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

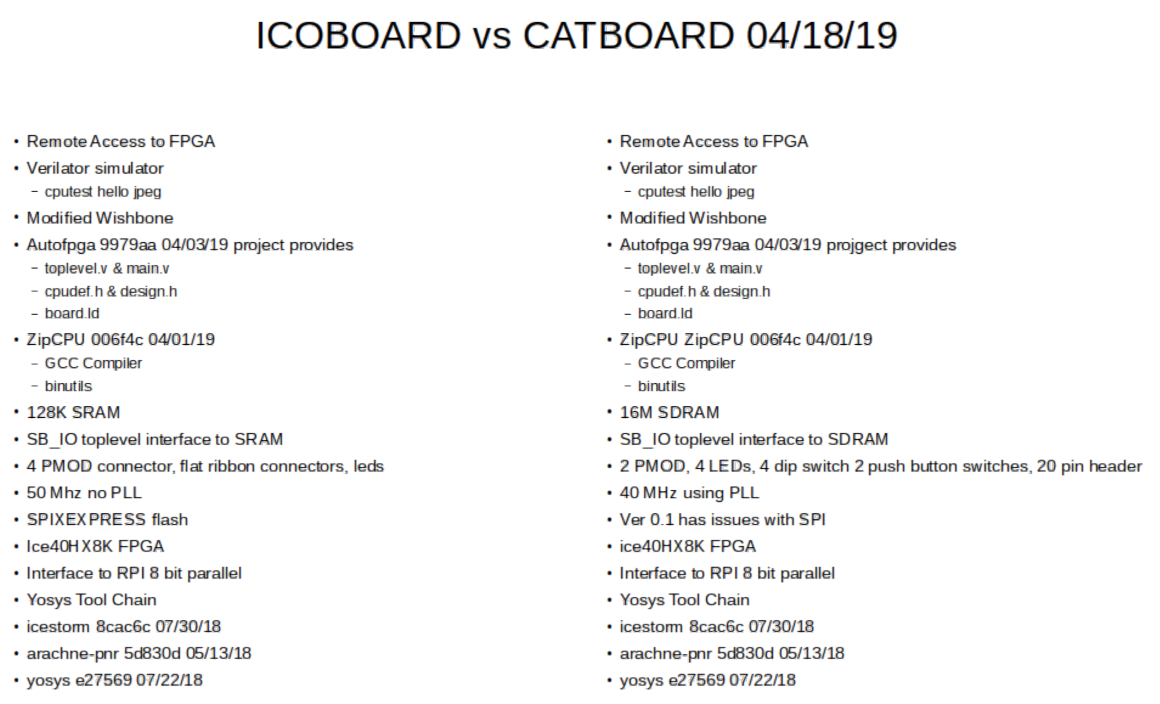
# *Adapting the ICOBOARD icozip to the CATBOARD catzip 04/18/19*

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

*The README.md https://github.com/develone/catzip*

*"git clone https://github.com/develone/catzip.git"*
*"cd catzip"*
*". myenv.sh"*
*"make"*
*This step will take several minutes.*

*Icoboard vs Catboard*



ICOBOARD vs CATBOARD 04/18/19

- Remote Access to FPGA
- Verilator simulator
  - cputest hello jpeg
- Modified Wishbone
- Autofpga 9979aa 04/03/19 project provides
  - toplevel.v & main.v
  - cpudef.h & design.h
  - board.ld
- ZipCPU 006f4c 04/01/19
  - GCC Compiler
  - binutils
- 128K SRAM
- SB_IO toplevel interface to SRAM
- 4 PMOD connector, flat ribbon connectors, leds
- 50 Mhz no PLL
- SPIXEXPRESS flash
- Ice40HX8K FPGA
- Interface to RPI 8 bit parallel
- Yosys Tool Chain
- icestorm 8cac6c 07/30/18
- arachne-pnr 5d830d 05/13/18
- yosys e27569 07/22/18

- Remote Access to FPGA
- Verilator simulator
  - cputest hello jpeg
- Modified Wishbone
- Autofpga 9979aa 04/03/19 projgect provides
  - toplevel.v & main.v
  - cpudef.h & design.h
  - board.ld
- ZipCPU ZipCPU 006f4c 04/01/19
  - GCC Compiler
  - binutils
- 16M SDRAM
- SB_IO toplevel interface to SDRAM
- 2 PMOD, 4 LEDs, 4 dip switch 2 push button switches, 20 pin header
- 40 MHz using PLL
- Ver 0.1 has issues with SPI
- ice40HX8K FPGA
- Interface to RPI 8 bit parallel
- Yosys Tool Chain
- icestorm 8cac6c 07/30/18
- arachne-pnr 5d830d 05/13/18
- yosys e27569 07/22/18

*In shell Used to communicate with FPGA and Rpi3 C prograns*

```
“pi@mypi3-1:~/testbuilds/catzip/sw/host $ ./arm-netpport”
Listening on port 8363
Listening on port 8364
```

*In shell arm- commands are executed.*

*arm-wbregs Peek & Poke into FPGA*

*Displays the date the bin file created*

```
./arm-wbregs version
00800010 ( VERSION) : [....] 20190415
```

*Writes & Reads Sdram data at address 0x01000000 & 01fffffffc*

```
pi@mypi3-1:~/testbuilds/catzip/sw/host $ ./arm-wbregs 0x01000000
0x55aaaa55
01000000 (   SDRAM)-> 55aaaa55
pi@mypi3-1:~/testbuilds/catzip/sw/host $ ./arm-wbregs 0x01000000
01000000 (   SDRAM) : [U..U] 55aaaa55
pi@mypi3-1:~/testbuilds/catzip/sw/host $ ./arm-wbregs 0x01ffffc 0xaa5555aa
01ffffc (        )-> aa5555aa
pi@mypi3-1:~/testbuilds/catzip/sw/host $ ./arm-wbregs 0x01ffffc
01ffffc (        ) : [.UU.] aa5555aa
```

*Reads data from Sdram writing to a file*

```
arm-rdsdram  dwt.bin
```

*Writes data from a file to Sdram*

```
arm-wrsdram rgb_pack.bin
```

*Provides the details to create the catzip/rtl/catzip/catzip.bin used to program the catboard fpga.*

```
sudo config_cat ../../rtl/catzip/catzip.bin

OK: GPIO 25 exported
OK: GPIO 17 exported
OK: GPIO 22 exported

OK: SPI driver loaded

Setting GPIO directions
out
out
in
Setting output to low
0
Reseting FPGA
0
1
Checking DONE pin
0
Continuing with configuration procedure
263+1 records in
```

```
263+1 records out
135100 bytes (135 kB, 132 KiB) copied, 0.0344357 s, 3.9 MB/s
Setting output to high
1
Checking DONE pin
1
```

See the documents in catzip/doc folder that desscrible the cammands to demonstrate the C programs HX8K_TESTS.odt, RPi3B_TESTS.odt, and SIMULATOR_TESTS.odt

Below is output of the jpeg.c compiled for zipcpu

```
 pi@mypi3-1:~/testbuilds/tstmul/catzip/sw/host $ ./arm-netpport
Listening on port 8363
Listening on port 8364
```

## jpeg.c

```
#include <stdio.h>
#include <stdlib.h>
#include "board.h"
#include "lifting.h"
#define SDRAM 0x800000
#define imgsize 256

void zip_clear_sdram(int *imbuf) {
int val,i;

val = 0;



for(i=0; i<256*256*12; i++) {
	*imbuf++ = val;
}
}
//0xc0024	786468		0x008093b0
struct PTRs {
	int inpbuf[65536];
	int act[131072];
	int flag;
	int w;
	int h;
	int *img;
	int *red;
	//int *fwd_inv;
	int *grn;
	int *blu;
	int *alt;
```

```
} ptrs;

void split(int ff, int loop, int *ibuf, int *obuf) {

   int   *ip = ibuf;
   int *op = obuf;
   int i,sp,x,y,z;
         for(i=0;i<loop;i++) {
                  x = *ip;
                  if (ff == 0) y = 0x1ff00000;
                  if (ff == 1) y = 0x7fc00;
                  if (ff == 2) y = 0x1ff;
                  z = x & y;
                  //printf("x = 0x%x z = 0x%x y = 0x%x ",x,z,y);
                  if (ff == 0) sp = z>>20;
                  if (ff == 1) sp = z>>8;
                  if (ff == 2) sp = z;
                  *op = sp;
                  if (i <= 3) printf("x = 0x%x sp = 0x%x z = 0x%x\n",x,sp,z);
                  if (i > 65532) printf("x = 0x%x sp = 0x%x z = 0x%x\n",x,sp,z);
                  ip++;
                  op++;
         }

}
int main(int argc, char **argv) {

         ptrs.w = 256;
         ptrs.h = 256;
         int *im_s_ptr;
         int i,x,y,z,ur,ug,ub;
         int *fwd_inv;
         ptrs.img = (int *)&ptrs.inpbuf;
         printf("w = 0x%x h = 0x%x\n",ptrs.w,ptrs.h);

         printf("ptrs-->img = 0x%x\n",ptrs.img);

         //split red
         ptrs.red = (int *)&ptrs.act;
         ptrs.alt = (int *)&ptrs.act[65536];
         ptrs.flag = 0;
         i = 65535;
         ptrs.red = ptrs.act;
         split(ptrs.flag, i, ptrs.img, ptrs.red);
         printf("back from split start of dwt \n");

         for(i=0;i<5;i++) printf("0x%x \n",ptrs.red[i]);
         for(i=0;i<5;i++) printf("0x%x \n",ptrs.alt[i]);
         *fwd_inv = 1;
   printf("0x%x 0x%x 0x%x 0x%x  \n", ptrs.w,ptrs.red,ptrs.alt,*fwd_inv);
```

```
        lifting(ptrs.w,ptrs.red,ptrs.alt,fwd_inv);
        printf("back from dwt\n");



}
```

*The 256 x 256 image was packed into file "rgb_pack.bin"*

*The following three images were extracted from the catboard following the reading of the file rgb_pack.bin and split into R G B in preparation to perform the lifting step.*

