

# Assessing the Security of Key Length

Iain Devlin, Alan Purvis

Centre for Electronic Systems, Durham University,  
Durham, DH1 3LH, UK  
{iain.devlin, alan.purvis}@durham.ac.uk

**Abstract** This paper introduces Deluge, a second generation FPGA based key search system that specifically targets stream ciphers. The economically feasible implementation described is in dramatic contrast to other attack techniques and lends weight to the argument that brute-force attacks are underestimated in the security evaluation of cipher designs. Moreover with exhaustive search substantially cheaper than state guessing it is suggested that the practice of designing the state to be twice key length is excessive. Finally, the paper sets out possible alterations to the eStream cipher proposals, Trivium and Grain v1, that would counteract the practical threat that enhanced brute-force key search poses.

## 1 Introduction

Traditionally the cryptographical brute-force attack focused mainly on block ciphers and involved searching the entire key-space for a single decryption key which produced the correct plaintext match. With synchronous stream ciphers an enhanced attack is possible due to their data independence and through use of the birthday paradox computational requirements may be significantly reduced in exchange for multiple plaintext/ciphertext pairs. The development of a key search machine based on this principle [1] indicates that the brute-force resistance of stream ciphers based on 80-bit keys may be insufficient for a wide variety of applications where such data is obtainable and the quoted \$7.5 million figure for the present day computing chip cost of a successful one year attack system would seem feasible for a medium sized organisation. However, it is the rapid and continual advance of silicon technology that will in the near-term turn this curiosity into a very real concern. Furthermore the affordability of such machines for 80-bit key search is still probably underestimated by what is essentially a concept system. In an attempt to find the true affordability Section 2 outlines a second generation key search engine named ‘Deluge’, that builds on the knowledge gained from the first system to deliver both system level practicality and performance improvements.

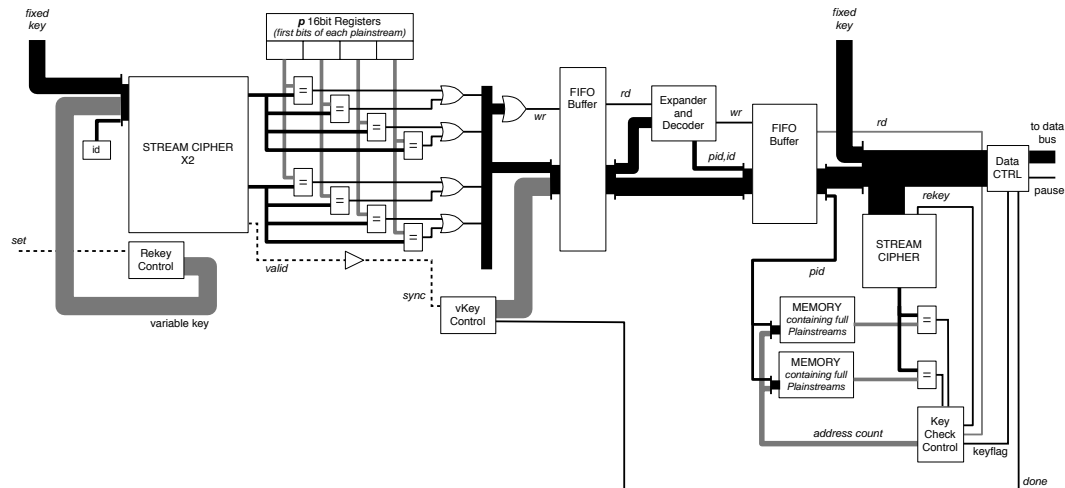
Despite suffering the associated issues of short 80-bit keys the symmetric ciphers, Trivium [2] and Grain v1 [3], resulting from the eStream project [4], are two very efficient and attractive solutions for high throughput hardware based systems [5]. It would appear rash to dismiss these stream ciphers purely on the single issue of exhaustive search security, especially when other non-invasive attacks, such as those of the algorithmic variety, have yet to show the efficiency necessary to be practical even for a well funded government agency. A more reasonable approach may be to increase the exhaustive key search resistance to a level equivalent or exceeding these other attacks and it is this concept that forms the basis of investigation in Section 3. Particular focus falls on the ciphers Trivium and Grain v1 and the document concludes in Section 4 with an outline of modifications which should proportionally address the associated issues and improve their underlying security.

