



**Computer Graphics and Animation Project Report**  
**“Programming Assignment 17 - Bubble Crab Sprite Animation”**

Group Members:

Luqmanul Hakim (001201800095)

Michael (001201800020)

Yodi Fakhri Muhammad (001201800107)

CIT-4

President University

Semester 5

20192

## **PREFACE**

This project report entitled “Bubble Crab Sprite Animation”, is submitted to fulfill one of the assessment in accomplishing Computer Graphics and Animation subject final project at the department of Computing, President University. As the writer, we would like to express our sincere gratitude and respect to our lecturer who already gave the feedbacks, suggestions, and guidance in making the project, Mr. Eddo Fajar Nugroho, S.T., M.T so that made us succeed to accomplish this project. Hopefully, this project would give a positive contribution to the educational development or those who want to carry out further research about sprite animation.

# CHAPTER I

## BACKGROUND

### 1.1. Introduction

#### 1.1.1. The Program

A program is about sprite animation, a basic application program that allows user to play/control Bubble Crab (one of the boss in Mega Man X2). This program is like a simple animation game that provides several movements/actions (introduction, stand, jump, attack and move diagonally) for the user to demonstrate or play. This program manipulates the backgrounds and characters or in this case are sprites to create a whole series of simple graphic game.

#### 1.1.2. The Program Language

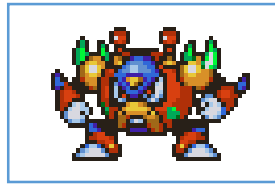
Programming language that used to make this application is a third-generation event-driven programming language from Microsoft for its Component Object Model (COM) programming model. Visual Basic was derived from BASIC and enables the rapid application development (RAD) of graphical user interface (GUI) applications, access to databases using Data Access Objects, Remote Data Objects, or ActiveX Data Objects, and creation of ActiveX controls and objects. There are several reason for this programming language to be implemented in this program, which are commonly used, easy to use, open source of reference, and the language is friendly.

### 1.2. Basic Theory

#### 1.2.1. Sprite Animation

##### Sprite

A *sprite* is a bitmap graphic that is designed to be part of a larger scene or in other words sprite is a single graphic image that is incorporated to a larger scene so that it appears to be part of the scene. It can either be a static image or an animated graphic. Examples of sprites include objects in 2D video games, icons that are part of an application user interface, and small images published on websites.



*A Sprite*

Developers referenced these sprites in the source code and assigned properties such as when the sprites were displayed and how they interacted with other sprites. For example, in a side-scroller, such as Super Mario Bros, the sprite of an enemy Koopa would turn into a flattened Koopa when Super Mario jumped on it. Today, some video games still use 2D sprites.

Sprites are still used by software developers for other purposes. For example, sprites are often used to add buttons, symbols, and other user interface elements to software programs. Developers can attach actions to sprites within the user interface, such as playing an animation or changing the current view of the window when the sprite is clicked. Sprites are especially useful for adding custom graphics that are not natively supported by the operating system's API.

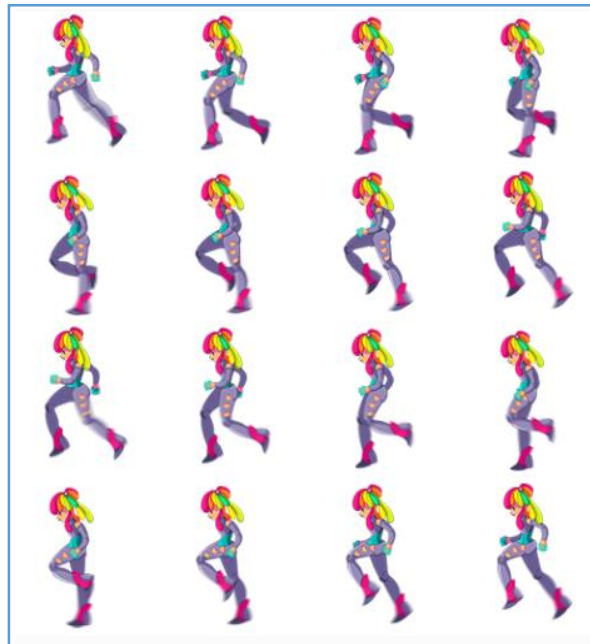
Sprites are also used on the Web for navigation buttons and for adding visual appeal to webpages. In recent years, *sprite sheets* have become a popular way for web developers to load website graphics. By combining a large number of sprites into a single image, all the sprites can be downloaded and cached by a user's browser with a single request to the server. The images are then displayed using CSS properties that define the locations of individual sprites within the image.

### Animation

*Animation* is the process of changing images in quick succession to give the illusion of movement or in other words animation is a method in which pictures are manipulated to appear as moving images. The images are put together one after another, and then played at a fast speed to give the illusion of movement.

## Sprite Sheet

Sprite sheets are used to speed up the process of displaying images to the screen; It is much faster to fetch one image and display only a part of that image than it is to fetch many images and display them. Sprite sheet animation is nothing more than taking a sprite sheet and changing which sprite is rendered in quick succession to give the illusion of movement, much like a film projector displaying a movie. When you put many sprites into a single image, you get a sprite sheet.










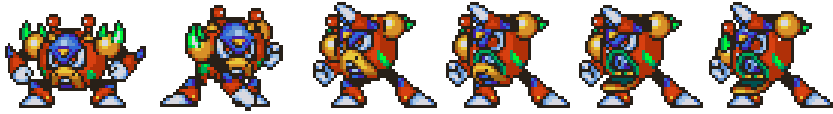



*A Sprite Sheet*

Sprite sheets are made up of two parts, frames and cycles. A frame is a single image (or sprite) from the sprite sheet. When the frames are put in an order that creates a continuous movement, it creates a cycle. There are three parts to coding a sprite sheet animation, creating the image, updating the image to each frame of the animation, and drawing the frame to the screen.

## Bubble Crab Animation Sprites

Series of sprites for every state of Bubble Crab's movement is shown below.

Actions Name	Animation Sprites
Introduction	
Stand	
Walk	
Jump Up	
Jump Down	
Bubble Shield	

Crab Claw	
Bubble Splash	
Bubble Robotic Crabs	
Fly Robotic Crabs	
Death	

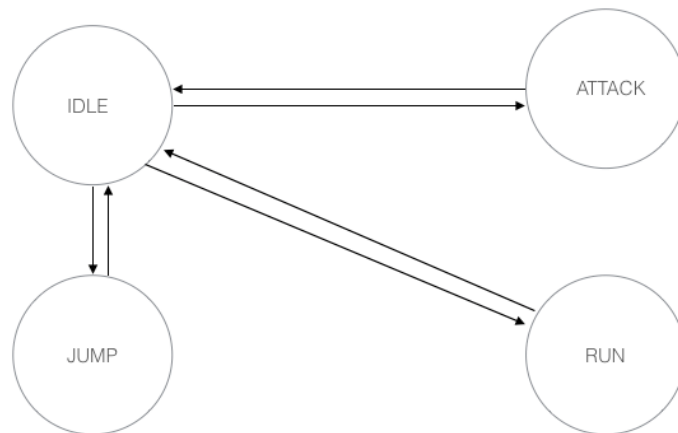
### 1.2.2. Sprite Mask

To put a sprite on a background, you will need to create a “mask” of the sprite. A mask is a black contour of the sprite on a white background. The sprite itself should be changed to have a black background. Sprite Masks are used to either hide or reveal parts of a Sprite or group of Sprites. The Sprite Mask only affects objects using the Sprite Renderer Component.

### 1.2.3. State Diagram and State Transition Table

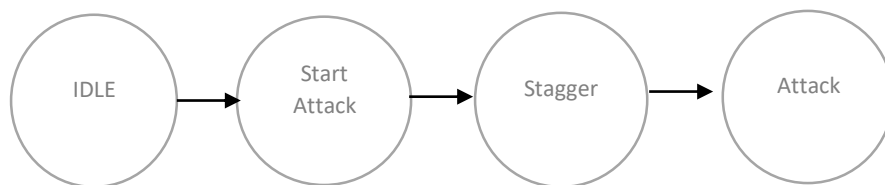
#### State Diagram

A character state diagram is a finite state machine to model transitions between character actions. The states indicate character actions, so then current states mean current actions. The animations are built based on the state diagram of related characters. The simple example of state diagram of a game is shown below.



#### In-between State

Even to make it more specific and smoother, there is specific states between the main states, those states are called “in-between states”. Example of in-between states are shown below.



#### State Transition Table

State Transition Table is a table that showing each possible transition from one state to another. This table shows the current state, next state, and also input or trigger.



#### 1.2.4. Collision Detection

Collision Detection is a mechanism to detect whether two objects are colliding each other. There are plenty of ways to do this, one of them is by making hit box. Collision Detection can also be created by declare a global variable with function as the collision point. If the first object had the same posX and posY with the second object/character, it assigns the posX and posY to the global variable and at the same time, the first object is collide. The second object will check whether the global variable had the same posX and posY, if it is true, the second object will be destroy/disappear/collide.