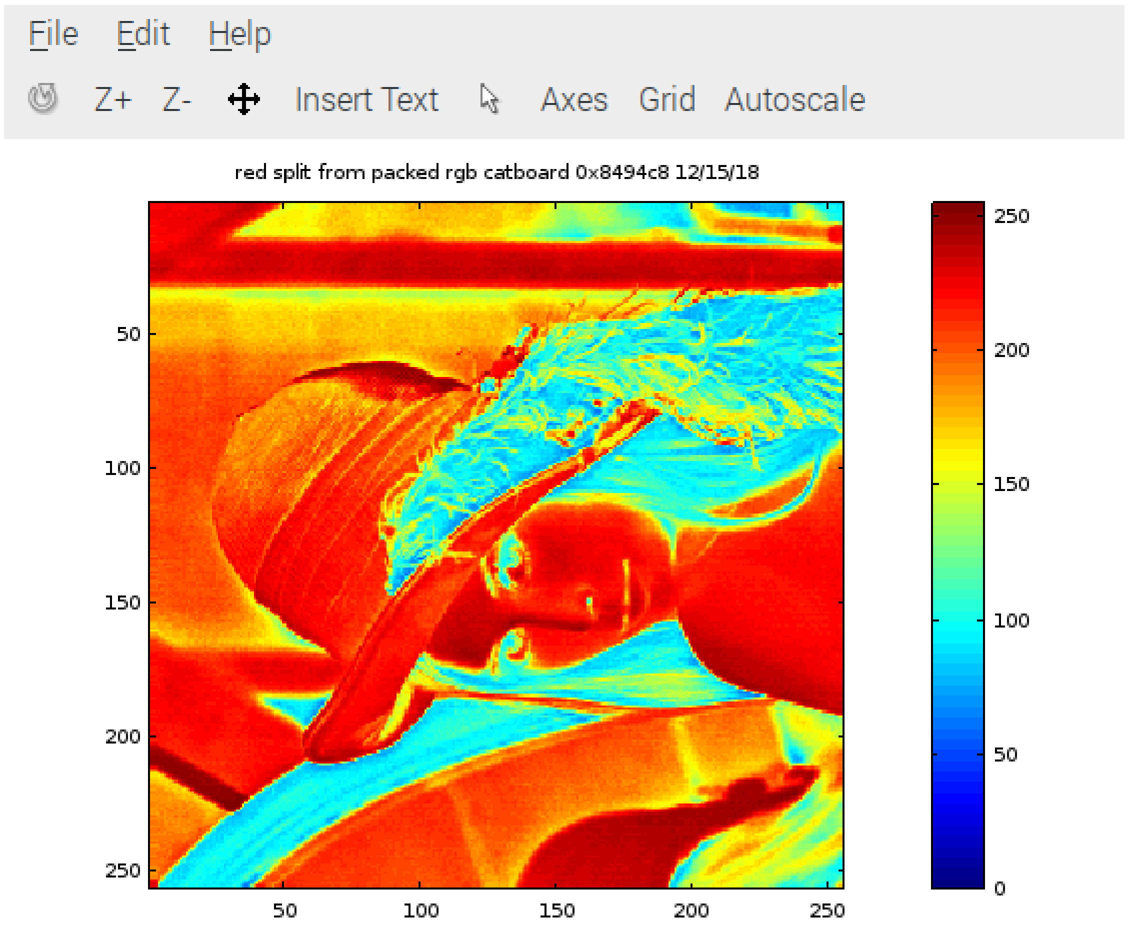
*Note:*

~~*12/2/18 Currently since the HX8K does not have implement the mul & div instructions*~~
~~*the lifting step halts and does not complete the lifting step.*~~

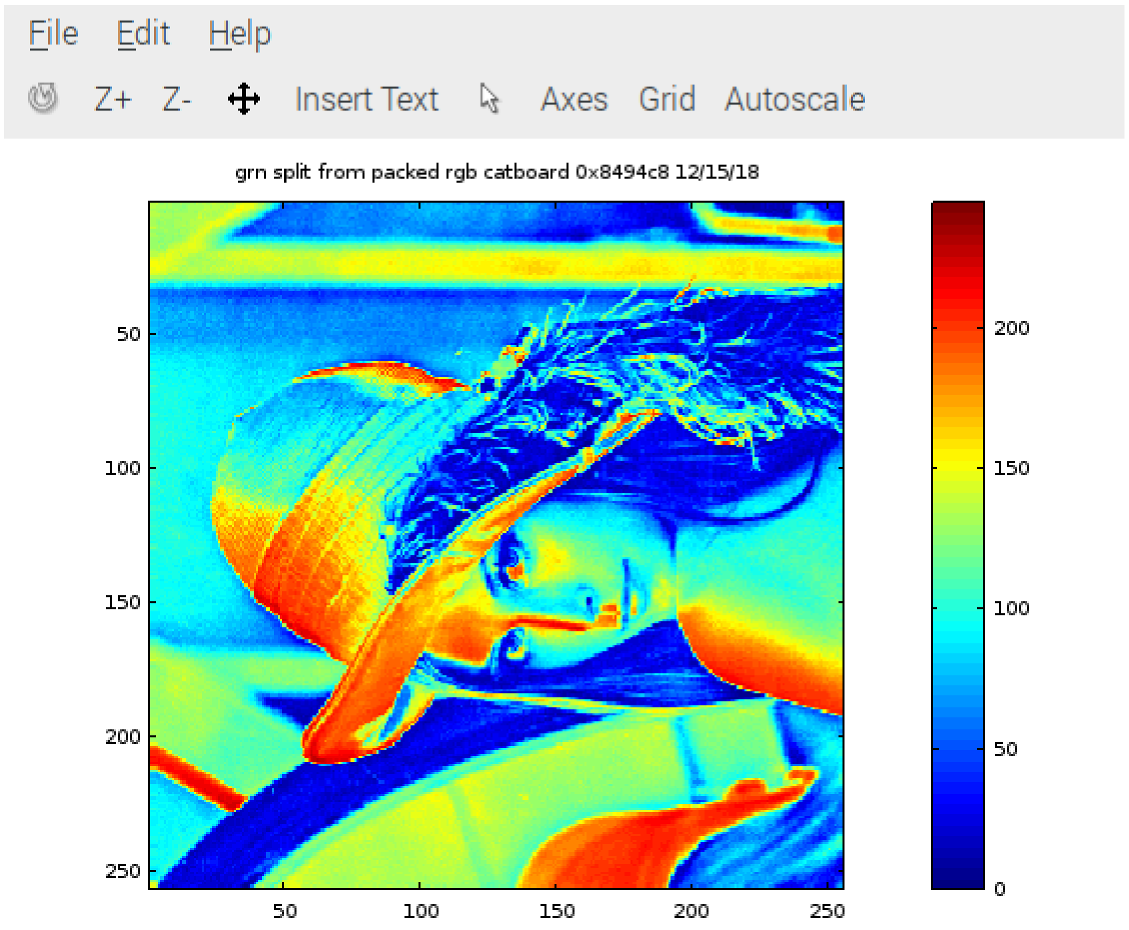
*This was fixed with the dev branch of zipcpu*

```
commit 7f02a5f8d8e1b6ba9ef701249ef996a02aa38aa4
Author: Dan Gisselquist <dgisselq@ieee.org>
Date:   Fri Sep 21 16:53:28 2018 -0400
pi@mypi3-4:~/zipcpu $ git branch -a
* dev
  master
  remotes/origin/HEAD -> origin/master
  remotes/origin/dev
  remotes/origin/master
```

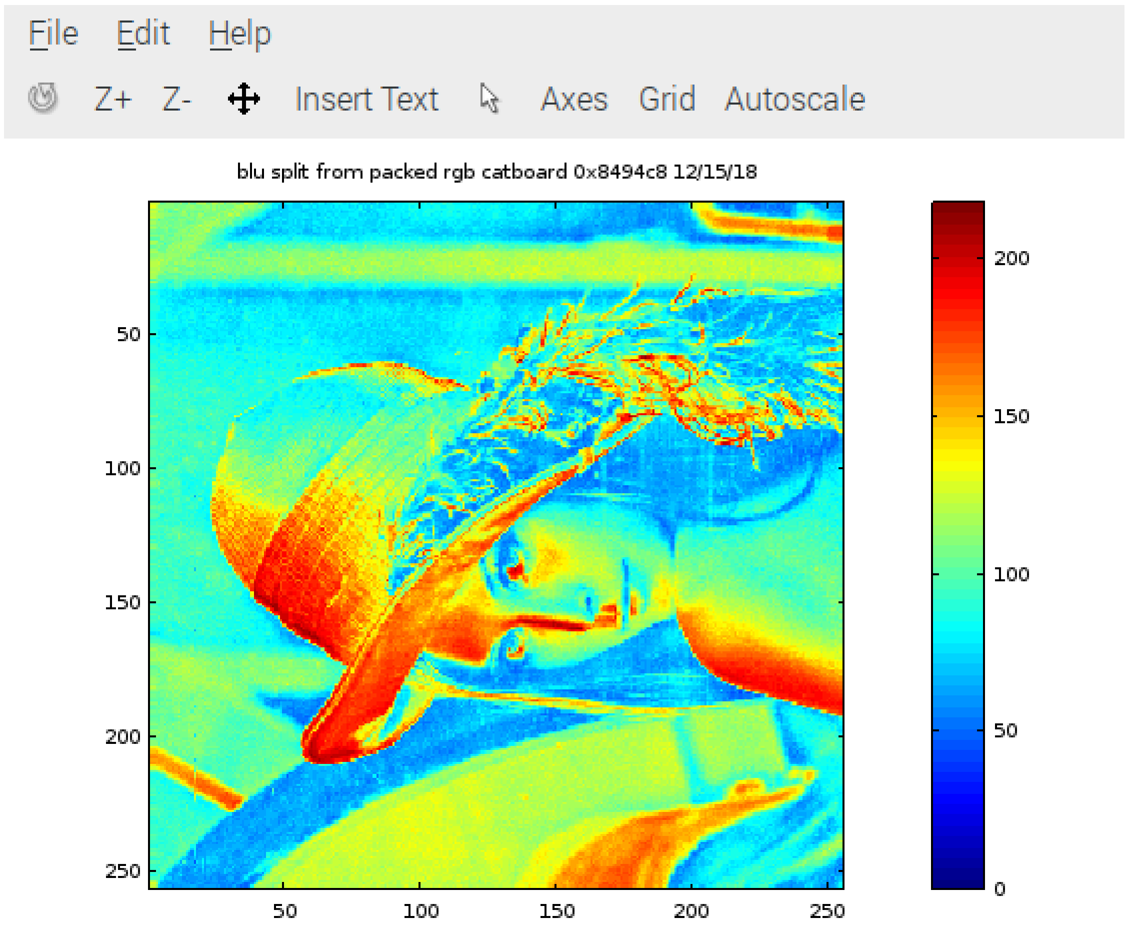
***red subband spiit using the latest jpeg.c  reads the rgb_pack,bin***