## 2 Deluge

The shortcomings of the original key search system described at SHARCS 2006 [1] were apparent in a number of areas: the failure to target structural aspects of a specific hardware platform was felt to underestimate performance potential while its simulation led design process failed to address many system level design aspects crucial to high chip count set-ups. The creation of a

second generation system, ‘Deluge’<sup>1</sup> allows these factors to be addressed and the potential of exhaustive search on synchronous stream ciphers to be fully realised.

The underlying design of Deluge still owes much to this first generation system and, as such, uses the enhanced form of exhaustive search as the foundation of its search core (Fig. 1). Furthermore it possesses the same three tiered structure, enabling rapid key elimination at the search core’s resource constrained front-end but still allowing reliable comparison of remaining keys.



**Figure 1.** System diagram of Deluge’s internal key search core for an 2 keystream set-up

## 2.1 Keystream Generation

One of the main areas ripe for improvement was that of keystream generation with the previous machine using just a basic implementation of Grain v0 [6] to generate keystreams for comparison purposes. From a security stand point it was chosen not to continue targeting this cipher [7] but instead make the almost equally efficient Grain v1 [3] the focus of search attempts. To improve search speed the potential of pipelining the stream cipher initialisation process was investigated and as Table 1 illustrates Grain v1 sees a small improvement in efficiency while Trivium sees none at all due to its longer initialisation procedure. However, it is the consequential improvement in comparison unit utilisation from the continuous feed of keystreams that is the real underlying benefit so in both cases the piped versions would be preferred.

**Table 1.** Performance of Stream Cipher Keystream Generation Schemes on Altera EP1C20

Keystream Generator	Acceleration	Logic (LE)	Cycles/Key	Keys/LE.s
Grain v0	-	498	11	42k
Grain v1	-	543	11	35k
Grain v1 pipe.	pipelined	4665	1	46k
Trivium	-	633	19	26k
Trivium pipe.	pipelined	8507	1	26k
Grain v1 X2	x-factor	7647	0.5	53k
Grain v1 X4	x-factor	13554	0.25	58k

<sup>1</sup> The name was chosen to reflect the way the machine inundates its targets with keystreams and to maintain the connotations to water conjured by the phrase ‘stream cipher’.

## 2.2 X-Factor Pipelining

This technique specifically introduced for Deluge’s search core modifies the pipelined initialisation process to take advantage of the fact that two nearly identical keys lead to similarity in cipher state until the later stages of initialisation. For Grain v1 ( $t=16$ ) if two keys are selected so that only  $K_{79}$  differs, the calculation for the next state is identical and so the logic required for the second key may be saved. For the second stage a small calculation must be done but most of the state will be identical (see Table 2) leading to similar logic savings. It is not until cycle 6 of the initialisation process that routing costs outweigh the logic cost advantages by which stage a 15% benefit has been realised (Table 1). The technique may be extended to produce 4 keystreams by choosing key bits  $K_{79,78}$  as in Grain v1 X4 and should also be applicable to Trivium.<sup>2</sup>

**Table 2.** Register Overlaps Used in Dual X-Factor Pipeline: Grain v1 X2

Stage	NFSR	LFSR
1	$b_{78..0}$	$s_{79..0}$
2	$b_{79..64}, b_{62..0}$	$s_{79..0}$
3	$b_{79..76}, b_{74..72}, b_{70..68}, b_{66}, b_{63..48}, b_{46..0}$	$s_{79..72}, s_{70..65}, s_{63..0}$
4	$b_{70}, b_{63..60}, b_{58..56}, b_{54..52}, b_{50}, b_{47..32}, b_{30..0}$	$s_{78}, s_{76}, s_{74}, s_{70..69}, s_{67}, s_{73..56}, s_{54..49}, s_{47..0}$
5	$b_{47..44}, b_{42..40}, b_{38..36}, b_{34}, b_{31..16}, b_{14..0}$	$s_{47..40}, s_{38..33}, s_{31..0}$
6	-	-
7	-	-
8	-	-
9	-	-
10	-	-

## 2.3 System Level Design

Although the large bus widths of [1] may be suitable for a single chip system the need for a large scale multi-chip design when undertaking an 80-bit key space renders such an interface impractical. Deluge comprehensively resolves this issue with a redesigned interface structure based on a more conventional data and address bus design (Fig. 2) that would be suitable for implementation in a generic computational machine such as to Copacobana [8]. The simplified interface greatly eases the set-up process and the introduction of separated clock domains means that the search speed is no longer at the mercy of interface requirements.

