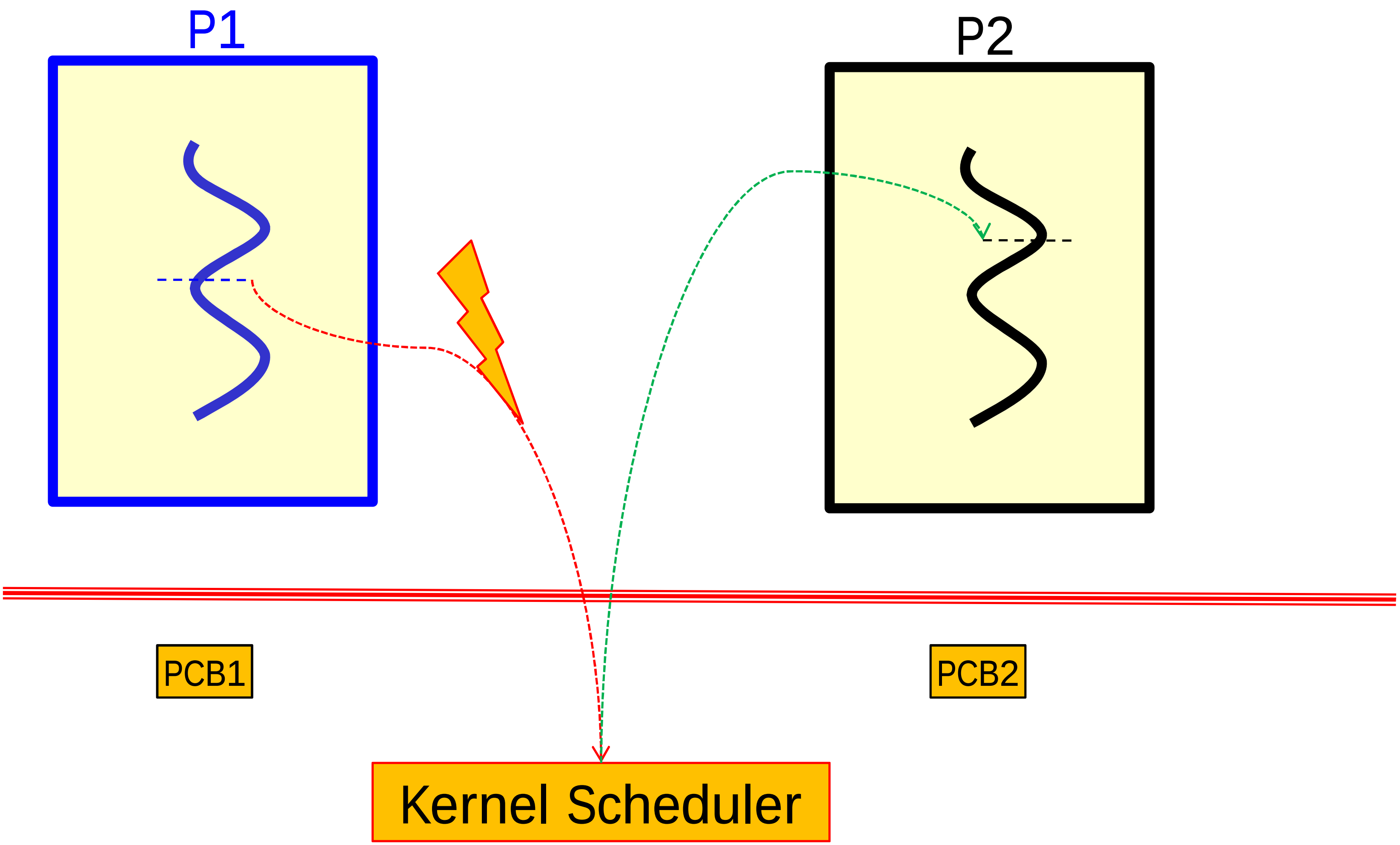
