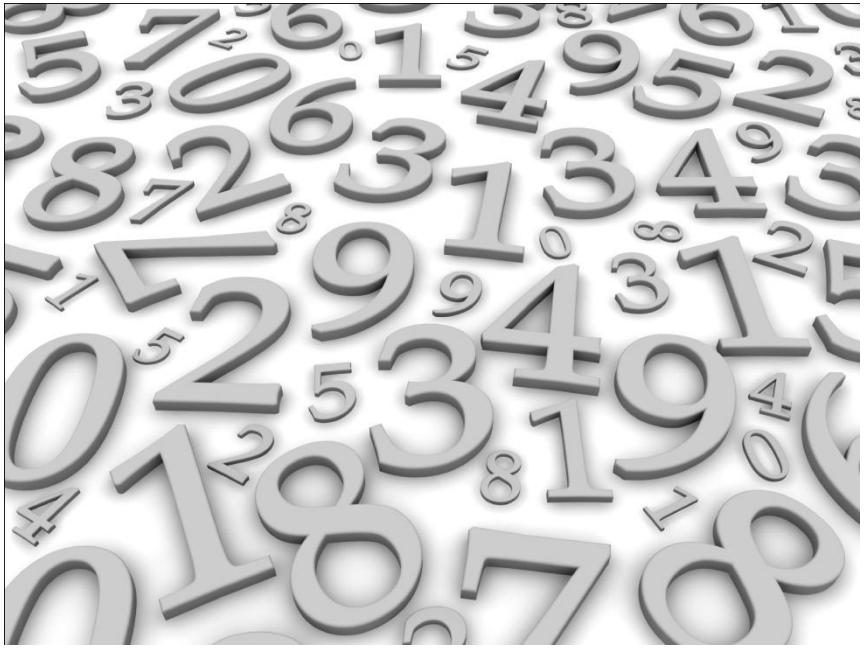
21st IEEE Symposium on Computer Arithmetic
ARITH 21
Austin, Texas, USA. April 7-10, 2013

A Fast Circuit Topology for Finding the Maximum of *n* k-bit Numbers

Bilgiday Yüce
Günhan Dündar
Bosphorus (Boğaziçi) University
Istanbul, Turkey

H. Fatih Uğurdağ
Özyeğin University
Istanbul, Turkey

Sezer Gören
Yeditepe University
Istanbul, Turkey



fatih.ugurdag@ozyegin.edu.tr

Bosphorus University

ÖZYEĞİN UNIVERSITY

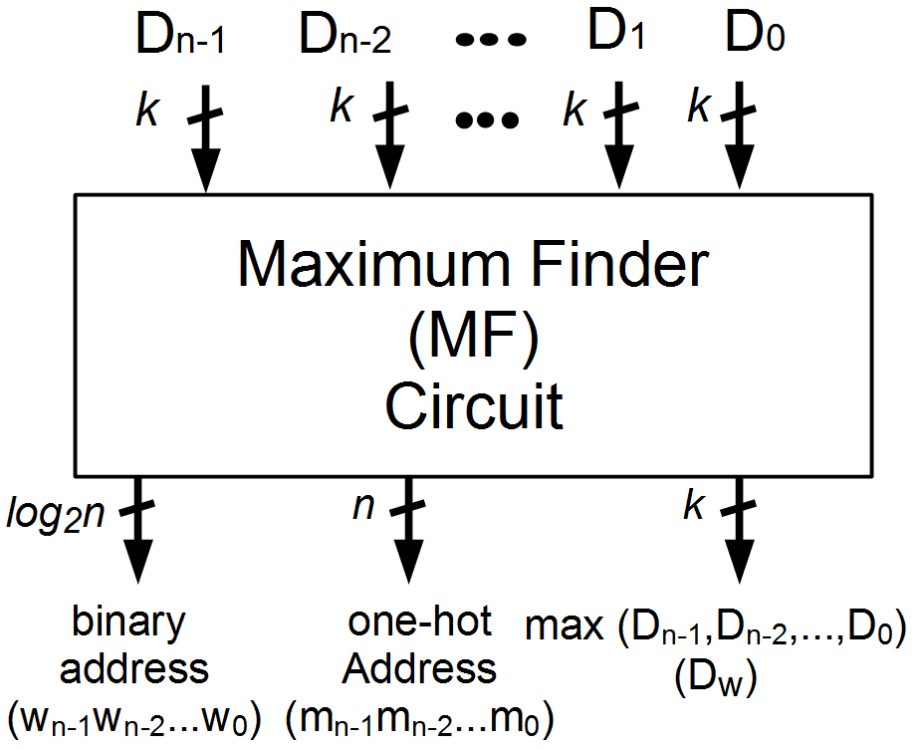
Yeditepe University

Outline

- Introduction
  - Problem Definition
  - In a Nutshell
  - Motivation
- Related Work
  - Traditional Binary Tree Topology (TBT)
  - Array Topology (AT)
  - Parallel Binary Tree Topology (PBT)
  - Multi-Level Topology (MLT)
  - Leading-Zero Counting Topology (LZC)
- Proposed Solutions:
  - Array-Based maximum finder (AB)
  - Quad Tree (QT)
- Synthesis Results
- Conclusions and Future Work

Introduction

Problem Definition



- Provided the circuit is:

  Combinational (i.e. Parallel)

- Minimize:

  Latency (i.e. Critical Path)

  with (if possible) good:

  - Area
  - Area-Timing Product (ATP)

Introduction
In a Nutshell

- We offer a timing complexity (T) of:

  $T = O(\log_2 k + \log_2 n)$

- whereas our competitions are:

  $T = O(\log_2 k \log_2 n)$

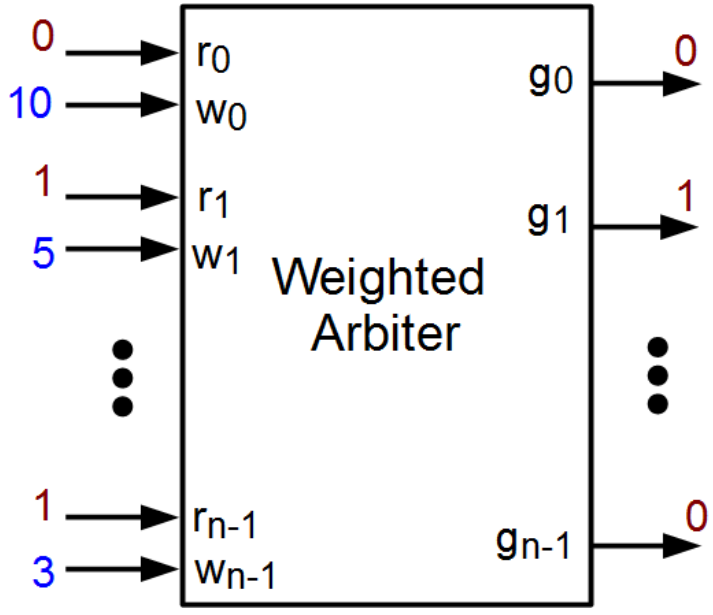
  $T = O(k + \log_2 n)$

Introduction

Motivation

Various applications require the value and/or index of the maximum element in a set:

- Weighted arbiters,
- Motion estimation algorithms,
- Sorting networks,
- Artificial neural networks, etc.



