arbitrary conic section

$$\{Ax^2 + By^2 + Cxy + Dx + Ey + F = 0\}$$

class of conic has nothing to do with value of $D + E$

3 classes: ellipses, hyperbolae, parabolas

parabola iff $4AB = C^2$

Memory to Memory Add -(others can be space/speed traded too)

speed optimized: (minimum count = 2)

7 words
1 + 3*count mcycles

```
src+   cnt                                   ; fetch S0
dst    ain:∅   bin:rddat                      ; fetch D0, save S0
src+   ain:sum bin:rddat CRY                  ; form Sum0, fetch
loop { WRT dst+ ain:∅   bin:rddat cnt
       dst      ain:sum
       src+     ain:sum bin:rddat cin:cry CRY TC:-2
WRT dst+ done
```

space optimized: (min. count = 1)

6 words
3 + 3*count mcycles

```
src+ CRY                                      ; clear carry reg., fetch S0
bin:rddat ain:∅                               ; fetch D0, save S0
1: { dst    ain:sum cnt
     src+   ain:sum bin:rddat cin:cry CRY
   WRT dst+ ain:∅   bin:rddat TC:-2
done
```

speed optimized mem from mem subtract Dst - Src → Dst

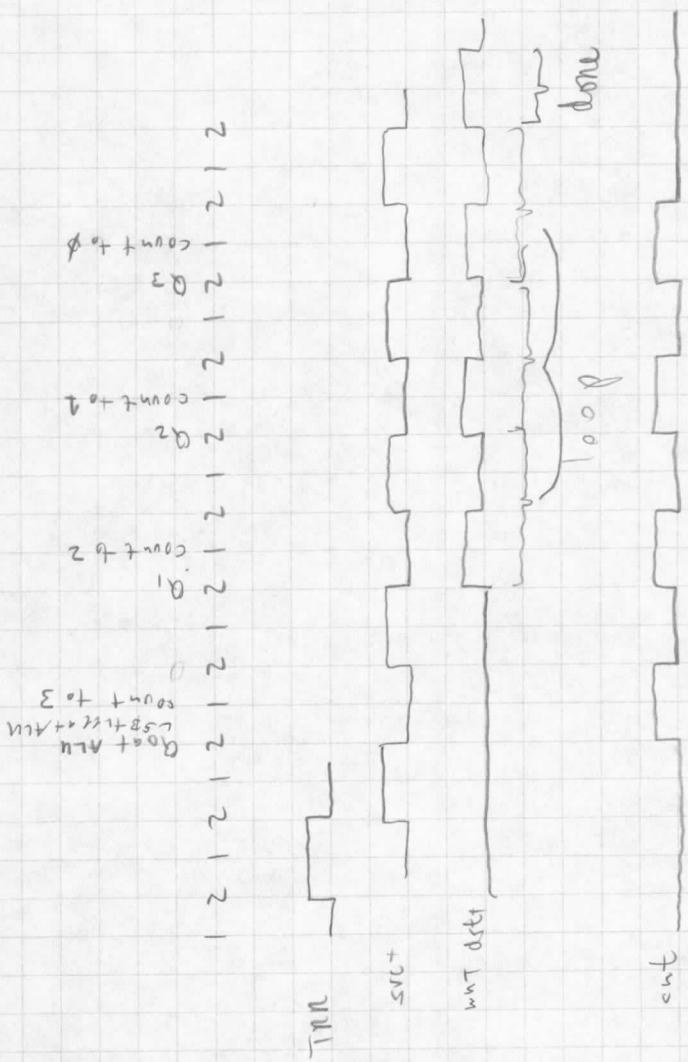
```
src+ cnt
dst  ain:∅   bin:rddatbar
src+ ain:sum bin:rddat cin:1 CRY
{ WRT dst+ ain:∅   bin:rddatbar cnt
  dst      ain:sum
  src+     ain:sum bin:rddat cin:cry CRY TC:-2
WRT dst+ done
```

space optimized

```
src+ ain:1 bin:1 CRY
bin:rddatbar ain:∅
{ dst    ain:sum cnt
  src+   ain:sum bin:rddat cin:cry CRY
  WRT dst+ ain:∅ bin:rddatbar TC:-2
done
```

