

Program In – Chip Out

CSE 291E / EE260C

Spring 2002

University of California

San Diego

The logo for the University of California San Diego, featuring a stylized gray graphic of a building or landscape behind the text.

Overview

- Quick review of basic architectures
 - What is Single Issue, Super Scalar, VLIW,
- Overview of Systolic Arrays
- Overview of PICO Project
- DataWidth Reduction Algorithm

Architecture Review

- Code Segment

```
For (n=0; n<100; n++)  
{  
    A[n+1] = A[n]*x[n];  
    B[n+1] = B[n]*y[n] + A[n];  
    C[n+1] = C[n]*z[n] + B[n];  
}
```

- How does this map on different architectures?
 - In-order Single Issue
 - Superscalar
 - VLIW

In-Order Single Issue

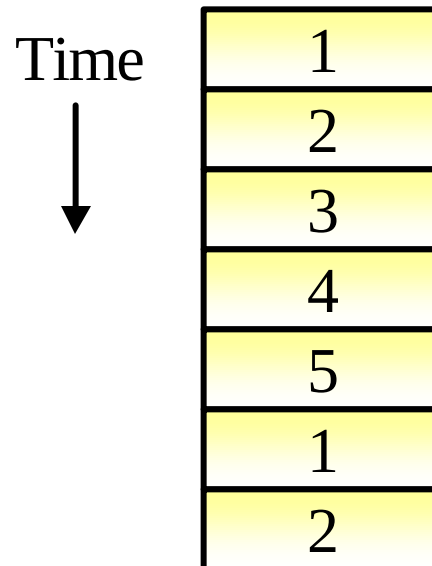
1) $A[n+1] = A[n] * x[n]$

2) $r1 = B[n] * y[n]$

3) $B[n+1] = r1 + A[n]$

4) $r2 = C[n] * z[n]$

5) $C[n+1] = r2 + B[n]$



Superscalar

1) $A[n+1] = A[n] * x[n]$

2) $r1 = B[n] * y[n]$

3) $B[n+1] = r1 + A[n]$

4) $r2 = C[n] * z[n]$

5) $C[n+1] = r2 + B[n]$

Time



1	2
3	4
5	1
2	
3	4
5	1
2	
3	4

VLIW

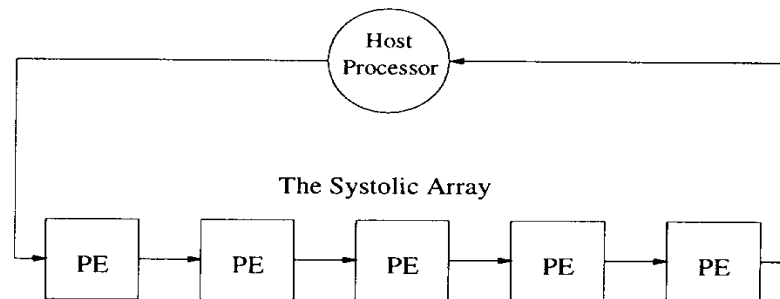
1:2) $A[n+1] = A[n] * x[n] : r1$ $= B[n] * y[n]$
3:4) $B[n+1] = r1 + A[n] : r2$ $= C[n] * z[n]$
5) $C[n+1] = r2 + B[n] : NOP$

Time
↓

1 : 2
3 : 4
5 : NOP
1 : 2
3 : 4
5 : NOP

Systolic Arrays

- Where does name “Systolic Array” come from?
 - Array: to set or place in order
 - Systolic: a rhythmically recurrent contraction; *especially* the contraction of the heart by which the blood is forced onward and the circulation kept up
- What is a Systolic Array?
 - A network of PEs that rhythmically compute and pass data through the system

