Super GP Library v0.0.0
-Instruction

Declaration:

v0.0.0 is the first version of SGL. There're many bugs in it. If users find any of them, try to contact the author with the information in the console. To be friendly to newcomers, the library is coded in C. The C++ library will come out soon.

1. Initialize

In Sample->Initialize->empty.c

The statement "**extern _win *Window;**" is needed.

The **sgSetup()** function is to set some values. It will run only once in the beginning of the program.

The **sgLoop()** function is the main loop. It will run infinitely which means while(1)sgLoop();

In Sample->Initialize->colorful.c

Use **setColor(int r, int g, int b)** to set one RGB color. Then use **clearScreen()** to fill the whole screen with the color set before.

2.Draw

In Sample->Draw->pixel.c

The function **putPixel(int x, int y)** is to draw a point on the screen. The coordinate (0, 0) is the top-left point. And the coordinate(Screen->buffer->sizeX, Screen->buffer->sizeY) is the bottom-right point.

The function **getPixel(int x, int y)** is to get the color of the point (x, y). The return type RGB is defined in screen.h

In Sample->Draw->figure.c

The function **putQuad(int x1, int y1, int x2, int y2, int mode)** is to draw a rectangle with top-left point (x1, y1) and bottom-right point (x2, y2).

The function **putCircle(int x, int y, int r, int mode)** is to draw a circle with centre (x, y) and radius r.

The parameter mode in both function can be set as SOLID_FILL or EMPTY_FILL. SOLID_FILL means to fill the whole figure with the set color while EMPTY_FILL means just draw the outline.

The function **putLine(int x1, int y1, int x2, int y2)** is to draw a line between(x1, y1) and (x2, y2).

The function **floodFill(int x, int y, RGB c)** is to fill

the area connected to (x, y) it will stop at the points with color c.

In Sample->Draw->bmp.c

The function **loadBmp(int x, int y, char *filename)** is to loat a bmp file named "filename" with its top-left at (x, y).

The function **getImage(int left, int top, int right, int bottom, bitMap *bitmap)** is to copy an area of the screen. The points information is saved in bitmap. So it need to be declared and malloced.

The function **putImage(int left, int top, bitMap *bitmap, int op)** is to paste the points saved in bitmap. The parameter op has not been activated yet.

In Sample->Draw->text.c

The function **putString(char *str, int x, int y)** is to put down a string with visible asciis.

3.Bios

In Sample->Bios->key.c

The function **initKey()** need to be added in the sgSetup() if other key functions will be used.

The function **bioskey(int cmd)** is the main key function. If cmd is 1, the return value is whether the key buffer is empty. Else if cmd is 0, the return value is the earliest key information. If the highest digit of the information is 1, it's a key-down signal. If the highest digit of the information is 0, it's a key-up signal. That is, when one key is pressed, a down-key will be send to the key buffer, and then a up-key will be send to the key buffer. The low 8 digit are the ascii of each key.

The function **clearKeyBuffer()** is used to clear the key buffer.

In Sample->Bios->mouse.c

The function **initMouse()** is similar to initKey().

The function **mousePos()** returns the current mouse position.

The function **mouseStatus(int b)** is used to get each botton's status. Parameter b can be set to SG_LEFT_BUTTON or SG_RIGHT_BUTTON or SG_MIDDLE_BUTTON. The return value is either SG_DOWN or SG_UP.

The function **biosMouse(int cmd)** is similar to biosKey(int cmd). But its return value is vectThree. The first two int is the coordinate of the click, and the third

int is which button is clicked.

The function **clearMouseBuffer()** is similar to clearKeyBuffer().

4.Time

In Sample->Time->time.c

The function **delay(int t)** is to wait for a while. Parameter t is the microseconds that will be waited.

The function **delayBegin()** and **delayEnd(int t)** are used in pairs. The former set the start time and the latter set the end time. When the program run into delayEnd, if the time gap between delayBegin and delayEnd is less than t, the program will wait. Or else, don't wait.

The function **random(int n)** returns a number between 0 and n-1 randomly.

5.Interrupt

In Sample->Interrupt->int.c

This part is to emulate DOS interrupts.

The function **getVect(int intn)** returns the current interrupt function of intn. In DOS, int8 represent clock

interrupt and int9 represent keyboard interrupt. If intn is 8, it returns the current clock interrupt function. And if intn is 9, it returns the current keyboard interrupt function.

The function **setVect(int intn, vect v)** is used to set a new interrupt function to intn.

The function **dosInt(int intn, int *ret)** is used to get the current key ascii when in keyboard interrupt function.

The function **setFreq(int f)** is used to set the frequence of clock interrupt. That is, every second the clock interrupt function will run f times.

6.Tool

In Sample->Tool->sound.c

The function **loadWave(char *filename, int mode)** is used to play a wave file. Parameter mode need to be set as SG_LOOP.

In Sample->Tool->full.c

The function **fullScreen()** is to change the window size to the whole screen.

The function **hideMouse()** is to hide the default

mouse icon of Windows.

The function **showMouse()** is to show the default mouse icon of Windows.

In Sample->Tool->page.c

The function **setActivePage(int page)** is to set the active page. Parameter page can be either 0 or 1. Then all drawings will operate on this page.

The function **setVisualPage(int page)** is to set the VisualPage. Parameter page can be either 0 or 1. Then this page will be shown on the screen.