Intra-Frame Compression for Bus Traffic and Memory Reduction

Hung-Chih Kuo Youn-Long Lin
Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan

THEDA.DESIGN.

0

Outline

- Motivation
- Related Works
- Proposed Algorithm
- Experimental Results
- Summary

1

Applications & Advantages of Intra-frame Encoding

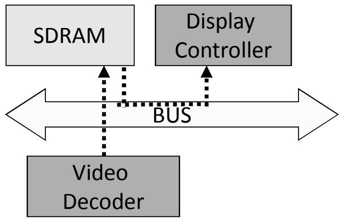
Applications – digital cinema, surveillance, digital photography, medical imaging, etc.

Advantages
- Ease of editing – each frame can be processed individually
- Good for variable bandwidth network – transmission loss or delay of one frame won't affect other frames
- Low complexity – require less computation and bus traffic

Representative Intra-frame Encoders

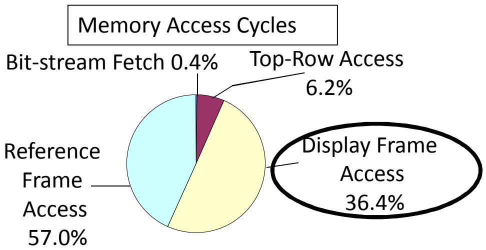
- Motion JPEG: widely adopted in digital cameras because of its low hardware cost
- Motion JPEG2000: proposed for application that requires high resolution or lossless video quality such as digital cinema and medical imaging
- H.264/AVC intra-frame encoder: propose novel intra coding tools such as "Intra Prediction" to achieve better coding performance
- HEVC intra-frame encoder: will also employ intra coding tools

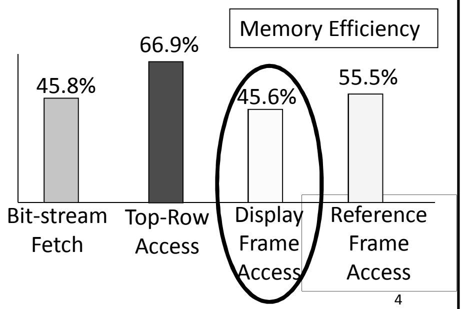
Profiling An H.264/AVC Decoder



Decoder Settings

Video resolution	1080p
GOP	IBP
Number of total frames	60
Number of ref. frames	4
Search range	64x64

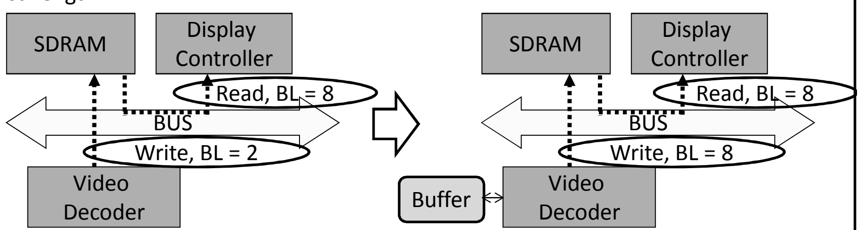


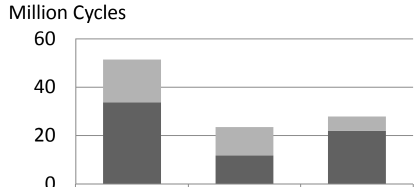


4

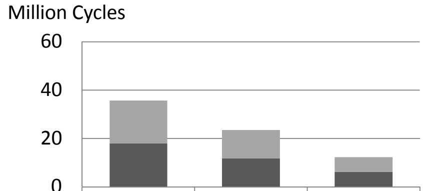
Bus Traffic for Accessing Display Frames

BL = Burst Length





	Total cycles = 51.47	Data cycles = 23.5	Penalty cycles = 27.97
Read	17.73	11.75	5.98
Write	33.74	11.75	21.99



