Power Efficient Message Retrieval Architecture on Data-Centric Environment over the Internet of Things

Hansoo Kim

Abstract—**A power efficient IoT (Internet of Things) message retrieval architecture for data-centric environment is proposed. To meet the requirements of the data-centric IoT environment, variable stride pipelined AC-DFA (Aho-Corasick Deterministic Finite Automata) string matching scheme is adopted for parsing and processing of the IoT message. Also, scalable power saving schemes for pipelined string matching architecture is applied and up to 12% of the power consumption is reduced compared to the conventional architectures.**

Index Terms—**Information retrieval, internet of things, pipelined string matching, power consumption.**

I. Introduction

The IoT (Internet of Things) vision is to allow things to be connected anytime, anywhere, with anything and anyone, ideally using any path, network, and service [1]. As an emerging technology, the IoT is expected to offer promising solutions to transform the operation and role of many existing industrial systems such as transportation systems and manufacturing systems. For example, when the IoT is used for creating intelligent transportation systems, the transportation authority will be able to track each vehicle's existing location, monitor its movement, and predict its future location and possible road traffic [2]. A commonly accepted definition for the IoT can be described as a dynamic global network infrastructure with self-configuring capabilities based on standard and interoperable communication protocols where physical and virtual 'Things' have identities, physical attributes, and virtual personalities and use intelligent interfaces, and are seamlessly integrated into the information network [3].

There is a growing interest in using IoT technologies in various industries. A number of the industrial IoT projects have been conducted in areas such as agriculture, food processing industry, environmental monitoring, security surveillance, and others [4]. Specifically, the integration of sensors, RFID tags, and communication technologies serves as the foundation of the IoT and explains how a variety of physical objects and devices around us can be associated to the Internet and allow these objects and devices to cooperate and communicate with one another to reach common goals [5].

Manuscript received September 8, 2017; revised December 10, 2017. This work (Grants No. C0564714) was supported by Business for Cooperative R&D between Industry, Academy, and Research Institute funded Korea Small and Medium Business Administration in 2017.

Hansoo Kim is with Department of Convergence Security, Seowon University, 377-3, Mushimseoro, Seowongu, Cheongjusi, Choongcheongbukdo, 28674 Republic of Korea (e-mail: kutestar@seowon.ac.kr).

As an emerging wave of the Internet deployments, the IoT requires the ability of parsing and processing of heterogeneous, robust and large data in low latency and secure environment. Also, as mobile environment such as battery-powered portable systems emerges, the power consumption required for the functioning of the IoT devices is of significant concern [6].

In this paper, a power efficient IoT message retrieval architecture is proposed which can parse and process heterogeneous data in a fast, low-latency and secure way. A pipelined AC-DFA trie string matching with variable-stride and adaptive clock scaling schemes is applied and the reduction of up to 12% in power consumption is obtained, compared with the conventional architectures.

The rest of this paper is organized as follows. Section II describes the characteristics of the IoT data and relevant recent studies, and Section III shows the proposed architecture which reduces the power consumption of the data-centric IoT message retrieval system. Section IV shows the experimental results and Section V concludes this paper.

II. Related Works

A. Characteristics of the IoT Data

The main characteristic of the IoT data is that data-centric, robust, and large data can be parsed and processed to meet the modern IoT environment requirements [7]. Detailed aspects are as follows.

First, the data are heterogeneous due to the various kinds of connected device working together. Technically, the design of the IoT architecture needs to consider extensibility, scalability, modularity, and interoperability among heterogeneous devices. As things might move or need interaction with their environment, an adaptive architecture is needed to help devices dynamically interact with other things. The heterogeneous nature of the IoT requires that the architecture provides the IoT efficient event-driven capability [2].

Second, robust and large data streams need to be processed with low latency for real-time interaction. The large-scale IoT applications need to process and manage massive data-sets across geographically distributed datacenters [7].

Third, Since the IoT consists of various network-connected devices, when one device is infiltrated by malware, it can become the starting point for the spread of the infiltration to other devices that could ultimately threaten critical infrastructure that should ordinarily be protected. Actual past security incidents have demonstrated that vulnerabilities in the communication software of devices connected to critical infrastructure such as work-use personal computers and surveillance cameras have been targeted to enable unauthorized access from outside. These devices have been used as starting points for making critical infrastructure operate abnormally. Thus, the increasing number of potential targets for attacks due to the increase in number of IoT devices, as well as the growing scope of influence of attacks has been pointed out. Also, it has been pointed out that IoT systems require little human involvement, so they can easily lapse into a situation where there is a shortage of administrators that makes attack detection difficult, and that long life cycles, over 10 years long, result in attacks that continue for long periods of time [8].