***grn subband spiit using the latest jpeg.c  reads the rgb_pack,bin***



***blu subband spiit using the latest jpeg.c  reads the rgb_pack,bin***



***red lifting step The image on left is ice40 HX8K and the image on RPi3B+***

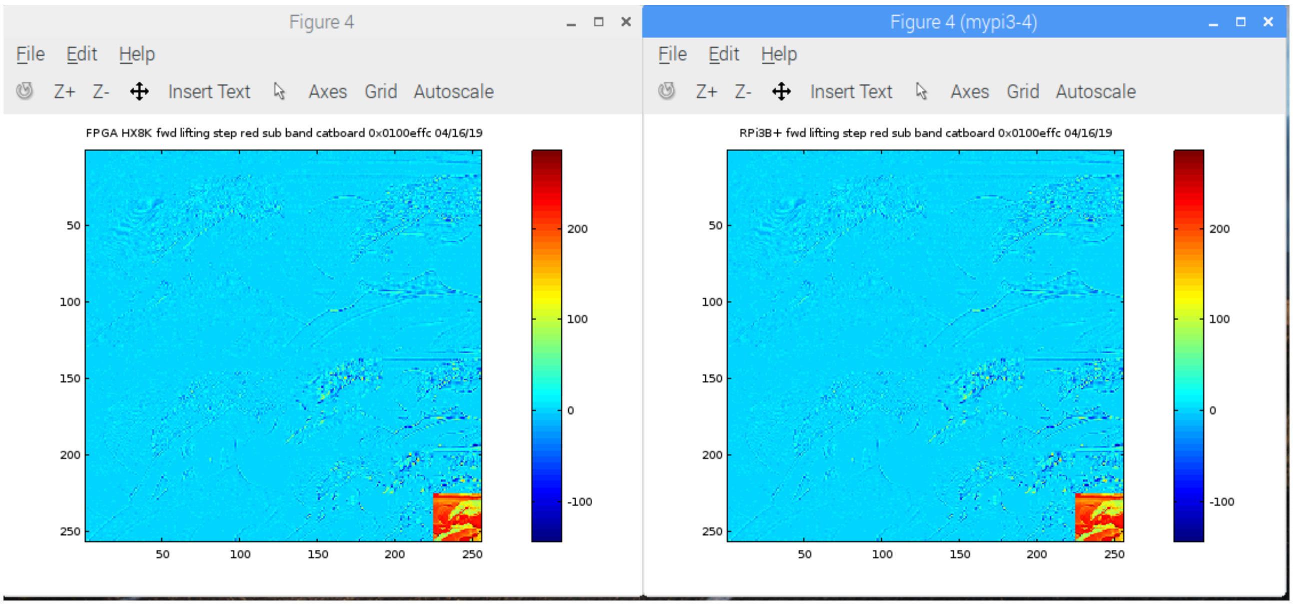
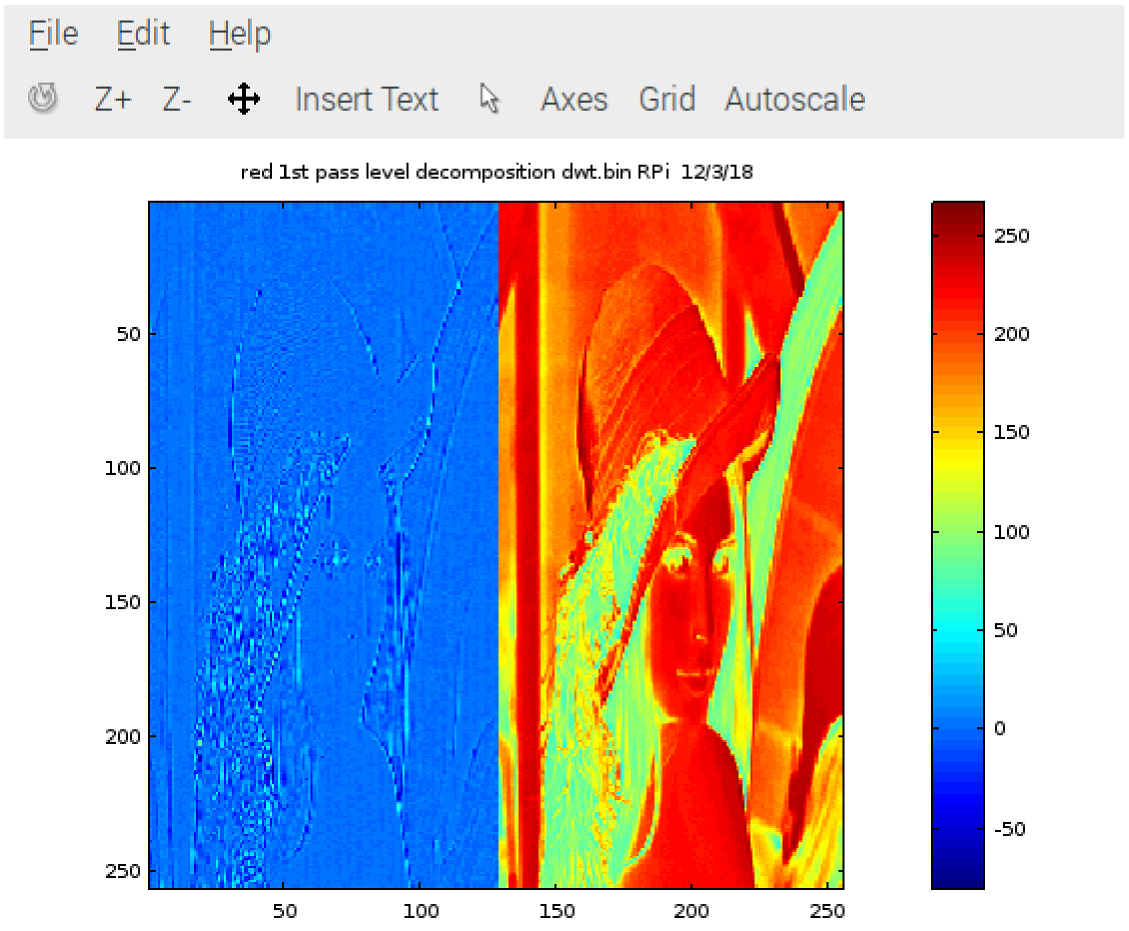
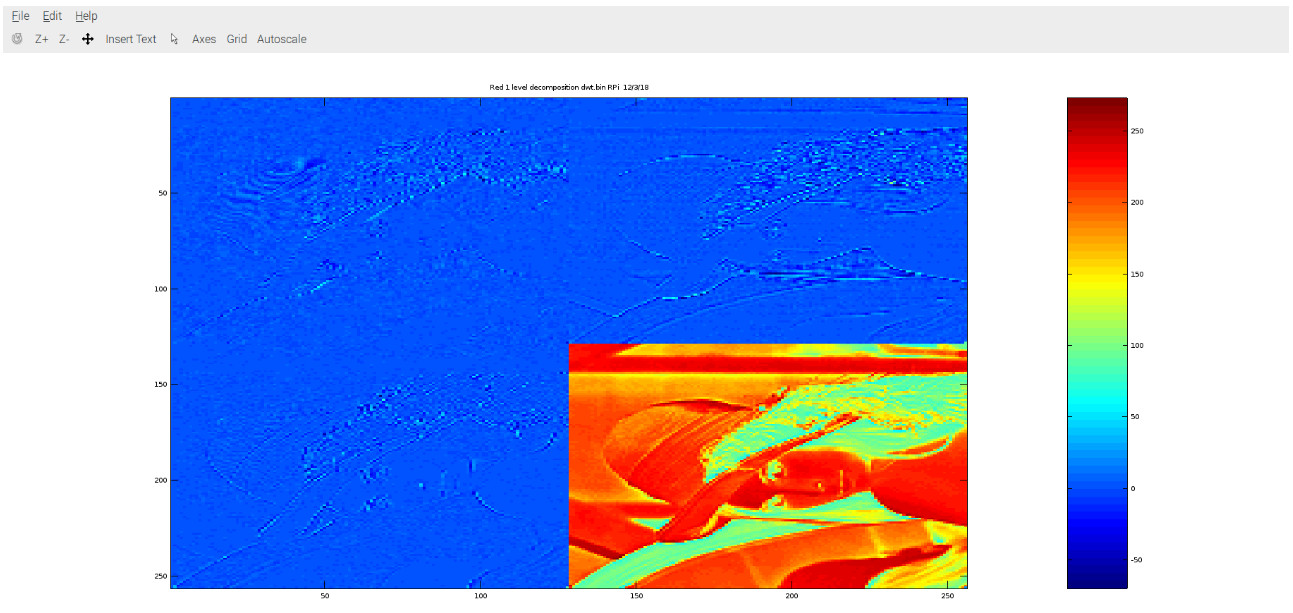


Figure 4
File Edit Help
Z+ Z- Insert Text Axes Grid Autoscale
FPGA HX8K fwd lifting step red sub band catboard 0x0100effc 04/16/19
Figure 4 (mypi3-4)
RPi3B+ fwd lifting step red sub band catboard 0x0100effc 04/16/19
50
100
150
200
250
0
-100

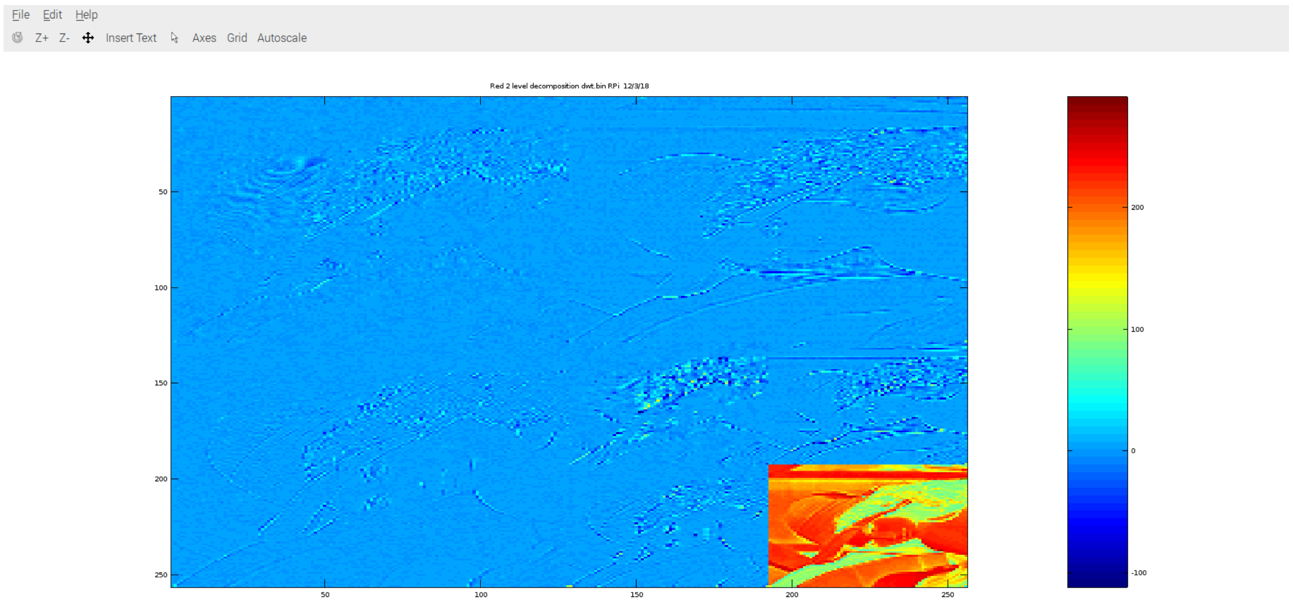
***The next 4 image were obtained on the RPi3B running the lifting step in preparation of the catboard.***
***First step of the lifting step singlelift***



***1 level of dwt image 128 X 128***


File Edit Help
Z+ Z- Insert Text Axes Grid Autoscale
Red 1 level decomposition dwt.bin RPi 12/3/18
50
100
150
200
250
50
100
150
200
250
250
200
150
100
50
0
-50

*2 levels of dwt image 64 x 64*


File Edit Help
Z+ Z- Insert Text Axes Grid Autoscale
Red 2 level decomposition dwt.bin RPi 12/3/18
50
100
150
200
250
50
100
150
200
250
200
100
0
-100

*3 levels of dwt image 32 x 32*

```
sudo config_cat hellopp.bin
. Hello, World!
sudo config_cat speechpp.bin
.  |=================================================================|
. |                                                                  |
. | Four score and seven years ago our fathers brought forth on this  |
. | continent, a new nation, conceived in Liberty, and dedicated to   |
. | the proposition that all men are created equal.                   |
. |                                                                  |
. | Now we are engaged in a great civil war, testing whether that     |
. | nation, or any nation so conceived and so dedicated, can long     |
. | endure. We are met on a great battle-field of that war. We have   |
. | come to dedicate a portion of that field, as a final resting      |
. | place for those who here gave their lives that that nation might  |
. | live. It is altogether fitting and proper that we should do this. |
. |                                                                    |
. | But, in a larger sense, we can not dedicate-we can not consecrate-|
. | we can not hallow-this ground. The brave men, living and dead,    |
. | who struggled here, have consecrated it, far above our poor power |
. | to add or detract. The world will little note, nor long remember  |
. | what we say here, but it can never forget what they did here. It  |
. | is for us the living, rather, to be dedicated here to the         |
. | unfinished work which they who fought here have thus far so nobly |
. | advanced. It is rather for us to be here dedicated to the great   |
. | task remaining before us-that from these honored dead we take     |
. | increased devotion to that cause for which they gave the last     |
. | full measure of devotion-that we here highly resolve that these   |
. | dead shall not have died in vain-that this nation, under God,     |
. | shall have a new birth of freedom-and that government of the      |
. | people, by the people, for the people, shall not perish from the  |
. | earth.                                                             |
. |                                                                    |
. |                                                                    |
. |==================================================================|
```