	Total cycles = 35.71	Data cycles = 23.5	Penalty cycles = 12.21
Read	17.73	11.75	5.98
Write	17.98	11.75	6.23

5

Bus Traffic Reduction Technologies

- DRAM penalty cycles - DRAM controller scheduling algorithms, DRAM address mappings
- Redundant data access - memory-efficient architectures
- Data access cycles – frame compression algorithms
  1. Reduce data access -> reduce penalty
  2. Can be integrated into various systems
  3. Can work together with other technologies

6

Previous Frame Compression Algorithms

Type	References	Advantages	Disadvantages
Lossy	[LeRL07], [CDTC08], [IvMo08], [ChTC09], [SZJG10], [SKSP10], [MaSe11], [GAMR11], [VoLK11]	• Guarantee compression ratio • Save both bus traffic and memory space	• **Cause video quality loss**
Lossless	[LZWF07], [SoSh07], [LiLY08], [LCPK09], [KiKK09], [YCKL09], [KiKy10], [BaZG10], [DiZh10], [KLKK11], [JKLY12], [ChCh12], [SSGP12], [LJMe12]	• Preserve the video quality	• **Save bus traffic only**

- As video resolution increases, video quality become more and more important

7

Previous Lossless Frame Compression Algorithms

DRR = (1 – Compressed_Size/Original_Size)*100%

Type	References	Processing Unit	Data Reduction Ratio (DRR)	Applications
Block-based	[SoSh07], [LCPK09], [KiKK09], [KiKy10], [BaZG10], [KLKK11], [JKLY12], [ChCh12], [SSGP12]	A MxN block	Around 60%	Reference frames
Line-based	[LZWF07], [LiLY08], [YCKL09], [DiZh10], [LJMe12]	1~N pixels in a video line	30%~50%	Display frames

- Display devices show frames line by line and may use interlaced format

Previous Line-based Frame Compression Algorithms

Test Pattern: 12 1080p videos

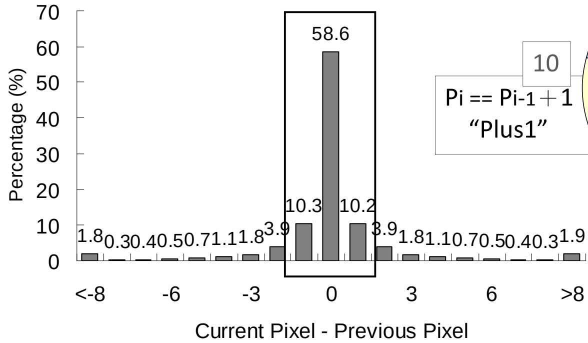
Work	Algorithm	Average DRR	Computation Resources		Local Memory (Bytes)
			Addition	Comparison	
LZWF07	Dictionary-based Coding	18.76%	0	3	3
LiLY08	Modified Hadamard Transform + Adaptive Golomb-Rice Coding	51.96%	18	0	1920
YCKL09	Dictionary-based Coding	44.39%	2	3	1
DiZh10	Integer Wavelet Transform + Adaptive Golomb-Rice Coding	17.98%	30	0	16
LJMe12	Dictionary-based Coding	31.7%	2	3	2

