CS294-7: Reconfigurable Computing

Day 25: Tuesday, April 22

Everything Else...

- Admin
- Multicontext

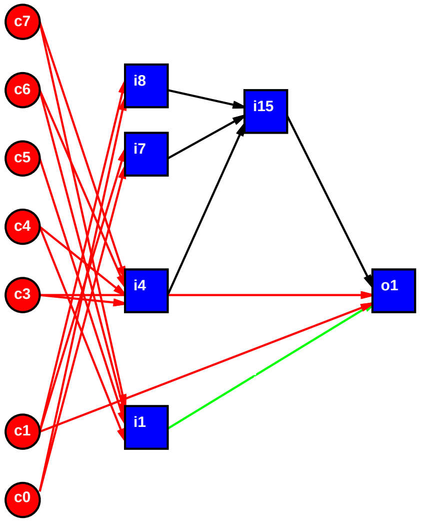
Admin

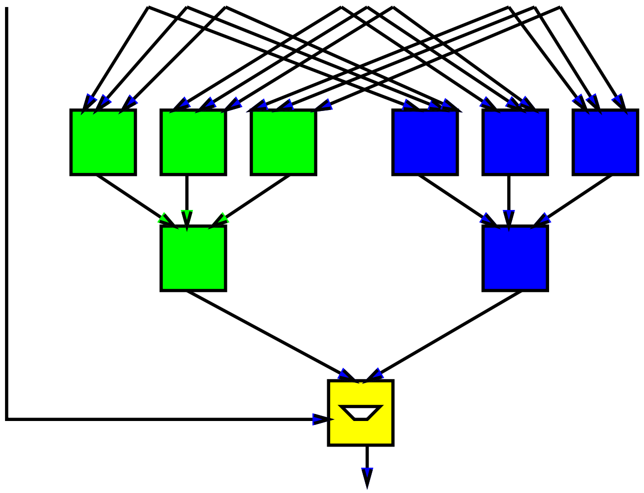
- Projects Due in 2 weeks
- Signup for Project report slots next time
  - 10 Minute talk slots
  - Three periods
    (Class time T and R + T afternoon 5-6:30pm)
- Reading Comments

Motivation

What happens when:

- Have cycles? (and latency important)
- Low throughput, unstructured tasks?
- Operation required is data dependent?





Consequence

Waste Resources:
$\Rightarrow$ Most of the resources idle most of the time.

Why does this happen?

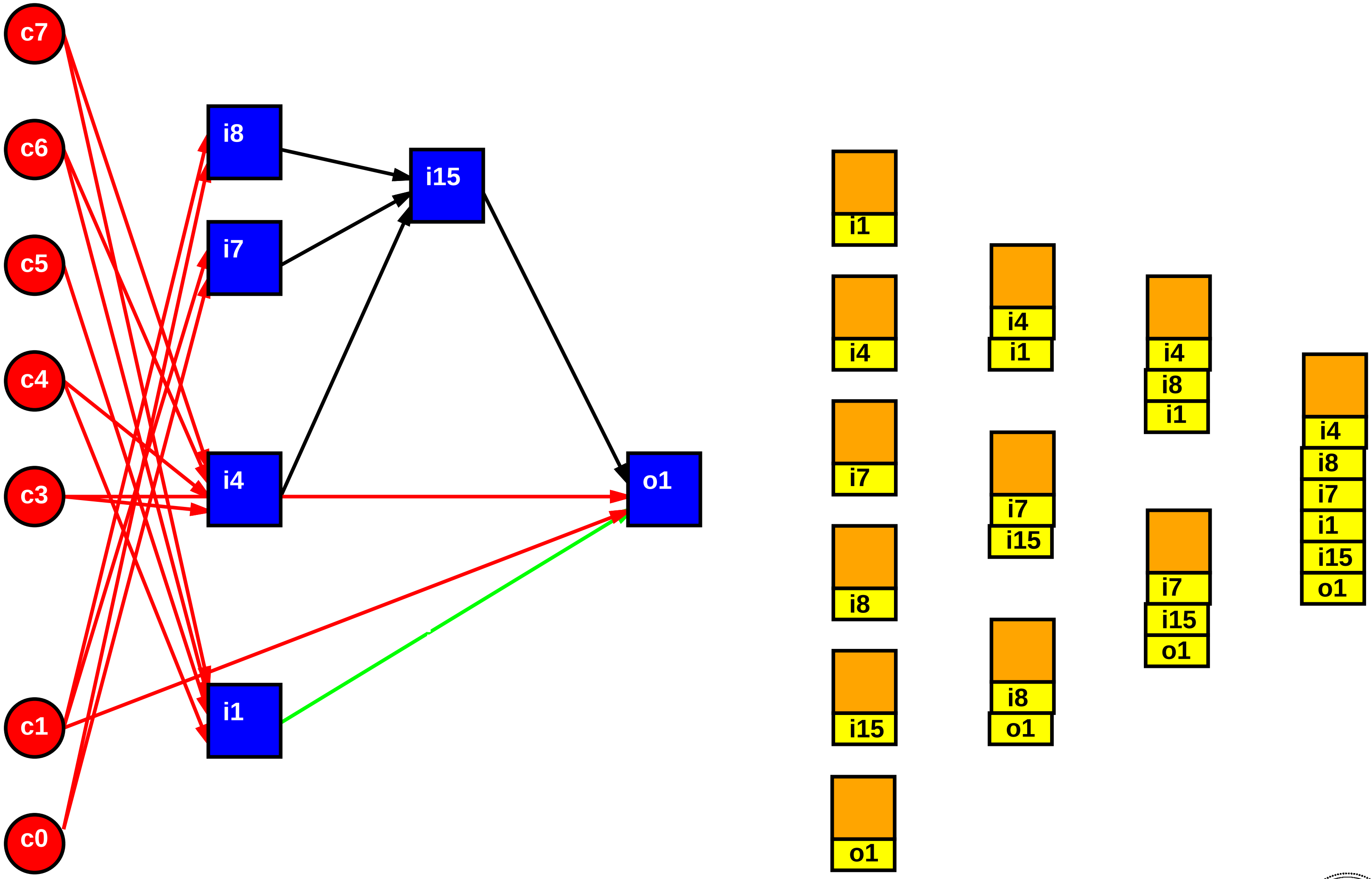
Reusing Resources

Want to *share* resources.

