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

Adapting the ICOBOARD icozip to the CATBOARD catzip

12/14/18

*********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.

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

Catboard & Icoboard

ICOBOARD vs CATBOARD

- Remote Access to FPGA
- Modified Wishbone
- Autofpga 2f4435 08/05/18 project provides
  - toplevel.v & main.v
  - cpudef.h & design.h
  - board.ld
- ZipCPU ab82a88 04/20/18
  - GCC Compiler
  - binutils
- 128K SRAM
- SB_IO toplevel interface to SRAM
- 4 PMOD connector, flat ribbon connectors, leds
- 50 Mhz no PLL
- SPIXEXPRESS flash
- IceHX8K FPGA
- Interface to RPI 8 bit parallel
- Yosys Tool Chain

- Remote Access to FPGA
- Modified Wishbone
- Autofpga 2f4435 08/05/18 projgect provides
  - toplevel.v & main.v
  - cpudef.h & design.h
  - board.ld
- ZipCPU ab82a88 04/20/18
  - GCC Compiler
  - binutils
- 8M SDRAM
- Tristate 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
- iceHX8K FPGA
- Interface to RPI 8 bit parallel
- Yosys Tool Chain

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
00400010 (VERSION) : [....] 20181209

Reads Sdram data at address 0x8098a8

./arm-wbregs 0x8098a8
008098a8 ( ) : [."$\|] 0e22247c

arm-zipload

./arm-zipload -v ../board/jpeg
./arm-wbregs cpu 0x0f

Reads data from Sdram writing to a file

arm-rdsdram sdram..

Writes data from a file to Sdram

arm-wrsdram rgb_pack.bin

rgb.bin was used to create rgb_pack.bin
const int bb = 0x1ff; lsb 9 bits of 32 bit word
const int gg = 0x7fc00; bits 10-19 of 32 bit word
const int rr = 0x1ff00000; msb 9 bits 32 bit word
r g b
0 e2 89 7c e22247c
1 de 88 83 de22083
2 e2 85 75 e221475
3 e3 88 7b e32207b
4 e1 8a 7a e12287a
5 e2 86 7c e22187c
6 e0 86 70 e021870
7 dd 83 75 dd20c75
8 dc 84 6d dc2106d
9 de 8b 76 de22c76
10 df 80 6a df2006a
11 de 83 69 de20c69
12 db 84 6f db2106f
13 e0 7f 65 e01fc65
14 dc 81 74 dc20474
15 dd 81 74 dd20474
16 df 82 69 df20869
17 e0 83 72 e020c72
18 e2 8a 76 e222876
19 e6 89 70 e622470

.
.
.

48003 d9 66 6a d91986a
48004 db 65 6a db1946a
48005 da 69 6e da1a46e
48006 dc 6f 72 dc1bc72
48007 d8 6c 75 d81b075
48008 dc 6d 6e dc1b46e
48009 de 73 78 de1cc78
48010 e0 77 78 e01dc78
48011 e3 7e 7e e31f87e
48012 e3 85 84 e321484
48013 e0 80 82 e020082
48014 e2 7f 83 e21fc83
48015 e6 82 84 e620884
48016 e4 8f 89 e423c89
48017 e7 95 8c e72548c
48018 e8 91 89 e824489
48019 e6 8d 84 e623484
48020 e7 90 8d e72408d
48021 e5 85 84 e521484
48022 e3 8b 88 e322c88
48023 de 88 86 de22086
48024 de 7d 7d de1f47d
48025 dc 75 7d dc1d47d

48026 da 7c 7d da1f07d

.

.

.

65514 a5 50 5f a51405f
65515 ac 63 6e ac18c6e
65516 b2 64 6f b21906f
65517 ae 6d 75 ae1b475
65518 a9 69 78 a91a478
65519 a5 64 71 a519071
65520 96 55 66 9615466
65521 85 43 5b 8510c5b
65522 75 29 44 750a444
65523 67 22 3f 670883f
65524 66 21 45 6608445
65525 5e 1a 3c 5e0683c
65526 5a f 30 5a03c30
65527 5f 13 37 5f04c37
65528 5e 16 39 5e05839
65529 6b 1d 3d 6b0743d
65530 77 2d 50 770b450
65531 7d 37 50 7d0dc50
65532 96 42 57 9610857
65533 a8 48 55 a812055
65534 b2 43 4c b210c4c
65535 b7 49 51 b712451
00000000 7C 24 22 0E 83 20 E2 0D 75 14 22 0E 7B 20 32 0E |$".. ..u.".{ 2.
00000010 7A 28 12 0E 7C 18 22 0E 70 18 02 0E 75 0C D2 0D z(..|.".p...u...
00000020 6D 10 C2 0D 76 2C E2 0D 6A 00 F2 0D 69 0C E2 0D m...v,..j...i...
00000030 6F 10 B2 0D 65 FC 01 0E 74 04 C2 0D 74 04 D2 0D o...e...t...t...
00000040 69 08 F2 0D 72 0C 02 0E 76 28 22 0E 70 24 62 0E i...r...v(".p$b.
Below is output of the jpeg.c compiled for zipcpu
pi@mypi3-1:~/testbuilds/testcatzipmypi3-4/catzip/sw/host $ ./arm-wrsdram rgb_pack.bin
The size of the buffer is 0x00ffff or 65535 words

READ-COMPLETE

pi@mypi3-1:~/testbuilds/testcatzipmypi3-4/catzip/sw/host $ ./arm-ziploa d -v ../board/jpeg
Halting the CPU
Memory regions:
Block RAM: 00500000 - 00502000
SDRAM : 00800000 - 01000000
Loading: ../board/jpeg
Section 0: 00800000 - 0080989c
Writing to MEM: 00800000-0080989c
Clearing the CPUs registers
Setting PC to 00800000
The CPU should be fully loaded, you may now
start it (from reset/reboot) with:
> wbregs cpu 0x0f

CPU Status is: 0000060f

pi@mypi3-1:~/testbuilds/testcatzipmypi3-4/catzip/sw/host $ ./arm-wbregs cpu 0x0f01000000 ( )-> 0000000f
. img_ptr = 0x8098a8
. buf_red = 0x8198b0
. buf_gr = 0x84b638
. buf_bl = 0x86b640
. fwd_inv = 0x8398b8
. Start of JPEG DWT!
. w = 0x100 h = 0x100
*. img_ptr = 0x8098a8 *img_ptr = 0xe22247c*
*. img_ptr = 0x8098ac *img_ptr = 0xde22083*
*. img_ptr = 0x8098b0 *img_ptr = 0xe221475*
*. img_ptr = 0x8098b4 *img_ptr = 0xe32207b*
*. img_ptr = 0x8098b8 *img_ptr = 0xe12287a*
.
*. img_ptr = 0x826d7c *img_ptr = 0xb611851*
*. img_ptr = 0x826d80 *img_ptr = 0xb812051*
*. img_ptr = 0x826d84 *img_ptr = 0xce1845b*
*. img_ptr = 0x826d88 *img_ptr = 0xc312c53*
*. img_ptr = 0x826d8c *img_ptr = 0xc31585a*
.
*. img_ptr = 0x849070 *img_ptr = 0x8b11058*
*. img_ptr = 0x849074 *img_ptr = 0x800f054*
*. img_ptr = 0x849078 *img_ptr = 0x7409c44*
*. img_ptr = 0x84907c *img_ptr = 0x680783b*
*. img_ptr = 0x849080 *img_ptr = 0x6608c43*
.
. im_s_ptr = 0x8098a8
*. buf_red = 0x8198b0 *buf_red = 0xe2*
*. buf_red = 0x8198b4 *buf_red = 0xde*
*. buf_red = 0x8198b8 *buf_red = 0xe2*
*. buf_red = 0x8198bc *buf_red = 0xe3*
*. buf_red = 0x8198c0 *buf_red = 0xe1*
.
*. buf_gr = 0x84b638 *buf_gr = 0x89*
*. buf_gr = 0x84b63c *buf_gr = 0x88*
*. buf_gr = 0x84b640 *buf_gr = 0x85*
*. buf_gr = 0x84b644 *buf_gr = 0x88*
*. buf_gr = 0x84b648 *buf_gr = 0x8a*
.
*. buf_bl = 0x86b640 *buf_bl = 0x7c*
*. buf_bl = 0x86b644 *buf_bl = 0x83*
*. buf_bl = 0x86b648 *buf_bl = 0x75*
*. buf_bl = 0x86b64c *buf_bl = 0x7b*
*. buf_bl = 0x86b650 *buf_bl = 0x7a*
.
. w = 0x100 buf_red wptr = 0x8198b0 alt = 0x8598b0 fwd_inverse = 0x8398b8 fwd_. verse = 0x1
. w = 0x100 buf_gr wptr1 = 0x84b638 alt1 = 0x88b638 fwd_inverse = 0x8398b8 fwd. nverse = 0x1

. w = 0x100 buf_bl wptr2 = 0x86b640 alt2 = 0x8ab640 fwd_inverse = 0x8398b8 fwd. nverse = 0x1
. all pointers for r g b dwt should be setup correctly
. starting red dwt
. in lifting
. in singlelift

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;
}
}

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


        int *im_s_ptr;
        int *red_s_ptr, *gr_s_ptr, *bl_s_ptr;
        int *wptr,*wptr1,*wptr2;
        int *alt,*alt1,*alt2;

        int *img_ptr;
        int *buf_red, *buf_gr, *buf_bl;
        int ur,ug,ub,x,y,z;
        int *fwd_inv;

        int val,i;
        int w,h;
        w = 256;
        h = 256;

        img_ptr = (int *)malloc( w*h);
        buf_red = (int *)malloc( w*h*2);
        buf_gr = (int *)malloc( w*h*2);
        buf_bl = (int *)malloc( w*h*2);
        fwd_inv = (int *)malloc(1);
```

```
        if(img_ptr == NULL) return 1;
        if(buf_red == NULL) return 2;

        if(buf_gr == NULL) return 3;
        if(buf_bl == NULL) return 4;
        if(fwd_inv == NULL) return 5;
        red_s_ptr = buf_red;
        gr_s_ptr = buf_gr;
        bl_s_ptr = buf_bl;
        //saving the img_ptr
        im_s_ptr = img_ptr;
        //*_gpio = GPIO_LEDR_SET;
        //printf("gpio 0x%x\n",_gpio);
        printf("img_ptr = 0x%x\n",img_ptr);
printf("buf_red = 0x%x\n",buf_red);
printf("buf_gr = 0x%x\n",buf_gr);
printf("buf_bl = 0x%x\n",buf_bl);
printf("fwd_inv = 0x%x\n",fwd_inv);

//printf("top_of_data = 0x%x\n",top_of_data);
        printf("Start of JPEG DWT!\n");




        //this was done by clr_sdram
        //printf("clearing buf_ptr + 786432 buf_ptr = 0x%x\n",img_ptr);
        //zip_clear_sdram(img_ptr);
        //printf("done clearing buf_ptr + 786432 \n");
        //printf("free img_ptr !\n");
        //const int bb = 0x1ff;
//const int gg = 0x7fc00;           shift right 8
//const int rr = 0x1ff00000;        shift right 20
        printf("w = 0x%x h = 0x%x\n",w,h);

        for(i=0;i<5;i++) {
                printf("img_ptr = 0x%x *img_ptr = 0x%x\n",img_ptr,*img_ptr);
                img_ptr ++;
        }
        printf("\n");

        img_ptr = img_ptr + 30000;
        for(i=0;i<5;i++) {
                printf("img_ptr = 0x%x *img_ptr = 0x%x\n",img_ptr,*img_ptr);
                img_ptr ++;
        }
        printf("\n");

        img_ptr = img_ptr + 35000;
```

```
for(i=0;i<5;i++) {
        printf("img_ptr = 0x%x *img_ptr = 0x%x\n",img_ptr,*img_ptr);
        img_ptr ++;
}
printf("\n");

printf("im_s_ptr = 0x%x\n",im_s_ptr);
//split red
img_ptr = im_s_ptr;
for(i=0;i<65535;i++) {
        x = *img_ptr;
        y = 0x1ff00000;
        z = x & y;
        ur = z>>20;
        img_ptr++;
        *buf_red = ur;
        buf_red++;
        //printf("%d 0x%x\n",i,ur);
}
//restore img_ptr & buf_red
img_ptr = im_s_ptr;
buf_red = red_s_ptr;

