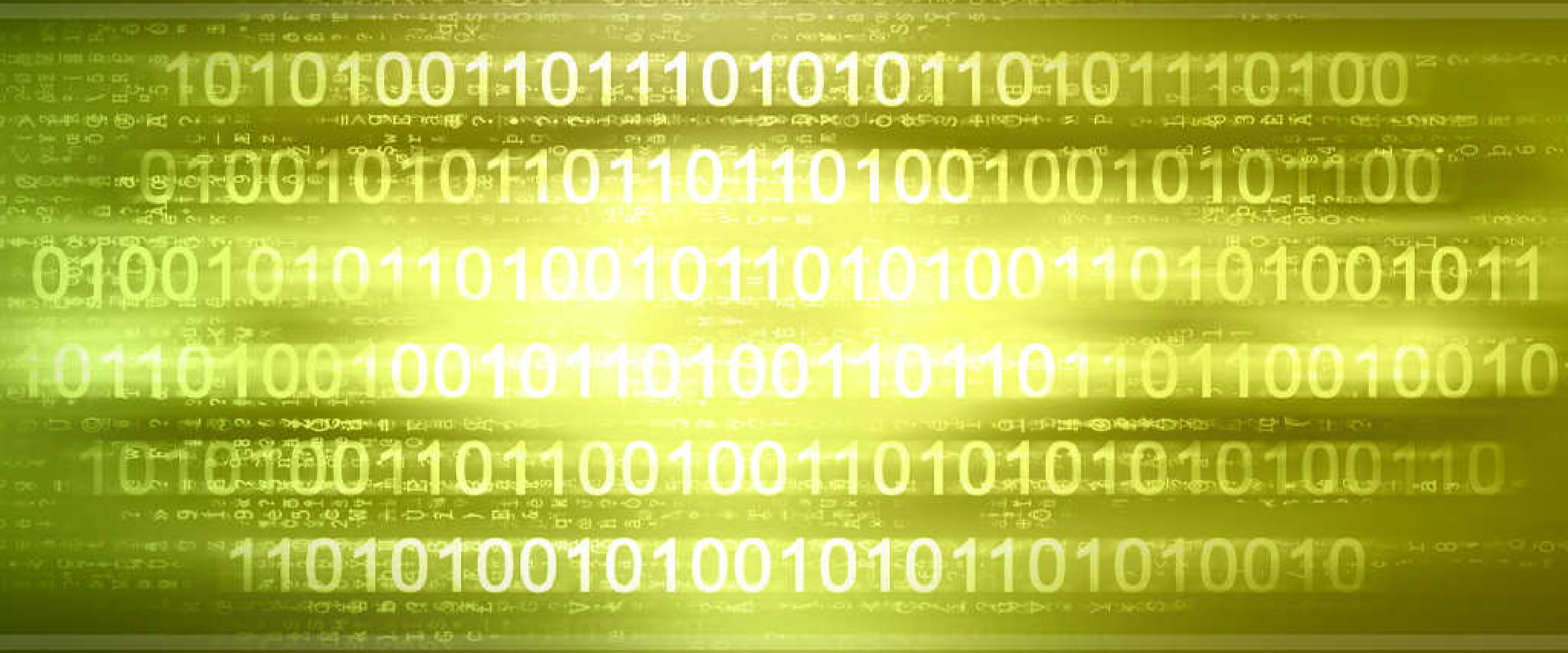
Ing. Michal Braško
Institute of Computer Science and Mathematics, FEI STU BA
ISCAMI 2012



