# Practical assignment of Digital Systems and Microprocessors

## Course 2022-2023

## Practice 2A

## LSShip

| | | |
|---|---|---|
| **Students** | **Login** | **Name** |
| | **pau.coderch** | **Pau Coderch Masallera** |
| | **joaquin.bressan** | **Joaquin Bressan Nazar** |

| | | | |
|---|---|---|---|
| **Delivery** | **Board** | **Report** | **Grade** |
| | | | |

| | |
|---|---|
| **Date** | **11/05/2023** |

*Copy for the students*

# Practical assignment of Digital Systems and Microprocessors

## Course 2022-2023

## Practice 2A

## LSShip

| Students | Login | Name |
|---|---|---|
| | pau.coderch | Pau Coderch Masallera |
| | joaquin.bressan | Joaquin Bressan Nazar |

| Delivery | Board | Report | Grade |
|---|---|---|---|
| | | | |

| | |
|---|---|
| Date | 11/05/2023 |

Cover of the report

## Summary of the statement

The system has four modes: Manual, Cruise, Record, and Autopilot. The mode is changed using push buttons: ManualMode and RecordMode. Pressing ManualMode button switches between manual and cruise mode, and pressing RecordMode button activates recording mode in manual mode.

The system records speed, direction, and delay between samples when Save button is pressed in record mode. Only the last recorded route is stored, and pressing RecordMode button again replaces the previous route. To replay a saved route, RecordMode button is pressed for 1 second.

The system indicates the current mode using an RGB LED. If no instructions are received for a minute, the engine alarm LED blinks, and it takes priority over reverse gear LED. Joystick is used to control speed and direction, with Y axis controlling speed and X axis controlling direction. Different LED arrays and motors are used to display speed and direction, and they update in real-time based on joystick position.

## Software design

The software design for our practice follows a structured approach. We start by initialising all the variables that we will be using, followed by configuring the HIGH_RSI, Interrupts, oscillator, and ports.

Next, we have the MAIN method, which initialises all the necessary configurations and sets the program to start in the Manual Mode. Inside the Manual Mode method, which is like the "Main Loop" where we manage all the modes, in addition to the functionalities of the manual mode.

Each mode has its own method, where we implement the necessary functionalities for that specific mode. We also have other methods that may be called within some of the modes, such as a Debouncer to manage bounces, which can be invoked as needed.

Additionally, we have functions for joystick intervals, loops to manage the DC motor and Servo Motor, and other methods as required for our practice.

In summary, our software design consists of a main structure managed within the manual loop, with multiple other methods called from the four mode methods to perform various functionalities.

## Microcontroller configurations

**Oscilator:**

In order to configure the oscillator we set the OSCCON register to a frequency of 8MHz, but as we enable the PLL in the OSCTUNE register, we have a total frequency of 32MHz. We also set the T0CON register to configure Timer0 as a 16-bit timer with no prescaler.

**Ports:**

- **ADCON0** → Is where we set the analog to digital converter (ADC). We enable the converter module by setting the bit 0 to 1.
- **ADCON1** → Is where we configure the voltage references, and also configure the pins AN0 and AN1 as analog inputs.
- **ADCON2** → Is where we configure the conversion clock for the ADC module. We set it to FOSC/8 and we also set an acquisition time of 2TAD.
- **TRISA** → Is used to configure the direction of the I/O pins in Port A. We set the 0th and 1st bits to 1 to configure the RA0 and RA1 pins as inputs for the joystick.
- **TRISB** → Is used to configure the direction of the I/O pins in Port B. We set the 0th, 1st and 2nd bits to 1 to configure the RB0, RB1 and RB2 pins as inputs for the manual and record button.
- **TRISC** → Is used to configure the direction of the I/O pins in Port C. We set the the 0th, 1st, 2nd, 3rd, 4th, 5th and 6th bits to 0. This is to configure the RC0, RC1, RC2, RC3, RC4, RC5 and RC6 pins as outputs for the LEDS and RGB LED. We set them to 0, because they work in negative logic.
- **TRISD** → Is used to configure the direction of the I/O pins in Port D.We set the 7th bit to 0 which basically configures RD7 pin as an output for one of the LEDS.

