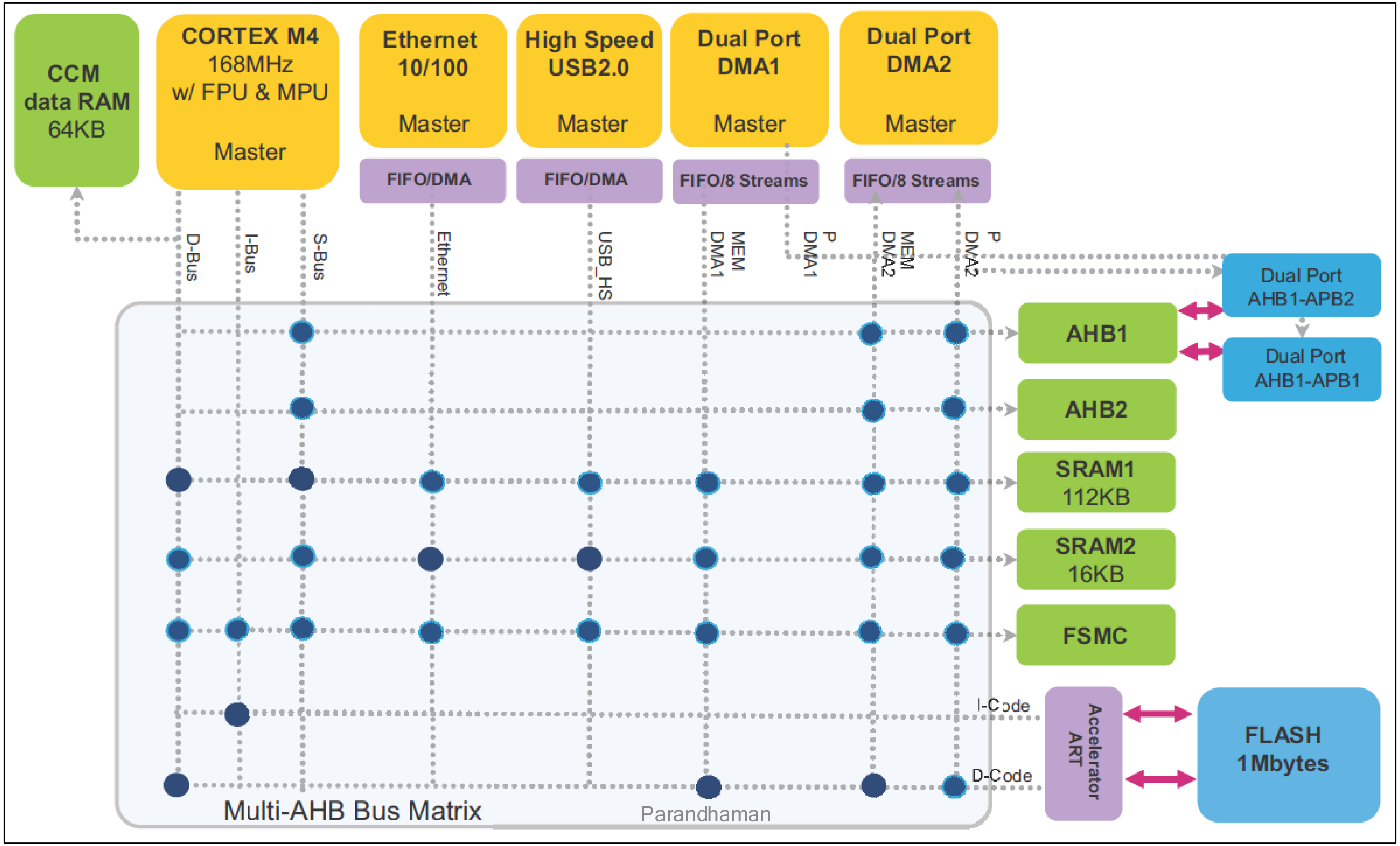
Introduction to STM32f407

Parandhaman

System Architecture



Power, Reset and Clock

Power controller

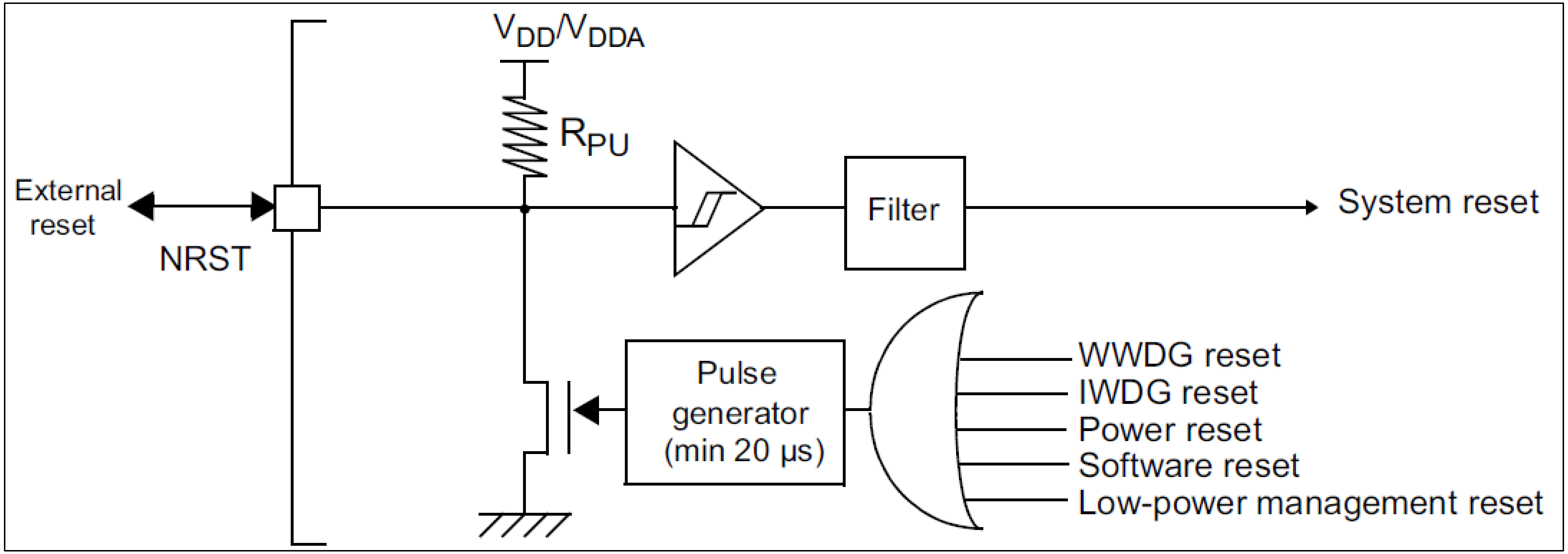
- There are 3 voltage rails which are available in STM32F407
  - V_{DD} → Powers everything excluding ADC
  - V_{DDA} → Powers ADC
  - V_{BAT} → powers Real time clock, RTC backup registers and backup SRAM when V_{DD} is turned off
- STM32F407 needs 1.8V – 3.6V as an input voltage
  - Internal linear voltage regulates input voltage into 1.2V

