FPGA and Dwarfs

Jens Hahne, Hongrui Deng

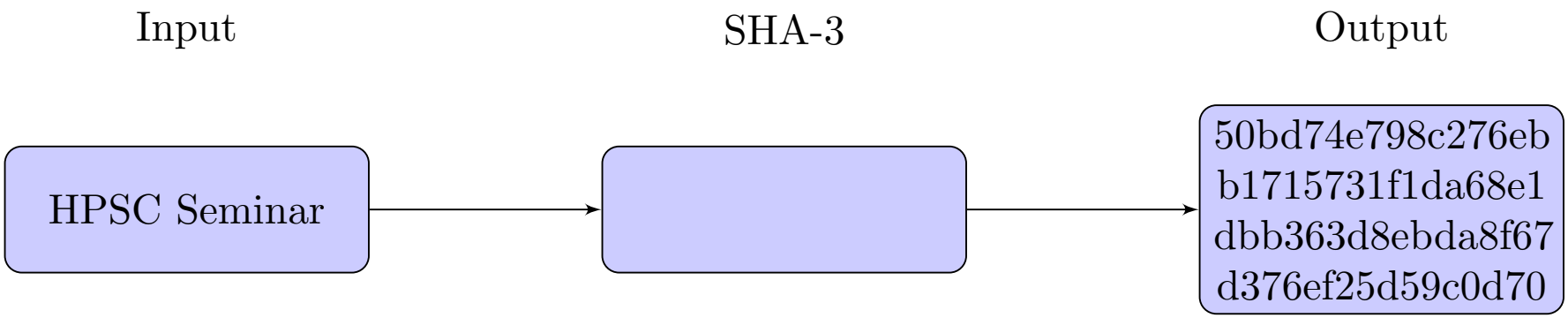
High-Performance and Automatic Computing Group
in RWTH Aachen

January 29, 2015

Overview

Secure Hash Algorithm-3 (SHA-3)

- Cryptographic hash algorithm
- Applications:
  - Authentication system
  - Digital signature algorithms



Main message

- Main message:
  - High speed implementation of SHA-3.
  - Combine all steps of SHA-3 logically.

- Why FPGA?
  - FPGA solutions provide high speed and real time results.
  - SHA-3 consist of simple Bit operation.

Secure Hash Algorithm-3 (SHA-3)

- SHA-3 hash function consists of three steps:
  - Initialization: Initialization of state matrix A with all zeros
  - Absorbing: -XOR each r-bit wide block with A
    -Perform 24 rounds of compression function
  - Squeezing: Truncate the state matrix to output value

- A is distributed upon twenty five 64-bit words
- A[0,0]=[1599:1536], A[1,0]=[1535:1472],....,A[4,4]=[63,0]

SHA-3 Algorithm compression function

Θ Step: $(0 \leq x, y \leq 4)$

- $C[x] = A[x,0] \oplus A[x,1] \oplus A[x,2] \oplus A[x,3] \oplus A[x,4];$ (1)
- $D[x] = C[x-1] \oplus ROT(C[x+1], 1);$ (2)
- $A[x,y] = A[x,y] \oplus D[x]$ (3)

ρ and π Step: $(0 \leq x, y \leq 4)$

- $B[y, 2x+3y] = ROT(A[x,y], r[x,y]);$ (4)

χ Step: $(0 \leq x, y \leq 4)$

- $F[x,y] = B[x,y] \oplus ((\neg B[x+1,y]) \wedge B[x+2,y]);$ (5)

ι Step: $(0 \leq x, y \leq 4)$

- $F'[0,0] = F[0,0] \oplus RC;$ (6)

Combine (1) and (2)

- Combine (1) and (2) into a single equation.
- $$C[x] = A[x,0] \oplus A[x,1] \oplus A[x,2] \oplus A[x,3] \oplus A[x,4]; \quad (1)$$
- $$D[x] = C[x-1] \oplus ROT(C[x+1],1); \quad (2)$$

$$\begin{aligned} D[x] =& \{A[x-1,0] \oplus A[x-1,1] \oplus A[x-1,2] \oplus A[x-1,3] \\ & \oplus A[x-1,4]\} \oplus \{ROT(A[x+1,0],1) \\ & \oplus ROT(A[x+1,1],1) \oplus (A[x+1,2],1) \\ & \oplus ROT(A[x+1,3],1) \oplus ROT(A[x+1,4],1)\}; \\ & (0 \leq x \leq 4) \end{aligned} \quad (7)$$

Combine (3) and (7)

- Combine (3) and (7)
- $$A[x, y] = A[x, y] \oplus D[x] \tag{3}$$
- $\Rightarrow$ 25 equations from A[0,0] to A[4,4]

$$\begin{aligned} A[x, y] =& \{A[x, y]\} \oplus \{A[x-1, 0] \oplus A[x-1, 1] \oplus A[x-1, 2] \\ & \oplus A[x-1, 3] \oplus A[x-1, 4]\} \oplus \{ROT(A[x+1, 0], 1) \\ & \oplus ROT(A[x+1, 1], 1) \oplus ROT(A[x+1, 2], 1) \\ & \oplus ROT(A[x+1, 3], 1) \oplus ROT(A[x+1, 4], 1)\}; \\ & (0 \leq x, y \leq 4) \end{aligned} \tag{8}$$

