Physical Synthesis of Bus Matrix for High Bandwidth Low Power On-chip Communications

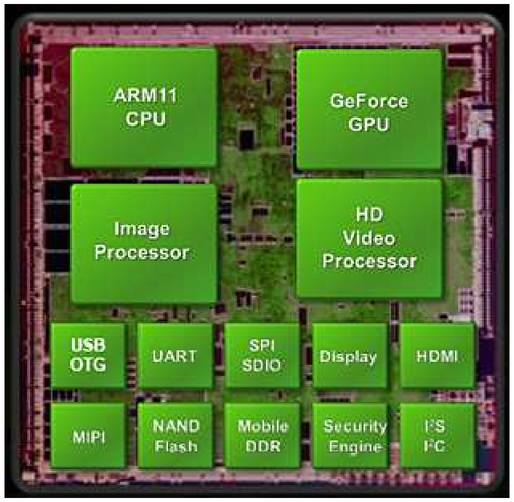
Renshen Wang[1], Evangeline Young[2], Ronald Graham[1] and Chung-Kuan Cheng[1]

[1]University of California San Diego
[2]The Chinese University of Hong Kong

Outline of This Talk

- Trends of on-chip communications
  - Bandwidth requirement ↗
    - Bus → bus matrix, network-on-chip
  - Power consumption ↗
    - Low power design techniques
- Optimizations and tradeoffs in physical synthesis of bus matrix
  - Bus gating on Steiner graph (power)
  - Weighted Steiner graph (bandwidth)
  - Edge merging heuristic (wire length)

Introduction



NVIDIA Tegra chip

- Importance of low power
  - Heat removal, battery life, performance, electricity, envioronment…
- SoC communication power increasing
  - Advances in manufacturing process → more components (n) → higher throughput ($n^{1.xx?}$)
  - Long wires (global on-chip interconnect) relatively scaling up on power
- Goal: power efficiency on data throughput
  - Simple bus → power efficient bus

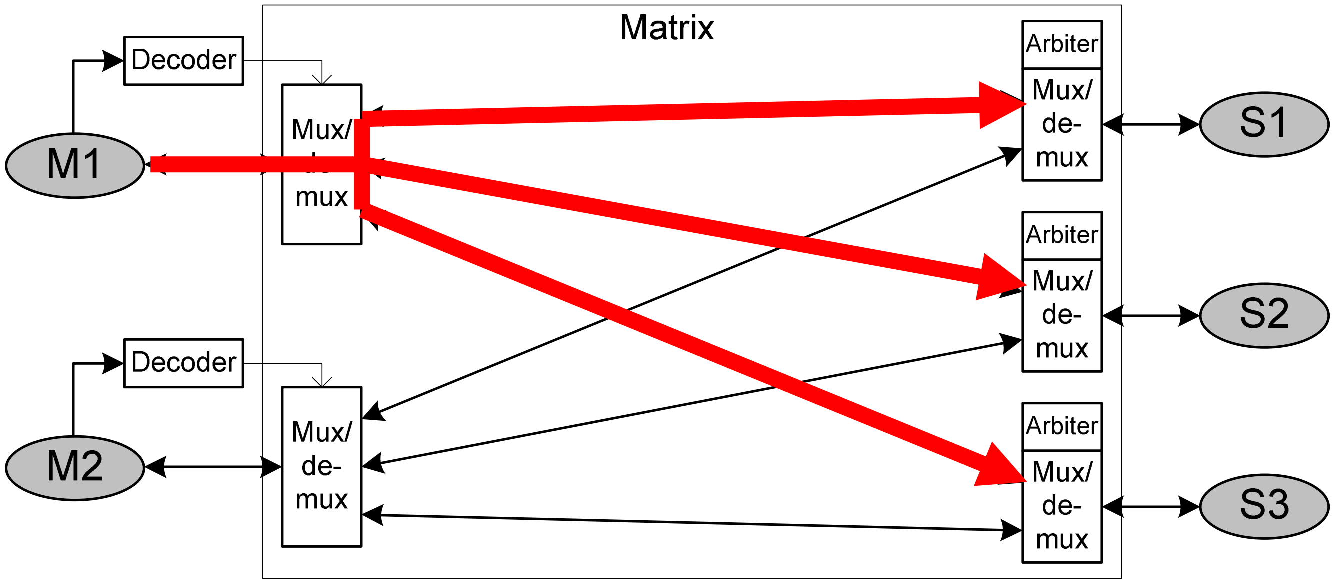
Bus vs. NoC

- Bus / Bus matrix and Network-on-chip comparisons

	Bus	NoC
Power	Bus gating ☹ → ☺	Packet, routing 😐
Latency	☺	☹
Bandwidth	Bus matrix ☹ → 😐	Flexibility ☺

Bus Matrix Overview

- Buses allowing multiple transactions
  - AMBA AHB/AXI protocols, *etc*
  - Example: a full (high bandwidth) bus matrix
    - Power efficient, but not wire efficient



Problem Formulations

- Communication constraint graph
  - Bipartite graph $G = (U, W, A)$
  - U : set of masters
  - W : set of slaves
  - A: set of arcs, arc (u, w) means u accesses w
- Given a placement and a communication constraint graph G, find a bus matrix with
  - Bandwidth capability for G
    - Each component can have at most 1 connection at a time
  - Minimal power on data (path length)
  - Minimal wires

Ideal Bus Matrix

- Definition 1: Given $G = (U, W, A)$ and placement function $P : U \cup W \rightarrow R^2$, an ideal bus matrix graph is a weighted graph $\Theta = (V, E, \omega)$ that
  - $U \subseteq V$
  - $W \subseteq V$
  - For any $A' \subseteq A$ such that
    $\underline{\forall\ (u_i, w_i), (u_j, w_j) \in A',\ i \neq j \Rightarrow u_i \neq u_j \text{ and } w_i \neq w_j,}$
    there is a set of paths $\Pi' \subseteq \Pi$, such that
    - $|\Pi'| = |A'|$
    - $\forall\ r \in \Pi',\ r \subseteq V \bigcup E$
    - $\forall\ (u, v) \in A',\ \exists\ r \in \Pi'$ such that $u \in r,\ v \in r$, and
      $\underline{\sum_{(i,j) \in r} \|P(i) - P(j)\|_1 = \|P(u) - P(v)\|_1}$
    - $\forall\ e \in E,\ |\{r \in \Pi' : e \in r\}| \leq \omega(e)$

