Binary Counters

Integer Representations towards Efficient Counting in the Bit Probe Model

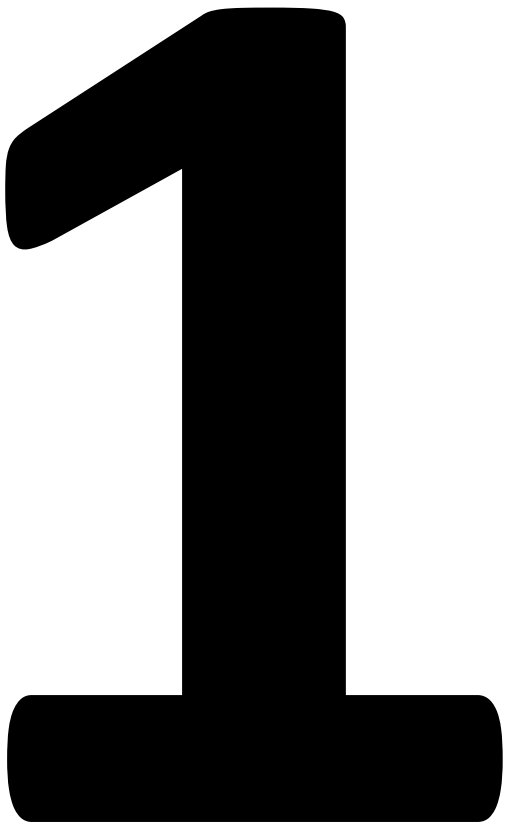
(presented at TAMC 2011)

Gerth Stølting Brodal (Aarhus University)

MADALGO
CENTER FOR MASSIVE DATA ALGORITHMICS

Mark Greve (Aarhus University)
Vineet Pandey (BITS Pilani, India)
S. Srinivasa Rao (Seoul, South Korea)

