GPUコンピューティング No.6

Unified Virtual Addressing
GPU Direct

東京工業大学　学術国際情報センター

青木　尊之

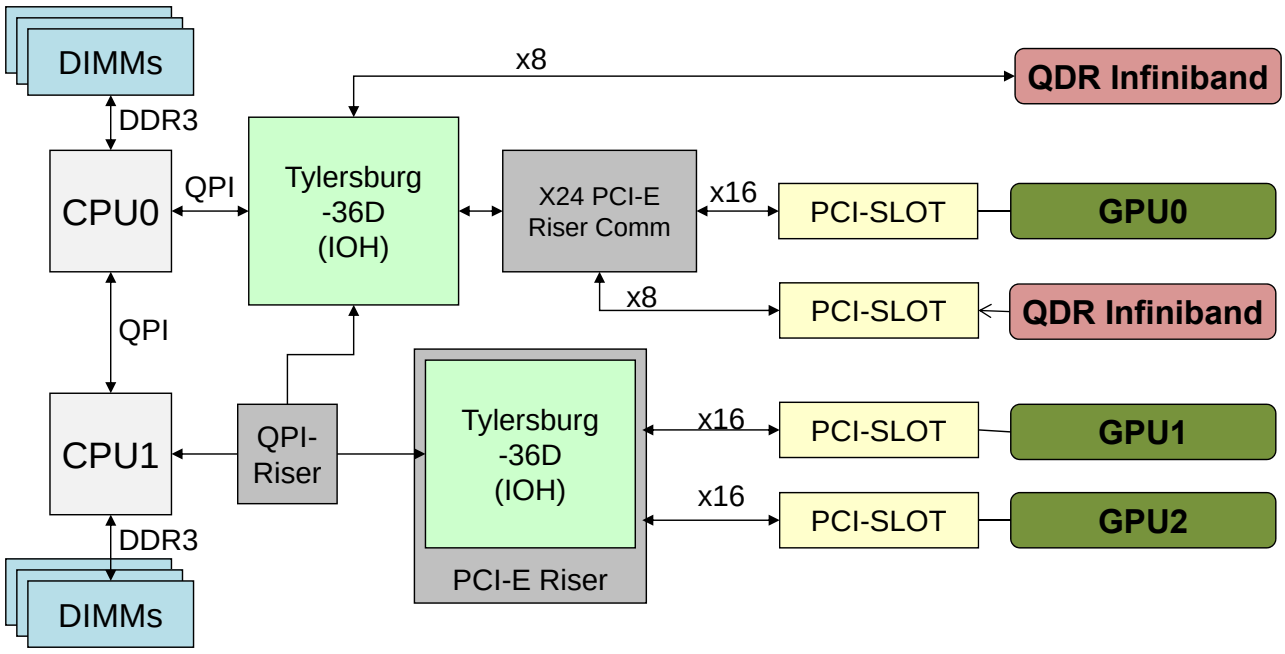
複数GPUの利用

GPU のメモリは Graphics Board 上にあり、物理的には孤立していて、サイズも 6GB（現時点）までであり、計算できる問題が限定される。

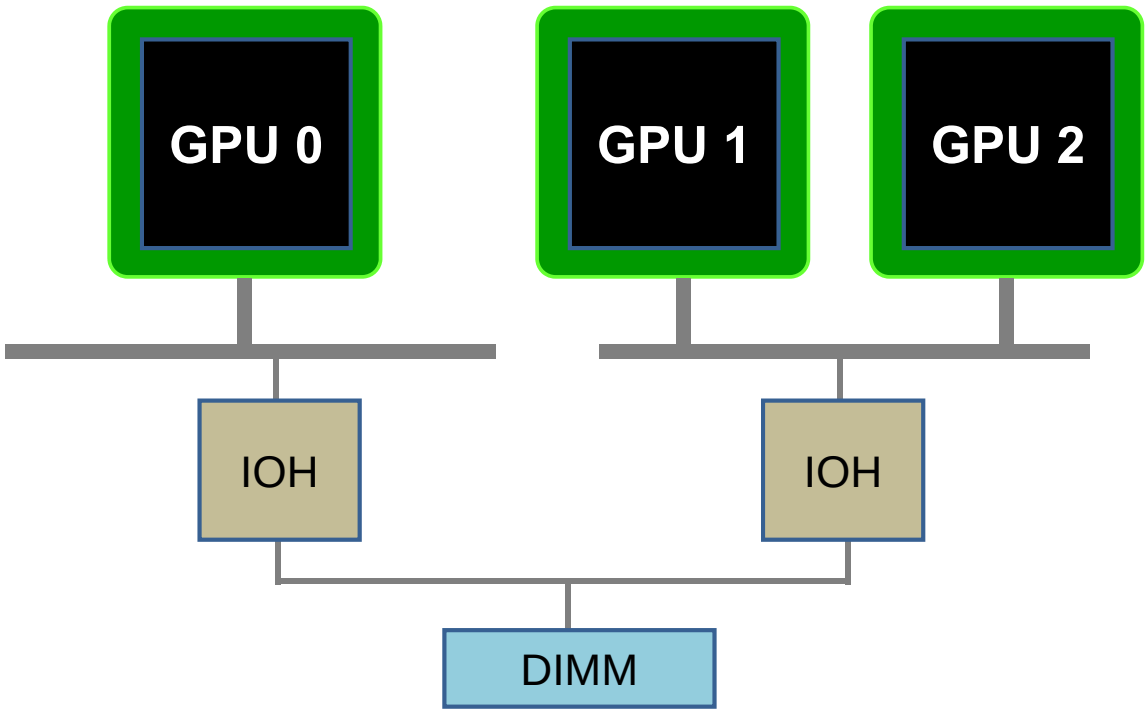
マルチ GPU を使う目的：　1 GPU に入りきらないメモリ使用量の計算を行う。

