From Streaming Models to Hardware Implementations

Arkadeb Ghosal, Rhishikesh Limaye and the NI Berkeley team

National Instruments, Berkeley, USA

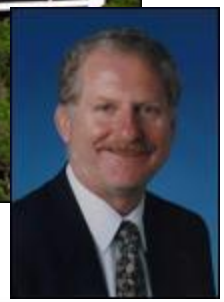
DREAMS Seminar, UC Berkeley, Berkeley, CA
November 7, 2011

Agenda

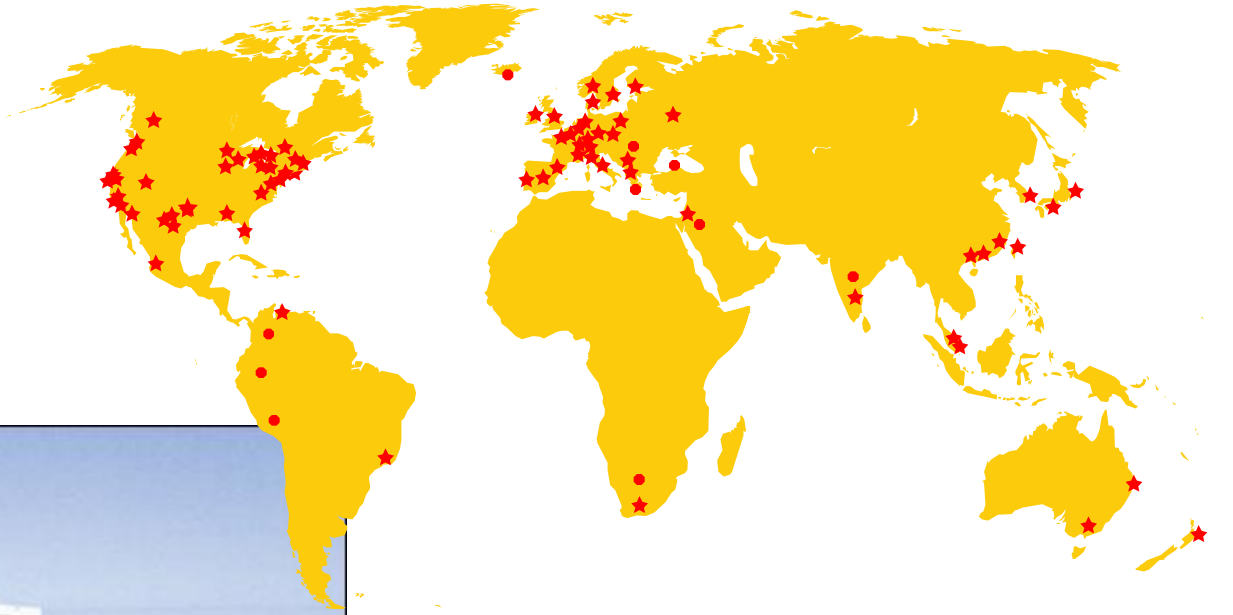
- **Background – NI**
- DSP Designer
- Models, Analysis and Exploration
- Model Extensions
- Looking Forward

National Instruments

- More than 45 international branches
- Corporate headquarters in Austin, TX



Dr. James Truchard

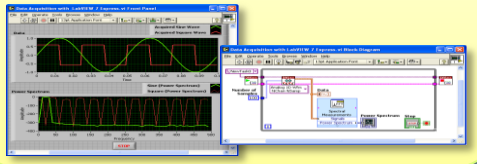


- 5,000+ employees
- 1,500+ engineers
- More than 1,000 products

What We Do

Low-Cost Modular Measurement and Control Hardware

Productive Software Development Tools



Highly Integrated Systems Platforms

Telecor

Automotive

Semiconductors

Electronics

Computers

ATE

Military/Aerospace

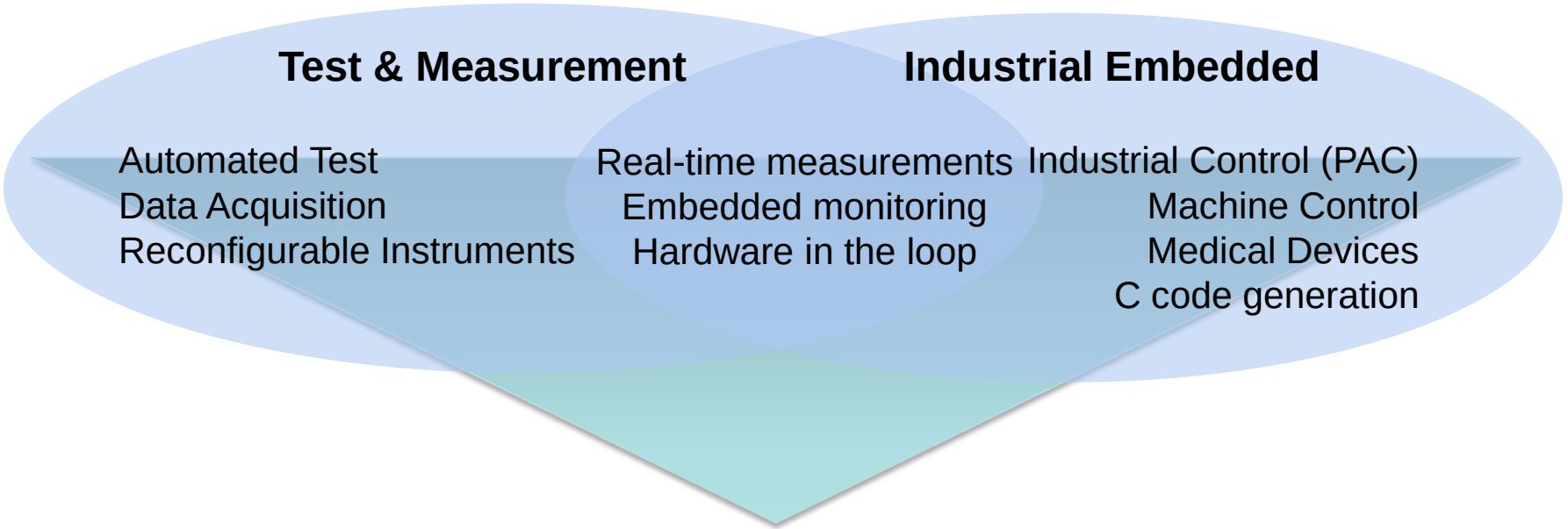
Advanced Research

Petrochemical

Food Processing

Textiles

Graphical System Design



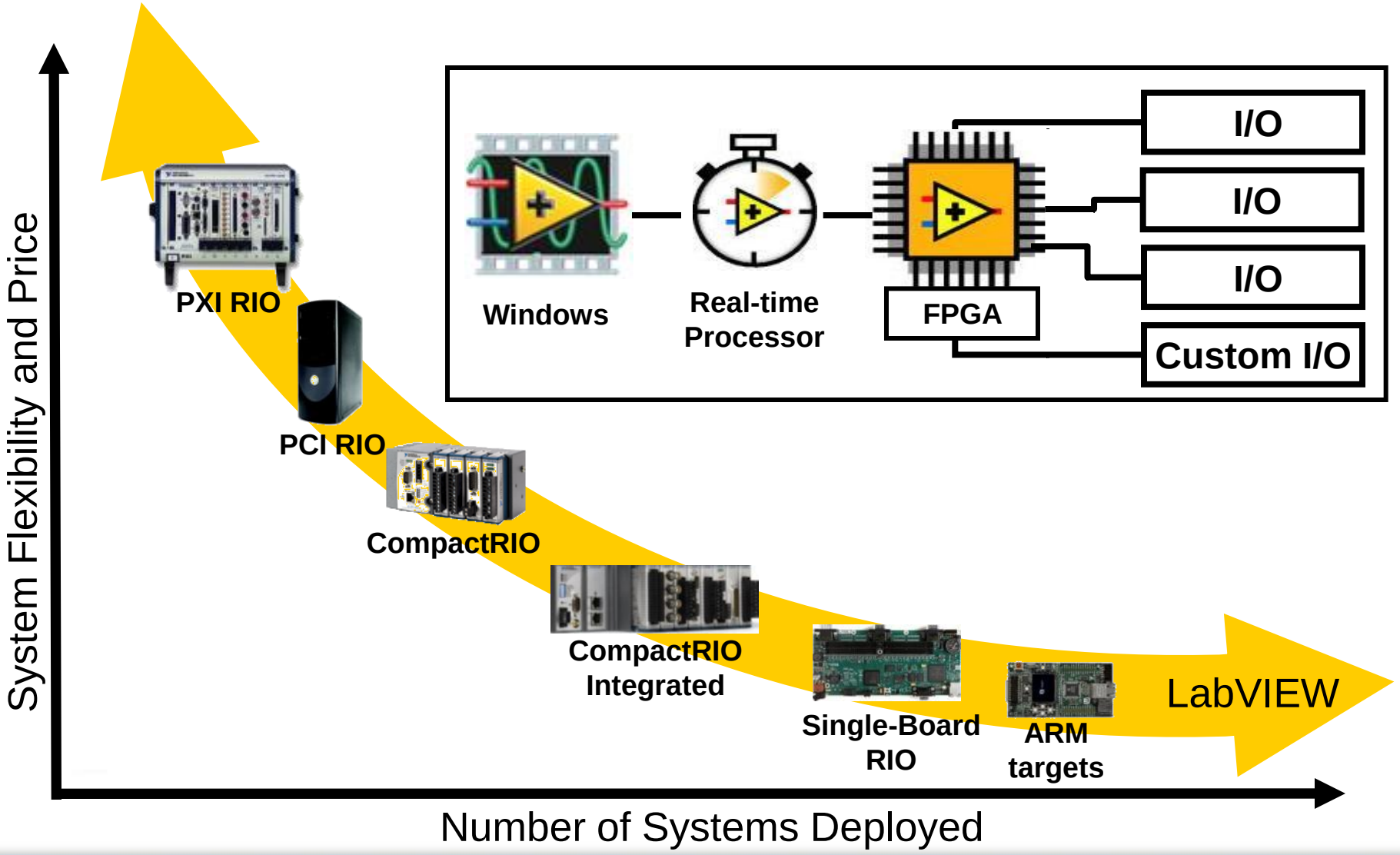
Hardware and Software Integration differentiate our solution

"To do for test and measurement what the spreadsheet did for financial analysis."

"To do for embedded what the PC did for the desktop."

"To do for embedded what the PC did for the desktop."

