U.PORTO FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

Integrated Master in
Informatics and Computer / Electrical and Computer Engineering
EMBEDDED SYSTEMS / EMBEDDED AND REAL_TIME SYSTEMS
Time limit: 1h30 **TEST 2021-06-25**

1 An industrial robot arm has 4 axes and it is used to move parts from an input buffer to a machine tool. The robot arm is controlled by a single processor that runs a set of sporadic tasks on a real-time operating system (RTOS). Task τ_1 controls all the 4 axes. Task τ_2 dynamically adjusts the setpoints of the 4 axes to produce a desired arm trajectory. Task τ_3 is triggered by an emergency push button and it just sets a global flag that will be used by task τ_2 to produce setpoints for a controlled shutdown of the robot arm. Finally, task τ_4 reads from a camera at the tip of the robot arm, for visual feedback. The tasks timing properties (execution time, deadline and minimum inter-arrival time) are the following (unit *ms*):

$\tau_1 \rightarrow C_1=4, \quad D_1=T_1=10$ (feedback loop of all axes)

$\tau_2 \rightarrow C_2=20, \quad D_2=T_2=100$ (arm trajectory control)

$\tau_3 \rightarrow C_3=1, \quad D_3=5, \quad T_3=10000$ (emergency push button)

$\tau_4 \rightarrow C_4=12, \quad D_4=T_4=50$ (camera for visual feedback)

a **(2 points)** Consider the RTOS uses **Earliest Deadline First** scheduling **(EDF)**. How are priorities assigned with this method? What is the total utilization of the task set? Using this value, can you determine the schedulability of the task set? Justify.

b **(2 points)** Would it be adequate to use the **CPU-demand** method to determine **the schedulability** of the task set in this case? Justify and apply it if adequate.

c **(2 points)** Draw a **Gantt chart** of the tasks execution for the 100ms that follow the critical instant. Would it be different if using **Deadline-Monotonic (DM)** scheduling, instead? Justify.

d **(1.5 points)** Using the Gantt chart what is the worst-case response time of the tasks? If you did not plot the Gantt chart, just explain how you would determine the worst-case response times.

e **(1.5 points)** Task τ_4 processes the camera stream and exhibits a very **irregular execution** time. The worst-case response-time could occasionally go over the 12ms declared in the specification above. However, on average the execution time is lower than 12 ms. How could you protect the timeliness of the remaining system against these overruns of task τ_4? Justify.

f **(1 point)** The **global flag** used by tasks τ_3 and τ_2 to communicate that the **emergency button** was pressed is implemented in **1 byte**. Do you need to protect it with some synchronization method lke interrupts disabling? Explain why.

2 The code in file schedule.c (see next page) implements a small kernel using a **pre-emptive priority-based scheduler** (identical to the code used in the laboratory sessions).

This program will run on a micro-controller. You are using a standard PC to run the engineering environment (i.e. source code editor, compiler, etc.).

Consider that the program requires an external library made available by a third party vendor (e.g., that implements a specific SPI protocol) to which you do not have the source code.

This scheduler is used to run an application consisting of 5 tasks.

Task1 and **Task3** both use the shared resource **var1**.

Task2 and **Task3** both use the shared resource **var2**.

a **(1 point)** The file main.c references the files schedule.h and tasks.h through '#include' directives (lines 69, 70).

- Explain what each of these .h files should contain;

b **(1,5 points)**

- Explain the sequence of commands/steps that you need to execute/take in order to run the program on the micro-controller. Consider that you are starting with the given source code (.c and .h files, residing on the PC) that do not contain errors.

- Mention the tools/programs needed to accomplish this task.

c **(2 points)**

- Do you ask the vendor for a static or dynamic version of the third party library? Explain why.

- A few months after having bought the third party library you decide that you need to change to a more powerful micro-controller. You decide on a different micro-controller architecture (e.g., you change from ATMEGA, to ARM). Can you keep using the same library? Explain why or in what circumstances.

d **(2 points)** In order to guarantee atomic access to the shared resources you decide to disable the interrupts before accessing each resource, and to re-enable them immediately after.

- Explain why, when using the current scheduler, it is necessary to use another mechanism to guarantee atomic access to the resources.

