# FDT
# Flattened Device Tree

Ka-Ro electronics GmbH, Pascalstr. 22, 52076 Aachen, GERMANY - Tel: +49 2408 1402-0 (FAX: -10)
**www.karo-electronics.de**

# Table of contents

# 1 Introduction

This document gives an introduction into the commands that can be used within U-Boot to edit the DTB. For viewing the FDT paths and how to *"printout"* of the current DT see **Chapter 5.**

**Important:**

Examples herein are general applicable to any Ka-Ro TX CoM running a current U-Boot and are not specific to any one.

# 2 Device Tree

(also see `U-Boot/TX##_U-Boot.pdf` - Ch. 4.4)

Accompanying the Linux Kernel changes (*since Linux 3.4*) of hardware initialization, configuration and the move from an effectively hard-coded, built-in structure to a generalized and interchangeable form saw the wide introduction of the *Flattened Device Tree* (**FDT**) [or *Device Tree* (**DT**) for short] structure. This FDT is usually provided in form of a *Device Tree Blob* (**DTB**) to the Kernel at boot time.

## 2.1 Device Tree Blob (DTB)

U-Boot has the capabilities to allow users either to exchange the provided Ka-Ro TX CoM StarterKit specific default **FDT DTB** by a customized - i.e. user specific - **DTB** or to edit the provided **DTB** while it's loaded and U-Boot is running. The latter also allows users to do on-fly non-volatile changes in case of a test and/or development setup. Editing a loaded **DTB** is described in the Chapters **3** & **4**.

At boot-time U-Boot always will try to load a DTB from the partition `"dtb"` and will output error messages if empty or invalid. Users not using a DT can ignore pertaining messages. Upload of a DTB to a TX CoM via U-Boot is described the Chapter **3.4** ff.

The provided (release) **DTB** are found on the StarterKit CD under:

(general notation)
`Linux/target/<soc>[-<board[-variation[-subvariant]]>].dtb`

e.g.:

```
Linux/target/imx6q-tx6q-1020-comtft.dtb
Linux/target/imx53-tx53-x13x.dtb
Linux/target/am335x-tx48.dtb
etc.
```

To upload a user created **DTB** to the NAND `"dtb"` partition; users first have to edit the appropriate device tree source files (DTS – Ch. **2.2**) in the Linux kernel sources and recompile them as a DTB output file(s).

## 2.2 Device Tree Source (DTS)

As the FDT is part of and developed in the Linux Kernel an DTB is created from the Linux Kernel sources. Therefore any changes to the FDT are Linux Kernel changes and vice versa. Therefore it has to be made sure to use the respective Linux Kernel appendant DTB sources.

The source of the DTB are to be found in the Linux Kernel source directory under:

`[Linux-src/]arch/arm/boot/dts/*.dts[i]`

where the notation follows (e.g. TX6):

- ➔ General notation:
  `[Linux-src/]arch/arm/boot/dts/<soc><-board[-variation[-subvariant]]>.dts[i]`
- ➔ CPU/SoC specific configuration (e.g. i.MX6Q and i.MX6DL ⇒ i.MX6QDL):
  `[Linux-src/]arch/arm/boot/dts/imx6qdl.dtsi`
- ➔ common board configuration (e.g. TX6 ⇒ TX6Q (i.MX6Q) **and** TX6U (i.MX6DL)):
  `[Linux-src/]arch/arm/boot/dts/imx6qdl-tx6.dtsi`
- ➔ specific board setup (e.g. TX6U-8010):
  `[Linux-src/]arch/arm/boot/dts/imx6dl-tx6u-801x.dts`
- ➔ specific platform (subvariant) setup (e.g. TX6Q-1010 COMpactTFT):
  `[Linux-src/]arch/arm/boot/dts/imx6q-tx6q-1010-comtft.dts`

Customisations of the FDT are done in DTS files of the "*specific platform (subvariant) setup*" or "*specific board setup*" level in the hierarchy. The syntax of the FDT allows for modifcation (removal, overwrite, expansion, etc.) of nodes and values as inherited from the root onward in the (device-) treestructure.

