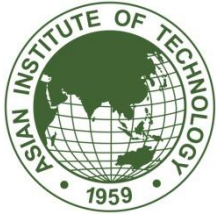
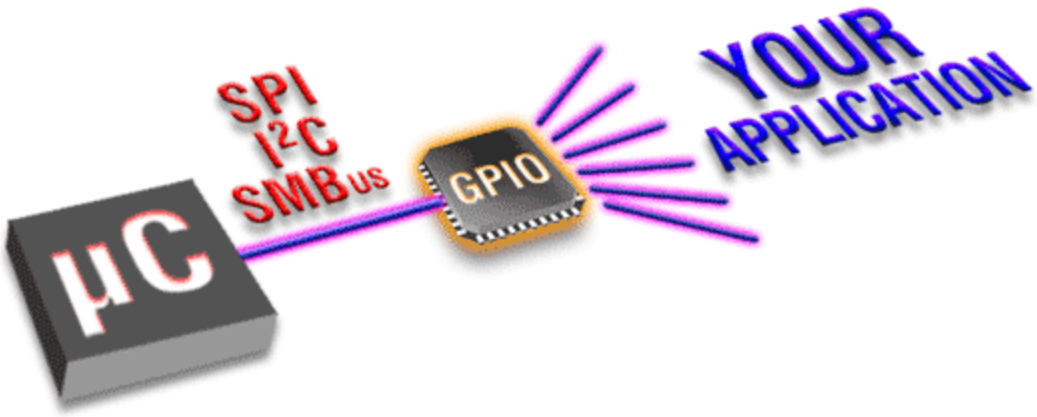


GPIO

General Purpose Input/Output (GPIO)

- Sometimes it is called bus expander
- GPIO is an interface available on microcontrollers/microprocessors
- It can act as input or output
- GPIO often are arranged into group of 8 pins
- GPIO port is often individually configurable

