**BÁO CÁO THỰC HÀNH “THIẾT KẾ SoC”**

*Tên bài thực hành: Thiết kế Đồng Hồ*

1. **Mô tả tóm tắt nội dung bài thực hành**

Thiết kế Đồng Hồ có các chức năng sau:   
- Chạy giờ, phút, giây lên LCD và HEX  
- Có thể chỉnh thời gian giờ và phút  
- Có thể Reset thời gian về thời gian mà đã đặt  
- Có thể cài thời gian báo thức thông qua LEDG 5s.

A diagram of a computer

Description automatically generated

1. **Tiến trình thực hiện bài thực hành**

***Phần 1: Tạo Hardware***

*Bước 1:* *Tạo File Quartus II với tên DongHo*

*Bước 2:* *Tạo Qsys với tên file là system  
Cần chỉnh thông số như sau:*A computer screen with text boxes

Description automatically generated

* *LCD ON*

A screenshot of a computer

Description automatically generated

* *LCD BLON*

A screenshot of a computer

Description automatically generated

* *LCD EN*

A screenshot of a computer

Description automatically generated

* *LCD RW*

A computer screen shot of a computer

Description automatically generated

* *LCD RS*

A screenshot of a computer

Description automatically generated

* *LCD D*

A computer screen shot of a computer

Description automatically generated

* *Timer 0*

A screenshot of a computer

Description automatically generated

* *Hex 0*

A screenshot of a computer

Description automatically generated

* *Hex 1*

A screenshot of a computer

Description automatically generated

* *Hex 2*

A computer screen shot of a computer

Description automatically generated

* *Hex 3*

A screenshot of a computer

Description automatically generated

* *Hex 4*

A screenshot of a computer

Description automatically generated

* *Hex 5*

A screenshot of a computer

Description automatically generated

* *Key 0*

A screenshot of a computer

Description automatically generated

* *Key 1*

A screenshot of a computer

Description automatically generated

* *Key 3*

A screenshot of a computer

Description automatically generated

* *LED*

A screenshot of a computer

Description automatically generated

* *SW 0*

A screenshot of a computer

Description automatically generated

* *SW 1*

A computer screen with a box and text

Description automatically generated with medium confidence

*Sau khi thực hiện có Qsys sau:*

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

*Thiết kế đúng sẽ hiển thị nội dung sau:*

A screenshot of a computer

Description automatically generated

*Sau đó qua Generation và nhấn Generate để chạy:*

A screenshot of a computer error

Description automatically generated

*Sau khi chạy xong thì nhấn Close và trở lại Quartus để thực hiện tiếp.*

*Bước 3: Ta vào Quartus nhấn vào Project + Add/Remove Files in Project….*

*Tiếp theo tìm kiếm file .qip để add vào như hình bên dưới:*

A screenshot of a computer

Description automatically generated

*Tạo file .v có tên là DongHo.v*

A screenshot of a computer program

Description automatically generated

*Chạy chương trình và nạp vào kit*



***Phần 2: Viết code firmware (code C trong Eclipse)***

*Bước 1: Mở phần mềm eclipse*

*Tạo Project với đường link bài làm  
Tiếp theo tạo file .c có tên là Clock.c*

*Bước 2: Chuột phải vào .bsp đưa chuột đến Nios II + BSP Editor… để điều chỉnh như hình dưới đây:*A screenshot of a computer

