A HIGH THROUGHPUT CABAC ALGORITHM USING SYNTAX ELEMENT PARTITIONING

Vivienne Sze

Anantha P. Chandrakasan

2009 ICIP – Cairo, Egypt

MTL
microsystems technology laboratories
massachusetts institute of technology

Motivation

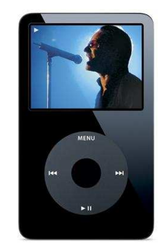
- High demand for video on mobile devices
- Compression to reduce storage and transmission
- Battery capacity limited by size, weight, and cost
- Need low power video coding
- Achieve performance required for real time HD

Palm Pre

iPhone

PSP

Digital Camera

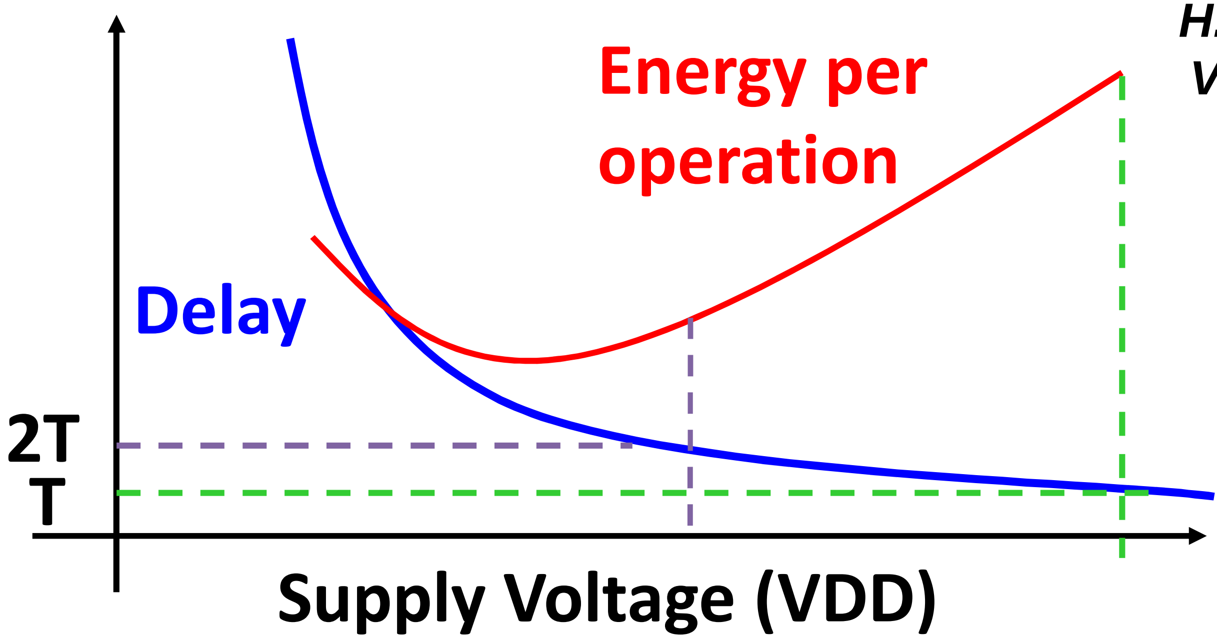


iPod

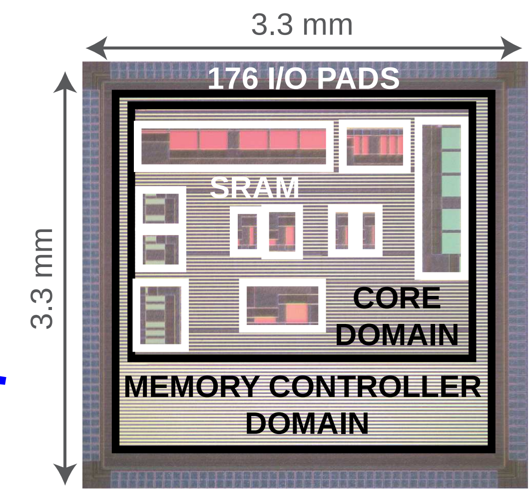
Video Conferencing

DVC

Low Power Video Coding



H.264/AVC Baseline Decoder
V.Sze et al. JSSC, Nov. 2009

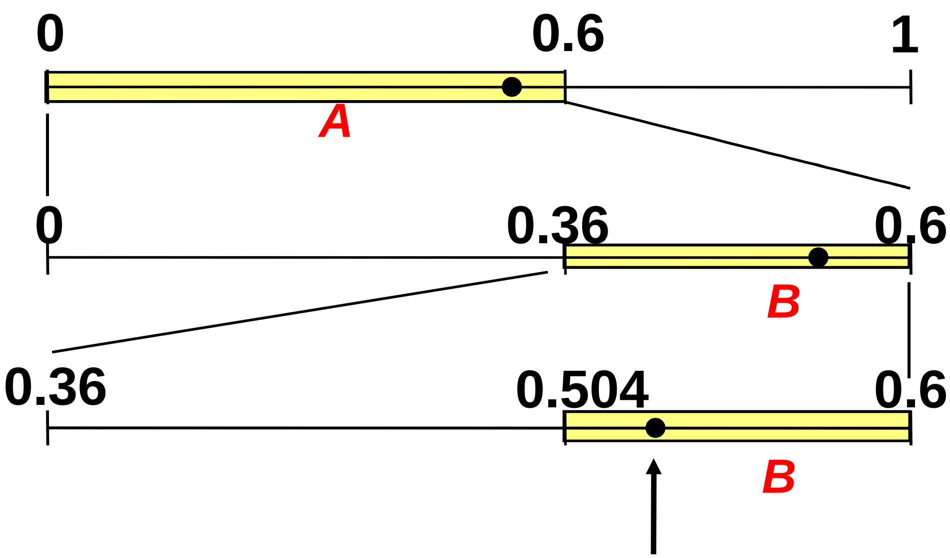


