Military Technical College
Kobry El-Kobbah, Cairo,
Egypt

8th International Conference
on Electrical Engineering
ICEENG 2012

HANK-1 ,A new Efficient and Secure Block Cipher Algorithm for Limited Resources Devices

By

Hazem Mostafa Eldeeb[1] hazem.eldeeb@gmail.com

Khaled Ali Shehata[2] k_shehata@aast.edu

Nabil Hamdy Shaker[3] nabil.hamdy@miuegypt.edu.eg

Ahmed Ali Abdel Hafez[4] aabelhafez@gmail.com

[1,4]Egyptian Armed Forces
[2] Arab Academy for Science & Technology and Maritime Transport (AAST).
[3] Misr International University (MIU)

Abstract

In this paper we present a new block cipher algorithm that can be used for data security over devices with limited resources ,e.g. smart cards, wireless sensors etc .The algorithm is 128-bit balanced Feistel structure cipher algorithm working in cipher block chaining mode of operation. The building components of the algorithm have good cryptographic properties in comparison with other standard cipher algorithms and it has passed the NIST statistical test with very good results. The algorithm has been implemented on Microblaze microprocessor (as an emulator) to evaluate its efficiency and suitability to work on constrained devices.

Keywords:

Feistel Networks, Constrained environments, Block Cipher, Maximal distance separable codes, Substitution Box, Statistical Tests , Microblaze.

1. Introduction:

As cellular phones, wireless sensors, smart cards, and other limited resources devices become popular and widely used, the need for secure data transmission and processing over these devices becomes more and more important. Several cipher algorithms with different techniques have already been employed [10,12], and most of them have been broken which increases the need for higher levels of security cipher algorithms.

The main problem with limited resources devices is that the highly secured standard cipher algorithm that designed for processors with high resources cannot be used because of the constrained processing power, low memory space and low battery power. This paper gives a detailed description of a new block cipher algorithm which we believe it has the required high level of security and efficiency to be employed in constrained devices. The algorithm is Feistel network algorithm which use the same algorithm for both encryption and decryption processes. The reversibility of the algorithm is independent of the reversibility of the round function and it is ensured only by the Feistel structure itself.

2.HANK-1 General structure and specifications:

HANK-1 is 8-rounds, 128-bit balanced Feistel structure block cipher algorithm with cipher key length equals 128 bit. In the traditional Feistel cipher, plaintext is partitioned into two equal halves (that is why it is called balanced). The round function ***F*** is applied to one half using the round sub-key and the output of ***F*** is xored with the other half. The two halves are then swapped. Each round follows the same pattern except for the last round where there is no swap.

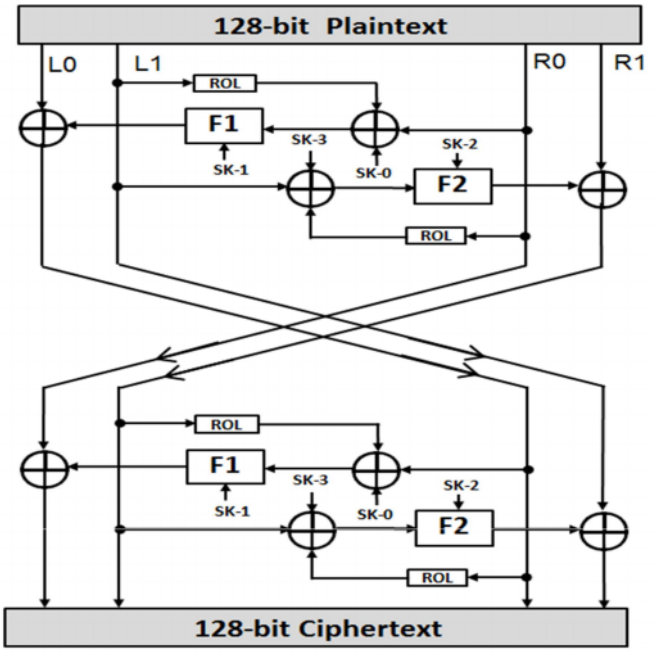

Figure (1) HANK-1 One Cycle

HANK-1 is adopting a different Feistel network technique, the algorithm could be seen as two traditional Feistel networks interrelated and interconnected to each other. The 128-bit plaintext block (P) is partitioned into four 32-bit sub-blocks (**L0, L1**, **R0**, **R1**)

