Exercise 2 Report

By Omer Ben Moshe and Yuvali Leibovich

The objective of this exercise is to implement a client-server communication system using RDMA (Remote Direct Memory Access) with Infiniband verbs and analyze its performance. The focus is on setting up a reliable RDMA channel and measuring throughput for varying message sizes to gain insights into the efficiency of RDMA for data transfer.

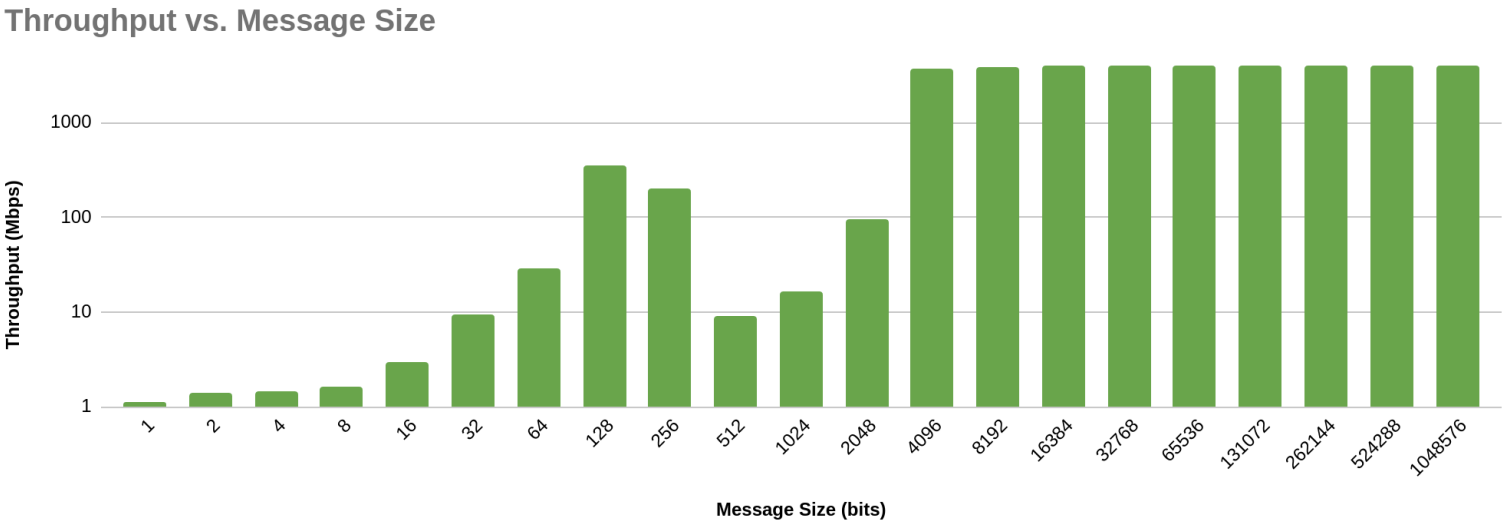
The implementation involves two primary components: the client and the server, both written in C and utilizing the Infiniband Verbs API (libibverbs) for RDMA operations.

The client connects to the server by exchanging Infiniband address information over a TCP connection. Once connected, the client iterates over different message sizes, starting from 1 bit and increasing up to 1 MB. For each message size, the client enters a warm-up phase to stabilize the connection. During this phase, a 1000 messages are sent. After warming up, the client measures the time taken to send 6000, and calculates the throughput in Mbps.

The server program is designed to listen for incoming RDMA connections from the client and receive data. The server initializes its RDMA resources similarly to the client and waits for connection requests. Upon accepting a connection, the server enters a loop to read and process incoming data from the client. After all messages are received in each iteration, the server sends the client an ack message. Any read errors are handled appropriately to ensure robust communication.

The throughput measurements indicated high data transfer rates, demonstrating the benefits of RDMA for high-performance networking applications. The results showed that RDMA provides significant performance improvements, particularly for large message sizes.

In Exercise #1, throughput was measured using TCP sockets. This exercise highlights the advantages of RDMA in achieving low-latency, high-throughput data transfers. RDMA bypasses the operating system's involvement in data transfers, allowing direct memory access between the client and server. This results in reduced latency and increased data transfer rates compared to traditional network communication methods.



Message Size	Throughput	Units
1	0.105532	Mbps
2	0.392418	Mbps
4	0.485619	Mbps
8	0.633583	Mbps
16	1.977381	Mbps
32	8.473574	Mbps
64	28.22	Mbps
128	348.772321	Mbps
256	197.738087	Mbps
512	8.008396	Mbps
1024	15.354233	Mbps
2048	92.91746	Mbps
4096	3724.968214	Mbps
8192	3861.838853	Mbps
16384	3931.971648	Mbps
32768	3966.910676	Mbps
65536	3984.741098	Mbps
131072	3995.184471	Mbps
262144	4001.60064	Mbps
524288	4004.158986	Mbps
1048576	4004.650734	Mbps