*Introduction*

*The goal of the ZipCPU was to be a very simple CPU. You might think of it as a poor manâϰ alternative to the OpenRISC architecture. You might also think of it as an Open Source microcontroller. For this reason, all instructions have been designed to be as simple as possible, and the base instructions are all designed to be executed in one instruction cycle per instruction, barring pipeline stalls.1 Indeed, even the bus has been simplified to a constant 32-bit width, with no option for more or less. This has resulted in the choice to drop push and pop instructions, pre-increment and post-decrement addressing modes, the integrated memory management unit (MMU), and more*

*For those who like buzz words, the ZipCPU is:*

*A 32-bit CPU: All registers are 32-bits, addresses are 32-bits, instructions are 32-bits wide, etc.*

*A RISC CPU. There is no microcode for executing instructions. All instructions are designed to be completed in one clock cycle.*

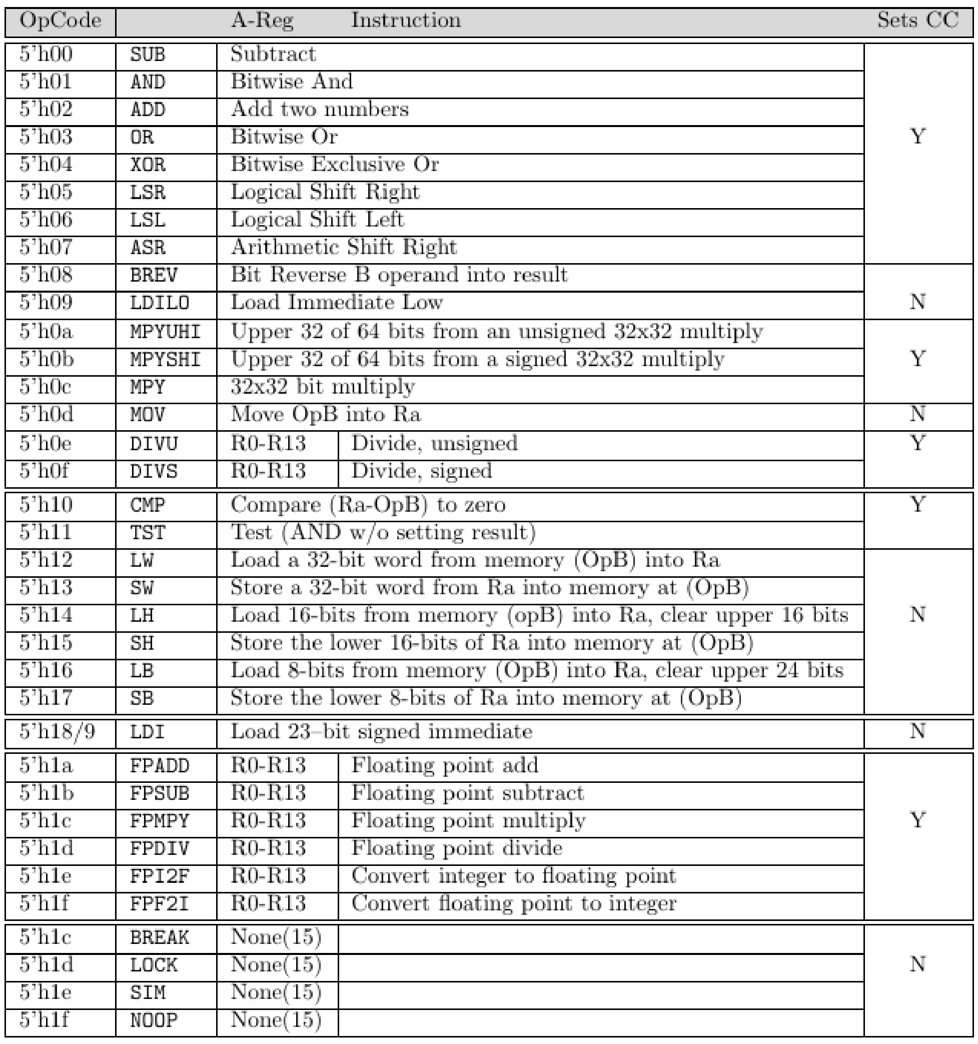
*A Load/Store architecture. (Only load and store instructions can access memory.)*

*Wishbone compliant. All peripherals are accessed just like memory across this bus.*

*A Von-Neumann architecture. The instructions and data share a common bus.*

*A pipelined architecture, having stages for Prefetch, Decode, Read-Operand, a combined stage containing the ALU, Memory, Divide, and Floating Point units, and then the final Write-back stage. See Fig. 1.1 for a diagram of this structure.*

*Completely open source, licensed under the GPL.3*



| OpCode | | A-Reg | Instruction | Sets CC |
|---|---|---|---|---|
| 5'h00 | SUB | | Subtract | Y |
| 5'h01 | AND | | Bitwise And | |
| 5'h02 | ADD | | Add two numbers | |
| 5'h03 | OR | | Bitwise Or | |
| 5'h04 | XOR | | Bitwise Exclusive Or | |
| 5'h05 | LSR | | Logical Shift Right | |
| 5'h06 | LSL | | Logical Shift Left | |
| 5'h07 | ASR | | Arithmetic Shift Right | |
| 5'h08 | BREV | | Bit Reverse B operand into result | N |
| 5'h09 | LDILO | | Load Immediate Low | |
| 5'h0a | MPYUHI | | Upper 32 of 64 bits from an unsigned 32x32 multiply | Y |
| 5'h0b | MPYSHI | | Upper 32 of 64 bits from a signed 32x32 multiply | |
| 5'h0c | MPY | | 32x32 bit multiply | |
| 5'h0d | MOV | | Move OpB into Ra | N |
| 5'h0e | DIVU | R0-R13 | Divide, unsigned | Y |
| 5'h0f | DIVS | R0-R13 | Divide, signed | |
| 5'h10 | CMP | | Compare (Ra-OpB) to zero | Y |
| 5'h11 | TST | | Test (AND w/o setting result) | |
| 5'h12 | LW | | Load a 32-bit word from memory (OpB) into Ra | N |
| 5'h13 | SW | | Store a 32-bit word from Ra into memory at (OpB) | |
| 5'h14 | LH | | Load 16-bits from memory (opB) into Ra, clear upper 16 bits | |
| 5'h15 | SH | | Store the lower 16-bits of Ra into memory at (OpB) | |
| 5'h16 | LB | | Load 8-bits from memory (OpB) into Ra, clear upper 24 bits | |
| 5'h17 | SB | | Store the lower 8-bits of Ra into memory at (OpB) | |
| 5'h18/9 | LDI | | Load 23–bit signed immediate | N |
| 5'h1a | FPADD | R0-R13 | Floating point add | Y |
| 5'h1b | FPSUB | R0-R13 | Floating point subtract | |
| 5'h1c | FPMPY | R0-R13 | Floating point multiply | |
| 5'h1d | FPDIV | R0-R13 | Floating point divide | |
| 5'h1e | FPI2F | R0-R13 | Convert integer to floating point | |
| 5'h1f | FPF2I | R0-R13 | Convert floating point to integer | |
| 5'h1c | BREAK | None(15) | | N |
| 5'h1d | LOCK | None(15) | | |
| 5'h1e | SIM | None(15) | | |
| 5'h1f | NOOP | None(15) | | |

Table 2.2: ZipCPU OpCodes

***"export PATH=/home/pi/zipcpu/sw/install/cross-tools/bin:/home/pi/autofpga/sw/:$PATH"***

***"cd catzip"***

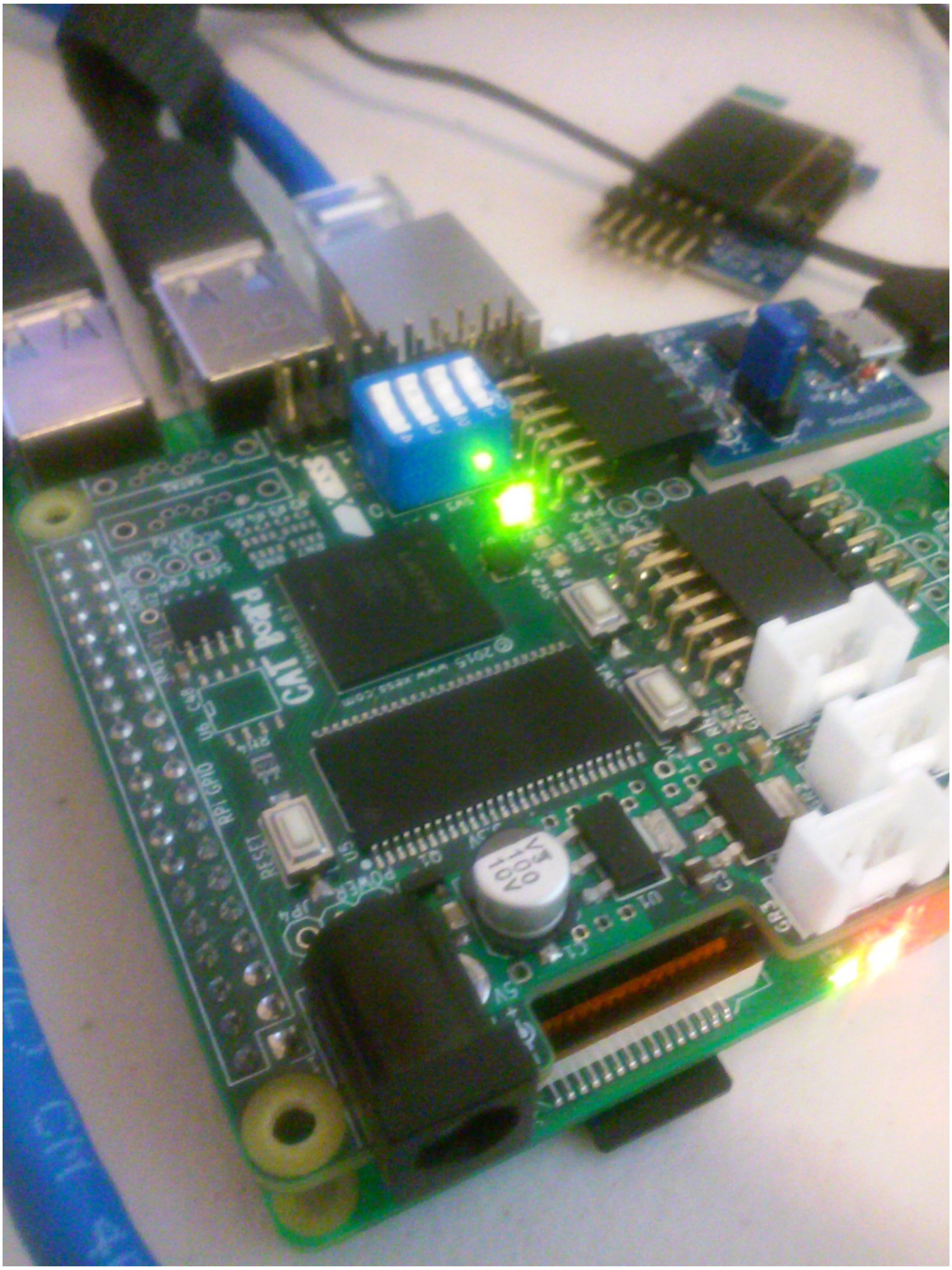
***"make clean"***

***"make"  This creates the 2 executeables arm-netpport & arm-wbregs used to communicate with the FPGA.in files for download to FPGA***
***In additon compiles the***
***The rtl has several folders basic, uart,catzip, leddigits, pptest, switch_leds, and sdram where *.bin files are created.***
***The sw/host has several executeables used to communicate with the FPGA.***

