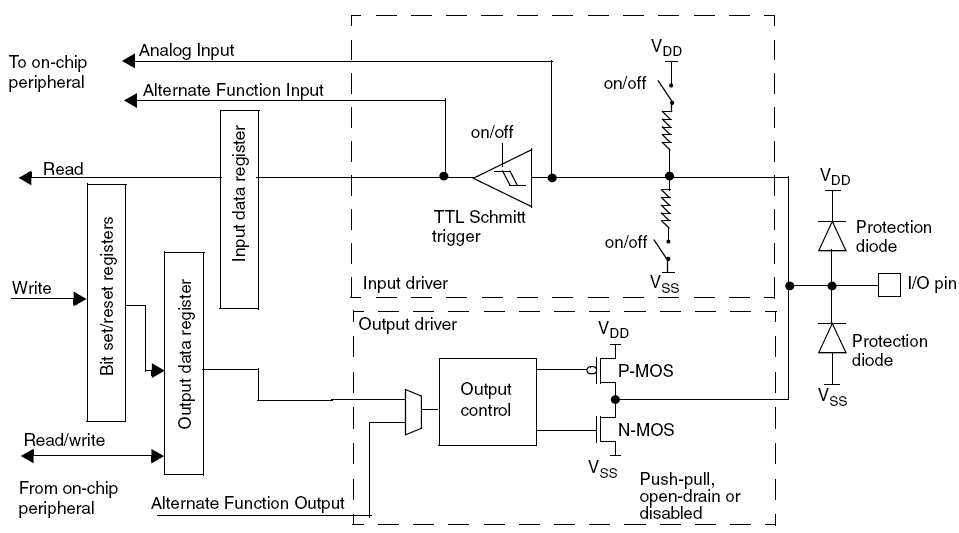
**GPIO Functional description**

* each GPIO ports has
  + Two 32-bit configuration registers (GPIOx\_CPL, GPIOx\_CRH)
  + Two 32-bit Data registers (GPIOx\_IDR, GPIOx\_ODR)
  + One 32-bit Set/Reset register (GPIOx\_BSRR)
  + One 16-bit Reset register (GPIOx\_BRR)
  + One 32-bit locking register (GPIOx\_LCKR)
* each pins in a port can be configured as
  + Input floating
  + Input pull-up
  + Input pull-down
  + Analog
  + Output open-drain
  + Output push-pull
  + Alternate function push-pull
  + Alternate function open-drain



**Port Bit configuration**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Configuration Mode** | | **CNF[1:0]**  **(I/O type)** | | **MODE[1:0]( output max speed)** |
| General Purpose Output | Push-pull | 0 | 0 | 00 – Reserved  01 – 10MHz output  10 – 20MHz output  11 – 50MHz output |
| Open-drain | 0 | 1 |
| Alternate Purpose output | Push-pull | 1 | 0 |
| Open-drain | 1 | 1 |
| Input | Analog | 0 | 0 | 00 |
| Input Floating | 0 | 1 |
| Input pull-down | 1 | 0 |
| Input pull-up | 1 |

**for input pull-up and pull-down, for pull down- ODR =0, for pull-up – ODR =1**

**Note:**

* After reset, the alternate functions are not active and I/O ports are Input Floating Mode.
* All ports can have external interrupt capability with wakeup controlled by External Interrupt/Event Controller (***EXTI***).

**Alternate functions**

|  |  |
| --- | --- |
| Alternate function input | Floating, pull-up, pull-down |
| Alternate function output | Alternate Function Output – push-pull or open-drain |
| Bidirectional Alternate function | Alternate Function Output – push-pull or open-drain |

