I. J. Computer Network and Information Security, 2016, 8, 11-19
Published Online August 2016 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2016.08.02

A Specialized Lightweight Metamorphic Function for KASUMI Metamorphic Cipher and Its FPGA Implementation

Rabie A. Mahmoud
Arab Academy for Science, Technology & Maritime Transport (AASTMT), Latakia Branch, Syria.
E-mail: rabiemah@yahoo.com

A. Baith Mohamed
Arab Academy of Science, Technology & Maritime Transport (AASTMT), Alexandria, Egypt.
E-mail: baithmm@hotmail.com

Abstract—To enhance the performance of the KASUMI Metamorphic Cipher, we apply a lightweight Metamorphic Structure. The proposed structure uses four lightweight bit-balanced operations in the function Meta-FO of the KASUMI Metamorphic Cipher. These operations are: XOR, INV, XNOR, and NOP for bitwise XOR, invert, XNOR, and no operation respectively building blocks of the Specialized Crypto Logic Unit (SCLU). In this work, we present a lightweight KASUMI Specialized-Metamorphic Cipher. In addition, we provide a Field Programmable Gate Array (FPGA) implementation of the proposed algorithm modification.

Index Terms—KASUMI, Metamorphic, Lightweight, Cipher, FPGA.

I. Introduction

The KASUMI Metamorphic Cipher [1] is a modified Feistel block cipher from KASUMI cipher [2], [3], [4], [5] which is a 64-bit block cipher using a 128-bit key with eight rounds and nonlinear S-boxes where KASUMI cipher forms the heart of the confidentiality and integrity algorithms of signalling and user data security within the Global Systems for Mobile Communications (GSM), General Packet Radio Service (GPRS), Enhanced Data Rates for GSM Evolution (EDGE), and the Third Generation Mobile System (3GPP) specifications for the Universal Mobile Telecommunications System (UMTS) networks. In this work, we present the KASUMI Specialized-Metamorphic Cipher to encrypt a 64-bit plaintexts using a 128-bit key. It is a lightweight metamorphic cipher that combines the specialized crypto logic unit with the function FO of KASUMI Cipher to encrypt 64-bit plaintext packets using 128-bit key. The Specialized Crypto Logic Unit (SCLU) is a special form of the Generalized Crypto Logic Unit (GCLU) [6] and Crypto Logic Unit (CLU) of the Stone Metamorphic Cipher [7], [8], [9] which are used in many famous ciphers to increase the cipher's entropy and improve its security such as the Metamorphic Twofish Cipher [10], the Metamorphic MARS Cipher [11], and the Metamorphic-Key-Hopping GOST Cipher [12]. This SCLU is built using four lightweight low-level bit-balanced operations: XORing a key bit with a plaintext bit (XOR), inverting a plaintext bit (INV), XNORing a key bit with a plaintext bit (XNOR), and producing a plaintext bit without any change (NOP). In the following few sections, we provide the structure of the KASUMI Specialized-Metamorphic Cipher by describing the SCLU and the enhanced function Specialized-Meta-FO. Subsequently, we discuss the results of the FPGA implementation of the KASUMI Specialized-Metamorphic Cipher including comparisons among modified KASUMI ciphers, a summary and our conclusions.

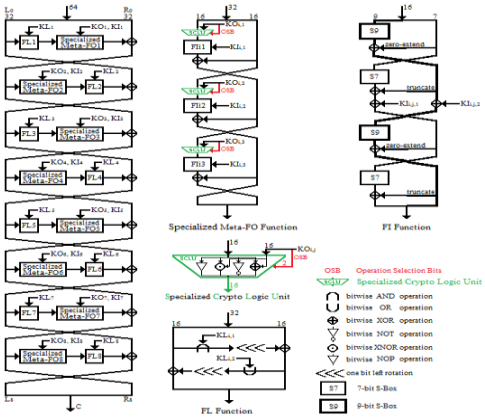

Fig.1. The structure of the KASUMI Specialized-Metamorphic Cipher

II. The KASUMI Specialized-Metamorphic Structure

The KASUMI Specialized-Metamorphic cipher is a lightweight form of KASUMI Metamorphic cipher, which is a Feistel Cipher with eight rounds encrypting 64-bit plaintext packets using 128-bit key, by replacing the crypto logic unit in the function Meta-FO with specialized

crypto logic unit converting the function Meta-FO into a Specialized-Meta-FO. Figure 1 shows the block diagram of the proposed KASUMI Specialized-Metamorphic Cipher.

A. The Specialized Crypto Logic Unit (SCLU)

The Specialized Crypto Logic Unit (SCLU) is a round key-dependent special function modified from key-driven Stone Metamorphic cipher and Generalized Crypto Logic Unit (GCLU) by selecting the four most lightweight operations from the GCLU operations. The four lightweight low-level bit-balanced operations are:

- (XOR) by XORing a key bit with a plaintext bit,
- (INV) by inverting a plaintext bit,
- (XNOR) by XNORing a key bit with a plaintext bit,
- (NOP) by producing the plaintext without any change.

Figure 2 shows the specialized crypto logic unit SCLU and Table 1 demonstrates each one of SCLU operations.

