

System-level Synthesis of Dataflow Applications for FPGA- based Distributed Platforms

Hugo A. Andrade, Kaushik Ravindran, Alejandro Asenjo, Casey Weltzin

NI Berkeley, NI Austin

National Instruments Corporation

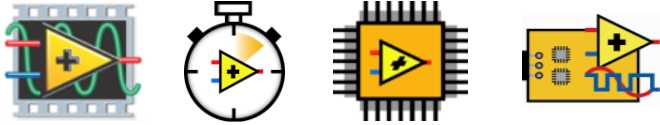
Tenth Biennial Ptolemy Miniconference

Berkeley, California, November 7th, 2013


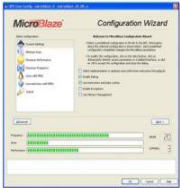
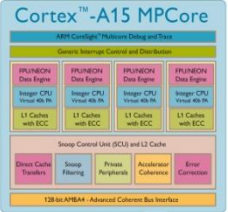




Modern FPGA-based Distributed Heterogeneous Platforms

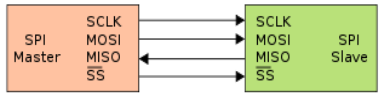
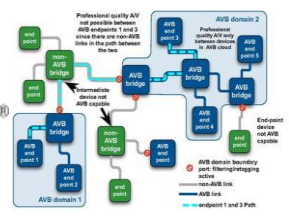
LabVIEW



Desktop, RT, FPGA, uP, DSP Design, IP Builder

PCI EXPRESS



High Level Synthesis

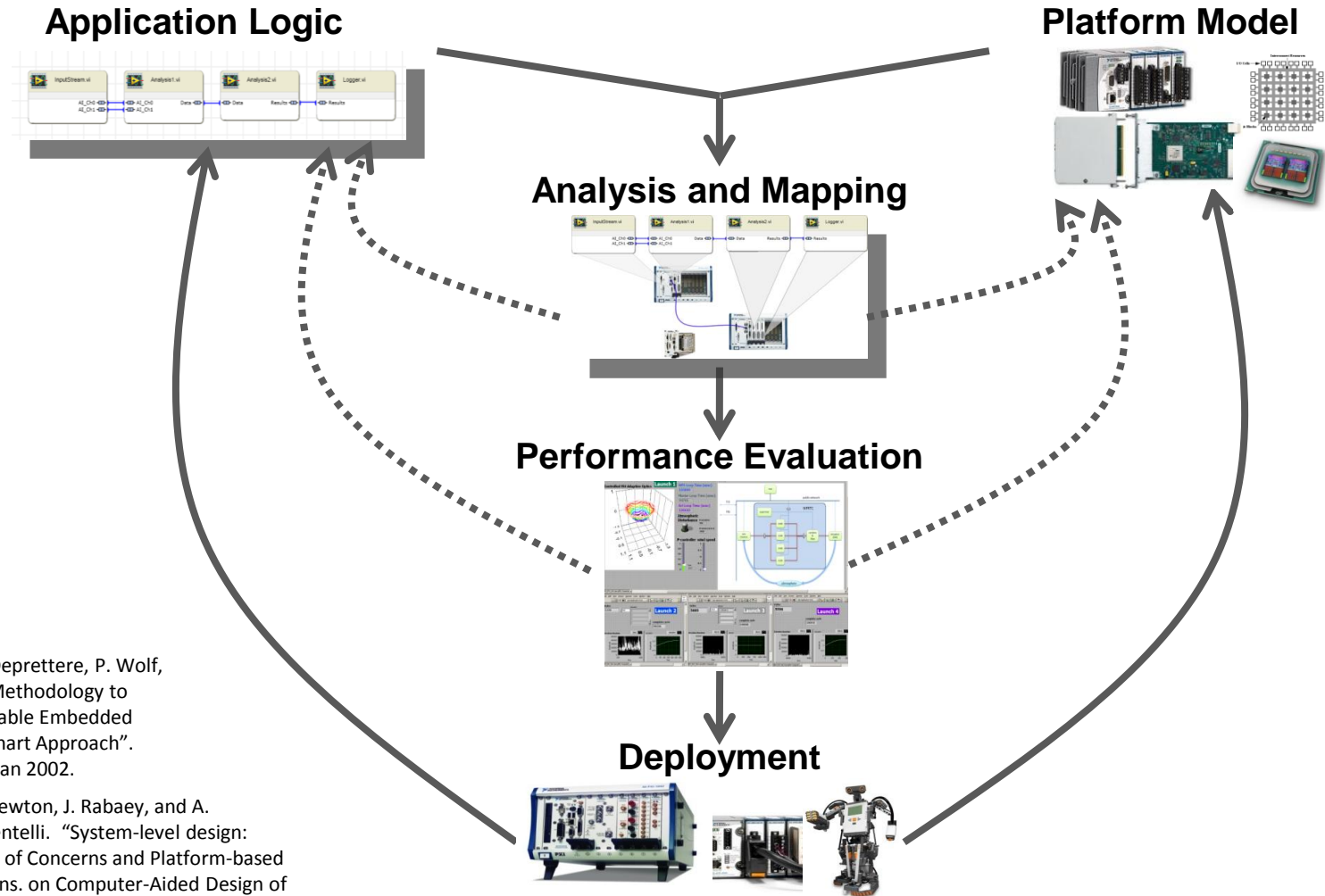
- A method for *increasing* (FPGA or system) *design productivity* while still providing an *efficient implementation*
 - Broadens design productivity to domain experts
 - They can use a high-level language such as ANSI C/C++, Python, MATLAB, LabVIEW
 - Design space exploration: optimizing among performance, footprint, or other constraints
 - Enables targeting more complex platforms from higher level program

HLS Techniques Covered

- Directive-based high-level synthesis
- Synthesis from analyzable domain specific languages
- System-level synthesis to heterogeneous targets

Applicable to textual and graphical hardware programming environments, e.g. VHDL/Verilog or LabVIEW FPGA

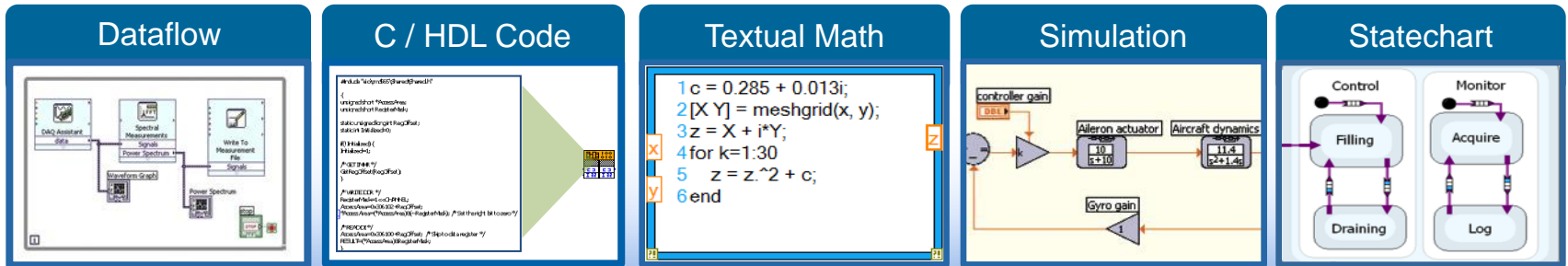
Y-Chart Disciplined Design Methodology



B. Kienhuis, E. F. Deprettere, P. Wolf, K. A. Vissers. "A Methodology to Design Programmable Embedded Systems - The Y-Chart Approach". SAMOS, p.18-37, Jan 2002.

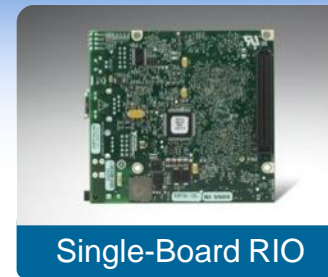
K. Keutzer, A. R. Newton, J. Rabaey, and A. Sangiovanni-Vincentelli. "System-level design: Orthogonalization of Concerns and Platform-based Design". IEEE Trans. on Computer-Aided Design of Integrated Circuits and System, 19(12): p.1523-1543, Dec 2000.

