

## [PATCH 1/2] Char: nozomi, Lindent the code

---

*Source:* <http://linux.derkeiler.com/Mailing-Lists/Kernel/2006-10/msg02460.html>

---

- *From:* Jiri Slaby <jirislaby@xxxxxxxxxx>
  - *Date:* Sat, 7 Oct 2006 01:53:58 +0200 (CEST)
- 

[on the top of nozomi-use-tty-wakeup]

nozomi, Lindent the code

Use script/Lindent to indent nozomi driver.

Signed-off-by: Jiri Slaby <jirislaby@xxxxxxxxxx>

```
---
commit e7e58c9f0d3ce7bed7f8c4b1921da37d65e3ee8f
tree c0021b53033d27540ed9211a85905ae36ce5668e
parent b19884f570ea41ff9100cc56962e8d6f435e2337
author Jiri Slaby <jirislaby@xxxxxxxxxx> Sat, 07 Oct 2006 01:47:22 +0200
committer Jiri Slaby <xslaby@xxxxxxxxxxxxxxxxxxxxxx> Sat, 07 Oct 2006 01:47:22 +0200
```

```
drivers/char/nozomi.c | 2759 ++++++-----
1 files changed, 1446 insertions(+), 1313 deletions(-)
```

```
diff --git a/drivers/char/nozomi.c b/drivers/char/nozomi.c
index 8d502d2..354a8a6 100644
--- a/drivers/char/nozomi.c
+++ b/drivers/char/nozomi.c
@@ -92,9 +92,8 @@ #include <linux/init.h>
#include <linux/kfifo.h>
#include <asm/uaccess.h>

-
-#define VERSION_STRING DRIVER_DESC " 2.1 (build date: " __DATE__ " " __TIME__ ")"
-
+#define VERSION_STRING DRIVER_DESC " 2.1 (build date: " __DATE__ " " \
+ __TIME__ ")"

/* Macros definitions */

@@ -105,11 +104,10 @@ #define VERSION_STRING DRIVER_DESC " 2.1
#define NOZOMI_DEBUG_LEVEL 0x1

#define P_BUF_SIZE 128
-#define NFO( _err_flag_, args...) \
```

[PATCH 1/2] Char: nozomi, Lindent the code

```
- do{ \
- char t_m_p_[P_BUF_SIZE]; \
- snprintf(t_m_p_, sizeof(t_m_p_), ##args); \
- printk( _err_flag_ "[%d] %s(): %s\n", __LINE__, __FUNCTION__, t_m_p_); \
+#define NFO( _err_flag_, args...) do { \
+ char t_m_p_[P_BUF_SIZE]; \
+ snprintf(t_m_p_, sizeof(t_m_p_), ##args); \
+ printk( _err_flag_ "[%d] %s(): %s\n", __LINE__, __FUNCTION__, t_m_p_); \
} while(0)

#define ERR(args...) NFO( KERN_ERR, ##args)
@@ -123,7 +121,6 @@ #define D6(args...) D_(0x20, ##args)
#define D7(args...) D_(0x40, ##args)
#define D8(args...) D_(0x80, ##args)

-
#ifdef NOZOMI_DEBUG
#define D_(lvl, args...) D(lvl, ##args)
/* Do we need this settable at runtime? */
@@ -135,74 +132,65 @@ #define D_(lvl, args...) D(lvl, ##args)
/* These printouts are always printed */

#else
- static const int nzdebug = 0;
+static const int nzdebug = 0;
#define D_(lvl, args...)
#endif

/* TODO: rewrite to optimize macros... */
-#define SET_FCR(value__) \
- do { \
- writew((value__), (dc->REG_FCR)); \
+#define SET_FCR(value__) do { \
+ writew((value__), (dc->REG_FCR)); \
} while(0)

-#define SET_IER(value__, mask__) \
- do { \
- dc->ier_last_written = (dc->ier_last_written & ~mask__) | (value__ & mask__); \
- writew( dc->ier_last_written, (dc->REG_IER)); \
+#define SET_IER(value__, mask__) do { \
+ dc->ier_last_written = (dc->ier_last_written & ~mask__) | \
+ (value__ & mask__); \
+ writew( dc->ier_last_written, (dc->REG_IER)); \
} while(0)

-#define GET_IER(read_val__) \
- do { \
- (read_val__) = readw((dc->REG_IER)); \
+#define GET_IER(read_val__) do { \
+ (read_val__) = readw((dc->REG_IER)); \
```

[PATCH 1/2] Char: nozomi, Lindent the code

```
} while(0)

-#define GET_IIR(read_val__) \
- do { \
- (read_val__) = readw( (dc->REG_IIR));\
+#define GET_IIR(read_val__) do { \
+ (read_val__) = readw( (dc->REG_IIR)); \
} while(0)

-#define GET_MEM(value__, addr__, length__) \
- do { \
-/* read_mem32( (u32*) (value__), (u32) (addr__), (length__)); */\
- read_mem32( (u32*) (value__), (addr__), (length__));\
+#define GET_MEM(value__, addr__, length__) do { \
+/* read_mem32( (u32*) (value__), (u32) (addr__), (length__)); */\
+ read_mem32( (u32*) (value__), (addr__), (length__)); \
} while(0)

-#define GET_MEM_BUF(value__, addr__, length__) \
- do { \
-/* read_mem32_buf( (u32*) (value__), (u32) (addr__), (length__)); */\
- read_mem32_buf( (u32*) (value__), (addr__), (length__));\
+#define GET_MEM_BUF(value__, addr__, length__) do { \
+/* read_mem32_buf( (u32*) (value__), (u32) (addr__), (length__)); */\
+ read_mem32_buf( (u32*) (value__), (addr__), (length__)); \
} while(0)

-#define SET_MEM(addr__, value__, length__) \
- do { \
- write_mem32( (addr__), (u32*) (value__), (length__));\
+#define SET_MEM(addr__, value__, length__) do { \
+ write_mem32( (addr__), (u32*) (value__), (length__)); \
} while(0)

-#define SET_MEM_BUF(addr__, value__, length__) \
- do { \
- write_mem32_buf( (addr__), (u32*) (value__), (length__));\
+#define SET_MEM_BUF(addr__, value__, length__) do { \
+ write_mem32_buf( (addr__), (u32*) (value__), (length__)); \
} while(0)

-
#define TMP_BUF_MAX 256

-#define DUMP(buf__,len__) \
- do { \
- char tbuf[TMP_BUF_MAX]={0};\
- if (len__>1) {\
- snprintf(tbuf, len__ > TMP_BUF_MAX ? TMP_BUF_MAX : len__, "%s",buf__);\
- if(tbuf[len__-2] == '\r') {\
- tbuf[len__-2] = 'r';\

```

[PATCH 1/2] Char: nozomi, Lindent the code

```
- }\
- D1( "SENDING: '%s' (%d+n)", tbuf, len__);\
- } else {\
- D1( "SENDING: '%s' (%d)", tbuf, len__);\
- }\
+ #define DUMP(buf__,len__) do { \
+ char tbuf[TMP_BUF_MAX]={0}; \
+ if (len__>1) { \
+ snprintf(tbuf, len__ > TMP_BUF_MAX ? TMP_BUF_MAX : len__, "%s",\
+ buf__); \
+ if(tbuf[len__-2] == '\r') { \
+ tbuf[len__-2] = 'r'; \
+ } \
+ D1( "SENDING: '%s' (%d+n)", tbuf, len__); \
+ } else { \
+ D1( "SENDING: '%s' (%d)", tbuf, len__); \
+ } \
} while(0)

#define RELEVANT_IFLAG(iflag) ((iflag) & (IGNBRK|BRKINT|IGNPAR|PARMRK|INPCK))

-
/* Defines */
#define NOZOMI_NAME "nozomi"
#define NOZOMI_NAME_TTY "nozomi_tty"
@@ -229,12 +217,12 @@ #define MSR_RI 0x20
#define MSR_DSR 0x40

/* Define all types of vendors and devices to support */
-#define VENDOR1 0x1931 /* Vendor Option */
-#define DEVICE1 0x000c /* HSDPA card */
+#define VENDOR1 0x1931 /* Vendor Option */
+#define DEVICE1 0x000c /* HSDPA card */

-#define R_IIR 0x0000 /* Interrupt Identity Register */
-#define R_FCR 0x0000 /* Flow Control Register */
-#define R_IER 0x0004 /* Interrupt Enable Register */
+#define R_IIR 0x0000 /* Interrupt Identity Register */
+#define R_FCR 0x0000 /* Flow Control Register */
+#define R_IER 0x0004 /* Interrupt Enable Register */

#define CONFIG_MAGIC 0xEFEFEFEF
#define TOGGLE_VALID 0x0000
@@ -275,7 +263,7 @@ #define NOZOMI_MAX_PORTS 5

/* There are two types of nozomi cards, one with 2048 memory and with 8192 memory */
enum card_type {
- F32_2 = 2048, /* Has 512 bytes downlink and uplink * 2 -> 2048 */
+ F32_2 = 2048, /* Has 512 bytes downlink and uplink * 2 -> 2048 */
F32_8 = 8192, /* Has 3072 bytes downlink and 1024 bytes uplink * 2 -> 8192 */
};
```

```
@@ -287,80 +275,80 @@ enum channel_type {

/* Port definition for the card regarding flow control */
enum ctrl_port_type {
- CTRL_CMD = 0x00,
- CTRL_MDM = 0x01,
- CTRL_DIAG = 0x02,
- CTRL_APP1 = 0x03,
- CTRL_APP2 = 0x04,
- CTRL_ERROR = -1,
+ CTRL_CMD = 0x00,
+ CTRL_MDM = 0x01,
+ CTRL_DIAG = 0x02,
+ CTRL_APP1 = 0x03,
+ CTRL_APP2 = 0x04,
+ CTRL_ERROR = -1,
};

/* Ports that the nozomi has */
enum port_type {
- PORT_MDM = 0,
- PORT_DIAG= 1,
- PORT_APP1= 2,
- PORT_APP2= 3,
- PORT_CTRL= 4,
- PORT_ERROR=-1,
+ PORT_MDM = 0,
+ PORT_DIAG = 1,
+ PORT_APP1 = 2,
+ PORT_APP2 = 3,
+ PORT_CTRL = 4,
+ PORT_ERROR = -1,
};

#ifdef __ARMEB__
/* Big endian */

typedef struct {
- unsigned enabled : 5; /* Toggle fields are valid if enabled is 0, else A-channels
- must always be used. */
- unsigned diag_dl : 1;
- unsigned mdm_dl : 1;
- unsigned mdm_ul : 1;
+ unsigned enabled:5; /* Toggle fields are valid if enabled is 0,
+ else A-channels must always be used. */
+ unsigned diag_dl:1;
+ unsigned mdm_dl:1;
+ unsigned mdm_ul:1;
} __attribute__((packed)) toggles_t;
```

