ASSIGNMENT 1

Simple Project for Beginner: Pac-Man

SAIGON, SEPTEMBER 2025

BKIT HARDWARE CLUB
Trần Nguyễn Minh Duy
tnmduy@hcmut.edu.vn

Contents

1. Assignment's outcome 2

2. Requirement 2

3. Introduce Pac-Man 2

4. Instruction 3

4.1. Simplified description of Pac-Man for BKIT ARM 4 3

4.2. Coding 4

4.2.1. Set up project 4

4.2.2. Design 4

4.2.3. Objects in game 5

4.2.4. Coding 7

4.2.5. Additional requirements 7

5. References 7

ASSIGNMENT 1

Simple Project for Beginner: Pac-Man

1. Assignment's outcome

After completing this assignment, students review and make good use of:

- Basic C programming: organize projects properly; use libraries.
- Review the knowledge learned in LAB lessons 1, 2, 3.
- Know how to make a simple game to play.

2. Requirement

- Limit the use of the HAL_Delay function, you should only use Timer.
- LCD library: The library functions of this library will display objects directly and immediately to the screen, so you should use it appropriately (only use it when there is a change) and avoid clearing the screen. image and output continuously.

3. Introduce Pac-Man

Originally launched in Japan as Puck Man, Pac-Man is a maze-based action video game from 1980, developed and launched by Namco for arcade platforms. In North America, Midway Manufacturing distributed the game under a licensing deal with Namco America. The gameplay involves guiding Pac-Man through a sealed maze, where the objective is to consume all the dots present while evading four colored ghosts. Consuming special, larger dots known as "Power Pellets" makes the ghosts turn blue temporarily, during which Pac-Man can eat the ghosts to score extra points.[1]

[1] "Pac-Man Official Website – History". Pac-Man Official Website. Retrieved April 26, 2022.

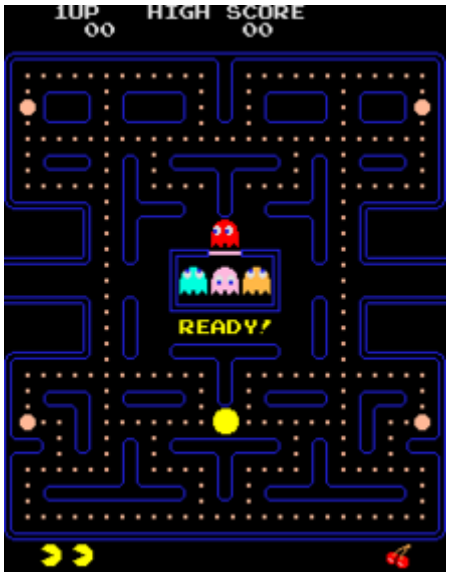

Figure 1: Pac-Man[1]

You can go to the following link https://www.google.com/logos/2010/pacman10-i.html to experience and try out the game.

Pac-Man, a maze chase action video game, features the player guiding the titular character through a confined labyrinth. The goal is to consume every dot scattered throughout the maze, all while dodging four differently colored ghosts—Blinky (red), Pinky (pink), Inky (cyan), and Clyde (orange)—that chase after Pac-Man. Completing a maze by eating all the dots moves the player to the subsequent level.[2]

Because Pac-Man is a fairly simple game and suitable for beginners to practice programming, it will be the topic of this Assignment. And remember that in this Assignment, you don't need to remake this game exactly like the original, but just make a simple version of it.

4. Instruction

You should do it yourself. We encourage your independence and creativity.

Now let's start coding. (You can leave here now!)

However, if you don't know where to start, that's okay. We will help you a little.

4.1. Simplified description of Pac-Man for BKIT ARM 4

Since we're beginners, I'll simplify this game as follows:

- Maze: The game maze is a large square containing NxN small squares. There is no wall in this maze, so Pac-Man and Ghost can move freely.
- Control Pac-Man: Use the 4 buttons numbers 2, 8, 4, 6 to replace the 4 arrow buttons up, down, left, right.

[1] "Pac-Man Official Website – History". Pac-Man Official Website. Retrieved April 26, 2022.
[2] "Video Game Flyers: Pac-Man, Midway Manufacturing Co. (France)". The Arcade Flyer Archive. Retrieved April 8, 2021.

- Display:
    - Game is displayed on LCD.
    - Score is displayed on 7-segment LEDs.
- Simplified game rules:
    - When the game starts, all cells in the maze will have one pac dot available, except Pac-man's cell.
    - Pac-Man eats a Pac dot to get a score. If Pac-Man eats all of the dots, the player wins and the game will be restarted.
    - Ghost will move randomly. If Ghost encounters Pac-Man, the game will end and be restarted. Moreover, Ghost cannot eat pac dot.

Remember that you can come up with your own game specifications.

4.2. Coding

I have created a code project sample, you can download and preview!

https://github.com/duytran1511/BKIT_Arm4/tree/main/Assignment/Assignment1_Pacman/Assignment1_Pacman_Code

After that, you can flash the code at the following link to try out the super simple program that my team prepared.

https://github.com/duytran1511/BKIT_Arm4/tree/main/Assignment/Assignment1_Pacman/Assignment1_Pacman_Firmware

Now let's start coding!

4.2.1. Set up project

Firstly, we need a code project template. Therefore, we duplicated the project “Lab3_LCD_Button” for this project. We do that because this project has all the necessary libraries (timer, button, lcd).

