5G Transport Channels and HARQ

Radha Krishna Ganti

L1 interface to MAC

38.211
38.212

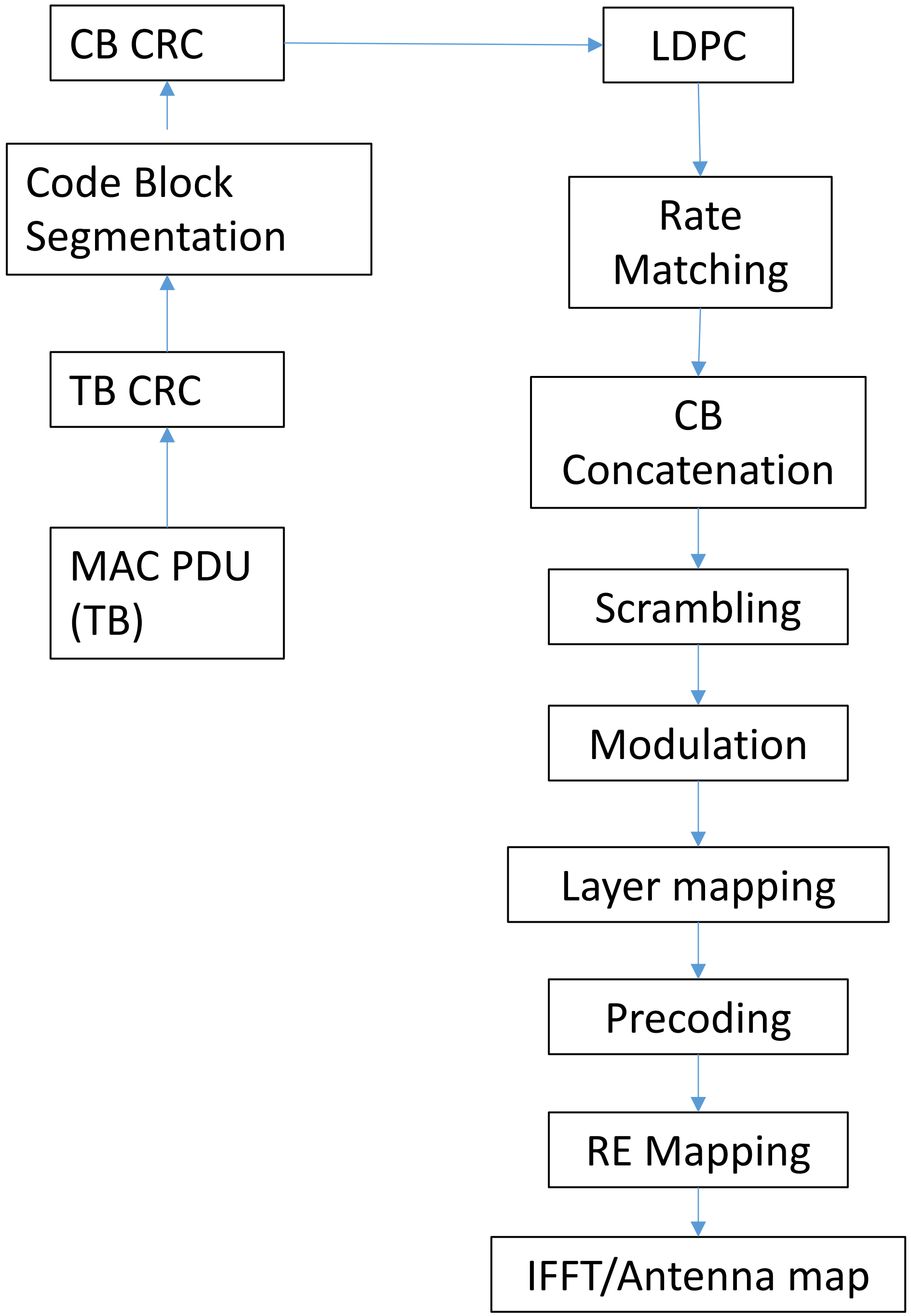
- Transport Channels
  - DLSCH
  - PCH
  - BCH
- Transport time interval (TTI)
  - Up to two transport blocks
    - Dynamic size
    - Two TB are used only for more than 4 layers

Table 4.2-1

TrCH	Physical Channel
DL-SCH	PDSCH
BCH	PBCH
PCH	PDSCH

Table 4.2-2

Control information	Physical Channel
DCI	PDCCH


CB CRC
LDPC
Code Block Segmentation
Rate Matching
TB CRC
CB Concatenation
MAC PDU (TB)
Scrambling
Modulation
Layer mapping
Precoding
RE Mapping
IFFT/Antenna map

Transport Blocks

- Mac PDUs
- Transport block is per UE.
- For 4 layers
  - 1 transport block
- For >4 layers
  - Two transport blocks

- What is the size of a transport block
  - Million $ question: We will answer later

38.212 [7.2.1]

Transport Block CRC (for PDSCH channel)

- Depends on the transport block size A (bits)
- If A>3824
  - Use a 24 bit CRC
  - $g_{CRC24A}(D) = [D^{24} + D^{23} + D^{18} + D^{17} + D^{14} + D^{11} + D^{10} + D^{7} + D^{6} + D^{5} + D^{4} + D^{3} + D + 1]$
  - Recall Parity: P(D) = Rem(M(D)* D^{24}, $g_{CRC24A}(D)$)
    - M(D) = Message polynomial
- Else
  - Use a 16 bit CRC
  - $g_{CRC16}(D)=[D^{16} + D^{12} + D^{5} + 1]$

You take the reminder of the bit string with the CRC polynomial and append the reminder to the input bit string.

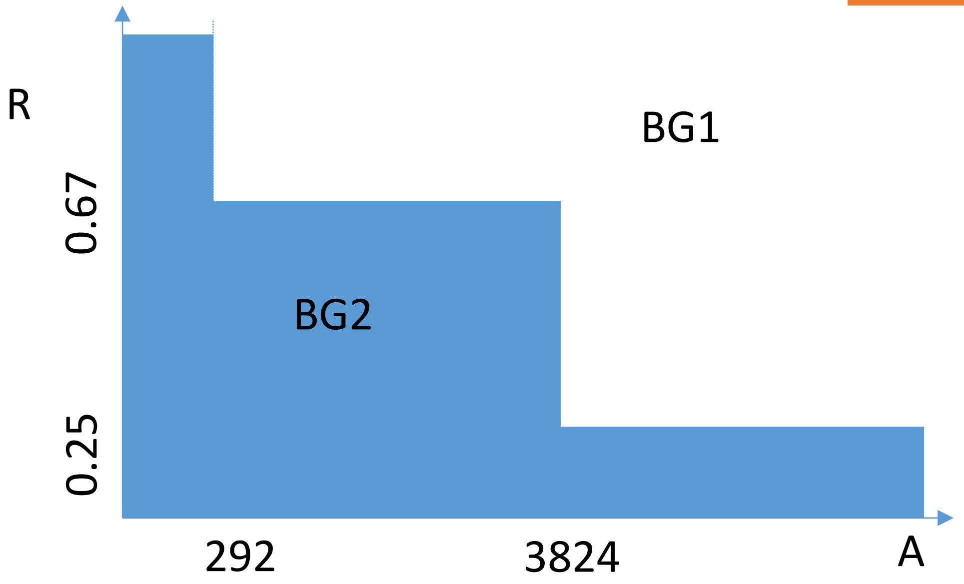
The encoding is performed in a systematic form, which means that in GF(2), the polynomial:

$$a_0 D^{A+L-1} + a_1 D^{A+L-2} + \ldots + a_{A-1} D^{L} + p_0 D^{L-1} + p_1 D^{L-2} + \ldots + p_{L-2} D^{1} + p_{L-1}$$

