### Revision History

<table>
<thead>
<tr>
<th>Ver</th>
<th>Description</th>
<th>Author</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1.0</td>
<td>Initial version</td>
<td>Wei Xianya</td>
<td>2019-05-27</td>
</tr>
<tr>
<td>V2.0</td>
<td>Modify testing</td>
<td>Zhou Lijun</td>
<td>2019-10-18</td>
</tr>
</tbody>
</table>
1. Introduction

1.1. About this Manual
This manual is intended to provide the user with an overview of the board and benefits, complete features specifications, and set up procedures. It contains important safety information as well.

1.2. Feedback and Update to this Manual
To help our customers make the most of our products, we are continually making additional and updated resources available on the Boardcon website (www.boardcon.com, www.armdesigner.com).
These include manuals, application notes, programming examples, and updated software and hardware. Check in periodically to see what’s new!
When we are prioritizing work on these updated resources, feedback from customers is the number one influence, If you have questions, comments, or concerns about your product or project, please no hesitate to contact us at support@armdesigner.com.

1.3. Limited Warranty
Boardcon warrants this product to be free of defects in material and workmanship for a period of one year from date of buy. During this warranty period Boardcon will repair or replace the defective unit in accordance with the following process:
A copy of the original invoice must be included when returning the defective unit to Boardcon. This limited warranty does not cover damages resulting from lighting or other power surges, misuse, abuse, abnormal conditions of operation, or attempts to alter or modify the function of the product.
This warranty is limited to the repair or replacement of the defective unit. In no event shall Boardcon be liable or responsible for any loss or damages, including but not limited to any lost profits, incidental or consequential damages, loss of business, or anticipatory profits arising from the use or inability to use this product.
Repairs make after the expiration of the warranty period are subject to a repair charge and the cost of return shipping. Please contact Boardcon to arrange for any repair service and to obtain repair charge information.
Content

1 EMA40i Introduction ................................................................. 4
   1.1 Summary ................................................................. 4
   1.2 Processor Features ..................................................... 4
   1.3 EMA40i specifications .................................................. 5

2. Compiler Environment ......................................................... 6
   2.1 VMware10.0+ubuntu16.04 ............................................. 6
   2.2 Install Tools ............................................................. 6

3. Compile the Source .............................................................. 7
   3.1 Unzip & Configuration .................................................. 7
   3.2 Compile Uboot ............................................................ 8
   3.3 Compile Kernel ........................................................... 8
   3.4 Compile Buildroot ....................................................... 9
   3.5 Compile QT ............................................................... 9
      3.5.1 Compile QT Library ............................................... 9
      3.5.2 Compile QT app demo ......................................... 10
   3.6 Compile sdk_lib Library .............................................. 10
   3.7 Pack Firmware .......................................................... 10
   3.8 Automatic Compilation ............................................... 11

4. Burning Guide ................................................................. 13
   4.1 Install Firmware Tool ................................................ 13
   4.2 Burn image to eMMC via USB ...................................... 13

5 EMA40i Application Guidance ................................................ 16
   5.1 Serial Terminal ........................................................ 16
   5.2 HDMI Display .......................................................... 17
   5.3 LVDS and Touch ....................................................... 17
   5.4 SD Card ................................................................. 17
   5.5 USB Host ............................................................... 17
   5.6 SSD (SATA 2.0) ........................................................ 18
   5.7 EC20(4G) ............................................................... 18
   5.8 GPS ..................................................................... 19
   5.9 RTC(PCF8563) ........................................................... 20
   5.10 Video Player ........................................................... 20
      5.10.1 Play Video ...................................................... 20
      5.10.2 xplayerdemo .................................................... 23
   5.11 Audio Player .......................................................... 23
   5.12 Recording .............................................................. 25
   5.13 Ethernet ............................................................... 26
   5.14 WiFi ................................................................. 27
   5.15 Bluetooth ............................................................ 29
   5.16 3D ................................................................. 32
   5.17 RS485 ................................................................. 33
   5.18 RS232&UART (UART5 / UART6 / UART4 / UART7) ............. 34
1 EMA40i Introduction

