作業三 LCM控制實驗文字走馬燈

資工三乙/林采昕/408262143 2021/11/05

Q1-1. 請說明 HD44780U晶片之功能?

A1-1. HD44780U可以顯示字母數字、日語、符號，一個HD44780U可以顯示一或兩行8 個字元。

Q1-2. 詳細解釋輸入腳位 RS、E、R/W與DB0 ~ DB7 之意義(作用)為何 ?

A1-2.

RS : 選取暫存器.

0 = Instruction register (for write) Busy flag: address counter (for read)

1 = Data register (for write and read)

E : Starts data read/write

R/W : 0 = 寫入, 1 = 讀取.

DB0 ~ DB3 : 四個低電位的雙向tristate data bus pins。讓MPU和HD44780U之間可以進行數據傳輸和接收。這些腳位在 4-bit操作期間不會被使用。

DB4 ~ DB7 : 四個高電位的雙向tristate data bus pins。 用於在MPU和HD44780U之間進行數據傳輸和接收。DB7可以當busy flag。

Q2. 請仔細說明 Sitronix ST7066U晶片之 4種指令分類。提示：與腳位 RS、與 R/W有關。

A2.

0 0 IR write as an internal operation (display clear, etc.)

0 1 讀取 busy flag (DB7) 和 address counter (DB0 to DB6)

1 0 DR write as an internal operation (DR to DDRAM or CGRAM)

1 1 DR read as an internal operation (DDRAM or CGRAM to DR)

Q3. 請解釋AC(Address Counter) content 與 cursor (游標 )之關係。

A3. cursor的位置是儲存在Address Counter

Q4-1. 請說明 DDRAM 與 CGRAM 之用途為何？

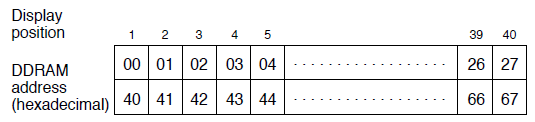
A4-1.

DDRAM : 可以存80 × 8 bits,或 80 字元的8-bit character codes，沒用到的區域可以做為general data RAM。

CGRAM : 用來存放字符點陣，並將能顯示的字符按照編號存放。

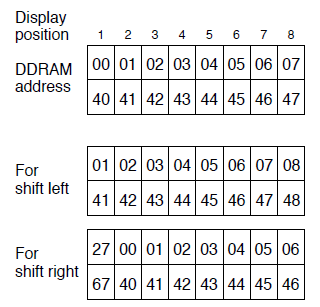
Q4-2. 2 line display mode時，display position (顯示位置 ) 與 DDRAM address (位址 )之關係為何 ?

A4-2. 當顯示的字元小於40 × 2，則從開頭顯示，且第一行結束的號碼與第二行開頭的數字不連續。



Q4-3. Display shift (位移)時， display position (顯示位置) 與 DDRAM address (位址)之關係會如何改變 ?

A4-3. 兩行(DDRAM address)會同時向左或向右移動



Q5. 請詳細解釋 Entry Mode Set指令之作用 ，包含各個 control flag說明 。

A5.

I/D:

當寫入一個字元，則DDRAM address + 1 (I/D = 1)，cursor往右移

當讀取一個字元，則DDRAM address - 1 (I/D = 0)，cursor往左移

S: 當S = 1，display右移(I/D = 0)，display左移(I/D = 1)，cursor不移。當S = 0，不移動。

Q8. 請詳細解釋 Function Set指令之作用 ，包含各個 control flag說明 。

A8.

DL: 設定介面字串長度

DL is 1 => data收到或寄出的長度為8-bit(DB7 to DB0)。

DL is 0 => data收到或寄出的長度為4-bit(DB7 to DB4)，且收到/寄出兩次。

N: 設定顯示出的行數。

F: Sets the character font.

Q9. 請詳細解釋 Display on/off Control指令之作用 ，包含各個 control flag說明 。

A9.