Computationally expensive

No common vertex

Path is shortest

- Minimize $L(\Theta) = \sum_{(u,v) \in E} \omega((u, v)) \|P(u) - P(v)\|_1$

Practical Formulation

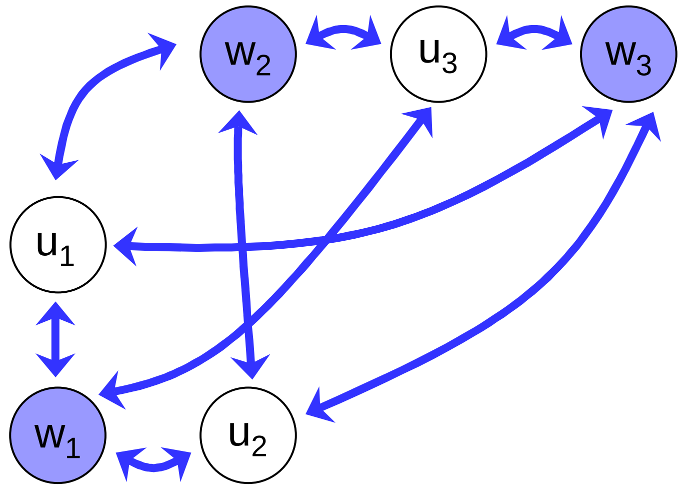
- Definition 2: Given $G = (U, W, A)$ and placement function $P : U \cup W \rightarrow R^2$, a bus matrix graph is a weighted graph $H = (V, E, \omega)$ with a set of paths $\rho : A \rightarrow \Pi$ that
  - $U \subseteq V$
  - $W \subseteq V$
  - $\forall\ a \in A,\ \rho(a) \subseteq V \bigcup E$
  - $\forall\ (u, v) \in A,$ $\underline{\sum_{(i,j)\in\rho(a)} \|P(i) - P(j)\|_1 = \|P(u) - P(v)\|_1}$ Path is shortest
  - For any $A' \subseteq A$ such that $\underline{\forall\ (u_i, w_i), (u_j, w_j) \in A',\ i \neq j \Rightarrow u_i \neq u_j \text{ and } w_i \neq w_j,}$ No common vertex we have $\forall\ e \in E,\ |\{a \in A' : e \in \rho(a)\}| \leq \omega(e)$

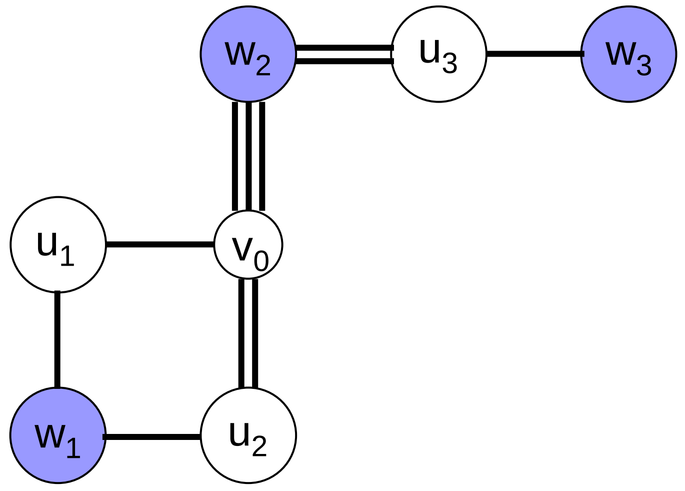
With fixed paths, no real-time computation needed

- Minimize $L(H) = \sum_{(u,v)\in E} \omega((u, v))\|P(u) - P(v)\|_1$

Constructing a Solution

- Communication & placement are given
  - Number of paths fixed
  - Path length fixed (Manhattan distance)

