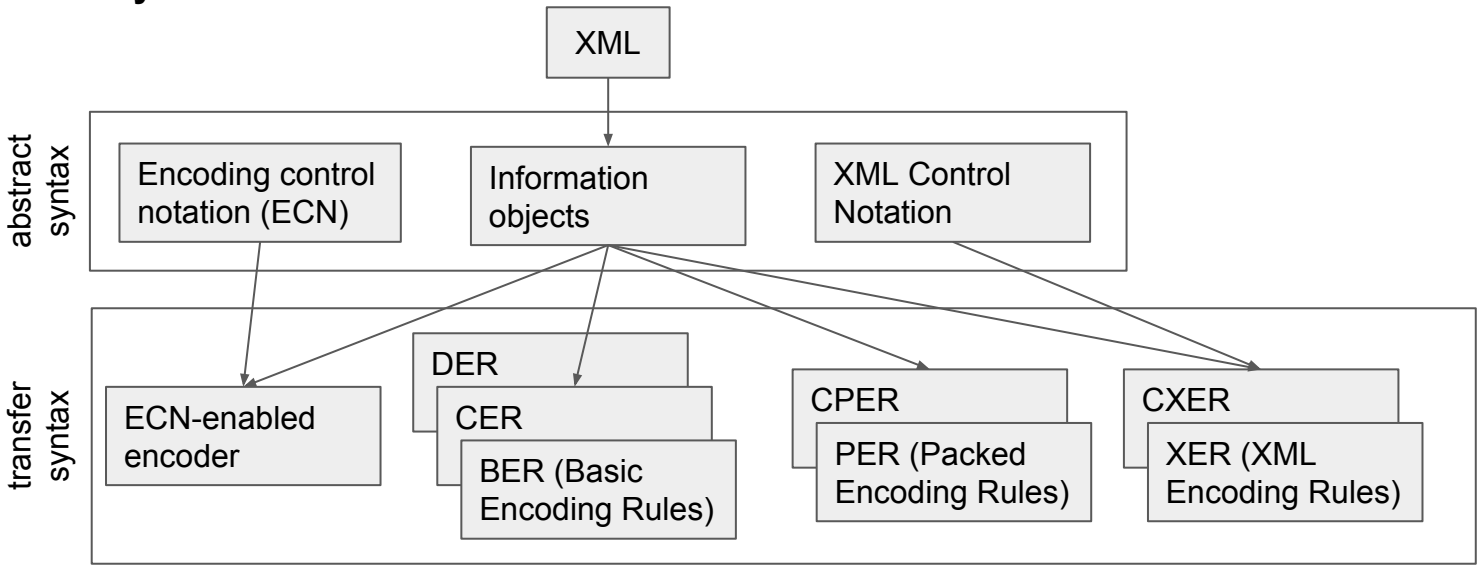
Formal verification of floating-point number conversion between ASN.1 BER and IEEE 754 binary encodings

Ilia Zaichuk
Taras Shevchenko National University of Kyiv,
Digamma.ai

ASN.1

- Used in GPS, LTE, DNS, SIM, RFID, etc.
- 8 ISO documents, 5 correction docs, 507 pages total
- 66 790 lines of code in reference C implementation

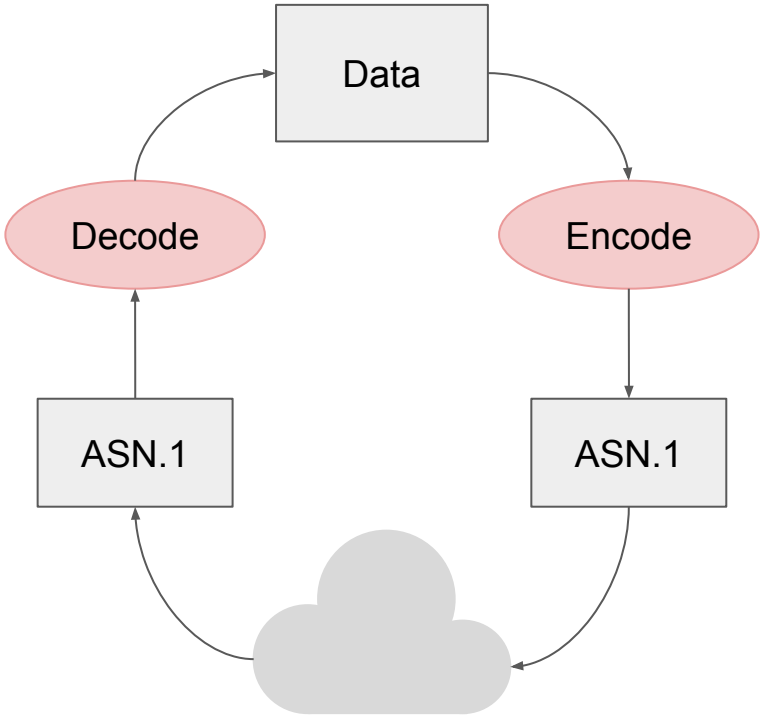
Abstract Syntax Notation One:



The Idea

ASN.1 formal verification

Use the Coq proof assistant to formalize encoders/decoders for different formats supported by ASN.1

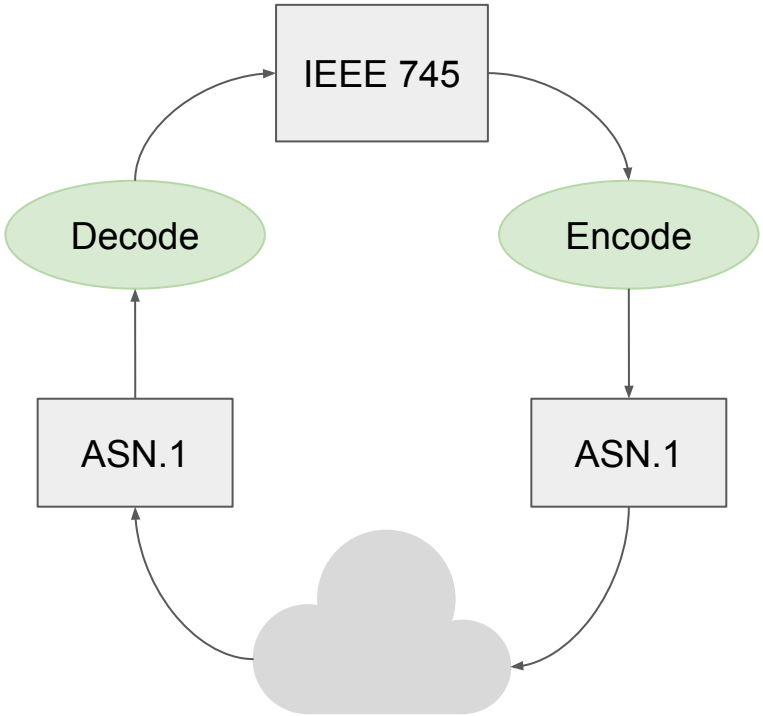


Proof of concept

Floating-point encoding/decoding is a small but error-prone part of ASN.1

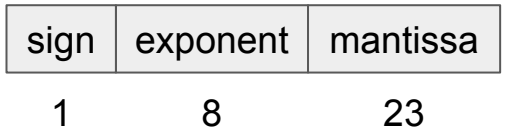
IEEE 754 FP numbers needs to be converted to ASN.1 for data transfer.

