GPIO Programming

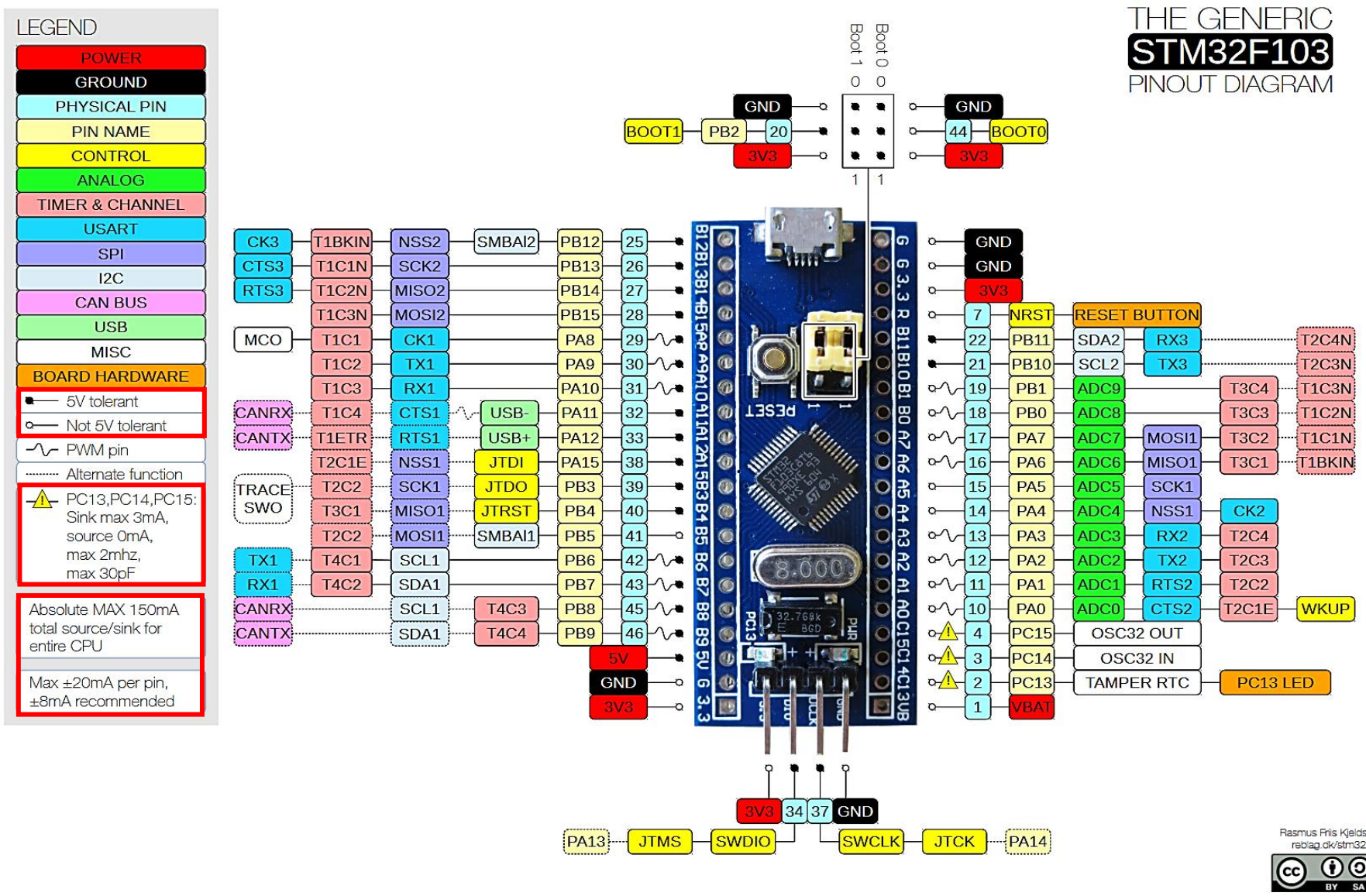
Topics

- Start using STM32Cube
- GPIO Programming

Starting to use STM32Cube

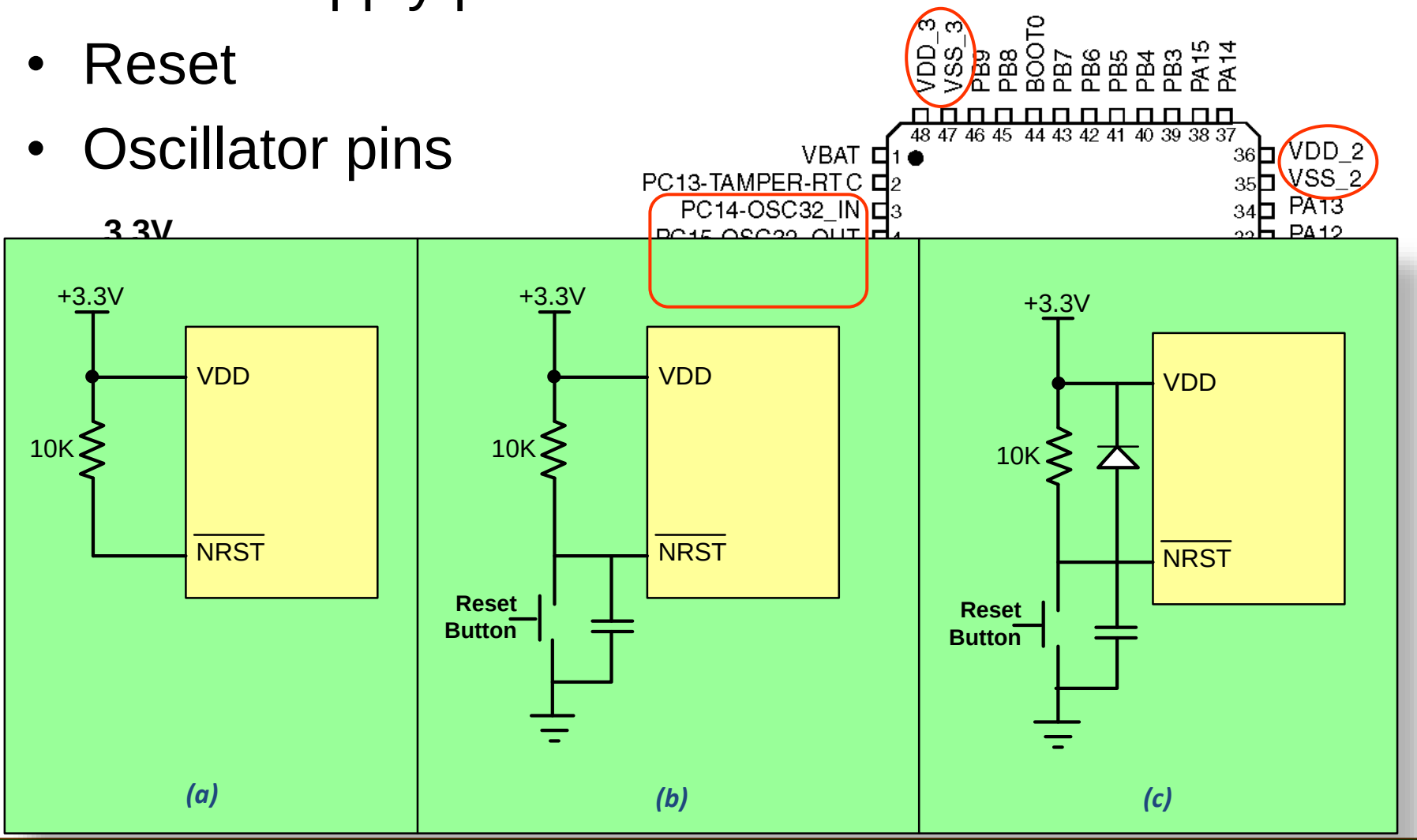
- Installing required software
- Create a project
- Clock configuration
- Peripheral configuration

