	1
8010	8010	cfgIdx	1	1	1
8011	8011	seqId	1	1	1
8012	8012	bwIdx=1	1	1	1
8013	8013	bwIdx=2	1	1	1
8014	8014	bwIdx=3	1	1	1
8015	8015	cmbSize	1	1	1
8016	8016	cmbOffset	1	1	1
8017	8017	cyclic shift	1	1	1
8018	8018	freqPosition	1	1	1
8019	8019	freqShift	1	1	1
8020	8020	freqHopping=1	1	1	1
8021	8021	freqHopping=2	1	1	1
8022	8022	freqHopping=3	1	1	1

8023	8023	grpSeqHopping=1	1	1	1
8024	8024	grpSeqHopping=2	1	1	1
8025	8025	rsrcType,Tsrs,Toffset	1	1	0
8026	8026	idxSlot	1	1	1
8027	8033	multi-SRS	1	1	1
8034	8034	rsrcType,Tsrs,Toffset	1	1	0
8035	8035	16 users wideband	1	1	1
8051	8057	multiple parameters	7	7	7
8101	8164	sweep cfgIdx	64	64	64
8201	8213	FR1 BW mu=1	13	13	13
8214	8221	FR1 BW mu=0	8	8	0
8222	8226	rx power	5	5	5
8227	8227	additional BW	1	1	1
8301	8302	SRS integration	2	2	2
8401	8415	32 nAnt	15	15	15
8420	8421	32 nAnt	2	2	2
8501	8524	64 nAnt	24	24	24
8801	8801	F09 perf TV	1	1	1
8802	8802	20M perf TV	1	1	1

mSlot_mCell

TC Start	TC End	Description	TV Generated	cuPHY Pass	cuBB Pass
90001	90007	single channel	7	7	7
90011	90012	dlmix/ulmix	2	2	2
90013	90015	s-slot	3	3	3
90016	90018	multi-cell base case	3	3	3
90019	90019	prcd+noPrcd	1	1	1

90020	90020	BFP14+BFP9	1	1	1
90021	90022	HARQ	2	2	2
90023	90023	empty slot	1	1	1
90032	90037	multi-slot combo TC	6	6	6
90041	90046	SRS + UL + DL	6	6	6
90051	90056	mixed cells	6	6	6
90057	90058	adaptive re-tx	2	2	2
90060	90060	SRS even/odd frames	1	1	1
90501	90505	bug TCs	5	5	5
90601	90603	multi-channel TCs	3	3	3

LDPC Performance

The `ldpc\_perf\_collect.py` Python script from the cuPHY repository can be used to perform error rate tests for the cuPHY LDPC decoder. There are test input files defined for Z = [64, 128, 256, 384], BG = [1,2]. The tests check whether the block error rate (BLER, also sometimes referred to as Frame Error Rate or FER) is less than 0.1.

From the build directory, the following commands run the tests:

```
../util/ldpc/ldpc\_perf\_collect.py --mode test -i
../util/ldpc/test/ldpc\_decode\_BG1\_Z64\_BLER0.1.txt -f -w 800 -P
../util/ldpc/ldpc\_perf\_collect.py --mode test -i
../util/ldpc/test/ldpc\_decode\_BG1\_Z128\_BLER0.1.txt -f -w 800 -P
../util/ldpc/ldpc\_perf\_collect.py --mode test -i
../util/ldpc/test/ldpc\_decode\_BG1\_Z256\_BLER0.1.txt -f -w 800 -P
../util/ldpc/ldpc\_perf\_collect.py --mode test -i
../util/ldpc/test/ldpc\_decode\_BG1\_Z384\_BLER0.1.txt -f -w 800 -P
../util/ldpc/ldpc\_perf\_collect.py --mode test -i
../util/ldpc/test/ldpc\_decode\_BG2\_Z64\_BLER0.1.txt -f -w 800 -P
../util/ldpc/ldpc\_perf\_collect.py --mode test -i
../util/ldpc/test/ldpc\_decode\_BG2\_Z128\_BLER0.1.txt -f -w 800 -P
../util/ldpc/ldpc\_perf\_collect.py --mode test -i
```

```
../util/ldpc/test/ldpc\_decode\_BG2\_Z256\_BLER0.1.txt -f -w 800 -P
../util/ldpc/ldpc\_perf\_collect.py --mode test -i
../util/ldpc/test/ldpc\_decode\_BG2\_Z384\_BLER0.1.txt -f -w 800 -P
```