- Parallelism and voltage scaling shown to be effective in power reduction → > 10x power reduction
- However, certain algorithms inherently serial
  - E.g. Context Adaptive Binary Arithmetic Coding (CABAC)
- H.264/AVC High Profile uses CABAC for entropy coding

Arithmetic Coding

Example: Pr(*A*) = 0.6; Pr(*B*) = 0.4

Entropy Encoding Symbol Sequence: "***A-B-B***"



Range	Offset
1	0
0.6	0
0.24	0.36

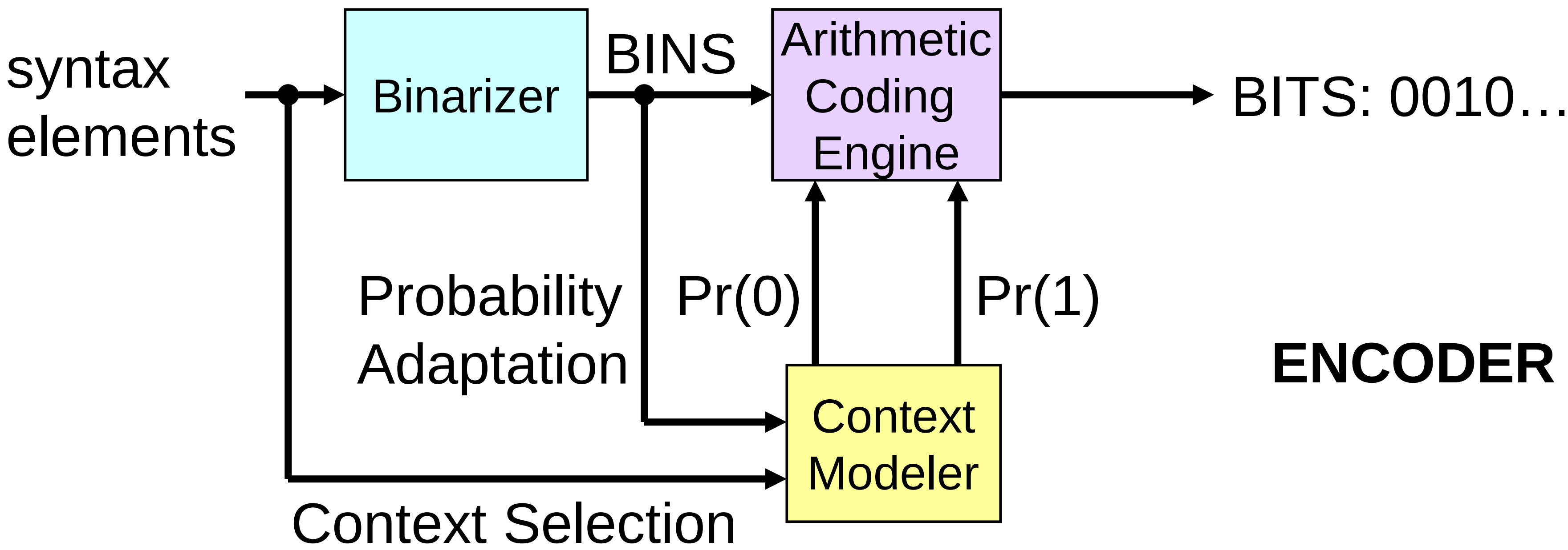
Output Binary Bitstream: .1001
(Binary Fraction)

- Binary Arithmetic Coding has binary symbols ('bins')
- Binarizer maps syntax elements to bins
- Range updated after every bin

Context-Adaptive

- Context (probability model)
- Adaptive estimation of probability (update context state)
- Context can be switched and updated ***every*** bin