Combine (4) and (8)

- Combine (4) and (8)
- $$B[y, 2x + 3y] = ROT(A[x, y], r[x, y]); \quad (4)$$
- $\Rightarrow$ 25 equations from B[0,0] to B[4,4]

$$\begin{aligned} B[y, 2x + 3y] =& ROT(\{A[x, y]\}, r[x, y]) \oplus \{ROT(A[x - 1, 0], r[x, y]) \\ & \oplus ROT(A[x - 1, 1], r[x, y]) \oplus ROT(A[x - 1, 2], r[x, y]) \\ & \oplus ROT(A[x - 1, 3], r[x, y]) \oplus ROT(A[x - 1, 3], r[x, y])\} \\ & \oplus \{ROT(ROT(A[x + 1, 0], 1), r[x, y]) \\ & \oplus ROT(ROT(A[x + 1, 1], 1), r[x, y]) \\ & \oplus ROT(ROT(A[x + 1, 2], 1), r[x, y]) \\ & \oplus ROT(ROT(A[x + 1, 3], 1), r[x, y]) \\ & \oplus ROT(ROT(A[x + 1, 4], 1), r[x, y])\}; \\ & (0 \leq x, y \leq 4) \end{aligned} \quad (9)$$

Combine (5) and (9)

- Combine equation (5) and (9)
- Put B[x,y], B[x+1,y], B[x+2,y] into (5)
- Perform ROT manually for each equation
- $$F[x,y] = B[x,y] \oplus ((\neg B[x+1,y]) \wedge B[x+2,y]); \qquad (5)$$
- $\Rightarrow$ 25 equations from F[0,0] to F[4,4]

Combine (5) and (9)

$$
\begin{aligned}
F[0,0] =& \{A[0,0]\} \oplus \{\{A[4,0]\} \oplus \{A[4,1]\} \oplus \{A[4,2]\} \oplus \{A[4,3]\} \oplus \{A[4,4]\}\} \\
& \oplus \{\{A[1,0][62:0], A[1,0][63]\} \oplus \{A[1,1][62:0]A[1,1][63]\} \\
& \oplus \{A[1,2][62:0], A[1,2][63]\} \oplus \{A[1,3][62:0], A[1,3][63]\} \\
& \oplus \{A[1,4][62:0], A[1,4][63]\}\} \oplus \{\neg(\{A[1,1][19:0], A[1,1][63:20]\} \\
& \oplus \{\{A[0,0][19:0], A[0,0][63:20]\} \oplus \{A[0,1][19:0], A[0,1][63:20]\} \\
& \oplus \{A[0,2][19:0], A[0,2][63:20]\} \oplus \{A[0,3][19:0], A[0,3][63:20]\} \\
& \oplus \{A[0,4][19:0], A[0,4][63:20]\}\} \oplus \{\{A[2,0][18:0], A[2,0][63:19]\} \\
& \oplus \{A[2,1][18:0], A[2,1][63:19] \oplus \{A[2,2][18:0], A[2,2][63,19] \\
& \oplus \{A[2,3][18:0], A[2,3][63,19]\} \oplus \{A[2,4][18,0], A[2,4][63,19]\}\}) \\
& \wedge (\{A[2,2][20:0], A[2,2][63:21]\} \oplus \{\{A[1,0][20:0], A[1,0][63:21]\} \\
& \oplus \{A[1,1][20:0], A[1,1][63:21]\} \oplus \{A[1,2][20:0], A[1,2][63:21]\} \\
& \oplus \{A[1,3][20:0], A[1,3][63:21]\} \oplus \{A[1,4][20:0], A[1,4][63:21]\}\} \\
& \oplus \{\{A[3,0][19:0], A[3,0][63:20]\} \oplus \{A[3,1][19:0], A[3,1][63:20]\} \\
& \oplus \{A[3,2][19:0], A[3,2][63:20]\} \oplus \{A[3,3][19:0], A[3,3][63:20]\} \\
& \oplus \{A[3,4][19:0], A[3,4][63:20]\}\})\}; \\
& (0 \leq x, y \leq 4)
\end{aligned} \tag{10}
$$

Combine (5) and (9)