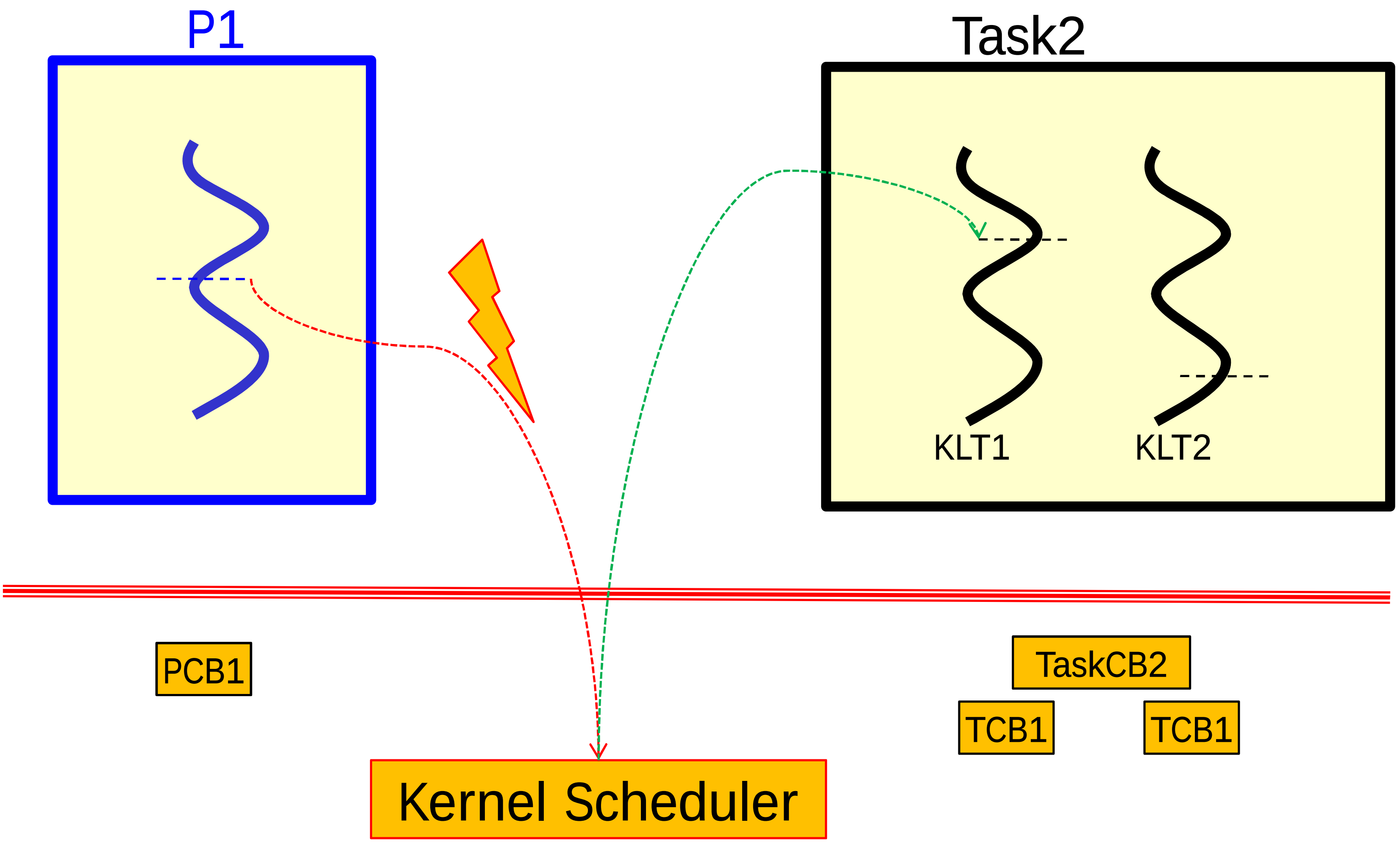
Activity Switch (1) Process-Switch