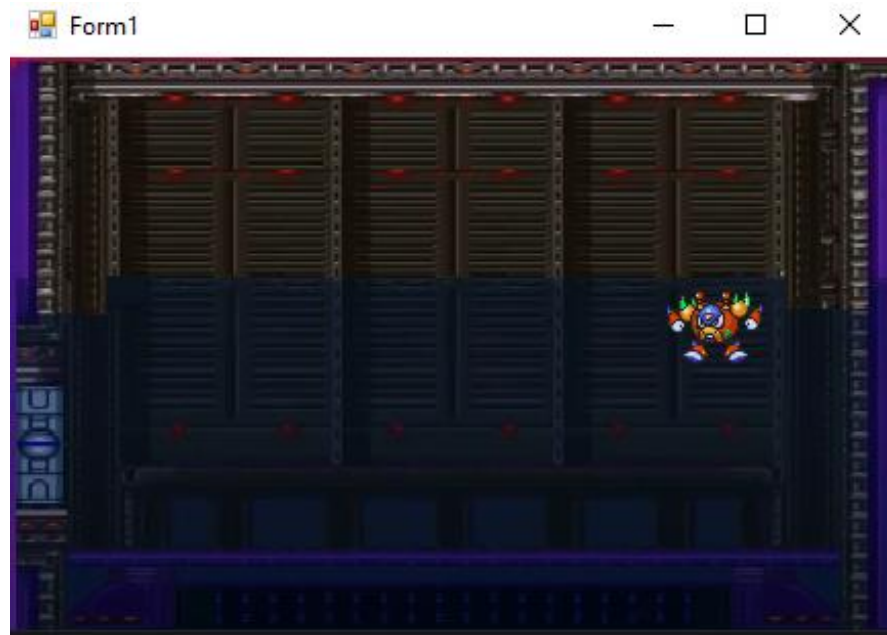
## CHAPTER II

### CONTENT

#### 2.1. Implementation

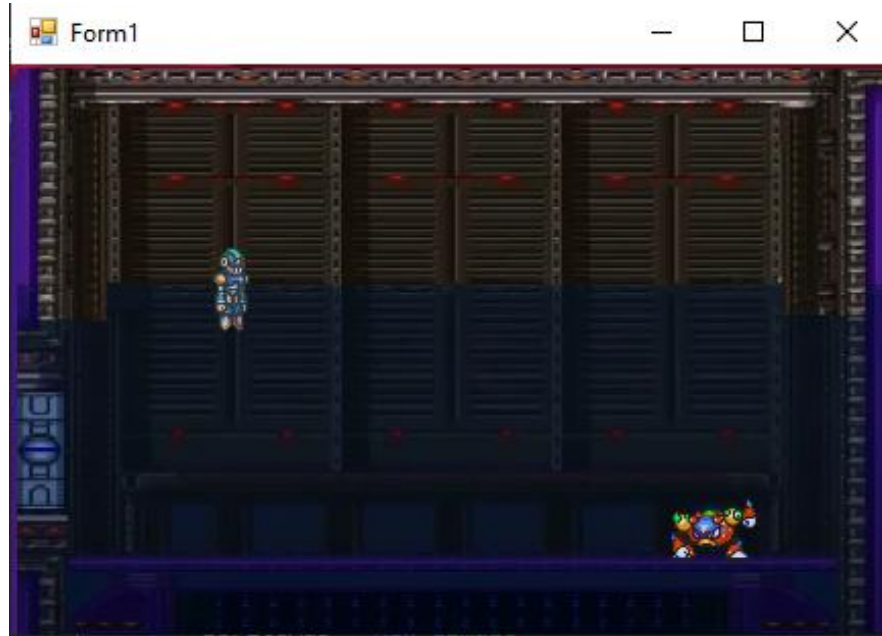
##### 2.1.1. Main Interface of the Program

The Interface when the program is running and doing all the movements (features) are shown in the picture below.



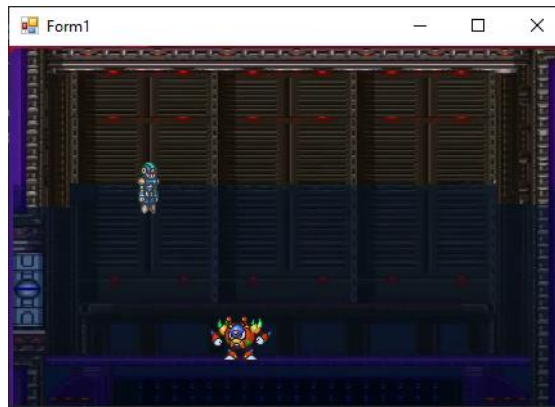
*Figure 1 (Come In)*

When the program just running, it started with the Bubble Crab comes down from right-upper side on the screen (figure 1).

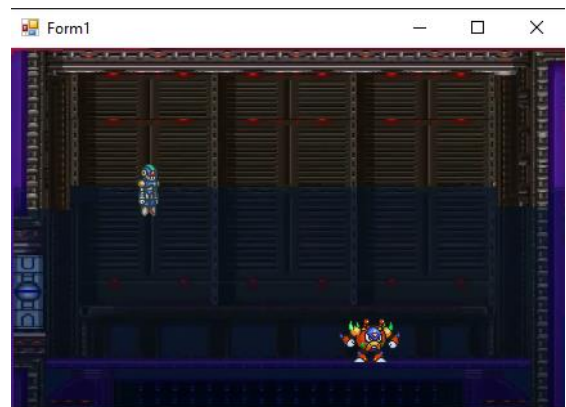


*Figure 2 (Introduction)*

After the Bubble Crab touch the ground, the Dummy Doll (Zombie) comes down from the left-upper side on the screen, the dummy stop at a fixed location floating on the water (half of the dummy is underwater). While the Dummy Doll shows up, Bubble Crab doing it introduction movements (figure 2).

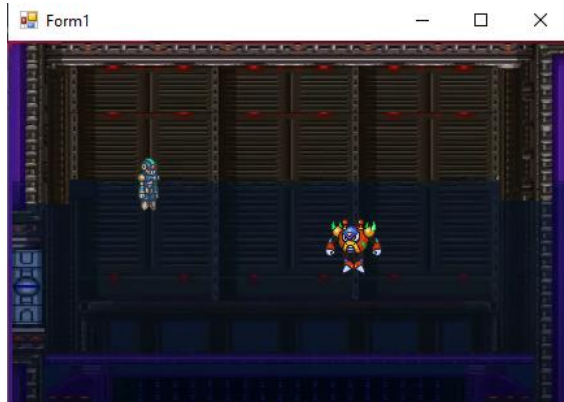


*Figure 3 (Move Left)*

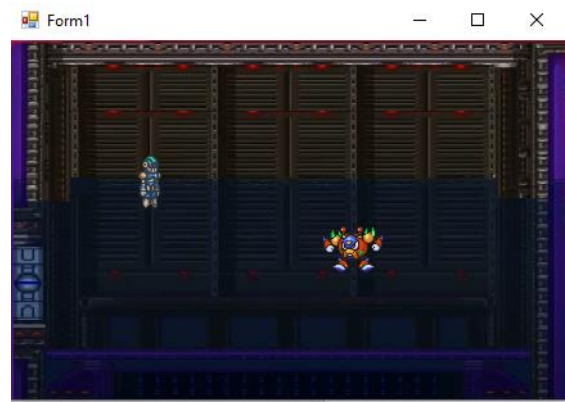


*Figure 4 (Move Right)*

Bubble Crab can also move to horizontally to left (figure 3) and right (figure 4) using arrows on the keyboard, the face directions when it moves depend on which directions the Bubble Crab moves. If Bubble Crab continuously move to left or right until it touch the wall, it will automatically stop and change position to stand.



