



Fig. 6. Examples of connector circuits in Reo

5.5 Ordering

The connector in Fig. 6.(d) consists of three channels: **ab**, **ac**, and **bc**. The channels **ab** and **ac** are **SyncDrain** and **Sync**, respectively. The channel **bc** is of type **FIFO1**. The behavior of this connector can be seen as imposing an order on the flow of the data items written to **a** and **b**, through to **c**: the data items obtained by successive read operations on **c** consist of the first data item written to **a**, followed by the first data item written to **b**, followed by the second data item written to **a**, followed by the second data item written to **b**, etc. The coordination pattern imposed by our connector can be summarized as $c = (ab)^*$, meaning the sequence of values that appear through **c** consist of zero or more repetitions of the pairs of values written to **a** and **b**, in that order.

5.6 Sequencer

Consider the connector in Fig. 6.(e). The enclosing box represents the fact that the details of this connector are abstracted away and it provides only the four nodes of the channel ends **a**, **b**, **c**, and **d** for other entities (connectors and/or component instances) to (in this case) read from. Inside this connector, we have four **Sync**, an initialized **FIFO1**, and three **FIFO1** channels connected together. The initialized **FIFO1** channel is the leftmost one and is initialized to have a data item in its buffer, as indicated by the presence of the symbol “o” in the box representing its buffer. The actual value of this data item is irrelevant. The read operations on the nodes (with channel ends) **a**, **b**, **c**, and **d** can succeed only in the strict left to right order. This connector implements a generic sequencing protocol: we can parameterize this connector to have as many nodes as we want, simply by inserting more (or fewer) **Sync** and **FIFO1** channel pairs, as required.