A 2014 Mbin/s Deeply Pipelined CABAC Decoder for HEVC

Yu-Hsin Chen, Vivienne Sze

Massachusetts Institute of Technology

Next-Generation Video Coding

Ultra-HD (UHD) comes into play

4K Streaming

4K Broadcasting

Moving toward **Mobile**

4K Recording

S/W Support

Next-Generation Video Coding

Ultra-HD (UHD) comes into play

4K Streaming

4K Broadcasting

Moving toward **Mobile**

4K Recording

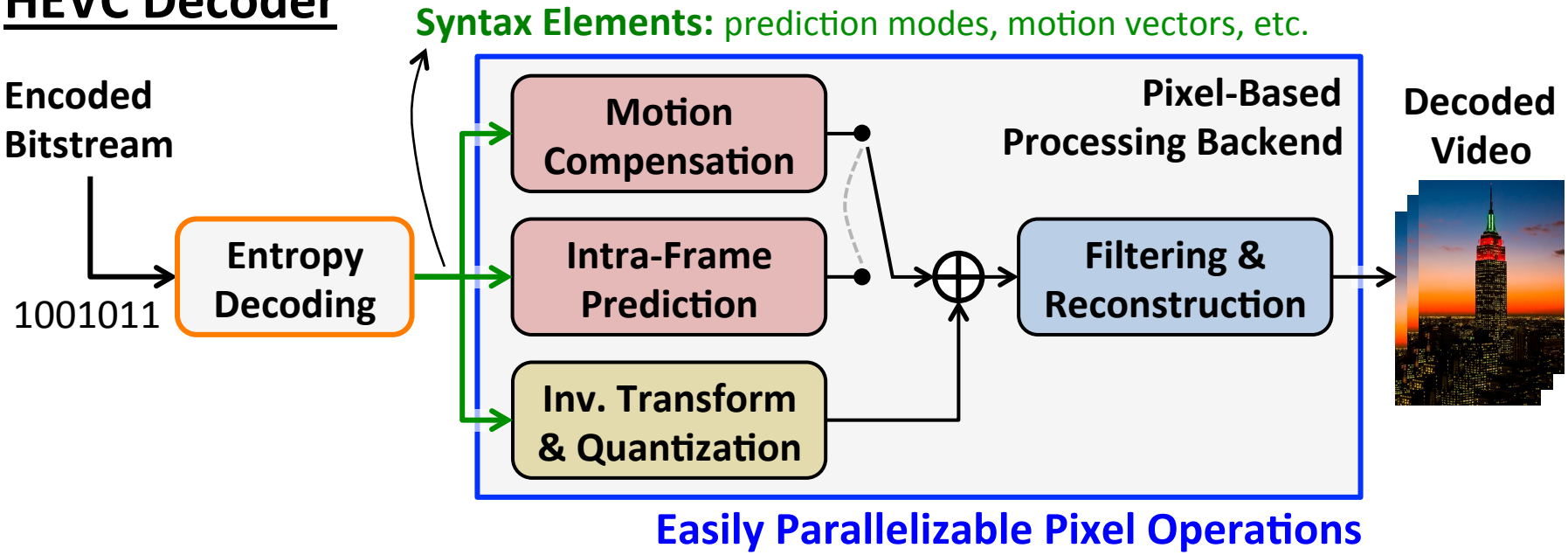
S/W Support

High Efficiency Video Coding (HEVC)

- 2× higher coding efficiency than H.264/AVC
- **High Throughput** → 8K UHD @ 120 fps
- **Low Power** → prolonged battery life

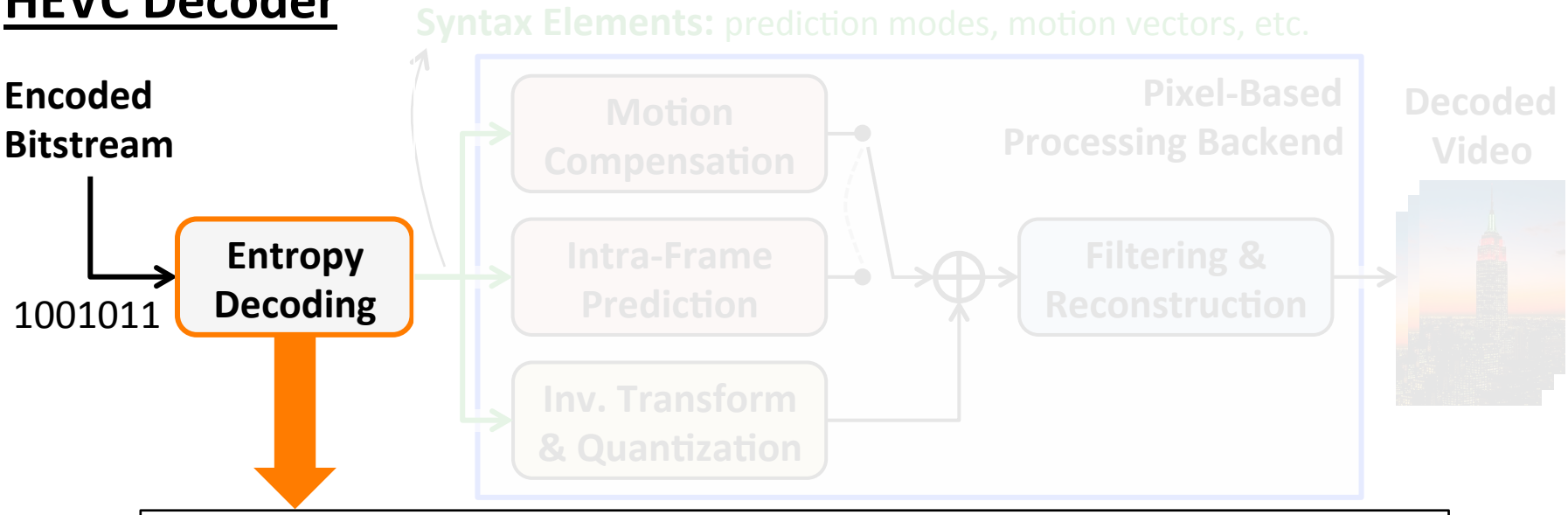
HEVC Throughput Bottleneck – CABAC

HEVC Decoder



HEVC Throughput Bottleneck – CABAC

HEVC Decoder



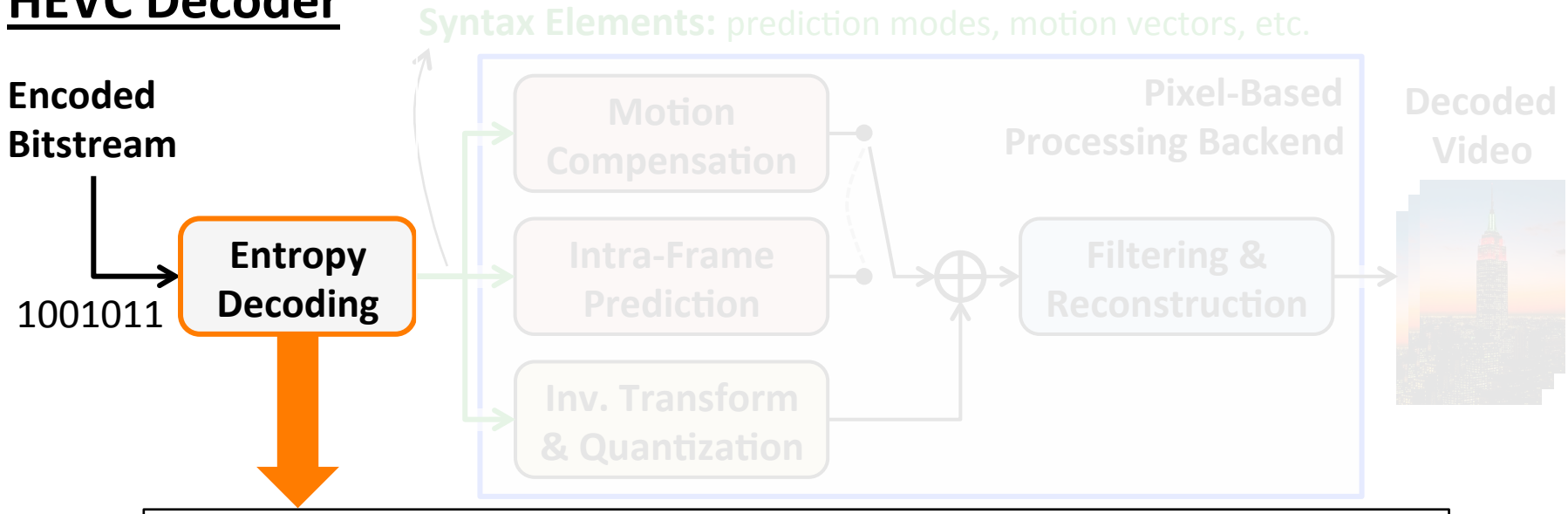
Context Adaptive Binary Arithmetic Coding (CABAC)*

- **Highly serial processing** → low CABAC throughput
- **Limits** decoder **throughput**
- **Limits** voltage scaling of the decoder for **low power**

* [D. Marpe, *IEEE TCSVT*, July 2003]

HEVC Throughput Bottleneck – CABAC

HEVC Decoder



Context Adaptive Binary Arithmetic Coding (CABAC)*

- **Highly serial processing** → low CABAC throughput
- **Limits** decoder **throughput**
- **Limits** voltage scaling of the decoder for **low power**

High-throughput CABAC decoding is the key to HEVC decoder performance

* [D. Marpe, *IEEE TCSVT*, July 2003]

CABAC Coding Basics

- **Task**: To code two syntax elements **AB** with CABAC

Syntax Element **A**		Context Model $\mathbf{P_A}$
Messages	$\mathbf{a_1}$	50%
	$\mathbf{a_2}$	30%
	$\mathbf{a_3}$	15%
	$\mathbf{a_4}$	5%

Syntax Element **B**		Context Model $\mathbf{P_B}$
Messages	$\mathbf{b_1}$	25%
	$\mathbf{b_2}$	45%
	$\mathbf{b_3}$	30%

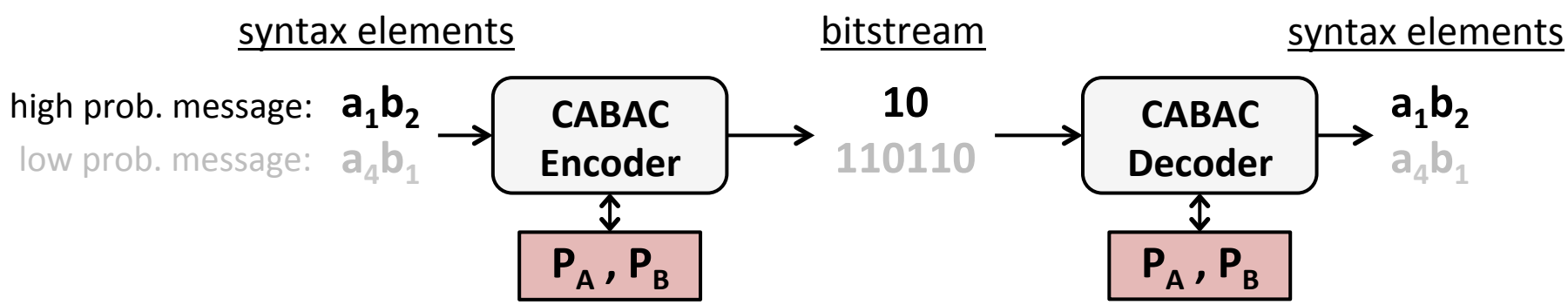
CABAC Coding Basics

- **Task**: To code two syntax elements **AB** with CABAC

Syntax Element **A**		Context Model P_A
Messages	a_1	50%
	a_2	30%
	a_3	15%
	a_4	5%