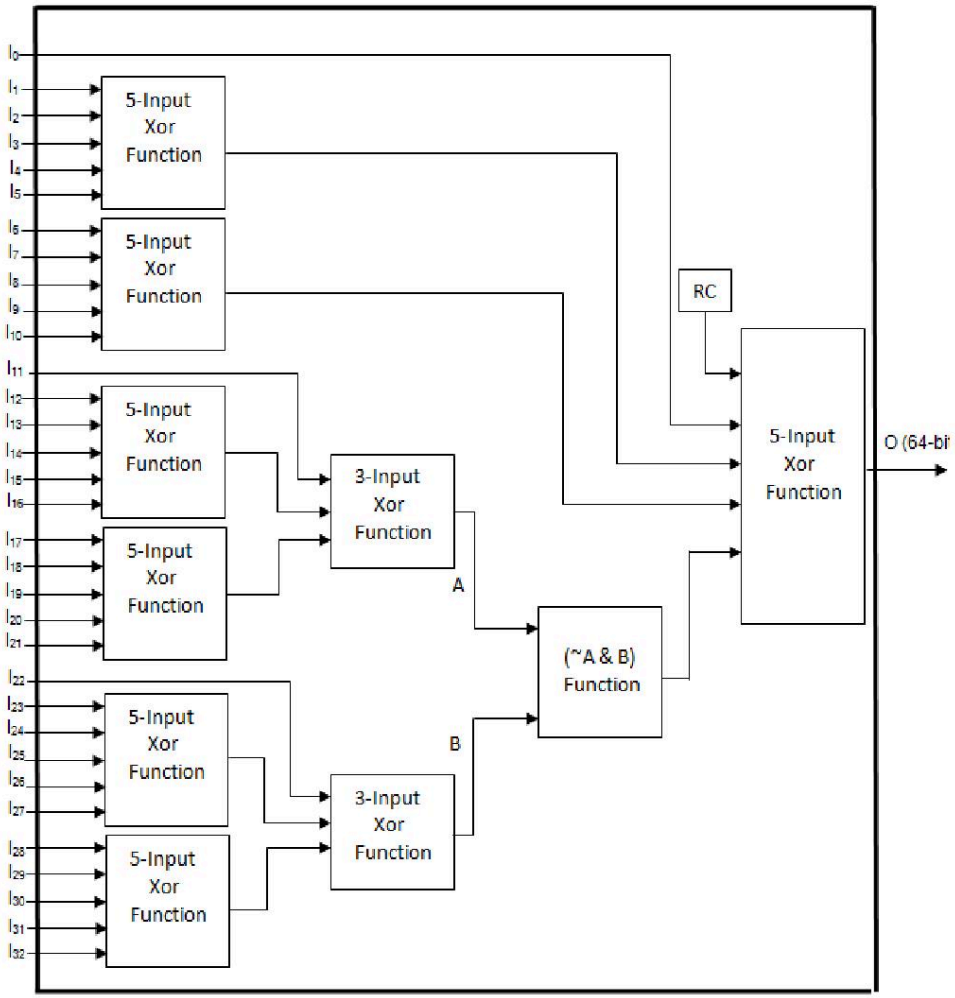
$$
\begin{aligned}
F[4,4] =& \{A[1,4][61:0], A[1,4][63:62]\} \oplus \{\{A[0,0][61:0], A[0,0][63:62]\} \\
&\oplus A[0,1][61:0], A[0,1][63:62]\} \oplus \{A[0,2][61:0], A[0,2][63:62]\} \\
&\oplus \{A[0,3][61:0], A[0,3][63:62] \oplus \{A[0,4][61:0], A[0,4][63:62]\}\} \\
&\oplus \{\{A[2,0][60:0], A[2,0][63:61]\} \oplus \{A[2,1][60:0] A[2,1][63:61]\} \\
&\oplus \{A[2,2][60:0], A[2,2][63:61]\} \oplus \{A[2,3][60:0], A[2,3][63:61]\} \\
&\oplus \{A[2,4][60:0], A[2,4][63:61]\}\} \oplus \{\neg(\{A[2,0][1:0], A[2,0][63:02]\} \\
&\oplus \{\{A[1,0][1:0], A[1,0][63:02]\} \oplus \{A[1,1][1:0], A[1,1][63:02]\} \\
&\oplus \{A[1,2][1:0], A[1,2][63:02]\} \oplus \{A[1,3][1:0], A[1,3][63:02]\} \\
&\oplus \{A[1,4][1:0], A[1,4][63:02]\}\} \oplus \{\{A[3,0][0], A[3,0][63:01]\} \\
&\oplus \{A[3,1][0], A[3,1][63:01] \oplus \{A[3,2][0], A[3,2][63,01] \\
&\oplus \{A[3,3][0], A[3,3][63,01]\} \oplus \{A[3,4][0], A[3,4][63,01]\}\}) \\
&\wedge (\{A[3,1][8:0], A[3,1][63:9]\} \oplus \{\{A[2,0][8:0], A[2,0][63:9]\} \\
&\oplus \{A[2,1][8:0], A[2,1][63:9]\} \oplus \{A[2,2][8:0], A[2,2][63:9]\} \\
&\oplus \{A[2,3][8:0], A[2,3][63:9]\} \oplus \{A[2,4][8:0], A[2,4][63:9]\}\} \\
&\oplus \{\{A[4,0][7:0], A[4,0][63:8]\} \oplus \{A[4,1][7:0], A[4,1][63:8]\} \\
&\oplus \{A[4,2][7:0], A[4,2][63:8]\} \oplus \{A[4,3][7:0], A[4,3][63:8]\} \\
&\oplus \{A[4,4][7:0], A[4,4][63:8]\}\})\}; \qquad (0 \le x, y \le 4)
\end{aligned} \tag{11}
$$

