IEEE Working Group P3109 Interim Report on 8-bit Binary Floating-point Formats

Questions and comments via GitHub issues at
https://github.com/P3109/Public

First public release: 18 September 2023

This version: 22 November 2023

Contents

1 Introduction

This document represents the results of discussions and decisions made by the IEEE Working Group P3109, "Standard for Arithmetic Formats for Machine Learning". The Project Authorization Request (PAR) for P3109 defines the scope, need, and stakeholders as follows:

> **Scope of proposed standard**: This standard defines a binary arithmetic and data format for machine learning-optimized domains. It also specifies the default handling of exceptions that occur in this arithmetic. This standard provides a consistent and flexible arithmetic framework optimized for Machine Learning Systems (MLS) in hardware and/or software implementations to minimize the work required to make MLS interoperable with each other, as well as other dependent systems. This standard is aligned with IEEE Std 754-2019 for Floating-Point Arithmetic.
>
> **Need for the Work**: Machine Learning Systems have different arithmetic requirements from most other domains. Precisions tend to be lower, and accuracy is measured in dimensions other than just numerical (e.g. inference accuracy). Furthermore, machine learning systems are often integrated into mission-critical and safety-critical systems. With no standards specifically addressing these needs, Machine Learning Systems are built with inconsistent expectations and assumptions that hinder the compatibility and reuse of machine learning hardware, software, and training data.
>
> **Stakeholders for the Standard**: System developers, vendors, and users of machine learning applications across many industries and interests including but not limited to computation, storage, medical, telecommunications, e-commerce, fleet management, automotive, robotics, and security.

The scope of this interim release is interchange formats only. The working group continues to deliberate on the specification of operations.

1.1 Typographical conventions and notation

Bold text describes the decisions and specifications of this document.

Text that is not bold is background material, typically providing rationale and arguments that represent discussions of the working group leading to a decision and specification.

This document specifies 8-bit floating-point interchange formats (binary formats) and associated operations. Binary formats are parameterized by their width, the number of bits spanned in memory (here, 8); and their precision (p), the number of bits spanned by the true significand (this is one more than the bits of the significand that are stored explicitly).

The formats defined herein shall be referred to as "binary8" formats, and further qualified by precision yielding names "binary8pp".

For example, "binary8p3" is a format with 3 bits of precision (one implied – "the hidden bit" – and two explicit),

2 Values

This section describes the set of values that a binary8 format shall represent. The universe of values in existing floating point usage encompasses some finite real numerical values, the non-finite numerical values positive and negative infinity ($-\mathsf{Inf}$, $+\mathsf{Inf}$), the non-numeric not-a-number values (NaN, NaN_1, ...), and negative zero (-0.0). The value set for each binary8 format specifies the set of values that are available in that format.

Each binary format shall be associated with a unique encoding. An 8-bit binary encoding is a mapping from 8-bit strings to values. Some of these mappings are included in Appendix C.

The special values have encodings that are shared by all binary8 formats, as shown in table 2.

The set of finite floating-point numbers representable with a binary format is determined by two *format-defining* parameters:

- Precision p, the number of digits in the significand including the implicit leading bit.
- Maximum exponent emax, the exponent of the largest finite value.

IEEE-754 2019 includes the radix b and the minimum exponent $emin$ in the list of format-defining parameters, while this document excludes them with the following rationale:

- This document covers binary (radix 2) formats only, so b is not a format parameter.
- The parameter $emin$ is determinable from other parameters (, so is also not a format-defining parameter.

P3109 formats shall define $\mathsf{emax}(p)$ to be $\lceil \mathbf{2^{8-p-1} - 1} \rceil$. In IEEE-754, emax was consistently chosen across formats to be $2^{w-1} - 1$, where w is the exponent field width in bits. In this report, this convention is formalized: emax is a fixed function of p, written $\mathsf{emax}(p)$, with the formula as given above.

This choice of formula yields the following properties:

- the binary8pp value sets are subsets of the IEEE-754 binary16 value set for $p > 2$
- values are distributed close to symmetrically below and above the value 1.

For $p = 8$, the IEEE-754 formula yields $\mathsf{emax} = -\frac{1}{2}$, meaning all non-special values are irrational. Rounding the computation upward yields $\mathsf{emax}(8) = 0$, with the consequence that the value sets and encodings for binary8p7 and binary8p8 are identical.

The choice of emax for a given format then determines the exponent bias for that format. The bias is chosen so that the exponent of the largest finite value is emax. For IEEE-754 formats, the largest finite value corresponds to an exponent field which has all but the zeroth bit set (e.g. 11110 for binary16), because all of the values with all-bits-one exponents (ABOE values) are occupied by non-finite values (Not-a-Numbers or Infinities). Thus, the unbiased exponent of the largest finite value is $2^w - 2$, from which bias is computed as

$$\mathsf{emax} = (2^w - 2) - \mathsf{bias} \implies \mathsf{bias} = (2^w - 2) - (2^{w-1} - 1) = 2^{w-1} - 2 + 1 = \mathsf{emax}$$

For the binary8 formats in this document where $p > 1$, only one of the ABOE values is non-finite ($\pm$Infinity), so the unbiased exponent of the largest finite value is $2^w - 1$. Hence the bias calculation becomes

$$\mathsf{bias} = (2^w - 1) - (2^{w-1} - 1) = 2^{w-1} - 1 + 1 = \mathsf{emax} + 1$$

For $p = 1$, there are zero trailing significand bits, so all ABOE values are special, so again $\mathsf{bias} = \mathsf{emax}$.

Parameter	binary8p{p}	binary8p5	binary8p4	binary8p3	binary16	binary32	binary64
Storage width in bits k	**8**	**8**	**8**	**8**	**16**	**32**	**64**
Precision in bits p	**p**	**5**	**4**	**3**	**11**	**24**	**53**
Max exponent emax	$\lceil 2^{8-p-1} - 1 \rceil$	**3**	**7**	**15**	**15**	**127**	**1023**
Sign bit	1	1	1	1	1	1	1
Exponent field width w	$8 - p$	3	4	5	5	8	11
Exponent bias, bias	$\mathsf{emax} + (p > 1)$	4	8	16	15	127	1023
Trailing significand field width in bits t	$p - 1$	4	3	2	10	23	52

Table 1: Parameters for binary formats. Format-defining parameters in bold, derived parameters in normal font. Adapted from Table 3.5 of IEEE-754 (2019), and extended to include proposed binary8pp formats. Concepts are explained in detail in section 2.

2.1 Subnormals

Binary8 value sets shall include subnormals.

IEEE-754 value sets include subnormals. A value with trailing significand field m and exponent e is interpreted as $1.m \times 2^{e-b}$ except when all bits of the exponent bitfield are 0, in which case the value is $0.m \times 2^{e-b}$.

When training models, it is common to represent near-zero values for gradients. Subnormal numbers induce equal quantization steps around zero; this expands the reach of binary8 trainable models. In statistical applications, the subnormal range is useful for uniform and similar distributions; subnormals are uniformly spaced around zero. They also support working with Gaussian-like distributions, where numbers around zero are more probable.

2.2 Not a number (NaN)

Binary8 value sets shall include exactly one NaN, which shall not signal.

Other floating-point formats define several NaN values, denoted (NaN, NaN_1, ...). NaNs are returned from operations with results outside the set of values. For example, $\mathsf{DIV}(0, 0)$, or $\mathsf{ADD}(\mathsf{Inf}, -\mathsf{Inf})$. Multiple NaN encodings are used in other formats to allow different exceptional conditions to be distinguished.

In the context of machine learning systems, uses of NaN include:

- Debugging of code running on accelerator hardware. In AI accelerators, exceptions may be difficult or expensive to convey back to user code, so it is common practice to allow NaN values to propagate through calculations to indicate that an error has occurred.
- Use as a 'notable value' indicator. In some datasets, for example, tabular data, values may be missing. It is useful to use a value outside the normal numeric range to indicate the position of these values. Particularly when memory usage is a concern, as may be expected in applications where 8-bit formats are being considered, the use of a separate "mask" array, or a list of indices, imposes additional memory overhead. In some cases, Inf can be used as a missing value, but given the restricted range of binary8 formats, it is likely that infinity shall be used as a separate indicator of rounding from values outside of the finite range.
- The use of multiple NaN payloads is known in statistical code (e.g. the R system has NaN and N/A), but it is not widely used, and in the context of binary8, multiple NaNs impose either additional hardware complexity (using

Value	Hexadecimal Encoding	Bit Sequence
Zero	0x00	0000 0000
Positive Infinity (+Inf)	0x7F	0111 1111
Negative Infinity (−Inf)	0xFF	1111 1111
Not a Number (NaN)	0x80	1000 0000

Table 2: Mappings from special values to encodings, common to all binary8 formats.

only a subset of the significand range), or a large reduction in encoding space (e.g. 8 codes for E5, 16 codes for E4, 32 codes for E3).

2.3 Zero

Binary8 formats shall have exactly one zero. This zero value is nonnegative.

The inclusion of negative zero would incur the cost of an additional code point. Given the decision to encode only a single NaN, placing that NaN at the negative zero code point enables the strictly positive and strictly negative number ranges to be symmetric.

A key rationale for including -0 in IEEE-754 was the consistent implementation of branch cuts in the atan2 function [4, 5]. Although the atan function is common in deep learning, it is generally used as an activation function, rather than a trigonometric operation, and the atan2 function is rare, if not unknown, in deep learning applications. Furthermore, it is not expected that this standard shall define either atan or atan2.

A secondary reason for providing -0 is the hardware simplification offered by its presence in the implementation of sign/magnitude arithmetic. However, the existence of in-market implementations is evidence that the small hardware simplification has not been sufficient to balance the loss of one code point.

It might be considered that the use of integer comparisons in sorting would argue against placing NaN at the negative zero code point. For example, the JAX machine learning framework is known to sort using integer comparison [3]. However, such sorting still requires $O(n)$ preprocessing and postprocessing steps to enable the use of twos-complement integer comparison, and already has special treatment of NaN and -0, so eliminating -0 and placing NaN in the -0 position imposes negligible additional burden. Sorting using comparison operations, as typically implemented, is undefined in the presence of NaNs. However, existing practice is to sort NaNs using totalOrder.

2.4 Infinities

Binary8 formats shall include positive and negative infinities.

This decision causes a reduction in dynamic range (252 values rather than 254), while offering improved numerical robustness in important machine learning use cases.

Two generic classes of such usage are:

- Mask values, for example, in Transformer models in machine learning [1].
- Representation of overflow.

