Splash: Architecture & Application
January 28, 1998

- Admin
- Splash System Architecture
- Splash Board Architecture
- Edit Distance Program

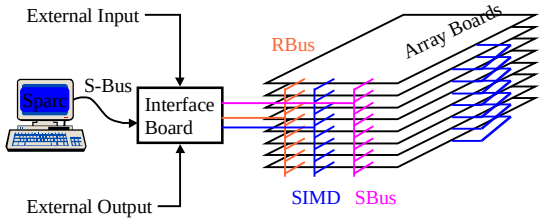
Admin

- Handin procedure revised
- Lab1 handin working ok?

Splash-2

- Attached processor system
- Reconfigurable processor supports 2 programming models:
  - Systolic
  - SIMD
- Programmed with “high-level” tools (e.g., VHDL, dbC)
- And, intimate knowledge of the architecture

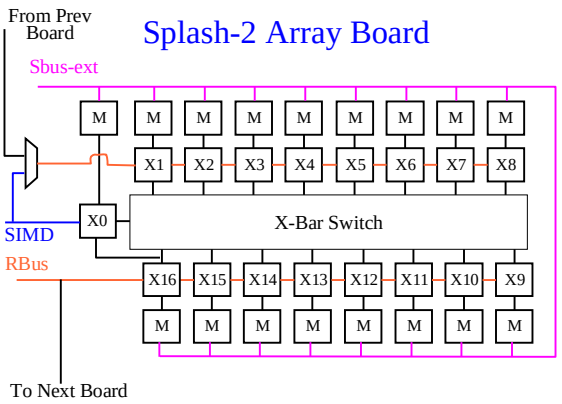
Splash-2 System



Splash-2 Components

- Host
- Interface card
- Array boards
  - 17 X4010s
  - 9 TI X-bar chips
  - 17 512K S-RAMs

Splash-2 Array Board



Splash-2 Area

Components	AREA
17 X4010	500 Mλ^2
17 256Kx16 RAM	2 Gλ^2
9 X-Bar	?
Total	43 Gλ^2

Misc.

- programmable clock
- 36 bit data path
- Load Programs using SBUS
  Must turn off card (so no dynamic reconfig)
- Keep Moore in mind: 17 X4010 = 3 X4062!

Splash-1

- Born to solve DNA string matching
- 33 Gλ^2
- 32 X3090s + 32 128K SRAMs
- Linear Interconnect only
- finish 1989
- Limited host transfer to 1MB/sec
  Compare to ~100MB/sec for S-Bus

Comparing Splash-1 & 2

	Splash-1	Splash-2
Year	1989	1992
FPGA	X3090	X4010
4-LUTs/Board	10240	13600
Max Input Rate	1 MB	100 MB
Interconnect	Linear Array	Linear
		Broadcast
		X-Bar
Memory/FPGA	128K	512K
Area	33 Gλ^2	43 Gλ^2

Programming Splash-2

- How Many Programs?
  - Host
  - Interface Board (XL & XR)
  - Array Board
    - X0
    - X1-X16
    - TI Chips

  FIVE PROGRAMS!
- Tag Bits
  - SIMD: instructions
  - Systolic: synchronization

Comparing Strings By Edit Distance

Motivation: The Human Genome Project. I.e., Do two genetic strings (E.g., DNA or Proteins) match? How are they related?

Abstraction: What is the cost of transforming *s* into *t* given a cost for
- insertion
- deletion
- substitution

Issues in Algorithm Design

- Alphabet
  letters in the string. For DNA, there are four: A, C, T, G
- Transformation Costs
  Insert:1, Delete:1, Substitute:2, match:0
- Type of comparison
  –one target to many sources
  –one target to one source

Example

		Cost
systolic		
↓	Delete s	1
syltolic		
↓	Insert m	1
symtolic		
↓	Substitute: t->b	2
symbolic		
	Total Cost:	4

