PS UART

Introduction

The UART operations are controlled by the configuration and mode registers. The state of the FIFOs, modem signals, and other controller functions are read using the status, interrupt
status, and modem status registers.The controller is structured with separate RX and TX data paths. Each path includes a 64-byte FIFO. The controller serializes and deserializes data
in the TX and RX FIFOs and includes a mode switch to support various loopback configurations for the RxD and TxD signals. The FIFO interrupt status bits support polling or interrupt
driven handler. Software reads and writes data bytes using the RX and TX data port registers.When using the UART in a modem-like application, the modem control module detects and
generates the modem handshake signals and also controls the receiver and transmitter paths according to the handshaking protocol.
The PS Uart is in Zynq Ultrascale+ MpSoC and Zynq platforms.

HW IP Features

- Programmable baud rate generator
- 64-byte receive and transmit FIFOs
- Programmable protocol:° 6, 7, or 8 data bits 1, 1.5, or 2 stop bits
- Odd, even, space, mark, or no parity
- Parity, framing and overrun error detection
- Line-break generation and detection
- Interrupts generation
- Automatic echo, local loopback, and remote loopback channel modes
- Modem control signals: CTS, RTS, DSR, DTR, RI, and DCD

Features supported in driver

- Programmable baud rate generator
- 64-byte receive and transmit FIFOs
- Programmable protocol:° 6, 7, or 8 data bits 1, 1.5, or 2 stop bits
- Odd, even, space, mark, or no parity
- Parity, framing and overrun error detection
- Line-break generation and detection
- Interrupts generation
- Automatic echo, local loopback, and remote loopback channel modes
- Modem control signals: CTS, RTS, DSR, DTR, RI, and DCD

Missing Features, Known Issues and Limitations

- None

Kernel Configuration

To enable the uart driver in the linux kernel you either have to integrate it or build it as kernel module (.ko). you can enable it with:

```
CONFIG_SERIAL_XILINX_PS_UART=y
CONFIG_SERIAL_XILINX_PS_UART_CONSOLE=y
```

When more than two UARTs are required in the system (including PL UARTs) the user should also configure the following configuration item to increase the number of UART ports.
The driver statically allocates port data structures based on this configuration item. A kernel error similar to "Cannot get uart_port structure" may be seen during kernel boot when this is an issue.

```
CONFIG_SERIAL_XILINX_NR_UARTS=<number of expected UARTs>
```

Devicetree

Here's how the devicetree entry could look like.
https://www.kernel.org/doc/Documentation/devicetree/bindings/serial/cdns%2Cuart.txt

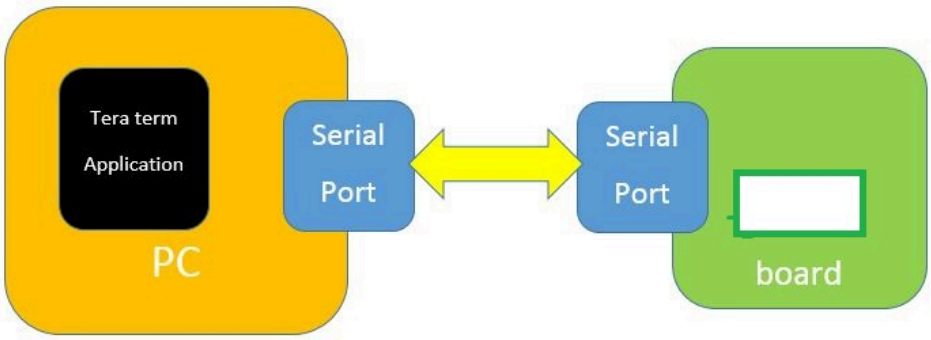
```
uart0: serial@e0000000 {
    compatible = "xlnx,xuartps", "cdns,uart-r1p8";
    clocks = <&clkc 23>, <&clkc 40>;
    clock-names = "uart_clk", "pclk";
    reg = <0xE0000000 0x1000>;
    interrupts = <0 27 4>;
};
```

