POSIX: What is it?

POSIX stands for Portable Operating System Interface, and it represents a set of standards implemented by Unix systems primarily

PThreads: What are they?

PThreads are an extension for a threads library introduced by POSIX

What uses them?

It’s mainly UNIX system that uses pThreads although some windows, such as Windowss XP can use some POSIX programs.

Aspects?

How they have evolved,

The history of pthreads:

To take the name, “Threads” as an analogy itself and run with it, when it comes it threads (The thread of string for example), and the processes (Being the needles in this case), why was it that strings were only getting used in conjunction with one needle, only when one was finished sewing one piece of string could the move onto the next, when the capability was there to split the string into two different ones, and run it through two different needles, which would cut the time taken in half. Well this is the thought behind multithreading, having more than one thread running, completing processes, and doing so in less time.

The concept of multiple threads, especially in UNIX machines had begun to become popular in the late 1980’s. Much like in the analogy, the main idea behind it being to take a limited number of a process’s resources and making multiple instances of them, allowing concurrency within a single process, each instances being known as a thread.

An API for the multi-thread model was put together and proposed for inclusion in the POSIX standard, and after being accepted, these POSIX threads began to be better known as pthreads.

Although they may seem very similar, The POSIX thread model and the one used in Linux are slightly different, while the POSIX is that of a single process that contains one or more threads, the linux multi-thread model, is made up of separate tasks than share resources.

Now lets take a look at a program implementing pthreads;

With pthreads, and a lot of the unix machine, it is mainly used in the form of C programs.

#include <stdio.h>

#include <stdlib.h>

#include <pthread.h>

#define NUM\_THREADS 2

/\* create thread argument struct for thr\_func() \*/

typedef struct \_thread\_data\_t {

int tid;

double stuff;

} thread\_data\_t;

/\* thread function \*/

void \*thr\_func(void \*arg) {

thread\_data\_t \*data = (thread\_data\_t \*)arg;

printf("hello from thr\_func, thread id: %d\n", data->tid);

pthread\_exit(NULL);

}

int main(int argc, char \*\*argv) {

pthread\_t thr[NUM\_THREADS];

int i, rc;

/\* create a thread\_data\_t argument array \*/

thread\_data\_t thr\_data[NUM\_THREADS];

/\* create threads \*/

for (i = 0; i < NUM\_THREADS; ++i) {

thr\_data[i].tid = i;

if ((rc = pthread\_create(&thr[i], NULL, thr\_func, &thr\_data[i]))) {

fprintf(stderr, "error: pthread\_create, rc: %d\n", rc);

return EXIT\_FAILURE;

}

}

/\* block until all threads complete \*/

for (i = 0; i < NUM\_THREADS; ++i) {

pthread\_join(thr[i], NULL);

}

return EXIT\_SUCCESS;

}

Now, in order for a C program to utilise pthreads, the header file of pthread.h must be declared at the top of the program. This program will create NUM\_THREADS threads and will print out their user assigned thread id respectively.

Now, in order for a C program to utilise pthreads, the header file of pthread.h must be declared at the top of the program.

To actually create a thread, pthread\_create() is called, and is passed some arguments.

“pthread\_t \*thread”: the actual thread object that contains pthread id

“pthread\_attr\_t \*attr": attributes to apply to this thread

“void \*(\*start\_routine)(void \*)”: the function this thread executes

“void \*arg”: arguments to pass to thread function above