ipPROCESS

ARCHITECTURE SPECIFICATION

Compact 128 bit blockcypher CLEFIA Implementation

Type-III Compact CLEFIA Implementation on FPGA

GNU LGPL Licence

1. Overview

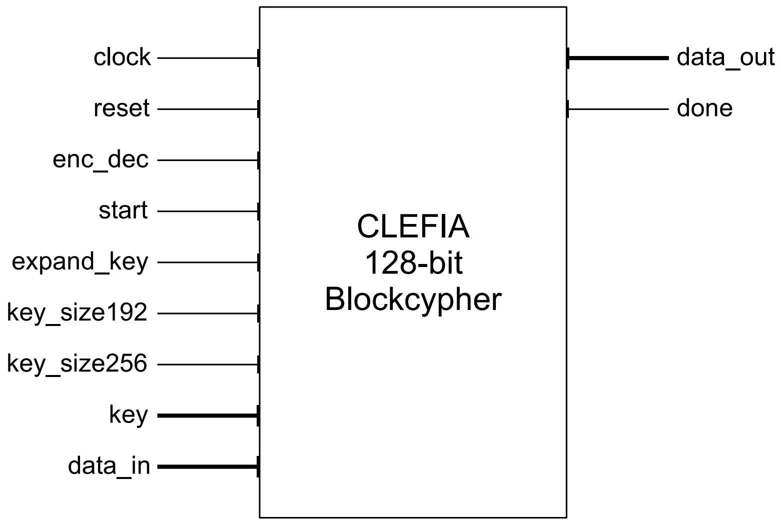
CLEFIA encryption algorithm is a novel symmetrical block ciphering algorithm proposed and developed by SONY Corporation focused on Digital Rights Management (DRM) purposes. This algorithm improves the security of encryption with the use of techniques such as Diffusion Switch Mechanisms, consisting of multiple diffusion matrices in a predetermined order, to ensure immunity against differential and linear attacks, and the use of Whitening Keys, combining data with portions of the key before the first round and after the last round.

The CLEFIA algorithm works with a key size of 128, 192, or 256 bits. It consists of a key scheduling phase and an input data block transformation phase computed over multiple rounds, employing a relatively homogeneous algorithm. The data path of CLEFIA Type-III is composed of a two 4-branch Feistel structure computed for several rounds, defined as $GFN_{8,n}$. However, this architecture places the two structures into the same design through pipeline and proper data scheduling.

The following architecture explores the resources provided by FPGA devices to provide a compact architecture capable of perform all CLEFIA modes of operation. This design it uses FPGA embbeded elements such as RAM memory and shift registers. The cypher block is designed in such way that it is capable of manage a 256 bit input block. Although CLEFIA cypher block is of 128 bit, this design is necessary in order to provide the data needed for key expansion and Round Key generation.

This documento describes the CLEFIA Blockcypher architecture. For complete Key Expansion design architecture descriptions please reffer to [reference] document.

2. Block Diagram



3. Signal Descriptions

Name	Length	Direction	Description
clock	1	Input	system main clock.
reset	1	Input	system main reset signal.
enc_dec	1	Input	enable encoding (1) and decoding (0) process.
start	1	Input	process start of operation signal.
expand_ key	1	Input	control system expansion key operation (1 = enable, 0 = diable).
key_ size192	1	Input	use a 192 bit key size.
key_ size256	1	Input	use a 256 bit key size.
key	256	Input	key used for expansion nad whitening.
data_ in	128	Input	input block to be cyphered.
data_ out	128	Output	cyphered block.
done	1	Output	end of operation signal.