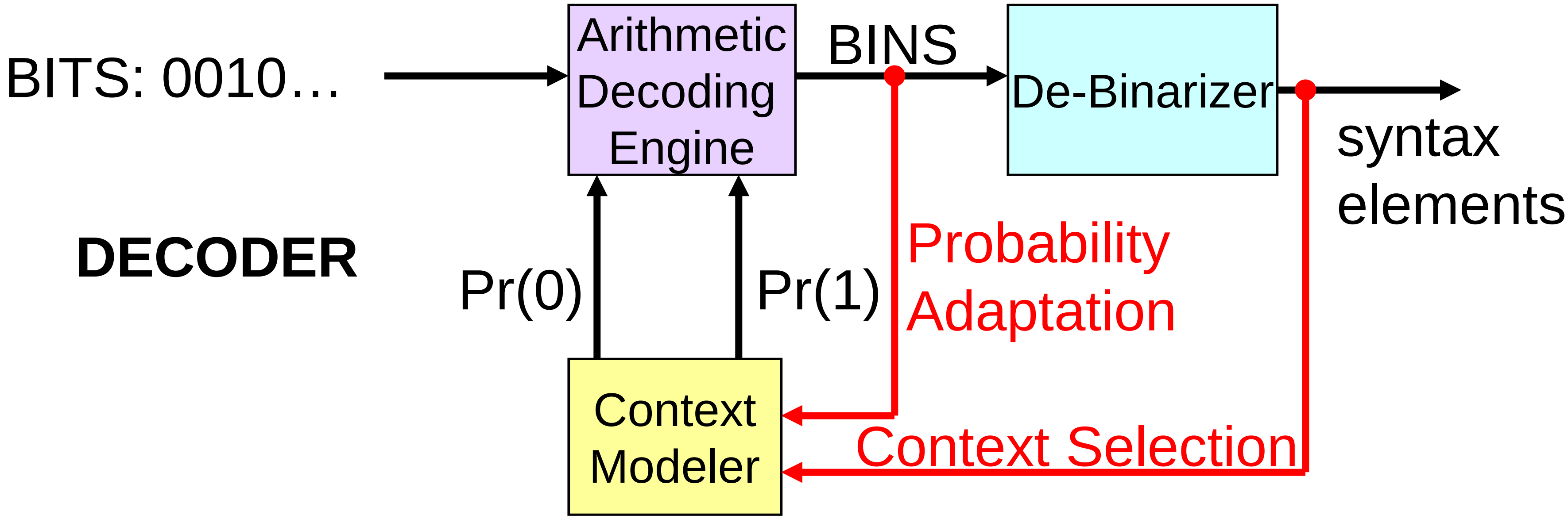
Bin-to-bin dependencies → Cycle-to-cycle dependencies



Encoder: Syntax Element → Bins → Bits

CABAC Challenges

Decoder: Bits → Bins → Syntax Element



- Data Dependencies (difficult to parallelize)
  - Contexts and Range are updated after every bin
  - At decoder, data feedback required
- Context modeling and interval division tied to bins (not bits)
  - Number of cycles proportional to number of bins

Real-time H.264 CABAC Requirements

Level	Max Frame Rate	Max Bins per Picture	Max Bit Rate	Peak Bin Rate
	fps	Mbins	Mbits/sec	Mbins/sec
4.0	30	9.2	25	275
5.1	26.7	17.6	300	2107

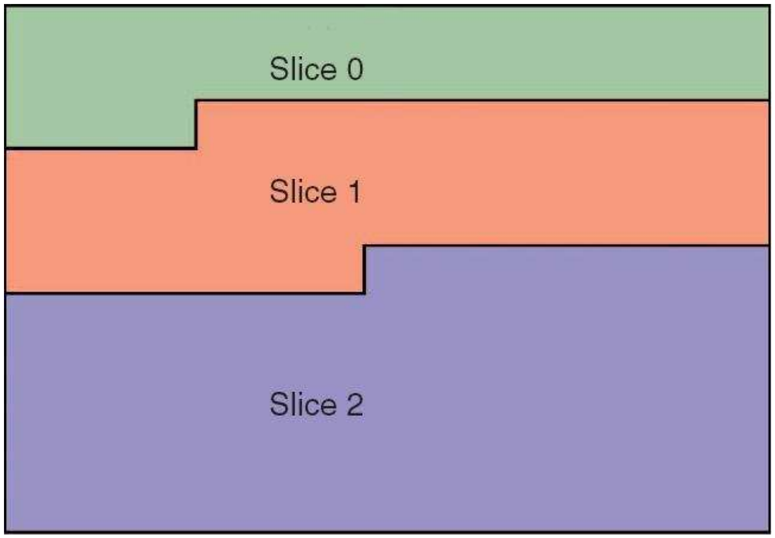
Max Bin Rate = (Max Bins per Picture) x (Max Frame Rate)

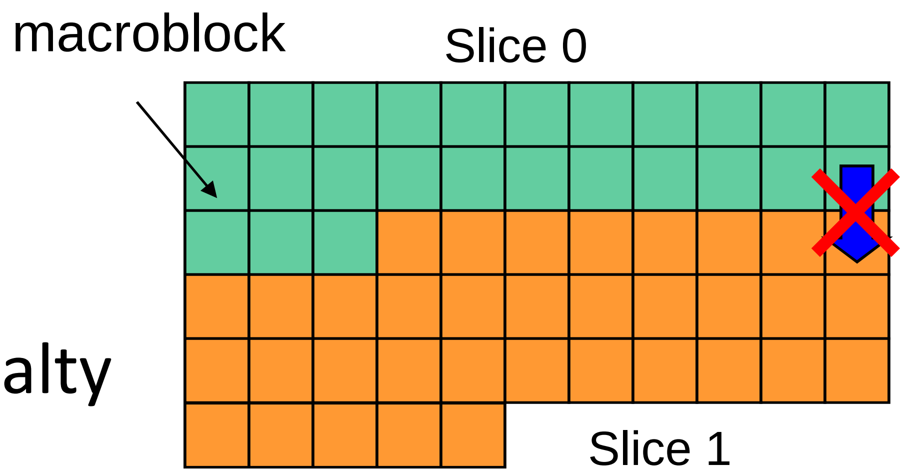
- For real-time decoding, decode frame within inter-frame time interval
- Frequency requirements reach *multi-GHz range*