The DTSI (**D**evice **T**ree **S**ource **I**nclude) files define and specify the general structure of the FDT, its nodes, properties and values and are applicable to **all** devices in the hierarchy.

Thus it is strongly recommended to copy a readily available "*specific platform (subvariant) setup*" file with appropriate unique (re-)naming, which then to adjust this file to the needs. This will also allow for future compatibility with any patches to the source code as they might be obtained via e.g. git from outside sources (e.g. kernel.org).

## 2.3 Documentation

For full and comprehensive documentation and further information about the FDT notation, structure, bindings and handling please refer to:

- Kernel (primary):

  [*Linux-src/*]Documentation/devicetree/*
  [*Linux-src/*]Documentation/devicetree/bindings/*

- URL:

  https://devicetree.org
  http://elinux.org/Main_Page
  http://elinux.org/Device_Tree_Usage
  http://elinux.org/Device_Tree_Reference
  https://wiki.ubuntu.com/KernelTeam/ARMDeviceTrees
  http://wiki.freebsd.org/FlattenedDeviceTree

- StarterKit CD:

  Linux/README
  U-Boot/TX##_U-Boot.pdf

# 3 Managing the Device Tree in U-Boot

If U-Boot finds a 'dtb' partition it will automatically load the therein saved DTB and provide it in human readable form to the user, e.g. for editing. For manipulating the loaded DT U-Boot provides the '**fdt**' command.

Further does U-Boot make use of the customization variables. These are variables set in the U-Boot environment, which allow to apply common changes to the FDT automatically. They are applied to the FDT by U-Boot either at boot (e.g. 'bootm') or by using the command '**fdt boardsetup**'. Please see also for about customization variables:

```
Linux/TX##-Driver.pdf
U-Boot/TX##_U-Boot.pdf
```

The command '**fdt boardsetup**' (which also is automatically being run by the 'bootm' command) will, according to environment settings, as well based on the Ka-Ro StarterKit baseboard designs, disable some device nodes.

The settings for the Ka-Ro StarterKit baseboard designs are read and applied by the command fdt boardsetup from the U-Boot environment variable '**baseboard**'. This behaviour is can be disabled or modified by changing the variable's value as outlined in *U-Boot/TX##_U-Boot.pdf - Ch. 4.1 Customization variables*.

E.g.:

| Environment Setting | Disabled Nodes |
|---|---|
| otg_mode=host | → usbotg → dr_mode=host |
| otg_mode=device | → usbotg → dr_mode=device |
| otg_mode=<*UNSET*> | → usbotg → status=disabled |
| | |
| touchpanel=edt-ft5x06 | → tsc2007 → status=disabled |
| touchpanel=tsc2007 | → edt-ft5x06 → status=disabled |
| touchpanel=<*UNSET*> | → tsc2007 + edt-ft5x06 |

***Note:***
The 'bootm' command is invoked
when booting Linux via: 'run bootm_cmd'

### 3.1 View nodes and properties

The following commands can be used to view the DTB, as loaded from the U-Boot DTB partition, either in its entirety or specific nodes:

- Show whole FDT

  ```
  fdt print
  ```

- Show specific node or property

  ```
  fdt print <alias|/path/to/node> [<property>]
  ```

  e.g.:

  ```
  fdt print /soc/aips-bus@02100000/ethernet@02188000 status
  fdt print /soc/aips-bus@02000000/flexcan@02094000
  fdt print /aliases
  fdt print display/display-timings
  fdt print display
  ```

The node `'display'` is an alias and thus is dereferenced by omitting the leadin `'/'`.

### 3.2 Aliases

The release FDT comes with a node called `'aliases'` which includes shortcuts to the most commonly used nodes. Nodes aliased can be dereferenced by omitting the leading `'/'` when accessed.

