

High Performance Flexible DSP Infrastructure Based on MPI

Tom McClean and *Steve Shank*

Lockheed Martin, Naval Electronics and Surveillance Systems (NESS)

Phone: (856)-914-6376

Fax: (856)-914-6815

Email: tom.mcclean@lmco.com

Email: steve.f.shank@lmco.com

Abstract:

Lockheed Martin has developed a platform independent, scalable and reconfigurable Digital Processor (DP) infrastructure for use in multiprocessor environments. This infrastructure is in use within the Small System Processor (SSP) program. This infrastructure provides communication, data flow, processor/algorithm scaling and configuration flexibility. All aspects of communication and processing are reconfigurable without the need to recompile. Pipeline, round robin, or hybrid processing architectures are supported, as well as modifying the number of processors without the need to recompile. This flexibility is provided by the use of text “flow graph” files, which describe a static processor mapping. Multiple flow graphs are supported.

A non-blocking multicast API is also provided. This is used to distribute the DP Stimulus messages to only the processors that are required to participate in processing.

The communication infrastructure provides an efficient mechanism, which decouples algorithm development from the specific details of the data distribution. Algorithm data flow routines support redistributing data from M to N processors with or without data overlap or minimum block sizes. Also provided are M to N corner turn and algorithm corner turn routines. Blocking and Non Blocking API's are provided.

This infrastructure is highly portable. The infrastructure was developed on CSPI 2841 multiprocessors using MPI as the underlying communication API and VSIPL as the Vector math library. Because it is based on industry standard API's, this infrastructure can be run on any platform that supports these API's. This has been validated on Server Class as well as Embedded platforms. No change to code was made, just a recompile for the particular platform.

Report Documentation Page

Form Approved
OMB No. 0704-0188

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

1. REPORT DATE 20 AUG 2004	2. REPORT TYPE N/A	3. DATES COVERED -	
4. TITLE AND SUBTITLE High Performance Flexible DSP Infrastructure Based on MPI		5a. CONTRACT NUMBER	
		5b. GRANT NUMBER	
		5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S)		5d. PROJECT NUMBER	
		5e. TASK NUMBER	
		5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Lockheed Martin, Naval Electronics and Surveillance Systems (NESS)		8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR'S ACRONYM(S)	
		11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release, distribution unlimited			
13. SUPPLEMENTARY NOTES See also ADM001694, HPEC-6-Vol 1 ESC-TR-2003-081; High Performance Embedded Computing (HPEC) Workshop (7th)., The original document contains color images.			
14. ABSTRACT			
15. SUBJECT TERMS			
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT
a. REPORT unclassified	b. ABSTRACT unclassified	c. THIS PAGE unclassified	UU
			18. NUMBER OF PAGES 18
			19a. NAME OF RESPONSIBLE PERSON