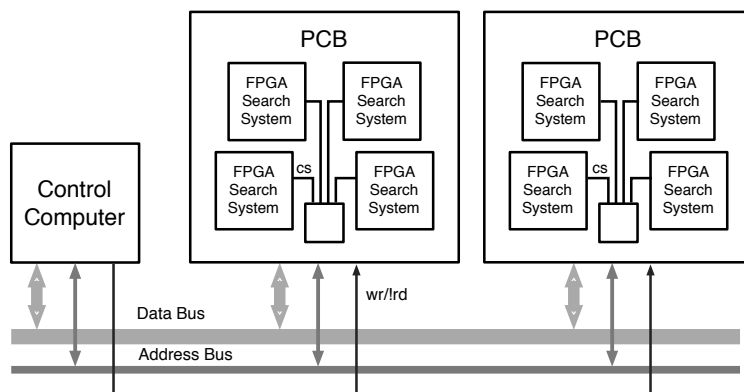
Deluge’s output register can be polled via the data bus to check if the allocated key space has been searched and if indeed a possible key has been found. In this way a single computer may monitor and manage a large number of Deluge search units with the only other requirement for Deluge’s correct multi-chip operation being that of the chip-select signal address decoders: all other interface signals are global to the system. The full outline of operating instructions implemented by the Deluge interface can be found in Appendix A.

## 2.4 Practical Demonstrator

To be confident the system would be practical in operation a demonstrator system was conceived. This placed a Deluge chip module with a Grain v1 search unit on a small Altera Stratix EP1S10 FPGA and the associated controller on a APEX 20K200EFC FPGA with a 7-segment display feeding information to the user. The system checked 2 keystreams against 64 plainstream<sup>3</sup> every

<sup>2</sup> Deluge itself is flexible as to the specifics of the search module so long as it produces at least one 16-bit keystream per clock cycle.

<sup>3</sup> The term “plainstream” is used throughout this paper and is defined as a keystream derived from knowledge of a plaintext-ciphertext pair.



**Figure 2.** Diagram of chip layout and inter-chip buses for Deluge.

cycle, covering  $2^{28}$  key space every 11 seconds while flagging key matches appropriately. The 10MHz bus speed and 50MHz search speed potentially could be taken much higher but the demonstrator was sufficient to show the feasibility of a large scale Deluge based search engine.

## 2.5 System Results

The Altera Cyclone II target platform is still one of the most economic FPGA offerings with the EP2C35 having a 250k unit volume pricing of \$22 per chip [9]. This chip can either fit a Deluge system with 2 keystreams and  $2^9$  plainstreams or 4 keystreams and  $2^8$  plainstreams<sup>4</sup> the simulation results of which are given in Table 3.

**Table 3.** Chip Cost of 80-bit Key Exhaustive Search using Deluge on a EP2C35F672C8

System	Logic (LE)	Searches/s	Recovery Time	FPGAs	Cost(05)
n=2X, p=512	25.4k	$141 \times 10^9$	1wk	7.1m	\$155m
n=2X, p=512	25.4k	$141 \times 10^9$	1yr	136k	\$3.0m
n=4X, p=256	29.8k	$136 \times 10^9$	1wk	7.3m	\$161m

The performance of both systems is 40 times that described in [1] on the FPGA platform and impressively half the price of the described structured-ASIC machine. Assuming technology trends continue to follow Moore's law [10] for another couple of years it is apparent that the 65nm FPGA process node will bring with it the possibility of a sub million dollar search system. As a consequence it is suggested that Deluge represents a very practical threat to stream ciphers with short 80-bit key lengths.

Finally, it is attractive to think of moving Deluge to an ASIC platform and bringing further cost reductions to short recovery time searches, however, unlike FPGAs the technology forces the creation of one-offs, making it arguably less attractive to an adversary considering investment in computing resource.