*Figure 5 (Jump Up)*



*Figure 6 (Jump Down)*

Bubble Crab can jump vertically upwards as high as the water (figure 5) and when it touch the maximum height of jump, Bubble Crab will goes down to the bottom and back to stand position on the ground (figure 6)

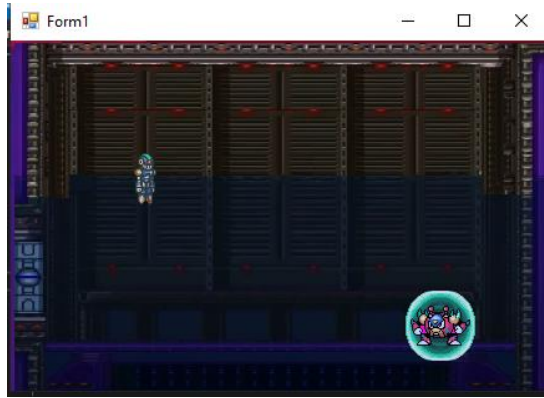


*Figure 7 (Leap Up)*

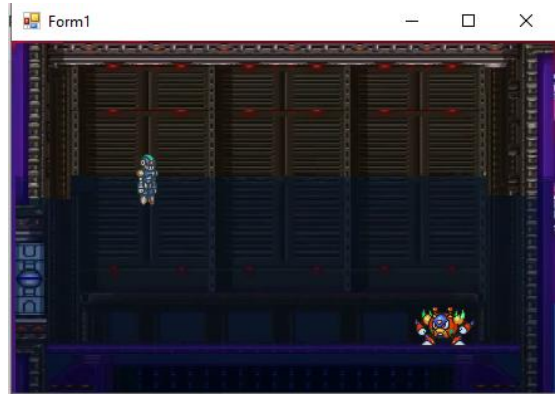


*Figure 8 (Leap Down)*

Bubble Crab can perform leap movement in the direction he is facing off. There are 2 leap movement, first when Bubble Crab start to leap till the maximum height, its body position will change to jump up position (figure 7), from the maximum height goes down to the ground, the body position will change to jump down position (figure 8).

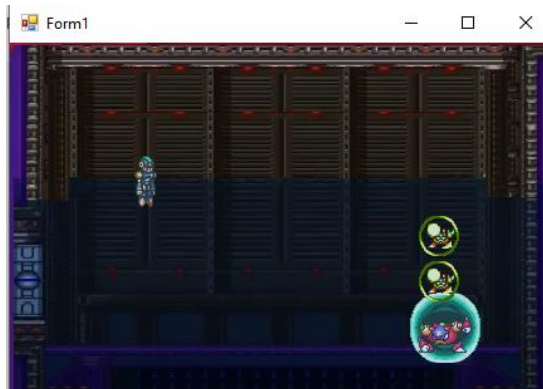


*Figure 9 (Bubble Shield)*

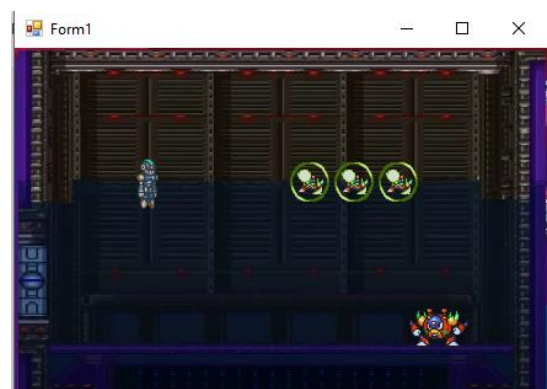


*Figure 10 (Shield Disappear)*

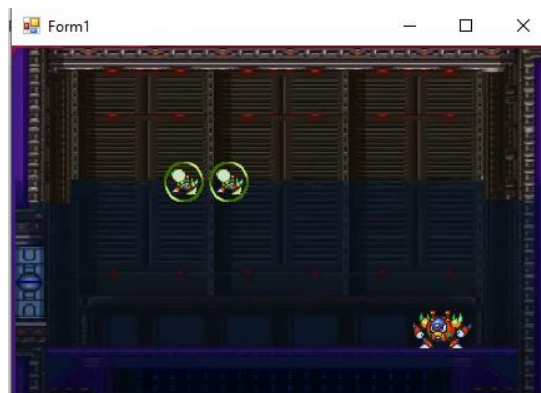
Bubble Crab can generate Bubble Shield in order to protect itself from other character. When Bubble Crab uses Bubble Shield, it will encase itself with bubble protection (figure 9). Bubble Shield just can exist in certain of time, so after several seconds, the Bubble Shield will automatically disappear (figure 10).



*Figure 11 (Robotic Crabs Shield)*



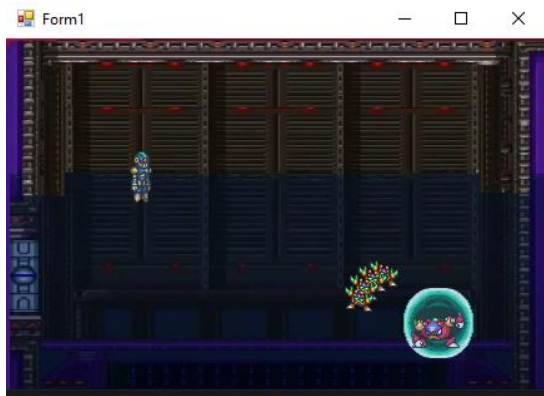
*Figure 12 (Crabs Floating on Water Surface)*



*Figure 13 (Crabs hit Dummy Doll)*



When Bubble Crab inside the Bubble Shield, it can release up to three Robotic Crabs simultaneously (Figure 11). The Robotic Crabs will go to the top of the water and floating horizontally forwards on the surface of the water (Figure 12). If the crabs hit the Dummy Doll, it will disappear along with the Dummy Doll, after few seconds, Dummy Doll will respawn and show up again (Figure 13).

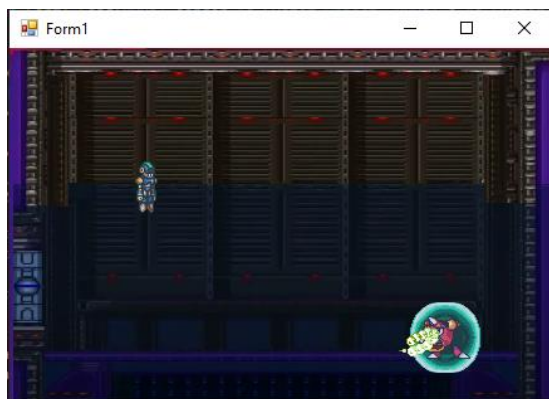


*Figure 14 (Fly Robotic Crabs)*

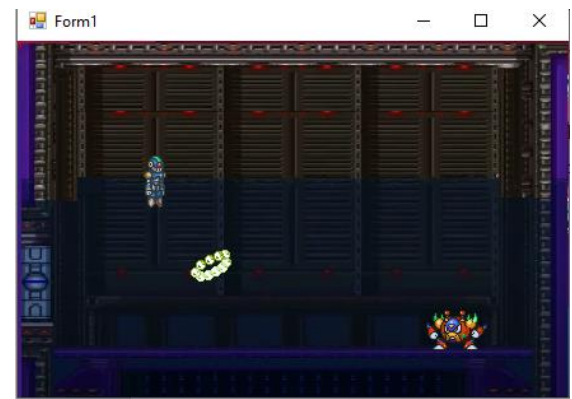


*Figure 15 (Robotic Crabs Fly in Random Directions)*

Bubble Crab can throw up to four robot crabs simultaneously in different directions (Figure 14) when it still inside its shield. The robot crabs can travel in linear path, when the robot crabs touch the edges of the screen (wall/ground/ceiling), it will bounce (Figure 15)



*Figure 16 (Bubble Splash inside Shield)*

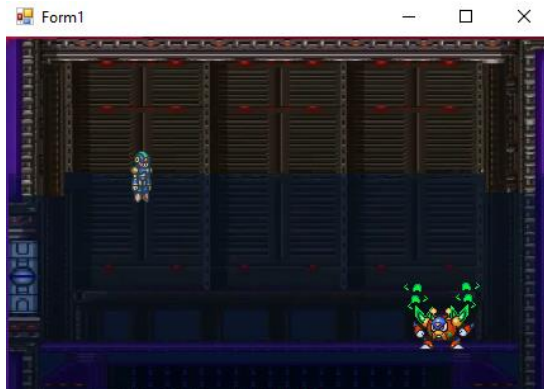


*Figure 17 (Bubble Ring Travels Diagonally Upwards)*

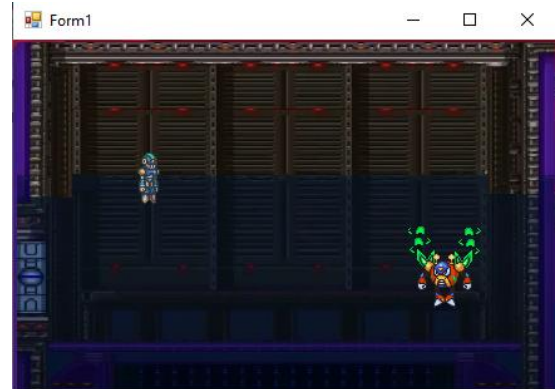


*Figure 18 (Bubble Splash)*

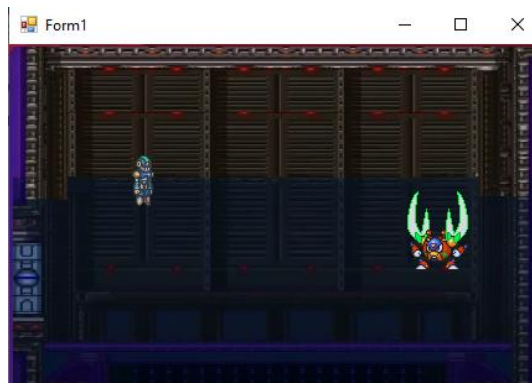
Bubble Crab can shoot as a ring of bubbles that travel in a parabolic path upwards (Figure 17). There are two types of Bubble Splash, Bubble Crab shoots Bubble splash when it inside the shield (Figure 16), and Bubble Crab can also shoot Bubble Splash when there is no shield that encased its body (Figure 18).



