# STiDC'08: Exercise 1

September 29, 2008

## 1 Problem

A *binary consensus* shared object has a single operation *propose* that takes a value $v$ equal to 0 or 1 as an argument and returns 0 or 1. When a process $p_i$ invokes *propose(v)*, we say that $p_i$ *proposes* value $v$. When $p_i$ is returned value $v'$ from *propose(v)*, we say that $p_i$ *decides* value $v'$ ($v'$ do not have to be equal to $v$). A binary consensus object satisfies the following properties:

**Agreement** No two processes decide different values.

**Validity** The value decided is one of the values proposed.

A *write-once register* is a shared object with the following sequential specification ($x$ is initially equal to $\perp$ and $v$ is always different than $\perp$):

```
upon write(v)
  if x = ⊥ then x := v
  return ok

upon read
  return x
```

Your tasks are:

1. To implement a binary consensus object using any number of write-once registers;

2. To implement a binary consensus object using one or more queue objects in a system of 2 processes.

**Remark:** Unless stated otherwise, we assume the following:

1. Every shared object is atomic and wait-free (so, in this exercise, the binary consensus object, write-once registers and queues are atomic and wait-free).

2. Every shared object implementation can use any number of atomic wait-free multi-valued MRMW registers (so, in this exercise, you can use atomic registers, together with write-once registers/queues, in the two binary consensus implementations).

## 2   A Solution

**Task 1.**   We use a single write-once register $r$:

```
upon propose(v)
  r.write(v)
  return r.read()
```

**Task 2.**   We use:

- a queue $q$ initialized to $\langle$ *winner*, *loser* $\rangle$,

- array of atomic registers $r[1..2]$.

Binary consensus algorithm for process $p_i$, $i = 1, 2$:

```
upon propose(v):
  r[i].write(v)
  w := q.deq()
  if w = winner then return v
  else return r[3-i].read()
```