

Appendix 3 Weak Keys and Early Wake Up

As a result of the following study, the safe loading of secret keys, in both the MAC and Cipher mode has been enhanced to preclude a delayed "wake-up" of the three TMB Control Units. An all zero Control Key Word would deactivate the three programmable counters for the first sixteen clock cycles. Inactivated counters preclude generation of both the Left and the Right Slip signals to the Register Bank and any change from Permutation Configuration Y1 [Ref 9 Figs. 13, 10S1N & 10S3] for the first 15 and 9 Primary Clock cycles, respectively. Therefore, the loading strategy includes a 32 cycle scramble following key loading;

16 cycles to insure a complete "weak key wake-up" and the additional 16 cycles to insure complete diffusion of random feedback from the Register Bank back into the TMB Control Units. This has been effectivly reduced in the enhanced versions.

Note: This study was compliant with a previous version of the ZK-Crypt. The previous version included a historic multi gate NOR "zero detector", necessary to preclude the "Stuck on Zero" nLFSR syndrome in normal closed loop operation. In the ZK-Crypt, each nLFSR is fed bits from either the Lower Feedback or the Super Tier Feedback, on every Primary Clock cycle. To assure positive wake up in non-keyed and keyed operation, the Global (Pre)set Command, sets a '1' in the MS cell of each nLFSR. This is especially important in the Super Tier, whose Initial Condition is not altered by the direct Key Load (as the TMB tiers are Loaded directly). This means that if an adversary has Loaded a 128 bit IV or Secret Key, on the first Primary Clock pulse, the Super Tier will "rain down" six '1' signals to the Churn, which suffices to provide pseudo randomly load the Feedback Stores. The following 32 Scrambles suffice to activate the whole engine.

We define a Weak Key as an initial configuration of the ZK-Crypt engine that can lead to a situation such that an adversary could find a way wherein an original configuration difference can be used to predict the stream differences with some probability. In this first version, all permutations and parameters were recorded and/or computed manually, to be checked against the software demonstrator [zk-code], wherein we will also prove conclusively, after having proved massive diffusion, and best possible statistical outputs, that there are no differentials in any nodes in the Data Churn [biham]; after having proved that both the deterministic and random noise source configurations pass the most rigorous noise statistic testing [zk-ais31].

As we have shown extensive proofs of diffusion in the cryptanalysis of the ZK-Crypt [zk-secure]; e.g., a single change in the feedback source is immediately reflected in the equations of more than 140 variables in the Register Bank and the Data Churn, (typically on more than 160 binary values) on the first clock cycle. The next time around, the differences caused by one bit in the first clock cycle will change, pseudo-randomly, virtually all 320 binary variables in the Data Manipulator with a probability of 0.5. Over the next 2 to 15 cycles, we are typically assured that all obscure binary variables in the Random Controller will be pseudo-randomly affected. This study is more rigorously conclusive in the concatenated mode of operation wherein two or more identical ZK-Crypt engines operate in tandem and wherein The Lower Feedback 32 bit channel of each engine is exclusively relayed to its near neighbor.

We decided on a more rigorous definition of a weak key initial configuration to measure what we call the "most delayed wake up" of the Random Controller. We experimented with two approaches, and quickly realized that using an all zero secret key, (literally no key whatsoever), wherein the machine must "wake itself up" from a global initializing preset, where for the first the nine clock cycles the adversary would know with a high probability, most of the 70 variables in the Random Controller, and up to the sixteenth Primary Clock, would deterministically know over 20 variables in the Controller, thereby setting a fixed sequence in much of the engine's configuration. All this knowledge proves to



be of no avail, being lost in the 32 cycle "random scramble" following the unique method of loading the running key, wherein the engine is run in MAC mode with dual track orthogonal feedback streams.

Starting from the Global Reset "weakest key" initialization, the initialized Secret Key Bits in the Random Controller; Top Tier, Middle Tier, Bottom Tier are all zero. At the first clock, the NFIX gates (designed to prevent the "stuck on zero" syndrome in linear feedback shift registers) will cause an internal Feedback into all 8 of the Register Bank nLFSRs. Note that at reset, several control variable bits are set to "1"; see Figs. 10S3, 4DC1, and 4DC2, in [zk-ccc]. Note that internal signal labels are not global; in each case the reader is requested to examine the referenced relevant figures in the [zk-ccc].

It is assumed that the reader is familiar with the ZK-Crypt architecture, explained in [zk-undst], and can reference the concept and circuits in [zk-ccc].

Note that: Txxx=Topxxx; Mxxx=Middlexxx=Midxxx; Bxxx=Bottomxxx=Botxxx.

A) The (P)Random Primary Clock Enabler Fig. 4DC1&2. See sections E), B), C) and D)in this document.

A8 =1 only when first 3LS 9 Bit Counter = 001, doesn't happen in first eight clocks, see F).
 9Bit nLFSR (Fig. 4DC2) cannot receive a Slip until at least one of the 3 Control Unit counters "hits" 15 (1111₂). As the Control Unit Initial counter conditions are all "0"; there are also no Left or Right Slip bits activating the TMB Register Bank in the first 15 Primary Clock intervals, therefore no slips to the Register Bank, and no flipping of the Control Unit Config flip-flops for the first 15 clock cycles. After initialization the minimum TMB Control Unit count is 4, and the maximum is 12. There can be no changes of the general Configuration, prior to the (P)Random missing clock's stepping the configuration stepper in Fig. 10S1 [zk-ccc]. Y1=1 is the chosen Config; i.e., Y2 and Y3 are "0" by definition for the first six Primary Clock.

ND (synched to the Primary Clock first pulse is I.C "1" while N is also I.C. "1". $sfr1IN = \text{NOT}(sfr1OUT \wedge sfr2OUT) \vee A8$; $sfr2IN = \text{OLDSfr1IN}$.

At rising Primary Clocks-

	$sfr1IN = \text{NOT}(sfr1OUT \wedge sfr2OUT) \vee A8$;				$sfr2OUT = \text{OLDSfr1OUT}$		$sfr1OUT = \text{OLDSfr1IN}$	
1 _{Init}	0	1	1	0	1 _{init}	X	1	X
2	1	0	1	0	1	1	0	0
3	1	1	0	0	0	0	1	1
4	0	1	1	0	1	1	1	1
5	1	0	1	0	1	1	0	0
6	1	1	0	0	0	0	1	1
7	0	1	1	0	1	1	1	1

At rising Primary Clocks-

	sfr1	sfr2	vB	vM	= N;	ND=oldN	A8
1 _{Init}	1	1	1	0	1	1	0
2	0	1	0	1	1	1	0
3	1	0	1	0	1	1	0
4	1	1	0	1	1	1	0
5	0	1	0	1	1	1	0
6	1	0	0	0	0	1	0
7	1	1	0	0	1	0	0
	\$	F	F	\$	\$	\$	

The 3 celled Johnson Counter rotates at every missing (P)Random Clock; a pulse enabled by the inversed N signal. At each shift the Johnson Counter re-encodes the outputs of the TMB Control Config Flip Flops. A TMB configuration flip flop may only change its outputs when its Control Unit Counter "hits 15"; e.g., the Top Control Unit hits 15₁₀ = 1111₂- THIT15=Q_{TA}·Q_{TB}·Q_{TC}·Q_{TD} in Fig. 13.

Initial Values for Johnson FFs' outputs: QJC0 = 1; QJC1 = 0; QJC2 = 0.

As the content of the Johnson Counter is a revolving "1" then if QJC_x = 1 for Present x value then; QJC(X-1) mod 3 = 0 and QJC(X+1) mod 3 = 0; then the next value of the cells of the counter will be- NXTQJC(X+1 mod 3) = 1 and NXTQJC(X) mod 3 = 0 and NXTQJC(X-1) mod 3 = 0.



As shown in Fig. 10S1

$Y1 = QJC0 \cdot [\text{NOT}(TCONF FF \vee BCONF FF)] \vee QJC2 \cdot [(MCONF FF \vee BCONF FF)]$; only relevant if the "1" in the Johnson

Counter is either in the JC0 or JC2 flip flop.

$Y2 = QJC1 \cdot [\text{NOT}(TCONF FF \vee MCONF FF)] \vee QJC0 \cdot [(TCONF FF \vee BCONF FF)]$; only relevant if the "1" in the Johnson

Counter is either in the JC0 or JC1 flip flop.