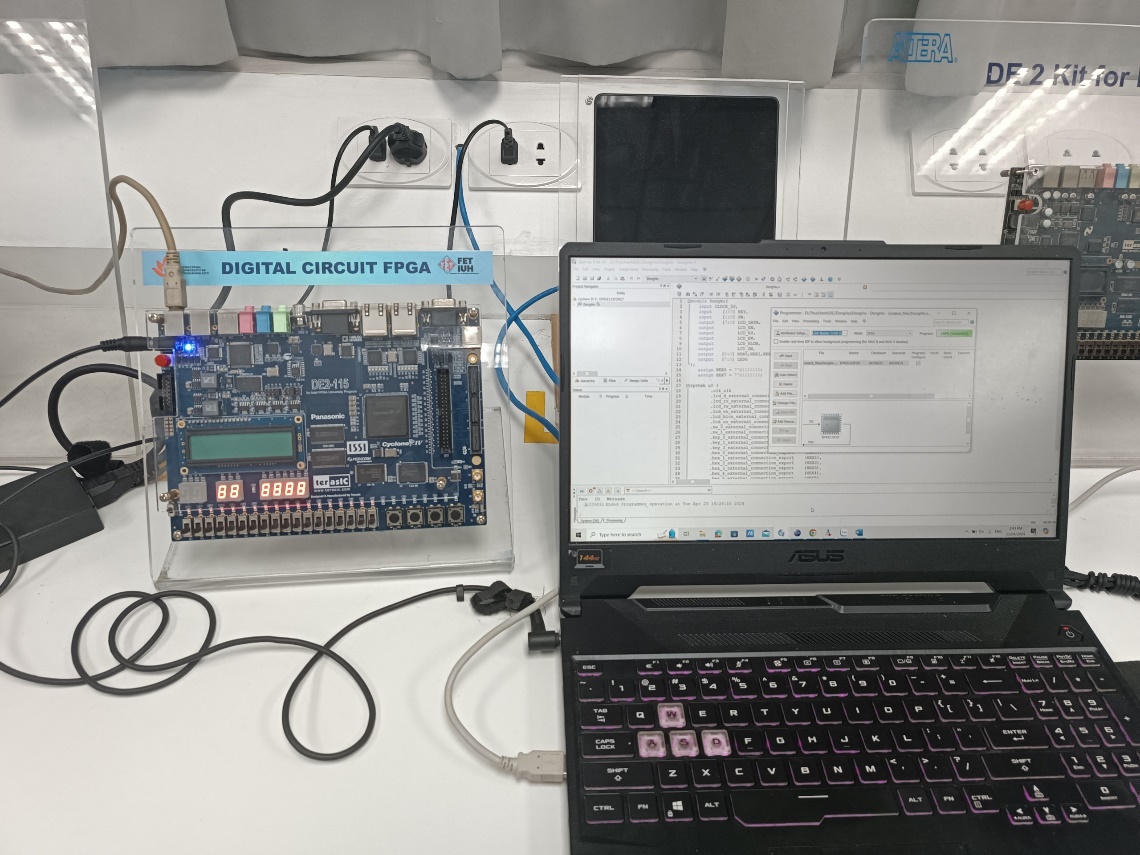
Description automatically generated

*Bước 3: Chạy code và nạp chương trình vào Kit*

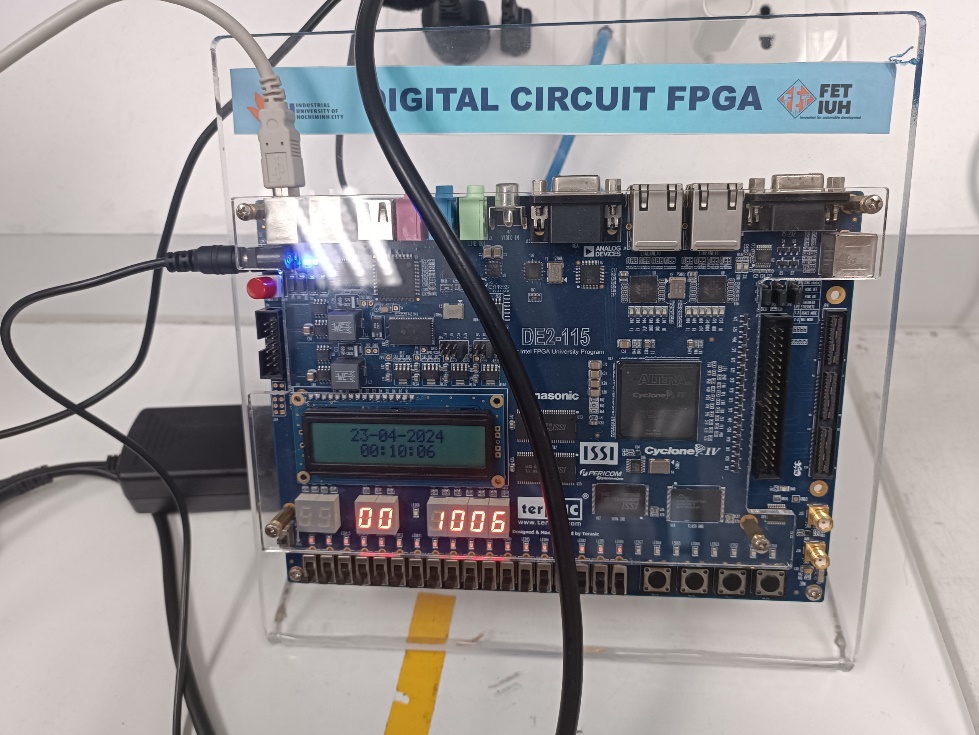
