t=0x81214a0 [0,0]
Contents: [Dobrica PavlinuÅ¡iÄ 's random unstructured stuff]

first steps

Here I will try to document correct order to read documentation to get setup for ULX3S:

https://github.com/emard/ulx3s-bin/blob/master/README.md

There is also useful things from chat at ULX3S Lobby

udev rule

ujprog

```
git clone https://github.com/f32c/tools f32c-tools
cd f32c-tools/ujprog/
dpavlin@x200:/mnt/nuc/FPGA/f32c-tools/ujprog$ rm ujprog
dpavlin@x200:/mnt/nuc/FPGA/f32c-tools/ujprog$ make -f Makefile.linux
cc -Wall -D__linux__ -std=gnu99 -static ujprog.c /usr/lib/x86_64-linux-gnu/libftdi.a /usr/lib/x86
dpavlin@x200:/mnt/nuc/FPGA/f32c-tools/ujprog$ sudo cp ujprog /usr/local/bin/
```

- create udev rule

passthru to access esp32

source at https://github.com/emard/ulx3s-passthru

```
dpavlin@x200:/mnt/nuc/FPGA/ulx3s-bin/fpga/passthru/passthru-v20-85f$ ujprog -j flash ulx3s_85f_pa
ULX2S / ULX3S JTAG programmer v 3.0.92 (built Nov 19 2019 10:55:50)
Using USB cable: ULX3S FPGA 12K v3.0.3
[Wed Nov 20 18:02:01 2019] ftdi_sio ttyUSB0: FTDI USB Serial Device converter now disconnected fr
[Wed Nov 20 18:02:01 2019] ftdi_sio 1-5.2:1.0: device disconnected
Programming: 100%
Completed in 24.36 seconds.
[Wed Nov 20 18:02:25 2019] usb 1-5.2: reset full-speed USB device number 56 using ehci-pci
[Wed Nov 20 18:02:26 2019] ftdi_sio 1-5.2:1.0: FTDI USB Serial Device converter detected
[Wed Nov 20 18:02:26 2019] usb 1-5.2: Detected FT-X
[Wed Nov 20 18:02:26 2019] usb 1-5.2: FTDI USB Serial Device converter now attached to ttyUSB0
```

update size of your FPGA

```
dpavlin@x200:/mnt/nuc/FPGA/ulx3s-bin$ usb-jtag/linux-amd64/ftx_prog --product "ULX3S FPGA 85K v3.
```

power cycle board to get new usb id, test that it's supported by ujprog

```
dpavlin@x200:/mnt/nuc/FPGA/ulx3s-bin$ ujprog -r
```

esptool and esp32 booting problems

You should be using ecptool from ulx3s-bin repository to quite @emard from https://gitter.im/ulx3s/Lobby#dark-theme

> OK then. If you have issues with ESP32 not booting with SD card but booting without SD card then then the fuse burn script from ulx3s-bin should be run. So far so good,

you erased its flash, try linux. If no issue then can try to flash micropython and my new ESP32 OTA programmer ecp5.py end uftpd.py

I have wisely taken some esptool.py which works and frozen it in ulx3s, versions change all the time and maybe you took something in the middle of development action :)

install micropython

https://github.com/emard/esp32ecp5/

```
dpavlin@nuc:/nuc/FPGA$ git clone https://github.com/emard/esp32ecp5/
dpavlin@nuc:/nuc/FPGA$ cd esp32ecp5/
dpavlin@x200:/mnt/nuc/FPGA/esp32ecp5$ wget https://micropython.org/resources/firmware/esp32-idf3-
```

It's important to erase flash or micropyhton will complain about corrupt fat filesystem like:

FAT filesystem appears to be corrupted. If you had important data there, you may want to make a flash snapshot to try to recover it. Otherwise, perform factory reprogramming of MicroPython firmware (completely erase flash, followed by firmware programming).

```
dpavlin@x200:/mnt/nuc/FPGA/esp32ecp5$ ../ulx3s-bin/esp32/serial-uploader/esptool.py --chip esp32
esptool.py v2.6-beta1
Serial port /dev/ttyUSB0
Connecting....
Chip is ESP32D0WDQ6 (revision 1)
Features: WiFi, BT, Dual Core, 240MHz, VRef calibration in efuse, Coding Scheme None
MAC: a4:cf:12:55:c5:60
Uploading stub...
Running stub...
Stub running...
Erasing flash (this may take a while)...
Chip erase completed successfully in 8.7s
Hard resetting via RTS pin...
```

```
dpavlin@x200:/mnt/nuc/FPGA/esp32ecp5$ ../ulx3s-bin/esp32/serial-uploader/esptool.py --chip esp32
esptool.py v2.6-beta1
Serial port /dev/ttyUSB0
Connecting....
Chip is ESP32D0WDQ6 (revision 1)
Features: WiFi, BT, Dual Core, 240MHz, VRef calibration in efuse, Coding Scheme None
MAC: a4:cf:12:55:c5:60
Uploading stub...
Running stub...
Stub running...
Changing baud rate to 460800
Changed.
Configuring flash size...
Auto-detected Flash size: 4MB
Compressed 1240192 bytes to 783187...
Wrote 1240192 bytes (783187 compressed) at 0x00001000 in 18.7 seconds (effective 529.3 kbit/s)...
```

```
Hash of data verified.

Leaving...
Hard resetting via RTS pin...

dpavlin@x200:/mnt/nuc/FPGA/esp32ecp5$ microcom -p /dev/ttyUSB0
connected to /dev/ttyUSB0
Escape character: Ctrl-\
Type the escape character to get to the prompt.

>>>
> help()
Welcome to MicroPython on the ESP32!

For generic online docs please visit http://docs.micropython.org/

For access to the hardware use the 'machine' module:

import machine
pin12 = machine.Pin(12, machine.Pin.OUT)
pin12.value(1)
pin13 = machine.Pin(13, machine.Pin.IN, machine.Pin.PULL_UP)
print(pin13.value())
i2c = machine.I2C(scl=machine.Pin(21), sda=machine.Pin(22))
i2c.scan()
i2c.writeto(addr, b'1234')
i2c.readfrom(addr, 4)

Basic WiFi configuration:

import network
sta_if = network.WLAN(network.STA_IF); sta_if.active(True)
sta_if.scan()                             # Scan for available access points
sta_if.connect("<AP_name>", "<password>") # Connect to an AP
sta_if.isconnected()                      # Check for successful connection

Control commands:
  CTRL-A        -- on a blank line, enter raw REPL mode
  CTRL-B        -- on a blank line, enter normal REPL mode
  CTRL-C        -- interrupt a running program
  CTRL-D        -- on a blank line, do a soft reset of the board
  CTRL-E        -- on a blank line, enter paste mode

For further help on a specific object, type help(obj)
For a list of available modules, type help('modules')
```

webrepl

```
dpavlin@klin:/klin/FPGA$ git clone https://github.com/hyperglitch/webrepl
```

You can send files from command-line:

```
dpavlin@x200:/mnt/nuc/FPGA/webrepl$ ./webrepl_cli.py -p ulx3s ../esp32ecp5/ecp5.py 192.168.3.130:
op:put, host:192.168.3.130, port:8266, passwd:ulx3s.
../esp32ecp5/ecp5.py -> /ecp5.py
Remote WebREPL version: (1, 11, 0)
Sent 22777 of 22777 bytes
```

```
dpavlin@x200:/mnt/nuc/FPGA/webrepl$ ./webrepl_cli.py -p ulx3s ../esp32ecp5/uftpd.py 192.168.3.130
op:put, host:192.168.3.130, port:8266, passwd:ulx3s.
../esp32ecp5/uftpd.py -> /uftpd.py
Remote WebREPL version: (1, 11, 0)
Sent 19482 of 19482 bytes
```

open source toolchain

Just use kost's binary builds: https://github.com/alpin3/ulx3s/releases

Or nightly builds: https://github.com/open-tool-forge/fpga-toolchain/releases

this is old and needs update

- https://github.com/SymbiFlow/prjtrellis

```
dpavlin@klin:/klin/FPGA$ git clone https://github.com/SymbiFlow/prjtrellis
dpavlin@klin:/klin/FPGA/prjtrellis$ ./download-latest-db.sh

dpavlin@klin:/klin/FPGA/prjtrellis$ cd libtrellis/
dpavlin@klin:/klin/FPGA/prjtrellis/libtrellis$ sudo apt-get install libpython3-dev libboost-pytho
dpavlin@klin:/klin/FPGA/prjtrellis/libtrellis$ cmake -DCMAKE_INSTALL_PREFIX=/usr/local .
dpavlin@klin:/klin/FPGA/prjtrellis/libtrellis$ make
sudo make install


dpavlin@klin:/klin/FPGA/nextpnr$ cmake -DARCH=ecp5 -DBUILD_GUI=OFF -DTRELLIS_ROOT=../prjtrellis/
make
make install
```