$Y3 = QJC2 \cdot [\text{NOT}(MCONF FF \vee BCONF FF)] \vee QJC1 \cdot [(TCONF FF \vee MCONF FF)]$; only relevant if the "1" in the Johnson

Counter is either in the JC1 or JC2 flip flop.

Y1, Y2 & Y3 may receive new values when at least one the TMB (Top, Mid & Bot) counters "Hit15" or with slightly less probability, when the Johnson Counter is stepped by the Missing Primary clock.

	BCONF FF	MCONF FF	TCONF FF	Y1	Y2	Y3	QJC0	QJC1	QJC2	ND
1	0	0	0	1	0	0	1	0	0	1
2	0	0	0	1	0	0	1	0	0	1 TMB _{CONF FF} FFs may change polarity if counter hits
15.										
3	0	0	0	1	0	0	1	0	0	1
4	0	0	0	1	0	0	1	0	0	1
5	0	0	0	1	0	0	1	0	0	1
6	0	0	0	1	0	0	1	0	0	1
7	0	0	0	0	1	0	0	1	0	0 Change triggered by the missing (P)Random Clock
8	0	0	0	0	1	0	0	1	0	1
9	0	0	0	0	1	0	0	1	0	1
10	0	0	0	0	1	0	0	1	0	X
11	0	0	0	0	1	0	0	1	0	0; Assuming no missing (P)Random Clock to activate the Johnson Counter.
12	0	0	0	0	1	0	0	1	0	0; Assuming no missing (P)Random Clock
13	0	0	0	0	1	0	0	1	0	0; Assuming no missing (P)Random Clock
14	0	0	0	0	1	0	0	1	0	0; Assuming no missing (P)Random Clock
15	0	0	0	0	1	0	0	1	0	0; Assuming no missing (P)Random Clock
16	0	0	0	0	1	0	0	1	0	0; Assuming no missing (P)Random Clock
XX	X	X	X	X	X	X	0	1	0	0; Assuming no missing (P)Random Clock
	D	C	B	\$	\$	\$	\$	\$	\$	A

For the 1st to the 6th Rising Primary Clocks the 3 Control Units are activated, as ND=1 → The (P)Random Clock is active



B) The Top Control Fig. 13. Top BRN=QT0 ⊕ QT1 ⊕ QT2; $nextTCONF FF = ([NOT(TCONF FF)] ⊕ 4TH TOGGLE) \cdot THIT15$ else (TCONF FF) = old(TCONF FF) ;
 $THIT15 = QTA \cdot QTB \cdot QTC \cdot QTD$;

Top Control Unit nLFSR At rising (P)Random Clocks- synched to PC until rising clk 7 (ND is N delayed one clk interval).

@RiseCk	T0	T1	T2	FB (T0⊕T2⊕SLIP⊕NFIX)	ND	Q17	QTA = CNTRL	SLIP	TopBRN
1st	0	0	0	1 (NFIX)	1	0	0	0	0
2	1	0	0	1	1	0	1	0	1
3	1	1	0	1	1	?	0	0	0
4	1	1	1	1⊕ Control Slip	1	x	1	X	1
5	X	1	1	X	1	x	0	0	X
6	X	X	1	X	1	x	1	X	X
7	Repeats 6 ND Clk=0				0	x	0	0	X
	\$	\$	\$		A	R	\$		\$

Top Control Unit Counter, Config FF and Slip Generation. After initialization all Up-count starts with QTC=0

@RiseCk	TA	TB	TC	TD	→ THIT15=0	→ NXT CNT=CNT+1	TCONF FF	JUG	QTO	QT1	QT2	Q13	RQ15	LQ15	das	If THIT15=1; THEN -	
1st	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	the nextQTA, QTB, QTC, QTD, TCONF FF are:
2	1	0	0	0	0	2	0	0	1	0	0	1	1	1	1		
3	0	1	0	0	0	3	0	1	1	1	0	1	φ	φ	1		
4	1	1	0	0	0	4	0	1	1	1	1	0	φ	φ	0	QTA=QT0⊕Q13 (Mid RnLFSR)	
5	0	0	1	0	0	5	0	1	X	1	1	0	φ	φ	0	QTB=QT0⊕QT1⊕LQ15 (Supr LnLFSR)	
6	1	0	1	0	0	6	0	1	X	X	1	φ	φ	φ	1	QTC= 0 (Count=4 to 15→12 max clocks)	
7	0	1	1	0	0	7	0	φ	φ	φ	φ	φ	φ	φ	φ	QTD=JUG⊕QT1⊕ TCONF FF⊕RQ15 (Supr RnLFSR)	
8	1	1	1	0	0	8	0	φ	φ	φ	φ	φ	φ	φ	φ	TLeftSlip=das⊕QT2	
9	0	0	0	1	0	9	0	φ	φ	φ	φ	φ	φ	φ	φ	TRightSlip=notdas⊕QT2	
10	1	0	0	1	0	10	0	φ	φ	φ	φ	φ	φ	φ	φ	φ = Often don't care	
11	0	1	0	1	0	11	0	φ	φ	φ	φ	φ	φ	φ	φ	X = Unknown - Data Dependent	
12	1	1	0	1	0	12	0	φ	φ	φ	φ	φ	φ	φ	φ		
13	0	0	1	1	0	13	0	φ	φ	φ	φ	φ	φ	φ	φ		
14	1	0	1	1	0	14	0	φ	φ	φ	φ	φ	φ	φ	φ		
15	0	1	1	1	0	15	0	φ	φ	φ	φ	φ	φ	φ	φ		
16	1	1	1	1	1	16	0	X	X	X	X	X	X	X	X	TCONF FF= (Old NotTCONF FF)⊕ 4thToggle	
XX	X	X	X	X	X	XX	X	φ	φ	φ	φ	φ	φ	φ	φ		
16	\$	\$	\$	\$	\$		\$	E				R	R	R	E		



C) The Middle Control Fig. 14. Mid BRN=QM0 ⊕ QM1 ⊕ QM2; $nextMCONF FF = \{ [NOT(MCONF FF)] \oplus 4TH TOGGLE\} \cdot M_{HIT15}$ else $(MCONF FF) = old(MCONF FF)$;
 $M_{HIT15} = Q_{MA} \cdot Q_{MB} \cdot Q_{MC} \cdot Q_{MD}$

Mid Control Unit nLFSR At rising (P)Random Clocks- synched to PC until rising clk 7 (ND is N delayed one clk interval).

@RiseCk	M0	M1	M2	M3	M4	FB=M ₄ ⊕M ₁ ⊕N _{FIX} ⊕SLIP	ND	Q12^AMA=CNTRL	SLIP	MidBRN
1 _{Init}	0	0	0	0	0	1	1	0	0	0
2	1	0	0	0	0	0	1	0	1	0
3	0	1	0	0	0	1	1	0	0	0
4	1	0	1	0	0	0⊕Cntrl Slip	1	X	1	X
5	X	1	0	1	0	X	1	X	0	0
6	X	X	1	0	1	X	1	X	1	X
7	Repeats 6 ND Clk=0						0	X	0	0
	\$	\$	\$	\$	\$		A	R		\$

Mid Control Unit Counter, Config FF and Slip Generation. Up-Count starts with Q_{MC}=0