Introduction

Motivation – cont'd

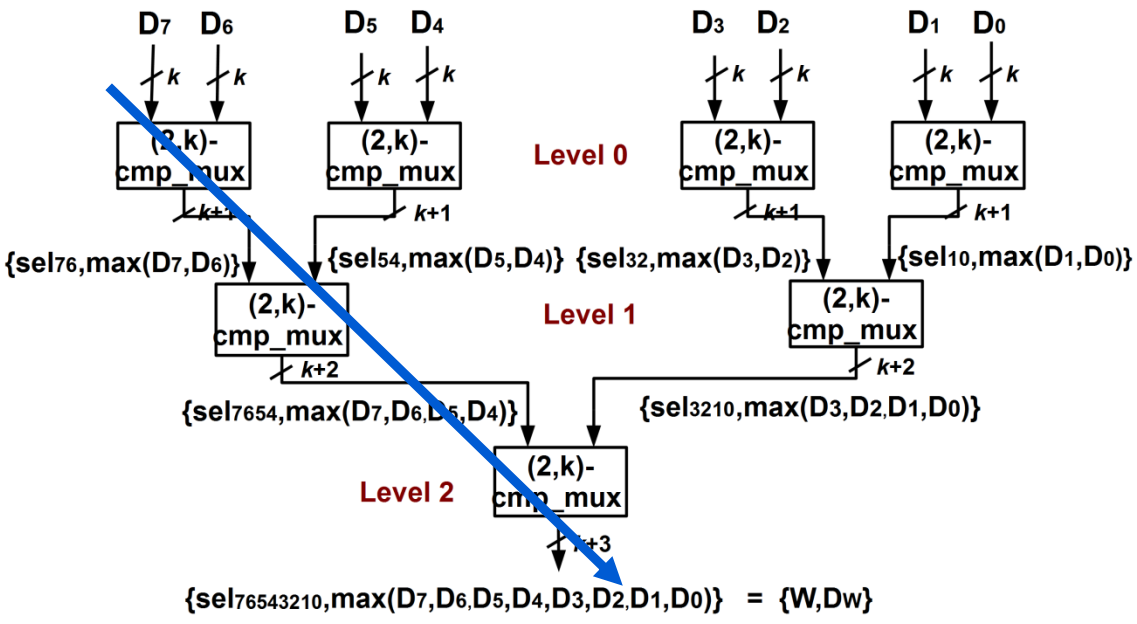
- Depending on the application, we may need
  - Only value of the maximum element,
  - Only the address (position or index) of the maximum element, or
  - Both the value and the address of the maximum element.
- Hence, fast computation of both value and index of the maximum is equally important.

21st IEEE Symposium on Computer Arithmetic
ARITH 21
Austin, Texas, USA. April 7-10, 2013

Related Work

Traditional Binary Tree Topology (TBT)

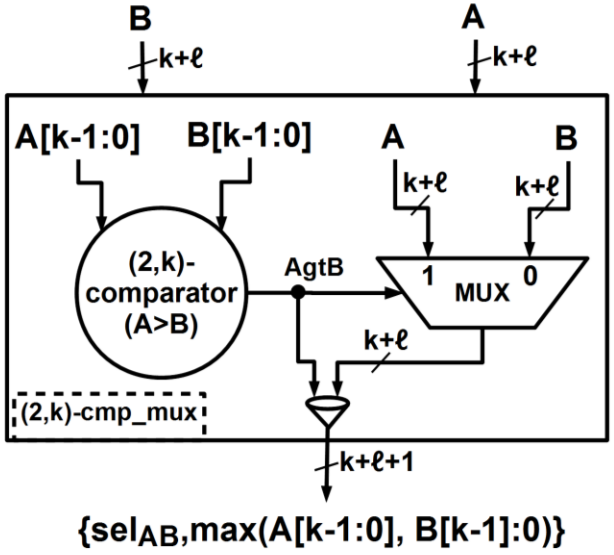
- Binary tree configuration of comparator-mux (cmp_mux) blocks.
- Signal propagation in each cmp_mux block : from LSB to MSB.
- $T = O(\log_2 k \times \log_2 n)$
  $A = O(k \times n)$
- Binary address and the value of the maximum.

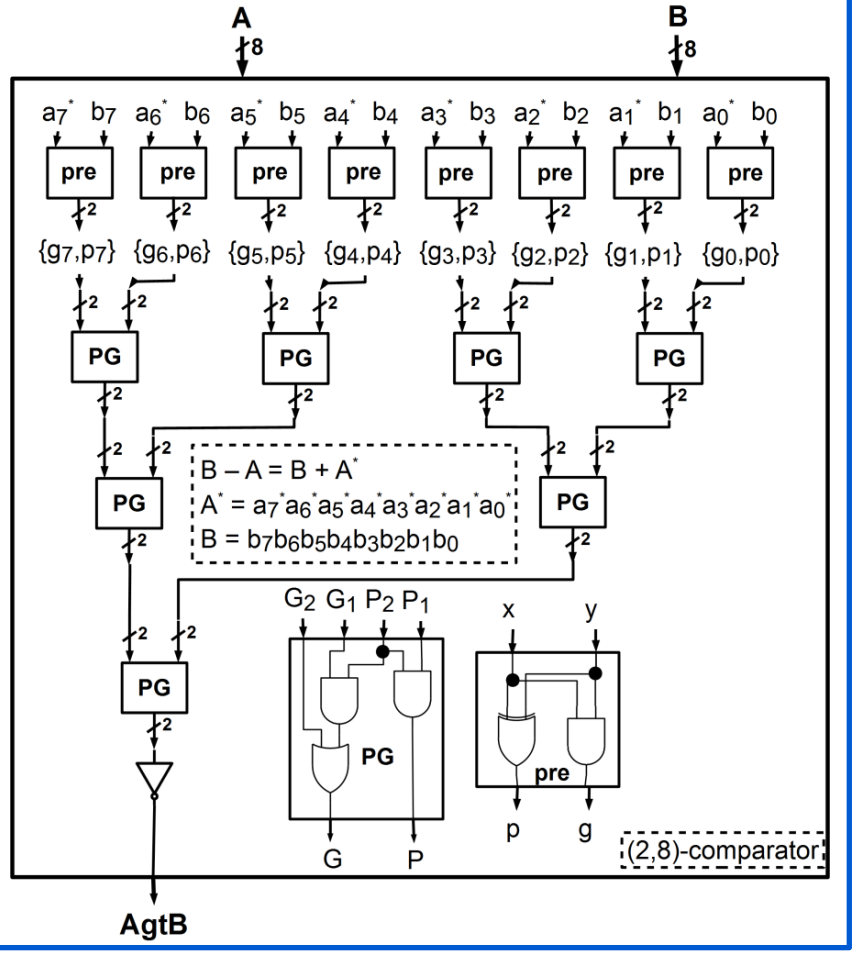


Related Work

Traditional Binary Tree Topology (TBT) – cont'd

- Comparator-mux (cmp_mux) block.
- Comparator (cmp) block.



