Asian Journal of Information Technology 15 (22): 4571-4577, 2016
ISSN: 1682-3915
© Medwell Journals, 2016

An Enhanced Approach of Message Digest Scheme for FPGA Based Sensor Node

[1]A. Saravanaselvan and [2]B. Paramasivan
[1]Department of ECE, National Engineering College, K.R.Nagar Post, Kovilpatti, Thoothukudi District, Tamil Nadu, India
[2]Department of CSE, National Engineering College, K.R.Nagar Post, Kovilpatti, Thoothukudi District, Tamil Nadu, India

Abstract: Most of the current sensor devices use general purpose processor which is not primarily designed to offer the inherent parallelism for event-driven sensor applications. FPGA based event-driven architecture can handle several events in parallel which removes the timing uncertainty and overhead. Due to random deployment of sensor nodes in remote areas, nodes are vulnerable to many security attacks related to data confidentiality, integrity and authentication. With respect to resource constraints of sensor nodes, traditional security mechanisms are not suitable for wireless sensor nodes. Considering this, a message digest scheme by using maximum length sequence primitive polynomial approach is presented for FPGA based event-driven architecture so as to ensure node-level data integrity for time and mission critical tasks. In addition, configurable serial transmitter and receiver blocks were added to the designed block using which data size can be configured as per the application scenario. The performance of this scheme is evaluated and compared with other existing processor-based schemes.

Key words: Event driven architecture, FPGA, sensor node, security attacks, integrity, authentication

INTRODUCTION

Wireless sensor nodes are more vulnerable to many security threats due to the usage of wireless medium. To withstand the vulnerabilities, security services such as authentication, confidentiality and data integrity need to be maintained in each node of the sensor network. Typically, a sensor node is severely constrained in terms of size, computation capability and energy resources (Chong and Kumar, 2003; Krasteva *et al*., 2011; Kumar *et al*., 2014). One of the most challenging issues in sensor networks is injection of false data by compromised nodes. In real-world WSNs, the nodes cannot be assumed to be trustworthy apriori. Therefore, researchers have focused on building a sensor node trust model to solve the problem in message integrity (Vece *et al*., 2015). Prior researches have studied different approaches for achieving message integrity in wireless sensor nodes. Digital watermarking is one of the popular technologies in wireless sensor network security. This approach can effectively prevent sensed data being intercepted and precisely detect whether the data have been tampered with. But, the digital watermarking model can be refined to detect the whole data lost (Jariwala *et al*., 2014). General purpose processors offer sequential execution of instructions and it will not support for inherent parallelism which is required for mission critical applications. Moreover, protocol stack based execution is always slower than the hardware based execution. In addition, processor based sensor nodes bring uncertainty as it is interrupted by other events any time (Lu *et al*., 2014). For real time mission critical applications, design of FPGA based sensor node is mandatory. In a larger wireless sensor network, efficient in-network data aggregation is required in order to save the node level energy and in turn it will reduces the unnecessary transmissions. This problem is significant since sensor networks are highly vulnerable to node compromises due to several integrity related attacks (Liao *et al*., 2013; Okobiah *et al*., 2014). Modern FPGA technology is used to solve the current challenges in wireless sensor networks due to its unique characteristics. Now a days, FPGA process technology offer optimized platform to carry out new designs with an enhanced manner. Hence, the FPGAs can provide high processing capabilities and high-speed acceleration of algorithms such as security, data compression and local data processing (Zhang *et al*., 2010; Sun *et al*., 2013, 2012; Dener and Bostancyoglu, 2015). A

Corresponding Author: A. Saravanaselvan, Department of ECE, National Engineering College, K.R. Nagar Post, Kovilpatti, Thoothukudi District, Tamil Nadu, India