GPIO interface



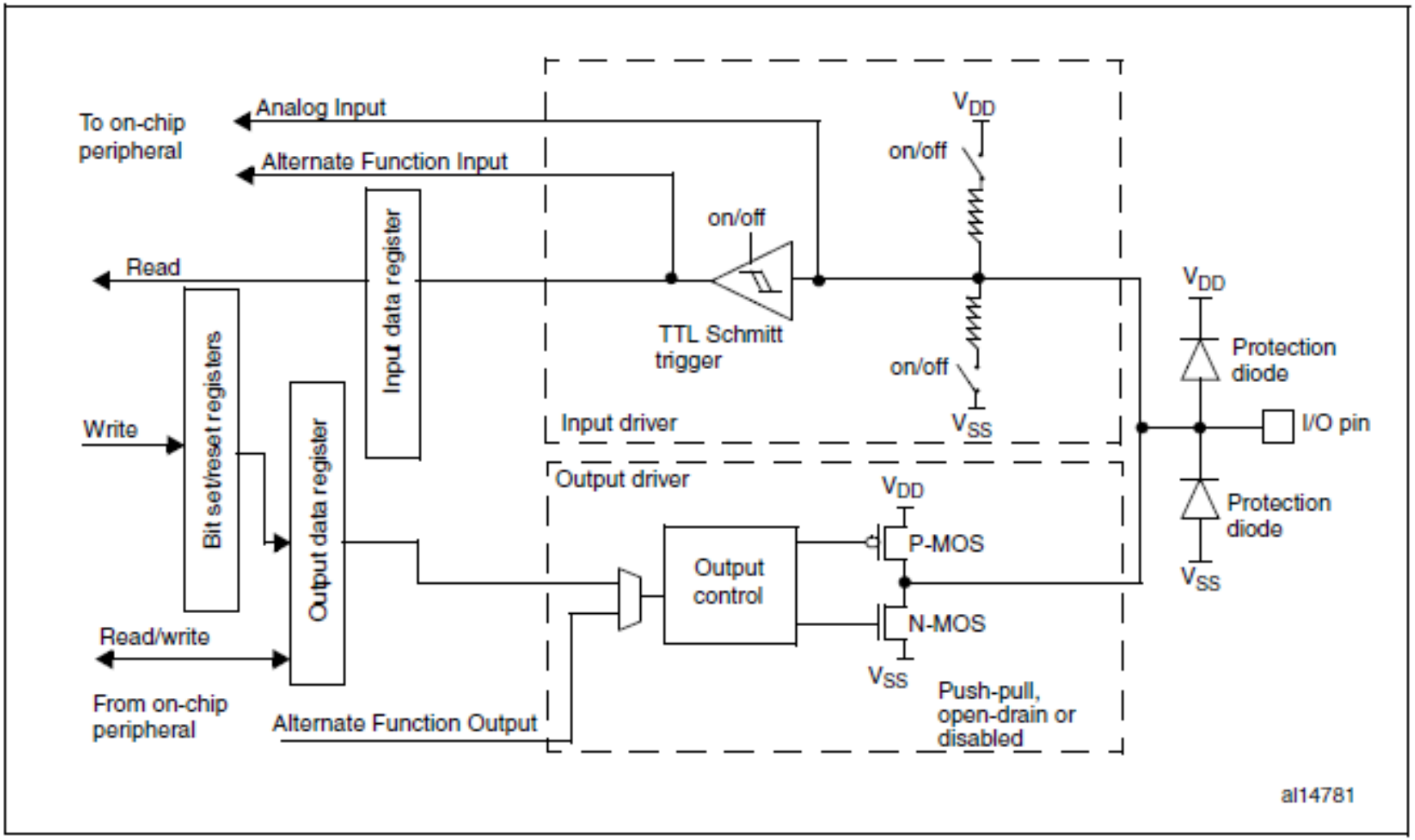
GPIO Advantage

- Lower power about 1 μA
- Can fit in small package
- Lower Cost
- Faster time to market

GPIO port for STM32F103

- STM32 has 5 ports: GPIOA, GPIOB,GPIOC, GPIOD, and GPIOE
- Each port has 16 pins, e.g., PC6 is bit 6 of port C
- Each pin can be individually configured to the following modes: input floating, input pull-up, input pull-down, analog input, output open drain, output push-pull, alternate push pull, alternate open-drain

