嵌入式C语言之-

结构体数据内存对齐与紧缩方法

讲师：叶大鹏

应用案例

```
typedef struct
{
    uint8_t id;
    uint8_t humi;
    float temp;
} TempHumiSensor;
```

sizeof(TempHumiSensor) = ?

```
typedef struct
{
    uint8_t id;
    float temp;
    uint8_t humi;
} TempHumiSensor;
```

sizeof(TempHumiSensor) = ?

```
typedef struct
{
    uint8_t id;
    uint8_t humi;
    float temp;
} TempHumiSensor;
```
sizeof(TempHumiSensor) = 8

```
typedef struct
{
    uint8_t id;
    float temp;
    uint8_t humi;
} TempHumiSensor;
```
sizeof(TempHumiSensor) = 12

```
typedef struct
{
    uint8_t id;
    uint8_t humi;
    double temp;
} TempHumiSensor;
```
sizeof(TempHumiSensor) = 16

```
typedef struct
{
    uint8_t id;
    double temp;
    uint8_t humi;
} TempHumiSensor;
```
sizeof(TempHumiSensor) = 24

```
typedef struct
{
    uint8_t id;
    uint8_t humi1;
    float temp;
    uint8_t humi2;
} TempHumiSensor;
```

sizeof(TempHumiSensor) = 12

```
typedef struct
{
    uint8_t id;
    uint8_t humi1;
    double temp;
    uint8_t humi2;
} TempHumiSensor;
```

sizeof(TempHumiSensor) = 24

```
typedef struct
{
    uint8_t id;
    uint8_t humi;
    float temp;
} TempHumiSensor;

typedef struct
{
    uint8_t  co2Level;
    TempHumiSensor tempHumiData;
    uint8_t  pm25Level;
} AirQuality;
```

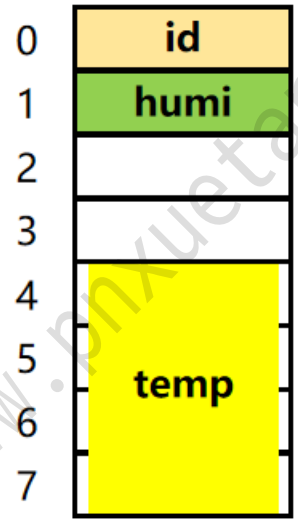
sizeof(AirQuality) = 16

```
typedef struct
{
    uint8_t id;
    uint8_t humi;
    double temp;
} TempHumiSensor;

typedef struct
{
    uint8_t  co2Level;
    TempHumiSensor tempHumiData;
    uint8_t  pm25Level;
} AirQuality;
```

sizeof(AirQuality) = 32

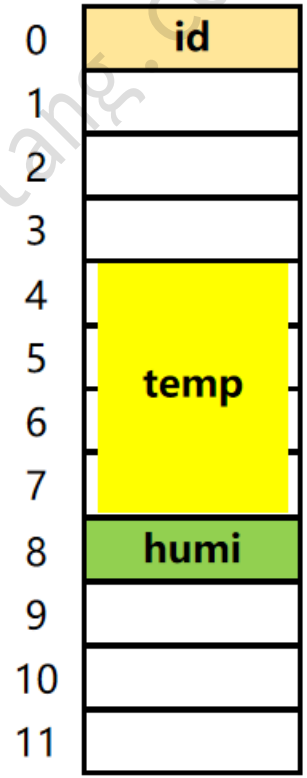
```
typedef struct
{
    uint8_t id;
    uint8_t humi;
    float temp;
} TempHumiSensor;
sizeof(TempHumiSensor) = 8
```



1. 第一个成员在与结构体变量起始地址偏移量为0的地址处；

2. 其他成员要对齐到成员自身大小（对齐数）的整数倍的地址处；

```
typedef struct
{
    uint8_t id;
    float temp;
    uint8_t humi;
} TempHumiSensor;
sizeof(TempHumiSensor) = 12
```