**Interrupts:**

In our case, we only used interrupts for the timer, so the only configuration that was necessary was for the timer.

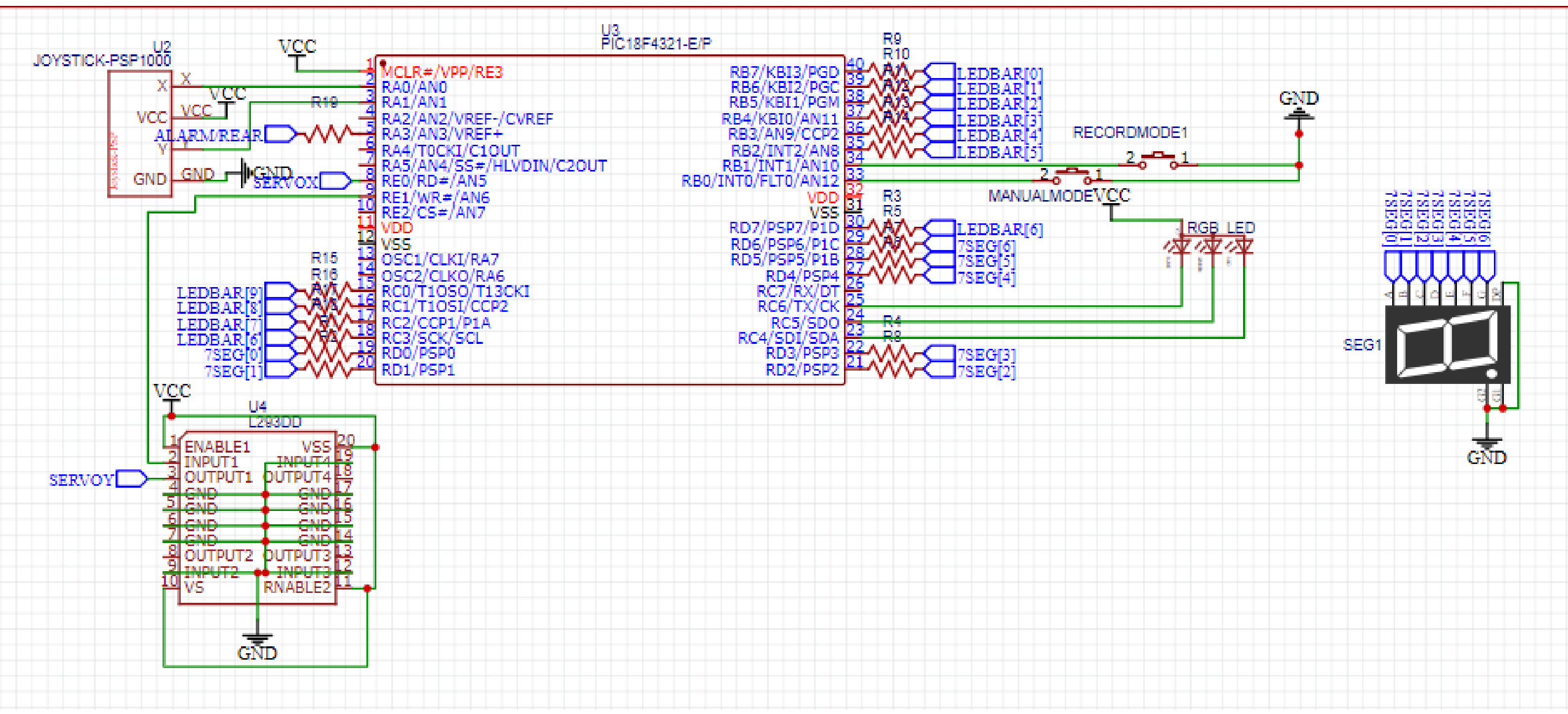
Here we configure the **RCON** register, which is used to configure general interrupt settings. We set it to 0, as we don’t want to have any priorities.