D: 控制顯示

D is 1 => Display on。

D is 0 => Display off。

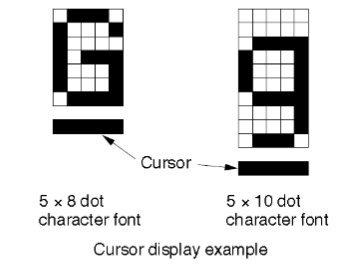
當Display off時，diaplay data仍保留在DDRAM，但在把D設定成1時，可以立即顯示出來。

C: 控制Cursor

C is 1 => Cursor display on。

C is 0 => Cursor display off。

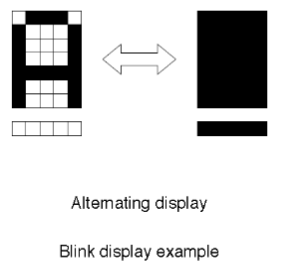
The cursor is displayed using 5 dots in the 8th line for 5 × 8 dot character font selection and in the 11th line for the 5 × 10 dot character font selection.



B: 控制Blinking

B is 1 => 顯示的字元閃爍

Cursor跟Blinking可以同時進行



Q10. 請詳細解釋 Display or Cursor Shift指令之作用 ，包含各個 control flag說明 。

A10.

控制Cursor 或 display shift移動，這個函式是用來確認或查詢輸出的字串，且第一行和第二行會同時右/左移。

在2-line display時，當經過40個字元，cursor會從第一行跳到第二行。

每行只會水平移動(第二行的輸出不會跑到第一行)

當display shift時，address counter不會變。

Q11. 請詳細解釋 Set DDRAM Address指令之作用。

A11. 將DDRAM address binary AAAAAAA 設成 the address counter.

Data 是從 MPU 讀取/寫入到 DDRAM.

N is 0 => 1-line display，AAAAAAA 是從 00H 到 4FH。

N is 1 => 2-line display，AAAAAAA 在第一行是從 00H 到 27H，第二行是從 40H 到 67H。

Q12. 請詳細解釋 Read Busy Flag & Address指令之作用 ，包含 BF (Busy Flag)說明 。

A12. Read busy flag and address reads the busy flag (BF)表示系統現在正在根據先前收到的指令internally operating。

若BF is 1，代表internal progress在程序中，在BF被設定成0之前，下一個指令不會被接收。

在下一個指令被寫入之前，要先檢查BF的當前狀態。同時，在binary AAAAAAA中address counter的值會被讀取，CG和DDRAM都會用到這個AC，且他的值是由之前的指令決定。address contents會跟設置CGRAM address、DDRAM address的指令相同。

程式碼:

/\*

LiquidCrystal Library - Hello World

Demonstrates the use a 16x2 LCD display. The LiquidCrystal

library works with all LCD displays that are compatible with the

Hitachi HD44780 driver. There are many of them out there, and you

can usually tell them by the 16-pin interface.

This sketch prints "Hello World!" to the LCD

and shows the time.

The circuit:

\* LCD RS pin to digital pin 8

\* LCD Enable pin to digital pin 9

\* LCD D4 pin to digital pin 4

\* LCD D5 pin to digital pin 5

\* LCD D6 pin to digital pin 6

\* LCD D7 pin to digital pin 7

\* LCD R/W pin to ground

\* LCD VSS pin to ground

\* LCD VCC pin to 5V

\* 10K resistor:

\* ends to +5V and ground

\* wiper to LCD VO pin (pin 3)

Library originally added 18 Apr 2008

by David A. Mellis

library modified 5 Jul 2009

by Limor Fried (http://www.ladyada.net)

example added 9 Jul 2009

by Tom Igoe

modified 22 Nov 2010

by Tom Igoe

This example code is in the public domain.

http://www.arduino.cc/en/Tutorial/LiquidCrystal

\*/

// include the library code:

#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins

LiquidCrystal lcd(8, 9, 4, 5, 6, 7);

char KeyValue[]={'1','2','3','A','4','5','6','B','7','8','9','C','\*','0','#','D'};

byte Row=0, Col=0;

int fg = 0;

char str1[] = "str1 : yuiopasdfghjkl;zxcvbnm,./qwertyu ";

//第一組文字lcd顯示的第一行

char str2[] = "str2 : bvcxz';lkjhgfdsa][poiuytrewq]/., ";

//第一組文字lcd顯示的第二行

char str3[] = "str3 : bvcxz';lkjhgfdsa][poiuytrewq]/., ";

//第二組文字lcd顯示的第一行

char str4[] = "str4 : yuiopasdfghjkl;zxcvbnm,./qwertyu ";

//第二組文字lcd顯示的第二行

void setup() {

pinMode(BT1, INPUT);

pinMode(BT2, INPUT);

pinMode(BT3, INPUT);

pinMode(BT4, INPUT);

pinMode(10, INPUT); //R1: S1,S2,S3,S4 (1,2,3,A)

pinMode(11, INPUT\_PULLUP); //R2: S5,S6,S7,S8 (4,5,6,B)

pinMode(12, INPUT\_PULLUP); //R3: S9, S10, S11,S12 (7,8,9,C)

pinMode(13, INPUT\_PULLUP); //R4: (\*,0,#,D)

pinMode(A0, OUTPUT); //A1, C1: S1,S5,S9 (1,4,7,\*)

pinMode(A1, OUTPUT); //A2, C2: S2,S6,S10 (2,5,8,0)

pinMode(A2, OUTPUT); //A3, C3: S3,S7,S11 (3,6,9,#)

pinMode(A3, OUTPUT); //A4, C4, S4,S8,S12 (\*,0, #,D) // (A, B, C, D) is correct.

//Pin left to right :R1 R2 R3 R4 C1 C2 C3 C4

digitalWrite(A0,HIGH);

digitalWrite(A1,HIGH);

digitalWrite(A2,HIGH);

digitalWrite(A3,HIGH);

lcd.begin(16, 2); // start the library

}

int buttonState1,buttonState2,buttonState3,buttonState4 = 0;

int delay\_number = 100;

void loop() {

static int keypressedcount=0;

byte keyindex=0;

buttonState1 = digitalRead(BT1);

buttonState2 = digitalRead(BT2);

buttonState3 = digitalRead(BT3);

buttonState4 = digitalRead(BT4);

if ((keyscan()==true) && (keyindex=(Row-1)\*4+Col))

{

if(keyindex == 1){

lcd.clear();

//清空lcd面板上的字

lcd.setCursor(0,0);

//座標移至lcd第一行第一格

lcd.print(str1);

lcd.setCursor(0,1);

//座標移至lcd第二行第一格

lcd.print(str2);

}

else if(keyindex == 4){

lcd.clear();

//清空lcd面板上的字

lcd.setCursor(0,0);

//座標移至lcd第一行第一格

lcd.print(str3);

lcd.setCursor(0,1);

//座標移至lcd第二行第一格

lcd.print(str4);

}

else if(keyindex == 5){

lcd.scrollDisplayLeft();

//lcd面板上的字兩行向左移

delay(200);

}

else if(keyindex == 16){

lcd.scrollDisplayRight();

//lcd面板上的字兩行向右移

delay(200);

}

}

if (buttonState1 == LOW) {

lcd.clear();

//清空lcd面板上的字

}

else if (buttonState2 == LOW) {

lcd.home();

//cursor回到第一行第一格

delay(150);

}

else if (buttonState4 == LOW) {

//fg紀錄當前cursor是on還是off

if(fg == 0){

lcd.cursor();

//cursor on

delay(500);

fg = 1;

}

else{

lcd.noCursor();

//cursor off

delay(500);

fg = 0;

}

}

// set the cursor to column 0, line 1

// (note: line 1 is the second row, since counting begins with 0):

//lcd.setCursor(0, 1);