//split gr
for(i=0;i<65535;i++) {
        x = *img_ptr;
        y = 0x7fc00;
        z = x & y;
        ug = z>>10;
        img_ptr++;
        *buf_gr = ug;
        buf_gr++;
        //printf("%d 0x%x\n",i,ug);
}

//restore img_ptr & buf_gr
img_ptr = im_s_ptr;
buf_gr = gr_s_ptr;

//split bl
for(i=0;i<65535;i++) {
        x = *img_ptr;
        y = 0x3ff;
        z = x & y;
        ub = z;
        img_ptr++;
        *buf_bl = ub;
        buf_bl++;
        //printf("%d 0x%x\n",i,ub);
}

//restore img_ptr & buf_bl
```

```
	img_ptr = im_s_ptr;
	buf_bl = bl_s_ptr;

	//debug for r g b
	for(i=0;i<5;i++) {
		printf("buf_red = 0x%x *buf_red = 0x%x\n",buf_red,*buf_red);
		buf_red++;
	}
	printf("\n");

	buf_red = red_s_ptr;

	for(i=0;i<5;i++) {
		printf("buf_gr = 0x%x *buf_gr = 0x%x\n",buf_gr,*buf_gr);
		buf_gr++;
	}
	printf("\n");

	buf_gr = gr_s_ptr;

	for(i=0;i<5;i++) {
		printf("buf_bl = 0x%x *buf_bl = 0x%x\n",buf_bl,*buf_bl);
		buf_bl++;
	}
	printf("\n");

	buf_bl = bl_s_ptr;
	*fwd_inv = 1;


	alt = &buf_red[w*h];
	alt1 = &buf_gr[w*h];
	alt2 = &buf_bl[w*h];
	wptr = buf_red;
	wptr1 = buf_gr;
	wptr2 = buf_bl;
	printf("w = 0x%x buf_red wptr = 0x%x alt =  0x%x fwd_inverse =  0x%x fwd_inverse =
0x%x \n",w, wptr,alt,fwd_inv,*fwd_inv);
	printf("w = 0x%x buf_gr wptr1 = 0x%x alt1 =  0x%x fwd_inverse =  0x%x fwd_inverse =
0x%x \n",w, wptr1,alt1,fwd_inv,*fwd_inv);
	printf("w = 0x%x buf_bl wptr2 = 0x%x alt2 =  0x%x fwd_inverse =  0x%x fwd_inverse =
0x%x \n",w, wptr2,alt2,fwd_inv,*fwd_inv);
	printf("all pointers for r g b dwt should be setup correctly\n");


	printf("starting red dwt\n");




	lifting(w,wptr,alt,fwd_inv);
```

```
        printf("finished ted dwt\n");
        /*

        lifting(w,wptr1,alt1,fwd_inv);

        alt2 = &buf_bl[w*h];
        lifting(w,wptr2,alt2,fwd_inv);
        */


        free(img_ptr);
        free(buf_red);
        free(buf_gr);
        free(buf_bl);
        return 0;
        //*_gpio = 0x010000;
}
```

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.

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.