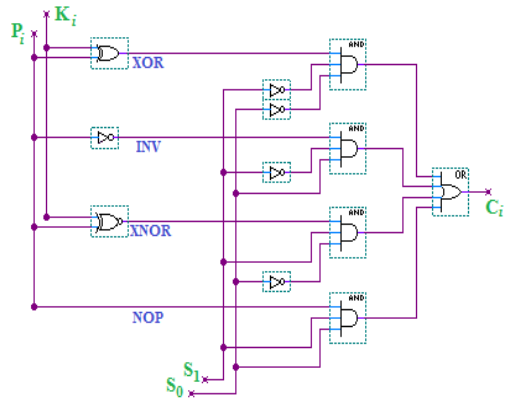

Fig.2. The specialized Crypto Logic Unit (SCLC)

Table 1. SCLU Operations

Mnemonic	*Operation*	*Select Operation code*
XOR	$C_i = K_i \oplus P_i$	"00"
INV	$C_i = \neg (P_i)$	"01"
XNOR	$C_i = K_i \odot P_i$	"10"
NOP	$C_i = P_i$	"11"

This SCLU is used as the encryptor and the decryptor where by changing the output cipher bit to become an input plain text bit, the new output will be the same as the old plain text bit. Appendix A shows the truth table of SCLU. Likewise, the operation selection bits (S1 S0) can be chosen from any two sub-key bits where the operation selection bits in the KASUMI Specialized-Metamorphic cipher are chosen from the $KO_{i,j}$ round keys. Figure 3 shows the location of operation selection bits.

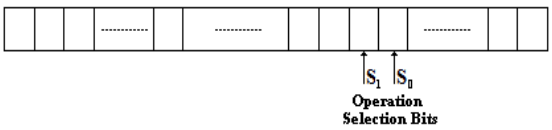

Fig.3. The Proposed Key Format Where The Location Of The Operation Selection Bits Is Shown

B. The Function Specialized-Meta-FO

The input to the function Specialized-Meta-FO comprises

- 32-bit data input $\boldsymbol{I}$,
- 48-bit subkey $\boldsymbol{KO_i}$,
- 48-bit subkey $\boldsymbol{KI_i}$.

The 32-bit data input is split into two halves, $\boldsymbol{L_0}$ and $\boldsymbol{R_0}$ where

$$\boldsymbol{I = L_0 \,||\, R_0}.$$

The 48-bit subkeys are subdivided into three 16-bit subkeys where

$$\boldsymbol{KO_i = KO_{i,1} \,||\, KO_{i,2} \,||\, KO_{i,3}} \text{ and } \boldsymbol{KI_i = KI_{i,1} \,||\, KI_{i,2} \,||\, KI_{i,3}},$$

and so for each integer $\boldsymbol{j}$ with $1 \le \boldsymbol{j} \le 3$ chose from $\boldsymbol{KO_{i,j}}$ 2-bit operation_selection_bits (OSB).

We define

If *operation_selection_bits = "00"* then

$$\boldsymbol{R_j = FI(L_{j-1} \oplus KO_{i,j}, KI_{i,j}) \oplus R_{j-1}}$$
$$\boldsymbol{L_j = R_{j-1}}$$

If *operation_selection_bits = "01"* then

$$\boldsymbol{R_j = FI(\neg L_{j-1}, KI_{i,j}) \oplus R_{j-1}}$$
$$\boldsymbol{L_j = R_{j-1}}$$

If *operation_selection_bits = "10"* then

$$\boldsymbol{R_j = FI(L_{j-1} \odot KO_{i,j}, KI_{i,j}) \oplus R_{j-1}}$$
$$\boldsymbol{L_j = R_{j-1}}$$

If *operation_selection_bits = "11"* then

$$\boldsymbol{R_j = FI(L_{j-1}, KI_{i,j}) \oplus R_{j-1}}$$
$$\boldsymbol{L_j = R_{j-1}}$$

Finally, we return the 32-bit value $(\boldsymbol{L_3 \,||\, R_3})$.

III. The FPGA Implementation

The KASUMI Specialized-Metamorphic Cipher FPGA-based implementation is applied to encrypt 64-bit plaintext packet using 128-bit user key producing 64-bit ciphertext packet at each cycle. We have implemented the cipher applying the VHDL hardware description language 2008 version [13], [14], [15] and utilizing Altera design environment Quartus II 15.0 (64-bit) Web Edition [16] with ModelSim Altera Starter Edition 10.3d [17]. The FPGA design was implemented using EP4CGX50DF27C6, Cyclone IV GX family device. The schematic diagram of proposed cipher with the implementation results is shown in Figure 4. RTL screen of the FPGA implementation is shown in Figure 5. Figure

6 shows the Technology Map Viewer for part of the hardware implementation of the cipher. Figure 7 demonstrates the floor plan for proposed modified cipher and Figure 8 displays the simulation showing the input, key, and the output cipher text bits. Appendix B displays the analysis and synthesis summary and timing analyzer of KASUMI Specialized-Metamorphic cipher. Major synthesis and timing differences among KASUMI, the KASUMI Metamorphic, the KASUMI Generalized-Metamorphic, and the KASUMI Specialized-Metamorphic ciphers in the balanced optimization technique are shown in Appendix C. Appendix D demonstrates the design and sample VHDL code of the function Specialized-Meta-FO.