4. Architecture Descriptions

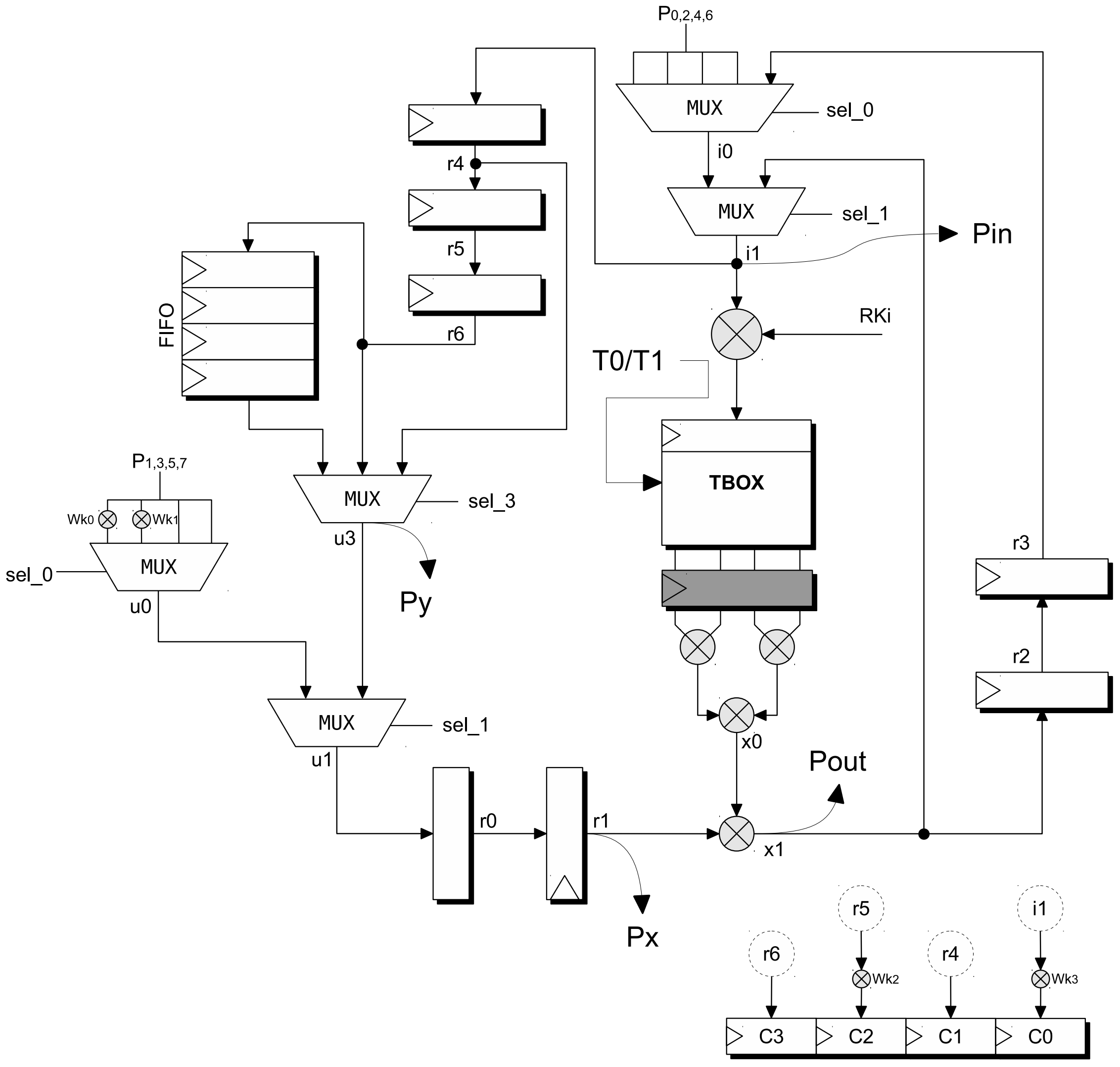
4.1. Basic Operation and Protocols

The CLEFIA blockcypher has two operation modes. The fist one is the cyphering operation which is placed into the three CLEFIA modes. The difference between them is the number of rounds. The second is the key expansion process, in witch a provided key is placed into the data input block in order to generate a L key. This key is futher used in Key Expansion, in order to generate the Round Keys.

In CLEFIA 128 bit key expansion the structure acts as a $GFN_{4,n}$ function due to the control machine logic. For 192 and 256 bit key expansion the system works as a full $GFN_{8,n}$ function capable of deal with 256 bit input block. Each F-Function has a 32 bit input/output data path. The different input key sizes that can be used in CLEFIA (128, 192, or 256 bits) directly influence the number of computed rounds, 18, 22, or 26, respectively. At each round it is used two previously genereated Round Keys.

The architecture is designed in a 2-deep pipeline where in first stage the F_0 function is calculates and in the second stage, the F_1 function is realized over the n rounds. The follwing table describe the scheduling of operations over the pipeline processe, and the number os clock cycles needed to keep the data ultil its usage.

4.2. Block Datapath



4.3. Data Path Scheduling

The notation used in the following table refered by $[X]_a^b$ implies that a is the 32-bit key index and b is the round being computed. The symbols T_0 and T_1 indicated the bit selection for T-Box 1 and 2. The columns C_x indicated the clock cycle being computed. It's content describe the number of clock cycles needed in order to save data from P_a^b in registers datapath.

