**Contents**

1. [Overview 2](#_TOC_250020)
   1. [Usage 2](#_TOC_250019)
   2. [Program Flow Chart 3](#_TOC_250018)
2. [Subroutines 4](#_TOC_250017)
   1. [Button Interrupt 4](#_TOC_250016)
      1. [Button Interrupt Flow Chart 4](#_TOC_250015)
   2. [UART0 Interrupt 5](#_TOC_250014)
      1. [UART0 Interrupt Flow Chart 5](#_TOC_250013)

* 1. [Timer Interrupt 6](#_TOC_250016)
     1. [Timer Interrupt Flow Chart 6](#_TOC_250015)
  2. [Move Action 7](#_TOC_250014)

2.4.1 [Move Action Flow Chart 7](#_TOC_250013)

# Overview

This program gives the user different actions and rules to simulate a game called Wee Dig Dug in a box 19 x15 character in Putty

1. Fan Yang wrote the subroutines Illuminate box making,enemy module and lab7\_documentation.

2. Kai Zheng wrote the subroutines Timer\_interrupt, player module, hardward utilized module and score module.

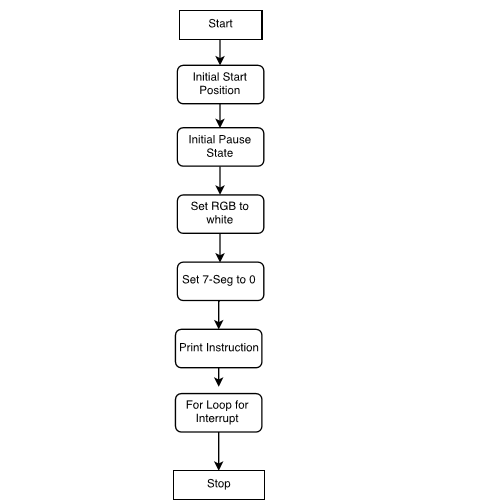
## Usage

The C wrapper calls the subroutine lab7,check Button Interrupt ,UART0 Interrupt,Timer0 Interrupt, and all Clear Interrupts to execute the game. The Button Interrupt subroutine is to promote the push button to halt/restart the game. The subroutine Timer0 Interrupt is to generate the timer to control the game speed , refresh the display, control time limit for the entire game, generate random position and random motion for enemies.The UART0 subroutine is to trigger a list of different keystroke commands for users to perform. The letter u,d,r,and l control the player to change the direction up, down, right and left corresponding. The key Spacebar is to launch the air pump. Exiting the program by entering ‘q’, or continuing the game by hitting enter key.The uart0 inteerupt also illuminate LED for the number of lives remaining, illuminate the RGB LED to display the status of the game, and illuminate the seven-segment display to show the current level he suer is on. All clear interrupt subroutine clears interrupted executed. The program runs the appropriate subroutines to perform the action then loops back to the beginning of the program so the user can perform another action.

## Program Flow Chart

**\**

no

]

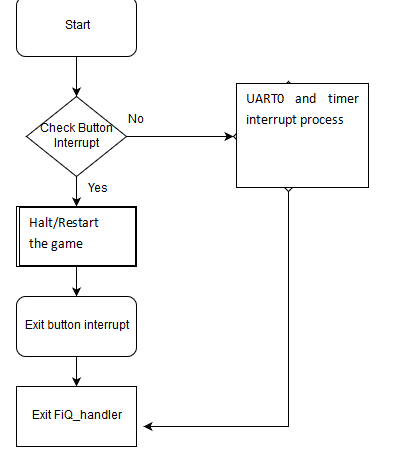
?

