Efficient Floating-Point Logarithm Unit for FPGAs

Nikolaos Alachiotis, Alexandros Stamatakis

The Exelixis Lab,
Dept. of Computer Science,
TUM, Munich, Germany
PRESENTATION OVERVIEW

- Introduction
- Approximation Strategy
- Reconfigurable Architecture
- Performance Evaluation
- Conclusion and Future Work
• The Project:

Design of HW accelerators for Phylogenetic Inference Programs
• The Project:

Design of HW accelerators for Phylogenetic Inference Programs

Calculation of evolutionary relationships between organisms

core function: the Phylogenetic Likelihood Function
• The Project:

Design of HW accelerators for Phylogenetic Inference Programs

Ancestral probability vector

Virtual Root

Tip probability vector
INTRODUCTION

• The Project:
  Design of HW accelerators for Phylogenetic Inference Programs

• The Phylogenetic Likelihood Function:
  85% of total execution time

• Log-Likelihood Scores:
  2% of total execution time
INTRODUCTION

• The Project:
  Design of HW accelerators for Phylogenetic Inference Programs

• The Phylogenetic Likelihood Function:
  85% of total execution time

• Log-Likelihood Scores:
  2% of total execution time

Need for a resource-efficient logarithm function

Open source C implementation: ICSILog 0.6 BETA

Floating-Point number in IEEE-754 standard

<table>
<thead>
<tr>
<th>sign</th>
<th>exponent</th>
<th>mantissa</th>
</tr>
</thead>
</table>

Number = sign * $2^{exponent}$ * mantissa
**APPROXIMATION STRATEGY**

Number = sign * 2^{exponent} * mantissa

Logarithm defined only for positive values

\[
\log(\text{Number}) = \log \left( 2^{\text{exponent}} \times \text{mantissa} \right)
\]

Multiplicative property of logarithm

\[
= \log \left( 2^{\text{exponent}} \right) + \log(\text{mantissa})
\]

\[
= \text{exponent} \times \log(2) + \log(\text{mantissa})
\]
Number = sign * 2^{exponent} * mantissa

\[ \text{LOG(Number)} = \text{LOG} \left( 2^{exponent} * \text{mantissa} \right) \]

= \text{LOG} \left( 2^{exponent} \right) + \text{LOG}(\text{mantissa})

= exponent * \text{LOG}(2) + \text{LOG}(\text{mantissa})

Logarithm defined only for positive values

Multiplicative property of logarithm

Lookup Table
LOG(Value) = exponent * LOG(2) + LOG(mantissa)
LOGARITHM APPROXIMATION UNIT (LAU) ARCHITECTURE

<table>
<thead>
<tr>
<th>Sign</th>
<th>Exponent</th>
<th>Mantissa</th>
</tr>
</thead>
<tbody>
<tr>
<td>63</td>
<td>62 downto 52</td>
<td>51 downto 0</td>
</tr>
</tbody>
</table>

```
log(input) = \log_2(N) + \log_2(2^{exp}) + mantissa
```

**INPUT CASE DETECTION**
- \(\log(\text{Negative number}) = \text{nan}\)
- \(\log(\text{Nan}) = \text{nan}\)
- \(\log(\text{Inf}) = \text{Inf}\)
- \(\log(-\text{Inf}) = \text{nan}\)
LOGARITHM APPROXIMATION UNIT (LAU) ARCHITECTURE

CREATE THE EXPLUT INDEX

<table>
<thead>
<tr>
<th>Decimal value</th>
<th>Exponent</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-1023</td>
</tr>
<tr>
<td>1</td>
<td>-1022</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>1022</td>
<td>-1</td>
</tr>
<tr>
<td>1023</td>
<td>0</td>
</tr>
<tr>
<td>1024</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>2046</td>
<td>1023</td>
</tr>
</tbody>
</table>

CASE DETECT

EXP LUT

FP VAL

MULT

ADD

P R

INPUT

Sign Exponent Mantissa

63 62 downto 52 51 downto 0

2046

SUB

1 0

log(2)

log(input)

1 0
LOGARITHM APPROXIMATION UNIT (LAU) ARCHITECTURE

CREATE THE EXPLUT INDEX

<table>
<thead>
<tr>
<th>Decimal value</th>
<th>Exponent</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-1023</td>
</tr>
<tr>
<td>1</td>
<td>-1022</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>1022</td>
<td>0</td>
</tr>
<tr>
<td>1023</td>
<td>1</td>
</tr>
<tr>
<td>1024</td>
<td>...</td>
</tr>
<tr>
<td>2046</td>
<td>1023</td>
</tr>
</tbody>
</table>

CASE DETECT

EXP LUT

2046

SUB

1 0

1 0

ADD

FP VAL

MANTISSA

MANTISSA

51 downto 0

63 62 downto 52
LOGARITHM APPROXIMATION UNIT (LAU) ARCHITECTURE

**Create the Explut Index**

<table>
<thead>
<tr>
<th>Decimal value</th>
<th>Exponent</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-1023</td>
</tr>
<tr>
<td>1</td>
<td>-1022</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>1022</td>
<td>0</td>
</tr>
<tr>
<td>1023</td>
<td>1</td>
</tr>
<tr>
<td>1024</td>
<td>...</td>
</tr>
<tr>
<td>2046</td>
<td>1023</td>
</tr>
</tbody>
</table>

**Input**

- **Sign**: 63
- **Exponent**: 62 downto 52
- **Mantissa**: 51 downto 0

**Logarithm Approximation**

1. **Case Detect**
2. **EXP LUT**
3. **FP VAL**
4. **MULT**
5. **ADD**
6. **EXP LUT**
7. **PRE**
8. **MAN LUT**
9. **SUB**
10. **FP VAL**
11. **LOG(2)**

