# The CAS Exchange Loop

```cpp
struct BiquadCoeffecients  {  float b0, b1, b2, a1, a2; };
std::atomic<BiquadCoeffecients*> coeffs;

BiquadCoeffecients calculateLowPassCoeffecients (float freq);

void audioThread (const float* src, float* dst, size_t n)
{
    auto* coeffsCopy = coeffs.load();
    processBiquad (src, dst, n, coeffsCopy);
}

void updateFrequencyParameter (float newValue)
{
    coeffs = new BiquadCoeffecients (calculateLowPassCoeffecients (newValue));
}
```

## ✗ Works but memory leak!

67

# The CAS Exchange Loop

```
struct BiquadCoeffecients  {  float b0, b1, b2, a1, a2; };
std::atomic<BiquadCoeffecients*> coeffs;

BiquadCoeffecients calculateLowPassCoeffecients (float freq);

void audioThread (const float* src, float* dst, size_t n)
{
    auto* coeffsCopy = coeffs.load();
    processBiquad (src, dst, n, coeffsCopy);
}

void updateFrequencyParameter (float newValue)
{
    coeffs = new BiquadCoeffecients (calculateLowPassCoeffecients (newValue));
}
```

✗ **Works but memory leak!**

# The CAS Exchange Loop

```
struct BiquadCoeffecients  {  float b0, b1, b2, a1, a2; };
BiquadCoeffecients* coeffs;
std::atomic<bool> isInAudioThread { false };

BiquadCoeffecients calculateLowPassCoeffecients (float freq);

void audioThread (const float* src, float* dst, size_t n)
{
    isInAudioThread = true;
    auto* coeffsCopy = coeffs;
    processBiquad (src, dst, n, coeffsCopy);
    isInAudioThread = false;
}

void updateFrequencyParameter (float newValue)
{
    auto* ptr = new BiquadCoeffecients (calculateLowPassCoeffecients (newValue));

    while (isInAudioThread.load())
        ;

    std::swap (ptr, coeffs);
    delete ptr;
}
```