IEEE Xplore® Digital Library

ISCAS 2020 Virtual, October 10-21

DEPLOYING CAS KNOWLEDGE TOWARDS SOCIETY GRAND CHALLENGES

IEEE INTERNATIONAL SYMPOSIUM ON CIRCUITS AND SYSTEMS

CAS

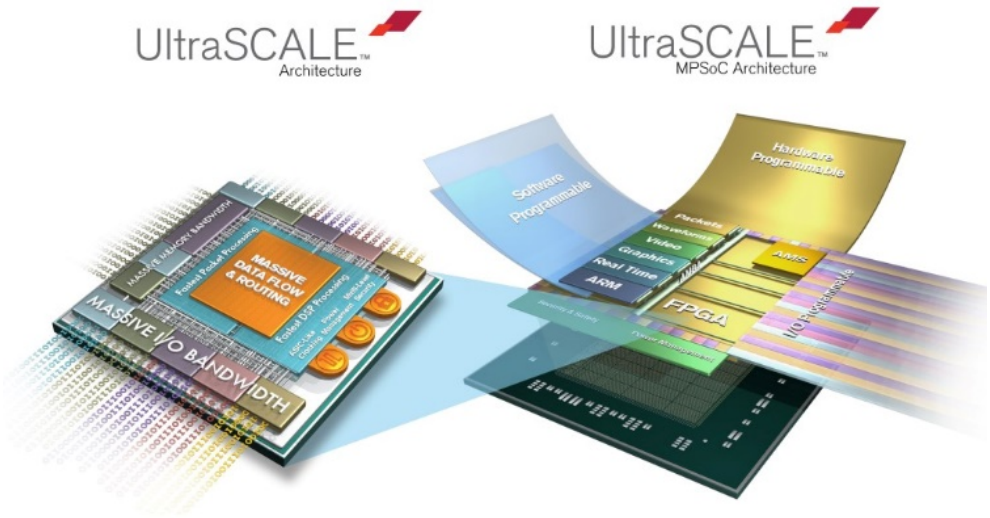
High Throughput Accelerator Interface Framework for a Linear Time-Multiplexed FPGA Overlay

Xiangwei Li[1], Kizheppatt Vipin[2], Douglas L. Maskell[3], Suhaib A. Fahmy[4] and Abhishek K. Jain[5]

[1]School of Electrical and Information Engineering, The University of Sydney, Australia
[2]School of Engineering and Digital Sciences, Nazarbayev University, Kazakhstan
[3]School of Computer Science and Engineering, Nanyang Technological University, Singapore
[4]School of Engineering, University of Warwick, United Kingdom
[5]Xilinx Inc., United States

2020 IEEE International Symposium on Circuits and Systems
Virtual, October 10-21, 2020

FPGA as Accelerator



- **Massive parallel processing units**
- **Low power (Vs. CPUs) & Reconfigurable (Vs. ASICs)**
- **Heterogeneous architecture**

