```
      technology  = copper26
      *ugd_copper = uc
      *bur_copper = bc
      *aer_copper = ac
      *ugd_fiber  = uf
      *bur_fiber  = bf
      *aer_fiber  = af
      pct_ugd     = pct_ugd
      pct_bur     = pct_bur
      pct_aer     = pct_aer

c26 = (uc * ac_ugd_cop + bc * ac_bur_cop + ac * ac_aer_cop
    + uf * ac_ugd_fib + bf * ac_bur_fib + af * ac_aer_fib)
    * feeder_distance + tmp3

c24 = call feed_cable_cost                                  (cable.pas)
      pass variables:
      lines       = l24
      density     = density
      technology  = copper24
      *ugd_copper = uc
      *bur_copper = bc
      *aer_copper = ac
      *ugd_fiber  = uf
      *bur_fiber  = bf
      *aer_fiber  = af
      pct_ugd     = pct_ugd
      pct_bur     = pct_bur
      pct_aer     = pct_aer

c24 = (uc * ac_ugd_cop + bc * ac_bur_cop + ac * ac_aer_cop
    + uf * ac_ugd_fib + bf * ac_bur_fib + af * ac_aer_fib)
    * feeder_distance + tmp3

ct1 = call feed_cable_cost                                  (cable.pas)
      pass variables:
      lines       = lt1
      density     = density
      technology  = t_1
      *ugd_copper = uc
      *bur_copper = bc
      *aer_copper = ac
      *ugd_fiber  = uf
      *bur_fiber  = bf
      *aer_fiber  = af
      pct_ugd     = pct_ugd
      pct_bur     = pct_bur
      pct_aer     = pct_aer

ct1 = (uc * ac_ugd_cop + bc * ac_bur_cop + ac * ac_aer_cop
    + uf * ac_ugd_fib + bf * ac_bur_fib + af * ac_aer_fib)
    * feeder_distance + tmp2


cf = call feed_cable_cost                                   (cable.pas)
     pass variables:
     lines       = lf
```



```
      density     = density
      technology  = fiber
      *ugd_copper = uc
      *bur_copper = bc
      *aer_copper = ac
      *ugd_fiber  = uf
      *bur_fiber  = bf
      *aer_fiber  = af
      pct_ugd     = pct_ugd
      pct_bur     = pct_bur
      pct_aer     = pct_aer

cf  = (uc * ac_ugd_cop + bc * ac_bur_cop + ac * ac_aer_cop
    + uf * ac_ugd_fib + bf * ac_bur_fib + af * ac_aer_fib)
    * feeder_distance + tmp1

technology = copper26

if (c24 < c26)
 or (feeder_distance + SA_array[i].MaxDistance > copper_gauge_xover) then
    technology = copper24
end if

if ((ct1 < min( c24, c26)))                                 (global.pas)
 or (feeder_distance + SA_array[i].MaxDistance > max_copper_distance)
 or (feeder_distance > copper_t1_xover) then
    technology = t_1
end if


if (cf < min( min(c24,c26), ct1))
 or (feeder_distance > t1_fiber_xover) then
    technology = fiber
end if


SA_array[i].feeder_technology = technology

if technology = fiber then
    SA_array[i].fiber_terminal_cost = call fiber_terminal_cost_fn   (terminal.pas)
                                      pass variables:
                                      lines    = SA_array[i].lines / FillFactor
                                      distance = feeder_distance
                                      density  = SA_array[i].density
                                      *n2016   = n2016
                                      *n672    = n672
                                      *n96     = n96
