*************DRAFT*************

***Testing C code with Ultibo Bare Metal, Ultibo TFTP and Ultibo Bitmaps***

***02/18/17***

*************DRAFT*************

Goal: This is in hopes of improving the speed of computing the JPEG 2000. The RPi2B or RPi3B will run Ultibo Bare Metal.

To transfer images over an Ethernet connection to a RPi2B or RPi3B.

Perform the JPEG 2000 lifting step which is the first step in the JPEG 2000.

The C code which which performs the Lifting step was develop by

Dan Gisselquist, Ph.D.
Gisselquist Technology, LLC

The C code that performs the DWT Lifting Step runs on x86_64 6 core is considerably faster.
The /proc/cpuinfo for the 3 tests is found at Appendix A. CPU info for x86_64 six core, x86_64 dual core, and RPi3B.

Number of cores and clock appears to improve the overall performance.
The cpu MHz 800.000 for both the x86_64 dual & x86_64 six core. While the clock of the RPi3B is 1.2GHz.

time ./liftmain lena_rgb_512.png

real 0m0.090s
user 0m0.043s
sys 0m0.009s

The C code that performs the DWT Lifting Step runs on the x86_64 dual core and RPi3B is approximately the same.
On x86_64 dual core
time ./liftmain lena_rgb_512.png

real 0m0.356s
user 0m0.209s
sys 0m0.040s

On a RPi3B
./ltime ./liftmain lena_rgb_512.png

real 0m0.380s
user 0m0.230s
sys 0m0.010s

C library, and provides a TFTP server.
Original 256 x 256 image

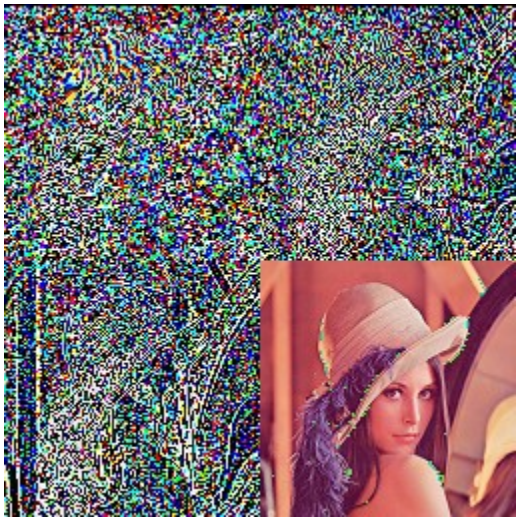
Dwt 1 lvl

H vertical H horizonatal in upper left quadant of image below.

H vertical  L horizonatal in upper right quadant of image below.

L vertical H horizonatal in  lower left quadant of image below.

L vertical L horizonatal in lower right quadant of image below.



1 lvl lifting step dwt.  The default is set to 3 LVLS
line 193 test.c
const int  LVLS = 3;
In the above image line 192 in test.c is set to 1 const int  LVLS = 1;
Dwt 2 lvls

The rgb to yuv further adds to compression of the image
Given rgb.bmp & yuv.bmp both of 196662 bytes.  The arithmetic encoder
wil further reduced by 13.16% for 256x256 & 7.92% for 1024x1024.

256x256
Rationale for RGB to YUV

| | | | |
|---|---|---|---|
| 196662 | 122842 | Arihmetic Encoding rgb | 73820 |
| 196662 | 96958 | Arihmetic Encoding yuv | 99704 |

37.54% RGB
50.70% YUV

13.16% additional compression with YUV

1024x1024
Rationale for RGB to YUV

| | | | |
|---|---|---|---|
| 3145782 | 1247709 | Arihmetic Encoding rgb | 1898073 |
| 3145782 | 998495 | Arihmetic Encoding yuv | 2147287 |

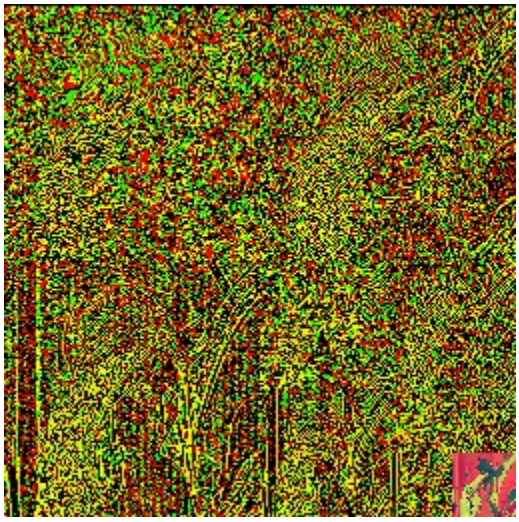
60.34% RGB
68.26% YUV

7.92% additional compression with YUV

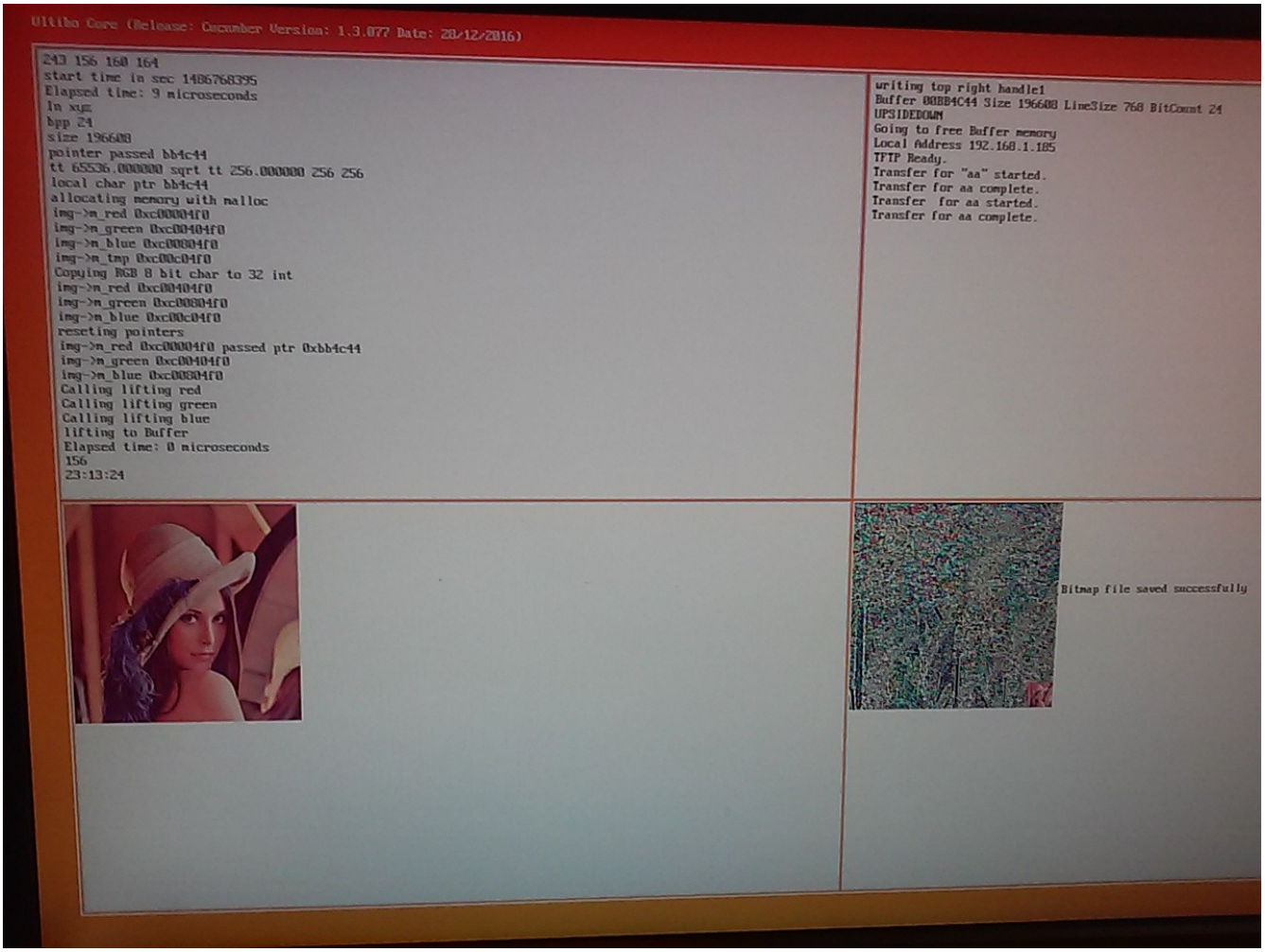
Embedded Block Coding with Optimal Truncation EBCOT will provide improved results over Arithmetic Encoding