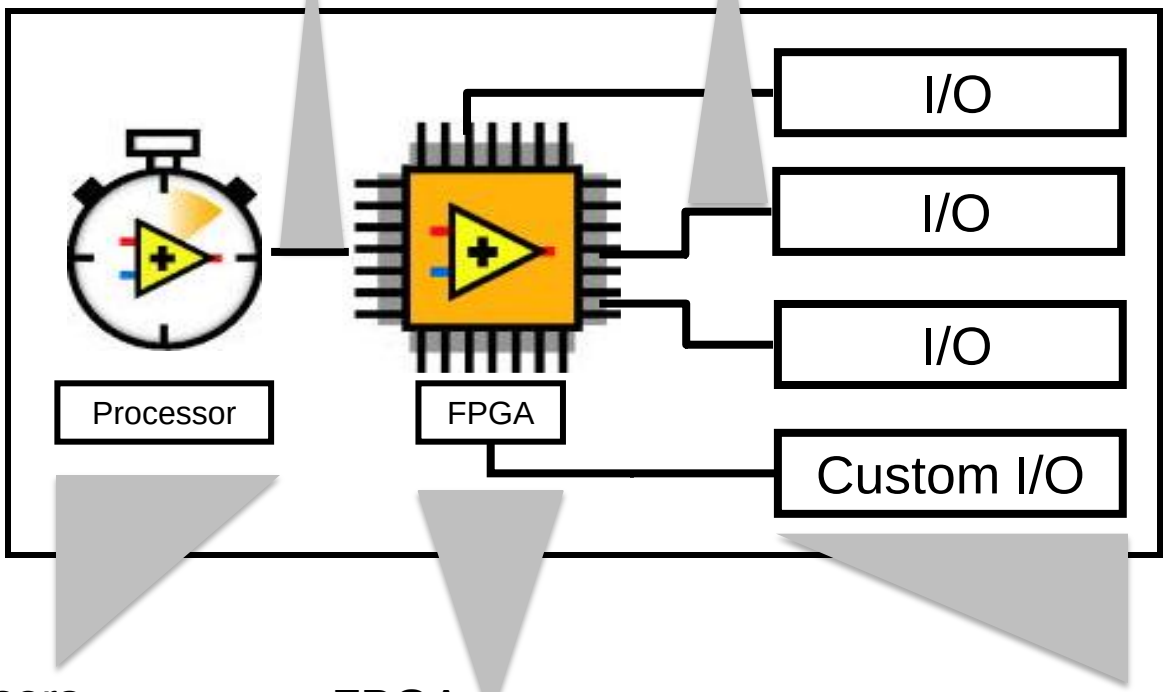
Deployment in Embedded Domain



Deployment Architecture

High-Speed Data Streaming
- Synchronize memory access
- Fast data links for maximum performance

A/D Technology
- Multirate sampling
- Individual channel triggering

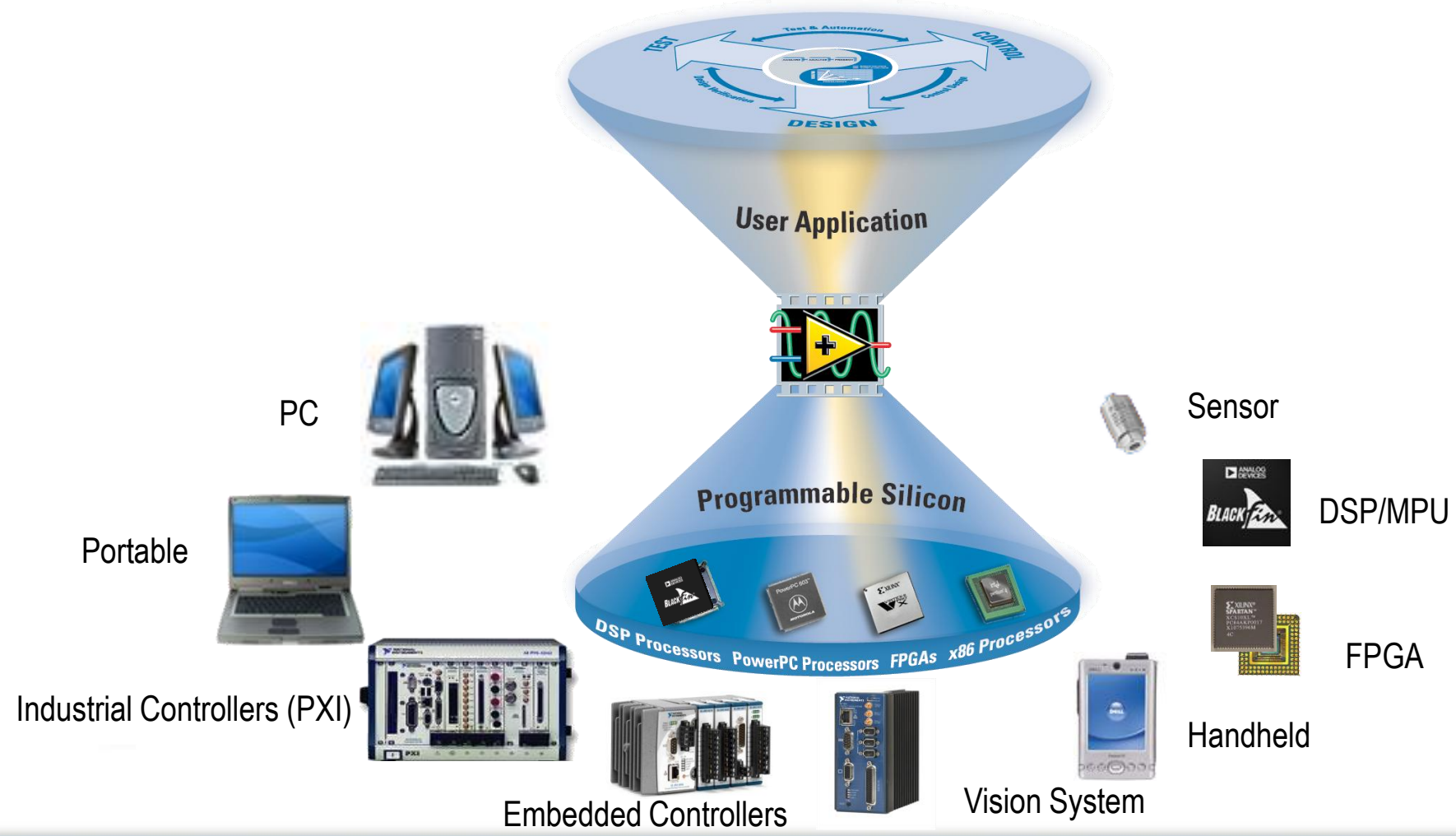


Microprocessors
- Floating-point processing
- Communications
- Multicore technology
- Reprogrammable

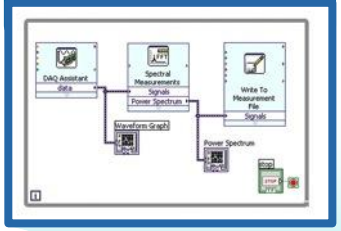
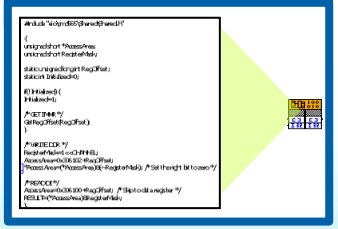
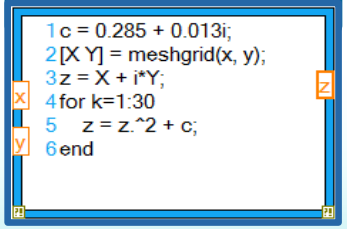
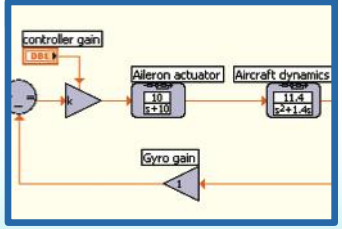
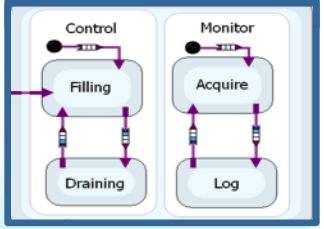
FPGAs
- High-speed control
- High-speed processing
- Reconfigurable
- True Parallelism
- High Reliability

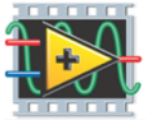
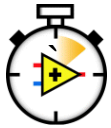
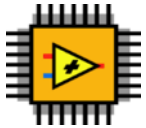
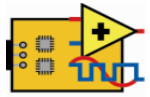
I/O
- Custom timing & triggering
- Modular I/O
- Calibration
- Custom modules