GPU の演算性能を高める

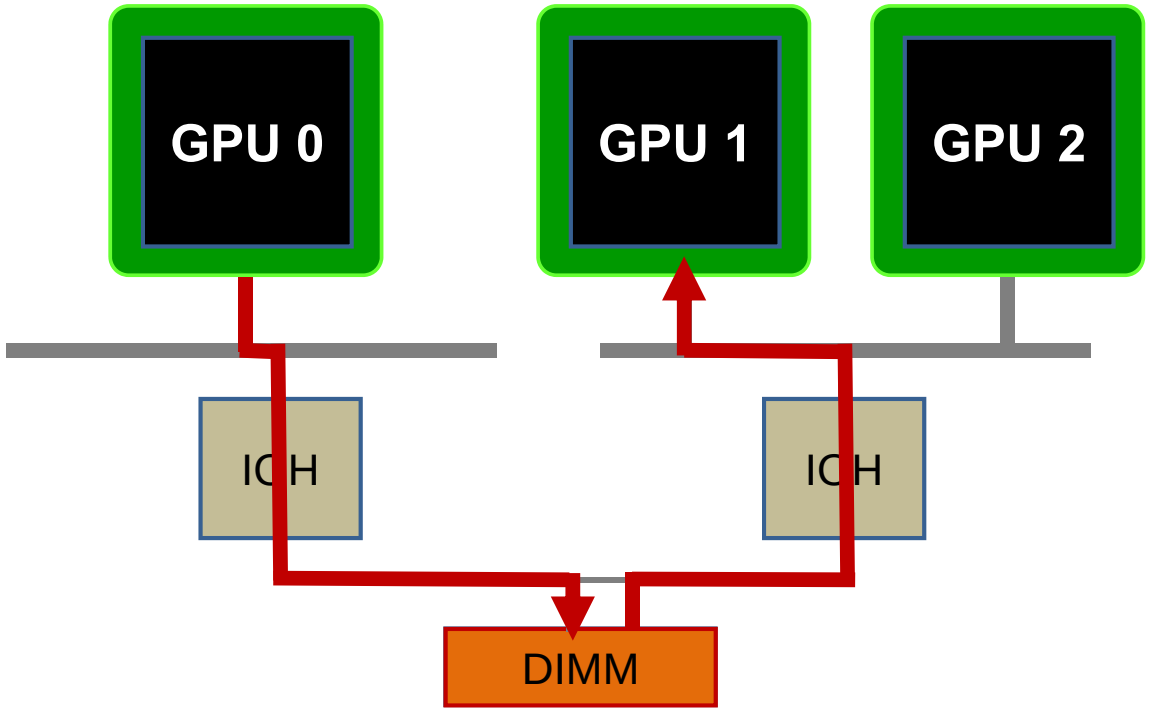
TSUBAME 2.0



TSUBAME 2.0



GPU-to-GPU Data Copy



Device指定とメモリ確保

```
double    *B_GPU0,   *B_GPU1;

cudaSetDevice(0);
cudaMalloc( (void**) &B_GPU0, n*sizeof(double) );
```

Device No.0 のGPUに、n*8 Byte のメモリを確保

```
cudaSetDevice(1);
cudaMalloc( (void**) &B_GPU1, n*sizeof(double) );
```