Dwt 3 lvls

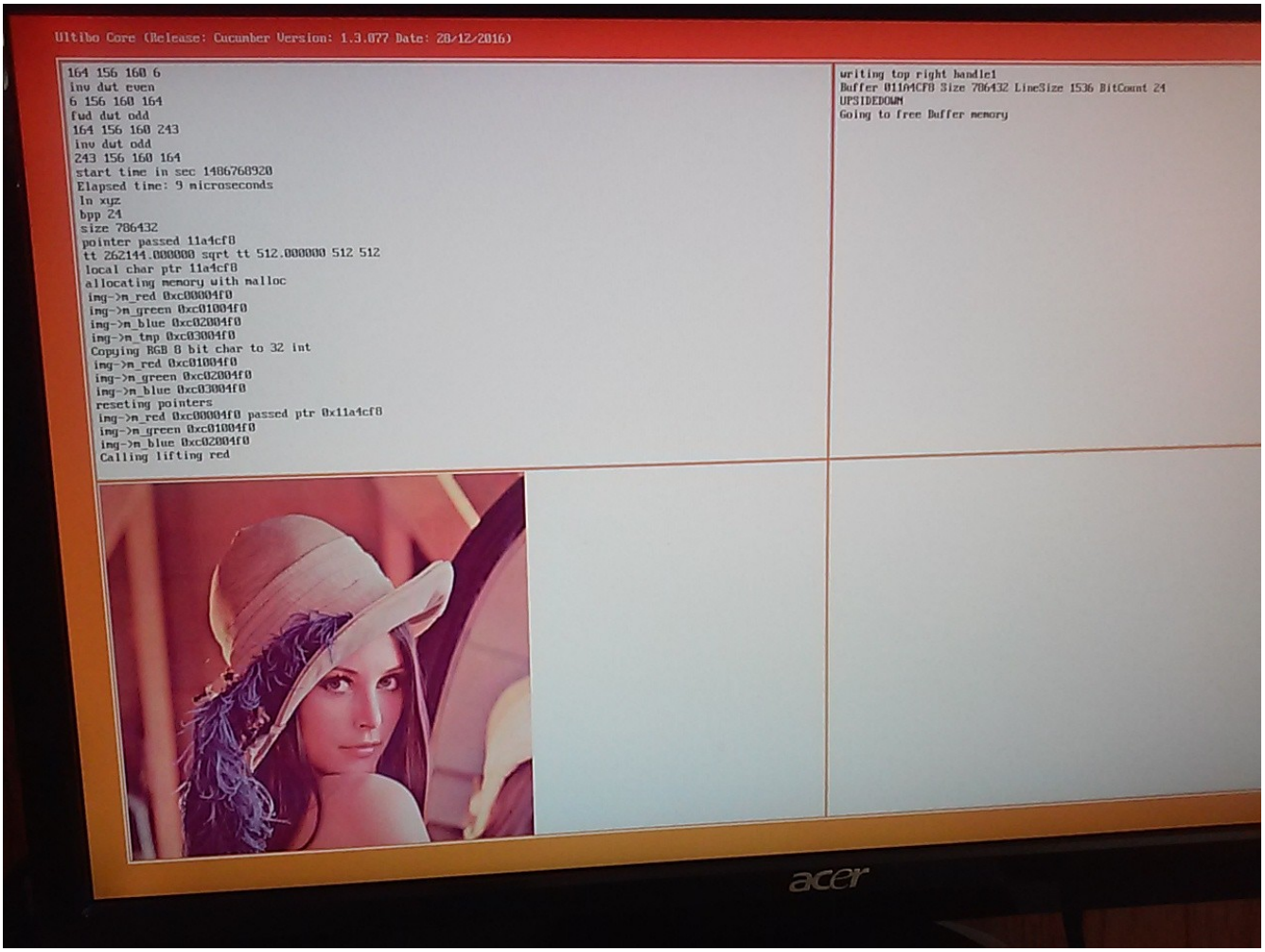
Dwt 3 lvls with rgb to yuv



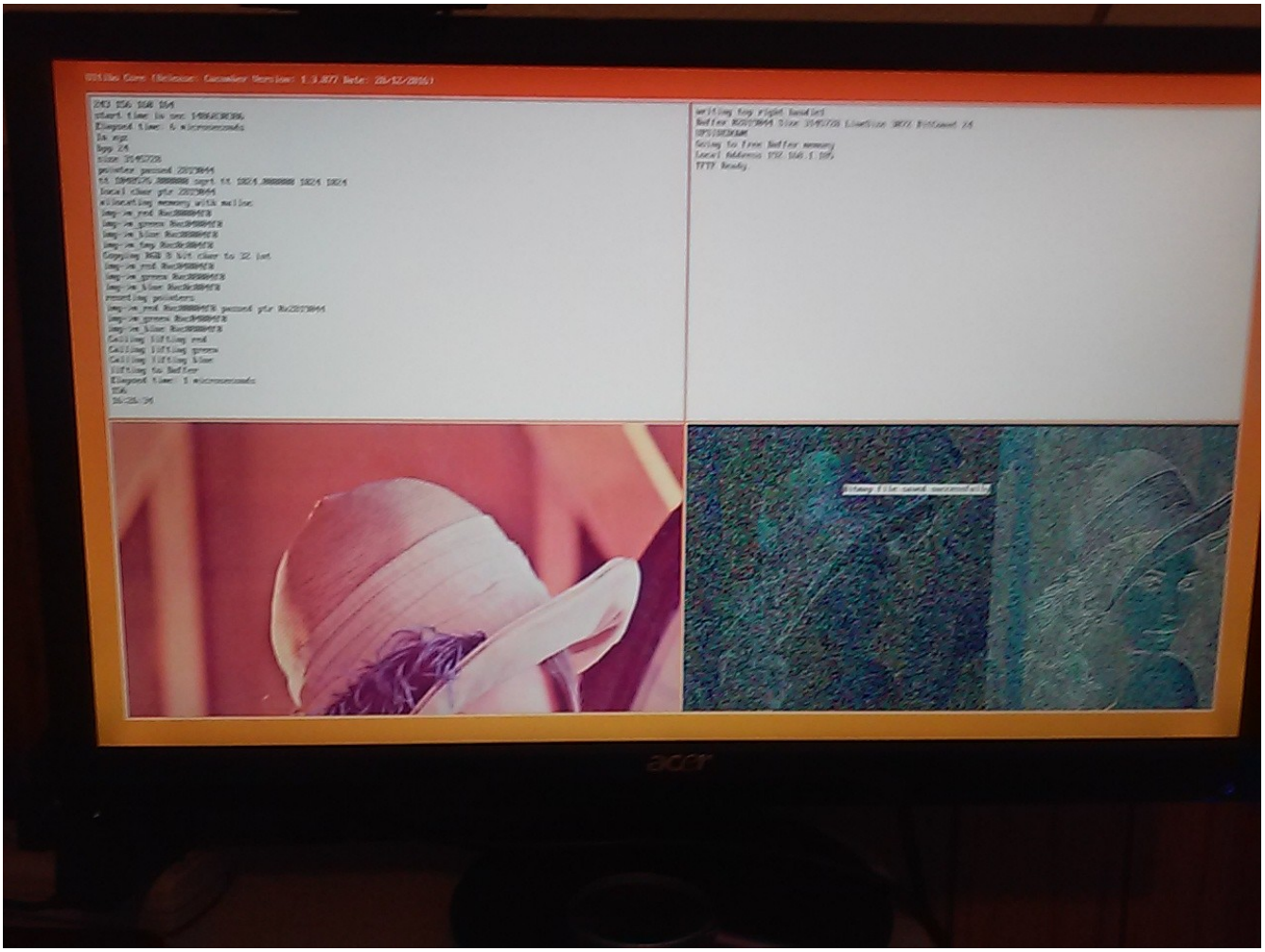
256 x 256


Ultiho Core (Release: Cucumber Version: 1.3.077 Date: 20/12/2016)
243 156 160 164
start time in sec 1486768395
Elapsed time: 9 microseconds
In xgs
bpp 24
size 196608
pointer passed bb4c44
tt 65536.000000 sqrt tt 256.000000 256 256
local char ptr bb4c44
allocating memory with malloc
img->m_red 0xc00004f0
img->m_green 0xc00404f0
img->m_blue 0xc00004f0
img->m_tmp 0xc00c04f0
Copying RGB 8 bit char to 32 int
img->m_red 0xc00404f0
img->m_green 0xc00804f0
img->m_blue 0xc00c04f0
reseting pointers
img->m_red 0xc00004f0 passed ptr 0xbb4c44
img->m_green 0xc00404f0
img->m_blue 0xc00804f0
Calling lifting red
Calling lifting green
Calling lifting blue
lifting to Buffer
Elapsed time: 0 microseconds
156
23:13:24
writing top right handle1
Buffer 00BB4C44 Size 196608 LineSize 768 BitCount 24
UPSIDEDOWN
Going to free Buffer memory
Local Address 192.168.1.185
TFTP Ready.
Transfer for "aa" started.
Transfer for aa complete.
Transfer for aa started.
Transfer for aa complete.
Bitmap file saved successfully