[PATCH 1/2] Char: nozomi, Lindent the code

```
/* Configuration table to read at startup of card */
/* Is for now only needed during initialization phase */
typedef struct {
- u32 signature;
- u16 product_information;
- u16 version;
- u8 pad3[3];
- toggles_t toggle;
- u8 pad1[4];
- u16 dl_mdm_len1; /* If this is 64, it can hold 60 bytes + 4 that is length field */
- u16 dl_start;
-
- u16 dl_diag_len1;
- u16 dl_mdm_len2; /* If this is 64, it can hold 60 bytes + 4 that is length field */
- u16 dl_app1_len;
-
- u16 dl_diag_len2;
- u16 dl_ctrl_len;
- u16 dl_app2_len;
- u8 pad2[16];
- u16 ul_mdm_len1;
- u16 ul_start;
- u16 ul_diag_len;
- u16 ul_mdm_len2;
- u16 ul_app1_len;
- u16 ul_app2_len;
- u16 ul_ctrl_len;
-} __attribute__((packed)) config_table_t;
+ u32 signature;
+ u16 product_information;
+ u16 version;
+ u8 pad3[3];
+ toggles_t toggle;
+ u8 pad1[4];
+ u16 dl_mdm_len1; /* If this is 64, it can hold 60 bytes + 4 that is length field */
+ u16 dl_start;
+
+ u16 dl_diag_len1;
+ u16 dl_mdm_len2; /* If this is 64, it can hold 60 bytes + 4 that is length field */
+ u16 dl_app1_len;
+
+ u16 dl_diag_len2;
+ u16 dl_ctrl_len;
+ u16 dl_app2_len;
+ u8 pad2[16];
+ u16 ul_mdm_len1;
+ u16 ul_start;
+ u16 ul_diag_len;
+ u16 ul_mdm_len2;
+ u16 ul_app1_len;
+ u16 ul_app2_len;
```

[PATCH 1/2] Char: nozomi, Lindent the code

```
+ u16 ul_ctrl_len;
+} __attribute__((packed)) config_table_t;

/* This stores all control downlink flags */
typedef struct {
- u8 port;
- unsigned reserved : 4;
- unsigned CTS : 1;
- unsigned RI : 1;
- unsigned DCD : 1;
- unsigned DSR : 1;
+ u8 port;
+ unsigned reserved:4;
+ unsigned CTS:1;
+ unsigned RI:1;
+ unsigned DCD:1;
+ unsigned DSR:1;
} __attribute__((packed)) ctrl_dl_t;

/* This stores all control uplink flags */
typedef struct {
- u8 port;
- unsigned reserved : 6;
- unsigned RTS : 1;
- unsigned DTR : 1;
+ u8 port;
+ unsigned reserved:6;
+ unsigned RTS:1;
+ unsigned DTR:1;
} __attribute__((packed)) ctrl_ul_t;

#else
@@ -368,227 +356,234 @@ #else

/* This represents the toggle information */
typedef struct {
- unsigned mdm_ul : 1;
- unsigned mdm_dl : 1;
- unsigned diag_dl : 1;
- unsigned enabled : 5; /* Toggle fields are valid if enabled is 0, else A-channels
- must always be used. */
+ unsigned mdm_ul:1;
+ unsigned mdm_dl:1;
+ unsigned diag_dl:1;
+ unsigned enabled:5; /* Toggle fields are valid if enabled is 0,
+ else A-channels must always be used. */
} __attribute__((packed)) toggles_t;

/* Configuration table to read at startup of card */
typedef struct {
- u32 signature;
```

[PATCH 1/2] Char: nozomi, Lindent the code

```
- u16 version;
- u16 product_information;
- toggles_t toggle;
- u8 pad1[7];
- u16 dl_start;
- u16 dl_mdm_len1; /* If this is 64, it can hold 60 bytes + 4 that is length field */
- u16 dl_mdm_len2;
- u16 dl_diag_len1;
- u16 dl_diag_len2;
- u16 dl_app1_len;
- u16 dl_app2_len;
- u16 dl_ctrl_len;
- u8 pad2[16];
- u16 ul_start;
- u16 ul_mdm_len2;
- u16 ul_mdm_len1;
- u16 ul_diag_len;
- u16 ul_app1_len;
- u16 ul_app2_len;
- u16 ul_ctrl_len;
-} __attribute__((packed)) config_table_t;
+ u32 signature;
+ u16 version;
+ u16 product_information;
+ toggles_t toggle;
+ u8 pad1[7];
+ u16 dl_start;
+ u16 dl_mdm_len1; /* If this is 64, it can hold 60 bytes + 4 that is length field */
+ u16 dl_mdm_len2;
+ u16 dl_diag_len1;
+ u16 dl_diag_len2;
+ u16 dl_app1_len;
+ u16 dl_app2_len;
+ u16 dl_ctrl_len;
+ u8 pad2[16];
+ u16 ul_start;
+ u16 ul_mdm_len2;
+ u16 ul_mdm_len1;
+ u16 ul_diag_len;
+ u16 ul_app1_len;
+ u16 ul_app2_len;
+ u16 ul_ctrl_len;
+} __attribute__((packed)) config_table_t;

/* This stores all control downlink flags */
typedef struct {
- unsigned DSR : 1;
- unsigned DCD : 1;
- unsigned RI : 1;
- unsigned CTS : 1;
- unsigned reserved : 4;
```

```
- u8 port;
+ unsigned DSR:1;
+ unsigned DCD:1;
+ unsigned RI:1;
+ unsigned CTS:1;
+ unsigned reserverd:4;
+ u8 port;
} __attribute__((packed)) ctrl_dl_t;

/* This stores all control uplink flags */
typedef struct {
- unsigned DTR : 1;
- unsigned RTS : 1;
- unsigned reserved : 6;
- u8 port;
+ unsigned DTR:1;
+ unsigned RTS:1;
+ unsigned reserved:6;
+ u8 port;
} __attribute__((packed)) ctrl_ul_t;
#endif

/* This holds all information that is needed regarding a port */
struct port {
- u8 update_flow_control;
- ctrl_ul_t ctrl_ul;
- ctrl_dl_t ctrl_dl;
- struct kfifo *fifo_ul;
-// u32 dl_addr[2];
+ u8 update_flow_control;
+ ctrl_ul_t ctrl_ul;
+ ctrl_dl_t ctrl_dl;
+ struct kfifo *fifo_ul;
+// u32 dl_addr[2];
void __iomem *dl_addr[2];
- u32 dl_size[2];
- u8 toggle_dl;
-// u32 ul_addr[2];
+ u32 dl_size[2];
+ u8 toggle_dl;
+// u32 ul_addr[2];
void __iomem *ul_addr[2];
- u32 ul_size[2];
- u8 toggle_ul;
- u16 token_dl;
-
- struct tty_struct *tty;
- int tty_open_count;
- struct semaphore tty_sem;
- wait_queue_head_t tty_wait;
- struct async_icount tty_icount;
```

```
- int tty_index;
- u32 rx_data, tx_data;
- u8 tty_dont_flip;
-
+ u32 ul_size[2];
+ u8 toggle_ul;
+ u16 token_dl;
+
+ struct tty_struct *tty;
+ int tty_open_count;
+ struct semaphore tty_sem;
+ wait_queue_head_t tty_wait;
+ struct async_icount tty_icount;
+ int tty_index;
+ u32 rx_data, tx_data;
+ u8 tty_dont_flip;
};

/* Private data one for each card in the system */
typedef struct {
-// u32 base_addr;
+// u32 base_addr;
void __iomem *base_addr;
- u8 closing;
+ u8 closing;

/* Pointers to registers */
void __iomem *REG_IIR;
void __iomem *REG_FCR;
void __iomem *REG_IER;

- u16 ier_last_written;
- enum card_type card_type;
- config_table_t config_table; /* Configuration table */
- struct pci_dev *pdev;
- struct port port[NOZOMI_MAX_PORTS];
- u8 *send_buf;
+ u16 ier_last_written;
+ enum card_type card_type;
+ config_table_t config_table; /* Configuration table */
+ struct pci_dev *pdev;
+ struct port port[NOZOMI_MAX_PORTS];
+ u8 *send_buf;

- struct tty_driver *tty_driver;
+ struct tty_driver *tty_driver;

- struct workqueue_struct *tty_flip_wq;
- struct work_struct tty_flip_wq_struct;
+ struct workqueue_struct *tty_flip_wq;
+ struct work_struct tty_flip_wq_struct;
```

