Upstream FEC Codeword Filling Based on Data Burst Size

Rich Prodan

Upstream Full FEC Codeword Sizes Previously Adopted

- **Long codeword:**
  - 16200 total bits
  - 14400 information bits including 40 CRC bits
  - 1800 parity bits
  - 88.9% code rate with 88.6% overall efficiency
- **Medium codeword:**
  - 5940 total bits
  - 5040 information bits including 40 CRC bits
  - 900 parity bits
  - 84.8% code rate with 84.2% overall efficiency
- **Short codeword:**
  - 1120 total bits
  - 840 information bits including 40 CRC bits
  - 280 parity bits
  - 75% code rate with 71.4% overall efficiency

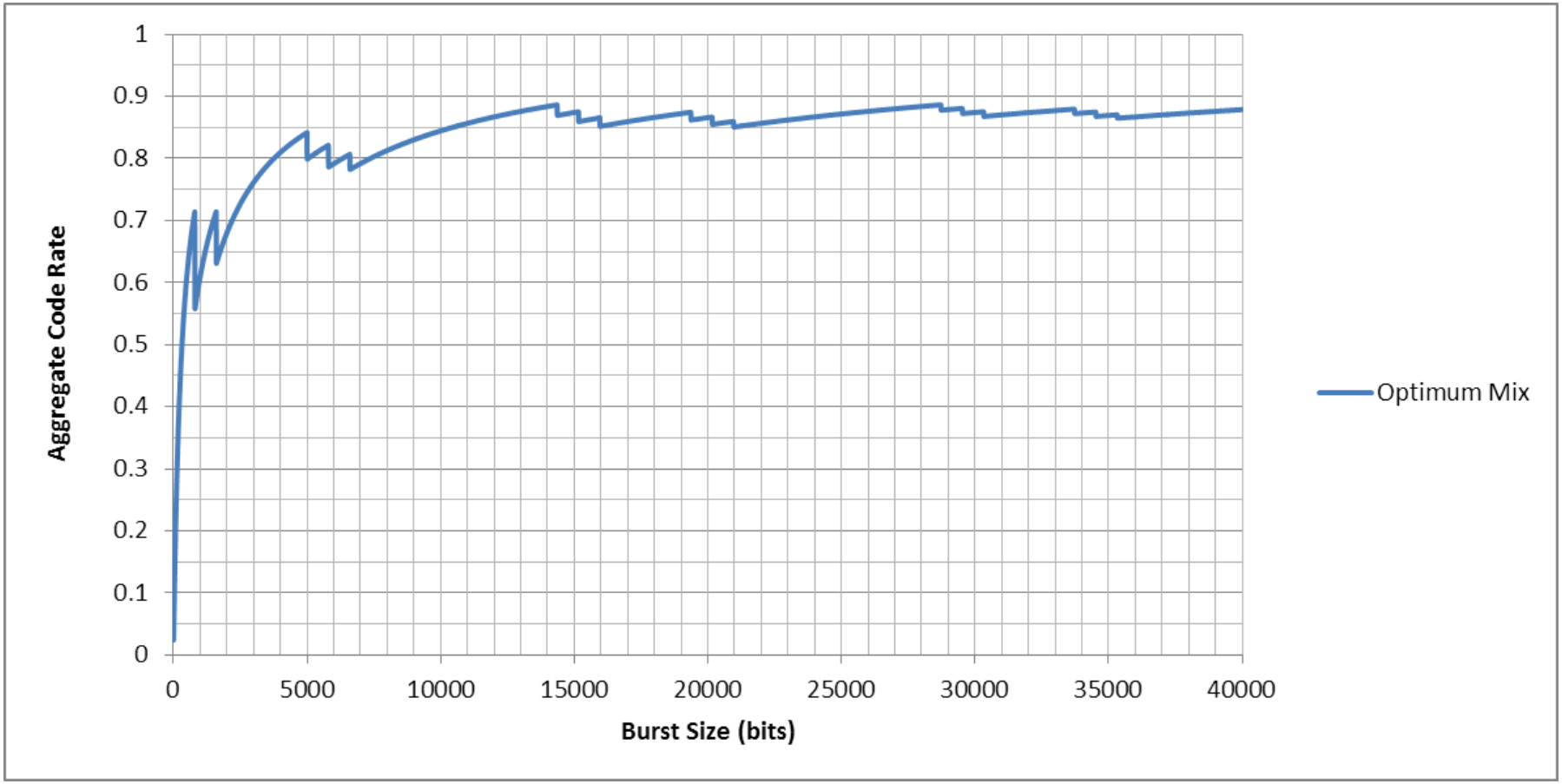
Basic Steps For Determining Codeword Sizes

