1）

偶校验位：$P = D_3 \oplus D_2 \oplus D_1 \oplus D_0$

最终输出：$[D_3, D_2, D_1, D_0, P]$

仿真代码：

```
module even_parity_generator(
    input  [3:0] data_in,   // 4位输入数据
    output parity_bit       // 偶校验位
);
    assign parity_bit = data_in[3] ^ data_in[2] ^ data_in[1] ^ data_in[0];
endmodule
```

原理：

输入 $D_3 D_2 D_1 D_0$	校验位 P	输出
0000	0	00000
0001	1	00011
1010	0	10100
1111	0	11110

2）

1. 模块接口：

输入：

clk：时钟信号，驱动状态机。

rst：异步复位，初始化所有状态。

command：8 位命令输入，支持 START, STOP, F, B, I, S, TUON, TUOF, A, C, D, E。

输出：

forward, backward：控制老鼠移动。

inject, stop_inject：控制咖啡注入。

heat_on, heat_off：控制火源。

error：指示无效命令或状态错误。

position：6 位有符号位置计数器（-16 到+16）。

2. 命令编码：

使用 8 位编码区分命令，A/C/D/E 映射到 F/B/I/S。

示例编码：START=8'h01, STOP=8'h02, F=8'h03, ..., E=8'h0C。

无效命令触发错误状态。

有限状态机：

IDLE：系统暂停，仅接受 START 命令。

EXEC：执行命令，检查状态和边界条件。

ERROR：处理无效命令或状态，仅 STOP 可退出。

3. 安全机制：

位置边界：position 限制在-16 到+16（6 位有符号数）。

状态互斥：注入/停止、加热/关闭避免重复操作。

火源安全：TUON 要求 is_injecting 为 1。

错误处理：无效命令或状态（如边界溢出、重复注入）进入 ERROR 状态。

4. 时序：

所有操作在时钟上升沿同步执行。

输出信号（如 forward、inject 等）为单周期脉冲，适合驱动硬件。

代码实现：

```
always @(posedge clk or posedge rst) begin
    if (rst) begin
        // 复位状态
        state <= IDLE;
        is_running <= 1'b0;
        is_injecting <= 1'b0;
        is_heating <= 1'b0;
        position_r <= 6'b000000;
        forward <= 1'b0;
        backward <= 1'b0;
        inject <= 1'b0;
        stop_inject <= 1'b0;
        heat_on <= 1'b0;
        heat_off <= 1'b0;
        error <= 1'b0;
    end else begin
        // 默认输出
        forward <= 1'b0;
        backward <= 1'b0;
        inject <= 1'b0;
        stop_inject <= 1'b0;
        heat_on <= 1'b0;
        heat_off <= 1'b0;
        error <= 1'b0;

        case (state)
            IDLE: begin
                if (command == CMD_START) begin
                    is_running <= 1'b1;
                    state <= EXEC;
                end else if (command != 8'h00) begin
                    error <= 1'b1; // 非START命令在IDLE状态下报错
                end
            end
```