LinTech

HBM11

Board Support Package

Manual v1.0.0

Date: 2024-10-07
Revision: 3

Contents

1 Changes 1

2 Introduction 2

3 Building the BSP 2

3.1 System requirements 2

3.1.1 Build requirements 2

3.1.2 Flash requirements 2

3.1.3 Serial console 3

3.2 Build process 4

4 Running the BSP 5

4.1 Audio 5

4.1.1 Mute/Unmute 5

4.1.2 Speaker test 5

4.1.3 Record 5

4.1.4 Volume Control 6

4.2 S/PDIF 6

4.3 Network 6

4.3.1 Ethernet 6

4.3.2 WLAN 7

4.3.3 Bluetooth Audio (A2DP) 8

4.3.4 Antenna 9

4.4 Multimedia Keys 9

4.5 USB Host 10

4.6 LEDs 10

LinTech GmbH Kommunikationstechnologien | Friedrich-Engels-Straße 35, D-13156 Berlin
Phone: +49 (030) 55 22 6273 | Fax: +49 (030) 55 22 62 84 | E-Mail: LinTech@LinTech.de | http://www.LinTech.de

1 Changes

Date	Version	Changes	Author
2018-07-12	1.0.0	1. Initial version	Jörg Krause
2024-10-08	1.0.0	adress line	Susanne Meinel

2 Introduction

The Board Support Package is based on Buildroot, an easy-to-use tool to generate embedded Linux systems, and is based on the latest long term support release: 2018.02. Please read the Buildroot user manual[1] for a better understanding of Buildroot, the BSP and for setting up your Linux host system. Buildroot allows to place project-specific customizations outside of the Buildroot tree using extern custom layers. LinTech provides an Buildroot external layer[2] for the HBM11.

3 Building the BSP

This section will guide you through the general build process of the BSP.

3.1 System requirements

3.1.1 Build requirements

For running the build system a Linux host system is required. Please follow the system requirements in the Buildroot manual for setting up the host system.

For fetching the Buildroot and the external layer sources `git`[3] is required.

3.1.2 Flash requirements

For flashing the firmware images the HBM11 flash tool requires a simple TFTP server, e.g. *tftpd-hpa*[4], to be installed on the Linux host system. Please refer to your Linux distribution about setting up the TFTP server properly. Make sure the root directory is `/srv/tftp`.

This is a sample configuration file for the *tftpd-hpa* tool:

```
$ cat /etc/conf.d/tftpd
TFTPD_ARGS="--secure /srv/tftp/"
```

Additionally to the TFTP server, a wired Ethernet connections needs to be configured to allow the HBM11 flash tool to download the image files from the TFTP directory to the HBM11 RAM. The following steps are aimed for the tool *Network Manager*, which is the common network management tool for Linux systems.

Get the systems Ethernet device names, e.g.: enp3s0:

```
$ nmcli device
enp3s0       ethernet  connected  Wired connection 1
lo           loopback  unmanaged  --
```

Add the Ethernet configuration:

```
$ nmcli connection add type ethernet ifname enp3s0
Connection 'ethernet-enp3s0' (719469cb-5860-4356-99be-e2ee07f66245) successfully
added.
```

[1] https://buildroot.org/downloads/manual/manual.html

[2] https://bitbucket.org/lintech-berlin/buildroot-external-hbm11

[3] https://git-scm.com

[4] http://www.kernel.org/pub/software/network/tftp/tftp-hpa

Modify the connection parameters:

```
$ nmcli connection modify ethernet-enp3s0 connection.id HBM11-FLASH
$ nmcli connection modify HBM11-FLASH ethernet.mac-address 00:19:B8:00:00:01
$ nmcli connection modify HBM11-FLASH ipv4.addresses 10.0.0.1/24
```

3.1.3 Serial console

You can easily communicate with the Linux system running on the HBM11 using a USB-to-Serial converter, e.g. from FTDI. The default serial baud rate is 115200.

Run a console application, e.g. using the `ckermit`[5] tool. Note, that in some Linux distributions the command is `kermit` (without a `c`):

```
$ sudo ckermit ~/.kermrc
```

You can use this a sample ckermit configuration file:

```
$ cat ~/.kermrc
set line /dev/ttyUSB0
set speed 115200
set reliable
fast
set carrier-watch off
set handshake none
set flow-control none
set prefixing all
set file type bin
set file name lit
set rec pack 4096
set send pack 4096
set window 5
connect
```

Note, that you need to set the serial port your Linux system uses for the serial converter, e.g. checking with `dmesg`:

```
usb 3-4.3: FTDI USB Serial Device converter now attached to ttyUSB0
```

[5] http://www.kermitproject.org/ck90.html

3.2 Build process

The following steps will guide you to the build process.

