Security Wrappers and Power Analysis for SoC Technologies

C.H. Gebotys, Y. Zhang
Dept of Electrical and Computer Engineering
University of Waterloo
(519)885-1211
cgebotys@uwaterloo.ca

ABSTRACT

Future wireless internet enabled devices will be increasingly powerful supporting many more applications including one of the most crucial, security. Although SoCs offer more resistance to bus probing attacks, power/EM attacks on cores and network snooping attacks by malicious code are relevant. This paper presents a methodology for security on NoC at both the network level (or transport layer) and at the core level (or application layer) is proposed. For the first time a low cost security wrapper design is presented, which prevents unencrypted keys from leaving the cores and NoC. This is crucial to prevent untrusted software on or off the NoC from gaining access to keys. At the core level (application layer) power analysis attacks are examined for the first time for parallel and adiabatic architectural cores. With the emergence of secure IP cores in the market, a security methodology for designing NoCs is crucial for supporting future wireless internet enabled devices.

Categories and Subject Descriptors

C.3 [**Special Purpose and Application Based Systems**].

General Terms: Security

Keywords: Design, Security, Performance, VLIW, Adiabatic.

1. INTRODUCTION

Future nanometer technologies will support large SoC or 3D ICs with a mix of architectures and technologies. A large number of heterogeneous cores on the SoC may be application specific. For example security cores already exist on the market and will be crucial in future wireless SoCs. Also cores may have varying degrees of parallelism or may be implemented in various technologies including adiabatic circuitry to significantly decrease core energy dissipation. Design methodologies must be developed for security on SoCs.

As networks move to the chip level, increasing the complexity of the system on a chip (SoC), security increases in importance. Architectures must be designed which protect the user's private key from attackers (in particular untrusted software resident from invasive computing or untrusted external devices such as EM reading devices or hostile IP hosts). In general it will be of primary importance to additionally protect the secure IP cores from untrusted software and prevent exposure of keys outside of the SoC and even within the SoC communication network. Although there is new research studying networks on chips (NoC) [2], there is a lack of research studying how security would be integrated into these SoCs. Furthermore outside of smartcard research[6,7] (which typically is limited to cheaper 8-16 bit processors) few researchers have examined secure implementations of cryptographic software under the threat of power attacks on highly parallel or adiabatic cores. Design for security involves secure design of hardware cores, secure algorithm design, and secure network protocols, all important for NoCs of the future.

This paper illustrates a general security methodology for NoCs. At the processor core level, security cores must be resistant to power/EM attacks. At the network level, a security wrapper supporting symmetric key cryptography is implemented for communication between IP cores on the NoC which are involved in security applications. A security wrapper for each core and a central key-keeper core are proposed to ensure that unencrypted keys do not leave any core and the NoC itself. Security for NoCs has not previously been studied, however this approach has advantages for IP core vendors providing further protection of not only their hardware IP but also software which will run on their IP core. At the core level the impact of parallelism and adiabatic circuits on power/EM analysis is illustrated for the first time. This paper presents a security methodology for NoCs at both the network and core level which is crucial for future nanometer technologies.

NoC research has emerged as an important and growing area of interest. Researchers are developing tools to customize or reprogram the communication network design[2] as well as investigating reliability issues. Network layers for NoCs have been suggested in [2], specifically: the physical layer, architecture and control layer (data link, network and transport) , and software layer (application and system). NoCs in general will be highly constrained by reliability in addition to power dissipation and error correcting codes will be a necessity[2]. However security or security protocols have not been researched for NoCs.

Researchers have examined ASIC implementations of various encryption algorithms, system architectures for wireless security[5], and recently a number of secure IP cores have emerged such as AES cores, SHA-1, 3DES cores and others. IP

CODES+ISSS'03, October 1–3, 2003, Newport Beach, California, USA.

core protection includes both hardware protection such as watermarking techniques and simulation with software module IP protection in a CAD design environment such as [8]. In the later case public key cryptography is used to support simulation of chips which include software IP modules as well as hardware IP cores. A JavaCAD environment is described for designers which protects IP yet simulates designs which include hardware IP cores running third party IP software modules for example.

