How to change the CPU frequency

Contents

Stable: 21.02.2020 - 08:48 / Revision: 13.02.2020 - 14:44

Stable: 21.02.2020 - 08:43 | Revision: 18.02.2020 - 14:44

A quality version of this page, approved on *21 February 2020*, was based off this revision.

Contents

1 Purpose

This article explains how to change the CPU **operating point** (also known as OPP). An operating point corresponds to the **frequency** of the processor and the **voltage** that needs to be supplied to sustain it.

2 Hardware side

On STM32MP1 Series products, the Cortex-A7 core is:

- clocked by PLL1 from the RCC internal peripheral. The PLL1P output **frequency** can be directly propagated to the core, or it can go through an intermediate MPUDIV divider
- supplied with VDDcore **voltage**.

The part number tells which devices can be clocked up to 800 MHz, with associated usage conditions. Otherwise, the frequency must be kept below 650 MHz.

3 Software side

3.1 For ecosystem release v1.2.0

The ecosystem release v1.2.0 is backward compatible with the previous deliveries, so it is still possible to set the CPU **frequency** in the TF-A device tree, as described in the paragraph below. If your part number supports up to 800 MHz, ensure that the VDDcore minimum **voltage** is increased from 1.2V to 1.35 V while running above 650 MHz; this is done in the **regulators** node of the board device tree.

By default however, the ecosystem release v1.2.0 does not set any configuration for PLL1 in the FSBL (TF-A). Instead, TF-A automatically selects the operating point that is suitable for the current part number according to the OPP table code below (embedding **frequency** and **voltage** couples) defined in the SOC device tree fdts/stm32mp157c.dtsi .

```
        cpu0_opp_table: cpu0-opp-table {
                compatible = "operating-points-v2";
                opp-shared;

                opp-650000000 {
                        opp-hz = /bits/ 64 <650000000>;
                        opp-microvolt = <1200000>;
                        opp-supported-hw = <0x1>;
                };
                opp-800000000 {
                        opp-hz = /bits/ 64 <800000000>;
                        opp-microvolt = <1350000>;
                        opp-supported-hw = <0x2>;
                };
        };
```

Notes:

- The operating point supported by devices supporting above 650 MHz (and up to 800 MHz) is identified by the **opp-supported-hw** property set to 0x2.
- This same OPP table must also be present in the Linux SOC device tree arch/arm/boot/dts/stm32mp157c.dtsi .
- It is possible to do a mix, specifying the PLL1 configuration (without an OPP table) in the TF-A device tree, whereas Linux uses the OPP table to increase the CPU frequency, once started.
- This description is valid for cold boot, but also when coming back from Standby low power mode.
- During cold boot, TF-A computes and saves the PLL1 settings for all operating points available in the device tree in compliance with the hardware capabilities. These saved paramaters are used later to increase the performance of the system-state transitions.

3.2 For ecosystem release v1.1.0

The Cortex-A7 core frequency is selected at boot time, by the FSBL (TF-A), following the Clock device tree configuration - Bootloader specific syntax. The frequency is set to 650 MHz by default, as shown in STM32MP15 clock tree. This configuration is set in the TF-A device tree, for instance fdts/stm32mp157a-dk1.dts file for STM32MP157C-DK1 board.

```
&rcc {
        ...
        /* VCO = 1300.0 MHz => P = 650 (CPU) */
        pll1: st,pll@0 {
```

```
                cfg = < 2 80 0 0 0 PQR(1,0,0) >;
                frac = < 0x800 >;
        };
        ...
};
```

The user can reduce this frequency by changing the above configuration, either manually or via the STM32CubeMX graphical user interface that allows generation of the corresponding device tree file.

Central processing unit

Operating Performance Point (link to voltage and frequency scalings)

Trusted Firmware for Arm Cortex-A

First Stage Boot Loader

Stable: 17.08.2020 - 12:42 / Revision: 09.04.2020 - 14:22

A quality version of this page, approved on *17 August 2020*, was based off this revision.

Contents

1 Article purpose

This article describes the specific **RCC** internal peripheral configuration done by the first stage bootloader:

- TF-A for the Trusted boot chain
- U-Boot SPL for the Basic boot chain

Warning

This article explains how to configure the clock tree in the RCC at boot time.
You can then refer to the clock device tree configuration article to understand how to derive each internal peripheral clock tree in Linux®OS from the RCC clock tree.

The configuration is performed using the device tree mechanism that provides a hardware description of the RCC peripheral.

This clock tree is only used in the device tree of the boot chain FSBL; so in the TF-A device tree for OpenSTLinux official delivery (or in SPL only for the DDR tuning tool).

Even if the clock tree information is also present in the U-Boot device tree, it is not used during boot by this SSBL.

2 DT bindings documentation

The bootloader clock device tree bindings correspond to the vendor clock DT bindings used by the clk-stm32mp1 driver of the FSBL (TF-A or U-Boot SPL), it is based on:

- binding described in Clock_device_tree_configuration
- bootloader specific properties described in #DT configuration

This binding document explains how to write the device tree files for clocks on the bootloader side:

- TF-A: *tf-a/docs/devicetree/bindings/clock/st,stm32mp1-rcc.txt*"[1]
- U-Boot SPL: *doc/device-tree-bindings/clock/st,stm32mp1.txt*[2]

3 DT configuration

This hardware description is a combination of the **STM32 microprocessor** device tree files (*.dtsi* extension) and **board** device tree files (*.dts* extension). See the Device tree for an explanation of the device tree file split.

STM32CubeMX can be used to generate the board device tree. Refer to How to configure the DT using STM32CubeMX for more details.

3.1 DT configuration (STM32 level)

The STM32MP1 clock nodes are located in *stm32mp157c.dtsi*[3] (see Device tree for more explanations):

- fixed-clock defined in clock node
- RCC node for #STM32MP1 clock node: clock generation and distribution.

```
/ {
...
        clocks {
                clk_hse: clk-hse {
                        #clock-cells = <0>;
                        compatible = "fixed-clock";
                        clock-frequency = <24000000>;
                };

...
          };
...
        soc {
...
                rcc: rcc@50000000 {
                        compatible = "st,stm32mp1-rcc", "syscon";
                        reg = <0x50000000 0x1000>;
                        #clock-cells = <1>;
                        #reset-cells = <1>;
                        interrupts = <GIC_SPI 5 IRQ_TYPE_LEVEL_HIGH>;
                };
...

        };

};
```

Please refer to clock device tree configuration for the bindings common with Linux® kernel.

3.2 DT configuration (board level)

3.2.1 Clock node

The clock tree is also based on five fixed clocks in the clock node. They are used to define the state of associated ST32MP1 oscillators:

- clk-lsi
- clk-lse
- clk-hsi

- clk-hse
- clk-csi

Please refer to clock device tree configuration for detailed information.

At boot time, the clock tree initialization performs the following tasks:

- enabling of the oscillators present in the device tree and not disabled (node with status="disabled"),
- disabling of the HSI oscillator if the node is absent or disabled (HSI is always activated by the ROM code).

This information is located in the following files:

- STM32MP157C-EV:
  - TF-A: *fdts/stm32mp157c-ed1.dts*[4]
  - U-Boot SPL: *arch/arm/dts/stm32mp157c-ed1-u-boot.dtsi*[5]
- STM32MP157X-DK:
  - TF-A: *fdts/stm32mp157a-dk1.dts*[6].
  - U-Boot SPL: *arch/arm/dts/stm32mp157a-dk1-u-boot.dtsi*[7].

3.2.1.1 *Optional properties for "clk-lse" and "clk-hse" external oscillators*

For external oscillator HSE and LSE, the default clock configuration is an external crystal/ceramic resonator.

Four optional fields are supported:

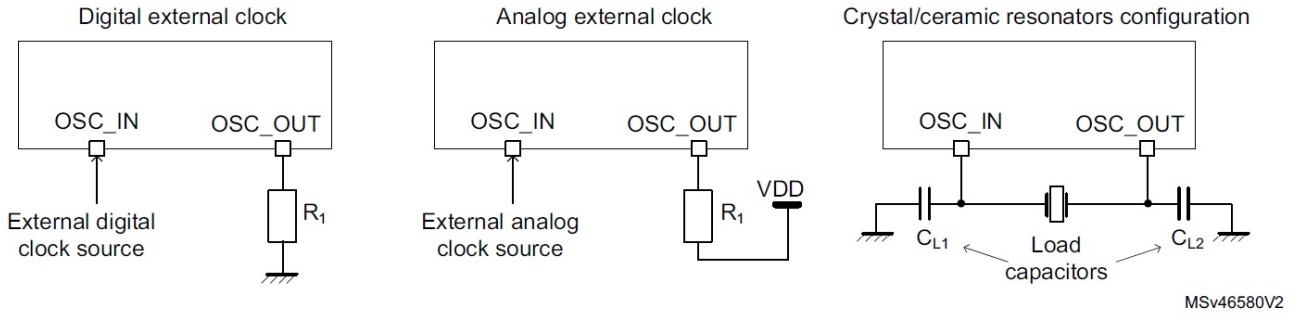
- "st,bypass" configures the external analog clock source (set HSEBYP, LSEBYP),
- "st,digbypass" configures the external digital clock source (set DIGBYP and HSEBYP, LSEBYP),
- "st,css" activates the clock security system (HSECSSON, LSECSSON),
- "st,drive" (LSE only) contains the value of the drive for the oscillator (see LSEDRV_ defined in the file *stm32mp1-clksrc.h*[8]).

