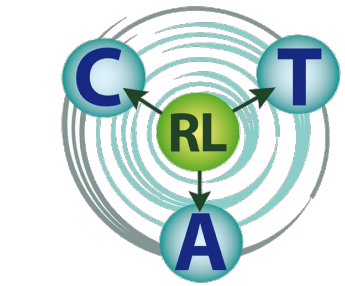
Peak Performance Model for a Custom Precision Floating-Point Dot Product on FPGAs

UCHPC - UnConventional High performance Computing Workshop
Europar 2010

Manfred Mücke, Bernd Lesser, Wilfried N. Gansterer
{manfred.muecke|bernd.lesser|wilfried.gansterer}@univie.ac.at



universität wien

Research Lab Computational Technologies and Applications
University of Vienna
http://rlcta.univie.ac.at

August 30th, 2010

Motivation

Accelerating scientific applications

- For instance:
  - accelerating linear solvers
  - accelerating matrix operations
  - ...
- A central part of many scientific computing applications: dot-product operation

Our work deals with

Performance analysis of custom-precision dot-product architectures on FPGAs

Why on FPGAs?

There are applications that do not require double-precision data types:

- Keep double precision range (11bit exponent)
- Reduce mantissa (mantissa bit width ≤ 52)

On CPUs / GPUs:

- Speedup can only be achieved if mantissa bit width
  - $= 23$ bit (single precision) or
  - $= 10$ bit (half precision)

On FPGAs:

- FPGAs are the only hardware platform that can benefit from bit width reduction on a fine-scaled level
- Lower precision translates directly into
  - increased parallelism $\rightarrow$ throughput $\rightarrow$ SPEEDUP
- Larger FPGAs translate into
  - increased parallelism $\rightarrow$ throughput $\rightarrow$ SPEEDUP

Dot-product:

- Our observation: The maximum size of a parallel floating-point dot-product on FPGAs scales superlinearly with decreasing mantissa bit width

Question:

How much more performance can we gain?

Goal:

- Give a quantitative model for the performance improvement as function of the mantissa bitwidth

Architecture

Canonical Dot-Product

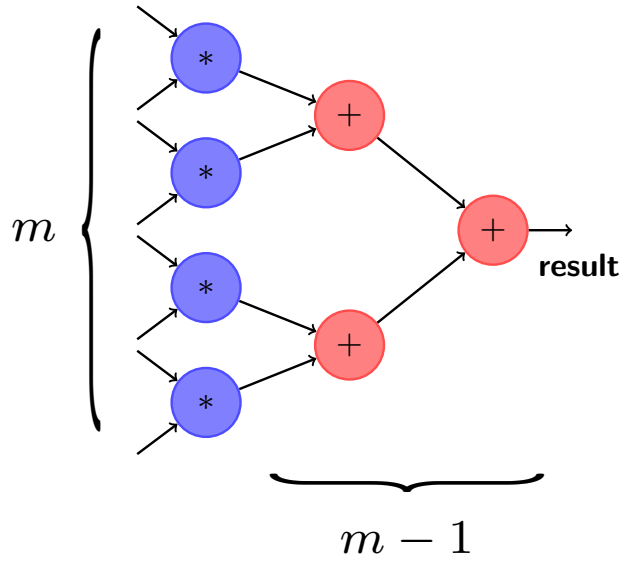
for real valued input vectors $\mathbf{a}, \mathbf{b}$:

$$< \mathbf{a}, \mathbf{b} >= \mathbf{a}^T \mathbf{b} = \sum_{i=1}^{n} a_i \cdot b_i.$$

- Different possibilities to implement a dot-product in hardware
- Our choice: binary-tree based dot-product architecture

Thus:

- m parallel multipliers
- $m - 1$ adders

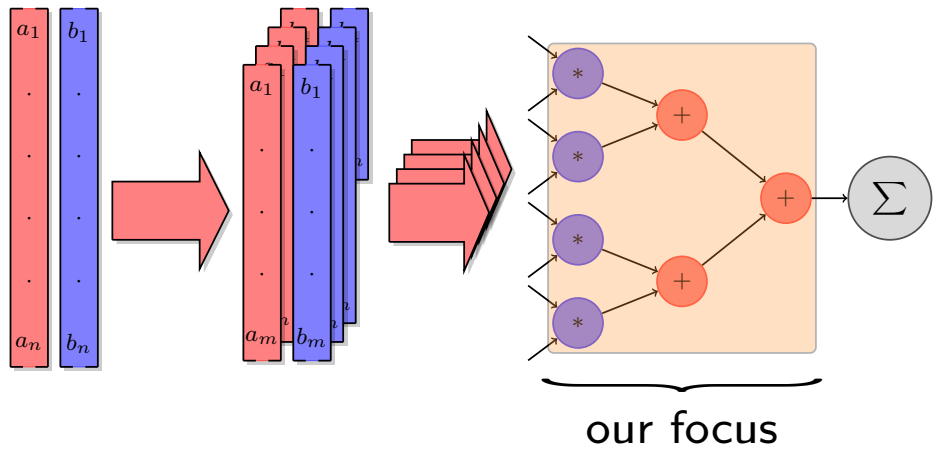


Splitting (arbitrary long) vectors:

$$< \mathbf{a}, \mathbf{b} >= \sum_{i=1}^{n} a_i \cdot b_i = \sum_{j=0}^{\lfloor \frac{n}{m} \rfloor - 1} \sum_{i=1}^{m} a_{i+j \cdot m} \cdot b_{i+j \cdot m} + \sum_{i=\lfloor \frac{n}{m} \rfloor \cdot m+1}^{n} a_i \cdot b_i$$

We investigate:

- Custom dot-product operator accepting a maximum input vector length m
- for different floating-point mantissa bit widths

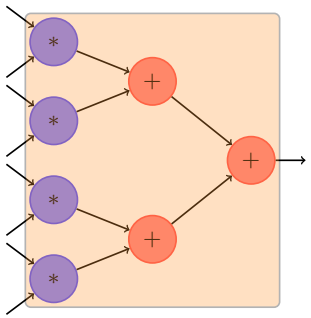


Given a certain sized FPGA, we want to know:

- Peak performance as a function of the used mantissa bit width

Dot-Product architecture: peak performance depends on

- Number of parallel multipliers m_{max}
- Maximum frequency f_{max}



Thus, we need:

- Implementation for each mantissa bit width
- Measure its hardware resource usage

Experiments

Implementation issues:

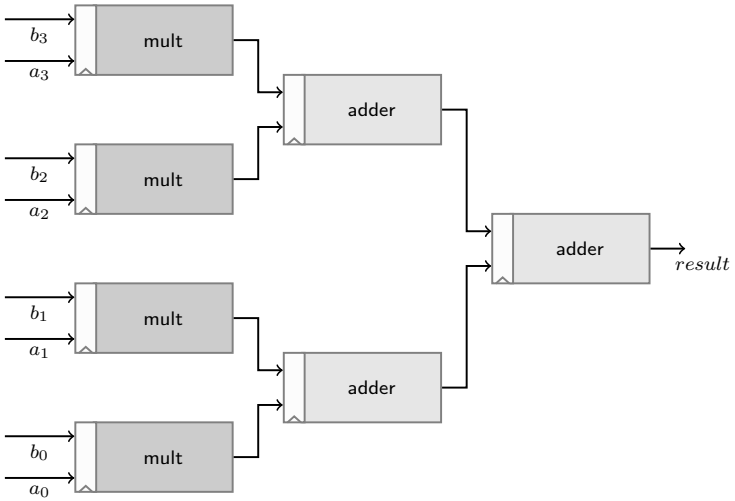
- We implemented a generic dot-product architecture for arbitrary vector lengths
- Standard IEEE 754 floating-point format
- Arbitrary precision floating-point modules:
  - chosen library: *FPLibrary* (*Arnaire project*, at *ENS Lyon*)
    `http://www.ens-lyon.fr/LIP/Arenaire/Ware/FPLibrary/`
  - combinatorial operators used

Measurement issues:

- Used synthesis tool: *QuartusII* (*Altera*)
- Automated measurements using TCL scripting language
  - Set generics
  - Synthesize implementation
  - Record hardware resource usage