Power or electromagnetic (EM) attacks of NoCs may be relevant since cores may have separate power pins and EM radiation from the communication network may be significant. Researched power attacks of smart cards, have utilized general purpose processors with low clock frequencies [1,6,7] and more sophisticated processors with parallel instruction execution have only recently been reported in the literature[11]. Researchers have suggested security against power attacks be achieved through 70% increase in computational cost[3]. However in portable devices, energy dissipation is very important hence a general purpose processor not optimized for power dissipation will not be suitable for running encryption in wireless communication[5].

Thus a methodology for security is crucial for future NoC designs and a necessity for designing with secure IP cores available on the market today. This research presents a security methodology at the network level (or transport layer) and at the core level (or application layer). At the network level the security design utilizes a special key-keeper core and a network key on the NoC. This security architecture is independent of the type of communication network on the chip in general. The security methodology ensures no unencrypted keys leave cores or the NoC and additionally supports IP secure cores running only trusted software. At the core level (or application layer) differential power analysis is demonstrated and the impact of parallelism and adiabatic circuits on security is analyzed. The next section will describe the security methodology for NoCs at the communication network level, followed by security methodology at the core level in section 3, specifically for software running on a VLIW core and an adiabatic circuit.

2. SECURITY FOR NoC

In NoCs the communication network may vary significantly from packet based communication to global bus communication[2]. The NoC will be required to support security for several reasons such as: 1) wireless communication or IP enabled applications requiring user authentication, encryption; 2) product authentication

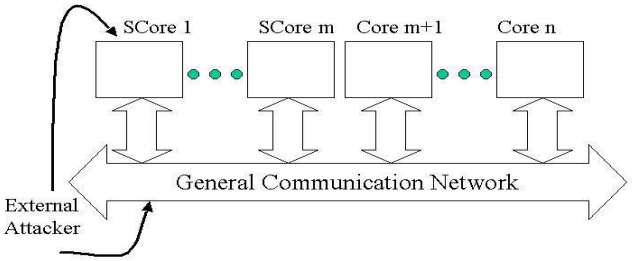

Figure 1. An example of a general NoC and security attacks.

(preventing counterfeit products); 3) NoC chip authentication; 4) even IP-core authentication on the NoC itself. We will assume NoCs have *m* secure cores (or SCores in figure 1) and another *n-m* other cores, where *m≥1*. A secure core is defined as a hardware IP core which can execute one or more security applications (such as encryption, authentication, key exchange, etc) and it may also be able to execute other general non-security application as well. This NoC scenario suits current secure IP cores on the market today as well where authentication and encryption requires several secure cores such as an AES core, a SHA-1 core, etc. The NoC should in either case be resistant to external attacks which can obtain data from the communication network (through external I/O pins attached to the communication network or EM radiation from it), shown as bottom arrow in figure 1 or through power analysis attacks on individual core power pins (see top arrow figure1). Key distribution is important, since keys are updated from time to time, the NoC will require a mechanism to allow downloading of the key onto the chip and into the memory of the secure IP core. Furthermore malicious software running on a regular core (Core) could attempt to acquire keys or send false keys to security cores. Protection against malicious software and untrusted cores must be supported in the NoC. The proposed security methodology at the network level will be described next. It supports key protection, and secure transmission on the communication network.

The network level methodology for security is based on symmetric key cryptography. A secure core can be a AES core, SHA-1 core, a prime field operation elliptic curve core, etc… Each secure core (*i*) has its own security wrapper, see shaded (*Kn*) boxes in figure 2. The security wrapper is located between the secure core and the network wrapper (see figure 2).