- **Choice of codeword sizes is based entirely on data burst size**
  - Unambiguous at both ends
- **Details of algorithm, including codeword sizes and thresholds for shortening, are embedded in spec**
- **Basic steps for conversion in either encode or decode side:**
  - If there are enough bits to create a full long codeword, do so. Keep doing this until there aren't enough bits left.
  - If there are now enough bits to create a shortened long codeword (subject to the thresholds above), do so, and end the burst.
  - Otherwise, if there are enough bits to create a full medium codeword, do so. Keep doing this until there aren't enough bits left.
  - If there are now enough bits left to create a shortened medium codeword (subject to the thresholds above), do so, and end the burst.
  - Otherwise, if there are enough bits to create a full short codeword, do so. Keep doing this until there aren't enough bits left.
  - Use whatever bits remain to create a shortened short codeword, and end the burst.

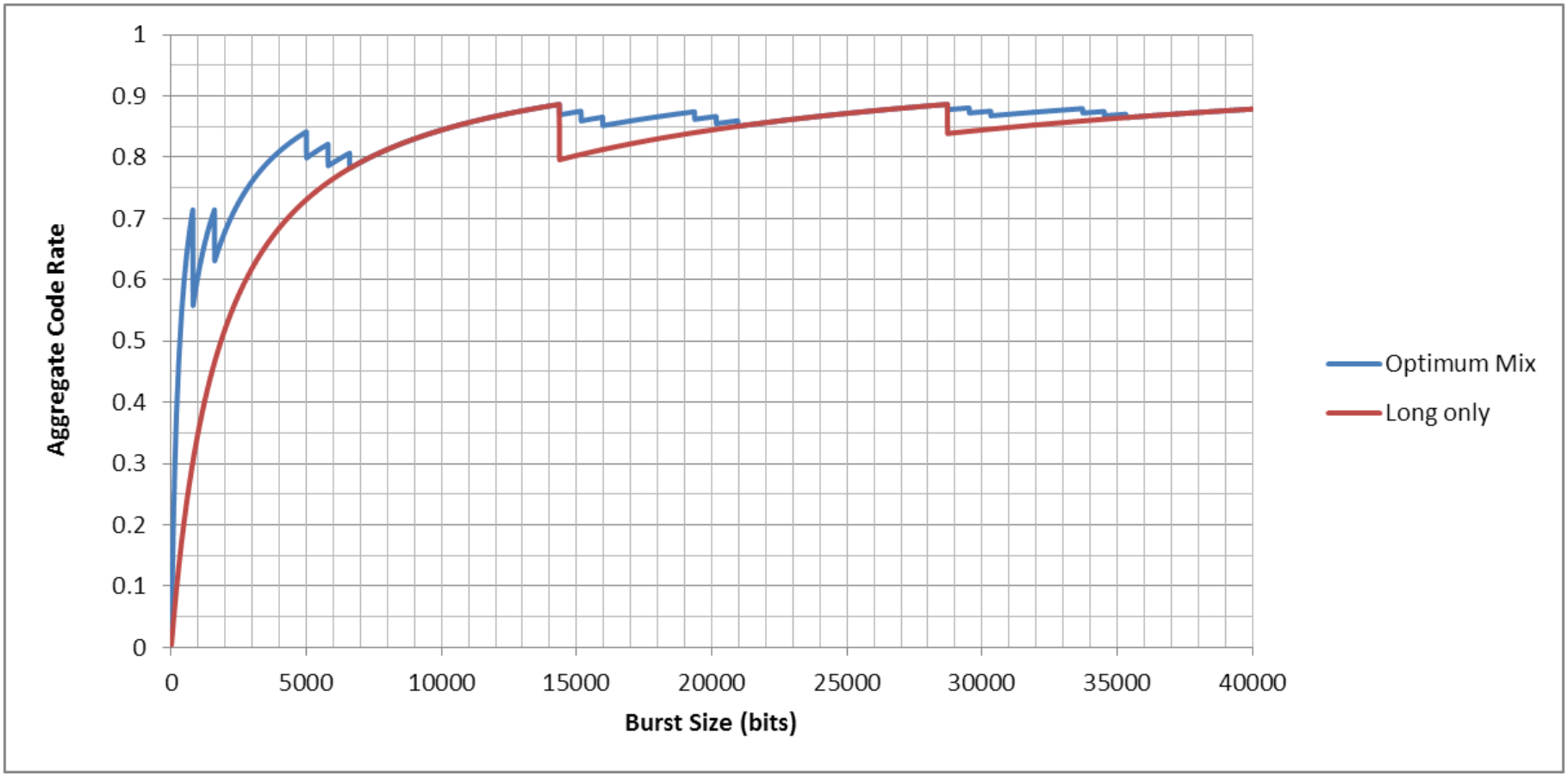
Optimum Shortening Solution

- **Upon completion of filling the last shortened short (N,K) codeword, check if this codeword is at least half full of information bits. That is, check if the last codeword contains at least K/2 information bits.**
- **If not, then move K/2 information bits from the next-to-last codeword into the shortened short last codeword.**
- **This results in two last shortened codewords where each is at least half full.**