On a performance optimization of the cipher NLSv2

Agenda

1. Introduction
2. NLSv2 construction
3. Testing environment
4. Implementations and their benchmark
5. Conclusion

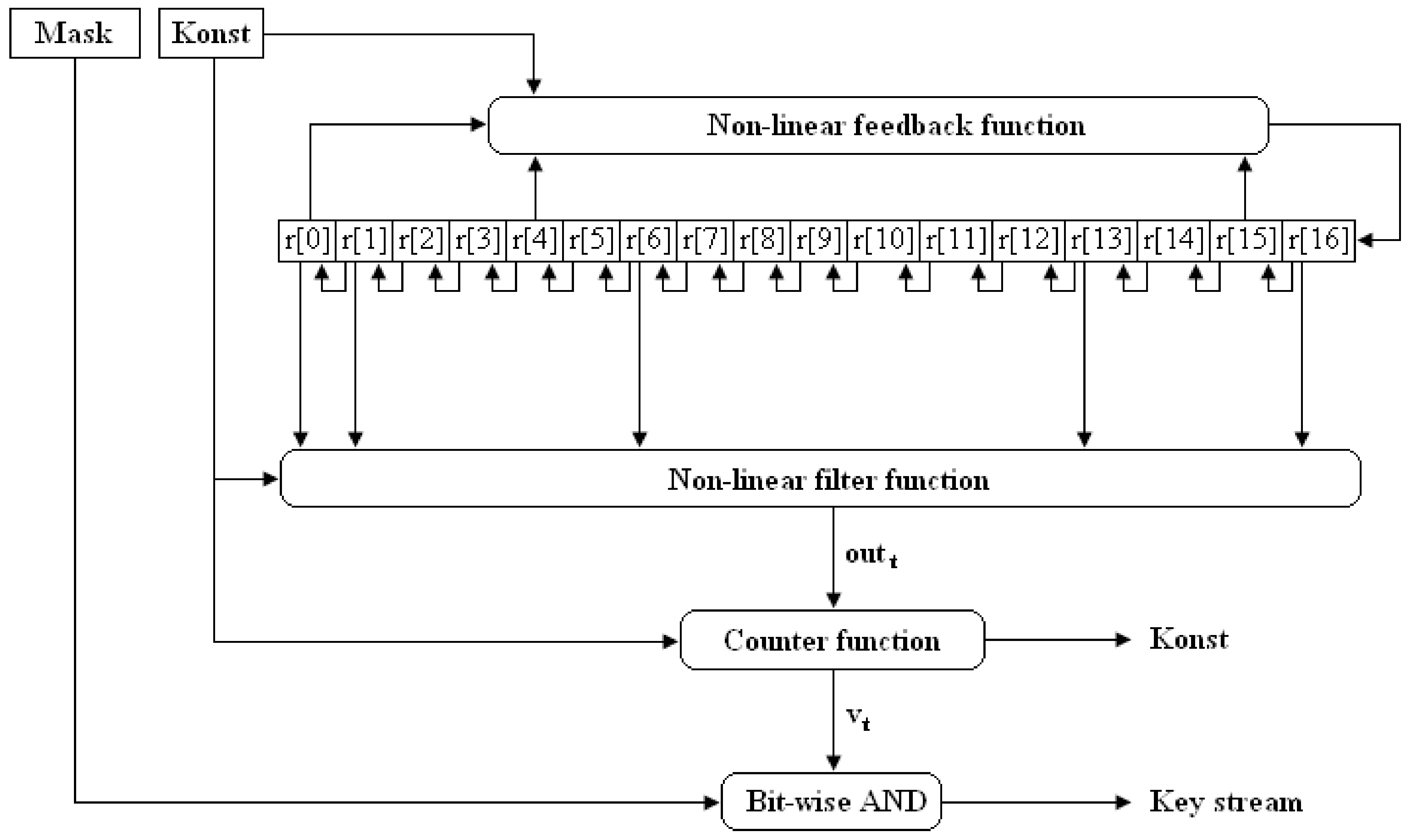
Introduction to the NLSv2

- stream cipher (pseudorandom generator)
- Australian proposal (2007)
- eSTREAM phase 3
- profile 1A
- variable key and IV length
- 128-bit security
- good performance

NLSv2 construction

- 32-bit words
- simple 32-bit operations (XOR, rotation, modular addition, bit-wise AND, SBox)
- internal state 18 words (576 bits)
- autentication: Mundja MAC
- 3 components - NFSR, NLF and counter

Internal structure



Functions

- NFSR:

  $r_{t+1}[16] = \text{SBox}((r_t[0] \lll 19) + (r_t[15] \lll 9) + \text{Konst}) \wedge r_t[4]$