Basic structure of standard I/O port bit

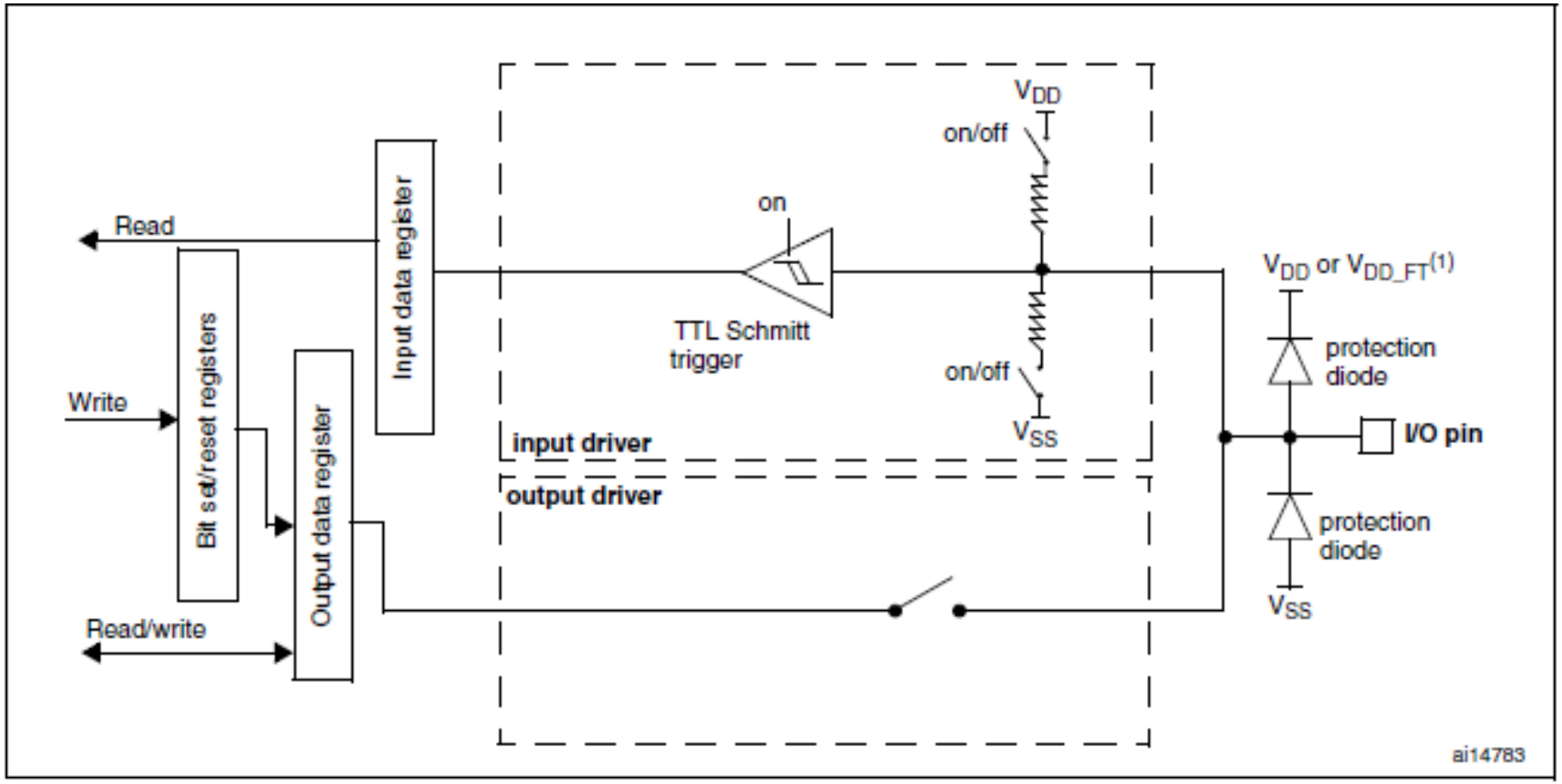


Input mode

- The output buffer is disabled
- The Schmitt Trigger input is activated
- The weak pull-up and pull-down resisters are activated or not depending on config.
- The data present on I/O pin is sampled into Input Data Register every APB2 clock cycle
- A read access to the Input Data Register to obtain the status

a Schmitt trigger is a circuit with positive feedback and a loop gain greater than 1. The circuit is named a "trigger" because the output retains its value until the input changes sufficiently to trigger a change