MICRO-TIMING FOR CIRCLE LOAD (cload)

St SrcAddr = LSB (or appr. bit) of Q-buffer
St DstAddr = LSB of intensity buffer
Instrcnt = length of intensity buffer
MBI = ∅
exp = DLEN − (NL+4)



MICRO TIMING FOR DISKDEFINE (disk)

StSrcAddr = LSB of Q-buffer (Q0)
StDstAddr = N/A
InstrCnt = length of Q-buffer (QLEN)
MBI = 1
exp = QLEN − (NL+2)

DISK-DEFINE

you define $A = 2x_c$, $B = 2y_c$, $C = r^2 - (x_c^2 + y_c^2)$
Q register = $x^2 + y^2$

for 512^2 system Q is 19 bits, unsigned

in ALU, compute $F = (A_x + B_y + C) - Q$
a pixel is enabled iff $F \geq 0$

specifically, add 1's complement of tree result to Q buffer, then AND "sum" of result with ENABLE register

so ain ← treebus
bin ← rddat
and look at MSB of result

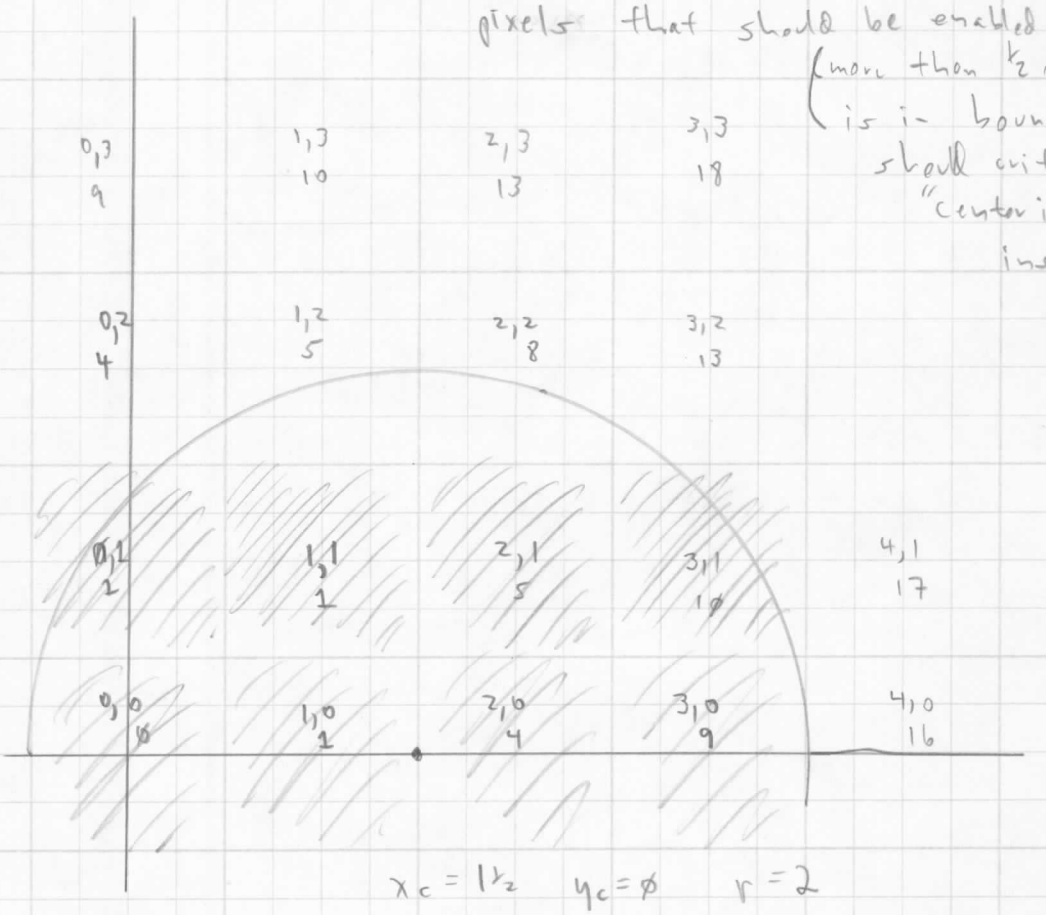
alternatively,
store $-(x^2+y^2+1)$ in Q-buffer (same as 1's complement of x^2+y^2
let $C = r^2 + 1 - (x_c^2 + y_c^2)$
and add tree result to Q-buffer [or add the 1 into C using carry bit at LSB
(although this would prevent using carry to add 0.1 or ½ roundoff into tree result)

