



## เอกสารอ้างอิง

### ภาษาไทย

ชาญชัย วิทย์สุภาเลิศ และ ทองทัสน์ จุลศักดิ์ศรี. กราฟฟิค-เอ็ดิตเตอร์ สำหรับแอปเปิล II. น. 2.48-2.54. ใน การประชุมทางวิชาการวิศวกรรมไฟฟ้า ครั้งที่ 6 เล่ม 2 คอมพิวเตอร์และวิศวกรรมสื่อสาร. คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์-หาดใหญ่. สงขลา, 2526.

บุญเลิศ เอี่ยมทัศนาศ, ยืน ภู่วรรณ, สมนึก ศิริโต. โปรแกรมคอมพิวเตอร์ภาษาซี. บริษัท ซีเอ็ดดูเคชั่น จำกัด, 2529.

ปานใจ ธารทัศนวงศ์. การพัฒนาระบบสำนักงานอัตโนมัติเชิงวัตถุ. บัณฑิตวิทยาลัยจุฬาลงกรณ์มหาวิทยาลัย, 2534

ภูชง อุทโยภาส. เอกสารประกอบคำบรรยาย. มหาวิทยาลัยเกษตรศาสตร์, 2532

สุชาติ เลื่องยศลือชากุล. โปรแกรมไมโครคอมพิวเตอร์วาดแบบวงจรอิเล็กทรอนิกส์. บัณฑิตวิทยาลัย มหาวิทยาลัยเกษตรศาสตร์, 2528.

สุชาย ธนเวสตีเยร ดร. และ วิชัย จิวังกูร. โครงสร้างข้อมูลเพื่อการออกแบบโปรแกรมคอมพิวเตอร์. บริษัท ซีเอ็ดดูเคชั่น จำกัด, 2527.

### ภาษาอังกฤษ

Bayegan, H.M. A Computer-Aided Schematic System. pp. 416-423, In CAD 78, IPC Science and Technology Press Limited, Guild, Surrey., 1987

OrCAD Systems Corporation. Schematic Design Tools. 1987

James, McCord. BORLAND C++ PROGRAMMER'S GUIDE TO GRAPHICS. SAMS, 1991

Herbert, Schildt. Using TURBO C++. Osborne McGraw-Hill

MARK, MULLIN. Object Oriented Program Design With Examples in C++. Addison-Wesley Publishing Company, Inc., 1989

BORLAND C++ 3.0. BORLAND, 1991

GREG, VOSS and PAUL, CHUI. Turbo C++ DISKTUTOR. McGraw-Hill, 1991

Richard, S., Wiener and Lewis, J., Pinson. An Introduction to Object-Oriented Programming and C++. Addison-Wesley Publishing Company, Inc., 1988

Peter, Coad and Edward, Yourdon. Objectt-Oriented Design. Prentice-Hall, Inc., 1991

Chris, H., Pappas and William, H., Murray, III. BORLAND C++ HANDBOOK. McGraw-Hill, 1992

ภาคผนวก



## ภาคผนวก

# โครงสร้างคลาสที่นำมาใช้ในงานวิจัย

ในงานวิจัยนี้มีโครงสร้างคลาอยู่ 2 ประเภท คือโครงสร้างคลาสที่ออกแบบในงานวิจัย ซึ่งกล่าวในบทที่ 3 และโครงสร้างคลาสที่ได้รับการออกแบบไว้แล้ว นำมาใช้ในงานวิจัย ได้แก่ คลาส PhysicalDisplay, Viewport, Bitbit ซึ่งจัดการเกี่ยวกับการ detect graphic, set graphic mode, close graphic, set viewport เป็นต้น ซึ่งมีการนิยามคลาสแสดงดังนี้

Lee Adams' SUPERCHARGED C++ GRAPHICS

Header file for graphics library language bindings

SOURCE FILE: LIB2DBC.HPP FOR use with Borland c++ 2.0

TYPE: declarations fou classes PhysicalDisplay and Viewport

### DEVELOPER'S GUIDE TO THE INTERFACE

Before using graphics, you must create objects of the classes PhysicalDisplay and Viewport, and call methods of each object to switch the display to a graphics mode and setup various graphics parameters. To create graphics output, call the appropriate methods of your objects. Refer to any of the demonstration programs in the book for examples of here-is-how-it's-done source code.