Models of Computation and Platform-Based Design

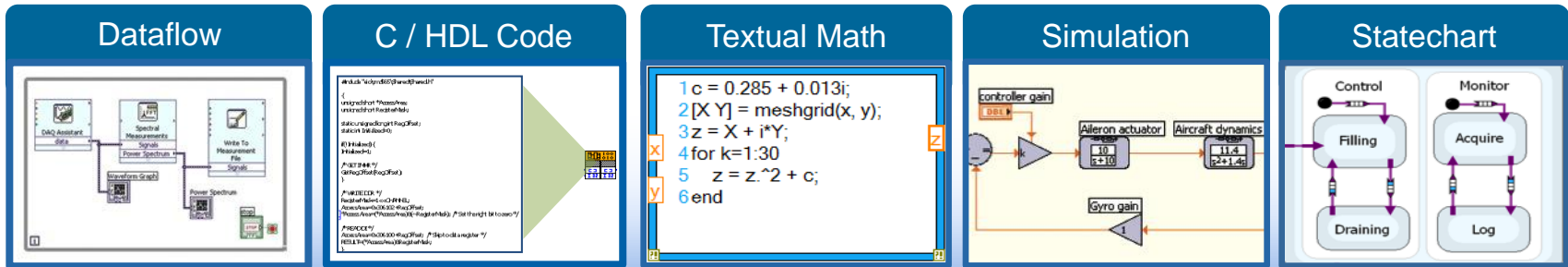


LabVIEW

Desktop, RT, FPGA, uP

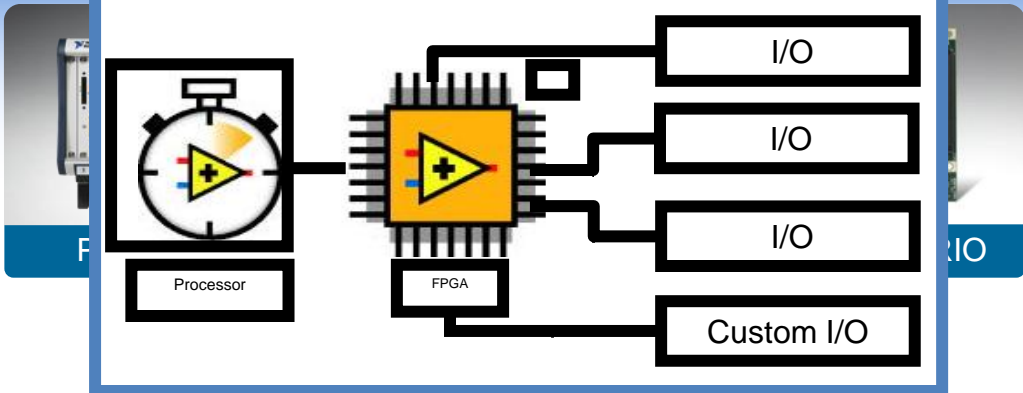


Models of Computation and Platform-Based Design

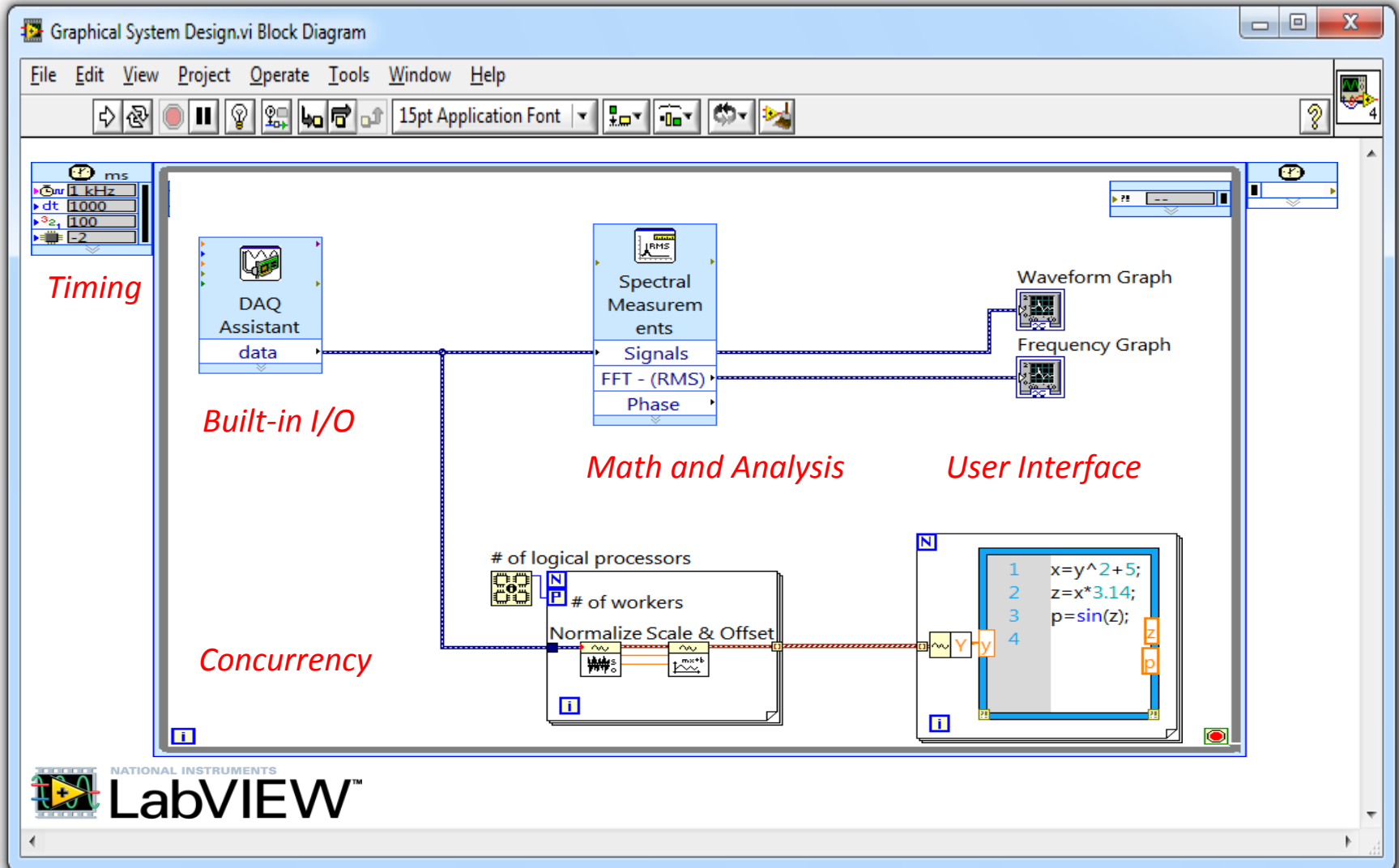


LabVIEW

Desktop, RT, FPGA, uP



Base Programming Language: G



Base Programming Language: G

The image displays the LabVIEW graphical programming environment. The main window is titled "Graphical System Design.vi Block Diagram" and features a menu bar (File, Edit, View, Project, Operate, Tools, Window, Help) and a toolbar. The block diagram includes several key components:

- Timing:** A control panel on the left with sliders for "ms", "kHz", "1000", "100", and "2".
- Built-in I/O:** A "DAQ Assistant data" block.
- Math and Analysis:** A "Spectral Measurements" block with sub-blocks for "Signals", "FFT - (RMS)", and "Phase".
- Concurrency:** A "Parallel" block with "# of logical processors" and "# of workers" controls, containing a "Normalize Scale & Offset" block.
- User Interface:** A "Frequency Graph" block and a "Code" block containing the following text:

```
1 x=y^2+5;  
2 z=x*3.14;  
3 p=sin(z);  
4
```

An inset window titled "Signal Generation and Processing.vi" provides a detailed view of signal processing parameters and results:

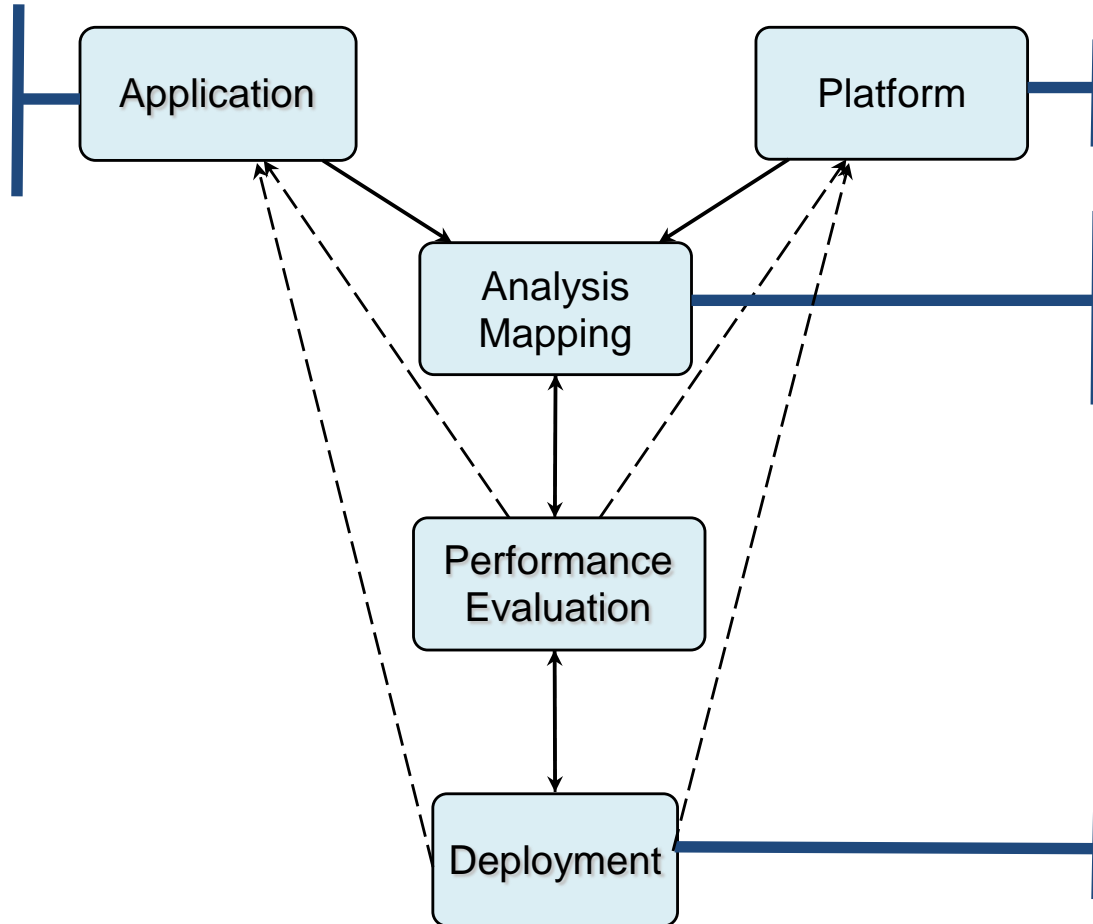
- Signal Generation:** Controls for "Input Signal 1" (square) and "Input Signal 2" (sine), both with "Frequency (Hz)" sliders.
- Signal Processing:** Controls for "Select Window" (Hanning) and "Select Filter" (Butterworth).
- Acquired Waveform:** A plot showing the input signals.
- Processed Waveform:** A plot showing the signals after Hanning windowing and Butterworth filtering.
- Power Spectrum:** A plot showing the magnitude spectrum of the processed signals, with a "filter cutoff" indicator.

The LabVIEW logo and "NATIONAL INSTRUMENTS" text are visible at the bottom left of the main window.