Input Length: A
Output Length A+L

Base graph Selection

- LDPC encoding
- LDPC encoder can support two input sizes (code block size)
  - 8448 bits for base graph 1
  - 3840 bits for base graph 2
- Two base graphs
  - BG1: 1/3 to 22/24
  - BG2: 1/5 to 5/6
- The rate can be increased through puncturing
- The BG is selected, and the code block (after CRC addition) is encoded by the appropriate LDPC encoder



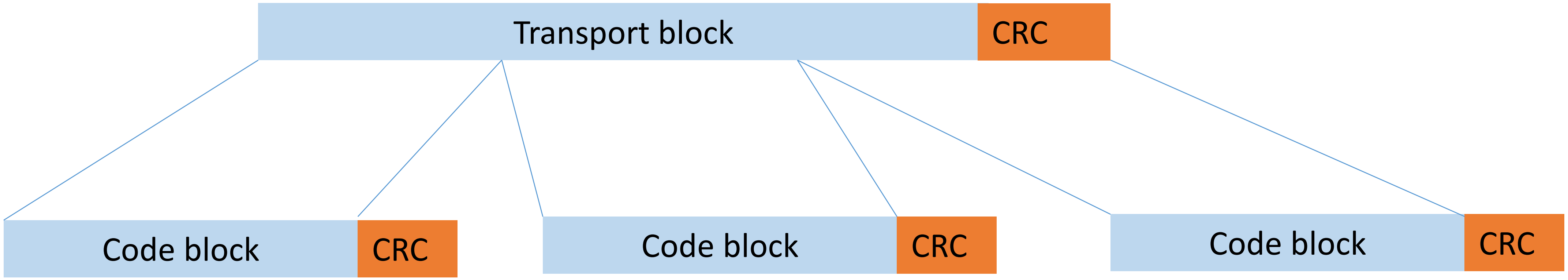
Let R be the code rate

- If A≤ 292, or if A≤ 3824 and R ≤0.67, or if R ≤ 0.25: BG2 is used
- Otherwise BG1 is used

Recall A is the TB size

Code Block segmentation

1. LDPC encoder can support two input sizes (code block size)
   1. 8448 bits for base graph 1
   2. 3840 bits for base graph 2
2. If input TB is larger than the maximum code block size, the input is segmented and 24 bit CRC is added



A

CBS (cont ...)

- Kcb = maximum code block size
  - For BG1: Kcb = 8448
  - For BG2: Kcb = 3840
- If B<= Kcb (Recall B = A+ TB_CRC)
  - L = 0 (No CB-CRC)
  - #of codeblocks C = 1
  - B’ = B
- If B > Kcb
  - L = 24 (CRC 24, Polynomial CRC24B polynomial)
  - # of codeblock C = ceil(B/(kcb-L));
  - B’ = B+ C*L

- Now compute the size of input to LDPC (for each code block)
- Let Kr be the number of bits in each codeblock

$K' = B'/C$;

For LDPC base graph 1,

$K_b = 22$.

For LDPC base graph 2,

if $B > 640$

$K_b = 10$;

elseif $B > 560$

$K_b = 9$;

elseif $B > 192$

$K_b = 8$;

else

$K_b = 6$;

end if

Find the minimum value of Z in the lifting sets such that

- Kb*Z > K'

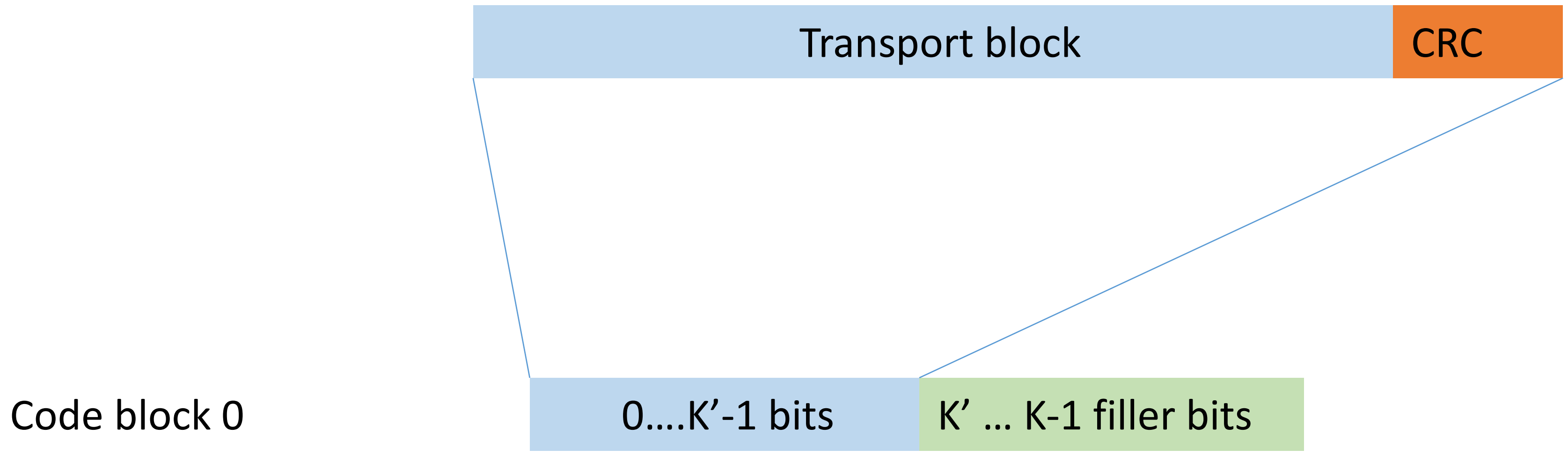
- For BG1
  - Set Kr = 22 Z (46zx68z is the size of parity check)
- For BG2
  - Set Kr = 10 Z (42zx52z is the of the parity check)

Observe that Kr= K is the same across all code words.

