Department of Electrical and Computer Engineering
Concordia University

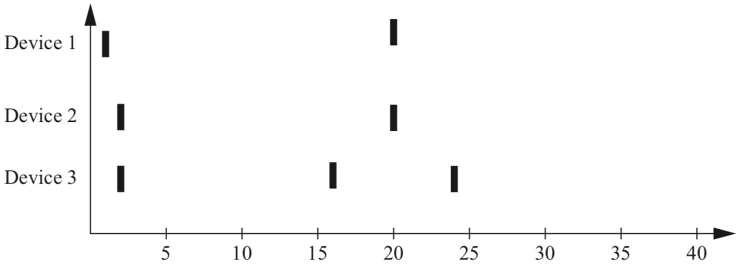
COEN 421/6341 – Embedded Systems Design
Winter 2023

Assignment #2
Submission: Moodle

1. **[10pts]** Compare two methods of responding to external events: polling and interrupts. Discuss the advantages and disadvantages of each approach.

2. **[5pts]** Discuss the process for data acquisition from an input device when polling technique is used.

3. **[5pts]** Three devices are attached to a microprocessor: Device 1 has the highest priority and device 3 has the lowest priority. Each device's interrupt handler takes 5-time units to execute. Show what interrupt handler (if any) is executing at each time given the sequence of device interrupts displayed below:



4. **[20pts]** The reference manual of the STM32L4 provides complete information on how to use the the microcontroller memory and peripherals. You can find the reference manual at: https://www.st.com/resource/en/reference_manual/rm0351-stm32l47xxx-stm32l48xxx-stm32l49xxx-and-stm32l4axxx-advanced-armbased-32bit-mcus-stmicroelectronics.pdf

Figure 1 shows the memory map for STM32L47x/L48x devices. The GPIO ports are connected to the *advanced high-performance bus 2 (AHB2)* and the boundary addresses for peripherals registers are showed in Fig.2, based on the GPIO port to be used.

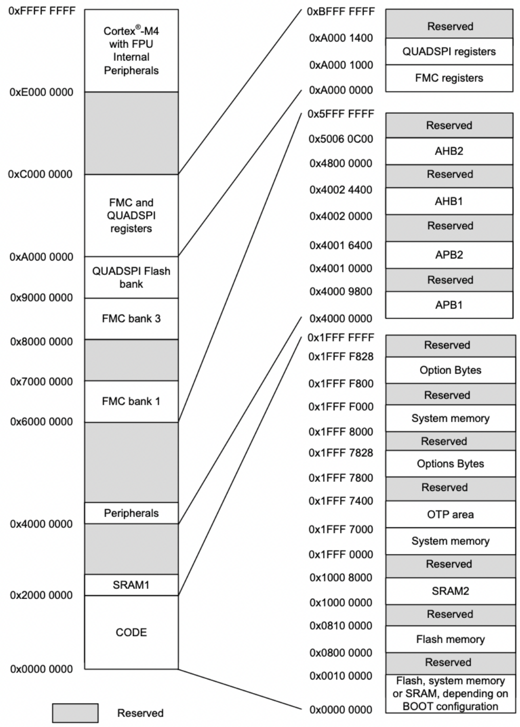
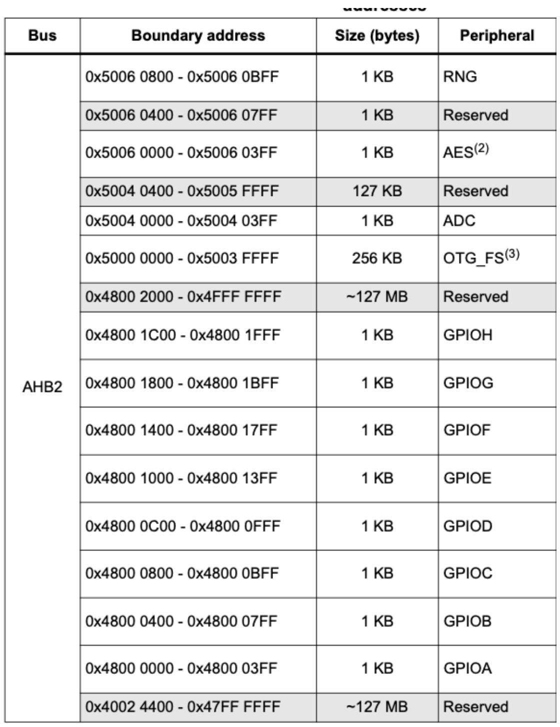