DIRECTIVE-BASED HIGH-LEVEL SYNTHESIS

Approach

- Focus on describing *algorithm* in high-level programming language
- Add separate directives



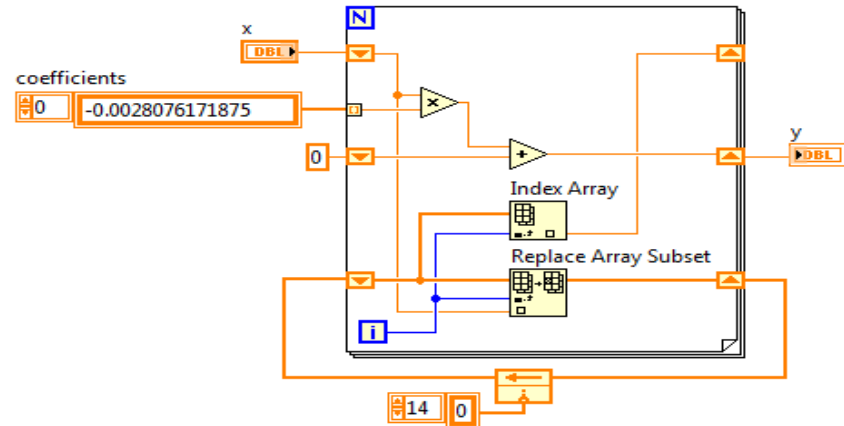
- Platform model
- Known single target FPGA

- Explore mapping based on directive sets
- User in the loop

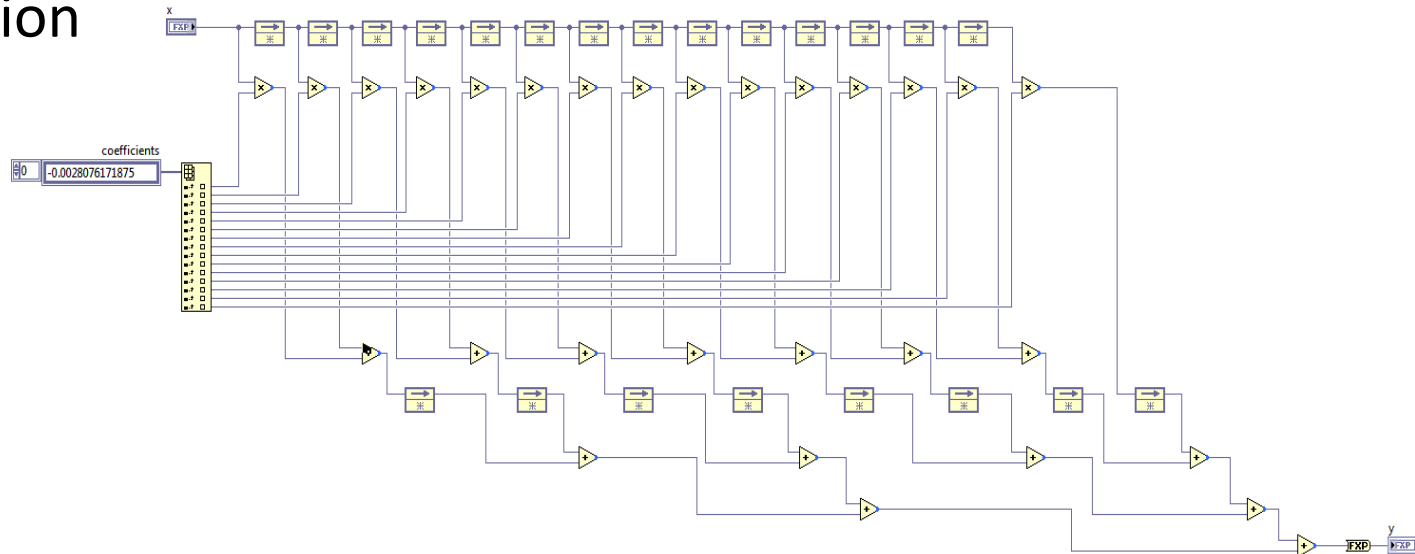
- Code generation
- Configured reusable IP

Current Challenges Programming in G

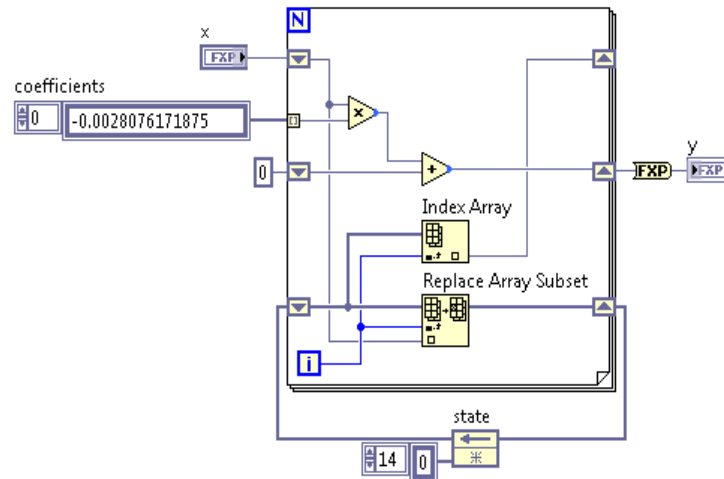
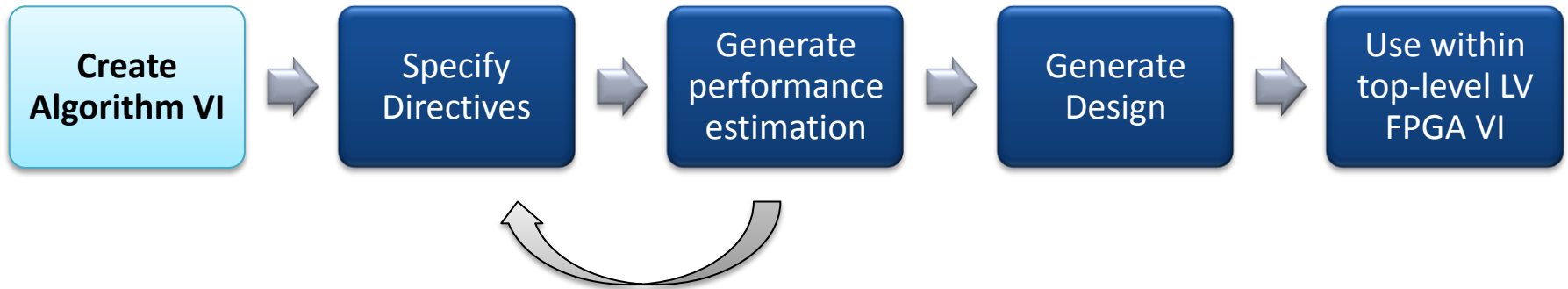
FIR filter in regular LabVIEW:



G implementation
after manual
unrolling and
pipelining:

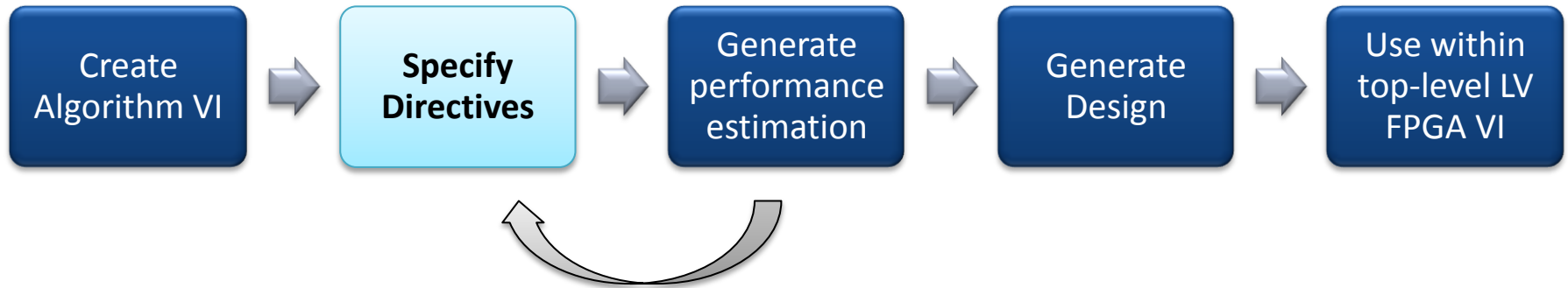


LabVIEW FPGA IP Builder User Flow



- Use dataflow programming
- Limited functions palette

LabVIEW FPGA IP Builder User Flow



Block Diagram Components

- FIR.vi (Top-level VI)
 - Clock rate (MHz): 200.00
 - Initiation interval (cycles): 1
 - Inline subVIs: True
 - Interface
 - y: Data
 - x: Data
 - Array[15]: Feedback node
 - Array Constant[15]: coefficients
 - For Loop: MainLoop
 - Multiply

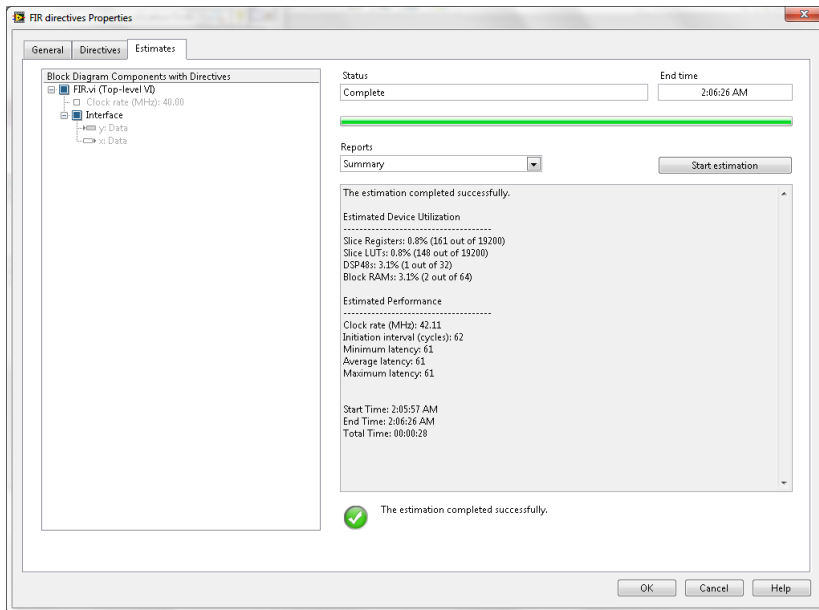
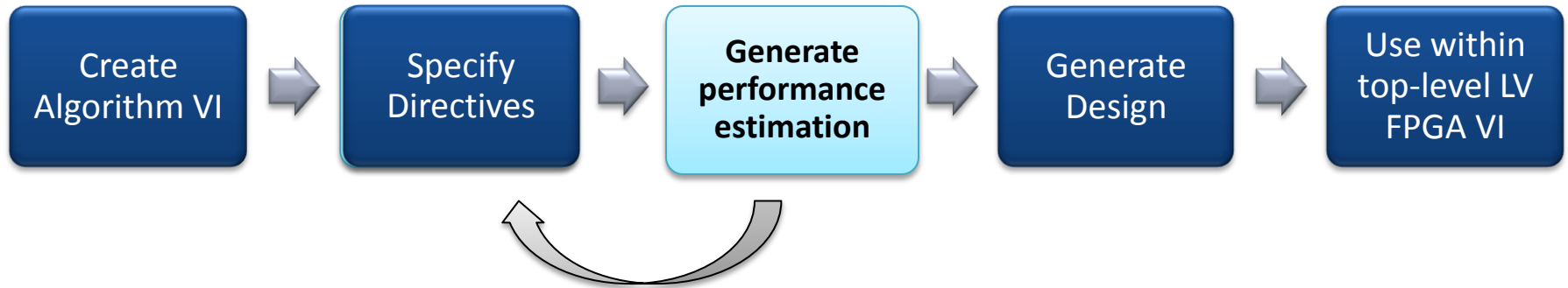
Show all directives