How is K' an integer?

(Will be ensured by the TB size. Later we will see the TB size computation)

Forming the Codeblocks (for C==1). Remember No CB CRC

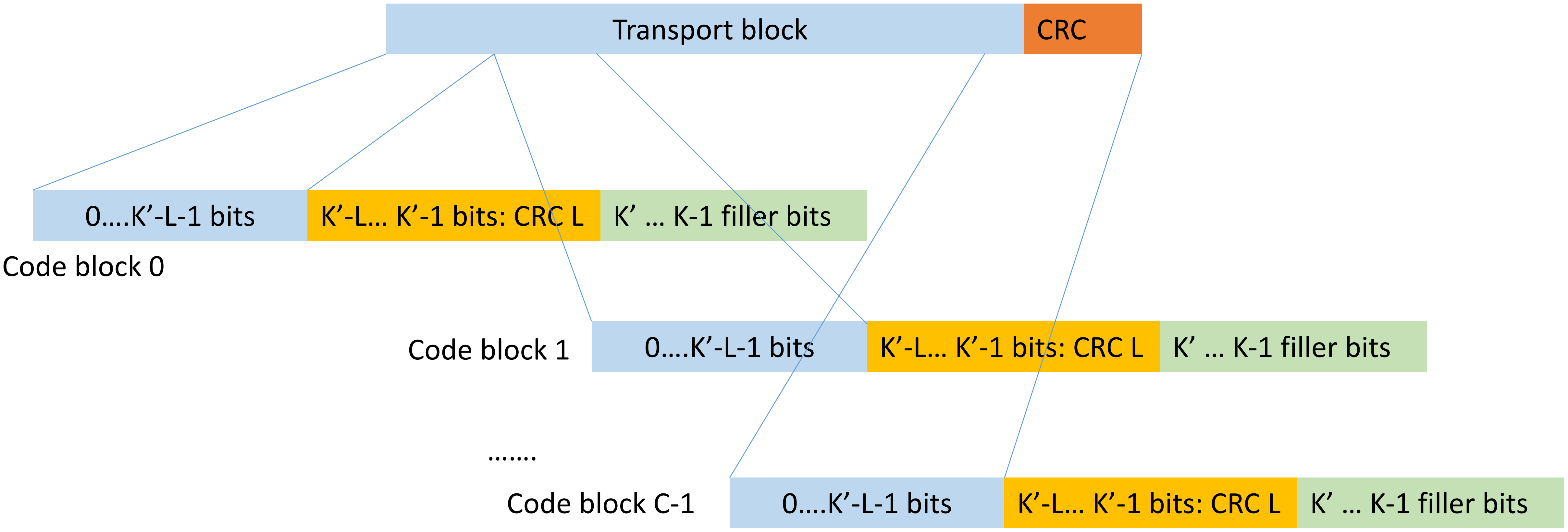


What are filler bits?
Something that the LDPC understands
Will be discarded after encoding.

Forming the Codeblocks (for C>1)

All the codeblocks of a given TB have the same size!
- CRC size (and #of bits): L
- # of Filler bits = K-K': "Null"
- # of information bits = K'-L
- Final Length of CB = (K'-L) + (K-K') + L = K
- # of Codeblocks = C

Recall that CRC with L=24 is used



- Is K-K' <=Z ??

Primer to LDPC codes

Prof. Andrew Thangaraj lectures

1. Introduction to block codes (Important parameters)
2. LDPC code construction
3. LDPC codes in 5G
4. Encoding LDPC in 5G

LDPC Codes in 5G

- Based on the photograph
- Defined by the parity check matrix
- The base graph(s) are sub-sets of the two base graphs
  - BG1 and BG2
- The two codes have a base rate(s) of 1/3 and 1/5.

Parity Matrix construction

- Remember that we have already chosen Z (lifting factor or expansion factor) in the code-block segmentation step.
- So we also get the set index iLS from the lifting factor from table 5.3.2-1.
- The Base graph H_{BG} dimensions are
  - BG1: 46 x 68
  - BG2: 42 x 52
- The non-zero entries of the base matrices are given in Tables 5.3.2-2 and 5.3.2-3
  - You choose the correct column based on iLS
  - All the other entries in the base graph are assumed to be zero.

Set index (i_{LS})	Set of lifting sizes (Z)
0	{2, 4, 8, 16, 32, 64, 128, 256}
1	{3, 6, 12, 24, 48, 96, 192, 384}
2	{5, 10, 20, 40, 80, 160, 320}
3	{7, 14, 28, 56, 112, 224}
4	{9, 18, 36, 72, 144, 288}
5	{11, 22, 44, 88, 176, 352}
6	{13, 26, 52, 104, 208}
7	{15, 30, 60, 120, 240}

BG1

$\mathbf{H}_{BG}$		$V_{i,j}$							
Row index i	Column index j	Set index i_{LS}							
		0	1	2	3	4	5	6	7
0	0	250	307	73	223	211	294	0	135
	1	69	19	15	16	198	118	0	227
	2	226	50	103	94	188	167	0	126
	3	159	369	49	91	186	330	0	134
	5	100	181	240	74	219	207	0	84
	6	10	216	39	10	4	165	0	83
	9	59	317	15	0	29	243	0	53
	10	229	288	162	205	144	250	0	225
	11	110	109	215	216	116	1	0	205
	12	191	17	164	21	216	339	0	128
	13	9	357	133	215	115	201	0	75
	15	195	215	298	14	233	53	0	135
	16	23	106	110	70	144	347	0	217
	18	190	242	113	141	95	304	0	220
	19	35	180	16	198	216	167	0	90
	20	239	330	189	104	73	47	0	105
	21	31	346	32	81	261	188	0	137
	22	1	1	1	1	1	1	0	1
	23	0	0	0	0	0	0	0	0
1	0	2	76	303	141	179	77	22	96
	2	239	76	294	45	162	225	11	236
	3	117	73	27	151	223	96	124	136
	4	124	288	261	46	256	338	0	221
	5	71	144	161	119	160	268	10	128
	7	222	331	133	157	76	112	0	92
	8	104	331	4	133	202	302	0	172
	9	173	178	80	87	117	50	2	56
	11	220	295	129	206	109	167	16	11
	12	102	342	300	93	15	253	60	189
	14	109	217	76	79	72	334	0	95
	15	132	99	266	9	152	242	6	85
	16	142	354	72	118	158	257	30	153
	17	155	114	83	194	147	133	0	87
	19	255	331	260	31	156	9	168	163
	21	28	112	301	187	119	302	31	216
	22	0	0	0	0	0	0	105	0
	23	0	0	0	0	0	0	0	0
	24	0	0	0	0	0	0	0	0
	0	106	205	68	207	258	226	132	189
	1	111	250	7	203	167	35	37	4
	2	185	328	80	31	220	213	21	225
	4	63	332	280	176	133	302	180	151
	5	117	256	38	180	243	111	4	236

