Yesterday, Today & Tomorrow: a view of progress in computer design

Michael Flynn

Stanford Univ. and Maxeler

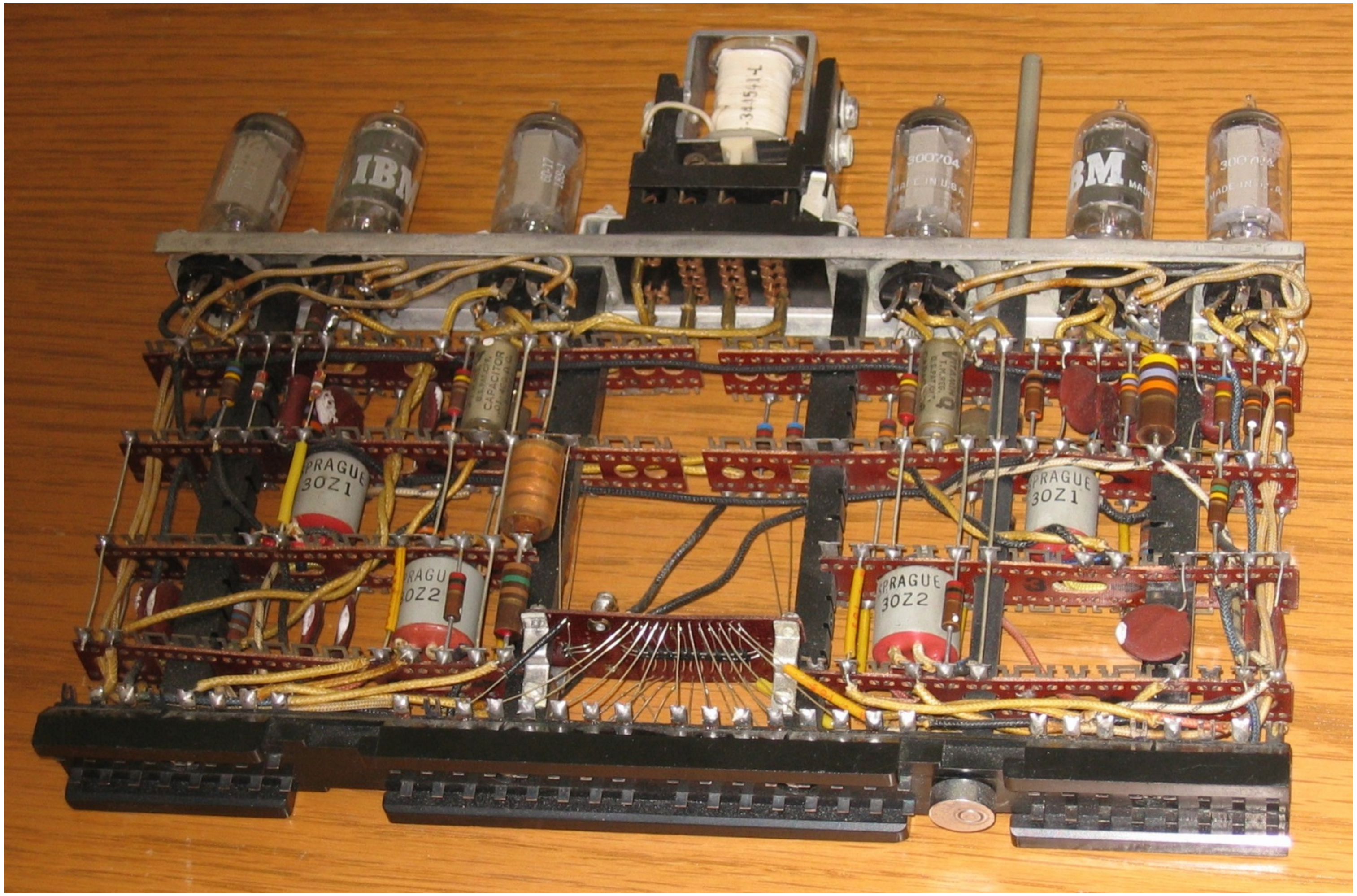
Outline

- Yesterday: a decisive moment in computer design ’55-56
- Today: creating a similar moment in parallel processing and programming

Yesterday: The Automation of design and manufacturing

- A move to minimize the human element in both design (draftsman, etc) and production, especially enabled by transistor logic with automatic component insertion in printed circuit cards.
- The dawn of design automation

