The 16 Boolean Connectives of Four Valued Bit Code (4VBC)

Keywords

4VBC, AND, Boolean connectives, Complementation, Conjunction, Contradiction, Converse, Dextro, Disjunction, EQV, False, four valued bit code, future, IF, IMP, Implication, LC, lens, logical operator, lookup table, LP, lune, minterm, multi-valued logic, NAND, NIF, NIMP, Nonimplication, NOR, NOT, NTAU, OR, past, present, Projection, RC, RP, Sinistro, TAU, Tautology, temporal, tense, True, universe of discourse, Venn polyominos, XNOR, XOR

Abstract

The four common Boolean connectives are AND, IMP, OR, and XOR. The NOT negation of these is NAND, NIMP, NOR and EQV, also known as XNOR. At the binary, bit level there are four rediscovered connectives and the four respective negations. These arise from using a multi-valued logic named four valued bit code (4VBC) to study the inequality in temporal logic of Past < Present < Future. It is shown that: Past in terms of Future is false; Past in terms of Present is false; Present in terms of Future is true; Present in terms of Present is a tautology; Present in terms of Past is true, Future in terms of Present is a tautology; and Future in terms of Past is a tautology. Connectives are presented as Venn polyominos and defined by the connectives NOR, NAND, AND, and OR.

Background

The four commonly used logical connectives at the bit level are AND, IMP, OR, and XOR. The two most commonly used negations are NAND for NOT(AND) and NOR for NOT(OR). The four commonly used connectives were developed by Charles S Peirce (1880) using NOR and Henry M Sheffer (1913) using NAND. The value and definitions of four valued bit code (4VBC) are the bit pairs of 00 contradiction, 01 true, 10 false, and 00 tautology. The bit pairs have a left or Sinistro and False side, and a right or Dextro and True side with meanings in Table 1.

Bit pairs	Left, Sinistro False Side	Right, Dextro True Side	Meaning
01	NOT(False)	True	True, NOT(Void)
10	False	NOT(True)	False, Void
00	NOT(False)	NOT(True)	Contradiction, NOT(Exist)
11	False	True	Tautology, Exist

Table 1. Bit pair meanings

Mapping temporal logic with 4VBC

The parts of the time continuum are evaluated as an inequality using 4VBC with the tense of the time pieces assigned to bit pairs in Table 2. Past is defined as false because it has transpired and is no longer true as Present. Future is defined as true or false, a tautology, because it is necessarily undetermined. No tense is assigned to both true and false at the same time, a contradiction. There is also no number line associated here because the number zero is the absence of a variable or lack of a proposition, neither case of which exists in Table 2.

10	<	01	<	11
Past	<	Present	<	Future

Table 2. Bit pairs for tenses of time

Three relations are deduced around the unitary fraction of tense / tense as unity in Table 3. From Table 3, relations 2 and 3 are combined into relation 4 in Table 4. Relation 1 is interleaved into relation 4 with the systematic substitution of N-1, N, and N+1 for Past, Present, and Future where N is the largest known counting number in relation 5 in Table 5. The groups in relation 5 are replaced by letters, the division symbol is removed, and the tenses are substituted with bit pairs from 4VBC to make relation 6 in Table 6, where D is Unity.

1.	Past / Present	<	Unity	<	Future / Present
2.	Unity	<	Present / Past	<	Future / Past
3.	Past / Future	<	Present / Future	<	Unity

Table 3. Inequality of tense based on unity

4.	Past/Future	<	Present/Future	<	Unity	<	Present/Past	<	Future/ Past

Table 4. Relations 2 and 3 combined into relation 4

5.	Past / Future	<	Past / Present	<	Present / Future	<	Unity	<	Present / Past	<	Future / Present	<	Future / Past

Table 5. Relations 1, 2, 3, and 4 combined into the inequality of relation 5

6.	A	<	B	<	C	<	D	<	E	<	F	<	G
	10		10		01		01		01		11		11
	11		01		11		01		10		01		10

Table 6. 4VBC for tense from Table 5

The logical connective IF and its negation NIF

The bit definition of p is 0011, and the bit definition of q is 0101. From these, all possible bit-wise operations are denoted for any Boolean connective. For example, the AND connective is given by p AND q, and the EQV connective is given by p EQV q, both in Table 7.

