

Fundamentals of Computer Systems

Bresenham's Line Algorithm in Hardware

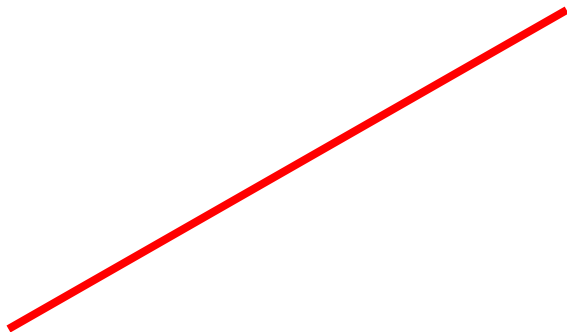
Stephen A. Edwards

Columbia University

Fall 2012

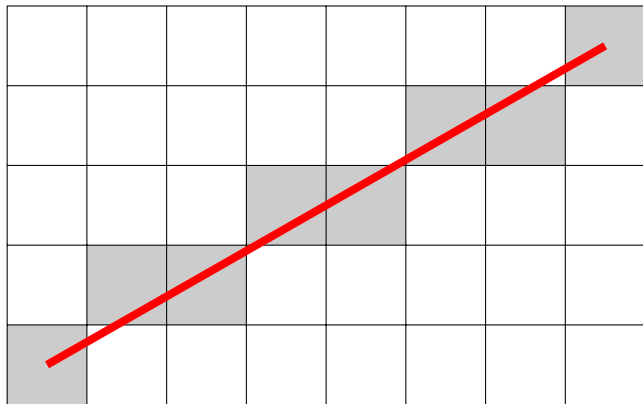
Bresenham's Line Algorithm

Objective: Draw a line...



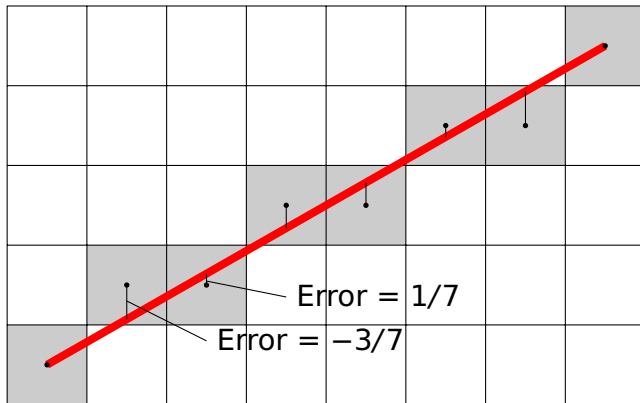
Bresenham's Line Algorithm

...with well-approximating pixels...



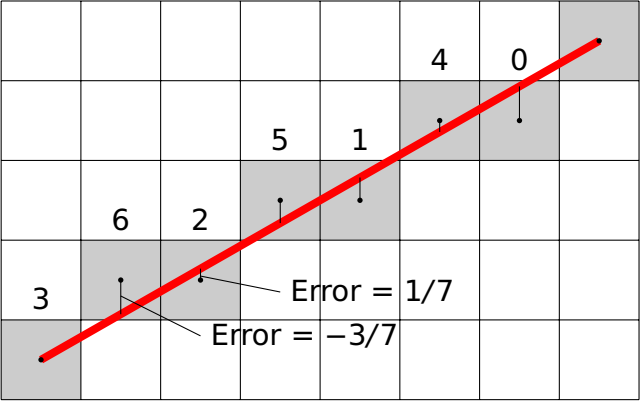
Bresenham's Line Algorithm

...by maintaining error information..