3.2.1.2 *DT configuration for HSE*

The HSE can accept an external crystal/ceramic or external clock source on OSC_IN, digital or analog : the user needs to select the correct frequency and the correct configuration in the device tree, corresponding to the hardware setup.

All the ST boards are using a digital external clock configuration (so device tree with = st,digbypass).

For example with the same 24MHz frequency, we have 3 configurations:



- Digital external clock = st,digbypass

```
        / {
                clocks {
                        clk_hse: clk-hse {
```

```
                                #clock-cells = <0>;
                                compatible = "fixed-clock";
                                clock-frequency = <24000000>;
                                st,digbypass;
                        };
        };
```

- Analog external clock = st,bypass

```
        / {
                clocks {
                        clk_hse: clk-hse {
                                #clock-cells = <0>;
                                compatible = "fixed-clock";
                                clock-frequency = <24000000>;
                                st,bypass;
                        };
        };
```

- Crystal/ ceramic resonators configuration

```
        / {
                clocks {
                        clk_hse: clk-hse {
                                #clock-cells = <0>;
                                compatible = "fixed-clock";
                                clock-frequency = <24000000>;
                        };
        };
```

3.2.1.3 *DT configuration for LSE*

Below an example of LSE on board file with 32,768kHz crystal resonators, the drive set to medium high and with activated clock security system.

```
        / {
                clocks {
                        clk_lse: clk-lse {
                                #clock-cells = <0>;
                                compatible = "fixed-clock";
                                clock-frequency = <32768>;
                                st,css;
                                st,drive = <LSEDRV_MEDIUM_HIGH>;
                        };
        };
```

3.2.1.4 *Optional property for "clk-hsi" internal oscillator*

The HSI clock frequency is internally fixed to 64 MHz for the STM32MP15 devices.

In the device tree, clk-hsi is the clock after HSIDIV divider (more information on clk_hsi can be found in the RCC chapter in the reference manual).
As a result the frequency of this fixed clock is used to compute the expected HSIDIV for the clock tree initialization.

Below an example with HSIDIV = 1/1:

```
	/ {
		clocks {
			clk_hsi: clk-hsi {
				#clock-cells = <0>;
				compatible = "fixed-clock";
				clock-frequency = <64000000>;
			};
	};
```

Below an example with HSIDIV = 1/2:

```
	/ {
		clocks {
			clk_hsi: clk-hsi {
				#clock-cells = <0>;
				compatible = "fixed-clock";
				clock-frequency = <32000000>;
			};
	};
```

3.2.1.5 *Clock node example*

An example of clocks node with:

- all oscillators switched on (HSE, HSI, LSE, LSI, CSI)
- HSI at 64MHZ (HSIDIV = 1/1)
- HSE using a digital external clock at 24MHz
- LSE using an external crystal a 32.768kHz (the typical frequency)

We highlight the customized parts:

```
/ {
	clocks {
		clk_hse: clk-hse {
			#clock-cells = <0>;
			compatible = "fixed-clock";
			clock-frequency = <24000000>;
			st,digbypass;
		};

		clk_hsi: clk-hsi {
			#clock-cells = <0>;
			compatible = "fixed-clock";
			clock-frequency = <64000000>;
		};

		clk_lse: clk-lse {
			#clock-cells = <0>;
			compatible = "fixed-clock";
			clock-frequency = <32768>;
		};

		clk_lsi: clk-lsi {
			#clock-cells = <0>;
			compatible = "fixed-clock";
			clock-frequency = <32000>;
		};

		clk_csi: clk-csi {
			#clock-cells = <0>;
```

```
                        compatible = "fixed-clock";
                        clock-frequency = <4000000>;
                };
        };
};
```

So the resulting board device tree, based on SoC device tree "stm32mp157c.dtsi", is :

```
#include "stm32mp157c.dtsi"
&clk_hse {
        clock-frequency = <24000000>;
        st,digbypass;
};

&clk_hsi {
        clock-frequency = <64000000>;
};

&clk_lse {
        clock-frequency = <32768>;
};
```

It is the configuration used by TF-A for STM32MP157C-EV [4]

3.2.2 STM32MP1 clock node

Please refer to clock device tree configuration for information on how to specify the number of cells in a clock specifier.

The bootloader performs a global clock initialization, as described below. The information related to a given board can be found in the board specific device tree files listed in clock node.

The bootloader uses other properties for RCC node ("st,stm32mp1-rcc" compatible):

- secure-status: related to TZEN bit configuration in RCC_TZCR register that allows to restrict RCC and PWR registers write access
- st,clksrc: clock source configuration array
- st,clkdiv: clock divider configuration array
- st,pll: specific PLL configuration
- st,pkcs: peripheral kernel clock distribution configuration array.

All the available clocks are defined as preprocessor macros in *stm32mp1-clks.h*[9] and can be used in device tree sources.

3.2.2.1 Defining clock source distribution with st,clksrc property

This property can be used to configure the clock distribution tree. When used, it must describe the whole distribution tree.

There are nine clock source selectors for the STM32MP15 devices. They must be configured in the following order: MPU, AXI, MCU, PLL12, PLL3, PLL4, RTC, MCO1, and MCO2.

The clock source configuration values are defined by the CLK_<NAME>_<SOURCE> macros located in *stm32mp1-clksrc.h*[8].

Example:

```
        st,clksrc = <
                CLK_MPU_PLL1P
                CLK_AXI_PLL2P
                CLK_MCU_PLL3P
                CLK_PLL12_HSE
```

```
                CLK_PLL3_HSE
                CLK_PLL4_HSE
                CLK_RTC_LSE
                CLK_MCO1_DISABLED
                CLK_MCO2_DISABLED
        >;
```

3.2.2.2 *Defining clock dividers with st,clkdiv property*

This property can be used to configure the value of the clock main dividers. When used, it must describe the whole clock divider tree.

There are 11 dividers values for the STM32MP15 devices. They must be configured in the following order: MPU, AXI, MCU, APB1, APB2, APB3, APB4, APB5, RTC, MCO1 and MCO2.

Each divider value uses the DIV coding defined in the RCC associated register, RCC_xxxDIVR. In most cases, this value is the following:

- 0x0: not divided
- 0x1: division by 2
- 0x2: division by 4
- 0x3: division by 8
- ...

Note that the coding differs for RTC MCO1 and MCO2:

- 0x0: not divided
- 0x1: division by 2
- 0x2: division by 3
- 0x3: division by 4
- ...

Example:

```
        st,clkdiv = <
                1 /*MPU*/
                0 /*AXI*/
                0 /*MCU*/
                1 /*APB1*/
                1 /*APB2*/
                1 /*APB3*/
                1 /*APB4*/
                2 /*APB5*/
                23 /*RTC*/
                0 /*MCO1*/
                0 /*MCO2*/
        >;
```

3.2.2.3 *Defining peripheral PLL frequencies with st,pll property*

This property can be used to configure PLL frequencies.

The PLL children nodes for PLL1 to PLL4 (see reference manual for details) are associated with an index from 0 to 3 (st,pll@0 to st,pll@3). PLLx is off when the associated node is absent.

For ecosystem release v1.2.0 ⓘ, TF-A automatically selects the most suitable operating point for the platform (please refer to How to change the CPU frequency), so the PLL1 node is no longer necessary.

Below the available properties for each PLL node:

- cfg contains the PLL configuration parameters in the following order: DIVM, DIVN, DIVP, DIVQ, DIVR, output.

  DIVx values are defined as in RCC:

  - 0x0: bypass (division by 1)
  - 0x1: division by 2
  - 0x2: division by 3
  - 0x3: division by 4
  - ...

  Output contains a bitfield for each output value (1:ON / 0:OFF)

  - BIT(0) output P : DIVPEN
  - BIT(1) output Q : DIVQEN
  - BIT(2) output R : DIVREN

  Note: PQR(p,q,r) macro can be used to build this value with p, q, r = 0 or 1.

- frac: fractional part of the multiplication factor (optional, when absent PLL is in integer mode).
- csg contains the clock spreading generator parameters (optional) in the following order: MOD_PER, INC_STEP and SSCG_MODE.

  MOD_PER: modulation period adjustment

  INC_STEP: modulation depth adjustment

  SSCG_MODE: Spread spectrum clock generator mode, defined in *stm32mp1-clksrc.h*[8]:

  - SSCG_MODE_CENTER_SPREAD = 0
  - SSCG_MODE_DOWN_SPREAD = 1

Example:

```
        st,pll@0 {
                cfg = < 1 53 0 0 0 1 >;
                frac = < 0x810 >;
        };
        st,pll@1 {
                cfg = < 1 43 1 0 0 PQR(0,1,1) >;
                csg = < 10 20 1 >;
        };
        st,pll@2 {
                cfg = < 2 85 3 13 3 0 >;
                csg = < 10 20 SSCG_MODE_CENTER_SPREAD >;
        };
        st,pll@3 {
                cfg = < 2 78 4 7 9 3 >;
        };
```

3.2.2.4 *Defining peripheral kernel clock tree distribution with st,pkcs property*

This property can be used to configure the peripheral kernel clock selection.

It is a list of peripheral kernel clock source identifiers defined by the CLK_<KERNEL-CLOCK>_<PARENT-CLOCK> macros in the *stm32mp1-clksrc.h*[8] header file.

st,pkcs may not list all the kernel clocks. No specific order is required.