	0011		0011
AND	0101	EQV	0101
=	0001	=	1001

Table 7. The logical bit connectives AND and EQV

The statements of (p AND q) EQV (q) and (p IF q) are shown in Table 8 to be identical.

	0011		0011
AND	0101	IF	0101
=	0001		:
EQV	0101		:
=	1011	=	1011

Table 8. (p IF q) shown identical to (p AND q) EQV (q)

The logical connective IF is defined by AND and EQV in Table 8. The connective IF is added to relation 9 of Table 6 in Table 9, where IF defines a temporal logic from Table 8.

9.	A	<	B	<	C	<	D	<	E	<	F	<	G
	10		10		01		01		01		11		11
IF	11		01		11		01		10		01		10
=	10		10		01		11		01		11		11

Table 9. 4VBC for tense defined by IF

Table 9 presents new results about how the tenses of time interrelate. Past in terms of Future (A: 10 IF 11 = 10) is false because the past is transpired and is no longer true as is Present. Past in terms of Present (B: 10 IF 01 = 10) is false for the same reason. Present in terms of Future (C: 01 IF 11 = 01) is true because the Present is true regardless of whether the Future is true or false. Present in terms of itself (D: 01 IF 01 = 11) is a tautology as being either a true or false. Present in terms of Past (E: 01 IF 10 = 01) is true because the Present is still true while the Past is always false. Future in terms of Present (F: 11 IF 01 = 11), and Future in terms of Past (G: 11 IF 10 = 11) are both either true or false as tautology. The negation of IF as NOT(IF) is named NIF and appears with IF in Table 10.

The Boolean connectives of LC, TAU, RC and respective negations of LP, NTAU, RP

Three more Boolean connectives were rediscovered, after Philo of Megara, ca 300 BC, and redefined below in Table 10 along with IF and NIF. The abbreviations follow the naming conventions of Donald E Knuth, *The Art of Computer Programming*, VF 4: 0, pp 48-9.

Logical definition with p = 0011, q = 0101	Connective name, Acronym, Bit-wise operation	Negation NOT( )
(p AND q) EQV q	Converse Implication 0011 IF 0101 = 1011	Converse Nonimplication 0011 NIF 0101 = 0100
(p EQV q) OR q	Left Complementation 0011 LC 0101 = 1100	Left Projection 0011 LP 0101 = 0011
(p AND q) IMP q	Tautology 0011 TAU 0101 = 1111	Contradiction 0011 NTAU 0101 = 0000
(p AND q) NOR q	Right Complementation 0011 RC 0101 = 1010	Right Projection 0011 RP 0101 = 0101

Table 10. Definitions of IF, LC, TAU, RC, and
the respective negations of NIF, LP, NTAU, RP

Derivation of all Boolean connectives from NOR

The definitions of the connectives in Table 10 may be reduced to the NOR connective in Table 11. This was developed independently of and subsequently verified by Donald E Knuth, *The Art of Computer Programming*, VFSa 4: 0: 7.1.1: 4(a), p 154 from Benjamin A Bernstein (1913).

NOT(p) = p NOR p
NOT(q) = q NOR q

p IF q ≡ ((p AND q) OR q) ≡ ((((NOT(p)) NOR (NOT(q)))) OR q)
≡ (NOT(((NOT(p)) NOR (NOT(q))) NOR q))

p NIF q ≡ (NOT(p AND q) OR q) ≡ (((NOT(p)) NOR (NOT(q))) NOR q))

p RC q ≡ ((p EQV q) OR q)
≡ (((NOT((p NOR q) NOR ((NOT(p)) NOR (NOT(q)))))) OR q)
≡ (NOT((NOT(((p NOR q) NOR ((NOT(p)) NOR (NOT(q)))))) NOR q))))

