An Optimal Algorithm for Allocation, Placement, and Delay Assignment of Adjustable Delay Buffers for Clock Skew Minimization in Multi-Voltage Mode Designs

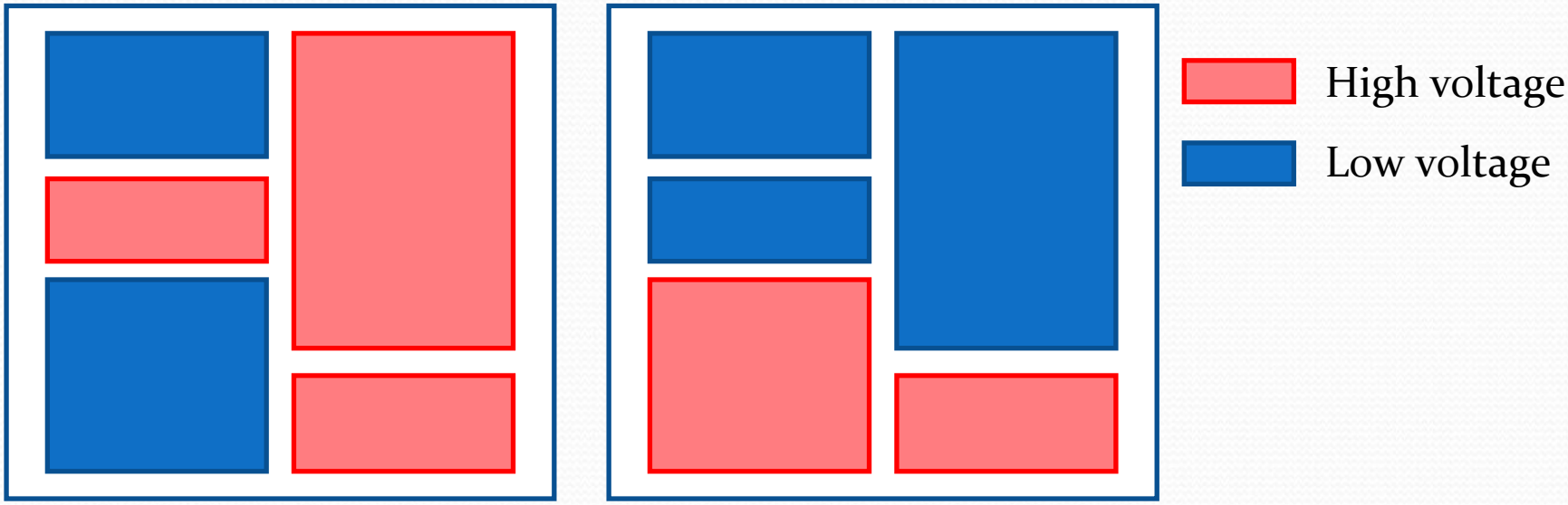
Kyoung-Hwan Lim and Taewhan Kim
Seoul National University

Table of Contents

- In synchronous circuit design, all sequential elements in the design are synchronized by a unified signal – clock signal

- Ideally, clock signal should be reached to all sequential elements at the same time for its synchronicity
  - Practical issues
    - Different wire length
    - Different buffer characteristics
  - Achieving zero skew is still difficult!

- Multi-voltage mode design

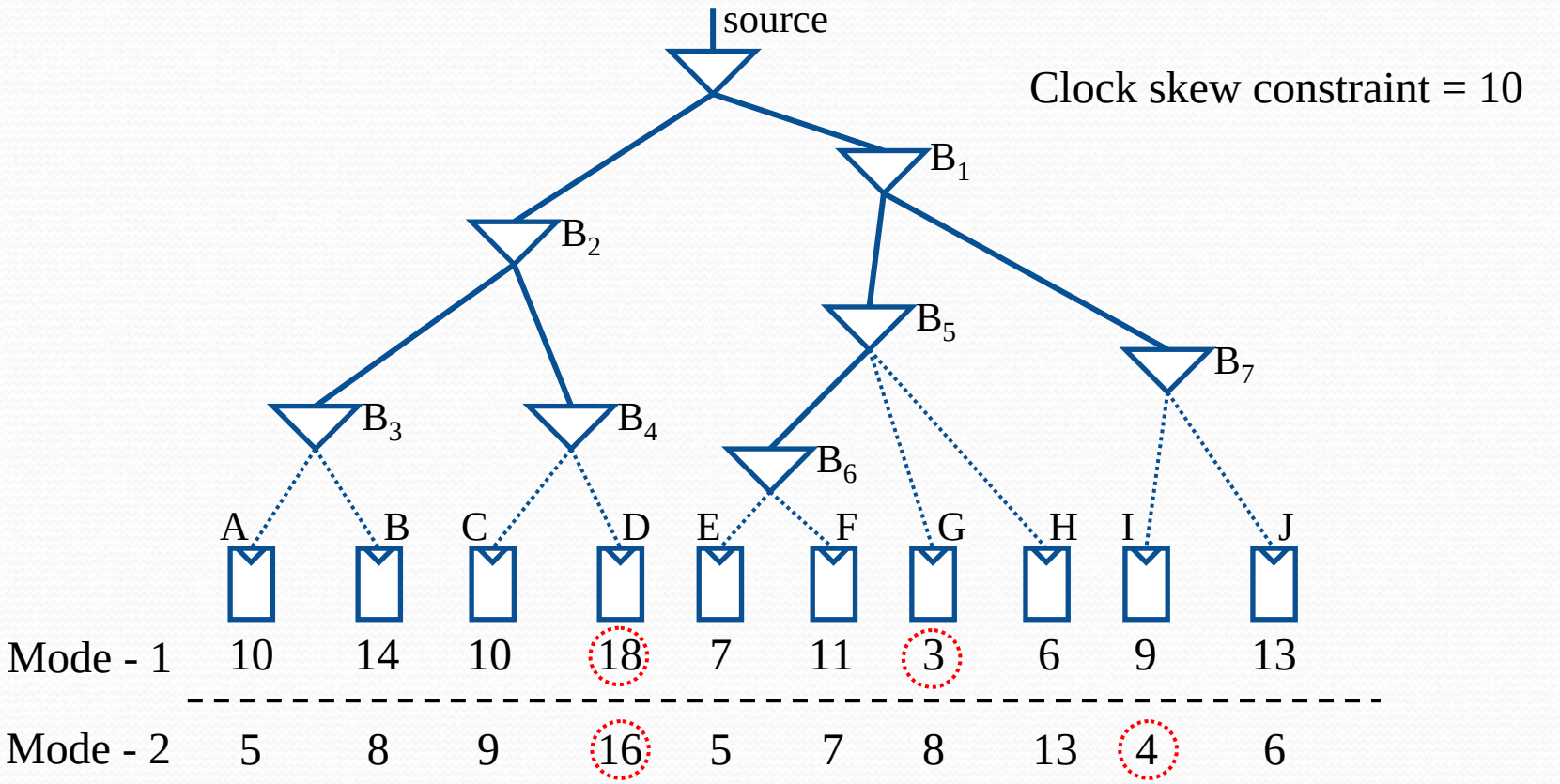


Whenever the power mode is changed, the delay of clock tree is also changed.
How can we simultaneously meet the skew constraint for every power mode?

- Adjustable Delay Buffer (ADB)
  - We can change its delay dynamically
  - Changing delay value depending on the current power mode – simple and easy to use
- Problems
  - How many ADBs should be used?
  - Where should we place ADBs?
  - What delay value should be for each power mode?

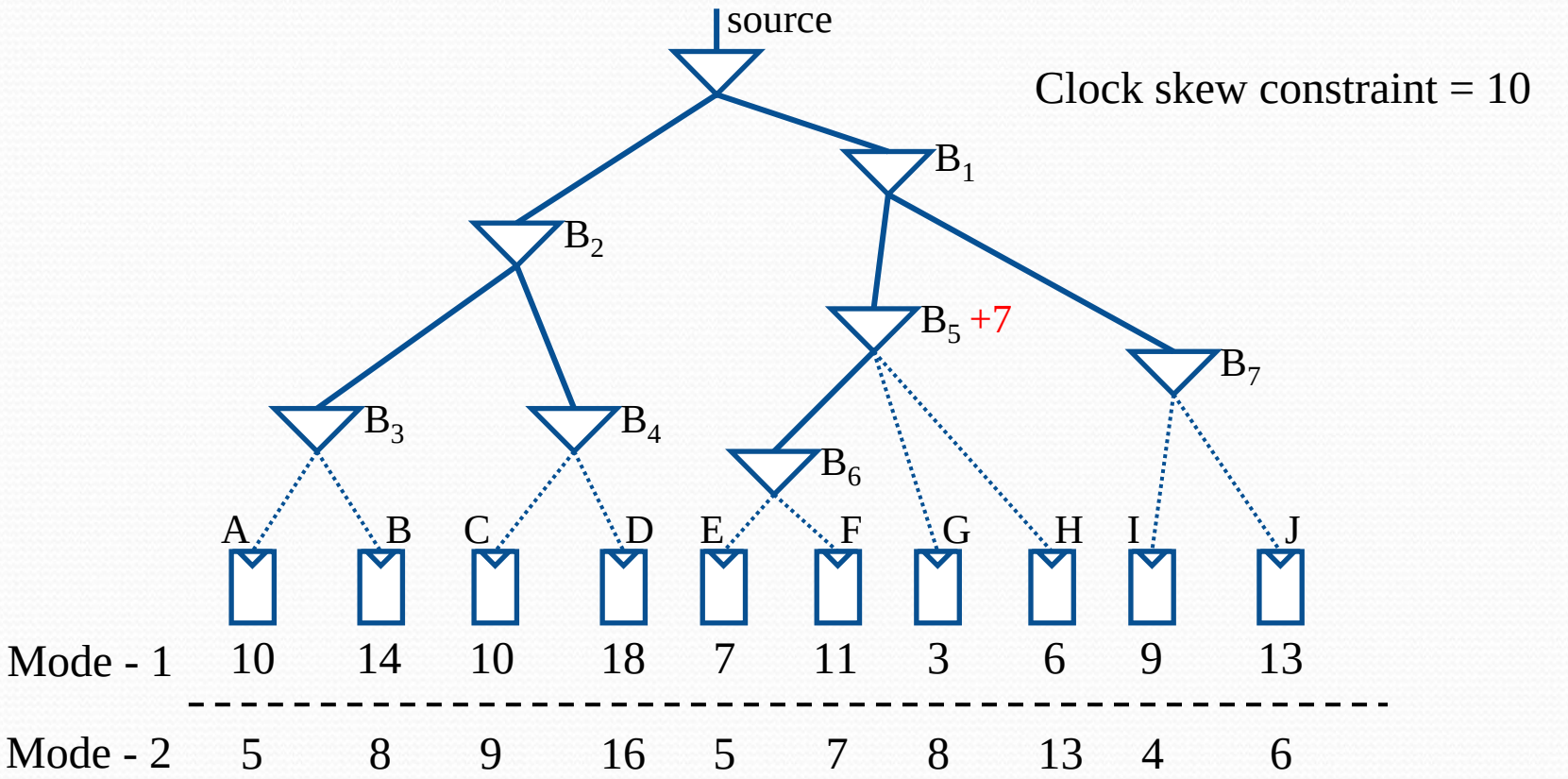
- Related work
  - Y. –S. Su, W. –K . Hon, C. –C. Yang, S. –C Chang, and Y. –J Chang, “Value assignment of adjustable delay buffers for clock skew minimization in multi-voltage mode designs,” *ICCAD*, 2009.
  - Only work using ADB for clock tree generation in multi-voltage mode designs
    - Optimal ADB adjustment value for given ADB positions
    - Finding ADB positions is still based on heuristics
      - Cannot guarantee minimum number of ADBs