Find on block diagram

Directives	Value
<input checked="" type="checkbox"/> Clock rate (MHz)	200.00
<input type="checkbox"/> Share multipliers	
<input checked="" type="checkbox"/> Initiation interval (cycles)	1
<input type="checkbox"/> Minimum latency	
<input type="checkbox"/> Maximum latency	
<input checked="" type="checkbox"/> Inline subVIs	True
<input type="checkbox"/> Inline recursively	

Directives	Value
<input checked="" type="checkbox"/> Clock rate (MHz)	40.00
<input checked="" type="checkbox"/> Share multipliers	True
<input checked="" type="checkbox"/> Pipeline initiation interval	160
<input type="checkbox"/> Minimum latency	
<input type="checkbox"/> Maximum latency	
<input type="checkbox"/> Inline subVIs	
<input type="checkbox"/> Inline recursively	

LabVIEW FPGA IP Builder User Flow



Reports

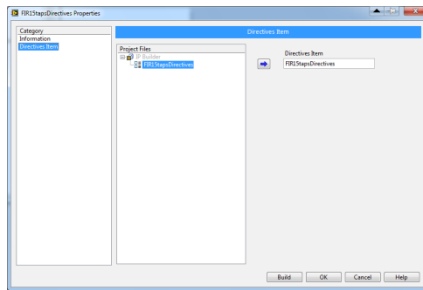
Estimated performance

Type	Requested	Estimated
FIR15staps_FXP.vi (Top-level VI)		
Clock rate (MHz)	200.00	274.73
Throughput (cycles/sample)		1
Minimum latency		5
Average latency		5
Maximum latency		5
Pipeline initiation interval	1	1
Pipeline depth		6

LabVIEW FPGA IP Builder User Flow



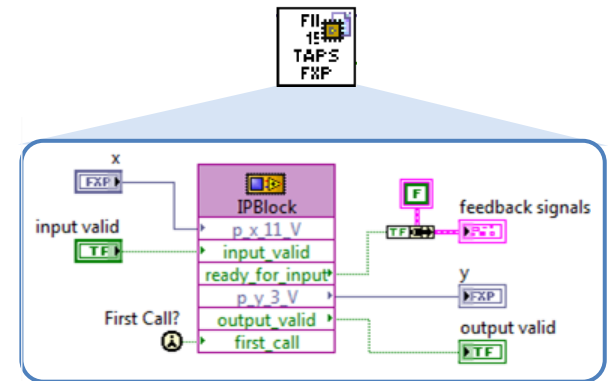
Choose VI and Directives



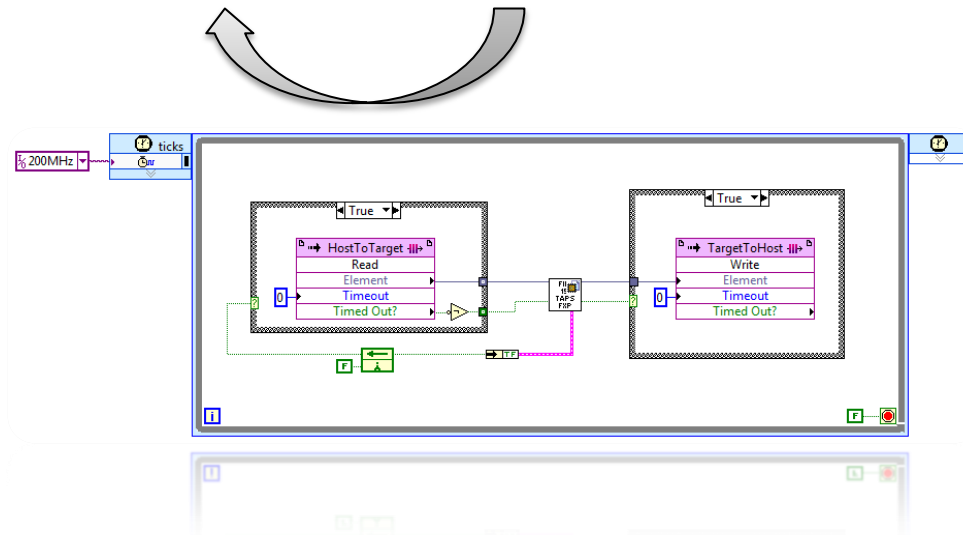
Generate HDL



Create IP VI



LabVIEW FPGA IP Builder User Flow



- Integrate into Single-Cycle Timed Loop
- Add I/O, DMA FIFOs, Host Communication, etc.

Exploration Example - Directives

$$\text{Throughput} = (\text{ClockRate} * \text{SamplesPerIteration}) / \text{InitiationInterval},$$

Reports
Summary Start estimation

The estimation completed successfully.

Estimated Device Utilization


- Slice Registers: 0.8% (161 out of 19200)
- Slice LUTs: 0.8% (148 out of 19200)
- DSP48s: 3.1% (1 out of 32)
- Block RAMs: 3.1% (2 out of 64)

Estimated Performance

- Clock rate (MHz): 40.11
- Initiation interval (cycles): 62
- Minimum latency: 61
- Average latency: 61
- Maximum latency: 61

Start Time: 11:31:33 PM
End Time: 11:32:01 PM
Total Time: 00:00:28

Directives	Value
<input checked="" type="checkbox"/> Clock rate (MHz)	40.00
<input type="checkbox"/> Share multipliers	
<input type="checkbox"/> Initiation interval (cycles)	
<input type="checkbox"/> Minimum latency	
<input type="checkbox"/> Maximum latency	
<input type="checkbox"/> Inline subVIs	
<input type="checkbox"/> Inline recursively	

 The estimation completed successfully.

Reports
Summary Start estimation

The estimation completed successfully.

Estimated Device Utilization


- Slice Registers: 0.9% (168 out of 19200)
- Slice LUTs: 0.8% (156 out of 19200)
- DSP48s: 3.1% (1 out of 32)
- Block RAMs: 3.1% (2 out of 64)

Estimated Performance

- Clock rate (MHz): 250.00
- Initiation interval (cycles): 77
- Minimum latency: 76
- Average latency: 76
- Maximum latency: 76

Start Time: 11:28:22 PM
End Time: 11:28:50 PM
Total Time: 00:00:28

Directives	Value
<input checked="" type="checkbox"/> Clock rate (MHz)	200.00
<input type="checkbox"/> Share multipliers	
<input type="checkbox"/> Initiation interval (cycles)	
<input type="checkbox"/> Minimum latency	
<input type="checkbox"/> Maximum latency	
<input type="checkbox"/> Inline subVIs	
<input type="checkbox"/> Inline recursively	

 The estimation completed successfully.

Reports
Summary Start estimation

The estimation completed successfully.

Estimated Device Utilization


- Slice Registers: 6.3% (1202 out of 19200)
- Slice LUTs: 2.7% (511 out of 19200)
- DSP48s: 46.9% (15 out of 32)
- Block RAMs: 0.0% (0 out of 64)

Estimated Performance

- Clock rate (MHz): 250.00
- Initiation interval (cycles): 1
- Minimum latency: 6
- Average latency: 6
- Maximum latency: 6
- Pipeline depth: 7

Start Time: 11:34:55 PM
End Time: 11:35:26 PM
Total Time: 00:00:30

Directives	Value
<input checked="" type="checkbox"/> Clock rate (MHz)	200.00
<input type="checkbox"/> Share multipliers	
<input checked="" type="checkbox"/> Initiation interval (cycles)	1
<input type="checkbox"/> Minimum latency	
<input type="checkbox"/> Maximum latency	
<input type="checkbox"/> Inline subVIs	
<input type="checkbox"/> Inline recursively	

 The estimation completed successfully.