Example of circle enable calculation

each big square represents a pixel, labeled by its x,y coordinates and $Q = x^2+y^2$ value

pixels that should be enabled are shaded (more than ½ of area is in boundary

should criterion be "center is in boundary" instead?



$x_c = 1\frac{1}{2}$ $y_c = 0$ $r = 2$

so $C = r^2 - (x_c^2 + y_c^2) = 1\frac{3}{4}$

$A = 2x_c = 3$

$B = 2y_c = 0$

inside if $Ax + By + C - Q \geq 0$

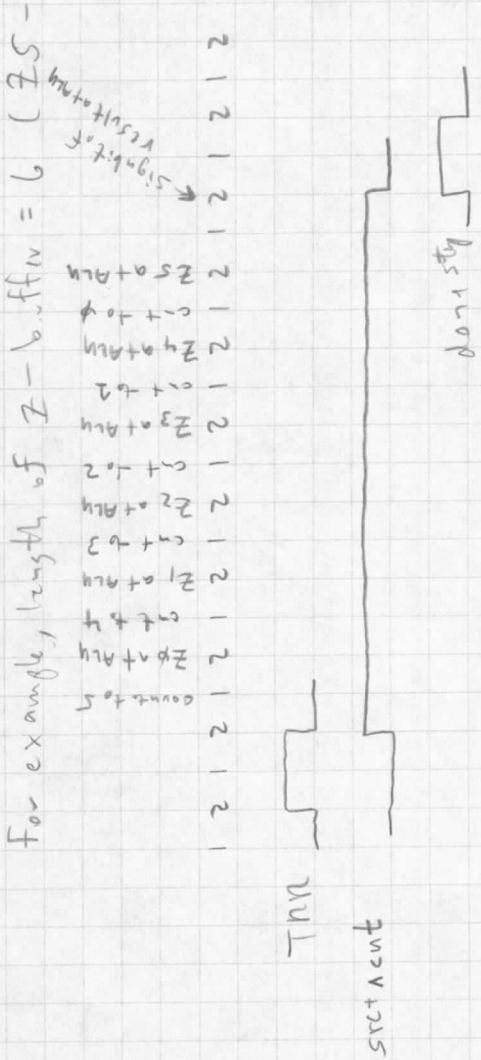
i.e: $3x + 1\frac{3}{4} \geq Q$

so apparently you don't want to add 0.5 to C on tree result

Z - COMPARE (Zcomp)

it is assumed that for all pixels which are still enabled, that the true result will be an unsigned integer which is no longer than the Z-buffer

So For this command, StSrcAddr = LSB of Z-buffer (Zφ)
StDstAddr = N/A
InstrCnt = length of Z-buffer (ZLEN)
MBI = φ
exp = ZLEN - (NL+4)