High Performance Flexible DSP Infrastructure Based on MPI and VSIPL

*7th Annual Workshop on High Performance Embedded
Computing*

*MIT Lincoln Laboratory
23-25 Sept 2003*

Tom McClean
Lead Member
Engineering Staff



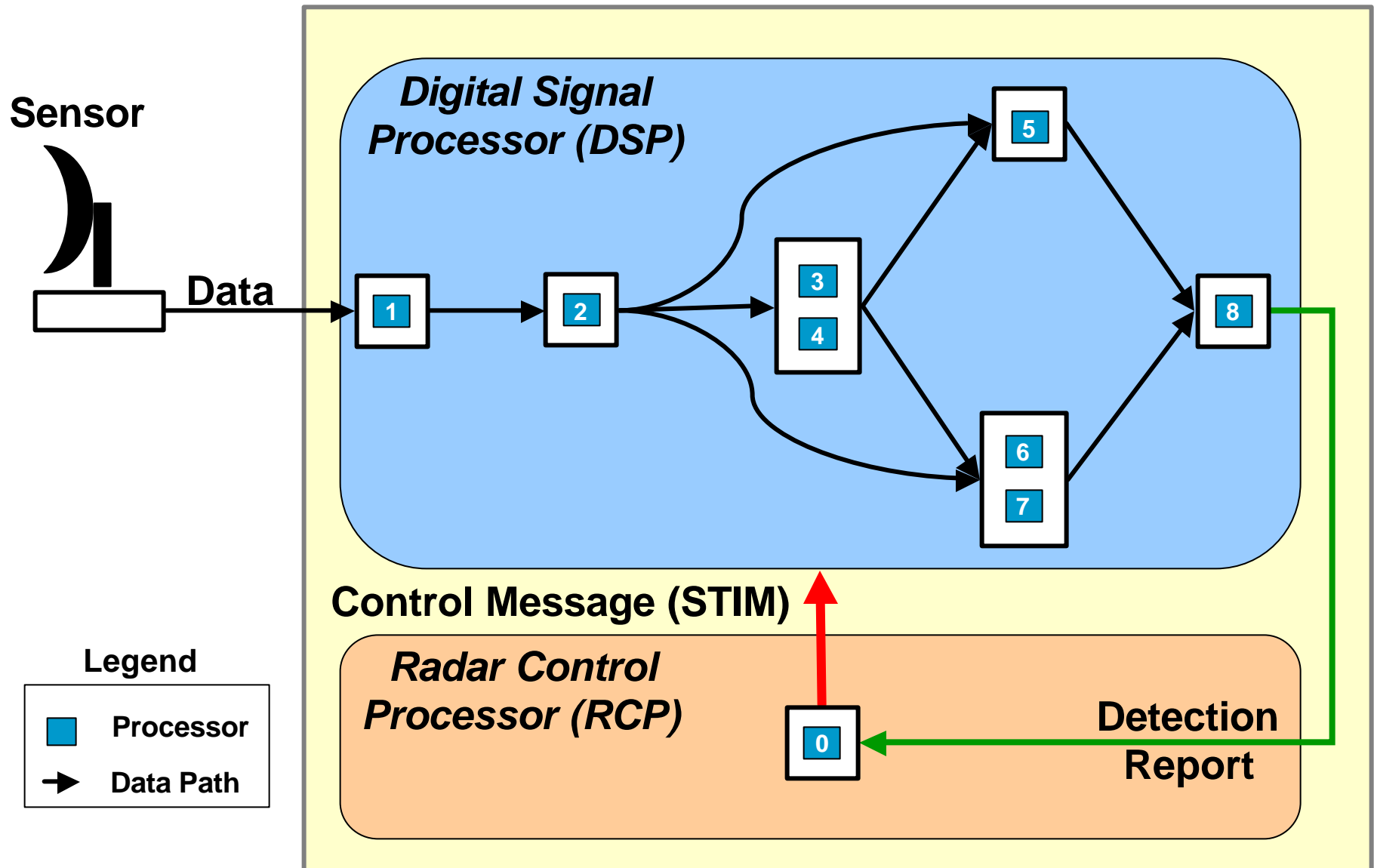
Hard Real Time DSP Challenge



- ***Develop a Portable and Easily Scalable DSP***
 - ***Portability requires the use of Open Architecture Standards***
 - ***Low Overhead Communication***
 - Message Passing Interface (MPI)
 - ***Vector Processing***
 - Vector Signal Image Processing Library (VSIPL)
 - ***Programming Language (C++)***
 - ***Scalability requires:***
 - ***An Infrastructure which is highly configurable.***
 - Number of Processors
 - Round Robin, Pipeline or Hybrid Data Flow
 - Data Redistribution Support
 - Frees the algorithm designer from the details of the data distribution

Open Architecture Standards allow for Platform Flexibility

Digital Processor Block Diagram



Real Time DSP Solution



- ***DSP Infrastructure Description***
 - ***Flow Graph***
 - ***Defines the Data Flow and Algorithm Mapping to a Network of Processors***
 - Based on a Static Map of DSP processors
 - Infrastructure Supports Multiple Flow Graphs
 - Text File Based (Easily Modified)
 - Loaded during Software Initialization
 - Easy to add algorithms or modify data flow
 - ***MPI Intercommunicators are formed based on Flow Graph information.***
 - ***Provides Stimulus and Data Flow Paths.***
 - ***Redistribution API uses the formed Data Paths.***
 - ***Infrastructure has been tested on Server and Embedded architectures using more than 64 processors.***
 - ***No code modification is needed.***
 - ***DSP recompiled for the particular architecture.***

Infrastructure is Platform Independent

Flow Graph Details



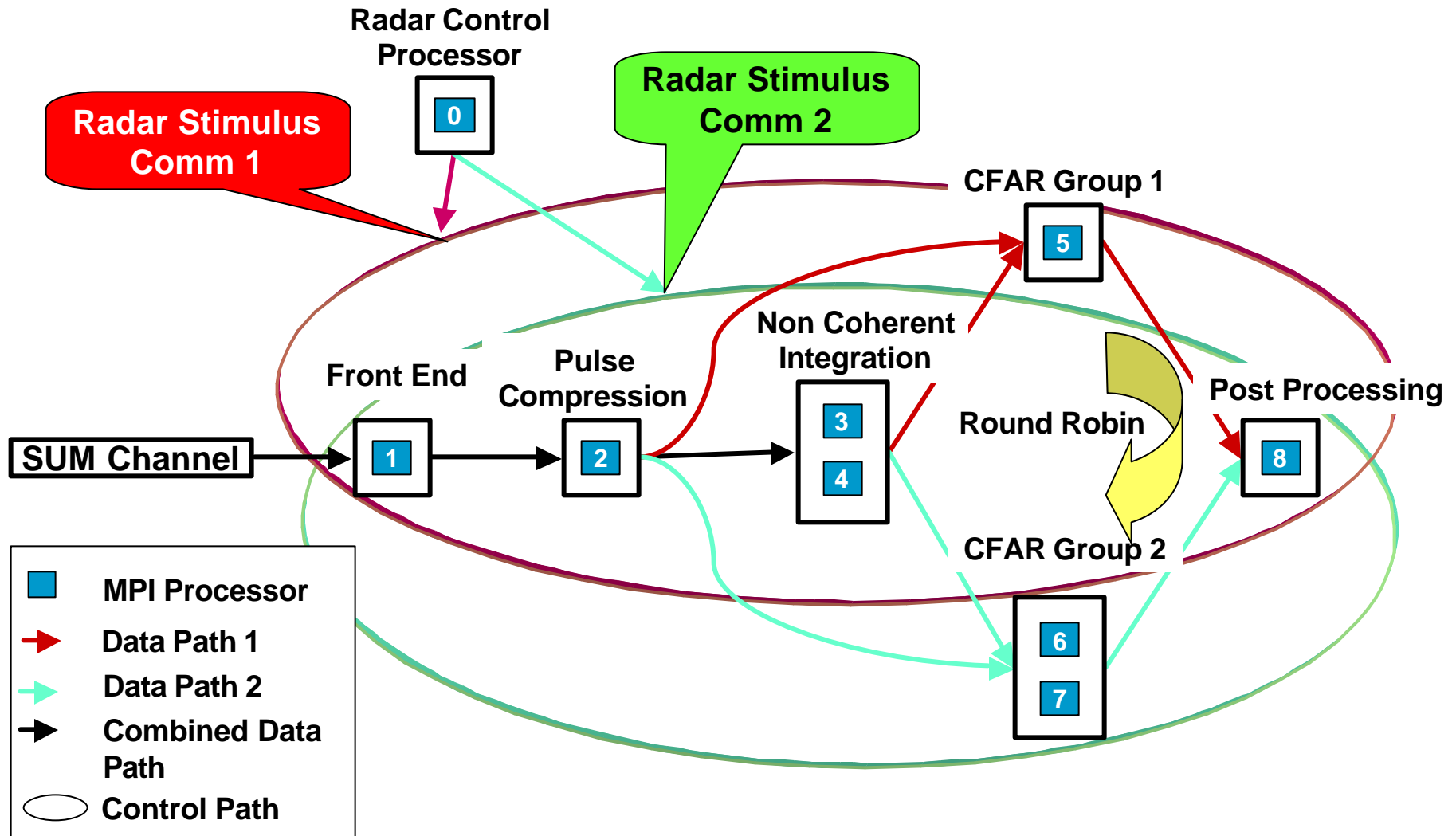
- ***MPI Stimulus and Data flow Communication paths are formed based on information read from text Flow_Graph files during initialization.***