## 3 Non-Invasive Attack Resistance

### 3.1 Algorithmic Attacks

The form of attack most dangerous to any cipher is one based on a unanticipated weakness in the way an algorithm functions. These attacks have the potential not just to lead to minor

<sup>4</sup> Despite the two keystream set-up producing a slightly lower cost system the four keystream system may be preferred in reality due to its lower data requirement.

compromises in security but may instead devastatingly break the cipher with an attack that is practical on discovery. In the case of stream ciphers correlation between input and output, resynchronisation, period and results of algebraic analysis are among a number of factors now relevant to their design and it is likely that as understanding matures this list will grow. The realisation of new forms and combinations of algorithm based attacks may be inevitable but their applicability to a specific stream cipher will remain something of a mathematical lottery.

### 3.2 Assessing Complexity

An attack of complexity  $2^{79}$  would typically be considered as breaking an 80-bit key stream cipher with intended security  $2^{80}$ . However, as alluded to by Bernstein [11], complexity is often a rather vague term when used in this context as although ideally it should relate to exhaustive search it is often impractical to do so making a true comparison difficult: attacks may have a large memory element or simply their complexity may not scale linearly. Yet, an attempt must be made as defining the true complexity of the differing attack types targeted at a stream cipher is crucial for the formation of balanced security picture.

### 3.3 Evaluation of Attack Complexity

Conveniently Berbain's paper [7] provides precisely this opportunity in its attack on the original version of Grain [6] as direct comparisons can be extrapolated from the estimated costs presented in [1].

Berbain's first attack uses linear approximation to obtain a supposed complexity  $2^{55}$  attack against Grain. The impracticalities of cheaply computing the attack mean a reduced version of the cipher becomes the focus with the same techniques producing a quoted complexity of  $2^{35}$  and taking around an hour on an 2.5GHz Intel Xeon workstation. Expanding this to the level of a  $2^{79}$  complexity attack, which should reasonably be equivalent to 80-bit key exhaustive search, would require 2 billion machines a year to compute and assuming optimistically that Xeon chip cost (\$198 per chip [12]) represents the entire cost of a system this would result in a total cost of \$400 billion. In reality this figure should be equivalent to the \$240 million<sup>5</sup> a non-enhanced exhaustive search system would cost and a conclusion can be made that the original attack complexity is understated by at least order 11, making its real complexity somewhere above  $2^{66}$ . Berbain's improved second attack has a described complexity of  $2^{43}$  and so using same methodology the figure  $2^{53}$  can be derived for the real complexity of this attack. Both values are significantly higher than those quoted and the large memory requirements of each attack,  $2^{49}$  and  $2^{42}$ , should raise these figures higher still and effectively eliminate savings possible through translation into hardware.

The values for real complexity indicate that in the production of Grain v1, when the original Grain was corrected to provide at least  $2^{80}$  security, the security level against algorithmic attacks may in fact have been raised to beyond  $2^{90}$ . This is in direct contrast to security against enhanced brute-force attacks where Grain v1 like its predecessor remains at level no better than  $2^{75}$  and so a key length extension of 15-bits or more may be necessary to counteract the imbalance.

### 3.4 State Guessing

Another method for attacking stream ciphers, state guessing, involves searching for the internal state of the cipher instead of the key. As the internal state stores all of a stream cipher's entropy it is typically larger in size than the key, therefore rendering exhaustive state search less efficient than one of key-space unless a cipher's initialisation complexity is high. To overcome the extra burden state guessing uses large amounts of data and memory to reduce computation requirements in a trade-off based arrangement [13, 14] similar to that of enhanced brute-force. The balance of key length, initialisation procedure and state size determine which of these two universal attacks will prove more efficient against a particular cipher.

<sup>5</sup> Calculated from the keysearch system of [1] running with one plainstream.

### 3.5 Consequences of Key Length Extension on State Guessing

Looking at a 15-bit key length extension from the context of state guessing attacks an 80-bit key, generic stream cipher can be imagined that has an initialisation period of  $2^5$  clock cycles and a state size of 160-bits. Thus in extending the key to 95-bits a state guessing attack with equivalent computational cost to brute-force would be capable of producing around  $2^{99}$  keystream sections<sup>6</sup> suitable for comparison purposes. These would, on average, need to be compared to  $2^{60}$  bits of collected plainstream data to achieve a successful outcome, leaving a data and memory requirement still well in the petabyte region.

