**System Clock**

**Code:**

void systemClockConfig(void)

{

//enable HSE nd wait for HSE to become ready

RCC->CR |= RCC\_CR\_HSEON;

while(!(RCC->CR & RCC\_CR\_HSERDY));

//enable power clock and voltage regulator

RCC->APB1ENR |= RCC\_APB1ENR\_PWREN;

PWR->CR |= PWR\_CR\_VOS;

//configure th flash prefetch and latency related setting

FLASH->ACR = FLASH\_ACR\_ICEN|FLASH\_ACR\_DCEN|FLASH\_ACR\_LATENCY\_5WS|FLASH\_ACR\_PRFTEN;

//configure the prescalers HCLK PCLK1 PCLK2

//AHB prescaler = /1

RCC->CFGR |= RCC\_CFGR\_HPRE\_DIV1;

//APB1 prescaler = /2

RCC->CFGR |= RCC\_CFGR\_PPRE1\_DIV2;

//APB2 prescaler = /1

RCC->CFGR |= RCC\_CFGR\_PPRE2\_DIV1;

//configure main PLL

RCC->PLLCFGR = (PLL\_M<<0)|(PLL\_N<<6)|(PLL\_P<<16)|(RCC\_PLLCFGR\_PLLSRC\_HSE);

//enable PLL and wait it to become ready

RCC->CR |= RCC\_CR\_PLLON;

while(!(RCC->CR & RCC\_CR\_PLLRDY));

//set clock source and wait it to be set

RCC->CFGR |= RCC\_CFGR\_SW\_PLL;

while((RCC->CFGR & RCC\_CFGR\_SWS) != RCC\_CFGR\_SWS\_PLL);

}

**Required Explanation:**

**VOS**

The VOS bits in the Power Control Register (PWR\_CR) control the output voltage of the main internal voltage regulator in an STM32 microcontroller. These settings allow a trade-off between performance and power consumption, depending on the operational requirements and maximum operating frequency of the device.

**Understanding VOS[1:0] Bits**

The VOS bits are located in the PWR\_CR register, which is part of the Power Control (PWR) peripheral. The VOS bits can be modified only when the Phase-Locked Loop (PLL) is off. When the PLL is turned off, the voltage regulator defaults to Scale 3 mode regardless of the value in the VOS bits. The new value programmed into the VOS bits takes effect only when the PLL is on.

Here’s the breakdown of the VOS bits:

* **00**: Reserved (Scale 3 mode selected)
* **01**: Scale 3 mode
* **10**: Scale 2 mode
* **11**: Reserved (Scale 2 mode selected)

**Regulator Voltage Scaling Modes**

1. **Scale 3 Mode (VOS[1:0] = 01 or 00)**:
   * This mode is intended for low power consumption.
   * The maximum operating frequency is reduced compared to Scale 2 mode.
   * Suitable for applications that do not require high performance but prioritize power efficiency.
2. **Scale 2 Mode (VOS[1:0] = 10 or 11)**:
   * This mode allows higher operating frequencies.
   * It provides better performance at the cost of increased power consumption.
   * Suitable for performance-critic

**Flash configuration**

**FLASH\_ACR\_ICEN (Instruction Cache Enable)**

 **Bit position**: 9

 **Description**: Enables the instruction cache.

 **Effect**: When set, the instruction cache is enabled. This can improve the execution speed of instructions by caching them in a small, fast memory close to the processor.

**FLASH\_ACR\_DCEN (Data Cache Enable)**

* **Bit position**: 10
* **Description**: Enables the data cache.
* **Effect**: When set, the data cache is enabled. This can improve data access speed by caching frequently accessed data in a small, fast memory close to the processor.

**FLASH\_ACR\_LATENCY\_5WS (Latency)**

* **Bit positions**: 0 to 2
* **Description**: Sets the number of wait states for flash memory access.
* **Effect**: Configures the flash latency, which is the number of wait states the processor must wait when accessing flash memory. The number of wait states needed depends on the system clock speed. For example, if the system clock speed is high, more wait states are required to ensure reliable flash memory access.
* **Value**: 5 wait states

**FLASH\_ACR\_PRFTEN (Prefetch Enable)**

* **Bit position**: 8
* **Description**: Enables the prefetch buffer.
* **Effect**: When set, the prefetch buffer is enabled. The prefetch buffer fetches instructions in advance before they are actually needed by the processor. This can help reduce wait times for instruction fetches, thereby improving execution speed.

