Politecnico di Milano
SCUOLA DI INGEGNERIA INDUSTRIALE E DELL'INFORMAZIONE

Advanced Operating Systems
A.A. 2017-2018 – Exam date: December, 20th 2017

Prof. William FORNACIARI

Surname (readable).....	Name (readable).....
Matr.....	Signature.....

Q1	Q2	TOT

NOTES

It is forbidden to refer to texts or notes of any kind as well as interact with their neighbors. Anyone found in possession of documents relating to the course, although not directly relevant to the subject of the examination will cancel the test. It is not allowed to leave during the first half hour, the task must still be returned, even if it is withdrawn. The presence of the writing (not delivered) implies the renunciation of any previous ratings.

Question Q1 (10 points)

Describe the problem of the CPU scheduling in a uniprocessor systems and the main scheduling algorithms.
Which are the main aspects to be considered having in mind the goal to provide real-time features?

Question Q2 (13 points)

In a Santa Claus hat five LEDs are connected to the following pins of a microcontroller: PA0, PA1, PA2, PA3, PA4. A button is connected to PA5. All connections are active high, so the LED turn on when the GPIO is high, and the button reads as 1 when pressed.
The microcontroller clock is 1MHz.

This is a fragment of the microcontroller datasheet describing the GPIO and TIMER peripherals

Bit access modes:

- **u** = undefined, writing has no effect, reading returns zero
- **rw** = software reading and writing is allowed

GPIOA_DIR: Address: 0x00120000, Initial value upon reset: 0xffff

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Port A has 16 GPIOs, named from PA0 to PA15. Setting to 1 the corresponding GPIOA_DIR bit will configure the pin as input, while clearing it to 0 will configure the pin as output.

GPIOA_IO: Address: 0x00120004, Initial value upon reset: undefined

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Reading a bit in this register when the corresponding GPIO pin is configured as input will return the logic level applied to the pin, writing to a bit in this register when the corresponding GPIO pin is configured as output will drive the pin high or low.

TIMER0_CTRL: Address: 0x00130000, Initial value upon reset: 0x0000

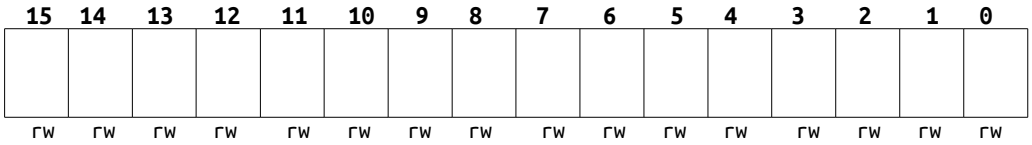
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							UD	PS5	PS4	PS3	PS2	PS1	PS0	IRQ EN	EN
u	u	u	u	u	u	u	rw	rw	rw	rw	rw	rw	rw	rw	rw

- EN: setting this bit to 1 starts the timer
- IRQEN: setting this bit to 1 causes an interrupt function named void TIMER0_irq(); to be called when
  - the timer is counting up and its value overflows, that is, it transitions from 0xffff to 0x0000
  - The timer is counting down and it underflows, that is, it transitions from 0x0000 to 0xffff
- PS0..PS5: These bits form a 6 bit number N. The microcontroller clock frequency is divided by N+1 to generate the frequency at which the

timer counter is incremented. For example, if the microcontroller clock is 8MHz and N is 15, the timer is incremented every 2 microseconds.

- UD: if this bit is 0, the timer counts up, otherwise it counts down

TIMER0_CTR: Address 0x00130002, Initial value upon reset: 0x0000



Timer counter. It is incremented/decremented at a frequency determined by the microcontroller frequency divided by the prescaler. Can be written at any time even when the timer is running.

You are required to:

1. Write **data structures** and **macros** which could be used to access these peripheral registers, by a C/C++ coded program, using a syntax similar to GPIOA->DIR;
2. Write a main program and an interrupt routine for the timer. The main program should initialize the peripherals and then start polling the button. The interrupt routine should blink the LEDs in two possible ways: light each LED for 100ms in sequence, with only one LED on at a time, or light them in sequence for 100ms at a time with all LEDs on but one. The button should alternate between the two blinking patterns.

Soluzione

```
struct GPIO_Struct {
	volatile unsigned short DIR;
	unsigned short padding;
	volatile unsigned short IO;
};

struct TIMER_Struct {
	volatile unsigned short CTRL;
	volatile unsigned short CTR;
};

#define GPIOA ((GPIO_Struct*)0x120000)
#define TIMER0 ((TIMER_Struct*)0x130000)

volatile bool invert=false;
unsigned short status=0b10000;

void TIMER0_irq() {
	TIMER0->CTR=50000;
	if(status==0b10000) status=0b00001;
	else status<<=1;
	if(invert==false) GPIOA->IO=status;
	else GPIOA->IO=~status;
}

int main() {
	GPIOA->DIR=~0b11111;
	TIMER0->CTRL=(1<<8) | (1<<2) | (1<<1) | (1<<0);
	for(;;) {
		//Wait till button pressed
		while((GPIOA→IO & (1<<5))==0) ;
		invert=!invert;
		//Wait till button released
		while((GPIOA→IO & (1<<5))==1) ;
	}
}
```