

# Requirements for any FPGA/HPC Application Development Tool Flow

... if you want any reasonable fraction of  
the FPGA's potential performance



*Tom VanCourt*

*Martin C. Herbordt*

*Computer Architecture and Automated Design Lab*

<http://www.bu.edu/caadlab>

# What is FPGA/HPC exactly?

- High performance computing

*Computational chem.      Electromagnetics*  
*Bioinformatics              Traffic modeling*  
*Astrophysics                ...*

- Field Programmable Gate Arrays

*App. specific processors on demand*  
*Massive fine-grained parallelism*  
*Drivers of silicon process development*

# What's so hard about it?

- Performance computing  $\neq$  logic design

*Standard languages hide parallelism\**

*FPGA tools address logic designers*

\*Jeroen Voeten,  
ACM Trans. CAD  
6(4)533-552,  
Oct 2001

- Contradictions in FPGA applications

*Applications should be widely applicable*

*... but finely tuned to each particular usage*

*Require customization by application specialist*

*... but require unfamiliar hardware constructs*

*Demand full use of hardware resources*

*... use is app-specific, resources are FPGA-specific*

# What's wrong with C to gates?

*“Unfortunately, and despite 40 years of parallelizing compilers for all sorts of machines, [optimization] algorithms don't work terribly well.” Ian Page, 2004*

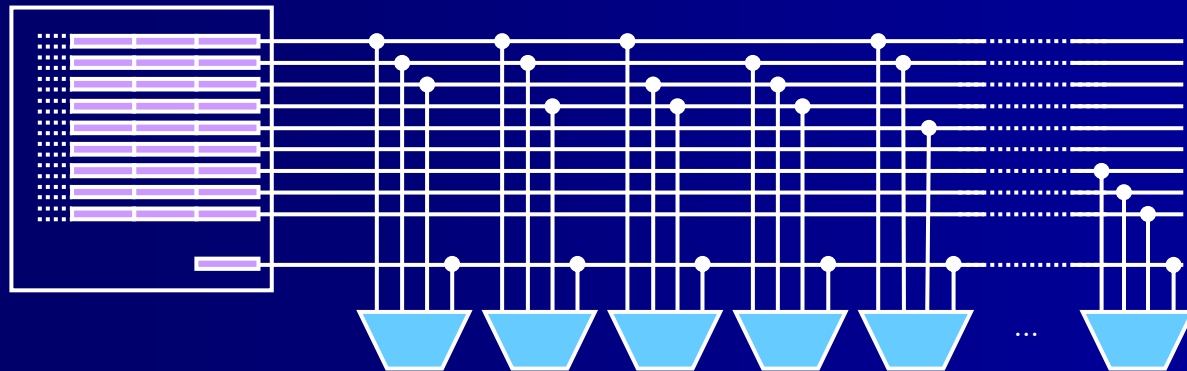
- The best you get is C code in gates  
*Good HW algorithm isn't SW algorithm*
- C distributes algorithms in time  
*FPGAs distribute algorithms in space  
... and a whole industry is dedicated to  
reinventing the von Neumann bottleneck*

# Example: Size-3 subsets

- C style:

```
for i = 0 to N
  for j = 0 to i
    for k = 0 to j
      // use x[i],x[j],x[k]
```

- HW-oriented solution:



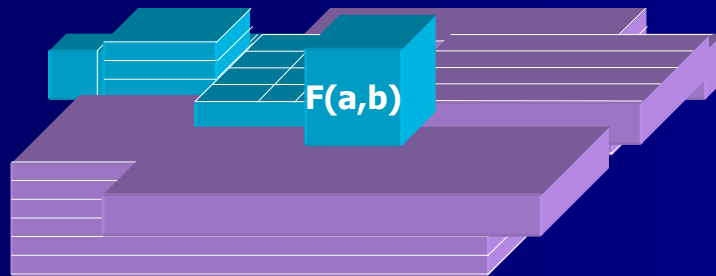
# Example: 3D Correlation

- Serial processor: Fourier transform  $\mathcal{F}$

$$A \otimes B = \mathcal{F}^{-1} ( \mathcal{F}(A) \times \mathcal{F}(B) )$$

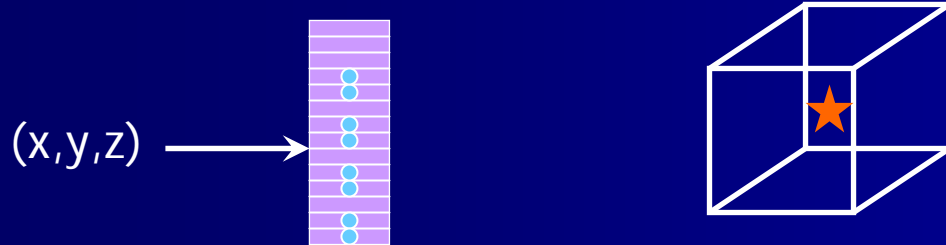
- FPGA: Direct summation

*RAM FIFO*

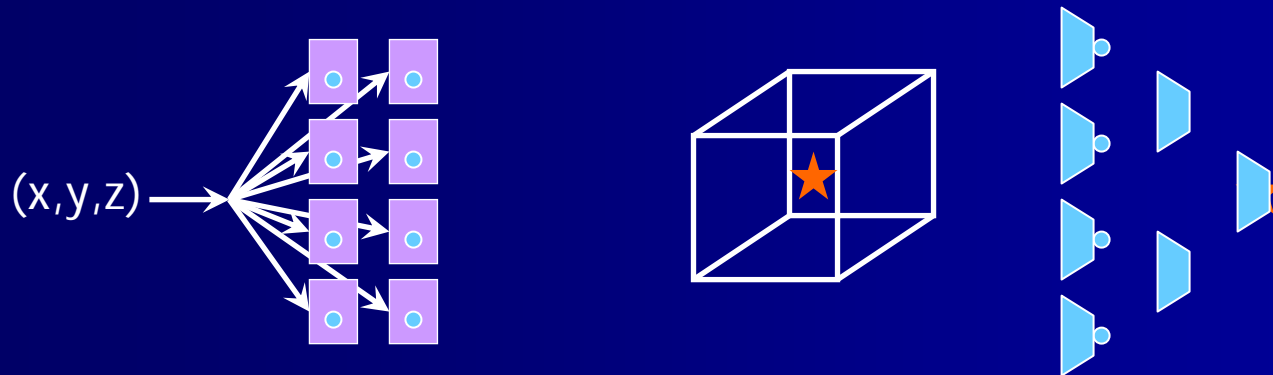


# Example: Trilinear Interpolation

- C style: Sequential RAM access



- HW style: App-specific interleaving



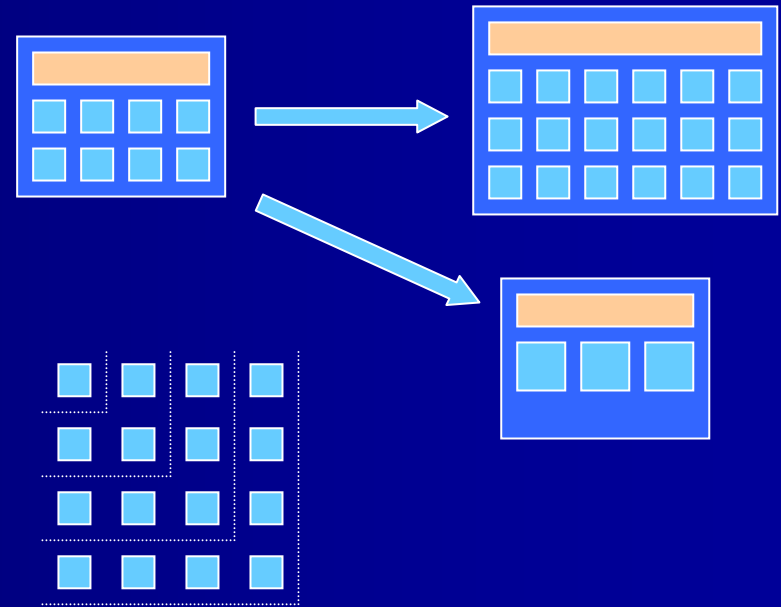
# Sizing applications to FPGAs

- Desired size of computing array:  
*As big as possible* – *whatever that means*

- Depends on:  
*FPGA capacity*

*Application details*

*Computing array*





# C Coding Style vs. Performance

- Hardware algorithms are different
  - Require non-SW algorithms*
  - Require non-von Neumann memory*
  - Require non-obvious data paths*
  - Require careful precision analysis*
- Explicit degree of parallelism is a bug
  - No commercial tools address all factors*
- App. specialists aren't logic designers
  - Need both – efficient HW & app. details*

# The Requirements

- Escape from the C code model
  - GPUs? Device organizes control & memory*
  - ... application is leaf calculations only*
- Support two developer groups:
  - Logic designers create efficient structures*
  - App specialists tailor it to specific usage*
- Full use of FPGA's computing resources
  - App-specific, FPGA-specific array sizes*