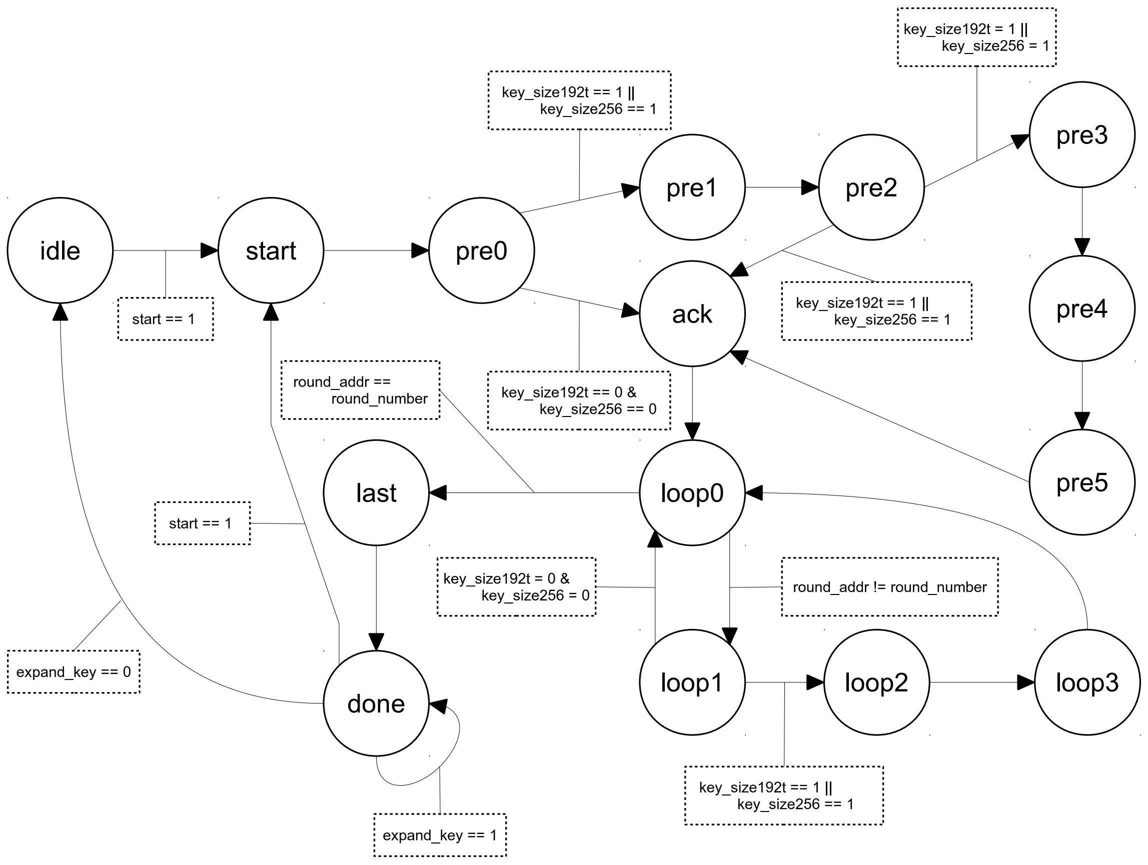
#	IN	$BRAM_{OUT}$	Px	Pout	Pin = Din(A)	Py	C_0	C_1
1	P_1; P_0	–	–	–	P_0^0	P_1^0	-	x
2	P_3; P_2	$T_0\ (P_0^0 + RK_0)$	–	–	P_2^0	P_3^0	-	#
3	P_5; P_4	$T_0\ (P_2^0 + RK_1)$	P_1^0	$\sum T_0 + P_1^0 = O_1^0$	P_4^0	P_5^0	o	#
4	P_7; P_6	$T_0\ (P_4^0 + RK_2)$	P_3^0	$\sum T_1 + P_3^0 = O_3^0$	P_6^0	P_7^0	o	#
5	–	$T_1\ (P_6^0 + RK_3)$	P_5^0	$\sum T_0 + P_5^0 = O_5^0$	$O_1^0 = P_0^1$	P_2^0	o	#
6	–	$T_1\ (P_0^1 + RK_4)$	P_7^0	$\sum T_1 + P_7^0 = O_7^0$	$O_3^0 = P_2^1$	P_4^0	o	#
7	–	$T_0\ (P_2^1 + RK_5)$	$P_1^1 = P_2^0$	$\sum T_0 + P_1^1 = O_1^1$	$O_5^0 = P_4^1$	P_6^0	5o	#
8	–	$T_1\ (P_4^1 + RK_6)$	$P_3^1 = P_4^0$	$\sum T_1 + P_3^1 = O_3^1$	$O_7^0 = P_6^1$	P_0^0		#
9	–	$T_0\ (P_6^1 + RK_7)$	$P_5^1 = P_6^0$	$\sum T_0 + P_5^1 = O_5^1$	$O_1^1 = P_0^2$	P_2^1		#
10	–	$T_1\ (P_0^2 + RK_8)$	$P_7^1 = P_0^0$	$\sum T_1 + P_7^1 = O_7^1$	$O_3^1 = P_2^2$	P_4^1		9#
11	–	$T_0\ (P_2^2 + RK_9)$	$P_1^2 = P_2^1$	$\sum T_0 + P_1^2 = O_1^2$	$O_5^1 = P_4^2$	P_6^1		
12	–	$T_1\ (P_4^2 + RK_{10})$	$P_3^2 = P_4^1$	$\sum T_1 + P_2^3 = O_2^3$	$O_7^1 = P_6^2$	P_0^1		
13	–	$T_0\ (P_6^2 + RK_{11})$	$P_5^2 = P_6^1$	$\sum T_0 + P_5^2 = O_5^2$	–	–		
14	–	–	$P_7^2 = P_0^1$	$\sum T_1 + P_7^2 = O_7^2$	–	–		