Syntax Element **B**		Context Model P_B
Messages	b_1	25%
	b_2	45%
	b_3	30%

Variable-Length Code: Use **less bits** to code **higher-probability** messages



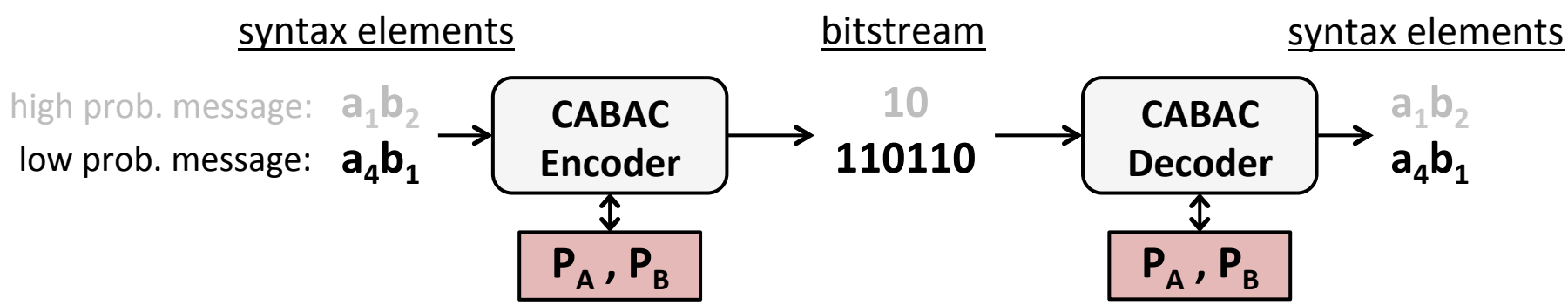
CABAC Coding Basics

- **Task**: To code two syntax elements **AB** with CABAC

Syntax Element **A**		Context Model P_A
Messages	a_1	50%
	a_2	30%
	a_3	15%
	a_4	5%

Syntax Element **B**		Context Model P_B
Messages	b_1	25%
	b_2	45%
	b_3	30%

Variable-Length Code: Use **less bits** to code **higher-probability** messages



CABAC Coding Basics

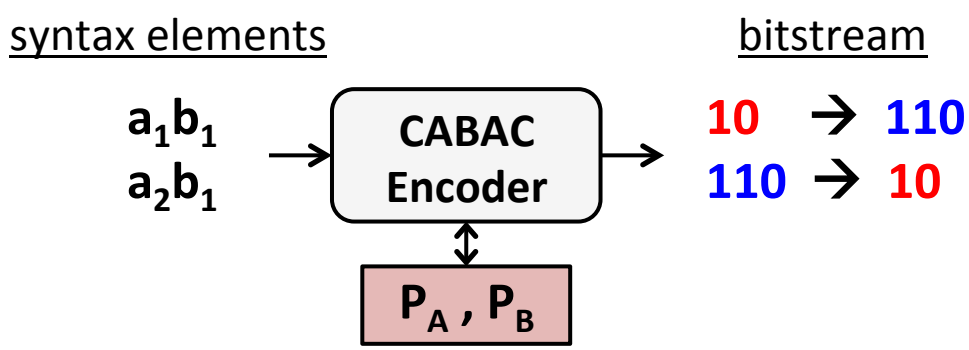
- **Task**: To code two syntax elements **AB** with CABAC

Syntax Element **A**		Context Model P_A
Messages	a_1	50% → 30%
	a_2	30% → 50%
	a_3	15%
	a_4	5%

Syntax Element **B**		Context Model P_B
Messages	b_1	25%
	b_2	45%
	b_3	30%

Variable-Length Code: Use **less bits** to code **higher-probability** messages

CABAC is **adaptive** to changing context models



CABAC Coding Basics

- **Task**: To code two syntax elements **AB** with CABAC

Syntax Element **A**		Context Model $\mathbf{P_A}$
Messages	$\mathbf{a_1}$	50% → 30%
	$\mathbf{a_2}$	30% → 50%
	$\mathbf{a_3}$	15%
	$\mathbf{a_4}$	5%

Syntax Element **B**		Context Model $\mathbf{P_B}$
Messages	$\mathbf{b_1}$	25%
	$\mathbf{b_2}$	45%
	$\mathbf{b_3}$	30%

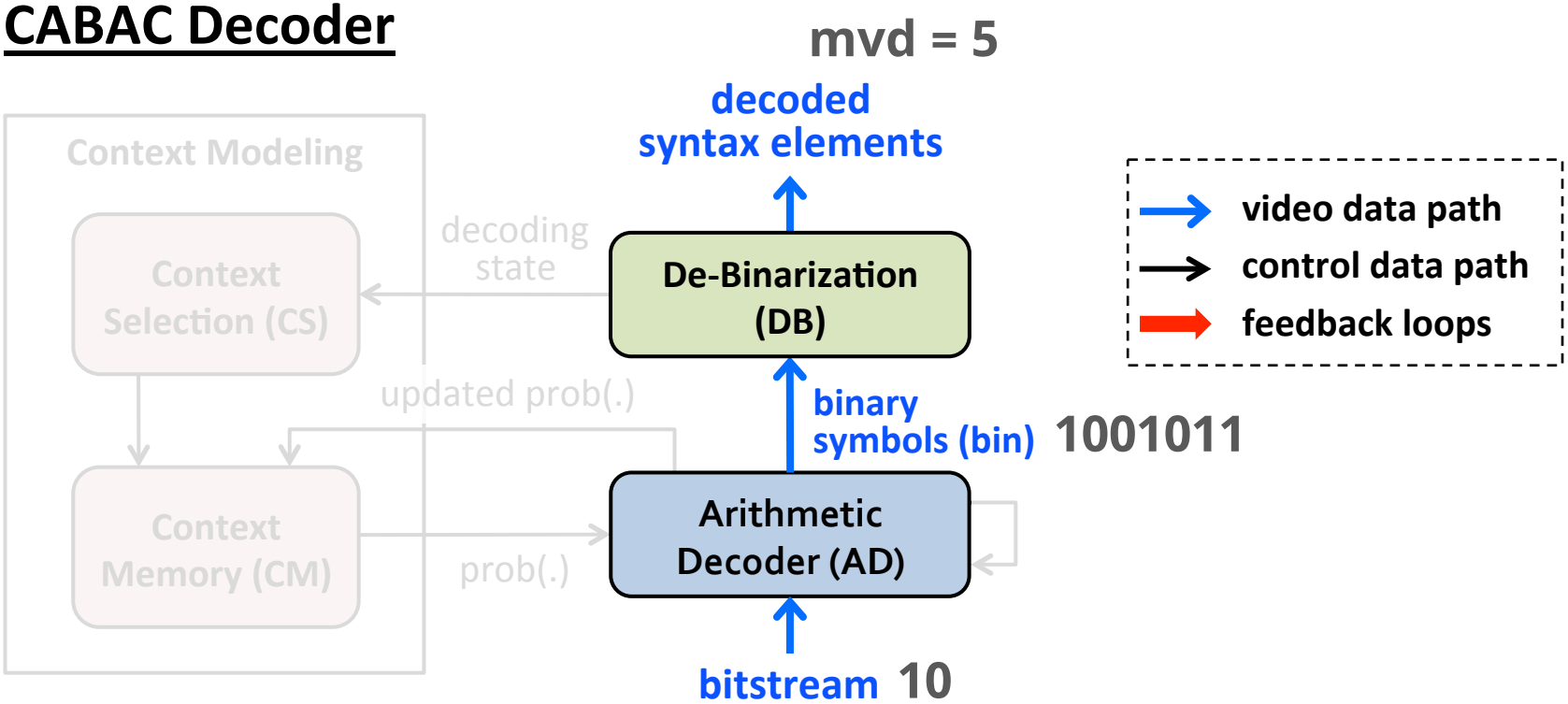
Variable-Length Code: Use **less bits** to code **higher-probability** messages

