NAME

ibv_create_qp_ex, ibv_destroy_qp − create or destroy a queue pair (QP)

SYNOPSIS

#include <infiniband/verbs.h>

struct ibv_qp *ibv_create_qp_ex(struct ibv_context **context*,
struct ibv_qp_init_attr_ex **qp_init_attr*);

int ibv_destroy_qp(struct ibv_qp **qp*);

DESCRIPTION

ibv_create_qp_ex() creates a queue pair (QP) associated with the protection domain *pd*. The argument *qp_init_attr_ex* is an ibv_qp_init_attr_ex struct, as defined in <infiniband/verbs.h>.

```
struct ibv_qp_init_attr_ex {
        void               *qp_context;    /* Associated context of the QP */
        struct ibv_cq      *send_cq;       /* CQ to be associated with the Send Queue (SQ) */
        struct ibv_cq      *recv_cq;       /* CQ to be associated with the Receive Queue (RQ) */
        struct ibv_srq     *srq;          /* SRQ handle if QP is to be associated with an SRQ, otherwise NULL
        struct ibv_qp_cap  cap;           /* QP capabilities */
        enum ibv_qp_type   qp_type;       /* QP Transport Service Type: IBV_QPT_RC, IBV_QPT_UC, I
        int                sq_sig_all;    /* If set, each Work Request (WR) submitted to the SQ generates a com
        uint32_t           comp_mask;     /* Identifies valid fields */
        struct ibv_pd      *pd;           /* PD to be associated with the QP */
        struct ibv_xrcd    *xrcd;         /* XRC domain to be associated with the target QP */
        enum ibv_qp_create_flags create_flags;       /* Creation flags for this QP */
        uint16_t           max_tso_header; /* Maximum TSO header size */
        struct ibv_rwq_ind_table *rwq_ind_tbl;  /* Indirection table to be associated with the QP */
        struct ibv_rx_hash_conf  rx_hash_conf;  /* RX hash configuration to be used */
        uint32_t           source_qpn;    /* Source QP number, creation flag IBV_QP_CREATE_SOURCE_
        uint64_t           send_ops_flags; /* Select which QP send ops will be defined in struct ibv_qp_ex. U
};

struct ibv_qp_cap {
        uint32_t           max_send_wr;   /* Requested max number of outstanding WRs in the SQ */
        uint32_t           max_recv_wr;   /* Requested max number of outstanding WRs in the RQ */
        uint32_t           max_send_sge;  /* Requested max number of scatter/gather (s/g) elements in a WR
        uint32_t           max_recv_sge;  /* Requested max number of s/g elements in a WR in the SQ */
        uint32_t           max_inline_data;/* Requested max number of data (bytes) that can be posted inline
};
enum ibv_qp_create_flags {
        IBV_QP_CREATE_BLOCK_SELF_MCAST_LB      = 1 << 1, /* Prevent self multicast loopback */
        IBV_QP_CREATE_SCATTER_FCS              = 1 << 8, /* FCS field will be scattered to host memory
        IBV_QP_CREATE_CVLAN_STRIPPING          = 1 << 9, /* CVLAN field will be stripped from inc
        IBV_QP_CREATE_SOURCE_QPN               = 1 << 10, /* The created QP will use the source_qpn a
        IBV_QP_CREATE_PCI_WRITE_END_PADDING    = 1 << 11, /* Incoming packets will be padded
};
struct ibv_rx_hash_conf {
        uint8_t            rx_hash_function;      /* RX hash function, use enum ibv_rx_hash_function_flags */
        uint8_t            rx_hash_key_len;       /* RX hash key length */
        uint8_t            *rx_hash_key;          /* RX hash key data */
        uint64_t           rx_hash_fields_mask;   /* RX fields that should participate in the hashing, use enum
};
enum ibv_rx_hash_fields {
        IBV_RX_HASH_SRC_IPV4          = 1 << 0,
```

```
                IBV_RX_HASH_DST_IPV4         = 1 << 1,
                IBV_RX_HASH_SRC_IPV6         = 1 << 2,
                IBV_RX_HASH_DST_IPV6         = 1 << 3,
                IBV_RX_HASH_SRC_PORT_TCP     = 1 << 4,
                IBV_RX_HASH_DST_PORT_TCP     = 1 << 5,
                IBV_RX_HASH_SRC_PORT_UDP     = 1 << 6,
                IBV_RX_HASH_DST_PORT_UDP     = 1 << 7,
                IBV_RX_HASH_IPSEC_SPI        = 1 << 8,
                /* When using tunneling protocol, e.g. VXLAN, then we have an inner (encapsulated packet) and oute
                 * For applying RSS on the inner packet, then the following field should be set with one of the L3/L4
                */
                IBV_RX_HASH_INNER            = (1UL << 31),
};
struct ibv_qp_create_send_ops_flags {
                IBV_QP_EX_WITH_RDMA_WRITE                = 1 << 0,
                IBV_QP_EX_WITH_RDMA_WRITE_WITH_IMM = 1 << 1,
                IBV_QP_EX_WITH_SEND                      = 1 << 2,
                IBV_QP_EX_WITH_SEND_WITH_IMM             = 1 << 3,
                IBV_QP_EX_WITH_RDMA_READ                 = 1 << 4,
                IBV_QP_EX_WITH_ATOMIC_CMP_AND_SWP   = 1 << 5,
                IBV_QP_EX_WITH_ATOMIC_FETCH_AND_ADD          = 1 << 6,
                IBV_QP_EX_WITH_LOCAL_INV                 = 1 << 7,
                IBV_QP_EX_WITH_BIND_MW                   = 1 << 8,
                IBV_QP_EX_WITH_SEND_WITH_INV             = 1 << 9,
                IBV_QP_EX_WITH_TSO                       = 1 << 10,
};
```

The function **ibv_create_qp_ex()** will update the *qp_init_attr_ex*->cap struct with the actual QP values of the QP that was created; the values will be greater than or equal to the values requested.

ibv_destroy_qp() destroys the QP *qp*.

RETURN VALUE

ibv_create_qp_ex() returns a pointer to the created QP, or NULL if the request fails. Check the QP number (**qp_num**) in the returned QP.

ibv_destroy_qp() returns 0 on success, or the value of errno on failure (which indicates the failure reason).

NOTES

The attributes max_recv_wr and max_recv_sge are ignored by **ibv_create_qp_ex()** if the QP is to be associated with an SRQ.

The attribute source_qpn is supported only on UD QP, without flow steering RX should not be possible.

Use **ibv_qp_to_qp_ex()** to get the *ibv_qp_ex* for accessing the send ops iterator interface, when QP create attr IBV_QP_INIT_ATTR_SEND_OPS_FLAGS is used.

ibv_destroy_qp() fails if the QP is attached to a multicast group.

IBV_QPT_DRIVER does not represent a specific service and is used for vendor specific QP logic.

SEE ALSO

ibv_alloc_pd(3), **ibv_modify_qp**(3), **ibv_query_qp**(3), **ibv_create_rwq_ind_table**(3)

AUTHORS

Yishai Hadas <yishaih@mellanox.com>