B. Variable-Stride Pipelined Architecture with AC-DFA Trie

The functions of the data processing architecture such as the IoT message retrieval rely on multi-pattern string matching, which scans the input stream to find all occurrences of a predefined set of string-based patterns [9]. Along with the considerations of speed, accuracy and the memory space required for storing and searching patterns, the power consumption required for the functioning of the IoT message retrieval is of significant concern. The majority of the power consumption used for the IoT message retrieval is regarded as the multi-pattern string matching [10], hence a large amount of research on reducing the power consumption is focused on string matching.

Many multi-pattern string matching solutions adopt the well-known Aho-Corasick (AC) algorithm, where the system is modeled as a deterministic finite automaton (DFA) [11] as well as pipelining [12] has performed a great role in string matching. AC-DFA converts a pattern set which contains n characters into a deterministic finite automaton with *O(n)* states. Once the DFA which can be stored as a state transition table is built, it reads the input stream one character per clock cycle. Each input character is processed only once and results in exactly one state transition [9].

In addition to this, the compressed AC-DFA trie [13] has one transition on multiple input characters, by combining *k* consecutive states of the original Aho-Corasick DFA trie. The original AC-DFA trie can be compressed by dividing each pattern to be matched into a certain number of characters, which is called a *stride*. Assuming that the *stride* = 2, two input characters are fetched at a time and compared against the patterns on edges from the current node. If there is any match for these two input characters, transition is made accordingly and the next two input characters are fetched. If there is no match for these two input characters on any edges from the current node, then next two input characters should be fetched with one-character offset to ensure that no patterns are missed.

Using the aspect that the failure transitions are added into the AC-DFA trie for a failed match in order to reuse the history information without restarting from the root, the input characters are delivered to all the pipeline stages in parallel, including the new input characters delivered to the root with a one character offset at each clock cycle so as the failure transitions to the stages that are on the pipelines are removed. With this approach, all the failure transitions can be removed which significantly reduces the memory space required for storing the pattern set [12].

Besides, when the input data is processed, the stages of the pipelined AC-DFA trie are sequentially accessed from the first stage. If there is a mismatch in the preceding stage, the rest of the stages are not accessed. So, the first stage is always accessed by any input data, while the last few stages are rarely accessed. Researches show that the number of accesses decreases drastically as input data traverses the stages of the pipelined AC-DFA trie. For example, the number of accesses of the fourth stage in a pipelined string matching system is one thousandth of that of the first stage [6].

C. Clock Scaling for the Reduction of Power Consumption in CMOS Devices

As the processor speed grows fast and mobile devices prevail widely over the world, the power dissipation becomes an important consideration in the system design. Most digital circuits are constructed using CMOS, therefore the analysis of power dissipation in CMOS circuits is essential and the dynamic power is found as the main portion of the CMOS power dissipation [14].

A lot of researches to reduce the power consumption in CMOS circuitry have been done recently [15]. Among these, clock scaling is a highly effective method to minimize the energy dissipation without any appreciable degradation in the quality of service. It scales the operating clock frequency to minimize the energy dissipation, but delays such as the processing latency and clock skew should be considered. Delays and clock skew may happen if a processing logic executes an instruction when its clock speed is scaled or lower than its original clock frequency. To reduce delays and clock skew in implementing clock scaling, additional hardware components such as clock buffers and repeaters should be inserted in high frequency pipelined data paths which require additional power and delays that may lead to performance degradation[16], [17].

III. Proposed Architecture

To meet the requirements of the IoT data that data-centric, robust, and large data can be parsed and processed [7] with battery-powered resource, the main contribution of this paper includes the application of the previous architecture [6] to the Internet of Things environment

A. Proposed Schemes and Definition

As seen in Section II, one of the promising data retrieval technology is pipelining which can parse and process heterogeneous and data-centric IoT data, as already used in various applications such as IP forwarding in routers, and DPI (Deep Packet Inspection) in IDS (Intrusion Detection System) So, pipelined AC-DFA string matching scheme [12] is adopted for the proposed IoT message retrieval architecture. Also, variable-stride AC-DFA trie construction scheme and clock scaling scheme of the pipeline stages [6] are adopted.