Theory of Circle Shading

I_E, I_c are desired color intensities at edge and center of circle respectively

i_e and i_c are the actual intensities obtained

n_p is the number of "passes" made

the algorithm first must select $k_1 < k_2 < \ldots < k_{n_p}$ such that

$$S = \sum_{i=1}^{n_p} 2^{k_i} \approx \frac{I_c - I_E}{r^2}$$

the accuracy will of course depend on n_p; 2, 3, or 4 is probably a good number

load the Q buffer ($Q_{18} \cdots Q_0$ for a 512 × 512 system) with $x^2 + y^2$, and provide additional bits, $Q_{-1}, Q_{-2}, \ldots Q_{-u}$ which are loaded with 0's

then, if we set $A = 2^{j+1} x_c$

$B = 2^{j+1} y_c$

$$C = \begin{cases} 2^j [r^2 - (x_c^2 + y_c^2)] & j \neq 0 \\ r^2 - (x_c^2 + y_c^2) + I_c - S r^2 & j = \phi \end{cases}$$

and Shift Src Addr = Q_{-j}

then $Ax + By + C + Q = 2^j \{ r^2 - [(x - x_c)^2 + (y - y_c)^2$

(for $j \neq \phi$)

theory of circle sliding

if we do this for $j = k_1, k_2, \ldots, k_{np}$ and add the results, we obtain

$$S\{r^2 - [(x-x_c)^2 + (y-y_c)^2]\} + I_c - Sr^2$$

so
$$\left.\begin{aligned} i_c &= I_c \\ i_e &= I_c - Sr^2 \approx I_E \\ i_c - i_e &= Sr^2 \end{aligned}\right\}$$ intensity at center is exact, intensity at edge is approximate

practical limitations: in a 512 x 512 system, we need 19 bits to represent $x^2 + y^2$;
if we only use 20 bits for the Q-buffer, i.e: $q = 1$, $Q_{-1} = \phi$, we "waste" 1 bit
then $k_{np} \leq 1$, since the k_i are in ascending order of magnitude, hence $S \leq 2^-$
i.e: if $n_p = 3$, $S \leq 3\frac{1}{2}$, so $i_c - i_e \leq 3\frac{1}{2} r^2$
so for 8 bit intensity buffers, the smallest circle for which intensity could go from 0 to max from edge to center would have radius 9
additionally, the constraint $k_1 \geq -12$ would have to be imposed,

THEORY: MBR TOKEN REGISTER + MBR SHIFTER

if length A (not including signbit) = $\bar{A}$
length B (") = $\bar{B}$
length C (") = $\bar{C}$

and the tree has NL levels,
then the result may be as long as $2 + \max(\bar{A}+NL, \bar{B}+NL, \bar{C})$ bits long, not including signbit

(signbit)
so if MBI is 1, the most significant bit of the tree result is important, then the coefficients must be sign extended to

$N = \max(\bar{A}+NL, \bar{B}+NL, \bar{C}) + 3$ bits

so a 1 must be placed in the MBR shifter at index $N-1$ (where rightmost bit is $\emptyset$)
these coefficient lengths include fracbits

9+fbits

note that the length of a coefficient, not including signbit, is

$\overline{COEF} = (\#\text{ fracbits}) + (exp) + 1$

where exp is the IEEE format exponent

assuming shifter indexes match, token register indexes
for A + B place a 1 in token register at

index = fbits + exp(A or B) + NL + 3

For C place a 1 in token register at

index = fbits + expC + 3

[the number of bits of a coefficient that get shifted out is (index of leftmost 1) + 1]

if instead, we place for C a 1

at index = fbits + expC + NL + 3

(analagously to A & B coefficients)

then we waste clock cycles whenever

$\bar{C} < \max(\bar{A}, \bar{B}) + NL$

and MBI = 1

so we don't waste clock cycles on planar loads where A = B = 0, because MBI = ∅ and A, B, C coefficient tokens are cleared on IAEH

