

# **TE8: Evaluation of RIM-V2V entropy coding**

**(JCTVC-C064/m18087)**

**Vivienne Sze, Madhukar Budagavi**

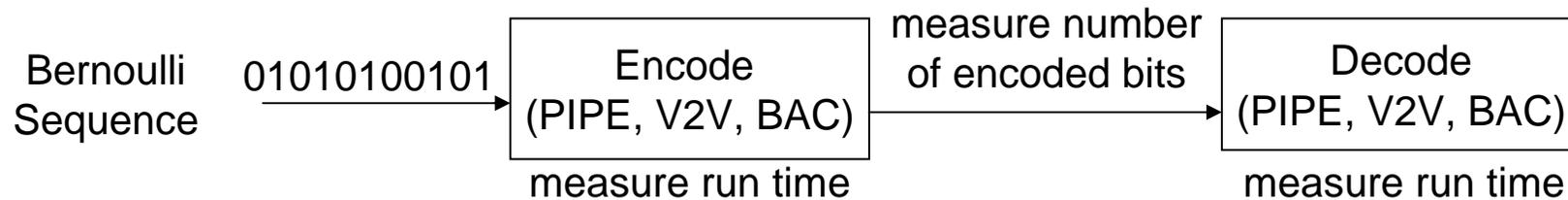
**Texas Instruments Inc.**

**Joint Collaborative Team on Video Coding (JCT-VC)  
of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11**

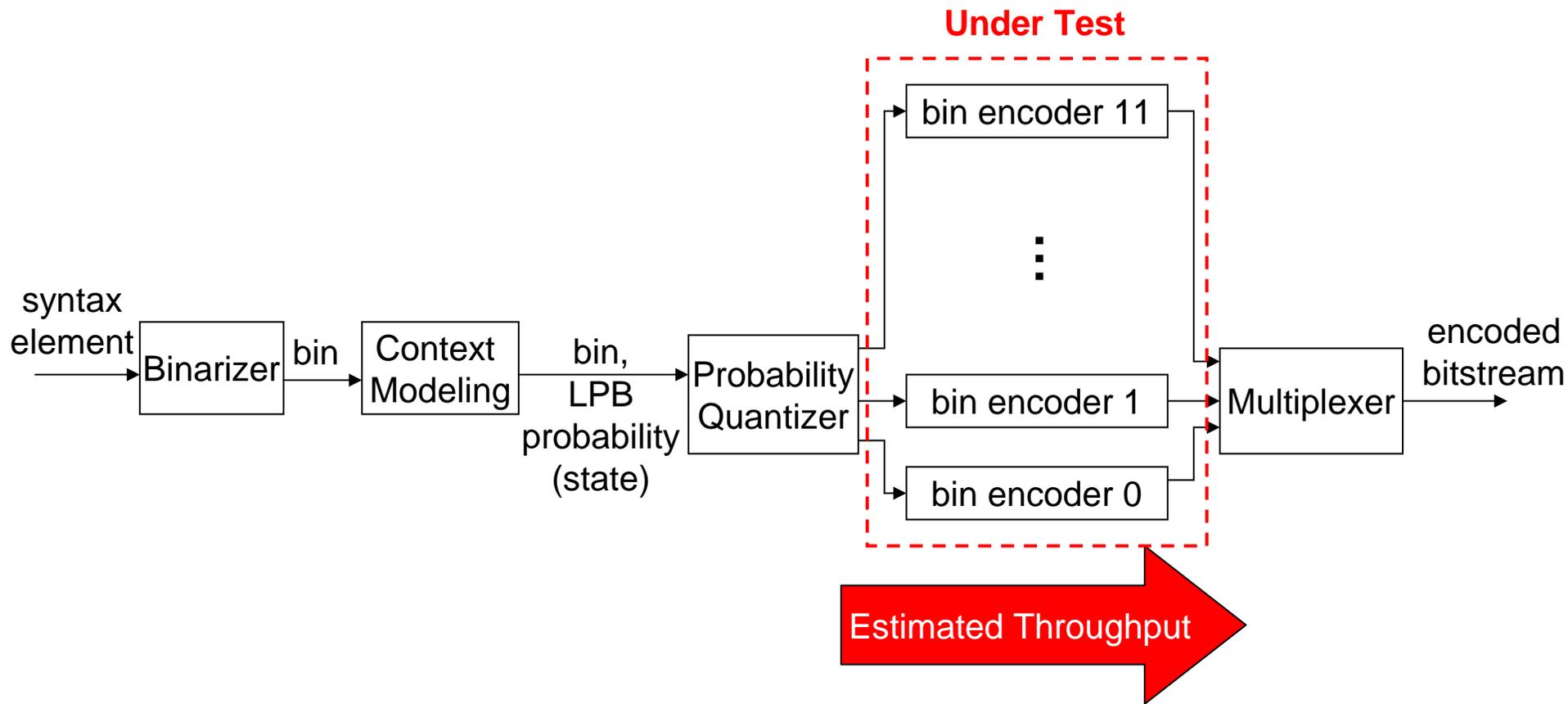
**3rd Meeting: Guangzhou, CN, 7-15 October, 2010**