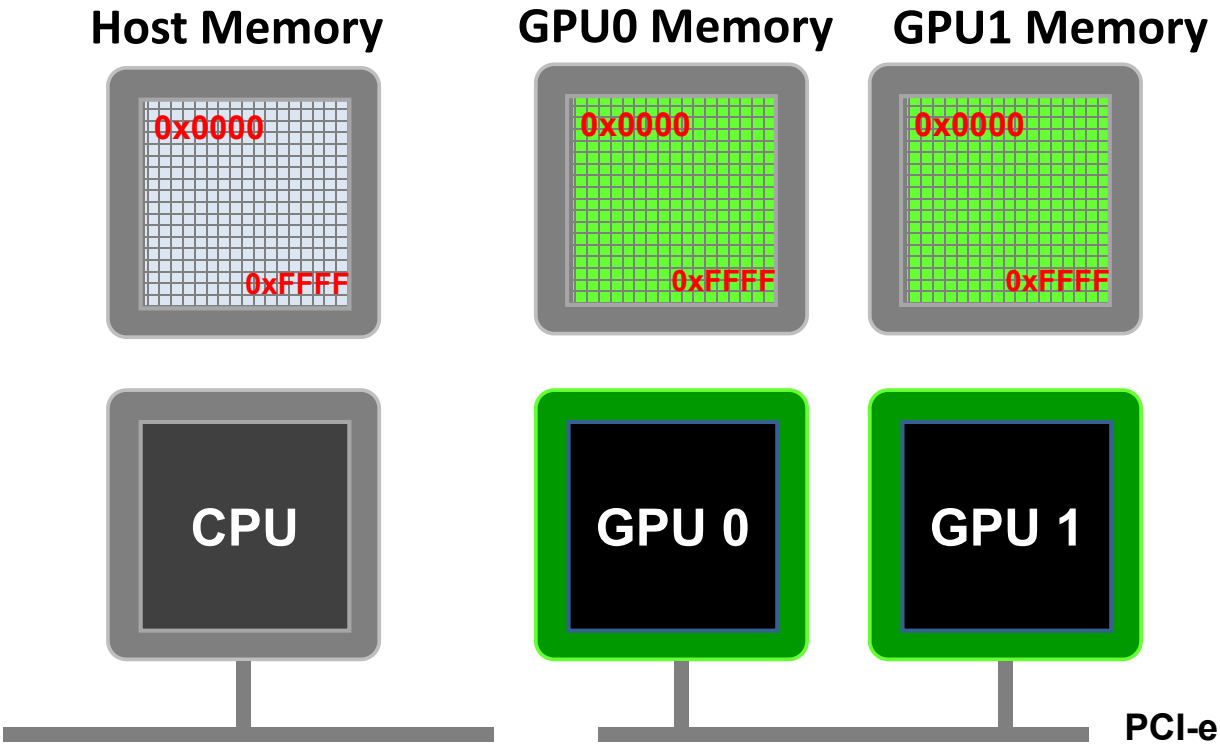
Device No.1 のGPUに、n*8 Byte のメモリを確保

GPU0 → Host → GPU1 データ転送

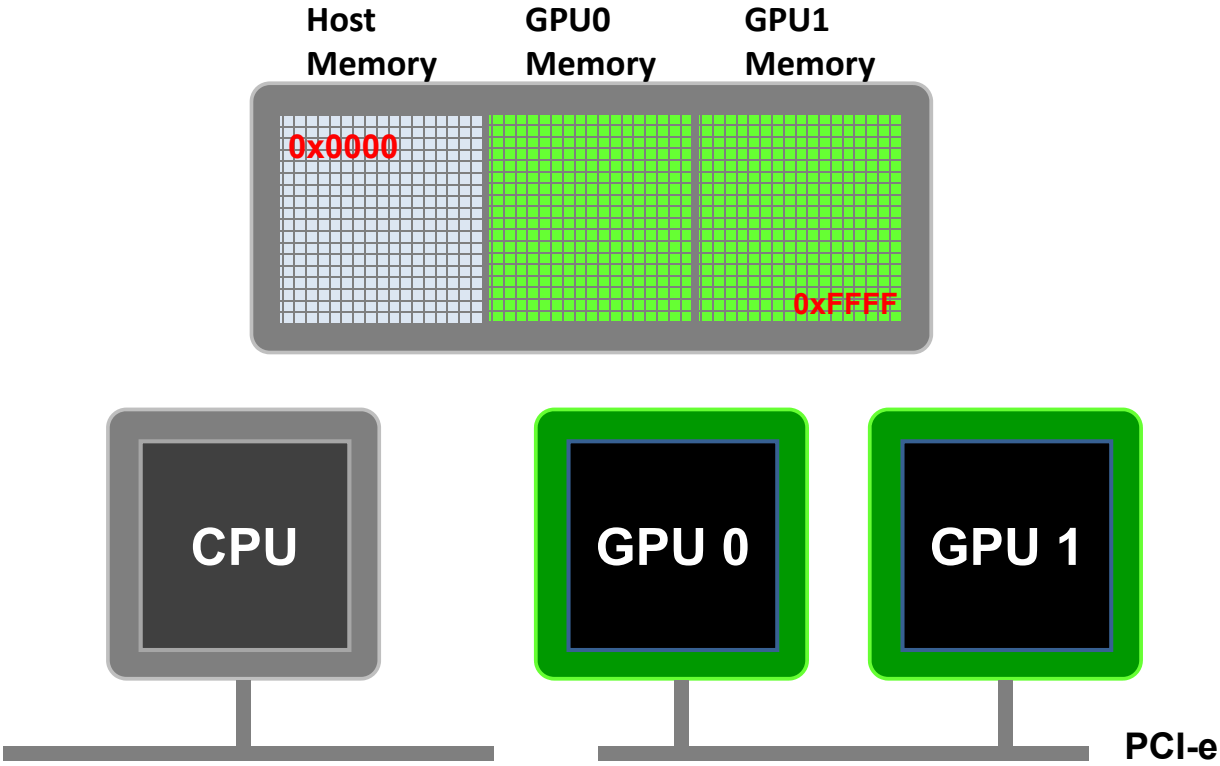

```
double   *A_h,  *B_GPU0,   *B_GPU1;

                                    //  Device No.0 を指定して GPU0 からホストへ
cudaSetDevice(0);
cudaMemcpy( A_h, B_GPU0, n*sizeof(double),
                cudaMemcpyDeviceToHost );

cudaSetDevice(1);                   //  Device No.1 を指定して ホストから GPU1へ
cudaMemcpy( B_GPU1, A_h, n*sizeof(double),
                cudaMemcpyHostToDevice );
```

