När/var/hur använder vi vad vi lärt oss?

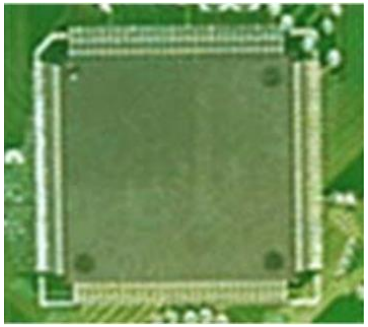
Viktor Kämpe

Maskinorienterad

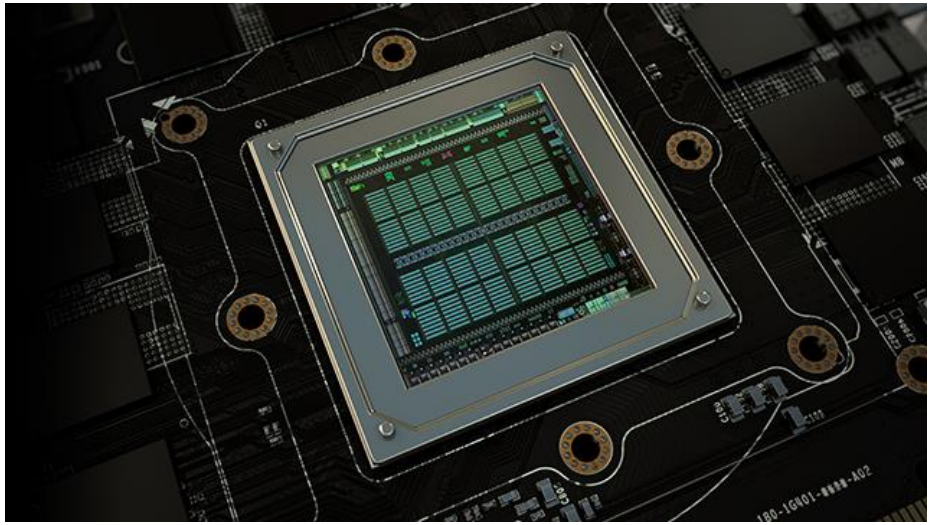
Maskinorienterad betyder nära hårdvaran.

Maskinorienterad?

Räknemaskinen (processorn)



68HCS12



NVIDIA GTX TITAN X

Hårdvaran

Anpassning av program till hårdvara är viktig när man har krav på:

- Latency [t ex mikrosekunder]
- Throughput [t ex flops/sekund]
- Tillförlitlighet [t ex realtidssystem]
- Energikonsumtion [t ex timmar batteritid]

Throughput i datorgrafik

- Generera en ny bild 30-60ggr per sekund
  - (15-30 ms per bild)