[PATCH 1/2] Char: nozomi, Lindent the code

```
- struct termios *tty_termios[NTTY_TTY_MINORS];
- struct termios *tty_termios_locked[NTTY_TTY_MINORS];
- spinlock_t spin_mutex;
+ struct termios *tty_termios[NTTY_TTY_MINORS];
+ struct termios *tty_termios_locked[NTTY_TTY_MINORS];
+ spinlock_t spin_mutex;

- u32 open_ttys;
+ u32 open_ttys;
} dc_t;

/* This is a data packet that is read or written to/from card */
struct buffer {
- u32 size; /* size is the length of the data buffer */
- u8 *data;
+ u32 size; /* size is the length of the data buffer */
+ u8 *data;
} __attribute__((packed));

/* Function declarations */
-static int ntty_tty_init(dc_t *dc);
+static int ntty_tty_init(dc_t * dc);
static void tty_flip_queue_function(void *tmp_dc);

/* Global variables */
static struct pci_device_id nozomi_pci_tbl[] = {
{PCI_DEVICE(VENDOR1, DEVICE1)},
- {0, }
+ {0,}
};
+
MODULE_DEVICE_TABLE(pci, nozomi_pci_tbl);

/* Used to store interrupt variables */
typedef struct {
- u16 read_iir; /* Holds current interrupt tokens */
+ u16 read_iir; /* Holds current interrupt tokens */
} irq_t;

/* Representing the pci device of interest */
static int cards_found;
-static dc_t* my_dev = NULL;
+static dc_t *my_dev = NULL;
static irq_t my_irq;

-
-
-static inline dc_t* get_dc_by_pdev(struct pci_dev* pdev) {
- return my_dev;
+static inline dc_t *get_dc_by_pdev(struct pci_dev *pdev)
```

[PATCH 1/2] Char: nozomi, Lindent the code

```
+{
+ return my_dev;
}

-static inline dc_t* get_dc_by_index(s32 index ) {
- return my_dev;
+static inline dc_t *get_dc_by_index(s32 index)
+{
+ return my_dev;
}

-static inline s32 get_index(struct tty_struct *tty) {
- return tty->index;
+static inline s32 get_index(struct tty_struct *tty)
+{
+ return tty->index;
}

-static inline struct port* get_port_by_tty(struct tty_struct *tty) {
- return &my_dev->port[ get_index(tty) ];
+static inline struct port *get_port_by_tty(struct tty_struct *tty)
+{
+ return &my_dev->port[get_index(tty)];
}

-static inline dc_t* get_dc_by_tty(struct tty_struct *tty ) {
- return my_dev;
+static inline dc_t *get_dc_by_tty(struct tty_struct *tty)
+{
+ return my_dev;
}

-
-/* TODO: */
-/* -Optimize */
-/* -Rewrite cleaner */
-//static void read_mem32(u32 *buf, u32 mem_addr_start, u32 size_bytes) {
-void read_mem32(u32 *buf, void __iomem *mem_addr_start, u32 size_bytes) {
- u32 i = 0;
- u32* ptr = (__force u32 *)mem_addr_start;
- u16* buf16;
-
- /* 2 bytes */
- if (size_bytes == 2) {
- buf16 = (u16*) buf;
- *buf16 = readw((void __iomem *)ptr);
- return;
- }
-
- while (i < size_bytes) {
- if ( size_bytes - i == 2) {
```

[PATCH 1/2] Char: nozomi, Lindent the code

```
- /* Handle 2 bytes in the end */
- buf16 = (u16*) buf;
- *(buf16) = readw((void __iomem *)ptr);
- i+=2;
- } else {
- /* Read 4 bytes */
- *(buf) = readl((void __iomem *)ptr);
- i+=4;
- }
- buf++; ptr++;
- }
+static void read_mem32(u32 * buf, void __iomem * mem_addr_start, u32 size_bytes)
+{
+ u32 i = 0;
+ u32 *ptr = (__force u32 *) mem_addr_start;
+ u16 *buf16;
+
+ /* 2 bytes */
+ if (size_bytes == 2) {
+ buf16 = (u16 *) buf;
+ *buf16 = readw((void __iomem *)ptr);
+ return;
+ }
+
+ while (i < size_bytes) {
+ if (size_bytes - i == 2) {
+ /* Handle 2 bytes in the end */
+ buf16 = (u16 *) buf;
+ *(buf16) = readw((void __iomem *)ptr);
+ i += 2;
+ } else {
+ /* Read 4 bytes */
+ *(buf) = readl((void __iomem *)ptr);
+ i += 4;
+ }
+ buf++;
+ ptr++;
+ }
+
+ /* TODO: */
+ /* - Rewrite cleaner */
+ /* - merge with read_mem32() */
+ //static void read_mem32_buf(u32 *buf, u32 mem_addr_start, u32 size_bytes) {
+ -static void read_mem32_buf(u32 *buf, void __iomem * mem_addr_start, u32 size_bytes) {
+ +static void read_mem32_buf(u32 * buf, void __iomem * mem_addr_start,
+ + u32 size_bytes)
+ +{
+ #ifdef __ARMEB__
+ - u32 i = 0;
+ - u32* ptr = (u32*) mem_addr_start;
```

[PATCH 1/2] Char: nozomi, Lindent the code

```
- u16* buf16;
-
- /* 2 bytes */
- if (size_bytes == 2) {
- buf16 = (u16*) buf;
- *buf16 = __le16_to_cpu( readw( ptr ));
- return;
- }
-
- while (i < size_bytes) {
- if ( size_bytes - i == 2) {
- /* Handle 2 bytes in the end */
- buf16 = (u16*) buf;
- *(buf16) = __le16_to_cpu( readw( ptr ));
- i+=2;
- } else {
- /* Read 4 bytes */
- *(buf) = __le32_to_cpu( readl( ptr ));
- i+=4;
- }
- buf++; ptr++;
- }
+ u32 i = 0;
+ u32 *ptr = (u32 *) mem_addr_start;
+ u16 *buf16;
+
+ /* 2 bytes */
+ if (size_bytes == 2) {
+ buf16 = (u16 *) buf;
+ *buf16 = __le16_to_cpu(readw(ptr));
+ return;
+ }
+
+ while (i < size_bytes) {
+ if (size_bytes - i == 2) {
+ /* Handle 2 bytes in the end */
+ buf16 = (u16 *) buf;
+ *(buf16) = __le16_to_cpu(readw(ptr));
+ i += 2;
+ } else {
+ /* Read 4 bytes */
+ *(buf) = __le32_to_cpu(readl(ptr));
+ i += 4;
+ }
+ buf++;
+ ptr++;
+ }
#else
- read_mem32(buf, mem_addr_start, size_bytes);
+ read_mem32(buf, mem_addr_start, size_bytes);
#endif
```

[PATCH 1/2] Char: nozomi, Lindent the code

```

}

@@ -596,242 +591,304 @@ #endif
/* -Optimize */
/* -Rewrite cleaner */
//static u32 write_mem32(u32 mem_addr_start, u32 *buf, u32 size_bytes) {
-static u32 write_mem32(void __iomem *mem_addr_start, u32 *buf, u32 size_bytes) {
- u32 i = 0;
- u32* ptr = (__force u32*) mem_addr_start;
- u16* buf16;
-
- /* 2 bytes */
- if (size_bytes == 2) {
- buf16 = (u16*) buf;
- writew( *buf16, (void __iomem *)ptr);
- return 2;
- }
-
- while (i < size_bytes) {
- if ( size_bytes - i == 2) {
- /* 2 bytes */
- buf16 = (u16*) buf;
- writew( *buf16, (void __iomem *)ptr);
- i+=2;
- } else {
- /* 4 bytes */
- writel( *buf, (void __iomem *)ptr );
- i += 4;
- }
- buf++; ptr++;
- }
- return size_bytes;
+static u32 write_mem32(void __iomem * mem_addr_start, u32 * buf, u32 size_bytes)
+{
+ u32 i = 0;
+ u32 *ptr = (__force u32 *) mem_addr_start;
+ u16 *buf16;
+
+ /* 2 bytes */
+ if (size_bytes == 2) {
+ buf16 = (u16 *) buf;
+ writew(*buf16, (void __iomem *)ptr);
+ return 2;
+ }
+
+ while (i < size_bytes) {
+ if (size_bytes - i == 2) {
+ /* 2 bytes */
+ buf16 = (u16 *) buf;
+ writew(*buf16, (void __iomem *)ptr);
+ i += 2;

```

```

+ } else {
+ /* 4 bytes */
+ writel(*buf, (void __iomem *)ptr);
+ i += 4;
+ }
+ buf++;
+ ptr++;
+ }
+ return size_bytes;
}

/* Todo: */
/* - Merge with write_mem32() */
//static u32 write_mem32_buf(u32 mem_addr_start, u32 *buf, u32 size_bytes) {
-static u32 write_mem32_buf(void __iomem *mem_addr_start, u32 *buf, u32 size_bytes) {
+static u32 write_mem32_buf(void __iomem * mem_addr_start, u32 * buf,
+ u32 size_bytes)
+{
#ifdef __ARMEB__
- u32 i = 0;
- u32* ptr = (u32*) mem_addr_start;
- u16* buf16;
-
- /* 2 bytes */
- if (size_bytes == 2) {
- buf16 = (u16*) buf;
- writew( __le16_to_cpu(*buf16), ptr);
- return 2;
- }
-
- while (i < size_bytes) {
- if ( size_bytes - i == 2) {
- /* 2 bytes */
- buf16 = (u16*) buf;
- writew( __le16_to_cpu(*buf16), ptr);
- i+=2;
- } else {
- /* 4 bytes */
- writel( __cpu_to_le32( *buf ), ptr);
- i += 4;
- }
- buf++; ptr++;
- }
- return size_bytes;
+ u32 i = 0;
+ u32 *ptr = (u32 *) mem_addr_start;
+ u16 *buf16;
+
+ /* 2 bytes */
+ if (size_bytes == 2) {
+ buf16 = (u16 *) buf;