Secondly, we create the files pacman.h and pacman.c to program this project. More specifically, in the header file, we only need to declare the game's configuration parameters (as #define) and 2 public functions, game_init and game_process, to call in the main.c.

4.2.2. Design

Before starting programming, we must design the game's interface and it is the maze.

To design the maze, we need to remember that the LCD screen is 240*320. Therefore, we will design the maze with dimensions of 10x10 squares, with the width of each small square being 20*20.

At the same time, we also let the maze exit the frame of the screen at a distance of 20.

The specific design is as follows:

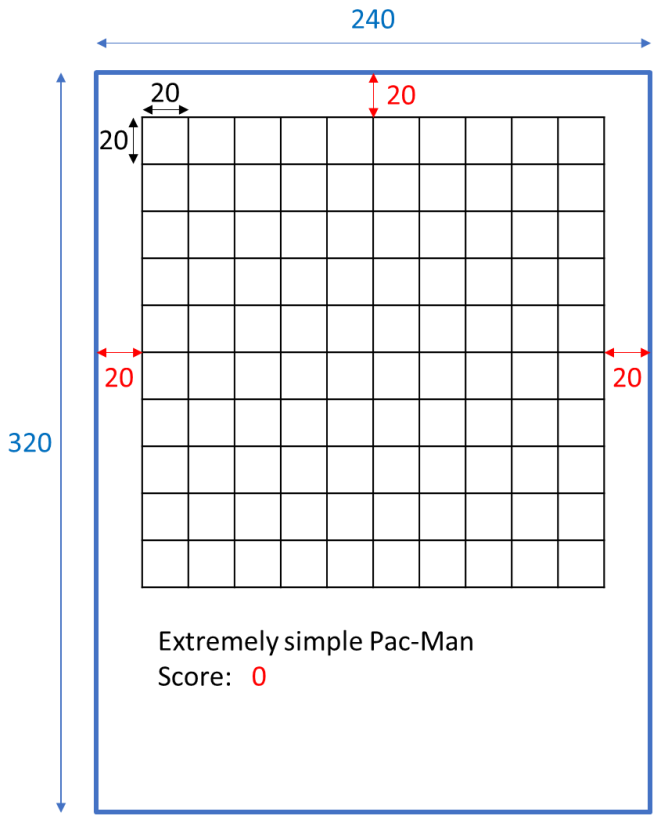

Figure 2: Game Design

4.2.3. Objects in game

Next, to make programming easier, we need to identify the objects in the game to handle them separately.

From the requirements described above, we can identify the following objects:

- Maze that has the number of Pac dots
- Pac-Man
- Ghost(s)
- Game Engine

To manage these objects, we will declare enum and structs for easier management. As follows:

```
/* Enums ------------------------------------------------------------------*/

typedef enum DIRECTION {
      UP, DOWN, LEFT, RIGHT, STOP
} E_DIRECTION;
```

```
/* Struct -----------------------------------------------------------*/
typedef struct CELL {
	uint8_t is_pac_dot;
} S_CELL;

typedef struct MAZE {
	S_CELL cells[MAZE_COLUMN_N][MAZE_ROW_N];
} S_MAZE;

typedef struct GHOST {
	uint8_t i, j;
	uint8_t i_pre, j_pre;
	E_DIRECTION direction;
} S_GHOST;

typedef struct PACMAN {
	uint8_t i, j;
	uint8_t i_pre, j_pre;
	E_DIRECTION direction;
	int score;
} S_PACMAN;
```

Figure 3: Enum and Structs in pacman.c

Next, we declare variables (in struct type) and functions to manage each corresponding object.

```
/* Private Objects ---------------------------------------------------*/
// Pac-Man object
S_PACMAN pacman;
void pacman_draw(uint8_t i, uint8_t j, uint16_t color);
void pacman_direction_process(void);
void pacman_moving_process(void);

// Ghost object
S_GHOST ghost;
void ghost_draw(uint8_t i, uint8_t j, uint16_t color);
void ghost_direction_process(void);
void ghost_moving_process(void);

// Maze object
S_MAZE maze;
void pac_dot_draw(uint8_t i, uint8_t j, uint16_t color);

// Game Engine object
void game_draw(void);
void game_handler(void);
```

Figure 4: Objects in pacmain.c

Now we can start to program.

4.2.4. Coding

After fully preparing, we start programming. We have built a template for the entire program (necessary variable declarations, functions...). Your task is to fill in the missing parts (where there is "TO DO"), so that the program works.

Now let's download the sample and start coding. Good luck!

4.2.5. Additional requirements

After completing the basic program, you need to add the following functions:

- If we press the enter button (last button at index 15) in 3 seconds, the game will start again.
- If score < 30% of total score, toggle LED Y0 every 1 second.
- If 30% <= score < 75% of total score, toggle LED Y1 every 500ms.
- If score >= 75% of total score, toggle LED Debug every 250ms.
- (Advanced) Create 4 ghosts that move randomly.
- (Advanced) Creates walls in the maze
- (Advanced) Make Pac-Man and Ghosts move smoothly.

5. References

- "Pac-Man Official Website – History". Pac-Man Official Website. Retrieved April 26, 2022.
- Pacman Doodle. Access from https://www.google.com/logos/2010/pacman10-i.html. Retrieved March 15, 2024.
- "Video Game Flyers: Pac-Man, Midway Manufacturing Co. (France)". The Arcade Flyer Archive. Retrieved April 8, 2021.