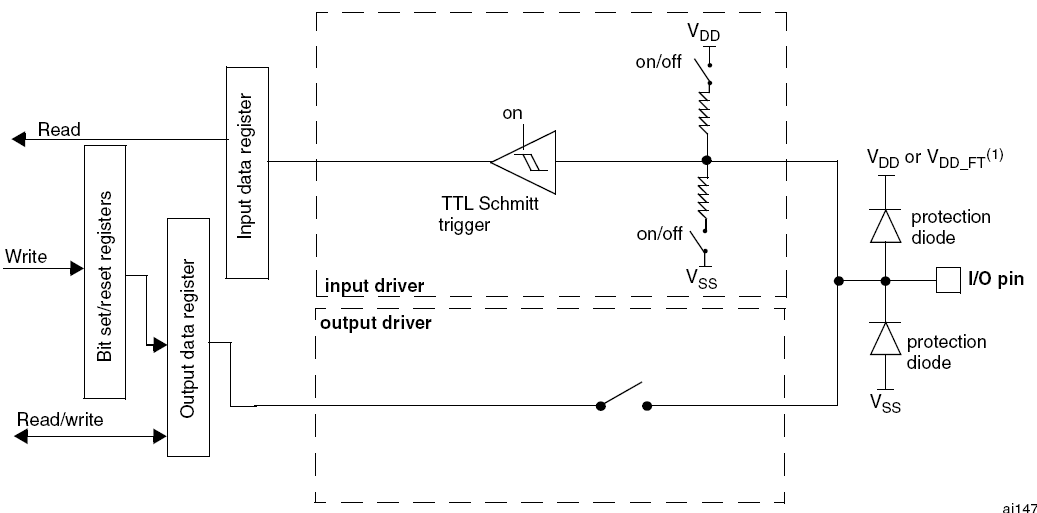
**Software remapping of I/O alternate function**

* possible to remap some alternate function to some other function
* achieved by programming the corresponding register ***AFIO***

**GPIO Locking mechanism**

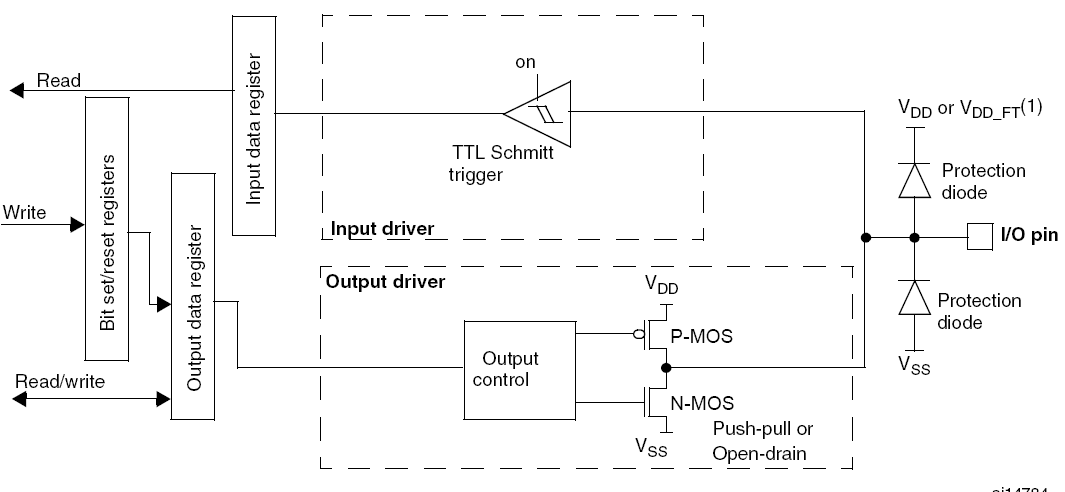
* allows the I/O configuration to be frozen
* Applying LOCK sequence locks the bits not to be modified – until the next reset.