Create a fresh project directory, e.g.

```
$ mkdir -p hbm11/bsp
```

Change into the directory:

```
$ cd hbm11/bsp
```

Clone Buildroot:

```
$ git clone git://git.buildroot.net/buildroot -b 2018.02.x
```

Clone the LinTech HBM11 external layer:

```
$ git clone https://bitbucket.org/lintech-berlin/buildroot-external-hbm11.git
```

Create an output folder for your build:

```
$ make BR2_EXTERNAL=$PWD/buildroot-external-hbm11 O=$PWD/output \
-C buildroot hbm11_defconfig
```

Change into the directory:

```
$ cd output
```

Build the image:

```
$ make
```

Install the image files into the tftp directory

```
$ make install
```

Start the HBM11 flash tool:

```
$ sudo make flash
```

The flash process takes some time. When finished your HBM11 is ready to boot!

4 Running the BSP

The BSP will boot into a virtual terminal session:

```
Welcome to Buildroot OS for HBM11
hbm11 login:
```

The login user name is **root**. No password is required.

4.1 Audio

The audio devices are unmuted when booting the system.

4.1.1 Mute/Unmute

To mute all audio devices type:

```
# audioctl mute
```

To unmute all audio devices type:

```
# audioctl unmute
```

4.1.2 Speaker test

To check if the soundcard driver is loaded correctly type:

```
# aplay -L
null
    Discard all samples (playback) or generate zero samples (capture)
sysdefault:CARD=hbm11audio
    hbm11-audio,
    Default Audio Device
```

Run *speaker-test* to test playback:

```
# speaker-test -c 2 -t wav
Playback device is default
Stream parameters are 48000Hz, S16_LE, 2 channels WAV file(s) Rate set to
48000Hz (requested 48000Hz) Buffer size range from 64 to 16384 Period size range
from 32 to 8192 Using max buffer size 16384 Periods = 4 was set period_size =
4096 was set buffer_size = 16384
 0 - Front Left
 1 - Front Right
```

4.1.3 Record

An external audio source can be connected to the analog line-in input in order to record a sound file which can then be played back:

```
# arecord -D linein -f dat -d 10 /tmp/test.wav
Recording WAVE 'test.wav' : Signed 16 bit Little Endian, Rate 48000 Hz, Stereo

# aplay /tmp/test.wav
Playing WAVE 'test.wav' : Signed 16 bit Little Endian, Rate 48000 Hz, Stereo
```

The HBM11 supports full duplex support for simultaneously recording and playing back:

```
# arecord -D linein -f dat | aplay –f dat
Recording WAVE 'null' : Signed 16 bit Little Endian, Rate 48000 Hz, Stereo
Playing raw data 'stdin' : Signed 16 bit Little Endian, Rate 48000 Hz, Stereo
```

4.1.4 Volume Control

Use *amixer* to show the current volume of the 'Master' device:

```
# amixer
Simple mixer control 'Master',0
  Capabilities: volume
  Playback channels: Front Left - Front Right
  Capture channels: Front Left - Front Right
  Limits: 0 - 255
  Front Left: 255 [100%]
  Front Right: 255 [100%]
```

Note, that amixer only shows the mixer control information if some audio was played before, e.g. using speaker-test or aplay.

To change the volume type:

```
# amixer set Master 5%-

# amixer set Master 5%+
```

4.2 S/PDIF

An external audio source can be connected to the digital S/PDIF input in order to record a sound file which can then be played back:

```
# arecord -D spdifin -f dat -d 10 /tmp/test.wav
Recording WAVE 'test.wav' : Signed 16 bit Little Endian, Rate 48000 Hz, Stereo

# aplay /tmp/test.wav
Playing WAVE 'test.wav' : Signed 16 bit Little Endian, Rate 48000 Hz, Stereo
```

The HBM11 supports full duplex support for simultaneously recording and playing back:

```
# arecord –D spdifin –f dat | aplay –f dat
```

4.3 Network

4.3.1 Ethernet

Bring the ethernet interface up:

```
# ip link set eth0 up
Micrel KSZ8081 or KSZ8091 2188000.ethernet-1:00: attached PHY driver [Micrel
KSZ8081 or KSZ8091] (mii_bus:phy_addr=2188000.ethernet-1:00, irq=107)
```

After plugging in the ethernet cable the following system message is printed to the console:

```
fec 2188000.ethernet eth0: Link is Up - 100Mbps/Full - flow control rx/tx
```