Each test input file contains multiple tests for different code rates, as specified by the number of parity nodes. The format of the input files has the following form:

```
# BG Z num_parity num_iter SNR max_BER max_BLER #---------------------------------------
---------- 1 384 4 10 6.87 1 0.1 1 384 5 10 6.15 1 0.1 1 384 6 10 5.64 1 0.1 1 384
7 10 5.17 1 0.1 1 384 8 10 4.79 1 0.1 ...
```

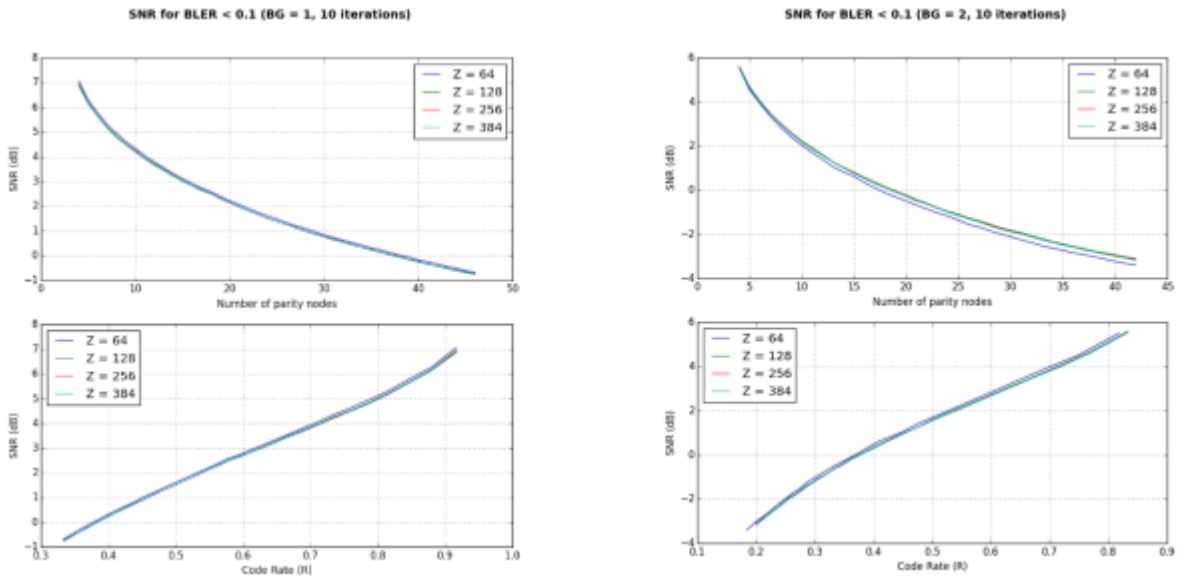
After running each of the test cases, the `ldpc\_perf\_collect.py` script

