Communication Architecture Synthesis of Cascaded Bus Matrix

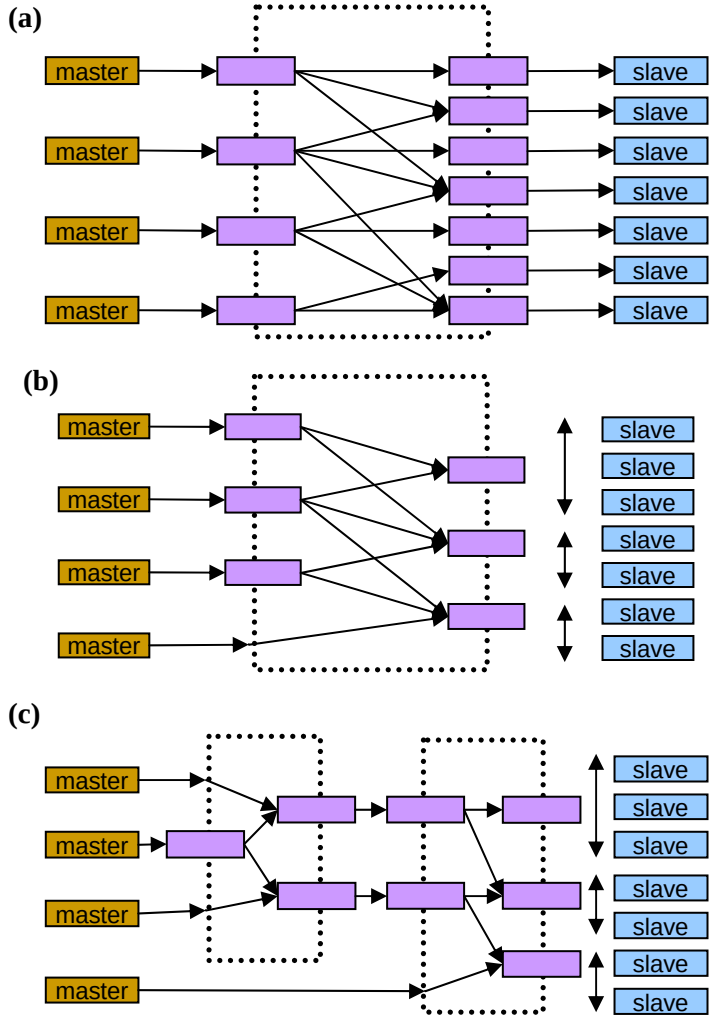
Junhee Yoo*, Dongwook Lee*, Sungjoo Yoo and Kiyoung Choi*

