This project is totally about improving the led - toggle project with help of bit fields and struct.

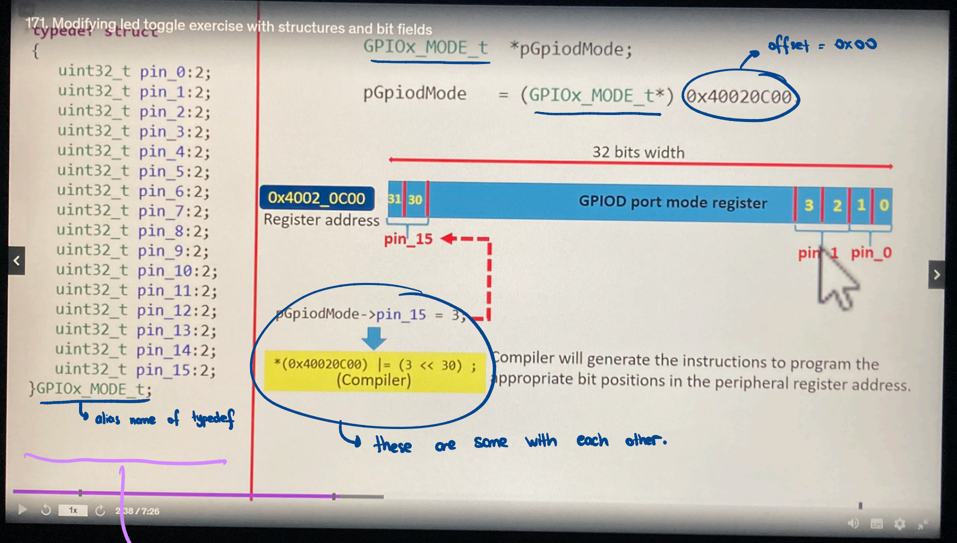
```
typedef    struct
{


} RCC_AHB1ENR_t;
```

$\underbrace{\text{RCC}}_{\text{peripheral name}}\_\underbrace{\text{AHB1ENR}\_}_{\text{peripheral's register name}}$

you have to create a different bit field structures for different peripheral registers.
for example:

$\underbrace{\text{GPIOx\_MODE\_t}}_{\text{GPIO mode register}}$, $\underbrace{\text{GPIOx\_ODR\_t}}_{\text{GPIO Output Data Register}}$;

- when we defined this typedef struct, for every GPIO mode we are using some bit fields for enable or disable our system. For generalize our system, we are defining a structure and use it for every condition.



171. Modifying led toggle exercise with structures and bit fields

```
typedef struct
{
    uint32_t pin_0:2;
    uint32_t pin_1:2;
    uint32_t pin_2:2;
    uint32_t pin_3:2;
    uint32_t pin_4:2;
    uint32_t pin_5:2;
    uint32_t pin_6:2;
    uint32_t pin_7:2;
    uint32_t pin_8:2;
    uint32_t pin_9:2;
    uint32_t pin_10:2;
    uint32_t pin_11:2;
    uint32_t pin_12:2;
    uint32_t pin_13:2;
    uint32_t pin_14:2;
    uint32_t pin_15:2;
}GPIOx_MODE_t;
```

alias name of typedef

```
GPIOx_MODE_t   *pGpiodMode;

pGpiodMode    = (GPIOx_MODE_t*) 0x40020C00
```

offset = 0x00

```
pGpiodMode->pin_15 = 3;

*(0x40020C00) |= (3 << 30) ;
(Compiler)
```

Compiler will generate the instructions to program the appropriate bit positions in the peripheral register address.

these are same with each other.

we defined this struct in main.h which we created.