

An experiment report on

**“Snake Game Player”**

Created by

Patwary Mohammad Sazzad Hossain (张一鸣)

ID: 1911521212

Class: 19lq

Under The Guidance of

**Dr. Liu Chun**

**(**Assistant professor, Dept. of CST)

**Department of Computer Science & Technology**

**Hubei University of Technology**

**Wuhan, Hubei Province, P. R. China**

**June 20, 2022**

**CHAPTER PAGE NO.**

**List of Figure i**

**List of Table ii**

**Acknowledgment iii**

**Abstract iv**

**1. INTRODUCTION 1-2**

**1.1 Introduction 1**

**1.2 Objectives 1**

**1.3 Project Description 1**

**2. HARDWARE DESIGN 3-10**

**2.1 Hardware Diagram 3**

2.1.1 The Hardware Diagram consists of 3

**2.2 Circuit Diagram 3**

2.2.1 Components Used in the Circuit 4

2.2.2 Components Description 4

**3. SOFTWARE DESIGN 11-23**

**3.1 Software Diagram 11**

**3.2 Flow Chart 11**

**3.3 Function 12**

3.3.1 Draw a robust snake 12

3.3.2 Double linked list 13

3.3.3 Back Insertion 14

3.3.4 Typical Dlink list operation 16

3.3.5 Snake move 17

3.3.6 Cross the boundary 18

3.3.7 Food reproduce 18

3.3.8 snake\_map[] 19

**3.4 Another method to design software 20**

3.4.1 Big point 20

3.4.2 Snake 20

3.4.3 Food 21

3.4.4 Snake move 22

3.4.5 Initial values used in the I/O ports 23

**4. SYSTEM DESIGN 25-27**

**4.1 Complete System Diagram (Hardware) 25**

**4.2 Complete System Diagram (Software) 25**

4.2.1 Game Running 26

4.2.2 Game Over 27

4.2.3 New Game 27

**5. DISCUSSION & REFERENCE 28**

**5.1 Discussions 28**

**5.2 References 28**

**6. CODE 29-96**

**6.1 main.c 29**

**6.2 lcddrv.c 35**

**6.3 target.c 52**

**6.4 Startup.s 57**

**6.5 config.h 72**

**6.6 lcddrv.h 76**

**6.7 LPC2124.h 81**

**6.8 target.h 93**

**6.9 LPC2124.sct 96**

**FIGURE PAGE NO.**

**Figure 1: Hardware Diagram 3**

**Figure 2: Circuit Diagram 3**

**Figure 3: LPC2124 4**

**Figure 4: Pin Diagram of LPC 2124 5**

**Figure 5: LM3228 LCD 7**

**Figure 6: 4X4 KEYPAD 8**

**Figure 7: KEYPAD PINOUT 9**

**Figure 8: 10K Resistor 10**

**Figure 9: Software Diagram 11**

**Figure 10: Flowchart 11**

**Figure 11: Nine small cells 12**

**Figure 12: Doubly Linked List Node 13**

**Figure 13: Back insertion 15**

**Figure 14: Complete System Diagram (Hardware) 25**

**Figure 15: Complete System Diagram (Software) 25**

**Figure 16: Game running-1 26**

**Figure 17: Game running-2 26**

**Figure 18: Game over 27**

**Figure 19: New game 27**

**TABLE PAGE NO.**

**Table 1: Components Used in the Circuit 4**

**Table 2: PINSEL Table 6**

**Table 3: LM3228 LCD Pin Description 8**

**Table 4: Description of KEYPAD Pin 10**

**Acknowledgment**

We offer gratitude to project supervisor Asst. Prof. **Dr. Liu Chun** (Dept. Of Computer Science & Technology, HBUT) for guiding us throughout the project, the effort and giving us lectures and explaining the project and helping when we were stops in the middle during the development phase. We couldn’t have built such a challenging project without his help.

**Abstract**

This project aims to bring the fun and simplicity of snake game with some new features. It will include computer controlled intelligent opponents whose aim will be to challenge the human players. It will also have the multiplayer feature that will allow more than one players to play the game over a network.

This project explores a new dimension in the traditional snake game to make it more interesting and challenging. The simplicity of this game makes it an ideal candidate for a minor project as we can focus on advanced topics like multiplayer functionality and implementation of computer controlled intelligent opponents.

**1.1 Introduction**

Project documentation is concerned with describing the delivered software product, in this case the Snake game project. Project documentation includes user documentation which tells users how to use the software product and system documentation which is principally intended for further development and understanding.

**1.2 Objectives**

Snake game is a mobile action game, whose goal is to control a snake to move and collect food in a map. In this paper we develop a controller based on movement rating functions considering smoothness, space, and food. Scores given by these functions are aggregated by linear weighted sum, and the snake takes the action that leads to the highest score. To find a set of good weight values, we apply an evolutionary algorithm. We examine several algorithm variants of different crossover and environmental selection operators. Experimental results show that our design method is able to generate smart controllers.

**1.3 Project Description**

Snake Game project is a mobile game application. This game is created for the recreation of people. To create this game we actually need some hardware device and software application. For hardware we mainly need micro-controller, LCD and Keypad. Now how this game may played.

Firstly Welcome screen,

This is the starting screen in your game, on pressing s should start the game. On this screen, you have to provide an overview of the scoring system and the controls.

Secondly Moving the Player

Currently, the snake doesn't move at all. Let's make it move. To Do:

• The snake should move up, down, left, and right when you press those arrows.

• The snake shouldn't have a sudden change in direction. It should not move left immediately if it is moving right or vice versa. Same is true for up and down movements i.e., if the snake is going up and the down arrow is pressed, nothing should happen. That is supposed to be true for every opposite direction.

• The enumeration for UP, DOWN, LEFT and RIGHT are defined in (key.h). Snake should eat the food

If you get the snake moving, you will see that it just walk (slithers) over the food right now. To Do:

• Make it eat the food.

• It should grow by size if it eats a growing food and should reduce in size if it eats a reducing (junk) food. These food are indicated by 'X' and 'O' on the board.

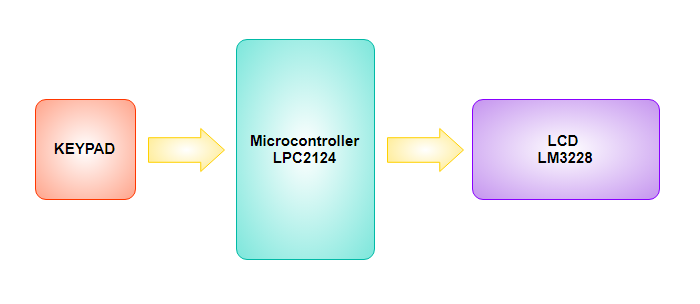
• Once eaten, the food should disappear and another food (of random type) should be spawn at a random location on the game window.

Show points

No points are printed right now. To Do:

• Calculate the points as per the food eaten\*

**2.1 Hardware Diagram**

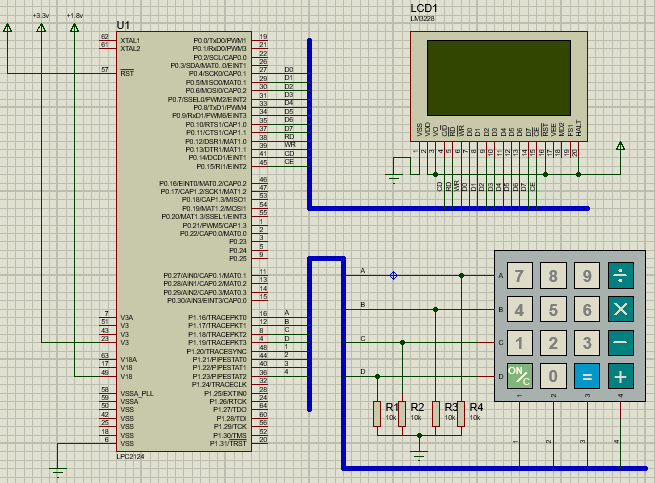
****

**Figure 1: Hardware Diagram**

**2.1.1 The Hardware Diagram consists of**

1. Microcontroller LPC2124
2. LCD LM3228
3. KEYPAD

**2.2** **Circuit Diagram**



**Figure 2: Circuit Diagram**

**2.2.1 Components Used in the Circuit**

|  |  |  |
| --- | --- | --- |
| **SL. No.** | **Components** | **Specification** |
| 1 | Microcontroller | LPC2124 |
| 2 | LCD | LM3228 |
| 3 | KEYPAD | SMALLCALC |
| 4 | Resistors | 10K |

**Table 1: Components Used in the Circuit**

**2.2.2 Components Description**

**1. Microcontroller**

In this project we’re using the NXP (founded by Philips) LPC2124 Microcontroller. The NXP LPC2124 is an ARM7TDMI-S based high-performance 32-bit RISC Microcontroller with Thumb extensions 256KB on-chip Flash ROM with In-System Programming (ISP) and In-Application Programming (IAP) 16KB RAM, Vectored Interrupt Controller, Two UARTs, I2C serial interface, 2 SPI serial interfaces, Two timers (7 capture/compare channels), PWM unit with up to 6 PWM outputs, 4-channels 10bit ADC, Real Time Clock, Watchdog Timer, General purpose I/O pins. CPU clock up to 60 MHz, On-chip crystal oscillator and On-chip PLL.

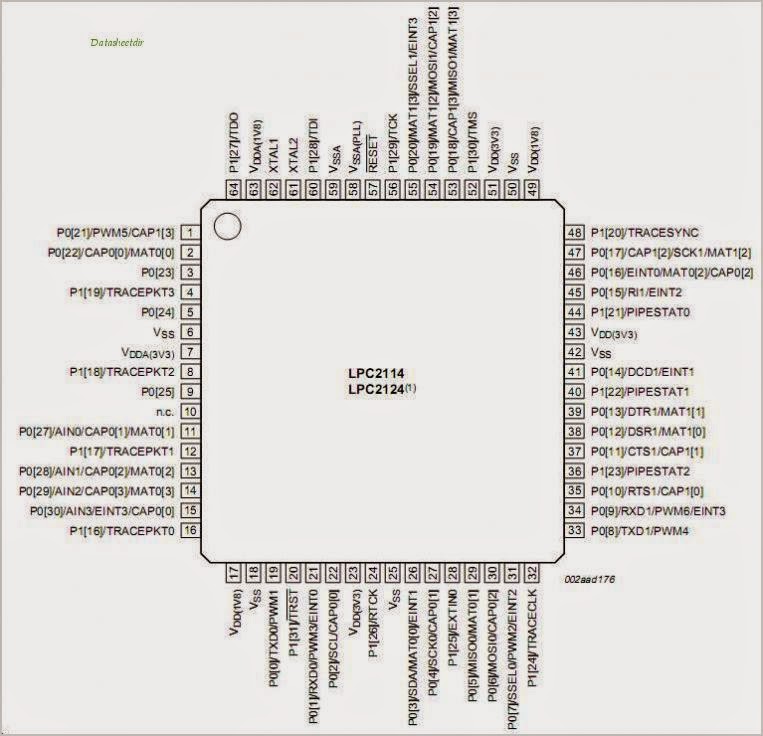


**Figure 3: LPC2124**

**Basics of LPC2124**

When coming to ARM7 Programming there are 5 things you need to be get familiarize with. They are-

* PINSEL
* IODIR
* IOSET
* IOCLR
* IOPIN



**Figure 4: Pin Diagram of LPC 2124**

**PINSEL:**

A 32 bit register which is used to select the function of the pins in which the user needs it to operate. There are four functions for each pins of the controller, in which the first function one was GPIO (General Purpose Input Output). It means that the pin can either act as an Input or Output with no specific functions.

There are totally three PINSEL register in LPC2124 Controller in order to control the functions of the Pins in the respective ports. The classification is given below

PINSEL0 – Controls functions of Port0.0 – Port0.15

PINSEL1 – Controls functions of Port0.16-Port0.31

PINSEL2 – Controls functions of Port1.16-Port1.31

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **PINSEL0** | **Pin Name** | **Function when 00** | **Function when 01** | **Function when 10** | **Function when 11** | **Reset Value** |
| 1:0 | P0.0 | GPIO Port 0.0 | TxD (UART0) | PWM1 | Reserved | 00 |
| 3:2 | P0.1 | GPIO Port 0.1 | RxD (UART0) | PWM3 | EINT0 | 00 |
| 5:4 | P0.2 | GPIO Port 0.2 | SCL (I2C) | Capture 0.0 (TIMER0) | Reserved | 00 |
| 7:6 | P0.3 | GPIO Port 0.3 | SDL (I2C) | Match 0.0 (TIMER0) | EINT1 | 00 |
| 9:8 | P0.4 | GPIO Port 0.4 | SCK (SPI0) | Capture 0.1 (TIMER0) | Reserved | 00 |
| 11:10 | P0.5 | GPIO Port 0.5 | MISO (SPI0) | Match 0.1 (TIMER0) | Reserved | 00 |
| 13:12 | P0.6 | GPIO Port 0.6 | MOSI (SPI0) | Capture 0.2 (TIMER0) | Reserved | 00 |
| 15:14 | P0.7 | GPIO Port 0.7 | SSEL (SPI0) | PWM2 | EINT2 | 00 |
| 17:16 | P0.8 | GPIO Port 0.8 | TxD (UART1) | PWM4 | Reserved | 00 |
| 19:18 | P0.9 | GPIO Port 0.9 | RxD (UART1) | PWM6 | EINT3 | 00 |
| 21:20 | P0.10 | GPIO Port 0.10 | RTS (UART1) | Capture 1.0 (TIMER1) | Reserved | 00 |
| 23:22 | P0.11 | GPIO Port 0.11 | CTS (UART1) | Capture 1.1 (TIMER1) | Reserved | 00 |
| 25:24 | P0.12 | GPIO Port 0.12 | DSR (UART1) | Match 1.0 (TIMER1) | Reserved | 00 |
| 27:26 | P0.13 | GPIO Port 0.13 | DTR (UART1) | Match 1.1 (TIMER1) | Reserved | 00 |
| 29:28 | P0.14 | GPIO Port 0.14 | CD (UART1) | EINT1 | Reserved | 00 |
| 31:30 | P0.15 | GPIO Port 0.15 | RI (UART1) | EINT2 | Reserved | 00 |

**Table 2: PINSEL Table**

**IODIR:**

Like DDR in AVR and TRIS in PIC, ARM uses IODIR register to specify the direction which in which we are going to use the pins. Two 32 bit registers IODIR0 for Port0 (P0.0 – P0.31) and IODIR1 for Port (P1.16- P1.31). Kindly note that loading values in IODIR, it will take effect only if the Pins are used as GPIO and the directions are controlled automatically if it was specified with any special functions.

**IOSET:**

This Register is meant to set the pins in the Ports where writing 1 to it will set the respective pin while 0 will have no effect. There are two registers dedicated for both the ports IOSET0 –P0.0 – P0.31 and IOSET1 for P1.16 – P1.31

**IOCLR:**

This Register is meant to clear the pins in the Ports where writing 1 will clear the respective pin while 0 will have no effect in the Ports. There are two registers dedicated for both the ports IOCLR0 –P0.0 – P0.31 and IOCLR1 for P1.16 – P1.31

**IOPIN:**

This is used only when we assign certain pins as Input in the IODIR register. There are two registers dedicated for both the ports IOPIN0 –P0.0 – P0.31 and IOPIN1 for P1.16 – P1.31.

**2. LCD**

In this project we’re using LM3228 LCD (Liquid Crystal Display). It’s a 128 x 64 Dots Graphic LCD. It has 20 pin to control whole things.

**FEATURES**

• 128 x 64 dots + 4 Icons

• Built-in controller (KS0108)

• + 5V power supply

• 1/64 duty cycle

• EL backlight (built-in EL inverter)

• Built-in N.V



**Figure 5: LM3228 LCD**

|  |  |  |
| --- | --- | --- |
| **PIN NUMBER** | **SYMBOL** | **FUNCTION** |
| 1 | VSS | Ground (0V) |
| 2 | VDD | Logic Supply Voltage (+5V) |
| 3 | VO | LCD drive voltage for contrast adjustment |
| 4 | C/D | WR=“L”...C/D=“H” : Command write C/D=“L”: Data write  RD=“L”...C/D=“H” : Status read C/D=“L”: Data read |
| 5 | RD | Data read Active Low |
| 6 | WR | Data write Active Low |
| 7 | D0 | Data Bus Line 0 |
| 8 | D1 | Data Bus Line 1 |
| 9 | D2 | Data Bus Line 2 |
| 10 | D3 | Data Bus Line 3 |
| 11 | D4 | Data Bus Line 4 |
| 12 | D5 | Data Bus Line 5 |
| 13 | D6 | Data Bus Line 6 |
| 14 | D7 | Data Bus Line 7 |
| 15 | CE | Chip enable Active Low |
| 16 | RST | Chip reset Active Low |
| 17 | VEE | Negative voltage input for LC drive (Negative voltage output for  models with on-board negative voltage generator) |
| 18 | MD2 | Mode Selection |
| 19 | FS1 | Terminals for selection of font size |
| 20 | HALT | Halt Function (H = Normal, L = Stop Oscillation) |

**Table 3: LM3228 LCD Pin Description**

**3. KEYPAD**

We’re using **4X4 KEYPAD in this project.**

****

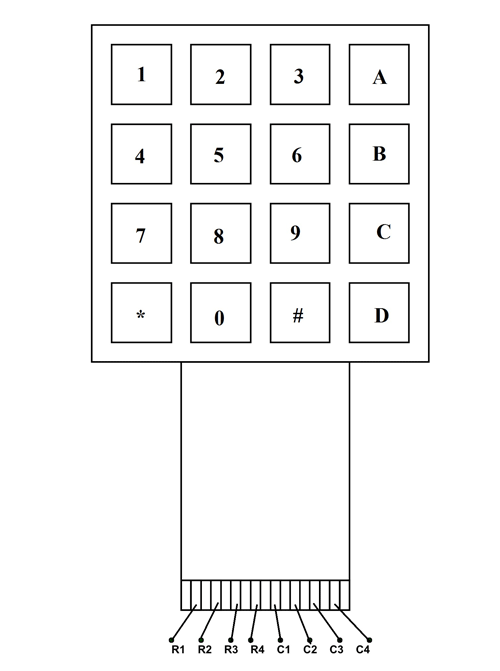
**Figure 6: 4X4 KEYPAD**

**4X4 KEYPAD MODULE Features and Specifications**

* Maximum Voltage across EACH SEGMENT or BUTTON: 24V
* Maximum Current through EACH SEGMENT or BUTTON: 30mA
* Maximum operating temperature: 0°C to + 50°C
* Ultra-thin design
* Adhesive backing
* Easy interface
* Long life.