Moreover, while the headline figure for memory seems plausible for a hard-disk set-up, this would simply ignore the very real time constraints on each search and compare operation. The output from each guess must be compared to all  $2^{60}$  different values and if an impractical  $2^{159}$  comparison operations are not to be required the output must be directly used to address memory locations with these locations returning a 'logic 1' if their position exists in the plainstream data block. Unfortunately this leads to an impracticably high-false positive rate unless the memory address space is comparable to the state size, something which would in itself lead to memory requirements becoming impractical.

By making the locations return the next part of the plainstream data block and a pointer to data location the false-positive rate could be eliminated by a subsequent round of comparison operations, with a small memory cost per location of around  $2^7$  bits. For half the current \$240 billion cost<sup>7</sup> of a 95-bit key length targeting enhanced brute-force system possibly a zettabyte of storage could be bought assuming a figure of \$100 per half terabyte of hard-disk space. This amount of memory would provide  $2^{65}$  storage locations leaving a still comparatively feasible number of comparisons at  $2^{94}$ . Nonetheless, memory access on this scale is relatively slow process with hard-disk access times typically in the millisecond range and therefore with  $2^{99}$  memory accesses required and only 31 billion milliseconds in a year, sextillions of separate disks would be needed to mount a successful attack within such a time-scale. Storage cost means access requirements cannot be sufficiently reduced to alleviate the issues of access time and thus state guessing may be considered impractical despite the increase in the ratio of key length to state size to above the conventional 1 to 2 design limit originally proposed by Babbage [13].

Translating the state guessing attack to a Biryukov style attack [15] though use of the Hellman time-memory trade-off [16] does not alter this conclusion as although it potentially reduces data collection it also introduces a large precomputational requirement comparable to that of exhaustively searching the entire 160-bit state space and so any such attack would again be impractical.

### 3.6 Key Space Based Time-Memory Trade-off

If as described by Hong in [17] the Hellman time-memory trade-off is applied to search the stream cipher key space rather than the enlarged state a much more effective attack can be realised despite the need for cipher initialisation calculations. Both online computational costs and memory requirements would see feasibility due the reduced search space, while the precomputation cost, that provides a complexity threshold to the attack's practicability, would also be reduced.

In a theoretical sense this precomputation should be of identical cost to that of an enhanced brute-force key space attack with a key length extension equally complicating both attack methods. However the similarity in duration of the 1 week precomputation described in [18] for a time-memory trade-off attack on 40-bit DES and that taken for Copacobana [8] to search a full 56-bits of DES key space using exhaustive search possibly suggests otherwise. The reality is that the complexity of precomputation will generally outweigh that of enhanced brute-force due to a corresponding need for point storage and the computational cost incurred by search

<sup>6</sup> The specifics of system design will affect this figure: need for a high rate of guess elimination will tend to favour brute-force efficiency; use of windowing on the state guessing system's keystream output will tend to increase section generation.

<sup>7</sup> Again calculated from the keysearch system of [1].

reduction techniques such as distinguishing points. Consequently when data availability is low enhanced brute-force will represent the lowest entry level attack.

### 3.7 Alternatives to Key Length Extension

Key extension is not the only way to increase brute-force resistance and one alternative is to simply increase the required length of the set-up protocol. Each doubling in length will equate to increasing the key length by one bit as extra effort must be expended in terms of hardware resources or time.

Although effective, lengthening the set-up protocol is not an attractive path to follow as initialisation time directly affects a cipher’s performance: in the case of Grain v1 if set-up was extended to provide an equivalent of an extra 10-bits of security the set-up time would rise from 10-cycles to over 10000-cycles, a level that is likely to be impractical in many applications. Other methods that increase the cipher’s implementation complexity to raise attack resistance share similar performance caveats.

## 4 Outline of Modifications

An inevitable consequence of enhanced brute-force attacks is that stream ciphers must fundamentally lose the direct relationship that presently exists between key length and security. This is of little consequence to software stream ciphers with their long keys as even an enhanced brute-force attack will simply be impractical. However for hardware focused ciphers with short 80-bit keys, such as Trivium and Grain v1, it is of great consequence with enhanced brute-force representing a practical threat. Unless these algorithms are to follow a path of massively increased initialisation set-ups and a redesign to increase implementation complexity, something which contradicts the performance aims that underpin this class of cipher, key length must be raised. The modifications described in the final sections are one method of achieving this.