<i>Example DSP Flow Graph</i>							
<i>Processor</i>	<i>MODE</i>	<i>Purpose</i>	<i>Group</i>	<i>Group Size</i>	<i>Group Count</i>	<i>Input</i>	<i>Output</i>
0	RCP	STIM	1	1	1	NONE	NONE
1	DSP	FE_SUM	1	1	1	NONE	PC_SUM
2	DSP	PC_SUM	1	1	1	FE_SUM	NCI,CFAR
3, 4	DSP	NCI	1	2	1	PC_SUM	CFAR
5	DSP	CFAR	1	1	2	PC_SUM,NCI	POST_PRO
6, 7	DSP	CFAR	2	2	2	PC_SUM,NCI	POST_PRO
8	DSP	POST_PRO	1	1	1	CFAR	NONE

Reconfiguration does not require code modification

Flow Graph Communicators

Resulting Stim and Data Communication Paths



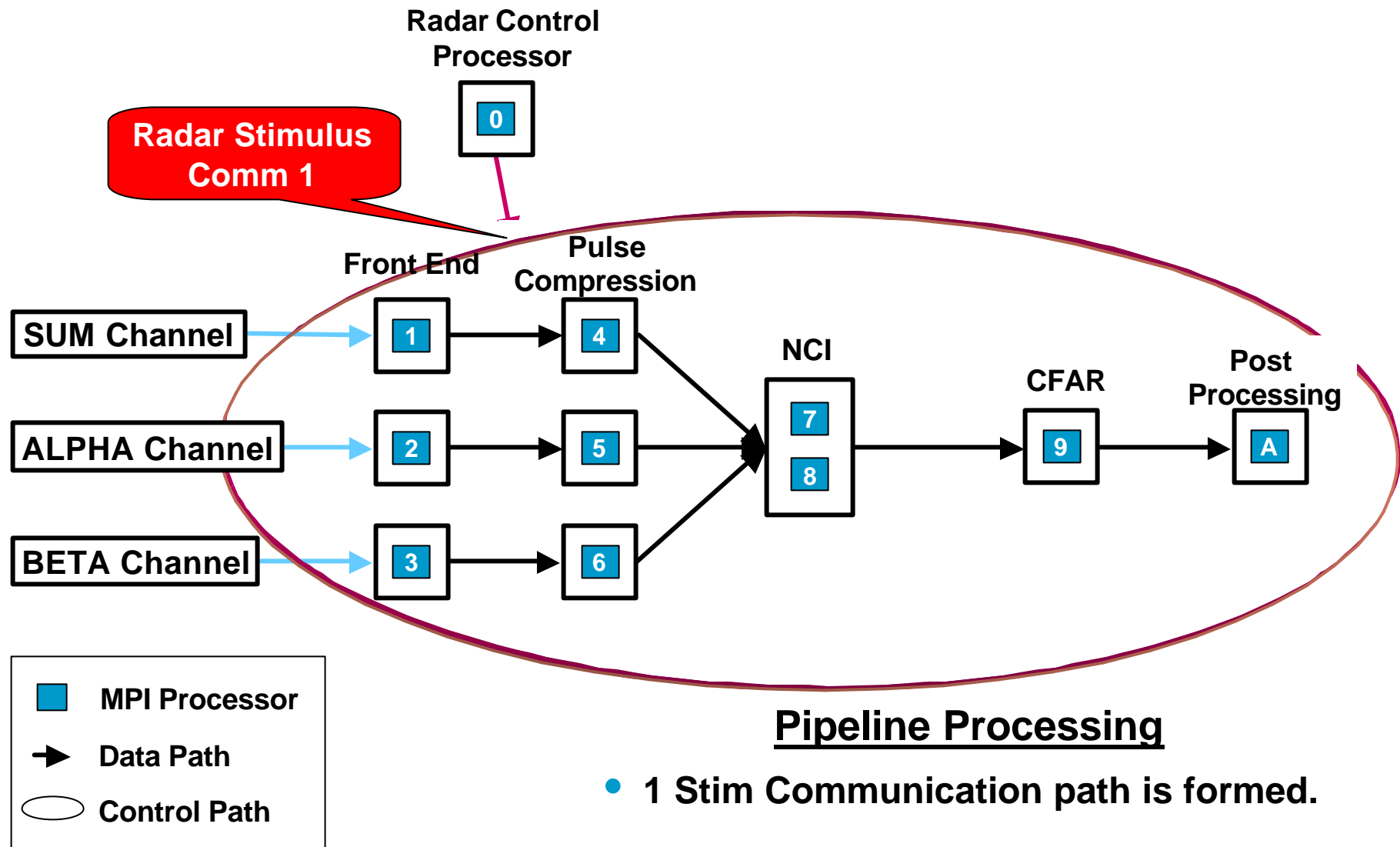
Pipeline Flow Graph



<i>Pipeline DSP Flow Graph</i>							