Bresenham's Line Algorithm

...encoded using integers



The Pseudocode from Wikipedia

```
function line(x0, y0, x1, y1)
  dx := abs(x1-x0)
  dy := abs(y1-y0)
  if x0 < x1 then sx := 1 else sx := -1
  if y0 < y1 then sy := 1 else sy := -1
  err := dx-dy

  loop
    setPixel(x0,y0)
    if x0 = x1 and y0 = y1 exit loop
    e2 := 2*err
    if e2 > -dy then
      err := err - dy
      x0 := x0 + sx
    end if
    if e2 < dx then
      err := err + dx
      y0 := y0 + sy
    end if
  end loop
```

My C Code

```
void line(Uint16 x0, Uint16 y0, Uint16 x1, Uint16 y1)
{
    Sint16 dx, dy; // Width and height of bounding box
    Uint16 x, y; // Current point
    Sint8 sx, sy; // -1 or 1
    Sint16 err; // Loop-carried value
    Sint16 e2; // Temporary variable
    int right, down; // Boolean

    dx = x1 - x0; right = dx > 0; if (!right) dx = -dx;
    dy = y1 - y0; down = dy > 0; if (down) dy = -dy;
    err = dx + dy; x = x0; y = y0;
    for (;;) {
        plot(x, y);
        if (x == x1 && y == y1) break; // Reached the end
        e2 = err << 1; // err * 2
        if (e2 > dy) { err += dy; if (right) x++; else x--;}
        if (e2 < dx) { err += dx; if (down) y++; else y--;}
    }
}
```

Datapath for dx , dy , *right*, and *down*

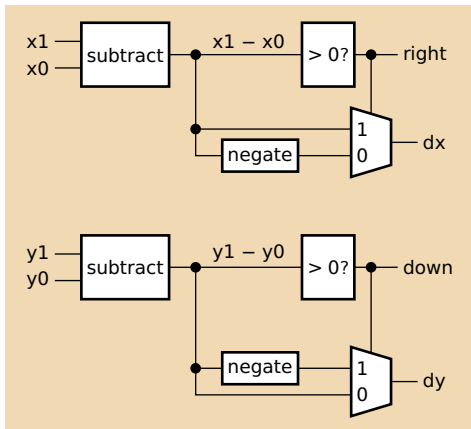
```
void line(Uint16 x0, Uint16 y0,
         Uint16 x1, Uint16 y1)
{
    Sint16 dx;    // Width of bounding box
    Sint16 dy;    // Height of BB (neg)
    Uint16 x, y;  // Current point
    Sint8  sx, sy; // -1 or 1
    Sint16 err;   // Loop-carried value
    Sint16 e2;    // Temporary variable
    int right;    // Boolean
    int down;     // Boolean
```

```
    dx = x1 - x0;
    right = dx > 0;
    if (!right) dx = -dx;
    dy = y1 - y0;
    down = dy > 0;
    if (down) dy = -dy;
```

```
    err = dx + dy;
    x = x0; y = y0;
```

```
    for (;;) {
        plot(x, y);
        if (x == x1 && y == y1)
            break;
        e2 = err << 1;
        if (e2 > dy) {
            err += dy;
            if (right) x++;
            else x--;
        }
        if (e2 < dx) {
            err += dx;
            if (down) y++;
            else y--;
        }
    }
}
```


Datapath for dx , dy , *right*, and *down*

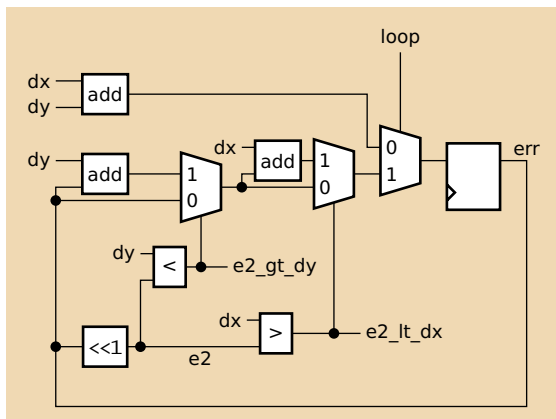


```
dx = x1 - x0;  
right = dx > 0;  
if (!right) dx = -dx;  
dy = y1 - y0;  
down = dy > 0;  
if (down) dy = -dy;
```

```
err = dx + dy;  
x = x0; y = y0;
```

```
for (;;) {  
    plot(x, y);  
    if (x == x1 && y == y1)  
        break;  
    e2 = err << 1;  
    if (e2 > dy) {  
        err += dy;  
        if (right) x++;  
        else x--;  
    }  
    if (e2 < dx) {  
        err += dx;  
        if (down) y++;  
        else y--;  
    }  
}
```