1.1 Summary

EMA40i is a single board computer featuring an Allwinner A40i Processor, comes with 1GB DDR3 RAM, 4GB eMMC, and other rich interfaces specifically designed for intelligent industrial control applications such as industrial control, communications and measurement. EMA40i provides a standard Gigabit Ethernet port and a 10-pin Extra GbE port. Supports HDMI 1.4, RGB, MIPI SDI and dual-LVDS output. The SBC is equipped with a microSD slot and a M.2 slot with NAME SSD support. A mini-PCIe slot supports 4G module and is accompanied by a Nano-SIM slot. Other features include USB 2.0 host ports, as well as a USB2.0 OTG. There’s also a RS232 DB9 port, as well as 4x 4-pin headers for RS232, UART(TTL) and USB host. EMA40i is equipped with WiFi 802.11ac and Bluetooth 4.0, as well as optional support for GPS, 4G LTE connectivity.

1.2 Processor Features

- **CPU**
  Quad-core ARM Cortex-A7 CPU Architecture, the most power efficient CPU core ARM’s ever developed.

- **GPU**
  Mali400 MP2

- **Video Engine**
  Supports mainstream high-definition video decoding including H.264, H.263, MPEG1/2/4, xvid, Sorenson Spark, VP6/8, AVS/AVS+, WMV7, WMV8 by 1080p@60fps. In the aspect of video encoding, the A40i supports 1080p@45fps H.264 encoding ability.

- **Camera**
  Supports dual COMS sensor parallel interfaces and 4-channel TVIN, which can easily finish multi channel video recording.

- **Display**
  Content can be display on 4-lane MIPI DSI display, or RGB panel, or LVDS panel. TV-out on HDMI V1.4 is also supported.

- **Audio**
  Integrated audio codec with 24 bit/192kHz DAC playback, and supports I2S/PCM interface for connecting to an external audio codec. I2C/PCM interface includes eight channels of TDM with sampling precision up to 32 bit/192kHz.

- **Memory**
  Supports external memory interface to NAND Flash, SD/EMMC, Nor Flash and SDRAM port. SDRAM port can be configured to support LPDDR2, LPDDR3, DDR2, DDR3, and DDR3L.
1.3 EMA40i specifications

* **Processor** – Allwinner A40i Quad-core ARM Cortex-A7 MPCore Processor @ 1.2 GHz
* **RAM** – 1GB DDR3
* **eMMC Flash** – 4GB
* **Interfaces** – Ethernet, 5x UART, TV-IN, 3x USB Host, USB OTG, HDMI, M.2 SATA, PCI-E, CAN, RS485, LCD(RGB, LVDS, MIPI), Camera, GPIO, Audio I/O, SD, SIM, etc.
* **Operating system**: Linux3.10, Android7.0.1
* **Application** – Industrial control, communications and measurement, etc.
* **Dimension** – 140mm x 100mm
2. Compiler Environment

2.1 Vmware10.0+ubuntu16.04

Install Vmware10.0 in windows OS, and then install ubuntu16.04 in VMware to compile. There is no longer describes how to install Ubuntu system, if the user has not understood, please visit the official website of Ubuntu, the operating system is also available for free download at the official website [http://www.ubuntu.com/](http://www.ubuntu.com/)

*Note:* Linux should be complied by ubuntu 64-bit OS.