# Bernoulli Test on Entropy Coding Approaches

- Proposed in JCTVC – B034(RIM)
- 12 Bernoulli sequences of one billion samples were generated for 12 distinct probabilities (PIPE).
- Compiled code for each of the entropy coding approaches (PIPE, RIM-V2V, BAC) was executed in 12 separate runs to encode/decode the same set of 12 sequences.
- For each run, the total encoded bits and the encoding and decoding times were measured.
- The average is reported across the 12 sequences.



# Key Blocks In Entropy Encoder



**End to End Throughput Still Unknown!**

# Differences with TMuC

- V2V tables don't match ones used in TMuC
  - 12 in RIM-V2V versus 24 in TMuC (TEncV2VTrees.cpp)

# Cross-Verification

- Simulation platform is LSF equipped with Intel(R) Xeon(R) CPU X5570@2.93GHz 64 bits Linux machines

Bin Encoder	Coding loss relative to entropy of Bernoulli sequence (%)	Encoder Throughput	Decoder Throughput	Code Size (Bytes)
<b>BAC</b>	0.159	73.8	141.9	16665
<b>PIPE</b>	0.147	55.9	113.4	45877
<b>V2V</b>	0.122	207.1	229.2	40425

- Results have been verified to mostly match (exact for coding loss, and in terms of ratios for throughput) those obtained from RIM.

# Concerns on Coding Efficiency Measurement

- Tests done on Bernoulli sequences rather than TMuC
- Only measured for **discrete probabilities**
  - RIM-V2V and PIPE were designed with the specific 12 probabilities in mind, while BAC used existing states which were the nearest to the 12 probabilities
  - i.e. the BAC was not customized for the given probabilities.
- Difficult to draw a conclusion on the coding efficiency.

# Concerns on Throughput Measurement

- Throughput estimated from simulation time.
- The software for each approach was written in a very different style/structure.
- Examples are provided in the next slide.

# Some Differences in Software Coding Styles

1. PIPE code includes the probability quantizer; not included for the RIM-V2V code.
2. PIPE code contain unnecessary 'for loops' (e.g. in function TEncBinPIPE::start)
3. Both PIPE and RIM-V2V use a tree structure; in RIM-V2V code, the codeword is embedded in the node, while for PIPE an additional table look up is required.
4. RIM-V2V is implemented in a very concise manner and traverses the tree with only shifts and grabs multiple bits at a time during decode. BAC is implemented using an early JM code and missing similar optimizations such a fast renormalization.
5. There are several function calls with BAC and PIPE which are not used in RIM-V2V.

# Concerns on Throughput Measurement

- Difficult to determine whether the difference in simulation time is due to the way in which the software was written or the actual entropy coding technique.
- Throughput only measured for bin encoder, not entire entropy coding engine.
- No conclusion can be drawn on throughput.
- Recommend that all proponents collaborate on a developing an accurate throughput measurement.

# Complexity

- Estimate complexity based on key metrics such as
  - number of leaves,
  - number of nodes,
  - the maximum length/bits of the codewords
  - and the maximum number of transitions (i.e. comparisons) required to reach a leaves (i.e. depth of the tree).
- Based on these metrics, the RIM-V2V tables seem to have higher complexity than the PIPE tables.

# Compare RIM-V2V and PIPE Tables

Prob.	Leaves	Nodes	Max. Depth	Max Length of CW
0.5	128	255	7	7
0.45	136	271	9	9
0.43	241	481	11	10
0.405	206	411	11	10
0.345	124	247	10	9
0.315	50	99	9	8
0.225	124	247	15	10
0.205	261	521	15	11
0.165	141	281	21	10
0.085	75	149	31	10
0.04	129	257	23	11
0.02	71	141	33	12

**RIM-V2V encoder table**

Prob.	Leaves	Nodes	Max. Depth	Max Length of CW
0.5	2	3	1	1
0.45	22	63	7	8
0.43	19	54	8	8
0.405	21	60	11	11
0.345	22	63	5	6
0.315	13	36	5	6
0.225	22	63	9	8
0.205	17	48	7	10
0.165	22	63	8	9
0.085	21	60	12	8
0.04	18	51	17	6
0.02	33	96	32	6