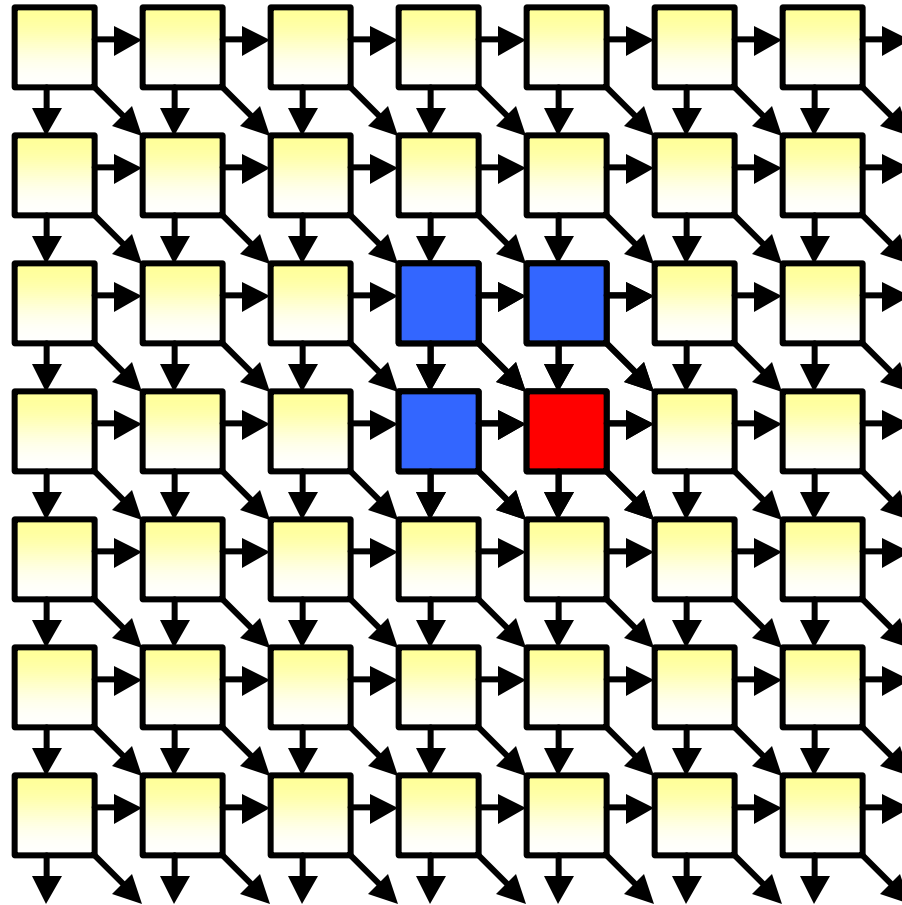


Systolic Arrays

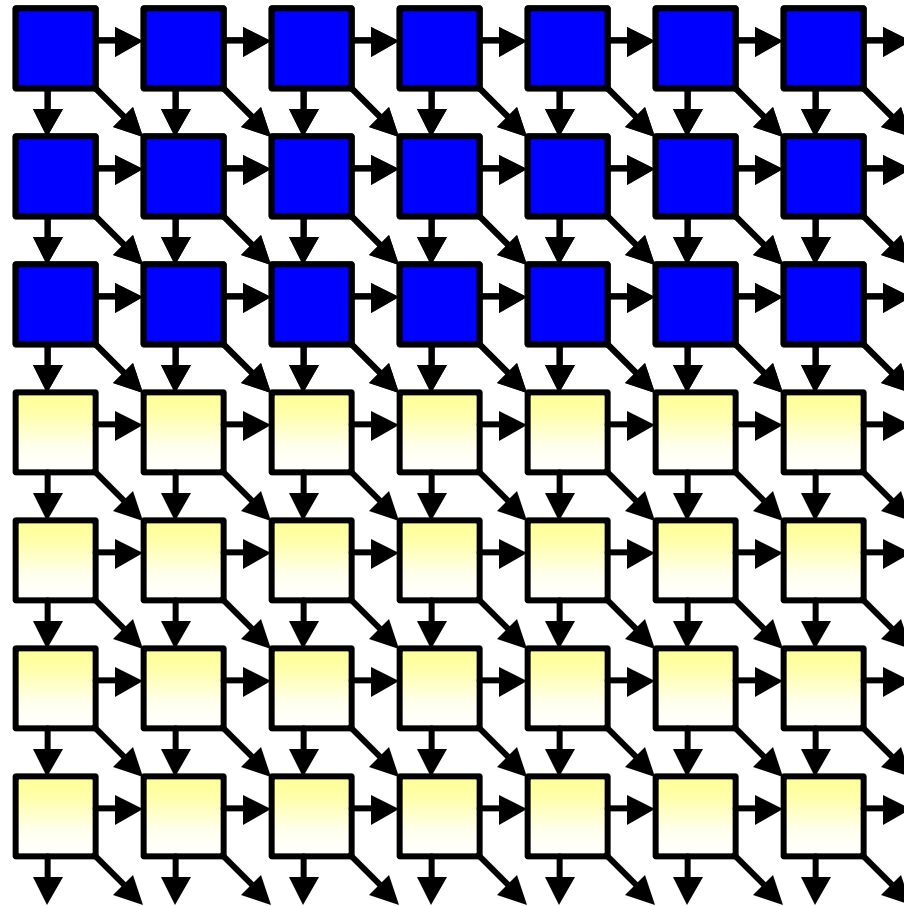
- All PEs are uniform and fully pipelined (usually)
- Only local interconnection (nearest neighbor)
- Some relaxations are introduction to increase the utility of systolic arrays
 - *Neighbor interconnection (near, but not nearest)*
 - *Data broadcast operations*
 - *Different PEs, especially at the boundaries*