2.2 Install Tools

```
# sudo apt-get install build-essential
# sudo apt-get install zlib1g-dev
# sudo apt-get install flex
# sudo apt-get install libx11-dev
# sudo apt-get install gperf
# sudo apt-get install libncurses5-dev
# sudo apt-get install bison
# sudo apt-get install lsb-core
# sudo apt-get install lib32z1-dev
# sudo apt-get install g++-multilib
# sudo apt-get install lib32ncurses5-dev
# sudo apt-get install uboot-mkimage
# sudo apt-get install g++-4.4-multilib
```

If the compilation encountered error, please according to the error message, the installation of the corresponding software package
3. Compile the Source

3.1 Unzip & Configuration

**Step 1**, unzip the source.

```bash
$tar -xvf A40i-linux-v1.3.tar.bz2
$cd a40i-linux-v1.3
```

**Step 2**, configuration environment

```bash
$ ./build.sh config
```

Welcome to mkscript setup progress

All available chips:
- 0. sun8iw11p1

Choice: 0

All available platforms:
- 0. android
- 1. dragonboard
- 2. linux
- 3. camdroid

Choice: 2

All available kernel:
- 0. linux-3.10

Choice: 0

All available boards:
- 0. a40i-boardcon
- 1. a40i-p3

Choice: 0

All available buildroot:
- 0. buildroot
- 1. buildroot-201611

Choice: 1

select buildroot-201611

All available float:
- 0. gnueabi
- 1. gnueabihf

Choice: 1

select gnueabihf

All available qt version:
- 0. 5.9.0

Choice: 0

select 5.9.0

create misc_config gnueabihf
Step 3, compile

$ ./build.sh

3.2 Compile Uboot

Method 1 compile uboot

$ cd brandy
$ ./build.sh -p sun8iw11p1

Method 2 Separately compile each part of the boot bin file

$ cd brandy/u-boot-2014.07
$ make distclean
$ make sun8iw11p1_config
$ make -j32

After compiling, **u-boot-sun8iw11p1.bin** are generated in the directory:

`lichee\tools\pack\chips\sun8iw11p1\bin\u-boot-sun8iw11p1.bin`

**NOTE:** After compiling separately, you need to repackage.

Back to **a40i-linux-v1.3** directory, execute ./build.sh, then ./build.sh pack, the updated **uboot** is compiled into the new package.

3.3 Compile Kernel

$ cd lichee/linux-3.10
$ make menuconfig ARCH=arm
$make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf-

If it is a soft floating-point toolchain, the last sentence is changed to bold `-gnueabi`-

### 3.4 Compile Buildroot

If add some packages, do the following (take add **mplayer** as an example)

```bash
$ cd buildroot-201611/
$ cp -f configs/sun8iw11p1_defconfig ./
$ make
```

Select **Audio and video applications** ----> **mplayer**, than *exit****save*, will get the new .config

```bash
$cp -f .config configs/sun8iw11p1_defconfig
$ cd ..
$ rm -f out/sun8iw11p1/linux/common/buildroot/.config
$ rm -f out/sun8iw11p1/linux/common/buildroot/..config.tmp
$./build.sh
```

### 3.5 Compile QT

#### 3.5.1 Compile QT Library

Before compiling the qt library, you need to execute **./build.sh config** and **./build.sh** to ensure that libraries other than Qt are up to the latest.

```bash
$ ./build.sh config
$ ./build.sh
$ cd buildroot-201611/dl
$ tar xzvf qt-everywhere-opensource-src-5.9.0.tgz
$ cp -f setenvs.sh qt-everywhere-opensource-src-5.9.0/
$ cd qt-everywhere-opensource-src-5.9.0/
$ source ./setenvs.sh
$makeconfig
$makeall
$makeinstall
$cd ../../
$./build.sh
```

QT5.9.0 are generated in the following directory:

```
lichee/out/sun8iw11p1/linux/common/buildroot/target/usr/local/Qt-5.9.0
```

or

```
lichee/buildroot-201611/dl/qt-everywhere-opensource-src-5.9.0/QT5.9.0
```

If there is something in the QT5.9.0 directory, the Qt library has been successfully.

Compiled: ~/lichee/buildroot-201611/dl/qt-everywhere-opensource-src-5.9.0/Qt-5.9.0$ ls
bin doc include lib mkspecs plugins qml translations
Note:
If do not modify the QT library later, or do not clear the above directory, it does not need to QT library directory to compile again, just needs to execute ./build.sh and ./build.sh pack to package the firmware.