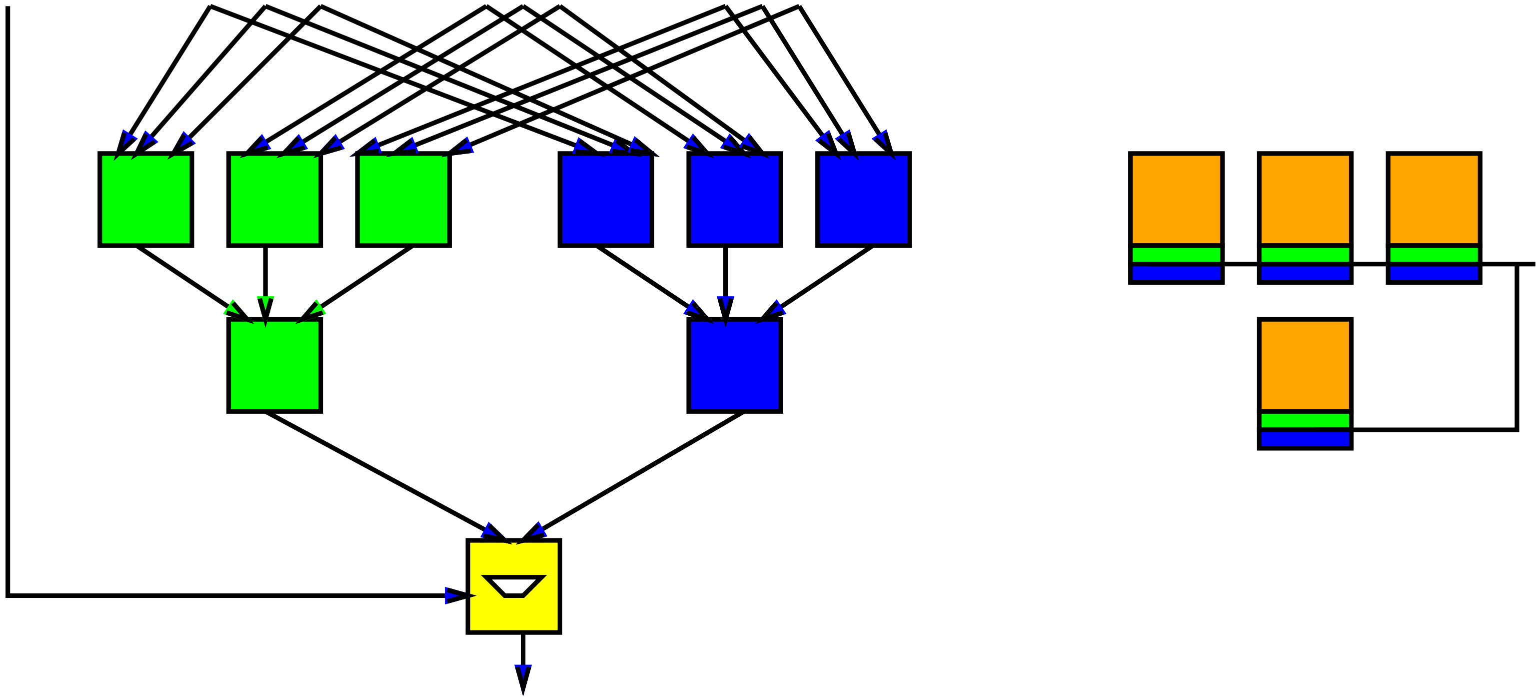
Must be able to tell resource how to behave.

Need separate instruction (pinst) for each behavior.

Reusing Resources



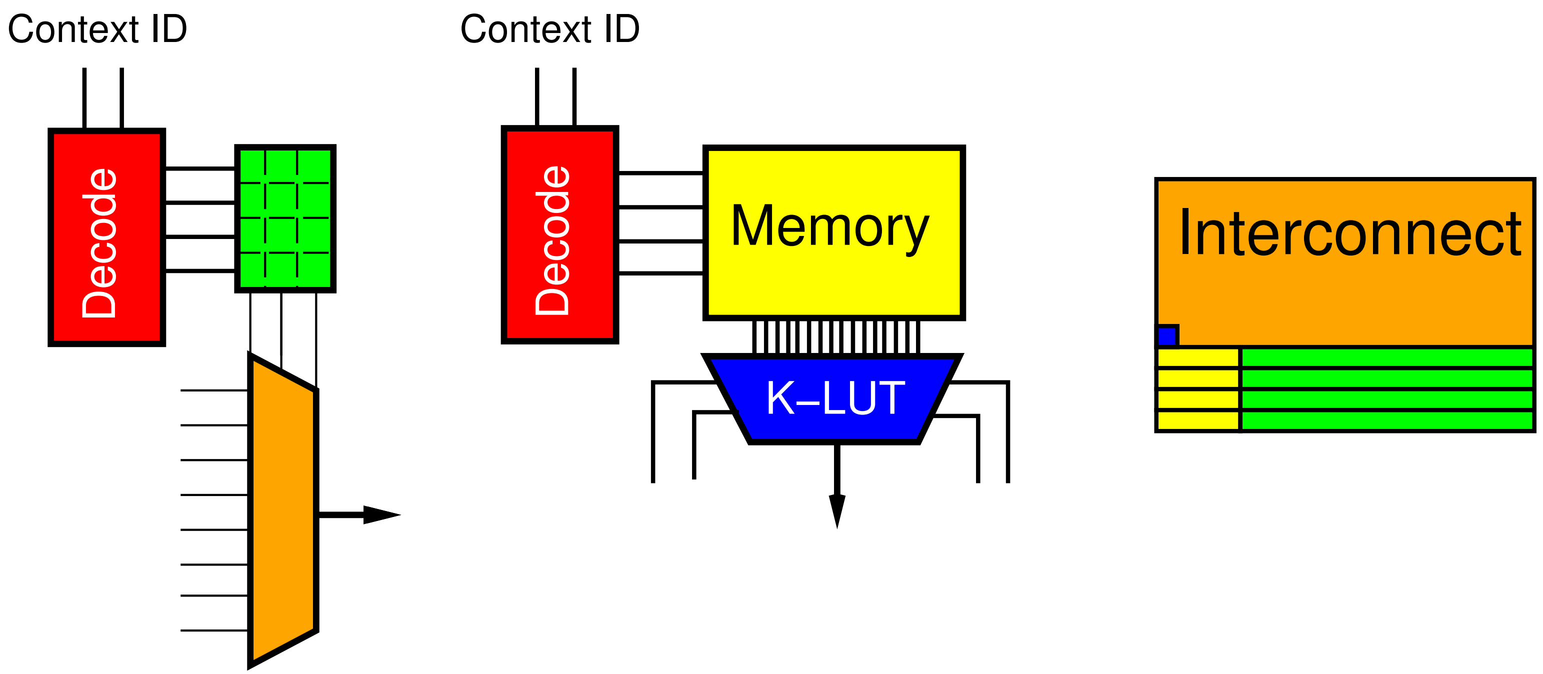
Reusing Resources



Multicontext Issues

- Context (instruction) costs
- Packing (Scheduling)
- Retiming (limitations, structures)

