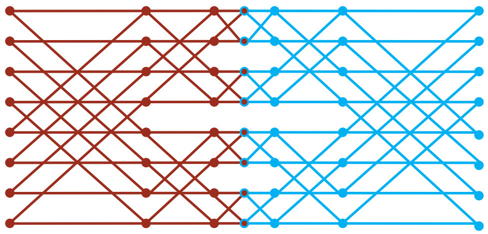
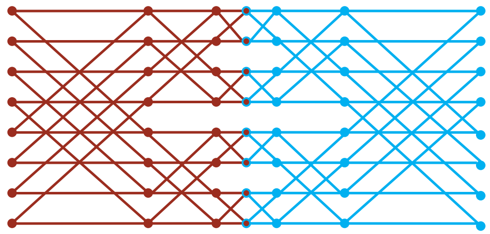
BENEŠ NETWORK (1)

- A possibility to avoid a routing with delays is providing a non blocking topology.
- Beneš network has this property
- It consists of two back-to-back butterflies



BENEŠ NETWORK (2)

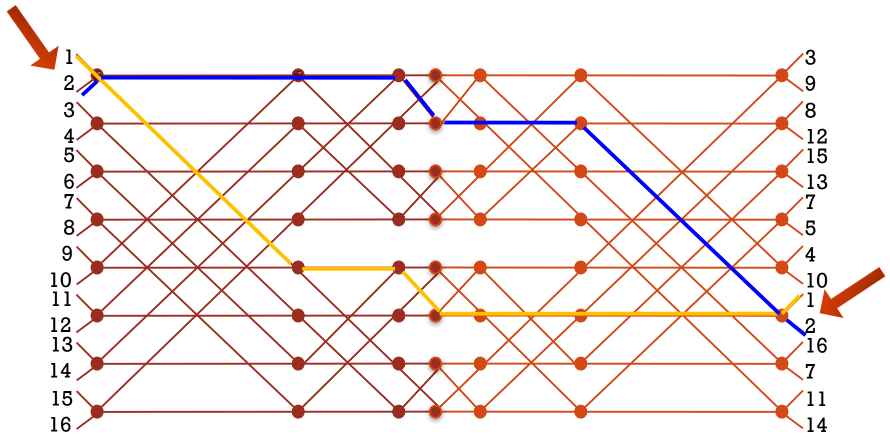
- The n-dimensional Beneš network has $2n+1$ layers, each with 2^n nodes.
- The first and last $n+1$ layers in the network form an n-dimensional Butterfly (the middle layer is shared).
- Not surprisingly, the Beneš network is very similar to the Butterfly, in terms of both its computational power and its network structure.



BENEŠ NETWORK (3)

- The reason for defining the Beneš network is that it is an excellent example of a rearrangeable network.
- **Def.** A network with N inputs and N outputs is said to be rearrangeable if for any one-to-one mapping π of the inputs to the outputs (i.e. for any permutation), we can construct edge-disjoint paths in the network linking the i-th input to the $\pi(i)$-*th* output for $1 \le i \le N$.
- In the case of the n-dimensional Beneš network, we can have *two* inputs for each node at layer 0 and *two* outputs for each node at layer $2n$, and still connect every permutation of inputs to outputs with edge-disjoint paths.
- Hence, in this case, # of inputs=2^{n+1}.

BENEŠ NETWORK (4)



BENEŠ NETWORK (5)

It seems extraordinary that we can find edge-disjoint paths for <u>any</u> permutation. Nevertheless, the result is true, and it is even fairly easy to prove, as we show in the following:

Th. *Given any one-to-one mapping π of 2^{n+1} inputs to 2^{n+1} outputs on an n-dimensional Beneš network, there is a set of edge-disjoint paths from the inputs to the outputs connecting input i to output $\pi(i)$ for $1 \le i \le 2^{n+1}$.*

Proof. …

BENEŠ NETWORK (6)

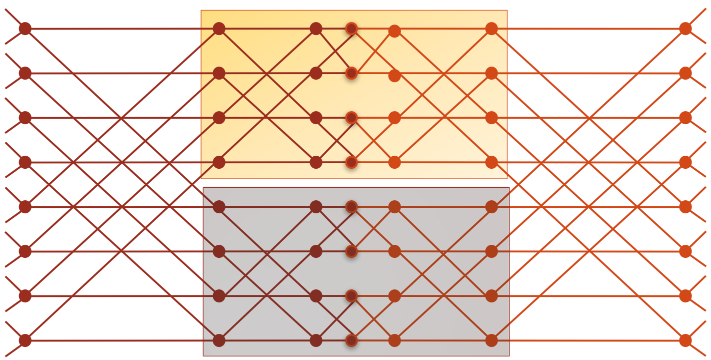
PROOF OF THE REARRANGEABILITY OF THE BENEŠ NETWORK (CNTD)

Proof. By induction on *n*.

