#### FPGA and Dwarfs

#### Jens Hahne, Hongrui Deng

# High-Performance and Automatic Computing Group in RWTH Aachen

#### January 29, 2015

1 Combinational Logic: SHA-3 Algorithm

- 2 Sparse Linear Algebra: Sparse Matrix-Vector Multiplication
- 3 Dynamic Programming:Biological Sequence Analysis
- 4 N-Body Problem: Fast Multipole Method



## Secure Hash Algorithm-3 (SHA-3)

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



#### • 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.

• 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

 $\Theta \text{ Step:} \qquad (0 \le x, y \le 4)$ 

• 
$$C[x] = A[x,0] \oplus A[x,1] \oplus A[x,2] \oplus A[x,3] \oplus A[x,4];$$
 (1)

• 
$$D[x] = C[x-1] \oplus ROT(C[x+1],1);$$
 (2)

• 
$$A[x,y] = A[x,y] \oplus D[x]$$
 (3)

 $\begin{array}{ll}
 \underline{\rho \text{ and } \pi \text{ Step:}} & (0 \le x, y \le 4) \\
 \bullet & B[y, 2x + 3y] = ROT(A[x, y], r[x, y]); \\
 \underline{\chi \text{ Step:}} & (0 \le x, y \le 4) \\
 \bullet & F[x, y] = B[x, y] \oplus ((\neg B[x + 1, y]) \land B[x + 2, y]); \\
 \underline{\iota \text{ Step:}} & (0 \le x, y \le 4) \\
 \bullet & F'[0, 0] = F[0, 0] \oplus RC; \\
 \end{array}$  (6)

# Combine (1) and (2)

• Combine (1) and (2) into a single equation.

• 
$$C[x] = A[x,0] \oplus A[x,1] \oplus A[x,2] \oplus A[x,3] \oplus A[x,4];$$
 (1)

• 
$$D[x] = C[x-1] \oplus ROT(C[x+1], 1);$$
 (2)

$$D[x] = \{A[x - 1, 0] \oplus A[x - 1, 1] \oplus A[x - 1, 2] \oplus A[x - 1, 3] \\ \oplus A[x - 1, 4]\} \oplus \{ROT(A[x + 1, 0], 1) \\ \oplus ROT(A[x + 1, 1], 1) \oplus (A[x + 1, 2], 1) \\ \oplus ROT(A[x + 1, 3], 1) \oplus ROT(A[x + 1, 4], 1)\}; \\ (0 \le x \le 4)$$
(7)

# Combine (3) and (7)

• Combine (3) and (7)

• 
$$A[x,y] = A[x,y] \oplus D[x]$$
 (3)

 $\bullet\,\Rightarrow\,25$  equations from A[0,0] to A[4,4]

$$A[x, y] = \{A[x, y]\} \oplus \{A[x - 1, 0] \oplus A[x - 1, 1] \oplus A[x - 1, 2] \\ \oplus A[x - 1, 3] \oplus A[x - 1, 4]\} \oplus \{ROT(A[x + 1, 0], 1) \\ \oplus ROT(A[x + 1, 1], 1) \oplus ROT(A[x + 1, 2], 1) \\ \oplus ROT(A[x + 1, 3], 1) \oplus ROT(A[x + 1, 4], 1)\}; \\ (0 \le x, y \le 4)$$
(8)

# Combine (4) and (8)

• Combine (4) and (8)

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

(4)

 $\bullet\,\Rightarrow\,25$  equations from B[0,0] to B[4,4]

$$B[y, 2x + 3y] = ROT(\{A[x, y]\}, r[x, y]) \oplus \{ROT(A[x - 1, 0], r[x, y]) \\ \oplus ROT(A[x - 1, 1], r[x, y]) \oplus ROT(A[x - 1, 2], r[x, y]) \\ \oplus ROT(A[x - 1, 3], r[x, y]) \oplus ROT(A[x - 1, 3], r[x, y]) \} \\ \oplus \{ROT(ROT(A[x + 1, 0], 1), r[x, y]) \\ \oplus ROT(ROT(A[x + 1, 1], 1), r[x, y]) \\ \oplus ROT(ROT(A[x + 1, 2], 1), r[x, y]) \\ \oplus ROT(ROT(A[x + 1, 3], 1), r[x, y]) \\ \oplus ROT(ROT(A[x + 1, 3], 1), r[x, y]) \\ \oplus ROT(ROT(A[x + 1, 4], 1), r[x, y]) \}; \\ (0 \le x, y \le 4)$$

