Pascal Validation Suite Report

Pascal processor identification

The ACK-Pascal compiler produces code for an EM machine as defined in [1]. It is up to the implementor of the EM machine whether errors like integer overflow, undefined operand and range bound error are recognized or not. Therefore it depends on the EM machine implementation whether these errors are recognized in Pascal programs or not. The validation suite results of all known implementations are given.

There does not (yet) exist a hardware EM machine. Therefore, EM programs must be interpreted, or translated into instructions for a target machine. The following implementations currently exist:

Implementation 1: an interpreter running on a PDP-11 (using UNIX). The normal mode of operation for this interpreter is to check for undefined integers, overflow, range errors etc.

Implementation 2: a translator into PDP-11 instructions (using UNIX). Less checks are performed than in the interpreter, because the translator is intended to speed up the execution of well-debugged programs.

Test Conditions

Tester: E.G. Keizer
Date: October 1983
Validation Suite version: 3.0

The final test run is made with a slightly modified validation suite.

Erroneous programs

Some test did not conform to the standard proposal of February 1979. It is this version of the standard proposal that is used by the authors of the validation suite.

Test 6.6.3.7-4:
The semicolon between high and integer on line 17 is replaced by a colon.

Test 6.7.2.2-13:
The div operator on line 14 replaced by mod.

Conformance tests

Number of tests passed = 150
Number of tests failed = 6

Details of failed tests

Test 6.1.2-1:
Character sequences starting with the 8 characters 'procedur' or 'function' are erroneously classified as the word-symbols 'procedure' and 'function'.

Test 6.1.3-2:
Identifiers identical in the first eight characters, but differing in ninth or higher numbered characters are treated as identical.

Test 6.5.1-1:
ACK-Pascal requires all formal program parameters to be declared with type *file*.

Test 6.6.6.5-1:
Gives run-time error eof seen at call to eoln. A have a hunch that this is a error in the suit.

Test 6.6.4.1-1:
Redefining the names of some standard procedures leads to incorrect behaviour of the runtime system. In this case it crashes without a sensible error message.

Test 6.9.3.5.1-1:
This test can not be translated by our compiler because two non-identical variables are used in the same block with the same first eight characters. The test passed after replacement of one of those names.

Deviance tests

Number of deviations correctly detected = 120
Number of tests not detecting deviations = 20

Details of deviations

The following tests are compiled without a proper error indication although they do not conform to the standard.

Test 6.1.6-5:
ACK-Pascal allows labels in the range 0..32767. A warning is produced when testing for deviations from the standard.

Test 6.1.8-5:
A missing space between a number and a word symbol is not detected.

Test 6.2.2-8:
Test 6.3-6:
Test 6.4.1-3:
Test 6.6.1-3:
Test 6.6.1-4:
Undetected scope error. The scope of an identifier should start at the beginning of the block in which it is declared. In the ACK-Pascal compiler the scope starts just after the declaration, however.

Test 6.4.3.3-7:
The values of fields from one variant are accessible from another variant. The correlation is exact.

Test 6.6.3.3-4:
The passing as a variable parameter of the selector of a variant part is not detected. A runtime error is produced because the variant selector is not initialized.

Test 6.8.2.4-2:
Test 6.8.2.4-3:
Test 6.8.2.4-4:
Test 6.8.2.4-5:
Test 6.8.2.4-6:
The ACK-Pascal compiler does not restrict the places from where a jump to a label by means of a goto-statement is allowed.

Test 6.8.3.9-5:
Test 6.8.3.9-6:
Test 6.8.3.9-7:
Test 6.8.3.9-16:

There are no errors produced for assignments to a variable in use as control-variable of a for-statement.

Test 6.8.3.9-8:
Test 6.8.3.9-9:

Use of a controlled variable after leaving the loop without intervening initialization is not detected.

Error handling

The results depend on the EM implementation.