*Seoul National University, Seoul, Korea

**Samsung Electronics, Yongin, Korea

Slides for ASPDAC 2007

Motivation

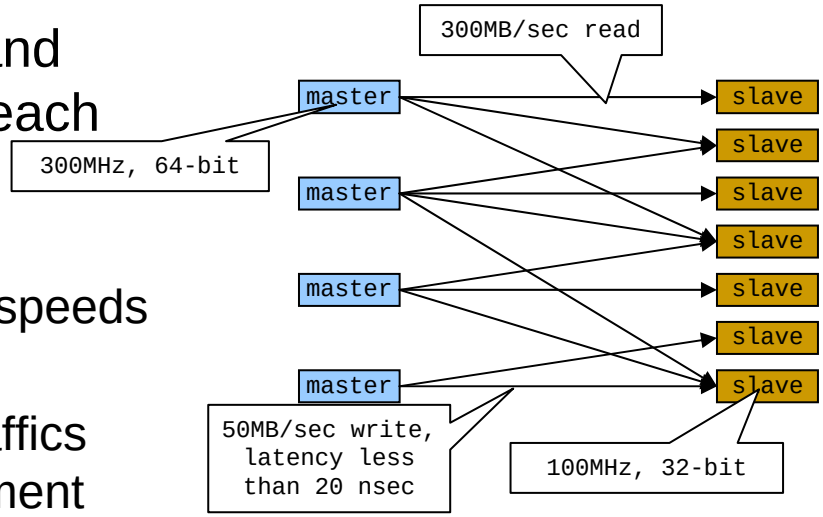


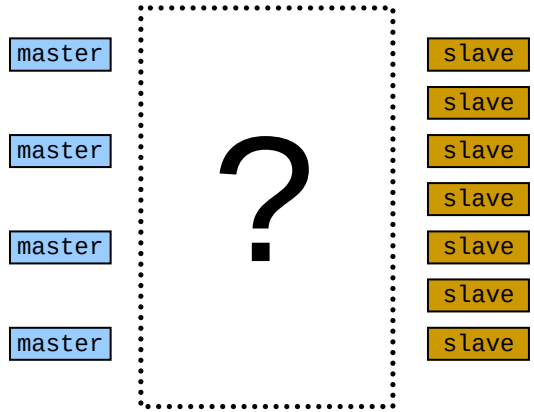
- Single bus matrix architecture
  - No bottleneck compared to shared bus, but number of connections increase rapidly as design gets larger
- Single bus matrix architecture with local buses [Pasricha 2006]
  - The bus matrix's size has reduced significantly
  - However, we can predict that it will become still too large as design gets larger
- Multiple bus matrix architecture
  - Can further reduce the bus matrix's size (?)

[Pasricha 2006] S. Pasricha, N. Dutt, M. Men-Romdhane, "Constraint-Driven Bus Matrix Synthesis for MPSoC", in proc. of ASPDAC, Jan. 2006