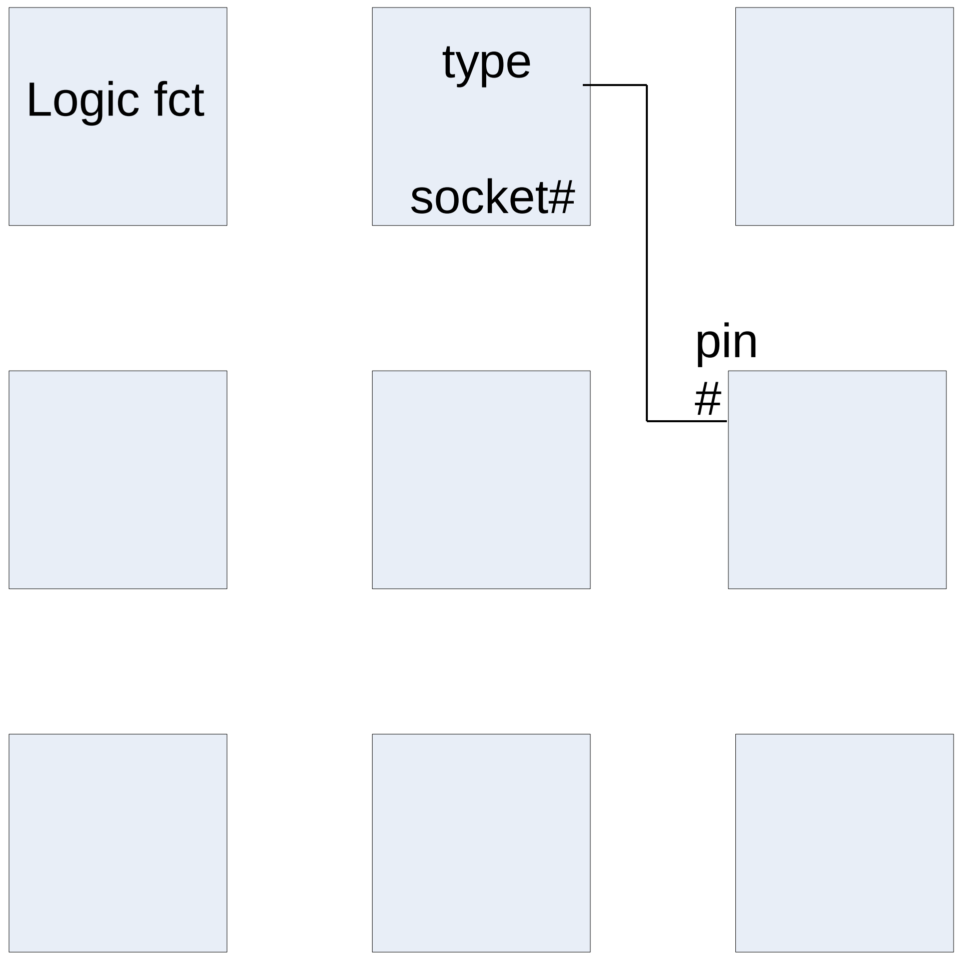
The old way


IBM
CAPACITOR
SPRAGUE 30Z1
SPRAGUE 30Z2
SPRAGUE 30Z1
SPRAGUE 30Z2
MADE IN U.S.A.

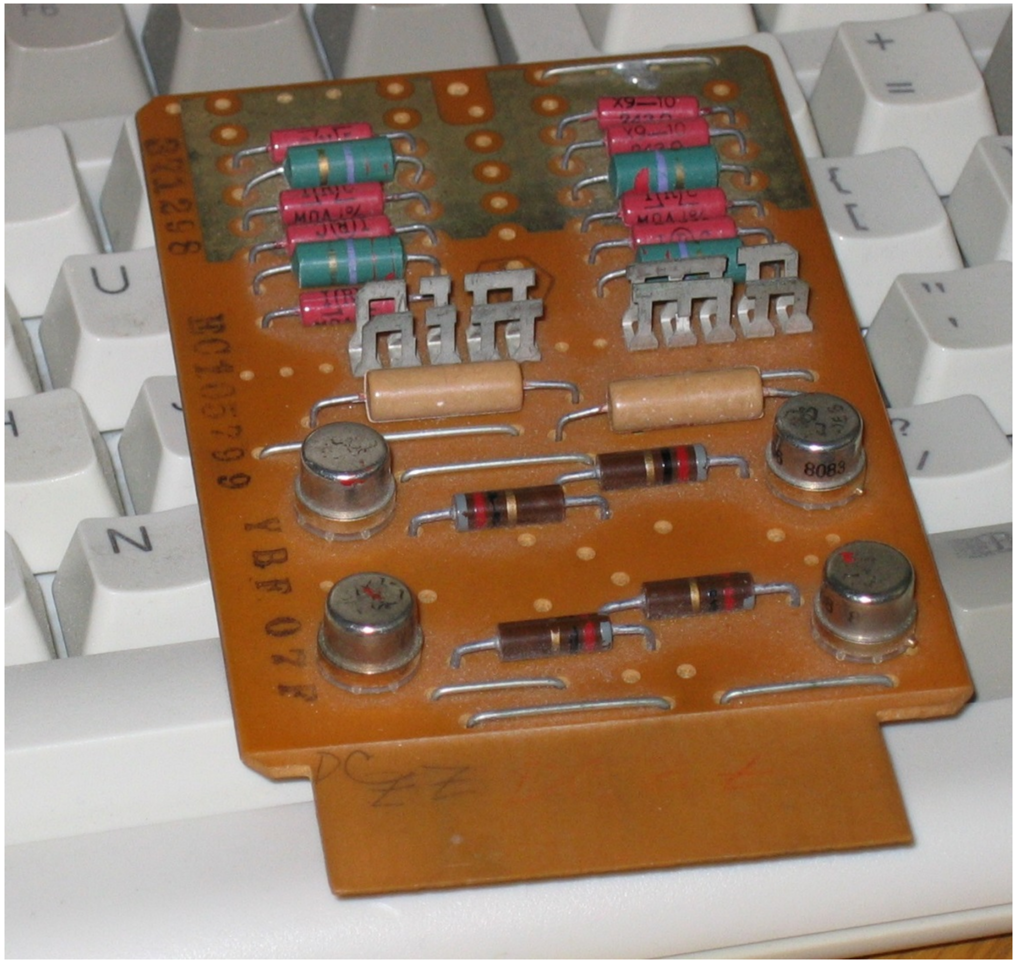
The SMS system: the end of hand crafted design

- Developed in ’56, first production ’58, end of new designs in ’63 (with intro. of SLT)
- Standard design system (ALDs)
- Standard printed cards and circuit families
- Automatic wire wrapped back panel
- Standard frames (large version, had 4 6’ pages, 4 panels/page, 240 cards/panel)
- Spanned small (1401) to very large (7030)

