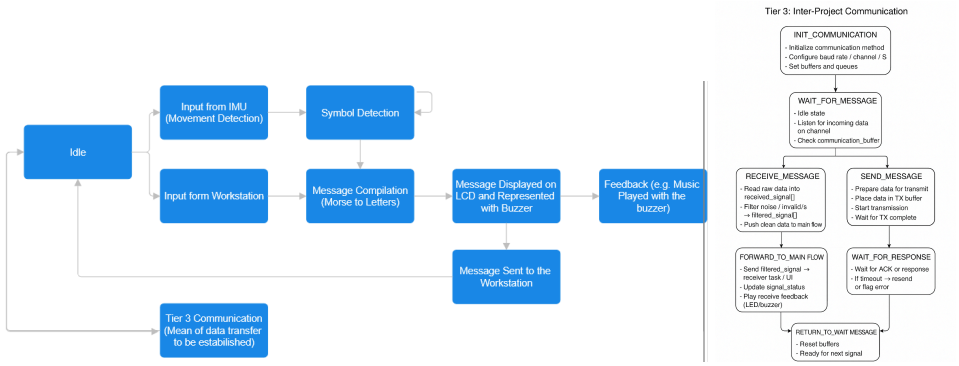
Bach Nguyen, Pedro Setti, Piero Cianciotta

1. **A brief description of the functionalities in the program, in order of Priority**
   Tier 1 Basic Functionalities: Generating Morse symbols, Recognizing at least two positions, Sending symbols to the workstation;
   Tier 2 Minimum Requirements: One or more state machines, Receiving messages from workstation, Feedback when the message is sent successfully;
   Tier 2 Additional Functionalities: IMU and data collection, User interface, Music played in menus, Translation of Symbols to letters;
   Tier 3 Inter-Project Communication, Communication channel to be decided;

2. **State machine that implements the project. It is easiest to present this as a diagram.***
   The state machine should cover the main tasks of the program; lower levels of abstraction are not necessary.



3. **Define the number of tasks and the role of each one.**
   Input from IMU and Symbol Detection
   Task 0.0: Detect movement
   Task 0.1: Classify each movement as . or -
   Task 0.2: Await three consecutive spaces
   Message Compilation
   Task 1.0: Create an array of characters divided by spaces
   Task 1.1: Translate each character into a letter
   Interface
   Task 2.0: Display message on screen
   Task 2.1: Play Sound with Buzzer
   Task 2.2: Play Music after displaying
   Task 2.3: Send message to the workstation
   Tier 3 Communication
   Task 3.0: Await signals
   Task 3.1: Filter signals
   Task 3.2: Send signals

4. **Outline of the peripherals to be used (buttons, sensors, etc.) and their purpose in the program.**
   Button;
   IMU Sensor;
   Lcd Display;
   Speaker;
   Led;

5. **Identify global data structures and variables.**
   IMU and Symbol Detection
   Variables:

? imu_data → input from IMU (e.g., acceleration, gyroscope vectors)
bool movement_detected → running permanently
char current_symbol → stores symbol obtained by input
uint8_t space_counter → counts consecutive spaces (terminate reading if 3)

Data Structures:

uint8_t[] symbol_buffer → list of symbols (['.', '-'])

Message Compilation

Variables:

char current_character → current Morse code character
char[] current_message → String showing the full message

Data Structures:

morse_to_letter_map() → dictionary table mapping Morse symbols to letters (TODO: find C equivalent for dictionaries)
char[] message_buffer → list of characters collected from symbol decoding

Interface / Output

Variables:

char[] display_message → String to show on display
bool buzzer_status → Buzzer on/off
bool music_status → Music on/off (to play after receiving/sending/displaying messages)

Tier 3 - Communication

Variables:

? received_signal → stores signal/message received from other projects
? filtered_signal → stores signal after filtering noise or invalid data
char[] send_queue → stores outgoing messages/signals to transmit

Data Structures:

communication_buffer → circular buffer or queue to temporarily hold signals for processing
signal_status_map → dictionary storing status flags for incoming/outgoing messages

6. **Preliminary schedule, including milestones and distribution of work.**

Week 0_ 13.10 - 26.10 Project Plan

- Milestones: Description, state machine implementation, Number of tasks - roles, List of peripherals, Global data structures and variables, Tier definition.
- Distribution of work:
  1. Bach: Outlining Tasks, Preliminary schedule outline (19/10 DONE)
  2. Piero: Outlining Tasks and Variables, State Machine
  3. Pedro: Outlining Variables, State Machine

Week 1_ 27.10 - 02.11 Tier 1

- Milestones: TBD
- Distribution of work:
  1. Bach:
  2. Piero:
  3. Pedro:

Week 2_ 03.11 - 09.11 Tier 2 Minimum Requirements

- Milestones: TBD
- Distribution of work:
  1. Bach:
  2. Piero:
  3. Pedro:

Week 3_ 10.11 - 16.11 Tier 2 Additional Functionalities and Tier 3, In parallel

- Milestones: TBD
- Distribution:
    1. Bach:
    2. Piero:
    3. Pedro:

7. **Define the tier you are aiming to. You can change your opinion later.**
   Tier 3, Grade 5