MA·MB·MC·MD	→ M _{HIT15} =0	N _{XT} C _{NT} =C _{NT+1}	M _{CONF FF}	JUG	Q _{M0}	Q _{M1}	Q _{M2}	Q _{M3}	Q _{M4}	L _{Q12}	R _{Q18}	das	If M _{HIT15} =1;
1 _{Init}	0	0	0	0	0	0	0	0	0	0	0	0	the next
2	1	0	0	0	0	1	0	0	0	0	1	1	Q _{MA} , Q _{MB} , Q _{MC} , Q _{MD} , M _{CONF FF} are:
3	0	1	0	0	0	1	0	1	0	0	0	0	
4	1	1	0	0	0	1	1	0	1	0	0	1	Q _{MA} =Q _{M0} ⊕Q ₁₂ ⊕Q ₁₈ (Top L, &RnLFSR)
5	0	0	1	0	0	1	0	1	0	0	0	1	Q _{MB} =Q _{M0} ⊕Q _{M1} ⊕Q ₁₂ (Top LnLFSR)
6	1	0	1	0	0	0	1	0	0	1	0	0	Q _{MC} = 0 (Count=4 to 15→12 max clocks)
7	0	1	1	0	0	1	0	0	0	0	1	0	Q _{MD} =JUG⊕Q _{M2} ⊕M ₃ ⊕M _{CONF FF}
8	1	1	1	0	0	0	0	0	0	1	1	0	M _{LeftSlip} =das⊕Q _{M4}
9	0	0	0	1	0	0	0	0	0	0	1	0	M _{RightSlip} =Notdas⊕Q _{M4} (inverse output)
10	1	0	0	1	0	0	0	0	0	0	0	0	φ = Often don't care
11	0	1	0	1	0	0	0	0	0	0	0	0	X = Unknown - Data Dependent
12	1	1	0	1	0	0	0	0	0	0	0	0	
13	0	0	1	1	0	0	0	0	0	0	0	0	
14	1	0	1	1	0	0	0	0	0	0	0	0	
15	0	1	1	1	0	0	0	0	0	0	0	0	φ &
16	1	1	1	1	1	0	X	X	X	X	X	X	φ M _{CONF FF} =(Old NotM _{CONF FF})⊕4thToggle
XX	X	X	X	X	X	X	φ	φ	φ	φ	φ	φ	φ
	\$	\$	\$	\$		\$	F				R	R	E



D) The Bottom Control Unit nLFSR – Fig. 15. Bot BRN=B0 ⊕ B1 ⊕ B2; $nx\text{BCONF FF} = ([\text{NOT}(\text{BCONF FF})] \oplus 4\text{THTOGGLE}) \cdot \text{BHIT15}$;
 else (BCONF FF)= old(BCONF FF); BHIT15=QBA·QBB·QBC·QBD

Bot Control Unit nLFSR At rising (P)Random Clocks- synced to PC until rising clock 7

@RiseClk	B0	B1	B2	B3	B4	B5	FB=B0⊕B2⊕SLIP⊕NFIK	ND	Q14∧BA=	CNTRL	SLIP	BotBRN	
1st	0	0	0	0	0	0	1 NFIK	1	0	0	0	0	
2	1	0	0	0	0	0	1	1	0	1	0	1	
3	1	1	0	0	0	0	1	1	0	0	0	0	
4	1	1	1	0	0	0	1⊕ Control Slip	1	0	1	0	1	
5	X	1	1	1	0	0	X	1	1	0	0	X	
6	X	X	1	1	1	0	X	1	X	1	X	X	
7	Repeats 6 ND Clk=0							0	X	0	0	0	X
	\$	\$	\$	\$	\$	\$		A	R		\$	\$	

Bot Control Unit Counter, BConfig FF and Slip Generation. Up-Count starts with Q_{MC}=0

@RiseClk	BA	BB	BC	BD	→ BHIT15=0 _{NXT}	C _{NT}	BCONF FF	JUG	QBO	QB1	QB2	QB3	QB4	QB5	LQ14	RQ16	das	If BHIT15=1;	
1st	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	the next QBA, QBB, QBC, QBD, BCONF FF are:
2	1	0	0	0	0	2	0	0	1	0	0	0	0	0	0	0	0	1	
3	0	1	0	0	0	3	0	1	1	1	0	0	0	0	0	0	0	1	
4	1	1	0	0	0	4	0	1	1	1	1	0	0	0	0	1	0	0	QBA=QB0⊕Q14⊕Q16 (Bot L, &RnLFSR)
5	0	0	1	0	0	5	0	1	X	1	1	1	0	0	1	1	0	0	QBB=QB0⊕QB1⊕Q14 (Bot LnLFSR)
6	1	0	1	0	0	6	0	1	X	φ	1	1	1	0	X	0	X	0	QBC= 0 (Count=4→15=12 max)
7	0	1	1	0	0	7	0	0	X	φ	φ	1	1	1	X	φ	X	φ	QBD=JUG⊕QB3⊕B4⊕BCONF FF
8	1	1	1	0	0	8	0	0	φ	φ	φ	φ	1	1	X	φ	φ	φ	BLeftSlip=das⊕QB5
9	0	0	0	1	0	9	0	0	φ	φ	φ	φ	φ	1	X	φ	φ	φ	MRightSlip=Notdas⊕QB5 (inverse output)
10	1	0	0	1	0	10	0	0	φ	φ	φ	φ	φ	φ	X	φ	φ	φ	φ = Often don't care
11	0	1	0	1	0	11	0	0	φ	φ	φ	φ	φ	φ	φ	X	φ	φ	X = Unknown - Data Dependent
12	1	1	0	1	0	12	0	0	φ	φ	φ	φ	φ	φ	φ	φ	φ	φ	
13	0	0	1	1	0	13	0	0	φ	φ	φ	φ	φ	φ	φ	φ	φ	φ	
14	1	0	1	1	0	14	0	0	φ	φ	φ	φ	φ	φ	φ	φ	φ	φ	
15	0	1	1	1	0	15	0	0	φ	φ	φ	φ	φ	φ	φ	φ	φ	φ	&
16	1	1	1	1	1	16	0	0	X	X	X	X	X	X	X	X	X	X	MCONF FF=(old NotMCONF FF)⊕ 4thToggle
XX	X	X	X	X	X	XX	X	X	φ	φ	φ	φ	φ	φ	φ	φ	φ	φ	φ
	\$	\$	\$	\$	\$		\$	F								R	R	E	

E) The 3 Pseudo Random Noise Sources - See Fig. 4DC1 and F)

@RiseClk **K(Juggle)**=OldK⊕OldB⊕OldE⊕QTA; **4th(Toggle)**=Old4th⊕OldB⊕OldM⊕QTA; **DAS**= OldDas ⊕ OldE ⊕ OldM ⊕ QTA.

1	0	Init	X	X	X	0	0	Init	X	X	X	0	0	Init	X	X	X	0
2	0	0	1	0	1	0	0	0	0	1	0	1	0	1	0	0	0	1
3	1	0	0	1	0	0	1	0	0	0	1	0	0	1	1	1	1	0
4	1	1	1	0	1	0	1	1	1	1	0	1	0	0	1	0	0	1
5	1	1	0	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0
6	1	1	0	1	1	0	0	0	0	0	1	1	1	1	0	1	1	1
7	1	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
	\$		F	F	B		\$	\$	F	F	B		\$		F	F	B	

F) See E), Fig. 4DC1 and 4DC2

FB taps into cells 0 2 4 5 7 8 B=Q0 ⊕ Q3 ⊕ Q6; E=Q1 ⊕ Q5 ⊕ Q7; M=Q2 ⊕ Q4 ⊕ Q8; A8 = Not (Q0 ∨ Q1 ∨ NotQ2)

@RiseClk 9Bit nLFSR sfr2 A B C3 E L3 M QTA Q2 Not(Q0 ∨ Q1 ∨ NotQ2) → A8

Init	1	1010	00001	1	100	1	000	0	101	0	0	1	1	0	0	0
2	1111	11011	1	110	0	111	1	111	1	1	1	1	1	1	0	0
3	1101	00110	0	111	1	101	0	000	0	0	0	1	1	1	0	0
4	0110	10011	1	000	0	101	0	111	1	1	1	1	0	1	0	0
5	1001	10010	1	110	0	001	1	010	1	0	0	1	0	1	1	0
6	0100	11001	0	000	0	110	0	011	0	1	0	0	1	0	0	0
7	1000	10111	1	101	0	001	1	011	0	0	0	1	0	1	0	0
8	1110	10100	1													
9	0111	01010	0													

During the 1st at least 9 clocks the 3LS bits ≠ 001 → A8=0; and the sfr2 string is constant =...011011011

\$\$\$ \$\$\$\$\$ A \$ \$ \$ B

G) EVNN Permutations in for the Splash Hybrid Filters—TMBEVNNs are generated in Fig. 10S3, 4thToggle is generated in Figs. 4DC1 & E.