*Figure 19 (Crab Claws)*



*Figure 20 (Crab Claws Up)*

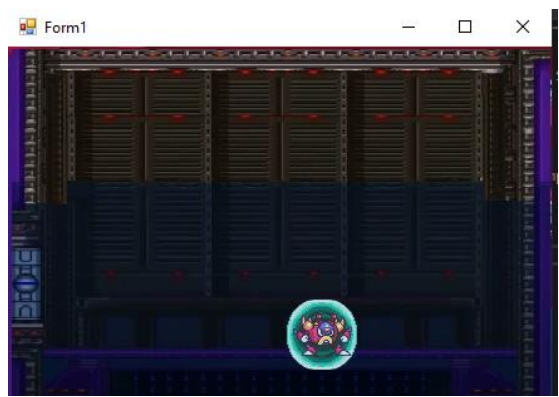


*Figure 21 (Crab Claws 18)*

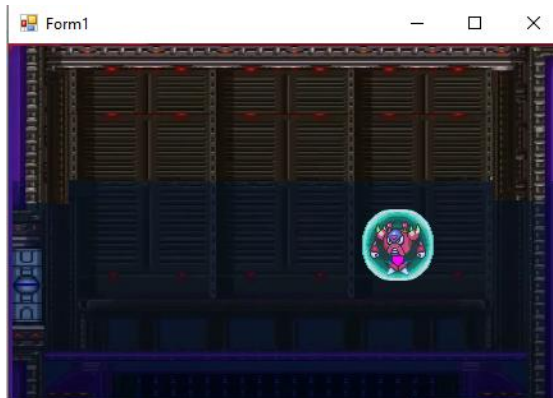
Bubble Crab can generate Crab Claws as one of its ability, it's started with Bubble Crab generates the energy in stand position (Figure 19). After the Claws already generated, Bubble Crab will fly/jump up with the claws on its shoulder as high as the water level (Figure 20). After it touch the top of the water, Bubble Crab goes back to the ground with its Claws (Figure 21).



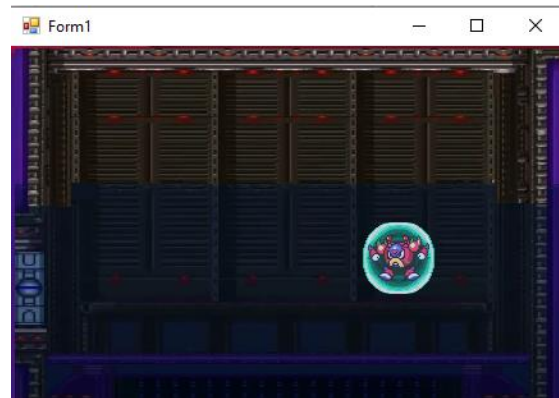
*Figure 22 (Move Left Inside Shield)*



*Figure 23 (Move Right Inside Shield)*



*Figure 24 (Jump Inside Bubble)*



*Figure 25 (Jump Down Inside Shield)*



*Figure 26 (Stand-  
Inside Shield)*



Those 5 figures are the interface of 5 basic movements inside Bubble Shield, which are Move Left (Figure 22), Mover Right (Figure 23), Jump (Figure 24), Jump Down (Figure 25), and Stand (Figure 26).

#### 2.1.2. Features of the Program

There are several features in this simple sprite animation game that using keyboard to activate/use it like shown in the table below.

Features	Function	Keys/Symbol (Keyboard)
Move to Left	Move Bubble Crab to the left position on the screen.	Left Arrow
Move to Right	Move Bubble Crab to the right position on the screen.	Right Arrow
Jump	Enables Bubble Crab to perform a vertical jump on the screen.	Up Arrow
Stand	Enables Bubble Crab to stop from other movement (walk) and directly change to stand position.	Down Arrow
Move to Left (Encased in Bubble Shield)	Move Bubble Crab to the left position on the screen when it inside the Bubble Shield.	A
Move to Right (Encased in Bubble Shield)	Move Bubble Crab to the right position on the screen when it inside the Bubble Shield.	D
Jump (Encased in Bubble Shield)	Enables Bubble Crab to perform a vertical jump on the screen when it inside the Bubble Shield.	W

Stand (Encased in Bubble Shield)	Enables Bubble Crab to stop from other movement (walk) and directly change to stand position.	S
Leap	Enables Bubble Crab to perform leap based on the direction it is facing off.	Space
Crab Claws	Enables Bubble Crab to generates Crab Claws energy to attack the other character.	1
Bubble Shield	Enables Bubble Crab to generate Bubble Shield to protect itself from other character.	2
Robotic Shield	Enables Bubble Crab to releases Robotic Crabs that covered by bubble to attack the other character.	3
Fly Robotic Crabs	Enables Bubble Crab to throw up Robotic Crabs in order to attack other character.	4
Bubble Splash	Enables Bubble Crab to shoot a ring of bubbles to attack other character.	5
Bubble Splash (Encased in Bubble Shield)	Enables Bubble Crab to shoot a ring of bubbles to attack other character when it inside the Bubble Shield.	6

## 2.2. Design

### 2.2.1. Main Data Structures

The main data structures that used in the program are array. It stored the sprites data, so then when there are cases or code that need the sprites, it will directly take from the variable (array).

#### Background and Sprite Data Implementation

For the background itself can be represented on the screen by function open image that put on the code of the program, so user just need to put the image file directory.

To make the sprite animation is working, user need to do sprite masking (can be done outside program or inside program), so user have to create mask for every sprite on sprite sheet that will use for the animation. After that, user have to perform an AND operation between the background and the sprites. For next step, user have to perform an OR operation between the background and the sprites in order to put the sprite on the background perfectly.

In this cases sprite sheet masking is done automatically by the program by take the coordinate that already assigned by the user, then the program will automatically do the AND and Or operation.

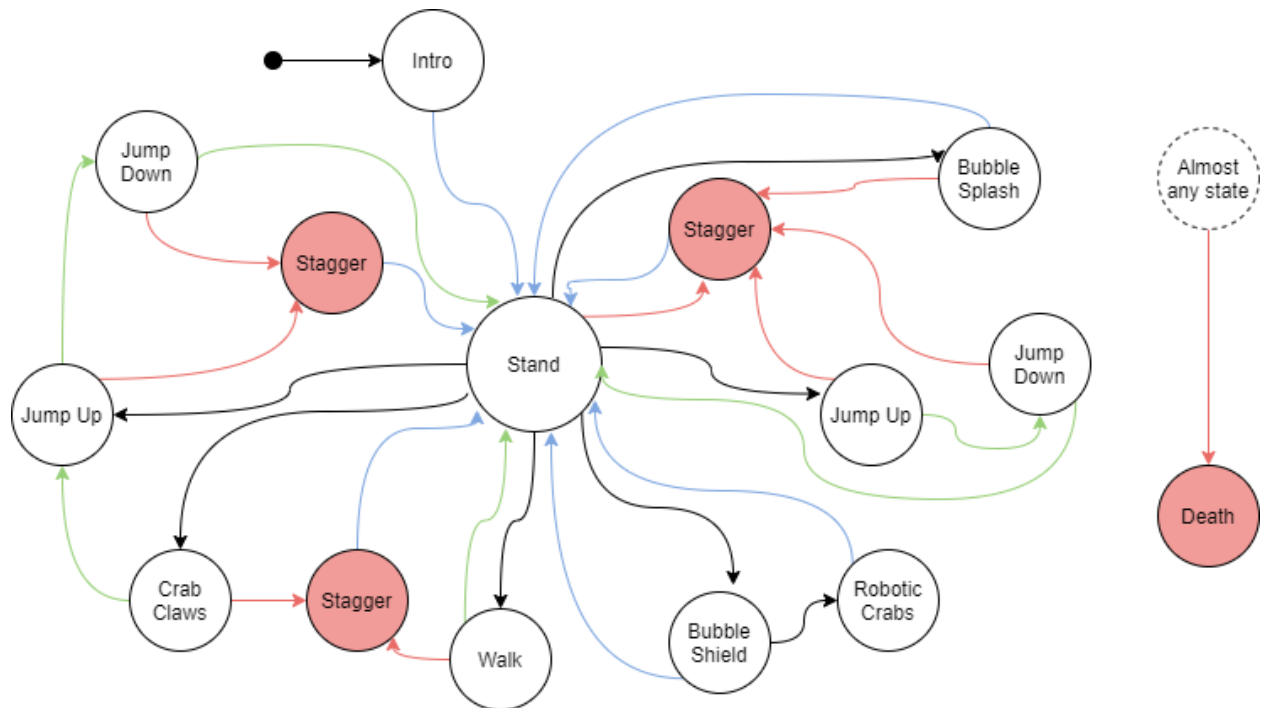
### 2.2.2. Main/Global Variables

The Main Variables are used in (Module1.vb), which are:

- TargetX and TargetY are declared as Integer with function as the collision point.
- RespawnTime is declared as Integer with function as the interval respawn time for dummy when destroyed.
- AvailTarget is declared as Boolean with function as the condition whether the dummy is available or not.