**Phần 3: Kiểm thử trên kit DE2**

*Bước 1: Chuẩn bị Kit, Dây cắm, Laptop*

*Bước 2: Kết nối vào kit và nạp chạy kit*

**

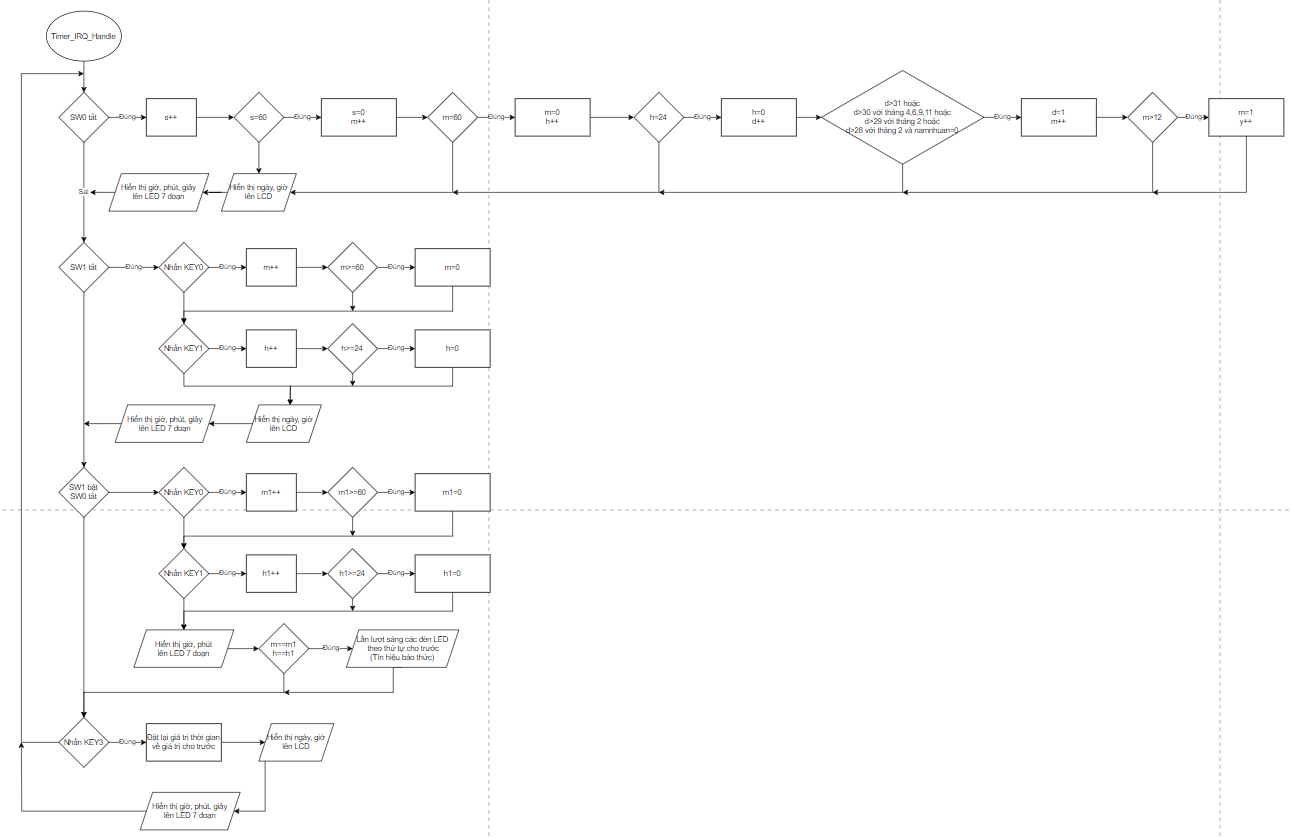
*Bước 3: Nạp code Clipse xuống kit để xem kết quả:*

