Amirkabir University of Technology
(Tehran Polytechnic)

Operating Systems

Processes-Part2

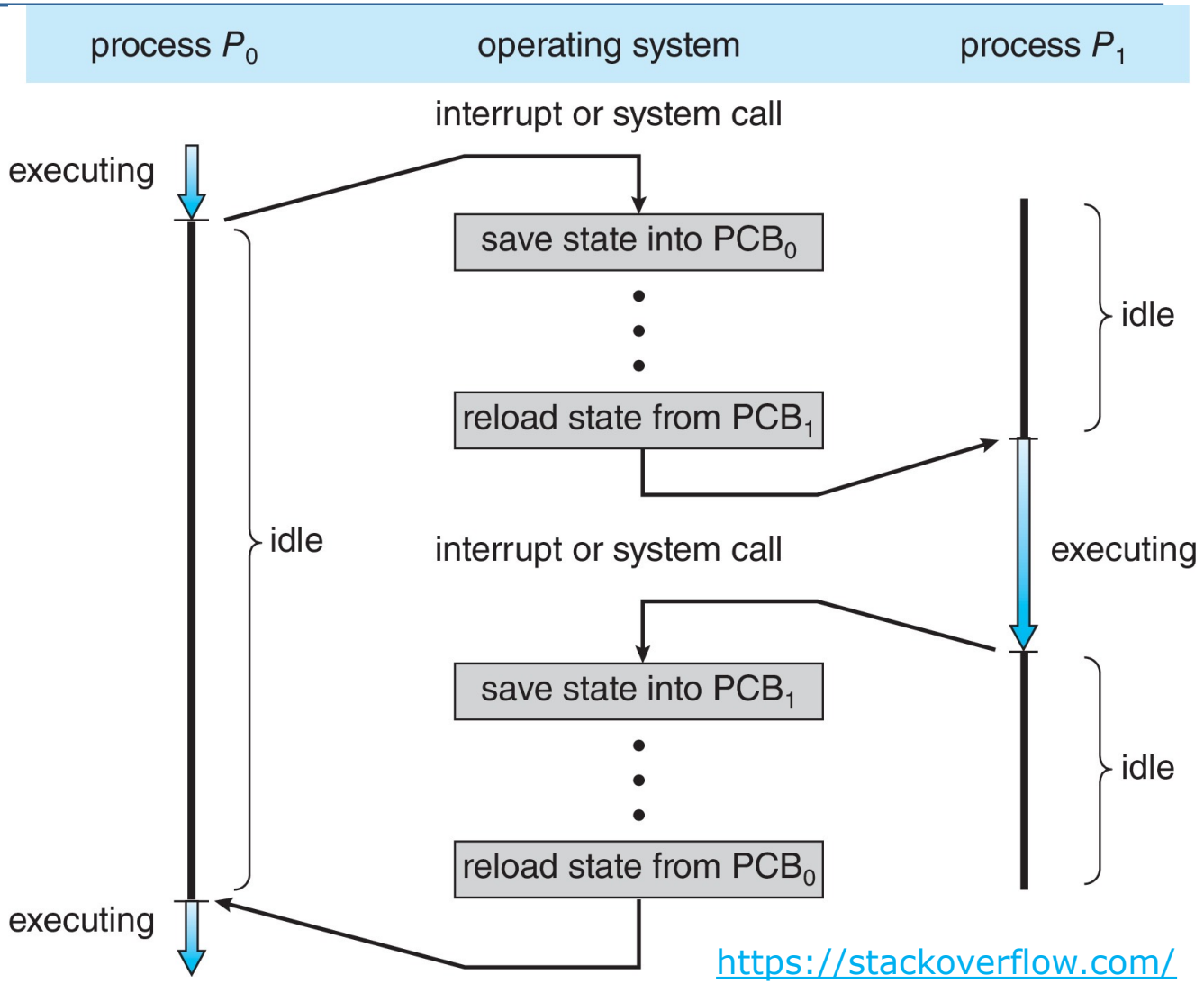
Seyyed Ahmad Javadi

sajavadi@aut.ac.ir

Fall 2021

CPU Switch From Process to Process

A **context switch** occurs when the CPU switches from one process to another.