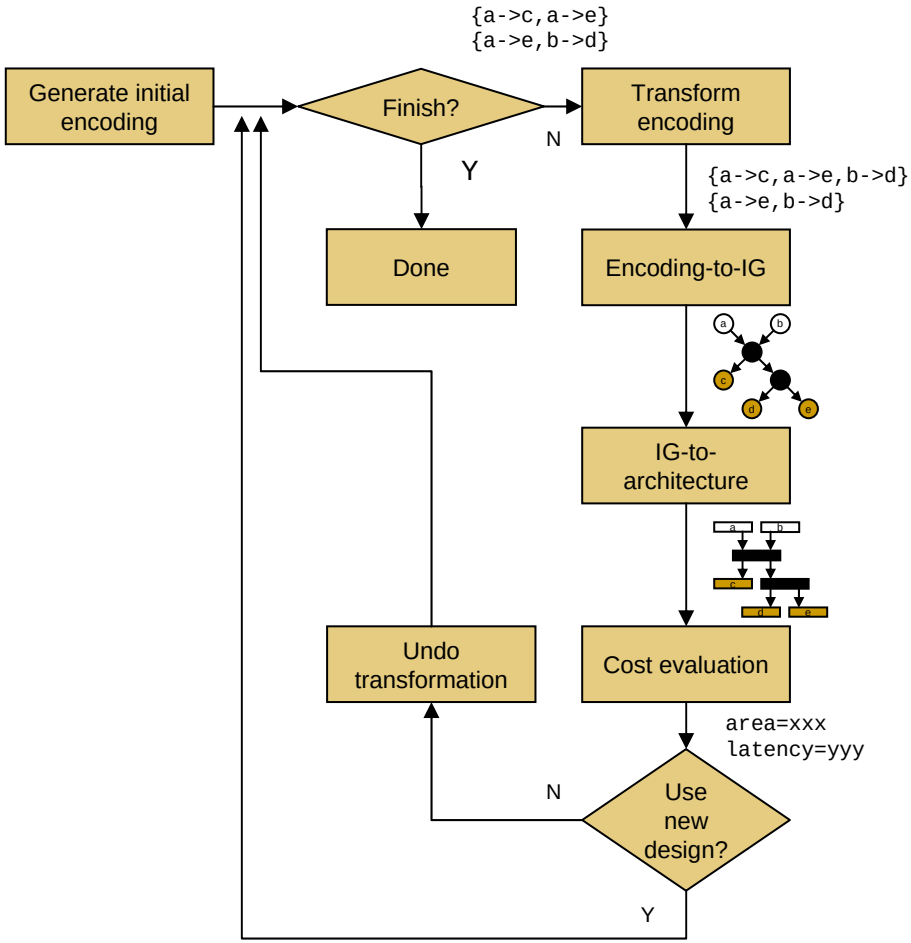
Problem Definition

- *(CTG)* You want some masters and some slaves to communicate to each other…
  - *(IP information)* Where each components have different clock speeds and different data width
  - *(Application information)* Each traffics have different bandwidth requirement and latency requirement
- …and what architecture will be optimal?
  - In terms of die area/power or whatsoever, but in this paper, we deal with bus area.
  - The generated architecture is based on AMBA3 AXI components from ARM

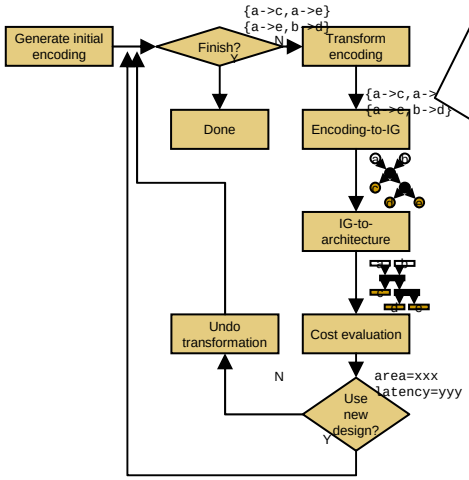


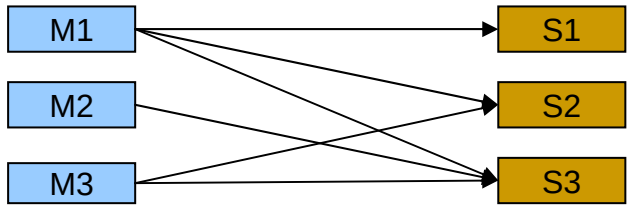


Simulated Annealing Flow



Introducing TGE



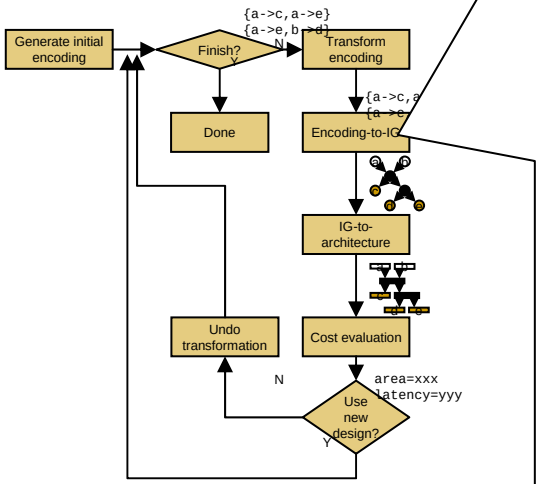


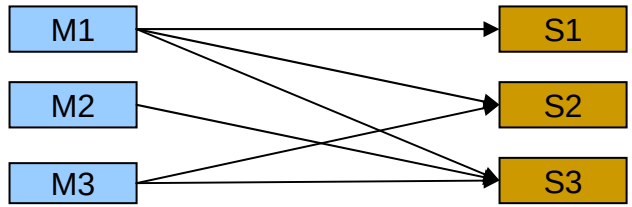

```
TGE = {{ M1->S1, M1->S2, M1->S3, M2->S3 },
       { M1->S2, M1->S3, M2->S3, M3->S2, M3      } }
```

The order between the groups represent the topological order of the two bus matrices – the preceding group may have a path to the succeeding group, but not vice versa.

