Programación de Multitareas utilizando Hilos

Enero/2012

Ing. Laura Sandoval Montaño, Ing. Manuel Enrique Castañeda Castañeda

PAPIME 104911

Programación de Multitareas utilizando Hilos

- Origen de los hilos como elementos necesarios en la programación de multitareas
- **Multihilos en un solo procesador**
- Multihilos en varios procesadores
- Implementación de hilos en diversas plataformas y lenguajes de programación.

Ing. Laura Sandoval Montaño, Ing. Manuel Enrique Castañeda Castañeda
PAPIME 104911

Programación de Multitareas utilizando Hilos

- Multihilos en un solo procesador.
  - **Hilos POSIX**
  - Hilos en C
  - Hilos en Java

Ing. Laura Sandoval Montaño, Ing. Manuel Enrique Castañeda Castañeda
PAPIME 104911

Multihilos en un solo procesador

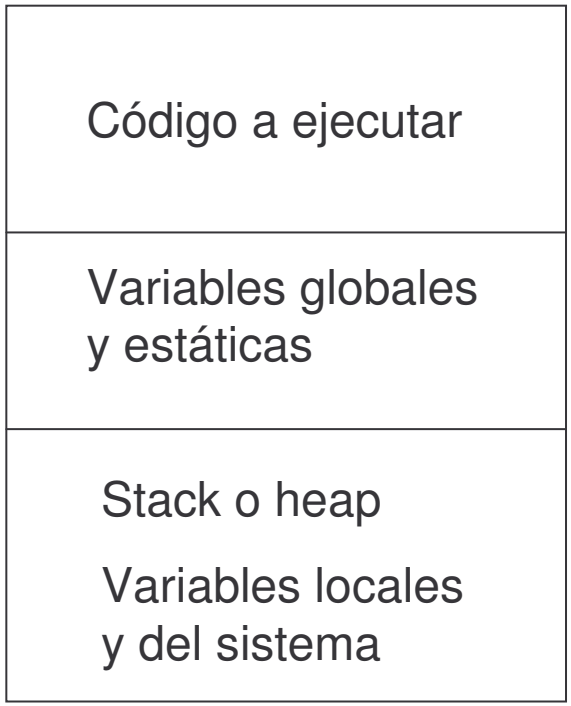
- Hilos POSIX

POSIX significa **P**ortable **O**perating **S**ystem **I**nterconnection (for **U**nix). Es un estándar orientado a facilitar la creación de aplicaciones confiables y portables. La mayoría de las versiones populares de UNIX (Linux, Mac OS X) están cumpliendo este estándar en gran medida. La biblioteca para el manejo de hilos en POSIX es pthread.

La última edición que define la interfaz con C es el Portable Operating System Interface (POSIX) [IEEE, 2001b]. Ver www.UNIX-systems.org

