Fork_wait.c

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

int main() {
   pid_t pid;
   int status;

   printf("Parent process started. PID: %d\n", getpid());

   pid = fork();  // Create child process

   if (pid < 0) {
      perror("fork failed");
      exit(1);
   }
   else if (pid == 0) {
      // Child process
      printf("Child process started. PID: %d, Parent PID: %d\n", getpid(), getppid());
      sleep(3);  // Simulate work
      printf("Child process exiting.\n");
      exit(0);
   }
   else {
      // Parent process
      printf("Parent waiting for child to finish...\n");
      wait(&status);  // Wait for child to exit
      printf("Child finished with status %d\n", WEXITSTATUS(status));
      printf("Parent process exiting.\n");
   }

   return 0;
}
```

OUTPUT:-

```
pict@mplab-12:~/Desktop/33164$ gcc fork_wait.c -o fork_wait
pict@mplab-12:~/Desktop/33164$ ./fork_wait
Parent process started. PID: 8912
Parent waiting for child to finish...
Child process started. PID: 8913, Parent PID: 8912
Child process exiting.
Child finished with status 0
Parent process exiting.
```