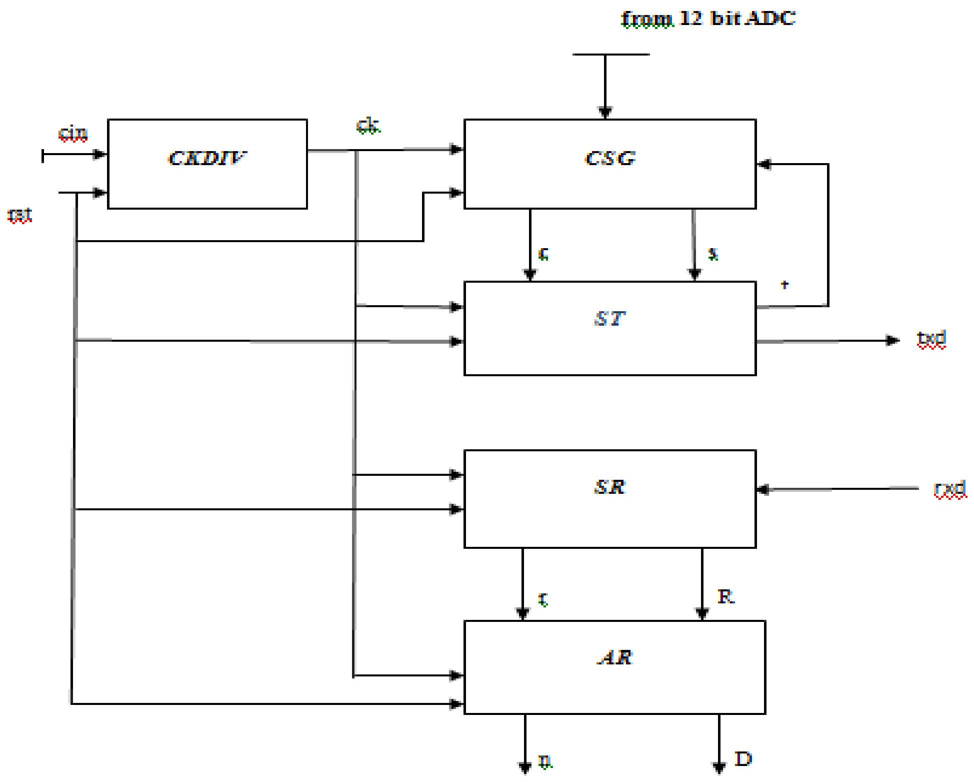

Fig. 1: Block diagram of DCSG and DCSV scheme

programmable ASIC tool support to process complex algorithms in an efficient way due to its unique characteristic (Zhang *et al*., 2010; Sun *et al*., 2013). In addition, power performances are often a crucial constraint in the design of sensor nodes, in consequence of the wide diffusion of battery-supplied devices as well as the reliability issues due to high clock frequencies (Dener and Bostancyoglu, 2015). Motivated by previous research, a maximum length sequence primitive polynomial approach based message digest scheme is developed for event-driven architecture. The message digest scheme uses maximum length sequence primitive polynomial approach and it is implemented on cyclone II FPGA device.

Experimental procedure: The architecture of the message digest scheme is shown in Fig. 1. This consists of a data compressor and signature generator module, data compressor and signature verifier module, serial transmitter module, serial receiver module and clock divider. After sensing the environmental parameters, the results should be processed and transmitted to the other sensor nodes. When two sensor nodes are used to communicate effectively, ensuring data integrity is very essential.

This DCSG and DCSV scheme contains Clock Divider (CKDIV), Compressor and Signature Generator (CSG), Serial Transmitter (ST), Serial Receiver (SR) and Analyzer and Reproducer modules.

Clock divider module: This module is designed in such a way to control the speed of data transmission and reception among sensor nodes by a required baud rates such as 4800,9600,12400 etc. The 50 MHz crystal oscillator is fed as input to this module and mod 5208 counter is designed so as to convert this into 9600 Hz using behavioral modeling. The procedure for conversion is given in Algorithm1.