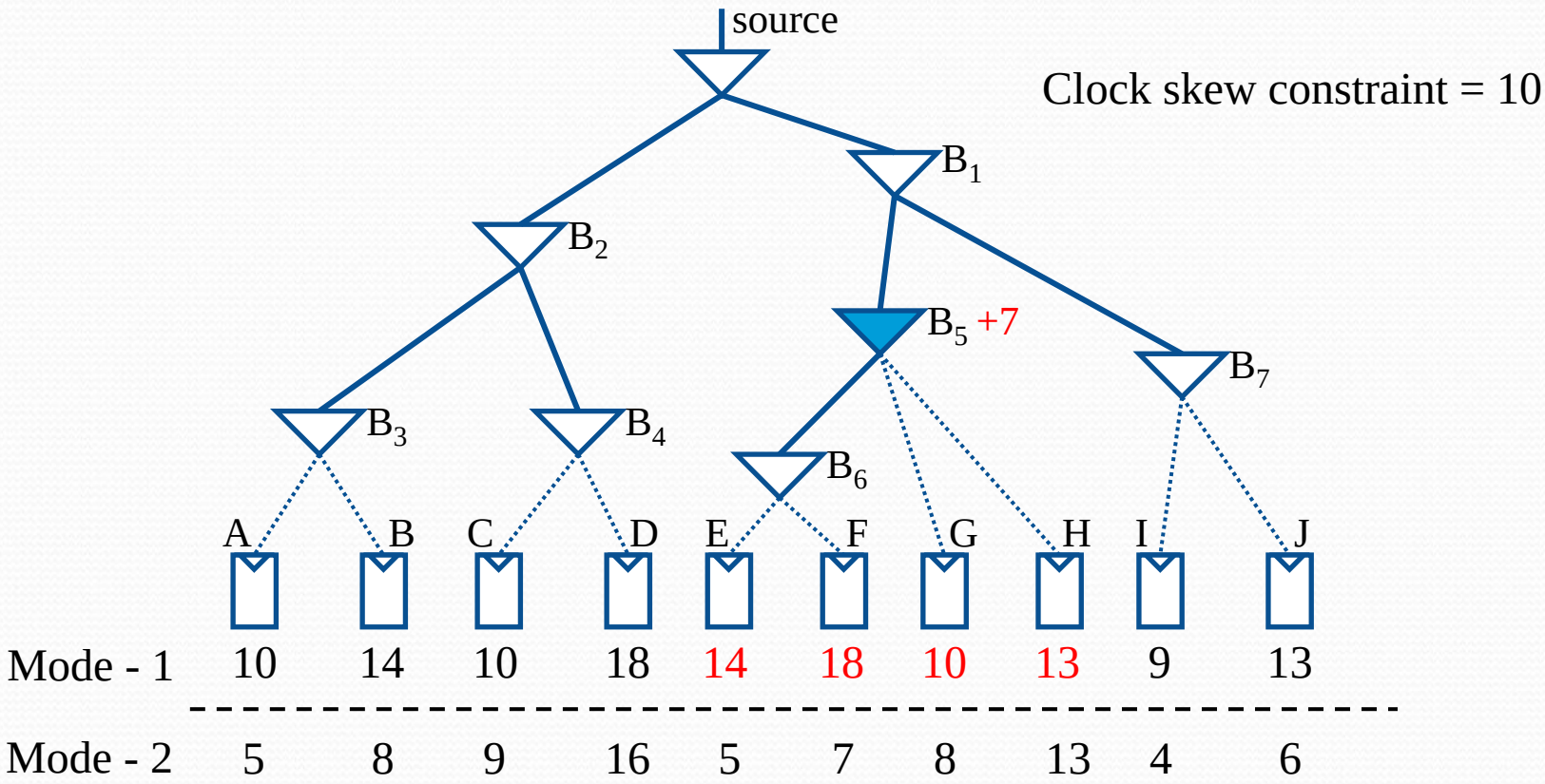
Table of Contents

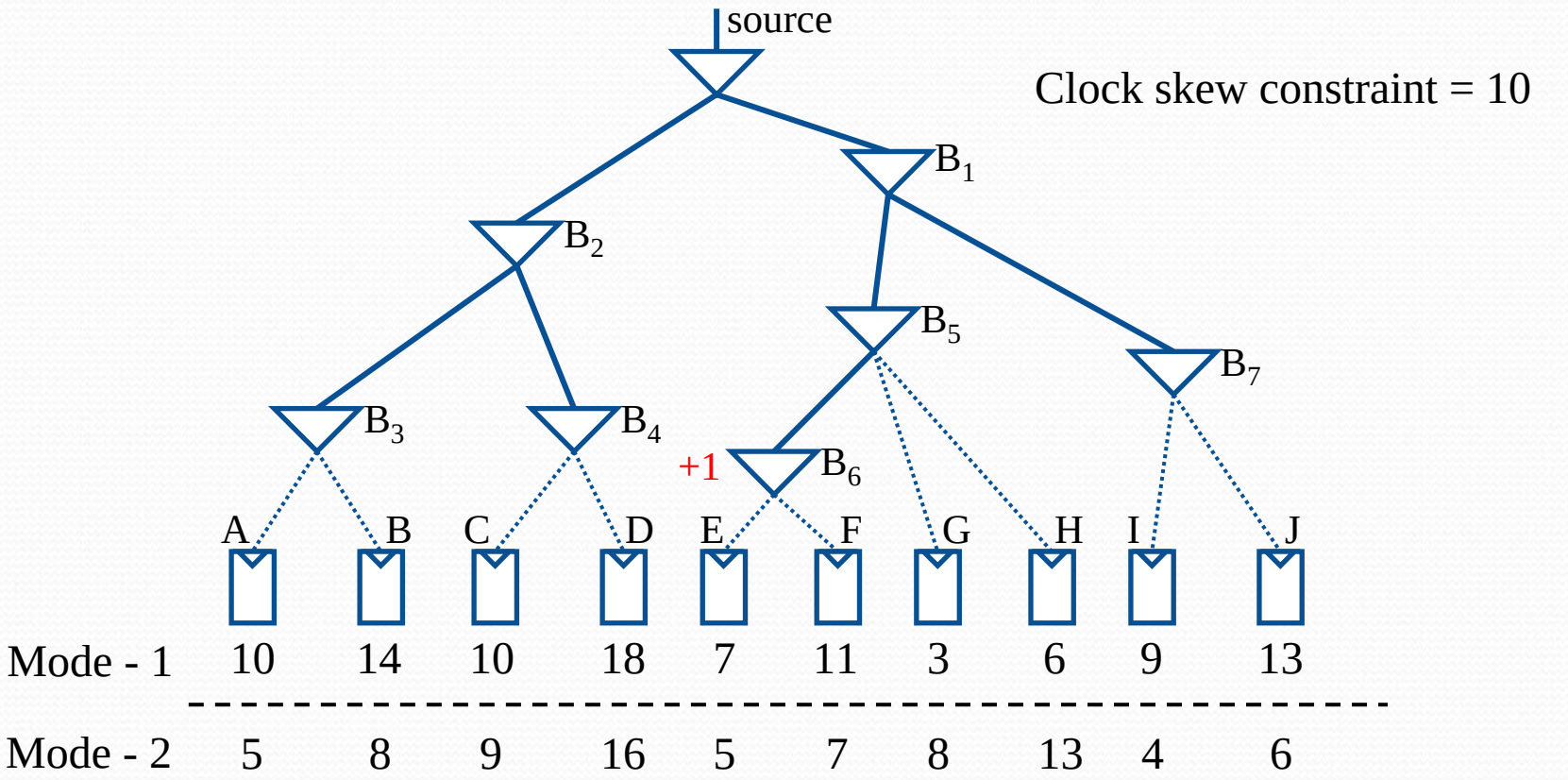

source
Clock skew constraint = 10
B1
B2
B5
B7
B3
B4
B6
A
B
C
D
E
F
G
H
I
J
Mode - 1
10
14
10
18
7
11
3
6
9
13
Mode - 2
5
8
9
16
5
7
8
13
4
6

Violation!!

- Avoiding violation
  - Reconstruct clock tree which can adaptively minimize all clock skew under constraint for all power modes
    - Almost impossible
  - Using Adjustable Delay Buffer (ADB)
    - Insert ADB instead of normal buffer
    - Change delay of inserted ADBs depending on power mode
    - ADB insertion should be restricted because of high cost of using ADB


source
Clock skew constraint = 10
B1
B2
B5 +7
B7
B3
B4
B6
A
B
C
D
E
F
G
H
I
J
Mode - 1
10
14
10
18
7
11
3
6
9
13
Mode - 2
5
8
9
16
5
7
8
13
4
6

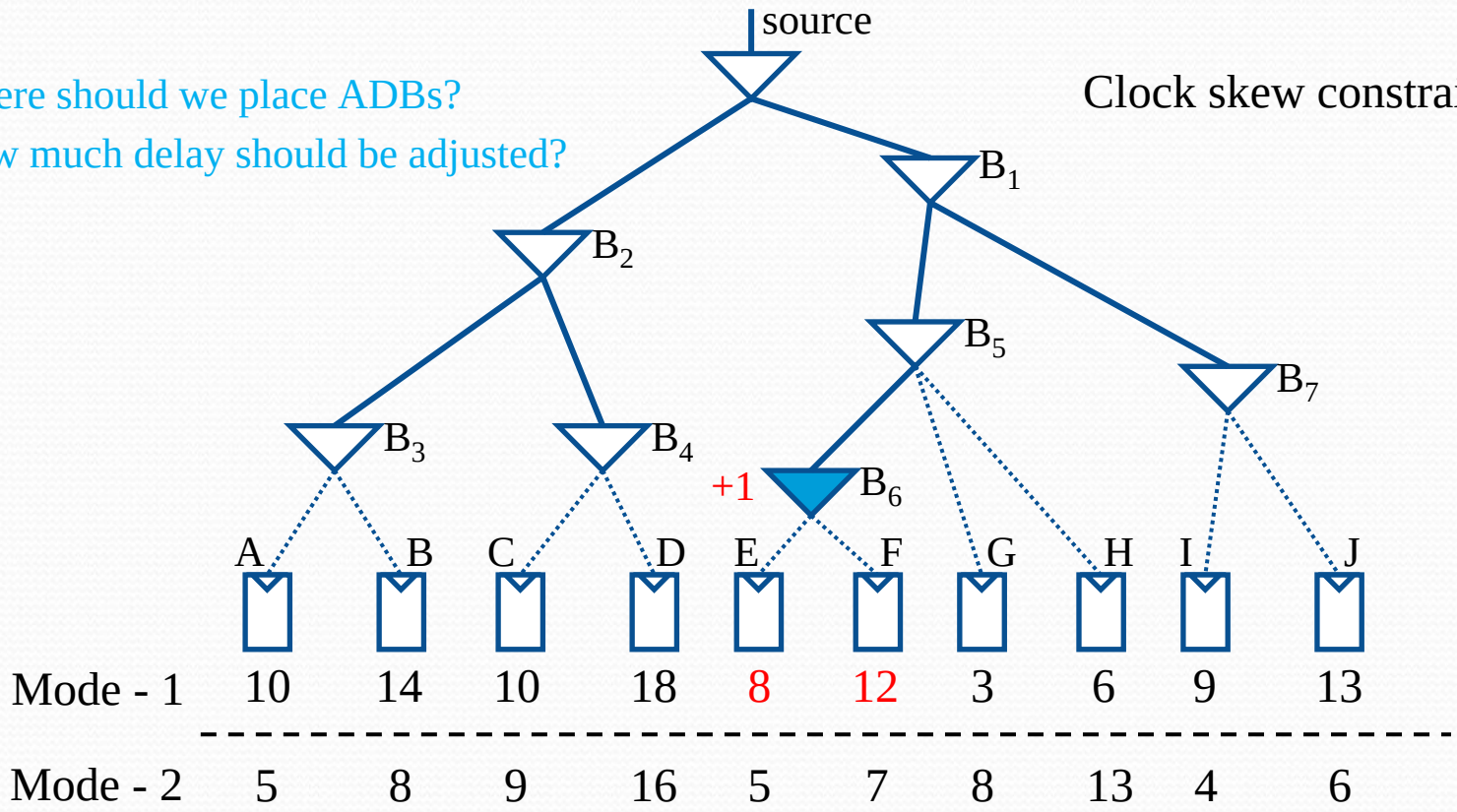

source
Clock skew constraint = 10
B1
B2
B5 +7
B7
B3
B4
B6
A
B
C
D
E
F
G
H
I
J
Mode - 1
10
14
10
18
14
18
10
13
9
13
Mode - 2
5
8
9
16
5
7
8
13
4
6

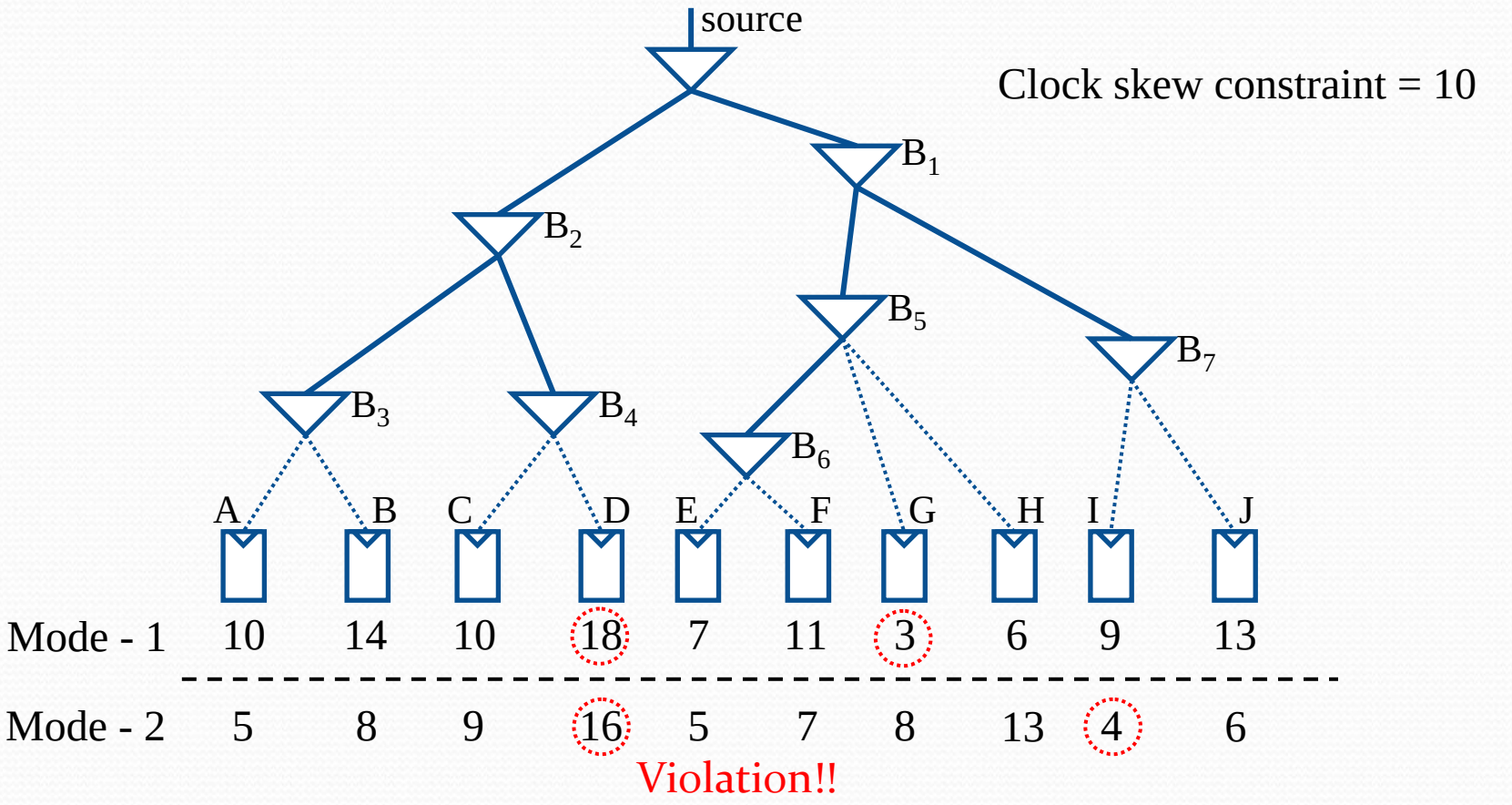

source
Clock skew constraint = 10
B1
B2
B5
B7
B3
B4
+1
B6
A
B
C
D
E
F
G
H
I
J
Mode - 1
10
14
10
18
7
11
3
6
9
13
Mode - 2
5
8
9
16
5
7
8
13
4
6