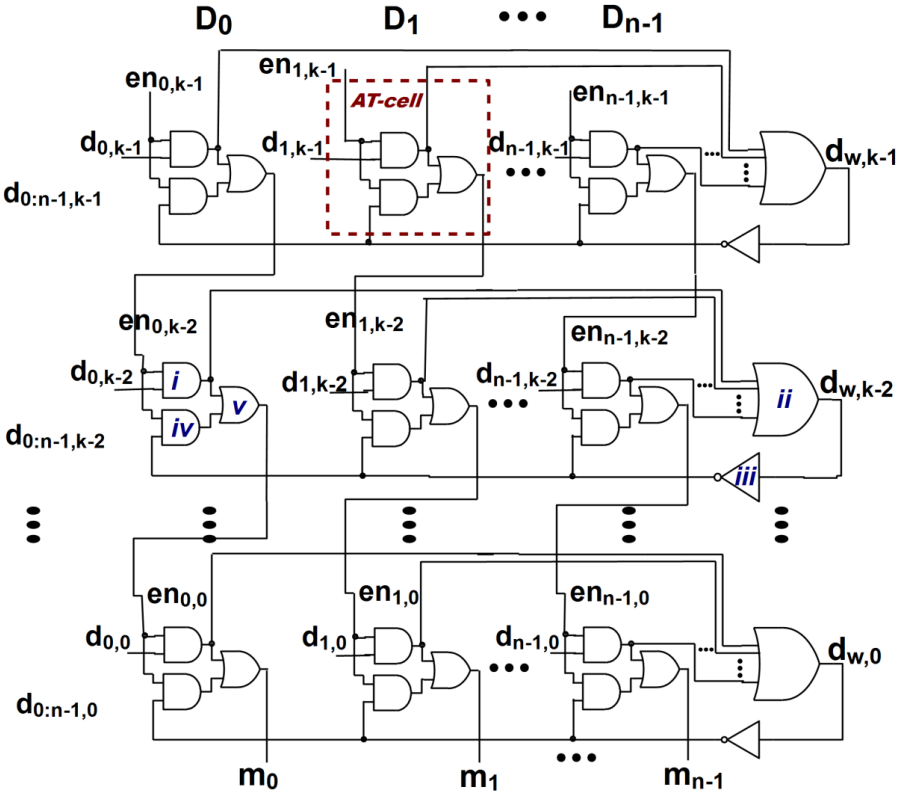


21st IEEE Symposium on Computer Arithmetic
ARITH 21
Austin, Texas, USA. April 7-10, 2013

Related Work

Array Topology (AT)

- A filter:
  - Compares each bit-slice from MSB to LSB
    and
  - Narrows down the candidates for the maximum after each bit-slice.
- Enable (en) signals ripple down from MSB to LSB.
- Critical Path: $k \times (i\text{-}ii\text{-}iii\text{-}iv\text{-}v)$
- $T = O(k \times \log_2 n)$
  $A = O(k \times n)$
- One-hot address and the value of the maximum.

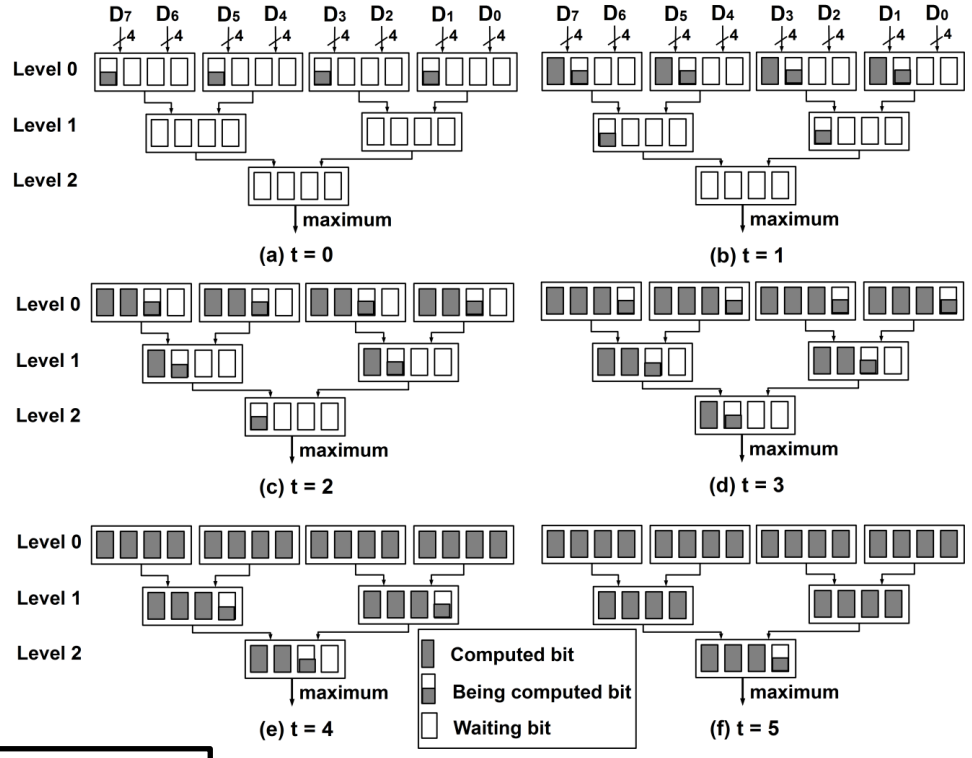


M. Vai and M.M. Moy (1993)

Related Work

Parallel Binary Tree Topology (PBT)

- Signal propagation in each cmp_mux block: from MSB to LSB
- Similar to addition problem
- Three adder topologies for cmp_mux blocks:
  - Ripple Carry (RC)
  - Carry Select (CS)
  - Carry Look-ahead (CL)



Method	Time Complexity	Area Complexity
RC	$O(k + \log_2 n)$	$O(k \times n)$
CS	$O(\sqrt{k}) + O(\lceil \sqrt{k} - \log_2 n \rceil \times \log_2 n)$	$O(k \times n)$
CL	$O(\log_2 k \times \log_2 n)$	$O(k \times n)$

K.G.I. Harteros (2002)