Test procedure

stty-Linux Commands Related to configuration and Testing:

- stty -all --> command can be used to display the terminal configuration parameters of a tty device. To display all of the active settings on a tty device.
- stty -F /dev/ttyPS0 crtscts /-crtscts) --> Enable/Disable hardware handshaking.
- stty -F /dev/ttyPS0 ixon / -ixon --> Enable/Disable XON/XOFF flow control.
- stty -F /dev/ttyPS0 clocal / -clocal --> Enable/Disable modem control signals such as DTR/DTS and DCD. This is necessary if you are using a "three wire" serial cable because it does not supply these signals.
- stty -F /dev/ttyPS0 cs5 / cs6 / cs7 / cs8 --> Set number of data bits to 5, 6, 7, or 8, respectively.
- stty -F /dev/ttyPS0 parodd / -parodd --> Enable odd parity. Disabling this flag enables even parity.
- stty -F /dev/ttyPS0 parenb / -parenb --> Enable parity checking. When this flag is negated, no parity is used.
- stty -F /dev/ttyPS0 cstopb / -cstopb --> Enable use of two stop bits per character. When this flag is negated, one stop bit per character is used.
- stty -F /dev/ttyPS0 echo / -echo --> Enable/Disable echoing of received characters back to sender.

UART BREAK DETECTION:



- To test the Uart break detection the above setup is required.
- Connect the ep108 serial port with PC serial port with the help of either DB 9 connector or USB to serial cable.
- Form PC side with the help of Tera term application go to following options and issue the break signal.
  - Contol ----> Send break or Alt + B
- With the help of proc file system (proc/tty/driver/xuartps) we can check whether the board serial port is receiving the break signal or not.

```
root@Xilinx-ZynqMP-EAApr2015:~# cat  /proc/tty/driver/xuartps
serinfo:1.0 driver revision:
0: uart:xuartps mmio:0xFF000000 irq:16 tx:21556 rx:401 brk:1 RTS|CTS|DTR|DSR|CD
```

