Birzeit University - Faculty of Engineering and Technology
Electrical & Computer Engineering Department - ENCS4330 - Final exam
Real-Time Applications & Embedded Systems - 1st semester 2021/22 - 120 minutes

Instructor: Dr. Hanna Bullata

Question 1 (25 points) - Game simulation

You are required to build a simulation for a game that involves 5 blind-folded players, 75 kittens, a referee and a playground. The idea is that the players need to catch the kittens in the playground based on their colors. The player who catches the kittens in the right order and gets the best final score is declared the winner of the game by the referee.

The simulation should consider the following:

- The playground size is 50m×50m. Thus, it contains 2500 locations
- The 75 kittens are placed on fixed locations in the playground. Thus their positions do not change during the game. In addition, the kittens are colored as follows:
  - 25 white kittens,
  - 20 dark grey kittens,
  - 15 light grey kittens,
  - 10 brown kittens,
  - 5 black kittens.
- Before the game starts, the players are placed on random locations in the playground by the referee. Each player has an initial score of 100 points.
- During the game, each player is allowed to move from location to location and try to catch 1 kitten at a time, if present at that location. Each player kitten catching should happen *in order*: 5 white kittens, then 4 dark grey kittens, then 3 light grey kittens, then 2 brown kittens, then 1 black kitten.
- Players scores are incremented by 10 points each time they catch a kitten in the correct order and players are informed about that.
- Kittens caught in the right order are taken out of the playground. Thus, their locations become empty.
- Players scores are decremented by 1 point if they catch the kittens in the wrong order. A Player is informed about the color of the kitten he caught in the wrong order.
- Kittens caught in the wrong order are placed again in the same location by the referee.
- If 2 players end up in the same location by chance, both players scores are decremented by 2 points.
- Players who end up with negative scores are taken out of the game by the referee. If all players end up with negative scores, then the game ends with no winner.

What you should do

- Create a folder called `final` if it doesn't exist already.
- Under folder `final`, create a folder called `final1`.
- Under the folder `final1`, create a file called `final1.txt`. In that file, describe shortly your vision of how you would implement the above-described problem and the data structure(s) you intend to employ. Discuss as well the communication techniques you would employ to make the implementation happen. Keep it short ... 15 - 20 lines max.
- From the above-described problem, implement the behavior of players *only* on your Linux machines using either a multi-processing approach or a multi-threading approach.

Question 2 (30 points) - Hardware

We intend to build a simple alarm system that monitors the level of temperature in a closed area, displays the average temperature values on leds and activates a buzzer if the detected values are out of range. The controller is composed of the following:

- 3 temperature sensors placed at different locations in the closed area (analog sensors). The normal range is [-5° - 40°].
- A set of leds to display the average integer temperature value.
- A buzzer to issue alarm signals.

When the temperature levels are normal, the voltage readings will be in the range [1.5V - 4.2V]. Assume the relationship between temperature and voltage is linear.

The buzzer must be activated with a 15KHZ signal when:

- The average temperature of the closed area is out of range.
- The temperature of any of the sensors is out of range. In that case, the leds must display the temperature sensor number [1 - 3], pause for a second, then display the sensor temperature value, pause for a second. The above behavior continues as long as the alarm situation is present.

The buzzer should go OFF when the temperature values go within the normal range.

What you should do

- Create a folder called `final` if it doesn't exist already.
- Under folder `final`, create a folder called `final2`
- Under the folder `final2`, create a file called `final2.txt`. In that file, describe shortly your vision of how you would implement the above-described problem and the connections you would make. Keep it short ... 15 - 20 lines max.
- Draw a clear design of the alarm system on your answer sheet. If you prefer to do the design using proteus, please include a snapshot of the design and save it as an image.
- From the above-described problem, write the PIC assembly code for the controller under mplab. Name the project `alarm` and name the PIC assembly file `alarm.asm`.
- Compile and run your code.

Question 3 (45 points) - Multiprocessor hardware

We intend to build a controller for a maternity hospital composed of 32 incubators. Each incubator can read the newly-born body temperature, heart beat & blood pressure. However, since the program is too big to fit in any single-core controller, we had to split the controller into 4 parts:

- A master controller connected to a 16× 2 LCD mainly responsible for data displaying. A push button is also connected to that master controller to change the display values (see below for details). In addition, a buzzer is also connected to signal alarming conditions (e.g. abnormal blood pressure, too low/high heart beats, etc).
- A slave 1 controller responsible to read the newly-born heart beat (integer value). The normal range is [60 - 120].
- A slave 2 controller responsible to read the newly-born body temperature (analog value). The normal range is [35° - 37°].
- A slave 3 controller responsible to read the newly-born blood pressure (digital value). 1: normal, 0: abnormal.

The operation of the overall system is summarized as follows:

- The 3 slave controllers are connected to the master controller for display and alarm purposes.
- Under normal conditions, the master controller should display on the first row the message "**Heart Beat X**" where **X** is the incubator number (range [0 - 31]) and then display the heart beat value of incubator **X** on the second row. After a second, the master controller should display the message "**Body temp X**" on the first row of the LCD and then display the body temperature value of incubator **X** on the second row. The same occurs to the blood pressure.
- If an emergency situation occurs in an incubator, the behavior described above should be changed. For example, if the heart beat at incubator 10 is higher than 120, the master controller should display the message "**Heart Warn 10**" on the first row in a blinking fashion (0.5 second ON, 0.5 second OFF) and the heart beat value should be displayed on the second row. The buzzer must be put ON to draw attention. Clicking on the push button of the master must reset that warning and turn OFF the buzzer.

What you should do

- Assume all controllers (master & slaves) are based on PIC16F877A.
- Create a folder called `final` if it doesn't exist already.
- Under folder `final`, create a folder called `final3`
- Under the folder `final3`, create a file called `final3.txt`. In that file, describe shortly your vision of how you would implement the above-described problem and the connections you would make between the master and slave controllers. Discuss as well how that communication can happen without collision. Keep it short ... 15 - 20 lines max.
- Draw a design of the whole system on your answer sheet. If you prefer to do the design using proteus, please include a snapshot of the design and save it as an image.
- Draw a clear design for the master controller *only*.
- From the above-described problem, implement the behavior of the master controller *only* under mplab. Name the project `master` and name the PIC assembly file `master.asm`.
- Compile and run your code.

Once done with your exam, please zip the folder `final` that contains the 3 folders `final1`, `final2` and `final3`. Send the zipped folder as a reply to my ritaj memo entitled **encs4330 final exam - February 27, 2022**.