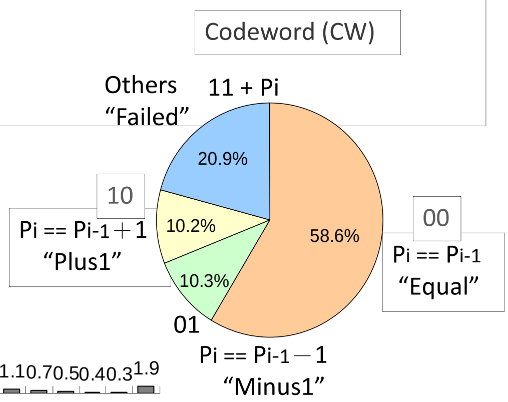
Main Idea of Yang's Algorithm

- More than 79% of differences are 0 and ±1
- Assuming

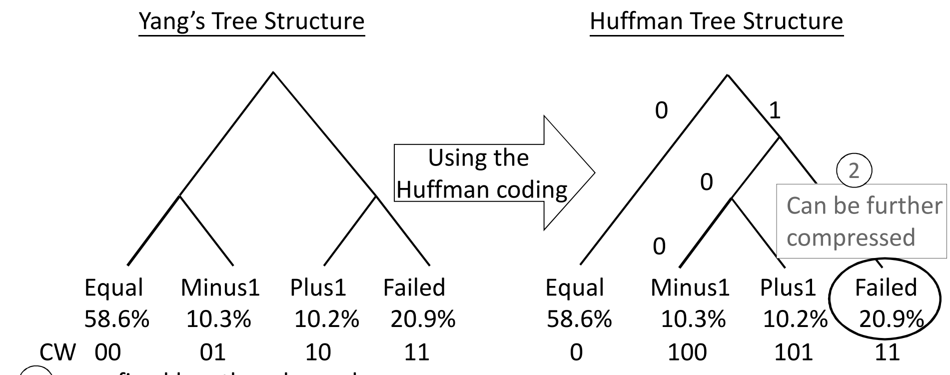
  P_i : current pixel

  P_{i-1} : previous pixel





Drawbacks of Yang's Algorithm



(1) uses fixed length codewords

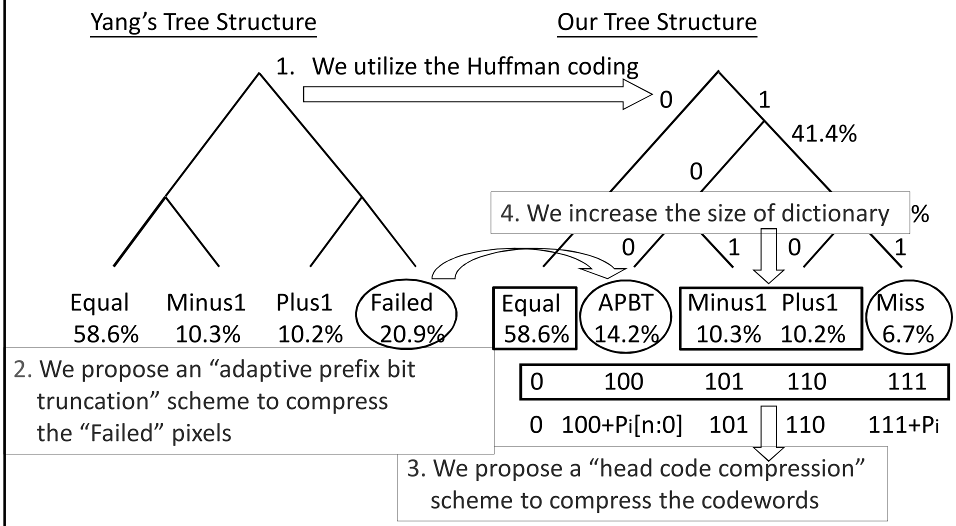
Output	00	01	10	11 + Pi	0	100	101	11 + Pi

[8–(2*79.1%+10*20.9%)]/8 = 54.1%

[8–(1*58.6+10*20.9%+3*20.5%)]/8 = 58.9%

Improve 4.8%

Four Proposed Improvements



12

Longest Prefix Match (LPM)

- We obtain it in binary format of every "Dictionary-miss" pixels

  Previous pixel P_{i-1} = 103 (01100111)$_2$

  Current pixel P_i = 98 (01100010)$_2$

  LPM_i = 5

- We can
  - truncate the first "LPM_i" (5) bits of P_i
  - use one bit to indicate that P_i is truncated
  - output only the remaining "8-LPM_i" (3) bits