CABAC is **adaptive** to changing context models

CABAC processing is highly **effective** but also highly **adaptive**

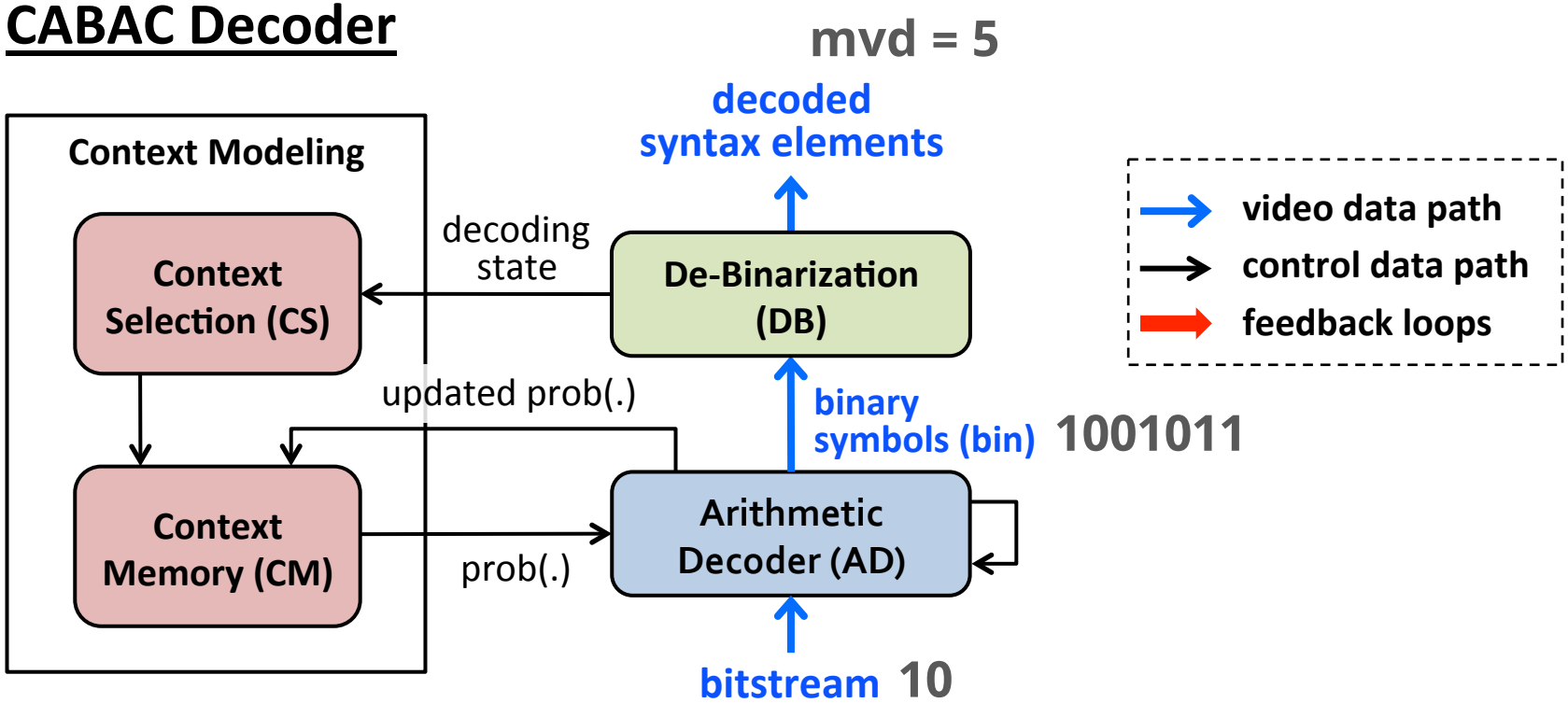
Key Feedback Loops in CABAC Decoding

CABAC Decoder



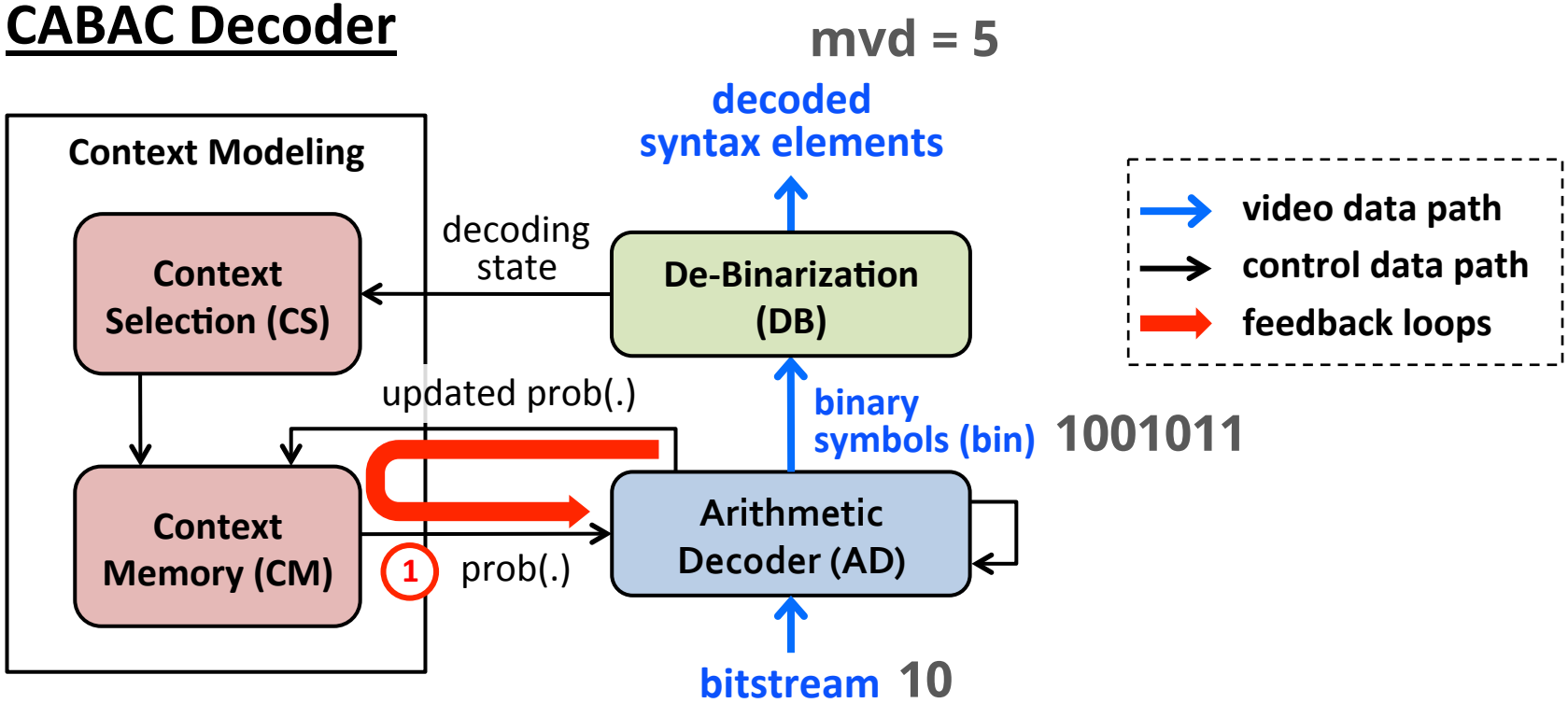
Key Feedback Loops in CABAC Decoding

CABAC Decoder



Key Feedback Loops in CABAC Decoding

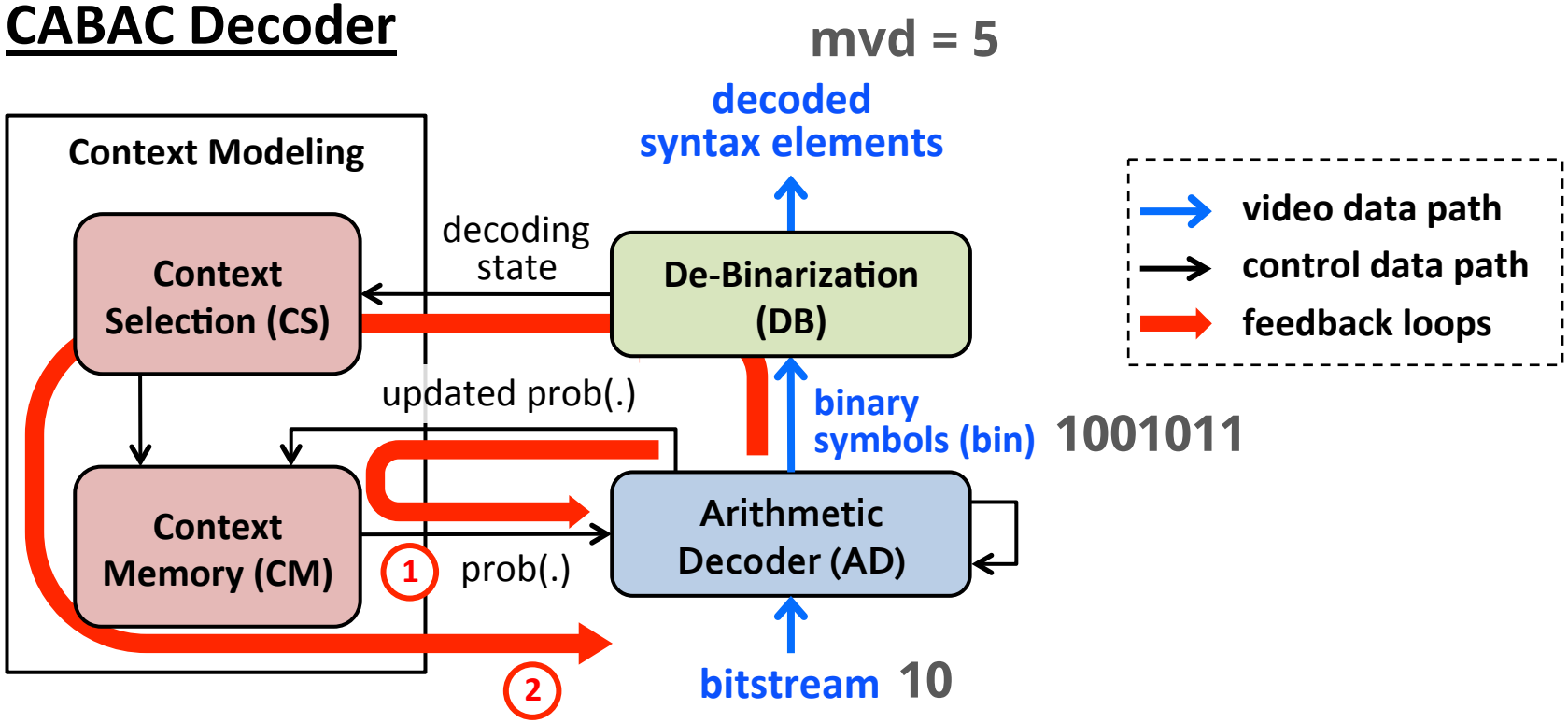
CABAC Decoder



- Adaptive context for accurate probability estimation (loop 1)

Key Feedback Loops in CABAC Decoding

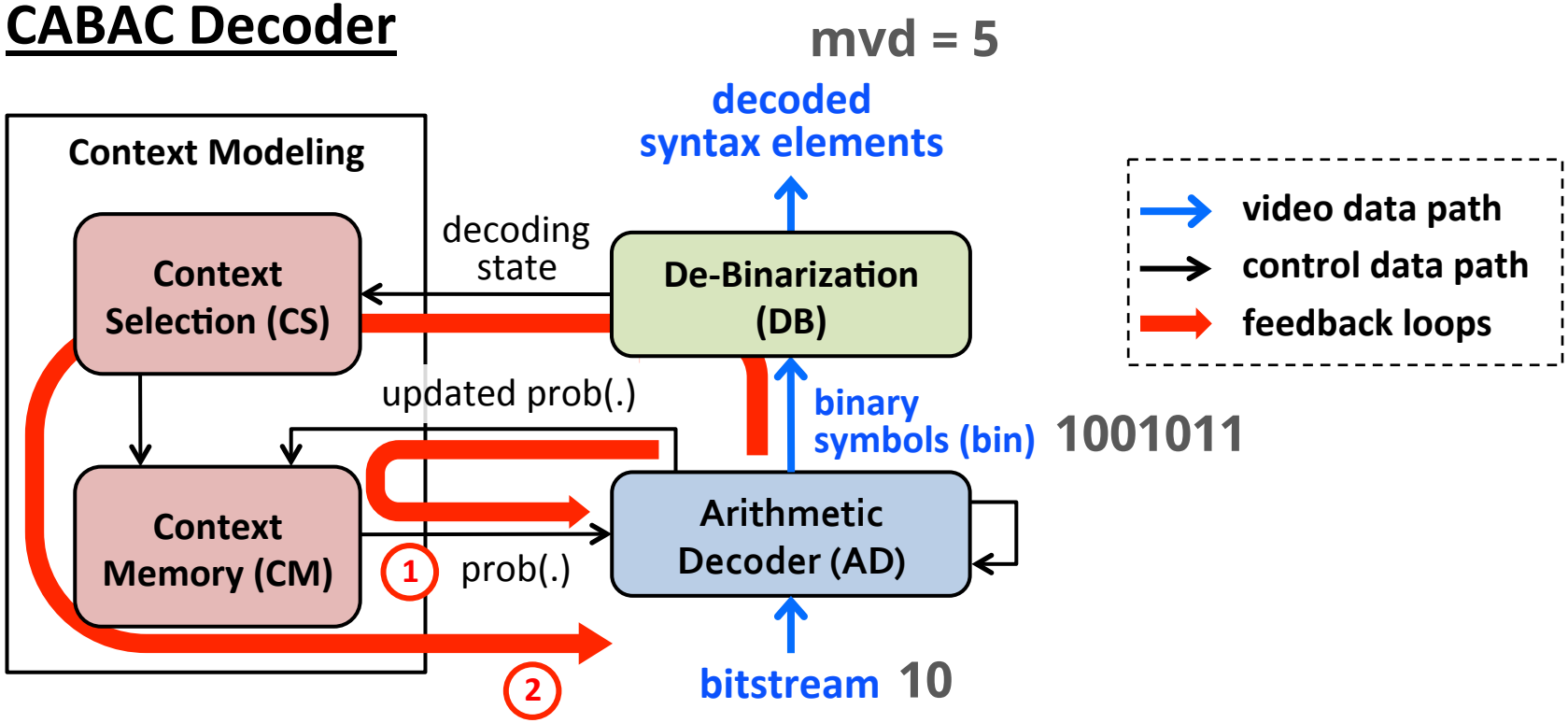
CABAC Decoder



- Adaptive context for accurate probability estimation (loop 1)
- Variable-length code contributes to bin-to-bin dependency (loop 2)

Key Feedback Loops in CABAC Decoding

CABAC Decoder

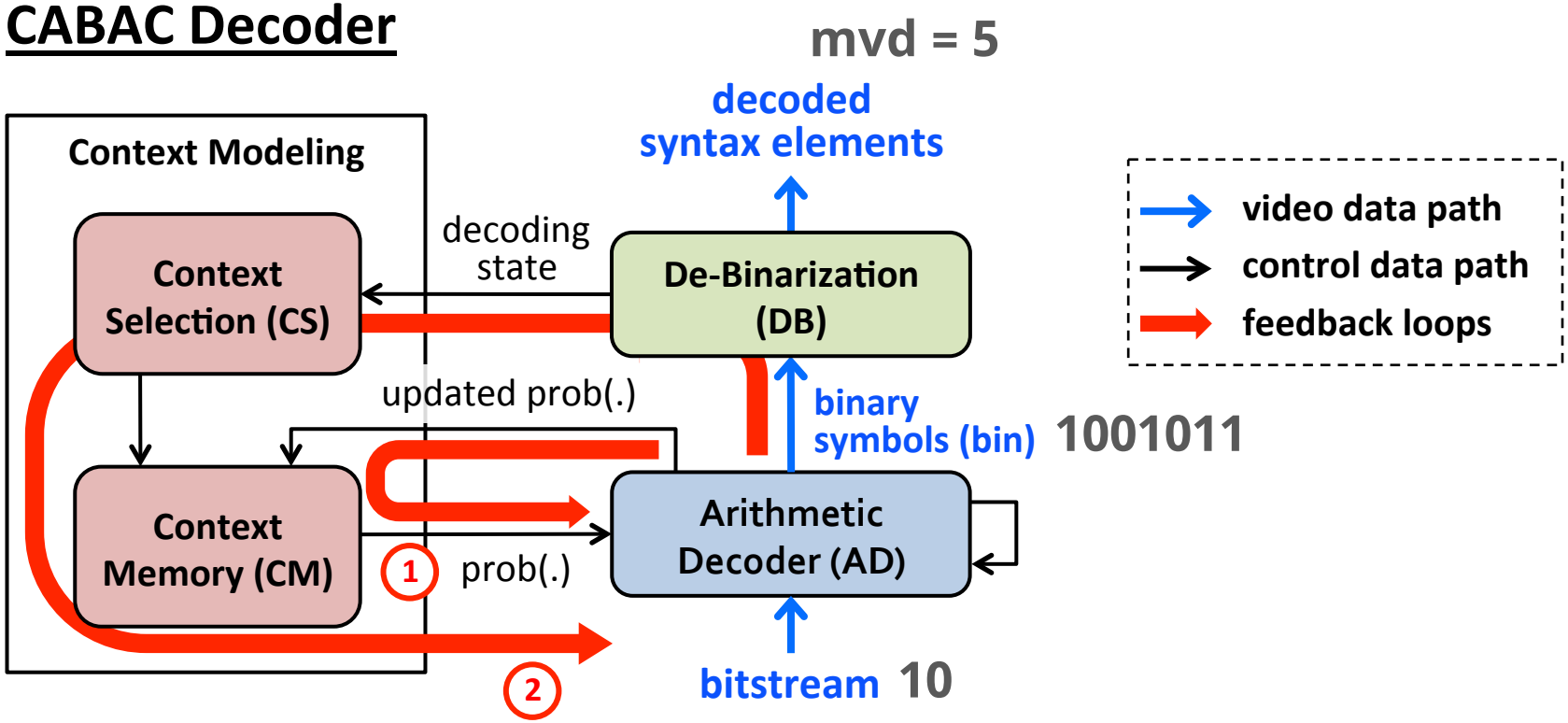


- Adaptive context for accurate probability estimation (loop 1)
- Variable-length code contributes to bin-to-bin dependency (loop 2)