Reset

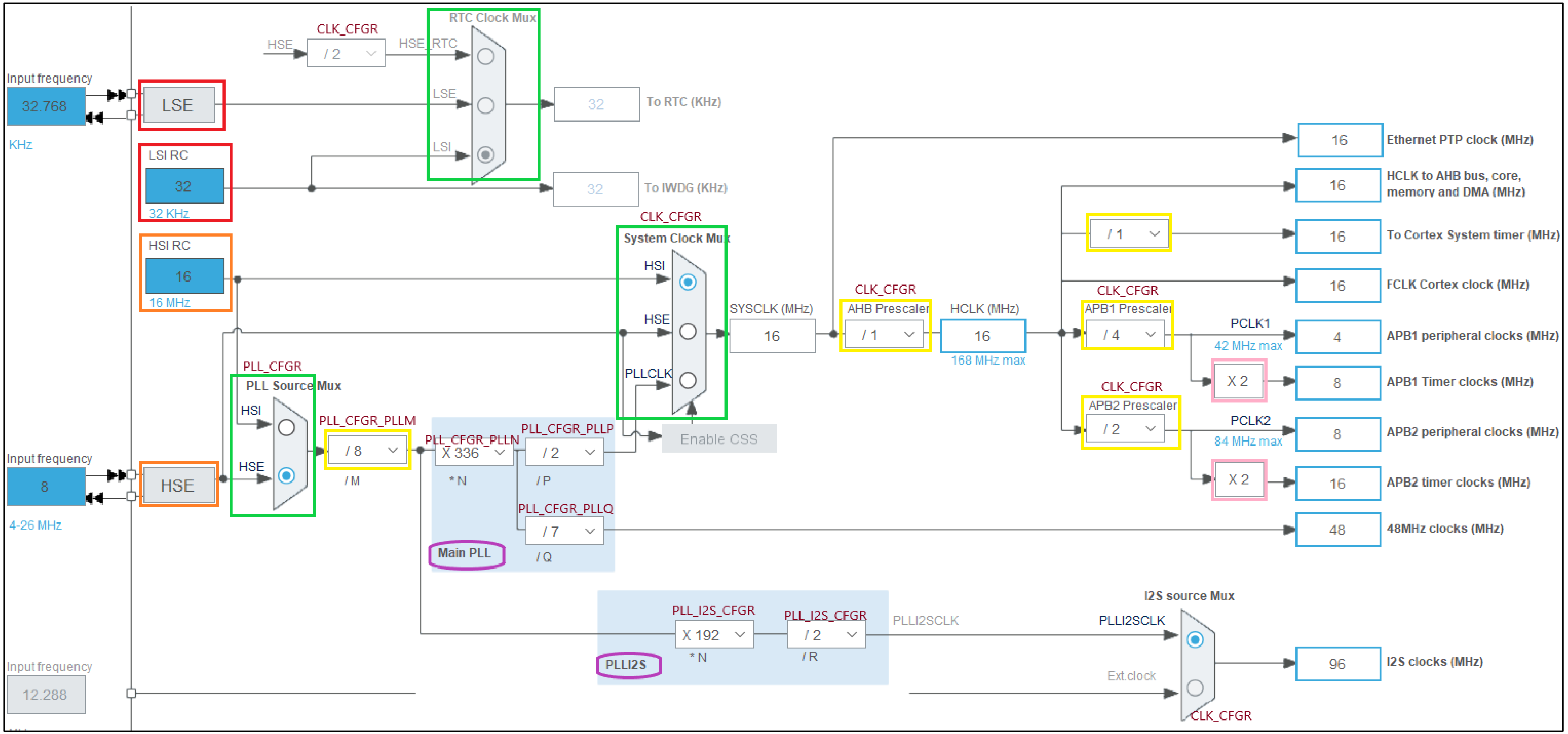
Reset type	Description	Sources
System reset	Resets all the registers to their reset value except, • Clock controller CSR registers • Backup domain registers	• Low level on NRST pin • Window watchdog end of count condition(WWDG reset) • Independent watchdog end of count condition(IWDG reset) • Software reset • Done by setting SYSRESETREQ bit from Application interrupt and reset control register (AIRCR) part of System control space of ARM processor • If we set SYSRESETREQ bit ARM processor will request external agent to reset the system (here stm32 RCC) • LPM reset • Reset when entering the STANDBY_MODE • Reset when entering the STOP_MODE
Power reset	Resets all the registers except backup domain	• Power on/Power down reset • Brownout (BOR) reset • Exiting the STANDBY_MODE
Backup domain reset		

Reset circuitry

- NRST is an external reset input signal (active low)



STM32F407 - Clock tree



Parandhaman

RCC registers

- RCC registers

Register	offset	Reset value	Access	Description
RCC Clock Control Register	0x00	0x0000_xx81		
RCC PLL Configuration Register	0x04	0x0000_0000		
RCC Clock Configuration Register	0x08	0x0000_0000		
RCC Clock Interrupt Register	0x0C	0x0000_0000		
RCC AHB1 Peripheral Reset Register	0x10	0x0000_0000		
RCC AHB2 Peripheral Reset Register	0x14	0x0000_0000		
RCC APB1 Peripheral Reset Register	0x20	0x0000_0000		
RCC APB2 Peripheral Reset Register	0x24	0x0000_0000		
RCC AHB1 Peripheral Clock Enable Register	0x30			
RCC AHB2 Peripheral Clock Enable Register	0x34			
RCC APB1 Peripheral Clock Enable Register	0x40			
RCC APB2 Peripheral Clock Enable Register	0x44			
RCC AHB1 Peripheral Clock Enable Register in Low power mode	0x50			
RCC AHB2 Peripheral Clock Enable Register in Low power mode	0x54			
RCC APB1 Peripheral Clock Enable Register in Low power mode	0x60			
RCC APB2 Peripheral Clock Enable Register in Low power mode	0x64			
RCC backup domain register	0x70			
RCC clock control and status register	0x74			
RCC spread spectrum clock generation register	0x80			
RCC PLLI2S configuration register	0x84			
RCC dedicated clock configuration register	0x8C			

