```fortran
!======================================================================!
!                                                                      !
! Software Name : FrontISTR Ver. 3.4                                   !
!                                                                      !
!      Module Name : Static Analysis                                   !
!                                                                      !
!            Written by K. Sato (Advancesoft)                          !
!                       X. Yuan (Advancesoft)                          !
!                                                                      !
!      Contact address :  IIS,The University of Tokyo, CISS           !
!                                                                      !
!      "Structural Analysis for Large Scale Assembly"                  !
!                                                                      !
!======================================================================!
!C
!C***
!> CONSTRUCT the GLOBAL STIFF MATRIX
!C***
!C
module m_static_mat_ass_main
   implicit none

   contains

   subroutine FSTR_MAT_ASS_MAIN (hecMESH, hecMAT, fstrSOLID)
      use m_fstr
      type (hecmwST_matrix)     :: hecMAT
      type (hecmwST_local_mesh) :: hecMESH
      type (fstr_solid)         :: fstrSOLID

!** Local variables
      real(kind=kreal) :: xx(20), yy(20), zz(20), stiffness(20*6, 20*6)
      integer(kind=kint) :: nodLOCAL(20)
      integer(kind=kint) :: ndof, itype, iS, iE, ic_type, nn, icel, iiS, j
!C
!C +-------+
!C | INIT. |
!C +-------+
!C===
      call hecmw_mat_clear(hecMAT)
      hecMAT%X = 0.d0
!C
!C +-------------------------------+
!C | ELEMENT-by-ELEMENT ASSEMBLING |
!C | according to ELEMENT TYPE     |
!C +--------------------------------+
      ndof = hecMAT%NDOF

      do itype= 1, hecMESH%n_elem_type
        iS= hecMESH%elem_type_index(itype-1) + 1
        iE= hecMESH%elem_type_index(itype  )
        ic_type= hecMESH%elem_type_item(itype)
!C** Ignore link elements
        if (hecmw_is_etype_link(ic_type)) cycle
!C** Set number of nodes
        nn = hecmw_get_max_node(ic_type)
!C element loop
        do icel= iS, iE
!C** node ID
          iiS= hecMESH%elem_node_index(icel-1)
          do j=1,nn
            nodLOCAL(j)= hecMESH%elem_node_item (iiS+j)
!C** nodal coordinate
            xx(j)=hecMESH%node(3*nodLOCAL(j)-2)
            yy(j)=hecMESH%node(3*nodLOCAL(j)-1)
            zz(j)=hecMESH%node(3*nodLOCAL(j))
          enddo
!C** Create local stiffness
          call fstr_local_stf_create(hecMESH, ndof, ic_type, icel, xx, yy, zz,
fstrSOLID%elements(icel)%gausses, &
                                     fstrSOLID%elements(icel)%iset, stiffness)
!== CONSTRUCT the GLOBAL MATRIX STARTED
          call hecmw_mat_ass_elem(hecMAT, nn, nodLOCAL, stiffness)
        enddo
      enddo

!* for EQUATION
!      call hecmw_mat_ass_equation ( hecMESH, hecMAT )

   end subroutine FSTR_MAT_ASS_MAIN


   !> Calculate stiff matrix of current element
   subroutine FSTR_LOCAL_STF_CREATE(hecMESH, ndof, ic_type, icel, xx, yy, zz, gausses, iset,
stiffness)
      use m_fstr
      use m_static_lib
      use mMechGauss

      type (hecmwST_local_mesh) :: hecMESH
      integer(kind=kint) :: ndof, ic_type, icel, iset
      real(kind=kreal) :: xx(:), yy(:), zz(:), stiffness(:, :)
      type( tGaussStatus ), intent(in) :: gausses(:)
      real(kind=kreal) :: ee, pp, thick,ecoord(3,20), coords(3,3)
      type( tMaterial ), pointer :: material

!** Local variables
      real(kind=kreal) :: local_stf(1830)
      integer(kind=kint) :: nn, isect, ihead


      nn = hecmw_get_max_node(ic_type)
      ecoord(1,1:nn) = xx(1:nn)
      ecoord(2,1:nn) = yy(1:nn)
      ecoord(3,1:nn) = zz(1:nn)
      material => gausses(1)%pMaterial
      ee = material%variables(M_YOUNGS)
      pp = material%variables(M_POISSON)
      if ( ic_type==241 .or. ic_type==242 .or.    &
           ic_type==231 .or. ic_type==232 .or. ic_type==2322 ) then
        thick =1.d0
        call
STF_C2( ic_type,nn,ecoord(1:2,1:nn),gausses(:),thick,stiffness(1:nn*ndof,1:nn*ndof),iset
)

      else if ( ic_type==301 ) then
        isect= hecMESH%section_ID(icel)
        ihead = hecMESH%section%sect_R_index(isect-1)
        thick = hecMESH%section%sect_R_item(ihead+1)
        call
STF_C1( ic_type,nn,ecoord(:,1:nn),thick,gausses(:),stiffness(1:nn*ndof,1:nn*ndof) )

      else if (ic_type==361) then
        call
STF_C3D8IC( ic_type,nn,ecoord(:,1:nn),gausses(:),stiffness(1:nn*ndof,1:nn*ndof))
      else if (ic_type==341 .or. ic_type==351 .or. ic_type==361 .or.     &
               ic_type==342 .or. ic_type==352 .or. ic_type==362 ) then
        call
STF_C3( ic_type,nn,ecoord(:,1:nn),gausses(:),stiffness(1:nn*ndof,1:nn*ndof),1.d0, coords)

      else if( ( ic_type==741 ) .or. ( ic_type==743 ) .or. ( ic_type==731 ) ) then
        isect= hecMESH%section_ID(icel)
        ihead = hecMESH%section%sect_R_index(isect-1)
        thick = hecMESH%section%sect_R_item(ihead+1)
        call STF_Shell_MITC(ic_type, nn, ndof, ecoord(1:3, 1:nn), gausses(:),
stiffness(1:nn*ndof, 1:nn*ndof), thick)

      else if ( ic_type==611) then
        isect= hecMESH%section_ID(icel)
        ihead = hecMESH%section%sect_R_index(isect-1)
        call STF_Beam(ic_type,nn,ecoord,hecMESH%section%sect_R_item(ihead+1:),ee,
pp,stiffness(1:nn*ndof,1:nn*ndof))

      else
        write(*,*) '###ERROR### : Element type not supported for linear static analysis'
        write(*,*) ' ic_type = ', ic_type
        call hecmw_abort(hecmw_comm_get_comm())
      endif

   end subroutine FSTR_LOCAL_STF_CREATE

end module m_static_mat_ass_main
```