Activity Switch (2): Process-Task



Blocking System Call?

- 2 major classes:
  - Waiting for work to be done on the peripheral or
  - Waiting for work to be done by some other KLT or process

- General template:
  1. Enter kernel
  2. If the desired service not yet done then
     - block caller and wait until desired event will happen
     - switch to some other executable activity (e.g. another process, another KLT)
  3. Exit kernel

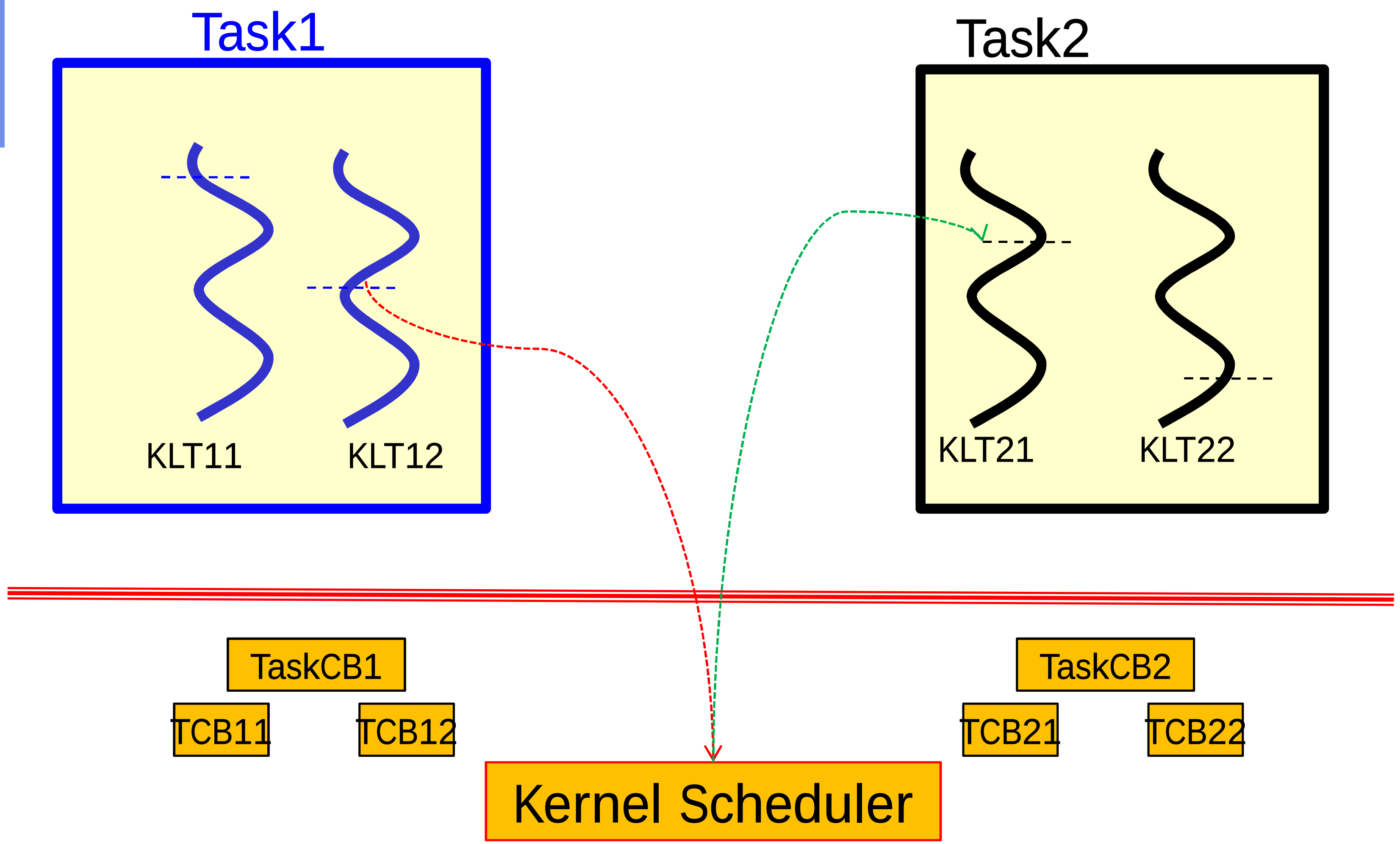
Blocking I/O System Call

1. Does some initial work in the kernel to prepare the desired I/O service

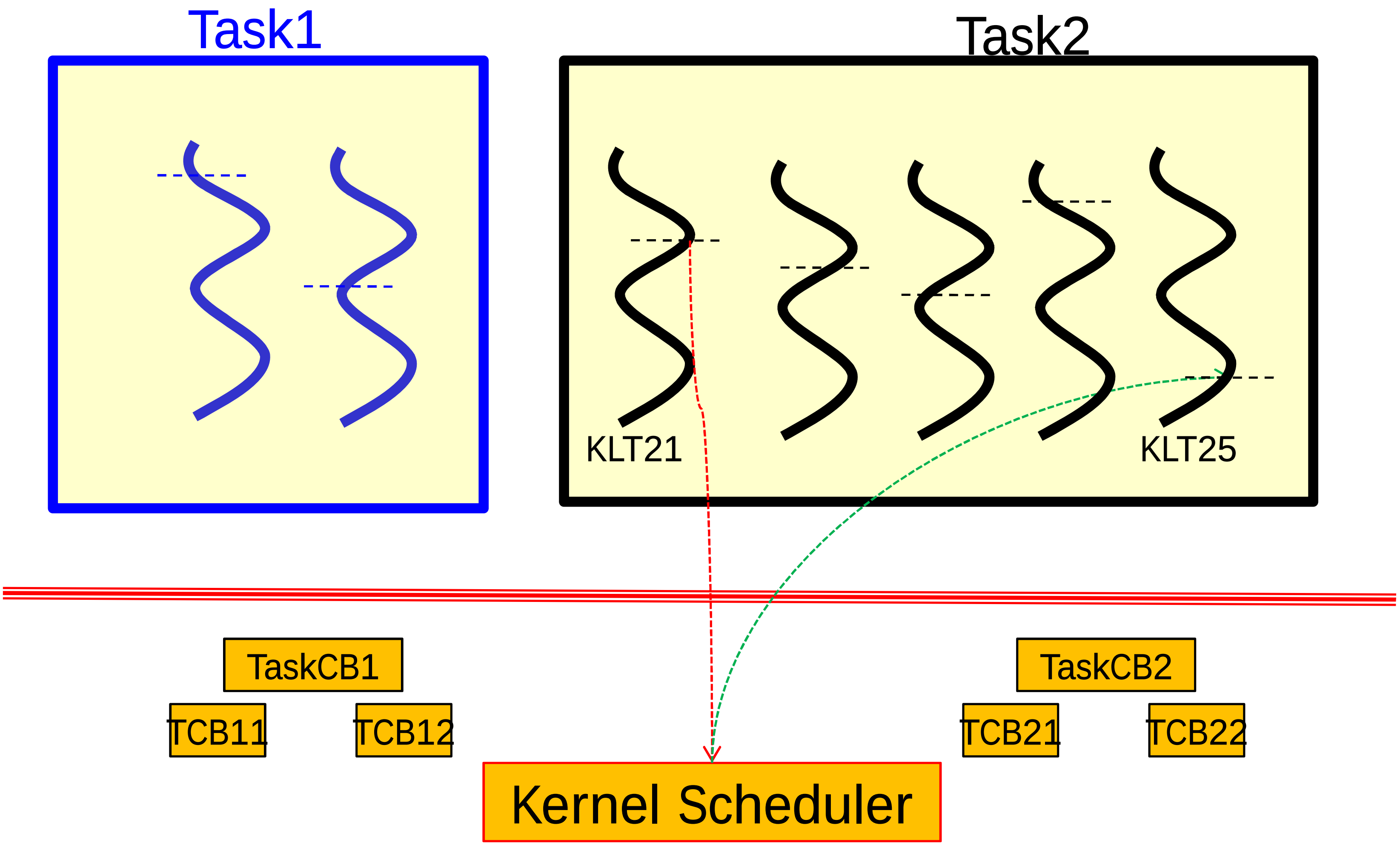
2. Starts service running on a peripheral device by updating peripheral registers, etc.