GPIO

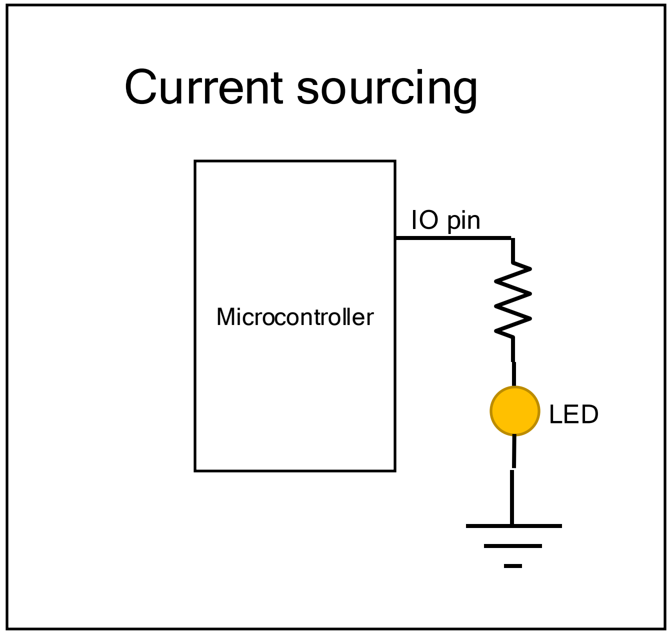
GPIO

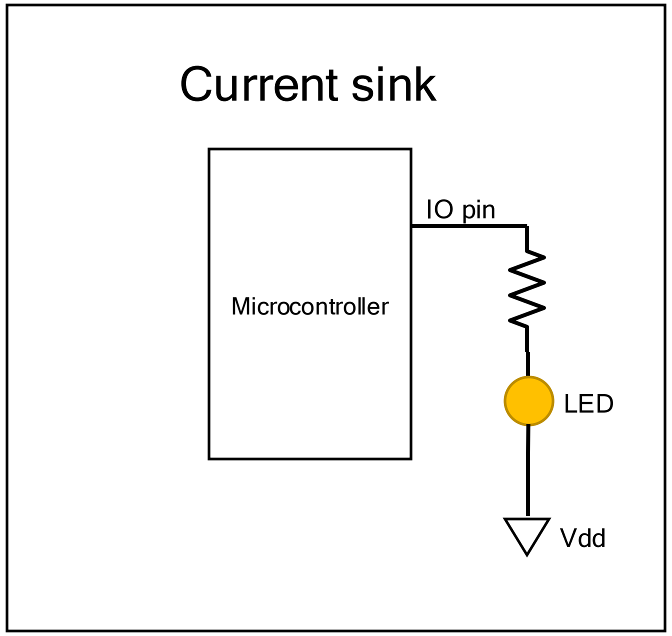
- GPIO ports are 16bit wide
- GPIO input states
  - HI-Z (High impedance)
  - Internal pull up or external pull up
  - Internal pull down or external pull down
- GPIO output states
  - Open drain (Internal pull up / external pull up)
  - Push pull
- At reset all the GPIO's in input mode with HI-Z (High impendence) state
- If we change GPIO mode into output mode then default configuration in push pull configuration
- If GPIO is configured as output mode with open drain with internal pull up resister, we can connect LED without any series current limiting resister
- Max Vin for GPIO's
  - There are 3 types of GPIO's (Check I/O structure of the GPIO to know this)

Type	Description	Minimum vg	Max vg	Max Current
TT	Three volt Tolerant	-0.3V	VDD+0.3V	• Max Sink current 25mA / GPIO • Max source current 25mA / GPIO • Max total source current 240mA
TTa	Three volt Tolerant connected to ADC	-0.3V	VDD+0.3V	
FT	Five volt Tolerant	-0.3V	5.5V (4V when device is off/vdd=0)	

GPIO-cont.