Graphical System Design Platform



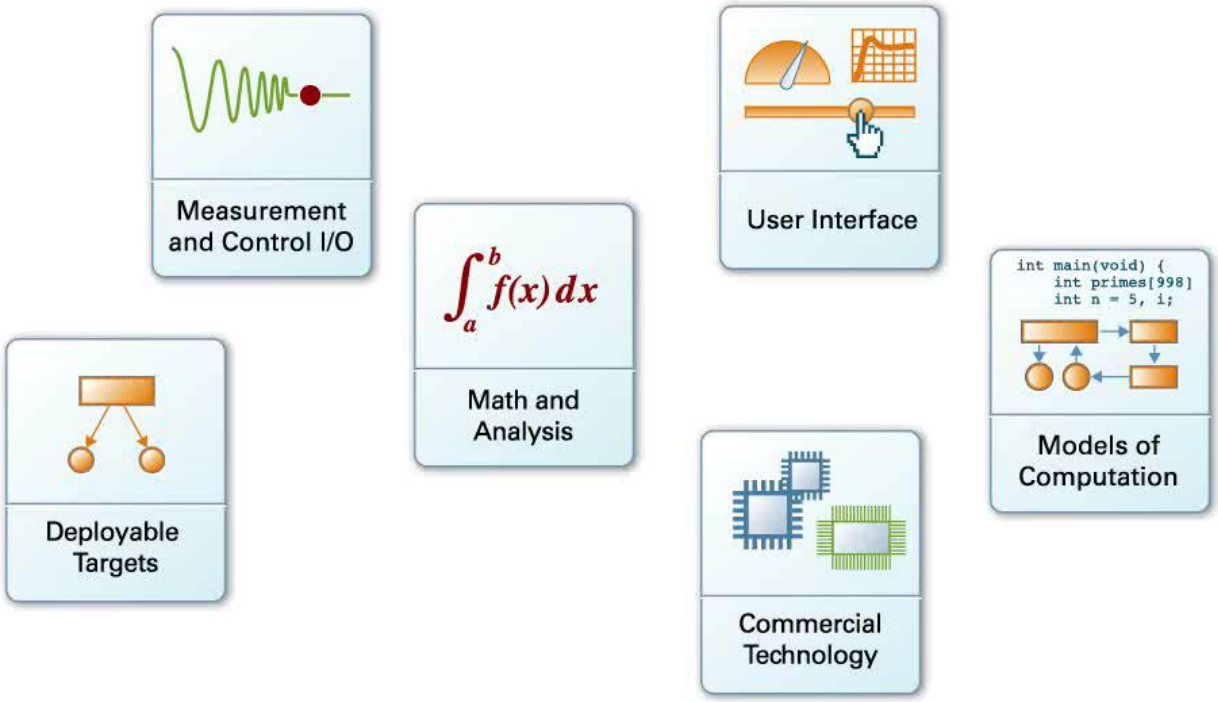
System Design to Deployment

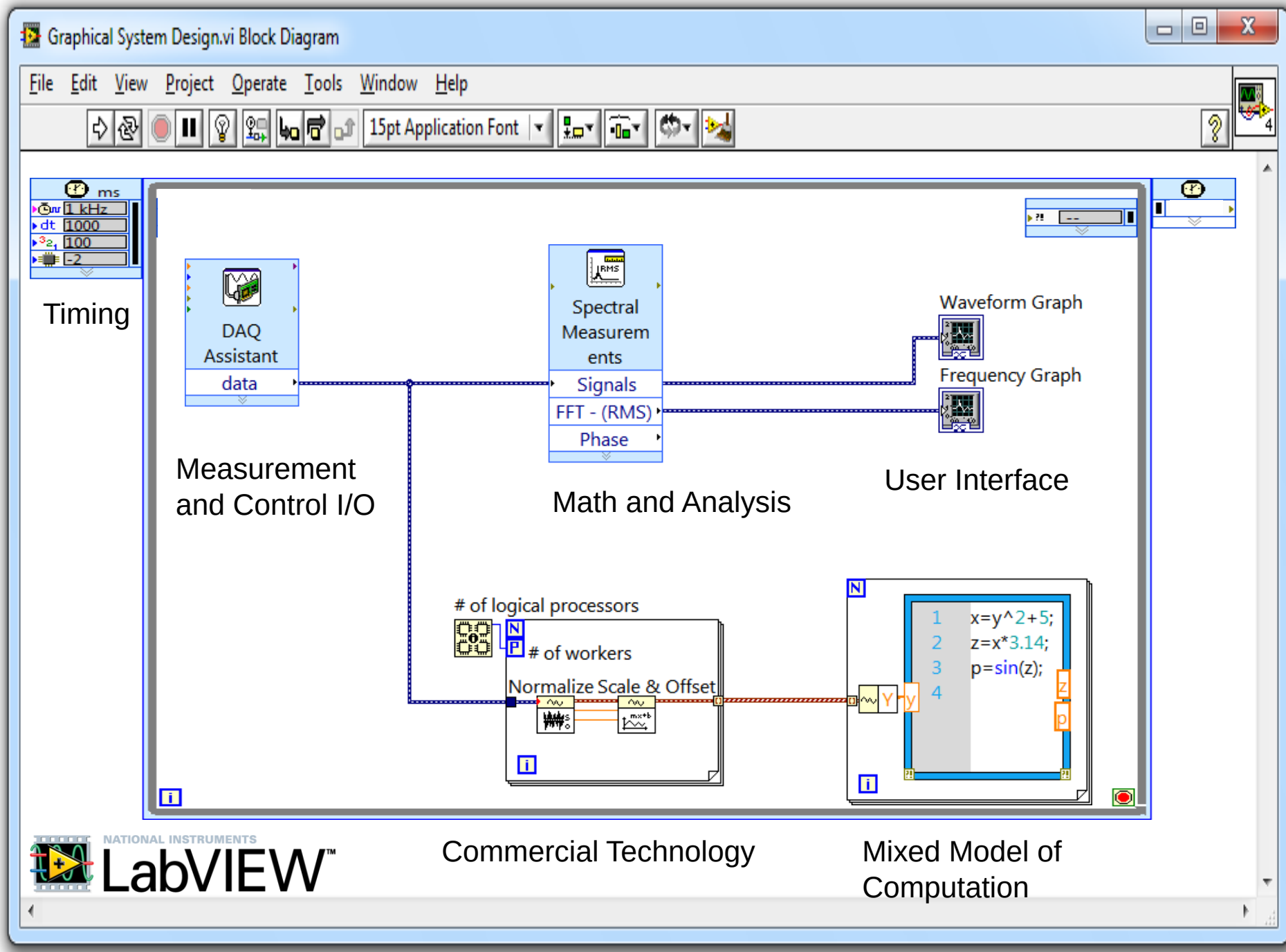
Dataflow	C / HDL Code	Textual Math	Simulation	Statechart
				

LabVIEW	LabVIEW	LabVIEW	LabVIEW
			
Desktop	Real-Time	FPGA	MPU/MCU

Personal Computers	PXI Systems	CompactRIO	Single-Board RIO	Custom Design
				

Integrating Elements



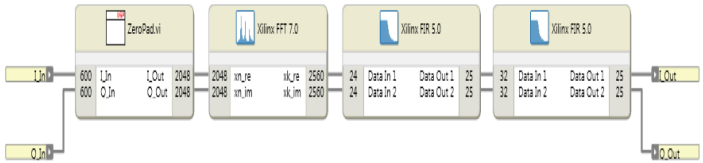

Graphical System Design.vi Block Diagram
File Edit View Project Operate Tools Window Help
15pt Application Font
ms
1 kHz
dt 1000
100
-2
Timing
DAQ Assistant
data
Measurement and Control I/O
Spectral Measurements
Signals
FFT - (RMS)
Phase
Math and Analysis
Waveform Graph
Frequency Graph
User Interface
# of logical processors
# of workers
Normalize Scale & Offset
x=y^2+5;
z=x*3.14;
p=sin(z);
Commercial Technology
Mixed Model of Computation
NATIONAL INSTRUMENTS
LabVIEW™

Agenda

- Background – NI
- **DSP Designer**
- Models, Analysis and Exploration
- Model Extensions
- Looking Forward

Motivation

Concurrent Application

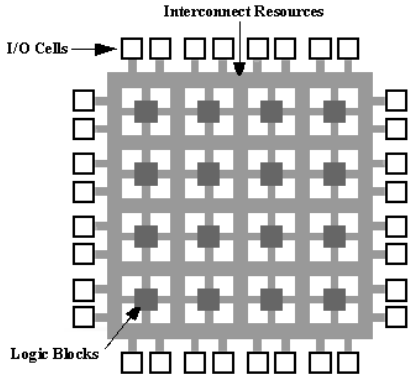


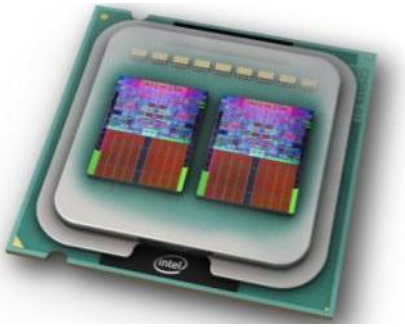
Application trends

- **1000's of parallel tasks**
- Large node/channel counts
- High performance requirements
- E.g. streaming DSP applications

Implementation Gap

Parallel Platform





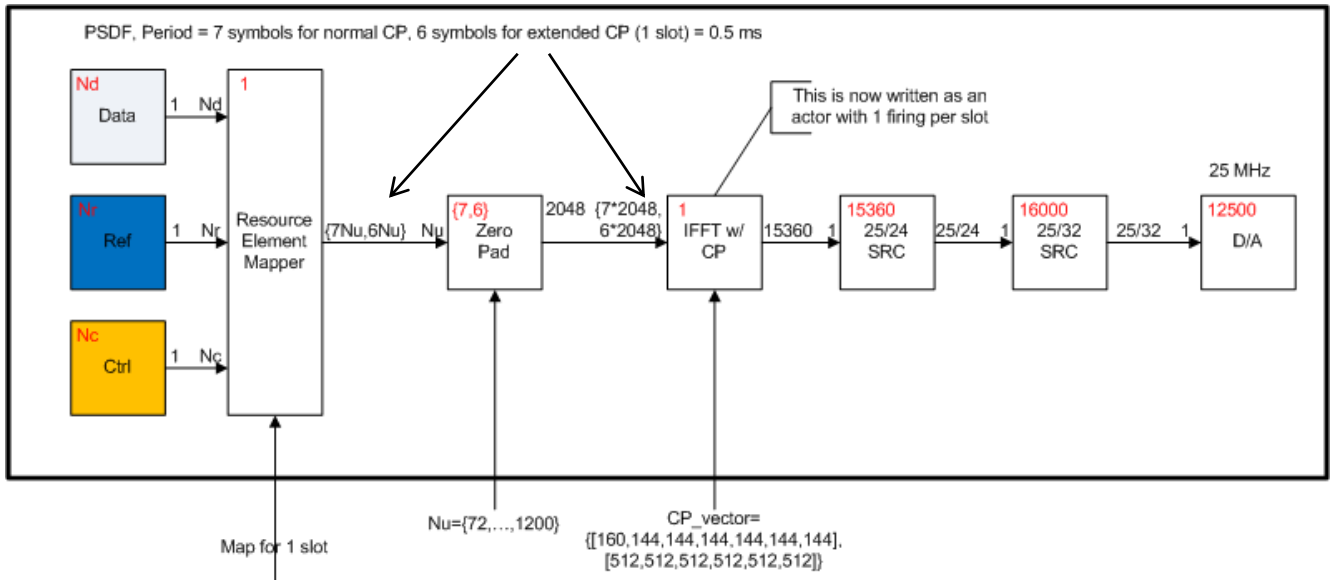
Platform trends

- **100's of processing elements**
- Heterogeneous processors and memories
- Distributed I/O
- E.g. Heterogeneous FPGA targets

Design Trends

- Abstraction based design
  - Manage size and complexity of design
  - Reason about key properties early in design
  - Hide lower level details
- Component based design
  - Support a modular design process
  - Enable reuse across designs
  - Allow easy integration of IP

Streaming Model of the OFDM Transmitter



- Nt = {1,2,4}
  - Compile time - # transmitters
- Nu = {72, 180, 300, 600, 900, 1200}
  - Initialization time - Bandwidth
- CP mode = {'Normal', 'Extended'}
  - Run time, To overcome Inter-symbol-interference, Can be applied at symbol boundary
- CP Vector
  - Selection based on CP mode, Elements must be applied at symbol boundary

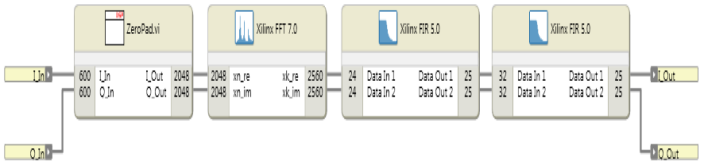
Challenge: How to express a domain expert's algorithm specification in a model that is viable for analysis and implementation?

DSP Designer

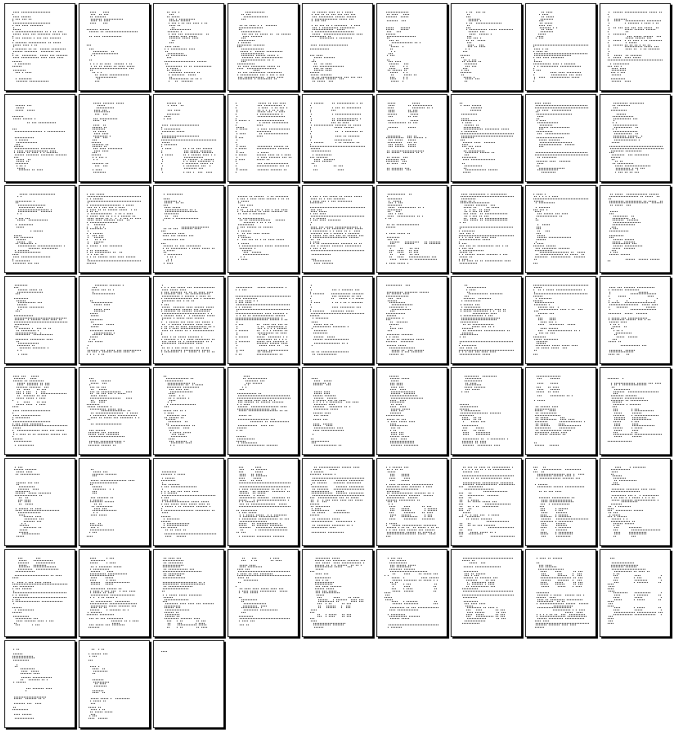
- Development environment for creating streaming high-performance RF and DSP applications for FPGA targets
- Driving applications
  - Spectral analysis, signal intelligence, software radios
- Platforms
  - Heterogeneous FPGA platforms (R-series, FlexRIO)
- Target users
  - RF and DSP domain experts

