

3. I2C 读写实验例程

3.1 MES50HP 开发板简介

MES50HP 开发板集成一片 EEPROM，型号为 24LC02。容量为： 2Kbit (1*256*8bit)，由 1 个 256byte 的 block 组成，通过 IIC 总线进行通信（详情请查看“MES50HP 开发板硬件使用手册”）。

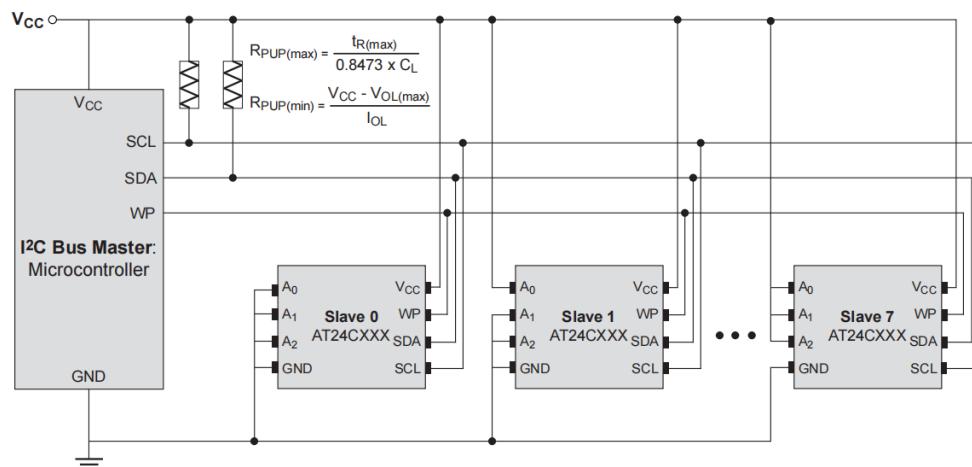
3.2 实验目的

实验在 EEPROM 的 8' b0 地址写入 8bit 数据 8' b10101010，按下 KEY2 表示写，再按下 KEY4 触发写开始；按下 KEY3 表示在 8' b0 地址读，再按下 KEY4 触发读开始，将读出的数据以二进制形式用 LED 灯显示，按下 KEY1 使 LED 恢复为全灭状态。

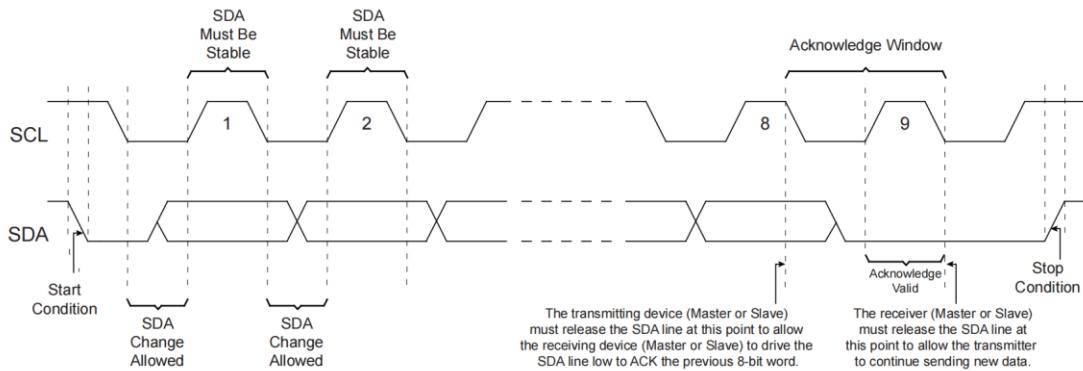
3.3 实验原理

3.3.1 IIC 协议简介

IIC 是一种两线式串行总线，由数据线 SDA 和时钟线 SCL 构成通信线路，属于半双工通信方式。所有接到 I2C 总线上的设备的串行数据 SDA 都接到总线的 SDA 上，各设备的时钟线 SCL 接到总线的 SCL 上。