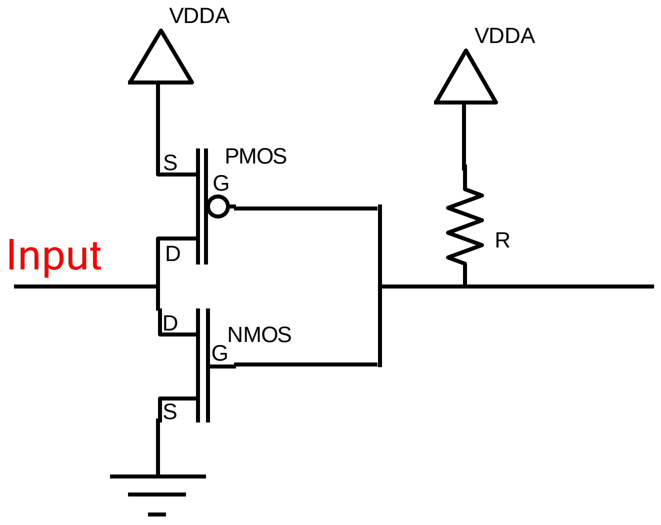
- What if I am using 20 GPIO's in output mode
  - 20 * 25mA = 500mA
  - Because of total current requirement was 500mA you won't get 25mA current at each GPIO because max current is 240mA
  - 240mA / 20 = 12mA
  - Hence you will get just 12mA current on each GPIO pin which is used
- Current sourcing vs Current sink
  - If current is sourced from VDD then its current sourcing
  - If current is sourced from external world then its current sink



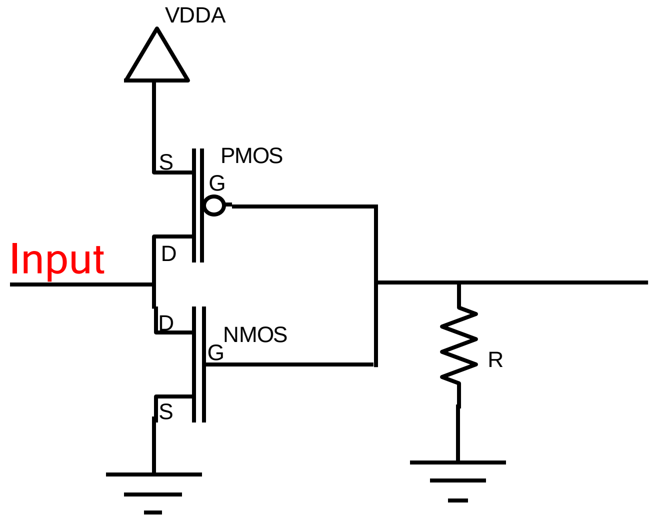


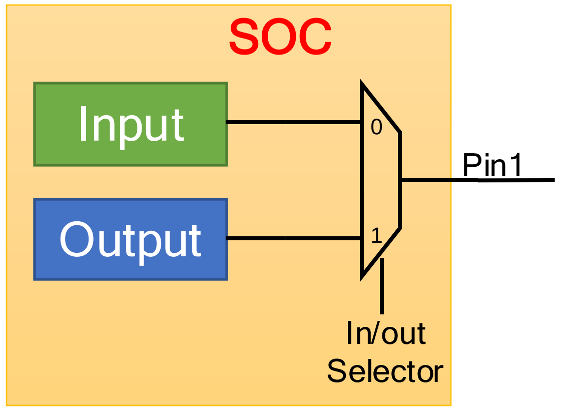
GPIO-cont.

Input mode Pull up



Input mode Pull down





When no input is connected to input pin it will be in **floating state** or it will be in **high impedance state (HI-Z)**

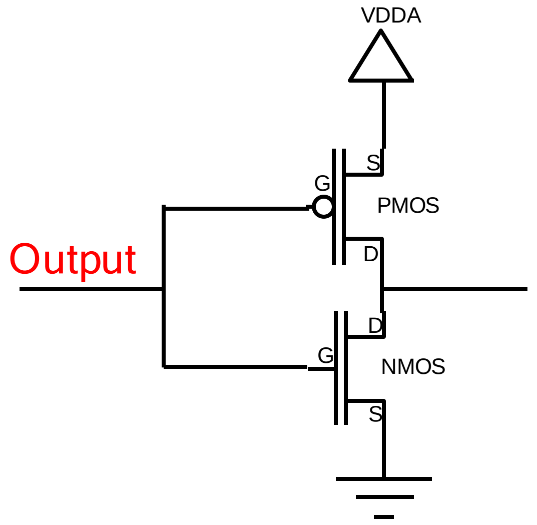
Why Pull up or pull down is required?

