

glm::detail::compute\_half< 1, Q >::unpack

glm::detail::compute\_half< 2, Q >::unpack

glm::detail::compute\_half< 3, Q >::unpack

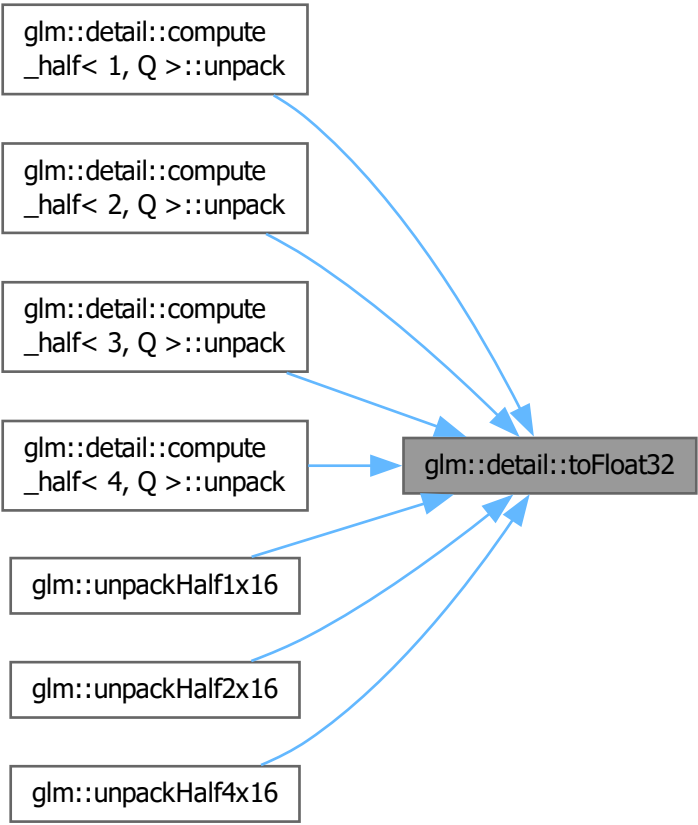
glm::detail::compute\_half< 4, Q >::unpack

glm::unpackHalf1x16

glm::unpackHalf2x16

glm::unpackHalf4x16

glm::detail::toFloat32



```
graph LR; A[glm::detail::compute_half< 1, Q >::unpack] --> F[glm::detail::toFloat32]; B[glm::detail::compute_half< 2, Q >::unpack] --> F; C[glm::detail::compute_half< 3, Q >::unpack] --> F; D[glm::detail::compute_half< 4, Q >::unpack] --> F; E[glm::unpackHalf1x16] --> F; G[glm::unpackHalf2x16] --> F; H[glm::unpackHalf4x16] --> F;
```

The diagram illustrates the internal structure of the glm::detail::toFloat32 function. It is a central grey box that receives input from seven other white boxes. The inputs are categorized into two groups: compute\_half functions (which take a half-precision value and a quantization level Q) and unpackHalf functions (which take a half-precision vector of a specific size). The arrows indicate that all these functions ultimately contribute to the toFloat32 conversion process.