- 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
- $F[x,y] = B[x,y] \oplus ((\neg B[x+1,y]) \land B[x+2,y]);$  (5)
- $\bullet\,\Rightarrow\,25$  equations from F[0,0] to F[4,4]

## Combine (5) and (9)

 $F[0,0] = \{A[0,0]\} \oplus \{\{A[4,0]\} \oplus \{A[4,1]\} \oplus \{A[4,2]\} \oplus \{A[4,3]\} \oplus \{A[4,4]\}\}$  $\oplus$  {{*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]}}  $\oplus$  {¬({*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]}});

$$(0\leq x,y\leq 4)$$

(10)

# Combine (5) and (9)

 $F[4,4] = \{A[1,4][61:0], A[1,4][63:62]\} \oplus \{\{A[0,0][61:0], A[0,0][63:62]\}\}$  $\oplus$  *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]}  $\oplus \{A[4,4][7:0], A[4,4][63:8]\}\}\}; (0 \le x, y \le 4)$ 

(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]

 $F[x, y] = RC \oplus \{l_0\} \oplus \{\{l_1\} \oplus \{l_2\} \oplus \{l_3\} \oplus \{l_4\} \oplus \{l_5\}\} \\ \oplus \{\{l_6\} \oplus \{l_7\} \oplus \{l_8\} \oplus \{l_9\} \oplus \{l_{10}\}\} \oplus \{\neg(\{l_{11}\} \oplus \{\{l_{12}\} \oplus \{l_{13}\} \oplus \{l_{14}\} \oplus \{l_{15}\} \oplus \{l_{16}\}\} \oplus \{\{l_{17}\} \oplus \{l_{18}\} \oplus \{l_{19}\} \oplus \{l_{20}\} \oplus \{l_{21}\}\}) \land (\{l_{22}\} \oplus \{\{l_{23}\} \oplus \{l_{24}\} \oplus \{l_{25}\} \oplus \{l_{26}\} \oplus \{l_{27}\}\} \oplus \{\{l_{28}\} \oplus \{l_{29}\} \oplus \{l_{30}\} \oplus \{l_{31}\} \oplus \{l_{32}\}\})\};$ 

#### 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

Jens Hahne, Hongrui Deng (RWTH)

HPSC Seminar

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.