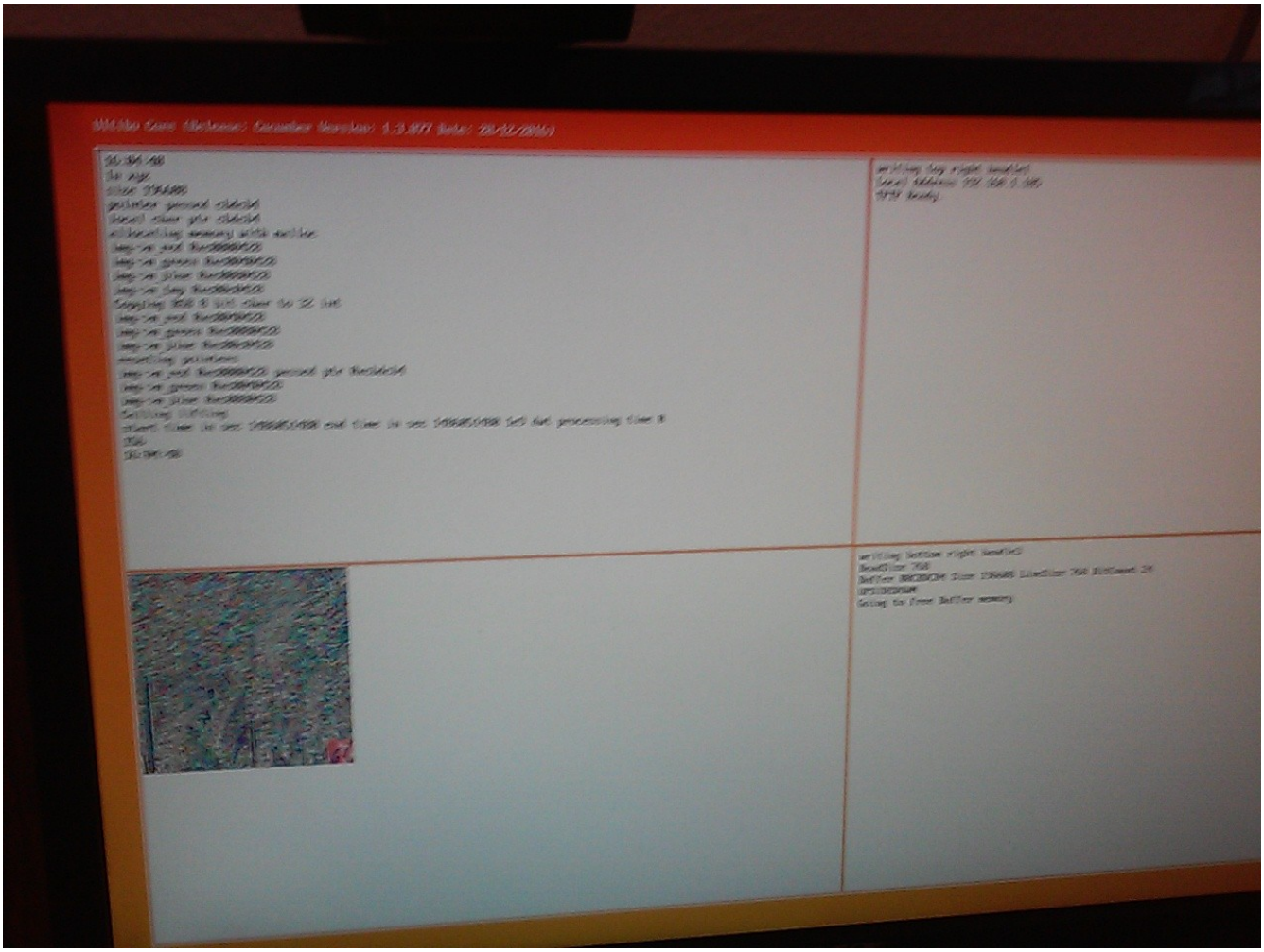
512 x 512


Ultibo Core (Release: Cucumber Version: 1.3.077 Date: 28/12/2016)
164 156 160 6
inv dut even
6 156 160 164
fwd dut odd
164 156 160 243
inv dut odd
243 156 160 164
start time in sec 1486768920
Elapsed time: 9 microseconds
In xyz
bpp 24
size 786432
pointer passed 11a4cf8
tt 262144.000000 sqrt tt 512.000000 512 512
local char ptr 11a4cf8
allocating memory with malloc
img->m_red 0xc00004f0
img->m_green 0xc01004f0
img->m_blue 0xc02004f0
img->m_tmp 0xc03004f0
Copying RGB 8 bit char to 32 int
img->m_red 0xc01004f0
img->m_green 0xc02004f0
img->m_blue 0xc03004f0
reseting pointers
img->m_red 0xc00004f0 passed ptr 0x11a4cf8
img->m_green 0xc01004f0
img->m_blue 0xc02004f0
Calling lifting red
writing top right handle1
Buffer 011A4CF8 Size 786432 LineSize 1536 BitCount 24
UPSIDEDOWN
Going to free Buffer memory
acer

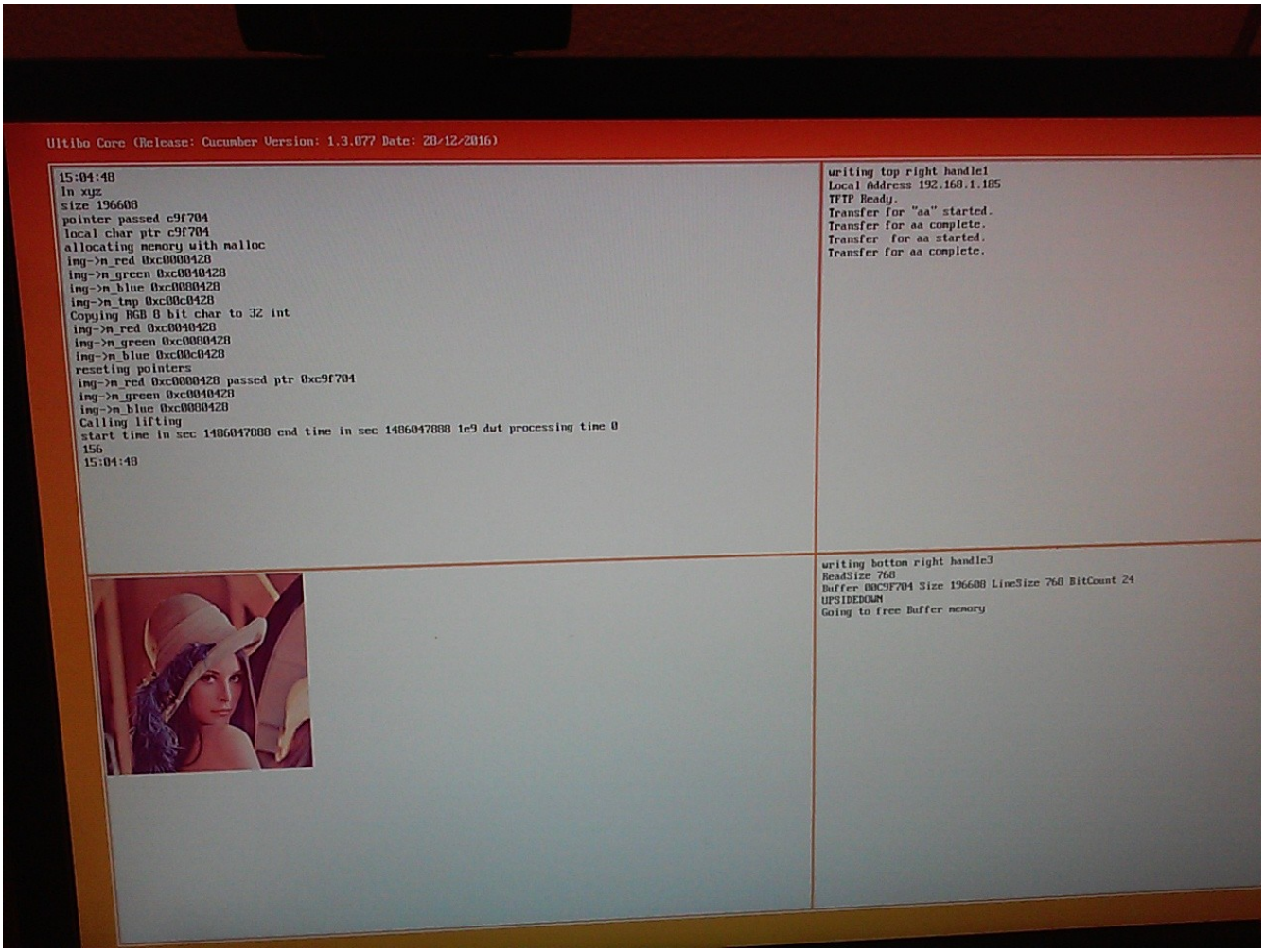
1024 x 1024



xxx



3 lvls lifting step dwt.




In the current test Size & Buffer pointer in Pascal are passed to C. Four areas are created with mallioc. Each area is 256 x 256 which is 65536 locations.

```
typedef struct {
        int     m_w, m_h;
        int     *m_red, *m_green, *m_blue, *m_tmp;
        int     data[1];

} IMAGE, *IMAGEP;
double tt;
if(bp==8) tt = (double)ss;
else tt = (double)(ss/3.0);
int ww, hh;
ww = (int)sqrt(tt);
hh = ww;

        IMAGEP          img;
        //int ww = 256;
```

```
        //int hh = 256;
        printf("allocating memory with malloc \n");
        img = (IMAGEP)malloc(sizeof(IMAGE)+4*ww*hh*sizeof(int));
        img->m_w = ww;
        img->m_h = hh;
        img->m_red   = img->data;
        img->m_green = &img->data[ww*hh];
        img->m_blue  = &img->data[2*ww*hh];
        img->m_tmp  = &img->data[3*ww*hh];
        //printf("the size of malloc %x \n",sizeof(img));
        printf("img->m_red 0x%x \n",img->m_red);
        printf("img->m_green 0x%x \n",img->m_green);
        printf("img->m_blue 0x%x \n",img->m_blue);
        printf("img->m_tmp 0x%x \n",img->m_tmp);
```

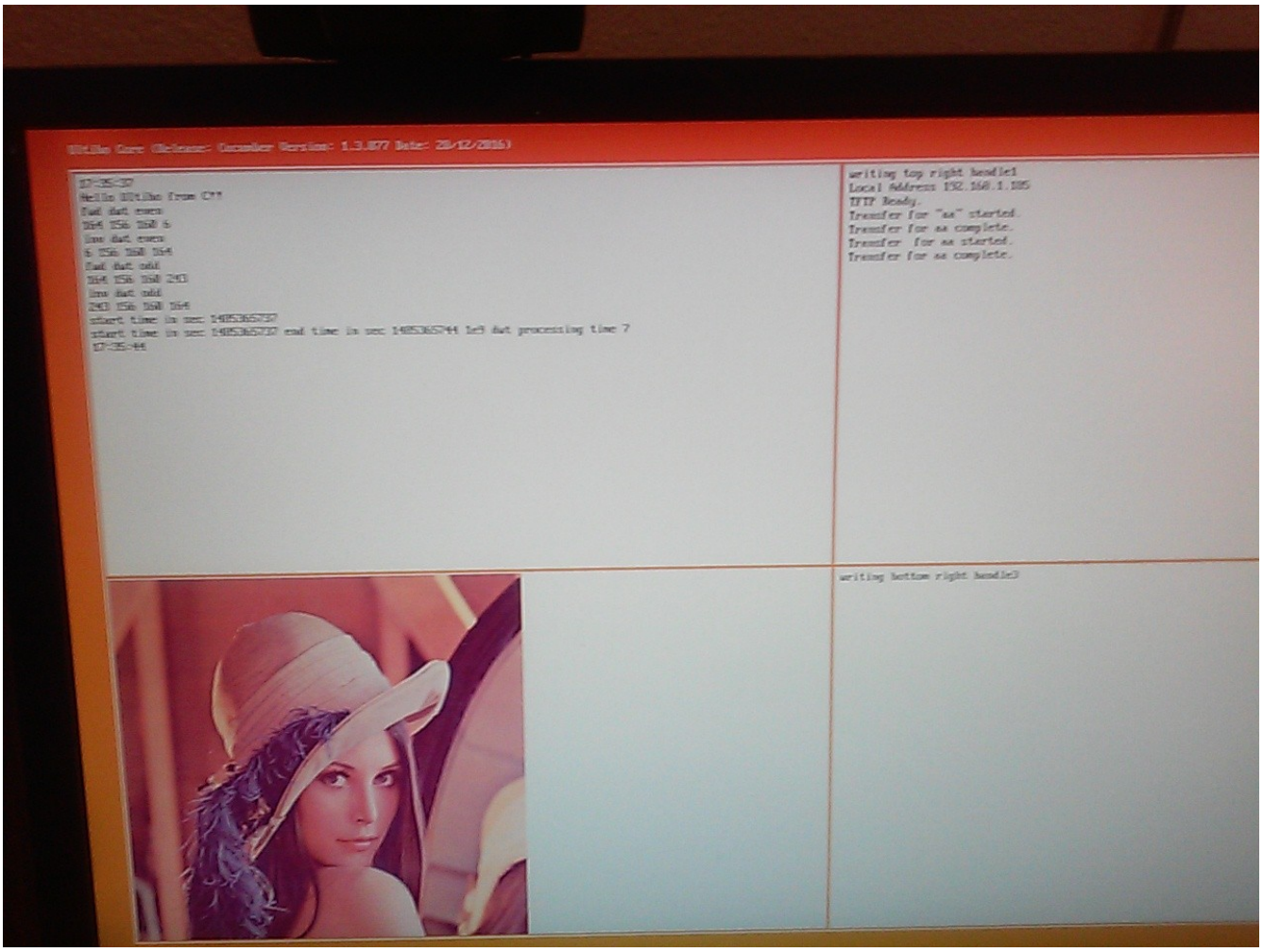
The R 8 bit is copied to the img->m_red 32 bits.
The G 8 bit is copied to the img->m_green 32 bits.
The B 8 bit is copied to the img->m_blue 32 bits.

```
        for (loop=0; loop < ss/3; loop++) {
                *img->m_red = lclip[0];
                lclip++;
                img->m_red++;
                *img->m_green = lclip[0];
                lclip++;
                img->m_green++;
                *img->m_blue = lclip[0];
                lclip++;
                img->m_blue++;
        }

        printf("img->m_red 0x%x  \n",img->m_red);
        printf("img->m_green 0x%x \n",img->m_green);
        printf("img->m_blue 0x%x \n",img->m_blue);

    printf("reseting pointers \n");

        img->m_red   = img->data;
        img->m_green = &img->data[ww*hh];
        img->m_blue  = &img->data[2*ww*hh];

        lclip = (char *)*xx;

        printf("img->m_red 0x%x passed ptr 0x%x\n",img->m_red, &lclip[0]);
        printf("img->m_green 0x%x \n",img->m_green);
        printf("img->m_blue 0x%x \n",img->m_blue);

        printf("Calling lifting red\n");
```

```
        //img->m_red   = img->data;
        lifting(ww, img->m_red, img->m_tmp);
        img->m_tmp  = &img->data[3*ww*hh];
        printf("Calling lifting green\n");

        //img->m_green = &img->data[ww*hh];
        lifting(ww, img->m_green, img->m_tmp);
        img->m_tmp  = &img->data[3*ww*hh];
        printf("Calling lifting blue\n");

        //img->m_blue  = &img->data[2*ww*hh];
        lifting(ww, img->m_blue, img->m_tmp);
        printf("lifting to Buffer\n");
        for (loop=0; loop < ss/3; loop++) {
                lclip[0] = *img->m_red ;
                lclip++;
                img->m_red++;
                lclip[0] = *img->m_green ;
                lclip++;
                img->m_green++;
                lclip[0] = *img->m_blue ;
                lclip++;
                img->m_blue++;
        }

        gettimeofday(&end, NULL);

        seconds  = end.tv_sec  - start.tv_sec;
        useconds = end.tv_usec - start.tv_usec;

        mtime = seconds + useconds;

    printf("Elapsed time: %ld microseconds\n", mtime);
        free(img);
```

