What's New in the Community Books

Since the ACL2-2023 Workshop

Alessandro Coglio[2,3,4] (presenter),
Grant Jurgensen[2], Matt Kaufmann[5],
Eric McCarthy[2,4], J Strother Moore[5],
Eric Smith[2,3], Yahya Sohail, Sol Swords[1]

[1]Arm Inc., [2]Kestrel Institute, [3]Kestrel Technology LLC,
[4]Provable Inc., [5]University of Texas at Austin (retired)

ACL2-2025 Workshop

Overview

- Over 7,000 non-merge commits since the last Workshop.
- From several contributors from several organizations.
- Spanning hardware, mathematics, cryptography, blockchain, programming languages, virtual machines, machine code, standards, analysis, synthesis, and more.
- These slides provide succinct summaries of major items, ordered by book path within each of the new and improved library parts. See the book release notes for more details.

New Libraries

`intel/ifp`: Floating-point arithmetic specs formerly used at Intel:

- Specs for core arithmetic operations and rounding.
- Some theorems about rounding and error bounds.

New Libraries

`kestrel/c/syntax`: Representation, with accompanying tools, of the syntax of all of C after preprocessing, including many GCC extensions:

- A tool to invoke an external C preprocessor.
- An abstract syntax that retains much of the information from the concrete syntax.
- An ABNF grammar.
- A parser that captures ambiguous constructs as such.
- A disambiguator to be used after parsing.
- A partial validator to check typing rules, scoping rules, etc., and to annotate the abstract syntax with validation information (e.g. types, linkage).
- A pretty-printer with some customization options.
- Tools to input/output C code from/to files.

New Libraries

`kestrel/c/transformation`: Tools for transforming C code, generating correctness proofs (in the supported subset of C):

- Constant propagation and folding.
- Simplification of certain kinds of expressions.
- Specialization.
- Function splitting.
- Struct splitting.

`kestrel/c/transformation/utilities`: Auxiliary utilities for C code:

- Call graph calculation.
- Identifier manipulation.

New Libraries

`kestrel/riscv`: A preliminary model of the RISC-V ISA:

- Unprivileged RV32IM and RV64IM (except `FENCE`, `HINT`, `ECALL`, and `EBREAK`).
- Little endian memory access.
- Fully readable and writable address space.
- Being extended to support different bases and extensions, via a data structure representing the ISA features.

New Libraries

`kestrel/treeset`: Ordered finite sets represented as treaps:

- A treap (= "tree" + "heap") is a binary search tree with an additional max heap property.
- Unique representations, like `osets`.
- Logarithmic complexity for membership, insertion, and deletion.

New Libraries

`kestrel/xml`: Add a simple XML parser.

New Libraries

`projects/acl2-in-hol`: Support for much of the ACL2 side of the dormant link from ACL2 into HOL.

`projects/hol-in-acl2`: A translator from HOL4 to ACL2, with examples.

New Libraries

`projects/aleo`: Formal specifications and proofs about the Aleo blockchain and ecosystem:

- Formal models and correctness proofs of AleoBFT, a Byzantine-fault-tolerant blockchain consensus protocol.
- Definition and proofs of the ABNF grammar of the Leo programming language for zero-knowledge applications.

New Libraries

`projects/poseidon`: Poseidon cryptographic hash:

- Parameterized definition.
- Various instantiations.
- Tests for the instantiations.

New Libraries

`projects/schroeder-bernstein`: A proof of the Schröder-Bernstein theorem:

- Described in this year's workshop paper, "A Proof of the Schröder-Bernstein Theorem in ACL2."

New Libraries

`projects/set-theory`: Set theory:

- Supports the use of ACL2 to reason about first-order set theory.
- Provides an alternative to `apply$` for higher-order functional programming.

Improved Libraries

build: New build options:

- Add new cert_param non-acl2p, to exclude books that don't work with ACL2(p).
- New FAILURE_LINES variable limits printing for failed/interrupted builds.

Improved Libraries

`centaur/fgl`: Bit-blasting rewriter.

- Added/improved some debugging utilities.
- Added feature to match a hyp containing free variables against assumptions.
- Added documentation about binder functions/rules.
- Added some utilities to help support rewriting of FTY sum/product types.
- Improved integration with AIG library to make it more feasible. to programmatically configure AIG transformations.

Improved Libraries

centaur/fty: Fixtype library (the part under centaur/):

- Added `multicase` macro for matching against multiple sum types at once.