- Both transistors should never turned on, but

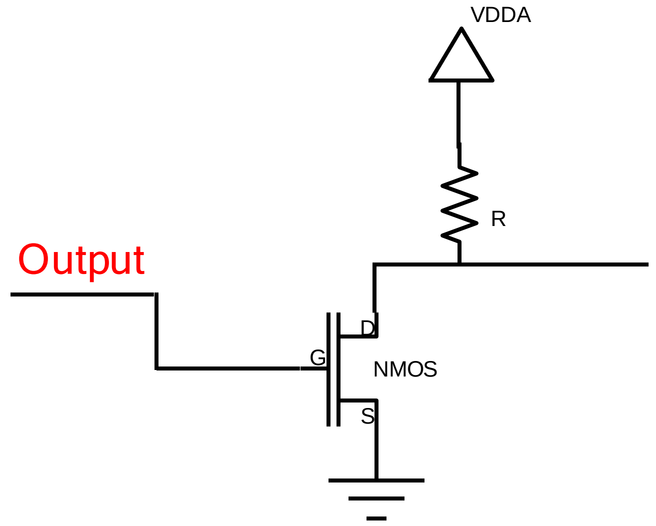
Due to some leakages in gate due to external noise Both transistors may be turned on
Ex, 2.5v due to noise given to the gate it's not high nor low hence both NMOS and PMOS transistors will be turned on, to avoid This pull up or pull down is required

- In modern processors **Schmitt trigger** is used

Output mode Push Pull



Output mode Open drain



GPIO registers

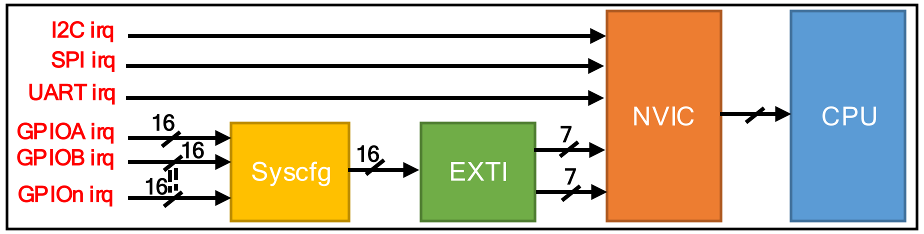
- For all the below registers only 0 to 22 bits are used, why?
  - Because we have only 23 external interrupt lines

Register	offset	Reset value	Access	Description
GPIO Port Mode Register	0x00	0x0C00_0000 PA 0x0000_0280 PB 0x0000_0000 Others		00: Input (reset state) 01: General purpose output mode 10: Alternate function mode 11: Analog mode
GPIO Port Output Type Register	0x04	0x0000_0000		0: Output push-pull (reset state) 1: Output open-drain
GPIO Port Output Speed Register	0x08	0x0000_0000		00: Low speed 01: Medium speed 10: High speed 11: Very high speed
GPIO Port PU/PD Register	0x0C	0x0000_0000		00: No pull-up, pull-down 01: Pull-up 10: Pull-down 11: Reserved
GPIO Port Input Data Register	0x10	0x0000_0000		
GPIO Port Output Data Register	0x14	0x0000_0000		
GPIO Port Bit Set/Reset Register	0x18	0x0000_0000		0-15 are used to set and 16-31 are used to clear
GPIO Port Configuration Lock Register	0x1C	0x0000_0000		Once we lock the configuration can't be changed until we assert the reset
GPIO Alternate Function Low Register	0x20	0x0000_0000		4 bits for each GPIO pin, hence there are 16 variants of alternate functions
GPIO Alternate Function High Register	0x24	0x0000_0000		