Algorithm1: Module ckdiv (cin,ck,rst):

```
Reg[12:0]Q = 13'd0;//Initial count value
Always@(cin,rst)
If (rst = = inactive)
Q=((Q+1)%(5208))
Assign ck = Q (Vivaksha Jariwala et al., 2014)
End module
```

In Algorithm 1, cin is an input clock and ck is an output clock in the module respectively. Q is a 13 bit counter which counts cin clock values up to the final count value 5208 for arriving at 9600 baud rate. The most

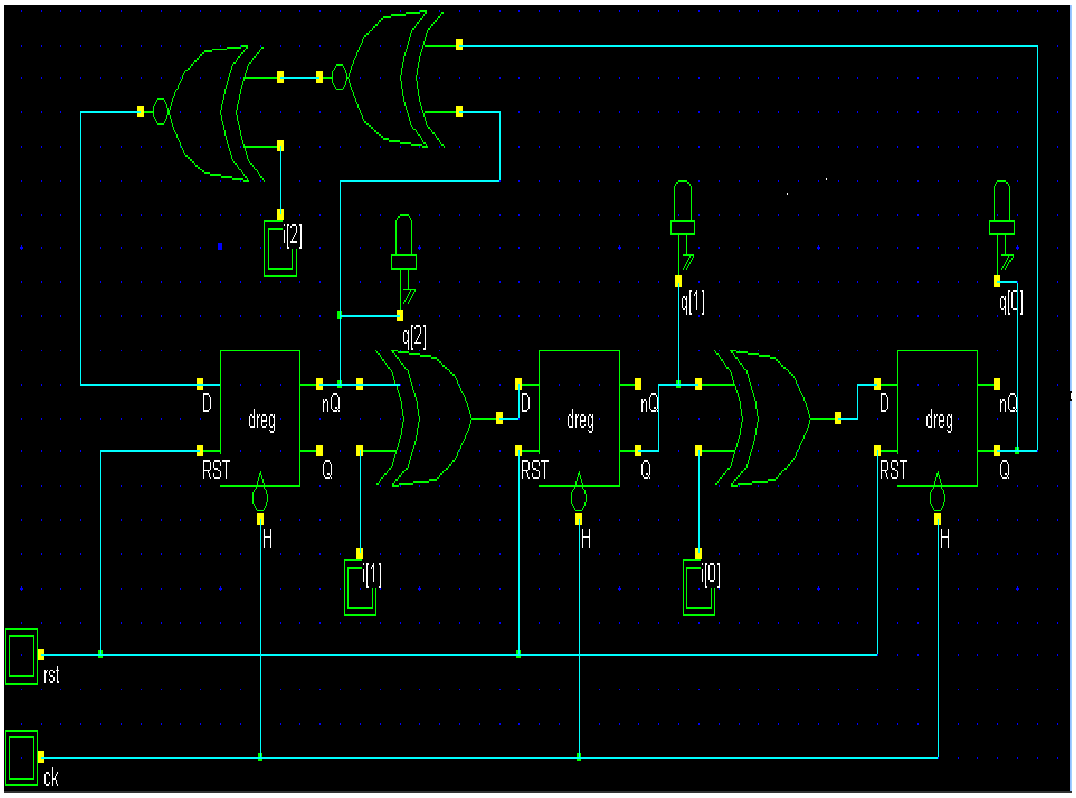

Fig. 2: Circuit diagram of MISR

significant bit of 13 bit counter is assigned as output clock ck. Depending upon required baud rates; the counter specification can be selected.

Compressor and signature generator module: The data compressor and signature generator module uses primitive polynomial p(x) as $P(x) = x^2+x^1+x$; where+sign represents the exclusive or operation and it is implemented by Multiple Input Signature Register (MISR) circuit shown in Fig. 2. Basically, MISR is a signature generator which accepts data input in a sequential order and finally produces unique sign for the whole data. The characteristic polynomial causes an MISR to generate a maximum length sequence called primitive polynomial.