                                      *n24     = n24
                                      pct_ugd  = pct_ugd
                                      pct_bur  = pct_bur
                                      pct_aer  = pct_aer

    SA_array[i].n2016 = n2016
    SA_array[i].n672 = n672
    SA_array[i].n96 = n96
    SA_array[i].n24 = n24
```

```
elseif technology = t_1 then

    SA_array[i].t1_terminal_cost = call t1_terminal_cost_fn        (terminal.pas)
                                   pass variables:
                                   lines = SA_array[i].lines / FillFactor
                                   *nc96 = n96
                                   *nc24 = n24

    SA_array[i].nc96 = n96
    SA_array[i].nc24 = n24
    n2016 = 0
    n672 = 0

else                                                    { technology is analog }

    SA_array[i].interface_cost = tmp3

    { Add in switched DS1 line terminals }

    if (technology = copper26) or (technology = copper24) then
        SA_array[i].t1_terminal_cost = SA_array[i].t1_terminal_cost
                                     + call t1_terminal_cost_fn    (terinal.pas)
                                       pass variables:
                                       lines = (SA_array[i].SwitchedDS1
                                               + SA_array[i].SpclAccessDS1) * 12
                                       *nc96 = n96
                                       *nc24 = n24

        SA_array[i].nc96 = SA_array[i].nc96 + n96
        SA_array[i].nc24 = SA_array[i].nc24 + n24
        n2016 = 0
        n672 = 0
        n96 = SA_array[i].nc96
        n24 = SA_array[i].nc24
    end if


end if
```



```
lotdiv.pas

the only procedure used outside this module is lot_divide


procedure lot_divide
    passed variables:
    number_of_lots
    var NS_lots
    var EW_lots

    local constatns:
    two = 2

    local variables:
    i
    waste
    minwaste
    NS_try
    EW_try
    NS_try_d
    sqnl
    sqrt2

    {    This procedure minimizes wasted lots within a square microgrid, subject
         to the constraint that lots have lengths no more than twice their widths.
         It returns the "optimal" number of lots in the NS and EW direction. }


    NS_lots = one
    EW_lots = one

    if (number_of_lots > one) then
        sqrt2 = sqrt(two)
        waste = number_of_lots
        minwaste = number_of_lots
        sqnl = sqrt(number_of_lots)
        for i = round(sqnl / sqrt2) to round(sqnl) + 1

        { Check from square root of number of lots/2 to square }
        { root of number of lots.  This guarantees that max    }
        { length - width ratio is no more than 2.              }

            EW_try = i
            NS_try_d = number_of_lots / EW_try
            NS_try = round(NS_try_d)
            waste = NS_try * EW_try - number_of_lots
            if (waste < 0) then waste = number_of_lots
            if (waste <= minwaste) then
                minwaste = waste
                EW_lots = round(EW_try)
                NS_lots = round(NS_try)
            end if
        next

    else
```