General equation

- Eq. (10) and eq. (11) have the same structure
- General equation represent F'[0,0] to F[4,4]
- Inputs I_0 to I_{32} (64 bit words) are different for every equation
- RC just updates F[0,0], zero for all other F[x,y]

$$
\begin{aligned}
F[x,y] = & RC \oplus \{I_0\} \oplus \{\{I_1\} \oplus \{I_2\} \oplus \{I_3\} \oplus \{I_4\} \oplus \{I_5\}\} \\
& \oplus \{\{I_6\} \oplus \{I_7\} \oplus \{I_8\} \oplus \{I_9\} \oplus \{I_{10}\}\} \oplus \{\neg(\{I_{11}\} \\
& \oplus \{\{I_{12}\} \oplus \{I_{13}\} \oplus \{I_{14}\} \oplus \{I_{15}\} \oplus \{I_{16}\}\} \oplus \{\{I_{17}\} \\
& \oplus \{I_{18}\} \oplus \{I_{19}\} \oplus \{I_{20}\} \oplus \{I_{21}\}\}) \wedge (\{I_{22}\} \oplus \{\{I_{23}\} \\
& \oplus \{I_{24}\} \oplus \{I_{25}\} \oplus \{I_{26}\} \oplus \{I_{27}\}\} \oplus \{\{I_{28}\} \oplus \{I_{29}\} \\
& \oplus \{I_{30}\} \oplus \{I_{31}\} \oplus \{I_{32}\}\})\};
\end{aligned}
\tag{12}
$$

Architecture



- 25 instances
- F'[0,0] to F[4,4]
- Each compression function requires a single clock cycle
- 24 clock cycles for complete compression function

[1]Efficient High Speed Implementation of Secure Hash Algorithm-3 on Virtex-5 FPGA

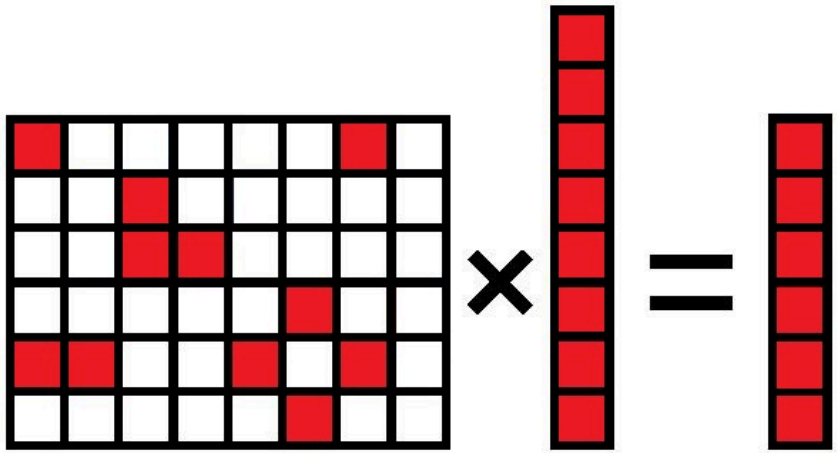
Comparison FPGA/CPU/GPU

Platform	Throughput	Output	Ref.
Virtex 5	17.132 (GB/s)	256-bit	[1]
Intel Core 2 Quad Q6600 64 bit	64.2 (MB/s)	512-bit	[3]
Intel Core 2 Quad Q6600 32 bit	22.6 (MB/s)	512-bit	[3]
Intel Core i5 2450M 64-bit	849 (MB/s)	512-bit	[3]
NVIDIA GTX 295 GPU	250 (MB/s)	512-bit	[4]

- Output length affects the throughput.

Sparse Matrix-Vector Multiplication

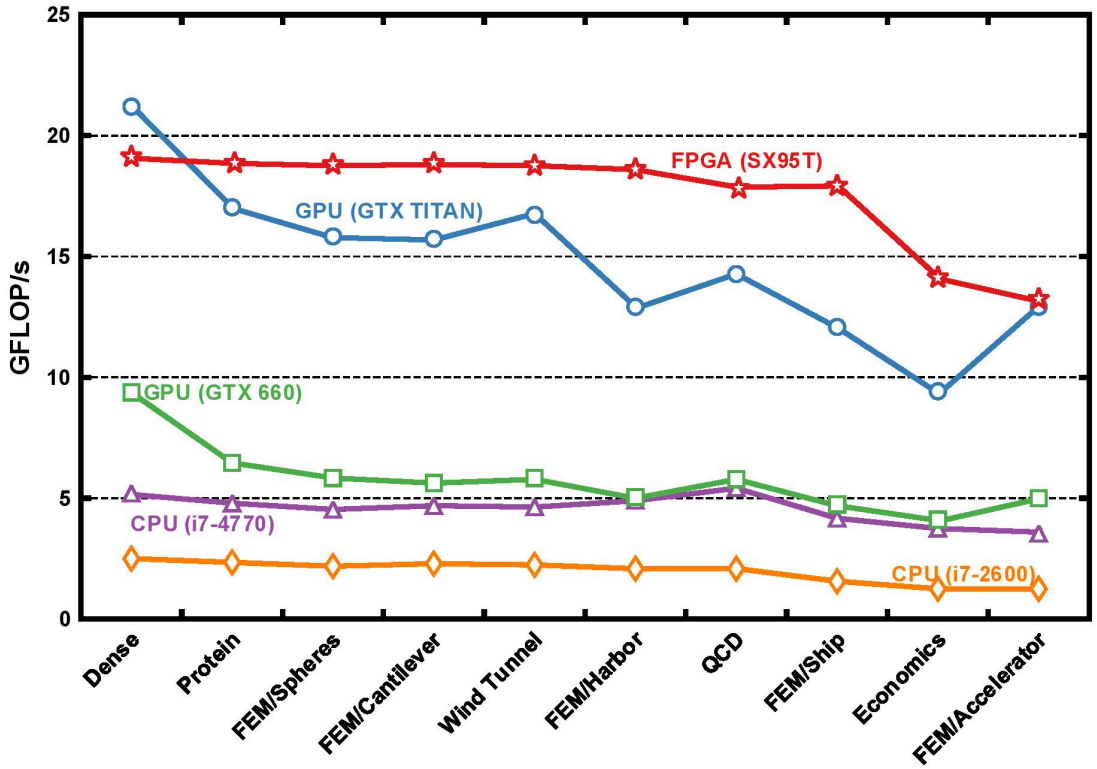
- Dwarf: Sparse Linear Algebra
- Sparse Matrix-Vector Multiplication (SpMxV)