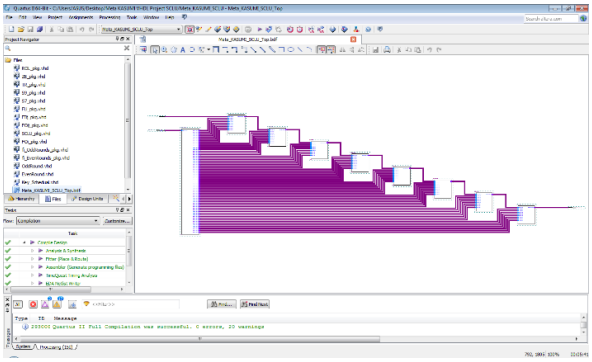

Fig.4. Compiler Tool Screen Showing Correct Implementation and Schematic Diagram of KASUMI Specialized-Metamorphic Cipher

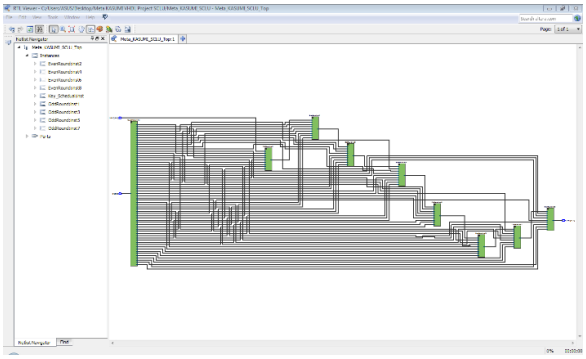

Fig.5. RTL Screen of KASUMI Specialized-Metamorphic Cipher

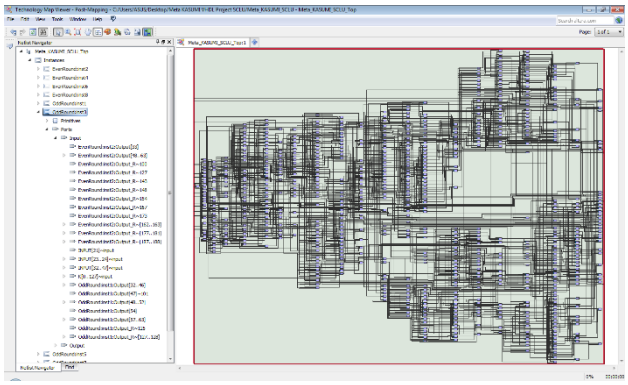

Fig.6. Technology Map Viewer for Part of one Round of KASUMI Specialized-Metamorphic cipher

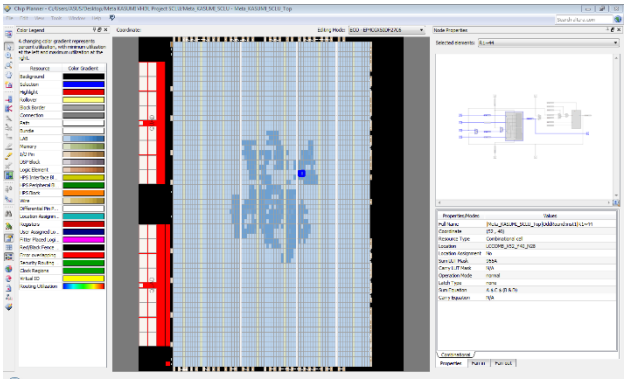

Fig.7. Floor-plan of Chip of KASUMI Specialized-Metamorphic Cipher

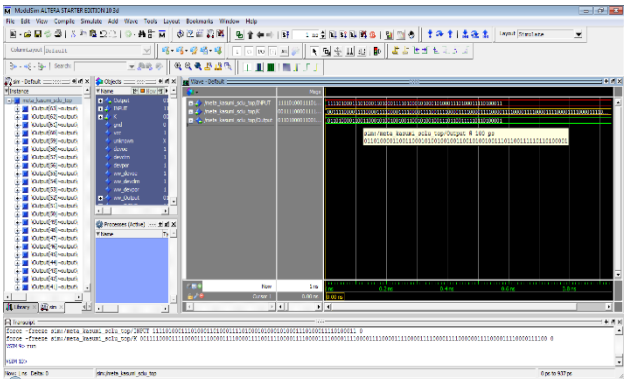

Fig. 8. ModelSim Simulator Screen Showing the Input, Key, and Output of KASUMI Specialized-Metamorphic Cipher

IV. Summary and Conclusions

We have proposed a lightweight modified cipher that is based on the KASUMI Metamorphic cipher. The modified cipher is called the KASUMI Specialized-Metamorphic cipher. A specialized crypto logic unit that utilizing the bit-balanced operations XOR, INV, XNOR, and NOP is merged in each round of the function Meta-FO of KASUMI Metamorphic cipher converting it into lightweight function Specialized-Meta-FO. In addition, we have presented a proof-of-concept FPGA hardware implementation of the proposed cipher. Various FPGA optimization techniques namely Balanced, High Performance Effort, Aggressive Performance, High Power Effort, Aggressive Power, and Aggressive Area optimization techniques were compared for proposed ciphers. Moreover, resources and timing delays comparisons between the KASUMI, the KASUMI Metamorphic, the KASUMI Generalized-Metamorphic, and the KASUMI Specialized-Metamorphic were shown. Certainly, KASUMI Specialized-Metamorphic cipher increased security with less consuming resources than KASUMI cipher itself.

