uc3m | Universidad Carlos III de Madrid
Departamento de Tecnología Electrónica

Laboratory Manual for Microprocessor-based Digital Systems

Bachelor in Mobile and Space Communications Engineering

Bachelor in Sound and Image Engineering

Bachelor in Telematics Engineering

Bachelor in Telecommunication Technologies Engineering

2nd Course, 2nd Semester

Academic year 2021-2022

REACTION GAMES

uc3m | Universidad Carlos III de Madrid

DEPARTAMENTO DE TECNOLOGÍA ELECTRÓNICA

Campus de Leganés

Avenida del Universidad 30

1. ORGANIZATION AND ASSESSMENT

The work will be done in teams of 3 people. There will be 4 mandatory 2-hour laboratory sessions. The development of the sessions should be worked on prior to the laboratory sessions.

For the development of the work, one of the following STMicroelectronics development boards will be used, either Discovery (STM32L152C-Discovery or STM32L-Discovery) or the NUCLEO-L152RE. All the documentation on the microcontroller used and the development boards is available online both on the manufacturer's website and in Aula Global

For each session, some milestones are proposed, to guide the students to achieve the complete project.

The assessment is made up of two parts: **the project mark (80%) and the laboratory exam (20%).** In the case of the project mark, each milestone has a weight in such mark: **Session 1 (15% of the project), Session 2 (35% of the project), Session 3 (35% of the project), and Session 4 (15% of the project).** For the assessment of the project, each student will upload to AG the source code of their project, its documentation and a demonstration video of the functionalities achieved. Specific instructions regarding this delivery will be provided through AG. The deadline for the delivery of the project is **May 3, 2022 at 11:59 p.m.**

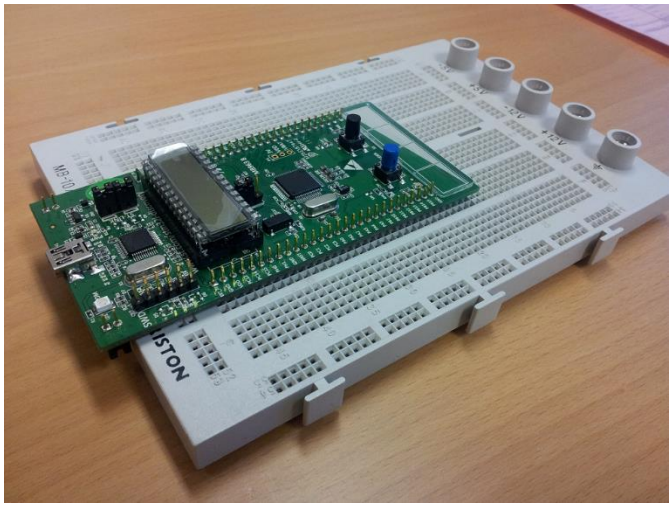
All students involved in trying to pass the Laboratory by copying or outsourcing the work, will automatically fail the Lab with a 0 out of 10. This will be applied to those having copied and those having allowed the others to copy

2. INTRODUCTION

The sessions are structured so that you will end up developing a full system. Therefore, all your work will be used in subsequent sessions. This is done to:

- Learn basic programming techniques for microcontrollers
- Learn to design a set of hardware-software to fulfil a particular application set of requirements
- Learn to debug and test the behaviour of the hardware-software development
- Offer an affordable challenge for students that will stimulate their design and innovation capabilities
- Stimulate the cooperation capabilities among students and their participation in team works
- For the Discovery boards:

- To build the design, it is advised to insert the development board (only for the Discovery boards) in a breadboard, to avoid wrong interconnection of the P1 and P2 port pins. This will allow to make independent connections to each pin, to connect resistances, potentiometers, etc.
- It is very important to keep the two bottom jumpers connected, as seen in the attached figures. If this connection is not possible, the jumpers should be removed and the pins, which were connected to the jumpers, should be connected through the breadboard.



- **It is mandatory to put jumper J1 in its OFF position** (connecting the two pins which are furthest from the LCD).