Datapath for *err*

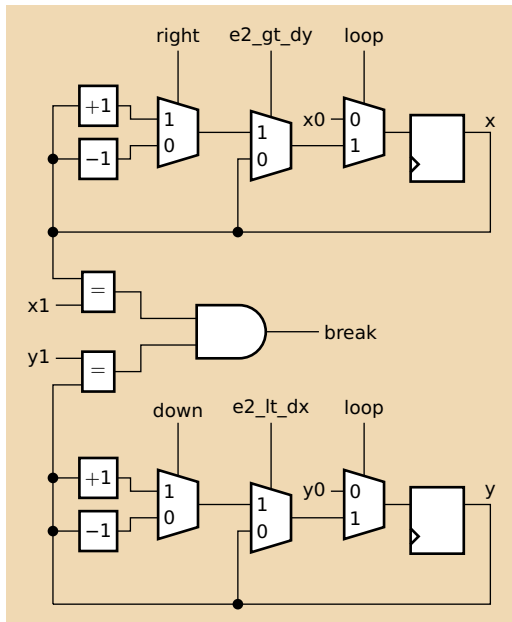


```
dx = x1 - x0;
right = dx > 0;
if (!right) dx = -dx;
dy = y1 - y0;
down = dy > 0;
if (down) dy = -dy;
```

```
err = dx + dy;
x = x0; y = y0;
```

```
for (;;) {
    plot(x, y);
    if (x == x1 && y == y1)
        break;
    e2 = err << 1;
    if (e2 > dy) {
        err += dy;
        if (right) x++;
        else x--;
    }
    if (e2 < dx) {
        err += dx;
        if (down) y++;
        else y--;
    }
}
```

Datapath for x and y

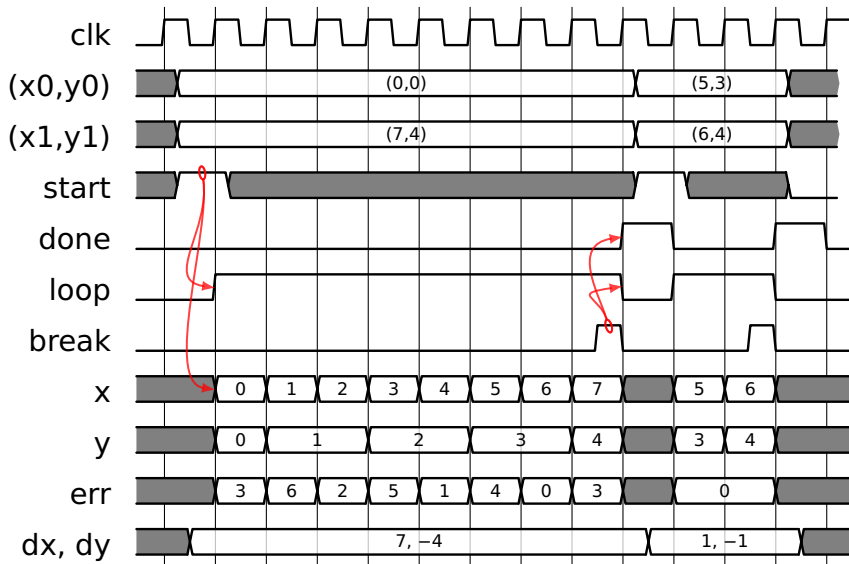


```
dx = x1 - x0;
right = dx > 0;
if (!right) dx = -dx;
dy = y1 - y0;
down = dy > 0;
if (down) dy = -dy;
```

```
err = dx + dy;
x = x0; y = y0;
```

```
for (;;) {
    plot(x, y);
    if (x == x1 && y == y1)
        break;
    e2 = err << 1;
    if (e2 > dy) {
        err += dy;
        if (right) x++;
        else x--;
    }
    if (e2 < dx) {
        err += dx;
        if (down) y++;
        else y--;
    }
}
```

Timing



Control FSM