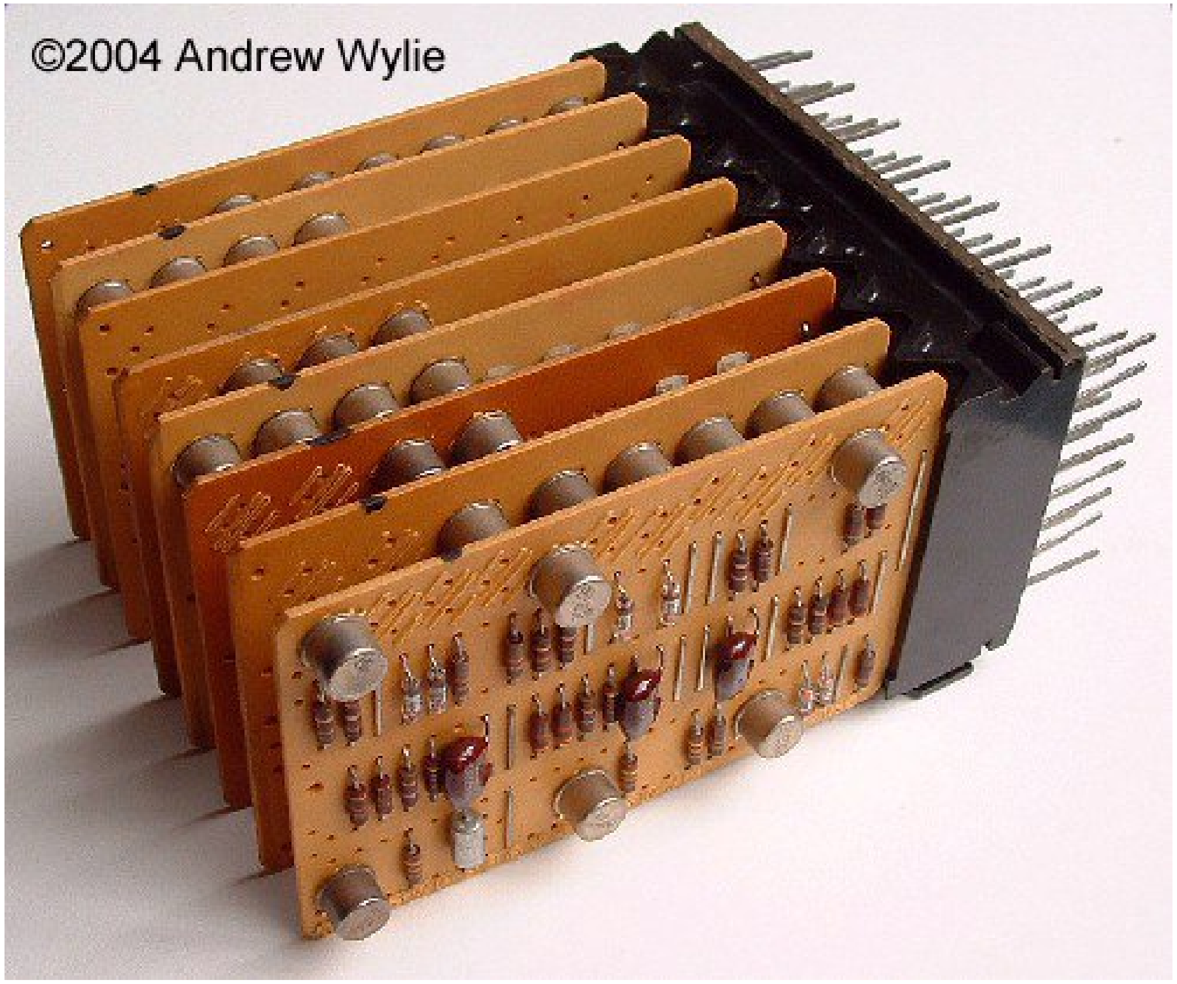
ALD: Automated Logic Diagram



SMS card



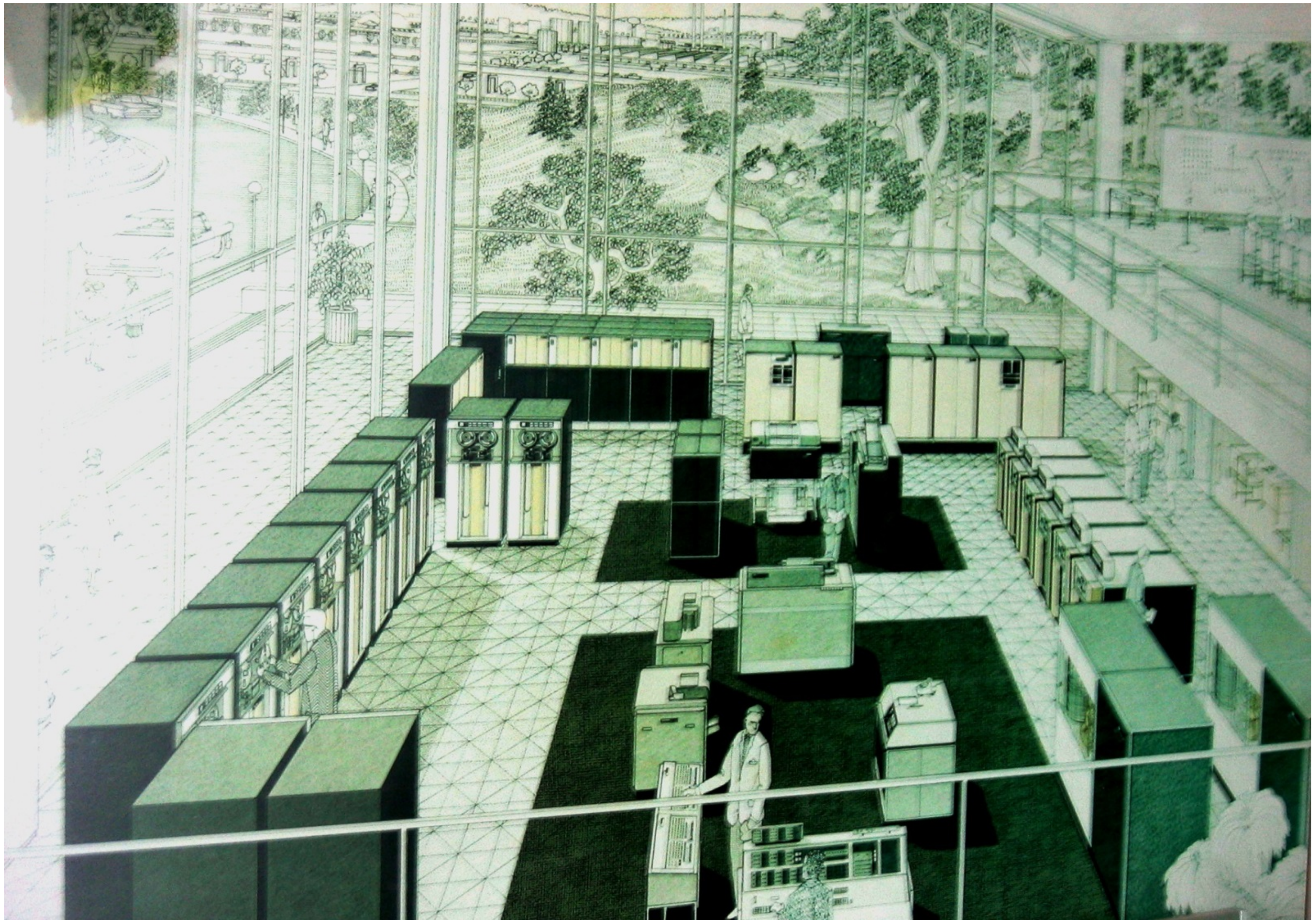
The SMS cards and sockets



SMS DA system

- Assigned card type to logic, assigned card to socket and designated pins for wire connection (upon design completion do a wire wrap routing schedule)
- Limited logic simulation /checking
- Print out updated ALD
- When design completed, provide card deck for automated back panel wiring, wiring a panel on a Gardener Denver took less than an hour.
- Bed of nails test to validate wiring, again with card deck.

