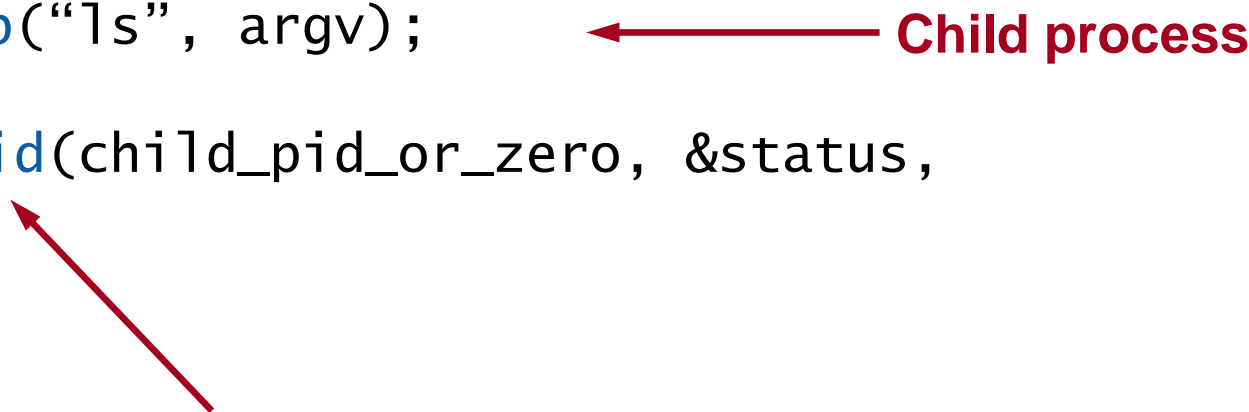


# Linux Fork/Exec Example

```
int child_pid_or_zero = fork();  
if (child_pid_or_zero == 0) {  
    execvp("ls", argv);  
} else {  
    waitpid(child_pid_or_zero, &status,  
options);  
};
```

**Child process**

**Parent process**



# Windows Process Creation

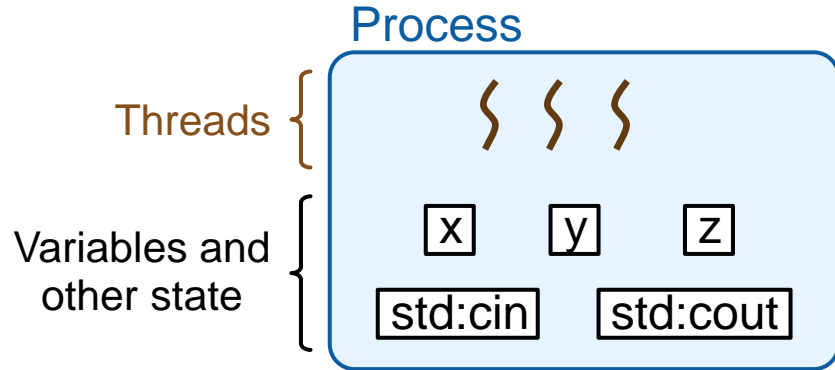
```
BOOL CreateProcess(  
    LPCTSTR lpApplicationName,  
    LPTSTR lpCommandLine,  
    LPSECURITY_ATTRIBUTES lpProcessAttributes,  
    LPSECURITY_ATTRIBUTES lpThreadAttributes,  
    BOOL bInheritHandles,  
    DWORD dwCreationFlags,  
    LPVOID lpEnvironment,  
    LPCTSTR lpCurrentDirectory,  
    LPSTARTUPINFO lpStartupInfo,  
    LPPROCESS_INFORMATION lpProcessInformation  
);  
  
WaitForSingleObject(lpProcessInformation->hProcess, INFINITE);
```

# Thread Creation

```
#include <thread>
...
std::thread t(func);
...
t.join();
```

```
void func()
{
    /* This code will run
       concurrently with the
       code to the left. */
    ...
}
```