As illustrated in Appendix A, both usages are facilitated by the presence of infinity.

2.5 Extremal Values

Format	minSubnormal	maxSubnormal	minNormal	maxNormal	maxFinite
p2	1×2^{-32}	1×2^{-32}	1×2^{-31}	1×2^{31}	1×2^{31}
p3	1×2^{-17}	$3/2 \times 2^{-16}$	1×2^{-15}	$3/2 \times 2^{15}$	$3/2 \times 2^{15}$
p4	1×2^{-10}	$7/4 \times 2^{-8}$	1×2^{-7}	$7/4 \times 2^{7}$	$7/4 \times 2^{7}$
p5	1×2^{-7}	$15/8 \times 2^{-4}$	1×2^{-3}	$15/8 \times 2^{3}$	$15/8 \times 2^{3}$
p6	1×2^{-6}	$31/16 \times 2^{-2}$	1×2^{-1}	$31/16 \times 2^{1}$	$31/16 \times 2^{1}$
p7	1×2^{-6}	$63/32 \times 2^{-1}$	1×2^{0}	$63/32 \times 2^{0}$	$63/32 \times 2^{0}$

Table 3: Extremal values

It is practical to offer extremal finite values for supported 8-bit binary interchange formats. Following IEEE 754-2019 naming patterns, we adopt: $\mathsf{maxNormal}(\mathsf{T}), \mathsf{minNormal}(\mathsf{T}), \mathsf{minSubnormal}(\mathsf{T})$ where T is a binary8 format. For example: $\mathsf{maxNormal}(\mathsf{binary8p4}) = 7/4 \times 2^{7}$, $\mathsf{minNormal}(\mathsf{binary8p5}) = 1 \times 2^{-3}$.

Table 3 shows these values in binary8 formats for $1 < p < 8$.

3 Classification operators

Conforming implementations shall provide these classification predicates and the classifier function. The classification predicates and the classifier function shall not signal exceptions.

The classification operators comprise: 1) a set of functions with a boolean return value, taking a single binary8 value as input; 2) a function $\mathrm{class}(x)$ that returns a single value of enumeration type, describing the input value's properties.

Predicates shall behave as follows:

Predicate	Definition
isZero	iff [a] x is 0
isNaN	iff x is NaN
isInfinite	iff x is infinite
isFinite	iff x is zero, subnormal or normal
isNormal	iff x is normal, hence finite
isSubnormal	iff x is subnormal
isSignMinus	iff x has a negative sign [b]
isCanonical	True [c]
isSignaling	False [d]

Table 4: Classification Predicates

[a] iff abbreviates "if and only if"
[b] isSignMinus(NaN) is True: NaN is 0x80 (0b10000000).
[c] There are no non-canonical binary8 interchange formats.
[d] All binary8 formats have one NaN; it does not signal.

The Classifier function $\mathrm{class}(x)$ shall return enumeration values as follows:

Enumeration	Condition
NaN	isNaN(x)
Zero	isZero(x)
positiveInfinity	isInfinite(x) and not(isSignMinus(x))
positiveNormal	isNormal(x) and not(isSignMinus(x))
positiveSubnormal	isSubnormal(x) and not(isSignMinus(x))
negativeInfinity	isInfinite(x) and isSignMinus(x)
negativeNormal	isNormal(x) and isSignMinus(x)
negativeSubnormal	isSubnormal(x) and isSignMinus(x)

Table 5: Classifier Logic

4 Comparison operators

Conforming implementations shall provide the following comparison operators and the $\mathrm{totalOrder}(x, y)$ **function.**

Comparison operators are two argument predicates and their negations that return True or False. Comparisons shall not raise exceptions. Comparisons are ordered or unordered. A comparison is unordered iff either argument is NaN. All other comparisons are ordered.

For $\{=, >, \geq, <, \leq, \lessgtr\}$, if any argument is NaN, the result is False.

For $\{\neq, \not>, \not\geq, \not<, \not\leq, \not\lessgtr\}$, if any argument is NaN, the result is True.

Otherwise, the result of a comparison shall match the mathematical result.

math symbol	predicate *true relations*	math symbol	negation *true relations*
$=$	CompareEqual *equal*	$\neq$, NOT $=$	CompareNotEqual *less than, greater than, unordered*
$>$	CompareGreater *greater than*	$\not>$, NOT $>$	CompareNotGreater *less than, equal, unordered*
$\geq$	CompareGreaterEqual *equal, greater than*	$\not\geq$, NOT $\geq$	CompareLessUnordered *less than, unordered*
$<$	CompareLess *less than*	$\not<$, NOT $<$	CompareNotLess *greater than, equal, unordered*
$\leq$	CompareLessEqual *less than, equal*	$\not\leq$, NOT $\leq$	CompareGreaterUnordered *greater than, unordered*
$\lessgtr$	CompareOrdered *less than, equal, greater than*	$\not\lessgtr$, NOT $\lessgtr$	CompareUnordered *unordered*

Table 6: Comparison Predicates and Negations

4.1 The $\mathrm{totalOrder}$ predicate

$\mathrm{totalOrder}(x, y)$ provides a total ordering over each binary8 format's value set.

The predicate $\mathrm{totalOrder}(x, y)$ **shall return { True, False } in accord with the logic given below. It shall not raise any exceptions.**

```
boolean totalOrder(x, y)
    if isNaN(x): return True
    if isNaN(y): return False
    return compareLessEqual(x, y)
end
```

Note: Following 754's definition of $\mathrm{totalOrder}(x, y)$, binary8 NaNs (0x80) compare as the most negative value. The most significant bit of NaN is set, so to be consistent with 754, NaN is ordered before all numerical values.

A Numerical Examples

A.1 Mask Values

A common use for ∞ is to create masks, for example, in Transformer models in machine learning, [1].

These values, assembled in mask matrix M with values $M_{ij} in \{0, -\infty\}$ are typically added to computed values A, in a computation such as:

$$\log(\mathrm{sum}(\exp(\tau * (A + M))))$$

where τ is a "temperature" or "base" parameter [2]. This calculation depends on the property $\exp(\tau * (A_{ij} - \infty)) = 0$.

If a floating point encoding does not provide infinity, then instead M_{ij} will be replaced by a large float (e.g. 480). This is not in itself a difficulty: if all the A values are bounded (e.g. the results of a softmax operation), then $\exp?(1.0 - 480)$ is an extremely small number, which will certainly round to zero. Therefore, an explicit representation of infinity is *not* needed in order for this computation to yield its desired value.

However, careful implementations do not execute the calculation as written, and instead fuse the $\log(\mathrm{sum}(\exp(v)))$ operation into a single operation $\mathrm{logsumexp}(v)$, whose implementation makes use of the identity transformation

$$\mathrm{logsumexp}(v) \rightarrow \mathrm{logsumexp}(v - \max(v)) + \max(v)$$

Without the "sticky" properties of Inf, this would produce incorrect answers. For example, in a format where MaxFloat=240 without Inf, and MaxFloat=224 with Inf:

$$\mathrm{logsumexp}(\tau * [-224, -\infty]) \rightarrow \mathrm{logsumexp}(\tau * [0, -\infty])$$

while

$$\mathrm{logsumexp}(\tau * [-224, -240]) \rightarrow \mathrm{logsumexp}(\tau * [0, -16])$$

If $\tau = 1$ and all calculations are done in 8-bit floating point, then the answer will be the same, because $\exp(-16) \approx 1.1 \times 10^{-7}$, which will round to zero in all precisions $p > 2$; but if τ is small, or calculations are done in mixed precision, as is common with 8-bit floating point, the loss of "stickiness" will silently yield unexpected answers. It is not expected that the full calculation shall be done in 8-bit floating point, but the subtraction of the maximum value (and computation of the maximum) might reasonably be in 8-bit floating point.

A.2 Overflow to Infinity

A second use of infinity is to indicate overflow on conversion to the binary8 type. Existing implementations offer several behaviors on overflow: overflow to infinity, saturation to MaxFloat, and overflow to NaN. The existence of a code point for infinity allows any of these options to be implemented in a given instantiation, while removing the code point removes the possibility of implementing the first.

B Comparison table

This table summarizes the points of difference and agreement between the formats proposed in this document and a number of existing formats, some of which have hardware implementations.

OCP: Open Compute Platform [6], describing hardware implementations including nVidia, Intel, and ARM.

AGQ: AMD, Graphcore, Qualcomm[7], implemented in Graphcore's C600 product, and AMD's gfx940.

TSL: Tesla Dojo Technology [8], A Guide to Tesla's Configurable Floating Point Formats & Arithmetic

Format	P3109			OCP		AGQ		TSL	
Subformat	P3	P4	P5	E5	E4	E5	E4	E4	E5
Special values shared by all subformats	Y			N		Y		N	
Exactly one NaN	Y			N		Y		Y	
Positive and negative infinity	Y			N	Y	N		N	
Include negative zero	N			N		Y		N	
Max exponent emax	15	7	3	15	8	15	7	N/A	N/A

C Value Tables

Value tables mapping 8-bit strings to value sets are provided in this section.

A typical entry is of the form:

```
HEX    BINARY      = BINARY_FLOAT     = DECIMAL
0x01 = 0_00000_01 = +0b0.01 x 2^-15 = 7.62939453125E-06
```

Where the fields are interpreted as follows:

`HEX`	Hexadecimal encoding of the code point
`BINARY`	Binary expansion of the code point, with underscores separating `sign_exponent_significand`
`BINARY_FLOAT`	The precise float value as a binary fraction followed by $2\hat{}e$ with decimal exponent e
`DECIMAL`	The decimal expansion of the value. If the decimal expansion is not an exact representation of the precise float value, the preceding equals sign is replaced by "approximately equals" $\approx$.