In Equation (1), the *stride* (*s*) of a node is defined to be the number of input characters to be matched at a time in traversing an AC-DFA trie. The *height* (*L*) of a trie is the length of the path from the root to the deepest node in the trie. All nodes of depth *t* are assigned at level *t* in the trie. The *degree* (*d*) of a node is defined as the number of outgoing edges [18]. The *alphabet size* (*p*) is the number of different patterns (characters) that can be mapped onto an edge in an AC-DFA trie (in bytes). $N_{fetch}(d, p, L)$ is defined as the number of memory accesses to fetch the patterns of the IoT message as a function of *d*, *p*, and *L* of the AC-DFA trie.

$$N_{fetch}(d,p,L)=\sum_{i=1}^{L}\left(\frac{d}{p}\right)^{i-1}\cdot\sum_{j=0}^{d-1}\frac{p-j}{p}\cong\frac{1}{1-\frac{d}{p}}\cdot d=\frac{p\cdot d}{p-d}\qquad(1)$$

The number of memory accesses in fetching the IoT message can be described as Equation (1), and since the actual AC-DFA trie may not be regular, an iterative method is devised to change the stride to minimize the number of memory accesses.

$$P=N_a\cdot P_w+N_p\cdot P_r+\left(N_x-N_a-N_p\right)\cdot P_s\qquad(2)$$

Also, the operating clock of the rarely-used stages in the pipeline is decreased to avoid unwanted power waste. In Equation (2), N_p is the number of characters of the input data to the pipelined AC-DFA trie. N_a is the total number of accesses to the pipelined AC-DFA trie in processing pattern matching. N_l is the length of the longest pattern, which equals to the height of the AC-DFA trie. N_x is the maximum number of accesses to the AC-DFA trie, which is $N_p \cdot N_l$ (Assume that N_l is equal to the number of stages of the pipeline). *P* is the total power consumption of the proposed IoT message retrieval system.

A stage of the AC-DFA trie is defined to be *working* when the stage performs a pattern matching. A stage of the AC-DFA trie is defined to be *ready* when the stage is ready for a pattern matching. A stage of the AC-DFA trie is defined to be *sleeping* when the stage is *ready* but its clock frequency is decreased. A stage is *turned on* when the stage changes from *sleeping* to *ready*. A stage is *turned off* when the stage changes from *ready* to *sleeping*.

P_w is the value of the power consumption of a stage when the stage is *working*. P_r is the value of the power consumption of a stage when the stage is *ready*. P_s is the value of power consumption of a stage when the stage is *sleeping*. It is assumed that all the pipeline stages in the IoT message retrieval architecture have the same hardware architecture and the power consumption of all the stages is equal for each status - *working*, *ready* or *sleeping*.

To reduce delays and clock skew while decreasing the clock frequency of stages, the first stage is *turned on* before a stage performs a pattern matching instead of using additional hardware components such as clock buffers or repeaters. The clock frequency of the stage following the *working* stage is set to the original clock frequency (*sleeping* to *ready*), so the power consumption of the stage is changed to *Pr* from *Ps*. The *turned on* stage acts as a clock buffer to eliminate glitches.

B. Architecture Overview and Workflow

The proposed IoT massage retrieval architecture is shown in Fig. 1, which mainly described in the previous literature [6]. Various kinds of the IoT data from the heterogeneous IoT devices (denoted as *the IoT cloud*) come into the proposed architecture, as the *input data stream*. The first stage of the pipeline (*Stage 0*) accesses the memory to fetch the IoT message, via the *memory access controller*. The IoT messages are constructed as a form of the AC-DFA trie, and the character sets of each level of the trie are assigned to the designated pipeline stage.

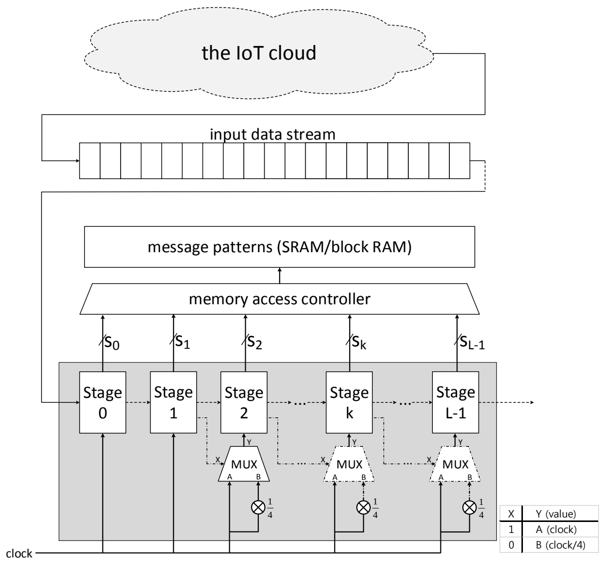

Fig. 1. The proposed IoT message retrieval architecture.