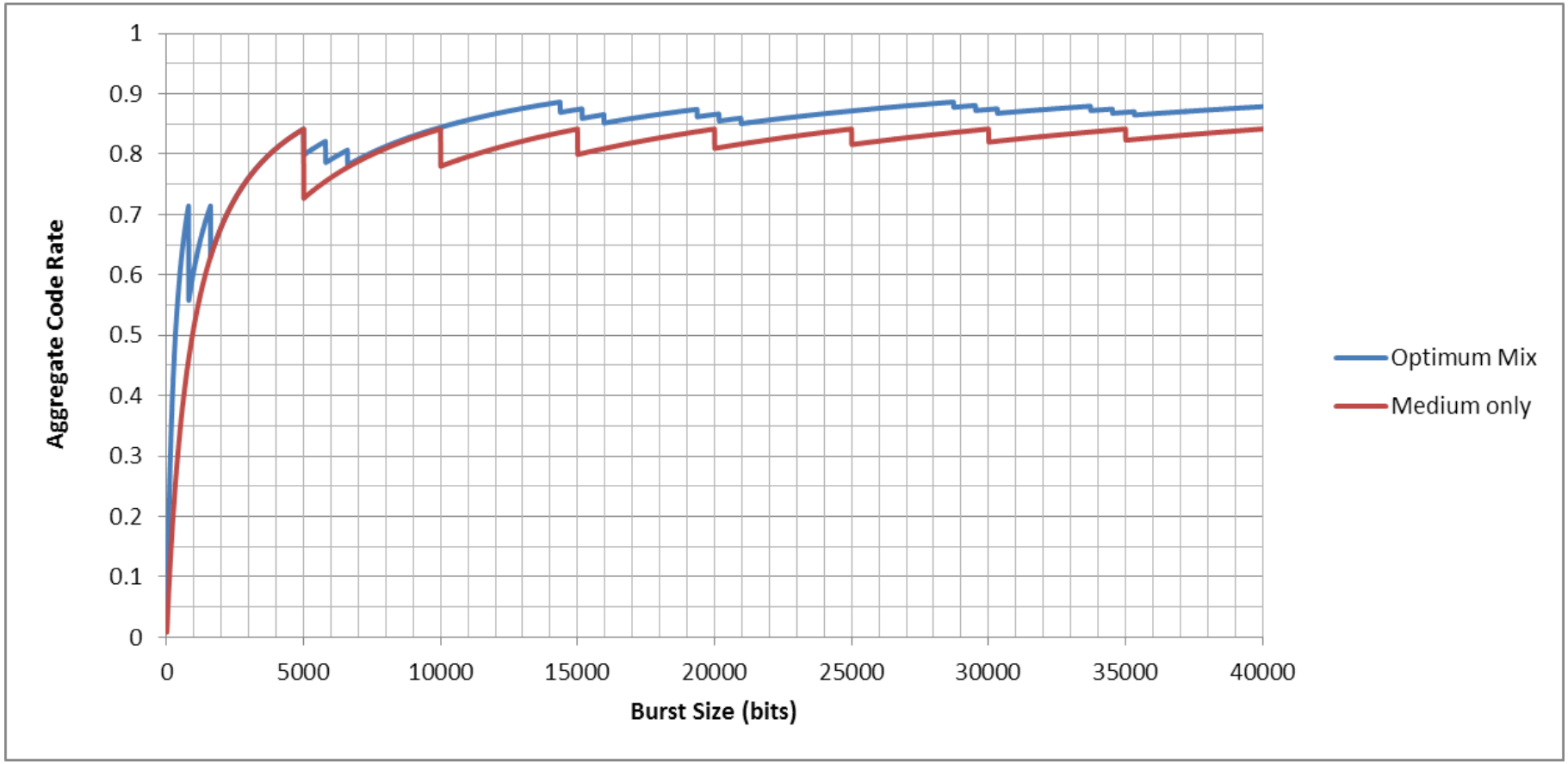
Efficiency (Code Rate) vs. Burst Size (Information Bits)



