Exam of 27/1/2021

The ACME-Vaccine decides to develop an Unmanned Ground Vehicle which is aimed at autonomously patrolling urban areas to find people that need to be vaccinated against COVID-19. The UGV, AstraZenecar II, is able to move autonomously in outdoor environments. In order to have an updated estimate of its own position, it uses a system based on differential GPS, which is able to provide localization data with an error less than 40cm.

In order to control the movement of the UGV – based on two drive wheels in a differential drive configuration – it is decided to use a dedicated I/O board ServoDG which is mounted on the onboard PC.

The ServoDG board uses:

- Two 16-bit digital outputs to control the actuators of the two drive wheels (i.e., each wheel is controlled by writing on two contiguous 8-bit registers, for a total of four contiguous 8-bit registers starting from base address 0x370)
- Four 16-bit digital inputs allowing to read the encoders of the two wheels, as well as pitch and roll data. Encoders are used to count wheel rotations, therefore allowing to compute the displacement of the UGV (each encoder requires to read from two contiguous 8-bit registers, for a total of four contiguous 8-bit registers starting from base address 0x270). Pitch and roll data are used to compute the attitude of the vehicle depending on the slope of the terrain (both pitch and roll require to read from two contiguous 8-bit registers, for a total of four contiguous 8-bit registers following encoder registers):
- Three 8-bit digital outputs (they are contiguous 8-bit registers starting from base address 0x400), used for
  - 1) the braking system,

- 2) controlling a pan/tilt thermoscan camera,
- 3) setting additional control parameters. Specifically, the ServoDG board triggers an interrupt on IRQ 9 whenever new data are available from the encoders at the corresponding I/O addresses: then, setting / unsetting the 4th bit of this control register has the effect of enabling / disabling interrupts.

By focusing exclusively on the tasks required for autonomous navigation, the on-board PC is requested to execute the following tasks.

J1: it reads wheel encoders and updates the current estimated position of the vehicle on the basis of the rotations of the wheel;

J2: it reads the absolute position from DGPS. The DGPS data are used to correct the position estimate produced by J1: in the case that a big discrepancy is detected between the position estimated by J1 and J2, a message is sent;

J3: it re-computes the trajectory to be followed when a message is received by J2, meaning that the DGPS estimates the position of the vehicle as significantly different from the previous estimate;

J4: it reads pitch and roll data;

J5: it computes the reference speed value to be sent to the two wheels taking into account the estimated current position, the trajectory to be followed, and the slope of the terrain (pitch e roll values);

J6: performs low level control of the two wheel motors on the basis of the reference speed computed by J5 and the values returned by encoders.

On the basis of statistical analyses, the WCET for the execution of tasks J1,..,J6 turn out to be: C1 < 2ms; C2 < 17ms; C3 < 25ms; C4 < 3ms; C5 < 12ms, C6<4ms.

Exercise 1 (max score 10): knowing that the frequency of J1 and J6 must be at least 50Hz, and DGPS data are available with a frequency of 1 Hz:

a (max score 1) tell which tasks should be considered periodic, sporadic, or aperiodic.

b (max score 4) propose a period for periodic tasks and deal properly with sporadic tasks such that the general schedulability theorem is satisfied, as well as sufficient conditions for EDF.

c (max score 4). In the system an aperiodic task J7 is added whose maximum duration is C7=13ms. Assume that all aperiodic tasks in the system are scheduled using Background Scheduling: compute the maximum delay between the request of J7 and its completion in the worst case, using sufficient conditions only (i.e., without computing the idle times).

d (max score 1) Assume that the context switch time is α. Briefly discuss what would happen when considering the context switch of tasks in the schedulability analysis.

Exercise 2 (max score 5): Consider three tasks ordered with descending priority, such that J1’ has maximum priority and J3’ has minimum priority (they are generic tasks with C’1=10, C’2=10, C’3=10, not necessarily corresponding to the previous exercise). Task J1’ and task J2’ share a semaphore S1 which protects critical regions

z_{11} and z_{21} whose duration is $d_{11} = d_{21}$ = 3ms; J2' and J3' share a semaphore S2 which protects critical regions z_{22} and z_{31} whose duration is $d_{22} = d_{31}$ = 3ms; J1' and J3' share a semaphore S3 which protects critical regions z_{12} and z_{32} whose duration is $d_{22} = d_{31}$ = 3ms. Assume that J3' is ready at t0, it tries to access the critical region protected by S3 after 1 ms of its own execution, and the region protected by S2 immediately after. J1' is ready at t0+2m, it tries to access the region protected by S1 after 3 ms of its own execution, and S3 immediately after. J2' is ready at t0+4ms, it tries to access the region protected by S2 after 5 ms of its own execution, and S1 immediately after. Draw the graph of task scheduling using Priority Ceiling for accessing semaphores.

Exercise 3 (max score 4): a – max score 2) Describe in no more than 6 bullet points the differences between RM, EDF and DM scheduling algorithms. **a – max score 2)** Describe, in no more than 10 bullet points, the pros and cons of different mechanisms of communication between real-time tasks, including pros and cons of semaphores and FIFO queues.

Exercise 4 (max score 7): for the tasks J2, J3, J4, J5 and J7 we use Posix threads

a (max score 3) Write a schematic diagram of the code contained in the application above (using natural language comments : "here you do this... here you do that ... "), in order to explain: i) where in the code tasks are initialized, and which actions are needed to perform initialization; ii) where is the code specific to each task; iii) how you can make a task periodic; iii) how it is possible to implement a Polling Server to schedule aperiodic tasks (by making the simplifying assumption that the maximum execution time of an aperiodic task is always less than the capacity of the server).

b (max score 4) write in detail, in the appropriate position: i) functions to initialize the tasks by assigning them priority and the scheduling policy RT_FIFO; ii) the internal structure of the code of a periodic task, highlighting the functions used to implement the waiting cycle which make them periodic; iii) the structure of the code of the Polling Server and of the aperiodic task. Use comments to indicate the points at which application-specific code should be inserted (for example, " here the task acquires sensor data ," etc.).

Please do not write schedulability analysis or other parts of the code that are not requested, but focus exclusively on the issues i), ii) and iii) above. If you want to refer to other parts of the code that are not included in i), ii), iii) just add comments like: "Here I perform schedulability analysis" or similar.

Exercise 5 (9 points): For the execution of the remaining tasks J1, J6 we use RTAI .

a (max score 3) using the reference manual, write a schematic diagram of the code contained in a "minimal" application, so that it is indicated by appropriate comments, i) where both tasks and the system timer are initialized; ii) where is the code specific to each task.

b (max score 3) write in detail in the appropriate position, i) functions to initialize the timer in periodic mode (by setting the proper period), ii) to initialize the tasks and make them periodic with an EDF scheduling policy (if you don't remember it, use a

RM scheduling policy – with a small penalty in your score) iii) the internal structure of the code of J6, including I/O operations required to send commands to actuators by writing in the proper output registers. Use comments in natural language ("the task here does this ...") to highlight the application-specific code of each task.

c (max score 3) In order for J1 to send encoder values to tasks in user space, and J6 to receive speed commands from tasks in user space, two FIFO queues are created. Show how these mechanisms can be implemented, by briefly showing the operations to be performed in kernel space and in user space.