NANYANG TECHNOLOGICAL UNIVERSITY

An Area-Efficient FPGA Overlay Architecture using Time-multiplexed DSP Block based Functional Units

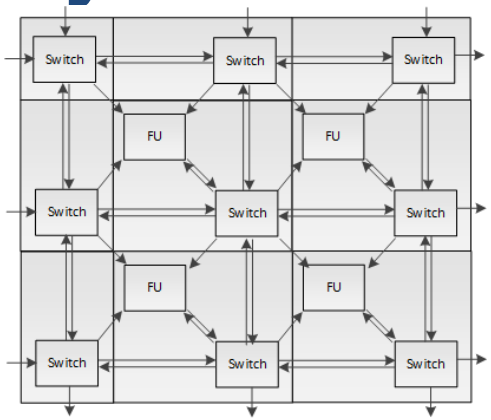
Xiangwei Li, Abhishek Jain, Douglas Maskell & Suhaib Fahmy †

School of Computer Engineering, NTU Singapore
† School of Engineering, University of Warwick, UK

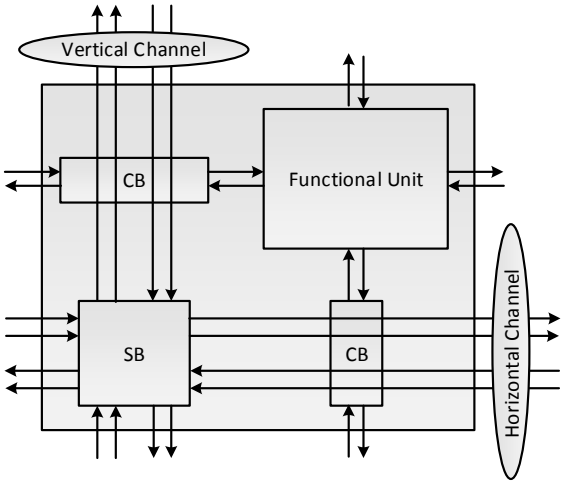
2nd International Workshop on Overlay Architectures for FPGAs (OLAF)
21st Feb 2016, Monterey, CA, USA

Coarse Grained FPGA Overlays

- Spatially Configured Overlays
- Throughput Oriented with an II of 1
- IF, DySER, DSP based overlay
- Resource hungry due to FU requirement for each operation and connection network
- Zynq-7020 can accommodate:
  - 6x6 DySER supporting up-to 36 OPs
  - 8x8 DSP based overlay supporting up-to 192 Ops
- No sharing of FUs among multiple operations to sustain high throughput
- So, can we share FUs to reduce area requirements at the cost of reduced throughput?