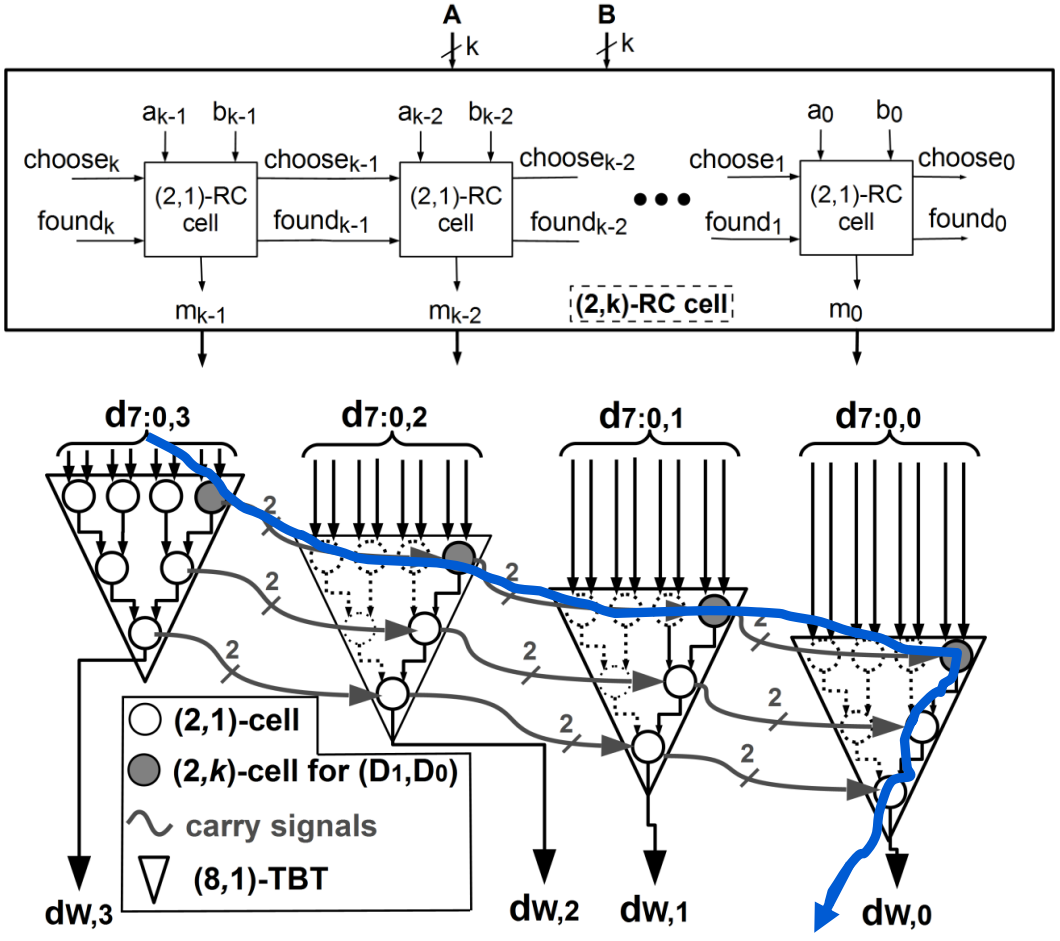
D.C. Hendry (2004)

21st IEEE Symposium on Computer Arithmetic
ARITH 21
Austin, Texas, USA. April 7-10, 2013

Related Work

Parallel Binary Tree Topology (PBT) – cont'd

- Example: RC-PBT
- 2-bit carry signal.
- Cascaded instances of TBT.
- Only the value of the maximum.
- EXTRA multiplexer tree for the binary address of the maximum.



fatih.ugurdag@ozyegin.edu.tr

Bosphorus University

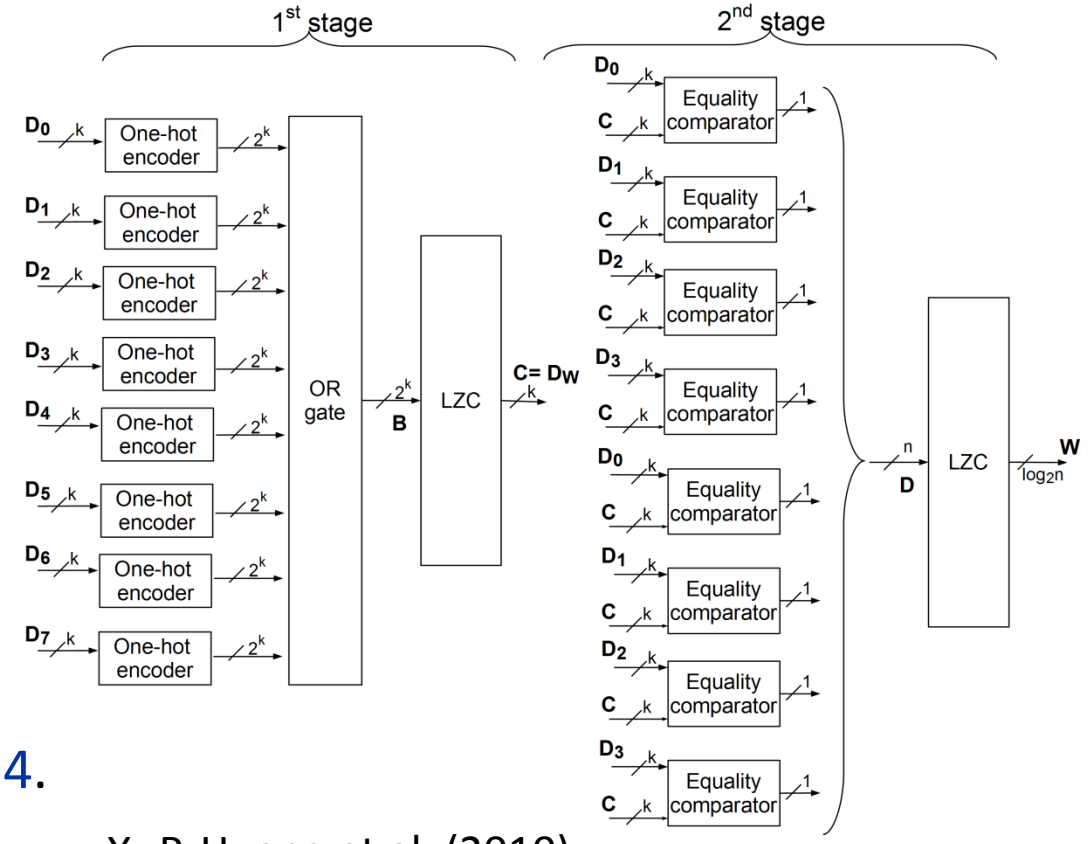
ÖZYEĞİN UNIVERSITY

Yeditepe University

Related Work

Leading-Zero Counting Topology (LZC)

- 2 stages:
  - Value computation,
  - Address computation.
- $T = O(k + \log_2 n + \log_2 k)$
  $A = O(2^k \times (k + n))$
- Binary address and the value of the maximum.
- DOES NOT scale for $k > 4$.

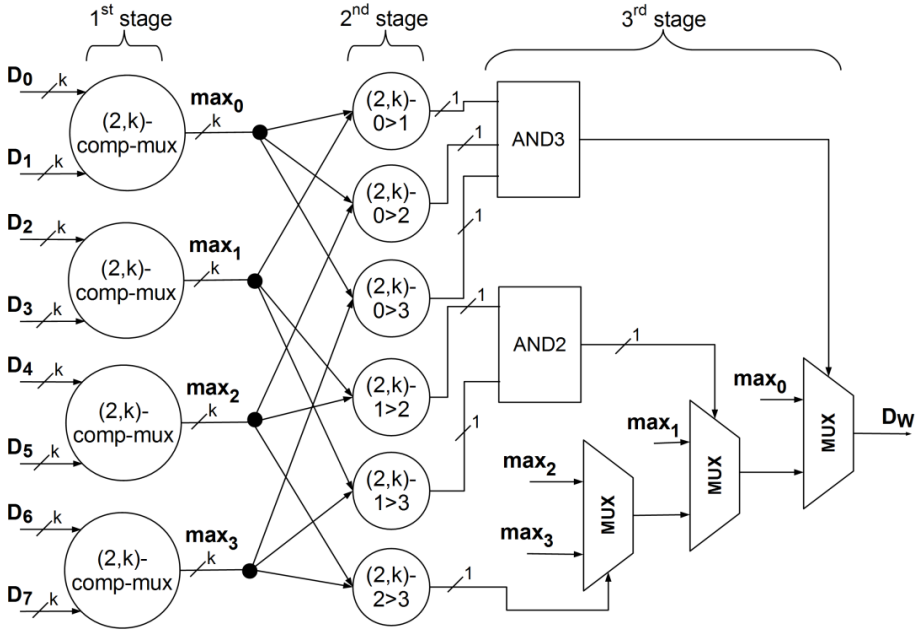


X.-P. Huang et al. (2010)

