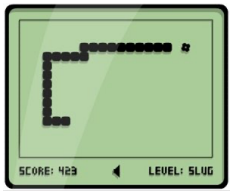
Embedded System Lab – Final Project : SNAKE GO !!

6130287521 Nasri Islam section 1

Snake is a simple game that's popularity peaked in the late 1990s when it was introduced to a new generation on Nokia mobile phones [3]. The game is not resource intensive and can be played on small, low resolution screens. Push buttons may be used to control the snake, or keyboard input via serial UART as an alternative. This report describes the operation of various functions.



This game consists of 3 main components (main, snake, queue)

Main.c & main.h : Handle 2 interrupts. UART Interrupting handle the receiving of data from the keyboard. Timmer Interrupts handles screen refresh.

`void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)` : timer Interrupt for screen refresh.
`void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart)` : Interrupted when there is input from the keyboard.

snake.c & snake.h : This file control snake movements, counting points (score) leveling and changing the screen.

`void snake_init(char scene[1920])` : Initialize snake.
`void snake_enqueue(int x, int y, char scene[1920])` : Add new snake head
`void snake_dequeue(char scene[1920])` : Remove snake tail
`void snake_setDirection(int direction)` : Change snake's direction of movement
`void snake_move(char scene[1920])` : Manage one-step movement of snakes
`void scene_setScore(char scene[1920])` : Updated score on the screen
`int _snake_foodScore(int x, int y, char scene[1920])` : Return the score of food in front of the snake. (return the value -99 if the front is the snake itself.)
`void snake_feed(int foodScore, char scene[1920])` : Increase points when the snake eats food.
`void snake_newFood(char scene[1920])` : Add a new piece of food ('O') on the screen.
`void snake_newObstacle(char scene[1920])` : Add new obstacles ('#') on the screen.
`void snake_gameOver(char scene[1920])` : Change the screen to the game over screen.
`void snake_levelUp(char scene[1920])` : Increases the level of snakes when they reach the required score.
`void scene_setLevel(char scene[1920])` : Updated snake level on screen
`void scene_setPixelX(int x1, int x2, int y, char scene[1920], char text[], int size)` : Add characters on the screen in the x-axis
`void scene_setPixelY(int x, int y1, int y2, char scene[1920], char text[], int size)` : Add characters on the screen in the y-axis
`void scene_mainmenu(char scene[1920])` : Updated the screen to the main menu screen.
`void scene_clear(char scene[1920])` : Update the screen to a blank screen.
`char intTOchar(int integers)` : convert integer to character.

Queue.c & queue.h : This file contains the queue, which is the data structure for snake movements.

`struct Queue* createQueue(unsigned capacity)` : Create a new queue
`int isFull(struct Queue* queue)` : Check if the queue is full
`int isEmpty(struct Queue* queue)` : Check if the queue is empty.
`void enqueue(struct Queue* queue, int item)` : add one element in queue.
`int dequeue(struct Queue* queue)` : delete one element in queue.
`int front(struct Queue* queue)` : return head of the queue.
`int rear(struct Queue* queue)` : return tail of the queue