5. Cipher Key Summary

Key Size	RKeys	Rounds	Cycles
128-bits	18	36	38
192-bits	22	44	46
256-bits	26	52	54

The two extra clock cycles are due the input text word reading.

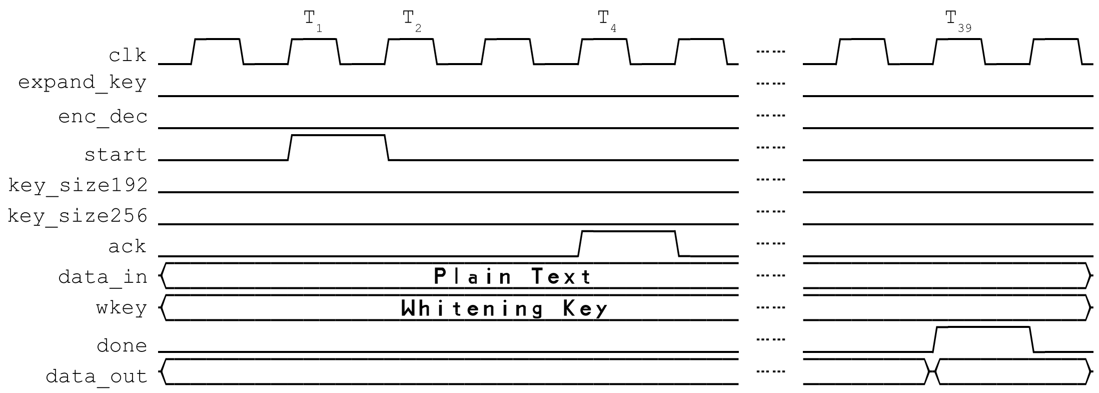
5.1. State Machine



idle
start
pre0
pre1
pre2
pre3
pre4
pre5
ack
last
loop0
loop1
loop2
loop3
done
start == 1
key_size192t == 1 || key_size256 == 1
key_size192t = 1 || key_size256 = 1
key_size192t == 0 & key_size256 == 0
round_addr == round_number
round_addr != round_number
start == 1
expand_key == 0
expand_key == 1

6. Timing

6.1. Blockcypher Processing with 128-bit Key



Time T_1

In this time the external control must provide a signal to start the processing. The system starts the pre-processing procedure, where all input data is going to be prossessed give descriptions above. As the processing example is a 128 bit block cyphering it must read each one of the 32 bit block parts before start the round loops.