Data Graphs for Systolic Arrays

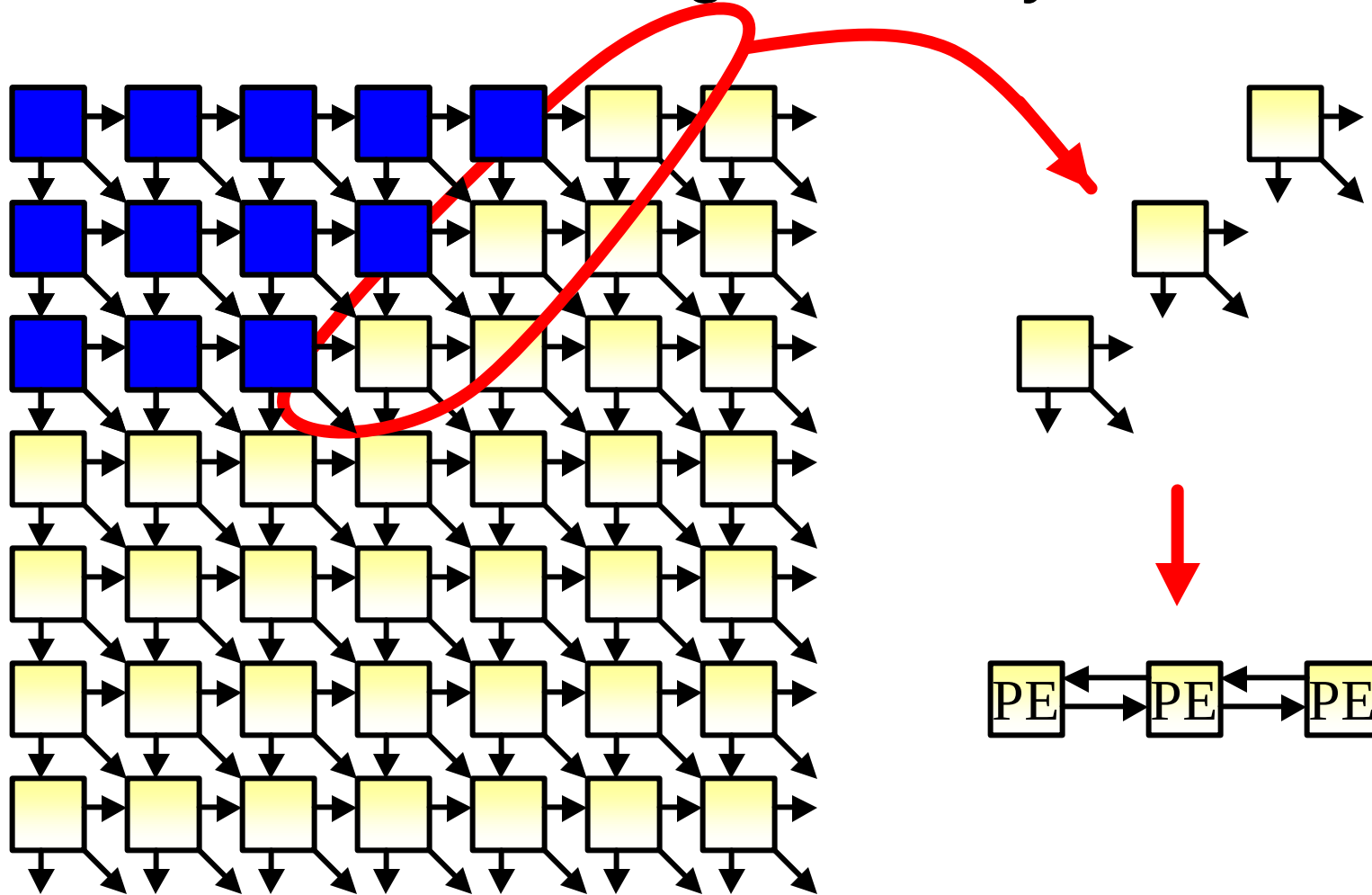
- Example: dynamic programming



Walking the Data Graph



Building the Array

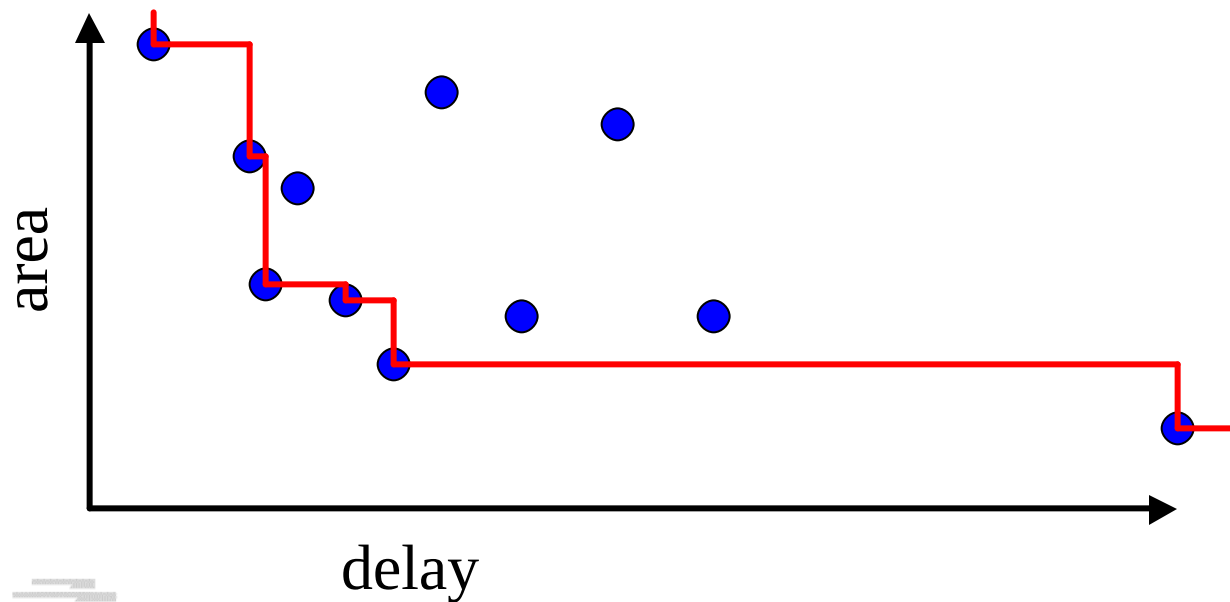