Class declarations are located in LIB2DBC.HPP (which must be used in your program as #include "LIB2DBC.HPP").

Function definitions are located in LIB2DBC.CPP.

Either LIB2DBC.CPP or LIB2DBC.OBJ must be named in the

project list for your program.

LICENSE: As purchaser of the book in which this source code appears you are granted a royalty-free license to reproduce and distribute executable files generated using this code provided that you accept the conditions of the License Agreement and Limited Warranty described in the book.

PUBLICATION: Contains material from Windcrest/McGraw-Hill book 3489, published by TAB BOOKS.

U.S. GOVERNMENT RESTRICTED RIGHTS: This software and its documentation is provided with restricted rights. The use, duplication, or disclosure by the Government is subject to restrictions set forth in subdivision (b)(3)(ii) of The Rights in Technical Data and Computer Software clause at 252.227-7013. Contractor/manufacturer is Lee Adams, c/o TAB BOOKS, 13311 Monterey Ave., Blue Ridge Summit PA 17294.

TRADEMARK: Lee Adams is a trademark of Lee Adams.

Copyright 1991 Lee Adams. All rights reserved worldwide.

\*/

```
#if !defined (__LARGE__)
```

```
#error Must use large memory model.
```

```
#endif
```

```
#if defined (BGI) // include the appropriate header file...
```

```
#include <graphics.h>
```

```
#elif defined (FG)
```

```
#include <fg.h>
```

```
#elif defined (MS)
```

```

#include <graph.h>
#endif

class PhysicalDisplay // physical display (adapter and monitor).
{
    // pen color, pen style, brush color, brush style, etc.
    // uses the physical coordinates of the current graphics mode.
private:
    // accessible by member functions and friends only.
    int ObjectReady;           // indicates if object initialized.
    int RAMpage;               // indicates if hidden page in RAM.
    int ScreenMode;            // current graphics mode.
    int TextWidth;             // character matrix...
    int TextHeight;
    int ClipLeft;              // viewport clipping coords...
    int ClipTop;
    int ClipRight;
    int ClipBottom;
    char SolidFill[8];         // 100% fill pattern.
    unsigned int Plane_Length; // length of bitplane.
    char far * BitMap0;        // storage of hidden page in RAM...
    char far * BitMap1;
    char far * BitMap2;
    char far * BitMap3;
    union{struct {unsigned int A1; unsigned int A2;} Address;
        char far * FarAddress;} U1; // union to calculate addresses.
    unsigned int Seg1;         // hidden bitplane segment address...
    unsigned int Seg2;
    unsigned int Seg3;
    unsigned int Seg4;
    unsigned int Off1;         // hidden bitplane offset address...
    unsigned int Off2;

```

```

unsigned int Off3;
unsigned int Off4;
int CurrentClr;           // for Zortech's fg.lib...
int WriteMode;
int CurrentX;
int CurrentY;
int FGmaxY;
int CurrentFill;
int CurrentTextX;
int CurrentTextY;
int ArrayWidth;
int ArrayDepth;
char far * InitHiddenPage(void); // initialize hidden bitmap.
void FreeMemory(char far *);    // releases a block of memory.
public: // data and methods accessible by any in-scope function.
PhysicalDisplay();           // constructor.
~PhysicalDisplay();        // Viewport objects have full access.
friend class Viewport;     // Viewport objects have full access.
friend class Layout;      // Layout objects have full access.
void Init2D(int,int,int,int,int); // sets 2D graphics state.
int SetupMode(void);       // detect and set up best graphics mode.
void ForceMode(int);       // forces a particular graphics mode.
void ShutDownGraphics(void); // return to text mode.
void SetHue(int);          // sets the current drawing color.
void SetWriteMode(int);    // sets PSET or XOR write mode.
void SetRGB(int,int,int,int); // sets RGB values for a color.
void SetLine(int);         // sets the current line style.
void SetFill(char *, int); // sets the area zFill style.
void BackUp(void);         // copies page 0 to hidden page.
void Restore(void);        // copies hidden page to page 0.

