## NAME

ibv_post_srq_ops − perform on a special shared receive queue (SRQ) configuration manipulations

## SYNOPSIS

**#include <infiniband/verbs.h>**

**int ibv_post_srq_ops(struct ibv_srq** **srq*, **struct ibv_ops_wr** **wr*,
**struct ibv_ops_wr** ***bad_wr*);

## DESCRIPTION

The **ibv_post_srq_ops()** performs series of offload configuration manipulations on special types of SRQ *srq*. Currenlty it is used to configure tag matching SRQ. Series of configuration operations defined by linked lists of struct ibv_ops_wr elements starting from *wr*.

```
struct ibv_ops_wr {
        uint64_t            wr_id;   /* User defined WR ID */
        /* Pointer to next WR in list, NULL if last WR */
        struct ibv_ops_wr     *next;
        enum ibv_ops_wr_opcode  opcode;  /* From enum ibv_ops_wr_opcode */
        int                 flags;   /* From enum ibv_ops_flags */
        struct {
                /* Number of unexpected messages
                 * handled by SW */
                uint32_t unexpected_cnt;
                /* Input parameter for the DEL opcode
                 * and output parameter for the ADD opcode */
                uint32_t handle;
                struct {
                        uint64_t            recv_wr_id;   /* User defined WR ID for TM
                        struct ibv_sge       *sg_list; /* Pointer to the s/g array */
                        int                 num_sge; /* Size of the s/g array */
                        uint64_t            tag;
                        uint64_t            mask;    /* Incoming message considered m
                                                        TMH.tag & entry.mask == entry.tag */
                } add;
        } tm;
};
```

First part of struct ibv_ops_wr retains ibv_send_wr notion. Opcode defines operation to perform. Currently supported IBV_WR_TAG_ADD, IBV_WR_TAG_DEL and IBV_WR_TAG_SYNC values. See below for detailed description.

To allow reliable data delivery TM SRQ maintains special low level synchronization primitive - phase synchronization. Receive side message handling comprises two concurrent activities - posting tagged buffers by SW and receiving incoming messages by HW. This process considered coherent only if all unexpected messages received by HW is completely processed in SW. To pass to hardware number of processed unexpected messages unexpected_cnt field should be used and IBV_OPS_TM_SYNC flag should be set.

To request WC for tag list operations IBV_OPS_SIGNALED flags should be passed. In this case WC will be generated on TM SRQ's CQ, provided wr_id will identify WC.

Opcode IBV_WR_TAG_ADD used to add tag entry to tag matching list. Tag entry consists of SGE list, tag & mask (matching parameters), user specified opaque wr_id (passed via recv_wr_id field) and uniquely identified by handle (returned by driver). Size of tag matching list is limited by max_num_tags. SGE list size is limited by max_sge.

Opcode IBV_WR_TAG_DEL removes previously added tag entry. Field handle should be set to value returned by previously performed IBV_WR_TAG_ADD operation. Operation may fail due to concurrent tag consumption - in this case IBV_WC_TM_ERR status will be returned in WC.

Opcode IBV_WR_TAG_SYNC may be used if no changes to matching list required, just to updated unexpected messages counter.

IBV_WC_TM_SYNC_REQ flag returned in list operation WC shows that counter synchronization required. This flag also may be returned by unexpected receive WC, asking for IBV_WR_TAG_SYNC operation to keep TM coherence consistency.

## RETURN VALUE

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

## SEE ALSO

**ibv_create_srq_ex**(3),

## AUTHORS

Artemy Kovalyov <artemyko@mellanox.com>