Multicontext Organization/Area

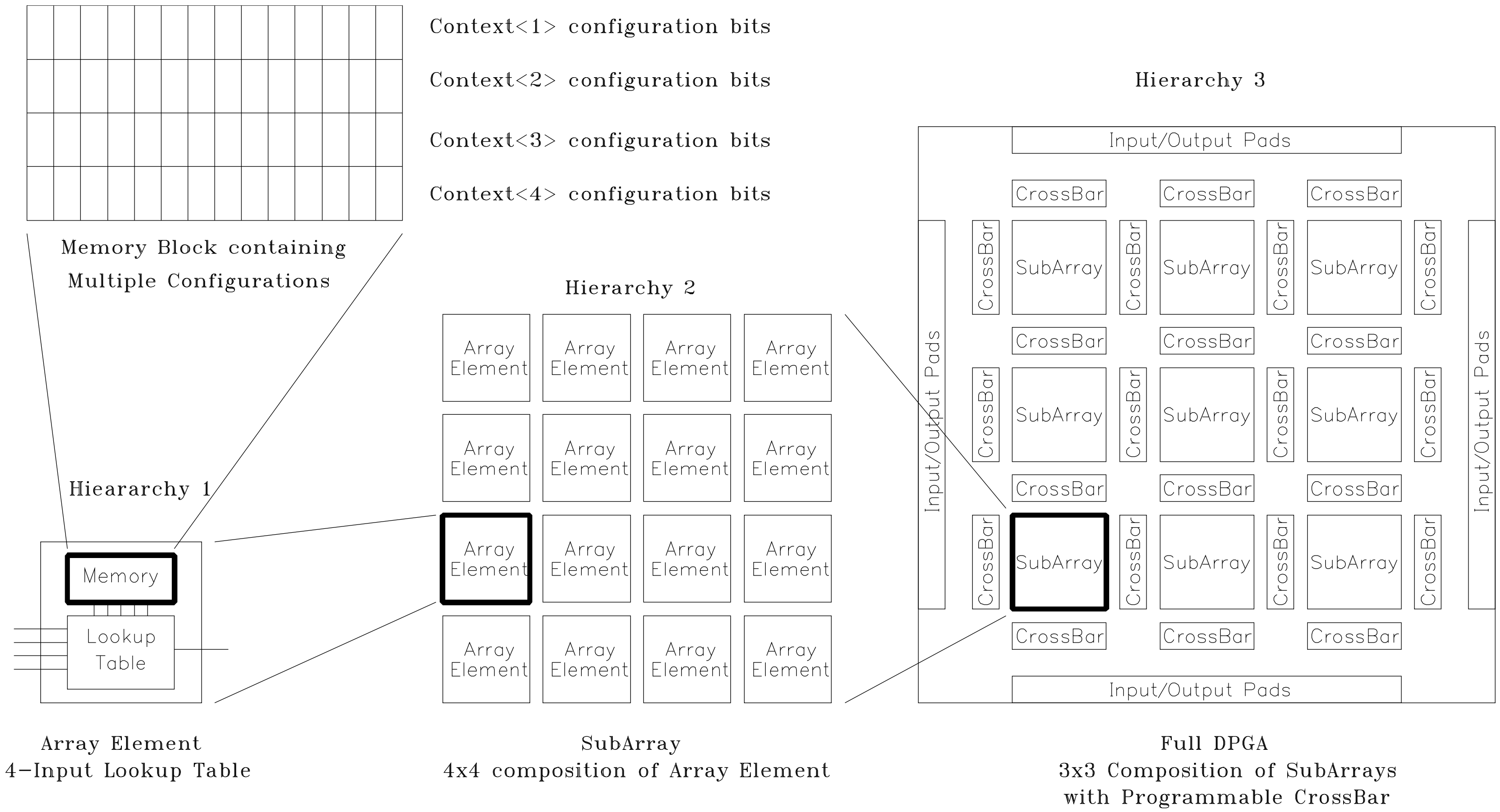


$$A_{ctxt} \approx 80\text{K}\lambda^2$$
$$A_{base} \approx 800\text{K}\lambda^2$$

$$A_{base} : A_{ctx} = 10 : 1$$

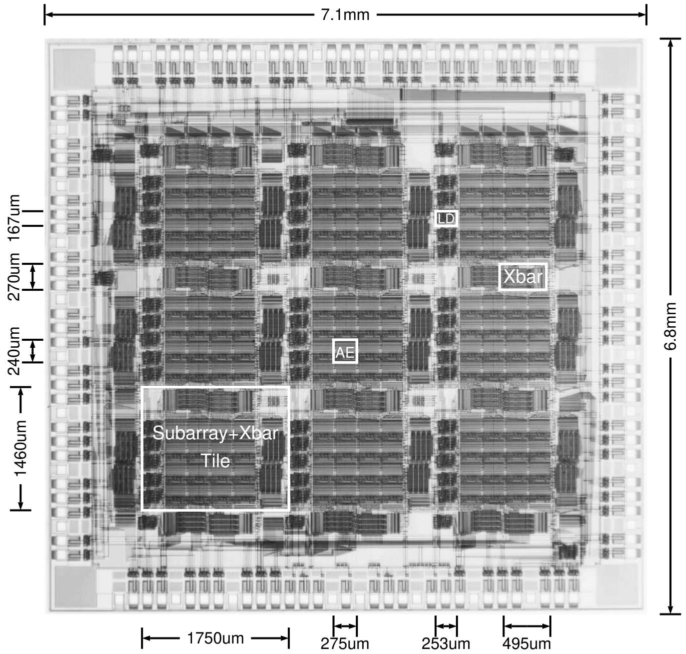
DPGA Organization

Dynamically Programmable Gate Array Prototype

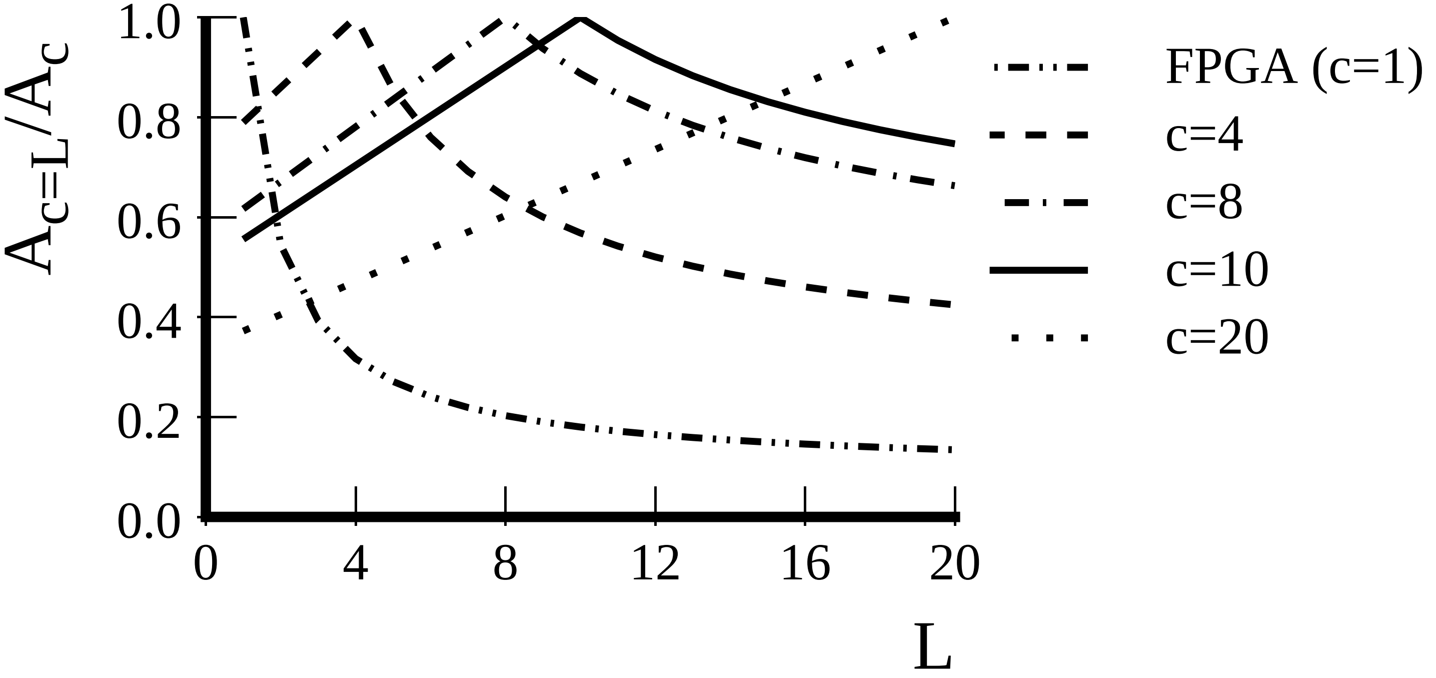


DPGA Prototype IC

Process	1.0μ CMOS
Chip	7.1mm$\times$6.8mm
AEs	144
Contexts	4
AE Area	640Kλ^2
A_{base}	544Kλ^2
A_{ctx}	24Kλ^2
$A_{base} : A_{ctx}$	20+:1
(nominal delay)	9ns



Ideal Tradeoff Curves



Assumes ideal packing ($n_p = \frac{n_d}{L}$)

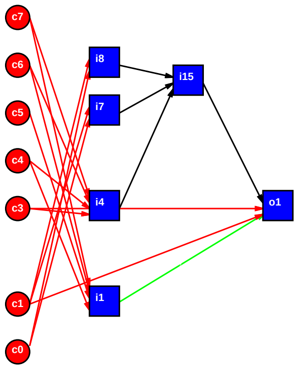
N.B. **Robust point when:** $c \cdot A_{ctx} = A_{base}$

Scheduling and Packing

Resource requirements dictated by the number required in the largest stage.