7090 (ca. 1961)



Today: automating programming for maximum speedup

- Emulate the production line in program via static dataflow.
- Prefer most restrictive parallel models, then less restrictive.
- Support parallelism: multicore for control flow, dataflow for compute intensive kernels
- Use 2D spatial programming to implement the dataflow

Slotnick's law

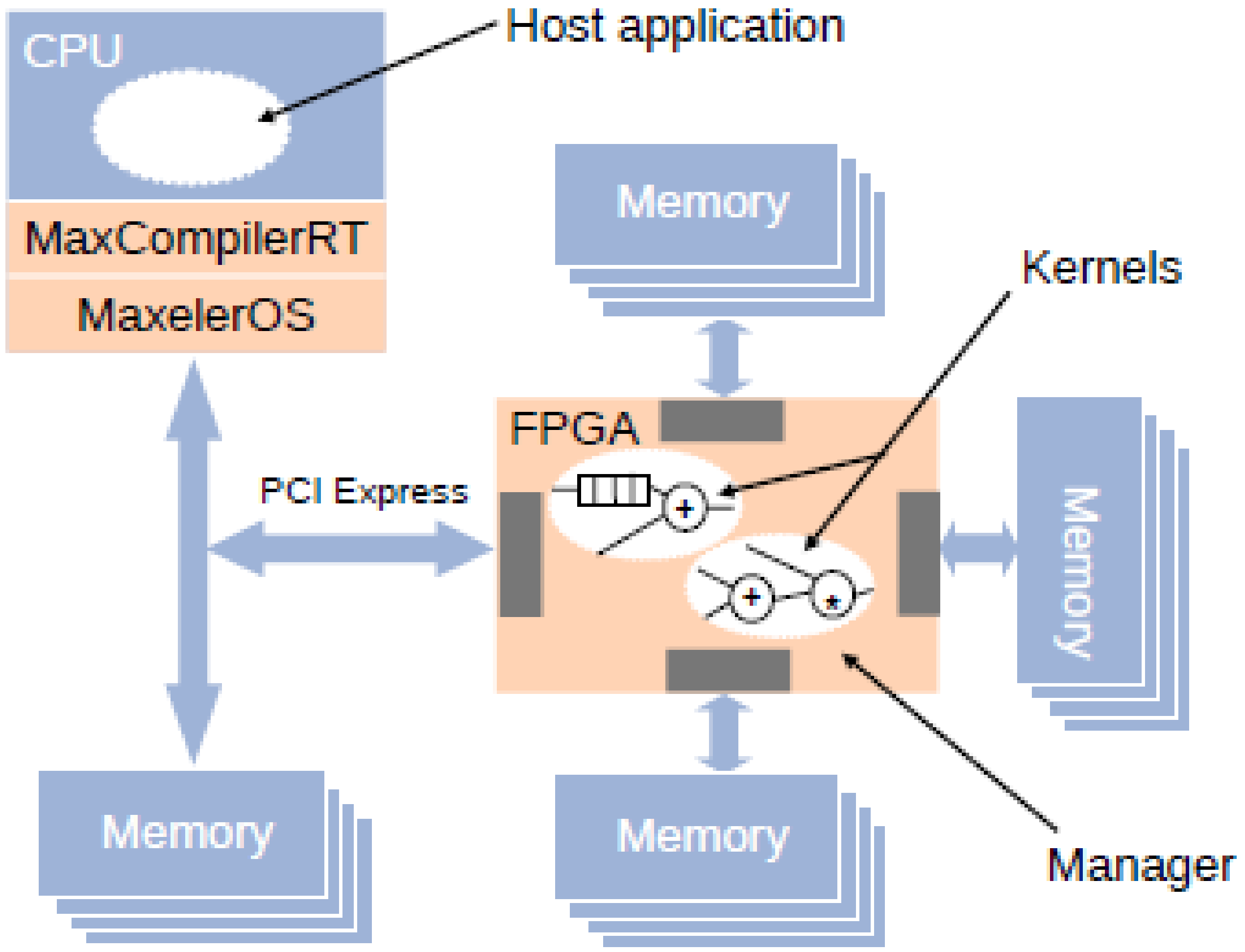
"The parallel approach to computing does require that some original thinking be done about numerical analysis and data management in order to secure efficient use.