BluePill board

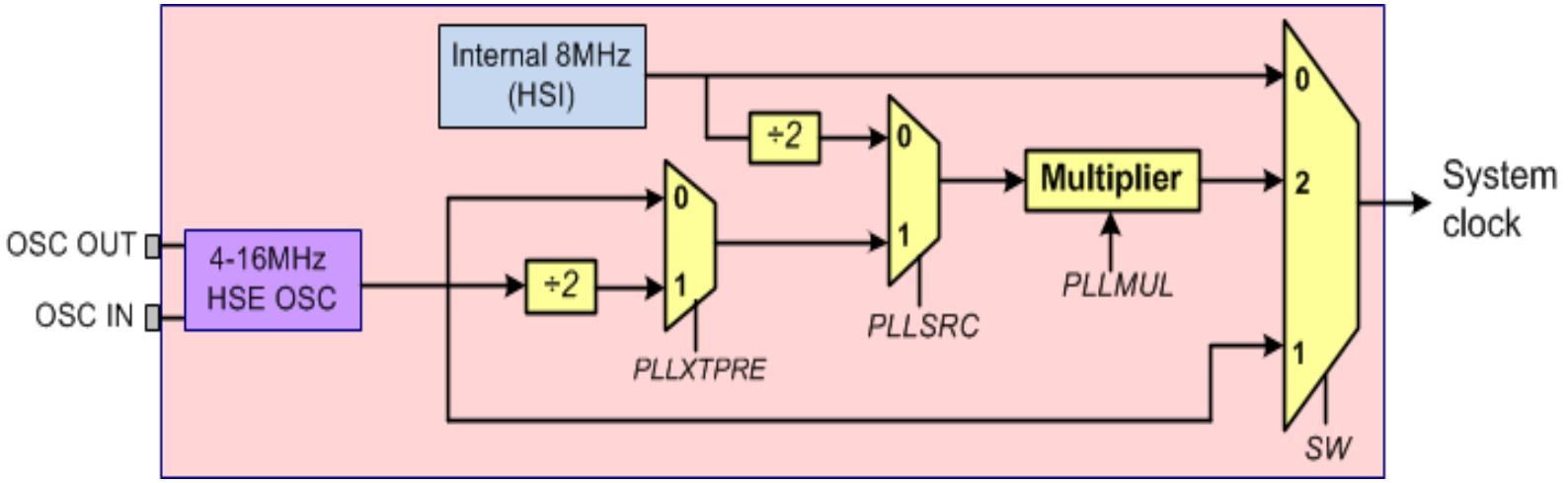


STM32F103C8 Pinout

- Power supply pins
- Reset
- Oscillator pins



System Clock



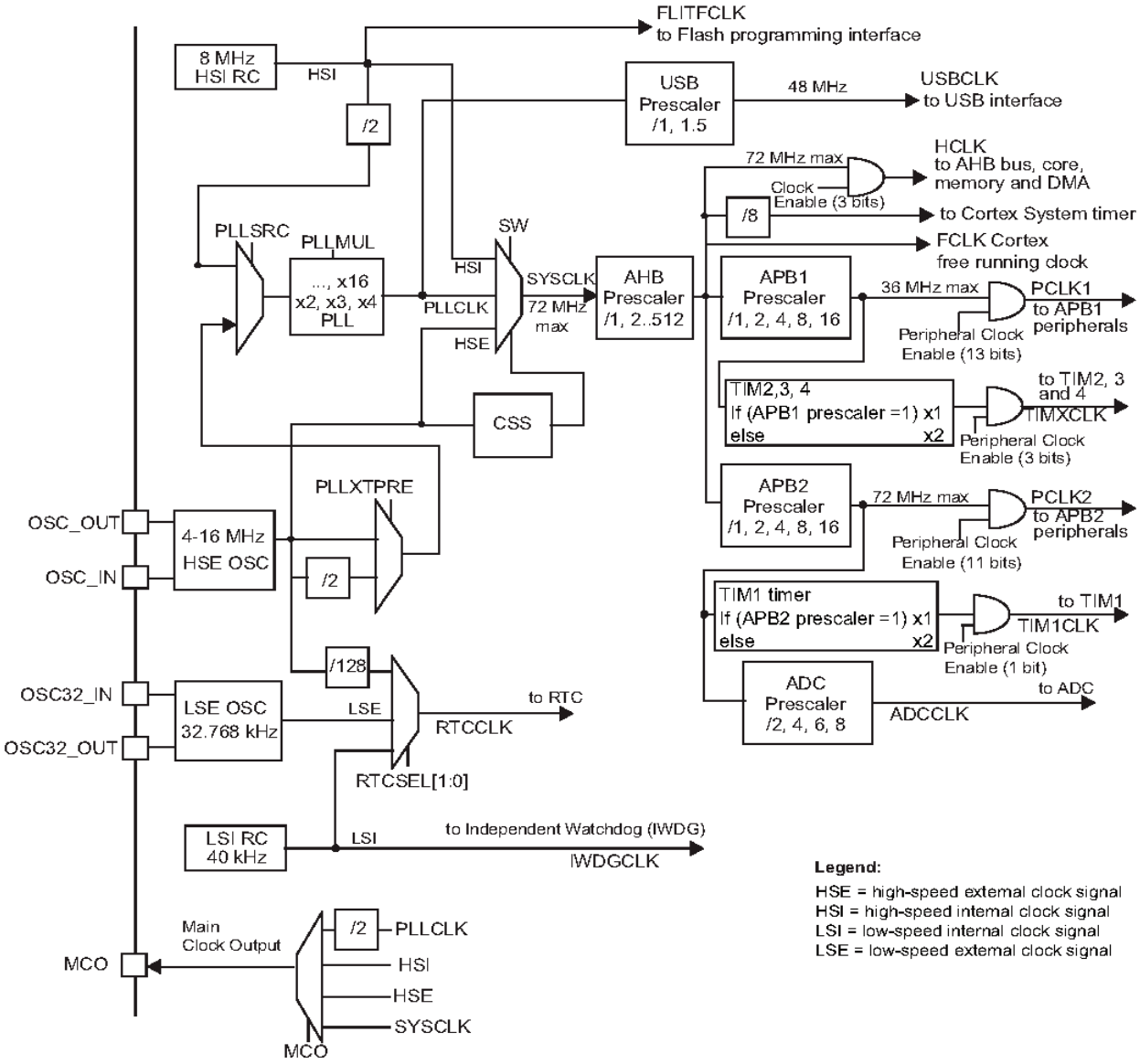
RCC_CFGR: (RCC->CFGR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					MCO			Res.	USB PRE	PLLMUL				PLLXT PRE	PLL SRC
				rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw

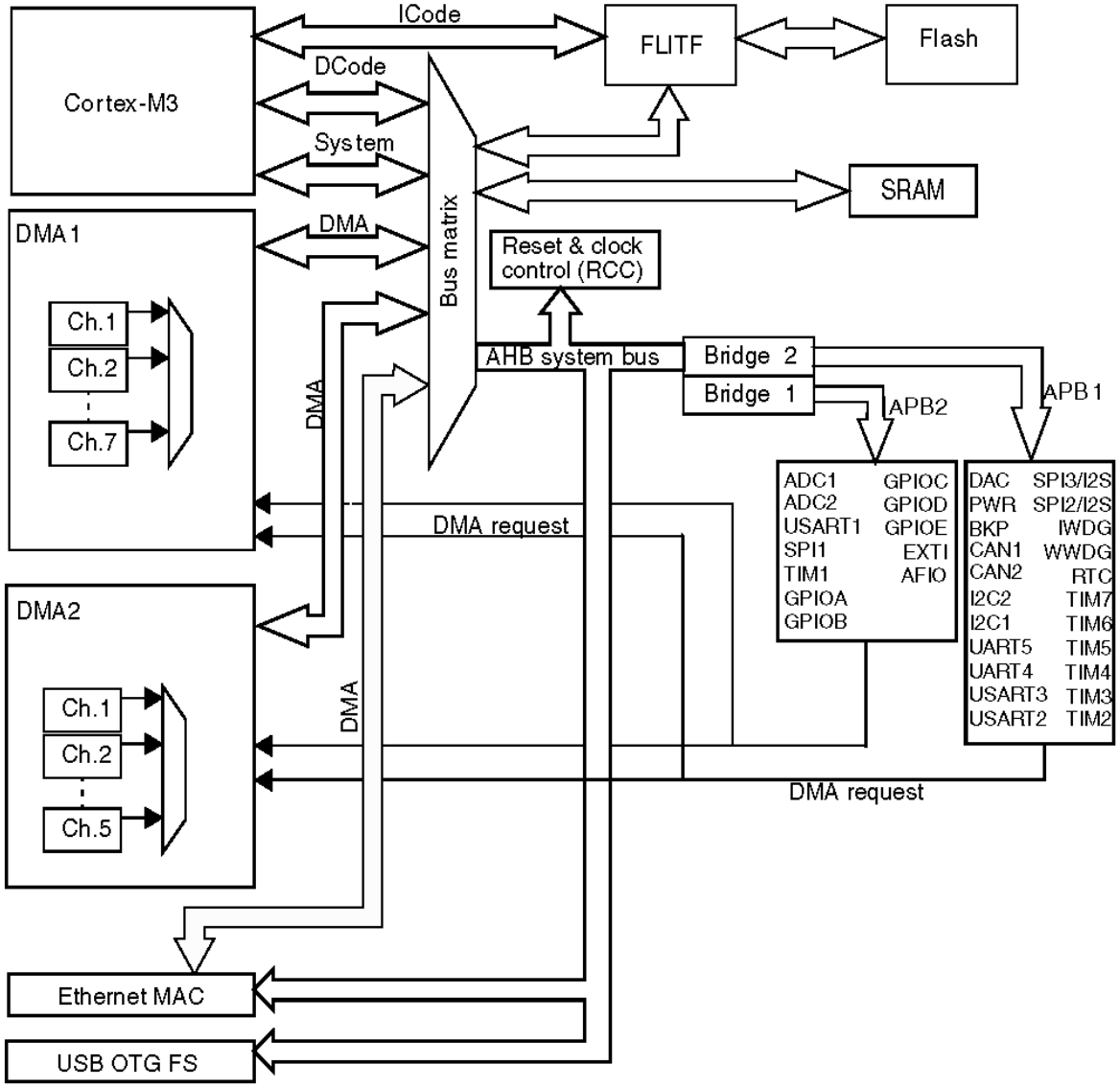
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADC PRE		PPRE2			PPRE1			HPRE				SWS		SW	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	r	r	rw	rw

PLLMUL	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Multiplication	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12	X13	X14	X15	X16	

STM32F10X Clock



APB1, APB2, and AHB



Enabling Clocks

RCC_APB1ENR:
(RCC->APB1ENR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved		DACEN	PWR EN	BKPEN	Res.	CAN EN	Res.	USBEN	I2C2EN	I2C1EN	USART 5EN	USART 4EN	USART 3EN	USART 2EN	Res.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPI3EN	SPI2EN	reserved		WWDG EN	reserved		TIM14 EN	TIM13 EN	TIM12 EN	TIM7 EN	TIM6 EN	TIM5 EN	TIM4 EN	TIM3 EN	TIM2 EN

RCC_APB2ENR:
(RCC->APB2ENR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved										TIM11 EN	TIM10 EN	TIM9 EN	Res.		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADC3 EN	USART 1EN	TIM8 EN	SPI1 EN	TIM1 EN	ADC2 EN	ADC1 EN	IOPG EN	IOPF EN	IOPE EN	IOPD EN	IOPC EN	IOPB EN	IOPA EN	Res.	AFIO EN

RCC_AHBENR:
(RCC->AHBENR)

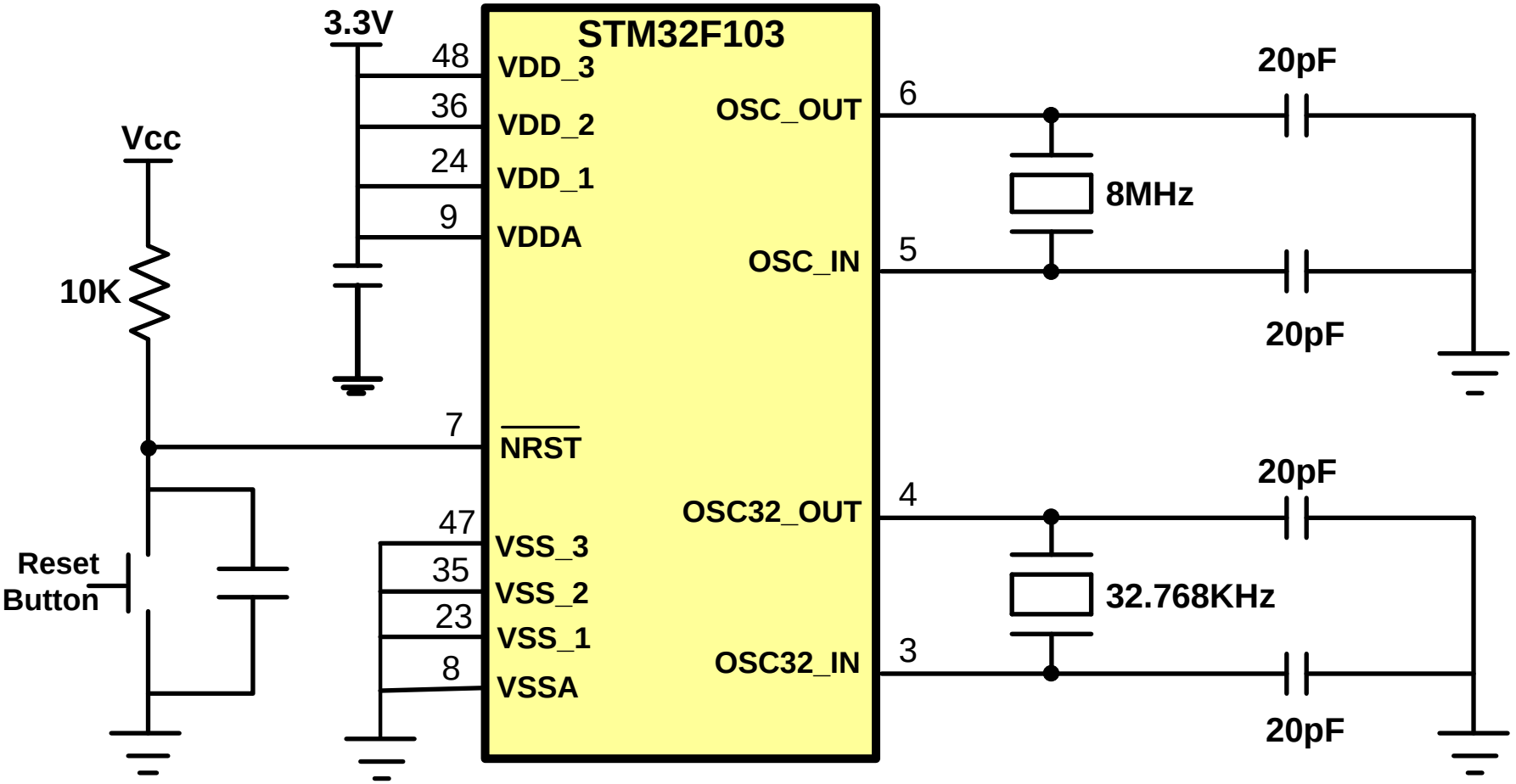
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved					SDIO EN	Res.	FSMC EN	Res.	CRC EN	Res.	FLITF EN	Res.	SRAM EN	DMA2 EN	DMA1 EN