Source Code #07

Previous Memory Address



Unified Virtual Addressing



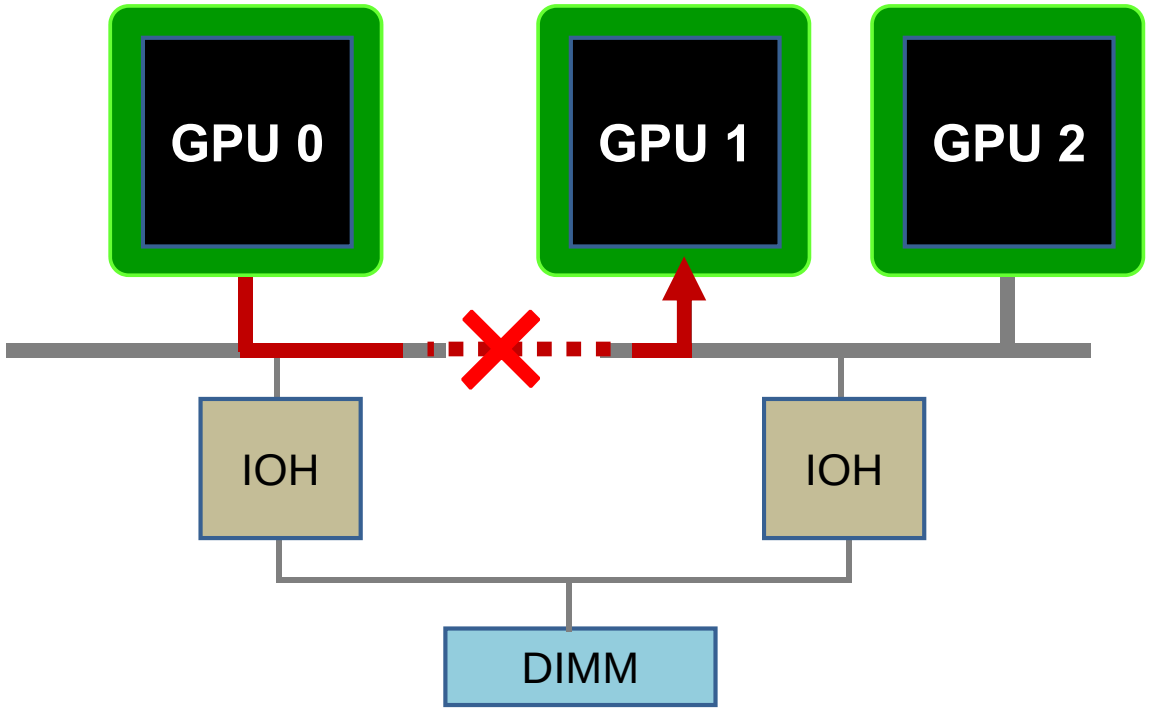
Unified Virtual Addressing


```
cudaMemcpy( A, B, size, cudaMemcpyDeviceToHost );
cudaMemcpy( A, B, size, cudaMemcpyHostToDevice );
cudaMemcpy( A, B, size, cudaMemcpyDeviceToDevice );
cudaMemcpy( A, B, size, cudaMemcpyHostToHost );
```