TMB Control Unit Config Cntrls remain in initial condition, at least until the 16th Primary Clock (we know that one (P)Random Clock will be skipped). Each of the TMB Config FFs (see Figs 13, 14 & 15) are toggled when the Control Unit Counter hits 15 - I.C.'s is 0000., The TMB EVNN I.C. values are 0, 1, 0 respectively.

@RiseClk TEVNN=Old-TConfigCntrl ⊕ OldINV Brown MEVNN=Old-MConfigCntrl ⊕ OldINV Brown BEVNN=Old-BConfigCntrl ⊕ OldINV Brown 4thToggleEVNN - see E)

1	0	Init	X	X	1	Init	X	X	0	Init	X	X	0
2	1	0	1	1	1	0	1	1	1	0	1	1	0
3	0	0	0	0	0	0	0	0	0	0	0	0	1
4	0	0	0	0	0	0	0	0	0	0	0	0	1
5	1	0	1	1	1	0	1	1	1	0	1	1	0
6	x	0	x	x	x	0	x	x	x	0	x	x	0
7	x	0	x	x	x	0	x	x	x	0	x	x	x
	\$		B	J	\$		C	J	\$		D	J	E

Order in Top Matrix	4th	Top	Mid	Bot;	in Bot Matrix	Bot	4th	Top	Mid
Initial EVNN Value	0	0	1	0	0	0	0	1	1
@ Rising 1st Clock	0	1	1	1	1	0	1	1	1
@ Rising 2nd Clock	1	0	0	0	0	1	0	0	0
@ Rising 3rd Clock	1	0	0	0	0	1	0	0	0

H) Tier Clocks

The Super Tier is driven by the Primary Clock, the TMB clocks are regulated by the Configs, the Inv BRN, and the (P)Random

Clock.

With this weak key, the Y1 configuration is constant for the first 15 clocks, no slip pulses are generated as all up-counters initially are forced to "0"; they are normally triggered randomly from 4 to 15 (between 1 and 12 Primary Clocks between slips) caused at random times from each of the counters, i.e., change of Config averaging 1 in 4 Clocks.

@RiseClk

	$T_{TCIKEN} = \text{Not}(\text{OldInvBRN} \cdot \text{OldY2});$			$M_{TCIKEN} = \text{Not}(\text{OldInvBRN} \cdot \text{OldY3});$			$B_{TCIKEN} = \text{Not}(\text{OldInvBRN} \cdot \text{OldY1})$		
1 Init	1	~(X	X)	1	~(X	X)	1	~(X	X)
2	1	~(1	0)	1	~(1	0)	0	~(1	1)
3	1	~(0	0)	1	~(0	0)	1	~(0	1)
4	1	~(0	0)	1	~(0	0)	1	~(0	1)
5	1	~(1	0)	1	~(1	0)	0	~(1	1)
6	1	~(X	0)	1	~(X	0)	X	~(X	1)
7	1	~(X	0)	1	~(X	0)	X	~(X	1)
	\$	J	A	\$	J	A	\$	J	A

J) Calculating TMB Browns See Fig. 10S3

$\text{NotNC}_{\text{rntComp}} = \text{NotNC}_{\text{rnt}} = 0$, except for 6th Clk; YSense (YS) insures that 2 of 3 TMB Browns are active on missing (P)Random Clock

@RiseClk $YS = (T2 \cdot M2 \cdot B2); T2 = (T_{BRN} + Y1 + \sim NC_{\text{rnt}}); M2 = (M_{BRN} + Y2 + \sim NC_{\text{rnt}}); B2 = (B_{BRN} + Y3 + \sim NC_{\text{rnt}})$

1 Init	0	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0
2	1	1	1	1	1	1	1	0	1	1	0	0	1	1	0	0
3	0	1	1	0	1	0	1	0	1	1	0	0	0	0	0	0
4	0	1	0	1	1	1	1	0	0	0	0	0	1	1	0	0
5	X	1	X	X	1	X	1	0	X	X	0	0	X	X	0	0
6	1	1	1	1	1	X	1	1	1	X	0	1	1	X	0	1
7	X	1	X	X	1	X	1	0	X	X	0	0	X	X	0	0
	\$				B	A	A		C	A	A		D	A	A	

@RiseClk	$T3 = YS \cdot Y1$			$M3 = YS \cdot Y2$			$B3 = YS \cdot Y3$			$M_{\text{BROWN}} = \text{OLD}(T3 \oplus M2)$			$B_{\text{BROWN}} = \text{OLD}(M3 \oplus B2)$			$T_{\text{BROWN}} = \text{OLD}(B3 \oplus T2)$			
1 Init	0	0	1	0	0	0	0	0	0	1 _{INIT}	X	X	0	X	X	1	X	X	
2	1	1	1	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	1
3	0	0	1	0	0	0	0	0	0	0	0	1	1	1	0	1	1	0	1
4	0	0	1	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0	1
5	X	X	1	0	X	0	0	X	0	0	0	0	0	1	0	1	1	0	1
6	1	1	1	0	1	0	0	1	0	0	X	X	X	X	0	X	1	0	1
7	0	X	0	X	X	1	0	X	0	0	1	1	1	0	1	1	0	1	
	\$	A	\$	A	\$	A	\$	A	\$		\$		\$		\$				

@RiseClk $\text{INV BRN} = \text{NOT}(T_{\text{BRN}} \oplus M_{\text{BRN}} \oplus B_{\text{BRN}})$

	T_{BRN}	M_{BRN}	B_{BRN}	Y1	Y2	Y3	INV BRN
1 Init	0	0	0	1	0	0	1
2	1	1	1	1	0	0	0
3	0	1	0	1	0	0	0
4	1	0	1	1	0	0	1
5	X	X	X	1	0	0	X
6	X	X	X	0	1	0	X
7	X	X	X	0	1	0	X
	B	C	D	A	A	A	\$

K) Splash Vector Select See Fig. 18SSEL

@RiseClk	Q1	$\text{NOT} \oplus$	T-15 = DA → Nxt Q0	JUG ⊕	T-31 ⊕	Q0 = DB → Nxt Q1	Q1	Q0	Active Vectors	
1 Init	0 _{Init}	0	1	0	0	0 _{Init}	0	1 _{Init}	D → A	
2	0	1	0	0	1	1	0	1	C → D	
3	0	1	0	1	0	0	1	0	D → A	
4	1	X	X	1	X	0	X	1	0	B → C
5	X	X	X	1	X	X	X	X	X	X → X
6	X	X	X	1	X	X	X	X	X	X → X
7	X	X	X	X	X	X	X	X	X	X → X
	SSEL	CHRN	\$	E	CHRN	SSEL	\$	\$	\$	

00 → Top D Bot A 01 → Top C Bot D 10 → Top B Bot C 11 → Top A Bot C



Feedback Masks – Tapping into Cells

STier L&R IntFB Mask - 1101 0011 0001 1100 1000 0010 0100 1000 NFIX FBs or Slip=1 Figs 28 & 29 0,1,3,6,7,11,12,13 0,6,9,12

TTier L&R IntFB Mask - 1001 1010 0110 0101 0010 1111 0100 1010 NFIX FBs or Slip=1 Figs 19 & 24 0,3,4,6,9,10 0,2 5,7,8,9,10,12,15,17

MTier L&R IntFB Mask - 1001 0101 1001 1110 1010 1001 1001 0100 NFIX FBs or Slip=1 Figs 23 & 20 0,3,5,7,8,11,12,13,14,16 0,2,5,6,9,11

BTier L&R IntFB Mask - 1110 0011 0001 0001 0100 1001 0110 1100 NFIX FBs or Slip=1 Figs 21 & 22 0,1,2,6,7,11 0,2,5,8,10,11,13,14

Top Matrix EVNN Vectors (to be copied into Top Hybrid Filter function)