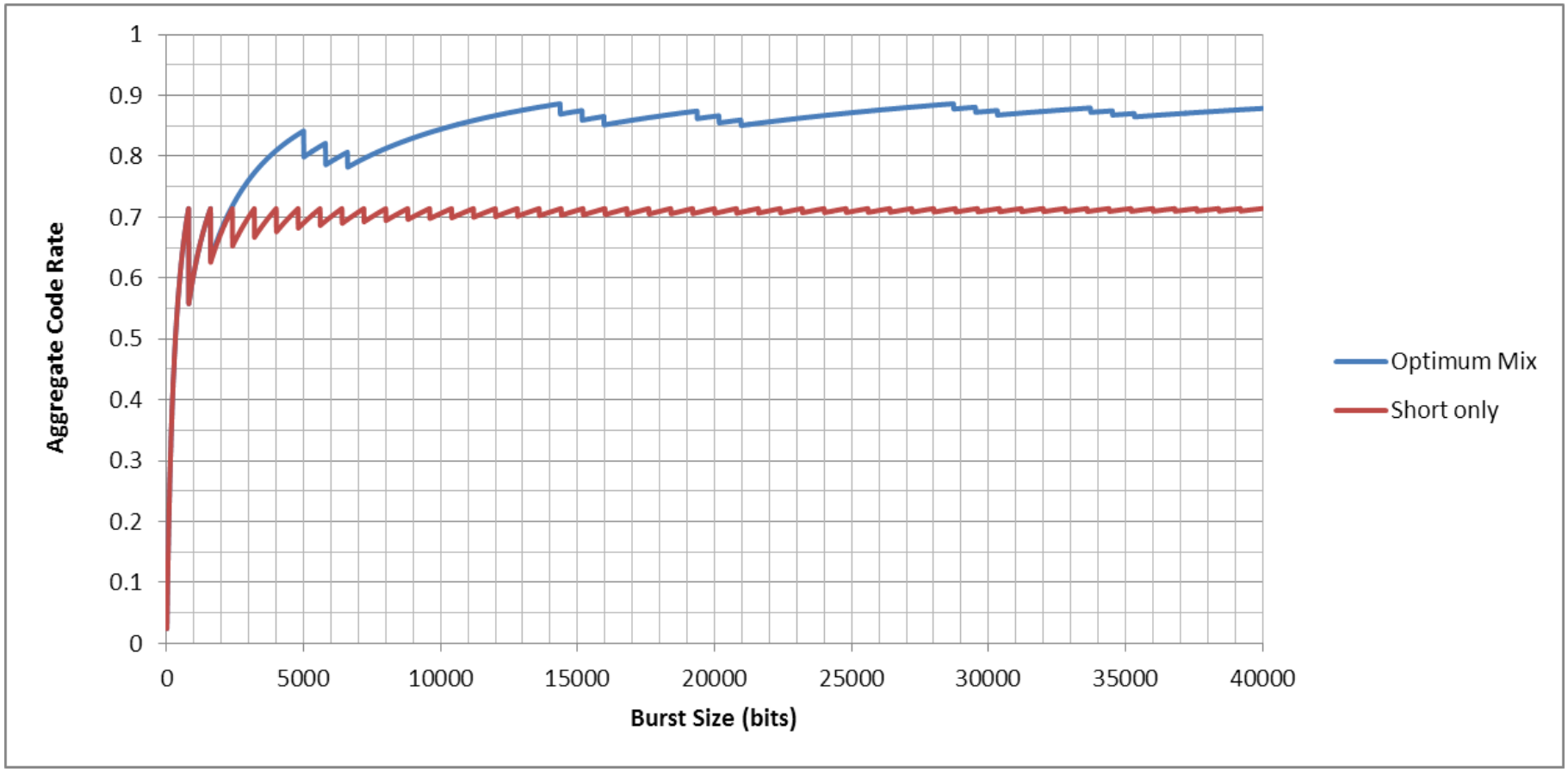
Efficiency (Code Rate) vs. Burst Size (Information Bits)