Combining these settings optimizes the performance of the flash memory interface:

* **FLASH\_ACR\_ICEN**: Enables the instruction cache to speed up instruction fetches.
* **FLASH\_ACR\_DCEN**: Enables the data cache to speed up data access.
* **FLASH\_ACR\_LATENCY\_5WS**: Sets the flash memory latency to 5 wait states, necessary for higher system clock speeds.
* **FLASH\_ACR\_PRFTEN**: Enables the prefetch buffer to improve instruction fetch efficiency.

**RCC clock configuration**

In the context of configuring the STM32 microcontroller, the Advanced High-performance Bus (AHB) and the Advanced Peripheral Bus (APB1 and APB2) prescalers are used to divide the system clock to generate the respective bus clocks. These prescalers help in managing power consumption and ensuring that peripherals are clocked within their operating frequency ranges

**RCC\_CFGR\_HPRE\_DIV1 (AHB Prescaler)**

* **Field**: HPRE (bits 7:4 in RCC->CFGR)
* **Value**: 0b0000 (DIV1)
* **Description**: This sets the AHB clock (HCLK) to be the same as the system clock (SYSCLK).
* **Effect**: The AHB bus, core, memory, and DMA are clocked at the full speed of the system clock.

**RCC\_CFGR\_PPRE1\_DIV2 (APB1 Prescaler)**

* **Field**: PPRE1 (bits 12:10 in RCC->CFGR)
* **Value**: 0b100 (DIV2)
* **Description**: This sets the APB1 clock (PCLK1) to be half of the system clock (SYSCLK / 2).
* **Effect**: The peripherals connected to the APB1 bus, such as timers, UARTs, and I2C, are clocked at half the system clock speed. This is often necessary because many peripherals on the APB1 bus have a maximum operating frequency lower than the maximum system clock frequency.

**RCC\_CFGR\_PPRE2\_DIV1 (APB2 Prescaler)**

* **Field**: PPRE2 (bits 15:13 in RCC->CFGR)
* **Value**: 0b000 (DIV1)
* **Description**: This sets the APB2 clock (PCLK2) to be the same as the system clock (SYSCLK).
* **Effect**: The peripherals connected to the APB2 bus, such as ADCs, SPI1, and USART1, are clocked at the full speed of the system clock.

**Why These Settings?**

* **AHB Prescaler (HPRE)**: The AHB prescaler is set to 1 to ensure that the CPU, memory, and DMA operate at the full system clock speed for maximum performance.
* **APB1 Prescaler (PPRE1)**: The APB1 prescaler is set to 2 because many peripherals on the APB1 bus have a maximum clock frequency of 42 MHz, which is half of the maximum system clock frequency of 84 MHz on an STM32F4 running at full speed.
* **APB2 Prescaler (PPRE2)**: The APB2 prescaler is set to 1 because the peripherals on the APB2 bus can typically operate at the full system clock speed

