```cpp
std::atomic<float> gain (1.0f);
static_assert (std::atomic<float>::is_always_lock_free);

void processSensorData (float* sensorInOut, int n)
{
    // do some dsp
    ...

    for (int i = 0; i < n; ++i)
        sensorInOut[i] *= gain.load();
}

// called on another thread
void setSensorGain (float newGain)
{
    gain.store (newGain);
}
```

Ensure manipulating a float is lock-free on your machine

Ensures loads and stores are synchronised

std::atomic<>

- Ensure “tear free” & synchronised manipulation of shared data
- May use traditional locks if data-type cannot be manipulated atomically in hardware. Always check **`std::atomic<>::is_always_lock_free`**!
- Only a subset of manipulations are supported:
  - Store
  - Load
  - Atomic addition/subtraction
  - exchange/compare-exchange
- More info here: https://herbsutter.com/2013/02/11/atomic-weapons-the-c-memory-model-and-modern-hardware/