### 2.2.3. Character State Diagram and State Transition of Bubble Crab

#### Bubble Crab's State Diagram



The image above is a state diagram of Bubble Crab animation, Bubble Crab can do several movements/actions, which are introduction, stand, jump up, jump down, activate bubble shield, generates crab claws, throw bubble splash, and throw robotic crabs. There are several arrow colour-rules that used in the state diagram, black arrow means Bubble Crab/user decides to do a movement, blue arrow means Bubble Crab is finish do the animation/movement, green arrow used to show some transitions that triggered when some condition is fulfilled (in other words, inputs that are based on the triggers are drawn using green arrows), while the red arrow is used for showing the inputs based on interruptions by other characters.

The state diagram shows that the Bubble Crab will start the animation by do the introduction movements, after finish doing the introduction Bubble Crab will directly stand. After that the user can choose to demonstrate any movement that start from “stand” state. The details explanations are shown below.

1. Intro → Stand

The animation is started by introduction movement and after that Bubble Crab will directly stand as a default movement, therefore the colour of the arrow from “Intro” to “Stand” is blue.

2. Stand → Jump Up → Jump Down → Stand

If the user decides to do jump movement, Bubble Crab will directly jump vertically upwards and automatically goes down after several seconds, and finally Bubble Crab will automatically back to the first state/action which is stand. From “Stand” to “Jump Up” is a decision from the user, so then the colour of the arrow is black, from “Jump Up” to “Jump Down” is a transition that triggered because the position of Bubble Crab is above the ground (not touch the ground), so then the colour of the arrow is green, and from “Jump Down” to “Stand” is also green arrow because it’s an automatic transition.

3. Stand → Jump Up → Stagger → Stand

When the Bubble Crab is on the air after doing jump movement and it attacked/interrupted by other character, Bubble Crab will stagger and it’s shown by red arrow. From “Stagger” to “Stand” is transition that based on finished animation, so then the colour of the arrow is blue.

4. Jump Up → Jump Down → Stagger → Stand

After the Bubble Crab jump, in the few seconds it will goes down to the ground again automatically, therefore the colour of the arrow from “Jump Up” to “Jump Down” is green. During the time when Bubble Crab goes down, it can also be interrupted with other character and causes the Bubble Crab stagger, because it’s an interruption so then the arrow is red.

5. Stand → Bubble Shield → Stand

If the user decides to do Bubble Shield movement, Bubble Crab will cover itself with a bubble protection, and the colour of the arrow on state diagram is black

because it's a decision. The colour of arrow from "Bubble Shield" goes back to "Stand" is blue because it shows that the animation is finish.

6. Stand → Bubble Shield → Robotic Crabs → Stand

After generates the shield, the user can directly go to another Bubble Crab ability which is Robotic Crabs. Robotic Crabs can only be released after the user use Bubble Shield, therefore there is no other arrow that goes in to Robotic Crabs besides Bubble Shield. The colour of the arrow is black because it's another decision that user made, and from "Robotic Crabs" back to "Stand" is also blue like the other finished movement.

7. Stand → Crab Claws → Jump Up

The user can decide to generates Crab Claws, it assigned by black arrow because it's a decision. Right after release the Crab Claws, Bubble Crab will directly Jump Up (it's a part of its ability), so then it's assigned with green arrow because it's an automatic movement.

8. Stand → Crab Claws → Stagger

When Bubble Crab demonstrating the Crab Claws movement, it can be interrupted by other character, so then the state can go to Stagger and the colour is red because it's an interruption.

9. Stand → Bubble Splash → Stand

Like the other movements, Bubble Splash is also a decision, it's one of Bubble Crab's ability, therefore the colour of the arrow is black. When the movement is finish, it will directly go back to default position which is stand and assigned with the blue arrow.

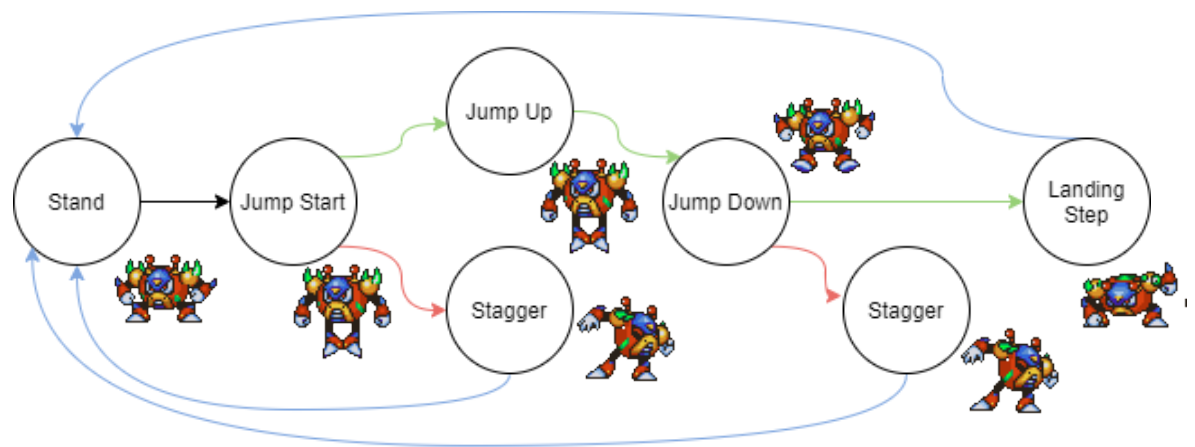
#### 10. Bubble Splash → Stagger → Stand

When the Bubble Crab throwing the Bubble Splash, it can be interrupted by another character, so then Bubble Crab also can be stagger during that time and it's assigned with the red arrow.

#### Bubble Crab's In-between State

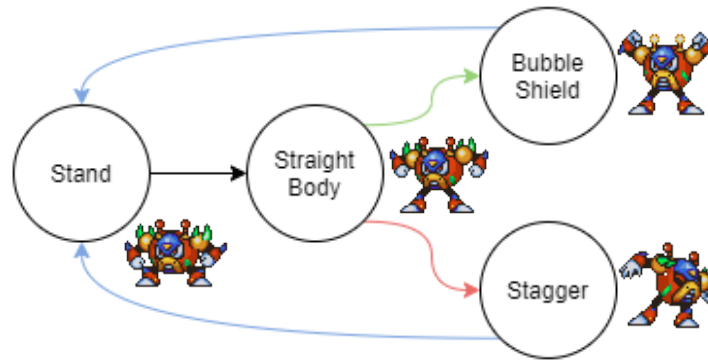
To make the movement become more real, states in-between transitions can be added in order to make the transition between states smoother. These added states are called “in-between states” as shown below.

##### 1. Jump Up and Jump Down



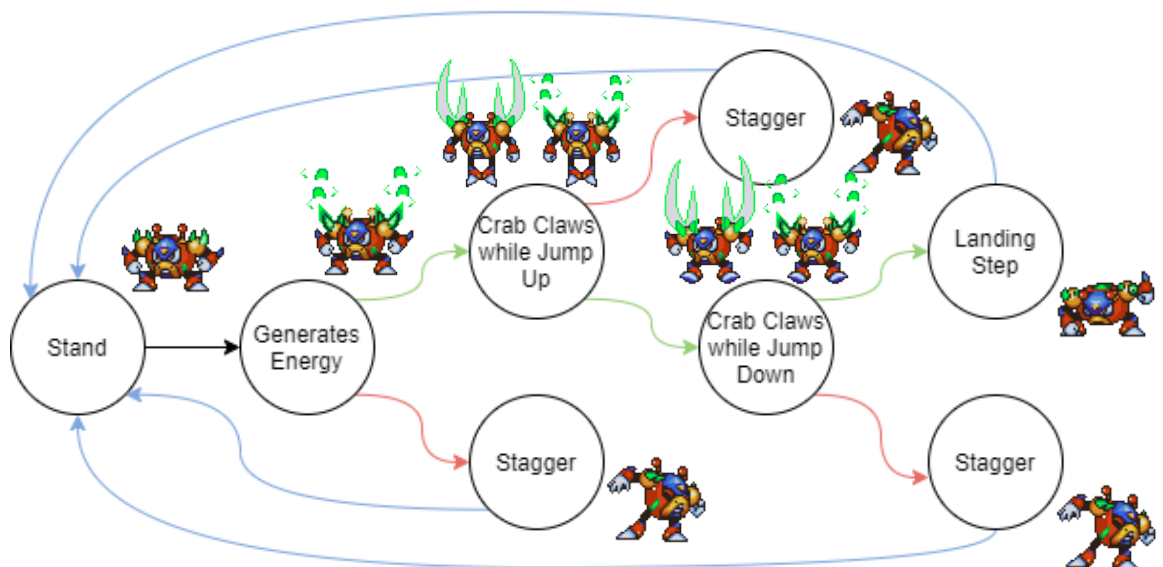
From “Start” to “Jump Up, there is one additional states which is “Jump Start”, where the Bubble Crab is do the jump movement but still in low condition. Second, for each states transitions, Bubble Crab can be interrupted by other character, therefore “Stagger” is added in-between those states. Lastly, when Bubble Crab is landing and touch the ground, there is one movement that called “Landing Step” before it perfectly stands on the ground.