<i>Processor</i>	<i>MODE</i>	<i>Purpose</i>	<i>Group</i>	<i>Group Size</i>	<i>Group Count</i>	<i>Input</i>	<i>Output</i>
0	RCP	STIM	1	1	1	NONE	NONE
1	DSP	FE_SUM	1	1	1	NONE	PC_SUM
2	DSP	FE_AZ	1	1	1	NONE	PC_AZ
3	DSP	FE_EL	1	1	1	NONE	PC_EL
4	DSP	PC_SUM	1	1	1	FE_SUM	NCI
5	DSP	PC_AZ	1	1	1	FE_AZ	NCI
6	DSP	PC_EL	1	1	1	FE_EL	NCI
7,8	DSP	NCI	1	2	1	FE_SUM, FE_AZ, FE_EL	CFAR
9	DSP	CFAR	1	1	1	NCI	POST_PRO
A	DSP	POST_PRO	1	1	1	CFAR	NONE

Pipeline Processing

Resulting Control and Data Communication Paths



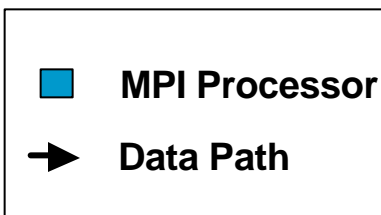
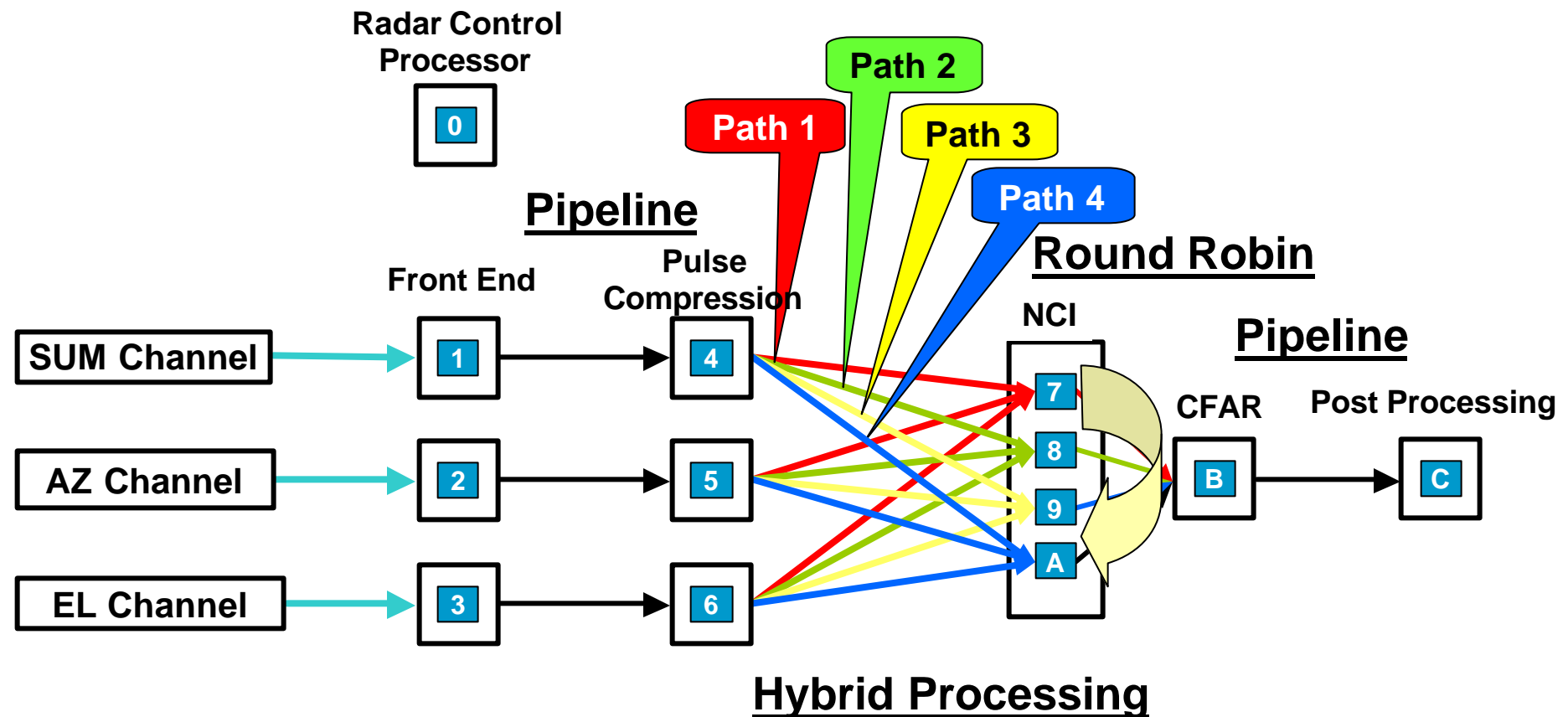
Hybrid Flow Graph