0



10

11

100

101

110

1 1 1

1000

1001

1010

1011

11100

11101

1110

11111

0000

- we are counting modulo $10000_2 = 16_{10}$

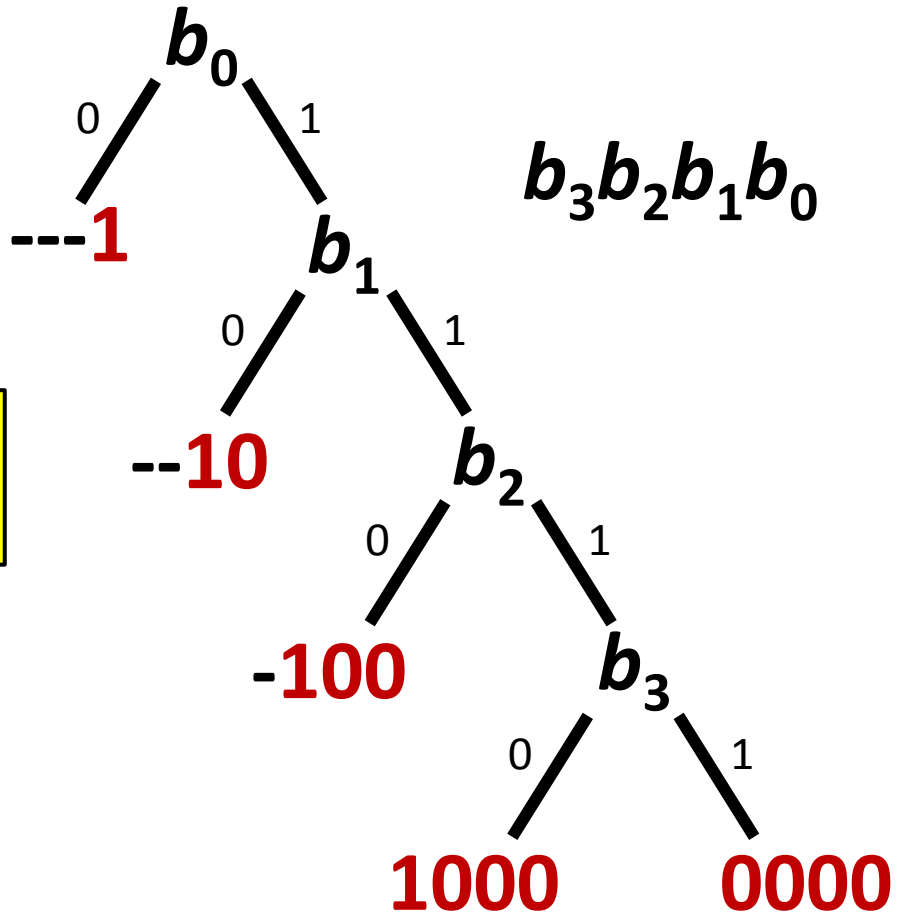
1011

$$1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 8+2+1 = 11_{10}$$

Decimal	Binary
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111
0	0000

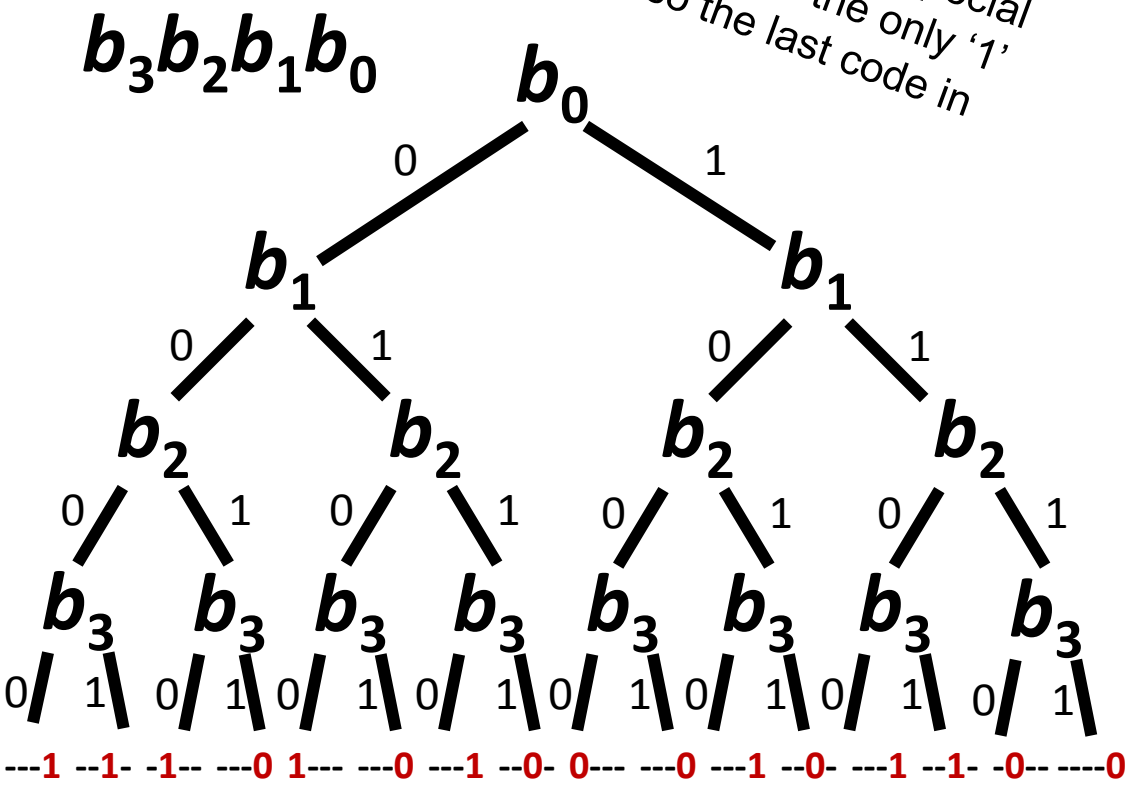
Algorithm

Reads 4 bits
Writes 4 bits



Decimal	Binary	Reflected Gray code
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000
0	0000	0000

"To get the next code, for a code with even parity, flip the rightmost bit. For a code with odd parity, find the rightmost '1' and flip the bit to its left. The only special case is when the last bit b_n is the only '1' in the code. This is also the last code in the sequence."