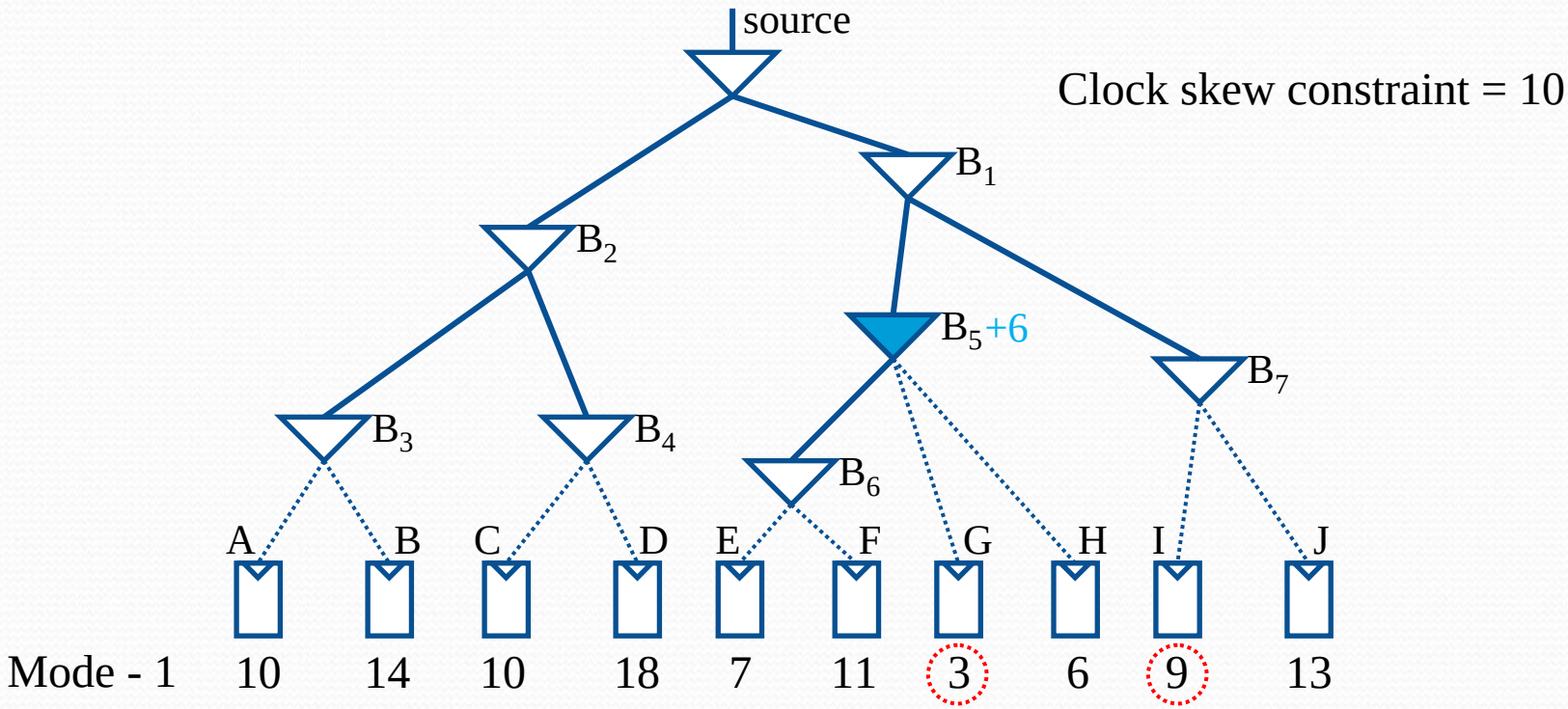
Where should we place ADBs?

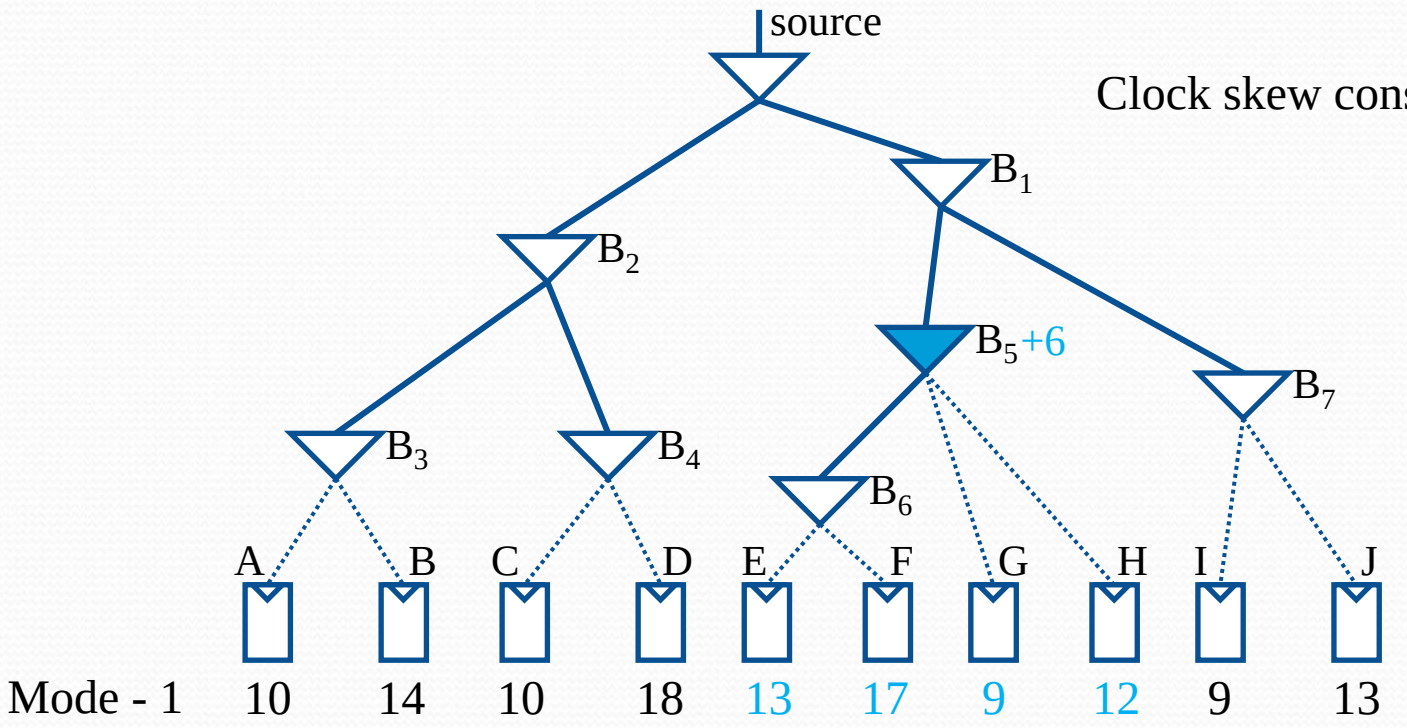
How much delay should be adjusted?

Clock skew constraint = 10



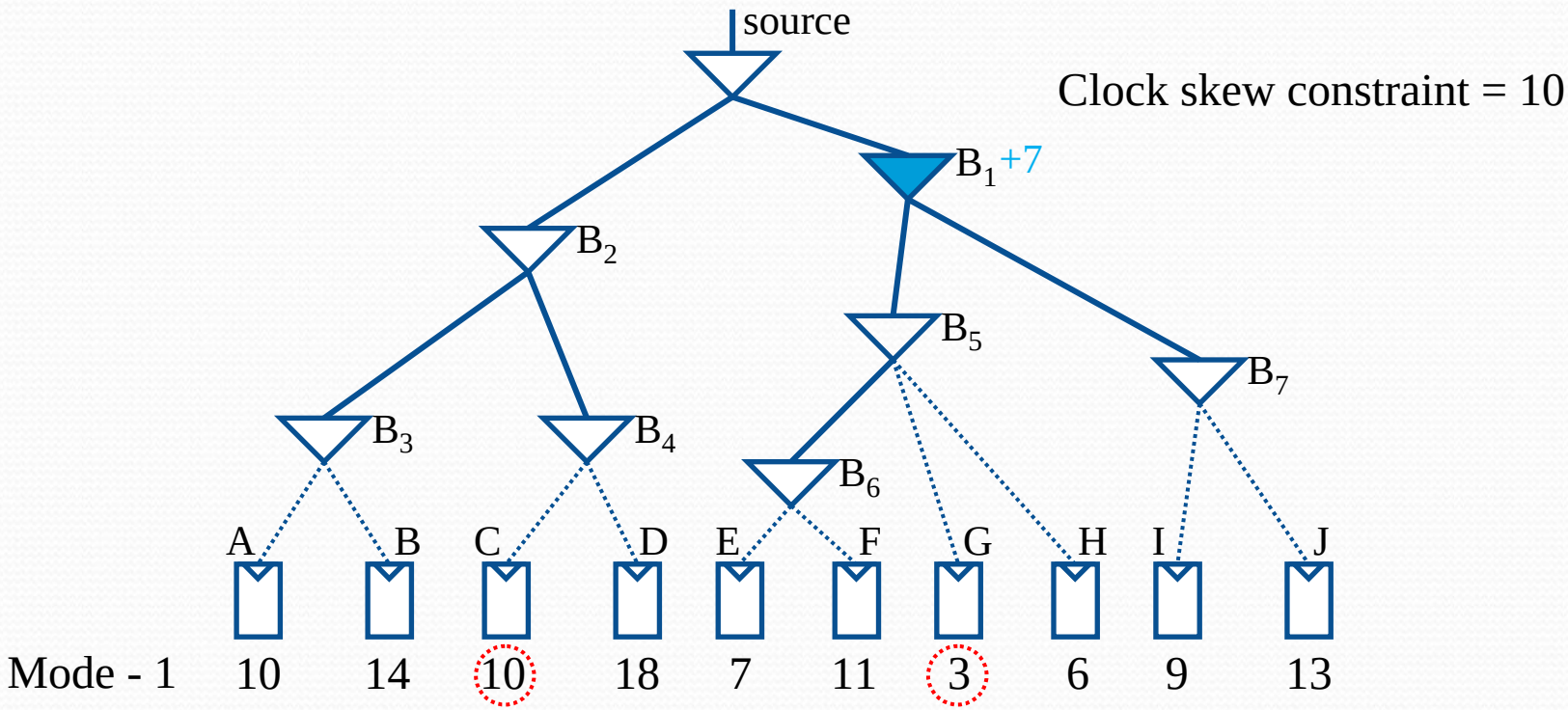

source
Clock skew constraint = 10
B1
B2
B5
B7
B3
B4
B6
A
B
C
D
E
F
G
H
I
J
Mode - 1
10
14
10
18
7
11
3
6
9
13
Mode - 2
5
8
9
16
5
7
8
13
4
6
Violation!!

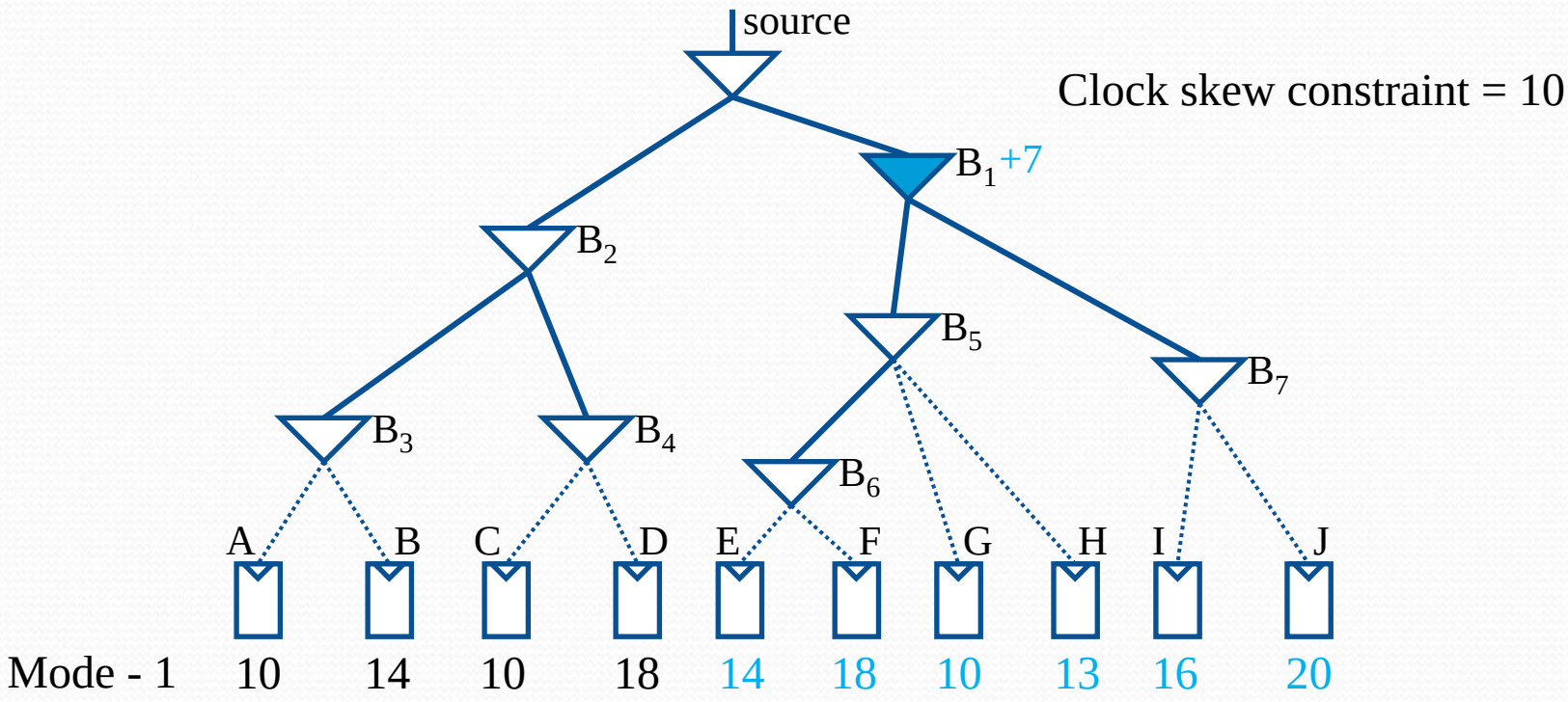

source
Clock skew constraint = 10
B1
B2
B5 +6
B7
B3
B4
B6
A
B
C
D
E
F
G
H
I
J
Mode - 1
10
14
10
18
7
11
3
6
9
13