APPENDIX A THE TRUTH TABLE OF THE SCLU

P_i	K_i	S_1	S_0	Operation	C_i
0	0	0	0	XOR	0
0	0	0	1	INV	1
0	0	1	0	XNOR	1
0	0	1	1	NOP	0
0	1	0	0	XOR	1
0	1	0	1	INV	1
0	1	1	0	XNOR	0
0	1	1	1	NOP	0
1	0	0	0	XOR	1
1	0	0	1	INV	0
1	0	1	0	XNOR	0
1	0	1	1	NOP	1
1	1	0	0	XOR	0
1	1	0	1	INV	0
1	1	1	0	XNOR	1
1	1	1	1	NOP	1

V. APPENDIX B THE FITTER AND TIMING REPORT DETAILS OF IMPLEMENTING KASUMI SPECIALIZED-METAMORPHIC CIPHER

FPGA synthesis of KASUMI Specialized-Metamorphic cipher is implemented with no time restrictions in Balanced, High Performance Effort, Aggressive Performance, High Power Effort, Aggressive Power, and Aggressive Area optimization techniques determining the usage number of logic elements, connections, and time delays. Table B1 shows the number of usage logic elements and their interconnections among optimization techniques of implementing KASUMI Specialized-Metamorphic cipher and Table B2 shows the timing delays among optimization techniques related with Slow 1200mV 85℃ Timing Model, Slow 1200mV 0℃ Timing Model, and Fast 1200mV 0℃ Timing Model. Figures B.1 and B.2 show a comparison chart of timing delays for KASUMI Specialized-Metamorphic cipher in Balanced optimization technique implementation.

Analysis & Synthesis and Fitter Summary

• Family: Cyclone IV GX
• Device: EP4CGX50DF27C6
• Nominal Core Voltage: 1.20 V
• Minimum Core Junction Temperature: 0 ℃
• Maximum Core Junction Temperature: 85 ℃.
• Optimization Technique: Balanced
• Total logic elements: 5,521 out of 49,888 (11%)
  -- Combinational with no register: 5,521
  -- Register only: 0
  -- Combinational with a register: 0

  Logic element usage by number of LUT inputs
  -- 4 input functions: 4,230
  -- 3 input functions: 690
  -- <=2 input functions: 601
  -- Register only: 0

  Logic elements by mode
  -- Normal mode: 5,521
  -- Arithmetic mode: 0

• Total LABs: 400 out of 3,118 (13 %)
• I/O pins: 256 out of 343 (75 %)
  -- Clock pins: 2 out of 10 (20 %)
  -- Dedicated input: 0 out of 25 (0 %)

• Total block memory bits: 0 out of 2,562,048 (0 %)
• Embedded Multiplier 9-bit elements: 0 out of 280 (0 %)
• Maximum fan-out: 34
• Highest non-global fan-out: 34
• Total fan-out: 20,517
• Average fan-out: 3.40

• Average interconnect usage (total/H/V): 4.0% / 3.6% / 4.5%
• Peak interconnect usage (total/H/V): 27.7% / 24.7% / 32.0%
• Block interconnects: 7,387 out of 232,464 (3 %)
• C16 interconnects: 670 out of 6,642 (10 %)
• C4 interconnects: 4,410 out of 136,080 (3 %)
• Direct links: 906 out of 232,464 (< 1 %)
• GXB block output buffers: 0 out of 2,640 (0 %)
• Global clocks: 0 out of 30 (0 %)
• Interquad Reference Clock Outputs: 0 out of 2 (0 %)
• Interquad TXRX Clocks: 0 out of 16 (0 %)
• Interquad TXRX PCSRX outputs: 0 out of 8 (0 %)
• Interquad TXRX PCSTX outputs: 0 out of 8 (0 %)
• Local interconnects: 3,254 out of 73,920 (4 %)
• R24 interconnects: 580 out of 6,930 (8 %)
• R4 interconnects: 4,515 out of 190,740 (2 %)

TimeQuest Timing Analyzer Summary

- Slow 1200mV 85℃ Model

  - Longest propagation delay RR which is measured from rising edge to rising edge was 143.172 ns from input port "K[114]" to output port "Output[47]". Also, longest delay RF which is measured from rising edge to falling edge was 143.277 ns, longest delay FR which is measured from falling edge to rising edge was 143.701 ns, and longest delay FF which is measured from falling edge to falling edge was 143.806 ns.

  - Longest minimum propagation delay was from input port "INPUT[13]" to output port "Output[27]" where RR was 25.782 ns, RF was 25.609 ns, FR was 26.290 ns, and FF was 26.117 ns.

- Slow 1200mV 0℃ Model

  - Longest propagation delay was from input port "K[114]" to output port "Output[47]" where RR was 129.061 ns, RF was 129.044 ns, FR was 129.381 ns, and FF was 129.364 ns.

  - Longest minimum propagation delay was from input port "INPUT[13]" to output port "Output[27]" where RR was 23.243 ns, RF was 23.219 ns, FR was 23.596 ns, and FF was 23.572 ns.

- Fast 1200mV 0℃ Model

- Longest propagation delay was from input port "K[114]" to output port "Output[47]" where RR was 83.980 ns, RF was 84.262 ns, FR was 84.751 ns, and FF was 85.033 ns.

- Longest minimum propagation delay was from input port "INPUT[13]" to output port "Output[27]" where RR was 14.732 ns, RF was 14.398 ns, FR was 15.395 ns, and FF was 15.061 ns.