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



- 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



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

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

<sup>[2]</sup>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

Q5E940_BOVIN	b	1 PREDRAT W	KSNYFLK	IQLLDDYP	KCFIV <mark>G</mark> ADNVG	SKOMOQ IRMS LRG	K-AVVLMGKNTMMF	KAIRGHLENNPALE	76
RLA0 HUMAN	b	<b>PREDRAT</b>	KSNYFLK	IQLLDDYP	KCFIV <mark>G</mark> A <mark>D</mark> NVG	SKOMOQIRMSLRG	K-AVVLMGKNTMMF	KAIRGHLENNPALE	76
RLA0 MOUSE	b	4 PREDRATW	KSNYFLK	IQLLDDYP	KCFIV <mark>GAD</mark> NVG	SKOMOQIRMSLRG	K - AVV LMGKNTMMF	KAIRGHLENNPALE	76
RLA0 RAT	b	4 PREDRATW	KSNYFLK	IQLLDDYP	KCFIV <mark>GAD</mark> NVG	SKOMOQIRMSLRG	K – AVV LM <mark>GKNTMM</mark> F	KAIRGHLENNPALE	76
RLA0 CHICK	b	<b>∕P</b> REDR <mark>A</mark> T₩	KSNYFMKJ	IQLLDDYP	KCFVV <mark>GAD</mark> NVG	SKOMOOTRMSLRG	K-AVVLMGKNTMMF	KAIRGHLENNPALE	76
RLA0 RANSY	b	1 PREDRATW	KSNYFLK	IQLLDDYP	KCFIV <mark>GAD</mark> NVG	SKOMOQIRMSLRG	K-AVVLM <mark>GKNT</mark> MMF	KAIRGHLENNSALE	76
Q7ZUG3 BRARE	b	<mark>4</mark> ₽REDR <mark>A</mark> TW	KSNYFLK	IQLLDDYP	KCFIV <mark>G</mark> ADNVG	SKOMOT IRLS LRG	K-AVVLMGKNTMMF	KAIRGHLENNPALE	76
RLA0 ICTPU	b	4 PREDRATW	KSNYFLK	IQLLNDYPE	KCFIV <mark>GAD</mark> NVG	SKOMOT IRLS LRG	K-AIVLMGKNTMMF	KAIRGHLENNPALE	76
RLA0 DROME	b	<b>AVRENKAA</b> W	KAQYFIK	VELFDEFP	KCFIV <mark>GAD</mark> NVG	SKOMON IRTS LRG	L-AVVLMGKNTMMF	KAIRGHLENNPOLE	76
RLA0 DICDI	b	ISGAG-SKR	KKLFIEK	TKLFTTYDE	KMIVAE ADFVG	S SOLOK IRKS IRG	I-GAVLMGKKTMIF	KVIRDLADSKPELD	75
Q54LP0 DICDI	b	45 <mark>6 AG</mark> – SKR	KNVFIEK	TKLFTTYDE	KMIVAE A <mark>D</mark> FVG	S SOLOK IRKS IRG	I-GAVLMGKKTMIF	KVIRDLADSKPELD	75
RLA0 PLAF8		ARLSKOOK	ROMYTERI	SSLIQOYSI	KILIVHV <mark>D</mark> NVG	S NOMAS VRKS LRG	K-ATILMGENTRIF	TALKKNLOAVPOIE	76
RLA0 SULAC	MIGLAVI	TTKKIAKW	KVDEVAEL	TEKLETHE	TIIIAN I <mark>EG</mark> FP	ADKLHE IRKK LRG	K-ADIKVTENNLEN	IALKNAGYDTK	79
RLA0 SULTO	MRIMAVIT	OERK IAKW	KIEEVKEI	EOKLREYHT	<b>TIIIANIEGFP</b>	ADKLHDIRKKMRG	M-AEIKVTKNTLFG	IAAKNAGLDVS	80
RLA0 SULSO	MKRLALAI	LKORKVASW	KLEEVKEI	TELIKNSNI	TILI <mark>G</mark> NL <mark>EG</mark> FP	ADKLHE IRKKLRG	K-ATIKVTKNTLFR	IAAKNAGIDIE	80
RLA0 AERPE	MSVVSLVGOMY	REKPIPEW	KTLMLREI	EELFSKHRY	VVLFADLTGTP	TFVVORVRKKLWE	K-YPMMVAKKRIII	RAMKAAGLE LDDN	86
RLA0 PYRAE	-MMLAIGKRRY	RT RQ YP AR	KVKIVSE	TELLQKYP	YVFLFDLHGLS	SRILHE YRYRLRF	Y-GVIKIIKPTLF	IAFTKVYGGIPAE	85
RLA0 METAC	MAEERF	INTEN TPON	KKDETENJ	KELIQSHKI	VFGMVGIEGII	ATKMOKIRRDLKI	V-AVLKVSRNTLT	RALNOLGETIP	78
RLAO METMA	MAEERF	HTEHIPOW	KKDETENI	KELIQSHKY	<b>VFGMVRIEGI</b> I	ATKICKIRRDLK	V-AVLKVSRNTLT	RALNOLGESIP	78
RLA0 ARCFU	MAAVR	SPPEY	KVRAVEE I	KRMISSKP	VVAIVSERNVP	AGOMOK IRRE FRG	K-AEIKVVKNTLLE	RALDALGGDYL	75