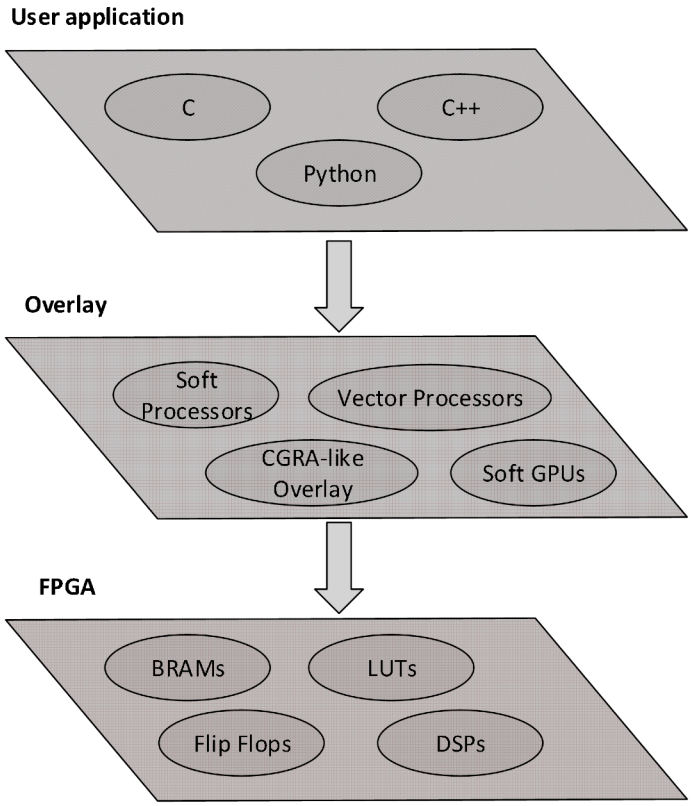
FPGA Design Productivity Issues

- **Low level of abstraction**
  - Register-transfer level (RTL) design
    - ❖Solved to some extent using High-level synthesis (HLS)
- **Complexity of SoC design**
  - CPU, GPU, hardware, interface, driver, OS support, …
    - ❖Vendors started developing SoC EDA tools
- **Lengthy hardware compilation time**
  - Fine-grained level placement and route

- FPGA overlays attempt to address all these issues

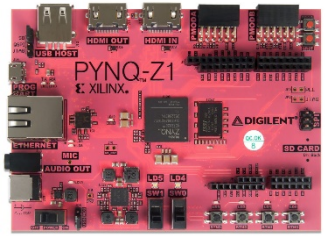
FPGA Overlays

- **A programmable coarse-grained hardware abstraction layer, implemented on top of an FPGA.**



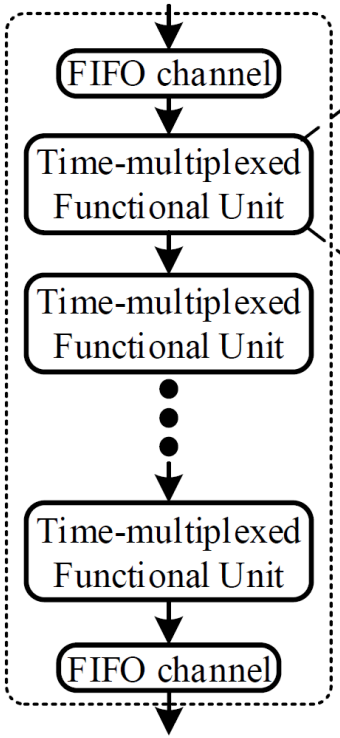
- **Programmability**
  - A higher level of abstraction
  - Just-in-time compilation/mapping
  - Fast kernel reconfiguration

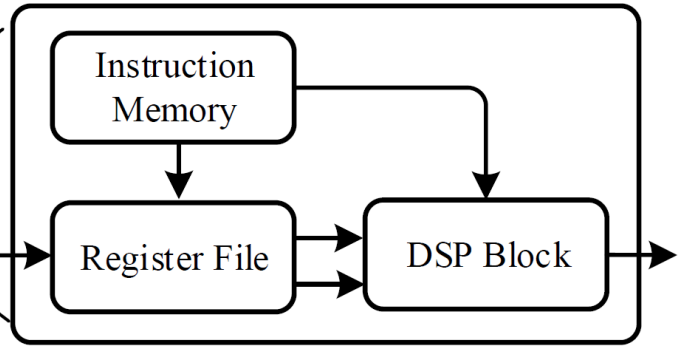
- **Industry support**
  - PYNQ [1] open source overlay