**PLL**

1. **PLL\_M**:
   * **Field**: PLLM (bits 5:0 in RCC\_PLLCFGR)
   * **Value**: PLL\_M (defined as 4 in the previous code snippet)
   * **Position**: (PLL\_M << 0) places the value of PLL\_M in bits 5:0 of the RCC\_PLLCFGR register.
   * **Function**: This sets the division factor for the PLL input clock. The input clock to the PLL is divided by this value.
   * **Example**: If HSE is 8 MHz, the input to the PLL would be 8 MHz / 4 = 2 MHz.
2. **PLL\_N**:
   * **Field**: PLLN (bits 14:6 in RCC\_PLLCFGR)
   * **Value**: PLL\_N (defined as 84 in the previous code snippet)
   * **Position**: (PLL\_N << 6) places the value of PLL\_N in bits 14:6 of the RCC\_PLLCFGR register.
   * **Function**: This sets the multiplication factor for the VCO (Voltage Controlled Oscillator) output clock. The input clock to the PLL is multiplied by this value.
   * **Example**: If the input to the PLL is 2 MHz, the VCO output would be 2 MHz \* 84 = 168 MHz.
3. **PLL\_P**:
   * **Field**: PLLP (bits 17:16 in RCC\_PLLCFGR)
   * **Value**: PLL\_P (defined as 0 in the previous code snippet, but it corresponds to a division factor of 2 since PLL\_P = 2 when the value is 0).
   * **Position**: (PLL\_P << 16) places the value of PLL\_P in bits 17:16 of the RCC\_PLLCFGR register.
   * **Function**: This sets the division factor for the main system clock. The VCO output clock is divided by this value to get the final PLL output clock.
   * **Example**: If the VCO output is 168 MHz, the PLL output would be 168 MHz / 2 = 84 MHz.
4. **RCC\_PLLCFGR\_PLLSRC\_HSE**:
   * **Field**: PLLSRC (bit 22 in RCC\_PLLCFGR)
   * **Value**: 1 (RCC\_PLLCFGR\_PLLSRC\_HSE sets this bit to use HSE as the PLL source)
   * **Function**: This selects the HSE (High-Speed External clock) as the PLL input clock source.
   * **Example**: The HSE clock is typically an external crystal oscillator, commonly 8 MHz or 16 MHz.

**Putting it All Together:**

* The HSE clock (e.g., 8 MHz) is divided by PLL\_M (4), resulting in 2 MHz.
* The 2 MHz is then multiplied by PLL\_N (84), resulting in 168 MHz.
* The 168 MHz is then divided by PLL\_P (2), resulting in 84 MHz, which is used as the system clock (SYSCLK).

**Example:**

If the HSE clock is 8 MHz:

1. Input to PLL: 8 MHz / 4 = 2 MHz
2. VCO output: 2 MHz \* 84 = 168 MHz
3. System clock: 168 MHz / 2 = 84 MHz

This final system clock of 84 MHz is then used to drive the CPU and other peripherals, ensuring they operate at the desired frequency.

**System clock select**

**Bits 1:0 SW:** System clock switch Set and cleared by software to select the system clock source. Set by hardware to force the HSI selection when leaving the Stop or Standby mode or in case of failure of the HSE oscillator used directly or indirectly as the system clock.

* **RCC->CFGR**: This is the Clock Configuration Register of the RCC (Reset and Clock Control) peripheral.
* **RCC\_CFGR\_SW\_PLL**: This is a macro that sets the system clock switch (SW) bits in the CFGR register to select the PLL as the system clock source.
  + The SW bits (bits 1:0) in the RCC\_CFGR register are used to select the system clock source.
  + Setting SW to 0b10 (RCC\_CFGR\_SW\_PLL) selects the PLL as the system clock source.

**System Clock Switch (SW) Bits in RCC\_CFGR:**

* 00: HSI oscillator selected as system clock
* 01: HSE oscillator selected as system clock
* 10: PLL selected as system clock
* 11: Not allowed (reserved)
* **RCC->CFGR**: The same Clock Configuration Register as above.
* **RCC\_CFGR\_SWS**: This macro represents the System Clock Switch Status (SWS) bits in the CFGR register.
  + The SWS bits (bits 3:2) in the RCC\_CFGR register reflect the current system clock source.
  + When SWS is 0b10 (RCC\_CFGR\_SWS\_PLL), it indicates that the PLL is currently being used as the system clock source.

The while loop continually checks the SWS bits to see if they equal 0b10 (PLL as the system clock source). The loop exits only when the SWS bits indicate that the PLL has successfully been set as the system clock source

Bits 3:2 SWS: System clock switch status Set and cleared by hardware to indicate which clock source is used as the system clock.

* 00: HSI oscillator used as the system clock
* 01: HSE oscillator used as the system clock 10: PLL used as the system clock
* 11: not applicable