Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 3, Issue. 4, April 2014, pg.648 – 652

RESEARCH ARTICLE

ISONET: HARDWARE BASED JOB QUEUE MANAGEMENT FOR MANY CORE ARCHITECTURES

[1] Jeyanthi.S, [2] Mrs. M. Deivakani

[1]Department of ECE @ Anna University, India

[2]Department of ECE @ Anna University, India

[1] jairehshalom@gmail.com, [2] mdeivakani@gmail.com

Abstract- In this paper, we introduces IsoNet, hardware- based conflict-free dynamic load distribution and balancing with heavy job transfers. IsoNet is a lightweight job queue manager which makes faster thee list of jobs to be executed, and maintaining load balance among all Chip multi processor cores. For balanced distribution of workload based Job Queue Management for Many-Core Architectures. To implement S-box method in each queue, The Plain text of 4 bit data is considered for encryption algorithm utilizing key. For high speed applications, the Non LUT based implementation of S-box is preferred. Performance evaluation of the design with respect to area, power, and time has been done.

Key words- Load Balancer, Encryption, Cipher text, S -box, Sub-bytes Transformation, Decryption, Rijaendal Algorithm

---

I. Introduction

The realization of IsoNet, a lightweight on-chip micro network of load distribution and balancing modules, which can swiftly transfer jobs between any two cores, based on prevailing load conditions. The enhanced IsoNet can transfer multiple jobs per cycle, to accommodate even higher load distribution and balancing demands. We utilize a full-system simulation framework with real application workloads for near-term scalability analysis involving CMPs with 4 to 64 processing cores. For long term scalability analysis involving CMPs up to 1024 cores, we employ a cycle accurate trace-driven simulator. The architectures that scale to the large number of packets (N) and large number of priority levels (P) necessary in modern switch designs. A tunable parameter allows switch designers to carefully balance the trade-off between bus loading and chip area. In this paper we proposed the encryption method using S-BOX. The forward Substitution byte transmission, called Sub Bytes, is a simple table. AES

Defines a 16*16 matrix of bytes values called S-BOX.

II. Existing work

They are three queue is assigned each queue is having specific operation that queue is connected with the balancer, that the balancer is used to give a instruction for processor to identify which queue is required to access first according to that instruction the processor will handle the queue than the final output is received from balancer.