- Explain why it isn't possible to add to this scheduler a mutex that supports the priority inheritance protocol.

e **(1,5 points)** The compiler tells you the maximum number of bytes each function places on the stack:

- Task1 (32 bytes); Task2 (32 bytes); Task3 (120 bytes);

- Task4 (240 bytes); Task5 (530 bytes);

- Main (10 bytes); work (56 bytes) (see line 84);

The compiler also allows you to configure the number of bytes to reserve for the global stack. Determine the minimum size of the global stack that allows for the safe execution of your program considering each of the following options:

i) you keep using the **pre-emptable** version of the scheduler;

ii) you opt for the **non pre-emptable** version of the scheduler;

Explain your reasoning for each of the scenarios.

3 **(2 points)** Answer **one, and only one** of the following questions:

a In the context of dependable systems, explain the concepts of 'error', 'fault', and 'failure'. Mention how they are related, and give an example.

b MISRA-C guidelines may be classified by type (rule, directive), by analysis (decidable, undecidable), or by category (mandatory, required, advisory, …). Explain meaning behind each of these classifications.

file: schedule.c

```c
typedef struct {
  int period; // period in ticks
  int delay;  // ticks to activate
  void *(*func)(void *);// function ptr
  int exec;   // activation counter
} Sched_Task_t;
Sched_Task_t Tasks[20];

void Sched_Schedule(void){
  for(int x=0; x<20; x++) {
    if !Tasks[x].func    continue;
    if  Tasks[x].delay
      {Tasks[x].delay--;}
    else {
      Tasks[x].exec  = 1;
      Tasks[x].delay = Tasks[x].period;
    }
}}

int  cur_task = 20;
void Sched_Dispatch(void){
 int prev_task = cur_task;
 for(int x=0; x<cur_task; x++){
  if Tasks[x].exec {
    Tasks[x].exec = 0;
    cur_task = x;
    enable_interrupts();
    Tasks[x].func(Tasks[x].parm);
    disable_interrupts();
    cur_task = prev_task;
    /*Delete if one-shot */
    if !Tasks[x].period
      Tasks[x].func = 0;
}}}

/* Int handler -> called every 10 ms */
void int_handler(void){
  disable_interrupts();
  Sched_Schedule();
  Sched_Dispatch();
  enable_interrupts();
}

int Sched_AddTask(
    void *(*f)(void *),
    int d, int p, int pri){
  if (Tasks[pri].func) return -1;
  Tasks[pri].period= p;
  Tasks[pri].delay = d;
  Tasks[pri].exec  = 0;
  Tasks[pri].func  = f;
  return pri;
}

void Sched_Init(void) {  …  }
```

file: tasks.c

```c
double var1 = 0;
double var2 = 0;

void task1()
  {// task1 algorithm …}
void task2()
  {// task2 algorithm …}
void task3()
  {// task3 algorithm …}
void task4()
  {// task4 algorithm …}
void task5()
  {// task5 algorithm …}
```

file: main.c

```c
#include “schedule.h”
#include “tasks.h”

int main(int argc, char **argv) {
Hardware_Init();
Sched_Init();

// launch tasks
Sched_AddTask(task1, 1, 5, 1);
Sched_AddTask(task2, 3, 5, 2);
Sched_AddTask(task3, 6, 20, 3);
Sched_AddTask(task4, 1,100, 4);
Sched_AddTask(task5, 1,650, 5);

// do some work in background...
while(1) work();
return 0; // humour the compiler
}
```

FORMULAS

Menor majorante	$U(n) = \Sigma^{n}_{i=1}(C_i/T_i) \leq n(2^{1/n}-1)$
Majorante hiperbólico	$\Pi^{n}_{i=1}(C_i/T_i+1) \leq 2$
Response Time	$Rwc_i\,(0) = C_i + Bi + \Sigma_{k\,\in hp(i)}\, C_k$ $Rwc_i\,(m+1) = C_i + Bi + \Sigma_{k\,\in hp(i)} \lceil Rwc_i(m)/T_k \rceil * C_k$
Synchronous busy period	$L(0) = \Sigma_i\, C$ $\qquad$ $L(m+1) = \Sigma_i \lceil L(m)/T_i \rceil * C_i$
Load function	$h(t) = \Sigma_{Di \leq t}\, (1+ \lfloor (t - D_i)/T_i \rfloor) * C_i$