Label	Description	Label	Description
IOPx	I/O port x clock enable	ADCnEN	ADCn clock enable
USARTnEN	USARTn clock enable	DACnEN	DACn clock enable
USBEN	USB clock enable	TIMnEN	TIMn timer clock enable
CANEN	CAN clock enable	SPInEN	SPI n clock enable
PWREN	Power interface clock enable	BKPEN	Backup interface clock enable
WWDG	Window watchdog clock enable	SDIOEN	SDIO clock enable
DMAnEN	DMAn clock enable	FSMCEN	FSMC clock enable
CRCEN	CRC clock enable	I2CnEN	I2Cn clock enable

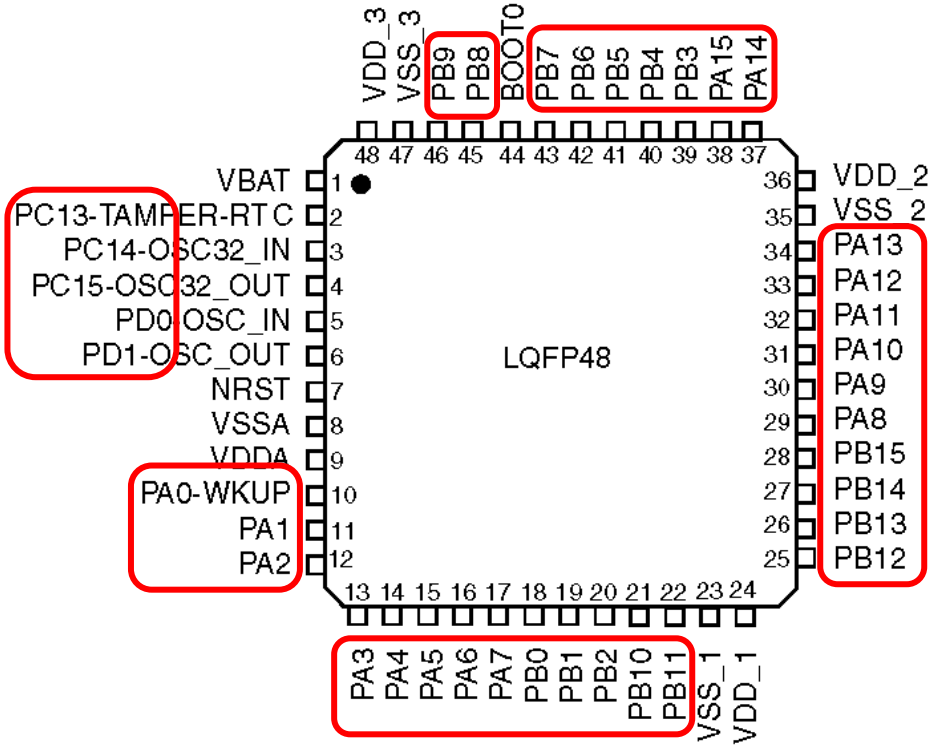
Note: (0: clock disabled, 1: clock enabled)