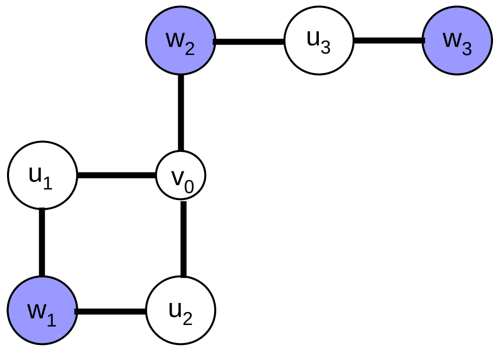


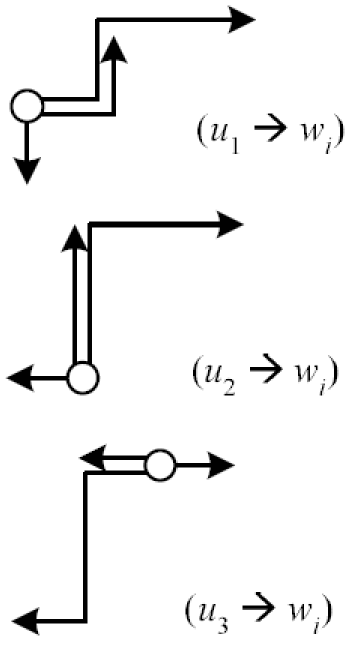


- Generate a structure for min wire length

Graph Construction Algorithm

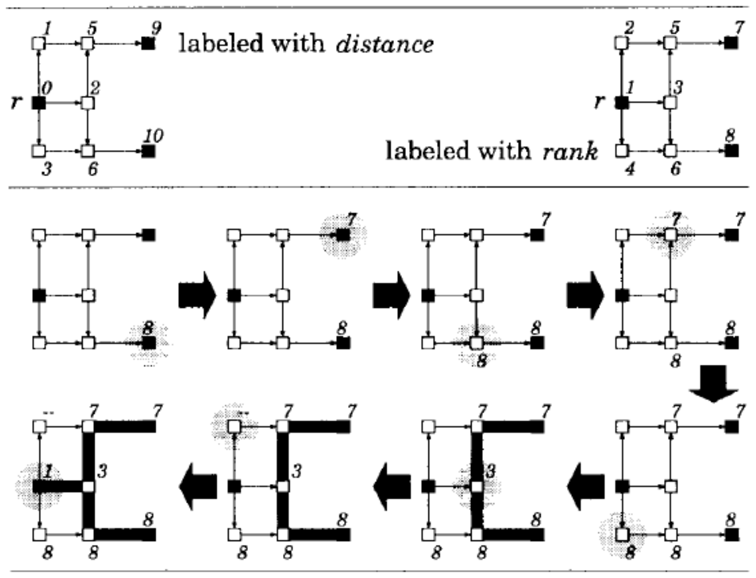
- 1. Generate a shortest-path Steiner graph
  - Algorithm from "*Low Power Gated Bus Synthesis using Shortest-Path Steiner Graph for System-on-Chip Communications*" DAC 2009
- 2. Pick a shortest path for each arc (u_i, w_j) in *A*
  - Randomly pick one if multiple shortest paths exist, to distribute the "load" evenly on graph edges
- 3. Compute edge weight for each edge in the Steiner graph





Minimum Rectilinear Steiner Arborescence (MRSA)

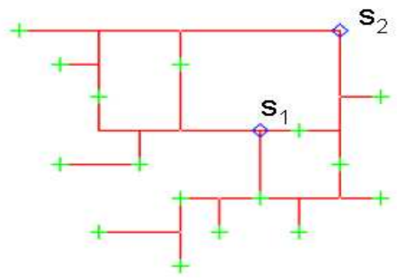
- Steiner tree w/ shortest root-to-leaf paths
  - Constructed by ***merging*** sub-trees with the furthest merging point from the root

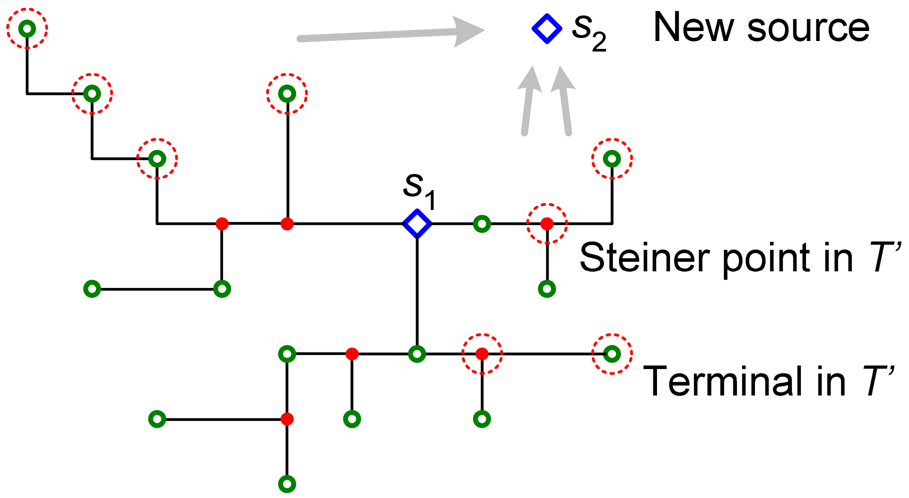


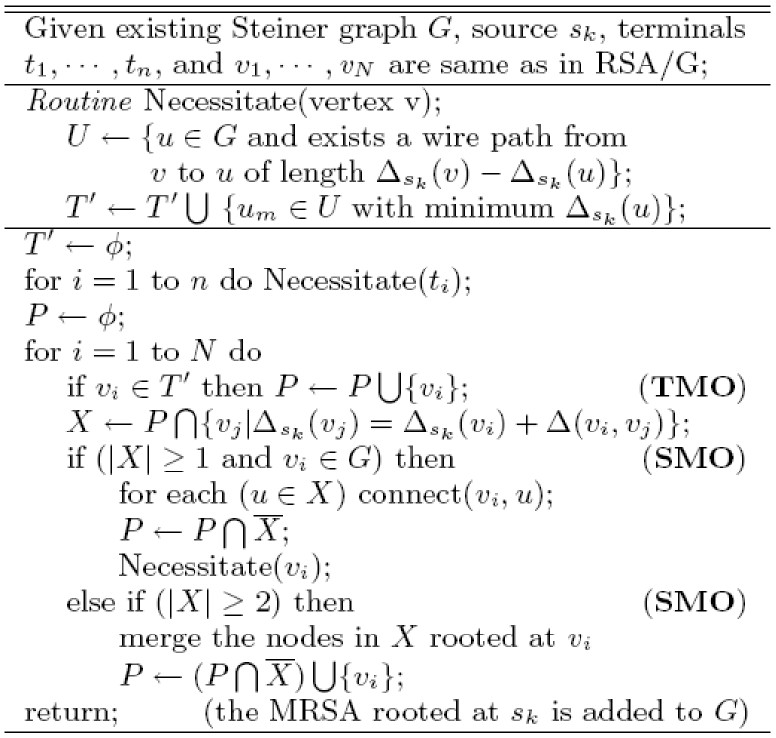
  - "*Efficient algorithms for the minimum shortest path Steiner arborescence problem…*" by Cong, Kahng & Leung. *IEEE TCAD* 1998

Shortest-path Steiner Graph

- Multiple MRSA constructions
  - Each master device as a root
  - 1st MRSA
  - From the 2nd MRSA, wires can be shared







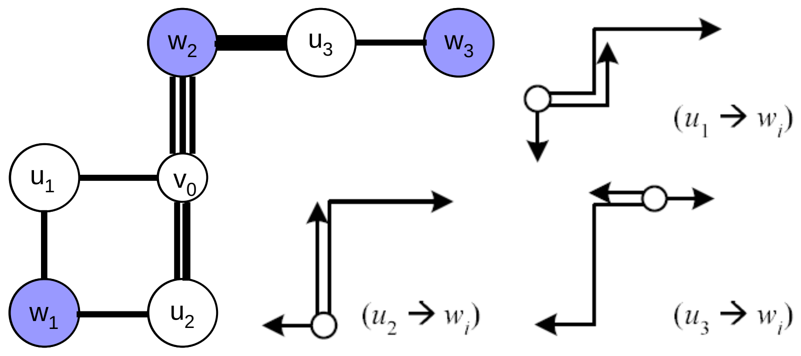
```
Given existing Steiner graph $G$, source $s_k$, terminals
$t_1, \cdots, t_n$, and $v_1, \cdots, v_N$ are same as in RSA/G;
```

