



guidance

☰ Tags

1.fail a test , type script and make qemu, which log all console output to a file.

and don't forget to exit script

```
14:28:40
• [~] script
  Script started, file is typescript
  [~] ls
  clash id_rsa id_rsa.pub mxy typescript xsr xv6
  [~] exit script
  Script done, file is typescript
```

```
15:02:50
• [~] cat typescript
  Script started on 2023-02-16 15:01:59+08:00 [TERM="xterm-256color" TTY="/dev/pts/11" COLUMNS="102" LINES="19"]
  [~] ls
  clash id_rsa id_rsa.pub mxy typescript xsr xv6
  [~] exit script
  Script done on 2023-02-16 15:02:20+08:00 [COMMAND_EXIT_CODE="0"]
```

1.5 gdb remote

```

0x0000000a <start+28> csrw mstatus,a5
0x0000000b <start+32> auipc a5,0x1
0x0000000c <start+36> addi a5,a5,-538
0x0000000d <start+40> csrw mpc,a5
0x0000000e <start+44> li a5,0
0x0000000f <start+48> csrw satp,a5
0x00000010 <start+52> lui a5,0x18
0x00000011 <start+56> addi a5,a5,-1
0x00000012 <start+60> csrw mdeleg,a5
0x00000013 <start+64> csrw mideleg,a5
0x00000014 <start+68> csrw a5,sie
0x00000015 <start+72> ori a5,a5,546
0x00000016 <start+76> csrw sie,a5
0x00000017 <start+80> li a5,-1
0x00000018 <start+84> li a5,-1

remote Thread 1.1 In: start L145 PC: 0x0000000c
(gdb) si
0x000000000000000a in w_mstatus (s=<optimized out>) at kernel/riscv.h:31
w_mpc (s=<optimized out>) at kernel/riscv.h:48
w_satp (s=<0>) at kernel/riscv.h:289
w_mdeleg (s=<05535>) at kernel/riscv.h:145
(gdb)

echo 2 4 31192
furtest 2 5 31200
grep 2 6 35672
init 2 7 31984
kill 2 8 31192
in 2 9 31112
is 2 10 34232
mkdir 2 11 31232
rm 2 12 31216
sh 2 13 34464
stressfs 2 14 32008
usertest 2 15 180576
grind 2 16 47236
wc 2 17 33312
rm 2 18 30768
console 3 19 0
$ QEMU: Terminated
[xv6-riscv] qemu-gdb 19:28:48 riscv
*** Now run 'gdb' in another window.
qemu-system-riscv64 -machine virt -bios none -kernel kernel -s 128M -sp
3 -nographic -global virtio-mio,force-legacy=false -drive file=fs.img,if=none
,format=raw,id=x0 -device virtio-blk-device,drive=x0,bus=virtio-mio-bus.0 -S -
gdb tcp:125000
QEMU: Terminated
[xv6-riscv] qemu-gdb 19:28:48 riscv
*** Now run 'gdb' in another window.
qemu-system-riscv64 -machine virt -bios none -kernel kernel -s 128M -sp
3 -nographic -global virtio-mio,force-legacy=false -drive file=fs.img,if=none
,format=raw,id=x0 -device virtio-blk-device,drive=x0,bus=virtio-mio-bus.0 -S -
gdb tcp:125000

```

```

[xv6-riscv] riscv64-linux-gnu-gdb 19:16:51 riscv
230: command not found: riscv64-linux-gnu-gdb
[xv6-riscv] qemu-gdb 19:17:13 riscv
GNU gdb (Ubuntu 9.2-0ubuntu1-20.04.1) 9.2
Copyright (C) 2020 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word".
The target architecture is assumed to be riscvrv64.
warning: No executable has been specified and target does not support
determining executable automatically. Try using the "file" command.
--Type <RET> for more, q to quit, c to continue without paging--c
0x0000000000000000 in ?? ()
(gdb)

19:28:17 riscv
[xv6-riscv] riscv
19:28:20 riscv
[xv6-riscv] qemu-gdb
19:28:36 riscv
*** Now run 'gdb' in another window.
qemu-system-riscv64 -machine virt -bios none -kernel kernel -s 128M -sp
3 -nographic -global virtio-mio,force-legacy=false -drive file=fs.img,if=none
,format=raw,id=x0 -device virtio-blk-device,drive=x0,bus=virtio-mio-bus.0 -S -
gdb tcp:125000
QEMU: Terminated
[xv6-riscv] qemu-gdb
19:28:48 riscv
*** Now run 'gdb' in another window.
qemu-system-riscv64 -machine virt -bios none -kernel kernel -s 128M -sp
3 -nographic -global virtio-mio,force-legacy=false -drive file=fs.img,if=none
,format=raw,id=x0 -device virtio-blk-device,drive=x0,bus=virtio-mio-bus.0 -S -
gdb tcp:125000

```

make qemu-gdb will open qemu-remote-debug on TCP:port .And run gdb(-multiarch in another window” can get the debug info.

2.see the /kernel/kernel.asm to find out what the assembly he compiler generates.

```

000000008000008e <start>:
{
    8000008e: 1141          addi sp,sp,-16
    80000090: e406          sd ra,8(sp)
    80000092: e022          sd s0,0(sp)
    80000094: 0800          addi s0,sp,16
    asm volatile("csrr %0, mstatus" : "=r" (x) );
    80000096: 300027f3      csrr a5,mstatus
    x &= ~MSTATUS_MPP_MASK;
    8000009a: 7779          lui a4,0xffffe
    8000009c: 7ff70713      addi a4,a4,2047 # fff
    800000a0: 8ff9          and a5,a5,a4
    x |= MSTATUS_MPP_S;
    800000a2: 6705          lui a4,0x1
    800000a4: 80070713      addi a4,a4,-2048 # 800
    800000a8: 8fd9          or a5,a5,a4
    asm volatile("csrw mstatus, %0" : "=r" (x));
    800000aa: 30079073      csrw mstatus,a5
}

```

3. When kernel panic: a function will print the value of program counter.

addr2line - convert addresses into file names and line numbers

addr2line -e kernel/kernel pc-value

```

???:0
[xv6-riscv] addr2line -e kernel/kernel 800000c4
/root/xv6/xv6-riscv/kernel/riscv.h:145
[xv6-riscv]

```

(pc-value 是前面的asm地址)

bt: backtrace can get all func_call while running. With it we can get all pc-value and the real func_code.

4. deadlock

When the kernel appears to hang hit Ctrl-C in the qemu-gdb window and type 'bt' to get a backtrace.

5. qemu's monitor ,type ctrl-a c.

usage: info mem :print the page table (type cpu to select the core)