**

1. **Lưu đồ giải thuật**

A diagram of a flowchart

Description automatically generated



A diagram of a flowchart

Description automatically generated

1. **Giải thích code Firmware**

**Code eclipse**

|  |
| --- |
| **#include** <stdio.h>  **#include** "io.h"  **#include** "system.h"  **#include** "altera\_avalon\_timer\_regs.h"  **#include** "sys/alt\_irq.h"  **#include** "sys/alt\_stdio.h"  **#include** "altera\_avalon\_pio\_regs.h"  **int** s=0,h=0,m=0,d=23,mt=4,y=2024,m1=0,h1=0,c=0,t=0;  **const** **unsigned** **char** decoder[10] = {0xC0, 0xF9, 0xA4, 0xB0, 0x99, 0x92, 0x82, 0xF8, 0x80, 0x90};  **void** **delay**(**int** a)  {  **volatile** **int** i = 0;  **while**(i < a\*7000)  {  i++;  }  }  **void** **delay1**(**int** a)  {  **volatile** **int** i = 0;  **while**(i < a\*1000)  {  i++;  }  }  **void** **command**(data)  {  IOWR\_ALTERA\_AVALON\_PIO\_DATA(LCD\_RS\_BASE, 0x00);  IOWR\_ALTERA\_AVALON\_PIO\_DATA(LCD\_RW\_BASE, 0x00);  IOWR\_ALTERA\_AVALON\_PIO\_DATA(LCD\_D\_BASE, data&0xFF);  IOWR\_ALTERA\_AVALON\_PIO\_DATA(LCD\_EN\_BASE, 0x01);  delay(2);  IOWR\_ALTERA\_AVALON\_PIO\_DATA(LCD\_EN\_BASE, 0x00);  delay(2);  }  **void** **command\_edit**(data)  {  IOWR\_ALTERA\_AVALON\_PIO\_DATA(LCD\_RS\_BASE, 0x00);  IOWR\_ALTERA\_AVALON\_PIO\_DATA(LCD\_RW\_BASE, 0x00);  IOWR\_ALTERA\_AVALON\_PIO\_DATA(LCD\_D\_BASE, data&0xFF);  IOWR\_ALTERA\_AVALON\_PIO\_DATA(LCD\_EN\_BASE, 0x01);  delay1(2);  IOWR\_ALTERA\_AVALON\_PIO\_DATA(LCD\_EN\_BASE, 0x00);  delay1(2);  }  **void** **lcd\_data**(**char** data)  {  IOWR\_ALTERA\_AVALON\_PIO\_DATA(LCD\_RS\_BASE, 0x01);  IOWR\_ALTERA\_AVALON\_PIO\_DATA(LCD\_RW\_BASE, 0x00);  IOWR\_ALTERA\_AVALON\_PIO\_DATA(LCD\_D\_BASE, data&0xFF);  IOWR\_ALTERA\_AVALON\_PIO\_DATA(LCD\_EN\_BASE, 0x01);  delay(1);  IOWR\_ALTERA\_AVALON\_PIO\_DATA(LCD\_EN\_BASE, 0x00);  delay(1);  }  **void** **lcd\_data\_edit**(**char** data)  {  IOWR\_ALTERA\_AVALON\_PIO\_DATA(LCD\_RS\_BASE, 0x01);  IOWR\_ALTERA\_AVALON\_PIO\_DATA(LCD\_RW\_BASE, 0x00);  IOWR\_ALTERA\_AVALON\_PIO\_DATA(LCD\_D\_BASE, data&0xFF);  IOWR\_ALTERA\_AVALON\_PIO\_DATA(LCD\_EN\_BASE, 0x01);  delay1(1);  IOWR\_ALTERA\_AVALON\_PIO\_DATA(LCD\_EN\_BASE, 0x00);  delay1(1);  }  **void** **lcd\_init**()  {  command(0x38);  delay(10);  command(0x0c);  delay(10);  command(0x06);  delay(10);  command(0x01);  }  **int** **namnhuan**(**int** year)  {  **if** ((year % 400 == 0) || ((year % 4 == 0) && (year % 100 != 0)))  **return** 1;  **else**  **return** 0;  }  **void** **Timer\_IRQ\_Handle**(**void**\* isr\_context)  {  //const unsigned int