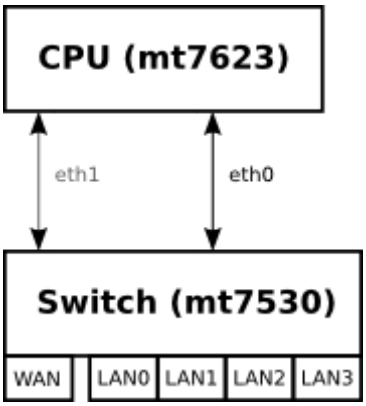
Network-Configuration

ip-command needs package iproute2

Configuration on this page is based on debian stretch, should work the same way with Ubuntu needs Kernel 4.14 or above (DSA-driver for Port-separation)

in Kernel 4.14 eth0 is the connection between CPU and the Switch-Circuit (mt7530), on which the Ports wan and lan0-4 are connected. this connection have to be set to „up“ first.



bringing up then cpu-port(s)

```
ip link set eth0 up
ip link set eth1 up
```

or via /etc/network/interfaces

```
auto eth0
iface eth0 inet manual
  pre-up ip link set $IFACE up
  post-down ip link set $IFACE down

auto eth1
iface eth1 inet manual
  pre-up ip link set $IFACE up
  post-down ip link set $IFACE down
```

the mapping of ports to gmac is defined in dts-file and can be shown with „ip a“

With 4.14 >.52 on my repo gmac #2 (eth1) is added and wan is connected to this.

by default each lan-port is separated and needs an own ip-configuration in different subnets

most users like to use all lan-ports in 1 network-segment, so these can be bridged together to make only 1 ip-configuration for „LAN“

my 6.3-rc brings some patches for re-introduce second gmac. by default port 6 (trgmii) is used and can be changed in userspace:

```
ip link set wan type dsa master eth1
```

* requires iproute2 v6.1+

add backports in sources.list:

```
deb http://deb.debian.org/debian bullseye-backports main contrib non-free
```

install

```
apt update
apt -t bullseye-backports install iproute2
```

MAC-Address

The MAC-address can only be set for the GMAC (connection between Switch and CPU). In Kernel 4.14 only 1 GMAC is detected (eth0). There are 2 GMACs in Hardware.

UDEV

here

```
$ cat /etc/udev/rules.d/00-static-mac-address.rules
ACTION=="add", SUBSYSTEM=="net", KERNELS=="1b100000.ethernet",
RUN+="/sbin/ip link set dev %k address ae:fc:de:ad:be:ef"
```

interfaces-file

/etc/network/interfaces

```
iface lan0 inet static
  address 192.168.0.10
  netmask 255.255.255.0
  gateway 192.168.0.5
#  pre-up ip link set $IFACE up
  pre-up ip link set $IFACE address 02:01:02:03:04:08 up
```

using systemd

/etc/systemd/network/10-wan.link

```
[Match]
OriginalName=wan

[Link]
```

```
MACAddress=XX:XX:XX:XX:XX:XX
```

http://forum.banana-pi.org/t/set-mac-address-on-boot/7224/7

device-tree

```
local-mac-address = [00 0a 35 00 00 01];
mac-address = [00 0a 35 00 00 01];
```

http://forum.banana-pi.org/t/set-mac-address-on-boot/7224/4

this can also be used in devicetree-overlays

set via uboot

if devicetree (with mac-address property) is loaded separately (fdt), an alias for ethernet-node is defined and ethaddr-variable is set in uboot this is used in linux

http://forum.banana-pi.org/t/set-mac-address-on-boot/7224/6

interface-name

Ubuntu 18.4+ (and debiaan 10+) using new interface names. Wifi devices are no more named like wlanX more like wlpXsY

to avoid this, „net.ifnames=0“ can be added to Kernel-Cmdline (uEnv.txt for uboot) and/or rename via udev

/etc/udev/rules.d/70-persistent-net.rules

```
SUBSYSTEM=="net", ACTION=="add", ATTR{address}=="f8:62:aa:50:15:c8",
NAME="wlan1"
```

find attributes like this

```
udevadm info --attribute-walk /sys/class/net/<interface-name>
```

to apply the rule(s) (but does not rename back):

```
udevadm control --reload-rules && udevadm trigger
```

after reboot all works

if driver is compiled as module it can be reloaded (after activating the udev-rules)

```
modprobe -r mt76x2e
```

```
modprobe mt76x2e
```