Hilos POSIX

- Estructura de un proceso

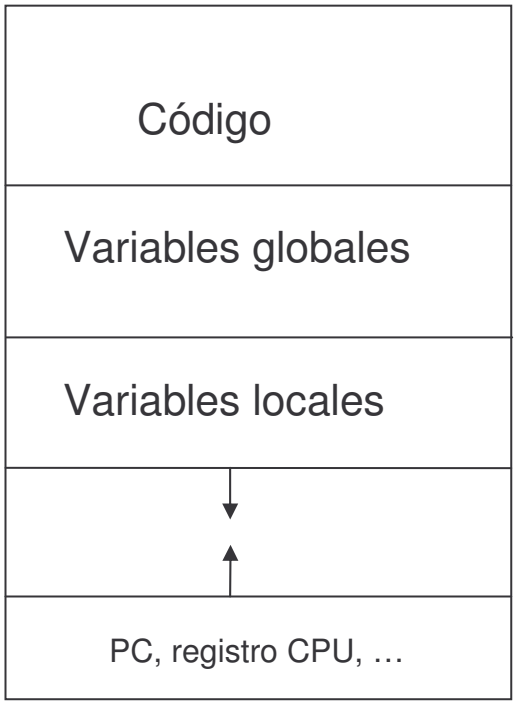


Hilos POSIX

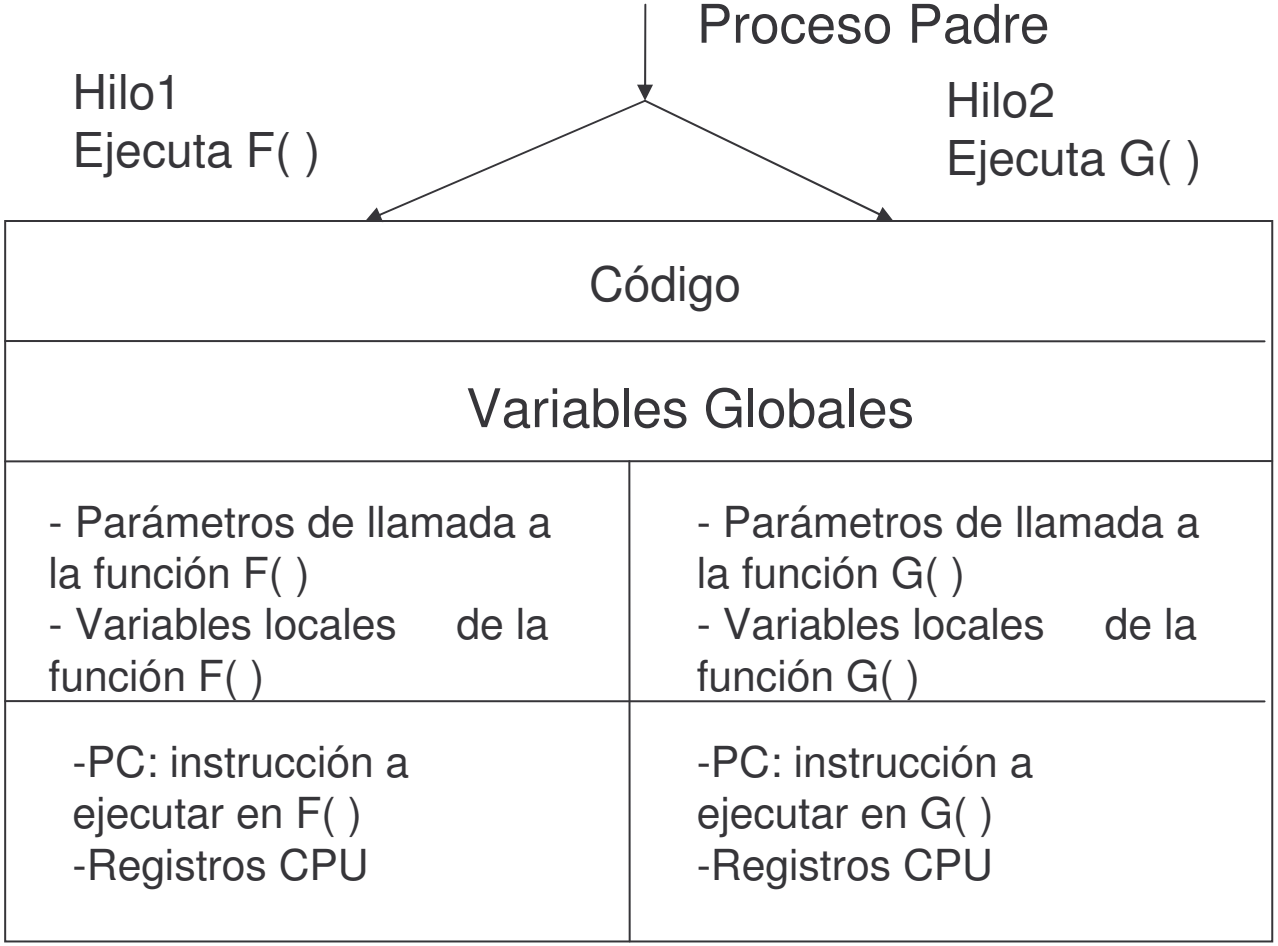
- Permiten la ejecución concurrente de varias secuencias de instrucciones asociadas a funciones dentro de un mismo proceso (hilo principal).

- Los hilos hermanos entre **sí comparten** la misma imagen de memoria, es decir:
  - Código,
  - Variables de memoria globales, y
  - Los dispositivos y ficheros que tuviera abierto el hilo principal.

- Los hilos hermanos **no comparten**:
  - El Contador de Programa: cada hilo podrá ejecutar una sección distinta de código.
  - Los registros de CPU.
  - La pila en la que se crean las variables locales de las funciones a las que se va llamando después de la creación del hilo.
  - El estado: puede haber hilos en ejecución, listos, o bloqueados en espera de un evento.



Hilos POSIX
Uso de memoria



Ing. Laura Sandoval Montaño, Ing. Manuel Enrique Castañeda Castañeda
PAPIME 104911

