

1DT301 - C Programming

Lecture 4

Linnaeus University, 2017

Lars Karlsson

Structures
struct

Structure

- A grouped collection of variables
- Organize complicated data
- A.k.a a record

Example

- Represent a GPS coordinate
 - Latitude, longitude.



Name of structure

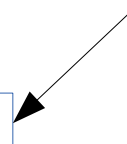
```
struct GPS_coord  
{  
    double lat;  
    double lon;  
};
```

Members

Example (cont'd)


Init (lat=12.43, lon=33.55)

```
struct GPS_coord gps1;  
struct GPS_coord gps2 = {12.43, 33.55};
```




access a member.

```
gps1.lat = 44.55;  
gps1.lon = 67.98;
```



```
printf("Lat=%f Lon=%f", gps1.lat, gps1.lon);
```

Pointer to struct.



A diagram with an arrow pointing from an exclamation mark (!) to a small blue square box that highlights the asterisk (*) in the code snippet below.

```
struct GPS_coord *p_gps = NULL;  
p_gps = &gps2;
```



A diagram with two arrows pointing to the code snippet below. One arrow points from the text 'No dereferencing, instead!' to a small blue square box highlighting 'p_gps'. The other arrow points to another small blue square box highlighting the arrow operator '->'.

```
p_gps->lat = 88.9;
```

No dereferencing, instead!

OR

```
(*p_gps).lat = 88.9;
```

typedef

- Creating new data types

```
typedef struct GPS_coord GPS;
```

```
GPS gps3;
```

GPS is a synonym for **struct** GPS_coord



Structures *union*

Union

- A type that holds objects of different types and sizes.
- Or, different viewpoints of *the same memory area*.

Example

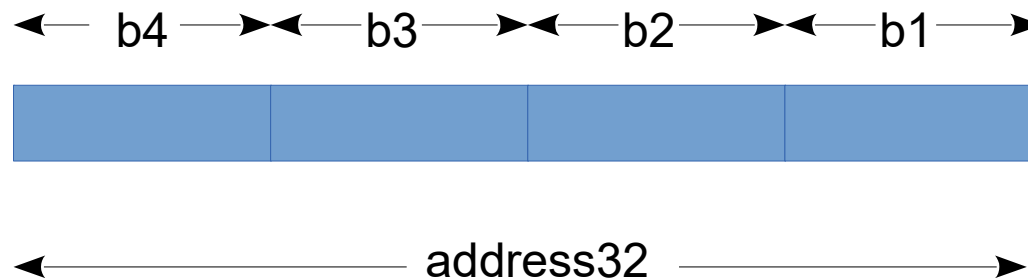
- IPv4 address¹
- Two views of the same data.
 - As int : 2886794753
 - As bytes : 172, 16, 254, 1

An IPv4 address (dotted-decimal notation)

172 . 16 . 254 . 1
↓ ↓ ↓ ↓
10101100,00010000,11111110,00000001
└───┘ └───┘
One byte = Eight bits
└──────────────────────────┘
Thirty-two bits (4 x 8), or 4 bytes

¹http://en.wikipedia.org/wiki/IP_address

Example (cont'd)



```
struct dotted
{
    unsigned char b1;
    unsigned char b2;
    unsigned char b3;
    unsigned char b4;
};

union ip_address
{
    int address32;
    struct dotted byte;
};
```

Example (cont'd)

```
union ip_address ip = {0};  
  
ip.address32 = 2886794753U;  
printf("b1=%u,b2=%u,b3=%u,b4=%u",  
       ip.byte.b1, ip.byte.b2, ip.byte.b3, ip.byte.b4);  
  
ip.byte.b2 = 255;  
printf("Address32=%u", ip.address32);
```

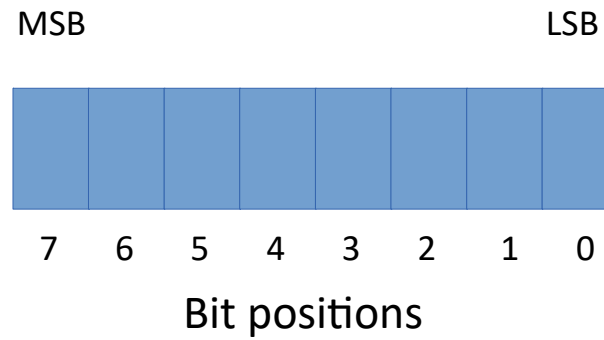


Structures
bitfields

Bitfield

- Using fragments of integers
 - e.g. use a single bit as a flag (boolean)
- bit-oriented protocols
- When memory is expensive

Bits



Note! MSB and LSB may be reversed!
See the concept of endianness.