C.1 Value Table: P3

0x00 = 0_00000_00 = 0.0
0x01 = 0_00000_01 = +0b0.01×2^-15 ≈ 7.6293945E-06
0x02 = 0_00000_10 = +0b0.10×2^-15 ≈ 1.5258789E-05
0x03 = 0_00000_11 = +0b0.11×2^-15 ≈ 2.2888184E-05
0x04 = 0_00001_00 = +0b1.00×2^-15 ≈ 3.0517578E-05
0x05 = 0_00001_01 = +0b1.01×2^-15 ≈ 3.8146973E-05
0x06 = 0_00001_10 = +0b1.10×2^-15 ≈ 4.5776367E-05
0x07 = 0_00001_11 = +0b1.11×2^-15 ≈ 5.3405762E-05
0x08 = 0_00010_00 = +0b1.00×2^-14 ≈ 6.1035156E-05
0x09 = 0_00010_01 = +0b1.01×2^-14 ≈ 7.6293945E-05
0x0a = 0_00010_10 = +0b1.10×2^-14 ≈ 9.1552734E-05
0x0b = 0_00010_11 = +0b1.11×2^-14 ≈ 0.00010681152
0x0c = 0_00011_00 = +0b1.00×2^-13 ≈ 0.00012207031
0x0d = 0_00011_01 = +0b1.01×2^-13 ≈ 0.00015258789
0x0e = 0_00011_10 = +0b1.10×2^-13 ≈ 0.00018310547
0x0f = 0_00011_11 = +0b1.11×2^-13 ≈ 0.00021362305
0x10 = 0_00100_00 = +0b1.00×2^-12 = 0.000244140625
0x11 = 0_00100_01 = +0b1.01×2^-12 ≈ 0.00030517578
0x12 = 0_00100_10 = +0b1.10×2^-12 ≈ 0.00036621094
0x13 = 0_00100_11 = +0b1.11×2^-12 ≈ 0.00042724609
0x14 = 0_00101_00 = +0b1.00×2^-11 = 0.0004882812 5
0x15 = 0_00101_01 = +0b1.01×2^-11 ≈ 0.00061035156
0x16 = 0_00101_10 = +0b1.10×2^-11 = 0.000732421875
0x17 = 0_00101_11 = +0b1.11×2^-11 ≈ 0.00085449219
0x18 = 0_00110_00 = +0b1.00×2^-10 = 0.0009765625
0x19 = 0_00110_01 = +0b1.01×2^-10 = 0.001220703125
0x1a = 0_00110_10 = +0b1.10×2^-10 = 0.00146484375
0x1b = 0_00110_11 = +0b1.11×2^-10 = 0.001708984375
0x1c = 0_00111_00 = +0b1.00×2^-9 = 0.001953125
0x1d = 0_00111_01 = +0b1.01×2^-9 = 0.00244140625
0x1e = 0_00111_10 = +0b1.10×2^-9 = 0.0029296875
0x1f = 0_00111_11 = +0b1.11×2^-9 = 0.00341796875
0x20 = 0_01000_00 = +0b1.00×2^-8 = 0.00390625
0x21 = 0_01000_01 = +0b1.01×2^-8 = 0.0048828125
0x22 = 0_01000_10 = +0b1.10×2^-8 = 0.005859375
0x23 = 0_01000_11 = +0b1.11×2^-8 = 0.0068359375
0x24 = 0_01001_00 = +0b1.00×2^-7 = 0.0078125
0x25 = 0_01001_01 = +0b1.01×2^-7 = 0.009765625
0x26 = 0_01001_10 = +0b1.10×2^-7 = 0.01171875
0x27 = 0_01001_11 = +0b1.11×2^-7 = 0.013671875
0x28 = 0_01010_00 = +0b1.00×2^-6 = 0.015625
0x29 = 0_01010_01 = +0b1.01×2^-6 = 0.01953125
0x2a = 0_01010_10 = +0b1.10×2^-6 = 0.0234375
0x2b = 0_01010_11 = +0b1.11×2^-6 = 0.02734375
0x2c = 0_01011_00 = +0b1.00×2^-5 = 0.03125
0x2d = 0_01011_01 = +0b1.01×2^-5 = 0.0390625
0x2e = 0_01011_10 = +0b1.10×2^-5 = 0.046875
0x2f = 0_01011_11 = +0b1.11×2^-5 = 0.0546875
0x30 = 0_01100_00 = +0b1.00×2^-4 = 0.0625
0x31 = 0_01100_01 = +0b1.01×2^-4 = 0.078125
0x32 = 0_01100_10 = +0b1.10×2^-4 = 0.09375
0x33 = 0_01100_11 = +0b1.11×2^-4 = 0.109375
0x34 = 0_01101_00 = +0b1.00×2^-3 = 0.125
0x35 = 0_01101_01 = +0b1.01×2^-3 = 0.15625
0x36 = 0_01101_10 = +0b1.10×2^-3 = 0.1875
0x37 = 0_01101_11 = +0b1.11×2^-3 = 0.21875
0x38 = 0_01110_00 = +0b1.00×2^-2 = 0.25
0x39 = 0_01110_01 = +0b1.01×2^-2 = 0.3125
0x3a = 0_01110_10 = +0b1.10×2^-2 = 0.375
0x3b = 0_01110_11 = +0b1.11×2^-2 = 0.4375
0x3c = 0_01111_00 = +0b1.00×2^-1 = 0.5
0x3d = 0_01111_01 = +0b1.01×2^-1 = 0.625
0x3e = 0_01111_10 = +0b1.10×2^-1 = 0.75
0x3f = 0_01111_11 = +0b1.11×2^-1 = 0.875

0x40 = 0_10000_00 = +0b1.00×2^0 = 1.0
0x41 = 0_10000_01 = +0b1.01×2^0 = 1.25
0x42 = 0_10000_10 = +0b1.10×2^0 = 1.5
0x43 = 0_10000_11 = +0b1.11×2^0 = 1.75
0x44 = 0_10001_00 = +0b1.00×2^1 = 2.0
0x45 = 0_10001_01 = +0b1.01×2^1 = 2.5
0x46 = 0_10001_10 = +0b1.10×2^1 = 3.0
0x47 = 0_10001_11 = +0b1.11×2^1 = 3.5
0x48 = 0_10010_00 = +0b1.00×2^2 = 4.0
0x49 = 0_10010_01 = +0b1.01×2^2 = 5.0
0x4a = 0_10010_10 = +0b1.10×2^2 = 6.0
0x4b = 0_10010_11 = +0b1.11×2^2 = 7.0
0x4c = 0_10011_00 = +0b1.00×2^3 = 8.0
0x4d = 0_10011_01 = +0b1.01×2^3 = 10.0
0x4e = 0_10011_10 = +0b1.10×2^3 = 12.0
0x4f = 0_10011_11 = +0b1.11×2^3 = 14.0
0x50 = 0_10100_00 = +0b1.00×2^4 = 16.0
0x51 = 0_10100_01 = +0b1.01×2^4 = 20.0
0x52 = 0_10100_10 = +0b1.10×2^4 = 24.0
0x53 = 0_10100_11 = +0b1.11×2^4 = 28.0
0x54 = 0_10101_00 = +0b1.00×2^5 = 32.0
0x55 = 0_10101_01 = +0b1.01×2^5 = 40.0
0x56 = 0_10101_10 = +0b1.10×2^5 = 48.0
0x57 = 0_10101_11 = +0b1.11×2^5 = 56.0
0x58 = 0_10110_00 = +0b1.00×2^6 = 64.0
0x59 = 0_10110_01 = +0b1.01×2^6 = 80.0
0x5a = 0_10110_10 = +0b1.10×2^6 = 96.0
0x5b = 0_10110_11 = +0b1.11×2^6 = 112.0
0x5c = 0_10111_00 = +0b1.00×2^7 = 128.0
0x5d = 0_10111_01 = +0b1.01×2^7 = 160.0
0x5e = 0_10111_10 = +0b1.10×2^7 = 192.0
0x5f = 0_10111_11 = +0b1.11×2^7 = 224.0
0x60 = 0_11000_00 = +0b1.00×2^8 = 256.0
0x61 = 0_11000_01 = +0b1.01×2^8 = 320.0
0x62 = 0_11000_10 = +0b1.10×2^8 = 384.0
0x63 = 0_11000_11 = +0b1.11×2^8 = 448.0
0x64 = 0_11001_00 = +0b1.00×2^9 = 512.0
0x65 = 0_11001_01 = +0b1.01×2^9 = 640.0
0x66 = 0_11001_10 = +0b1.10×2^9 = 768.0
0x67 = 0_11001_11 = +0b1.11×2^9 = 896.0
0x68 = 0_11010_00 = +0b1.00×2^10 = 1024.0
0x69 = 0_11010_01 = +0b1.01×2^10 = 1280.0
0x6a = 0_11010_10 = +0b1.10×2^10 = 1536.0
0x6b = 0_11010_11 = +0b1.11×2^10 = 1792.0
0x6c = 0_11011_00 = +0b1.00×2^11 = 2048.0
0x6d = 0_11011_01 = +0b1.01×2^11 = 2560.0
0x6e = 0_11011_10 = +0b1.10×2^11 = 3072.0
0x6f = 0_11011_11 = +0b1.11×2^11 = 3584.0
0x70 = 0_11100_00 = +0b1.00×2^12 = 4096.0
0x71 = 0_11100_01 = +0b1.01×2^12 = 5120.0
0x72 = 0_11100_10 = +0b1.10×2^12 = 6144.0
0x73 = 0_11100_11 = +0b1.11×2^12 = 7168.0
0x74 = 0_11101_00 = +0b1.00×2^13 = 8192.0
0x75 = 0_11101_01 = +0b1.01×2^13 = 10240.0
0x76 = 0_11101_10 = +0b1.10×2^13 = 12288.0
0x77 = 0_11101_11 = +0b1.11×2^13 = 14336.0
0x78 = 0_11110_00 = +0b1.00×2^14 = 16384.0
0x79 = 0_11110_01 = +0b1.01×2^14 = 20480.0
0x7a = 0_11110_10 = +0b1.10×2^14 = 24576.0
0x7b = 0_11110_11 = +0b1.11×2^14 = 28672.0
0x7c = 0_11111_00 = +0b1.00×2^15 = 32768.0
0x7d = 0_11111_01 = +0b1.01×2^15 = 40960.0
0x7e = 0_11111_10 = +0b1.10×2^15 = 49152.0
0x7f = 0_11111_11 = +Inf