```
pi@mypi3-1:~/testbuilds/testcatzipmypi3-4/catzip/sw/host $ ./arm-ziploa
Halting the CPU
Memory regions:
        Block RAM: 00400000 - 00402000
```

SDRAM : 00800000 - 01000000
Loading: ../board/jpeg
Section 0: 00800000 - 008c94fc
Writing to MEM: 00800000-008c94fc
Clearing the CPUs registers
Setting PC to 00800000
The CPU should be fully loaded, you may now
start it (from reset/reboot) with:
> wbregs cpu 0x0f

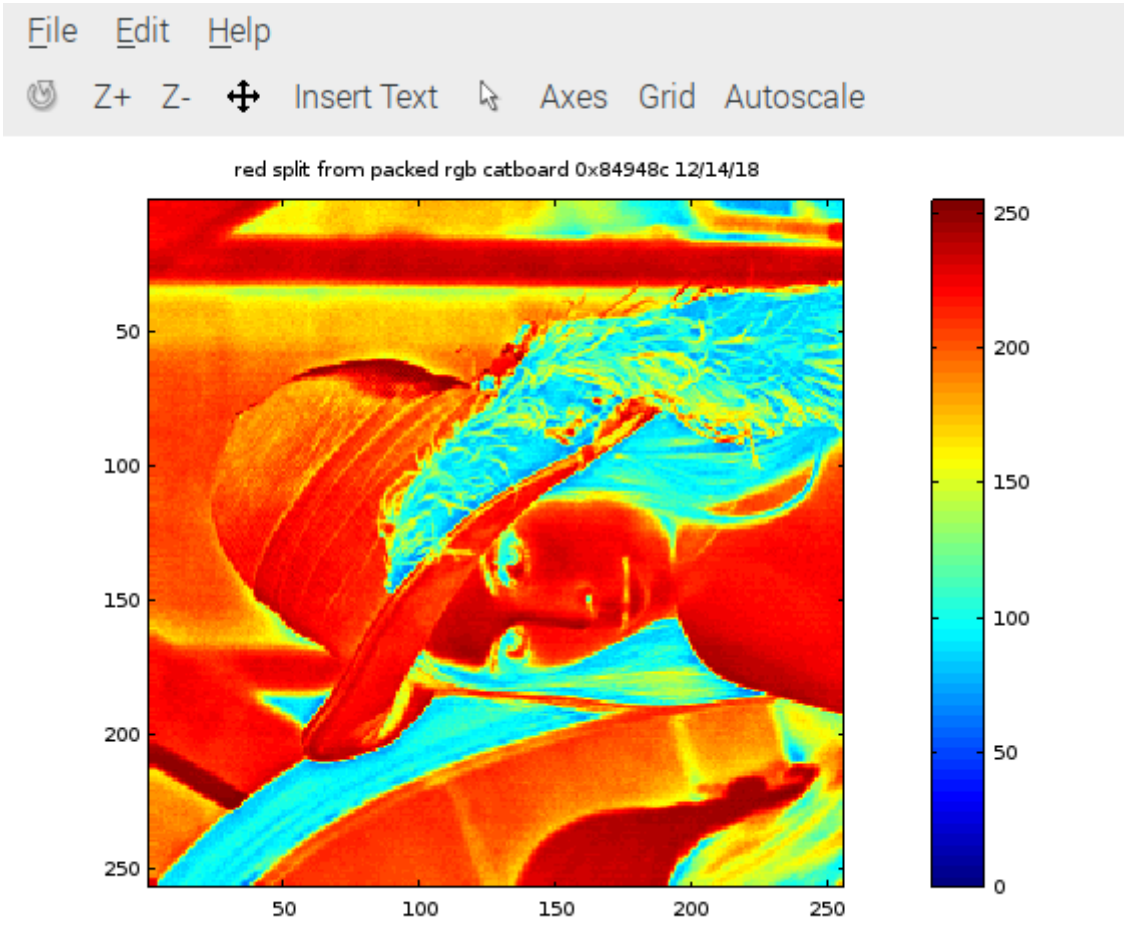
CPU Status is: 0000060f
pi@mypi3-1:~/testbuilds/testcatzipmypi3-4/catzip/sw/host $ ./arm-wrsdram rgb_pack.bin
The size of the buffer is 0x00ffff or 65535 words

READ-COMPLETE

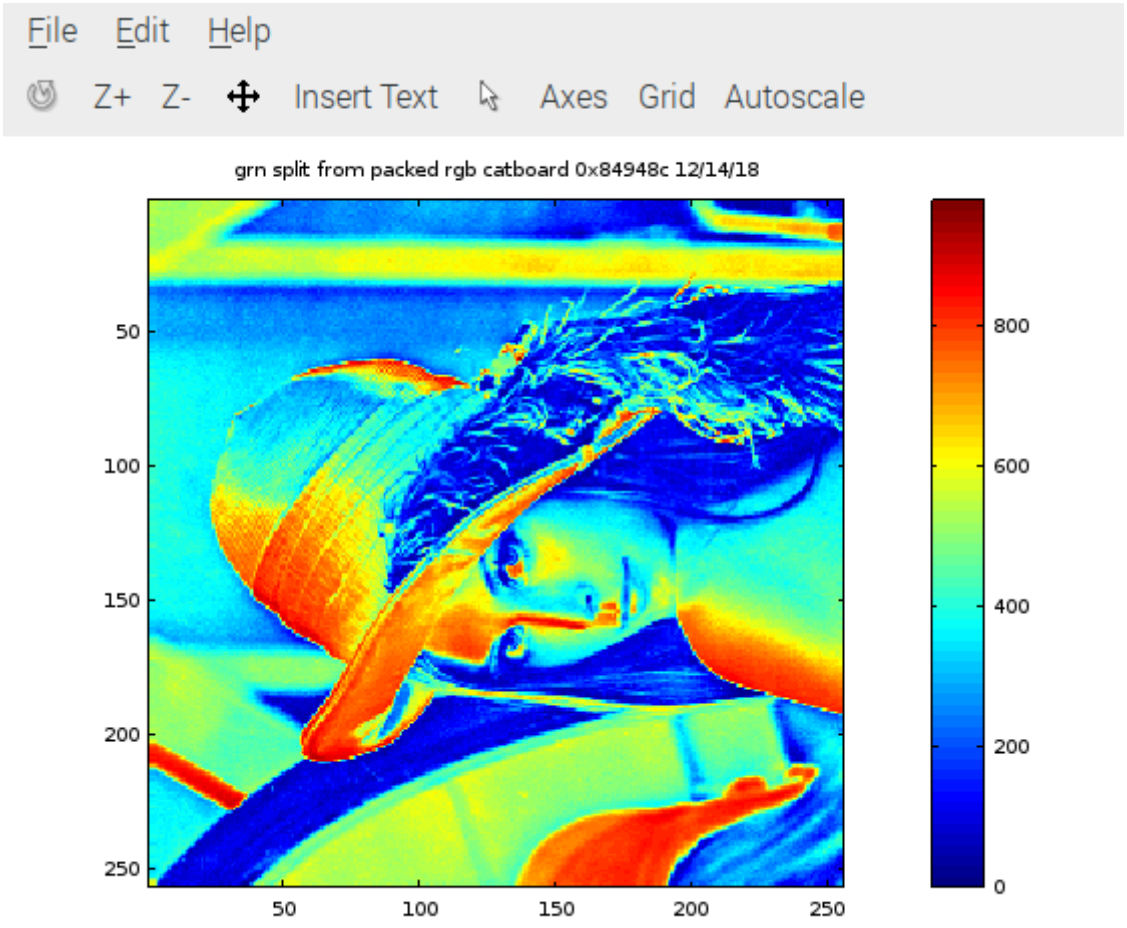
pi@mypi3-1:~/testbuilds/testcatzipmypi3-4/catzip/sw/host $ ./arm-rdsdram dwt.bin
Write-COMPLETE
The size of the buffer is 0x00ffff or 65535 words
red subband spiit using the latest jpeg.c reads the rgb_pack,bin

pi@mypi3-1:~/testbuilds/testcatzipmypi3-4/catzip/sw/host $ ./arm-wbregs cpu 0x0f01000000 ( )-> 0000000f

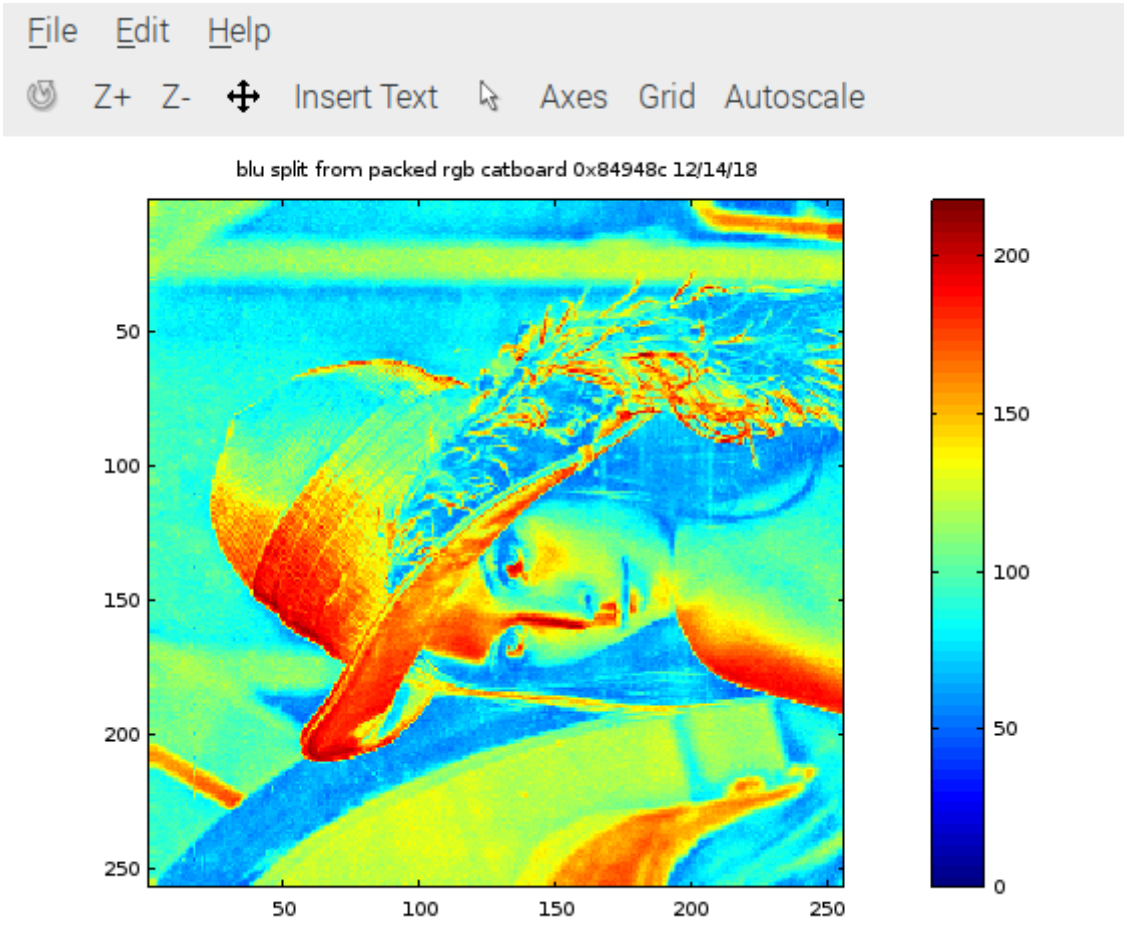
. w = 0x100 h = 0x100
. ptrs-->img = 0x80948c
. x = 0xe22247c sp = 0x7c z = 0x7c
. x = 0xde22083 sp = 0x83 z = 0x83
. x = 0xe221475 sp = 0x75 z = 0x75
. x = 0xe32207b sp = 0x7b z = 0x7b
. x = 0xa812055 sp = 0x55 z = 0x55
. x = 0xb210c4c sp = 0x4c z = 0x4c
. back from split start of dwt
. 0x7c
. 0x83
. 0x75
. 0x7b
. 0x7a
. 0x0
. 0x0
. 0x0
. 0x0
. 0x0
. 0x100 0x84948c 0x88948c



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

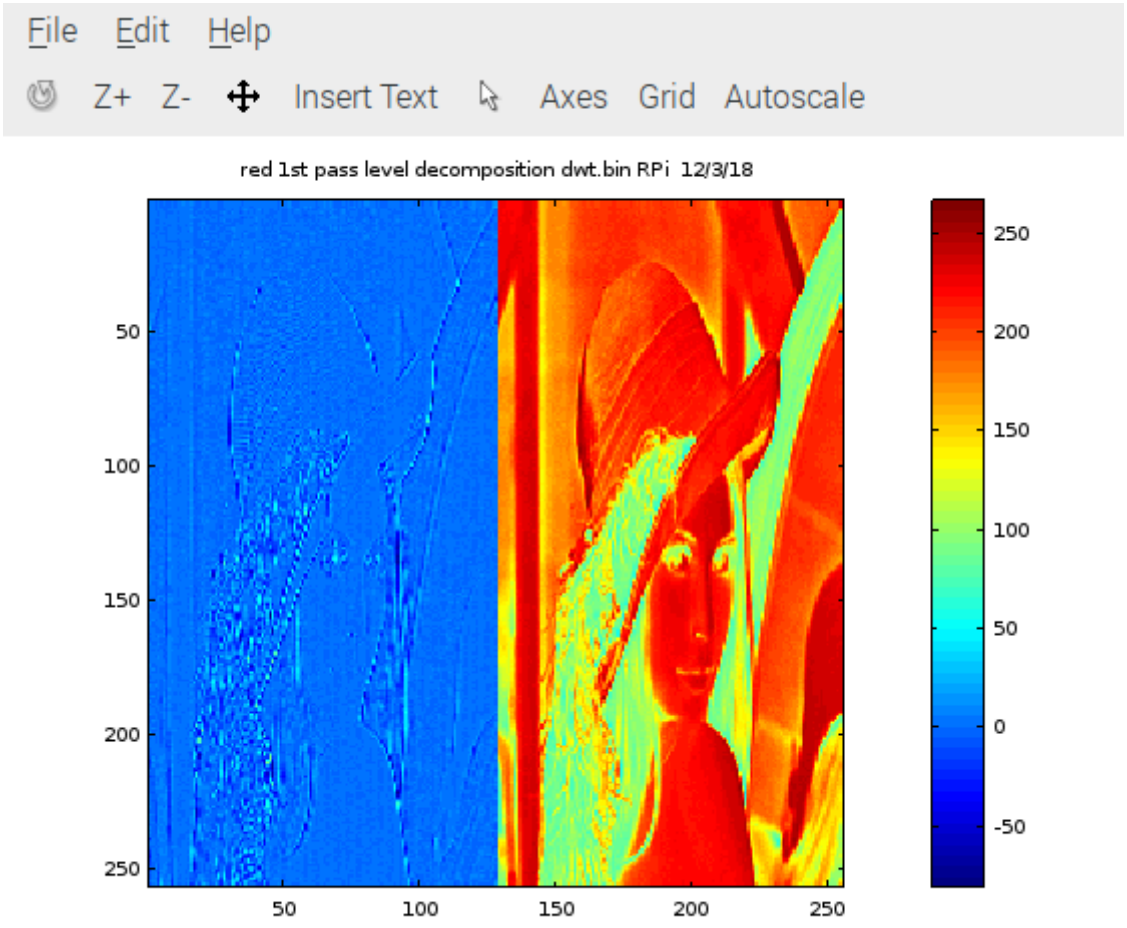


blu subband spiit using the latest jpeg.c reads the rgb_pack,bin

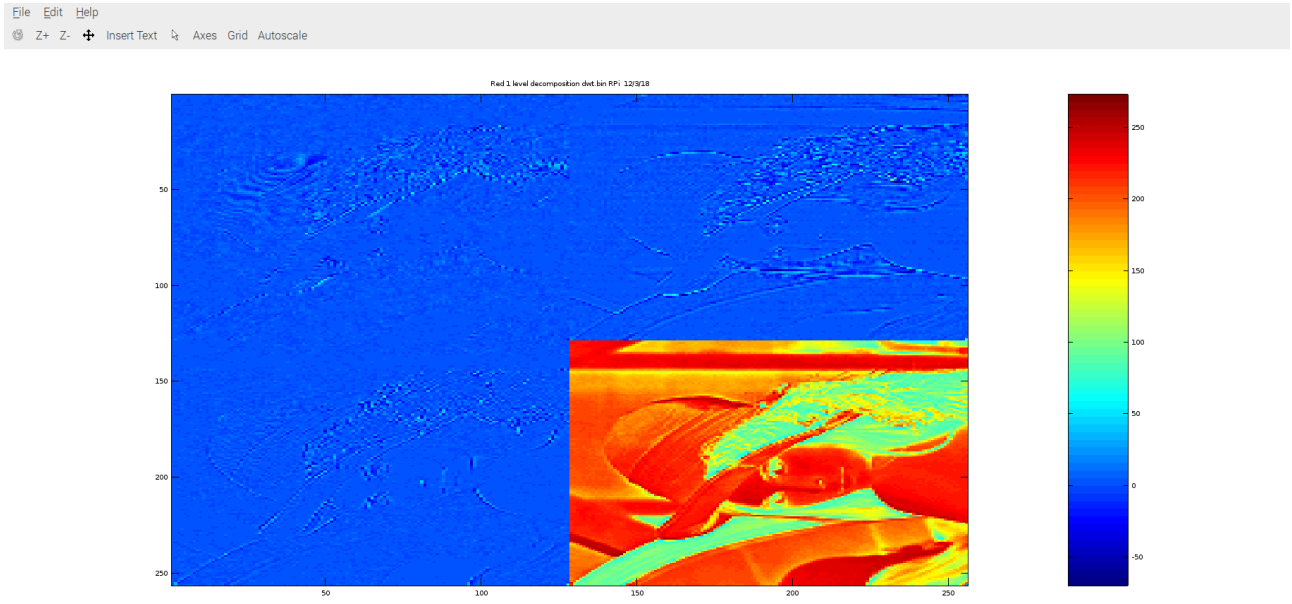


The next 4 image were obtained on the RPi3B running the lifting step in preparation of the catboard.

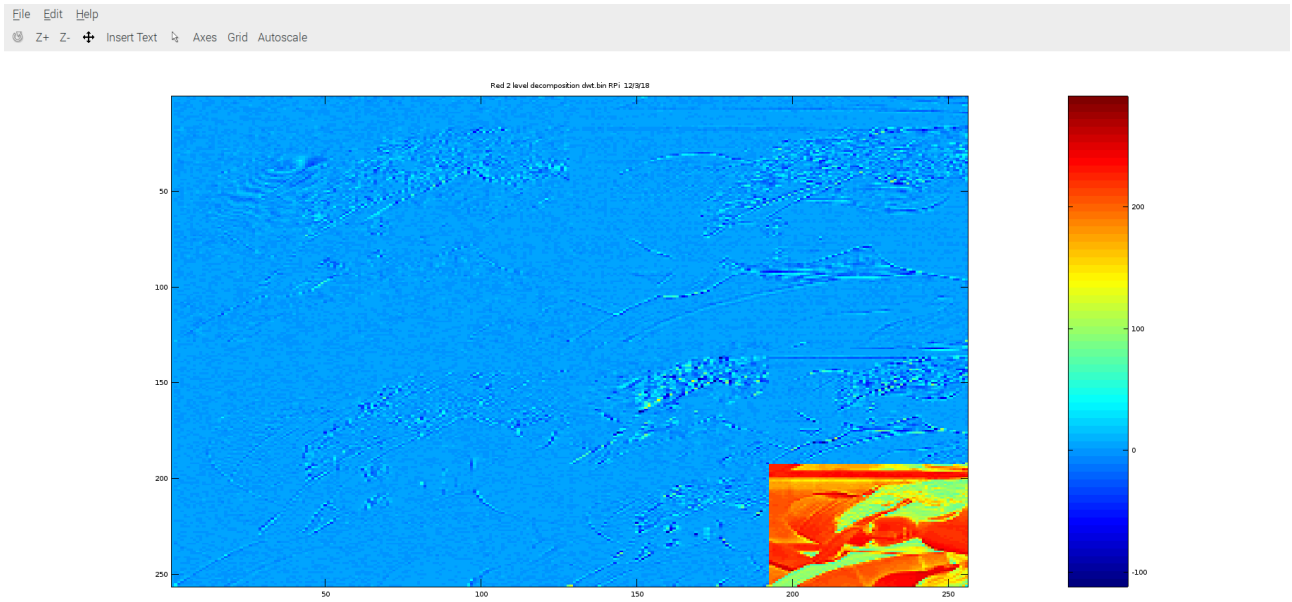
First step of the lifting step singlelift



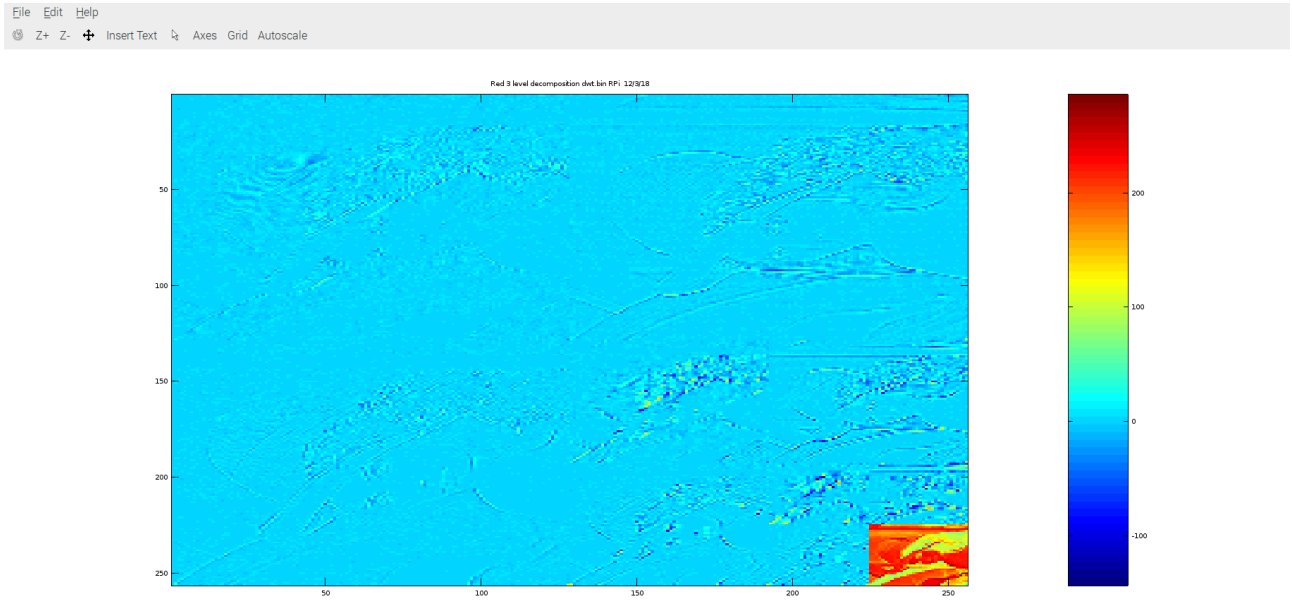
1 level of dwt image 128 X 128



2 levels of dwt image 64 x 64



3 levels of dwt image 32 x 32



Testing Remote Access & building on Ubuntu 16.04
modified the script to perform remote control access of the FPGA interfaced to the RPi3B+
testing build on ubuntu 16.04
Linux ws010 4.4.0-137-generic #163-Ubuntu SMP Mon Sep 24 13:14:43 UTC 2018 x86_64 x86_64 x86_64 GNU/Linux
cd testbuilds/

git clone https://github.com/develone/catzip.git