Always reads 4 bits

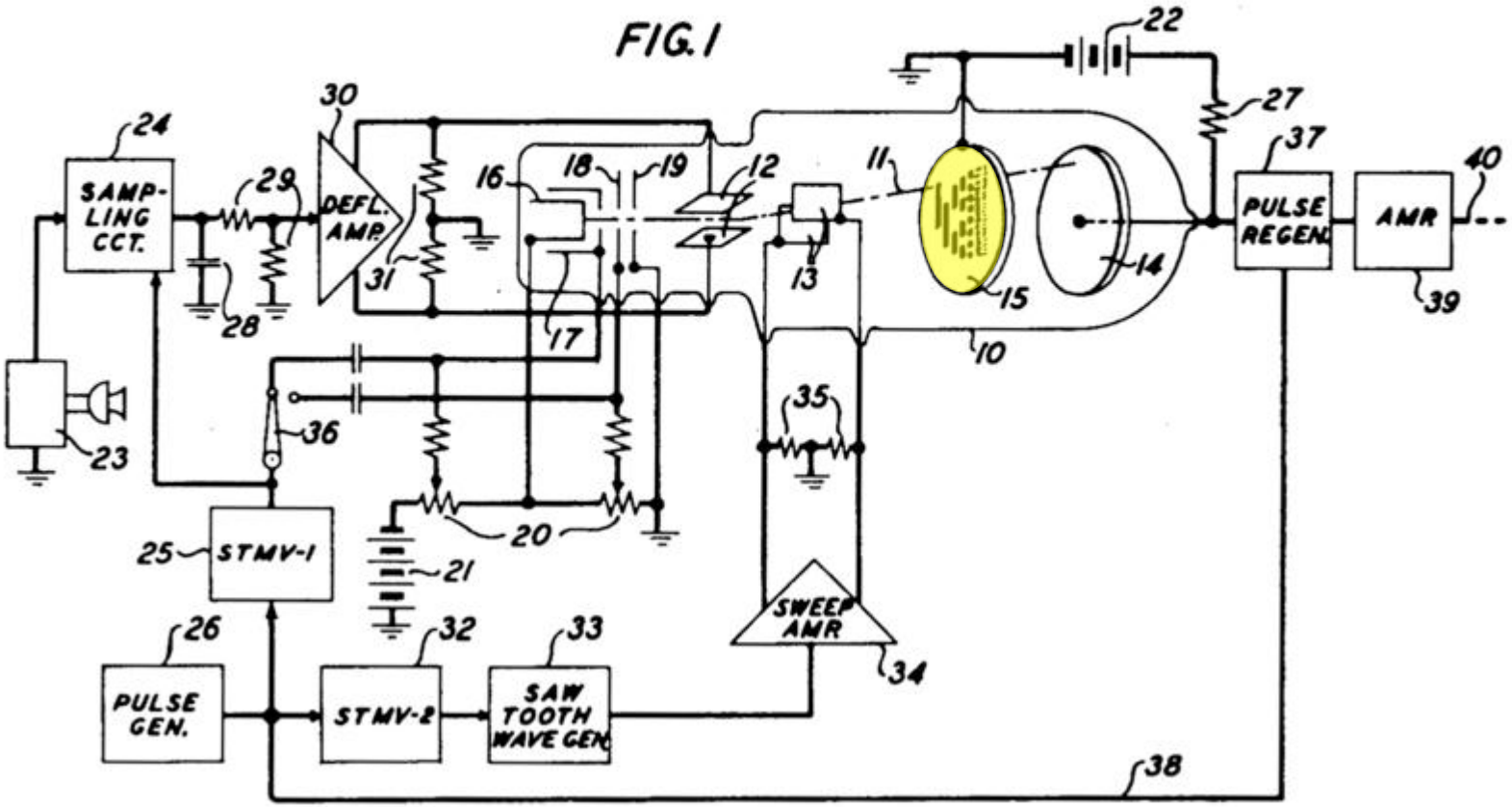
Always writes 1 bit

March 17, 1953 F. GRAY 2,632,058

PULSE CODE COMMUNICATION

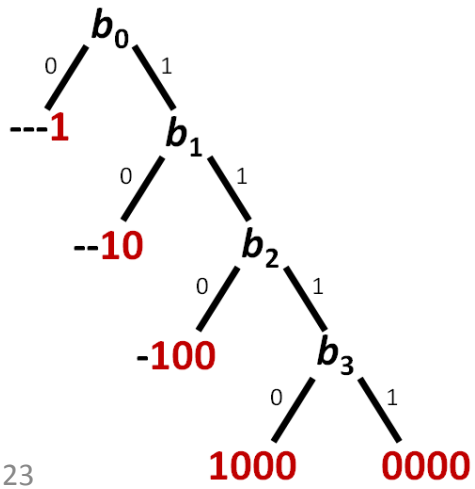
Filed Nov. 13, 1947 4 Sheets-Sheet 1

