1.在需要发送信号的class_init中注册信号；

```
static void ipm_timed_shutdown_class_init(IpmTimedShutdownClass *klass)
{
    GObjectClass *object_class = G_OBJECT_CLASS(klass);
    g_type_class_add_private(klass, sizeof(IpmTimedShutdownPrivate));

    g_signal_new("signal-name",    //自定义信号名字
            G_TYPE_FROM_CLASS(object_class);  //GTYPE 定义信号属于IpmTimedShutdown这个类型的对象
            G_SIGNAL_RUN_LAST,    //在用户用g_signal_connect连接的回调函数之后调用，并在用户用g_signal_connect _after连接的回调函数之前调用；
            0 ,
            NULL,
            NULL,
            g_cclosure_marshal_VOID__VOID,  //第一回调函数类型为 typedef void (*callback)  (gpointer instance, gpointer user_data);
            G_TYPE_NONE,    //G_TYPE_NONE表示没有返回值
            0);       //0个额外参数
}
```

2.连接信号

在需要调用函数的地方连接信号

```
static void itep_power_manager_init(ItepPowerManager *o)
{
    o->priv = ITEP_POWER_MANAGER_GET_PRIVATE(o);
    ...
    o->priv->timed_shutdown = ipm_timed_shutdown_new();
    g_siganl_connect(o->priv->timed_shutdown, "signal-name", G_CALLBACK(callback), o);  //o->priv->timed_shutdown 对象 连接 “signal-name"信号， 如果接收到该对象发出的"signal-name"信号就执行 callback 回调函数。
}
```

3.回调函数

```
void callback(IpmTimedShutdown *its, ItepPowerManager *ipm)
{
    ;
}
```

4.发送信号

```
g_signal_emit_by_name(its, "signal-name");
//its 与绑定信号的o->priv->timed_shutdown 必须为同一个对象，发出信号为 “signal-name"
```