3.5.2 Compile QT app demo

➢ Compile all QT app demo
$.cd buildroot-201611/target/user_rootfs_misc/qt_demo/
$build.sh
or
$rebuildAll.sh
Rebuildall.sh will make disclean before compilation.

➢ Compile qt application demo separately
Enter a single demo directory and execute the following commands:
$cd CameraUI/
$.makeCameraUI

$cd Launcher/
$.makeLauncher

$cd MediaUI/
$.makeMediaUI

3.6 Compile sdk_lib Library

The compilation of sdk_lib is already integrated in ./build.sh, and for development purposes, this section only explains how to compile separately.
$cd buildroot/target/user_rootfs_misc/sdk_lib
$make cleanall
$make

3.7 Pack Firmware

After the compilation is completed, pack
# ./build.sh pack
The final burnable image is `sun8iw11p1_linux_a40i-boardcon_uart0.img`, which is in the directory of `a40i-linux-v1.3/tools\pack`.

**Note:** A40i login: `root`

### 3.8 Automatic Compilation

The steps for automatic compilation are as follows.

```
$ cd a40i-linux-v1.3
$ ./build.sh config
```
Customize the embedded system based on your idea.

bash scripts:

```
$ ./build.sh
$ ./comp_qtLib-590_only.sh
$ ./build.sh pack

```

`comp_qtLib-590_only.sh` is the compilation of QT library, you can see in the directory of `a40i-linux-v1.3/comp_qtLib-590_only.sh`. 
4. Burning Guide

4.1 Install Firmware Tool

Please install PhoenixSuit1.0.6 (Path: CD\Tools\PhoenixSuit1.0.6) according to the PC system.

4.2 Burn image to eMMC via USB

Open the tool PhoenixSuit

Preparation Work:
1. Connect Device to PC with USB Cable
2. Turn on the Device’s ADB (USB Debug Mode: Go to Settings>Developer Options>USB debugging ON)

Select “Firmware”
Click “Image” and select the image (Path: CD\image)
connect PC and board with Micro USB cable, keep pressing the **Recover Key** and power on until the windows PC pops up dialogue box.
Select “Yes” to enter the format upgrade mode. After the system update is completed, it will restart automatically.
5 EMA40i Application Guidance

5.1 Serial Terminal

Connect the board and PC with USB Serial cable, then power on, the terminal will output startup information.

![Serial Terminal Configuration](image1)

![Terminal Output](image2)
5.2 HDMI Display

Connect the board and HDMI monitor with a HDMI cable, then power on. After the board boot into Linux Qt, HDMI monitor will show QT UI.

5.3 LVDS and Touch

Connect LVDS to the board, then power on the board. After the board boot into Linux Qt, The LVDS will auto display QT UI.

EMA40 supports HDMI & LVDS display sync.

5.4 SD Card

The system will automatically mount the SD card.

```
# ls /mnt/sdcard/mmcblk1p1/
```

```
ls /mnt/sdcard/mmcblk1p1/test_file/
```

5.5 USB Host

Insert USB device (e.g. U-disk) to USB Host, the system will automatically read the contents.

```
# ls /mnt/usb/sda4/
```
5.6 SSD (SATA 2.0)

*Note: SSD does not support hot plugging, please insert before power on.*

Insert SSD mass storage device to SATA interface, then power the board when system booting it can auto mount.

```
# ls /mnt/usb/sda1/
```

5.7 EC20(4G)

Execute the follow commands to modify /etc/resolv.conf file:
```
# vi /etc/resolv.conf
Add:
nameserver 8.8.8.8
nameserver 192.168.0.2
#
```
```
# pppd call wcdma&
# ping www.boardcon.com
```
5.8 GPS

Execute the following command to test GPS.

```
# cat /dev/ttyS4
```
Customize the embedded system based on Your Idea