Comparison

Example Set #		1	2	3
Directive Settings	Clock Rate (MHz)	40	200	200
	Initiation Interval (Cycles)	-	-	1
Estimated Performance	Clock Rate (MHz)	42	250	250
	Initiation Interval (Cycles)	62	77	1
	Latency (cycles)	61	76	6
	Throughput (MS/s)	0.68	3.25	250
Estimated Resource Utilization	Registers	0.80%	0.90%	6.30%
	LUTs	0.80%	0.80%	2.70%
	Multipliers	1	1	15
	Block RAMs	2	2	0
Estimate Time (s)		28	28	30

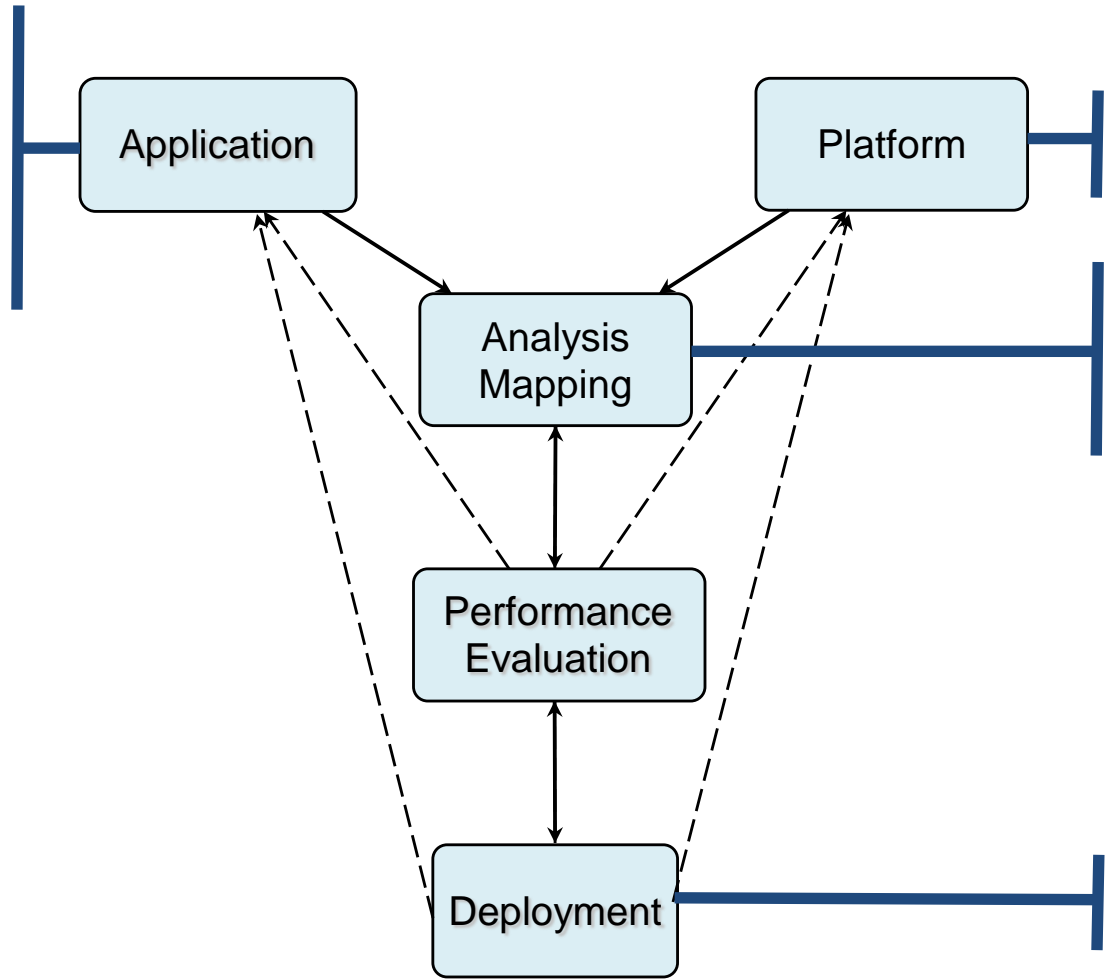
SYNTHESIS FROM ANALYZABLE DOMAIN SPECIFIC LANGUAGES

Approach

- Domain specific application language

- E.g. streaming applications in communications domain

- Prequalified IP Component Library



- Platform model
- Known single target FPGA

- Mapping based on domain specific intent
- Dataflow optimizations
- Hardware specific optimizations

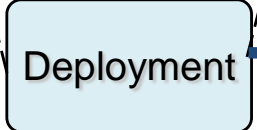
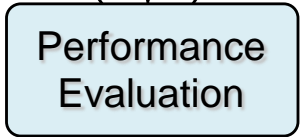
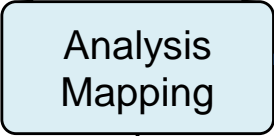
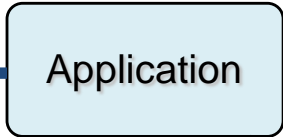
- Code generation
- Configured reusable IP

Approach

• Domain specific application language

- E.g. streaming applications in communications domain

• Prequalified IP Component Library

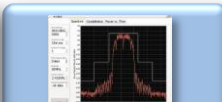


- Platform model
- Known single target FPGA

- Mapping based on domain specific intent
- Dataflow optimizations
- Hardware specific optimizations

- Code generation
- Configured reusable IP

RF Communications Applications

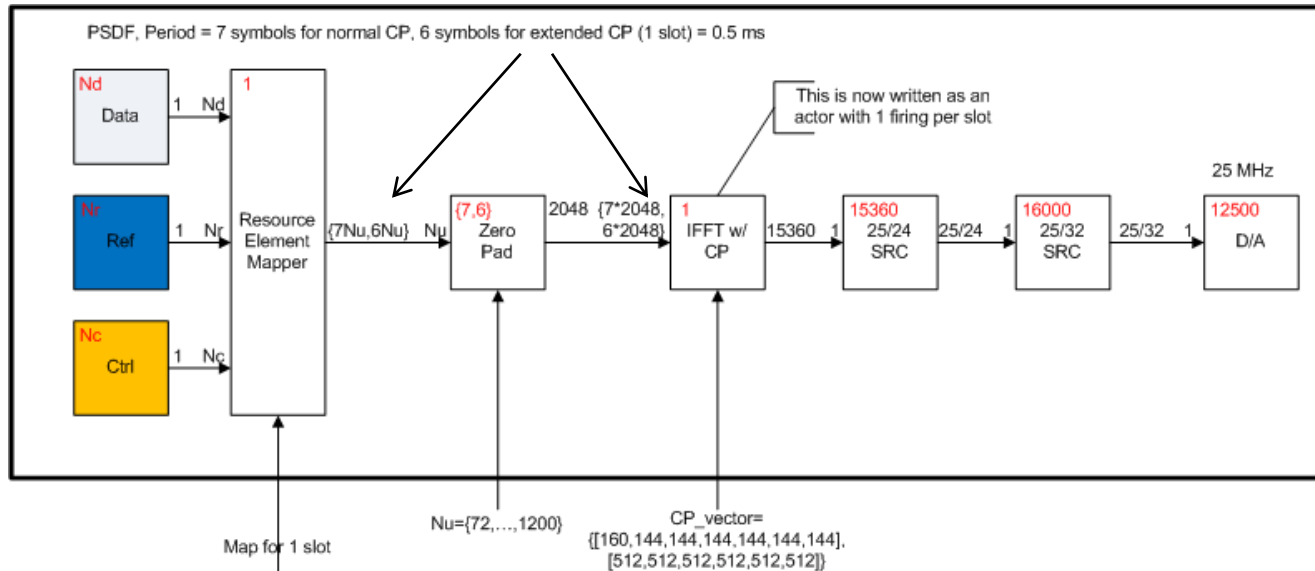


- Research**
- Network Optimization
 - Interference Alignment
 - Single Ant Diversity
 - Cognitive Radio

- Surveillance**
- Signal INT
 - Communications INT
 - Electronic INT
 - Spectrum Sniffer

- Advanced Test**
- RF Channel Emulation
 - BTS Emulation
 - Protocol Analysis
 - Next Gen Test

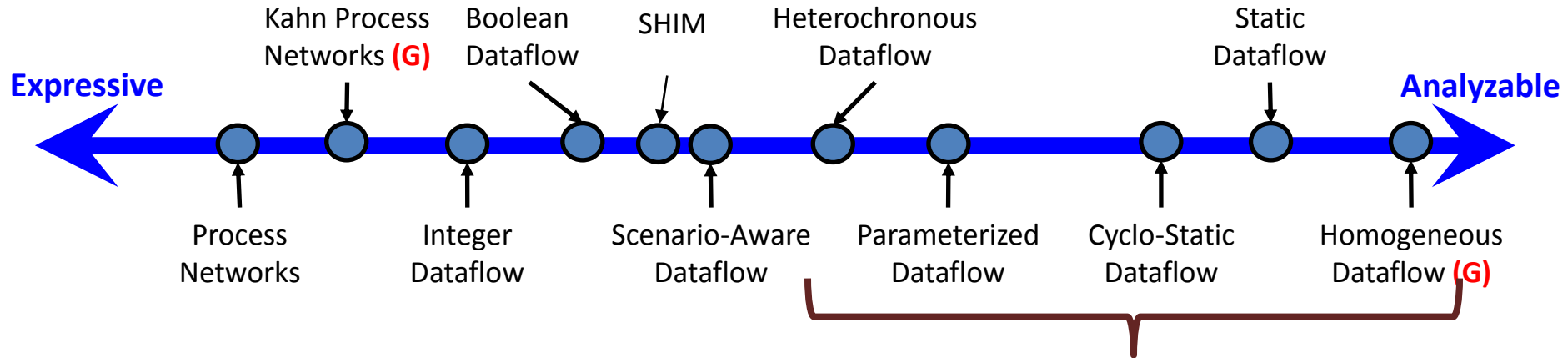
Streaming Model of the OFDM Transmitter



- $N_t = \{1, 2, 4\}$
 - Compile time - # transmitters
- $N_u = \{72, 180, 300, 600, 900, 1200\}$
 - 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?