IP

iproute2 cheatsheet

permanent

/etc/network/interfaces:

```
#first set the upstream-Port (NIC between CPU and MT7530-Switch) up
auto eth0
iface eth0 inet manual
  pre-up ip link set $IFACE up
  post-down ip link set $IFACE down

auto eth1
iface eth1 inet manual
  pre-up ip link set $IFACE up
  post-down ip link set $IFACE down

#then configure the lan-ports
auto lan0
iface lan0 inet static
  hwaddress ether 08:00:00:00:00:00 # if you want to set MAC manually
  address 192.168.0.10
  netmask 255.255.255.0
  gateway 192.168.0.5
  pre-up ip link set $IFACE up
  post-down ip link set $IFACE down
```

systemd

/etc/systemd/network/eth0.network:

```
[Match]
Name=eth0

[Network]
DHCP=no
LinkLocalAddressing=no
ConfigureWithoutCarrier=true
```

/etc/systemd/network/wan.network

```
[Match]
```

```
Name=wan

[Network]
BindCarrier=eth0
#ConfigureWithoutCarrier=true

#IPForward=yes
#IPMasquerade=yes
Address=192.168.0.18/24
DNS=192.168.0.10
Gateway=192.168.0.10
```

Is ConfigureWithoutCarrier set on wan-port, the default-route will not be set,because Address is invalid (Network is down at time of configuration). This should only be set if no default-route is needed.

systemd

second Ethernet lane (gmac)

Needs kernel-patch for eth1 + aux interfaces (currently only in 5.15)

```
- create a bridge for use for wan

    /etc/systemd/network/11-wanbr.netdev
    [NetDev]
    Name=wanbr
    Kind=bridge

    [Bridge]
    DefaultPVID=0 # should be different to other vlan-aware bridges (like
lanbr)
    VLANFiltering=1

- map aux and wan to vlan-aware bridge
- traffic will be tagged inside with vlan-id 99

    /etc/systemd/network/12-wanbr-bind.network
    [Match]
    Name=wan aux

    [Link]
    RequiredForOnline=no

    [Network]
    BindCarrier=eth0
    Bridge=wanbr

    [BridgeVLAN]
    VLAN=99
```

```
    PVID=99
    EgressUntagged=99

- put wanbr up by default

    /etc/systemd/network/13-wanbr.network
    [Match]
    Name=wanbr

    [Network]
    BindCarrier=eth0
    ConfigureWithoutCarrier=true

- configure eth1 as wan

    /etc/systemd/network/15-wan.network
    [Match]
    Name=eth1

    [Network]
    BindCarrier=eth0

    Address=192.168.0.18/24
    Gateway=192.168.0.10
    DNS=192.168.0.10

    IPForward=yes
```

temporary way

```
brdev=gmacbr
ip link add name $brdev type bridge
ip link set dev $brdev up
ip link set dev wan master $brdev
ip link set dev aux master $brdev
ip link set $brdev type bridge vlan_filtering 1
ip a del 192.168.0.18/24 dev wan #remove ip from original interface
ip a add 192.168.0.18/24 dev eth1
ip link set eth1 up
bridge vlan add vid 100 dev wan pvid untagged
bridge vlan add vid 100 dev aux pvid untagged
bridge vlan del vid 1 dev aux
bridge vlan del vid 1 dev wan
ip link set aux up
ip link set wan up
```

how to check

```
bridge vlan
```

```
ip a
ping 192.168.0.21
ethtool -S eth1 #traffic on this gmac?
iperf3 -c 192.168.0.21 #throughput in one direction
iperf3 -c 192.168.0.21 -R #throughput in the other direction
```

temporary

```
ifconfig lan0 192.168.0.10 netmask 255.255.255.0 broadcast 192.168.0.255
```

```
ip addr add 192.168.0.10/24 broadcast 192.168.0.255 dev lan0
```

make sure only 1 port is in the specific subnet.

```
ip a
#or
ip addr show lan0
```