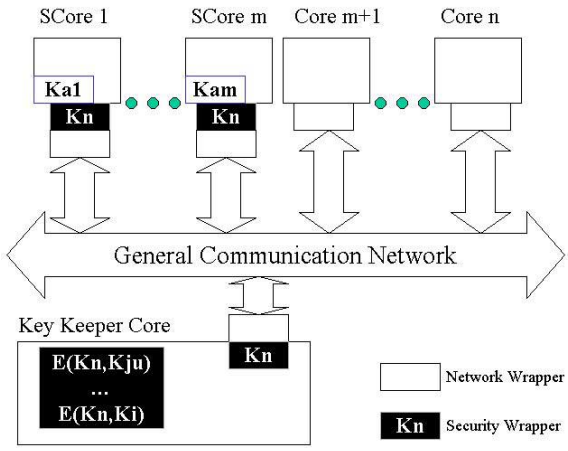

Figure 2. Security methodology at network level protecting keys.

The keys stored in the security wrapper are 1) the network master key, *Kn*, and 2) the working key *Kn'* (generated from the master key *Kn*), both stored in non-volatile memory within the security wrapper and 3) message authentication code (MAC) keys, *Kmac i* and *Kmac k* (key keepers mac key), are also stored. Since SCores may often receive keys from the key keeper (ie. user keys for encryption, or authentication), the key keeper's mac key (*Kmac k*) is stored for convenience in the security wrapper for quick

authentication. If sufficient memory is available and security requirements are high it is possible that mac keys of other SCores could be also stored in the security wrapper. Inside the secure core an authentication key (*Kai* in figure 2) is stored in non-volatile memory, for use in core authentication and core software authentication. The security wrapper performs a number of functions including 1) encryption of messages from the secure core to the network E(.), 2) decryption of messages from the network to the secure core D(.), 3) create a hash of a message h(.), and 4) create a MAC mac(.). The hardware of the security wrapper will be described later.

A key-keeper core in the NoC (see figure 2) is responsible for secure key distribution on the NoC. This core is also a secure core (and therefore has its own security wrapper). It stores encrypted keys, such as encrypted keys for individual applications, encrypted user private keys (*Kju*) (ie. authentication or encryption keys to be used for decryption of audio, video, email, ebanking or signature generation or authentication of outgoing messages, etc), other users public keys (*Ki*) (ie. for encryption of outgoing audio, video, email, signature verification, etc) and SCore MAC keys (*Kmac i*). The key keeper is also responsible for updating the master network key (*Kn*) at random times using a symmetric key exchange protocol.

The following terminology will be used to describe the functionality of the security in the security wrapper: E(a,m) represents encryption of *m* using key *a*; D(b,c) is decryption of *c* using key *b*; h(m) is a *x*-bit hash of a message *m*; and MAC(k,m) is a *n*-bit MAC of message *m* using key *k*. Also for SCore i, let $tcnt_j$ represent the count of how many messages were sent to SCore j, $rcnt_j$ represent how many message were received from SCore j and ‖ represent concatenation. Consider sending a message from SCore i to SCore j. The following three steps are involved 1) generate a working key (since it is well known that one should not use the same key to transmit different messages), 2) Encrypt the message, and 3) provide authentication (step 3 below) or integrity (step 4 below) by attaching a tag to the encrypted message. These three steps are performed within the security wrapper of SCore i and are illustrated below:

1. $Kn' = h(Kn \| tcnt_j)$
   $tcnt_j$ ++
2. $c = E(Kn' , m)$
3. $t = mac(Kmac\ i , c)$
   Send $c \| t$
4. or
   $i = h(c)$
   Send $c \| i$

Only SCore's have access to the mac keys of SCores. Hence an untrusted core cannot obtain the mac keys. It is assumed that the SCores only run trusted software (and do not have any malicious codes). The security methodology tries to minimize the chance that the untrusted cores may be able to obtain the network master key, through changing it at random times. However even if the master network key is obtained, the working keys and mac keys would be difficult to obtain. SCore j will receive the message ($\{c \| i\}$ or $\{c \| t\}$) by performing the following steps inside the security wrapper.