```

[PATCH 1/2] Char: nozomi, Lindent the code

```
+ writew(__le16_to_cpu(*buf16), ptr);
+ return 2;
+ }
+
+ while (i < size_bytes) {
+ if (size_bytes - i == 2) {
+ /* 2 bytes */
+ buf16 = (u16 *) buf;
+ writew(__le16_to_cpu(*buf16), ptr);
+ i += 2;
+ } else {
+ /* 4 bytes */
+ writel(__cpu_to_le32(*buf), ptr);
+ i += 4;
+ }
+ buf++;
+ ptr++;
+ }
+ return size_bytes;
#else
- return write_mem32(mem_addr_start, buf, size_bytes);
+ return write_mem32(mem_addr_start, buf, size_bytes);
#endif
}

/* Setup pointers to different channels and also setup buffer sizes. */
-static void setup_memory(dc_t *dc)
+static void setup_memory(dc_t * dc)
{
- void __iomem *offset = dc->base_addr + dc->config_table.dl_start;
- /* The length reported is including the length field of 4 bytes, hence subtract with 4. */
- u16 buff_offset = 4;
-
- /* Modem port dl configuration */
- dc->port[PORT_MDM].dl_addr[CH_A] = offset;
- dc->port[PORT_MDM].dl_addr[CH_B] = (offset += dc->config_table.dl_mdm_len1);
- dc->port[PORT_MDM].dl_size[CH_A] = dc->config_table.dl_mdm_len1 - buff_offset;
- dc->port[PORT_MDM].dl_size[CH_B] = dc->config_table.dl_mdm_len2 - buff_offset;
-
- /* Diag port dl configuration */
- dc->port[PORT_DIAG].dl_addr[CH_A] = (offset += dc->config_table.dl_mdm_len2);
- dc->port[PORT_DIAG].dl_size[CH_A] = dc->config_table.dl_diag_len1 - buff_offset;
- dc->port[PORT_DIAG].dl_addr[CH_B] = (offset += dc->config_table.dl_diag_len1);
- dc->port[PORT_DIAG].dl_size[CH_B] = dc->config_table.dl_diag_len2 - buff_offset;
-
- /* App1 port dl configuration */
- dc->port[PORT_APP1].dl_addr[CH_A] = (offset += dc->config_table.dl_diag_len2);
- dc->port[PORT_APP1].dl_size[CH_A] = dc->config_table.dl_app1_len - buff_offset;
-
- /* App2 port dl configuration */
- dc->port[PORT_APP2].dl_addr[CH_A] = (offset += dc->config_table.dl_app1_len);
```

[PATCH 1/2] Char: nozomi, Lindent the code

```
- dc->port[PORT_APP2].dl_size[CH_A] = dc->config_table.dl_app2_len - buff_offset;
-
- /* Ctrl dl configuration */
- dc->port[PORT_CTRL].dl_addr[CH_A] = (offset += dc->config_table.dl_app2_len);
- dc->port[PORT_CTRL].dl_size[CH_A] = dc->config_table.dl_ctrl_len - buff_offset;
-
-
- /* Modem Port ul configuration */
- dc->port[PORT_MDM].ul_addr[CH_A] = (offset = dc->base_addr + dc->config_table.ul_start);
- dc->port[PORT_MDM].ul_size[CH_A] = dc->config_table.ul_mdm_len1 - buff_offset;
- dc->port[PORT_MDM].ul_addr[CH_B] = (offset += dc->config_table.ul_mdm_len1);
- dc->port[PORT_MDM].ul_size[CH_B] = dc->config_table.ul_mdm_len2 - buff_offset;
-
- /* Diag port ul configuration */
- dc->port[PORT_DIAG].ul_addr[CH_A] = (offset += dc->config_table.ul_mdm_len2);
- dc->port[PORT_DIAG].ul_size[CH_A] = dc->config_table.ul_diag_len - buff_offset;
-
- /* App1 port ul configuration */
- dc->port[PORT_APP1].ul_addr[CH_A] = (offset += dc->config_table.ul_diag_len);
- dc->port[PORT_APP1].ul_size[CH_A] = dc->config_table.ul_app1_len - buff_offset;
-
- /* App2 port ul configuration */
- dc->port[PORT_APP2].ul_addr[CH_A] = (offset += dc->config_table.ul_app1_len);
- dc->port[PORT_APP2].ul_size[CH_A] = dc->config_table.ul_app2_len - buff_offset;
-
- /* Ctrl ul configuration */
- dc->port[PORT_CTRL].ul_addr[CH_A] = (offset += dc->config_table.ul_app2_len);
- dc->port[PORT_CTRL].ul_size[CH_A] = dc->config_table.ul_ctrl_len - buff_offset;
+ void __iomem *offset = dc->base_addr + dc->config_table.dl_start;
+ /* The length reported is including the length field of 4 bytes, hence
+ subtract with 4. */
+ u16 buff_offset = 4;
+
+ /* Modem port dl configuration */
+ dc->port[PORT_MDM].dl_addr[CH_A] = offset;
+ dc->port[PORT_MDM].dl_addr[CH_B] = (offset +=
+ dc->config_table.dl_mdm_len1);
+ dc->port[PORT_MDM].dl_size[CH_A] =
+ dc->config_table.dl_mdm_len1 - buff_offset;
+ dc->port[PORT_MDM].dl_size[CH_B] =
+ dc->config_table.dl_mdm_len2 - buff_offset;
+
+ /* Diag port dl configuration */
+ dc->port[PORT_DIAG].dl_addr[CH_A] = (offset +=
+ dc->config_table.dl_mdm_len2);
+ dc->port[PORT_DIAG].dl_size[CH_A] =
+ dc->config_table.dl_diag_len1 - buff_offset;
+ dc->port[PORT_DIAG].dl_addr[CH_B] = (offset +=
+ dc->config_table.dl_diag_len1);
+ dc->port[PORT_DIAG].dl_size[CH_B] =
+ dc->config_table.dl_diag_len2 - buff_offset;
```

```
+
+ /* App1 port dl configuration */
+ dc->port[PORT_APP1].dl_addr[CH_A] = (offset +=
+ dc->config_table.dl_diag_len2);
+ dc->port[PORT_APP1].dl_size[CH_A] =
+ dc->config_table.dl_app1_len - buff_offset;
+
+ /* App2 port dl configuration */
+ dc->port[PORT_APP2].dl_addr[CH_A] = (offset +=
+ dc->config_table.dl_app1_len);
+ dc->port[PORT_APP2].dl_size[CH_A] =
+ dc->config_table.dl_app2_len - buff_offset;
+
+ /* Ctrl dl configuration */
+ dc->port[PORT_CTRL].dl_addr[CH_A] = (offset +=
+ dc->config_table.dl_app2_len);
+ dc->port[PORT_CTRL].dl_size[CH_A] =
+ dc->config_table.dl_ctrl_len - buff_offset;
+
+ /* Modem Port ul configuration */
+ dc->port[PORT_MDM].ul_addr[CH_A] = (offset =
+ dc->base_addr +
+ dc->config_table.ul_start);
+ dc->port[PORT_MDM].ul_size[CH_A] =
+ dc->config_table.ul_mdm_len1 - buff_offset;
+ dc->port[PORT_MDM].ul_addr[CH_B] = (offset +=
+ dc->config_table.ul_mdm_len1);
+ dc->port[PORT_MDM].ul_size[CH_B] =
+ dc->config_table.ul_mdm_len2 - buff_offset;
+
+ /* Diag port ul configuration */
+ dc->port[PORT_DIAG].ul_addr[CH_A] = (offset +=
+ dc->config_table.ul_mdm_len2);
+ dc->port[PORT_DIAG].ul_size[CH_A] =
+ dc->config_table.ul_diag_len - buff_offset;
+
+ /* App1 port ul configuration */
+ dc->port[PORT_APP1].ul_addr[CH_A] = (offset +=
+ dc->config_table.ul_diag_len);
+ dc->port[PORT_APP1].ul_size[CH_A] =
+ dc->config_table.ul_app1_len - buff_offset;
+
+ /* App2 port ul configuration */
+ dc->port[PORT_APP2].ul_addr[CH_A] = (offset +=
+ dc->config_table.ul_app1_len);
+ dc->port[PORT_APP2].ul_size[CH_A] =
+ dc->config_table.ul_app2_len - buff_offset;
+
+ /* Ctrl ul configuration */
+ dc->port[PORT_CTRL].ul_addr[CH_A] = (offset +=
+ dc->config_table.ul_app2_len);
```