such that (P) = L0‖L1‖R0‖R1. The round sub-key block (SK) is also partitioned into four 32-bit sub-blocks (SK0, SK1, SK2, SK3) such that (SK) = SK0‖SK1‖SK2‖SK3.
The plaintext block is processed according to the following equations:

$$L0_i = L0_{i-1} \oplus F1\left(SK1_{i-1}, R0_{i-1} \oplus SK0_{i-1} \oplus ROL(L1_{i-1})\right) \quad \text{Equation (1)}$$

$$R1_i = R1_{i-1} \oplus F2\left(SK2_{i-1}, L1_{i-1} \oplus SK3_{i-1} \oplus ROL(R0_{i-1})\right) \quad \text{Equation (2)}$$

i : round number $1 \le i \le 8$
$\oplus$: Bit-wise xor
ROL : 1-bit Rotate Left

F1,F2 are two different round functions, they have the same structure and employing the same s-boxes but with different diffusion matrices.
In accordance with equations (1,2), after one round, each 32-bit plaintext sub-block is dependent on two sub-blocks. And after one cycle, each 32-bit sub-block is dependent on the other three sub-blocks.

3.Key Expansion:

Key expansion algorithm (it is also called key scheduling) is the process of generation the set of all sub-keys required for the encryption/decryption processes .It must be impossible to recover the cipher key knowing one or more of the sub-keys.

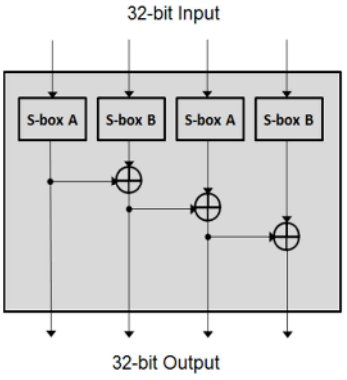

Figure (2) S-box-X

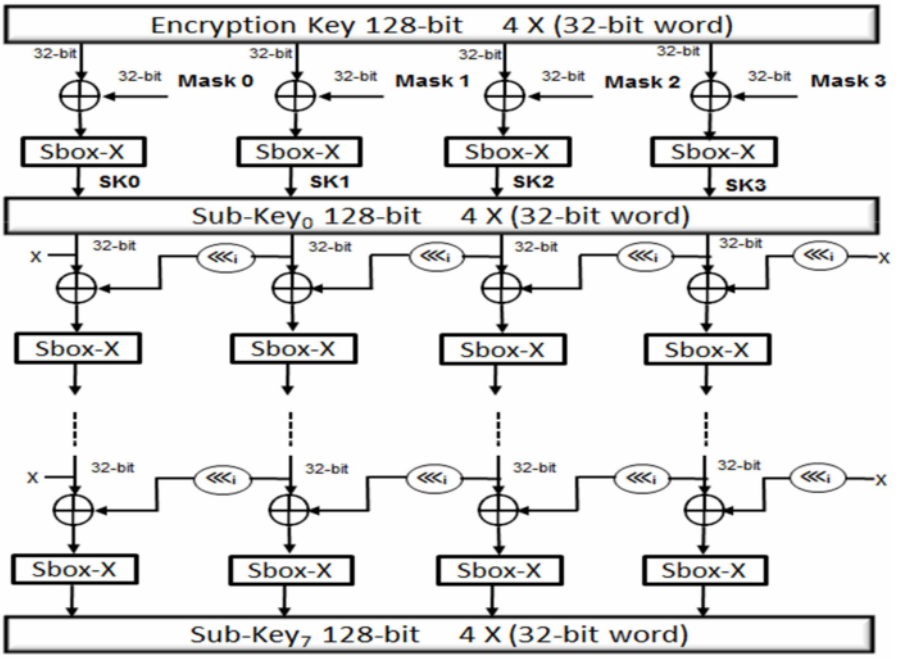

Figure (3) Key Expansion Algorithm

In HANK-1 key expansion algorithm, the cipher key is expanded into eight different sub-keys, each of length 128-bit (4x32-bit word).
The algorithm consists of eight rounds, initialization round and seven identical rounds. In the initialization round, the first sub-key is composed -as shown in equation(3)- by xoring a fixed random value MASK with the cipher key before the substitution box in order to deform the special pattern in the cipher key like all ones, all zeros and any other regular pattern. In the next seven rounds, each 32-bit sub-key word is composed according to equation (4) :