IIC 总线上每一个设备都有唯一地址，通过“唯一地址”进行寻址通信。并且数据传输过程中还必须有应答信号，所以 SDA 数据线是双向的。

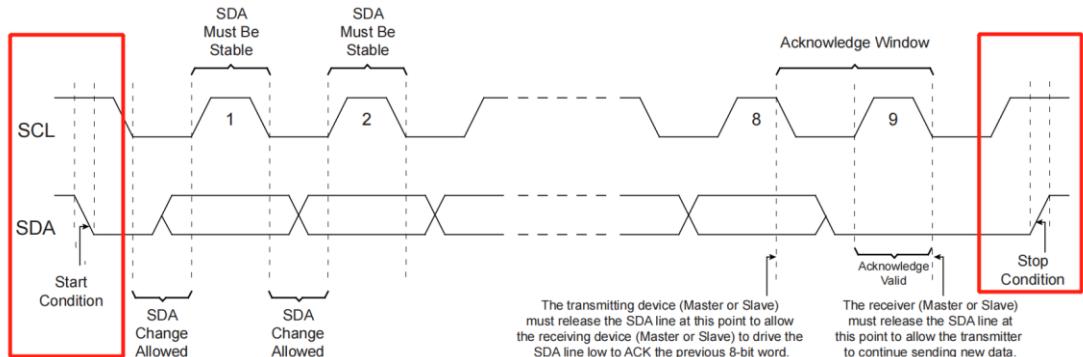


传输速率: 标准模式下可达 100Kbit/s, 快速模式下可达 400Kbit/s, 高速模式下可达

3.4Mbit/s

3.3.2 IIC 时序实现

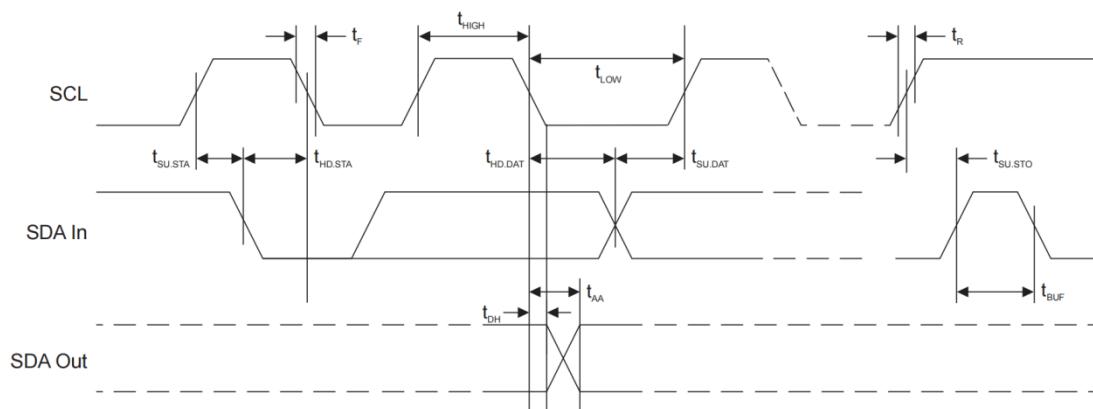
在空闲状态下, SCL 和 SDA 都为高电平。SCL 输出高电平时, SDA 由高到低变化为开始传输标志。SCL 为高电平, SDA 再出现上升沿, 表示传输结束



时序要求:

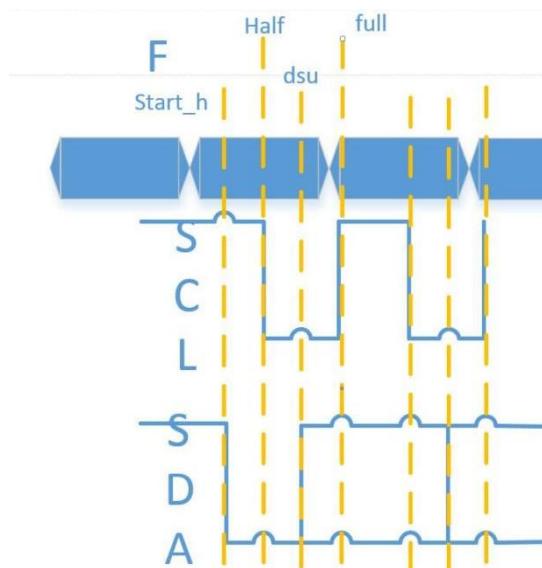
Parameter	Symbol	Fast Mode		Fast Mode Plus		Units	
		$V_{CC} = 1.7V \text{ to } 2.5V$		$V_{CC} = 2.5V \text{ to } 5.5V$			
		Min.	Max.	Min.	Max.		
Clock Frequency, SCL	f_{SCL}	—	400	—	1000	kHz	
Clock Pulse Width Low	t_{LOW}	1,200	—	500	—	ns	
Clock Pulse Width High	t_{HIGH}	600	—	400	—	ns	
Input Filter Spike Suppression	t_i	—	100	—	50	ns	
Clock Low to Data Out Valid	t_{AA}	100	900	50	450	ns	
Bus Free Time between Stop and Start	t_{BUF}	1,200	—	500	—	ns	
Start Hold Time	$t_{HD STA}$	600	—	250	—	ns	
Start Set-Up Time	$t_{SU STA}$	600	—	250	—	ns	
Data In Hold Time	$t_{HD DAT}$	0	—	0	—	ns	

.....continued							
Parameter	Symbol	Fast Mode		Fast Mode Plus		Units	
		$V_{CC} = 1.7V \text{ to } 2.5V$		$V_{CC} = 2.5V \text{ to } 5.5V$			
		Min.	Max.	Min.	Max.		
Data In Set-up Time	$t_{SU.DAT}$	100	—	100	—	ns	
Inputs Rise Time ⁽²⁾	t_R	—	300	—	300	ns	
Inputs Fall Time ⁽²⁾	t_F	—	300	—	100	ns	
Stop Set-Up Time	$t_{SU.STO}$	600	—	250	—	ns	
Data Out Hold Time	t_{DH}	50	—	50	—	ns	
Write Cycle Time	t_{WR}	—	5	—	5	ms	



本实验采用 400Kbit/s 的传输速率，所以 SCL 的频率应设置为 400KHz，且占空比为 50%，即 $t_{HIGH}=1250\text{ns}$, $t_{LOW}=1250\text{ns}$;

为满足上表中的时序要求，设计一 SCL 周期计数器，在计数器上取 4 个位置：



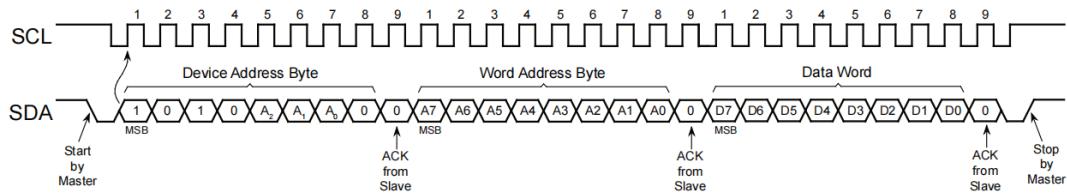
SCL 变化点为 scl_cnt 计数器的“half”处和“full”处；

SDA 变化点为 scl_cnt 计数器的“dsu”处。

3.3.3 IIC 读写操作流程

Byte Write:

Device Address Byte



主设备每传输 8bit 数据需释放 SDA，接收来自从设备的 ACK 应答信号。

Byte Write:

开始第一个字节都为从设备的 device ID:

Package	Device Type Identifier				Hardware Slave Address Bits			R/W Select
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SOIC, TSSOP, UDFN, PDIP, VFBGA	1	0	1	0	A2	A1	A0	R/W
SOT23	1	0	1	0	0	0	0	R/W

Device Address Byte 的最低位若为“0”表示“写”，若为“1”则表示“读”。

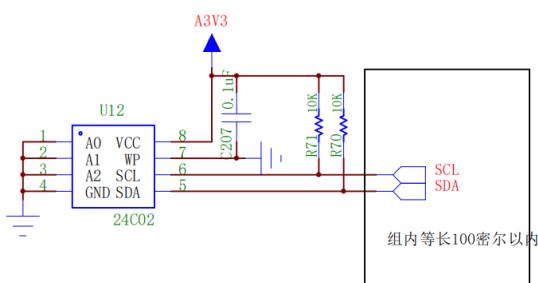
第二个字节是写地址的字节:

Table 6-2. Word Address Byte

Device	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
AT24C01C	X ⁽¹⁾	A6	A5	A4	A3	A2	A1	A0
AT24C02C	A7	A6	A5	A4	A3	A2	A1	A0

第三个字节是写数据的字节。其中 Device Address Byte 的 A₂、A₁和 A₀与硬件连接相关:

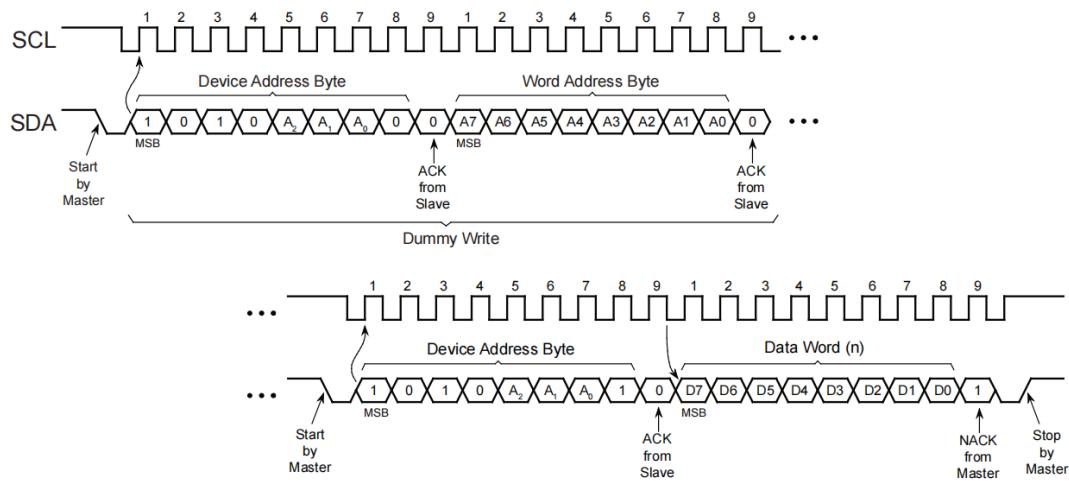
EEPROM



因此，开发板上的 EEPROM 的 device ID 为: 7' b1010000;

Random Read:

Figure 8-2. Random Read

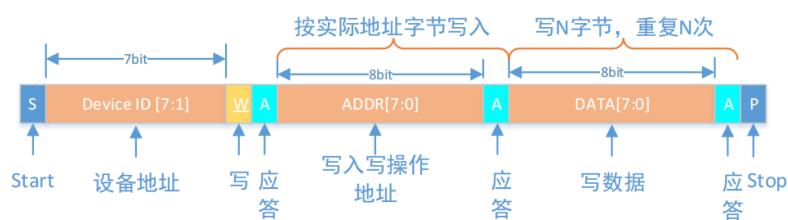


读操作的过程中，先进行 Dummy Write，此部分与写操作类似，不管读或者写，主机首先是要找到从机，然后指明要读或者写的数据。

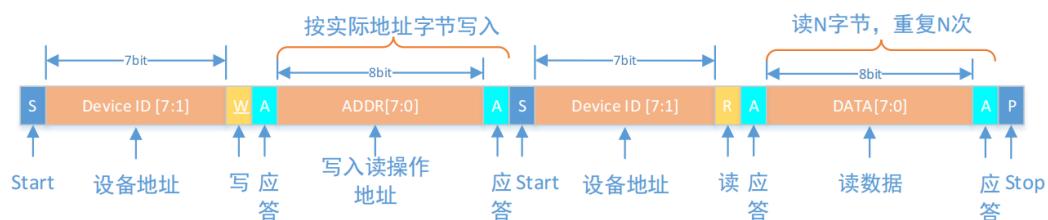
Dummy Write 之后，主设备再发送一次从设备 Device Address Byte，最低位为“1”表示读，Data Word 表示读取从设备数据。