Table B1. A resource and Routing Usage Comparison Among Optimization Technique Implementations of KASUMI Specialized-Metamorphic Cipher

		Balanced	High Performance Effort	Aggressive Performance	High Power Effort	Aggressive Power	Aggressive Area
Total LEs		5521	5521	5525	5521	5521	5521
Functions	4 input	4230	4230	4222	4230	4230	4230
	3 input	690	690	695	690	690	690
	<=2 input	601	601	608	601	601	601
Fan-Out	Total	20517	20517	20514	20517	50517	20517
	Max	34	34	34	34	34	34
	Average	3.40	3.40	3.39	3.40	3.40	3.40
Interconnects	Block	7387	7306	7239	7362	7209	7387
	C16	670	628	653	656	588	670
	C4	4410	3216	3210	3343	3356	4410
	Local	3254	3196	3201	3204	3242	3254
	R24	580	529	567	543	439	580
	R4	4515	3060	3080	3366	3517	4515
Direct Links		906	1488	1459	1406	1105	906

Table B2. A Timing Delays Comparison Among Optimization Technique Implementations of KASUMI Specialized-Metamorphic Cipher

			Balanced	High Performance Effort	Aggressive Performance	High Power Effort	Aggressive Power	Aggressive Area
Slow 1200mV 85℃ Model	Longest	RR	143.172	131.178	128.260	134.018	134.307	143.172
		RF	143.277	131.050	128.186	133.957	134.273	143.277
		FR	143.701	131.857	128.899	134.515	134.861	143.701
		FF	143.806	131.729	128.825	134.454	134.827	143.806
	Longest Min	RR	25.782	23.112	23.607	23.474	23.728	25.782
		RF	25.609	22.946	23.701	23.489	23.649	25.609
		FR	26.290	23.554	24.096	23.937	24.254	26.290
		FF	26.117	23.388	24.190	23.952	24.175	26.117
Slow 1200mV 0℃ Model	Longest	RR	129.061	117.923	115.660	120.528	120.957	129.061
		RF	129.044	117.936	115.615	120.573	120.946	129.044
		FR	129.381	118.477	116.098	120.813	121.379	129.381
		FF	129.364	118.490	116.053	120.858	121.368	129.364
	Longest Min	RR	23.243	20.696	21.263	21.180	21.348	23.243
		RF	23.219	20.646	21.219	21.127	21.390	23.219
		FR	23.596	21.091	21.626	21.545	21.758	23.596
		FF	23.572	21.041	21.582	21.492	21.800	23.572
Fast 1200mV 0℃ Model	Longest	RR	83.980	76.456	74.539	77.714	78.158	83.980
		RF	84.262	76.237	74.357	77.474	78.041	84.262
		FR	84.751	77.236	75.373	78.507	78.886	84.751
		FF	85.033	77.017	75.191	78.267	78.769	85.033
	Longest Min	RR	14.732	13.269	13.408	13.143	13.625	14.732
		RF	14.398	13.035	13.575	13.351	13.370	14.398
		FR	15.395	13.864	14.055	13.760	14.314	15.395
		FF	15.061	13.630	14.222	13.968	14.059	15.061

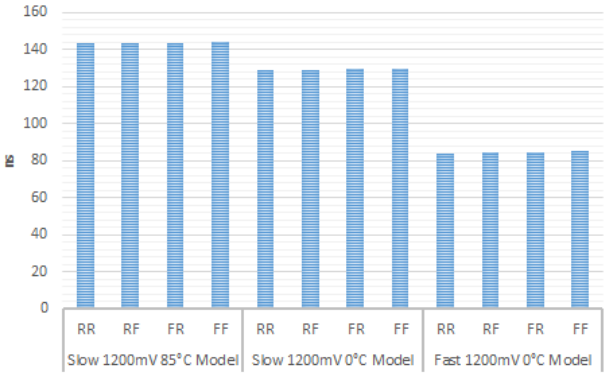

Fig.B.1. Longest Delays of Balanced Optimization Technique Implementation of KASUMI Specialized-Metamorphic Cipher

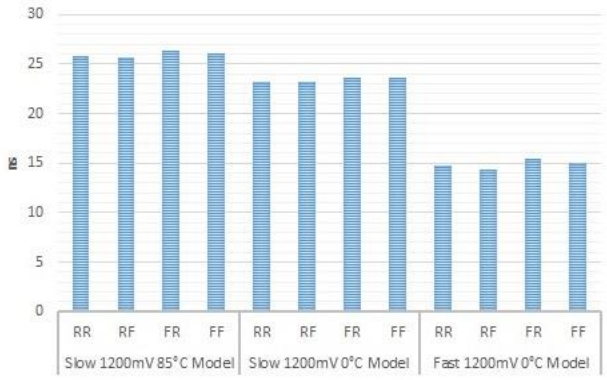

Fig.B.2. Longest Min Delays of Balanced Optimization Technique Implementation of KASUMI Specialized-Metamorphic Cipher

VI. Appendix C Major Implementing Differences Among KASUMI Specialized-Metamorphic Cipher And Modified KASUMI Ciphers In Balanced Optimization Technique

KASUMI Specialized-Metamorphic cipher consumes less resources and routing even than KASUMI cipher where the four operations XOR, INV, NOP, and XNOR of specialized crypto logic unit are low synthesized operations and the rising of maximum fan-out implements the KASUMI Specialized-Metamorphic cipher to consume less number of logic elements. Table C1 shows the number of usage logic elements and their interconnects and Table C2 shows the timing delays among KASUMI, the KASUMI Metamorphic, the KASUMI Generalized-Metamorphic, and the KASUMI Specialized-Metamorphic ciphers in Balanced optimization technique. Figures C.1 and C.2 show a comparison chart of those timing delays.