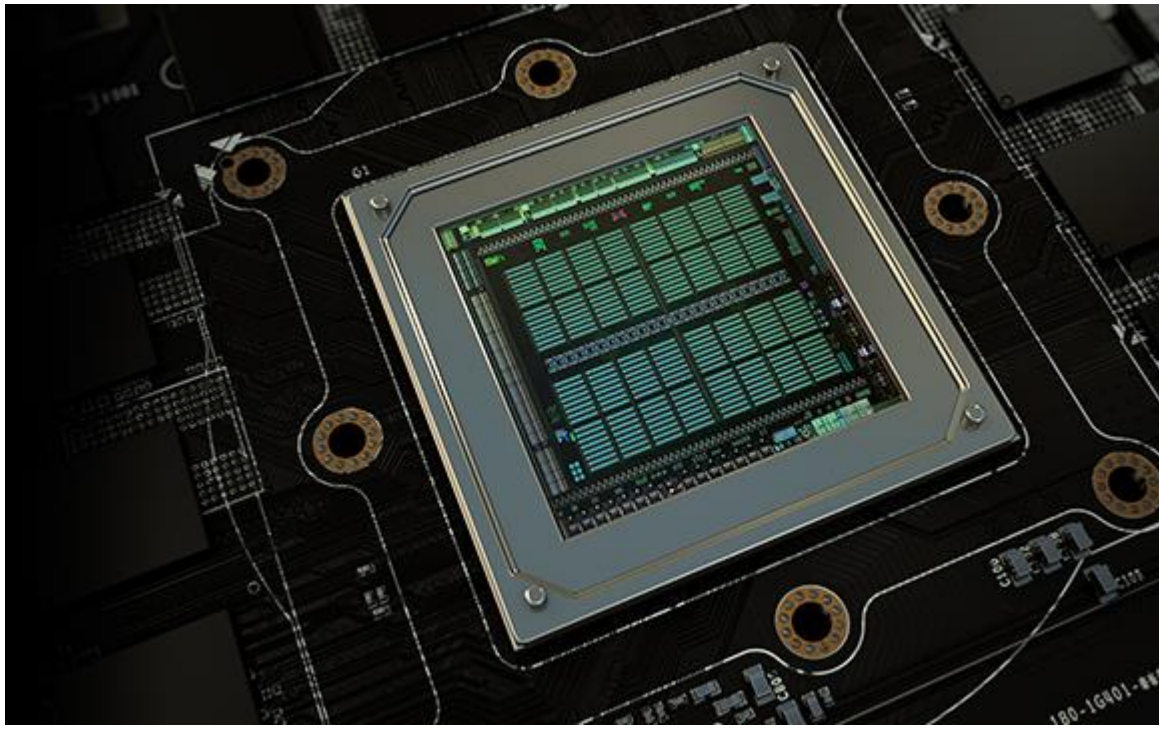
Throughput i datorgrafik

- Generera en ny bild 60ggr per sekund
  - (15ms per bild)
- Varje bild genereras från ca 1 miljon trianglar

Throughput i datorgrafik

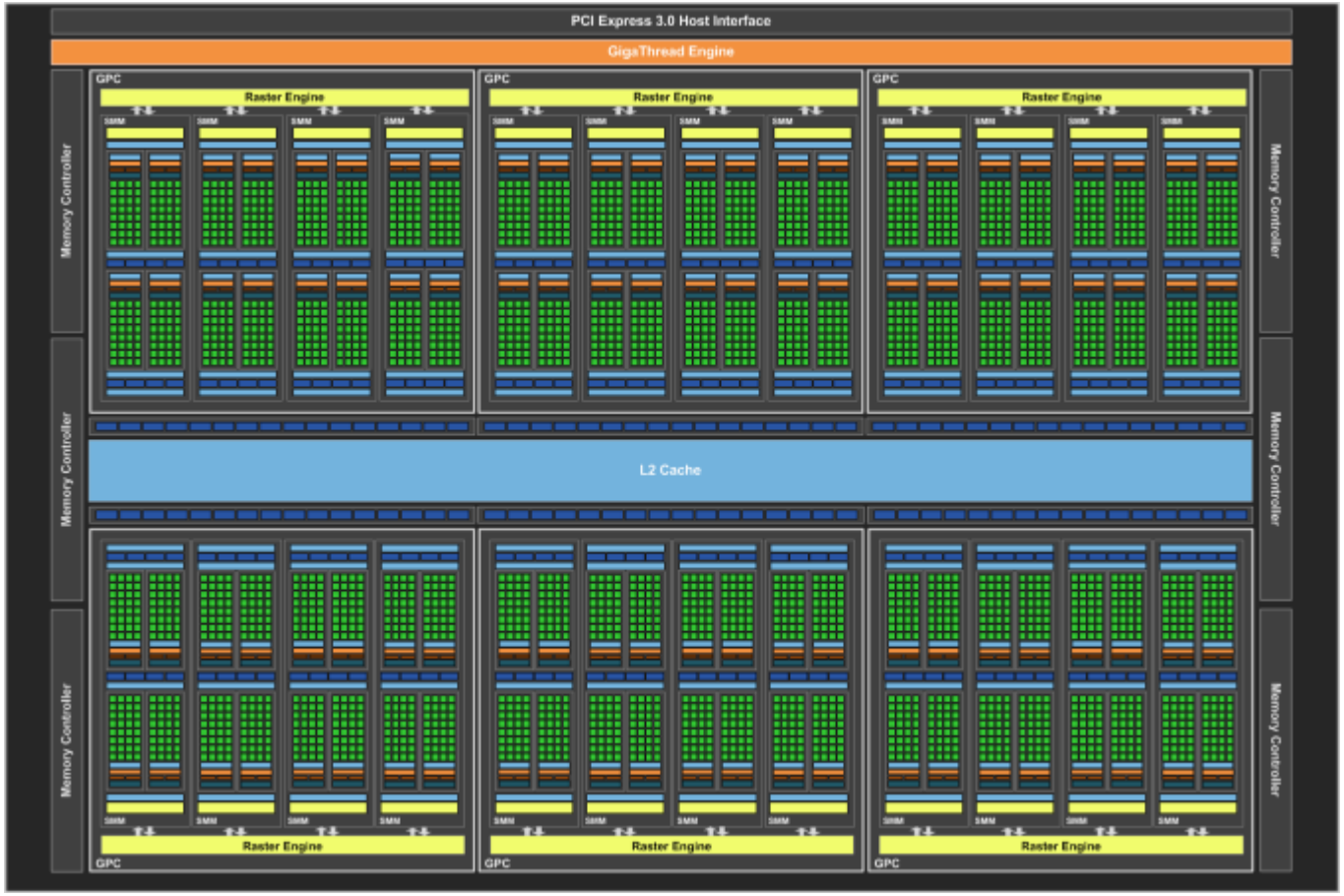
- Generera en ny bild 60ggr per sekund
  - (15ms per bild)
- Varje bild genereras från ca 1 miljon trianglar
- För varje pixel körs ett litet program
  - (för full HD ca 2 miljoner pixlar)