**4X4 KEYPAD Pin Configuration**

4X4 KEYPAD MODULES are available in different sizes and shapes. But they all have same pin configuration. It is easy to make 4X4 KEYPAD by arranging 16 buttons in matrix formation by yourself.



**Figure 7: KEYPAD PINOUT**

|  |  |
| --- | --- |
| **Pin Number** | **Description** |
| 1 | PIN1 is taken out from 1st ROW |
| 2 | PIN1 is taken out from 2nd ROW |
| 3 | PIN1 is taken out from 3rd ROW |
| 4 | PIN1 is taken out from 4th ROW |
| 5 | PIN1 is taken out from 1st COLUMN |
| 6 | PIN1 is taken out from 2nd COLUMN |
| 7 | PIN1 is taken out from 3rd COLUMN |
| 8 | PIN1 is taken out from 4th COLUMN |

**Table 4: Description of KEYPAD Pin**

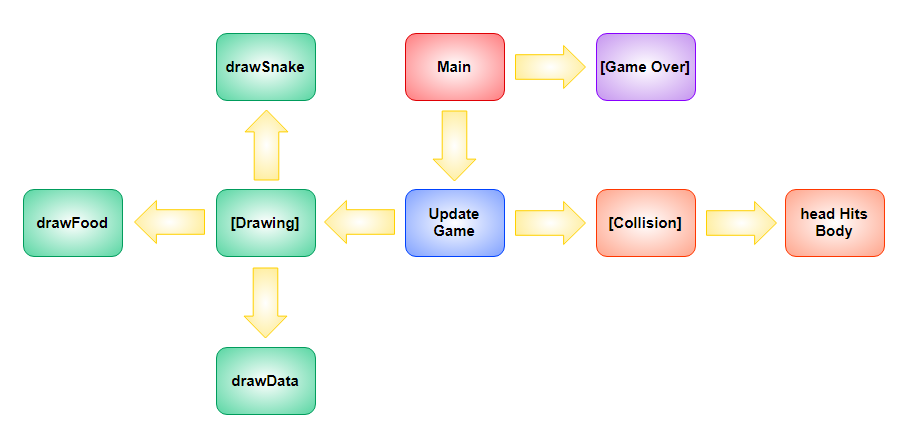
**4. Resistor**

A resistor is a passive two-terminal electrical component that implements electrical resistance as a circuit element. In electronic circuits, resistors are used to reduce current flow, adjust signal levels, to divide voltages, bias active elements, and terminate transmission lines, among other uses. Here we using 10K resistor.



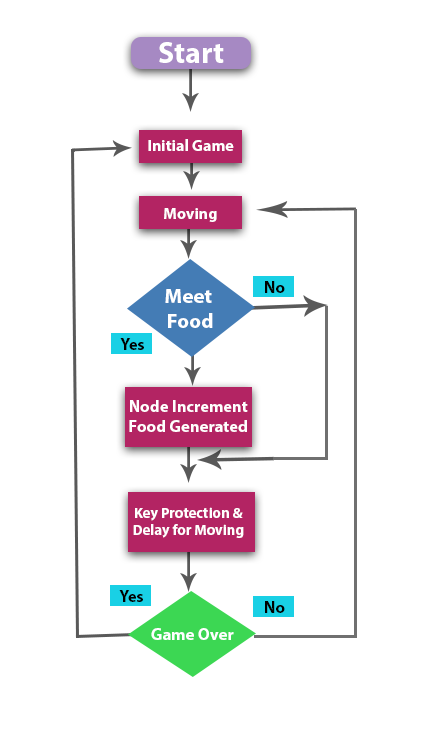
**Figure 8: 10K Resistor**

**3.1 Software Diagram**

****

**Figure 9: Software Diagram**

**3.2 Flow Chart**

****

**Figure 10: Flowchart**

**3.3 Function**

**3.3.1 Draw a robust snake**

We should draw a robust snake instead of just one point.

****

**Figure 11: Nine small cells**

**Robust Snake Code:**

void Display(uint8 x,uint8 y)

{

GUI\_Point(x, y, LCD\_DISP\_COLOR);

GUI\_Point(x+1, y, LCD\_DISP\_COLOR);

GUI\_Point(x-1, y, LCD\_DISP\_COLOR);

GUI\_Point(x, y+1, LCD\_DISP\_COLOR);

GUI\_Point(x, y-1, LCD\_DISP\_COLOR);

GUI\_Point(x+1, y+1, LCD\_DISP\_COLOR);

GUI\_Point(x+1, y-1, LCD\_DISP\_COLOR);

GUI\_Point(x-1, y+1, LCD\_DISP\_COLOR);

GUI\_Point(x-1, y-1, LCD\_DISP\_COLOR);

}

void Clear(uint8 x,uint8 y)

{

GUI\_Point(x, y, LCD\_BACK\_COLOR);

GUI\_Point(x+1, y, LCD\_BACK\_COLOR);

GUI\_Point(x-1, y, LCD\_BACK\_COLOR);

GUI\_Point(x, y+1, LCD\_BACK\_COLOR);

GUI\_Point(x, y-1, LCD\_BACK\_COLOR);

GUI\_Point(x+1, y+1, LCD\_BACK\_COLOR);

GUI\_Point(x+1, y-1, LCD\_BACK\_COLOR);

GUI\_Point(x-1, y+1, LCD\_BACK\_COLOR);

GUI\_Point(x-1, y-1, LCD\_BACK\_COLOR);

}

**3.3.2 Double linked list**

struct part

{

uint8 x, y;

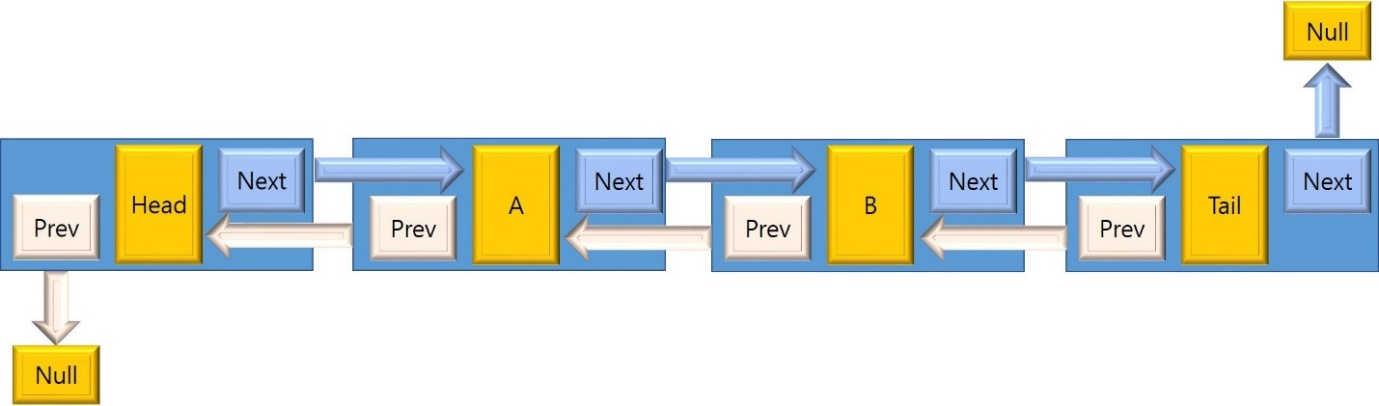
struct part \*next,\*per;

};

head\_x=21;head\_y=10;

struct part \*head,\*tail;

uint8 snake\_map[42][21];

****

**Figure 12: Doubly Linked List Node**

In contrast to the singly linked list, our doubly linked list node will have two pointers LITERALLY pointing to the next and previous node.

For all linked list implementations, we must have either a head and/or a tail. I will mention this just in case. The head and tail node are the first and last node of a linked list respectively.

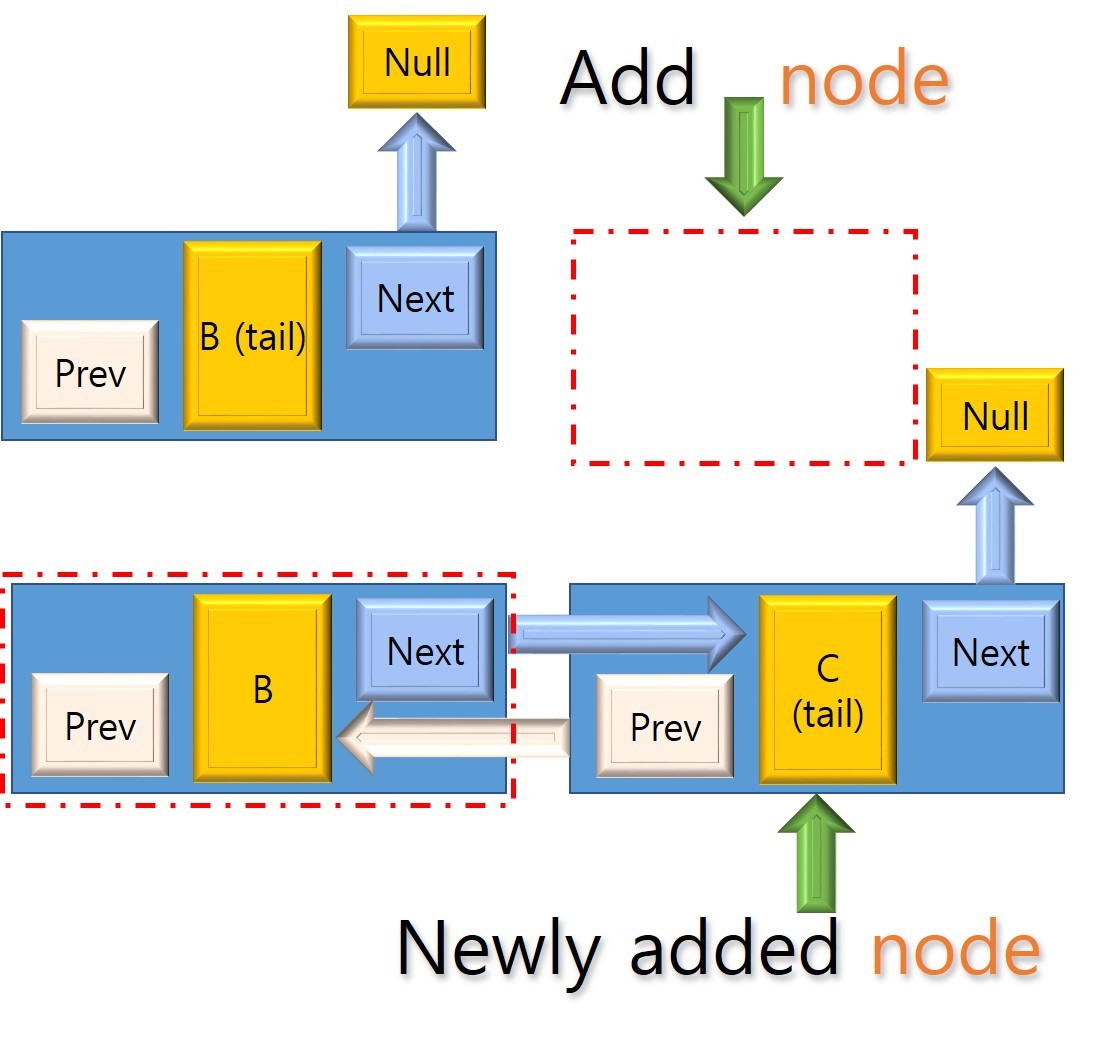
So our Node container will have the following attributes.

* Data.
* Next node.
* Previous node.

**3.3.3 Back Insertion**

The back insertion mirrors the front insertion.

1. Create new node (C).
2. Check if list is empty.
   * If head is empty
     1. set C as the new head. Existence of head node indicates that there are elements in the list (well, at least for this implementation).
   * Otherwise, check if tail is null.
     1. If tail is null
        1. Set next node pointer of current head (B) to point to new node (C).
        2. Set prev node pointer of new node (C) to point at current head (B).
     2. Otherwise,
        1. Set the prev node of C to point at current tail (B).
        2. Set the next node of B to point at new node (C).
   * Assign the newly created node (C) as the new tail.
   * Set next node of C to null.

****

**Figure 13: Back insertion**

**Back Insertion Pseudo Code:**

begin insertAtBack(T dataToInsert):

// First element.

if head is null:

nodeToInsert = new Node(dataToInsert)

head = nodeToInsert

// Tail and head cannot point at the same node

tail = null

else:

// Second element to add to list

if tail is null:

tail = new Node(dataToInsert)

// Update references

// curHead --> Tail ，curHead <-- Tail

curhead.setNext(this.tail)

tail.setPrev(this.curhead)

else:

prevTail = tail

newTail = new Node(dataToInsert)

// Update references

// prevTail --> newTail ， prevTail <-- newTail

newTail.setPrev(prevTail)

prevTail.setNext(newTail)

tail = newTail

end if else;

end if else;

increment size

end insertAtBack;

**3.3.4 Typical Dlink list operation**

int dlink\_append\_last(void \*pval)

{

node \*pnode=create\_node(pval);

if (!pnode)

return -1;

pnode->next = phead; //insert before phead

pnode->prev = phead->prev;

phead->prev->next = pnode;

phead->prev = pnode;

count++;

return 0;

}

void Init\_snake() //control snake with dlink

{ struct part \*body;

head=(struct part\*)malloc(sizeof(struct part));

tail=(struct part\*)malloc(sizeof(struct part));

body=(struct part\*)malloc(sizeof(struct part));

body->x=21;

body->y=10;

body->next=NULL;

body->per=NULL;

snake\_map[21][10]=1;

head->next=body;

tail->per=body;

size=1;

Display(tail->per->x\*3+1,tail->per->y\*3+1);

//begin at the middle point 64\*32

}

**3.3.5 Snake move**

Move(dis){

if(dis==1) head\_x++; // 1:move to right side

if(dis==2) head\_x--; // 2: move to left side

if(dis==3) head\_y++; // 3: move down

if(dis==4) head\_y--; // 4: move up }

**3.3.6 Cross the boundary**

if(head\_x==43) head\_x=1;//42\*3+1=127

if(head\_x==0) head\_x=42;

if(head\_y==20) head\_y=1;

if(head\_y==0) head\_y=20;

**3.3.7** **Food reproduce**

if(head\_x==X&&head\_y==Y)

{

food();

size++;

}

void food()

{

while(1)

{

X=rand()%40;

Y=rand()%20;

if(X>0&&Y>0)

if(snake\_map[X][Y]==0)

{

Display(X\*3+1,Y\*3+1);

break;

}

}

}

**3.3.8 snake\_map[]**

* Snake\_map[] is used to record the location of the snake and decide whether to eat it.
* That is to say, each time we only need to determine whether the corresponding point of map is 1.If each snake traverses every point, the time complexity is high.
* To judge whether you eat yourself, you have to go through the whole snake, but if I store it in map according to the corresponding coordinates when drawing snakes, I will find the corresponding point directly when judging, so I don't need to go through the snake.

unsigned int display\_snake()

{

struct part \*body;

body=(struct part\*)malloc(sizeof(struct part));

body->x=head\_x;

body->y=head\_y;

head->next->per=body;

body->next=head->next;

head->next=body;

Display(head\_x\*3+1,head\_y\*3+1);

if(snake\_map[head\_x][head\_y]==1) return 0;

else snake\_map[head\_x][head\_y]=1;

if(head\_x==X&&head\_y==Y) {

food();

size++; }

else {

snake\_map[tail->per->x][tail->per->y]=0;

Clear(tail->per->x\*3+1,tail->per->y\*3+1);

tail->per=tail->per->per;

free(tail->per->next);

}

return 1;

}

**3.4 Another method to design software**

**3.4.1 Big point**

//画一个大点，0<x<127,0<y<63

void Big\_Point(uint8 x, uint8 y,TCOLOR color)

{

int i;

for(i = 0;i < 9;i++)

{

GUI\_Point(x+dir[i][0],y+dir[i][1],color);

}

}

**3.4.2 Snake**

//初始化一条蛇，长度为30，头部在显示屏中心

void snake\_init(int x,int y)

{

int i ;

snake\_length = 0; //初始长度为0

for(i = 0;i < 10;i++) //长度为10的蛇

{

Big\_Point(x,y,LCD\_DISP\_COLOR);

snake[snake\_length].x = x;

snake[snake\_length].y = y;

snake\_length++;

x -= 3;

}

}

**3.4.3 Food**

//随机产生一个食物

void creat\_food()

{

int i,flag = 1;

Big\_Point(food\_x,food\_y,LCD\_BACK\_COLOR); //消除旧点

do//先执行一次，产生一个新事物

{

//srand((unsigned)time(NULL));

food\_x = rand() % 127;

food\_y = rand() % 63;

for(i = 0;i < snake\_length-1;i++) //判断新产生的点是否在蛇身上

{

if(snake[i].x+2 <= food\_x || food\_x <= snake[i].x-2 || snake[i].y+2 <= food\_y || food\_y <= snake[i].y-2)

flag = 0;

}

}while(flag);

Big\_Point(food\_x,food\_y,LCD\_DISP\_COLOR); //产生新点

}

**3.4.4 Snake move**

void snake\_move(int direction){//蛇运动

int i;

struct Snake last; //保留最后一个点的坐标

last = snake[snake\_length-1];

//将后面的点向前移动，就是将前面的点坐标保存在后面一个数组节点上

for(i = snake\_length - 1;i > 0;i--) {

snake[i]= snake[i-1]; }

snake[0].x += dir[direction][0] \* 3; //将头向direction移动

snake[0].y += dir[direction][1] \* 3;

if(snake[0].x >= 127)snake[0].x = 1; //判断是否出界

else if(snake[0].x <= 0)snake[0].x = 126;

else if(snake[0].y <= 0)snake[0].y = 62;

else if(snake[0].y >= 63)snake[0].y = 1;

for(i = 1;i < snake\_length-1;i++) {//判断头是否撞到蛇身

if(snake[i].x+2 >= snake[0].x && snake[0].x >= snake[i].x-2

&& snake[i].y+2 >= snake[0].y && snake[0].y >= snake[i].y-2)

{ gameover = 1; //游戏结束

return ; } }

if(snake[0].x+2 >= food\_x && food\_x >= snake[0].x-2 //判断是否吃到食物

&& snake[0].y+2 >= food\_y && food\_y >= snake[0].y-2) {

snake[snake\_length] = last; //吃到食物将食物放到最后，就是之前保存的最后一个点

creat\_food(); //产生新食物

snake\_length++; //蛇身加长 }

else {

Big\_Point(last.x,last.y,LCD\_BACK\_COLOR); //消除蛇尾 }

//重新显示蛇

Big\_Point(snake[0].x,snake[0].y,LCD\_DISP\_COLOR); //将头显示出来}

**3.4.5 Initial values used in the I/O ports**

/获取运动方向

int GetDir()

