1. (2 points) Your hardware device, like any other, has a very limited number of hardware timers. Does that limit extend to the number of hrtimers that can be created? Why or why not?

The number of hardware timer does not extend to the number of hrtimers. This is because hrtimers are based on software. So theoretically, there is no limit for the number of hrtimers. Hardware timers are based on hardware so the number is limited and cannot extend to the number of hrtimers.

2. (2 points) Assume a thread makes a blocking file read I/O call and the OS does not have the data handy in memory. The OS blocks and deschedules the thread until the data arrives from disk. When the data does arrive, how does the OS know which thread to wake up? Which kernel mechanism is used?

When the data arrives from the disk, a hardware interrupt will be issued and the OS will wake some specific threads. On the other hand, when a thread is going to sleep, a wait_queue_t structure will be initialized. When handling the data arriving interrupt, the OS can get information about the sleeping thread from the wait_queue_t structure and wake up the thread according to the relative interrupt. This mechanism used is wait queue kernel.

3. (2 points) Periodic work in the kernel can be performed by adding it to a work queue or to an hrtimer callback. What is a work queue handler allowed to do that an hrtimer handler is not?

The work queue handler allows sleep and wait but hrtimer callback cannot.

4. (2 points) What is the difference between regular signals and real-time signals?

First, there can be only one regular signal be queued. but more than one real-time signal can be queued if they are blocked.
Second, the delivery order of regular signals cannot be guaranteed, but real-time signals are guaranteed to be delivered in the same order as sending.
Third, regular signal cannot carry any other additional data, but real-time signals can.

5. (2 points) How do Android applications work without a main function?

Android application doesn't need main function because its entry point is specified in the manifest file. In this file, android.intent.action.MAIN is used to define the entry of the application. The activity which is set as Android.intent.category.LAUNCHER will start the execution of the application from its onCreat() method.