**Input Configuration**



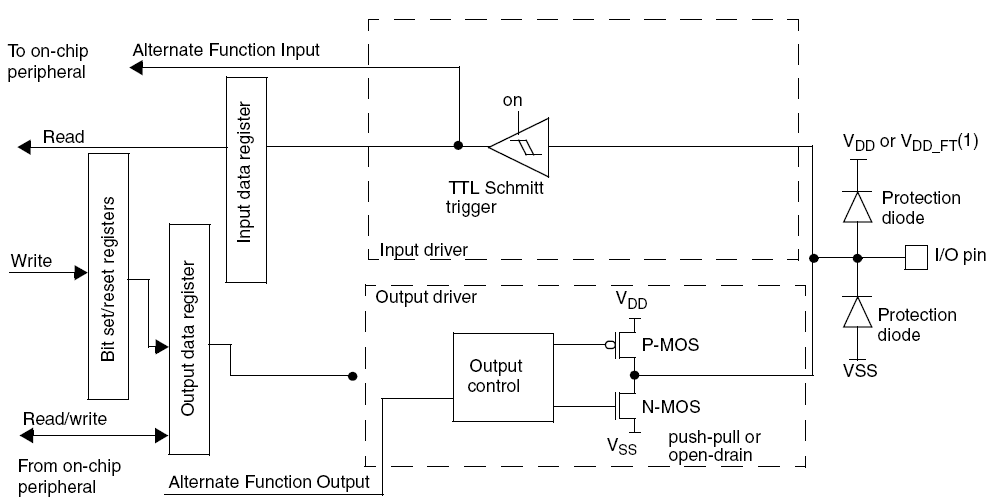
* Output Buffer is disabled
* Schmitt Trigger Input is activated
* The weak pull-up and weak pull-down resistors are activated depending on input configuration (pull-up, pull-down, floating).
* The data present on that I/O pin is sampled into Input Data Register every APB2 clock cycles.

**Output configuration**



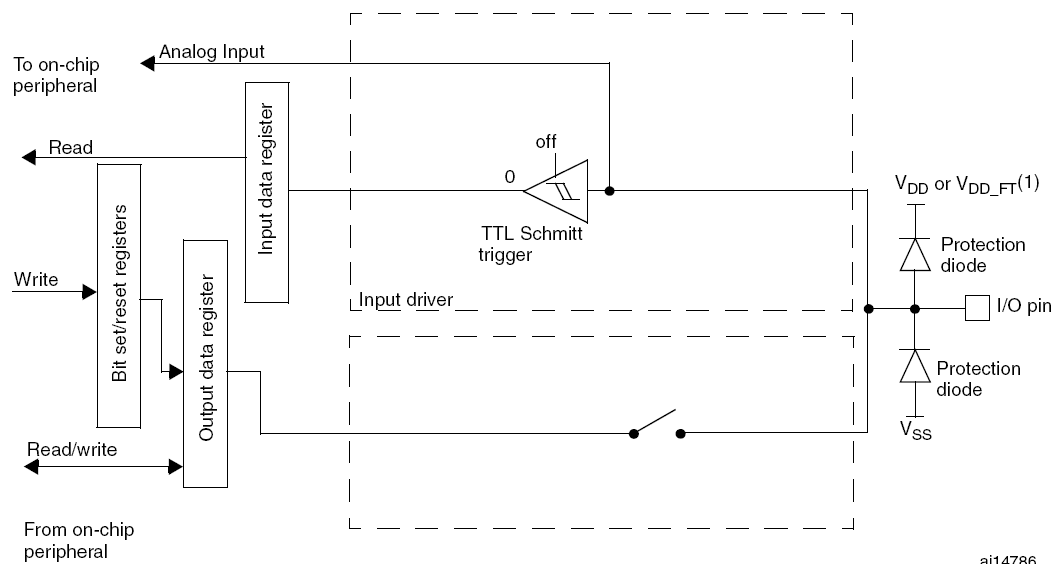
* The output Buffer is enabled
  + **Open Drain Mode**
    - **0** in Output register activates the N-MOS
    - **1** in Output register leaves the port in HI-Z
  + **Push-Pull Mode**
    - **0** in Output register activates the N-MOS
    - **1** in Output register activates the P-MOS
* Schmitt Trigger is activated
* weak pull-up and pull-down is disabled
* The data present on that I/O pin is sampled into Input Data Register every APB2 clock cycles.
* Read access to input data register gets I/O state in open drain mode
* Read access to output data register gets last written value in push-pull mode