Table C1. A resource and Routing Usage Comparison Among KASUMI Specialized-Metamorphic and Modified KASUMI Ciphers

		KASUMI	Metamorphic-KASUMI	Generalized-Metamorphic-KASUMI	Specialized-Metamorphic-KASUMI
Total LEs		5582	7622	10559	5521
Functions	4 input	4365	5732	7759	4230
	3 input	665	1333	2026	690
	<=2 input	552	557	774	601
Fan-Out	Total	20884	28366	38987	20517
	Max	31	93	97	34
	Average	3.42	3.48	3.52	3.40
Interconnects	Block	7733	10033	13633	7387
	C16	702	783	894	670
	C4	4773	5906	8324	4410
	Local	3124	4412	6142	3254
	R24	599	703	846	580
	R4	5103	5818	7819	4515
Direct Links		939	1193	1564	906

Table C2. A timing Delays Comparison Among KASUMI Specialized-Metamorphic and Modified KASUMI Ciphers

			KASUMI	Metamorphic-KASUMI	Generalized-Metamorphic-KASUMI	Specialized-Metamorphic-KASUMI
Slow 1200mV 85℃ Model	Longest	RR	148.443	194.146	230.329	143.172
		RF	148.506	194.096	230.327	143.277
		FR	149.028	194.815	231.083	143.701
		FF	149.091	194.765	231.081	143.806
	Longest Min	RR	25.630	26.335	28.950	25.782
		RF	25.627	26.322	28.829	25.609
		FR	26.143	26.891	29.433	26.290
		FF	26.140	26.878	29.312	26.117
Slow 1200mV 0℃ Model	Longest	RR	134.001	174.609	207.348	129.061
		RF	134.175	174.587	207.314	129.044
		FR	134.426	175.064	207.863	129.381
		FF	134.600	175.042	207.829	129.364
	Longest Min	RR	23.116	23.697	26.042	23.243
		RF	23.051	23.723	26.035	23.219
		FR	23.483	24.152	26.410	23.596
		FF	23.418	24.178	26.403	23.572
Fast 1200mV 0℃ Model	Longest	RR	87.418	113.251	135.252	83.980
		RF	87.090	112.990	135.013	84.262
		FR	88.242	114.087	136.112	84.751
		FF	87.914	113.826	135.873	85.033
	Longest Min	RR	14.740	15.098	16.722	14.732
		RF	14.455	14.835	16.399	14.398
		FR	15.407	15.816	17.361	15.395
		FF	15.122	15.553	17.038	15.061

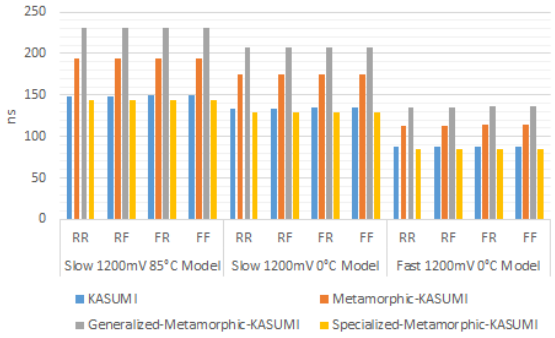

Fig.C.1. Longest Delays of Implementations of Modified KASUMI Ciphers

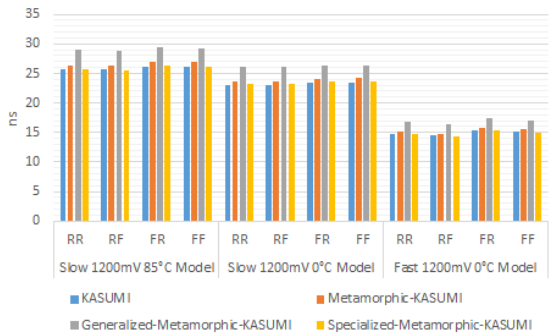

Fig.C.2. Longest Min Delays of Implementations of Modified KASUMI Ciphers

VII. Appendix D Sample VHDL Code of the Function Specialized-Meta-FO

Multi-nested VHDL functions in packages and component configuration of generated statements are used to program Key_Schedual, OddRound, and EvenRound components which are connected together in block diagram/schematic file to implement KASUMI Specialized-Metamorphic cipher. The function Specialized_Meta_FO of KASUMI Specialized-Metamorphic cipher defined through Specialized_Meta_FOi function in Specialized_Meta_FOi_pkg package using SCLU function in SCLU_pkg package and Specialized_Meta_FOij function in Specialized_Meta_FOij_pkg package:

- SCLU function represents the Specialized Crypto Logic Unit with two input parameters 16-bit input and 16-bit KOij. SCLU function returns a 16-bit output after applying the SCLU operations which are related to OSB.
- Specialized_Meta_FOij function represents one round of the three rounds of Specialized_Meta_FOi function where three input parameters 32-bit input, 16-bit KOij, and 16-bit KIij returns a 32-bit output after applying the related operations and calling the FIij and SCLU functions from FIij_pkg, and SCLU_pkg packages respectively.