In an environment which has represented the absence of the need to think as the highest virtue this is a decided disadvantage."

-Daniel Slotnick (1967)

….***Speedup in parallel processing is achieved by programming effort***……

FPGA emulating dataflow: using a server with acceleration cards



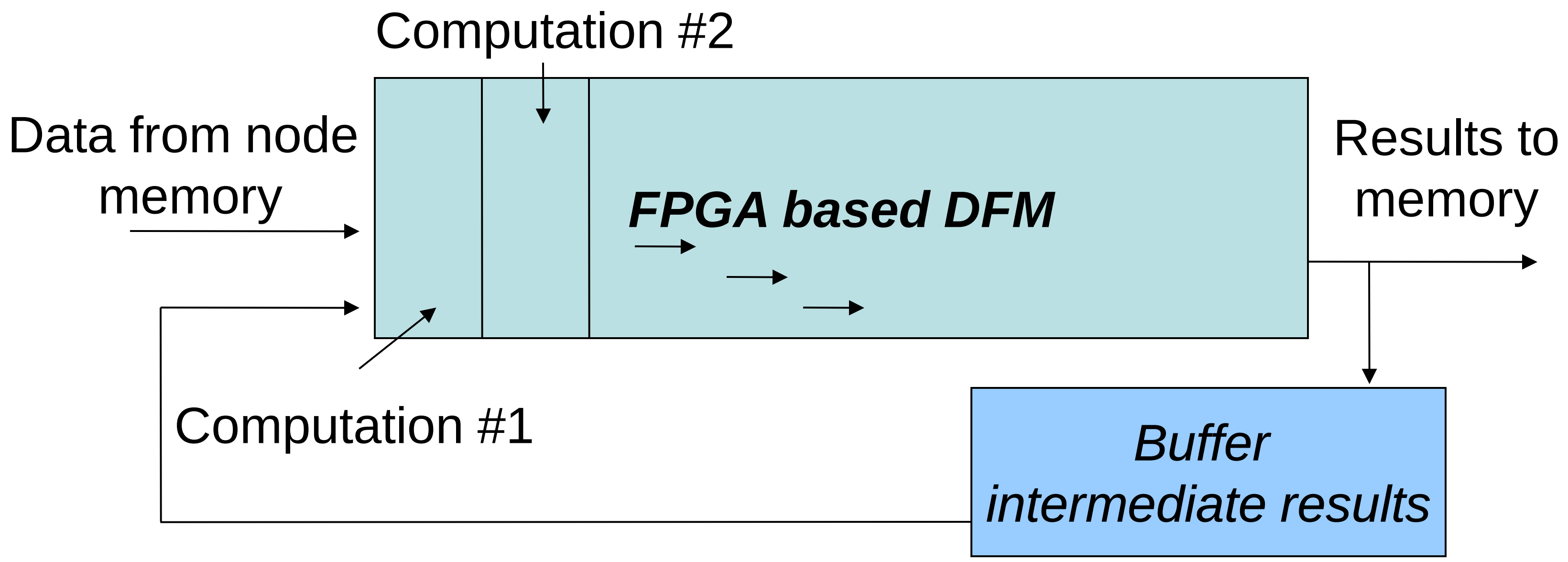
Acceleration with Static, Synchronous, Streaming DFMs

- Create a static DFM (unroll loops, etc.); the goal can be throughput (compute intensive) or latency (networking).
- Create a fully synchronous DFM synchronized to multiple memory channels. The time through the DFM is always the same.
- Stream computations across the long DFM pipelined array.
- If silicon area and pin BW allow, create multiple copies of the DFM (as with SIMD or vector computations).
- Iterate on the DFM aspect ratio to optimize speedup.