## 2. Bubble Shield



There is one additional state between “Stand” and “Bubble Shield” to make this movement more real and smoother which is “Straight Body”, this additional state represents the Bubble Crab is getting ready before generates the shield. But when the Bubble Crab is getting ready, it can be interrupted, therefore there is stagger state and if Bubble Crab stagger, it will fail to construct the shield.

## 3. Crab Claws

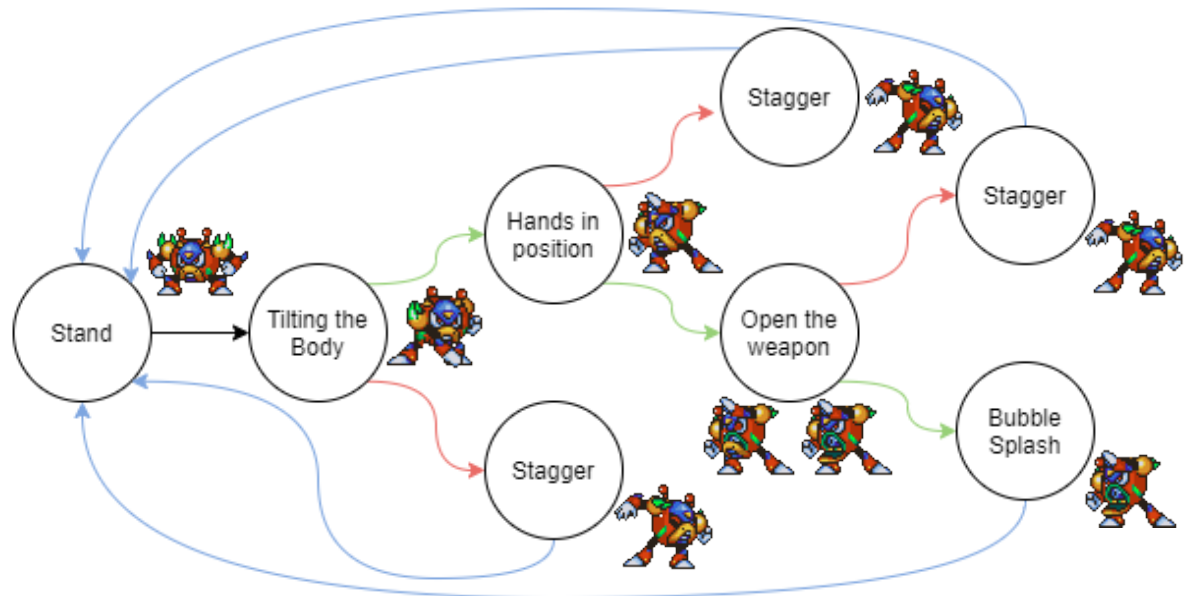


“Generates Energy” become one of additional state before the Bubble Crab release the claws and jump right after it. As we can see, the diagram shows that Bubble Crab can also be interrupted in every moment/states when it doing its Crab Claws movement, therefore there is an additional “Stagger” state between those related



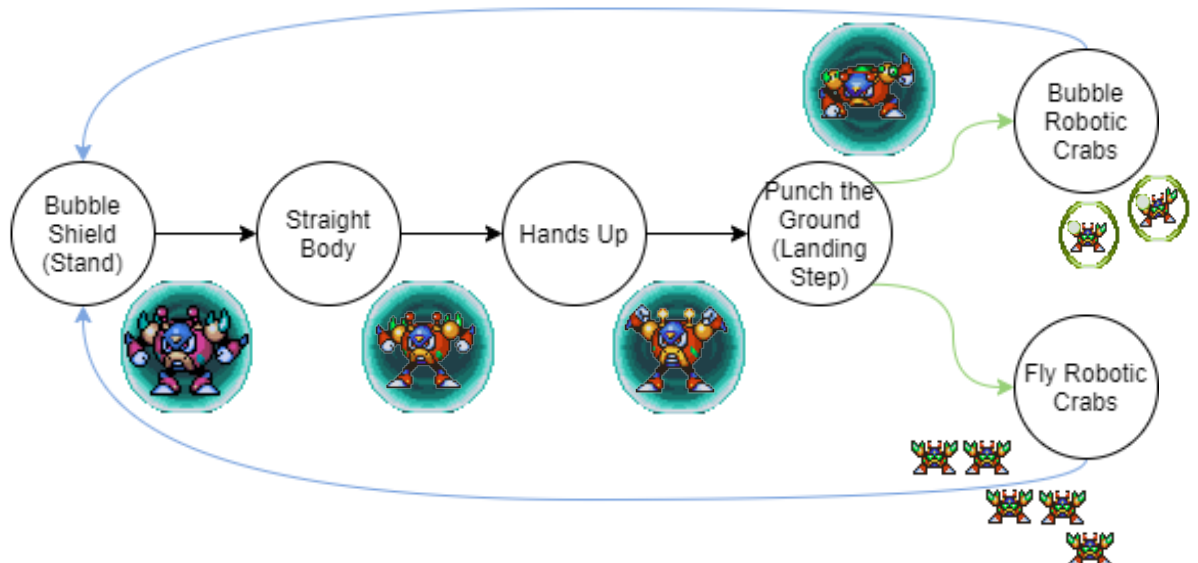
states. After Bubble Crab finish do the movement, it's not directly stand from jump, Bubble Crab will do landing step when it touches the ground right before it perfectly stands.

#### 4. Bubble Splash



There are several in-between states for this movement/action, which are “Tilting the Body”, “Hands in Position”, and “Open the Weapon” before Bubble Crab really throw the Bubble Splash. When the Bubble Crab doing those movement/states, it can be interrupted, therefore there is an additional “Stagger” state between those related states.

#### 5. Robotic Crabs



Robotic Crabs movement become the last movement that need in-between states. “Straight Body”, “Hands Up”, and “Punch the Ground” are the additional states right before Bubble Crab release the robotic crabs. Because this movement can only be done when Bubble Crab is inside the shield, and during that moment the Bubble Crab can’t be interrupted, so there is no “Stagger” state between those related states.

#### Bubble Crab’s State Transitions Table

To make clear about Bubble Crab’s movements, the transitions between states are shown by the table below.

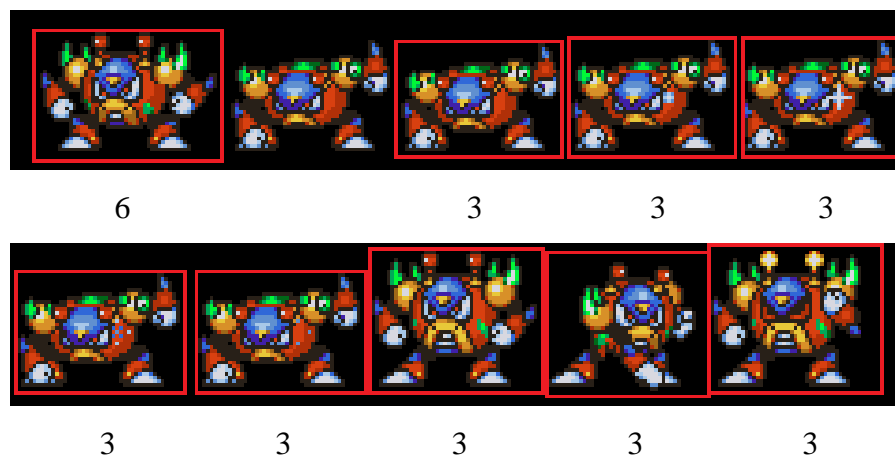
<b>Previous State</b>	<b>New State</b>	<b>Trigger</b>
Introduction	Stand	Intro animation is finished
Stand	Walk	BC decides to move left/right
Stand	Jump Up	BC decides to jump
Stand	Crab Claws	BC decides to generate Crab Claws
Stand	Bubble Splash	BC decides to attack with bubble splash
Stand	Bubble Shield	BC surrounds himself with a shield that protects it from damages
Walk	Stand	Walking movement is finished
Stand	Stagger	BC gets hit by Mega Man
Stand	Death	BC gets hit and HP = 0
Jump Up	Jump Down	Mega Man is below BC
Jump Up	Stagger	BC gets hit by Mega Man
Jump Up	Death	BC gets hit and HP = 0
Jump Down	Stand	BC lands on the ground
Jump Down	Stagger	BC gets hit by Mega Man
Jump Down	Death	BC gets hit and HP = 0
Crab Claws	Jump Up	BC automatically jump after generate Crab Claws

Crab Claws	Stand	BC lands on the ground and the attack is finished
Crab Claws	Stagger	BC gets hit by Mega Man
Crab Claws	Death	BC gets hit and HP = 0
Bubble Splash	Stand	Attack is finished
Bubble Splash	Stagger	BC gets hit by Mega Man
Bubble Splash	Death	BC gets hit and HP = 0
Bubble Shield	Stand	Attack is finished
Bubble Shield	Stagger	BC gets hit by Mega Man
Bubble Shield	Death	BC gets hit and HP = 0
Bubble Shield	Robotic Crabs	BC throws a robotic crabs diagonally upwards
Robotic Crabs	Stand	Throw robotic crabs is finished
Stagger	Stand	Stagger animation is finished

#### 2.2.4. The Sprites

The sprites of each movement/action that used in the program along with the duration of each frame (max frame time) are explained below.

