HOW TO TURN A LED IN STM32F407 DISCOVERY BOARD

1. Identify internal LEDs in my board. In this case, if we check the pinout of our board we can find four different leds which are in port D.

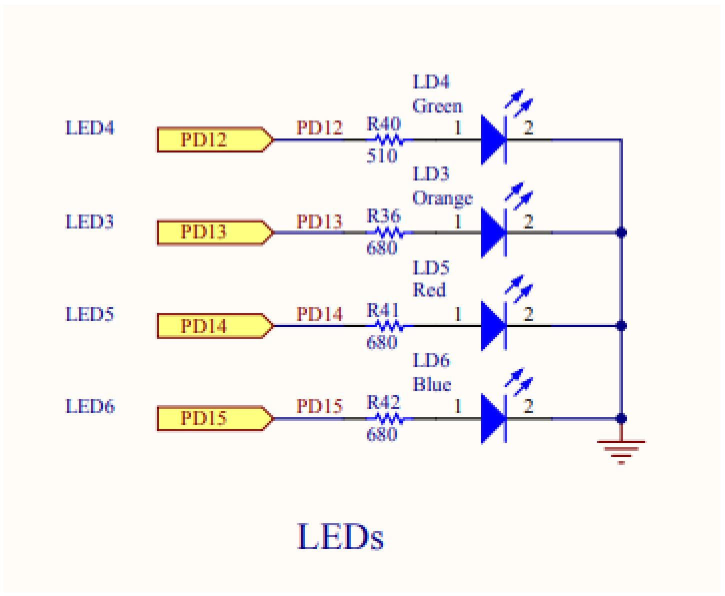
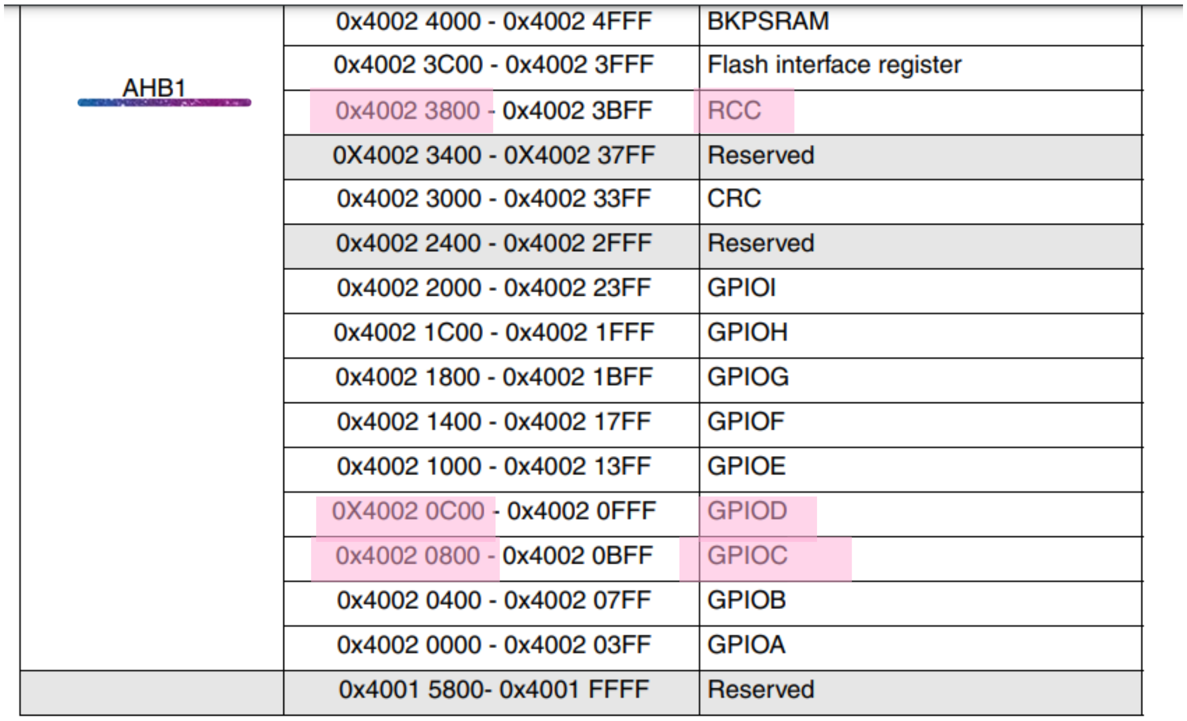