[PATCH 1/2] Char: nozomi, Lindent the code

```
+ dc->port[PORT_CTRL].ul_size[CH_A] =
+ dc->config_table.ul_ctrl_len - buff_offset;
// offset = dc->config_table.ul_start;
}

/* Dump config table under initialization phase */
#ifdef NOZOMI_DEBUG
-static void dump_table(dc_t *dc) {
- D3("signature: 0x%08X", dc->config_table.signature);
- D3("version: 0x%04X", dc->config_table.version);
- D3("product_information: 0x%04X", dc->config_table.product_information);
- D3("toggle enabled: %d", dc->config_table.toggle.enabled);
- D3("toggle up_mdm: %d", dc->config_table.toggle.mdm_ul);
- D3("toggle dl_mdm: %d", dc->config_table.toggle.mdm_dl);
- D3("toggle dl_dbg: %d", dc->config_table.toggle.diag_dl);
-
- D3("dl_start: 0x%04X", dc->config_table.dl_start);
- D3("dl_mdm_len0: 0x%04X, %d", dc->config_table.dl_mdm_len1, dc->config_table.dl_mdm_len1);
- D3("dl_mdm_len1: 0x%04X, %d", dc->config_table.dl_mdm_len2, dc->config_table.dl_mdm_len2);
- D3("dl_diag_len0: 0x%04X, %d", dc->config_table.dl_diag_len1, dc->config_table.dl_diag_len1);
- D3("dl_diag_len1: 0x%04X, %d", dc->config_table.dl_diag_len2, dc->config_table.dl_diag_len2);
- D3("dl_app1_len: 0x%04X, %d", dc->config_table.dl_app1_len, dc->config_table.dl_app1_len);
- D3("dl_app2_len: 0x%04X, %d", dc->config_table.dl_app2_len, dc->config_table.dl_app2_len);
- D3("dl_ctrl_len: 0x%04X, %d", dc->config_table.dl_ctrl_len, dc->config_table.dl_ctrl_len);
- D3("ul_start: 0x%04X, %d", dc->config_table.ul_start, dc->config_table.ul_start);
- D3("ul_mdm_len[0]: 0x%04X, %d", dc->config_table.ul_mdm_len1, dc->config_table.ul_mdm_len1);
- D3("ul_mdm_len[1]: 0x%04X, %d", dc->config_table.ul_mdm_len2, dc->config_table.ul_mdm_len2);
- D3("ul_diag_len: 0x%04X, %d", dc->config_table.ul_diag_len, dc->config_table.ul_diag_len);
- D3("ul_app1_len: 0x%04X, %d", dc->config_table.ul_app1_len, dc->config_table.ul_app1_len);
- D3("ul_app2_len: 0x%04X, %d", dc->config_table.ul_app2_len, dc->config_table.ul_app2_len);
- D3("ul_ctrl_len: 0x%04X, %d", dc->config_table.ul_ctrl_len, dc->config_table.ul_ctrl_len);
+static void dump_table(dc_t * dc)
+{
+ D3("signature: 0x%08X", dc->config_table.signature);
+ D3("version: 0x%04X", dc->config_table.version);
+ D3("product_information: 0x%04X", dc->config_table.product_information);
+ D3("toggle enabled: %d", dc->config_table.toggle.enabled);
+ D3("toggle up_mdm: %d", dc->config_table.toggle.mdm_ul);
+ D3("toggle dl_mdm: %d", dc->config_table.toggle.mdm_dl);
+ D3("toggle dl_dbg: %d", dc->config_table.toggle.diag_dl);
+
+ D3("dl_start: 0x%04X", dc->config_table.dl_start);
+ D3("dl_mdm_len0: 0x%04X, %d", dc->config_table.dl_mdm_len1,
+ dc->config_table.dl_mdm_len1);
+ D3("dl_mdm_len1: 0x%04X, %d", dc->config_table.dl_mdm_len2,
+ dc->config_table.dl_mdm_len2);
+ D3("dl_diag_len0: 0x%04X, %d", dc->config_table.dl_diag_len1,
+ dc->config_table.dl_diag_len1);
+ D3("dl_diag_len1: 0x%04X, %d", dc->config_table.dl_diag_len2,
+ dc->config_table.dl_diag_len2);
+ D3("dl_app1_len: 0x%04X, %d", dc->config_table.dl_app1_len,
+ dc->config_table.dl_app1_len);
+ D3("dl_app2_len: 0x%04X, %d", dc->config_table.dl_app2_len,
+ dc->config_table.dl_app2_len);
+ D3("dl_ctrl_len: 0x%04X, %d", dc->config_table.dl_ctrl_len,
+ dc->config_table.dl_ctrl_len);
+
+ D3("ul_start: 0x%04X, %d", dc->config_table.ul_start,
+ dc->config_table.ul_start);
+ D3("ul_mdm_len[0]: 0x%04X, %d", dc->config_table.ul_mdm_len1,
+ dc->config_table.ul_mdm_len1);
+ D3("ul_mdm_len[1]: 0x%04X, %d", dc->config_table.ul_mdm_len2,
+ dc->config_table.ul_mdm_len2);
+ D3("ul_diag_len: 0x%04X, %d", dc->config_table.ul_diag_len,
+ dc->config_table.ul_diag_len);
+ D3("ul_app1_len: 0x%04X, %d", dc->config_table.ul_app1_len,
+ dc->config_table.ul_app1_len);
+ D3("ul_app2_len: 0x%04X, %d", dc->config_table.ul_app2_len,
+ dc->config_table.ul_app2_len);
+ D3("ul_ctrl_len: 0x%04X, %d", dc->config_table.ul_ctrl_len,
+ dc->config_table.ul_ctrl_len);
}
```

[PATCH 1/2] Char: nozomi, Lindent the code

```
+ dc->config_table.dl_app1_len);
+ D3("dl_app2_len: 0x%04X, %d", dc->config_table.dl_app2_len,
+ dc->config_table.dl_app2_len);
+ D3("dl_ctrl_len: 0x%04X, %d", dc->config_table.dl_ctrl_len,
+ dc->config_table.dl_ctrl_len);
+ D3("ul_start: 0x%04X, %d", dc->config_table.ul_start,
+ dc->config_table.ul_start);
+ D3("ul_mdm_len[0]: 0x%04X, %d", dc->config_table.ul_mdm_len1,
+ dc->config_table.ul_mdm_len1);
+ D3("ul_mdm_len[1]: 0x%04X, %d", dc->config_table.ul_mdm_len2,
+ dc->config_table.ul_mdm_len2);
+ D3("ul_diag_len: 0x%04X, %d", dc->config_table.ul_diag_len,
+ dc->config_table.ul_diag_len);
+ D3("ul_app1_len: 0x%04X, %d", dc->config_table.ul_app1_len,
+ dc->config_table.ul_app1_len);
+ D3("ul_app2_len: 0x%04X, %d", dc->config_table.ul_app2_len,
+ dc->config_table.ul_app2_len);
+ D3("ul_ctrl_len: 0x%04X, %d", dc->config_table.ul_ctrl_len,
+ dc->config_table.ul_ctrl_len);
+ }
#endif

/* Read configuration table from card under intalization phase */
/* Returns 1 if ok, else 0 */
-static int nozomi_read_config_table(dc_t *dc) {
+static int nozomi_read_config_table(dc_t * dc)
+{

- GET_MEM( &dc->config_table, dc->base_addr + 0, sizeof(config_table_t));
+ GET_MEM(&dc->config_table, dc->base_addr + 0, sizeof(config_table_t));

- /* D1( "0x%08X == 0x%08X ", dc->config_table.signature, CONFIG_MAGIC); */
+/* D1( "0x%08X == 0x%08X ", dc->config_table.signature, CONFIG_MAGIC); */

- if (dc->config_table.signature != CONFIG_MAGIC ) {
+ if (dc->config_table.signature != CONFIG_MAGIC) {
dev_err(&dc->pdev->dev, "ConfigTable Bad! 0x%08X != 0x%08X\n",
dc->config_table.signature, CONFIG_MAGIC);
return 0;
}

- if( (dc->config_table.version == 0) || (dc->config_table.toggle.enabled == TOGGLE_VALID) ) {
- int i;
- D1( "Second phase, configuring card");
+ if ((dc->config_table.version == 0)
+ || (dc->config_table.toggle.enabled == TOGGLE_VALID)) {
+ int i;
+ D1("Second phase, configuring card");

- setup_memory(dc);
+ setup_memory(dc);
```

[PATCH 1/2] Char: nozomi, Lindent the code

```
- dc->port[PORT_MDM].toggle_ul = dc->config_table.toggle.mdm_ul;
- dc->port[PORT_MDM].toggle_dl = dc->config_table.toggle.mdm_dl;
- dc->port[PORT_DIAG].toggle_dl = dc->config_table.toggle.diag_dl;
- D1("toggle ports: MDM UL:%d MDM DL:%d, DIAG DL:%d",
- dc->port[PORT_MDM].toggle_ul,
- dc->port[PORT_MDM].toggle_dl,
- dc->port[PORT_DIAG].toggle_dl);
+ dc->port[PORT_MDM].toggle_ul = dc->config_table.toggle.mdm_ul;
+ dc->port[PORT_MDM].toggle_dl = dc->config_table.toggle.mdm_dl;
+ dc->port[PORT_DIAG].toggle_dl = dc->config_table.toggle.diag_dl;
+ D1("toggle ports: MDM UL:%d MDM DL:%d, DIAG DL:%d",
+ dc->port[PORT_MDM].toggle_ul,
+ dc->port[PORT_MDM].toggle_dl, dc->port[PORT_DIAG].toggle_dl);

#ifdef NOZOMI_DEBUG
- dump_table(dc);
+ dump_table(dc);
#endif
- for (i=PORT_MDM; i< MAX_PORT;i++) {
- dc->port[i].fifo_ul = kfifo_alloc( FIFO_BUFFER_SIZE_UL, GFP_ATOMIC , NULL);
- memset( &dc->port[i].ctrl_dl, 0, sizeof (ctrl_dl_t));
- memset( &dc->port[i].ctrl_ul, 0, sizeof (ctrl_ul_t));
- }
-
- /* Enable control channel */
- SET_IER( CTRL_DL, CTRL_DL );
+ for (i = PORT_MDM; i < MAX_PORT; i++) {
+ dc->port[i].fifo_ul =
+ kfifo_alloc(FIFO_BUFFER_SIZE_UL, GFP_ATOMIC, NULL);
+ memset(&dc->port[i].ctrl_dl, 0, sizeof(ctrl_dl_t));
+ memset(&dc->port[i].ctrl_ul, 0, sizeof(ctrl_ul_t));
+ }
+
+ /* Enable control channel */
+ SET_IER(CTRL_DL, CTRL_DL);

- dev_info(&dc->pdev->dev, "Initialization OK!\n");
- return 1;
- }
+ dev_info(&dc->pdev->dev, "Initialization OK!\n");
+ return 1;
+ }

- if( (dc->config_table.version > 0) && (dc->config_table.toggle.enabled != TOGGLE_VALID) ) {
- u32 offset = 0;
- D1("First phase: pushing upload buffers, clearing download");
+ if ((dc->config_table.version > 0)
+ && (dc->config_table.toggle.enabled != TOGGLE_VALID)) {
+ u32 offset = 0;
+ D1("First phase: pushing upload buffers, clearing download");
```