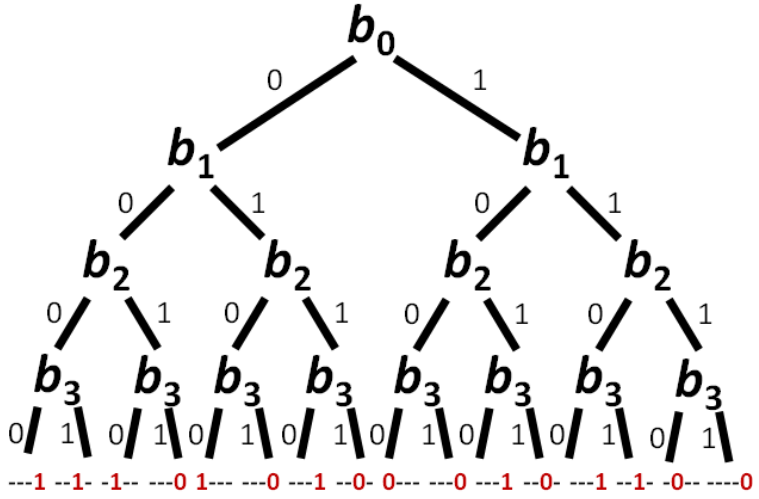
FIG. 1



Question

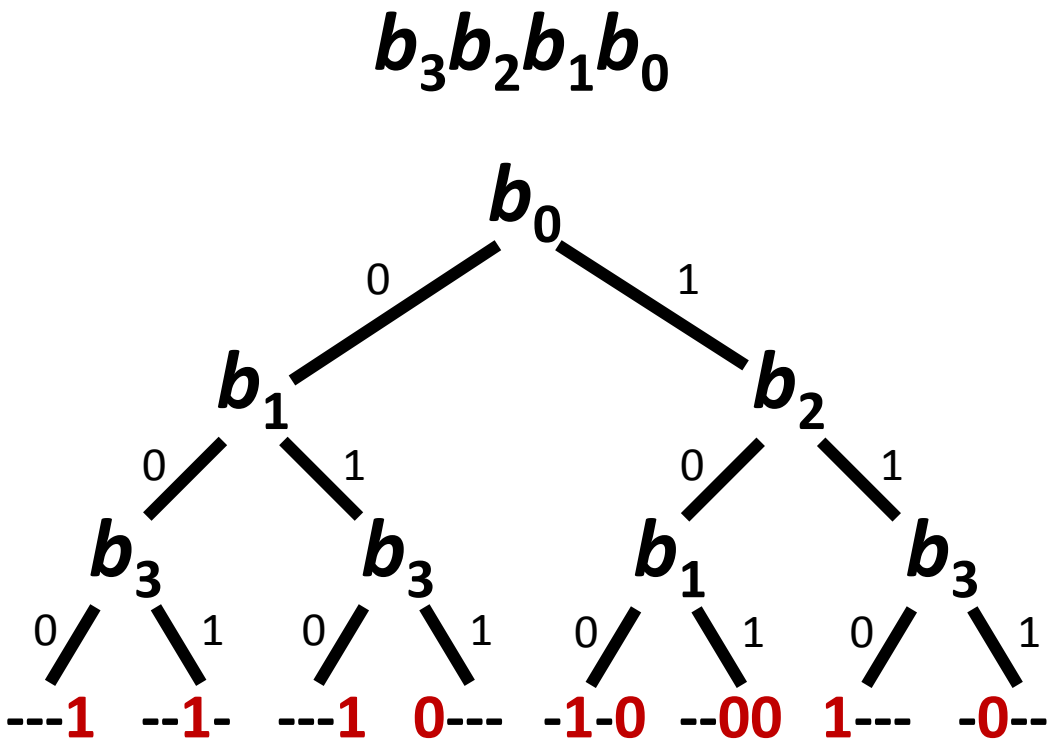
Does there exist a counter where one never needs to read all bits to increment the counter ?