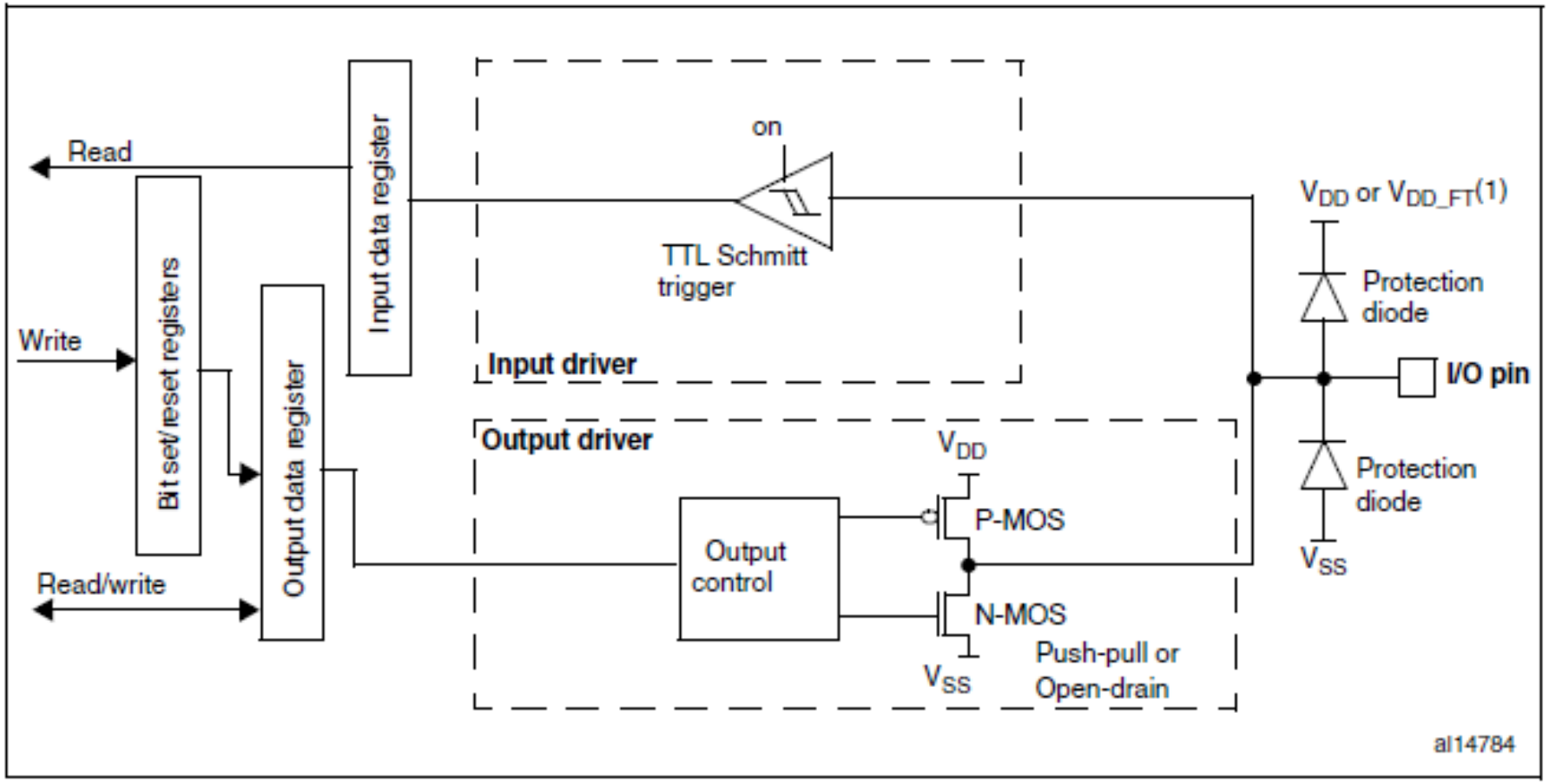
Input configurations



Output mode

- The output buffer is enabled
  - Open Drain mode: (turn off PMOS, NMOS)
  - Push/Pull mode: week zero and one
- The Schmitt Trigger is activated
- The weak pull up and pull down are disabled
- The data is sampled into Input Data Register every APB2 clock
- In open drain mode: a read will access to Input Data Register and get I/O state
- In push/pull mode: a read will access to Output Data register and get last written value

