1 Single reader $k$-ary atomic register from binary atomic registers

We have now shown that all single writer regular and safe register can be implemented from each other. In the safe register case, the reader needs to return either the value of the most recent non-overlapping write or any values from the register. In the regular register case, the reader needs to return either the value of the most recent non-overlapping write or the values of the overlapping writes. The writes are non-overlapping because of single write write-order. For atomic registers, on top of the requirement for a regular register, there must be a linearizable sequence of the real time reads and writes such that

1. Sequential specification of the register is satisfied. The sequence is legal in the sense that readers read from most recent write.

2. The real time precedence order (an earlier operation’s response preceding a later operation’s invocation defines a two operation precedence order) must be preserved in the linearized sequence.

Another key feature is that for safe or regular registers, if there are multiple readers with overlapping writes, readers do not interfere with each other. What reader A reads does not change what reader B reads, as long as their respective relationship with the write meet safe/regular requirements. This is because there is no linearization order requirement between the readers. However, this is not the case for atomic registers. For multiple-reader when write writes into different registers sequentially, and each reader read from its own register and returns the value read, problem occur. In real time if two reads have precedence order, due to linearization requirement, the second read must be put behind the first read in linearized sequence. However, if there is an overlapping write with these two reads, it is possible that the first read reads the new value in its register, and the second read reads the old value because of the write has not cleaned the old value in its register. Then the sequential specification requires the second write be put before write and also before the first read. Contradiction. So the first read has to write, to communicate with the second read. We will discuss this in the next lecture.

Note in the read algorithm, the linearization point of read is at the last 1 that it reads. The linearization point of write is at the first point in time that registers evaluated to $v$, that is, $x_v = 1$, $x_w = 0$, for $w < v$.

We prove linearizability of operations executing the above algorithms. Our strategy is first to construct a sequence and then prove that it satisfies legality as well as preserves operations precedence for any two kinds of operations.
Algorithm 1 WRITE ($v$)

1: Write $x_v = 1$
2: For $i = v - 1$ down to 0
3: Write $x_i = 0$
4: Acknowledge

Algorithm 2 READ

1: Read $x_i$ for $i = 0$ to $k - 1$
2: until $x_i = 1$
3: $v := i$
4: Read $x_j$ for $j = v - 1$ to 0
5: if $x_j = 1$ then $v := j$
6: Return $v$.

Proof: Fix some admissible execution $\alpha$. There exist linearization points for the physical operations. So first we define a linearization of the logical operations $\pi$.

1. Order each WRITE in $\pi$ by actual order in $\alpha$, since there is only one writer, two WRITEs do not overlap and this order is well-defined.
2. Consider each READ in order they occur in $\alpha$ since there is only one reader, this order is well-defined. We say that READ $R$ reads from WRITE $W$ if $R$ returns $v$ and $W$ contains the write to $x_v$ that $R$’s last read of $x_v$ reads from. Place each READ $R$ immediately before the WRITE following the WRITE that $R$ reads from.

We need to show that $\pi$ is a linearization of $\alpha$. By construction, $\pi$ already satisfies the sequential specification, since we always place a READ after the WRITE it reads from but before the next WRITE. We need to show that the order of operations in $\alpha$ is maintained in $\pi$. Since linearization defines a total order, any real time precedence orders between any two operations must be preserved. We have the following four orders to preserve.

1. WRITE$_{<_\alpha}$WRITE $\Rightarrow$ WRITE$_{<_\pi}$WRITE
2. READ$_{<_\alpha}$WRITE $\Rightarrow$ READ$_{<_\pi}$WRITE
3. WRITE$_{<_\alpha}$READ $\Rightarrow$ WRITE$_{<_\pi}$READ
4. READ$_1_{<_\alpha}$READ$_2$ $\Rightarrow$ READ$_1_{<_\pi}$READ$_2$

The first two cases are easy, since

1. For 2 WRITEs, the order is preserved by the construction.
2. If READ $R$ precedes WRITE $W$ in $\alpha$, then clearly $R$ must read from an earlier WRITE so $R$ will be placed before $W$ in $\pi$. 

2
The other two cases are more complicated, and we need a lemma. Intuitively this lemma says if you, a reader, find a writer’s fresh footprint going downstairs when you are going upstairs, you will get what this writer wrote at an upper floor just now and nothing else from other writers. This lemma only involves real time reality. It does not involve linearization. During the proof of this lemma, a useful technique employed is to create an infinitely increasing sequence of integers to contradict the fact that there are only limited integers between two integer numbers.

**Lemma 1** Consider 2 values $u$ and $v$, where $u < v$. If $R$ returns $v$ and $R$’s read of $x_u$ during the upward scan reads from a write contained in $W_1$, then $R$ does not read from any WRITE that precedes $W_1$.

**Proof:** Suppose in contradiction that $R$ reads from a WRITE $W$ that precedes $W_1$.

![Diagram](image.png)

Figure 1: Proof of Lemma 1: an infinitely increasing sequence to mop out a previous 1. This figure shows an impossible scenario for contradiction and cannot happen in reality.

