The Number Behind the Calculator

Unsigned arithmetic, reusable code, and evidence on the HD61700

A calculator is only the visible part

A calculator seems a modest thing to build. Enter two numbers, choose an operation, and print an answer. On an HD61700, however, even that familiar interaction asks a deeper question: where does a trustworthy number come from?

This chapter has two products: a reusable unsigned 32-bit numerical library and a native calculator that consumes it. The arithmetic should be established independently, before the finished screen is allowed to speak for it.

One implementation, many proofs, many consumers.

What unsigned 32-bit arithmetic means

The numerical domain is 0 through 4,294,967,295. Addition, subtraction, and multiplication return the low 32 bits: they wrap modulo 2^{32}. Division returns the unsigned integer quotient and remainder, with a nonzero divisor as an explicit precondition.

Operation	Illustrative result
0 − 1	4294967295
4294967295 + 1	0
4294967295 × 2	4294967294
7 / 2	3 (remainder 1)
7 % 3	1

This is not signed arithmetic disguised as a calculator. A displayed result must reflect the unsigned contract.

Build the arithmetic once

The project-facing surface includes UADD32, USUB32, UMUL32, UDIV32NZ, UDIV3216NZ, UMOD32NZ and UMOD3216NZ. These are capability names, not imposed register conventions or particular algorithms.

Historical Casio code can be recovered where its provenance and compatibility are supported. A new wrapper or composition is equally legitimate when documented honestly. In either case, the application must call the common implementation rather than hide a private arithmetic copy.

Every claimed primitive needs an independent check. A useful chain runs from source and assembler listing to exact ROM bytes, independent predicted values, observed behavior, and a conclusion no broader than the experiment.

The decimal boundary

People type decimal digits; the machine works with bounded words. UDECAPP(current, digit) appends one digit, 0–9, by computing current × 10 + digit. It must reject a result above 4294967295 without changing the old value.

The boundary can be tested without relying on accidental wraparound: floor(4294967295 / 10) = 429496729. When the current value is that number, only final digits 0 through 5 are legal.

The reverse service, UFMT32DEC, prints ordinary ASCII decimal without leading zeroes. Zero prints as “0”; the maximum prints as “4294967295”. The caller must be able to determine where the representation ends.

Make the calculator usable

The interactive form has two unsigned operands and one operator: [op1][op][op2]. It exposes addition, subtraction, multiplication, division, and remainder.

- Digits append to the active operand. Selecting an operator advances to the next field; an early replacement of the operator is allowed.
- Evaluating a complete expression makes its result the new first operand so work can continue.
- Backspace edits digits and can move back across empty fields. A fresh digit after a result begins a new expression.
- The active field is visible. A zero divisor produces an explicit error rather than calling an unchecked primitive.

The accepted calculator used both = and EXE for evaluation. Exact key mappings and screen composition remain implementation choices; the normal BRK recovery role must be preserved.

What counts as evidence

Keep the library and calculator sources, actual assembler listings, complete 64 KiB ROM images, independent numerical predictions, runtime observations, code provenance and executable identities. A failed candidate also belongs in the history; repairing it is not permission to erase the experiment that exposed it.

The student has freedom over registers, memory, algorithms, layout, wrappers and test organization. The fixed part is the observable contract and the evidence that supports it.

Historical perspective: the original Chapter 01 engineering record is preserved separately. This teaching chapter explains the accepted problem and its method; it does not replace that record.

A question to take with you

How would you know that your arithmetic library is correct if the calculator display had never been written? The answer to that question is the heart of the chapter.

Implementation update | November 2026

The current Chapter 01 ROM preserves the unsigned 32-bit arithmetic and calculator interaction contracts while consuming pinned, versioned lib2 components for arithmetic, keyboard, LCD, splash and timer services. Calculator editing, errors and display policy remain application-owned.

The underscore cursor has the same appearance; its blink is now paced by shared TM sampling rather than keyboard-poll counts. The modernized source and ROM have passed source-closure, listing and hash audits. Those build checks do not replace independent arithmetic or behavioral evidence.