Reset, Power, and Crystals

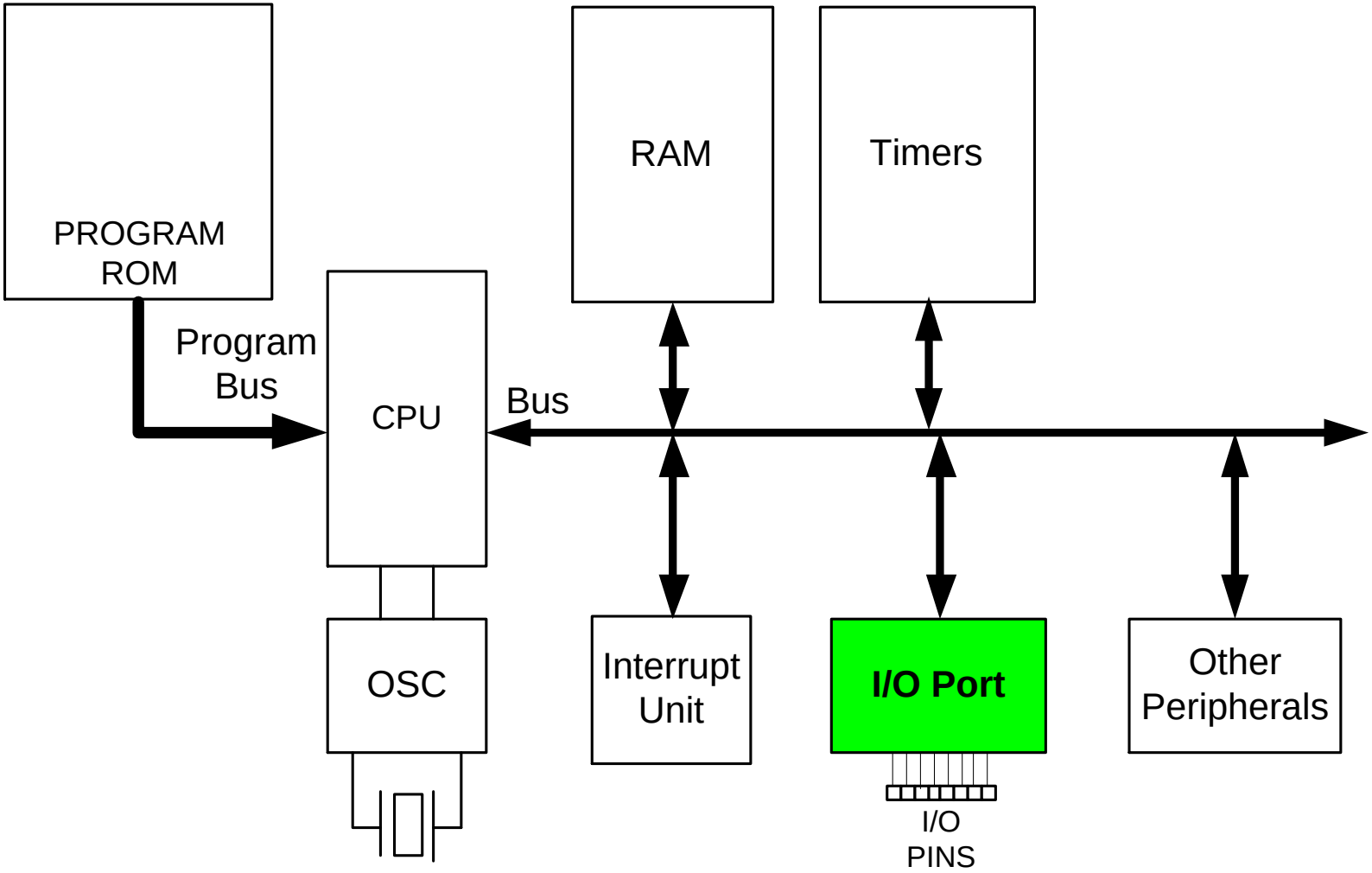


GPIO pins

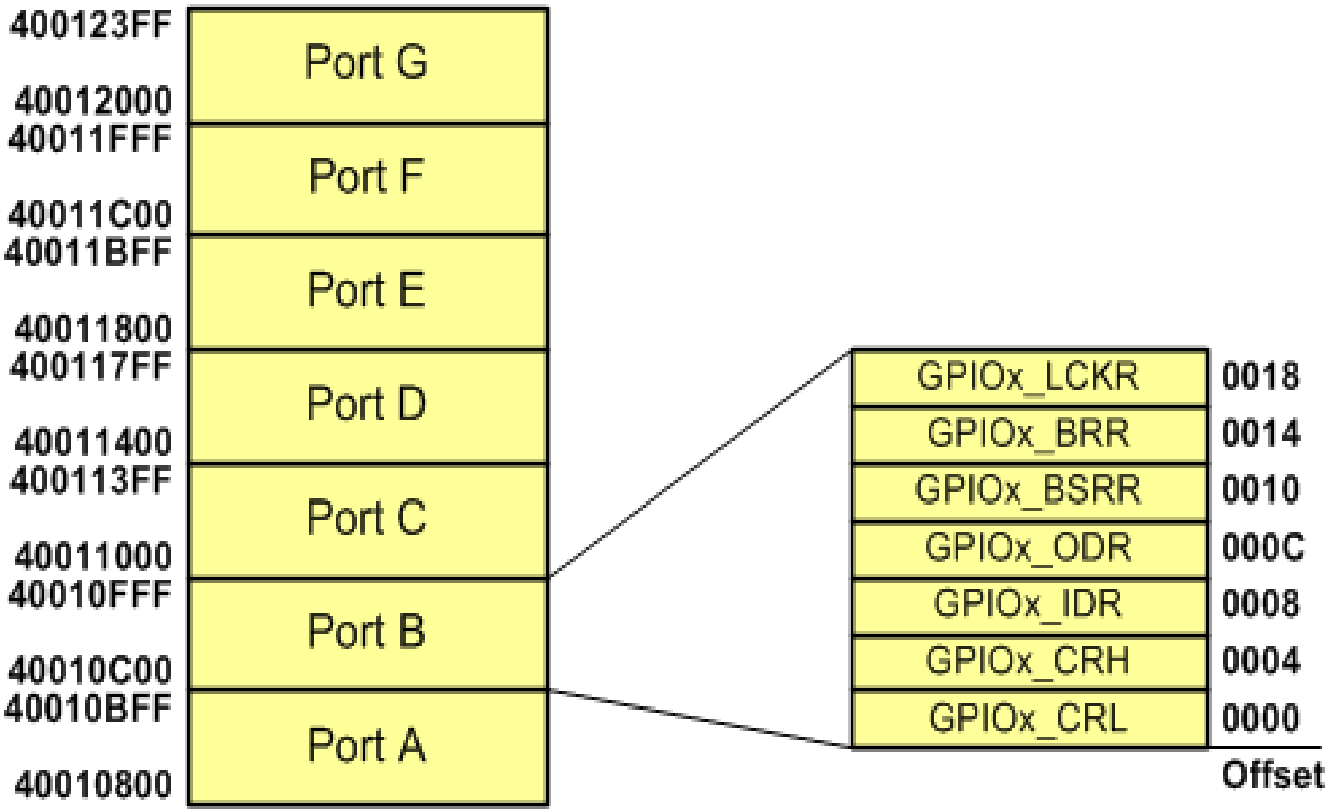
- PORTA (PAn)
- PORTB (PBn)
- PORTC (PCn)
- PORTD



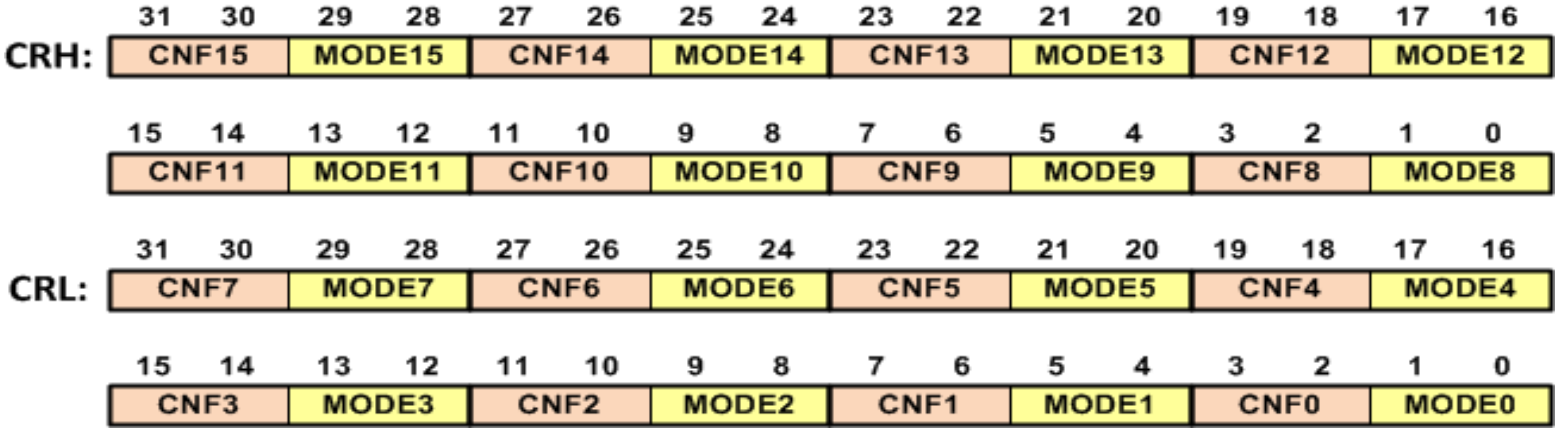
I/O unit in ARM



GPIO Registers



CRL and CRH (Configuration Registers)