Example:

```
        st,pkcs = <
                CLK_STGEN_HSE
                CLK_CKPER_HSI
                CLK_USBPHY_PLL2P
                CLK_DSI_PLL2Q
                CLK_I2C46_HSI
                CLK_UART1_HSI
                CLK_UART24_HSI
        >;
```

3.2.2.5 *HSI and CSI clocks calibration*

The calibration is an optional feature that can be enabled from the device tree. It allows requesting the HSI or CSI clock calibration by several means:

- SiP SMC service
- Periodic calibration every X seconds
- Interrupt raised by the MCU

This feature requires that a hardware timer is assigned to the calibration sequence.

A dedicated interrupt must be defined using "mcu_sev" name to start a calibration on detection of an interrupt raised by the MCU.

- st,hsi-cal: used to enable HSI clock calibration feature.
- st,csi-cal; used to enable CSI clock calibration feature.
- st,cal-sec: used to enable periodic calibration at specified time intervals from the secure monitor. The time interval must be given in seconds. If not specified, a calibration is only processed for each incoming request.

Example:

```
        &rcc {
                st,hsi-cal;
                st,csi-cal;
                st,cal-sec = <15>;
                secure-interrupts = <GIC_SPI 144 IRQ_TYPE_LEVEL_HIGH>,
                                    <GIC_SPI 145 IRQ_TYPE_LEVEL_HIGH>;
                interrupt-names = "mcu_sev", "wakeup";
        };
```

4 How to configure the DT using STM32CubeMX

The STM32CubeMX tool can be used to configure the STM32MPU device and get the corresponding platform configuration device tree files.

The STM32CubeMX may not support all the properties described in the above DT bindings documentation paragraph.
If so, the tool inserts **user sections** in the generated device tree.
These sections can then be edited to add some properties and they are preserved from one generation to another.

Refer to STM32CubeMX user manual for further information.

5 References

Please refer to the following links for additional information:

- docs/devicetree/bindings/clock/st,stm32mp1-rcc.txt TF-A clock binding information file
- doc/device-tree-bindings/clock/st,stm32mp1.txt U-Boot SPL clock binding information file
- fdts/stm32mp157c.dtsi (for TF-A), arch/arm/dts/stm32mp157c.dtsi (for U-Boot SPL): STM32MP157C device tree files
- 4.04.1 fdts/stm32mp157c-ed1.dts STM32MP157C-EV device tree file for TF-A
- arch/arm/dts/stm32mp157c-ed1-u-boot.dtsi STM32MP157C-EV device tree file for U-Boot SPL
- fdts/stm32mp157a-dk1.dts STM32MP157A-DK device tree file for TF-A
- arch/arm/dts/stm32mp157a-dk1-u-boot.dtsi STM32MP157A-DK device tree file for U-Boot SPL
- 8.08.18.28.3 include/dt-bindings/clock/stm32mp1-clksrc.h (for TF-A), include/dt-bindings/clock/stm32mp1-clksrc.h (for U-Boot SPL): STM32MP1 DT bindings clock source files
- include/dt-bindings/clock/stm32mp1-clks.h (for TF-A), include/dt-bindings/clock/stm32mp1-clks.h (for U-Boot SPL): STM32MP1 DT bindings clock identifier files

Operating System

First Stage Boot Loader

Trusted Firmware for Arm Cortex-A

Secondary Program Loader, *Also known as* ***U-Boot SPL***

Doubledata rate (memory domain)

Second Stage Boot Loader

Device Tree

Generic Interrupt Controller

Serial Peripheral Interface

High Speed Internal oscillator (STM32 clock source) or High Speed Synchronous Serial Interface (MIPI® Alliance standard)

Read Only Memory

High Speed External oscillator (STM32 clock source)

Low Speed External oscillator (STM32 clock source)

Reset and Clock Control

Low Speed Internal oscillator (STM32 clock source)

Multi Speed Internal oscillator (STM32 clock source)

Microprocessor Unit

Microcontroller Unit (MCUs have internal flash memory and are intended to operate with a minimum amount of external support ICs. They commonly are a self-contained, system-on-chip (SoC) designs.)

Real Time Clock

System Time Generator

Display Serial Interface (MIPI® Alliance standard)

Silicon Provider

Secure Monitor Call

Stable: 16.10.2020 - 09:22 / Revision: 16.10.2020 - 08:17

A quality version of this page, approved on *16 October 2020*, was based off this revision.

Contents

1 Framework purpose

The purpose of this article is to explain how to handle the STM32MP15x low-power modes:

- Low-power modes available on the device
- Linux software overview
- How to enter and exit the low-power modes on Arm® Cortex®-A7 core
- How to enter a platform low-power mode

2 Low-power modes available on the device

Refer to STM32MP15 reference manuals for the full description of low-power modes.
The AN5109 low-power application note also gives much more information on these modes, including:

- the detailed description of the operating modes,
- the low-power mode entry and exit sequences,
- the low-power mode control registers.

The modes are handled by the RCC and the PWR peripherals.

The table below summarizes the device hardware states corresponding to each low-power mode.

The term "subsystem" either refers to Arm® Cortex®-A7 (also called MPU) or to Arm® Cortex®-M4 (also called MCU). A mode prefixed by 'C' corresponds to a subsystem mode.

A platform mode is the combination of MPU and MCU modes.

Level	Mode	Vddcore state	Clocks state
Subsystem	MPU CRun	on	on
	MPU CStop	on	Subsystem off
	MPU CStandby	on	Subsystem off
	MCU CRun	on	on
	MCU CStop	on	Subsystem off

MPU mode	MCU mode	Platform mode	Vddcore state	Clocks state
CRun	CRun	Run	On	On
CStop	CRun	Run	On	On
CStandby	CRun	Run	On	On
CRun	CStop	Run	On	On
CStop	CStop	Stop/LPLV-Stop/Standby	On/Retention/Off	Off/Off/Off
CStandby	CStop	Stop/LPLV-Stop/Standby	On/Retention/Off	Off/Off/Off

2.1 Wakeup sources

The above modes are exited due to a wakeup event.

Again, the AN5109 low-power application note details, among other things, the wakeup sources, the software mechanism that ensures the consistency between the low-power mode and the activated wakeup source, and the low-power mode exit sequence.

The following table gives the list of wakeup sources available in each mode.

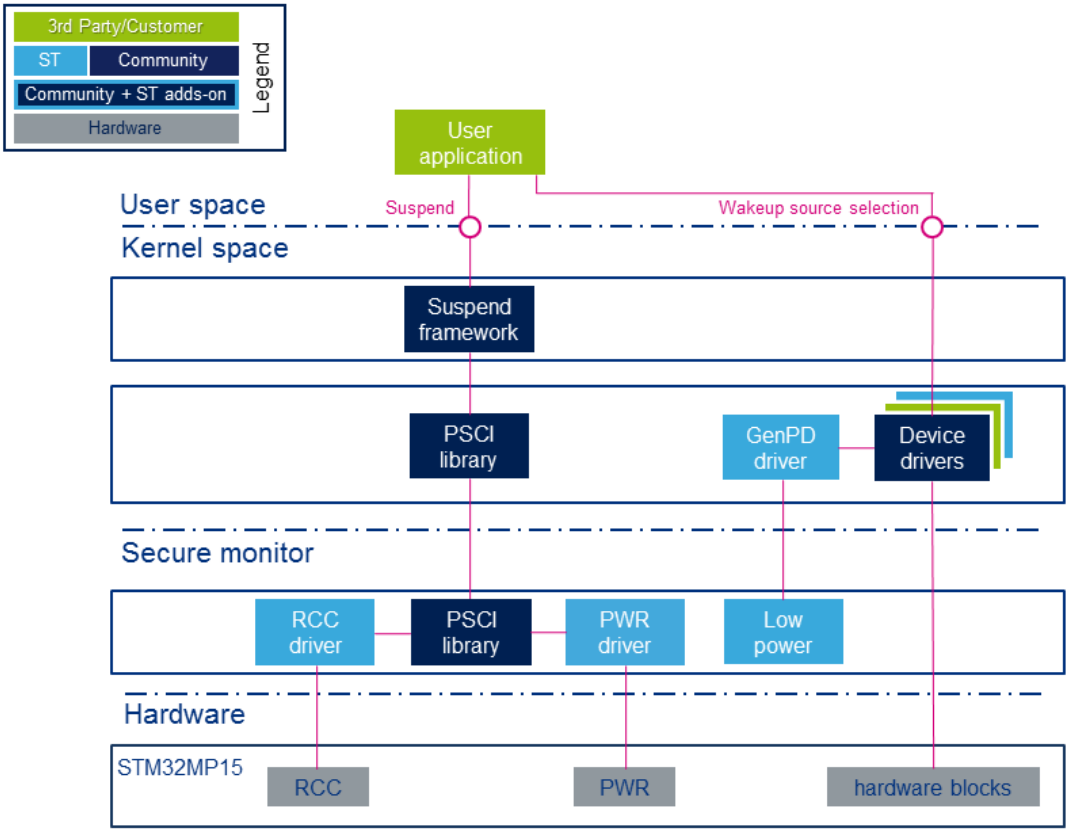
Mode	Available wakeup sources
CStop	BOR, PVD, AVD, Vbat mon, Temp mon, LSE CSS, RTC, TAMP, USB, CEC, ETH, USA

Mode	Available wakeup sources
/CStandby /Stop	RT, I²C, SPI, LPTIM, IWDG, GPIO, Wakeup pins (from PWR)
LPLV-Stop	BOR, PVD, AVD, Vbat mon, Temp mon, LSE CSS, RTC, TAMP, IWDG, GPIO, Wakeup pins (from PWR)
Standby	BOR, Vbat mon, Temp mon, LSE CSS, RTC, TAMP, IWDG, Wakeup pins (from PWR)

3 Software overview

The Linux® suspend framework is used to trigger a low-power mode entry/exit sequence.