0x80 = 1_00000_00 = NaN
0x81 = 1_00000_01 = −0b0.01×2^-15 ≈ −7.6293945E-06
0x82 = 1_00000_10 = −0b0.10×2^-15 ≈ −1.5258789E-05
0x83 = 1_00000_11 = −0b0.11×2^-15 ≈ −2.2888184E-05
0x84 = 1_00001_00 = −0b1.00×2^-15 ≈ −3.0517578E-05
0x85 = 1_00001_01 = −0b1.01×2^-15 ≈ −3.8146973E-05
0x86 = 1_00001_10 = −0b1.10×2^-15 ≈ −4.5776367E-05
0x87 = 1_00001_11 = −0b1.11×2^-15 ≈ −5.3405762E-05
0x88 = 1_00010_00 = −0b1.00×2^-14 ≈ −6.1035156E-05
0x89 = 1_00010_01 = −0b1.01×2^-14 ≈ −7.6293945E-05
0x8a = 1_00010_10 = −0b1.10×2^-14 ≈ −9.1552734E-05
0x8b = 1_00010_11 = −0b1.11×2^-14 ≈ -0.00010681152
0x8c = 1_00011_00 = −0b1.00×2^-13 ≈ -0.00012207031
0x8d = 1_00011_01 = −0b1.01×2^-13 ≈ -0.00015258789
0x8e = 1_00011_10 = −0b1.10×2^-13 ≈ -0.00018310547
0x8f = 1_00011_11 = −0b1.11×2^-13 ≈ -0.00021362305
0x90 = 1_00100_00 = −0b1.00×2^-12 ≈ -0.00024414062
0x91 = 1_00100_01 = −0b1.01×2^-12 ≈ -0.00030517578
0x92 = 1_00100_10 = −0b1.10×2^-12 ≈ -0.00036621094
0x93 = 1_00100_11 = −0b1.11×2^-12 ≈ -0.00042724609
0x94 = 1_00101_00 = −0b1.00×2^-11 = -0.0004882812 5
0x95 = 1_00101_01 = −0b1.01×2^-11 ≈ -0.00061035156
0x96 = 1_00101_10 = −0b1.10×2^-11 ≈ -0.00073242188
0x97 = 1_00101_11 = −0b1.11×2^-11 ≈ -0.00085449219
0x98 = 1_00110_00 = −0b1.00×2^-10 = -0.0009765625
0x99 = 1_00110_01 = −0b1.01×2^-10 ≈ -0.0012207031
0x9a = 1_00110_10 = −0b1.10×2^-10 = -0.00146484375
0x9b = 1_00110_11 = −0b1.11×2^-10 ≈ -0.0017089844
0x9c = 1_00111_00 = −0b1.00×2^-9 = -0.001953125
0x9d = 1_00111_01 = −0b1.01×2^-9 = -0.00244140625
0x9e = 1_00111_10 = −0b1.10×2^-9 = -0.0029296875
0x9f = 1_00111_11 = −0b1.11×2^-9 = -0.00341796875
0xa0 = 1_01000_00 = −0b1.00×2^-8 = -0.00390625
0xa1 = 1_01000_01 = −0b1.01×2^-8 = -0.0048828125
0xa2 = 1_01000_10 = −0b1.10×2^-8 = -0.005859375
0xa3 = 1_01000_11 = −0b1.11×2^-8 = -0.0068359375
0xa4 = 1_01001_00 = −0b1.00×2^-7 = -0.0078125
0xa5 = 1_01001_01 = −0b1.01×2^-7 = -0.009765625
0xa6 = 1_01001_10 = −0b1.10×2^-7 = -0.01171875
0xa7 = 1_01001_11 = −0b1.11×2^-7 = -0.013671875
0xa8 = 1_01010_00 = −0b1.00×2^-6 = -0.015625
0xa9 = 1_01010_01 = −0b1.01×2^-6 = -0.01953125
0xaa = 1_01010_10 = −0b1.10×2^-6 = -0.0234375
0xab = 1_01010_11 = −0b1.11×2^-6 = -0.02734375
0xac = 1_01011_00 = −0b1.00×2^-5 = -0.03125
0xad = 1_01011_01 = −0b1.01×2^-5 = -0.0390625
0xae = 1_01011_10 = −0b1.10×2^-5 = -0.046875
0xaf = 1_01011_11 = −0b1.11×2^-5 = -0.0546875
0xb0 = 1_01100_00 = −0b1.00×2^-4 = -0.0625
0xb1 = 1_01100_01 = −0b1.01×2^-4 = -0.078125
0xb2 = 1_01100_10 = −0b1.10×2^-4 = -0.09375
0xb3 = 1_01100_11 = −0b1.11×2^-4 = -0.109375
0xb4 = 1_01101_00 = −0b1.00×2^-3 = -0.125
0xb5 = 1_01101_01 = −0b1.01×2^-3 = -0.15625
0xb6 = 1_01101_10 = −0b1.10×2^-3 = -0.1875
0xb7 = 1_01101_11 = −0b1.11×2^-3 = -0.21875
0xb8 = 1_01110_00 = −0b1.00×2^-2 = -0.25
0xb9 = 1_01110_01 = −0b1.01×2^-2 = -0.3125
0xba = 1_01110_10 = −0b1.10×2^-2 = -0.375
0xbb = 1_01110_11 = −0b1.11×2^-2 = -0.4375
0xbc = 1_01111_00 = −0b1.00×2^-1 = -0.5
0xbd = 1_01111_01 = −0b1.01×2^-1 = -0.625
0xbe = 1_01111_10 = −0b1.10×2^-1 = -0.75
0xbf = 1_01111_11 = −0b1.11×2^-1 = -0.875

0xc0 = 1_10000_00 = −0b1.00×2^0 = -1.0
0xc1 = 1_10000_01 = −0b1.01×2^0 = -1.25
0xc2 = 1_10000_10 = −0b1.10×2^0 = -1.5
0xc3 = 1_10000_11 = −0b1.11×2^0 = -1.75
0xc4 = 1_10001_00 = −0b1.00×2^1 = -2.0
0xc5 = 1_10001_01 = −0b1.01×2^1 = -2.5
0xc6 = 1_10001_10 = −0b1.10×2^1 = -3.0
0xc7 = 1_10001_11 = −0b1.11×2^1 = -3.5
0xc8 = 1_10010_00 = −0b1.00×2^2 = -4.0
0xc9 = 1_10010_01 = −0b1.01×2^2 = -5.0
0xca = 1_10010_10 = −0b1.10×2^2 = -6.0
0xcb = 1_10010_11 = −0b1.11×2^2 = -7.0
0xcc = 1_10011_00 = −0b1.00×2^3 = -8.0
0xcd = 1_10011_01 = −0b1.01×2^3 = -10.0
0xce = 1_10011_10 = −0b1.10×2^3 = -12.0
0xcf = 1_10011_11 = −0b1.11×2^3 = -14.0
0xd0 = 1_10100_00 = −0b1.00×2^4 = -16.0
0xd1 = 1_10100_01 = −0b1.01×2^4 = -20.0
0xd2 = 1_10100_10 = −0b1.10×2^4 = -24.0
0xd3 = 1_10100_11 = −0b1.11×2^4 = -28.0
0xd4 = 1_10101_00 = −0b1.00×2^5 = -32.0
0xd5 = 1_10101_01 = −0b1.01×2^5 = -40.0
0xd6 = 1_10101_10 = −0b1.10×2^5 = -48.0
0xd7 = 1_10101_11 = −0b1.11×2^5 = -56.0
0xd8 = 1_10110_00 = −0b1.00×2^6 = -64.0
0xd9 = 1_10110_01 = −0b1.01×2^6 = -80.0
0xda = 1_10110_10 = −0b1.10×2^6 = -96.0
0xdb = 1_10110_11 = −0b1.11×2^6 = -112.0
0xdc = 1_10111_00 = −0b1.00×2^7 = -128.0
0xdd = 1_10111_01 = −0b1.01×2^7 = -160.0
0xde = 1_10111_10 = −0b1.10×2^7 = -192.0
0xdf = 1_10111_11 = −0b1.11×2^7 = -224.0
0xe0 = 1_11000_00 = −0b1.00×2^8 = -256.0
0xe1 = 1_11000_01 = −0b1.01×2^8 = -320.0
0xe2 = 1_11000_10 = −0b1.10×2^8 = -384.0
0xe3 = 1_11000_11 = −0b1.11×2^8 = -448.0
0xe4 = 1_11001_00 = −0b1.00×2^9 = -512.0
0xe5 = 1_11001_01 = −0b1.01×2^9 = -640.0
0xe6 = 1_11001_10 = −0b1.10×2^9 = -768.0
0xe7 = 1_11001_11 = −0b1.11×2^9 = -896.0
0xe8 = 1_11010_00 = −0b1.00×2^10 = -1024.0
0xe9 = 1_11010_01 = −0b1.01×2^10 = -1280.0
0xea = 1_11010_10 = −0b1.10×2^10 = -1536.0
0xeb = 1_11010_11 = −0b1.11×2^10 = -1792.0
0xec = 1_11011_00 = −0b1.00×2^11 = -2048.0
0xed = 1_11011_01 = −0b1.01×2^11 = -2560.0
0xee = 1_11011_10 = −0b1.10×2^11 = -3072.0
0xef = 1_11011_11 = −0b1.11×2^11 = -3584.0
0xf0 = 1_11100_00 = −0b1.00×2^12 = -4096.0
0xf1 = 1_11100_01 = −0b1.01×2^12 = -5120.0
0xf2 = 1_11100_10 = −0b1.10×2^12 = -6144.0
0xf3 = 1_11100_11 = −0b1.11×2^12 = -7168.0
0xf4 = 1_11101_00 = −0b1.00×2^13 = -8192.0
0xf5 = 1_11101_01 = −0b1.01×2^13 = -10240.0
0xf6 = 1_11101_10 = −0b1.10×2^13 = -12288.0
0xf7 = 1_11101_11 = −0b1.11×2^13 = -14336.0
0xf8 = 1_11110_00 = −0b1.00×2^14 = -16384.0
0xf9 = 1_11110_01 = −0b1.01×2^14 = -20480.0
0xfa = 1_11110_10 = −0b1.10×2^14 = -24576.0
0xfb = 1_11110_11 = −0b1.11×2^14 = -28672.0
0xfc = 1_11111_00 = −0b1.00×2^15 = -32768.0
0xfd = 1_11111_01 = −0b1.01×2^15 = -40960.0
0xfe = 1_11111_10 = −0b1.10×2^15 = -49152.0
0xff = 1_11111_11 = −Inf