Improved Libraries

centaur/sv: VL/SV hardware verification package:

- Improvements in reasoning about designs as FSMs (as opposed to unrolled pipeline functions).
- Add SRT4 divide tutorial example.

Improved Libraries

demos: Demos (1/2):

- Added demos for `attach-stobjs` (in `demos/attach-stobj`); see also directory `system/tests/attachable-stobjs/`.
- Added demos for floating-point operations (in `demos/fp`); see also extensive suite in `books/demos/floating-point-input.lsp` with corresponding output file `books/demos/floating-point-log.txt`.
- Added `demos/swap-stobj-fields.lisp` to demonstrate how to swap two stobj fields of a given stobj and to illustrate the corresponding implicit update of the parent stobj.
- Added `demos/element-type.lisp` to show how to specify the raw-Lisp element type for a stobj array field.

Improved Libraries

demos: Demos (2/2):

- Added demos/defabsstobj-example-1-df.lisp to illustrate the use of dfs (ACL2 floats) with abstract stobjs.
- Added demos/ppr1-experiments-thm-1-ppr1.lisp and demos/ppr1-experiments.lisp to illustrate how ruler-extenders can sometimes improve the induction scheme suggested by a recursive function (see :DOC induction-coarse-v-fine-grained).
- Added demos/toy-csort.lisp as an interesting exercise in generalization.
- Added answers to the problems in :DOC introduction-to-apply$ in projects/apply/answers-to-doc-intro-to-apply.lisp.

Improved Libraries

`doc/100-theorems`: More of the 100 Theorems now in ACL2:

- Cramer's Rule.
- Mathematical Induction.
- Schröder-Bernstein (2 versions).
- Cantor's Theorem.

Improved Libraries

kestrel/axe: Major improvements to:

- Rewriters (e.g., binding hyps, SMT to relieve hyps, XOR nest handling).
- SMT connection.
- Equivalence Checker.
- Provers (Tactic Prover, R1CS Prover, etc.).
- Clause Processors.
- x86 Binary Lifter.
- JVM (Java) Code Lifter.
- Formal Unit Testers.
- Supporting rules.

Improved Libraries

`kestrel/crypto`: Add formalizations:

- Add spec of the CHACHA20 stream cipher.
- Add spec of the BLAKE2b hash function.

Improved Libraries

`kestrel/ethereum`: Library about the Ethereum blockchain:

- Add partial formal specification of the Ethereum Virtual Machine.

Improved Libraries

`kestrel/fty`: Fixtype library (the part under the Kestrel Books):

- Added a new `deffold-reduce` tool to generate a class of "reducing" fold functions over fixtypes.
- Added some utilities to query the fixtype database, extending the ones under `centaur/fty`.
- Made `defset` and `defomap` more robust.

Improved Libraries

`kestrel/helpers`: Improvements to tools that:

- Help find new proofs and fix broken proofs.
- Suggest improvements to functions/theorems/books.
- Find problems (Linter).
- Speed up proofs/books.

Improved Libraries

`kestrel/terms-light`: Add several verified term simplifiers:

- Improve lambdas (subst constants, drop unused/trivial/unnecessary bindings).
- Simplify ORs.

Improved Libraries

`kestrel/x86`: Tools for reasoning about x86 binary code:

- Normal forms for x86 reasoning.
- Equivalent alternate defs for certain instructions.
- Many rules (including new rewriting strategies).
- Auto-generation of assumptions for proving/lifting.
- Handling of memory reads, writes and disjointness.

Improved Libraries

kestrel: Many improvements to basic libraries:

- acl2-arrays (e.g., much longer arrays)
- arithmetic-light
- booleans
- bv and bv-lists (including bv-arrays)
- clause-processors
- evaluators (e.g., defevaluator+)
- file-io-light
- floats (rounding, connection to RTL)
- json-parser
- lists-light, typed-lists-light, alists-light
- prime-fields
- sequences
- strings-light and unicode-light
- utilities
- world-light

Improved Libraries

`projects/x86isa`: Formal model of the x86 ISA:

- Added support for a number of new instructions.
- Added some test tools.
- Added TLB model.
- Added TTY and timer peripherals.
- Added Linux support with patched kernel.
- Added cosimulation validation tool.
- Sped up many books.
- New summary of instruction coverage (see `:doc sdm-instruction-set-summary`).