Related Work

Multi-Level Topology (MLT)

- 3 stages:
  - Pair-wise comparison,
  - Parallel comparison,
  - Output selection.
- Only the value of the maximum.
- DOES NOT utilize concurrency between comparison and selection.
- Needs an EXTRA stage for the binary/one-hot address of the maximum.
- $T = O(n + \log_2 n + \log_2 k)$
  $A = O(2^k \times k)$

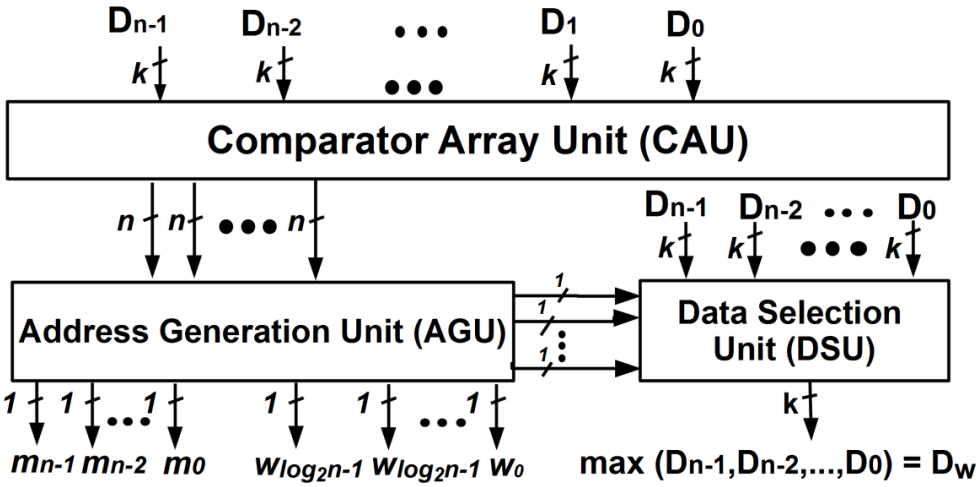


X.-P. Huang et al. (2010)

Proposed MF Circuit Topology

Array-Based maximum finder (AB)

- Main idea:
  - First, compare every pair of inputs in parallel.
  - Then, compute concurrently address and value of the maximum.
- $T = O(\log_2 k + \log_2 n)$
  $A = O(k \times n^2)$
- 3 blocks:
  - Comparator Array Unit (CAU)
  - Address Generation Unit (AGU)
  - Data Selection Unit (DSU)

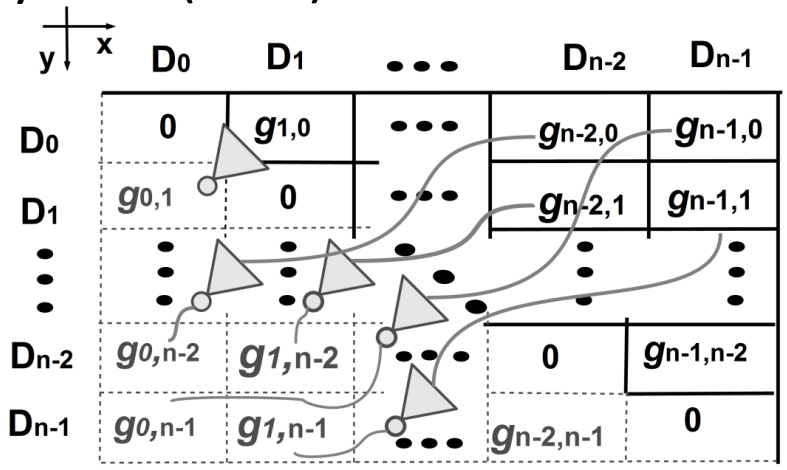


Proposed MF Circuit Topology

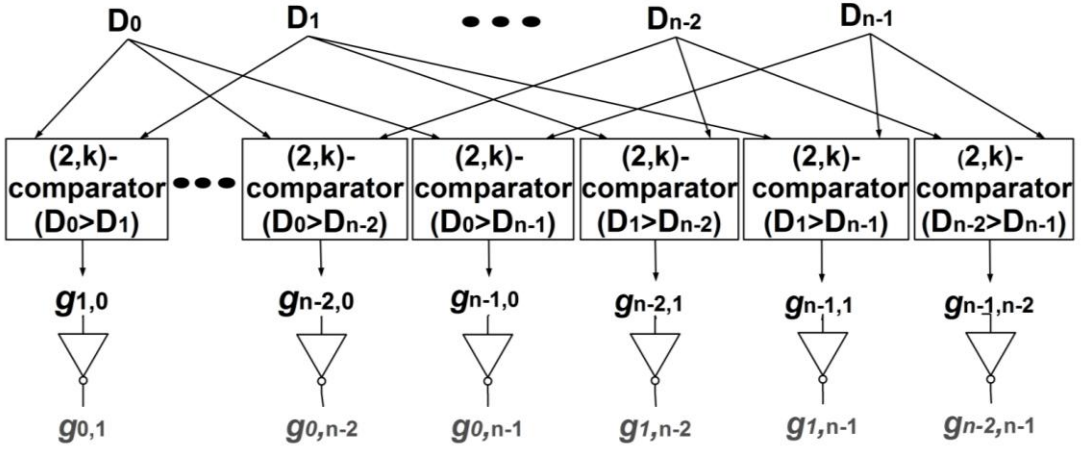
Comparator Array Unit (CAU)

- Produces an *n*×*n* array of 1-bit comparison results

$$
\begin{aligned}
m_0 &= (D_0 \geq D_0) \cdot (D_0 > D_1) \cdot \ldots \cdot (D_0 > D_{k-1}) \\
m_1 &= (D_1 \geq D_0) \cdot (D_1 \geq D_1) \cdot \ldots \cdot (D_1 > D_{k-1}) \qquad (1) \\
\vdots &= \quad \vdots \quad \cdot \quad \vdots \quad \cdot \ddots \cdot \quad \vdots \\
m_{k-1} &= (D_{k-1} \geq D_0) \cdot (D_{k-1} \geq D_1) \cdot \ldots \cdot (D_{k-1} \geq D_{k-1})
\end{aligned}
$$