13

An Example of Utilizing “LPM”

dictionary pixels
$\{P_{i-1}-1, P_{i-1}, P_{i-1}+1\}$

P_0 P_1 P_2 P_3 P_4 P_5 P_6
Input pixels: 108, 99, 100, 101, 117, 84, 103

Current Pixel	Previous Pixel	LPM	Truncation Length (TLen)			
			1	2	3	4
P_0:108$_{10}$ **01101100**$_2$	**P_{-1}:0**$_{10}$ **00000000**$_2$	1	CW_{apbt} + $P_0[6:0]$	CW_{miss} + P_0	CW_{miss} + P_0	CW_{miss} + P_0
P_1:99$_{10}$ **01100011**$_2$	**P_0:108**$_{10}$ **01101100**$_2$	4	CW_{apbt} + $P_1[6:0]$	CW_{apbt} + $P_1[5:0]$	CW_{apbt} + $P_1[4:0]$	CW_{apbt} + $P_1[3:0]$
P_4:117$_{10}$ **01110101**$_2$	**P_3:101**$_{10}$ **01100101**$_2$	3	CW_{apbt} + $P_4[6:0]$	CW_{apbt} + $P_4[5:0]$	CW_{apbt} + $P_4[4:0]$	CW_{miss} + P_4
P_5:84$_{10}$ **01010100**$_2$	**P_4:117**$_{10}$ **01100110**$_2$	2	CW_{apbt} + $P_5[6:0]$	CW_{apbt} + $P_5[5:0]$	CW_{miss} + P_5	CW_{miss} + P_5
P_6:103$_0$ **01100111**$_2$	**P_5:84**$_{10}$ **01010100**$_2$	2	CW_{apbt} + $P_6[6:0]$	CW_{apbt} + $P_6[5:0]$	CW_{miss} + P_6	CW_{miss} + P_6
Total Codeword and Bits			5 CW+35 bits	5 CW+32 bits	5 CW+34 bits	5 CW+36 bits

14

Worst Case Performance Analysis

Algorithms	Type	Compression Ratio = Compressed/Original
Proposed	Line-based	1.003
LZWF07	Line-based	1.5
LiLY08	Line-based	1.77
YCKL09	Line-based	1.25
DiZh10	Line-based	2.73
LJMe12	Line-based	1.25
KiKK09	Block-based	1.063
KiKy10	Block-based	1.004
BaZG10	Block-based	1.002

15

Display Frames for Analysis

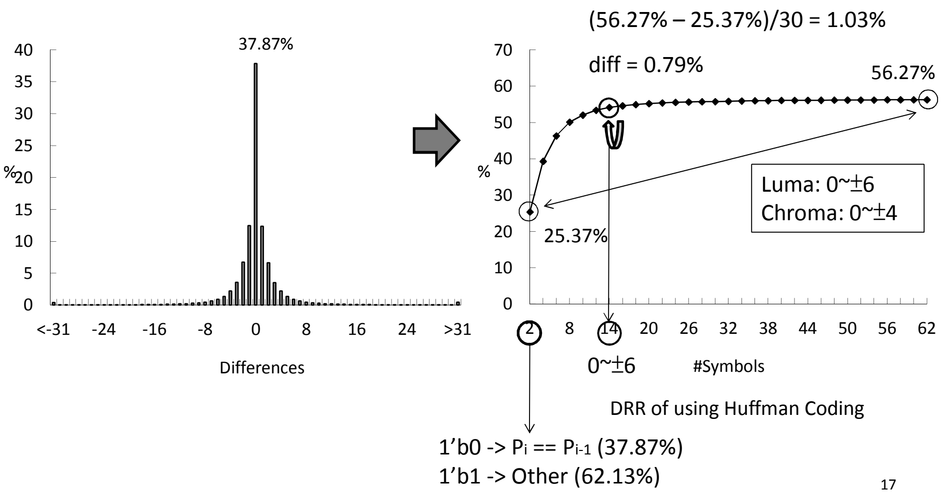
- Using 12 1080p videos that encoded and decoded by H.264/AVC reference software JM11.0