$$SKn_0 = \text{SBOX-}X\,[MASKn] \qquad \text{Equation (3)}$$

$$SKn_i = \text{SBOX-}X\,[(SK(n-1)_{i-1})\lll_i \oplus SKn_{i-1}] \qquad \text{Equation (4)}$$

n : sub-key word index ($0 \le n \le 3$) mod 4

i : round number $1 \le i \le 7$

$\lll_i$: circular shift left with i bits

The algorithm is easy to describe and analyze and it exhibits a high degree of non-linearity to prevent the full determination of round key differences from cipher key differences. There is no symmetry between the rounds (i.e the round transformation is not the same for all rounds due to the variable circular shift between rounds).

4.HANK-1 Round Functions (*F*)

The round function of any Feistel structure cipher algorithm is the core of security and the main source of confusion and diffusion in the algorithm [1,2]. The most important requirement for the round function is to be single value function (i.e for all the set of plaintext P and encryption key K, there is a function F^{-1} such that $F^{-1}(F(P,K),K) = P$).
The function also must be complex, highly non-linear, easy to analyze and implement.

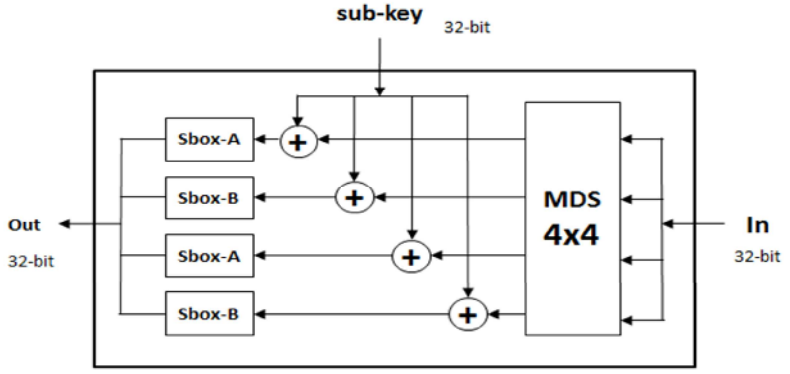

Figure (4) HANK-1 round function F1/F2

In HANK-1, there are two round functions (*F1,F2*), their general structures are identical but each one is parameterized by the XOR process with different portion of the round sub-key and with employing a different diffusion matrix.
The detailed structure of the round function is shown in figure (4), the diffusion part is represented in the 4x4 diffusion matrix which will be explained in details later. The confusion part is represented in the xoring process with the sub-key word SKn_i (key whitening) followed by substitution with four non-linear S-boxes.

5.Non-linear(S-box) Component:

The strength of various Feistel networks algorithms and specifically their resistance to differential and linear cryptanalysis is tied directly to their S-boxes [2, 4].
S-box (substitution box) is also called multi-output Boolean function; it is a mapping of m-bit inputs to n-bit outputs. It is generally the main nonlinear component in an algorithm and it gives the cipher algorithm its security. S-box is widely used in cryptographic algorithms design. Therefore, each cryptographic property of s-box is an important topic in the security evaluation.
The components of multi-output Boolean function interact and affect each other. Although each component function has certain cryptographic properties, the multi-output function constructed from them does not necessarily take on similar properties [4]. Hence, for multi- output function we must regard it as an integrated one.
There are several methods to construct or select S-box. The following three methods are very interesting [2]:

1. *Random and test selection*: the s-box is generated randomly with any software random number generator then the whole substitution table is tested for the desired properties. The size of the s-box must be large (8-bit input or more) in order to be strong and secure.
2. *Mathematical construction*: the s-box is constructed according to mathematical Principles so that they have proven security against differential and linear cryptanalysis. Power functions generation is an example for this method.
3. *Custom construction*: in this approach the s-box is constructed in a custom manner to realize a certain property. An example for this method is Kam&Davida structure sbox.in this structure, every output bit is a function of all input bits.

