

What's New in the Community Books

Since the ACL2-2020 Workshop

Jagadish Bapanapally^{9,5}, Cuong Chau²,
Alessandro Coglio^{1,5,6} (presenter), Shilpi Goel³,
Matt Kaufmann⁸, Panagiotis Manolios⁷,
Eric McCarthy^{1,5}, J Strother Moore⁸,
David Russinoff², Eric W. Smith^{1,5,6},
Sol Swords⁴, Stephen Westfold⁵

¹Aleo, ²ARM, ³Centaur, ⁴Intel,

⁵Kestrel Institute, ⁶Kestrel Technology,

⁷Northeastern University,

⁸University of Texas at Austin (retired),

⁹University of Wyoming

ACL2-2022 Workshop

Overview

- Over 8,000 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 provides a more succinct overview than the book release notes, ordered by book path within each of the new and improved library parts.

New Libraries

centaur/bigmem: A library that defines a big array (up to 2^{64} bytes).

- Offers the reasoning efficiency of records.
- Offers efficient execution via nested stobjs containing resizable arrays.

New Libraries

kestrel/acl2-arrays: Support for reasoning about programs that use ACL2 arrays (e.g., values satisfying `array1p`).

- Many rules about existing functions.
- New operations that make arrays expandable.
- Tool for defining typed ACL2 arrays.

New Libraries

kestrel/acl2p1: Preliminary model of the ACL2 programming (not logical) language.

- Abstract syntax consisting of translated terms, functions, and packages.
- Small-step evaluation semantics.
- Program-mode interpreter.
- Lifter from packages and functions in the world to the formal model.

New Libraries

kestrel/algebra: A library for abstract algebra.

- Add a formalization of groups.
- Prove some simple properties, in a calculational style.

New Libraries

kestrel/algorithm-theories: A library to collect algorithm schemes.

- Scheme for a generic tail-recursive function.

New Libraries

kestrel/arrays-2d: A formalization of two-dimensional arrays as lists of lists.

- Supports arrays with arbitrary elements, and with elements that are known to be bit-vectors.

New Libraries

kestrel/c: Models, proofs, and tools for C (described in a paper at this workshop).

- ATC, the C code generator for ACL2.
- Deep embedding of C in ACL2.
- Shallow embedding of C in ACL2.

New Libraries

kestrel/clause-processors: Modular collection of clause-processors.

- Collect several clause-processors (subst, flatten literals, simple subsumption), many of which are verified.

New Libraries

kestrel/crypto/blake: BLAKE Library.

- Formal specifications of the BLAKE2s, BLAKE2s-extended, and BLAKE-256 hash functions.

kestrel/crypto/mimc: Minimal Multiplicative Complexity (MiMC) hash function.

- Formalization of the MiMC hash function.
- Uses a sponge construction.
- Used in Ethereum's Semaphore zero-knowledge gadget.

kestrel/crypto/salsa: Formal spec of the Salsa20 hash function.

New Libraries

kestrel/crypto/pfcs: Prime Field Constraint Systems (PFCS).

- Generalization of Rank-1 Constraint Systems (R1CS).
- Formal syntax and semantics.
- Preliminary reasoning support.
- Useful in zero-knowledge cryptographic proofs.

kestrel/crypto/r1cs: R1CS Library.

- A formal semantics for rank-1 constraint systems (R1CSes). These are often used in zero-knowledge proofs.
- Extensive support for reasoning about R1CSes (using ACL2 and Axe).
- Verified R1CS gadgets / gadget generators.

New Libraries

kestrel/evaluators: Simple evaluators for verifying clause-processors and metafunctions.

- Several evaluations for common sets of functions.
- New defevaluator+ tool (better defaults, extra/better theorems, support for lifting results to richer evaluators).

New Libraries

kestrel/isar: Tools for Isar-style proofs in ACL2.

- Isar = Intelligible semi-automated reasoning (proof language of the Isabelle theorem prover).
- Human-oriented (vs. machine-oriented) readable proofs.
- `:assume ... :let ... :derive ... :qed.`
- Useful, for instance, in proofs involving algebraic manipulations that do not follow simple rewriting directions.
- This library is just a small start.