**Alternate function configuration**



* Output Buffer is turned on in Open Drain or Push-pull configuration
* Output Buffer is driven by signal coming from peripheral (alternate function out).
* Schmitt Trigger input is activated
* weak pull-up and pull-down resistors are disabled
* The data present on that I/O pin is sampled into Input Data Register every APB2 clock cycles.
* Read access to input data register gets I/O state in open drain mode
* Read access to output data register gets last written value in push-pull mode

**Analog Configuration**



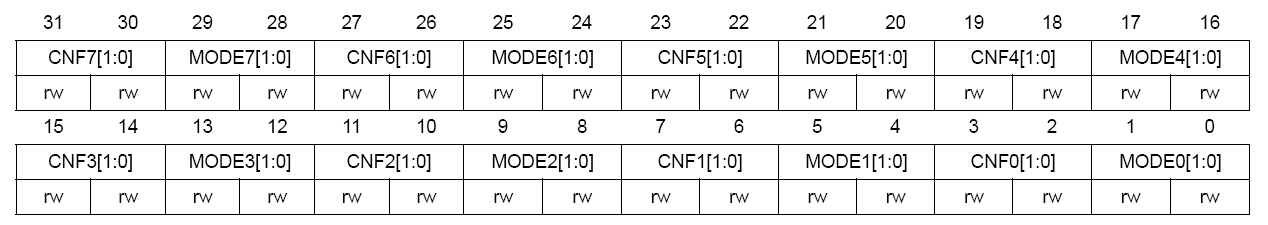
* Output Buffer is disabled
* Schmitt Trigger input is de-activated providing zero consumption for every analog value of I/O pin.
* Weak pull-up and weak pull-down registers are disabled.
* Read access to Input data registers gets the value “0”.

**GPIO pins and corresponding configuration**

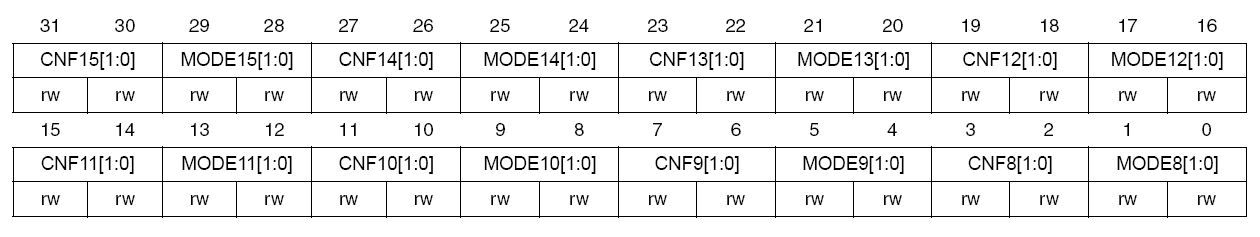
* See Table 5 of **stm32\_MD\_charect.pdf** and GPIO section(9.1.11) of **stm3210x.pdf**