Main message

- Description of a FPGA-based SpMxV kernel.
- Architecture for FPGA with high computational efficiency
- High computational efficiency leads to energy-efficient.

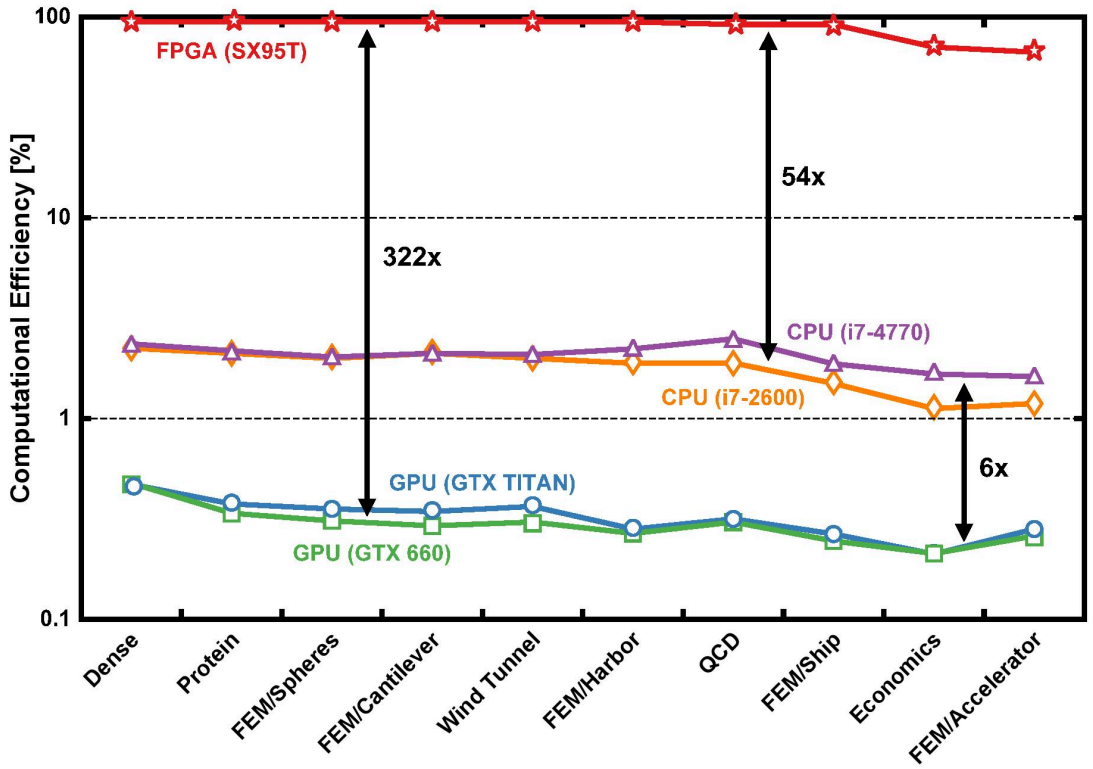
Computational performance



	Rows	Columns	Nonzeros	Density
Dense	2000	2000	4000000	100%
Protein	36417	36417	4344765	0.3276%
WindTunnel	217918	217918	11524432	0.0243%
Economics	206500	206500	1273389	0.0030%

[2]A Scalable Sparse Matrix-Vector Multiplication Kernel for Energy-Efficient Sparse-Blas on FPGAs

Computational efficiency



	Rows	Columns	Nonzeros	Density
Dense	2000	2000	4000000	100%
Protein	36417	36417	4344765	0.3276%
WindTunnel	217918	217918	11524432	0.0243%
Economics	206500	206500	1273389	0.0030%

[2]A Scalable Sparse Matrix-Vector Multiplication Kernel for Energy-Efficient Sparse-Blas on FPGAs

Power consumption

Platform	average power consumption	power efficiencies
Virtex-5 SX95T	5.1 W	3460 MLFOP/s/W
i7-2600	77.2 W	26 MLFOP/s/W
i7-4770	66.3 W	26 MLFOP/s/W
GTX 660	99 W	58 MLFOP/s/W
GTX Titan	163 W	91 MLFOP/s/W

