Proof of the $\Omega(n)$ lower bound on the number of registers for mutual exclusion:

**Theorem 1** Any algorithm using read/write registers (even multi-writer) that satisfies mutual exclusion and no-deadlock must use $\Omega(n)$ registers.

**indistinguishability** Configurations $C, D$ are indistinguishable to a process $p$ if $p$ has the same local state in both configurations and $\text{mem}(C) = \text{mem}(D)$, denoted $C \lessdot D$. A similar definition is used for a set of processes $P$.

For every finite schedule $\sigma$, we denote by $\sigma(C)$ the configuration reached by $\text{exec}(C, \sigma)$.

**Lemma 2** If $C \lessdot P D$ for a set of processes $P$ and $\alpha$ is a $P$-only schedule, then $\sigma(C) \lessdot P \sigma(D)$.

**quiescent configuration** A configuration $D$ is quiescent if all the processes are in the remainder.

**P-quiescent configuration** A configuration $C$ is P-quiescent for a set of processes $P$ if there is a quiescent configuration $D$ such that $C \lessdot P D$.

**Covers** A process $p$ covers a variable $R$ in configuration $C$ if the next step of $p$ is writing to $R$. This is a characteristic of the local state of $p$.

The intuition is that if $p$ covers $R$ in $C$, then after any finite $p$-free schedule from $C$ $p$ still covers $R$.

**Lemma 3** Let $C$ be a $p_i$-quiescent configuration. Then there is a $p_i$-only schedule $\sigma$ such that $p_i$ is in the critical section in $\sigma(C)$, and during $\text{exec}(C, \sigma)$ $p_i$ writes to a variable that is not covered in $C$ by any other process.

**Proof:** Since $C$ is $p_i$-quiescent there is a quiescent configuration $D$ which is indistinguishable from $C$ to $p_i$. If $p_i$ runs alone from $D$ then the no-deadlock condition implies that it will eventually enter the critical section. Let $\sigma$ be that $p_i$-only finite schedule for which $p_i$ is in the critical section in $\sigma(D)$, then by Lemma 1, $p_i$ is in the critical section also in $\sigma(C)$. Assume that any variable to which $p_i$ writes in $\text{exec}(C, \sigma)$ is covered by another process in $C$. Let $W$ be the set of variables covered in $C$ not by $p_i$, and let $P$ be the set of processes that covers them. From $C$, we let every process in $P$ take one step, so that all variables in $W$ are over-written. We then run all the processes that are not in the remainder until they reach the remainder (by no-deadlock) in a quiescent configuration $Q$. We then run some $p_j$ other than $p_i$ until it is in the critical section in a configuration $E$. Running this same schedule from $\sigma(C)$ instead of from $C$ results in $p_j$ being in the critical section as in $E$, since the first steps have over-written anything that $p_i$ wrote in $\text{exec}(C, \sigma)$. But $p_i$ is also in the critical section, which is a contradiction. $\blacksquare$
Lemma 4  For every \( k, 1 \leq k \leq n \), and a quiescent configuration \( C \), there is a configuration \( D \) reachable from \( C \) by a \( p_0, \ldots, p_{k-1} \)-only schedule, such that \( p_0, \ldots, p_{k-1} \) cover \( k \) different variables in \( D \), and \( D \) is \( p_k, \ldots, p(n-1) \)-quiescent.

Proof:  By induction on \( k \). Induction step: From \( C \) we can reach a configuration \( C_1 \) which is \( p_k, \ldots, p_{n-1} \)-quiescent and in which \( p_0, \ldots, p_{k-1} \) cover a set \( W \) of \( k \) different variables. By Lemma 3 there is a \( p_k \)-only schedule after which \( p_k \) covers a variable \( X \) not in \( W \) for the first time. But this might not be a \( p_{k+1}, \ldots, p_{n-1} \)-quiescent configuration because \( p_k \) might have written to other variables during this execution. We do something similar to the previous proof and let \( p_0, \ldots, p_{k-1} \) overwrite \( W \) and run to the remainder. This configuration \( D'_1 \) is not quiescent since \( p_k \) is in the entry, but if we had run this from \( C_1 \) it would have been a quiescent configuration \( D_1 \), so we could have used the induction hypothesis again to get these processes covering \( W \) again in a configuration \( C_2 \). But only \( p_k \) distinguishes between \( D_1 \) and \( D'_1 \) and hence running this from \( D'_1 \) results in a configuration \( C'_2 \) in which they cover \( W \), and also \( p_k \) covers \( X \) not in \( W \).

We cheated: the lemma does not guarantee that the set \( W \) of covered variables is always the same. So in \( C'_2 \) we might be covering a set \( W' \) which does include \( X \). But since there is only a finite number of variables, there is only a finite number of sets of \( k \) variables. We go from \( C \) to \( C_1 \) to \( D_1 \) to \( C_2 \) to \( D_2 \) and so on, each time covering a set \( W_i \) in \( C_i \). After a finite number of iterations, we get \( W_i = W_j \) and now our previous argument works.

Proof of the Theorem:  Apply Lemma 4 to the initial configuration \( C \) with \( k=n \).