- SBox: 32 bit to 32 bit
- NLF:

  $\text{out}_t = \text{NLF}(\sigma_t) = (r_t[0] + r_t[16]) \wedge (r_t[1] + r_t[13]) \wedge (r_t[6] + \text{Konst})$

- Counter - every f16 round:

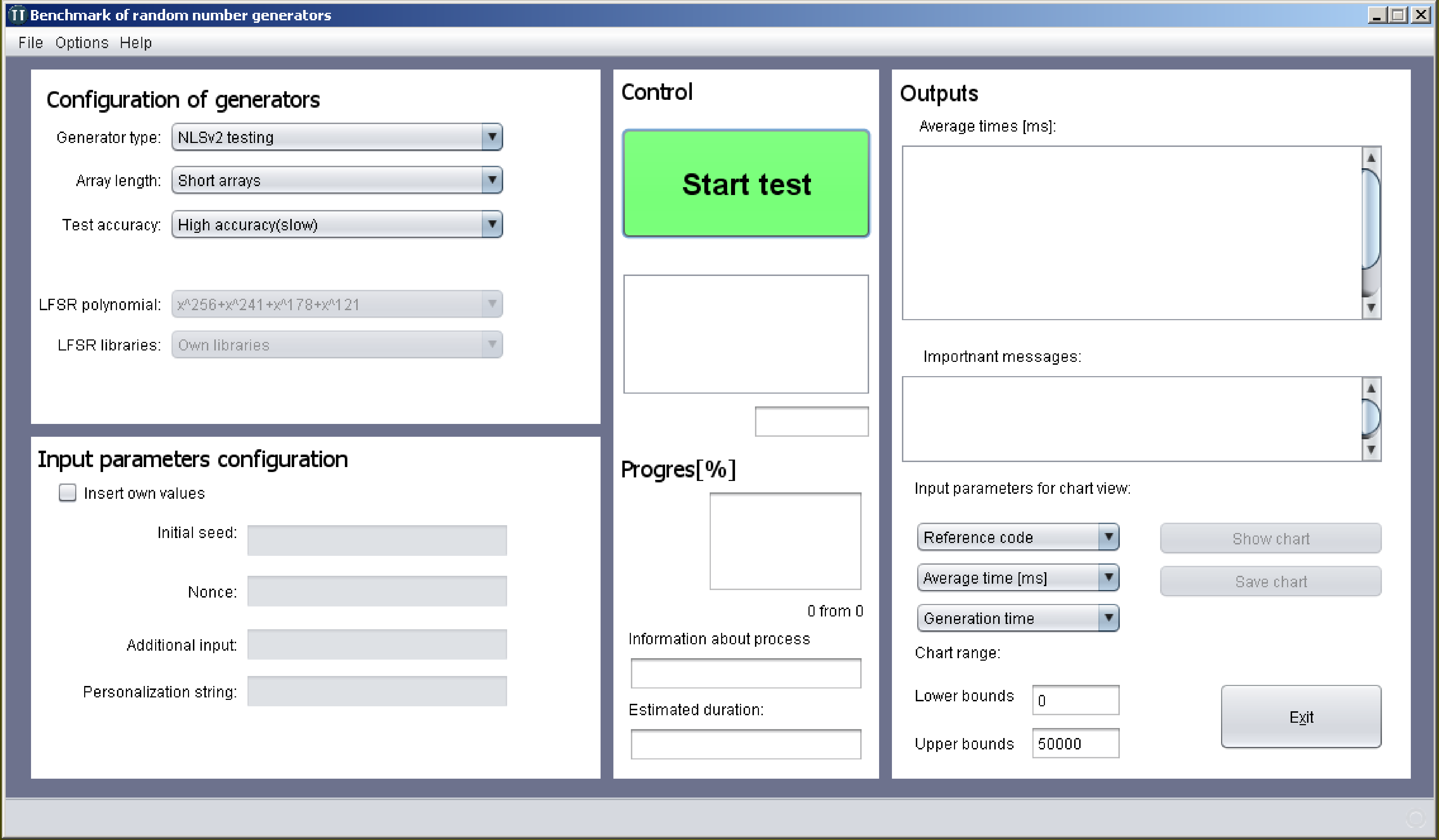
  $r_{t+1}[2] = r_{t+1}[2] + t$

  $\text{Konst} = \text{out}_{t+1}$

Testing environment

- MS Windows XP SP3 32 bit
- Intel Core i5 M520 @ 2.4 GHz
- 3 GB RAM
- MS Visual Studio 2010 C/C++ compiler
- Generator Benchmark (Martin Hutník 2012)
- 30 000 tests for each category
- 100 generated sequences for each test < 1 MB

Testing application



Configuration of generators

Generator type: NLSv2 testing

Array length: Short arrays

Test accuracy: High accuracy(slow)

LFSR polynomial: x^256+x^241+x^178+x^121

LFSR libraries: Own libraries

Input parameters configuration