 arm-netpport arm-rdsdram arm-wbregs arm-wrsdram arm-zipdbg arm-zipload arm-zipstate arm-sdramscope

```
ild.pcf      gpio.txt       pwmaudio.txt        toplevel.v
buserr.txt     hbconsole.txt   pwmfifo.txt         version.txt
clock50.txt     legalgen.txt    pwrcount.txt        wbuart.txt
clockpll40.txt  main_tb.cpp     regdefs.cpp         wbuconsole.txt
clockpll45.txt  main.v         regdefs.h           zipbones.txt
pi@mypi3-1:~/testbuilds/catzip/sw/host $ ls
arch.sh          exe_wbregs.c     pi_jpeg_2_1.png       sdramscope.cpp
arm-netpport      fpga_C.png       port.h                sim_hw_test.sh
arm-rdsdram       fwddwt.png       pptest.cpp            sim_jpeg_0_1.png
arm-sdramscope    fwdtmp.png       rd_bytes.c            testscope.sh
arm-wbregs        grn.bin          rd_pattern.sh         test_sim102218.sh
arm-wrsdram       grn-out.8        rdsdram.cpp           ttybus.cpp
arm-zipdbg        hexbus.cpp       red.bin               ttybus.h
arm-zipload       hexbus.h         red-out.8             twoc.cpp
arm-zipstate      jpeg_0_1.png     regdefs.cpp           twoc.h
blu.bin           jpeg_1_1.png     regdefs.h             wbregs.cpp
blu-out.8         jpeg_2_1.png     rem_sim_hw_test.sh    wr_pattern.sh
byteswap.cpp      jpeg.sh          rem_test_sim102218.sh  wrsdram.cpp
byteswap.h        llcomms.cpp      resetbus.cpp          wr_test.sh
convertrgb.m      llcomms.h        rgb.m                 zipdbg.cpp
cputest_hello.sh  Makefile         rgb_pack.bin          zipelf.cpp
devbus.h          netpport.cpp     runs1.txt             zipelf.h
disp_rgb.m        obj-arm          runs2.txt             zipload.cpp
dwt.bin           pi_jpeg          runs.txt              zipstate.cpp
dwtjpeg21.bin     pi_jpeg_0_1.png  scopecls.cpp          zopcodes.cpp
dwt_rpi.bin       pi_jpeg_1_1.png  scopecls.h            zopcodes.h
pi@mypi3-1:~/testbuilds/catzip/sw/host $ ^C
pi@mypi3-1:~/testbuilds/catzip/sw/host $ ./sim_hw_test.sh
The date built
00800010 ( VERSION) : [....] 20190415
00a00000 (     RAM)-> 10000001
00a00004 (        )-> 10000002
00a00008 (        )-> 10000003
00a0000c (        )-> 10000004
00a00000 (     RAM)-> 10000001
00a00004 (        )-> 10000002
00a00008 (        )-> 10000003
00a0000c (        )-> 10000004
00a00000 (     RAM) : [....] 10000001
00a00004 (        ) : [....] 10000002
00a00008 (        ) : [....] 10000003
00a0000c (        ) : [....] 10000004
```

```
00a00000 (     RAM) : [....] 10000001
00a00004 (        ) : [....] 10000002
00a00008 (        ) : [....] 10000003
00a0000c (        ) : [....] 10000004
01000000 (   SDRAM)-> 10000001
01000004 (        )-> 10000002
01000008 (        )-> 10000003
0100000c (        )-> 10000004
01000000 (   SDRAM)-> 10000001
01000004 (        )-> 10000002
01000008 (        )-> 10000003
0100000c (        )-> 10000004
01fffffc (        )-> 1ffffffc
01000000 (   SDRAM) : [....] 10000001
01000004 (        ) : [....] 10000002
01000008 (        ) : [....] 10000003
0100000c (        ) : [....] 10000004
01000000 (   SDRAM) : [....] 10000001
01000004 (        ) : [....] 10000002
01000008 (        ) : [....] 10000003
0100000c (        ) : [....] 10000004
01fffffc (        ) : [....] 1ffffffc
Turning on the 4th led
00800008 (    GPIO)-> 00010001
Turning on the 1st led
00800008 (    GPIO)-> 00020002
Turning on the 2nd led
00800008 (    GPIO)-> 00040004
Turning off the leds
00800008 (    GPIO)-> 00070000
```