Feedback loops restrict the parallelism of CABAC

Key Feedback Loops in CABAC Decoding

CABAC Decoder



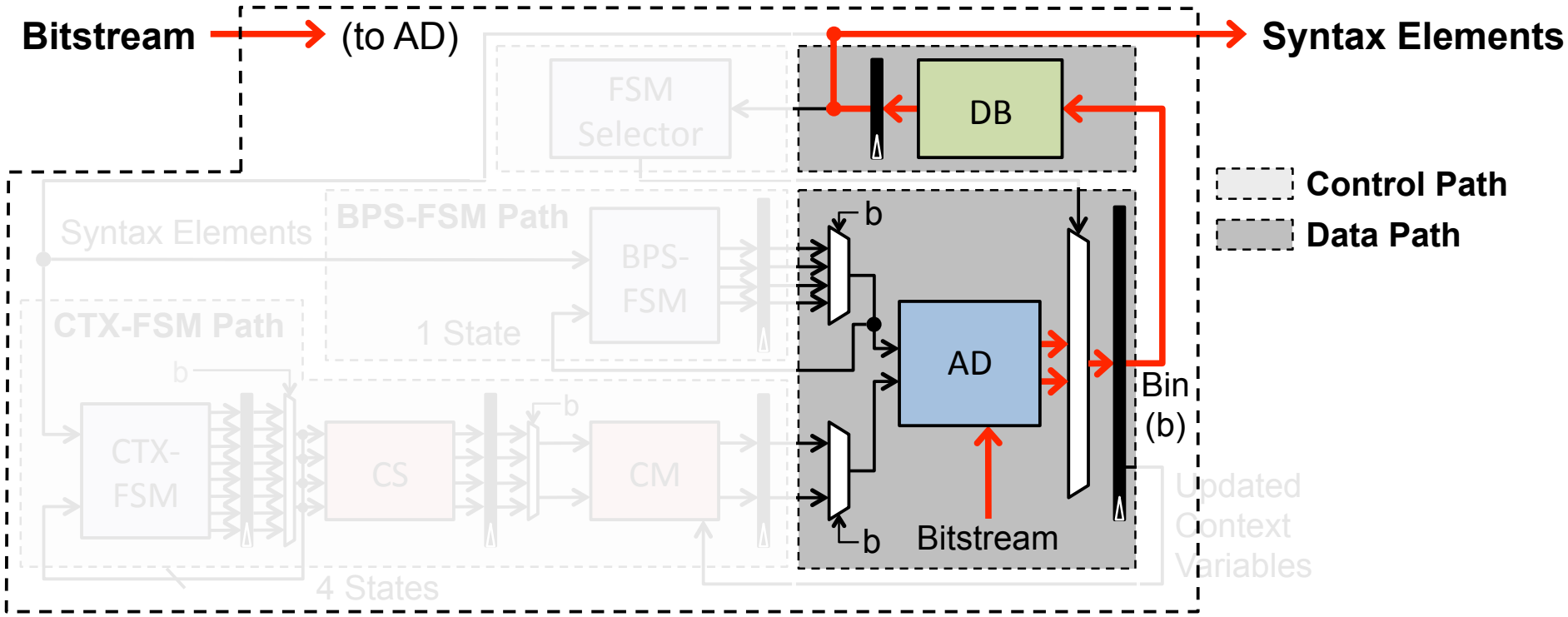
Decoding of bins

- **Context-coded Bins:** Require a context model for each bin
- **Bypass Bins:** Do not require context modeling

Proposed HEVC CABAC Decoder Architecture

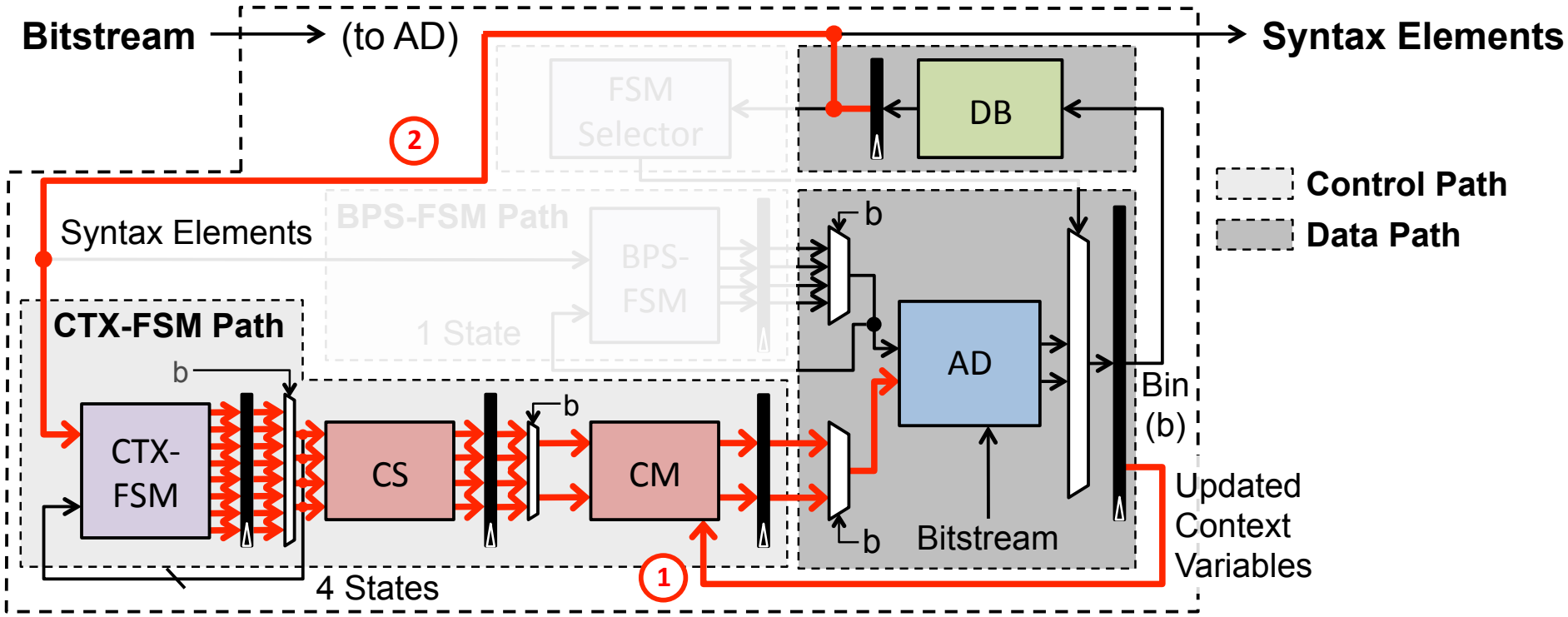
- **Goal**: to achieve the **highest throughput possible** in **bin/sec**

Proposed HEVC CABAC Decoder Architecture



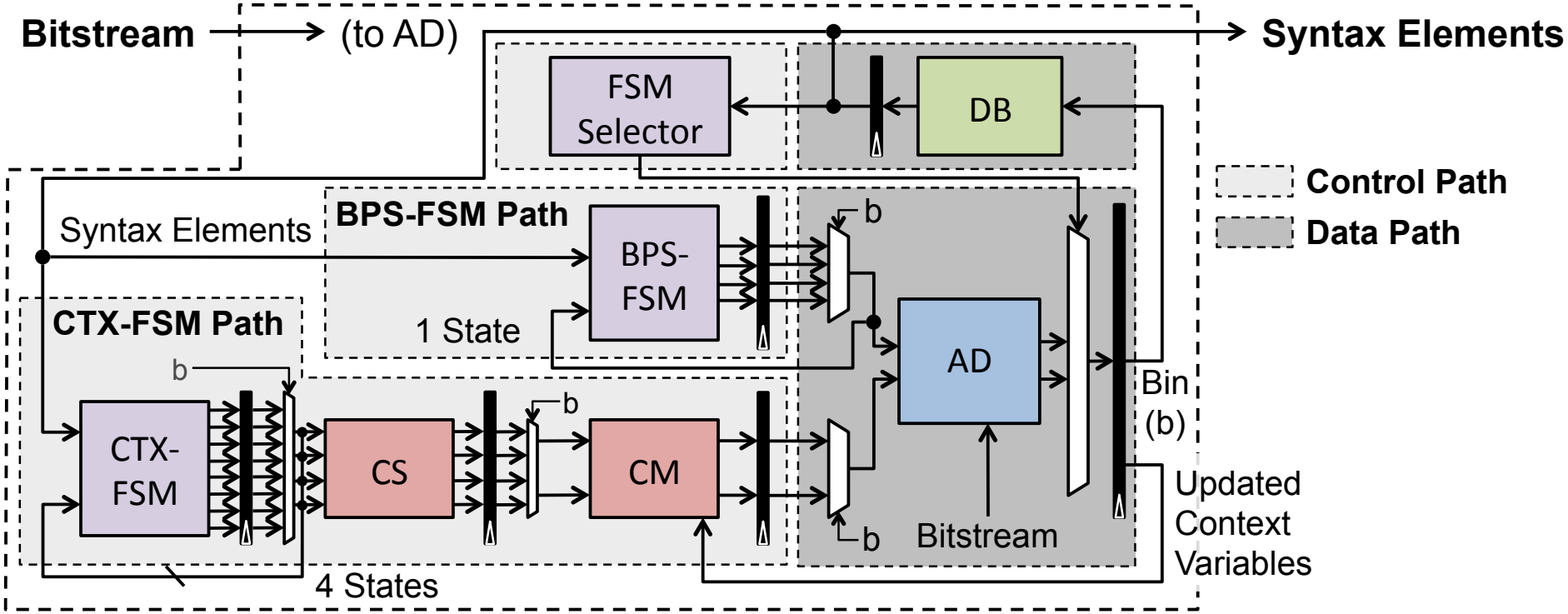
- **Goal**: to achieve the **highest throughput possible** in **bin/sec**

