A mathematical description of C

Formalin (CH₂O) project
PhD of Robbert Krebbers

Formal semantics of (large subset of) C in Coq

Coq = proof assistant
  = interactive theorem prover
  = mathematical programming language
A mathematical description of C

Formalin (CH₂O) project
PhD of Robbert Krebbers

Formal semantics of (large subset of) C in Coq

Coq = proof assistant
    = interactive theorem prover
    = mathematical programming language

C11 is inconsistent on a very fundamental level
Defect Report #260

Formalin deviates from C11
Many more undefined behaviors
Three kinds of bits in the Formalin semantics

zero bit = BBit false
one bit = BBit true
indeterminate bit = BIndet
Does this have to print the same number twice?

```c
int i; // i intentionally uninitialized

printf("%d\n", i);
printf("%d\n", i);
```
Does this have to print the same number twice?

unsigned char i; // i intentionally uninitialzed
               // i cannot contain a trap value (6.2.6.1/3)

printf("%d\n", i);
printf("%d\n", i);
printf("%d\n", i);
Does this have to print the same number twice?

unsigned char i;  // i intentionally unitialized
                  // i cannot contain a trap value (6.2.6.1/3)
&i;              // i is not in a register (6.3.2.1/2)

printf("%d\n", i);
printf("%d\n", i);
unsigned char i; // i intentionally uninitialize
    // i cannot contain a trap value (6.2.6.1/3)
&i; // i is not in a register (6.3.2.1/2)
i = i; // i now has a ‘last-stored’ value (6.2.4/2)
printf("%d\n", i);
printf("%d\n", i);
Does this have to print the same number twice?

```c
int32_t i;     // i intentionally uninitialised
               // i cannot contain a trap value (7.20.1.1)
&i;           // i is not in a register (6.3.2.1/2)
i = i;        // i now has a ‘last-stored’ value (6.2.4/2)
printf("%"PRId32"\n", i);
printf("%"PRId32"\n", i);
```
Does this have to print the same number twice?

unsigned char i;  // i intentionally uninitialised

&i;              // i cannot contain a trap value (6.2.6.1/3)
i = i;            // i is not in a register (6.3.2.1/2)
printf("%d\n", i); // i now has a ‘last-stored’ value (6.2.4/2)
printf("%d\n", i);
Defect Report #260

Question
(2001-09-07)

*If an object holds an indeterminate value, can that value change other than by an explicit action of the program?*
Question
(2001-09-07)

If an object holds an indeterminate value, can that value change other than by an explicit action of the program?

Answer
(2003-03-06)

An object with indeterminate value has a bit pattern representation which remains constant during its lifetime.
Question
(2001-09-07)

*If an object holds an indeterminate value, can that value change other than by an explicit action of the program?*

Answer
(2003-03-06)

*An object with indeterminate value has a bit pattern representation which remains constant during its lifetime.*

Answer
(2004-09-28)

*In the case of an indeterminate value [...] the actual bit-pattern may change without direct action of the program.*
Status of Defect Report #260

- Decided no change to the standard text was needed
- Defect report about C99
- Superseded by C11
- All relevant text in C11 identical to the same text in C99
What does the standard say?

(6.2.4/2)

An object [...] retains its last-stored value throughout its lifetime.

(6.7.9/10)

If an object that has automatic storage duration is not initialized explicitly, its value is indeterminate.
Indeterminate versus unspecified values?

For types without trap representations:

indeterminate value = unspecified value
Indeterminate versus unspecified values?

For types without trap representations:

\[ \text{indeterminate value} = \text{unspecified value} \]

(3.19.1+3.19.2)

**indeterminate value**

either an unspecified value or a trap representation

**unspecified value**

[...]

*NOTE* An unspecified value cannot be a trap representation.
void printhex(int d) {
    putchar(d < 10 ? '0' + d : 'A' + d - 10);
}

void printbyte(int i) {
    printhex(i>>4); printhex(i&0xf);
}
void printhex(int d) {
    putchar(d < 10 ? '0' + d : 'A' + d - 10);
}

void printbyte(int i) {
    printhex(i>>4); printhex(i&0xf);
}

struct foo {
    short x1;
    /* padding */
    int x2;
};
Our recommendation for a resolution

- Revert decision of Defect Report #260
- Indeterminate data in a non-volatile object *can not change* without an explicit action of the program
- No change to the standard text is needed
Contents

Our Project

The Issue

Defect Report #260

The Standard

Discussion