5. $Kn' = h(Kn \| rcnt_i)$
   $rcnt_i$ ++
6. $m = D(Kn' , c)$
7. $t1 = mac(Kmac\ i , c)$
   If t1 == t accept else request retransmission
8. or
   $i1 = h(c)$
   if i1 == i accept else request retransmission

This basic protocol, for exchanging messages, supports key transfer from the key keeper to SCores, the updating of new authenticated keys which have been sent to the portable device, and others. For example using this basic protocol new authenticated user keys can be authenticated using authentication SCores, decrypted using AES SCores, and reencrypted with working network keys and sent back to the key keeper for storage. Each transmission on the network ensures unencrypted keys or messages are never sent between SCores. Hash tags ensure that the data has not changed during transmission. MACs attached to encrypted mesasge are used to verify that the encrypted message came from a trusted source.

One possible suggestion for hardware of the security wrapper will be presented next. The hardware to be described is a very low cost approach to supporting encryption, hashing and mac generation. The security wrapper will contain LFSRs (linear feedback shift registers), general shift registers, counters, XOR gates and small amount of non-volatile memory for storage of the working key, and the mac keys. The following terminology will be used to describe the hardware: LFSR(s) represents a 1 bit output from a LFSR with seed *s* (or it is clocked for *y* cycles if a *y*-bit output is implied in the formula); $\{\}_{m-x\ ..\ m}$ represent the last *x* bits of a *m*-bit quantity in brackets; let + represent exclusive or operation; and let m_i represent bit i of M. Now the hardware defining the functions of the security wrapper will be given:

$$E(k , m) = C \mid c_i = m_i + [LFSR(k) + c_{i-m}]$$

$$D(k , c) = M \mid m_i = c_i + [LFSR(k) + c_{i-m}]$$

$$mac(k , m) = E(k , m)_{m-n\ ..\ m}$$

$$h(m) = LFSR(m)_{m-x\ ..\ m}$$

The LFSR is used as the low cost key stream generator. The mac() or h() will identify if the message was modified in transit (due to noise or reliability problems on NoC). The LFSR streaming cipher is used in cipher feedback mode which is ideal for NoC's since synchronization errors are recoverable. Furthermore cipher text errors affect only the corresponding bits in the plaintext in the design above, hence minimal retransmission is required for correcting errors.

The internal SCore authentication key (key located in SCore, not in security wrapper) which has not been used so far is important for a number of security purposes. For example secure cores could receive new software upgrades (wirelessly) from their IP core vendors. Since only trusted software is assumed to be running on security cores, this new software would have to be authenticated. This is performed by sending to the NoC executables authenticated with the internal private authentication key of the secure core, *Kai* in figure 2, which only the vendor of the IP core can do since they are the only ones who know the secure core's

private key, *Kai*. Again this upgrade can be performed remotely or wirelessly. Authentication (for example with crypto checksums to ensure all security code executables are authenticated) could occur within the SCore (not the security wrapper) to safeguard the private SCore authentication key.

Additionally the vendor of the IP core can use the core's private authentication key to prevent illegal use of the core in unauthorized NoCs (which have not paid license fees etc for use of the core). The core vendors would create an activation key *(K_activation*) for their core to be used in the NoC from the cores Kai and the IDs of the other cores in the network. Legal uses of the core would receive this activation key from the core vendor and store it encrypted in the key keeper core. For example on reset of the NoC the secure core would receive an activation key (*K_activation*) from the key keeper and verify this key by calculating its own security function (F(), which would be encryption for an encryption core, etc) with its private key (*Kai*) and the concatenation of all other core IDs (obtained from polling all other cores) on the NoC. For example if *K_activation* = = F(*Kai*, core ID 1‖ core ID 2 ‖…core ID n), then the secure core will function properly else it will not operate and shut down permanently. Standards would have to exist so that each core would have a core ID number. Ensuring confidentiality of communications on the on-chip network, and authenticity of executables which utilize private keys, a secure SoC environment for security applications is possible. Next the core level security will be addressed illustrating how different architectures within a core can affect the security (specifically susceptibility to differential power attacks).

