ONLINE APPENDIX

A. TABLE OF REWRITES

Annotation / method	**Rewrite**
`\Wt`	`m.Wt`
`\Ot`	`m.Ot`
`\lock(`*m*`)`	Rewrite in parent expression
`\cond(`*m*`)`	Rewrite in parent expression
`\wait_level(\lock(`*m*`))`	`m.wait_level_lock`
`\wait_level(\cond(`*m*`))`	`m.wait_level_cond`
`\has_ob(`*z*`)`	Check if there is at least one obligation for `z` in `obs`
`\has_obs(`*z*`, `*n*`)`	Check if there are at least n obligation for `z` in `obs`
`\no_obs`	Check if `obs` is empty
`charge_ob` *m*	• Add one obligation for the condition variable of m to `obs` • Increment `m.Ot`
`charge_obs` *m*`, `*n*	• Add n obligations for the condition variable of m to `obs` • Add n to `m.Ot`
`discharge_ob` *m*	• Assert `m.Wt == 0 || m.Ot - 1 > 0` • Remove one obligation for the condition variable of m from `obs` • Decrement `m.Ot`
`discharge_obs` *m*`, `*n*	• Assert `m.Wt == 0 || m.Ot - n > 0` • Remove n obligations for the condition variable of m from `obs` • Subtract n from `m.Ot`
`transfer_ob` *m*`, `*t*	• Assert that the current thread has at least one obligation for the condition variable of m • Copy the obligation to the thread with identifier t • Remove the obligation from `obs`
`transfer_obs` *m*`, `*n*`, `*t*	• Assert that the current thread has at least n obligation for the condition variable of m • Copy the obligations to the thread with identifier t • Remove the obligations from `obs`
`set_wait_level(\lock(`*m*`), `*r*`)`	Assign r to `m.wait_level_lock`
`set_wait_level(\cond(`*m*`), `*r*`)`	Assign r to `m.wait_level_cond`
Synchronized block or method	Prepend to body: • Check if the lock of m has the lowest wait level among the wait levels of `obs` – If not, assert that there is already an obligation for the lock of m in `obs` and that it has the lowest wait level among the wait levels of `obs` • Add one obligation for the lock of m to `obs` • Give full permissions for `m.Wt` and `m.Ot` to the current thread Append to body: • Remove one obligation for the lock of m from `obs` • Revoke the permissions for $m.Wt$ and $m.Ot$

`m.wait()`	• Assert that there is an obligation for the lock of m in `obs` • Assert that the condition variable of m has the lowest wait level among the wait levels of `obs` • Assert that the lock of m has the lowest wait level among the wait levels of `obs` • Assert `m.Ot` is strictly positive • Increment `m.Wt`
`m.notify()`	Decrement `m.Wt`
`m.notifyAll()`	Assign `0` to `m.Wt`