<i>Hybrid DSP Flow Graph</i>							
<i>Processor</i>	<i>MODE</i>	<i>Purpose</i>	<i>Group</i>	<i>Group Size</i>	<i>Group Count</i>	<i>Input</i>	<i>Output</i>
0	RCP	STIM	1	1	1	NONE	NONE
1	DSP	FE_SUM	1	1	1	NONE	PC_SUM
2	DSP	FE_AZ	1	1	1	NONE	PC_AZ
3	DSP	FE_EL	1	1	1	NONE	PC_EL
4	DSP	PC_SUM	1	1	1	FE_SUM	NCI
5	DSP	PC_AZ	1	1	1	FE_AZ	NCI
6	DSP	PC_EL	1	1	1	FE_EL	NCI
7	DSP	NCI	1	1	4	FE_SUM, FE_AZ,FE_EL	CFAR
8	DSP	NCI	1	1	4	FE_SUM, FE_AZ,FE_EL	CFAR
9	DSP	NCI	1	1	4	FE_SUM, FE_AZ,FE_EL	CFAR
A	DSP	NCI	1	1	4	FE_SUM, FE_AZ,FE_EL	CFAR
B	DSP	CFAR	1	1	1	NCI	POST_PRO
C	DSP	POST_PRO	1	1	1	CFAR	NONE

Hybrid Processing

Resulting Stim and Data Communication Paths



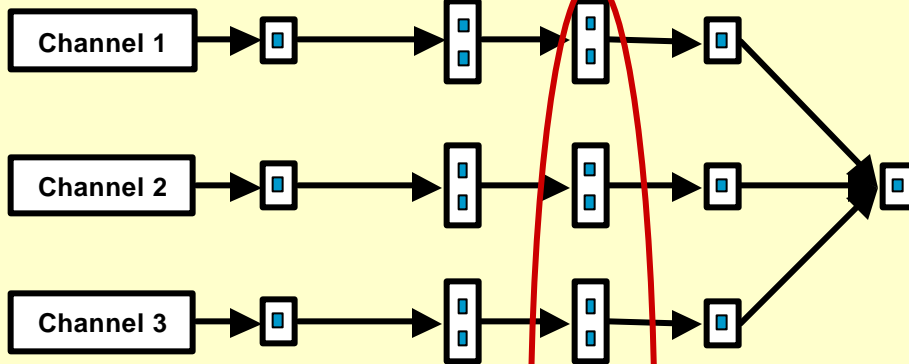
- 4 Distinct Communication paths are formed.
- A path is used per Radar Sequence. (ROUND ROBIN)
- Stim distributor determines which Comm path is in use.
- Stimulus is only distributed to processors that need it.

Multiple Flow Graphs

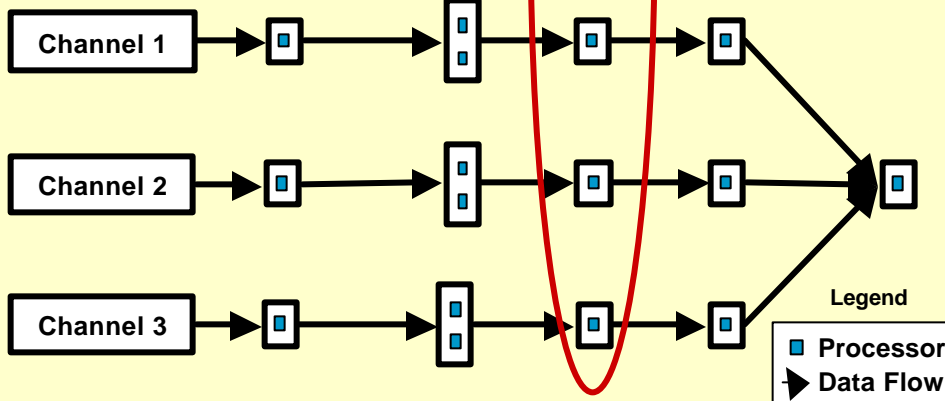


Receiver Interface Pipe 0 Pipe 1 Pipe 2 Pipe 3 Post Processing

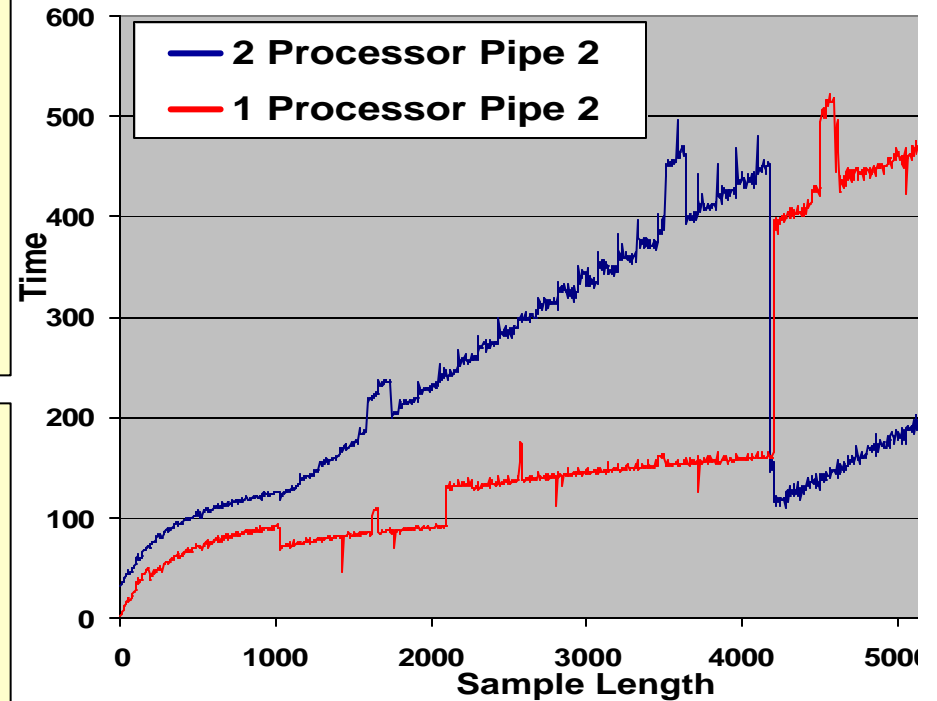
Flow Graph 1 Optimized for Large Sample Lengths