$\mathbf{H}_{BG}$		$V_{i,j}$						
Row index i	Column index j	Set index i_{LS}						
		0	1	2	3	4	5	6
15	1	96	2	290	120	0	348	6
	10	65	210	60	131	183	15	81
	13	63	318	130	209	108	81	182
	18	75	55	184	209	68	176	53
	25	179	269	51	81	64	113	46
	37	0	0	0	0	0	0	0
16	1	64	13	69	154	270	190	88
	3	49	338	140	164	13	293	198
	11	49	57	45	43	99	332	160
	20	51	289	115	189	54	331	122
	22	154	57	300	101	0	114	182
	38	0	0	0	0	0	0	0
17	0	7	260	257	56	153	110	91
	14	164	303	147	110	137	228	184
	16	59	81	128	200	0	247	30
	17	1	358	51	63	0	116	3
	21	144	375	228	4	162	190	155
	39	0	0	0	0	0	0	0
18	1	42	130	260	199	161	47	1
	12	233	163	294	110	151	286	41
	13	8	280	291	200	0	246	167
	18	155	132	141	143	241	181	68
	19	147	4	295	186	144	73	148
	40	0	0	0	0	0	0	0
19	0	60	145	64	8	0	87	12
	1	73	213	181	6	0	110	6
	7	72	344	101	103	118	147	166
	8	127	242	270	198	144	258	184
	10	224	197	41	8	0	204	191
	41	0	0	0	0	0	0	0
20	0	151	187	301	105	265	89	6
	3	186	206	162	210	81	65	12
	9	217	264	40	121	90	155	15
	11	47	341	130	214	144	244	5
	22	160	59	10	183	228	30	30
	42	0	0	0	0	0	0	0
21	1	249	205	79	192	64	162	6
	5	121	102	175	131	46	264	86
	16	109	328	132	220	266	346	96
	20	131	213	283	50	9	143	42
	21	171	97	103	106	18	109	199
	43	0	0	0	0	0	0	0
	0	64	30	177	53	72	280	44

The matrix $\mathbf{H}$ is obtained by replacing each element of $\mathbf{H}_{\text{BG}}$ with a $Z_c \times Z_c$ matrix, according to the following:

- Each element of value 0 in $\mathbf{H}_{\text{BG}}$ is replaced by an all zero matrix $\mathbf{0}$ of size $Z_c \times Z_c$;
- Each element of value 1 in $\mathbf{H}_{\text{BG}}$ is replaced by a circular permutation matrix $\mathbf{I}(P_{i,j})$ of size $Z_c \times Z_c$, where i and j are the row and column indices of the element, and $\mathbf{I}(P_{i,j})$ is obtained by circularly shifting the identity matrix $\mathbf{I}$ of size $Z_c \times Z_c$ to the right $P_{i,j}$ times. The value of $P_{i,j}$ is given by $P_{i,j} = \text{mod}(V_{i,j}, Z_c)$. The value of $V_{i,j}$ is given by Tables 5.3.2-2 and 5.3.2-3 according to the set index i_{LS} and LDPC base graph.

So the size of the H matrix is

- BG1: 46Z x 68Z
- BG2: 42Z x 52Z

Recall the size of the parity matrix of a (N',K) block code is is (N'-K) x N'

Also from CBS, the input sizes are

- BG1: K= 22Z
- BG2: K= 10Z

<u>Before puncturing</u>

BG1: (68Z, 22Z) LDPC code
BG2: (52Z, 10Z) LDPC code

LDPC bit input and output BG1

Output of CBS

$c_0, c_1, \ldots, c_{K-1}$

Input 22Z bits

Filler bits (K-K') in the CBS are replaced by zeros.

$c_0, c_1, \ldots, c_{K'-1}, 0, 0, \ldots 0$

LDPC code

Output 68Z bits

$f_0, f_1, \ldots, f_{N'-1} = [c_0, c_1, \ldots, c_{2Z-1}, c_{2Z}, \ldots, c_{K'-1}, 0, 0 \ldots, 0, P_0, P_1, P_{68Z-K-1}]$, $N' = 68Z$

Discarded/punctured 2Z bits

Punctured and cleaned output (Filler bits re-introduced): 66Z bits

$d_0, d_1, \ldots, d_{N-1} = [c_{2Z}, \ldots, c_{K'-1}, Null, Null, \ldots, Null, P_0, P_1, P_{68Z-K-1}]$, $N = 66Z$

After puncturing

Input: 22Z

Output: N= 66Z

Effective Code rate: 1/3

#of filler bits in output = K-K'

38.212

LDPC bit input and output BG2

Output of CBS
10 Z bits
$c_0, c_1, \ldots, c_{K-1}$

Filler bits (K-K') in the CBS are replaced by zeros. Still 10 Z bits
$c_0, c_1, \ldots, c_{K'-1}, 0, 0, ..0$

LDPC code

<u>Output bits: 52Z bits</u>
$f_0, f_1, \ldots, f_{N'-1} = [c_0, c_1, .., c_{2Z-1}, c_{2Z}, \ldots, c_{K'-1}, 0, 0.., 0, P_0, P_1, P_{52Z-K-1}]$, N' = 52Z

<u>Punctured and cleaned output (Filler bits re-introduced)</u>
$d_0, d_1, \ldots, d_{N-1} = [c_{2Z}, \ldots, c_{K'-1}, \text{Null}, \text{Null}, .., \text{Null}, P_0, P_1, P_{52Z-K-1}]$, N=50Z

<u>After puncturing</u>

Input: 10Z
Output: 50Z

Effective Code rate: 1/5
#of filler bits in output = K-K'

Rate Matching

- Used to make sure that the output of the FEC (all blocks) fit well (and snugly) in the resource grid allocated to the users
- Let G be the number of bits that can fit in the REs allocated to a user
  - You make sure you remove all the DMRS locations, CSI-RS, overlaps with SSBlocks
  - So G also includes the Modulation (Qm) and no of Layers (NL).
- If you divide G into C number of codewords, then what you get per code word is
  - G/C (need not be an integer)
- You also want the codeword to have exact bits to be divided equally among all the layers

Rate Matching contd..

- So each codeword should contain??
  - Nl*Qm*Ceil(G/(C*Nl*Qm))
  - But the problem is C* Nl*Qm*Ceil(G/(C*Nl*Qm)) is not equal to G
  - So what do you do
    - 3*Ceil(10/3) neq 10.
    - Floor(10/3)+floor(10/3)+Ceil(10/3) =10
    - So you have 3 – mod(10,3) floors and mod(10,3) ceils.
- Generalizing the above we have
  - Let KK = C - mod(G/ Nl*Qm,C).
  - Then KK* Nl*Qm*floor(G/(C*Nl*Qm)) + mod(G/ Nl*Qm,C)* Nl*Qm*ceil(G/(C*Nl*Qm)) = G

E_r = Rate making length of the r-th coded block.