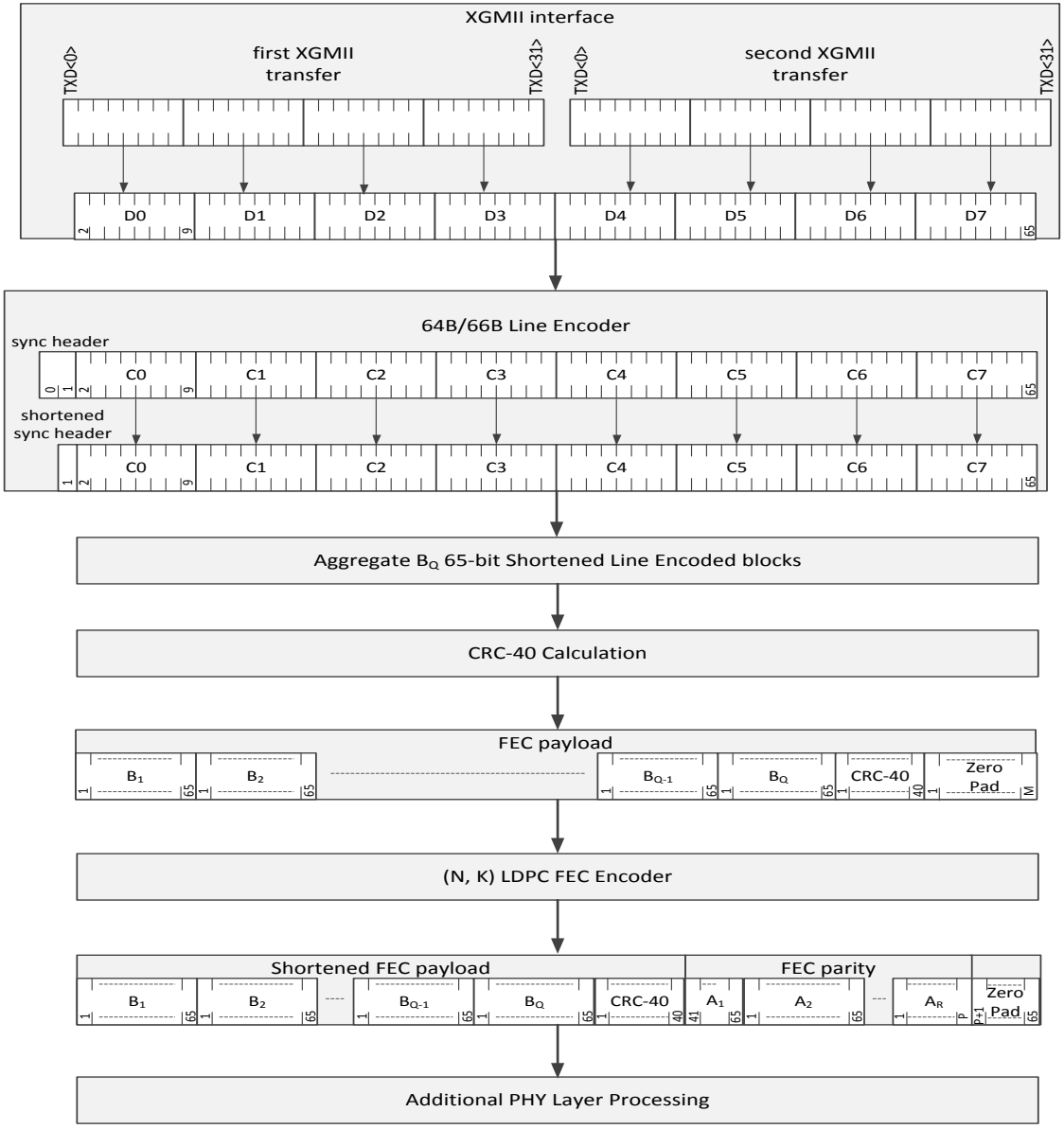
Efficiency (Code Rate) vs. Burst Size (Information Bits)



