GPIO on the STM32

Adapted from The Insiders Guide to the STM32

STM32F4 Discovery

General Purpose Input Output (GPIO)

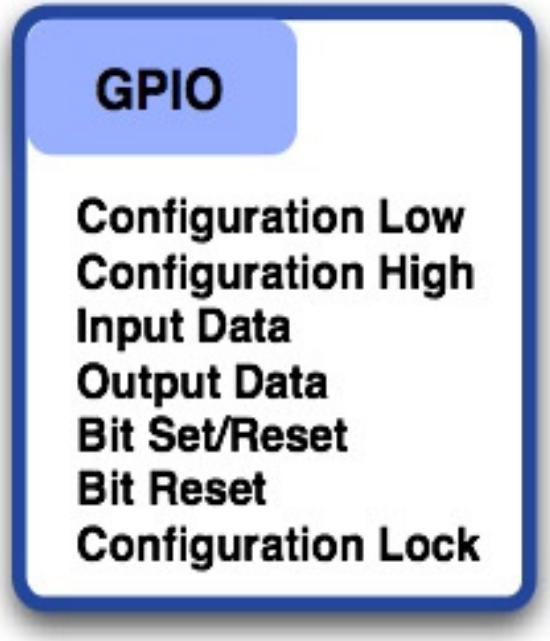
- The STM32 is well served with general purpose IO pins, having typically 80 bidirectional IO pins. The IO pins are arranged as five ports each having 16 IO lines.

Each digital pin may be configured as GPIO or as an alternate function. Each pin can simultaneously be configured as one of 16 external interrupt lines.

GPIO on the STM32

- These ports are named A-E and are all 5v tolerant. Many of the external pins may be switched from general purpose IO to serve as the Input/Output of a user peripheral, for example a USART or I2C peripheral.
- Additionally there is an external interrupt unit which allows 16 external interrupt lines to be mapped onto any combination of GPIO lines.

GPIO configuration



Each GPIO port can configure individual pins as input or output with different driver configurations. It has registers to write word-wide or for atomic bit manipulation. Once a configuration is defined it can be locked.

GPIO Configuration

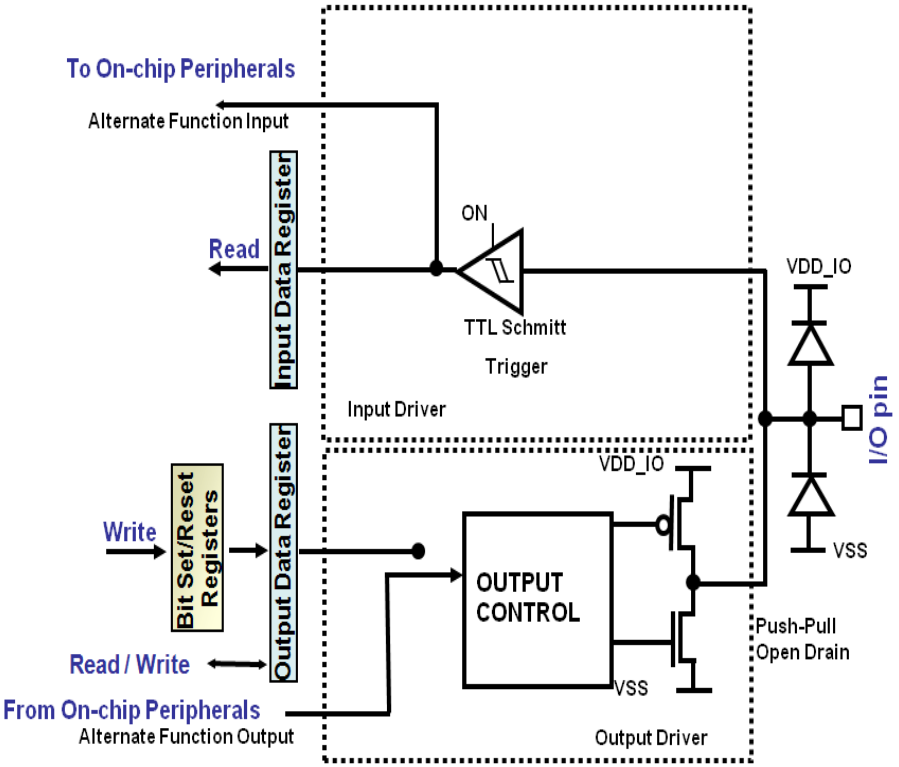
- Each GPIO port has two 32-bit wide configuration registers; these two registers combine to give a 64-bit wide configuration register.
- Within these 64 bits each pin has a four bit field that allows its characteristics to be defined. The four bit configuration field is made up of a two bit wide mode field and a two bit wide configuration field.
- The mode field allows the user to define the pin as an input or an output, while the configuration field defines the drive characteristics:

GPIO Configuration

- As well as being able to define a port pin as an input or output, its drive characteristics may also be selected.
- In the case of an input, an internal resistor can be connected as a pull up or pull down resistor.
- For an output, each port pin may be configured with a push pull or an open drain driver. Each output pin can also be configured with a maximum output speed of 2MHz,10MHz or 50MHz.

