

# OS Project Phase-1

Shivam Pandey (918972220)

## API Documentation

### **Name**

getCurrentControllerStatus – To fetch the current register status of the game controller.

### **Signature**

```
#include "basicfunc.h"  
char *getCurrentControllerStatus();
```

### **Parameters**

NA

### **Return Value**

A character array of length 8 (a byte/8 bits), representing the current state/value of the register.

-----

### **Name**

changeBackground – To change the background on a particular layer.

### **Signature**

```
#include "basicfunc.h"  
int changeBackground(int layer, int backgroundImageIndex, int backgroundPalette);
```

### **Parameters**

int layer : The layer number on which the background image will be displayed. Accepted values: 0,1,2.

int backgroundImageIndex : The background image data to be used. Accepted values: 0,1,2,3,4.

int backgroundPalette : The palette to be used for the background image. Accepted values: 0,1,2,3.

### **Return Value**

Return 0 if the operation is performed.

Return -1 if operation failed.

---

**Name**

moveSpriteAround – Change the location of a particular sprite.

**Signature**

#include "basicfunc.h"

int moveSpriteAround(int type, int spriteControllIndex, int xOffset, int yOffset);

**Parameters**

int type : 0 means large sprite & 1 means small sprite.

int spriteControllIndex : Index of the sprite control (0-63 for large) and (0-127) for small.

int xOffset : Change in the X-value of sprite.

int yOffset : Change in the Y-value of sprite.

**Return Value**

Return 0 if the operation is performed.

Return -1 if operation failed.

---

**Name**

changeSpriteColor – Change the color palette of a particular sprite.

**Signature**

#include "basicfunc.h"

int changeSpriteColor(int type, int spriteControllIndex, int newPaletteIndex);

**Parameters**

int type : 0 means large sprite & 1 means small sprite.

int spriteControllIndex : Index of the sprite control (0-63 for large) and (0-127) for small.

int newPaletteIndex : Index of the new palette to be used. Accepted values: 0,1,2,3.

**Return Value**

Return 0 if the operation is performed.

Return -1 if operation failed.

---

**Name**

defineControls – To perform operation when one of the eight controller buttons is pressed.

Define the actions of the multi controller. Aforementioned predefined APIs like

changeBackground() can be assigned to a particular *toggle* AND/OR moveSpriteAround() can be called to move around the agent of the game on using the *direction pad* buttons.

**Signature**

```
#include "basicfunc.h"  
int defineControls(int type, int position);
```

**Parameters**

int type : Specify the type of button, "0" for direction-pad button and "1" for toggle button.  
int position : Specify the position of the button. *W-0, A-1, X-2, D-3 & U-0, J-1, K-2, I-3.*

**Return Value**

Return 0 if the operation is performed successfully.  
Return -1 if operation failed.

---

**Name**

createNewThread – This API will create a new thread and start running it.

**Signature**

```
#include "basicfunc.h"  
int createNewThread(int id, int function());
```

**Parameters**

int id : Unique ID of the thread.  
int function : The predefined function to be run on the new thread.

**Return Value**

Return 0 if the operation is performed.  
Return -1 if operation failed.

---

**Name**

stopThread– This API will stop a given thread.

**Signature**

```
#include "basicfunc.h"  
int stopThread(int id);
```

**Parameters**

int id : Unique ID of the thread.

**Return Value**

Return 0 if the operation is performed.  
Return -1 if operation failed.

---

**Name**

startThread– This API will start a given thread.

**Signature**

```
#include "basicfunc.h"  
int startThread(int id);
```

**Parameters**

int id : Unique ID of the thread.

**Return Value**

Return 0 if the operation is performed.  
Return -1 if operation failed.

---

**Name**

terminateThread – This API will terminate/kill a given thread.

**Signature**

```
#include "basicfunc.h"  
int terminateThread(int id);
```

**Parameters**

int id : Unique ID of the thread.

**Return Value**

Return 0 if the operation is performed.  
Return -1 if operation failed.

---

**Name**

changeTextData – Change the text data.

**Signature**

```
#include "basicfunc.h"  
int changeTextData(int mode, char[] newData)
```

**Parameters**

int mode : 0 means overwrite mode, 1 means append mode.  
char[] newData : Stream of characters to be shown on screen.

**Return Value**

Return 0 if the operation is performed successfully.

Return -1 if operation failed. (In case of memory overflow or if the characters are not supported by the MSX font)

---

### **Name**

changeSystemMode – Swap between text & graphics mode.

### **Signature**

#include "basicfunc.h"

int changeSystemMode(int mode, int refreshRate)

### **Parameters**

int mode : 0 means text mode, 1 means graphics mode.

int refreshRate : 0 means once every 10 ticks. 1 means once every tick and 127 means once every 127 ticks. Acceptable values are from 0 to 127.

### **Return Value**

Return 0 if the operation is performed.

Return -1 if operation failed.

---

## **Answers to the questions in Project Phase-1 PDF Doc**

- 1) Yes, my OS would support multithreading.
  - 2) APIs like *defineControls*, *changeBackground*, *moveSpriteAround*, *changeTextData*, *changeSystemMode*, *getCurrentControllerStatus* will be used to map user input with the game process, and these APIs will provide abstractions to the game developers to interface with the video graphics.
  - 3) According to me, APIs written would not break or be limited if resources are increased, as my architecture is scalable.
  - 4) I have designed the APIs keeping this thought process that what kind of abstractions will make game development easier.
-

**COMMENT :-**

Professor, as I have communicated with you in email, I could not give adequate time to this project as I am mentally unstable/depressed for the past few days. Please let me know what all comments you have on this proposal. Still, several things are to be included in this architecture, but I had very less time to think all about it.