[2]A Scalable Sparse Matrix-Vector Multiplication Kernel for Energy-Efficient Sparse-Blas on FPGAs

Dynamic Programming

Dwarf: Dynamic Programming

Problem: High Speed Biological Sequence Analysis with Hidden Markov Models on FPGA

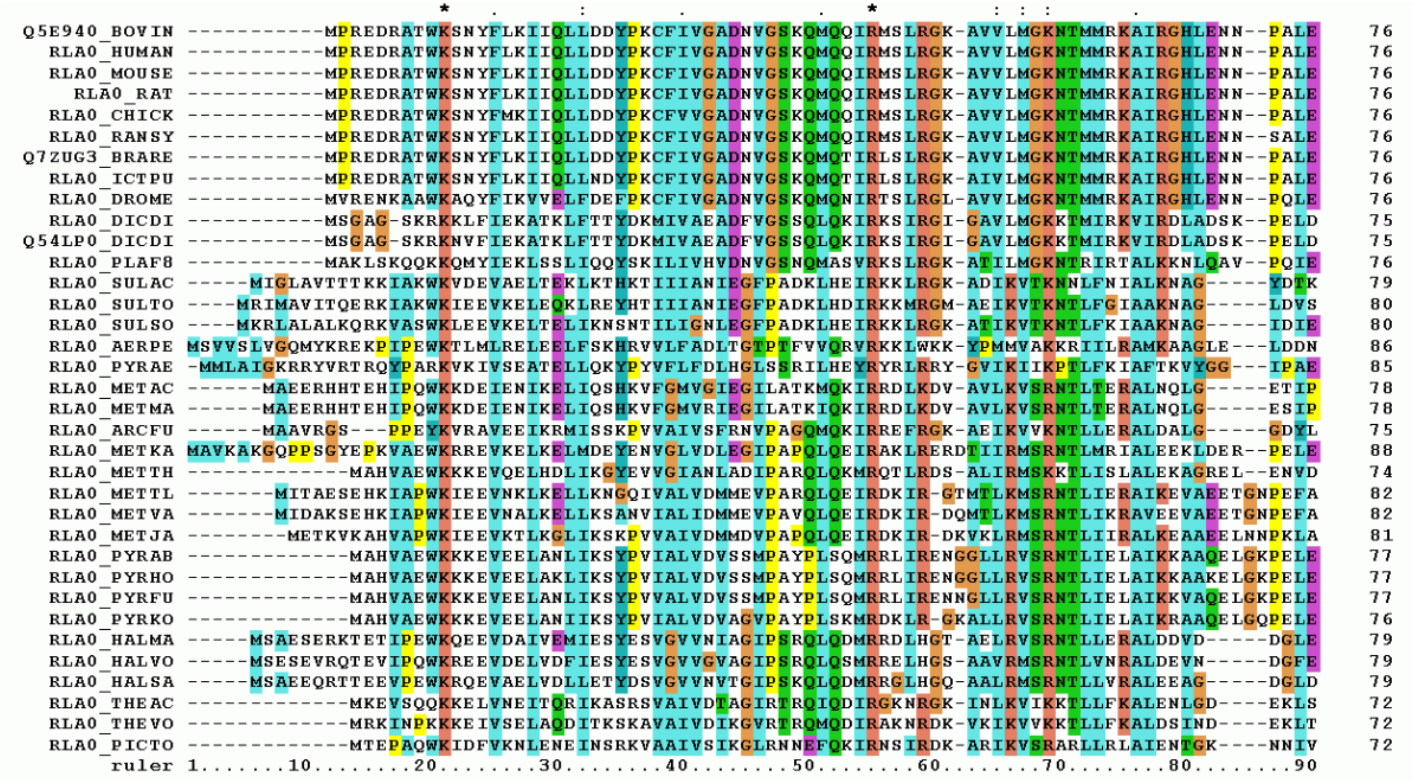


Figure: A protein multiple sequence alignment

Speciality of Implementation on FPGA

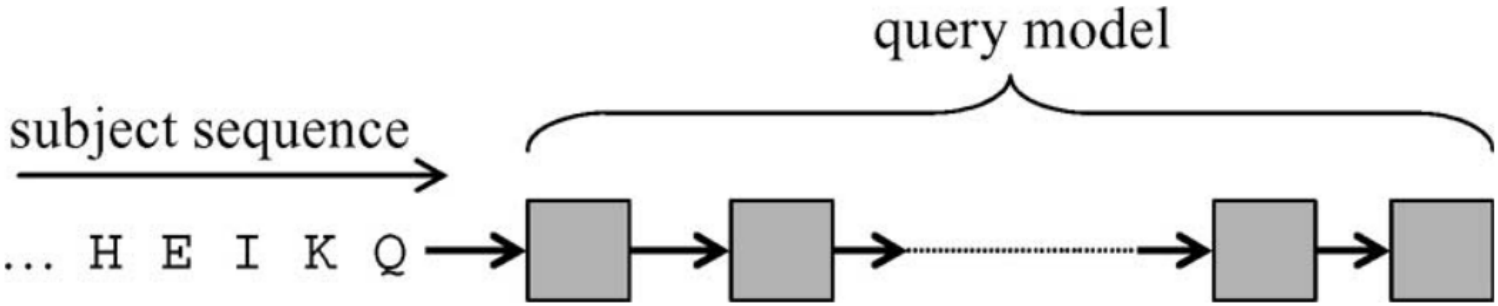


Figure: Sequence comparison on a linear processor array.[6]

Length of subject sequence: **M**
Length of query HMM: **K**

Computation steps: **M+K-1**,
instead of **M** $\times$ **K** on a sequential processor.

