HLS 4 HPC: Some Missing Links

George A. Constantinides[1]

Imperial College London

January 22, 2013

[1]Thanks to: Dr Sam Bayliss, Dr David Boland

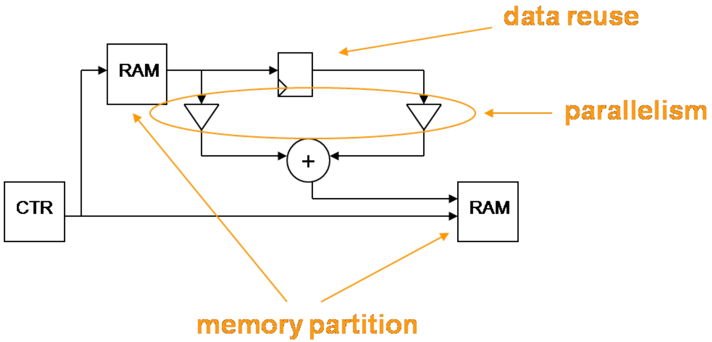
Outline

- A simple HLS example
- What are the missing links?
    - Part 1: Representing real numbers
        - Benefits
        - Number systems
        - Precision
    - Part 2: Dealing with external memory
        - The central role of SDRAM
        - Using the polyhedral model
- Some open questions

Example Code

```
#define N 1024
#define L 2
void main()
{
  double x[ N ], y[ N - 1 ];
  double k[ L ] = {0.12, 0.2};
  for( int i = 0; i < N - 1; i++ ) {
    y[ i ] = 0.0;
    for( int j = 0; j < L; j++ )
      y[ i ] += k[ j ] * x[ i - j + L - 1 ];
  }
}
```

Example Design



- Structural concerns: parallelization, memory subsystem design. External memory?
- Numerical concerns: is `double` the right number representation?

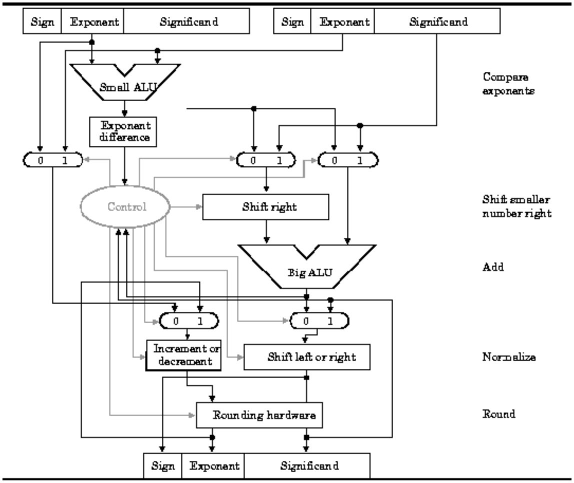
Part I

Numerics

Missing Link 1: Representing the Reals

- Hardware represents numerical data as bit strings.
- A bit string of length n can represent at most 2^n distinct values.
- Representations vary in how these strings are mapped onto reals: $f : \{0,1\}^n \rightarrow \mathbb{R}$.
  - Floating-point (s,m,e),
  - Fixed-point (s,m)/2c/1c,
  - LNS (s,e),
  - RNS (m,m,...), *etc.*
- We have always cared about using ‘enough’ precision. Now we should care about using ‘just enough’ precision.
  - Lower precision $\Rightarrow$ smaller area units, less bandwidth $\Rightarrow$ more units, more transfer $\Rightarrow$ faster.