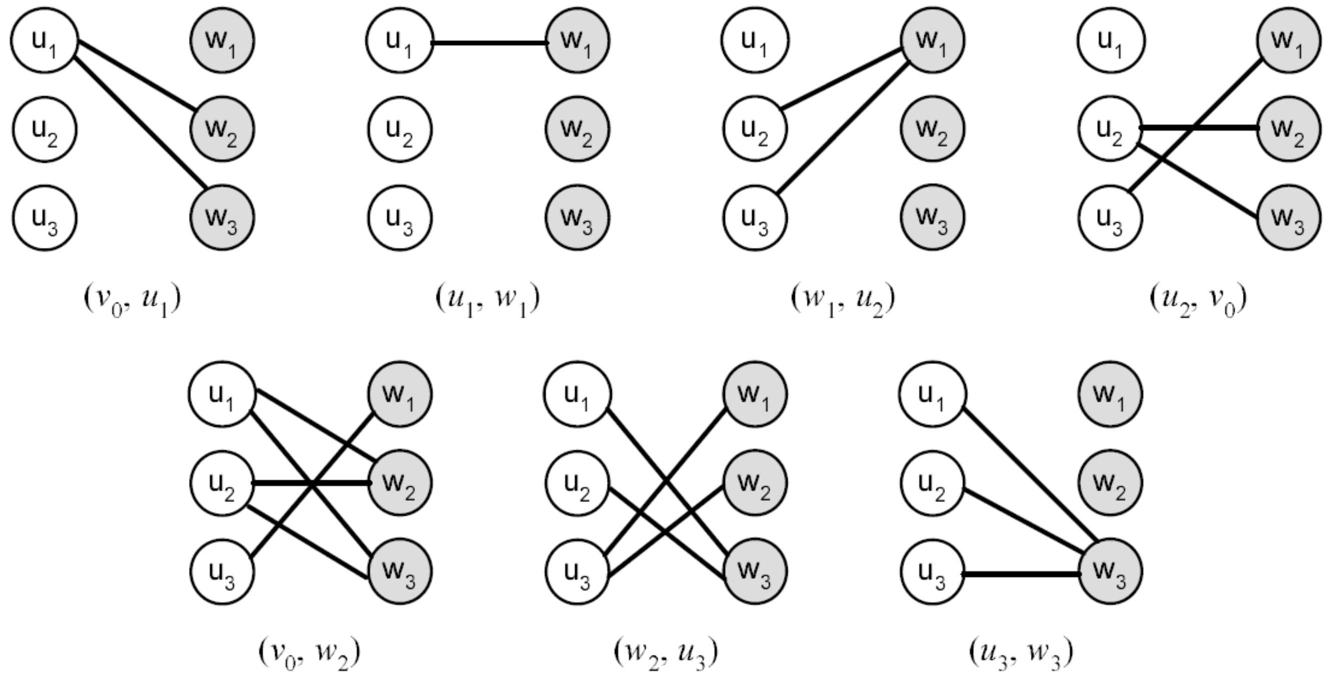
```
Routine Necessitate(vertex v);
    $U \leftarrow \{u \in G$ and exists a wire path from
            $v$ to $u$ of length $\Delta_{s_k}(v) - \Delta_{s_k}(u)\}$;
    $T' \leftarrow T' \bigcup \{u_m \in U$ with minimum $\Delta_{s_k}(u)\}$;
```

```
$T' \leftarrow \phi$;
for $i = 1$ to $n$ do Necessitate($t_i$);
$P \leftarrow \phi$;
for $i = 1$ to $N$ do
    if $v_i \in T'$ then $P \leftarrow P \bigcup \{v_i\}$;                    (TMO)
    $X \leftarrow P \bigcap \{v_j | \Delta_{s_k}(v_j) = \Delta_{s_k}(v_i) + \Delta(v_i, v_j)\}$;
    if ($|X| \geq 1$ and $v_i \in G$) then                                   (SMO)
        for each ($u \in X$) connect($v_i, u$);
        $P \leftarrow P \bigcap \overline{X}$;
        Necessitate($v_i$);
    else if ($|X| \geq 2$) then                                               (SMO)
        merge the nodes in $X$ rooted at $v_i$
        $P \leftarrow (P \bigcap \overline{X}) \bigcup \{v_i\}$;
return;          (the MRSA rooted at $s_k$ is added to $G$)
```

Edge Weight by Max-Matching

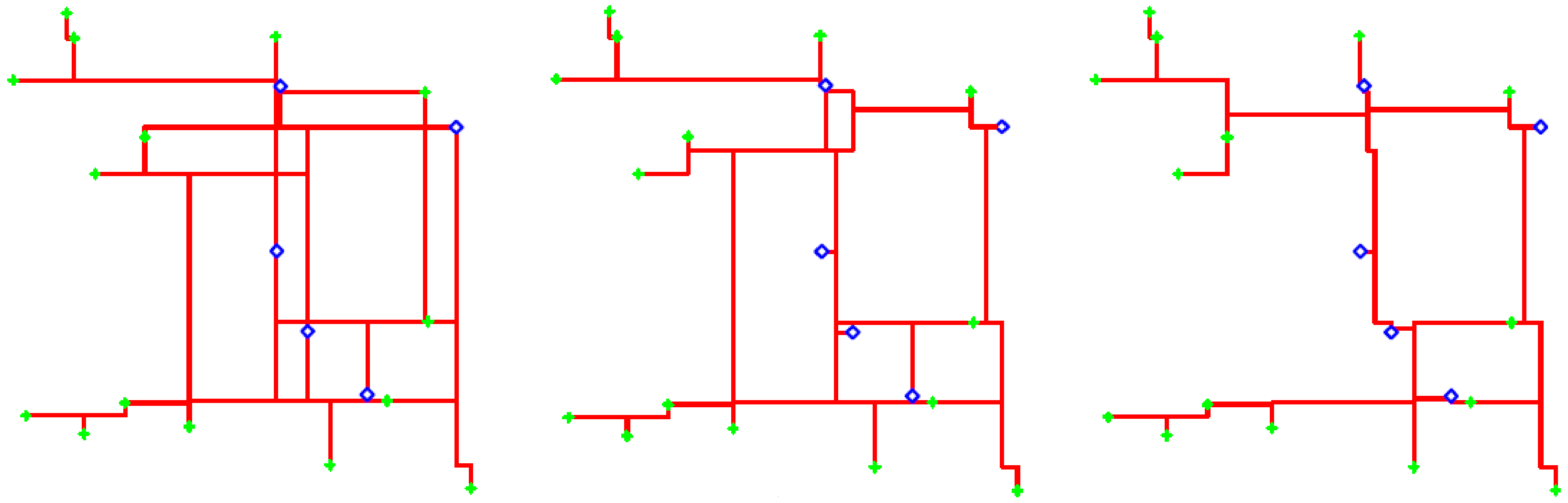
- To allow multiple transactions/paths, add edge weight (multiple bus lines)