DySER Overlay

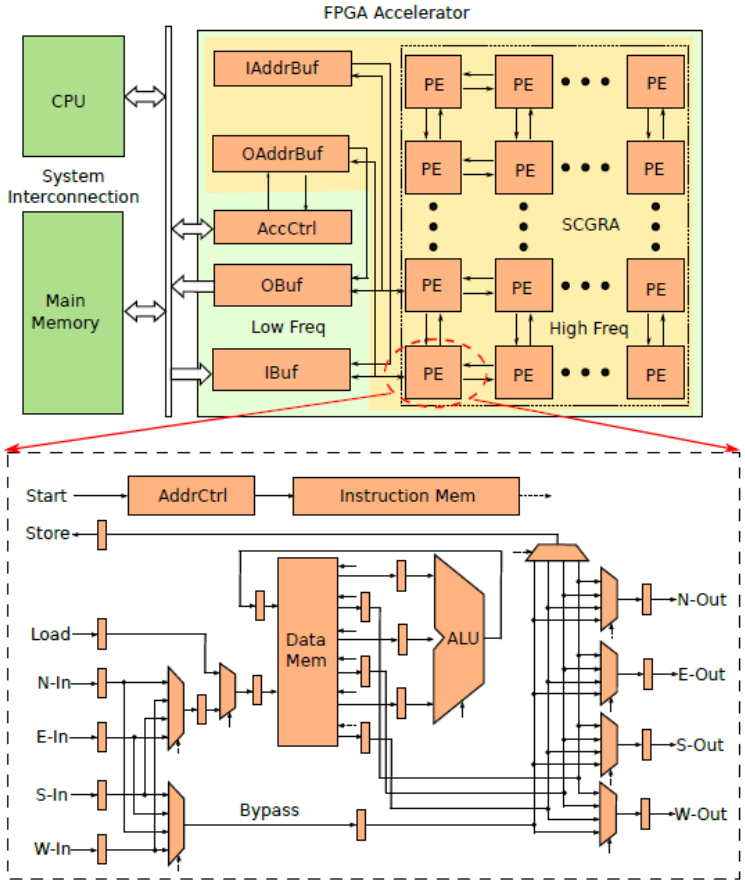


Island-style DSP Block based Overlay

NANYANG TECHNOLOGICAL UNIVERSITY

Time-multiplexing Functional Units

- Reduced FU requirements
- 5x5 SCGRA can fit on Zynq-7020
  - Limited Scalability due to instruction storage requirement
  - Need to store completely unrolled instruction stream in BRAMs
- reMORPH: Another similar overlay
  - Same problem of instruction storage
  - FU not really FPGA architecture friendly
- So, can we reduce the instruction storage requirements?