3. PROJECT DESCRIPTION AND DEVELOPMENT

The Lab Project consists of the development and implementation of 2 different Reaction Games. The user will first select which game to play and then start playing the chosen game. The following games will be designed:

- Game 1: Reaction Time
- Game 2: Countdown

Bill of Materials

In order to develop the work, you will have to buy the following material. It is provided in Spanish (with English translation) so as to help you when going to a shop:

- 2 placas de pruebas (protoboards) pequeñas o una placa grande. (*2 small breadboards or just one single large breadboard*). In case you use the NUCLEO board, you may be able to develop this project with one single breadboard.
- 2 pulsadores para inserción en placa. Se recomienda el pulsador 11.521/2 MEMBRANA. En algunas tiendas lo identifican con el código 0330052. (2 *push buttons for breadboard insertion coded as 11.521/2 membrane. At some shops, it is identified with the code 0330052*)

- 1 zumbador HS1212A. En algunas tiendas lo identifican con el código 0280075. (1 *buzzer coded as HS121A. At some shops, it is identified with the code 0280075*)

- Potenciómetro de 1 vuelta de 5 kΩ. **SE RECOMIENDA QUE EL POTENCIOMETRO NO SEA MULTIVUELTA** (*1 variable resistor of 5KΩ – do not use multi-turn ones*)
- 2 LEDs de colores diferentes + 2 resistencias de polarización de los LEDs (330Ω). *(2 LEDs with different colours and 2 resistors of 300 Ω)*
- Cable 0.5 mm para protoboard, y Cables DuPont macho-hembra (*Cable for protoboard connections of 0.5mm and DuPont cables male-female*). If you're using the NUCLEO board, you may also use male-male DuPont cables for some connections.
- Pelacable y cortacable (*cable cutter and cable peeler*)

There is no need to supply any external voltage to the protoboard, as all power is supplied by the development board, which gets the main power from the USB connector. The boards also provide some pins connected to GROUND, 3V and 5V.

The development is structured in several intermediate steps, as described below. Pins to be used for the whole project are given in Annex 2, to allow that all groups will have the same hardware distribution and port configuration.

We recommend reading carefully all steps (i.e., this whole manual) previously to the beginning of the project.

Each milestone is designed to be completed after one lab session, although you shall get to the lab with your solution already developed for that milestone, leaving the lab session for the professors to mark your achievements, locate errors, and allow you to correct such errors. To achieve the maximum score in the laboratory, it is recommended to reach the milestones regularly in each session (i.e., it is not mandatory to achieve each milestone at each lab session, but it is strongly encouraged).

Milestone 1: Player interface and Game 1: Reaction Time

To achieve Milestone 1 designed for laboratory session 1, two objectives must be met:

- Player Interface
- Game 1: Reaction Time

Player Interface

The USER button on the development board will be used to switch between the implemented games. In this way, when you start the application, it will start in Game 1 and “GAME 1” will be displayed on the LCD/console. If the user presses the USER button, then it will switch to game 2, and “GAME 2” is displayed on the LCD/console. If you press the USER button again, then “GAME 1” will be displayed on the LCD/console again, and so on. Therefore, the USER button will be used to switch games.

For this first session, we are only going to implement GAME 1, so when we have the text “GAME 1” on the LCD/console, this game will start to be played.

Pressing the USER button at any time will change the current active game to the next one.

Game 1: Reaction Time

This game will be played by 2 players and two buttons will be used, BUTTON1 and BUTTON2, and a led, LED1. Player 1 will use BUTTON1 and player 2 will use BUTTON2. Once the game starts, both players must pay attention to when LED1 lights up and when this happens, they must press their

respective buttons. The player who pressed the button first wins. The winning player will be shown on the LCD/console display (P1 or P2).

This game will be completed in the next milestone (session 2) adding a random time for the LED to be turned on. For this session, you can use a fixed time. For generating such time, for this milestone, it is not needed to user timers, i.e., you can use the espera() function.

Either pull-up or pull-down resistors must be used depending on the connection made to the buttons. The resistors can be external or internally configurable.

Interrupts must be used to attend the pressing of the buttons. Delays or writing to the LCD/console are not allowed in ISRs.

As long as we are in one of the games, matches will be played continuously. When one match is finished, the next one will start and the game can only be abandoned by pressing the USER button.

LED1 will have to be on for a certain time and then go off for the game to start again.

Tips for the implementation:

Use the debugging mechanisms available to check the functionality of your solution, and if you find some errors or misbehaviour solve them. Remember that within the debugger, you have the following mechanisms among many others:

- Step by step execution
- Breakpoints
- Variable inspection
- Peripheral registers inspection

Milestone 2: Game 1 improvement and Game 2 development.

Game 1: Add randomness and use timers with interrupts.

In this session, game 1 will be modified to add a random time for the game 1 led to turn on. In addition, the time it took for the winning player to press the BUTTON after the LED1 turned on must be calculated and the player displayed on the LCD/console. winner and the time it took to click in milliseconds with "-Yxxxx" format, where Y is the number of the winning player and xxxx is the time.

All timing requirements within the game must be changed/solved by using timers with interrupts. Therefore, change the way you solved timing in Milestone 1, with one using timers and interrupts.

Game 2: Countdown

For this game, BUTTON1, LED1, BUTTON2, and LED2 will be used and two players will participate, so player 1 will use BUTTON1 and LED1, while player 2 will use BUTTON2 and LED2.

