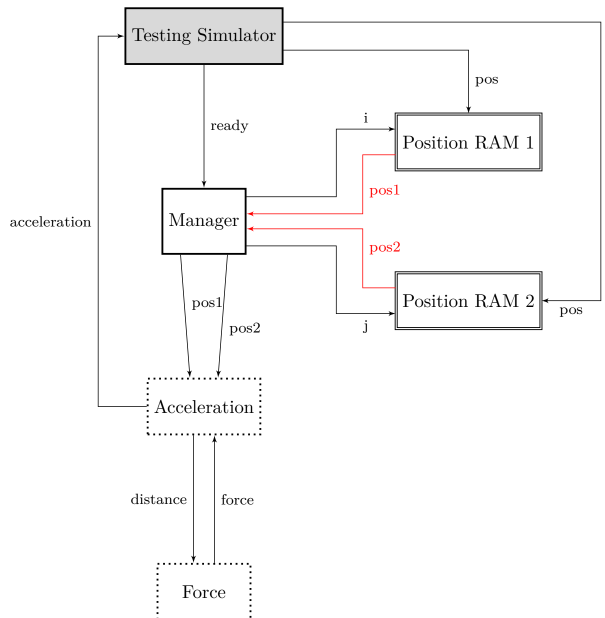

Figure 1: Acceleration module 1.1

1 Modules

2 Function wrappers

Force:

$$f = \left(48 \cdot \epsilon \cdot \left(\frac{\sigma^{12}}{r^{14}}\right)\right) - \left(24 \cdot \epsilon \cdot \left(\frac{\sigma^{6}}{r^{8}}\right)\right)$$

Acceleration:

$$r = x1 - x2mag = \sqrt{r^2} f_s calar = force(mag) f = f_s calar * raccel = f/mass$$

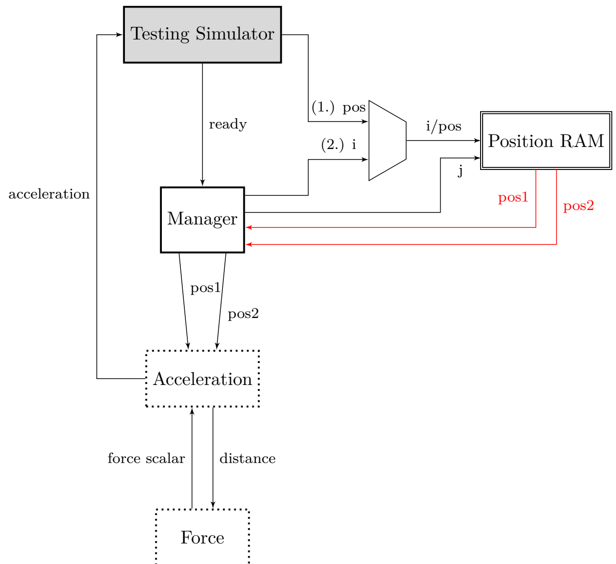

Figure 2: Acceleration module 1.2

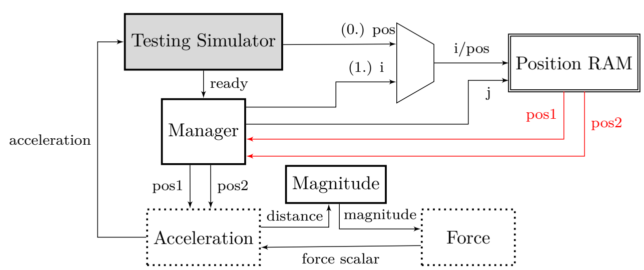

Figure 3: The complete design of the `Acceleration` module. The grayed out box represents `SimulationProcess`, the unfilled boxes represent `SimpleProcesses`, and double edge boxes represents RAM. The trapeze-shaped processes are multiplexor processes that choose between one bus or another, the numbering in the figure shows the priority order. The dotted squares represents a collection of `SimpleProcesses`, for instance `Force`. The internal structure of `Force` can be seen in Figure 9. The large blue dotted squares represent the different modules. The red lines represent the data being communicated from the RAM.