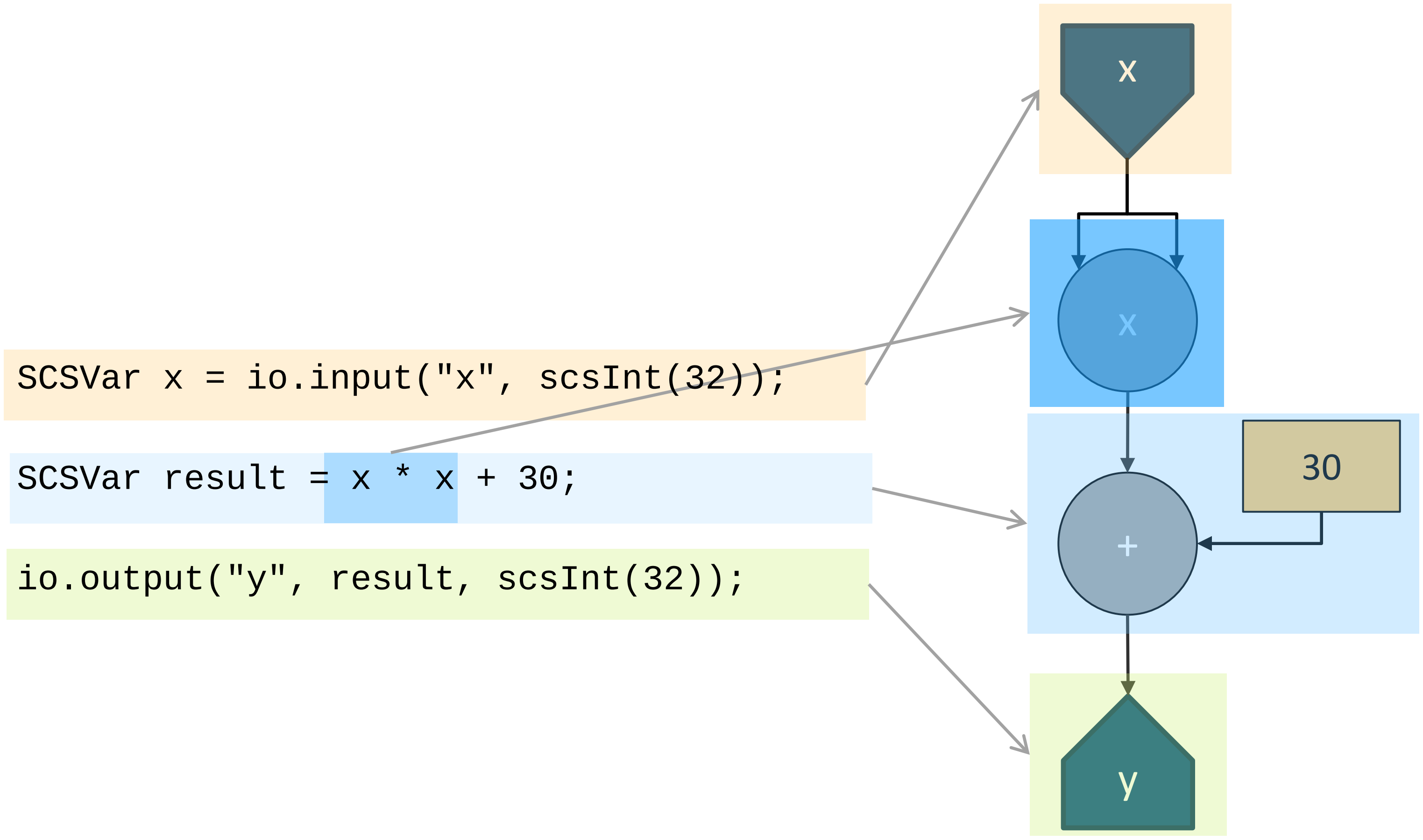
Acceleration with Static, Synchronous, Streaming DFMs

- Create a fully synchronous data flow machine synchronized to multiple memory channels, then stream computations across a long array

PCIe accelerator card w memory is DFE (Engine)



graphic dataflow programming: $X^2 + 30$



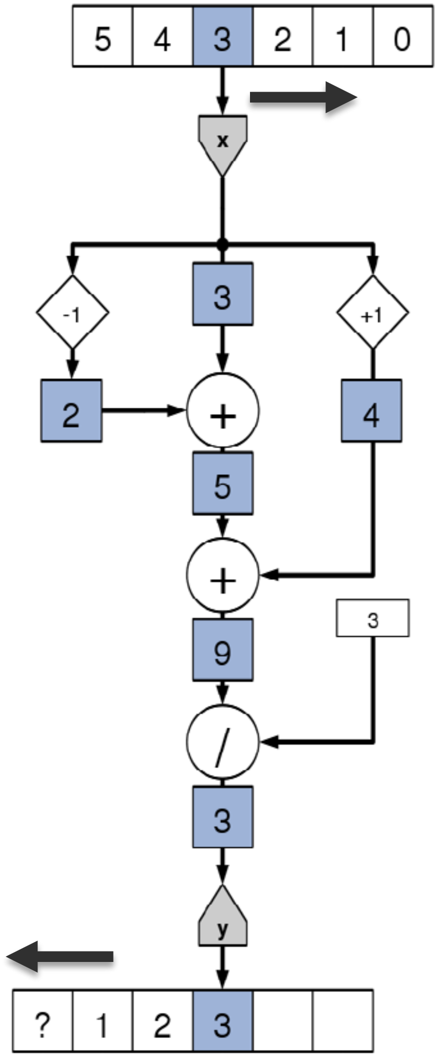
```
SCSVar x = io.input("x", scsInt(32));

SCSVar result = x * x + 30;

io.output("y", result, scsInt(32));
```

MAXELER Technologies

Dataflow: Moving Average

$$Y = (X_{n-1} + X + X_{n+1}) / 3$$

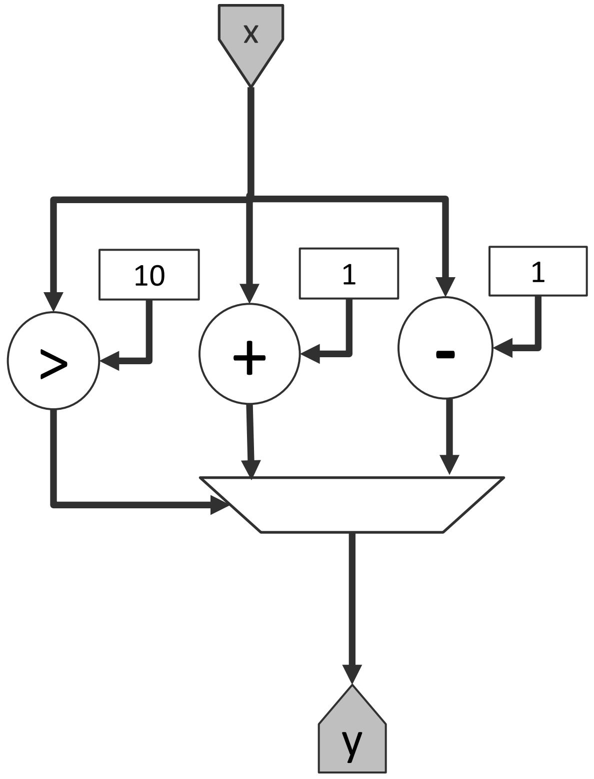
```
SCSVar x = io.input("x", scsFloat(7,17));
SCSVar prev = stream.offset(x, -1);
SCSVar next = stream.offset(x,  1);
SCSVar sum = prev + x + next;
SCSVar result = sum / 3;
io.output("y", result, scsFloat(7,17));
```



