14th ANNUAL WORKSHOP 2018

DEVICE MEMORY

Liran Liss

Mellanox Technologies

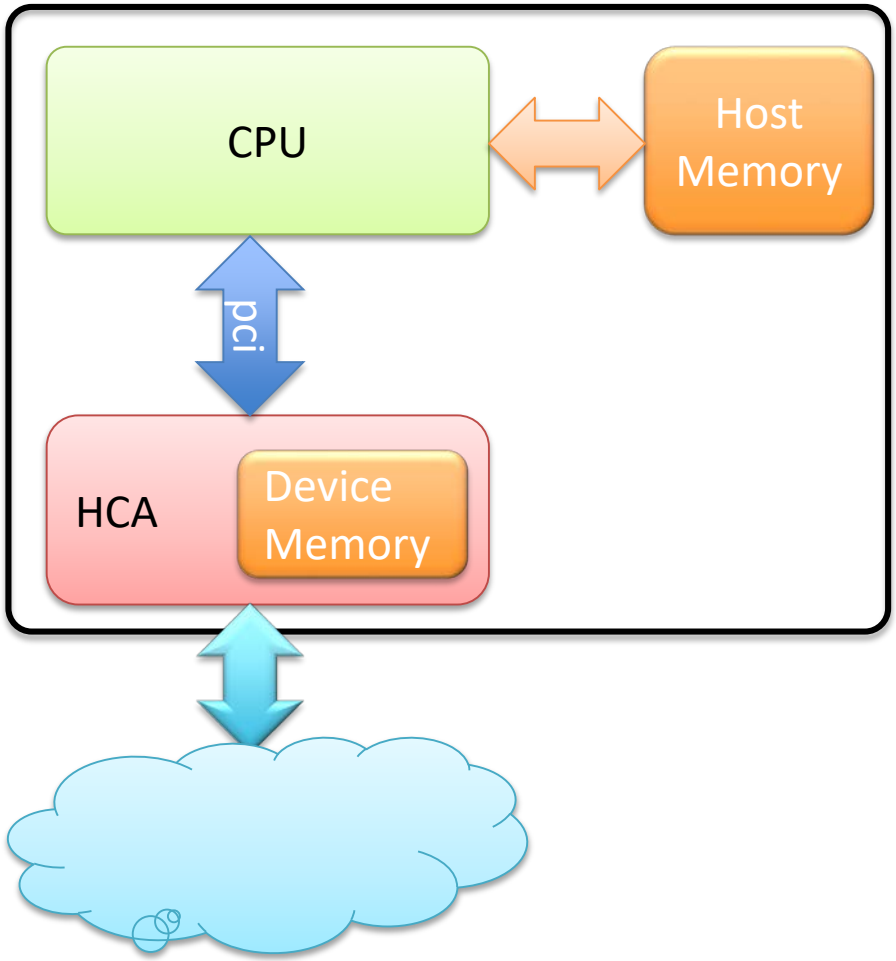
April, 2018

AGENDA

- **Introduction**
- **Motivation**
- **API concepts**
- **User-space Verbs API**
- **Kernel API**
- **Status and future work**

INTRODUCTION

- **Device memory refers to device-specific memory**
- **Resides "closer" to the device than host memory**
  - Higher bandwidth
  - Significantly lower latency
  - Deterministic performance
    - No NUMA effects
    - No contention
  - Does not consume system memory resources
- **Provides applications with a new type of memory target for RDMA transactions**



MOTIVATION

- **Low-latency memory for remote operations**
  - Distributed locks
  - Counters
- **Staging buffer for outgoing messages**
  - Pipeline a device memory update + network transaction
  - Avoid PCI read round-trip to fetch data
- **Staging buffer for peer devices**
  - E.g., a Controller Memory Buffer (CMB)
- **Application level multicast**
  - Write once to device memory, send multiple times to different destinations

CONCEPTS

- **Device memory is a shared resource**
  - A process may ***allocate*** device memory buffer(s) for its own use
  - Allocation is isolated from other processes
- **Device memory buffers are a new type of object**
  - Do not reside in process address space
  - Each object forms its own, zero-based, address space
  - Bound to corresponding device context
- **Device memory buffers may be accessed from**
  - CPU via API
  - RDMA operations via memory regions
- **Registration is independent from device memory allocation**
  - A single buffer may support multiple memory regions

USER PROGRAMMING MODEL

Operation	Memory	Device memory
Buffer allocation	malloc() / free()	ibv_alloc_dm() / ibv_free_dm()
Buffer registration	ibv_reg_mr() / ibv_free_mr()	ibv_reg_dm_mr() / ibv_free_mr()
Host access	memcpy()	ibv_memcpy_to_dm() / Ibv_memcpy_from_dm()
Local network access	ibv_post_send() / ibv_post_recv()	
Remote network access	Target of incoming RDMA-R/W/Atomics	

DEVICE MEMORY CAPABILITIES

```
struct ibv_device_attr_ex {
     ...
     uint64_t max_dm_size;
     ...
};

int ibv_query_device_ex(struct ibv_context *context,
             const struct ibv_query_device_ex_input *input,
             struct ibv_device_attr_ex *attr);
```

DEVICE MEMORY ALLOCATION

```
struct ibv_alloc_dm_attr {
    size_t length;        /* byte granularity */
    uint32_t log_align_req;
    uint32_t comp_mask;   /* enable future extensions */
};

struct ibv_dm {
    struct ibv_context *context;
    int (*memcpy_to_dm)(struct ibv_dm *dm, uint64_t dm_offset,
                        const void *host_addr, size_t length);
    int (*memcpy_from_dm)(void *host_addr, struct ibv_dm *dm,
                        uint64_t dm_offset, size_t length);
    uint32_t comp_mask;
};

struct ibv_dm *ibv_alloc_dm(struct ibv_context *context,
                            struct ibv_alloc_dm_attr *dm_attr);

int ibv_free_dm(struct ibv_dm *dm);
```

REGISTRATION

```
struct ibv_mr *ibv_reg_dm_mr(struct ibv_pd *pd, struct ibv_dm *dm,
                             uint64_t dm_offset,
                             size_t length, unsigned int access);
```

- **Offset relative to device memory object**

HOST ACCESS

```
int ibv_memcpy_to_dm(struct ibv_dm *dm, uint64_t dm_offset,
              const void *host_addr, size_t length);

int ibv_memcpy_from_dm(void *host_addr, struct ibv_dm *dm,
                uint64_t dm_offset, size_t length);
```

- **Offset relative to device memory object**

KERNEL DEVICE MEMORY

- **Similar API to user-space**
  - ib_alloc_dm()
  - ib_free_dm()
  - ib_reg_dm_mr()
  - ib_memcpy_to_dm()
  - Ib_memcpy_from_dm()
- **LKey required**
- **Integration with peer-to-peer DMA**

STATUS

- **User DM API accepted for kernel 4.17**
  - Uses IOCTL UAPI (!)
- **Future work**
  - Kernel device memory
  - Additional device memory types
    - Device memory classes
    - Cache-coherent device memory

14th ANNUAL WORKSHOP 2018

THANK YOU

Liran Liss

Mellanox Technologies