4th	T	M	B	EVNN-	4TMB	4TMB	4TMB	4TMB	4TMB	4TMB	4TMB	4TMB
0	0	0	0	TM EVN	0000	0000	0000	0000	0000	0000	0000	0000
0	0	0	1	TM EVN	0001	0001	0001	0001	0001	0001	0001	0001
0	0	1	0	TM EVN	0010	0010	0010	0010	0010	0010	0010	0010
0	0	1	1	TM EVN	0011	0011	0011	0011	0011	0011	0011	0011
0	1	0	0	TM EVN	0100	0100	0100	0100	0100	0100	0100	0100
0	1	0	1	TM EVN	0101	0101	0101	0101	0101	0101	0101	0101
0	1	1	0	TM EVN	0110	0110	0110	0110	0110	0110	0110	0110
0	1	1	1	TM EVN	0111	0111	0111	0111	0111	0111	0111	0111
1	0	0	0	TM EVN	1000	1000	1000	1000	1000	1000	1000	1000
1	0	0	1	TM EVN	1001	1001	1001	1001	1001	1001	1001	1001
1	0	1	0	TM EVN	1010	1010	1010	1010	1010	1010	1010	1010
1	0	1	1	TM EVN	1011	1011	1011	1011	1011	1011	1011	1011
1	1	0	0	TM EVN	1100	1100	1100	1100	1100	1100	1100	1100
1	1	0	1	TM EVN	1101	1101	1101	1101	1101	1101	1101	1101
1	1	1	0	TM EVN	1110	1110	1110	1110	1110	1110	1110	1110
1	1	1	1	TM EVN	1111	1111	1111	1111	1111	1111	1111	1111

Bottom Matrix EVNN Vectors (to be copied into Bottom Hybrid Filter function)

BOT	4	T	M	EVNN-	B4TM	B4TM	B4TM	B4TM	B4TM	B4TM	B4TM	B4TM
0	0	0	0	BM EVN	0000	0000	0000	0000	0000	0000	0000	0000
0	0	0	1	BM EVN	0001	0001	0001	0001	0001	0001	0001	0001
0	0	1	0	BM EVN	0010	0010	0010	0010	0010	0010	0010	0010
0	0	1	1	BM EVN	0011	0011	0011	0011	0011	0011	0011	0011
0	1	0	0	BM EVN	0100	0100	0100	0100	0100	0100	0100	0100
0	1	0	1	BM EVN	0101	0101	0101	0101	0101	0101	0101	0101
0	1	1	0	BM EVN	0110	0110	0110	0110	0110	0110	0110	0110
0	1	1	1	BM EVN	0111	0111	0111	0111	0111	0111	0111	0111
1	0	0	0	BM EVN	1000	1000	1000	1000	1000	1000	1000	1000
1	0	0	1	BM EVN	1001	1001	1001	1001	1001	1001	1001	1001
1	0	1	0	BM EVN	1010	1010	1010	1010	1010	1010	1010	1010
1	0	1	1	BM EVN	1011	1011	1011	1011	1011	1011	1011	1011
1	1	0	0	BM EVN	1100	1100	1100	1100	1100	1100	1100	1100
1	1	0	1	BM EVN	1101	1101	1101	1101	1101	1101	1101	1101
1	1	1	0	BM EVN	1110	1110	1110	1110	1110	1110	1110	1110
1	1	1	1	BM EVN	1111	1111	1111	1111	1111	1111	1111	1111

Starting at all Zero Key Initial Condition after Global Pre/Reset – 128/160 Bit All-Zero Initialization Key

Super Tier FB In - 0000 0000 0000 0000 0000 0000 0000 0000

Super Tier nLFSRs - 0000 0000 0000 0000 0000 0000 0000 0000

Super Tier Output - 0000 0000 0000 0000 0000 0000 0000 0000

Lower FB In - 0000 0000 0000 0000 0000 0000 0000 0000

Top Tier nLFSRs +ROT- 0000 0000 0000 0000 0000 0000 0000 0000

Middle Tier nLFSRs - 0000 0000 0000 0000 0000 0000 0000 0000



Bottom Tier nLFSRs - 0000 0000 0000 0000 0000 0000 0000 0000

3 Tier MAJ Output - 0000 0000 0000 0000 0000 0000 0000 0000

3 Tier Combiner Output - 0000 0000 0000 0000 0000 0000 0000 0000

4 Tier Combiner Output - 0000 0000 0000 0000 0000 0000 0000 0000

13 RROT Feedback - 0000 0000 0000 0000 0000 0000 0000 0000

Top Store In - 0000 0000 0000 0000 0000 0000 0000 0000

Top Store Out - 0000 0000 0000 0000 0000 0000 0000 0000

Top XOR Out - 0000 0000 0000 0000 0000 0000 0000 0000

Top D-Splash Matrix Out - 0000 0000 0000 0000 0000 0000 0000 0000

Top Hybrid Filter Out - 0000 0000 0000 0000 0000 0000 0000 0000 EVNN Permutes=0

7 LROT Feedback - 0000 0000 0000 0000 0000 0000 0000 0000

Intermediate Store In - 0000 0000 0000 0000 0000 0000 0000 0000

Intermediate Store Out - 0000 0000 0000 0000 0000 0000 0000 0000

Intermediate XOR Out - 0000 0000 0000 0000 0000 0000 0000 0000

Bot Splash Matrix Out - 0000 0000 0000 0000 0000 0000 0000 0000

Bot Hybrid Filter Out - 0000 0000 0000 0000 0000 0000 0000 0000

Bottom Store In - 0000 0000 0000 0000 0000 0000 0000 0000

Bottom Store Out - 0000 0000 0000 0000 0000 0000 0000 0000

Cipher Mask [Bot XOR] - 0000 0000 0000 0000 0000 0000 0000 0000

Result = Cipher Mask \oplus Message (Which could be the last zero word of the secret key)

SuperMIX In - 0000 0000 0000 0000 0000 0000 0000 0000 IntrmdtXOR \oplus Bot Fltr

SuperMIX Out - 0000 0000 0000 0000 0000 0000 0000 0000

MACMIX Out - 0000 0000 0000 0000 0000 0000 0000 0000 fMMX (CipherMask)

Super Tier FB Store In - 0000 0000 0000 0000 0000 0000 0000 0000

Result Store In - 0000 0000 0000 0000 0000 0000 0000 0000

Result Store Out - 0000 0000 0000 0000 0000 0000 0000 0000

Lower FB Store In - 0000 0000 0000 0000 0000 0000 0000 0000 CipherMask \oplus RsltStore



After 1st Clock 32 Bit Word Manipulator Condition

Super Tier FB In - 0000 0000 0000 0000 0000 0000 0000 0000 STT1
 STierOLDnLFSRsRrotD1- 0000 0000 0000 0000 0000 0000 0000 0000 Insert LS zeros del msb in each nLFSR
 STier L&R IntFB Mask - 1101 0011 0001 1100 1000 0010 0100 1000 ←LQ15 NF15 SFB15 FG28-29

STier nLFSRs after FBs - 1101 0011 0001 1100 1000 0010 0100 1000 LQ15=0; RQ15=0
 STier nLFSRs 7L ROT - 1000 1110 0100 0001 0010 0100 0110 1001 CtrlX 7 L/H LS bits Insert at end (MS)
 Super Tier Out - 0101 1101 0101 1101 1010 0110 0010 0001 Super Tier Rotated Image always XORed to nLFSRs Output

Lower FB In - 0000 0000 0000 0000 0000 0000 0000 0000 TTT1 TIER CLOCK
 ENABLED
 TTierOLDnLFSRsRrotD1- 0000 0000 0000 0000 0000 0000 0000 0000 Insert LS zeros del MSbs in each IF CLOCK
 ENABLED
 TTier L&R IntFB Mask - 1001 1010 0110 0101 0010 1111 0100 1010 ←LRQ12/18 NF LWRfb LRslip FG 19/24

T Tier nLFSRs after FBs - 1001 1010 0110 0101 0010 1111 0100 1010 XOR Sum of 3 above; Q12=0; Q18=0
 T Tier nLFSRs 1L ROT - 0011 0100 1100 1010 0101 1110 1001 0101 Brown Enabled - CtrlX 1 LH MSbit Insert in LS
 Top Tier Out - 1010 1110 1010 1111 0111 0001 1101 1111 Sum if Brown Enabled, else copy XOR Sum of 3 above