5.9 RTC(PCF8563)

```
#date -s "2019-05-28 15:20:00" (set the system time)
#hwclock -w
#hwclock
```

5.10 Video Player

5.10.1 Play Video

Put MP4 files to the SD card/U-disk and power on. Click the video player icon to play:
Customize the embedded system based on your idea.
Customize the embedded system based on your idea.
5.10.2 xplayerdemo

Cancel `/etc/init.d/runqt` in the `/etc/init.d/rcS` and reboot.

```bash
# cd /
# vi /etc/init.d/rcS //find out runqt, than cancel runqt.
# reboot
```

Restart and enter the root directory

```bash
# cd /
# chmod 755 /xplayerdemo
# /xplayerdemo
# set url: /mnt/usb/sda4/test_file/Xueziqian-1080P.mp4 (the directory of video)
# show media info
# play
# pause
```

5.11 Audio Player

Put MP3 files to the SD card/U-disk and power on. Click the Audio Player icon:
Customize the embedded system based on your idea.
5.12 Recording

```
# arecord -l

# arecord -l
**** List of CAPTURE Hardware Devices ****
card 0: audiocodec [audiocodec], device 0: SUNXI-CODEC sun8iw11codec-0 []
Subdevices: 0/1
Subdevice #0: subdevice #0
```

Cancel `/etc/init.d/runqt` in the `/etc/init.d/rcS` and reboot.

```
# cd /
# vi /etc/init.d/rcS  //find out runqt, than cancel runqt. In front of the /etc/init.d/runqt add #
# reboot
```

Restart and enter the root directory

```
# amixer cset numid=32 50
# amixer cset numid=25 50
```

Record:

```
# tinycap 123.wav -D "hw:0,0" -r 48000 -c 2 -f "S16_LE"
```
or

```bash
# arecord -D "hw:0,0" -r 48000 -c 2 -f "S16_LE" 123.wav
```

Play:

```bash
# aplay -D "hw:0,0" -f "S16_LE" 123.wav
or
# tinyplay 123.wav
```

5.13 Ethernet

Plug in an Ethernet cable (RJ45).

```bash
# ifconfig eth0 up
# udhcpc -i eth0 -R
# ping www.boardcon.com
```

```bash
# ifconfig eth0 up
192.168.0.151 eth00...```

```bash
# ping www.boardcon.com
PING www.boardcon.com (67.224.64.190) 56 data bytes
67.224.64.190 ping statistics:
100% packet loss
```
5.14 WiFi

**Step 1.** Execute the follow commands to search SSID.

```
# ifconfig wlan0 up
# iw dev wlan0 scan | grep SSID
```

![SSID Scan Output](image)

**Step 2.** Type the SSID and password from the list of available networks.

```
# ifconfig eth0 down
# wpa_passphrase Boardcon (Boardcon is wifi name)
# reading passphrase from stdin
boardcon43435656 (boardcon43435656 is the password)
```

```
network={
    ssid="Boardcon"
    #psk="boardcon43435656"
    psk=bafeb7f673814bf7e06002a652e6d9f1ed749312a54481b5fa8efdd53bfa1123
}
```

Copy `ssid(e.g. Boardcon)` and `#psk(e.g. boardcon43435656)` replace ssid and psk in the directory of `/etc/wpa_supplicant.conf`

```
# vi /etc/wpa_supplicant.conf
```
Step 3, add DNS server.

```bash
#vi /etc/resolv.conf
```

In the resolv.conf file add follow command:

```bash
nameserver 8.8.8.8
nameserver 192.168.0.2  # gateway
```

Step 4, Connect WiFi

**Method 1**

set the gateway

```bash
# ifconfig wlan0 192.168.0.165 up
# route add default gw 192.168.0.2 dev wlan0  # Must be add your Router gateway
```

Execute the follow command to connect the `wpa_supplicant` tool.