C.2 Value Table: P4

0x00 = 0_0000_000 = 0.0
0x01 = 0_0000_001 = +0b0.001×2^-7 = 0.0009765625
0x02 = 0_0000_010 = +0b0.010×2^-7 = 0.001953125
0x03 = 0_0000_011 = +0b0.011×2^-7 = 0.0029296875
0x04 = 0_0000_100 = +0b0.100×2^-7 = 0.00390625
0x05 = 0_0000_101 = +0b0.101×2^-7 = 0.0048828125
0x06 = 0_0000_110 = +0b0.110×2^-7 = 0.005859375
0x07 = 0_0000_111 = +0b0.111×2^-7 = 0.0068359375
0x08 = 0_0001_000 = +0b1.000×2^-7 = 0.0078125
0x09 = 0_0001_001 = +0b1.001×2^-7 = 0.0087890625
0x0a = 0_0001_010 = +0b1.010×2^-7 = 0.009765625
0x0b = 0_0001_011 = +0b1.011×2^-7 = 0.0107421875
0x0c = 0_0001_100 = +0b1.100×2^-7 = 0.01171875
0x0d = 0_0001_101 = +0b1.101×2^-7 = 0.0126953125
0x0e = 0_0001_110 = +0b1.110×2^-7 = 0.013671875
0x0f = 0_0001_111 = +0b1.111×2^-7 = 0.0146484375
0x10 = 0_0010_000 = +0b1.000×2^-6 = 0.015625
0x11 = 0_0010_001 = +0b1.001×2^-6 = 0.017578125
0x12 = 0_0010_010 = +0b1.010×2^-6 = 0.01953125
0x13 = 0_0010_011 = +0b1.011×2^-6 = 0.021484375
0x14 = 0_0010_100 = +0b1.100×2^-6 = 0.0234375
0x15 = 0_0010_101 = +0b1.101×2^-6 = 0.025390625
0x16 = 0_0010_110 = +0b1.110×2^-6 = 0.02734375
0x17 = 0_0010_111 = +0b1.111×2^-6 = 0.029296875
0x18 = 0_0011_000 = +0b1.000×2^-5 = 0.03125
0x19 = 0_0011_001 = +0b1.001×2^-5 = 0.03515625
0x1a = 0_0011_010 = +0b1.010×2^-5 = 0.0390625
0x1b = 0_0011_011 = +0b1.011×2^-5 = 0.04296875
0x1c = 0_0011_100 = +0b1.100×2^-5 = 0.046875
0x1d = 0_0011_101 = +0b1.101×2^-5 = 0.05078125
0x1e = 0_0011_110 = +0b1.110×2^-5 = 0.0546875
0x1f = 0_0011_111 = +0b1.111×2^-5 = 0.05859375
0x20 = 0_0100_000 = +0b1.000×2^-4 = 0.0625
0x21 = 0_0100_001 = +0b1.001×2^-4 = 0.0703125
0x22 = 0_0100_010 = +0b1.010×2^-4 = 0.078125
0x23 = 0_0100_011 = +0b1.011×2^-4 = 0.0859375
0x24 = 0_0100_100 = +0b1.100×2^-4 = 0.09375
0x25 = 0_0100_101 = +0b1.101×2^-4 = 0.1015625
0x26 = 0_0100_110 = +0b1.110×2^-4 = 0.109375
0x27 = 0_0100_111 = +0b1.111×2^-4 = 0.1171875
0x28 = 0_0101_000 = +0b1.000×2^-3 = 0.125
0x29 = 0_0101_001 = +0b1.001×2^-3 = 0.140625
0x2a = 0_0101_010 = +0b1.010×2^-3 = 0.15625
0x2b = 0_0101_011 = +0b1.011×2^-3 = 0.171875
0x2c = 0_0101_100 = +0b1.100×2^-3 = 0.1875
0x2d = 0_0101_101 = +0b1.101×2^-3 = 0.203125
0x2e = 0_0101_110 = +0b1.110×2^-3 = 0.21875
0x2f = 0_0101_111 = +0b1.111×2^-3 = 0.234375
0x30 = 0_0110_000 = +0b1.000×2^-2 = 0.25
0x31 = 0_0110_001 = +0b1.001×2^-2 = 0.28125
0x32 = 0_0110_010 = +0b1.010×2^-2 = 0.3125
0x33 = 0_0110_011 = +0b1.011×2^-2 = 0.34375
0x34 = 0_0110_100 = +0b1.100×2^-2 = 0.375
0x35 = 0_0110_101 = +0b1.101×2^-2 = 0.40625
0x36 = 0_0110_110 = +0b1.110×2^-2 = 0.4375
0x37 = 0_0110_111 = +0b1.111×2^-2 = 0.46875
0x38 = 0_0111_000 = +0b1.000×2^-1 = 0.5
0x39 = 0_0111_001 = +0b1.001×2^-1 = 0.5625
0x3a = 0_0111_010 = +0b1.010×2^-1 = 0.625
0x3b = 0_0111_011 = +0b1.011×2^-1 = 0.6875
0x3c = 0_0111_100 = +0b1.100×2^-1 = 0.75
0x3d = 0_0111_101 = +0b1.101×2^-1 = 0.8125
0x3e = 0_0111_110 = +0b1.110×2^-1 = 0.875
0x3f = 0_0111_111 = +0b1.111×2^-1 = 0.9375
0x40 = 0_1000_000 = +0b1.000×2^0 = 1.0
0x41 = 0_1000_001 = +0b1.001×2^0 = 1.125
0x42 = 0_1000_010 = +0b1.010×2^0 = 1.25
0x43 = 0_1000_011 = +0b1.011×2^0 = 1.375
0x44 = 0_1000_100 = +0b1.100×2^0 = 1.5
0x45 = 0_1000_101 = +0b1.101×2^0 = 1.625
0x46 = 0_1000_110 = +0b1.110×2^0 = 1.75
0x47 = 0_1000_111 = +0b1.111×2^0 = 1.875
0x48 = 0_1001_000 = +0b1.000×2^1 = 2.0
0x49 = 0_1001_001 = +0b1.001×2^1 = 2.25
0x4a = 0_1001_010 = +0b1.010×2^1 = 2.5
0x4b = 0_1001_011 = +0b1.011×2^1 = 2.75
0x4c = 0_1001_100 = +0b1.100×2^1 = 3.0
0x4d = 0_1001_101 = +0b1.101×2^1 = 3.25
0x4e = 0_1001_110 = +0b1.110×2^1 = 3.5
0x4f = 0_1001_111 = +0b1.111×2^1 = 3.75
0x50 = 0_1010_000 = +0b1.000×2^2 = 4.0
0x51 = 0_1010_001 = +0b1.001×2^2 = 4.5
0x52 = 0_1010_010 = +0b1.010×2^2 = 5.0
0x53 = 0_1010_011 = +0b1.011×2^2 = 5.5
0x54 = 0_1010_100 = +0b1.100×2^2 = 6.0
0x55 = 0_1010_101 = +0b1.101×2^2 = 6.5
0x56 = 0_1010_110 = +0b1.110×2^2 = 7.0
0x57 = 0_1010_111 = +0b1.111×2^2 = 7.5
0x58 = 0_1011_000 = +0b1.000×2^3 = 8.0
0x59 = 0_1011_001 = +0b1.001×2^3 = 9.0
0x5a = 0_1011_010 = +0b1.010×2^3 = 10.0
0x5b = 0_1011_011 = +0b1.011×2^3 = 11.0
0x5c = 0_1011_100 = +0b1.100×2^3 = 12.0
0x5d = 0_1011_101 = +0b1.101×2^3 = 13.0
0x5e = 0_1011_110 = +0b1.110×2^3 = 14.0
0x5f = 0_1011_111 = +0b1.111×2^3 = 15.0
0x60 = 0_1100_000 = +0b1.000×2^4 = 16.0
0x61 = 0_1100_001 = +0b1.001×2^4 = 18.0
0x62 = 0_1100_010 = +0b1.010×2^4 = 20.0
0x63 = 0_1100_011 = +0b1.011×2^4 = 22.0
0x64 = 0_1100_100 = +0b1.100×2^4 = 24.0
0x65 = 0_1100_101 = +0b1.101×2^4 = 26.0
0x66 = 0_1100_110 = +0b1.110×2^4 = 28.0
0x67 = 0_1100_111 = +0b1.111×2^4 = 30.0
0x68 = 0_1101_000 = +0b1.000×2^5 = 32.0
0x69 = 0_1101_001 = +0b1.001×2^5 = 36.0
0x6a = 0_1101_010 = +0b1.010×2^5 = 40.0
0x6b = 0_1101_011 = +0b1.011×2^5 = 44.0
0x6c = 0_1101_100 = +0b1.100×2^5 = 48.0
0x6d = 0_1101_101 = +0b1.101×2^5 = 52.0
0x6e = 0_1101_110 = +0b1.110×2^5 = 56.0
0x6f = 0_1101_111 = +0b1.111×2^5 = 60.0
0x70 = 0_1110_000 = +0b1.000×2^6 = 64.0
0x71 = 0_1110_001 = +0b1.001×2^6 = 72.0
0x72 = 0_1110_010 = +0b1.010×2^6 = 80.0
0x73 = 0_1110_011 = +0b1.011×2^6 = 88.0
0x74 = 0_1110_100 = +0b1.100×2^6 = 96.0
0x75 = 0_1110_101 = +0b1.101×2^6 = 104.0
0x76 = 0_1110_110 = +0b1.110×2^6 = 112.0
0x77 = 0_1110_111 = +0b1.111×2^6 = 120.0
0x78 = 0_1111_000 = +0b1.000×2^7 = 128.0
0x79 = 0_1111_001 = +0b1.001×2^7 = 144.0
0x7a = 0_1111_010 = +0b1.010×2^7 = 160.0
0x7b = 0_1111_011 = +0b1.011×2^7 = 176.0
0x7c = 0_1111_100 = +0b1.100×2^7 = 192.0
0x7d = 0_1111_101 = +0b1.101×2^7 = 208.0
0x7e = 0_1111_110 = +0b1.110×2^7 = 224.0
0x7f = 0_1111_111 = +Inf
0x80 = 1_0000_000 = NaN
0x81 = 1_0000_001 = −0b0.001×2^-7 = -0.0009765625
0x82 = 1_0000_010 = −0b0.010×2^-7 = -0.001953125
0x83 = 1_0000_011 = −0b0.011×2^-7 = -0.0029296875
0x84 = 1_0000_100 = −0b0.100×2^-7 = -0.00390625
0x85 = 1_0000_101 = −0b0.101×2^-7 = -0.0048828125
0x86 = 1_0000_110 = −0b0.110×2^-7 = -0.005859375
0x87 = 1_0000_111 = −0b0.111×2^-7 = -0.0068359375
0x88 = 1_0001_000 = −0b1.000×2^-7 = -0.0078125
0x89 = 1_0001_001 = −0b1.001×2^-7 = -0.0087890625
0x8a = 1_0001_010 = −0b1.010×2^-7 = -0.009765625
0x8b = 1_0001_011 = −0b1.011×2^-7 = -0.0107421875
0x8c = 1_0001_100 = −0b1.100×2^-7 = -0.01171875
0x8d = 1_0001_101 = −0b1.101×2^-7 = -0.0126953125
0x8e = 1_0001_110 = −0b1.110×2^-7 = -0.013671875
0x8f = 1_0001_111 = −0b1.111×2^-7 = -0.0146484375
0x90 = 1_0010_000 = −0b1.000×2^-6 = -0.015625
0x91 = 1_0010_001 = −0b1.001×2^-6 = -0.017578125
0x92 = 1_0010_010 = −0b1.010×2^-6 = -0.01953125
0x93 = 1_0010_011 = −0b1.011×2^-6 = -0.021484375
0x94 = 1_0010_100 = −0b1.100×2^-6 = -0.0234375
0x95 = 1_0010_101 = −0b1.101×2^-6 = -0.025390625
0x96 = 1_0010_110 = −0b1.110×2^-6 = -0.02734375
0x97 = 1_0010_111 = −0b1.111×2^-6 = -0.029296875
0x98 = 1_0011_000 = −0b1.000×2^-5 = -0.03125
0x99 = 1_0011_001 = −0b1.001×2^-5 = -0.03515625
0x9a = 1_0011_010 = −0b1.010×2^-5 = -0.0390625
0x9b = 1_0011_011 = −0b1.011×2^-5 = -0.04296875
0x9c = 1_0011_100 = −0b1.100×2^-5 = -0.046875
0x9d = 1_0011_101 = −0b1.101×2^-5 = -0.05078125
0x9e = 1_0011_110 = −0b1.110×2^-5 = -0.0546875
0x9f = 1_0011_111 = −0b1.111×2^-5 = -0.05859375
0xa0 = 1_0100_000 = −0b1.000×2^-4 = -0.0625
0xa1 = 1_0100_001 = −0b1.001×2^-4 = -0.0703125
0xa2 = 1_0100_010 = −0b1.010×2^-4 = -0.078125
0xa3 = 1_0100_011 = −0b1.011×2^-4 = -0.0859375
0xa4 = 1_0100_100 = −0b1.100×2^-4 = -0.09375
0xa5 = 1_0100_101 = −0b1.101×2^-4 = -0.1015625
0xa6 = 1_0100_110 = −0b1.110×2^-4 = -0.109375
0xa7 = 1_0100_111 = −0b1.111×2^-4 = -0.1171875
0xa8 = 1_0101_000 = −0b1.000×2^-3 = -0.125
0xa9 = 1_0101_001 = −0b1.001×2^-3 = -0.140625
0xaa = 1_0101_010 = −0b1.010×2^-3 = -0.15625
0xab = 1_0101_011 = −0b1.011×2^-3 = -0.171875
0xac = 1_0101_100 = −0b1.100×2^-3 = -0.1875
0xad = 1_0101_101 = −0b1.101×2^-3 = -0.203125
0xae = 1_0101_110 = −0b1.110×2^-3 = -0.21875
0xaf = 1_0101_111 = −0b1.111×2^-3 = -0.234375
0xb0 = 1_0110_000 = −0b1.000×2^-2 = -0.25
0xb1 = 1_0110_001 = −0b1.001×2^-2 = -0.28125
0xb2 = 1_0110_010 = −0b1.010×2^-2 = -0.3125
0xb3 = 1_0110_011 = −0b1.011×2^-2 = -0.34375
0xb4 = 1_0110_100 = −0b1.100×2^-2 = -0.375
0xb5 = 1_0110_101 = −0b1.101×2^-2 = -0.40625
0xb6 = 1_0110_110 = −0b1.110×2^-2 = -0.4375
0xb7 = 1_0110_111 = −0b1.111×2^-2 = -0.46875
0xb8 = 1_0111_000 = −0b1.000×2^-1 = -0.5
0xb9 = 1_0111_001 = −0b1.001×2^-1 = -0.5625
0xba = 1_0111_010 = −0b1.010×2^-1 = -0.625
0xbb = 1_0111_011 = −0b1.011×2^-1 = -0.6875
0xbc = 1_0111_100 = −0b1.100×2^-1 = -0.75
0xbd = 1_0111_101 = −0b1.101×2^-1 = -0.8125
0xbe = 1_0111_110 = −0b1.110×2^-1 = -0.875
0xbf = 1_0111_111 = −0b1.111×2^-1 = -0.9375
0xc0 = 1_1000_000 = −0b1.000×2^0 = -1.0
0xc1 = 1_1000_001 = −0b1.001×2^0 = -1.125
0xc2 = 1_1000_010 = −0b1.010×2^0 = -1.25
0xc3 = 1_1000_011 = −0b1.011×2^0 = -1.375
0xc4 = 1_1000_100 = −0b1.100×2^0 = -1.5
0xc5 = 1_1000_101 = −0b1.101×2^0 = -1.625
0xc6 = 1_1000_110 = −0b1.110×2^0 = -1.75
0xc7 = 1_1000_111 = −0b1.111×2^0 = -1.875
0xc8 = 1_1001_000 = −0b1.000×2^1 = -2.0
0xc9 = 1_1001_001 = −0b1.001×2^1 = -2.25
0xca = 1_1001_010 = −0b1.010×2^1 = -2.5
0xcb = 1_1001_011 = −0b1.011×2^1 = -2.75
0xcc = 1_1001_100 = −0b1.100×2^1 = -3.0
0xcd = 1_1001_101 = −0b1.101×2^1 = -3.25
0xce = 1_1001_110 = −0b1.110×2^1 = -3.5
0xcf = 1_1001_111 = −0b1.111×2^1 = -3.75
0xd0 = 1_1010_000 = −0b1.000×2^2 = -4.0
0xd1 = 1_1010_001 = −0b1.001×2^2 = -4.5
0xd2 = 1_1010_010 = −0b1.010×2^2 = -5.0
0xd3 = 1_1010_011 = −0b1.011×2^2 = -5.5
0xd4 = 1_1010_100 = −0b1.100×2^2 = -6.0
0xd5 = 1_1010_101 = −0b1.101×2^2 = -6.5
0xd6 = 1_1010_110 = −0b1.110×2^2 = -7.0
0xd7 = 1_1010_111 = −0b1.111×2^2 = -7.5
0xd8 = 1_1011_000 = −0b1.000×2^3 = -8.0
0xd9 = 1_1011_001 = −0b1.001×2^3 = -9.0
0xda = 1_1011_010 = −0b1.010×2^3 = -10.0
0xdb = 1_1011_011 = −0b1.011×2^3 = -11.0
0xdc = 1_1011_100 = −0b1.100×2^3 = -12.0
0xdd = 1_1011_101 = −0b1.101×2^3 = -13.0
0xde = 1_1011_110 = −0b1.110×2^3 = -14.0
0xdf = 1_1011_111 = −0b1.111×2^3 = -15.0
0xe0 = 1_1100_000 = −0b1.000×2^4 = -16.0
0xe1 = 1_1100_001 = −0b1.001×2^4 = -18.0
0xe2 = 1_1100_010 = −0b1.010×2^4 = -20.0
0xe3 = 1_1100_011 = −0b1.011×2^4 = -22.0
0xe4 = 1_1100_100 = −0b1.100×2^4 = -24.0
0xe5 = 1_1100_101 = −0b1.101×2^4 = -26.0
0xe6 = 1_1100_110 = −0b1.110×2^4 = -28.0
0xe7 = 1_1100_111 = −0b1.111×2^4 = -30.0
0xe8 = 1_1101_000 = −0b1.000×2^5 = -32.0
0xe9 = 1_1101_001 = −0b1.001×2^5 = -36.0
0xea = 1_1101_010 = −0b1.010×2^5 = -40.0
0xeb = 1_1101_011 = −0b1.011×2^5 = -44.0
0xec = 1_1101_100 = −0b1.100×2^5 = -48.0
0xed = 1_1101_101 = −0b1.101×2^5 = -52.0
0xee = 1_1101_110 = −0b1.110×2^5 = -56.0
0xef = 1_1101_111 = −0b1.111×2^5 = -60.0
0xf0 = 1_1110_000 = −0b1.000×2^6 = -64.0
0xf1 = 1_1110_001 = −0b1.001×2^6 = -72.0
0xf2 = 1_1110_010 = −0b1.010×2^6 = -80.0
0xf3 = 1_1110_011 = −0b1.011×2^6 = -88.0
0xf4 = 1_1110_100 = −0b1.100×2^6 = -96.0
0xf5 = 1_1110_101 = −0b1.101×2^6 = -104.0
0xf6 = 1_1110_110 = −0b1.110×2^6 = -112.0
0xf7 = 1_1110_111 = −0b1.111×2^6 = -120.0
0xf8 = 1_1111_000 = −0b1.000×2^7 = -128.0
0xf9 = 1_1111_001 = −0b1.001×2^7 = -144.0
0xfa = 1_1111_010 = −0b1.010×2^7 = -160.0
0xfb = 1_1111_011 = −0b1.011×2^7 = -176.0
0xfc = 1_1111_100 = −0b1.100×2^7 = -192.0
0xfd = 1_1111_101 = −0b1.101×2^7 = -208.0
0xfe = 1_1111_110 = −0b1.110×2^7 = -224.0
0xff = 1_1111_111 = −Inf