[PATCH 1/2] Char: nozomi, Lindent the code

```
- dev_info(&dc->pdev->dev, "Version of card: %d\n", dc->config_table.version);
+ dev_info(&dc->pdev->dev, "Version of card: %d\n",
+ dc->config_table.version);

- /* Here we should disable all I/O over F32. */
- setup_memory(dc);
+ /* Here we should disable all I/O over F32. */
+ setup_memory(dc);

- /* We should send ALL channel pair tokens back along with reset token */
+ /* We should send ALL channel pair tokens back along with reset
+ token */

- /* push upload modem buffers */
- SET_MEM( dc->port[PORT_MDM].ul_addr[CH_A], &offset, 4);
- SET_MEM( dc->port[PORT_MDM].ul_addr[CH_B], &offset, 4);
+ /* push upload modem buffers */
+ SET_MEM(dc->port[PORT_MDM].ul_addr[CH_A], &offset, 4);
+ SET_MEM(dc->port[PORT_MDM].ul_addr[CH_B], &offset, 4);

- SET_FCR( MDM_UL | DIAG_DL | MDM_DL );
+ SET_FCR(MDM_UL | DIAG_DL | MDM_DL);

- D1( "First phase done");
- }
+ D1("First phase done");
+ }

- return 1;
+ return 1;
}

/* Enable uplink interrupts */
-static void enable_transmit_ul( enum port_type port , dc_t *dc ) {
-
- switch( port ) {
- case PORT_MDM: SET_IER( MDM_UL , MDM_UL ); break;
- case PORT_DIAG: SET_IER( DIAG_UL, DIAG_UL ); break;
- case PORT_APP1: SET_IER( APP1_UL, APP1_UL ); break;
- case PORT_APP2: SET_IER( APP2_UL, APP2_UL ); break;
- case PORT_CTRL: SET_IER( CTRL_UL, CTRL_UL ); break;
- default:
- dev_err(&dc->pdev->dev, "Called with wrong port?\n");
- break;
- };
+static void enable_transmit_ul(enum port_type port, dc_t * dc)
+{
+
+ switch (port) {
+ case PORT_MDM:
```

```
+ SET_IER(MDM_UL, MDM_UL);
+ break;
+ case PORT_DIAG:
+ SET_IER(DIAG_UL, DIAG_UL);
+ break;
+ case PORT_APP1:
+ SET_IER(APP1_UL, APP1_UL);
+ break;
+ case PORT_APP2:
+ SET_IER(APP2_UL, APP2_UL);
+ break;
+ case PORT_CTRL:
+ SET_IER(CTRL_UL, CTRL_UL);
+ break;
+ default:
+ dev_err(&dc->pdev->dev, "Called with wrong port?\n");
+ break;
+ };
}

/* Disable uplink interrupts */
-static void disable_transmit_ul(enum port_type port, dc_t *dc)
+static void disable_transmit_ul(enum port_type port, dc_t * dc)
{
switch (port) {
case PORT_MDM:
- SET_IER(0, MDM_UL);
+ SET_IER(0, MDM_UL);
break;
case PORT_DIAG:
SET_IER(0, DIAG_UL);
@@ -852,9 +909,9 @@ static void disable_transmit_ul(enum por
}

/* Enable downlink interrupts */
-static void enable_transmit_dl(enum port_type port, dc_t *dc)
+static void enable_transmit_dl(enum port_type port, dc_t * dc)
{
- switch(port) {
+ switch (port) {
case PORT_MDM:
SET_IER(MDM_DL, MDM_DL);
break;
@@ -877,14 +934,24 @@ static void enable_transmit_dl(enum port
}

/* Disable downlink interrupts */
-static void disable_transmit_dl(enum port_type port, dc_t *dc)
+static void disable_transmit_dl(enum port_type port, dc_t * dc)
{
switch (port) {
```

[PATCH 1/2] Char: nozomi, Lindent the code

```
- case PORT_MDM: SET_IER( 0 ,MDM_DL ); break;
- case PORT_DIAG: SET_IER( 0, DIAG_DL ); break;
- case PORT_APP1: SET_IER( 0, APP1_DL ); break;
- case PORT_APP2: SET_IER( 0, APP2_DL ); break;
- case PORT_CTRL: SET_IER( 0, CTRL_DL ); break;
+ case PORT_MDM:
+ SET_IER(0, MDM_DL);
+ break;
+ case PORT_DIAG:
+ SET_IER(0, DIAG_DL);
+ break;
+ case PORT_APP1:
+ SET_IER(0, APP1_DL);
+ break;
+ case PORT_APP2:
+ SET_IER(0, APP2_DL);
+ break;
+ case PORT_CTRL:
+ SET_IER(0, CTRL_DL);
+ break;
default:
dev_err(&dc->pdev->dev, "Called with wrong port?\n");
break;
@@ -893,203 +960,224 @@ static void disable_transmit_dl(enum por

/* Return 1 - send buffer to card and ack. */
/* Return 0 - don't ack, don't send buffer to card. */
-static int send_data( enum port_type index, dc_t *dc ) {
- u32 size = 0;
- struct port *port = &dc->port[index];
- u8 toggle = port->toggle_ul;
- void __iomem *addr = port->ul_addr[toggle];
- u32 ul_size = port->ul_size[toggle];
- struct tty_struct *tty = port->tty;
-
- if (index >= NTTY_TTY_MINORS) {
- dev_err(&dc->pdev->dev, "Called with wrong index?\n");
- return 0;
- }
+static int send_data(enum port_type index, dc_t * dc)
+{
+ u32 size = 0;
+ struct port *port = &dc->port[index];
+ u8 toggle = port->toggle_ul;
+ void __iomem *addr = port->ul_addr[toggle];
+ u32 ul_size = port->ul_size[toggle];
+ struct tty_struct *tty = port->tty;
+
+ if (index >= NTTY_TTY_MINORS) {
+ dev_err(&dc->pdev->dev, "Called with wrong index?\n");
+ return 0;

```

[PATCH 1/2] Char: nozomi, Lindent the code

```
+ }

- /* Get data from tty and place in buf for now */
- size = __kfifo_get( port->fifo_ul, dc->send_buf, ul_size < SEND_BUF_MAX ? ul_size :
SEND_BUF_MAX );
+ /* Get data from tty and place in buf for now */
+ size =
+ __kfifo_get(port->fifo_ul, dc->send_buf,
+ ul_size < SEND_BUF_MAX ? ul_size : SEND_BUF_MAX);

- if (size == 0) {
- D4("No more data to send, disable link:");
- return 0;
- }
+ if (size == 0) {
+ D4("No more data to send, disable link:");
+ return 0;
+ }

- port->tx_data += size;
+ port->tx_data += size;

- /* DUMP(buf, size); */
+ /* DUMP(buf, size); */

- /* Write length + data */
- SET_MEM( addr, &size, 4 );
- SET_MEM_BUF( addr + 4, dc->send_buf, size);
+ /* Write length + data */
+ SET_MEM(addr, &size, 4);
+ SET_MEM_BUF(addr + 4, dc->send_buf, size);

- if (tty)
- tty_wakeup(tty);
+ if (tty)
+ tty_wakeup(tty);

- return 1;
+ return 1;
}

/* If all data has been read, return 1, else 0 */
-static int receive_data( enum port_type index, dc_t* dc ) {
- u8 buf[RECEIVE_BUF_MAX] = {0};
- int size;
- u32 offset = 4;
- struct port *port = &dc->port[index];
- u8 toggle = port->toggle_dl;
- void __iomem * addr = port->dl_addr[toggle];
- struct tty_struct *tty = port->tty;
- int i;
```

```

-
- if ( !tty ) {
- D1("tty not open for port: %d?", index);
- return 1;
- }
-
- GET_MEM( &size, addr, 4 );
- /* D1( "%d bytes port: %d", size, index); */
-
- if ( test_bit( TTY_THROTTLED, & tty->flags ) ) {
- D1("No room in tty, don't read data, don't ack interrupt, disable interrupt");
-
- /* disable interrupt in downlink... */
- disable_transmit_dl(index, dc);
- return 0;
- }
-
- if (size == 0) {
- dev_err(&dc->pdev->dev, "size == 0?\n");
- return 1;
- }
-
- while( size > 0 ) {
- GET_MEM_BUF(buf, addr + offset, 4);
-
- i = 0;
- while (i < 4 && size > 0) {
- if (tty_buffer_request_room(tty, 1) < 1)
- {
- tty_flip_buffer_push(tty);
- }
- tty_insert_flip_char(tty, buf[i], TTY_NORMAL);
- port->rx_data++;
- i++;
- size--;
- }
-
- offset += 4;
- }
-
- tty_flip_buffer_push(tty);
-
- return 1;
+static int receive_data(enum port_type index, dc_t * dc)
+{
+ u8 buf[RECEIVE_BUF_MAX] = { 0 };
+ int size;
+ u32 offset = 4;
+ struct port *port = &dc->port[index];
+ u8 toggle = port->toggle_dl;
+ void __iomem *addr = port->dl_addr[toggle];

```

[PATCH 1/2] Char: nozomi, Lindent the code

```
+ struct tty_struct *tty = port->tty;
+ int i;
+
+ if (!tty) {
+ D1("tty not open for port: %d?", index);
+ return 1;
+ }
+
+ GET_MEM(&size, addr, 4);
+ /* D1( "%d bytes port: %d", size, index); */
+
+ if (test_bit(TTY_THROTTLED, &tty->flags)) {
+ D1("No room in tty, don't read data, don't ack interrupt, "
+ "disable interrupt");
+
+ /* disable interrupt in downlink... */
+ disable_transmit_dl(index, dc);
+ return 0;
+ }
+
+ if (size == 0) {
+ dev_err(&dc->pdev->dev, "size == 0?\n");
+ return 1;
+ }
+
+ while (size > 0) {
+ GET_MEM_BUF(buf, addr + offset, 4);
+
+ i = 0;
+ while (i < 4 && size > 0) {
+ if (tty_buffer_request_room(tty, 1) < 1) {
+ tty_flip_buffer_push(tty);
+ }
+ tty_insert_flip_char(tty, buf[i], TTY_NORMAL);
+ port->rx_data++;
+ i++;
+ size--;
+ }
+
+ offset += 4;
+ }
+
+ tty_flip_buffer_push(tty);
+
+ return 1;
+ }

/* Debug for interrupts */
#ifdef NOZOMI_DEBUG
-static char* interrupt2str( u16 interrupt) {
- static char buf[TMP_BUF_MAX];
```