New Libraries

kestrel/json: Models and tools for JSON.

- Abstract syntax of JSON.
- Converter from parser's abstract syntax to this abstract syntax.
- Includes b^* binder for pattern matching on JSON structures.

kestrel/json-parser: A JSON parser implemented in ACL2.

- Unicode support.

New Libraries

kestrel/jvm: Kestrel's model of the Java Virtual Machine.

- Support for code proofs and lifting.
- Works with Kestrel's Axe toolkit.
- Includes a class file parser.

New Libraries

kestrel/number-theory: A library about number theory.

- Primality, divisibility, quadratic residues, etc.
- Includes the `defprime` and `defprime-alias` tools to introduce standard reasoning machinery for primes.
- Includes verified Tonelli-Shanks algorithm for modular square roots.

New Libraries

kestrel/random: A lightweight library containing some simple random number generators.

New Libraries

kestrel/sequences: A library for defining higher-order operations over lists.

- `defforall`
- `defexists`
- `defmap`
- `deffilter`

New Libraries

kestrel/simpl-imp: A simple didactic programming language, Imp.

- Formal syntax and semantics.
- Useful for didactic purposes.
- Used in the literature.

New Libraries

kestrel/solidity: Small start towards a model of Solidity, the main smart contract language for Ethereum.

- A model of (some) Solidity values.

New Libraries

kestrel/strings-light: A lightweight library about strings.

- Changing case.
- Splitting strings and character lists.
- Checking suffixes.
- Books about reverse and length.
- Parsing chars as digits (lightweight).

New Libraries

kestrel/syntheto: Models, proofs, and tools for Syntheto (described in a paper at this workshop).

- Surface language for APT and ACL2.
- Formalization of abstract syntax.
- Formalization of static semantics.
- Translation to ACL2 (Syntheto back end).

New Libraries

kestrel/terms-light: A lightweight library of operations on terms.

- Find free and bound vars. Rename vars.
- Substitution and evaluation.
- Add/remove/serialize lets / lambdas.
- Reconstruct mv-lets.
- Check terms (closed lambdas, no duplicate lambda vars, nil not used as a var).
- Search, count, create, and transform terms.

New Libraries

kestrel/typed-lists-light: A lightweight library dealing with lists of objects of particular types (rather than lists in general).

- Lists of integers, lists of symbols, lists of pseudo-terms, etc.

New Libraries

kestral/unicode-light: A lightweight library about Unicode.

- UTF-8 encoding.
- UTF-16 surrogate code points.

New Libraries

kestrel/untranslated-terms: A new library for manipulating untranslated terms.

- Allows structure to be maintained that would be lost by translation.

New Libraries

kestrel/x86: Kestrel's x86 proof machinery, which complements the X86ISA model.

- Focus on readability of proof terms.
- Supports the lifting of x86 code into logic with the Axe toolkit.
- Includes parsers for PE and Mach-O executables.

New Libraries

kestrel/yul: Models, proofs, and tools for Yul, an intermediate language used in the Solidity compiler.

- Concrete syntax and parser.
- Abstract syntax.
- Static semantics.
- Dynamic semantics.
- Static soundness proof.
- Covers all of 'generic Yul'.
- Some verified Yul-to-Yul transformations (used in the Solidity compiler).

New Libraries

kestrel/zcash: Models and proofs for the Zcash blockchain.

- Formalization of some zero-knowledge-related operations.
- Formalization and verification of some R1CS gadgets / gadget generators.
- New `verify-zcash-r1cs` tool based on *Axe*.

New Libraries

projects/execloader: Binary loaders.

- Read in sections of ELF/Mach-O files into ACL2. An older version of these books used to live in the x86isa library.
- Simplified elf-reader; ELF binary header, all section headers, and all section contents are now stored in the elf stobj. Previously, only a handful of commonly-used sections (e.g., .text, .data, .rodata, etc.) were parsed.
- Added support for getting information from ELF symbol table using functions `get-symtab-entries` and `get-label-address`.