Fig. 1. Memory map

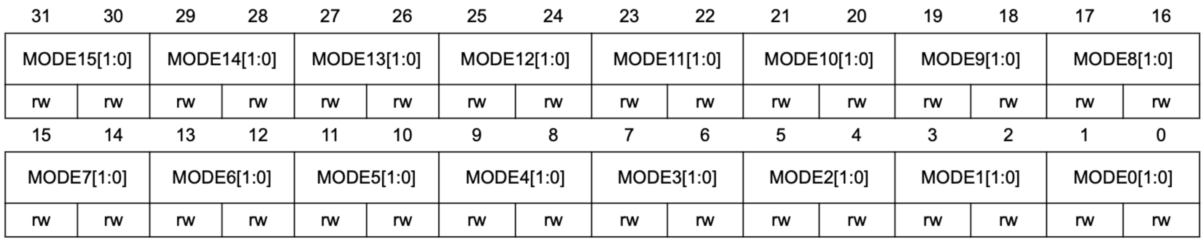


Bus	Boundary address	Size (bytes)	Peripheral
AHB2	0x5006 0800 - 0x5006 0BFF	1 KB	RNG
	0x5006 0400 - 0x5006 07FF	1 KB	Reserved
	0x5006 0000 - 0x5006 03FF	1 KB	AES(2)
	0x5004 0400 - 0x5005 FFFF	127 KB	Reserved
	0x5004 0000 - 0x5004 03FF	1 KB	ADC
	0x5000 0000 - 0x5003 FFFF	256 KB	OTG_FS(3)
	0x4800 2000 - 0x4FFF FFFF	~127 MB	Reserved
	0x4800 1C00 - 0x4800 1FFF	1 KB	GPIOH
	0x4800 1800 - 0x4800 1BFF	1 KB	GPIOG
	0x4800 1400 - 0x4800 17FF	1 KB	GPIOF
	0x4800 1000 - 0x4800 13FF	1 KB	GPIOE
	0x4800 0C00 - 0x4800 0FFF	1 KB	GPIOD
	0x4800 0800 - 0x4800 0BFF	1 KB	GPIOC
	0x4800 0400 - 0x4800 07FF	1 KB	GPIOB
	0x4800 0000 - 0x4800 03FF	1 KB	GPIOA
	0x4002 4400 - 0x47FF FFFF	~127 MB	Reserved

Fig. 2. Boundary address for peripherals

In the mentioned microcontroller, each GPIO port consists of a set of 16 pins and each pin has various registers associated with it. The behavior of a particular pin is controlled by changing the content of the associated registers. For instance, if your system has a sensor and you want to read data from that sensor, you need to set the mode of the pin, to which your sensor is connected, as input mode. Similarly, you need to set the mode of the pin that your output device (e.g., LED) is connected to, as general-purpose output mode.

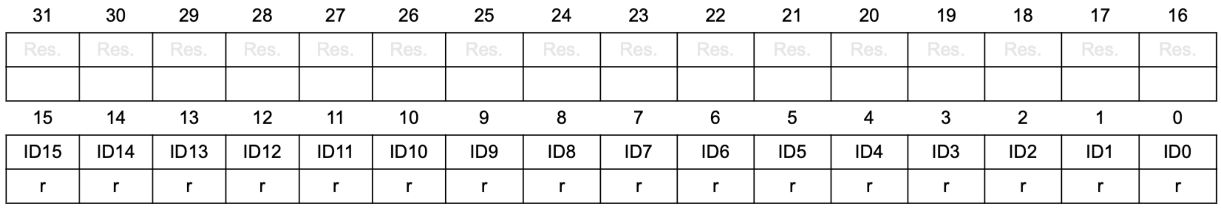
Thus, the programmer sets the mode of the pins (00: input mode or 01: General purpose output mode) on the GPIO port mode register, i.e., GPIOx_MODER (x=A to I), of the GPIO port to be used. Figure 3 shows the mode register used to configure the pins to be used by a given GPIO port. The behavior of each pin (0: MODE0 to 15: MODE15) is defined by the proper setting of the corresponding 2 bits of the pin (e.g., bits 0 and 1 in the register define the behavior of pin 0).



