

Content

1. Table of contents
2. Three bit supersymmetry theory
3. **N3Lang** Basics
4. Additions to the **N3Lang** language
5. Language Representation
6. Controlled exchange and unconditional permutation
7. Copyright

Three bit supersymmetry theory

| | | | | |
|---|---|---|----------------|--------------|
| 0 | 0 | 0 | Particle 1 | Symmetrical |
| 1 | 1 | 1 | Antiparticle 1 | Symmetrical |
| 0 | 1 | 0 | Particle 2 | Symmetrical |
| 1 | 0 | 1 | Antiparticle 2 | Symmetrical |
| 1 | 0 | 0 | Particle 3 | Asymmetrical |
| 0 | 1 | 1 | Antiparticle 3 | Asymmetrical |
| 0 | 0 | 1 | Particle 4 | Asymmetrical |
| 1 | 1 | 0 | Antiparticle 4 | Asymmetrical |

Supersymmetry in physics

Supersymmetry, or Fermi-Bose symmetry, is a hypothetical symmetry connecting bosons and fermions in nature. The abstract supersymmetry transformation links bosonic and fermionic quantum fields so that they can transform into each other. Figuratively, we can say that the supersymmetry transformation can transform matter into interaction (or into radiation), and vice versa.

Supersymmetry involves doubling (at least) the number of known elementary particles due to the presence of superpartners. For a photon - photino, quark - squark, Higgs - higgsino, W-boson - wine, gluon - gluino, and so on. Superpartners must have a spin value that is half an integer different from the spin value of the original particle

N3Lang Basics


| | | | |
|---------|---------|-----|--|
| N P | | | One- and two-bit negation (Not), 0 and 1 – control bits, P – bit remains unchanged (Pass) |
| N 0 1 P | N 0 1 P | | |
| S | S | P | Unconditional exchange of two bits (Swap), the third bit of P remains unchanged (Pass) |
| S | P | S | |
| P | S | S | |
| 0 1 | S | S | A control bit is supplied to one of the inputs; if the bit is zero or one, then the other two, designated as S, are swapped (Control Swap and Anti-Control Swap) |
| S | 0 1 | S | |
| S | S | 0 1 | |
| P | P | P | Does not perform any actions, incoming bits are sent to the output without changes (Pass) |
| N P | N P | N P | The unconditional negation of one, two or three bits (Not) |
| 0 1 | N P | N P | Conditional negation (Not depending on the value of one or two control bits). A vertical bar indicates alternative values, and a P means the bit does not change (Pass). |
| N P | 0 1 | N P | |
| N P | N P | 0 1 | |
| 0 1 | 0 1 | N | |
| 0 1 | N | 0 1 | |
| N | 0 1 | 0 1 | |

Additions to the N3Lang language

| | | | | |
|---|---|-----|-----|--|
| D | D | D | D | For step-by-step debugging, D (Debug) will be replaced with the current bit value. The character D can be specified any number of times |
| E | E | N S | N S | The E (Equal) character appears two or more times. If all bits of E are equal, an action is performed in the remaining bits: negation of N (Not) or exchange of two bits of S (Swap) |
| M | L | N S | N S | If the M (More) bit is greater than the L (Less) bit, an action is performed in the remaining bits: negation of N (Not) or exchange of two S bits (Swap) |

The N3Lang language is used to transform binary input data, since all operations are reversible, the length of the input is equal to the length of the output, so the language can be presented in tabular form. In addition, all the above operations are easily scalable, specifying, for example, many equality conditions using the symbol E (Equal) or many negation operations N (Not). It is also possible to specify multiple control bits as 0 or 1.

Representation of the language in tabular form

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| > | S | . | D | . | E | > | The program runs from top to bottom (priority) and from left to right. Data comes in from the left along the first vertical, output is the last vertical. |
| > | P | S | D | . | S | > | |
| > | S | 1 | D | L | E | > | |
| > | . | S | D | M | S | > | |
| > | . | . | D | S | N | > | > at the beginning of the line - input, > at the end of the line - output, dot – empty space (does not perform any action) |
| > | . | . | D | S | . | > | |
| # | . | . | . | . | . | > | A one-line comment is specified by the hash symbol # (only the first character in the line) |
| > | N | . | N | . | . | > | Single-bit operations |
| * | . | N | . | . | . | > | A multi-line comment is specified by the * symbol, only the first characters |
| > | . | N | . | . | . | > | |
| > | . | P | . | . | . | > | |
| * | . | . | . | . | . | > | |
| > | . | . | . | . | . |  | The last line of the program contains an arrow symbol ↖ at the end of the line, which can be pulled while holding the left mouse button, used to change the size of the input-output and the width of the program |

Equivalent text representation

| | |
|--|---|
| 6:#, 8:*, 11:*; 0:S + 1:P + 2:S, 7:N; 1:S + 1 + S, 8:N + 9:N + 10:P; 0:D:6 + 7:N; 2:L + 3:M + 4:S:2; 0:E + 1:S + 2:E + 3:S + 4:N; | The difference between a text view and a table view is that the program is executed from left to right (priority) and from top to bottom, and insignificant information, for example, empty space, is not indicated |
|--|---|

Example No. 1. Controlled exchange and unconditional permutation

| The input has three bits, each can have the value 0 or 1 | | The values here 0, 1 are control ones, and S values on the same vertical are swapped if the control bit is 0 or 1 | | | | | | The output is three bits, each can have the value 0 or 1 | |
|--|---|---|---|---|---|---|---|--|--------|
| {0, 1} | ➡ | 0 | 1 | S | S | S | S | ➡ | {0, 1} |
| {0, 1} | ➡ | S | S | 0 | 1 | S | S | ➡ | {0, 1} |
| {0, 1} | ➡ | S | S | S | S | 0 | 1 | ➡ | {0, 1} |

The transformation above is equivalent to swapping the first and last inputs

| | | |
|----|---|----|
| x1 | ➡ | x3 |
| x2 | ➡ | x2 |
| x3 | ➡ | x1 |

It is obvious that these transformations are reversible, since in the first case we used a reversible logic gate Control Swap and its opposite Anti-Control Swap, in the second case the exchange of the first and last inputs is unconditional, that is, it is also reversible.

Copyright

This is free and unencumbered software released into the public domain. Anyone is free to copy, modify, publish, use, compile, sell, or distribute this software, either in source code form or as a compiled binary, for any purpose, commercial or non-commercial, and by any means.

In jurisdictions that recognize copyright laws, the author or authors of this software dedicate any and all copyright interest in the software to the public domain. We make this dedication for the benefit of the public at large and to the detriment of our heirs and successors. We intend this dedication to be an overt act of relinquishment in perpetuity of all present and future rights to this software under copyright law.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

For more information, please refer to [<https://unlicense.org>](https://unlicense.org)

N3Lang is an experimental language based on three-bit supersymmetry; basic language operations have three inputs and three outputs. A program consists of a finite sequence of such instructions.

Author: Belyanin Aleksey

Login: xayam

E-Mail: xayam@yandex.ru

Project N3Lang on GitHub

<https://github.com/xayam/N3Lang>