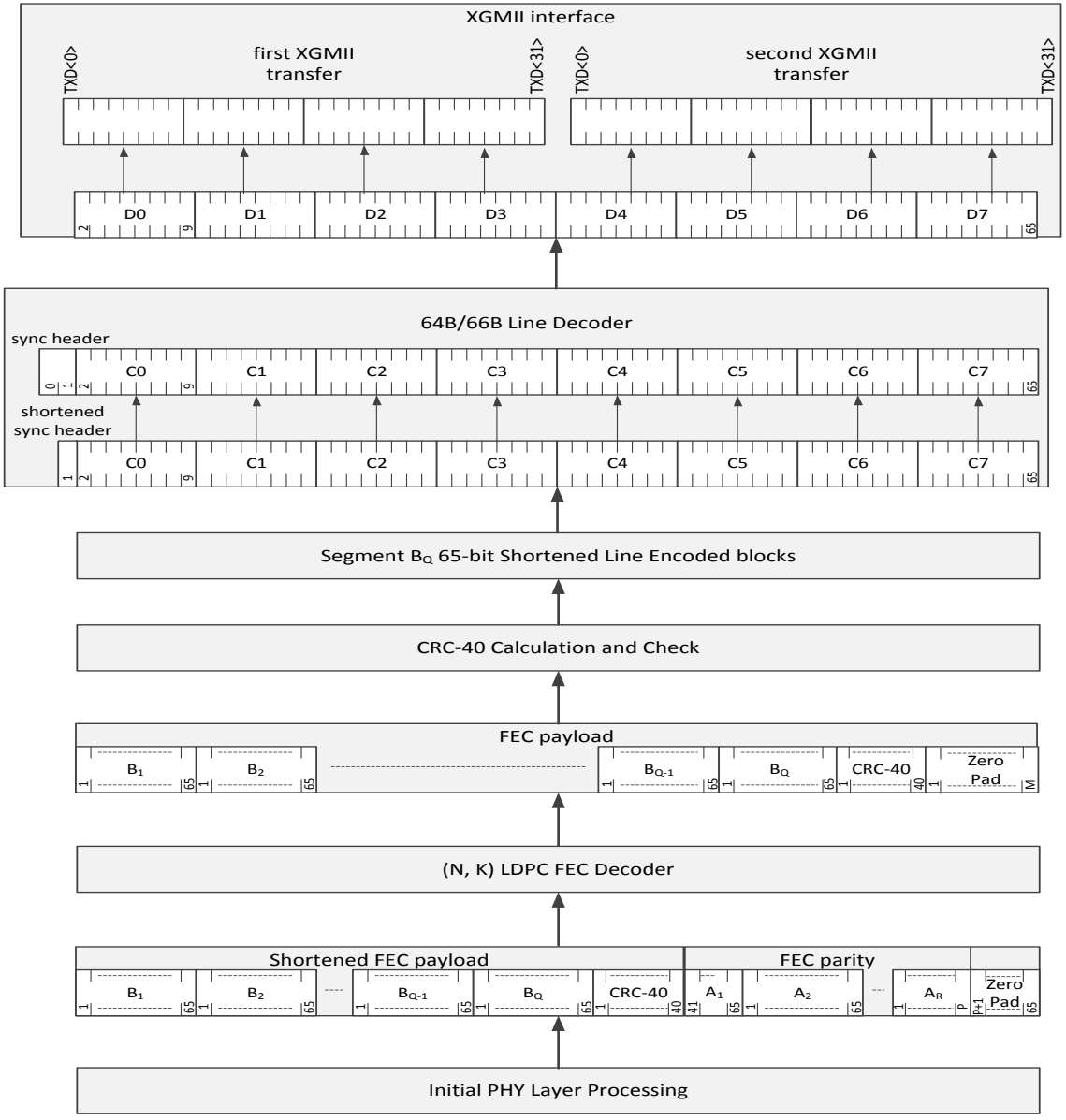
Efficiency (Code Rate) vs. Burst Size (Information Bits)



UPSTREAM FEC ENCODE PROCESS



UPSTREAM FEC DECODE PROCESS



CODEWORD PARAMETERS

New

Codeword (N,K) (bits)	Info K	Parity N - K	CRC	Info-CRC	Parity+CRC	Total	Rate (bits)	Encode Threshold	Decode Threshold
(16200, 14400)	14400	1800	40	14360	1840	16200	0.88641975	6600	8440
(5940, 5040)	5040	900	40	5000	940	5940	0.84175084	1600	2540
(1120, 840)	840	280	40	800	320	1120	0.71428571		

Codeword (N,K) (bits)	Information 65-bit blocks	Information Pad (bits)	Parity 65-bit blocks	Codeword 65-bit blocks	Codeword Pad (bits)	Rate (64/[66-1])
(16200, 14400)	220	60	29	249	45	0.8699413
(5940, 5040)	76	60	15	91	35	0.82231615
(1120, 840)	12	20	5	17	5	0.69502262