Our implementation:

- Accepts generic parameters: mantissa bit width, exponent bitwidth, m
- m parallel multipliers (accepts $2m$ input operands)
- Binary adder tree of depth $\lceil log_2 m \rceil$
- Stages pipelined (registers)
- Total latency: $\lceil log_2 m \rceil + 3$



Peak performance:

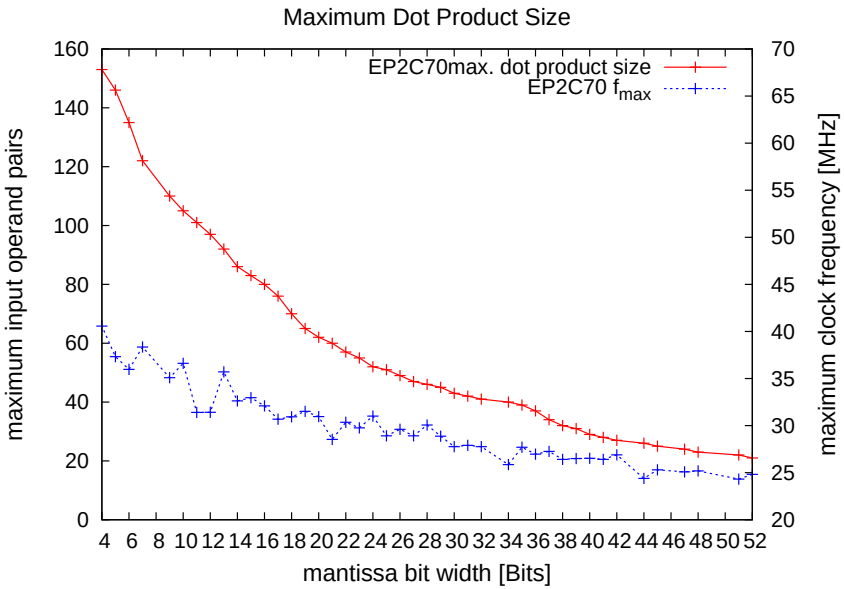
- $P = (2m - 1) * f_{\text{max}}$ [Flop/s]

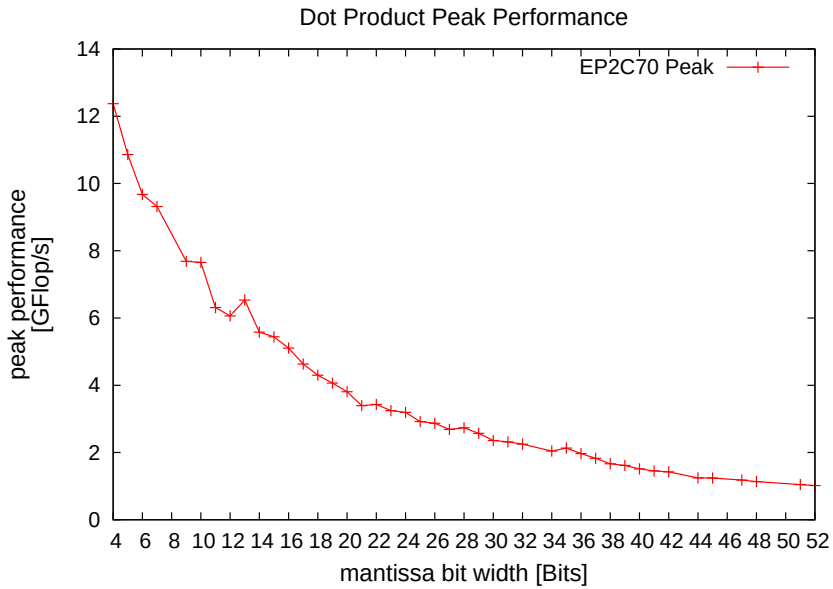
Methodology:

- First: we perform measurements on largest Cyclone II FPGA device (EP2C70)
- Then: Develop model for approximating best these original measurements
- Finally: Verify the model class with the measurements obtained from two more recent devices

FPGA Device	FPGA Family	Logic elements	DSP blocks [9x9bit blocks]	Emb. Memory [kbits]
EP2C70	Cyclone II	68,416	300	1,125
EP3C80	Cyclone III	81,264	488	2,745
EP3SL70	Stratix III	67,500	576	2,214

Table: Hardware resources of used FPGAs.





