Kahn Process Networks as FPGA Overlay Architectures

Mike Butts
personal project

First Workshop on Overlay Architectures for FPGAs (OLAF)
Berkeley Wireless Research Center

18 November 2013
KPN on FPGA

- The Kahn Process Network (KPN) is a well known but underused model of parallel computation, especially well suited to embedded computing of streaming real-time data.
  - It combines low development and debug effort with efficiency, modular reuse, reliability and extreme scalability.

- A 336 processor KPN in a single silicon device with tools were commercially released by Ambric in 2007.
  - It found wide acceptance in real-time embedded video, computer vision, baseband wireless and medical imaging applications, as well as university research.

- Today's FPGAs are ideal silicon platforms for KPN embedded systems.
  - A simple KPN implemented as an FPGA overlay architecture is presented.
Model of Computation: Kahn Process Network

**Process Domains**

- PN
  - Kahn-MacQueen Process Network
  - Components are sequential processes that run concurrently
  - Communication channels are **unbounded** FIFOs
    - Get operation blocks until data is available.
    - Processes cannot poll for data
  - Deterministic execution
  - Bounded memory with blocking writes
  - Good for streaming signal processing applications

- Kahn Process Network (KPN) with bounded FIFOs.
  - FIFO channels carry data and control, and strictly preserve sequence.
Processor Behavior with Channels

- KPN processors are interconnected by KPN FIFO channels
  - Output to channel stalls processor if the channel is full.
  - Input from channel stalls processor if the channel is empty.
  - Channel tutorial: http://inst.eecs.berkeley.edu/~cs150/Documents/Interfaces.pdf

- Each processor runs independently, in an asynchronous system
  - Synchronized by input and output channels
  - Developer has no HW issues like timing closure

- Each inter-processor transfer is *communication and synchronization*
  - Can’t do one without the other, it’s self-synchronizing.
Choose the Right Parallel Programming Model \textit{First}

Everyone draws block diagrams, great!

Everyone wants to write software, great!

\begin{verbatim}
for (int r = 1; r <= numRefPictures; r++)
{
    // generate TMS workload requests for current macroblock
    outCtl.writeTag(outCtl.intOfTag("TagCurrentBlock"));
    outCtl.writeDEO(((mbSizeX + 3) >> 2) * mbSizeY);
    GenerateWorkloadCB(outTms, addrPtr[0] + j, picLinePitch[0]);

    // zero motion vector
    outCtl.writeTag(outCtl.intOfTag("TagMotionVector"));
    outCtl.writeDEO(0);

    int xRadius = searchRFAPxRadius[s];
\end{verbatim}

\textbf{Software objects running on processors} \hspace{2cm} \textbf{in a structure of self-synchronizing channels} \hspace{2cm} \textbf{== KPN}

KPNs as FPGA Overlay Architectures - Mike Butts - OLAF13
Published Research on KPNs in Silicon

- Lesley Shannon, Paul Chow et al - Simon Fraser U., U. Toronto

- Mike Butts, Mark Jones, Paul Wasson – Ambric
  - Am2045 arch., chip, tools: FCCM 2007
  - Reconfigurable Work Farms: FCCM 2008

- Zain ul-Abdin, Bertil Svensson et al - Halmstad U., Sweden
  - Occam-pi language for KPN: FPL 2009
  - Real-time autofocus for SAR: FCCM 2011
  - 3D structure tensor (3D-STA) for motion estimation: ReConFig 2012

- Brad Hutchings, Brent Nelson, Reed Curtis, et al - BYU
  - Optical Flow on Am2045: FCCM 2009
FPGA has hard silicon for KPN elements

- Processors: 25x18 MACs, 1Kx18 dual-port RAMs, 32x2 reg files
- Channels: 32-deep FIFOs ≈ 1 LUT per bit (SRL32 shift-register mode)
- Memories: 1Kx18 BlockRAM FIFOs

Xilinx 7-series

25x18 MAC/ALU datapath

Block RAM with FIFO pointers
Q18: Simple KPN in FPGA Silicon

- KPN starter system: Q18
  - Open-source HW/SW
  - 18-bit integer processors with MAC, 1K word local memory
  - Configurable 18-bit KPN channels
    - Configurable FIFOs up to 1K
  - Assembler, Interconnector
  - Interactive monitor on host via USB

- Target platform: Xilinx Spartan-6
  - Atlys board: XC6SLX45 + SDRAM, etc.
  - 50 Brics = 50 CPUs, 50 FIFO switches

- Target applications:
  - KPN teaching and research
  - Computer Vision, Software-Defined Radio, Audio Synthesis

- Expecting first open-source release in early 2014
Each bric has one CPU, and one FIFO Switch (FS) node.

Circuit-switched configurable interconnect
- Nearest-neighbor channels connect CPUs NESW.
- Long-distance routes through FIFO Switch network.
Input/output channels are first-class elements of ISA, like general registers.
- 5-bit src/dest fields: registers 0-23, channels 0-7
- register+register, register+channel, channel+channel → register, channel

Zero-overhead looping for one MAC per cycle, channel throughput of 1.

Local 1Kx18 Block RAM for instructions, R/W data thru channels.
**Q18 FIFO Switch**

- 1 or 2 FIFOs (Block RAM)
  - Up to 1K deep total
  - From any input to any output
- 16-deep FIFO on each output
- All configured thru Panel Bus

- Two channels output in each direction
  - From either channel in the same direction, or turn corner from same-numbered channel, or either CPU output, or either BRAM FIFO.
  - Each output can have Alt/Or/And Join.
4-bit daisy chain of addressable “front panels”

- Host reads/writes control and status registers, and all Block RAMs, unintrusively.
- Load code, data, configure switches and FIFOs.
One or more brics may have Hardware Objects instead of CPUs.

- **HW** must observe KPN channel rules: stall on empty in, full out.

- Any mix of HW and SW objects for domain-specific applications.
Potential Scale & Speed of KPN Overlays

- Estimates based on available Block RAMs, DSP48s and logic
- One Q18 bric == one CPU processor + one FIFO switch
  - about 800 logic cells, 1 DSP48, 2 block RAMs
- Xilinx 6-series on 40nm silicon:
  - 100-200 MHz (easy synthesis vs. handcrafted)
  - Spartan-6 family: 16 to 130 brics, $1 each
- Xilinx 7-series on 28nm silicon:
  - 200-400 MHz
  - DSP48 has ALU, Block RAMs have FIFO hardware
  - Zynq-7 family: 30 to 375 brics + dual-core ARM A9 SoC
  - Artix-7 family: 50 to 350 brics
  - Virtex-7 family: 750 to 1900 brics
- Altera presumably similar.
Conclusion

- Kahn Process Network is an effective and powerful model of computation for massively parallel streaming embedded systems.
  - Shown by research and production on real hardware platforms: ASIC and FPGA

- FPGAs are a powerful and scalable hardware platform for continuing to develop and use KPNs in real systems.

Gilles Kahn 1946-2006