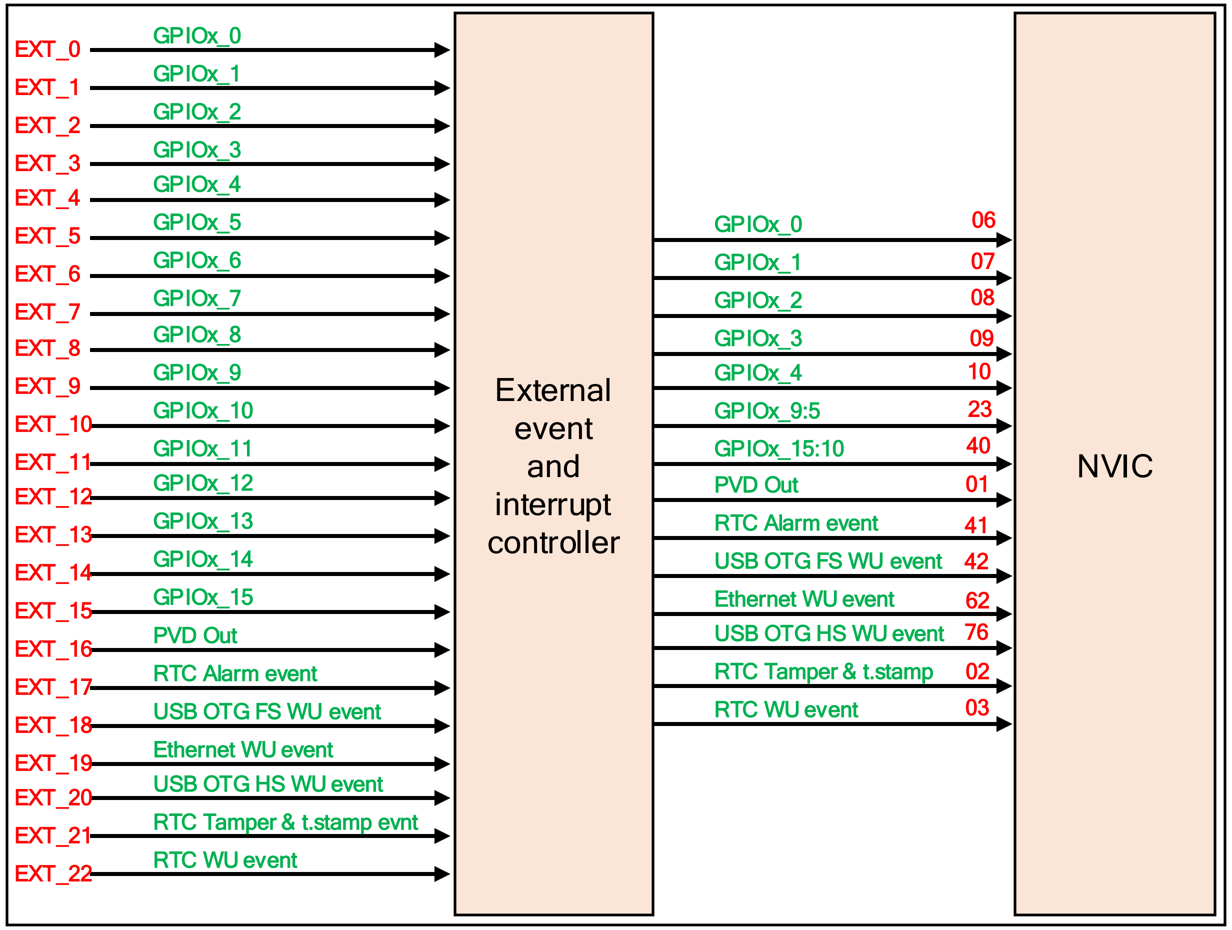
External Interrupt/Event Controller