GPIO Configuration

Configuration Mode	CNF1	CNF0	MOD1	MOD0
Analog Input	0	0	00	
Input Floating **(Reset State)**	0	1		
Input Pull-Up	1	0		
Input Pull-Down	1	0		
Output Push-Pull	0	0	00: Reserved 01: 10 MHz 10: 2 MHz 11: 50 MHz	
Output Open-Drain	0	1		
AF Push-Pull	1	0		
AF Open-Drain	1	1		



GPIO Configuration

```
/*Initialise GPIOD bit 12 as output connected to the green (left) LED
on discovery board */

void LED_Init_1 (void) {

  RCC->AHB1ENR    |= ((1UL <<  3) );      /* Enable GPIOD clock  */
  GPIOD->MODER    &= ~((3UL << 2*12));    /* PD.12  is output    */
  GPIOD->MODER    |=  ((1UL << 2*12)  );
  GPIOD->OTYPER   &= ~((1UL <<   12) );   /* PD.12 output Push-Pull */
  GPIOD->OSPEEDR  &= ~((3UL << 2*12) );   /* PD.12 50MHz Fast Speed */
  GPIOD->OSPEEDR  |=  ((2UL << 2*12) );
  GPIOD->PUPDR    &= ~((3UL << 2*12) );   /* PD.12 is Pull up      */
  GPIOD->PUPDR    |=  ((1UL << 2*12) );
}
```

GPIO port mode register: `GPIOD->MODER`

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]	
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]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 2y:2y+1 **MODERy[1:0]:** Port x configuration bits (y = 0..15)
These bits are written by software to configure the I/O direction mode.
00: Input (reset state)
01: General purpose output mode
10: Alternate function mode
11: Analog mode

2 bits in the register define the properties of 1 Port pin

```
GPIOD->MODER     &= ~((3UL << 2*12));
GPIOD->MODER     |=  ((1UL << 2*12));
```

GPIO Data Registers

- The input and output data registers allow port wide access to the IO pins. Atomic bit manipulation is supported by either using the Cortex bit banding technique on the input and output data registers, or through two dedicated bit manipulation registers.

```
/* Using bit masking */
GPIOD->ODR = GPIOD->ODR | 0x00001000;

/* 0x00001000   =  0000 0000 0000 0000 0001
0000 0000 0000, ODR is the Output Data
Register */
```

GPIOD->ODR &= ~(1UL << 12);

/* 1UL << 12 = 1 bitwise shifted left 12 places every bit is then inverted using the complement (bit inversion) operator

1UL = (UL = unsigned Long):
0000 0000 0000 0000 0000 0000 0000 0001
or 0x00000001

<<12 Bitwise shifted left 12 places :
0000 0000 0000 0000 0001 0000 0000 0000
or 0x00001000

~invert
1111 1111 1111 1111 1110 1111 1111 1111

& AND with the current status of ODR:
Setting bit 12 only to zero all other pins remain unchanged.

Bit Set/Reset Register (BSRR)

- The bit set/reset register is a 32-bit wide register. The upper 16 bits are mapped to each port pin. Writing a logic 1 to these locations will reset the matching port pin. Similarly, writing a logic 1 to any of the lower 16 bits will set the matching port pin.
- The second bit manipulation register is a bit reset register. This is a 16-bit wide register where writing logic 1 in the lower 16 bits will reset the matching port pin.

Bit Set/Reset Register (BSRR)

```
/* BSRRL refer to bits 0-15 of BSRR which set the
corresponding bit of the port bits 0 - 15 of a port */

/* BSRRH refer to bits 16-31 of BSRR which reset the
corresponding bit of the port bits 0 - 15 of a port */

GPIOD->BSRRL = 0x00001000;

GPIOD->BSRRH = 0x00001000;
```

Denary, Hex & Binary

Denary	Hex	Binary			
0	0	0	0	0	0
1	1	0	0	0	1
2	2	0	0	1	0
3	3	0	0	1	1
4	4	0	1	0	0
5	5	0	1	0	1
6	6	0	1	1	0
7	7	0	1	1	1
8	8	1	0	0	0
9	9	1	0	0	1
10	A	1	0	1	0
11	B	1	0	1	1
12	C	1	1	0	0
13	D	1	1	0	1
14	E	1	1	1	0
15	F	1	1	1	1