New Libraries

std/obags: Ordered bags (obags).

- Similar to osets and omaps, for bags (i.e. multisets).
- Modeled as totally (non-strictly) ordered lists.
- Include operations and theorems.

Improved Libraries

ac12s: ACL2s Sedan.

- Added properties book with `definec`-like syntax supporting property-based design, testing and verification.
- Various improvements to `definec` and `defunc` regarding performance, debugging support, extensions.
- Updated utilities, such as `ac12s::match` macro, for pattern matching.
- More polymorphic support, built-in types, alias types, etc. in `defdata`.
- Improvements for counterexample generations with `cgen`, including using fixers in ACL2s.

Improved Libraries

arithmetic: Arithmetic library.

- Reduced what is exported.

Improved Libraries

build: Build system.

- Now `cert.pl` makes use of useless runes just as `make` does.
- Now `make` of the community books also writes book dependency information in S-expression form.
- Swapped the roles of `green` and `bold green` in build output (`bold` for slower books).

Improved Libraries

centaur/defrstobj2: Record-like stobj.

- `defrstobj2` can now be used to define stobjs with child stobj fields, i.e., fields based on another stobj.

Improved Libraries

centaur/fgl: FGL symbolic execution engine.

- Added incremental-minimize/maximize and minimize/maximize-ratio tools.

Improved Libraries

centaur/sv: Hardware verification backend.

- New flow for producing a symbolic unrolling (SVTV).
- More complete logical story for process from hierarchical design to unrolling.
- New utilities for proof (de)composition (def-svtv-override-fact).
- Improvements to SVTV-CHASE utility.

Improved Libraries

centaur/v1: SystemVerilog frontend.

- New support for SystemVerilog calls of static methods of parametrized classes.
- Improved preprocessor performance when there are lots of defines.
- Reduce deps of vl/util/namedb, used by defrstobj and the x86 model.

Improved Libraries

doc: Documentation.

- Web-based manual now includes clickable links to GitHub for doc topics from the Community Books.
- The file `[books]/doc/top-slow.lisp` is now `[books]/top.lisp` and no longer builds a manual. Instead, it detects name conflicts between community books.
- Manual building is now taken care of by `[books]/doc/top.lisp`.

Improved Libraries

kestrel1/abnf: Augmented Backus-Naur Form (ABNF).

- Refactored to move parser verification proof to separate file from parser.
- Refactored to collect parsing primitives usable in other parsers.
- Added preliminary parsing generation tools.

Improved Libraries

kestrel/alists-light: Lightweight alists library.

- New rules and books, including books on alistp, clear-key, rassoc-equal, and keep-pairs.
- Improve modularity.

Improved Libraries

kestrel/apt: Automated Program Transformations (APT) (1).

- Added new `schemalg` transformation to apply algorithm schemas, currently supporting divide-and-conquer schemas.
- Added new `solve` transformation to attempt to solve a specification, currently supporting the ACL2 and Axe rewriters as solvers.
- Added new `expdata` transformation to refine data types where each instance of the old data may be represented by multiple instances of the new data (i.e not isomorphic).
- Added new `drop-irrelevant-params` and `rename-params` transformations.

Improved Libraries

kestrel/apt: Automated Program Transformations (APT) (2).

- Added new wrap-output transformation to change a function's return type.
- Added new finite-difference transformation for incrementalization.
- Added new copy-function example transformation.
- Improved and extended existing isodata, restrict, simplify, and tailrec transformations.
- New tools, deftransformation and def-equality-transformation, to generate transformations, handling the boilerplate.

Improved Libraries

kestrel/arithmetic-light: Lightweight arithmetic library.

- Many new rules and books have been added, including books on integer-length ceiling-of-lg, evenness and oddness, truncate, rem, ash, min, max, <=, abs, and natp.

Improved Libraries

kestrel/axe: Axe Toolkit.