For the exponent field of the I word, there are several things we may want to do:

1) for an instruction with no coeffs, set expfield = 255 and MBI = 0; the MBR shifter will be loaded with all 0's

2) for an instruction for which only the first FN bits of the tree result are important, only the first FN bits of the coefficients need be transmitted; set MBI = ∅ to clear the tokens loaded for the coeffs, and place a 1 in the MBR shifter at

index = FN - 1

this would be done by setting

expfield = 127 + FN' - (NL + 4) where FN' = FN - fbi

3) For an instruction for which the MSB (signbit) of the tree result is important, MBI = 1, but the result need have no minimum length, i.e.: edges, stripes, we must insure that MBR occurs no earlier than (or maybe) 2 clock cycles after TRR since TRR occurs 2 cycles before the LS wholebit of result appears at ALU and MBR occurs 1 cycle before signbit appears at ALU, (i.e.: signbit arrives at tree 1 cycle after LSB wholebit arrives) we want to pretend that result can be 2+fbits long including signbit so should plan 1 in MSB token regist at index = fbits + 1

so expfield = 127 − (NL+2)

so a cycle is wasted if fbits < NL+1

4) For an instruction for which the MSB (signbit) of tree result is important, but result must be sign-extended to FW bits in any case (such as disk-define), set MBI = 1, and: we want to insure that MBR occurs no earlier than 1 cycle after ICT2H

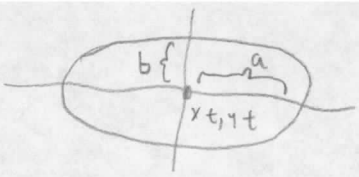
individual cases must be handled differently

for disk define, MBR must occur no earlier than QLEN+2 microcycles after TRR

plan 1 in token reg. at index = fbits + QLEN+1

expfield = 127 + QLEN − (NL+2)

Drawing Ellipses



$$E = a^2 x_r^2 + b^2 (1 - x_r^2)$$
$$F = a^2 (1 - x_r^2) + b^2 x_r^2$$
$$G = 2 x_r \sqrt{1 - x_r^2}\,(b^2 - a^2)$$

x_r refers to rotation about origin

$$\begin{cases} A = 2 x_t F + y_t G \\ B = 2 y_t E + x_t G \\ C = a^2 b^2 - (x_t^2 F + x_t y_t G + y_t^2 E) \end{cases}$$

x_t, y_t is center

$$Q = F x^2 + G x y + E y^2$$

within ellipse $\equiv \left[A x + B y + C - Q \geq 0 \right]$

equivalent to within ellipse $\equiv$

$$F(x - x_t)^2 + G(x - x_t)(y - y_t) + E(y - y_t)^2 \leq a^2 b^2$$

with a along x-axis, b along y-axis:

$$E = a^2,\ F = b^2,\ G = \emptyset$$
$$A = 2 x_t b^2,\ B = 2 y_t a^2,\ C = a^2 b^2 - x_t^2 b^2 - y_t^2 a^2$$
$$Q = b^2 x^2 + a^2 y^2$$

test example of ellipse

let $x_t = y_t = 15$ (center at 15, 15)
$x_r = \sqrt{2}/2$ (rotated 45° to normal)
$a = 10, b = 6$ (bounding box 20 × 12)

then $E = 68, F = 68, G = -64$

$A = 30 \cdot 68 - 15 \cdot 64 = 1080$
$B = 30 \cdot 68 - 15 \cdot 64 = 1080$
$C = 3600 - (225 \cdot 68 + 225 \cdot 68 - 225 \cdot 64) = -12600$
$Q = 68x^2 - 64x_4 + 68y^2$

scale everything by $1/4$
$A = B = 270, \quad C = -3150$
$Q = 17x^2 - 16x_4 + 17y^2$
$= x(17x - 16y) + y(17y)$

implemented in e12.in

17 bits should suffice for Q-buffer