Refer to Documentation/power for more details.



The user application issues a suspend request to the kernel. This request is handled by the suspend Framework, which notifies all the device drivers to prepare for low-power entry. It then calls the PSCI service.

In addition to this centralized suspend process, most of the drivers implement the runtime pm feature. It is used to dynamically disable the resources of the peripherals (clocks and power when applicable) in case of inactivity (see Documentation/power /runtime_pm.txt).

3.1 Component description

Kernel components:

- **Suspend framework**: this framework schedules the overall sequence by stopping all the ongoing tasks
- **GenPD driver**: this driver is used for low-power mode selection according to the activated wakeup sources.
- **PSCI library**: this is a set of standardized functions to request a low-power service to the secure monitor
- **RCC driver**: this driver handles the circuit non-secure clocks

Secure monitor components:

- **PWR driver**: this driver is responsible for configuring the low-power mode
- **PSCI library**: this is a set of standardized functions handling the low-power services
- **Low power driver**: the role of this driver is to choose the low-power mode according to the programmed wakeup source(s)
- **RCC driver**: this driver handles the circuit secure clocks

3.2 API description

The suspend process is triggered from the user space through standard commands.

The system sleep control file is the *state* file, located under: /sys/power/

Only the 'mem' command is supported:

- The whole system activity is stopped and a low-power mode is entered. The software selects the deepest mode according to the activated wakeup source(s).

```
Example: Board $> echo mem > /sys/power/state
```

Further details can be found in Documentation/power/interface.txt

STMicroelectronics deliveries propose a default mapping of the low-power modes for each type of board.

Note that this default mapping can be changed thanks to the device tree. Refer to paragraph 3.3.2.

3.3 Software configuration

3.3.1 Menuconfig (Linux® kernel)

The suspend to RAM feature is activated by default in STMicroelectronics deliveries.

It can be deactivated through the kernel menuconfig using Power management options/Suspend to RAM and standby: Menuconfig or how to configure kernel .

3.3.2 Device tree (secure monitor)

The default low-power mode mapping can be modified through the secure monitor device tree.

Below an example:

```
&pwr {
    system_suspend_supported_modes = <
        STM32_PM_CSLEEP_RUN
        STM32_PM_CSTOP_ALLOW_STOP
```

```
            STM32_PM_CSTOP_ALLOW_LP_STOP
            STM32_PM_CSTOP_ALLOW_LPLV_STOP
            STM32_PM_CSTOP_ALLOW_STANDBY_DDR_SR
        >;
        system_off_soc_mode = <STM32_PM_CSTOP_ALLOW_STANDBY_DDR_OFF>;
    };
```

For detailed information on the device tree concept, refer to Device tree.

3.3.3 Example of wakeup source activation

The activation of a wakeup source is done in the corresponding driver.

For example, activating UART4 as wakeup source is done thanks to the following commands:

```
Board $> echo enabled > /sys/devices/platform/soc/40010000.serial/tty/ttySTM0/power/wakeup
Board $> echo enabled > /sys/devices/platform/soc/40010000.serial/power/wakeup
```

It is possible to check the state of each wakeup source (activated or not) by displaying the 'wakeup' attribute.

Note that the software implements a consistency check between the selected wakeup source and the appropriate low-power mode.

4 How to enter and exit low-power modes

4.1 Platform low-power

Select the platform allowed modes depending on the required wakeup source.

Activate the wakeup source(s) (peripheral dependent).

Call the low-power mode on both sides (MPU and MCU).

4.2 MPU side

Activate the wakeup source(s) (peripheral dependent)

Call the low-power mode by issuing the following command:

echo mem > /sys/power/state

```
  Note that in Weston configuration the low-power mode is entered upon a 'systemctl
suspend' command.
```

4.3 MCU side

Please refer to Coprocessor power management for Arm® Cortex®-M4 commands.

4.4 Example: entering/exiting MPU CStop mode

Enable at least one wakeup source from table 2.1 in CStop category, for example USART:

```
Board $> echo enabled > /sys/devices/platform/soc/40010000.serial/tty/ttySTM0/power/wakeup
Board $> echo enabled > /sys/devices/platform/soc/40010000.serial/power/wakeup
```

Call the low-power entry:

```
Board $> echo mem > /sys/power/state
```

or for the Weston configuration:

```
Board $> cat /etc/systemd/sleep.conf
[Sleep]
SuspendMode=
HibernateMode=
HybridSleepMode=
SuspendState=mem
HibernateState=mem
HybridSleepState=mem
```

```
 Board $> systemctl suspend
```

The MPU is now in CStop mode, and can be woken up by sending a character to the console.

5 How to trace and debug

The suspend/resume process execution is logged in the MPU console. It gives useful information on the platform state (sleeping or active).

```
root@stm32mp1:~# echo mem > /sys/power/state
[ 1072.267571] PM: suspend entry (deep)
[ 1072.269687] PM: Syncing filesystems ... done.
[ 1072.279114] Freezing user space processes ... (elapsed 0.008 seconds) done.
[ 1072.292835] OOM killer disabled.
[ 1072.296046] Freezing remaining freezable tasks ... (elapsed 0.001 seconds) done.
[ 1072.303431] Suspending console(s) (use no_console_suspend to debug)
[ 1072.332520] dwc2 49000000.usb-otg: suspending usb gadget configfs-gadget
[ 1072.332537] dwc2 49000000.usb-otg: dwc2_hsotg_ep_disable: called for ep0
[ 1072.332546] dwc2 49000000.usb-otg: dwc2_hsotg_ep_disable: called for ep0
[ 1072.468536] Disabling non-boot CPUs ...
[ 1072.507876] CPU1 killed.
[ 1072.509635] Enabling non-boot CPUs ...
[ 1072.510508] CPU1 is up
[ 1072.527553] dwmac4: Master AXI performs any burst length
[ 1072.527583] stm32-dwmac 5800a000.ethernet eth0: No Safety Features support found
[ 1072.527621] stm32-dwmac 5800a000.ethernet eth0: ERROR failed to create debugfs
directory
[ 1072.527631] stm32-dwmac 5800a000.ethernet eth0: stmmac_hw_setup: failed debugFS
registration
[ 1072.588234] dwc2 49000000.usb-otg: resuming usb gadget configfs-gadget
[ 1072.738469] OOM killer enabled.
[ 1072.741575] Restarting tasks ... done.
[ 1072.752596] PM: suspend exit
```

It is also possible to monitor the hardware signals related to the system low-power modes thanks to the HDP internal peripheral. Please refer to HDP Linux driver for its configuration.

6 To go further

Refer to STM32MP15 reference manuals for a detailed description of low-power modes and peripheral wakeup sources.

The AN5109 low power application note gives additional information on the hardware settings used for low-power management.

Microprocessor Unit

Microcontroller Unit (MCUs have internal flash memory and are intended to operate with a minimum amount of external support ICs. They commonly are a self-contained, system-on-chip (SoC) designs.)

Brownout reset

Programmable Voltage Detector

Analog Voltage Detector

Low Speed External oscillator (STM32 clock source)

Cascading Style Sheets (web standard)

Real Time Clock

Tamper

Consumer Electronics Control (HDMI standard)

Ethernet

Universal Synchronous/Asynchronous Receiver/Transmitter

Serial Peripheral Interface

low-power timer (STM32 specific)

Independent Watchdog

General-Purpose Input/Output (A realization of open ended transmission between devices on an embedded level. These pins available on a processor can be programmed to be used to either accept input or provide output to external devices depending on user desires and applications requirements.)

Power State Coordination Interface

Reset and Clock Control

Application programming interface

Random Access Memory (Early computer memories generally hadserial access. Memories where any given address can be accessed when desired were then called "random access" to distinguish them from the memories where contents can only be accessed in a fixed order. The term is used today forvolatile random-accesssemiconductor memories.)

Low Power (MIPI® Alliance DSI standard)

Doubledata rate (memory domain)

Out Of Memory

Configuration File System (See https://en.wikipedia.org/wiki/Configfs for more details)

Debug File System (See https://en.wikipedia.org/wiki/Debugfs for more details)

Stable: 04.02.2020 - 15:40 / Revision: 04.02.2020 - 15:27

A quality version of this page, approved on *4 February 2020*, was based off this revision.

Contents

1 Article purpose

The purpose of this article is to:

- briefly introduce the RCC peripheral and its main features
- indicate the level of security supported by this hardware block
- explain, when necessary, how to configure the RCC peripheral.

2 Peripheral overview

The **RCC** peripheral is used to control the internal peripherals, as well as the **reset** signals and **clock** distribution. The RCC gets several internal (LSI, HSI and CSI) and external (LSE and HSE) clocks. They are used as clock sources for the hardware blocks, either directly or indirectly, via the four PLLs (PLL1, PLL2, PLL3 and PLL4) that allow to achieve high frequencies.