IIC 的读写操作流程可总结为：

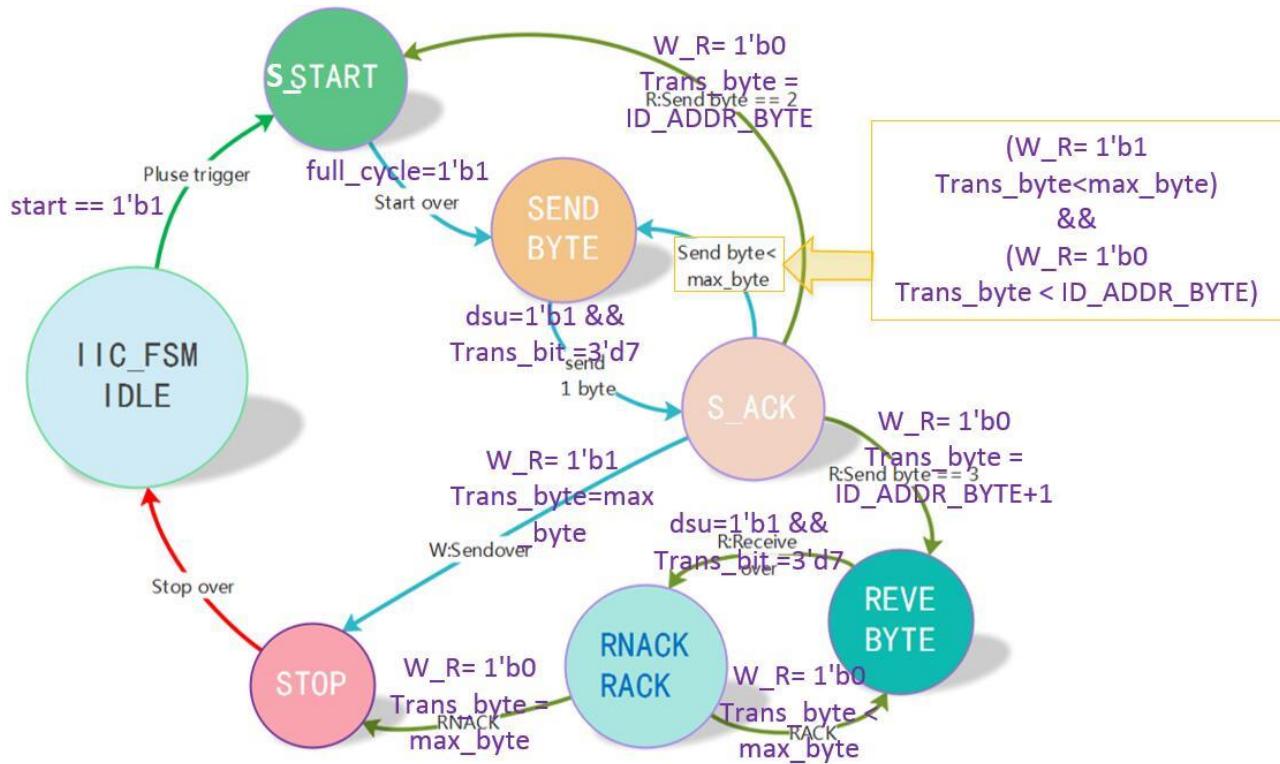
写操作流程



读操作流程



3.3.4 IIC 读写操作流程的状态机实现



默认 IDLE 状态，若接收到开始触发信号，则跳转到 Start 状态；

Start 状态下，在计数器计到 full_cycle 跳转到 send_byte 状态；

send_byte 状态下，发送完 8bit 数据时跳转到 s_ack 状态，接收从设备的应答信号；

s_ack 状态下：若为写操作，byte 数未发送完成 (ID+ADDR+DATA)，则跳转到 send_byte 继续发送下一 byte，若 byte 数已发送完成，则跳转到 stop 状态；若为读操作，Dummy Write 阶段的 byte 数 (ID+ADDR) 未发送完成则跳转到 send_byte 状态继续发送下一 byte，Dummy Write 阶段的 byte 数 (ID+ADDR) 发送完成则跳转到 Start 状态再重新发送一次 device ID，第二次发送完成 device ID 后接收 ack 信号，从 s_ack 状态跳转到 reve_byte 状态。