For this game, a countdown starting at 10 will be displayed on the LCD/console. At a random instant, the countdown will stop displaying on the LCD/console. Players will have to continue "mentally" decreasing the count and guess when the count reaches 0. Whenever one of the players thinks that the count has reached zero, they must press their corresponding button (player 1, BUTTON1; and player 2, BUTTON2). The countdown will decrease in fixed time steps (this time will be configurable in the following milestones. You can use 1 second for this session).

The player who has got the closest time in pressing compared to when the count reaches 0, will win. Information on how close the buttons have been pressed with respect to the moment the count reaches the end will be displayed on the LCD/console. For example, if player 1 has pressed 400ms before the countdown has reached the end and player 2 has pressed 800ms before the countdown has reached the end, then LED1 (game winner) will light up and display by LCD/console “-400”.

In the event that player 1 has pressed 1000 ms before the count reaches the end and player 2 has pressed 200 ms after the count reaches the end, then LED 2 will light up and the LCD/console will show “+200”.

For the implementation of this milestone, all timings must be done using Timers, which must be attended by interrupts.

For both games, a new match starts 3 seconds after the end of the current match.

Milestone 3: An extended version of Game 2

In this milestone we are going to allow variations of the countdown speed. The countdown steps can last from 500 ms to 2 seconds. That is, if the countdown starts at 10 (10 is displayed on the LCD/console), the value 9 will appear on the LCD/console after the selected time. This variation of the counting speed will be configured by means of the potentiometer. The student must find an equivalence between the voltage value read through the analog to digital converter (ADC) and the countdown speed selected in the game. For the achievement of this milestone, it is enough to be able to distinguish among 3 different step timings: 500 ms, 1 s, and 2 s.

In addition, in the event that the winning player has pressed his button before the countdown reaches the end, then the BUZZER will sound MELODY 1, while, if the winning player has pressed his button after the countdown, then the BUZZER will sound MELODY 2. In Annex 1 you can find a summary of musical notes. To generate a melody, you must generate a square signal with varying frequency depending on the notes used.

After 2 seconds from the end of the countdown, if there is a winning player, the time will be displayed on the LCD/console, the corresponding LED will light up and the appropriate song will sound. If none of the players has pressed their button, the match will be over 2 seconds after the end of the countdown, and “END” will be displayed on the LCD/console.

Milestone 4: Project Improvement

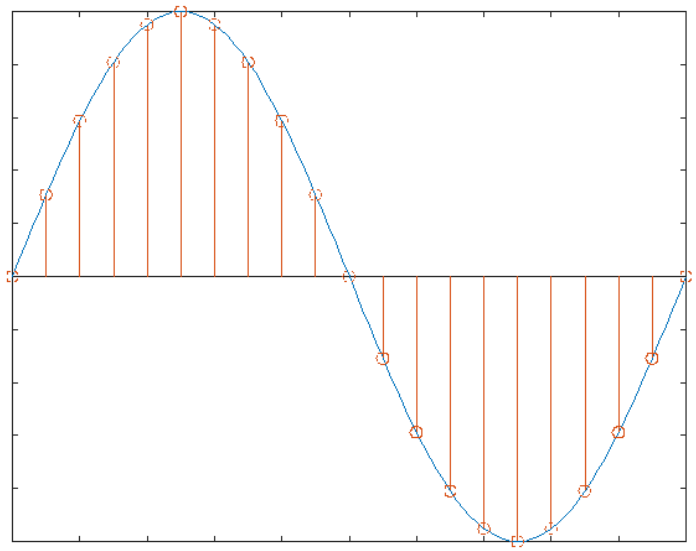
The second extension will consist of using serial communication to communicate the microcontroller with a PC. To do this, the UART peripheral and a terminal on the PC will be used both to display the messages sent from the microcontroller and to send messages to it. USART reception will have to be implemented with interrupts. Serial communication will be used for:

- Send to the PC the same messages that are shown on the LCD/console regarding the winner and the reaction times. If you're using the NUCLEO board, the printf() function is no longer allowed in the project, so you will have to change all the previously programmed calls to printf().
- Replace the selection of the game (Game 1 or Game 2) that was previously done through the USER button, and now do it by sending commands from the PC to the microcontroller indicating the game that you want to play. When starting the game, a welcome message will be displayed: “Welcome to the game of reflexes, select: game 1 REACTION TIME (press 1), game 2 COUNTDOWN (press 2).
- If you want to end the game in progress, press the X key and the initial message will be displayed again.