2.1 Features

Refer to the STM32MP15 reference manuals for the complete list of features, and to the software components, introduced below, to see which features are really implemented.

2.2 Security support

The RCC is a **secure** peripheral. There are two levels of security, which are controlled via two bits in the RCC_TZCR register (only accessible in secure mode):

- **TZEN** allows to set some RCC registers in secure mode, in particular registers for configuring PLL1 and PLL2, in order to secure a TrustZone perimeter for the Cortex®-A7 secure core and its peripherals.
- **MCKPROT** allows extending the TZEN secure clock control perimeter to PLL3 and to the MCU subsystem, so to the Cortex® -M4 and its bus clock.

Please note that all RCC registers can be read from the non-secure world.

3 Peripheral usage and associated software

3.1 Boot time

The RCC security level differs for each boot chain:

- the trusted boot chain sets TZEN to 1 an MCKPROT to 0
- the basic boot chain sets TZEN to 0 an MCKPROT to 0

The RCC is used by all the boot components: the ROM code, the FSBL, the SSBL and up to the Linux® kernel. Nevertheless, the main initialization step is performed by the FSBL that is responsible for the clock tree initialization: it consists in configuring all the input clocks, the PLL and the clock sources that are selected as kernel clocks for all peripherals. The whole configuration is carried out by the device tree.

The STM32CubeMX tool allows configuring in one place the clock tree that will be applied at boot time and used at runtime, so it is highly recommended to use it to generate your device tree. Moreover, the STM32CubeMX integrates all the information documented in the STM32MP15 reference manuals, making this configuration step straighforward.

3.2 Runtime

3.2.1 Overview

The RCC peripheral is shared at runtime:

- the Arm® Cortex®-A7 secure core controls all the secure registers (refer to TZEN and MCKPROT bit descriptions) through the RCC OP-TEE driver. The access to some secure registers from the Cortex®-A7 non-secure core can be achieved via runtime secure services implemented in the secure monitor (from the OP-TEE if it is present, otherwise from the TF-A).
- the Arm® Cortex®-A7 non-secure core controls the clock management via the clock framework, and the reset management via the reset framework in Linux®.
- the Arm® Cortex®-M4 core controls all the clock and reset managements in STM32Cube with the RCC HAL driver

Concurrent control from each context is possible because the above managements are performed via independent registers.

3.2.2 Software frameworks

Do	Pe	Software frameworks	Comment
Cortex-A7	Cortex-A7	Cortex-M4	

Do	Pe	Software frameworks			Comment
secure (OP-TEE)	non-secure (Linux)	(STM32Cube)			
Power & Thermal	RCC	OP-TEE RCC driver	Reset framework Clock framework	STM32Cube RCC driver	

3.2.3 Peripheral configuration

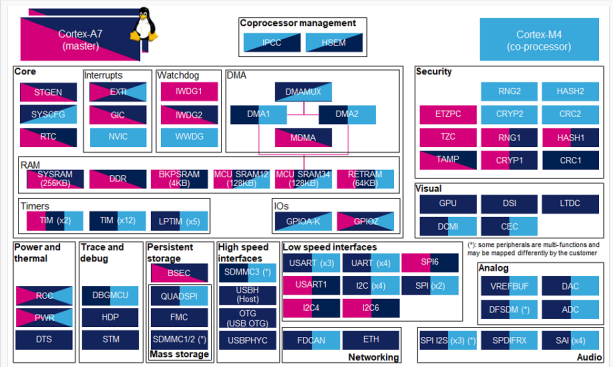
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration can be done alone via the STM32CubeMX tool for all internal peripherals, and then manually completed (particularly for external peripherals), according to the information given in the corresponding software framework article.

3.2.4 Peripheral assignment

Check boxes illustrate the possible peripheral allocations supported by STM32 MPU Embedded Software:

- means that the peripheral can be assigned () to the given runtime context.
- is used for system peripherals that cannot be unchecked because they are statically connected in the device.

Refer to How to assign an internal peripheral to a runtime context for more information on how to assign peripherals manually or via STM32CubeMX.
The present chapter describes STMicroelectronics recommendations or choice of implementation. Additional possiblities might be described in STM32MP15 reference manuals .



Internal peripherals

D	P	Runtime allocation			Comment
domain	per Cortex-A7 secure Cortex-A7 non-secure				

Domain	Peripheral (OP-TEE)	Runtime allocation				Comment
		Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Power & Thermal	RCC	RCC				

4 How to go further

The RCC is interfaced with the HDP internal peripheral, thus offering the flexibility to monitor the main RCC state signals on the debug pins.

Please refer to the STM32MP15 reference manuals for the full list of signals that can be monitored.

5 References

Reset and Clock Control

Low Speed Internal oscillator (STM32 clock source)

High Speed Internal oscillator (STM32 clock source) or High Speed Synchronous Serial Interface (MIPI® Alliance standard)

Multi Speed Internal oscillator (STM32 clock source)

Low Speed External oscillator (STM32 clock source)

High Speed External oscillator (STM32 clock source)

Microcontroller Unit (MCUs have internal flash memory and are intended to operate with a minimum amount of external support ICs. They commonly are a self-contained, system-on-chip (SoC) designs.)

Read Only Memory

First Stage Boot Loader

Second Stage Boot Loader

Open Portable Trusted Execution Environment

Stable: 31.01.2020 - 13:04 / Revision: 31.01.2020 - 13:02

A quality version of this page, approved on *31 January 2020*, was based off this revision.

1 STM32CubeMX overview

This article describes STM32CubeMX, an official STMicroelectronics graphical software configuration tool.

The STM32CubeMX application helps developers to use the STM32 by means of a user interface, and guides the user through to the initial configuration of a firmware project.

It provides the means to:

- configure pin assignments, the clock tree, or internal peripherals
- simulate the power consumption of the resulting project
- configure and tune DDR parameters
- generate HAL initialization code for Cortex-M4
- generate the Device Tree for a Linux kernel, TF-A and U-Boot firmware for Cortex-A7

It uses a rich library of data from the STM32 microcontroller portfolio.

The application is intended to ease the initial development phase by helping developers to select the best product in terms of features and power.

2 STM32CubeMX main features

- Peripheral and middleware parameters

  Presents options specific to each supported software component

- Peripheral assignment to processors

  Allows assignment of each peripheral to Cortex-A Secure, Cortex-A Non-Secure, or Cortex-M processors

- Power consumption calculator

  Uses a database of typical values to estimate power consumption, DMIPS, and battery life

- Code generation

  Makes code regeneration possible, while keeping user code intact

- Pinout configuration

  Enables peripherals to be chosen for use, and assigns GPIO and alternate functions to pins

- Clock tree initialization

  Chooses the oscillator and sets the PLL and clock dividers

- DDR tunning tool

  Ensures the configuration, testing, and tuning of the MPU DDR parameters

3 How to get STM32CubeMX

Please, refer to the following link STM32CubeMX to find STM32CubeMX, the Release Note, the User Manual and the product specification.

Doubledata rate (memory domain)

Hardware Abstraction Layer

Trusted Firmware for Arm Cortex-A

Das U-Boot -- the Universal Boot Loader (see U-Boot_overview)

General-Purpose Input/Output (A realization of open ended transmission between devices on an embedded level. These pins available on a processor can be programmed to be used to either accept input or provide output to external devices depending on user desires and applications requirements.)

Microprocessor Unit

Stable: 11.02.2020 - 13:06 / Revision: 11.02.2020 - 12:59

A quality version of this page, approved on *11 February 2020*, was based off this revision.

All the peripherals receive one or several clocks that are generated via RCC internal peripheral. RCC relies on several clocks sources (LSI, LSE, HSI, HSE, CSI) and four PLL in order to provide adequate input frequencies to all the peripherals. The clock tree covers all the system clock distribution aspects, from the clock sources to the consumer peripherals (internal and external), except clock gating management that is locally controlled by each peripheral driver.

Contents

1 Overview

The clock tree is managed via RCC internal peripheral hardware block and it is configured at different steps from the Cortex-A7:

- When the device is **reset**, all RCC registers take their reset values: the four PLL are disabled and most of the clock source selectors are pointing to the HSI.
- The ROM Code configures the minimum clock tree needed to boot on the selected boot device.
- The **FSBL** (Cf. Boot chains overview) completely configures the clock tree as expected, all the way up to Linux, thanks to the configuration given in the device tree.
  - U-Boot SPL (basic boot chain) binding is available in doc/device-tree-bindings/clock/st,stm32mp1.txt
  - TF-A (trusted boot chain) binding follows the same syntax as U-Boot SPL
- **Linux** is responsible for the clock tree at runtime and it may partly modify it thanks to information taken from the device tree:
  - Linux binding is available in Documentation/devicetree/bindings/clock/clock-bindings.txt (and surrounding files): 'fixed-clock' compatible, 'clocks' and 'assigned-clocks' properties are important concepts to understand the management of clocks providers/consumers.

Notice that STM32Cube (running on the Cortex-M4) does not control the clock tree setup so the configuration applied by the Cortex-A7 must provide suitable clocks to Cortex-M4, like it is explained in resource manager description. One exception to this rule is the engineering boot, allowing to directly load (and debug) the Cortex-M4 whereas the Cortex-A7 execution is stalled in the ROM code.

2 How to build a clock tree?

Building a clock tree is quite complex as it needs to take into account the constraints set by each internal and external peripheral, including external clock sources.

