Subject: Some new functions
Posted by koldo on Sat, 20 Nov 2010 00:01:54 GMT
View Forum Message <> Reply to Message

Hello

Here there are a few functions that could be added to U++:

```
inline bool Odd(int val)    {return val%2;}
inline bool Even(int val)     {return !Odd(val);}
inline int RoundEven(int val)   {return Even(val) ? val : val+1;}
template<class T>
inline int Sign(T a)    {return (a > 0) - (a < 0);}

inline const RGBA *GetPixel(const Image &img, int x, int y) {
 return img + x + y*img.GetWidth();
}
inline RGBA *GetPixel(ImageBuffer &img, int x, int y) {
 return img + x + y*img.GetWidth();
}
```

Subject: Re: Some new functions
Posted by Didier on Sat, 20 Nov 2010 16:22:05 GMT
View Forum Message <> Reply to Message

Hi Koldo,

I think the Odd() function would be faster this way (although the compiler might already optimize it this way):

```
inline bool Odd(int val) {return (val & 0x1);}
```

Not a big optimization, rather a very tiny one. But with drops you can fill the sea

Subject: Re: Some new functions
Posted by mirek on Sat, 20 Nov 2010 17:07:05 GMT
View Forum Message <> Reply to Message

Instead of GetPixel, you can write

image[y][x]

Mirek

Subject: Re: Some new functions
Posted by dolik.rce on Sat, 20 Nov 2010 19:55:36 GMT
View Forum Message <> Reply to Message

Fun fact: I tried to come up with optimal solution for Even() and all of the following appear to have same speed as the one from Koldo (with gcc optimal+speed flag):inline bool even1(int val) {return !(val&1);}
inline bool even2(int val) {return ~val&1;}
inline bool even3(int val) {return !(val%2);}
Without the speed flag even2() seems to be slightly faster.

Also val%2 and val&1 for Odd() yields the same speed in both cases

However, the proposed RoundEven() function is suboptimal thanks to the branching. Even though it won't probably be used often, I would suggest faster version:inline int roundeven(int val) {return ((1+val)>>1)<<1;}
//for completeness also rounding to odd numbers:
inline int roundodd(int val) {return ((val>>1)<<1)+1;}

Regarding the image access: The img[y][x] is great, but still it would be nice to have a wrapper that would allow to put the arguments in (imho) more natural order. For example something like RGBA* Image::Get(int x,int y){return (*this)[y][x];}

Best regards,
Honza

Subject: Re: Some new functions
Posted by koldo on Sat, 20 Nov 2010 22:22:37 GMT
View Forum Message <> Reply to Message

Good comments from all.

I like Image::Get() in addition to image[y][x].

About Odd()... you are much smarter than me .

Subject: Re: Some new functions
Posted by koldo on Sat, 20 Nov 2010 23:05:45 GMT
View Forum Message <> Reply to Message

Just some more ...

```
byte BW(Color color) {
 return byte(0.299*color.GetR() + 0.587*color.GetG() + 0.114*color.GetB());
}
```

```
Image Rotate180(const Image& orig) {
 Size sz = orig.GetSize();
 ImageBuffer dest(sz);
 for(int rw = 0; rw < sz.cy; rw++)
  for(int cl = 0; cl < sz.cx; cl++)
   dest[rw][cl] = orig[sz.cy - rw - 1][sz.cx - cl - 1];
 return dest;
}

Image GetRect(const Image& orig, const Rect &r) {
 if(r.IsEmpty())
  return Image();
 ImageBuffer ib(r.GetSize());
 for(int y = r.top; y < r.bottom; y++) {
  const RGBA *s = orig[y] + r.left;
  const RGBA *e = orig[y] + r.right;
  RGBA *t = ib[y - r.top];
  while(s < e) {
   *t = *s;
   t++;
   s++;
  }
 }
 return ib;
}

Color RandomColor() {
 int num = Random();
 return Color(num&0xFF, (num&0xFF00)>>8, (num&0xFF0000)>>16);
}
```

Subject: Re: Some new functions
Posted by dolik.rce on Sat, 20 Nov 2010 23:40:08 GMT
View Forum Message <> Reply to Message

koldo wrote on Sun, 21 November 2010 00:05Just some more ...

...

```
Color RandomColor() {
 int num = Random();
 return Color(num&0xFF, (num&0xFF00)>>8, (num&0xFF0000)>>16);
}
```

Just some more comments
Color RandomColor() {Color(Random(),0);}
BTW: Color BW() is useful sometimes, but it might deserve bit more readable name. What about ToGrayscale() ?

Honza

EDIT: Now I see conversion to grayscale is already available in Core: int Grayscale(const Color& c)
{
 return (77 * c.GetR() + 151 * c.GetG() + 28 * c.GetB()) >> 8;
}

Subject: Re: Some new functions
Posted by koldo on Sun, 21 Nov 2010 07:26:47 GMT
View Forum Message <> Reply to Message

Thank you Honza. Grayscale implementation is interesting, I like it.

Subject: Re: Some new functions
Posted by mr_ped on Mon, 22 Nov 2010 08:09:06 GMT
View Forum Message <> Reply to Message

inline int roundeven(int val) {return ((1+val)&(~1);}
//for completeness also rounding to odd numbers:
inline int roundodd(int val) {return val|1;}

golfing, aren't we?

Subject: Re: Some new functions
Posted by Novo on Thu, 25 Nov 2010 15:43:06 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Sat, 20 November 2010 14:55Fun fact: I tried to come up with optimal solution for Even() and all of the following appear to have same speed as the one from Koldo (with gcc optimal+speed flag):inline bool even1(int val) {return !(val&1);}
inline bool even2(int val) {return ~val&1;}
inline bool even3(int val) {return !(val%2);}
Without the speed flag even2() seems to be slightly faster.

Also val%2 and val&1 for Odd() yields the same speed in both cases

However, the proposed RoundEven() function is suboptimal thanks to the branching. Even though it won't probably be used often, I would suggest faster version:inline int roundeven(int val) {return ((1+val)>>1)<<1;}
//for completeness also rounding to odd numbers:

inline int roundodd(int val) {return ((val>>1)<<1)+1;}

Best regards,
Honza

Hi Honza,

I posted a link to a collection of optimized functions here
http://www.ultimatepp.org/forum/index.php?t=msg&th=5683& amp;start=0&

It looks like it might be helpful for your experiments.

Subject: Re: Some new functions
Posted by mirek on Sat, 04 Dec 2010 19:38:19 GMT
View Forum Message <> Reply to Message

koldo wrote on Fri, 19 November 2010 19:01Hello

Here there are a few functions that could be added to U++:

[code]inline bool Odd(int val) {return val%2;}
inline bool Even(int val) {return !Odd(val);}

Thinking about it, for me it is much easier to remember that a byte has a least significant bit than to remember what is even and what is odd...

Subject: Re: Some new functions
Posted by koldo on Sat, 04 Dec 2010 22:44:50 GMT
View Forum Message <> Reply to Message

mirek wrote on Sat, 04 December 2010 20:38koldo wrote on Fri, 19 November 2010 19:01Hello

Here there are a few functions that could be added to U++:

[code]inline bool Odd(int val) {return val%2;}
inline bool Even(int val) {return !Odd(val);}

Thinking about it, for me it is much easier to remember that a byte has a least significant bit than to remember what is even and what is odd...

For me it is the opposite. When I have to work with bits I need to open the manual to know if it is >> or << ... . I have poor memory. However Odd and Even is just that.

Anyway, there will be always Functions4U .