3. SECURITY FOR CORES

3.1 Application Layer

The previous section provided a methodology for ensuring that key's are safe within the communication network. Security within the secure IP cores is also important. Since cores will typically have their own power pins on SoC chips, differential power analysis (DPA) attacks on cores will be relevant. The cores typically may be application specific and vary significantly with respect to architecture. Cores may also be optimized for power. For example application-specific cores which dissipate large amounts of energy may have to use adiabatic architectues to avoid hotspots on the NoC chip. Specifically adiabatic circuits will be briefly introduced in section 3.1 followed by an introduction to differential power analysis in section 3.2. Finally results of DPA applied to both parallel architectures and adiabatic architectures are presented in section 3.3.

3.2 Introduction to Adiabatic Circuits

Adiabatic CMOS logic circuits have been reported for their potentials for low-power VLSI applications. Early adiabatic logic circuits used diodes or diode-like devices for the purpose of precharge, which caused the unavoidable energy loss due to the threshold voltage of the diodes. Later on, the cascode voltage switch logic concept was adopted to get rid of the diode from the circuit. An efficient charge recovery logic was proposed based on this concept in which the diodes are eliminated. But the outputs were still not full-swing due to the existance of the threshold voltage of two switching PMOS devices, resulting in non-adiabatic dissipation. After that, a full swing energy efficient logic (eel) [9] circuit was reported to give out a better performance, but the clock system was very complex (it needed eight clocks: four ramp-like power clocks and four pulse clocks) and therefore not practical. A novel adiabatic differential switch logic (adsl) with bootstrap technique introduced in [10] achieved better noise immunity, higher operation frequency and less energy dissipation. The conventional dynamic logic (con), full swing energy efficient logic (eel) and adiabatic differential swing switch logic (adsl) will be power analyzed for security purposes in section 3.3. To illustrate how secure these varying architectures may be, differential power analysis, DPA, is used. The next section will introduce DPA.

3.3 Introduction to DPA

The DPA attack which correlates power to data placed on the bus utilizes the following variables: $C_i(t)$ represents the instantaneous power at time t of the ith power trace, where $i=1,...,n$; and j is the bit of the data placed on the bus being attacked in the ith execution of the algorithm. The attack makes a guess on how the data was transformed (for example in DES the attacker guesses the 6bit key). Since the attacker knows the input data used to generate each power trace (the plaintext in DES), the attacker can determine the data value computed resulting from the guessed computation. With the guess the attacker partitions the power traces into two groups according to bit j of the guessed data value (or data being attacked in DES which is output of Sbox). One group contains all power traces where bit j, or b_j, of the guessed data value on the bus is equal to 0 and the other group contains all power traces whose bit j of the guessed data value is equal to 1. Let $x_1(t) = \{C_i(t) \mid \forall i = 1,..n, b_j = 0\}$ $x_2(t) = \{C_i(t) \mid \forall i = 1,..n, b_j = 1\}$ be the two groups of power traces. Then the difference of means or differential power trace is :

$$\overline{x_1(t)} - \overline{x_2(t)}$$

The standard deviation of the difference of means is defined as :

$$\sqrt{\frac{\sigma_1(t)^2}{n1} + \frac{\sigma_2(t)^2}{n2}}$$

If the guess made was correct there would be a DPA characteristic or "peak" in the differential trace, otherwise the guess would be wrong. The DPA characteristic (or peak) is significant if it is much greater than the standard deviation of the difference of means. In this paper we use 2 standard deviations to test the significance as given below:

$$\overline{x_1(t)} - \overline{x_2(t)} > 2\sqrt{\frac{\sigma_1(t)^2}{n1} + \frac{\sigma_2(t)^2}{n2}}$$

In DPA attacks on DES applications, the largest DPA characteristic from 64 differential traces is chosen to determine which of the 6 bits of the key were the correct guess. However in other public key cryptographic attacks since only one differential trace is used to guess the first key bit, the above formula is important in helping to determine if the guess was correct or not.

