PID_Manager

AUTHOR: Dr. Christer Karlsson, Aaron Alphonsus

Version: 1.0
DATE: 27 February 2017

File Index

File List

Here is a list of all files with brief descriptions:

File Documentation

pid.c File Reference

A solution to the pid manager problem. Contains function definitions to create and initialize the pid array, allocate a pid, and release a pid.

```
#include <errno.h>
#include <pthread.h>
#include <stdio.h>
#include "pid.h"
```

Functions

- int **allocate_map** (void)
- int **allocate_pid** (void)
- void **release_pid** (int pid)

Detailed Description

A solution to the pid manager problem. Contains function definitions to create and initialize the pid array, allocate a pid, and release a pid.

Authors:

Dr. Christer Karlsson, Aaron Alphonsus

Date:

27 February 2017

Function Documentation

int allocate_map (void)

Initializes pid map and checks if successful

Returns:

-1 if unsuccessful, 0 if successful

Mark all pids unused

Set last to smallest pid

Check if initializaiton succeeds

int allocate_pid (void)

Finds the next available pid and marks it as in use if one is found

Returns:

Allocated pid if successful, -1 if none available

Acquire the mutex lock and warn if unable

Set found to 0

Find the next available pid. Iterate from last to PID_MAX

If pid available, set last to pid and indicate found

Iterate from PID_MIN to last

If pid available, set last to pid and indicate found

Release and warn if the mutex was not released

Returns pid if available, -1 if none available

void release_pid (int *pid*)

Releases a pid making sure that the process is synchronized.

Parameters:

in	*pid*	The pid to set to unused

Acquire the mutex lock and warn if unable

Release and warn if the mutex was not released */

pid.h File Reference

Header file for pid manager. Contains declaration of array keeping track of pids in use, last pid in use, and mutex lock for synchronization.
`#include <pthread.h>`

Macros

- #define **PID_MIN**   300
  Range of possible pids.
- #define **PID_MAX**   500

Variables

- pthread_mutex_t **mutex**
  Mutex lock for accessing pid_map.
- int **pid_map** [**PID_MAX**+1]
  Array representing pids in use.
- int **last**

Detailed Description

Header file for pid manager. Contains declaration of array keeping track of pids in use, last pid in use, and mutex lock for synchronization.

Authors:

Dr. Christer Karlsson, Aaron Alphonsus

Date:

27 February 2017

Macro Definition Documentation

#define PID_MAX   500

#define PID_MIN   300

Range of possible pids.

Variable Documentation

int last

pthread_mutex_t mutex

Mutex lock for accessing pid_map.

int pid_map[PID_MAX+1]

Array representing pids in use.

test.c File Reference

Tests the implementation of the PID manager by creating 100 threads, and having each thread request a pid, sleep for a random period of time and releast the pid.

```
#include <errno.h>
#include <pthread.h>
#include <unistd.h>
#include <stdio.h>
#include <time.h>
#include "pid.h"
```

Macros

- #define **NUM_THREADS** 100
  Define constants.
- #define **ITERATIONS** 10
- #define **SLEEP** 5

Functions

- void * **allocator** (void *param)
- int **main** (void)

Variables

- int **in_use** [**PID_MAX**+1]
  Declare in_use array.
- pthread_mutex_t **test_mutex**

Detailed Description

Tests the implementation of the PID manager by creating 100 threads, and having each thread request a pid, sleep for a random period of time and release the pid.

Authors:

Dr. Christer Karlsson, Aaron Alphonsus

Date:

27 February 2017

Macro Definition Documentation

#define ITERATIONS 10

#define NUM_THREADS 100

Define constants.

#define SLEEP   5

Function Documentation

void* allocator (void * *param*)

This function defines the test strategy. Each thread requests a pid to be allocated, sleeps for a random period of time and then releases it. The pid allocated and released is printed. If no pid is available, a message is displayed.

Parameters:

in	*param*	Void pointer

Declare local variables

Iterate ITERATIONS times

Sleep for a random period of time

Allocate a pid

If pid = -1, no pid available

If pid allocation is sucessful: Indicate in the in_use map the pid is in use

Sleep for a random period of time

Release the pid

int main (void)

Main function. Initializes data structures, creates the threads and has them execute **allocator()**. Concludes by joining the threads.

Returns:

0 Indicates normal termination of main.

Declare variables

Initialize in_use array

Allocate the pid map

Seed random generator

Create the threads

Join the threads

Test is finished

Variable Documentation

int in_use[PID_MAX+1]

Keeps track of pids in use

pthread_mutex_t test_mutex

mutex lock used when accessing data structure to ensure there are no duplicate pids in use.