reve_byte 状态下，若接收完 8bit 数据则跳转到 rack 状态；

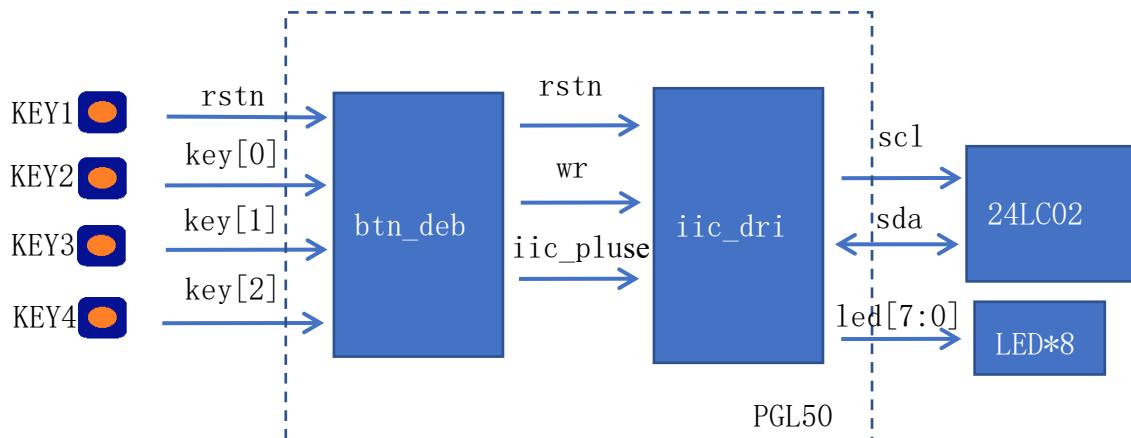
rack 状态下，若需要接收多 byte 数据，为接收完所有 byte 则需要跳转回 reve_byte 状态，若接收完成所有 byte，接收完 ack 应答信号后跳转到 stop 状态；

Stop 状态下，完成 IIC 传输结束标志后跳转到 IDLE 空闲状态；

IIC 读写操作流程的状态机实现，verilog 代码详见参考工程。

3.4 实验源码

实验代码主要框架如下:



KEY1 按下表示复位，KEY2 按下后 wr 信号为 1，表示写；KEY3 按下后 wr 信号为 0，表示读；KEY4 按下表示 IIC 写/读触发开始。

verilog 代码详见参考例程。

3.5 实验现象

KEY1 为复位按键；

写数据：按下 KEY2 后再按一次 KEY4 开始写入 1byte 数据，代码中固定在 8'b0 地址，写入 8'b10101010 数据，可在顶层文件 iic_top.v 中修改此值，如下图：

```
iic_top.v
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123

iic_dri #(
    .CLK_FRE( 27'd50_000_000 ),//parameter
    .IIC_FRE( 20'd400_000 ),//parameter
    .T_WR( 10'd5 ),//parameter
    .DEVICE_ID( 8'hA0 ),//parameter
    .ADDR_BYTE( 2'd1 ),//parameter
    .LEN_WIDTH( 8'd8 ),//parameter
    .DATA_BYT( 2'd1 ) //parameter
)iic_dri(
    .clk( clk ),
    .rstn( rstn ),
    .pulse( iic_pluse ),
    .w_r( wr ),
    .byte_len( 8'd8 ),//input [LEN_WIDTH:0] byte_len,
    .addr( 8'd0 ),//input [7:0] addr,
    .data_in( 8'b10101010 ),//input [7:0] data_in,
    .busy( busy ),
    .byte_over( byte_over ),
    .data_out( data_out ),
    .scl( scl ),
    .sda_in( sda_in ),
    .sda_out( sda_out ),
    .sda_out_en( sda_out_en )
);
```

读数据：按下 KEY3 再按一次 KEY4，从上一次写入的地址中读出数据，LED 灯的状态为写入的数据 8'b10101010。