Lower FB In - 0000 0000 0000 0000 0000 0000 0000 0000 MMM1 TIER CLOCK
 ENABLED
 MTierOLDnLFSRsRrotD1- 0000 0000 0000 0000 0000 0000 0000 0000 LInsert LS zeros del MSbs in each IF CLOCK ENABLED
 MTier L&R IntFB Mask - 1001 0101 1001 1110 1010 1001 1001 0100 ←LRQ17/13 NF LWRfb LRslip FG23-20 IF CLK EN
BIG MISTAKE MID BROWN = 1XXX
 M Tier nLFSRs after FBs- 1001 0101 1001 1110 1010 1001 1001 0100 XOR Sum of 3 above Q17=0; Q13=0
~~M Tier nLFSRs 3L ROT - 1 0101 1001 1110 1010 1001 1001 0100~~ Brown Not Enabled If Yes CtrlX 3 RH MSbits Insert in LS
 Middle Tier Out - 1001 0101 1001 1110 1010 1001 1001 0100 Sum if Brown Enabled, else copy

Lower FB In - 0000 0000 0000 0000 0000 0000 0000 0000 BBB1 TIER CLOCK
NOT EN
 BTierOLDnLFSRsRrotD1- 0000 0000 0000 0000 0000 0000 0000 0000 LInsert LS zeros del MSbs in each IF CLOCK ENABLED
~~BTier L&R IntFB Mask - 1110 0011 0001 0001 0100 1001 0110 1100~~ ←LRQ14/16 LWRfb LRslip FG21-22 IF CLK ENABLED

B Tier nLFSRs after FBs- 0000 0000 0000 0000 0000 0000 0000 0000 Q14=0; Q16=0
~~B Tier nLFSRs 5L ROT - 0000 0000 0000 0000 0000 0000 0000 0000~~ Brown Not Enabled If Yes CtrlX 5 RH MSbits Insert in LS

Bottom Tier Out - 0000 0000 0000 0000 0000 0000 0000 0000 Sum if Brown Enabled, else copy
 Middle Tier Out - 1001 0101 1001 1110 1010 1001 1001 0100 Copied
 Top Tier Out - 1010 1110 1010 1111 0111 0001 1101 1111 Copied

3 Tier MAJ Output - 1000 0100 1000 1110 0010 0001 1001 0100
 MAJ Out 5RROT- 1010 0100 0010 0100 0111 0001 0000 1100 The Maj Out Is 5 Right Rotated
 Super Tier Out - 0101 1101 0101 1101 1010 0110 0010 0001 Copied

4 Tier Combiner Output - 0111 1101 1111 0111 1111 0110 1011 1001 XOR of the MAJ, 5 R Rot MAJ and Super Tier

shove into LH
 13 RROT LWR FB - 0000 0000 0000 0000 0000 0000 0000 0000 Right Rotating Template 13 RH bits and
 FB = /all Zero Take the LWR FB, 13>>> Here the LWR

Top Store In - 0111 1101 1111 0111 1111 0110 1011 1001 4tierCombiner@13ROT FB
 Top Store Out - 0000 0000 0000 0000 0000 0000 0000 0000
 4 Tier Combiner Output - 0111 1101 1111 0111 1111 0110 1011 1001 Copied
 Top XOR Out - 0111 1101 1111 0111 1111 0110 1011 1001 4tierCombiner@TopStoreOut

0 1 1 1 1 1 0 1 1 1 1 1 0 1 1 1 1 1 1 0 1 1 0 1 0 1 1 1 0 0 1

I00	I01	I02	I03	I04	I05	I06	I07	I08	I09	I10	I11	I12	I13	I14	I15	I16	I17	I18	I19	I20	I21	I22	I23	I24	I25	I26	I27	I28	I29	I30	I31
A09	A18	A05	A11	A22	A12	A30	A19	A07	A15	A31	A25	A28	A24	A06	A03	A17	A13	A27	A23	A01	A02	A26	A21	A04	A20	A08	A16	A00	A14	A10	A29
B30	B15	B06	B12	B25	B18	B16	B09	B19	B07	B03	B31	B00	B29	B27	B21	B14	B28	B24	B17	B23	B05	B10	B02	B11	B22	B13	B26	B20	B08	B04	B01
C19	C07	C14	C29	C03	C27	C00	C13	C25	C16	C15	C30	C20	C01	C26	C31	C08	C06	C02	C04	C09	C18	C12	C10	C21	C11	C22	C05	C24	C23	C28	C17
O00	O01	O02	O03	O04	O05	O06	O07	O08	O09	O10	O11	O12	O13	O14	O15	O16	O17	O18	O19	O20	O21	O22	O23	O24	O25	O26	O27	O28	O29	O30	O31

0 1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1 1 1 0 0 1 1 0 1 1 1 1 0 1 1 1

C-Vector – Splash out- 0111 1111 1000 1111 1110 0110 1111 0111 TSPL-15=1; TSPL-31=1

4th T M B EVNN- 4TMB 4TMB 4TMB 4TMB 4TMB 4TMB 4TMB 4TMB
 0 1 1 1 1 TM EVN 0111 0111 0111 0111 0111 0111 0111 0111 Top EVNN 0111 Vector to MAJ Filter

H-1 to MAJ to H 1>>>- 1011 1111 1100 0111 1111 0011 0111 1011 Splash Inputs to MAJ H-1 & H-2 see Figs. 32
 H-2 to MAJ to H 2>>>- 1101 1111 1110 0011 1111 1001 1011 1101

Hybrid MAJ Out- 1111 1111 1110 0111 1111 0011 0111 1111
 C Vector – Splash Out- 0111 1111 1000 1111 1110 0110 1111 0111 Copied
 H+1 to 3XOR 1<<<- 1111 1111 0001 1111 1100 1101 1110 1110

Top Hybrid Filter Out - 0111 1111 0111 0111 1101 1000 0110 0110 XOR of 3 Previous Values;
 7 LROT LWR Feedback - 0000 0000 0000 0000 0000 0000 0000 0000 Take 7 LS bits append to RH vector

Intermediate Store In - 0111 1111 0111 0111 1101 1000 0110 0110 FilterOut@7ROT LWR FB

Intermediate Store Out - 0000 0000 0000 0000 0000 0000 0000 0000
 Intermediate XOR Out - 0111 1111 0111 0111 1101 1000 0110 0110 FilterOut@IntrmdtStoreOut

Bot Splash Matrix Out- 0111 1111 0111 0111 1101 1000 0110 0110 D Vector Straight
 Out

BOT 4 T M EVNN- B4TM B4TM B4TM B4TM B4TM B4TM B4TM B4TM
 1 0 1 1 BM EVN 1011 1011 1011 1011 1011 1011 1011 1011 Bot EVNN 1011 Vector to MAJ Filter

H-1 to MAJ to H 1>>>- 0011 1111 1011 1011 1110 1100 0011 0011 Splash Inputs to MAJ H-1 & H-2 see Fig. 32
 H-2 to MAJ to H 2>>>- 1001 1111 1101 1101 1111 0110 0001 1001

Hybrid MAJ Out- 1011 1111 1011 1011 1111 1110 0011 1011
 D Vec Bot Splash Out- 0111 1111 0111 0111 1101 1000 0110 0110 Copied
 H+1 to 3XOR 1<<<- 1111 1110 1110 1111 1011 0000 1100 1100

Bot Hybrid Out/ Store In- 0011 1110 0010 0011 1001 0110 1001 0001 XOR of 3 Previous Values
 Bottom Store Out - 0000 0000 0000 0000 0000 0000 0000 0000

Cipher Mask-	0011 1110 0010 0011 1001 0110 1001 0001	FilterOut⊕BottomStoreOut
Message In-	0000 0000 0000 0000 0000 0000 0000 0000	
Result & Result Store In-	0011 1110 0010 0011 1001 0110 1001 0001	Message⊕Cipher Mask
Intermediate XOR Out -	0111 1111 0111 0111 1101 1000 0110 0110	Copied
Bot Hybrid Filter Out -	0011 1110 0010 0011 1001 0110 1001 0001	Copied
SuperMIX In -	0100 0001 0101 0100 0100 1110 1111 0111	IntrmdtXOR ⊕ Bot Ftr
SuperMIX Out -	1111 1110 0010 1000 1010 0010 0010 0111	8RROT after Reversed Nibble
Result is MAC MIX In-	0011 1110 0010 0011 1001 0110 1001 0001	Copied
MACMIX Out -	1100 0111 0100 1100 1001 0110 1001 1000	MMX (Result)
SuperMIX Out -	1111 1110 0010 1000 1010 0010 0010 0111	Copied
Super Tier FB Store In -	0011 1001 0110 0100 0011 0100 1011 1111	SupMIXout⊕MACMIXout
Result -	0011 1110 0010 0011 1001 0110 1001 0001	Copied
Result Store Out -	0000 0000 0000 0000 0000 0000 0000 0000	Old Result StoreIn
Lower FB Store In -	0011 1110 0010 0011 1001 0110 1001 0001	Old Result ⊕ New Result
TCU Fig 13 &	nnnn nnnn nnnn VnVn nVnn nnnn nnnn nnn3	Q18 RT to MCU Fig 14; Q13 RMid to Q12 LTop to MCU V Q17 LMid to BCU Q16 RB to Bottom Control Unit Q14 LBot to TCU

Next Splash Select K) . 8 TSPL-15 = 1; TSPL-31= 1; to Splash Select Fig. 18SSEL.