3.4 DPA Results: parallelism, adiabatic

DPA results using real power measurements and HSPICE power measurements are analyzed in this section. Real power measurements of a VLIW processor core[4] is utilized for analysis

of the impact of parallelism on DPA. Three adiabatic circuits are analyzed with HSPICE to study their DPA characteristics. For the adiabatic circuits, a 4-bit pipelined carry lookahead adder (CLA) is used along with a bus and small memory to analyze susceptibility to power attacks. Matlab was used to analyze the power traces and perform the DPA calculations. The DPA peaks are treated as significant if their height exceeds two times the standard deviation of difference of means presented in section 3.2.

A program (ILP 1) was created with single instruction execution sets (no parallelism) using only load data instructions. The program was executed 56 times (each time recording the power trace) and only one data word loaded by 3 pairs of load instructions was changed. Each of the 56 power traces was an average of 50 power traces (for a total of 2800 power traces). The first two instructions were executed near 2.4us, the next two were executed near 2.9us and the last two instructions were executed near 3.5us. The power trace is shown at the top of figure 3. All three DPA peaks are significant as illustrated in figure 4 by taking the differential trace and subtracting the two standard deviations (data above x-axis indicates significance).

The parallel program (ILP 5) utilized all parallel execution sets with 4 ALU (logic and arithmetic) instructions and 1 load instruction per cycle. The program was executed 62 times (and power was recorded) each time with different data being loaded in 2 pairs of load instructions. Each power trace was an average of 50 power traces. One power trace, the differential trace of a correct guess of data bit 0 and the differential trace for an incorrect guess of data bit 0 are shown at the bottom of figure 3. The first pair of load instructions occurs near 1.6us and the next pair near 2.2us. The DPA characteristics were 0.0287mA and 0.025mA, lower than the ILP 1 characteristics at the top of figure 3. However both DPA characteristics were also significant as seen in figure 5, where again the differential minus the two standard deviations are plotted.

The DPA of the three adiabatic circuits (adsl, eel, con) are illustrated in figure 6, all shown with same y-axis range (0.06 to -0.02mA). The adsl and eel adiabatic circuits clearly showed a lower DPA characteristic than the same circuit using conventional dynamic logic. Also the adsl dissipates 35% less energy than eel and 68% less energy than con. In the adiabatic circuits the DPA peaks are significant with respect to two standard deviations.

Table 1 compares the DPA characteristics and circuits with previous research. The supply voltage (V), frequency (f in MHz), maximum standard deviation of all power traces together (σ), DPA characteristic (pk), and maximum amplitude of incorrect differential (inc) are listed in the table. The adiabatic circuit characteristics are also listed in this table, however no direct comparison with previous research smartcard processor [6,7] or the parallel processor core (ILP 1, ILP 5) would be applicable since the DPA was only performed on the 4bit ALU circuit. Nevertheless comparisons between adsl, eel and con, the two adiabatic circuits versus conventional logic can clearly be made since all circuits were identical except for the circuit technology. Clearly, as supported by figure 6 the adiabatic circuits have smaller DPA peaks than conventional logic (con) which is advantageous for security.