MoCs for Streaming Applications



Area of focus for DSP Designer

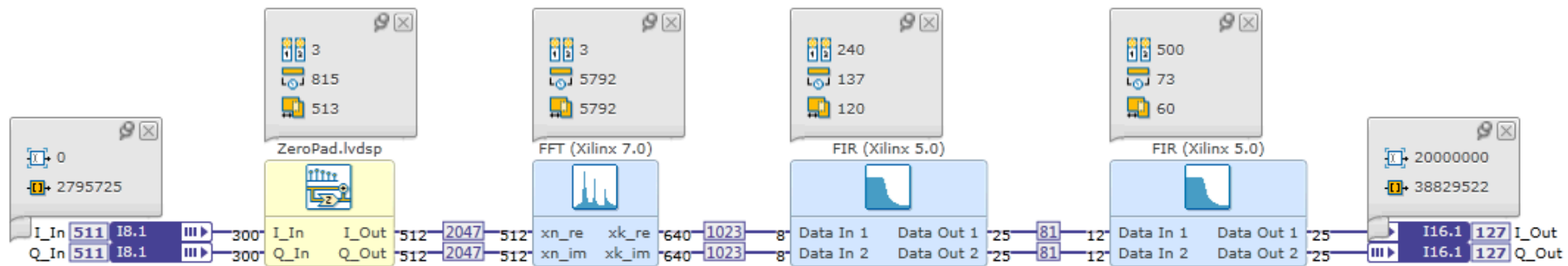
Deterministic?	No	Yes
Bounded data rates?	No	Yes
Deadlock and boundedness decidable?	No	Yes
Static scheduling?	No	Yes

Key trade-off: Analyzability vs. Expressability

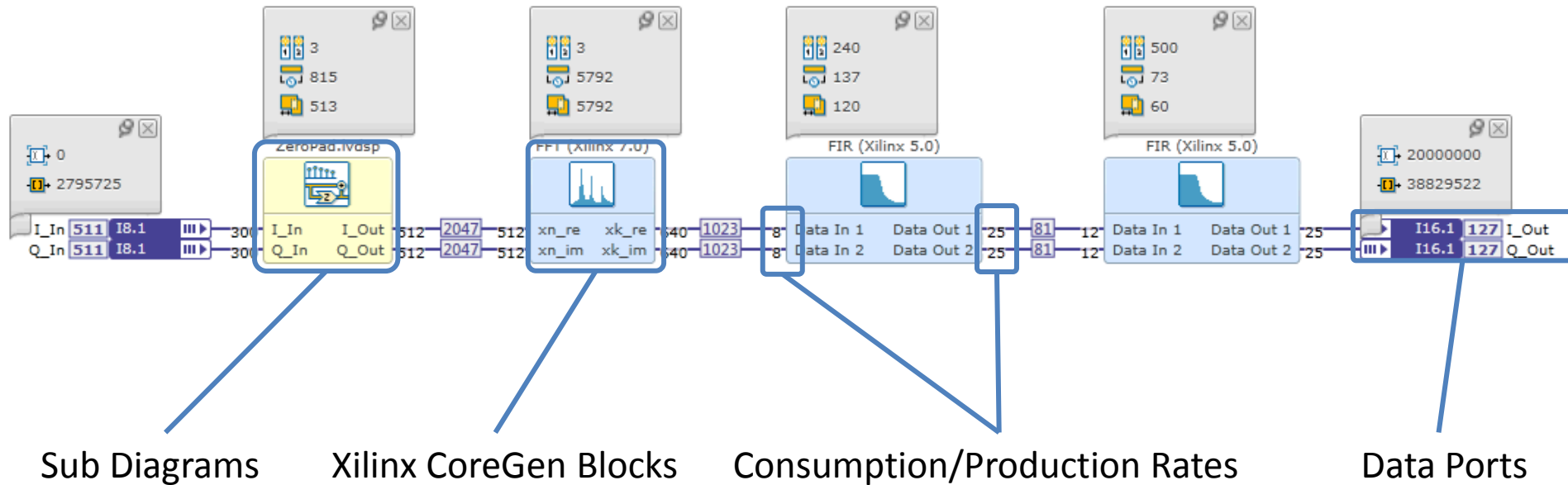
[1] Edward A. Lee, "Concurrent Models of Computation for Heterogeneous Software", EECS 290, 2004.

[2] Stephen Edwards, "SHIM: A Deterministic Model for Heterogeneous Embedded Systems", UCB EECS Seminar, 2006.

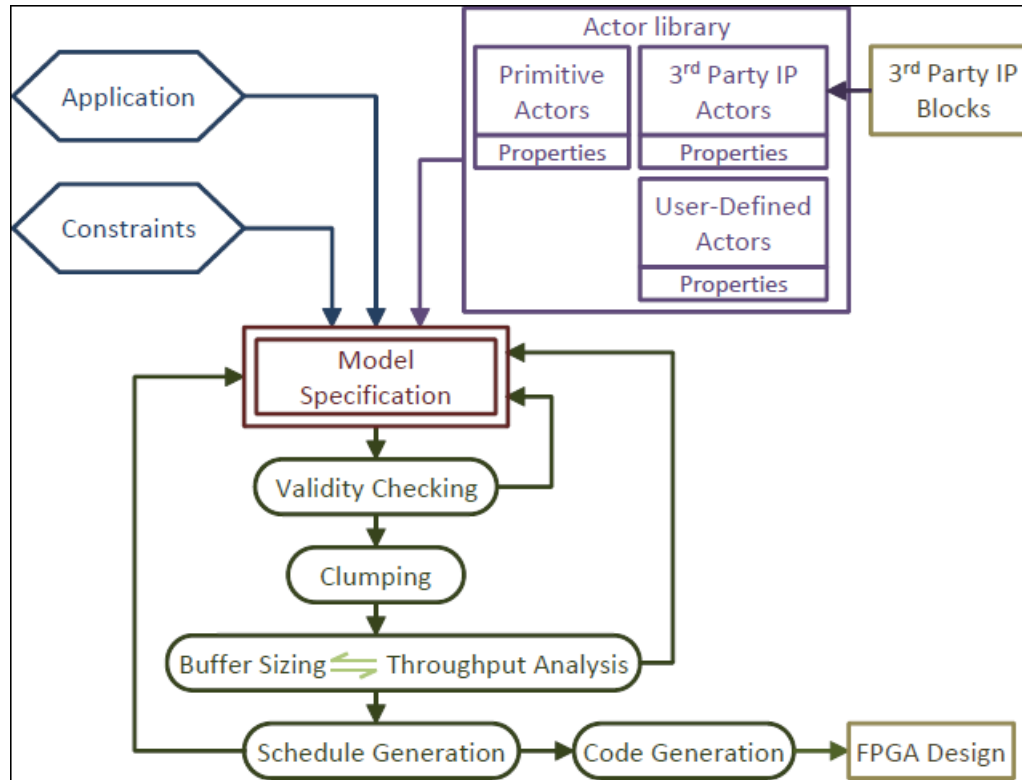
Application Specification in LabVIEW DSP Design Module



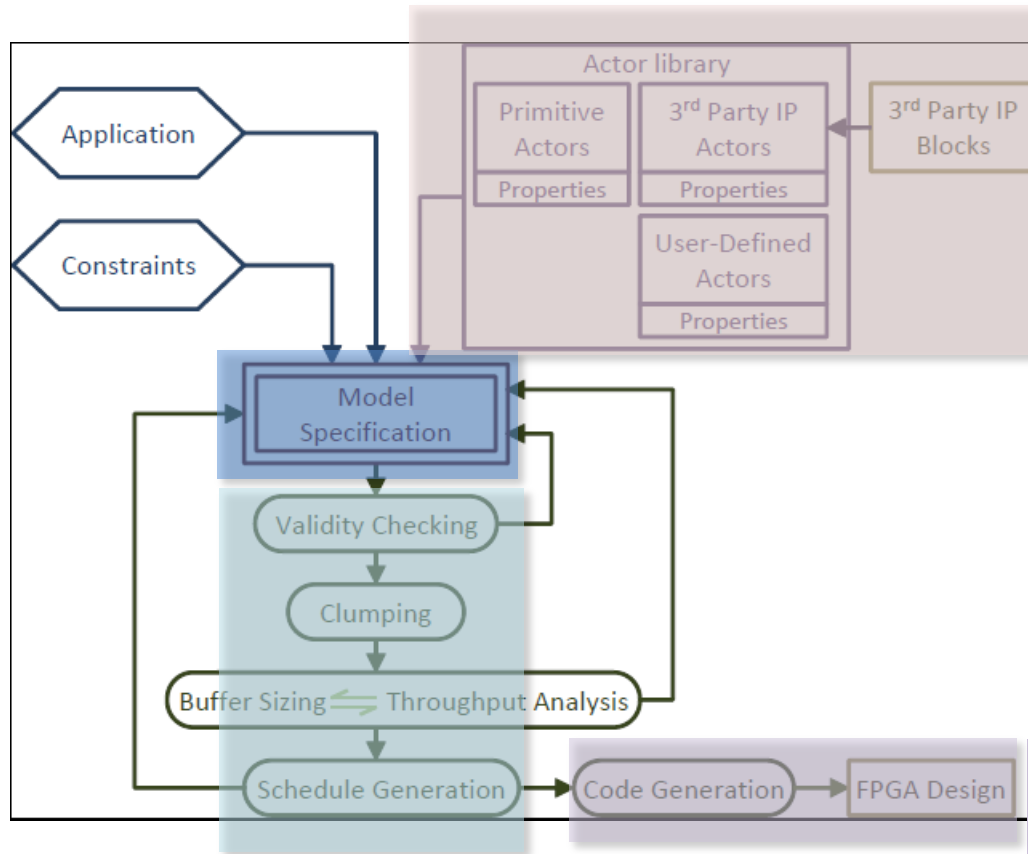
Application Specification in LabVIEW DSP Design Module



Tool Flow



Tool Flow



Models of
Computation

Analysis and
Optimization
Back End

Simulation and
Verification

- Actor Definition
- Performance Models and Timing Library
- IP Modeling and Integration

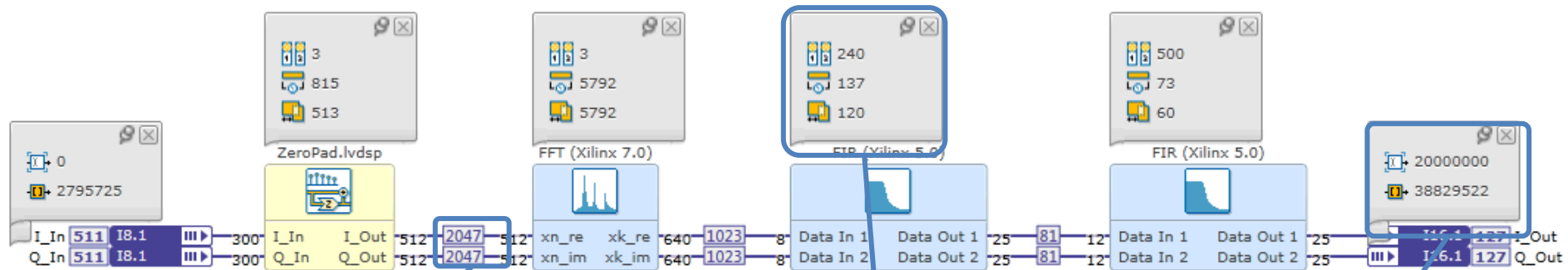
- Code Generation and Implementation

Analysis and Optimization Features

- Core dataflow optimizations
 - Model validation
 - Deadlock detection and boundedness check
 - 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.

