NIKHILESH KUMAR REDDY ANAM

Q1.

a. Boot ROM loaded by QEMU at the address 0x1000.
b. Steps taken by the loaded Boot ROM is it first initializing the stack pointer by pointing it to the start address of bl_stack + 4KB and loads it to sp and then loads 4KB to the register a0 next it and then it reads from the control and status register csrr the mhartid which is the number of cores to a1 and then adds 1 to a1 and then we multiply it by a0 and then add a0 to the stack pointer sp and it jumps to this final calculated address (and then calls start in start.c).
c. Boot ROM jumps to the address 0x80000000 at the end of its execution.

Q2.

a. _entry is the specified entry function of the bootloader in the linker descriptor
b. The code at _entry loads the stack pointer register sp with the address stack0+4096 if the linker descriptor is correctly specified.

Q3.

a. If we jump to C code without setting up the stack sp goes to the negative address so it cannot access due to memory error because there will be no place for the C program to store the variables.
b. If there are 2 CPUs instead of one then bl_stack will be allocated 8KB of memory with one stack pointer pointing to the start address of bl_stack + 4KB and the other pointing to the start address of bl_stack + 8KB.

Q4.

a. Since there is no status initially we read the value of mstatus and then perform a bitwise-and with value of mstatus and the bitwise-not of MSTATUS_MPP_MASK to set the bit corresponding to M-mode as 0. Then we perform a bitwise-or operation to the above result and MSTATUS_MPP_S to set the bit corresponding to S-mode as 1. So the MPP field in the register gets updated from 0 to 1.

Q5.

a. The function kernel_copy checks if there is sufficient space in the filesystem to load the kernel and if isn't it goes to a loop which is busy else it calculates the diskaddr and the calculates the addr based on diskaddr and b->data which are destination and source address, grabs the BSIZE and then it calls the memmove function which is essentially memcpy to copy the kernel.