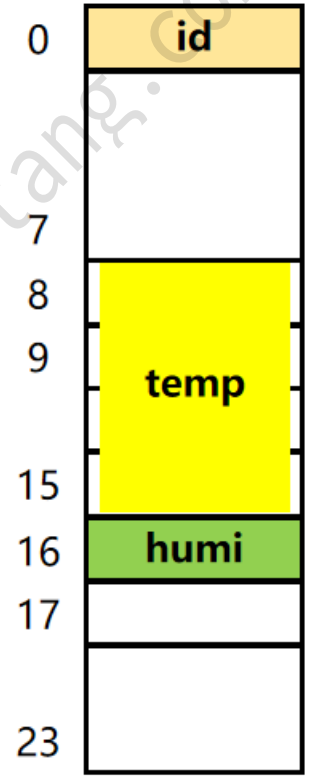
1. 第一个成员在与结构体变量起始地址偏移量为0的地址处；

2. 其他成员要对齐到成员自身大小（对齐数）的整数倍的地址处；

3. 结构体最大对齐数，指的是所有成员中最大的对齐数值；

4. 结构体总大小为结构体最大对齐数的整数倍；

```
typedef struct
{
    uint8_t id;
    double temp;
    uint8_t humi;
} TempHumiSensor;
sizeof(TempHumiSensor) = 24
```



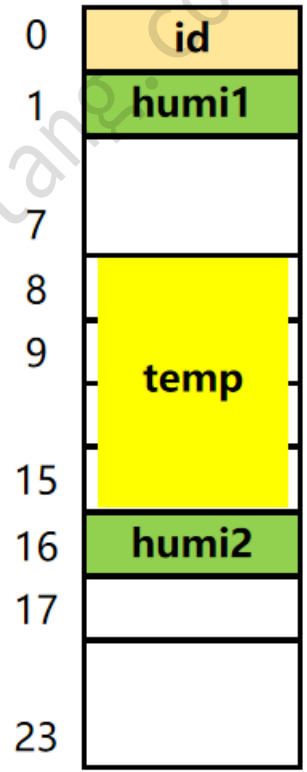
1. 第一个成员在与结构体变量起始地址偏移量为0的地址处；

2. 其他成员要对齐到成员自身大小（对齐数）的整数倍的地址处；

3. 结构体最大对齐数，指的是所有成员中最大的对齐数值；

4. 结构体总大小为结构体最大对齐数的整数倍；

```
typedef struct
{
    uint8_t id;
    uint8_t humi1;
    double temp;
    uint8_t humi2;
} TempHumiSensor;
sizeof(TempHumiSensor) = 24
```



1. 第一个成员在与结构体变量起始地址偏移量为0的地址处；

2. 其他成员要对齐到成员自身大小（对齐数）的整数倍的地址处；

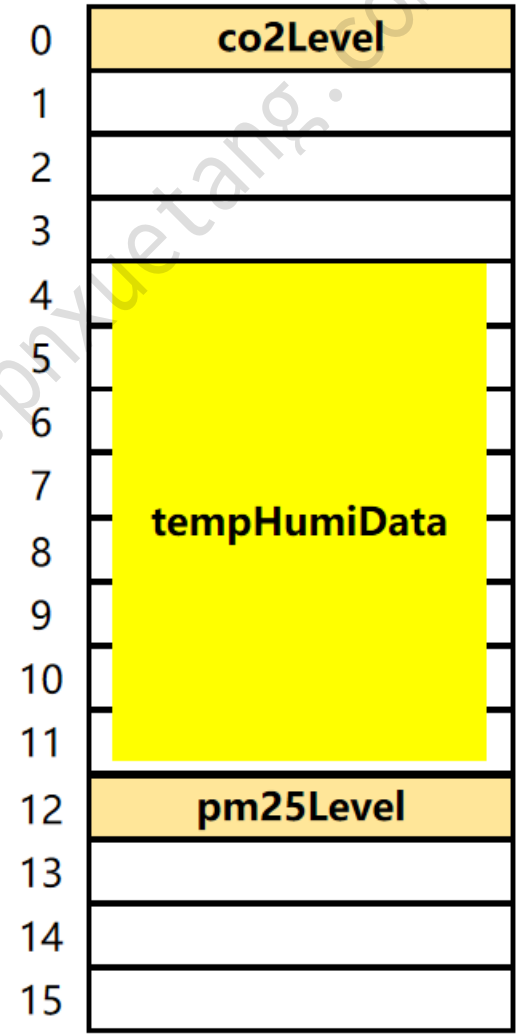
3. 结构体最大对齐数，指的是所有成员中最大的对齐数值；

4. 结构体总大小为结构体最大对齐数的整数倍；

```
typedef struct
{
    uint8_t id;
    uint8_t humi;
    float temp;
} TempHumiSensor;

typedef struct
{
    uint8_t  co2Level;
    TempHumiSensor tempHumiData;
    uint8_t  pm25Level;
} AirQuality;
```