In this process, the order of this polynomial is selected as 3 and the corresponding MISR generates a sequence of length 2^3-1=7. For a primitive polynomial of order n, the length of the sequence is 2^n-1. The q output of the above circuit is as follows: $q[0]^+ = i[0]+q[1]$; $q[1]^+$ $i[1]+q[2]$; $q[2]^+ = i[2] + q[0]+q[2]$; At the end of 7 cycles, MISR produces unique sign for the applied input at time t. The algorithm for CSG module is given in Algorithm 2.

Algorithm 2: Module CSG (ck,rst,d,t,c,s)

```
Initially set q=3'b100;id=9'b001001000;t=1'b0;c=1'b0;
Set local variables i=0;k=0;
always@(ck,rst)
if(rst= = inactive)
Check if (i= =0)&&(t= =1'b0)
z={d,id};i=i+1;
else if ((i>=1)&&(i<=7)&&(t=1'b0))
Perform $q[0]^+ = i[0] + q[1]$; $q[1]^+ = i[1] + q[2]$; $q[2]^+ = i[2] + q[0] + q[2]$;
where + sign represents exclusive or operation. Then, increment
k=k+3;i=i+1;c=1'b0;
else if ((i= =8)&&(t= =1'b0))
s={z,q};c=1'b1;i=i+1;
else if ((i= =9)&&(t= =1'b0))
i=i+1;c=1'b1;
else if ((i= =10)&&(t= =1'b0))
c=1'b0;i=0;k=0;
end module
```

In Algorithm 2, local variable i is used as pulse counter; k is used as control input for data entry; c is used as valid output indicator and t is used as transmission over indicator in the module respectively. During the active reset condition, initial values keep as such when reset is inactive, sensor data and node ID both are concatenated at first clock interval. Then, the next 8 clock pulses, data input is given in sequential order to produce unique signature after compression. During the

9th clock interval, the signature is appended to whole input data and it is given as input to serial transmitter module for transmission.

MATERIALS AND METHODS

Serial transmitter module: This module is designed in such a way that it can pack the data at any size. The user can configure the data size based on the type of application. Here the data size is chosen as 8 bits with a 9600 baud rate. For proper synchronization, 1 start bit and 1 stop bit is inserted between each frame. The algorithm for this module is given in Algorithm 3.

Algorithm 3: Module ST (ck,s,txd,t,c,rst)

```
Initially set txd=1'b1;t=1'b0;
Set local variables u=1'b0;v=1'b1;
always@(ck,rst)
if(rst= = inactive)
check if ((u= =1'b0)&&(c= =1'b1)
txd=1'b0;t=1'b1;u=u+1;
else if (((u>=1)&&(u<=8))&&(v<=24)&&(c=1'b1))
txd=s[v];u=u+1;v=v+1;
else if ((u= =9)&&(v<=24)&&(c= =1'b1))
txd=1'b1;u=0;
else if ((u= =9)&&(v= =25)&&(c= =1'b1))
t=1'b0;u=u+1;
else if ((u= =10)&&(v= =25)&&(c= =1'b1))
u=0;v=0;
end module
```

In Algorithm 3, local variable u is used as pulse counter; v is used as control input for serial transmission (each frame) in the module respectively; t and c are the same as those of the previous case. Here txd refers transmit port. Initially, txd port is in high impedance state and no such transmission takes place. Before transmission, txd port must be brought into active low which indicates that the transmitter is ready for transmitting data serially under clock control with a baud rate of 9600Hz. For one complete frame transmission, the expected value of v is 24. Once it is reached, once again txd line is brought into active high state in order to enable the next frame which is to be transmitted and pulse count variable u and serial transmission control bit v are set into initial states.

Serial receiver module: This module is designed in such a way that it can receive data frame one by one under the same clock (used in transmitter side) for proper synchronization in order to avoid any data losses. Each sensor node is able to transmit and receive data bidirectionally like full duplex mode. Before reception, the receive port (rxd line) is in high impedance state. The algorithm for this module is given in Algorithm 4.