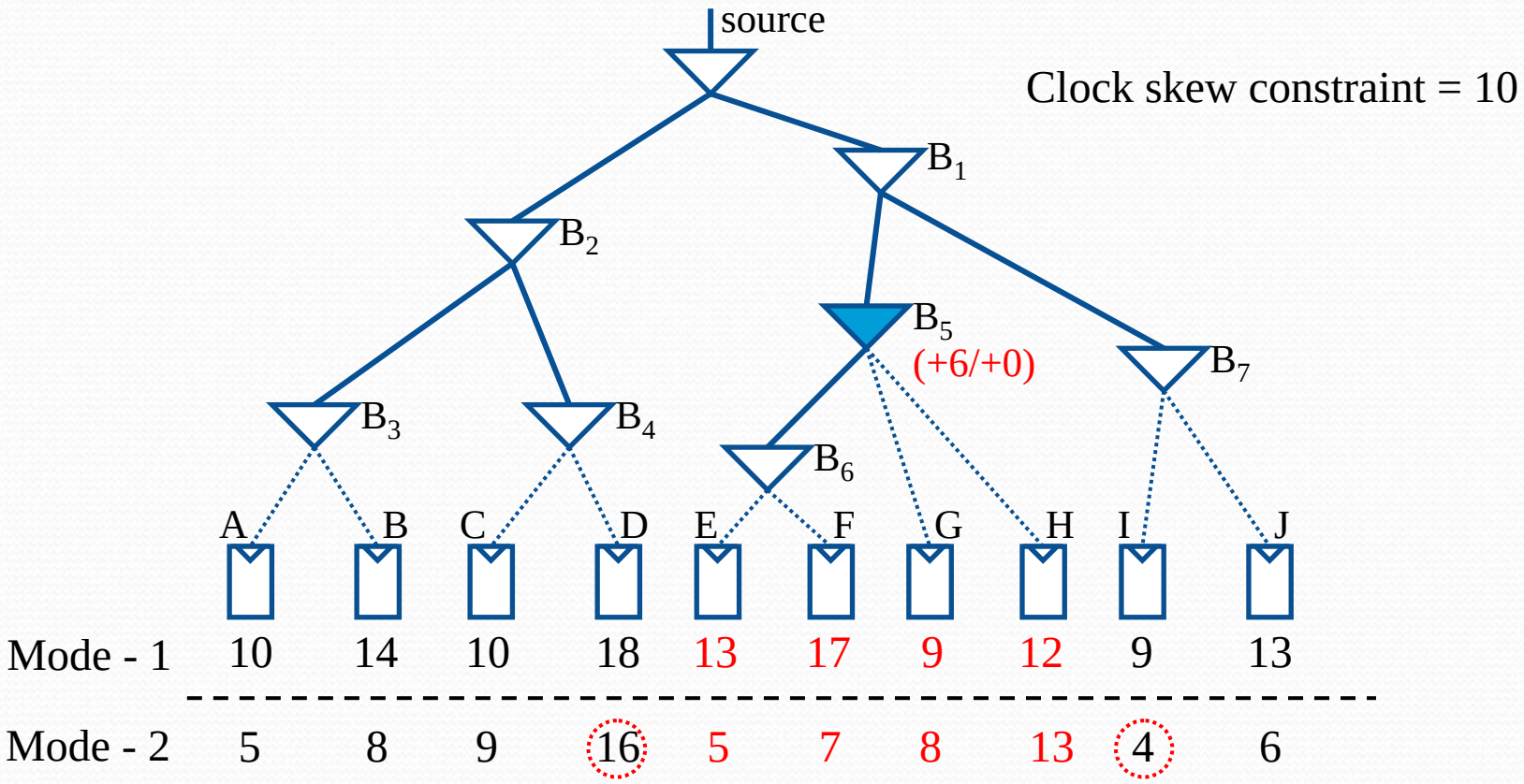
Clock skew constraint = 10

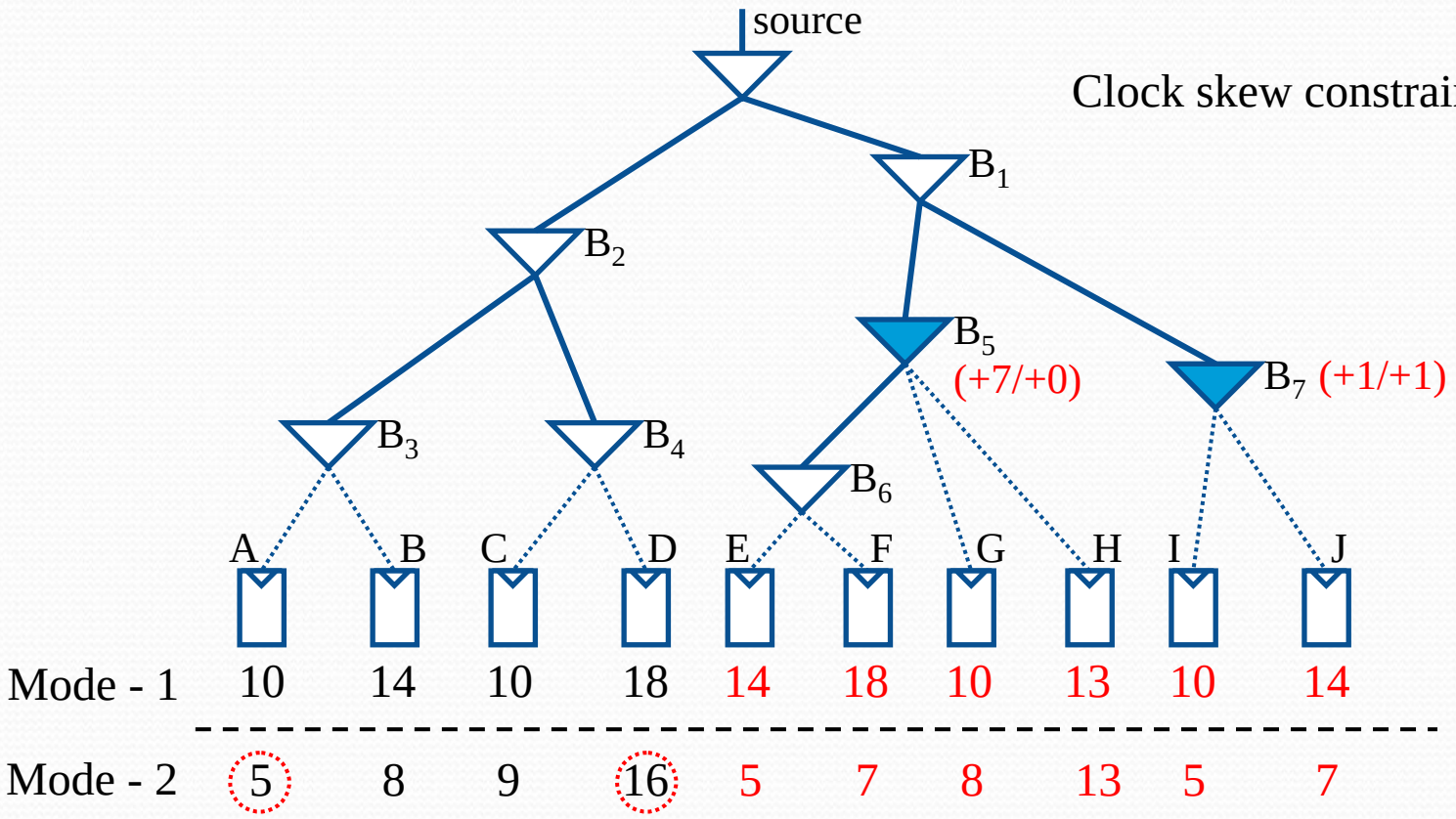
Clock skew of current clock tree at Mode – 1 = 18 – 9 = 9

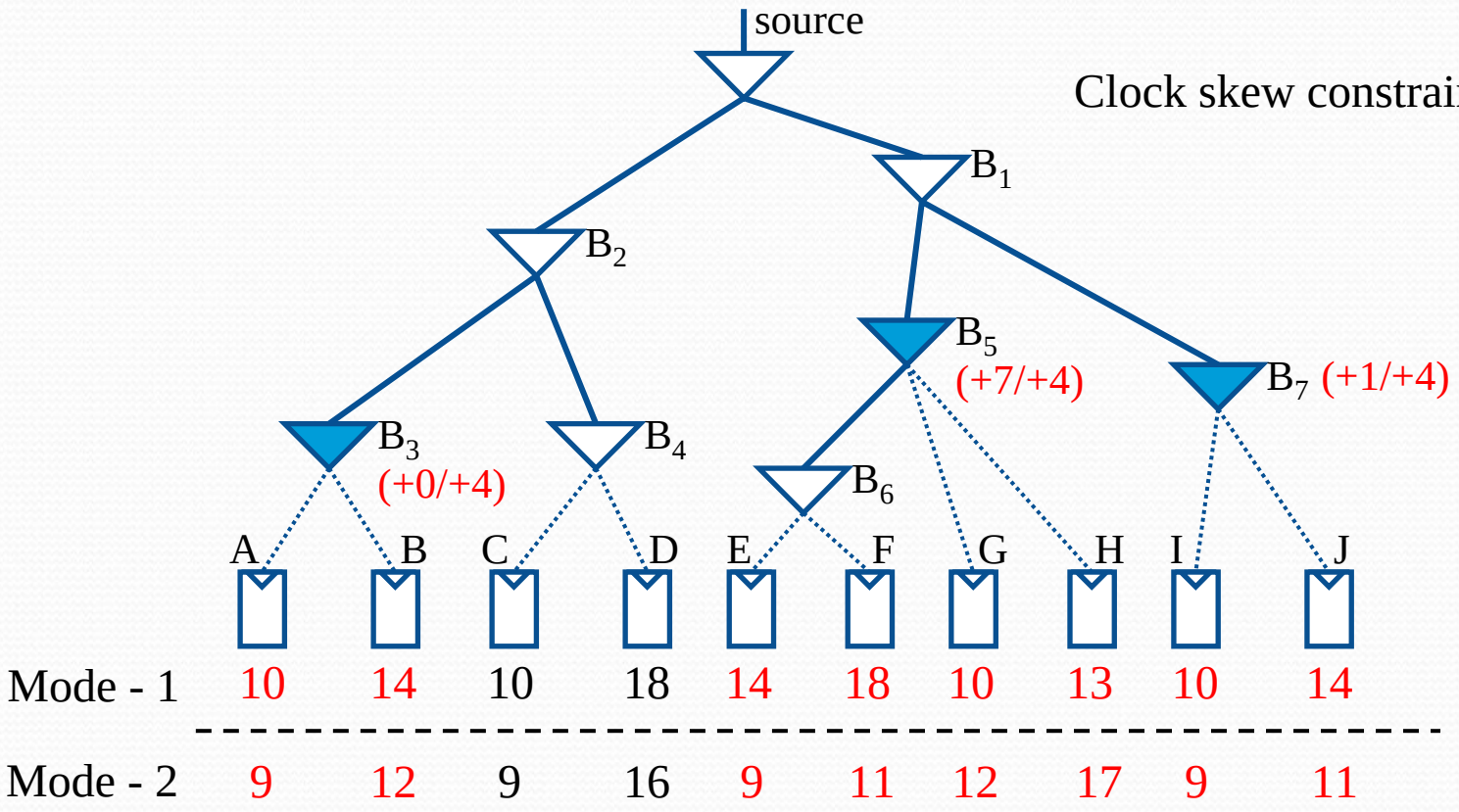

source
Clock skew constraint = 10
B1 +7
B2
B5
B7
B3
B4
B6
A
B
C
D
E
F
G
H
I
J
Mode - 1
10
14
10
18
7
11
3
6
9
13



Clock skew of current clock tree at Mode – 1 = 20 – 10 = 10


source
Clock skew constraint = 10
B1
B2
B5
(+6/+0)
B7
B3
B4
B6
A
B
C
D
E
F
G
H
I
J
Mode - 1
10
14
10
18
13
17
9
12
9
13
Mode - 2
5
8
9
16
5
7
8
13
4
6
Violation!!


source
Clock skew constraint = 10
B1
B2
B5
(+7/+0)
B7 (+1/+1)
B3
B4
B6
A
B
C
D
E
F
G
H
I
J
Mode - 1
10
14
10
18
14
18
10
13
10
14
Mode - 2
5
8
9
16
5
7
8
13
5
7
Violation!!



No Violation!!