Model Based Design

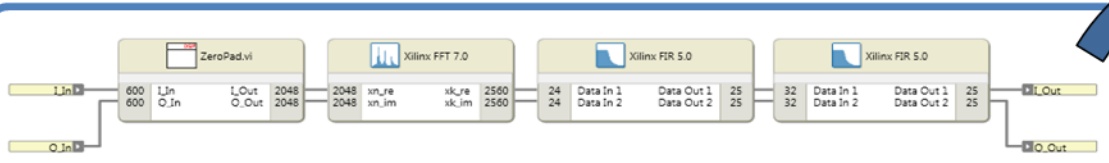
Abstract Model



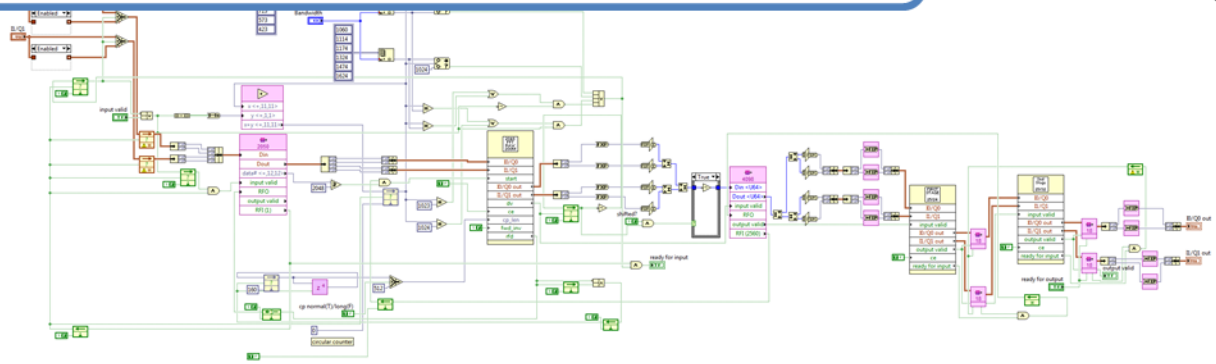
HDL Implementation



Value of DSP Designer



LabVIEW DSP Designer

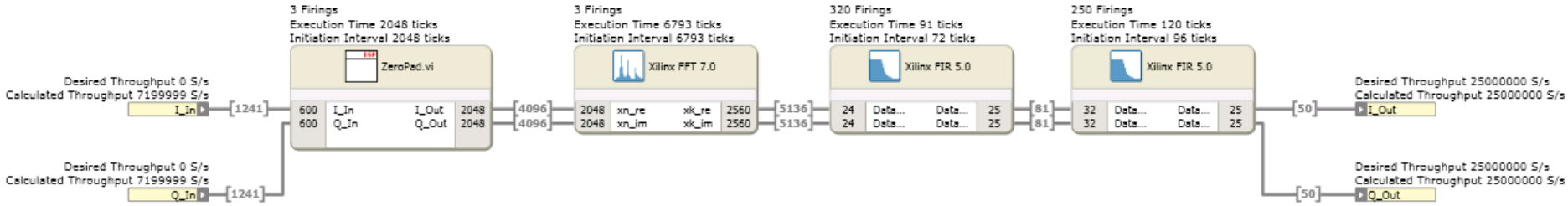


LabVIEW FPGA

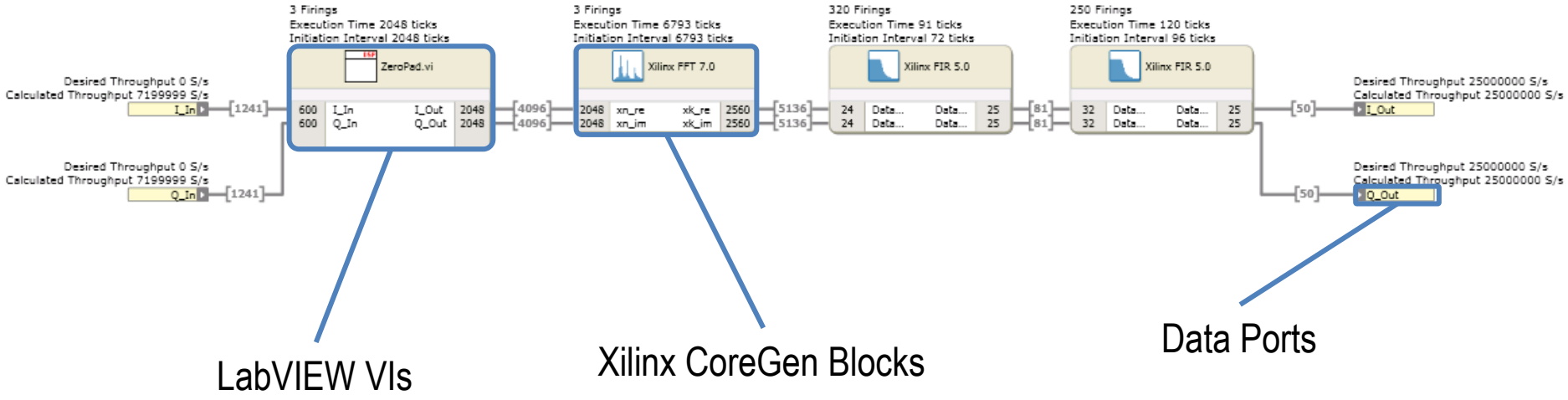
```
-- synthesis translate_off
component wrapped_secondStage25o32_32MSPS100MHz
      port (
      clk: IN std_logic;
      ce: IN std_logic;
      nd: IN std_logic;
      rfd: OUT std_logic;
      rdy: OUT std_logic;
      din_1: IN std_logic_VECTOR(31 downto 0);
      din_2: IN std_logic_VECTOR(31 downto 0);
      din_3: IN std_logic_VECTOR(31 downto 0);
      din_4: IN std_logic_VECTOR(31 downto 0);
      dout_1: OUT std_logic_VECTOR(33 downto 0);
      dout_2: OUT std_logic_VECTOR(33 downto 0);
      dout_3: OUT std_logic_VECTOR(33 downto 0);
      dout_4: OUT std_logic_VECTOR(33 downto 0));
end component;
```