Set $j = 0$

for $r = 0$ to $C - 1$

if the r -th coded block is not scheduled for transmission as indicated by CBGTI according to Subclause 5.1.7.2 for DL-SCH and 6.1.5.2 for UL-SCH in [6, TS 38.214]

$E_r = 0$;

else

if $j \leq C' - \mathrm{mod}(G/(N_L \cdot Q_m), C') - 1$

$$E_r = N_L \cdot Q_m \cdot \left\lfloor \frac{G}{N_L \cdot Q_m \cdot C'} \right\rfloor;$$

else

$$E_r = N_L \cdot Q_m \cdot \left\lceil \frac{G}{N_L \cdot Q_m \cdot C'} \right\rceil;$$

end if

$j = j + 1$;

end if

end for

What is this sub-clause?

- N_L is the number of transmission layers that the transport block is mapped onto;
- Q_m is the modulation order;
- G is the total number of coded bits available for transmission of the transport block;
- $C' = C$ if CBGTI is not present in the DCI scheduling the transport block and C' is the number of scheduled code blocks of the transport block if CBGTI is present in the DCI scheduling the transport block.

What is the relation between N and Er ? Is E_r >= N?

See that Er is a multiple of modulation order and Layers.

Rate Matching: Circular buffer

$$N_{cb} = \min\left(N, N_{ref}\right) \text{ otherwise, where } N_{ref} = \left\lfloor \frac{TBS_{\text{LBRM}}}{C \cdot R_{\text{LBRM}}} \right\rfloor$$

- N_{ref} is a large number and we will look at it later
  - Most cases, N will be smaller than N_{ref}
- N_{cb} is the size of the circular buffer
- $e_0, e_1, e_2, \ldots, e_{E-1}$ is the output of the rate matching
  - Not same across all the bits

