

NAME

`rdma_create_ep` - Allocate a communication identifier and optional QP.

SYNOPSIS

```
#include <rdma/rdma_cma.h>
```

```
int rdma_create_ep (struct rdma_cm_id **id, struct rdma_addrinfo *res, struct ibv_pd *pd, struct  
ibv_qp_init_attr *qp_init_attr);
```

ARGUMENTS

id	A reference where the allocated communication identifier will be returned.
res	Address information associated with the <code>rdma_cm_id</code> returned from <code>rdma_getaddrinfo</code> .
pd	Optional protection domain if a QP is associated with the <code>rdma_cm_id</code> .
qp_init_attr	Optional initial QP attributes.

DESCRIPTION

Creates an identifier that is used to track communication information.

RETURN VALUE

Returns 0 on success, or -1 on error. If an error occurs, `errno` will be set to indicate the failure reason.

NOTES

After resolving address information using `rdma_getaddrinfo`, a user may use this call to allocate an `rdma_cm_id` based on the results.

If the `rdma_cm_id` will be used on the active side of a connection, meaning that `res->ai_flag` does not have `RAI_PASSIVE` set, `rdma_create_ep` will automatically create a QP on the `rdma_cm_id` if `qp_init_attr` is not NULL. The QP will be associated with the specified protection domain, if provided, or a default protection domain if not. Users should see `rdma_create_qp` for details on the use of the `pd` and `qp_init_attr` parameters. After calling `rdma_create_ep`, the returned `rdma_cm_id` may be connected by calling `rdma_connect`. The active side calls `rdma_resolve_addr` and `rdma_resolve_route` are not necessary.

If the `rdma_cm_id` will be used on the passive side of a connection, indicated by having `res->ai_flag` `RAI_PASSIVE` set, this call will save the provided `pd` and `qp_init_attr` parameters. When a new connection request is retrieved by calling `rdma_get_request`, the `rdma_cm_id` associated with the new connection will automatically be associated with a QP using the `pd` and `qp_init_attr` parameters. After

calling `rdma_create_ep`, the returned `rdma_cm_id` may be placed into a listening state by immediately calling `rdma_listen`. The passive side call `rdma_bind_addr` is not necessary. Connection requests may then be retrieved by calling `rdma_get_request`.

The newly created `rdma_cm_id` will be set to use synchronous operation. Users that wish asynchronous operation must migrate the `rdma_cm_id` to a user created event channel using `rdma_migrate_id`.

Users must release the created `rdma_cm_id` by calling `rdma_destroy_ep`.

SEE ALSO

`rdma_cm(7)`, `rdma_getaddrinfo(3)`, `rdma_create_event_channel(3)`, `rdma_connect(3)`, `rdma_listen(3)`,
`rdma_destroy_ep(3)`, `rdma_migrate_id(3)`