5.为quard-star添加RTC和UART

yanglianoo.github.io/2023/06/16/QEMU中自定义开发板5-为quard-star添加RTC和UART

2023年6月16日

1.quard_star.h 修改

```
enum {
    QUARD_STAR_MROM,
    QUARD_STAR_SRAM,
    QUARD_STAR_CLINT,
    QUARD_STAR_PLIC,
    QUARD_STAR_UART0,
    QUARD_STAR_UART1,
    QUARD_STAR_UART2,
    QUARD_STAR_RTC,
    QUARD_STAR_FLASH,
    QUARD_STAR_DRAM,
};

enum {
    QUARD_STAR_UART0_IRQ = 10,  //定义了串口中断号为10
    QUARD_STAR_UART1_IRQ = 11,
    QUARD_STAR_UART2_IRQ = 12,
    QUARD_STAR_RTC_IRQ   = 13,
};
```

2.quard_star.c 修改

quard_star_memmap修改

```
static const MemMapEntry quard_star_memmap[] = {
    [QUARD_STAR_MROM]  = {        0x0,        0x8000 },
    [QUARD_STAR_SRAM]  = {     0x8000,        0x8000 },
    [QUARD_STAR_CLINT] = { 0x02000000,       0x10000 },
    [QUARD_STAR_PLIC]  = { 0x0c000000,     QUARD_STAR_PLIC_SIZE(QUARD
    [QUARD_STAR_UART0] = { 0x10000000,         0x100 },
    [QUARD_STAR_UART1] = { 0x10001000,         0x100 },
    [QUARD_STAR_UART2] = { 0x10002000,         0x100 },
    [QUARD_STAR_RTC]   = { 0x10003000,        0x1000 },
    [QUARD_STAR_FLASH] = { 0x20000000,     0x2000000 },
    [QUARD_STAR_DRAM]  = { 0x80000000,          0x80 },
};
```

RTC

c

```
static void quard_star_rtc_create(MachineState *machine)
{
    QuardStarState *s = RISCV_VIRT_MACHINE(machine);
    sysbus_create_simple("goldfish_rtc",
quard_star_memmap[QUARD_STAR_RTC].base,
        qdev_get_gpio_in(DEVICE(s->plic[0]), QUARD_STAR_RTC_IRQ));
}
```

3路UART

c

```
/* 创建3个 uart */
static void quard_star_serial_create(MachineState *machine)
{
    MemoryRegion *system_memory = get_system_memory();
    QuardStarState *s = RISCV_VIRT_MACHINE(machine);

    serial_mm_init(system_memory,
quard_star_memmap[QUARD_STAR_UART0].base,
        0, qdev_get_gpio_in(DEVICE(s->plic[0]), QUARD_STAR_UART0_IRQ),
399193,
        serial_hd(0), DEVICE_LITTLE_ENDIAN);
    serial_mm_init(system_memory,
quard_star_memmap[QUARD_STAR_UART1].base,
        0, qdev_get_gpio_in(DEVICE(s->plic[0]), QUARD_STAR_UART1_IRQ),
399193,
        serial_hd(1), DEVICE_LITTLE_ENDIAN);
    serial_mm_init(system_memory,
quard_star_memmap[QUARD_STAR_UART2].base,
        0, qdev_get_gpio_in(DEVICE(s->plic[0]), QUARD_STAR_UART2_IRQ),
399193,
        serial_hd(2), DEVICE_LITTLE_ENDIAN);
}
```

machine_init

```c
/* quard-star 初始化各种硬件 */
static void quard_star_machine_init(MachineState *machine)
{
    //创建CPU
    quard_star_cpu_create(machine);
   // 创建主存
    quard_star_memory_create(machine);
    //创建flash
    quard_star_flash_create(machine);
    //创建PLIC
    quard_star_plic_create(machine);
    //创建RISCV_ACLINT
    quard_star_aclint_create(machine);
    //创建三个uart
    quard_star_serial_create(machine);
    //创建 RTC
    quard_star_rtc_create(machine);
}
```

3. Kconfig修改

选中了RTC

c

```
config QUARD_STAR
    bool
    select SERIAL
    select PFLASH_CFI01
    select RISCV_ACLINT
    select RISCV_APLIC
    select SIFIVE_PLIC
    select GOLDFISH_RTC
//RTC
```