{

int temp,temp1,temp2;

IO0DIR |= 0x0000F; //列输出

IO0SET |= 0x0000F; //列拉高

temp1 = IO0PIN & 0x000F0000; //判断行

//获取行

switch(temp1)

{

case 0x0010000:temp1 = 0;break;

case 0x0020000:temp1 = 1;break;

case 0x0040000:temp1 = 2;break;

case 0x0080000:temp1 = 3;break;

default:temp1 = 4;

}

if(temp1 != 4) //有键按下

{

IO0DIR &= 0xFFFFFFF0; //列输入

IO0DIR |= 0x000F0000; //行输出

IO0CLR |= 0x0000000F; //列置0

IO0SET |= 0x000F0000; //行拉高

temp2 = IO0PIN & 0x0000000F;

//获取列

switch(temp2)

{

case 0x001:temp = temp1\*4 + 0;break;

case 0x002:temp = temp1\*4 + 1;break;

case 0x004:temp = temp1\*4 + 2;break;

case 0x008:temp = temp1\*4 + 3;break;

default:temp = 0;

}

IO0CLR |= 0xF0000; //行拉低

IO0DIR &= 0x0FFFF; //行输入

}

return temp;

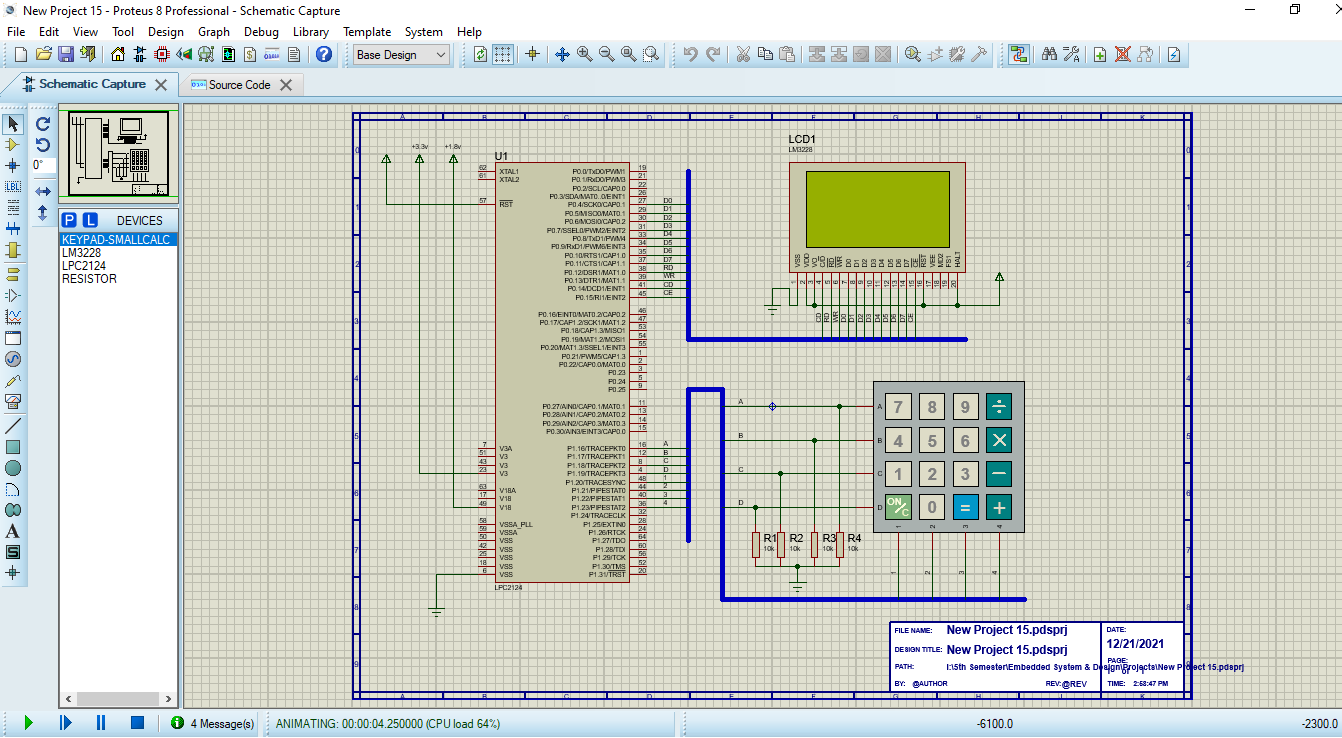
}

**4.1 Complete System Diagram (Hardware)**

****

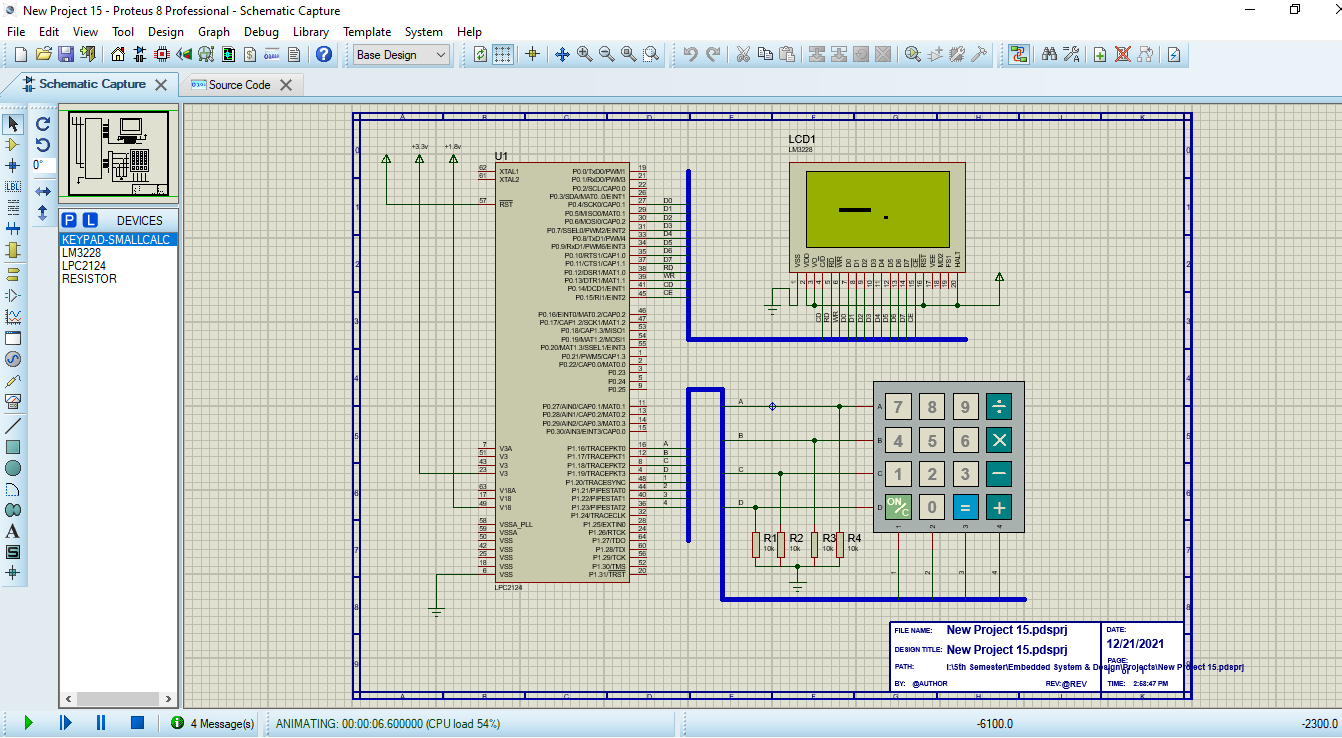
**Figure 14: Complete System Diagram (Hardware)**

**4.2 Complete System Diagram (Software)**

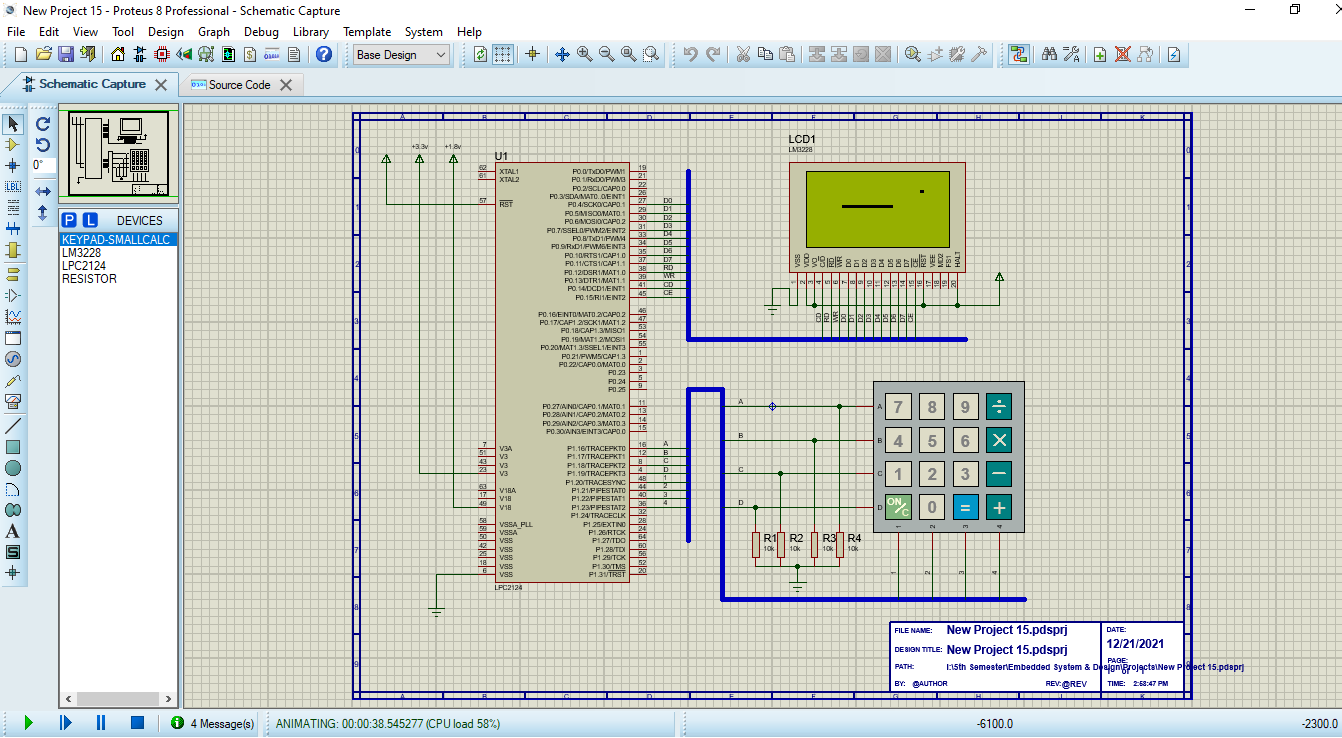


**Figure 15: Complete System Diagram (Software)**

**4.2.1 Game Running**

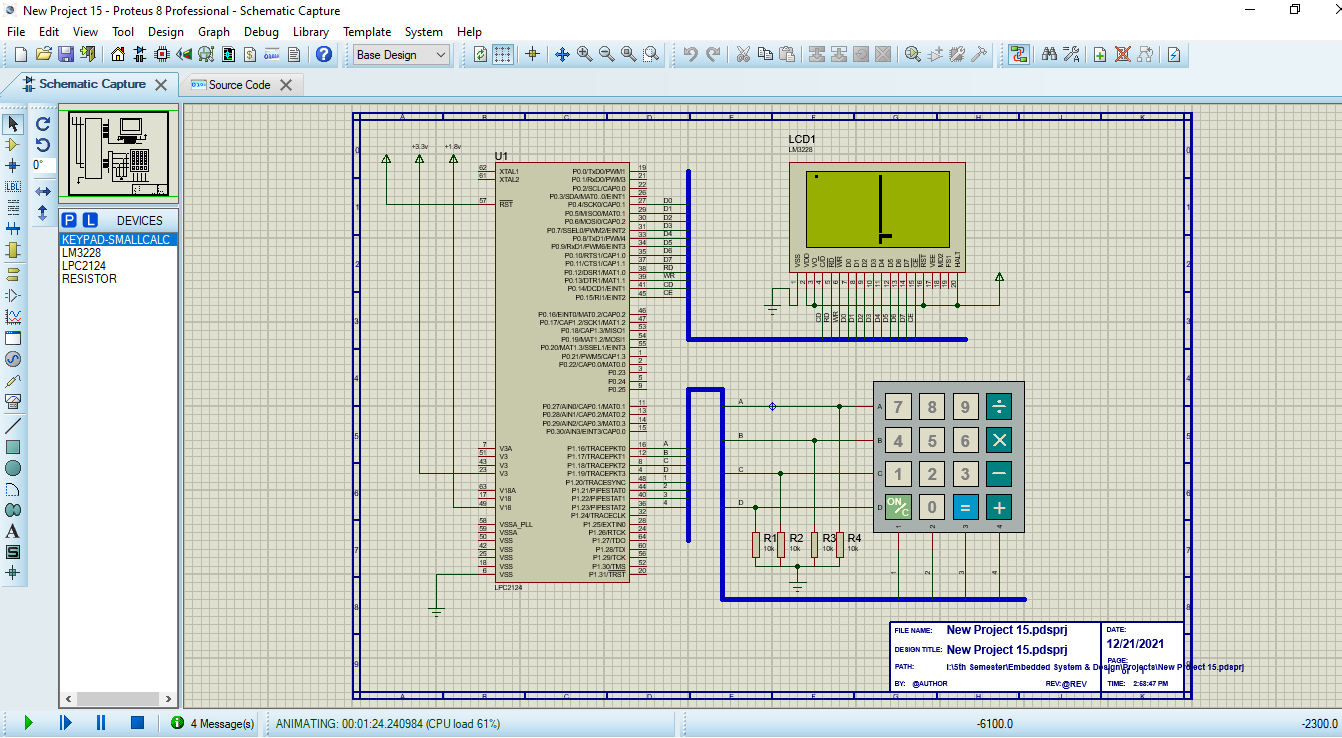


**Figure 16: Game running-1**

****

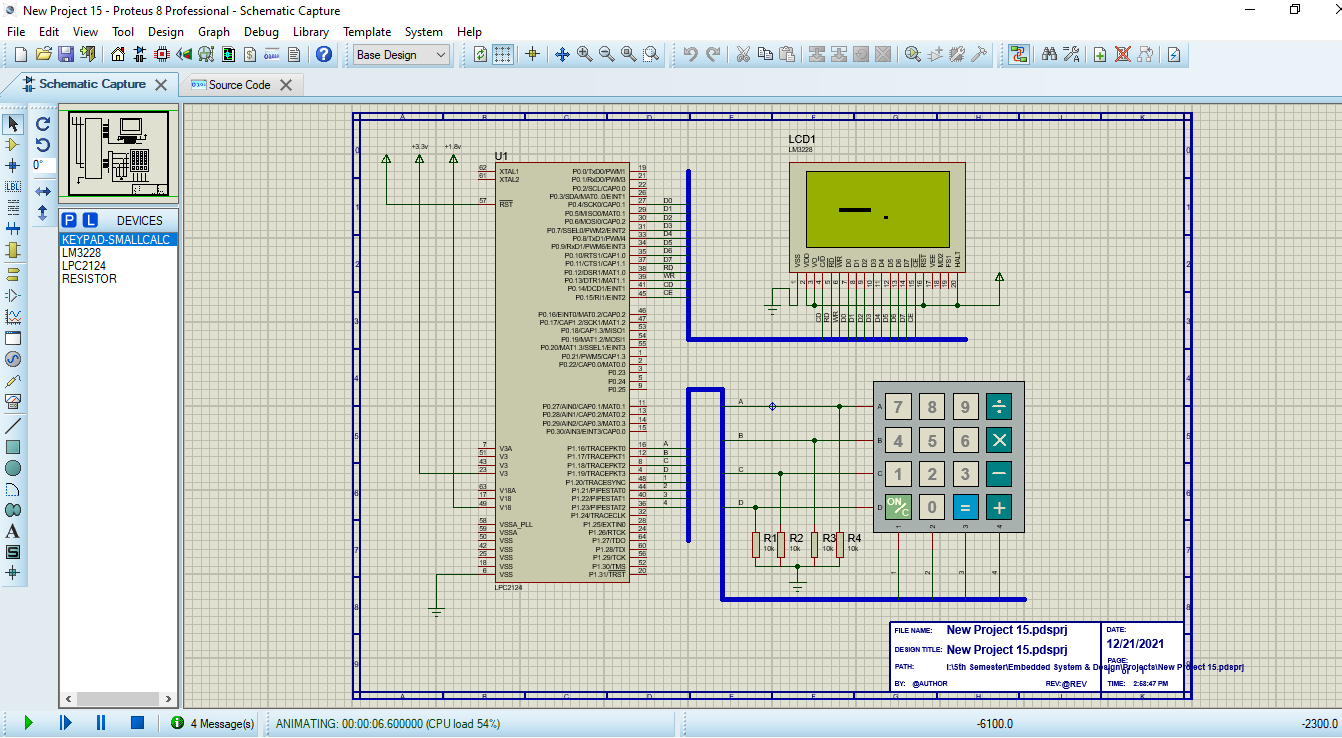
**Figure 17: Game running-2**

**4.2.2 Game Over**

****

**Figure 18: Game over**

**4.2.3 New Game**

****

**Figure 19: New game**

**5.1 Discussions**

The coding of Snake was extremely difficult with many errors arising. Many systems had to be written numerous ways before a final working solution was found. For example, two different movement methods were used prior to final version; however, even the final version is flawed as vertical movement causes the snake to change scale. There were also issues with the food – snake collision detection. While the final version resulted in a snake that could eat food, the movement glitch caused the food to cause further size issues.

Despite the fact that the game could not truly be played due to the fact no score could be given, the game is still satisfying. With the exception of the size glitch when turning, the snake responds to user input and moves around the screen as directed. Given longer to work on this, the collision detection with the movement would be the first thing fixed. By fixing this, all other sections of code that are currently not working would run. The leaderboard would work as there would be correct scores input, and the snake would grow as the food would cause it to only increase by one and not varying numbers based on direction. In addition, fixing the movement would allow for the snake to die when colliding with itself. In the current state, the snake moves as a matrix so it can not kill itself as it would be impossible to move in any direction. This failure to establish a perfect movement system was the biggest disappointment of the game as all other problems stemmed from it.

For these reasons, it is recommended that anyone who wishes to recreate this game starts simply when writing the code. It is advisable that they first perfect the snake movement controls before messing with the food generation. By taking the code in small sections, it is easier to get individual features to work. Building off this, use functions to contain each aspect of the game. Using functions made it easier to determine where errors were occurring when debugging the code. It also kept the code more organized.