Reducing Wire Length

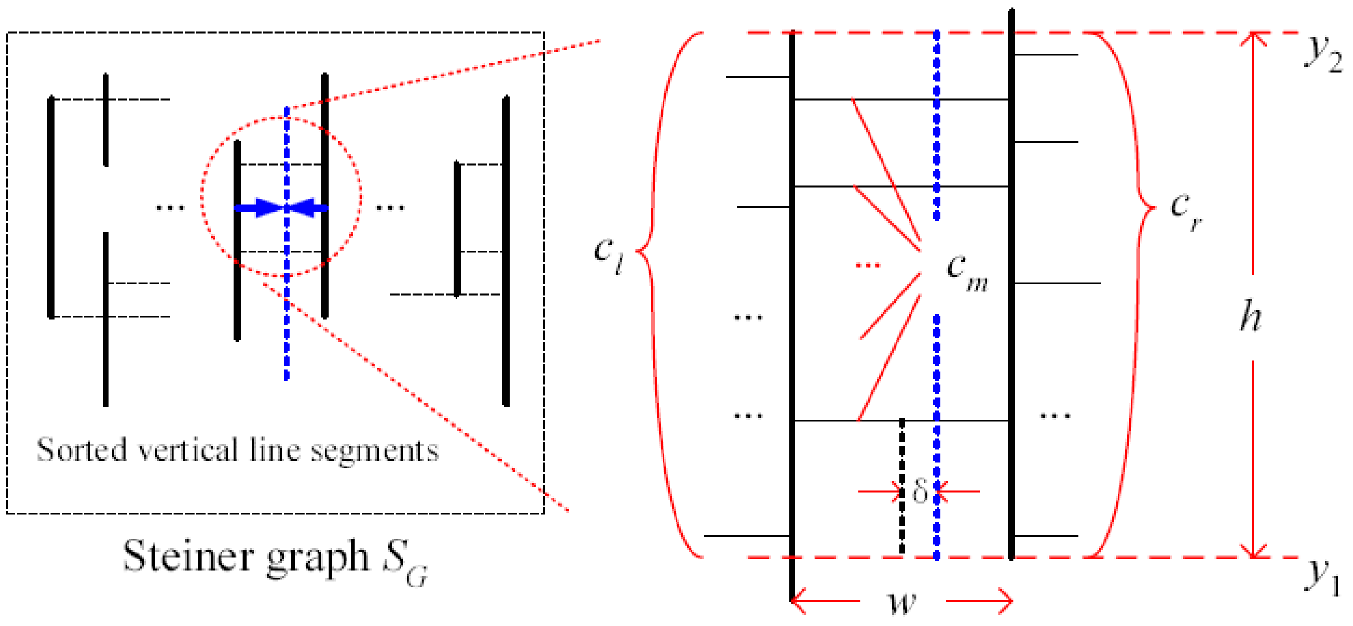
- High bandwidth+short paths → more wires
- Loosen the shortest-path constraint
  - E.g. (1+ ε) Manhattan distance
  - Merge parallel edges → reduce wires
  - Low increase on path length / dynamic power