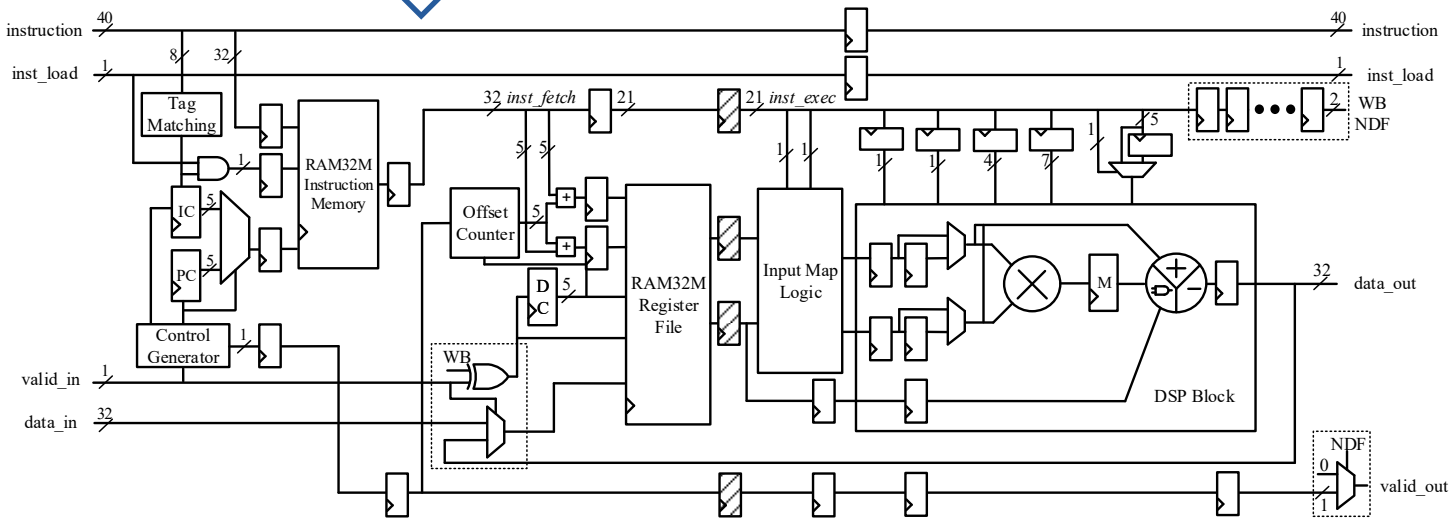
[1] Xilinx Ltd., "PYNQ: Python productivity for Zynq." [Online]. Available: http://www.pynq.io

A Linear TM Overlay [2]