☐ Insert own values

Initial seed:

Nonce:

Additional input:

Personalization string:

Control

Progres[%]

0 from 0

Information about process

Estimated duration:

Outputs

Average times [ms]:

Importnant messages:

Input parameters for chart view:

Reference code

Average time [ms]

Generation time

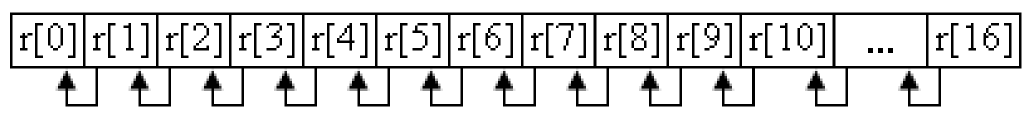
Chart range:

Lower bounds 0

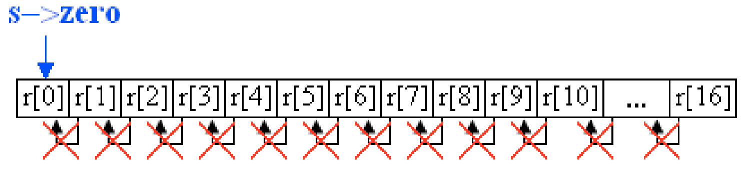
Upper bounds 50000

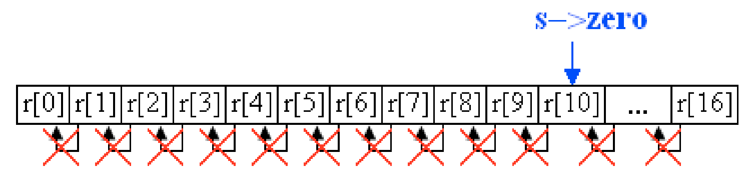
Algorithm implementation description

- cycling the register:
  1. real



  2. virtual





  3. combination - real (short) and virtual (long)
- internal state: register as unsigned int array
- preprocessor directives

Implementation no.1

- virtual cycling for each keystream length in a same way
- index access - always modulo:

  `s->r[((s->zero)+i) % N]`

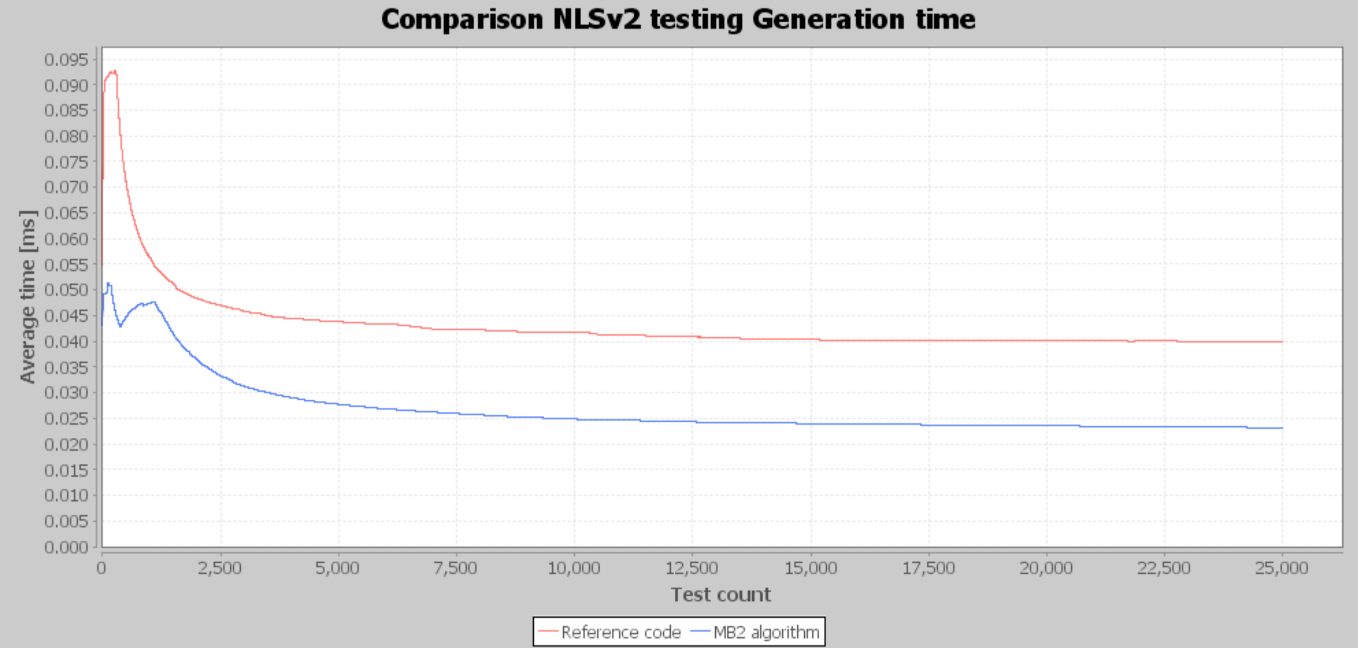
- benchmark results:
  - -98,9% .. 4 B
  - -131,9% .. 32 B
  - -129,9% .. 67 B
  - -548,6% .. 1 KB
  - -530,2% .. 1 MB

Implementation no.2

- length>=N*4: generated by complete phases with 17 defined rounds
- length<N*4: 17 incomplete phases with 1-16 rounds, need to shift register to state “zero == 0”
- benchmark results:
  - -232.7% .. 4 B
  - 1.2% .. 32 B
  - 43.0% .. 67 B
  - -14.7% .. 1 KB
  - -10.9% .. 1 MB