https://stackoverflow.com/questions/9238326/system-call-and-context-switch

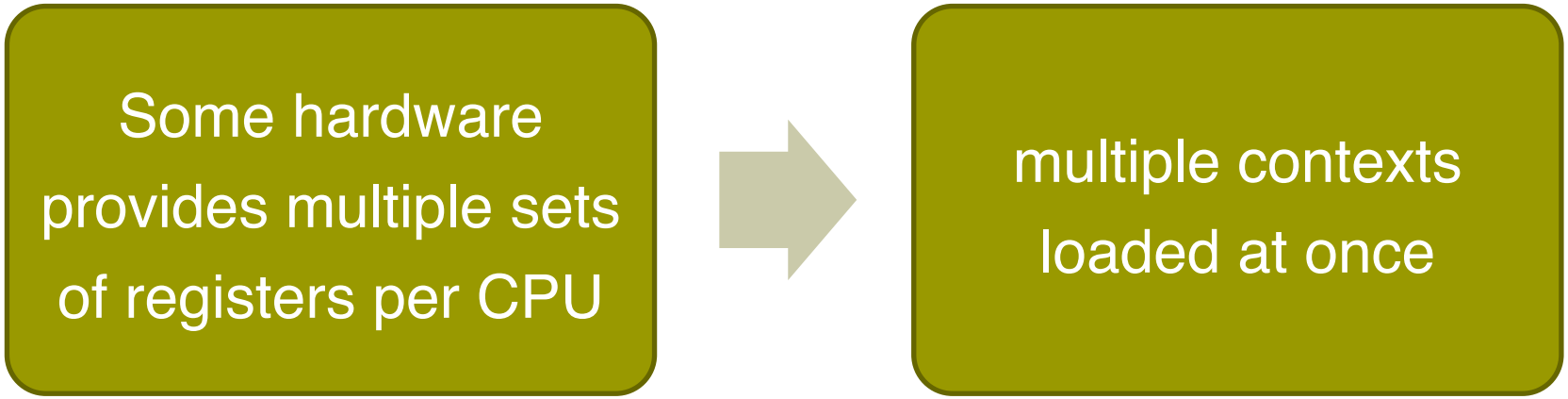
Context Switch

- The system must *save the state* of the old process and load the *saved state* for the new process via a ***context switch***.
- ***Context*** of a process represented in the ***PCB***.
- Context-switch time is ***pure overhead***
  - The system does no useful work while switching.

The more complex the OS and the PCB → the longer the context switch

Context Switch (cont.)

- **Time dependent on hardware support**



Operations on Processes

- System must provide mechanisms for:
  - Process creation
  - Process termination

