Brian Burton

ECE 5780

2/9/2022

**Post Lab 01 (Intro/GPIO)**

1. What are the GPIO control registers that the lab mentions? Briefly describe each of their

functions.

***MODER*** – The GPIOx\_MODER register is used to select the I/O mode (input, output, AF, analog). This is a control register.

***OTYPER*** – The GPIOx\_OTYPER and GPIOx\_OSPEEDR registers are used to select the output type (push-pull or open-drain) and speed. This is a control register.

**OSPEEDR** – The GPIOx\_OTYPER and GPIOx\_OSPEEDR registers are used to select the output type (push-pull or open-drain) and speed. This is a control register.

***PUPDR*** – The GPIOx\_PUPDR register is used to select the pull- up/pull-down whatever the I/O direction. This is a control register.

***ODR*** – Each GPIO has two 16-bit memory-mapped data registers: input and output data registers (GPIOx\_IDR and GPIOx\_ODR). GPIOx\_ODR stores the data to be output, it is read/write accessible. The data input through the I/O are stored into the input data register (GPIOx\_IDR), a read-only register.

***IDR***— Each GPIO has two 16-bit memory-mapped data registers: input and output data registers (GPIOx\_IDR and GPIOx\_ODR). GPIOx\_ODR stores the data to be output, it is read/write accessible. The data input through the I/O are stored into the input data register (GPIOx\_IDR), a read-only register.

***BSRR*** – The bit set reset register (GPIOx\_BSRR) is a 32-bit register which allows the application to set and reset each individual bit in the output data register (GPIOx\_ODR). The bit set reset

register has twice the size of GPIOx\_ODR. This is a bitwise handling register.

2. What values would you want to write to the bits controlling a pin in the GPIOx\_MODER

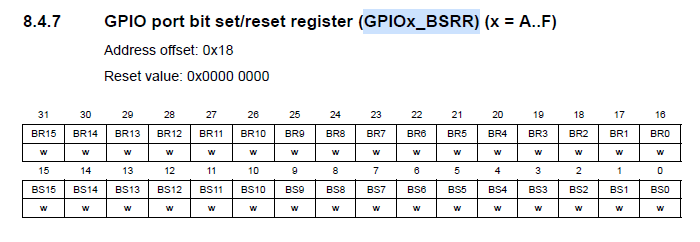
register in order to set it to analog mode?

You would want to write the values 11 to the bits controlling a pin in the GPIOx\_MODER register to change it to 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?

You would want to set the BS3 bit in order to clear the fourth bit in the ODR.



4. Perform the following bitwise operations:

0xAD in binary is 10101101

0xC7 in binary is 11000111

• 0xAD | 0xC7 = ? 10101101

| 11000111

----------------

11101111

0xAD | 0xC7 = 0xEF

• 0xAD & 0xC7 = ? 10101101

& 11000111

----------------

10000101

0xAD & 0xC7 = 0x85

• 0xAD & ~(0xC7) = ?

~(0xC7) in binary is 00111000

10101101

& 00111000

----------------

00101000

0xAD & ~(0xC7) = 0x28

• 0xAD ^0xC7 = ? 10101101

^ 11000111

----------------

01101010

0xAD & ^0xC7 = 0x6A

5. How would you clear the 5th and 6th bits in a register while leaving the others alone?

If we want to clear the 5th and 6th bits in a register and leave the others alone, all we need to do is AND the 5th and 6th bits with 0 and the rest of the bits with 1.

For example:

01100011 in hex is 0x63

01100011

& 10011111

----------------

00000011

00000011= 0x3

6. What is the maximum speed the STM32F072R8 GPIO pins can handle in the lowest speed

setting?

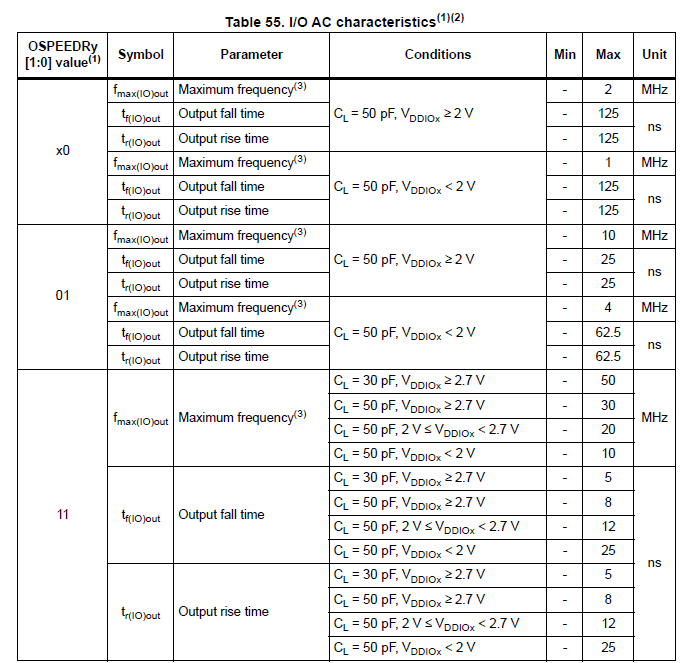
The maximum speed that the STM32F072R8 GPIO pins can handle in the lowest speed setting is 10MHz.