### 4.1 Grain

Grain v1’s state size of 160-bits is right on the conventional limit to avoid state guessing attacks so care has to be taken in the level of key length extension used. In light of the analysis in Section 3.5 it is proposed that key length should be extended by 22-bits to a total of 102-bits. This leaves the state-guessing attack possibly preferential to enhanced brute-force but still in the realm of impracticality medium-term. As such algorithmic attacks will be re-established as the main theoretical concern.

Due to the comparatively small state size implementing the proposed extra key bits must necessarily have an impact on the available IV range of Grain v1 as only 15-bits of the initial state serve no function and hence a compromise must be made. With often predictable values and open transmission the role of IVs in entropy provision as described by Hong [17] is negligible so their implementation should be regarded primarily as a convenience to the user. Use of a 42-bit IV provides over 4 trillion possible IVs and should be sufficient for almost all applications, so the proposed modification involves placing the extra key material in 22-bits locations where IV material would have previously resided:

$$\begin{aligned} (b_{79}, \dots, b_0) &\leftarrow (K_{79}, \dots, K_0) \\ (s_{79}, \dots, s_0) &\leftarrow (1, \dots, 1, K_{101}, \dots, K_{80}, IV_{41}, \dots, IV_0) \end{aligned}$$

By swapping IV state bits directly for key material the expectation is that the algorithmic security of Grain v1 will be left unaltered by the change but it may be more prudent to simply migrate designs to its 128-bit keyed cousin, Grain-128 [19], which with its similar structure maintains the family’s size and speed advantages. Crucially the introduction of the 128-bit key dispels the possibility of practical generic attacks in all likely applications and leaves system designers to only concern themselves with its maintenance of algorithmic strength.

## 4.2 Trivium

Trivium's state size of 288-bits is much larger than that of Grain v1 and so state guessing attacks are not an issue of concern when considering key length extensions. In light of this it is proposed that the key should be lengthened by 32-bits to a total of 112-bits. These extra bits need not impact on the possible IV size and it is proposed that they are loaded into the lower 32-bits of the register containing state bits  $s_{178}, \dots, s_{288}$ :

$$\begin{aligned}(s_1, \dots, s_{93}) &\leftarrow (K_1, \dots, K_{80}, 0, \dots, 0) \\(s_{94}, \dots, s_{177}) &\leftarrow (IV_1, \dots, IV_{80}, 0, \dots, 0) \\(s_{178}, \dots, s_{288}) &\leftarrow (K_{81}, \dots, K_{112}, 0, \dots, 0, 1, 1, 1)\end{aligned}$$

The 32-bit key extension not only makes current brute-force attacks impractical but should in reality remove enhanced brute-force from the security equation and leave algorithmic attacks as the non-invasive method most likely to first compromise the cipher. Indeed, even these attacks would be made more complex by the additional key material, so although the medium term security of Trivium would clearly not be guaranteed, the algorithm could avoid the spectre of a practical attack.

## 4.3 Further Modifications

Although key length modification is proposed in the preceding paragraphs, in the implementation of stream cipher hardware modules it may be prudent to include a facility to make the set-up length selectable as this would provide a facility to partially mitigate against future cryptanalysis advances: extending the set-up time by a multiple of 32 would potentially make attacks 32-times as costly. By invoking this flexibility the performance hit need only be taken as and when circumstances dictate.

## References

- [1] I. Devlin, A. Purvis: A fundamental evaluation of 80 bit keys employed by hardware oriented stream ciphers. SHARCS 2006. <http://www.sharcs.org/>
- [2] C. De Cannière, B. Preneel: TRIVIUM - Specifications. eSTREAM, ECRYPT Stream Cipher Project, Report 2005/030 (2005). <http://www.ecrypt.eu.org/stream>
- [3] M. Hell, T. Johansson, W. Meier: Grain - A Stream Cipher for Constrained Environments. Special Issue on Security of Computer Network and Mobile System. International Journal of Wireless and Mobile Computing, 2006.
- [4] Preliminary Call for Stream Cipher Primitives. ECRYPT NoE, v1.0, 30th Nov 2004. <http://www.ecrypt.eu.org/stream/call/>
- [5] T. Good, W. Chelton, M. Benaïssa: Review of stream cipher candidates from a low resource hardware perspective. SASC 2006, Stream Ciphers Revisited. <http://www.ecrypt.eu.org/stv1/sasc2006/>
- [6] M. Hell, T. Johansson, W. Meier: Grain - A Stream Cipher for Constrained Environments. <http://www.ecrypt.eu.org/stream/>
- [7] C. Berbain, H. Gilbert, A. Maximov: Cryptanalysis of Grain. SASC 2006, Stream Ciphers Revisited.
- [8] S. Kumar, C. Paar, J. Pelzl, G. Pfeiffer, A. Rupp, M. Schimmler,: How to Break DES for €8,980. SHARCS 2006. <http://www.copacobana.org/>