Control signals for Next Round-

Note: B) Refers to logic tables in the Random controller Section, Page 4; Figs. refer to circuit diagrams in [zk-ccc].

- B) Super LQ₁₅ = 0 to Top Cntrl Unit (TCU); Super RQ₁₅ = 0 to TCU Fig. 13
- B) Top Left Q₁₂ = 0 to Mid Cntrl Unit Slip(MCU); Top Right Q₁₈ = 0 to MCU Fig. 14
- C) Mid Left Q₁₇ = 0 to Top Cntrl Unit Slip(TCU); Mid Right Q₁₃ = 0 to TCU Fig. 15
- D) Bot Left Q₁₂ = 0 to Bot Cntrl Unit Slip(BCU); Top Right Q₁₈ = 0 to BCU Fig. 14

Slip Signals to the TMB Control Units are randomly active on every (P)Random Clock.

All other signals in B), C) and D) only affect random up count preset of Control Unit Counters Figs. 13, 14 & 15.

G) EVNN Matrix Control Signals Vectors on Page 9

Order in Top Matrix 4th Top Mid Bot; in Bot Matrix Bot 4th Top Mid
Next 1 0 0 0 0 0 1 0 0

J) TMB Browns Page 8 TBrown MBrown BBrown

Next 1 0 1

H) TMB Tier Clock Signals Page 8 TTier MTier BTier

Next 1 1 1

K) Splash Vector Select Page 8 see circuit diagram in Fig. 18SSEL in [zk-ccc].

TSPL-15=1; TSPL-31=1 → Next D→A

After 2nd Clock 32 Bit Word Manipulator Condition

Super Tier FB In -	0011 1001 0110 0100 0011 0100 1011 1111	STT2 STFB in ALWAYS
STierOLDnLFSRsRrotD1-	0110 1001 1000 1110 0100 0001 0010 0100	Insert LS zeros del msb in each nLFSR ←
STier L&R IntFB Mask -	1101 0011 0001 1100 1000 0010 0100 1000	←LQ15 NF15 SFB15 FG28-29
STier nLFSRs after FBs -	0101 0000 1110 1010 0111 0101 1001 1011	LQ ₁₅ =0; RQ ₁₅ =1 ↓D
STier nLFSRs 7L ROT -	0111 0101 0011 1010 1100 1101 1010 1000	CtrlX 7 L/H LS bits Insert at end (MS)
Super Tier Out - nLFSRs Output	0010 0101 1101 0000 1011 1000 0011 0011	Super Tier Rotated Image always XORed to
Lower FB In - ENABLED ←	0011 1110 0010 0011 1001 0110 1001 0001	TTT2 LWRFB in if CLK
TTierOLDnLFSRsRrotD1-	0100 1101 0011 0010 1001 0111 1010 0101	Insert LS zeros del MSBs IF CLOCK ENABLED
TTier L&R IntFB Mask -	1001 1010 0110 0101 0010 1111 0100 1010	←LRQ12/18 NF LWRfb LRslip FG 19/24
T Tier nLFSRs after FBs -	0111 0011 0001 0001 0000 0001 0011 0100	XOR Sum of 3 above; Q ₁₂ =0; Q ₁₈ =0 ↓D
T Tier nLFSRs 1L ROT -	1110 0110 0010 0010 0000 0010 0110 1000	Brown Enabled -CtrlX 1 LH MSbit Insert in LS
Top Tier Out - above	1001 0101 0011 0011 0000 0011 0101 1100	Sum if Brown Enabled, else copy XOR Sum of 3
Lower FB In - ENABLED ←	0011 1110 0010 0011 1001 0110 1001 0001	MMM2 TIER IF CLK
MTierOLDnLFSRsRrotD1-	0100 1010 1100 1111 0101 0100 1100 1010	LInsert LS zeros del MSBs IF CLOCK ENABLED
MTier L&R IntFB Mask -	1001 0101 1001 1110 1010 1001 1001 0100	←LRQ17/13 NF LWRfb LRslip FG23-20 IF CLK EN
M Tier nLFSRs after FBs-	0111 0100 1110 1100 1100 0010 0101 1011	XOR Sum of 3 above Q ₁₇ =1; Q ₁₃ =1 ↓D
M Tier nLFSRs 3L ROT in LS	1010 0101 1010 0101 1010 0101 1010 0100	Brown Not Enabled If Yes CtrlX 3 RH MSbits Insert
Middle Tier Out -	0111 0100 1110 1100 1100 0010 0101 1011	Sum if Brown Enabled, else copy
Lower FB In - ENABLED ←	0011 1110 0010 0011 1001 0110 1001 0001	BBB2 TIER IF CLK
BTierOLDnLFSRsRrotD1-	0000 0000 0000 0000 0000 0000 0000 0000	LInsert LS zeros del MSBs IF CLOCK
BTier L&R IntFB Mask -	1110 0011 0001 0001 0100 1001 0110 1100	←LRQ14/16 LWRfb LRslip FG21-22 IF CLK EN
B Tier nLFSRs after FBs-	1101 1101 0011 0010 1101 1111 1111 1101	XOR Sum of 3 above Q ₁₄ =1; Q ₁₆ =1 ↓D
B Tier nLFSRs 5L ROT in LS	1010 0110 0101 1011 1111 1111 1011 1011	Brown Enabled If Yes CtrlX 5 RH MSbits Insert
Bottom Tier Out -	0111 1011 0110 1001 0010 0000 0100 0110	Sum if Brown Enabled, else copy
Middle Tier Out -	0111 0100 1110 1100 1100 0010 0101 1011	Copied
Top Tier Out -	1001 0101 0011 0011 0000 0011 0101 1100	Copied
3 Tier MAJ Output -	0111 0101 0110 1001 0000 0010 0101 1110	
MAJ Out 5RROT-	1111 0011 1010 0001 0100 1000 0001 0010	The Maj Out Is 5 Right Rotated
Super Tier Out -	0010 0101 1101 0000 1011 1000 0011 0011	Copied
4 Tier Combiner Output -	1010 0011 0001 1000 1111 0010 0111 1111	XOR of the MAJ, 5 R Rot MAJ and Super Tier
13 RROT LWR FB -	1011 0100 1000 1001 1111 0001 0001 1100	Take the LWR FB, 13>>>



Top Store In -	0010 1001 1011 0010 1001 0101 1111 0010	4tierCombiner@I3ROT FB	↓ <u>D</u>
Top Store Out -	0111 1101 1111 0111 1111 0110 1011 1001	OLDTop Store IN	← ↓
4 Tier Combiner Output -	1010 0011 0001 1000 1111 0010 0111 1111	Copied	
Top XOR Out -	1101 1110 1110 1111 0000 0100 1100 0110	4tierCombiner@TopStoreOut	
D-Vector – Splash out- ← _D VEC	<u>1</u> 101 1110 1110 1111 0000 0100 1100 011 <u>0</u>	CopiedTSPL-15= <u>1</u> ; TSPL-31= <u>0</u>	

FortressGB

4th T M B EVNN- 4TMB 4TMB 4TMB 4TMB 4TMB 4TMB 4TMB 4TMB
 1 0 0 0 TM EVN 1000 1000 1000 1000 1000 1000 1000 1000 **Top EVNN 1000 to MAJ Filter From PG 9**
 H-1 to MAJ to H 1>>>- **0110 1111 0111 0111 1000 0010 0110 0011** **Splash Inputs to MAJ H-1 & H-2 see Fig. 32**
 H-2 to MAJ to H 2>>>- **1011 0111 1011 1011 1100 0001 0011 0001**