PICO

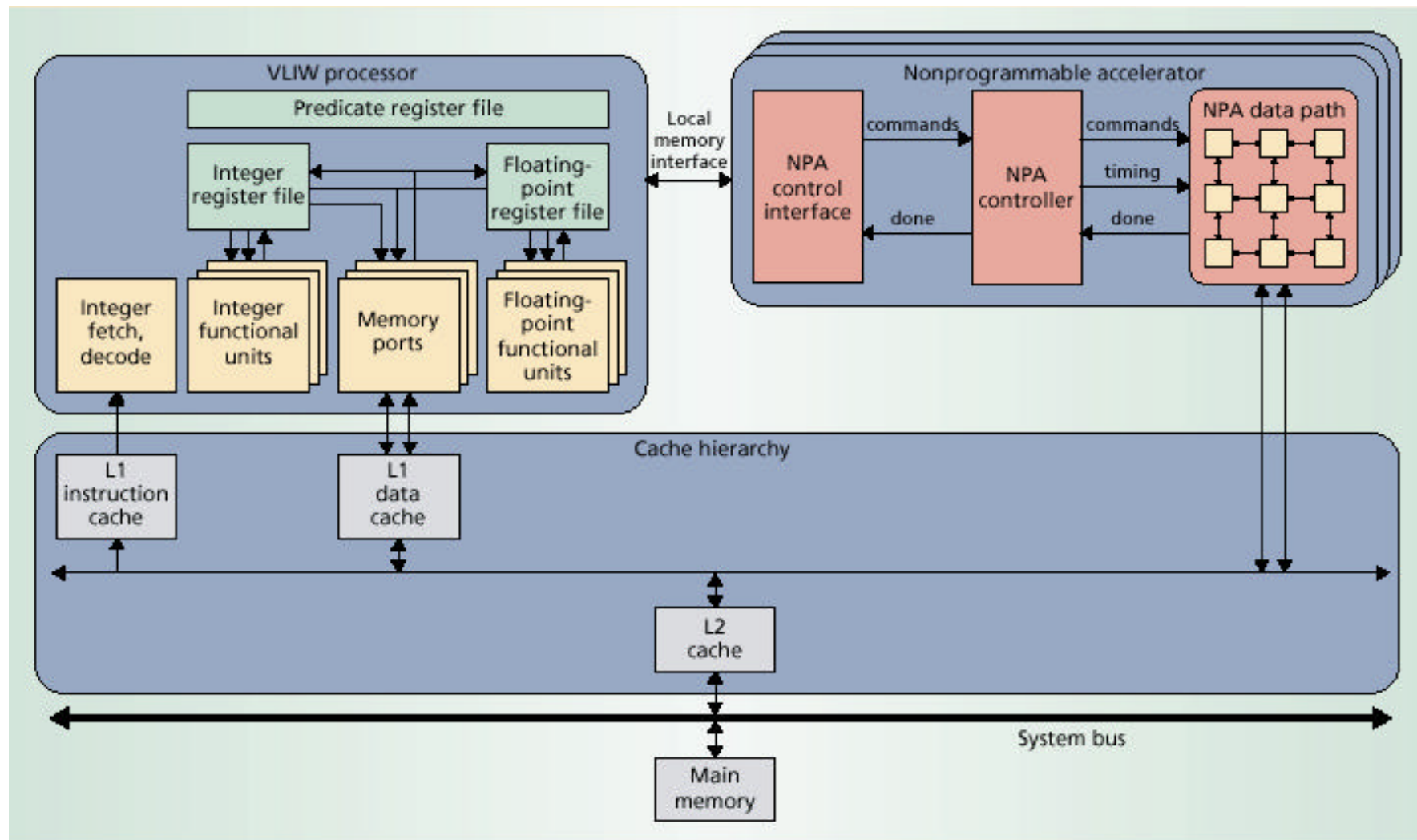
- Program In Chip Out (PICO)
 - Architecture synthesis system from HP
 - Work done by Bob Rau's group
 - Input: Application written in subset of C
 - No complex pointer
 - No wacky array indexing
 - Metric: Chip area and performance
 - Output: H/W as VHDL & S/W as binary
 - Generates Pareto-optimal architecture