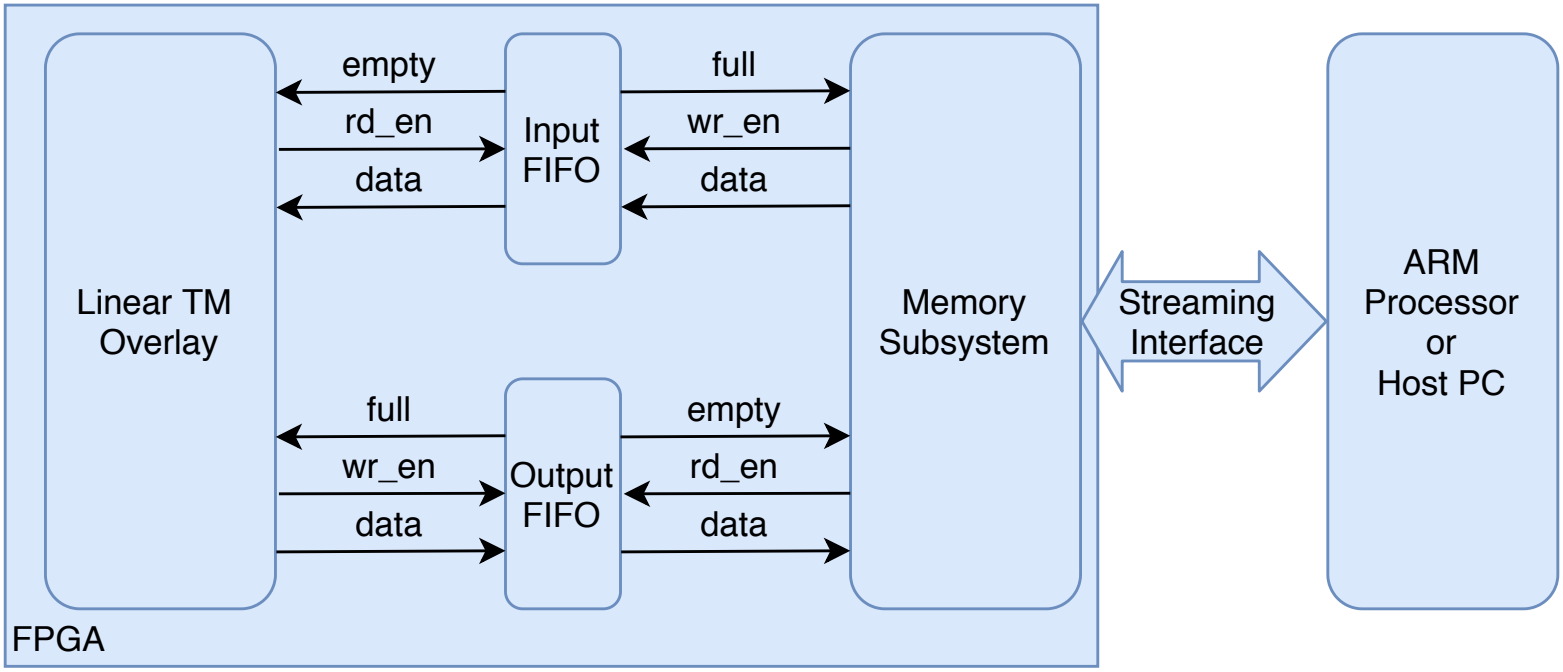
No need for switch box and connection box compared to a conventional array-based overlay.



Uses RAM32M primitives for the instruction memory and register file instead of BRAMs. FU = 1 DSP + 212 LUTs + 228 FFs. It achieves up to 325 MHz on Zynq and 600 MHz on V7.

[2] X. Li, et al. "A time-multiplexed FPGA overlay with linear interconnect." DATE'2018.

Overlay Accelerator System



Available Streaming Interfaces

Most of the existing FPGA solutions for memory interfaces fall into one of two categories:

- AXI bus-based solutions for FPGA SoC systems
- PCIe-based solutions for stand-alone systems

Interface	Frequency	Upstream BW	Downstream BW	Ports/Lanes	Total BW[1]
AXI HP	150 MHz	1200 MB/s	1200 MB/s	4	9600 MB/s
AXI ACP	150 MHz	1200 MB/s	1200 MB/s	1	2400 MB/s
PCIe Gen2	250 MHz	500 MB/s	500 MB/s	8	8000 MB/s
PCIe Gen3	250 MHz	984 MB/s	984 MB/s	8	15800 MB/s

[1] In a streaming interface, the throughput is limited by either the upstream or downstream BW, depending on the number of inputs or outputs, and so the maximum theoretical throughput would be half of the total bandwidth reported here.