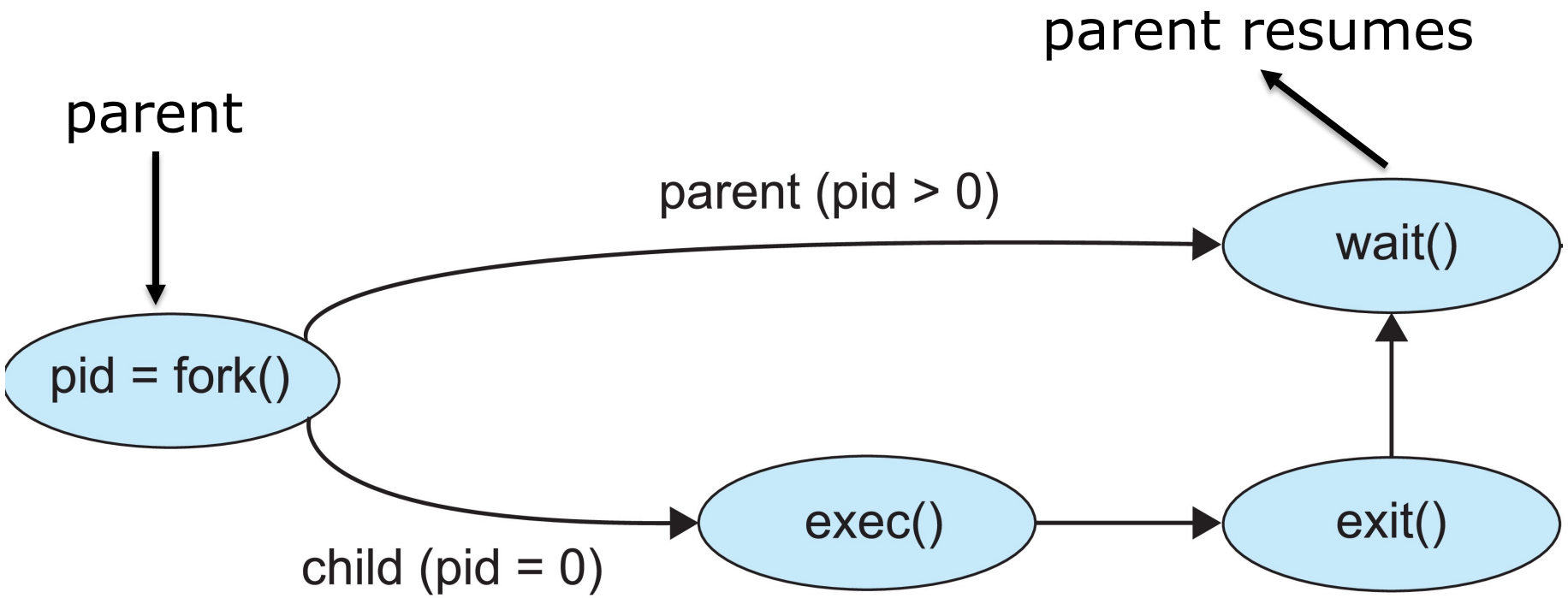
Process Creation

- **Parent** process create **children** processes, which, in turn create other processes, forming a tree of processes.
- Process identified and managed via a process identifier (pid).
- **Resource sharing options**
  - Parent and children share all resources
  - Children share subset of parent's resources
  - Parent and child share no resources
- **Execution options**
  - Parent and children execute concurrently
  - Parent waits until children terminate

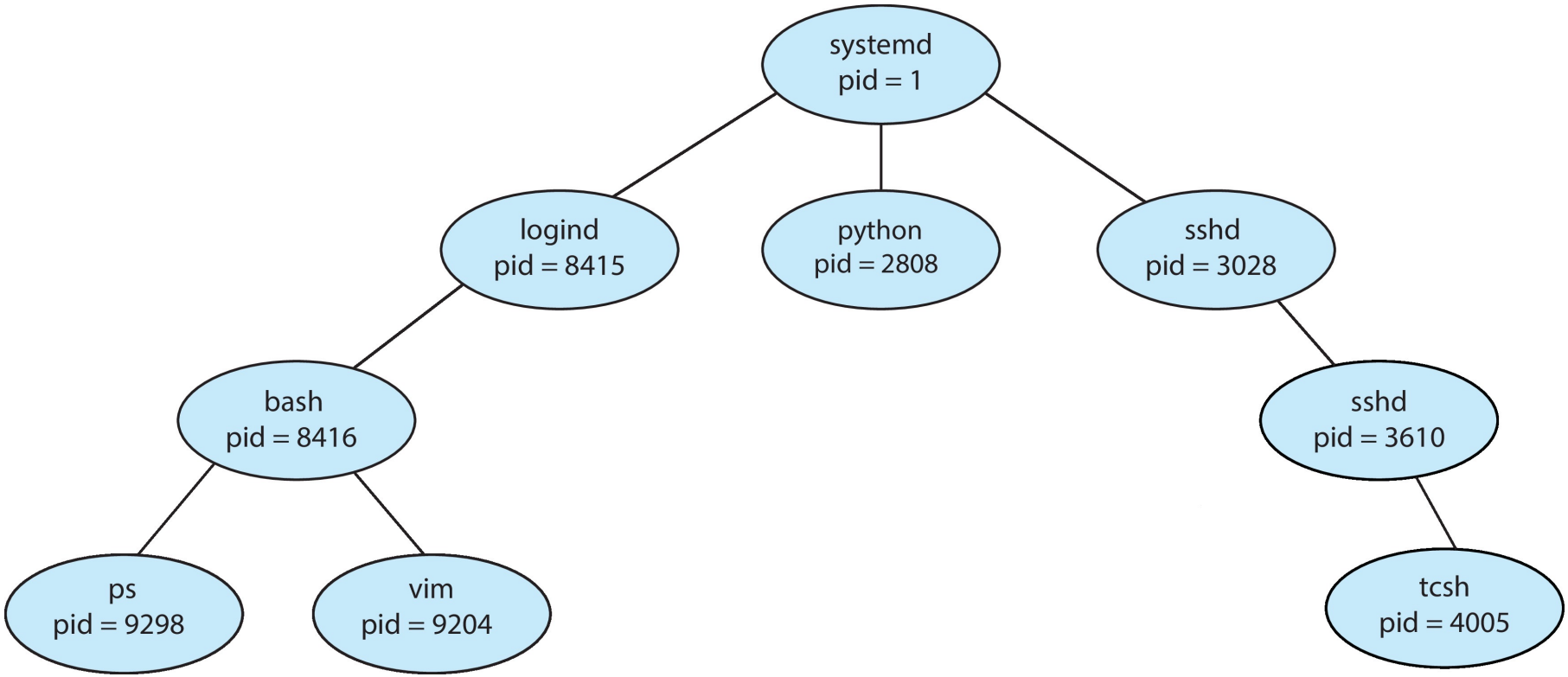
Process Creation (Cont.)