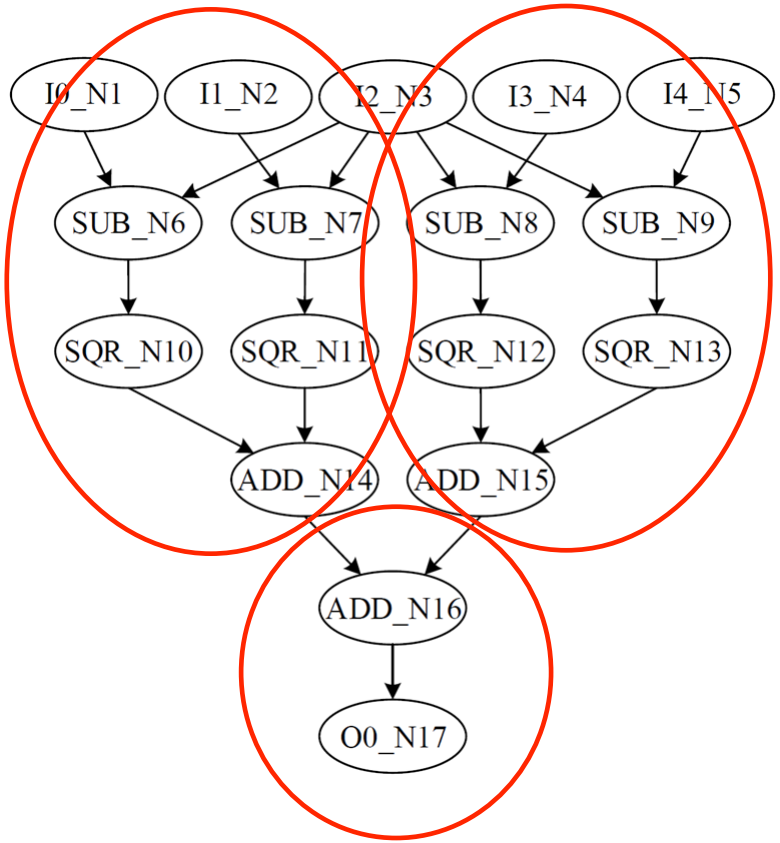
SCGRA Overlay

NANYANG TECHNOLOGICAL UNIVERSITY

Issues

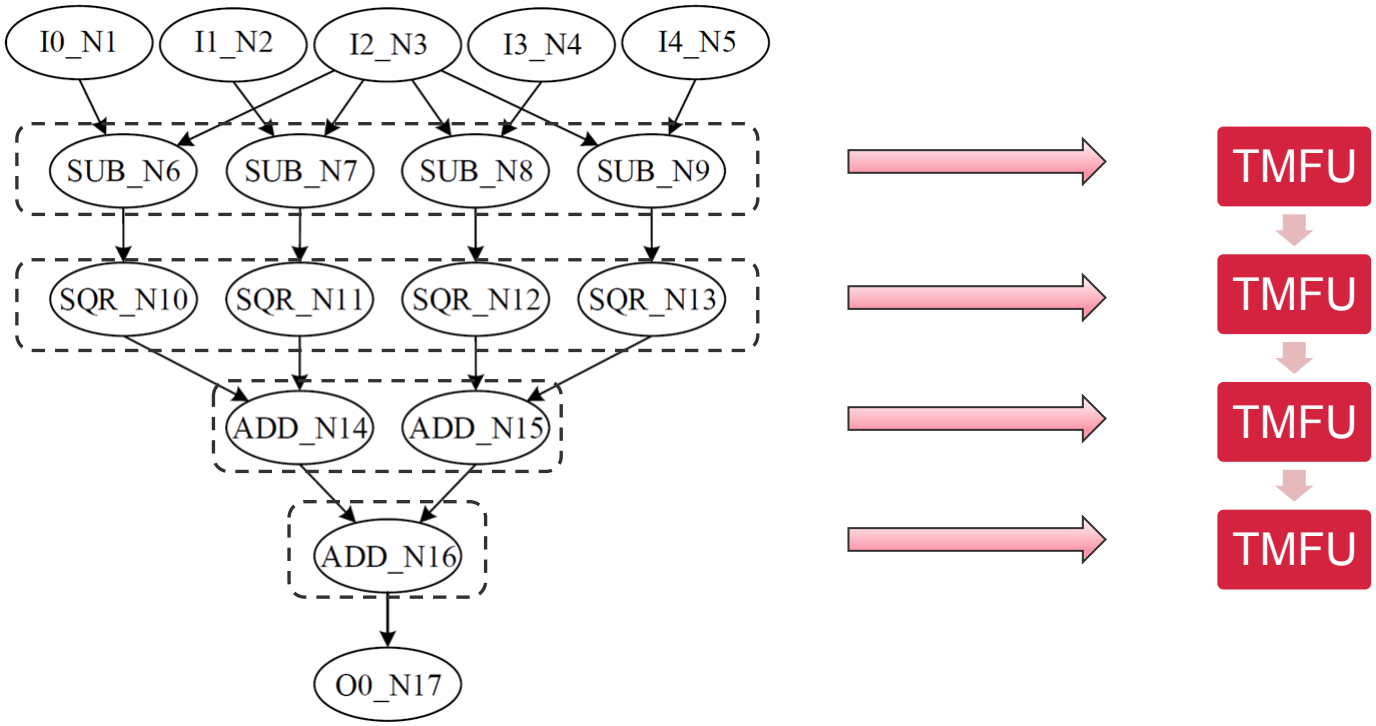
- Can we find a better scheduling strategy so that we avoid excessive BRAM utilization (while minimizing II)?
- How can we make the FU and routing network as area efficient as possible?
- Can we better optimize the FU's operating frequency to the FPGA devices?

Sharing FU among Kernel Operations



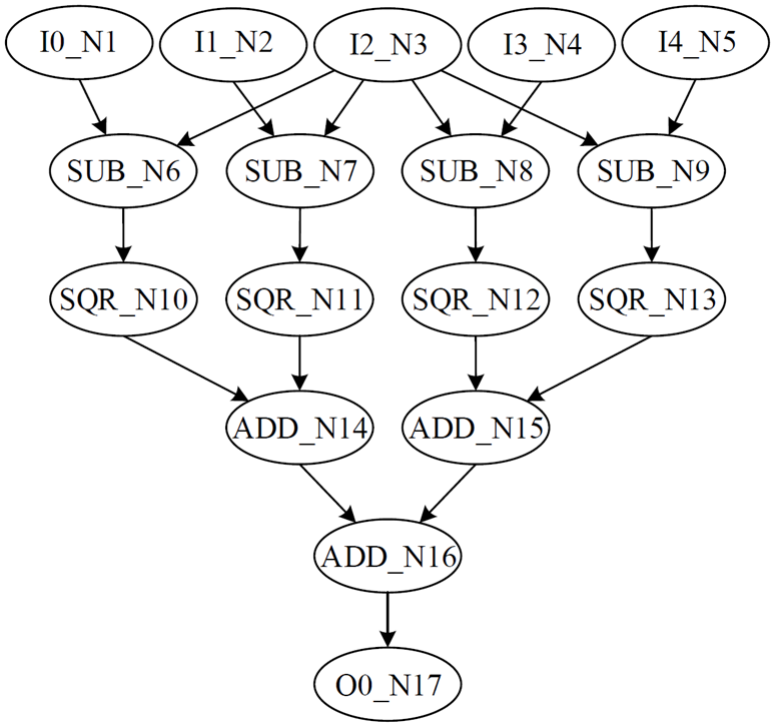
Sharing the FU among kernel operations can reduce the No. of FUs required.