- [9] Altera: Altera's New Cyclone II FPGAs Offer 30 Percent Lower Costs Than Previous Generation. Altera Press Release, 28 June 2004.  
[http://www.altera.com/corporate/news\\_room/releases/releases\\_archive/nr-releases\\_archive.html](http://www.altera.com/corporate/news_room/releases/releases_archive/nr-releases_archive.html)
- [10] G.E. Moore: Cramming more components onto integrated circuits. *Electronics*. Vol. 30, No. 8, April 19, 1965. <http://www.intel.com/technology/mooreslaw/>
- [11] D. J. Bernstein: Understanding Brute Force. eSTREAM, ECRYPT Stream Cipher Project, Report 2005/036, 2005. <http://www.ecrypt.eu.org/stream>
- [12] Intel: Intel Continues Push Down Power-Optimization Path With Intel Xeon Processor Line. Intel News Release, Sept. 26, 2005.  
<http://www.intel.com/pressroom/archive/releases/20050926corp.a.htm>
- [13] S. Babbage: A Space/Time Tradeoff in Exhaustive Search Attacks on Stream Ciphers. *European Convention on Security and Detection, IEE*, Vol. 408, May 1995.
- [14] J. Golic: Cryptanalysis of alleged A5 stream cipher. *Eurocrypt97, LNCS 1233*, Springer-Verlag, 1997.
- [15] A. Biryukov, A. Shamir: Cryptanalytic Time/Memory/Data Tradeoffs for Stream Ciphers. *Asiacrypt 2000, LNCS 1976*, Springer-Verlag, 2000.
- [16] M. E. Hellman: A Cryptanalytic Time-Memory Trade-Off. *IEEE Trans. on Information Theory*, Vol. 26, No. 4, July 1980.
- [17] J. Hong, P. Sarkar, Rediscovery of Time Memory Tradeoffs. *Cryptography ePrint Archive*, 2005, 090. <http://eprint.iacr.org/>
- [18] , J. Quisquater, F. Standaert, G. Rouvroy, J. David, J. Legat: A Cryptanalytic Time-Memory Tradeoff: First FPGA Implementation. *FPL 2002*.
- [19] M. Hell, T. Johansson, A. Maximov, W. Meier: A Stream Cipher Proposal: Grain-128.  
<http://www.ecrypt.eu.org/stream/>

## A System Operation Information for Deluge

```

// -----
// xclk | interface clock - optimize for compatability |
// clk  | core clock      - optimize for max search speed |
// -----
// xaclr | reset system interface to default state |
//       | (clears plain, key & IV registers) |
// aclr  | reset search core to default state |
//       | (use after loading new register key) |
// -----
// gpause | pause core (sleep) |
// -----

// -----
// cs, wr | addr[msb..lsb] | data | NOTES... |
// -----
// 0 X | X | ZZZZ | |
// -----
// 1 0 | X I 0 | {0001} =>done | read status reg |
//       | I | {0002} =>flag | (outputing data) |
//       | X I C I 1 | {search key} | read o/p key reg |
//       | I I | | (C selects part) |
// -----
// 1 1 | 00 I X | XXX1 | write ack |
//       | I | (rekeys core) |
//       | I | XXX2 | (clears flag) |
//       | 01 I X I A | rKey | write reg key |
//       | I I | (A -> part) |
//       | 10 I X I A | IV | write IV |
//       | I I | (A -> part) |
//       | 11 I J I C | plain | write plainstream |
//       | I I | (J -> position) |
//       | I I | (C -> part ) |
// -----

// -----
// ** raised flag indicates a successful key has been found
// ** note: a two cylce latency is present on outputs
// -----

```