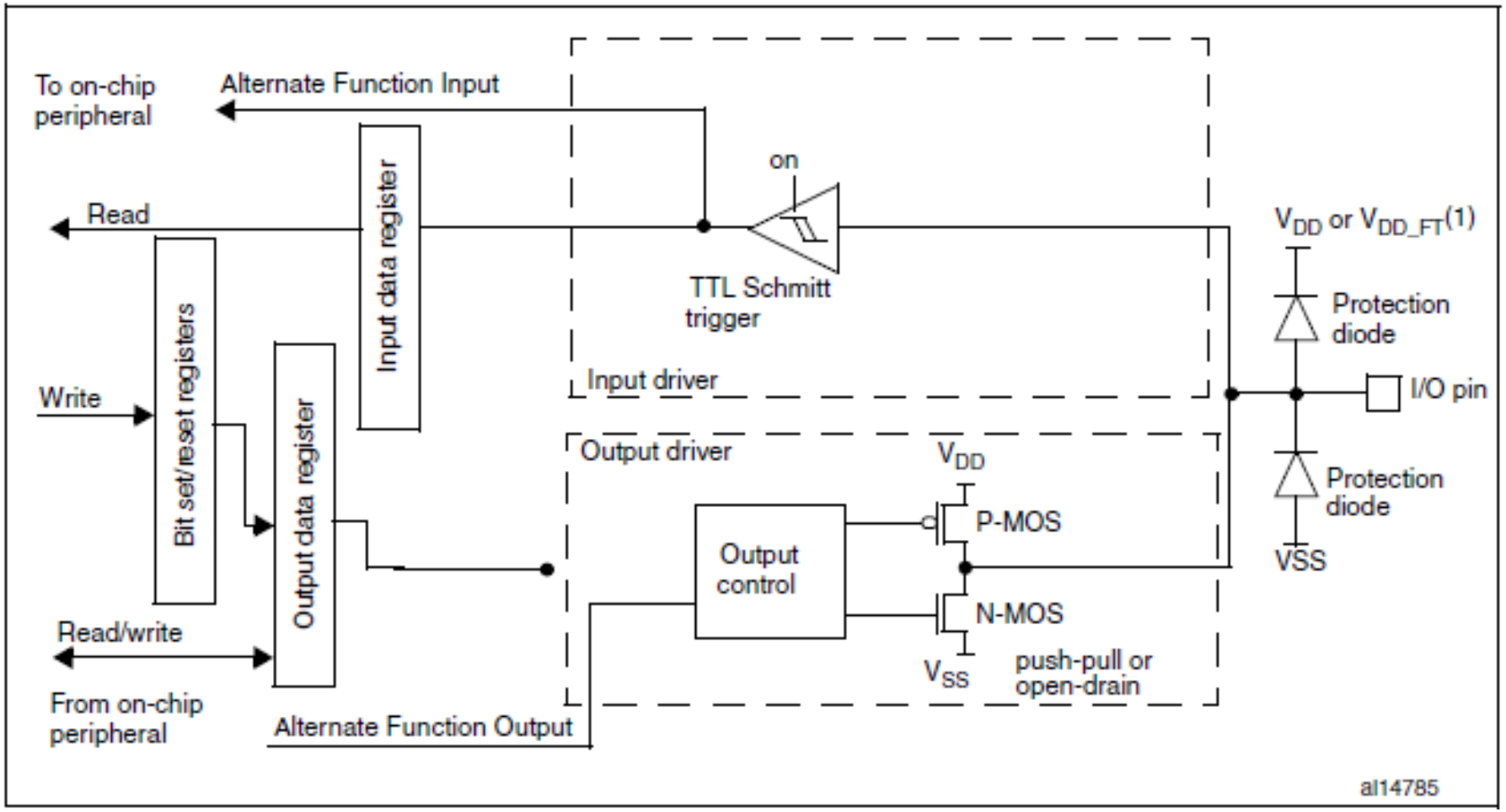
Output configurations



Alternate function (input/output)

- Output buffer is turned on in open drain or push-pull
- The Schmitt Trigger is activated
- The weak pull up and pull down are disabled
- The data is sampled into Input Data Register every APB2 clock
- In open drain mode: a read will access to Input Data Register and get I/O state
- In push/pull mode: a read will access to Output Data register and get last written value