Analysis and Optimization Results

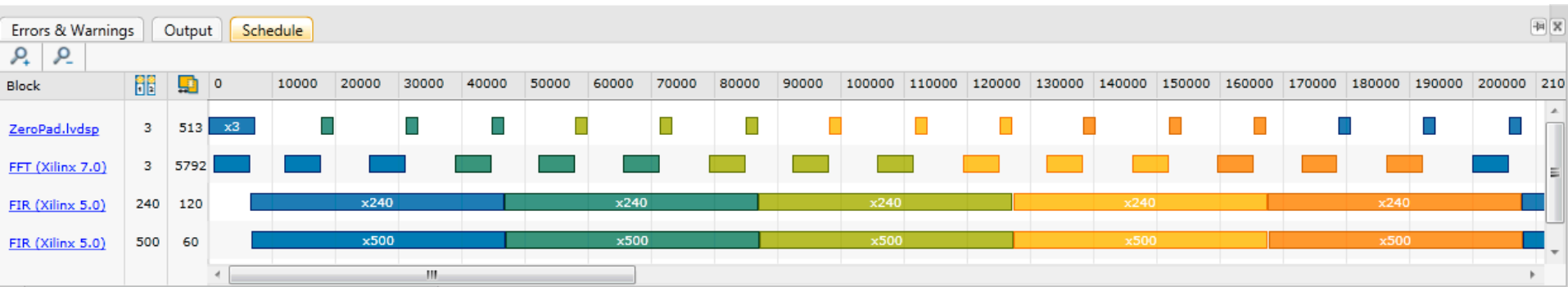
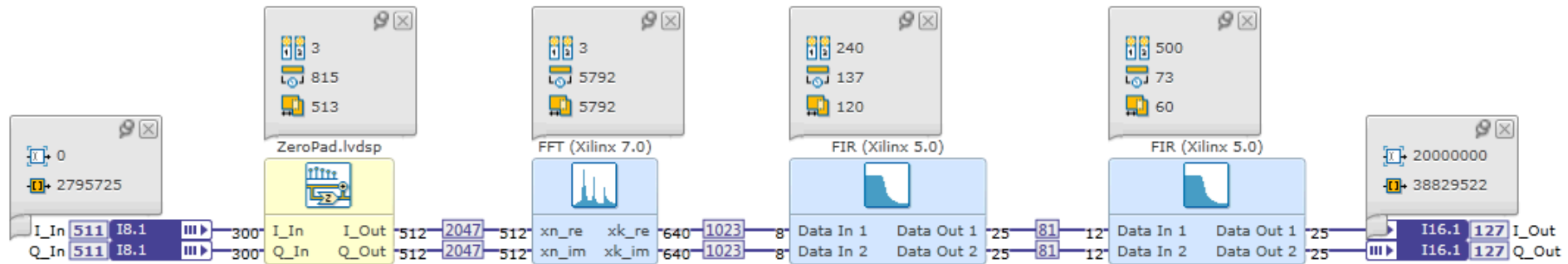


Auto buffer sizing to minimize resources

Calculated firing counts and timing data

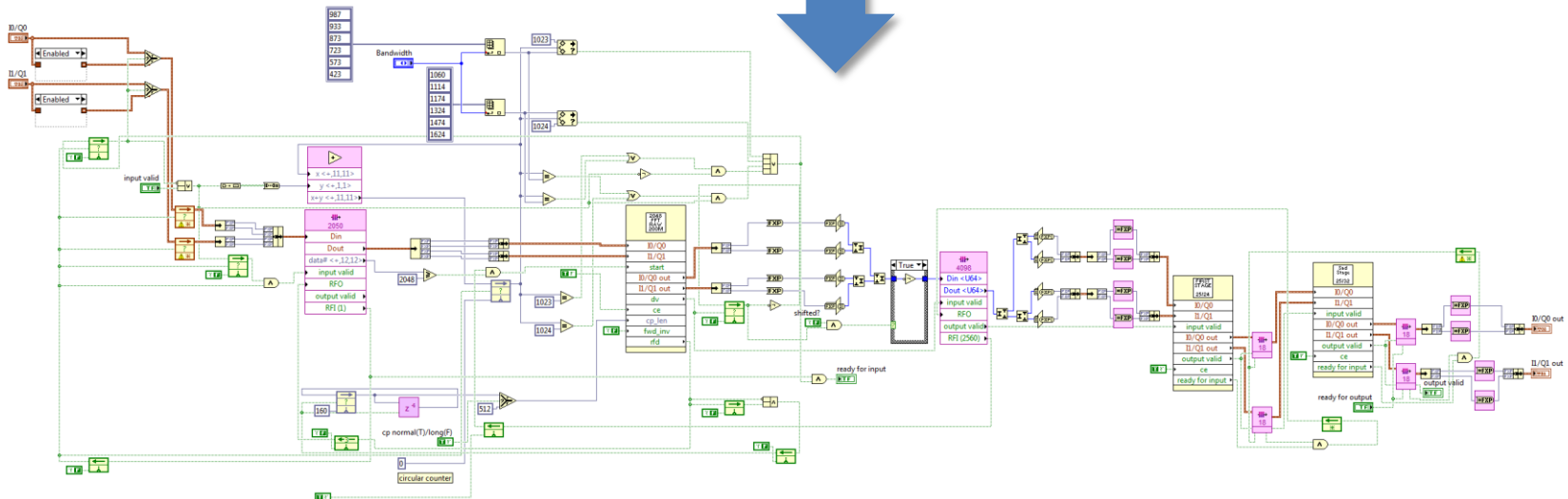
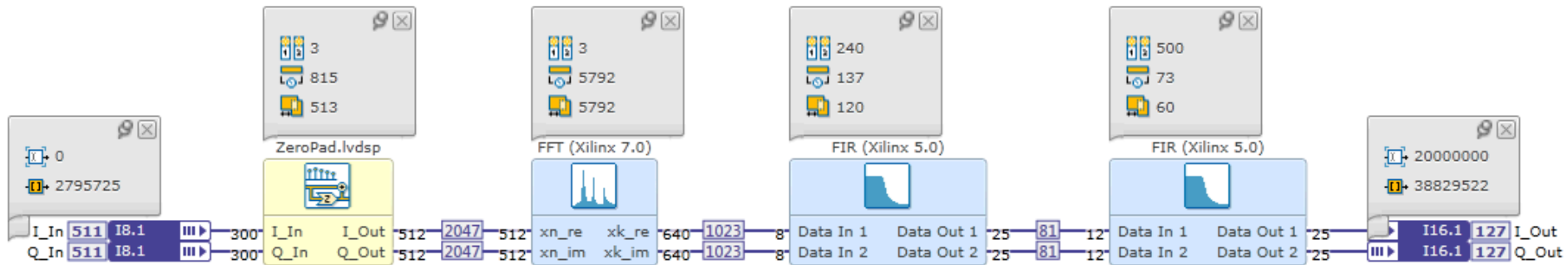
Throughput constraints

Analysis and Optimization Results

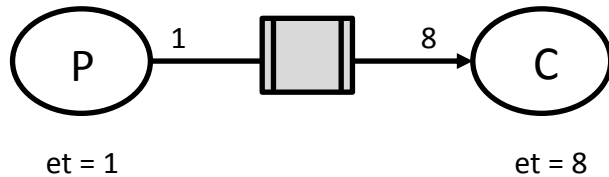


Calculated Schedule View

High-Level Model to FPGA blocks

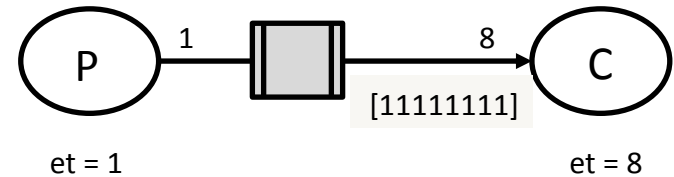


Extending SDF with Access Patterns



SDF

Optimal throughput = 1 sample/cycle
Buffer Size for optimal throughput = 16



SDF-AP

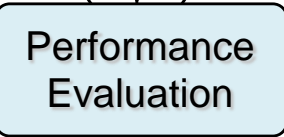
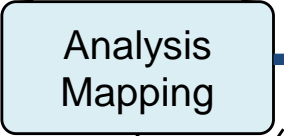
Optimal throughput = 1 sample/cycle
Buffer Size for optimal throughput = 2

- SDF-AP Model of Computation
 - Access Patterns viewed only by the tool, not the user
 - Formal syntax and operational semantics
 - Definitions of key model properties
 - Executability, boundedness and throughput
 - Case studies to evaluate these algorithms
 - Executability for SDF -AP model is decidable
 - Boundedness for SDF -AP model is decidable

SYSTEM-LEVEL SYNTHESIS TO HETEROGENEOUS TARGETS

Approach

- Application language
 - System Specifications & Constraints
 - Target agnostic
- Prequalified IP Component Library



- Platform model
 - General element library
 - Platform constraints

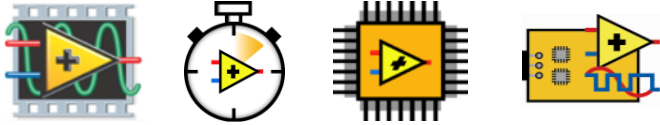
- General mapping model
- Mapping constraints
- Mapping objectives
- Optimization toolbox
- Platform selection