The most practical subset - binary floats

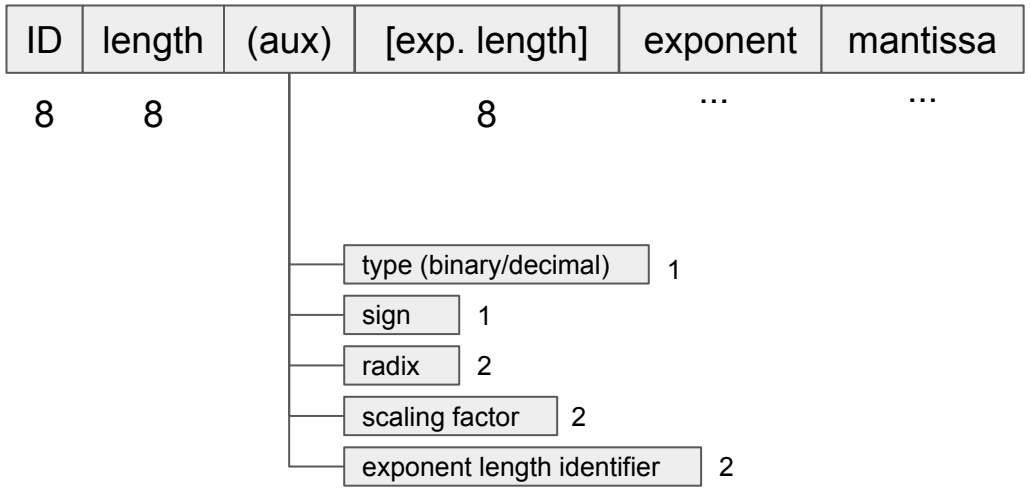


The Encodings

IEEE 754 “binary32”:



ASN.1 BER real:



Key components

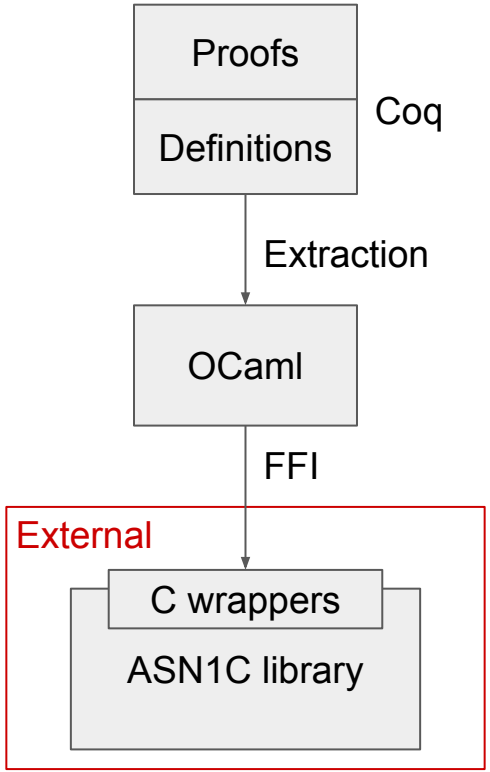
1. Formalization of ASN.1 data structures as Coq types
2. Executable specification (encoder/decoder implemented in gallina)
3. Flocq (Floats for Coq) library for IEEE 754 handling
4. Library for bit string representation and manipulation
5. Abstract formulation of encoder/decoder correctness via the notion of a “roundtrip” (heterogenous inverse)
6. Error handling (`F : A -> option B`)

Flocq: flocq.gforge.inria.fr

System Architecture

“Executable specification” - verified implementation of encoder/decoder

Extract it. Link it.



Top-level theorem

Heterogeneous inverse

```
Definition bool_het_inverse
           (A1 B A2 : Type)
           (f : A1 -> B)
           (b : B -> A2)
           (e : A1 -> A2 -> bool)
           (x : A1)
: Prop :=
  e x (b (f x)) = true.
```

Full roundtrip

```
Theorem main_roundtrip (scaled : bool) (f : IEEE_float) :
  is_Some_b (BER_of_IEEE_exact f) = true ->
  bool_het_inverse
    (option IEEE_float) BER_float IEEE_float
    BER_of_IEEE_exact
    IEEE_of_BER_exact
    float_eqb_nan_t
    f
  = Some true.
```

Results

- Reference C implementation: ~200 LOC
- Coq specifications: 1,488 LOC
- Coq proofs: 822 LOC
- OCaml test suite: 141 LOC

Lessons learned

- Floating-point number conversion is not the simplest case for a trial
- ASN.1 is even more complex than it seems
- Substantial amount of work is required to finish full ASN.1 in Coq
- Coq is not perfect for implementing bit encoders/decoders

Future work

- Plug extracted code into ASN1C library
  - Perform unit tests
  - Measure performance
- Attempt the same task using F* instead of Coq
- Try to validate C implementation as semantically equivalent to our executable specification

Questions

Git: `github.com/digamma-ai/asn1fpcoq`

Email: `izaichuk@digamma.ai`

Digamma.ai is looking for academic and industrial partners to collaborate on the development of a fully certified ASN.1 stack