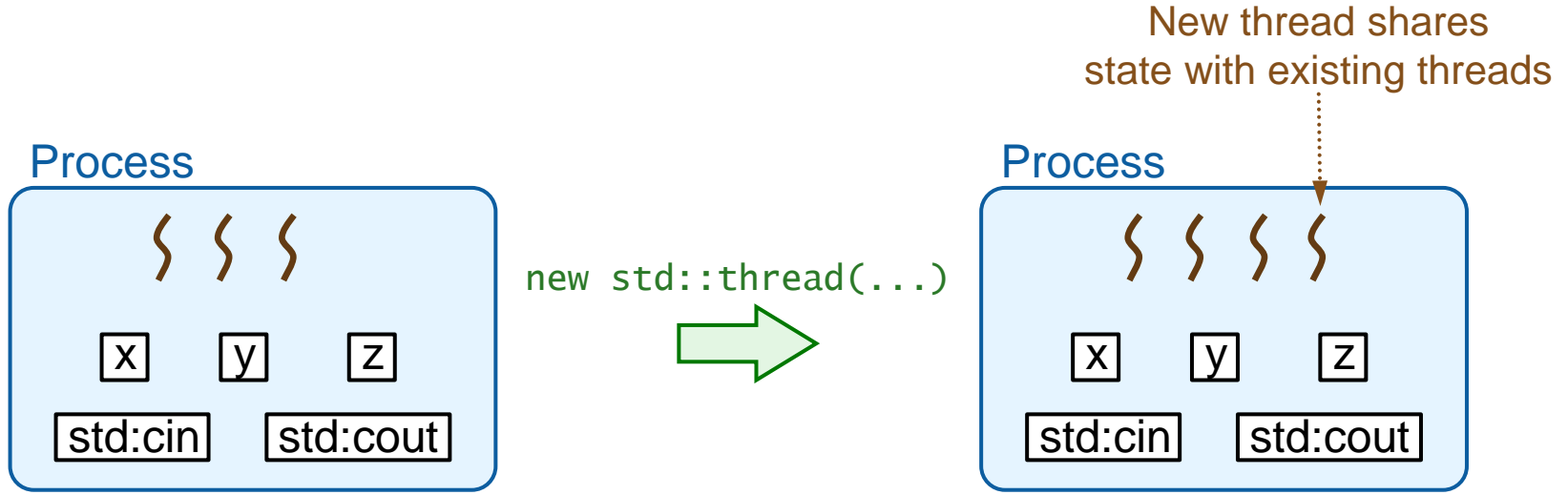
# Processes vs. Threads



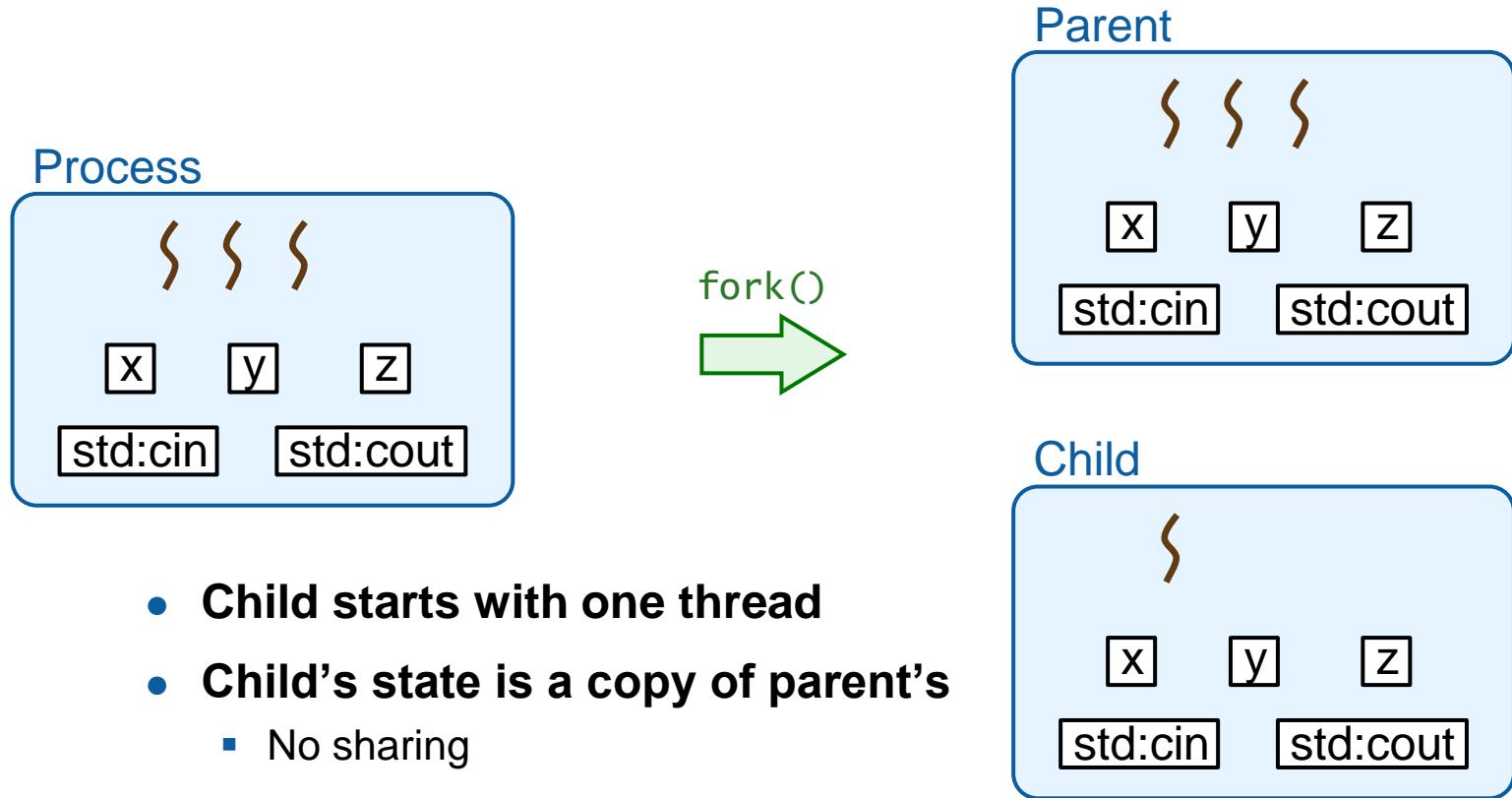
## A thread is part of a process:

- A process can contain many threads
- A process also contains state

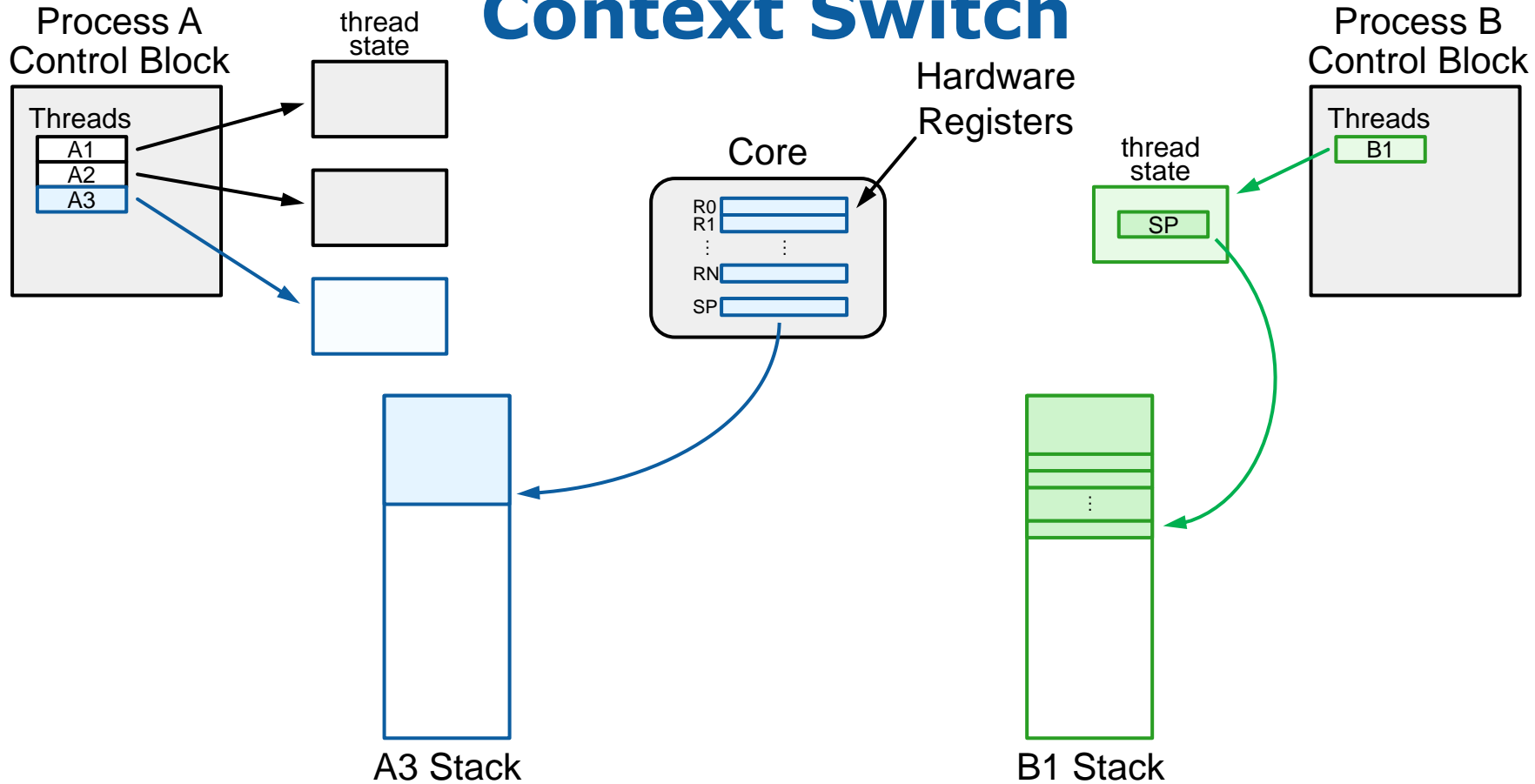