Packing limits: for correctness, can evaluate a LUT only if all of its predecessors were evaluated in a previous stage.

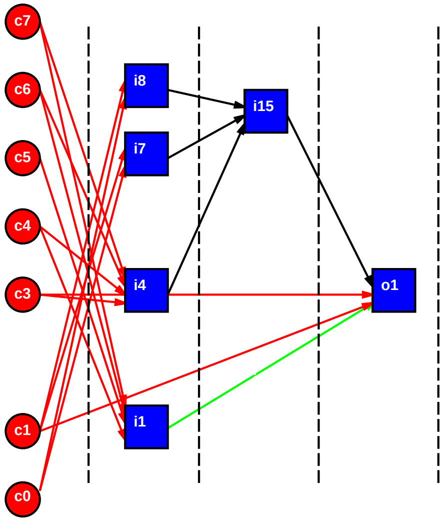
When latency important, can schedule in L (=critical path length) cycles.

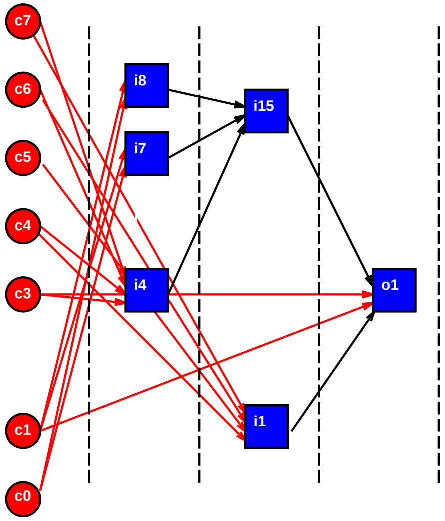


Scheduling and Packing

Precedence constraints generally prevents perfect (balanced) packing.

Slack in network provides freedom to balance resources.



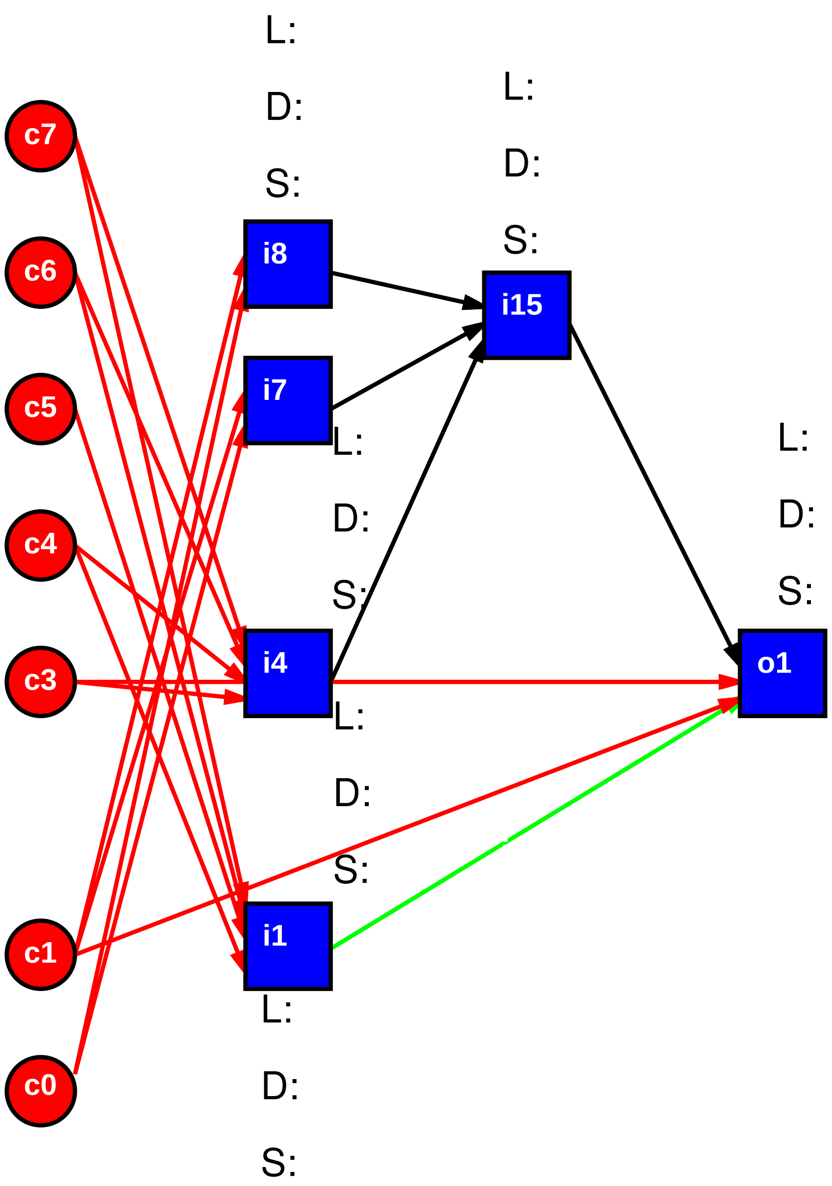


Scheduling and Packing

We know how to compute slack in the network:

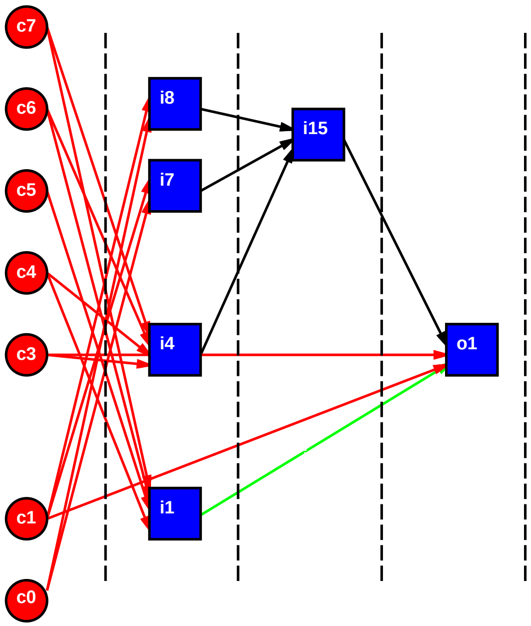
- Propogate depth forward from primary inputs
  (ASAP schedule)
  (depth = 1 + max input depth)
- Propogate distance from outputs back from outputs
  (ALAP schedule)
  (level = 1 + max output consumption level)
- slack = L - (depth+level)
- Once start assigning evaluation time slots:
  (slack = L - (max input time + min output time + 1)

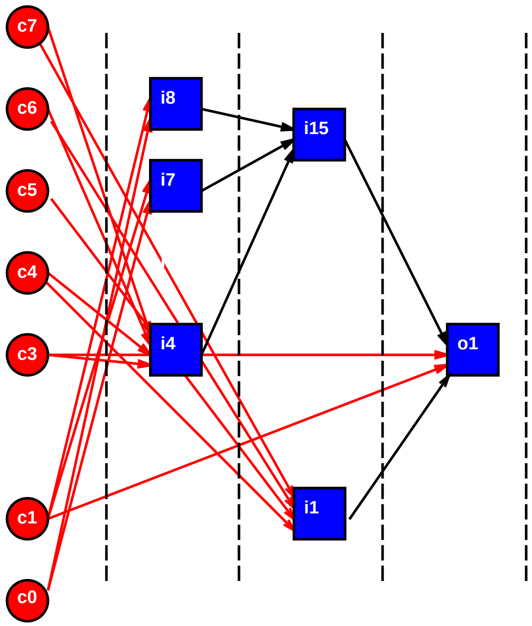
Scheduling and Packing



Scheduling and Packing

Allowable schedules:

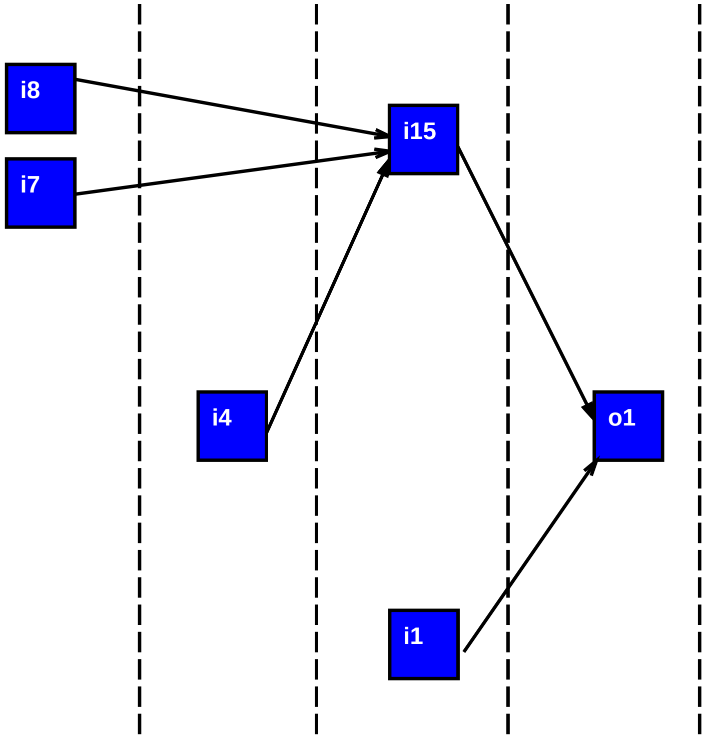




Requires 3 physical LUTs.

Scheduling and Packing

Adding time slots, adds slacks, and permits better balancing (at cost of increased latency).



4 time slots, 2 physical LUTs, (maybe 3 contexts?)

Scheduling and Packing

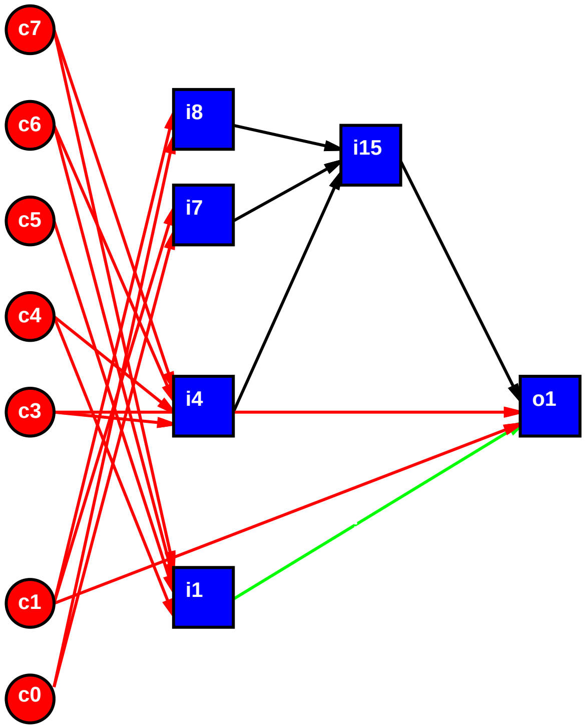
Retiming in this case $\rightarrow$ spreading out resource usage (minimizing peak resource requirements).

Much like minimizing peak channel capacities during placement

Rabaey shows retiming for scheduling is NP-complete.

Signal Retiming

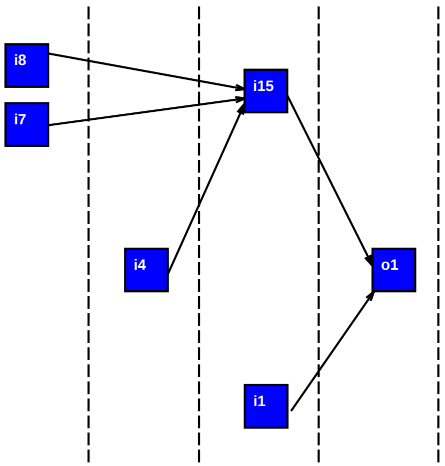
What about intermediate data storage (retiming)?



Signal Retiming

Non-pipelined, single context
$\rightarrow$ worked by holding values on wires.
(wasteful of critical wire resources)

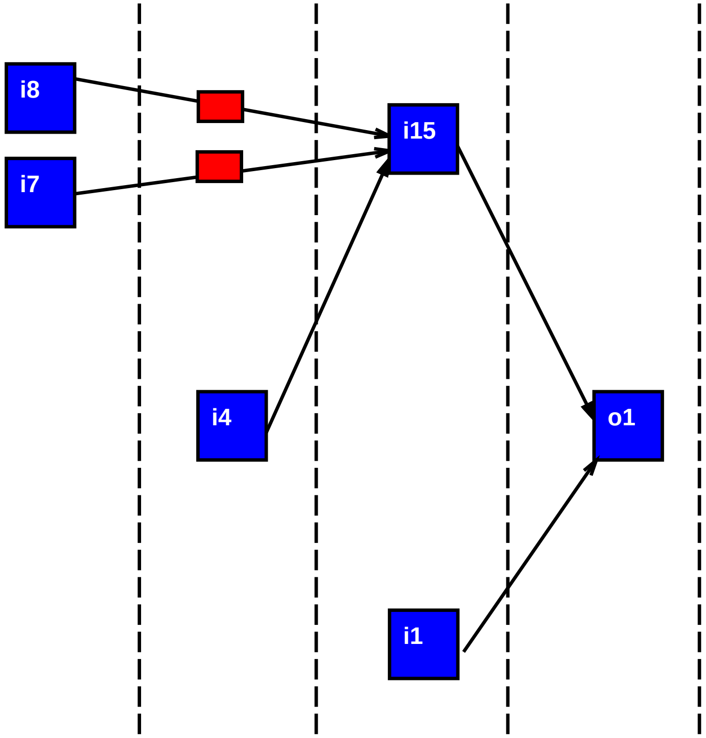
Single output flip-flop
$\rightarrow$ requires one LUT per signal crossing through a stage