Number of errors correctly detected =
Implementation 1: 32
Implementation 2: 17
Number of errors not detected =
Implementation 1: 21
Implementation 2: 36
Number of errors incorrectly detected =
Implementation 1: 2
Implementation 2: 2

Details of errors not detected

The following test fails because the ACK-Pascal compiler only generates a warning that does not prevent to run the tests.

Test 6.6.2-8:

A warning is produced if there is no assignment to a function-identifier.

With this test the ACK-Pascal compiler issues an error message for a legal construct not directly related to the error to be detected.

Test 6.5.5-2:

Program does not compile. Buffer variable of text file is not allowed as variable parameter.

The following errors are not detected at all.

Test 6.2.1-11:

Implementation 2: The use of an undefined integer is not caught as an error.

Test 6.4.3.3-10:
Test 6.4.3.3-11:
Test 6.4.3.3-12:
Test 6.4.3.3-13:

The notion of ’current variant’ is not implemented, not even if a tagfield is present.

Test 6.4.5-15:
Test 6.4.6-9:
Test 6.4.6-10:
Test 6.4.6-11:
Test 6.5.3.2-2:

Implementation 2: Subrange bounds are not checked.

Test 6.4.6-12:
Test 6.4.6-13:
Test 6.7.2.4-4:

If the base-type of a set is a subrange, then the set elements are not checked against the bounds of the subrange. Only the host-type of this subrange-type is relevant for ACK-Pascal.

Test 6.5.4-1:

Implementation 2: Nil pointers are not detected.

Test 6.5.4-2:

Implementation 2: Undefined pointers are not detected.

Test 6.5.5-3:

Changing the file position while the window is in use as actual variable parameter or as an element of the record variable list of a with-statement is not detected.

Test 6.6.2-9:

An undefined function result is not detected, because it is never used in an expression.

Test 6.6.5.3-6:
Test 6.6.5.3-7:

Disposing a variable while it is in use as actual variable parameter or as an element of the record variable list of a with-statement is not detected.

Test 6.6.5.3-8:
Test 6.6.5.3-9:
Test 6.6.5.3-10:

It is not detected that a record variable, created with the variant form of new, is used as an operand in an expression or as the variable in an assignment or as an actual value parameter.

Test 6.6.5.3-11:

Use of a variable that is not reinitialized after a dispose is not detected.

Test 6.6.6.4-4:
Test 6.6.6.4-5:
Test 6.6.6.4-7:

Implementation 2: There are no range checks for pred, succ and chr.

Test 6.6.6.5-6:
ACK-Pascal considers a rewrite of a file as a defining occurence.

Test 6.7.2.2-8:
Test 6.7.2.2-9:
Test 6.7.2.2-10:
Test 6.7.2.2-12:
Implementation 2: Division by 0 or integer overflow is not detected.

Test 6.8.3.9-18:
The use of the some control variable in two nested for statements in not detected.

Test 6.8.3.9-19:
Access of a control variable after leaving the loop results in the final-value, although an error should be produced.

Test 6.9.3.2-3:
The program stops with a file not open error. The rewrite before the write is missing in the program.

Test 6.9.3.2-4:
Test 6.9.3.2-5:
Illegal FracDigits values are not detected.

Implementation dependence

Number of tests run = 14
Number of tests incorrectly handled = 0

Details of implementation dependence

Test 6.1.9-5:
Alternate comment delimiters are implemented

Test 6.1.9-6:
The equivalent symbols @ for ^, (. for [and .) for] are not implemented.

Test 6.4.2.2-10:
Maxint = 32767

Test 6.4.3.4-5:
Only elements with non-negative ordinal value are allowed in sets.

Test 6.6.6.1-1:

Standard procedures and functions are not allowed as parameters.

Test 6.6.6.2-11:

Details of the machine characteristics regarding real numbers:

```
beta =      2
t =        56
rnd =       1
ngrd =      0
machep =   -56
negep =   -56
iexp =      8
minexp =  -128
maxexp =   127
eps =     1.387779e-17
epsneg =  1.387779e-17
xmin =    2.938736e-39
xmax =    1.701412e+38
```

Test 6.7.2.3-3:
Test 6.7.2.3-4:

All operands of boolean expressions are evaluated.