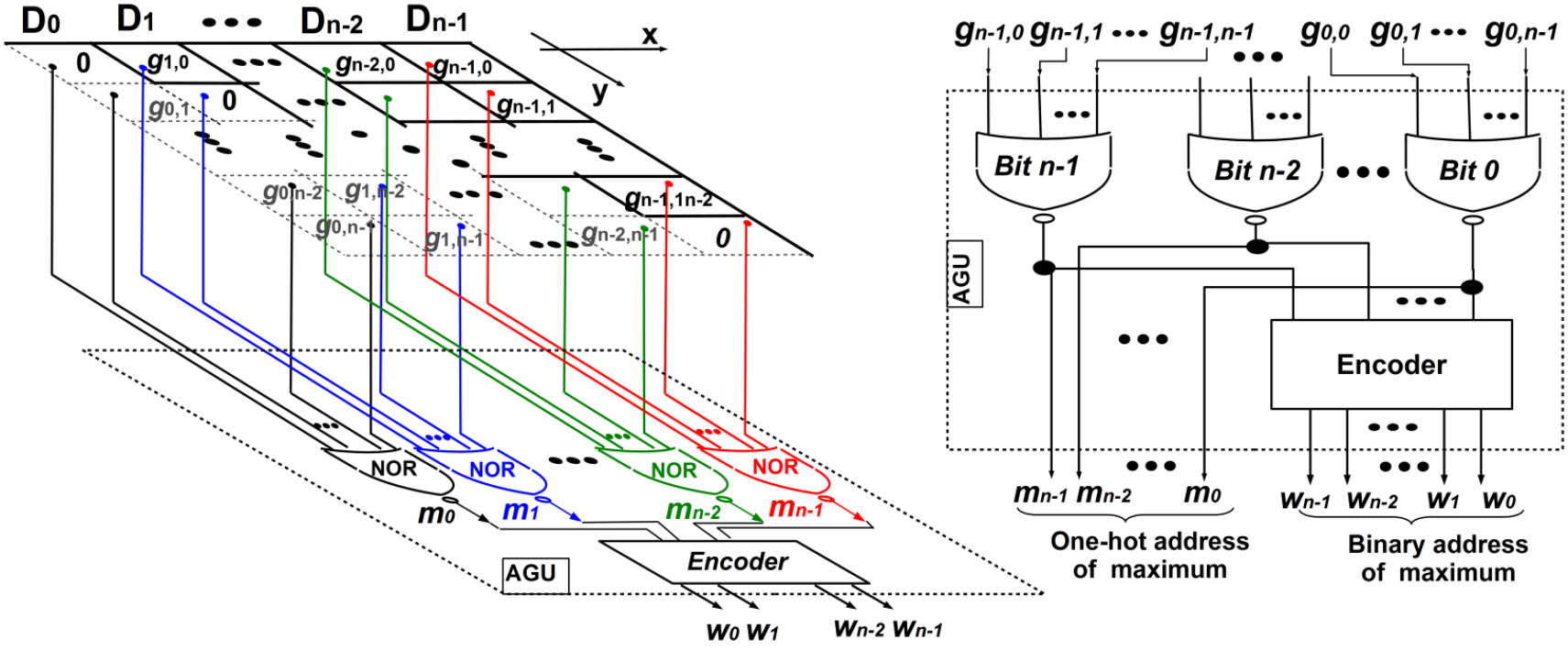
$$
\begin{aligned}
m_x &= \left(\prod_{y=0}^{x}(D_x \geq D_y)\right) \cdot \left(\prod_{y=x+1}^{n-1}(D_x > D_y)\right) \\
&= \overline{\left(\sum_{y=0}^{x}(D_x < D_y)\right) + \left(\sum_{y=x+1}^{n-1}(D_x \leq D_y)\right)} \\
&= \overline{\left(\sum_{y=0}^{x} g_{xy}\right) + \left(\sum_{y=x+1}^{n-1} \overline{g_{yx}}\right)}
\end{aligned}
$$



Proposed MF Circuit Topology

Address Generation Unit (AGU)

- Computes one-hot and/or binary address of the maximum



Proposed MF Circuit Topology

Address Generation Unit (AGU) – cont'd

- Example: n=4 k=5.

Upper triangular part

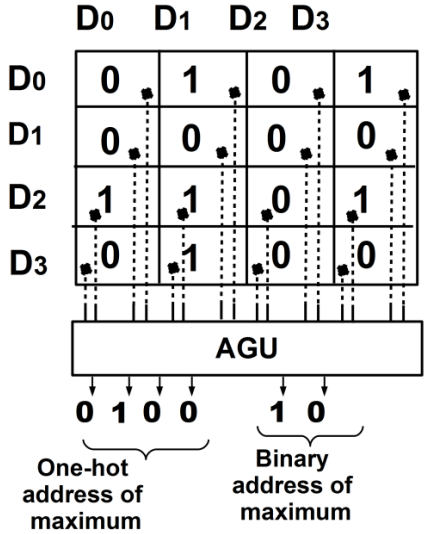
$$g_{1,0} = (D_0 > D_1) = 1$$
$$g_{2,0} = (D_0 > D_2) = 0$$
$$g_{3,0} = (D_0 > D_3) = 1$$
$$g_{2,1} = (D_1 > D_2) = 0$$
$$g_{3,1} = (D_1 > D_3) = 0$$
$$g_{3,2} = (D_2 > D_3) = 1$$

② ①

Lower triangular part

$$g_{0,1} = \overline{g_{1,0}} = 0$$
$$g_{0,2} = \overline{g_{2,0}} = 1$$
$$g_{0,3} = \overline{g_{3,0}} = 0$$
$$g_{1,2} = \overline{g_{2,1}} = 1$$
$$g_{1,3} = \overline{g_{3,1}} = 1$$
$$g_{2,3} = \overline{g_{3,2}} = 0$$

(2) ③



$$m_0 = \overline{(D_0 \le D_1) + (D_0 \le D_2) + (D_0 \le D_3)} = \overline{g_{0,1} + g_{0,2} + g_{0,3}}$$
$$m_1 = \overline{(D_1 < D_0) + (D_1 \le D_2) + (D_1 \le D_3)} = \overline{g_{1,0} + g_{1,2} + g_{1,3}}$$
$$m_2 = \overline{(D_2 < D_0) + (D_2 < D_1) + (D_2 \le D_3)} = \overline{g_{2,0} + g_{2,1} + g_{2,3}}$$
$$m_3 = \overline{(D_3 < D_0) + (D_3 < D_1) + (D_3 < D_2)} = \overline{g_{3,0} + g_{3,1} + g_{3,2}}$$

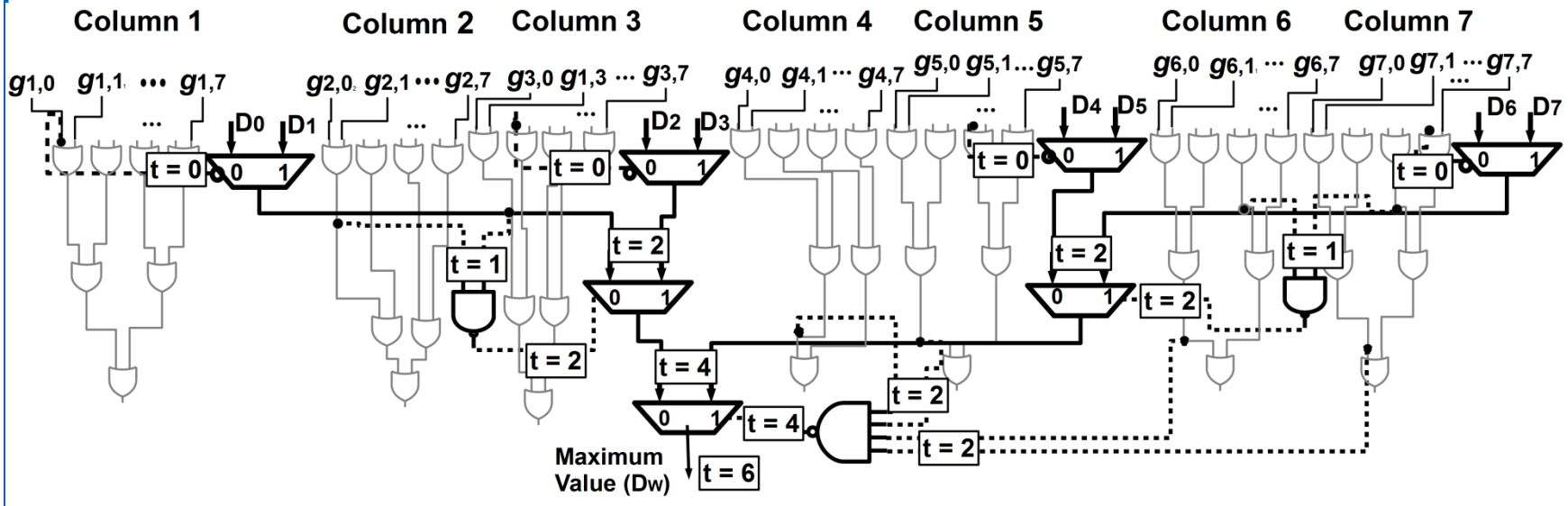
(1)