We encourage the use of STM32CubeMX to build the clock tree, and avoid having to know all internal peripherals details: the tool allows to select the peripherals that will be present on the board, fix the clock sources frequencies and automatically find an optimized clock tree. It is then able to generate the device tree that is directly consumed by the boot chain and Linux.

3 ST boards clock tree

3.1 For ecosystem release v1.1.0

3.1.1 STM32MP157C-EV1 case

This chapter shows the boot time clock tree set by the FSBL on STM32MP157C-EV1 evaluation board. Linux eventual runtime modifications are not covered here.

3.1.1.1 Clock tree

The following table shows what STM32MP157C-EV1 clock tree looks like, as a result of the boot chain execution with the device tree built with STM32CubeMX.

Component	Parent	Frequency	Used?	Comment
LSI	N.A.	0.032000 MHz	yes	Mandatory for IWDG, DAC
DAC	LSI	0.032000 MHz	yes	
RNG1	LSI	0.032000 MHz	yes	Input frequency should be as low as possible
RNG2	LSI	0.032000 MHz	yes	Input frequency should be as low as possible
IWDG1	LSI	0.032000 MHz	yes	
IWDG2	LSI	0.032000 MHz	yes	
LSE	N.A.	0.032768 MHz	yes	Mandatory for DTS
RTC	LSE	0.032768 MHz	yes	
TAMP	LSE	0.032768 MHz	yes	
CEC	LSE	0.032768 MHz	yes	
LPTIM4	LSE	0.032768 MHz	yes	
LPTIM5	LSE	0.032768 MHz	yes	
HSI	N.A.	64.000000 MHz	yes	
SPI4	HSI	64.000000 MHz	no	
SPI5	HSI	64.000000 MHz	no	
SPI6	HSI	64.000000 MHz	yes	
I2C4	HSI	64.000000 MHz	yes	PMIC
I2C6	HSI	64.000000 MHz	no	
I2C1	HSI	64.000000 MHz	no	
I2C2	HSI	64.000000 MHz	yes	Rpi and peripherals
I2C3	HSI	64.000000 MHz	no	
I2C5	HSI	64.000000 MHz	yes	Rpi
USART1	HSI	64.000000 MHz	yes	
USART2	HSI	64.000000 MHz	no	
USART3	HSI	64.000000 MHz	yes	Rpi
UART4	HSI	64.000000 MHz	yes	Linux console
UART5	HSI	64.000000 MHz	no	
USART6	HSI	64.000000 MHz	no	
UART7	HSI	64.000000 MHz	no	
UART8	HSI	64.000000 MHz	no	
MCO1	HSI	64.000000 MHz	no	Available so can be used
HSE	N.A.	24.000000 MHz	yes	
DSIPLL	HSE	125.000000 MHz	no	DSI DPHY PLL
DSIBL	DSIPLL	125.000000 MHz	no	DSI lanebyte clock
RTCDIV	HSE	1.000000 MHz	yes	Only used when RTC source is HSE
ck_per	HSE	24.000000 MHz	yes	
ADC	ck_per	24.000000 MHz	yes	Can use internal divider to be < 40MHz

```
  PLL1                |HSE       | 1300.000000 MHz |  yes   |
    PLL1P             |PLL1      |  650.000000 MHz |  yes   |
      MPUDIV          |PLL1P     |  325.000000 MHz |  yes   |
      Cortex-A7       |PLL1P     |  650.000000 MHz |  yes   |
        MCO2          |Cortex-A7 |  650.000000 MHz |  no    | Available so can be used
  PLL2                |HSE       | 1066.000000 MHz |  yes   |
    PLL2P             |PLL2      |  266.500000 MHz |  yes   |
      AXI             |PLL2P     |  266.500000 MHz |  yes   | < 266MHz
        FMC           |AXI       |  266.500000 MHz |  yes   | NAND flash
        QSPI          |AXI       |  266.500000 MHz |  yes   | NOR flash
        SYSRAM        |AXI       |  266.500000 MHz |  yes   |
        ROM           |AXI       |  266.500000 MHz |  yes   |
        AHB5          |AXI       |  266.500000 MHz |  yes   | < 266MHz
          CRYP1       |AHB5      |  266.500000 MHz |  yes   |
          HASH1       |AHB5      |  266.500000 MHz |  yes   |
          GPIOZ       |AHB5      |  266.500000 MHz |  yes   |
          BKPSRAM     |AHB5      |  266.500000 MHz |  yes   |
        AHB6          |AXI       |  266.500000 MHz |  yes   | < 266MHz
          CRC1        |AHB6      |  266.500000 MHz |  yes   |
          MDMA        |AHB6      |  266.500000 MHz |  yes   |
          USBH        |AHB6      |  266.500000 MHz |  yes   | USB Host
          APB4        |AHB6      |  133.250000 MHz |  yes   |
          APB5        |AHB6      |   66.625000 MHz |  yes   |
            BSEC      |APB5      |   66.625000 MHz |  yes   | < 67MHz
            ETZPC     |APB5      |   66.625000 MHz |  yes   |
            TZC       |APB5      |   66.625000 MHz |  yes   |
        DBGAPB        |AXI       |  133.250000 MHz |  yes   | JTAG & Coresight
          DBGMCU      |DBGAPB    |   66.625000 MHz |  yes   |
          STM         |DBGAPB    |   66.625000 MHz |  yes   |
    PLL2Q             |PLL2      |  533.000000 MHz |  yes   |
      GPU             |PLL2Q     |  533.000000 MHz |  yes   | < 533MHz
    PLL2R             |PLL2      |  533.000000 MHz |  yes   |
      DDR             |PLL2R     |  533.000000 MHz |  yes   | < 533MHz
  PLL3                |HSE       |  417.755859 MHz |  yes   |
    PLL3P             |PLL3      |  208.877930 MHz |  yes   |
      MLAHB           |PLL3P     |  208.877930 MHz |  yes   | < 209MHz
        Cortex-M4     |MLAHB     |  208.877930 MHz |  yes   |
        SRAM1         |MLAHB     |  208.877930 MHz |  yes   |
        SRAM2         |MLAHB     |  208.877930 MHz |  yes   |
        SRAM3         |MLAHB     |  208.877930 MHz |  yes   |
        RETRAM        |MLAHB     |  208.877930 MHz |  yes   |
        AHB1          |MLAHB     |  208.877930 MHz |  yes   | < 209MHz
        AHB2          |MLAHB     |  208.877930 MHz |  yes   | < 209MHz
          DMA1        |AHB2      |  208.877930 MHz |  yes   |
          DMA2        |AHB2      |  208.877930 MHz |  yes   |
          DMAMUX      |AHB2      |  208.877930 MHz |  yes   |
        APB1          |MLAHB     |  104.438965 MHz |  yes   |
          LPTIM1      |APB1      |  104.438965 MHz |  yes   |
          WWDG        |APB1      |  104.438965 MHz |  yes   |
        APB2          |MLAHB     |  104.438965 MHz |  yes   |
        TIM2          |MLAHB     |  208.877930 MHz |  yes   | TIM Group 1
        TIM3          |MLAHB     |  208.877930 MHz |  yes   | TIM Group 1
        TIM4          |MLAHB     |  208.877930 MHz |  yes   | TIM Group 1
        TIM5          |MLAHB     |  208.877930 MHz |  yes   | TIM Group 1
        TIM6          |MLAHB     |  208.877930 MHz |  yes   | TIM Group 1
        TIM7          |MLAHB     |  208.877930 MHz |  yes   | TIM Group 1
        TIM12         |MLAHB     |  208.877930 MHz |  yes   | TIM Group 1
        TIM13         |MLAHB     |  208.877930 MHz |  yes   | TIM Group 1
        TIM14         |MLAHB     |  208.877930 MHz |  yes   | TIM Group 1
        TIM1          |MLAHB     |  208.877930 MHz |  yes   | TIM Group 2
        TIM8          |MLAHB     |  208.877930 MHz |  yes   | TIM Group 2
        TIM15         |MLAHB     |  208.877930 MHz |  yes   | TIM Group 2
        TIM16         |MLAHB     |  208.877930 MHz |  yes   | TIM Group 2
        TIM17         |MLAHB     |  208.877930 MHz |  yes   | TIM Group 2
        APB3          |MLAHB     |  104.438965 MHz |  yes   |
          LPTIM2      |APB3      |  104.438965 MHz |  yes   |
          LPTIM3      |APB3      |  104.438965 MHz |  yes   |
```

```
          SYSCFG        |APB3      |   104.438965 MHz |   yes    |
          VREFBUF       |APB3      |   104.438965 MHz |   yes    |
          DTS           |APB3      |   104.438965 MHz |   yes    |
          HDP           |APB3      |   104.438965 MHz |   yes    |
        AHB3            |MLAHB     |   208.877930 MHz |   yes    | < 209MHz
          CRC2          |AHB3      |   208.877930 MHz |   yes    |
          CRYP2         |AHB3      |   208.877930 MHz |   yes    |
          HASH2         |AHB3      |   208.877930 MHz |   yes    |
          DCMI          |AHB3      |   208.877930 MHz |   yes    | Camera
          IPCC          |AHB3      |   208.877930 MHz |   yes    | Mailbox
        AHB4            |MLAHB     |   208.877930 MHz |   yes    | < 209MHz
          PWR           |AHB4      |   208.877930 MHz |   yes    |
          RCC           |AHB4      |   208.877930 MHz |   yes    |
          GPIOA-K       |AHB4      |   208.877930 MHz |   yes    |
          EXTI          |AHB4      |   208.877930 MHz |   yes    |
    PLL3Q               |PLL3      |    24.573874 MHz |   yes    |
      SPI1              |PLL3Q     |    24.573874 MHz |   no     |
      SPI2              |PLL3Q     |    24.573874 MHz |   no     |
      SPI3              |PLL3Q     |    24.573874 MHz |   no     |
      DFSDM             |PLL3Q     |    24.573874 MHz |   yes    | Digital micro
      SAI1              |PLL3Q     |    24.573874 MHz |   no     |
      SAI2              |PLL3Q     |    24.573874 MHz |   yes    | AudCodec 48kHz (use PLL3R
for 44.1kHz)
      SAI3              |PLL3Q     |    24.573874 MHz |   no     |
      SAI4              |PLL3Q     |    24.573874 MHz |   yes    | SPDIF TX 48kHz (use PLL3R
for 44.1kHz)
    PLL3R               |PLL3      |    11.290699 MHz |   yes    |
  PLL4                  |HSE       |   594.000000 MHz |   yes    |
    PLL4P               |PLL4      |    99.000000 MHz |   yes    |
      SDMMC1            |PLL4P     |    99.000000 MHz |   yes    | µSD card
      SDMMC2            |PLL4P     |    99.000000 MHz |   yes    | eMMC
      SDMMC3            |PLL4P     |    99.000000 MHz |   yes    |
      SPDIF             |PLL4P     |    99.000000 MHz |   yes    | SPDIF RX
    PLL4Q               |PLL4      |    74.250000 MHz |   yes    |
      LCD               |PLL4Q     |    74.250000 MHz |   yes    | LTDC & DSI display pixel
clock
    PLL4R               |PLL4      |    74.250000 MHz |   yes    |
      FDCAN             |PLL4R     |    74.250000 MHz |   yes    | Should be as high as
possible and < 100MHz
  STGEN                 |HSE       |    24.000000 MHz |   yes    |
  USBPHYC               |HSE       |    24.000000 MHz |   yes    | USB PHY Ctrl for USB Host
and OTG
    USBPLL              |USBPHYC   |    48.000000 MHz |   yes    |
      USBO              |USBPLL    |    48.000000 MHz |   yes    | USB OTG
CSI                     |N.A.      |     4.000000 MHz |   yes    | Mandatory for IO
compensation
ETH                     |N.A.      |     0.000000 MHz |   no     | ETH clocked by RGMII PHY
------------------------|----------|------------------|----------
|-----------------------------------|
```