**GPIO Registers**

**Port configuration register Low (GPIOx\_CRL) (x=A:G)**

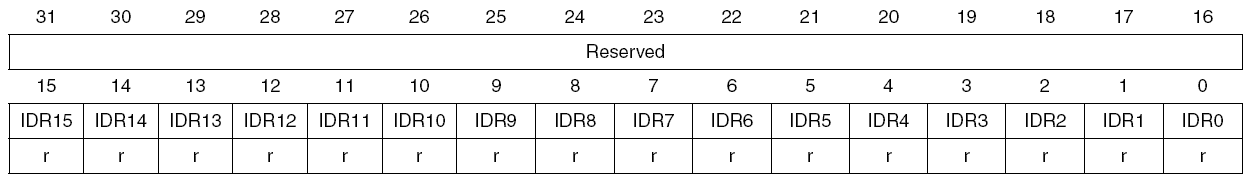


* for GPIO pins 0 to 7
* You can use formula to solve this

**Port configuration register high (GPIOx\_CRH) (x=A..G)**

* for GPIO pins 8 to 15
* You can use formula to solve this

**Port input data register (GPIOx\_IDR) (x=A..G)**

 **Port output data register (GPIOx\_ODR) (x=A..G)**



**Port bit set/reset register (GPIOx\_BSRR) (x=A..G)**

* to set and reset bits



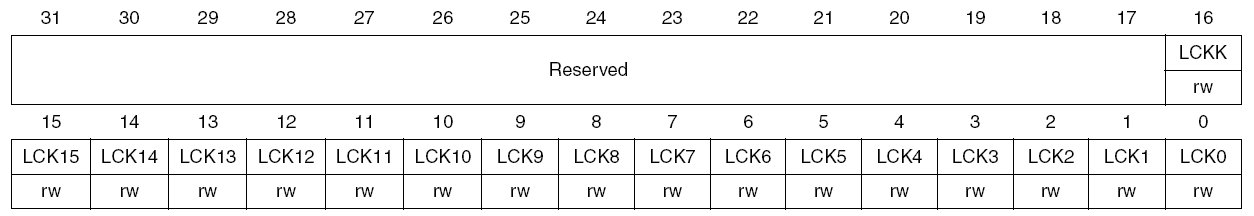
**Port bit reset register (GPIOx\_BRR) (x=A..G)**

* to reset bits



**Port configuration lock register (GPIOx\_LCKR) (x=A..G)**

* lock the configuration of the port bits
* freezes the ***CRL***, ***CRH*** bits
* lock is done by writing the corresponding lock bits and 16th bit (***LCKK***)



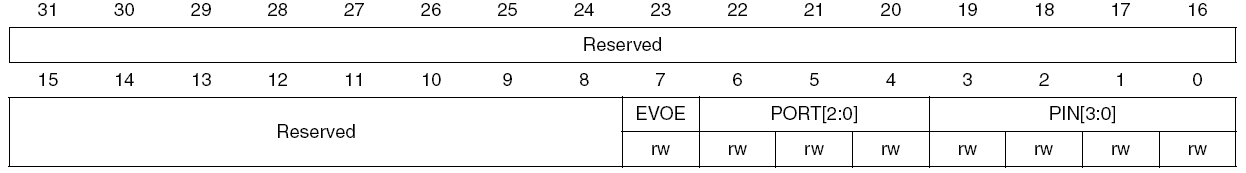
**Lock sequence**

* + Write 1
  + Write 0
  + Write 1
  + Read 0
  + Read 1

**AFIO Registers**

* **AFIO** clock should be enabled in ***RCC\_APB2ENR***

**Event control register (AFIO\_EVCR)**



* ***EVENTOUT*** Cortex output is connected to said pins

**EVOE – Event output enable**

* to connect the EVENTOUT output to pin selected by PIN[3:0] and PORT[2:0]

**PORT [2:0] – selection port for EVENTOUT output pin**

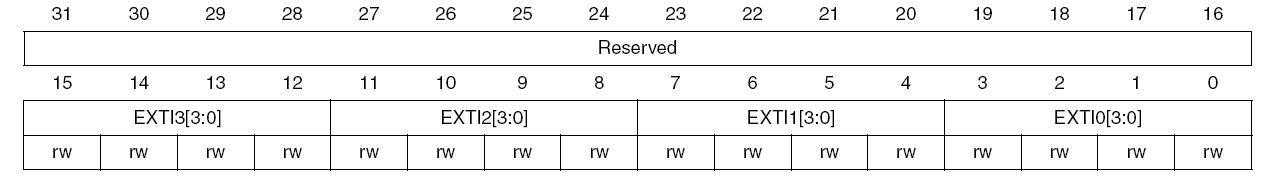
|  |  |
| --- | --- |
| **PORT[2:0]** | **Port Selected** |
| **000** | **PA** |
| **001** | **PB** |
| **010** | **PC** |
| **011** | **PD** |
| **100** | **PE** |