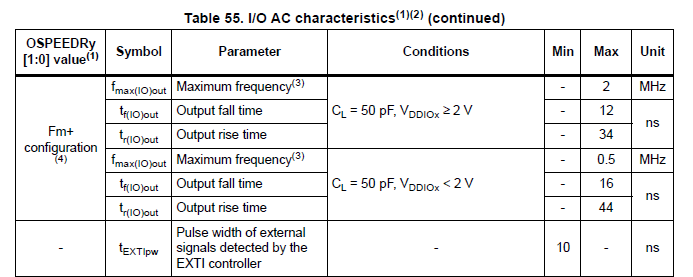
The lowest bit setting for OSPEEDER is either 00 or 10.

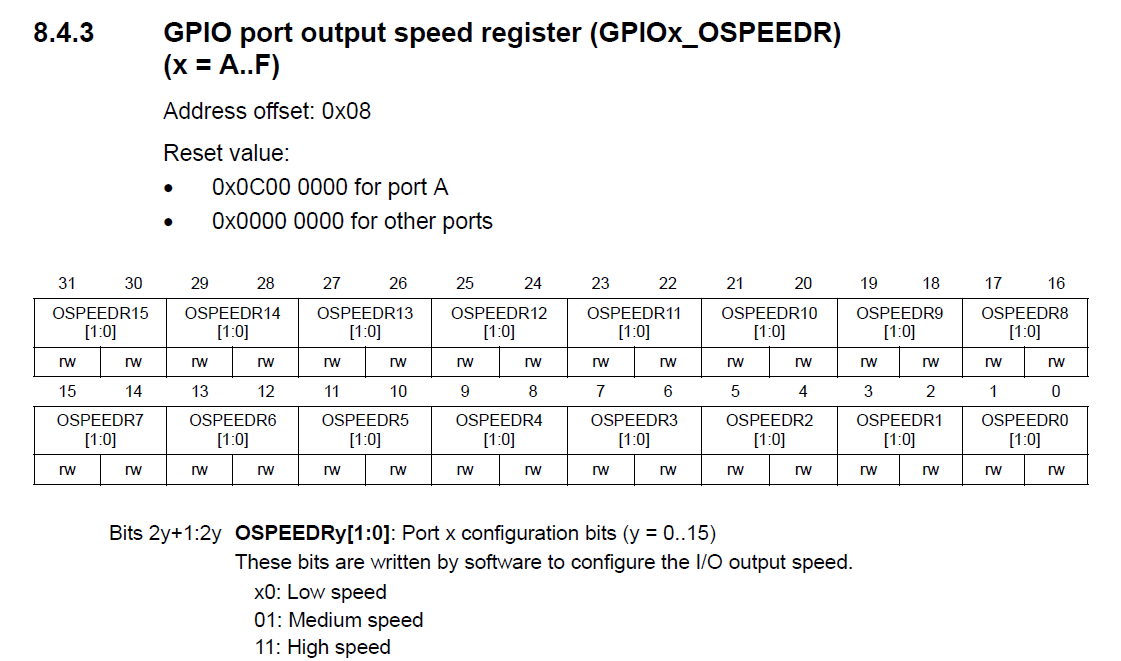
• Use the chip datasheet: lab section 1.4.1 gives a hint to the location. You’ll want to

search the I/O AC characteristics table. You will also need to view the OSPEEDR

settings to find the bit pattern indicating the slowest speed.







7. What RCC register would you manipulate to enable the following peripherals: (use the

comments next to the bit defines for better peripheral descriptions)

• TIM1 (TIMER1)

You would manipulate the RRC\_APB2ENR register to enable the TIM1 peripheral.

• DMA1

You would manipulate the RRC\_AHBENR register to enable the DMA1 peripheral.

• I2C1

You would manipulate the RRC\_APB1ENR register to enable the I2C1 peripheral.