Figure 1. Pinout Internal LEDs in STM32F407 Discovery Board.

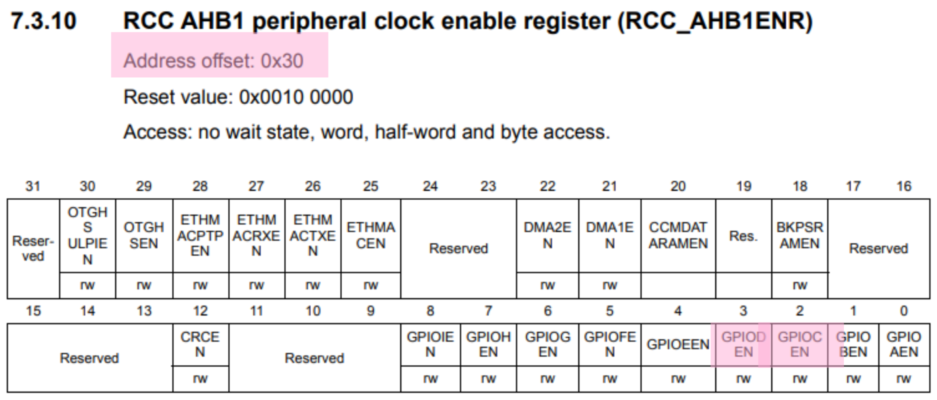
2. Search for registers as: clock, port D and port C in the datasheet.



AHB1	0x4002 4000 - 0x4002 4FFF	BKPSRAM
	0x4002 3C00 - 0x4002 3FFF	Flash interface register
	0x4002 3800 - 0x4002 3BFF	RCC
	0X4002 3400 - 0X4002 37FF	Reserved
	0x4002 3000 - 0x4002 33FF	CRC
	0x4002 2400 - 0x4002 2FFF	Reserved
	0x4002 2000 - 0x4002 23FF	GPIOI
	0x4002 1C00 - 0x4002 1FFF	GPIOH
	0x4002 1800 - 0x4002 1BFF	GPIOG
	0x4002 1400 - 0x4002 17FF	GPIOF
	0x4002 1000 - 0x4002 13FF	GPIOE
	0X4002 0C00 - 0x4002 0FFF	GPIOD
	0x4002 0800 - 0x4002 0BFF	GPIOC
	0x4002 0400 - 0x4002 07FF	GPIOB
	0x4002 0000 - 0x4002 03FF	GPIOA
	0x4001 5800- 0x4001 FFFF	Reserved

Figure 2. Memory mapping

3. Go to RCC Registers and click in the section to enable the clock of the AHB1. Here we are going to add the *Address Offset* with the RCC register.



7.3.10 RCC AHB1 peripheral clock enable register (RCC_AHB1ENR)

Address offset: 0x30

Reset value: 0x0010 0000

Access: no wait state, word, half-word and byte access.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reser-ved	OTGHS ULPIEN	OTGHSEN	ETHMACPTPEN	ETHMACRXEN	ETHMACTXEN	ETHMACEN	Reserved		DMA2EN	DMA1EN	CCMDATARAMEN	Res.	BKPSRAMEN	Reserved	
	rw	rw	rw	rw	rw	rw			rw	rw			rw		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			CRCEN	Reserved			GPIOIEN	GPIOHEN	GPIOGEN	GPIOFEN	GPIOEEN	GPIODEN	GPIOCEN	GPIOBEN	GPIOAEN
			rw				rw	rw	rw	rw	rw	rw	rw	rw	rw