Parameters	Values
#Frames per Sequence	60
GOP	IPBPB
QP	4, 16, 28, 40
#Reference Frames	2
Entropy Coder	CABAC
Hadamard Transform	On
Search Range	128 ×128

16

Dictionary Size Selection

Process of luma component



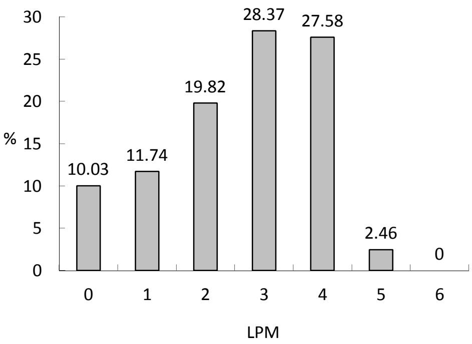
17

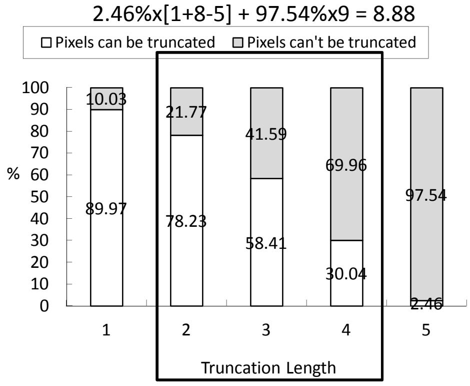
Truncation Length Candidates Selection

- Using a 1-bit code to indicate that if the current pixel is truncated
- Pixels can be truncated -> 1+ (8-TL) bits
- Pixels can not be truncated -> 1+8 = 9 bits

Luma: 2, 3, 4
Chroma: 2, 3, 4

2.46%x[1+8-5] + 97.54%x9 = 8.88



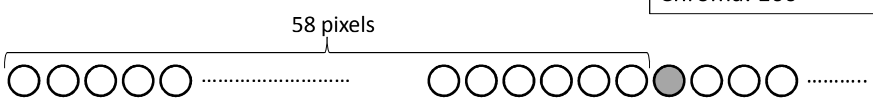


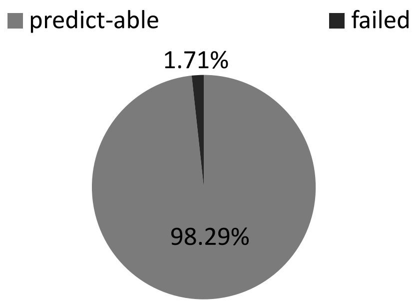
18

Pixel Group Size Selection

- To determine the size of a group
- Luma -> 1/0.0171 = 58.5 => 64

Luma: 64
Chroma: 160





19

Head Code Compression

- Head code: the first bit of the codeword

Current Pixel	Previous Pixel	Codeword
P_0 = 106	0	11111
P_1 = 105	106	100
P_2 = 106	105	101
P_3 = 106	106	0 ← Head Code

- Examples of compressing 4 continuous head codes

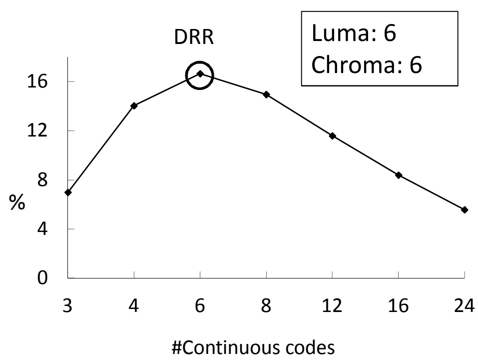
4 codes	Flag Bit + Data Bits
1111	0 + 1
1110	1 + 1110
0101	1 + 0101
0000	0 + 0

20

Best Run Length Selection

- There is a tradeoff between the number of continuous codes (run length) and the probabilities that they contain all 0 or 1