**5.2 References**

[**https://www.w3schools.com/**](https://www.w3schools.com/)

[**https://stackoverflow.com/**](https://stackoverflow.com/)

[**https://www.geeksforgeeks.org/**](https://www.geeksforgeeks.org/)

[**https://wenku.baidu.com/**](https://wenku.baidu.com/)

**6.1 main.c**

#include "config.h"

#include "lcddrv.h"

#include <string.h>

/\*\*\*\*\*\*\*\*\*ASCII \*\*\*\*/

TCOLOR disp\_color;

TCOLOR back\_color;

int dir[9][2]={0,-1,0,1,-1,0,1,0,-1,1,-1,-1,1,1,1,-1,0,0};

int snake\_length, snake\_dir = 3,food\_x,food\_y,gameover;

int keyvalue[16]={

-1, 0,-1,-1,

2,-1, 3,-1,

-1, 1,-1,-1,

-1,-1,-1,-1};

////////////////

struct Snake

{

int x,y;

}snake[1000];

void delayMs(int n)

{

int i;

for(i = 0;i < 1000;i++)

for(;n>0;n--);

}

/////ASCII color1//color2////

void GUI\_SetColor(TCOLOR color1, TCOLOR color2)

{

GUI\_CopyColor(&disp\_color,color1);

GUI\_CopyColor(&back\_color,color2);

}

//////////0<x<127,0<y<63

void Big\_Point(uint8 x,uint8 y,TCOLOR color)

{

int i;

for (i=0;i < 9;i++)

{

GUI\_Point(x+dir[i][0],y+dir[i][1],color);

}

}

///////////////////////////////////

void snake\_init(int x,int y)

{

int i ;

snake\_length = 0;

for (i = 0;i < 10;i++)

{

Big\_Point(x,y,LCD\_DISP\_COLOR);

snake[snake\_length].x = x;

snake[snake\_length].y = y;

snake\_length++;

x-=3;

}

}

///////////////////

void creat\_food()

{

int i,flag = 1;

Big\_Point(food\_x,food\_y,LCD\_BACK\_COLOR);

do

{

//stand((unsigned)time(NULL));

food\_x = rand() % 127;

food\_y = rand() % 63;

for( i= 0;i < snake\_length-1;i++)

{

if(snake[i].x+2 <= food\_x || food\_x <= snake[i].x-2 || snake[i].y+2 <= food\_y || food\_y <= snake[i].y-2)

flag = 0;

}

}while(flag);

Big\_Point(food\_x,food\_y,LCD\_DISP\_COLOR);

}

/////////////////////////

void snake\_move(int direction)

{

int i;

struct Snake last;

last = snake[snake\_length-1];

/////////////////////////////////////////////

for(i = snake\_length - 1;i>0;i--)

{

snake[i]=snake[i-1];

}

////////// diraction/////////////////

snake[0].x += dir[direction][0] \* 3;

snake[0].y += dir[direction][1] \* 3;

///////////////

if(snake[0].x>= 127)snake[0].x = 1;

else if(snake[0].x <= 0)snake[0].x = 126;

else if(snake[0].y <= 0)snake[0].y = 62;

else if(snake[0].y >= 63)snake[0].y = 1;

///////////////////

for(i = 1; i< snake\_length -1; i++)

{

if(snake[i].x + 2>= snake[0].x && snake[0].x>= snake[i].x-2 && snake[i].y+2 >= snake[0].y && snake [0].y >= snake[i].y-2)

{

gameover =1;

return;

}

}

////////////

if(snake[0].x +2 >= food\_x && food\_x >= snake[0].x-2 &&snake[0].y+2 >= food\_y && food\_y >= snake[0].y-2)

{

snake[snake\_length]=last;

creat\_food();

snake\_length++;

}

else

{

Big\_Point(last.x,last.y,LCD\_BACK\_COLOR);

}

//////////

Big\_Point(snake[0].x,snake[0].y,LCD\_DISP\_COLOR);

}

//////////////

int GetDir()

{

int temp,temp1,temp2;

IO0DIR |=0x0000F;

IO0SET |=0x0000F;

temp1= IO0PIN& 0X000F0000;

/////////////

switch(temp1)

{

case 0X0010000:temp1 = 0;break;

case 0X0020000:temp1 = 1;break;

case 0X0040000:temp1 = 2;break;

case 0X0080000:temp1 = 3;break;

default:temp1 = 4;

}

if(temp1 != 4)

{

IO0DIR &= 0xFFFFFFF0;

IO0DIR |= 0x000F0000;

IO0CLR |= 0x0000000F;

IO0SET |= 0x000F0000;

temp2 = IO0PIN & 0x0000000F;

///////////////////////////

switch(temp2)

{

case 0x001:temp = temp1\*4 + 0;break;

case 0x002:temp = temp1\*4 + 1;break;

case 0x004:temp = temp1\*4 + 2;break;

case 0x008:temp = temp1\*4 + 3;break;

default:temp = 0;

}

IO0CLR |= 0xF0000;

IO0DIR &= 0x0FFFF;

}

return temp;

}

int main (void)

{

int dir = 3;

GUI\_Initialize();

GUI\_SetColor(LCD\_DISP\_COLOR,LCD\_BACK\_COLOR);

snake\_init(64,32);

creat\_food();

while (!gameover)

{

dir = keyvalue[GetDir()];

if(dir == -1)

dir = snake\_dir;

if(snake\_dir + dir != 1 && snake\_dir + dir !=5)

{

snake\_dir = dir;

}

snake\_move(snake\_dir);

delayMs(100000);

}

}

**6.2 lcddrv.c**

#include "config.h"

#include "lcddrv.h"

/\* ¶¨Òå×ÜÏßÆðÊ¼µÄGPIO£¬¼´D0¶ÔÓ¦µÄGPIOÖµ(P0.4) \*/

/\* ¶¨ÒåÏÔÊ¾»º³åÇø \*/

#define BUS\_NO 4

/\* Êä³ö×ÜÏßÊý¾Ýºê¶¨Òå \*/

#define OutData(dat) IO0DIR = IO0DIR |(0xff<<BUS\_NO); IO0CLR = 0xff<<BUS\_NO; IO0SET = (dat&0xff)<<BUS\_NO

#define InData() IO0DIR = IO0DIR &~(0x000000ff<<BUS\_NO);dat = (uint8)((IO0PIN&(0xFFFFFFFF))>>BUS\_NO)

/\* ¶¨ÒåREAD¿ØÖÆ \*/

#define LCM\_RD 12

#define LCM\_UNREAD() IO0SET = 1<<LCM\_RD

#define LCM\_READ() IO0CLR = 1<<LCM\_RD

/\* ¶¨ÒåWRITE¿ØÖÆ \*/

#define LCM\_WR 13

#define LCM\_UNWRITE() IO0SET = 1<<LCM\_WR

#define LCM\_WRITE() IO0CLR = 1<<LCM\_WR

/\* ¶¨ÒåC/D#¿ØÖÆ \*/

#define LCM\_CD 14

#define LCM\_COM() IO0SET = 1<<LCM\_CD

#define LCM\_DATA() IO0CLR = 1<<LCM\_CD

/\* ¶¨ÒåC/D#¿ØÖÆ \*/

#define LCM\_CE 15

#define LCM\_DISABLE() IO0SET = 1<<LCM\_CE

#define LCM\_ENABLE() IO0CLR = 1<<LCM\_CE

/\* ¶¨ÒåLCM²Ù×÷µÄÃüÁî×Ö \*/

// T6963C ÃüÁî¶¨Òå

#define LCM\_CUR\_POS 0x21 // ¹â±êÎ»ÖÃÉèÖÃ

#define LCM\_CGR\_POS 0x22 // CGRAM Æ«ÖÃµØÖ·ÉèÖÃ

#define LCM\_ADD\_POS 0x24 // µØÖ·Ö¸ÕëÎ»ÖÃ

#define LCM\_TXT\_STP 0x40 // ÎÄ±¾ÇøÊ×Ö·

#define LCM\_TXT\_WID 0x41 // ÎÄ±¾Çø¿í¶È

#define LCM\_GRH\_STP 0x42 // Í¼ÐÎÇøÊ×Ö·

#define LCM\_GRH\_WID 0x43 // Í¼ÐÎÇø¿í¶È

#define LCM\_MOD\_OR 0x80 // ÏÔÊ¾·½Ê½Âß¼­»ò

#define LCM\_MOD\_XOR 0x81 // ÏÔÊ¾·½Ê½Âß¼­Òì»ò

#define LCM\_MOD\_AND 0x82 // ÏÔÊ¾·½Ê½Âß¼­Óë

#define LCM\_MOD\_TCH 0x83 // ÏÔÊ¾·½Ê½ÎÄ±¾ÌØÕ÷

#define LCM\_DIS\_SW 0x90 // ÏÔÊ¾¿ª¹ØD0=1/0:¹â±êÉÁË¸ÆôÓÃ/½ûÓÃ

// D1=1/0:¹â±êÏÔÊ¾ÆôÓÃ/½ûÓÃ

// D2=1/0:ÎÄ±¾ÏÔÊ¾ÆôÓÃ/½ûÓÃ

// D3=1/0:Í¼ÐÎÏÔÊ¾ÆôÓÃ/½ûÓÃ

#define LCM\_CUR\_SHP 0xA0 // ¹â±êÐÎ×´Ñ¡Ôñ0xA0-0xA7±íÊ¾¹â±êÕ¼µÄÐÐÊý

#define LCM\_AUT\_WR 0xB0 // ×Ô¶¯Ð´ÉèÖÃ

#define LCM\_AUT\_RD 0xB1 // ×Ô¶¯¶ÁÉèÖÃ

#define LCM\_AUT\_OVR 0xB2 // ×Ô¶¯¶Á/Ð´½áÊø

#define LCM\_INC\_WR 0xC0 // Êý¾ÝÒ»´ÎÐ´µØÖ·¼Ó1

#define LCM\_INC\_RD 0xC1 // Êý¾ÝÒ»´Î¶ÁµØÖ·¼Ó1

#define LCM\_DEC\_WR 0xC2 // Êý¾ÝÒ»´ÎÐ´µØÖ·¼õ1

#define LCM\_DEC\_RD 0xC3 // Êý¾ÝÒ»´Î¶ÁµØÖ·¼õ1

#define LCM\_NOC\_WR 0xC4 // Êý¾ÝÒ»´ÎÐ´µØÖ·²»±ä

#define LCM\_NOC\_RD 0xC5 // Êý¾ÝÒ»´Î¶ÁµØÖ·²»±ä

#define LCM\_SCN\_RD 0xE0 // ÆÁ¶Á

#define LCM\_SCN\_CP 0xE8 // ÆÁ¿½±´

#define LCM\_BIT\_OP 0xF0 // Î»²Ù×÷

uint8 const turnf[8] = {7,6,5,4,3,2,1,0};

uint8 const DEC\_HEX\_TAB1[8] = {0x80, 0x40, 0x20, 0x10, 0x08, 0x04, 0x02, 0x01};

uint8 const DEC\_HEX\_TAB[8] = {0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80};

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: LCM\_READSTATE

\*\* ¹¦ÄÜÃèÊö: ¶ÁÈ¡LCMÄÚ²¿µÄ×´Ì¬

\*\* Êä¡¡Èë: ÎÞ

\*\* Êä¡¡³ö: LCMÄÚ²¿×´Ì¬Öµ

\*\* È«¾Ö±äÁ¿:

\*\* µ÷ÓÃÄ£¿é:

\*\* Modified by:

\*\* Modified date:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

uint8 LCM\_READSTATE()

{

uint8 dat;

IO0DIR &= ~(0x000000ff<<BUS\_NO);

LCM\_UNWRITE();

LCM\_COM();

LCM\_READ();

LCM\_ENABLE();

//DELAY5();

//DELAY5();

//DELAY5();

//InData();

dat = (uint8)((IO0PIN)>>BUS\_NO);

//LCM\_UNREAD();

//LCM\_UNWRITE();

LCM\_DISABLE();

return dat;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: LCM\_STA01

\*\* ¹¦ÄÜÃèÊö: ×´Ì¬Î»STA1,STA0ÅÐ¶Ï¶ÁÐ´Ö¸ÁîºÍ¶ÁÐ´Êý¾Ý£¬ÔÚ¶ÁÐ´Êý¾Ý»òÕßÐ´ÈëÃüÁîÇ°±ØÐë±£Ö¤¾ùÎª1

\*\* Êä¡¡Èë: ÎÞ

\*\* Êä¡¡³ö: ÎÞ

\*\* È«¾Ö±äÁ¿: ÎÞ

\*\* µ÷ÓÃÄ£¿é: LCM\_READSTATE

\*\* Modified by:

\*\* Modified date:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

uint8 LCM\_STA01(void)

{

uint8 i;

for(i=10;i>0;i--)

{

if(( LCM\_READSTATE() & 0x03) == 0x03) // ¶ÁÈ¡×´Ì¬

{

break;

}

}

return(i); // Èô·µ»ØÁãËµÃ÷´íÎó

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: LCM\_STA3

\*\* ¹¦ÄÜÃèÊö: ×´Ì¬Î»STA3

\*\* Êä¡¡Èë: ÎÞ

\*\* Êä¡¡³ö: ÎÞ

\*\* È«¾Ö±äÁ¿: ÎÞ

\*\* µ÷ÓÃÄ£¿é: LCM\_READSTATE

\*\* Modified by:

\*\* Modified date:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

uint8 LCM\_STA3(void)

{

uint8 i;

for(i=10;i>0;i--)

{

if(( LCM\_READSTATE() & 0x08) == 0x08) // ¶ÁÈ¡×´Ì¬

{

break;

}

}

return(i); // Èô·µ»ØÁãËµÃ÷´íÎó

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: LCM\_WrCommand

\*\* ¹¦ÄÜÃèÊö: Ð´ÃüÁî×Ó³ÌÐò

\*\* Êä¡¡Èë: command ÒªÐ´ÈëLCMµÄÃüÁî×Ö

\*\* Êä¡¡³ö: ÎÞ

\*\* È«¾Ö±äÁ¿: ÎÞ

\*\* µ÷ÓÃÄ£¿é: ÎÞ

\*\* Modified by:

\*\* Modified date:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void LCM\_WrCommand(uint8 command)

{

LCM\_UNREAD();

LCM\_COM();

LCM\_WRITE();

LCM\_ENABLE();

OutData(command);

//LCM\_UNWRITE();

//LCM\_READ();

LCM\_DISABLE();

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: LCM\_WrData

\*\* ¹¦ÄÜÃèÊö: Ð´Êý¾Ý×Ó³ÌÐò

\*\* Êä¡¡Èë: wrdata ÒªÐ´ÈëLCMµÄÊý¾Ý

\*\* Êä¡¡³ö: ÎÞ

\*\* È«¾Ö±äÁ¿: ÎÞ

\*\* µ÷ÓÃÄ£¿é: ÎÞ

\*\* Modified by:

\*\* Modified date:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void LCM\_WrData(uint8 wrdata)

{

LCM\_UNREAD();

LCM\_DATA();

LCM\_WRITE();

LCM\_ENABLE();

OutData(wrdata);

//LCM\_UNWRITE();

//LCM\_READ();

LCM\_DISABLE();

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: LCM\_WrParameter

\*\* ¹¦ÄÜÃèÊö: ÏòLCMÐ´Èë²ÎÊý£¬´øË«²ÎÊý£¬Ò»¸ö²ÎÊý£¬»òÕß²»´ø²ÎÊý

\*\* Êä¡¡Èë: cmd²ÎÊý£»para1²ÎÊý1£»para2²ÎÊý2£»num²ÎÊý¸öÊý

\*\* Êä¡¡³ö: ·µ»Ø²Ù×÷½á¹û

\*\* È«¾Ö±äÁ¿: ÎÞ

\*\* µ÷ÓÃÄ£¿é: ÎÞ

\*\* Modified by:

\*\* Modified date:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

uint8 LCM\_WrParameter(uint8 cmd,uint8 para1,uint8 para2,uint8 num)

{

switch (num)

{

case 0x00:

/\*

if(LCM\_STA01() == 0)

{

return 1;

}

\*/

LCM\_WrCommand(cmd);

break;

case 0x01:

/\*

if(LCM\_STA01() == 0)

{

return 1;

}

LCM\_WrData(para1);

if(LCM\_STA01() == 0)

{

return 2;

}

LCM\_WrCommand(cmd);

\*/

LCM\_WrData(para1);

LCM\_WrCommand(cmd);

break;

case 0x02:

/\*

if(LCM\_STA01() == 0)

{

return 1;

}

LCM\_WrData(para1);

if(LCM\_STA01() == 0)

{

return 2;

}

LCM\_WrData(para2);

if(LCM\_STA01() == 0)

{

return 3;

}

LCM\_WrCommand(cmd);

\*/

LCM\_WrData(para1);

LCM\_WrData(para2);

LCM\_WrCommand(cmd);

break;

}

return 0;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: LCM\_ReadByte

\*\* ¹¦ÄÜÃèÊö: ¶ÁÈ¡Ö¸¶¨µãÉÏµÄ×Ö½ÚÊý¾Ý

\*\* Êä¡¡Èë: x,y×ø±êÖµ

\*\* Êä¡¡³ö: ·µ»Ø¸ÃµãÉÏµÄ×Ö½ÚÊý¾Ý

\*\* È«¾Ö±äÁ¿: ÎÞ

\*\* µ÷ÓÃÄ£¿é: ÎÞ

\*\* Modified by:

\*\* Modified date:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

uint8 LCM\_ReadByte(uint8 x, uint8 y)

{

uint8 dat=0xff;

uint8 x1;

uint32 iPos;

x1 = x >> 3; // È¡Y·½Ïò·ÖÒ³µØÖ·

iPos = (uint32)y \* 0x1e + x1;

LCM\_WrParameter(LCM\_ADD\_POS,iPos&0xff,iPos/256,2);

LCM\_WrParameter(LCM\_NOC\_RD,0,0,0);

/\*

if(LCM\_STA01() == 0)

{

return 1;

}

\*/

IO0DIR = IO0DIR &~(0x000000ff<<BUS\_NO);

LCM\_UNWRITE();

LCM\_DATA();

LCM\_READ();

LCM\_ENABLE();

//InData();

dat = (uint8)((IO0PIN)>>BUS\_NO);

LCM\_DISABLE();

return dat;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: LCM\_DispIni

\*\* ¹¦ÄÜÃèÊö: LCMÓ²¼þ³õÊ¼»¯

\*\* Êä¡¡Èë: ÎÞ

\*\* Êä¡¡³ö: ÎÞ

\*\* È«¾Ö±äÁ¿: ÎÞ

\*\* µ÷ÓÃÄ£¿é: ÎÞ

\*\* Modified by:

\*\* Modified date:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void LCM\_DispIni(void)

{

uint32 i;

// ÉèÖÃÒý½ÅÁ¬½ÓÄ£¿é

#if LCM\_RD < 16

PINSEL0 &= ~(3 << (2 \* LCM\_RD));

#else

PINSEL1 &= ~(3 << (2 \* (LCM\_RD - 16)));

#endif

#if LCM\_WR < 16

PINSEL0 &= ~(3 << (2 \* LCM\_WR));

#else

PINSEL1 &= ~(3 << (2 \* (LCM\_WR - 16)));

#endif

#if LCM\_CD < 16

PINSEL0 &= ~(3 << (2 \* LCM\_CD));

#else

PINSEL1 &= ~(3 << (2 \* (LCM\_CD - 16)));

#endif

#if BUS\_NO<9

for (i = BUS\_NO; i < BUS\_NO+8; i++)

{

PINSEL0 &= ~(3 << (2 \* i));

}

#else

for (i = BUS\_NO; i < 16; i++)

{

PINSEL0 &= ~(3 << (2 \* i));

}

for (; i < (BUS\_NO+8); i++)

{

PINSEL1 &= ~(3 << (2 \* (i-16)));

}

#endif

// ÉèÖÃI/OÎªÊä³ö·½Ê½

IO0DIR = IO0DIR|(1<<LCM\_RD)|(1<<LCM\_WR)|(1<<LCM\_CD)|(1<<LCM\_CE);

IO0DIR = IO0DIR|(0xFF<<BUS\_NO);

LCM\_WrParameter(LCM\_TXT\_STP,0x00,0x00,2);

LCM\_WrParameter(LCM\_TXT\_WID,0x1E,0x00,2);

LCM\_WrParameter(LCM\_GRH\_STP,0x00,0x00,2);

LCM\_WrParameter(LCM\_GRH\_WID,0x1E,0x00,2);

LCM\_WrParameter(LCM\_CUR\_SHP|0x01,0,0,0);

LCM\_WrParameter(LCM\_MOD\_OR,0,0,0);

LCM\_WrParameter(LCM\_DIS\_SW|0x08,0,0,0);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: GUI\_FillSCR()

\*\* ¹¦ÄÜÃèÊö: È«ÆÁÌî³ä¡£Ö±½ÓÊ¹ÓÃÊý¾ÝÌî³äÏÔÊ¾»º³åÇø¡£¸ù¾ÝLCMµÄÊµ¼ÊÇé¿ö±àÐ´´Ëº¯Êý

\*\* Êä¡¡Èë: dat Ìî³äµÄÊý¾Ý

\*\* Êä¡¡³ö: ÎÞ

\*\* È«¾Ö±äÁ¿: ÎÞ

\*\* µ÷ÓÃÄ£¿é: ÎÞ

\*\* Modified by:

\*\* Modified date:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void GUI\_FillSCR(TCOLOR dat)

{

uint32 i;

LCM\_WrParameter(LCM\_ADD\_POS,0x00,0x00,2);

LCM\_WrParameter(LCM\_AUT\_WR,0x00,0x00,0);

for(i=0;i<240\*128/8;i++)

{

//LCM\_STA3();

LCM\_WrData(dat);

}

LCM\_WrParameter(LCM\_AUT\_OVR,0x00,0x00,0);

LCM\_WrParameter(LCM\_ADD\_POS,0x00,0x00,2);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: GUI\_Initialize

\*\* ¹¦ÄÜÃèÊö: ³õÊ¼»¯GUI£¬°üÀ¨³õÊ¼»¯ÏÔÊ¾»º³åÇø£¬³õÊ¼»¯LCM²¢ÇåÆÁ

\*\* Êä¡¡Èë: ÎÞ

\*\* Êä¡¡³ö: ÎÞ

\*\* È«¾Ö±äÁ¿: ÎÞ

\*\* µ÷ÓÃÄ£¿é: ÎÞ

\*\* Modified by:

\*\* Modified date:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void GUI\_Initialize(void)

{

LCM\_DispIni(); // ³õÊ¼»¯LCMÄ£¿é¹¤×÷Ä£Ê½£¬´¿Í¼ÐÎÄ£Ê½

GUI\_FillSCR(0x00); // ³õÊ¼»¯»º³åÇøÎª0x00£¬²¢Êä³öÆÁÄ»(ÇåÆÁ)

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: GUI\_Point

\*\* ¹¦ÄÜÃèÊö: ÔÚÖ¸¶¨Î»ÖÃÉÏ»­µã

\*\* Êä¡¡Èë: xÖ¸¶¨µãËùÔÚÁÐµÄÎ»ÖÃ£»yÖ¸¶¨µãËùÔÚÐÐµÄÎ»ÖÃ£»colorÏÔÊ¾ÑÕÉ«(¶ÔÓÚºÚ°×É«LCM£¬Îª0Ê±Ãð£¬Îª1Ê±ÏÔÊ¾)

\*\* Êä¡¡³ö: ·µ»ØÖµÎª1Ê±±íÊ¾²Ù×÷³É¹¦£¬Îª0Ê±±íÊ¾²Ù×÷Ê§°Ü

\*\* È«¾Ö±äÁ¿: ÎÞ

\*\* µ÷ÓÃÄ£¿é: ÎÞ

\*\* Modified by:

\*\* Modified date:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

uint8 GUI\_Point(uint8 x, uint8 y, TCOLOR color)

{

uint8 x1;

uint32 iPos;

x1 = x >> 3; // È¡Y·½Ïò·ÖÒ³µØÖ·,ÒòÎª×îÐ¡´æ´¢µ¥ÔªÎª8\*8,°´8ÐÐÒ»¸öµ¥Ôª·ÃÎÊ

iPos = (uint32)y \* 0x1e + x1;//¼ÆËãµØÖ·:0xleÊÇÎÄ±¾µÄ¿í¶È,¸ß¶È

LCM\_WrParameter(LCM\_ADD\_POS,iPos&0xff,iPos/256,2);//·Ö±ðÈ¡³öµÍµØÖ·£¬¸ßµØÖ·;Ð´ÈëLCD

x1 = turnf[ x & 0x07 ];//¼ÆËã¾ßÌåµÄÐÐ

//uint8 const turnf[8] = {7,6,5,4,3,2,1,0};

color = color <<3;

x1 = LCM\_BIT\_OP|x1|color; // ×Ö½ÚÄÚÎ»ÖÃ¼ÆËã,LCM\_BIT\_OPÎªÎ»²Ù×÷Ö¸Áî

/\*Î»²Ù×÷£º

1 1 1 1 N3 N2 N2 N0

ÎÞ²ÎÊý

¸ÃÖ¸Áî¿É½«ÏÔÊ¾»º³åÇøÄ³µ¥ÔªµÄÄ³Ò»Î»ÇåÁã»òÖÃ1£¬¸Ãµ¥ÔªµØÖ·ÓÉµ±Ç°µØÖ·Ö¸ÕëÌá¹©¡£

N3£½1ÖÃ1£¬N3£½0 ÇåÁã¡£N2£­N0£º²Ù×÷Î»¶ÔÓ¦µ¥ÔªµÄD0£­D7Î»¡£\*/

LCM\_WrParameter(x1,0,0,0);;

return 1;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: GUI\_ReadPoint

\*\* ¹¦ÄÜÃèÊö: ¶ÁÈ¡Ö¸¶¨µãµÄÑÕÉ«¡£¶ÔÓÚµ¥É«£¬ÉèÖÃretµÄd0Î»Îª1»ò0£¬4¼¶»Ò¶ÈÔòÎªd0¡¢d1ÓÐÐ§£¬8Î»RGBÔòd0--d7ÓÐÐ§£¬RGB½á¹¹ÔòR¡¢G¡¢B±äÁ¿ÓÐÐ§

\*\* Êä¡¡Èë: xÖ¸¶¨µãËùÔÚÁÐµÄÎ»ÖÃ£» yÖ¸¶¨µãËùÔÚÐÐµÄÎ»ÖÃ£»ret±£´æÑÕÉ«ÖµµÄÖ¸Õë

\*\* Êä¡¡³ö: ·µ»Ø0±íÊ¾Ö¸¶¨µØÖ·³¬³ö»º³åÇø·¶Î§

\*\* È«¾Ö±äÁ¿: ÎÞ

\*\* µ÷ÓÃÄ£¿é: ÎÞ

\*\* Modified by:

\*\* Modified date:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

uint8 GUI\_ReadPoint(uint8 x, uint8 y, TCOLOR \*ret)

{

TCOLOR bak;

uint8 x1;

bak = LCM\_ReadByte(x,y);

x1 = turnf[ x & 0x07 ];

if( (bak & (DEC\_HEX\_TAB[x1&0x07]) ) ==0)

\*ret = 0x00;

else

\*ret = 0x01;

return 1;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: GUI\_HLine

\*\* ¹¦ÄÜÃèÊö: »­Ë®Æ½Ïß£¬²Ù×÷Ê§°ÜÔ­ÒòÊÇÖ¸¶¨µØÖ·³¬³ö»º³åÇø·¶Î§

\*\* Êä¡¡Èë: x0 Ë®Æ½ÏßÆðµãËùÔÚÁÐµÄÎ»ÖÃ

\* y0 Ë®Æ½ÏßÆðµãËùÔÚÐÐµÄÎ»ÖÃ

\* x1 Ë®Æ½ÏßÖÕµãËùÔÚÁÐµÄÎ»ÖÃ

\* color ÏÔÊ¾ÑÕÉ«(¶ÔÓÚºÚ°×É«LCM£¬Îª0Ê±Ãð£¬Îª1Ê±ÏÔÊ¾)

\*\* Êä¡¡³ö: ÎÞ

\*\* È«¾Ö±äÁ¿: ÎÞ

\*\* µ÷ÓÃÄ£¿é: ÎÞ

\*\* Modified by:

\*\* Modified date:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void GUI\_HLine(uint8 x0, uint8 y0, uint8 x1, TCOLOR color)

{ uint8 bak;

if(x0>x1) // ¶Ôx0¡¢x1´óÐ¡½øÐÐÅÅÁÐ£¬ÒÔ±ã»­Í¼

{ bak = x1;

x1 = x0;

x0 = bak;

}

do

{ GUI\_Point(x0, y0, color); // ÖðµãÏÔÊ¾£¬Ãè³ö´¹Ö±Ïß

x0++;

}while(x1>=x0);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: GUI\_RLine

\*\* ¹¦ÄÜÃèÊö: »­ÊúÖ±Ïß¡£

\*\* Êä¡¡Èë: x0 Ë®Æ½ÏßÆðµãËùÔÚÁÐµÄÎ»ÖÃ

\* y0 Ë®Æ½ÏßÆðµãËùÔÚÐÐµÄÎ»ÖÃ

\* x1 Ë®Æ½ÏßÖÕµãËùÔÚÁÐµÄÎ»ÖÃ

\* color ÏÔÊ¾ÑÕÉ«(¶ÔÓÚºÚ°×É«LCM£¬Îª0Ê±Ãð£¬Îª1Ê±ÏÔÊ¾)

\*\* Êä¡¡³ö: ÎÞ

\*\* È«¾Ö±äÁ¿: ÎÞ

\*\* µ÷ÓÃÄ£¿é: ÎÞ

\*\* Modified by:

\*\* Modified date:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void GUI\_RLine(uint8 x0, uint8 y0, uint8 y1, TCOLOR color)

{ uint8 bak;

if(y0>y1) // ¶Ôx0¡¢x1´óÐ¡½øÐÐÅÅÁÐ£¬ÒÔ±ã»­Í¼

{ bak = y1;

y1 = y0;

y0 = bak;

}

do

{ GUI\_Point(x0, y0, color); // ÖðµãÏÔÊ¾£¬Ãè³ö´¹Ö±Ïß

y0++;

}while(y1>=y0);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* End Of File

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**6.3 target.c**

#define IN\_TARGET

#include "config.h"

void \_\_irq IRQ\_Exception(void)

{

while(1);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void FIQ\_Exception(void)

{

while(1);

}

/\*\*\*\*\*\*\*\*\*\*\* Target limit \*\*\*\*\*\*\*\*\*\*\*/

void TargetInit(void)

{

}

void TargetResetInit(void)

{

MAMCR=2;

#if Fcclk < 20000000

MAMTIM=1;

#else

#if Fcclk < 40000000

MAMTIM=2;

#else

MAMTIM= 3;

#endif

#endif

VICIntEnClr=0xffffffff;

VICVectAddr=0;

VICIntSelect=0;

}

#include "rt\_sys.h"

#include "stdio.h"

#pragma import(\_\_use\_no\_semihosting\_swi)

#pragma import(\_\_use\_two\_region\_memory)

int \_\_rt\_div0(int a)

{

a = a;

return 0;

}

int fputc(int ch,FILE\*f)

{

ch = ch;

f = f;

return 0;

}

int fgetc(FILE\*f)

{

f = f;

return 0;

}

int \_sys\_close(FILEHANDLE fh)

{

fh = fh;

return 0;

}

int \_sys\_write(FILEHANDLE fh,const unsigned char \*buf,

unsigned len, int mode)

{

fh = fh;

buf = buf;

len =len;

mode = mode;

return 0;

}

int \_sys\_read(FILEHANDLE fh, unsigned char \*buf,

unsigned len, int mode)

{

fh = fh;

buf = buf;

len =len;

mode = mode;

return 0;

}

void \_ttywrch(int ch)

{

ch=ch;

}

int \_sys\_istty(FILEHANDLE fh)

{

fh = fh;

return 0;

}

int \_sys\_seek(FILEHANDLE fh,long pos)

{

fh = fh;

return 0;

}

int \_sys\_ensure(FILEHANDLE fh)

{

fh = fh;

return 0;

}

long \_sys\_flen(FILEHANDLE fh)

{

fh =fh;

return 0;

}

int \_sys\_tmpnam(char \* name, int sig, unsigned maxlen)

{

name = name;

sig = sig;

maxlen=maxlen;

return 0;

}

void \_sys\_exit(int returncode)

{

returncode = returncode;

}

char\* \_sys\_command\_string(char \* cmd, int len)

{

cmd = cmd;

len = len;

return 0;

}

**6.4 Startup.s**

;/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

;/\* STARTUP.S: Startup file for Philips LPC2000 \*/

;/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

;/\* <<< Use Configuration Wizard in Context Menu >>> \*/

;/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

;/\* This file is part of the uVision/ARM development tools. \*/

;/\* Copyright (c) 2005-2007 Keil Software. All rights reserved. \*/

;/\* This software may only be used under the terms of a valid, current, \*/

;/\* end user licence from KEIL for a compatible version of KEIL software \*/

;/\* development tools. Nothing else gives you the right to use this software. \*/

;/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

;/\*

; \* The STARTUP.S code is executed after CPU Reset. This file may be

; \* translated with the following SET symbols. In uVision these SET

; \* symbols are entered under Options - ASM - Define.

; \*

; \* REMAP: when set the startup code initializes the register MEMMAP

; \* which overwrites the settings of the CPU configuration pins. The

; \* startup and interrupt vectors are remapped from:

; \* 0x00000000 default setting (not remapped)

; \* 0x80000000 when EXTMEM\_MODE is used

; \* 0x40000000 when RAM\_MODE is used

; \*

; \* EXTMEM\_MODE: when set the device is configured for code execution

; \* from external memory starting at address 0x80000000.

; \*

; \* RAM\_MODE: when set the device is configured for code execution

; \* from on-chip RAM starting at address 0x40000000.

; \*

; \* EXTERNAL\_MODE: when set the PIN2SEL values are written that enable

; \* the external BUS at startup.

; \*/

; Standard definitions of Mode bits and Interrupt (I & F) flags in PSRs

Mode\_USR EQU 0x10

Mode\_FIQ EQU 0x11

Mode\_IRQ EQU 0x12

Mode\_SVC EQU 0x13

Mode\_ABT EQU 0x17

Mode\_UND EQU 0x1B

Mode\_SYS EQU 0x1F

I\_Bit EQU 0x80 ; when I bit is set, IRQ is disabled

F\_Bit EQU 0x40 ; when F bit is set, FIQ is disabled

;// <h> Stack Configuration (Stack Sizes in Bytes)

;// <o0> Undefined Mode <0x0-0xFFFFFFFF:8>

;// <o1> Supervisor Mode <0x0-0xFFFFFFFF:8>

;// <o2> Abort Mode <0x0-0xFFFFFFFF:8>

;// <o3> Fast Interrupt Mode <0x0-0xFFFFFFFF:8>

;// <o4> Interrupt Mode <0x0-0xFFFFFFFF:8>

;// <o5> User/System Mode <0x0-0xFFFFFFFF:8>

;// </h>

UND\_Stack\_Size EQU 0x00000000

SVC\_Stack\_Size EQU 0x00000008

ABT\_Stack\_Size EQU 0x00000000

FIQ\_Stack\_Size EQU 0x00000000

IRQ\_Stack\_Size EQU 0x00000080

USR\_Stack\_Size EQU 0x00000400

ISR\_Stack\_Size EQU (UND\_Stack\_Size + SVC\_Stack\_Size + ABT\_Stack\_Size + \

FIQ\_Stack\_Size + IRQ\_Stack\_Size)