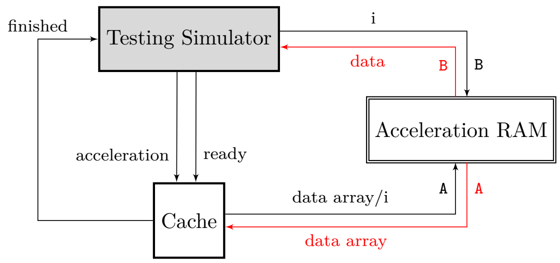

Figure 4: Cache module 1.1

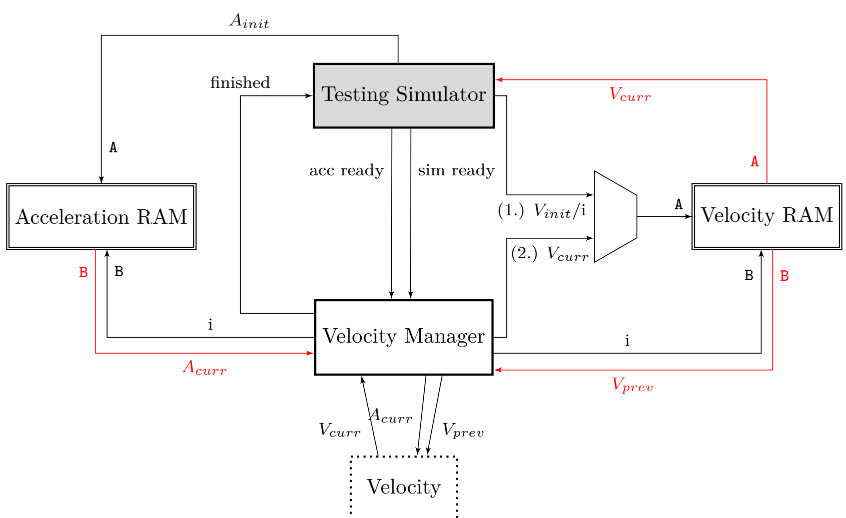

Figure 5: Velocity-(/Position)-update module 1.1

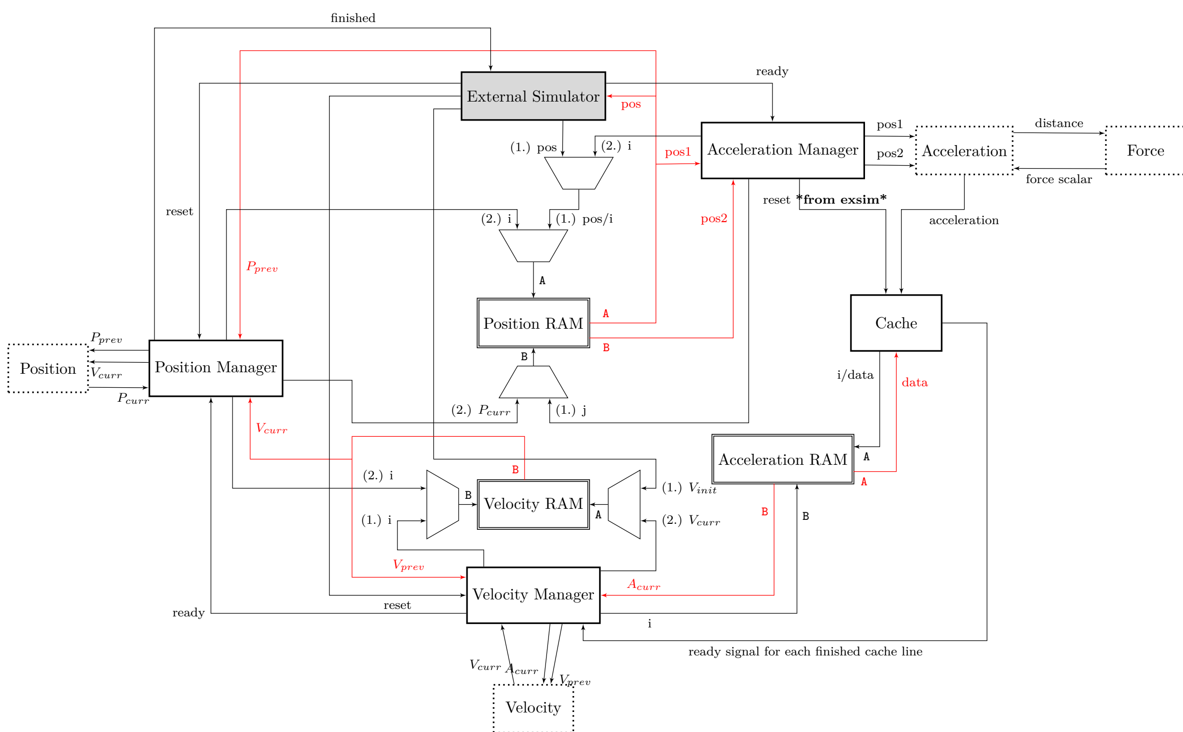

Figure 6: Full LJ circuit

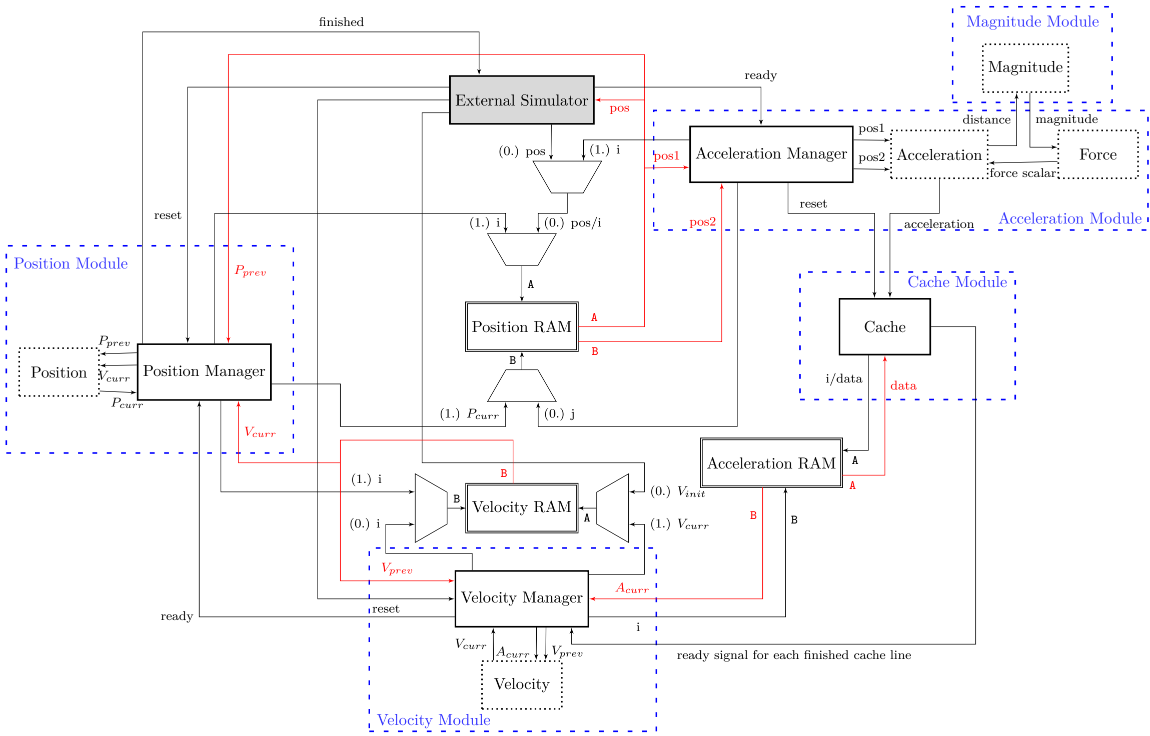

Figure 7: The entire MD simulation network. Each box represents a process. The grayed out box represents `SimulationProcess`, the unfilled boxes represent `SimpleProcesses`, and double edge boxes represents RAM. The trapeze-shaped processes are multiplexor processes that choose between one bus or another, the numbering in the figure shows the priority order. The dotted squares represents a collection of `SimpleProcesses`, for instance `Force`. The internal structure of `Force` can be seen in Figure 9. The large blue dotted squares represent the different modules. The red lines represent the data being communicated from the RAM.