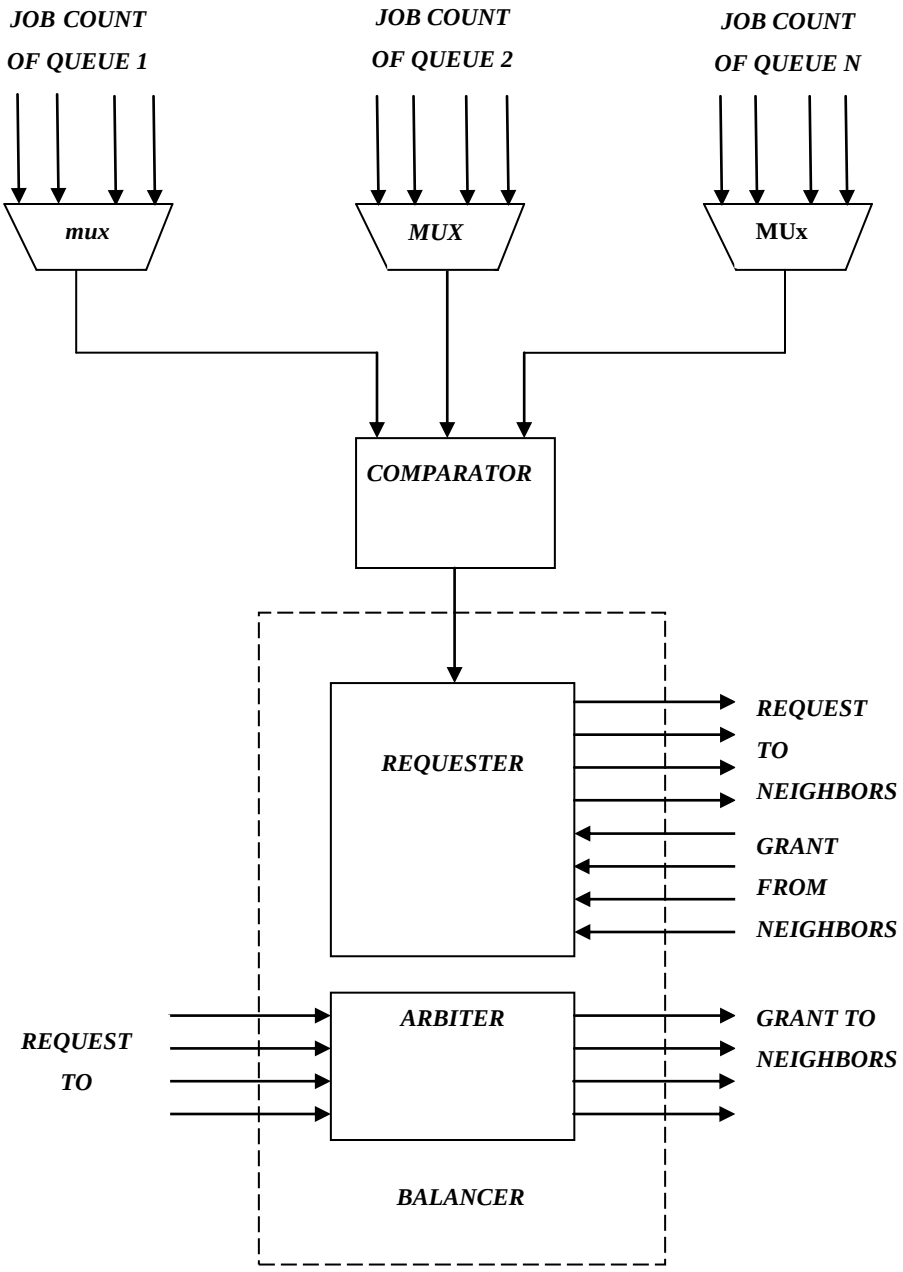

FIG (1) THE ENHANCED ISONET DESIGN THAT SUPPORTS MULTIPLE JOB TRANSFERS PER ISONET CYCLE

This feature enables the definition of "classes" of resources—a class being defined as a set of resources that share common characteristics and are mapped to the same key value. Structured systems like Chord can produce imbalance problems depending on the location of peers and the statistical distribution of the values of resource keys. In Self-Chord, the keys are fairly distributed over the peers, irrespective of the location of peers and the distribution Of key values.

In Chord, appropriate operations are necessary when a peer joins the ring or when new resources are published: These resources must be immediately assigned to the peers whose indexes match the resource keys. These operations are not necessary in Self-Chord because the mobile agents are always active and will spontaneously reorganize the keys. This assures scalability (keys are continuously reordered as the Network grows) and robustness with respect to environmental changes.

The system is not centralized, making administration difficult. And also Lack of security. No computer in the network is reliable.

III. **Proposed System**

Sub bytes Transformations

Sub bytes works on bytes or words, so the received 128 bits will be grouped to 16 words of length 8 bits. Each word is mapped to the rows and columns of the content of constant S-box matrix and the value is replaced. Similarly all the 16 words are replaced and arranged in a form of matrix by filling the columns first.

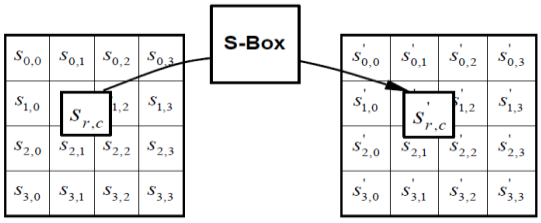

Fig.2.Subbytes

IV. Key Schedule

Each Round Key is derived from the original cipher key through the Key Schedule. The cipher Key is expanded into a large array of words, the size of which is subject to the specified key length.

S BOX

	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xa	xb	xc	xd	xe	xf
0x	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1x	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2x	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3x	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4x	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5x	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6x	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7x	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8x	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9x	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
ax	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
bx	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
cx	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
dx	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
ex	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
fx	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

The Rijaendal S-Box was specifically designed to be resistant to linear and differential cryptanalysis. This was done by minimizing the correlation between linear transformations of input/output bits, and at the same time minimizing the difference propagation probability.

In addition, to strengthen the S-Box against algebraic attacks, the affine transformation was added. In the case of suspicion of a backdoor being built into the cipher, the current S-box might be replaced by another one. The authors claim that the Rijaendal cipher structure should provide enough resistance against differential and linear cryptanalysis, even if an S-Box with "average" correlation / difference propagation properties is used.