Sharing FU among Kernel Operations



So instead, share the FU among the same scheduling stage kernel operations.
Can again reduce the No. of FUs.

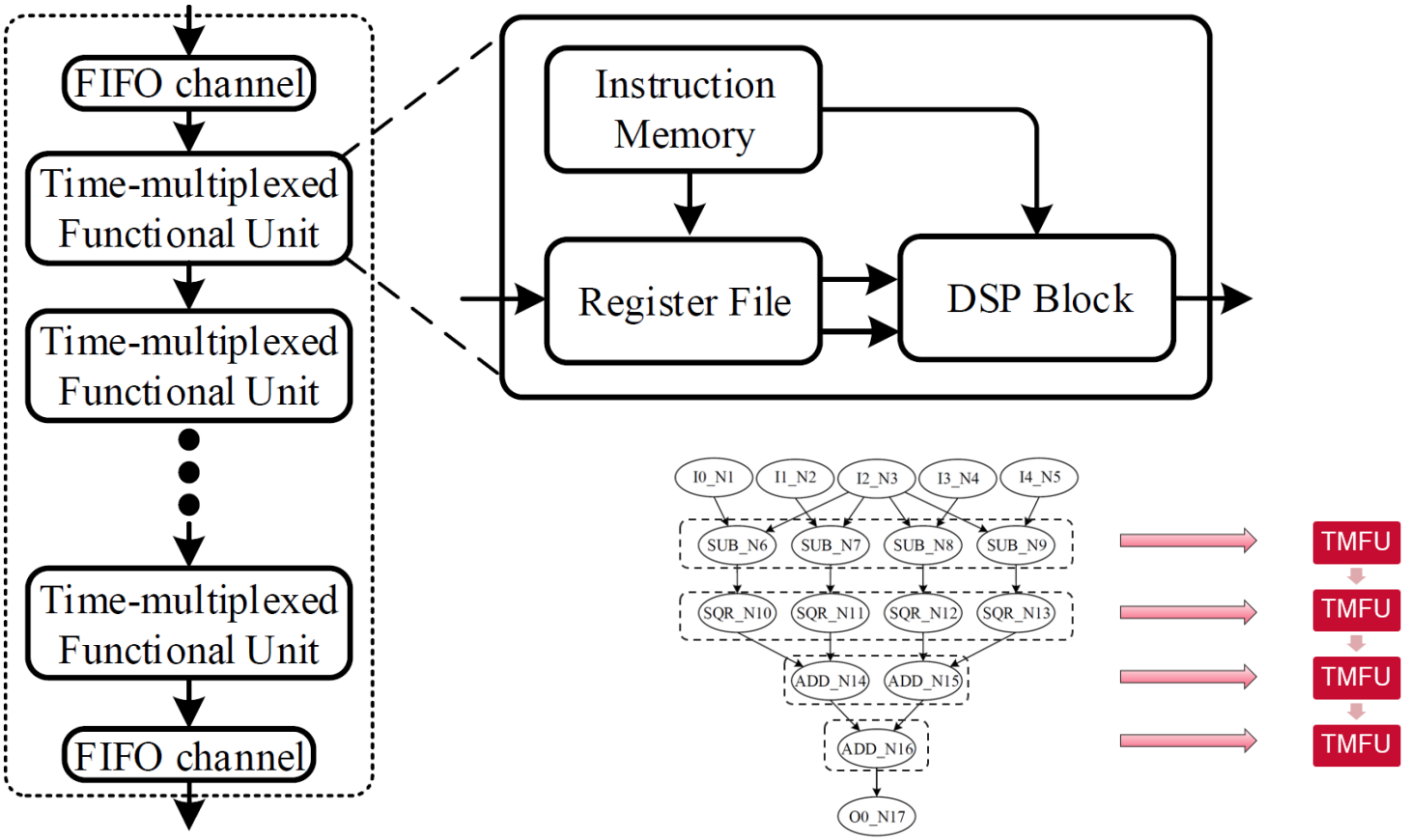
Instruction Scheduling



cycle	FU0	FU1	FU2	FU3
1	Load R0			
2	Load R1			
3	Load R2			
4	Load R3			
5	Load R4			
6	SUB (R0 R2)			
7	SUB (R1 R2)			
8	SUB (R2 R3)	Load R0		
9	SUB (R2 R4)	Load R1		
10		Load R2		
11		Load R3		
12	Load R0	SQR (R0 R0)		
13	Load R1	SQR (R1 R1)		
14	Load R2	SQR (R2 R2)	Load R0	
15	Load R3	SQR (R3 R3)	Load R1	
16	Load R4		Load R2	
17	SUB (R0 R2)		Load R3	
18	SUB (R1 R2)		ADD (R0 R1)	