Edit Distance by Dynamic Programming

Given source, $s_1s_2s_3\ldots s_n$, and target, $t_1t_2t_3\ldots t_m$, find $d_{n,m}$, Where $d_{i,j}$ is the edit distance between $s_1s_2s_3\ldots s_i$ and $t_1t_2t_3\ldots t_j$ defined as follows:

$$d_{0,0} = 0$$
$$d_{i,0} = d_{i-1,0} + \text{deleteCost}$$
$$d_{0,j} = d_{0,j-1} + \text{insertCost}$$
$$d_{i,j} = \min\begin{cases} d_{i-1,j} + \text{deleteCost} \\ d_{i,j-1} + \text{insertCost} \\ \begin{cases} 0 & s_i = t_j \\ d_{i-1,j-1} + \text{substCost} & s_i \neq t_j \end{cases} \end{cases}$$

DP Example

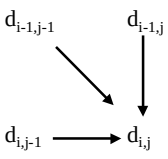
		s	y	m	b	o
	0	1	2	3	4	5
s	1					
y	2					
s	3					
t	4					
o	5					

DP Example

		s	y	m	b	o
	0	1	2	3	4	5
s	1	0	1	2	3	4
y	2	1	0	1	2	3
s	3	2	1	2	3	4
t	4	3	2	3	4	5
o	5	4	3	4	5	4

Parallelism on Anti-Diagonal

		s	y	m	b	o
	0	1	2	3	4	5
s	1	0	1	2	3	4
y	2	1	0	1	2	3
s	3	2	1	2	3	4
t	4	3	2	3	4	5
o	5	4	3	4	5	4



Bidirectional PE Example-1

S_{char}	o		t		s		y		s									
S_{in}	5		4		3		2		1									
S_{out}	5		4		3		2		1									
T_{char}										s		y		m		b		o
T_{in}										1		2		3		4		5
T_{out}										1		2		3		4		5
PE_{dist}	5	4	4	3	3	2	2	1	1	0	1	1	2	2	3	3	4	4	5

S_{char}		o		t		s		y		s								
S_{in}		5		4		3		2		1								
S_{out}		5		4		3		2		0								
T_{char}									s		y		m		b		o	
T_{in}									1		2		3		4		5	
T_{out}									0		2		3		4		5	
PE_{dist}		5	4	4	3	3	2	1	0	1	2	2	3	3	4	4	5	

Bidirectional PE Example-2

S_{char}		o		t		s		y		s								
S_{in}		5		4		3		2		1								
S_{out}		5		4		3		2		0								
T_{char}									s		y		m		b		o	
T_{in}									1		2		3		4		5	
T_{out}									0		2		3		4		5	
PE_{dist}		5	4	4	3	3	2	2	1	0	1	2	2	3	3	4	4	5	

S_{char}			o		t		s		y		s							
S_{in}			5		4		3		2		1							
S_{out}			5		4		3		1		0							
T_{char}								s		y		m		b		o		
T_{in}								0		1		3		4		5		
T_{out}								1		1		3		4		5		
PE_{dist}			5	4	4	3	3	2	1	0	1	2	3	3	4	4	5	

Bidirectional PE Example-3

S_{char}			o		t		s		y		s							
S_{in}			5		4		3		2		1							
S_{out}			5		4		3		1		0							
T_{char}								s		y		m		b		o		
T_{in}								0		1		3		4		5		
T_{out}								1		1		3		4		5		
PE_{dist}			5	4	4	3	3	2	1	0	1	2	3	3	4	4	5		

S_{char}				o		t		s		y		s						
S_{in}				5		4		3		2		1						
S_{out}				5		4		3		2		0						
T_{char}							s		y		s							
T_{in}							2		1		2							
T_{out}							2		0		2							
PE_{dist}				5	4	4	3	2	1	0	1	2	3	4	4	5		

Bidirectional PE Example-4

S_{char}				o		t		s		y		s						
S_{in}				5		4		3		1		1						
S_{out}				5		4		2		0		2						
T_{char}							s		y		m		b		o			
T_{in}							2		0		2		4		5			
T_{out}							1		0		2		4		5			
PE_{dist}				5	4	4	3	2	1	0	1	2	3	4	4	5		

S_{char}				o		t		s		y		s						
S_{in}				5		3		2		0		2						
S_{out}				5		3		1		1		3						
T_{char}							s		y		m		b		o			
T_{in}							1		1		2		4		5			
T_{out}							1		1		2		4		5			
PE_{dist}				5	4	3	2	2	0	1	2	3	4	5				

