Review of C-Programming: Structure, Enum

Embedded Controller

Young-Keun Kim

mod .2024.08.26

Structure-Basics

Structure

- **Structure type**
  - We can define our own data type
  - A set of related field members
  - Each field member can be defined with a different data type
  - Structure declaration and definition
    - Structure variables, Tagged structures, Type-defined structures

Example 1: Vector

Structure define
(typedef structures)

```
typedef struct {
        int32_t * val;
        unint16_t rows;
        unint32_t norm;
}Vector;
```

Field member

Structure name

Examle 2:
Matrix Structure

```
typedef struct {
        double** at;
        int rows;
        int cols;
}Matrix;
```

Structure

- **Example code**

```
typedef struct {
        uint16_t sec;
        uint16_t min;
        uint16_t hour;
} TIME_TypeDef;
```

```
//variable with position_t type.
TIME_TypeDef time;

time.hour=18;
time.min=20;
time.sec=01;
```

```
// Also, we can define pointers to structures
TIME_TypeDef *pTime;
pTime=&time;
pTime->hour=17;
pTime->min=20;
pTime->sec=10;
```

C_structure_example.c

Structure: Exercise

Exercise 2

C_structure_exercise2.c

- Define a structure member
- Structure
  - Typedef struct Handong
  - Members: char building_name[100], int room_number, char room_name[100];

- Create structure variables room1, room2, room3. Assign the member values as

	Building name	Room number	Room name
room1	Newton	109	SSS-LAB
room2	Newton	118	Control-Lab
room3	Newton	119	SW-Lab

- Create `room1Pt` as Pointer variable of Handong type `room1`
- Print each room names as follows

```
Newton building, room  109  is SSSLAB
Newton building, room  118  is Control-Lab
Newton building, room  119  is SW-Lab
Newton building, room  119  is SW-Lab

 room3 address=a43d7160 ,  roomPt = a43d7160
```

Structure: Exercise

Exercise 3

C_structure_exercise3.c

- Define a structure member for 3D position
- Create the following functions

```
typedef struct {
        int x;
        int y;
        int z;
} POSITION_TypeDef;
```

```
void addPos(POSITION_TypeDef pos1, POSITION_TypeDef pos1, POSITION_TypeDef *posOut);
void getDist(POSITION_TypeDef pos1, POSITION_TypeDef pos1, POSITION_TypeDef *posOut);
void printPos (POSITION_TypeDef Pos);
```

*Structure for Embedded C

For Embedded Controller

Structure: for Embedded C

- Example in MCU programming: Structure
  - Use structure to define I/O Memory register

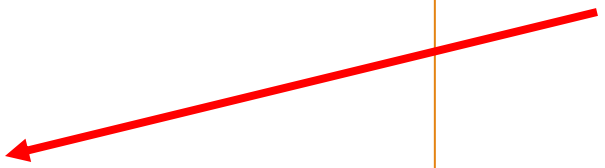
GPIO register


```
typedef struct {
  volatile uint32_t MODER;    // Mode register
  volatile uint32_t OTYPER;   // Output type register
  volatile uint32_t OSPEEDR;  // Output speed register
  volatile uint32_t PUPDR;    // Pull-up/pull-down register
  volatile uint32_t IDR;      // Input data register
  volatile uint32_t ODR;      // Output data register
  volatile uint32_t BSRR;     // Bit set/reset register
  volatile uint32_t LCKR;     // Configuration lock register
  volatile uint32_t AFR[2];   // Alternate function registers
  volatile uint32_t BRR;      // Bit Reset register
  volatile uint32_t ASCR;     // Analog switch control
register
} GPIO_TypeDef;

// Casting memory address to a pointer
#define GPIOA ((GPIO_TypeDef *) 0x48000000)
```

Structure

- **Structure within Structure**
  - Useful technique for embedded programming
  - Example: structures for Finite State Machine, which you have learnt in Digital Logic

```
typedef struct{
        uint8_t Out
        uint8_t Time;
        const struct State *Next[2];
} State;
State State_t ;

State_t  FSM[4]={
        {0x21, 3000,  {&FSM[0], &FSM[1] }}
        {0x22, 500,    {&FSM[1], &FSM[1] }}
};
```



State Structure within State Structure

Structure, Enum

- Enumeration
  - You may use enum to define argument parameter for I/O control
  - Example: GPIO Pin Mode: Digital In, Digital Out, etc..

```
typedef enum
{
  MODE_IN          =0,          // DIn
  MODE_OUT         =1,          // DOut
  MODE_AF                ,      // =2 alternative function
  MODE_AN                       // =3 analog
} GPIO_MODE_Type;
```

```
void GPIO_Initialize(GPIO_TypeDef *Port, uint32_t Pin, GPIO_Mode_Type mode);
...

GPIO_MODE_Type mode=MODE_IN;
GPIO_Initialize(GPIOA, PIN_5, mode);
```