SCIENCE RESEARCH COUNCIL
RUTHERFORD & APPLETON LABORATORIES

COMPUTING DIVISION

D I S T R I B U T E D C O M P U T I N G N O T E 3 7 7

PERQ
TECHNICAL NOTE 6

Window Sliding Demonstration

issued by
R W Witty

13 March 1981

Distribution:

F R A Hopgood
R W Witty
D A Duce
C J Prosser
W P Sharpe
I D Benest
J Brown
A S Williams
RL Support/Comp Facils/PERQ/ General file

Appended is a little program to move simulated windows ('fixed' is a solid black square, 'moving' is a stipled square). The moving square moves down over the fixed square. The fixed square is regenerated as the moving square passes over it as opposed to 3RCC Slider which erases the 'lower' window. Having passed completely over the fixed square the moving window goes back up to the top, this time passing 'under' the fixed square. Again the pictures are properly reconstituted.

The point of the demonstration is to show that the 3RCC Slider program is just bad programming rather than an inadequacy of the PERQ.

(This demonstration program is 'quick and dirty'; not my usual 'engineering' standard!).

```
    program overandunder (input,output);
imports screeen from screen;
imports raster  from raster;
imports memory  from memory;


const
  rowbit=256;
  colbit=256;

type
  blk = packed array[1..rowbit,1..colbit] of boolean;
  blkptr = ^blk;

var
  ystep,ypos : integer;
  fixptr,movptr,scrptr : blkptr;
  segno,i,j : integer;
  newtop,newbot,
  oldtop,oldbot,movleft,
  fixtop,fixbot,fixleft : integer;
   procedure drawblock(func:integer;
                       block:blkptr;
                       t,b,l:integer);
     begin
       rasterop(func,
                rowbit,
                b-t,
                l,
                t,
                48,
                scrptr,
                0,
                0,
                16,
                block);
     end;


   function max(a,b:integer):integer;
     begin
       if a>b then max:=a else max:=b;
     end;


   function min(a,b:integer):integer;
     begin
       if a<b then min:=a else min:=b;
     end;

   procedure waitabit;
     var
       i,j,k : integer;

     begin
       for i:=1 to 8  do
         for j:= 1 to 100 do k:=k;
     end;

   begin

     reset(input); rewrite(output);
     write(chr(#14));

     CreateSegment(segno,32,1,32);
     new(segno,4,fixptr);
     new(segno,4,movptr);

     for i:=1 to rowbit do
     for j:=1 to colbit do
     begin
       fixptr^[i,j]:=true;
       if odd(j) then movptr^[i,j]:=true
                 else movptr^[i,j]:=false;
     end;

     scrptr:=makeptr(screenseg,0,blkptr);
```

```
    ystep:=2;
    for ypos:= 10 downto 1 do

      begin
      fixtop:=400;
      fixbot:=fixtop+colbit;
      fixleft:=128;

      oldtop:=10;
      oldbot:=oldtop+colbit;
      movleft:=256;


      {going down - move over}
      drawblock(RRpl,movptr,oldtop,oldbot,movleft);
      drawblock(RRpl,fixptr,fixtop,fixbot,fixleft);
      while oldtop<700 do
      begin
        newtop:=oldtop+ystep;
        newbot:=newtop+colbit;

        drawblock(RRpl,movptr,newtop,newbot,movleft);
        drawblock(RXor,movptr,oldtop,newtop,movleft);

        if ((fixtop <= oldtop) and (oldtop <= fixbot)) or
           ((fixtop <= newtop) and (newtop <= fixbot))
        then
           drawblock(RRpl,fixptr,max(oldtop,fixtop),
                                 min(newtop,fixbot),fixleft);
        waitabit;

        oldtop:=newtop;
        oldbot:=newbot;
      end;
      waitabit;

      {going up - move under}
      drawblock(RRpl,fixptr,fixtop,fixbot,fixleft);
      while oldtop>10 do
        begin
          newtop:=oldtop-ystep;
          newbot:=newtop+colbit;

          drawblock(ROr,movptr,newtop,newbot,movleft);
          drawblock(RNot,fixptr,newbot,oldbot,movleft);

          if ((fixtop <= oldbot) and (oldbot <= fixbot)) or
             ((fixtop <= newbot) and (newbot <= fixbot))
          then
             drawblock(RRpl,
                       fixptr,
                       max(fixtop,newbot),
                       fixbot,fixleft);
          waitabit;

          oldtop:=newtop;
          oldbot:=newbot;
        end;
      waitabit;

      end;

end.
```