GPU Prestanda



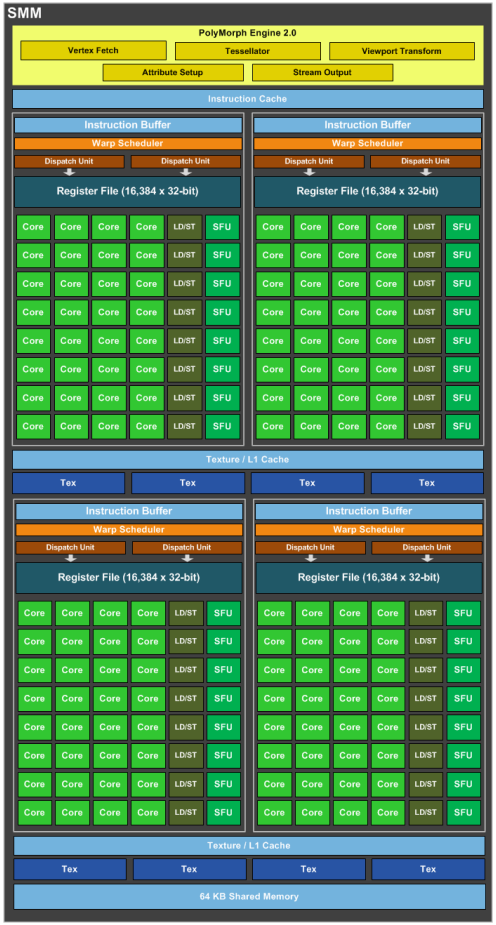
- 8 Miljarder Transistorer
- 24 Multi-processorer
- Ca 1 GHz
- Ca 6 Tera Flops/s
- 1.5 Miljoner register (32bit)
- 336 GByte/s

Parallellism



3072 ”cores” som kör upp till 49152 trådar samtidigt.

En multiprocessor (av 24)



Generell användning av GPU

- Massor av prestanda
- Kan användas för annat än grafik
- Programmeras med C/C++

CUDA

```
__global__ void reductionKernel(const int * __restrict__ data, int iterations, int *
__restrict__ result)
{
    int globalId = blockIdx.x*blockDim.x + threadIdx.x;
    int warpId = globalId /32;
    int laneId = globalId%32;

    int idx = warpId*32*iterations +laneId;

    int threadMax = data[idx];
    for(int i=1; i<iterations; i++) {
        idx +=32;
        threadMax = threadMax > data[idx] ? threadMax : data[idx];
    }

    atomicMax(result, threadMax);
}

int main() {
    //...
    reductionKernel<<< numBlocks, blockSize >>>(device_data, iterations, device_maxValue);
    return 0;
}
```

Assembler

```
.loc 1 10 1
mov.u32 %r10, %ntid.x;
mov.u32 %r11, %ctaid.x;
mov.u32 %r12, %tid.x;
mad.lo.s32 %r13, %r10, %r11, %r12;
.loc 1 11 1
shr.s32 %r14, %r13, 31;
shr.u32 %r15, %r14, 27;
add.s32 %r16, %r13, %r15;
shr.s32 %r1, %r16, 5;
.loc 1 12 1
and.b32  %r17, %r16, -32;
sub.s32 %r2, %r13, %r17;
.loc 1 14 1
shl.b32 %r18, %r9, 5;
mad.lo.s32 %r19, %r18, %r1, %r2;
```

Prestanda

- Minnestransaktioner dyra
- Artimetik billigt

```
Räkna inte Flops/s utan Mbyte/s
```

Energikonsumption

Tills alldeles nyss gjordes högprestanda processorer på ett sätt, och energisnåla processorer på ett annat sätt.

Nu börjar de göras på samma sätt.

```
Prestanda == energieffektivitet
```

Assembler

- Förstår man assembler för en maskin så förstår man hur maskinen fungerar.

C programmering

- När man programmerar för maskinen så kan man använda C.
- C har många bra utvecklingsverktyg.
- Det finns många som förstår C, vilket gör att man snabbt kan komma igång att programmera för nya maskiner.

Summering

- Assembler
  - Förstå maskinen
- C
  - Programmera maskinen
- Maskinorienterad programmering
  - Anpassa program till maskinen