Flow Graph 2 Optimized for Small Sample Lengths



Flow Graph Performance Comparison



Each Flow Graph is Optimized for Radar Modes or Data Size

Data Redistribution



- ***Data Flow paths are used for Redistribution***
- ***Data Redistribution Calls are Layered on MPI***
- ***Provide 2D->2D with Overlap and Modulo support***
- ***Insulates Algorithm from Redistribution***

Algorithm Pseudo Code Fragment

Data Input From 2
Processors to 3
Processors

*VSIPL provides a
Platform independent
API*

Data Output From 3
Processors to 1
Processor

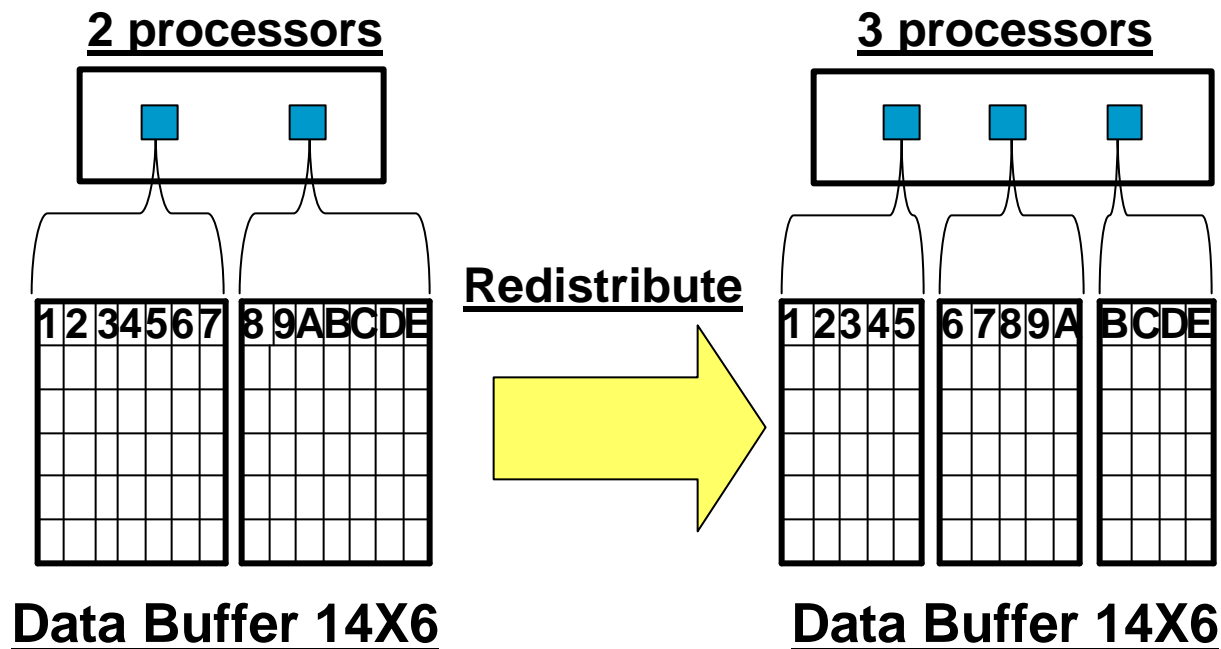
```
// Data input scalable across processors  
// Receive Input Data  
blocks = Redist( Buffer, 14, 23, 1, 0);  
  
// Perform algorithm on received data  
for( int i=0; i<blocks; i++)  
{  
    vsip_ccfftop_f(...);  
    ...  
}  
  
// Data output scalable across processors  
// Send Output Data  
blocks = Redist( Buffer, 14, 32, 1, 0);
```