- **Address space**
  - Child duplicate of parent
  - Child has a program loaded into it
- **UNIX examples**
  - **fork()** system call creates new process.
  - **exec()** system call used after a **fork()** to replace the process' memory space with a new program.
  - Parent process calls **wait()** waiting for the child to terminate.

Process Creation (Cont.)



A Tree of Processes in Linux



C Program Forking Separate Process

#include <sys/types.h> <stdio.h> <***unistd***.h>

```
int main()
{
pid_t pid;

    /* fork a child process */
    pid = fork();

    if (pid < 0) { /* error occurred */
       fprintf(stderr, "Fork Failed");
       return 1;
    }
    else if (pid == 0) { /* child process */
       execlp("/bin/ls","ls",NULL);
    }
    else { /* parent process */
       /* parent will wait for the child to complete */
       wait(NULL);
       printf("Child Complete");
    }

    return 0;
}
```

Process Termination

- Process executes last statement and then asks the operating system to delete it using the **`exit()`** system call.
  - Returns status data from child to parent (via **`wait()`**)
  - Process' resources are deallocated by operating system.

- Parent may terminate the execution of children processes using the **`abort()`** system call. Some reasons for doing so:
  - Child has exceeded allocated resources.
  - Task assigned to child is no longer required.
  - The parent is exiting, and the operating systems does not allow a child to continue if its parent terminates.

Process Termination (Cont.)

- Some OSs do not allow child to ***exists*** if its parent has terminated.
  - If a process terminates, then all its children must also be terminated.
  - **Cascading termination:** All children, grandchildren, etc., are terminated.
  - The termination is initiated by the operating system.

Process Termination (Cont.)

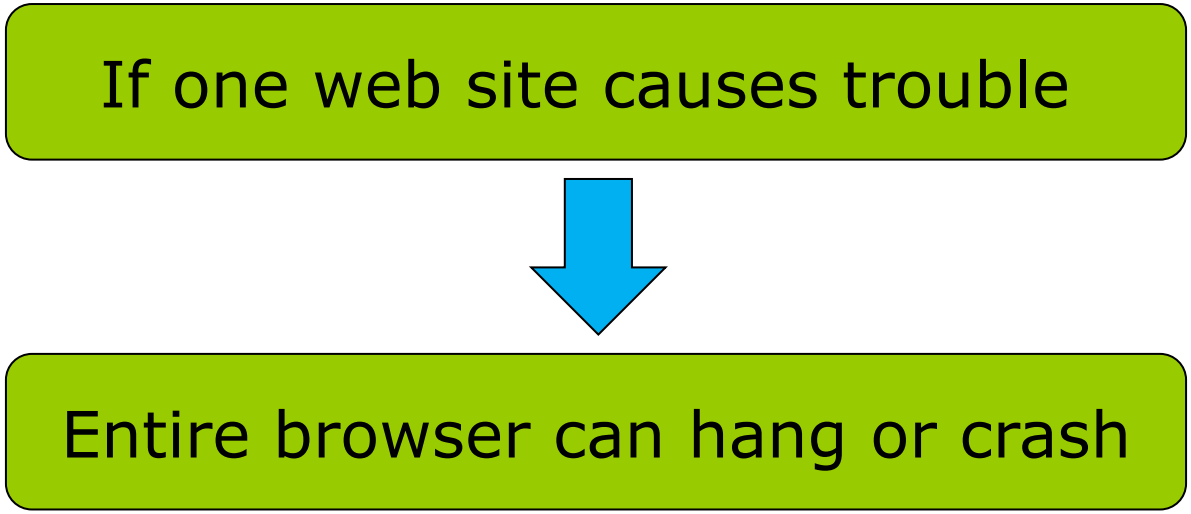
- The parent process may wait for termination of a child process by using the ***wait() system call***.
  - The call returns status information and the pid of the terminated process.

pid = wait(&status);

- If no parent waiting (did not invoke wait()) process is a zombie.
- If parent terminated without invoking wait(), process is an orphan.

Multiprocess Architecture – Browser (Cont.)

- Many web browsers ran as single process (some still do)



Multiprocess Architecture – Chrome Browser (Cont.)

- Google Chrome is multiprocess with 3 different types of processes:
  - Browser process manages user interface, disk and network I/O.
  - Renderer process renders web pages, deals with HTML, Javascript.
    - A new renderer created for each website opened
    - Runs in sandbox restricting disk and network I/O, minimizing effect of security exploits.
  - Plug-in process for each type of plug-in.