In the above example 0xc0000424, 0xc0100424, and 0xc020424 are the memory start.

Now displaying the 256 x 256 image.
Created two units uPointersToC.pas & uBufferToC.pas. This allows function to call the C code Pbuff which xyz in test.c. The Pointer is being passed correctly by the size appears in error.




Processing time on Raspbian takes over 10 times more than Ultibo

./epochtime
time sec 1485366551
start time in sec 1485366551 end time in sec 1485366624 1e9 dwt processing time 73

This only takes 7 sec to perform the 1e9 dwt on Ultibo

Topleft is where the C routine is being called.  Bottomleft is a 512 x 512 bitmap
In the file test.c the contents of lifing.c
In the topright is the tftp process.
# ultibo-tftp

A reasonably quick method of transferring files in an Ultibo project.
It uses Trival FTP based on RFC 1350

Approx upload times around 16 secs for kernel7.img of approx 2.2 MB

https://github.com/pjde/ultibo-tftp.git

```
tftp 192.168.1.185tftp> binary
tftp> put grn-out.32t aa
Sent 1048576 bytes in 5.0 seconds    1.67MBits/s
tftp> get aa bb
Received 1048580 bytes in 4.1 seconds        2.04MBits/s
tftp> quit
```

```
extern  void    singlelift(int rb, int w, int * const ibuf, int * const obuf);
extern  void    ilift(int rb, int w, int * const ibuf, int * const obuf);
extern  void    lifting(int w, int *ibuf, int *tmpbuf);
```

***Updating the kernel7.img Dynamically. First sending the image to a dummy file and retieving the dummy file to a new name.  Testing the md5sum of the retrieved file and kernel7.img sent.***

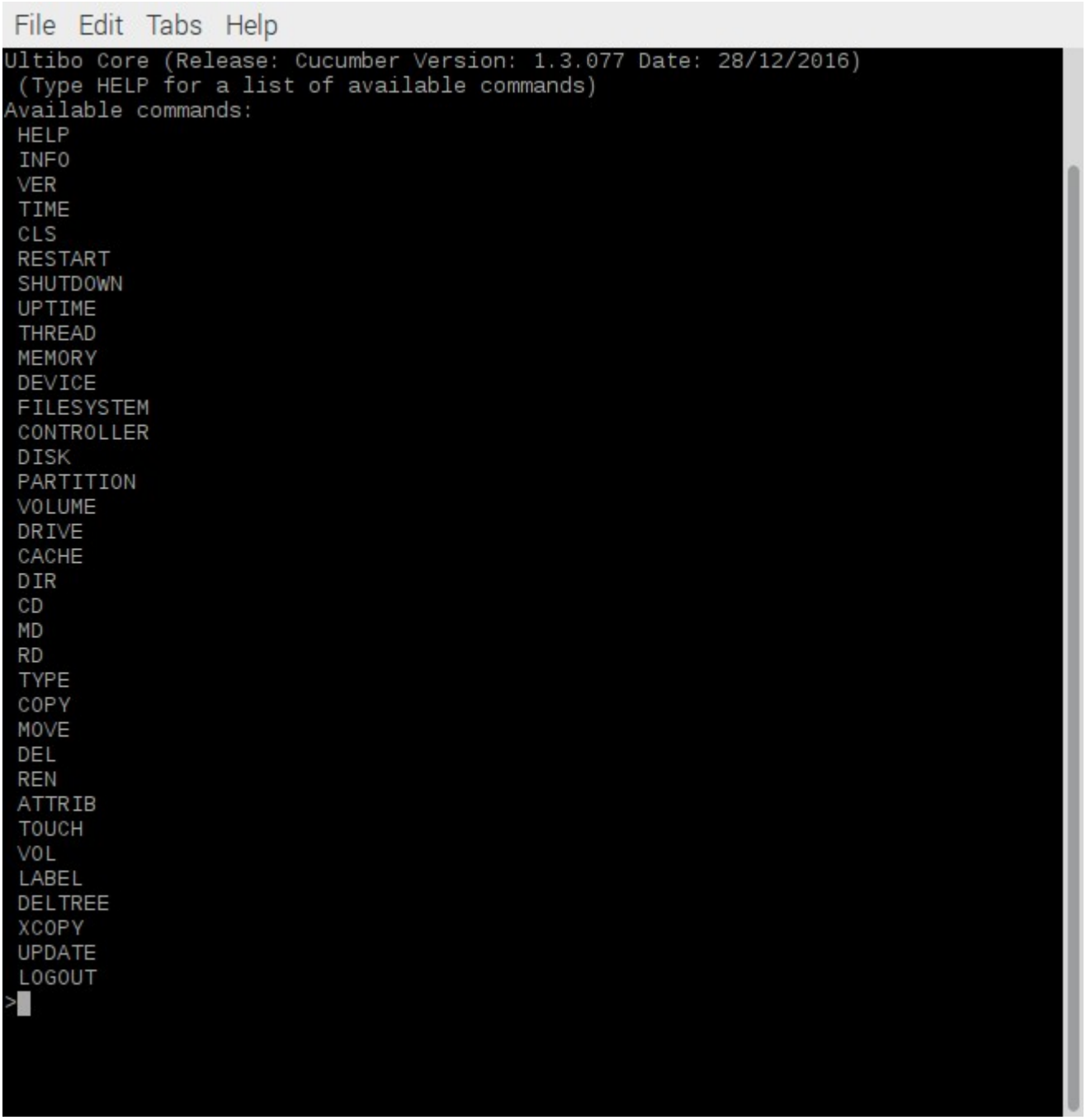
Adding the following uses allows a remote system to telnet into a running Ultibo RPi.

```
 { needed for telnet }
     Shell,
    ShellFilesystem,
    ShellUpdate,
    RemoteShell,
  { needed for telnet }
```

Transfer a new image to Ultibo RPi.

```
pi@raspberrypi3:~/jpeg-2000-test/bare-metal/LibC $ tftp 192.168.1.185
tftp> binary
tftp> put kernel7.img tt
Sent 2548944 bytes in 10.8 seconds
tftp> get tt ss
Received 2548944 bytes in 9.2 seconds
tftp> quit
pi@raspberrypi3:~/jpeg-2000-test/bare-metal/LibC $ md5sum kernel7.img ss
df2a3aaf79570eb16f344655fae65357  kernel7.img
df2a3aaf79570eb16f344655fae65357  ss
```

On a remote system ***"telnet 192.168.1.185"*** where the 192.168.1.185 is the IP of the RPi2B running Ultibo.