Algorithm 4: Module SR (ck,R,rxd,r,rst)

```
Set local variables r=1'b0; y=0;z=1;
if(rst= = inactive)
check if ((y= =0)&&(rxd= =1'b0))
y=y+1;
else if ((y>=1)&&(y<=8)&&(z<=24))
R[z]=rxd;y=y+1;z=z+1;
else if ((y= =9)&&(z<=24))
if(rxd= = 1'b1)
                    y=0;
else if ((y= =9)&&(z= =25))
r=1'b1;y=y+1;
else if ((y= =10)&&(z= =25))
y=0; z=0; r=1'b0;
end module
```

In Algorithm 4, local variable y is used as pulse counter and z is used as control input for serial reception (each frame) in the module respectively; r is a valid output indicator when one entire frame has been received for every transmission. Here rxd refers to receive port or pin. Initially r×d port is in high impedance state and no reception takes place. When start bit is received via rxd pin, pulse count value is incremented by one. Following this, under the clock control with a baud rate of 9600 Hz, the transmitted frame is received via r×d pin and correspondingly z value is incremented. For one complete frame reception, the expected value of z is 24. Then, the valid output indicator r is set into logic high in order to indicate that the output that appeared in port R is a valid one; after that pulse counter control input and valid output indicator are set into the initial states in order to receive the successive frame transmissions.

Analyzer and reproducer module: This module is designed in such a way that to verify the integrity of the received frame. To carry out this, AR module has been designed. The algorithm for this module is given in Algorithm 5.

Algorithm 5: Module AR (ck,rst,D,r,R,n)

```
Initially set local variables p=3'b100;id[0]=9'b001001000
id[1]=9'b001001100
n=1'b0; Z=23'b000…0; I=0; K=3 where I and K are positive integers
if(rst= = inactive)
Check if ((I= =0) &&(r= = 1'b1) go to next step
Check if ((R[11:3]= = ID[0] || (R[11:3]= = ID[1]))
Z=R; I=I+1; else I=0
Check if ((I>=1)&&(I<=7)), Then perform
$P[2]^{+}$ = p[2] + p[0] + Z[K+2]; $p[1]^{+}$ = p[2] + Z[K+1]; $p[0]^{+}$ = p[1] + Z[K]
where + sign represents exclusive or operation. Then, increment K=K+3 and
I=I+1; Also assign P=p
Check if (I= =8) go to next step
Check if (P= = R[2:0]) assign D=Z[23:12]; Then, increment I=I+1; Else
Assign n=1'b1
Check if (I= =9), clear I=0; K=3
```

In Algorithm 5, local variable p is used as initial MISR value; id[0],id[1] are stored node ID values; n is a wrong output indicator; Z is an 24 bit temporary register;

I is a pulse counter and K is a message frame counter. During the active reset condition, initial values keep as such; when reset is inactive, check the r value; If it is one, consider that the R output is a valid one; Then, extract 12 bit node ID from the received frame so as to compare that ID with the already stored authenticated node ID values in order to validate the node authentication. Then, the entire received frame is passed into MISR circuit which is shown in Fig. 2 in order to generate the new signature in AR module. Finally, the new signature generated in AR module is compared with the initial signature value generated by the CSG module in order to validate the message integrity. Suppose, this match is not correct, the wrong output indication bit n is set into logic high and in turn, the entire received message frame is also ignored.

