//This program monitors X and Y position commands continuously at 10 KHz using

//the StatusGetAxisItemFast command to collect arrays of trajectory points. The X and Y

//position commands are then used to calculate the corresponding desired command

//position of a Z axis, based off of a spherical surface relationship. Because the

//collection of X and Y position trajectory and subsequent spherical calculation

//imparts a delay on the execution of output Z-axis command, the

//TrajectoryDelayTime parameter for the X & Y axes must be adjusted to realign

//the phasing of X, Y, and Z axis motions to a synchronous state.

program

 var $index //Define an index tracking variable

 var $Xposition[20] //Initiate an 10 element array to store X Position commands

 var $Yposition[20] //Initiate an 10 element array to store Y Position commands

 var $OffAxisDistance[20] //Variable for current distance from hemisphere's axis

 var $Zposcommand[20] //Variable to store calculated Z axis position command

 #define Radius 100 //Define the hemisphere radius as 100 mm

 Enable(Z)

 Home(Z)

 G90 //Set the move targets to absolute mode

 CriticalSectionStart() //Execute following code in 1 [ms], to CriticalSectionEnd command

 while (1) //Perform following code endlessly in a loop, to end command

 //Get a sample at the SampleRate of 20 kHz

 StatusGetAxisItemFast(X, AxisDataSignal.PositionCommandRawUnfiltered, 0, 20, $Xposition)

 StatusGetAxisItemFast(Y, AxisDataSignal.PositionCommandRawUnfiltered, 0, 20, $Yposition)

 for $index = 0 to 19

 $OffAxisDistance[$index] = Sqrt($Xposition[$index] \*\* 2 + $Yposition[$index] \*\* 2)

 $Zposcommand[$index] = Radius - Sqrt(Radius \*\* 2 - $OffAxisDistance[$index] \*\* 2)

 MovePt([Z], [-$Zposcommand[$index]], 1.0/20) //Time base must match SampleRate, 20 kHz

 end

 end

 CriticalSectionEnd()

end