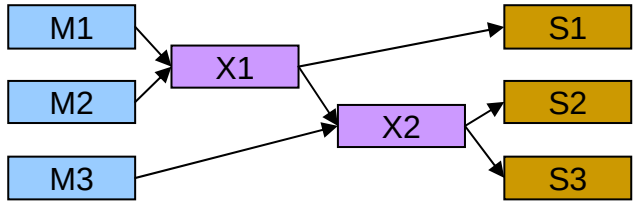
- We added two more parameters – '*group clock speed*' and '*group data width*' which are the clock speed / data width of the generated bus matrix.

Encoding-to-topology transformation



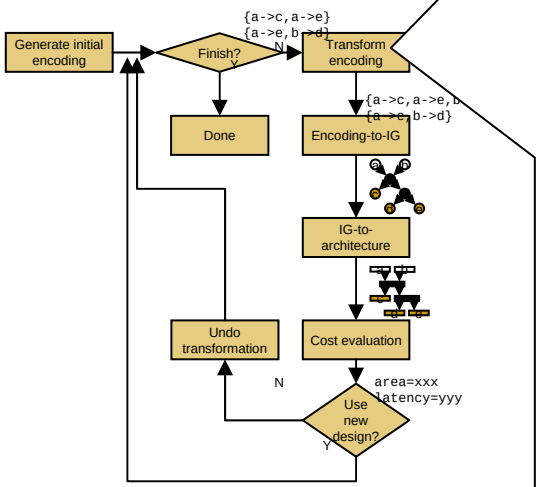


TGE = { { M1->S1, M1->S2, M1->S3, M2->S3 } ,
{ M1->S2, M1->S3, M2->S3, M3->S2, M3->S3 } }



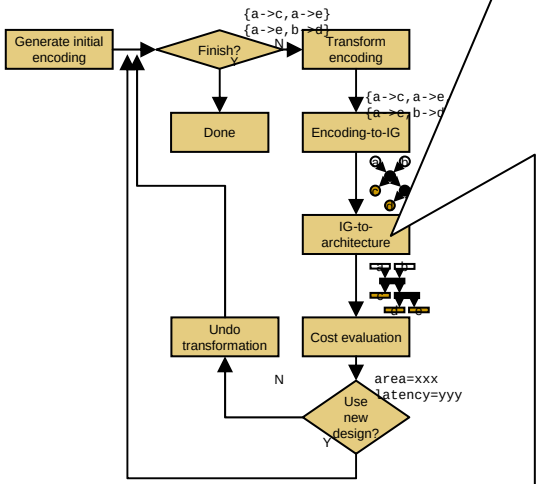
- For any encoding, the generated topology is always legal
- Easily applicable to any other meta-heuristic method

