Thực hành Lập Trình Nhúng Căn Bản

Báo Cáo Lab 05

Thành viên nhóm:

Trần Thanh Duy 16520308

Lương Quốc Hải 16520327

Phan Thanh Duy 14521199

Đề bài: Truyền SPI giữa 2 board.

Đối với code Master

```
int main(void)
{
    uint8_t key;
    uint8_t i=0;
    /* Unlock the protected registers */
    UNLOCKREG();
    /* Enable the 12MHz oscillator oscillation */
    DrvSYS_SetOscCtrl(E_SYS_XTL12M, 1);
    /* Waiting for 12MHz Xtal stable */
    DrvSYS_Delay(5000);
    /* HCLK clock source. 0: external 12MHz. */
    DrvSYS_SelectHCLKSource(0);
    /* Configure general GPIO interrupt */

    DrvGPIO_Open(E_GPB, 15, E_IO_INPUT);
    /* Configure external interrupt */
    DrvGPIO_EnableEINT1(E_IO_BOTH_EDGE, E_MODE_EDGE, EINT1Callback);

    LOCKREG();
    initMaster();
    Initial_panel();
    clr_all_panel();
```

➔ Các bước setup clock, thanh ghi

```
state=STATE_START;
while(1)
{
  switch(state){
    case STATE_START:{
      state=STATE_INPUT;
      startInputState();
      break;
    }
    case STATE_INPUT: {
      key=0;
      while(key==0){
        key=scan_key();
        if(state != STATE_INPUT){
         break;
        }
      }
      while(scan_key()!=0){
        DrvSYS_Delay(1000);
        if(state != STATE_INPUT){
         break;
        }
      }
      if(state== STATE_INPUT){
        key+=64;
        //sendDataMasterToSlave_Single(key);
        Show_Word(3,lengthTXBuffer,key);
        tXbuffer[lengthTXBuffer]=key;
        lengthTXBuffer++;
        if(lengthTXBuffer >= MAX_LENGTH_TX_BUFFER){
          state=STATE_SEND_DATA;

        }
        else{

        }
      }
      break;
    }
```

Đầu tiên cho trạng thái mặc định nó là Start, để sau khi hoạt động thì sẽ nhảy vào trạng thái Start luôn.

Ở trạng thái Start, hàm có nhiệm vụ set trạng thái thành input và chuyển trạng thái sang input.

Ở trạng thái Input, hàm có nhiệm vụ đợi phím bấm và sau đó hiển thị các phím bấm được lên màn hình hiện tại, đồng thời lưu vào TX buffer để chuẩn bị cho việc gởi dữ liệu.

Dữ liệu được gởi đi khi số chữ bấm vào lớn hơn 8 hoặc khi ta bấm phím Ex Interrupt.

```
    }
    case STATE_SEND_DATA:{
      /*End character*/
      tXbuffer[lengthTXBuffer]=0;
      sendDataMasterToSlave(tXbuffer,lengthTXBuffer+1);
      lengthTXBuffer=0;
      clearTXBuffer();
      clr_all_panel();
      state=STATE_START;
      break;
    }

  }
}
return 0;
```

Ở trạng thái Send, hàm này có nhiệm vụ gởi lần lượt từng word sang board Slave. Độ dài của chuỗi gởi được luôn tính bằng Txbuffer+1 vì giá trị cuối cùng là giá trị NULL nhằm đánh dấu cho việc kết thúc chuỗi dữ liệu.

Đối với code Slave

Ở Slave thì có 2 trạng thái chính đó là Recive và Disable.

```
switch(state){
  case STATE_REC:
  {
    while(g_Spi2IntFlag == 0);
    g_Spi2IntFlag=0;
    DrvSPI_SingleRead(eDRVSPI_PORT2,&recData );
    if(recData!=0 && count<8){
      recBuffer[count]=recData;
      count++;
    }
    else{
      state=STATE_DIS;
    }
    break;
```

Ở trạng thái Recive, hàm có nhiệm vụ kiểm tra cờ truyền, nếu nó dược bật thì liên tục nhận dữ liệu. Nhưng chỉ nhận tối đa 8 kí tự.

```
    }
    else{
      state=STATE_DIS;
    }
    break;
  }
  case STATE_DIS:{
    clr_all_panel();
    for(i=0;i<count;i++){
      Show_Word(3,i,recBuffer[i]);
      recBuffer[i]=0;
    }

    for(i=0; i<32;i++) {
      recBuffer[i] = 0;
    }
    count=0;
    state=STATE_REC;
    break;
  }
```

Ở trạng thái Disable. Nếu như đã nhận được chuỗi kí tự từ Master, board sẽ disable chân interrupt để không nhận thêm dữ liệu nửa.

Cuối cùng là xóa đi Buffer và chuyển về trạng thái chờ nhận dữ liệu (Recive).