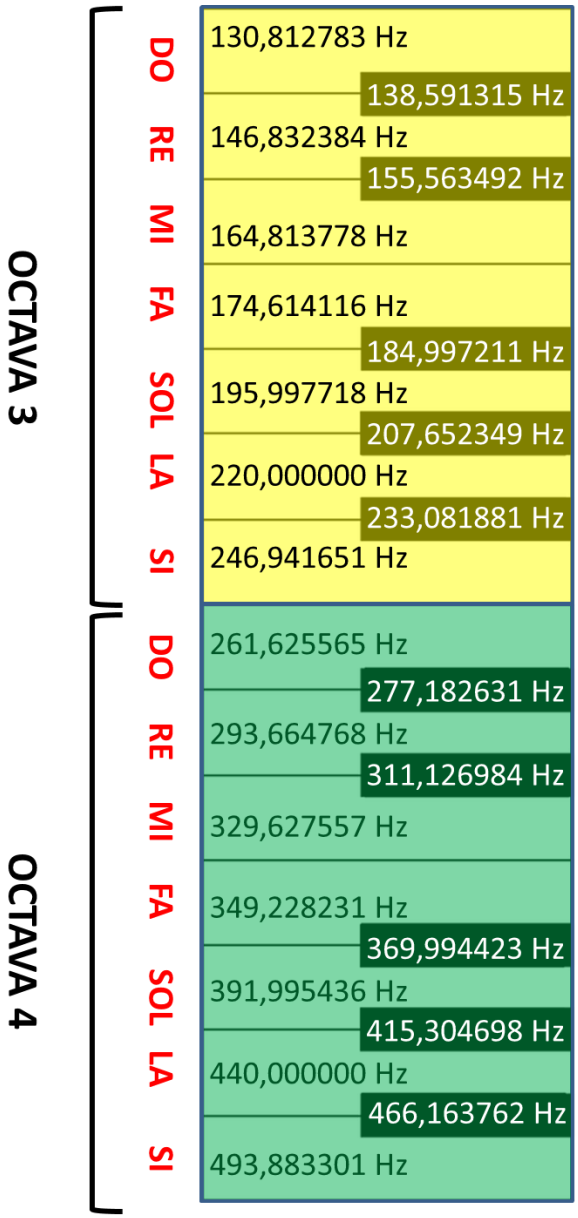
Students have the possibility to carry out an **optional improvement**. This extension can be proposed by the students, but it must be previously approved by the group coordinator, so that he/she validates the degree of difficulty of the improvement.

The achievement of the optional improvement will allow the group to obtain an extra point that will be added to the final grade of the Lab work.

As an example, an extension can be to use the DAC to generate the sounds of Milestone 3. By using the DAC register, the programmer must generate a sine wave with the frequency of the selected melody. For the implementation, a table of values of a sine must be used so that these values are periodically read depending on the frequency of the note to be played. The following figure shows the form of the values that would have to be calculated and placed in the DAC register periodically. For this purpose, you must take into account the number of bits of the DAC.



Annex 1. Frequencies for the main musical notes, Octaves 3 and 4



Annex 2. Pin-out distribution and port configuration

Discovery Boards

Name	Pin	Configuration	Component
USER button	PA0	Digital Input	USER button (Discovery board)
BOTON1 button	PB7	Digital Input	External button
LED1	PA12	Digital Output	External LED
BOTON2 button	PB6	Digital Input	External button
LED2	PD2	Digital Output	External LED
ADC_IN4	PA4	Analog input	Variable resistor
BUZZER	PA5	Alternate Function (Timer)	Buzzer
Optional: (DAC)	PA5	Analog output	Optional improvement
(USART2_RX)	PD6	Alternate Function	PC connection with UART-USB dongle In the default setup of the Discovery board, this pin is part of the LCD connection. As you're not using the USART connection, you do not have to change the setup for this pin
(USART2_TX)	PD5	Alternate Function	PC connection with UART-USB dongle In the default setup of the Discovery board, this pin is part of the LCD connection. As you're not using the USART connection, you do not have to change the setup for this pin

NUCLEO Boards

Name	Pin	Configuration	Component
USER button	PC13	Digital Input	USER button (NUCLEO board)
BOTON1 button	PB7	Digital Input	External button
LED1	PA12	Digital Output	External LED
BOTON2 button	PB6	Digital Input	External button
LED2	PD2	Digital Output	External LED
ADC_IN4	PA4	Analog input	Variable resistor
BUZZER	PA5	Alternate Function (Timer)	Buzzer
Optional: (DAC)	PA5	Analog output	Optional improvement
(USART2_RX)	PA3	Alternate Function	PC connection through USB cable
(USART2_TX)	PA2	Alternate Function	PC connection through USB cable, but without using printf() functions within Milestone 4 and the final version of the project.