Signal Retiming

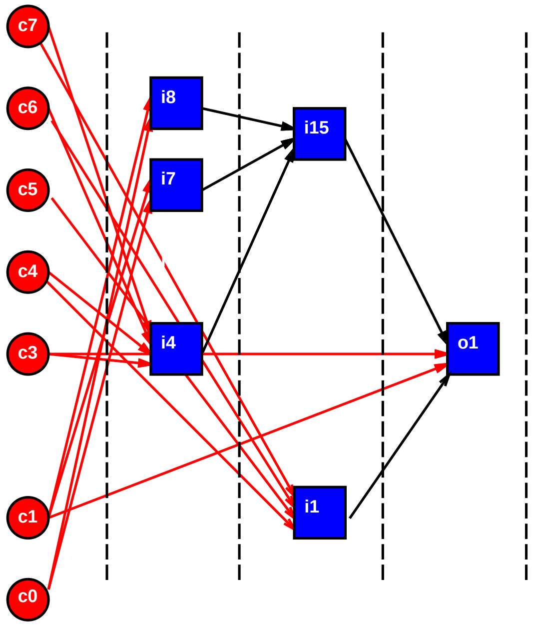
Net buffer

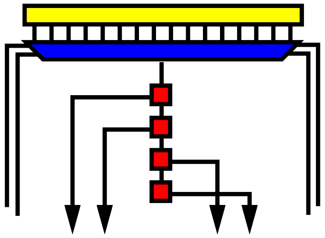
→ **smaller resource for retiming**



Signal Retiming

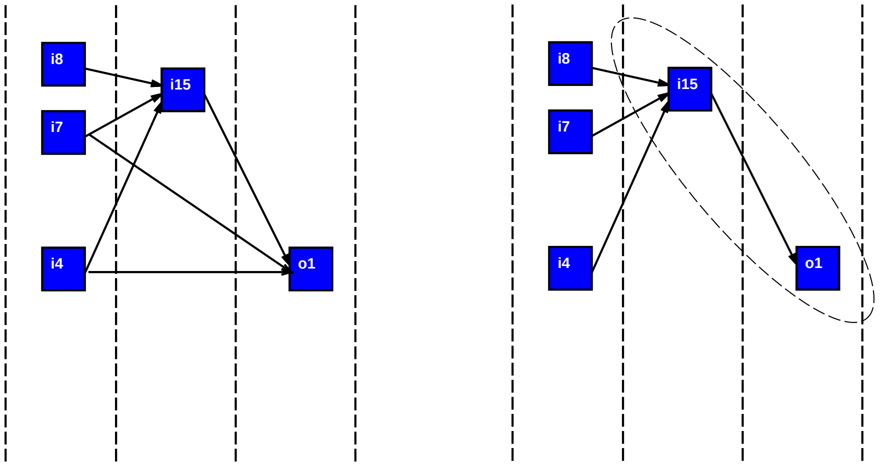
Output buffer chain
→ may have to route signal multiple times





Signal Retiming

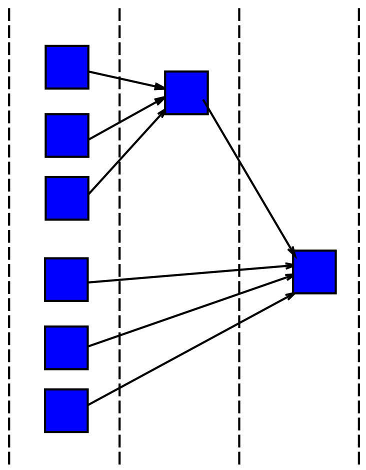
Input buffer chain
$\rightarrow$ **signal consumes a LUT every i cycles retimed**



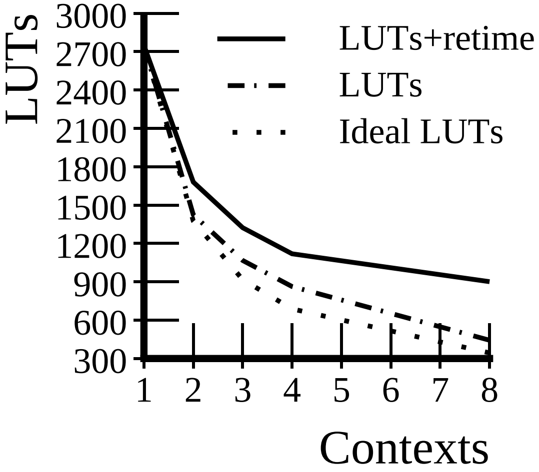
Signal Retiming

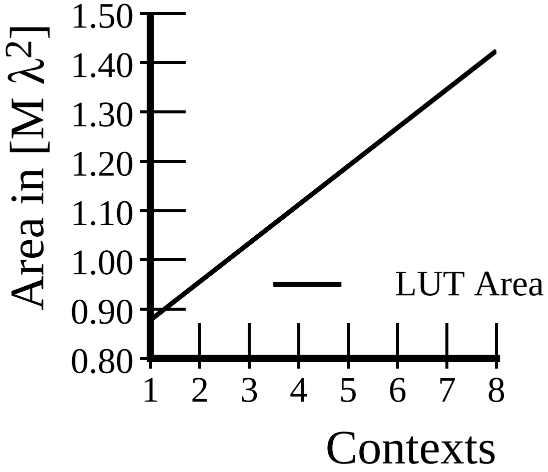
N.B. **Input buffer chain may only take k unique inputs per cycle**

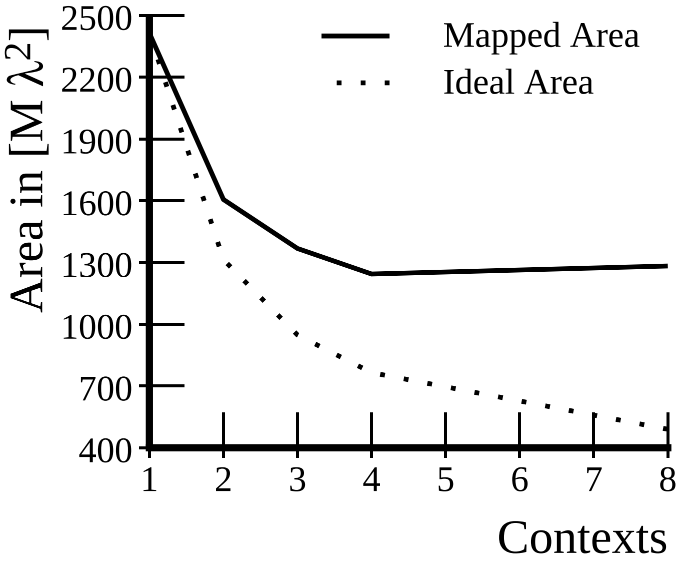
$\rightarrow$ limits which logical LUTs may share physical LUTs.