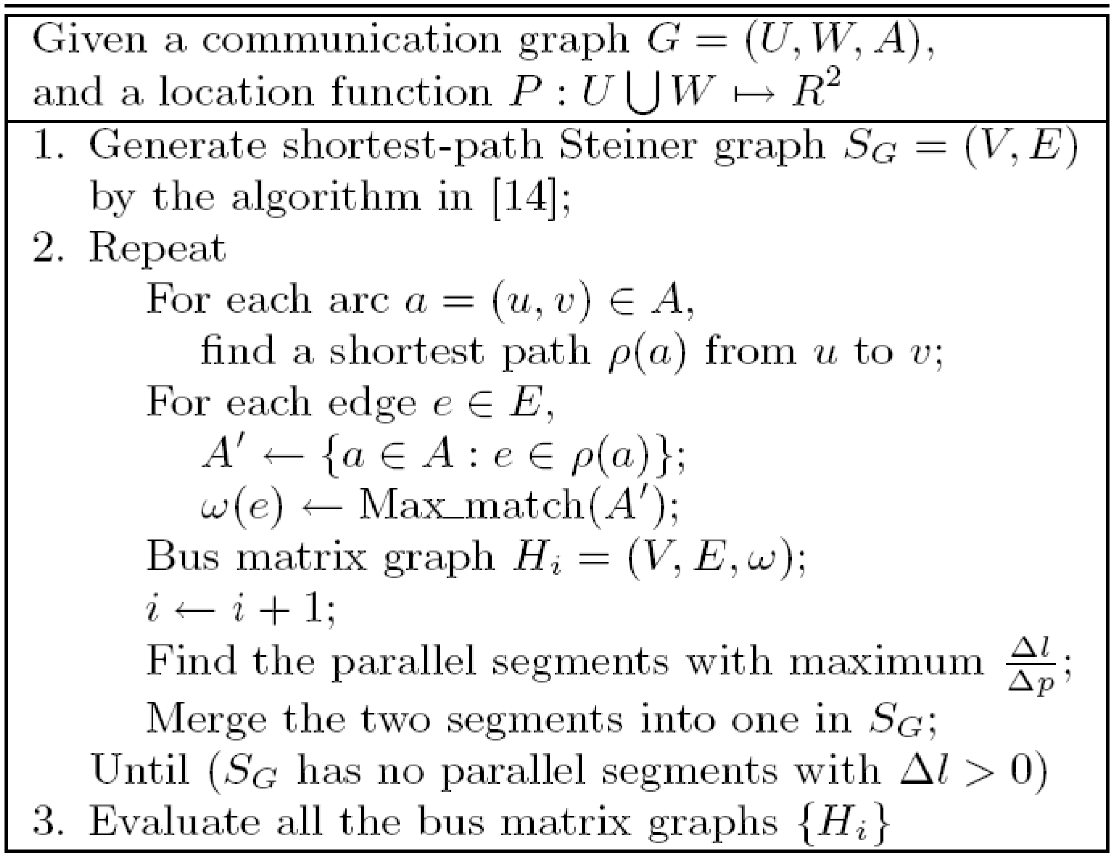
Parallel Segment Merging

- Iteratively, find parallel double segments
  - Δl – edge length (not wire length) reduction
  - Δp – possible path length increase
  - Merge the pair with maximum $\Delta l / \Delta p$

$$= \frac{h + c_m w - c_l(\frac{w}{2} + \delta) - c_r(\frac{w}{2} - \delta)}{w + 2\delta}$$



Overall Flow

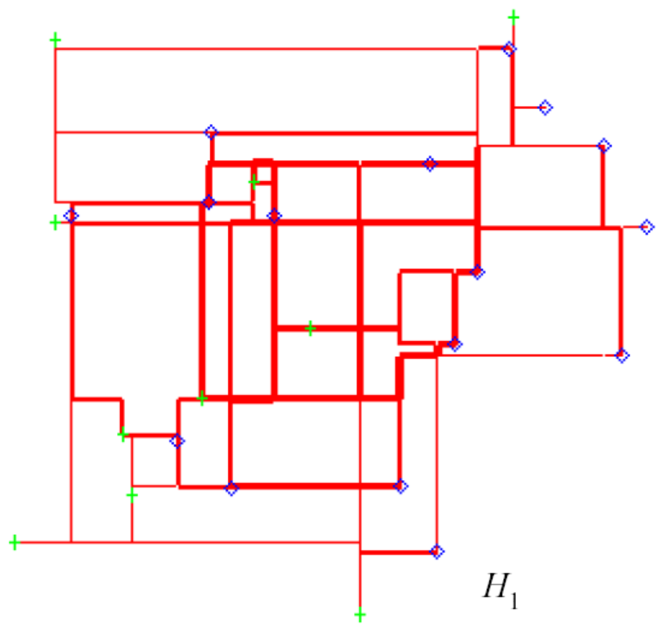


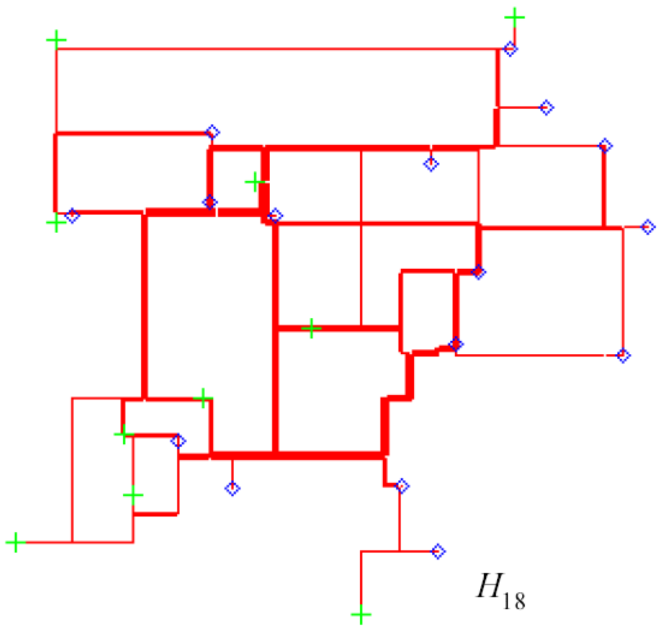
```
Given a communication graph $G = (U, W, A)$,
and a location function $P : U \bigcup W \mapsto R^2$
1. Generate shortest-path Steiner graph $S_G = (V, E)$
   by the algorithm in [14];
2. Repeat
       For each arc $a = (u, v) \in A$,
           find a shortest path $\rho(a)$ from $u$ to $v$;
       For each edge $e \in E$,
           $A' \leftarrow \{a \in A : e \in \rho(a)\}$;
           $\omega(e) \leftarrow \text{Max\_match}(A')$;
       Bus matrix graph $H_i = (V, E, \omega)$;
       $i \leftarrow i + 1$;
       Find the parallel segments with maximum $\frac{\Delta l}{\Delta p}$;
       Merge the two segments into one in $S_G$;
   Until ($S_G$ has no parallel segments with $\Delta l > 0$)
3. Evaluate all the bus matrix graphs $\{H_i\}$
```

- Low complexity in each iteration
  - Most time consumed by max-matching O(|U+W||A||E|)