Proposed HEVC CABAC Decoder Architecture



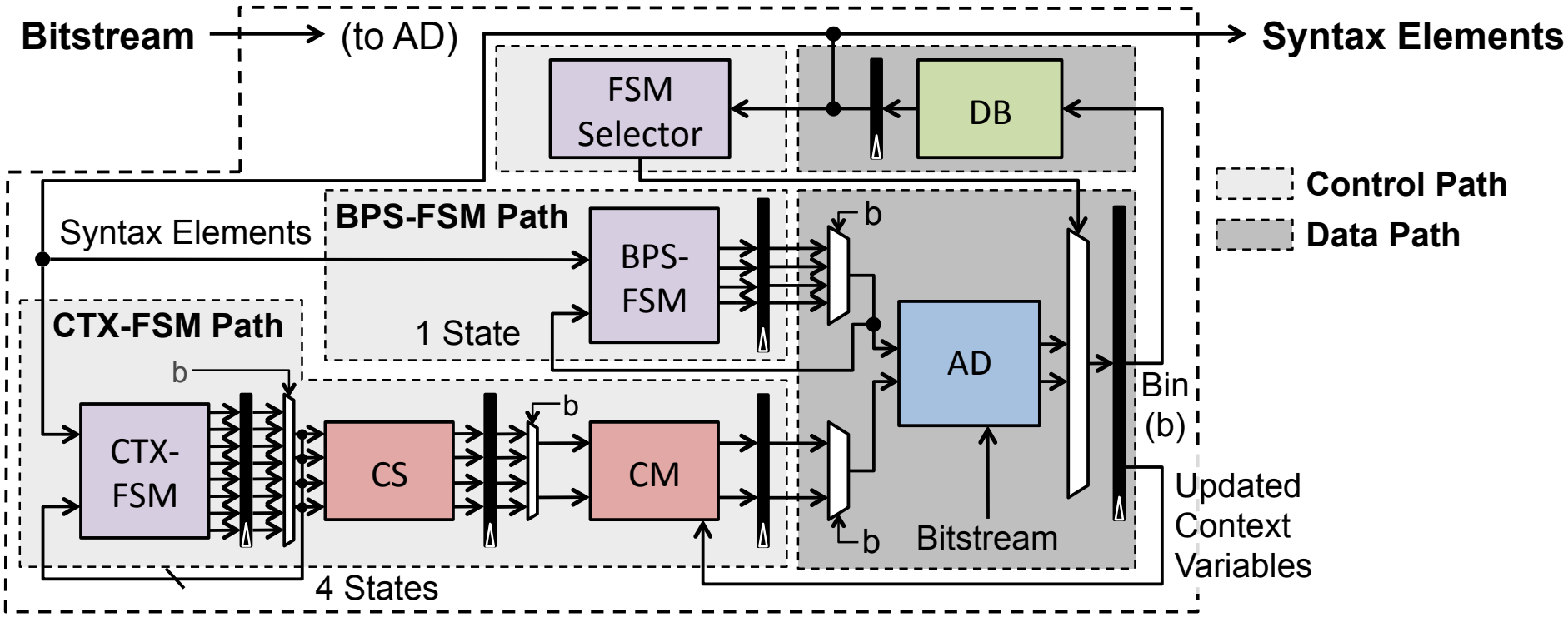
- **Goal**: to achieve the **highest throughput possible** in **bin/sec**

Proposed HEVC CABAC Decoder Architecture



- **Goal**: to achieve the **highest throughput possible** in **bin/sec**

Proposed HEVC CABAC Decoder Architecture

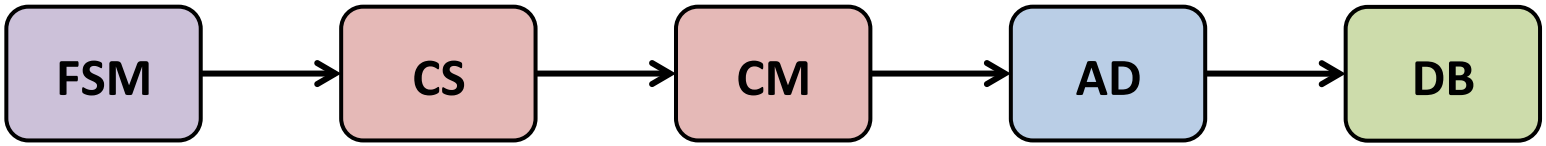


- **Goal**: to achieve the **highest throughput possible** in **bin/sec**

- bin/sec = bin/cycle × cycle/sec
  - cycle/sec → **increase the clock rate**
  - bin/cycle → **decode more bins per cycle**

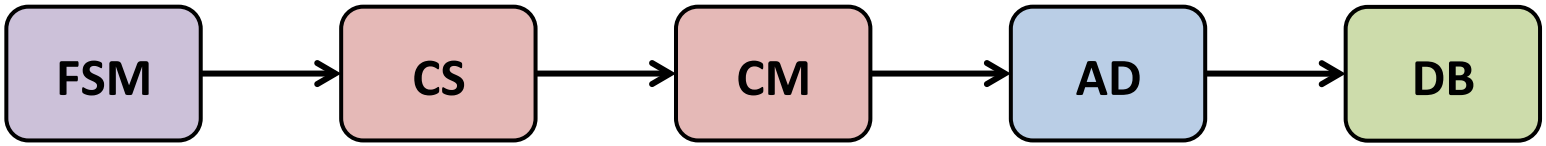
Feature 1: Deeply Pipelined Architecture

- Longest timing path in CABAC (w/o pipelining)

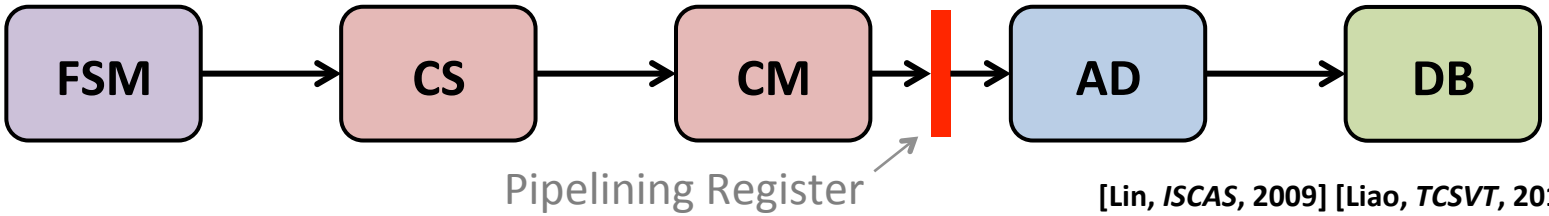


Feature 1: Deeply Pipelined Architecture

- Longest timing path in CABAC (w/o pipelining)



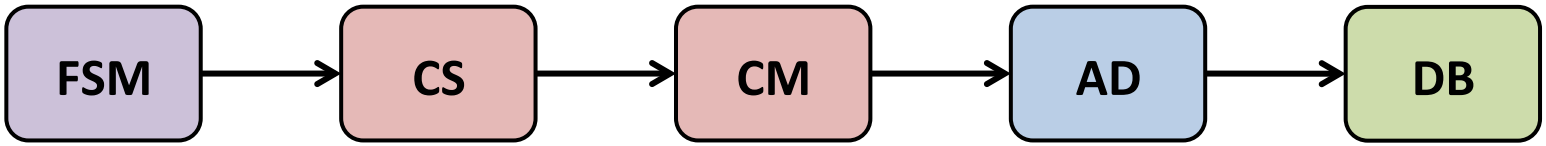
- A popular design: 2-stage pipelining



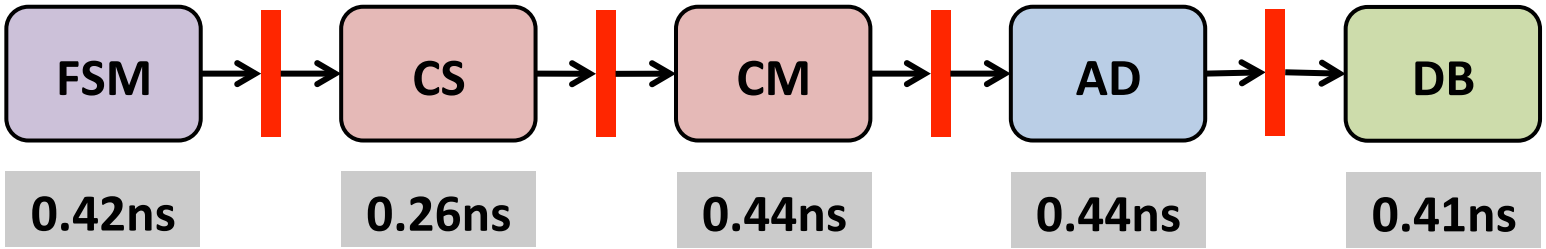
[Lin, *ISCAS*, 2009] [Liao, *TCSVT*, 2012]

Feature 1: Deeply Pipelined Architecture

- Longest timing path in CABAC (w/o pipelining)



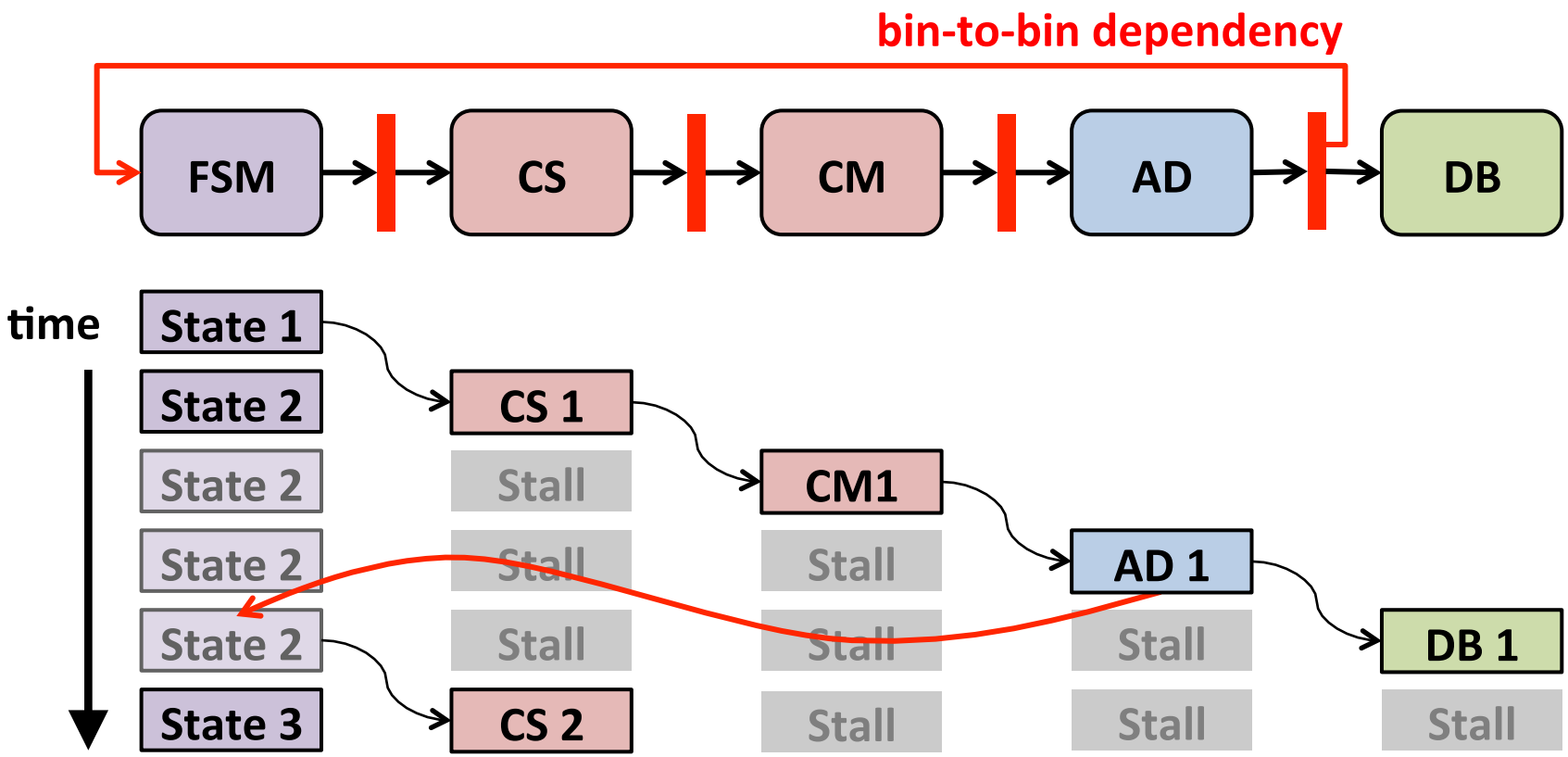
- **Proposed deeply pipelined architecture: 5-stage pipelining**