Benchmark results, 67 bytes

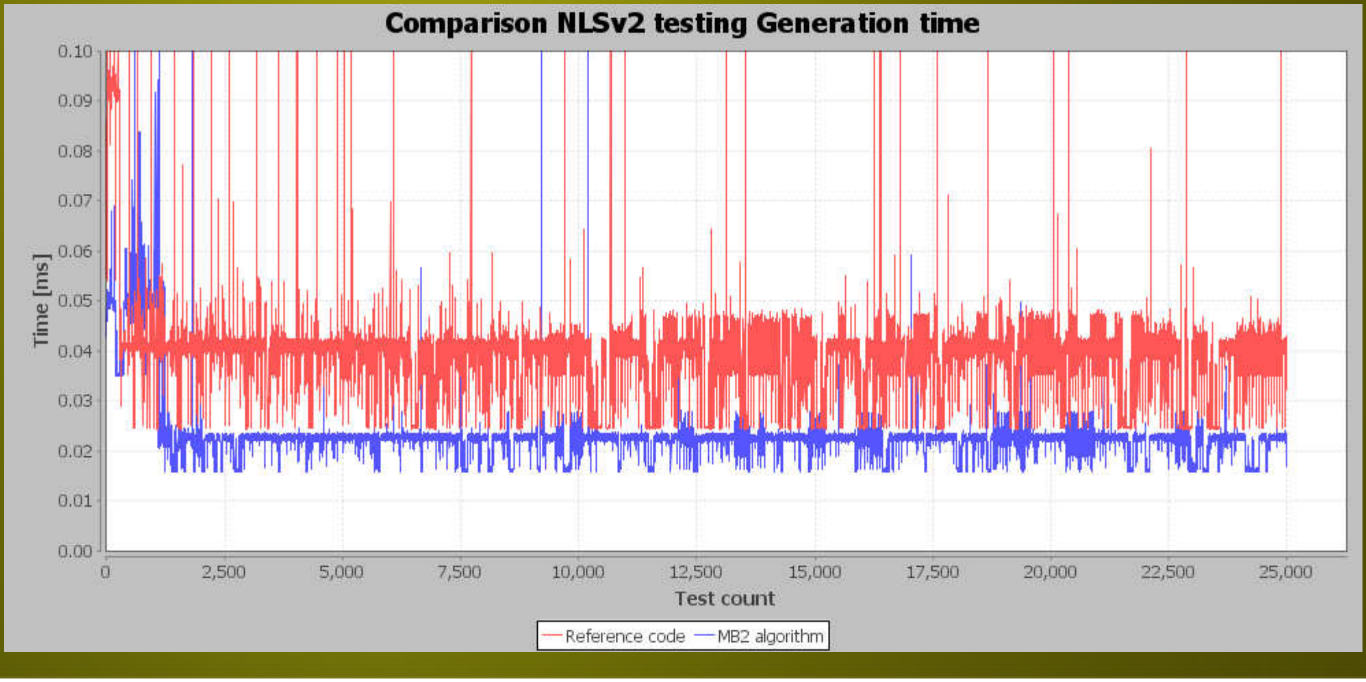
a) average time



red curve – reference implementation

blue curve – implementation no.2

b) instant time



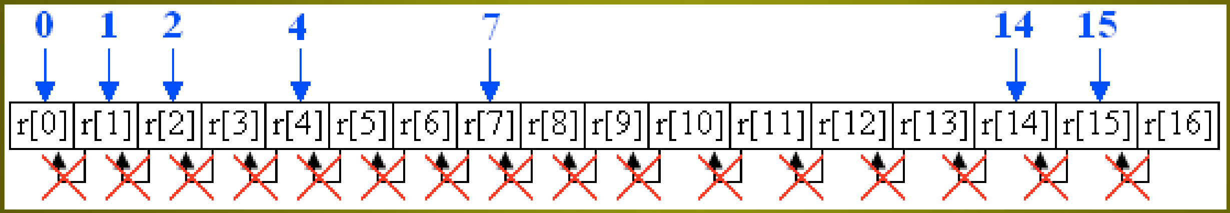
Implementation no.3

- based on the implementation no.2
- enhanced instantiate function
- switch replaced by if-then-else construction
- benchmark results:
  - -272,3% .. 4 B
  - -6.6% .. 32 B
  - 25,0% .. 67 B
  - -12,1% .. 1 KB
  - -3,6% .. 1 MB

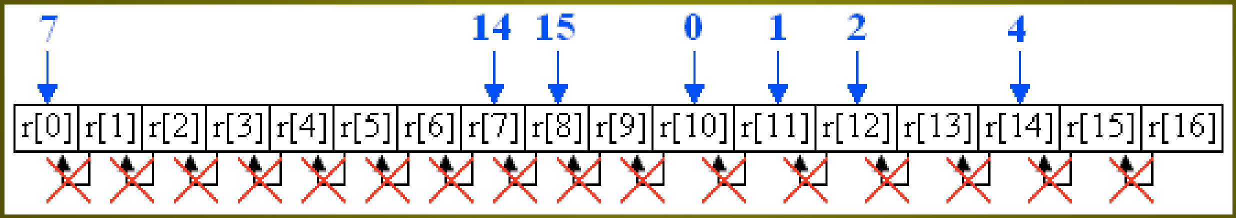
Implementation no.4

- virtual cycling for each keystream length
- shorter lengths with buffering of 17 words
- no modulo when accessing register:

t = 1: ROUND(0,1,2,4,7,14,15)



t = 10: ROUND(**10,11,12,14,0,7,8**)