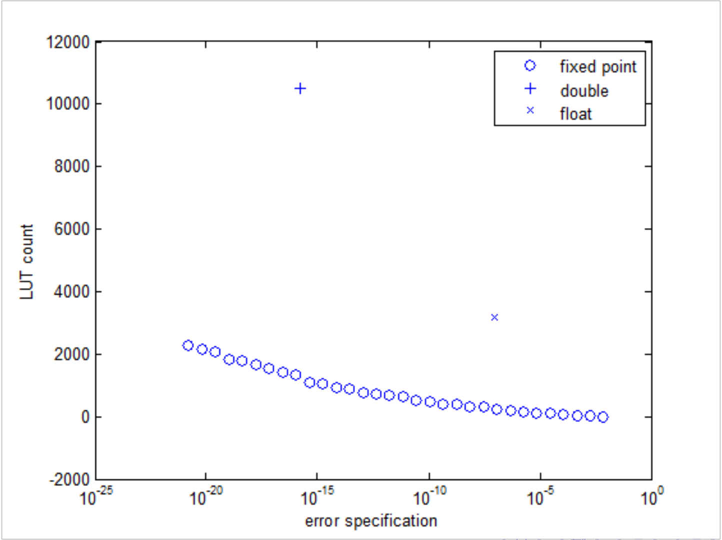
Hardware Implications



(from http://www.cise.ufl.edu/˜mssz/CompOrg/CDA-arith.html)

Numerics Matter!

Area / Error Tradeoff for LMS Adaptive Filter



Our Vision

The Central Problem

$$\begin{aligned} &\max_{s,p} \mathrm{perf}(s,p) \\ &\text{subject to} : \forall i.\mathrm{Pre}(i) \rightarrow \mathrm{Out}(s,p,i) \in \mathrm{Accept}(i). \end{aligned} \tag{1}$$

- Here s denotes circuit structural parameters, and p denotes circuit precisions. Note plural - we are in a parallel environment, so specialise!
- i denotes inputs, $Pre(i)$ denotes precondition predicate, $Out(s,p,i)$ denotes output, $Accept(i)$ denotes acceptable outputs.
- For simplicity, let's restrict ourselves to fixed s.

The General Setting

A toy sample problem would be

Toy Problem

$$\begin{array}{ll} \max_{p \in \mathbb{Z}_{++}} \operatorname{perf}(p) & \\ \text{subject to:} & \forall (a,b) \in \mathbb{R}^2 . m \le a \le M \wedge m \le b \le M \rightarrow \\ & \forall \delta \in \mathbb{R} . |\delta| \le 2^{-p} \rightarrow \left|\frac{\delta}{a+b}\right| < \epsilon . \end{array} \tag{2}$$

This corresponds to a fixed-point addition of a and b with a condition on the relative error of the result.

Reformulation of Toy Problem

$$\begin{array}{ll} \max_{p \in \mathbb{Z}_{++}} \operatorname{perf}(p) & \\ \text{subject to:} & \neg \exists (a,b,\delta) \in \mathbb{R}^3 . m \le a \le M \wedge m \le b \le M \wedge \\ & -2^{-p} \le \delta \le 2^{-p} \wedge \delta^2 - \epsilon^2 (a+b)^2 \ge 0 . \end{array} \tag{3}$$

Quantifier Elimination

Notice that the feasibility condition defines a semi-algebraic set. Applying a standard quantifier elimination gives the following simplified problem, assuming $m < M$, $\epsilon > 0$.

Simplified Problem

$$\left.\begin{array}{ll}\max\limits_{p \in \mathbb{Z}_{++}} \operatorname{perf}(p) & \\ \text{subject to:} & p > -1 - \log_2 \epsilon |m|\end{array}\right\} \text{if } m > 0 \text{ and } M > 0 \tag{4}$$

$$\left.\begin{array}{ll}\max\limits_{p \in \mathbb{Z}_{++}} \operatorname{perf}(p) & \\ \text{subject to:} & p > -1 - \log_2 \epsilon |M|\end{array}\right\} \text{if } M < 0 \tag{5}$$

$$\text{infeasible otherwise.} \tag{6}$$

Objective Function

- In general, $1/\mathit{perf}$ can be approximated as a (multi-variate) polynomial.
- For our toy example, if perf is monotonically decreasing in p, then $p^* = \lfloor -\log_2 \epsilon |m| \rfloor$ for $m > 0$ and $M > 0$ and $p^* = \lfloor -\log_2 \epsilon |M| \rfloor$ for $M < 0$.
- One can imagine that with multivariate p (due to parallel implementation) and more complex examples, things get difficult!