CODEWORD FILLING TABLE

New

Payload (65-bit blocks)	Shortened Last CW (mixed) # Long	# Medium	# Short	Shortened Rate (bits)	Shortened Rate (65-bit blocks)	Info Payload (bits)	Last Codeword Info Zero Pad (bits)	Encoded Payload (bits)
1	0	0	1	0.168831169	0.166666667	65	735	390
12	0	0	1	0.709090909	0.705882353	780	20	1105
13	0	0	2	0.569023569	0.565217391	845	735	1495
24	0	0	2	0.709090909	0.705882353	1560	20	2210
25	0	1	0	0.633528265	0.625	1625	3375	2600
76	0	1	0	0.840136054	0.835164835	4940	60	5915
77	0	1	1	0.798882682	0.793814433	5005	795	6305
89	0	1	1	0.821149752	0.816513761	5785	15	7085
90	0	1	2	0.787348587	0.782608696	5850	750	7475
101	0	1	2	0.806015961	0.801587302	6565	795	8190
102	1	0	0	0.782762692	0.778625954	6630	7730	8515
220	1	0	0	0.885997522	0.883534137	14300	60	16185
221	1	0	1	0.869288956	0.866666667	14365	795	16575
233	1	0	1	0.875180584	0.872659176	15145	795	17355
234	1	0	2	0.859807801	0.857142857	15210	730	17745
245	1	0	2	0.865254007	0.862676056	15925	795	18460
246	1	1	0	0.851891316	0.848275862	15990	3370	18850
297	1	1	0	0.874122708	0.870967742	19305	4995	22165
298	1	1	1	0.862038273	0.858789625	19370	790	22555
310	1	1	1	0.866666667	0.863509749	20150	790	23335
311	1	1	2	0.855299344	0.852054795	20215	725	23725
322	1	1	2	0.859548255	0.856382979	20930	790	24440
323	2	0	0	0.850861196	0.847769029	20995	7665	24765
441	2	0	0	0.886226619	0.883767535	28665	14295	32435
442	2	0	1	0.877787962	0.875247525	28730	790	32825
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Codeword Filling Thresholds

New

	Encode (bits)		Decode (bits)	
	Threshold	ΔThreshold	Threshold	ΔThreshold
0	800	800	1120	1120
1	1600	800	2240	1120
2	5000	3400	5940	3700
3	5800	800	7060	1120
4	6600	800	8180	1120
5	14360	7760	16200	8020

Codeword Filling Algorithm

New

- **0) Initialization**
  - Δthreshold[0:5]; threshold=Δthreshold[0]; #blocks=0; I=0
- **1) Find start Burst Marker**
- **2) Get next block**
  - add to CW_Buffer[ ]; #blocks = #blocks+1
- **3) If end Burst Marker found**
  - Go to step 7
- **4) If #blocks > threshold**
  - I = I+1 modulo 6; threshold = threshold + Δthreshold[I]
  - Else Go to step 2
- **5) If I = 0**
  - Process and output Long Codeword in CW_Buffer[ ]; reset CW_Buffer[ ]
- **6) If end Burst Marker not found**
  - Go to step 2
- **7) Process and output Codewords in CW_Buffer[ ] using value of I**

Codeword Filling Algorithm (continued)

New

- **Codewords in CW_Buffer[ ] processed for value of I as follows:**

I	CW_Buffer[ ]
0	shortened SHORT
1	full SHORT; shortened SHORT
2	shortened MEDIUM
3	full MEDIUM; shortened SHORT
4	full MEDIUM; full SHORT; shortened SHORT
5	shortened LONG

CONCLUSION

- **The method for selecting, combining, and shortening of multiple codeword sizes and rates that maximizes overall efficiency (aggregate code rate) is described**
- **A solution to mitigate the sparse last shortened codeword decoding problem is described which maintains the optimum efficiency**
- **Loss of efficiency for single size codewords has been shown**
- **A reversible upstream FEC encoding and decoding procedure is proposed**

PROPOSED MOTION

Move to:

Adopt the upstream codeword filling algorithm from prodan_3bn_01_0114.pdf for EPoC.

Moved: Richard Prodan

Second:

Thank You