The full list for each TX CoM can be view by using the command:

```
fdt print /aliases
```

Which will show a list similar to the following (shortened):

```
aliases {
        ethernet0 = "/soc/aips@60000000/ethernet@63fec000";
        gpio0 = "/soc/aips@50000000/gpio@53f84000";
[...]
        i2c0 = "/soc/aips@60000000/i2c@63fc8000";
[...]
        mmc0 = "/soc/aips@50000000/spba@50000000/esdhc@50004000";
[...]
        display = "/display@di0";
};
```

### 3.3 Enabling, Disabling and Modifying individual nodes in FDT

Any changes made by the commands herein are temporary till written to the DTB as outlined in Ch. **3.4.2** - **Modification Save**.

The following commands can be used to alter the DTB, as loaded from the U-Boot DTB partition, on the TX CoM:

- Remove nodes

  `fdt rm`

- Create additional nodes

  `fdt mknode`

- Modify or add new properties

  `fdt set`

Most device nodes in FDT use a property named '*status*' that can be set to either '**disabled**' or '**okay**' to disable or enable the device. This property can be manipulated thus:

```
fdt set </path> status disabled
fdt set </path> status okay
```

e.g. (*TX6Q*):

```
fdt set /soc/aips-bus@02100000/ethernet@02188000 status disabled
```

will disable the Ethernet interface on an e.g. TX6Q.

Refer to 'Linux/TX##-Driver.pdf' for a list of the device paths and aliases used on the StarterKit & TX CoM assembly. Comprehensive path listing please see below in Ch. **5** (**FDT paths**).

In case of an aliased node the user can truncate the leading '/' for accessing the desired node, e.g.:

```
fdt set display/display-timings/hsd100pxn1 de-active <0x00000000>
```

**Important:**

A commonly encounter error message:

`running libfdt fdt_setprop(): FDT_ERR_NOSPACE`

The routine loading the FDT allocates memory as needed; therefore it might be necessary for successfully complementary editing of the DT that the user executes the following command:

```
fdt resize
```

This will reallocate memory and expand it to allow complementary editing.

This also will resize by expanding or contracting the allocated memory depending on the de-/activated or added/removed nodes.

### 3.4 Write and Read the DTB

Hereafter a short step-by-step description for how a DTB on a TX CoM can be written or read.

It should be noted that the whole DT data can be saved to and reloaded from any NAND/eMMC partition, whereas the provided 'dtb' partition (i.e. eMMC: boot partition) is the pre-defined default and is auto-loaded from at boot.

The general procedure would be similar to Ch. **3.4.3** - **(Re-)Loading**; in turn this will **disable** the features of logo and splash image in U-Boot.

At boot-time U-Boot will always try to load a DTB from the partition 'dtb' – in case of NAND, or the protected boot partition in case of eMMC - and will output error messages if empty or invalid. Users not using a DT can ignore the pertaining messages.

#### 3.4.1 Initial (or Re-)Upload

In the case of initially uploading or re-upload either the BSP provided or the user's own customized DTB (see also TX##_U-Boot.pdf):

```
bootp
tftp ${fdtaddr} am335x-tx48.dtb
run fdtsave
```

The command '*fdtsave*' is a U-Boot script that implements the combination of following manual commands depending on available NVM:

- NAND:

```
bootp
tftp ${fdtaddr} am335x-tx48.dtb
nand erase.part dtb
nand write ${fileaddr} dtb ${filesize}
```

- eMMC:

```
bootp
tftp ${fdtaddr} am335x-tx48.dtb
mmc partconf 0 1 1 1
mmc write ${fileaddr} 0x680 80
mmc partconf 0 1 1 0
```

> ***Note:***
> Actual filename depends on TX CoM in use. The release supplied DTB can be found on StarterKit CD: Linux/target/

The variables '***${fileaddr} ${filesize} ${fdtaddr}***' are general available variables in U-Boot's namespace. Commands like 'tftp' will automatically set '***${fileaddr} ${filesize}***'. Using these variables guaranties feature stable, trouble-free and general applicable usage of commands.