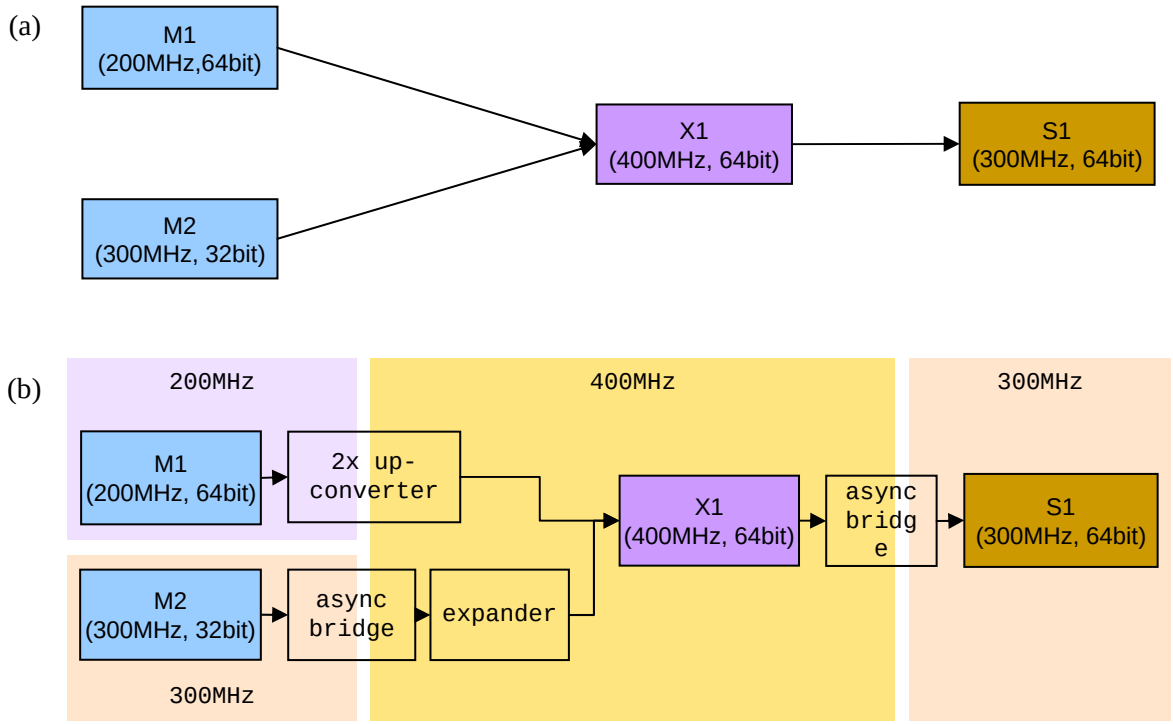
Transform encoding



- Create a random group with two or more traffics
- Remove a random traffic from a random group
- Merging two random groups into one
- Adding all traffics from a group to another group
- Removing all traffics from a group from another group
- Changing a random parameter of the group (clock speed and/or data width)
- Adding/removing all traffics from the same random master/slave
- Changing the order of two random groups

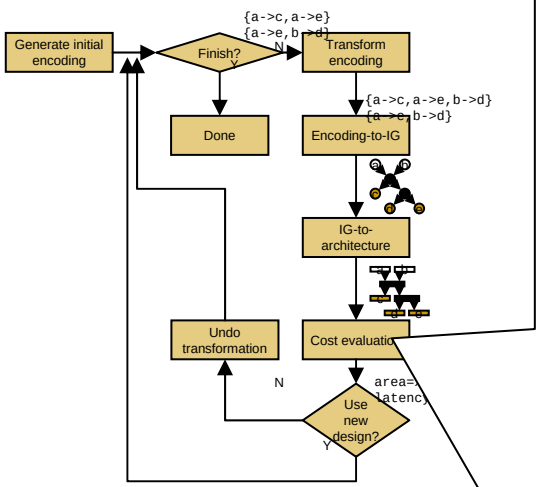
IG-to-architecture





- Bridges are added when ports or bus matrices have different characteristics.
- Timing analysis is done, and register slices are added where required.