Parallelism needed to lower frequency to acceptable range

H.264/AVC CABAC Parallelism

- Bin
  - Speculation required
- Frame
  - Buffering required for frames
  - Limited by latency requirement
- Slice
  - Coding Efficiency Penalty

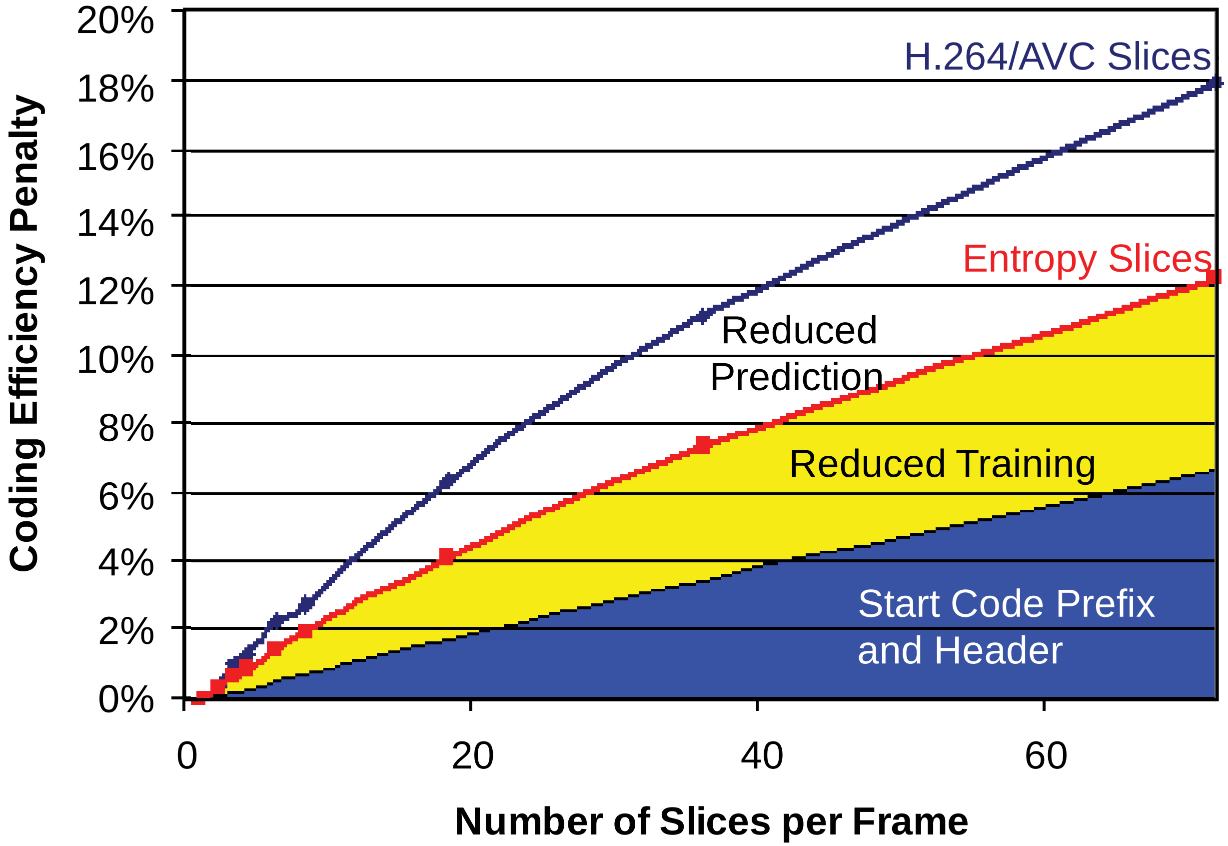




Can we do better by changing the algorithm?

