ΙΟΝΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ

Παράλληλος Προγραμματισμός 2017

Προγραμματιστική Εργασία #1

Καθηγητής : Στεφανιδάκης Μιχαήλ

Φοιτήτρια : Κονιτοπούλου Πανωραία Π2013134

Απαντήσεις…

α) Χωρίς χρήση SSE

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/time.h>
#define N 100
#define R 10

void get_walltime(double *wct) {
        struct timeval tp;
        gettimeofday(&tp,NULL);
        *wct = (double)(tp.tv_sec+tp.tv_usec/1000000.0);
}

int main() {
float *p[R],*newP[R]; //dhlwsh pinakwn
int i,j;
double ts,te,mflops;
float k = 5.0;
float k1 = 0.5;

  // allocate test arrays
for(i=0;i<R;i++){   //desmeush mnhmhs pinakwn
        p[i] = (float *)malloc(N*sizeof(float));
        newP[i] = (float *)malloc(N*sizeof(float));
```

```
}

        if (p==NULL) exit(1);
        if (newP==NULL)      exit(1);

  //initialize all arrays - cache warm-up
        for (i=0;i<R;i++) {
                for(j=0; j<N; j++){
//dinoume tyxaies times ston pinaka kai ton 2o pinaka ton
//gemizoume me 0
                        p[i][j]=(float)rand()/RAND_MAX;
                        newP[i][j] = 0;
                        printf("%f\n", p[i][j]);
                }

        }

  // get starting time (double, seconds)
        get_walltime(&ts);

  // do triad artificial work
//bazoyme mexri -1 wste na mhn bgei ektos oriwn tou
//pinaka
        for (i=1;i < R - 1; i++) {
                for (j=1; j < N - 1; j++) {
```

```
//upologismos toukathe pixel
newP[i][j] =p[i][j] * k + (p[i-1][j] + p[i-1][j-1] + p[i][j-1] + p[i+1][j] + p[i+1][j+1] + p[i][j+1] + p[i-1][j+1] + p[i+1][j-1]) * k1 ;
                }
        }


  // get ending time
        get_walltime(&te);


  // compute mflops/sec (2 operations per R*N passes)
        mflops = (R*N*2.0)/((te-ts)*1e6);


        printf("MFLOPS/sec = %f\n",mflops);


        return 0;
}
```

β) Με χρήση SSE

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/time.h>
#define N 10
#define R 10

void get_walltime(double *wct) {
        struct timeval tp;
        gettimeofday(&tp,NULL);
        *wct = (double)(tp.tv_sec+tp.tv_usec/1000000.0);
}

int main() {
float *p,*newP;
int i,j;
double ts,te,mflops;
float k = 5.0;
float k1 = 0.5;
__m128 *vp, *vnewP

  // allocate test arrays
//desmeush pinakwn
i=posix_memalign((void**)&p,16,P*N*sizeof(float));
        if(i!=0){printf("Wrong");
```

```
                exit(1);}

i=posix_memalign((void**)&newP,16,P*N*sizeof(float));
        if(i!=0){printf("Wrong");
                exit(1);}


  //initialize all arrays - cache warm-up
        for (i=0;i<R;i++) {
                for(j=0; j<N; j++){
//gemisma pinaka me tuxaious arithmous
//kai gemisma neou pinaka me 0
                        p[(i * N) + j]=(float)rand()/RAND_MAX;
                        newP[(i * N) + j]=0;
                        printf("%f\n", p[i][j]);
                }

        }

  // get starting time (double, seconds)
        get_walltime(&ts);
        vp=(__m128 *)p;
        vnewp=(__m128 *)newP;

  // do triad artificial work
```

```
        for (i=1;i < R - 1; i++) {
                for (j=1; j < N - 1; j++) {
        *vnewP=_mm_add_ps(_mm_mul_ps(5.0,*vp),_mm_mul_
ps (0.5,*vp[i-1]));
                vp++; vnewP++;                  }
        }


  // get ending time
        get_walltime(&te);


  // compute mflops/sec (2 operations per R*N passes)
        mflops = (R*N*2.0)/((te-ts)*1e6);


        printf("MFLOPS/sec = %f\n",mflops);


        return 0;
}
```