We also configure the **INTCON** register, so that enables us to enable or disable global interrupts (GIE) and peripheral interrupts (PEIE). In our case, we disable both global and peripheral interrupts by setting these bits to 0, as we don't want any interrupts to occur initially.

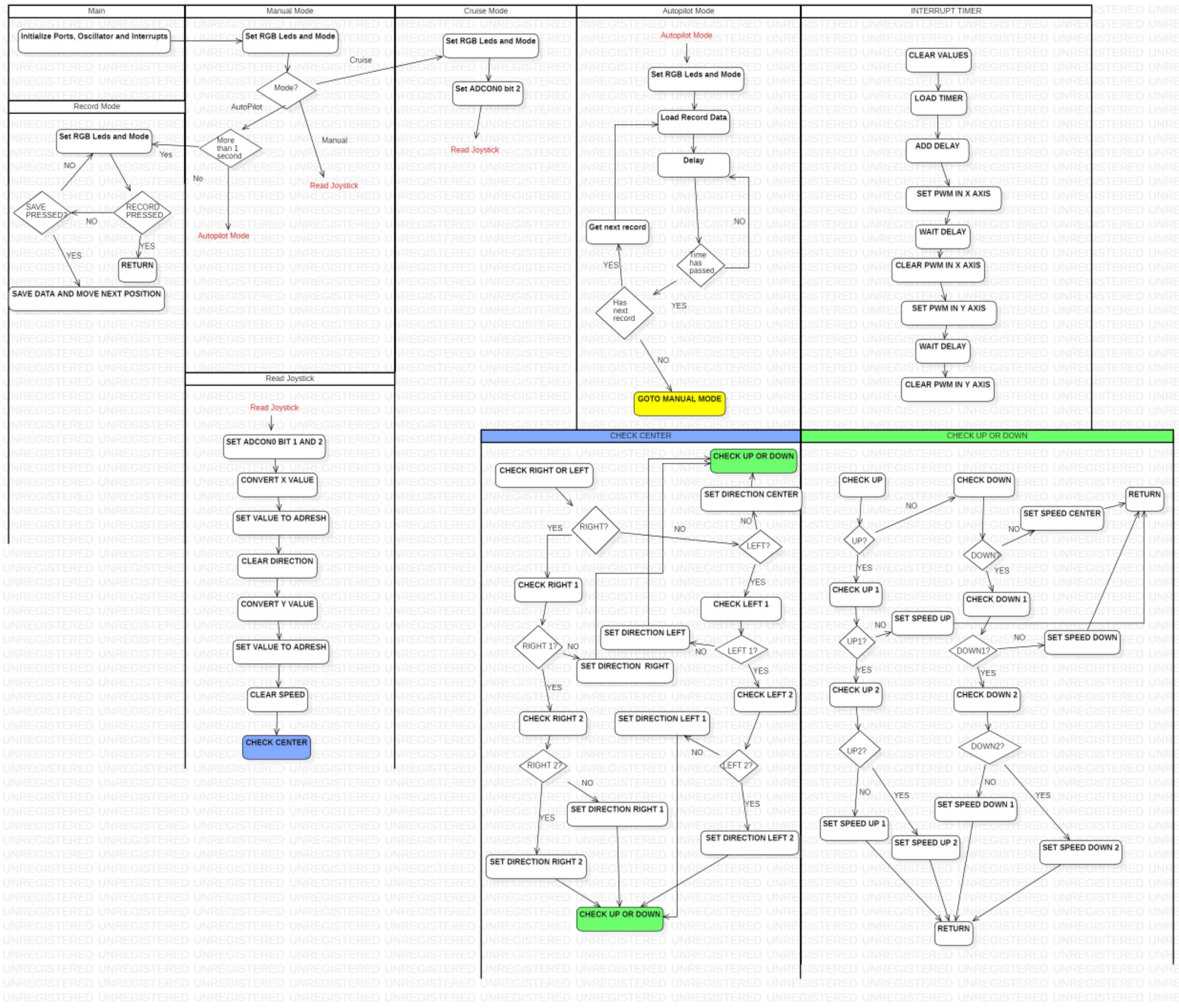
Next, we configure the **PIE1** register, which is used to enable or disable specific peripheral interrupts. We set the TMR1IE bit to 1, which enables the Timer 1 interrupt. This allows the microcontroller to respond to interrupts generated by the Timer 1 module.