Interfaces Bandwidth

AXI Interface

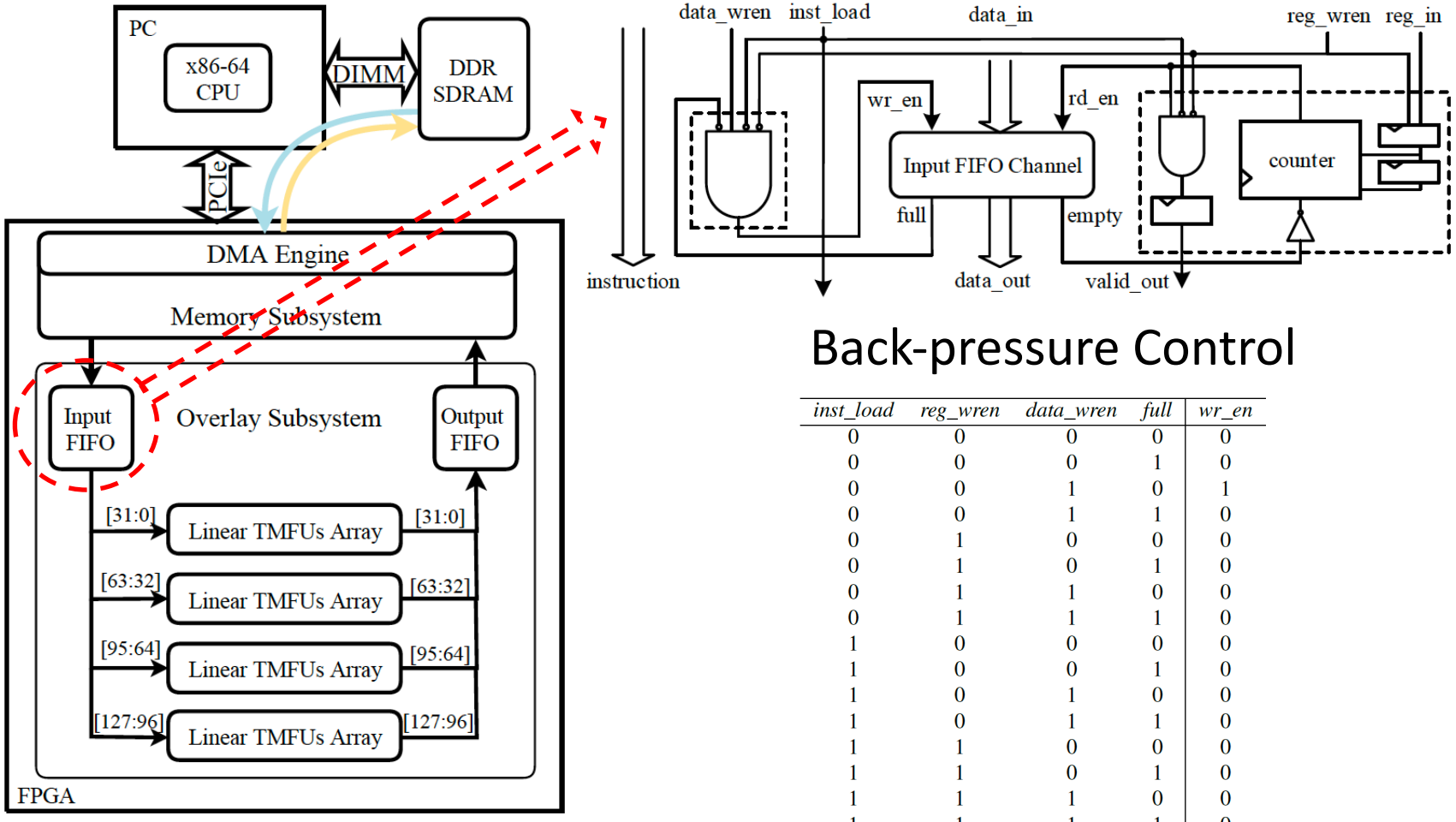
- Xillybus Version A [3]
  - 32-bit AXI ACP port, Max 350 MB/s
- VectorBlox MXP [4]
  - 64-bti AXI HP port, Max 275 MB/s

PCIe Interface

- Xillybus Revision XL [3]
- RIFFA [5]
- DyRACT [6]
  - PCIe Gen2x8, Gen3x4, Max 3.6 GB/s

[3] Xillybus Ltd., "IP core product brief." [Online]. Available: https://www.xillybus.com
[4] A. Severance, et al. "Embedded supercomputing in FPGAs with the VectorBlox MXP matrix processor," CODES+ISSS'2013.
[5] M. Jacobsen, et al. "RIFFA 2.1: A reusable integration framework for FPGA accelerators," ACM TRETS, 2015.
[6] K. Vipin, et al. "DyRACT: A partial reconfiguration enabled accelerator and test platform," FPL'2014.