RLA0 METKA	MAYKAKGOPPS	YEPKVAEW	KRREVKEI	KELMDE YEN	NVGLVDLEGIP	APOLOE IRAKLRE	RDTIIRMSRNTLMF	IALEEKLDERPELE	88
RLA0 METTH		-MAHVAEW	KKKE VOEI	HDLIKGYEY	VVGIANLADIP	AROLOKMROT LRI	S-ALIRMSKKTLIS	LALEKAGRELENVD	74
RLAO METTL	<b>MITA</b>	ESERK TAP	KIEEVNKI	KELLKNGO:	IVAL VOMME VP	AROLOE IRDE IR-	GTMTLEMSENTLIE	BATKEVARETGNPEFA	82
RLAO METVA	<mark>M</mark> IDAF	KSEHK IAPW	KIEEVNAI	KELLKSAN	VIALIDMME VP	AVOLOE IRDKIR-	DOMTLEMSENTLIS	RAVEEVARETGNPEFA	82
RLA0 METJA	METH	KYKAH <mark>VAP</mark> W	KIEEVKTI	KGLIKSKP	VVAIVDMMDVP	APOLOE IRDK IR-	DKYKLEMSENTLI	RALKE AAEE LNNPKLA	81
RLA0 PYRAB		MAHVAEW	KKKE VEEL	ANLIKSYP	VIALVDVSSMP	AYPLSOMRRLIRE	NGGLLRVSRNTLIE	LAIKKAAOELGKPELE	77
RLA0 PYRHO		MAHVAEW	KKKEVEEI	AKLIKSYPY	VIALVDVSSMP	AYPLSOMERL IRE	NGGLLEVSENTLIE	LAIKKAAKELGKPELE	77
RLA0 PYRFU		MAHVAEW	KKKEVEEL	ANLIKSYPY	VALVDVSSMP	AYPLSOMRRL IRE	NNGLLRVSRNTLIE	LAIKKVAGELGKPELE	77
RLA0 PYRKO		MAHVAEW	KKKEVEEI	ANTIKSYPY	VIALVDVAGVP	AYPLSKMRDKLR-	GKALLEVSENTLIE	LAIKRAADELGOPELE	76
RLA0 HALMA	MSAESEE	RKTET <mark>IP</mark> EW	KQEEVDAJ	VEMIESYES	SVGVVNIAGIP	SROLODMERDLHG	T-AELRVSRNTLLE	RALDDVDDGLE	79
RLA0 HALVO	MSESEVE	ROTEVIPON	KREE VDEI	VDFIESYES	SVGVVGVAGIP	SROLOSMRRE LHG	S-AAVRMSRNTLVN	RALDEVNDGFE	79
RLA0 HALSA	MSAEEQE	RTTEEVPEW	KROEVAEL	VOLLET YD:	SVGVVNVTGIP	SKOLODMRRGLHG	O-AALRMSRNTLLV	RALEE AGDGLD	79
RLA0 THEAC		MKEVSOO	KKELVNEL	TORIKASRS	SVAIVDTAGIS	TROIODIRGKNRG	K-INLKVIKKTLLF	KALENLGDEKLS	72
RLA0 THE VO		MRKINPK	KKE IVSEI	AOD ITKSK	AVAIVDIKGVB	TROMODIRAKNEL	K-VKIKVVKKTLLF	KALDSINDEKLT	72
RLA0 PICTO		MTEPAQW	KIDFVKNI	ENEINSRK	VAAIVSIKGLB	NNEFOKIENSIE	K-ARIKVSRARLLF	LAIENTGKNNIV	72
ruler	1	20		30	40	. 50			

Figure: A protein multiple sequence alignment

### Speciality of Implementation on FPGA



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

Length of subject sequence:  ${\bf M}$  Length of query HMM:  ${\bf K}$ 

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

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 chip	xc5vsx50t-1ff1136					
	Maxinum	160.46	157.31	144.26	133.87		
	Frequency	MHz	MHz	MHz	MHz		
Б	Flip-Flops	13978	5665	18393	7433		
F P G A		(43%)	(17%)	(56%)	(22%)		
	LUTs	10092	3471	12828	4412		
		(31%)	(10%)	(39%)	(13%)		
	DSP48e	104	104	144	144		
		(36%)	(36%)	(50%)	(50%)		
	Running Time	24.928ns	25.428ns	34.66ns	37.35ns		
С	CPU Model	Intel Pentium Dual E2200 @2.2GHz					
P U	Running Time	1.48us		2.1us			

Figure: Performance and resource utilization comparison[7]

- 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

- 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



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

- 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)