### 3.4.2 Modification Save

After having modified the DT in memory via U-Boot '**fdt**' command, the user has two choices to write these modifications back to NVM. The prefered method is to use the predefined macro variable 'fdtsave' which contains the commands to facilitate the update DTB in the NVM with the currently active FDT. Thus the command (without quotes – 1st):

```
"run fdtsave"
```

This will achieve the same as the commands below with independence to the NVM technology as deployed by the TX CoM in question. DTB into NVM (2nd):

- NAND:

```
nand erase.part dtb
nand write ${fdtaddr} dtb ${fdtsize}
```

- eMMC:

```
mmc partconf 0 1 1 1
mmc write ${fdtaddr} 0x680 80
mmc partconf 0 1 1 0
```

### 3.4.3 (Re-)Loading

(Re-)Loading the DT data to memory from:

- NAND

```
nand read ${fdtaddr} dtb
```

- eMMC

```
mmc read ${fdtaddr} 0x680 80
```

## 3.5 Display

The U-Boot environment variable `'video_mode'` is evaluated for (pre-)defined values in the FDT structure and has the double-acting function of managing:

- U-Boot attached display for logo or splash screen output
- Linux Kernel command line for graphical output

The DTB, as provided by Ka-Ro, has already (pre-)defined values. The user can display these using the following command:

```
fdt print display
```

Setting `'video_mode'` to one of the there given values will allow users to display the already integrated Logo or further use the U-Boot splash screen feature as described in here:

```
U-Boot/TX##_U-Boot.pdf
```

Setting the environment variable (permanently) is done by the commands (e.g. `ET0700`):

```
set video_mode ET0700
save
```

Following a short list of available (i.e. pre-defined) modes:
(The list follows displays as given under `"/Datasheets/Display/*.pdf"`)

| *video_mode* **value** | **Display** | **Resolution** | **Initials** |
|---|---|---|---|
| **LCD** | | | |
| ET0350 | ET0350G0DH6 / ET0430G2DH6 | 480x272 | HVGA |
| ET0430 | ET0430G0DH6 / ET0430G2DH6 | 480x272 | HVGA |
| ETQ570 | ETQ570G0DH6 / ET0430G2DH6 | 320x240 | QVGA |
| ETV570 | ETV570G0DH6 / ETV570G2DH6 | 640x480 | VGA |
| ET0500 | ET0500G0DH6 / ET0500G2DH6 | 800x480 | WVGA |
| ET0700 | ET0700G0DH6 / ET0700G2DH6 | 800x480 | WVGA |
| VGA | standard VGA configuration | 640x480 | *VGA* |
| **LVDS[1)]** | | | |
| hsd100pxn1 | HSD100PXN1 | 1024x768 | XGA |
| **CoMTFT[2)3)]** | | | |
| COMTFT | ET070001DM6 (CoMTFT) | 800x480 | WVGA |

1) Only LVDS version of: TX53, TX6
2) Only TX6
3) CoMTFT platform is compatible with non-TX6 CoM, unsupported

### 3.5.1 User defined Video Modes

An example output of available settings for a mode as seen above:

```
ET0500 {
	clock-frequency = <0x01fb9180>;
	hactive = <0x00000320>;
	vactive = <0x000001e0>;
	hback-porch = <0x00000058>;
	hsync-len = <0x00000080>;
	hfront-porch = <0x00000028>;
	vback-porch = <0x00000021>;
	vsync-len = <0x00000002>;
	vfront-porch = <0x0000000a>;
	hsync-active = <0x00000000>;
	vsync-active = <0x00000000>;
	de-active = <0x00000001>;
	pixelclk-active = <0x00000000>;
};
```

Taking a provided mode (e.g. directly above) as an example the user can create custom/user defined video mode(s) using the U-Boot '**fdt**' commands as outlined in Ch. **3.3** (**Enabling, Disabling and Modifying individual nodes in FDT**).

