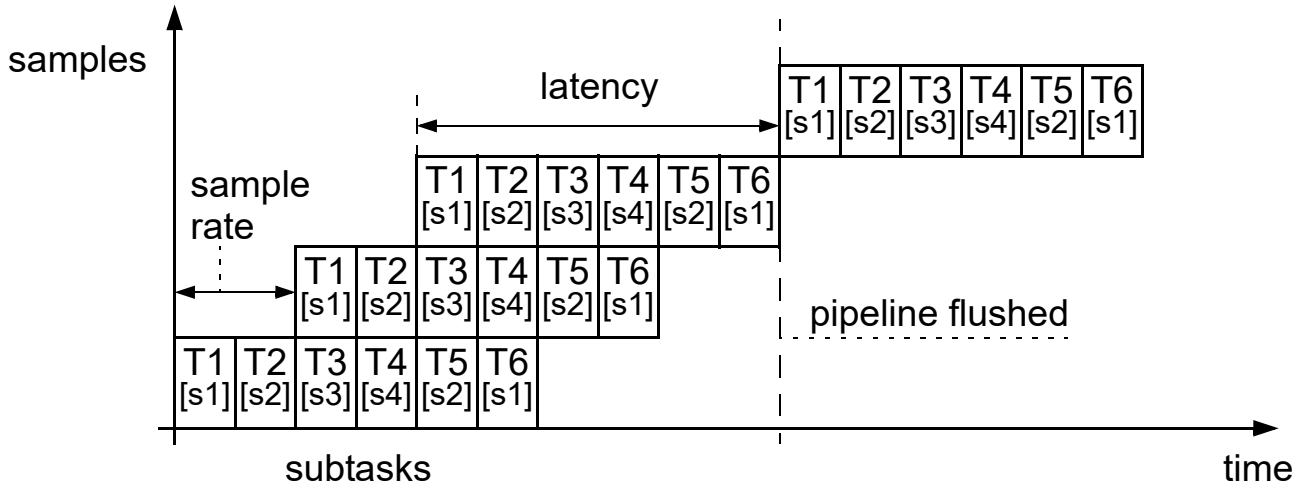
Pipelining

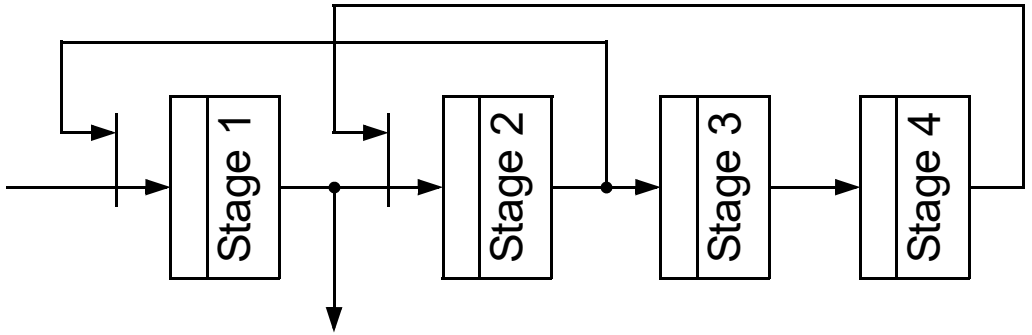
- *Pipelining* - an implementation technique whereby multiple instructions are overlapped in execution
- *Latency (L)* - total number of time units needed to complete the computation on one input sample
- *Sample rate (R)* - the number of time units between two consecutive initiations, where initiation is the start of a computation on an input sample
- *A (pipe) stage* is a piece of HW that is capable of executing certain subtask of the computation
- *The reservation table* is a two-dimensional representation of the data flow during one computation. One dimension corresponds to the stages, and the other dimension corresponds to time units.

- Actions in pipeline: *flushing*, *refilling*, *stalling*.