Pareto Optimality

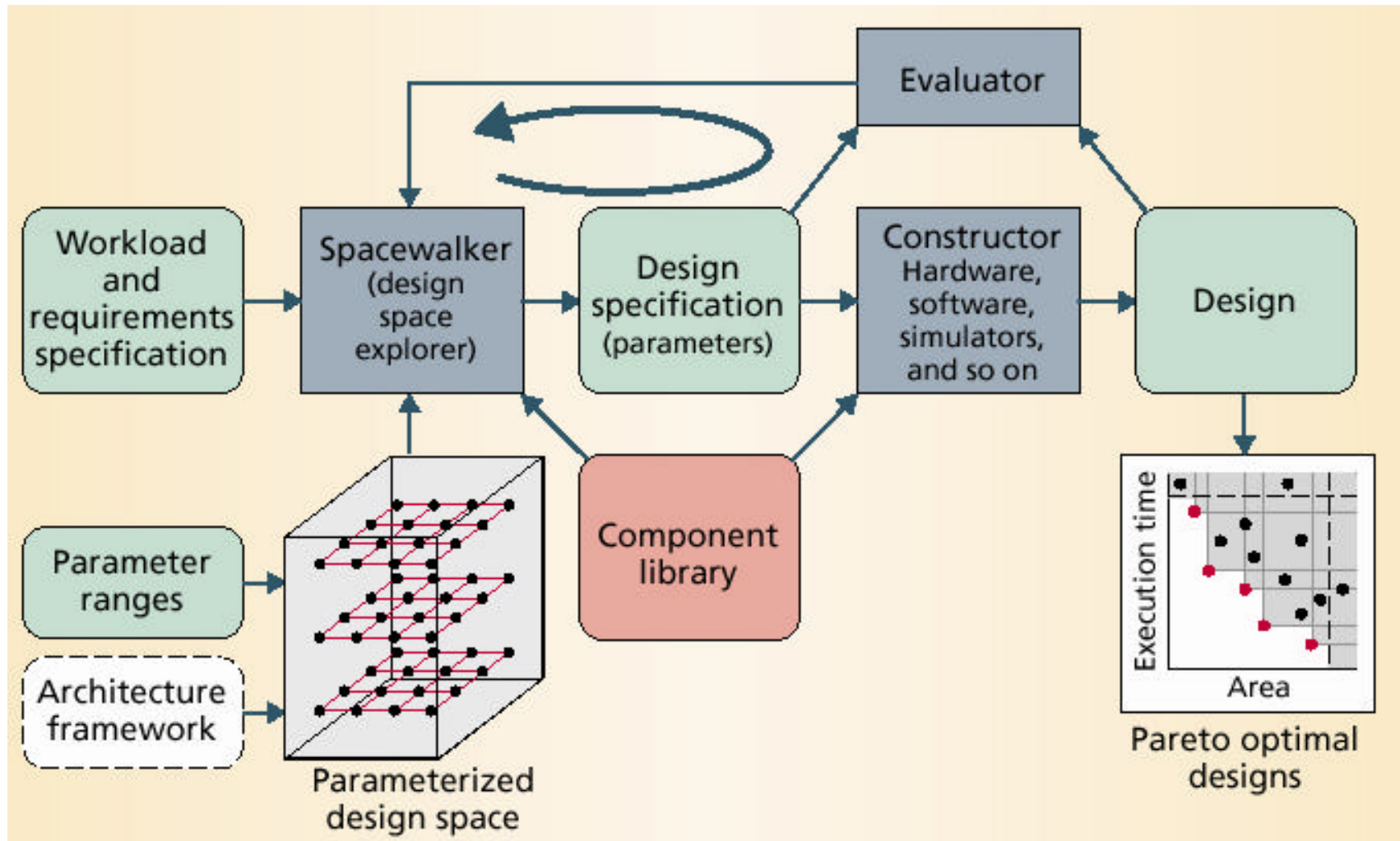
- For a set of design points, a given design is pareto optimal if:
 - No other design is better with respect to every evaluation metric
 - This means there can be multiple pareto optimal points