Let $v_1$ be the value written by $W_1$, since $W_1$ writes $x_u = 0$, it follows that $u < v_1$. Also $v_1 < v$, since otherwise $W_1$ would overwrite $W$’s write to $x_v$ before $R$ could read $x_v$, a contradiction, since $R$ reads $x_v = 1$.

Thus, in $R$’s upward scan, $R$ reads $x_u = 0$, then $x_{v_1} = 0$ and then $x_v = 1$. So, there must be another WRITE $W_2$ that writes $x_{v_1} = 0$ before $R$ reads $x_{v_1}$. So $W_2$ writes $v_2$, where $v_2 > v_1$ since $W_2$ writes $x_{v_1} = 0$, and $v_2 < v$, for the same reason as $v_1 < v$. Continuing this argument, there must be a WRITE $W_3$ that writes $x_{v_2} = 0$, since $R$ reads $x_{v_2} = 0$, where $v_2 < v_3 < v$, and so on. Thus, there exists an infinite increasing sequence of integers $v_1, v_2, v_3, \ldots$, all of which are less than $v$, a contradiction.

**CASE 3.** WRITE $W$ precedes READ $R$ in $\alpha$.

Suppose for contradiction, that $R$ is placed before $W$ in $\pi$. Then, $R$ reads from $W'$ that precedes $W$. Let WRITE $W$ write $v$ and $W'$ write $v'$. So $R$ reads $v'$.
Consider the case $v' \leq v$. Then $W$ writes $x_{v'} = 0$ after $W'$ writes $x_{v'} = 1$, so $R$ cannot read $x_{v'} = 1$, and return $v'$. Otherwise, if $v' = v$, then $R$ reads $W$’s write $x_{v'} = 1$.

Consider the case $v' > v$. $W$ writes $x_v = 1$. Since $R$ does not read $x_v = 1$ during upward scan, there must be a WRITE $W''$ after $W$ that writes $x_v = 0$ that $R$ reads $x_v = 0$. Now, by Lemma 1, $R$ cannot read from a WRITE that precedes $W''$, so it cannot read from $W'$. Contradiction.  

Note in Case 3, there is no need for read to do a downward scan.

**CASE 4.** READ $R_1$ precedes $R_2$ in $\alpha$.

Suppose for contradiction, that READ $R_1$ follows $R_2$ in $\pi$. This implies that $R_1$ reads from $W_1(v_1)$ and $R_2$ reads from $W_2(v_2)$, where $W_1$ follows $W_2$.

$v_1 = v_2$. $W_2$ writes $x_{v_1} = 1$ followed by $W_1$ writing $x_{v_1} = 1$, followed by $R_1$’s last read of $x_{v_1}$. Since this is followed by $R_2$’s last read of $x_{v_1}$, $R_2$ cannot read the value written by $W_2$, as $W_1$ has overwritten it. Contradiction.

*Note in Case 4 $v_1 = v_2$, there is no need for read to do a downward scan.*

$v_1 > v_2$. Since $R_2$ reads from $W_2$, no write to $x_{v_2}$ is linearized between $W_2$’s write $x_{v_2} = 1$ and $R_2$’s read $x_{v_2} = 1$. Since $R_1$ reads from $W_1$, $W_1$’s write $x_{v_1} = 1$ precedes $R_1$’s read $x_{v_1} = 1$. So $x_{v_2} = 1$ starting before $R_1$ reads $x_{v_1} = 1$ and ending after $R_2$ reads $x_{v_2} = 1$. But, then $R_1$’s read $x_{v_2}$ during its downward scan would return 1, not 0, a contradiction. Figure 2 shows this impossible scenario.

A special note here is that in Case 4 $v_1 > v_2$, if there is no downward scan, the scenario shown in Figure 2 is possible! Therefore, the two reads are not linearizable without downward scan.

$v_1 < v_2$. Since $R_1$ reads from $W_1$, $W_1$’s write of $x_{v_1} = 1$ precedes $R_1$’s read $x_{v_1} = 1$. Since $R_2$ returns $v_2 > v_1$, $R_2$’s first read of $x_{v_1} = 0$. So there must be a later WRITE after $W_1$ containing
Figure 3: Case 4 when $v_1 < v_2$. This figure shows an impossible scenario for contradiction and cannot happen in reality.

a write $x_{v_1} = 0$ that $R_2$ reads from. Now by Lemma 1, $R_2$ cannot read from an earlier WRITE, so cannot read from $W_1$, a contradiction. Figure 3 shows this impossible scenario.

Note in Case 4 $v_1 < v_2$, there is no need for read to do a downward scan.

In summary, the downward scan by logical atomic register is necessary to achieve linearizability.

**Theorem 2** There exists a wait-free simulation of a $k$-valued register using $k$ binary registers in which each logical operations perform $O(k)$ physical operations.