HANK-1 round functions deploy two different substitution boxes (S-box A, S-box B) each of 8-bit input size. The first substitution box (S-box-A) has been constructed based on mathematical basis which is the power function. Power functions are functions based on a transformation ^{d} for different exponents d .The calculations is performed over the finite field GF (2^8) with irreducible polynomial equals:

$$F(x) = x^8 + x^7 + x^6 + x^5 + x^4 + x^2 + 1.$$

The exponents is calculated according to Dobbertin &Niho function [11] shown below which gives maximum non-linearity :

Exponent	***Conditions on k***	***Condition on n***
$\sum_{i\ 0}^{n/2} 2^{ik}$	gcd(k,n) = *1* and $0 < k < t$	n *0 [4]*

The dimension (n=8) and 8 mod 4 =0 then t=4 because n=2*t
We choose k=3 such that GCD (3,8)=1 then:

Exponent = $\sum_{i\ 0}^{n/2} 2^{ik}$ = $2^{(0*3)} + 2^{(1*3)} + 2^{(2*3)} + 2^{(3*3)} + 2^{(4*3)}$ = 4681

The second substitution box (S-box-B) has been constructed randomly. All the s-box lookup table elements have been generated with a software random generator with values between [0:255], then the whole S-box has been tested for the differential uniformity property until the desired value has been reached. Table (2) shows the values for the cryptographic properties of the s-boxes compared to the properties of AES s-box.

Property	**Walsh Maximum**	**Auto-correlation Maximum**	**Differential Uniformity**	**Algebraic degree**
S-Box A	**32**	**48**	**16**	**5**
S-Box B	**68**	**100**	**8**	**6**
AES S-Box	**32**	**32**	**4**	**7**

Table (1) cryptographic properties values for HANK-1 s-boxes

6.Linear (Diffusion) Component:

Claude Shannon mentioned that confusion and Diffusion are basic requirement for all modern cryptosystems [5]. diffusion means that, every plaintext bit should influence every ciphertext bit and every key bit should influence every ciphertext bit, this is could be realized by using linear transformations. One of the most interesting and popular kind of linear transformations is these transformations generated on the basis of maximal distance separable codes (MDS-codes).Such linear transformations guarantees large number of active S-boxes in the context of differential or linear cryptanalysis. The Branch Number (β) of a linear transformation (T) is a measure of its diffusion power. It is the minimum number of nonzero elements in the input and output when the input elements are not all zero.

There are two basic requirements to MDS matrices of modern ciphers:

1- The maximum distance property.
2- The effectiveness of implementation.

The first requirement: if every square sub-matrix is nonsingular (its determinant is not equal to 0), then it is a necessary and sufficient condition to ensure that the matrix is MDS.

The second requirement: an efficiently implemented MDS matrix is a matrix which performs less number of multiplications and the number of ones in each entry is minimal. This property could be achieved by using circulant matrix where every row is consisted of the same element, shifted over one position ($x_{i,j} = x_{0,j-i \bmod n}$).

The proposed diffuser in HANK-1 algorithm consists of two matrices for F1 and F2 each one is 4X4 MDS matrix. By a special software tool that has been designed for this purpose, Each matrix has been generated randomly and then tested for the MDS property .the branch number for each is optimal and equals five, which is considered high diffusion rate and guarantees a maximum number of active s-boxes .

$$\begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \qquad \begin{bmatrix} 4 & 1 & 3 & 4 \\ 4 & 4 & 1 & 3 \\ 3 & 4 & 4 & 1 \\ 1 & 3 & 4 & 4 \end{bmatrix}$$

Figure (5) MDS Matrices for round functions F1 and F2

As shown in figure (5),The two matrices are circulant and the Hamming weight of any element is ≤ 2 which decrease the number of multiplication operation and so guarantees fast processing time and efficient implementation on either software and hardware platforms.

7.Mode of Operation and the Padding Technique

In block cipher, in order to prevent the creation of a code book of the plaintexts and corresponding ciphertexts as a result of the encryption of identical plaintext blocks into identical ciphertext blocks under the same encryption key, another process is used to combine the plaintext with another value or some sort of feedback. This process is known as the mode of operation. The mode of operation must not negatively affect the security and the efficiency of the underlying cipher [2].