tdiv.pas

```
      EW_lots = round(1)
      NS_lots = round(1)
  end if
```


global.pas – data structures

```
global.pas

Data Structures

type d4vector = array[1..4] of double
     i4vector = array[1..4] of integer

     d50 = array[1..50] of double
     d50_ptr = ^d50

     d4vector_ptr = ^d4vector
     i4vector_ptr = ^i4vector

     string8 = string[8]
     string4 = string[4]
     string12 = string[12]

     techtype = (copper26, copper24, t_1, fiber)

     d50x50 = array[1..50,1..50] of double
     i50x50 = array[1..50,1..50] of integer

     d50x50_ptr = ^d50x50
     i50x50_ptr = ^i50x50

type GridRecordType = record
       nrow, ncol : integer
       MicroGridNS, MicroGridEW : double
       cx1, cy1,
       cx2,cy2,
       cx3,cy3,
       cx4,cy4 : d4vector
       gTotalLines, gHouseholds, gBusinessLines, gPrivateLines, gSpecialLines : integer
       LowerLeftX, LowerLeftY, UpperRightX, UpperRightY : double
       Households, Buslines      : array[1..50,1..50] of integer
       SwitchX, SwitchY  : double
       quadrant : integer
       DepthToBedrock : double
       Hardness : string4
       SoilTexture : string8
       WaterTb : double
       MinSlope : double
       MaxSlope : double
       CBG       : string12 }
     end
     CoordinateRecordType = record
       OriginX, OriginY, reference_latitude : double
     end

     t1 = record
       CableSize : integer
       CostUgd : double
       CostBur : double
       CostAer : double
     end
     t1_ptr = ^t1
```

```
t2 = record
  size : integer
  CostBur : double
  CostAer : double
  CostUgd : double
end
t2_ptr = ^t2

t3 = record
   size : integer
   CostUgd : double
   CostBur : double
   CostAer : double
end
t3_ptr = ^t3

t4 = t3
t4_ptr = ^t4

t7 = record
   NumLines : integer
   Cost : double
end
t7_ptr = ^t7

t8 = record
   density : double
   FeedUgd : double
   DistUgd : double
   FeedBur : double
   DistBur : double
   FeedAer : double
   DistAer : double
end
t8_ptr = ^t8

t9=t8
t9_ptr = ^t9

t10 = t8
t10_ptr = ^t10

t11 = record
   DuctCap : integer
   NormalCost : double
   SoftCost : double
   HardCost : double
end
t11_ptr = ^t11

t12 = record
   density : double
   ManholeSpacing : double
end
t12_ptr = ^t12

t13 = record
```



```
   density : double
   UgdPct  : double
   BurPct  : double
   AerPct  : double
end
t13_ptr = ^t13

t14=t13
t14_ptr = ^t14

t15=t13
t15_ptr = ^t15

t16 = record
   density : double
   FeedFillFactor : double
   DistFillFactor : double
end
t16_ptr=^t16

t22 = record
   texture : string8
   impact : integer
end
t22_ptr=^t22

t23=record
   density : double
   bur_share : double
   ugd_share : double
   aer_share : double
end
t23_ptr=^t23

IntfcType = (primary,secondary)

GaugeType = (g19, g22, g24, g26)

SARecordType = record
   number_of_SAIs          : integer
   TypeOfSAI               : array[1..4] of IntfcType
   x                       : d4vector
   y                       : d4vector
   SAI_lines               : d4vector
   SwitchX                 : double
   SwitchY                 : double
   households              : double
   lines                   : double
   ResLines                : double
   BusLines                : double
   SpclAccessLines         : double
   SpclAccessDS1           : double
   SwitchedDS1             : double
   Grid_Distribution_Cost  : double
   MaxDistance             : double
   n2016, n672, n96, n24   : integer
   nc96, nc24              : integer
```