The ethernet interface is now ready to use and an IP address can be obtained using the *udhcpc* tool:

```
# udhcpc -i eth0
udhcpc: sending discover
udhcpc: sending select for 192.168.178.83
udhcpc: lease of 192.168.178.83 obtained, lease time 864000
deleting routers
adding dns 192.168.178.1
```

ifconfig shows that the ehternet interface is up and running:

```
# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 5A:7B:39:A8:A0:AA
          inet addr:192.168.178.83  Bcast:192.168.178.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:128 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:10360 (10.1 KiB)  TX bytes:684 (684.0 B)
```

4.3.2 WLAN

4.3.2.1 Connection with wpa_cli

This connection method allows scanning for available networks, making use of wpa_cli, a command line tool which can be used to configure wpa_supplicant.

```
# wpa_supplicant -B -i wlan0 -c /etc/wpa_supplicant.conf
Successfully initialized wpa_supplicant
```

At this point run:

```
# wpa_cli
```

This will present an interactive prompt (>), which has tab completion and descriptions of completed commands.

Use the *scan and scan_results* commands to see the available networks:

```
> scan
OK
<3>CTRL-EVENT-SCAN-RESULTS
> scan_results
bssid / frequency / signal level / flags / ssid
00:00:00:00:00:00 2462 -49 [WPA2-PSK-CCMP][ESS] MYSSID
11:11:11:11:11:11 2437 -64 [WPA2-PSK-CCMP][ESS] ANOTHERSSID
```

To associate with MYSSID, add the network, set the credentials and enable it:

```
> add_network
0
> set_network 0 ssid "MYSSID"
> set_network 0 psk "passphrase"
> enable_network 0
<2>CTRL-EVENT-CONNECTED - Connection to 00:00:00:00:00:00 completed (reauth)
[id=0 id_str=]
```

Finally save this network in the configuration file:

```
> save_config
OK
```

```
> quit
```

Once association is complete, you must obtain an IP address using:

```
# udhcpc -i wlan0
udhcpc: sending discover
udhcpc: sending select for 192.168.178.77
udhcpc: lease of 192.168.178.77 obtained, lease time 864000
deleting routers
adding dns 192.168.178.1
```

4.3.2.2 Connecting with wpa_passphrase

This connection method allows quickly connecting to a network whose SSID is already known, making use of *wpa_passphrase*, a command line tool which generates the minimal configuration needed by *wpa_supplicant*. For example:

```
# wpa_passphrase MYSSID passphrase >> /etc/wpa_supplicant.conf
```

Now start *wpa_supplicant* with:

```
# wpa_supplicant -B -i wlan0 -c /etc/wpa_supplicant.conf
Successfully initialized wpa_supplicant
```

Once association is complete, you must obtain an IP address using:

```
# udhcpc -i wlan0
udhcpc: sending discover
udhcpc: sending select for 192.168.178.77
udhcpc: lease of 192.168.178.77 obtained, lease time 864000
deleting routers
adding dns 192.168.178.1
```

4.3.3 Bluetooth Audio (A2DP)

Load the Bluetooth firmware image into the Bluetooth module (needs up to 10 seconds):

```
# btloadfw
```

Start the Bluetooth daemon:

```
# /usr/libexec/bluetooth/bluetoothd &
```

Start the simple Bluetooth agent:

```
# bluetooth-agent &
```

Open and initialize HCI device:

```
# hciconfig hci0 up
```

Enable page and inquiry scan:

```
# hciconfig hci0 piscan
```

Start the Bluetooth ALSA daemon:

```
# bluealsa -S --disable-hsp --disable-hfp &
```

Start the Bluetooth ALSA player:

```
# bluealsa-aplay --profile-a2dp 00:00:00:00:00:00
```

Connect your A2DP source to the HBM11 Bluetooth device and start playing some audio.

4.3.4 Antenna

The BSP contains a small helper tool *xant* to easy select the antenna path used by the wireless module. Called without any arguments returns the current antenna path, e.g.:

```
# xant
0: on-board antenna
```

Invoking *xant* with 0 for the on-board or 1 for the external path, e.g.:

```
# xant 1
Enabling external antenna
```