D3 = 10001
D2 = 11010
D1 = 01001
D0 = 11010

Proposed MF Circuit Topology

Data Selection Unit (DSU)

- Binary tree configuration of *k*-bit multiplexers
- Parallel operation with AGU.
- NO (theoretical) delay overhead



Proposed MF Circuit Topology

Generalization of AB

- Can be used as a minimum finder circuit by changing input connections.
- Any type of comparison operation (>, <, ≥, ≤) can be used.

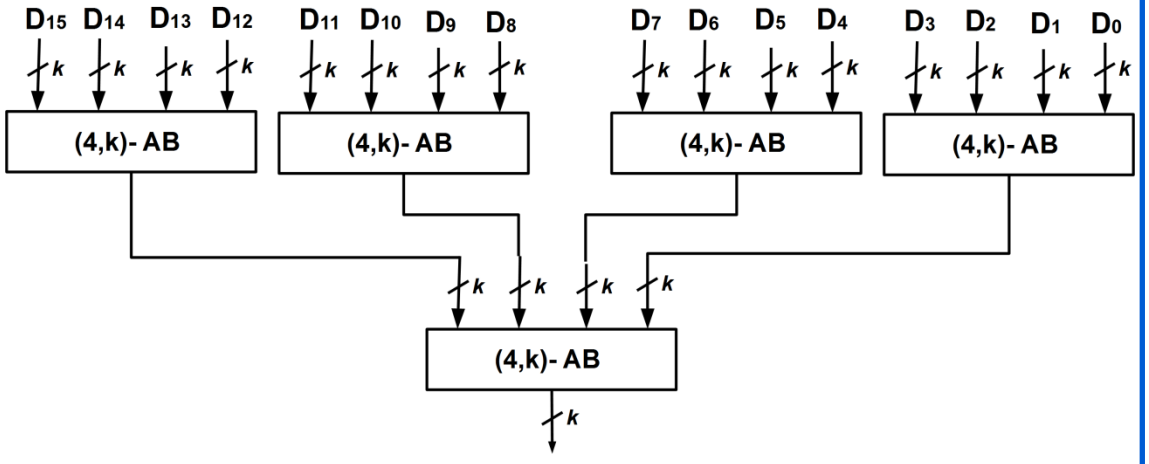
$$g_{xy} = \begin{cases} 0 & if\ y = x, \\ D_y\ op\ D_x & if\ y < x, \\ \overline{g_{yx}} & if\ y > x, \end{cases}$$

op	Finding Maximum		Finding Minimum	
	AGU Inputs	***Priority***	***AGU Inputs***	***Priority***
>	g_{ix}	Descending	g_{yi}	Ascending
≥	g_{ix}	Ascending	g_{yi}	Descending
<	g_{yi}	Ascending	g_{ix}	Descending
≤	g_{yi}	Descending	g_{ix}	Ascending

Proposed MF Circuit Topology
Using AB as a Building Block

- Two potential problems of AB (for large values of *n*):
  - Area
  - Fanout
- Solution: Use AB as a building block

Example: Quad-Tree Topology (QT)



Synthesis Results

Experimental Setup

- Synopsys Design Compiler (DC) v2010.12
  - UMC Faraday 180nm worst-case (slow) standard-cell library with a wire-load model
- Verification Script
- Synthesis Script
  - Binary search based, iterative synthesis script written in TCL

Theoretical comparison of all methods

Method	Time Complexity	Area Complexity
AT [8]	$O(k \times \log_2 n + k)$	$O(k \times n)$
RT [9], ST[10]	$O(k \times \log_2 n + k)$	$O(n)$
TBT	$O(\log_2 k \times \log_2 n + \log_2 n)$	$O(k \times n)$
RC [11]	$O(k + \log_2 n)$	$O(k \times n)$
CS [11]	$O(\sqrt{k}) + O(\lceil \sqrt{k} - \log_2 n \rceil \times \log_2 n)$	$O(k \times n)$
CL [11]	$O(\log_2 k \times \log_2 n)$	$O(k \times n)$
ML [12]	$O(\log_2 k + \log_2 n + n)$	$O(k \times 2^k)$
LZC [12]	$O(\log_2 k + \log_2 n + k)$	$O(2^k \times (n + k))$
AB	$O(\log_2 k + \log_2 n)$	$O(k \times n^2)$

Synthesis Results

Timing results in nanoseconds

- 1.2–2.1 times faster than the closest competitor
- On the average, 1.6 times faster than the closest competitor

$n \times k$	AB_1	AB_2	RC	CL	CS	TBT	QT
4 × 8	1.65	**1.64**	2.37	2.19	2.24	2.22	1.64
4 × 16	1.90	**1.87**	3.25	2.75	2.71	2.74	1.87
4 × 32	2.08	**2.07**	4.85	3.25	3.26	3.14	2.07
8 × 8	1.94	**1.97**	3.03	2.84	3.06	3.09	2.76
8 × 16	2.17	**2.14**	4.02	3.77	3.66	3.89	3.08
8 × 32	2.48	**2.51**	5.81	4.67	4.53	4.52	3.67
16 × 8	2.26	**2.27**	3.59	3.45	3.76	4.09	2.97
16 × 16	2.57	**2.64**	4.76	4.60	4.72	5.06	3.44
16 × 32	2.86	**2.86**	6.72	6.02	5.79	6.00	3.97
32 × 8	2.73	**2.81**	4.09	3.98	4.33	4.94	4.15
32 × 16	3.07	**3.05**	5.52	5.32	5.57	6.19	4.94
32 × 32	3.47	**3.49**	7.92	7.21	7.25	7.71	5.74
64 × 8	3.26	**3.34**	4.61	4.70	5.27	6.15	4.53
64 × 16	4.56	**5.32**	6.43	6.48	6.67	7.76	5.20
64 × 32	5.62	**5.69**	9.02	8.42	8.62	9.37	6.14