p RP q ≡ (NOT((p EQV q) OR q))
≡ ((NOT(((p NOR q) NOR ((NOT(p)) NOR (NOT(q)))))) NOR q)))

p TAU q ≡ ((p AND q) IMP q) ≡ (((NOT(p)) NOR (NOT(q))) IMP q)
≡ ((((NOT(p)) NOR (NOT(q)) NOR q) NOR q) NOR (((NOT(p)) NOR (NOT(q)) NOR q) NOR q))

p NTAU q ≡ (NOT(p AND q) IMP q) ≡ (NOT(((NOT(p)) NOR (NOT(q))) IMP q))
≡ (NOT((((NOT(p)) NOR (NOT(q)) NOR q) NOR q) NOR (((NOT(p)) NOR (NOT(q)) NOR q) NOR q)))

p LC q ≡ ((p AND q) NOR q) ≡ ((((NOT(p)) NOR (NOT(q)))) NOR q)

p LP q ≡ (NOT((p AND q) NOR q)) ≡ (NOT((((NOT(p)) NOR (NOT(q)))) NOR q))

Table 11. Definitions of IF, LC, TAU, RC, and the respective negations of NIF, LP, NTAU, RP in terms of NOT and NOR

In addition, the derivation of the other eight connectives is based on NOR and listed as a point of reference in Table 12.

NOT(p) = p NOR p, the definition from above
NOT(q) = q NOR q, the definition from above

p AND q ≡ ((NOT(p)) NOR (NOT(q)))

p NAND q ≡ (NOT((NOT(p)) NOR (NOT(q))))

p OR q≡ (NOT(p NOR q))

p IMP q ≡ (((p NOR p) NOR q) NOR ((p NOR p) NOR q))

p NIMP q ≡ (NOT(((p NOR p) NOR q) NOR ((p NOR p) NOR q)))

p EQV q ≡ (NOT((p NOR q) NOR ((NOT(p)) NOR (NOT(q)))))

p XOR q ≡ ((p NOR q) NOR ((NOT(p)) NOR (NOT(q))))

Table 12. Definitions of AND, OR, IMP, EQV and the respective negations of NAND, NOR, NIMP, XOR in terms of NOT and NOR

Derivation of all Boolean connectives from AND

NOT(p) = NOT(p AND p)

NOT(q) = NOT(q AND q)

p NAND q ≡ (NOT(p AND q))

p OR q ≡ (NOT((NOT(p)) AND (NOT(q))))

p NOR q ≡ ((NOT(p)) AND (NOT(q)))

p EQV q ≡ ((NOT (p AND (NOT(q)))) AND (NOT ((NOT(p)) AND q)))

p XOR q ≡ (NOT((NOT (p AND (NOT(q)))) AND (NOT ((NOT(p)) AND q))))

p IMP q ≡ (NOT(p AND(NOT(q))))

p NIMP q ≡ (p AND(NOT(q)))

p IF q ≡ (NOT((NOT(p AND q)) AND (NOT(q))))

p NIF q ≡ ((NOT(p AND q)) AND (NOT(q)))

p LC q ≡ ((NOT(p AND q)) AND (NOT(q)))

p LP q ≡ (NOT((NOT(p AND q)) AND (NOT(q))))

p RC q ≡ (NOT(((p AND (NOT(q))) AND (NOT((NOT(p)) AND (q)))) AND (NOT(q))))

p RP q ≡ (((p AND (NOT(q))) AND (NOT((NOT(p)) AND (q)))) AND (NOT(q)))

p TAU q ≡ (NOT((p AND q) AND (NOT(q))))

p NTAU q ≡ ((p AND q) AND (NOT(q)))

Table 13. Definitions of OR, EQV, IMP, IF, LC, RC, and TAU and the respective negations of NOR, XOR, NIMP, NIF, LP, RP, and NTAU in terms of NOT and AND