- Traditional approach can find optimal delay assignment value for given ADB position
  - However, cannot check whether given ADB position is optimal or not
  - They found ADB positions in greedy manner
  - More comprehensive approach is still needed

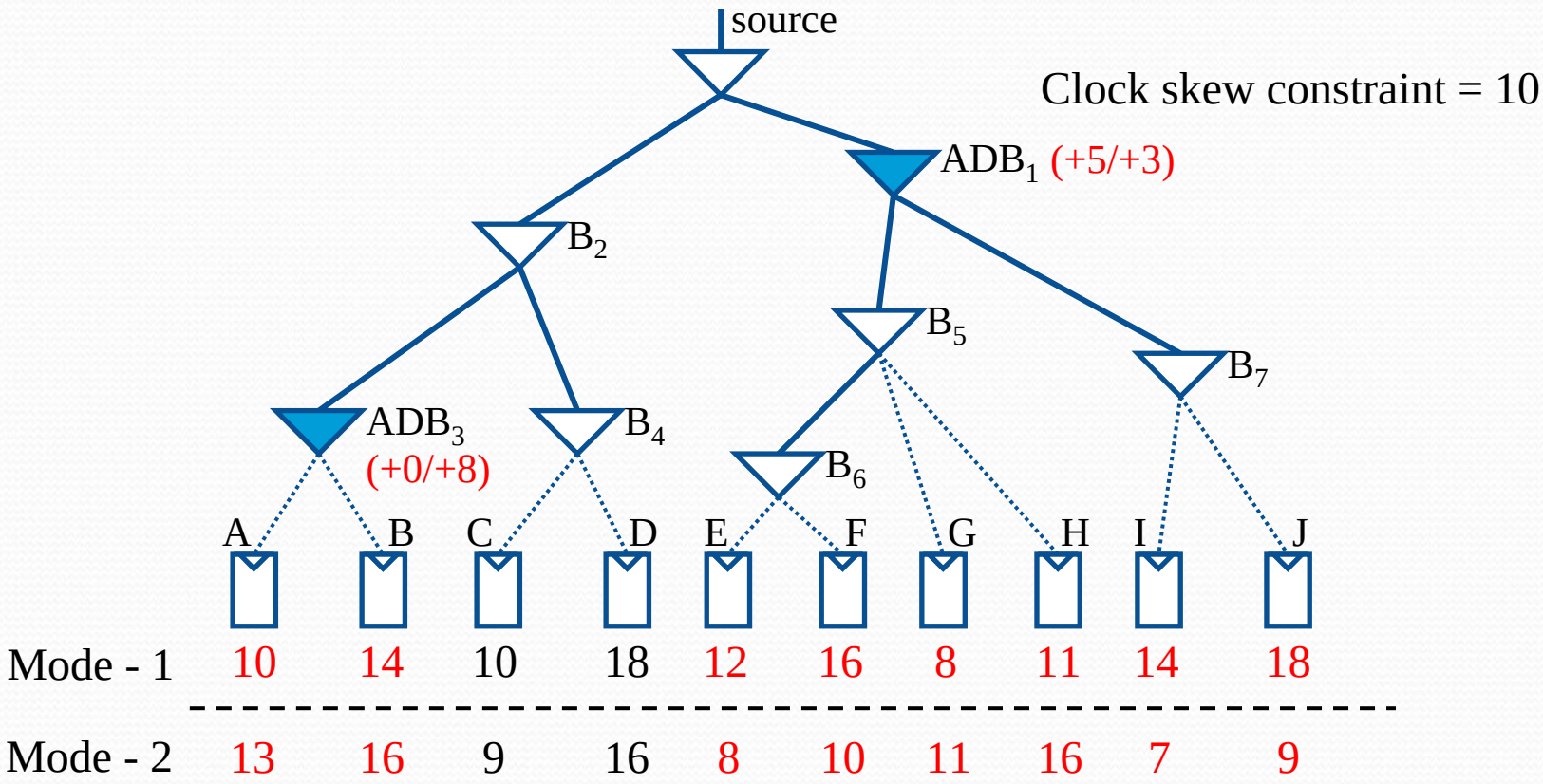
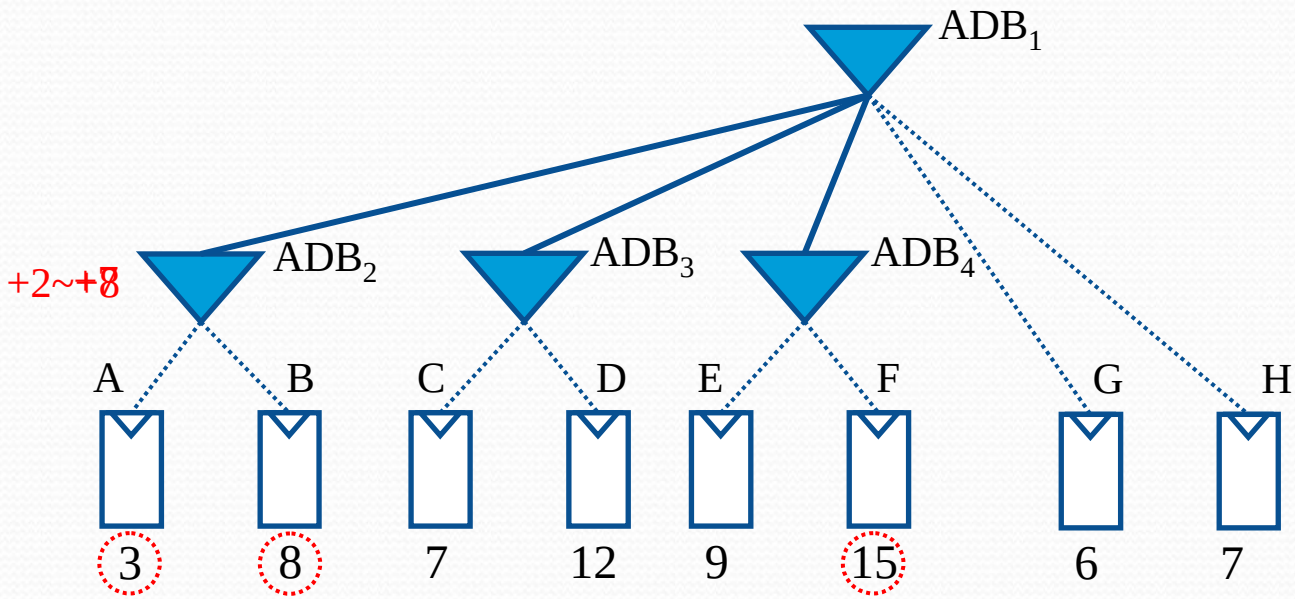

source
Clock skew constraint = 10
ADB1 (+5/+3)
B2
B5
B7
ADB3
(+0/+8)
B4
B6
A
B
C
D
E
F
G
H
I
J
Mode - 1
10
14
10
18
12
16
8
11
14
18
Mode - 2
13
16
9
16
8
10
11
16
7
9

Table of Contents

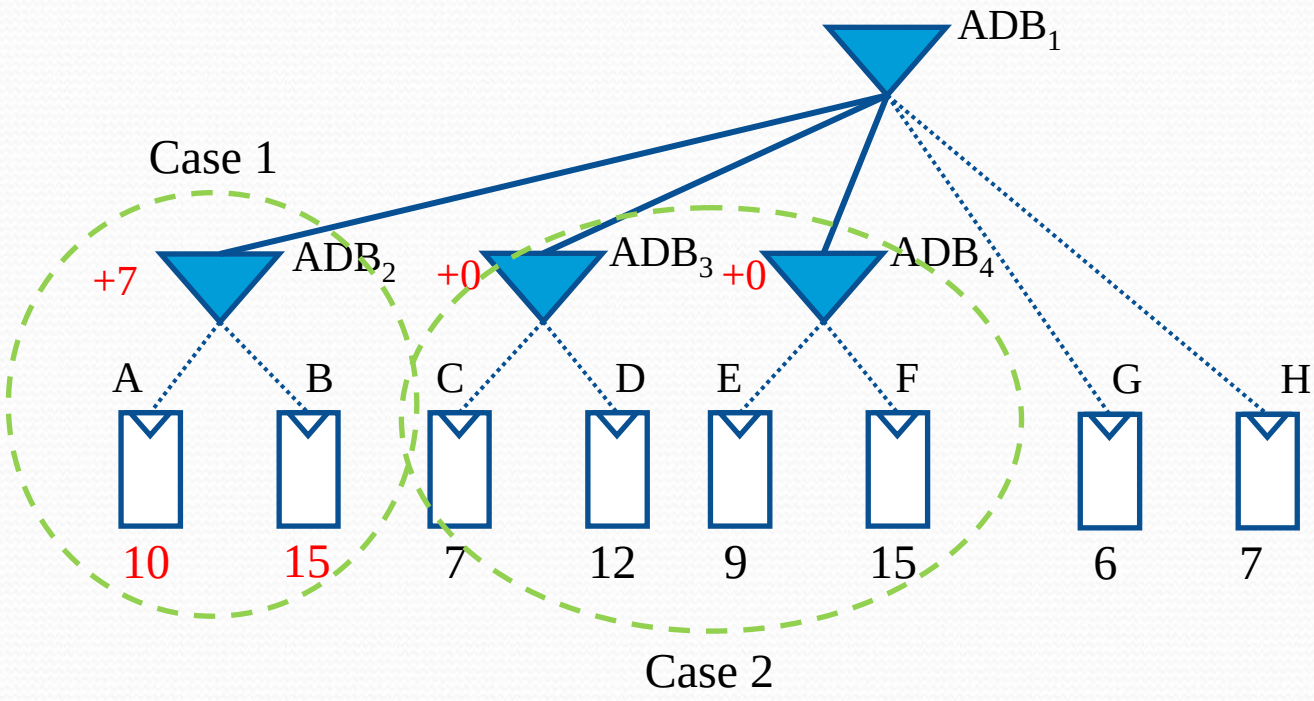
- Problem description
  - Given initial buffered clock tree $\mathcal{T}$, power modes, clock signal arrival time, and clock skew bound B
  - Allocate ADBs and replace buffers in $\mathcal{T}$
  - Assign delay increment value to each ADB for every power mode
  - Clock skew bound constraint should be satisfied
  - Minimize the number of ADBs used