Need to delete the kernel7.img.
Need to copy the dummy file to the kernel7.img,
Need to restart for the new kernel to start.

telnet 192.168.1.185
Trying 192.168.1.185...
Connected to 192.168.1.185.
Escape character is '^]'.

```
Ultibo Core (Release: Cucumber Version: 1.3.077 Date: 28/12/2016)
 (Type HELP for a list of available commands)
  Directory of C:\

 22-1-17 17:24:24             17932  bootcode.bin
 22-1-17 17:24:24              6621  fixup.dat
 28-1-17 17:33:54           2548944  kernel7.img
 28-1-17 17:33:54           2548944  tt
 22-1-17 17:24:24            786554  MyBitmap.bmp
 22-1-17 17:24:24           2817796  start.elf
 23-1-17 23:41:54      <DIR>         kernelDbitmap
 9-1-17 18:03:38       <DIR>         kerneldemo
 9-1-17 18:05:00       <DIR>         kernelLibCTest
 11-1-17 17:09:20      <DIR>         kernelTFTP
 12-1-17 11:49:44      <DIR>         kernelUDPServer
 24-1-17 15:39:18                4  lena_rgb_512.bmp
 13-1-17 20:03:06      <DIR>         My Files
 22-1-17 17:23:54      <DIR>         old_files
        7 file(s) 8726795 bytes
        7 dir(s)

C:\>del kernel7.img
  Directory of C:\

 22-1-17 17:24:24             17932  bootcode.bin
 22-1-17 17:24:24              6621  fixup.dat
 28-1-17 17:33:54           2548944  tt
 22-1-17 17:24:24            786554  MyBitmap.bmp
 22-1-17 17:24:24           2817796  start.elf
 23-1-17 23:41:54      <DIR>         kernelDbitmap
 9-1-17 18:03:38       <DIR>         kerneldemo
 9-1-17 18:05:00       <DIR>         kernelLibCTest
 11-1-17 17:09:20      <DIR>         kernelTFTP
 12-1-17 11:49:44      <DIR>         kernelUDPServer
 24-1-17 15:39:18                4  lena_rgb_512.bmp
 13-1-17 20:03:06      <DIR>         My Files
 22-1-17 17:23:54      <DIR>         old_files
        6 file(s) 6177851 bytes
        7 dir(s)

C:\ttopy tt kernel7.img
        1 file(s) copied

  Directory of C:\

 22-1-17 17:24:24             17932  bootcode.bin
 22-1-17 17:24:24              6621  fixup.dat
 28-1-17 17:33:54           2548944  tt
```

```
 22-1-17 17:24:24            786554  MyBitmap.bmp
 22-1-17 17:24:24           2817796  start.elf
 23-1-17 23:41:54      <DIR>         kernelDbitmap
 9-1-17 18:03:38       <DIR>         kerneldemo
 9-1-17 18:05:00       <DIR>         kernelLibCTest
 11-1-17 17:09:20      <DIR>         kernelTFTP
 12-1-17 11:49:44      <DIR>         kernelUDPServer
 24-1-17 15:39:18                4  lena_rgb_512.bmp
 13-1-17 20:03:06      <DIR>         My Files
 22-1-17 17:23:54      <DIR>         old_files
 28-1-17 17:33:54           2548944  kernel7.img
        7 file(s) 8726795 bytes
        7 dir(s)

Restarting in 1000 milliseconds
C:\>
```

This is needed to add the fpc compiler to the PATH.

```
export PATH=/home/pi/ultibo/core/fpc/bin:$PATH
echo $PATH
/
home/pi/ultibo/core/fpc/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/local/games:/usr/games
```

```
arm-none-eabi-gcc -O2 -mabi=aapcs -marm -march=armv7-a -mfpu=vfpv3-d16 -mfloat-abi=hard -c test.c
```

```
arm-none-eabi-ar rcs libtest.a test.o
```

```
fpc -vi -B -Tultibo -Parm -CpARMV7A -WpRPI2B @/home/pi/ultibo/core/fpc/bin/rpi2.cfg -O2 LibCTestRPi2.lpr
```

./build_liftmain.sh compiles lifting.c & liftmain.c --> liftmain

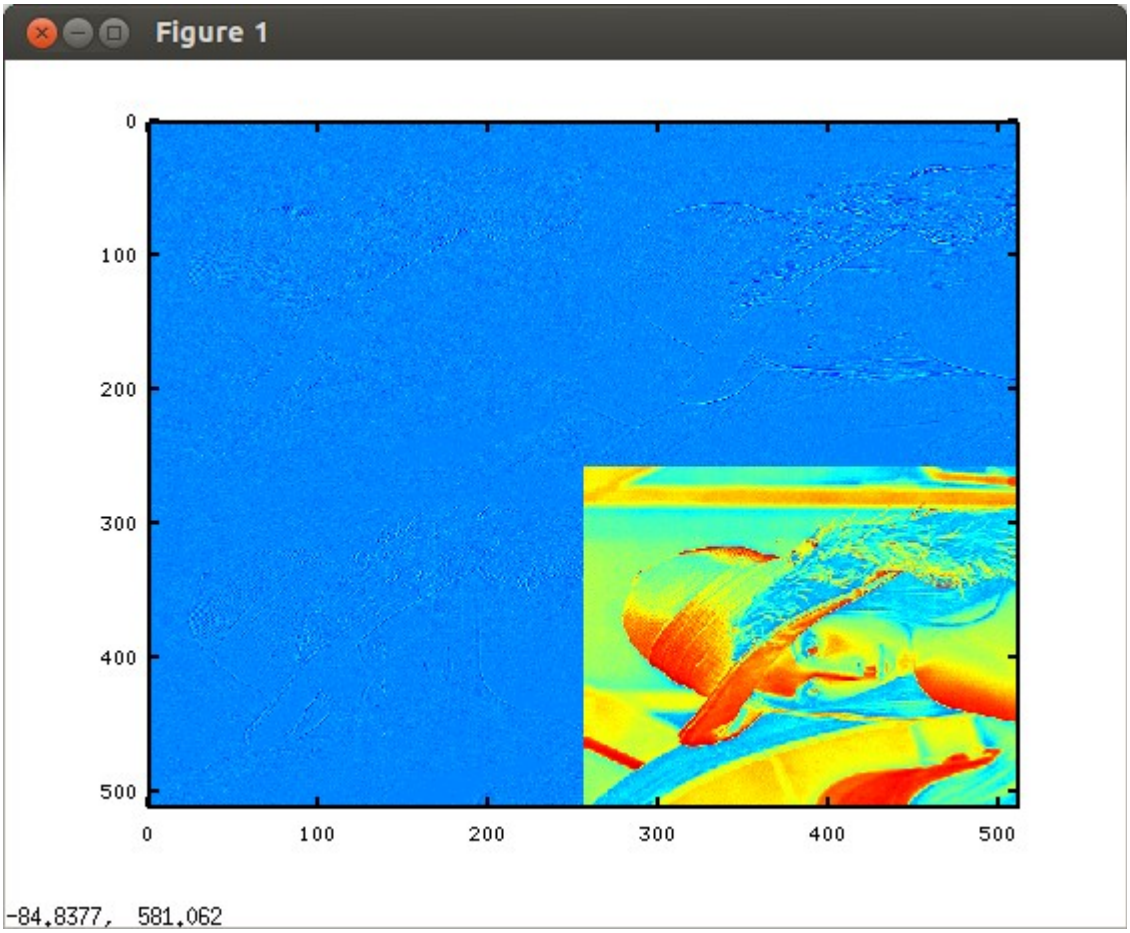
```
iftmain lena_rgb_512.png
        red-out.32t
```

line 101 lifting.c  const int  LVLS = 1;  performs 1 level forward DWT

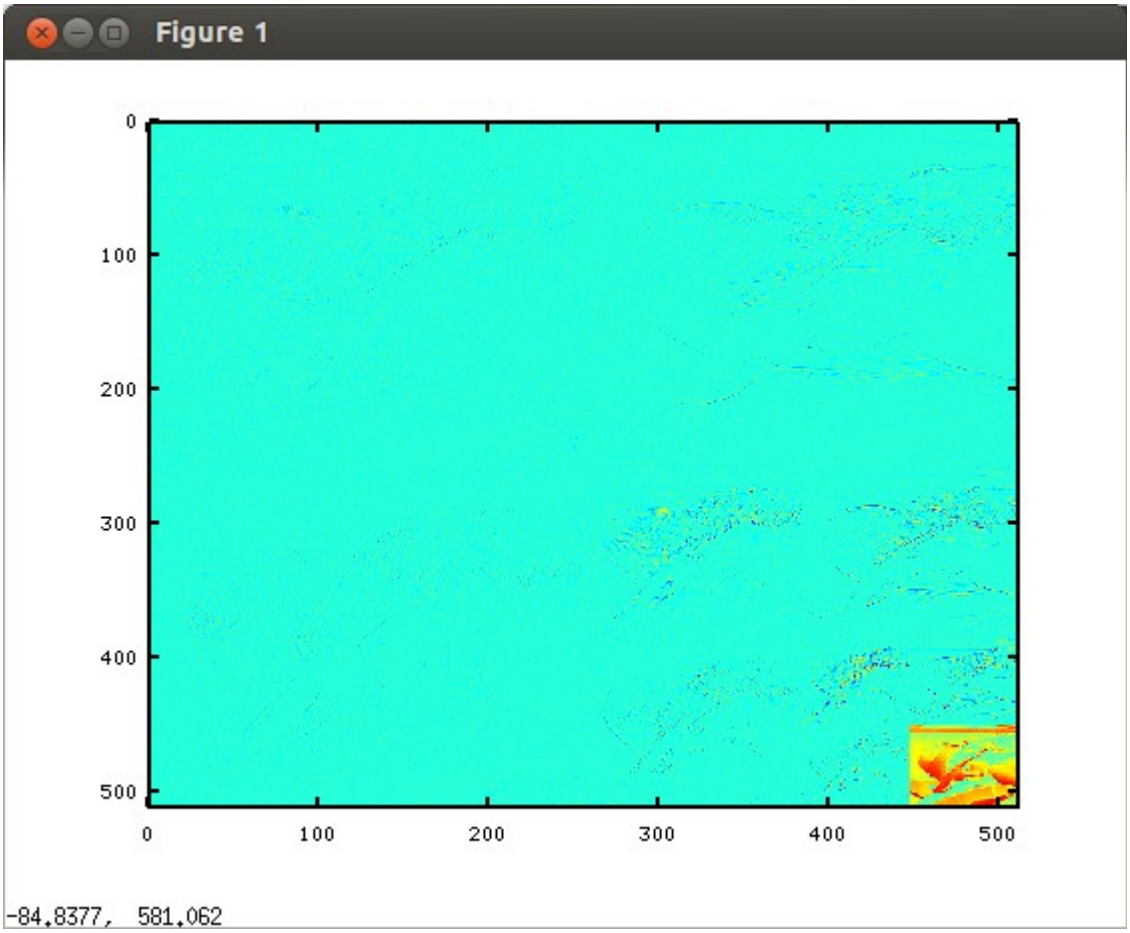
lines 230-246 in lifting.c when commented  does not perform the inverse DWT.

```
/*
        for(lvl=(LVLS-1); lvl>=0; lvl--) {
                int     offset;

                w <<= 1;

                if (lvl)
                        offset = ov[lvl-1];
                else
```