Test 6.8.2.2-1:
Test 6.8.2.2-2:

The expression in an assignment statement is evaluated before the variable selection if this involves pointer dereferencing or array indexing.

Test 6.8.2.3-2:

Actual parameters are evaluated in reverse order.

Test 6.9.3.2-6:

The default width for integer, Boolean and real are 6, 5 and 13.

Test 6.9.3.5.1-2:

The number of digits written in an exponent is 2.

Test 6.9.3.6-1:

The representations of true and false are (true) and (false). The parenthesis serve to indicate width.

Quality measurement

Number of tests run = 60
Number of tests handled incorrectly = 1

Results of tests

Several test perform operations on reals on indicate the error introduced by these operations. For each of these tests the following two quality measures are extracted:

maxRE: maximum relative error
rmsRE: root-mean-square relative error

Test 1.2-1:

Implementation 1: 25 thousand Whetstone instructions per second.
Implementation 2: 169 thousand Whetstone instructions per second.

Test 1.2-2:

The value of (TRUEACC-ACC)*2^56/100000 is 1.4 . This is well within the bounds specified in [3].
The GAMM measure is:
Implementation 1: 238 microseconds
Implementation 2: 26.3 microseconds.

Test 1.2-3:

The number of procedure calls calculated in this test exceeds the maximum integer value. The program stops indicating overflow.

Test 6.1.3-3:

The number of significant characters for identifiers is 8.

Test 6.1.5-8:

There is no maximum to the line length.

Test 6.1.5-9:

The error message "too many digits" is given for numbers larger than maxint.

Test 6.1.5-10:
Test 6.1.5-11:
Test 6.1.5-12:

Normal values are allowed for real constants and variables.

Test 6.1.7-14:

A reasonably large number of strings is allowed.

Test 6.1.8-6:
No warning is given for possibly unclosed comments.

Test 6.2.1-12:
Test 6.2.1-13:
Test 6.2.1-14:
Test 6.2.1-15:
Test 6.5.1-2:
Large lists of declarations are possible in each block.

Test 6.4.3.2-6:
An ’array[integer] of’ is not allowed.

Test 6.4.3.2-7:
Test 6.4.3.2-8:
Large values are allowed for arrays and indices.

Test 6.4.3.3-14:
Large amounts of case-constant values are allowed in variants.

Test 6.4.3.3-15:
Large amounts of record sections can appear in the fixed part of a record.

Test 6.4.3.3-16:
Large amounts of variants are allowed in a record.
Test 6.4.3.4-4:
Size and speed of Warshall’s algorithm depend on the implementation of EM:

Implementation 1:
size: 122 bytes
speed: 5.2 seconds

Implementation 2:
size: 196 bytes
speed: 0.7 seconds

Test 6.5.3.2-3:
Deep nesting of array indices is allowed.

Test 6.5.3.2-4:
Test 6.5.3.2-5:
Arrays can have at least 8 dimensions.

Test 6.6.1-8:
Deep static nesting of procedure is allowed.

Test 6.6.3.1-6:
Large amounts of formal parameters are allowed.

Test 6.6.5.3-12:

Dispose is fully implemented.