The build proces on Ubuntu take a relative short time.
-rw-rw-r-- 1 vidal vidal 6373 Oct 30 10:55 Makefile
-rw-rw-r-- 1 vidal vidal 135100 Oct 30 11:03 rtl/catzip/catzip.bin
./rtl/pptest/linepp.bin
./rtl/pptest/hellopp.bin
./rtl/pptest/speechpp.bin
./rtl/switch_leds/switch_leds.bin
./rtl/uart/speechfifo.bin
./rtl/uart/helloworld.bin
./rtl/catzip/catzip.bin
./rtl/leddigits/leddigits.bin
./rtl/basic/pmodtest.bin
./rtl/basic/dimmer.bin
./rtl/basic/blinky.bin
./rtl/basic/clktest.bin

pi@mypi3-1:~/testbuilds/ws010/catzip/sw/host $ diff rem_test_sim102218.sh test_sim102218.sh
2,18c2,18
< ./pc-wbregs -n mypi3-1 version
< ./pc-wbregs -n mypi3-1 0x2000004 0x55aaaa55
< ./pc-wbregs -n mypi3-1 0x2000004
< ./pc-wbregs -n mypi3-1 0x20ffff8 0x55aaaa55
< ./pc-wbregs -n mypi3-1 0x20ffff8
< ./pc-wbregs -n mypi3-1 0x2800004 0x55aaaa55
< ./pc-wbregs -n mypi3-1 0x2800004

< ./pc-wbregs -n mypi3-1 0x28ffff8 0x55aaaa55
< ./pc-wbregs -n mypi3-1 0x28ffff8
< ./pc-wbregs -n mypi3-1 0x2c00004 0x55aaaa55
< ./pc-wbregs -n mypi3-1 0x2c00004
< ./pc-wbregs -n mypi3-1 0x2cffff8 0x55aaaa55
< ./pc-wbregs -n mypi3-1 0x2cffff8
< ./pc-wbregs -n mypi3-1 0x2f00004 0x55aaaa55
< ./pc-wbregs -n mypi3-1 0x2f00004
< ./pc-wbregs -n mypi3-1 0x2fffff8 0x55aaaa55
< ./pc-wbregs -n mypi3-1 0x2fffff8

> ./arm-wbregs version
> ./arm-wbregs 0x2000004 0x55aaaa55
> ./arm-wbregs 0x2000004
> ./arm-wbregs 0x20ffff8 0x55aaaa55
> ./arm-wbregs 0x20ffff8
> ./arm-wbregs 0x2800004 0x55aaaa55
> ./arm-wbregs 0x2800004
> ./arm-wbregs 0x28ffff8 0x55aaaa55
> ./arm-wbregs 0x28ffff8
> ./arm-wbregs 0x2c00004 0x55aaaa55
> ./arm-wbregs 0x2c00004
> ./arm-wbregs 0x2cffff8 0x55aaaa55
> ./arm-wbregs 0x2cffff8
> ./arm-wbregs 0x2f00004 0x55aaaa55
> ./arm-wbregs 0x2f00004
> ./arm-wbregs 0x2fffff8 0x55aaaa55
> ./arm-wbregs 0x2fffff8
pi@mypi3-1:~/testbuilds/ws010/catzip/sw/host $ diff sim_hw_test.sh rem_sim_hw_test.sh
5c5
< ./arm-wbregs version

> ./pc-wbregs -n mypi3-1 version
7c7
< ./arm-wbregs 0x1000000 0x10000001

> ./pc-wbregs -n mypi3-1 0x1000000 0x10000001
9c9
< ./arm-wbregs 0x1000004 0x10000002

> ./pc-wbregs -n mypi3-1 0x1000004 0x10000002
11c11
< ./arm-wbregs 0x1000008 0x10000003

> ./pc-wbregs -n mypi3-1 0x1000008 0x10000003
13c13
< ./arm-wbregs 0x100000c 0x10000004

> ./pc-wbregs -n mypi3-1 0x100000c 0x10000004
15c15
< ./arm-wbregs 0x1000000 0x10000001