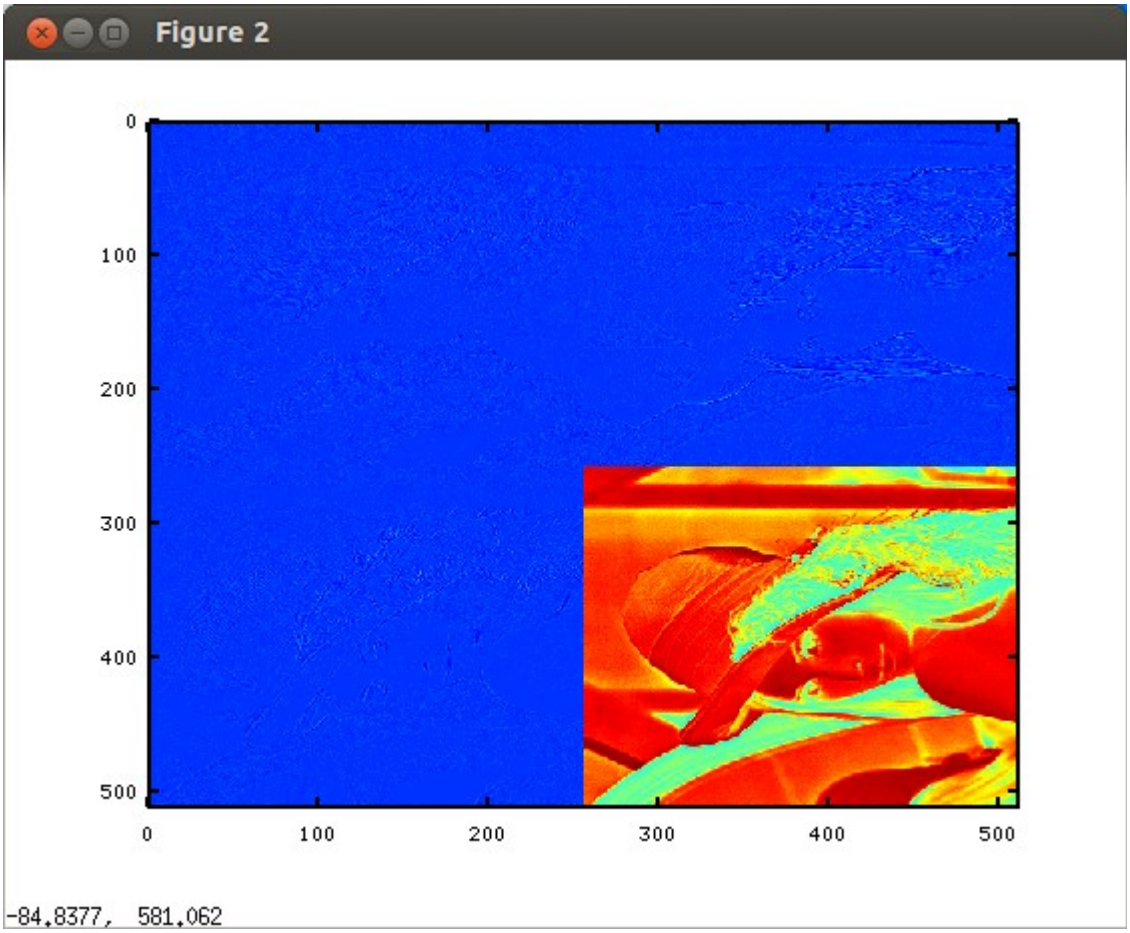
```
                offset = 0;
        ip = &ibuf[offset];
        tp = &tmpbuf[offset];

        ilift(rb, w, ip, tp);
        ilift(rb, w, tp, ip);
}
*/
```