- Blocks the calling user-land activity, e.g. the KLT (or process) and induces an activity_switch, i.e.
  - a thread_switch to another KLT of the same task or of another task or
  - a process_switch to another process

Cross-AS KLT Switch

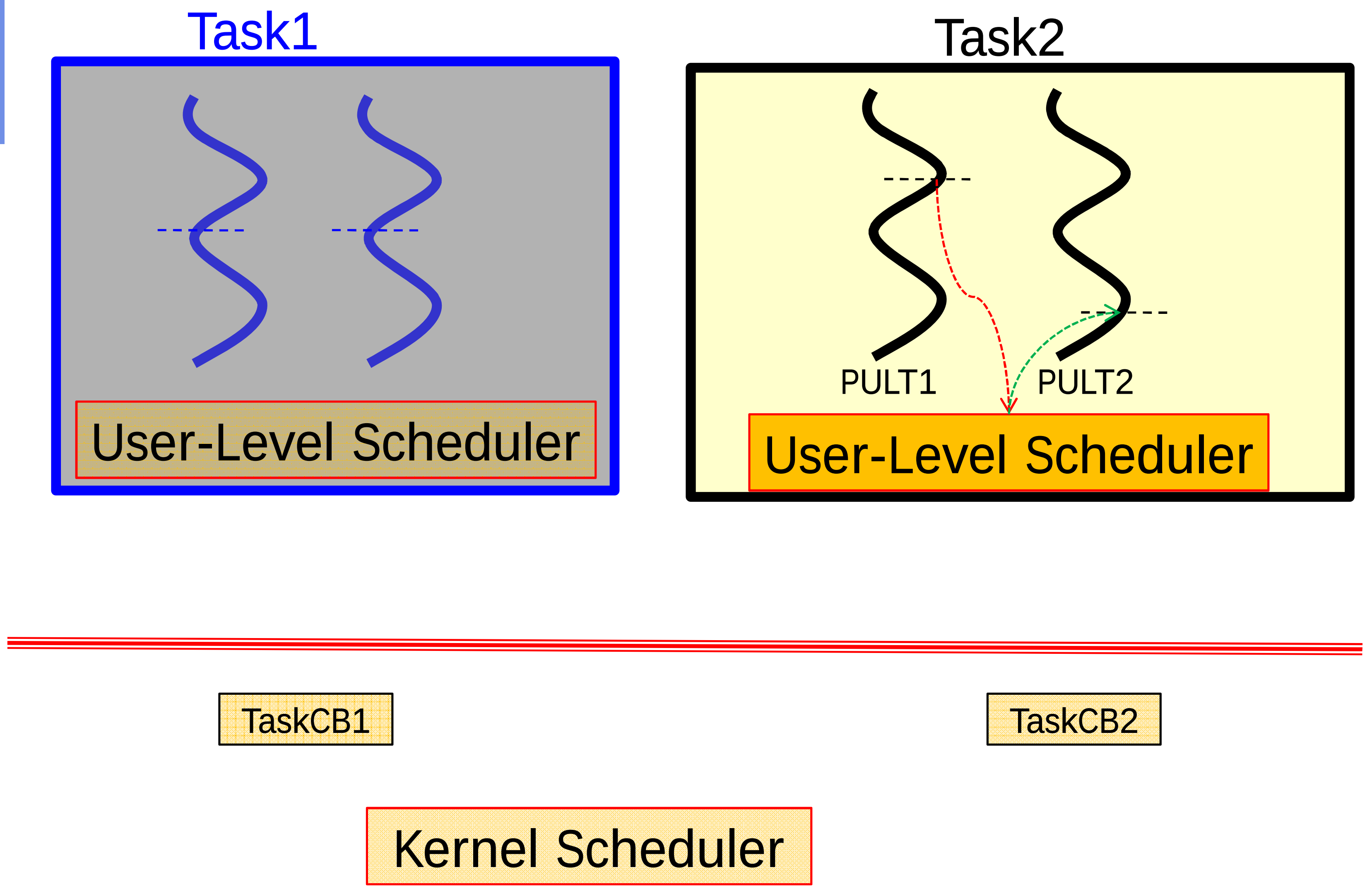


AS-Internal KLT Switch



© 2007 Universität Karlsruhe(TH), System Architecture Group

AS-Internal PULT Switch



Task1

User-Level Scheduler

Task2

PULT1

PULT2

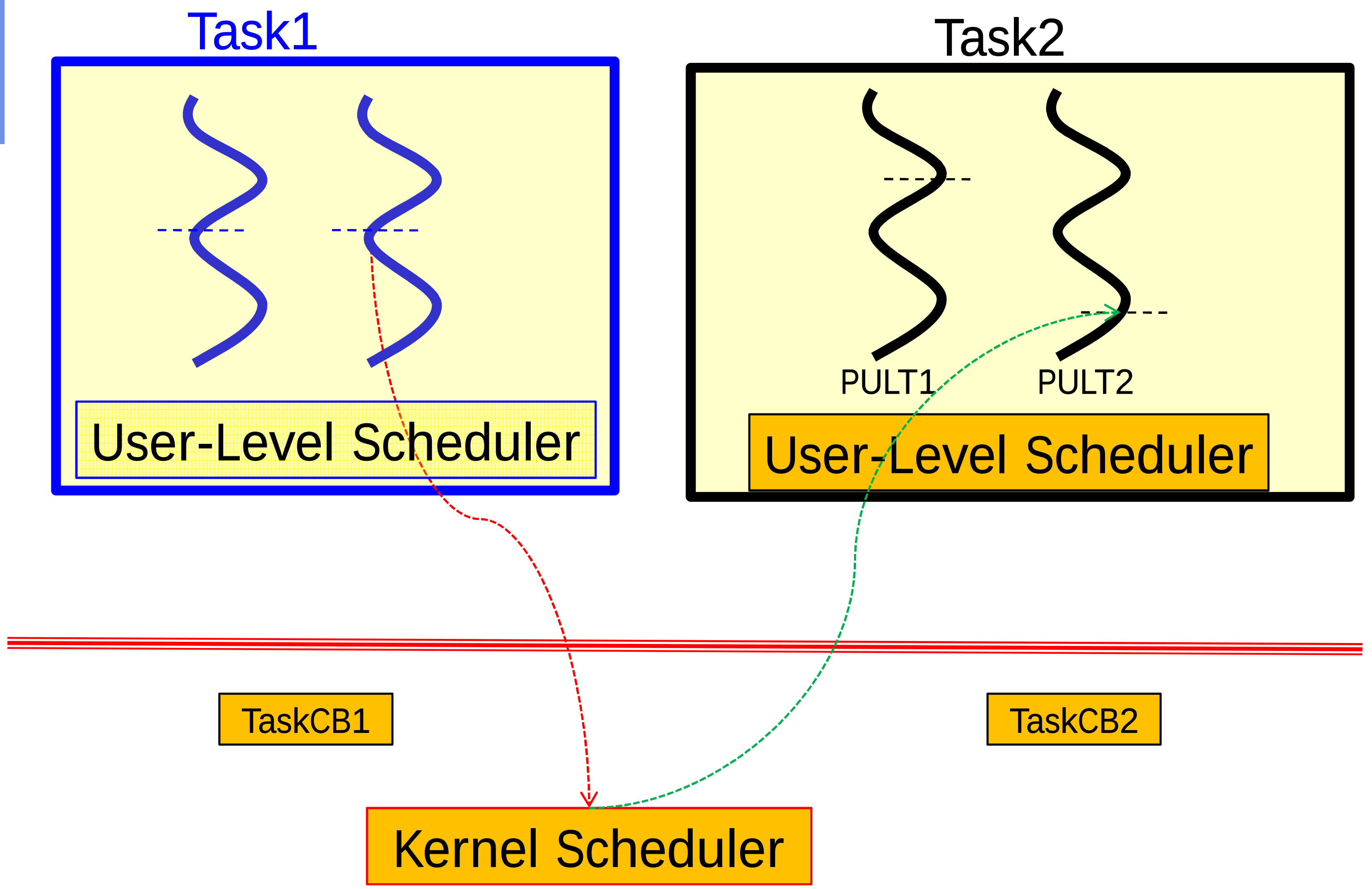
User-Level Scheduler

TaskCB1

TaskCB2

Kernel Scheduler

Cross-AS PULT Switch



Thread-Save Programming

- Threads share their AS
  - Code of a thread should be reentrant, i.e. it should also work if executed multiple by another thread
    - Threads should not overwrite global variables mutually, but should use only local variables
    - Local variables are pushed onto their stacks
  - Per thread there must exist a private global variable `errno`