- Major additions; most of Axe is now open-source.
- Machinery for making customized Axe rewriters and provers.
- A new general-purpose prover and rewriter (guard-verified, :logic-mode code).
- Legacy prover and rewriter.
- Axe Tactic Prover.
- Axe Equivalence Checker.
- Connection to the STP SMT solver.
- Tools to expand/unroll/simplify specifications.
- Many rules useful in Axe proofs.
- Utilities to unroll specifications through rewriting.

Improved Libraries

kestrel/axe/jvm: Axe toolkit for JVM.

- Tools to lift JVM code into logic.
- Formal Unit Tester tool, for small solver-backed proofs about bounded executions of programs.

kestrel/axe/r1cs: Axe toolkit for R1CS (rank-1 constraint systems).

- Tools to lift R1CSes into logic and verify them.

kestrel/axe/x86: Axe toolkit for x86.

- Tools to lift x86 code into logic.

Improved Libraries

kestrel/bitcoin: Bitcoin library.

- Added formalization of the Bech32 and Bech32m checksummed base32 formats used to encode addresses in Segwit.

Improved Libraries

kestrel/booleans: Booleans library.

- New rules and defcongs.

Improved Libraries

kestrel/bv: BV (bit-vector) library.

- Over 1000 new rules.
- New books have been added covering many more BV operations, including subtraction, arithmetic negation, multiplication, shifts, bitwise OR and AND, logical negation, signed and unsigned comparisons, signed and unsigned division and remainder, trimming, sign extension, various single-bit operations, bit-vector-valued conditionals, converting between bits and booleans, recognizing bits and (signed and unsigned) bytes, repeating a bit, and counting the number of 1 bits.
- Rules to characterize signed addition overflow and underflow.
- Rules to turn BV ops into more common or more idiomatic operations.
- A formalization of one's complement numbers and addition.
- Various syntactic functions over BV-valued terms.

Improved Libraries

kestrel/bv-lists: BV-Lists library.

- Many new rules.
- New books about `bv-array-p`, `bv-array-read`, `bv-array-write`, `all-all-unsigned-byte-p`, `width-of-widest-int`, `bvnot-list`, `getbit-list`, `map-slice`, `bvplus-list`, `logext-list`, `bv-nth`, `map-packbv`, `all-signed-byte-p`, conversions between lists and bv-arrays, `packbv-little`, and `byte-listp`.
- New utilities that deal with patterns in the elements of BV lists.

Improved Libraries

kestre1/crypto/ecurve: Elliptic curve cryptography.

- Extended and improved formalization of short Weierstrass curves.
- Added formalization of twisted Edwards curves.
- Added formalization of Montgomery curves.
- Added formalization of birational equivalence between Montgomery and twisted Edwards.
- Added formalization of Edwards BLS12 curve.
- Added refinement of pfield-squarep.

Improved Libraries

kestrel/ethereum: Ethereum library.

- A new sub-library for the Semaphore gadget. Includes various specifications and proofs of Semaphore-related R1CSes, including a mixing function from BLAKE2s and 3 variants of the MiMC hash function.
- Semaphore-specialized Axe tools to lift R1CSes into logic and verify them.

Improved Libraries

kestrel/event-macros: Tools for event macros.

- Added utilities to create events from structured information.
- Added utility to set up a more controlled proof environment for generating proofs designed to never fail.
- Other improvements.

Improved Libraries

kestrel/file-io-light: Lightweight file I/O library.

- Various new lightweight utilities to read and write files (of bytes, characters, and objects).
- New utilities to read bytes and characters from files into stobj arrays.
- Reasoning support for various built-in I/O functions.
- Utilities to check whether a file exists or is newer than a given date.
- Proofs that bad channel names sometimes cannot occur.

Improved Libraries

kestrel1/fty: Fixtype library extensions in the Kestrel books.

- Mutual recursion (i.e. `deftypes`) now supports `defset` and `defomap`.
- Added a macro `defsubtype` for subtypes of existing fixtypes.
- Added a macro `defresult` for result types, i.e. unions of good results and error results (similar to Rust's `Result` type).
- Added several common fixtypes.

Improved Libraries

kestrel/helpers: Proof helpers.

