Bit-wise
- NOT
- AND - Isolate a bit; clear
- OR - Set
- XOR - Toggle

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>A XOR B</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ A \oplus 0 = A \]
\[ A \oplus 1 = \overline{A} \]

Shift
- >> - \approx 12
- << - \approx 32?
GPI0 Pin

if configured as input — only reads work

if configured as output — reads & writes work

PSR

N V C Z

Zero Indicator

-ve Indicator

Overflow

V: Signed Numbers
C: Unsigned

LDR R1, #1
SUBS R1, #1
ADD S R1, #10
ADD S R1, # -20
BEQ loc

-2^{31} \leftrightarrow 2^{31} - 1
volatile long delay;

delay = 100;

C Rules

#include
#define

All other statements
- Declarations
- Assignment

Blocks { }

main need to Run/Execute
Debugging

Aids
- Single-step
  - Watch
  - Register
  - Variable
  - Memory locations
- Breakpoints
- Step-over

1. Correctness
2. Timeliness
   - Functional

Logic Analyzer
- Oscilloscope
Variable Data Types

long delay; // 32-bit

c99 Standard

int32_t delay; // Signed 32-bit
uint32_t count; // Unsigned 32-bit
int16_t, uint16_t
int8_t, uint8_t
char
Condition

{ if (cond) }
{ while (cond) }
{ else }
{ }
{ }

Equals: ==
Not Equals: !=
Greater than: >
Less than: <
Greater than or equal to: >=
Less than or equal to: <=

Any statement whose result can be interpreted as True / False
Non-zero: 0