

**Modules**

The key of designing complex system is dividing the operations into submodules. This division simplifies the design process and makes it easier to work on the project as a team. Hence in this project we divided the project into 8 clearly defined submodules. These submodules will be examined briefly in the next part.

**Main Function**

The main function is the core of process. All the submodules and operations are controlled in this block. In the game after the victory or lose condition the game starts over, hence our main function runs in an infinite loop. The operation of main block starts after the initialization stage. In initialization stage; ports PF0 and PF4, ADC0 and ADC1, Timers and Nokia screen are initialized made ready for operation. After this stage the main process starts:

Deployment Stage

1. Set operation to ship placement
2. Determine ship cursor location using Cursor Read Submodule
3. Send ship cursor data to Screen Driver Submodule
4. Control Button status using Button Read Submodule
5. Return to 2nd step if no button is pressed
6. Save the ship type and location to Ship Status Submodule
7. Return to 2nd step for next ship input until all ships are placed
8. After all ships are placed send end of operation signal to Screen Driver submodule to show ship locations
9. Control Button Read Submodule for any user input
10. Wait until any user input
11. If any input received send clear screen signal to Screen Driver Submodule

Attack Stage

1. Control Button Read Submodule for any user input
2. Wait until any user input
3. After input received activate sneak peek mode for screen and activate 0.5 sec timer
4. Set operation to Mine Placement
5. Determine mine cursor location using Cursor Read Submodule
6. Send mine cursor data to Screen Driver submodule
7. Control Button status using Button Read Submodule
8. Return to 16th step if no button is pressed
9. Send the mine location to Ship Status Submodule
10. Return to 16th step for next mine input until mines are placed
11. After all ships are placed send end of operation signal to Ship Status Submodule and Screen Driver Submodule
12. Take end of game results from Ship Status Submodule
13. Send the end of game results to Screen Driver Submodule
14. Wait for any button input from Button Read Submodule
15. Wait for new game input

**Cursor Read Submodule**

The cursor read submodule reads the analog value from the potentiometers and transform this analog value to respected pixel value inside the current operation limits. This submodule takes operation mode (Ship placement or Mine placement) from the main function and maps the analog value in this range. This module has 2 analog input one is vertical input and horizonal input. This module reads data once since we use polling in the main function and called in every polling.

**Button Read Submodule**

This submodule is responsible for reading the status of button and informing the main function if any of the button pressed. This submodule reads the button data of Button 1 from PF0 and Button 2 from PF4. This module reads data once since we use polling in the main function and call this function only when it is needed.

**Ship Status Submodule**

This submodule has two different responsibility in two stages of game respectively. In the Deployment Stage, this submodule takes the coordinates and types of the ships from the main function. Then controls the validity of the ship data (is there any battleship). If the data is valid, then send the ships location to Location Submodule.

At the Attack Stage, this submodule takes mine location data from the main function and compares with ship location data. If any ship hit, sends ship hit data to location submodule. If the player hit a civilian ship or miss any of a battleship, send Loss end game result to the Main function and activate loss message in Location submodule. If all the battleships are hit and no civilian ship is harmed, send Victory end game result to the Main function the activate Victorious message in Location Submodule.

**Location Submodule**

When using the Nokia screen, it is hard and illogical to keep all the pixel map directly in the main function. For proper operation of a SPI connected screen, we need to send serial pixel data with exact periods. Since a pixel map hold lots of static information and modification of this data will took uncertain time, we need to define this map in a different place. The Location Submodule adjust the pixel map data and modifies it with the incoming inputs from Ship Status Submodule. Hence, the Screen Driver submodule only reads related pixel data from the memory.

**Screen Driver Submodule**

Using 5110 LCD Screen is one of the most challenging part of this project. To perform this operation this submodule must send the respected control signals to the screen and serial pixel data. For this serial pixel data transfer, we need to use SPI module of the Tiva Board. Hence, this submodule is responsible for the SPI usage with the incoming pixel data from the Location Submodule, status of Data/Command pin.

**0.5 sec Timer Submodule**

This timer is responsible for the sneak peak of the ship configuration to the Player 2. When the player 2 presses any button at the start of Attack Stage, this timer activates for a one-shot mode. At the end of timer this submodule sends multiple signals. First signal is clear screen signal sent to location submodule, second signal is start of mine placement signal for main function and third signal is activation of 20 sec counter submodule.

**20 sec Counter Submodule**

This submodule is responsible for the adjustment of the play time of the Attack stage. In the attack stage player 2 have 20 seconds play time prior to sneak peak of ship placement. During the game this submodule sends remaining time to the Screen Driver Submodule with 1 second resolution. Hence in this submodule the game time is hold by configuration of 1 second timers which run 20 times. Timer module gave interrupt in every second and the new time is sent to Screen Driver Submodule. At the end of 20th run when the game time ends, the end of game signal is sent to main function to finish the game.