AREA STACK, NOINIT, READWRITE, ALIGN=3

Stack\_Mem SPACE USR\_Stack\_Size

\_\_initial\_sp SPACE ISR\_Stack\_Size

Stack\_Top

;// <h> Heap Configuration

;// <o> Heap Size (in Bytes) <0x0-0xFFFFFFFF>

;// </h>

Heap\_Size EQU 0x00000000

AREA HEAP, NOINIT, READWRITE, ALIGN=3

\_\_heap\_base

Heap\_Mem SPACE Heap\_Size

\_\_heap\_limit

; VPBDIV definitions

VPBDIV EQU 0xE01FC100 ; VPBDIV Address

;// <e> VPBDIV Setup

;// <i> Peripheral Bus Clock Rate

;// <o1.0..1> VPBDIV: VPB Clock

;// <0=> VPB Clock = CPU Clock / 4

;// <1=> VPB Clock = CPU Clock

;// <2=> VPB Clock = CPU Clock / 2

;// <o1.4..5> XCLKDIV: XCLK Pin

;// <0=> XCLK Pin = CPU Clock / 4

;// <1=> XCLK Pin = CPU Clock

;// <2=> XCLK Pin = CPU Clock / 2

;// </e>

VPBDIV\_SETUP EQU 0

VPBDIV\_Val EQU 0x00000000

; Phase Locked Loop (PLL) definitions

PLL\_BASE EQU 0xE01FC080 ; PLL Base Address

PLLCON\_OFS EQU 0x00 ; PLL Control Offset

PLLCFG\_OFS EQU 0x04 ; PLL Configuration Offset

PLLSTAT\_OFS EQU 0x08 ; PLL Status Offset

PLLFEED\_OFS EQU 0x0C ; PLL Feed Offset

PLLCON\_PLLE EQU (1<<0) ; PLL Enable

PLLCON\_PLLC EQU (1<<1) ; PLL Connect

PLLCFG\_MSEL EQU (0x1F<<0) ; PLL Multiplier

PLLCFG\_PSEL EQU (0x03<<5) ; PLL Divider

PLLSTAT\_PLOCK EQU (1<<10) ; PLL Lock Status

;// <e> PLL Setup

;// <o1.0..4> MSEL: PLL Multiplier Selection

;// <1-32><#-1>

;// <i> M Value

;// <o1.5..6> PSEL: PLL Divider Selection

;// <0=> 1 <1=> 2 <2=> 4 <3=> 8

;// <i> P Value

;// </e>

PLL\_SETUP EQU 1

PLLCFG\_Val EQU 0x00000024

; Memory Accelerator Module (MAM) definitions

MAM\_BASE EQU 0xE01FC000 ; MAM Base Address

MAMCR\_OFS EQU 0x00 ; MAM Control Offset

MAMTIM\_OFS EQU 0x04 ; MAM Timing Offset

;// <e> MAM Setup

;// <o1.0..1> MAM Control

;// <0=> Disabled

;// <1=> Partially Enabled

;// <2=> Fully Enabled

;// <i> Mode

;// <o2.0..2> MAM Timing

;// <0=> Reserved <1=> 1 <2=> 2 <3=> 3

;// <4=> 4 <5=> 5 <6=> 6 <7=> 7

;// <i> Fetch Cycles

;// </e>

MAM\_SETUP EQU 1

MAMCR\_Val EQU 0x00000002

MAMTIM\_Val EQU 0x00000004

; External Memory Controller (EMC) definitions

EMC\_BASE EQU 0xFFE00000 ; EMC Base Address

BCFG0\_OFS EQU 0x00 ; BCFG0 Offset

BCFG1\_OFS EQU 0x04 ; BCFG1 Offset

BCFG2\_OFS EQU 0x08 ; BCFG2 Offset

BCFG3\_OFS EQU 0x0C ; BCFG3 Offset

;// <e> External Memory Controller (EMC)

EMC\_SETUP EQU 0

;// <e> Bank Configuration 0 (BCFG0)

;// <o1.0..3> IDCY: Idle Cycles <0-15>

;// <o1.5..9> WST1: Wait States 1 <0-31>

;// <o1.11..15> WST2: Wait States 2 <0-31>

;// <o1.10> RBLE: Read Byte Lane Enable

;// <o1.26> WP: Write Protect

;// <o1.27> BM: Burst ROM

;// <o1.28..29> MW: Memory Width <0=> 8-bit <1=> 16-bit

;// <2=> 32-bit <3=> Reserved

;// </e>

BCFG0\_SETUP EQU 0

BCFG0\_Val EQU 0x0000FBEF

;// <e> Bank Configuration 1 (BCFG1)

;// <o1.0..3> IDCY: Idle Cycles <0-15>

;// <o1.5..9> WST1: Wait States 1 <0-31>

;// <o1.11..15> WST2: Wait States 2 <0-31>

;// <o1.10> RBLE: Read Byte Lane Enable

;// <o1.26> WP: Write Protect

;// <o1.27> BM: Burst ROM

;// <o1.28..29> MW: Memory Width <0=> 8-bit <1=> 16-bit

;// <2=> 32-bit <3=> Reserved

;// </e>

BCFG1\_SETUP EQU 0

BCFG1\_Val EQU 0x0000FBEF

;// <e> Bank Configuration 2 (BCFG2)

;// <o1.0..3> IDCY: Idle Cycles <0-15>

;// <o1.5..9> WST1: Wait States 1 <0-31>

;// <o1.11..15> WST2: Wait States 2 <0-31>

;// <o1.10> RBLE: Read Byte Lane Enable

;// <o1.26> WP: Write Protect

;// <o1.27> BM: Burst ROM

;// <o1.28..29> MW: Memory Width <0=> 8-bit <1=> 16-bit

;// <2=> 32-bit <3=> Reserved

;// </e>

BCFG2\_SETUP EQU 0

BCFG2\_Val EQU 0x0000FBEF

;// <e> Bank Configuration 3 (BCFG3)

;// <o1.0..3> IDCY: Idle Cycles <0-15>

;// <o1.5..9> WST1: Wait States 1 <0-31>

;// <o1.11..15> WST2: Wait States 2 <0-31>

;// <o1.10> RBLE: Read Byte Lane Enable

;// <o1.26> WP: Write Protect

;// <o1.27> BM: Burst ROM

;// <o1.28..29> MW: Memory Width <0=> 8-bit <1=> 16-bit

;// <2=> 32-bit <3=> Reserved

;// </e>

BCFG3\_SETUP EQU 0

BCFG3\_Val EQU 0x0000FBEF

;// </e> End of EMC

; External Memory Pins definitions

PINSEL2 EQU 0xE002C014 ; PINSEL2 Address

PINSEL2\_Val EQU 0x0E6149E4 ; CS0..3, OE, WE, BLS0..3,

; D0..31, A2..23, JTAG Pins

PRESERVE8

; Area Definition and Entry Point

; Startup Code must be linked first at Address at which it expects to run.

AREA RESET, CODE, READONLY

ARM

; Exception Vectors

; Mapped to Address 0.

; Absolute addressing mode must be used.

; Dummy Handlers are implemented as infinite loops which can be modified.

Vectors LDR PC, Reset\_Addr

LDR PC, Undef\_Addr

LDR PC, SWI\_Addr

LDR PC, PAbt\_Addr

LDR PC, DAbt\_Addr

NOP ; Reserved Vector

; LDR PC, IRQ\_Addr

LDR PC, [PC, #-0x0FF0] ; Vector from VicVectAddr

LDR PC, FIQ\_Addr

Reset\_Addr DCD Reset\_Handler

Undef\_Addr DCD Undef\_Handler

SWI\_Addr DCD SWI\_Handler

PAbt\_Addr DCD PAbt\_Handler

DAbt\_Addr DCD DAbt\_Handler

DCD 0 ; Reserved Address

IRQ\_Addr DCD IRQ\_Handler

FIQ\_Addr DCD FIQ\_Handler

Undef\_Handler B Undef\_Handler

SWI\_Handler B SWI\_Handler

PAbt\_Handler B PAbt\_Handler

DAbt\_Handler B DAbt\_Handler

IRQ\_Handler B IRQ\_Handler

FIQ\_Handler B FIQ\_Handler

; Reset Handler

EXPORT Reset\_Handler

Reset\_Handler

; Setup External Memory Pins

IF :DEF:EXTERNAL\_MODE

LDR R0, =PINSEL2

LDR R1, =PINSEL2\_Val

STR R1, [R0]

ENDIF

; Setup External Memory Controller

IF EMC\_SETUP <> 0

LDR R0, =EMC\_BASE

IF BCFG0\_SETUP <> 0

LDR R1, =BCFG0\_Val

STR R1, [R0, #BCFG0\_OFS]

ENDIF

IF BCFG1\_SETUP <> 0

LDR R1, =BCFG1\_Val

STR R1, [R0, #BCFG1\_OFS]

ENDIF

IF BCFG2\_SETUP <> 0

LDR R1, =BCFG2\_Val

STR R1, [R0, #BCFG2\_OFS]

ENDIF

IF BCFG3\_SETUP <> 0

LDR R1, =BCFG3\_Val

STR R1, [R0, #BCFG3\_OFS]

ENDIF

ENDIF ; EMC\_SETUP

; Setup VPBDIV

IF VPBDIV\_SETUP <> 0

LDR R0, =VPBDIV

LDR R1, =VPBDIV\_Val

STR R1, [R0]

ENDIF

; Setup PLL

IF PLL\_SETUP <> 0

LDR R0, =PLL\_BASE

MOV R1, #0xAA

MOV R2, #0x55

; Configure and Enable PLL

MOV R3, #PLLCFG\_Val

STR R3, [R0, #PLLCFG\_OFS]

MOV R3, #PLLCON\_PLLE

STR R3, [R0, #PLLCON\_OFS]

STR R1, [R0, #PLLFEED\_OFS]

STR R2, [R0, #PLLFEED\_OFS]

; Wait until PLL Locked

PLL\_Loop LDR R3, [R0, #PLLSTAT\_OFS]

ANDS R3, R3, #PLLSTAT\_PLOCK

BEQ PLL\_Loop

; Switch to PLL Clock

MOV R3, #(PLLCON\_PLLE:OR:PLLCON\_PLLC)

STR R3, [R0, #PLLCON\_OFS]

STR R1, [R0, #PLLFEED\_OFS]

STR R2, [R0, #PLLFEED\_OFS]

ENDIF ; PLL\_SETUP

; Setup MAM

IF MAM\_SETUP <> 0

LDR R0, =MAM\_BASE

MOV R1, #MAMTIM\_Val

STR R1, [R0, #MAMTIM\_OFS]

MOV R1, #MAMCR\_Val

STR R1, [R0, #MAMCR\_OFS]

ENDIF ; MAM\_SETUP

; Memory Mapping (when Interrupt Vectors are in RAM)

MEMMAP EQU 0xE01FC040 ; Memory Mapping Control

IF :DEF:REMAP

LDR R0, =MEMMAP

IF :DEF:EXTMEM\_MODE

MOV R1, #3

ELIF :DEF:RAM\_MODE

MOV R1, #2

ELSE

MOV R1, #1

ENDIF

STR R1, [R0]

ENDIF

; Initialise Interrupt System

; ...

; Setup Stack for each mode

LDR R0, =Stack\_Top

; Enter Undefined Instruction Mode and set its Stack Pointer

MSR CPSR\_c, #Mode\_UND:OR:I\_Bit:OR:F\_Bit

MOV SP, R0

SUB R0, R0, #UND\_Stack\_Size

; Enter Abort Mode and set its Stack Pointer

MSR CPSR\_c, #Mode\_ABT:OR:I\_Bit:OR:F\_Bit

MOV SP, R0

SUB R0, R0, #ABT\_Stack\_Size

; Enter FIQ Mode and set its Stack Pointer

MSR CPSR\_c, #Mode\_FIQ:OR:I\_Bit:OR:F\_Bit

MOV SP, R0

SUB R0, R0, #FIQ\_Stack\_Size

; Enter IRQ Mode and set its Stack Pointer

MSR CPSR\_c, #Mode\_IRQ:OR:I\_Bit:OR:F\_Bit

MOV SP, R0

SUB R0, R0, #IRQ\_Stack\_Size

; Enter Supervisor Mode and set its Stack Pointer

MSR CPSR\_c, #Mode\_SVC:OR:I\_Bit:OR:F\_Bit

MOV SP, R0

SUB R0, R0, #SVC\_Stack\_Size

; Enter User Mode and set its Stack Pointer

MSR CPSR\_c, #Mode\_USR

IF :DEF:\_\_MICROLIB

EXPORT \_\_initial\_sp

ELSE

MOV SP, R0

SUB SL, SP, #USR\_Stack\_Size

ENDIF

; Enter the C code

IMPORT \_\_main

LDR R0, =\_\_main

BX R0

IF :DEF:\_\_MICROLIB

EXPORT \_\_heap\_base

EXPORT \_\_heap\_limit

ELSE

; User Initial Stack & Heap

AREA |.text|, CODE, READONLY

IMPORT \_\_use\_two\_region\_memory

EXPORT \_\_user\_initial\_stackheap

\_\_user\_initial\_stackheap

LDR R0, = Heap\_Mem

LDR R1, =(Stack\_Mem + USR\_Stack\_Size)

LDR R2, = (Heap\_Mem + Heap\_Size)

LDR R3, = Stack\_Mem

BX LR

ENDIF

END

**6.5 config.h**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Copyright (c)\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* Guangzou ZLG-MCU Development Co.,LTD.

\*\* graduate school

\*\* http://www.zlgmcu.com

\*\*

\*\*--------------File Info-------------------------------------------------------------------------------

\*\* File Name: config.h

\*\* Last modified Date: 2004-09-17

\*\* Last Version: 1.0

\*\* Descriptions: User Configurable File

\*\*

\*\*------------------------------------------------------------------------------------------------------

\*\* Created By: Chenmingji

\*\* Created date: 2004-09-17

\*\* Version: 1.0

\*\* Descriptions: First version

\*\*

\*\*------------------------------------------------------------------------------------------------------

\*\* Modified by:

\*\* Modified date:

\*\* Version:

\*\* Descriptions:

\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#ifndef \_\_CONFIG\_H

#define \_\_CONFIG\_H

//ÕâÒ»¶ÎÎÞÐè¸Ä¶¯

//This segment should not be modified

#ifndef TRUE

#define TRUE 1

#endif

#ifndef FALSE

#define FALSE 0

#endif

typedef unsigned char uint8; /\* defined for unsigned 8-bits integer variable ÎÞ·ûºÅ8Î»ÕûÐÍ±äÁ¿ \*/

typedef signed char int8; /\* defined for signed 8-bits integer variable ÓÐ·ûºÅ8Î»ÕûÐÍ±äÁ¿ \*/

typedef unsigned short uint16; /\* defined for unsigned 16-bits integer variable ÎÞ·ûºÅ16Î»ÕûÐÍ±äÁ¿ \*/

typedef signed short int16; /\* defined for signed 16-bits integer variable ÓÐ·ûºÅ16Î»ÕûÐÍ±äÁ¿ \*/

typedef unsigned int uint32; /\* defined for unsigned 32-bits integer variable ÎÞ·ûºÅ32Î»ÕûÐÍ±äÁ¿ \*/

typedef signed int int32; /\* defined for signed 32-bits integer variable ÓÐ·ûºÅ32Î»ÕûÐÍ±äÁ¿ \*/

typedef float fp32; /\* single precision floating point variable (32bits) µ¥¾«¶È¸¡µãÊý£¨32Î»³¤¶È£© \*/

typedef double fp64; /\* double precision floating point variable (64bits) Ë«¾«¶È¸¡µãÊý£¨64Î»³¤¶È£© \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* uC/OS-II specital code \*/

/\* uC/OS-IIµÄÌØÊâ´úÂë \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define USER\_USING\_MODE 0x10 /\* User mode ,ARM 32BITS CODE ÓÃ»§Ä£Ê½,ARM´úÂë \*/

//

/\* Chosen one from 0x10,0x30,0x1f,0x3f.Ö»ÄÜÊÇ0x10,0x30,0x1f,0x3fÖ®Ò» \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* ARMµÄÌØÊâ´úÂë \*/

/\* ARM specital code \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//ÕâÒ»¶ÎÎÞÐè¸Ä¶¯

//This segment should not be modify

#include "LPC2124.h"

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Ó¦ÓÃ³ÌÐòÅäÖÃ \*/

/\*Application Program Configurations\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//ÒÔÏÂ¸ù¾ÝÐèÒª¸Ä¶¯

//This segment could be modified as needed.

#include <stdio.h>

#include <ctype.h>

#include <stdlib.h>

#include <setjmp.h>

#include <rt\_misc.h>

#include <math.h>

/\*

#include "LCMDRV.h"

#include "LOADBIT.H"

#include "GUI\_StockC.h"

#include "GUI\_CONFIG.H"

#include "keyboard.h"

\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* ±¾Àý×ÓµÄÅäÖÃ \*/

/\*Configuration of the example \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* System configuration .Fosc¡¢Fcclk¡¢Fcco¡¢Fpclk must be defined \*/

/\* ÏµÍ³ÉèÖÃ, Fosc¡¢Fcclk¡¢Fcco¡¢Fpclk±ØÐë¶¨Òå\*/

#define Fosc 11059200 //Crystal frequence,10MHz~25MHz£¬should be the same as actual status.

//Ó¦µ±ÓëÊµ¼ÊÒ»ÖÁ¾§ÕñÆµÂÊ,10MHz~25MHz£¬Ó¦µ±ÓëÊµ¼ÊÒ»ÖÁ

#define Fcclk (Fosc \* 4) //System frequence,should be (1~32)multiples of Fosc,and should be equal or less than 60MHz.

//ÏµÍ³ÆµÂÊ£¬±ØÐëÎªFoscµÄÕûÊý±¶(1~32)£¬ÇÒ<=60MHZ

#define Fcco (Fcclk \* 4) //CCO frequence,should be 2¡¢4¡¢8¡¢16 multiples of Fcclk, ranged from 156MHz to 320MHz.

//CCOÆµÂÊ£¬±ØÐëÎªFcclkµÄ2¡¢4¡¢8¡¢16±¶£¬·¶Î§Îª156MHz~320MHz

#define Fpclk (Fcclk / 4) \* 1 //VPB clock frequence , must be 1¡¢2¡¢4 multiples of (Fcclk / 4).