Our status

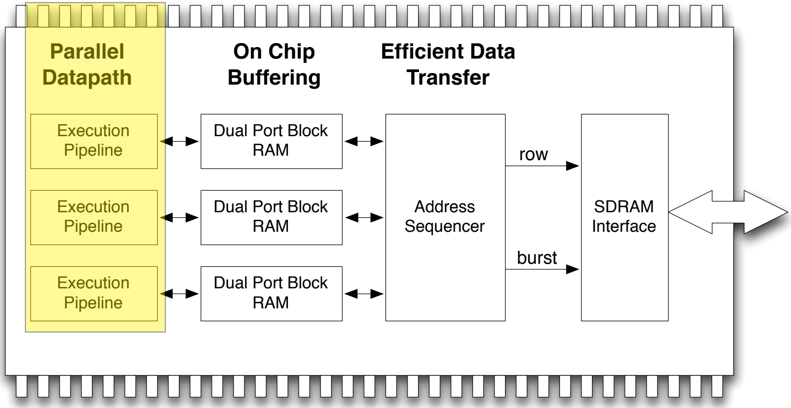
We have been focusing on:

- The feasibility problem.
- The algorithm is defined by fixed combinations of the four basic operators `*`, `+`, `/`, `-`.
- Accept(i) is defined by $f(i) + [l, u]$.
- Convex relaxations leading to computationally tractable formulations.
- Proof techniques resulting in simple machine checkable proofs.

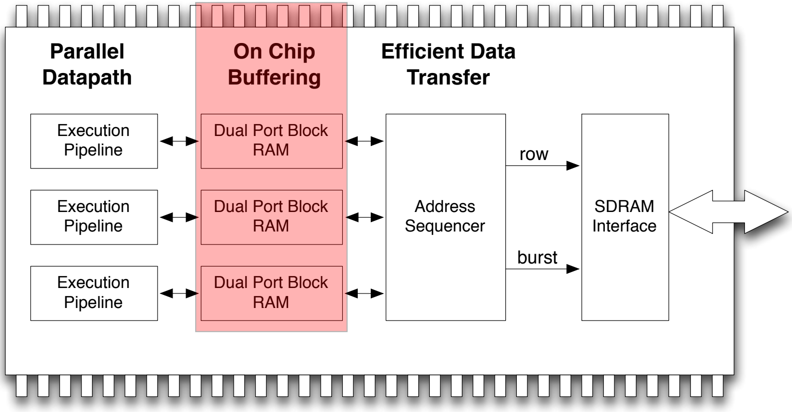
Part II

Numerics

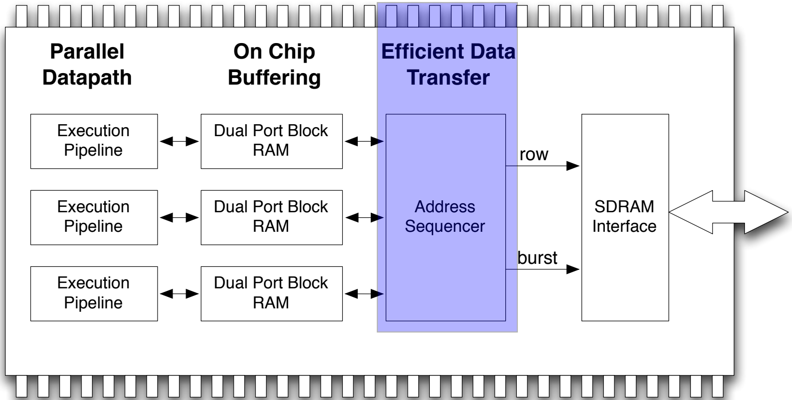
Missing Link 2: External Memory



Missing Link 2: External Memory



Missing Link 2: External Memory



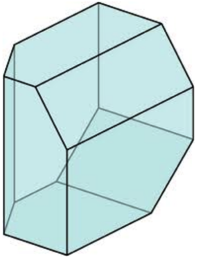
SDRAM Memory Characteristics

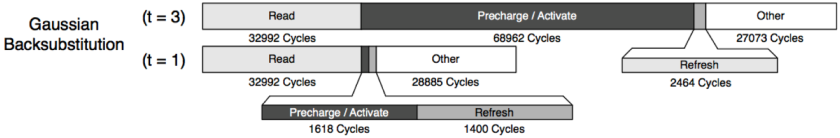
- Advantage : SDRAM is **cheap**, high capacity, commodity memory
- Disadvantage : Internal device architecture means high latency (20-30 clock cycles in FPGA)
- Physical device structure imposes timing constraints
  - Explicit 'activation' of a row before data is read from it
  - Explicit 'precharge' of a row before another row is 'activated'
- Significant bandwidth difference improvements possible when reordering external memory transactions
  - Typically $> 5\times$ bandwidth difference between optimal and worst-case performance