decoder[10] = {0xC0,0xF9,0xA4,0xB0,0x99,0x92,0x82,0xF8,0x80,0x90};  **while**(1)  {  **if**(IORD(SW\_0\_BASE, 0) == 0)  {  s++;  **if** (s == 60)  {  s = 0;  m++;  **if** (m == 60)  {  m = 0;  h++;  **if** (h == 24)  {  h = 0;  d++;  **if** (d > 31 || (d > 30 && (mt == 4 || mt == 6 || mt == 9 || mt == 11)) || (d > 29 && mt == 2) || (d > 28 && mt == 2 && !namnhuan(y))) {  d = 1;  mt++;  **if** (mt > 12)  {  mt = 1;  y++;  }  }  }  }  }  command(0x02); // Clear display  lcd\_data((d / 10) + '0');  lcd\_data((d % 10) + '0');  lcd\_data('-');  lcd\_data((mt / 10) + '0');  lcd\_data((mt % 10) + '0');  lcd\_data('-');  lcd\_data((y / 1000) + '0');  lcd\_data(((y % 1000) / 100) + '0');  lcd\_data(((y % 100) / 10) + '0');  lcd\_data((y % 10) + '0');  command(0xc0);  lcd\_data((h / 10) + '0');  lcd\_data((h % 10) + '0');  lcd\_data(':');  lcd\_data((m / 10) + '0');  lcd\_data((m % 10) + '0');  lcd\_data(':');  lcd\_data((s / 10) + '0');  lcd\_data((s % 10) + '0');  IOWR(LED\_BASE, 0, 0x00);  IOWR(HEX\_0\_BASE, 0, decoder[s % 10]);  IOWR(HEX\_1\_BASE, 0, decoder[s / 10]);  IOWR(HEX\_2\_BASE, 0, decoder[m % 10]);  IOWR(HEX\_3\_BASE, 0, decoder[m / 10]);  IOWR(HEX\_4\_BASE, 0, decoder[h % 10]);  IOWR(HEX\_5\_BASE, 0, decoder[h / 10]);  }  **else**  {  **if**(IORD(SW\_1\_BASE, 0) == 0)  {  **if**(IORD(KEY\_0\_BASE, 0)==0)  {  m++;  **if**(m>=60)  {  m=0;  }  }  **if**(IORD(KEY\_1\_BASE, 0)==0)  {  h++;  **if**(h>=24)  {  h=0;  }  }  command\_edit(0x02); // Clear display  lcd\_data\_edit((d / 10) + '0');  lcd\_data\_edit((d % 10) + '0');  lcd\_data\_edit('-');  lcd\_data\_edit((mt / 10) + '0');  lcd\_data\_edit((mt % 10) + '0');  lcd\_data\_edit('-');  lcd\_data\_edit((y / 1000) + '0');  lcd\_data\_edit(((y % 1000) / 100) + '0');  lcd\_data\_edit(((y % 100) / 10) + '0');  lcd\_data\_edit((y % 10) + '0');  command\_edit(0xc0);  lcd\_data\_edit((h / 10) + '0');  lcd\_data\_edit((h % 10) + '0');  lcd\_data\_edit(':');  lcd\_data\_edit((m / 10) + '0');  lcd\_data\_edit((m % 10) + '0');  lcd\_data\_edit(':');  lcd\_data\_edit((s / 10) + '0');  lcd\_data\_edit((s % 10) + '0');  IOWR(LED\_BASE, 0, 0x00);  IOWR(HEX\_0\_BASE, 0, decoder[s % 10]);  IOWR(HEX\_1\_BASE, 0, decoder[s / 10]);  IOWR(HEX\_2\_BASE, 0, decoder[m % 10]);  IOWR(HEX\_3\_BASE, 0, decoder[m / 10]);  IOWR(HEX\_4\_BASE, 0, decoder[h % 10]);  IOWR(HEX\_5\_BASE, 0, decoder[h / 10]);  }  }  **if**((IORD(SW\_1\_BASE, 0) == 1)&&(IORD(SW\_0\_BASE, 0) == 0))  {  **if**(IORD(KEY\_0\_BASE, 0)==0)  {  m1++;  **if**(m1==60)  {  m1=0;  }  }  **if**(IORD(KEY\_1\_BASE, 0)==0)  {  h1++;  **if**(h1==24)  {  h1=0;  }  }  IOWR(HEX\_0\_BASE, 0, 0xFF);  