##### 1. Introduction Sprites



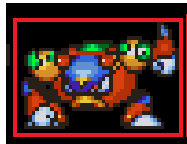
3. Jump Up and Jump Down



1

1

4. Jump End



5

5. Move Left and Right



1

1

1

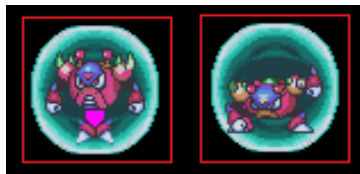
1

6. Stand



1

7. Jump Up and Down (Inside Shield)



1

1

8. Move Left and Right (Inside Shield)



1

1

1

1

9. Stand (Inside Shield)



1

10. Crab Claws



1

1

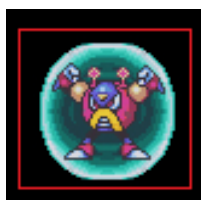
1

1

1

1

11. Bubble Shield



1

12. Bubble Robotic Crabs



1

1

1

1

13. Fly Robotic Crabs



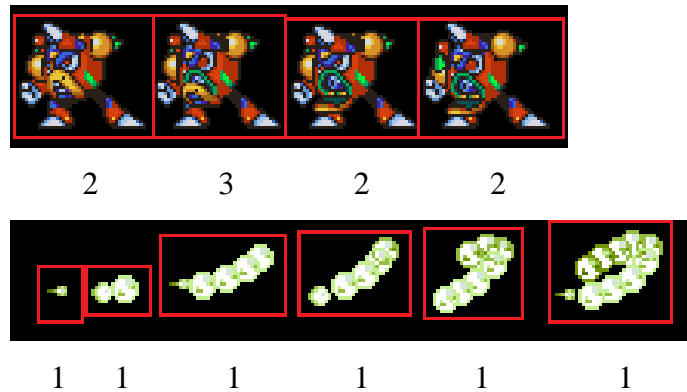
1

1

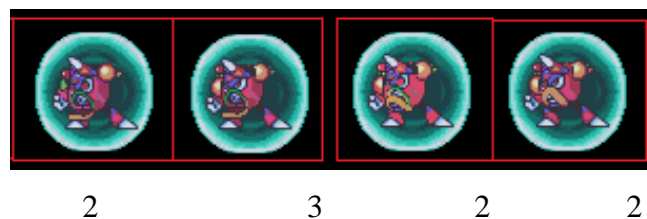
1

1

#### 14. Bubble Splash



#### 15. Bubble Splash (Inside Shield)



### 2.2.5. Collision Detection Implementation

The Collision Detection is implemented by declaring global variables with function as the collision point. If the projectile had the same posX and posY with the dummy, it assigns the posX and posY to the global variable and at the same time, the projectile is destroyed. The dummy check whether the global variable had the same posX and posY. if it is true, the dummy will be destroyed/disappear.

### 2.2.6. The Pseudocode

Refresh Screen

CheckDummyAvail

If False

    RespawnTime Start

Else \\True

ResetRespawnTime

    CreateDummy

Reset

For EveryChar in List

UpdateState

CheckCurrState

If StateIntro

    CreateDummy       \\DummyAvail = True

ElseIf StateBubbleSplash

CreateBubbleProjectile

ElseIf StateBubbleSplashInShield

CreateBubbleProjectile

ElseIf StateShootRoboticShield

Create3RoboticShield \\with the interval between frametime is 6

ElseIf StateShootFlyRobotic

    Create4FlyRobotic \\with different initial position and same frametime

CreateNewList   \\for new char

CheckDestroyChar

If False

    AddToNewList

UpdateList

DisplayOnScreen

### 2.2.7. Bonus Implementation

There are several bonuses that successfully implemented, which are:

1. Bubble Crab can throw up to four robot crabs simultaneously in different directions. 4 Fly Robotic Crab created at the same time with different initial positions. And it bounced with the order (Left - Top - Right - Bottom).
2. Bubble Crab can release up to three robot crabs simultaneously in different directions. 3 Bubble Robotic Crab created with the interval of the framerate is 6. And it floated on the water surface and moved horizontally, and also bounced when it hit the wall.

## 2.3. Evaluation

The Cases (for the screenshot of interface while performing actions are put in main interface section):

### 1. Introduction

```
6 references
Public Overrides Sub Update()
    Select Case CurrState
        Case StateSplitBC.ComeIn
            PosY = PosY + Vy
            GetNextFrame()
            If PosY >= 225 Then
                State(StateSplitBC.Intro, 1)
                Vx = 0
                Vy = 0
            End If

        Case StateSplitBC.Intro
            GetNextFrame()
            If FrameIdx = 0 And CurrFrame = 0 Then
                State(StateSplitBC.Stand, 3)
                Vx = 0
                Vy = 0
            End If

        Case StateSplitBC.Stand
            GetNextFrame()
```

This case is successful

### 2. Bubble Crab encasing himself in bubble

```
Case StateSplitBC.BubbleShieldStart
    GetNextFrame()
    If FrameIdx = 0 And CurrFrame = 0 Then
        State(StateSplitBC.BubbleShieldStand, 24)
    End If
```



This case is successful

### 3. Bubble Crab leaping

```
Case StateSplitC.LeapStart
  GetNextFrame()
  If rDir = FaceDir.Right Then
    State(StateSplitC.Leap, 5)
    Vx = 5
    Vy = -5
  elseif rDir = FaceDir.Left Then
    State(StateSplitC.Leap, 5)
    Vx = -5
    Vy = -5
  end If

Case StateSplitC.Leap
  PosX = PosX + Vx
  PosY = PosY + Vy
  Vy = Vy + 0.1
  GetNextFrame()
  If rDir = FaceDir.Right Then
    If PosX >= 400 Then
      PosX = 400
      Vx = 0
    end If
    If Vy >= 0 And CurrFrame = 0 And rDir = FaceDir.Right Then
      State(StateSplitC.LeapDown, 5)
      Vx = 5
      Vy = 5
    end If
  end If
  If rDir = FaceDir.Left Then
    If PosX <= 50 Then
      PosX = 50
      Vx = 0
    end If
    If Vy >= 0 And CurrFrame = 0 And rDir = FaceDir.Left Then
      State(StateSplitC.LeapDown, 5)
      Vx = -5
      Vy = 5
    end If
  end If
end If
```

This case is successful

### 4. Bubble Crab jumping.

```
Case StateSplitBC.Jump
  Vx = 0
  Vy = -8
  PosX = PosX + Vx
  PosY = PosY + Vy
  GetNextFrame()
  If PosY <= 100 And CurrFrame = 0 Then
    State(StateSplitBC.JumpDown, 13)
    Vx = 0
    Vy = 8
  end If

Case StateSplitBC.JumpDown
  PosX = PosX + Vx
  PosY = PosY + Vy
  GetNextFrame()
  If PosY >= 225 And CurrFrame = 0 Then
    State(StateSplitBC.JumpEnd, 14)
    Vx = 0
    Vy = 0
  end If

Case StateSplitBC.JumpEnd
  GetNextFrame()
  If FrameIdx = 0 And CurrFrame = 0 Then
    State(StateSplitBC.Stand, 3)
    Vx = 0
    Vy = 0
  end If
```

This case is successful

5. Bubble Crab shooting a ring of bubbles.

```
Case StateSplitBC.BubbleSplashStart
    GetNextFrame()
    If FrameIdx = 0 And CurrFrame = 0 Then
        State(StateSplitBC.BubbleSplashShoot, 16)
    End If

Case StateSplitBC.BubbleSplashShoot
    GetNextFrame()
    If FrameIdx = 0 And CurrFrame = 0 Then
        State(StateSplitBC.BubbleSplashEnd, 17)
    End If

Case StateSplitBC.BubbleSplashEnd
    GetNextFrame()
    If FrameIdx = 0 And CurrFrame = 0 Then
        State(StateSplitBC.Stand, 3)
        Vx = 0
        Vy = 0
    End If
```