19	SUB (R2 R3)	Load R0	ADD (R2 R3)	
20	SUB (R2 R4)	Load R1		Load R0
21		Load R2		Load R1
22		Load R3		ADD (R0 R1)
23	Load R0	SQR (R0 R0)		
24	Load R1	SQR (R1 R1)		
25	Load R2	SQR (R2 R2)	Load R0	
26	Load R3	SQR (R3 R3)	Load R1	
27	Load R4		Load R2	
28	SUB (R0 R2)		Load R3	
29	SUB (R1 R2)		ADD (R0 R1)	
30	SUB (R2 R3)	Load R0	ADD (R2 R3)	
31	SUB (R2 R4)	Load R1		
32		Load R2		Load R0

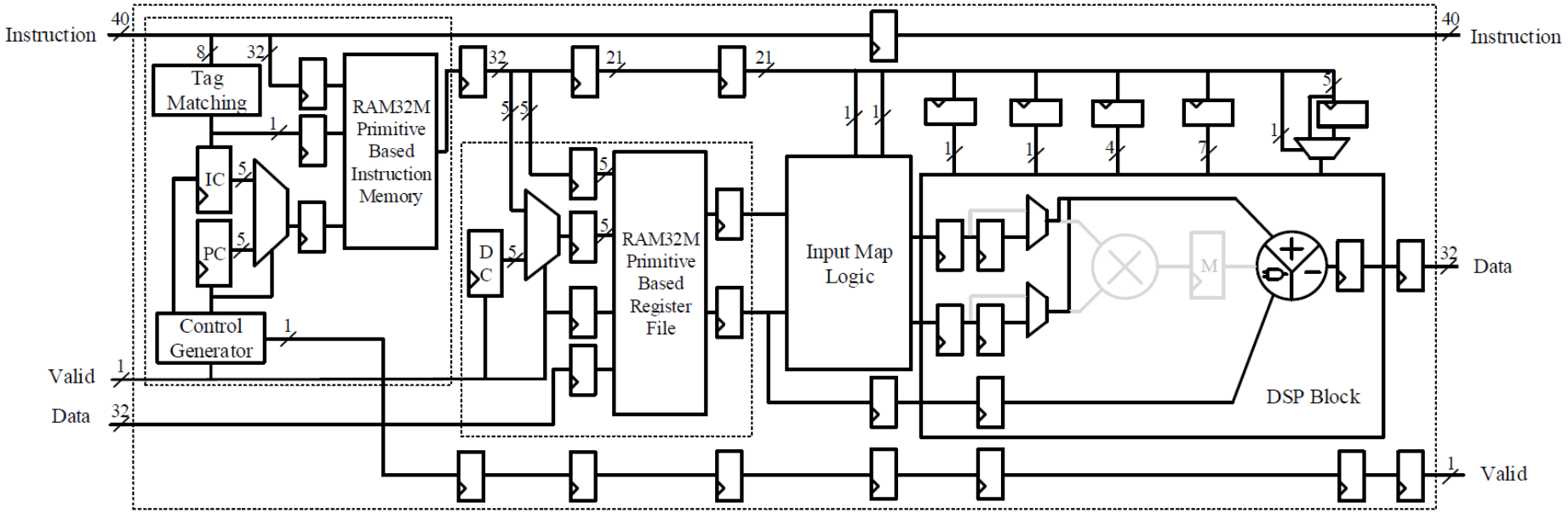
Initial Interval (II) = 11

An Overlay with Linear Interconnection



No need for switch box and connection box in comparison with island style interconnect!