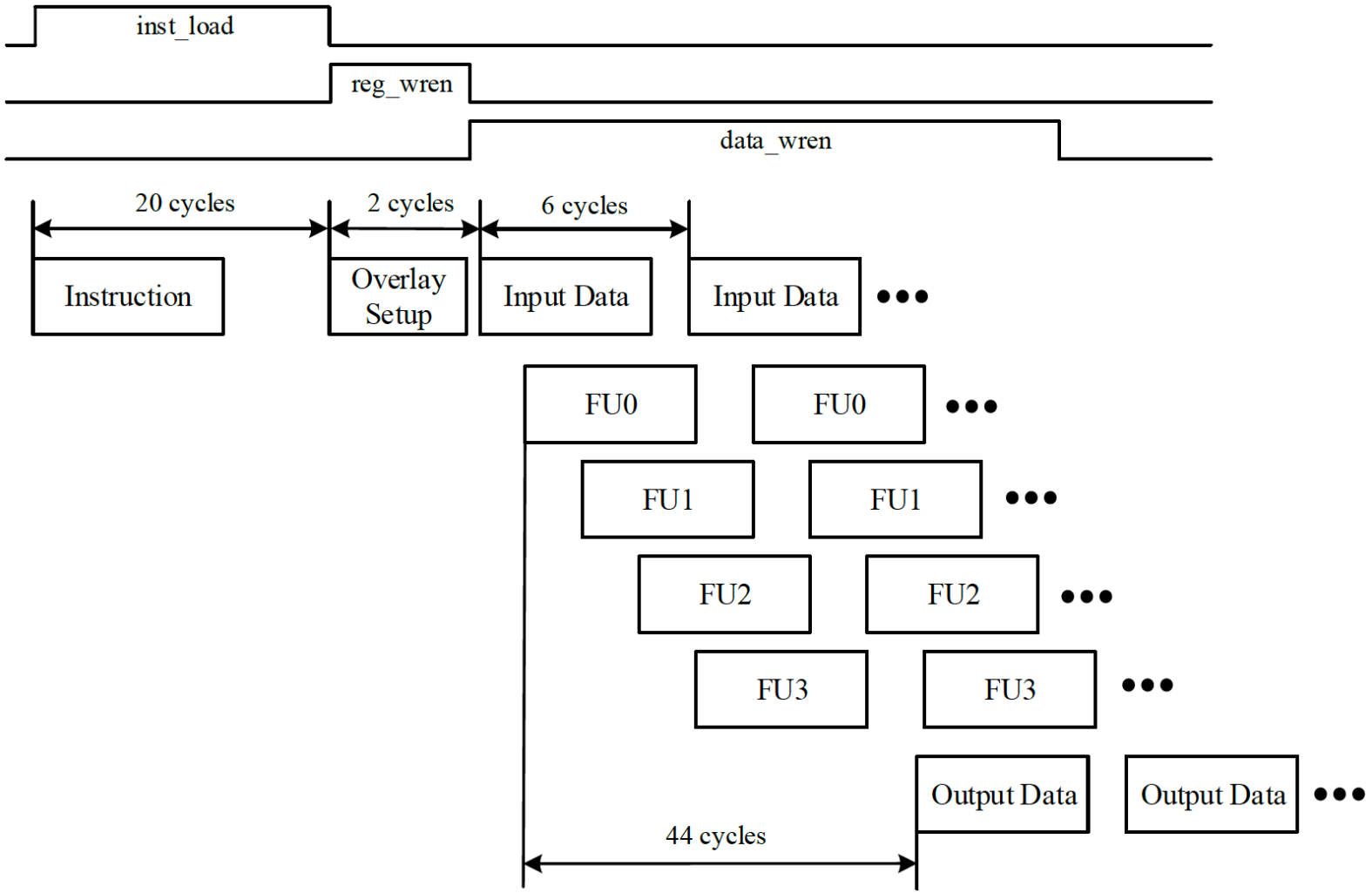
System Integration



Back-pressure Control

inst_load	*reg_wren*	*data_wren*	*full*	*wr_en*
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