```

```

int InitUndo(void);           // sets up undo page.
void SetWritePage(int);      // sets the active page.
void SetDisplayPage(int);    // sets the displayed page.
void BlankPage(void);        // blanks the current active page.
};

class Viewport                // a viewport on the physical display.
{
    // has access to all members of class PhysicalDisplay.
    // is a virtual display using its own local coordinate system.
private: // data and methods accessible by member functions only.
    int ViewportReady;       // indicates if object initialized.
    PhysicalDisplay *GenericDisplay; // backward pointer.
public: // data and methods accessible by any in-scope function.
    Viewport(PhysicalDisplay *); // constructor.
    ~Viewport();               // destructor.
    void SetPosition(int,int); // sets the current xy position.
    void DrawLine(int,int);    // draws line from current xy position.
    void DrawBorder(int,int,int,int); // draws rectangle.
    void DrawPanel(int,int,int,int); // draws solid rectangle.
    void DrawPolygon(int, int far *); // draw a solid polygon.
    void DrawFacet(int, int far *); // draws transparent polygon.
    void DrawCircle(int,int,int); // draws circle.
    void PutText(int,int,int,char *); // display text.
    void SetTextRowCol(int,int); // set text position.
    void ClearTextLine(void); // blanks the dialog text line.
    void Fill(int,int,int); // area fill.
#if defined (BGI) // if using Borland's graphics.lib...
    char far * MemBlock(int,int,int,int); // allocate memory.
    void FreeBlock(char far *); // deallocate memory.
    void GetBlock(int,int,int,int,char far *); // save array.

```



```

void PutXOR(int,int,char far *);           // show XOR array.
void PutPSET(int,int,char far *);        // show PSET array.
void PutAND(int,int,char far *);         // show AND array.
#elif defined (FG)                       // if using Zortech's fg.lib...
    fg_color_t far * MemBlock(int,int,int,int);
    void FreeBlock(fg_color_t far 80);
    void GetBlock(int,int,int,int,fg_color_t far *);
    void PutXOR(int,int,fg_color_t far *);
    void PutPSET(int,int,fg_color_t far *);
    void PutAND(int,int,fg_color_t far *);
#elif defined (ms)                       // if using Microsoft's graph.lib...
    char far * MemBlock(int,int,int,int);
    void FreeBlock(char far *);
    void GetBlock(int,int,int,int,char far *);
    void PutXOR(int,int,char far *);
    void PutPSET(int,int,char far *);
    void PutAND(int,int,char far *);
#endif

unsigned long GetBlockSize(int,int,int,int); // size of array.
void WindowOpen(int,int,int,int); // open a viewport, overloaded.
void WindowOpen(int,int,int,int,int); //...plus border.
void WindowOpen(int,int,int,int,int,int); //...plus repaint.
void WindowClose(int,int); // shuts down viewport.
void WindowClear(int bgclr=0); // clear viewport, default 0.

};

/*

```

End of LIB2DBC.HPP header file.

## Lee Adams' SUPERCHARGED C++ GRAPHICS

Header file for class Bitblt

SOURCE FILE: BITBLT.HPP

TYPE: include file for class Bitblt.

Bitblt class: PROGRAMMER'S GUIDE TO THE INTERFACE

Each instance of the Bitblt class can manipulate one copy of a graphic array image in the bitmap buffer. Use multiple instances of the Blitter class to control multiple images or copies of the same image.

GrabImage().....allocates memory, grabs the image from the bitmap and stores it in a graphic array.

DisplayImage()...displays the bitblt if not already displayed.

EraseImage().....restores the previous background.

MoveImage().....moves the currently-displayed bitblt to another location.

NameImageFile()..specifies the filename for disk IO.

SaveImageFile()..saves the graphic array to disk using the current filename.

LoadImageFile()..loads a graphic array from disk using the current filename.

Class declarations are located in BITBLT.HPP (which must be used in your program as #include "BITBLT.HPP").

Function definitions are located in BITBLT.CPP (which uses #include file alloc.h, graphics.h, io.h, stdio.h, and string.h). Either BITBLT.CPP or BITBLT.OBJ must be named

in the make file project list for your program.

