```
blade 1 (demand pitch angle), rad.
REAL(ReKi), PARAMETER           :: R2D = 57.295780  ! Factor to convert radians to
degrees.

LOGICAL, SAVE                   :: Initialize1 = .TRUE. !Flag used to initialize some
saved variables on the first call to this subroutine
LOGICAL, SAVE                   :: Initialize2 = .TRUE. !Flag used to initialize some
saved variables on the first call to this subroutine

INTEGER(4)       :: TqCount = 1   ! Counter to see how many time subroutine is called

!=======================================================================
   ! Initialize saved variables on first call to subroutine
    IF ( Initialize1 ) THEN
       WRITE(*,*)  'Running with torque control programmed by Eric Anderson '// &
             'in subroutine UserVSCont(), which can be found in UserSubs.f90 '
    Initialize1 = .FALSE.
        ! NOTE: LastGenTrq, though SAVEd, is initialized below for simplicity, not
here.
        LastTimeVS = ZTime - VS_DT    ! This will ensure that the torque controller is
called on the first pass
    ENDIF
!=======================================================================
    ! Variable-speed torque control:

   ! Compute the elapsed time since the last call to the controller:
   ElapTime = ZTime - LastTimeVS


   ! Only perform the control calculations if the elapsed time is greater than
   !   or equal to the communication interval of the torque controller:
   ! NOTE: Time is scaled by OnePlusEps to ensure that the contoller is called
   !       at every time step when VS_DT = DT, even in the presence of
   !       numerical precision errors.

IF ( ( ZTime*OnePlusEps - LastTimeVS ) >= VS_DT )  THEN

    ! Get up to date control parameters
    CALL updateControlParameters( HSS_Spd, ZTime )

    IF ( EmergencyShutdown ) THEN
        GenTrq = 0
    ELSE

       ! Determine some torque control parameters not specified directly:
        VS_RtGnSp = 0.99*PC_RefSpd
        VS_SySp    = VS_RtGnSp/( 1.0 +  0.01*VS_SlPc )
        VS_Slope15 = ( VS_Rgn2_K*VS_Rgn2Sp*VS_Rgn2Sp )/( VS_Rgn2Sp - VS_CtInSp )
        VS_Slope25 = ( VS_RtPwr/VS_RtGnSp           )/( VS_RtGnSp - VS_SySp   )
        VS_TrGnSp = ( VS_Slope25 - SQRT( VS_Slope25*( VS_Slope25 - &
                     4.0*VS_Rgn2_K*VS_SySp ) ) )/( 2.0*VS_Rgn2_K ) !Transition speed
from region 2 to region 2.5

        BlPitchCom =  AllOuts(PtchPMzc1)/R2D
       ! Compute the generator torque, which depends on which region we are in:

          IF ( (   GenSpeedF >= VS_RtGnSp ) .OR. (  BlPitchCom >= (VS_Rgn3MP + &
                              PC_MinPit ) ) ) THEN ! We are in region 3 - power is
constant
             GenTrq = VS_RtPwr/GenSpeedF
          ELSEIF ( GenSpeedF <= VS_CtInSp )  THEN   ! We are in region 1 - torque is
```