Synthesis Results

Normalized area results

- On the average, 5.7 times worse than the fastest competitor

$n \times k$	AB_1	AB_2	RC	CL	CS	TBT	QT
4 × 8	1.01	1.00	0.98	1.08	1.03	**0.78**	1.00
4 × 16	1.09	1.00	0.97	1.01	0.98	**0.79**	1.00
4 × 32	0.93	1.00	1.01	1.52	1.00	**0.77**	1.00
8 × 8	1.06	1.00	0.57	0.53	0.66	**0.49**	0.63
8 × 16	1.02	1.00	0.49	0.49	0.58	**0.39**	0.62
8 × 32	1.04	1.00	0.52	0.76	0.59	**0.44**	0.58
16 × 8	1.01	1.00	0.26	0.29	0.30	**0.23**	0.30
16 × 16	0.99	1.00	0.23	0.23	0.27	**0.19**	0.30
16 × 32	0.88	1.00	0.25	0.35	0.27	**0.19**	0.24
32 × 8	1.04	1.00	0.14	0.16	0.18	**0.14**	0.17
32 × 16	0.98	1.00	0.14	0.13	0.16	**0.12**	0.16
32 × 32	0.98	1.00	0.14	0.18	0.15	**0.11**	0.14
64 × 8	0.99	1.00	0.07	0.08	0.09	**0.06**	0.10
64 × 16	0.52	1.00	0.06	0.06	0.07	**0.05**	0.07
64 × 32	0.95	1.00	0.06	0.07	0.07	**0.05**	0.07

Synthesis Results

Normalized Area-Timing Product (ATP) results

- On the average, 3.8 times worse than the fastest competitor

$n \times k$	AB_1	AB_2	RC	CL	CS	TBT	QT
4 × 8	1.01	**1.00**	1.39	1.43	1.39	1.04	1.00
4 × 16	1.11	**1.00**	1.69	1.49	1.41	1.15	1.00
4 × 32	0.93	**1.00**	2.36	2.39	1.57	1.17	1.00
8 × 8	1.04	1.00	0.87	**0.76**	1.03	0.77	0.88
8 × 16	1.03	1.00	0.92	0.87	0.99	**0.71**	0.88
8 × 32	1.03	1.00	1.21	1.42	1.06	**0.80**	0.85
16 × 8	1.01	1.00	0.41	0.44	0.49	0.41	**0.40**
16 × 16	0.96	1.00	0.41	0.39	0.49	**0.36**	0.39
16 × 32	0.88	1.00	0.58	0.74	0.55	0.41	**0.34**
32 × 8	1.01	1.00	**0.21**	0.22	0.27	0.24	0.25
32 × 16	0.99	1.00	0.25	0.23	0.29	**0.24**	0.26
32 × 32	0.98	1.00	0.33	0.36	0.32	0.25	**0.23**
64 × 8	0.96	1.00	**0.10**	0.11	0.14	0.11	0.13
64 × 16	0.44	1.00	0.07	0.07	0.08	**0.07**	0.07
64 × 32	0.94	1.00	0.10	0.11	0.10	**0.08**	0.07

Experimental Work
Quad Tree (QT) results

- QT: On the average
  - 1.24 times faster than the closest competitor (except AB)
  - Has only 1.1 times larger area than the fastest competitor (except AB)
  - Has only 1.03 times larger ATP than the most ATP-efficient competitor

- QT: On the average
  - 1.3 times slower than AB
  - 5.1 times more area-efficient than AB
  - 4.2 times more ATP-efficient than AB

Synthesis Results

DSU Delay Overhead

- NO theoretical delay overhead
- In practice, 14.5% delay overhead on the average

$n \times k$	AB_2 (without DSU)	AB_2	Delay Overhead (%)
4 × 8	1.47	1.66	12.93
4 × 16	1.64	1.87	14.02
4 × 32	1.74	2.07	18.97
8 × 8	1.77	1.97	11.30
8 × 16	1.96	2.14	9.18
8 × 32	2.18	2.51	15.14
16 × 8	2.13	2.27	6.57
16 × 16	2.45	2.64	7.76
16 × 32	2.67	2.86	7.12
32 × 8	2.55	2.81	10.20
32 × 16	2.90	3.05	5.17
32 × 32	3.26	3.49	7.06
64 × 8	3.13	3.34	6.71
64 × 16	3.64	5.32	46.15
64 × 32	4.08	5.69	39.46

Conclusions

- We propose a fast circuit topology (AB) to find the maximum element of a set of binary numbers.

- We wrote code generators
  - that produces RTL and self-checking testbench code
  - for AB as well as competitive existing MF circuit topologies

- We automated
  - design,
  - verification,
  - synthesis, and
  - extraction of the area/timing results

  via a batch script.

Conclusions

- AB yields better performance (latency) than any existing topology for any k and n.
- AB introduces a significant area overhead for $n > 8$.
- AB has better performance (latency) and area than any existing topology for $n = 4$.
- We showed that AB can be used as a building block in an MF circuit topology to overcome its area and fanout disadvantages.

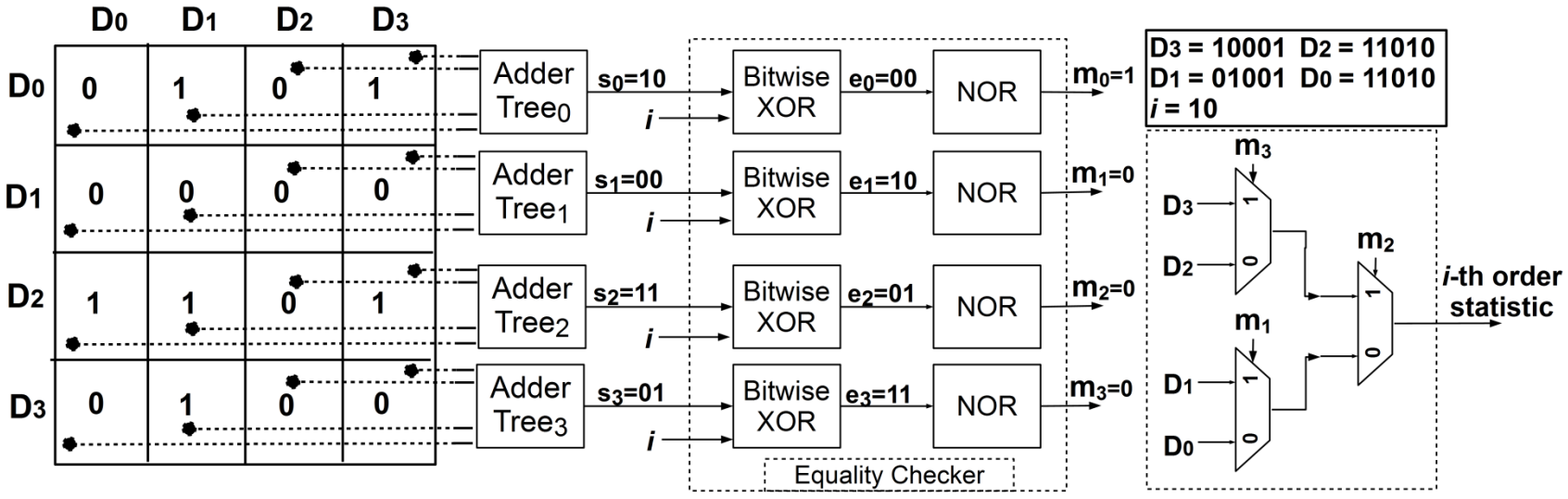
Future Work

- Looking for more area-efficient implementations
- Synthesis results on FPGA
- *i*-th Maximum

Future Work

Generalization of AB to i-th Maximum

- Can be used to find *i*-th smallest element (*i*-th order statistic) a set
- $T = O(\log_2 k + \log_2 n)$
  $A = O(k \times n^2)$



Thank You

fatih.ugurdag@ozyegin.edu.tr

Bosphorus University

ÖZYEĞİN UNIVERSITY

Yeditepe University