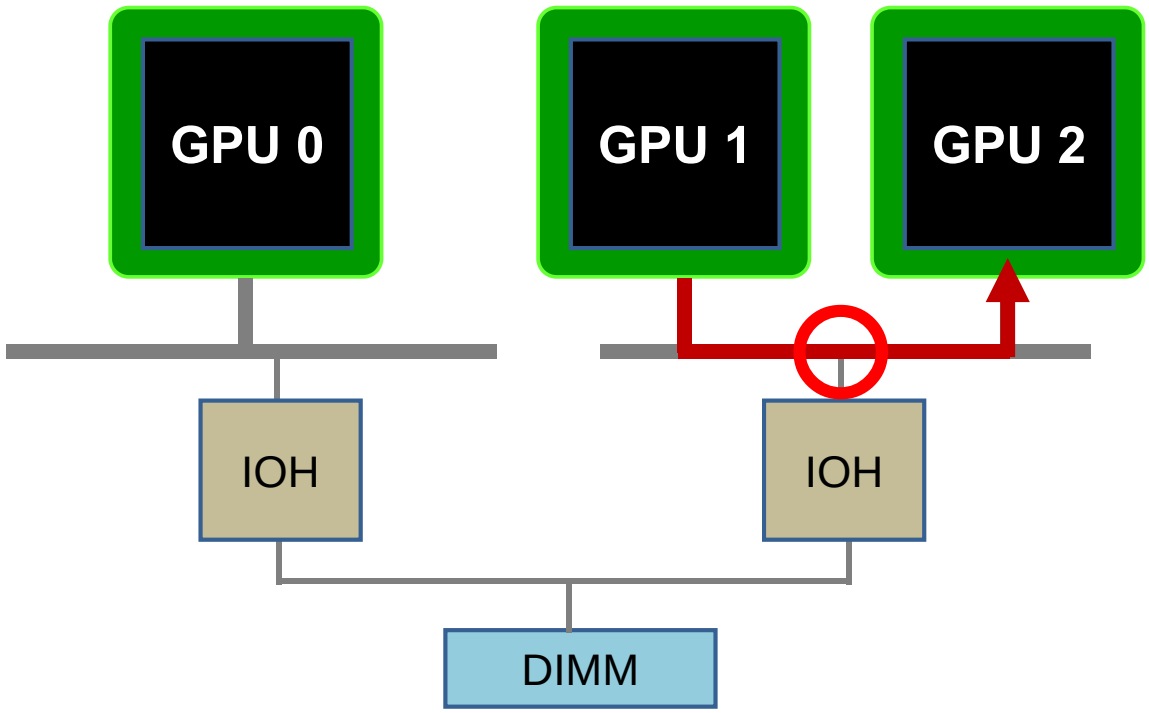
```
cudaMemcpy( A, B, size, cudaMemcpyDefault );
```

Source Code #08 Source Code #09

GPU Direct Ver.2



GPU Direct Ver.2



Peer-to-Peer Direct Transfer

Host メモリを介さない GPU間のDMA転送。

必要条件:

- 64 bit applicaion
- Fermi core 移行の GPU
- CUDA 4.0 以降, Driver v270.41.19 以降
- 同一 IOH 配下に接続された GPU間

```
cudaDeviceCanAccessPeer( &can_access, GPU_id0,
                                      GPU_id1 );
```


Peer-to-Peer Direct Transfer

■ Peer アクセスの可能化:

```
cudaSetDevice(GPU_id0);
cudaDeviceEnablePeerAccess(GPU_id1, 0);
cudaSetDevice(GPU_id1);
cudaDeviceEnablePeerAccess(GPU_id0, 0);
```

Peer-to-Peer Memory Copy

```
cudaMemcpyPeer( GPU0, gpu_id0, GPU1, gpu_id1, size);
```

Source Code #10

演習

『P2P Direct Access プログラミング』

Exercise 9 の cudaMemcpyPeer( ) 代わりに同じ機能のGPUメモリを単純にコピーするカーネル関数を作成し、device 0 – 1 間、device 0 – 2 間、Device 1 – 2 間の転送バンド幅を測定してみす。エラーが出る場合は、どの場合にエラーが出たかを確認する。