Expected Output

```
root@Xilinx-ZCU102-2015_4:~# cat /proc/tty/driver/xuartps
serinfo:1.0 driver revision:
```

```
0: uart:xuartps mmio:0xFF000000 irq:204 tx:1008 rx:51 RTS|CTS|DTR|DSR|CD
1: uart:xuartps mmio:0xFF010000 irq:205 tx:0 rx:0 CTS|DSR|CD
root@Xilinx-ZCU102-2015_4:~# [   79.344585] sysrq: SysRq : HELP : loglevel(0-9) reboot(b) crash(c) terminate-all-ta

root@Xilinx-ZCU102-2015_4:~# cat /proc/tty/driver/xuartps
serinfo:1.0 driver revision:
0: uart:xuartps mmio:0xFF000000 irq:204 tx:1267 rx:64 brk:1 RTS|CTS|DTR|DSR|CD
1: uart:xuartps mmio:0xFF010000 irq:205 tx:0 rx:0 CTS|DSR|CD
root@Xilinx-ZCU102-2015_4:~# stty -F /dev/ttyPS0 -a
speed 115200 baud;stty: /dev/ttyPS0
 line = 0;
intr = ^C; quit = ^\; erase = ^?; kill = ^U; eof = ^D; eol =
; eol2 = <undef>;
swtch = <undef>; start = ^Q; stop = ^S; susp = ^Z; rprnt = ^R; werase = ^W;
lnext = ^V; flush = ^O; min = 1; time = 0;
-parenb -parodd cs8 hupcl -cstopb cread clocal -crtscts
-ignbrk -brkint -ignpar -parmrk -inpck -istrip -inlcr -igncr icrnl ixon ixoff
-iuclc -ixany -imaxbel -iutf8
opost -olcuc -ocrnl onlcr -onocr -onlret -ofill -ofdel nl0 cr0 tab0 bs0 vt0 ff0
isig icanon -iexten echo echoe echok -echonl -noflsh -xcase -tostop -echoprt
echoctl echoke
root@Xilinx-ZCU102-2015_4:~# stty -F /dev/ttyPS1 -a
speed 9600 baud;stty: /dev/ttyPS1
 line = 0;
intr = ^C; quit = ^\; erase = ^?; kill = ^U; eof = ^D; eol = <undef>;
eol2 = <undef>; swtch = <undef>; start = ^Q; stop = ^S; susp = ^Z; rprnt = ^R;
werase = ^W; lnext = ^V; flush = ^O; min = 1; time = 0;
-parenb -parodd cs8 hupcl -cstopb cread clocal -crtscts
-ignbrk -brkint -ignpar -parmrk -inpck -istrip -inlcr -igncr icrnl ixon -ixoff
-iuclc -ixany -imaxbel -iutf8
opost -olcuc -ocrnl onlcr -onocr -onlret -ofill -ofdel nl0 cr0 tab0 bs0 vt0 ff0
isig icanon iexten echo echoe echok -echonl -noflsh -xcase -tostop -echoprt
echoctl echoke
root@Xilinx-ZCU102-2015_4:~# cat < /dev/ttyPS1
^C
root@Xilinx-ZCU102-2015_4:~# cat /proc/tty/driver/xuartps
serinfo:1.0 driver revision:
0: uart:xuartps mmio:0xFF000000 irq:204 tx:2975 rx:162 brk:1 RTS|CTS|DTR|DSR|CD
1: uart:xuartps mmio:0xFF010000 irq:205 tx:0 rx:1 brk:1 CTS|DSR|CD
root@Xilinx-ZCU102-2015_4:~# stty -F /dev/ttyPS1 inpck
root@Xilinx-ZCU102-2015_4:~# cat < /dev/ttyPS1
^C
root@Xilinx-ZCU102-2015_4:~# cat /proc/tty/driver/xuartps
serinfo:1.0 driver revision:
0: uart:xuartps mmio:0xFF000000 irq:204 tx:3646 rx:200 brk:1 RTS|CTS|DTR|DSR|CD
1: uart:xuartps mmio:0xFF010000 irq:205 tx:0 rx:2 fe:1 brk:1 CTS|DSR|CD
root@Xilinx-ZCU102-2015_4:~# cat < /dev/ttyPS1
^C
root@Xilinx-ZCU102-2015_4:~# cat /proc/tty/driver/xuartps
serinfo:1.0 driver revision:
0: uart:xuartps mmio:0xFF000000 irq:204 tx:4042 rx:215 brk:1 RTS|CTS|DTR|DSR|CD
1: uart:xuartps mmio:0xFF010000 irq:205 tx:0 rx:7 fe:6 brk:1 CTS|DSR|CD
root@Xilinx-ZCU102-2015_4:~# stty -F /dev/ttyPS1 parenb
root@Xilinx-ZCU102-2015_4:~# cat < /dev/ttyPS1
^C
root@Xilinx-ZCU102-2015_4:~# cat /proc/tty/driver/xuartps
serinfo:1.0 driver revision:
0: uart:xuartps mmio:0xFF000000 irq:204 tx:4835 rx:263 brk:1 RTS|CTS|DTR|DSR|CD
1: uart:xuartps mmio:0xFF010000 irq:205 tx:0 rx:8 fe:6 pe:1 brk:1 CTS|DSR|CD
root@Xilinx-ZCU102-2015_4:~# cat < /dev/ttyPS1
^C
root@Xilinx-ZCU102-2015_4:~# cat /proc/tty/driver/xuartps
serinfo:1.0 driver revision:
0: uart:xuartps mmio:0xFF000000 irq:204 tx:5236 rx:278 brk:1 RTS|CTS|DTR|DSR|CD
1: uart:xuartps mmio:0xFF010000 irq:205 tx:0 rx:9 fe:6 pe:1 brk:2 CTS|DSR|CD
root@Xilinx-ZCU102-2015_4:~# cat < /dev/ttyPS1[  363.850712] random: nonblocking pool is initialized

^C
root@Xilinx-ZCU102-2015_4:~# cat /proc/tty/driver/xuartps
serinfo:1.0 driver revision:
```

```
0: uart:xuartps mmio:0xFF000000 irq:204 tx:5637 rx:293 brk:1 RTS|CTS|DTR|DSR|CD
1: uart:xuartps mmio:0xFF010000 irq:205 tx:0 rx:13 fe:6 pe:5 brk:2 CTS|DSR|CD
root@Xilinx-ZCU102-2015_4:~#
```