Experimental Results

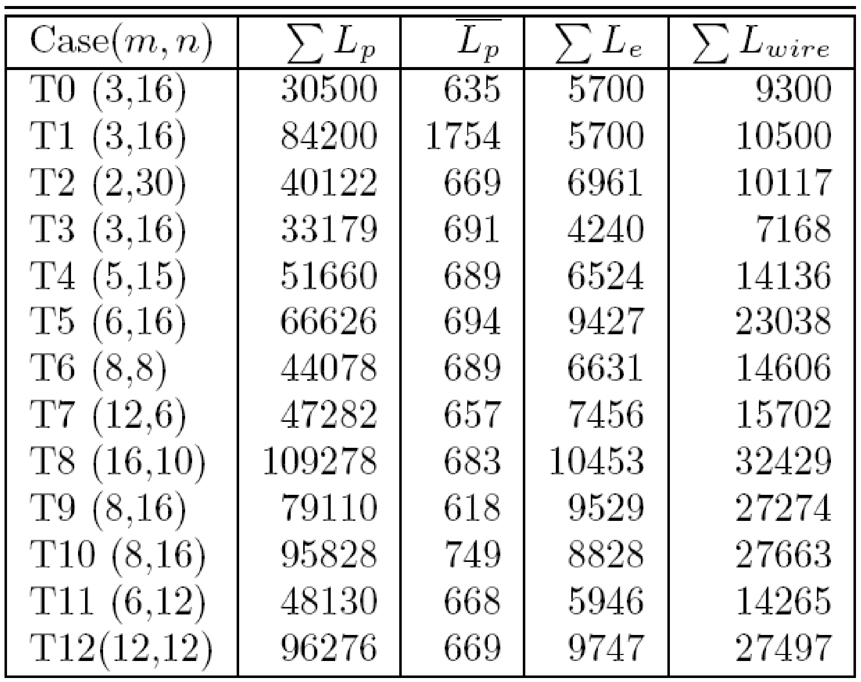
- Same random cases as in [Wang09]
- Maximum bandwidth guaranteed
  - Min-power bus matrix (w/o segment merging)
  - Min-wire bus matrix



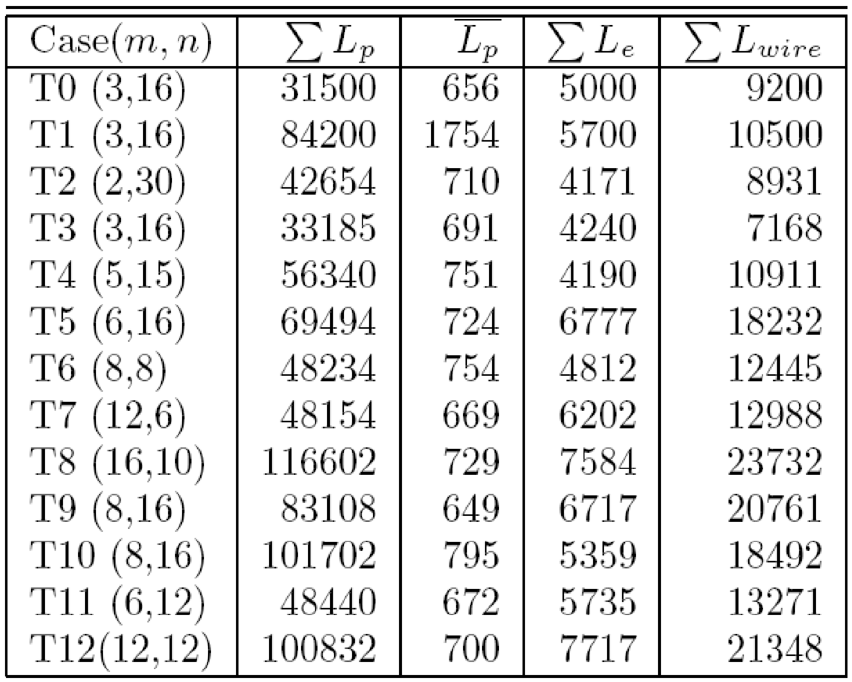


Experimental Results (cont.)

- Min-power to Min-wire, on average
  - Total wire length reduced by 15.5%
  - Average path length increased by 4.4%



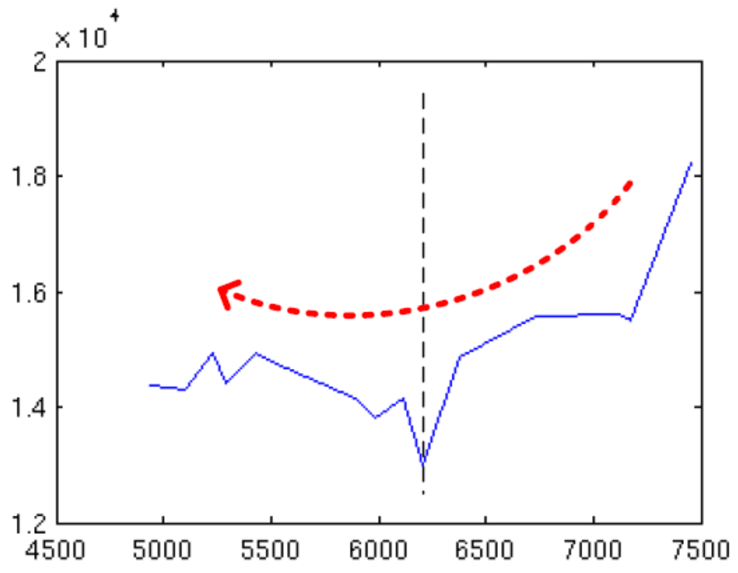
Case(m,n)	$\sum L_p$	$\overline{L_p}$	$\sum L_e$	$\sum L_{wire}$
T0 (3,16)	30500	635	5700	9300
T1 (3,16)	84200	1754	5700	10500
T2 (2,30)	40122	669	6961	10117
T3 (3,16)	33179	691	4240	7168
T4 (5,15)	51660	689	6524	14136
T5 (6,16)	66626	694	9427	23038
T6 (8,8)	44078	689	6631	14606
T7 (12,6)	47282	657	7456	15702
T8 (16,10)	109278	683	10453	32429
T9 (8,16)	79110	618	9529	27274
T10 (8,16)	95828	749	8828	27663
T11 (6,12)	48130	668	5946	14265
T12(12,12)	96276	669	9747	27497