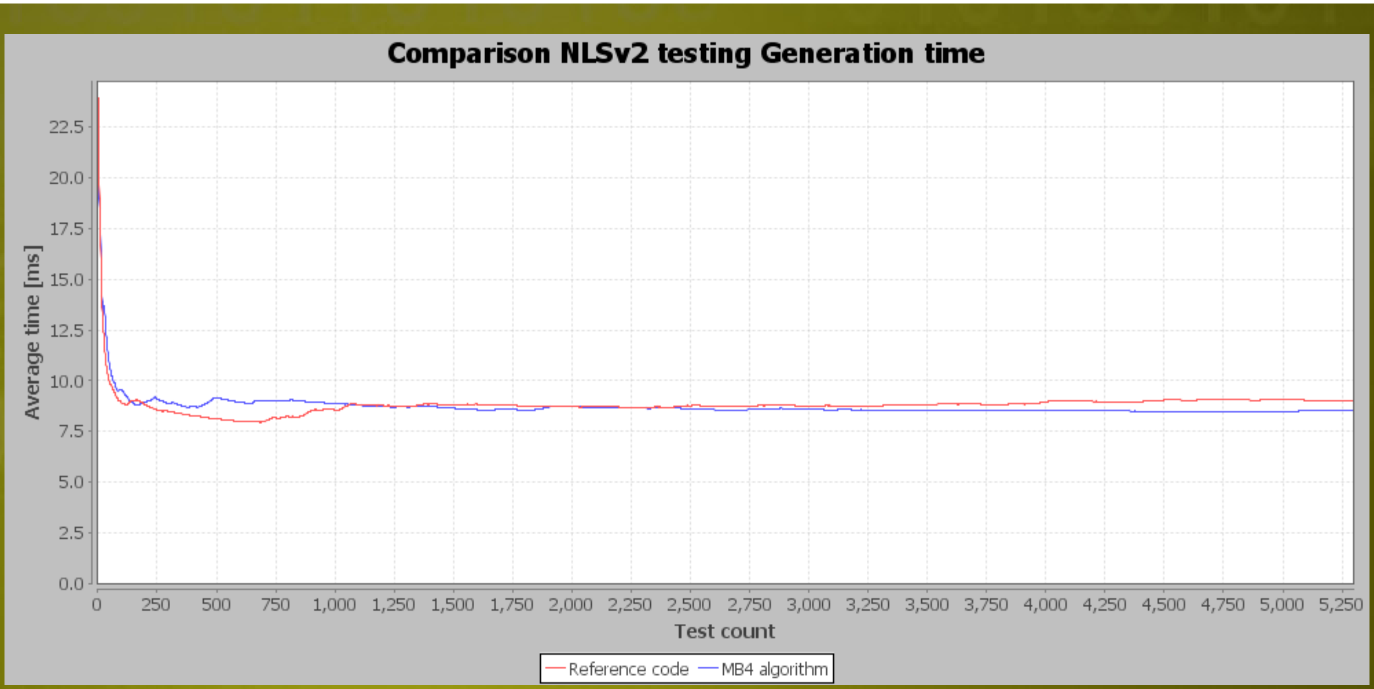
Benchmark results

Keystream length	Enhancement
4 B	16,1%
8 B	13,8%
12 B	14,8%
24 B	20,0%
32 B	19,0%
67 B	27,9%
100 B	-4,4%
500 B	-5,1%

Keystream length	Enhancement
1 KB	-6,1%
2 KB	-1,9%
4 KB	0,2%
10 KB	-1,9%
100 KB	-0,5%
1 MB	0,0%
2 MB	6,3%
4 MB	5,2%