MAXELER Technologies

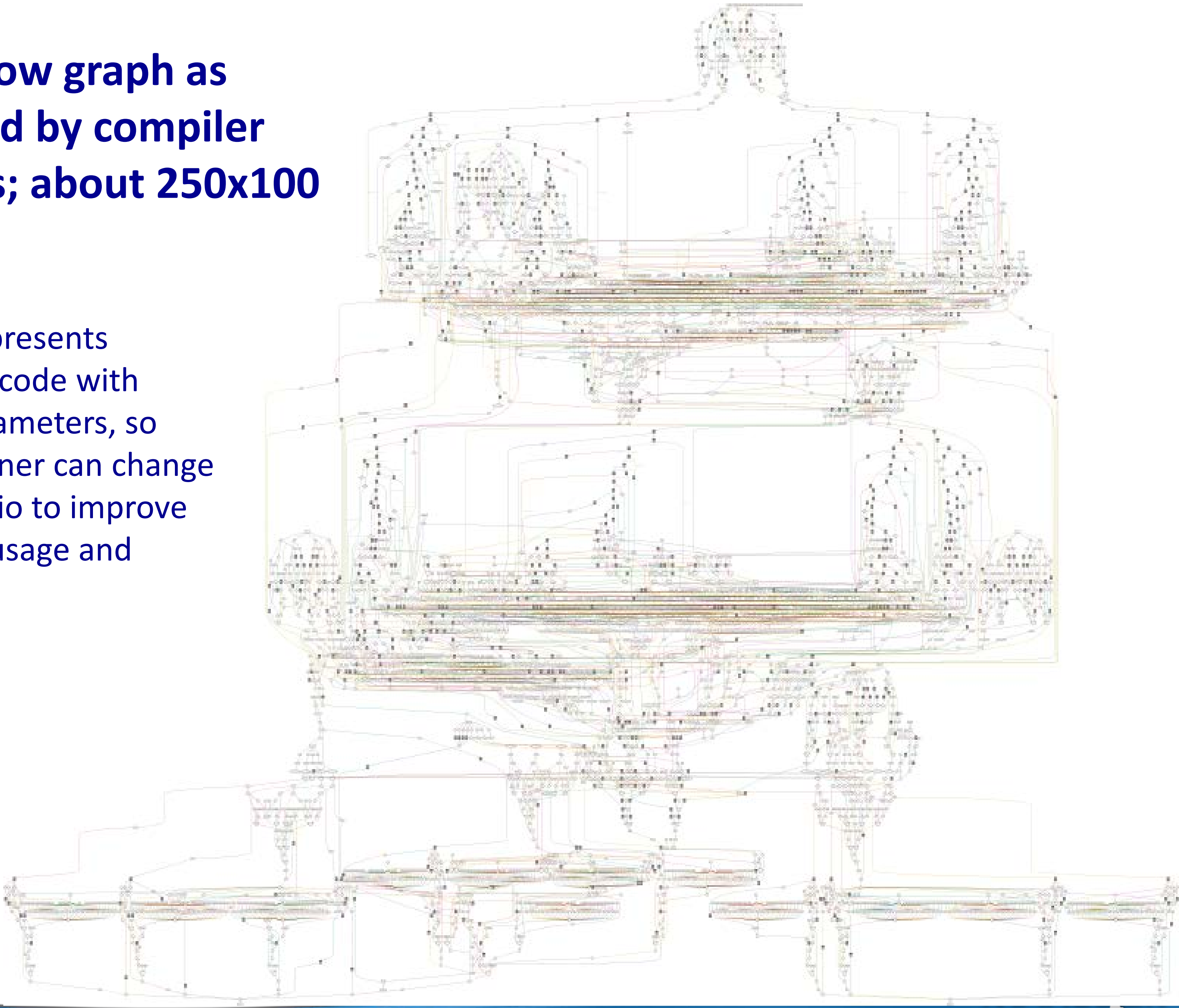
Dataflow: Choices

```
SCSVar x = io.input(“x”, scsUInt(24));
SCSVar result = (x>10) ? x+1 : x-1;
io.output(“y”, result, scsUInt(24));
```



Data flow graph as generated by compiler 4866 nodes; about 250x100

Each node represents a line of JAVA code with area time parameters, so that the designer can change the aspect ratio to improve pin BW, area usage and speedup

