Export CPU topology info via sysfs. Items (attributes) are similar
to /proc/cpuinfo.

1) /sys/devices/system/cpu/cpuX/topology/physical_package_id:

        physical package id of cpuX. Typically corresponds to a physical
        socket number, but the actual value is architecture and platform
        dependent.

2) /sys/devices/system/cpu/cpuX/topology/core_id:

        the CPU core ID of cpuX. Typically it is the hardware platform's
        identifier (rather than the kernel's). The actual value is
        architecture and platform dependent.

3) /sys/devices/system/cpu/cpuX/topology/thread_siblings:

        internel kernel map of cpuX's hardware threads within the same
        core as cpuX

4) /sys/devices/system/cpu/cpuX/topology/core_siblings:

        internal kernel map of cpuX's hardware threads within the same
        physical_package_id.

To implement it in an architecture-neutral way, a new source file,
drivers/base/topology.c, is to export the 4 attributes.

For an architecture to support this feature, it must define some of
these macros in include/asm-XXX/topology.h:
#define topology_physical_package_id(cpu)
#define topology_core_id(cpu)
#define topology_thread_cpumask(cpu)
#define topology_core_cpumask(cpu)

The type of **_id is int.
The type of siblings is (const) struct cpumask *.

To be consistent on all architectures, include/linux/topology.h
provides default definitions for any of the above macros that are
not defined by include/asm-XXX/topology.h:
1) physical_package_id: -1
2) core_id: 0
3) thread_siblings: just the given CPU
4) core_siblings: just the given CPU

Additionally, CPU topology information is provided under
/sys/devices/system/cpu and includes these files. The internal
source for the output is in brackets ("[]").

    kernel_max: the maximum CPU index allowed by the kernel configuration.
                [NR_CPUS-1]

    offline: CPUs that are not online because they have been
                HOTPLUGGED off (see cpu-hotplug.txt) or exceed the limit

of CPUs allowed by the kernel configuration (kernel_max above). [~cpu_online_mask + cpus >= NR_CPUS]

online: CPUs that are online and being scheduled [cpu_online_mask]

possible: CPUs that have been allocated resources and can be brought online if they are present. [cpu_possible_mask]

present: CPUs that have been identified as being present in the system. [cpu_present_mask]

The format for the above output is compatible with cpulist_parse() [see <linux/cpumask.h>]. Some examples follow.

In this example, there are 64 CPUs in the system but cpus 32-63 exceed the kernel max which is limited to 0..31 by the NR_CPUS config option being 32. Note also that CPUs 2 and 4-31 are not online but could be brought online as they are both present and possible.

```
kernel_max: 31
   offline: 2,4-31,32-63
    online: 0-1,3
  possible: 0-31
   present: 0-31
```

In this example, the NR_CPUS config option is 128, but the kernel was started with possible_cpus=144. There are 4 CPUs in the system and cpu2 was manually taken offline (and is the only CPU that can be brought online.)

```
kernel_max: 127
   offline: 2,4-127,128-143
    online: 0-1,3
  possible: 0-127
   present: 0-3
```

See cpu-hotplug.txt for the possible_cpus=NUM kernel start parameter as well as more information on the various cpumasks.