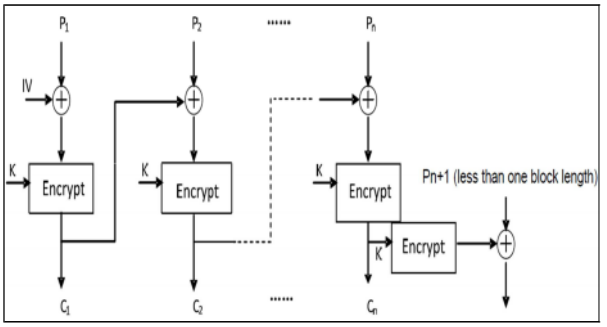

Figure (6) HANK-1 CBC-Encryption with padding

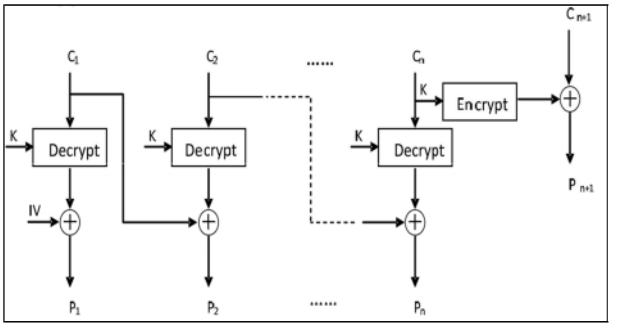

Figure (7) HANK-1 CBC-Decryption with padding

HANK-1 is operating in cipher block chaining (CBC) mode of operation. Each plaintext block is xored with the preceding 128- bits of ciphertext except the first plaintext block that is xored with non-secret nonzero random value called the initialization vector (IV).This means that each ciphertext block is dependent on all the preceding ciphertext block and consequently on all the preceding plaintext blocks. In some certain application like file or hard desk and digital voice frames encryption, the last incomplete plaintext block must be encrypted to the same length of ciphertext block. So a suitable padding mechanism should be used to handle this problem. The padding technique adopted in HANK-1 realize this concept by xoring the last n-bit incomplete plaintext block with n-bit of the encryption of the last complete ciphertext block [2]. Figures (6, 7) illustrate the HANK-1 encryption and decryption in CBC mode of operation with the employed padding mechanism.

8. Security Assessment:

In this section we present a set of tests performed to evaluate the randomness and the security strength of HANK-1 cipher algorithm.

Statistical Tests:

In symmetric cryptography, suitable metrics are needed to investigate the degree of randomness for a binary sequence. (NIST) Statistical Test suite and is an application used to evaluate the randomness of a binary sequence .It consists of set of tests based on hypothesis testing. A hypothesis test is a procedure for determining if an assertion about a characteristic of a population is reasonable. In this case, the test involves determining whether or not a specific sequence of zeroes and ones is random. This process is performed in the following steps

1. State the null hypothesis. “Assume that the binary sequence is random”.
2. Compute a sequence test statistic. “Testing is carried out at the bit level”.
3. Compute the P-value. “P-value ∈ [0, 1]”.
4. Compare the P-value to , where ∈ [0.001, 0.01].

Such that P-value is the probability that the chosen test statistic will assume values that are equal to or worse than the observed test statistic value, and represent the significance level [6].

Success is Declared whenever P-value ; otherwise, failure is declared.

A file of size 5 M byte in binary format was taken from the output of HANK-1 cipher algorithm working in CBC mode of operation with the input plaintext block equal zeros and non-zero random value for the initialization vector (IV).

Table (2) shows the tests results:

No.	Statistical Test	Average Calculated Test Statistic	Threshold Value	Overall Test Result
1	Frequency (Monobit)	0.82662	3.7469	Pass
2	Serial	3.051403	5.9371	Pass
3	Poker	131.0345	154.3002	Pass
4	Runs	1.725407	3.7469	Pass
5	Longest Runs of Ones	5.234451	12.5689	Pass
6	Binary Matrix Rank	1.906176	5.9371	Pass
7	Autocorrelation	0.864537	3.7469	Pass
8	Maurer's Universal	0.927862	3.7469	Pass
9	Block Frequency	15.77136	17.0794	Pass
10	Non-overlapping Template Matching	8.475958	15.4894	Pass
11	Overlapping Template Matching	7.324768	11.0442	Pass
12	Lempel-Ziv Compression	0.759953	3.7469	Pass
13	Approximate Entropy	8.096573	15.4894	Pass
14	Cumulative Sums (Cusum)	1.417298	3.7469	Pass
15	Random Excursions	5.389472	11.0442	Pass
16	Random Excursions Variant	1.254711	3.7469	Pass

Table (2) Statistical Tests results for HANK-1 cipher algorithm

Avalanche effect:

One of the most important metrics in cryptography for cipher algorithms specially block cipher is that a slight change (flipping single bit) in either the plaintext or the cipher key causes a significant change in the output ciphertext (typically half the ciphertext change) [1].

