**Design Document for Serial Driver**

 

# Outline

This document describes the serial driver in Linux kernel of MVF TOWER BOARD (XTWR-VF600) with MVF SoC.

# Code to be added

In order to implement the serial driver, the following source and header file is added to the Linux Kernel Source tree.

* drivers/tty/serial/serial\_mvf.c

Serial Driver Source File

* arch/arm/plat-mxc/include/mach/mvf\_uart.h

Definitions for the Serial Driver

# Existing code to be changed

No modification will be made in source code since this is a newly added driver.

However, the following files are changed for the use of driver.

* drivers/tty/serial/Kconfig

Serial driver Configuration

* drivers/tty/serial/Makefile

Serial driver Makefile

# API of new functions

A standard framework will be implemented as a new serial driver.

Name of functions to be implemented are either mvf\_xxxxx() or serial\_mvf\_xxx().

Functions to be implemented are as follows.

mvf\_serial\_init

Initialization processing of the driver

mvf\_serial\_exit

Termination processing of the driver

serial\_mvf\_probe

Probe processing of the device

serial\_mvf\_remove

General deletion processing of the device

* uart\_ops

Create the following functions for uart\_ops struct.

Content of such functions conforms to a standard serial driver. Refer to the “Documentation/serial/driver” in Linux Kernel Source Tree.

mvf\_tx\_empty

As a callback function of tx\_empty member in uart\_ops struct

mvf\_set\_mctrl

As a callback function of set\_mctrl member in uart\_ops struct

mvf\_get\_mctrl

As a callback function of get\_mctrl member in uart\_ops struct

,

mvf\_stop\_tx

As a callback function of stop\_tx member in uart\_ops struct

mvf\_start\_tx

As a callback function of start\_tx member in uart\_ops struct

mvf\_stop\_rx

As a callback function of stop\_rx member in uart\_ops struct

mvf\_enable\_ms

As a callback function of enable\_ms member in uart\_ops struct

mvf\_break\_ctl

As a callback function of break\_ctl member in uart\_ops struct

mvf\_startup

As a callback function of startup member in uart\_ops struct

mvf\_shutdown

As a callback function of shutdown member in uart\_ops struct

mvf\_set\_termios

As a callback function of set\_termios member in uart\_ops struct

mvf\_type

As a callback function of type member in uart\_ops struct

mvf\_release\_port

As a callback function of release\_port member in uart\_ops struct

mvf\_request\_port

As a callback function of request\_port member in uart\_ops struct

mvf\_config\_port

As a callback function of config\_port member in uart\_ops struct

mvf\_verify\_port

As a callback function of verify\_port member in uart\_ops struct

* console struct

Create the following functions as console struct member functions of serial driver.

mvf\_console\_write

As a callback function of writel function

mvf\_console\_setup

Console initialization

Console Initial Setting

Speed : 9600bps (TBD: device clock-dependent)

Data bits: 8bits

Parity: none

Stop bits: 1bit

* dev\_pm\_ops

Create the following functions as dep\_pm\_ops struct functions for Power Management. (Power Management will be implemented in Release 4.)

. suspend = serial\_mvf\_suspend

suspend processing (TBD: Release 4)

.resume = serial\_mvf\_resume()

resume processing (TBD: Release 4)

For Release 1, minimum functions for console I/O will be implemented. UART driver implementation will be done on Release 2.

Other than above, create functions to be used locally in this source as static function or macro.

# Expected register settings

This serial driver will implement RS232, and not implement ISO-7816 and CEA709.1B. Therefore, registers to be set are as follows.

UART\_BDH: UART Baud Rate Register

UART\_BDL: UART, Baud Rate Register

UART\_C1: UART Control Register 1

UART\_C2: UART Control Register 2

UART\_S1: UART Status Register 1

UART\_S2: UART Status Register 2

UART\_C3: UART Control Register 3

UART\_D: UART Data Register

For the one below, set 0.

UART\_C4: Control Register

# Expected functionality and usage

The second release of this serial driver will be implemented with the Asynchronous Serial ports (minimum 2: 1 RS232 /w RTS/CTS/DSR/DTR control signals, 1 RS232 TXD/RDX/RTS/CTS) as a standard UART function.

It will be used as a console or a tty serial device.

This driver will enable all serial ports (6CH).

Serial port No.1 is used for console output.

# Any other pertinent information

None