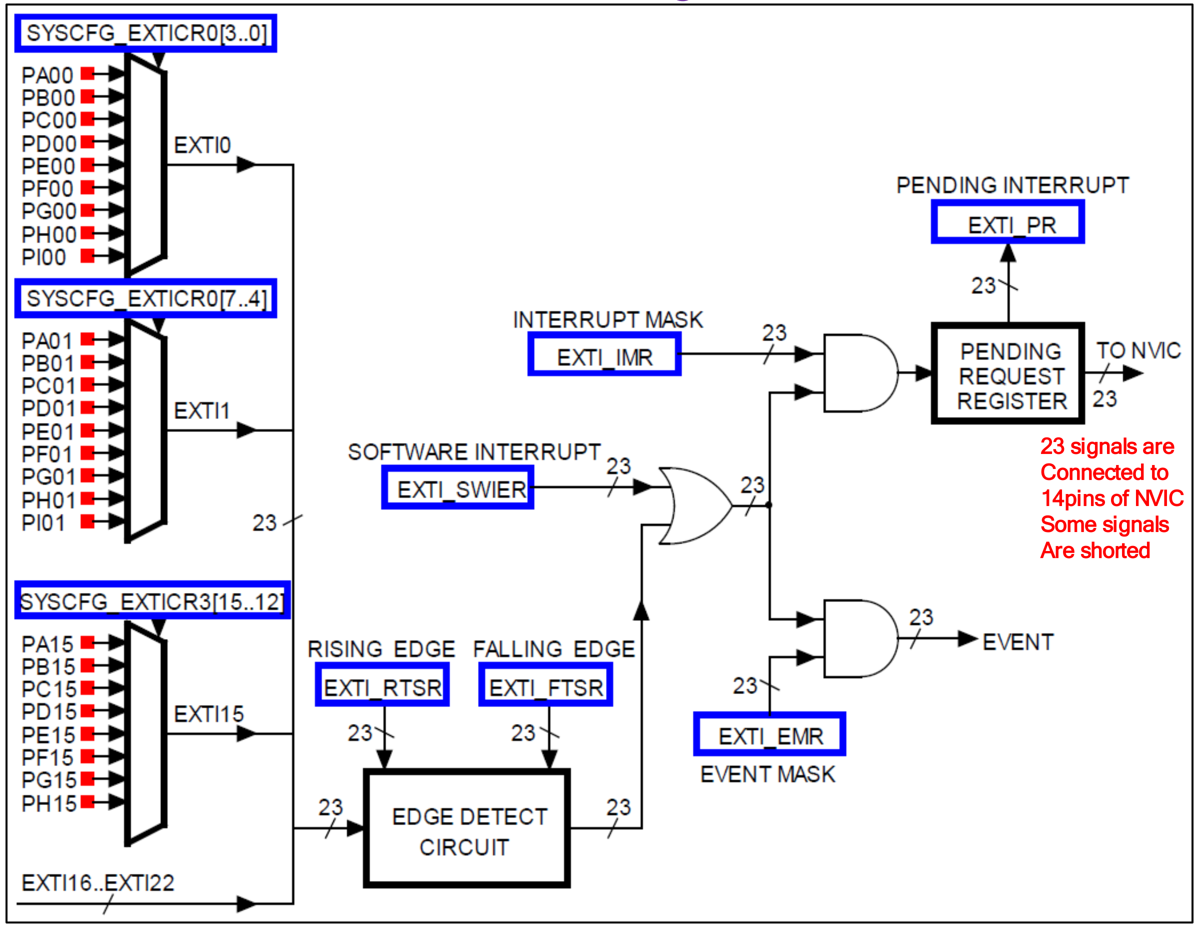
STM32F4xx - External Interrupt/Event Controller