IOWR(HEX\_1\_BASE, 0, 0xFF);  IOWR(HEX\_2\_BASE, 0, decoder[m1 % 10]);  IOWR(HEX\_3\_BASE, 0, decoder[m1 / 10]);  IOWR(HEX\_4\_BASE, 0, decoder[h1 % 10]);  IOWR(HEX\_5\_BASE, 0, decoder[h1 / 10]);  **if** (m1==m && h1==h)  {  **if**(t==0)  {  c=5;  t=1;  }  **switch** (c)  {  **case** 1:  IOWR(LED\_BASE, 0, 0x18);  **break**;  **case** 2:  IOWR(LED\_BASE, 0, 0x24);  **break**;  **case** 3:  IOWR(LED\_BASE, 0, 0x42);  **break**;  **case** 4:  IOWR(LED\_BASE, 0, 0x81);  **break**;  **case** 5:  IOWR(LED\_BASE, 0, 0xFF);  **break**;  **default**:  **break**;  }  c--;  }  **else** t=0;  }  **if**(IORD(KEY\_3\_BASE, 0) == 0){  s=20,h=14,m=50,d=23,mt=4,y=2024;  command\_edit(0x02); // Clear display  lcd\_data\_edit((d / 10) + '0');  lcd\_data\_edit((d % 10) + '0');  lcd\_data\_edit('-');  lcd\_data\_edit((mt / 10) + '0');  lcd\_data\_edit((mt % 10) + '0');  lcd\_data\_edit('-');  lcd\_data\_edit((y / 1000) + '0');  lcd\_data\_edit(((y % 1000) / 100) + '0');  lcd\_data\_edit(((y % 100) / 10) + '0');  lcd\_data\_edit((y % 10) + '0');  command\_edit(0xc0);  lcd\_data\_edit((h / 10) + '0');  lcd\_data\_edit((h % 10) + '0');  lcd\_data\_edit(':');  lcd\_data\_edit((m / 10) + '0');  lcd\_data\_edit((m % 10) + '0');  lcd\_data\_edit(':');  lcd\_data\_edit((s / 10) + '0');  lcd\_data\_edit((s % 10) + '0');  IOWR(LED\_BASE, 0, 0x00);  IOWR(HEX\_0\_BASE, 0, decoder[s % 10]);  IOWR(HEX\_1\_BASE, 0, decoder[s / 10]);  IOWR(HEX\_2\_BASE, 0, decoder[m % 10]);  IOWR(HEX\_3\_BASE, 0, decoder[m / 10]);  IOWR(HEX\_4\_BASE, 0, decoder[h % 10]);  IOWR(HEX\_5\_BASE, 0, decoder[h / 10]);  }  }  IOWR\_ALTERA\_AVALON\_TIMER\_STATUS(TIMER\_0\_BASE, ALTERA\_AVALON\_TIMER\_STATUS\_TO\_MSK);  }  **void** **timer\_Init**()  {  **unsigned** **int** period = 0;  IOWR\_ALTERA\_AVALON\_TIMER\_CONTROL(TIMER\_0\_BASE, ALTERA\_AVALON\_TIMER\_CONTROL\_STOP\_MSK);  period = 50000000 - 1;  IOWR\_ALTERA\_AVALON\_TIMER\_PERIODL(TIMER\_0\_BASE, period);  IOWR\_ALTERA\_AVALON\_TIMER\_PERIODL(TIMER\_0\_BASE, (period >> 16));  IOWR\_ALTERA\_AVALON\_TIMER\_CONTROL(TIMER\_0\_BASE, ALTERA\_AVALON\_TIMER\_CONTROL\_CONT\_MSK |  ALTERA\_AVALON\_TIMER\_CONTROL\_ITO\_MSK | ALTERA\_AVALON\_TIMER\_CONTROL\_START\_MSK);  }  **int** **main**()  {  timer\_Init();  lcd\_init();  IOWR\_ALTERA\_AVALON\_PIO\_DATA(LCD\_ON\_BASE, 0x01);  IOWR\_ALTERA\_AVALON\_PIO\_DATA(LCD\_BLON\_BASE, 0x01);  alt\_ic\_isr\_register(0, TIMER\_0\_IRQ, Timer\_IRQ\_Handle, (**void**\*)0, (**void**\*)0);  **while**(1);  **return** 0;  } |

1. **Báo cáo lỗi trong thực hành**

Không có lỗi

1. **Kết luận bài thực hành**

Chạy đúng với yêu cầu đề ra