Mainline status

This driver is currently in sync with mainline kernel driver.

Change Log

2024.1

Summary:

- Use uart_xmit_advance()
- use console_is_registered()
- Make uart_remove_one_port() return void
- Do not check for 0 return after calling
- Relocate cdns_uart_tx_empty to facilitate rs485
- Add rs485 support to uartps driver

Commits:

edc62b1 - serial: xuartps: Use uart_xmit_advance()

4b71a44 - tty: serial: xilinx_uartps: use console_is_registered()

d5b3d02 - serial: Make uart_remove_one_port() return void

2c2d01a - tty: serial: xilinx_uartps: Do not check for 0 return after calling

d647dc5 - tty: serial: uartps: Relocate cdns_uart_tx_empty to facilitate rs485

6379f64 - tty: serial: uartps: Add rs485 support to uartps driver

2023.2

Summary:

- Revert Make the timeout unsigned
- Revert Add check for runtime_get_sync calls

Commits:

dd19c98 - Revert "tty: xilinx_uartps: Make the timeout unsigned"

f2a7265 - Revert "tty: xilinx_uartps: Add check for runtime_get_sync calls"

2023.1

Summary:

- Return early in cdns_uart_handle_tx()
- Cache xmit in cdns_uart_handle_tx()
- Check clk_enable return value
- Update copyright text to correct format
- Initialise the read_status_mask
- Fix the ignore_status
- Prevent writes when the controller is disabled
- Add timeout waiting for loop
- Check the clk_enable return value
- Make ->set_termios() old ktermios const
- Add check for runtime_get_sync calls
- Make the timeout unsigned
- Fix stuck ISR if RX disabled with non-empty FIFO
- Add check for runtime_get_sync calls
- Make the timeout unsigned

Commits:

a28ef75 - serial: xilinx_uartps: return early in cdns_uart_handle_tx()
08814cd - serial: xilinx_uartps: cache xmit in cdns_uart_handle_tx()
ec33b19 - tty: xilinx_uartps: Check clk_enable return value
7bdd444 - tty: xilinx_uartps: Update copyright text to correct format
03a9480 - tty: xilinx_uartps: Initialise the read_status_mask
b8a6c3b - tty: xilinx_uartps: Fix the ignore_status
b369628 - tty: xilinx_uartps: Prevent writes when the controller is disabled
a17fa12 - tty: xilinx_uartps: Add timeout waiting for loop
ec33b19 - tty: xilinx_uartps: Check the clk_enable return value
bec5b81 - serial: Make ->set_termios() old ktermios const
12f014f - tty: xilinx_uartps: Add check for runtime_get_sync calls
588845d - tty: xilinx_uartps: Make the timeout unsigned
ee4ee95 - serial: uartps: Fix stuck ISR if RX disabled with non-empty FIFO
f2a7265 - Revert "tty: xilinx_uartps: Add check for runtime_get_sync calls"

dd19c98 - Revert "tty: xilinx_uartps: Make the timeout unsigned"

2022.2

Summary:

- Add missing mutex_unlock in cdns_get_id()

Commits:
6a9e37e - tty: serial: uartps: add missing mutex_unlock in cdns_get_id()

2022.1

- None

Related Links

- Linux Drivers
- tty/serial/xilinx_uartps.c
- https://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf
- https://www.xilinx.com/support/documentation/user_guides/ug1085-zynq-ultrascale-trm.pdf