Entropy Slices

- Proposed by Sharp in 2008 [VCEG-AI32]
- Only entropy coding is independent
- Coding penalty overhead due to reduced training

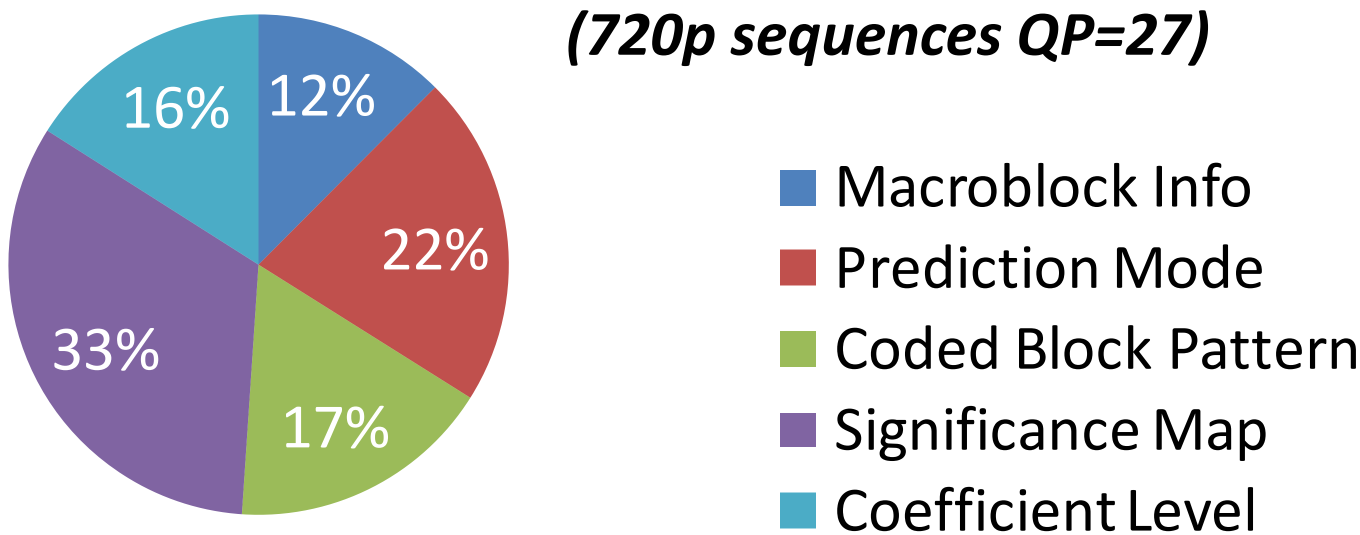


Can we further reduce coding penalty?