Decimal	
0	0000
1	0001
2	0100
3	0101
4	1101
5	1001
6	1100
7	1110
8	0110
9	0111
10	1111
11	1011
12	1000
13	1010
14	0010
15	0011
0	0000

[B., Greve , Pandey, Rao 2011]

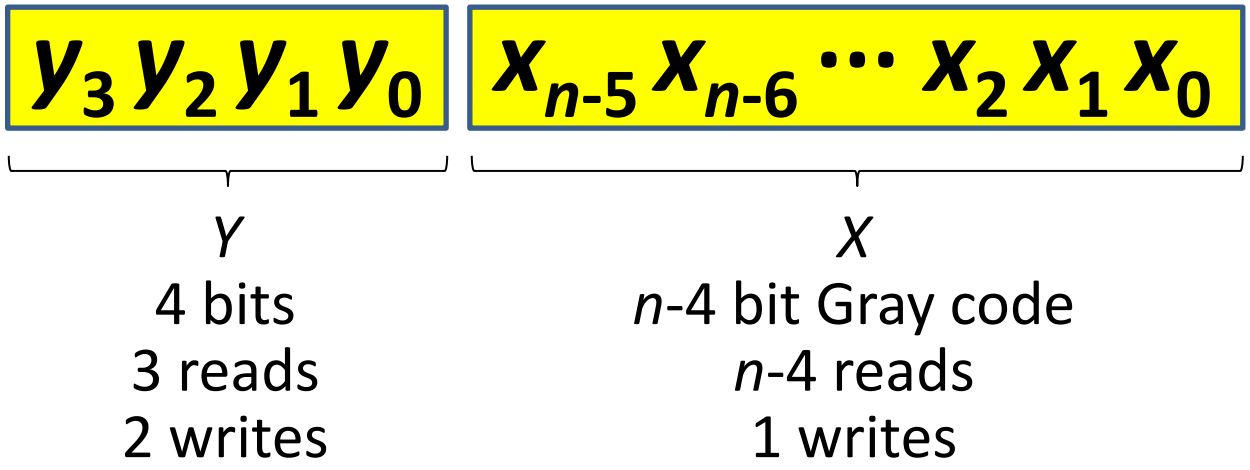


Always reads 3 bits

Always writes ≤ 2 bits

[B., Greve , Pandey, Rao 2011]

Generalization to *n* bit counters



metode Increment(*YX*)

inc(*X*)

if (*X* == 0) inc(*Y*)

