# WaterLevelTask

**Concurrency with:** WashingProgramTask

**Description:** When WashingProgramTask updates goalWaterLevel and WaterLevelTask tries to read when this is being updated, this might result in WaterLevelTask reading a non-valid value.

**Solution:** Pool with Mutex. WaterLevelTask will only read from the pool when Mutex allows him to. This way WaterLevelTask will not read while WashingProgramTask tries to write.

**Other:** WaterLevelTask has a clock for polling the UART to update the currentWaterLevel. This happens every 500 ms.

# HeatingTask

**Concurrency with:** WashingProgramTask

**Description:** When WashingProgramTask updates goalTemperature and HeatingTask tries to read when this is being updated, this might result in HeatingTask reading a non-valid value.

**Solution:** Pool with Mutex. HeatingTask will only read from the pool when Mutex allows him to. This way HeatingTask will not read while WashingProgramTask tries to write.

**Other:** HeatingTask has a clock for polling the UART to update the currentTemperature. This happens every 500 ms.

# MotorTask

**Concurrency with:** WashingProgramTask

**Description:** When WashingProgramTask updates goalRPM and MotorTask tries to read when this is being updated, this might result in MotorTask reading a non-valid value.

**Solution:** Pool with Mutex. MotorTask will only read from the pool when Mutex allows him to. This way MotorTask will not read while WashingProgramTask tries to write.

**Other:** MotorTask has a clock for polling the UART to update the currentRPM. This happens every 500 ms.

# WebInterfaceTask

**Concurrency with:** WashingProgramTask

**Description:** WashingProgramTask and WebInterfaceTask both write and read values to/from each other. Values could get corrupted if a task tries to read while the other is writing, and since there are multiple values being written/read some could be lost without a buffer.

**Solution:** Two channels for both ways of reading and writing. This provides a buffer so no values get lost, and allows for synchronised reading and writing to both sides without chance on non-valid data.

**Other:** WebInterfaceTask has a timer for polling the WebSocket forupdating it’s values. This happens every 1000 ms.

# UARTInterfaceTask

**Concurrency with:** WaterLevelTask, HeatingTask, MotorTask

**Description:** When WaterLevelTask, HeatingTask and MotorTask try to read currentValue while it is being updated, it might result in them reading a non-valid value.

**Solution:** Pool. The concurrency tasks can keep reading from the pool without any issue, and when UARTInterfaceTask has updated it’s currentValue it will write this to the pool at the request of the concurrency task.