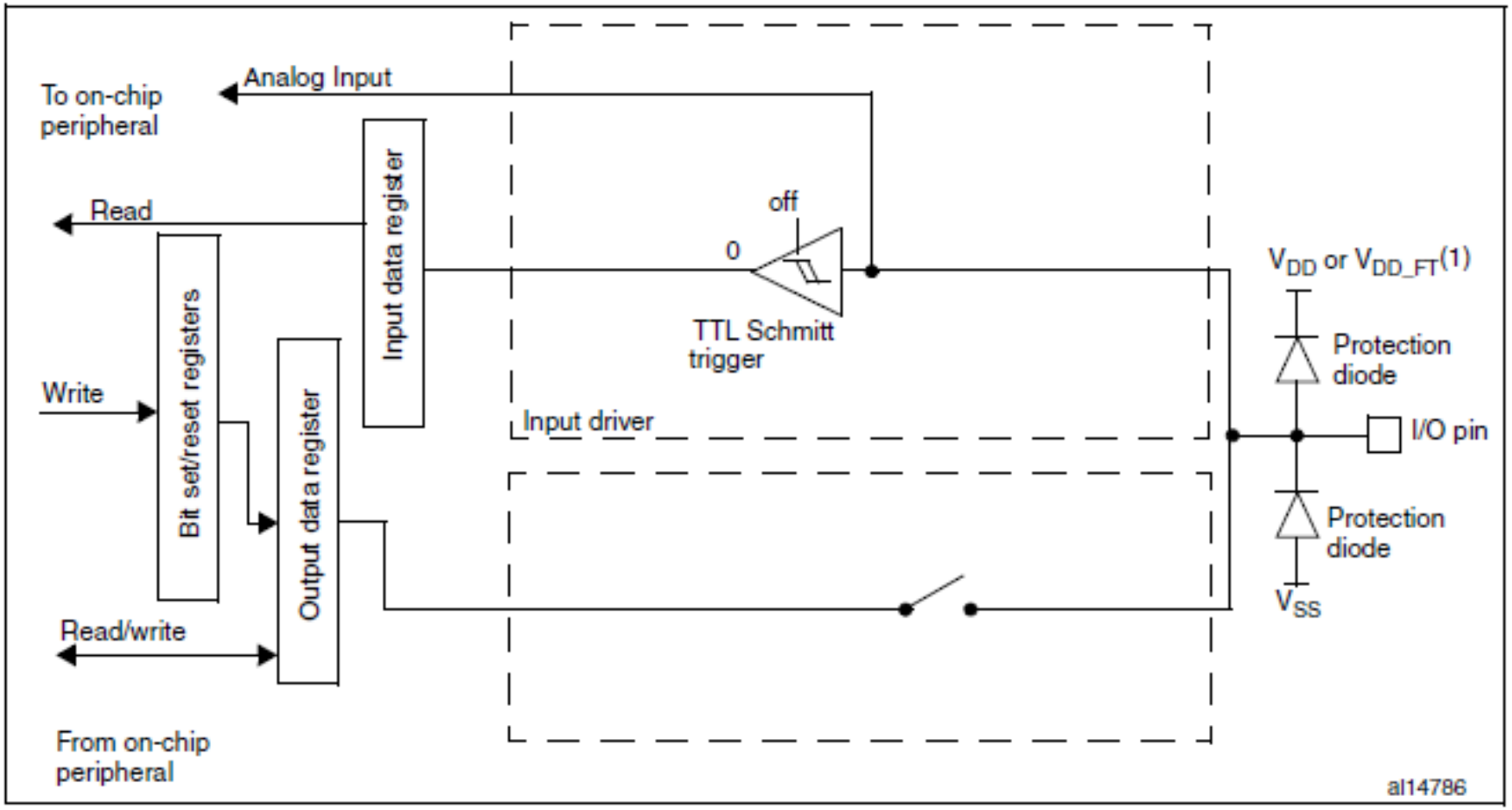
Alternate function configuration



Analog configuration

- Output buffer is disabled
- The Schmitt Trigger is deactivated
- The weak pull-up pull-down are disabled
- Read access to Input Data Register gets the value “0”

Analog configuration



GPIO init function

Can setup

- GPIO pin: can specify which pin to enable
- GPIO: speed 2MHz, 10MHz, and 50MHz
- GPIO mode:

  AIN = input analog

  IN_FLOATING = input float

  IPD = input down

  IPU = input up

  Out_OD = output open drained

  Out_PP = output push-pull

  AF_OD = alternate open drained

  AF_PP = alternate push-pull

GPIO_ReadInputDatabit

Format:

bool
GPIO_ReadInputDataBit(GPIO_TypeDef*, GPIO_Pin);

GPIO_TypeDef = GPIO peripheral can be any port from A to E

GPIO_Pin = pin number

GPIO_Writebit

Format:

void GPIO_Writebit(GPIO_TypeDef*, GPIO_Pin,BitVal);

GPIO_TypeDef = GPIO peripheral can be any port from A to E

GPIO_Pin = pin number

BitVal = the written value

Example of codes

```
void  GPIOInit(){

    GPIO_InitTypeDef GPIO_InitStructure;
    // Initial LED PB[8..15]
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOB, ENABLE);
    // provide the clock signal for GPIOB
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_Out_PP; // push pull mode
    GPIO_InitStructure.GPIO_Pin =       GPIO_Pin_8 |
                        GPIO_Pin_9 |        GPIO_Pin_10 |
                        GPIO_Pin_11 |       GPIO_Pin_12 |
                        GPIO_Pin_13 |       GPIO_Pin_14 |
                        GPIO_Pin_15;
    GPIO_Init(GPIOB, &GPIO_InitStructure);
}
```

```
int main(void)
{
    RCC_Configuration(); // System Clocks Configuration
    NVIC_Configuration(); // NVIC Configuration
    GPIOInit(); // Init GPIO

    while(1){
        LED1_HI();          LED2_LO();
        LED3_HI();          LED4_LO();
        LED5_HI();          LED6_LO();
        LED7_HI();          LED8_LO();

        DelaymS(500);

        LED1_LO();          LED2_HI();
        LED3_LO();          LED4_HI();
        LED5_LO();          LED6_HI();
        LED7_LO();          LED8_HI();

        DelaymS(500);
    }
}
```

```
#define  LED1_PIN                GPIO_Pin_8                    // LED1 = PB[8]
#define  LED2_PIN                GPIO_Pin_9                    // LED2 = PB[9]
#define  LED3_PIN                GPIO_Pin_10                   // LED3 = PB[10]
#define  LED4_PIN                GPIO_Pin_11                   // LED4 = PB[11]
#define  LED5_PIN                GPIO_Pin_12                   // LED5 = PB[12]
#define  LED6_PIN                GPIO_Pin_13                   // LED6 = PB[13]
#define  LED7_PIN                GPIO_Pin_14                   // LED7 = PB[14]
#define  LED8_PIN                GPIO_Pin_15                   // LED8 = PB[15]
#define  LED_PORT                GPIOB
#define  RCC_APB2Periph_GPIO_LED     RCC_APB2Periph_GPIOB

#define  LED1_HI()               GPIO_WriteBit(LED_PORT,LED1_PIN,Bit_SET)
#define  LED1_LO()               GPIO_WriteBit(LED_PORT,LED1_PIN,Bit_RESET)

#define  LED2_HI()               GPIO_WriteBit(LED_PORT,LED2_PIN,Bit_SET)
#define  LED2_LO()               GPIO_WriteBit(LED_PORT,LED2_PIN,Bit_RESET)

#define  LED3_HI()               GPIO_WriteBit(LED_PORT,LED3_PIN,Bit_SET)
#define  LED3_LO()               GPIO_WriteBit(LED_PORT,LED3_PIN,Bit_RESET)

#define  LED4_HI()               GPIO_WriteBit(LED_PORT,LED4_PIN,Bit_SET)
#define  LED4_LO()               GPIO_WriteBit(LED_PORT,LED4_PIN,Bit_RESET)
```

```
#define  LED5_HI()          GPIO_WriteBit(LED_PORT,LED5_PIN,Bit_SET)
#define  LED5_LO()          GPIO_WriteBit(LED_PORT,LED5_PIN,Bit_RESET)

#define  LED6_HI()          GPIO_WriteBit(LED_PORT,LED6_PIN,Bit_SET)
#define  LED6_LO()          GPIO_WriteBit(LED_PORT,LED6_PIN,Bit_RESET)

#define  LED7_HI()          GPIO_WriteBit(LED_PORT,LED7_PIN,Bit_SET)
#define  LED7_LO()          GPIO_WriteBit(LED_PORT,LED7_PIN,Bit_RESET)

#define  LED8_HI()          GPIO_WriteBit(LED_PORT,LED8_PIN,Bit_SET)
#define  LED8_LO()          GPIO_WriteBit(LED_PORT,LED8_PIN,Bit_RESET)
```

