Lab #2: Verbs API Throughput

Goal

The goal of this exercise is to apply the Verbs API for measuring point-to-point (unidirectional) throughput, by writing a C application (using an existing template in C provided with this exercise).

What and How to measure?

Same as with exercise #1, only with the Verbs API instead of TCP Sockets. Note that completion on the client does not necessarily indicate the server has read the message, so while you should use RDMA WRITE to send the data to the server, the control message back to the client should be sent with IBV_WR_SEND.

Items to pay attention to:

1. The items from exercise #1 apply here as well. Same output, so remove any debug "printf"-s.
2. Building the provided source code requires custom arguments, and it will only run on the course setup (more details below):
   ```
   #> gcc bw_template.c -libverbs -o server && ln -s server client
   ```
3. Remember you have to post receive-buffers before the other side sends you a message.
4. Use IBV_SEND_INLINE whenever possible (you need to find the max_inline limit).
5. Make sure your timer resolution is adequate: seconds may be too low a resolution for benchmarking sub-microsecond events…

Bonus for best result: +5 points

The student submission reaching the best result in the class on the course hardware setup (and satisfies requirements) will get additional points, limited by 100 for the entire exercise.

Setup & Submission

For this exercise you need to use one of the two pairs of nodes installed with Mellanox Connect-X3 (56Gb) NICs: dell-srv2 + dell-srv3 OR dell-srv4 + dell-srv5 (all remotely accessible via SSH). The example supports hostname as well as IP, so you can either run "./client 1.2.3.4" or "./client dell-srv3".

You should submit an archive named "<id1>_<id2>.tgz", containing a Makefile (calling "make" will build the code), building "server" and "client" executables.

We recommend all students use the first pair for **development only** and the second pair to test once the code is complete – this will prevent low throughput results due to concurrent runs.

<u>**Tip:**</u> if you want to develop on your own linux machine, you can install the package "libibverbs-devel" to build the code (but it will not run without a suitable NIC).