The Stream Cipher “Polar Bear“

Johan Håstad
Royal Institute of Technology

Mats Näslund
Ericsson Research

Stockholm, Sweden

Outline

- Design criteria/principles
- Cipher Overview
- Rationale
- Security
- Performance

Design Criteria

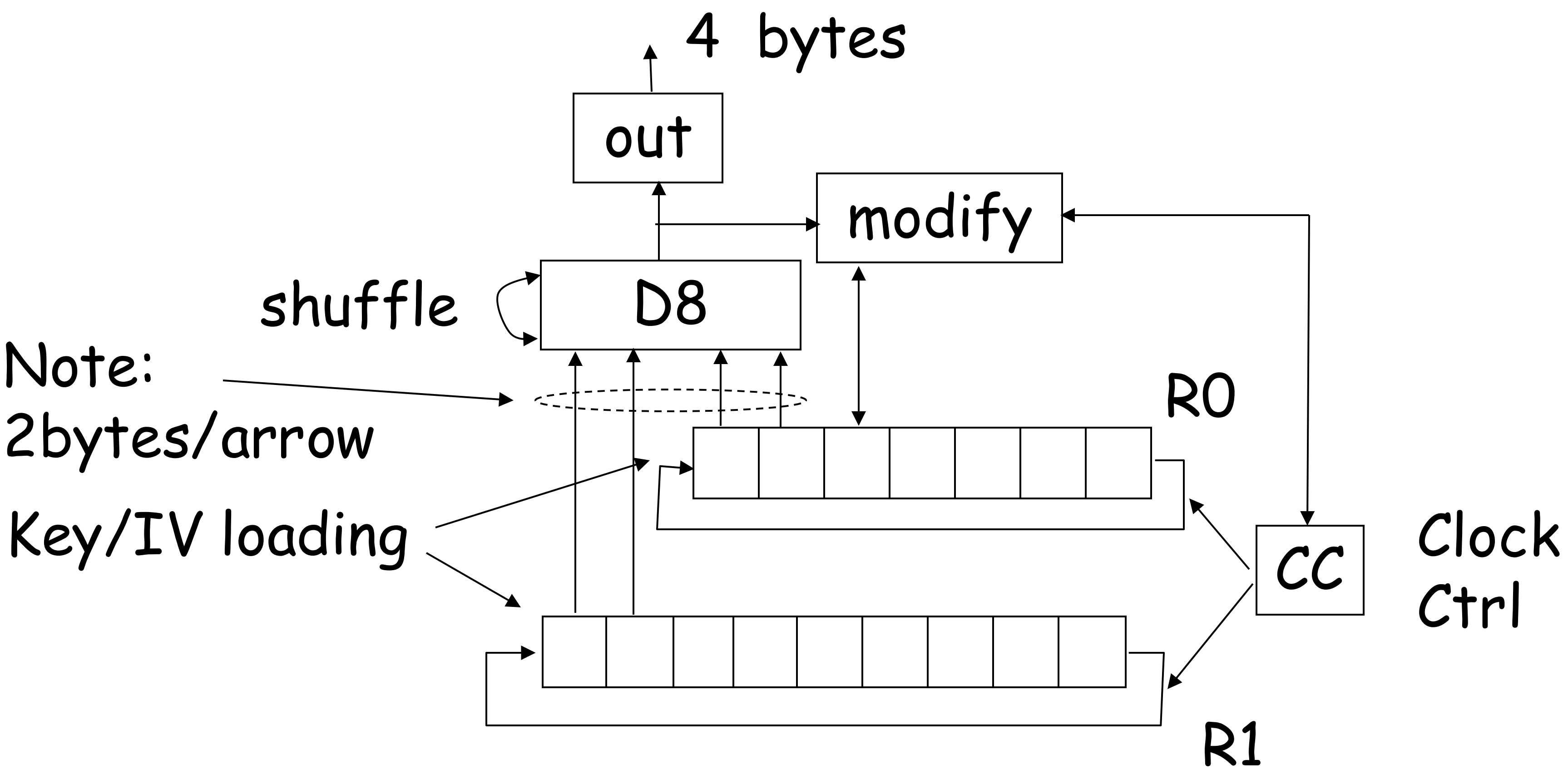
- Security:
  key-size dependent, 80-128 bit
- Re-keying:
  flexible IV handling (up to ~ 256 bits)
- Performance:
  signifcantly faster than AES_CTR

Design Principles

Re-use of well-known components:

- LFSRs, guarantee for large period
- Irregular clocking
- “Table shuffling” a la RC4, highly non-linear
- Borrow Rijndael (AES) components, good mixing of key/IV.