Benchmark results, 4 megabytes

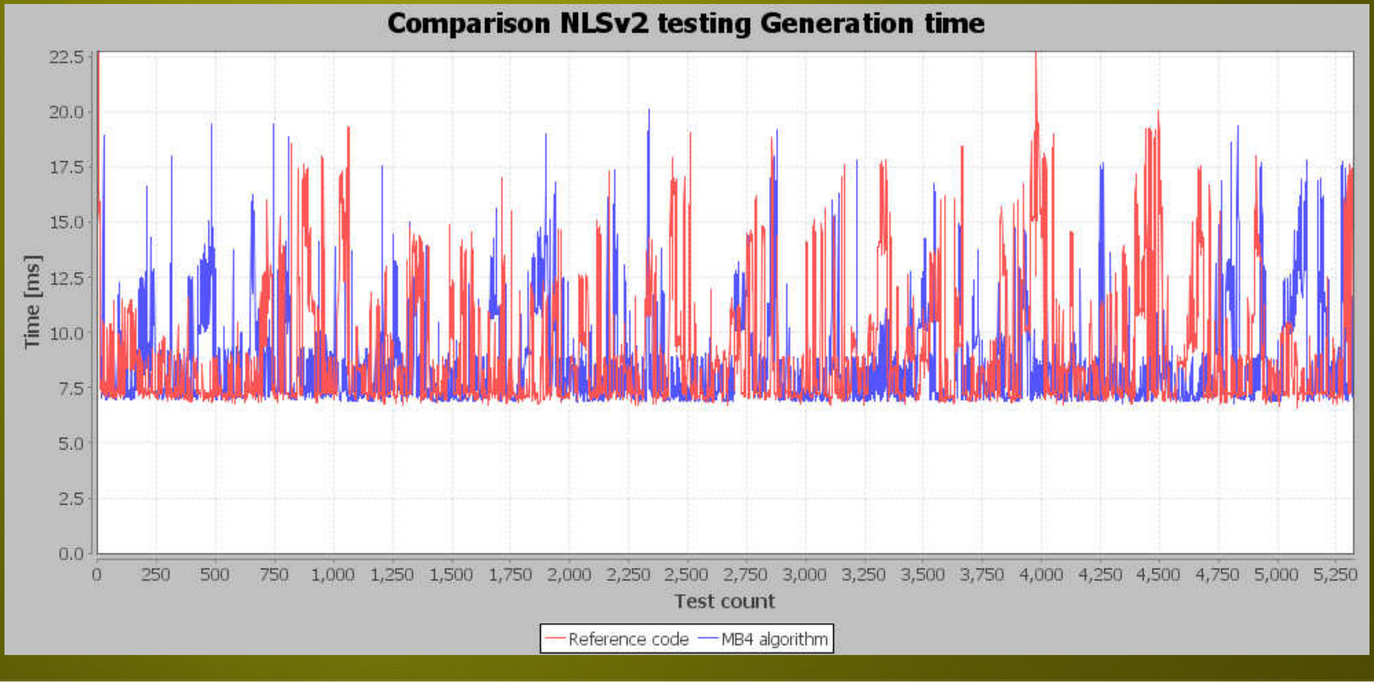
a) average time



red curve – reference implementation

blue curve – implementation no.4

b) instant time



Conclusion

- Results:
  - faster generation of short keystreams up to 28%
  - slower generation of middle sized keystreams up to 5%
  - faster generation of longer keystreams about 5%
- Future work:
  - profiling analysis of the tested implementations
  - find reason why middle sized keystream generation is slow
  - other improvements of the cipher NLSv2 (quality, statistical tests, etc.)

References

[1] P. HAWKES, M. PADDON, G.G. ROSE, M.W. DE VRIES: *Primitive Specification for NLSv2*. Published at the project eSTREAM web page: http://www.ecrypt.eu.org/stream/p3ciphers/nls/nls_p3.pdf (2007), pages 1-25.

[2] S. BABBAGE, CH. DE CANNIÈRE, A. CANTEAUT, C. CID, H. GILBERT, T. JOHANSSON, M. PARKER, B. PRENEEL, V. RIJMEN, M. ROBSHAW: *The eSTREAM Portfolio*. Published at the project eSTREAM web page: http://www.ecrypt.eu.org/stream/portfolio.pdf (2008), page 4.

[3] M. HUTNÍK: *Benchmark of random bit generators in different programming languages.* Bachelor thesis (2012).

Thank you

for your attention ☺