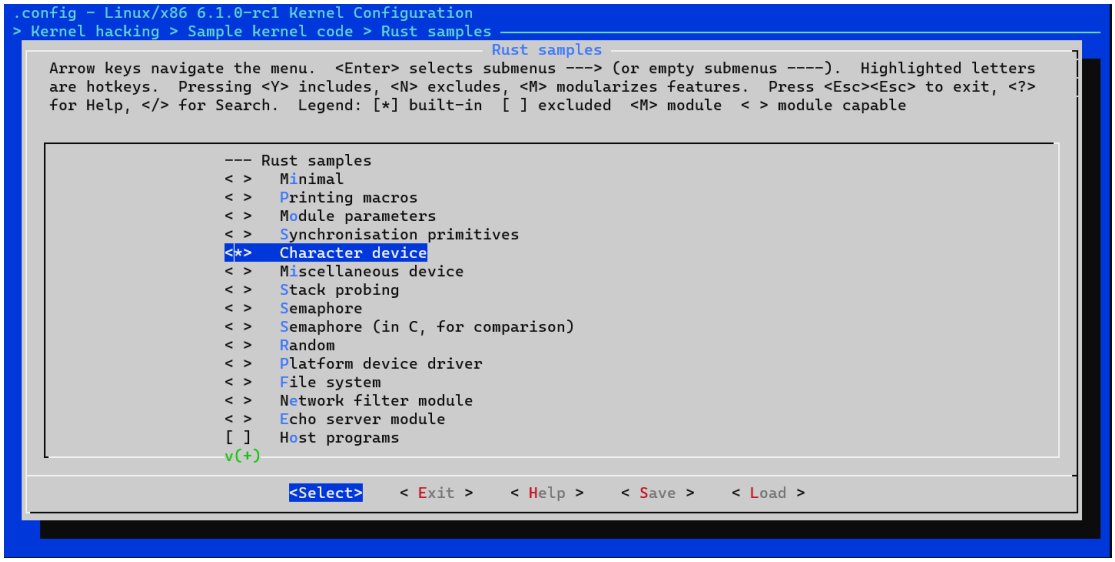
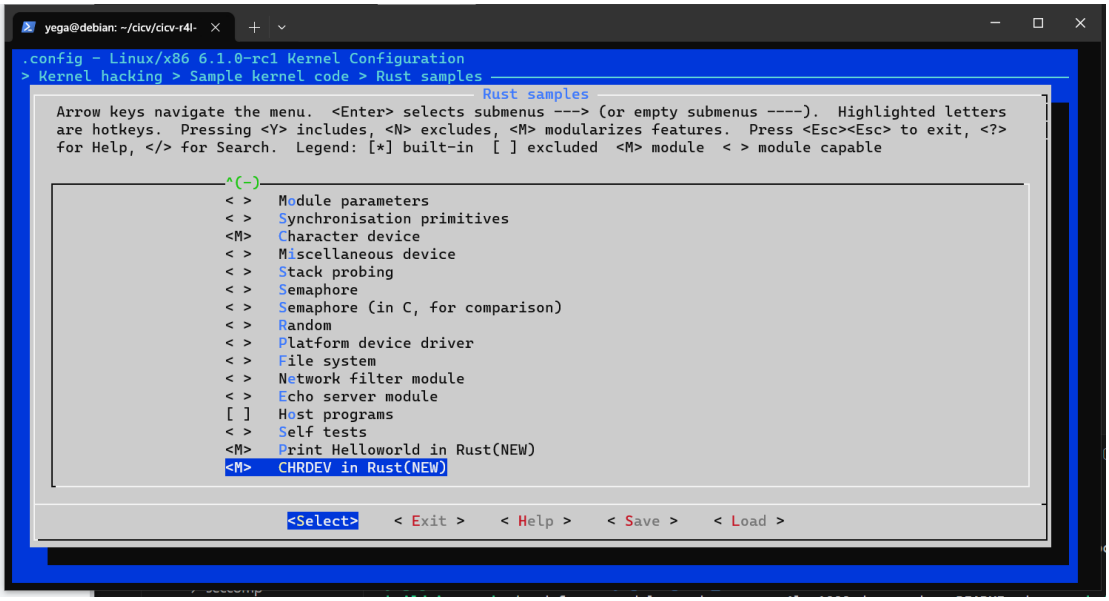
修改 menuconfig





修改 Kconfig

```
config SAMPLE_RUST_CHRDEV
        tristate "CHRDEV in Rust(NEW)"
        help
                This option builds the i/o /dev/cicv cases for Rust.

                If unsure, say N
endif # SAMPLES_RUST
```

修改 Makefile

```
obj-$(CONFIG_SAMPLE_RUST_SELFTESTS)             += rust_selftests.o
obj-$(CONFIG_SAMPLE_RUST_HELLOWORLD)            += rust_helloworld.o
obj-$(CONFIG_SAMPLE_RUST_CHRDEV)                += rust_chrdev.o
```

修改代码

```
41
42      fn write(_this: &Self,_file: &file::File,_reader: &mut impl kernel::io_buffer::IoBuf
43          // Err(EPERM)
44          let mut _reader_len: usize = _reader.len();
45          if _reader_len > GLOBALMEM_SIZE {
46              _reader_len = GLOBALMEM_SIZE;
47          }
48
49          let buf: &mut {unknown} = &mut *_this.inner.lock();
50          _reader.read_slice(data: &mut buf[.._reader_len])?;
51          Ok(_reader_len)
52      }
53
54      fn read(_this: &Self,_file: &file::File,_writer: &mut impl kernel::io_buffer::IoBuff
55          // Err(EPERM)
56          if _offset as usize > GLOBALMEM_SIZE {
57              return Ok(0)
58          }
59
60          let buf: &{unknown} = &*_this.inner.lock();
61          let data: &[u8] = &buf[_offset as usize..];
62          _writer.write_slice(data)?;
63          Ok(data.len())
64      }
65  } impl Operations for RustFile
```

make LLVM=1 -j$(nproc) 编译内核，复制编译生成的 ko 文件到 rootfs 文件夹下，启动 qemu.

确认设备存在

```
Please press Enter to activate this console.
~ # ls /dev/cicv
/dev/cicv
~ #
```

```
[   14.027284] mdev (74) used greatest stack depth: 13928 bytes left
[   14.029994] mknod (75) used greatest stack depth: 13920 bytes left

Please press Enter to activate this console.
~ # insmod rust_chrdev.ko
[   26.585860] rust_chrdev: Rust character device sample (init)
[   26.586794] insmod (80) used greatest stack depth: 13816 bytes left
~ # echo "Hello" > /dev/cicv
~ # cat /dev/cicv
Hello
~ #
```

Q: 作业 5 中的字符设备/dev/cicv 是怎么创建的？它的设备号是多少？它是如何与我们写的字符设备驱动关联上的？

答:

1.通过 mknod 创建的 mknod /dev/cicv c 248 0" >> etc/init.d/rcS

2. /dev/cicv 的主设备号是 248，次设备号是 0。

3.mknod 创建了字符设备，当驱动挂载时通过注册和加载设备驱动可控制两个字符设备。内核就可以实现对硬件的控制和管理。