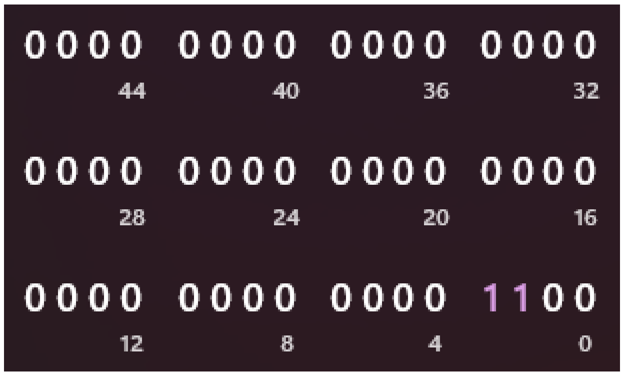
Figure 3. RCC Registers: RCC AHB1 clock enable register

RCC: 0x40023800

EN: 0x30

pClkEN = 0x40023830

4. Then, we are going to enable the clock according to the ports we chose. In this case, as we can see in figure 3, we must set the bit 2 and 3.



```
0000 0000 0000 0000
  44   40   36   32

0000 0000 0000 0000
  28   24   20   16

0000 0000 0000 1100
  12    8    4    0
```

Figure 4. Setting bits.

**pClkEN |= 0xC;* or **pClkEN |= (1 << 2) | (1 << 3);*

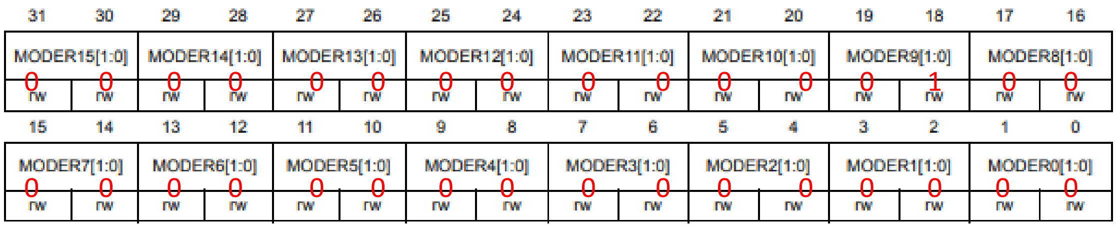
5. We are going to calculate the register with the address offset of GPIO moder, which is 0x00. So, we are going to have basically the main register of our ports.

**pPortModeD= 0x40020C00;*
**pPortModeC=0x40020800;*

6. Configure the ports as outpus. We must go to GPIOx_ODR section and adding the address offset to our ports.

**pOutputC= 40020814;*
**pOutputD = 0x40020C14;*

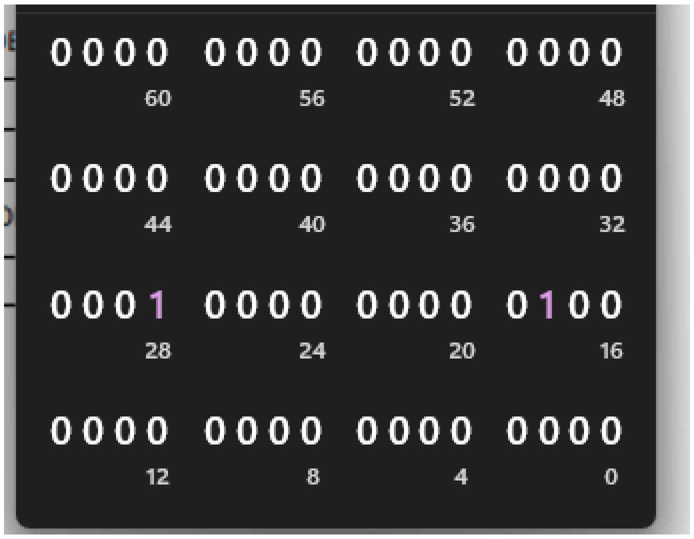
7. Finally, we are going to configurate the bits. Basically we are going to the GPIOx_MODER and start to setting all the ports or pins that we are going to use. In this case, my pins were 9 and 14, so we are going to clear all the bytes and then, set only the bit 28 and 18 due to the port configuration that we have that is 01: General purpose.