31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODE15[1:0]		MODE14[1:0]		MODE13[1:0]		MODE12[1:0]		MODE11[1:0]		MODE10[1:0]		MODE9[1:0]		MODE8[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODE7[1:0]		MODE6[1:0]		MODE5[1:0]		MODE4[1:0]		MODE3[1:0]		MODE2[1:0]		MODE1[1:0]		MODE0[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Fig. 3. GPIO port mode register (GPIOx_MODER)

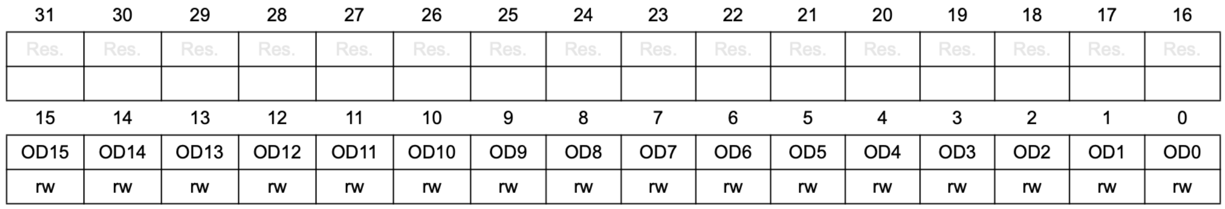
Then, the state of the pin for an input device is determined by reading the corresponding bit on the GPIOx_IDR register. For instance, bit 15 on the IDR register would need to be read to check if a button connected to pin 15 is pressed (i.e., bit 15=1) or not (i.e., bit 15 = 0). Figure 4 shows the GPIOx_IDR input data register for a GPIO port. Since there are only 16 pins, bits from 16 to 31 are not used, that is, they are reserved.



31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Fig. 4. GPIO port input data register (GPIOx_IDR)

Similarly, an output is produced on a GPIO pin by setting 1 to its corresponding bit on the output data register. Figure 5 shows the output data register for a GPIO port. Since there are only 16 pins, bits from 16 to 31 are not used, that is, they are reserved.



31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OD15	OD14	OD13	OD12	OD11	OD10	OD9	OD8	OD7	OD6	OD5	OD4	OD3	OD2	OD1	OD0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Fig. 5. GPIO port output data register (GPIOx_ODR)

The offset for the GPIOx_MODER, GPIOx_IDR, and GPIOx_ODR register is 0x00, 0x10, and 0x14 respectively.

Assume a switch connected to pin 4 and a LED connected to pin 3 on the GPIO port B. When the switch is pressed, the LED light is turned on. Considering the data presented in the description and figures, discuss what will be the memory addresses used to configure the read and write pins, as well as the memory address used to read the input provided by the switch and write the output for the LED.

5. **[10pts]** Assume an input device is connected to the SPI3 port on an STM32L4 microcontroller. Discuss where the handler for dealing with interrupts generated from that device is located in memory.

6. **[10pts]** Discuss the motivation and the process for stacking and unstacking when the CPU is dealing with an interrupt.

7. **[5pts]** What is the average memory access time of a machine whose hit rate is 96%, with a cache access time of 3ns and a main memory access time of 70ns?

8. **[10pts]** Explain the process for address translation when the segmented memory organization is used.

9. Consider that the below code is executed by an ARM processor with each instruction executed exactly once. For the sake of simplicity, the following assumptions are made:
   1. The main memory has size of 4Gb.
   2. The size of the cache memory is 256 bytes.
   3. The size of each instruction is 16 bits.
   4. Each memory read operation returns 4 bytes.
   5. Assume the base address for the following code segment is 0x08000160.

```
            MOV r0,#0                   ; use r0 for i, set to 0
            LDR r1,#10                  ; get value of N for loop termination test
            MOV r2,#0                   ; use r2 for f, set to 0
            ADR r3,c                    ; load r3 with address of base of c array
            ADR r5,x                    ; load r5 with address of base of x array
loop        CMP r0,r1
            BGE loopend                 ; if i >= N, exit loop
            LDR r4,[r3,r0]              ; get value of c[i]
            LDR r6,[r5,r0]              ; get value of x[i]
            MUL r4,r4,r6                ; compute c[i]*x[i]
            ADD r2,r2,r4                ; add into running sum f
            ADD r0,r0,#1                ; add 1 to i
            B loop                      ; unconditional branch to top of loop
```

Show the content of the cache memory upon the execution of each instruction presented above, when:

a) **[12.5pts]** A direct-mapped cache memory is used.

b) **[12.5pts]** A 2-way set-associative cache memory is used.