RESULTS AND DISCUSSION

Design and implementation: To develop the algorithms, we used Verilog Hardware Description Language (Verilog HDL) and to verify the functionality of the algorithms, we used Active HDL simulator. To implement these algorithms on Cyclone II FPGA chips, we used Quartus II software and DE2 development boards. The DE2 package contains all components needed to use the DE2 board in conjunction with a computer that runs the Microsoft Windows software. Quartus II is a programmable ASIC support Integrated Development Environment (IDE) software tool which includes synthesis, RTL viewer, Fitter and Assembler and SRAM object file generation tool sets. These tool sets are helpful to process our design so as to make the design compatible to implement on Cyclone II FPGA chips for initial prototype IC design. To develop the final design block, we applied the top to bottom design hierarchy. In this approach, the sub blocks are first designed individually using behavioral modeling approach and its own functionality was verified using Active HDL simulator. Second, the sub blocks are used to construct the top level design block using structural modeling and its integrated functionality was verified using Active HDL simulator. The primary mechanism for modeling the behavior of a design, behavioral modeling is used and to concatenate the design blocks together to form the top level entity, structural modeling is used. The simulation settings of DCSG and DCSV schemes are shown in Table 1 and 2, respectively. The simulated graphs of DCSG and DCSV schemes are shown in Fig. 3 and 4.

After the functionality verification of DCSG and DCSV schemes, the design files are processed using Quartus II synthesizer tool to convert design blocks into gate level net list which contains gate level components and its interconnections and then it is applied into RTL viewer tool. This tool accepts gate level net list as an input file and produces RTL (Register Transfer Logic) diagrams as an output which are shown in Fig. 5 and 6. Then, the gate level net list files of DCSG and DCSV blocks are applied into fitter and assembler tool which converts the design blocks to be compatible with FPGA resources called CLB's (Configurable Logic Blocks) and the synthesis reports of DCSG and DCSV blocks are generated. Then, the pin assignment details of DCSG and DCSV modules are given as User Constraint (UC) file to the pin assignment editor tool in order to assign the I/O pins of the designed block. Finally, the entire design file is converted into SRAM Object file (SOF) by using program device tool which is downloaded into Cyclone II FPGA chip placed in DE2 development board shown in Fig. 7.

Table 1: Simulation settings of DCSG scheme

Signal name	Initial value
id[8:0]	9'b001001000
d[11:0]	12'b010000010100
i,k,c	1'b0
q[2:0],Q[2:0]	3'b100

Table 2: Simulation settings of DCSV scheme

Signal name	Initial value
n	1'b0
id[8:0]	9'b001001000
r	1'b0

Test result analysis: To evaluate the increase in speed provided by this implementation, execution cycle counts are compared with the conventionally used integrity algorithms like SHA-1 and MD5 with processor based implementation. In addition, embedded C code for DCSG and DCSV scheme were also written and executed in ARM7TDM, MSP430 and Atmega128 microcontrollers in order to evaluate the speed enhancement that has been achieved in FPGA based implementation. From this, how fast a task can be executed can be observed by using the clock frequency. In addition, the verilog codes for SHA-1 and MD-5 algorithms were also implemented in FPGA so as to evaluate how DCSG and DCSV scheme is better than other schemes. As a whole, it is identified that FPGA based implementation of DCSG and DCSV module enhances the execution speed when compare to the processor based solution and it is shown in Table 3.

However, the energy consumption comparison is not carried out in this study. Because FPGA based sensor node is mainly for improving timing efficiency to meet time