[PATCH 1/2] Char: nozomi, Lindent the code

```
- char *p = buf;
+static char *interrupt2str(u16 interrupt)
+{
+ static char buf[TMP_BUF_MAX];
+ char *p = buf;

- interrupt & MDM_DL1 ? p += snprintf(p, TMP_BUF_MAX, "MDM_DL1 ");:0;
- interrupt & MDM_DL2 ? p += snprintf(p, TMP_BUF_MAX, "MDM_DL2 ");:0;
+ interrupt & MDM_DL1 ? p += snprintf(p, TMP_BUF_MAX, "MDM_DL1 "); : 0;
+ interrupt & MDM_DL2 ? p += snprintf(p, TMP_BUF_MAX, "MDM_DL2 "); : 0;

- interrupt & MDM_UL1 ? p += snprintf(p, TMP_BUF_MAX, "MDM_UL1 ");:0;
- interrupt & MDM_UL2 ? p += snprintf(p, TMP_BUF_MAX, "MDM_UL2 ");:0;
+ interrupt & MDM_UL1 ? p += snprintf(p, TMP_BUF_MAX, "MDM_UL1 "); : 0;
+ interrupt & MDM_UL2 ? p += snprintf(p, TMP_BUF_MAX, "MDM_UL2 "); : 0;

- interrupt & DIAG_DL1 ? p += snprintf(p, TMP_BUF_MAX, "DIAG_DL1 ");:0;
- interrupt & DIAG_DL2 ? p += snprintf(p, TMP_BUF_MAX, "DIAG_DL2 ");:0;
+ interrupt & DIAG_DL1 ? p += snprintf(p, TMP_BUF_MAX, "DIAG_DL1 "); : 0;
+ interrupt & DIAG_DL2 ? p += snprintf(p, TMP_BUF_MAX, "DIAG_DL2 "); : 0;

- interrupt & DIAG_UL ? p += snprintf(p, TMP_BUF_MAX, "DIAG_UL ");:0;
+ interrupt & DIAG_UL ? p += snprintf(p, TMP_BUF_MAX, "DIAG_UL ") : 0;

- interrupt & APP1_DL ? p += snprintf(p, TMP_BUF_MAX, "APP1_DL ");:0;
- interrupt & APP2_DL ? p += snprintf(p, TMP_BUF_MAX, "APP2_DL ");:0;
+ interrupt & APP1_DL ? p += snprintf(p, TMP_BUF_MAX, "APP1_DL ") : 0;
+ interrupt & APP2_DL ? p += snprintf(p, TMP_BUF_MAX, "APP2_DL ") : 0;

- interrupt & APP1_UL ? p += snprintf(p, TMP_BUF_MAX, "APP1_UL ");:0;
- interrupt & APP2_UL ? p += snprintf(p, TMP_BUF_MAX, "APP2_UL ");:0;
+ interrupt & APP1_UL ? p += snprintf(p, TMP_BUF_MAX, "APP1_UL ") : 0;
+ interrupt & APP2_UL ? p += snprintf(p, TMP_BUF_MAX, "APP2_UL ") : 0;

- interrupt & CTRL_DL ? p += snprintf(p, TMP_BUF_MAX, "CTRL_DL ");:0;
- interrupt & CTRL_UL ? p += snprintf(p, TMP_BUF_MAX, "CTRL_UL ");:0;
+ interrupt & CTRL_DL ? p += snprintf(p, TMP_BUF_MAX, "CTRL_DL ") : 0;
+ interrupt & CTRL_UL ? p += snprintf(p, TMP_BUF_MAX, "CTRL_UL ") : 0;

- interrupt & RESET ? p += snprintf(p, TMP_BUF_MAX, "RESET ");:0;
+ interrupt & RESET ? p += snprintf(p, TMP_BUF_MAX, "RESET ") : 0;

- return buf;
+ return buf;
}
#endif

/* Receive flow control */
/* Return 1 - If ok, else 0 */
-static int receive_flow_control( dc_t *dc, irq_t *m) {
- enum port_type port = PORT_MDM;
```

[PATCH 1/2] Char: nozomi, Lindent the code

[PATCH 1/2] Char: nozomi, Lindent the code

```
- ctrl_dl_t ctrl_dl;
- ctrl_dl_t old_ctrl;
- u16 enable_ier = 0;
-
- GET_MEM( &ctrl_dl, dc->port[PORT_CTRL].dl_addr[CH_A], 2);
-
- switch( ctrl_dl.port ) {
- case CTRL_CMD:
- D1( "The Base Band sends this value as a response to a request for IMSI detach sent"
- " over the control channel uplink (see section 7.6.1).");
- break;
- case CTRL_MDM: port = PORT_MDM; enable_ier = MDM_DL; break;
- case CTRL_DIAG: port = PORT_DIAG; enable_ier = DIAG_DL; break;
- case CTRL_APP1: port = PORT_APP1; enable_ier = APP1_DL; break;
- case CTRL_APP2: port = PORT_APP2; enable_ier = APP2_DL; break;
- default:
- dev_err(&dc->pdev->dev, "ERROR: flow control received for non-existing port\n");
- return 0;
- };
-
- D1( "0x%04X->0x%04X", *((u16*) &dc->port[port].ctrl_dl), *((u16*)&ctrl_dl));
-
- old_ctrl = dc->port[port].ctrl_dl;
- dc->port[port].ctrl_dl = ctrl_dl;
-
- if ( old_ctrl.CTS == 1 && ctrl_dl.CTS == 0 ) {
- D1( "Disable interrupt (0x%04X) on port: %d", enable_ier, port);
- disable_transmit_ul(port, dc);
-
- } else if ( old_ctrl.CTS == 0 && ctrl_dl.CTS == 1 ) {
-
- if ( __kfifo_len(dc->port[port].fifo_ul) ) {
- D1( "Enable interrupt (0x%04X) on port: %d", enable_ier, port);
- D1( "Data in buffer [%d], enable transmit! ", __kfifo_len(dc->port[port].fifo_ul) );
- enable_transmit_ul( port, dc );
- } else {
- D1( "No data in buffer...");
- }
- }
-
- if(*(u16*)&old_ctrl == *(u16*)&ctrl_dl)
- {
- D1( " No change in mctrl");
- return 1;
- }
- /* Update statistics */
- if(old_ctrl.CTS != ctrl_dl.CTS) {
- dc->port[port].tty_icount.cts++;
- }
- if(old_ctrl.DSR != ctrl_dl.DSR) {
- dc->port[port].tty_icount.dsr++;
```

```

- }
- if(old_ctrl.RI != ctrl_dl.RI) {
- dc->port[port].tty_icount.rng++;
- }
- if(old_ctrl.DCD != ctrl_dl.DCD) {
- dc->port[port].tty_icount.dcd++;
- }
- D1("port: %d DCD(%d), CTS(%d), RI(%d), DSR(%d)",
- port,
- dc->port[port].tty_icount.dcd, dc->port[port].tty_icount.cts,
- dc->port[port].tty_icount.rng, dc->port[port].tty_icount.dsr);
-
- return 1;
+static int receive_flow_control(dc_t * dc, irq_t * m)
+{
+ enum port_type port = PORT_MDM;
+ ctrl_dl_t ctrl_dl;
+ ctrl_dl_t old_ctrl;
+ u16 enable_ier = 0;
+
+ GET_MEM(&ctrl_dl, dc->port[PORT_CTRL].dl_addr[CH_A], 2);
+
+ switch (ctrl_dl.port) {
+ case CTRL_CMD:
+ D1("The Base Band sends this value as a response to a request "
+ "for IMSI detach sent" " over the control channel "
+ "uplink (see section 7.6.1).");
+ break;
+ case CTRL_MDM:
+ port = PORT_MDM;
+ enable_ier = MDM_DL;
+ break;
+ case CTRL_DIAG:
+ port = PORT_DIAG;
+ enable_ier = DIAG_DL;
+ break;
+ case CTRL_APP1:
+ port = PORT_APP1;
+ enable_ier = APP1_DL;
+ break;
+ case CTRL_APP2:
+ port = PORT_APP2;
+ enable_ier = APP2_DL;
+ break;
+ default:
+ dev_err(&dc->pdev->dev, "ERROR: flow control received for "
+ "non-existing port\n");
+ return 0;
+ };
+
+ D1("0x%04X->0x%04X", *((u16 *) & dc->port[port].ctrl_dl),