DHCP

Client

/etc/network/interfaces:

```
auto lan3
allow-hotplug lan3
iface lan3 inet dhcp
```

Renew ip via

```
sudo dhclient -v -r lan3
```

Server

/etc/dnsmasq.conf (activate line by removing # on begin of line)

```
conf-dir=/etc/dnsmasq.d
```

/etc/dnsmasq.d/interfaces.conf

```
interface=wlan1
interface=ap0

# DHCP-Server not active for Interface
no-dhcp-interface=eth0
no-dhcp-interface=eth1
```

```
#dhcp-authoritative (interface+range+leasetime, default-gateway-ip as option 3)
dhcp-range=ap0,192.168.10.100,192.168.10.150,255.255.255.0,48h
dhcp-option=ap0,3,192.168.10.1
dhcp-range=wlan1,192.168.11.100,192.168.11.150,255.255.255.0,48h
dhcp-option=wlan1,3,192.168.11.1
```

/etc/dnsmasq.d/interfaces.conf

```
service dnsmasq start
```

more info here: dnsmasq

IPv6

disabling

https://www.thomas-krenn.com/de/wiki/IPv6_deaktivieren

temporary

```
echo 1 > /proc/sys/net/ipv6/conf/all/disable_ipv6
```

to make it permanent create file /etc/sysctl.d/01-disable-ipv6.conf with content:

```
net.ipv6.conf.all.disable_ipv6 = 1
```

testing:

```
ip addr show | grep inet6
```

NAT/Routing

NAT

to enable Network Adress Translation (net with private IPs behind one public IP)

```
ipt=/sbin/iptables
if_wan=wan
${ipt} -t nat -A POSTROUTING -o ${if_wan} -j MASQUERADE
```

HW-Nat

HW-Nat is currently only available in LEDE (Kernel 4.9)

i have merged the Lede-Patches to my 4.9-main and ported to 4.14 (4.14-hnat), see HW-NAT

Routing

enable routing for IPv4

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

alternative:

```
nano /etc/sysctl.conf
#activate net.ipv4.ip_forward=1 and net.ipv6.conf.all.forwarding=1 by
removing # at beginning of line
sysctl -p /etc/sysctl.conf
```

manipulating default route:

```
ip route del default
ip route add default via 192.168.50.2
```

show routing table

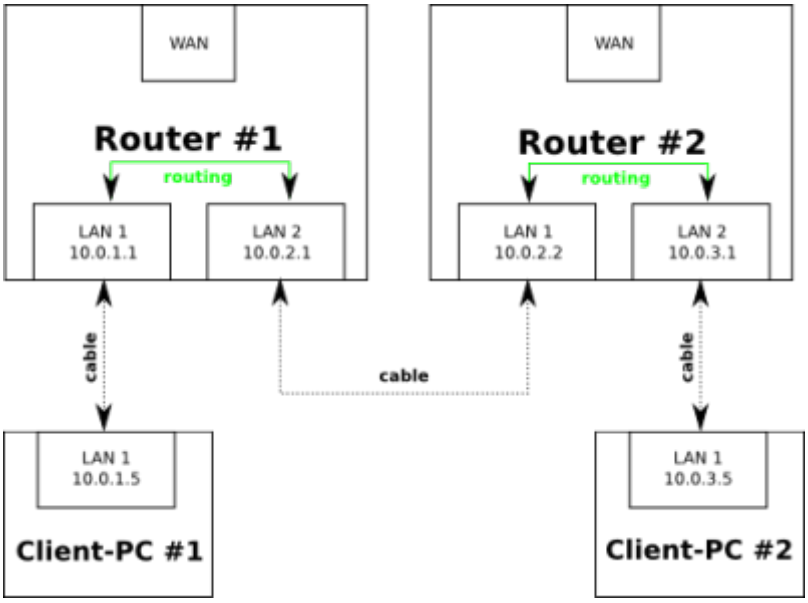
```
ip route show
```

remember you need DNS-resolving (/etc/resolv.conf) for translating domains to ip-addresses

adding static routes to other networks

Pakets are sent to the default-gateway, if the net is not known (directly connected or route available). In normal home-networks there is only 1 router and in this the default-gateway is the Internet-interface and on client-PCs the default-gateway is this router.

