Weitere Serielle Schnittstellen

Inhaltsverzeichnis

1 Welche Schnittstellen stehen zur Verfügung **2**
1.1 Debug-Schnittstelle „/dev/ttyS0” . . . 2
1.2 UAR0 . . . 2
1.3 UART1 . . . 2
1.4 UART2 . . . 2
1.5 UART3 . . . 2

2 Aktivierung serieller Schnittstellen **3**
2.1 Im Linux-Kernel . . . 3
2.2 Für den Userspace . . . 3
2.3 Erzeugen der Device-Nodes . . . 4

1 Welche Schnittstellen stehen zur Verfügung

1.1 Debug-Schnittstelle „/dev/ttyS0”

Für Kernelnahe Ausgaben.

1.2 UAR0

Port	Funktion
PA17	**TXD**
PA18	**RXD**
PA20	CTS
PA21	RTS

1.3 UART1

Verbundene Ports:

Port	Funktion
PB18	RING
PB19	DTR
PB20	**TXD**
PB21	**RXD**
PB23	DCD
PB24	CTS
PB25	DSR
PB26	RTS

1.4 UART2

Port	Funktion
PA22	**RXD**
PA23	**TXD**

1.5 UART3

Bei Bedarf stünde noch /dev/ttyS4 zur Verfügung. (Derzeit nicht eingerichtet.)

2 Aktivierung serieller Schnittstellen

2.1 Im Linux-Kernel

Die Datei kernel/arch/arm/mach-at91rm9200/**board-carmeva.c** enthält eine Zeile mit etwa folgendem Inhalt:

```
#define CARMEVA_UART_MAP { 4, 1, -1, -1, -1 }
```

Diese Konfiguration besagt, dass UART0 („/dev/ttyS0”) mit dem Debug-Port verbunden sein soll, sowie eine weitere serielle Schnittstelle (siehe 1.3) bedient werden soll. Da in aufsteigender Reihenfolge zugeordnet wird, handelt es sich dabei um /dev/ttyS1.

Beispiel

Um zum Beispiel den UART0, UART1, UART2, sowie die Debug-Schnittstelle zu bedienen muss diese Zeile mit folgender ersetzt werden:

```
#define CARMEVA_UART_MAP { 4, 0, 1, 2, -1 }
```

Nachdem der Kernel neu kompiliert wurde, stehen (zumindest für den Kernel) vier serielle Schnittstellen zur Verfügung.

2.2 Für den Userspace

Benutzerprogramme kommunizieren mit Hilfe von sog. Device-Nodes mit dem Kernel. Diese Device-Nodes bedinden sich im Verzeichnis „/dev/”. Für serielle Schnittstellen handelt es sich um Dateien mit der Nomenklatur „/dev/ttySx”, wobei das x eine Ziffer darstellt, welche die entsprechende Schnittstelle addressiert. (Z.B: /dev/ttyS0 für die erste Schnittstelle.)

Damit die im vorherigen Abschnitt gewonnenen Schnittstellen tatsächlich benutzbar werden, müssen entsprechende Device-Nodes erzeugt- bzw. deren Existenz bestätigt werden. Folgende Device-Nodes müssen also existieren:

/dev/ttyS0 *// Debug-Schnittstelle*
/dev/ttyS1 *// UART0*
/dev/ttyS2 *// UART1*
/dev/ttyS3 *// UART2*

2.3 Erzeugen der Device-Nodes

Sollten die Device-Nodes **nicht** vorhanden sein können sie mit **mknod** erstellt werden.

```
mknod /dev/ttyS1 c 4 65
mknod /dev/ttyS2 c 4 66
mknod /dev/ttyS3 c 4 67
mknod /dev/ttyS4 c 4 68
```