4.4 Multimedia Keys

You can use the tool *evtest* to check for input events:

```
# evtest /dev/input/event1
Input driver version is 1.0.1
Input device ID: bus 0x19 vendor 0x1 product 0x1 version 0x100 Input device
name: "gpio-keys"
Supported events:
  Event type 0 (EV_SYN)
  Event type 1 (EV_KEY)
    Event code 114 (KEY_VOLUMEDOWN)
    Event code 115 (KEY_VOLUMEUP)
    Event code 128 (KEY_STOP)
    Event code 143 (KEY_WAKEUP)
    Event code 163 (KEY_NEXTSONG)
    Event code 164 (KEY_PLAYPAUSE)
    Event code 165 (KEY_PREVIOUSSONG)
Properties:
Testing ... (interrupt to exit)
Event: time 7332.855101, type 1 (EV_KEY), code 115 (KEY_VOLUMEUP), value 1
Event: time 7332.855101, ------------- SYN_REPORT ------------
Event: time 7333.045168, type 1 (EV_KEY), code 115 (KEY_VOLUMEUP), value 0
Event: time 7333.045168, ------------- SYN_REPORT ------------
Event: time 7334.175137, type 1 (EV_KEY), code 114 (KEY_VOLUMEDOWN), value 1
Event: time 7334.175137, ------------- SYN_REPORT ------------
Event: time 7334.335247, type 1 (EV_KEY), code 114 (KEY_VOLUMEDOWN), value 0
Event: time 7334.335247, ------------- SYN_REPORT ------------
Event: time 7335.055098, type 1 (EV_KEY), code 165 (KEY_PREVIOUSSONG), value 1
Event: time 7335.055098, ------------- SYN_REPORT ------------
Event: time 7335.205171, type 1 (EV_KEY), code 165 (KEY_PREVIOUSSONG), value 0
Event: time 7335.205171, ------------- SYN_REPORT ------------
Event: time 7335.765225, type 1 (EV_KEY), code 163 (KEY_NEXTSONG), value 1
Event: time 7335.765225, ------------- SYN_REPORT ------------
Event: time 7335.865107, type 1 (EV_KEY), code 163 (KEY_NEXTSONG), value 0
Event: time 7335.865107, ------------- SYN_REPORT ------------
Event: time 7336.466143, type 1 (EV_KEY), code 128 (KEY_STOP), value 1
Event: time 7336.466143, ------------- SYN_REPORT ------------
Event: time 7336.645303, type 1 (EV_KEY), code 128 (KEY_STOP), value 0
```

```
Event: time 7336.645303, ---------- SYN_REPORT ------------
Event: time 7337.125206, type 1 (EV_KEY), code 164 (KEY_PLAYPAUSE), value 1
Event: time 7337.125206, ---------- SYN_REPORT ------------
Event: time 7337.325207, type 1 (EV_KEY), code 164 (KEY_PLAYPAUSE), value 0
Event: time 7337.325207, ---------- SYN_REPORT ------------
```

4.5 USB Host

The Linux system automatically detects if a USB device, e.g. a pendrive, has been plugged in:

```
usb 1-1: new high-speed USB device number 2 using ci_hdrc
usb 1-1: New USB device found, idVendor=1307, idProduct=0165
usb 1-1: New USB device strings: Mfr=1, Product=2, SerialNumber=3
usb 1-1: Product: USB Mass Storage Device
usb 1-1: Manufacturer: USBest Technology
usb 1-1: SerialNumber: 0809231243e61a
usb-storage 1-1:1.0: USB Mass Storage device detected
scsi host0: usb-storage 1-1:1.0
scsi 0:0:0:0: Direct-Access     USB 2.0  Flash Drive      0.00 PQ: 0 ANSI: 2
sd 0:0:0:0: [sda] 3948544 512-byte logical blocks: (2.02 GB/1.88 GiB)
sd 0:0:0:0: [sda] Write Protect is off
sd 0:0:0:0: [sda] Mode Sense: 00 00 00 00
sd 0:0:0:0: [sda] Asking for cache data failed
sd 0:0:0:0: [sda] Assuming drive cache: write through
 sda:
sd 0:0:0:0: [sda] Attached SCSI removable disk
```

You can check that USB device:

```
# lsusb
Bus 001 Device 001: ID 1d6b:0002
Bus 001 Device 003: ID 1307:0165
```

Create a mount point and mount the USB device, e.g.:

```
# mkdir /media/usb-drive
# mount /dev/sda /media/usb-drive
```

Unmount the USB device:

```
# umount /dev/sda
```

4.6 LEDs

Mute the audio device so the green status LED is turned off:

```
# audioctl mute
```

Turn red LED on and of:

```
# echo -n 255 > /sys/class/leds/red/brightness
# echo -n 0 > /sys/class/leds/red/brightness
```

Enable trigger on blue LED to signal wifi activity:

```
# echo -n mmc0 > /sys/class/leds/blue/trigger
```