**Logarithm Calculation**

- **log(input)**
- **log(2)**
LOGARITHM APPROXIMATION UNIT (LAU) ARCHITECTURE

CREATE THE EXPLUT INDEX

Decimal value

<table>
<thead>
<tr>
<th>Exponent</th>
<th>FP VAL</th>
<th>EXP LUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1023</td>
<td>-1022</td>
<td>-1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
<tr>
<td>1022</td>
<td>1023</td>
<td>1024</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
<tr>
<td>2046</td>
<td>1023</td>
<td>1023</td>
</tr>
</tbody>
</table>

1023 - (X - 1023) = 2046 - X

Input

log(input)

Sign  Exponent  Mantissa

63  62 downto 52  51 downto 0
LOGARITHM APPROXIMATION UNIT (LAU) ARCHITECTURE

FLOATING-POINT VALUE

Single-precision values

Single-precision MULT and ADD

For single-precision inputs
EXPLUT contains 128 entries

to construct a
single-precision value

For double-precision inputs
EXPLUT contains 1024 entries

to construct a
double-precision value

CASE DETECT

EXP LUT

SUB

1 0

FP VAL

log(2)

MAN LUT

MULT

ADD

P R

2046

input

log(input)
PERFORMANCE EVALUATION

Accuracy Versus Hardware resources

Average Error ($x10^3$) vs Resources (Number of 18Kb block rams)
PERFORMANCE EVALUATION

Accuracy Versus Hardware resources

Log-Likelihood score deviation

<table>
<thead>
<tr>
<th>Dataset (Organisms)</th>
<th>DP-GNU</th>
<th>DP-ICSILog</th>
</tr>
</thead>
<tbody>
<tr>
<td>150</td>
<td>-39606.3</td>
<td>-39606.6</td>
</tr>
<tr>
<td>218</td>
<td>-134173.8</td>
<td>-134167.5</td>
</tr>
<tr>
<td>140 (Prot)</td>
<td>-124777.2</td>
<td>-124780.1</td>
</tr>
</tbody>
</table>

6 block rams = 4096 LUT entries
PERFORMANCE EVALUATION

VIRTEX 5 SX95T for mapping and verification
XILINX ISE 10.1 and CHIPSCOPE Pro Analyzer

F. de Dinechin, C. Klein, B. Pasca,
PERFORMANCE EVALUATION

Resource Utilization and Performance: Single Precision

- **Slice Registers**
  - SP-FPLog: 1000
  - SP-LAU: 900

- **Slice LUTs**
  - SP-FPLog: 800
  - SP-LAU: 700

- **Occupied Slices**
  - SP-FPLog: 400
  - SP-LAU: 300
Resource Utilization and Performance: Single Precision

- BRAMs 18k
- BRAMs 36k
- DSP48Es

Legend:
- SP-FPLog
- SP-LAU
Resource Utilization and Performance: Single Precision

<table>
<thead>
<tr>
<th></th>
<th>SP-FPLog</th>
<th>SP-LAU</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>BRAMs 18k</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>BRAMs 36k</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>DSP48Es</strong></td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td><strong>Clock Latency</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Max Frequency</strong></td>
<td>244.7</td>
<td>353.5</td>
</tr>
</tbody>
</table>
PERFORMANCE EVALUATION

Resource Utilization and Performance: Double Precision
Resource Utilization and Performance: Double Precision
Resource Utilization and Performance: Double Precision

<table>
<thead>
<tr>
<th>Resource</th>
<th>FPLLog</th>
<th>LAU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clock Latency</td>
<td>34</td>
<td>22</td>
</tr>
<tr>
<td>Max Frequency</td>
<td>192.3</td>
<td>320.6</td>
</tr>
</tbody>
</table>
Resource Utilization and Performance: Double Precision

DP-FPLog with same accuracy as DP-LAU
Resource Utilization and Performance: Double Precision

DP-FPLog with same accuracy as DP-LAU
Resource Utilization and Performance: Double Precision

DP-FPLog with same accuracy as DP-LAU

<table>
<thead>
<tr>
<th></th>
<th>FPLLog</th>
<th>LAU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clock Latency</td>
<td>20</td>
<td>22</td>
</tr>
<tr>
<td>Max Frequency</td>
<td>239.6</td>
<td>320.6</td>
</tr>
</tbody>
</table>
PERFORMANCE EVALUATION

Performance:
LAU vs SP/DP-ICSILog vs GNU Log vs MKL Log

Intel Core2 DUO
T9600 @ 2.8GHz
6MB L2 Cache

SP-LAU VS
GNU-LOG : 11X
MKL-LOG : 1.6X

DP-LAU VS
GNU-LOG: 18X
MKL-LOG: 2.5X
CONCLUSION and FUTURE WORK

AVAILABILITY

DP-ICSILog C Implementation and SP/DP LAU FPGA core for Virtex4 and Virtex5 FPGAs

http://wwwkrammer.in.tum.de/exelixis/nikos/ipcores.html

Or

OpenCores.org: Project name: fp_log

http://www.opencores.org/project,fp_log
Related Projects

Implementation of a UDP/IP core for Virtex 5 FPGAs (optimized for PC-FPGA communication)

http://wwwkrammer.in.tum.de/exelixis/nikos/ipcores.html

Or

OpenCores.org: Project name: udp_ip__core

http://www.opencores.org/project,udp_ip__core

Future Work

Implementation of a resource-efficient exponential function

Integration of the LOG and EXP cores into the general Phylogenetic Architecture