diamond

https://github.com/jandob/lattice-diamond-archlinux/blob/master/eth0DummyToggle

docker

https://gitter.im/ulx3s/Lobby?at=5dff4b08d2dadb38935c570a

https://github.com/dok3r/diamond/

```
docker run -it -v /host/fpga:/fpga -- local /host/fpga will end up in /fpga in docker

yes path will be fine
you will be missing make
so inside container you need to yum install make
and yum install libxslt
export ETHMAC=b0:5a:da:XX:XX:XX
set your MAC
docker run -it -v /media/internal/FPGA:/fpga -e LM_LICENSE_FILE=/fpga/license.dat --mac-address=$
```

```
run docker
yum install make libxslt
go tu project inside fpga folder and find makefile for diamond and then just make

then you share it with docker container with -v /yourHOSTfpgadir:/fpgadockerdir -e LM_LICENSE_FIL
for version you need to use like this dok3r/diamond:version
versions are here
https://hub.docker.com/r/dok3r/diamond/tags
docker run -it -v /media/internal/FPGA:/fpga -e LM_LICENSE_FILE=/fpga/license.dat --mac-address=$
like this
Not understanding -v /media/internal/FPGA
that is my local FPGA folder with samples and license.dat
it will mount on docker /fpga
and I see now that I need to share prjtrallis folder to docker so it can do ecppll
docker run -it -v /media/internal/FPGA:/fpga -v /local/prjtrellis/libtrellis:/mt/scratch/tmp/open
but for that we will need @kost
we probably need ecppll and tools already there and compiled with centos- maybe just binaries
```

NES

https://gitter.im/ulx3s/Lobby?at=5de033f49319bb5190a9c3b6

- https://github.com/ironsteel/nes_ecp5
- flash arbitrary data to flash: https://github.com/ironsteel/tools/commit/cb0c43b6681a52f1cc19b6b70ddd587a307da90c#diff-3b94
- list of idcode: https://github.com/SymbiFlow/prjtrellis/blob/master/devices.json

- ported to ulx3s: https://github.com/lawrie/nes_ecp5

oberon

https://gitter.im/ulx3s/Lobby?at=5e007d1e8897197969e3331c

So, I have now managed to build oberon with diamond 3.7.
What I have to do is:

1. Build it with diamond 3.11, which fails
2. mv clocks clocks_save
3. make clean
4. cp -r clocks_save clocks
5. run docker for diamond 3.7
6. edit synpbase/bin/config/platform_check to allow 5.* linux.
7. make
8. Use ujprog in host linux to upload generated bit file

Thanks @kost for adding for adding make and libxslt to the docker image. It would be useful if you could patch the platform_check to allow versions before 3.11 to run on 5.* linux.
I got a lot of errors in the diamond 3.7 docker build, but the .bit file was created.
I can now run oberon and can see windows on the screen, but I don't have a working mouse or keyboard. I would need Goran's USB board to get both mouse and keyboard.
@lawrie i fixed in latest v3.7 - just make sure that you're running latest:

docker pull dok3r/diamond:v3.7

woohoo! Cool
@kost I pulled the latest v3.7 about 10 minutes ago, but still had to edit platform_check.
synpbase/bin/config/platform_check has:

```
case $VERSION in
  4.* | 3.* | 2.4.* | 2.6.* )
```

It needs:

```
case $VERSION in
  5.* | 4.* | 3.* | 2.4.* | 2.6.* )
```

I did the docker pull to make sure I had the latest version.
I changed oberon makefile to generate clocks in already existing directory to get rid of annoying mkdir clocks
In my instructions above it is safer to do make ECPPLL=echo in docker, so that it does not try to use ecppll, but uses the saved clocks that were generated on host linux.