C.3 Value Table: P5

0x00 = 0_000_0000 = 0.0
0x01 = 0_000_0001 = +0b0.0001×2^-3 = 0.0078125
0x02 = 0_000_0010 = +0b0.0010×2^-3 = 0.015625
0x03 = 0_000_0011 = +0b0.0011×2^-3 = 0.0234375
0x04 = 0_000_0100 = +0b0.0100×2^-3 = 0.03125
0x05 = 0_000_0101 = +0b0.0101×2^-3 = 0.0390625
0x06 = 0_000_0110 = +0b0.0110×2^-3 = 0.046875
0x07 = 0_000_0111 = +0b0.0111×2^-3 = 0.0546875
0x08 = 0_000_1000 = +0b0.1000×2^-3 = 0.0625
0x09 = 0_000_1001 = +0b0.1001×2^-3 = 0.0703125
0x0a = 0_000_1010 = +0b0.1010×2^-3 = 0.078125
0x0b = 0_000_1011 = +0b0.1011×2^-3 = 0.0859375
0x0c = 0_000_1100 = +0b0.1100×2^-3 = 0.09375
0x0d = 0_000_1101 = +0b0.1101×2^-3 = 0.1015625
0x0e = 0_000_1110 = +0b0.1110×2^-3 = 0.109375
0x0f = 0_000_1111 = +0b0.1111×2^-3 = 0.1171875
0x10 = 0_001_0000 = +0b1.0000×2^-3 = 0.125
0x11 = 0_001_0001 = +0b1.0001×2^-3 = 0.1328125
0x12 = 0_001_0010 = +0b1.0010×2^-3 = 0.140625
0x13 = 0_001_0011 = +0b1.0011×2^-3 = 0.1484375
0x14 = 0_001_0100 = +0b1.0100×2^-3 = 0.15625
0x15 = 0_001_0101 = +0b1.0101×2^-3 = 0.1640625
0x16 = 0_001_0110 = +0b1.0110×2^-3 = 0.171875
0x17 = 0_001_0111 = +0b1.0111×2^-3 = 0.1796875
0x18 = 0_001_1000 = +0b1.1000×2^-3 = 0.1875
0x19 = 0_001_1001 = +0b1.1001×2^-3 = 0.1953125
0x1a = 0_001_1010 = +0b1.1010×2^-3 = 0.203125
0x1b = 0_001_1011 = +0b1.1011×2^-3 = 0.2109375
0x1c = 0_001_1100 = +0b1.1100×2^-3 = 0.21875
0x1d = 0_001_1101 = +0b1.1101×2^-3 = 0.2265625
0x1e = 0_001_1110 = +0b1.1110×2^-3 = 0.234375
0x1f = 0_001_1111 = +0b1.1111×2^-3 = 0.2421875
0x20 = 0_010_0000 = +0b1.0000×2^-2 = 0.25
0x21 = 0_010_0001 = +0b1.0001×2^-2 = 0.265625
0x22 = 0_010_0010 = +0b1.0010×2^-2 = 0.28125
0x23 = 0_010_0011 = +0b1.0011×2^-2 = 0.296875
0x24 = 0_010_0100 = +0b1.0100×2^-2 = 0.3125
0x25 = 0_010_0101 = +0b1.0101×2^-2 = 0.328125
0x26 = 0_010_0110 = +0b1.0110×2^-2 = 0.34375
0x27 = 0_010_0111 = +0b1.0111×2^-2 = 0.359375
0x28 = 0_010_1000 = +0b1.1000×2^-2 = 0.375
0x29 = 0_010_1001 = +0b1.1001×2^-2 = 0.390625
0x2a = 0_010_1010 = +0b1.1010×2^-2 = 0.40625
0x2b = 0_010_1011 = +0b1.1011×2^-2 = 0.421875
0x2c = 0_010_1100 = +0b1.1100×2^-2 = 0.4375
0x2d = 0_010_1101 = +0b1.1101×2^-2 = 0.453125
0x2e = 0_010_1110 = +0b1.1110×2^-2 = 0.46875
0x2f = 0_010_1111 = +0b1.1111×2^-2 = 0.484375
0x30 = 0_011_0000 = +0b1.0000×2^-1 = 0.5
0x31 = 0_011_0001 = +0b1.0001×2^-1 = 0.53125
0x32 = 0_011_0010 = +0b1.0010×2^-1 = 0.5625
0x33 = 0_011_0011 = +0b1.0011×2^-1 = 0.59375
0x34 = 0_011_0100 = +0b1.0100×2^-1 = 0.625
0x35 = 0_011_0101 = +0b1.0101×2^-1 = 0.65625
0x36 = 0_011_0110 = +0b1.0110×2^-1 = 0.6875
0x37 = 0_011_0111 = +0b1.0111×2^-1 = 0.71875
0x38 = 0_011_1000 = +0b1.1000×2^-1 = 0.75
0x39 = 0_011_1001 = +0b1.1001×2^-1 = 0.78125
0x3a = 0_011_1010 = +0b1.1010×2^-1 = 0.8125
0x3b = 0_011_1011 = +0b1.1011×2^-1 = 0.84375
0x3c = 0_011_1100 = +0b1.1100×2^-1 = 0.875
0x3d = 0_011_1101 = +0b1.1101×2^-1 = 0.90625
0x3e = 0_011_1110 = +0b1.1110×2^-1 = 0.9375
0x3f = 0_011_1111 = +0b1.1111×2^-1 = 0.96875
0x40 = 0_100_0000 = +0b1.0000×2^0 = 1.0
0x41 = 0_100_0001 = +0b1.0001×2^0 = 1.0625
0x42 = 0_100_0010 = +0b1.0010×2^0 = 1.125
0x43 = 0_100_0011 = +0b1.0011×2^0 = 1.1875
0x44 = 0_100_0100 = +0b1.0100×2^0 = 1.25
0x45 = 0_100_0101 = +0b1.0101×2^0 = 1.3125
0x46 = 0_100_0110 = +0b1.0110×2^0 = 1.375
0x47 = 0_100_0111 = +0b1.0111×2^0 = 1.4375
0x48 = 0_100_1000 = +0b1.1000×2^0 = 1.5
0x49 = 0_100_1001 = +0b1.1001×2^0 = 1.5625
0x4a = 0_100_1010 = +0b1.1010×2^0 = 1.625
0x4b = 0_100_1011 = +0b1.1011×2^0 = 1.6875
0x4c = 0_100_1100 = +0b1.1100×2^0 = 1.75
0x4d = 0_100_1101 = +0b1.1101×2^0 = 1.8125
0x4e = 0_100_1110 = +0b1.1110×2^0 = 1.875
0x4f = 0_100_1111 = +0b1.1111×2^0 = 1.9375
0x50 = 0_101_0000 = +0b1.0000×2^1 = 2.0
0x51 = 0_101_0001 = +0b1.0001×2^1 = 2.125
0x52 = 0_101_0010 = +0b1.0010×2^1 = 2.25
0x53 = 0_101_0011 = +0b1.0011×2^1 = 2.375
0x54 = 0_101_0100 = +0b1.0100×2^1 = 2.5
0x55 = 0_101_0101 = +0b1.0101×2^1 = 2.625
0x56 = 0_101_0110 = +0b1.0110×2^1 = 2.75
0x57 = 0_101_0111 = +0b1.0111×2^1 = 2.875
0x58 = 0_101_1000 = +0b1.1000×2^1 = 3.0
0x59 = 0_101_1001 = +0b1.1001×2^1 = 3.125
0x5a = 0_101_1010 = +0b1.1010×2^1 = 3.25
0x5b = 0_101_1011 = +0b1.1011×2^1 = 3.375
0x5c = 0_101_1100 = +0b1.1100×2^1 = 3.5
0x5d = 0_101_1101 = +0b1.1101×2^1 = 3.625
0x5e = 0_101_1110 = +0b1.1110×2^1 = 3.75
0x5f = 0_101_1111 = +0b1.1111×2^1 = 3.875
0x60 = 0_110_0000 = +0b1.0000×2^2 = 4.0
0x61 = 0_110_0001 = +0b1.0001×2^2 = 4.25
0x62 = 0_110_0010 = +0b1.0010×2^2 = 4.5
0x63 = 0_110_0011 = +0b1.0011×2^2 = 4.75
0x64 = 0_110_0100 = +0b1.0100×2^2 = 5.0
0x65 = 0_110_0101 = +0b1.0101×2^2 = 5.25
0x66 = 0_110_0110 = +0b1.0110×2^2 = 5.5
0x67 = 0_110_0111 = +0b1.0111×2^2 = 5.75
0x68 = 0_110_1000 = +0b1.1000×2^2 = 6.0
0x69 = 0_110_1001 = +0b1.1001×2^2 = 6.25
0x6a = 0_110_1010 = +0b1.1010×2^2 = 6.5
0x6b = 0_110_1011 = +0b1.1011×2^2 = 6.75
0x6c = 0_110_1100 = +0b1.1100×2^2 = 7.0
0x6d = 0_110_1101 = +0b1.1101×2^2 = 7.25
0x6e = 0_110_1110 = +0b1.1110×2^2 = 7.5
0x6f = 0_110_1111 = +0b1.1111×2^2 = 7.75
0x70 = 0_111_0000 = +0b1.0000×2^3 = 8.0
0x71 = 0_111_0001 = +0b1.0001×2^3 = 8.5
0x72 = 0_111_0010 = +0b1.0010×2^3 = 9.0
0x73 = 0_111_0011 = +0b1.0011×2^3 = 9.5
0x74 = 0_111_0100 = +0b1.0100×2^3 = 10.0
0x75 = 0_111_0101 = +0b1.0101×2^3 = 10.5
0x76 = 0_111_0110 = +0b1.0110×2^3 = 11.0
0x77 = 0_111_0111 = +0b1.0111×2^3 = 11.5
0x78 = 0_111_1000 = +0b1.1000×2^3 = 12.0
0x79 = 0_111_1001 = +0b1.1001×2^3 = 12.5
0x7a = 0_111_1010 = +0b1.1010×2^3 = 13.0
0x7b = 0_111_1011 = +0b1.1011×2^3 = 13.5
0x7c = 0_111_1100 = +0b1.1100×2^3 = 14.0
0x7d = 0_111_1101 = +0b1.1101×2^3 = 14.5
0x7e = 0_111_1110 = +0b1.1110×2^3 = 15.0
0x7f = 0_111_1111 = +Inf
0x80 = 1_000_0000 = NaN
0x81 = 1_000_0001 = −0b0.0001×2^-3 = -0.0078125