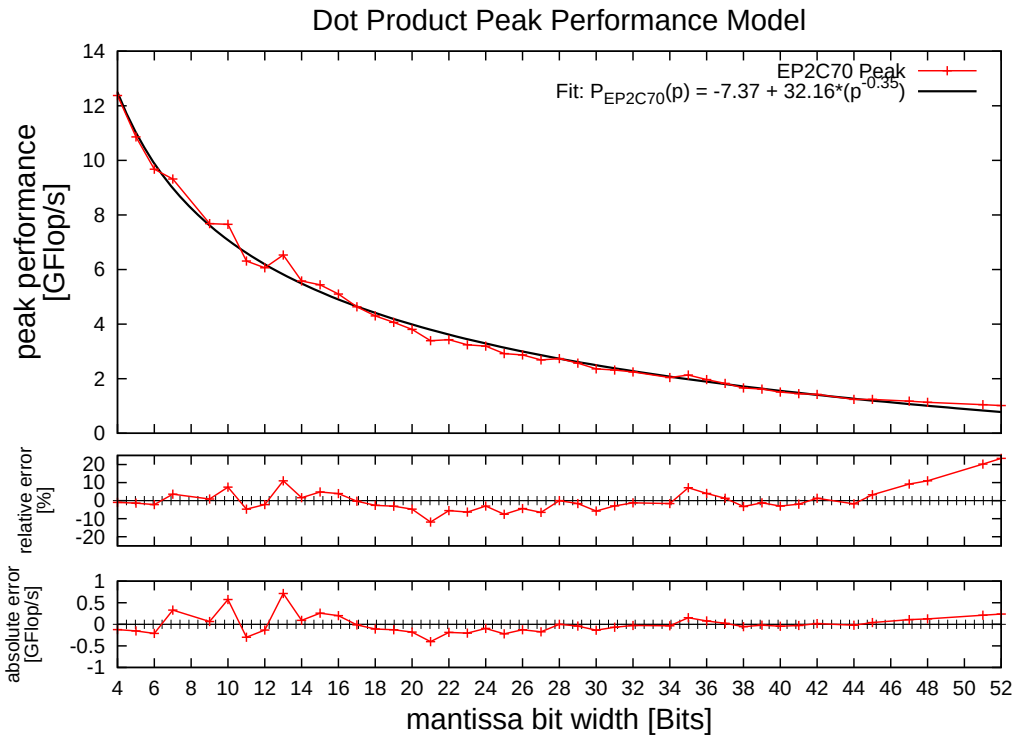
FPGA Peak Perf vs. Mantissa bit width

- Measure maximum dot-prod size $m_{\max}$ and maximum frequency $f_{\max}$
- Mantissa sizes: 52 downto 4
- Calculate peak performance $P = (2m - 1) * f_{\max}$ [Flop/s]
- Observation: peak performance grows exponentially

Dot-product performance model

Model:

- Fit: fractional polynomial of form $P(p) = c_1 + c_2 \cdot p^{c_3},\ c_1, c_2, c_3 \in \mathbb{Q}$
- *EP2C70*: $P(p) = -7.37 + 32.16 \cdot p^{-0.35}$