# Create New Thread



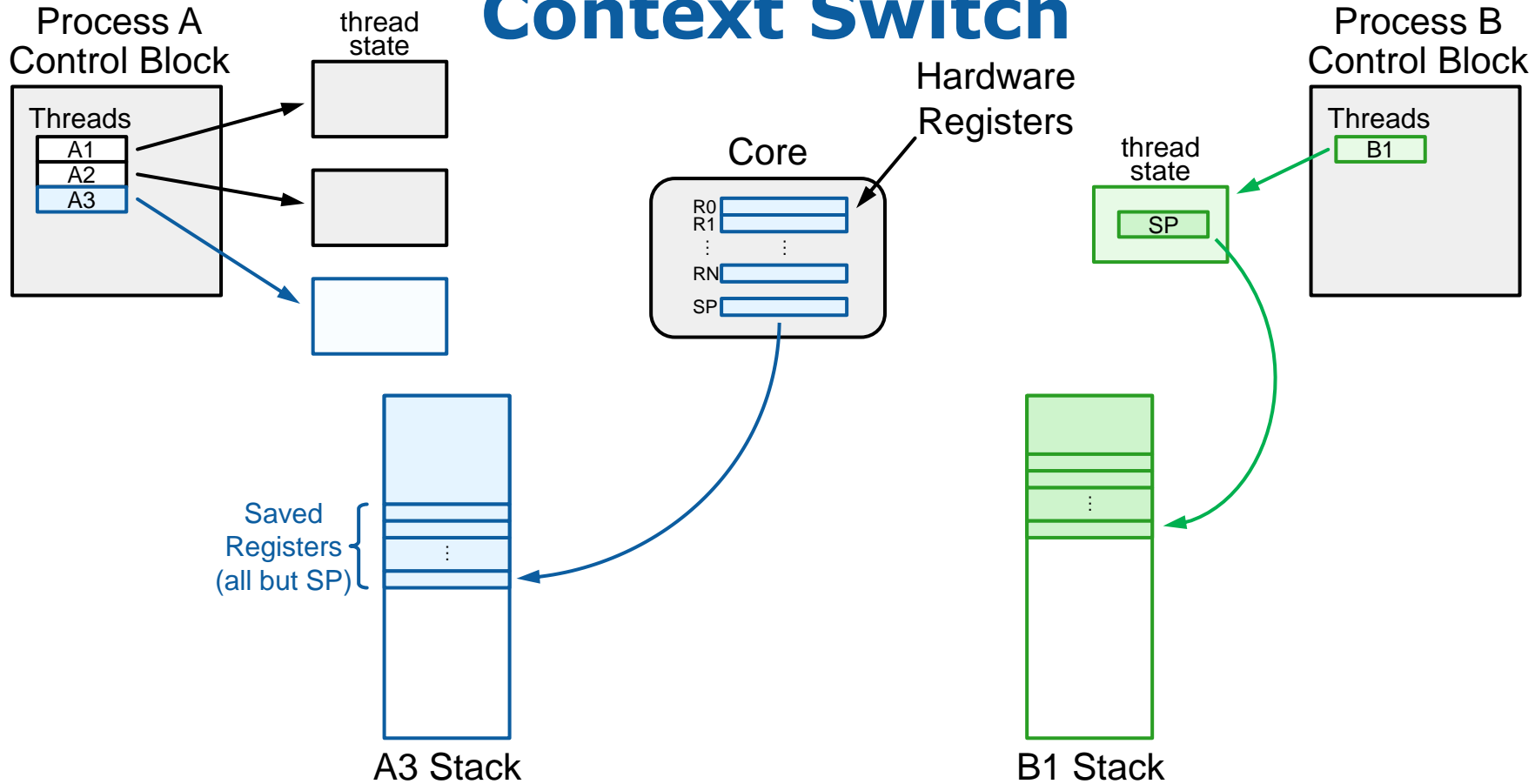
# Create New