```bash
# wpa_supplicant -i wlan0 -Dnl80211 -c /etc/wpa_supplicant.conf -B
```

**Method 2**

Auto obtain wifi IP address:

Execute the follow commands to connect the `wpa_supplicant` tool.

```bash
# wpa_supplicant -i wlan0 -Dnl80211 -c /etc/wpa_supplicant.conf -B
# udhcpc -i wlan0
```

busbox apt-get install scrot

**Step 5**, ping

```bash
# ping www.boardcon.com
```
5.15 Bluetooth

# cd /
# source /qt5.9.sh
# openbt
# hciconfig hci0 up
# hciconfig hci0 piscan
# /usr/libexec/bluetooth/bluetoothd -C &
# /usr/libexec/bluetooth/obexd -a -d -n &

# hciconfig hci0 up
# hciconfig hci0 piscan
# hciconfig hci0 -a

# /usr/libexec/bluetooth/bluetoothd -C &
# /usr/libexec/bluetooth/obexd -a -d -n &

# hciconfig hci0 piscan
# bt-adapter -d (search BT device, press CTRL+C to exit after the Bluetooth device appears)
Customize the embedded system based on your idea.

Connect Bluetooth:
```
# bt-device -c CC:4B:73:04:D4:ED
```

Send file:
```
# bt-obex -p CC:4B:73:04:D4:ED /opt/test.jpg
```
Customize the embedded system based on your idea

```
# bt-obex -s /opt/ (Receive file. Default received path: ~root/.cache/obexd/)
```

```
bt-obex -s /opt/
```

```
bt-obex -s /opt/
```

```
bt-obex -s /opt/
```

```
bt-obex -s /opt/
```

```
bt-obex -s /opt/
```

```
bt-obex -s /opt/
```

```
bt-obex -s /opt/
```

```
bt-obex -s /opt/
```

```
bt-obex -s /opt/
```

```
bt-obex -s /opt/
```

```
bt-obex -s /opt/
```

```
bt-obex -s /opt/
```
5.16 3D

Execute the follow command to run 3D:

```
# glmark2-es2-fbdev
```

Return to the desktop after execute the command

```
# debug /Launcher
CTRL+C to exit.
```
5.17 RS485

Connect the RS485 ports of Board A and B with the test line.

For Board A, execute the follow commands to set RS485 as Receiver.
Customize the embedded system based on your idea

# echo 49 > /sys/class/gpio/export
# echo out > /sys/class/gpio/gpio49/direction
# echo 1 > /sys/class/gpio/gpio49/value  (Set high for receiving.)
# com /dev/ttyS7 115200 8 0 1

For Board B, execute the following commands to set RS485 as Transmitter.

# echo 49 > /sys/class/gpio/export
# echo out > /sys/class/gpio/gpio49/direction
# echo 0 > /sys/class/gpio/gpio49/value  (Set low for sending.)
# com /dev/ttyS7 115200 8 0 1

The Transmitter and receiver can be converted by executing the command

# echo 1 > /sys/class/gpio/gpio49/value  (Set high for receiving.)
# com /dev/ttyS7 115200 8 0 1

Or

# echo 0 > /sys/class/gpio/gpio49/value  (Set low for sending.)
# com /dev/ttyS7 115200 8 0 1

5.18 RS232&UART (UART5 / UART6 / UART4 / UART7)

Execute the commands as follow:

# cd /system/bin
# rz  (send “com” file)
# chmod 777 com
# cd /
Customize the embedded system based on Your Idea

Connect RX&TX, then execute the commands to run the test program at serial terminal:

```
# com /dev/ttyS5 115200 8 0 1 (Connect COM2 PIN2&3)
# com /dev/ttyS6 115200 8 0 1 (Connect RS232 PIN1&2)
# com /dev/ttyS4 115200 8 0 1 (Connect UART4 PIN1&2)
# com /dev/ttyS7 115200 8 0 1 (Connect UART7 PIN1&2)
```