Finally, we configure the **IPR1** register, which determines the priority levels of the peripheral interrupts. In our code, we set the TMR1IP bit to 0, which sets the priority level of the Timer 1 interrupt to low priority. This means that other interrupts with higher priority will take precedence over the Timer 1 interrupt if they occur simultaneously.

## Electrical Schematic



## Software activity diagram



## Observed Issues and Conclusions

During our practice, we encountered several issues that impacted our progress and added unnecessary stress to the project. One of the most significant problems was a poorly organized log of tasks. As a result, we found ourselves scrambling to complete the practice and submit it on time for the second phase of the project, despite having more than a month to work on it. This lack of proper task management and planning resulted in a last-minute rush and increased our workload over the past two weeks.

Another challenge we faced was the use of assembly language for programming, which was unfamiliar to us. Initially, we found it difficult to understand and write code in this language, leading to delays and confusion in the implementation process. Additionally, we struggled with configuring the ports, although we later realized that it was a relatively straightforward process. However, this initial setback cost us valuable time at the beginning of the project.

One major hurdle we encountered was with the programming of the joystick. Despite spending over four days attempting to configure it and make it work, we were unable to resolve the issues. Fortunately, we sought help from an intern who suggested an alternative solution of using left justified programming instead of right justified. This suggestion proved to be effective, and we were able to fix our errors and complete the functionality successfully.

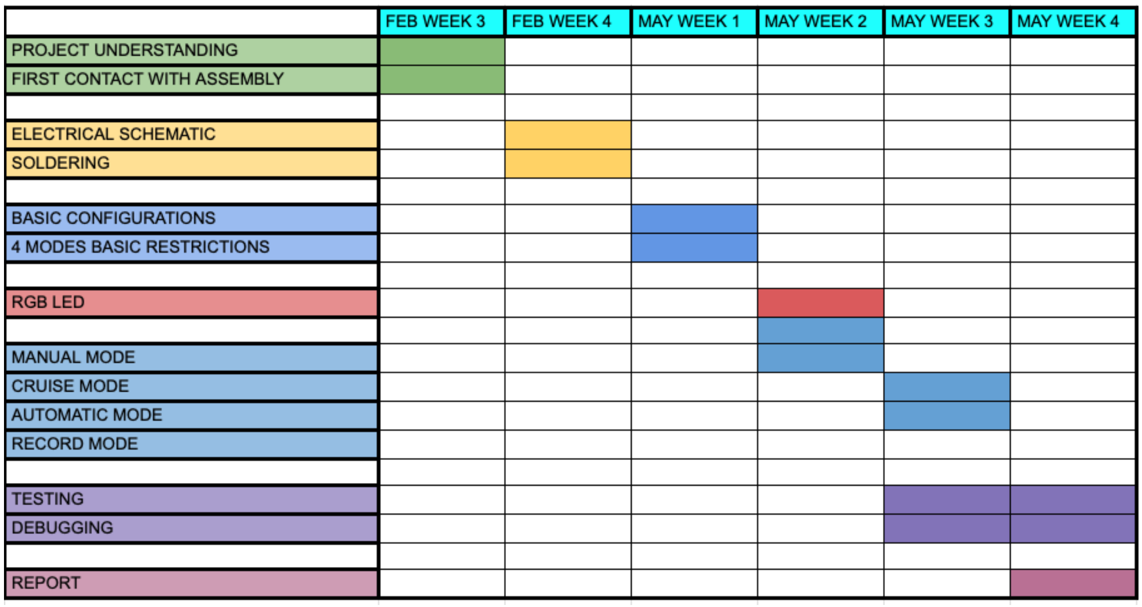
Furthermore, understanding the concepts of PWM and Interrupts posed challenges for us. It took considerable effort and time to grasp the configurations and the process that needed to be followed. However, eventually, we were able to comprehend these concepts and implement them in our project.