	X	X	.	.	.	X
bat	A	G	-	-	-	C
rat	A	-	A	G	-	C
cat	A	G	-	A	A	-
gnat	-	-	A	A	A	C
goat	A	G	-	-	-	C

Performance and productivity

Query HMM length	**Number of Processing Passes**	**Performance (in Giga *CUPS**)**	**Speedup**
24	1	1.510	62.9
72	1	4.692	195.5
112	2	3.718	154.9
236	4	3.954	164.7

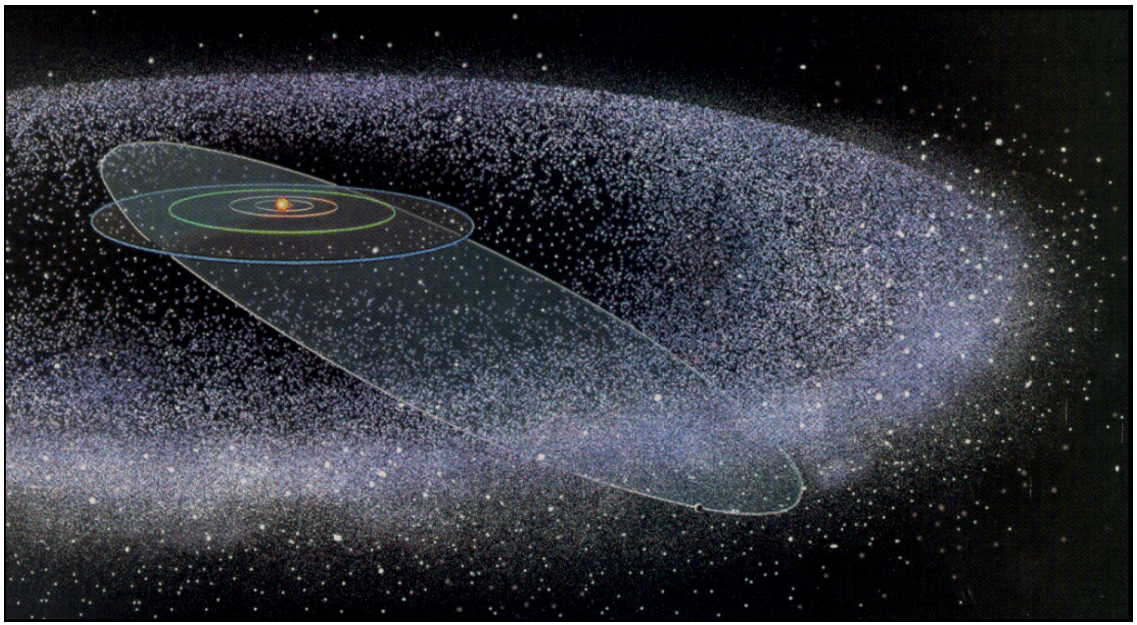
*. *cell updates per second(CUPS)*

Table: The Speedup compared to a Pentium 4 3GHZ is reported.[3]

N-Body Problem

Dwarf: N-Body Problem

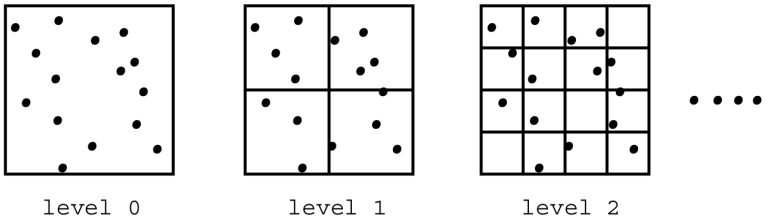
- given initial positions, masses, and velocities of bodies
- simulate the evolution of N celestial bodies



Problem: PP(Particle-Particle): O(N^2),
Fast Multipole Method(FMM): O(N).

Speciality of Implementation on FPGA

Basic idea of FMM...