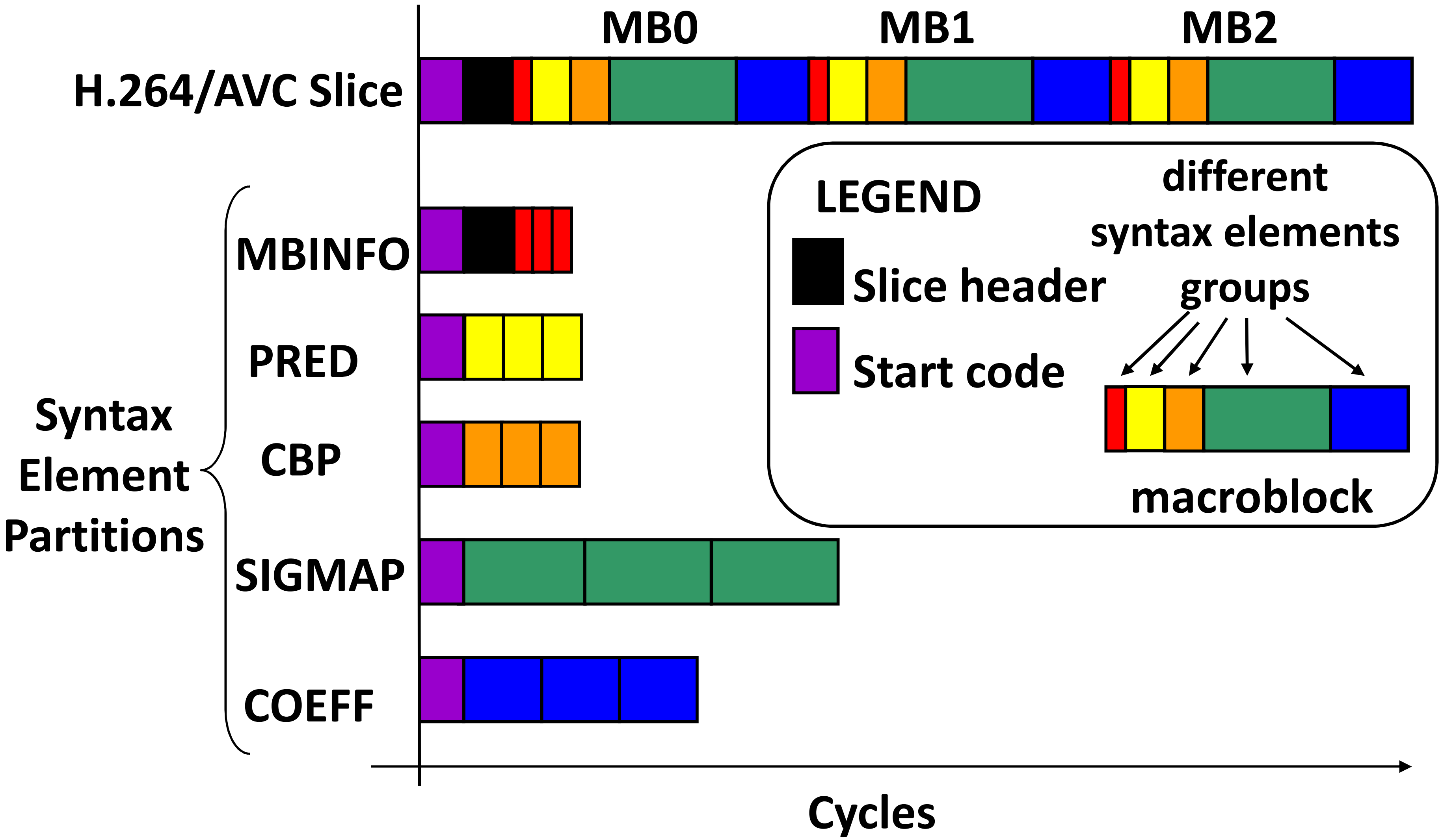
Syntax Element Parallelism

- Place syntax elements in different groups
- Assign groups to different partitions and process partitions in parallel
- Allocation of syntax elements to partitions based on distribution (balance workload)

E.g. Average distribution of bins (720p sequences QP=27)



Reduce Cycle Count



Context Training for Coding Efficiency

- Coding efficiency depends on accuracy of bin probability estimate
- Better estimate achieved with more bins (context training)
- Syntax element partitioning does not reduce number of bins used with each context

Entropy Slices per frame [MB/slice]	Total Coding Penalty	Coding Penalty due to Reduced Training
1 [3600]	0.00%	0.00%
2 [1800]	0.30%	0.20%
3 [1200]	0.61%	0.41%
4 [900]	0.88%	0.57%
6 [600]	1.47%	0.95%
8 [450]	1.93%	1.20%
18 [200]	4.13%	2.38%
36 [100]	7.36%	3.87%
72 [50]	12.21%	5.50%

e.g. BigShips QP=27, IPPP

Improved Coding Efficiency

Area Cost (ASIC)

- Entire CABAC does NOT have to be replicated
  - Context selection, and context memory are not replicated
- Area increase due to
  - Replicated arithmetic decoder
  - Control and FIFO between engines

Experimental Results

- Validated with JM12.0 under common conditions across 720p: BigShips, City, Crew, Night, ShuttleStart

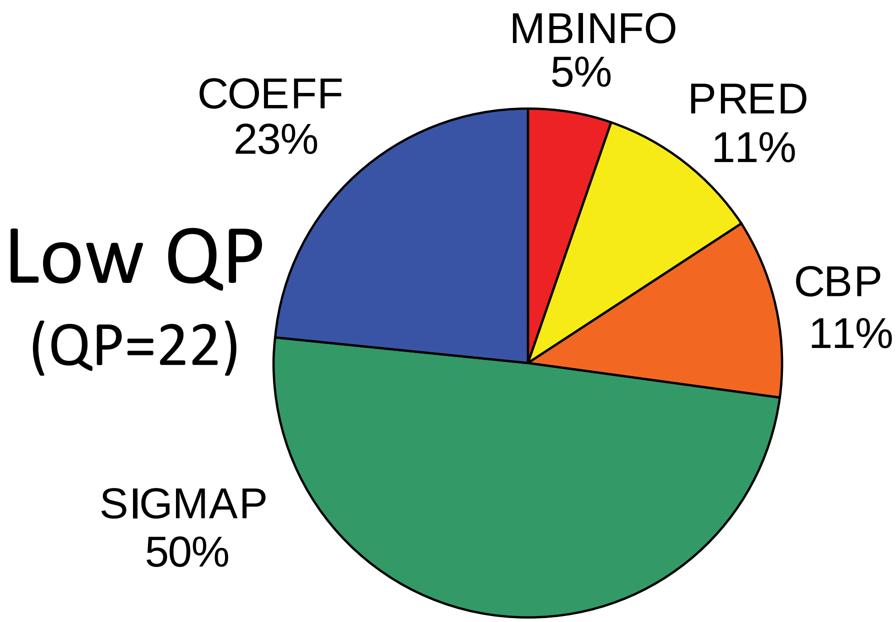
For approx. same speed-up (~2.4 to 2.7x)