Bidirectional PE Example-5

S_{char}					o		t		s		y		s					
S_{in}					5		3		2		0		2					
S_{out}					5		3		1		1		3					
T_{char}								s		y		m		b		o		
T_{in}								1		1		2		4		5		
T_{out}								1		1		2		4		5		
PE_{dist}					5	4	3	2	1	0	1	2	3	4	5			

S_{char}						o		t		s		y		s				
S_{in}						5		3		1		2		3				
S_{out}						4		2		2		1		4				
T_{char}							s		y		m		o					
T_{in}							3		2		2		5					
T_{out}							4		2		2		4					
PE_{dist}						5	3	2	1	2	1	2	3	4				

Bidirectional Summary

- 16 CLBs/PE
- 384 PEs/Board
- 2,100 Million Cells/sec
- Requires 2*(m+n) PEs
- Uses only half the processors at any one time.
- Must stream both source and target for each comparison. Makes comparison against large DB impractical.

Unidirectional Approach

- Store source in array, then stream target past it.
- Can handle many targets for each source.
- Can overlap initialization, computation, and finish.
- Uses only *n* PEs

Unidirectional Example 1

Time=1

Tag	S
Cin	S
Din	?
PEDin	1
Dout	1
PEDout	1
Src	S

Time=2

Tag	S	S
Cin	Y	∅
Din	?	1
PEDin	2	1
Dout	1	1
PEDout	2	1
Src	S	∅

Time=3

Tag	S	S	S
Cin	M	Y	∅
Din	?	1	1
PEDin	3	2	1
Dout	1	2	1
PEDout	3	2	1
Src	S	Y	∅

Time=4

Tag	S	S	S
Cin	B	M	∅
Din	?	1	2
PEDin	4	3	2
Dout	1	2	2
PEDout	4	3	2
Src	S	Y	∅

Unidirectional Example 2

Time=4

Tag	S	S	S	S
Cin	B	M	∅	∅
Din	?	1	2	1
PEDin	4	3	2	1
Dout	1	2	2	1
PEDout	4	3	2	1
Src	S	Y	∅	∅

Time=5

Tag	S	S	S	S	S
Cin	O	B	∅	∅	∅
Din	?	1	2	2	1
PEDin	5	4	3	2	1
Dout	1	2	3	2	1
PEDout	5	4	3	2	1
Src	S	Y	M	∅	∅

Time=6

Tag	P	S	S	S	S	S
Cin	?	O	B	∅	∅	∅
Din	0	1	2	3	2	1
PEDin	?	5	4	3	2	1
Dout	1	2	3	3	2	1
PEDout	0	5	4	3	2	1
Src	S	Y	M	∅	∅	∅

Time=7

Tag	T	P	S	S	S	S
Cin	S	?	O	∅	∅	∅
Din	1	1	2	3	3	2
PEDin	?	0	5	4	3	2
Dout	0	2	1	3	2	1
PEDout	1	1	5	3	3	2
Src	S	Y	M	B	∅	∅

Unidirectional Example 3

Time=7

Tag	T	P	S	S	S	S
Cin	S	?	O	∅	∅	∅
Din	1	1	2	3	3	2
PEDin	?	0	5	4	3	2
Dout	0	2	1	3	2	1
PEDout	1	1	5	3	3	2
Src	S	Y	M	B	∅	∅

Time=8

Tag	T	T	P	S	S	S
Cin	Y	S	?	O	∅	∅
Din	2	0	2	1	3	2
PEDin	?	1	1	5	3	2
Dout	1	1	3	3	3	2
PEDout	2	0	2	5	3	2
Src	S	Y	M	B	∅	∅

Time=9

Tag	T	T	T	P	S	S
Cin	S	Y	S	?	∅	∅
Din	3	1	1	3	3	3
PEDin	?	2	0	2	5	3
Dout	2	0	2	3	5	3
PEDout	3	1	1	3	5	3
Src	S	Y	M	B	O	∅

Time=10

Tag	T	T	T	T	P	S
Cin	T	S	Y	S	?	∅
Din	4	2	0	2	3	5
PEDin	?	3	1	1	3	5
Dout	3	1	1	3	5	5
PEDout	4	2	0	2	3	5
Src	S	Y	M	B	O	∅