Developer can Concentrate on Algorithm Implementation

Data Redistribution Without Overlap



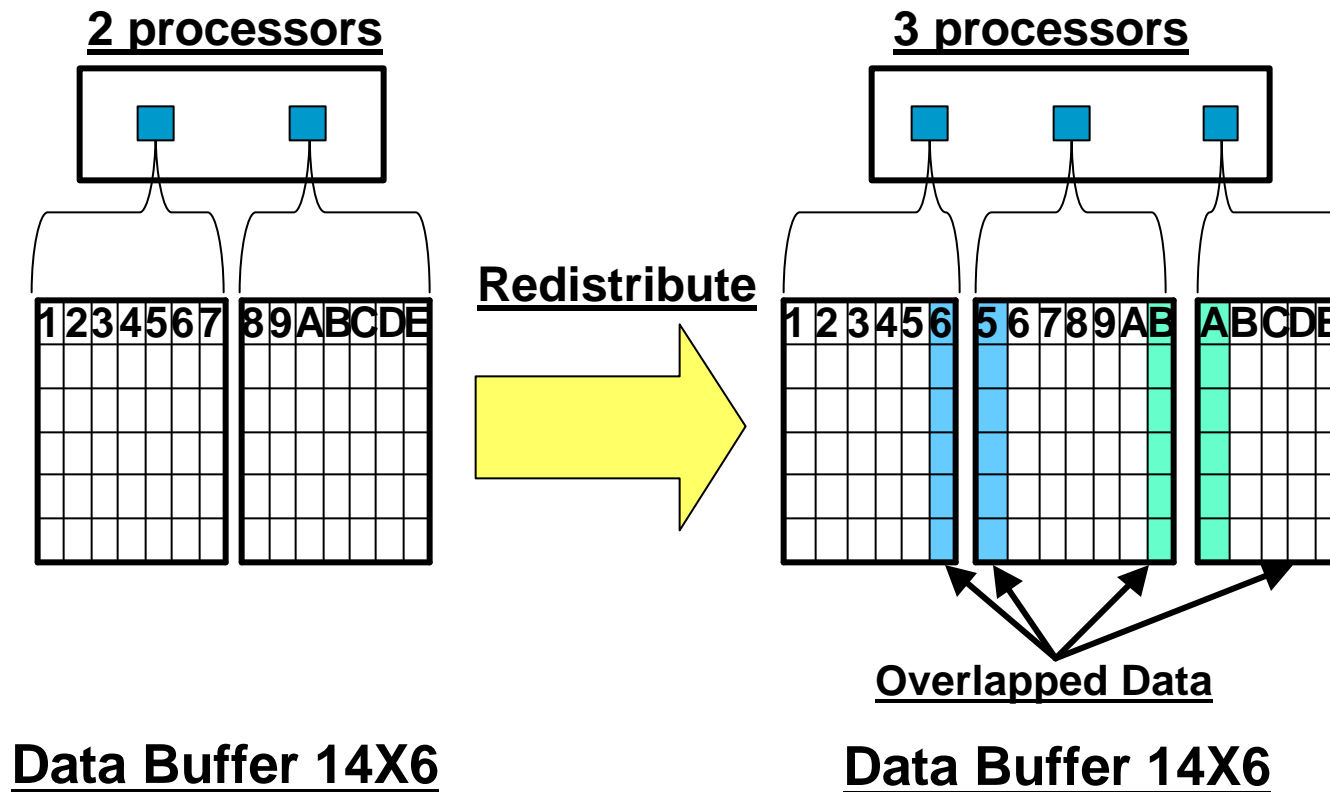
- *Data flow Communication paths are used for Redistribution*
 - *Data Redistribution Calls are Layered on MPI*
 - *Provide 2D->2D with Overlap and Modulo support*
 - *Insulates Algorithm from Redistribution*
- Redist(Data_Buffer, Splitable Dimension, Unsplitable Dimension, Modulo,Overlap);*
- Redist(Buffer, 14, 6, 1, 0); -Application Redistribution Call*



Data Redistribution With Overlap



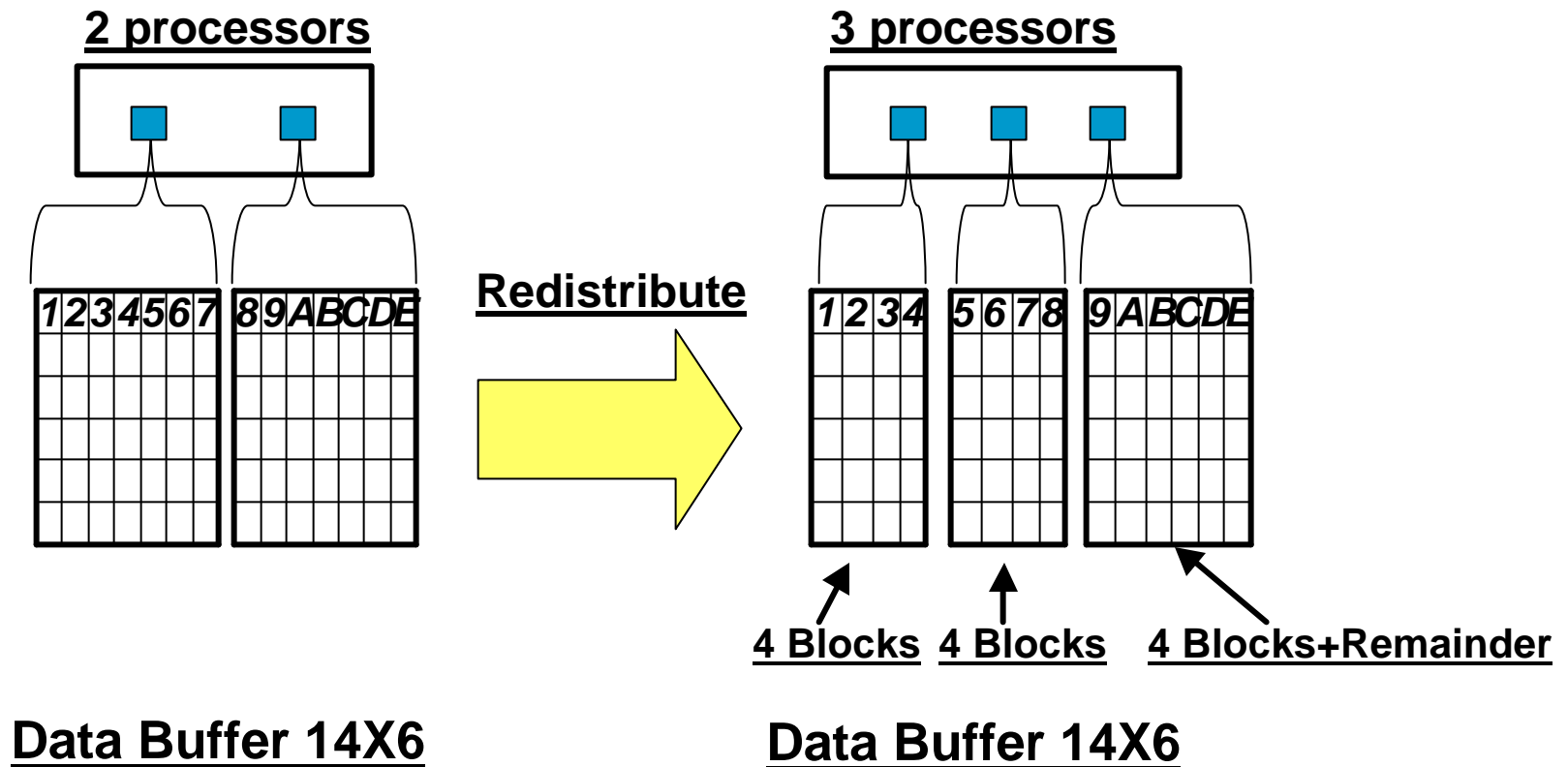
- *Redist(Data_Buffer, Splitable Dimension, Unsplitable Dimension, Modulo, Overlap);*
- *The Same Call is Made by all 5 Processors*
- *Redist(Buffer, 14, 6, 1, 1); -Application Redistribution Call With Overlap 1*



