ON-BOARD SPACECRAFT SOFTWARE

PROJECT 1 (Part C)
i386 Module

Tutor: Javier Fernández Muñoz

(jfmunoz@inf.uc3m.es)

Description

The objective is to build a main computing hardware system that controls the main tasks using i386. This subsystem is part of a prototype for reduced onboard software for a satellite, together with a microcontroller subsystem that has a direct access to the sensors and actuators. Both subsystems communicate using a master/slave message protocol defined for this project.

1 Characteristics of the main computing subsystem:

The main computing subsystem oversees the control tasks, sending the command to the Arduino and the interface with the user. Some of the tasks and auxiliary functions were developed in part A of this project. Specifically, these functions are the following:

- Tasks:
    - Select when to activate the heater. (Using the information from the microcontroller).
- Auxiliary functions
    - Send any commands from the protocol list to the Arduino system,
    - Receive the corresponding responses to each command from the Arduino.

Also, there are new tasks that should be implemented this subsystem, such as:

- Send the status of the satellite to the ground system, periodically. (In this case is going to be printed on the screen).
- Send the command to set the Heater to the Arduino.
- Send the command to get the temperature from the Arduino.
- Send the command to get the position from the Arduino.
- Send the command to get the sun light from the Arduino.

The hardware for this subsystem will include:

- One i386 computing system

The connection between both subsystems is done using a master/slave USB-UART serial connection. Specifically, the main computer (i386) will be the master while the Arduino will be the slave.

The source code required for this microcontroller subsystem should be contained in one source file:

- ***i386_code.c***: This is the source file where each of the former tasks should be implemented with one function. Also, the state of the systems is stored using global variables.

2 Description of the software for the main computing subsystem.

The software of the main computing system is based on the i386 code from the first part of the project with the necessary code to include the USB-UART communication functions.

The whole code is comprised on a single file: ***i386_code.c***

This file includes the stubs for all the tasks of the subsystem and for the loop function. It also includes several pre-made functions:

- Function ***getClock ()***: This function returns a double value that represents the current time measured in seconds.
- Function ***delayClock ()***: This function sleeps several seconds expressed with a double value.
- Function ***execute_cmd (enum command cmd)***: This function:
  - Fill the request buffer (***send_cmd_msg***) with the selected command,
  - Send it to the Arduino subsystem,
  - Wait 400 ms until the Arduino has the response ready,
  - Receive the response from the Arduino.
  - Read the response from the buffer and store it (***recv_res_msg)***.
-

The student should codify all these functions as following:

- Auxiliary functions:
  - Function ***send_cmd_msg ()***: This function should set the ***next_cmd_msg*** buffer with the content of the desired request message to be sent. The student should use the version of this function developed on the first part of the project.
  - Function ***recv_res_msg ()***: This function should read from the ***last_res_msg*** buffer the content of the received response message. The student should use the version of this function developed on the first part of the project

- Tasks functions:
  - Function ***control_temperature ()***: This function should activate the heater when the temperature is below the average value and should deactivate it when it is above. The student should use the version of this function developed on the first part of the project.
  - Function ***print_state ()***: This function should print on the screen the state of the system (position, temperature, sunlight, and heater).

The student should also codify the main function as following:

- Function ***controller ()***: This function should implement a cycling or a priority scheduler for all the tasks mentioned above (the student should choose which one to do). This scheduler should implement the following requirements.
  NOTE: The execution times in all command tasks is basically equal to the delay time. The rest of the operations are supposed to be 10 milliseconds or less.
  NOTE 2: The memory should contain the design and validation of this cycling scheduler.

Tasks	Periodo/ Deadline (milisec.)	Execution time (milisec)
A: print_state	5000	10
B: control_temperature	2000	10
C: execute_cmd (READ_TEMP_CMD)	2000	400
D: execute_cmd (SET_HEAT_CMD)	2000	400
E: execute_cmd (READ_SUN_CMD)	4000	400
F: execute_cmd (READ_POS_CMD)	4000	400

3 Documentation to be submitted

To submit the project, the submission web page for the course must be used. It is located on "Aula global" on the project paragraph.

The files to be delivered are the following:

memory.txt | memory.pdf | memory.doc
: Project memory. Including the design and validation of the cycling scheduler

autores.txt
: Names and NIAs of all the students involved in the project.

i386_code.c
: Source file of the tasks for the Arduino subsystem.

<video_file>
: A video recorded of the full project (using the i386 and the Arduino) executing where it can be seen both the screen obtained from the i386 and the breadboard with the LED and the Light Sensor.

Project Deadline

Day of the exam.