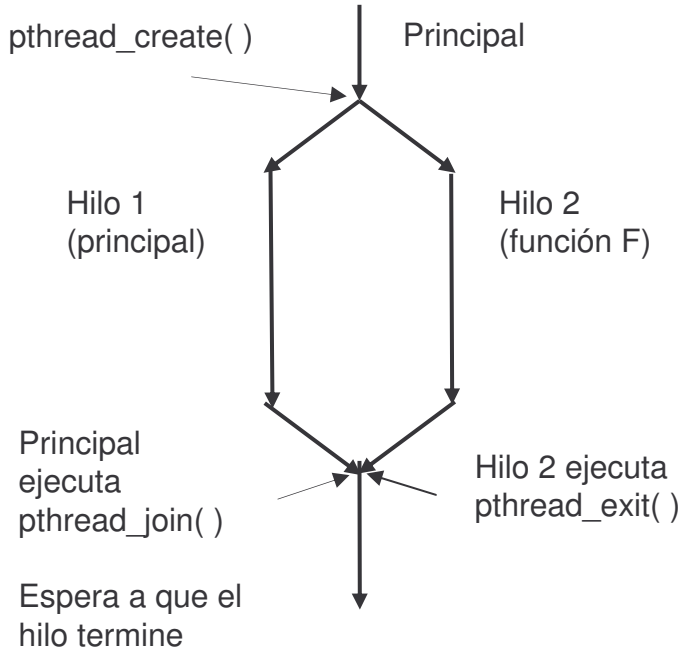
Hilos POSIX
Características

- Consumen menos memoria.
- Cuesta menos crearlos.
- Ya tienen disponible un mecanismo de comunicación entre ellos: las variables globales que son compartidas.
  - Si bien el uso de variables globales está totalmente desaconsejado en otras circunstancias, en concurrencia a través de hilos es el método idóneo para compartir información.
- Cuando alguien se atreve a programar de manera concurrente con hilos, se da por supuesto que sabe programar de forma correcta, y podrá afrontar sin problemas las posibles tareas de depuración.
- Es más sencillo hacer un cambio de contexto entre hilos, lo que los hace ideales para la realización de tareas concurrentes cooperativas.
- La creación de hilos no es una tarea estándar de UNIX
  - Hay que incluir una biblioteca de funciones específica en la línea decompilación: gcc fuente.c -lpthread

Hilos POSIX
Funciones básicas

- Deben incluirse los ficheros de cabecera
  #include <stdio.h>
  #include <pthread.h>
- pthread_create(&id, NULL, función, args): crea un hilo que se asocia a la ejecución de la función que se indica en forma de puntero a función, siempre de tipo void; también se adjunta la lista de argumentos.
- pthread_exit(·): final de la ejecución de un hilo. Será la última sentencia ejecutada por el hilo.
- pthread_kill(·):detiene la ejecución del hilo especificado
- pthread_join(·): espera la finalización del hilo especificado.
- pthread_self(·): permite que un hilo se identifique a sí mismo

Graficamente:



Hilos POSIX
Funciones básicas para hilos del tipo pthread_t

Función POSIX	**Descripción** (referencia)
pthread_equal	verifica igualdad de dos identificados de hilos
pthread_self	retorna ID de propio hilo
pthread_create	crea un hilo
pthread_exit	termina el hilo sin terminar el proceso
pthread_join	espera por el término de un hilo
pthread_cancel	Termina otro hilo
pthread_detach	Configura liberación de recursos cuando termina
pthread_kill	envía una señal a un hilo

Sincronización de hilos en POSIX: Mutex

- Funciones POSIX para Sincronización

  Estas funciones incluyen mecanismos de exclusión mutua (mutex), mecanismos de señalización del cumplimiento de condiciones por parte de variables, y mecanismos de acceso de variables que se modifican en forma exclusiva, pero pueden ser leídas en forma compartida. Las funciones para el manejo de zonas de acceso exclusivo tienen el prefijo pthread_mutex

  Un mutex es una variable especial que puede tener estado tomado (locked) o libre (unlocked). Es como una compuerta que permite el acceso controlado. Si un hilo tiene el mutex entonces se dice que es el dueño del mutex. Si ningún hilo lo tiene se dice que está libre (o unclucked). Cada mutex tiene una cola de hilos que están esperando para tomar el mutex. El uso de mutex es eficiente, pero debería ser usado sólo cuando su acceso es solicitado por corto tiempo.

Hilos POSIX
Funciones básicas para hilos del tipo pthread_mutex_t

Función POSIX	Descripción
pthread_mutex_destroy	Destruye la variable (de tipo pthread_mutex_t) usada para manejo de exclusión mutua, o candado mutex
pthread_mutex_init	permite dar las condiciones iniciales a un candado mutex
pthread_mutex_lock	Permite solicitar acceso a la región crítica, el hilo se bloquea hasta su obtención
pthread_mutex_trylock	permite solicitar acceso a la región crítica, el hilo retorna inmediatamente. El valor retornado indica si otro hilo lo tiene.
pthread_mutex_unlock	Permite liberar un mutex.

Referencias

Bibliografía:

David R. Butenhof
Programming with POSIX Threads
Ed. Addison Wesley, 1997

Francisco M. Márquez
UNIX. Programación avanzada
Ed. Alfaomega Ra-Ma

Sitios WEB

Manual de referencial de manejo de hilos POSIX (incluye en Windows32)
http://sourceware.org/pthreads-win32/manual/index.html
Curso de manejo de Hilos en español
http://profesores.elo.utfsm.cl/~agv/elo330/2s07/lectures/POSIX_Threads.html
http://profesores.elo.utfsm.cl/~agv/elo330/2s06/lectures/POSIX_threads/POSIX_Threads_Synchronization.html