RL	$P_{0/1}$ (%)	Output Bits	P_{other} (%)	Output Bits
3	60.48	2	39.52	4
4	52.04	2	47.96	5
6	39.39	2	60.02	7
8	31.38	2	68.62	9
12	21.73	2	78.27	13
16	15.63	2	84.37	17
24	10.16	2	89.84	25



21

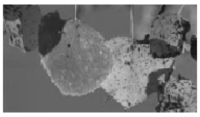
An Illustrative Compression Example

Input Pixels: 92, 94, 97, 98, 106, 105, 103, 103, 103, 96, 95, 95

	Yang's Algorithm [0~±1]		Proposed [0~±6, BTL = 4, BRL = 6]		
Input	Dictionary	CW + Output Data	Dictionary	LPM	CW + Output Data
92	[0~1]	11 + 01011100	[0~6]	1	11111 + 01011100
94	[91~93]	11 + 01011110	[86~98]	6	11001
97	[93~95]	11 + 01100001	[88~100]	2	11110
98	[96~98]	10	[91~103]	6	101
106	[97~99]	11 + 01101010	[92~104]	4	11011 + 1010
105	[105~107]	01	[100~112]	6	100
103	[104~106]	11 + 01100111	[99~111]	4	11000
103	[102~104]	00	[97~109]	8	0
103	[102~104]	00	[97~109]	8	0
96	[102~104]	11 + 01100000	[97~109]	5	11011 + 0000
95	[95~97]	01	[90~102]	2	100
95	[94~96]	00	[89~101]	8	0
	Total bits for encoding 12 pixels	72	111111_100110 => 01_1100110		2+56-3=55