Cost evaluation



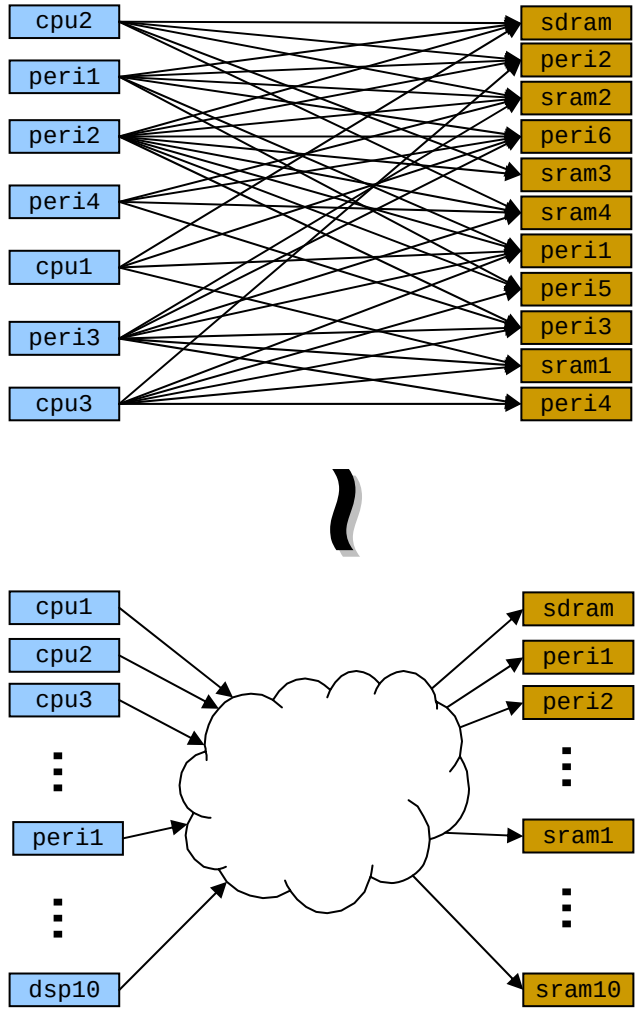
- Two types of costs – *penalty* and *optimization goal*
- Penalty
  - The ‘amount’ of **timing violation** and **traffic latency violation** of the architecture
- Optimization goal
  - **Gate count** of the architecture
- Cost function for annealing
  - Initially, only *penalty*
  - After *penalty* converges to zero,

$$cost = optimization\_goal \cdot e^{penalty}$$

- Final solution
  - The solution with the smallest optimization goal where penalty is zero

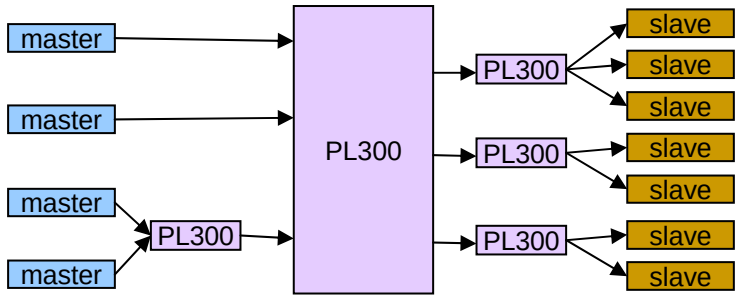
Experiments

- Used 120 randomly generated synthetic CTGs
  - There weren't enough CTGs available that were large enough to show the usefulness of this flow
  - Smallest ones had 18 ports, while largest ones had 100+ ports
- Execution time took around 10 minutes for smallest ones, and 2 hours for largest ones



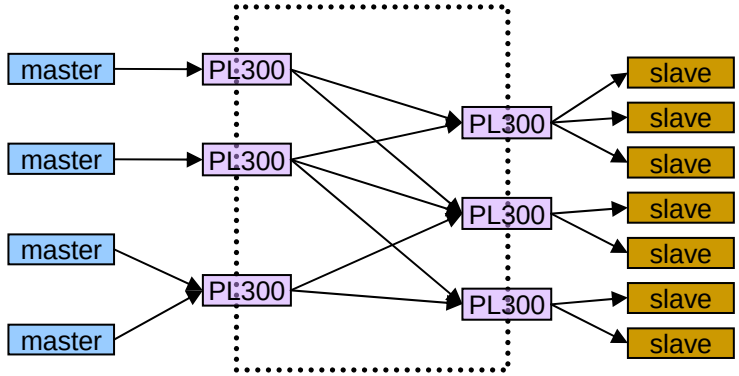
The other methods for comparison

- SBM



traditional single bus matrix with shared buses

- 2BM



A trivial extension of SBM

- Two-phase:
  - Use 2BM or SBM for the 70% of annealing, and then change encoding to TGE
  - Searches a smaller space initially (so that it can converge quickly), but utilizes the larger design space later on

