York University EECS 6117 November 8, 2016

Homework Assignment #5
Due: November 17, 2016 at 1:00 p.m.

1. Consider an asynchronous system of n processes, where processes may experience halting failures.

We can define the `1-2-Counter` type as follows. The state of the object stores a natural number. It provides three operations: Read returns the state of the object without changing it, Inc increases the state of the object by 1 and returns ack, Inc-by-2 increases the state of the object by 2 and returns ack.

(a) Here is a proposed implementation of a `1-2-Counter` for n processes with ids $1..n$ from n shared `read/write registers`, $A[1], \ldots, A[n]$. The process with id i would execute the following code to perform an operation on the `1-2-counter`. (Here, x and v are local variables of the process performing the operation.)

```
Read
    $v \leftarrow 0$
    for $j \leftarrow 1$ to $n$
        $v \leftarrow v + \text{read}(A[j])$
    end for
    return $v$
end Read

Inc
    $x \leftarrow \text{read}(A[i])$
    write $x + 1$ into $A[i]$
end Inc

Inc-by-2
    $x \leftarrow \text{read}(A[i])$
    write $x + 2$ into $A[i]$
end Inc-by-2
```

Prove this is *not* a linearizable implementation.

(b) Show that it is possible to build a wait-free, linearizable implementation of a `1-2-counter` from registers. Make your answer as simple as possible, and prove your answer is correct.

(c) Is there a non-blocking, linearizable, *anonymous* implementation of a `1-2-counter` from registers? Prove your answer is correct. Recall that in an anonymous implementation, all processes are programmed identically and do not have ids. Hint: think about what happens when two processes trying to do the same operation run at exactly the same speed.