3.1.1.2 *Device tree*

Here are the corresponding device tree **rcc_clk** sub node properties consumed by the first stage boot loader (FSBL) to configure the clock tree above:

```
        st,clksrc = <
                CLK_MPU_PLL1P
                CLK_AXI_PLL2P
                CLK_MCU_PLL3P
                CLK_PLL12_HSE
                CLK_PLL3_HSE
                CLK_PLL4_HSE
                CLK_RTC_LSE
                CLK_MCO1_DISABLED
```

```
                CLK_MCO2_DISABLED
        >;

        st,clkdiv = <
                1 /*MPU*/
                0 /*AXI*/
                0 /*MCU*/
                1 /*APB1*/
                1 /*APB2*/
                1 /*APB3*/
                1 /*APB4*/
                2 /*APB5*/
                23 /*RTC*/
                0 /*MCO1*/
                0 /*MCO2*/
        >;

        st,pkcs = <
                CLK_CKPER_HSE
                CLK_FMC_ACLK
                CLK_QSPI_ACLK
                CLK_ETH_DISABLED
                CLK_SDMMC12_PLL4P
                CLK_DSI_DSIPLL
                CLK_STGEN_HSE
                CLK_USBPHY_HSE
                CLK_SPI2S1_PLL3Q
                CLK_SPI2S23_PLL3Q
                CLK_SPI45_HSI
                CLK_SPI6_HSI
                CLK_I2C46_HSI
                CLK_SDMMC3_PLL4P
                CLK_USBO_USBPHY
                CLK_ADC_CKPER
                CLK_CEC_LSE
                CLK_I2C12_HSI
                CLK_I2C35_HSI
                CLK_UART1_HSI
                CLK_UART24_HSI
                CLK_UART35_HSI
                CLK_UART6_HSI
                CLK_UART78_HSI
                CLK_SPDIF_PLL4P
                CLK_FDCAN_PLL4R
                CLK_SAI1_PLL3Q
                CLK_SAI2_PLL3Q
                CLK_SAI3_PLL3Q
                CLK_SAI4_PLL3Q
                CLK_RNG1_LSI
                CLK_RNG2_LSI
                CLK_LPTIM1_PCLK1
                CLK_LPTIM23_PCLK3
                CLK_LPTIM45_LSE
        >;

        /* VCO = 1300.0 MHz => P = 650 (CPU) */
        pll1: st,pll@0 {
                cfg = < 2 80 0 0 0 PQR(1,0,0) >;
                frac = < 0x800 >;
                u-boot,dm-pre-reloc;
        };

        /* VCO = 1066.0 MHz => P = 266 (AXI), Q = 533 (GPU), R = 533 (DDR) */
        pll2: st,pll@1 {
                cfg = < 2 65 1 0 0 PQR(1,1,1) >;
                frac = < 0x1400 >;
                u-boot,dm-pre-reloc;
```

```
        };

        /* VCO = 417.8 MHz => P = 209, Q = 25, R = 11 */
        pll3: st,pll@2 {
                cfg = < 1 33 1 16 36 PQR(1,1,1) >;
                frac = < 0x1a04 >;
                u-boot,dm-pre-reloc;
        };

        /* VCO = 594.0 MHz => P = 99, Q = 74, R = 74 */
        pll4: st,pll@3 {
                cfg = < 3 98 5 7 7 PQR(1,1,1) >;
                u-boot,dm-pre-reloc;
        };
```

3.1.2 STM32MP157C-DK2 case

This chapter shows the boot time clock tree set by the FSBL on STM32MP157C-DK2 DISCO board. Linux eventual runtime modifications are not covered here.