Overlay Flow Chart

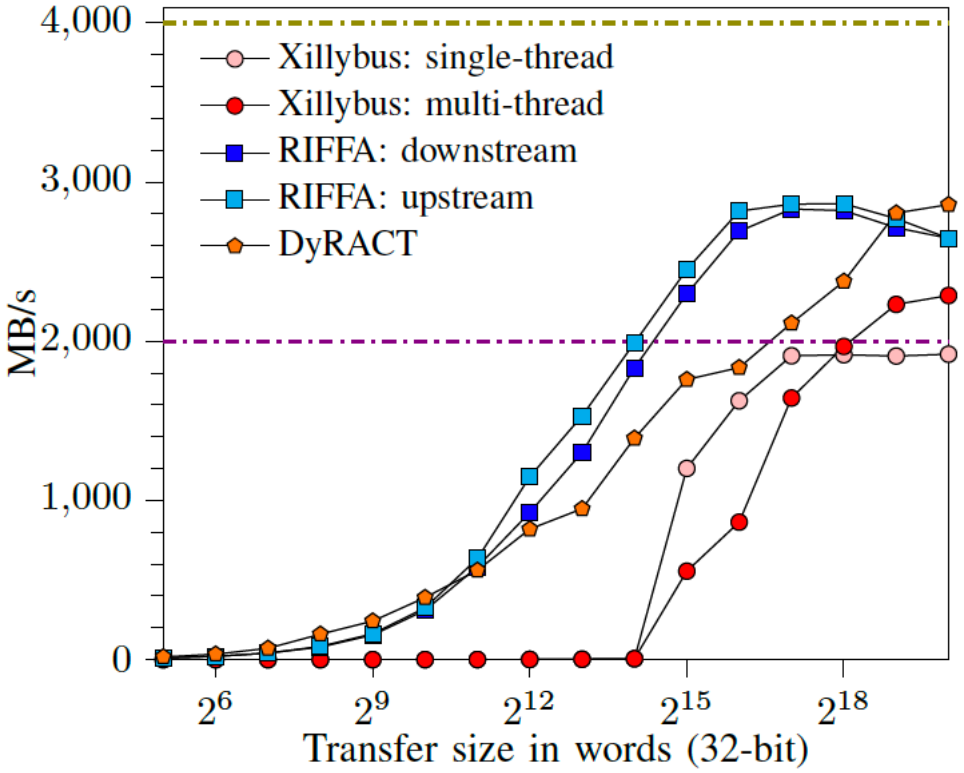


Programming Model

Example Instructions

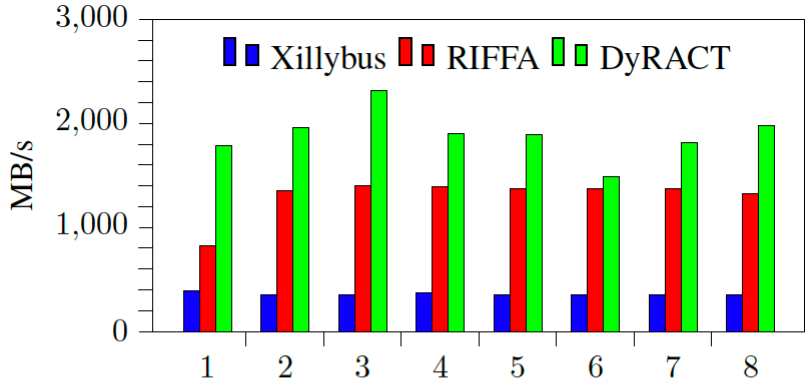
```
fpga_reg_wr(0x30, 0x0);                  // Tag of FU0
fpga_reg_wr(0x34, 0x3033D080); // Instruction 0
fpga_reg_wr(0x30, 0x1);                  // Tag of FU1
fpga_reg_wr(0x34, 0x8852000);    // Instruction 1
fpga_reg_wr(0x38, 5);                    // No. of input data
fpga_reg_wr(0x38, 5);                    // (II-1)
dyract_send_data((unsigned char *)mydata,
sendSize*sizeof(int));                   // Send data
dyract_recv_data((unsigned char *) recvdata,
recvSize*sizeof(4));                     // Receive data
```

