LANGUAGE INDEPENDENT STANDARDS

Craig Schaffert
Digital Equipment Corporation
Cambridge Research Lab

Language Independent Work in SC22/WG11

Language Independent Arithmetic

Language Independent Datatypes

Language Independent Procedure Calling

Language Independent Arithmetic

Family of 3 standards

- Part 1: Integer and floating point arithmetic
- Part 2: Elementary numerical functions
- Part 3: Complex arithmetic and functions

Part 1 is an international standard

Part 2 is up for CD registration

Part 3 is just starting

LIA-1 Goals

Primarily a documentation standard

- covers IEEE 754 and 854 systems
- covers "almost IEEE" systems
- covers traditional mini and mainframe architectures

Specifications strong enough for serious numerical analysis

Programs can be written to

- test suitability of platform
- adapt to platform variations

LIA-1 Features

Defines the value set in terms of parameters

- maxint, minint, ...
- radix, precision, max-exponent, min-exponent, ...

Provides accuracy parameters

- epsilon, round-error, round-style, iec-559, ...

Requires the usual operations plus a few others

- +, -, *, /, abs, mod, sign, exponent, fraction, succ, pred, ...

Stipulates minimum requirements on error detection

- all numeric errors are detected and reported to the program
- the program can compensate for errors and continue

Requires "complete" documentation from vendors

Integer Details

Integer parameters

- INT_MAX
- INT_MIN
- INT_MODULO †
- bounded

Integer operations

- + - * / %
- == != < <= > >=
- abs
- modulo †
- sgn †

Floating Point Details

Floating point parameters

- FLT_RADIX
- FLT_MANT_DIG
- FLT_MAX_EXP
- FLT_MIN_EXP
- FLT_DENORM †
- FLT_IEC_559 †
- FLT_MAX
- FLT_MIN
- FLT_TRUE_MIN †
- FLT_EPSILON
- FLT_RND_ERR †
- FLT_ROUNDS

Floating point operations

- + - * /
- == != < <= > >=
- fabs †
- fsgn †
- expon †
- fract †
- scale †
- succ †
- pred †
- ulp †
- truncto †
- roundto †
- intprt †
- frcprt †

Additional Operations

Conversion operations

- between any two integer or floating point types
- conversions to float must round to nearest

IEEE Binding

For each IEC 559 conforming type, syntax for

- positive infinity
- at least one quiet NaN
- at least one signalling NaN
- remainder, square-root, and round-to-integral-value
- conversions between the floating point values and decimal strings (LIA-2)
- the "unordered" comparison, and (possibly) 19 other comparison operations (optional in IEC 559)
- The "recommended functions" copysign, negate, scaleb, logb, nextafter, finite, isnan, <>, and class (optional in IEC 559)

Ability to read/write

- the rounding mode.
- the five exception flags: inexact, underflow, (floating_)overflow, divide_by_zero, and invalid
- the disable/enable flags for each of the five exceptions (optional in IEC 559)
- the handlers for each of the exceptions (optional in IEC 559)

Boolean parameters for each implementor choice

- whether trapping is implemented
- whether tininess is detected "before rounding" or "after rounding"
- whether loss-of-accuracy is detected as a denormalization loss or as an inexact result

Notification

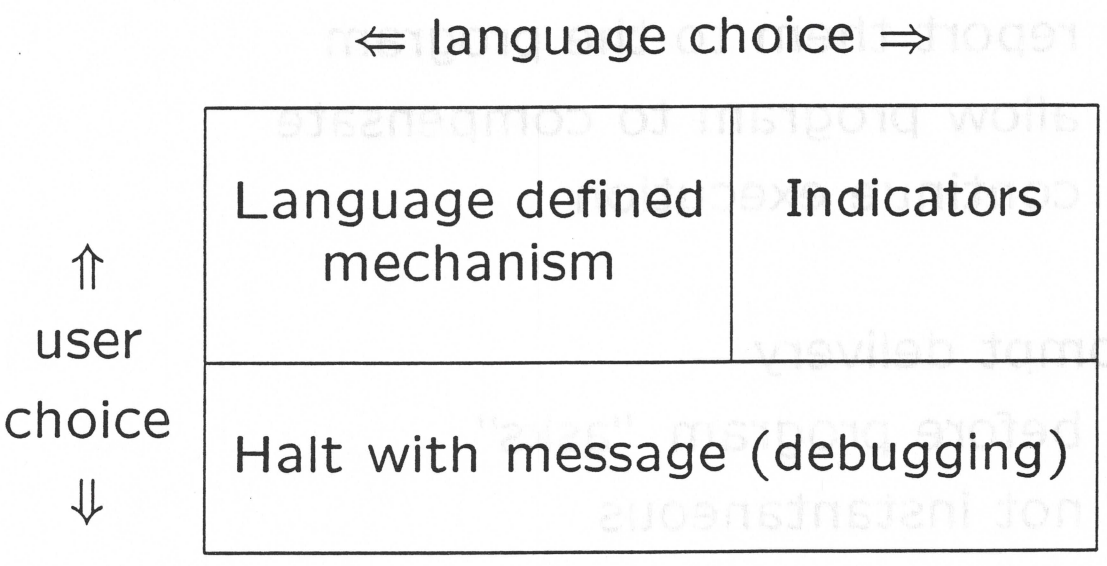
Requirements

- detect occurence of arithmetic errors
- report them to the program
- allow program to compensate
- continue execution

Prompt delivery

- before program "asks"
- not instantaneous

Notification Choices



Notification by Indicators

"Flags" for the following exceptions

- INT_OVERFLOW
- FLT_OVERFLOW
- UNDERFLOW
- UNDEFINED

A representation for sets of exceptions

- INT_OVERFLOW + UNDEFINED

Notification operations

- set_indicators †
- clear_indicators †
- test_indicators †
- current_indicators †

Documentation

Conforming types *

Parameter values

Implementation choices

- rounding, underflow detection, ...

Syntax of parameters and operations *

Notification mechanism *

Translation of expressions (*)

- some implementation freedom desirable
- implicit type conversions, reordering, short-circuit evaluation, code motion, ...
- programmer-control of order of evaluation, intermediate precision

Proposed Work

Require LIA-1 conformity

- defines arithmetic semantics without limiting implementations

Adopt standard binding for parameters and operations

- LIA-1 and IEEE

Adopt standard notification mechanism

Consider expression control

LIA-2

Part 2: Elementary numerical functions

Sample operations

- sqrt, exp, power, log
- sin, cos, tan, ...
- arcsin, arccos, arctan, ...
- sinh, cosh, tanh, ...
- arcsinh, arccosh, arctanh, ...
- conversions (nearest, truncate, ...)
- "I/O" conversions
- integer mulhi, mullo
- float addlo, sublo, mullo, rem_div, rem_sqrt
- max, min, hypot, gcd, lcm, ...

Crucial decisions

- exceptions
- accuracy
- extensions for IEEE -0, $\pm\infty$, NaN

Eventually will need language bindings

Participate! Help us get this right!

LIA-3

Part 3: Complex floating point arithmetic and elementary numerical functions

Work is just starting

Crucial decisions

- exceptions
- accuracy
- extensions for IEEE -0, $\pm\infty$, NaN
- cartesian vs polar form

Eventually will need language bindings

Participate! Help us get this right!