Hybrid MAJ Out- 1010 1111 1011 1011 1000 0000 0010 0001 MAJ(TM EVNN, H-1, H-2)
D Vector – Splash Out- 1101 1110 1110 1111 0000 0100 1100 0110 **Copied**
 H+1 to 3XOR 1<<<- 1011 1101 1101 1110 0000 1001 1000 1101

Top Hybrid Filter Out - 1100 1100 1000 1010 1000 1101 0110 1010 XOR of 3 Previous Values;
 7 LROT LWR FB- 0001 0001 1100 1011 0100 1000 1001 1111 **Take 7 LS bits append to RH vector**

Intermediate Store In - 1101 1101 0100 0001 1100 0101 1111 0101 FilterOut ⊕ 7ROT LWR FB

Intermediate Store Out - 0111 1111 0111 0111 1101 1000 0110 0110 **Copied** ← Old Int Store In
 Top Hybrid Filter Out - 1100 1100 1000 1010 1000 1101 0110 1010 **Copied**

Intermediate XOR Out - 1011 0011 1111 1101 0101 0101 0000 1100 FilterOut ⊕ IntrmdtStoreOut

1 0 1 1 0 0 1 1 1 1 1 1 1 0 1 0 1 0 1 0 1 0 1 0 0 0 0 1 1 0 0

I00	I01	I02	I03	I04	I05	I06	I07	I08	I09	I10	I11	I12	I13	I14	I15	I16	I17	I18	I19	I20	I21	I22	I23	I24	I25	I26	I27	I28	I29	I30	I31
A09	A18	A05	A11	A22	A12	A30	A19	A07	A15	A31	A25	A28	A24	A06	A03	A17	A13	A27	A23	A01	A02	A26	A21	A04	A20	A08	A16	A00	A14	A10	A29
B30	B15	B06	B12	B25	B18	B16	B09	B19	B07	B03	B31	B00	B29	B27	B21	B14	B28	B24	B17	B23	B05	B10	B02	B11	B22	B13	B26	B20	B08	B04	B01
C19	C07	C14	C29	C03	C27	C00	C13	C25	C16	C15	C30	C20	C01	C26	C31	C08	C06	C02	C04	C09	C18	C12	C10	C21	C22	C05	C24	C23	C28	C17	
O00	O01	O02	O03	O04	O05	O06	O07	O08	O09	O10	O11	O12	O13	O14	O15	O16	O17	O18	O19	O20	O21	O22	O23	O24	O25	O26	O27	O28	O29	O30	O31

1 0 1 1 0 1 0 1 0 1 0 1 0 1 1 1 0 0 0 1 0 1 0 1 1 1 0 0 1 0 1 1

A VecBot Splash Out- 1011 0101 0101 0111 0001 0101 1100 1011

BOT 4 T M EVNN- B4TM B4TM B4TM B4TM B4TM B4TM B4TM B4TM
 0 1 0 0 BM EVN 0100 0100 0100 0100 0100 0100 0100 0100 **Bot EVNN 0100 Vector to MAJ Filter**
 H-1 to MAJ to H 1>>>- 1101 1010 1010 1011 1000 1010 1110 0101 **Splash Inputs to MAJ H-1 & H-2 see Fig. 32**
 H-2 to MAJ to H 2>>>- 1110 1101 0101 0101 1100 0101 0111 0010

Hybrid MAJ Out- 1100 1100 0100 0101 1100 0100 0110 0100 MAJ(BM EVNN, H-1, H-2)
A Vec Bot Splash Out- 1011 0101 0101 0111 0001 0101 1100 1011 **Copied**
 H+1 to 3XOR 1<<<- 0110 1010 1010 1110 0010 1011 1001 0111

Bot Hybrid Out/ Store In- 0001 0011 1011 1100 1111 1010 0011 1000 XOR of 3 Previous Values ↓D
 Bottom Store Out - 0011 1110 0010 0011 1001 0110 1001 0001 **Copied** ← ←

Cipher Mask- 0010 1101 1001 1111 0110 1100 1010 1001 BotHybridOut ⊕ BotStoreOut
 Message In- 0000 0000 0000 0000 0000 0000 0000 0000
 Result & Result Store In- 0010 1101 1001 1111 0110 1100 1010 1001 **Copied if Message = 0 Message ⊕ Cipher Mask ↓D**

Intermediate XOR Out - 1011 0011 1111 1101 0101 0101 0000 1100 **Copied**
Bot Hybrid Filter Out - 0001 0011 1011 1100 1111 1010 0011 1000 **Copied**
 SuperMIX In - 1010 0000 0100 0001 1010 1111 0011 0100 IntrmdtXOR ⊕ Bot Fltr
 SuperMIX Out - 1100 0010 0101 0000 0010 1000 0101 1111 8RROT after Reversed Nibble

Result (MAC MIX In)- 0010 1101 1001 1111 0110 1100 1010 1001 **Copied** ↓D

MACMIX Out - 0100 1011 1001 1111 0110 0011 0101 1001 fMMX (Result)



SuperMIX Out - 1100 0010 0101 0000 0010 1000 0101 1111
 Super Tier FB Store - 1000 1001 1100 1111 0100 1011 0000 0110 SupMIXout@MACMIXout ↓D

Result - 0010 1101 1001 1111 0110 1100 1010 1001 Copied ↓D
 Result Store Out- 0011 1110 0010 0011 1001 0110 1001 0001 Old Result StoreIn ←
 Lower FB Store In - 0001 0011 1011 1100 1111 1010 0011 1000 Old Result @ New Result ↓D
 nnnn nnnn nnnn VnVn nVnn nnnn nnnn nnn3 Q18 RT to MCU Fig 14; Q13 RMid to TCU Fig 13 &
 Q12 LTop to MCU | V | Q17 LMid to BCU Q16 RB to Bottom Control Unit
 Q14 LBot to TCU

SuperLeftQ₁₅ = 0 to Top Control Unit (TCU); Super Right Q₁₅ = 1 to TCU- Fig. 13
 Top Left Q₁₂ = 0 to Mid Control Unit (MCU)-Slip; Top Right Q₁₈ = 0 to MCU- Fig. 14
 Mid Left Q₁₇ = 1 to Top Control Unit (TCU)-Slip; Top Right Q₁₃ = 1 to TCU- Fig. 13
 Bot Left Q₁₄ = 1 to Bot Control Unit (BCU)-Slip; Bot Right Q₁₆ = 1 to BCU- Fig. 15

Next Splash Select K) Pg. 8 TSPL-15 = 1; TSPL-31= 1; to Splash Select Fig. 18SSEL.

G) EVNN Vectors Top 4 T M B Pg 9
 Present 1 0 0 0
 Next 1 0 0 0

G) EVNN Vectors Bot B 4 T M Pg 9
 Present 0 1 0 0
 Next 0 1 0 0

J) Brown T M B Pg. 8
 Present 1 0 1
 Next 1 1 0

H) Tier Clocks T M B Pg. 8
 Present 1 1 1
 Next 1 1 1

References:

[biham] E. Biham & O. Dunkelman, Differential Cryptanalysis in Stream Ciphers, Technion CS Dept., Technical Report CS-2007-10-2007.

[zk-ais31] C. Gressel & Michael Rivkin, Read Me, Instructions for Using the ZK-Crypt Proprietary AIS-31 Deterministic and Random Noise Source Design Demonstrator, available under NDA, FortressGB Ltd., London and Omer, November 2007.

[zk-ccc] C. Gressel, ZK-Crypt Circuit Concept Drawings, www.fortressgb.com, London & Omer, November 2007.

[zk-code] A. Hecht, ZK-Crypt C Code Simulator, www.fortressgb.com, London & Omer, November, 2007.

[zk-secure] O. Dunkelman, A. Hecht, C. Gressel, The ZK-Crypt Security Analysis, www.fortressgb.com, London & Omer, 2007.

[zk-undst] C. Gressel, O. Dunkelman, & Avi Hecht, Understanding the ZK-Crypts – Ciphers for (Almost) all Reasons, www.fortressgb.com, London & Omer, November, 2007.