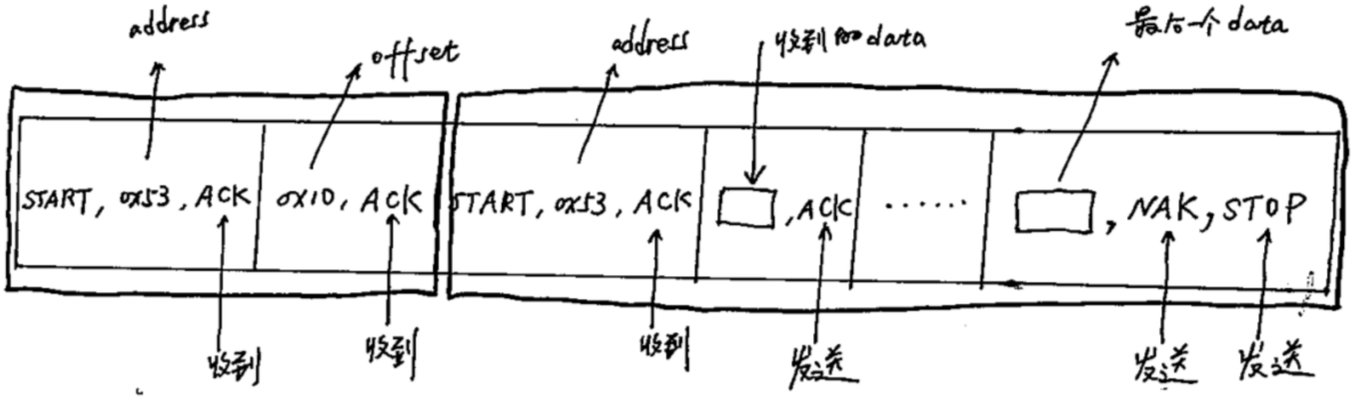
eeprom tool是i2c-tools-3.1.2 package中的一个用于read / write eeprom的工具。

in eeprom.c

```c
/* write len bytes (stored in buf) to eeprom at address addr, page-offset offset */
/* if len=0 (buf may be NULL in this case) you can reposition the eeprom's read-pointer */
/* return 0 on success, -1 on failure */
int eeprom_write(int fd,
                 unsigned int addr,
                 unsigned int offset,
                 unsigned char *buf,
                 unsigned char len
){
        struct i2c_rdwr_ioctl_data msg_rdwr;
        struct i2c_msg             i2cmsg;
        int i;
        char _buf[MAX_BYTES + 1];    ①

        if(len>MAX_BYTES){
            fprintf(stderr,"I can only write MAX_BYTES bytes at a time!\n");
            return -1;
        }

        if(len+offset >256){
            fprintf(stderr,"Sorry, len(%d)+offset(%d) > 256 (page boundary)\n",
                        len,offset);
            return -1;
        }

        _buf[0]=offset;    /* _buf[0] is the offset into the eeprom page! */    ②
        for(i=0;i<len;i++) /* copy buf[0..n] -> _buf[1..n+1] */    ③
            _buf[1+i]=buf[i];

        msg_rdwr.msgs = &i2cmsg;
        msg_rdwr.nmsgs = 1;            ④

        i2cmsg.addr  = addr;
        i2cmsg.flags = 0;              ⑤
        i2cmsg.len   = 1+len;          ⑥
        i2cmsg.buf   = _buf;

        if((i=ioctl(fd,I2C_RDWR,&msg_rdwr))<0){
            perror("ioctl()");
            fprintf(stderr,"ioctl returned %d\n",i);
            return -1;
        }

        if(len>0)
            fprintf(stderr,"Wrote %d bytes to eeprom at 0x%02x, offset %08x\n",
                    len,addr,offset);
        else
            fprintf(stderr,"Positioned pointer in eeprom at 0x%02x to offset %08x\n",
                    addr,offset);

        return 0;
}
```

对eeprom的write并没有分成像下图那样的



I2C device 的 address 为 0x53

从 0x10 offset 处读取 10个 bytes

每发送/接收一个byte，都要有 ACK/NAK 对应

分为两部分(有两个START signal开始)，而是把offset也整合进data里一块儿发送了。

①

#define MAX_BYTES 8 /* max number of bytes to write in one chunk */

eeprom_write()可就收的最多data是8，即len <= 8

这里+1就是为了给offset腾地方

②③

第一个自己是offset,然后才是buf中真正的data

④

由于把offset放入了data，所以只有一个i2c_msg

⑤

0 means write operation

⑥

offset + data len

```c
/* read len bytes stored in eeprom at address addr, offset offset in array buf */
/* return -1 on error, 0 on success */
int eeprom_read(int fd,
                 unsigned int addr,
                 unsigned int offset,
                 unsigned char *buf,
                 unsigned char len
){
        struct i2c_rdwr_ioctl_data msg_rdwr;
        struct i2c_msg             i2cmsg;
        int i;

        if(len>MAX_BYTES){
            fprintf(stderr,"I can only write MAX_BYTES bytes at a time!\n");
            return -1;
        }

        if(eeprom_write(fd,addr,offset,NULL,0)<0)    ①
            return -1;

        msg_rdwr.msgs = &i2cmsg;
        msg_rdwr.nmsgs = 1;            ②

        i2cmsg.addr  = addr;
        i2cmsg.flags = I2C_M_RD;        ③
        i2cmsg.len   = len;
        i2cmsg.buf   = buf;

        if((i=ioctl(fd,I2C_RDWR,&msg_rdwr))<0){
            perror("ioctl()");
            fprintf(stderr,"ioctl returned %d\n",i);
            return -1;
        }

        fprintf(stderr,"Read %d bytes from eeprom at 0x%02x, offset %08x\n",
                len,addr,offset);

        return 0;
}
```

eeprom_read()的len也不能超过MAX_BYTES(8)

①

首先要通知eeprom从哪儿开始read，所以先要write offset这个byte。

②③

接着的就是一个read i2c_msg

eeprom_write()对应的是i2c-dev.c中的一次调用I2C_RDWR ioctrl handler，这样eeprom_read()就对应 2 次调用I2C_RDWR ioctrl handler。

I2C_RDWR ioctrl handler ==> i2cdev_ioctl_rdrw()

Question:

`i2cdev_ioctl_rdrw()会发STOP signal吗？`