Table (3) shows the results taken from HANK-1when we change one bit of the plaintext for different bit positions:

Bit Position	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**12**	**14**	**15**
# Bits changed	54	63	66	53	63	54	54	59	57	61	55	66	74	80	68	71
Bit Position	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**
# Bits changed	66	69	63	66	68	62	67	63	66	54	72	63	67	62	57	72
Bit Position	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**	**43**	**44**	**45**	**46**	**47**
# Bits changed	**63**	**66**	**63**	**64**	**57**	69	62	69	68	63	63	60	69	69	65	65
Bit Position	**48**	**49**	**50**	**51**	**52**	**53**	**54**	**55**	**56**	**57**	**58**	**59**	**60**	**61**	**62**	**63**
# Bits changed	56	76	59	65	58	72	67	65	72	73	66	63	80	64	60	70

Table (3) cryptographic properties values for HANK-1 s-boxes

According to the values shown in table (4), the average calculated value for the number of changed bits equals to 64.The same test has been repeated with the cipher key and the average calculated value equals to 64 also. These results mean that HANK-1 exhibits a strong avalanche effect criterion.

Histogram of ASCII character set

Figures (8, 9) illustrate the ASCII character set and their frequency distribution for the text of the novel" The Merchant of Venice" before and after the encryption. It can be seen that unlike the plaintext, the ciphertext is uniformly distributed.

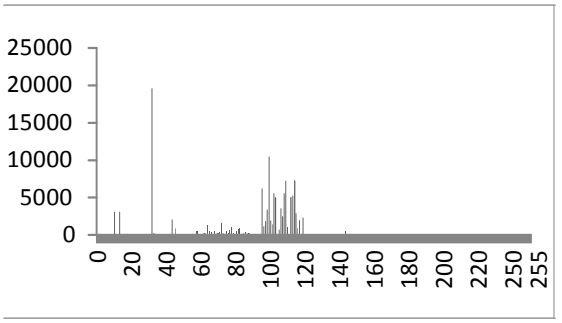

Figure (8) The plaintext Histogram

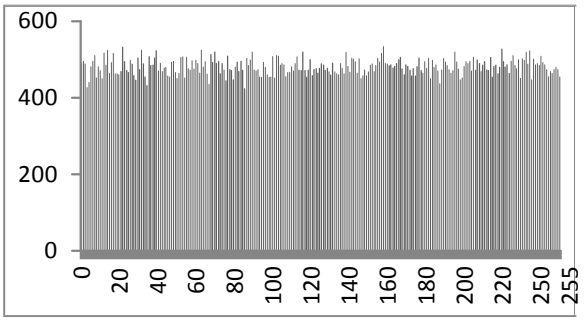

Figure (9) The ciphertext histogram

9. Software Implementation and Performance issue:

HANK-1 has been implemented in ANSI C language on Microblaze microprocessor. Microblaze is a 32-bit 'Harvard Architecture' RISC soft-core (synthesizable) processor that enables embedded developers to tune performance to match the requirements of target applications [8]. The basic architecture consists of 32 general-purpose registers, an Arithmetic Logic Unit (ALU), a shift unit, and two levels of interrupt and it executes most instructions in two clock cycles [7, 9]. We can configure this basic design with

more advanced features to allow us to balance the required performance of the target application against the logic area cost of the soft processor.

For the implementation of HANK-1, Microblaze has been configured with 62.5 MHz clock frequency, 64K bytes on-chip (local) RAM with no cache memory for either data or instructions. For the test purpose, a 32-bit soft-core timer/counter has been connected to Microblaze processor local bus (PLB) in order to count the CPU cycles. The Linker Script has been configured to place the instructions in the local RAM and place the data in the flash. The steps shown in Algorithm (1) illustrate the process of calculating the number of CPU cycles when executing HANK-1 in CBC mode of operation. HANK-1 encrypts one byte of input block in 5944 CPU cycles

1. Define the timer instance.
2. Initialize the timer (Timer ID).
3. Set the timer starting value (0x00000000).
4. Reset the timer.
5. Start the timer.