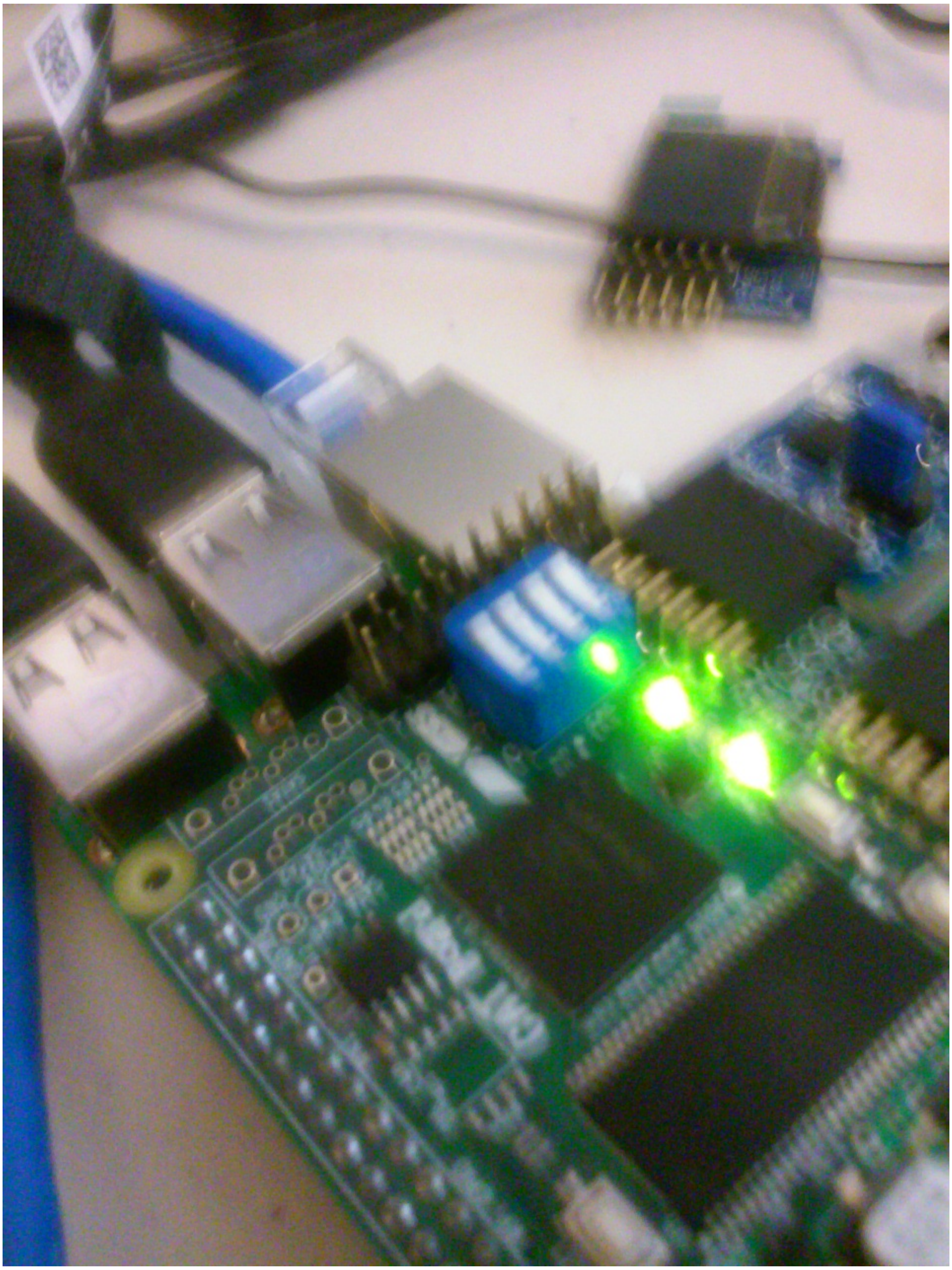

CAT Board
2015 www.xess.com
RESET
RPI GPIO
100
10V

1st Led off

```
./arm-wbregs gpio 0x00010000
00001008 (    GPIO)-> 00010000
```

2n led on

```
./arm-wbregs gpio 0x00020002
00001008 (    GPIO)-> 00020002
```

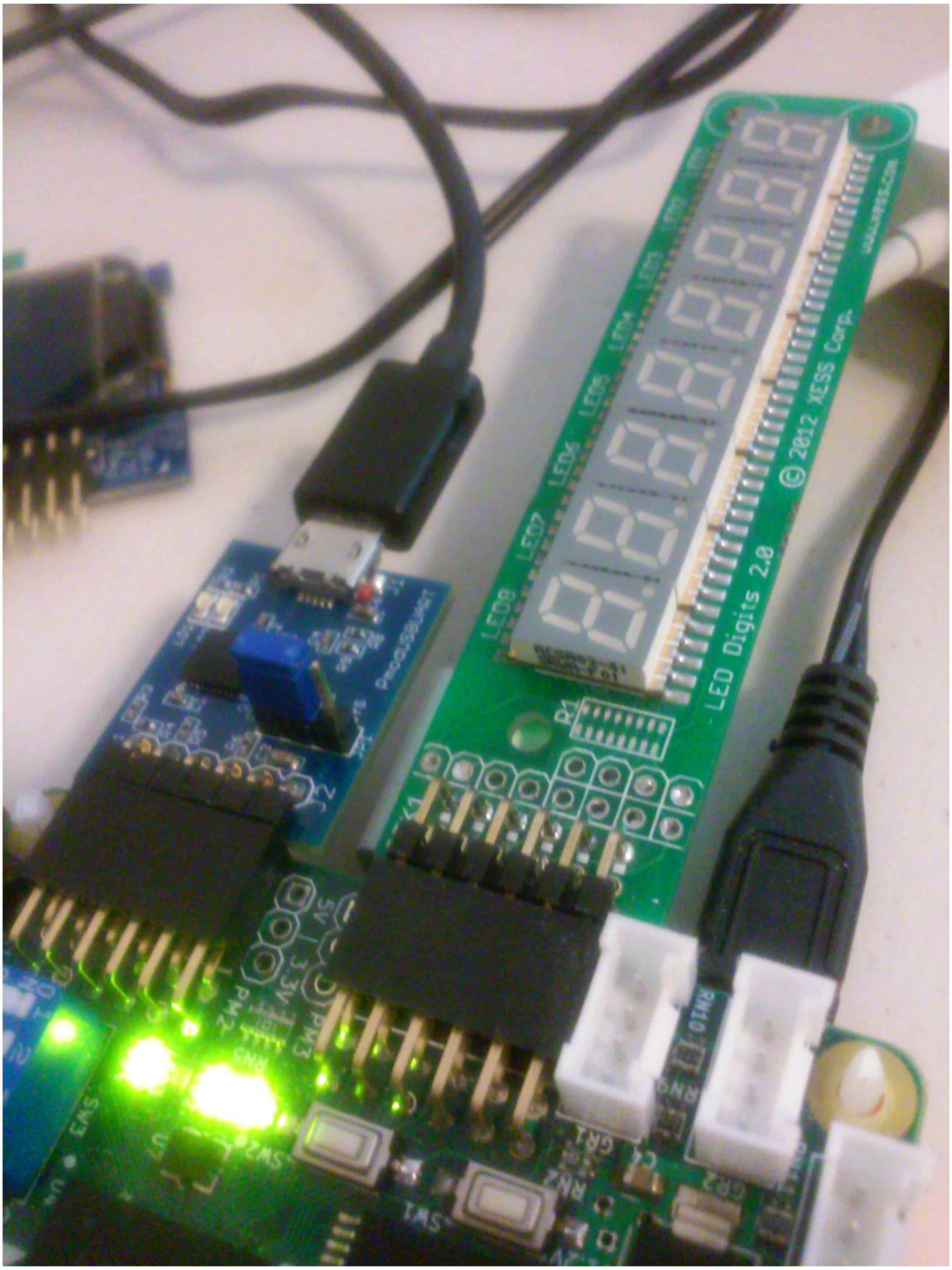


2nd led off

```
./arm-wbregs gpio 0x00020000
00001008 (    GPIO)-> 00020000
```

3rd led on

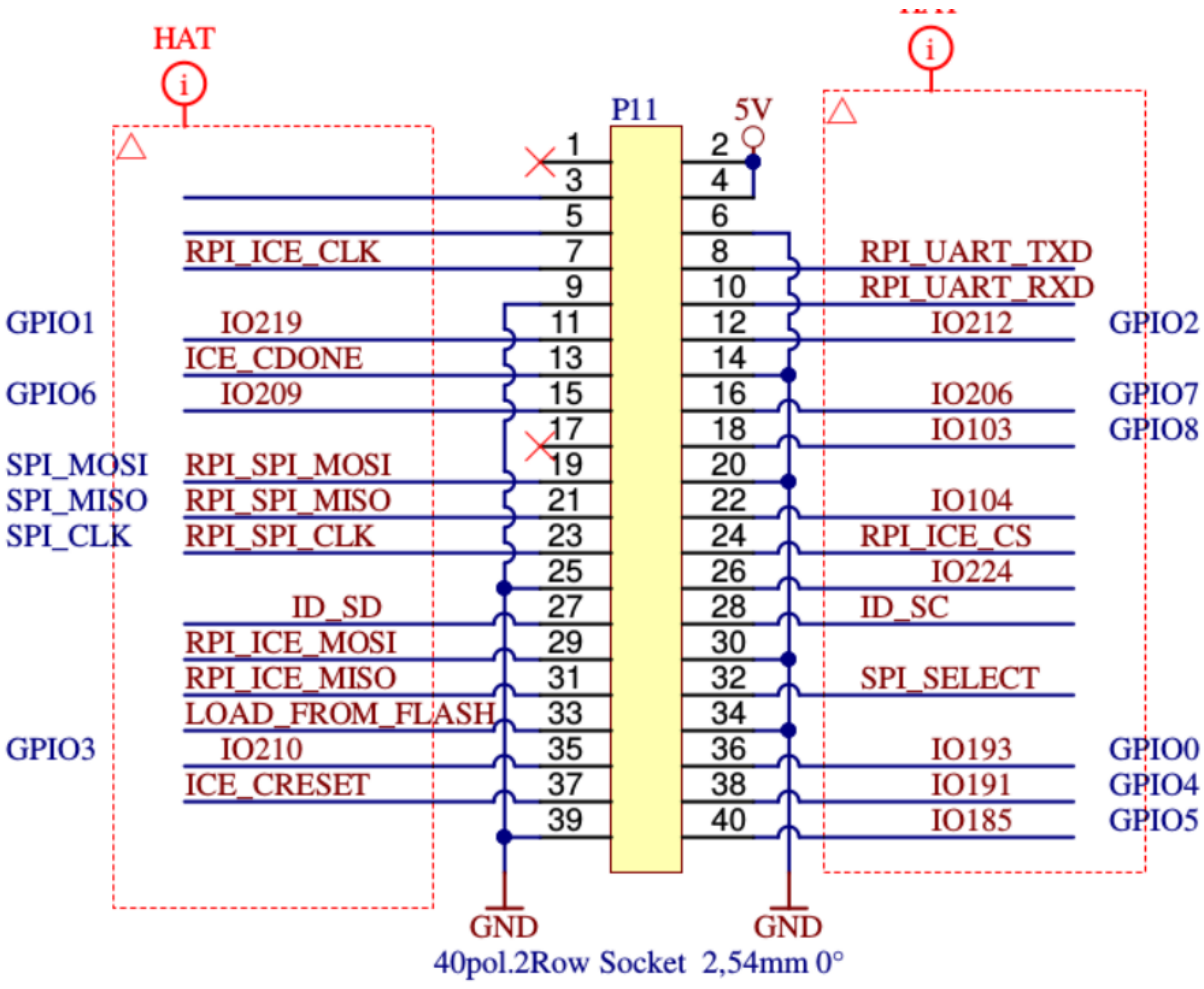
```
./arm-wbregs gpio 0x00040004
00001008 (    GPIO)-> 00040004
```


LED Digits 2.0
© 2012 XESS Corp.
www.xess.com
PmodUSBUART
PM2
PM3
3.3V
5V
SW1
SW2
SW3

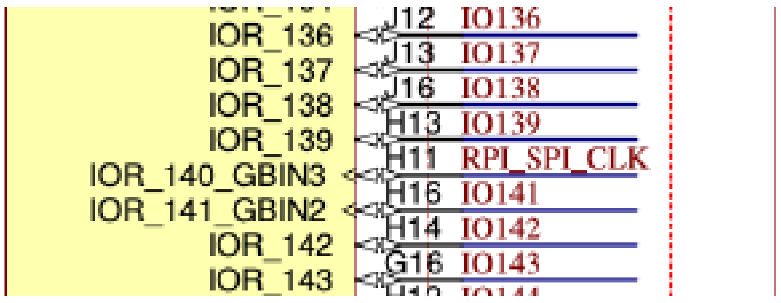
3rd led off

```
./arm-wbregs gpio 0x00040000
00001008 (    GPIO)-> 00040000

./arm-wbregs 0x2000 0x01
00002000 (     RAM)-> 00000001
pi@pi3-5:~/catzip/sw/host $ ./arm-wbregs ram
00002000 (     RAM) : [....] 00000001
pi@pi3-5:~/catzip/sw/host $ ./arm-wbregs 0x2000 0x02
00002000 (     RAM)-> 00000002
pi@pi3-5:~/catzip/sw/host $ ./arm-wbregs ram
00002000 (     RAM) : [....] 00000002

./arm-wbregs pic
00001004 (     PIC) : [....] 00000003

./arm-wbregs ufifo
00000804 (   UFIFO) : [@?@.] 403f4000
```