- <u>Basis</u>: if $n=0$, the Beneš network consists of a single node (i.e. a single 2x2 switch) and the result is obvious.
- <u>Induction</u>: assume that the result is true for an $(n-1)$-dimensional Beneš network
- Key observation: the middle $2n-1$ layers of an n-dimensional Beneš network comprise two $(n-1)$-dimensional Beneš networks
- …

BENEŠ NETWORK (7)

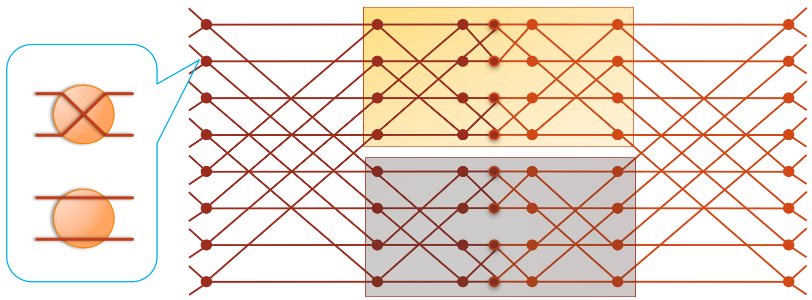
PROOF OF THE REARRANGEABILITY OF THE BENEŠ NETWORK (CNTD)



BENEŠ NETWORK (8)

PROOF OF THE REARRANGEABILITY OF THE BENEŠ NETWORK (CNTD)

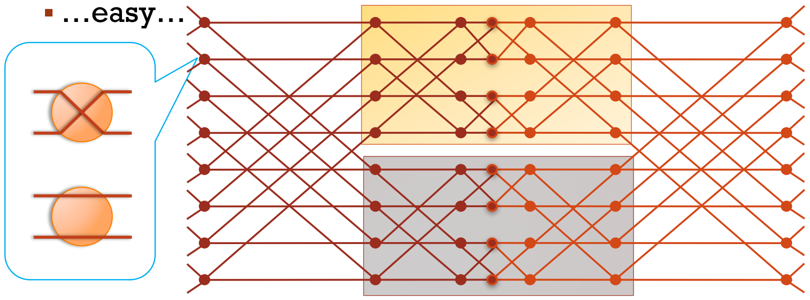
- Hence, for each path, it will be sufficient to decide whether it is to be routed through the upper sub-Beneš network or through the lower sub-Beneš network.



BENEŠ NETWORK (9)

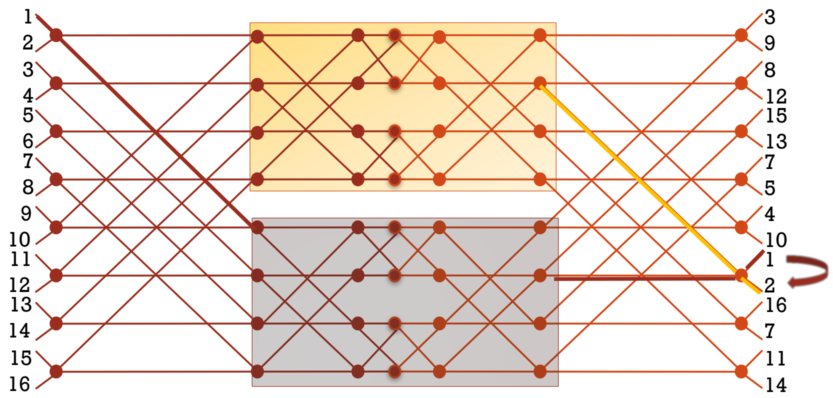
PROOF OF THE REARRANGEABILITY OF THE BENEŠ NETWORK (CNTD)

- The only constraints we have to consider to decide whether paths use the upper or lower subnetworks are that paths from inputs *2i-1* and *2i* must use different subnetworks for $1 \le i \le 2n$, and that paths to outputs *2i-1* and *2i* must use different sub-networks.
- …easy…