Output (MODE>00)

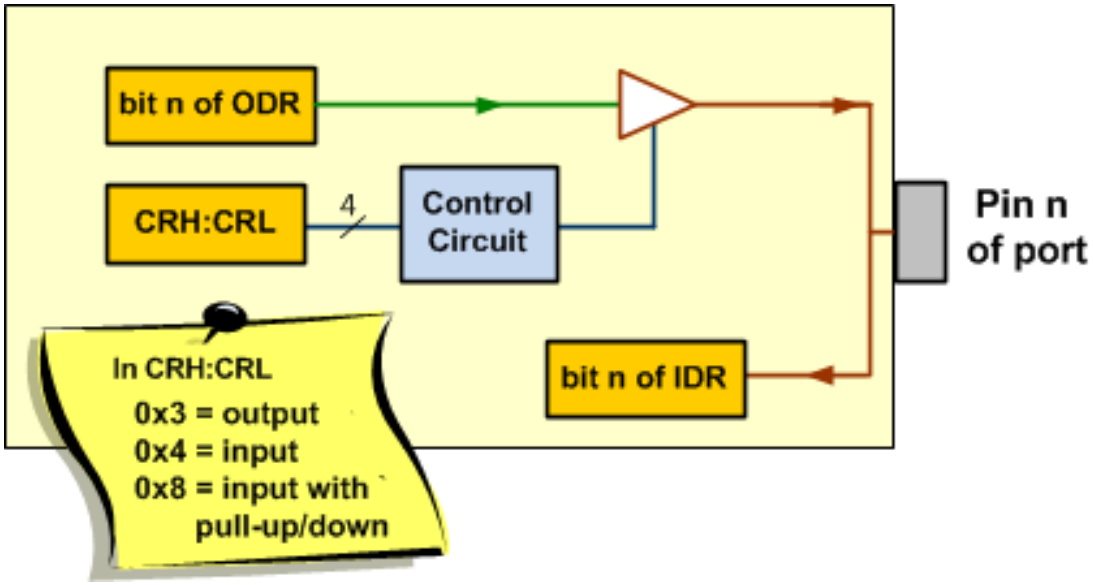
CNFx bits	
00	General purpose output push-pull
01	General purpose output Open-drain
10	Alternate function output Push-pull
11	Alternate function output Open-drain

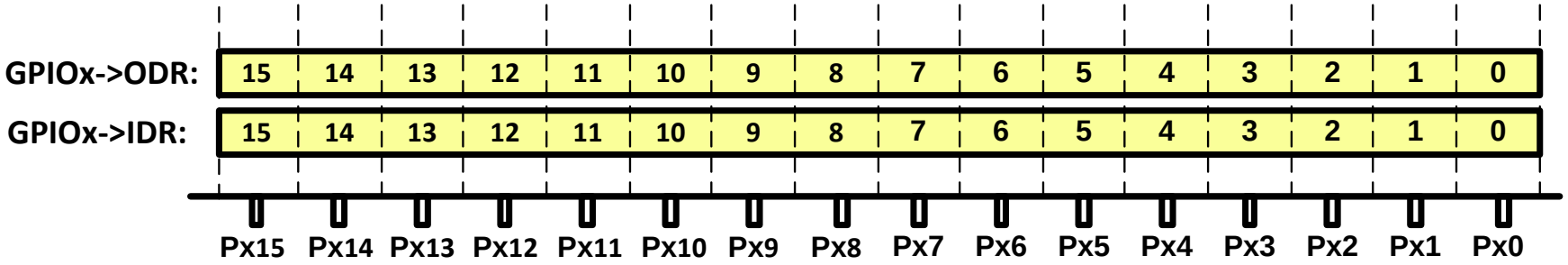
MODEx bits	Direction	Max speed
00	Input	
01	Output	10 MHz
10		2 MHz
11		50 MHz

Input (MODE=00)

CNFx bits	Configuration	Description
00	Analog mode	Select this mode when you use a pin as an ADC input.
01	Floating input	In this mode, the pin is high-impedance.
10	Input with pull-up/pull-down	The value of ODR chooses if the pull-up or pull-down resistor is enabled. (1: pull-up, 0:pull-down)
11	reserved	

IDR (Input Data Reg.) and ODR (Output Data Reg.)



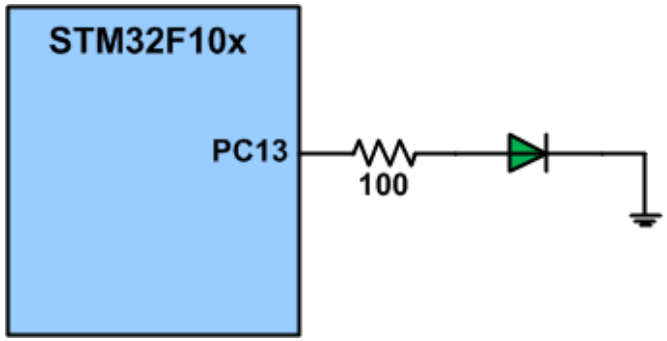


Toggle Port A

```
#include "stm32f10x.h"

void delay_ms(uint16_t t) {
        volatile unsigned long l = 0;
        for(uint16_t i = 0; i < t; i++)
                for(l = 0; l < 6000; l++)
                { }
}
int main() {
        RCC->APB2ENR |= 0xFC; //Enable the clocks for GPIO ports

        GPIOA->CRL = 0x33333333; //PA0 to PA7 as outputs
        GPIOA->CRH = 0x33333333; //PA8 to PA15 as outputs

        while(1) {
                GPIOA->ODR = 0x0000; //make all the pins of Port A low
                delay_ms(1000);  //wait 1000ms
                GPIOA->ODR = 0xFFFF; //make all the pins of Port A high
                delay_ms(1000);  //wait 1000ms
        }
}
```

Toggling PC13

```
#include <stm32f10x.h>

void delay_ms(uint16_t t);

int main()
{
        RCC->APB2ENR |= 0xFC; //Enable GPIO ports clocks

        GPIOC->CRH = 0x44344444; //PC13 as output

        while(1)
        {
                GPIOC->ODR ^= (1<<13); //toggle PC13
                delay_ms(1000);
        }
}
```



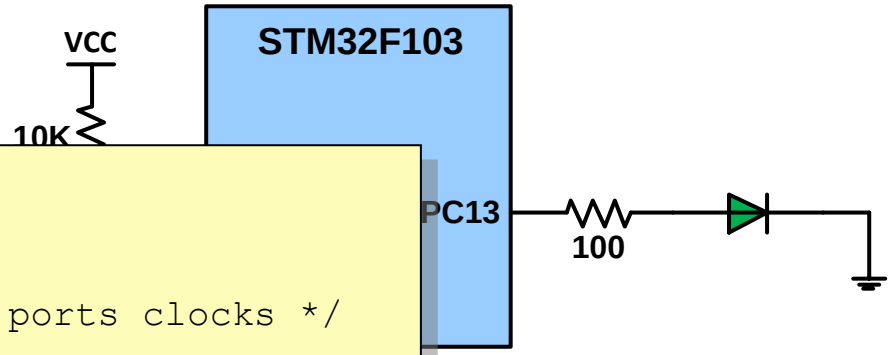
Example: Input

- The following code gets the data present at the pins of port A and sends it to port B indefinitely, after adding the value 5 to it:

```
#include <stm32f10x.h>

int main()
{
  RCC->APB2ENR |= 0xFC; /* Enable GPIO ports clocks */

  GPIOA->CRL = 0x33333333; /* PA0-PA7 as outputs */
  GPIOA->CRH = 0x33333333; /* PA8-PA15 as outputs */
  GPIOB->CRL = 0x44444444; /* PB0-PB7 as inputs */
  GPIOB->CRH = 0x44444444; /* PB8-PB15 as inputs */

  while(1)
  {
    GPIOA->ODR = GPIOB->IDR + 5;
        /* read from port B and write to port A */
  }
}
```

