Implementing System Calls in Haiku OS – `fork()` system call

Overview:

The `fork()` system call is used to create a new process by duplicating the calling process.
This is a fundamental mechanism in UNIX-like operating systems, including Haiku OS, which follows POSIX standards.

What Happens When `fork()` is Called?

- **A new process (child) is created.**
- **The child receives a copy of the parent's:**
  - **Memory space**
  - **File descriptors**
  - **Execution context**

Return Values

The `fork()` system call returns different values in the parent and child processes:

- **`0` → Returned to the child process**
- **`> 0` (PID of child) → Returned to the parent process**
- **`-1` → Indicates that fork failed**

Why It Matters:-

- **Essential for multitasking and multiprocessing**
- **Often used alongside `exec()` and `wait()` to manage process trees**

```c
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>

int main() {
    pid_t pid = fork();

    if (pid < 0) {
        perror("fork failed");
        return 1;
    }

    if (pid == 0) {
        printf("This is the child process! PID: %d\n", getpid());
    } else {
        printf("This is the parent process! Child PID: %d\n", pid);
    }

```

Compiling and Running in Haiku:

Save the code as my`fork.cpp`.

Open Haiku Terminal.

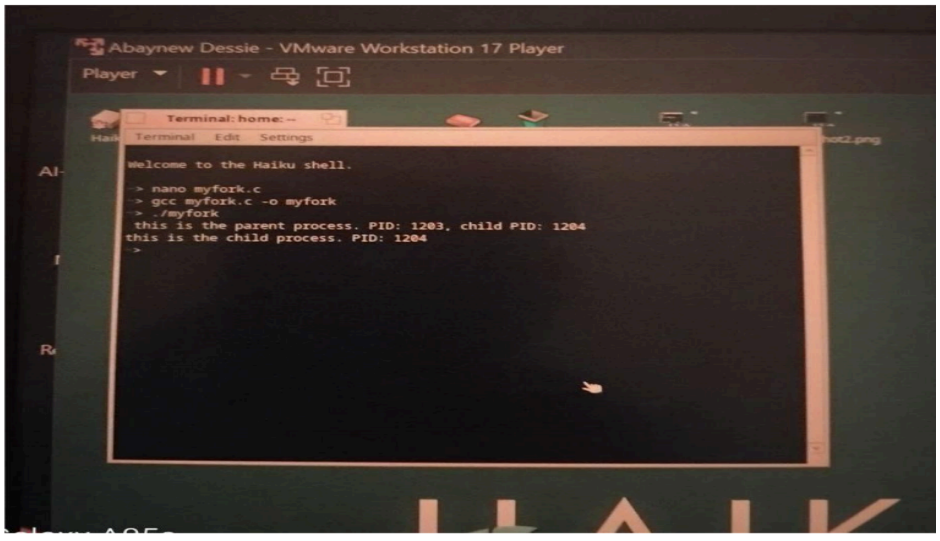
Compile it:

```
g++ myfork.cpp -o myfork
```

Run the program:

```
./myfork
```

Expected Output:



You will see two lines printed:
this is the parent process
This is the child process

Example:
This is the parent process! Child PID: 1203
This is the child process! PID: 1204

Note: Order may vary due to scheduling.

How fork() Works in the OS Kernel:

The Haiku kernel handles fork() by:
o Allocating a new process ID.
o Copying the process memory space (Copy-On-Write optimization may apply).
o Duplicating file descriptors and CPU registers.

The child starts executing exactly where the parent left off—right after the fork() call.

Use Cases:
Launching new programs (fork() + exec()).
Creating daemon/background processes.
Simulating parallel processing for practice.

Summary

The `fork()` system call in Haiku OS offers a practical opportunity to explore how the operating system creates and manages processes. It serves as an entry point for understanding multitasking, inter-process communication, and kernel-level resource management.