VHDL

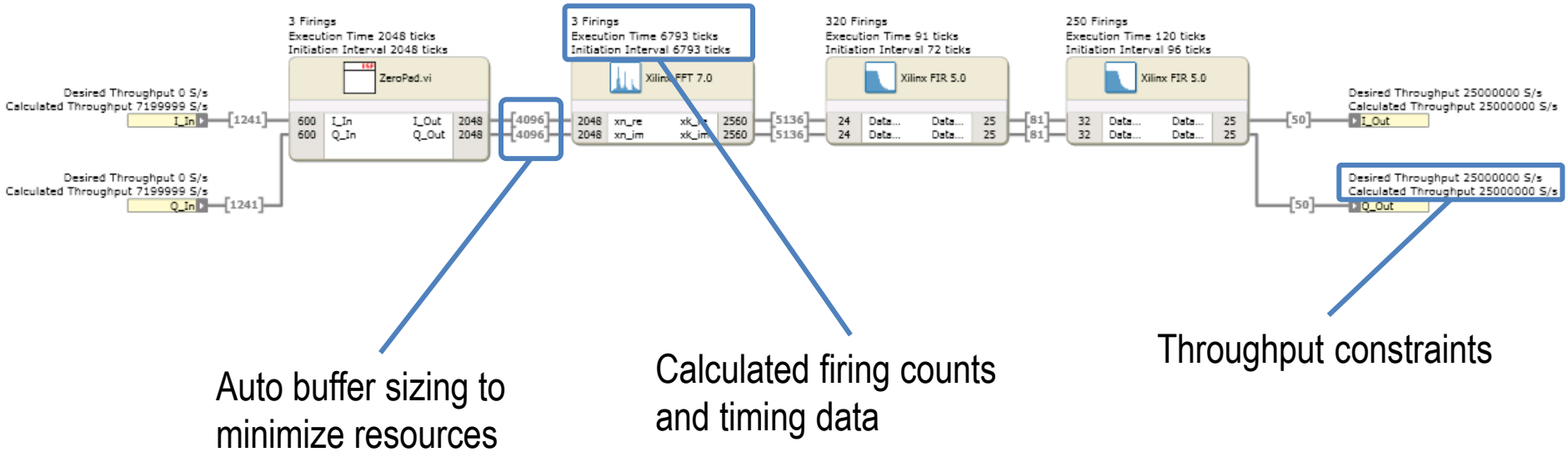
LabVIEW DSP Designer



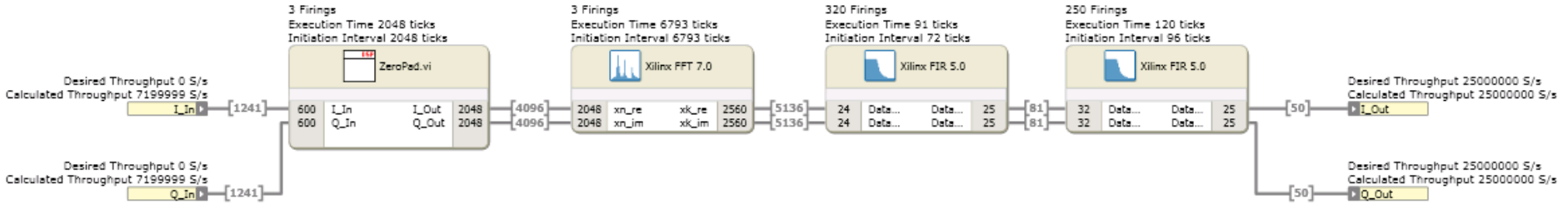
LabVIEW DSP Designer

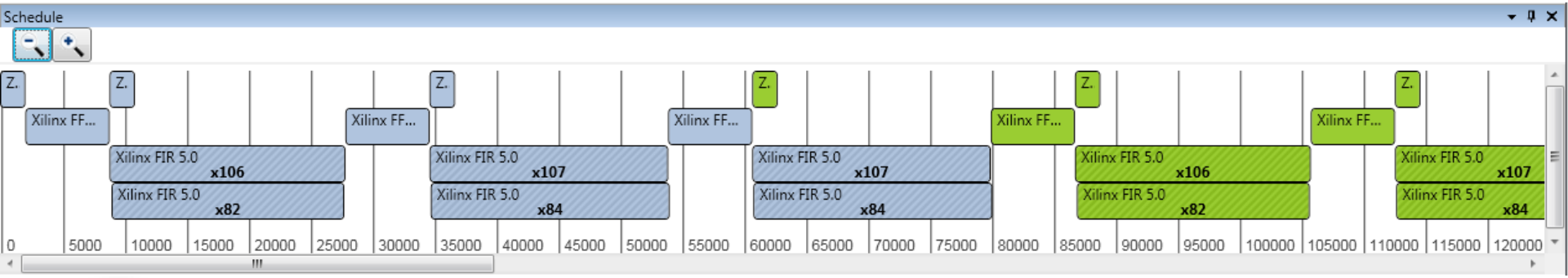


LabVIEW DSP Designer



LabVIEW DSP Designer

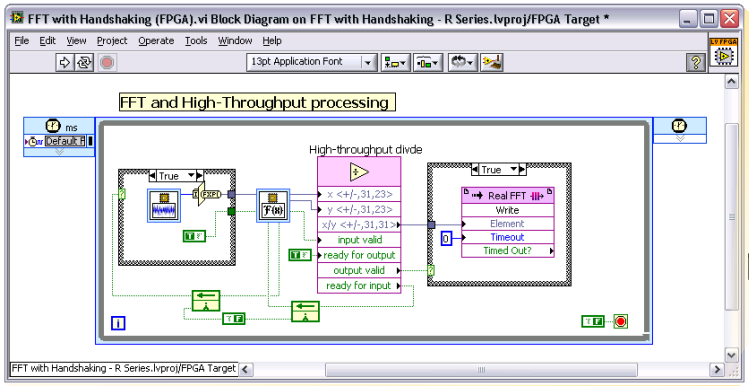




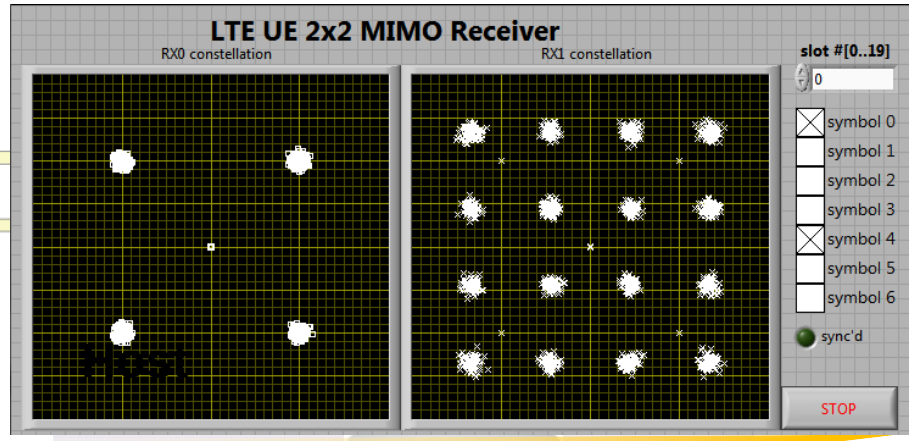
Calculated Schedule View

RF Design Flow

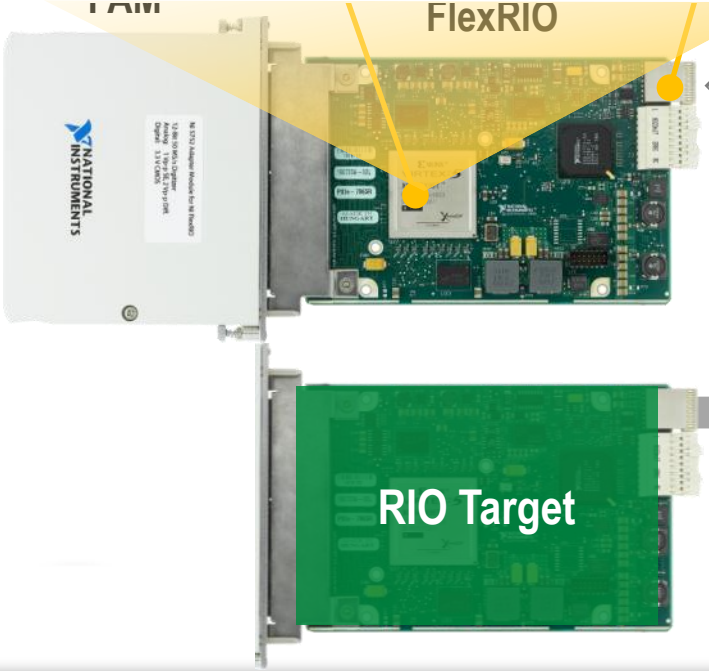
LabVIEW FPGA



Transfer

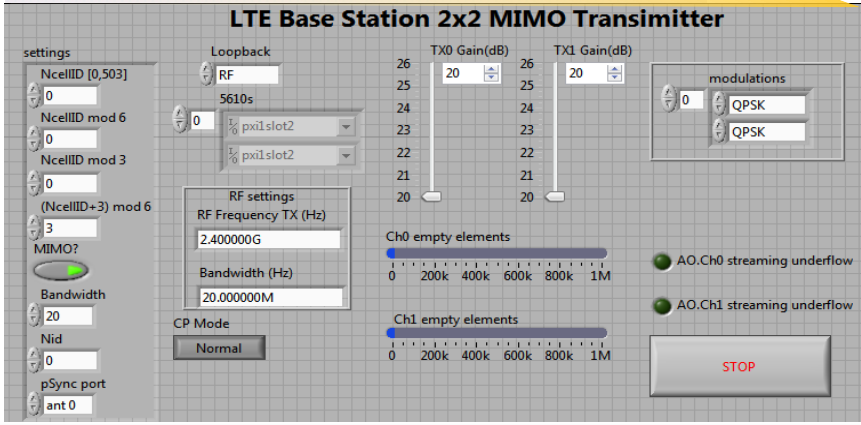


FlexRIO

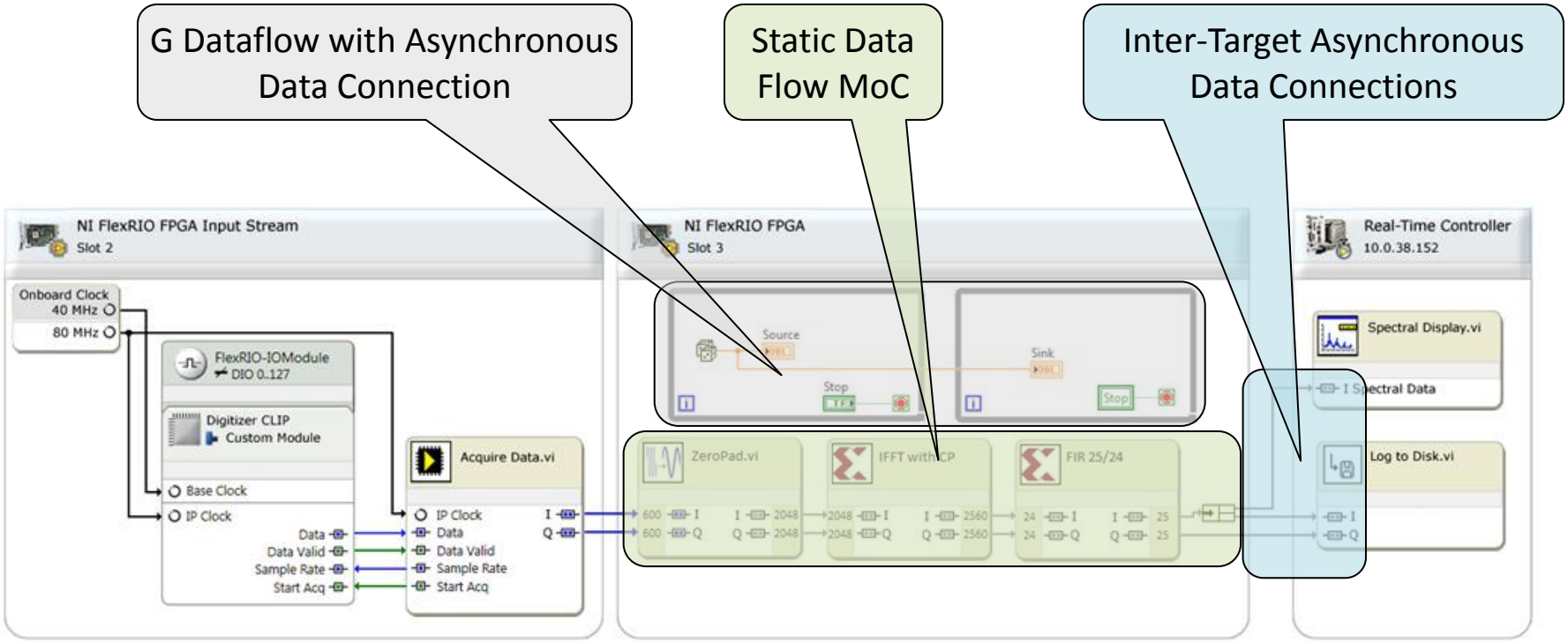


RIO Target

Peer-to-Peer Streaming



Modeling System-Level Designs



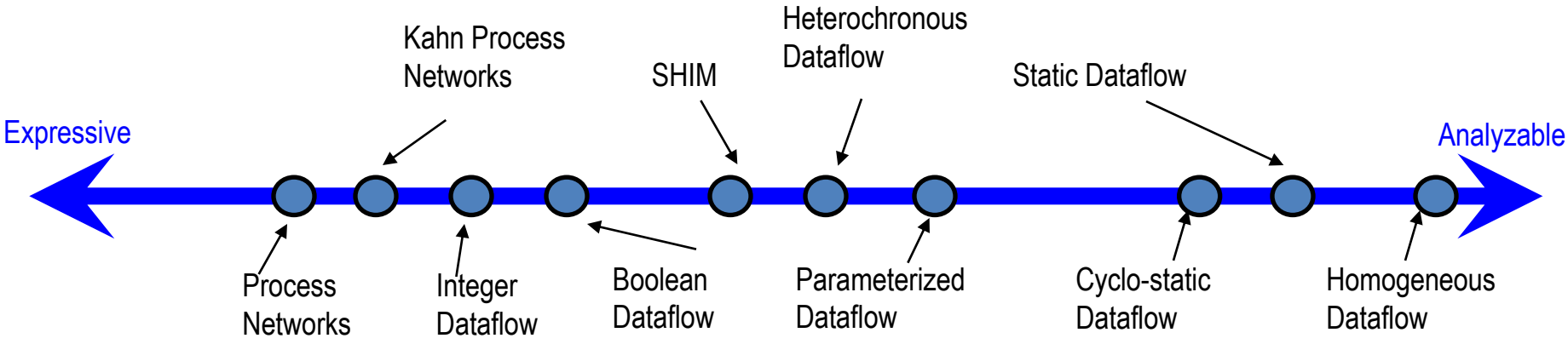
Agenda

- Background – NI
- DSP Designer
- **Models, Analysis and Exploration**
- Model Extensions
- Looking Forward

DSP Designer Focus Areas

- Models of Computation
- Analysis and Optimization Back End
- Actor Definition
- Performance Models and Timing Library
- IP Modeling and Integration
- Simulation and Verification
- Code Generation and Implementation

MoCs for Streaming Applications



Deter-ministic?	No		Yes					
Bounded data rates?	No			Yes				
Deadlock and boundedness decidable?			No			Yes		
Static scheduling?						No		Yes

Area of focus for DSP Designer

Key trade-off: Analyzability vs. Expressibility

[1] Edward A. Lee, "Concurrent Models of Computation for Heterogeneous Software", EECS 290, 2004.
[2] Stephen Edwards, "SHIM: A Deterministic Model for Heterogeneous Embedded Systems", UCB EECS Seminar, 2006.

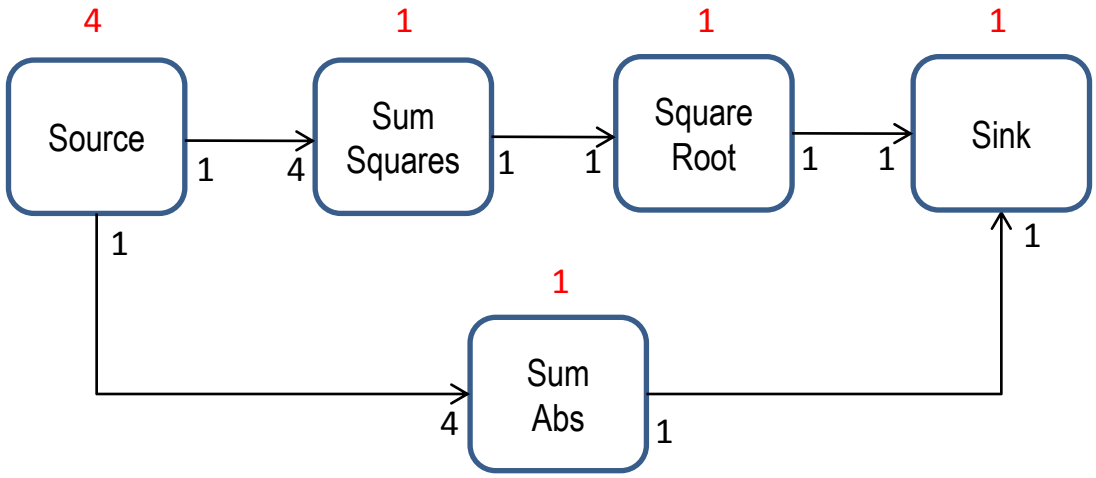
Analysis and Optimization Features

- Core dataflow optimizations
  - Model validation (deadlock and unboundedness detection)
  - Throughput and latency computation
  - Buffer size optimization (under throughput constraints)
  - Schedule computation

- Hardware specific optimizations
  - Resource constrained schedule computation
  - Retiming and fusion
  - Rate matching
  - IP interface synthesis

[1] S. S. Bhattacharyya, P. K. Murthy and E. A. Lee, "Software Synthesis from Dataflow Graphs," Kluwer Academic Publishers, Norwell, Mass, 1996.

