Name: Khoi Nguyen

Class: ECE 5780

Post lab 1

1. What are the GPIO control registers that the lab mentions? Briefly describe each of their functions.
   1. GPIO port mode register (GPIOx\_MODER): defines each GPIO pin’s operating mode. Each pin can be configured as an input, output, analog, or alternate function.
   2. GPIO port output type register (GPIOx\_OTYPER): determines the output type for each GPIO pin configured as an output.
   3. GPIO port output speed register (GPIOx\_OSPEEDR): sets the speed for each GPIO pin configured as an output.
   4. GPIO port pull-up/pull-down register (GPIOx\_PUPDR): is used to activate the internal pull-up or pull-down resistors on GPIO pins.
   5. GPIO port input data register (GPIOx\_IDR): provides the current input value of the GPIO pins.
   6. GPIO port output data register (GPIOx\_ODR): holds the output value of the GPIO pins.
   7. GPIO port bit set/reset register (GPIOx\_BSRR): used to set/clear the output value of the GPIO pins atomically, without affecting the state of other GPIO pins.
   8. GPIO port configuration lock register (GPIOx\_LCKR): is used to lock the configuration of the GPIO pins.
   9. GPIO alternate function low/high registers (GPIOx\_AFRL/GPIOx\_AFRH): used to select the alternate function for GPIO pins when they are configured in alternate function mode.
   10. GPIO port bit reset register (GPIOx\_BRR): similar to the bit set/reset register (GPIOx\_BSRR), but it only handles the reset operation.
2. What values would you want to write to the bits controlling a pin in the GPIOx\_MODER register in order to set it to analog mode?
   1. To set a pin to analog mode in the GPIOx\_MODER register, you would want to write the value 11 to the bits controlling that pin.
   2. The bit pattern 11 corresponds to the analog mode
3. Examine the bit descriptions in GPIOx\_BSRR register: which bit would you want to set to clear the fourth bit in the ODR?
   1. To clear the fourth bit in the ODR using the GPIOx\_BSRR register, you would want to set the 20th bit.
4. Perform the following bitwise operations:
   1. 0xAD | 0xC7 = 1010 1101 | 1100 0111 = 1110 1111 = 0xEF
   2. 0xAD & 0xC7 = 1010 1101 & 1100 0111 = 1000 0101 = 0x85
   3. 0xAD & ~(0xC7) = 1010 1101 & 0011 1000 = 0010 1000 = 0x28
   4. 0xAD ^0xC7 = 1010 1101 ^ 1100 0111 = 0110 1010 = 0x6A
5. How would you clear the 5th and 6th bits in a register while leaving the other’s alone?
   1. To clear the 5th and 6th bits in a register while leaving the other bits unchanged, you would perform a bitwise AND operation with a bitmask that has all bits set to 1 except for the 5th and 6th bits, which should be set to 0.
   2. The bitmask for this operation would be 0xFFFFFFCF in hexadecimal (11001111 = 0xCF)
   3. Structure code: *register\_value &= 0xFFFFFFCF;*
6. What is the maximum speed the STM32F072R8 GPIO pins can handle in the lowest speed setting?
   1. The slowest speed setting is x0 and based on the I/O AC characteristics table:

A white sheet with black text and black letters

Description automatically generated

* 1. The maximum frequency (speed) is 2MHz for the condition that VDDIOx ≥ 2V or 1 MHz for the condition that VDDIOx < 2V

1. What RCC register would you manipulate to enable the following peripherals: (use the comments next to the bit defines for better peripheral descriptions)
   1. TIM1 (TIMER1): To enable TIM1, you would need to set ‘1’ for the TIM1EN bit in the register RCC\_APB2ENR (APB peripheral clock enable register 2).
   2. DMA1: To enable DMA1, you would need to set ‘1’ for the DMAEN bit in the register RCC\_ AHBENR (AHB peripheral clock enable register).
   3. I2C1: To enable I2C1, you would need to set ‘1’ for the I2C1EN bit in the register RCC\_APB1ENR (APB peripheral clock enable register 1).

Github link: https://github.com/kenzend94/ECE\_5780/tree/main/Lab1