test needs to read all bits of *X*

Always reads *n*-1 bits

Always writes ≤ 3 bits

Theorem

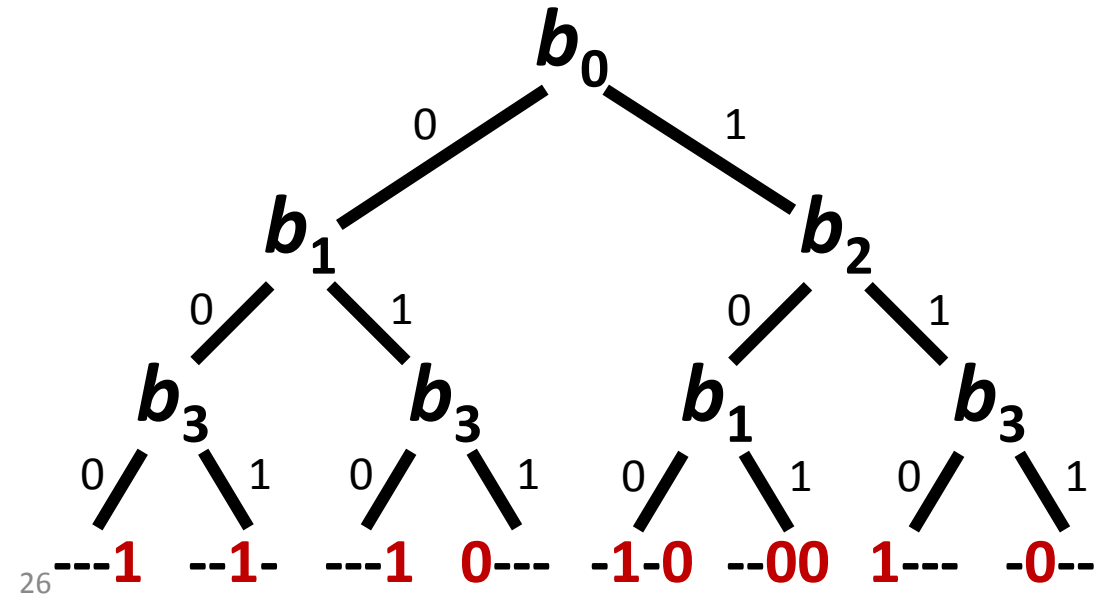
4-bit counter 3 reads and 2 writes

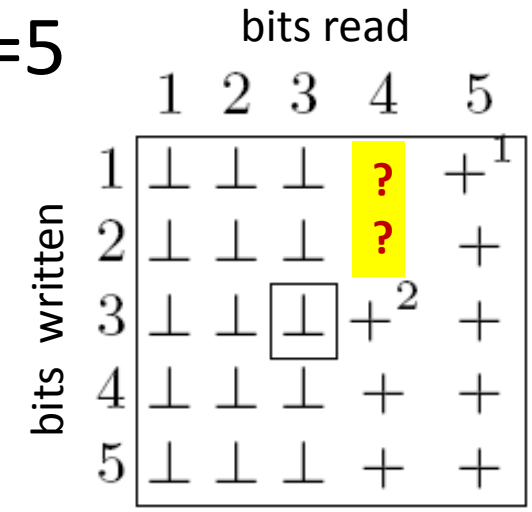
n-bit counter *n*-1 reads and 3 writes

Open problems

n-1 reads and 2 writes, *n*>4?

« *n* reads and writes ? [number of reads at least $\log_2 n$]





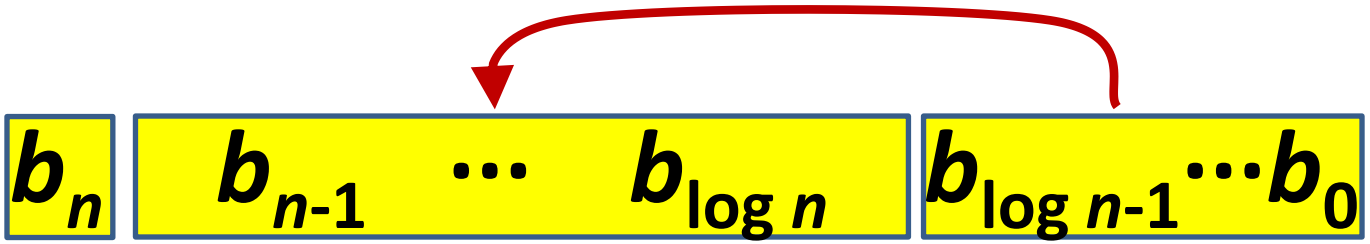
Redundant Counters

Represent ***L*** different values
using ***d*** > log *L* bits

Efficiency ***E*** = $L / 2^d$

[B., Greve , Pandey, Rao 2011]

Redundant counter with $E = ½$ ($L = 2^n$)



carry
1 bit
1 read
1 write

X_H
n-log n bits
1 read
1 write

standard binary counter with delayed increment

X_L
log n bits Gray code
log n reads
1 write

Idea: Each increment of X_L performs one step of the delayed increment of X_H

n+1 bits	2^n values	log n + 2 reads	3 writes

Redundant counter with $E = 1/2$

increment

Value	carry	X_H					X_L		
0	1	0	1	0	1	1	0	0	0
1	1	0	1	0	1	0	0	0	1
2	1	0	1	0	0	0	0	1	1
3	0	0	1	1	0	0	0	1	0
4	0	0	1	1	0	0	1	1	0
5	0	0	1	1	0	0	1	1	1
6	0	0	1	1	0	0	1	0	1
7	0	0	1	1	0	0	1	0	0
8	1	0	1	1	0	0	0	0	0
10	0	0	1	1	0	1	0	0	1