21f repack from 25f image

```
ecpunpack --input ulx3s_25.bit --textcfg ulx3s_12f.config --idcode 0x41111043
ecppack --input ulx3s_12f.config --bit ulxs3_12f.bit --idcode 0x21111043
```

compress bitstream

```
ecppack --compress
```

esp32ps2

https://github.com/emard/esp32ps2

saxonsoc

linux

Instructions at https://github.com/lawrie/saxonsoc-ulx3s-bin/tree/master/linux
work for me on 85f :-)

https://gitter.im/ulx3s/Lobby?at=5de8ba2f08d0c961b7f3a25f

```
git clone https://github.com/SpinalHDL/buildroot.git -b saxon buildroot
git clone https://github.com/SpinalHDL/linux.git -b vexriscv --depth 1 linux
cd buildroot
cp board/spinal/saxon_default/linux_nonet.config board/spinal/saxon_default/linux.config
# Add extra options to board/spinal/saxon_default/linux.config
make spinal_saxon_default_defconfig
make linux-rebuild all -j$(nproc)
output/host/bin/riscv32-linux-objcopy -O binary output/images/vmlinux output/images/Image
```

```
# Make sure Image is at least 116KB less than 4MB
```

85f version

https://gitter.im/ulx3s/Lobby?at=5dea74995ac7f22fb57055ae

https://github.com/lawrie/saxonsoc-ulx3s-bin/blob/master/linux/README.md

https://github.com/lawrie/saxonsoc-ulx3s-bin/tree/master/linux/u-boot

https://github.com/SpinalHDL/SaxonSoc/tree/dev/bsp/Ulx3sLinuxUboot

leds

https://gitter.im/ulx3s/Lobby?at=5dec101f46397c721ca4c814

```
#!/bin/sh
cd /sys/class/gpio
echo 488 > export
echo out > gpio488/direction
for i in 1 0 1 0 1 0
do
  sleep  0.1
  echo   $i > gpio488/value
done
```

slirp

https://gitter.im/ulx3s/Lobby?at=5df1467d0616d6515e20d197

modifications

https://gitter.im/ulx3s/Lobby?at=5dfced993e3f133894ca9b4b

u-boot config for 85f with 64M SDRAM

Modify bootcmd to include:

```
load mmc 0:1 0x80000000 /boot/uImage
load mmc 0:1 0x81EF0000 /boot/dtb
fdt add 0x81EF0000
fdt memory 0x80000000 0x04000000
bootm 0x80000000 - 0x81EF0000
```

ppp networking

- https://github.com/dok3r/ulx3s-saxonsoc/wiki/ulx3s-networking
- https://github.com/emard/esp32ppp

smp support

https://gitter.im/ulx3s/Lobby?at=5f4ea80bd4f0f55ebbf6ec33

https://github.com/SpinalHDL/SaxonSoc/tree/dev-0.1/bsp/radiona/ulx3s/smp

Instructions there need a bit of modification to run on blue 85f board with 64Mb of ram:

```
# Sourcing the build script
source SaxonSoc/bsp/radiona/ulx3s/smp/source.sh

# Clone opensbi, u-boot, linux, buildroot, openocd
saxon_clone

# Build the FPGA bitstream
saxon_standalone_compile bootloader CFLAGS_ARGS="-DSDRAM_TIMING=AS4C32M16SB_7TCN_ps"
SDRAM_SIZE=64 saxon_netlist
FPGA_SIZE=85 saxon_bitstream

# Build the firmware
saxon_opensbi
saxon_uboot
saxon_buildroot

# Build the programming tools
saxon_standalone_compile sdramInit CFLAGS_ARGS="-DSDRAM_TIMING=AS4C32M16SB_7TCN_ps"
saxon_openocd
```

Copy generated bitstream

```
dpavlin@klin:/klin/FPGA/saxonsoc$ cp SaxonSoc/hardware/synthesis/radiona/ulx3s/smp/bin/toplevel.b
dpavlin@klin:/klin/FPGA/saxonsoc$ gzip -9 saxon.bit
```

Transfer it using ftp

```
ftp> put saxon.bit.gz
local: saxon.bit.gz remote: saxon.bit.gz
200 OK
150 Opened data connection.
226 Done.
359484 bytes sent in 10.27 secs (34.1994 kB/s)
ftp> site saxon.bit.gz
```

u-boot will fail to boot if you have rootfs on second partition

```
SDRAM init
OpenSBI copy
U-Boot copy
```

```
OpenSBI boot

OpenSBI v0.6-8-gd7b62b8
   ____                    _____ ____ _____
  / __ \                  / ____|  _ \_   _|
 | |  | |_ __   ___ _ __ | (___ | |_) || |
 | |  | | '_ \ / _ \ '_ \ \___ \|  _ < | |
 | |__| | |_) |  __/ | | |____) | |_) || |_
  \____/| .__/ \___|_| |_|_____/|____/_____|
        | |
        |_|

Platform Name          : VexRiscv SMP simulation
Platform HART Features : RV32AIMS
Platform Max HARTs     : 4
Current Hart           : 0
Firmware Base          : 0x80f80000
Firmware Size          : 84 KB
Runtime SBI Version    : 0.2

MIDELEG : 0x00000222
MEDELEG : 0x0000b101


U-Boot 2020.07-08304-gd361dd3997 (Sep 05 2020 - 09:45:52 +0200)

DRAM:  32 MiB
MMC:   spi@10020000:mmc@1: 0
Loading Environment from FAT... Unable to use mmc 0:1... In:    serial@10010000
Out:   serial@10010000
Err:   serial@10010000
Net:   No ethernet found.
Hit any key to stop autoboot:  0
Wrong Image Format for bootm command
ERROR: can't get kernel image!
=>
```