Speciality on FPGA :

- parallel computing in hardware logic
- high computational efficiency

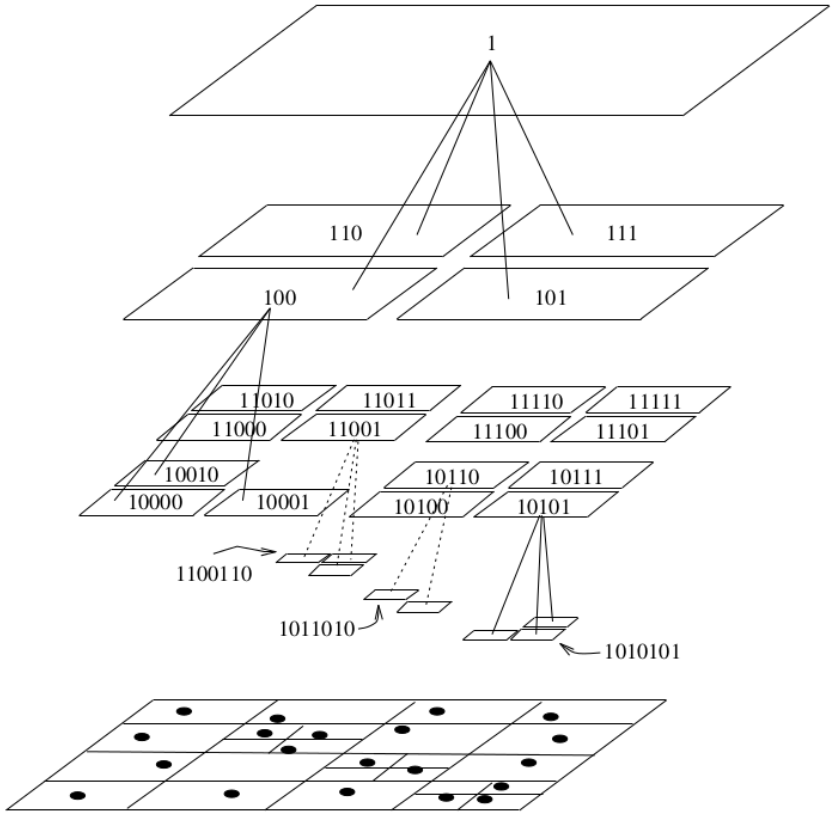


Figure: A quad-tree shown along with the binary key coordinates of the nodes.[8]

Performance and productivity

		p = 4		p = 5	
L2L and L2P		***Separated***	***Merged***	***Separated***	***Merged***
FPGA	FPGA chip	xc5vsx50t-1ff1136			
	Maxinum Frequency	160.46 MHz	157.31 MHz	144.26 MHz	133.87 MHz
	Flip-Flops	13978 (43%)	5665 (17%)	18393 (56%)	7433 (22%)
	LUTs	10092 (31%)	3471 (10%)	12828 (39%)	4412 (13%)
	DSP48e	104 (36%)	104 (36%)	144 (50%)	144 (50%)
	Running Time	24.928ns	25.428ns	34.66ns	37.35ns
CPU	CPU Model	Intel Pentium Dual E2200 @2.2GHz			
	Running Time	1.48us		2.1us	

Figure: Performance and resource utilization comparison[7]

Summary

- FPGA is an integrated circuit designed to be configured by a costumer.
- Advantages:
  - Reprogrammability
  - Parallel data processing
  - Flexibility
  - Relatively small price/unit
  - Allow regularly updating to state-of-art technology
- Good for:
  - Prototypes
  - Real time applications
  - High-speed image/video processing

Summary

- Idea of FPGAs not too difficult, but implementation seems challenging.
- Needs a hardware description language (Verilog/VHDL) to configure.
- For an implementation as $x := \alpha x + y$ more studies are needed.

The End

Thank you for your attention!

References

Muaffar Rao, Thomas Newe, Ian Grout (2014)

[1] Efficient High Speed Implementation of Secure Hash Algorithm-3 on Virtex-5 FPGA

Richard Dorrance, Fengbo Ren, Dejan Markovic (2014)

[2] A Scalable Sparse Matrix-Vector Multiplication Kernel for Energy-Efficient Sparse-Blas on FPGAs

Aisha Malikl, Arshad Aziz, Dur- e-Shahwar Kunde, Moiz Akhter (2013)

[3] Software Implementation of Standard Hash Algorithm (SHA-3) Keccak on Intel Core-i5 and Cavium Networks Octeon Plus embedded platform

Fbio Dacncio Pereira, Edward David Moreno Ordonez, Ivan Daun Sakai, Allan Mariano de Souza (2013)

[4] Exploiting Parallelism on Keccak: FPGA and GPU Comparison

Johathan Rose, Abbas El Gamal and Alberto Sangiovanni (1993)

[5] Architecture of Field-Programmable Gate Arrays

Timothy F. Oliver, Bertil Schmidt, Yanto Jakop and Douglas L. Maskell (2012)

[6] High Speed Biological Sequence Analysis With Hidden Markov Models on Reconfigurable Platforms

References

Zhe Zheng, Youngxin Zhu, Xu Wang, Zhiqiang Que, Tian Huang, Xiaojing Yin, Hui Wang, Guoguang Rong and Meikang Qiu (2010)

[7] Revealing Feasibility of FMM on ASIC: Efficient Implementation of N-Body Problem on FPGA

Michael S. Warren, John K. Salmon (1993)

[8] A Parallel Hashed Oct-Tree N-Body Algorithm

Credits

- Logic Block Architecture (Hongrui)
- Comparison to CPU (Jens)
- Computational Logic (Jens)
- Sparse Linear Algebra (Jens)
- Dynamic Programming (Hongrui)
- N-Body Problem (Hongrui)
- Summary (Hongrui, Jens)