- System-level simulation, verification, validation


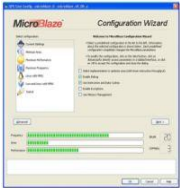
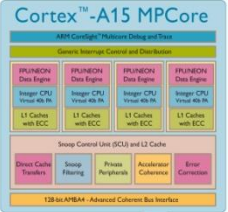


- Target code generation
- Performance results
- Cost results

Modern FPGA-based Distributed Heterogeneous Platforms

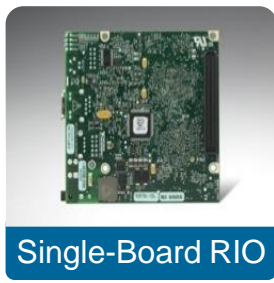
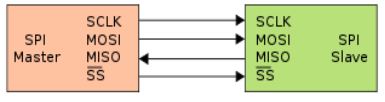
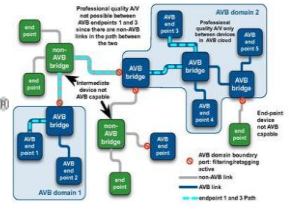
LabVIEW



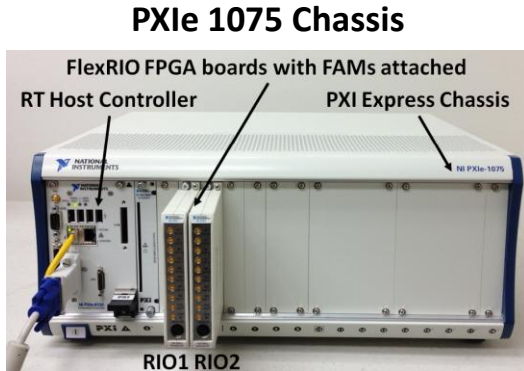
Desktop, RT, FPGA, uP, DSP Design, IP Builder

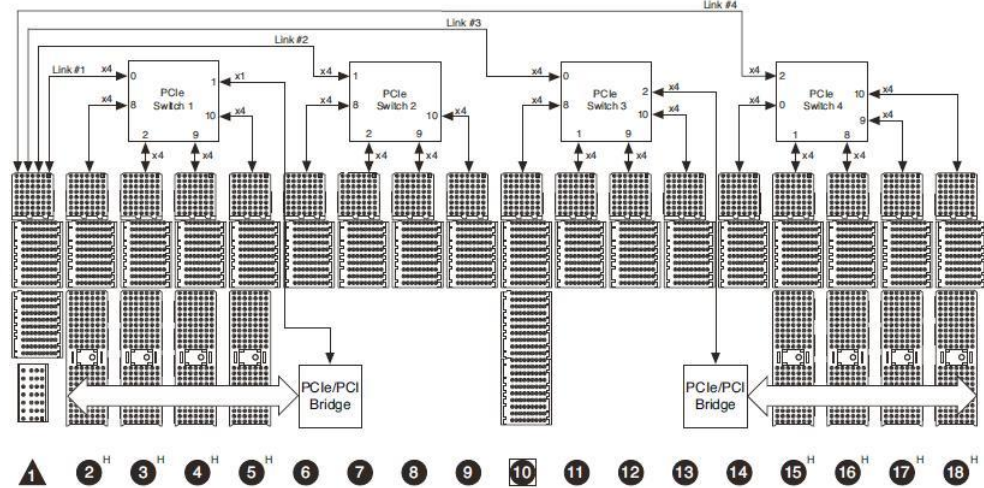
PCI EXPRESS



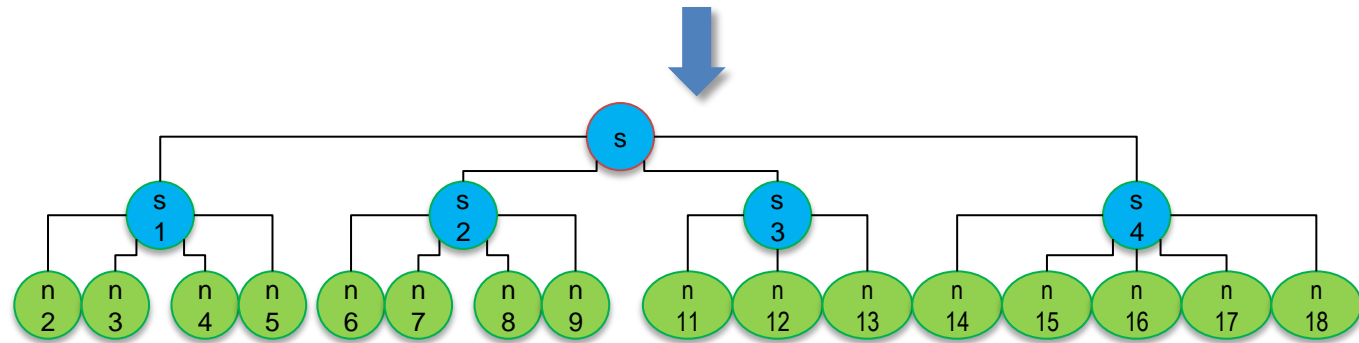
Platform Model



PXIe 1075 Internal Architecture



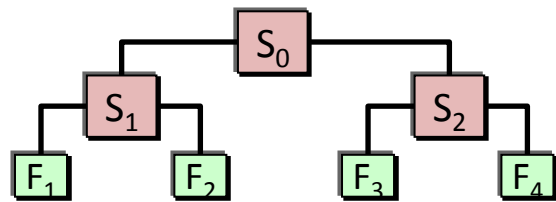
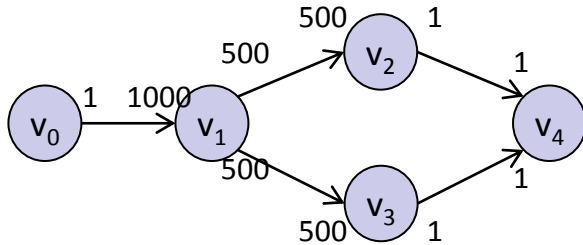
PXIe 1075 Communication Behavior Abstraction



On any edge, max bandwidth supported is 800 MB/s for unidirectional traffic, and 750 for bidirectional traffic for 128 byte packets. The bandwidth can be shared linearly among competing streams

Analysis and Mapping

Problem Inputs



Max BW: 800MB/s

Actor	v_0	v_1	v_2	v_3	v_4
II	1	1000	500	1000	1
Area	100	1000	2000	1500	100

PE	F_1	F_2	F_3	F_4
Area	3000	4000	2000	1000

Throughput: 250 MB/s

Frequency: 100MHz

Optimization problem

Enforce

- Bandwidth constraint per link/switch
- Area constraint per target
- Link bound constraint per link
- Affinity, grouping and exclusion

Optimize

- Throughput
- Area
- Latency

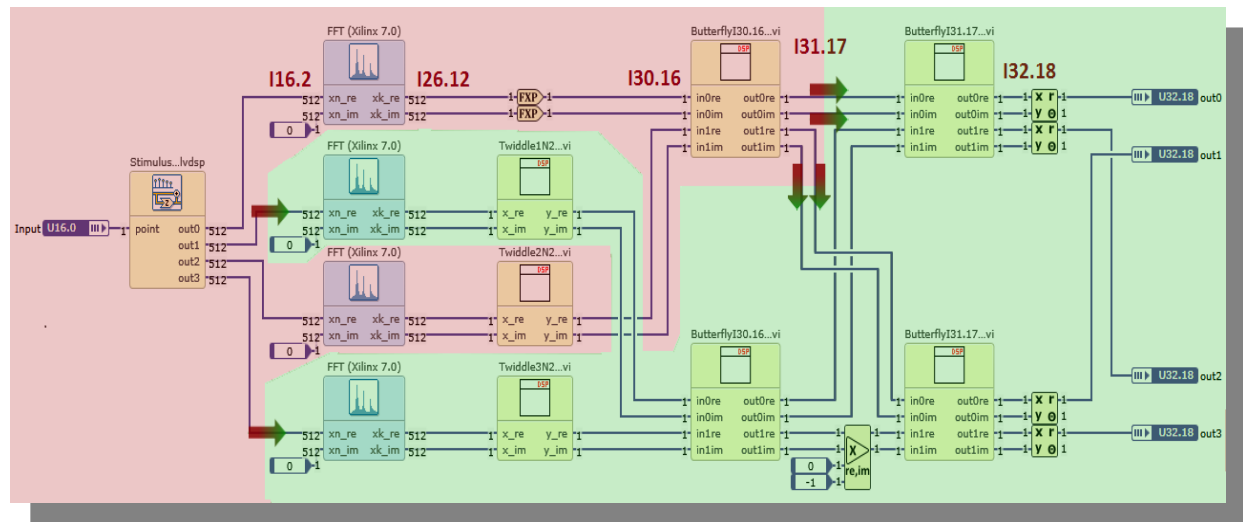
Solver and heuristic methods applied, and generate valid mappings

Results:

- Optimal mappings for OCT and OFDM models on 2- and 4- FPGAs in 1-3 mins
- Exact solver methods intractable for problems with over 50 tasks
- Heuristics within 20% of optimal; runtime of 1s for problems over 100 tasks

Experimental Results

- Platform and mapping model enabled experiments
- Static problem formulation (SMT solvers)
- Intuitive optimal results, sometimes even better than manual
- Reasonable scalability (alternate heuristics)



Summary

- Reviewed current trends in HLS
 - Viable method for *increasing* (FPGA or system) *design productivity* while still providing an *efficient implementation*
- Techniques Covered (Y-chart Methodology)
 - Directive-based synthesis
 - Synthesis from analyzable domain specific application models
 - System-level synthesis to heterogeneous targets
- These techniques can be combined hierarchically

Thank you

Questions? Comments?

Acknowledgements

This work was done in collaboration with:

Arkadeb Ghosal, Rhishikesh Limaye, Douglas Kim, Jeff Correll, Jacob Kornerup, Ian Wong, Guoqiang Wang, Guang Yang, Amal Ekbal, Mike Trimborn, Ankita Prasad, Trung N Tran, and Randy Allen.