https://github.com/dok3r/ulx3s-saxonsoc/wiki/SaxonSoc-on-ULX3s

```
setenv bootcmd "load mmc 0:1 0x80000000 /boot/uImage;load mmc 0:1 0x80FF0000 /boot/dtb;fdt add 0x
setenv bootargs "rootwait console=hvc0 root=/dev/mmcblk0p2 init=/sbin/init mmc_core.use_spi_crc=0
saveenv
```

> Lawrie Griffiths @lawrie Sep 01 21:59

The new SaxonSoc is now working on a 12F for me. Here are the instructions to build from source - https://github.com/SpinalHDL/SaxonSoc/tree/dev-0.1/bsp/radiona/ulx3s/smp
The images and bitstream are here - https://github.com/lawrie/saxonsoc-ulx3s-bin/tree/master/Smp
There is no sdcard image at the moment, but all the files are there for you to build your own.

ov7670 pmod

https://github.com/goran-mahovlic/fpga-odysseus/tree/master/projects/OV7670-HDMI

pmod pin mapping:

https://github.com/goran-mahovlic/fpga-odysseus/blob/master/projects/OV7670-HDMI/ulx3s.lpf#L335

SCCB Pullup Resistors

from https://github.com/westonb/OV7670-Verilog

The SCCB interface for the camera requires pull up resistors. You need to solder 4.7K resistors from the SIOD and SIOC pins on the camera to the 3.3V supply. You can do this yourself or have the staff in the EDS help you.

ov7670_rgb_yuv_320x240_colorfilter

https://github.com/JdeRobot/FPGA-robotics/tree/master/Projects/ComputerVision

nmigen

https://github.com/lawrie/ulx3s-nmigen-examples/blob/master/image/camtest.py

csi

https://twitter.com/mad_archer_/status/1231249513509261313

https://github.com/libv/fosdem-video-linux

litex

(just links, need to test it)

- https://gojimmypi.blogspot.com/2020/03/litex-soft-cpu-on-ulx3s-reloading.html
- https://github.com/timvideos/litex-buildenv/wiki/LiteX-for-Hardware-Engineers
- https://github.com/enjoy-digital/litex

spiram

https://gitter.im/ulx3s/Lobby?at=5ef22d4c54d7862dc4a42395

@Speccery thereis also commandline "spiram.py" for some low-level inspection, so to reset TI this works for me

```
spiram.poke(0x100008,bytearray(0xFC))
spiram.poke(0x100008,bytearray(0xFF))
```

and to read bytes

```
spiram.peek(0,16)
```

bytearray(b'\x83\xe0\x00$\x83\xc0\t\x00\x83\xc0\n\x920\xaa\x04`')

led

```
ftx_prog --cbus 3 DRIVE_0 # green OFF

ftx_prog --cbus 3 SLEEP # green ON if enumerated
```

This is active after power cycle

micropython blue led

```
>>> from machine import Pin
>>> led=Pin(5,Pin.OUT)
>>> led.on() # upali plavu
>>> led.off() # ugasi plavu
```

micropython

```
from upysh import *
```

TODO

try various projects for ulx3s

- https://gitlab.com/pnru/cortex

c64

part of https://github.com/lawrie/ulx3s_retro

https://github.com/emard/ulx3s_c64

```
dpavlin@klin:/klin/FPGA/ulx3s_c64/proj$ time make FPGA_SIZE=25
```

tfmicro on LiteX/VexRiscv

https://github.com/dlobato/tfmicro-on-litex-vexriscv

ML CNN accelerator

https://github.com/BracketMaster/maeri

kianRiscV

https://gitAhub.com/splinedrive/kianRiscV/tree/master/linux_socs/kianv_mc_rv32ima_sv32/demo

openFPGALoader -f -o $((1024*1024)) --board=ulx3s bootloader.bin