4. DISCUSSION/CONCLUSIONS

This paper presented for the first time a methodology for security on NoCs. It presented a symmetric-key based cryptography design for securing the communication network within the NoC. Authentication, encryption, key exchange, new user keys, public key storage, etc are supported in this methodology. Additionally there are benefits to the IP core vendor for this setup including more protection of software running on their hardware IP core (a solution to untrusted or invasive software). Another advantage of this new methodology is that copying IP cores without the activation keys (provided by vendors) will prevent their illegal use in SoCs. The symmetric key system simplifies the NoC design

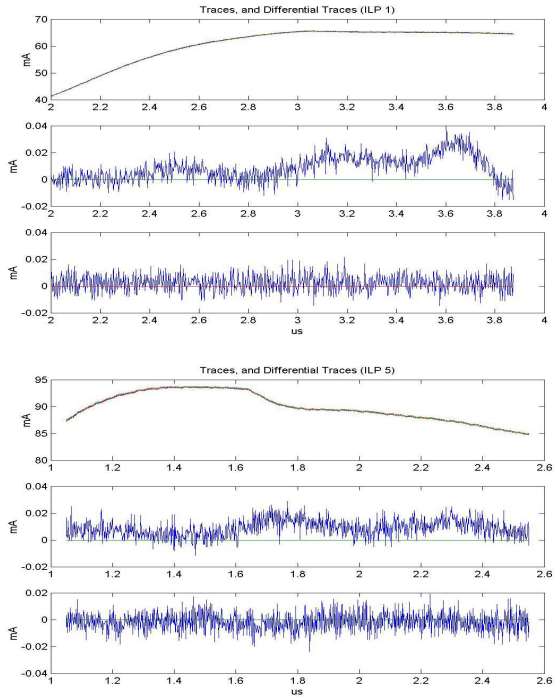

Figure 3 Power traces, differential and incorrect guess for ILP of 1 (top) and 5 (bottom).

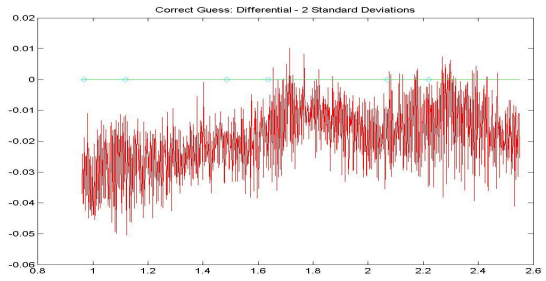

Figure 4. Differential minus two standard deviations for ILP=1.

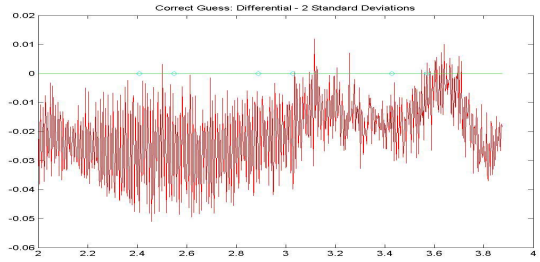

Figure 5. Differential minus two standard deviations for ILP=5.

Table 1. Comparison of DPA results.

	[6]	[7]	ILP 1	ILP 5	con	eel	adsl
V	-	5	2	2	0.8	0.8	0.8
f	3.57	4	100	100	200	200	200
σ	0.3	-	0.1	0.1	9.8E-4	6E-5	3E-5
pk	0.04	0.46	0.03	0.02	.05	.003	.003
inc	0.005	0.20	0.02	0.01	71E-4	3E-4	2E-4

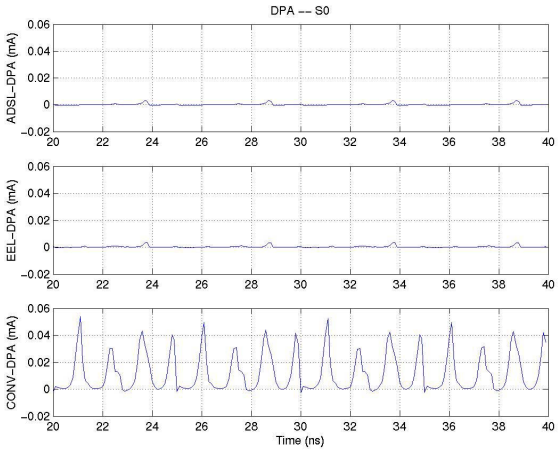

Figure 6. Spice Generated Differentials for adsl, eel, con (top to bottom).