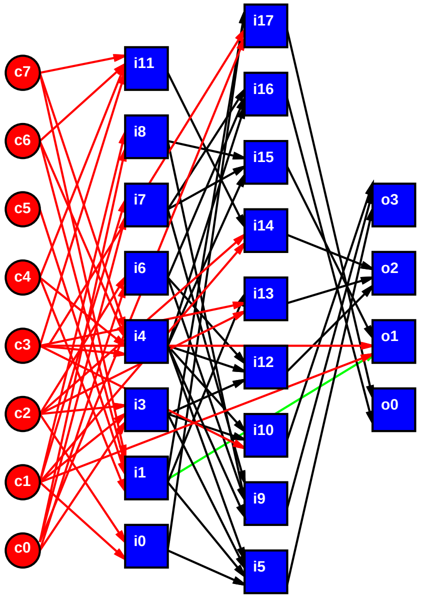
DES Latency Example





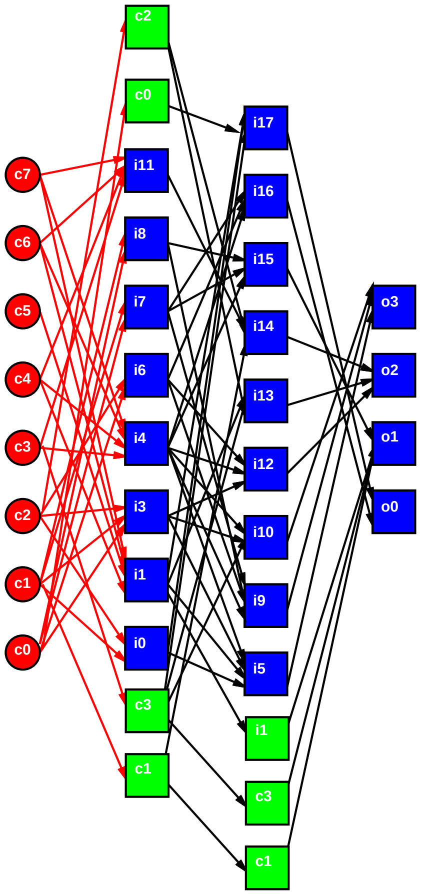


ASCII→Hex Example



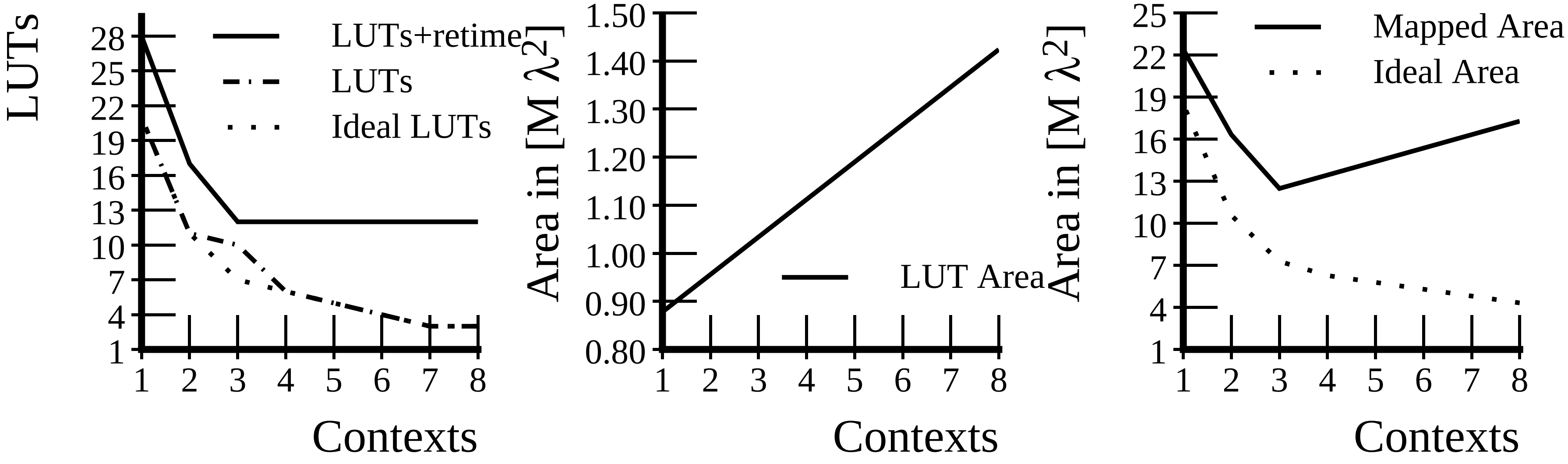
Single context, 21 LUTs at 880Kλ^2=18.5Mλ^2

ASCII→Hex Example



Three context, 12 LUTs at $1040K\lambda^2 = 12.5M\lambda^2$

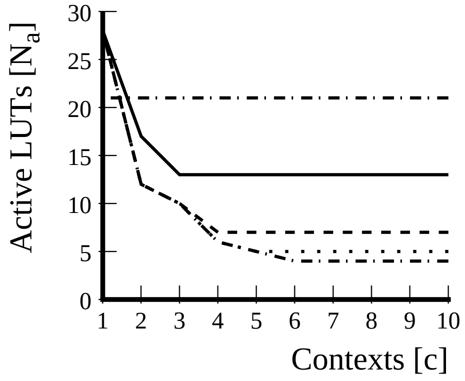
DPGA on ASCII→Hex Binary Conversion

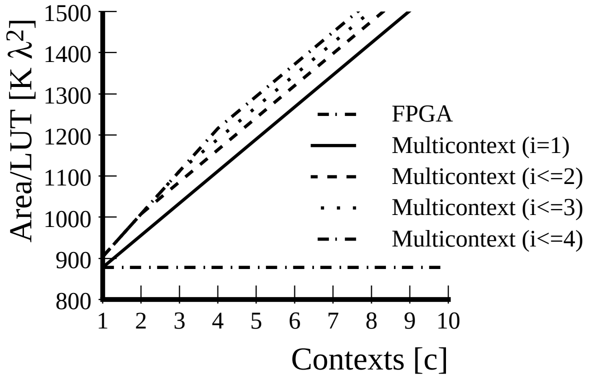


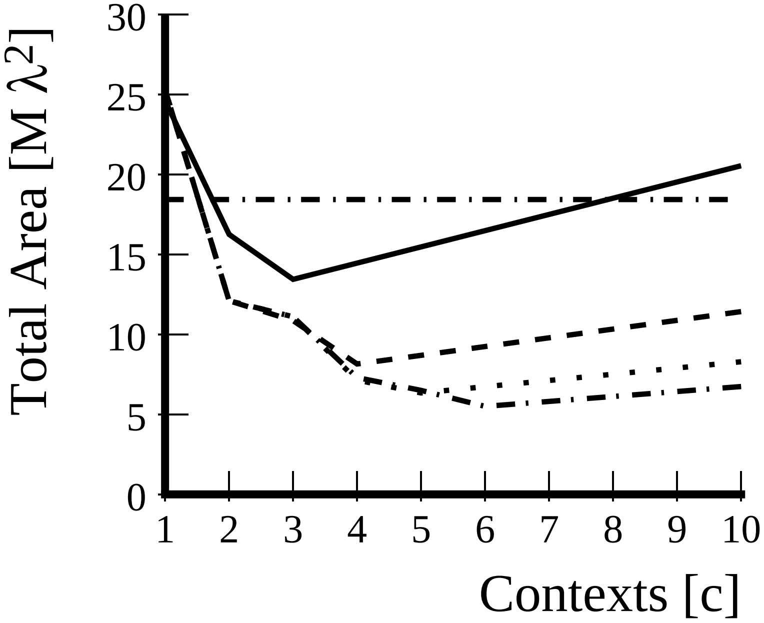
(Ideal Case ≡ Perfect packing and no retiming overhead)

N.B. **If all retiming must occur on wires (active outputs), we reach a point of saturation equal to the number of inputs needed by the largest stage.**

ASCII→Hex Example







At $i = 4, c = 6$ **area is 5.5**Mλ^2 **(compare 18.5**Mλ^2**)**

Throughput Mapping Experiment

- target producing a new result every t cycles
- spatially pipeline every t stages
- retime to minimize register requirements
- for multicontext cases, evaluate multicontext within a spatial pipeline stage
- retime to minimize resource usage
- unify LUT inputs