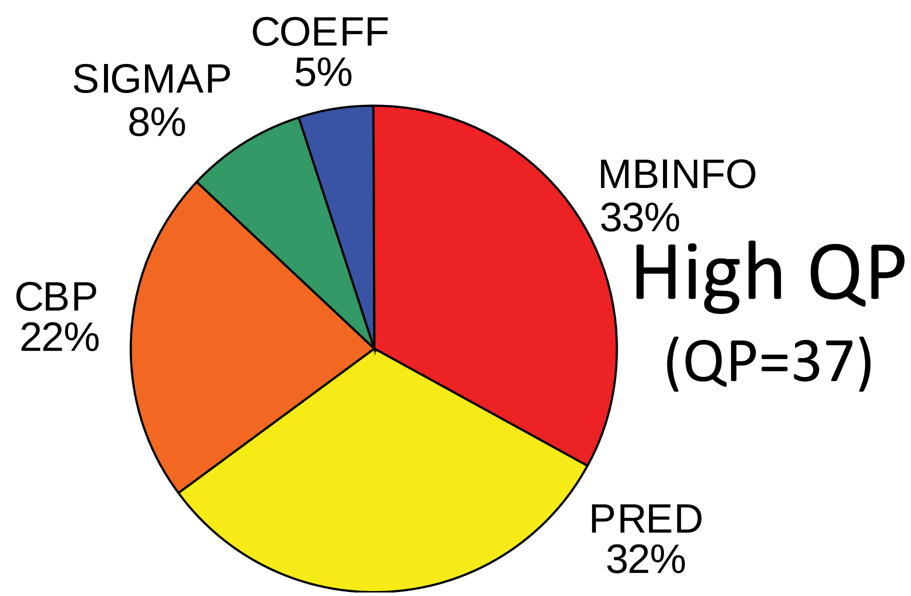
	H.264/AVC Slices		Entropy Slices		Syntax Element Partitioning	
Area Cost	3x		3x		1.5x	
Prediction Structure	BD-rate	Speed-up	BD-rate	Speed-up	BD-rate	Speed-up
Ionly	0.87	2.43	0.25	2.43	0.06	2.60
IPPP	1.44	2.42	0.55	2.44	0.32	2.72
IBBP	1.71	2.46	0.69	2.47	0.37	2.76

2 to 4x reduction in coding penalty

Adaptive Bin Allocation (Varying QP)

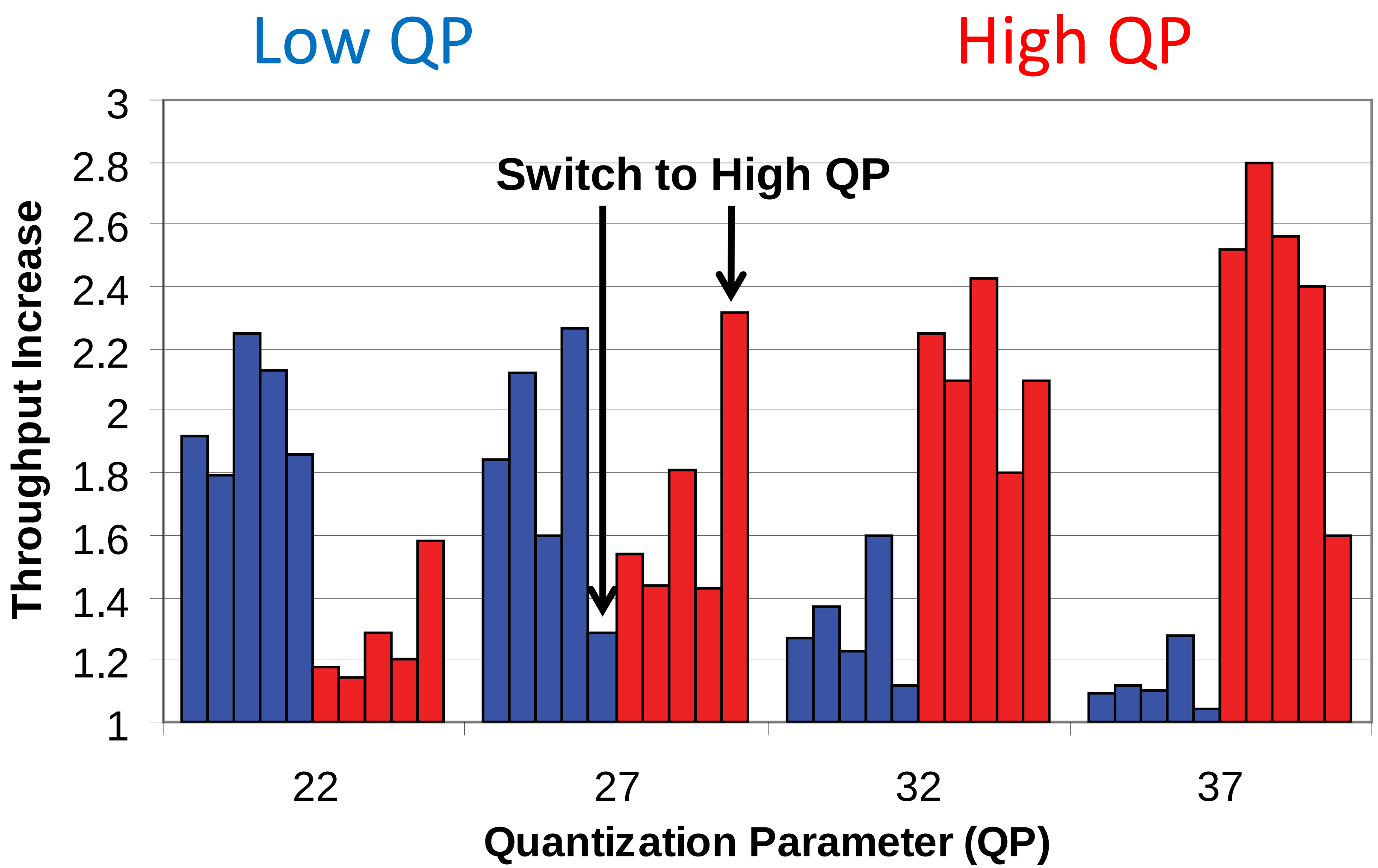
- To reduce Start Code overhead – assign multiple groups to each partition and reduce partitions (5→3)
- Bin distribution changes with QP – combine adaptively