[synthesized in 45nm SOI, only combinational logic included]

At least **2.2× higher clock rate** than 2-stage pipelined design

Feature 1: Deeply Pipelined Architecture

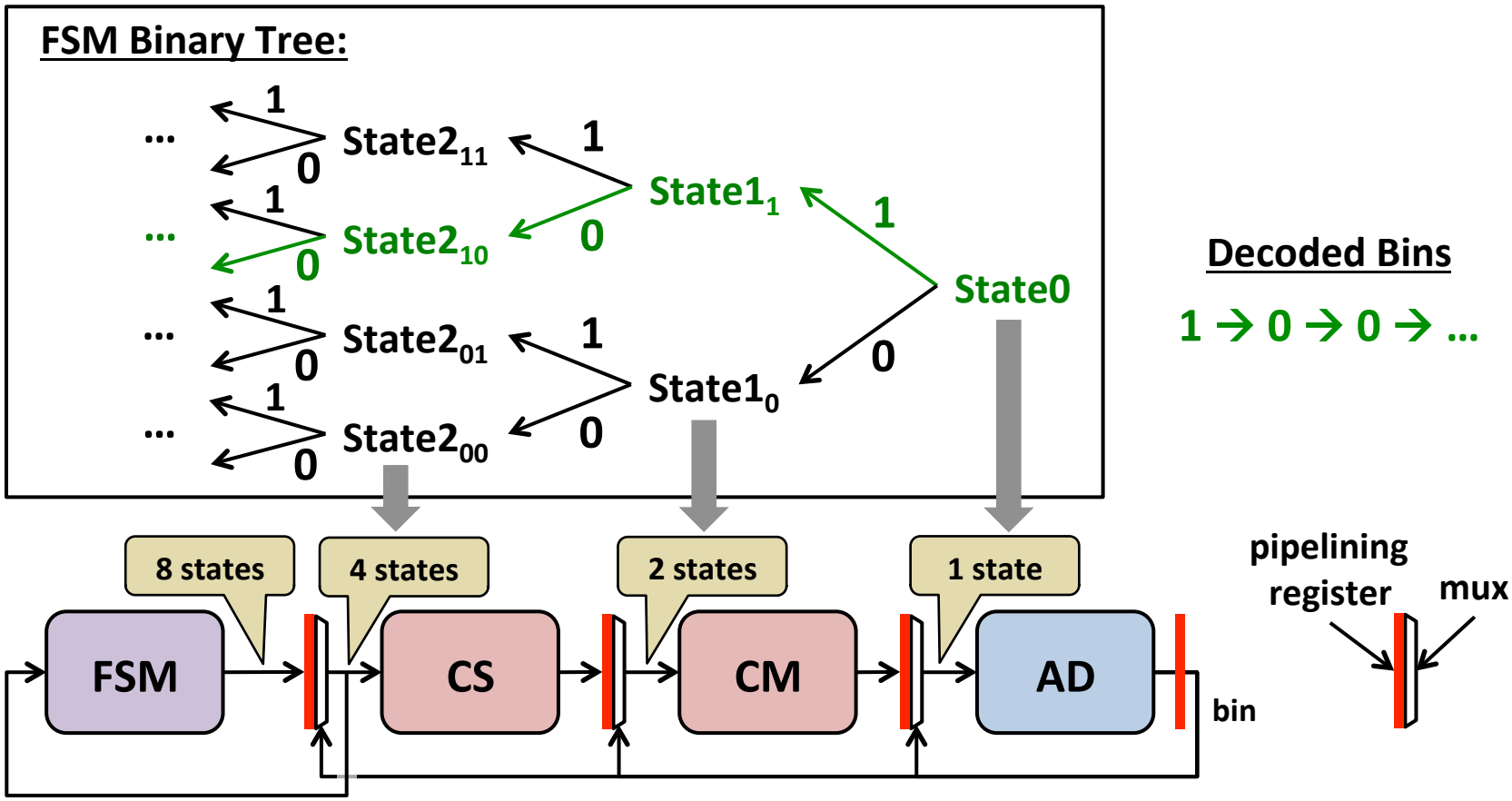


- Feedback loops introduce **Stalls**, **reduce** bin/cycle

Stalls needs to be removed to take advantage of pipelining

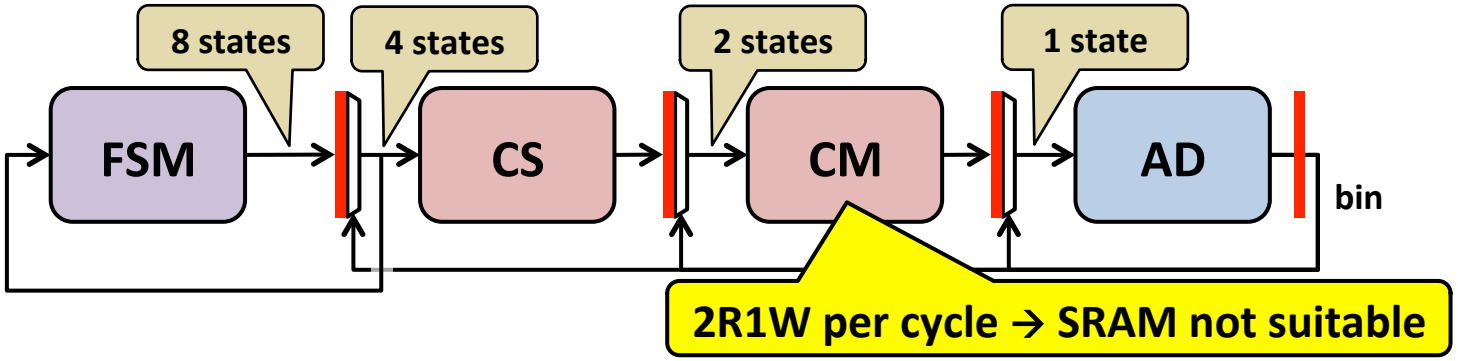
Feature 1: Pipelining with State Prefetch Logic

- FSM prefetches **next 2 possible states** based on the current bin binary value
- Optimized between **number of stalls** and **number of states**



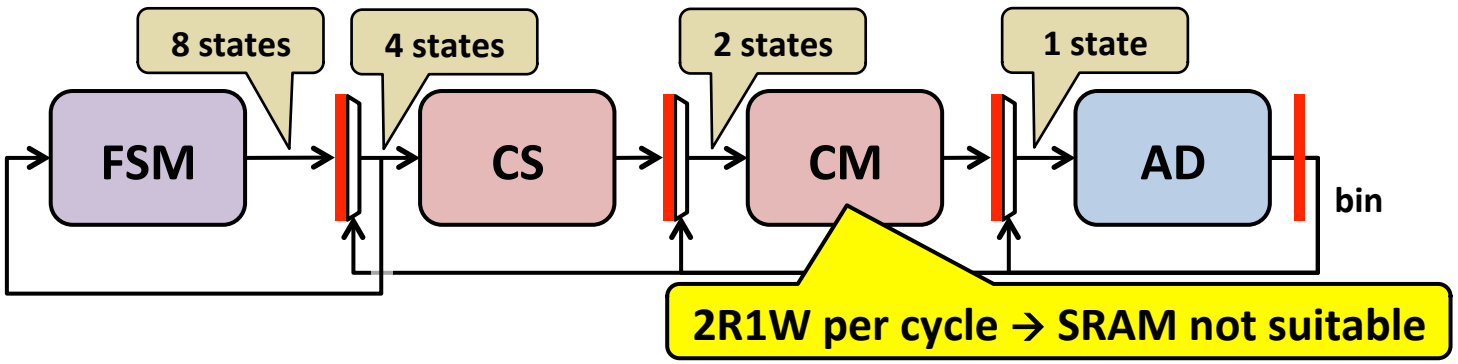
State Prefetch Logic **removes majority of the stalls** from pipelining

Feature 2: Latch-Based Context Memory



Memory Type	Pros	Cons
SRAM	Compact	No Multi-R/W
Registers	Fast	Power/Area Hungry
Latches	Compact, Low Power	**Timing Issue**

Feature 2: Latch-Based Context Memory

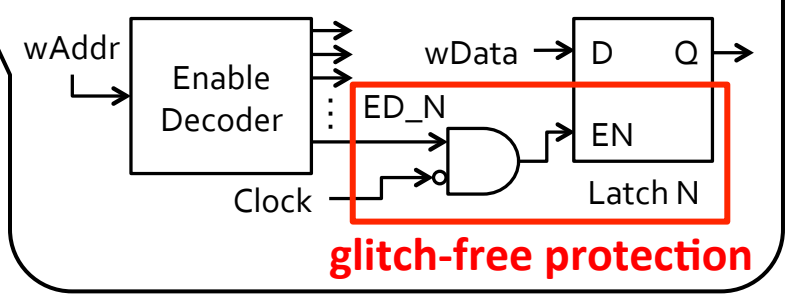


Memory Type	Pros	Cons
SRAM	Compact	No Multi-R/W
Registers	Fast	Power/Area Hungry
Latches	Compact, Low Power	**Timing Issue**

Memory Type	Eq. Gate Count	Avg. Power (mW)*
SRAM	11.7k	8.10
Registers	20.2k	13.10
Latches	**13.4k**	**4.68**

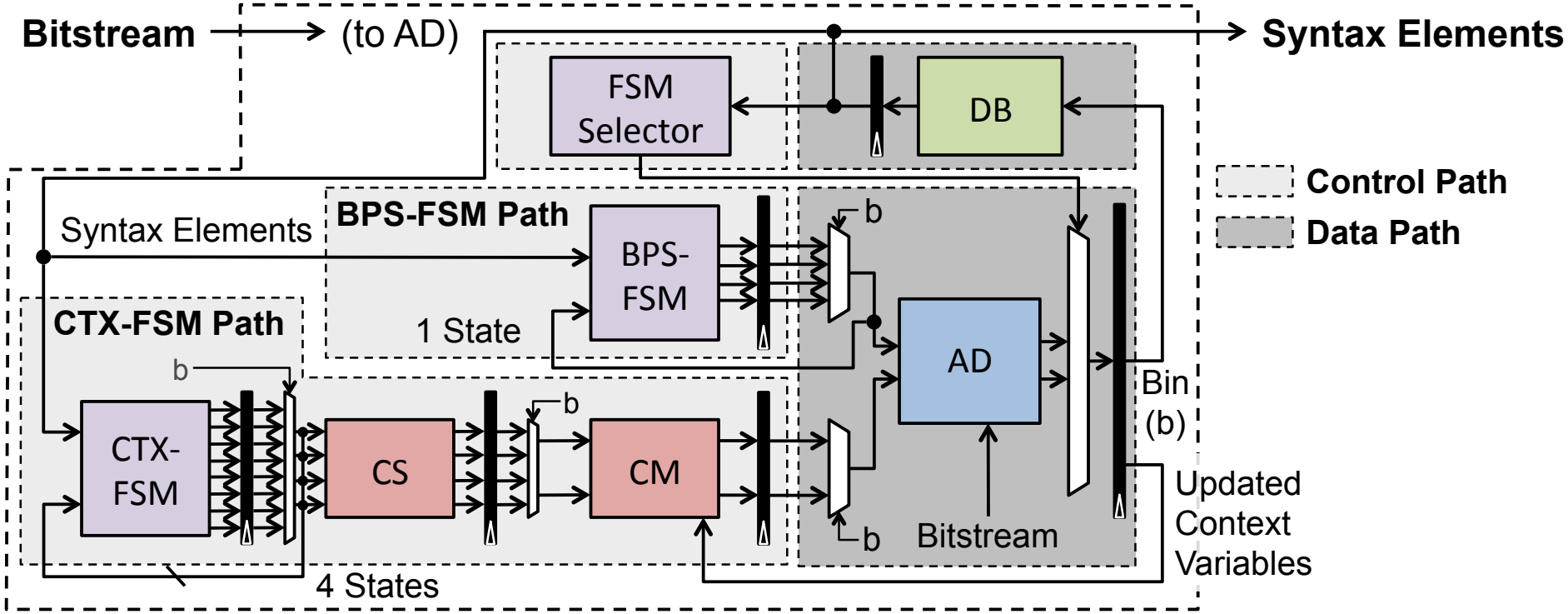
* at 2 GHz clock rate

Solution:

