UART Serial ports

There are 6 serial ports defined, of which at least 5 can operate with terminals or in any other way that serial ports can be used. Some of the ports have the RTS/CTS handshaking line pair also, as shown. UART0 TX and RX appear on both Euler (J48) and Expansion (J51). Ground connections refer to pins physically near the TX and RX pins; any other Ground pins can be used as well. The serial ports defined are available at the following pins:

UART# /dev/ttyS#	Tx	Rx	RTS (output)	CTS (input)	GND	
0	X7	X8			X6	Voh = 3.3 V nominal. Gated so there is no back-feeding when power is off.
0	E29	E30			E34	Voh = 3.3 V nominal
1	B2	B4	B6	B8	B10	Voh = 1.8 V nominal
2	P8	P10	E11	E27	P6	Voh = 3.3 V nominal
3	E24	E23			E25	Voh = 2.5 V nominal
3	T2	P29	P31	(P26)	P30	Voh = 2.5 V nominal. Alternate connections seen on some recent kernels. See discussion below.
4	E19	E21	E18	E22	E20	Voh = 2.5 V nominal
S-	W9	W11			W12	Voh = 1.8 V nominal
Adafruit USB cable (#954)	White (Rx)	Green (Tx)			Black (Gnd)	(Red is 5V, NC)
Sparkfun USB cable (CAB-12977)	Yellow (Rx)	Orange (Tx)			Black (Gnd)	(Red is 5V, NC)
Adafruit FTDI Serial TTL-232 USB cable (#70)	Yellow (Rx)	Orange (Tx)	Brown (CTS)	Green (RTS)	Black (Gnd)	(Red is 5V, NC)
DE9M (DTE) pin #	3	2	7	8	5	(Via level translator)
DB25M (DTE) pin #	2	3	4	5	7	(Via level translator)

The signal level on UARTs 0 and 2 is 3.3V, on UARTs 3 and 4 it is 2.5V, and on UARTs 1 and S it is 1.8 V. so the connections to a standard DE9 plug shown will have to be made via level translators in an ST3232 or similar that converts from RS-232 Mark and Space to H and L.

There is a getty going on ttyS0 (UART0) and in order to start these on other serial ports, use something like what is shown here for ttyS2 here:

```
systemctl enable serial-getty@ttyS2.service
systemctl start serial-getty@ttyS2.service
```

Similarly, in order to remove the getty the equivalent commands are:

```
systemctl stop serial-getty@ttyS2.service
systemctl disable serial-getty@ttyS2.service
```

Where has uart3 moved?

2017-11-13: For some reason the /dev/ttyS3 seemed inoperative on the more recent kernels. (3.10.105-bsp-1.2-ayufan-118). Code for opening, writing and reading didn't fail, but nothing appeared on the wires. Some more investigation of the GPIO control registers for pins PD0 and PD1 showed that these two are at function 7 which means they are disabled. Notably, bits PD5, PD4, PD3, and PD2, which are associated with UART4, were at the expected function 3, which is the serial-port settings. Changing these settings to 3 for PD1 and PD0 was all that was needed for the UART3 to start emitting and receiving signals on the assigned wires, PD0 at pin E24 for transmission, and PD1 at pin E23 for reception.

Inspection of the device-tree file, indicates that the UART3 is actually enabled on PH4=Tx (T2) PH5=Rx (P29) PH6=RTS (P31) PH7=CTS (P26). Now that is somewhat less convenient since PH4, transmit data, is one of the Touchpad connector pins, identified as at connection T2. Unless this is broken out, it will not make the UART3 particularly useful... Besides that, the PH7 is already being used for a codec output of some sort, and is being allocated, as per /sys/kernel/debug/gpio.

The file checku.c contains the code I used for performing this check and making the changes. This code is similar to what is shown on the page for /dev/mem gpio handling.

This function shows the functions of all the first 8 PD and PH ports, PD7 (GPIO103, at pin E28) and PD6 (GPIO102 at pin E26) are shown as function 7, disabled, also. This makes sense, since these are unallocated, and available for us to use. PH7 (GPIO231 at pin P26) shows as 1, output, and the rest of PH6-PH0 are function 2. Now for PH3-PH0 this function means they operate as the two i2c busses, so that has to remain the same!

Now, it turns out, that when any gpio not otherwise allocated, is exported via sysfs (see the sysfs gpio details), its function is not changed. So if it started out as disabled, it remains disabled, even though the direction file (/sys/class/gpio/gpio103/direction) says "in". Only when this direction file is written with the values "in" or "out" does this register value change, to 0 for input and 1 for output. (Allwinner_A64_User_Manual_V1.0.pdf, section 3.21.2.19, page 384) These settings also remain after the gpio is unexported, and thus a gpio will have the same direction after another exporting as it had before. This was noted as a kind of bug in the forum.

To Pine 64+ main page
