112學年專題 - Feature #46

Feature # 58 (Closed): 光源模組連接ESP32

光源模組連接ESP32_韌體研發

2023-09-25 05:03 - Chifu Chung

狀態:	Closed	**開始日期:**	2023-09-25
優先權:	Normal	**完成日期:**	2023-10-31
被分派者:	宏益 廖	**完成百分比:**	100%
分類:		**預估工時:**	0:00 小時
版本:		**耗用工時:**	0:00 小時

概述

相關的議題清單:

關聯至 硬體組 - Task #231: AWPPG P type 連續裝置製作	New	2024-09-02

歷史

#1 - 2023-09-25 05:08 - Chifu Chung

- 狀態 從 New 變更為 In Progress

#2 - 2023-09-25 05:12 - Chifu Chung

- 完成日期 設定為 2023-10-31

#3 - 2023-09-25 05:19 - Chifu Chung

- 父議題 設定為 #58

#4 - 2023-10-02 06:04 - 宏益 廖

- 檔案 clipboard-202310021404-gethv.png 已新增

- 檔案 clipboard-202310021404-di9nw.png 已新增

*I2C相關知識

韌體開發前請先研讀I2C相關知識包含：需要幾條線、電路、接線、時序圖、START Condition、Stop Condition...

參考資料 or 自行Google：

https://wiki.csie.ncku.edu.tw/embedded/I2C

https://rexpighj123.pixnet.net/blog/post/219960237

==========

*TCA6507

8.5.2.1 Writes

To write on the I²C bus, the master sends a START condition on the bus with the address of the slave, as well as the last bit (the R/$\overline{W}$ bit) set to 0, which signifies a write. After the slave sends the acknowledge bit, the master then sends the register address of the register to which it wishes to write. The slave acknowledges again, letting the master know it is ready. After this, the master starts sending the register data to the slave until the master has sent all the data necessary (which can be only a single byte), and the master terminates the transmission with a STOP condition.

Figure 14 shows an example of writing a single byte to a register.

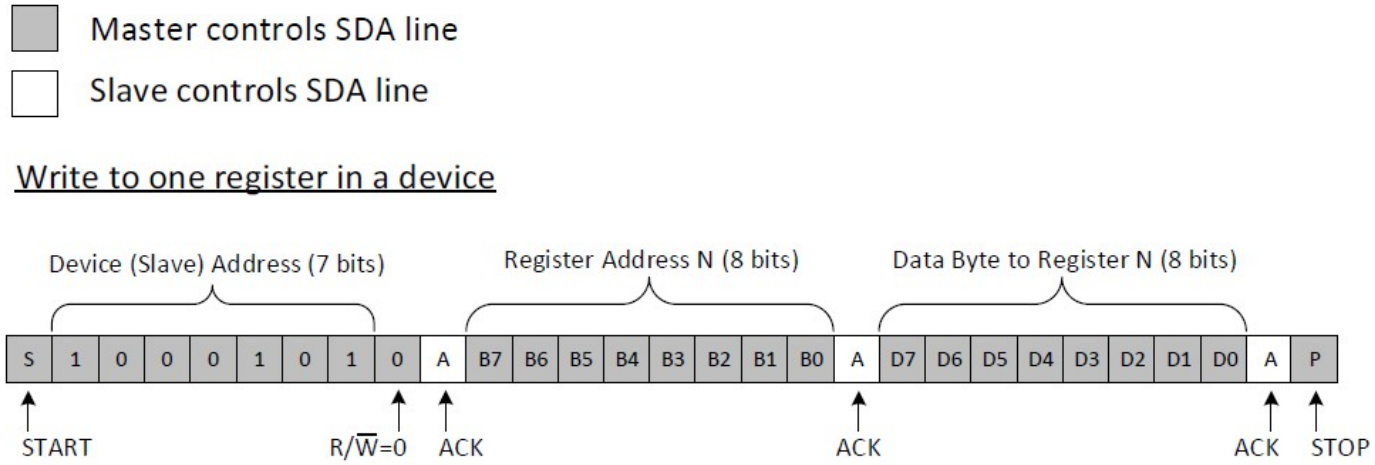
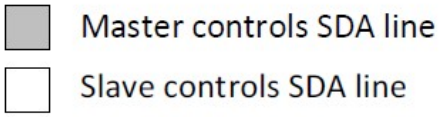


Figure 14. Write to Register

Figure 15 shows an example of writing to a Fully On register.



8.5.2.2 Reads

Reading from a slave is very similar to writing, but requires some additional steps. To read from a slave, the master must first instruct the slave which register it wishes to read from. This is done by the master starting off the transmission in a similar fashion as the write, by sending the address with the R/$\overline{W}$ bit equal to 0 (signifying a write), followed by the register address it wishes to read from. Once the slave acknowledges this register address, the master sends a START condition again, followed by the slave address with the R/$\overline{W}$ bit set to 1 (signifying a read). This time, the slave acknowledges the read request, and the master releases the SDA bus, but continues supplying the clock to the slave. During this part of the transaction, the master becomes the master-receiver, and the slave becomes the slave-transmitter.

The master continues to send out the clock pulses, but releases the SDA line so that the slave can transmit data. At the end of every byte of data, the master sends an ACK to the slave, letting the slave know that it is ready for more data. Once the master has received the number of bytes it is expecting, it sends a NACK, signaling to the slave to halt communications and release the bus. The master follows this up with a STOP condition.

Figure 16 shows an example of reading a single byte from a slave register.