Derivation of some Boolean connectives from OR

NOT(p) = NOT(p OR p)
NOT(q) = NOT(q OR q)

p NAND q ≡ ((NOT(p)) OR (NOT(q)))

p AND q ≡ (NOT((NOT(p)) OR (NOT(q)))) p NOR q ≡ NOT(p OR q)

p EQV q ≡ ((NOT(p OR q)) OR (NOT(NOT(p) OR NOT(q))))

p XOR q ≡ (NOT((NOT(p OR q)) OR (NOT(NOT(p) OR NOT(q)))))

p IMP q ≡ (NOT((NOT(p)) OR q))

p NIMP q ≡ ((NOT(p)) OR q)

p IF q ≡ (p OR (NOT(q))

p NIF q ≡ (NOT(p OR (NOT(q)))

p LC q ≡

p LP q ≡

p RC q ≡

p RP q ≡

p TAU q ≡

p NTAU q ≡

Table 14. Definitions of OR, EQV, IMP, IF, LC, RC, and TAU and the respective negations of NOR, XOR, NIMP, NIF, LP, RP, and NTAU in terms of NOT and OR

Bit-wise table for the 16 Boolean connectives

The bit-wise operation of each of the 16 Boolean connectives on the propositions of p and q is listed as a point of reference in Table 15.

```
      0011 0011 p        0011 0011 p
AND   0101 0101 q   NAND 0101 0101 q
 17   0001 0001     238  1110 1110

      0011 0011 p        0011 0011 p
OR    0101 0101 q   NOR  0101 0101 q
119   0111 0111     136  1000 1000

      0011 0011 p        0011 0011 p
EQV   0101 0101 q   XOR  0101 0101 q
153   1001 1001     102  0110 0110

      0011 0011 p        0011 0011 p
RC    0101 0101 q   RP   0101 0101 q
170   1010 1010      85  0101 0101

      0011 0011 p        0011 0011 p
IF    0101 0101 q   NIF  0101 0101 q
187   1011 1011      68  0100 0100

      0011 0011 p        0011 0011 p
LC    0101 0101 q   LP   0101 0101 q
204   1100 1100      51  0011 0011

      0011 0011 p        0011 0011 p
IMP   0101 0101 q   NIMP 0101 0101 q
221   1101 1101      34  0010 0010

      0011 0011 p        0011 0011 p
TAU   0101 0101 q   NTAU 0101 0101 q
255   1111 1111       0  0000 0000
```

Table 15. Bit-wise operation of the 16 logical connectives

Lookup tables for the 16 Boolean connectives

Results of the connectives operating on the propositions of p and q are tabulated in lookup tables as row major with p as the index to the rows and q as the index to the columns. Headings for the rows and columns arrange the bits in the order of 00, 01, 10, 11 for Contradiction, True, False, Tautology with least significant bit to the right. This is presented in Table 16 below.