- Specialized_Meta_FOi function calls Specialized_Meta_FOij function three times sequentially to implement three rounds and returning a 32-bit output by using seven input parameters 32-bit input, 16-bit KOi1, 16-bit KIi1, 16-bit KIi2, 16-bit KOi2, 16-bit KOi3, and 16-bit KIi3.

Sample VHDL codes are:

VHDL Code for SCLU_pkg

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;

PACKAGE SCLU_pkg IS
 FUNCTION SCLU(
                Input   :in std_logic_vector(15 downto 0);
                KOij    :in std_logic_vector(15 downto 0))
 RETURN  std_logic_vector;
END SCLU_pkg;

PACKAGE BODY SCLU_pkg IS
 FUNCTION SCLU(
                Input   :in std_logic_vector(15 downto 0);
                KOij    :in std_logic_vector(15 downto 0))
 RETURN std_logic_vector IS
   VARIABLE OSB   : std_logic_vector(1   downto 0);
   VARIABLE Output: std_logic_vector(15 downto 0);
 BEGIN
   -- Operation_selection_bits from KOij --
   OSB := KOij(7) & KOij(5);

   --SCLU operations
   If     OSB ="00" then Output := Input XOR  KOij;
   Elsif OSB ="01" then Output := NOT Input;
   Elsif OSB ="10" then Output := Input XNOR KOij;
   Elsif OSB ="11" then Output := Input;
   End If;
   RETURN Output;
 END FUNCTION SCLU;
END PACKAGE BODY SCLU_pkg;
```

VHDL Code for Specialized_Meta_FOij_pkg

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;
-- To call FIij Function --
USE WORK.FIij_pkg.ALL;
-- To call SCLU Function --
USE WORK.SCLU_pkg.ALL;

PACKAGE Specialized_Meta_FOij_pkg IS
  FUNCTION Specialized_Meta_FOij (
                Input   :in std_logic_vector(31 downto 0);
                KOij    :in std_logic_vector(15 downto 0);
                KIij    :in std_logic_vector(15 downto 0))
  RETURN std_logic_vector;
END Specialized_Meta_FOij_pkg;

PACKAGE BODY Specialized_Meta_FOij_pkg IS
 FUNCTION Specialized_Meta_FOij (
                Input   :in std_logic_vector(31 downto 0);
                KOij    :in std_logic_vector(15 downto 0);
                KIij    :in std_logic_vector(15 downto 0))
 RETURN  std_logic_vector IS
   -- The input halves Lj-1 and Rj-1 --
   VARIABLE Input_L : std_logic_vector(15 downto 0);
   VARIABLE Input_R : std_logic_vector(15 downto 0);
   -- The output halves Lj and Rj --
   VARIABLE Output_L:std_logic_vector(15 downto 0);
   VARIABLE Output_R:std_logic_vector(15 downto0);
   -- The output of SCLU Function --
   VARIABLE Output_SCLU :
                        std_logic_vector(15 downto 0);
   VARIABLE Output :  std_logic_vector(31 downto 0);
 BEGIN
   -- Splitting the Input
   Input_L := Input(31 downto 16);
   Input_R := Input(15 downto 0);
   -- Operation Series
   Output_L := Input_R;
   Output_SCLU := SCLU(Input_L , KOij);
   Output_R := FIij(Output_SCLU , KIij) XOR Input_R;
   -- Output of Function
   Output := Output_L & Output_R;
   RETURN Output;
 END FUNCTION Specialized_Meta_FOij;
END PACKAGE BODY Specialized_Meta_FOij_pkg;
```

VHDL Code for Specialized_Meta_FOi_pkg

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;
-- To call Specialized_Meta_FOij Function --
USE WORK. Specialized_Meta_FOij_pkg.ALL;

PACKAGE Specialized_Meta_FOi_pkg IS
 FUNCTION Specialized_Meta_FOi (
                Input   :in std_logic_vector(31 downto 0);
                KOi1    :in std_logic_vector(15 downto 0);
                KOi2    :in std_logic_vector(15 downto 0);
                KOi3    :in std_logic_vector(15 downto 0);
                KIi1    :in std_logic_vector(15 downto 0);
                KIi2    :in std_logic_vector(15 downto 0);
                KIi3    :in std_logic_vector(15 downto 0))
  RETURN std_logic_vector;
END Specialized_Meta_FOi_pkg;

PACKAGE BODY Specialized_Meta_FOi_pkg IS
 FUNCTION Specialized_Meta_FOi (
                Input   :in std_logic_vector(31 downto 0);
                KOi1    :in std_logic_vector(15 downto 0);
                KOi2    :in std_logic_vector(15 downto 0);
                KOi3    :in std_logic_vector(15 downto 0);
                KIi1    :in std_logic_vector(15 downto 0);
                KIi2    :in std_logic_vector(15 downto 0);
                KIi3    :in std_logic_vector(15 downto 0))
  RETURN std_logic_vector IS
```



```
    VARIABLE Round1 : std_logic_vector(31 downto 0);
    VARIABLE Round2 : std_logic_vector(31 downto 0);
    VARIABLE Round3 : std_logic_vector(31 downto 0);
    VARIABLE Output  : std_logic_vector(31 downto 0);