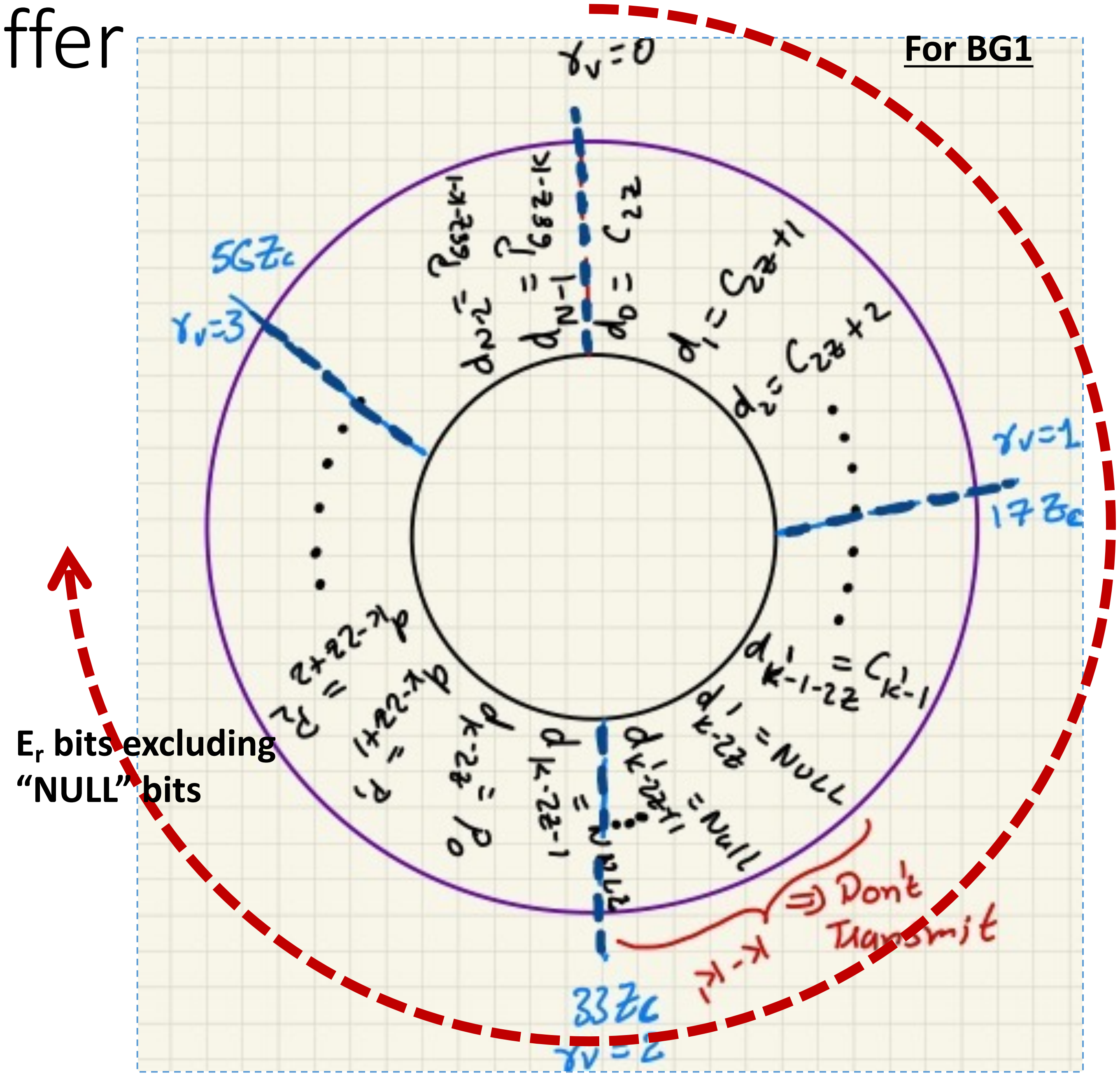


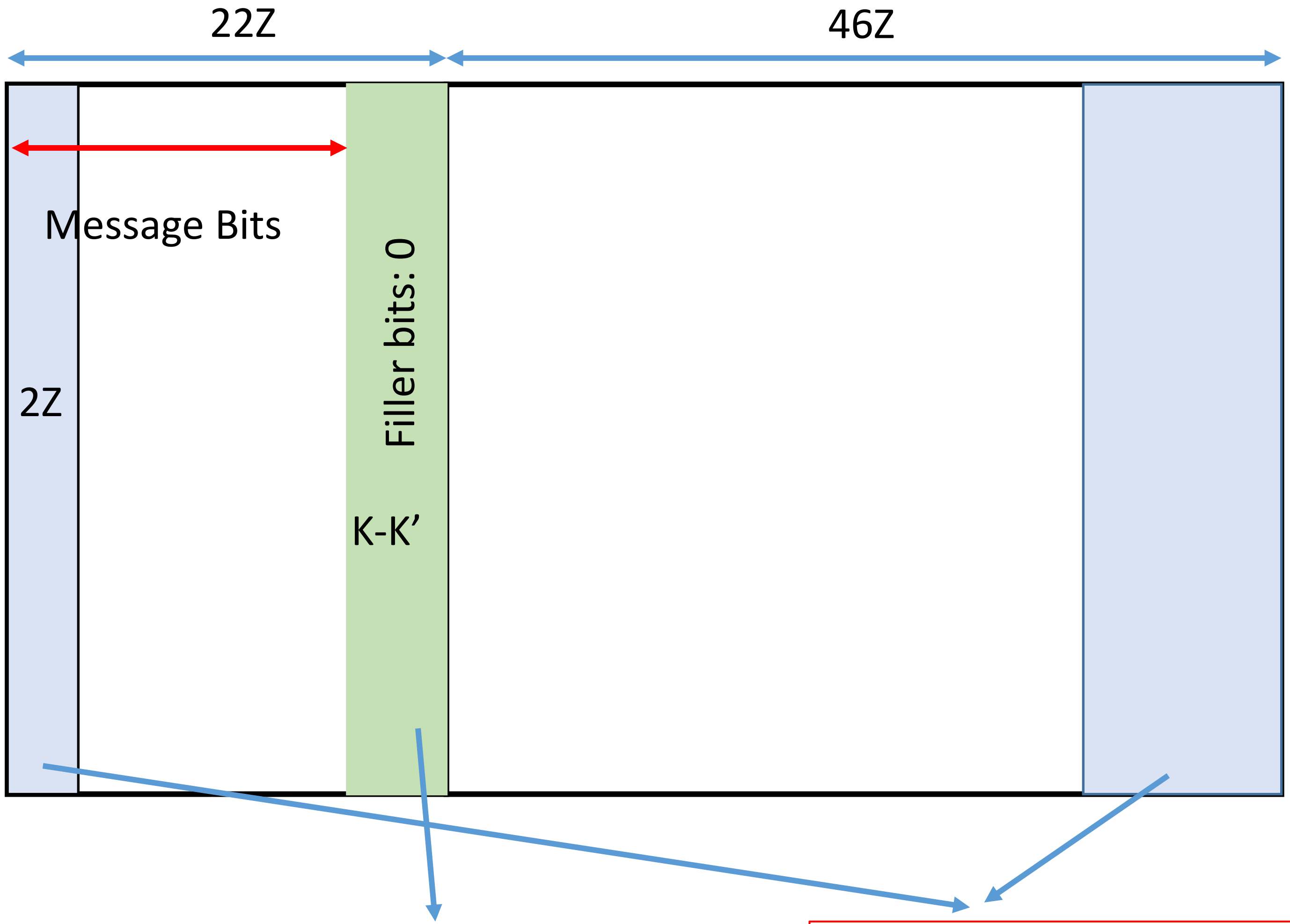
Table 5.4.2.1-2: Starting position of different redundancy versions, k_0

rv_{id}	k_0	
	LDPC base graph 1	LDPC base graph 2
0	0	0
1	$\left\lfloor \frac{17N_{cb}}{66Z_c} \right\rfloor Z_c$	$\left\lfloor \frac{13N_{cb}}{50Z_c} \right\rfloor Z_c$
2	$\left\lfloor \frac{33N_{cb}}{66Z_c} \right\rfloor Z_c$	$\left\lfloor \frac{25N_{cb}}{50Z_c} \right\rfloor Z_c$
3	$\left\lfloor \frac{56N_{cb}}{66Z_c} \right\rfloor Z_c$	$\left\lfloor \frac{43N_{cb}}{50Z_c} \right\rfloor Z_c$

Rate matching comments

- For lower rates (most probably less than 1/3 for BG1 and 1/5 for BG2)
  - You might have to go round the circle
    - This implies you transmit the same message and parity bits more than once
- For higher rates, you might transmit fewer than N bits
  - And hence might not complete the *entire* circle even once

Shortening and puncturing (Optional slide)



Total # of bits required = E
Message bits = K’
Transmitted message bits: K’-2Z
Transmitted Parity bits = E- (K’-2Z)

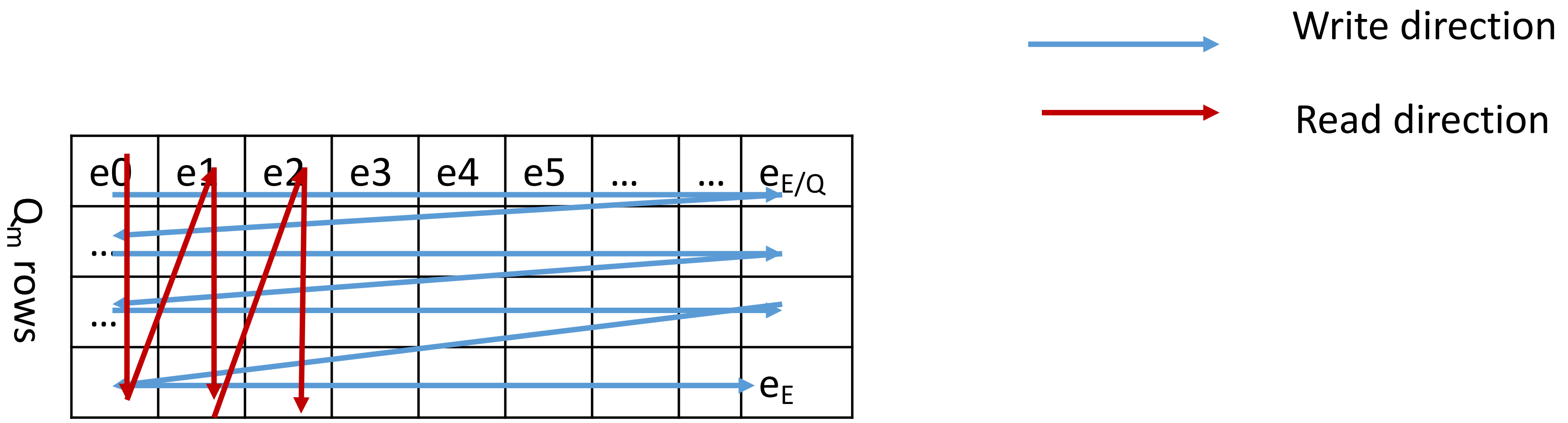
<u>At the RX:</u>

- The punctured bits LLR is set to 0 (probability ½ of being 1 and ½ of being 0)
- The shortened bits LLR is set to the LLR corresponding to bit zero

<u>Shortening:</u> Set to zero for encoding, but discarded later

<u>Puncturing:</u>
Actual message bits used and partiy bits computed
But Not transmitted

Bit Interleaving



On the output of rate matcher (code-block wise)
Q_m: Modulation order

- Write row wise in a rectangle of Q_m X E/Q_m
- Read column wise

Output bits are denoted by $f_0, f_1, \ldots f_E$

Helps avoid burst errors in codeblocks.