Clock skew constraint = 10

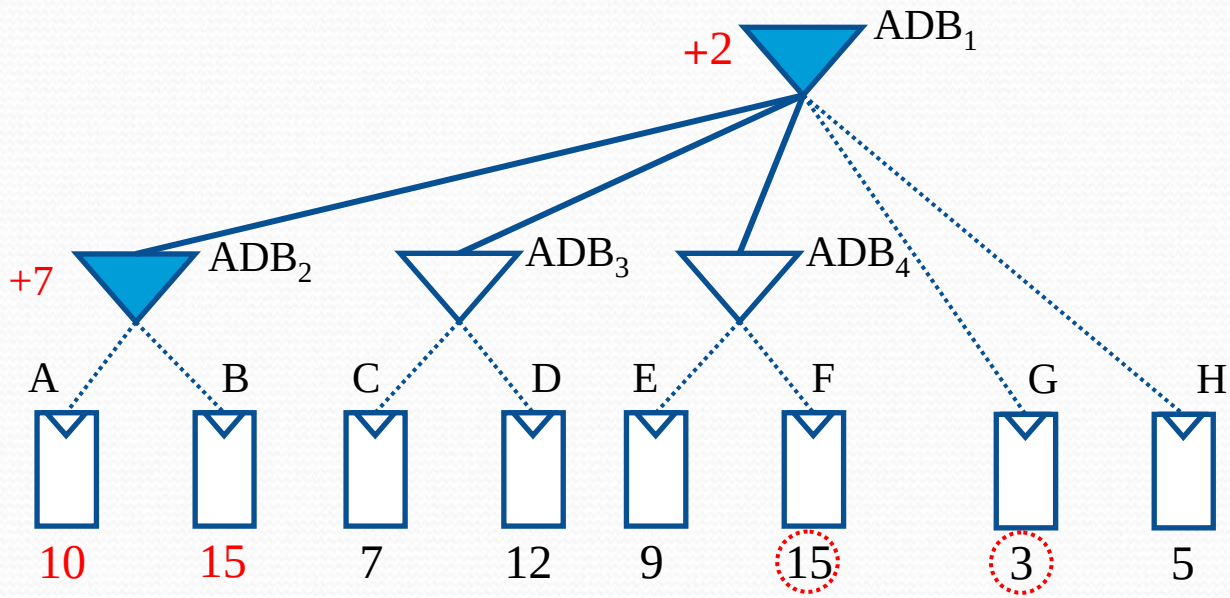


There exists at least one feasible adjustment value $LAT_v - LAT_2$

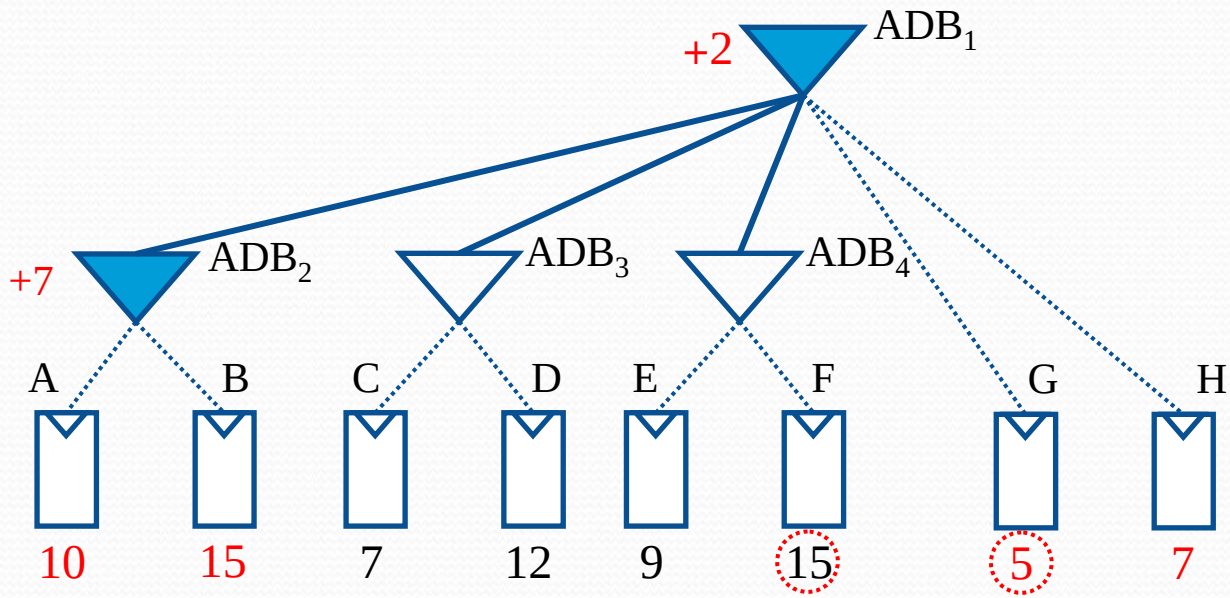
Clock skew constraint = 10



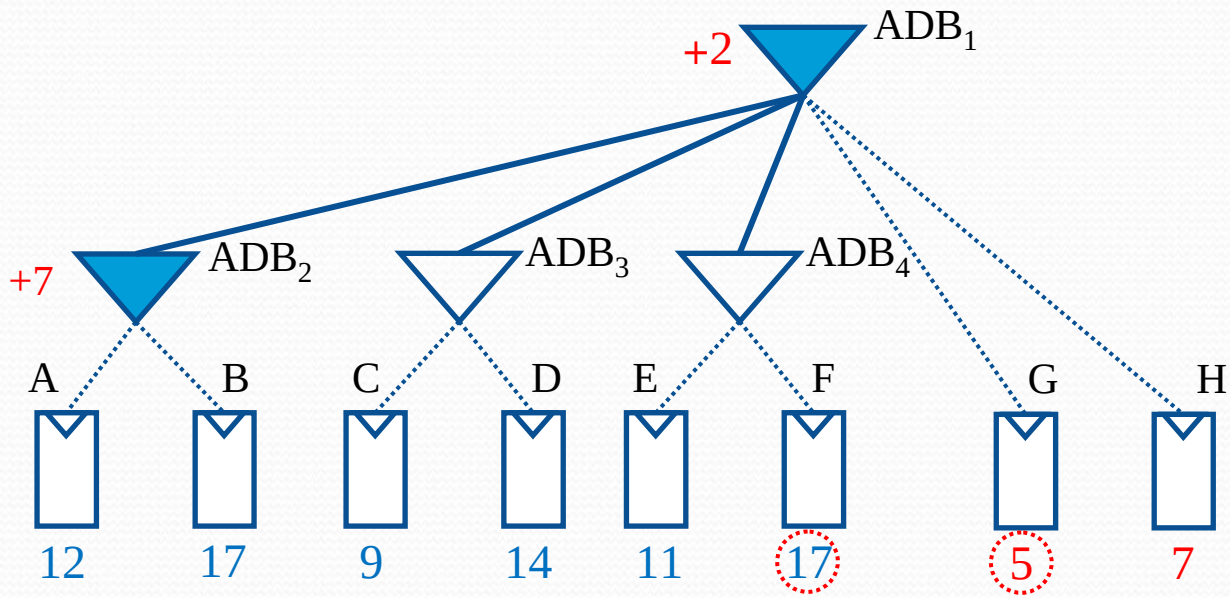
Clock skew constraint = 10



Clock skew constraint = 10



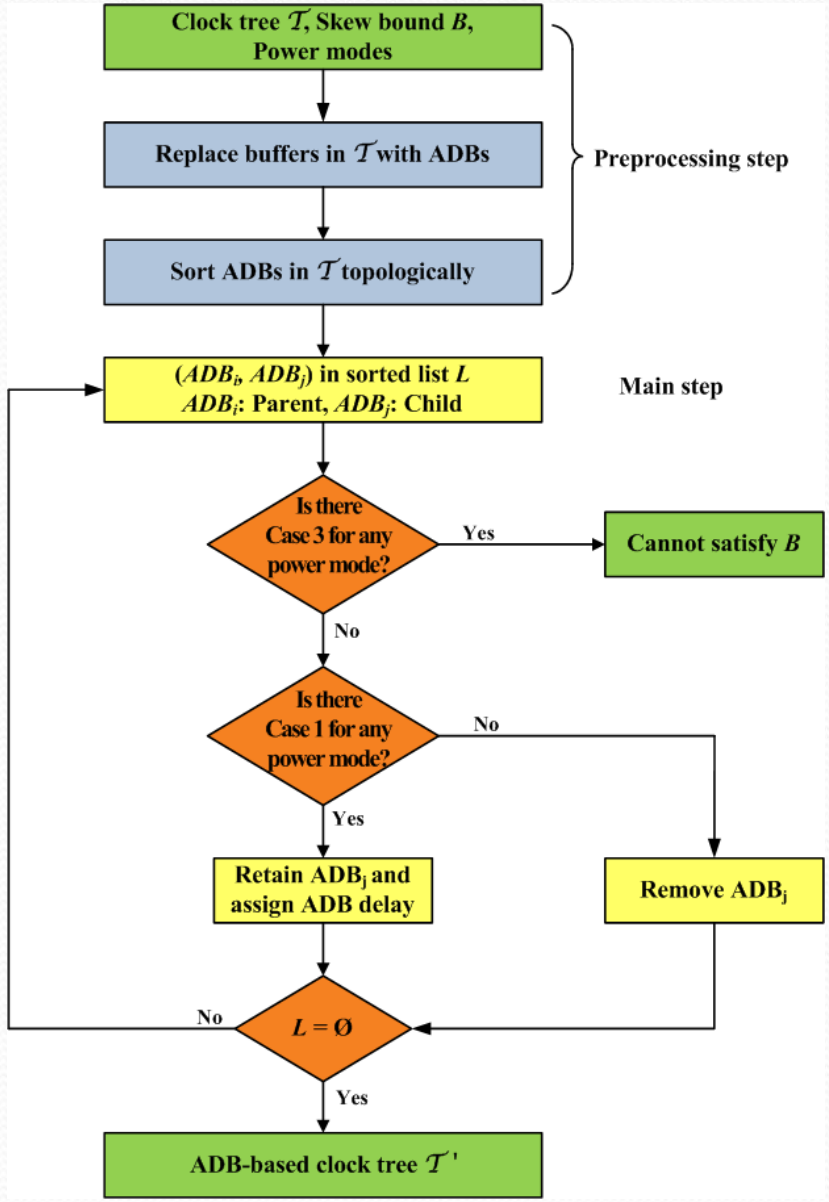
Clock skew constraint = 10

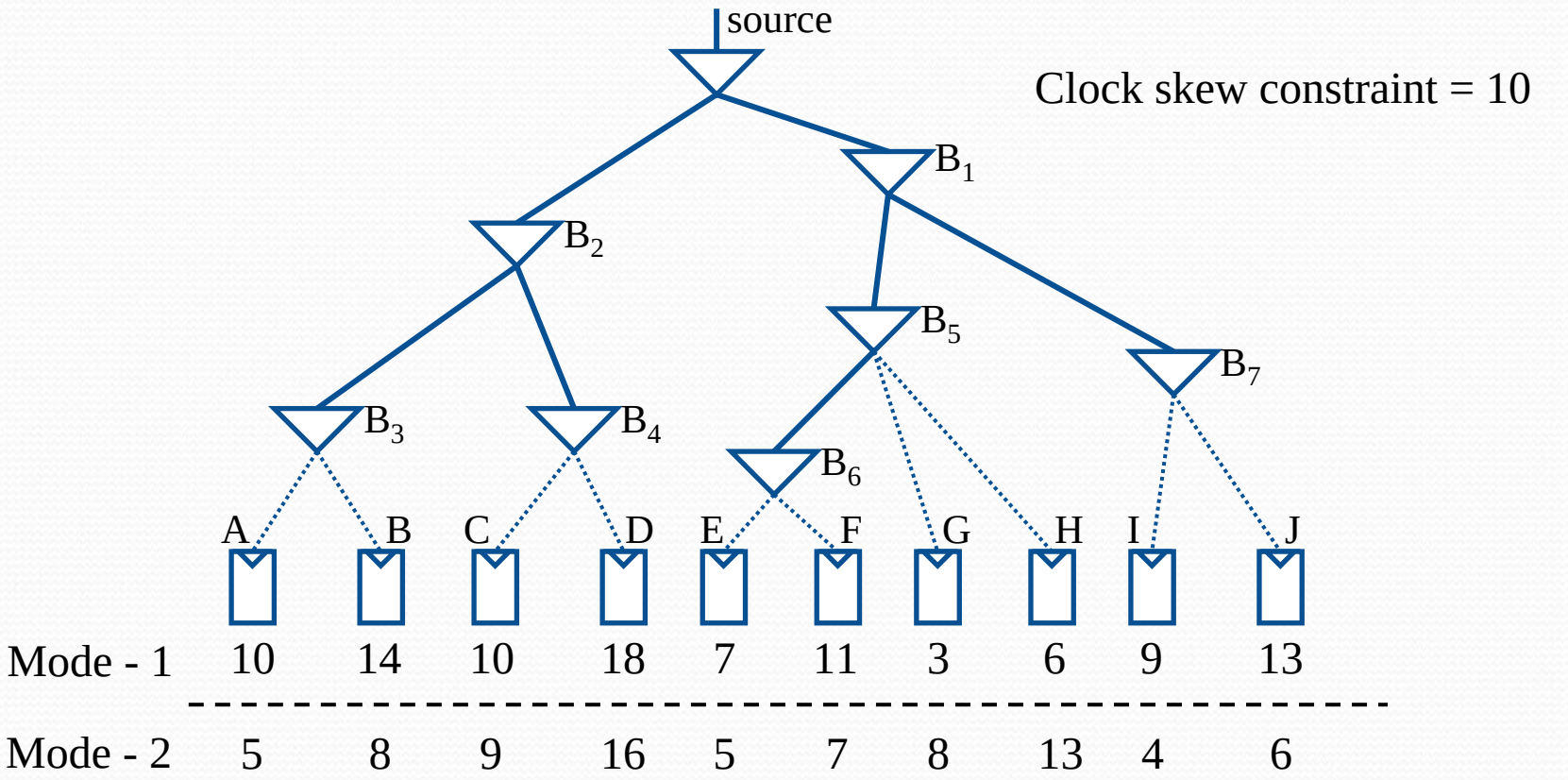


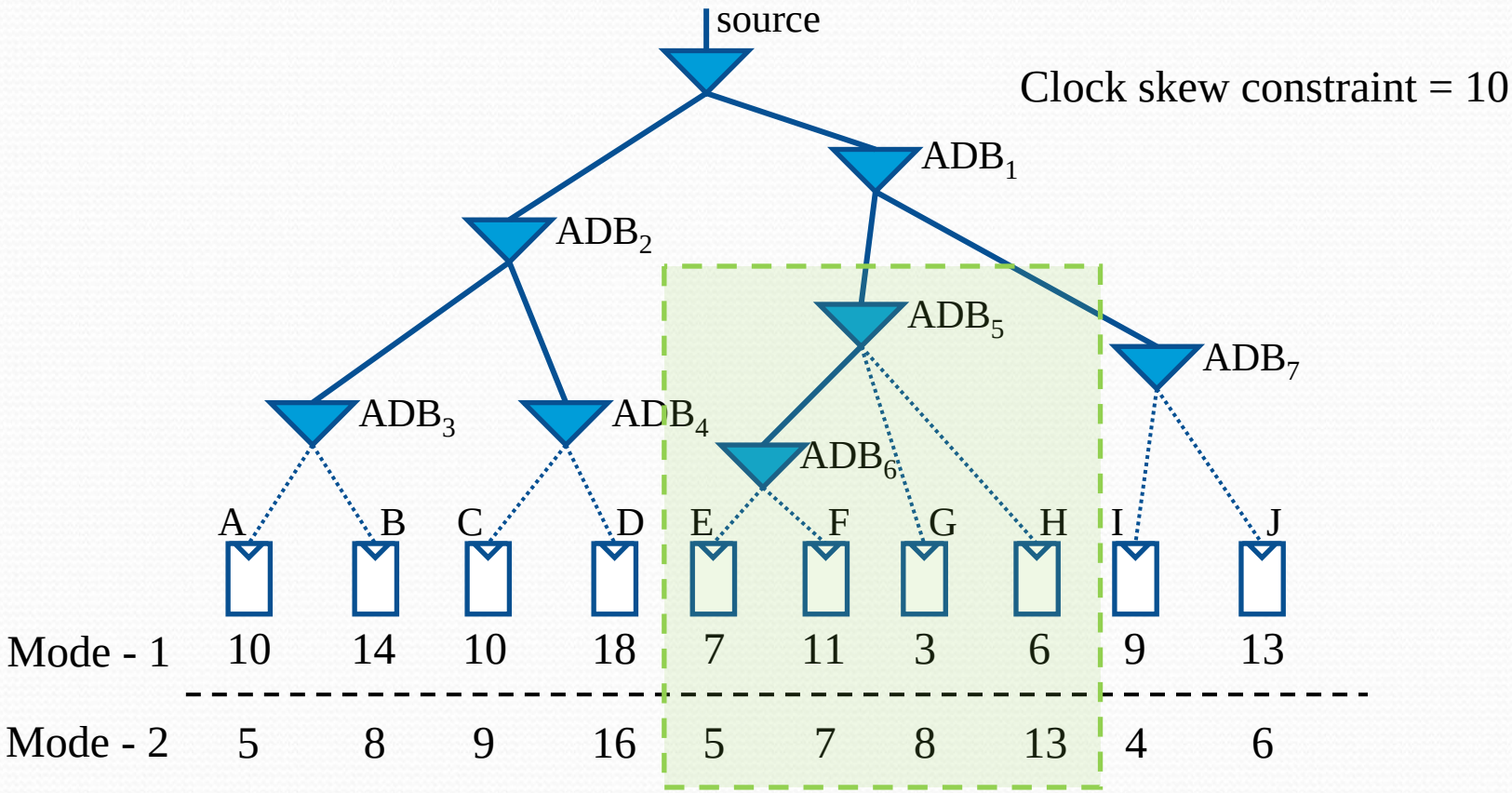
We cannot meet the constraints using current tree! – Case 3

- Summary of all cases
  - Case 1: $EAT_1 \geq LAT_v - B$ and $EAT_2 < LAT_v - B$
    - ADB2 is essential (cannot remove)
    - $\alpha_2 = LAT_v - LAT_2$
  - Case 2: $EAT_1 \geq LAT_v - B$ and $EAT_2 \geq LAT_v - B$
    - ADB2 is useless (remove)
  - Case 3: $EAT_1 < LAT_v - B$
    - Cannot satisfy constraint B
    - Relax the constraints or reconstruct clock tree