```
> ./pc-wbregs -n mypi3-1  0x1000000 0x10000001
17c17
< ./arm-wbregs 0x1000004 0x10000002
---
> ./pc-wbregs -n mypi3-1  0x1000004 0x10000002
19c19
< ./arm-wbregs 0x1000008 0x10000003
---
> ./pc-wbregs -n mypi3-1  0x1000008 0x10000003
21c21
< ./arm-wbregs 0x100000c 0x10000004
---
> ./pc-wbregs -n mypi3-1  0x100000c 0x10000004
23c23
< ./arm-wbregs 0x1000000
---
> ./pc-wbregs -n mypi3-1  0x1000000
25c25
< ./arm-wbregs 0x1000004
---
> ./pc-wbregs -n mypi3-1  0x1000004
27c27
< ./arm-wbregs 0x1000008
---
> ./pc-wbregs -n mypi3-1  0x1000008
29c29
< ./arm-wbregs 0x100000c
---
> ./pc-wbregs -n mypi3-1  0x100000c
31c31
< ./arm-wbregs 0x1000000
---
> ./pc-wbregs -n mypi3-1  0x1000000
33c33
< ./arm-wbregs 0x1000004
---
> ./pc-wbregs -n mypi3-1  0x1000004
35c35
< ./arm-wbregs 0x1000008
---
> ./pc-wbregs -n mypi3-1  0x1000008
37c37
< ./arm-wbregs 0x100000c
---
> ./pc-wbregs -n mypi3-1  0x100000c
40c40
< ./arm-wbregs gpio 0x00010001
---
> ./pc-wbregs -n mypi3-1  gpio 0x00010001
43c43
< ./arm-wbregs gpio 0x00020002
---
```

> ./pc-wbregs -n mypi3-1 gpio 0x00020002
46c46
< ./arm-wbregs gpio 0x00040004

> ./pc-wbregs -n mypi3-1 gpio 0x00040004
49c49
< ./arm-wbregs gpio 0x00070000

> ./pc-wbregs -n mypi3-1 gpio 0x00070000
Adding SDRAM to catzip
Starting with this version of catzip
catzip
commit 1e0dd38e890187ae3d6629fd35ddeeaad39a6701
Author: Edward Vidal Jr <develone@sbcglobal.net>
Date: Mon Aug 13 02:04:34 2018 +0000

*   speechfifo not working*
Using this version of learning_hdl to copy the files to add
the sdram support to catzip

commit 59da9a3ae53a03c20d516ee4c75da5597e448205
Author: Edward Vidal Jr <develone@sbcglobal.net>
Date: Wed Oct 24 16:51:00 2018 -0600

*   using hardware*

./config_catzip_simulation.sh

~/testbuilds/catzip_simulation/catzip
. myenv.sh
make datestamp
make autodata

~/testbuilds/catzip_simulation/catzip/rtl/catzip
. ../../myenv.sh
make clean
make cpudefs.h
make design.h
make verilated
make bin
sudo config_cat catzip.bin

~/testbuilds/catzip_simulation/catzip/sim/verilated
. ../../myenv.sh
make clean
make

~/testbuilds/catzip_simulation/catzip/sw/host
. ../../myenv.sh
make clean
make
~/testbuilds/catzip_simulation/catzip/sw/host $ ./arm-netpport

```
pi@mypi3-1:~/testbuilds/catzip_simulation/catzip/sw/host $ ./test_sim102218.sh
00c00010 ( VERSION) : [...)] 20181029
02000004 (       )-> 55aaaa55
02000004 (       ) : [.U.U] aa55aa55
020ffff8 (       )-> 55aaaa55
020ffff8 (       ) : [.U.U] aa55aa55
02800004 (       )-> 55aaaa55
02800004 (       ) : [.U.U] aa55aa55
028ffff8 (       )-> 55aaaa55
028ffff8 (       ) : [.U.U] aa55aa55
02c00004 (       )-> 55aaaa55
02c00004 (       ) : [.U.U] aa55aa55
02cffff8 (       )-> 55aaaa55
02cffff8 (       ) : [.U.U] aa55aa55
02f00004 (       )-> 55aaaa55
02f00004 (       ) : [.U.U] aa55aa55
02fffff8 (       )-> 55aaaa55
02fffff8 (       ) : [.U.U] aa55aa55
pi@mypi3-1:~/testbuilds/catzip_simulation/catzip/rtl/catzip $ diff wbsdram.v
~/testbuilds/xulalx25soc/rtl/wbsdram.v
66c66
<       parameter [0:0]       F_OPT_CLK2FFLOGIC = 1'b1;
---
>       parameter [0:0]       F_OPT_CLK2FFLOGIC = 1'b0;
122,123c122
<                       refresh_clk <= 10'd391; // Make suitable for 50 MHz clk
<                       //refresh_clk <= 10'd625; // Make suitable for 80 MHz clk
---
>                       refresh_clk <= 10'd625; // Make suitable for 80 MHz clk
183c182
<       //initial       fwd_addr = 1;
---
>       initial  fwd_addr = 1;
cd ../basic
~/testbuilds/catzip_simulation/catzip/rtl/basic
make clean
make
sudo config_cat blinky.bin
sudo config_cat dimmer.bin
sudo config_cat clktest.bin
pmodtest.bin not testing.
cd ../leddigits/
~/testbuilds/catzip_simulation/catzip/rtl/leddigits
make clean
make
sudo config_cat leddigits.bin
cd ../switch_leds/
~/testbuilds/catzip_simulation/catzip/rtl/switch_leds
make clean
make
sudo config_cat switch_leds.bin
```