static routes are needed, if a net is not directly connected to a router and not accessable via its default-gateway



- in router #1 a static route must be added for net 10.0.3.0/24 with next-hop 10.0.2.2 (send pakets over lan#2)
  - `ip route add 10.0.3.0/24 via 10.0.2.2`
- in router #2 a static route must be added for net 10.0.1.0/24 with next-hop 10.0.2.1 (send pakets over lan#1)
  - `ip route add 10.0.1.0/24 via 10.0.2.1`

example for net 192.168.50.x behind router with ip 192.168.0.10

```
ip route add 192.168.50.0/24 via 192.168.0.10
```

DNS

/etc/resolv.conf

contains ip-adress to nameserver, e.g.

```
nameserver 192.168.0.10
```

on newer debian/ubuntu this file is a symlink to

/run/systemd/resolve/stub-resolv.conf

Netbridge

if 2 or more lan-ports should use same network-segment (configure only 1 IP-address for „LAN“), you can bridge ports together.

```
apt-get install bridge-utils
```

/etc/network/interfaces:

```
auto lan1
iface lan1 inet manual
auto lan2
iface lan2 inet manual

auto br0
iface br0 inet static
    address 192.168.40.1
    netmask 255.255.255.0
    bridge_ports lan1 lan2
    bridge_fd 5
    bridge_stp no
```

temporary bridge

```
brctl addbr br0
brctl addif br0 lan1
brctl addif br0 lan2
ip addr add 192.168.40.1/24 dev br0
ip link set br0 up

brctl show br0
```

Note:

brctl is deprecated please use ip/bridge

```
ip link add name br0 type bridge
ip link set dev br0 up
ip link set dev lan0 master br0
ip link set dev lan1 master br0

#remove interface from bridge
ip link set dev lan0 nomaster

#remove bridge
ip link del br0
```

https://unix.stackexchange.com/a/255489

VLAN

vlan on dsa-ports need

additional Patch

/etc/network/interfaces:

```
auto lan3.60
iface lan3.60 inet static
  address 192.168.60.10
  netmask 255.255.255.0
```

temporary

```
#!/bin/bash
netif=wan
ip link set $netif up
ip link add link $netif name vlan110 type vlan id 110
ip link set vlan110 up
ip addr add 192.168.110.1/24 dev vlan110
#tcpdump -i $netif -nn -e vlan &
```

vlan aware bridge

With 4.16 vlan aware bridge support was added.


vlan_filtering needs to be enabled before dsa-ports are added to the bridge, else all traffic (untagged too) is blocked after this setting.

```
#!/bin/bash
BRIDGE=lanbr0
netif=lan0
vid=500
vlanip=192.168.110.5/24

#ip link add name ${BRIDGE} type bridge
ip link add name ${BRIDGE} type bridge vlan_filtering 1 vlan_default_pvid 1
ip link set ${BRIDGE} up
ip link set $netif master ${BRIDGE}
ip link set $netif up
bridge vlan add vid $vid dev ${BRIDGE} self
bridge vlan add vid $vid dev $netif

#extract vlan from bridge to own netdev
ip link add link ${BRIDGE} name vlan$vid type vlan id $vid
ip a a $vlanip dev vlan$vid
ip link set vlan$vid up
```

testing

```
sudo tcpdump -ei lan1 arp or icmp
```

-e shows link-layer information like vlan

```
sudo tcpdump -XXi lan1 arp or icmp
```

shows arp and icmp-packets as hex-dump on the interface

offset 0x0c should show 8100 followed by hex-value of vlan-number (here vlan 500 = 0x01f4)

```
12:16:26.491644 IP 192.168.50.11 > frank-G5: ICMP echo reply, id 4294, seq
5, length 64
0x0000:  3c18 a003 c3a4 c63a 3897 5920 8100 01f4  <......:8.Y.....
```

Firewall (iptables)

iptables NFTables

Monitoring

```
sudo tcpdump -i eth0 port not 22 > tcpdump.log
sudo tcpdump -XXi lan1 arp or icmp
```