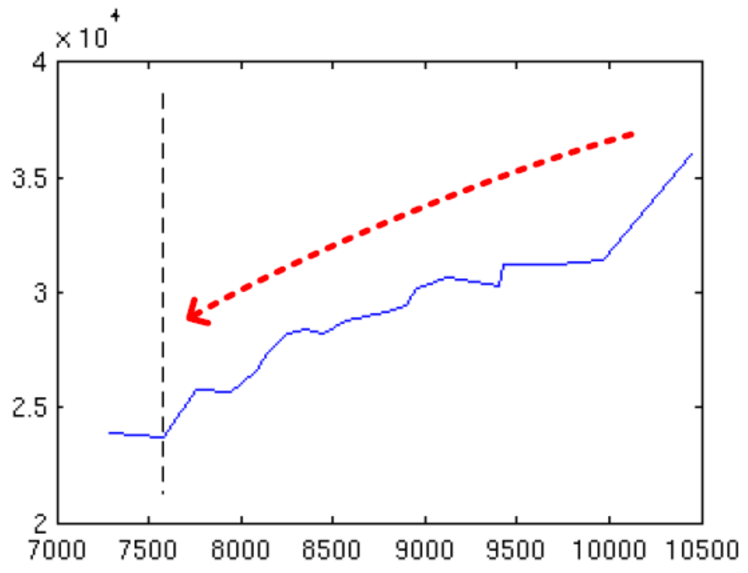
Case(m,n)	$\sum L_p$	$\overline{L_p}$	$\sum L_e$	$\sum L_{wire}$
T0 (3,16)	31500	656	5000	9200
T1 (3,16)	84200	1754	5700	10500
T2 (2,30)	42654	710	4171	8931
T3 (3,16)	33185	691	4240	7168
T4 (5,15)	56340	751	4190	10911
T5 (6,16)	69494	724	6777	18232
T6 (8,8)	48234	754	4812	12445
T7 (12,6)	48154	669	6202	12988
T8 (16,10)	116602	729	7584	23732
T9 (8,16)	83108	649	6717	20761
T10 (8,16)	101702	795	5359	18492
T11 (6,12)	48440	672	5735	13271
T12(12,12)	100832	700	7717	21348

Experimental Results (cont.)

- Total wire length vs. total edge length along parallel segment merging operations
  - First decreasing (less edges)
  - Then increasing (longer paths)



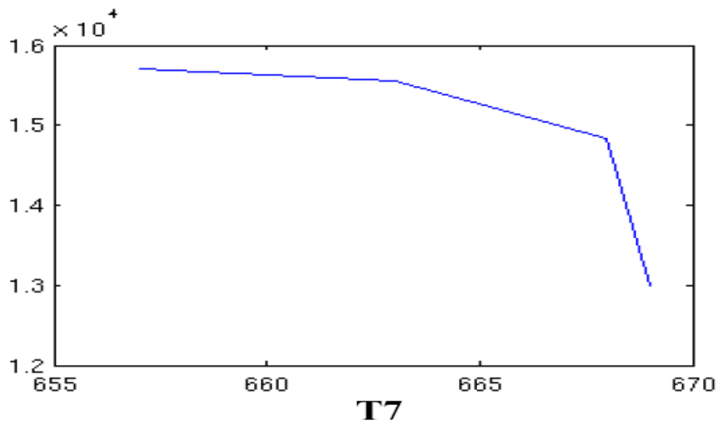
T7

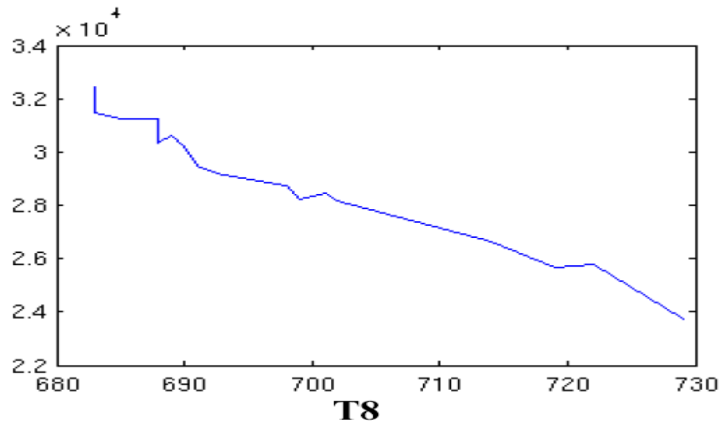


T8

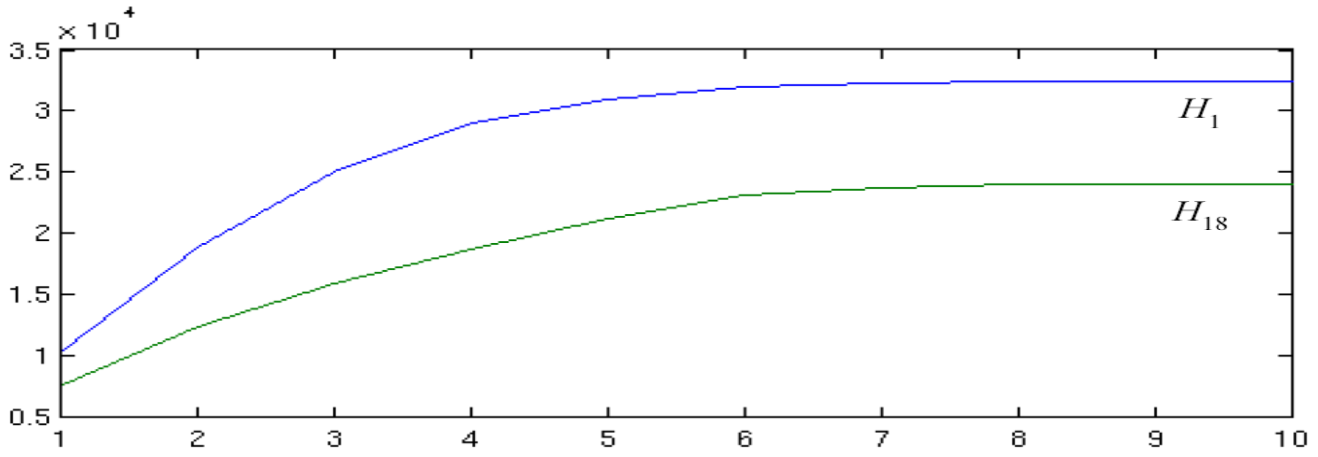
Experimental Results (cont.)

- Tradeoff between wire & power





- Tradeoff between wire & bandwidth



Conclusions

- On chip bus matrix can be strong at
  - Performance
    - Small delay (by centralized arbitration & control)
    - Consistent bandwidth
  - Efficiency
    - on power (shortest connections)
    - on wire (sharing bus lines in Steiner graphs)
- More possibilities
  - Architectures (AMBA AHB, CoreConnect…)
  - Communication patterns

Questions & Answers

- Thank you for your attention!