//VPBÊ±ÖÓÆµÂÊ£¬Ö»ÄÜÎª(Fcclk / 4)µÄ1¡¢2¡¢4±¶

#include "target.h" //This line may not be deleted ÕâÒ»¾ä²»ÄÜÉ¾³ý

#endif

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* End Of File

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**6.6 lcddrv.h**

/\* ¶¨ÒåÑÕÉ«Êý¾ÝÀàÐÍ(¿ÉÒÔÊÇÊý¾Ý½á¹¹) \*/

#define TCOLOR uint8

#define LCD\_DISP\_COLOR 1 //marco¶¨ÒåLCDµÄÇ°¾°É«ºÍ±³¾°É«

#define LCD\_BACK\_COLOR 0

/\* ¶¨ÒåLCMÏñËØÊýºê \*/

#define GUI\_LCM\_XMAX 256 //240 /\* ¶¨ÒåÒº¾§xÖáµÄÏñËØÊý \*/

#define GUI\_LCM\_YMAX 64 //128 /\* ¶¨ÒåÒº¾§yÖáµÄÏñËØÊý \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: GUI\_Initialize

\*\* ¹¦ÄÜÃèÊö: ³õÊ¼»¯GUI£¬°üÀ¨³õÊ¼»¯ÏÔÊ¾»º³åÇø£¬³õÊ¼»¯LCM²¢ÇåÆÁ

\*\* Êä¡¡Èë: ÎÞ

\*\* Êä¡¡³ö: ÎÞ

\*\* È«¾Ö±äÁ¿: ÎÞ

\*\* µ÷ÓÃÄ£¿é: ÎÞ

\*\* Modified by:

\*\* Modified date:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

extern void GUI\_Initialize(void);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: GUI\_FillSCR()

\*\* ¹¦ÄÜÃèÊö: È«ÆÁÌî³ä¡£Ö±½ÓÊ¹ÓÃÊý¾ÝÌî³äÏÔÊ¾»º³åÇø¡£¸ù¾ÝLCMµÄÊµ¼ÊÇé¿ö±àÐ´´Ëº¯Êý

\*\* Êä¡¡Èë: dat Ìî³äµÄÊý¾Ý

\*\* Êä¡¡³ö: ÎÞ

\*\* È«¾Ö±äÁ¿: ÎÞ

\*\* µ÷ÓÃÄ£¿é: ÎÞ

\*\* Modified by:

\*\* Modified date:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

extern void GUI\_FillSCR(TCOLOR dat);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: GUI\_ClearSCR()

\*\* ¹¦ÄÜÃèÊö: ÇåÆÁ

\*\* Êä¡¡Èë: ÎÞ

\*\* Êä¡¡³ö: ÎÞ

\*\* È«¾Ö±äÁ¿: ÎÞ

\*\* µ÷ÓÃÄ£¿é: ÎÞ

\*\* Modified by:

\*\* Modified date:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//extern void GUI\_ClearSCR(void);

#define GUI\_ClearSCR() GUI\_FillSCR(0x00)

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: GUI\_Point

\*\* ¹¦ÄÜÃèÊö: ÔÚÖ¸¶¨Î»ÖÃÉÏ»­µã

\*\* Êä¡¡Èë: xÖ¸¶¨µãËùÔÚÁÐµÄÎ»ÖÃ£»yÖ¸¶¨µãËùÔÚÐÐµÄÎ»ÖÃ£»colorÏÔÊ¾ÑÕÉ«(¶ÔÓÚºÚ°×É«LCM£¬Îª0Ê±Ãð£¬Îª1Ê±ÏÔÊ¾)

\*\* Êä¡¡³ö: ·µ»ØÖµÎª1Ê±±íÊ¾²Ù×÷³É¹¦£¬Îª0Ê±±íÊ¾²Ù×÷Ê§°Ü

\*\* È«¾Ö±äÁ¿: ÎÞ

\*\* µ÷ÓÃÄ£¿é: ÎÞ

\*\* Modified by:

\*\* Modified date:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

extern uint8 GUI\_Point(uint8 x, uint8 y, TCOLOR color);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: GUI\_ReadPoint

\*\* ¹¦ÄÜÃèÊö: ¶ÁÈ¡Ö¸¶¨µãµÄÑÕÉ«¡£¶ÔÓÚµ¥É«£¬ÉèÖÃretµÄd0Î»Îª1»ò0£¬4¼¶»Ò¶ÈÔòÎªd0¡¢d1ÓÐÐ§£¬8Î»RGBÔòd0--d7ÓÐÐ§£¬RGB½á¹¹ÔòR¡¢G¡¢B±äÁ¿ÓÐÐ§

\*\* Êä¡¡Èë: xÖ¸¶¨µãËùÔÚÁÐµÄÎ»ÖÃ£» yÖ¸¶¨µãËùÔÚÐÐµÄÎ»ÖÃ£»ret±£´æÑÕÉ«ÖµµÄÖ¸Õë

\*\* Êä¡¡³ö: ·µ»Ø0±íÊ¾Ö¸¶¨µØÖ·³¬³ö»º³åÇø·¶Î§

\*\* È«¾Ö±äÁ¿: ÎÞ

\*\* µ÷ÓÃÄ£¿é: ÎÞ

\*\* Modified by:

\*\* Modified date:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

extern uint8 GUI\_ReadPoint(uint8 x, uint8 y, TCOLOR \*ret);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: GUI\_HLine

\*\* ¹¦ÄÜÃèÊö: »­Ë®Æ½Ïß£¬²Ù×÷Ê§°ÜÔ­ÒòÊÇÖ¸¶¨µØÖ·³¬³ö»º³åÇø·¶Î§

\*\* Êä¡¡Èë: x0 Ë®Æ½ÏßÆðµãËùÔÚÁÐµÄÎ»ÖÃ

\* y0 Ë®Æ½ÏßÆðµãËùÔÚÐÐµÄÎ»ÖÃ

\* x1 Ë®Æ½ÏßÖÕµãËùÔÚÁÐµÄÎ»ÖÃ

\* color ÏÔÊ¾ÑÕÉ«(¶ÔÓÚºÚ°×É«LCM£¬Îª0Ê±Ãð£¬Îª1Ê±ÏÔÊ¾)

\*\* Êä¡¡³ö: ÎÞ

\*\* È«¾Ö±äÁ¿: ÎÞ

\*\* µ÷ÓÃÄ£¿é: ÎÞ

\*\* Modified by:

\*\* Modified date:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

extern void GUI\_HLine(uint8 x0, uint8 y0, uint8 x1, TCOLOR color);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: GUI\_RLine

\*\* ¹¦ÄÜÃèÊö: »­ÊúÖ±Ïß¡£

\*\* Êä¡¡Èë: x0 Ë®Æ½ÏßÆðµãËùÔÚÁÐµÄÎ»ÖÃ

\* y0 Ë®Æ½ÏßÆðµãËùÔÚÐÐµÄÎ»ÖÃ

\* x1 Ë®Æ½ÏßÖÕµãËùÔÚÁÐµÄÎ»ÖÃ

\* color ÏÔÊ¾ÑÕÉ«(¶ÔÓÚºÚ°×É«LCM£¬Îª0Ê±Ãð£¬Îª1Ê±ÏÔÊ¾)

\*\* Êä¡¡³ö: ÎÞ

\*\* È«¾Ö±äÁ¿: ÎÞ

\*\* µ÷ÓÃÄ£¿é: ÎÞ

\*\* Modified by:

\*\* Modified date:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

extern void GUI\_RLine(uint8 x0, uint8 y0, uint8 y1, TCOLOR color);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: GUI\_CmpColor()

\*\* ¹¦ÄÜÃèÊö: ÅÐ¶ÏÑÕÉ«ÖµÊÇ·ñÒ»ÖÂ¡£

\*\* Êä¡¡Èë: color1 ÑÕÉ«Öµ1

\* color2 ÑÕÉ«Öµ2

\* color ÏÔÊ¾ÑÕÉ«(¶ÔÓÚºÚ°×É«LCM£¬Îª0Ê±Ãð£¬Îª1Ê±ÏÔÊ¾)

\*\* Êä¡¡³ö: ·µ»Ø1±íÊ¾ÏàÍ¬£¬·µ»Ø0±íÊ¾²»ÏàÍ¬¡£

\*\* È«¾Ö±äÁ¿: ÎÞ

\*\* µ÷ÓÃÄ£¿é: ÎÞ

\*\* Modified by:

\*\* Modified date:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//extern int GUI\_CmpColor(TCOLOR color1, TCOLOR color2);

#define GUI\_CmpColor(color1, color2) ( (color1&0x01) == (color2&0x01) )

//ÑÕÉ«Öµ¸´ÖÆ color1£ºÄ¿±êÑÕÉ«±äÁ¿ color2£ºÔ´ÑÕÉ«±äÁ¿ color ÏÔÊ¾ÑÕÉ«(¶ÔÓÚºÚ°×É«LCM£¬Îª0Ê±Ãð£¬Îª1Ê±ÏÔÊ¾)

#define GUI\_CopyColor(color1, color2) \*color1 = color2&0x01

**6.7 LPC2124.h**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* This file is part of the uVision/ARM development tools \*/

/\* Copyright KEIL ELEKTRONIK GmbH 2002-2005 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* \*/

/\* LPC21XX.H: Header file for Philips LPC2114 / LPC2119 \*/

/\* LPC2124 / LPC2129 \*/

/\* LPC2194 \*/

/\* \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#ifndef \_\_LPC21xx\_H

#define \_\_LPC21xx\_H

/\* Vectored Interrupt Controller (VIC) \*/

#define VICIRQStatus (\*((volatile unsigned long \*) 0xFFFFF000))

#define VICFIQStatus (\*((volatile unsigned long \*) 0xFFFFF004))

#define VICRawIntr (\*((volatile unsigned long \*) 0xFFFFF008))

#define VICIntSelect (\*((volatile unsigned long \*) 0xFFFFF00C))

#define VICIntEnable (\*((volatile unsigned long \*) 0xFFFFF010))

#define VICIntEnClr (\*((volatile unsigned long \*) 0xFFFFF014))

#define VICSoftInt (\*((volatile unsigned long \*) 0xFFFFF018))

#define VICSoftIntClr (\*((volatile unsigned long \*) 0xFFFFF01C))

#define VICProtection (\*((volatile unsigned long \*) 0xFFFFF020))

#define VICVectAddr (\*((volatile unsigned long \*) 0xFFFFF030))

#define VICDefVectAddr (\*((volatile unsigned long \*) 0xFFFFF034))

#define VICVectAddr0 (\*((volatile unsigned long \*) 0xFFFFF100))

#define VICVectAddr1 (\*((volatile unsigned long \*) 0xFFFFF104))

#define VICVectAddr2 (\*((volatile unsigned long \*) 0xFFFFF108))

#define VICVectAddr3 (\*((volatile unsigned long \*) 0xFFFFF10C))

#define VICVectAddr4 (\*((volatile unsigned long \*) 0xFFFFF110))

#define VICVectAddr5 (\*((volatile unsigned long \*) 0xFFFFF114))

#define VICVectAddr6 (\*((volatile unsigned long \*) 0xFFFFF118))

#define VICVectAddr7 (\*((volatile unsigned long \*) 0xFFFFF11C))

#define VICVectAddr8 (\*((volatile unsigned long \*) 0xFFFFF120))

#define VICVectAddr9 (\*((volatile unsigned long \*) 0xFFFFF124))

#define VICVectAddr10 (\*((volatile unsigned long \*) 0xFFFFF128))

#define VICVectAddr11 (\*((volatile unsigned long \*) 0xFFFFF12C))

#define VICVectAddr12 (\*((volatile unsigned long \*) 0xFFFFF130))

#define VICVectAddr13 (\*((volatile unsigned long \*) 0xFFFFF134))

#define VICVectAddr14 (\*((volatile unsigned long \*) 0xFFFFF138))

#define VICVectAddr15 (\*((volatile unsigned long \*) 0xFFFFF13C))

#define VICVectCntl0 (\*((volatile unsigned long \*) 0xFFFFF200))

#define VICVectCntl1 (\*((volatile unsigned long \*) 0xFFFFF204))

#define VICVectCntl2 (\*((volatile unsigned long \*) 0xFFFFF208))

#define VICVectCntl3 (\*((volatile unsigned long \*) 0xFFFFF20C))

#define VICVectCntl4 (\*((volatile unsigned long \*) 0xFFFFF210))

#define VICVectCntl5 (\*((volatile unsigned long \*) 0xFFFFF214))

#define VICVectCntl6 (\*((volatile unsigned long \*) 0xFFFFF218))

#define VICVectCntl7 (\*((volatile unsigned long \*) 0xFFFFF21C))

#define VICVectCntl8 (\*((volatile unsigned long \*) 0xFFFFF220))

#define VICVectCntl9 (\*((volatile unsigned long \*) 0xFFFFF224))

#define VICVectCntl10 (\*((volatile unsigned long \*) 0xFFFFF228))

#define VICVectCntl11 (\*((volatile unsigned long \*) 0xFFFFF22C))

#define VICVectCntl12 (\*((volatile unsigned long \*) 0xFFFFF230))

#define VICVectCntl13 (\*((volatile unsigned long \*) 0xFFFFF234))

#define VICVectCntl14 (\*((volatile unsigned long \*) 0xFFFFF238))

#define VICVectCntl15 (\*((volatile unsigned long \*) 0xFFFFF23C))

/\* Pin Connect Block \*/

#define PINSEL0 (\*((volatile unsigned long \*) 0xE002C000))

#define PINSEL1 (\*((volatile unsigned long \*) 0xE002C004))

#define PINSEL2 (\*((volatile unsigned long \*) 0xE002C014))

/\* General Purpose Input/Output (GPIO) \*/

#define IOPIN0 (\*((volatile unsigned long \*) 0xE0028000))

#define IOSET0 (\*((volatile unsigned long \*) 0xE0028004))

#define IODIR0 (\*((volatile unsigned long \*) 0xE0028008))

#define IOCLR0 (\*((volatile unsigned long \*) 0xE002800C))

#define IOPIN1 (\*((volatile unsigned long \*) 0xE0028010))

#define IOSET1 (\*((volatile unsigned long \*) 0xE0028014))

#define IODIR1 (\*((volatile unsigned long \*) 0xE0028018))

#define IOCLR1 (\*((volatile unsigned long \*) 0xE002801C))

#define IO0PIN (\*((volatile unsigned long \*) 0xE0028000))

#define IO0SET (\*((volatile unsigned long \*) 0xE0028004))

#define IO0DIR (\*((volatile unsigned long \*) 0xE0028008))

#define IO0CLR (\*((volatile unsigned long \*) 0xE002800C))

#define IO1PIN (\*((volatile unsigned long \*) 0xE0028010))

#define IO1SET (\*((volatile unsigned long \*) 0xE0028014))

#define IO1DIR (\*((volatile unsigned long \*) 0xE0028018))

#define IO1CLR (\*((volatile unsigned long \*) 0xE002801C))

/\* Memory Accelerator Module (MAM) \*/

#define MAMCR (\*((volatile unsigned char \*) 0xE01FC000))

#define MAMTIM (\*((volatile unsigned char \*) 0xE01FC004))

#define MEMMAP (\*((volatile unsigned char \*) 0xE01FC040))

/\* Phase Locked Loop (PLL) \*/

#define PLLCON (\*((volatile unsigned char \*) 0xE01FC080))

#define PLLCFG (\*((volatile unsigned char \*) 0xE01FC084))

#define PLLSTAT (\*((volatile unsigned short\*) 0xE01FC088))

#define PLLFEED (\*((volatile unsigned char \*) 0xE01FC08C))

/\* VPB Divider \*/

#define VPBDIV (\*((volatile unsigned char \*) 0xE01FC100))

/\* Power Control \*/

#define PCON (\*((volatile unsigned char \*) 0xE01FC0C0))

#define PCONP (\*((volatile unsigned long \*) 0xE01FC0C4))

/\* External Interrupts \*/

#define EXTINT (\*((volatile unsigned char \*) 0xE01FC140))

#define EXTWAKE (\*((volatile unsigned char \*) 0xE01FC144))

#define EXTMODE (\*((volatile unsigned char \*) 0xE01FC148))

#define EXTPOLAR (\*((volatile unsigned char \*) 0xE01FC14C))

/\* Timer 0 \*/

#define T0IR (\*((volatile unsigned long \*) 0xE0004000))

#define T0TCR (\*((volatile unsigned long \*) 0xE0004004))

#define T0TC (\*((volatile unsigned long \*) 0xE0004008))

#define T0PR (\*((volatile unsigned long \*) 0xE000400C))

#define T0PC (\*((volatile unsigned long \*) 0xE0004010))

#define T0MCR (\*((volatile unsigned long \*) 0xE0004014))

#define T0MR0 (\*((volatile unsigned long \*) 0xE0004018))

#define T0MR1 (\*((volatile unsigned long \*) 0xE000401C))

#define T0MR2 (\*((volatile unsigned long \*) 0xE0004020))

#define T0MR3 (\*((volatile unsigned long \*) 0xE0004024))

#define T0CCR (\*((volatile unsigned long \*) 0xE0004028))

#define T0CR0 (\*((volatile unsigned long \*) 0xE000402C))

#define T0CR1 (\*((volatile unsigned long \*) 0xE0004030))

#define T0CR2 (\*((volatile unsigned long \*) 0xE0004034))

#define T0CR3 (\*((volatile unsigned long \*) 0xE0004038))

#define T0EMR (\*((volatile unsigned long \*) 0xE000403C))

/\* Timer 1 \*/

#define T1IR (\*((volatile unsigned long \*) 0xE0008000))

#define T1TCR (\*((volatile unsigned long \*) 0xE0008004))

#define T1TC (\*((volatile unsigned long \*) 0xE0008008))

#define T1PR (\*((volatile unsigned long \*) 0xE000800C))

#define T1PC (\*((volatile unsigned long \*) 0xE0008010))

#define T1MCR (\*((volatile unsigned long \*) 0xE0008014))

#define T1MR0 (\*((volatile unsigned long \*) 0xE0008018))

#define T1MR1 (\*((volatile unsigned long \*) 0xE000801C))

#define T1MR2 (\*((volatile unsigned long \*) 0xE0008020))

#define T1MR3 (\*((volatile unsigned long \*) 0xE0008024))

#define T1CCR (\*((volatile unsigned long \*) 0xE0008028))

#define T1CR0 (\*((volatile unsigned long \*) 0xE000802C))

#define T1CR1 (\*((volatile unsigned long \*) 0xE0008030))

#define T1CR2 (\*((volatile unsigned long \*) 0xE0008034))

#define T1CR3 (\*((volatile unsigned long \*) 0xE0008038))

#define T1EMR (\*((volatile unsigned long \*) 0xE000803C))

/\* Pulse Width Modulator (PWM) \*/

#define PWMIR (\*((volatile unsigned long \*) 0xE0014000))

#define PWMTCR (\*((volatile unsigned long \*) 0xE0014004))

#define PWMTC (\*((volatile unsigned long \*) 0xE0014008))

#define PWMPR (\*((volatile unsigned long \*) 0xE001400C))

#define PWMPC (\*((volatile unsigned long \*) 0xE0014010))

#define PWMMCR (\*((volatile unsigned long \*) 0xE0014014))

#define PWMMR0 (\*((volatile unsigned long \*) 0xE0014018))

#define PWMMR1 (\*((volatile unsigned long \*) 0xE001401C))

#define PWMMR2 (\*((volatile unsigned long \*) 0xE0014020))

#define PWMMR3 (\*((volatile unsigned long \*) 0xE0014024))

#define PWMMR4 (\*((volatile unsigned long \*) 0xE0014040))

#define PWMMR5 (\*((volatile unsigned long \*) 0xE0014044))

#define PWMMR6 (\*((volatile unsigned long \*) 0xE0014048))

#define PWMPCR (\*((volatile unsigned long \*) 0xE001404C))

#define PWMLER (\*((volatile unsigned long \*) 0xE0014050))

/\* Universal Asynchronous Receiver Transmitter 0 (UART0) \*/

#define U0RBR (\*((volatile unsigned char \*) 0xE000C000))

#define U0THR (\*((volatile unsigned char \*) 0xE000C000))

#define U0IER (\*((volatile unsigned char \*) 0xE000C004))

#define U0IIR (\*((volatile unsigned char \*) 0xE000C008))

#define U0FCR (\*((volatile unsigned char \*) 0xE000C008))