This case is successful

6. Bubble Crab releasing a robotic crab encased in a bubble.

```
6 references
Public Overrides Sub Update()
    Select Case CurrState
        Case StateRoboticShield.Create
            GetNextFrame()
            If FrameIdx = 0 And CurrFrame = 0 Then
                State(StateRoboticShield.Ascend, 1)
                Vy = -6
            End If

        Case StateRoboticShield.Ascend
            GetNextFrame()
            PosY = PosY + Vy
            If PosY <= 108 Then
                PosY = 108
                State(StateRoboticShield.Float, 1)
                If FDir = FaceDir.Left Then
                    Vx = -6
                Else
                    Vx = 6
                End If
            End If
    End If
```

This case is successful

7. Bubble Crab throwing a robotic crab.

```
Select Case CurrState
Case StateFlyRobotic.Create
    GetNextFrame()
    If FrameIdx = 0 And CurrFrame = 0 Then
        State(StateFlyRobotic.MoveDiagonal, 1)
        If FDir = FaceDir.Left Then
            Vx = -10
        Else
            Vx = 10
        End If
        Vy = -5
    End If

Case StateFlyRobotic.MoveDiagonal
    GetNextFrame()
    PosX = PosX + Vx
    PosY = PosY + Vy
    If PosX <= 50 Then
        Vx = 10
    ElseIf PosX >= 400 Then
        Vx = -10
        Vy = 5
    ElseIf PosY <= 20 Then
        Vx = 10
        Vy = 5
    ElseIf PosY >= 250 Then
        Vx = -10
        Vy = -5
    ElseIf PosX = 100 And PosY = 100 And AvailTarget = True Then
        Destroy = True
        TargetX = 100
```

This case is successful

8. The dummy getting hit by Bubble Crab's attacks.

This case is partially implemented

9. The bonuses:

- a. Bubble Crab can throw up to four robot crabs simultaneously in different directions. (the case is successful)
- b. 4 Fly Robotic Crab created at the same time with different initial positions. And it bounced with the order (Left - Top - Right - Bottom). (the case is successful)
- c. Bubble Crab can release up to three robot crabs simultaneously in different directions. (the case is successful)
- d. 3 Bubble Robotic Crab created with the interval of the frametime is 6. And it floated on the water surface and moved horizontally, and also bounced when it hit the wall. (the case is successful)

## CHAPTER III

### CONCLUSION

#### 3.1. Work Log

##### 3.1.1. Assignment/Progress Record

Date	Activities	Personal Involved
26 - 04- 2020	Learn the material	Michael Luqmanul Hakim Yodi Fakhri
27 - 04 - 2020	Make a first report	Michael Luqmanul Hakim Yodi Fakhri
30 - 04 - 2020	Provide a background. The background must contain water that fills half the screen. The bottom part of the arena is underwater.	Yodi Fakhri
01 - 05 - 2020	Bubble Shield. Bubble Crab encases himself in a bubble, enabling him to jump higher.	Luqmanul Hakim

01 - 05 - 2020	Leap. Bubble Crab leaps in the direction he is facing. If Bubble Crab is encased in a bubble, he will leap higher	Michael
01 - 05 - 2020	Jump. Bubble Crab performs a vertical jump	Michael
02 - 05 - 2020	Change facing direction.	Luqmanul Hakim
02 - 05 - 2020	Combine the character with a skill	Yodi Fakhri
02 - 05 - 2020	Shoot a ring of bubbles. Bubble Crab shoots a ring of bubbles that travel in a parabolic path upwards.	Luqmanul Hakim
02 - 05 - 2020	Release robotic crab. Bubble Crab releases a robotic crab encased in a bubble.	Michael
03 - 05 - 2020	Throw robotic crab. Bubble Crabs throws a robotic crab diagonally upwards.	Luqmanul Hakim
03 - 05 - 2020	Recoloring the character	Yodi Fakhri
04 - 05 - 2020	Place a dummy, doll, or puppet of Megaman (or anything else that you like) as a target for Bubble Crab.	Luqmanul Hakim
05 - 05 - 2020	If the dummy gets hit by Bubble Crab or his projectiles, the dummy will disappear.	Michael

### 3.1.2. Implementation Summary

1. Provide a background. The background must contain water that fills half the screen. The bottom part of the arena is underwater. (fully implemented)
  - 1.1. It does not need to have spikes on the ceiling like in the original game. (partially implemented)
  - 1.2. The level of water can change occasionally. (not implemented)
2. Start with Bubble Crab doing his intro animation. (fully implemented)
3. Allow the user to control Bubble Crab. Bubble Crab can do the following actions:
  - 3.1. Bubble Shield. Bubble Crab encases himself in a bubble (fully implemented)
  - 3.2. Leap. Bubble Crab leaps in the direction he is facing. (fully implemented)
  - 3.3. Jump. Bubble Crab performs a vertical jump. (fully implemented)
4. Bubble Crab pauses for a moment before jumping. At the same time, huge pincers appear out of his shoulders.
  - 4.1. If Bubble Crab is encased in a bubble, the bubble will disappear. (fully implemented)
  - 4.2. Bubble Crab only jumps as high as the water level when performing this move. (fully implemented)
  - 4.3. When Bubble Crab lands on the ground, the huge pincers will disappear. (not implemented)
5. Change facing direction. (not implemented)
6. Shoot ring of bubbles. Bubble Crab shoots a ring of bubbles that travel in a parabolic path upwards. (not implemented)
  - 6.1. The ring is shot with an initial downward elevation of 15 to 30 degrees. (partially implemented)
  - 6.2. The ring disappears when it goes offscreen or hits the Megaman dummy (fully implemented)
7. Release robotic crab. Bubble Crab releases a robotic crab encased in a bubble. (fully implemented)
  - 7.1. The bubble travels upwards until it reaches water level, where it will then float. (fully implemented)

- 7.2. The bubble will burst if it collides with Bubble Crab (or shot by the Megaman dummy). (not implemented)
- 7.3. When the bubble bursts, the robotic crab will move towards the Megaman dummy. The robotic crab disappears when it hits the dummy or a wall. (partially implemented)
- 7.4. Bubble Crab can release up to three robot crabs simultaneously in different directions. (fully implemented)
- 8. Throw robotic crab. Bubble Crabs throws a robotic crab diagonally upwards. (fully implemented)
  - 8.1. The robotic crab travels in a linear path. When it hits a wall, the floor, or the ceiling, it will bounce. (fully implemented)
  - 8.2. The robotic crab disappears when it hits the Megaman dummy. (fully implemented)
  - 8.3. Bubble Crab can throw up to four robot crabs simultaneously in different directions. (fully implemented)
- 9. Place a dummy, doll, or puppet of Megaman (or anything else that you like) as a target for Bubble Crab. (fully implemented)
  - 9.1. The dummy starts at a fixed location floating on the water. Half of the dummy is underwater. (fully implemented)
  - 9.2. The dummy is by default passive. It does not perform any action. (fully implemented)
  - 9.3. The dummy is able to move around. You decide how the dummy moves around. (not implemented)
  - 9.4. The dummy can shoot projectiles. You decide how the projectiles move. (not implemented)
  - 9.5. If Bubble Crab gets hit by a projectile shot by the dummy, he staggers back. (not implemented)
- 10. If the dummy gets hit by Bubble Crab or his projectiles, the dummy will disappear. (fully implemented)
  - 10.1. After a few seconds, it will respawn at a random location floating on the water. (partially implemented)

- 10.2. The dummy must not collide with Bubble Crab when it respawns. (partially implemented)
- 10.3. Display an animation of the dummy getting destroyed. (not implemented)
- 10.4. Display an animation of the dummy getting burned when it gets hit by a fireball. (not implemented)
- 10.5. If the dummy collides with Bubble Crab while Bubble Crab is encased in a bubble, the dummy will stagger back instead. (not implemented)
- 11. Provide the proper animations for each action. (partially implemented)
- 12. Bonus points for user friendliness. Negative points for extreme user unfriendliness. (partially implemented)

### **3.2. Conclusion and Remarks**

1. Overall the program does work as expected even though in some conditions, the program doesn't work as expected.
2. The collision detection parts still don't work as expected because when the dummy was destroyed, it changed the posX and posY of the collision point. It was different with the initial collision point.
3. It was a really challenging and brainstorming assignment. But there was some fun when the program could work as expected even though there was some bug. At first, the key for controlling the character was inspired by a game called Dota 2, we set the skill or abilities key with qwerty style but after a short discussion, we changed to numeric. Hopefully this program could fulfill the requirement. And this assignment could give us another experience that can be used in future.