**PIPE encoder table**

# Compare RIM-V2V and PIPE Tables

Prob.	Leaves	Nodes
0.5	128	129
0.45	136	161
0.43	241	281
0.405	206	251
0.345	124	155
0.315	50	65
0.225	124	161
0.205	261	331
0.165	141	175
0.085	75	103
0.04	129	163
0.02	71	81

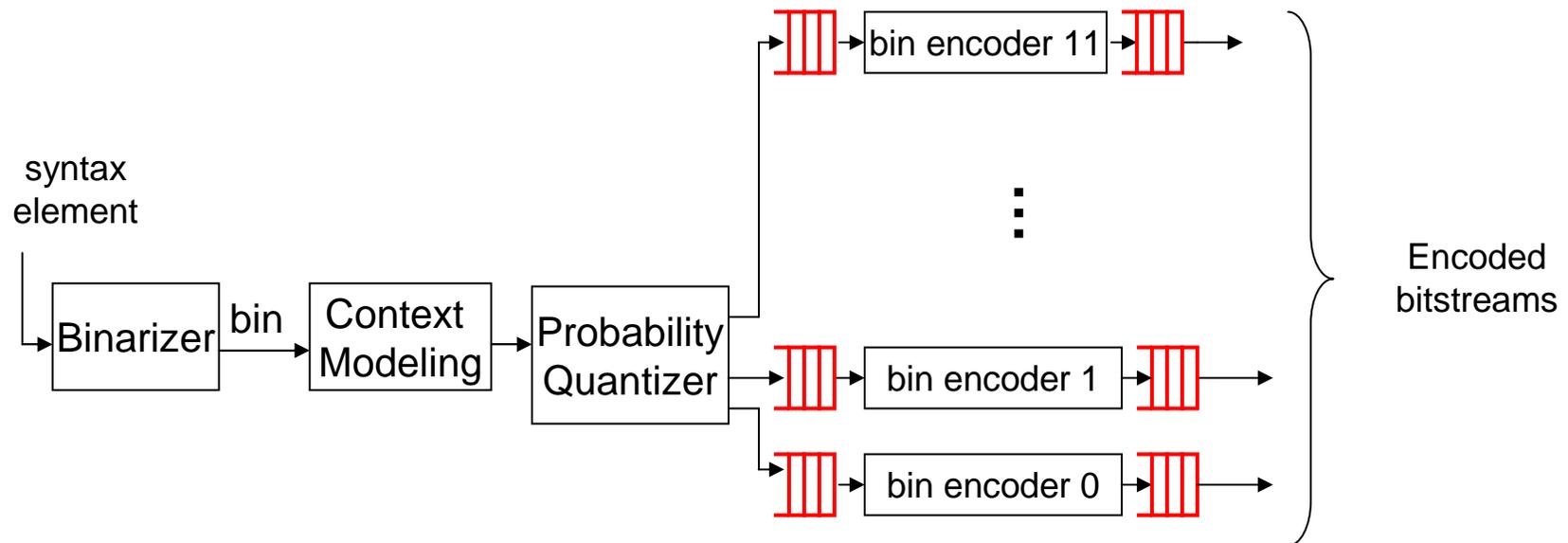
**RIM-V2V decoder table**

Prob.	Leaves	Nodes	Max. Depth	Max Length of CW
0.5	2	3	1	1
0.45	22	63	7	8
0.43	19	54	8	8
0.405	21	60	11	11
0.345	22	63	5	6
0.315	13	36	5	6
0.225	22	63	9	8
0.205	17	48	7	10
0.165	22	63	8	9
0.085	21	60	12	8
0.04	18	51	17	6
0.02	33	96	32	6

**PIPE decoder table**

# Complexity

- PIPE tables 5x larger than BAC
- RIM-V2V tables > 3x larger than PIPE; >15x larger than BAC
- Additional buffers may result in another ~10x area increase



# Conclusions and Recommendations

- Throughput Measurements
  - Throughput measured using simulation time on a Linux machine.
  - Code structure for three bin encoders (V2V, PIPE, BAC) are very different – difficult to tell whether speed up due to code structure or type of bin encoder
    - e.g. Are techniques used to construct tree (e.g. embedding Huffman code in node, storing min depth of tree for max bit parsing) transferrable to PIPE?
  - Throughput measured for each bin encoder and averaged; does not account for distribution (loading) across encoders/decoders

# Conclusions and Recommendations

- Coding Efficiency
  - Not measured in TMuC environment
  - BAC is not designed for the probabilities used in test (select state which is closest); whereas V2V and PIPE design specifically for those probabilities
- Complexity requires further analysis, including area **and** power costs
- Area cost estimated to be 15x higher than BAC
- Further study is recommended