GPIO Input Example

```
/* Initial SW PA0 = Input */
 RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOA, ENABLE);

 GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
 GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN_FLOATING;
 GPIO_InitStructure.GPIO_Pin = GPIO_Pin_0;
 GPIO_Init(GPIOA, &GPIO_InitStructure);

 /* Initial SW PC13 = Input */
 RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOC, ENABLE);

 GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
 GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN_FLOATING;
 GPIO_InitStructure.GPIO_Pin = GPIO_Pin_13;
 GPIO_Init(GPIOC, &GPIO_InitStructure);
```

```
void LEDsSet (unsigned int State){
  GPIO_WriteBit(GPIOB,GPIO_Pin_8  ,((State&0x01))?Bit_SET:Bit_RESET);
  GPIO_WriteBit(GPIOB,GPIO_Pin_9  ,((State&0x02))?Bit_SET:Bit_RESET);
  GPIO_WriteBit(GPIOB,GPIO_Pin_10 ,((State&0x04))?Bit_SET:Bit_RESET);
  GPIO_WriteBit(GPIOB,GPIO_Pin_11 ,((State&0x08))?Bit_SET:Bit_RESET);
  GPIO_WriteBit(GPIOB,GPIO_Pin_12 ,((State&0x10))?Bit_SET:Bit_RESET);
  GPIO_WriteBit(GPIOB,GPIO_Pin_13 ,((State&0x20))?Bit_SET:Bit_RESET);
  GPIO_WriteBit(GPIOB,GPIO_Pin_14 ,((State&0x40))?Bit_SET:Bit_RESET);
  GPIO_WriteBit(GPIOB,GPIO_Pin_15 ,((State&0x80))?Bit_SET:Bit_RESET);
}
```

```
int main(){
 init(); // init RCC, NVIC, GPIO
 while(1){
     LEDsSet(0x00);  // Default LED = OFF

    //[1] Read Switch PC13
     if(GPIO_ReadInputDataBit(GPIOC, GPIO_Pin_13) == Bit_RESET){
       do{
         LEDsSet(0x01);
       }while(GPIO_ReadInputDataBit(GPIOC, GPIO_Pin_13) == Bit_RESET);
     }

     //[2] Read Switch PA0
     if(GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_0) == Bit_RESET){
       do{
         LEDsSet(0x80);
       }while(GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_0) == Bit_RESET);
     }
  }
}
```

Question?

- Write a program to read from switch PC13, if it is on, make all LED on. Otherwise, make all LED off.

Questions?