Time Multiplexed FU



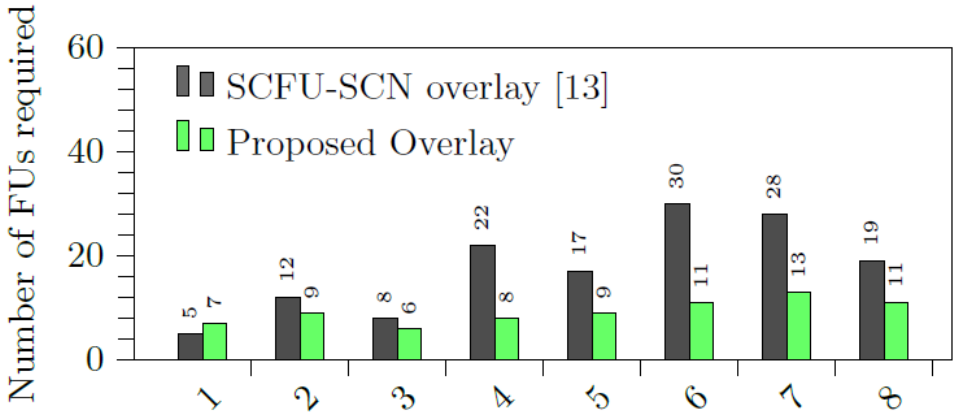
1 TMFU = 1 DSP + 160 LUTs + 293 FFs

Use RAM32M primitives for the instruction memory and register file instead of BRAMs

Can achieve up to 325 MHz on Zynq and 600 MHz on Virtex 7

Reduction in FU requirement

Benchmark Name	Characteristics						
	i/o nodes	graph edges	op nodes	graph depth	average parallelism	II	eOPC
chebyshev	1/1	12	7	7	1.00	6	1.2
sgfilter	2/1	27	18	9	2.00	10	1.8
mibench	3/1	22	13	6	2.16	11	1.2
qspline	7/1	50	26	8	3.25	18	1.4
poly5	3/1	43	27	9	3.00	14	1.9
poly6	3/1	72	44	11	4.00	17	2.6
poly7	3/1	62	39	13	3.00	17	2.3
poly8	3/1	51	32	11	2.90	15	2.1



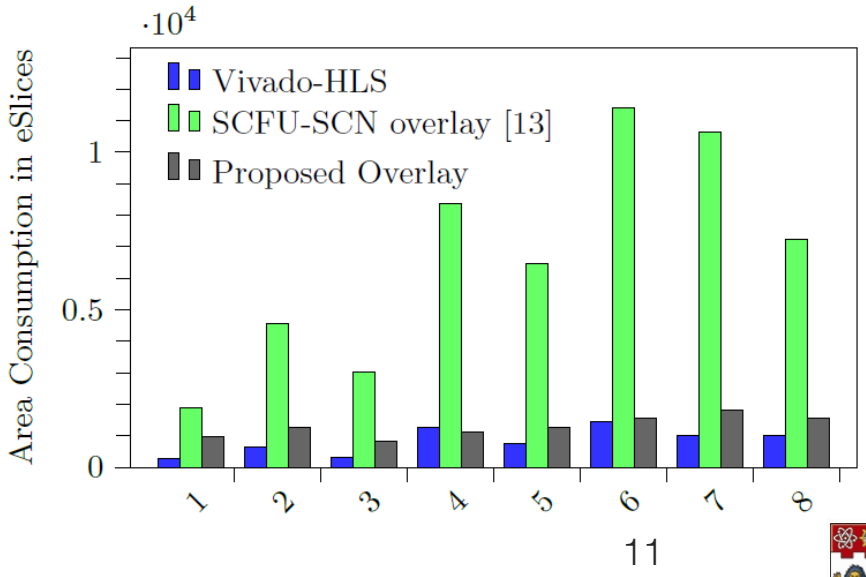
Proposed overlay can save up to 65% of FUs!