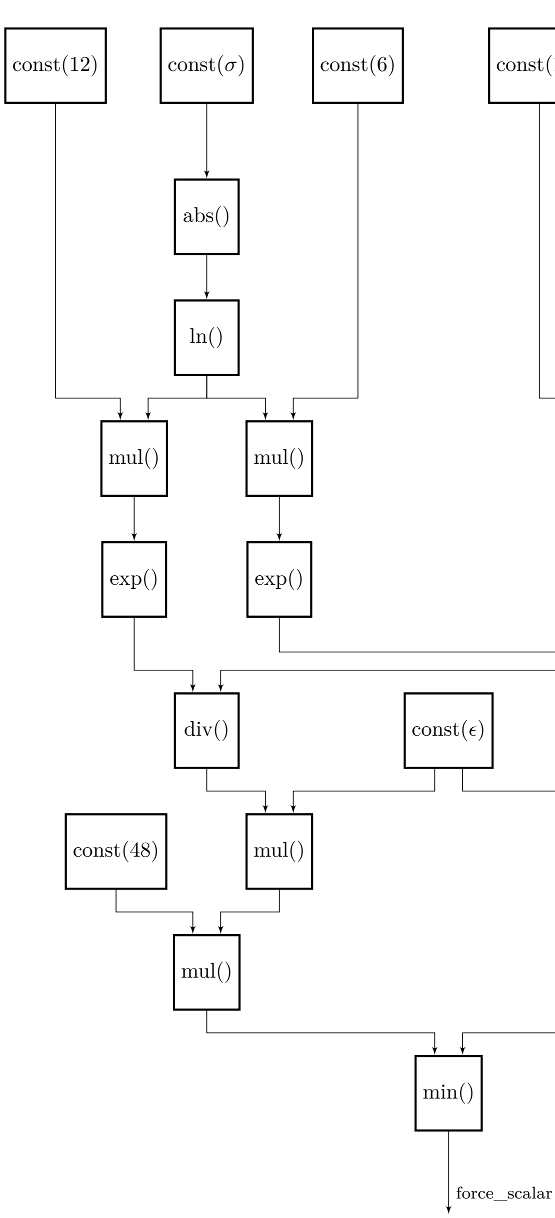

Figure 8: Force wrapper

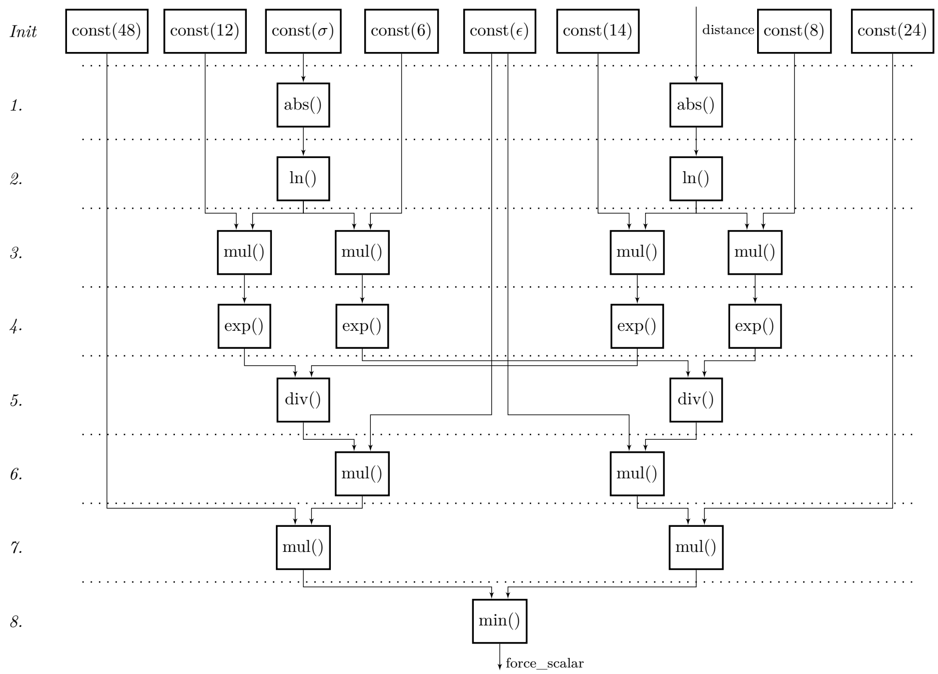

Figure 9: The `Force` calculation representing the entire LJ potential pipeline. Each square is a process that executes the function noted, given the input from the processes above. The horizontal lines represent each layer where the data flows between. The numbering in the left are for referencing purposes only. For each clock cycle, the data flows one layer down.

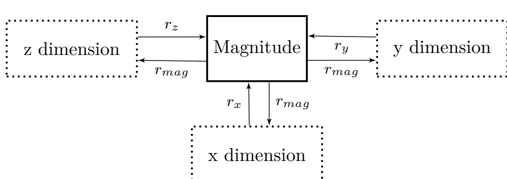

Figure 10: The magnitude is calculated between the different dimensions. The dimensions are running in parallel and communicating with the `Magnitude` module only when the magnitude is needed in the acceleration calculations.

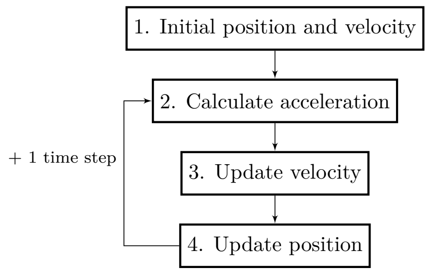

Figure 11: The specific steps of an MD simulation. Each step requires the result of the previous step. After step 4 it loops around to step 2. Each loop is a time step.

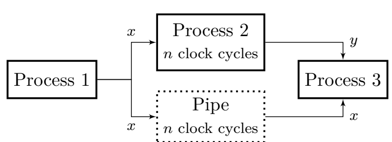

Figure 12: An example of a pipe structure that is inserted to retain data for a certain amount of clock cycles. In the figure, the `Pipe` is a collection of n pipe processes, each one taking one clock cycle.