When a pipeline stage performs the pattern matching, it signals to the following stage to *turn* it *on*. The signal is called the *toggle signal* and denoted as chain lines with arrows. The stage results in the *match notification* to the following stage as a result of the pattern matching. Dotted lines with arrows denote the *match notifications* (*L* denotes the number of the pipeline stages).

When there is a mismatch on Stage *k* ($0 \leq k \leq H$-1), it sends a *toggle signal* (x=0) to Stage *k*+1 to *turn* it *off* and the clock frequency of Stage *k*+1 is set to *clock*/4.

IV. Experimental Results

Experiments on the IoT message samples based on the recommendations of COAP [19] are performed, where the AC-DFA trie is constructed from the generated message samples. The number of the generated message is 2,048 and the constructed AC-DFA trie has a maximum of 32 levels (the largest message has 32 characters) with a total of 38,524 characters.

The IoT message retrieval system with the AC-DFA trie is implemented with Microsoft Visual C++ 2017. In the experiment, the essential hardware parameters are adopted from the latest IoT device design board so that the power consumption is correctly predicted and implicit assumptions are insignificant. The parameters from Xilinx Spartan 6 XC6SLX16 [20] are adopted as the target device and the power consumption of the stage of the pipelined AC-DFA trie is obtained using the Xilinx Power Estimator (XPE) 14.3 [20].

The operating clock frequency is set as 800 (MHz) and slowed down frequency as 1/4 of the original clock frequency, which is 200 (MHz). The appropriate value of the scaling constant (1/4) depends on the practical hardware system and implementation techniques, and the value of 1/4 is suitable for the implemented system.

Only the levels of the AC-DFA trie are considered which are the destinations of the cross transitions [9], [12], so the first 21 levels of the original AC-DFA trie are compressed and assigned to the stages of the pipeline. Also, it is assumed that no hashing algorithms are applied so that the memory storing the patterns is to be accessed each time the pattern matching from the pipeline stage is performed. Data samples from Geospecies Knowledge Base RDF Dump [21] are used as the input stream. The experimental results are shown in Table I.

TABLE I: THE EXPERIMENTAL RESULTS

architectures	stride	memory accesses	power consumption (mW)	reduction (%)
no pipeline	N/A	42.39	6,400	11.94
pipeline	fixed	39.00	6,150	7.57
Proposed	variable	33.12	5,717	-

Note: the reduction ratio is the power consumption reduction ratio of the proposed architecture compared to the relevant architectures.

As a result from Table I, the proposed architecture reduces up to 11.94% of the power consumption, compared to the conventional architectures.

V. CONCLUSION

In this paper, power efficient IoT message retrieval architecture is proposed. Using this architecture, up to 12% of the power consumption is reduced compared with the conventional message retrieval architectures. In addition, the proposed architecture meets the requirements of the IoT environment, adopting the pipelined string matching scheme for parsing and processing of heterogeneous, low-latency required, and security-aware IoT data.

It is obvious that the proposed architecture is applicable for various kinds of the IoT system with little cost and complexity in implementation. Also, the proposed architecture can be applied to other applications such as the information retrieval from the Big Data, and finding evidence over large digital data on the crime scene.

REFERENCES

[1] L. Wang and R. Ranjan, "Processing distributed Internet of Things data in clouds," *IEEE Cloud Computing*, vol. 2, no. 1, pp. 76-80, April 2015.

[2] L. D. Xu, W. He, and S. Li, "Internet of Things in industries: A survey," *IEEE Trans. Industrial Informatics*, vol. 10, no. 4, pp. 2233-2243, Jan. 2014.

[3] R. V. Kranenburg, *The Internet of Things: A Critique of Ambient Technology and the All-Seeing Network of RFID,* Amsterdam, The Netherlands: Institute of Network Cultures, 2007.

[4] Y. Li, M. Hou, H. Liu, and Y. Liu, "Towards a theoretical framework of strategic decision, supporting capability and information sharing under the context of Internet of Things," *Information Technology and Management*, vol. 13, no. 4, pp. 205-216, April 2012.