Loopback Test Evaluation

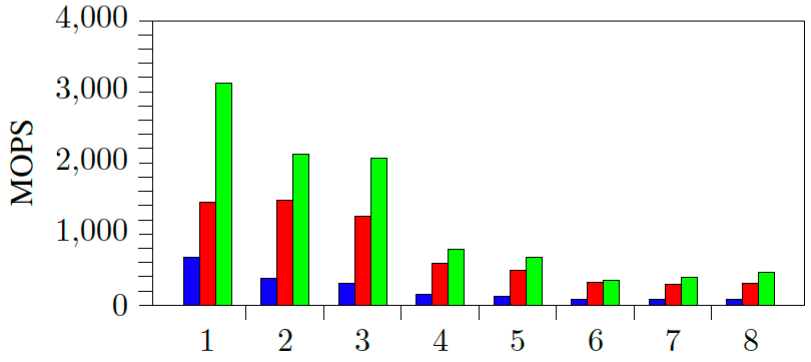


Xillybus: full-duplex system; saturates at 2300 MB/s in multithread mode
RIFFA: simplex system; read/write saturates at 2850 MB/s
DyRACT: full-duplex system; saturates at 2860 MB/s.

Benchmark Evaluation —> Throughput



(a) Data processing throughput in MB/s



(b) Throughput in MOPS

No.	Benchmark Name	Characteristics				
		I/O nodes	graph edges	op nodes	graph depth	graph width
1.	chebyshev	1/1	12	7	7	1
2.	mibench	3/1	22	13	6	3
3.	qspline	7/1	50	25	8	6
4.	fft	6/4	24	10	3	4
5.	kmeans	16/1	39	23	5	8
6.	mm	16/1	31	15	4	8
7.	spmv	16/2	30	14	4	8
8.	stencil	15/2	30	14	3	6

Xillybus: 358 MB/s

RIFFA: 1300 MB/s (3.6x higher than Xillybus)

DyRACT: 1892 MB/s (1.45x higher than RIFFA)

Benchmark Evaluation —> Area

Resource Usage on VC707 FPGA

System	Resource Usage			
	LUTs	FFs	BRAMs	DSPs
Xillybus	16,027	12,488	14.5	32
RIFFA	13,657	18,581	289.5	32
DyRACT	17,029	16,302	10	32
Available	303,600	607,200	1,030	2,800

Xillybus: most area efficient in FFs (33% less than RIFFA)
RIFFA: most area efficient in LUTs (20% less than DyRACT)
DyRACT: most area efficient in BRAM consumption (96% less than RIFFA)

Summary

- **Proposed three PCIe based high-performance overlay accelerator systems**
  - **Xillybus: most area efficient; lowest throughput (358 MB/s)**
  - **RIFFA: large overhead on BRAMs; moderate throughput (1300 MB/s)**
  - **DyRACT: area efficient; highest throughput (1892 MB/s)**
- **Designed a programming model for DyRACT based system**
  - **ANSI C based**
  - **Easy to use**

Future Work

- **Improving the memory interfaces**
  - Developing a full-duplex system for RIFFA
- **Python productivity for the overlays**
  - PYNQ extended overlay accelerator

Thank you!