Chapter 9. BDF Domain

Authors: Joseph T. Buck

9.1 Writing BDF Stars

BDF stars are written in almost exactly the same way as SDF stars are written. When the go method of the star is executed, it is guaranteed that all required input data are present, and after execution, any particles generated by the star are correctly sent off to their destinations. The only additional thing the star writer must know is how to specify that a porthole is conditional on other portholes. This is accomplished with a method of the class BDFPort-Hole called setBDFParams.

The setBDFParams method takes four arguments. The first argument is the number of particles transferred by the port when the port is enabled. Note that unconditional ports are always enabled. The second argument is either a pointer or a reference to another BDFPort-Hole, which is called the associated port (the function has two overloaded forms, which is why the argument may be specified either as a pointer or as a reference). The third argument is a code specifying the relation between the porthole this method is called on and the associated port:

- **BDF_NONE** This code indicates no relation at all.
- **BDF_TRUE** This code indicates that data are transferred by the port only when the conditional port has a TRUE particle.
- **BDF_FALSE** This code indicates that data are transferred by the port only when the conditional port has a FALSE particle.
- **BDF_SAME** This code indicates that the stream transferred by the associated port is the same as the stream transferred by this port. This relationship is specified for the BDF Fork actor and aids the operation of the clustering algorithm.
- **BDF_COMPLEMENT** This code indicates that the stream transferred by the associated port is the logical complement of the stream transferred by this port. This relationship is specified for the BDF Not actor and aids the operation of the clustering algorithm.

The fourth argument for setBDFParams is the maximum delay, that is, the largest value that the star may specify as an argument to the % operator on that porthole. The default value is zero. This argument serves the same purpose as the second argument to setSDFParams.

The setSDFParams function may be used on BDF portholes; it does not alter the associated port or the relation type, but does alter the other two parameters of setBDFParams. By default, BDF portholes transfer one token, unconditionally.
Calls to setBDFParams may be placed in the setup method of a star, or alternatively in the constructor if the call does not depend on any parameters of the star. Consider as an example a Switch star. This star’s functionality is as follows: on each execution, it reads a particle from its control input port. If the value is TRUE, it reads a particle from its trueInput port; otherwise it reads a particle from its falseInput port. In any case, the particle is copied to the output port. Using the ptlang preprocessor, the setup method could be written

```plaintext
setup {
    trueInput.setBDFParams(1, control, BDF_TRUE, 0);
    falseInput.setBDFParams(1, control, BDF_FALSE, 0);
}
```

and the go method could be written

```plaintext
go {
    if (int(control%0))
        output%0 = trueInput%0;
    else
        output%0 = falseInput%0;
}
```