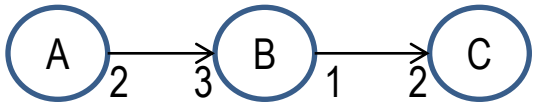
Model Validation



Analysis determines at compile time that this application executes in bounded memory and is deadlock free

Buffer Size Optimization

Example SDF Graph

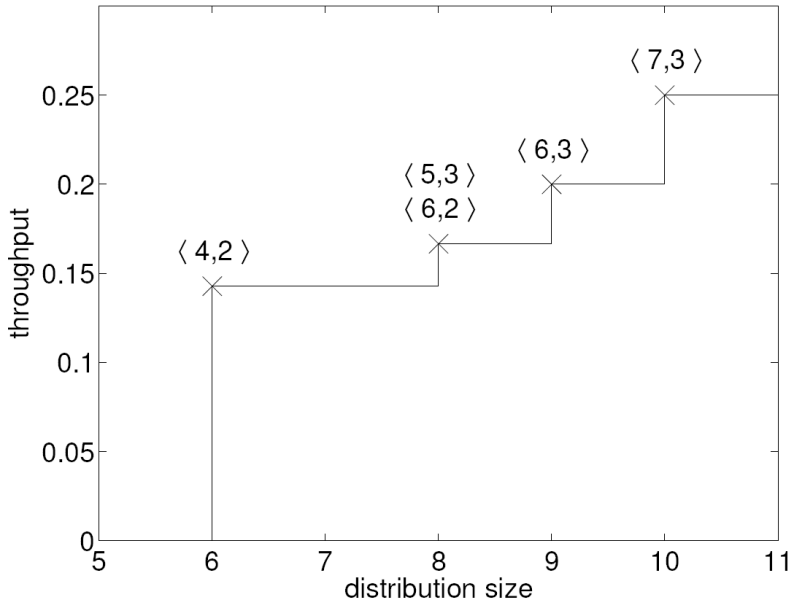


Repetitions vector: (3, 2, 1)

Problem Assumptions

- Execution times: (A,1), (B,2), (C,2)
- No resource constraints

Pareto Space of Throughput and Buffer Sizes

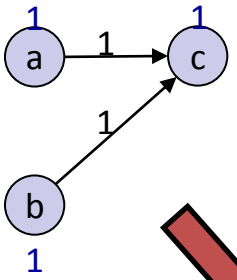


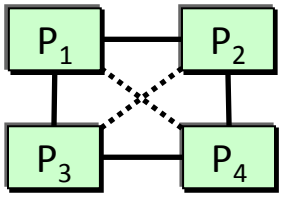
Exploration results in a Pareto space of throughput and buffer sizes

[1] S. Stuijk, M.C.W. Geilen and T. Basten, “Exploring Trade-Offs in Buffer Requirements and Throughput Constraints for Synchronous Dataflow Graphs”, DAC 2006

Optimization Toolbox

Application model
(with cost annotations)





Platform model

Application and platform constraints

Toolbox of optimization methods

- Heuristic list scheduling
- Graph partitioning
- Integer linear programming
- Constraint programming
- Simulated annealing

Implementation and Performance Analysis

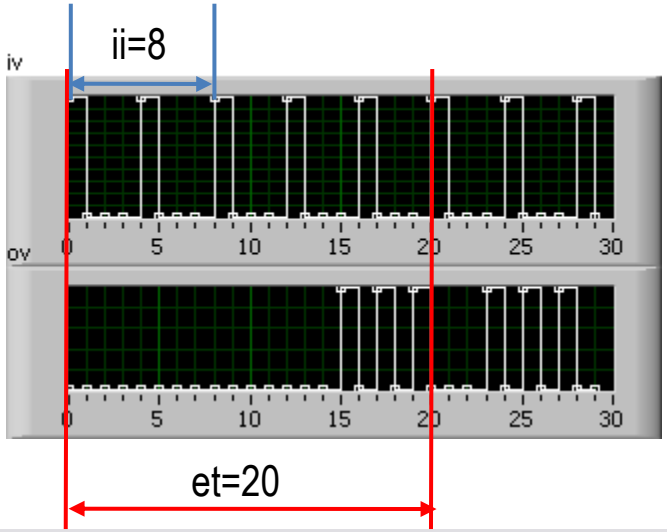
Agenda

- Background – NI
- DSP Designer
- Models, Analysis and Exploration
- **Model Extensions**
- Looking Forward

Timing Models for IP Blocks

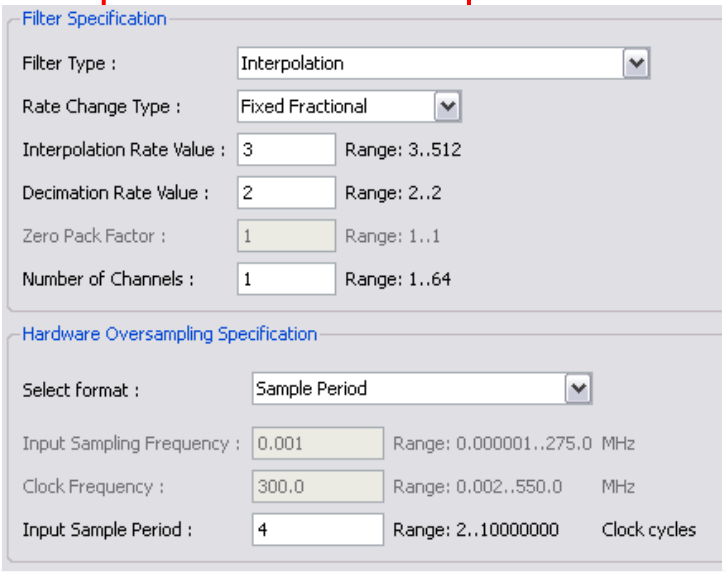
Rational Resampler

Execution time: 20
Initiation interval: 8



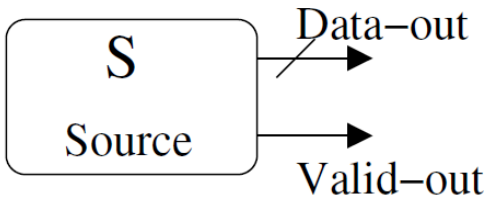
- Execution time: time to complete one firing, i.e. read 2 samples and produce 3 samples

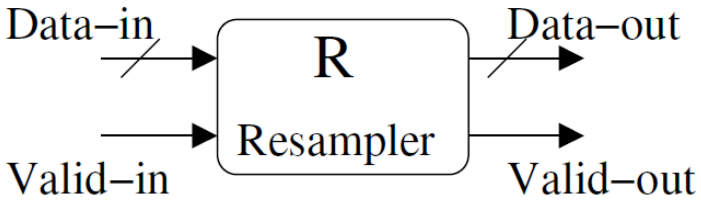
- Initiation interval: minimum time between start of successive firings



[1] Xilinx Inc., Datasheet for CoreGen Resampler block.

Example: Source-Resampler System

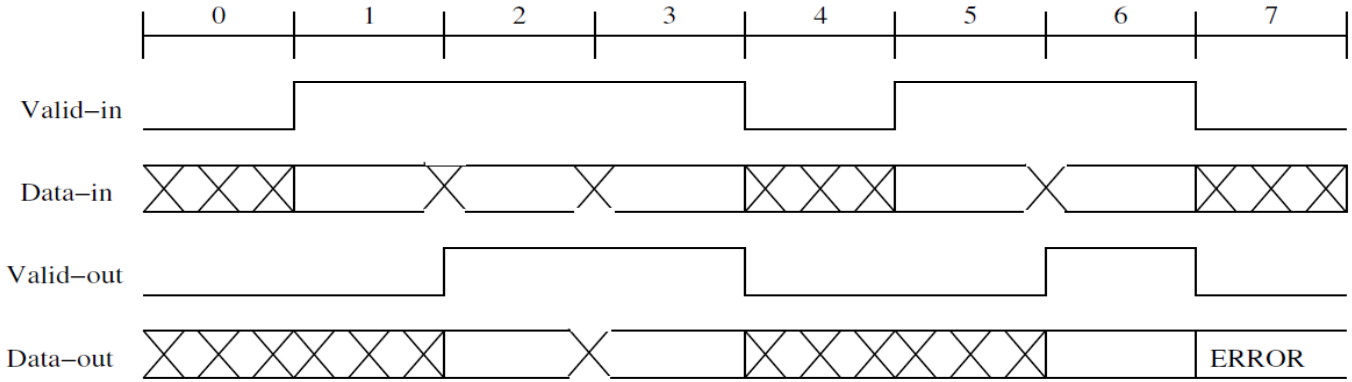




S outputs 1 sample every 2 cycles

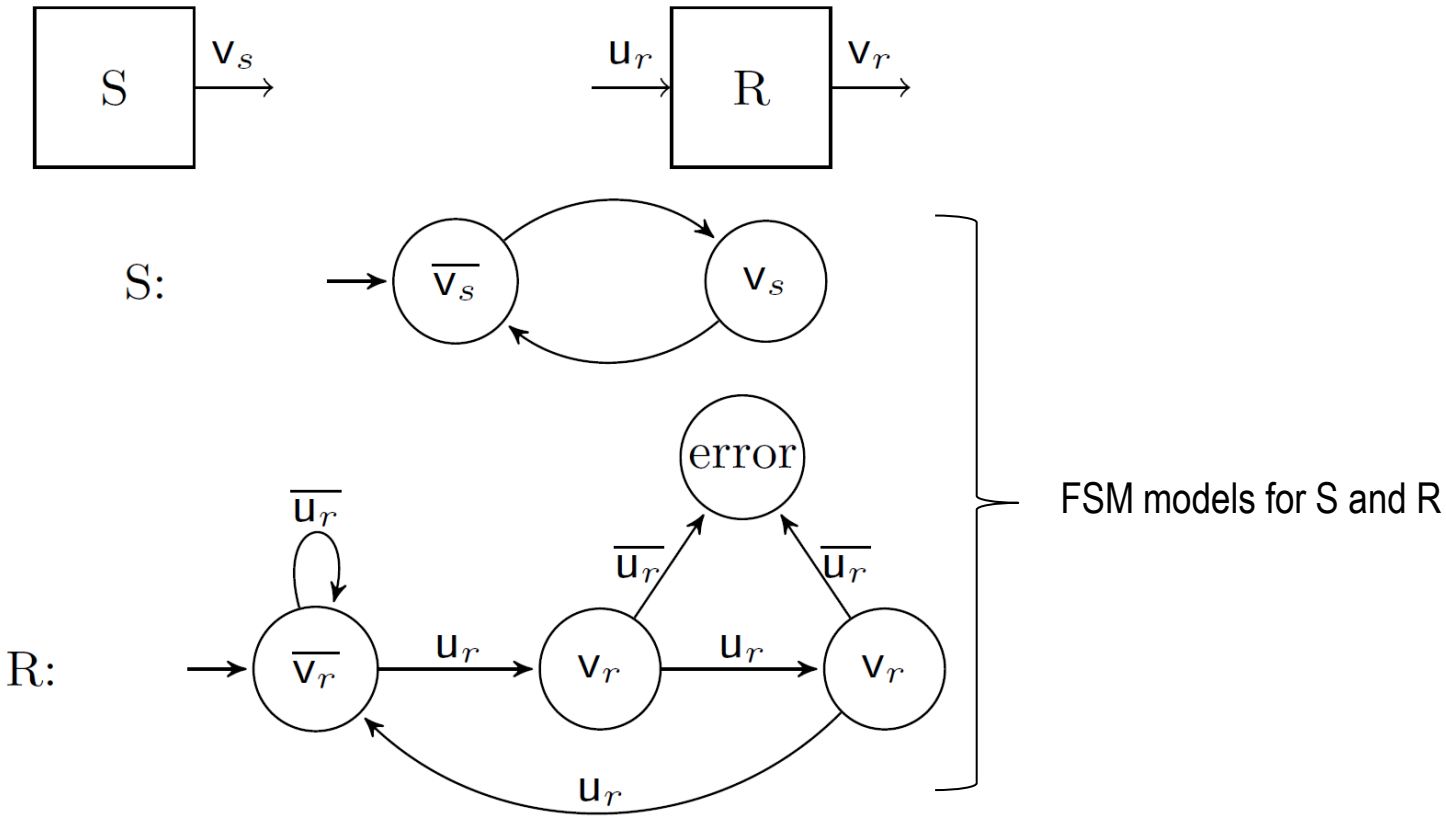
R reads 3 inputs samples in consecutive cycles and outputs 2 samples in the 2nd and 3rd cycles