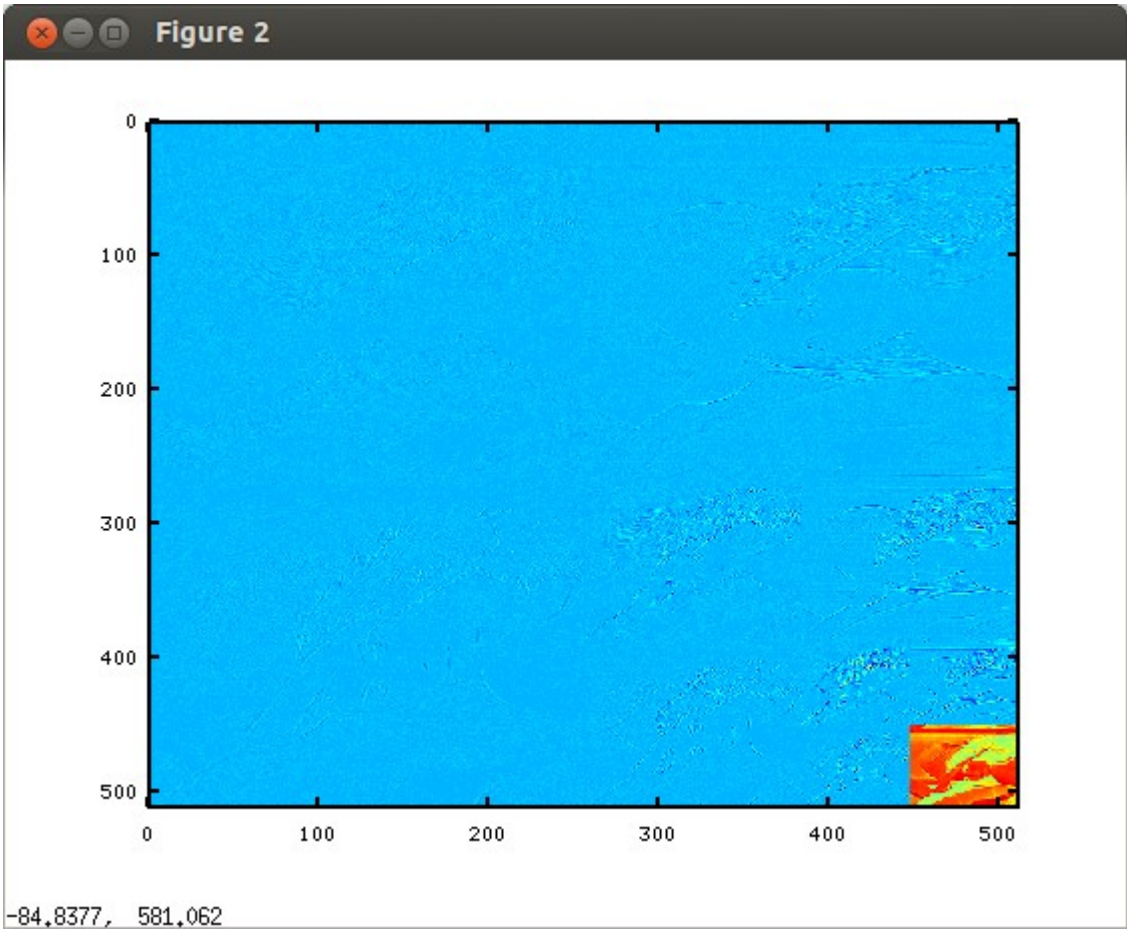

The image above is 1 level forward DWT red subband
The file red-out.32t



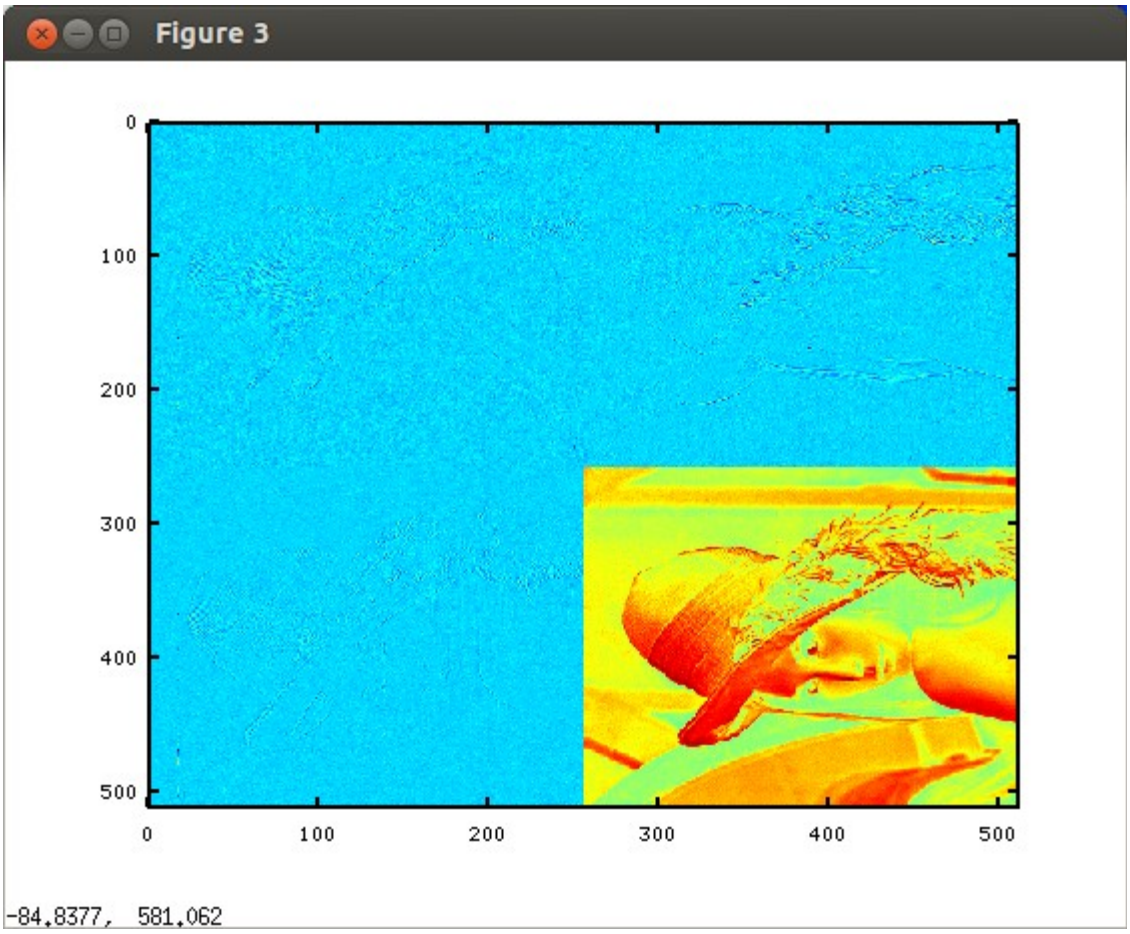

The image above is 3 levels forward DWT red subband
The file red-out.32t



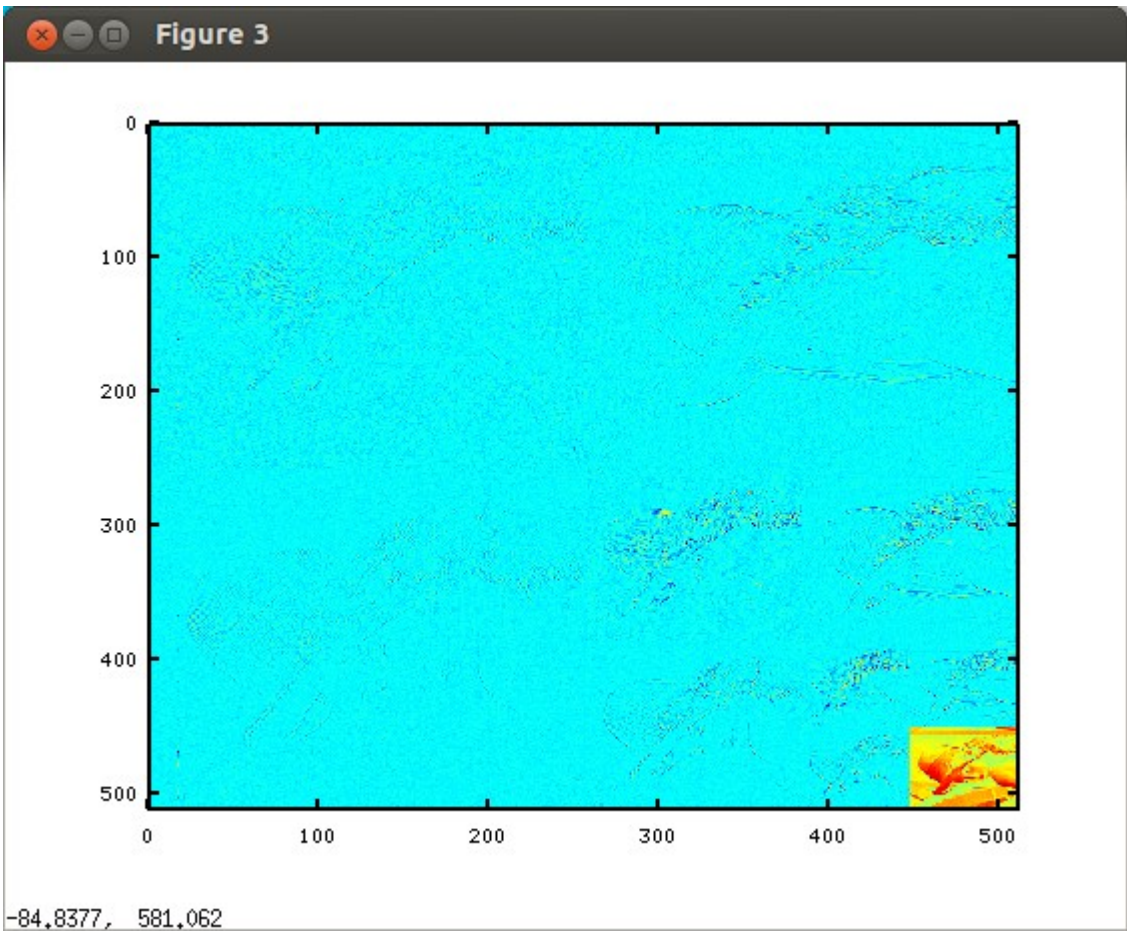

The image above is 1 level forward DWT grn subband
The file grn-out.32t



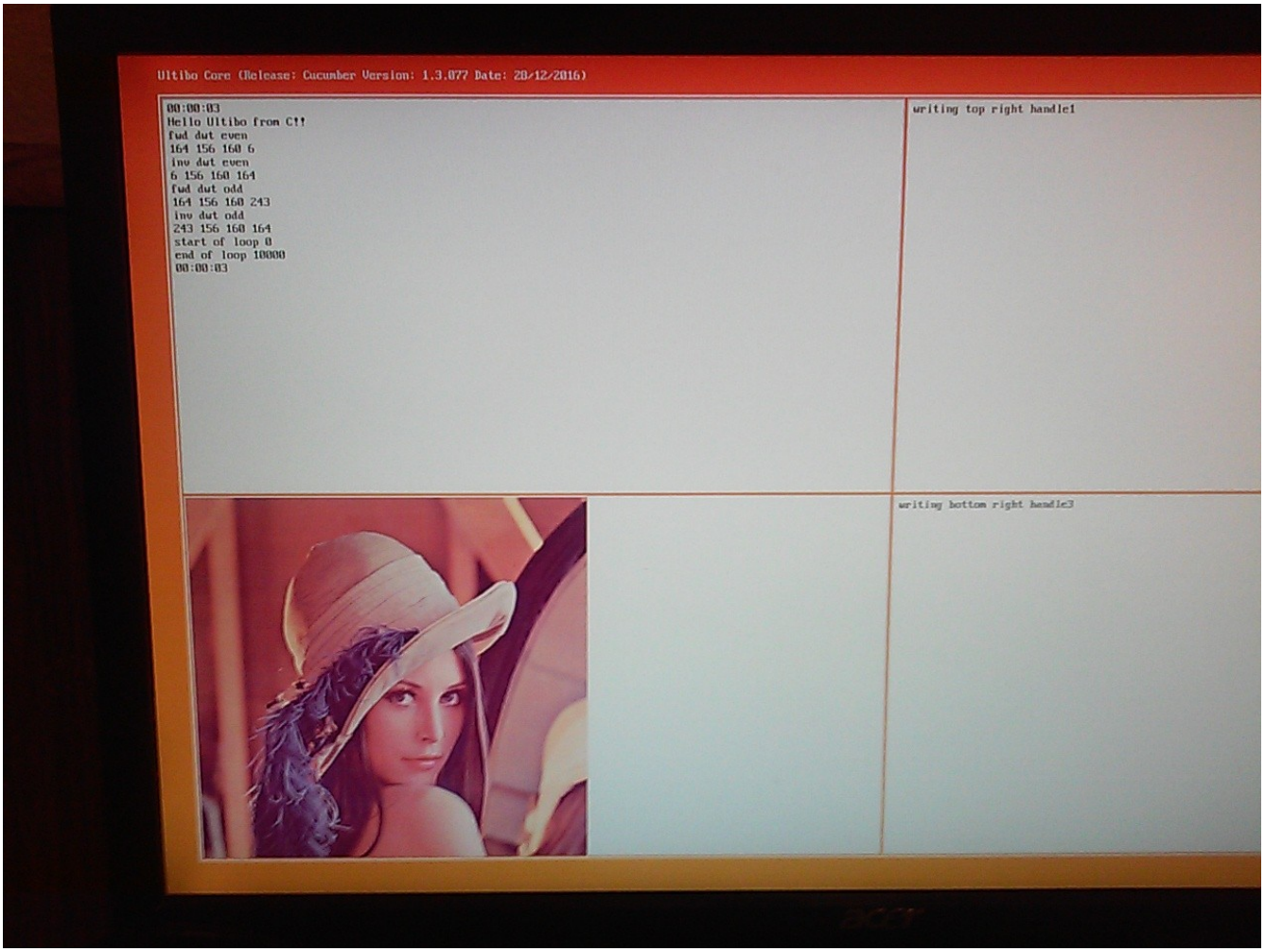

The image above is 3 levels forward DWT grn subband
The file grn-out.32t



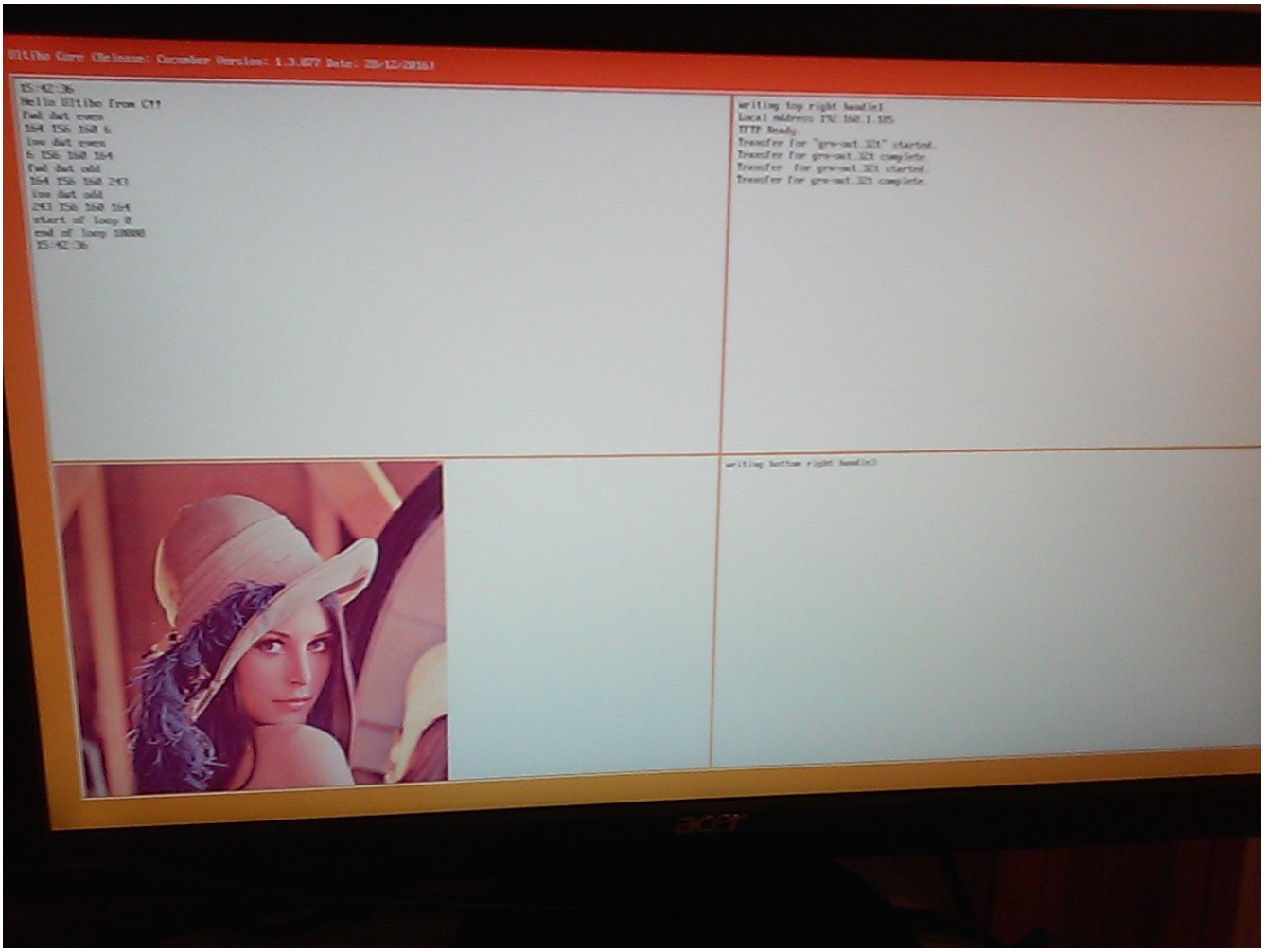

The image above is 1 level forward DWT blu subband
The file blu-out.32t




The image above is 3 levels forward DWT blu subband
The file blu-out.32t



The above image is the running on RPi3B as  compiled for RPi2B on 01/23/17.



