Architetture dei Sistemi **Di Elaborazione** **Extra point #1** **Max 2 points**  	Delivery date: 12th January 2025 Expected delivery of extapoint1.zip must include: - zipped project folder - this document compiled in pdf - A 4 minutes video with audio (.mp4 or .avi) explaining how the project works; the video has to show a software debug session with all significant peripheral windows opened; the audio must be a voice recording of you describing (in Italian or English) the behavior of the running system.

Purpose of Part 1: to acquire full confidence in the usage of the KEIL **software debug** (Compilation Target SW_Debug) environment to emulate the behaviour of the LPC1768 and the LANDTIGER Board.

This part is evaluated to assign a maximum of 2 extra-points for qualified students taking the exam with a mark >= 18

Pac-Man

Pac-Man, originally known as Puck Man in Japan, is a maze video game developed and released by Namco in 1980. The game was later released in North America by Midway Manufacturing. The inspiration for the Pac-Man character came from a pizza with a slice removed, and the game's characters were designed to be cute and colorful to appeal to younger players. In the game, Pac-Man eats the pills scattered throughout the maze to make points, while avoiding the ghosts that chase him. You can try online Pac-Man here.

1) Implementation details for LandTiger Board

In Keil µVision, use the LANDTIGER **emulator** (Compilation target: SW_Debug) for implementing a Pac-Man game.

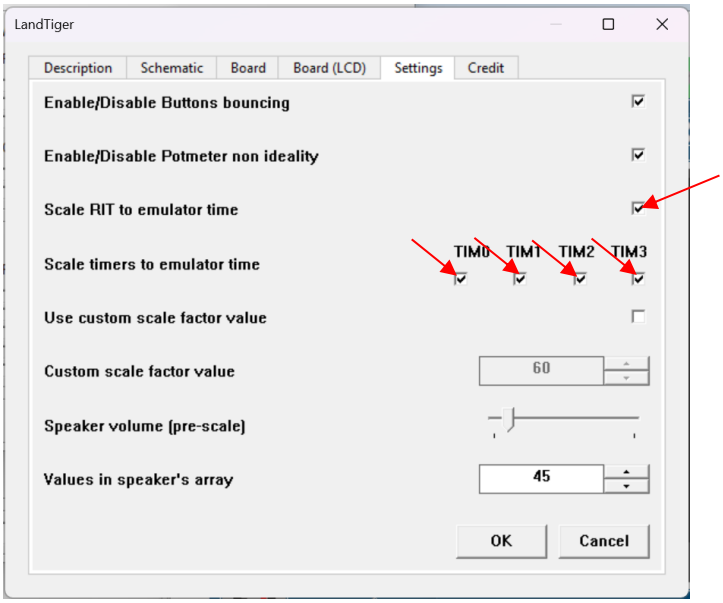
If you have an available physical LandTiger Board you can directly develop the extra points without using the emulator.

Please deliver a zip folder with all the files of your project (you must save the project with all the compilation options you used).

Example: extrapoint1.zip

Please attach any useful comments and your emulator configurations (substitute the image below).

Your project will be evaluated according to your specification!!



Additional Comments (C-variable/defines defined in your code, e.g., scaling factors) :
Il primo extrapoint (oltre che il secondo) viene **consegnato su scheda**, pertanto si consiglia l'utilizzo della board per testare il lavoro (seppur sia perfettamente funzionante anche in emulazione). Come riferito dal Prof. Sanchez, la gestione delle **vite rimanenti** in questa prima versione del gioco è a nostra libera interpretazione. Pertanto, ho introdotto l'utilizzo del pulsante **KEY1** per riavviare il gioco quando il tempo termina ma ci sono ancora vite rimanenti. Si sceglie inoltre di riavviare il gioco nella stessa situazione della partita precedente, rendendolo in questa prima versione più facile. **Attenzione**!: ho riscontrato che, in caso di sessione di debug via software, è necessario aprire nella sezione "Peripherals" il **RIT**. In caso contrario potrebbe non funzionare correttamente. Principali **variabili** di gioco: uint16_t countdown; uint16_t numLives; uint16_t score; uint16_t gamePause;

uint16_t direction;
Si sceglie uno **scaling ratio** pari ad **8** per la matrice rappresentante la mappa di gioco rispetto ai pixel del display effettivo.