31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODER15[1:0]		MODER14[1:0]		MODER13[1:0]		MODER12[1:0]		MODER11[1:0]		MODER10[1:0]		MODER9[1:0]		MODER8[1:0]	
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODER7[1:0]		MODER6[1:0]		MODER5[1:0]		MODER4[1:0]		MODER3[1:0]		MODER2[1:0]		MODER1[1:0]		MODER0[1:0]	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

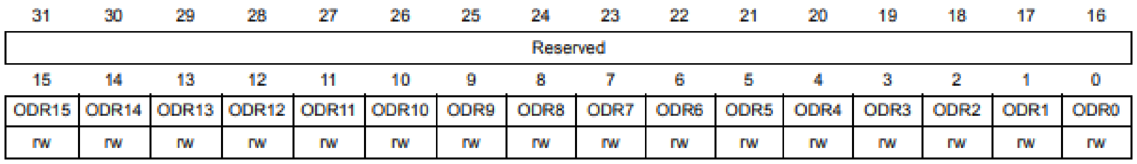
Figure 5. GPIO port moder register.



```
0000  0000  0000  0000
  60    56    52    48

0000  0000  0000  0000
  44    40    36    32

0001  0000  0000  0100
  28    24    20    16

0000  0000  0000  0000
  12     8     4     0
```

*pPortModeC = 0;
*pPortModeC |=0x40000; //Setting first one bit 18.
*pPortModeD=0;
*pPortModeD|=0x100000;

8. To finish, we are going to set our pines. So, we are going to look at the GPIOx_ODR, and set one by one the bit in where our pin is going.



31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ODR15	ODR14	ODR13	ODR12	ODR11	ODR10	ODR9	ODR8	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Figure 6. GPIO port output data register.

*pOutputC|=0x200;

pOutputD|=0x4000;

9. And then, we are going to build our code and debug it to see the function of it. In the following figure, we can see the internal led turned on as the external led.

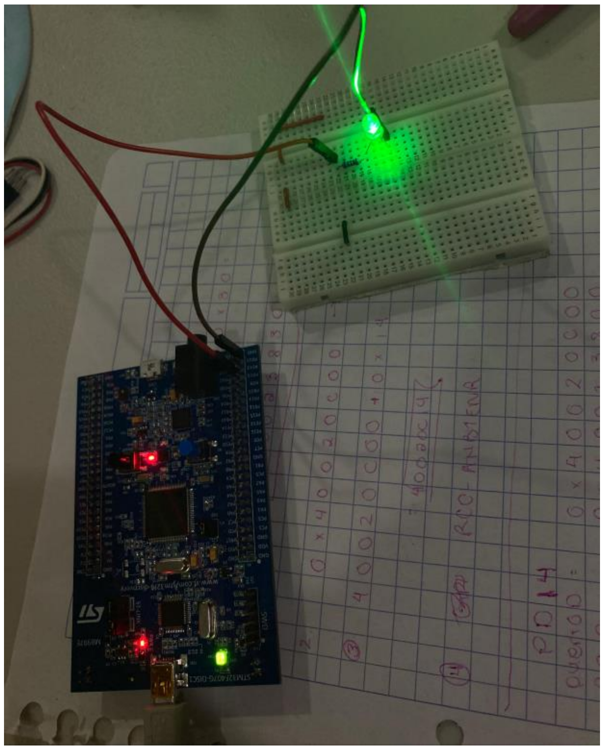


Figure 7. Final product