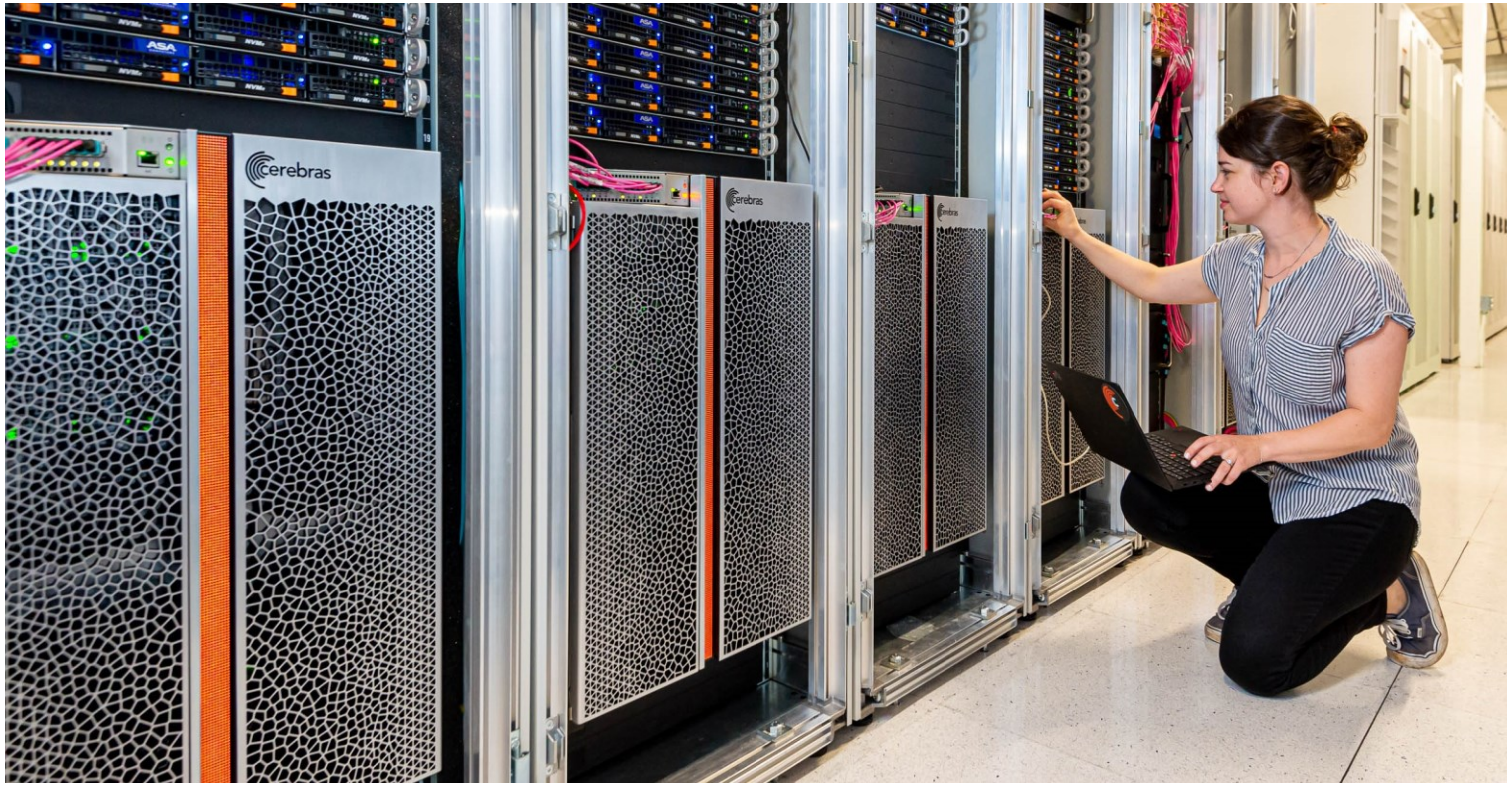
Hands-on 3: GEMV on a Multiple PEs with Collective Communications



Objective

- Perform Matrix-Vector Multiplication on multiple adjacent PEs
- Perform final reduction step using the collective communications library

- Matrix A of size (MxN – M rows, N columns)
    - N columns will be split across the multiple PEs
- Vector x of size N
    - x will be split across multiple PEs
- Vectors b and y of size M

- y = b + A@x

What you must do here

- In *pe_program.csl*:
  - TO DO 1: Fill in mpi_x details for reduction

- Once you have this running, let’s scale it
  - TO DO 2: What needs to be changed to run on more than 2 PEs, initially let’s say 4 PEs and then increasing this number

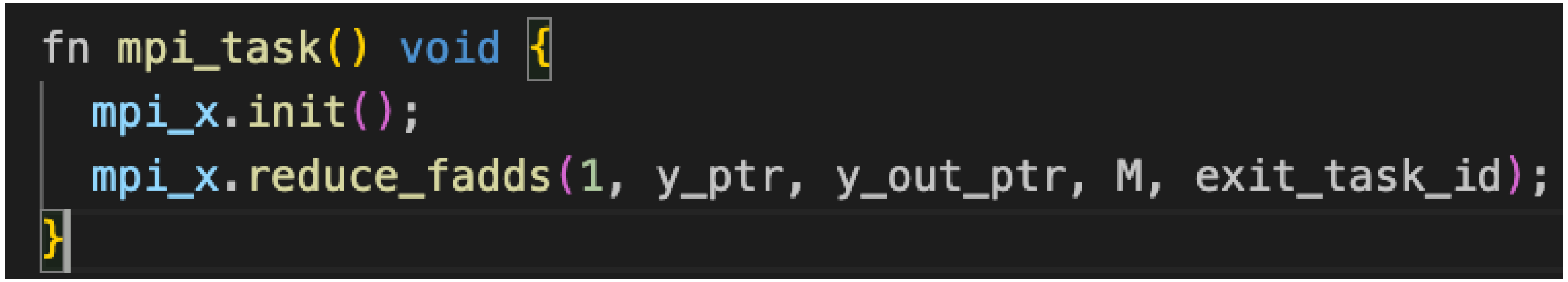
Hints

- Example of using the collective communications library from walk-through 7

Wash-up for hands on exercise three:

- You need to initialize the communications library (it is fairly standard to instantiate this as *mpi* as we have done in the exercise)
  - Then issue the *mpi_x.init* function call
  - To undertake the reduction we use the *reduce_fadds* function from the library

- Remember, you need to activate the exit_task_id as callback to run this task and clean up the stream



```
fn mpi_task() void {
  mpi_x.init();
  mpi_x.reduce_fadds(1, y_ptr, y_out_ptr, M, exit_task_id);
}
```