# Subroutines

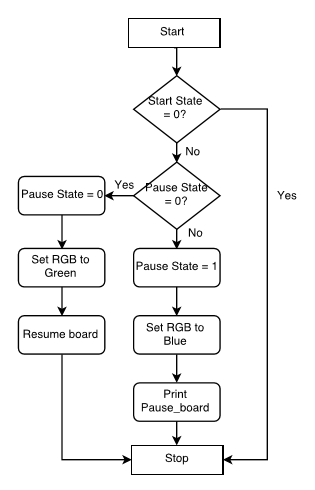
## Button Interrupt

This subroutine allows the user to push the interrupt button to halve/restart the game. It triggers the the RGB LED to display the status of the game. The RGB is green when the game begins, it is blue when the game is paused, and it is purple when the game ends. When Button state is 0, it is off, and keyboard input is ignored. When button is 1, it is on, and the previous state is displayed.

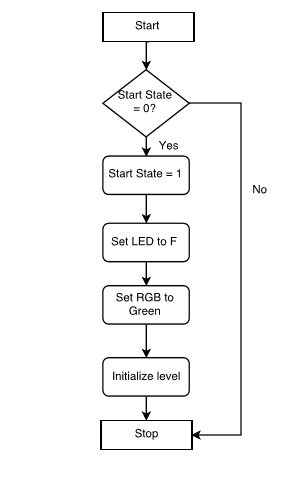
### 2.1.1 Button Interrupt Flow Chart



### 2.1.2 Pause game Flow Chart



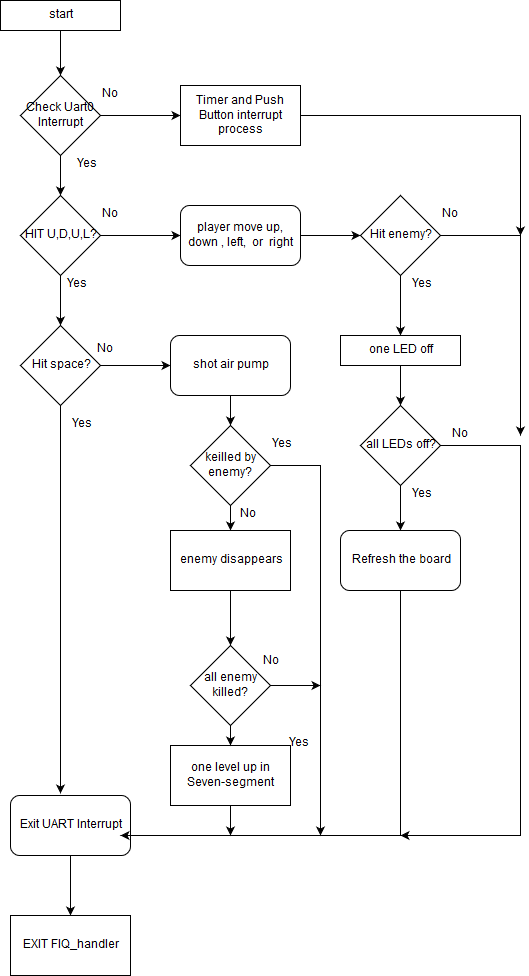
### 2.1.3 Start game Flow Chart



## UART0 Interrupt

This subroutine gives the user a list of the keyboard commands to control the game stare.The letter u,d,r,and l control the player to change the direction up, down, right and left corresponding. The key spacebar is to launch the air pump.The ‘Q’ exit the program. All other keyboard inputs are ignored. Four LEDs illuminate the number of lives remaining. The user starts the game with four lives(all LEDs on). WhenEach time a life is lost, an LED is turned off. The RGB blinks red when the the shot is fired in the game. The seven-segment display shows the current level the user is on. The value is incremented by one with each completed level.

### 2.2.1 UART0 Interrupt Flow Chart

****

## TIMER0 Interrupt

This subroutine generates random number to promote a list of random events such as the random position, the random direction, and the random characters.

### 2.3.1 TIMER0 Interrupt Flow Chart

**Untitled Diagram3**

## MOVING ACTION

This subroutine gives the user a list of the keyboard commands to control the character stare.The direction is changed when user hits the u,d,r,or l (for up, down,right or left). Every time the character encounters the boundary of the box, it should bounce off the wall and start moving in he opposite direction. A count of cumber of walls encountered increase by 1 when character hit wall once.

### 2.4.1 moving action Flow Chart