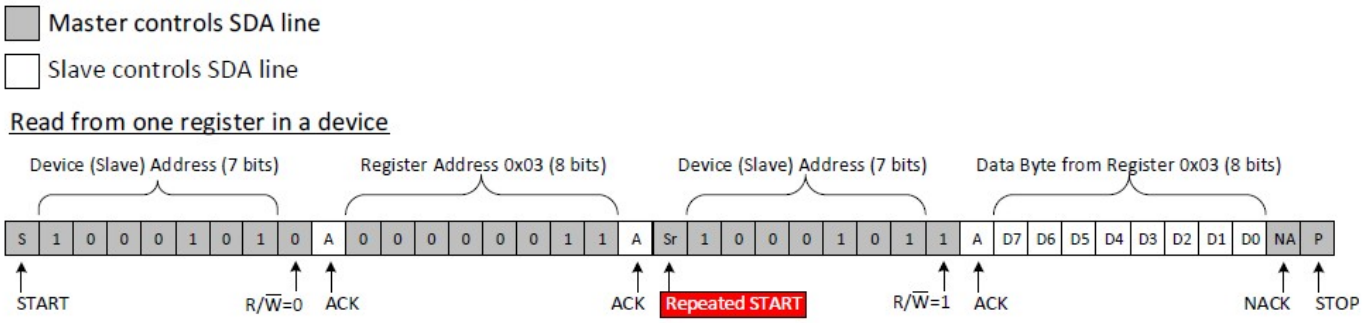


Figure 16. Read From Register Example

8.5.3 Device Address

The address of the TCA6507 is shown in Figure 17.

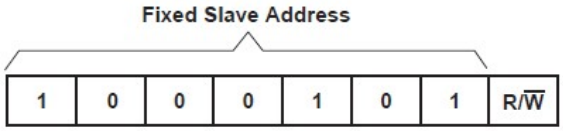


Figure 17. TCA6507 Address

以下程式應用 Wire.h 完成 Datasheet 上要求的規則，達到傳輸資料的效果

==========

*程式

```
#include  <Wire.h>

//ESP32  I2C  腳位  &  TCA6507  Enable
#define  SDA  21
#define  SCL  22
#define  EN  26

enum  TCA_registers{
    SELECT0  =  0x00,
    SELECT1  =  0x01,
    SELECT2  =  0x02,
    FADE_ON_TIME  =  0x03,
    FULLY_ON_TIME  =  0x04,
    FADE_OFF_TIME  =  0x05,
    FIRST_FULLY_OFF_TIME  =  0x06,
    SECOND_FULLY_OFF_TIME  =  0x06,
    MAXIMUM_INTENSITY  =  0x08,
    ONE_SHOT  =  0x09,
    INITIALIZATION  =  0x10,
};
```

```
int8_t TCA_writeBytes(uint8_t devAddr, uint8_t regAddr, uint8_t length, uint8_t *data){
    uint8_t status = 0;

    Wire.beginTransmission(devAddr);
    Wire.write(regAddr);

    for (uint8_t i = 0; i < length; i++) {
        Wire.write((uint8_t) data[i]);
    }

    status = Wire.endTransmission();
    return status == 0;
}

int8_t TCA_writeByte(uint8_t devAddr, uint8_t regAddr, uint8_t data){
    return TCA_writeBytes(devAddr, regAddr, 1, &data);
}

int8_t TCA_read(uint8_t devAddr, uint8_t regAddr, uint8_t length, uint8_t *data, uint16_t timeout){

    int8_t count = 0;
    uint32_t t1 = millis();

    for (uint8_t k = 0; k < length; k += std::min<int>(length, BUFFER_LENGTH)) {
        Wire.beginTransmission(devAddr);
        Wire.write(regAddr);
        Wire.endTransmission();

        Wire.requestFrom(devAddr, (uint8_t) std::min<int> (length - k, BUFFER_LENGTH));

        for (; Wire.available() && (timeout == 0 || millis() - t1 < timeout); count++) {
                data[count] = Wire.read();
        }
    }

    if (timeout > 0 && millis() - t1 >= timeout && count < length) count = -1; // timeout

    return count;
}

bool TCA_isConnected(){
    uint8_t result = 0;

    return ( TCA_read(device_address, SELECT0, 1, &result, 0) == 1);
}

void setup() {
    Serial.begin(115200);

    Wire.begin(SDA, SCL);

    pinMode(EN, OUTPUT);
    digitalWrite(EN, LOW);
    delay(20);
    digitalWrite(EN, HIGH);
    delay(20);

    if(TCA_isConnected()){
        Serial.print("TCA6507 connected!!");
    }
    else{
        Serial.print("TCA6507 doesn't connected.");
    }
}

void loop() {
    //on_off在後續LCD螢幕上可以用觸控來控制
    //先試試用Serial port控制 或 讓on_off每隔1秒+1 讓燈閃爍
    if(on_off % 2 == 1){
        uint8_t selects[3] = {0x00, 0x00, 0x1E};
        TCA_writeByte(device_address, SELECT2, selects[2]);
    }

    if(on_off % 2 == 0){
        TCA_writeByte(device_address, SELECT2, 0x00);
```

```
    }
}
```

#5 - 2023-10-02 06:07 - 宏益 廖

- 狀態 從 In Progress 變更為 Resolved

#6 - 2023-10-03 09:26 - Chifu Chung

- 完成百分比 從 0 變更為 100

#7 - 2023-12-14 18:45 - Chifu Chung

- 狀態 從 Resolved 變更為 Closed

#8 - 2024-09-03 14:45 - Chifu Chung

- 關聯至 Task #231: AWPPG P type 連續裝置製作 已新增

檔案

clipboard-202310021404-gethv.png	540 KB	2023-10-02	宏益 廖
clipboard-202310021404-di9nw.png	771 KB	2023-10-02	宏益 廖