/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 \* Copyright 2012 VMware, Inc.  All rights reserved. VMware Confidential

 \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

|  |
| --- |
| #include <stdio.h> |
|  | #include <stdlib.h> |
|  | #include <pthread.h> |
|  |  |
|  | #define M 3 |
|  | #define K 2 |
|  | #define N 3 |
|  | #define NUM\_THREADS M \* N |
|  |  |
|  | /\* Global variables for threads to share \*/ |
|  | int A[M][K] = {{1, 4}, {2, 5}, {3, 6}}; |
|  | int B[K][N] = {{8, 7, 6}, {5, 4, 3}}; |
|  | int C[M][N]; |
|  |  |
|  | /\* Structure for passing data to threads \*/ |
|  | struct v |
|  | { |
|  | int i; /\* row \*/ |
|  | int j; /\* column \*/ |
|  | }; |
|  |  |
|  | void \*runner(void \*ptr); /\* the thread \*/ |
|  |  |
|  | int main(int argc, char \*\*argv) |
|  | { |
|  | int i, j; |
|  | int thread\_counter = 0; |
|  |  |
|  | pthread\_t workers[NUM\_THREADS]; |
|  |  |
|  | /\* We have to create M \* N worker threads \*/ |
|  | for (i = 0; i < M; i++) |
|  | { |
|  | for (j = 0; j < N; j++) |
|  | { |
|  | struct v \*data = (struct v \*) malloc(sizeof(struct v)); |
|  | data->i = i; |
|  | data->j = j; |
|  | /\* Now we will create the thread passing it data as a paramater\*/ |
|  | pthread\_create(&workers[thread\_counter], NULL, runner, data); |
|  | pthread\_join(workers[thread\_counter], NULL); |
|  | thread\_counter++; |
|  | } |
|  | } |
|  |  |
|  | /\* Waiting for threads to complete \*/ |
|  | for (i = 0; i < NUM\_THREADS; i++) |
|  | { |
|  | pthread\_join(workers[i], NULL); |
|  | } |
|  |  |
|  | for(i = 0; i < M; i++) |
|  | { |
|  | for(j = 0; j < N; j++) |
|  | { |
|  | printf("%d\t", C[i][j]); |
|  | } |
|  | printf("\n"); |
|  | } |
|  | return 0; |
|  | } |
|  |  |
|  | void \*runner(void \*ptr) |
|  | { |
|  | /\* Casting paramater to struct v pointer \*/ |
|  | struct v \*data = ptr; |
|  | int i, sum = 0; |
|  |  |
|  | for(i = 0; i < K; i++) |
|  | { |
|  | sum += A[data->i][i] \* B[i][data->j]; |
|  | } |
|  |  |
|  | C[data->i][data->j] = sum; |
|  | pthread\_exit(0); |
|  | } |