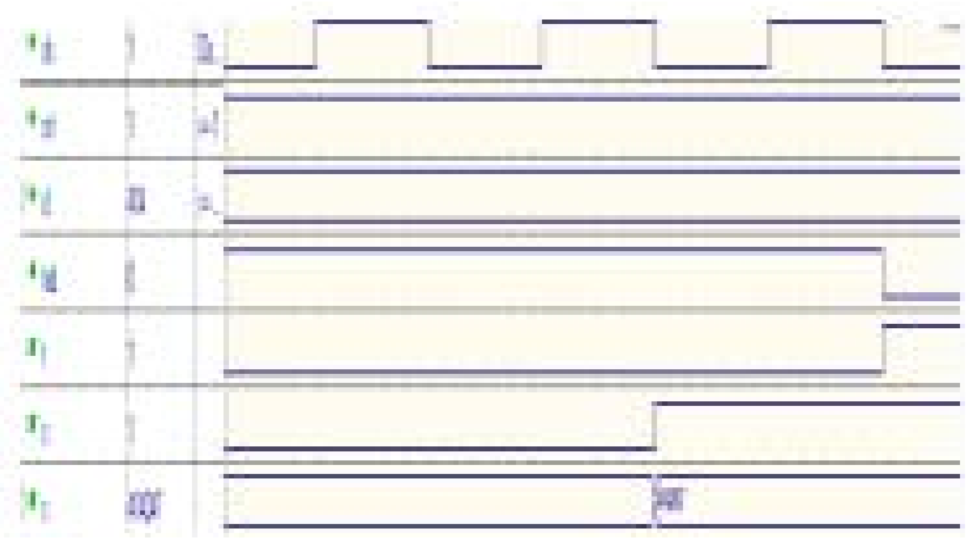

Fig. 3: Simulated result of DCSG scheme

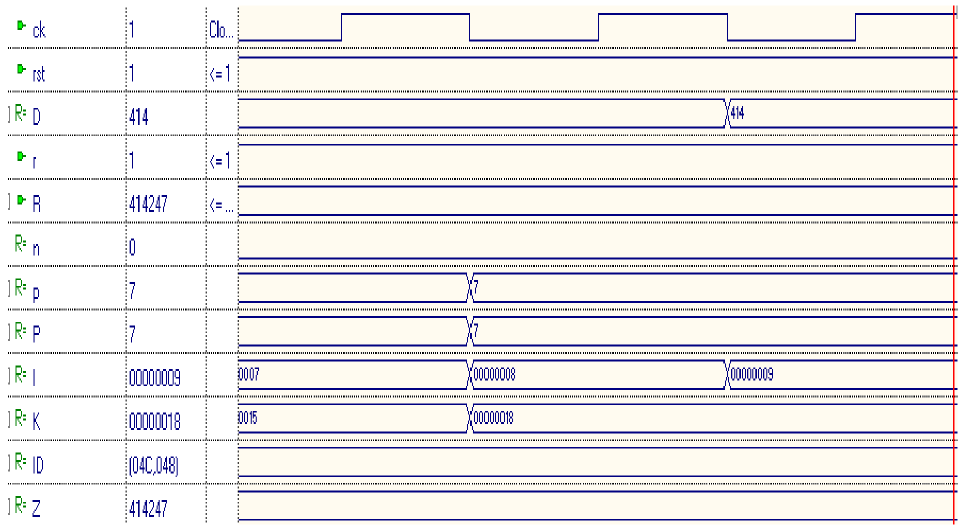

Fig. 4: Simulated result of DCSV scheme

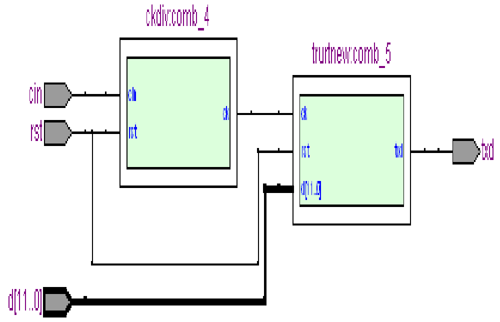

Fig. 5: RTL view of DCSG block

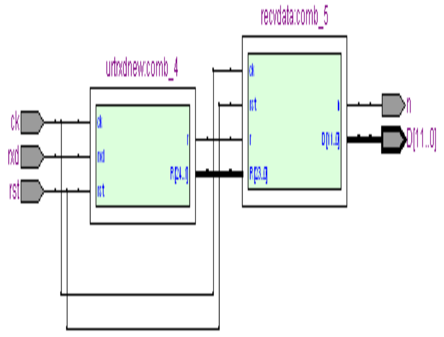

Fig. 6: RTL view of DCSV block

Table 3: Number of clock cycles count for different algorithmic implementations number of clock cycles count

Algorithm	ARM7TDM microcontroller based implementation	MSP 430 microcontroller based implementation	Atmega128 microcontroller based implementation	FPGA based implementation
SHA-1	136	143	189	71
MD-5	128	159	204	84
DCSG and DCSV scheme	96	114	152	53