0x82 = 1_000_0010 = −0b0.0010×2^-3 = -0.015625
0x83 = 1_000_0011 = −0b0.0011×2^-3 = -0.0234375
0x84 = 1_000_0100 = −0b0.0100×2^-3 = -0.03125
0x85 = 1_000_0101 = −0b0.0101×2^-3 = -0.0390625
0x86 = 1_000_0110 = −0b0.0110×2^-3 = -0.046875
0x87 = 1_000_0111 = −0b0.0111×2^-3 = -0.0546875
0x88 = 1_000_1000 = −0b0.1000×2^-3 = -0.0625
0x89 = 1_000_1001 = −0b0.1001×2^-3 = -0.0703125
0x8a = 1_000_1010 = −0b0.1010×2^-3 = -0.078125
0x8b = 1_000_1011 = −0b0.1011×2^-3 = -0.0859375
0x8c = 1_000_1100 = −0b0.1100×2^-3 = -0.09375
0x8d = 1_000_1101 = −0b0.1101×2^-3 = -0.1015625
0x8e = 1_000_1110 = −0b0.1110×2^-3 = -0.109375
0x8f = 1_000_1111 = −0b0.1111×2^-3 = -0.1171875
0x90 = 1_001_0000 = −0b1.0000×2^-3 = -0.125
0x91 = 1_001_0001 = −0b1.0001×2^-3 = -0.1328125
0x92 = 1_001_0010 = −0b1.0010×2^-3 = -0.140625
0x93 = 1_001_0011 = −0b1.0011×2^-3 = -0.1484375
0x94 = 1_001_0100 = −0b1.0100×2^-3 = -0.15625
0x95 = 1_001_0101 = −0b1.0101×2^-3 = -0.1640625
0x96 = 1_001_0110 = −0b1.0110×2^-3 = -0.171875
0x97 = 1_001_0111 = −0b1.0111×2^-3 = -0.1796875
0x98 = 1_001_1000 = −0b1.1000×2^-3 = -0.1875
0x99 = 1_001_1001 = −0b1.1001×2^-3 = -0.1953125
0x9a = 1_001_1010 = −0b1.1010×2^-3 = -0.203125
0x9b = 1_001_1011 = −0b1.1011×2^-3 = -0.2109375
0x9c = 1_001_1100 = −0b1.1100×2^-3 = -0.21875
0x9d = 1_001_1101 = −0b1.1101×2^-3 = -0.2265625
0x9e = 1_001_1110 = −0b1.1110×2^-3 = -0.234375
0x9f = 1_001_1111 = −0b1.1111×2^-3 = -0.2421875
0xa0 = 1_010_0000 = −0b1.0000×2^-2 = -0.25
0xa1 = 1_010_0001 = −0b1.0001×2^-2 = -0.265625
0xa2 = 1_010_0010 = −0b1.0010×2^-2 = -0.28125
0xa3 = 1_010_0011 = −0b1.0011×2^-2 = -0.296875
0xa4 = 1_010_0100 = −0b1.0100×2^-2 = -0.3125
0xa5 = 1_010_0101 = −0b1.0101×2^-2 = -0.328125
0xa6 = 1_010_0110 = −0b1.0110×2^-2 = -0.34375
0xa7 = 1_010_0111 = −0b1.0111×2^-2 = -0.359375
0xa8 = 1_010_1000 = −0b1.1000×2^-2 = -0.375
0xa9 = 1_010_1001 = −0b1.1001×2^-2 = -0.390625
0xaa = 1_010_1010 = −0b1.1010×2^-2 = -0.40625
0xab = 1_010_1011 = −0b1.1011×2^-2 = -0.421875
0xac = 1_010_1100 = −0b1.1100×2^-2 = -0.4375
0xad = 1_010_1101 = −0b1.1101×2^-2 = -0.453125
0xae = 1_010_1110 = −0b1.1110×2^-2 = -0.46875
0xaf = 1_010_1111 = −0b1.1111×2^-2 = -0.484375
0xb0 = 1_011_0000 = −0b1.0000×2^-1 = -0.5
0xb1 = 1_011_0001 = −0b1.0001×2^-1 = -0.53125
0xb2 = 1_011_0010 = −0b1.0010×2^-1 = -0.5625
0xb3 = 1_011_0011 = −0b1.0011×2^-1 = -0.59375
0xb4 = 1_011_0100 = −0b1.0100×2^-1 = -0.625
0xb5 = 1_011_0101 = −0b1.0101×2^-1 = -0.65625
0xb6 = 1_011_0110 = −0b1.0110×2^-1 = -0.6875
0xb7 = 1_011_0111 = −0b1.0111×2^-1 = -0.71875
0xb8 = 1_011_1000 = −0b1.1000×2^-1 = -0.75
0xb9 = 1_011_1001 = −0b1.1001×2^-1 = -0.78125
0xba = 1_011_1010 = −0b1.1010×2^-1 = -0.8125
0xbb = 1_011_1011 = −0b1.1011×2^-1 = -0.84375
0xbc = 1_011_1100 = −0b1.1100×2^-1 = -0.875
0xbd = 1_011_1101 = −0b1.1101×2^-1 = -0.90625
0xbe = 1_011_1110 = −0b1.1110×2^-1 = -0.9375
0xbf = 1_011_1111 = −0b1.1111×2^-1 = -0.96875
0xc0 = 1_100_0000 = −0b1.0000×2^0 = -1.0
0xc1 = 1_100_0001 = −0b1.0001×2^0 = -1.0625
0xc2 = 1_100_0010 = −0b1.0010×2^0 = -1.125
0xc3 = 1_100_0011 = −0b1.0011×2^0 = -1.1875
0xc4 = 1_100_0100 = −0b1.0100×2^0 = -1.25
0xc5 = 1_100_0101 = −0b1.0101×2^0 = -1.3125
0xc6 = 1_100_0110 = −0b1.0110×2^0 = -1.375
0xc7 = 1_100_0111 = −0b1.0111×2^0 = -1.4375
0xc8 = 1_100_1000 = −0b1.1000×2^0 = -1.5
0xc9 = 1_100_1001 = −0b1.1001×2^0 = -1.5625
0xca = 1_100_1010 = −0b1.1010×2^0 = -1.625
0xcb = 1_100_1011 = −0b1.1011×2^0 = -1.6875
0xcc = 1_100_1100 = −0b1.1100×2^0 = -1.75
0xcd = 1_100_1101 = −0b1.1101×2^0 = -1.8125
0xce = 1_100_1110 = −0b1.1110×2^0 = -1.875
0xcf = 1_100_1111 = −0b1.1111×2^0 = -1.9375
0xd0 = 1_101_0000 = −0b1.0000×2^1 = -2.0
0xd1 = 1_101_0001 = −0b1.0001×2^1 = -2.125
0xd2 = 1_101_0010 = −0b1.0010×2^1 = -2.25
0xd3 = 1_101_0011 = −0b1.0011×2^1 = -2.375
0xd4 = 1_101_0100 = −0b1.0100×2^1 = -2.5
0xd5 = 1_101_0101 = −0b1.0101×2^1 = -2.625
0xd6 = 1_101_0110 = −0b1.0110×2^1 = -2.75
0xd7 = 1_101_0111 = −0b1.0111×2^1 = -2.875
0xd8 = 1_101_1000 = −0b1.1000×2^1 = -3.0
0xd9 = 1_101_1001 = −0b1.1001×2^1 = -3.125
0xda = 1_101_1010 = −0b1.1010×2^1 = -3.25
0xdb = 1_101_1011 = −0b1.1011×2^1 = -3.375
0xdc = 1_101_1100 = −0b1.1100×2^1 = -3.5
0xdd = 1_101_1101 = −0b1.1101×2^1 = -3.625
0xde = 1_101_1110 = −0b1.1110×2^1 = -3.75
0xdf = 1_101_1111 = −0b1.1111×2^1 = -3.875
0xe0 = 1_110_0000 = −0b1.0000×2^2 = -4.0
0xe1 = 1_110_0001 = −0b1.0001×2^2 = -4.25
0xe2 = 1_110_0010 = −0b1.0010×2^2 = -4.5
0xe3 = 1_110_0011 = −0b1.0011×2^2 = -4.75
0xe4 = 1_110_0100 = −0b1.0100×2^2 = -5.0
0xe5 = 1_110_0101 = −0b1.0101×2^2 = -5.25
0xe6 = 1_110_0110 = −0b1.0110×2^2 = -5.5
0xe7 = 1_110_0111 = −0b1.0111×2^2 = -5.75
0xe8 = 1_110_1000 = −0b1.1000×2^2 = -6.0
0xe9 = 1_110_1001 = −0b1.1001×2^2 = -6.25
0xea = 1_110_1010 = −0b1.1010×2^2 = -6.5
0xeb = 1_110_1011 = −0b1.1011×2^2 = -6.75
0xec = 1_110_1100 = −0b1.1100×2^2 = -7.0
0xed = 1_110_1101 = −0b1.1101×2^2 = -7.25
0xee = 1_110_1110 = −0b1.1110×2^2 = -7.5
0xef = 1_110_1111 = −0b1.1111×2^2 = -7.75
0xf0 = 1_111_0000 = −0b1.0000×2^3 = -8.0
0xf1 = 1_111_0001 = −0b1.0001×2^3 = -8.5
0xf2 = 1_111_0010 = −0b1.0010×2^3 = -9.0
0xf3 = 1_111_0011 = −0b1.0011×2^3 = -9.5
0xf4 = 1_111_0100 = −0b1.0100×2^3 = -10.0
0xf5 = 1_111_0101 = −0b1.0101×2^3 = -10.5
0xf6 = 1_111_0110 = −0b1.0110×2^3 = -11.0
0xf7 = 1_111_0111 = −0b1.0111×2^3 = -11.5
0xf8 = 1_111_1000 = −0b1.1000×2^3 = -12.0
0xf9 = 1_111_1001 = −0b1.1001×2^3 = -12.5
0xfa = 1_111_1010 = −0b1.1010×2^3 = -13.0
0xfb = 1_111_1011 = −0b1.1011×2^3 = -13.5
0xfc = 1_111_1100 = −0b1.1100×2^3 = -14.0
0xfd = 1_111_1101 = −0b1.1101×2^3 = -14.5
0xfe = 1_111_1110 = −0b1.1110×2^3 = -15.0
0xff = 1_111_1111 = −Inf