It is also possible to create a custom mode by using the auto-create feature as used in the below shown Ch. **3.5.2** (**Minimal structure**).

### 3.5.2 Minimal structure

U-Boot extracts its video setup information from the FDT. Creating a minimal structure allows users to outright forego to install a DTB, such as the one provided by Ka-Ro. The required minimal structure in FDT looks similar following output for an exemplary VGA display:

```
/ {
     aliases {
          display = /display;
     };
     display {
          display-timings {
               VGA {
                    clock-frequency = <0x01808580>;
                    hactive = <0x00000280>;
                    vactive = <0x000001e0>;
                    hback-porch = <0x00000030>;
                    hsync-len = <0x00000060>;
                    hfront-porch = <0x00000010>;
                    vback-porch = <0x0000001f>;
                    vsync-len = <0x00000002>;
                    vfront-porch = <0x0000000c>;
                    hsync-active = <0x00000000>;
                    vsync-active = <0x00000000>;
                    de-active = <0x00000001>;
                    pixelclk-active = <0x00000000>;
               };
          };
     };
};
```

#### 3.5.2.1 Creating a minimal structure

U-Boot searches the nodes below `'display-timing'` for a node matching the `'video_mode'` variable and extracts the information for configuring the display from this node.

If such a node is not found, U-Boot tries to interpret the contents of the `'video_mode'` variable as a mode definition in this form:

```
<xres>x<yres>-<bpp>@<frame rate>
```

e.g. for VGA:

```
640x480-24@60
```

If the 'video_mode' follows above given syntax it can be parsed as a valid mode and a node with the same name will be auto-created under 'display-timings' at boot time. The parameters which can not be derived from the mode string will be assigned default values which in turn can be further edited manually before saving the newly created FDT with 'run fdtsave'.

#### 3.5.2.2 Minimum Structure

The minimal structure as indicated by the chapter **3.5.2.1** can not be auto-created with an empty 'dtb' partition. Users therefore need to create a FDT minimum structure, i.e. non-empty DTB partition.

This has to be created before any display timing can be successfully setup and thus creating the mentioned minimal structure (Ch. **3.5.2.1**); as the auto creation of a video mode requires the existence of a 'display-timings' node.

A minimum structure itself can be established with the following commands (incl. an alias for display):

```
fdt mk / aliases
fdt set /aliases display /display
fdt mk / display
fdt mk /display display-timings
run fdtsave
```

***Note:***
The nodes created by U-Boot will be volatile unless explicitly saved with the 'fdtsave' script.

# 4 U-Boot commands

Following an excerpt from the U-Boot Guide (see `U-Boot/TX##_U-Boot.pdf` (Ch. 7 )) about U-Boot's FDT command.

U-Boot will auto-complete the commands by pressing the `"TAB"` key. Respectively it will complete to the least common denominator and show a list of commands, while allowing to auto-complete step-by-step.

## 4.1 Overview of commands

(Click command to jump)