Area Reduction for a Set of Benchmarks

- Equivalent slices (e-Slices). The No. of slices per DSP block. ~60 on Zynq

- Proposed overlay can save up to 85% of e-Slices!!!

Benchmark Name	Characteristics						
	i/o nodes	graph edges	op nodes	graph depth	average parallelism	II	eOPC
chebyshev	1/1	12	7	7	1.00	6	1.2
sgfilter	2/1	27	18	9	2.00	10	1.8
mibench	3/1	22	13	6	2.16	11	1.2
qspline	7/1	50	26	8	3.25	18	1.4
poly5	3/1	43	27	9	3.00	14	1.9
poly6	3/1	72	44	11	4.00	17	2.6
poly7	3/1	62	39	13	3.00	17	2.3
poly8	3/1	51	32	11	2.90	15	2.1

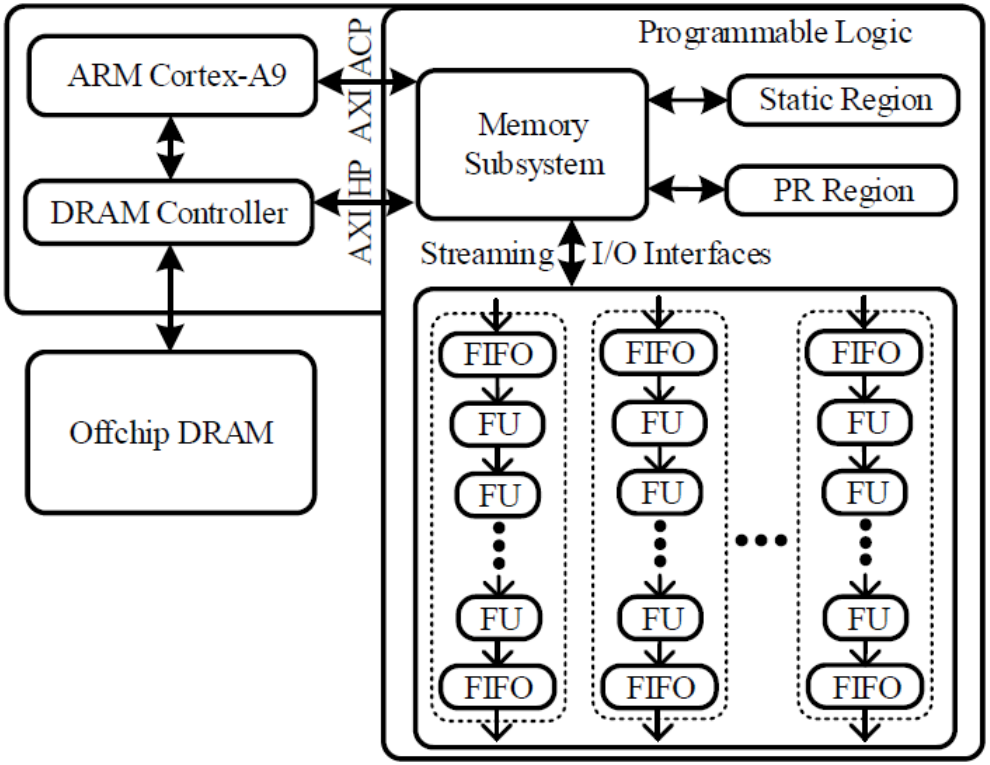


NANYANG TECHNOLOGICAL UNIVERSITY

Context Switch Time

	Configuration data	Context switch time
Proposed overlay	65-410 Bytes	0.27 us
SCFU-SCN overlay	323 Bytes	13 us
Vivado HLS	75K Bytes	200 us

Future Work: Cascading pipelines



A cascade of multiple processing pipelines helps reduce the value of II; thus improve the throughput.

Conclusion

- Benefits
  - Less FUs
  - Less interconnect
  - Less instructions
  - Fast context switch

- Limitations
  - Larger II
  - Feed forward data flow graphs only

Thank you!