displays an output table:

```
+-----------------------------------------------------------------------------------------------------+ | # BG
Z num\_parity num\_iter SNR max\_BER BER max\_BLER BLER STATUS | | | | 1 384 4
10 6.870 1.000000e+00 4.833980e-04 1.000000e-01 8.750000e-02 PASS | | | | 1 384
5 10 6.150 1.000000e+00 1.481120e-04 1.000000e-01 7.250000e-02 PASS | | | | 1
384 6 10 5.640 1.000000e+00 5.652230e-05 1.000000e-01 8.000000e-02 PASS | | | |
1 384 7 10 5.170 1.000000e+00 7.886480e-05 1.000000e-01 8.750000e-02 PASS | | |
| 1 384 8 10 4.790 1.000000e+00 1.673470e-04 1.000000e-01 8.375000e-02 PASS | |
| | 1 384 9 10 4.480 1.000000e+00 1.185190e-04 1.000000e-01 7.625000e-02 PASS |
| | | 1 384 10 10 4.200 1.000000e+00 8.552320e-05 1.000000e-01 8.875000e-02
PASS | | | | 1 384 11 10 3.920 1.000000e+00 5.385890e-05 1.000000e-01
8.375000e-02 PASS | | | | 1 384 12 10 3.660 1.000000e+00 1.234020e-04
1.000000e-01 9.125000e-02 PASS | | | | 1 384 13 10 3.450 1.000000e+00
7.013490e-05 1.000000e-01 8.000000e-02 PASS | | | | 1 384 14 10 3.220
1.000000e+00 7.620150e-05 1.000000e-01 8.125000e-02 PASS | | | | 1 384 15 10
3.020 1.000000e+00 5.800190e-05 1.000000e-01 7.250000e-02 PASS | | | | 1 384 16
10 2.830 1.000000e+00 8.774270e-05 1.000000e-01 8.375000e-02 PASS | | | | 1 384
17 10 2.640 1.000000e+00 4.838420e-05 1.000000e-01 7.750000e-02 PASS | | | | 1
384 18 10 2.500 1.000000e+00 3.950640e-05 1.000000e-01 7.875000e-02 PASS | | |
| 1 384 19 10 2.310 1.000000e+00 3.551140e-05 1.000000e-01 8.375000e-02 PASS |
| | | 1 384 20 10 2.150 1.000000e+00 2.500590e-05 1.000000e-01 8.500000e-02
```

```
PASS | | | | 1 384 21 10 1.980 1.000000e+00 3.181230e-05 1.000000e-01
7.625000e-02 PASS | | | | 1 384 22 10 1.810 1.000000e+00 3.299600e-05
1.000000e-01 8.000000e-02 PASS | | | | 1 384 23 10 1.670 1.000000e+00
2.618960e-05 1.000000e-01 9.125000e-02 PASS | | | | 1 384 24 10 1.530
1.000000e+00 3.136840e-05 1.000000e-01 7.875000e-02 PASS | | | | 1 384 25 10
1.400 1.000000e+00 2.663350e-05 1.000000e-01 8.375000e-02 PASS | | | | 1 384 26
10 1.270 1.000000e+00 3.255210e-05 1.000000e-01 8.625000e-02 PASS | | | | 1 384
27 10 1.140 1.000000e+00 2.692950e-05 1.000000e-01 7.500000e-02 PASS | | | | 1
384 28 10 0.999 1.000000e+00 5.149150e-05 1.000000e-01 9.250000e-02 PASS | | |
| 1 384 29 10 0.889 1.000000e+00 3.225620e-05 1.000000e-01 8.750000e-02 PASS |
| | | 1 384 30 10 0.772 1.000000e+00 3.536340e-05 1.000000e-01 9.375000e-02
PASS | | | | 1 384 31 10 0.650 1.000000e+00 4.113400e-05 1.000000e-01
9.125000e-02 PASS | | | | 1 384 32 10 0.547 1.000000e+00 3.965440e-05
1.000000e-01 8.750000e-02 PASS | | | | 1 384 33 10 0.428 1.000000e+00
5.489460e-05 1.000000e-01 9.625000e-02 PASS | | | | 1 384 34 10 0.333
1.000000e+00 5.030780e-05 1.000000e-01 8.875000e-02 PASS | | | | 1 384 35 10
0.220 1.000000e+00 3.906250e-05 1.000000e-01 8.875000e-02 PASS | | | | 1 384 36
10 0.127 1.000000e+00 2.929690e-05 1.000000e-01 8.250000e-02 PASS | | | | 1 384
37 10 0.034 1.000000e+00 3.225620e-05 1.000000e-01 9.000000e-02 PASS | | | | 1
384 38 10 -0.066 1.000000e+00 2.737330e-05 1.000000e-01 8.375000e-02 PASS | |
| | 1 384 39 10 -0.170 1.000000e+00 2.722540e-05 1.000000e-01 8.500000e-02
PASS | | | | 1 384 40 10 -0.253 1.000000e+00 3.521540e-05 1.000000e-01
7.500000e-02 PASS | | | | 1 384 41 10 -0.344 1.000000e+00 5.563450e-05
1.000000e-01 9.375000e-02 PASS | | | | 1 384 42 10 -0.424 1.000000e+00
2.559780e-05 1.000000e-01 8.750000e-02 PASS | | | | 1 384 43 10 -0.515
1.000000e+00 4.690460e-05 1.000000e-01 9.500000e-02 PASS | | | | 1 384 44 10
-0.605 1.000000e+00 5.755800e-05 1.000000e-01 9.125000e-02 PASS | | | | 1 384
45 10 -0.693 1.000000e+00 3.980230e-05 1.000000e-01 8.000000e-02 PASS | | | | 1
384 46 10 -0.766 1.000000e+00 5.208330e-05 1.000000e-01 9.875000e-02 PASS | |
| | 43 TESTS PASSED, 0 TESTS FAILED | +----------------------------------------------------------
-------------------------------------------+
```