Appendix A. CPU infor for x86_64 six core, x86_64 dual core, and RPi3B.

CPUINFO X86_64 six core

processor : 0
vendor_id : AuthenticAMD
cpu family : 16
model : 10
model name : AMD Phenom(tm) II X6 1090T Processor
stepping : 0
microcode : 0x10000dc
cpu MHz : 800.000
cache size : 512 KB
physical id : 0
siblings : 6
core id : 0
cpu cores : 6
apicid : 0
initial apicid : 0
fpu : yes
fpu_exception : yes
cpuid level : 6
wp : yes

flags : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 ht syscall nx mmxext fxsr_opt pdpe1gb rdtscp lm 3dnowext 3dnow constant_tsc rep_good nopl nonstop_tsc extd_apicid aperfmperf pni monitor cx16 popcnt lahf_lm cmp_legacy svm extapic cr8_legacy abm sse4a misalignsse 3dnowprefetch osvw ibs skinit wdt cpb hw_pstate npt lbrv svm_lock nrip_save pausefilter
bogomips : 6400.28
TLB size : 1024 4K pages
clflush size : 64
cache_alignment : 64
address sizes : 48 bits physical, 48 bits virtual
power management: ts ttp tm stc 100mhzsteps hwpstate cpb

processor : 1
vendor_id : AuthenticAMD
cpu family : 16
model : 10
model name : AMD Phenom(tm) II X6 1090T Processor
stepping : 0
microcode : 0x10000dc
cpu MHz : 800.000
cache size : 512 KB
physical id : 0
siblings : 6
core id : 1
cpu cores : 6
apicid : 1
initial apicid : 1
fpu : yes
fpu_exception : yes
cpuid level : 6
wp : yes
flags : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 ht syscall nx mmxext fxsr_opt pdpe1gb rdtscp lm 3dnowext 3dnow constant_tsc rep_good nopl nonstop_tsc extd_apicid aperfmperf pni monitor cx16 popcnt lahf_lm cmp_legacy svm extapic cr8_legacy abm sse4a misalignsse 3dnowprefetch osvw ibs skinit wdt cpb hw_pstate npt lbrv svm_lock nrip_save pausefilter
bogomips : 6400.28
TLB size : 1024 4K pages
clflush size : 64
cache_alignment : 64
address sizes : 48 bits physical, 48 bits virtual
power management: ts ttp tm stc 100mhzsteps hwpstate cpb

processor : 2
vendor_id : AuthenticAMD
cpu family : 16
model : 10
model name : AMD Phenom(tm) II X6 1090T Processor
stepping : 0

microcode : 0x10000dc
cpu MHz : 800.000
cache size : 512 KB
physical id : 0
siblings : 6
core id : 2
cpu cores : 6
apicid : 2
initial apicid : 2
fpu : yes
fpu_exception : yes
cpuid level : 6
wp : yes
flags : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 ht syscall nx mmxext fxsr_opt pdpe1gb rdtscp lm 3dnowext 3dnow constant_tsc rep_good nopl nonstop_tsc extd_apicid aperfmperf pni monitor cx16 popcnt lahf_lm cmp_legacy svm extapic cr8_legacy abm sse4a misalignsse 3dnowprefetch osvw ibs skinit wdt cpb hw_pstate npt lbrv svm_lock nrip_save pausefilter
bogomips : 6400.28
TLB size : 1024 4K pages
clflush size : 64
cache_alignment : 64
address sizes : 48 bits physical, 48 bits virtual
power management: ts ttp tm stc 100mhzsteps hwpstate cpb

processor : 3
vendor_id : AuthenticAMD
cpu family : 16
model : 10
model name : AMD Phenom(tm) II X6 1090T Processor
stepping : 0
microcode : 0x10000dc
cpu MHz : 800.000
cache size : 512 KB
physical id : 0
siblings : 6
core id : 3
cpu cores : 6
apicid : 3
initial apicid : 3
fpu : yes
fpu_exception : yes
cpuid level : 6
wp : yes
flags : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 ht syscall nx mmxext fxsr_opt pdpe1gb rdtscp lm 3dnowext 3dnow constant_tsc rep_good nopl nonstop_tsc extd_apicid aperfmperf pni monitor cx16 popcnt lahf_lm cmp_legacy svm extapic cr8_legacy abm sse4a misalignsse 3dnowprefetch osvw ibs skinit wdt cpb hw_pstate npt lbrv svm_lock nrip_save pausefilter

bogomips : 6400.28
TLB size : 1024 4K pages
clflush size : 64
cache_alignment : 64
address sizes : 48 bits physical, 48 bits virtual
power management: ts ttp tm stc 100mhzsteps hwpstate cpb

processor : 4
vendor_id : AuthenticAMD
cpu family : 16
model : 10
model name : AMD Phenom(tm) II X6 1090T Processor
stepping : 0
microcode : 0x10000dc
cpu MHz : 800.000
cache size : 512 KB
physical id : 0
siblings : 6
core id : 4
cpu cores : 6
apicid : 4
initial apicid : 4
fpu : yes
fpu_exception : yes
cpuid level : 6
wp : yes
flags : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 ht syscall nx mmxext fxsr_opt pdpe1gb rdtscp lm 3dnowext 3dnow constant_tsc rep_good nopl nonstop_tsc extd_apicid aperfmperf pni monitor cx16 popcnt lahf_lm cmp_legacy svm extapic cr8_legacy abm sse4a misalignsse 3dnowprefetch osvw ibs skinit wdt cpb hw_pstate npt lbrv svm_lock nrip_save pausefilter
bogomips : 6400.28
TLB size : 1024 4K pages
clflush size : 64
cache_alignment : 64
address sizes : 48 bits physical, 48 bits virtual
power management: ts ttp tm stc 100mhzsteps hwpstate cpb

processor : 5
vendor_id : AuthenticAMD
cpu family : 16
model : 10
model name : AMD Phenom(tm) II X6 1090T Processor
stepping : 0
microcode : 0x10000dc
cpu MHz : 800.000
cache size : 512 KB
physical id : 0
siblings : 6

core id : 5
cpu cores : 6
apicid : 5
initial apicid : 5
fpu : yes
fpu_exception : yes
cpuid level : 6
wp : yes
flags : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 ht syscall nx mmxext fxsr_opt pdpe1gb rdtscp lm 3dnowext 3dnow constant_tsc rep_good nopl nonstop_tsc extd_apicid aperfmperf pni monitor cx16 popcnt lahf_lm cmp_legacy svm extapic cr8_legacy abm sse4a misalignsse 3dnowprefetch osvw ibs skinit wdt cpb hw_pstate npt lbrv svm_lock nrip_save pausefilter
bogomips : 6400.28
TLB size : 1024 4K pages
clflush size : 64
cache_alignment : 64
address sizes : 48 bits physical, 48 bits virtual
power management: ts ttp tm stc 100mhzsteps hwpstate cpb

CPUINFO x86_64 dual core
processor : 0
vendor_id : AuthenticAMD
cpu family : 15
model : 36
model name : AMD Turion(tm) 64 Mobile Technology ML-32
stepping : 2
cpu MHz : 800.000
cache size : 512 KB
fpu : yes
fpu_exception : yes
cpuid level : 1
wp : yes
flags : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 syscall nx mmxext fxsr_opt lm 3dnowext 3dnow rep_good nopl pni lahf_lm
bogomips : 1591.68
TLB size : 1024 4K pages
clflush size : 64
cache_alignment : 64
address sizes : 40 bits physical, 48 bits virtual
power management: ts fid vid ttp tm stc

CPUINFO RPi3B
processor : 0
model name : ARMv7 Processor rev 4 (v7l)
BogoMIPS : 76.80
Features : half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva idivt vfpd32 lpae evtstrm crc32

CPU implementer : 0x41
CPU architecture: 7
CPU variant : 0x0
CPU part : 0xd03
CPU revision : 4

processor : 1
model name : ARMv7 Processor rev 4 (v7l)
BogoMIPS : 76.80
Features : half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva idivt vfpd32 lpae evtstrm crc32
CPU implementer : 0x41
CPU architecture: 7
CPU variant : 0x0
CPU part : 0xd03
CPU revision : 4

processor : 2
model name : ARMv7 Processor rev 4 (v7l)
BogoMIPS : 76.80
Features : half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva idivt vfpd32 lpae evtstrm crc32
CPU implementer : 0x41
CPU architecture: 7
CPU variant : 0x0
CPU part : 0xd03
CPU revision : 4

processor : 3
model name : ARMv7 Processor rev 4 (v7l)
BogoMIPS : 76.80
Features : half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva idivt vfpd32 lpae evtstrm crc32
CPU implementer : 0x41
CPU architecture: 7
CPU variant : 0x0
CPU part : 0xd03
CPU revision : 4

Hardware : BCM2709
Revision : a22082
Serial : 0000000071ada62c
Status: Several Ultibo examples have been merged into a 4 program which displays a bitmap, calls a