Accelerate Performance on the Paallel Programming Super Highway

National Instruments, 11500 N Mopac Expwy, Austin, TX, 78759-3504

Presented at the 22nd Systems and Software Technology Conference (SSTC), 26-29 April 2010, Salt Lake City, UT. Sponsored in part by the USAF. U.S. Government or Federal Rights License

Security Classification of:

- Report: Unclassified
- Abstract: Unclassified
- This Page: Unclassified

Limitation of Abstract: Same as Report (SAR)

Number of Pages: 25
Presentation Highlights

- Computational programming demands continue to increase at a rapid pace despite technological challenges and limitations.
- Parallelism is the [new] principal method for increasing and improving processor performance.
- Dataflow programming languages address several barriers associated with parallel programming.
- Dataflow languages ought to be considered along with traditional (imperative) programming solutions.

Email garth.black@ni.com with questions
Programming Demands and Limitations

- Rising demand for faster execution and increasingly complex programming
- Clock frequency (speed) is trending to an asymptotic condition (3 GHz)
- Moore’s Law may still be valid, but the Law of Thermodynamics is also valid
- Parallel Programming options exist, but can be complicated
Just increase Clock Frequency?

- **Old (Conventional Wisdom)**
  - Increasing clock frequency is the primary method of improving processor performance.

- **New [conventional wisdom]:**
  - Increasing parallelism is the primary method of improving processor performance.

- “Even representatives from Intel warned that traditional approaches to maximizing performance through maximizing clock speed have been pushed to their limit.”

The Human Parallel Processor

- Billions of Nerve Cells (Neurons)
- Networks of neurons form massive parallel processing system
- Parallelism: Vision, Hearing, Motion

Image courtesy of Wikipedia
“Massive” CPU Parallel Processor

“Massively Parallel Processor”

- A cabinet from Blue Gene/L, ranked as the fourth fastest supercomputer in the world.
- More than 100 CPUs with high speed interconnect
- Analogous to Human Brain
How do we program Parallel Processes?

- Newsweek Article
  *(Moore’s Law Doesn’t Matter; August 15, 2009)*
- Imperative vs. Dataflow programming
Imperative Programming vs. Dataflow Programming

- Imperative programming is modeled as a series of operations, the data paths between operations being effectively invisible
  - Examples: C/C++, Fortran, Pascal

- Dataflow programming explicitly illustrates the “flow of data” between program operations
  - Examples: SISAL, SAC, LabVIEW, VEE
## Contrast: Imperative Programming vs. Dataflow Programming

<table>
<thead>
<tr>
<th>Imperative Language</th>
<th>Dataflow Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line 1: ( x = 5 )</td>
<td>( y \times 3 \rightarrow z )</td>
</tr>
<tr>
<td>Line 2: ( y = x + 1 )</td>
<td>( x + 1 \rightarrow y )</td>
</tr>
<tr>
<td>Line 3: ( z = y \times 3 )</td>
<td>( 5 \rightarrow x )</td>
</tr>
</tbody>
</table>

Execute each statement in order.

Identify all rules and then provide inputs.

The compiler determines that \( y \) needs to be calculated before \( z \).
Contrast: Imperative Programming vs. Dataflow Programming

**Imperative Language**

Line 1: \( x = 5 \)
Line 2: \( y = x + 1 \)
Line 3: \( z = y \times 3 \)

**Dataflow Language**

The exact order of rule statements is not important in dataflow code!

- \( x + 1 \rightarrow y \)
- \( y \times 3 \rightarrow z \)
- \( 5 \rightarrow x \)

**Execute each statement in order.**

**Identify all rules and then provide inputs.**
Dataflow Programming correlates to standard flowchart models

Example: A flow chart represents the relationships between inputs and outputs. Dataflow programming uses the same “flow” paradigm.
Formula: Result = (A+B*C) / (D-E)
Dataflow Languages Enable Automatic Parallelization

Both the multiply/add and subtract operations can execute at the same time
Dataflow Languages Naturally Express Parallel Applications

Task Parallelism
Dataflow Languages Naturally Express Parallel Applications

Data Parallelism
Sequential Operations

- CPU Core
- Acquire
- Filter
- Analyze
- Log
Parallel Computing: Pipelining

Diagram showing the process of pipelining in parallel computing. The diagram illustrates the flow of tasks such as Acquire, Filter, Analyze, and Log across different CPU cores at specific time points $t_0$, $t_1$, $t_2$, and $t_3$. The tasks are executed in sequence, with each CPU core handling different stages of the process.
Parallel Operations on Multiple CPUs

CPU #1
Closed-Loop Control Thread
- Process Variable
- Output Variable

CPU #2
Data Logging Thread
- Input Data
- Resulting FFT

CPU #3
Mathematical Algorithm

CPU #4
Signal Processing
- Result
Dataflow Languages are Actively Used in Academia and Industry

- **Academic Efforts**
  - SISAL (University of Manchester & Colorado State)
  - LUSTRE (University of Victoria)

- **Commercial Products and Standards**
  - VHDL (based on IEEE standards)
  - National Instruments LabVIEW
  - Agilent VEE
  - Northwoods Software Sanscript

- Many others...
Outside the CPU sphere: Other Parallel Hardware Targets

- Market is demanding smaller, cheaper, faster targets
- FPGAs, DSPs, Embedded Real-time products
- Programmable hardware targets are converging
## Advantages and Caveats of Dataflow Languages

<table>
<thead>
<tr>
<th>Caveats</th>
<th>Advantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Typically no by-reference data accesses (by-value only)</td>
<td>Can be automatically be mapped to parallel hardware including multicore CPUs</td>
</tr>
<tr>
<td>Some overhead due to run-time scheduler (if present)</td>
<td>Naturally expressed graphically; can improve productivity</td>
</tr>
<tr>
<td>Different paradigm from imperative languages: requires a learning curve</td>
<td>May reduce the need for multiple development tools</td>
</tr>
</tbody>
</table>
Increasingly parallel embedded hardware warrants new methods of parallel software development

Dataflow languages can address some major challenges associated with parallel programming

Many dataflow languages exist today, and should be considered along with other programming solutions

Email garth.black@ni.com with questions
NI Support at HAFB

- Skilled engineering & developer support. Current work includes:
  - Solar Radiometer System (Embedded Real-time)
  - EFV (Expeditionary Fighting Vehicle)
  - CBATS Test Platform
  - Metrology Lab

- Base Contractor’s Badge

- Familiarity and history with base operation
  - Weekly Visits
  - Complimentary - quarterly training sessions
References:

How to contact me?

Garth Black
Field Engineering Manager
N. Utah & E. Idaho
Phone: (801) 447-3343
E-mail: garth.black@ni.com