Bitwise Operators, Bit Manipulation, Testing Bits

TECH 3230 – Lecture 3

Prof Kohn

Presentation created with the help of ChatGDP

THE UNIVERSITY OF MEMPHIS.

1

Bitwise Operators in C

- Used to manipulate data at the bit level.
- Common in embedded systems and performance-critical code.
- True/False test for IF and Loop structures and how they really work.

THE UNIVERSITY OF MEMPHIS.

2

Binary Basics

- Computers use binary (0s and 1s).
- Each bit represents a power of 2.
- Example: 5 = 00000101

THE UNIVERSITY OF MEMPHIS.

3

Bitwise AND (&)

- Compares bits: 1 & 1 = 1, otherwise 0.
- Example:
- 5 & 3 = 1
- (0101 & 0011 = 0001)

4

Bitwise OR (|)

- Compares bits: 1 if either bit is 1.
- Example:
- 5 | 3 = 7
- (0101 | 0011 = 0111)

5

Bitwise XOR (^)

- 1 if bits are different.
- Example:
- 5 ^ 3 = 6
- (0101 ^ 0011 = 0110)

6

Bitwise NOT (~)

- Flips all bits.
- Example:
- ~5 = -6 (two's complement representation)

7

Left Shift (<<)

- Shifts bits left, adds zeros on right.
- Example:
- 5 << 1 = 10

8

Right Shift (>>)

- Shifts bits right.
- Example:
- 5 >> 1 = 2

9

Bit Manipulation Techniques

- Set a bit: x | (1 << n)
- Clear a bit: x & ~(1 << n)
- Toggle a bit: x ^ (1 << n)

10

Checking a Bit

- Check if nth bit is set:
- if (x & (1 << n))

11

Real-World Applications

- Embedded systems
- Flags and masks
- Data compression
- Cryptography

12

Example Code

```
#include <stdio.h>

// program starts with x=5 (b00000101) and ANDs with 3 (b00000011) and prints answer of 1 (b00000001)

int main()
{
    int x = 5;
    printf("%d", x & 3);
    return 0;
}
```

THE UNIVERSITY OF MEMPHIS

13

Bitwise vs. Boolean Operators

in ANSI C

`&` vs `&&`
`|` vs `||`

Two very different operators that look almost identical

THE UNIVERSITY OF MEMPHIS

14

Bitwise AND `&` vs Logical AND `&&`

`&` Bitwise AND (&)

Compares operands **bit by bit**. Works on the binary representation of integers.

```
unsigned char a = 6;   /* 0000 0110 */
unsigned char b = 3;   /* 0000 0011 */

unsigned char r = a & b;
/* r = 0000 0010 = 2 */

/* common use: masking bits */
flags = flags & 0x0F;
if (perm & READ_BIT) {
    /* bit is set */
}
```

`&&` Logical AND (&&)

Compares the **truth value** of two expressions. Result is always 0 or 1. **Short-circuits**: right side isn't evaluated if the left is false.

```
int x = 6, y = 3;

if (x > 0 && y > 0) {
    printf("both positive\n");
}

/* short-circuit avoids crash: */
if (p != NULL && p->val > 0) {
    /* safe: p->val not read
       unless p != NULL */
}
```

& operates on bits and returns an integer • && operates on conditions and returns 0 or 1

THE UNIVERSITY OF MEMPHIS

15

Bitwise OR | vs Logical OR ||

| Bitwise OR (|)

Sets each result bit to 1 if **either** operand has that bit set. Used to combine bit flags.

```
unsigned char a = 4;   /* 0000 0100 */
unsigned char b = 1;   /* 0000 0001 */

unsigned char r = a | b;
/* r = 0000 0101 = 5 */

/* common use: setting flags */
#define READ_BIT  0x01
#define WRITE_BIT 0x02
perm = perm | WRITE_BIT;
```

|| Logical OR (||)

True if **either** condition is true. Result is 0 or 1. **Short-circuits**: stops as soon as one side is true.

```
int x = 0, y = 5;

if (x > 0 || y > 0) {
    printf("at least one\n");
}

/* short-circuit skips call: */
if (cached || compute_slow()) {
    /* compute_slow() only runs
       if cached is false */
}
```

| combines bit patterns into an integer • || combines conditions into 0 or 1

THE UNIVERSITY OF MEMPHIS

16

ANSI C • CONTROL FLOW

0 Zero is False. Everything Else is True. 1

How if, while, and for actually test their conditions in C

```
if (x) { /* runs when x != 0 */ }
```

Intro to Programming | C Language Fundamentals

THE UNIVERSITY OF MEMPHIS

17

THE CORE RULE

In classic C, every condition is really just an integer. The machine only asks one question:

0

FALSE

Exactly one value counts as false: the integer 0 (also written as a null pointer or '\0').

≠0

TRUE

Every other value is true — including negative numbers: 1, -1, 42, -7, 0.001 (as an int), all true.

THE UNIVERSITY OF MEMPHIS

18

WHY THIS IS TRUE

Other languages have a dedicated Boolean type. ANSCII C does not — **`if`**, **`while`**, and **`for`** all evaluate their condition as an **`int`** and branch on whether it is zero.

01 x > 5 evaluates to the int 1 (true) or 0 (false).

02 &&, ||, and ! also produce plain 1 or 0 values.