#define U0LCR (\*((volatile unsigned char \*) 0xE000C00C))

#define U0LSR (\*((volatile unsigned char \*) 0xE000C014))

#define U0SCR (\*((volatile unsigned char \*) 0xE000C01C))

#define U0FDR (\*((volatile unsigned char \*) 0xE000C028))

#define U0DLL (\*((volatile unsigned char \*) 0xE000C000))

#define U0DLM (\*((volatile unsigned char \*) 0xE000C004))

/\* Universal Asynchronous Receiver Transmitter 1 (UART1) \*/

#define U1RBR (\*((volatile unsigned char \*) 0xE0010000))

#define U1THR (\*((volatile unsigned char \*) 0xE0010000))

#define U1IER (\*((volatile unsigned char \*) 0xE0010004))

#define U1IIR (\*((volatile unsigned char \*) 0xE0010008))

#define U1FCR (\*((volatile unsigned char \*) 0xE0010008))

#define U1LCR (\*((volatile unsigned char \*) 0xE001000C))

#define U1MCR (\*((volatile unsigned char \*) 0xE0010010))

#define U1LSR (\*((volatile unsigned char \*) 0xE0010014))

#define U1MSR (\*((volatile unsigned char \*) 0xE0010018))

#define U1SCR (\*((volatile unsigned char \*) 0xE001001C))

#define U1FDR (\*((volatile unsigned char \*) 0xE0010028))

#define U1DLL (\*((volatile unsigned char \*) 0xE0010000))

#define U1DLM (\*((volatile unsigned char \*) 0xE0010004))

/\* I2C Interface \*/

#define I2CONSET (\*((volatile unsigned char \*) 0xE001C000))

#define I2STAT (\*((volatile unsigned char \*) 0xE001C004))

#define I2DAT (\*((volatile unsigned char \*) 0xE001C008))

#define I2ADR (\*((volatile unsigned char \*) 0xE001C00C))

#define I2SCLH (\*((volatile unsigned short\*) 0xE001C010))

#define I2SCLL (\*((volatile unsigned short\*) 0xE001C014))

#define I2CONCLR (\*((volatile unsigned char \*) 0xE001C018))

/\* SPI0 (Serial Peripheral Interface 0) \*/

#define S0SPCR (\*((volatile unsigned short\*) 0xE0020000))

#define S0SPSR (\*((volatile unsigned char \*) 0xE0020004))

#define S0SPDR (\*((volatile unsigned short\*) 0xE0020008))

#define S0SPCCR (\*((volatile unsigned char \*) 0xE002000C))

#define S0SPINT (\*((volatile unsigned char \*) 0xE002001C))

/\* SPI1 (Serial Peripheral Interface 1) \*/

#define S1SPCR (\*((volatile unsigned short\*) 0xE0030000))

#define S1SPSR (\*((volatile unsigned char \*) 0xE0030004))

#define S1SPDR (\*((volatile unsigned short\*) 0xE0030008))

#define S1SPCCR (\*((volatile unsigned char \*) 0xE003000C))

#define S1SPINT (\*((volatile unsigned char \*) 0xE003001C))

/\* Real Time Clock \*/

#define ILR (\*((volatile unsigned char \*) 0xE0024000))

#define CTC (\*((volatile unsigned short\*) 0xE0024004))

#define CCR (\*((volatile unsigned char \*) 0xE0024008))

#define CIIR (\*((volatile unsigned char \*) 0xE002400C))

#define AMR (\*((volatile unsigned char \*) 0xE0024010))

#define CTIME0 (\*((volatile unsigned long \*) 0xE0024014))

#define CTIME1 (\*((volatile unsigned long \*) 0xE0024018))

#define CTIME2 (\*((volatile unsigned long \*) 0xE002401C))

#define SEC (\*((volatile unsigned char \*) 0xE0024020))

#define MIN (\*((volatile unsigned char \*) 0xE0024024))

#define HOUR (\*((volatile unsigned char \*) 0xE0024028))

#define DOM (\*((volatile unsigned char \*) 0xE002402C))

#define DOW (\*((volatile unsigned char \*) 0xE0024030))

#define DOY (\*((volatile unsigned short\*) 0xE0024034))

#define MONTH (\*((volatile unsigned char \*) 0xE0024038))

#define YEAR (\*((volatile unsigned short\*) 0xE002403C))

#define ALSEC (\*((volatile unsigned char \*) 0xE0024060))

#define ALMIN (\*((volatile unsigned char \*) 0xE0024064))

#define ALHOUR (\*((volatile unsigned char \*) 0xE0024068))

#define ALDOM (\*((volatile unsigned char \*) 0xE002406C))

#define ALDOW (\*((volatile unsigned char \*) 0xE0024070))

#define ALDOY (\*((volatile unsigned short\*) 0xE0024074))

#define ALMON (\*((volatile unsigned char \*) 0xE0024078))

#define ALYEAR (\*((volatile unsigned short\*) 0xE002407C))

#define PREINT (\*((volatile unsigned short\*) 0xE0024080))

#define PREFRAC (\*((volatile unsigned short\*) 0xE0024084))

/\* A/D Converter \*/

#define ADCR (\*((volatile unsigned long \*) 0xE0034000))

#define ADDR (\*((volatile unsigned long \*) 0xE0034004))

/\* CAN Acceptance Filter RAM \*/

#define AFRAM (\*((volatile unsigned long \*) 0xE0038000))

/\* CAN Acceptance Filter \*/

#define AFMR (\*((volatile unsigned long \*) 0xE003C000))

#define SFF\_sa (\*((volatile unsigned long \*) 0xE003C004))

#define SFF\_GRP\_sa (\*((volatile unsigned long \*) 0xE003C008))

#define EFF\_sa (\*((volatile unsigned long \*) 0xE003C00C))

#define EFF\_GRP\_sa (\*((volatile unsigned long \*) 0xE003C010))

#define ENDofTable (\*((volatile unsigned long \*) 0xE003C014))

#define LUTerrAd (\*((volatile unsigned long \*) 0xE003C018))

#define LUTerr (\*((volatile unsigned long \*) 0xE003C01C))

/\* CAN Central Registers \*/

#define CANTxSR (\*((volatile unsigned long \*) 0xE0040000))

#define CANRxSR (\*((volatile unsigned long \*) 0xE0040004))

#define CANMSR (\*((volatile unsigned long \*) 0xE0040008))

/\* CAN Controller 1 (CAN1) \*/

#define C1MOD (\*((volatile unsigned long \*) 0xE0044000))

#define C1CMR (\*((volatile unsigned long \*) 0xE0044004))

#define C1GSR (\*((volatile unsigned long \*) 0xE0044008))

#define C1ICR (\*((volatile unsigned long \*) 0xE004400C))

#define C1IER (\*((volatile unsigned long \*) 0xE0044010))

#define C1BTR (\*((volatile unsigned long \*) 0xE0044014))

#define C1EWL (\*((volatile unsigned long \*) 0xE0044018))

#define C1SR (\*((volatile unsigned long \*) 0xE004401C))

#define C1RFS (\*((volatile unsigned long \*) 0xE0044020))

#define C1RID (\*((volatile unsigned long \*) 0xE0044024))

#define C1RDA (\*((volatile unsigned long \*) 0xE0044028))

#define C1RDB (\*((volatile unsigned long \*) 0xE004402C))

#define C1TFI1 (\*((volatile unsigned long \*) 0xE0044030))

#define C1TID1 (\*((volatile unsigned long \*) 0xE0044034))

#define C1TDA1 (\*((volatile unsigned long \*) 0xE0044038))

#define C1TDB1 (\*((volatile unsigned long \*) 0xE004403C))

#define C1TFI2 (\*((volatile unsigned long \*) 0xE0044040))

#define C1TID2 (\*((volatile unsigned long \*) 0xE0044044))

#define C1TDA2 (\*((volatile unsigned long \*) 0xE0044048))

#define C1TDB2 (\*((volatile unsigned long \*) 0xE004404C))

#define C1TFI3 (\*((volatile unsigned long \*) 0xE0044050))

#define C1TID3 (\*((volatile unsigned long \*) 0xE0044054))

#define C1TDA3 (\*((volatile unsigned long \*) 0xE0044058))

#define C1TDB3 (\*((volatile unsigned long \*) 0xE004405C))

/\* CAN Controller 2 (CAN2) \*/

#define C2MOD (\*((volatile unsigned long \*) 0xE0048000))

#define C2CMR (\*((volatile unsigned long \*) 0xE0048004))

#define C2GSR (\*((volatile unsigned long \*) 0xE0048008))

#define C2ICR (\*((volatile unsigned long \*) 0xE004800C))

#define C2IER (\*((volatile unsigned long \*) 0xE0048010))

#define C2BTR (\*((volatile unsigned long \*) 0xE0048014))

#define C2EWL (\*((volatile unsigned long \*) 0xE0048018))

#define C2SR (\*((volatile unsigned long \*) 0xE004801C))

#define C2RFS (\*((volatile unsigned long \*) 0xE0048020))

#define C2RID (\*((volatile unsigned long \*) 0xE0048024))

#define C2RDA (\*((volatile unsigned long \*) 0xE0048028))

#define C2RDB (\*((volatile unsigned long \*) 0xE004802C))

#define C2TFI1 (\*((volatile unsigned long \*) 0xE0048030))

#define C2TID1 (\*((volatile unsigned long \*) 0xE0048034))

#define C2TDA1 (\*((volatile unsigned long \*) 0xE0048038))

#define C2TDB1 (\*((volatile unsigned long \*) 0xE004803C))

#define C2TFI2 (\*((volatile unsigned long \*) 0xE0048040))

#define C2TID2 (\*((volatile unsigned long \*) 0xE0048044))

#define C2TDA2 (\*((volatile unsigned long \*) 0xE0048048))

#define C2TDB2 (\*((volatile unsigned long \*) 0xE004804C))

#define C2TFI3 (\*((volatile unsigned long \*) 0xE0048050))

#define C2TID3 (\*((volatile unsigned long \*) 0xE0048054))

#define C2TDA3 (\*((volatile unsigned long \*) 0xE0048058))

#define C2TDB3 (\*((volatile unsigned long \*) 0xE004805C))

/\* CAN Controller 3 (CAN3) \*/

#define C3MOD (\*((volatile unsigned long \*) 0xE004C000))

#define C3CMR (\*((volatile unsigned long \*) 0xE004C004))

#define C3GSR (\*((volatile unsigned long \*) 0xE004C008))

#define C3ICR (\*((volatile unsigned long \*) 0xE004C00C))

#define C3IER (\*((volatile unsigned long \*) 0xE004C010))

#define C3BTR (\*((volatile unsigned long \*) 0xE004C014))

#define C3EWL (\*((volatile unsigned long \*) 0xE004C018))

#define C3SR (\*((volatile unsigned long \*) 0xE004C01C))

#define C3RFS (\*((volatile unsigned long \*) 0xE004C020))

#define C3RID (\*((volatile unsigned long \*) 0xE004C024))

#define C3RDA (\*((volatile unsigned long \*) 0xE004C028))

#define C3RDB (\*((volatile unsigned long \*) 0xE004C02C))

#define C3TFI1 (\*((volatile unsigned long \*) 0xE004C030))

#define C3TID1 (\*((volatile unsigned long \*) 0xE004C034))

#define C3TDA1 (\*((volatile unsigned long \*) 0xE004C038))

#define C3TDB1 (\*((volatile unsigned long \*) 0xE004C03C))

#define C3TFI2 (\*((volatile unsigned long \*) 0xE004C040))

#define C3TID2 (\*((volatile unsigned long \*) 0xE004C044))

#define C3TDA2 (\*((volatile unsigned long \*) 0xE004C048))

#define C3TDB2 (\*((volatile unsigned long \*) 0xE004C04C))

#define C3TFI3 (\*((volatile unsigned long \*) 0xE004C050))

#define C3TID3 (\*((volatile unsigned long \*) 0xE004C054))

#define C3TDA3 (\*((volatile unsigned long \*) 0xE004C058))

#define C3TDB3 (\*((volatile unsigned long \*) 0xE004C05C))

/\* CAN Controller 4 (CAN4) \*/

#define C4MOD (\*((volatile unsigned long \*) 0xE0050000))

#define C4CMR (\*((volatile unsigned long \*) 0xE0050004))

#define C4GSR (\*((volatile unsigned long \*) 0xE0050008))

#define C4ICR (\*((volatile unsigned long \*) 0xE005000C))

#define C4IER (\*((volatile unsigned long \*) 0xE0050010))

#define C4BTR (\*((volatile unsigned long \*) 0xE0050014))

#define C4EWL (\*((volatile unsigned long \*) 0xE0050018))

#define C4SR (\*((volatile unsigned long \*) 0xE005001C))

#define C4RFS (\*((volatile unsigned long \*) 0xE0050020))

#define C4RID (\*((volatile unsigned long \*) 0xE0050024))

#define C4RDA (\*((volatile unsigned long \*) 0xE0050028))

#define C4RDB (\*((volatile unsigned long \*) 0xE005002C))

#define C4TFI1 (\*((volatile unsigned long \*) 0xE0050030))

#define C4TID1 (\*((volatile unsigned long \*) 0xE0050034))

#define C4TDA1 (\*((volatile unsigned long \*) 0xE0050038))

#define C4TDB1 (\*((volatile unsigned long \*) 0xE005003C))

#define C4TFI2 (\*((volatile unsigned long \*) 0xE0050040))

#define C4TID2 (\*((volatile unsigned long \*) 0xE0050044))

#define C4TDA2 (\*((volatile unsigned long \*) 0xE0050048))

#define C4TDB2 (\*((volatile unsigned long \*) 0xE005004C))

#define C4TFI3 (\*((volatile unsigned long \*) 0xE0050050))

#define C4TID3 (\*((volatile unsigned long \*) 0xE0050054))

#define C4TDA3 (\*((volatile unsigned long \*) 0xE0050058))

#define C4TDB3 (\*((volatile unsigned long \*) 0xE005005C))

/\* Watchdog \*/

#define WDMOD (\*((volatile unsigned char \*) 0xE0000000))

#define WDTC (\*((volatile unsigned long \*) 0xE0000004))

#define WDFEED (\*((volatile unsigned char \*) 0xE0000008))

#define WDTV (\*((volatile unsigned long \*) 0xE000000C))

#endif // \_\_LPC21xx\_H

**6.8 target.h**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Copyright (c)\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* ¹ãÖÝÖÜÁ¢¹¦µ¥Æ¬»ú·¢Õ¹ÓÐÏÞ¹«Ë¾

\*\* ÑÐ ¾¿ Ëù

\*\* ²úÆ·Ò»²¿

\*\*

\*\* http://www.zlgmcu.com

\*\*

\*\*--------------ÎÄ¼þÐÅÏ¢--------------------------------------------------------------------------------

\*\*ÎÄ ¼þ Ãû: target.h

\*\*´´ ½¨ ÈË: ³ÂÃ÷¼Æ

\*\*×îºóÐÞ¸ÄÈÕÆÚ: 2003Äê5ÔÂ30ÈÕ

\*\*Ãè Êö: lpc210x£¨·ÉÀûÆÖµÄARM£©Ä¿±ê°åÌØÊâµÄ´úÂëÍ·ÎÄ¼þ

\*\* Ã¿¸ö¹¤³ÌÓ¦µ±¾ßÓÐÕâ¸öÎÄ¼þµÄ¿½±´£¬ÓÃ»§¸ù¾Ý³ÌÐòµÄÐèÒªÐÞ¸Ä±¾ÎÄ¼þ

\*\*--------------ÀúÊ·°æ±¾ÐÅÏ¢----------------------------------------------------------------------------

\*\* ´´½¨ÈË: ³ÂÃ÷¼Æ

\*\* °æ ±¾: v1.0

\*\* ÈÕ¡¡ÆÚ: 2003Äê5ÔÂ30ÈÕ

\*\* Ãè¡¡Êö: Ô­Ê¼°æ±¾

\*\*

\*\*------------------------------------------------------------------------------------------------------

\*\* ÐÞ¸ÄÈË:

\*\* °æ ±¾:

\*\* ÈÕ¡¡ÆÚ:

\*\* Ãè¡¡Êö:

\*\*

\*\*--------------µ±Ç°°æ±¾ÐÞ¶©------------------------------------------------------------------------------

\*\* ÐÞ¸ÄÈË:

\*\* ÈÕ¡¡ÆÚ:

\*\* Ãè¡¡Êö:

\*\*

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#ifndef IN\_TARGET

extern void Reset(void);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: Reset

\*\* ¹¦ÄÜÃèÊö: Ä¿±ê°åÈí¸´Î»

\*\* Êä¡¡Èë: ÎÞ

\*\*

\*\* Êä¡¡³ö: ÎÞ

\*\*

\*\* È«¾Ö±äÁ¿: ÎÞ

\*\* µ÷ÓÃÄ£¿é: ÎÞ

\*\*

\*\* ×÷¡¡Õß: ³ÂÃ÷¼Æ

\*\* ÈÕ¡¡ÆÚ: 2003Äê5ÔÂ30ÈÕ

\*\*-------------------------------------------------------------------------------------------------------

\*\* ÐÞ¸ÄÈË:

\*\* ÈÕ¡¡ÆÚ:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

extern void TargetInit(void);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* º¯ÊýÃû³Æ: TargetInit

\*\* ¹¦ÄÜÃèÊö: Ä¿±ê°å³õÊ¼»¯´úÂë£¬ÔÚÐèÒªµÄµØ·½µ÷ÓÃ£¬¸ù¾ÝÐèÒª¸Ä±ä

\*\* Êä¡¡Èë: ÎÞ

\*\*

\*\* Êä¡¡³ö: ÎÞ

\*\*

\*\* È«¾Ö±äÁ¿: ÎÞ

\*\* µ÷ÓÃÄ£¿é: ÎÞ

\*\*

\*\* ×÷¡¡Õß: ³ÂÃ÷¼Æ

\*\* ÈÕ¡¡ÆÚ: 2003Äê5ÔÂ30ÈÕ

\*\*-------------------------------------------------------------------------------------------------------

\*\* ÐÞ¸ÄÈË:

\*\* ÈÕ¡¡ÆÚ:

\*\*------------------------------------------------------------------------------------------------------

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#endif

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* End Of File

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**6.9 LPC2124.sct**

; \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

; \*\*\* Scatter-Loading Description File for LPC2103 \*\*\*

; \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

LR\_ROM1 0x00000000 0x00020000

{ ; load region size\_region

ER\_ROM1 0x00000000 0x00020000

{ ; load address = execution address

\*.o (RESET, +First)

\*(InRoot$$Sections)

.ANY (+RO)

}

RW\_IRAM1 0x40000000 0x00004000

{ ; RW data

.ANY (+RW +ZI)

}

}

**Thank You**