Use the Polyhedral Model

Affine Loop Nests

```
char A[56];
for (x1 = 0 ; x1 <= 2 ; x1++) {
    for (x2 = 2-x1 ; x2 <= 2 ; x2++) {
        for (x3 = x1; x3 <= x2 ; x3++) {
            A[7*x1+8*x2+9*x3] = x3;
        }
    }
}
```



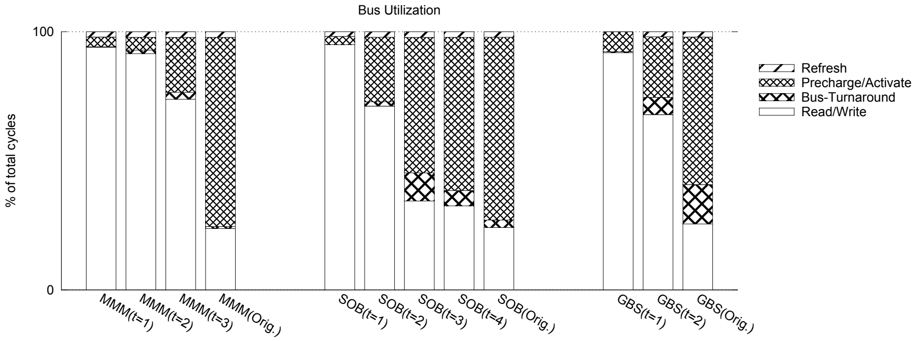


- Augmented matrix captures each loop iteration and its associated SDRAM row (r) and burst (u)

$$\begin{pmatrix} -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 7 & 8 & 9 & -R & 0 \\ -7 & -8 & -9 & R & 0 \\ 7 & 8 & 9 & -R & -B \\ -7 & -8 & -9 & R & B \end{pmatrix} \begin{pmatrix} \mathrm{x1} \\ \mathrm{x2} \\ \mathrm{x3} \\ \mathrm{r} \\ \mathrm{u} \end{pmatrix} \leq \begin{pmatrix} 0 \\ 2 \\ -2 \\ 2 \\ 0 \\ 0 \\ 15 \\ 0 \\ 3 \\ 0 \end{pmatrix}$$

$$\begin{tikzcd}
\overset{\text{5 Variables}}{\xleftrightarrow{\hspace{6em}}} \\
\begin{pmatrix} -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 7 & 8 & 9 & -R & 0 \\ -7 & -8 & -9 & R & 0 \\ 7 & 8 & 9 & -R & -B \\ -7 & -8 & -9 & R & B \end{pmatrix} \arrow[dr, Rightarrow, "\text{Using loop Transformation}"'] & & \overset{\text{3 Variables}}{\xleftrightarrow{\hspace{4em}}} \\
& & \begin{pmatrix} -29 & 12 & 48 \\ 2 & -4 & -1 \\ -1 & 2 & 0 \\ -11 & 16 & 4 \\ -1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & -2 & 0 \end{pmatrix} \\
& \begin{pmatrix} -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ 1 & 1 & -1 & 0 & 0 \\ 1 & -2 & 1 & 0 & 0 \\ -1 & -1 & 9 & -16 & 0 \\ 1 & 1 & -9 & 16 & 0 \\ -1 & -1 & 9 & -16 & -4 \\ 1 & 1 & -9 & 16 & 4 \end{pmatrix} \arrow[ur, Rightarrow, "\text{Using safe variable elimination conditions}"'] &
\end{tikzcd}$$

What was the impact on bandwidth efficiency?



Some Open Questions

- Numerics
  - Natural and scalable methods to deal with iterative algorithms where loop conditions depend on rounded variables.
  - Techniques to deal with other source of error, *e.g.* overclocking.
- Memory
  - Effective commercial tool integration
  - Optimizing SDRAM refresh for latency guarantees
  - Optimal bank partitioning