Co-design of optimization-based controllers on FPGAs

Dr Eric Kerrigan
Reader in Control Engineering and Optimization
Department of Electrical and Electronic Engineering
Department of Aeronautics
Challenge

What is the best to implement an optimal controller on an FPGA at a fraction of the compared to double precision floating point?
How Close to Optimal is a Cyber-Physical System?

\[ u^*(y) := \arg \min_u f(u, y) \]
\[ \text{s.t. } g(u, y) = 0 \]
\[ h(u, y) \leq 0 \]

\[ (p^*, c^*) := \arg \min_{p, c} \phi(p, c) \]
\[ \text{s.t. } \alpha(p, c) = 0 \]
\[ \beta(p, c) \leq 0 \]

optimal inputs \( u^*(y) \)

physical system

disturbances

computing system

optimal design parameters for physical system

co-designer

optimal design parameters for computing system

numerical errors

measurements \( y \)
Why Co-Design?

\[ f(\text{algorithm, hardware, physics}) \leq 0 \]

Trade-offs:
- **Computing system:**
  - time + energy + space
- **Physical system:**
  - performance + robustness

Explore larger set of behaviours
Why is Co-Design Challenging?

Multi-objective

Hybrid dynamics

Non-smooth/mixed-integer

Uncertainties
Model-Based Design: Model, Simulate, Elaborate, Prototype, Verify

**Design Tools**

- MATLAB
- Simulink
- Control System Toolbox
- Optimization Toolbox
- Fixed-Point Designer
- HDL Coder
- Vivado Design Suite

**Hardware-in-the-Loop Testing**

- Desktop PC
- System Model
- Embedded Processor
- Control Algorithm

measurement → control input

- System Model

Desktop PC ➔ Embedded Processor

- Hardware-in-the-Loop Testing

- MathWorks
- Xilinx

- Control Algorithm
Computing Sub-Systems and Physical Resources

- Processing
- Communication
- Storage
- Time
- Space
- Energy
## Possible Design Parameters for Computing System

<table>
<thead>
<tr>
<th>Hardware</th>
<th>Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>cost, space, energy, power</td>
<td>accuracy, termination tolerances</td>
</tr>
<tr>
<td># processors/cores/arithmetic units</td>
<td># iterations in each loop</td>
</tr>
<tr>
<td>pipeline depth</td>
<td>step length parameters</td>
</tr>
<tr>
<td>clock frequency and supply voltage</td>
<td>amount of data/results to store</td>
</tr>
<tr>
<td>memory architecture, latency, size</td>
<td>time horizon</td>
</tr>
<tr>
<td>communication architecture, bandwidth</td>
<td>complexity of physical model</td>
</tr>
<tr>
<td>number representation, word length</td>
<td>scaling of data</td>
</tr>
<tr>
<td>actuation and sampling schedule/rate</td>
<td>scheduling/communication strategy</td>
</tr>
</tbody>
</table>
Why a Field-Programmable Gate Array (FPGA)?
Size is Very Important in Microprocessor Design

Cost per die = $f(area^x)$, $x \in [2, 4]$
Computational Resources for an Adder

Xilinx Virtex-7 XT 1140 FPGA:

<table>
<thead>
<tr>
<th>Number representation</th>
<th>Registers/Flip-Flops (FFs)</th>
<th>Lookup-Tables (LUTs)</th>
<th>Latency/delay (clock cycles)</th>
</tr>
</thead>
<tbody>
<tr>
<td>double floating-point</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>52-bit mantissa</td>
<td>1035</td>
<td>852</td>
<td>12</td>
</tr>
<tr>
<td>single floating-point</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>23-bit mantissa</td>
<td>542</td>
<td>445</td>
<td>12</td>
</tr>
<tr>
<td>fixed-point</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>53 bits</td>
<td>53</td>
<td>53</td>
<td>1</td>
</tr>
<tr>
<td>fixed-point</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24 bits</td>
<td>24</td>
<td>24</td>
<td>1</td>
</tr>
</tbody>
</table>

Cheap and low power processors often only have fixed-point
Computational Resources for an Adder

Given a fixed amount of silicon (£/$/€) one can do:

200 times more fixed point additions

per second

per Joule

than floating point additions.
Floating-Point Arithmetic

**Round-off error**

```
1100000010011110001
```

```
0101010001001111111
```

```
0010010011001110000
```
Fixed-Point Arithmetic

overflow error

\[ \begin{array}{cccccccccccccc}
0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1
\end{array} \]

round-off error

overflow error

\[ \begin{array}{cccccccccccccc}
1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1
\end{array} \]

round-off error

overflow error

\[ \begin{array}{cccccccccccccc}
0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1
\end{array} \]

round-off error

overflow error

\[ \begin{array}{cccccccccccccc}
0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0
\end{array} \]
Consequences of Overflow Error

First flight of Ariane 5, 4 June 1996: $500 million loss
Lanczos Algorithm: Trade off computing effort against accuracy in eigenvalue problems, SVD, solving linear equations

\[ A = A^T. \quad Q_i^T A Q_i = T_i =: \begin{bmatrix} \alpha_1 & \beta_1 & 0 \\ \beta_1 & \alpha_2 & \ddots \\ \ddots & \ddots & \beta_{i-1} \\ 0 & \beta_{i-1} & \alpha_i \end{bmatrix} \]

**Require:** Initial iterate \( r_1 \) such that \( \| r_1 \|_2 = 1 \), \( q_0 := 0 \) and \( \beta_0 := 1 \).

1: **for** \( i = 1 \) to \( i_{\text{max}} \) **do**

2: \( q_i \leftarrow \frac{r_i}{\beta_{i-1}} \)

3: \( z_i \leftarrow A q_i \)

4: \( \alpha_i \leftarrow q_i^T z_i \)

5: \( r_{i+1} \leftarrow z_i - \alpha_i q_i - \beta_{i-1} q_{i-1} \)

6: \( \beta_i \leftarrow \| r_{i+1} \|_2 \)

7: **end for**

8: **return** \( q_i, \alpha_i \) and \( \beta_i \)
Problem: Evolution of variables in primal-dual interior point solver for Model Predictive Control of a Boeing 747

![Graph showing the evolution of variables with and without scaling method](image)
Solution: Diagonal Scaler

Theorem:

All variables in Lanczos algorithm are between -2 and 2

Can also prevent overflow due to round-off errors
Histogram of final log relative error at termination

- scaled 32-bit fixed-point
- double precision floating-point
- single precision floating-point
- unscaled single precision floating-point

\[ \log_2 \left( \frac{\|Ax - b\|_2}{\|b\|_2} \right) \]
Benefit 1:
Latency vs Silicon Usage for Same Accuracy

Xilinx Virtex-7 XT 1140 with matrices from the model predictive control of a Boeing 747
Benefit 2:
Speed vs Error for MINRES on an FPGA vs GPU

Xilinx Virtex-7 XT 1140
400MHz, 22W
>180 GOP/s/W

NVIDIA C2050
1.03 TFLOP/s
1.15GHz, 100W
10 GFLOP/s/W
Conclusions

**Number representation** is a major factor determining:
- cost, energy, size, computational speed and accuracy

Lanczos method:
- **Scale matrix**: tight analytical bounds to avoid overflow in fixed point

Hardware-algorithm **co-design** for model predictive control of a Boeing 747:
- Fix latency: < 1/10th silicon, cost and energy
- Fix silicon: > 10x faster

**MathWorks** tools from algorithm development to hardware-in-the-loop testing
- Reduces design time and costs
A Low Complexity Scaling Method for the Lanczos Kernel in Fixed-Point Arithmetic

Juan Luis Jerez, Student Member, IEEE, George A. Constantinides, Senior Member, IEEE, and Eric C. Kerrigan, Member, IEEE

Predictive Control Using an FPGA With Application to Aircraft Control

Edward Nicholas Hartley, Juan Luis Jerez, Student Member, IEEE, Andrea Suardi, Jan M. Maciejowski, Fellow, IEEE, Eric C. Kerrigan, Member, IEEE, and George A. Constantinides, Senior Member, IEEE

Embedded Online Optimization for Model Predictive Control at Megahertz Rates

Juan L. Jerez, Student Member, IEEE, Paul J. Goulart, Stefan Richter, George A. Constantinides, Senior Member, IEEE, Eric C. Kerrigan, Member, IEEE, and Manfred Morari, Fellow, IEEE