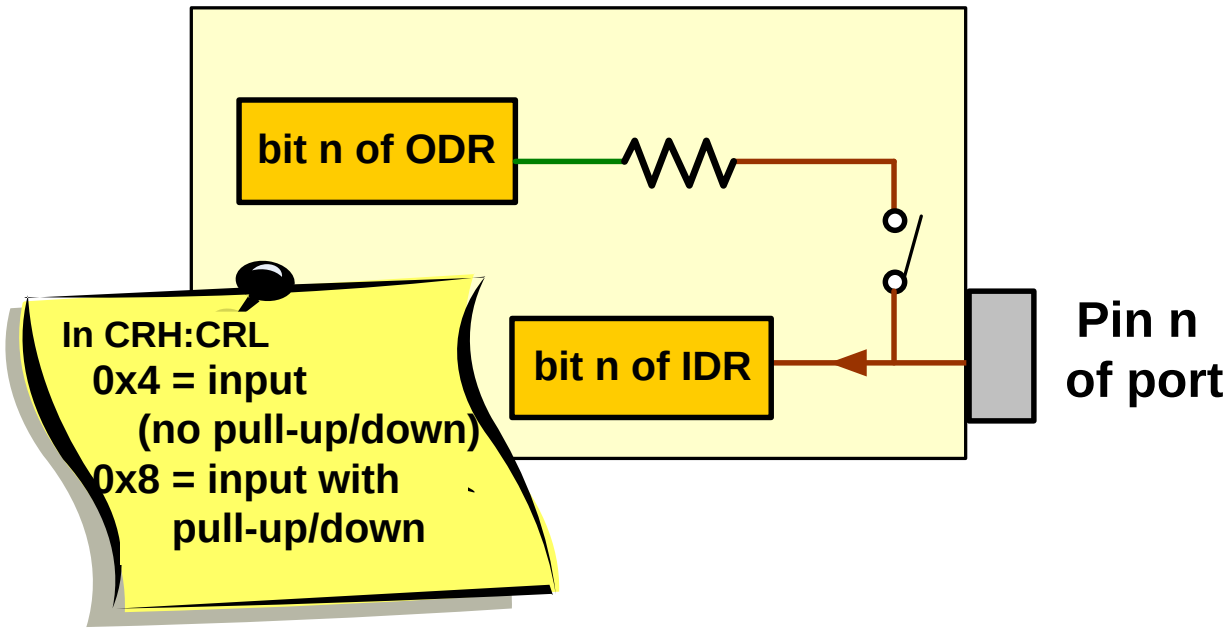
Example

- A switch is connected to pin PB10 and an LED to pin PC13. Write a program to get the status of SW and send it to the LED.



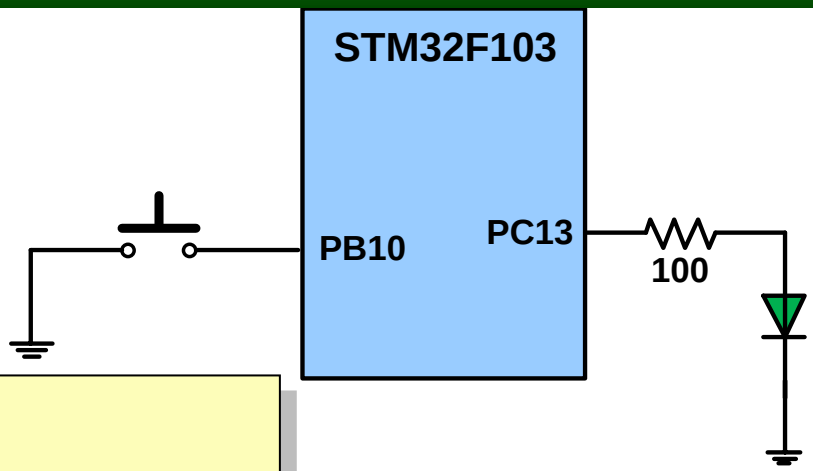

```
#include <stm32f10x.h>
int main() {
  RCC->APB2ENR |= 0xFC; /* Enable GPIO ports clocks */

  GPIOB->CRH = 0x44444444; /* PB8-PB15 as inputs */
  GPIOC->CRH = 0x44344444; /* PC13 as output */

  while(1) {
    if((GPIOB->IDR & (1<<10)) != 0) /* is PB10 high */
      GPIOC->ODR |= (1 << 13); /* make PC13 high */
    else
      GPIOC->ODR &= ~(1 << 13); /* make PC13 low */
  }
}
```

Internal Pull-up/Pull-down resistor



Example

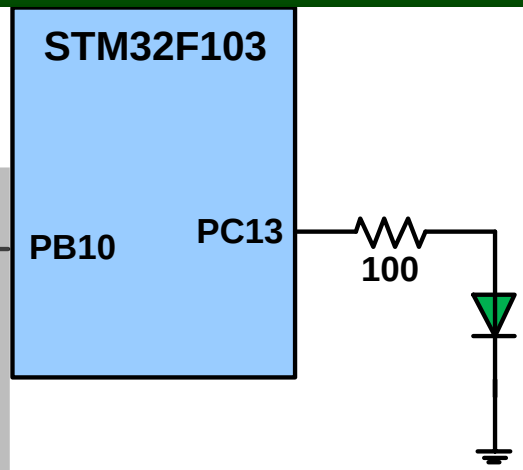
- A switch is connected to pin PB10 and an LED to pin PC13. Write a program to get the status of SW and send it to the LED




```
#include <stm32f10x.h>
int main() {
  RCC->APB2ENR |= 0xFC; /* Enable GPIO ports clocks */

  GPIOB->CRH = 0x44444844; // pull-up PB10
  GPIOB->ODR |= (1<<10); //set bit 10 of ODR to pull-up
  GPIOC->CRH = 0x44344444; /* PC13 as output */
  while(1) {
    if((GPIOB->IDR & (1<<10)) != 0) /* is PB10 high */
      GPIOC->ODR |= (1 << 13); /* make PC13 high */
    else
      GPIOC->ODR &= ~(1 << 13); /* make PC13 low */
  }
}
```