Timing Diagram of Resampler



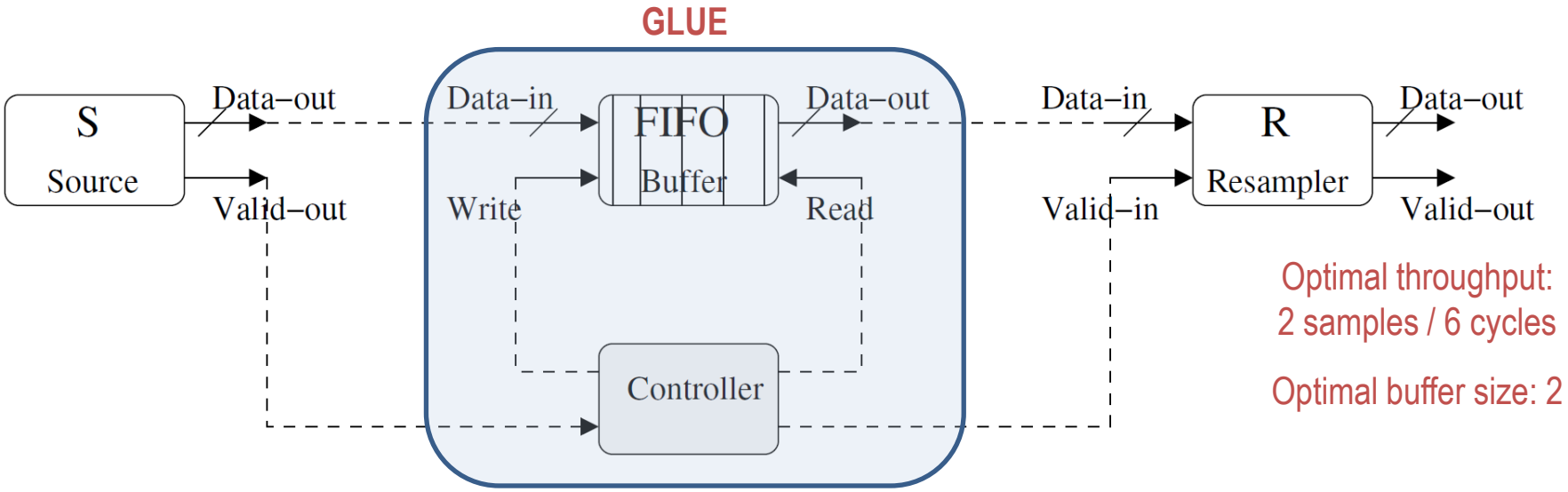
Objective: Create a valid composition of these blocks so that behavior and timing constraints are respected

Source-Resampler: Interface Signals



Connecting v_s output of Source to u_r input of Resampler results in an invalid composition

Source-Resampler: System Implementation

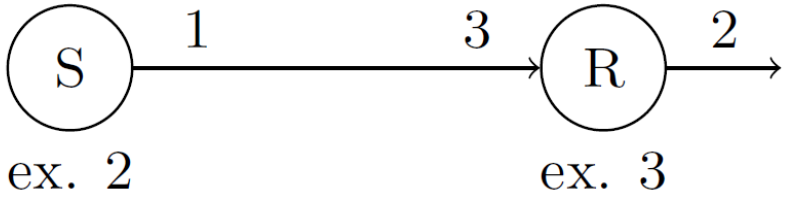


Glue design problem: Given a set of actors, synthesize a glue (buffers and controllers), such that the resulting system satisfies correctness and performance properties

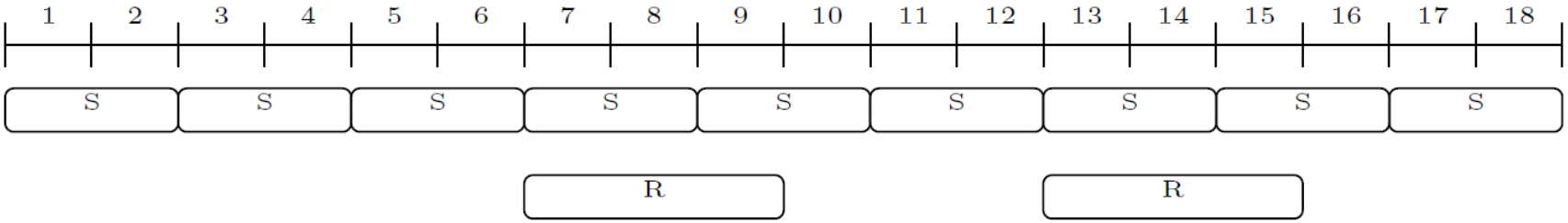
Solutions to the Glue Design Problem

- Solutions at the HDL and FSM level
  - Suffer state space explosion problem
  - Infeasible for most practical systems
- Solutions based on abstract models
  - Enable efficient static analysis
  - Support automated synthesis
- But important to choose the right abstractions that yield correct and non-defensive implementations

Source-Resampler: Static Dataflow Model



Self-timed schedule to achieve optimal throughput
(requires buffer of size 5)



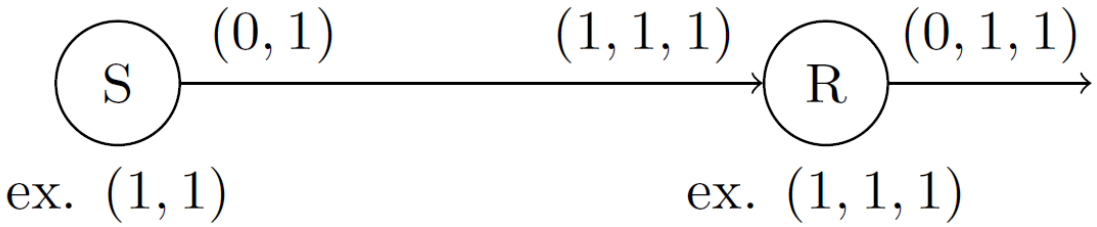
Limitations of the SDF model:

- Does not capture how tokens are accessed
- Analysis conservatively allocates space for tokens from firings of S that occur while R executes
- Resulting implementation is defensive

[1] E. A. Lee and D. Messerschmitt, “Synchronous Dataflow”, in Proc. of the IEEE, 75(9), 1987.

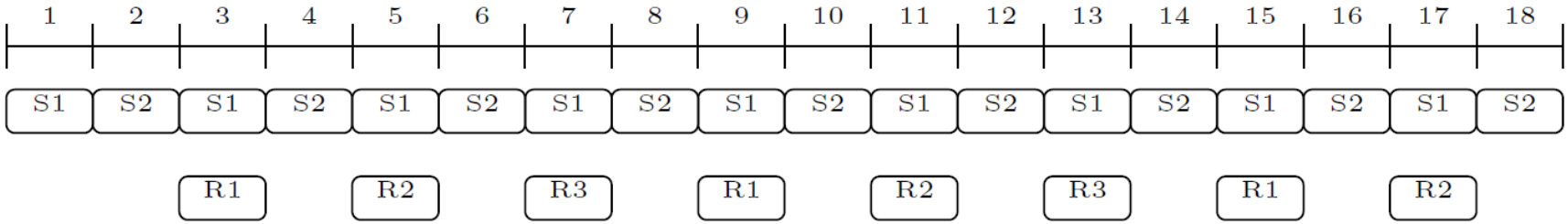
[2] S. Stuijk, M.C.W. Geilen and T. Basten, “Exploring Trade-Offs in Buffer Requirements and Throughput Constraints for Synchronous Dataflow Graphs”, DAC 2006

Cyclo-Static Dataflow Model



Self-timed schedule to achieve optimal throughput

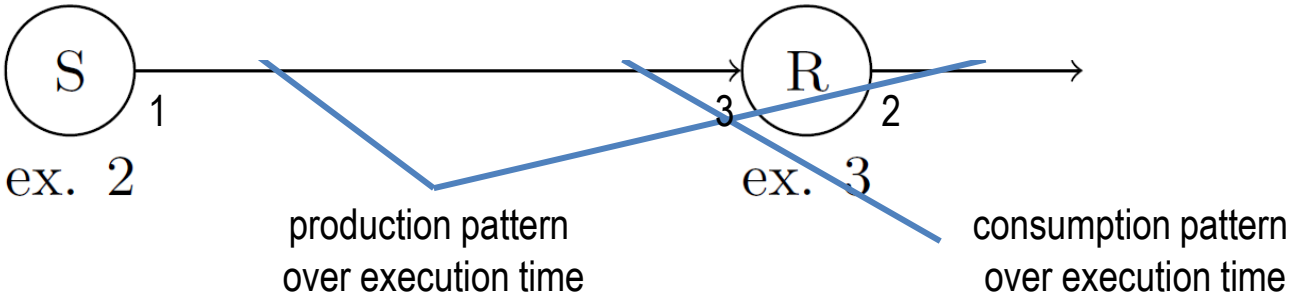
(requires buffer of size 1)



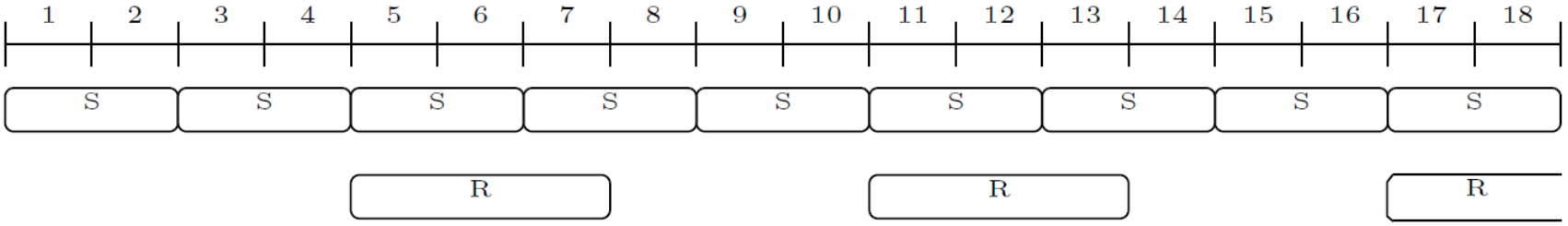
Limitations of the CSDF model:

- Does not capture requirement that Resampler IP must receive 3 tokens in consecutive cycles
- Analysis underestimates space needed for the buffer
- Resulting implementation is incorrect