Mode	MBINFO	PRED	CBP	SIGMAP	COEFF
Low QP	0	0	0	1	2
High QP	0	1	2	2	2

Throughput Increase

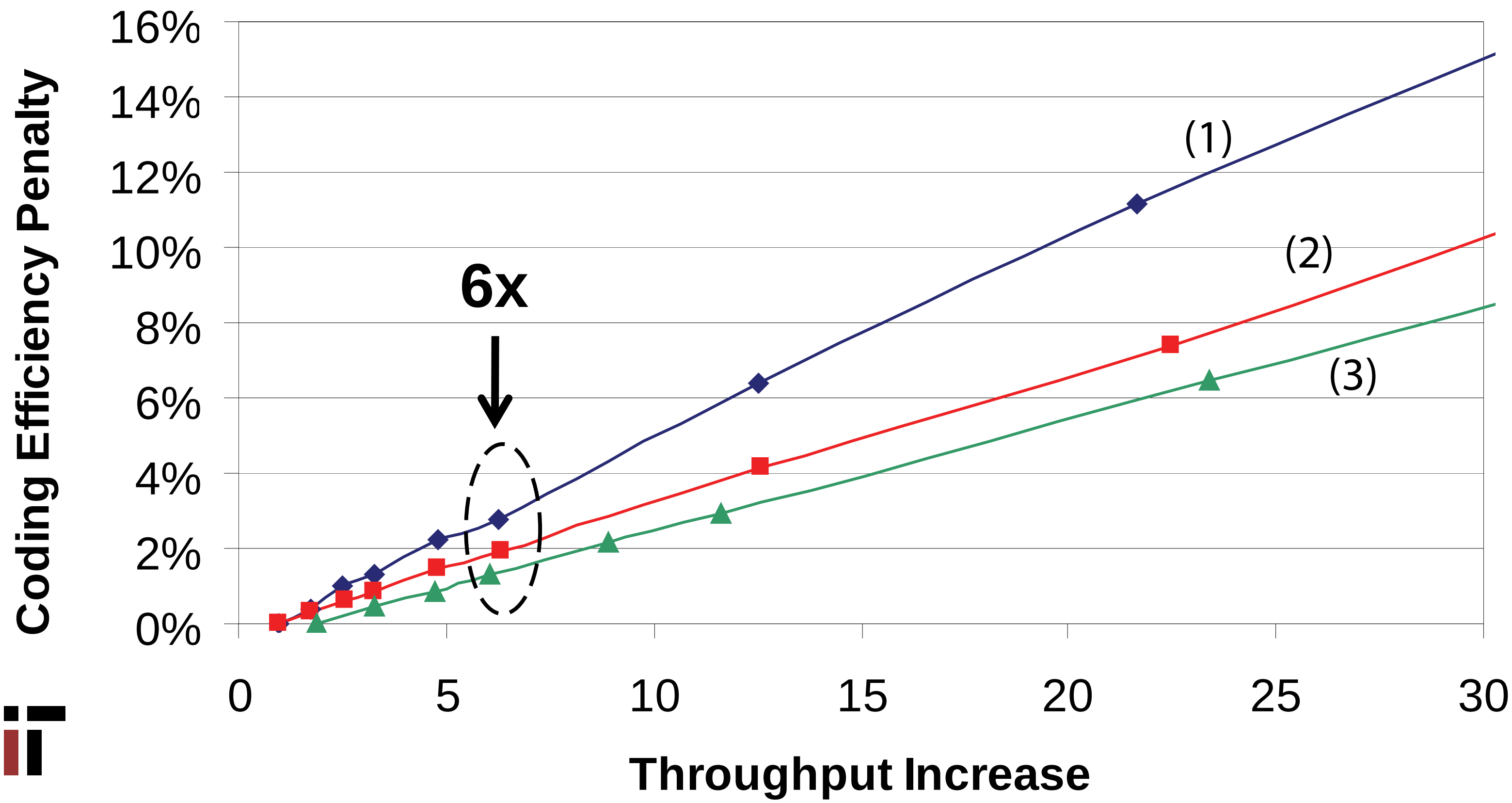


(Left to Right) BigShips, City, Crew, Night, ShuttleStart

Additional Parallelism

- Combine with slice level parallelism

1. H.264/AVC Slices (8 slices)
2. Entropy Slices (8 slices)
3. Entropy Slices (4 slices) + Syntax Element Partitioning



Conclusions

- A new CABAC algorithm for next generation standard to increase concurrency by processing the bins of different syntax elements in parallel.
- Achieve a throughput increase of up to 3x without sacrificing coding efficiency, power, or delay and minimal area cost.
- Can be combined with other approaches for improved coding efficiency and throughput/power.

Acknowledgements:

Funding from Texas Instruments and NSERC