Data Redistribution With Modulo



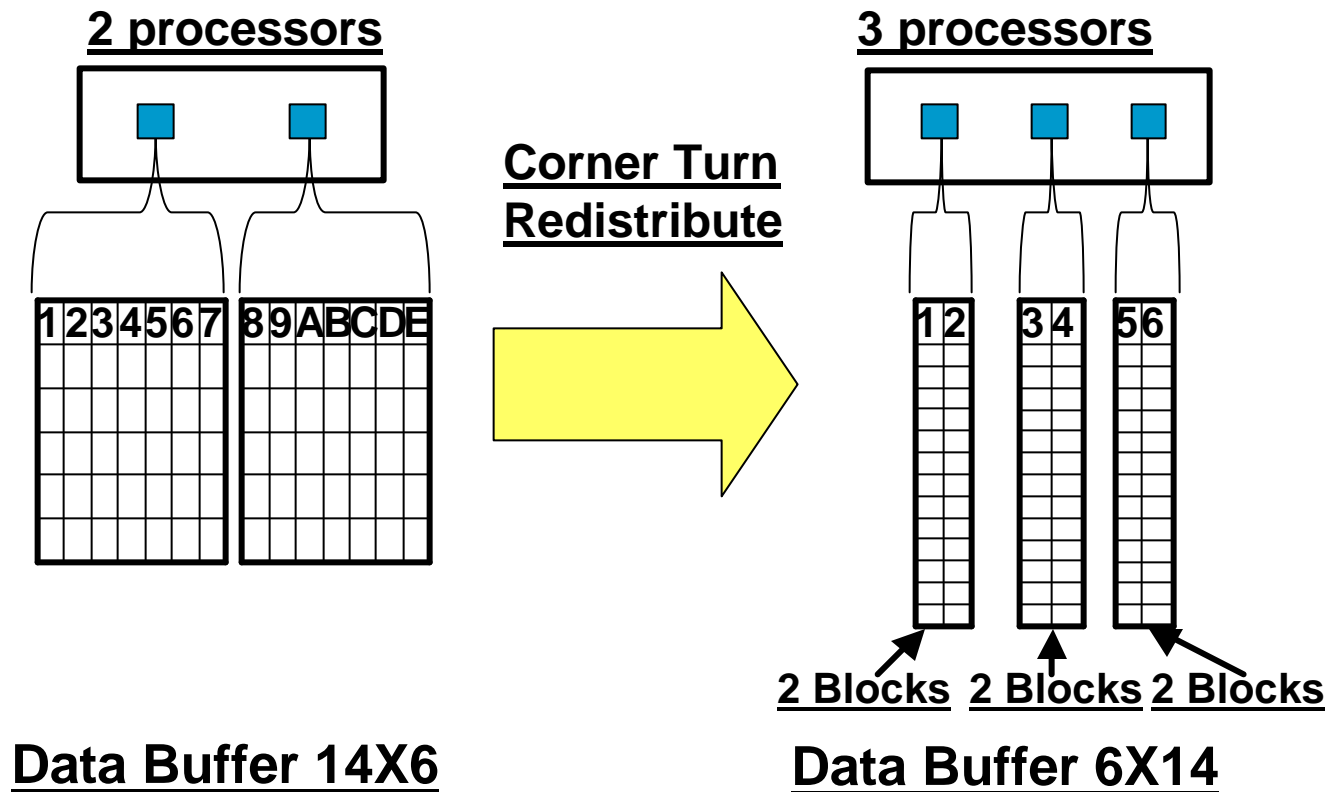
- *Redist(Data_Buffer, Splitable Dimension, Unsplitable Dimension, Modulo, Overlap);*
- *The Same Call is Made by all 5 Processors*
- *Redist(Buffer, 14, 6, 4, 0); -Application Redistribution Call With Modulo 4*



Matrix Transpose



- *Ct_Transfer(Data_Buffer, Splitable Dimension, Unsplitable Dimension, Modulo);*
- *The Same Call is Made by all 5 Processors*
- *Ct_Transfer(Buffer, 14, 6, 1); - Application Matrix Transpose*



Summary



- **DSP Infrastructure:**
 - **Supports Real-Time High-Performance Embedded Radar Applications**
 - **Low Overhead**
 - **Scalable to requirements**
 - **Built on Open Architecture Standards**
 - **MPI and VSIPL**
 - Reduces development costs
 - Scalable to applications with minimal changes to software
 - Provides for Platform Independence
 - **Provides DSP Lifecycle Support**
 - **Scale DSP from Development to Delivery Without Code Modifications**
 - **Add Algorithms with Minimal Software Changes**
 - **Reusable Infrastructure and Algorithms**
 - **Easily Scale DSP for Various Deployments**

Infrastructure Reduces Development Cost