Embedded C Programming

Agenda

- Data Types
  - Signed vs Unsigned
  - volatile
- Bitwise Operators
  - &, |, ^, ~, <<, >>
  - Applications
- Structures
  - Internals of . and -> operator
  - Member offsets
  - Slack bytes

Data Types

- Bit -- Binary Digit -- 1 or 0
- Data Type -- Size, Endianness, Sign (+ve or -ve), Integer or Floating point.
- Integer types -- Size, Sign and Range
  - signed char -- 8 bits -- 1 (sign) + 7 (number) -- +/- 2$7 --> -128 to +127
  - unsigned char -- 8 bits -- 8 (number) -- + 2$8 --> 0 to 255
  - signed short -- 16 bits -- 1 (sign) + 15 (number) -- +/- 2$15 --> -32768 to +32767
  - unsigned short -- 16 bits -- 16 (number) -- + 2$16 --> 0 to 65535
  - int
  - longz
- Floating point types -- sign + expoent and mantissa
  - float -- 4 bytes -- IEEE-754 (32 bit)
  - double -- 8 bytes -- IEEE-754 (64 bit)
  - long double -- 16 bytes or 12 bytes
  - 12345.6789 = 1.23456789 * 10$4 = 1.23456789 e +4
  - https://en.wikipedia.org/wiki/IEEE_754

Positive vs Negative

- signed char -- 8 bits -- 1 (sign MSB) + 7 (number)
  - +5 = 0000 0101
  - -5 = 1000 0101 (wrong)
  - -5 = 2's complement form = ~5 + 1 = ~(0000 0101) + 1 = 1111 1010 + 1 = 1111 1011
  - +251 = 1111 1011
- -ve number bit pattern can be same as some +ve number.
  - e.g. 8 bits --> -5 is same as 251 (bit representation).
- For binary, octal and hexa-decimal there is no concept of -ve values.

- 2's complement = 1's complement + 1
    - -5 = ~5 + 1
- 1's complement = 2's complement - 1
    - ~5 = -5 - 1 = -6

Magic number

- Identification of the file format.
- It is first 2 or 4 bytes of binary files.
- Example: https://miro.medium.com/max/1400/1*l35WfTrEJpdQ3jYi5PLYjg.png
    - .exe --> MZ
    - .out --> ?ELF
    - .bmp --> BM
    - .jpg --> 0xFFD8FF
    - .wav --> RIFF
    - .mp3 --> ID3
- Magic number is added by the software/program that creates the file.
    - .bmp --> paint-brush
    - .exe --> Windows Linker
    - .out --> Linux Linker
- A software/program that reads the file, first read magic number to confirm if file is in proper format (or corrupted). If magic number is as expected, then only further file will be read and processed.
    - .bmp --> checked by paint-brush or any other image reading software.
    - .mp3 --> checked by media/music player.
    - .exe --> checked by Windows Loader.
    - .out --> checked by Linux Loader.

Bitwise Operator

Boolean algebra or Logic gates

- AND -- &
    - x & 0 = 0
    - x & 1 = x

```
0     0      0
0     1      0
1     0      0
1     1      1
```

    - 5 & 9 = 1

```

```

```
5 ->    0000 0101
9 ->    0000 1001
------------------
        0000 0001   -> 1
```

- OR -- |
  - x | 0 = x
  - x | 1 = 1

```
0     0      0
0     1      1
1     0      1
1     1      1
```

  - 5 | 9 = 13

```
5 ->    0000 0101
9 ->    0000 1001
------------------
        0000 1101   --> 13
```

- XOR -- ^
  - x ^ x = 0
  - x ^ x' = 1
  - x ^ 1 = x'

```
0     0      0
0     1      1
1     0      1
1     1      0
```

  - 5 ^ 9 = 12

```
5 ->    0000 0101
9 ->    0000 1001
------------------
        0000 1100   --> 12
```

- NOT -- ~
  - ~x = x'

```
0   1
1   0
```

- ~5 = ??? (8-bits)

```
5 ->    0000 0101
-------------------
~ ->    1111 1010   --> 250 (256-6) or -6
```

- Logical AND/OR/NOT vs Bitwise AND/OR/NOT
  - Logical operators work on condtions (false=0 or true=non-zero).
  - Bitwise operators work on bits of numbers.
  - Logical operator always result in 0 (false) or 1 (true).
  - Bitwise operator result can be any number (as per bit pattern).
  - 5 (0101) & 9 (1001) = 1
  - 5 (non-zero - true) && 9 (non-zero - true) = 1 (true)
  - 5 (0101) | 9 (1001) = 13
  - 5 (non-zero - true) || 9 (non-zero - true) = 1 (true)
  - ~5 (0000 0101) = -6 or 250 (8 bits)
  - !5 (non-zero - true) = 0 (false)