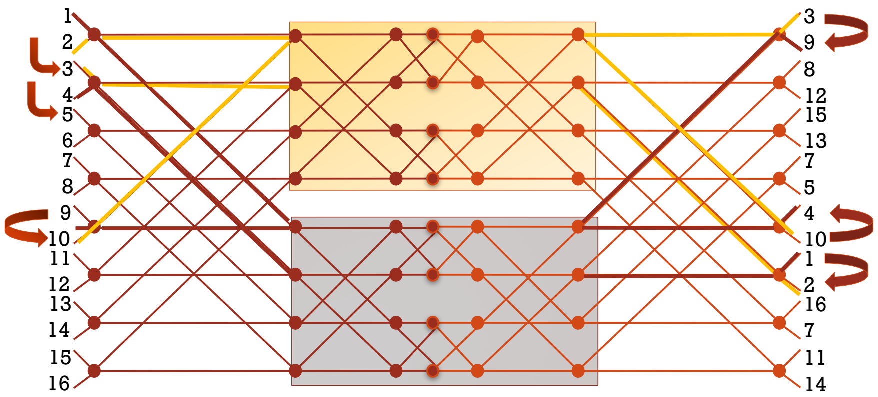
BENEŠ NETWORK (10)

PROOF OF THE REARRANGEABILITY OF THE BENEŠ NETWORK (CNTD)



BENEŠ NETWORK (11)

PROOF OF THE REARRANGEABILITY OF THE BENEŠ NETWORK (CNTD)



And so on…

BENEŠ NETWORK (12)

PROOF OF THE REARRANGEABILITY OF THE BENEŠ NETWORK (CNTD)

Summary of the steps:

- We start by routing the first path through the upper sub-network.
- We next satisfy the constraint generated at the output by routing the corresponding path through the lower sub-network.
- We keep on going back and forth through the network, satisfying constraints at the inputs by routing through the upper sub-network and satisfying constraints at the outputs by routing through the lower sub-network.
- …

BENEŠ NETWORK (13)

PROOF OF THE REARRANGEABILITY OF THE BENEŠ NETWORK (CNTD)

- …
- Eventually, we will close the loop by routing a path through the lower sub-network (in response to an output constraint) that shares an input switch with the first path that was routed.
- If any additional paths needs to be routed, we con- tinue as before, starting over again with an arbitrary unrouted path.
- In this way, all paths can be assigned to the upper or lower sub-networks without conflict.

BENEŠ NETWORK (14)

PROOF OF THE REARRANGEABILITY OF THE BENEŠ NETWORK (CNTD)

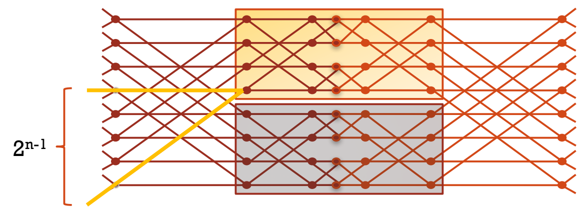
- This algorithm is called looping algorithm.
- It is easy to see that all paths can be assigned to the upper or lower sub-networks without conflict:
- By construction, if we start going to the upper sub-network, we will arrive to the corresponding output in the upper sub-network and we will leave it to the lower sub-network, and so on.
- For parity reason, when a loop is close, we will correctly arrive from the right sub-network.
- The remainder of the path routing and switch setting is handled by induction in the sub-networks. ■

BENEŠ NETWORK (15)

- In the case that each layer 0 node of the n-dimensional Beneš network has just <u>one input</u> and each layer $2n$ node has just <u>one output</u>, then the paths from the inputs to the outputs can be constructed so as to be <u>node-disjoint</u> (instead of only edge-disjoint):
- …

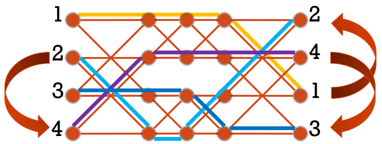
BENEŠ NETWORK (16)

- **Th.** *Given any one-to-one mapping of π of 2^n inputs to 2^n outputs in an n-dim. Beneš network, there is a set of node-disjoint paths from the inputs to the outputs connecting input i to output $\pi(i)$ for $1 \leq i \leq 2^n$.*
- Proof. Identical to the previous one, but the paths needing to use different Beneš networks are now i and $i+2^{n-1}$, $1 \leq i \leq 2^{n-1}$ (and not $2i-1$ and $2i$). ■



BENEŠ NETWORK (17)

- Exemple: n=2, hence 2^{n-1}=2



BENEŠ NETWORK (18)

Drawbacks of the looping algorithms (both versions):

- we do not know how to set the switches on-line. In other words, each switch needs to be told what to do by a **global control** that has knowledge of the permutation being routed
  - There exist numerous methods for overcoming this difficulty (not studied here).
- every time a new permutation must be routed, $\Theta(N \log N)$ time is necessary to re-set switches.