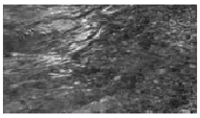
22

1080p Test Video Sequences


aspen


blue_sky


kimono

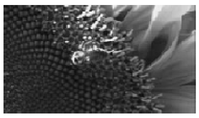

riverbed


rush_hour

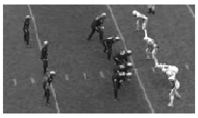

pedestrian_area


station2


snow_mnt


sunflower

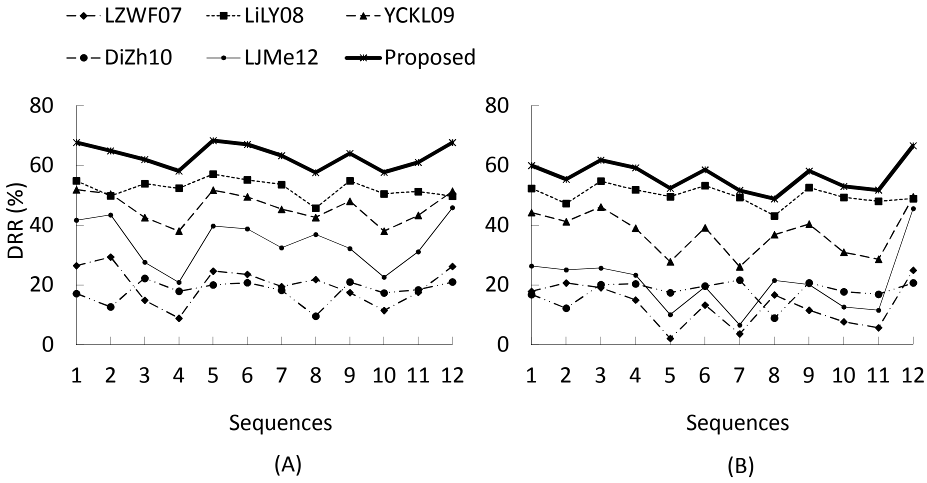

tractor


touchdown_pass

west_wind_easy

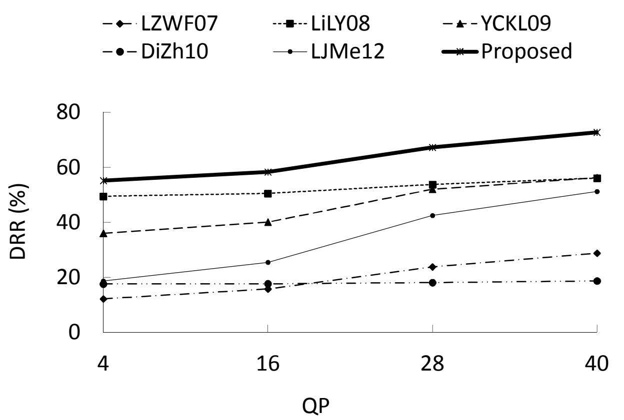
23

Comparison of DRR with Previous Line-based Algorithms for (A) Display & (B) Source Frames



24

DRR of All Line-based Algorithms for Various QP



25

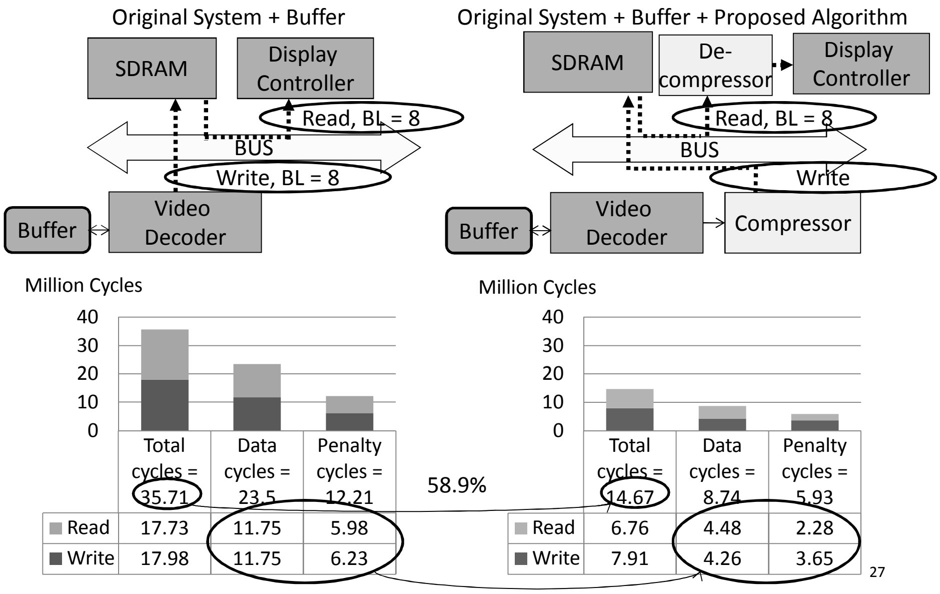
Comparison of Computational Complexity

	Computation Resources per Pixel		Memory Space (Bytes)	Equivalent Gates	DRR (%)
	Addition	Comparison			
YCKL09	2	3	1	312	44.39
LiLY08	18	0	1920	19224	51.96
Proposed	12	16	160	3200	61.97

Synthesized using TSMC 130nm Library
8-bit Adder => 108 gates
8-bit Comparator => 29 gates
1-byte Single-port Register File => 9 gates

26

Bus Traffic Reduction by Using Proposed Algorithm



	Total cycles = 35.71	Data cycles = 23.5	Penalty cycles = 12.21
Read	17.73	11.75	5.98
Write	17.98	11.75	6.23

	Total cycles = 14.67	Data cycles = 8.74	Penalty cycles = 5.93
Read	6.76	4.48	2.28
Write	7.91	4.26	3.65

27

Summary

- What – Line-based display frame compression algorithm
- Why – For reducing bus traffic and memory usage
- How – Dictionary coding + Huffman coding + Proposed APBT and HCC schemes
- Results – Reduces 59% of bus traffic of a video decoder
  – Improves at least 10% of DRR than prior arts

28

Thank you for your attention!!

THEDA.DESIGN.

29