- Draft tool for auto-generating return type theorems.
- Rudimentary tools to discover simple proofs.
- Rudimentary tools to improve existing books.
- Tools for processing book dependency info.

Improved Libraries

kestrel/java: Models, proofs, and tools for Java.

- AIJ has been optimized, and extended with more Java implementations of ACL2 built-in functions.
- ATJ has been extended and improved (see the paper on ATJ at this Workshop).
- Extended the formalization of (some aspects of) the Java language.

Improved Libraries

kestrel/library-wrappers: Library Wrappers.

- A new, robust variant of make-flag (sped up some proofs by 100x using a custom clause processor).

Improved Libraries

kestrel/lists-light: Lightweight lists library.

- Many new rules and congruences.
- New books about the functions `subsequencep`, `subsequencep-equal`, `last-elem`, `subrange`, `update-subrange`, `finalcdr`, `all-equal$`, `all-eql$`, `all-same`, `all-same-eql`, `add-to-end`, `first-non-member`, `group` and `ungroup` (for splitting and flattening), `count-occs`, `prefixp`, `len-at-least`, `remove-equal`, `remove-duplicates-equal`, `find-index`, `remove-nth`, `make-list-ac`, `resize-list`, and `replace-item`.
- New book about functions that treat lists like sets.

Improved Libraries

kestrel/prime-fields: Prime fields library.

- Many new/improved rules.
- Rules for recognizing R1CS gadgets.
- bind-free rules for canceling addends and moving negations.

Improved Libraries

kestrel/soft: Second-Order Functions and Theorems (SOFT).

- Added macro `defsoft` to record already introduced functions into the SOFT table for possible later instantiation.
- Added macros `define2`, `defund-sk2`, `define-sk2` as second-order versions of the existing macros.
- Added macro `defequal` to introduce second-order equalities.

Improved Libraries

kestrel/std: Standard library extensions in the Kestrel books.

- Added several system utilities.
- Added macro `defund-sk` that disables function and theorem.
- Added macros `defmapping`, `definj`, `defsurj` to introduce and verify mappings between predicates.
- Added macro `tuple` to mimic `mv` return specifiers inside components of `mv` return specifiers (particularly, the value component of error triples).
- Added macro `defmin-int` to declaratively define the minimum of a (possibly infinite) set of integers.

Improved Libraries

kestrel/utilities: Utilities in the Kestrel books (1).

- Added macro checkpoint-list, which provides a programmatic, flexible interface to the key checkpoint information.
- A new ACL2 Lint tool can detect common ACL2 errors and suggest improvements to functions and theorems. Led to quite a few fixes in the Community books.
- Reasoning about I/O channels has been improved.
- New utilities support computing a constant using make-event, reading a value from a file into a defconst, and printing constants nicely.
- A new tool, bind-from-rules can bind free variables in rules by searching existing rules.
- Various improvements have been made to defopeners (and it now subsumes defopeners-mut-rec).

Improved Libraries

kestrel/utilities: Utilities in the Kestrel books (2).

- A new data structure, string trees, can efficiently represent a sequence of strings (e.g., for writing to a file).
- A new tool supports polarity-based rewriting, whereby a term can be either strengthened or weakened depending on whether it is an assumption or a conclusion.
- New sorting utilities, including `split-list-fast`, `merge-sort-generic`, and `defmergesort`.
- New XDOC constructors, e.g., one that creates XDOC paragraphs from blocks of text separated by blank lines.
- A new tool, `gen-xdoc-for-file`, to generate XDOC topics for every event in a file by extracting the relevant lines of code (the definition and any immediately preceding or following comment lines).
- Helpful wrappers for XDOC archiving utilities.

Improved Libraries

kestrel/utilities: Utilities in the Kestrel books (3).

- New utilities about event forms, making fresh names, manipulating hints, building simple list structures, dealing with quoted entities, building strings, checking whether a symbol has properties, dealing with runes, parsing options, processing keyword args, and recognizing legal variable names.
- New utilities about redundant events, guard-holders, ruler-extendors, lets/lambdas, worlds, clause identifiers, progn, unification, dependencies, ensuring rules are known, quieting make-event, processing defun and defthm forms, processing declares, the ACL2 state, system-books-dir, fixing functions, acl2-count, make-ord, coerce, map-symbol-name, tuples, myif, mv-nth, explode-nonnegative-integer, explode-atom, intern-in-package-of-symbol, supporting functions, constant names, nat-to-string, and binary-pack.

