Installing GRUB on a hard disk image file

- Go to directory MK
- Look at file pack.sh
- Understand the meaning of each script command

- Run sudo ../pack.sh DIM (ex. 520)

  to create a grub bootable HD with a partition of dimension

  DIM Kb, without kernel

Installing GRUB on a hard disk image file

- **Check / install assembler program nasm**
- **Run ./build.sh**

  to make the kernel on /src/kernel

Debugging GRUB and the kernel

- Check / install the machine emulator and virtualizer qemu
- Check / install the debugger ddd
- Check that a copy (of files .gdbinit and build.sh is in the directory or a symbolic link to the files located in MK)

- Run ../qemu hd.img

- Understand the meaning of the script ../qemu.sh
- Run ../qemu.sh hd.img

- Open another terminal and run ddd &

- NOTE: Edit the File browser preferences
  - on TAB Media deselect Brows media when inserted

Debugging GRUB

- In ddd, display the machine code window
- Follow the first bootstrap instructions with the aid of the slides in file Booting a PC_with_GRUB.pdf
- set a breakpoint at 0c7c00
  - b *0x7c00
- set a watchpoint to the VGA display, which is mapped in memory at address 0xB8000
  - awatch *(char *) 0xB8000
- continue - starts the bootstrap
  - c

Debugging GRUB

- Use Nexti to execute the current instruction
- see the executed instructions in the listing of Grub-Stage1
  - Nexti up to the instruction in address 0x7C70
- Look at the content of register SI: it should be 7D7F
- Examine the characters GRUB that will be displayed by the next call
  - x\4c 0x7D7F examine 4 chars
  - Nexti displays GRUB
- Set a breakpoint to the kernel entry point in main.c
- b main
- Set a breakpoint to the kernel entry point in 0x8130
  - b *0x8130
  - c several times will print a character at a time the string Loading Grub stage 2 ...
- then the graphic menu of GRUB appears, click on it, ddd will stop at main

Debugging the kernel

- **The kernel is a program that can be debugged by ddd**
- **You can examine the content of the registers and variables, and debug both C and assembler code.**