sizeof(AirQuality) = 16



内存分布对齐规则

1. 第一个成员在与结构体变量起始地址偏移量为0的地址处；

2. 其他成员要对齐到成员自身大小（对齐数）的整数倍的地址处；

3. 结构体最大对齐数，指的是所有成员中最大的对齐数值；

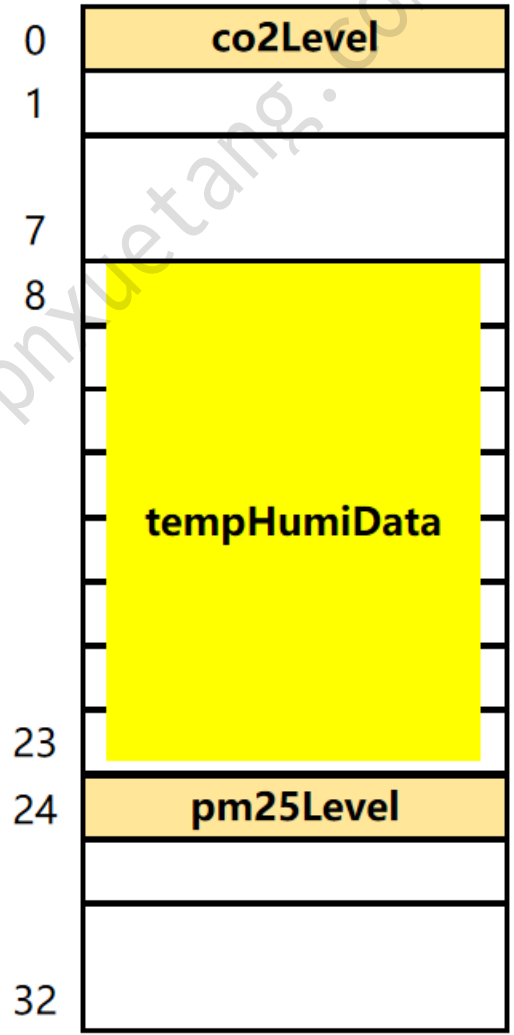
4. 结构体总大小为结构体最大对齐数的整数倍；

5. 如果结构体1嵌套了结构体2的情况，嵌套的结构体2对齐到自己的最大对齐数的整数倍处，结构体1的整体大小就是最大对齐数的整数倍。

```
typedef struct
{
    uint8_t id;
    uint8_t humi;
    double temp;
} TempHumiSensor;

typedef struct
{
    uint8_t  co2Level;
    TempHumiSensor tempHumiData;
    uint8_t  pm25Level;
} AirQuality;
```

sizeof(AirQuality) = 32



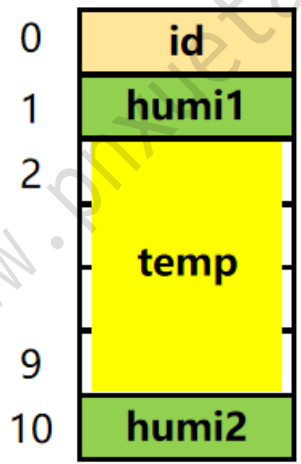
PN学堂

内存分布对齐规则

- ✓ 结构体中的成员时最好按照占用字节数从小到大依此排列，可以使得结构体占用空间最小。

设置紧缩排列1

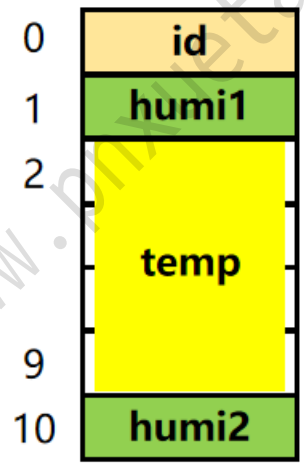
```
typedef  __packed struct
{
    uint8_t id;
    uint8_t humi1;
    double temp;
    uint8_t humi2;
} TempHumiSensor;
sizeof(TempHumiSensor) = 11
```



设置紧缩排列2

```
#pragma  pack  (1)
typedef  struct
{
    uint8_t id;
    uint8_t humi1;
    double temp;
    uint8_t humi2;
} TempHumiSensor;
#pragma pack()

sizeof(TempHumiSensor) = 11
```