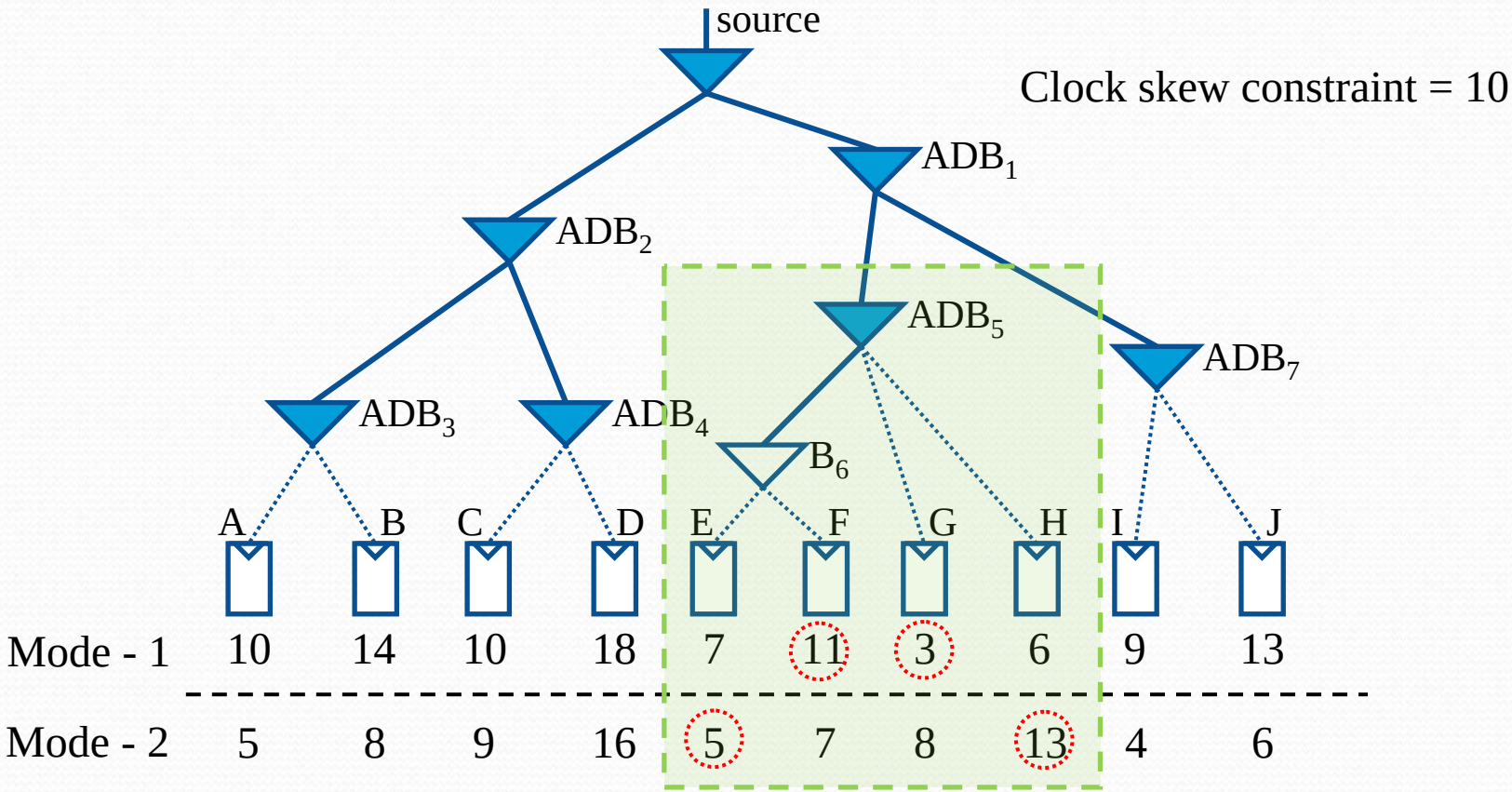
- The flow of algorithm

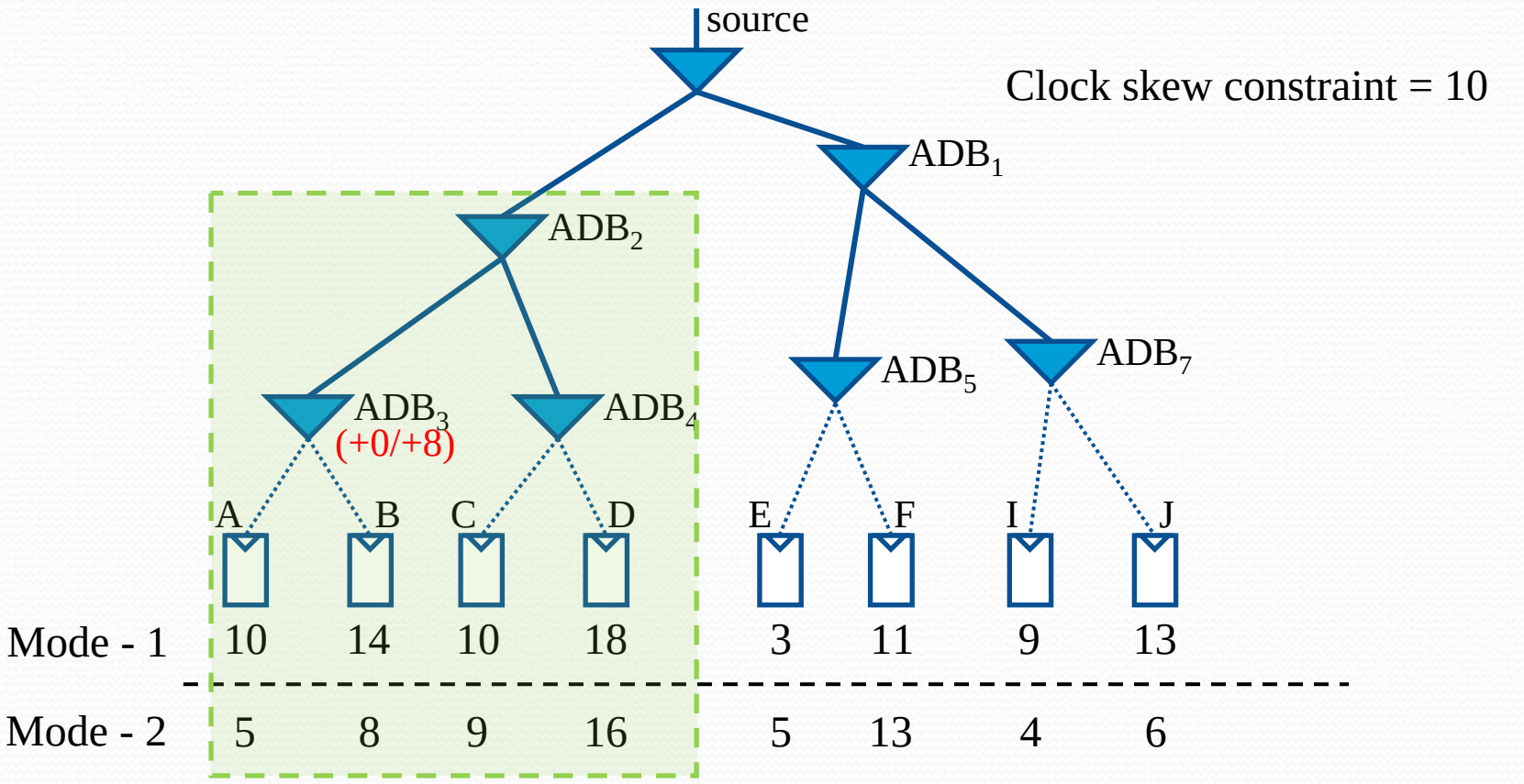



source
Clock skew constraint = 10
B1
B2
B5
B7
B3
B4
B6
A
B
C
D
E
F
G
H
I
J
Mode - 1
10
14
10
18
7
11
3
6
9
13
Mode - 2
5
8
9
16
5
7
8
13
4
6

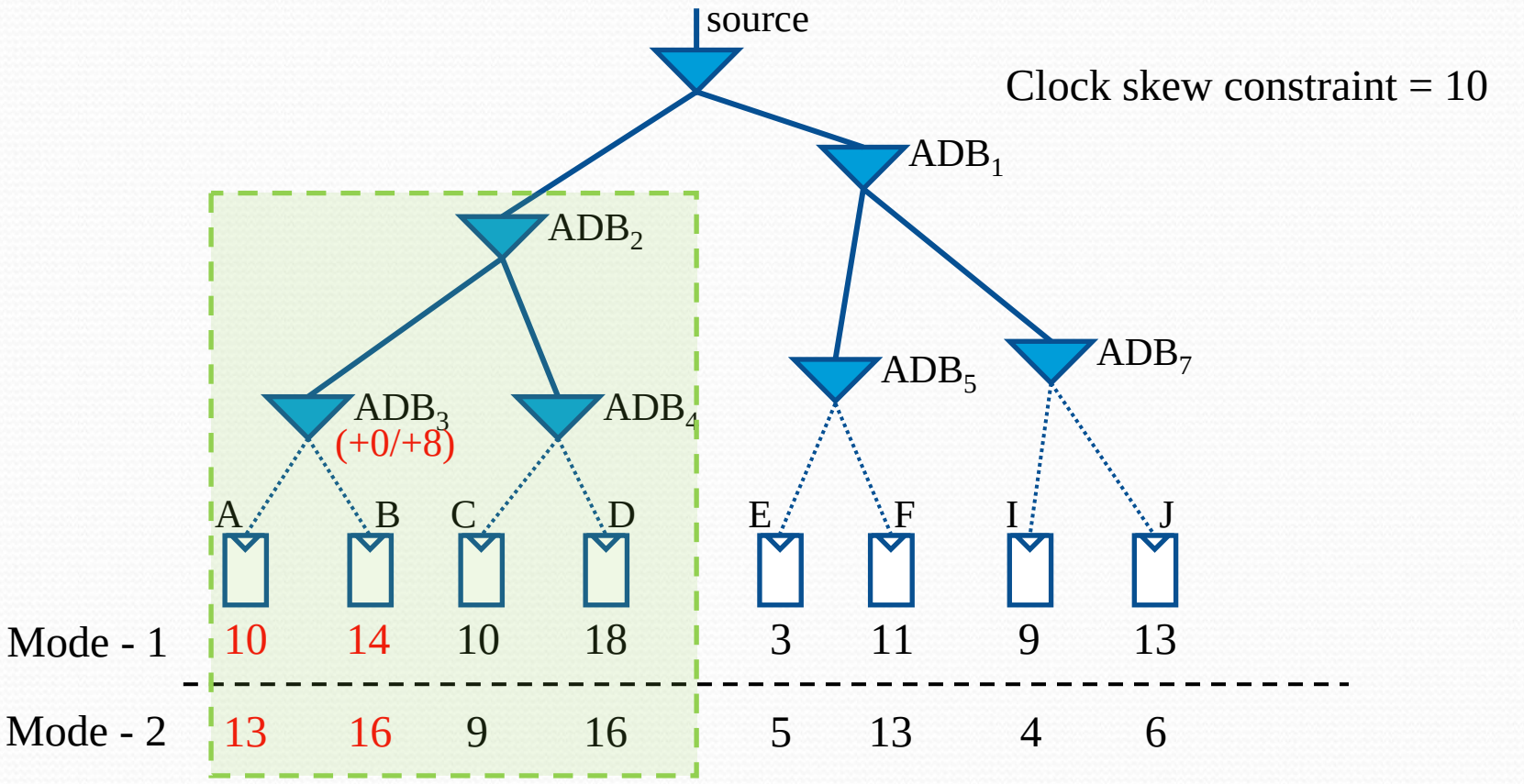


We can remove ADB_6 (Case 2 for all power modes)

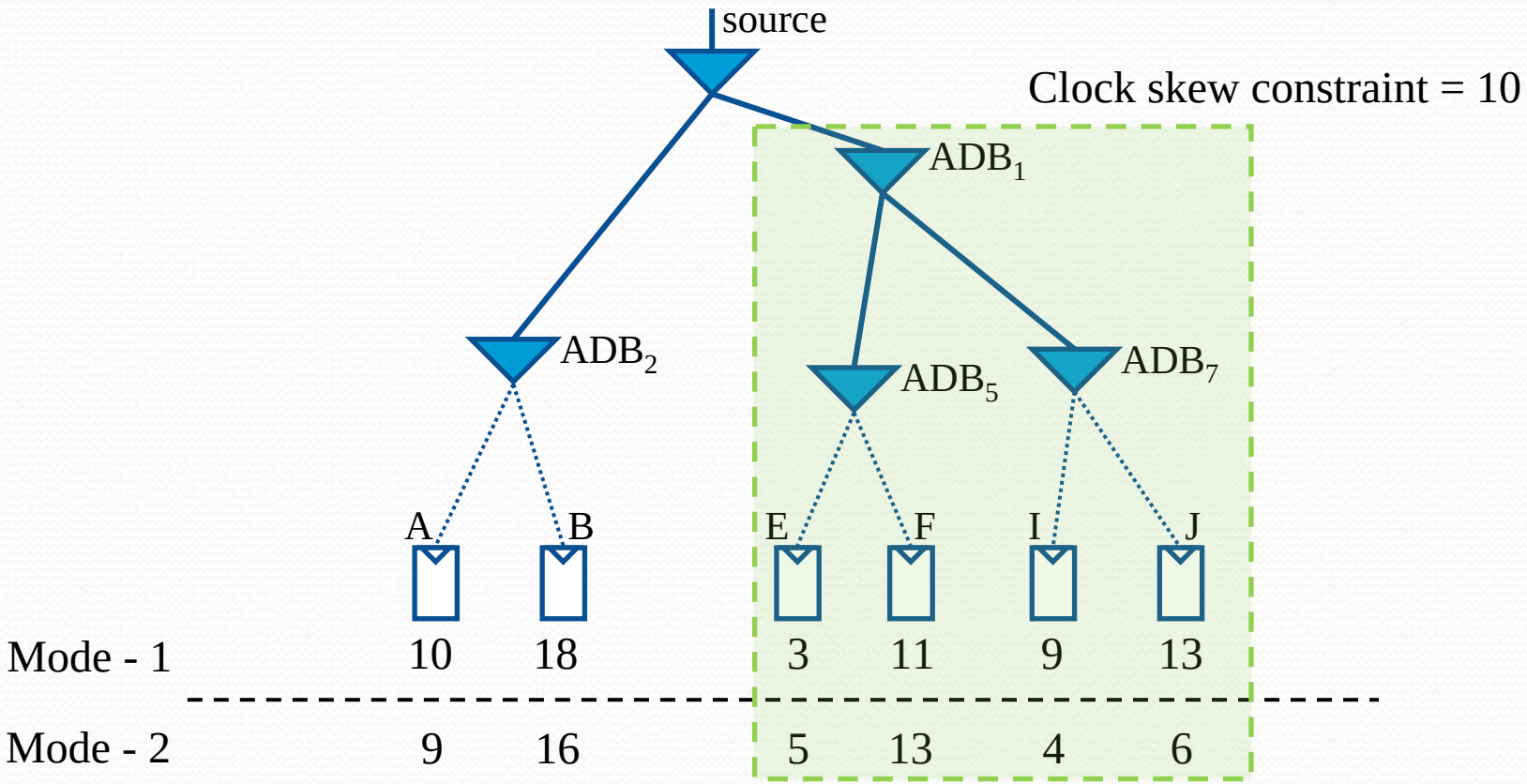

source
Clock skew constraint = 10
ADB1
ADB2
ADB3
ADB4
ADB5
ADB7
B6
A
B
C
D
E
F
G
H
I
J
Mode - 1
10
14
10
18
7
11
3
6
9
13
Mode - 2
5
8
9
16
5
7
8
13
4
6