PPPoE

example creates a pppoe-connection over vlan 140 like my ISP

Server

```
sudo apt install pppoe

sudo ip link add link enx00e04c680683 name wan.140 type vlan id 140

/etc/ppp/pap-secrets
"bpi-r2"    *   "1234578"   *

/etc/ppp/pppoe-server-options
# PPP options for the PPPoE server
# LIC: GPL
debug
#plugin /etc/ppp/plugins/rp-pppoe.so
```

```
require-pap
mtu 1492
mru 1492
ktune
proxyarp
lcp-echo-interval 10
lcp-echo-failure 2
nobsdcomp
noccp
novj
noipx
```

```
iptables -t nat -A POSTROUTING -s 192.168.1.0/24 -o enp3s0 -j MASQUERADE
echo 1 > /proc/sys/net/ipv4/ip_forward

pppoe-server -I wan.140 -L 192.168.1.1 -R 192.168.1.100 -F &
```

Client

```
apt install pppoeconf
ip link add link wan name wan.140 type vlan id 140
ip link set wan.140 up
pppoeconf wan.140
```

/etc/ppp/peers/dsl-provider (should be created by pppoeconf)

```
# Minimalistic default options file for DSL/PPPoE connections

noipdefault
defaultroute
replacedefaultroute
hide-password
#lcp-echo-interval 30
#lcp-echo-failure 4
noauth
persist
#mtu 1492
#persist
#maxfail 0
#holdoff 20
plugin rp-pppoe.so wan.140
user "bpi-r2"
usepeerdns
```

```
pon dsl-provider
```

i had to delete old route to my local lan-subnet which is used to connect my dns (then the default route through ppp is used).

```
root@bpi-r2:~# ping 192.168.0.10
PING 192.168.0.10 (192.168.0.10) 56(84) bytes of data.
^C
--- 192.168.0.10 ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 12ms
root@bpi-r2:~# cat /etc/resolv.conf
nameserver 192.168.0.10
root@bpi-r2:~# ip route
default dev ppp0 scope link
192.168.0.0/24 dev wan proto kernel scope link src 192.168.0.12
192.168.1.1 dev ppp0 proto kernel scope link src 192.168.1.100
192.168.90.0/24 dev lan3 proto kernel scope
root@bpi-r2:~# ip route del 192.168.0.0/24
root@bpi-r2:~# ping 192.168.0.10
PING 192.168.0.10 (192.168.0.10) 56(84) bytes of data.
64 bytes from 192.168.0.10: icmp_seq=1 ttl=63 time=1.56 ms
64 bytes from 192.168.0.10: icmp_seq=2 ttl=63 time=2.22 ms
^C
--- 192.168.0.10 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 3ms
rtt min/avg/max/mdev = 1.563/1.893/2.224/0.333 ms
root@bpi-r2:~# ping www.google.de
PING www.google.de (142.250.185.99) 56(84) bytes of data.
64 bytes from fra16s49-in-f3.1e100.net (142.250.185.99): icmp_seq=1 ttl=120
time=7.57 ms
64 bytes from fra16s49-in-f3.1e100.net (142.250.185.99): icmp_seq=2 ttl=120
time=7.48 ms
^C
--- www.google.de ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 2ms
rtt min/avg/max/mdev = 7.483/7.526/7.570/0.097 ms
root@bpi-r2:~# ping www.google.de
PING www.google.de (142.250.185.99) 56(84) bytes of data.
64 bytes from fra16s49-in-f3.1e100.net (142.250.185.99): icmp_seq=1 ttl=120
time=7.57 ms
64 bytes from fra16s49-in-f3.1e100.net (142.250.185.99): icmp_seq=2 ttl=120
time=7.48 ms
^C
--- www.google.de ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 2ms
rtt min/avg/max/mdev = 7.483/7.526/7.570/0.097 ms

root@bpi-r2:~# traceroute 8.8.8.8
traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets
 1  192.168.1.1 (192.168.1.1)  0.594 ms  0.389 ms  0.278 ms
 2  bpi-r2-emmc (192.168.0.10)  1.408 ms  1.327 ms  1.109 ms
 3  me60.stadtnetz-bamberg.de (217.61.144.1)  4.962 ms  4.873 ms  4.789 ms
```