$$\text{Value} = \text{Val}(X_L) + 2^{|X_L|} \cdot (\text{Val}(X_H) + \text{carry} \cdot 2^{\text{Val}(X_L)})$$

n+1 bits	2^n values	log n + 2 reads	3 writes

Redundant counter with $E = 1/2$

increment

delayed reset

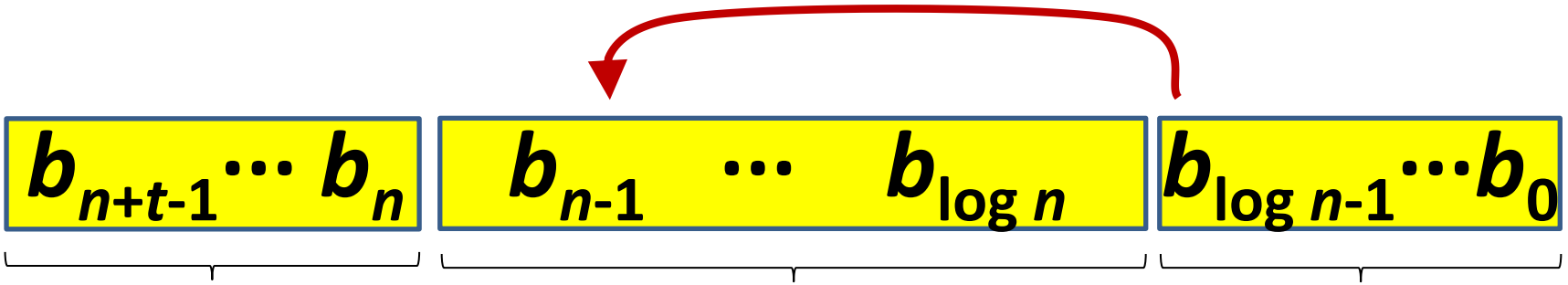
delayed reset

Value	carry	X_H					X_L		
0	1	0	1	0	1	1	0	0	0
1	1	0	1	0	1	0	0	0	1
2	1	0	1	0	0	0	0	1	1
3	1	0	1	1	0	0	0	1	0
4	0	0	1	1	0	0	1	1	0
5	0	0	1	1	0	0	1	1	1
6	0	0	1	1	0	0	1	0	1
7	0	0	1	1	0	0	1	0	0
8	1	0	1	1	0	0	0	0	0
10	1	0	1	1	0	1	0	0	1

n+1 bits **2^n values** **log n + 3 reads** **2 writes**

[B., Greve , Pandey, Rao 2011]