In conclusion, we acknowledge that the difficulties we encountered during this practice were largely due to our own shortcomings in organizing ourselves and managing our tasks effectively. However, despite the challenges, we found the project interesting and satisfying upon completion. Nevertheless, the process was frustrating at times, as we were working with an unfamiliar programming language and faced difficulties in identifying and resolving errors. Moving forward, we recognize the importance of better planning and organization to achieve our objectives more efficiently in future projects.

## Planning

Here is our comprehensive analysis of the planning and performance of our practice, including the initial plan formulated and the actual plan executed.

### Initial Planning



| | FEB WEEK 3 | FEB WEEK 4 | MAY WEEK 1 | MAY WEEK 2 | MAY WEEK 3 | MAY WEEK 4 |
|---|---|---|---|---|---|---|
| PROJECT UNDERSTANDING | 🟩 | | | | | |
| FIRST CONTACT WITH ASSEMBLY | 🟩 | | | | | |
| | | | | | | |
| ELECTRICAL SCHEMATIC | | 🟨 | | | | |
| SOLDERING | | 🟨 | | | | |
| | | | | | | |
| BASIC CONFIGURATIONS | | | 🟦 | | | |
| 4 MODES BASIC RESTRICTIONS | | | 🟦 | | | |
| | | | | | | |
| RGB LED | | | | 🟥 | | |
| | | | | 🟦 | | |
| MANUAL MODE | | | | 🟦 | | |
| CRUISE MODE | | | | | 🟦 | |
| AUTOMATIC MODE | | | | | 🟦 | |
| RECORD MODE | | | | | | |
| | | | | | | |
| TESTING | | | | | 🟪 | 🟪 |
| DEBUGGING | | | | | 🟪 | 🟪 |
| | | | | | | |
| REPORT | | | | | | 🟪 |

This was the plan we initially intended to follow in order to successfully complete the practice. It was designed to ensure that we wouldn't have to allocate too much time per week to the project, and would have ample time to overcome any obstacles that mi ght arise along the way, without having to rush things at the last minute.

We divided the tasks into two phases: first, understanding what we needed to do, and then applying it in practice until the completion of the practice itself. This approach allowed us to have a clear understanding of the project requirements and avoid unnecessary delays or complications.

## Log of tasks

| | MAY WEEK 3 | MAY WEEK 4 | APR WEEK 1 | APR WEEK 2 | APR WEEK 3 | APR WEEK 4 |
|---|---|---|---|---|---|---|
| PROJECT UNDERSTANDING | | | | 🟩 | | |
| FIRST CONTACT WITH ASSEMBLY | | | | 🟩 | | |
| | | | | | 🟨 | |
| ELECTRICAL SCHEMATIC | | | | | 🟨 | |
| SOLDERING | | | | | | |
| | | | | | | |
| BASIC CONFIGURATIONS | | | | | 🟦 | |
| 4 MODES BASIC RESTRICTIONS | | | | | 🟦 | |
| | | | | | | |
| RGB LED | | | | | | 🟥 |
| | | | | | | 🟦 |
| MANUAL MODE | | | | | | 🟦 |
| CRUISE MODE | | | | | | 🟦 |
| AUTOMATIC MODE | | | | | | 🟦 |
| RECORD MODE | | | | | | |
| | | | | | | |
| TESTING | | | | | | 🟪 |
| DEBUGGING | | | | | | 🟪 |
| | | | | | | |
| REPORT | | | | | | 🟪 |

This is the actual plan. This is what happens when you procrastinate and leave everything until the last minute, resulting in a rushed effort to complete everything on time.

Even though we initially had ample time to complete this practice, as we were also submitting to 5 other projects, we felt it was necessary to rush this practice in order to catch up with the second phase and meet the submission deadline for that one.