```
cd ../pptest/
~/testbuilds/catzip_simulation/catzip/rtl/pptest
make clean
make
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.                                                             |
. |                                                                |
. |                                                                |
. |================================================================|
cd ../uart/
~/testbuilds/catzip_simulation/catzip/rtl/uart
make clean
make
sudo config_cat helloworld.bin

pi@mypi3-1:~/testbuilds/catzip_simulation/catzip/sw/host $ ./sim_hw_test.sh
he date built
00c00010 ( VERSION) : [...)] 20181029
01000000 (     RAM)-> 10000001
01000004 (        )-> 10000002
01000008 (        )-> 10000003
0100000c (        )-> 10000004
01000000 (     RAM)-> 10000001
```

01000004 (   )-> 10000002
01000008 (   )-> 10000003
0100000c (   )-> 10000004
01000000 (RAM) : [....] 10000001
01000004 (   ) : [....] 10000002
01000008 (   ) : [....] 10000003
0100000c (   ) : [....] 10000004
01000000 (RAM) : [....] 10000001
01000004 (   ) : [....] 10000002
01000008 (   ) : [....] 10000003
0100000c (   ) : [....] 10000004
Turning on the 4th led
00c00008 (GPIO)-> 00010001
Turning on the 1st led
00c00008 (GPIO)-> 00020002
Turning on the 2nd led
00c00008 (GPIO)-> 00040004
Turning off the leds
00c00008 (GPIO)-> 00070000
Versions of support sopport software to
****
autofpga
commit 2f443503d1edcff1a401d30207790906cb01df32
Author: ZipCPU <dgisselq@ieee.org>
Date: Sun Aug 5 20:55:25 2018 -0400

*   Fixed an uninitialized error in the subdirectory string*
icestorm
commit 8cac6c584044034210fe0ba1e6b930ff1cc59465
Author: Clifford Wolf <clifford@clifford.at>
Date: Mon Jul 30 16:04:04 2018 +0200

*   Also install text timing databases*

*   Signed-off-by: Clifford Wolf <clifford@clifford.at>*
arachne-pnr
commit 5d830dd94ad956d17d77168fe7718f22f8b55b33
Merge: 3a40328 9763e6e
Author: Clifford Wolf <clifford@clifford.at>
Date: Sun May 13 20:58:41 2018 +0200

*   Merge pull request #115 from awygle/lm*

*   Add basic lm4k support (no hard IP)*
yosys
commit e275692e84c935d0cdf42c2a4adf7ac949a88132
Author: Clifford Wolf <clifford@clifford.at>
Date: Sun Jul 22 18:44:05 2018 +0200

*   Verific: Produce errors for instantiating unknown module*

*   Because if the unknown module is connected to any constants, Verific will*

actually break all constants in the same module, even if they have nothing
to do structurally with that instance of an unknown module.

Signed-off-by: Clifford Wolf <clifford@clifford.at>
commit ab82a886305ceec79b5516d7fc356f95a762c9fd
Author: ZipCPU <dgisselq@ieee.org>
Date: Fri Apr 20 12:49:05 2018 -0400

Renamed the autoreload value of the ziptimer to be the interval count
verilator v3.926
commit c8e437c45cf0a9849e0a0465ecca882dbb66933a
Author: Wilson Snyder <wsnyder@wsnyder.org>
Date: Wed Aug 22 18:09:06 2018 -0400

Version bump
The following information is from https://github.com/ZipCPU/zipcpu/blob/master/doc/spec.pdf

Introduction
The goal of the ZipCPU was to be a very simple CPU. You might think of it as a poor manâ 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"

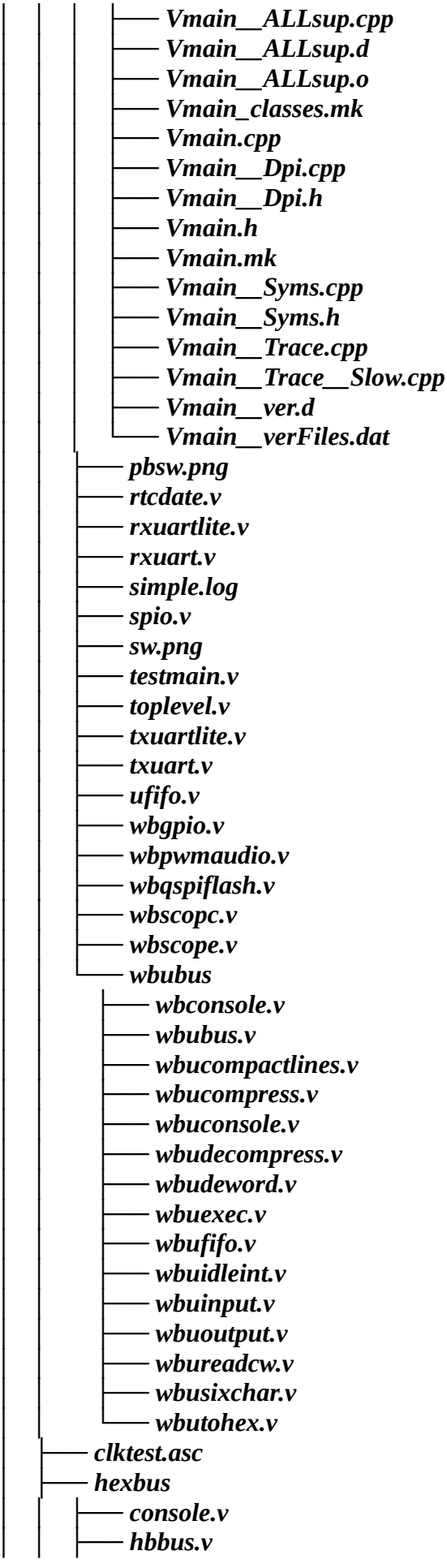
"cd catzip"

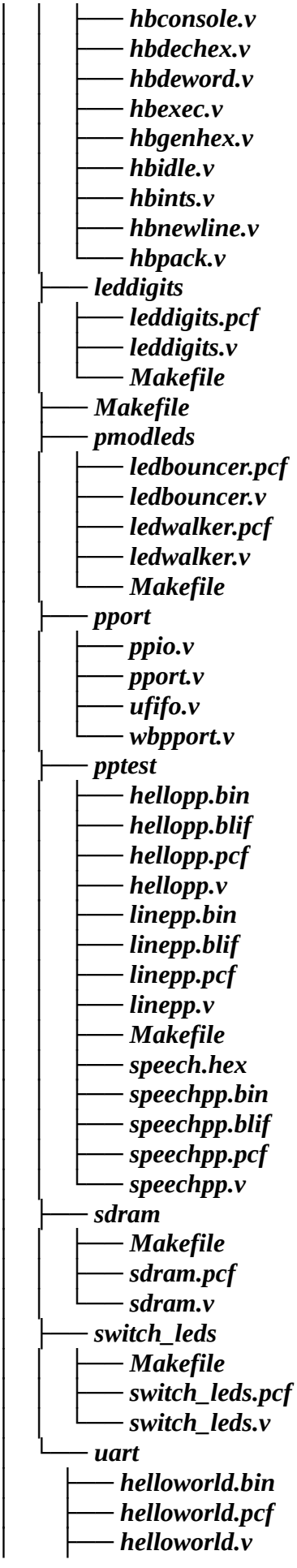
"make clean"

“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 2 executeables arm-netpport & arm-wbregs used to communicate with the FPGA.

```
├── 20180605-build.v
├── 20180605-cat.tjz
├── auto-data
│   ├── bkram.txt
│   ├── board.h
│   ├── board.ld
│   ├── buserr.txt
│   ├── clock.txt
│   ├── dlyarbiter.txt
│   ├── global.txt
│   ├── gpio.txt
│   ├── hbconsole.txt
│   ├── legalgen.txt
│   ├── main_tb.cpp
│   ├── main.v
│   ├── Makefile
│   ├── pic.txt
│   ├── pwrcount.txt
│   ├── regdefs.cpp
│   ├── regdefs.h
│   ├── rtl.make.inc
│   ├── testb.h
│   ├── toplevel.v
│   ├── version.txt
│   ├── wbuconsole.txt
│   └── zipbones.txt
├── diff060418.txt
├── doc
│   ├── Makefile
│   ├── remotefpga.odt
│   ├── remotefpga.pdf
│   ├── src
│   │   └── gpl-3.0.tex
│   ├── zipcpu.odt
│   └── zipcpu.pdf
├── Makefile
├── mkdatev.pl
├── README.md
├── rtl
│   ├── basic
│   │   ├── blinky.bin
│   │   ├── blinky.pcf
│   │   ├── blinky.v
│   │   ├── clktest.bin
```

```
│   │   ├── clktest.pcf
│   │   ├── clktest.v
│   │   ├── dimmer.bin
│   │   ├── dimmer.pcf
│   │   ├── dimmer.v
│   │   ├── Makefile
│   │   ├── notes.txt
│   │   ├── pmodtest.bin
│   │   ├── pmodtest.pcf
│   │   └── pmodtest.v
│   ├── catzip
│   │   ├── auto.mk
│   │   ├── builddate.v
│   │   ├── catzip.pcf
│   │   ├── cpu
│   │   │   ├── busdelay.v
│   │   │   ├── cpudefs.v
│   │   │   ├── cpuops.v
│   │   │   ├── dblfetch.v
│   │   │   ├── div.v
│   │   │   ├── icontrol.v
│   │   │   ├── idecode.v
│   │   │   ├── memops.v
│   │   │   ├── pfcache.v
│   │   │   ├── pipemem.v
│   │   │   ├── prefetch.v
│   │   │   ├── wbarbiter.v
│   │   │   ├── wbdblpriarb.v
│   │   │   ├── wbdmac.v
│   │   │   ├── wbpriarbiter.v
│   │   │   ├── wbwatchdog.v
│   │   │   ├── zipbones.v
│   │   │   ├── zipcounter.v
│   │   │   ├── zipcpu.v
│   │   │   ├── zipjiffies.v
│   │   │   ├── zipsystem.v
│   │   │   └── ziptimer.v
│   │   ├── cpudefs.h
│   │   ├── dbg.png
│   │   ├── design.h
│   │   ├── flashconfig.v
│   │   ├── hdr.png
│   │   ├── icozip.pcf
│   │   ├── leds.png
│   │   ├── main.v
│   │   ├── Makefile
│   │   ├── memdev.v
│   │   ├── obj-arm
│   │   │   ├── Vmain__ALL.a
│   │   │   ├── Vmain__ALLcls.cpp
│   │   │   ├── Vmain__ALLcls.d
│   │   │   ├── Vmain__ALLcls.o
```


Vmain__ALLsup.cpp
Vmain__ALLsup.d
Vmain__ALLsup.o
Vmain_classes.mk
Vmain.cpp
Vmain__Dpi.cpp
Vmain__Dpi.h
Vmain.h
Vmain.mk
Vmain__Syms.cpp
Vmain__Syms.h
Vmain__Trace.cpp
Vmain__Trace__Slow.cpp
Vmain__ver.d
Vmain__verFiles.dat
pbsw.png
rtcdate.v
rxuartlite.v
rxuart.v
simple.log
spio.v
sw.png
testmain.v
toplevel.v
txuartlite.v
txuart.v
ufifo.v
wbgpio.v
wbpwmaudio.v
wbqspiflash.v
wbscopc.v
wbscope.v
wbubus
wbconsole.v
wbubus.v
wbucompactlines.v
wbucompress.v
wbuconsole.v
wbudecompress.v
wbudeword.v
wbuexec.v
wbufifo.v
wbuidleint.v
wbuinput.v
wbuoutput.v
wbureadcw.v
wbusixchar.v
wbutohex.v
clktest.asc
hexbus
console.v
hbbus.v


hbconsole.v
hbdechex.v
hbdeword.v
hbexec.v
hbgenhex.v
hbidle.v
hbints.v
hbnewline.v
hbpack.v
leddigits
leddigits.pcf
leddigits.v
Makefile
Makefile
pmodleds
ledbouncer.pcf
ledbouncer.v
ledwalker.pcf
ledwalker.v
Makefile
pport
ppio.v
pport.v
ufifo.v
wbpport.v
pptest
hellopp.bin
hellopp.blif
hellopp.pcf
hellopp.v
linepp.bin
linepp.blif
linepp.pcf
linepp.v
Makefile
speech.hex
speechpp.bin
speechpp.blif
speechpp.pcf
speechpp.v
sdram
Makefile
sdram.pcf
sdram.v
switch_leds
Makefile
switch_leds.pcf
switch_leds.v
uart
helloworld.bin
helloworld.pcf
helloworld.v