03 #include <stdbool.h> gives you bool, true, and false — but they're just 1 and 0 underneath.

bool_demo.c

```c
#include <stdbool.h>
#include <stdio.h>

int main(void) {
    bool done = false;
    int  tries = 0;

    while (!done) {
        tries++;
        if (tries == 3)
            done = true;  // nonzero
    }
    return 0;
}
```


THE UNIVERSITY OF MEMPHIS

19

EXAMPLE 1

if_demo.c

```c
#include <stdio.h>

int main(void) {
    int score = 5;

    if (score) {        // score != 0
        printf("Nonzero -> TRUE\n");
    } else {
        printf("Zero -> FALSE\n");
    }

    if (score - 5) {  // 0
        printf("Nonzero -> TRUE\n");
    } else {
        printf("Zero -> FALSE\n");
    }
    return 0;
}
```

PROGRAM OUTPUT

```
Nonzero -> TRUE
Zero -> FALSE
```

Reading it like a C compiler

1. score holds 5 — a plain int, not a Boolean.
2. if (score) asks: "Is this expression's value 0?"
3. 5 is nonzero, so the condition is TRUE and the first branch runs.
4. score - 5 evaluates to 0, so that condition is FALSE.

THE UNIVERSITY OF MEMPHIS

20

EXAMPLE 2

countdown.c

```c
#include <stdio.h>

int main(void) {
    int n = 3;

    while (n) {         // loop while n != 0
        printf("%d...\n", n);
        n--;
    }
    printf("Liftoff! (n is now %d)\n", n);

    return 0;
}
```

n	while (n)	Action
3	TRUE	print 3, n--
2	TRUE	print 2, n--
1	TRUE	print 1, n--
0	FALSE	loop stops

The loop re-checks n every pass. The moment n becomes 0, the condition is false and the loop exits — no explicit comparison needed.

THE UNIVERSITY OF MEMPHIS

21

EXAMPLE 3

A for loop's middle clause is exactly the same truth test as if and while — it just runs before every pass.

```
for ( i = 0 ; i < 5 ; i++ )
```

1 init — runs once 2 test — checked before every pass (0 = stop, nonzero = go) 3 update — runs after each pass

for_demo.c

```
for (int i = 0; i < 5; i++) {
    printf("%d squared = %d\n", i, i*i);
}

// test becomes 0 (false) once i reaches 5
// -> loop body runs for i = 0,1,2,3,4
```

Key Idea

`i < 5` is a comparison expression. C evaluates it to the `int` 1 while it holds and 0 the instant it doesn't — the for loop is just feeding that number into the same zero/nonzero test as if and while.

6

THE UNIVERSITY OF MEMPHIS

22

REFERENCE

Expression	Evaluates To	Verdict
5	5	TRUE
0	0	FALSE
-3	-3	TRUE
x == 10	1 or 0	depends on x
x != 10	1 or 0	depends on x
a && b	1 or 0	TRUE only if both nonzero
a || b	1 or 0	TRUE if either nonzero
!x	1 or 0	flips x's truth value
NULL pointer	0	FALSE

7

THE UNIVERSITY OF MEMPHIS

23

COMMON PITFALL

Because ANY nonzero value is true, a single = inside a condition still compiles — it just silently assigns instead of comparing, and the result is almost always true.

✗ BUGGY

flag = 0 assigns

```
int flag = 0;

if (flag = 5) {
    printf("Always runs!\n");
}
// flag is now 5 -- reassigned!
```

flag = 5 assigns 5 to flag, and the whole expression's value is 5 — nonzero, so it's always TRUE, no matter what flag used to hold.

✓ CORRECT

flag == 5 compares

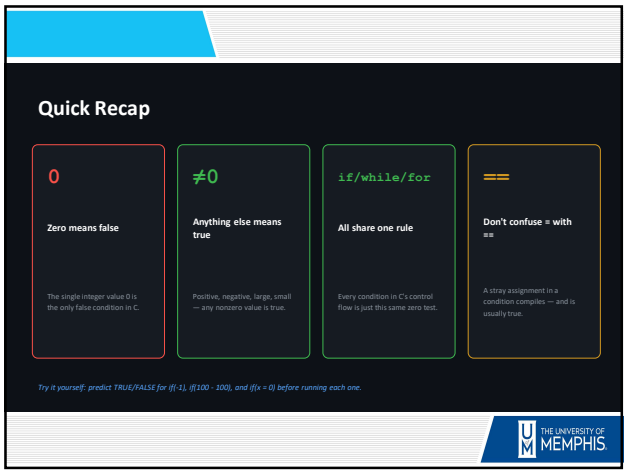
```
int flag = 0;

if (flag == 5) {
    printf("Runs only if flag is 5\n");
}
// flag is untouched -- still 0
```

flag == 5 compares flag to 5 and produces 1 (true) or 0 (false) without ever changing flag itself.

8

THE UNIVERSITY OF MEMPHIS

24


Quick Recap
0
Zero means false
The single integer value 0 is the only false condition in C.
≠0
Anything else means true
Positive, negative, large, small — any nonzero value is true.
if/while/for
All share one rule
Every condition in C's control flow is just this same zero test.
==
Don't confuse = with ==
A stray assignment in a condition compiles — and is usually true.
Try it yourself: predict TRUE/FALSE for if(-1), if(100 - 100), and if(x = 0) before running each one.
THE UNIVERSITY OF MEMPHIS

25