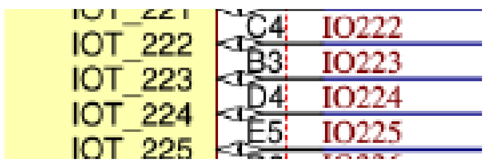
ICOBOARD RPi



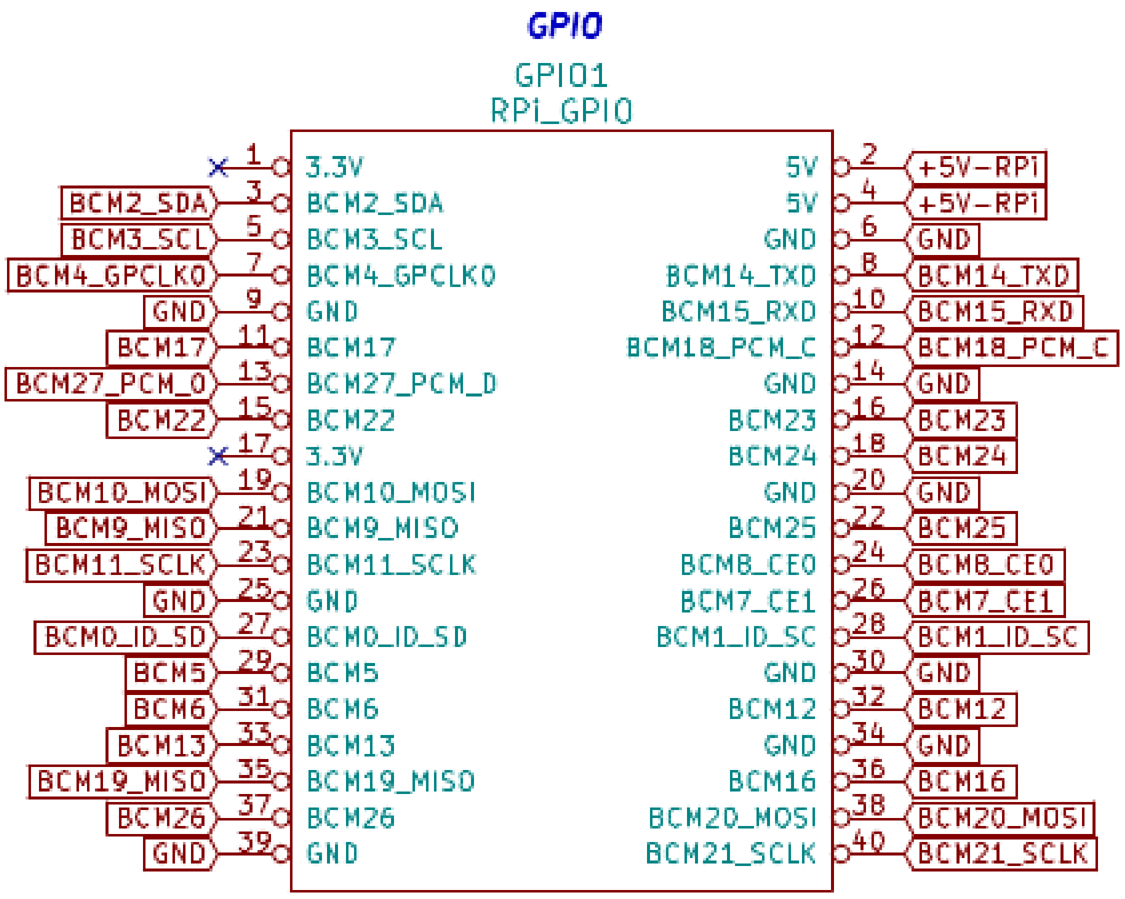
RPI_SPI_CLK H11 Pin 23 Pi icoboard



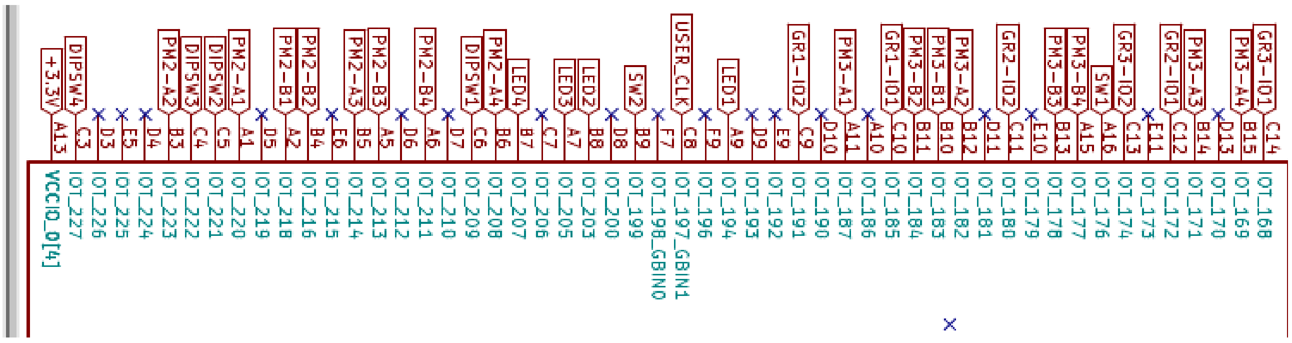
rpi_cs  D4  IOT_224 Pin 26 Pi icoboard



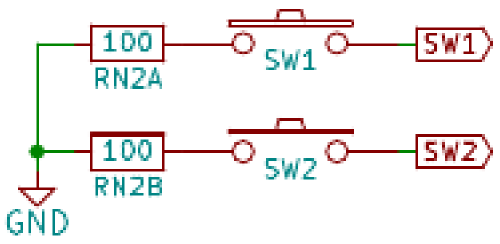
CATBOARD RPi



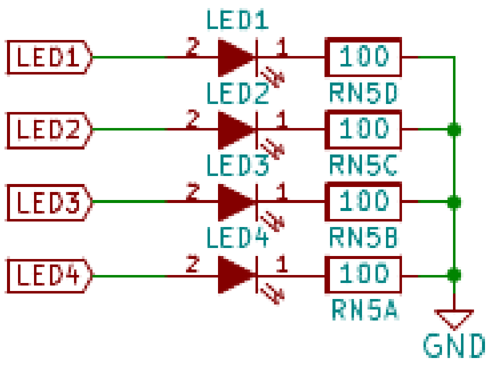
CATBOARD connection to FPGA pins PMOD 2 & PMOD 3 push button switches, dip switch, and leds.


+3.3V
DIPSW4
PM2-A2
DIPSW3
DIPSW2
PM2-A1
PM2-B1
PM2-B2
PM2-A3
PM2-B3
PM2-B4
DIPSW1
PM2-A4
LED4
LED3
LED2
SW2
USER_CLK
LED1
GR1-IO2
PM3-A1
GR1-IO1
PM3-B2
PM3-B1
PM3-A2
GR2-IO2
PM3-B3
PM3-B4
SW1
GR3-IO2
GR2-IO1
PM3-A3
PM3-A4
GR3-IO1
A13 C3 D3 E5 D4 B3 C4 C5 A1 D5 A2 B4 E6 B5 A5 D6 A6 D7 C6 B6 B7 C7 A7 B8 D8 B9 F7 C8 F9 A9 D9 E9 C9 D10 A11 A10 C10 B11 B10 B12 D11 C11 E10 B13 A15 A16 C13 E11 C12 B14 D13 B15 C14
VCCIO_0[4]
IOT_227 IOT_226 IOT_225 IOT_224 IOT_223 IOT_222 IOT_221 IOT_220 IOT_219 IOT_218 IOT_216 IOT_215 IOT_214 IOT_213 IOT_212 IOT_211 IOT_210 IOT_209 IOT_208 IOT_207 IOT_206 IOT_205 IOT_203 IOT_200 IOT_199 IOT_198_GBIN0 IOT_197_GBIN1 IOT_196 IOT_194 IOT_193 IOT_192 IOT_191 IOT_190 IOT_187 IOT_186 IOT_185 IOT_184 IOT_183 IOT_182 IOT_181 IOT_180 IOT_179 IOT_178 IOT_177 IOT_176 IOT_174 IOT_173 IOT_172 IOT_171 IOT_170 IOT_169 IOT_168

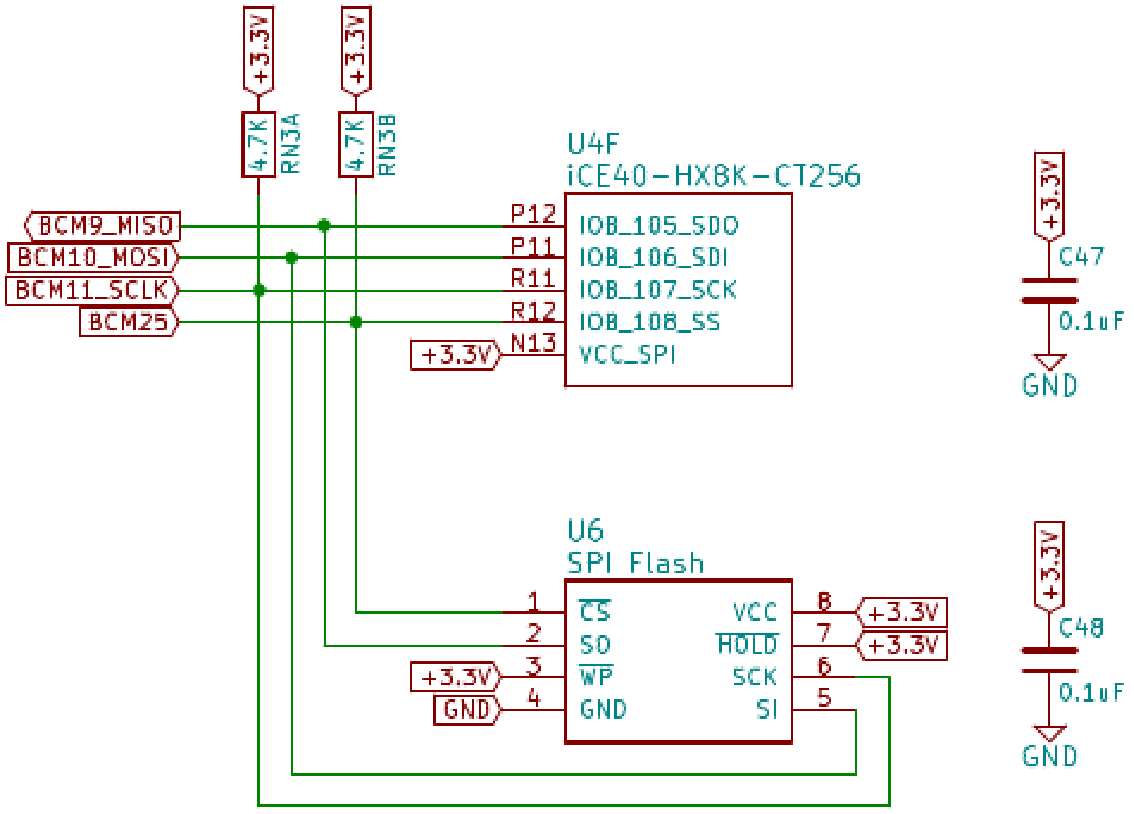
CATBOARD sw1 & sw2



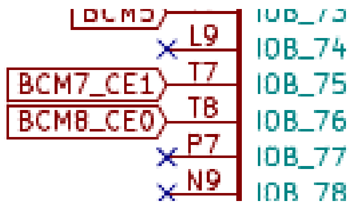
CATBOARD leds



BCM11_SCLK Pin 23 CATBOARD

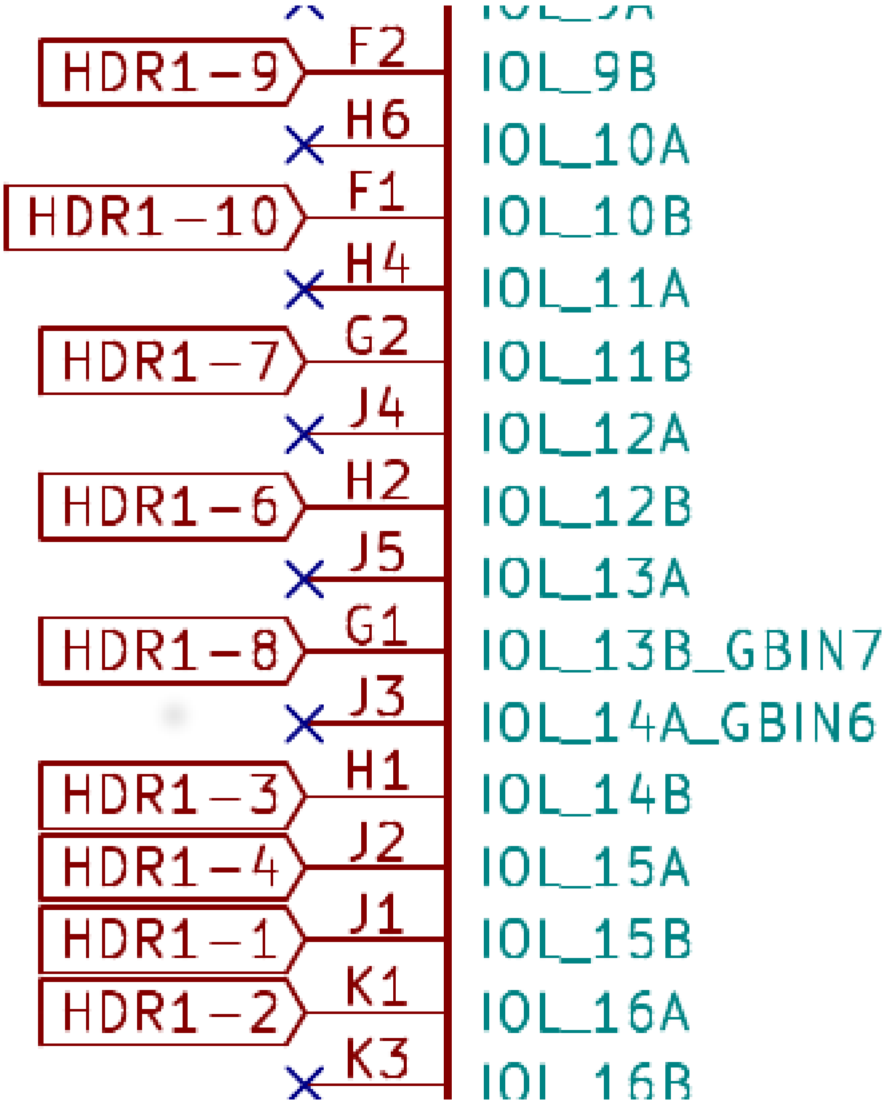


BCM7_CE1 Pin 26 CATBOARD



BCM5
L9
IOB_73
IOB_74
BCM7_CE1
T7
IOB_75
BCM8_CE0
T8
IOB_76
P7
IOB_77
N9
IOB_78

CATBOARD



HDR1-9
F2
IOL_9B
H6
IOL_10A
HDR1-10
F1
IOL_10B
H4
IOL_11A
HDR1-7
G2
IOL_11B
J4
IOL_12A
HDR1-6
H2
IOL_12B
J5
IOL_13A
HDR1-8
G1
IOL_13B_GBIN7
J3
IOL_14A_GBIN6
HDR1-3
H1
IOL_14B
HDR1-4
J2
IOL_15A
HDR1-1
J1
IOL_15B
HDR1-2
K1
IOL_16A
K3
IOL_16B

2.) The 2nd issue is the PMOD connections to FPGA are different.
3.) Third, I do not have a Diglient PMOD 4 push button switch module.
4.) The 4th issue is the PHASE LOCK LOOP difference.

