Statements required in B section

B2. There are no hidden weaknesses inserted by the designers in any algorithm of the ten proposed in NUSH proposal.

B3. Security evaluation of the algorithms of NUSH proposal showed no weak keys but zero key together with the zero constants and also showed that only generic cryptographic attacks are effective with the “natural” security level.
To be more precise.
The best linear analogue for a single round of the algorithm NUSH (the same analogue for all separate rounds) has probability of the form

(1 + (1/2)**4)/2

or

(1 - (1/2)**4)/2.

So, the number of the rounds of the algorithm NUSH (that equals 17) is taken such that the number of ciphertext blocks that are necessary for the linear cryptoanalysis method should be more than the number of all possible ciphertext blocks.

This number of rounds of the algorithm NUSH guarantees also that differential cryptoanalysis could not be more effective than brute force attack.

Weak keys methods, semi weak keys methods, related keys methods, global and local deduction methods, meet-in-the-middle method were found less effective for the algorithm NUSH than the brute force method with memory (see the cover page).

All these and other known crypto analytical methods were evaluated for all the standard crypto attacks: a ciphertext-only attack, known-plaintext attack, chosen-plaintext attack, adaptive chosen-plaintext attack, adaptive chosen-ciphertext attack.

Thus, the expected security level is 10**61 of elementary operations with memory for at most 10**16 blocks of cipher text corresponding to different keys.

B4. All the primitives of NUSH proposal are secure (see the cover for the security estimation), flexible (see the source codes and tests implementation) and easy-to-use for the most of contemporary applications.
This has been proven by years of their practical use in LAN Crypto products for more than 1000 customers in Russia and other CIS states.

B5. A design rationale.

We do not use such complicated arithmetic operations as integer or modular multiplication and division on purpose of the potential NUSH algorithm implementation for the less-than-Pentium power chips and other small electronic devices.

The registers rotations are used with constant and predetermined lengths of the rotations only. This gives us the following advantages with respect to algorithms with the text-dependent or even key-dependent rotations of registers:

- fixed lenth rotations are faster in implementation,
- they are easy-to-implement in hardware,
- they provide us with more security in a hardware implementation with respect to linear and differential cryptanalysis,
- we could choose the rotations lengths in the most effective way with respect to it's security.

Each round of the algorithm NUSH is extreamely simple: one linear operation, one non linear operation and one rotation of a fixed length for each of the registers used.
The linear operation and the rotation are necessary to protect a key of the algorithm NUSH with respect to differential cryptanalysis.
The non linear operation and the rotation are necessary to protect a key of the algorithm NUSH with respect to linear cryptanalysis.

The exact sequence of the basic operations of the algorithm NUSH is chosen (after many days of computational experiments) in such a way that it guarantees the most homogenuous and balanced work of the both pipelines of Pentium-type processors (with two pipelines).

The exact sequence of the algorithm NUSH parameters (arguments of the round functions) guarantees the maximal possible speed of the grouth of polynomial degree of each output bit as a boolean function of input bits.

Number of rounds (equals 17) is taken such that it guarantees security with respect to the linear and differential cryptanalysis methods.

The exact sequence of the registers rotations is taken in such a way that it provides us with the fastest possible confusion and diffusion.

The exact sequence and set of the basic operations in the NUSH algorithm is taken such that its portable C software implementation would be the most effective (with respect to speed) for the most popular computer platfors. That is the differnces between it's C implementation and it's different Asm implementations are as minimal as possible.

The simplicity of the basic operations of the NUSH algorithm gives to an independent expert an additional opportunity to verify the designers statement of the absence of any back door in the algorithm.

More than 25 years of professional cryptographic experiences of the NUSH authors and of many their friends and partners among Russian-speaking professional cryptographers gives us additional trust that the estimated security level is not far from the real one.

B6. The algorithms of the NUSH submission are computationally effective and very flexible.

Block size and the set of basic operations of the NUSH block algorithm are chosen such, that it can be easily implemented as by any Pentium-like processor with two pipelines as by a small device like smart card processor.

Key setup computational efficiency may be estimated in the following way.
For any Pentium-like processor with two pipelines and balanced distribution of operations between both pipelines we have.

Key setup	C implementation (cycles per key)	Asm implementation (cycles per key)
Block of 64 bits	64	64
Block of 128 bits	112	112

There is no particular primitive setup for the NUSH algorithm. So, this stage of the implementation is extreamely effective.

Encryption/decryption computational efficiency (ECB mode) can be estimated as follows.

Encryption/Decryption	C implementation	Asm implementation
Block of 64 bits	Cycles per block	Cycles per block
	180	130
	Cycles per Byte	Cycles per Byte
	23	17
Block of 128 bits	Cycles per block	Cycles per block
	340	250
	Cycles per Byte	Cycles per Byte
	22	16

Memory requirements for the NUSH block algorithm are the following.

Key setup	Block of 64 bits	Block of 128 bits
Memory to store a session key (Bytes)	80	276

Memory requirements in the case of speed-optimal implementation of the NUSH algorithm are the following (in Kbytes for the program code)

Encryption/decryption	C implementation	Asm implementation
Block of 64 bits	≈1.08K	≈0.8K
Block of 128 bits	≈2.04K	≈1.5K

For implementation of the NUSH algorithm optimal with respect to memory requirements, code of the progrem takes less than 100 Bytes of memory.

The algorithm NUSH is so simple and clear in description, that anybody can easily verify these statements by himself implementing it.

B7. Basic techniques for implementers.

It is not recommended to use zero key with zero constants of the algorithm NUSH together.

If the lengths of the registers rotations will be taken in a different way (that is in principle possible) security level of the algorithm may fall down.

If the number of the rounds will be =< 8, then the algorithm NUSH security level is less that it is stated in this submission.

Simple key scheduling of the algorithm NUSH makes it more sensitive to side channel eavesdropping. An implementer should pay attention to protection of a program or a device with NUSH from phisical intrusion.

We recommend implementations with key storing in the first level cach memory. It could be done easily because of the very simple structure of the algorithm.

For the stream ciphers from the NUSH submission second use of the same key is possible only in combination with the one-time publicly transmitted marcants.

For the self-synchronizing stream ciphers of the NUSH submission the second use of a pair (key – synchro message) is prohibited.

For the MAC and HASH of the NUSH submission it is prohibited to extend MAC or HASH values by using intermidiate values of the process of MAC or HASH computation.

The algorithm NUSH may be used with almost any other (than described in the text) reasonable key scheduling scheme without security level falling down.

_____________________/ Anatoly Lebedev, President, LAN Crypto, Int.