```
  PROGRAM BLOCH
  IMPLICIT NONE

  INTEGER, PARAMETER :: Prec14=SELECTED_REAL_KIND(14)
  INTEGER :: i,j,n,isign,ik,jmi,ierr,matz,jj
  INTEGER, PARAMETER :: jmax=64,ikmax=20
  COMPLEX(KIND=Prec14), DIMENSION(0:jmax-1) :: v ! potential and Fourier tra
  COMPLEX(KIND=Prec14), DIMENSION(0:jmax,0:jmax) :: h ! Hamiltonian matrix
  REAL(KIND=Prec14) :: psir,psii,prob ! real, imaginary components of psi, probab
  REAL(KIND=Prec14), DIMENSION(0:jmax,0:jmax) :: hr,hi,zr,zi ! arrays for diago
  REAL(KIND=Prec14), DIMENSION(0:jmax) :: w ! eigenvalue array
  REAL(KIND=Prec14), DIMENSION(0:jmax) :: fv1,fv2 ! work arrays
  REAL(KIND=Prec14), DIMENSION(2,0:jmax) :: fm1 ! work array
  COMPLEX(KIND=Prec14) :: ar,ai,probc,p1,p2,psi
  REAL(KIND=Prec14), PARAMETER :: L=4.0d0,sigma=4.0d0
  REAL(KIND=Prec14), PARAMETER :: x0=L/2.0d0,v0=-10.0d0
  REAL(KIND=Prec14), PARAMETER :: dx=L/jmax
  REAL(KIND=Prec14) :: x,pi,k,dk,G,dG
  open(unit=50,  file="energy.dat")
  open(unit=51, file='prob.dat')
  pi=4.0d0*datan(1.0d0)
  ai=(0.0d0,1.0d0)
  ar=(1.0d0,0.0d0)
  ! Set the periodic potential over the region 0<x<L
  ! Initialize the system.
  do j=0,jmax-1
    x=dx*j
    v(j)=ar*v0*dexp(-(x-x0)**2/(2.0d0*sigma**2))
  enddo
  ! Perform a fourier transform
  isign=-1
  call four1(v,jmax,isign)
  v=v/jmax

! Populate off-diagonal elements
  do i=0,jmax-1
    do j=i+1,jmax-1
      jmi=j-i
      h(i,j)=v(jmi)
      h(j,i)=dconjg(v(jmi)) ! Hermitian matrix
    enddo
  enddo

  dk=pi/(L*ikmax)
  dG=2.0d0*pi/L
    do ik=-ikmax,ikmax
      k=dk*ik
      ! Populate H matrix diagonal, ignore potential term along diagonal
      do i=0,jmax ! Slight change from before!
        ! Diagonal element
        G=-pi/dx+dG*i
        h(i,i)=0.5d0*(G+k)**2
      enddo
      ! adds a real / imaginary to hr and hi respectively
      do i=0,jmax
        do j=0,jmax
          hr(i,j)=dreal(h(i,j))
          hi(i,j)=dimag(h(i,j))
        enddo
      enddo
      ! Diagonalize H-matrix for this k-point
      matz=1 ! get eigenvectors
      call ch(jmax+1,jmax+1,hr,hi,w,matz,zr,zi,fv1,fv2,fm1,ierr)
      ! Writes the lowest 5 energy levels to file
      do j=0,4 ! Lowest 5 eigenvalues output
        write(50,*) k,w(j)
      enddo
      ! Evaluates initial probabilities.
      if(ik.eq.0) then
        do i=0,4 ! loop over lowest 5 eigenstates
          do j=0,jmax-1 ! loop over x -points
            x=dx*j
            psir=0.0d0
            psii=0.0d0
            do jj=0,jmax ! loop over all G-vectors (all basis states)
              G=-pi/dx+dG*jj
              psir=psir+zr(jj,i)*dcos(G*x)-zi(jj,i)*dsin(G*x) ! Real part of psi
              psii=psii+zr(jj,i)*dsin(G*x)+zi(jj,i)*dcos(G*x) ! Imag part of psi
            enddo
            prob=psir**2+psii**2 ! Probability density at current point x, for state i
            ! Writes probability to file.
            write(51,100) x,prob
          enddo
        enddo
      endif ! End probability loop
    enddo
  close(50)
100 format(2f12.6)
200 format(5f12.6)
  stop
  end program
```