Post on #yosys

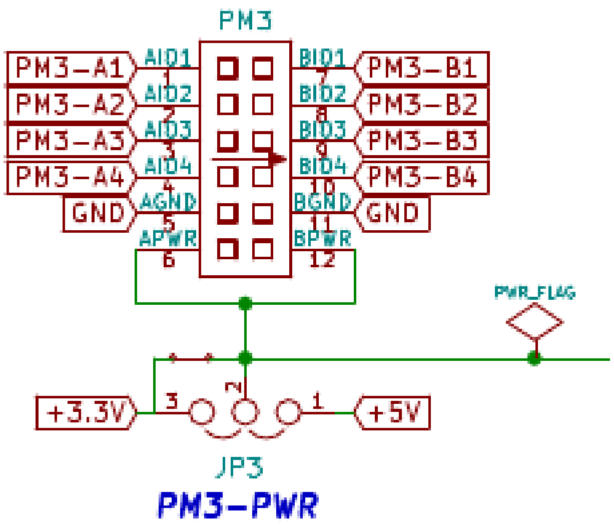
*Pin C8 is my USER_CLK comes from a 100MHz osc. It is connected to IOT_197_GBIN1 on HX8K. When I try using it for as an input to PLL I get the fatal error: bad constraint on `i_clk': no PLL at pin C8.*

*Can only certain pins be used as inputs to PLL?*

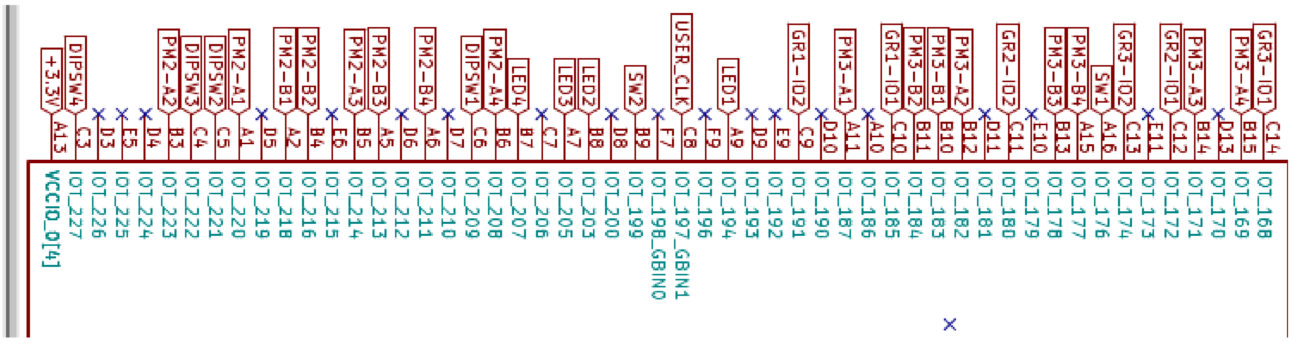
*daveshah*

*develonepi3: use the SB_PLL40_CORE instead of SB_PLL40_PAD variant (and REFERENCECLK in instead of PACKAGEPIN)*

```
set_io clk_100mhz C8    #R9

set_io pmod1_1 A11      #D8
set_io pmod1_2 B12      #B9
set_io pmod1_3 B14      #B10
set_io pmod1_4 B15      #B11
#   654321      catboard    #   654321  icoboard
#       xxxxxx  PMOD3 A         #       xxxxxx  PMOD1 A
#       xxxxxx  PMOD3 B         #       xxxxxx  PMOD1 B
#   654321                      #   654321
#
set_io pmod1_7 B10      #B8
set_io pmod1_8 B11      #A9
set_io pmod1_9 B13      #A10
set_io pmod1_10 A15     #A11
```



CATBOARD connection to FPGA pins PMOD 2 & PMOD 3 push button switches, dip switch, and leds.



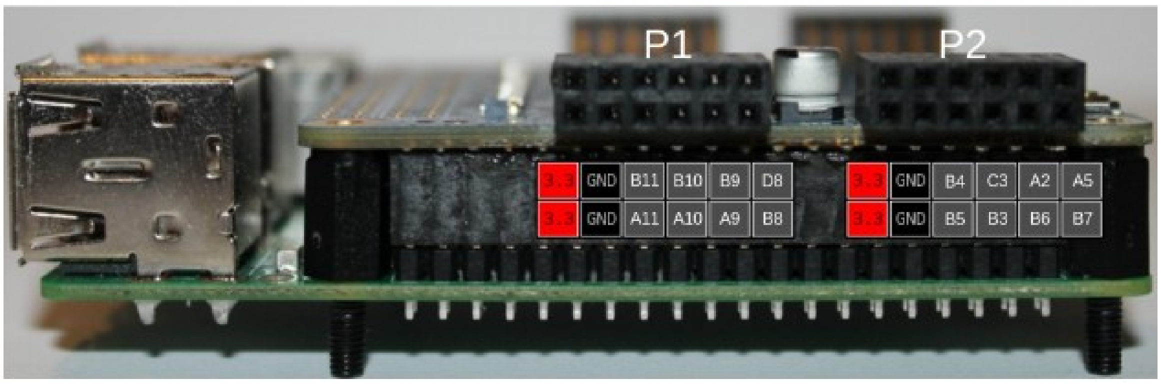
In top.v

module top(clk_100mhz, pmod1_1, pmod1_2, pmod1_3, pmod1_4, pmod1_7, pmod1_8,
  pmod1_9, pmod1_10, pmod2_7, pmod2_8, pmod2_9, pmod2_10, rpi_sck, rpi_cs,
  rpi_mosi);

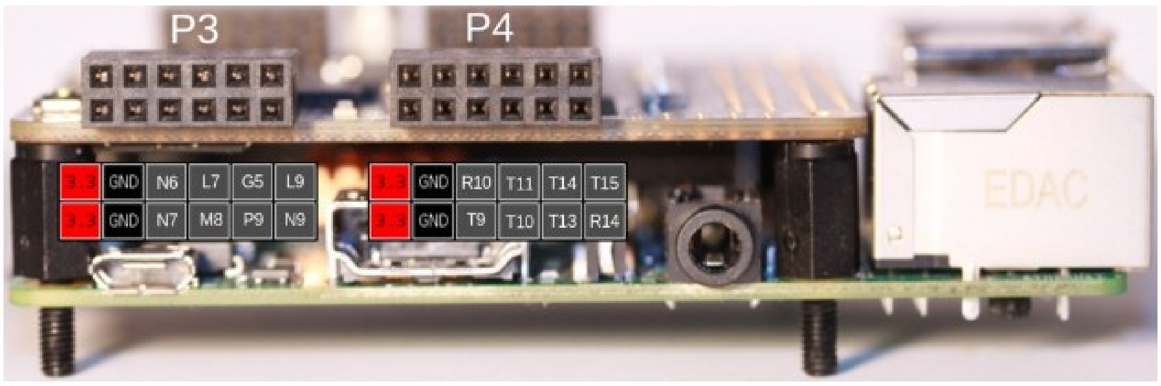
input rpi_sck, rpi_cs, rpi_mosi;
rpi_sck
rpi_cs
rpi_mosi

spi_ram_slave spi_ram_slave(clk, rpi_sck, rpi_cs, rpi_mosi,
  ram_addr, ram_data, ram_wr);
module spi_ram_slave(clk, sck, cs, mosi, ram_addr, ram_data, ram_wr);
PMOD pin out on  icoboard

Pinout Pmod P1 and P2



Pinout PMOD P3 and P4



```
“lrwxrwxrwx 1 root staff 34 May 18 20:10 /usr/local/bin/config_cat ->
/home/pi/catboard_yosys/config_cat”

#!/bin/bash

#    A script to configure Lattice iCE40 FPGA by SPI from Raspberry Pi
#
#    Copyright (C) 2015 Jan Marjanovic <jan@marjanovic.pro>
#
#    This program is free software: you can redistribute it and/or modify
#    it under the terms of the GNU General Public License as published by
#    the Free Software Foundation, either version 3 of the License, or
#    (at your option) any later version.
#
#    This program is distributed in the hope that it will be useful,
#    but WITHOUT ANY WARRANTY; without even the implied warranty of
#    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
#    GNU General Public License for more details.
#
#    You should have received a copy of the GNU General Public License
#    along with this program.  If not, see <http://www.gnu.org/licenses/>.


echo ""
if [ $# -ne 1 ]; then
   echo "Usage: $0 FPGA-bin-file "
   exit 1
fi

if [ $EUID -ne 0 ]; then
   echo "This script must be run as root" 1>&2
   exit 1
fi


if [ ! -d /sys/class/gpio/gpio25 ]; then
   echo "GPIO 25 not exported, trying to export..."
   echo 25 > /sys/class/gpio/export
   if [ ! -d /sys/class/gpio/gpio25 ]; then
         echo "ERROR: directory /sys/class/gpio/gpio25 does not exist"
         exit 1
   fi
else
   echo "OK: GPIO 25 exported"
fi

if [ ! -d /sys/class/gpio/gpio17 ]; then
   echo "GPIO 17 not exported, trying to export..."
   echo 17 > /sys/class/gpio/export
   if [ ! -d /sys/class/gpio/gpio17 ]; then
         echo "ERROR: directory /sys/class/gpio/gpio17 does not exist"
         exit 1
   fi
else
   echo "OK: GPIO 17 exported"
fi

if [ ! -d /sys/class/gpio/gpio22 ]; then
   echo "GPIO 22 not exported, trying to export..."
   echo 22 > /sys/class/gpio/export
   if [ ! -d /sys/class/gpio/gpio22 ]; then
         echo "ERROR: directory /sys/class/gpio/gpio22 does not exist"
         exit 1
   fi
else
   echo "OK: GPIO 22 exported"
fi
```

```
echo ""
if [ -e /dev/spidev0.0 ]; then
   echo "OK: SPI driver loaded"
else
   echo "spidev does not exist"

   lsmod | grep spi_bcm2708 >& /dev/null

   if [ $? -ne 0 ]; then
         echo "SPI driver not loaded, try to load it..."
         modprobe spi_bcm2708

         if [ $? -eq 0 ]; then
            echo "OK: SPI driver loaded"
         else
            echo "Could not load SPI driver"
            exit 1
         fi
   fi
fi

echo ""
echo "Setting GPIO directions"
echo out > /sys/class/gpio/gpio25/direction
cat /sys/class/gpio/gpio25/direction
echo out > /sys/class/gpio/gpio22/direction
cat /sys/class/gpio/gpio22/direction
echo in > /sys/class/gpio/gpio17/direction
cat /sys/class/gpio/gpio17/direction

echo "Setting output to low"
echo 0 > /sys/class/gpio/gpio25/value
cat /sys/class/gpio/gpio25/value

#echo ""
#echo "Please reset the iCE40 FPGA board"
#echo "Press any key..."
#read

echo "Reseting FPGA"
echo 0 > /sys/class/gpio/gpio22/value
cat /sys/class/gpio/gpio22/value
echo 1 > /sys/class/gpio/gpio22/value
cat /sys/class/gpio/gpio22/value

echo "Checking DONE pin"
cat /sys/class/gpio/gpio17/value

echo "Continuing with configuration procedure"
dd if=$1 of=/dev/spidev0.0
```

```
echo -e "\x0\x0\x0\x0\x0\x0\x0" > /dev/spidev0.0

echo "Setting output to high"
echo 1 > /sys/class/gpio/gpio25/value
cat /sys/class/gpio/gpio25/value

echo "Checking DONE pin"
cat /sys/class/gpio/gpio17/value


“cd otl-icoboard-pmodoledrgb-demo/stream-tool/”

“ffmpeg -f v4l2 -i /dev/video0 -s 96x64 -f rawvideo -pix_fmt rgb565 - | ./stream-tool”
```