嵌入式C语言之-

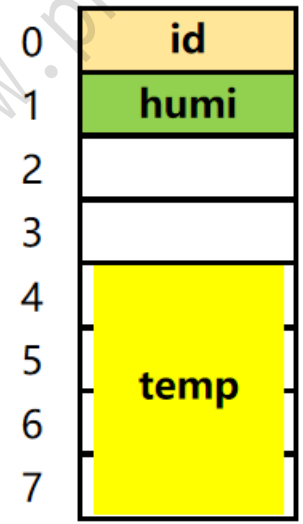
结构体数据为什么要内存对齐

讲师：叶大鹏

结构体为什么要内存对齐

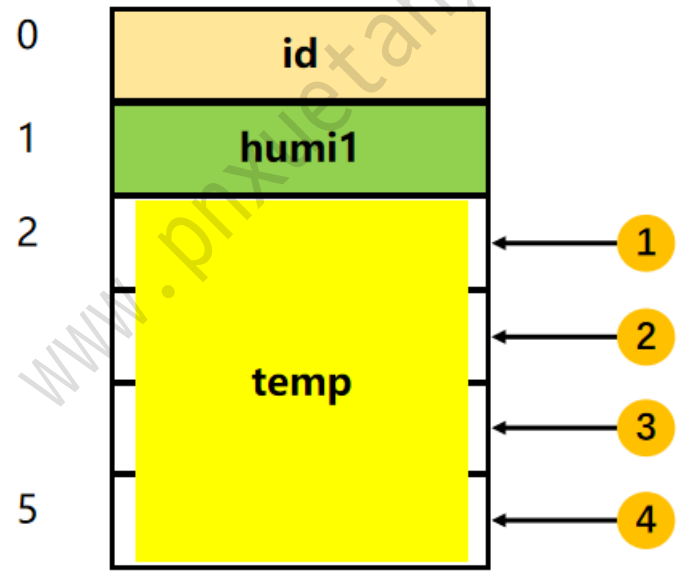
- 对于某些硬件平台，当访问int32_t、float、double这些类型的成员时，4字节访问数据的代码指令只能按照4字节对齐访问，也就是成员地址必须是4的整数倍。

```
typedef struct
{
    uint8_t id;
    uint8_t humi;
    float temp;
} TempHumiSensor;
sizeof(TempHumiSensor) = 8
```



结构体为什么要内存对齐

```
typedef __packed struct
{
    uint8_t id;
    uint8_t humi1;
    float temp;
} TempHumiSensor;
sizeof(TempHumiSensor) = 6
```



- 如果内存不按照4字节对齐，要访问int32_t、float、double这些类型的成员时，只能按照1字节/2字节访问多次的方式，这样会导致CPU效率和性能降低；
- 采用内存对齐，实际上就是用空间换取时间。

ARM单片机支持非内存对齐访问

```
typedef struct
{
    uint8_t id;
    uint8_t humi;
    float temp;
} TempHumiSensor;
int main()
{
    volatile TempHumiSensor temHumiData;
    temHumiData.humi = 20;
    temHumiData.temp = 20.5f;
    return 0;
}
```

```
R13 (SP)          0x200003FC
```

```
main
    0x000004d0:   b50c      ..    PUSH    {r2,r3,lr}
    0x000004d2:   2014      .     MOVS    r0,#0x14
    0x000004d4:   f88d0001  ....  STRB    r0,[sp,#1]
    0x000004d8:   4801      .H    LDR     r0,[pc,#4] ;
    0x000004da:   9001      ..    STR     r0,[sp,#4]
    0x000004dc:   2000      .     MOVS    r0,#0
    0x000004de:   bd0c      ..    POP     {r2,r3,pc}
```

ARM单片机支持非内存对齐访问

```
typedef __packed struct
{
    uint8_t id;
    uint8_t humi;
    float temp;
} TempHumiSensor;
int main()
{
    volatile TempHumiSensor temHumiData;
    temHumiData.humi = 20;
    temHumiData.temp = 20.5f;
    return 0;
}
```

```
R13 (SP)          0x200003FC
```

```
main
    0x000004d0:   b50c      ..    PUSH    {r2,r3,lr}
    0x000004d2:   2014      .     MOVS    r0,#0x14
    0x000004d4:   f88d0001  ....  STRB    r0,[sp,#1]
    0x000004d8:   4802      .H    LDR     r0,[pc,#8] ;
    0x000004da:   f8cd0002  ....  STR     r0,[sp,#2]
    0x000004de:   2000      .     MOVS    r0,#0
    0x000004e0:   bd0c      ..    POP     {r2,r3,pc}
```

- 0x200003FC + 2 = 0x200003FE(十进制536,871,934)，并不能被4整除，ARM单片机支持内存非对齐访问。

THANK YOU!