- At a time we can enable only 23 external interrupts
- EXTI controller block takes care of,
  - Edge detection (Raising edge, Falling edge or both the edges)
  - Enabling or Disabling the delivery of interrupt to the NVIC/Interrupt controller



Block Diagram





Parandhaman

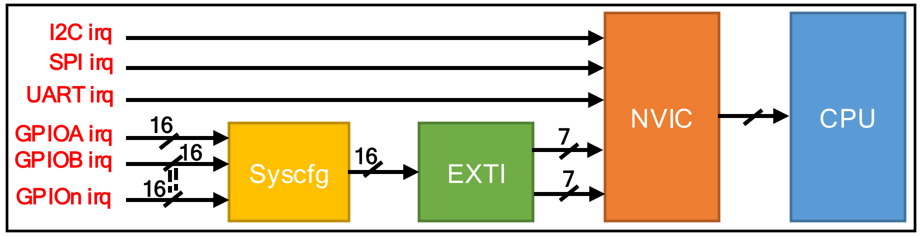
EXTI controller registers

- For all the below registers only 0 to 22 bits are used, why?
  - Because we have only 23 external interrupt lines

Register	offset	Reset value	Access	Description
Interrupt Mask Register (IMR)	0x00	0x0000_0000		
Event Mask Register (EMR)	0x04	0x0000_0000		
Rising Trigger Selection Register (RTSR)	0x08	0x0000_0000		
Falling Trigger Selection Register (FTSR)	0x0C	0x0000_0000		
Software Interrupt Event Register (SWIER)	0x10	0x0000_0000		
Pending Register	0x14	Undefined		

Steps to enable external interrupt

- GPIO Configuration
  - Enable GPIO clock
  - GPIO pin mode should be in input mode
  - GPIO pin should be configured either pull up or pull down
- System Configuration
  - Enable SysCfg clock
  - Short the GPIO pin to the EXTI line using SysConfig External Interrupt Configuration Register
- EXTI Configuration
  - When interrupt should be triggered, falling edge or raising edge
    - Falling edge → Falling Trigger Selection Register
    - Raising edge → Rising Trigger Selection Register
  - Enable the delivery of the EXTI line signal to the NVIC using Interrupt Mask Register
- NVIC Configuration
  - Enable the IRQ using NVIC register by setting particular IRQ number
  - Set the priority of the IRQ using NVIC register (if required)



UART

Steps to configure the UART

- Program the UART word length
- Program the number of stop bits
- Program the desired baud rate using UART_BRR register
- Set the UART Transmission Enable/Reception Enable bits to enable TX/RX
- Enable to the UART
- If TXE is set then data is sent
- If RXE is set then data is received

SPI

Backup