References

[1] PyTorch authors. Pytorch torchtext package: _t5_multi_head_attention_forward .

[2] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*, chapter 6.2.2.3 Softmax Units for Multinoulli Output Distributions, pages 180–184. MIT Press, 2016.

[3] Google. Jax lax package: _float_to_int_for_sort .

[4] W. Kahan. Branch cuts for complex elementary functions or much ado about nothing's sign bit. *Inst. Math. Appl. Conf. Ser. New Ser.*, 1987.

[5] W. Kahan and J. W. Thomas. Augmenting a programming language with complex arithmetic. Technical report, EECS Department, University of California, Berkeley, 1991.

[6] P. Micikevicius, S. Oberman, P. Dubey, M. Cornea, A. Rodriguez, I. Bratt, R. Grisenthwaite, N. Jouppi, C. Chou, A. Huffman, M. Schulte, R. Wittig, D. Jani, and S. Deng. OCP 8-bit floating point specification (OFP8). Technical report, opencompute.org, 2023.

[7] B. Noune, P. Jones, D. Justus, D. Masters, and C. Luschi. 8-bit numerical formats for deep neural networks. Technical report, arXiv cs.LG, 2022.

[8] Tesla, Inc. Tesla Dojo Technology: A guide to Tesla's configurable floating point formats and arithmetic, 2023. `https://web.archive.org/web/20230503235751/https://tesla-cdn.thron.com/static/MXMU3S_tesla-dojo-technology_1WDVZN.pdf`.