Each function returns 1 if successful, -1 if error.

LICENSE: As purchaser of the book in which this source code appears you are granted a royalty-free license to reproduce and distribute executable files generated using this code provided that you accept the conditions of the License Agreement and Limited Warranty described in the book.

PUBLICATION: Contains material from Windcrest/McGraw-Hill book 3489, published by TAB BOOKS.

U.S. GOVERNMENT RESTRICTED RIGHTS: This software and its documentation is provided with restricted rights. The use, duplication, or disclosure by the Government is subject to restrictions set forth in subdivision (b)(3)(ii) of The Rights in Technical Data and Computer Software clause at 252.227-7013. Contractor/manufacturer is Lee Adams, c/o TAB BOOKS, 13311 Monterey Ave., Blue Ridge Summit PA 17294.

TRADEMARK: Lee Adams is a trademark of Lee Adams.

Copyright 1991 Lee Adams. All rights reserved worldwide.

\*/

```
#ifndef BITBLT_HPP
```

```
#define BITBLT_HPP
```

```
class Bitblt // class declarations.
```

```
{
```

```
private: // can be accessed by member functions only.
```

```
Viewport *Generic; // points to a viewport object.
```

```
#if defined (BGI)
```

```

char far * Block;          // points to memory block for image.
char far * BgBlock;      // points to memory block for background.
#ifdef FG
    fg_color_t far * Block;
    fg_color_t far * BgBlock;
#endif
#ifdef MS
    char far * Block;
    char far * BgBlock;
#endif
#endif

unsigned long BlockSize;   // size in bytes of graphic array.
int YES,NO,ERROR,SUCCESS; // token.
int OVERWRITE;           // copy as original form.
int NEGATIVE;           // copy as reverse form.
int SUPERIMPOSE;        // OR: superimpose as opaque.
int FILTER;             // XOR: superimpose as semi-transparent.
int STENCIL;            // AND: common areas.
int MAXIMUM_X,MAXIMUM_Y; // screen dimensions.
int X1,Y1,X2,Y2;        // bounding box.
int Width,Depth;        // dimension of box.
int OldX1,OldY1;        // previous origin coords.
int Logic;              // current write mode.
int DefaultLogic;       // default write mode.
char FileName[13];      // filename.
int Ready;              // indicates if save in memory.
int Displayed;         // indicate if displayed.
int MemAllocated;      // indicates if memory allocated.
int BgMemAllocated;    // indicates if memory allocated.
int GraphicsError;     // result of library function.
int MaxX,MaxY;         // max legal origin coords.
void FreeBlock(void);   // deallocates memory.

```

```
public:          // can be accessed by any in-scope function.
    Bitblt(int,int,int,int,int,int,Viewport *);
    ~Bitblt();          // destructor.
    int GrabImage(void);
    int DisplayImage(int Xa,int ya,int Mode);
    int EraseImage(void);
    int MoveImage(int XOffset,int YOffset,int Mode);
    int NameImageFile(char * Name);
    int SaveImageFile(char * Name="UNTITLED.BLK");
    int LoadImageFile(char * Name="UNTITLED.BLK");
};

#endif
```

Supercharged C++ Graphics -- end of source file BITBLT.HPP



## ประวัติผู้เขียน

นายประสิทธิ์ จิยะพานิชกุล เกิดเมื่อวันที่ 4 กันยายน พ.ศ. 2511 ที่จังหวัดราชบุรี สำเร็จการศึกษาปริญญาตรี วิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า (คอมพิวเตอร์ อิเล็กทรอนิกส์) จากคณะวิศวกรรมศาสตร์ มหาวิทยาลัยเกษตรศาสตร์ในปีการศึกษา 2533 และเข้าทำงานที่บริษัท บางกอกโพลทกลาส จำกัด (บริษัทในเครือกระจกไทยอาชาสี) ในตำแหน่งวิศวกรเป็นเวลา 7 เดือน จากนั้นเข้าศึกษาต่อในหลักสูตร วิศวกรรมศาสตรมหาบัณฑิตสาขาวิศวกรรมไฟฟ้า (ระบบเชิงเลข) ณ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัยในปีการศึกษา 2534