Experiment results

Method		SBM	2BM	TGE	Two-phase
All designs	Average size (geometric mean)	-	6.403	7.172	5.325
	Number of successful synthesizes	78	120	119	120
	number of best results among 4 methods	33	1	31	55

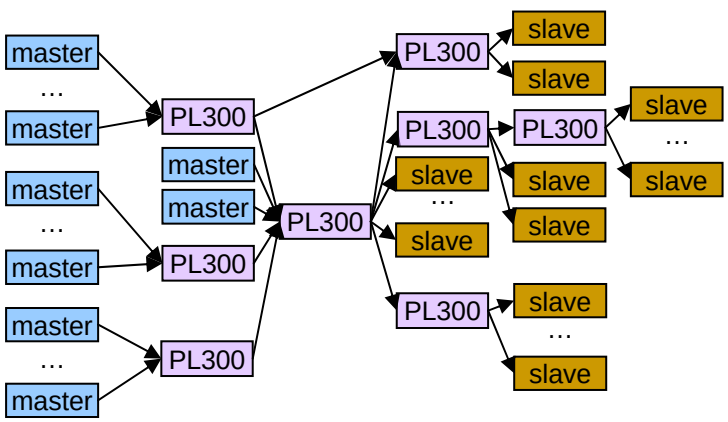
Experiment results

Method		SBM	2BM	TGE	Two-phase
small 27 designs (< 25 ports)	Average size (geometric mean)	2.991	3.641	3.097	3.159
	Number of successful synthesizes	27	27	27	27
	number of best results among 4 methods	9	0	4	14
large 22 designs (> 45 ports)	Average size (geometric mean)	-	9.434	15.799	8.309
	Number of successful synthesizes	2	22	22	22
	number of best results among 4 methods	1	0	5	16

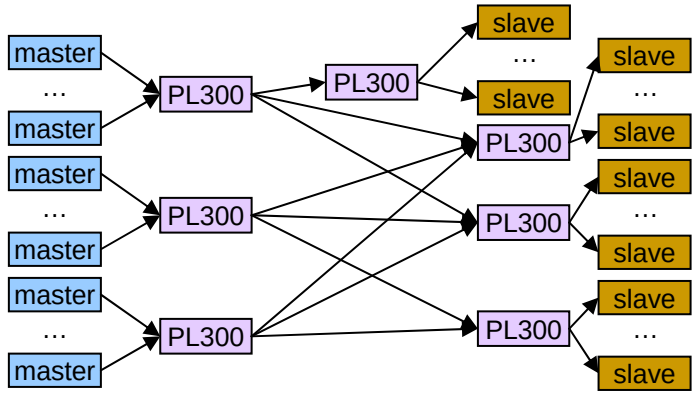
- Results are not so good for smaller designs
  - The search space is much bigger than a single bus matrix, thus it takes more iterations to converge to a reasonable solution
- …but tends to get better for larger designs
  - For larger designs, cascading greatly help increasing scalability

One example result

- A simplified topology of an architecture with 32 masters and 75 slaves.



TGE-based



2BM-based

- Using the TGE method, an architecture with 4 cascaded stages was generated.
- The 2bm method had to use large bus matrices on the slave side, where a single bus matrix connects to 20+ slaves.
- Gate count ratio is 0.81:1

Conclusions

- We propose a new synthesis method for cascaded bus matrices
  - Due to a larger solution space, It gives better results on large designs.
  - Probably possible to extend to network-on-chip designing
- Need much more improvements
  - Need to consider other meta-heuristic methods (genetic algorithm or ant colony optimization)
  - Need more accurate performance/area/power modeling methods