[5] R. V. Kranenburg, E. Anzelmo, A. Bassi, D. Caprio, S. Dodson, and M. Ratto, "The Internet of Things," in *Proc. 1st Berlin Symposium of Internet Society*, Berlin, 2011, pp. 25-27.

[6] H. Kim, "A scalable architecture for reducing power consumption in pipelined deep packet inspection system," *Microelectronics Journal*, vol. 46, no. 10, pp. 950-955, Oct. 2015.

[7] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the Internet of Things," in *Proc. 1st MCC Workshop*, Helsinki, 2012, pp. 13-16.

[8] S. Tanaka, K. Fujishima, N. Mimura, T. Ohashi, and M. Tanaka, "IoT system security issues and solution approaches," *Hitachi Review*, vo. 65, no. 8, pp. 359-363, Sep. 2016.

[9] W. Jiang, Y. E. Yang, and V. K. Prasanna, "Scalable multi-pipeline architecture for high performance multi-pattern string matching," presented at the IEEE International Parallel and Distributed Processing Symposium (IPDPS), Atlanta, Georgia, April 19-23, 2010.

[10] N. Hua, H. Song, and T. V. Lakshman, "Variable-stride multi pattern matching for scalable deep packet inspection," presented at the IEEE International Conference on Computer Communications (INFOCOM), Rio de Janeiro, Brazil, April 19-25, 2009.

[11] A. V. Aho and M. J. Corasick, "Efficient string matching: An aid to bibliographic search," *Communications of the ACM*, vol. 18, no. 6, pp. 333-340, June 1975.

[12] D. Pao, W. Lin, and B. Liu, "Pipelined architecture for multistring matching," *Computer Architecture Letters*, vol. 7, no. 2, pp. 33-36, Feb. 2008.

[13] M. Alicherry, M. Muthuprasanna, and V. Kumar, "High speed pattern matching for network IDS/IPS," in *Proc. 2006 IEEE International Conference on Network Protocols*, 2006, pp. 187-196.

[14] D. R. Suleiman, M. A. Ibrahim, and I. I. Hamarash, "Dynamic voltage frequency scaling (DVFS) for microprocessors power and energy reduction," presented at the 4th International Conference on Electrical and Electronics Engineering, Bursa, Turkey, Dec. 7-11, 2005.

[15] T. D. Burd, T. A. Pering, A. J. Stratakos, and R. W. Brodersen, "A dynamic voltage scaled microprocessor system," *IEEE Journal of Solid-State Circuits*, vol. 35, no. 11, pp. 1571-1580, August 2002.

[16] M. Pedram and J. Rabeay, *Power Aware Design Methodologies*, Norwell, MA, USA: Kluwer Academic Publishers, 2002, pp. 171-174 and pp. 386-412.

[17] V. Tiwari, D. Singh, S. Rajgopal, G. Mehta, R. Patel, and F. Baez, "Reducing power in high-performance microprocessors," in *Proc. Design Automation Conference*, June 1998, pp. 732-737.

[18] C. A. Shaffer, *A Practical Introduction to Data Structures and Algorithm Analysis*, 3rd ed. (C++ Version). Blackburg, VA: Virginia Tech., 2010, p. 206.

[19] C. Bormann and E. Z. Shelby, "Block-wise transfers in the constrained application protocol (CoAP), RFC7252," *Internet Engineering Task Force*, Aug. 2016.

[20] Xilinx Power Estimator (XPE), Xilinx Inc. [Online]. Available: http://www.xilinx.com

[21] P. J. Devries. Data Set RDF Dumps of Geospecies Knowledge Base. World Wide Web Consorsium. [Online]. Available: https://www.w3.org/wiki/DataSetRDFDumps

Hansoo Kim was born in Seoul, Korea, in 1975. He received the B.E., M.E., and Ph.D. degrees in electronic engineering from Sogang University, Korea, in 2002, 2004, and 2014, respectively. He worked as a Junior Engineer at the Digital Media Research Laboratory in LG Electronics until 2005, as a Senior Engineer in Nextreaming Corp. until 2006, and as a Forensic Scientist in National Forensic Service until 2017. He is currently an Assistant Professor in Seowon University, Korea. His research interest and work experience include string matching, IoT security, machine learning, IPv6, multimedia streaming, digital image forensics, document analysis, network security and cybercrime.