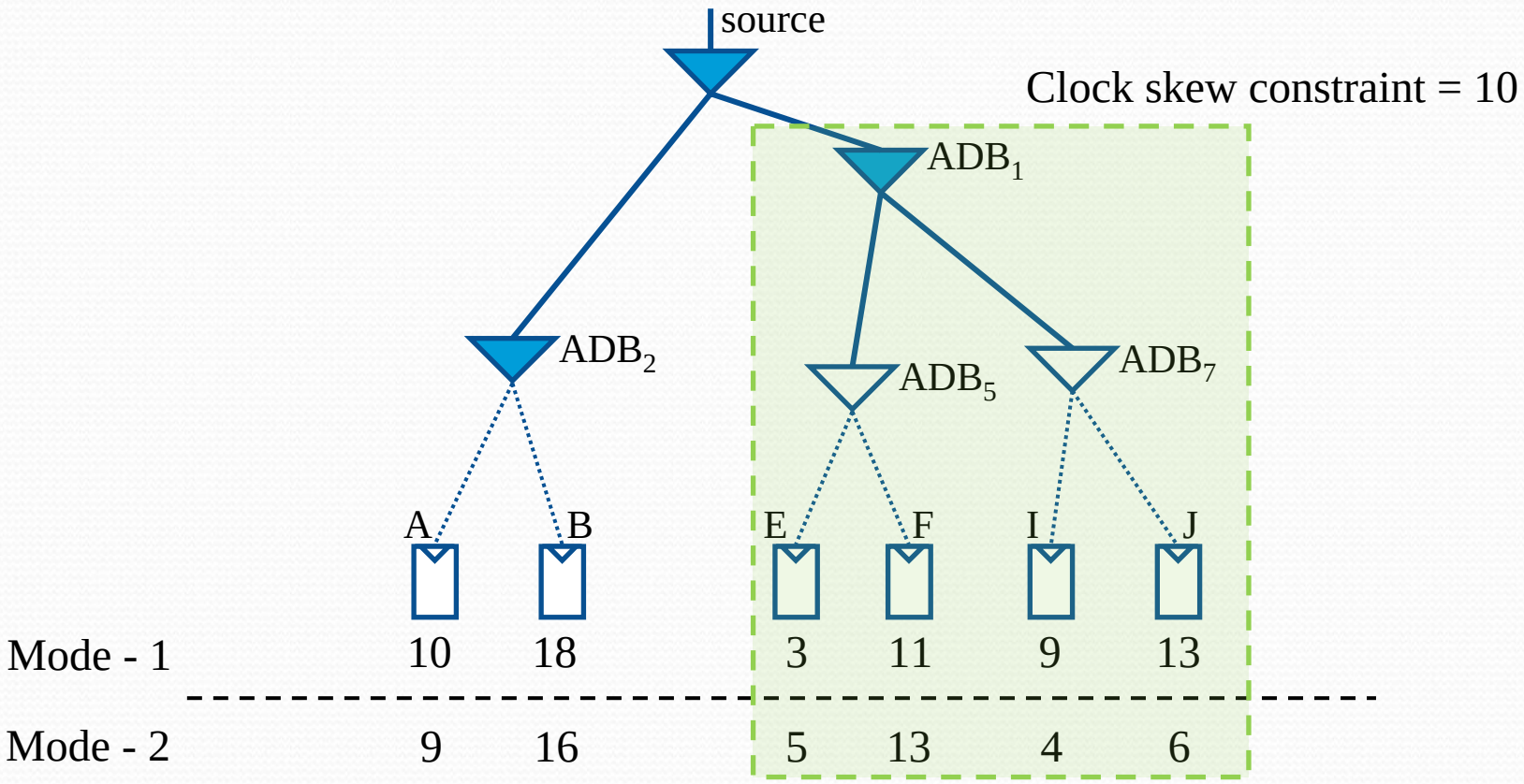
We have to retain ADB_3 (Case 1 at power mode 2)



We have to retain ADB_3 (Case 1 at power mode 2)
We can remove ADB_4 (Case 2 for all power modes)

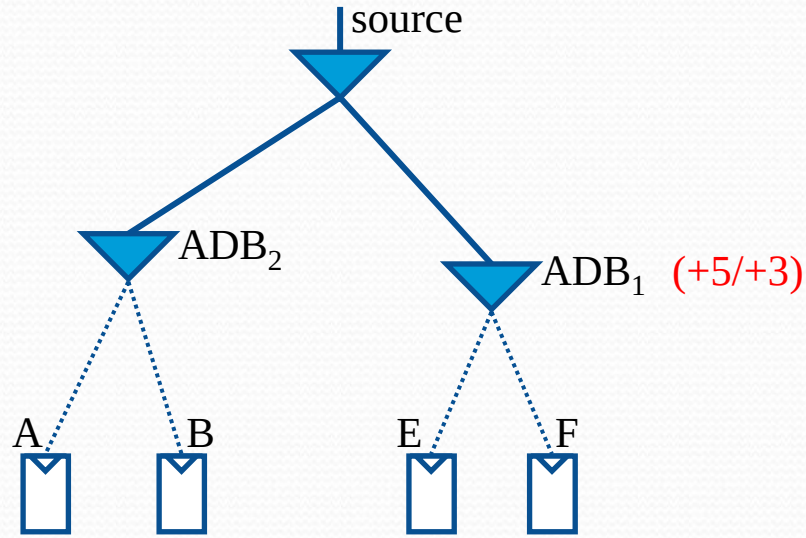


We can remove ADB_5 and ADB_7 (Case 2 for all power modes)



We can remove ADB_5 and ADB_7 (Case 2 for all power modes)

Clock skew constraint = 10



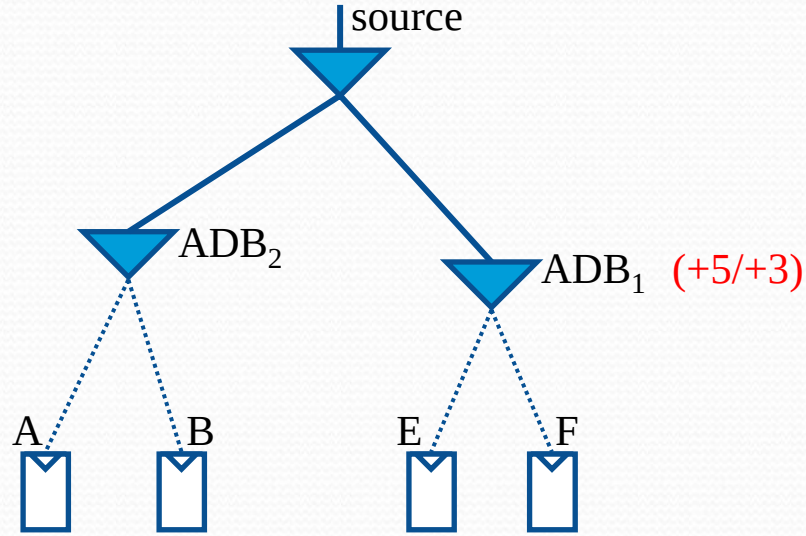
Mode - 1

Mode - 2

Mode - 1	10	18	3	13
Mode - 2	9	16	4	13

We have to retain ADB_1 (Case 1 at power mode 1 and 2)

Clock skew constraint = 10



Mode - 1

Mode - 2

9 16 7 16

We have to retain ADB_1 (Case 1 at power mode 1 and 2)

We can remove ADB_2 (Case 2 for all power modes)

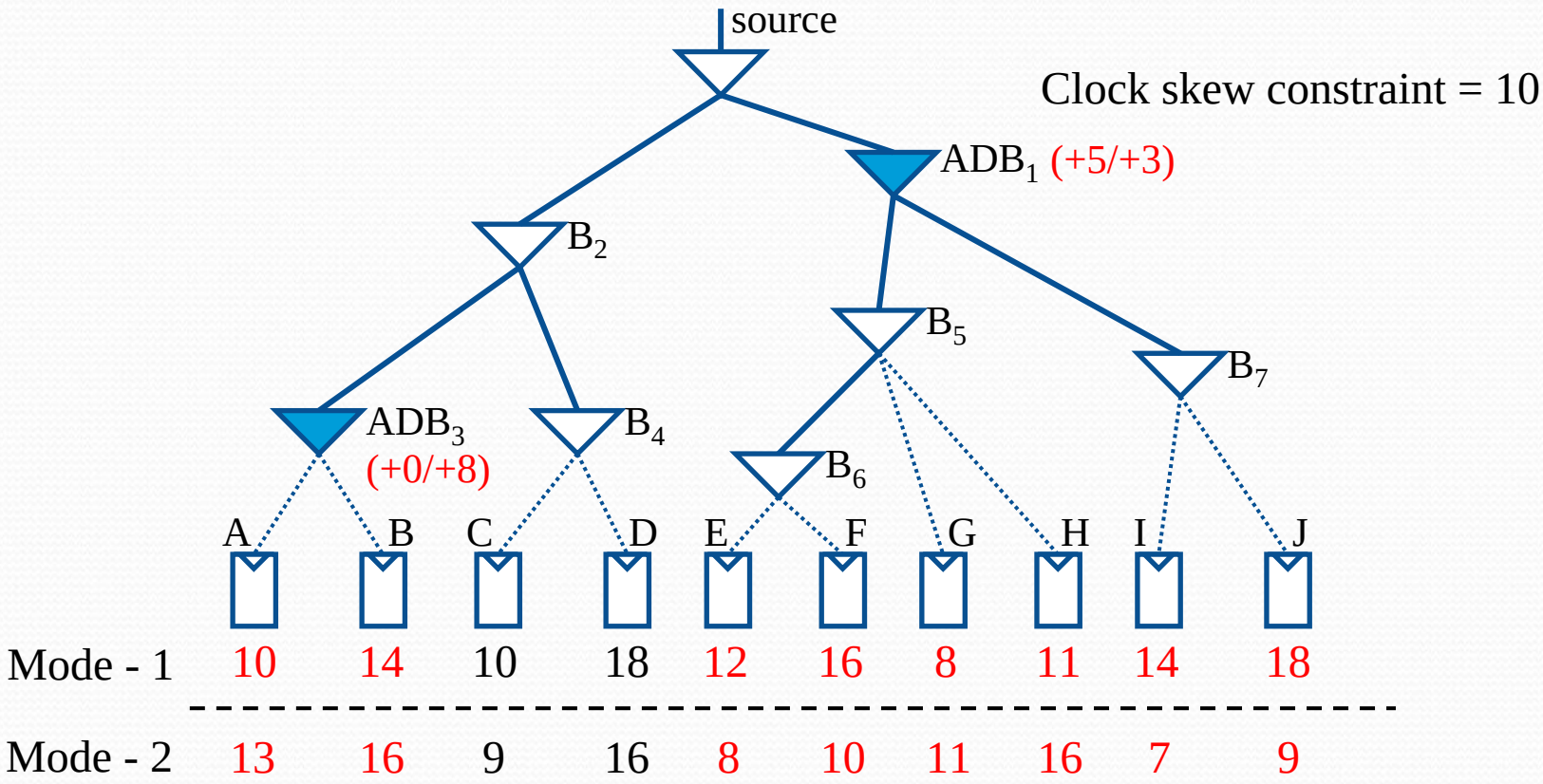

source
Clock skew constraint = 10
ADB1 (+5/+3)
B2
B5
B7
ADB3
(+0/+8)
B4
B6
A
B
C
D
E
F
G
H
I
J
Mode - 1
10
14
10
18
12
16
8
11
14
18
Mode - 2
13
16
9
16
8
10
11
16
7
9

Table of Contents

- Experiment setup
  - System with eight 2.5GHz Intel Xeon CPU and 8GB memory
  - 45nm Nangate Open Cell library
  - Input clock tree is generated by using Synopsys IC compiler
  - Three ISCAS'95 benchmarks and three ITC'99 benchmarks partitioned into 6~10 power subdomains which can operates in two voltage levels – 0.95V and 1.1V
  - Assumed that each ADB can be adjusted with a granularity of 10ps

Bench-marks	#FFs	#Bufs	Worst Skew (ps)	Skew Bound B	#ADBs [1]	#ADBs (CLK-ADB)	Red. of #ADB
s35932	1728	95	281.24	30 ps	60	55	8.33%
				40 ps	38	37	2.63%
				50 ps	20	18	10.0%
s38417	1564	85	292.24	30 ps	47	43	8.51%
				40 ps	29	27	6.70%
				50 ps	20	18	10.0%
s38584	1178	68	285.85	30 ps	44	42	4.54%
				40 ps	39	35	10.26%
				50 ps	27	25	7.41%

[1] Y. –S. Su, W. –K . Hon, C. –C. Yang, S. –C Chang, and Y. –J Chang, "Value assignment of adjustable delay buffers for clock skew minimization in multi-voltage mode designs," *ICCAD*, 2009

Bench-marks	#FFs	#Bufs	Worst Skew (ps)	Skew Bound B	#ADBs [1]	#ADBs (CLK-ADB)	Red. of #ADB
b17	1312	79	286.62	30 ps	26	25	3.94%
				40 ps	18	17	5.56%
				50 ps	15	13	13.3%
b18	2752	176	463.23	30 ps	118	109	7.62%
				40 ps	95	86	9.47%
				50 ps	71	67	5.63%
b22	583	46	326.19	30 ps	14	13	7.14%
				40 ps	12	12	0.00%
				50 ps	8	8	0.00%

[1] Y. –S. Su, W. –K . Hon, C. –C. Yang, S. –C Chang, and Y. –J Chang, "Value assignment of adjustable delay buffers for clock skew minimization in multi-voltage mode designs," *ICCAD*, 2009

Conclusion

- This work provides:
  - Optimal algorithm to solve problem of clock skew optimization using ADBs under multi-voltage design.
  - Finding a solution with minimum number of ADB allocation while satisfying the clock skew constraint for every power mode.
  - 9.27% further decreased number of ADBs compared to the existing algorithm.