// print the number of seconds since reset:

//lcd.print(millis() / 1000);

}

bool keyscan( )

{

Row=0;Col=0;

bool keypressed = false;

//scan col1

digitalWrite(A0, LOW);

digitalWrite(A1, HIGH);

digitalWrite(A2, HIGH);

digitalWrite(A3, HIGH);

delayMicroseconds(100);

//Read keys in row.1

if(digitalRead(10)==LOW)

{

digitalWrite(A0, HIGH);

Col=1;Row=1;

keypressed = true;

return(keypressed);

}

//Read keys in row.2

if(digitalRead(11)==LOW)

{

digitalWrite(A0, HIGH);

Col=1;Row=2;

keypressed = true;

return(keypressed);

}

//Read keys in row.3

if(digitalRead(12)==LOW)

{

digitalWrite(A0, HIGH);

Col=1;Row=3;

keypressed = true;

return(keypressed);

}

//Read keys in row.4

if(digitalRead(13)==LOW)

{

digitalWrite(A0, HIGH);

Col=1;Row=4;

keypressed = true;

return(keypressed);

}

//scan col 2

digitalWrite(A0, HIGH);

digitalWrite(A1, LOW);

digitalWrite(A2, HIGH);

digitalWrite(A3, HIGH);

delayMicroseconds(100);

//Read keys in row.1

if(digitalRead(10)==LOW)

{

digitalWrite(A1, HIGH);

Col=2;Row=1;

keypressed = true;

return(keypressed);

}

//Read keys in row.2

if(digitalRead(11)==LOW)

{

digitalWrite(A1, HIGH);

Col=2;Row=2;

keypressed = true;

return(keypressed);

}

//Read keys in row.3

if(digitalRead(12)==LOW)

{

digitalWrite(A1, HIGH);

Col=2;Row=3;

keypressed = true;

return(keypressed);

}

//Read keys in row.4

if(digitalRead(13)==LOW)

{

digitalWrite(A1, HIGH);

Col=2;Row=4;

keypressed = true;

return(keypressed);

}

//scan col 3

digitalWrite(A0, HIGH);

digitalWrite(A1, HIGH);

digitalWrite(A2, LOW);

digitalWrite(A3, HIGH);

delayMicroseconds(100);

//Read keys in row.1

if(digitalRead(10)==LOW)

{

digitalWrite(A2, HIGH);

Col=3;Row=1;

keypressed = true;

return(keypressed);

}

//Read keys in row.2

if(digitalRead(11)==LOW)

{

digitalWrite(A2, HIGH);

Col=3;Row=2;

keypressed = true;

return(keypressed);

}

//Read keys in row.3

if(digitalRead(12)==LOW)

{

digitalWrite(A2, HIGH);

Col=3;Row=3;

keypressed = true;

return(keypressed);

}

//Read keys in row.4

if(digitalRead(13)==LOW)

{

digitalWrite(A2, HIGH);

Col=3;Row=4;

keypressed = true;

return(keypressed);

}

//scan col 4

digitalWrite(A0, HIGH);

digitalWrite(A1, HIGH);

digitalWrite(A2, HIGH);

digitalWrite(A3, LOW);

delayMicroseconds(100);

//Read keys in row.1

if(digitalRead(10)==LOW)

{

digitalWrite(A3, HIGH);

Col=4;Row=1;

keypressed = true;

return(keypressed);

}

//Read keys in row.2

if(digitalRead(11)==LOW)

{

digitalWrite(A3, HIGH);

Col=4;Row=2;

keypressed = true;

return(keypressed);

}

//Read keys in row.3

if(digitalRead(12)==LOW)

{

digitalWrite(A3, HIGH);

Col=4;Row=3;

keypressed = true;

return(keypressed);

}

//Read keys in row.4

if(digitalRead(13)==LOW)

{

digitalWrite(A3, HIGH);

Col=4;Row=4;

keypressed = true;

return(keypressed);

}

return(false);

}