AND	00	01	10	11
00	00	00	00	00
01	00	01	00	01
10	00	00	10	10
11	00	01	10	11

NAND	00	01	10	11
00	11	11	11	11
01	11	10	11	10
10	11	11	01	01
11	11	10	01	00

OR	00	01	10	11
00	00	01	10	11
01	01	01	11	11
10	10	11	10	11
11	11	11	11	11

NOR	00	01	10	11
00	11	10	01	00
01	10	10	00	00
10	01	00	01	00
11	00	00	00	00

EQV	00	01	10	11
00	11	10	01	00
01	10	11	00	01
10	01	00	11	10
11	00	01	10	11

XOR	00	01	10	11
00	00	01	10	11
01	01	00	11	10
10	10	11	00	01
11	11	10	01	00

RC	00	01	10	11
00	11	10	01	00
01	11	10	01	00
10	11	10	01	00
11	11	10	01	00

RP	00	01	10	11
00	00	01	10	11
01	00	01	10	11
10	00	01	10	11
11	00	01	10	11

IF	00	01	10	11
00	11	10	01	00
01	11	11	01	01
10	11	10	11	10
11	11	11	11	11

NIF	00	01	10	11
00	00	01	10	11
01	00	00	10	10
10	00	01	00	01
11	00	00	00	00

LC	00	01	10	11
00	11	11	11	11
01	10	10	10	10
10	01	01	01	01
11	00	00	00	00

LP	00	01	10	11
00	00	00	00	00
01	01	01	01	01
10	10	10	10	10
11	11	11	11	11

IMP	00	01	10	11
00	11	11	11	11
01	10	11	10	11
10	01	01	11	11
11	00	01	10	11

NIMP	00	01	10	11
00	00	00	00	00
01	01	00	01	00
10	10	10	00	00
11	11	10	01	11

TAU	00	01	10	11
00	11	11	11	11
01	11	11	11	11
10	11	11	11	11
11	11	11	11	11

NTAU	00	01	10	11
00	00	00	00	00
01	00	00	00	00
10	00	00	00	00
11	00	00	00	00

Table 16. Lookup tables for the 16 Boolean Connectives based on 00, 01, 10, 11 with index p \ q

Venn diagrams of the 16 Boolean connectives

The 16 Boolean connectives map to Venn diagrams where bits in the template “abcd” represent “a” for the box (the universe of discourse) surrounding the inner circular regions (minterms), “b” for the right lune minterm, “c” for the left lune minterm, “d” for the center lens. The coordinates for the linear four-space over a two-element field are represented by “0” for white and “1” for black. This was shown by Steven H. Cullinane, “The Geometry of Logic: Finite Geometry and the 16 Boolean Connectives”, 2007. In Table 17 we use “Venn polyominos” (Mark Thompson, 2000) that are easier to draw.

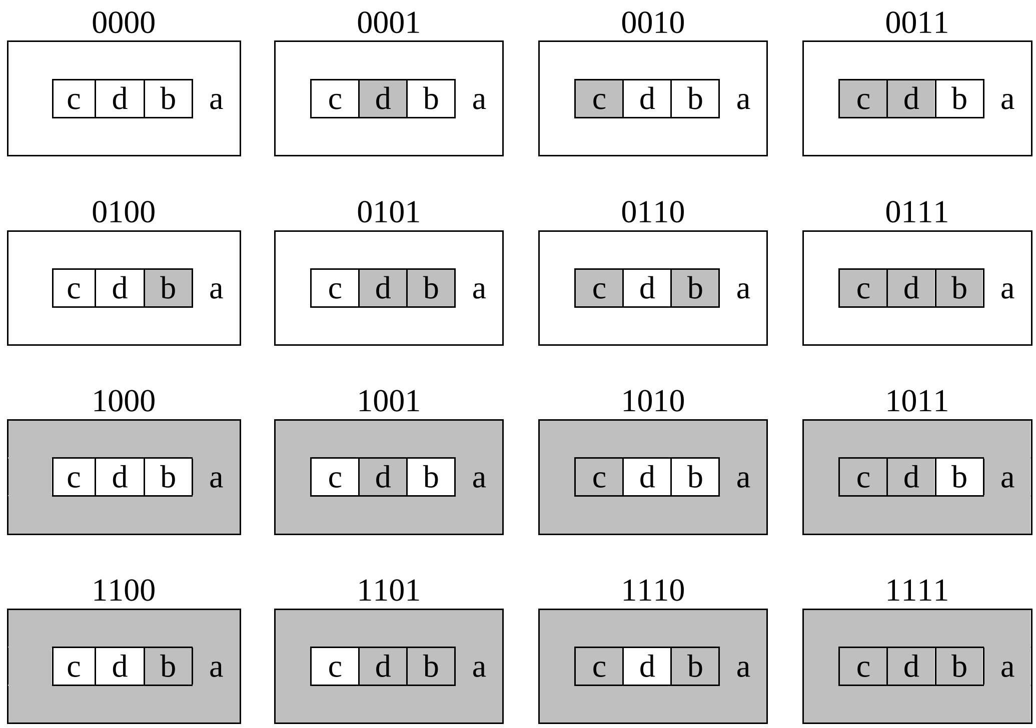

Table 17. Venn polyominos of the 16 Boolean connectives with respective binary coordinates for the linear four-space over a two-element field