Code-block concatenation

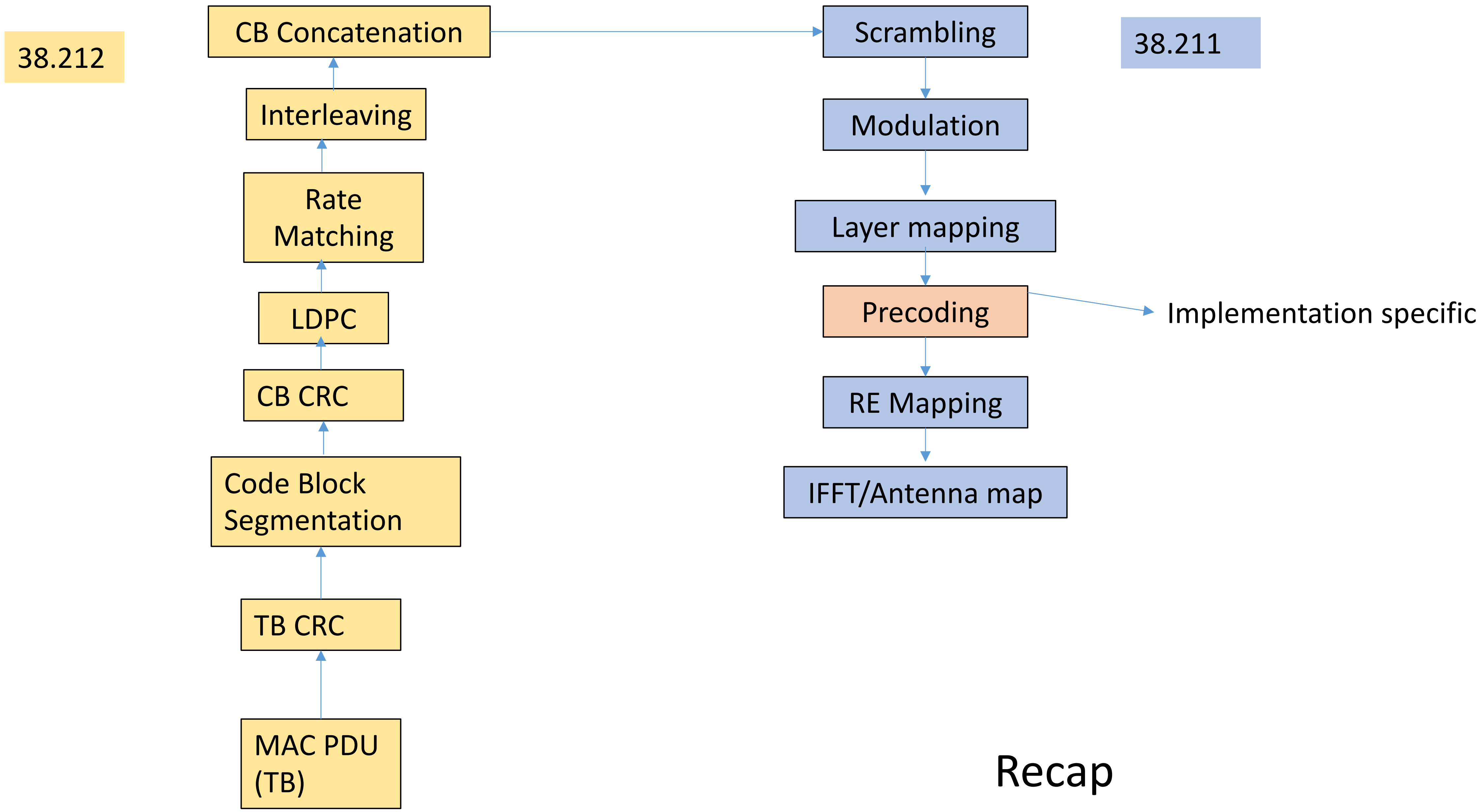
- Concatenate all the code blocks (outputs of bit interleaving) serially.
- Output size: G
- Sequence denoted by g_k

Transport block size determination by the receiver

38.214
5.1.3.2

- The transport size computation is done to approximately fill the RE grid (across layers) taking into consideration the rate to be used.
  - This value can be computed both by the UE and the BS
  - So only DMRS is considered
    - CSI-RS, SRS, PRACH and SSB are not considered as these happen very infrequently.
  - Both UE and gNB can compute the TB size using the using the knowledge of allocation and DMRS.

- The exact filling of the RE is taken care of by the circular buffer through the value G
  - G considers all the overheads.


38.212
CB Concatenation
Interleaving
Rate Matching
LDPC
CB CRC
Code Block Segmentation
TB CRC
MAC PDU (TB)
Scrambling
38.211
Modulation
Layer mapping
Precoding
Implementation specific
RE Mapping
IFFT/Antenna map

Recap

Scrambling

- Up to two Transport blocks can be transmitted
  - Indicated by q=0 or q=1
  - Two code words for more than 4 layers
- The codeword is scrambled by a gold sequence
  - Initialization: $c_{\text{init}} = n_{\text{RNTI}} \cdot 2^{15} + q \cdot 2^{14} + n_{\text{ID}}$
- n_{ID}: Data scrambling identity
  - Comes from the PUSCH-CONFIG IE
  - Takes values from 0 – 1023
- RNTI: Radio Network Temporary Identifiers
  - C-RNTI
- MAC/L1-controller figures out the exact RNTI to be used

$$\tilde{b}^{(q)}(i) = \left(b^{(q)}(i) + c^{(q)}(i)\right) \bmod 2$$

RNTI	Usage	Transport Channel	Logical Channel
P-RNTI	Paging and System Information change notification	PCH	PCCH
SI-RNTI	Broadcast of System Information	DL-SCH	BCCH
RA-RNTI	Random Access Response	DL-SCH	N/A
Temporary C-RNTI	Contention Resolution (when no valid C-RNTI is available)	DL-SCH	CCCH
Temporary C-RNTI	Msg3 transmission	UL-SCH	CCCH, DCCH, DTCH
C-RNTI, MCS-C-RNTI	Dynamically scheduled unicast transmission	UL-SCH	DCCH, DTCH
C-RNTI	Dynamically scheduled unicast transmission	DL-SCH	CCCH, DCCH, DTCH
MCS-C-RNTI	Dynamically scheduled unicast transmission	DL-SCH	DCCH, DTCH
C-RNTI	Triggering of PDCCH ordered random access	N/A	N/A
CS-RNTI	Configured scheduled unicast transmission (activation, reactivation and retransmission)	DL-SCH, UL-SCH	DCCH, DTCH
CS-RNTI	Configured scheduled unicast transmission (deactivation)	N/A	N/A
TPC-PUCCH-RNTI	PUCCH power control	N/A	N/A
TPC-PUSCH-RNTI	PUSCH power control	N/A	N/A
TPC-SRS-RNTI	SRS trigger and power control	N/A	N/A
INT-RNTI	Indication pre-emption in DL	N/A	N/A
SFI-RNTI	Slot Format Indication on the given cell	N/A	N/A
SP-CSI-RNTI	Activation of Semi-persistent CSI reporting on PUSCH	N/A	N/A
NOTE: The usage of MCS-C-RNTI is equivalent to that of C-RNTI in MAC procedures (except for the C-RNTI MAC CE).			