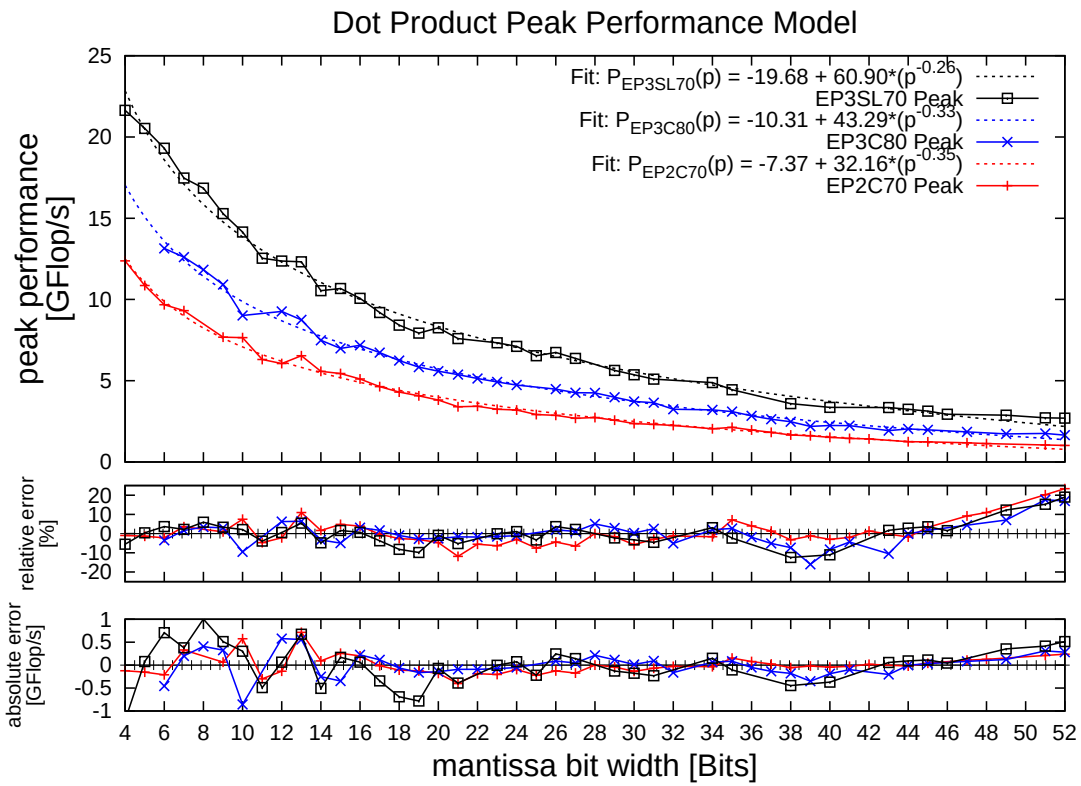
$$\mathsf{P} = (2m-1) * f_{\mathsf{max}}\ [\mathsf{Flop/s}]$$

$P :=$ Measured value

$\hat{P} :=$ Modelled value

$$\mathsf{Error}_{\mathsf{rel}} = \frac{(P-\hat{P})}{P} \cdot 100\ [\%]$$

$$\mathsf{Error}_{\mathsf{abs}} = P - \hat{P}\ [\mathsf{Flop/s}]$$



Verify observations on more recent FPGA devices (families):

- Given appropriate constants, peak performance as a function of mantissa bit width can be modeled quite accurately
- Maximum absolute error: 1GFlop/s
- Average relative error: $\approx 5 - 7\%$

Conclusions

FPGA	Perf. Model [GFlop/s]	Device family
EP2C70	$-7.37 + 32.16 \cdot p^{-0.35}$	*CycloneII* (EP2C5,EP2C8,EP2C15,EP2C20,EP2C35,EP2C50,EP2C70)
EP3C80	$-10.31 + 43.29 \cdot p^{-0.33}$	*CycloneIII* (EP3C5,EP3C10,EP3C16,EP3C25,EP3C40,EP3C55,EP3C80,EP3C120)
EP3SL70	$-19.68 + 60.90 \cdot p^{-0.26}$	*StratixIII* (EP3SL50,EP3SL70,EP3SL110,EP3SL150,EP3SL200,EP3SE260,EP3SL340)

Table: Performance Model Overview

We have shown:

- Performance benefit of reduced precision can be reliably quantified for the devices considered and for comparable settings
- The model is very simple and requires practically no runtime to compute
- Can serve as a tool for designers wishing to explore the design space
- Could stimulate work identifying the minimal required precision of dot-products in given applications

Future work

Future work:

- Robustness of our results (synthesis settings)
- Behaviour of different floating-point libraries
- Explore impact of pipelining

Thank you for your attention!

Questions?

Research Lab Computational Technologies and Applications
University of Vienna
http://rlcta.univie.ac.at