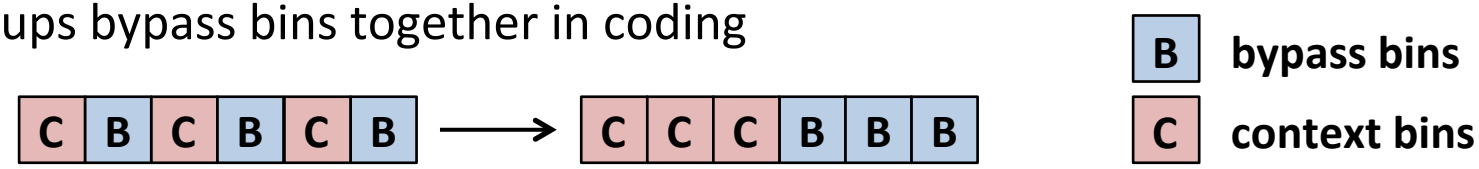


Latch-based CM supports **multiple R/W with low power consumption**

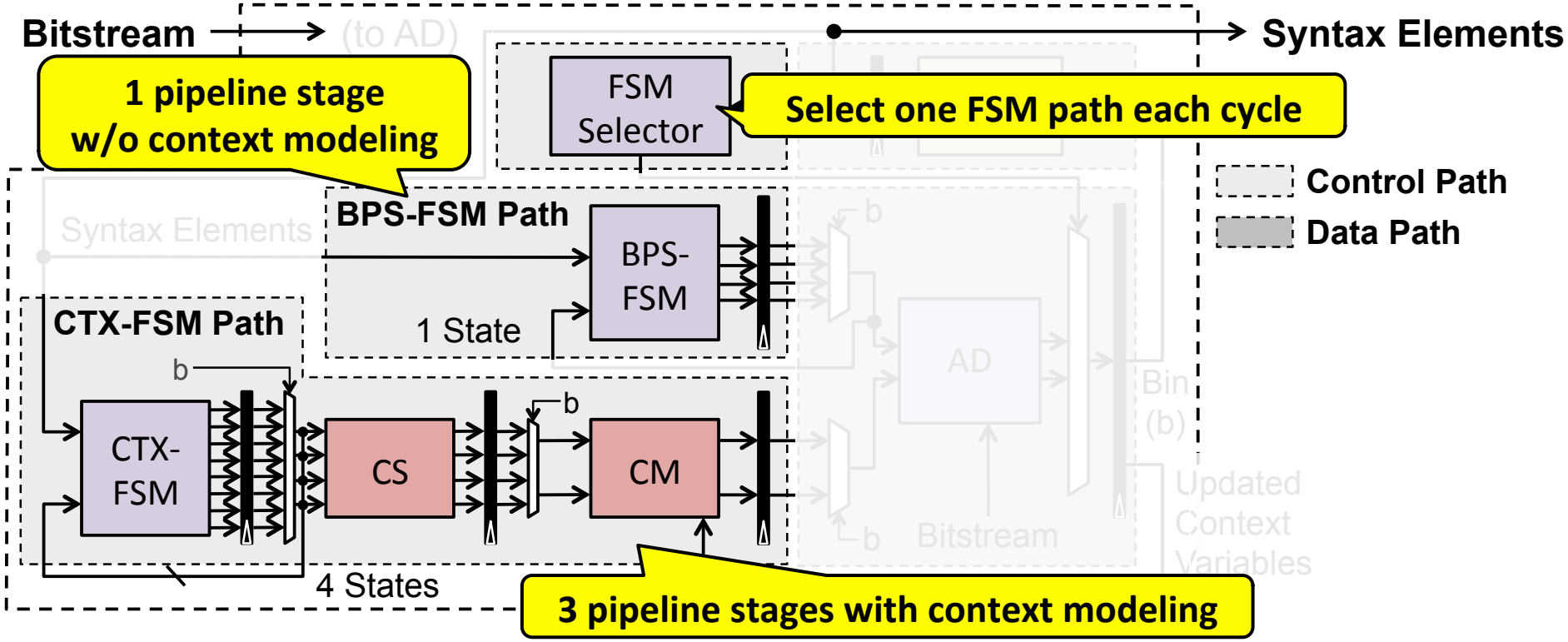
Feature 3: Separate FSM for Bypass Bins



- Bypass bins do **not** require context modeling
- HEVC groups bypass bins together in coding

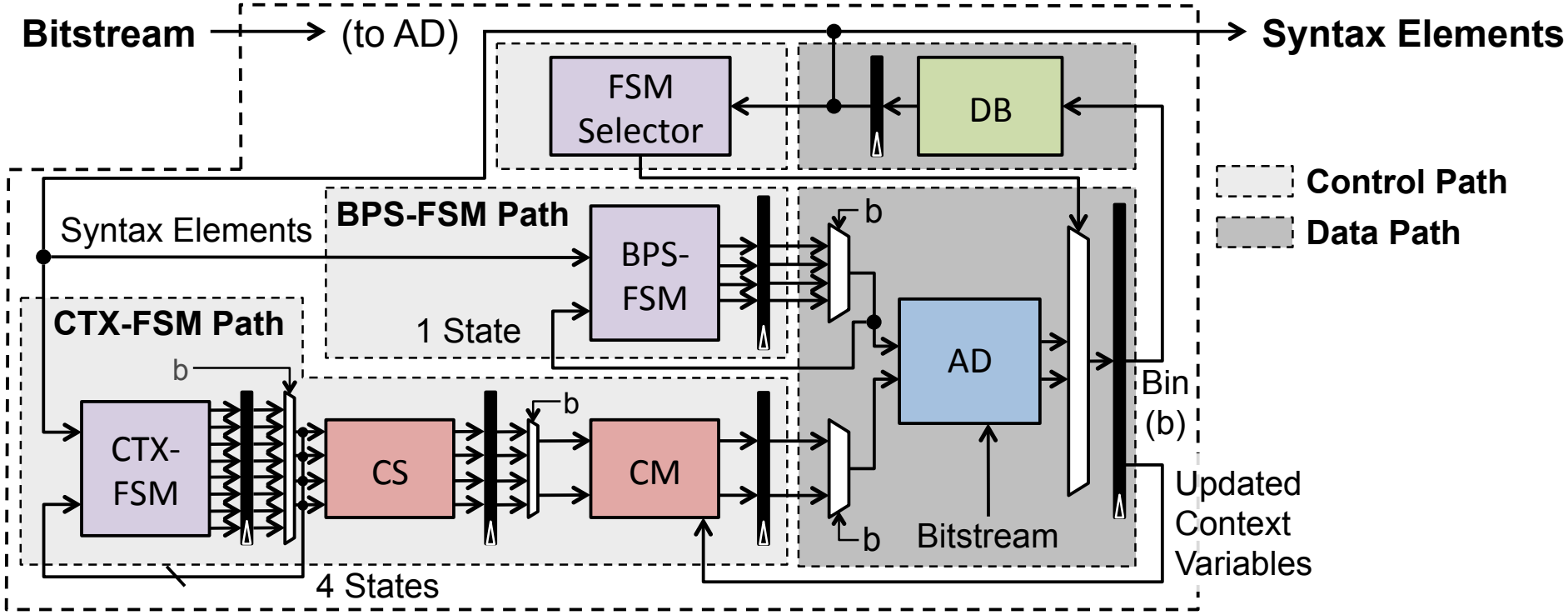


Feature 3: Separate FSM for Bypass Bins



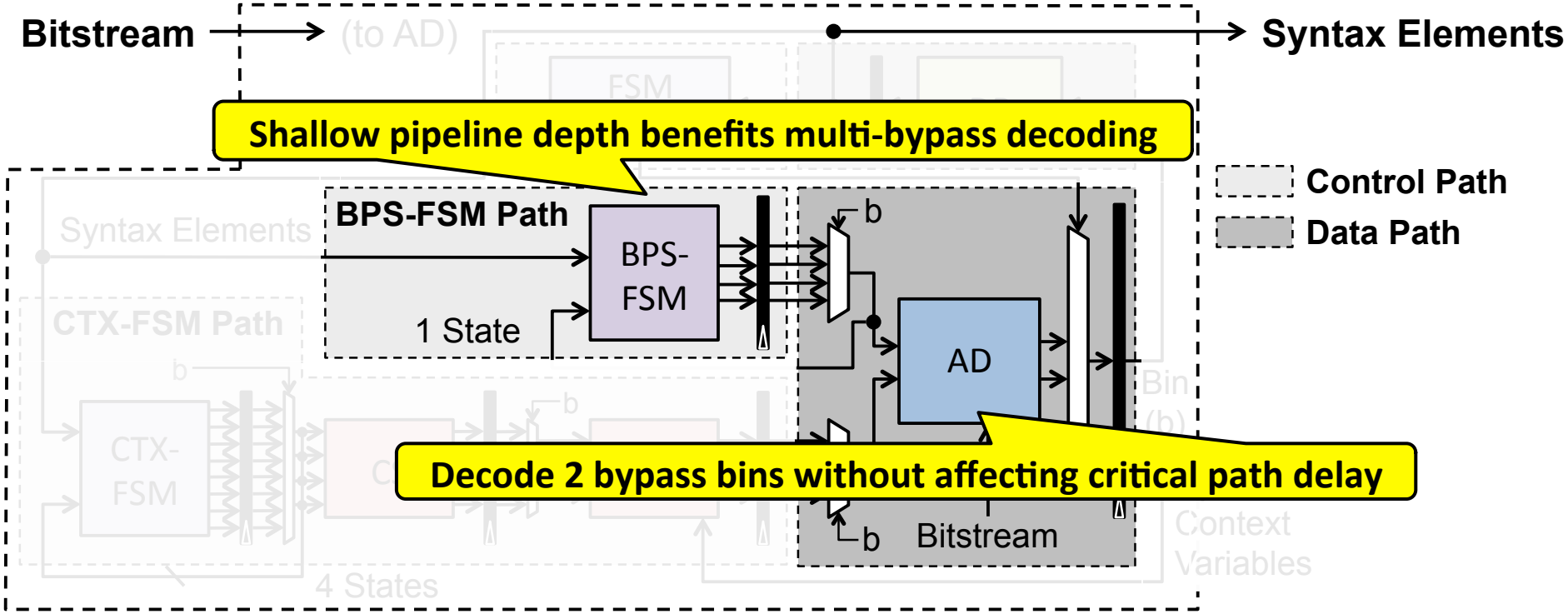
Separate FSM reduces the pipeline depth for bypass bins to **avoid stalls**

Feature 4: Multi-Bypass-Bin Decoding



- **High bitrate** bitstream tends to have **more grouped bypass bins**
- Decoding of bypass bins is fast