Static Dataflow Model with Access Patterns



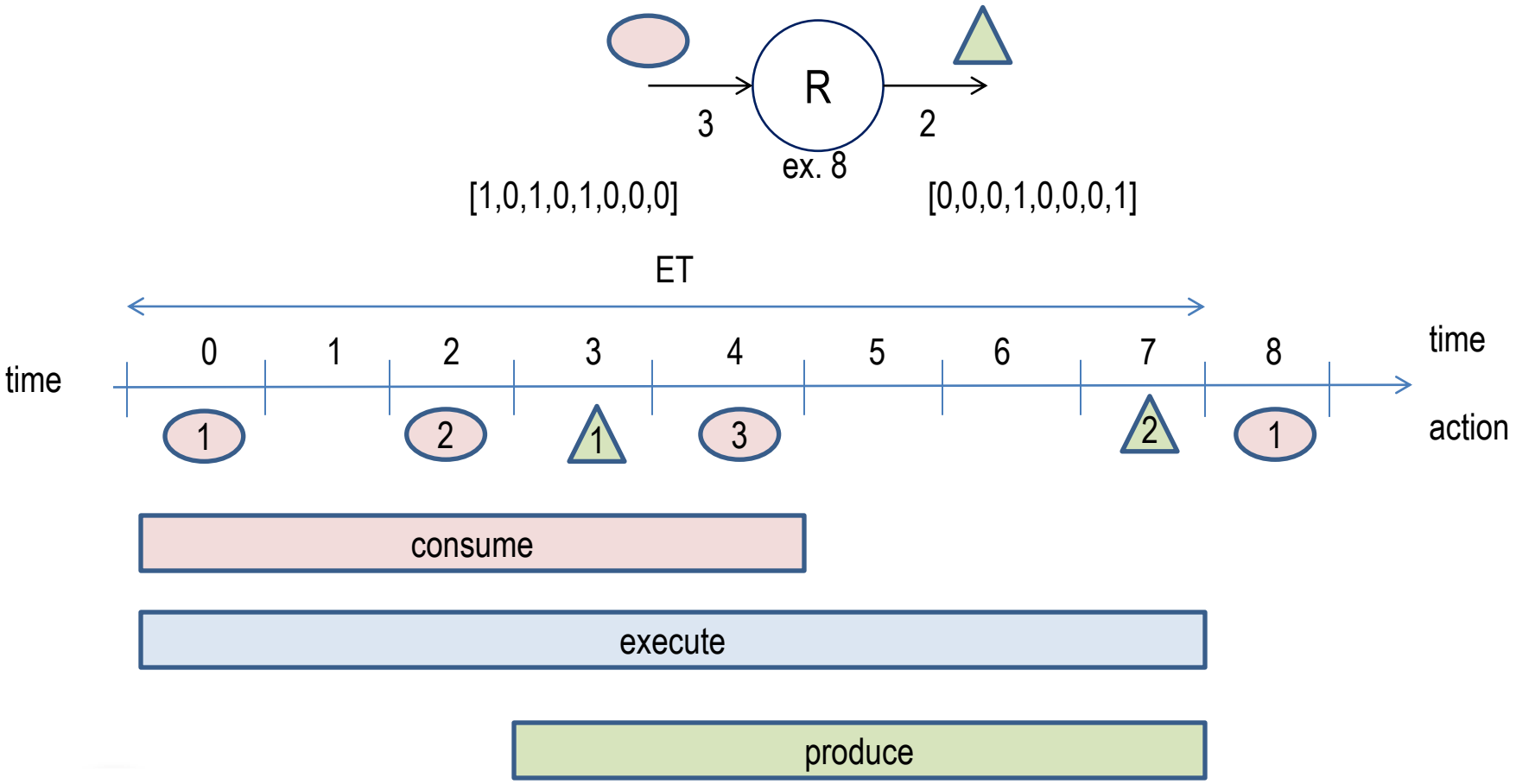
Self-timed schedule to achieve optimal throughput
(requires buffer of size 2)



Strengths of the SDF-AP model:

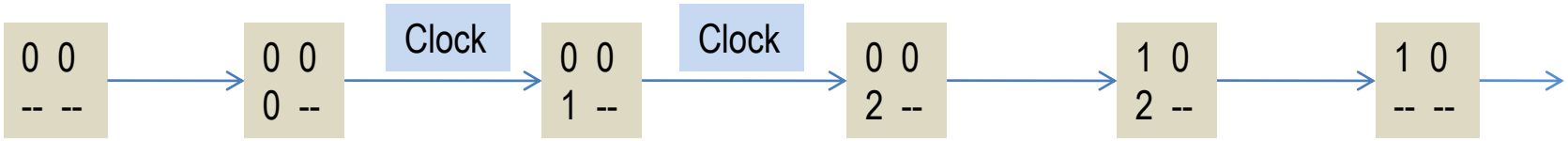
- Explicitly specifies how tokens are consumed and produced in time
- Analysis yields a buffer of size 2
- Resulting implementation is correct and non-defensive

Access Pattern Example



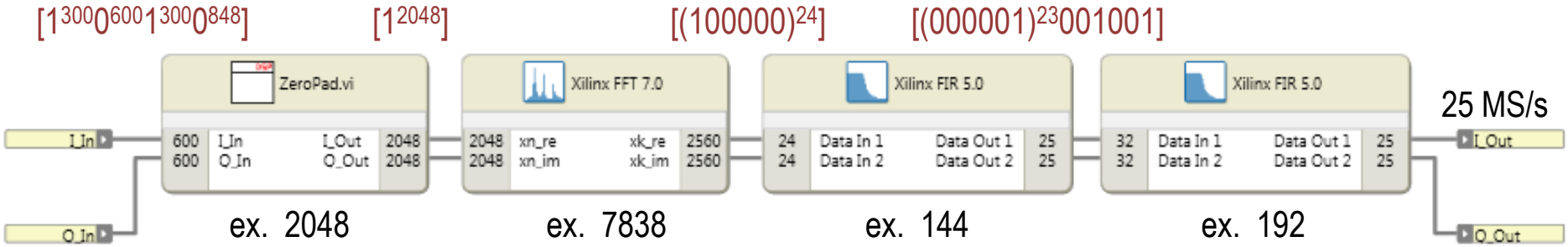
Semantics of SDF-AP

- SDF-AP is formalized as a state transition system where each state tracks the number of tokens for channels and the number of clock cycles the actors have been executing



- Trace is *AP-compatible* if for all states all actors read/write tokens as per access patterns
- An AP-compatible trace is *deadlock-free* if an actor cannot fire after a certain state on the trace
- A deadlock free trace is *bounded* if for all states on the trace channel sizes are finitely bounded

Case Study: OFDMA Transmitter



Sizes

SDF	600	2048	2133	56	25
SDF-AP	600	1	2133	2	25

- Glue design using SDF-AP models
  - Improves buffer sizes compared to SDF
  - Results in a non-defensive controller implementation

Summary

- Glue design problem
  - Synthesize glue (buffers, controllers) between actors so that system meets correctness and performance properties
- Abstract models for glue design
  - Choose the right abstractions that yield correct and non-defensive implementations
- Static dataflow model with access patterns
  - Suitable model for glue design for hardware IP

Directions Ahead

- Analysis extensions
  - Efficient methods to check correctness and non-defensiveness
  - Automatic characterization of access patterns
  - Formalize relation between abstract and concrete models
- Specification for control and timing with dataflow
  - Heterochronous dataflow
  - Scenario Aware dataflow

Agenda

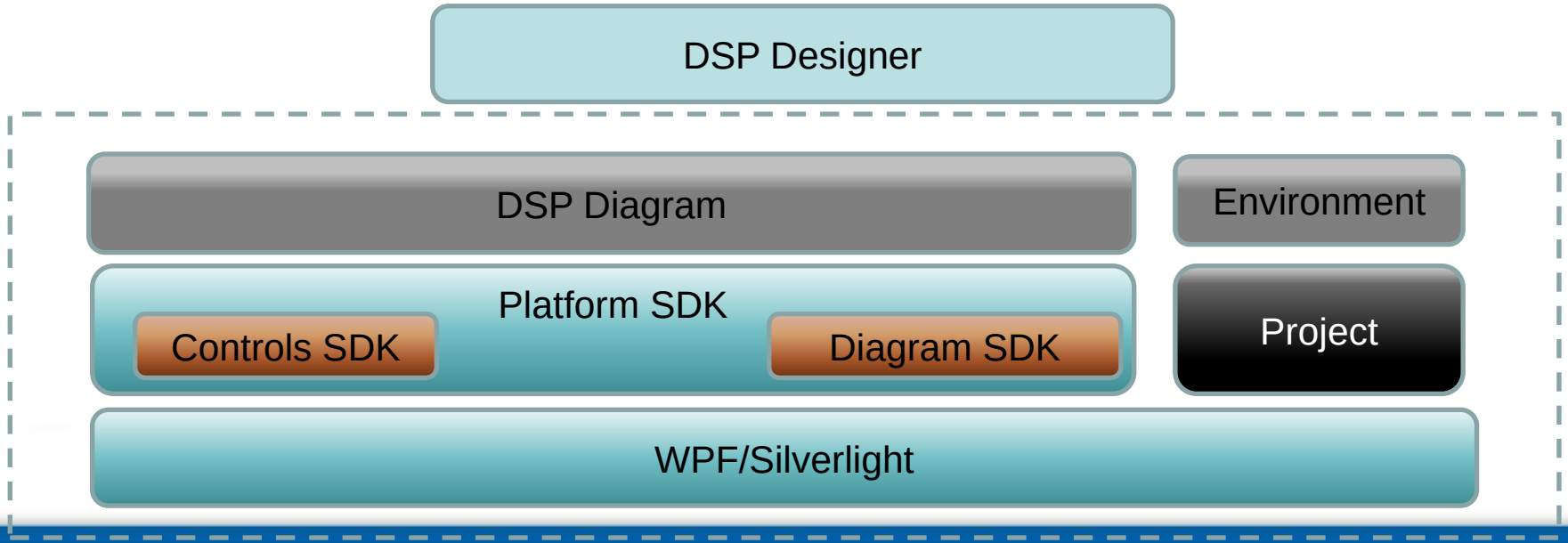
- Background – NI
- DSP Designer
- Models, Analysis and Exploration
- Model Extensions
- **Looking Forward**

Future Opportunities for Collaboration

- Targeting heterogeneous architectures
  - Modeling Correct abstractions
  - Providing analysis, mapping and implementation capabilities
- Specification for real-time systems
  - Accounting for timing and reliability metrics
- Other hardware specific problems
  - Behavioral interface formalisms
  - IP interface standardization

An Open API and Software Stack

- Create products for new problem spaces
- Leverage common infrastructure across products
- Scale implementations across platforms – desktop OSs to web
- Improve developer efficiency in creating products
- Expect infrastructure to external partners to create products



Progression of Our Vision

1980	1990	2000	2010	2020
Traditional Instrument Control	PC-Based Data Acquisition	PXI Systems & Real-Time Technology	RIO Architecture RF Systems	Graphical System Design
				 
Laboratories & Research	Measurement Automation	Automated Test	RF , MEMS, Medical, Audio and Video Test Industrial Measurement and Control	RF Design Medical Industrial Big Physics Embedded DSP Design Robotics

Thanks to our UC Berkeley Collaborators - CHESS, BWRC, ParLab -

- Prof. Edward A. Lee
- Prof. Alberto Sangiovanni-Vincentelli
- Prof. Kurt Keutzer
- Prof. Jan Rabaey
- Prof. Bora Nicolic
- Prof. Ras Bodik
- Prof. Sanjit Seshia
- Dr. John Eidson
- Dr. Stavros Tripakis
- Dr. Slobodan Matic
- Milos Jorgovanovic
- Vinayak Nagpal
- Isaac Liu

National Instruments Information Session

Wednesday, November 9th
Soda Hall, Wozniak Lounge
5:30pm – 6:30pm

Please join us for pizza and sodas, and learn more about the exciting career opportunities at National Instruments!

Employ Your Imagination

FORTUNE 100 BEST COMPANIES TO WORK FOR

ni.com/careers