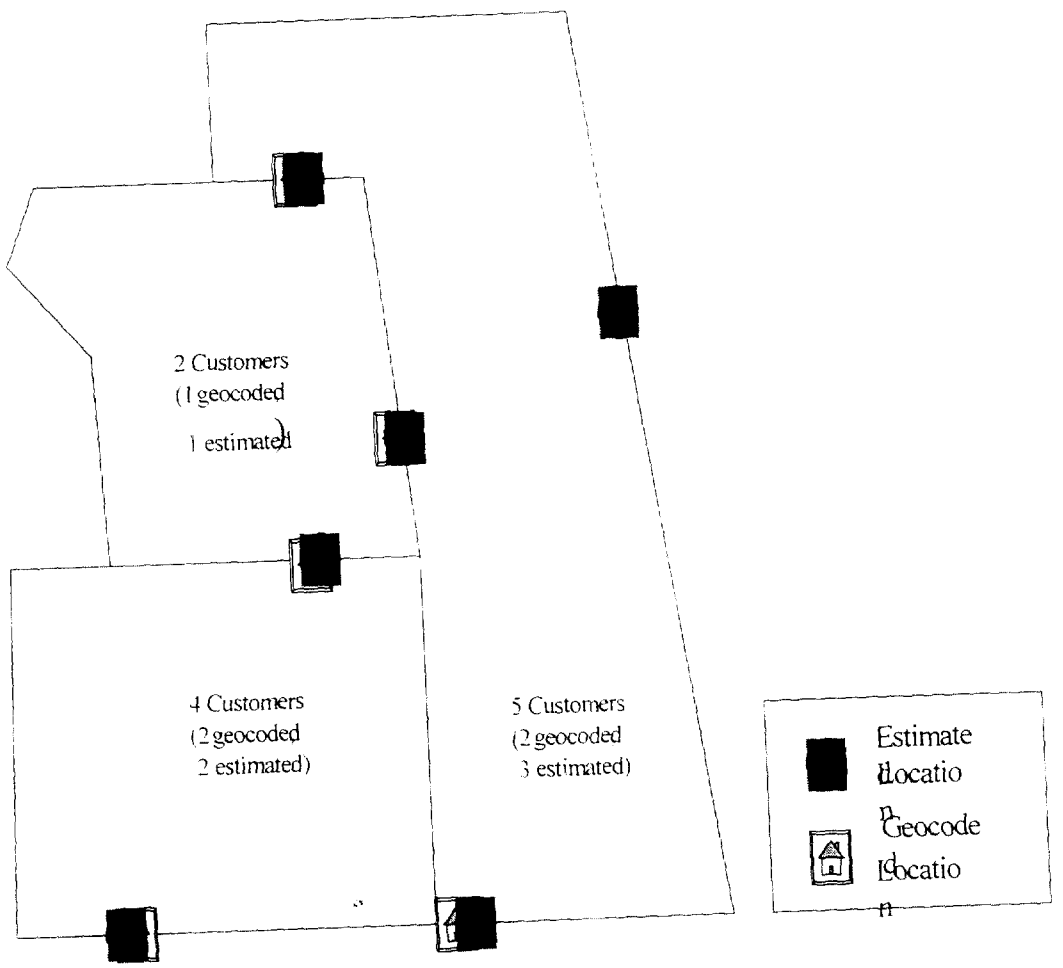
```
    anc96,anc24              : integer
    fiber_terminal_cost      : double
    t1_terminal_cost         : double
    secondary_tterm_cost     : double
    interface_cost           : double
    drop_cost                : double
    nid_cost                 : double
    drop_terminal_cost       : double
    feeder_technology        : techtype
    grid_line_feet           : double
    link_line_feet           : double
    grid_drop_feet           : double
    density                  : double
    DepthToBedrock           : double
    Hardness                 : string4
    SoilTexture              : string8
    WaterTb                  : double
    MinSlope                 : double
    MaxSlope                 : double
    DistToSwitch             : double
    subfeeder_try            : double
    subfeeder_coord          : double
    quadrant                 : integer
    ugd_cable                : double
    bur_cable                : double
    aer_cable                : double
    ugd_structure            : double
    bur_structure            : double
    aer_structure            : double
    ManholeCost              : double
    lines_served             : double
    CBG                      : string12
end

SARecordType_ptr = ^SARecordType
GridRecordType_ptr = ^GridRecordType

glsarray = ARRAY [1..8000] OF double
glrarray = ARRAY [1..8000] of SARecordType_ptr

glsarray_ptr = ^glsarray
glrarray_ptr = ^glrarray

pe d8000 = array[1..8000] of double
   i8000 = array[1..8000] of integer
   b8000 = array[1..8000] of boolean

   d8000_ptr = ^d8000
   i8000_ptr = ^i8000
   b8000_ptr = ^b8000

   d8000x8000 = array[1..8000] of d8000_ptr
   d8000x8000_ptr = ^d8000x8000
pe d8000 = array[1..8000] of double;
   i8000 = array[1..8000] of integer;
   b8000 = array[1..8000] of boolean;
```



```
    d8000_ptr = ^d8000;
    i8000_ptr = ^i8000;
    b8000_ptr = ^b8000;

    d8000x8000 = array[1..8000] of d8000_ptr;
    d8000x8000_ptr = ^d8000x8000;


type d4 = array[1..4] of double;
     i4 = array[1..4] of integer;
     b4 = array[1..4] of boolean;

     d4_ptr = ^d4;
     i4_ptr = ^i4;
     b4_ptr = ^b4;

     d4x4 = array[1..4] of d4_ptr;
     d4x4_ptr = ^d4x4;
```