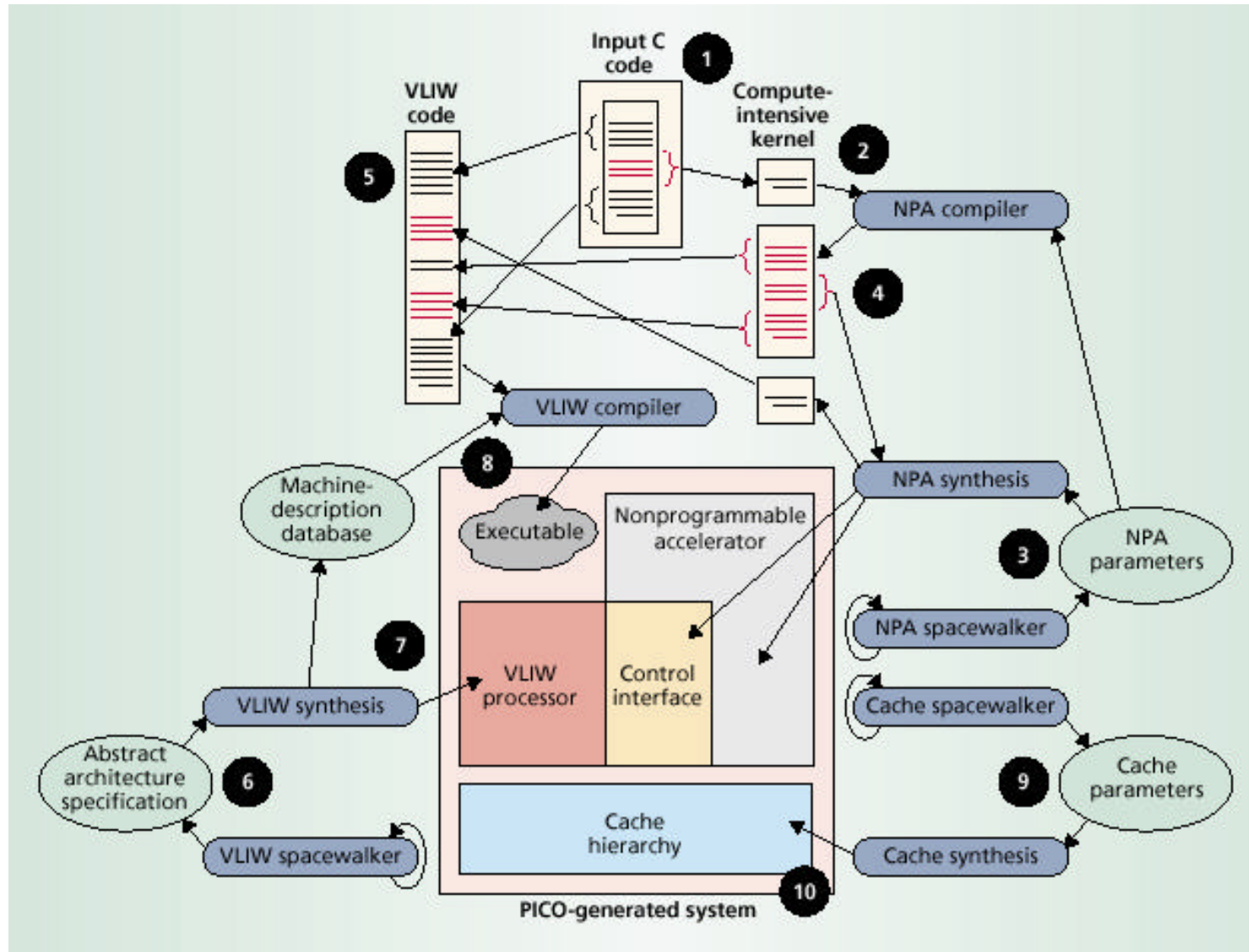
PICO Architecture



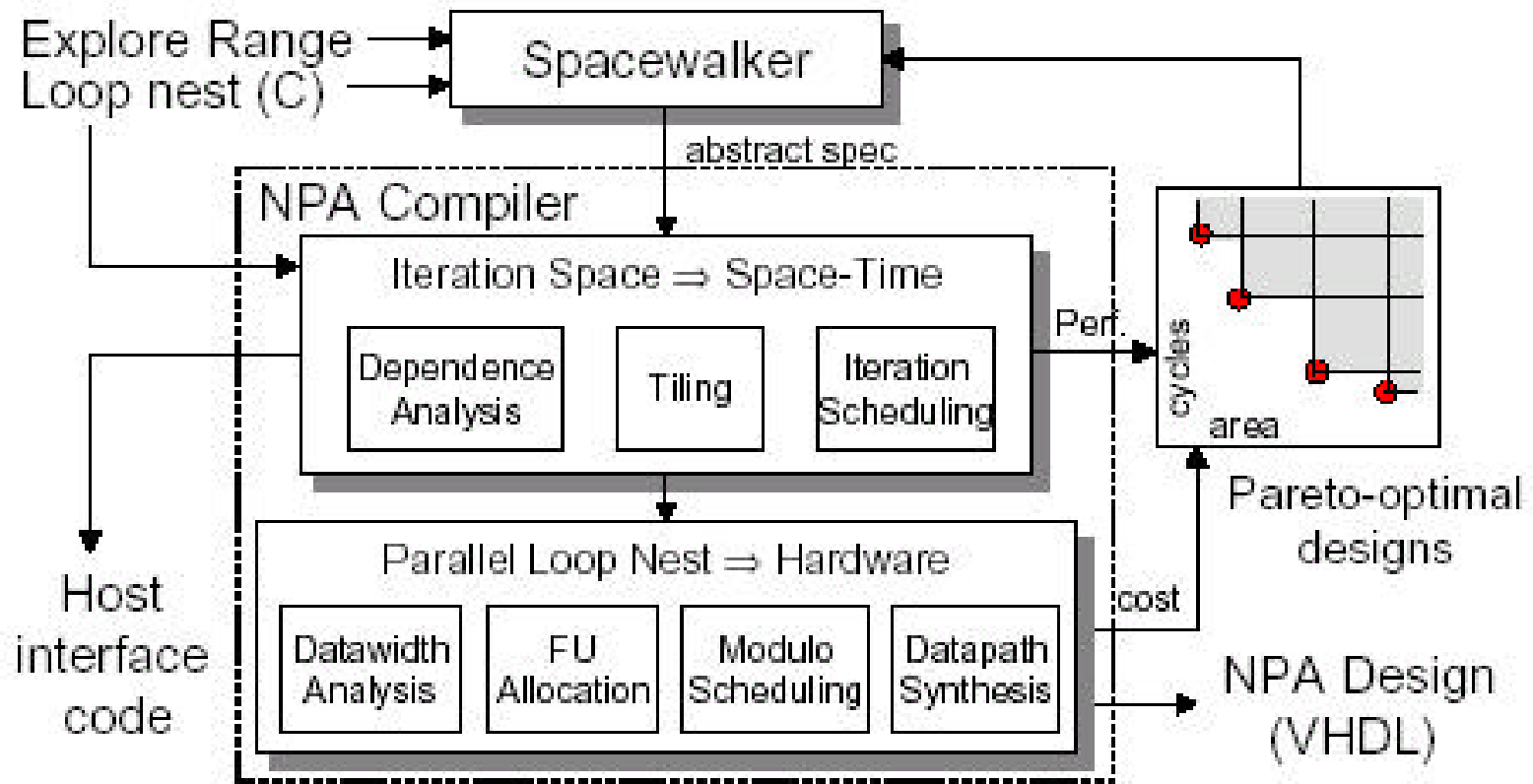
PICO Design Framework



PICO Design Flow



PICO NPA Design



PICO Analysis

L1: $x = a + 1$

L2: $y = x * b$

Loop:

L3: $y = y + 1$

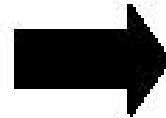
If () goto loop

L4: $z = y + c$

PICO Datewidth Analysis

l1: $x = a + 1$ 32 3 2
l2: $y = x * b$ 32 32 11
Loop: l3: $y = y + 1$ 32 32 2 if () goto Loop
l4: $z = y + c$ 16 32 32

original code



l1: $x = a + 1$ 4 3 2
l2: $y = x * b$ 15 4 11
Loop: l3: $y = y + 1$ 32 32 2 if () goto Loop
l4: $z = y + c$ 16 32 32

after forward prop



l1: $x = a + 1$ 4 3 2
l2: $y = x * b$ 15 4 11
Loop: l3: $y = y + 1$ 16 16 2 if () goto Loop
l4: $z = y + c$ 16 16 16

after backward prop