```c
/*
   Esempio di fork. Una sola fork, e sia il figlio che il padre non
   fanno altro che scrivere i due ID. Il padre aspetta il figlio (se
   do_wait è diverso da zero, il che dipende dall'opzione -w e dalla
   variabile do_wait) e quindi, se uno dei figli "ritarda" diventa uno
   'zombie' e viene 'adottato' dal processo 'init', per cui il parent
   ID che stampa non è quello del padre originale. Le due opzioni -c e
   -p permettono di scegliere di quanto "ritardare" l'esecuzione della
   printf nei figli e nel padre; il default è zero per tutti.
*/

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>             /* fork */
#include "fun.h"

int main(int argc, char *argv[])
{
  int do_wait = 1;

  pid_t pid;
  int i;
  int sleep_child = 0;
  int sleep_parent = 0;

  get_opt(argc, argv,
          &sleep_child, &sleep_parent,
          &do_wait);

  pid = fork();
  if (pid == 0) {

    sleep(sleep_child);
    printf("child:  %d with parent: %d\n",
           getpid(), getppid());
    return;
  } else {

    int status;
    sleep(sleep_parent);
    if (do_wait) waitpid(pid, &status, 0);
    printf("parent: %d with child:  %d\n",
           getpid(), pid);
  }
  return 0;
}
```