```
ITI_44 => printenv bootcmd
bootcmd=echo Hello
ITI_44 => setenv bootcmd fatload mmc 0:1 0x60000000 main.elf ; md 0x60000000
```

```
Hit any key to stop autoboot:  0
11512 bytes read in 15 ms (749 KiB/s)
ITI_44 => md 0x60000000
60000000: 464c457f 00010101 00000000 00000000  .ELF............
60000010: 00530002 00000001 00000000 00000034  ..S.........4...
60000020: 00002b18 00000005 00200034 00280002  .+......4. ...(.
60000030: 000b000c 00000001 00000074 00000000  ........t.......
60000040: 00000000 000014a6 000014a6 00000005  ................
60000050: 00000002 00000001 0000151a 00800060  ............`...
60000060: 000014a6 00000040 00000040 00000006  ....@...@.......
60000070: 00000001 002a940c 003f940c 003f940c  ......*...?...?.
60000080: 003f940c 003f940c 003f940c 003f940c  ..?...?...?...?.
60000090: 003f940c 003f940c 003f940c 003f940c  ..?...?...?...?.
600000a0: 003f940c 003f940c 003f940c 003f940c  ..?...?...?...?.
600000b0: 003f940c 003f940c 003f940c 003f940c  ..?...?...?...?.
600000c0: 003f940c 003f940c be1f2411 e0d8e5cf  ..?...?..$......
600000d0: bfcdbfde e6a0e010 eae6e0b0 c002e1f4  ................
600000e0: 920d9005 07b13aa0 940ef7d9 940c0041  .....:......A...
600000f0: 940c0a51 93cf0000 b7cd93df d5aeb7de  Q...............
```

During the bootloader stage, the bootloader itself is responsible for interacting with the hardware. The bootloader is a small program that is loaded into memory by the firmware (such as BIOS or UEFI) during the initial stages of the boot process. Its primary purpose is to locate, load, and transfer control to the operating system kernel.

In the bootloader stage of the Linux boot process, the bootloader itself interacts directly with the hardware. The bootloader is a small program that is responsible for loading the operating system kernel into memory and transferring control to it. This interaction occurs before the full kernel is loaded and initialized.

The bootloader's primary responsibilities include:

1. **Loading the Kernel:**

   -The bootloader locates the kernel image on the storage device (such as a hard drive or SSD).

-It reads the kernel into memory, preparing it for execution.

2. **Initializing Basic Hardware:**

   -The bootloader may perform basic hardware initialization, depending on the specific bootloader and hardware architecture.
   -For example, it might set up memory, configure display settings (text mode), and handle input devices to some extent.

3. **Transferring Control to the Kernel:**

   -Once the kernel is loaded into memory, the bootloader transfers control to the kernel by jumping to its entry point.
   -At this point, the kernel takes over the responsibility of interacting with and managing the hardware.

It's important to note that the bootloader's interaction with hardware during this stage is relatively minimal compared to what the kernel will later do. The bootloader's role is to perform just enough initialization and loading of the kernel to enable the transition from the bootloader stage to the kernel execution.