Thread



# Context Switch

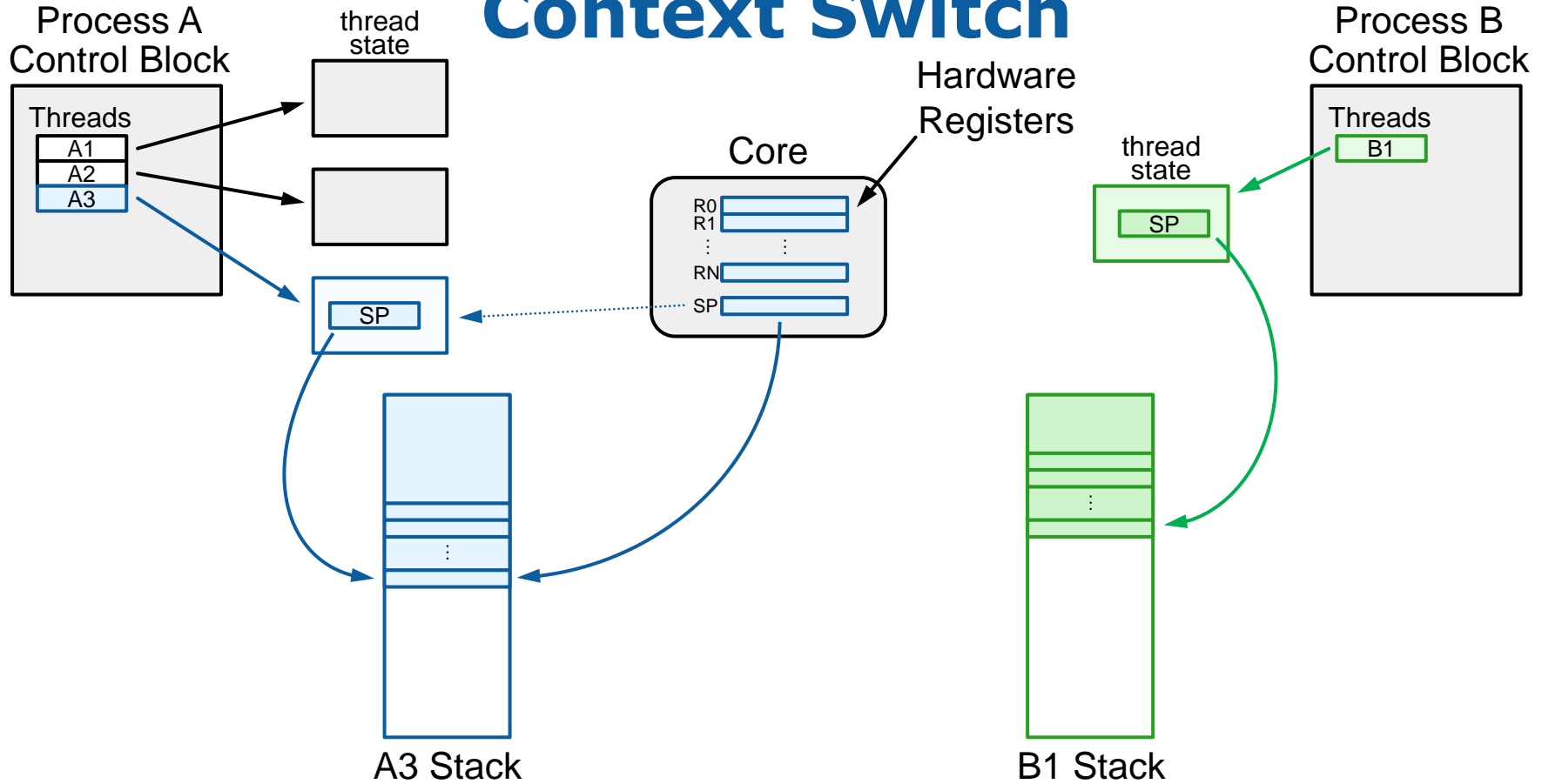


# Context Switch