> ***Note:***
> * Commands are not available on all TX Series CoM

## 4.2 Commands and explanations

`fdt`

⇨ flattened device tree utility commands

| | |
|---|---|
| `fdt addr [-c] <addr> [<length>]` | - Set the fdt location to `<addr>` |
| `fdt boardsetup` | - Do board-specific set up |
| `fdt move <fdt> <newaddr> <length>` | - Copy the fdt to `<addr>` and make it active |
| `fdt resize` | - Resize fdt to size + padding to 4k addr |
| `fdt print <path> [<prop>]` | - Recursive print starting at `<path>` |
| `fdt list <path> [<prop>]` | - Print one level starting at `<path>` |
| `fdt get value <var> <path> <prop>` | - Get `<property>` and store in `<var>` |
| `fdt get name <var> <path> <index>` | - Get name of node `<index>` and store in `<var>` |
| `fdt get addr <var> <path> <prop>` | - Get start address of `<property>` and store in `<var>` |
| `fdt get size <var> <path> [<prop>]` | - Get size of [`<property>`] or num nodes and store in `<var>` |
| `fdt set <path> <prop> [<val>]` | - Set `<property>` [to `<val>`] |
| `fdt mknode <path> <node>` | - Create a new node after `<path>` |
| `fdt rm <path> [<prop>]` | - Delete the node or `<property>` |
| `fdt header` | - Display header info |
| `fdt bootcpu <id>` | - Set boot cpuid |
| `fdt memory <addr> <size>` | - Add/Update memory node |
| `fdt rsvmem print` | - Show current mem reserves |
| `fdt rsvmem add <addr> <size>` | - Add a mem reserve |
| `fdt rsvmem delete <index>` | - Delete a mem reserves |
| `fdt chosen [<start> <end>]` | - Add/update the /chosen branch in the tree `<start>`/`<end>` - initrd start/end addr |

> ***NOTE***:
> Dereference aliases by omitting the leading `'/'`, e.g. `fdt print ethernet0`

# 5 FDT paths

- U-Boot:

  A printout of the whole FDT as can be shown by using the U-Boot command:

  ```
  fdt print
  ```

  Any specific nodes can be shown by:

  ```
  fdt print <alias|/path/to/node> [<property>]
  ```

  e.g. (aliases are dereferenced by omitting the leading '/'):

  ```
  fdt print display
  fdt print /ahb@80080000/usb@80080000
  ```

- Linux:

  Linux users can (while running) see the Device Tree under (*rootfs*):

  ```
  /proc/device-tree
  ```

  or:

  ```
  /sys/firmware/devicetree/base
  ```

  The Linux kernel has to be compiled with the option "CONFIG_PROC_DEVICETREE" set ***true***.

  Refer to 'Linux/TX##-Driver.pdf' for a list of the device paths and aliases provided and used on the StarterKit & TX-CoM assembly.

# 6 Document revision history

| Revision | Changes |
|---|---|
| 2017-03-29 | Minor Changes: Ch. 2.3 - Updated URI, Ch. 4 - Command 'fdt' updated; Changed Monospaced-Font to Roboto-Mono |
| 2016-03-18 | Minor Changes: Ch. 3 ff. - Added eMMC, Ch. 5 - Added 'aliases'; Extended Ch. 2.2; Changed Monospaced-Font to Liberation-Mono |
| 2015-05-27 | Minor Changes: Ch. 2.3, Ch. 3 |
| 2015-04-02 | Minor Changes: Typo: Ch. 3.4.2 – Incorrect command. |
| 2015-03-26 | Minor Changes: Formatting and typos |
| 2015-01-06 | Minor Changes: Front page footer DIN 5008/2005-05 correction |
| 2014-11-27 | Changed order of chapters (commands before usage); Added: Ch. 3.1 – Viewing, Ch. 3.2 - Aliases |
| 2014-10-10 | Formatting Ch. 3, 3.1 ff., Ch.4.1 (Frame), Ch. 5 (Command visibility & example); Typos; Added: Ch. 3.1 comprehensive table instead of simple listing. |
| 2014-09-04 | Formatting, Typos |
| 2014-08-19 | Formatting, clarified: auto-create (`video_mode`), |
| 2014-08-12 | Formatting, Splitted Ch. 2 – adding 2.1 DTB & 2.2 DTS & 2.3 Documentation, Rephrased Ch. 3 first paragraphs, expanded Ch. 3.1 Display, added "Minimum Structure", Rephrased Ch 3.3 |
| 2014-07-30 | Updated to newest revision and unified TX CoM Documents |
| 2014-04-11 | Minor changes: `fdtsave` initialize |
| 2014-01-17 | Update to 3.13-rc7 release, split TX-Com general and special particulars |
| 2013-12-18 | Initial release |