امیرمحمد پیرحسینلو 9531014
مهدی صفری

به نام خدا

برنامه‌نویسی چندهسته‌ای

دستور کار آزمایشگاه ۷

هدف آزمایش:

در این ازمایش با استفااذه از ابزار پروفایلینگ به اندازه‌گیری میزان Occupancy در یک مثال ساده پرداخته و نشان می‌دهیم افزایش میزان Occupancy لزوما به معنی افزایش سرعت کد نیست. همچنین نحوه استفاده از استریم برای هم‌زمان کردن اجرای کارها روی gpu نشان داده خواهد شد

```
#define MAX_HISTORGRAM_NUMBER 10000
#define ARRAY_SIZE 102400000
#define SCALER 80
#define CHUNK_SIZE 100
```

پارامتر اول برای مشخص کردن بازه اعداد موردبررسی و پارامتر دوم تعداد آنها را مشخص میکند .با استفاده از پارامتر scaler میزان کار هر نخ با انجام محاسبات تکراری افزایش می‌یابد

عدد پیشنهادی برای Scaler در GPU با معماری پاسکال ۸۰ است

ترجیحا Scaler را برای هر قسمت ازمایش ثابت قرار دهید .

قسمت اول: بررسی اثر occupancy بر میزان تسریع

این تابع کرنل و همراه کد host در فایل histogram.cu موجود است. ابتدا کدها را مطالعه کنید و پس از اطمینان از درک درست آزمایش با توجه به اندازه آرایه ورودی و تغییر THREAD_COUNT و CHUNK_SIZE تعداد بلوکهای مورد نیاز برای فراخوانی تابع را بدست می آورید و سپس جدول زیر را کامل کنید.

```
__global__ void histogramKernelSingle(unsigned long long int *c, int *a)
{
    unsigned long long int worker =  blockIdx.x*blockdim.x + threadIdx.x;
    unsigned long long int start = worker * CHUNK_SIZE;
    unsigned long long int end = start + CHUNK_SIZE;
    for (int ex = 0; ex < SCALER; ex++)
        for (long long int i = start; i < end; i++)
        {
            if (i < ARRAY_SIZE)
                atomicAdd(&c[a[i]], 1);
            else
            {

                break;
            }
        }

}
```

اندازه‌ی ارایه ورودی و بازه اعداد داده شده در کد در بالا امده است .

Thread count	8	16	32	256	1024
Scaler	20	20	20	20	20
Number of Blocks	128000	64000	32000	4000	1000
Theoretical Occupancy	50	50	50	100	100
Achieved Occupancy	50	50	50	91.5	91
Kernel Execution Time (s)	2.01	2.02	2.04	2.05	2.04

دلیل کاهش سرعت با افزایش Occupancy چه بوده است؟

عملیات انجام شده در بلاک ها بیشتر کار با حافظه است که درهر صورت باعث می شود به طور مداوم بین بلاک ها سوییچ کنیم تا آن ها کار ورودی خروجی خود را انجام دهند و دوباره منبع در اختیار آن ها قرار دهیم

هر چند تعداد بلاک های در اختیار بیشتر شده اما چون اکثرا در وضعیت کار با ورودی خروجی هستند پس سرعت کاهش یافته است

قسمت دوم : استفاده از استریم

کد قسمت قبل را با تغییرات مناسب جهت استفاده از Stream تغییر دهید و کرنل مربوطه را ۴ بار فراخوانی کنید هربار Scaler را برابر ۲۰ قرار دهید و سپس جدول زیر را کامل کنید.

Thread count	8	16	32	256	1024	۸۸	16	32	256	1024
Scaler	20	20	20	20	20	20	20	20	20	20
Chunk size	100	100	100	100	100	1000	1000	1000	1000	1000
Stream	4	4	4	4	4	4	4	4	4	4
Number of blocks	25600	12800	6400	800	200	2560	1280	640	80	20
Theoretical Occupancy	50	50	50	100	100	50	50	50	100	100
Fully Concurrent Streams	-	-	-	-	+	-	-	-	+	+
Achieved Occupancy	49.9	49.9	49.8	97.1	91.5	50	50	50	92.8	91.7
Kernel Execution Time (s)	2.80	2.69	2.62	2.67	3.04	2.78	2.70	2.72	3.11	3.11

برای مشاهده اجرای همزمان Stream ها باید از Visual profiler استفاده نمایید

پس از انجام قسمت های بالا سوالات زیر را جواب دهید :

۱. به نظر شما Stream ها برای چه نوع مساله‌هایی مناسب تر است؟
۲. آیا با زیاد کردن Stream میزان Occupancy تغییر می کند؟
۳. کاهش یا افزایش تعداد بسته‌های هیستوگرام چه تاثیری روی occupancy دارد

1- مساعلی که که شامل قسمت های مجزا است و داده های مربوط به هر قسمت جداگانه در اختیار قرار دارد همچنین مساعلی که می توان یک تسک را به چندین تسک که هر کدام داده های مجزا دارند شکاند

2- با توجه به نتایج دو قسمت پاسخ منفی است زیرا باید تعداد بلاک های هر استریم را کم کنیم سپس تعداد استریم ها را زیاد کنیم

3- با افزایش آن ها میزان تداخل ها در به روز رسانی یک خانه کمتر می شود و نخ ها کمتر در وضعیت ست کردن و ریست کردن بلاک قرار می گیرند پس میزان اشتغال بیشتر می شود