```
template<int DG> using DGVec = Eigen::Matrix<T, Eigen::Dynamic, DG>;

template <int CG, int DGstress, int DGadvection>
void StressUpdateHighOrder(const VPParameters& vpparameters,
    const ParametricMomentumMap<CG>& pmap, const ParametricMesh& smesh,
    DGVec<DGstress>& S11, DGVec<DGstress>& S12, DGVec<DGstress>& S22,
    const DGVec<DGstress>& E11, const DGVec<DGstress>& E12, const DGVec<DGstress>& E22,
    const DGVec<DGadvection>& H, const DGVec<DGadvection>& A, double alpha, double beta)
{
    constexpr int NGP = ((DGstress == 8) || (DGstress == 6)) ? 3 : (DGstress == 3 ? 2 : -1);
    using EdgeVec = Eigen::Matrix<T, 1, NGP * NGP>;
#pragma omp parallel for
    for (size_t i = 0; i < smesh.nelements; ++i) {
        auto hGauss = (H.row(i) * PSI<DGadvection, NGP>).array().max(0.0).matrix();
        auto aGauss = (A.row(i) * PSI<DGadvection, NGP>).array().max(0.0).min(1.0).matrix();
        EdgeVec P = (_vpparameters.Pstar * hGauss.array()
            * (-20.0 * (1.0 - aGauss.array()))).exp()).matrix();

        const EdgeVec e11Gauss = E11.row(i) * PSI<DGstress, NGP>;
        const EdgeVec e12Gauss = E12.row(i) * PSI<DGstress, NGP>;
        const EdgeVec e22Gauss = E22.row(i) * PSI<DGstress, NGP>;
        const auto DELTA = (vpparameters.DeltaMin * vpparameters.DeltaMin
            + 1.25 * (e11Gauss.array().square() + e22Gauss.array().square())
            + 1.50 * e11Gauss.array() * e22Gauss.array() + e12Gauss.array().square())
                    .sqrt().matrix();

        const T alphaInv = 1.0 / alpha;
        const T fac = 1.0 - alphaInv;
        const EdgeVec PDelta = P.array() / DELTA.array();
        S11.row(i) = fac * S11.row(i) + (pmap.iMJwPSI[i]
            * (alphaInv * (PDelta.array()
                * ((5.0 / 8.0) * e11Gauss.array() + (3.0 / 8.0) * e22Gauss.array())
                - 0.5 * P.array()).matrix().transpose())).transpose();
        S12.row(i) = fac * S12.row(i) + (pmap.iMJwPSI[i]
            * (alphaInv * (PDelta.array() * (1.0 / 4.0) * e12Gauss.array())
                .matrix().transpose())).transpose();
        S22.row(i) = fac * S22.row(i) + (pmap.iMJwPSI[i]
            * (alphaInv * (PDelta.array()
                * ((5.0 / 8.0) * e22Gauss.array() + (3.0 / 8.0) * e11Gauss.array())
                    - 0.5 * P.array()).matrix().transpose())).transpose();
    }
}

```