Test 6.6.6.2-6:

Test sqrt(x): no errors. The error is within acceptable bounds.

maxRE: 2 ** -55.50
rmsRE: 2 ** -57.53

Test 6.6.6.2-7:

Test arctan(x): may cause underflow or overflow errors. The error is within acceptable bounds.

maxRE: 2 ** -55.00
rmsRE: 2 ** -56.36

Test 6.6.6.2-8:

Test exp(x): may cause underflow or overflow errors. The error is not within acceptable bounds.

maxRE: 2 ** -50.03
rmsRE: 2 ** -51.03

Test 6.6.6.2-9:

Test sin(x): may cause underflow errors. The error is not within acceptable bounds.

maxRE: 2 ** -38.20
rmsRE: 2 ** -43.68

Test cos(x): may cause underflow errors. The error is not within acceptable bounds.

maxRE: 2 ** -41.33
rmsRE: 2 ** -46.62

Test 6.6.6.2-10:

Test ln(x): The error is not within acceptable bounds.

maxRE: 2 ** -54.05
rmsRE: 2 ** -55.77

Test 6.7.1-3:
Test 6.7.1-4:
Test 6.7.1-5:

Complex nested expressions are allowed.

Test 6.7.2.2-14:

Test real division: The error is within acceptable bounds.

maxRE: 0
rmsRE: 0

Test 6.7.2.2-15:

Operations of reals in the integer range are exact.

Test 6.7.3-1:
Test 6.8.3.2-1:
Test 6.8.3.4-2:

Test 6.8.3.5-15:
Test 6.8.3.7-4:
Test 6.8.3.8-3:
Test 6.8.3.9-20:
Test 6.8.3.10-7:

Static deep nesting of function calls, compound statements, if statements, case statements, repeat loops, while loops, for loops and with statements is possible.

Test 6.8.3.2-2:

Large amounts of statements are allowed in a compound statement.

Test 6.8.3.5-12:

The compiler requires case constants to be compatible with the case selector.

Test 6.8.3.5-13:
Test 6.8.3.5-14:

Large case statements are possible.

Test 6.9-2:

Recursive IO on the same file is well-behaved.

Test 6.9.1-6:

The reading of real values from a text file is done with sufficient accuracy.

maxRE: 2 ** -54.61
rmsRE: 2 ** -56.32

Test 6.9.1-7:
Test 6.9.2-2:
Test 6.9.3-3:
Test 6.9.4-2:

Read, readln, write and writeln may have large amounts of parameters.

Test 6.9.1-8:

The loss of precision for reals written on a text file and read back is:

maxRE: 2 ** -53.95
rmsRE: 2 ** -55.90

Test 6.9.3-2:

File IO buffers without trailing marker are correctly flushed.

Test 6.9.3.5.2-2:

Reals are written with sufficient accuracy.

maxRE: 0
rmsRE: 0

Level 1 conformance tests

Number of test passed = 4
Number of tests failed = 1

Details of failed tests

Test 6.6.3.7-4:
An expression indicated by parenthesis whose value is a conformant array is not allowed.

Level 1 deviance tests

Number of deviations correctly detected = 4
Number of tests not detecting deviations = 0

Level 1 error handling

The results depend on the EM implementation.

Number of errors correctly detected =
Implementation 1: 1
Implementation 2: 0
Number of errors not detected =
Implementation 1: 0
Implementation 2: 1

Details of errors not detected

Test 6.6.3.7-9:
Implementation 2: Subrange bounds are not checked.

Level 1 quality measurement

Number of tests run = 1

Results of test

Test 6.6.3.7-10:
Large conformant arrays are allowed.

Extensions

Number of tests run = 3

Details

Test 6.1.9-7:
The alternative relational operators are not allowed.

Test 6.1.9-8:
The alternative symbols for colon, semicolon and assignment are not allowed.

Test 6.8.3.5-16:
The otherwise selector in case statements is not allowed.

References

[1] A.S.Tanenbaum, E.G.Keizer, J.W.Stevenson, Hans van Staveren, "Description of a machine architecture for use with block structured languages", Informatica rapport IR-81.

[2] ISO standard proposal ISO/TC97/SC5-N462, dated February 1979. The same proposal, in slightly modified form, can be found in: A.M.Addyman e.a., "A draft description of Pascal", Software, practice and experience, May 1979. An improved version, received March 1980, is followed as much as possible for the current ACK-Pascal.

[3] B. A. Wichman and J du Croz, A program to calculate the GAMM measure, Computer Journal, November 1979.