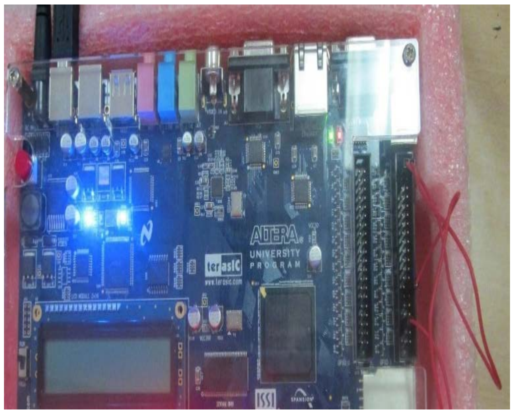

Fig. 7: FPGA Implementation of DCSG and DCSV scheme

and mission critical tasks in wireless Sensor Networks (WSN). When low power consumption is required simultaneously with greater speed, emerging low power FPGAs can be used to extend the lifetime of batteries on the WSN nodes.

CONCLUSION

This study presents an approach of message digest scheme for FPGA based event-driven architecture by using maximum length sequence primitive polynomial approach which incorporates DCSG and DCSV modules. The experimental result demonstrates that the performance of this approach is superior than processor based solution in terms of speed enhancement for real-time mission critical applications.

REFERENCES

Chong, C.Y. and S.P. Kumar, 2003. Sensor networks: Evolution, opportunities and challenges. Proc. IEEE, 91: 1247-1256.

Dener, M. and C. Bostancýoglu, 2015. Smart technologies with wireless sensor networks. Procedia Soc. Behav. Sci., 195: 1915-1921.

Jariwala, V., V. Singh, P. Kumar and D.C. Jinwala, 2014. Investigating approaches of data integrity preservation for secure data aggregation in wireless sensor networks. J. Inf. Secur., 5: 1-11.

Krasteva, Y.E., J. Portilla, D.L.E. Torre and T. Riesgo, 2011. Embedded runtime reconfigurable nodes for wireless sensor networks applications. IEEE. Sens. J., 11: 1800-1810.

Kumar, V., A. Jain and P.N. Barwal, 2014. Wireless sensor networks: Security issues, challenges and solutions. Int. J. Inf. Comput. Technol., 4: 859-868.

Liao, J., B.K. Singh, M.A. Khalid and K.E. Tepe, 2013. FPGA based wireless sensor node with customizable event-driven architecture. EURASIP. J. Embedded Syst., 2013: 1-5.

Lu, H., J. Li and M. Guizani, 2014. Secure and efficient data transmission for cluster-based wireless sensor networks. IEEE. Trans. Parall. Distrib. Syst., 25: 750-761.

Okobiah, O., S.P. Mohanty and E. Kougianos, 2014. Nano-CMOS thermal sensor design optimization for efficient temperature measurement. Integr. VLSI. J., 47: 195-203.

Sun, H., Y. Qian and R. Yan, 2012. Design and realization of an intelligent sensor node with its application in energy-aware WSNs. Proceedings of the 2012 IEEE International Conference on Instrumentation and Measurement Technology (I2MTC), May 13-16, 2012, IEEE, Nanjing, China, ISBN: 978-1-4577-1773-4, pp: 941-946.

Sun, X., J. Su, B. Wang and Q. Liu, 2013. Digital watermarking method for data integrity protection in wireless sensor networks. Int. J. Secur. Appl., 7: 407-416.

Vece, G.B., M. Conti and S. Orcioni, 2015. Transaction-level power analysis of VLSI digital systems. Integr. VLSI. J., 50: 116-126.

Zhang, X., H. Jiang, L. Zhang, C. Zhang and Z. Wang *et al*., 2010. An energy-efficient ASIC for wireless body sensor networks in medical applications. IEEE. Trans. Biomed. Circuits Syst., 4: 11-18.