Example

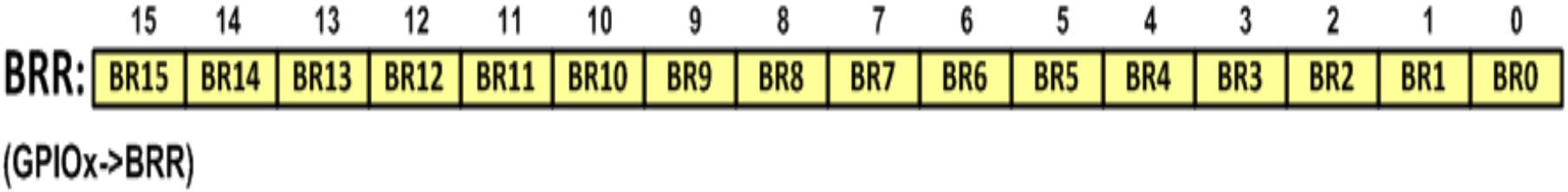
- A switch is connected to pin PB10 and an LED to pin PC13




```
#include <stm32f10x.h>
void delay_ms(uint16_t t);

int main() {
  RCC->APB2ENR |= 0xFC; /* Enable GPIO ports clocks */
  GPIOC->CRH = 0x44344444; /* PC13 as output */

  GPIOA->CRL = 0x44444844; /* PA2 as input with pull-up */
  GPIOA->ODR |= (1<<2);          /* pull-up PA2 */

  while(1) {
    if((GPIOA->IDR&(1<<2)) == 0) /* is PA2 low? */
      GPIOC->ODR ^= (1<<13); /* toggle PC13 */
    else
      GPIOC->ODR &= ~(1<<13);
    delay_ms(500);
    }
}
```

Clearing pins

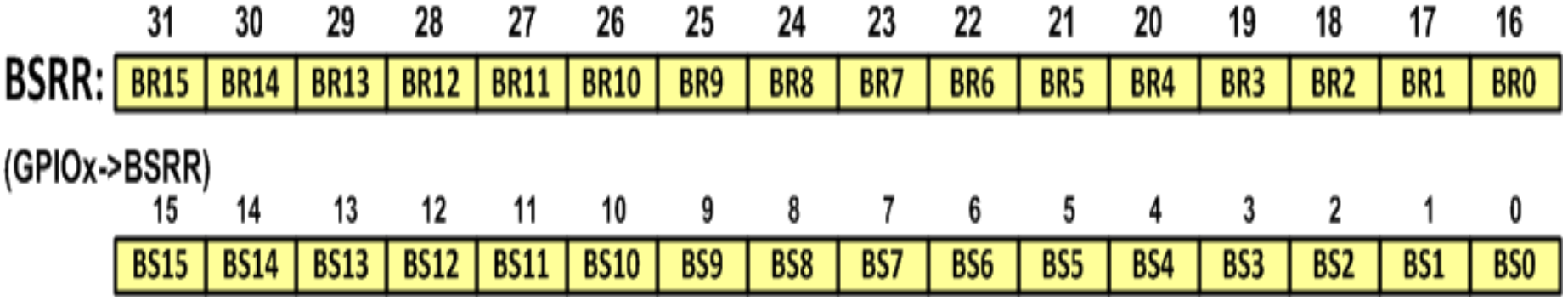
- BRR (Bit Reset Register)



- Examples:
  - GPIOB->BRR = 1<<5; //make PB5 low
  - GPIOA->BRR = (1<<3)|(1<<5); /* make PA3 and PA5 low */

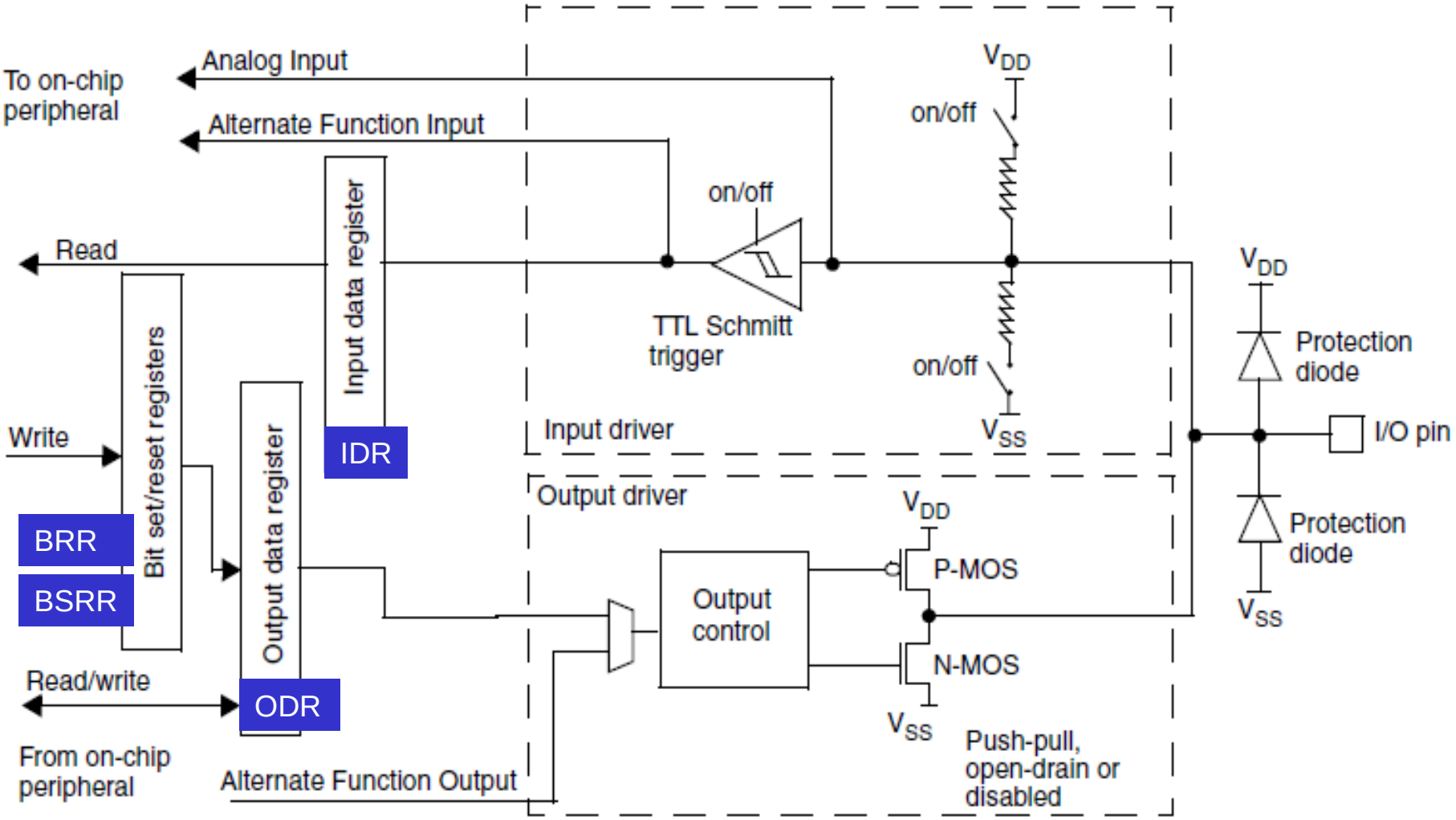
Setting & Clearing Pins

- BSRR (Bit Set/Reset Register)



- Examples:
  - GPIOC->BSRR = (1<<5); //make PC5 high
  - GPIOB->BSRR = (1<<5)|(1<<19); /*makes PB5 high and PB3 low */

The structure of I/O pins



The structure of 5-volt tolerant I/O pins