Polar Bear Overview



LFSRs of size 7 and 9 over GF(2^{16}): 256-bit state (++)

Polar Bear Operation

1. Key-schedule (once per key)
2. Initialization (once per message)
3. State update/Output generation

Polar Bear Key Schedule

Identical to Rijndael key schedule:

128-bit key expanded to key for 5-round Rijndael (256-bit block version), i.e. (5+1) x 4 x 8 bytes = 192 bytes

Polar Bear Initialization

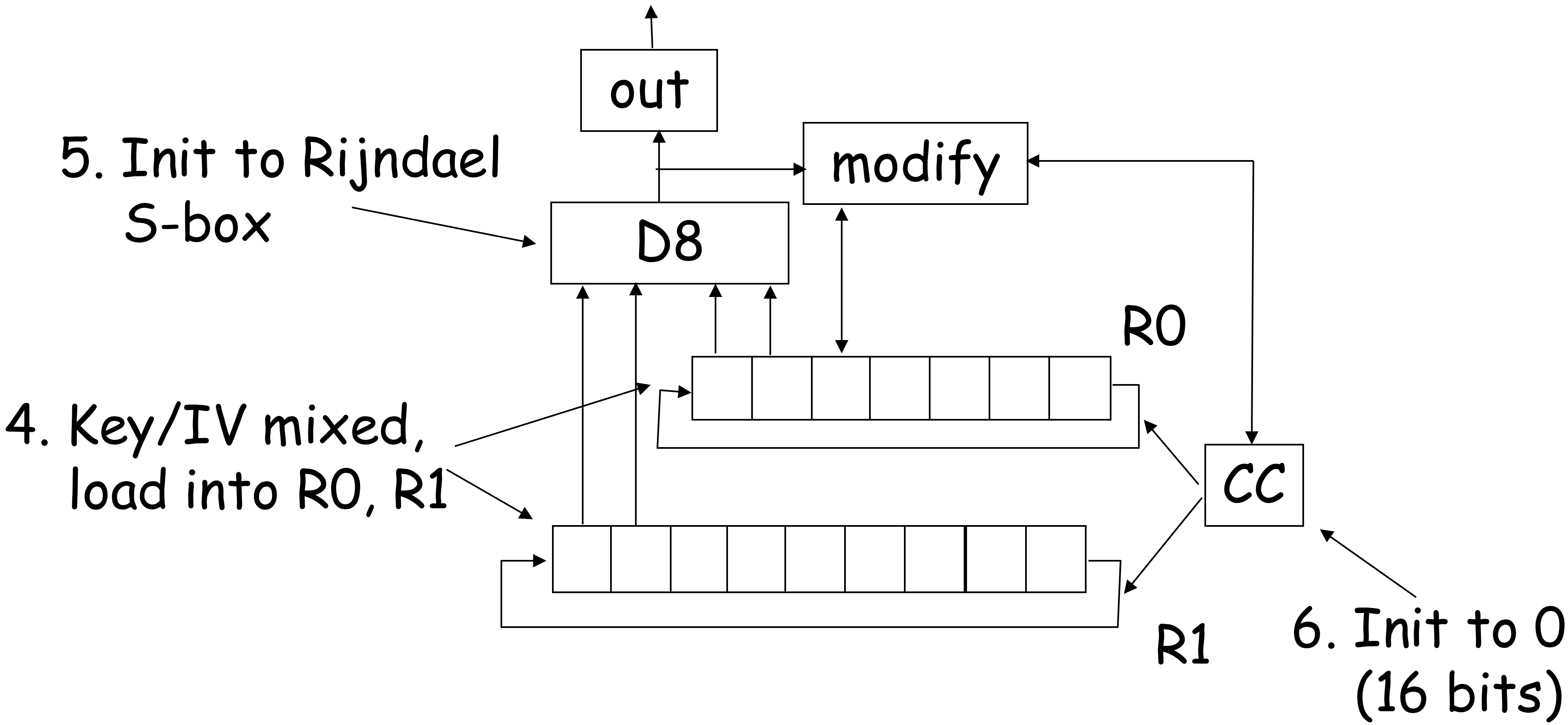
Also borrowed from Rijndael

1. Pad IV (prefix free 100… pad) to 32 bytes
2. Write IV as 4 x 8 "Rijndael matrix":

IV_0	IV_4		
		…	
IV_3			IV_{31}

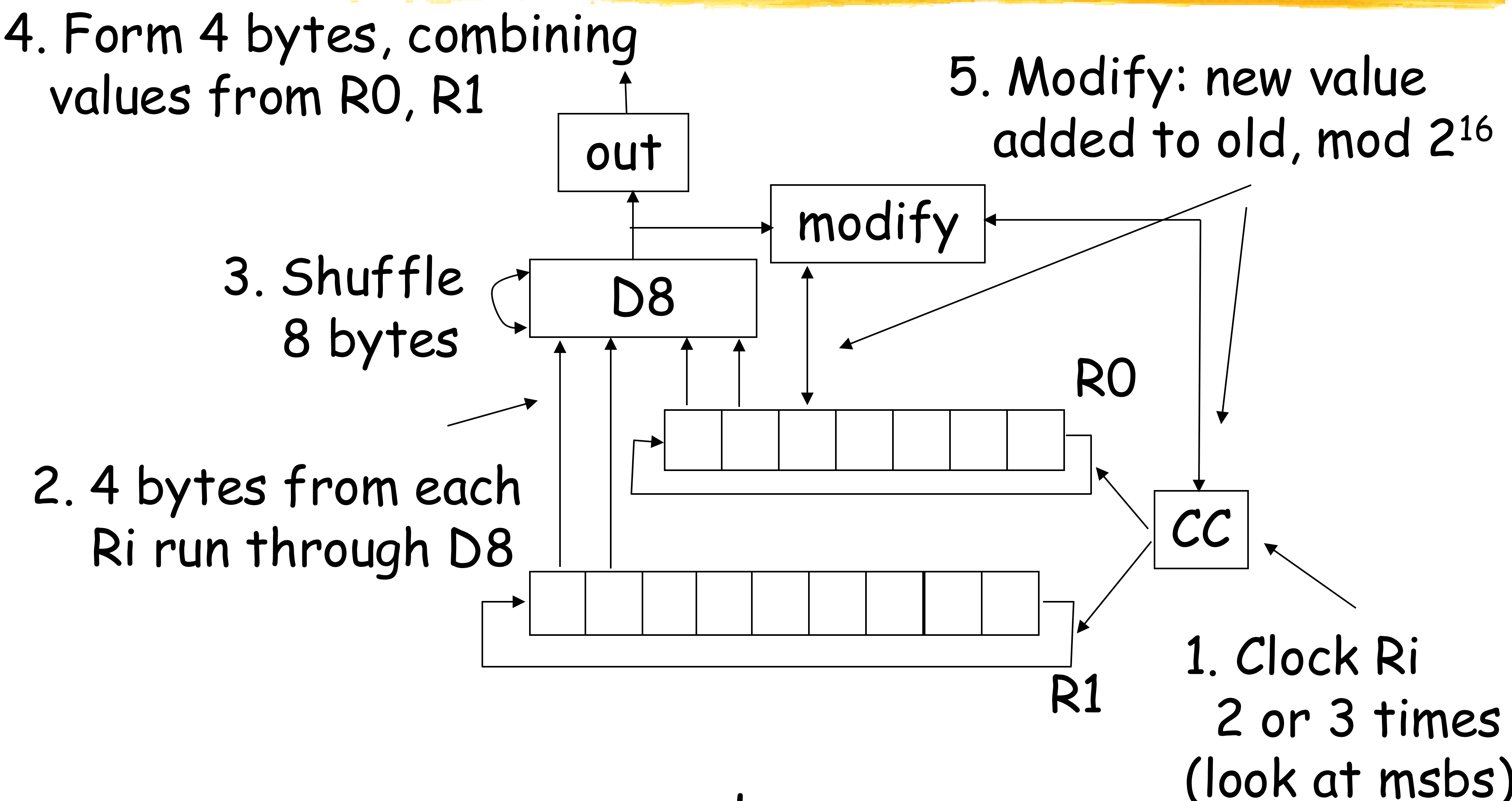
3. Encrypt this using key for 5 rounds ("MixColumn" in all rounds)

Polar Bear Initialization (cont'd)



Ready to go…

Polar Bear Output Generation



…and so on…

Rationale: Initialization

- Good mixing of IV with key: resembles block-cipher encrypt operation
- Rijndael-128 has some good properties already for 4 rounds
- Since we use Rijndael-256, we chose to add one round
- Good Rijndael implementation can easily be "plugged in", re-use optimizations

Rationale: use of LFSRs

- Well-understood properties
- One "conventional" LFSR to guarantee period
- One "non-conventional" plus irregular clocking to make cryptanalysis harder

Rationale: “RC4” Table Shuffling

- Seems to rule out linear cryptanalysis
- Increased state-space
- By feeding table by LFSR values, we still have control over period
- Better pre-mix to avoid RC4 weaknesses

Caveat: on short messages, not much shuffling will take place. Propose smaller table version for such applications.

Security

On long messages, we argue we have a strengthened RC4:

- Better initial mixing
- Individual table-entries not output directly
- Period guarantee

On short messages, a “reduced Rijndael”:

- Outputs are combinations of two D8-values, each dependent on at least three Rijndael bytes

Performance

Only preliminary benchmarks done.

- Short packets: expect to be almost three times faster than Rijndael-256. (By using 5 out of 14 rounds.)
- Significantly faster than RC4 on short packets
- Longer packets: RC4 is 50-100% faster.

Performance (II)

Preliminary (non-optimized) benchmarks (init + keystream).

Msg size	Speed (Mbit/s)
32	135
64	155
128	170
256	180
512	184
1024	190

(1400MHz Pentium, MS Visual C, $M = 2^{20}$)

Final Remarks

- Please refer to paper for details, in particular
  - ✓ Handling of other key-sizes
  - ✓ Alternative with smaller table
- Implementation suggestions