**PIN[3:0] – selection pin for EVENTOUT output pin**

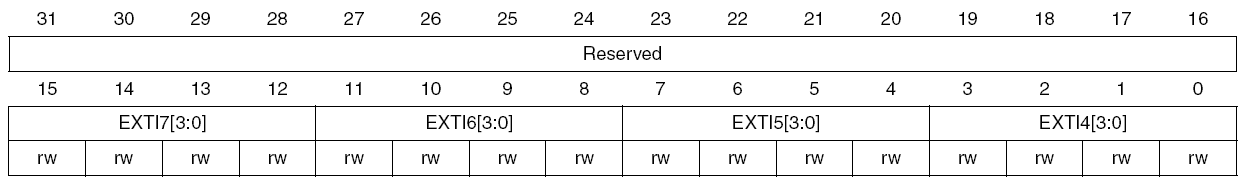
|  |  |
| --- | --- |
| **PIN[3:0]** | **Pins selected** |
| **0000** | **P0** |
| **..** |  |
| **..** |  |
| **1111** | **P15** |

**External interrupt configuration register 1 (AFIO\_EXTICR1)**

* to select source input for EXTIx external interrupt for pins 0 to 3

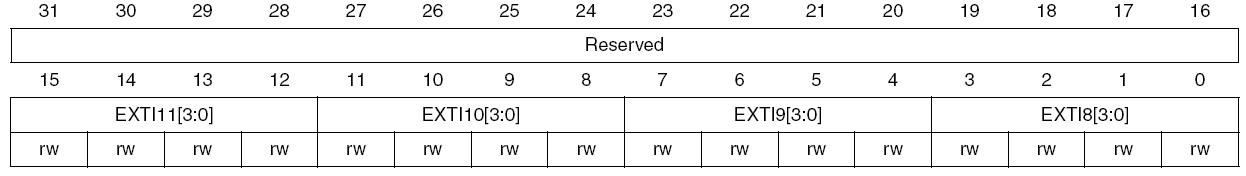
**External interrupt configuration register 2 (AFIO\_EXTICR2)**

* to select source input for EXTIx external interrupt for pins 4 to 7



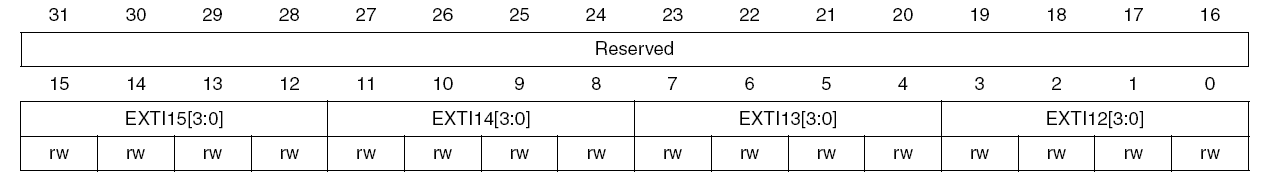
**External interrupt configuration register 3 (AFIO\_EXTICR3)**

* to select source input for EXTIx external interrupt for pins 8 to 11



**External interrupt configuration register 4 (AFIO\_EXTICR4)**

* to select source input for EXTIx external interrupt for pins 12 to 15



|  |  |
| --- | --- |
| **EXTIx[3:0]** | **Port pins selected** |
| 0000 | PA[x] |
| 0001 | PB[x] |
| 0010 | PC[x] |
| 0011 | PD[x] |
| 0100 | PE[x] |
| 0101 | PF[x] |
| 0110 | PG[x] |

**NOTE**

* **For setting and clearing IO pins, use**
  + **GPIOx->BSRR = (1<<pin);**
    - **and not GPIOx->BSRR |= (1<<pin);**
    - **because this will try to set again the other bits which results in errors**
  + **GPIOx->BRR = (1<<pin);**
    - **and not GPIOx->BRR |= (1<<pin);**
    - **because this will try to clear again the other bits which results in errors**
* **Also don’t’ use ODR for writing the output pins**