Utilizzo delle seguenti **define** per la gestione della **randomicità** (il metodo è meglio esplicato nel video):

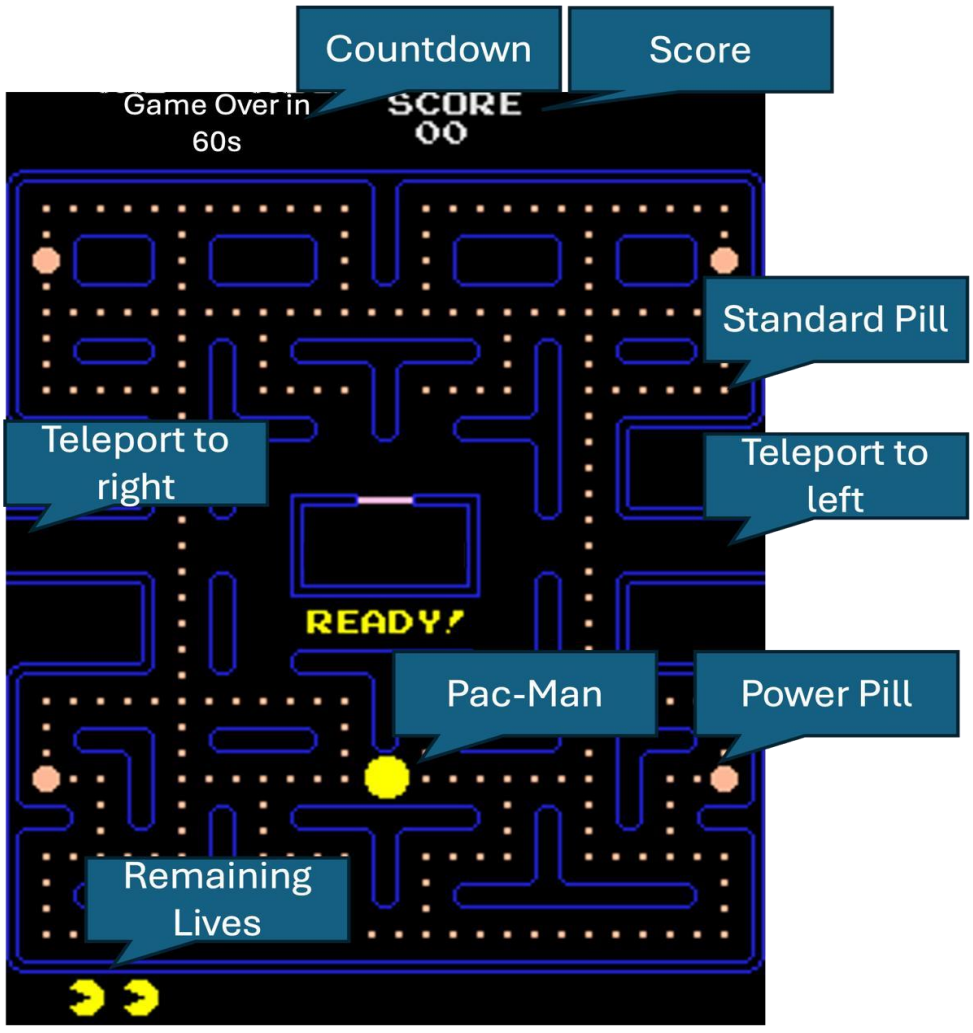
```
#ifdef SIMULATOR
#define PROBABILITY_TRESHOLD 60
#else
#define PROBABILITY_TRESHOLD 10
#endif
#define MAGIC_RANDOM_NUMBER 1103515245
```

Specifications

Note that images are only for explanation purposes.

You must implement for the LandTiger board the Pac-Man game.

An implementation example could be the following. Please note that your project will not be evaluated based on your "artistic" skills, but on your programming skills.



Use the display of the LandTiger to show the labyrinth for Pac-Man.

You are free to choose the dimension and the layout of the labyrinth but make it so it fits exactly 240 Standard Pills. Please keep the “central” box, as it may be used later.

You also need to print the current **Score** of the game and the **Remaining Lives** and a **Countdown timer**. In this initial version there will be no ghosts and no dangers in general (unless Countdown timer expires).

As for the game mechanics you are requested to integrate the following rules:

Spec. 1) Fill the Labyrinth with **240** Standard Pills.

Spec. 2) Generate **6** Power pills in random position. These pills replace the Standard Pill in their location. You have to implement the **randomness** of Power Pills appearance both for the position and time.

Spec. 3) The player can move Pac-Man through the **joystick**. Once the direction is chosen (left, right, up, down), Pac-Man continues to go in the specified direction until:

a. The player selects a new direction through the joystick.
b. Pac-Man encounters a wall, in this case he stops and wait for player input.

Freely choose Pac-Man speed (just make it “playable”)

Spec. 4) When the player reaches the central 'teleport' locations (refer to the figure), Pac-Man exits from one side and re-enters from the other side while maintaining the original movement direction.

Spec. 5) Each time Pac-Man goes on top of a Pill, it eats it and the Score counter increase by 10 (Standard Pills) or 50 (Power Pills).

Spec. 6) Every 1000 points, Pac-Man earns a new life (from a starting count of 1).

Spec. 7) **INT0** *Pause* the game: Place a “PAUSE” message in the middle of the screen and stop Pac-Man movement. Pressing again INT0 *resume* the game (delete the Pause message and resume movement). The game starts in “PAUSE” mode.

Spec. 8) The Countdown starts from 60 seconds and count down up to 0.

Spec. 9) Once the player cleans up the labyrinth (by eating all the Pills), Pac-Man stops moving and a 'Victory!' screen is shown.

Spec. 10) If the countdown timer expires before all pills have been eaten, a “Game Over!” screen is shown instead.