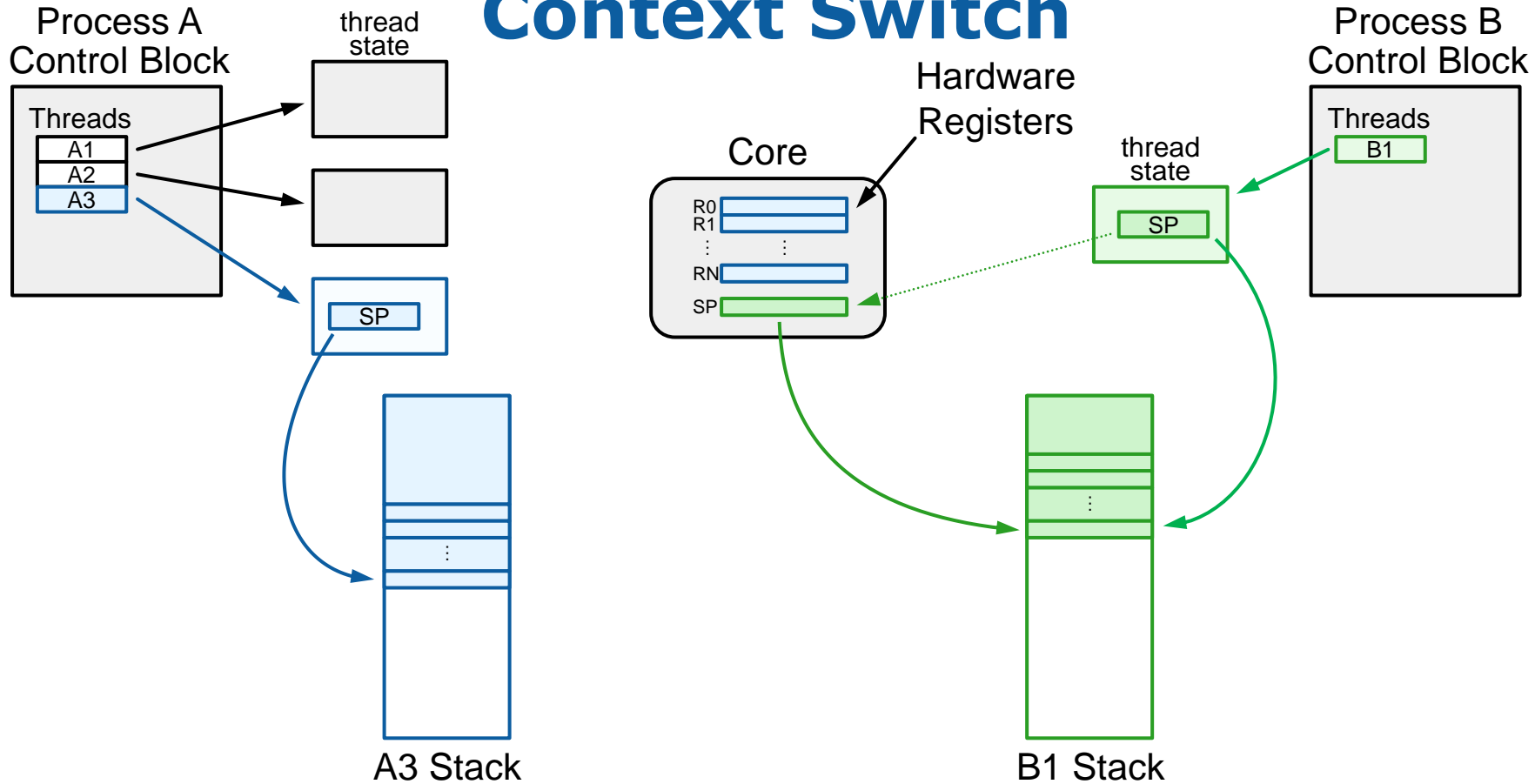




# Context Switch



# Context Switch



# Context Switch