Feature 4: Multi-Bypass-Bin Decoding



Multi-bypass-bin decoding **increases bin/cycle at the same clock rate**

Feature Summary

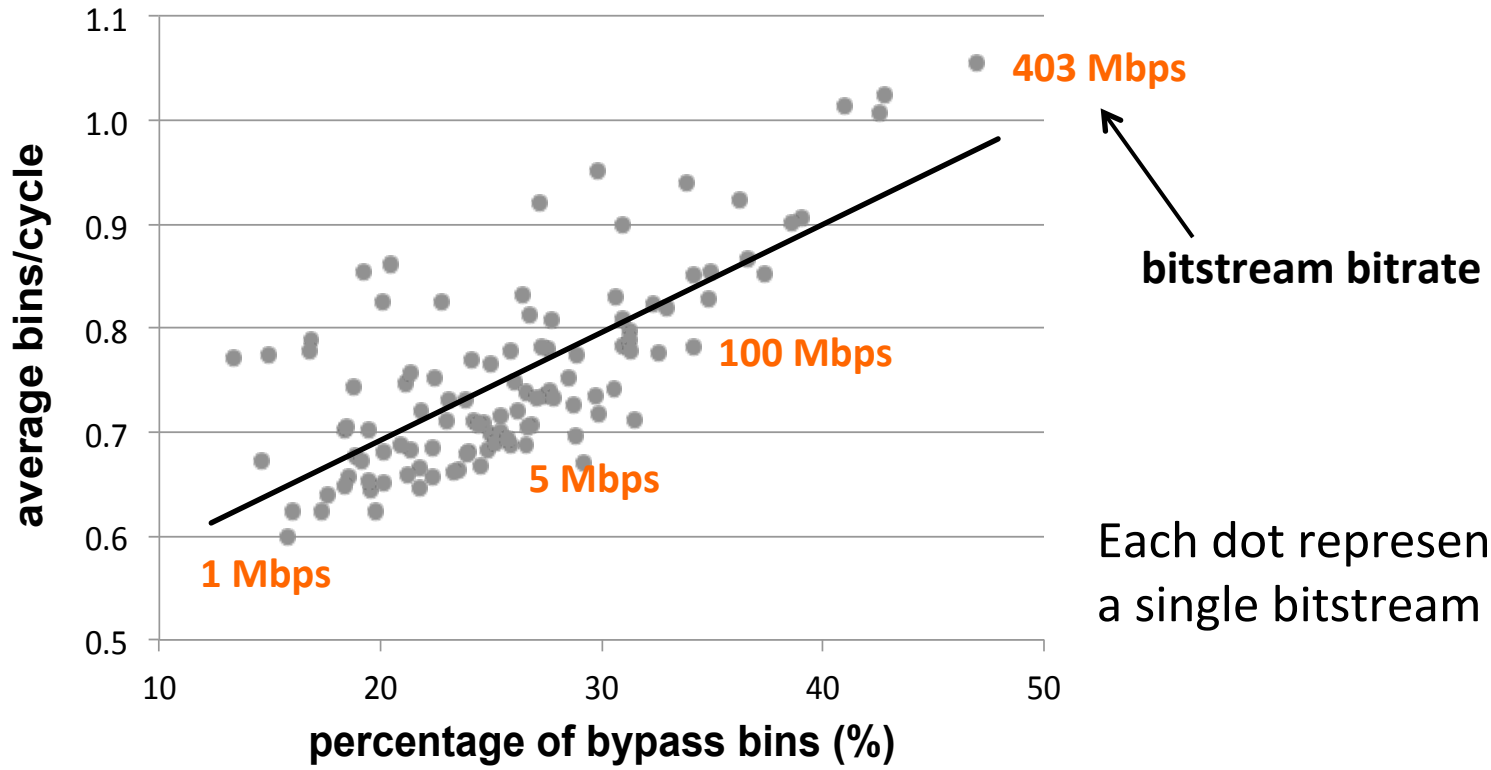
Throughput = bin/cycle × cycle/sec

Increase cycle/sec	
• **5-Stage Deep Pipelining**	2.2× faster than 2-stage design
• **State Prefetch Logic**	reduce impact of stalls down to 12%
• **Latch-based Memory**	multiple R/W at lower power
Increase bin/cycle	
• **Separate FSM**	Increase throughput by up to 33%
• **2 Bypass Bins / Cycle**	Increase throughput by up to 15%

*** All tested with common test bitstreams**

Performance Evaluation

- High bitrate bitstream → More bypass bins → High throughput
- Reaches up to **1.06 bin/cycle** for sequence at 400 Mbps



Each dot represents a single bitstream

CABAC performance increases with higher bitrate sequences

Result Comparison

- Throughput up to **2000 Mbin/sec** at **1.9 GHz** clock rate
- **Real-time** decoding of **8K UHD @ 120 fps** bitstreams
- Can trade-off throughput for **low power** (voltage scaling)

Result Comparison

- Throughput up to **2000 Mbin/sec** at **1.9 GHz** clock rate
- **Real-time** decoding of **8K UHD @ 120 fps** bitstreams
- Can trade-off throughput for **low power** (voltage scaling)

	Lin [7]	Liao [8]	Choi [11]	**This Work**
Standard	AVC	AVC	HEVC	HEVC
Tech.	UMC 90nm	UMC 90nm	Samsung 28nm	IBM 45nm SOI
Gate Count	82.4k	51.3k	100.4k[1]	85.3k
SRAM Size	N/A	179B	N/A	N/A
Max. Freq. (MHz)	222	264	333	1900
Bins/Cycle	1.96	1.84[2]	1.30	1.06[3]
Throughput (Mbin/s)	435	486	433	2014

[1] without the bitstream parser buffer
[2] with the test bitstream bit-rate at 130 Mbps
[3] with the test bitstream bit-rate at 403 Mbps

[7] Lin, *ISCAS*, 2009 [8] Liao, *TCSVT*, 2012 [11] Choi, *Elec. Letters*, 2013

Take-Home Messages

- **A high-throughput CABAC decoder**
  - *Deeply pipelined* (increase **cycle/sec**)
  - *Multi-bypass-bin decoding* (increase **bin/cycle**)
- **Throughput up to 2000 Mbin/s**
  - Real-time decoding of 8K UHD @ 120 fps
- **Trade-off throughput for low power**
  - Voltage scaling can be applied to the entire HEVC decoder

Thank you!

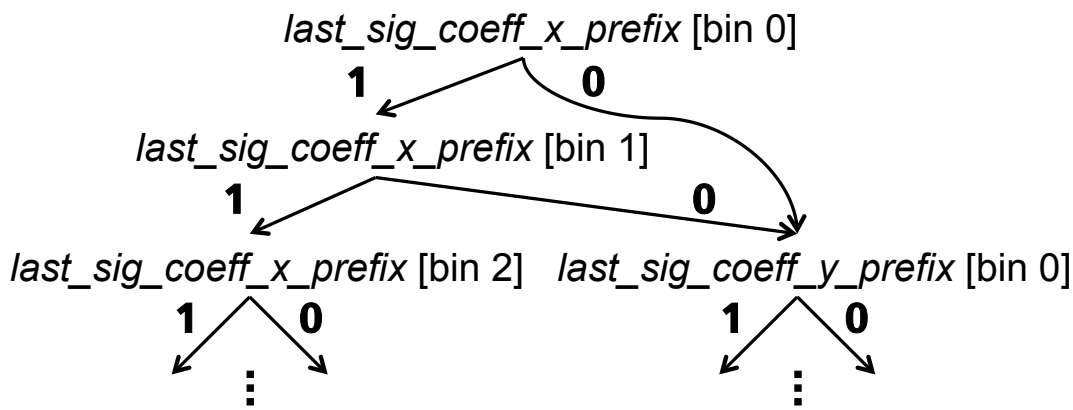
Contact: yhchen@mit.edu

Backup Materials

State Prefetch Logic

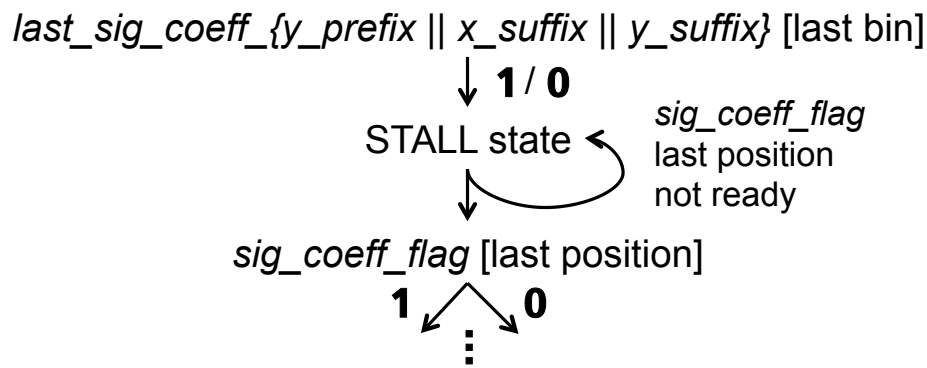
Case 1: with State prefetch

- Binary tree expanded
- No stalls left



Case 2: w/o State prefetch

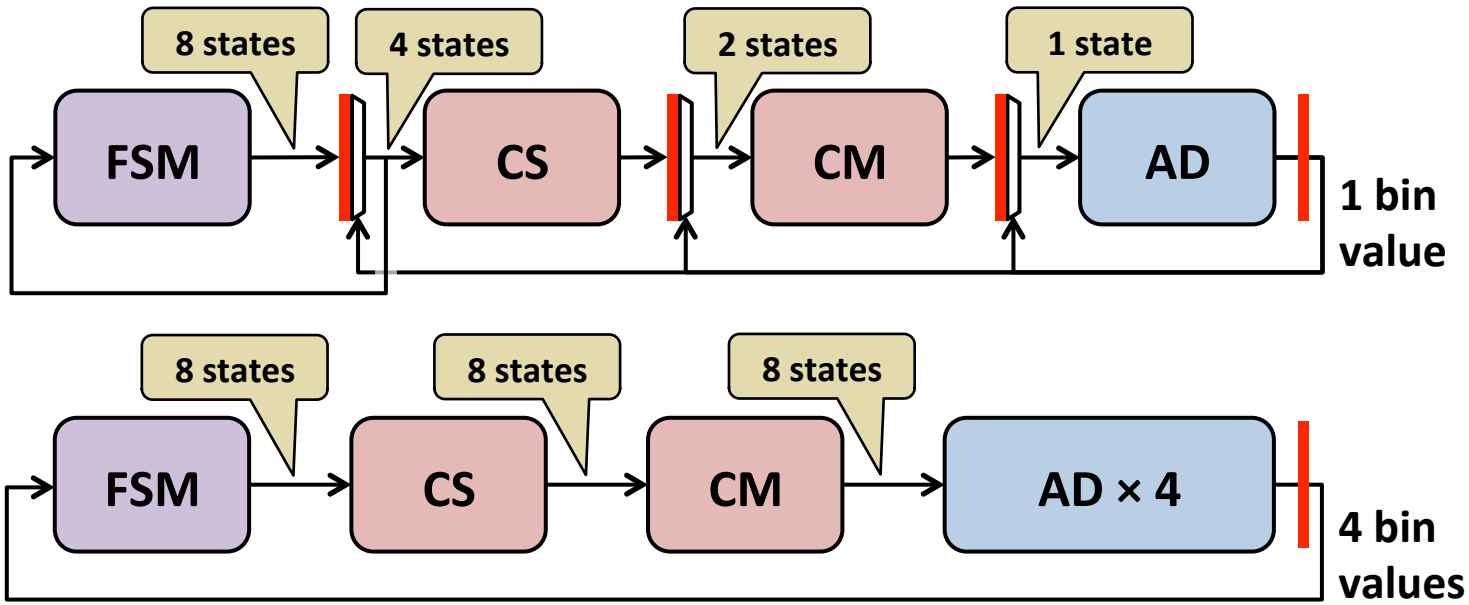
- Stalls are kept
- Save dozens of states



Pipelining vs. Multi-Bin Decoding

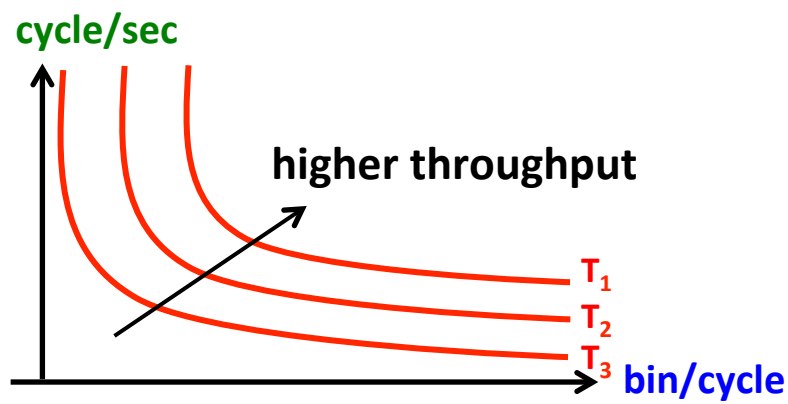
Pipelining
(cycle/sec)

Multi-Bin Decoding
(bin/cycle)



- Deep multi-bin decoding suffers from **higher complexity** & **reduced clock rate**
- This work: increase the clock rate for all bins and further speed up processing of bypass bins

The Design Space



The Design Space