Plots of current SNR values used for BLER testing are shown below:



SHM Logger Configuration

The Shared Memory (SHM) configuration file is located at `./cuPHY/nvlog/config/nvlog_config.yaml`. It contains the following:

> # nvlog config # name: can see log file at /dev/shm/${name}.log and /tmp/${name}.log # primary: In all processes logging to the same file, set the first starting porcess to be primary, set others to be secondary. # Log levels: -1 - LOG_NONE, 0 - LOG_ERROR, 1 - LOG_CONSOLE, 2 - LOG_WARN, 3 - LOG_INFO, 4 - LOG_DEBUG, 5 - LOG_VERBOSE # prefix_opts: refer to nvlog.h # b7 - CPU core number that the caller thread is running on # b6 - Caller thread name # b5 - Caller thread ID # b4 - Global 64-bit sequence number # b3 - Log level: 'E', 'C', 'W', 'I', 'D', 'V' (Error, Console, Warning, Info, Debug, Verbose) # b2 - Module type: 'P', 'S', 'U' (Primary, Secondary, Unknown) # b1 - Date and Time: 1970-01-01 00:00:00.000000 # b0 - Time Only: 00:00:00.000000 nvlog: name: phy shm_log_level: 3 # SHM log level console_log_level: 1 # Console log level max_file_size_bits: 28 # Size = 2 ^ bits: The rotating log file /var/log/aerial/${name}.log size shm\_cache\_size\_bits: 21 # Size = 2 ^ bits: SHM cache file /dev/shm/${name}.log size log_buf_size: 1024 # Max length of one time call of the nvlog API max_threads: 64 # The MAX thread count which using nvlog togehter save_to_file: 1 # Whether to save to a file cpu_core_id: -1 # CPU core ID for the background log saving thread.-1 means not bind. prefix_opts: 0x89 # Log prefix of each line max_tag_len: 32 # Max 31

printable characters max_tag_num: 1024 # Max tag number nvlog_tags: - 0: "" # Reserve number 0 for no tag print shm_level: 3 # Example: overlay shm_log_level for a tag console_level: 1 # Example: overlay console_log_level for a tag - 10: "NVLOG" # nvlog - 11: "NVLOG.TEST" - 12: "NVLOG.ITAG"

> **Note**
>
> Max_threads specifies the maximum number of process threads. The default value is 64 threads. To improve performance, the lock-free mechanism doesn't use Compare-And-Swap, but does configure a maximum thread number.

String Tags

To print with a string tag, define a string TAG in code and use it directly:

```
#define STAG "NVLOG.STAG" NVLOGI(STAG, "This is STAG C printf style log. level=%d\\n",LOG\_INFO); NVSLOGI(STAG) << "This is STAG C++ stream style log. level=" <<LOG\_INFO << "\\n";
```

Pre-Defined Integer Tags

To print with a pre-defined integer tag, define an integer TAG in `nvlog\_config.yaml`:

```
nvlog\_tags: - 10: "NVLOG" # nvlog - 11: "NVLOG.TEST" - 12: "NVLOG.ITAG"
```

Then use the integer TAG in code:

```
#define NVLOG\_TAG\_BASE\_NVLOG 10 // nvlog #define ITAG (NVLOG\_TAG\_BASE\_NVLOG + 2) // "NVLOG.ITAG" NVLOGI(ITAG, "This is ITAG C printf style log. level=%d\\n", LOG\_INFO); NVSLOGI(ITAG) << "This is ITAG C++ stream style log. level=" << LOG\_INFO << "\\n";
```