Christian-Albrechts-Universität zu Kiel
Institut für Informatik und Praktische Mathematik
Prof. Dr. W.-P. de Roever · Prof. Dr. R. v. Hanxleden
Dipl. Inf. K. Baukus · Dipl. Inf. J. Lukoschus

Synchrone Sprachen

Sommersemester 2002

Serie 4

10. Mai 2002

Thema : Statemate

Ausgabetermin: 10. Mai 2002

Abgabe: 17. Mai 2002

Aufgabe 1: [Doppelrechner (4 Punkte)]

Gegeben sei folgende informelle Spezifikation eines Doppelrechnersystems, geben Sie die Statechart und Activitychart für diese Spezifikation an.

The objective is to specify a fault-tolerant solution for a computation service P that can be characterized as follows:

- P inputs data provided by a *producer* on channel `A`.
- For each input x on `A`, a computation $y = f(x)$ is performed by P and delivered via channel `B` to a *consumer*.
- We assume a *synchronous communication* between server and environment: The producer will only send a new job after having received a `NEXT`-message from the server computer indicating that P has finished the previous computation.

Next, we describe the boundary conditions for the desired type of a fault-tolerant server platform: The fault-tolerant system shall be designed as a dual computer system `DCP` according to the *master-slave principle*: `DCP` consists of two computers `CP1` and `CP2`. Each of these components may fail independently. As a*fault hypothesis*, we may assume that each computer acts as a *fail-stop component*, i.e., the failure events leads to the computer's total deactivation without any remaining sub-activities. In normal operation (when both components are available), `CP1` acts as the *master*: a copy `P1` of P runs on `CP1`, producing computations after which a protocol handler of `CP1` requests a new job by sending a message `NEXT1`. `CP2` operates in *standby* mode by only storing jobs in its local memory without activation of a P-copy. Each job is kept by `CP2` at least until the `NEXT1`-message indicates that it has been successfully delivered to the consumer.

If `CP1` fails, this will also be detected by `CP2` which then continues as the master component by activating a copy `P2` of P and producing messages `NEXT2` to request new jobs. Though `CP1`'s failure may occur while a job is still being processed, it is required that this job should not be lost: `CP2` shall use its (still available) copy of the input and calculate the corresponding result. It must be taken into account, that `CP1`'s failure can occur *after* having delivered a result on channel `B` and *before* having produced the `NEXT1`-message. In such a case it cannot be avoided that `CP2` also processes this job, and the result is sent to the consumer for a second time. To this end, each input is equipped by the producer with an alternating bit, that is also attached to the result transferred to the consumer. We assume that the consumer has implemented an *alternating-bit protocol* to detect duplicated bits and discard the corresponding results.