Calculating Time=9 to 10

Time=9

Tag	T	T	T	P	S	S
Cin	S	Y	S	?	O	∅
Din	3	1	1	3	3	3
PEDin	?	2	0	2	5	3
Dout	2	0	2	3	5	3
PEDout	3	1	1	3	5	3
Src	S	Y	M	B	O	∅

Time=10

Tag	T	T	T	T	P	S
Cin	T	S	Y	S	?	∅
Din	4	2	0	2	3	5
PEDin	?	3	1	1	3	5
Dout	3	1	1	3	5	5
PEDout	4	2	0	2	3	5
Src	S	Y	M	B	O	∅

1 = Min(0+1, 2+1, 1+2)

Compare Processors to table

Time=10

Tag	T	T	T	T	P	S
Cin	T	S	Y	S	?	∅
Din	4	2	0	2	3	5
PEDin	?	3	1	1	3	5
Dout	3	1	1	3	5	5
PEDout	4	2	0	2	3	5
Src	S	Y	M	B	O	∅

		s	y	m	b	o
	0	1	2	3	4	5
s	1	0	1	2	3	4
y	2	1	0	1	2	3
s	3	2	1	2	3	4
t	4	3	2	3	4	5
o	5	4	3	4	5	4

Unidirectional Example 4

Time=10

Tag	T	T	T	T	P	S
Cin	T	S	Y	S	?	∅
Din	4	2	0	2	3	5
PEDin	?	3	1	1	3	5
Dout	3	1	1	3	5	5
PEDout	4	2	0	2	3	5
Src	S	Y	M	B	O	∅

Time=12

Tag	P	T	T	T	T	T
Cin	?	O	T	S	Y	S
Din	0	4	2	2	2	4
PEDin	?	5	3	1	1	3
Dout	4	3	3	3	3	4
PEDout	0	4	2	2	2	4
Src	S	Y	M	B	O	∅

Time=11

Tag	T	T	T	T	T	P
Cin	O	T	S	Y	S	∅
Din	5	3	1	1	3	5
PEDin	?	4	2	0	2	3
Dout	4	2	2	2	4	3
PEDout	5	3	1	1	3	3
Src	S	Y	M	B	O	∅

Time=13

Tag			T	T	T	T
Cin			O	T	S	Y
Din			3	3	3	3
PEDin			4	2	2	2
Dout			4	4	4	3
PEDout			3	3	3	3
Src			M	B	O	∅

Do we need an additional tag?

Time=12

Tag	P	T	T	T	T	T
Cin	?	O	T	S	Y	S
Din	0	4	2	2	2	4
PEDin	?	5	3	1	1	3
Dout	4	3	3	3	3	4
PEDout	0	4	2	2	2	3
Src	S	Y	M	B	O	∅

How is Dout reset unless a fourth tag is used?

Unidirectional Example 5

Time=13

Tag			T	T	T	T
Cin			O	T	S	Y
Din			3	3	3	3
PEDin			4	2	2	2
Dout			4	4	4	3
PEDout			3	3	3	3
Src			M	B	O	∅

Time=15

Tag					T	T
Cin					O	T
Din					5	5
PEDin					4	4
Dout					4	5
PEDout					4	5
Src					O	∅

Time=14

Tag				T	T	T
Cin				O	T	S
Din				4	4	4
PEDin				3	3	3
Dout				5	5	4
PEDout				4	4	4
Src				B	O	∅

Final result (finally☺)

Unidirectional Summary

- 24 CLBs/PE
- 248 PEs/Board
- 3,000 Million Cells/sec
- Requires only n PEs
- Could use constant propagation to reduce size and increase speed - in this case the advantage is marginal

Reducing the Datapath

- How big must the comparators, registers, adders, etc be?
- What is maximum difference between any two table entries?
- How can one utilize this to reduce the datapath?

Performance Comparison

Machine	Million CUPS (Cell updates/sec)	CUPS/λ^2
Splash 2 (unidir)	3,000	0.38
Splash 2 (bidir)	1,050	0.13
Splash 1 (bidir)	370	0.028
P-NAC (34)	500	1.9
CM-2 (64K)	150	
CM-5 (32)	33	<0.0007
Sparc 10	1.2	0.00075
Sparc 1	0.87	0.0032

The Perfect Application

- High efficiency: O(n) parallelism
- local communication
- small datapath
- simple data controller