© Peeter Ellervee

Pipeline - example





© Peeter Ellervee

Pipeline measurements

- Average initiation rate (measure of pipeline performance):

  $R_{init,N\to\infty} = 1 / (R \times t_{stage} + r_{synchro} \times (L-R) \times t_{stage})$

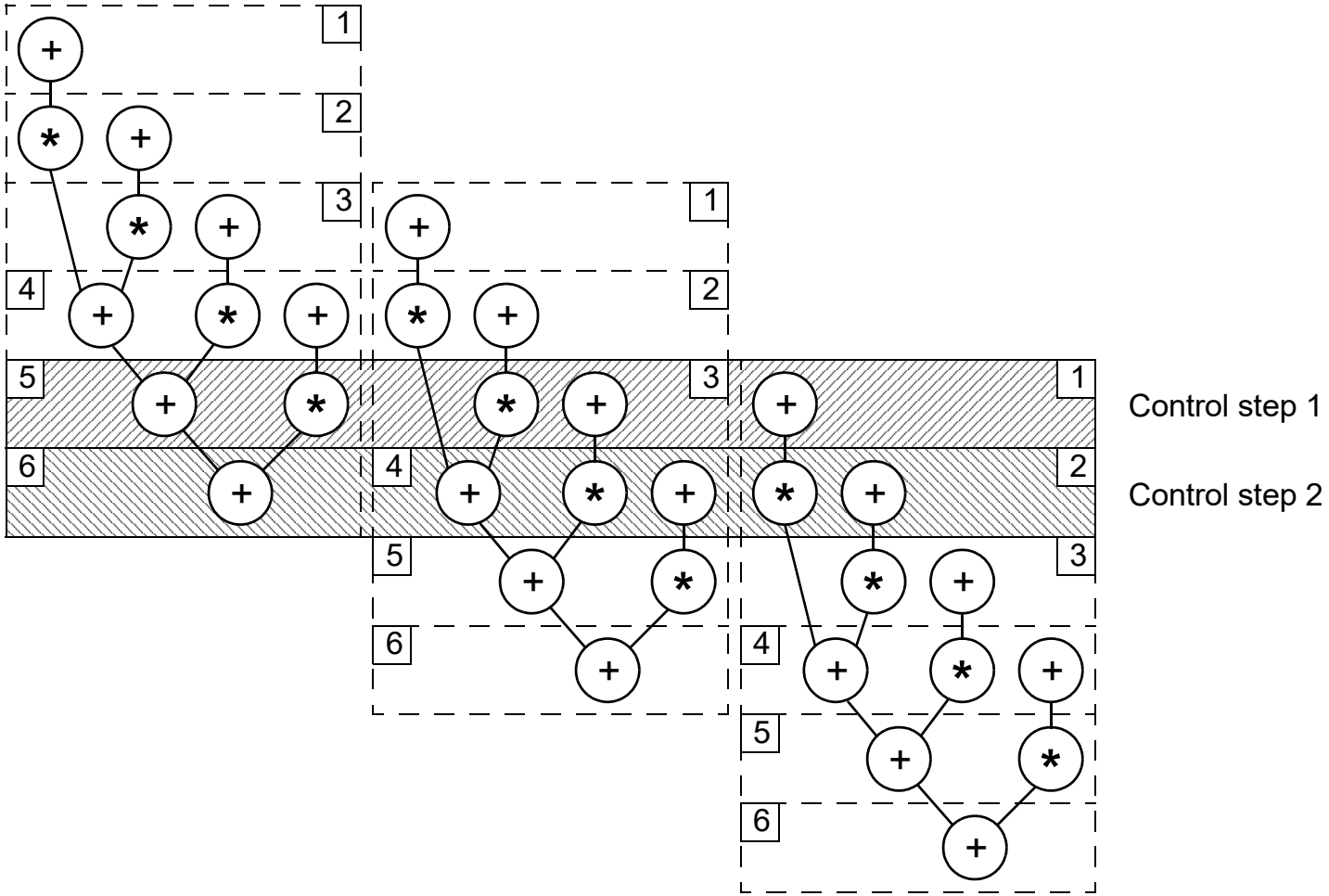
  - R – sample rate, L – latency,
  - t_{stage} – the time one stage needs to complete its subtask,
  - $r_{synchro} = N_{flush} / N$ – resynchronization rate,
  - N_{flush} – the number of input samples that cause flushing,
  - N – number of input samples.

Functional pipelining

- In conventional pipelining, stages have physical equivalents, i.e. the stage hardware is either shared completely in different time units or not shared at all.
- In the case of large functional units, there is no physical stage corresponding to the logical grouping of operations in a time step.
- A *control step* corresponds to a group of time steps that overlap in time. Operations belonging to different control steps may share functional units without conflict.
- Operations, belonging to the time steps $s+n\times L$, for $n\geq 0$, are executed simultaneously and cannot share hardware.

© Peeter Ellervee

Functional pipelining of 8-point FIR filter



© Peeter Ellervee