```
│       ├── Makefile
│       ├── notwkg_speechfifo.png
│       ├── README.md
│       ├── speechfifo.bin
│       ├── speechfifo.pcf
│       ├── speechfifo.v
│       ├── speechfifo.v.notwkg
│       ├── speechfifo.v.wkg
│       ├── speech.hex
│       └── wkg_speechfifo.png
├── sim
│   └── verilated
│       ├── automaster_tb.cpp
│       ├── byteswap.cpp
│       ├── byteswap.h
│       ├── dblpipecmdr.cpp
│       ├── dblpipecmdr.h
│       ├── hellopp.cpp
│       ├── linepp.cpp
│       ├── main_tb.cpp
│       ├── Makefile
│       ├── memsim.cpp
│       ├── memsim.h
│       ├── obj-arm
│       │   ├── depends.txt
│       │   ├── verilated.o
│       │   ├── verilated_vcd_c.o
│       │   └── xdepends.txt
│       ├── port.h -> ../../sw/host/port.h
│       ├── pportsim.cpp
│       ├── pportsim.h
│       ├── qspiflashsim.cpp
│       ├── qspiflashsim.h
│       ├── regdefs.h
│       ├── speechpp.cpp
│       ├── tags
│       ├── tbclock.h
│       ├── testb.h
│       ├── twoc.cpp
│       ├── twoc.h
│       ├── uartsim.cpp
│       ├── uartsim.h
│       ├── zipelf.cpp
│       └── zipelf.h
├── sim-err.txt
```

Initial testing
06/05/18

cd catzip/rtl/catzip

sudo config_cat catzip.bin This programs the FPGA

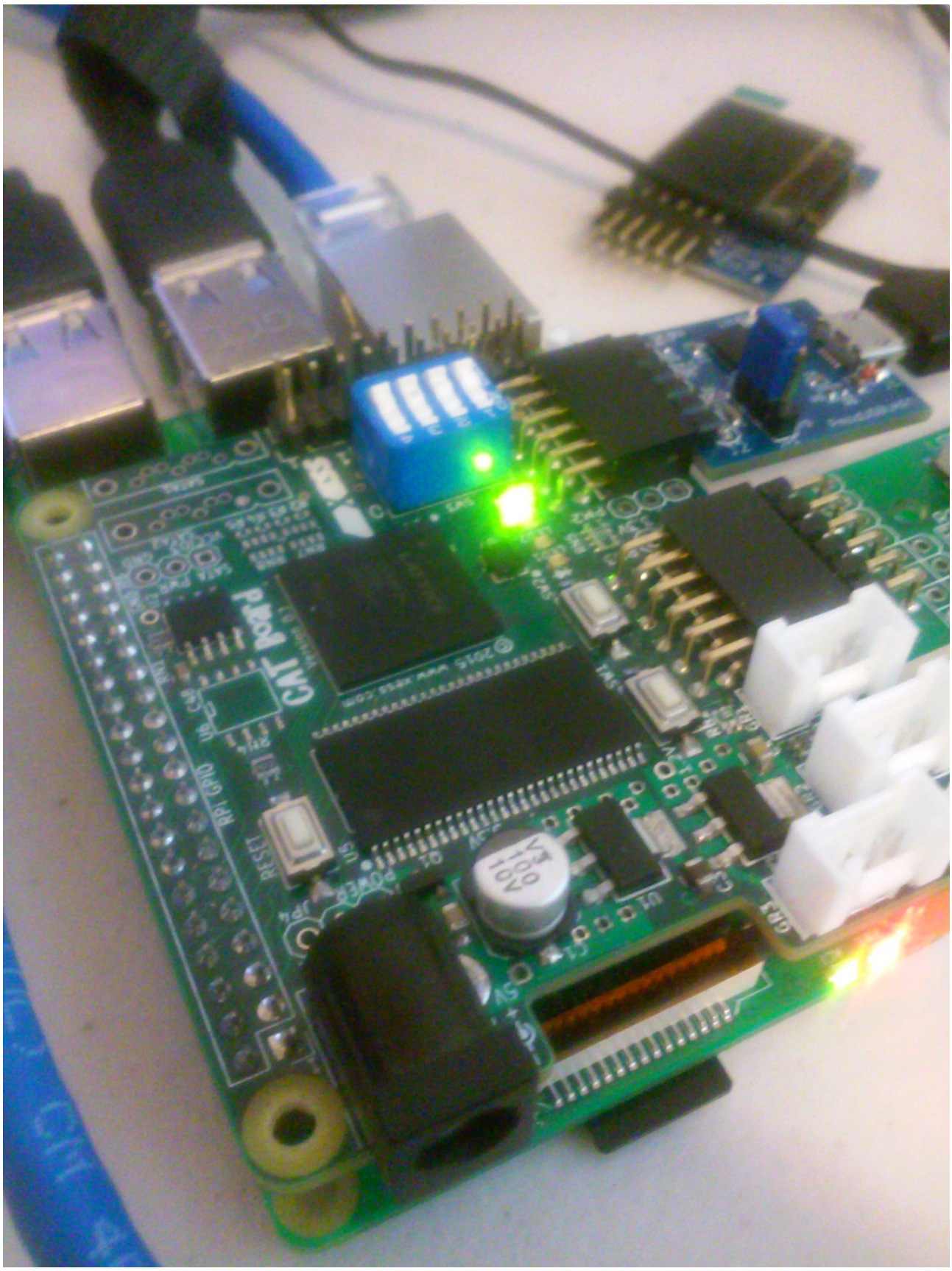
cd catzip/sw/host

./arm-netpport
Listening on port 8363
Listening on port 8364

cd catzip/sw/host

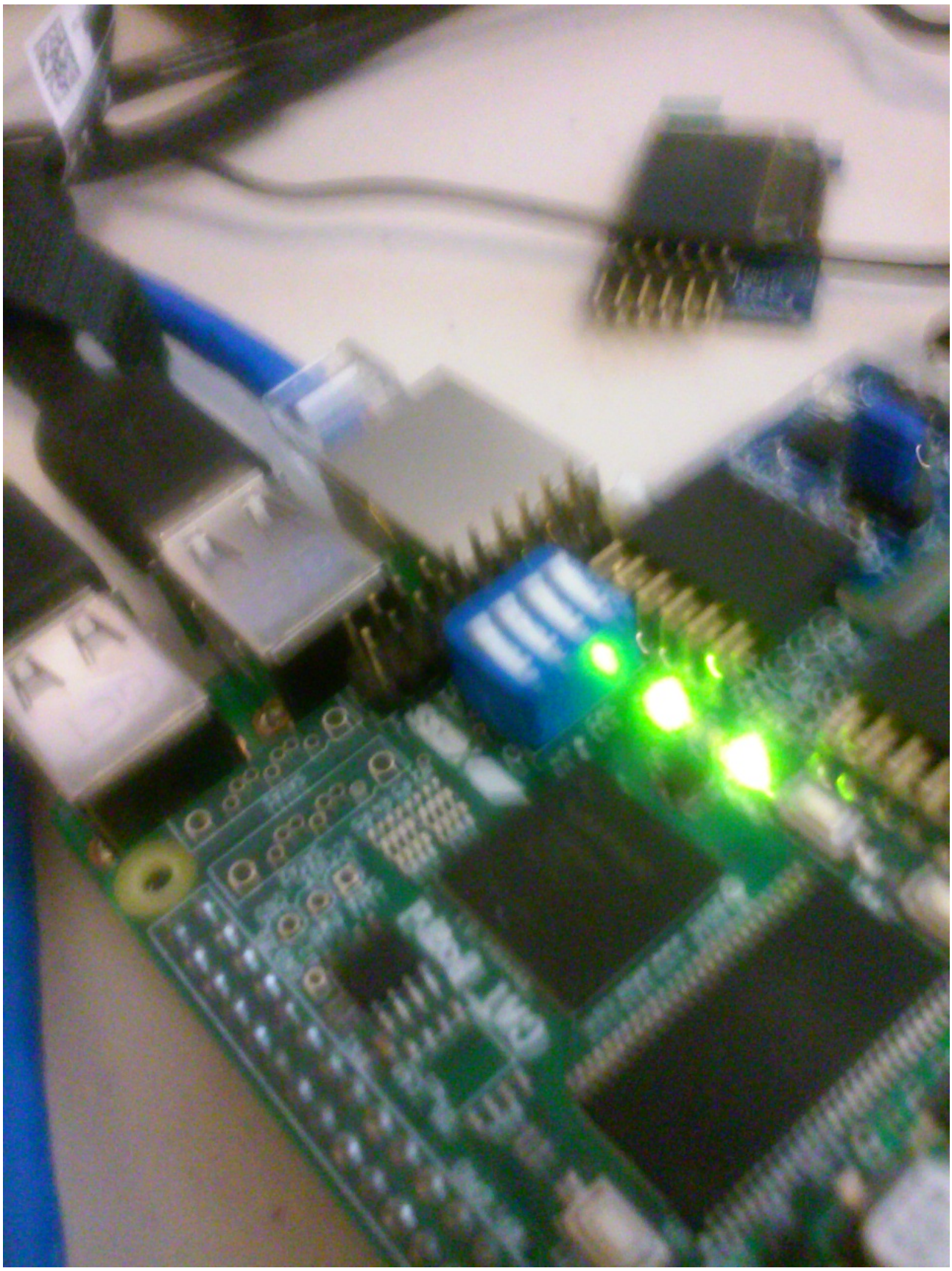
./arm-wbregs version
00001010 (VERSION) : [....] 20180605

1st Led on
/arm-wbregs gpio 0x00010001
00001008 (GPIO)-> 00010001


CMT Board
RESET

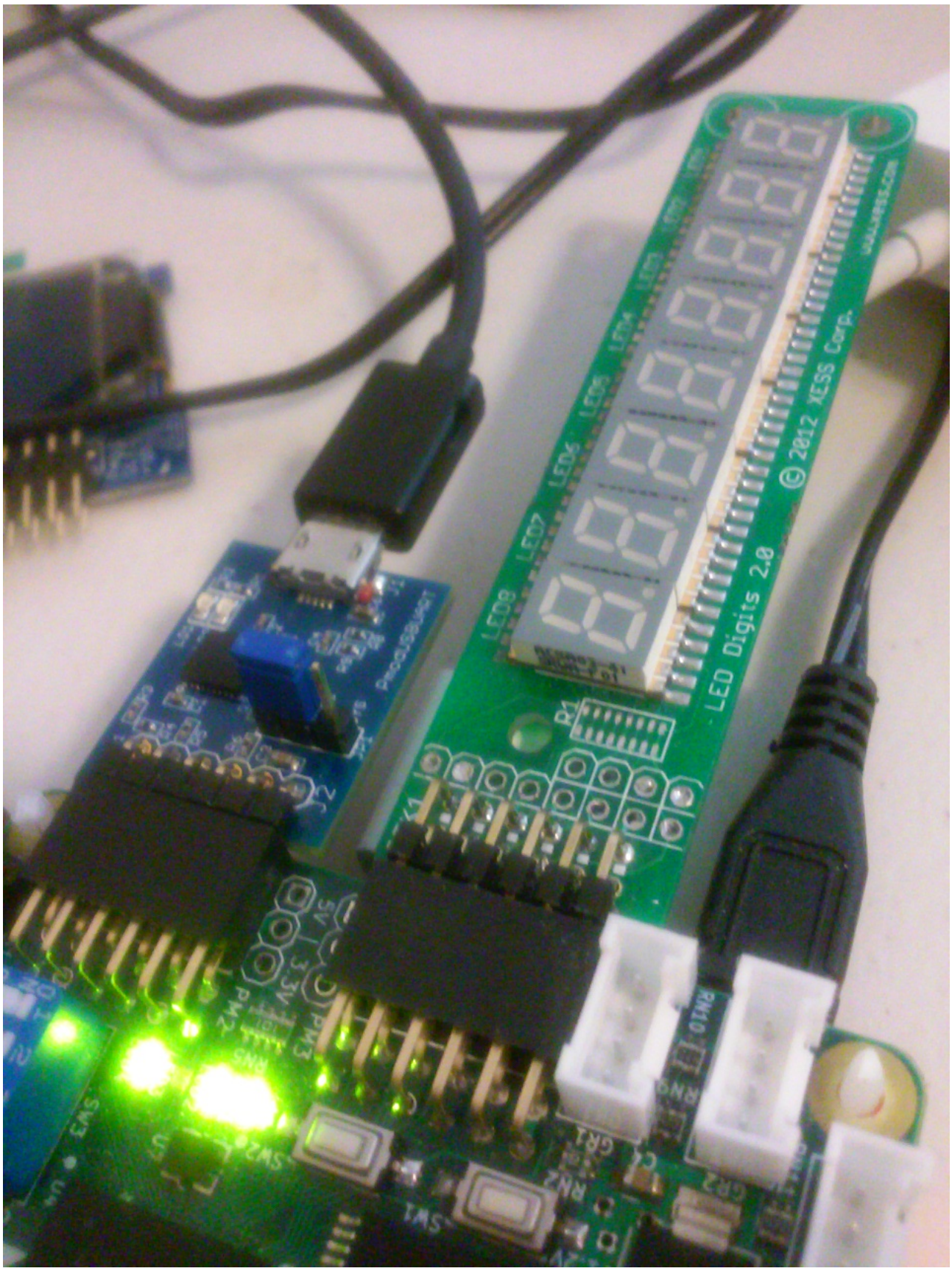
1st Led off
./arm-wbregs gpio 0x00010000
00001008 (GPIO)-> 00010000

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

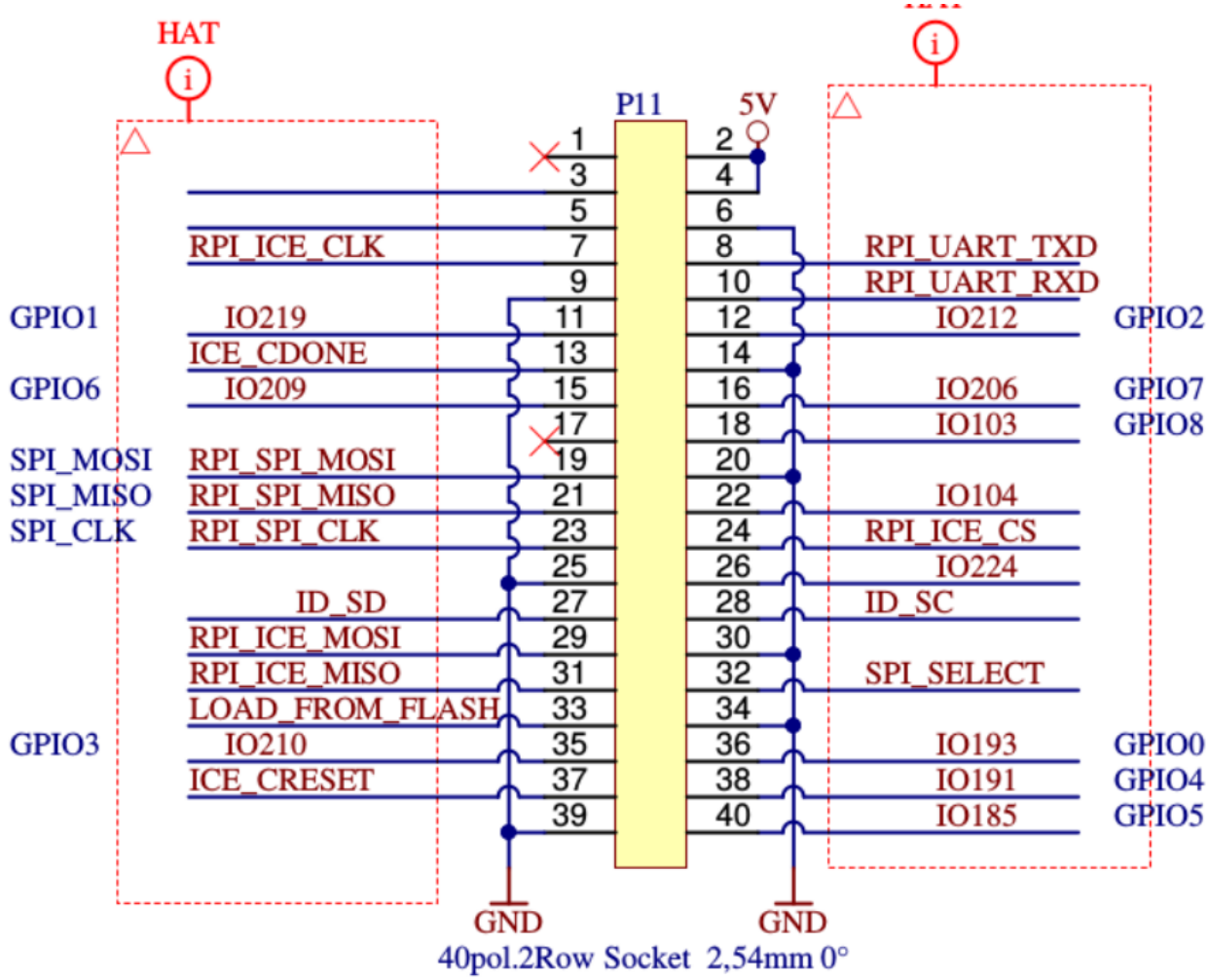

LED8
LED7
LED6
LED5
LED4
LED Digits 2.0
© 2012 XESS Corp.
R1
SW1
SW2
SW3
TMS

3rd led off
./arm-wbregs gpio 0x00040000
