MediaTek

Boot ROM Design Specification

Documents Number:

Preliminary Information

Revision: 2.00

Release Date: June, 16, 2006

Revision History

Revision	Date	Author	Comments
1.01	06/27/2002	Jensen Hu	Draft version
1.02	07/23/2002	Jensen Hu	Add Boot ROM chapter
1.03	06/01/2004	Shalyn Chua	Support additional command types, 32bits read and 32bits write.
1.04	07/08/2004	Shalyn Chua	Baud rate correction.
2.00	06/16/2006	Karen Hsu	Support security system

Table of contents

1 Boot ROM

Every MTK base-band chips embed a Boot ROM, which is used to store a simple boot program. Internal BOOT ROM will be selected if BOOT pin (GPIO0) is tied to GND once reset. The Boot ROM contains a small program for downloading data via UART, and this feature is applied to download program or factory testing.

1.1 Download Procedure

1.1.1 Description

Purpose of Boot ROM is to interpret commands received from UART1. An external download host is associated for loading download agent into internal SRAM, and starts its execution. The Boot ROM is useful in the absence of off-chip memory or when the external Flash memories have not been initialized yet.

1.1.2 Configuration

Boot ROM code is executed in ARM 32 bit mode, and the stack is allocated as follows:

	Boot ROM not support security	Boot ROM support security
Without TCM	0x40000000 ~ 0x400007FF	0x40000000 ~ 0x40000BFF
With TCM	0xA0000000 ~ 0xA0007FF	0xA0000000 ~ 0xA0000BFF

Boot ROM does not use interrupts, and the two interrupts FIQ and IRQ are masked after system reset. The UART driver in Boot ROM use polling mode to be able to run with minimal code size. The Boot ROM doesn't setup memory wait state. When accessing external RAM or external flash, a wait state configuration is required.

UART1 is the pre-defined communication channel, and its configuration as below:

- 8 bit
- No parity
- 1 stop bit
- Baud rate: 9600 baud if MCU runs at 13M, 19200 baud if MCU runs at 26M.
- No flow control.

Except the above description, the Boot ROM code does not modify default reset values of registers.

1.1.3 Commands

By default, Boot ROM code responses the transaction with data it receives, and the host can be aware of what operation is being processed. The host can emit another command, if it has already received the returned characters corresponding to the previous command. This handshake procedure is applicable for all command data send by the host.

Below table lists all commands, which are supported in Boot ROM code.

Command Name	Command Code	First Parameter	Second Parameter	Third Parameter	Data	Results
Start	0xA0, 0x0A, 0x50, 0x05	None	None	None	None	None
Write	0xA1	Base address 32 bits	Length 32 bits	None	Size 16 bits	Size 16 bits
Read	0xA2	Base address 32 bits	Length 32 bits	None	None	Size 16 bits
Checksum	0xA4	Base address 32 bits	Length 32 bits	None	None	Size 16 bits
Jump (non-secure version)	0xA8	Jump address 32 bits	None		None	None
Jump (secure version)			Address 32 bits	Length 32 bits		
32bits Write	0xAE	Base address 32 bits	Length 32 bits	None	Size 32 bits	Size 32 bits
32bits Read	0xAF	Base address 32 bits	Length 32 bits	None	None	Size 32 bits
Serial-Link	0xC1	None	None	None	None	None
Version	0xFF	None	None	None	None	None

One or two parameters and data should be associated with a command. The parameters of commands should follow the **MSB-first** rule to send these data.

The **Start** command is the first command to process any further operations. If Boot ROM never receives this command, Boot ROM code will not execute any commands.

The **Write/32bits Write** command starts to execute after reception of the write command header. Received data is written into RAM or HW registers. Boot ROM should read these parameters (address, length, data) and send it through UART.

The **Read/32bits Read** command starts to execute after reception of the read command header. Boot ROM should read these parameters (address, length) and send it through UART. Boot ROM will read data from address and length parameters, and send these data back through UART.

The **Checksum** command consists in computing a logical XOR of the whole data area.

The **Jump** command allows connected tool to execute another program by specifying a jump address, and exit the control of Boot ROM . However, in secure version, another two parameters, signature address and signature length of the program, should be provided, too. Because Boot ROM will verify this program before execute it.

The **Serial-Link** command means connected tool requests to do verification. It will trigger a lot of security system flow to verify if the connection is legal or not. Note that Boot ROM with security support will do a lot of restriction on other commands before connection triggers this verification flow. Any failure in the verification flow will make Boot ROM output an error code to UART and trap into endless loop.

The **Version** command is used to indicate if the Boot ROM supports security mechanism or not. For a normal Boot ROM, it will echo this command (0xFF) only. Otherwise, it will output the security version to UART instead of echo.

Note that the read or write command should consider the alignment problem carefully. If boot up by using internal Boot ROM and Boot ROM doesn't receive any data during 150ms in 26M environment, the Boot Rom will automatically jump to address 0 in flash.

1.1.4 Example