process and key management for security in SoCs. It also helps to protect hardware IP (ie. preventing unauthorized access to it) as well as software IP (ie. preventing modification of software IP through authentication). If user keys were not expected to change, each secure core could also have copies of these keys in local non-shared nonvolatile memory (however this situation is very restrictive and costly, and future NoCs should support new user keys). Key management is still necessary since public keys would still be distributed and sent to secure cores. For added security multiple LFSRs could be used however since the network master key will be changed at random times, the attacks on the LFSR are limited.

The security wrapper could be standardized and implemented by the NoC designer or core vendor. Specific encryption and authentication algorithms and very low cost hardware (with low performance and power overheads) for the security wrapper have been described, however other designs such as RC4 could be used as well depending upon the NoC requirements (where all secure cores in the NoC would have the same secure wrapper). Since reliability is crucial in NoCs of the future its possible that combined decryption and error correcting would be highly advantageous (similar to the cipher feedback mode chosen in this paper). The security methodology could also be extended to support dynamic nature of NoCs where depending upon current battery energy levels and QoS of messages, secure cores could run at different clock frequencies, power levels, etc and operate different encryption algorithms (varying from weak to strong security). Additionally during reset, a new NoC key could also be generated using a pseudo random number generator with a key exchange algorithm to ensure further security.

This study presents for the first time a methodology for security of NoCs by providing network level symmetric-key cryptography for key distribution and at the core level by illustrating the impact of parallelism and adiabatic technology on DPA. Unlike previous research, which has not studied security on NoCs, this scheme has added advantages for protection of software and hardware IP. For core-level security, it is shown that architecture can have an important impact on security, making DPA more difficult (as illustrated by smaller DPA characteristics for parallelism and adiabatic technology). This research is crucial for supporting a methodology for designing security for NoCs which will be prevalent in wireless IP-enabled devices designed with nanometer technologies of the future. The authors would like to thank RIM, CITO, NSERC and Motorola for their support.

5. REFERENCES

[1] P.Kocher, "Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems", LNCS, 1998.

[2] L.Benini, G.DeMicheli, "Networks on Chips: A New SOC Paradigm" Computer, Vol.35, No.1, Jan 2002, pp.70-78

[3] P.Liardet, N.Smart "Preventing SPA/DPA in ECC systems using the Jacobi Form", LNCS 2162, May 2001, pp391-401

[4] "Star*Core 140 DSP Core Reference Manual", Motorola/Lucent, Sept 1999.

[5] S.Ravi, A.Raghunathan, N.Potlapally "Securing Wireless Data: System architecture challenges", ISSS, 2002, pp.195-200.

[6] P.Kocher, J.Jaffe, B.Jun "Differential Power Analysis" Crypto'99, LNCS 1666, 1999

[7] T.Messerges, E.Dabbish, R.Sloan "Investigations of Power analysis attacks on Smartcards" USENIX workshop on Smartcard Technology, 1999.

[8] M.Dalpasso, A.Bogliolo, L.Benini "Hardware /Software IP Protection", Proceedings of DAC, 2000.

[9] C.Yeh, J.Lou and J.Kuo, "1.5 V CMOS full-swingenergy efficient logic (EEL) circuit suitable for low-voltage andlow-power VLSI", Electronics Letters, 33(16), 1997, pp. 1375-1376

[10] Y.Zhang, H.Chen, J.Kuo, "0.8 V CMOS adiabatic differential switch logic circuit using bootstrap technique for low-voltage low-power VLSI", Electronics Letters, 38(24), 2002, pp. 1497 –1499

[11] C.Gebotys, R.Gebotys, "Secure Elliptic Curve Implementations: An analysis of resistance to power-attacks in a DSP processor", CHES 2002, LNCS 2523, pp114-128.