```

[PATCH 1/2] Char: nozomi, Lindent the code

```
+ *((u16 *) & ctrl_dl));
+
+ old_ctrl = dc->port[port].ctrl_dl;
+ dc->port[port].ctrl_dl = ctrl_dl;
+
+ if (old_ctrl.CTS == 1 && ctrl_dl.CTS == 0) {
+ D1("Disable interrupt (0x%04X) on port: %d", enable_ier, port);
+ disable_transmit_ul(port, dc);
+
+ } else if (old_ctrl.CTS == 0 && ctrl_dl.CTS == 1) {
+
+ if (__kfifo_len(dc->port[port].fifo_ul)) {
+ D1("Enable interrupt (0x%04X) on port: %d", enable_ier,
+ port);
+ D1("Data in buffer [%d], enable transmit! ",
+ __kfifo_len(dc->port[port].fifo_ul));
+ enable_transmit_ul(port, dc);
+ } else {
+ D1("No data in buffer...");
+ }
+ }
+
+ if (*(u16 *) & old_ctrl == *(u16 *) & ctrl_dl) {
+ D1(" No change in mctrl");
+ return 1;
+ }
+ /* Update statistics */
+ if (old_ctrl.CTS != ctrl_dl.CTS) {
+ dc->port[port].tty_icount.cts++;
+ }
+ if (old_ctrl.DSR != ctrl_dl.DSR) {
+ dc->port[port].tty_icount.dsr++;
+ }
+ if (old_ctrl.RI != ctrl_dl.RI) {
+ dc->port[port].tty_icount.rng++;
+ }
+ if (old_ctrl.DCD != ctrl_dl.DCD) {
+ dc->port[port].tty_icount.dcd++;
+ }
+ D1("port: %d DCD(%d), CTS(%d), RI(%d), DSR(%d)", port,
+ dc->port[port].tty_icount.dcd, dc->port[port].tty_icount.cts,
+ dc->port[port].tty_icount.rng, dc->port[port].tty_icount.dsr);
+
+ return 1;
+ }

/* TODO: */
/* - return enum ctrl_port_type */
-static u8 port2ctrl (enum port_type port, dc_t *dc)
+static u8 port2ctrl(enum port_type port, dc_t * dc)
{
```

```

- switch(port) {
+ switch (port) {
case PORT_MDM:
return CTRL_MDM;
case PORT_DIAG:
@@ -1099,7 +1187,8 @@ static u8 port2ctrl (enum port_type port
case PORT_APP2:
return CTRL_APP2;
default:
- dev_err(&dc->pdev->dev, "ERROR: send flow control received for non-existing port\n");
+ dev_err(&dc->pdev->dev, "ERROR: send flow control received for "
+ "non-existing port\n");
};
return -1;
}
@@ -1107,237 +1196,252 @@ static u8 port2ctrl (enum port_type port
/* Send flow control, can only update one channel at a time */
/* Return 0 - If we have updated all flow control */
/* Return 1 - If we need to update more flow control, ack current enable more */
-static int send_flow_control( dc_t *dc ) {
- u32 i, more_flow_control_to_be_updated = 0;
- u16* ctrl;
-
- for( i=PORT_MDM; i<MAX_PORT ; i++ ) {
- if( dc->port[i].update_flow_control ) {
- if ( more_flow_control_to_be_updated ) {
- /* We have more flow control to be updated */
- return 1;
- }
- dc->port[i].ctrl_ul.port = port2ctrl(i, dc);
- ctrl = (u16*) &dc->port[i].ctrl_ul;
- /* D1( "sending flow control 0x%04X for port %d, %d", (u16) *ctrl, i, dc->port[i].ctrl_ul.port ); */
- SET_MEM( dc->port[PORT_CTRL].ul_addr[0], (u32*) ctrl, 2 );
- dc->port[i].update_flow_control = 0;
- more_flow_control_to_be_updated = 1;
- }
- }
- return 0;
+static int send_flow_control(dc_t * dc)
+{
+ u32 i, more_flow_control_to_be_updated = 0;
+ u16 *ctrl;
+
+ for (i = PORT_MDM; i < MAX_PORT; i++) {
+ if (dc->port[i].update_flow_control) {
+ if (more_flow_control_to_be_updated) {
+ /* We have more flow control to be updated */
+ return 1;
+ }
+ dc->port[i].ctrl_ul.port = port2ctrl(i, dc);
+ ctrl = (u16 *) & dc->port[i].ctrl_ul;

```

[PATCH 1/2] Char: nozomi, Lindent the code

```
+/* D1("sending flow control 0x%04X for port %d, %d",
+ (u16)*ctrl, i, dc->port[i].ctrl_ul.port); */
+ SET_MEM(dc->port[PORT_CTRL].ul_addr[0], (u32 *) ctrl,2);
+ dc->port[i].update_flow_control = 0;
+ more_flow_control_to_be_updated = 1;
+ }
+ }
+ return 0;
+ }

/* Handle donlink data, ports that are handled are modem and diagnostics */
/* Return 1 - ok */
/* Return 0 - toggle fields are out of sync */
-static int handle_data_dl(dc_t *dc, irq_t *m, enum port_type port, u8 *toggle, u16 mask1, u16 mask2) {
-
- if ( *toggle == 0 && m->read_iir & mask1 ) {
- if (receive_data( port, dc )) {
- SET_FCR( mask1 );
- *toggle = !(*toggle);
- }
-
- if ( m->read_iir & mask2 ) {
- if (receive_data( port, dc )) {
- SET_FCR( mask2 );
- *toggle = !(*toggle);
- }
- }
- } else if ( *toggle == 1 && m->read_iir & mask2 ) {
- if (receive_data( port, dc )) {
- SET_FCR( mask2 );
- *toggle = !(*toggle);
- }
- }
-
- if ( m->read_iir & mask1 ) {
- if (receive_data( port, dc )) {
- SET_FCR( mask1 );
- *toggle = !(*toggle);
- }
- }
- } else {
- ERR("port out of sync!, toggle:%d", *toggle);
- return 0;
- }
- return 1;
+static int handle_data_dl(dc_t * dc, irq_t * m, enum port_type port,
+ u8 * toggle, u16 mask1, u16 mask2)
+{
+
+ if (*toggle == 0 && m->read_iir & mask1) {
+ if (receive_data(port, dc)) {
+ SET_FCR(mask1);
```

```

+ *toggle = !(*toggle);
+ }
+
+ if (m->read_iir & mask2) {
+ if (receive_data(port, dc)) {
+ SET_FCR(mask2);
+ *toggle = !(*toggle);
+ }
+ }
+ } else if (*toggle == 1 && m->read_iir & mask2) {
+ if (receive_data(port, dc)) {
+ SET_FCR(mask2);
+ *toggle = !(*toggle);
+ }
+ }
+
+ if (m->read_iir & mask1) {
+ if (receive_data(port, dc)) {
+ SET_FCR(mask1);
+ *toggle = !(*toggle);
+ }
+ }
+ } else {
+ ERR("port out of sync!, toggle:%d", *toggle);
+ return 0;
+ }
+ return 1;
+ }
}

/* Handle uplink data, this is currently for the modem port */
/* Return 1 - ok */
/* Return 0 - toggle field are out of sync */
-static int handle_data_ul(dc_t- D1( "CTRL_UL");
- SET_IER( 0, CTRL_UL );
- if ( send_flow_control(dc) ) {
- SET_FCR( CTRL_UL );
- SET_IER( CTRL_UL, CTRL_UL );
- }
- }
- if ( m->read_iir & CTRL_DL ) {
- receive_flow_control(dc, m);
- SET_FCR( CTRL_DL );
- }
- if ( m->read_iir & MDM_DL ) {
- if ( !(handle_data_dl(dc, m, PORT_MDM, &(dc->port[PORT_MDM].toggle_dl), MDM_DL1,
MDM_DL2)) ) {
- ERR("MDM_DL out of sync!");
- goto exit_handler;
- }
- }
- if ( m->read_iir & MDM_UL ) {
- if ( !handle_data_ul(dc, m, PORT_MDM) ) {

```

```
- ERR("MDM_UL out of sync!");
- goto exit_handler;
- }
- }
- if ( m->read_iir & DIAG_DL ) {
- if ( !(handle_data_dl(dc, m, PORT_DIAG, &(dc->port[PORT_DIAG].toggle_dl), DIAG_DL1,
DIAG_DL2)) ) {
- ERR("DIAG_DL out of sync!");
- goto exit_handler;
- }
- }
- if ( m->read_iir & DIAG_UL ) {
- SET_IER( 0, DIAG_UL );
- if( send_data( PORT_DIAG, dc ) ) {
- SET_FCR( DIAG_UL );
- SET_IER( DIAG_UL, DIAG_UL );
- }
- }
- if ( m->read_iir & APP1_DL ) {
- if (receive_data( PORT_APP1, dc ) ) {
- SET_FCR( APP1_DL );
- }
- }
- if ( m->read_iir & APP1_UL ) {
- SET_IER( 0, APP1_UL );
- if(send_data( PORT_APP1, dc )) {
- SET_FCR( APP1_UL );
- SET_IER( APP1_UL, APP1_UL );
- }
- }
- if ( m->read_iir & APP2_DL ) {
- if (receive_data( PORT_APP2, dc )) {
- SET_FCR( APP2_DL );
- }
- }
- if ( m->read_iir & APP2_UL ) {
- SET_IER( 0, APP2_UL );
- if(send_data( PORT_APP2, dc )) {
- SET_FCR( APP2_UL );
- SET_IER( APP2_UL, APP2_UL );
- }
- }
-
- exit_handler:
- spin_unlock(&dc->spin_mutex);
- return IRQ_HANDLED;
+static irqreturn_t interrupt_handler(int irq, void *dev_id,
+ struct pt_regs *regs)
+{
+ dc_t *dc = NULL;
+ irq_t *m = &my_irq;
```

```

+
+ if (my_dev && my_dev->pdev != dev_id) {
+ return IRQ_NONE;
+ }
+
+ if (!(dc = get_dc_by_pdev(dev_id))) {
+ ERR("Could not find device context from pci_dev: %p", dev_id);
+ return IRQ_NONE;
+ }
+
+ GET_IIR(m->read_iir);
+
+ /* Just handle interrupt enabled in IER (by masking with
+ dc->ier_last_written) */
+ m->read_iir &= dc->ier_last_written;
+
+ if (m->read_iir == 0) {
+ return IRQ_NONE;
+ }
+
+ if (dc == NULL) {
+ ERR("ERROR!!");
+ return IRQ_NONE;
+ }
+
+ spin_lock(&dc->spin_mutex);
+
+ D4("%s irq:0x%04X, prev:0x%04X", interrupt2str(m->read_iir),
+ m->read_iir, dc->ier_last_written);
+
+ if (m->read_iir & RESET) {
+ if (!nozomi_read_config_table(dc)) {
+ SET_IER(0, 0xFFFF);
+ ERR("ERR: Could not read status from card, we should "
+ "disable interface");
+ } else {
+ SET_FCR(RESET);
+ }
+ goto exit_handler; /* No more useful info if this was the reset interrupt. */
+ }
+ if (m->read_iir & CTRL_UL) {
+ D1("CTRL_UL");
+ SET_IER(0, CTRL_UL);
+ if (send_flow_control(dc)) {
+ SET_FCR(CTRL_UL);
+ SET_IER(CTRL_UL, CTRL_UL);
+ }
+ }
+ if (m->read_iir & CTRL_DL) {
+ receive_flow_control(dc, m);
+ SET_FCR(CTRL_DL);

```

[PATCH 1/2] Char: nozomi, Lindent the code

```
+ }  
+ if (m->read_iir & MDM_DL) {  
+ if (!(handle_data_dl(dc, m, PORT_MDM,  
+ &(dc->port[PORT_MDM].toggle_dl), MDM_DL1,  
+ 
```