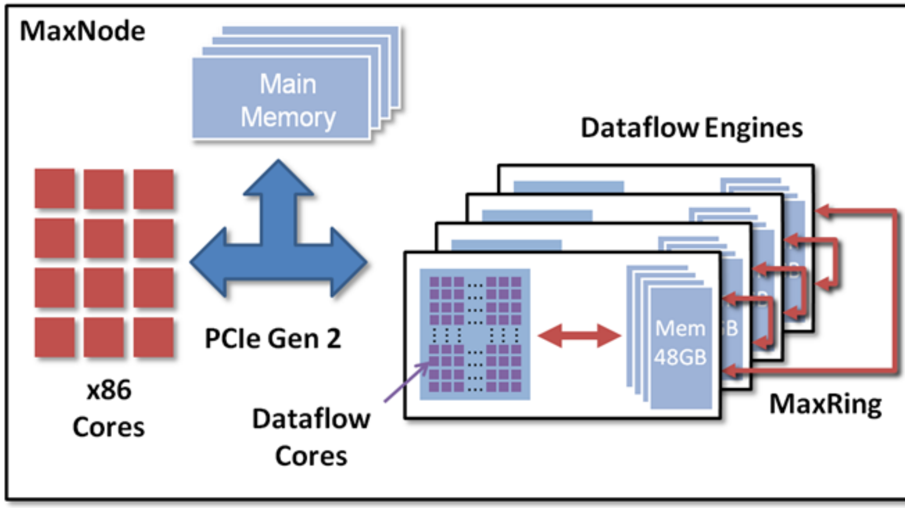


As with SMS; it's the software that counts

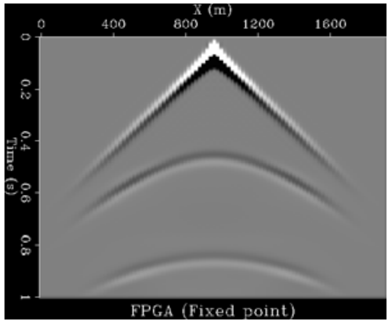
- Compiler provides graphic interface (ALD like)
- Nodes are grouped by designer into actions
- Compiler organizes data flow streams and memory choreography…new data to pins each cycle
- Provides both area and timing information
- Enables cycle accurate simulation of the design. Since the design is static performance is known
- When the design is complete (with speedup determined) the place and route is invoked.

MPC-C500, for compute intensive apps

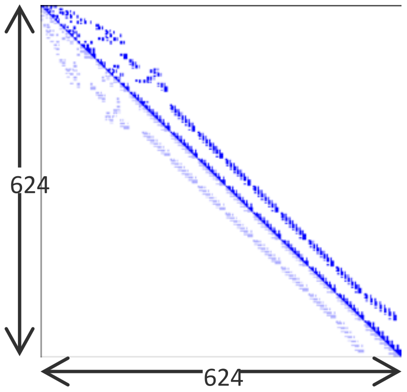
- 1U Form Factor
- 4x dataflow engines
- 12 Intel Xeon cores
- 192GB DFE RAM
- 192GB CPU RAM
- PCIe Gen2 x8
- *MaxRing* interconnect
- 3x 3.5” hard drives
- Infiniband



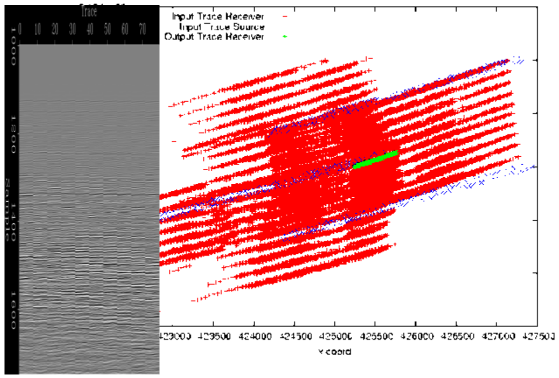
Achieved Computational Speedup for the entire application (not just the kernel) compared to Intel server



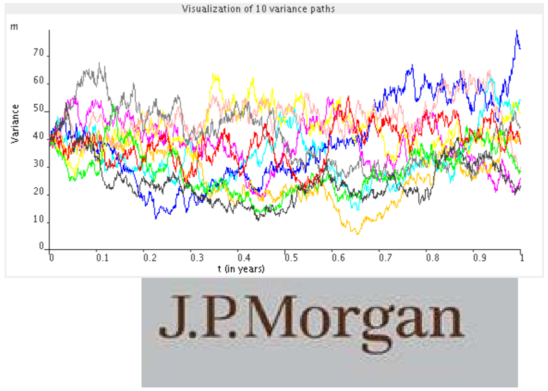
RTM with Chevron VTI 19x and TTI 25x



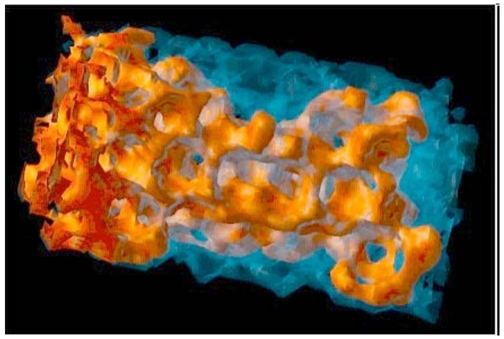
Sparse Matrix 20-40x



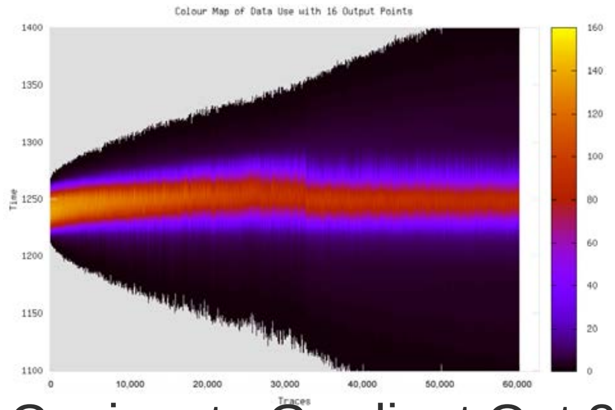
Seismic Trace Processing 24x



Credit 32x and Rates 26x



Lattice Boltzman Fluid Flow 30x



Conjugate Gradient Opt 26x

Programming model is available as OpenSPL   http://www.openspl.org/

- Open spatial programming language, an orderly way to expose parallelism
- Control and Data-flows are decoupled
  - Both are fully programmable
- Operations exist in space and by default run in parallel
  - Their number is limited only by the available space
- 2D dataflow is programmer's model, JAVA the syntax
- Could target hardware implementations.
  - map on to CPUs (e.g. using OpenMP/MPI)
  - GPUs, other accelerators

So for HPC, how can dataflow emulation with FPGAs be better than multi core?

- FPGAs emulate the ideal data flow machine
- Success comes about from their flexibility in matching the DFG with a synchronous DFM and streaming data through and shear size > 1 million cells
- Effort and support tools provide significant application speedup
- With a really effective dedicated dataflow chip there’s probably 2-3 orders of magnitude improvement possible in Area x Time x Power.

So there’s lots more work to be done