00001008 (GPIO)-> 00040000

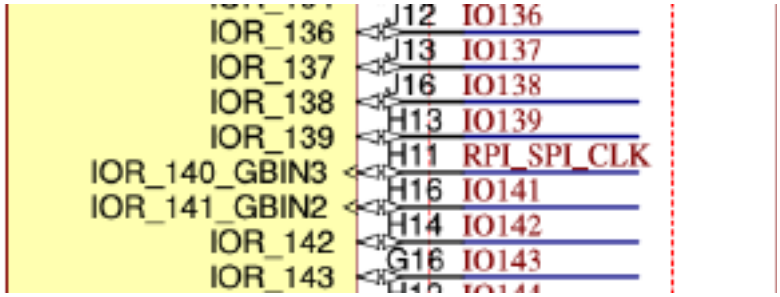
./arm-wbregs 0x2000 0x01
00002000 (RAM)-> 00000001
pi@pi3-5:~/catzip/sw/host $ ./arm-wbregs ram
00002000 (RAM) : [....] 00000001
pi@pi3-5:~/catzip/sw/host $ ./arm-wbregs 0x2000 0x02
00002000 (RAM)-> 00000002
pi@pi3-5:~/catzip/sw/host $ ./arm-wbregs ram
00002000 (RAM) : [....] 00000002

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

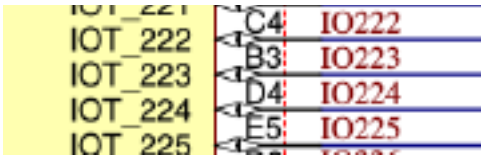
./arm-wbregs ufifo
00000804 (UFIFO) : [@?@.] 403f4000
ICOBOARD RPi



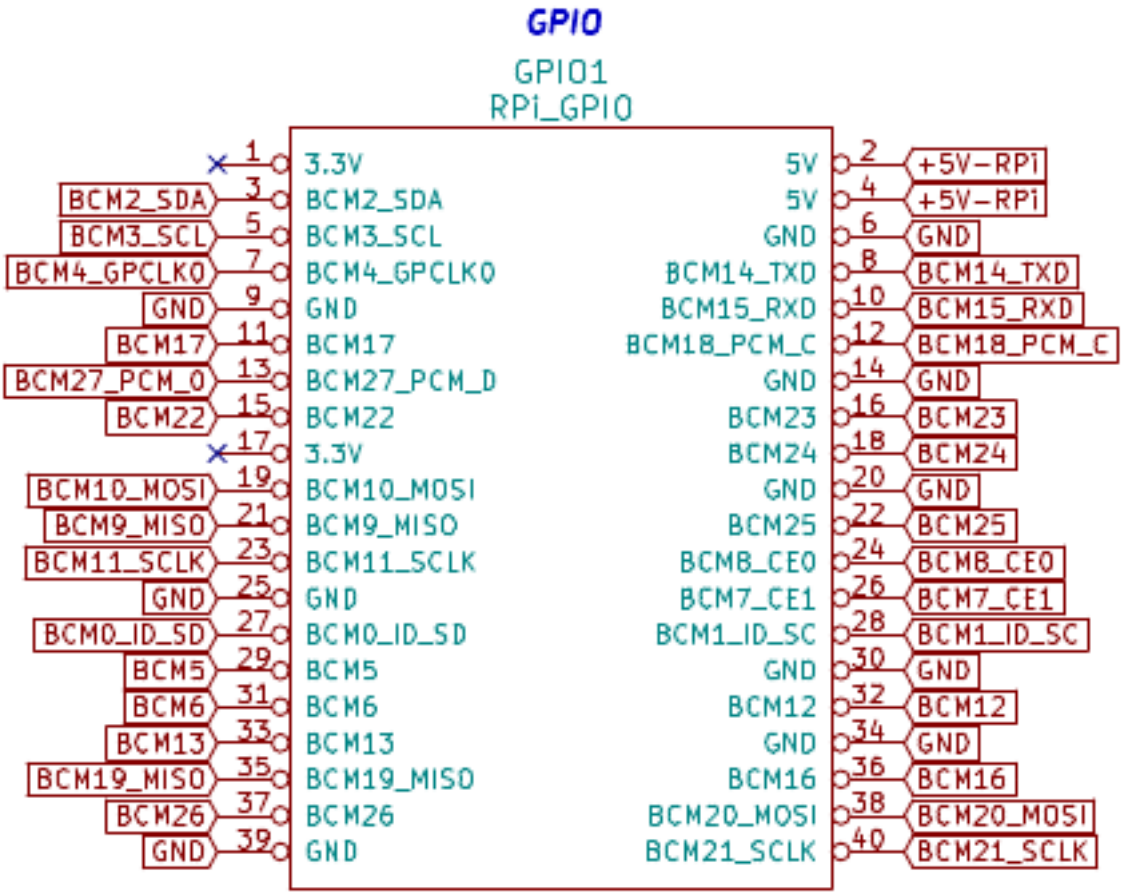
RPI_SPI_CLK H11 Pin 23 Pi icoboard



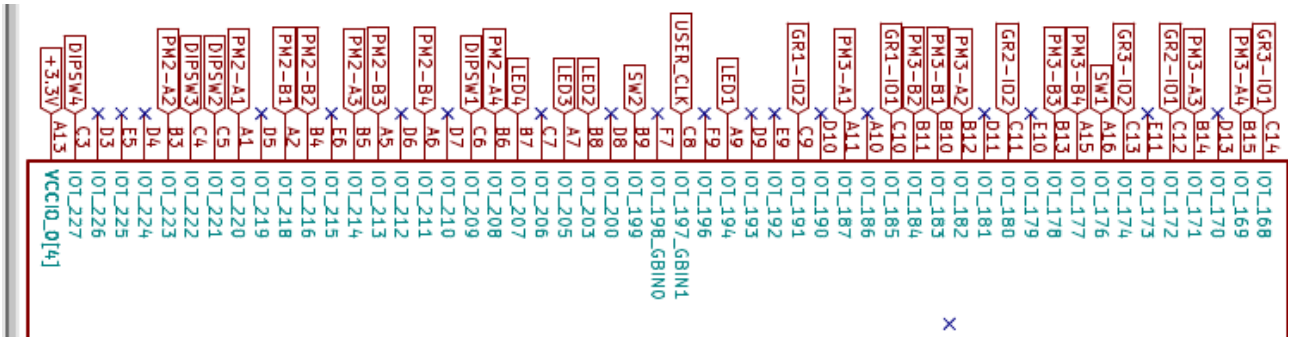
rpi_cs D4 IOT_224 Pin 26 Pi icoboard



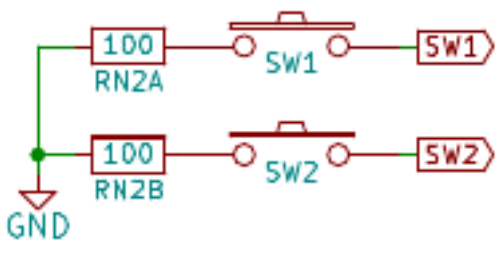
CATBOARD RPi



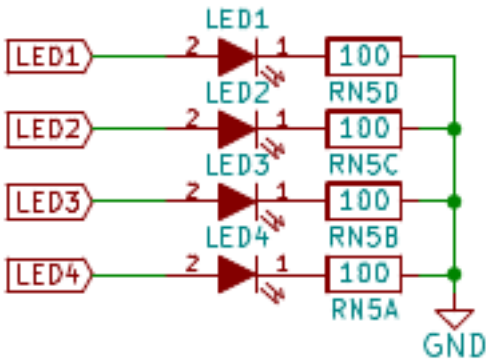
CATBOARD connection to FPGA pins PMOD 2 & PMOD 3 push button switches, dip switch, and leds.



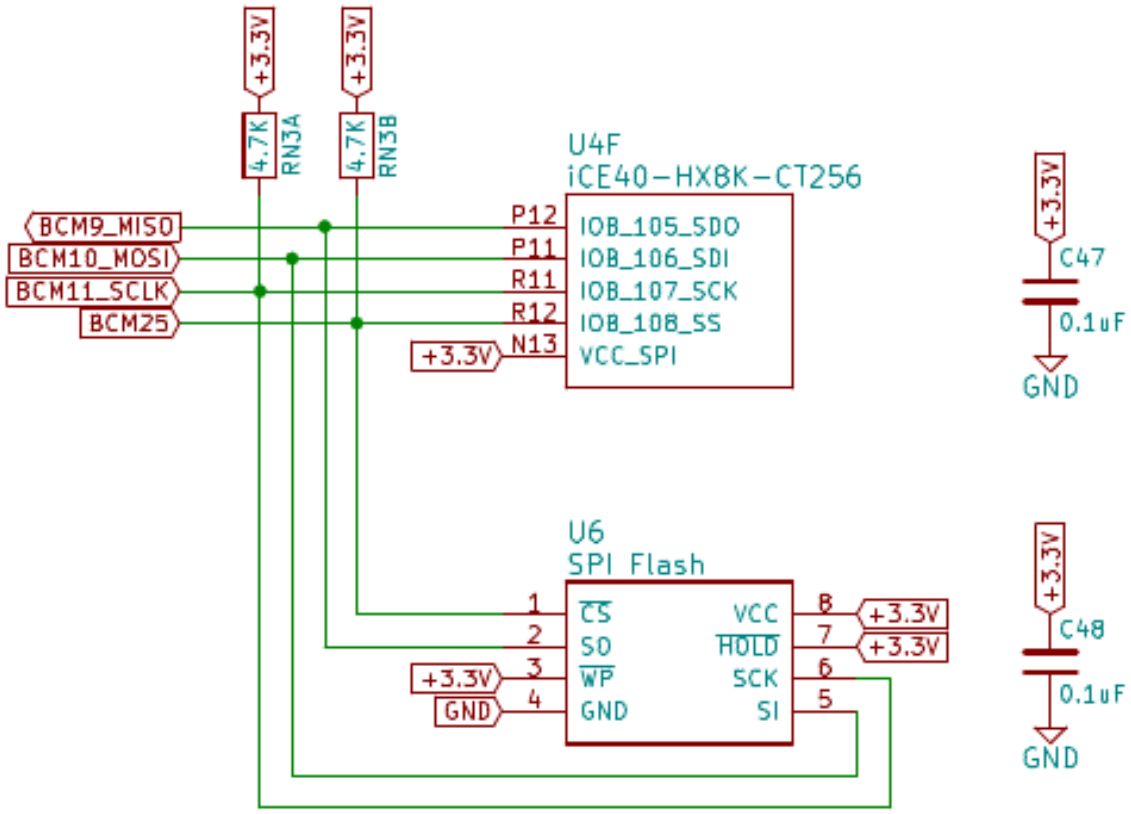
CATBOARD sw1 & sw2



CATBOARD leds

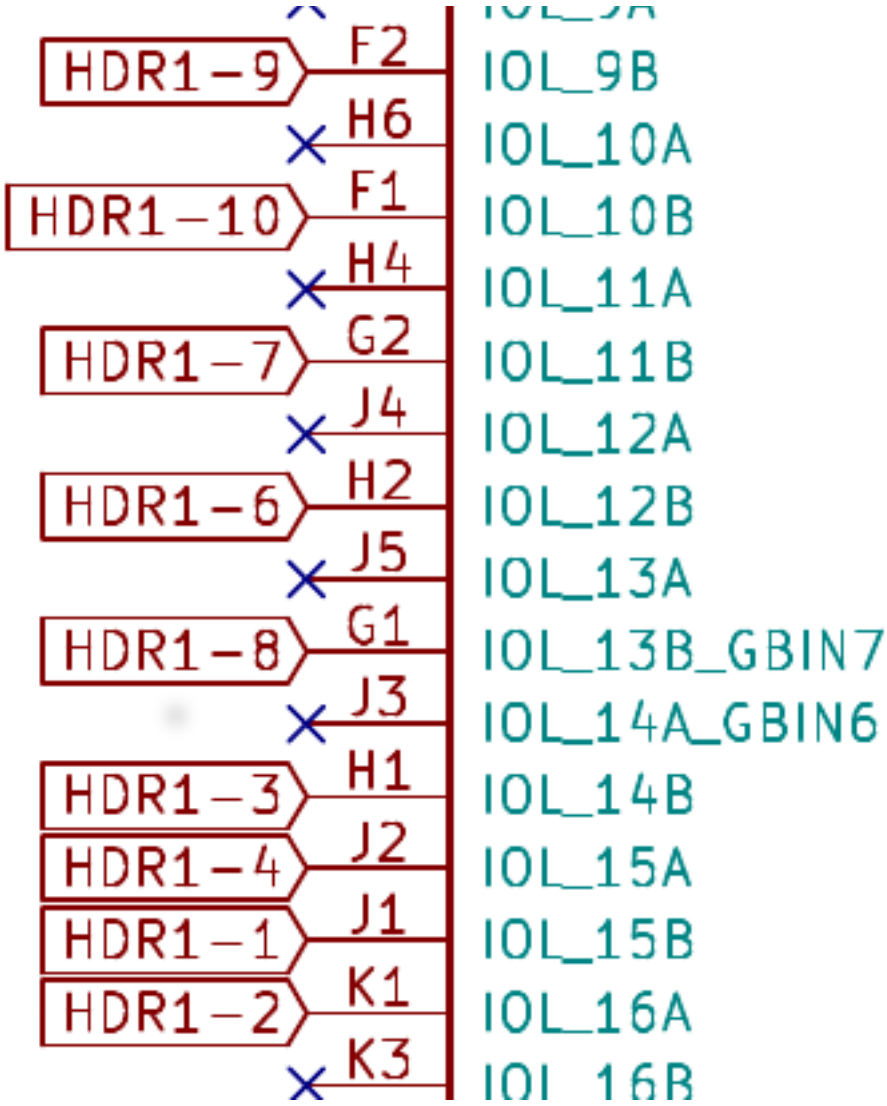


BCM11_SCLK Pin 23 CATBOARD



BCM7_CE1 Pin 26 CATBOARD

CATBOARD



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

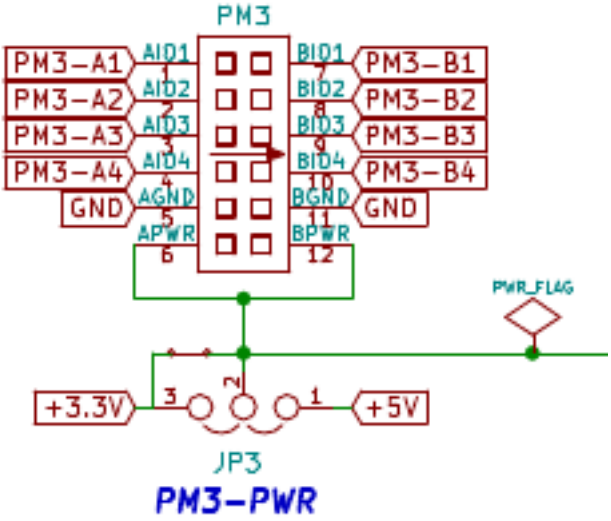
Pin C8 is my USER_CLK comes from a 100MHz osc. It is connected to IOT_197_GBIN1 on HX8K. When I try using it for as an input to PLL I get the fatal error: bad constraint on `i_clk': no PLL at pin C8.

Can only certain pins be used as inputs to PLL?

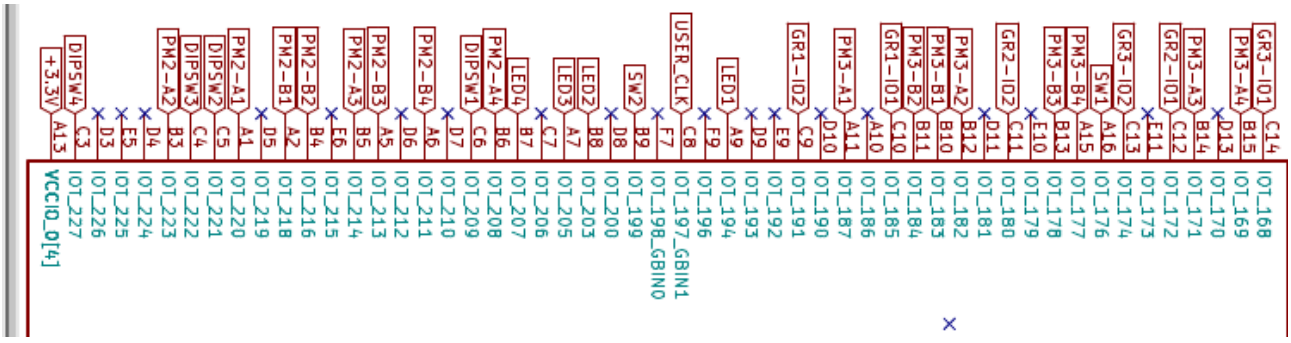
daveshah

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.

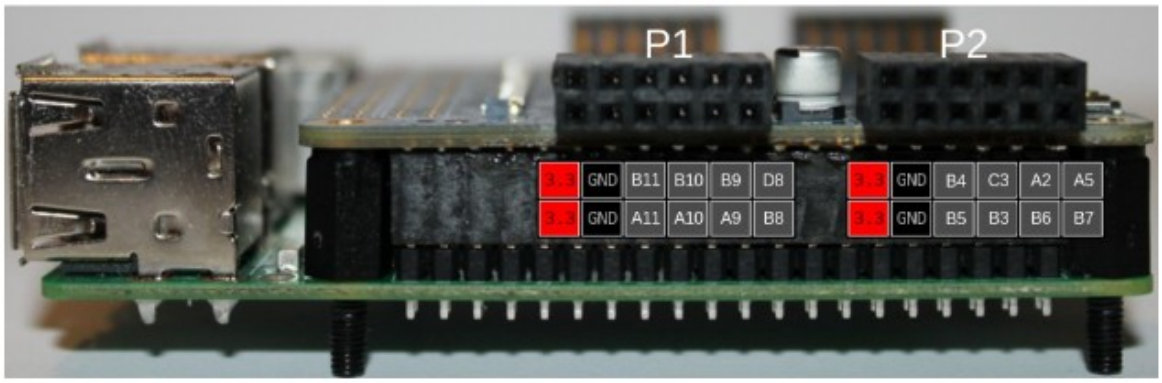


In top.v

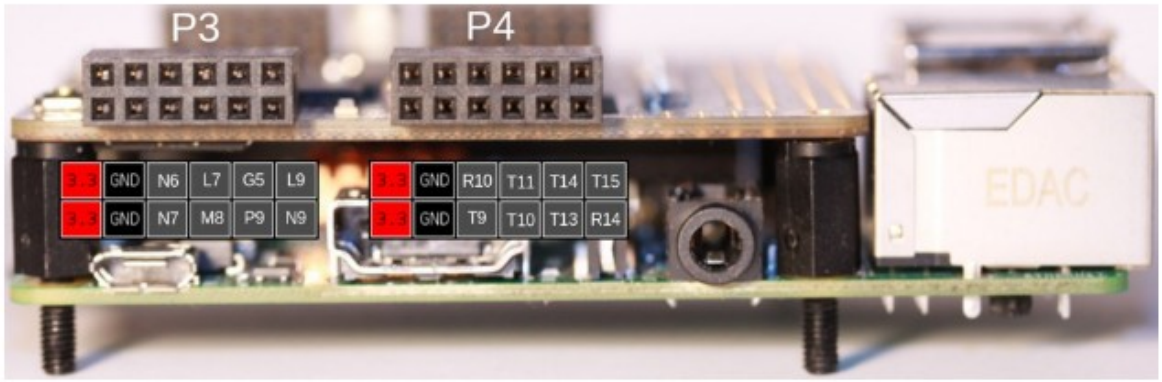
```
module top(clk_100mhz, pmod1_1, pmod1_2, pmod1_3, pmod1_4, pmod1_7, pmod1_8,
  pmod1_9, pmod1_10, pmod2_7, pmod2_8, pmod2_9, pmod2_10, rpi_sck, rpi_cs,
  rpi_mosi);

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”