Inverse S-Box

The inverse S-box is simply the S-box run in reverse. For example, the inverse S-box of `0xb8` is `0x9a`. It is calculated by first calculating the inverse affine transformation of the input value, followed by the multiplicative inverse. The inverse affine transformation is as follows:

	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xa	xb	xc	xd	xe	xf
0x	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1x	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2x	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3x	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4x	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5x	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6x	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7x	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8x	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9x	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
ax	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
bx	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
cx	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
dx	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
ex	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
fx	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Encryption

In cryptography, *encryption* is the process of encoding messages (or information) in such a way that only authorized parties can read it. Encryption doesn't prevent hacking but it reduces the likelihood that the hacker will be able to read the data that is encrypted. In an **encryption scheme**, the message or information (referred to as plaintext) is encrypted using an encryption algorithm, turning it into an unreadable cipher text .

This is usually done with the use of an encryption key, which specifies how the message is to be encoded. Any adversary that can see the cipher text should not be able to determine anything about the original message. An authorized party, however, is able to decode the cipher text using a **decryption** algorithm, that usually requires a secret decryption key, that adversaries do not have access to. For technical reasons, an encryption scheme usually needs a key-generation algorithm to randomly produce keys.

V. Conclusion

Nowadays security is the most important one. We provide the better security using a cryptographic algorithm. To encrypt the plain text using a key to produce cipher text. Rijaendal algorithm has high security margin compared to another AES candidates. It has 32bit uniform across CPUs, easily fit on small smart cards. It should be efficient across all platforms in hardware implementation. The designed core supports both encryption and decryption standards. Its functionality has been verified using simulation, by taking various inputs and is synthesized by using Xilinx ISE 12.1.

References

[1] Alan Kaminsky, Michael Kurdziel, Stanislaw Radziszowski, (2010) "An overview of cryptanalysis research for the advanced encryption standard", IEEE, the 2010 military communications conference unclassified program cyber security and network management, pp. 1310.

[2] .L. Soares, C. Menier, B. Raffin, and J. L. Roach, (2007), "Work stealing for time constrained octree exploration: Application to real-time 3D modeling," in Proc. Euro graph. Symp. Parallel Graph. Visualization, pp. 1–9.

[3] D. Sanchez, R. M. Yoo, and C. Kozyrakis,(2010) "Flexible architectural support for fine-grain scheduling," in Proc. Int. Conf. Arch. Support Program. Lang. Operat. Syst., pp. 311–322

[4] B. Saha, A.-R. Adl-Tabatabai, A. Ghuloum, M. Rajagopalan, R. L. Hudson, L. Petersen, V. Menon, B. Murphy, T. Shpeisman, E. Sprangle, A. Rohillah, D. Carmean, and J. S. Fang, (2007)"Enabling scalability and performance in a large scale CMP environment," ACM SIGOPS Operat. Syst. Rev., vol. 41, no. 3, pp. 73–86.

[5] J. Giacomoni, T. Moseley, and M. Vachharajani, (2008)"Fast-forward for efficient pipeline parallelism a cache-optimized concurrent lock-free queue," in Proc. ACM SIGPLAN Symp. Principles Practice Parallel Program, pp. 43–52.

[6] W. N. Scherer, III, D. Lea, and M. L. Scott, (2009),"Scalable synchronous queues," Commun. ACM, vol. 52, no. 5, pp. 100–111.

[7] M. Moir, D. Nussbaum, O. Shalev, and N. Shavit,(2005), "Using elimination to implement scalable and lock-free FIFO queues," in Proc. 17th Annu. ACM Symp. Parallel. Algorithms Arch., pp . 253–262.

[8] Y. Etsion, F. Cabarcas, A. Rico, A. Ramirez, R. Badia, E. Ayguade, J. Labarta, and M. Valero, Dec 2010,"Task superscalar: An out-of-order task pipeline," in Proc. 43rd Annu. IEEE/ACM Int. Symp. Microarch., pp . 89–100.

[9] M. Sjalander, A. Terechko, and M. Duranton, Sep 2008,"A look-ahead task management unit for embedded multi-core architectures," in Proc. 11th EUROMICRO Conf. Digital Syst. Design Arch., Methods Tools, pp. 149–157.

[10] .C. H. Shann, T. L. Huang, and C. Chen,(2000) "A practical nonblocking queue algorithm using compare-and-swap," in Proc. Int. Conf. Parallel Distrib. Syst., pp. 470–475.