BEGIN
    -- Round 1 of Specialized_Meta_FOi
Round1 :=Specialized_Meta_FOij(Input, KOi1, KIi1);
    -- Round 2 of Specialized_Meta_FOi
Round2 :=Specialized_Meta_FOij(Round1, KOi2, KIi2);
    -- Round 3 of Specialized_Meta_FOi
Round3 :=Specialized_Meta_FOij(Round2, KOi3, KIi3);

    Output := Round3; -- The Output of Function
    RETURN Output;
  END FUNCTION Specialized_Meta_FOi;
END PACKAGE BODY Specialized_Meta_FOi_pkg;
```

REFERENCES

[1] Rabie A. Mahmoud, A. Baith Mohamed, Magdy Saeb, "Enhancing KASUMI Security by Affixing A Metamorphic Function and The Ensuing Hardware Implementation," International Journal of Computer Science and Communication Security (IJCSCS), Vol.6, No.1, Jan., 2016.

[2] 3GPP's site: http://www.3gpp.org

[3] 3GPP TS 35.201 Version12.0.0, "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Specification of the 3GPP Confidentiality and Integrity Algorithms; Document 1: f8 and f9 Specification," Sep., 2014.

[4] 3GPP TS 35.202 Version12.0.0, "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Specification of the 3GPP Confidentiality and Integrity Algorithms; Document 2: KASUMI Specification," Sep., 2014.

[5] Wikipedia, KASUMI's site: https://en.wikipedia.org/wiki/KASUMI

[6] Rabie A. Mahmoud, Magdy Saeb, "A Generalized Crypto Logic Unit (GCLU) with Software and Hardware Implementations," International Journal of Computer Science and Communication Security (IJCSCS), Vol.4, No.1, March, 2014.

[7] Magdy Saeb, "The Stone Cipher-192 (SC-192): A Metamorphic Cipher," International Journal of Computers and Network Security (IJCNS), Vol.1, No.2, pp.1-7, Nov., 2009.

[8] Rabie A. Mahmoud, Magdy Saeb, "Hardware Implementation of the Stone Metamorphic Cipher," International Journal of Computer Science and Network Security (IJCSNS), Vol.10, No.8, pp.54-60, 2010.

[9] Magdy Saeb, "Metamorphic Feistel Networks," International Journal of Computer Science and Communication Security (IJCSCS), Vol.5, No.3, July, 2015.

[10] Rabie A. Mahmoud, Magdy Saeb, "A Metamorphic-Enhanced Twofish Block Cipher And Associated FPGA Implementation," International Journal of Computer Science and Communication Security (IJCSCS), Vol.2, No.1, Jan., 2012.

[11] Ahmed Helmy, Magdy Saeb, A. Baith Mohamed, "A Metamorphic-Enhanced MARS Block Cipher," International Journal of Computer Science and Communication Security (IJCSCS), Vol.3, No.4, July, 2013.

[12] Rabie A. Mahmoud, Magdy Saeb, "A Metamorphic-Key-Hopping GOST Cipher and Its FPGA Implementation," International Journal of Computer Science and Communication Security (IJCSCS), Vol.3, No.7, Oct., 2013.

[13] Çetin Kaya Koç, "Cryptographic Engineering," Springer, 2009.

[14] Volnei A. Pedroni, "Circuit Design and Simulation with VHDL," 2nd Edition, MIT Press, 2010.

[15] Andrew Rushton, "VHDL for Logic Synthesis," 3rd Edition, John Wiley and Sons Ltd Publication, 2011.

[16] Altera's user-support site: https://www.altera.com/support/support-resources/design-examples/design-software/vhdl.html

[17] ModelSim-Altera's software-support site: https://www.altera.com/support/support-resources/design-software/modelsim.html

Authors' Profiles

A. Baith MOHAMED received the BSc. in Computer Science, Vienna University, MSc. and Ph.D. in Computer Science Vienna University in 1992. He is a Professor at the Arab Academy for Science and Technology and Maritime Transport (AASTMT), Computer Engineering Department. In addition, he holds the position of Vice Dean for Training and Community Services, College of Engineering and Technology (2010). He is also get the position of Director of Arab Academy for Science and Technology and Maritime Transport, Latakia, Syria branch (2013). Now he is a President Councilor at the AASTMT in Alexandria Egypt. His research interests include computer and Network Security, Bioinformatics, Steganography, cryptography, and Genetic Algorithms. He was also a member of an International project team in Europe, for design and implementation and maintenance of subsystems in the environment of peripheral processor controls as part of a larger Public Switched Systems (EWSD) in SIEMENS, AG. Austria. Also, he was a scientific researcher in the department of Information Engineering, Seibersdorf Research Institute (Atomic Energy Agency) in Austria, for the design and implementation of security software system in the domain of railway automation project (VAX/VMS, DEC systems). He was also a member of software testing for distribution points in an international project in AEG, Vienna, Austria. He is a senior member of IEEE Computer Society, USA since 2001. baithmm@hotmail.com

Rabie A. Mahmoud received the B.Sc. Degree, Faculty of Science, Tishreen University, Latakia-Syria, in 2001, the MS. and Ph.D. in Computational Science, Faculty of Science, Cairo University, Egypt, in 2007 and 2011 respectively. Currently, he is with General Organization of Remote Sensing (GORS), Syria and the Department of Computer Engineering, Arab Academy of Science, Technology & Maritime Transport, Latakia, Syria branch. His current interests include Cryptography, FPGA Implementations of Cryptography and Data Security Techniques. rabiemah@yahoo.com