4. 测试

在进行测试时，我们需要在qemu的monitor中使用info qtree来查看设备树信息，但是由于qemu无法滚屏，所以不能查看完整的设备树，这里修改了一下run.sh，将monitor映射到了控制台。run.sh修改如下:

shell

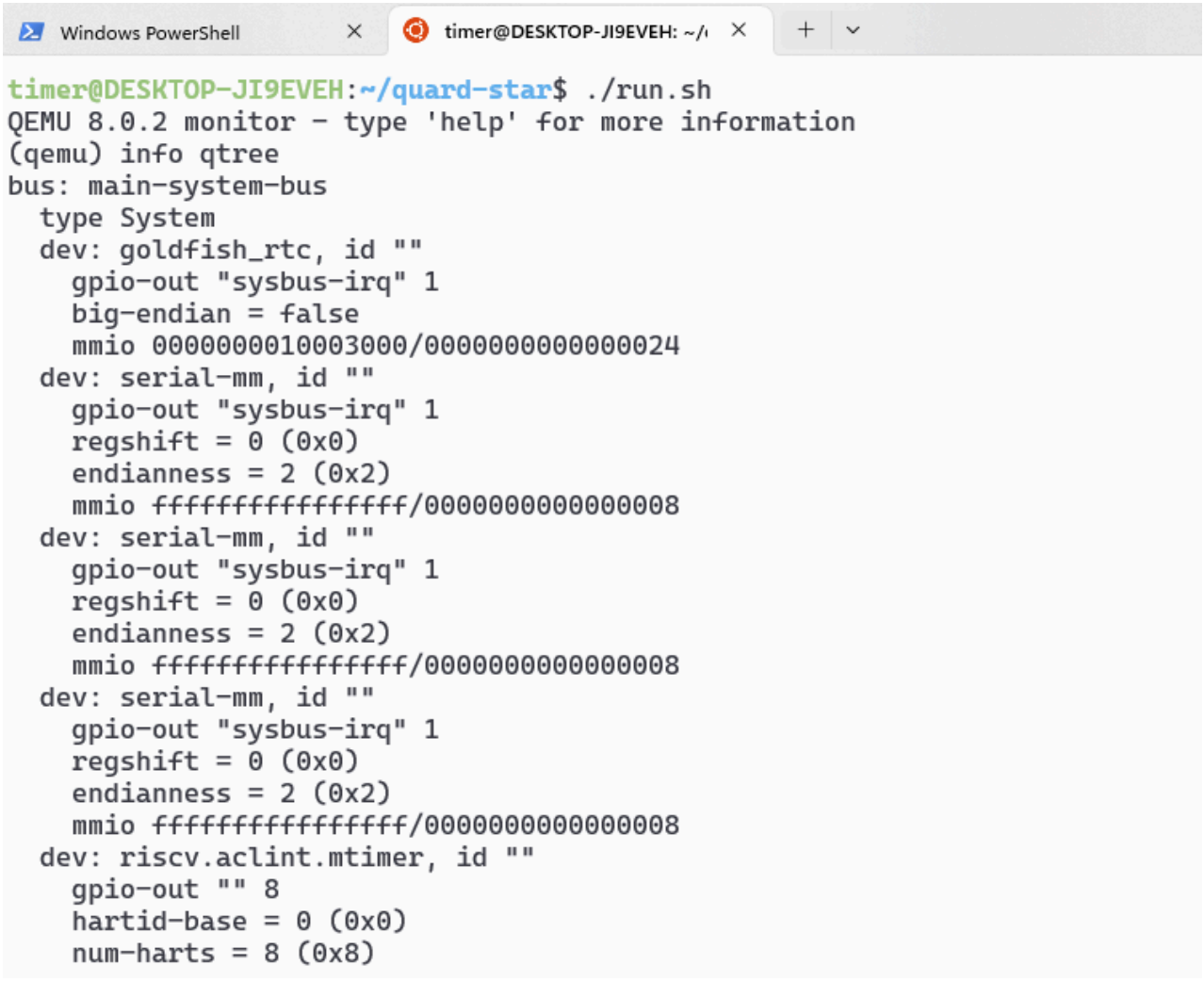
```
SHELL_FOLDER=$(cd "$(dirname "$0")";pwd)

$SHELL_FOLDER/output/qemu/bin/qemu-system-
riscv64 \
-M quard-star \
-m 1G \
-smp 8 \
-bios none \
-monitor stdio \   #映射monitor
```

执行脚本测试：

sh

```
timer@DESKTOP-JI9EVEH:~/quard-star$
./build.sh
timer@DESKTOP-JI9EVEH:~/quard-star$
./run.sh
```

可以看到monitor中输出的内容被映射到控制台中了，且RTC和UART都挂载成功了。



> 代码地址：yanglianoo/quard-star: 从零基于qemu创建riscv嵌入式开发板，并移植操作系统 (github.com)
>
> 有问题请与我联系：wechat：13699648817