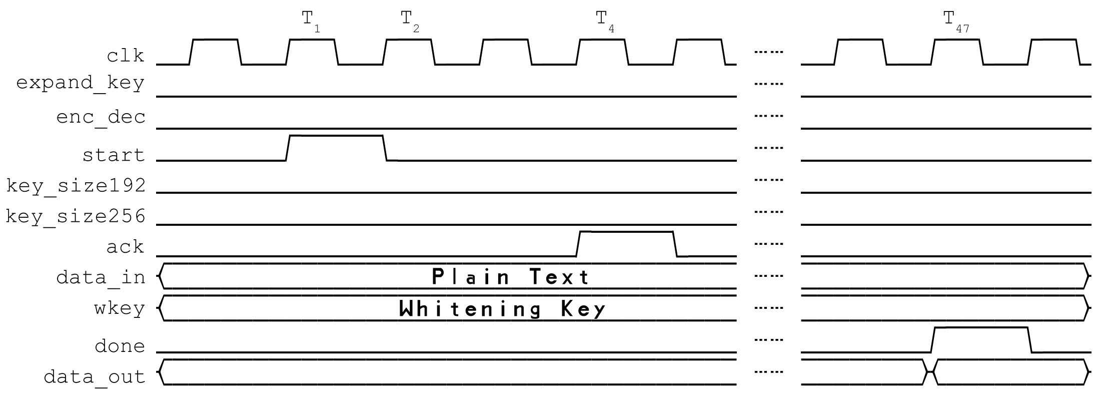
Time T_4

At this time, the system has already registeres all input words comming from the 128 bit block. For now, the ack signal says that the loop process has started and the cyphering process is running.

Time T_{39}

After the entire set of rounds finished, the process is completed and the done signal is set to high. This signal is also used to signal the start of the key expansion, when necessary.

6.2. Blockcypher Processing with 192-bit Key



Time T_1

In this time the external control must provide a signal to start the processing. The system starts the pre-processing procedure, where all input data is going to be prossessed give descriptions above. As the processing example is a 128 bit block cyphering it must read each one of the 32 bit block parts before start the round loops.

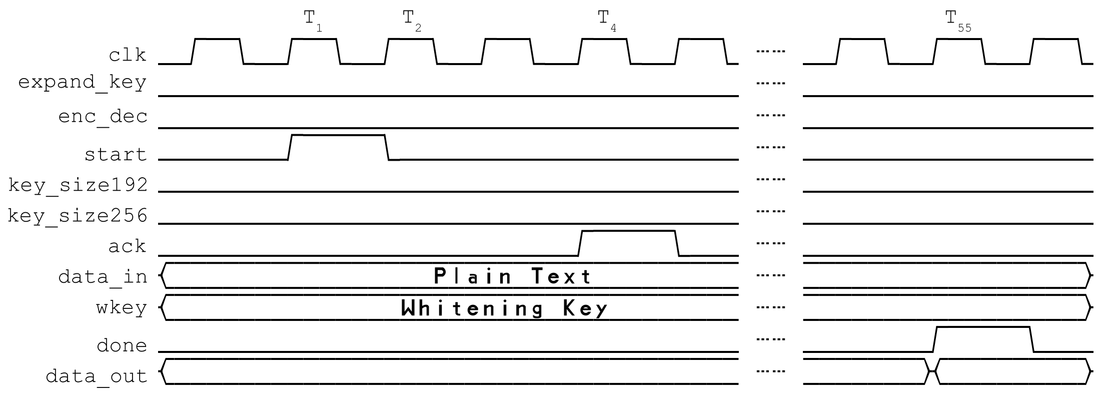
Time T_4

At this time, the system has already registeres all input words comming from the 128 bit block. For now, the ack signal says that the loop process has started and the cyphering process is running.

Time T_{47}

After the entire set of rounds finished, the process is completed and the done signal is set to high. This signal is also used to signal the start of the key expansion, when necessary.

6.3. Blockcypher Processing with 256-bit Key



Time T_1

In this time the external control must provide a signal to start the processing. The system starts the pre-processing procedure, where all input data is going to be prossessed give descriptions above. As the processing example is a 128 bit block cyphering it must read each one of the 32 bit block parts before start the round loops.

Time T_4

At this time, the system has already registeres all input words comming from the 128 bit block. For now, the ack signal says that the loop process has started and the cyphering process is running.

Time T_{55}

After the entire set of rounds finished, the process is completed and the done signal is set to high. This signal is also used to signal the start of the key expansion, when necessary.

6.4. LKey Generation

Revision History

Date	Description	Owner
06/11/2013	Document conception	João Carlos
07/11/2013	• Design component external interface; • Design basic datapaph; • Describe cyphering state machine; • Place and system design overview;	João Carlos
28/11/2013	• Timing for 128-bit key cyphering; • Timing for 192-bit key cyphering; • Timing for 256-bit key cyphering;	João Carlos