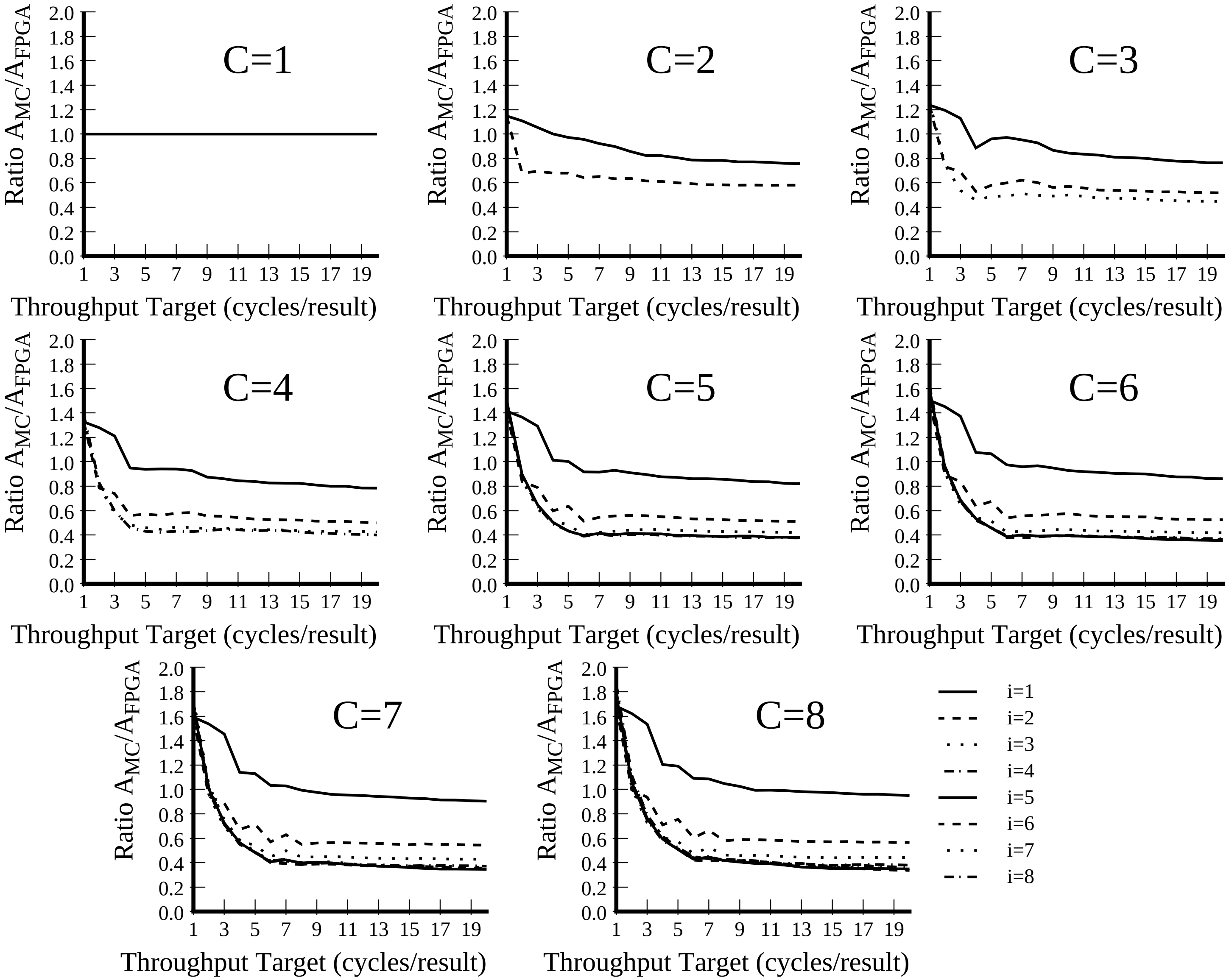
Map for various values of i and c.
Use area model to get area from number of LUTs.
Show area ratio.

Circuit Benchmarks

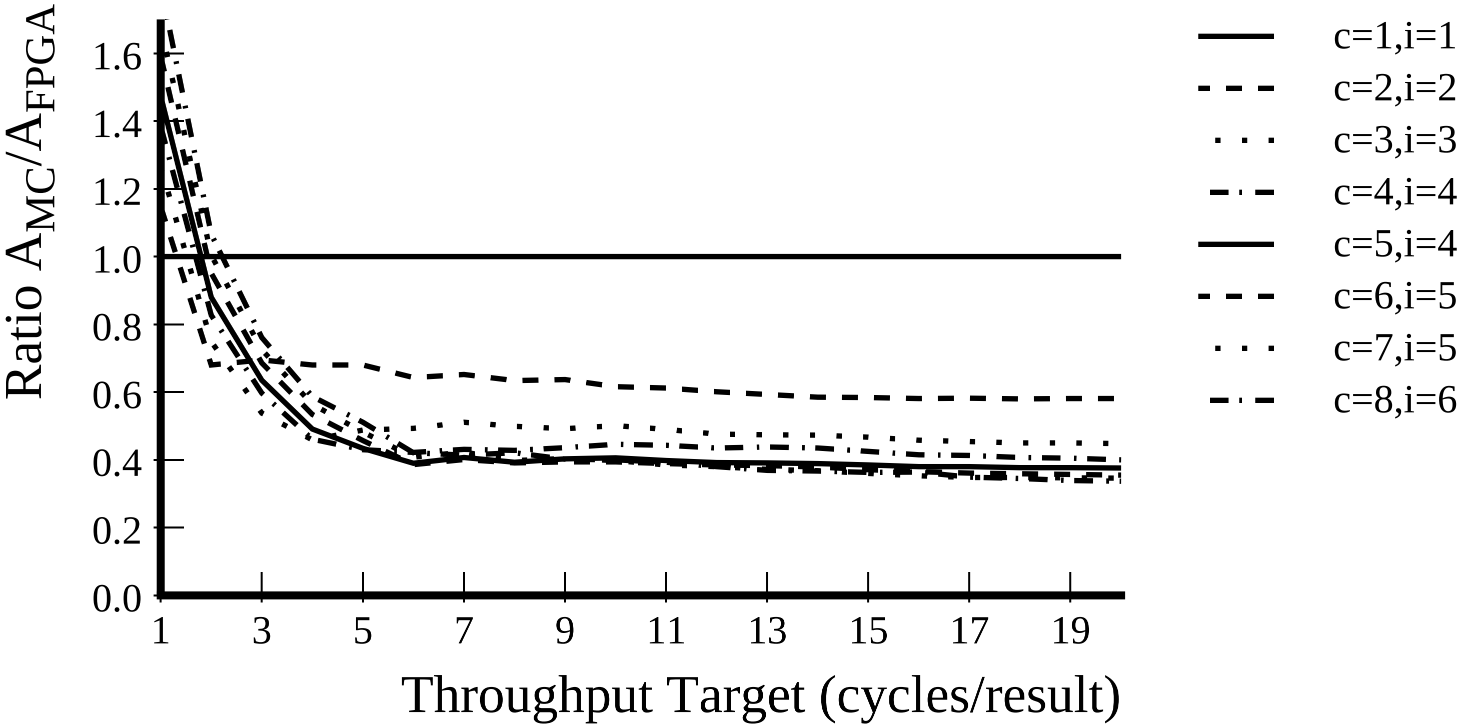
23 standard MCNC circuit Benchmarks
Area mapped with SIS and Chortle

Circuit	Mapped LUTs	Path Length	Circuit	Mapped LUTs	Path Length
5xp1	46	10	des	1267	13
9sym	123	7	e64	230	9
9symml	108	8	f51m	45	17
C499	85	10	misex1	20	6
C880	176	21	misex2	38	8
alu2	169	19	rd73	105	10
apex6	248	9	rd84	150	9
apex7	77	7	rot	293	16
b9	46	7	sao2	73	9
clip	121	9	vg2	60	9
cordic	367	13	z4ml	8	7
count	46	16			

DPGAs v/s Throughput on Circuit Benchmarks



DPGAs v/s Throughput on Circuit Benchmarks



Summary

- Reminder computing resources:
  instructions, active (compute, interconnect), retiming
- Applications with low through/cycles
  are instruction limited not active limited
- Since $A_{pinst} << A_{active}$, less area if share active among instructions
- As share wires, must retime data more economically
  $\rightarrow$ richer retiming resources
- $c =$4–8, $i =$4–6 automatically mapped designs
  half to one-third single context size
- In range where most FPGA designs run today
  ($c > 3$, $F <$50MHz) multicontext designs always smaller than single context