   /**************************/

   Run HANK-CBC

   /**************************/
6. Stop the timer.
7. Read the timer value (#CPU cycles).

Algorithm (1) HANK-1 pseudo-code

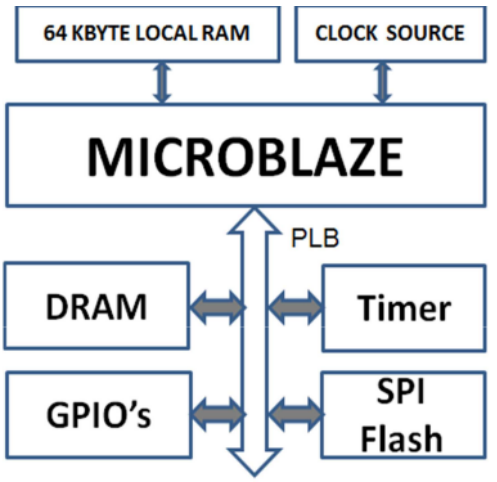

Figure (11) Microblaze Architecture

$$\text{Byte encryption time (seconds)} \quad \frac{\text{Number of CPU cycles}}{\text{Clock Frequency (HZ)}} \qquad \text{Equation (5)}$$

By substitution in equation(5) , HANK-1 throughput =84.1468 K bit/s .This high value of throughput makes it suitable to be implemented on various constrained devices (e.g. a smart card placed in a mobile handset for the purpose of voice encryption).

10. Conclusions and future work:

A new 128-bit Feistel structure block cipher algorithm with 8-rounds working in CBC mode of operation has been presented.

In this paper the high level of security of the proposed cipher algorithm have been shown through some tests like NIST suite statistical tests, avalanche effect and histogram of ASCII character set .the building blocks of the cipher algorithm have been built and chosen carefully to provide high resistance to known linear and differential

cryptanalytic attacks. It has been also shown that HANK-1 algorithm has a compact size and high throughput which makes it suitable for several confidentiality applications on limited resources devices.

Two interesting and important points are suggested for future work:

1-Finding a set of linear approximations to describe the transformations performed in HANK-1 algorithm in order to estimate its resistance to known plaintext attack.

2-Investagation of the high probability occurrence of certain plaintext differences and the corresponding differences into the output of the HANK-1 cipher which may be exploited to recover the secret key.

References:

[1] William Stalling, *Cryptography and Network Security*, 3, Ed. New Jersy, USA: Prentice Hall, 2003.

[2] Bruce Schneier, *Applied Cryptography*, 2nd ed. Toronto, Canda: WIELY, 1996

[3] Claude Carlet, *Boolean Methods and Models*, 1st ed. Paris, France: Cambridge University Press, 2006.

[4] Claude Carlet , *Boolean Functions for Cryptography and Error Correcting Codes* , University of Paris 8, France 2005.

[5] Shannon, Claude, *Communications theory of Secrecy Systems*, Bell systems Technical Journal.

[6] Juan Soto, James Nechvatal , Andrew Rukhin, "*A Statistical Test Suite for Random and Pseudorandom Numbers Generators*," May 2001.

[7] Michael Barr, *Programming Embedded Systems in C and C++*, 1st ed. O'Reilly & Associates, 1999.

[8] XILINX, EE677 VLSI *Architectures and Algorithms (Reconfigurable Computing) Performance profiling of Xilinx MicroBlaze.*

[9] XILINX, "*MicroBlazeReference Guide*," 2008.

[10] David J. Wheeler & Roger Needham, "*TEA, a Tiny Encryption Algorithm*", Cambrige University,England.

[11] Hans Dobbertin , *"Almost Perfect Nonlinear Power Functions on GF(2^n): The Niho Case"* German Information Security Agency, Bonn, Germany,1999.

[12] Wenling Wu and Lei Zhang, *"LBlock: A Lightweight Block Cipher",* Chinese Academy of Sciences, Beijing.