Improved Libraries

kestrel/utilities: Utilities in the Kestrel books (4).

- New tool `defstobj+`: Drop-in replacement for `defstobj` that disables functions and proves many rules (scalar and array fields only).
- New utility `with-local-stobjs` (extends `with-local-stobj` to support multiple `stobjs`).
- Draft of a tool to specialize theorems.
- New `defcalculation` tool for proofs that chain equalities.
- New books about `assoc-keyword`, `theory-invariants`, `chk-length-and-keys`, `member-symbol-name`, `arities`, `negation`, `logic-term`, `messages`, reconstructing macro calls, `defun` and `mutual-recursion` forms, `macro args`, `digit-to-char`, finding where a name was introduced, making lists of symbols, and manipulating conjuncts/disjuncts.

Improved Libraries

kestrel/utilities: Utilities in the Kestrel books (5).

- New utilities about imported symbols, format strings, printing, translation, manipulating terms, invariant risk, submitting events to ACL2, creating temp dirs, process IDs, usernames, calling scripts, macroexpansion and translation, and asserts.
- New or improved utilities about :program mode, prove\$, directed-untranslate, ignores, translation (tolerating ignored vars), tables, symbol creation, disjoin, adding documentation to macros (defmacrodoc), verifying guards, and non-normalized names.

Improved Libraries

nonstd: Non-standard analysis.

- Formalization of Banach-Tarski paradox in ACL2(r), at `nonstd/nsa/Banach-Tarski/`.
- Properties of 3D rotations using the `array2p` data structure in ACL2(r), at `nonstd/nsa/Banach-Tarski/rotations.lisp`.
- Integration by substitution in ACL2(r), and proof of the area of a circle in ACL2(r), at `nonstd/integrals/u-substitution.lisp`.

Improved Libraries

projects/apply: apply\$ and loop\$ tools.

- Replaced top.lisp with:
 - apply.lisp, to reason about apply\$.
 - loop.lisp, to reason about loop\$ (this also includes apply.lisp).
 - top.lisp, which includes apply.lisp and loop.lisp, and is thus the same as loop.lisp.
- Made the inclusion of certain supporting books local.

Improved Libraries

projects/rac: Restricted Algorithmic C (RAC).

- The tuple template can have up to *eight* arguments.
- Support the struct data type.
- Report more detailed error messages.

Improved Libraries

projects/x86isa: X86ISA, the formal model of the x86 Instruction Set Architecture.

- Simplified the treatment of CPUID features.
- Added the MOVD and MOVQ instruction variants that move data from/to the XMM registers.
- Simplified state definition by using centaur/bigmem and centaur/defrstobj2. See :doc x86isa-state-history for details.

Improved Libraries

rtl: Register-transfer logic library.

- Added a signed version of the radix-4 Booth encoding algorithm.
- A new section of `books/rtl/rel11/lib/round.lisp` on underflow detection, corresponding to Section 6.7 of “Formal Verification of Floating-Point Hardware Design”, 2nd edition.

Improved Libraries

std: Standard library.

- Added macros `add-io-pairs` and `merge-io-pairs` to speed up execution using verified input/output pairs. The idea is if you have a simple function that is infeasible to execute but feasible to verify on a concrete I/O pair, then that proof can be used to memoize the function to make execution feasible on that input, without changing its definition or any callers.
- Moved macro `define-sk` from the Kestrel books.
- Added several typed alists.

Improved Libraries

tools: Miscellaneous tools.

- Added macro `rewrite$`, which provides a programmatic, flexible interface to the ACL2 rewriter.
- Improved macro `prove$`, which provides a programmatic, flexible interface to the ACL2 prover.
- Improved `make-flag` (support computed hints better, better template theorems).