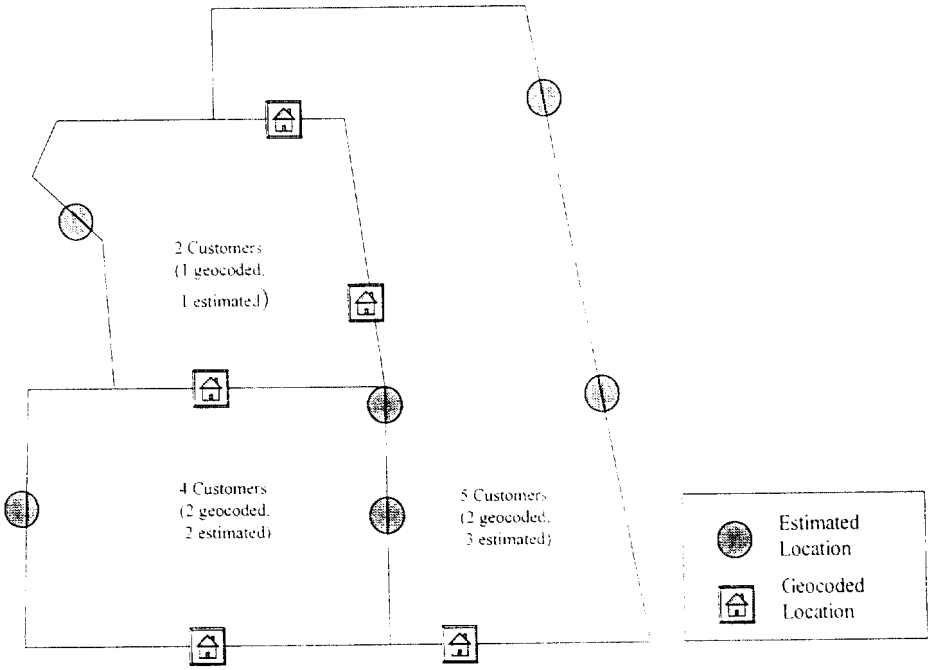
Attachment C-1

Pure Uniform Placement

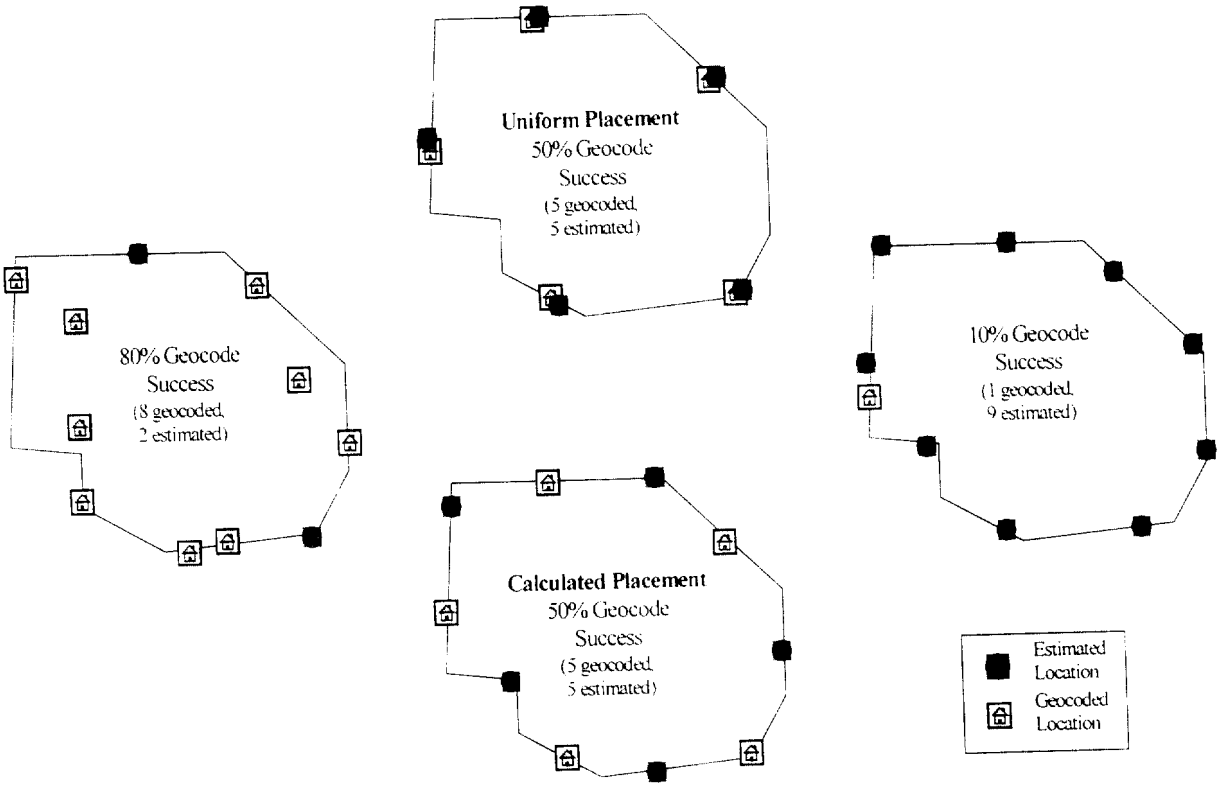


Attachment C-2

Calculated Uniform Placement



Attachment C-3



CERTIFICATE OF SERVICE

I do hereby certify that I have this 28th day of August 1998 served the following parties to this action with a copy of the foregoing JOINT COMMENTS OF BELLSOUTH TELECOMMUNICATIONS, INC., U S WEST, INC., AND SPRINT CORPORATION TO COMMON CARRIER BUREAU REQUEST FOR COMMENT ON MODEL PLATFORM DEVELOPMENT by placing a true and correct copy of the same in the United States Mail, postage prepaid, addressed to the parties listed below.

*Magalie Roman Salas
Secretary-Federal Communications Commission
1919 M Street, N. W.
Room 222
Washington, D. C. 20554

*Sheryl Todd
Common Carrier Bureau
Federal Communications Commission
2100 M Street, N. W.
8th Floor
Washington, D.C. 20554

*International Transcription Service
1231 20th Street, N. W.
Washington, D. C. 20036

Julia W. Spires