Example

- Represent a byte's high and low nibble
 - A nibble consists of 4 bits.

name of bitfields

```
struct
{
    unsigned int lo:4;
    unsigned int hi:4;
} byte;
```

size in bits

The diagram illustrates the bitfield definitions in the struct. The text 'name of bitfields' has an arrow pointing to the variable names 'lo' and 'hi' in the declarations 'unsigned int lo:4;' and 'unsigned int hi:4;'. The text 'size in bits' has an arrow pointing to the ':4' values in the same declarations. The variable names 'lo' and 'hi' are highlighted with blue boxes in the original image.

Headers

- Contains declarations
 - So there need to be definitions somewhere!
- *AKA Include files.*
- Standard libraries (need .lib-files!)
- Modularisation¹
- Abstract Data Types¹

¹ Not in this course.

Headers

- File (.h)
- Referred to by using `#include`
- `#include <stdio.h>` \leftrightarrow `#include "stdio.h"`
- Custom made header files.
 - Typically `#include "filename.h"`
- Problem with circular includes.
 - `#ifndef` + `#endif`

Some standard header files

Include	Content
assert.h	Diagnostics.
math.h	Mathematical functions and macros
stdio.h	Input and output functions and macros.
stdlib.h	Number conversions, storage allocations etc..
string.h	String handling.
time.h	Manipulating time and date.

Some examples

Example

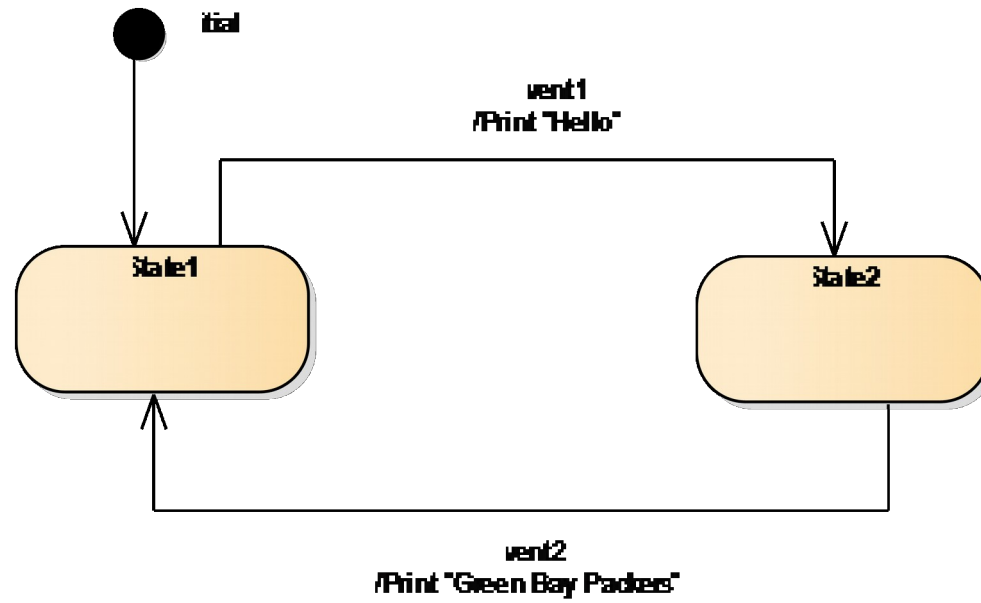
- Create a program which allows a user to enter an arbitrary number of floats.

The program shall calculate the average using a function and print the entered numbers.

Bit fiddling example

- In a project You have two values that both have their value range from 0-15. Find a way to store these in as little space as possible (not using bit fields!). Also, it should be possible to assign and retrieve these.

State machine example



Header example

- Create a module (= .h and .c file) where there are two math functions for adding and subtracting two integer values.
- Use that module.

(This is a simple example)