hw0306

All reference are form ISO/IEC 9899:201x

float <-> int32_t

6.3.1.4 Real floating and integer

> 1.When a finite value of real floating type is converted to an integer type other than _Bool, the fractional part is discarded (i.e., the value is truncated toward zero). If the value of the integral part cannot be represented by the integer type, the behavior is undefined.

> 2.When a value of integer type is converted to a real floating type, if the value being converted can be represented exactly in the new type, it is unchanged. If the value being converted is in the range of values that can be represented but cannot be represented exactly, the result is either the nearest higher or nearest lower representable value, chosen in an implementation-defined manner. If the value being converted is outside the range of values that can be represented, the behavior is undefined. Results of some implicit conversions may be represented in greater range and precision than that required by the new type (see 6.3.1.8 and 6.8.6.4)

IEC 60559 floating-point arithmetic F.4 Floating to integer conversion

> If the integer type is _Bool, 6.3.1.2 applies and no floating-point exceptions are raised (even for NaN). Otherwise, if the floating value is infinite or NaN or if the integral part of the floating value exceeds the range of the integer type, then the ‘‘invalid’’ floating point exception is raised and the resulting value is unspecified. Otherwise, the resulting value is determined by 6.3.1.4. Conversion of an integral floating value that does not exceed the range of the integer type raises no floating-point exceptions; whether conversion of a non-integral floating value raises the inexact floating-point exception is unspecified.

float -> int32_t

- If the value of rhe integral part can be represented by integer32 type, the fractional part is discarded, if not, the behavior is undefined.

- If the floating value is infinite or NaN or if the integral part of the floating value exceeds the range of the integer type, then the invalid floating point exception is raised and the resulting value is unspecified.

example

1111.1111 -> 1111

54321.12345 -> 54321

NaN & ntegral part of the floating value exceeds -> unspecified

int32_t -> float

- If the integer can be represented by integral part of float, the intergral will be presented by integral part of float directly, if not, the behavior is undefined.

example

1111 -> 1111.000000

3402823467 -> undefined

---

int32_t <-> uint32_t

6.3.1.3 Signed and unsigned integers

> 1 When a value with integer type is converted to another integer type other than _Bool, if the value can be represented by the new type, it is unchanged. 2 Otherwise, if the new type is unsigned, the value is converted by repeatedly adding or subtracting one more than the maximum value that can be represented in the new type until the value is in the range of the new type.60) 3 Otherwise, the new type is signed and the value cannot be represented in it; either the result is implementation-defined or an implementation-defined signal is raised

int32_t -> uint32_t

- If the value can be represented by the new type, it is unchanged
- if the new type is unsigned, the value is converted by repeatedly adding or subtracting one more than the maximum value that can be represented in the new type

example

1111 -> 1111

-1000 -> 4294966296

uint32_t -> int32_t

- he new type is signed and the value cannot be represented in it; either the result is implementation-defined or an implementation-defined signal is raised

example

1111 -> 1111

4294967294 -> -2

double <-> float

6.3.1.5 Real floating types

> When a value of real floating type is converted to a real floating type, if the value being converted can be represented exactly in the new type, it is unchanged. If the value being converted is in the range of values that can be represented but cannot be represented exactly, the result is either the

> nearest higher or nearest lower representable value, chosen in an implementation-defined manner. If the value being converted is outside the range of values that can be represented, the behavior is undefined. Results of some implicit conversions may be represented in greater range and precision than that required by the new type (see 6.3.1.8 and 6.8.6.4)

double -> float

- When a value of real floating type is converted to a real floating type, if the value being converted can be represented exactly in the new type, it is unchanged

- If the value being converted is in the range of values that can be represented but cannot be represented exactly, the result is either the nearest higher or nearest lower representable value, chosen in an implementation-defined manner.

exmaple

1111.1111 -> 1111.1111

1111111111111.1111111111 -> NaN

float -> double

*When a value of real floating type is converted to a real floating type, if the value being converted can be represented exactly in the new type, it is unchanged

111111.11111 -> 111111.11111