Redundant counter with $E = 1\text{-}1/2^t$



"carry"
t bits
t read
1 write

X_H
n-log n bits
1 read
1 write

X_L
log *n* bit Gray code
log *n* reads
1 write

delayed standard binary counter

"Carry" : part of counter = 0.. 2^t-3, set = 2^t-2, clear 2^t-1

***n+t* bits**	**$(2^t\text{-}1)\cdot 2^n$ values**	**log *n*+*t*+1 reads**	**3 writes**

Redundant Counters

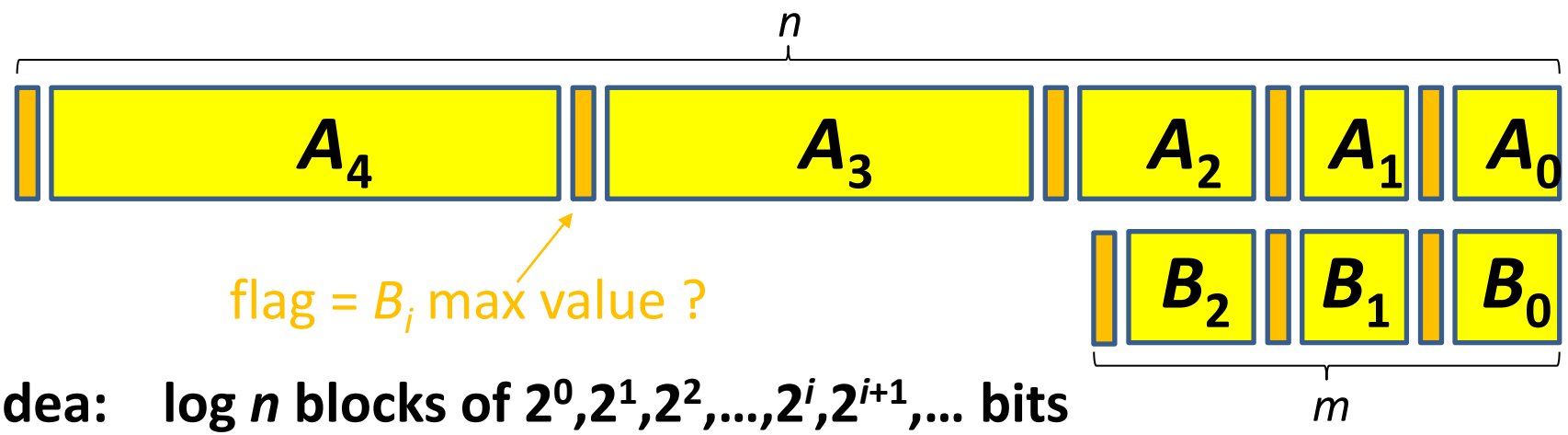
Efficiency	Space	Reads	Writes
1/2	$n + 1$	$\log n + 2$	3
		$\log n + 3$	2
$1 - 1/2^t$	$n + t$	$\log n + t + 1$	3
		$\log n + t + 2$	2

Open problem 1 write and « n reads ?

Addition of Counters

Numbers in the range $0..2^n-1$ and $0..2^m-1$ ($m \le n$)

Space	Reads	Writes
$n + O(\log n)$	$\Theta(m + \log n)$	$\Theta(m)$
$n + O(\log\log n)$	$\Theta(m + \log n \cdot \log\log n)$	
$n + O(1)$	$\Theta(m + \log^2 n)$	



Idea: **log *n* blocks of $2^0,2^1,2^2,\ldots,2^i,2^{i+1},\ldots$ bits**

Space (d)	Space efficiency	Bits read (B_R)		Bits written (B_W)	Inc. & Dec.	Ref.
		Average-case	Worst-case	Worst-case		
n	1	$2 - 2^{1-n}$	n	n	Y	Binary
		n		1	Y	[3]
		$6 \log n$		1	Y	[1]
		$O(\log^{(2c-1)} n)$		c	N	[1]
$n+1$	1/2	$O(1)$	$\log n + 4$	4	Y	[4]
$n + O(t \log n)$	$1 - O(n^{-t})$	$O(\log^{(2c)} n)$	$O(t \log n)$	$2c+1$	N	[1]

Space (d)	Space efficiency	Average-case		Worst-case		Inc. & Dec.	Ref.
		B_R	B_W	B_R	B_W		
4	1	3	1.25	3	2	Y	Th. 1
n		$6 \log(n-4) + O(2^{-n})$	$1 + O(2^{-n})$	$n-1$	3		Th. 2
$n+1$	1/2	$O(\log \log n)$	$1 + O(n^{-1})$	$\log n + 2$	3	N	Th. 3
				$\log n + 3$	2		Th. 4
		$O(\log n)$		$\log n + 3$	3	Y	Th. 6
				$\log n + 4$	2		
n	$1 - \frac{1}{2^{t-1}}$	$O(\log \log n)$	$1 + O(n^{-1})$	$\log n + t + 1$	3	N	Th. 5
				$\log n + t + 2$	2		
		$O(\log n)$		$\log n + t + 2$	3	Y	Cor. 1
				$\log n + t + 3$	2		

[1] Bose, Carmi, Jansens, Maheshwai, Morin, Smid, SWAT 2010
[3] Gray, Patent 1953 [4] Rahman, Munro, Algorithmica 2010


Thank You
Gerth Stølting Brodal
Aarhus University
MADALGO
CENTER FOR MASSIVE DATA ALGORITHMICS