Value (hexa-decimal)	RNTI
0000	N/A
0001–FFEF	RA-RNTI, Temporary C-RNTI, C-RNTI, MCS-C-RNTI, CS-RNTI, TPC-PUCCH-RNTI, TPC-PUSCH-RNTI, TPC-SRS-RNTI, INT-RNTI, SFI-RNTI, and SP-CSI-RNTI
FFF0–FFFD	Reserved
FFFE	P-RNTI
FFFF	SI-RNTI

Modulation

- Mapped to the constellation (Downlink)

Modulation scheme	Modulation order Q_m
QPSK	2
16QAM	4
64QAM	6
256QAM	8

Layer Mapping

- Data is mapped to multiple layers
  - Spatial multiplexing
- One codeword is at most mapped to 4 layers

Number of layers	Number of codewords	Codeword-to-layer mapping $i = 0,1,\ldots,M_{\text{symb}}^{\text{layer}} - 1$	
1	1	$x^{(0)}(i) = d^{(0)}(i)$	$M_{\text{symb}}^{\text{layer}} = M_{\text{symb}}^{(0)}$
2	1	$x^{(0)}(i) = d^{(0)}(2i)$ $x^{(1)}(i) = d^{(0)}(2i+1)$	$M_{\text{symb}}^{\text{layer}} = M_{\text{symb}}^{(0)}/2$
3	1	$x^{(0)}(i) = d^{(0)}(3i)$ $x^{(1)}(i) = d^{(0)}(3i+1)$ $x^{(2)}(i) = d^{(0)}(3i+2)$	$M_{\text{symb}}^{\text{layer}} = M_{\text{symb}}^{(0)}/3$
4	1	$x^{(0)}(i) = d^{(0)}(4i)$ $x^{(1)}(i) = d^{(0)}(4i+1)$ $x^{(2)}(i) = d^{(0)}(4i+2)$ $x^{(3)}(i) = d^{(0)}(4i+3)$	$M_{\text{symb}}^{\text{layer}} = M_{\text{symb}}^{(0)}/4$

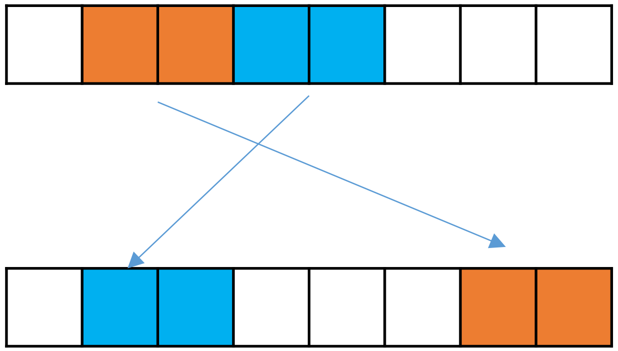
5	2	$x^{(0)}(i)=d^{(0)}(2i)$ $x^{(1)}(i)=d^{(0)}(2i+1)$ $x^{(2)}(i)=d^{(1)}(3i)$ $x^{(3)}(i)=d^{(1)}(3i+1)$ $x^{(4)}(i)=d^{(1)}(3i+2)$	$M_{\text{symb}}^{\text{layer}}=M_{\text{symb}}^{(0)}/2=M_{\text{symb}}^{(1)}/3$
6	2	$x^{(0)}(i)=d^{(0)}(3i)$ $x^{(1)}(i)=d^{(0)}(3i+1)$ $x^{(2)}(i)=d^{(0)}(3i+2)$ $x^{(3)}(i)=d^{(1)}(3i)$ $x^{(4)}(i)=d^{(1)}(3i+1)$ $x^{(5)}(i)=d^{(1)}(3i+2)$	$M_{\text{symb}}^{\text{layer}}=M_{\text{symb}}^{(0)}/3=M_{\text{symb}}^{(1)}/3$
7	2	$x^{(0)}(i)=d^{(0)}(3i)$ $x^{(1)}(i)=d^{(0)}(3i+1)$ $x^{(2)}(i)=d^{(0)}(3i+2)$ $x^{(3)}(i)=d^{(1)}(4i)$ $x^{(4)}(i)=d^{(1)}(4i+1)$ $x^{(5)}(i)=d^{(1)}(4i+2)$ $x^{(6)}(i)=d^{(1)}(4i+3)$	$M_{\text{symb}}^{\text{layer}}=M_{\text{symb}}^{(0)}/3=M_{\text{symb}}^{(1)}/4$
8	2	$x^{(0)}(i)=d^{(0)}(4i)$ $x^{(1)}(i)=d^{(0)}(4i+1)$ $x^{(2)}(i)=d^{(0)}(4i+2)$ $x^{(3)}(i)=d^{(0)}(4i+3)$ $x^{(4)}(i)=d^{(1)}(4i)$ $x^{(5)}(i)=d^{(1)}(4i+1)$ $x^{(6)}(i)=d^{(1)}(4i+2)$ $x^{(7)}(i)=d^{(1)}(4i+3)$	$M_{\text{symb}}^{\text{layer}}=M_{\text{symb}}^{(0)}/4=M_{\text{symb}}^{(1)}/4$

Resource Mapping (to Virtual Resource Blocks)

- Map the modulated symbols to resource elements on the time-frequency grid
- Decided by the scheduler
- Some RE are not available for PDSCH
  - DMRS, Co-Scheduled DMRS
  - CSI-RS
  - L1/L2 control signaling
  - Synchronisation signals (SSBlocks)
- Mapped first across frequency and then time
  - Latency reduction
- Resource map granularity
  - Later

VRB to Physical Resource blocks (PRBs) mapping

- Two types of mappings
  - Non-interleaved mapping
  - Interleaved mapping
- Non-interleaved mapping
  - Default case
  - PRB = VRB (mostly)
- Interleaved mapping
  - Frequency diversity
  - 2RB or 4RB interleaving
    - More details when we look at similar mapping in PDCCH.



Interleaved mapping

Precoder

- Not specified for the downlink
- Transparent to the user
  - DMRS
- Precoding matrix is suggested by the UE
  - From a codebook/ or not
  - It is up to the network to use it
- We will get back to this when discussing CSI