3.1.2.1 Clock tree

The following table shows what STM32MP157C-DK2 clock tree looks like, as a result of the boot chain execution with the device tree built with STM32CubeMX.

```
Component               | Parent  |   Frequency     | Used?  |
Comment                          |
-----------------------|---------|-----------------|--------
|-----------------------------------|
LSI                     |N.A.     |    0.032000 MHz |  yes   | Mandatory for IWDG, DAC
  DAC                   |LSI      |    0.032000 MHz |  no    |
  RNG1                  |LSI      |    0.032000 MHz |  yes   | Input frequency should be
as low as possible
  RNG2                  |LSI      |    0.032000 MHz |  yes   | Input frequency should be
as low as possible
  IWDG1                 |LSI      |    0.032000 MHz |  yes   |
  IWDG2                 |LSI      |    0.032000 MHz |  yes   |
LSE                     |N.A.     |    0.032768 MHz |  yes   | Mandatory for DTS
  RTC                   |LSE      |    0.032768 MHz |  yes   |
  TAMP                  |LSE      |    0.032768 MHz |  yes   |
  CEC                   |LSE      |    0.032768 MHz |  yes   |
  LPTIM4                |LSE      |    0.032768 MHz |  yes   |
  LPTIM5                |LSE      |    0.032768 MHz |  yes   |
HSI                     |N.A.     |   64.000000 MHz |  yes   |
  SPI4                  |HSI      |   64.000000 MHz |  yes   | Arduino
  SPI5                  |HSI      |   64.000000 MHz |  yes   | Rpi
  SPI6                  |HSI      |   64.000000 MHz |  no    |
  I2C4                  |HSI      |   64.000000 MHz |  yes   | PMIC
  I2C6                  |HSI      |   64.000000 MHz |  no    |
  I2C1                  |HSI      |   64.000000 MHz |  yes   | Rpi and peripherals
  I2C2                  |HSI      |   64.000000 MHz |  no    |
  I2C3                  |HSI      |   64.000000 MHz |  no    |
  I2C5                  |HSI      |   64.000000 MHz |  yes   | Rpi and Arduino
  USART1                |HSI      |   64.000000 MHz |  no    |
  USART2                |HSI      |   64.000000 MHz |  yes   | Bluetooth
  USART3                |HSI      |   64.000000 MHz |  yes   | Rpi
  UART4                 |HSI      |   64.000000 MHz |  yes   | Linux console
  UART5                 |HSI      |   64.000000 MHz |  no    |
  USART6                |HSI      |   64.000000 MHz |  no    |
  UART7                 |HSI      |   64.000000 MHz |  yes   | Arduino
  UART8                 |HSI      |   64.000000 MHz |  no    |
  MCO1                  |HSI      |   64.000000 MHz |  no    | Available so can be used
HSE                     |N.A.     |   24.000000 MHz |  yes   |
  DSIPLL                |HSE      |  125.000000 MHz |  no    | DSI DPHY PLL
```

```
    DSIBL              |DSIPLL    |  125.000000 MHz |  no    | DSI lanebyte clock
  RTCDIV               |HSE       |    1.000000 MHz |  yes   | Only used when RTC source
is HSE
  ck_per               |HSE       |   24.000000 MHz |  yes   |
    ADC                |ck_per    |   24.000000 MHz |  yes   | Can use internal divider to
be < 40MHz
  PLL1                 |HSE       | 1300.000000 MHz |  yes   |
    PLL1P              |PLL1      |  650.000000 MHz |  yes   |
      MPUDIV           |PLL1P     |  325.000000 MHz |  yes   |
      Cortex-A7        |PLL1P     |  650.000000 MHz |  yes   |
        MCO2           |Cortex-A7 |  650.000000 MHz |  no    | Available so can be used
  PLL2                 |HSE       | 1066.000000 MHz |  yes   |
    PLL2P              |PLL2      |  266.500000 MHz |  yes   |
      AXI              |PLL2P     |  266.500000 MHz |  yes   | < 266MHz
        FMC            |AXI       |  266.500000 MHz |  no    |
        QSPI           |AXI       |  266.500000 MHz |  no    |
        SYSRAM         |AXI       |  266.500000 MHz |  yes   |
        ROM            |AXI       |  266.500000 MHz |  yes   |
        AHB5           |AXI       |  266.500000 MHz |  yes   | < 266MHz
          CRYP1        |AHB5      |  266.500000 MHz |  yes   |
          HASH1        |AHB5      |  266.500000 MHz |  yes   |
          GPIOZ        |AHB5      |  266.500000 MHz |  yes   |
          BKPSRAM      |AHB5      |  266.500000 MHz |  yes   |
        AHB6           |AXI       |  266.500000 MHz |  yes   | < 266MHz
          CRC1         |AHB6      |  266.500000 MHz |  yes   |
          MDMA         |AHB6      |  266.500000 MHz |  yes   |
          USBH         |AHB6      |  266.500000 MHz |  yes   | USB Host
          APB4         |AHB6      |  133.250000 MHz |  yes   |
          APB5         |AHB6      |   66.625000 MHz |  yes   |
            BSEC       |APB5      |   66.625000 MHz |  yes   | < 67MHz
            ETZPC      |APB5      |   66.625000 MHz |  yes   |
            TZC        |APB5      |   66.625000 MHz |  yes   |
        DBGAPB         |AXI       |  133.250000 MHz |  yes   | JTAG & Coresight
          DBGMCU       |DBGAPB    |   66.625000 MHz |  yes   |
          STM          |DBGAPB    |   66.625000 MHz |  no    |
    PLL2Q              |PLL2      |  533.000000 MHz |  yes   |
      GPU              |PLL2Q     |  533.000000 MHz |  yes   | < 533MHz
    PLL2R              |PLL2      |  533.000000 MHz |  yes   |
      DDR              |PLL2R     |  533.000000 MHz |  yes   | < 533MHz
  PLL3                 |HSE       |  417.755859 MHz |  yes   |
    PLL3P              |PLL3      |  208.877930 MHz |  yes   |
      MLAHB            |PLL3P     |  208.877930 MHz |  yes   | < 209MHz
        Cortex-M4      |MLAHB     |  208.877930 MHz |  yes   |
        SRAM1          |MLAHB     |  208.877930 MHz |  yes   |
        SRAM2          |MLAHB     |  208.877930 MHz |  yes   |
        SRAM3          |MLAHB     |  208.877930 MHz |  yes   |
        RETRAM         |MLAHB     |  208.877930 MHz |  yes   |
        AHB1           |MLAHB     |  208.877930 MHz |  yes   | < 209MHz
        AHB2           |MLAHB     |  208.877930 MHz |  yes   | < 209MHz
          DMA1         |AHB2      |  208.877930 MHz |  yes   |
          DMA2         |AHB2      |  208.877930 MHz |  yes   |
          DMAMUX       |AHB2      |  208.877930 MHz |  yes   |
        APB1           |MLAHB     |  104.438965 MHz |  yes   |
          LPTIM1       |APB1      |  104.438965 MHz |  yes   |
          WWDG         |APB1      |  104.438965 MHz |  yes   |
        APB2           |MLAHB     |  104.438965 MHz |  yes   |
        TIM2           |MLAHB     |  208.877930 MHz |  yes   | TIM Group 1
        TIM3           |MLAHB     |  208.877930 MHz |  yes   | TIM Group 1
        TIM4           |MLAHB     |  208.877930 MHz |  yes   | TIM Group 1
        TIM5           |MLAHB     |  208.877930 MHz |  yes   | TIM Group 1
        TIM6           |MLAHB     |  208.877930 MHz |  yes   | TIM Group 1
        TIM7           |MLAHB     |  208.877930 MHz |  yes   | TIM Group 1
        TIM12          |MLAHB     |  208.877930 MHz |  yes   | TIM Group 1
        TIM13          |MLAHB     |  208.877930 MHz |  yes   | TIM Group 1
        TIM14          |MLAHB     |  208.877930 MHz |  yes   | TIM Group 1
        TIM1           |MLAHB     |  208.877930 MHz |  yes   | TIM Group 2
        TIM8           |MLAHB     |  208.877930 MHz |  yes   | TIM Group 2
```

```
        TIM15           |MLAHB    |  208.877930 MHz |  yes   | TIM Group 2
        TIM16           |MLAHB    |  208.877930 MHz |  yes   | TIM Group 2
        TIM17           |MLAHB    |  208.877930 MHz |  yes   | TIM Group 2
        APB3            |MLAHB    |  104.438965 MHz |  yes   |
          LPTIM2        |APB3     |  104.438965 MHz |  yes   |
          LPTIM3        |APB3     |  104.438965 MHz |  yes   |
          SYSCFG        |APB3     |  104.438965 MHz |  yes   |
          VREFBUF       |APB3     |  104.438965 MHz |  yes   |
          DTS           |APB3     |  104.438965 MHz |  yes   |
          HDP           |APB3     |  104.438965 MHz |  no    |
        AHB3            |MLAHB    |  208.877930 MHz |  yes   | < 209MHz
          CRC2          |AHB3     |  208.877930 MHz |  yes   |
          CRYP2         |AHB3     |  208.877930 MHz |  yes   |
          HASH2         |AHB3     |  208.877930 MHz |  yes   |
          DCMI          |AHB3     |  208.877930 MHz |  no    |
          IPCC          |AHB3     |  208.877930 MHz |  yes   | Mailbox
        AHB4            |MLAHB    |  208.877930 MHz |  yes   | < 209MHz
          PWR           |AHB4     |  208.877930 MHz |  yes   |
          RCC           |AHB4     |  208.877930 MHz |  yes   |
          GPIOA-K       |AHB4     |  208.877930 MHz |  yes   |
          EXTI          |AHB4     |  208.877930 MHz |  yes   |
    PLL3Q               |PLL3     |   24.573874 MHz |  yes   |
      SPI1              |PLL3Q    |   24.573874 MHz |  yes   | SPI2S1 for BT PCM
      SPI2              |PLL3Q    |   24.573874 MHz |  yes   | SPI2S2 for HDMI
      SPI3              |PLL3Q    |   24.573874 MHz |  no    |
      DFSDM             |PLL3Q    |   24.573874 MHz |  no    |
      SAI1              |PLL3Q    |   24.573874 MHz |  no    |
      SAI2              |PLL3Q    |   24.573874 MHz |  yes   | AudCodec 48kHz (use PLL3R
for 44.1kHz)
      SAI3              |PLL3Q    |   24.573874 MHz |  no    |
      SAI4              |PLL3Q    |   24.573874 MHz |  no    |
    PLL3R               |PLL3     |   11.290699 MHz |  yes   |
  PLL4                  |HSE      |  594.000000 MHz |  yes   |
    PLL4P               |PLL4     |   99.000000 MHz |  yes   |
      SDMMC1            |PLL4P    |   99.000000 MHz |  yes   | µSD card
      SDMMC2            |PLL4P    |   99.000000 MHz |  yes   | Wifi
      SDMMC3            |PLL4P    |   99.000000 MHz |  yes   | Rpi
      SPDIF             |PLL4P    |   99.000000 MHz |  no    |
    PLL4Q               |PLL4     |   74.250000 MHz |  yes   |
      LCD               |PLL4Q    |   74.250000 MHz |  yes   | LTDC & DSI display pixel
clock
    PLL4R               |PLL4     |   74.250000 MHz |  yes   |
      FDCAN             |PLL4R    |   74.250000 MHz |  no    |
  STGEN                 |HSE      |   24.000000 MHz |  yes   |
  USBPHYC               |HSE      |   24.000000 MHz |  yes   | USB PHY Ctrl for USB Host
and OTG
    USBPLL              |USBPHYC  |   48.000000 MHz |  yes   |
      USBO              |USBPLL   |   48.000000 MHz |  yes   | USB OTG
CSI                     |N.A.     |    4.000000 MHz |  yes   | Mandatory for IO
compensation
ETH                     |N.A.     |    0.000000 MHz |  no    | ETH clocked by RGMII PHY
------------------------|---------|-----------------|--------
|-----------------------------------|
```

3.1.2.2 *Device tree*

Here are the corresponding device tree **rcc_clk** sub node properties consumed by the first stage boot loader (FSBL) to configure the clock tree above:

```
        st,clksrc = <
                CLK_MPU_PLL1P
                CLK_AXI_PLL2P
                CLK_MCU_PLL3P
                CLK_PLL12_HSE
```

```
                CLK_PLL3_HSE
                CLK_PLL4_HSE
                CLK_RTC_LSE
                CLK_MCO1_DISABLED
                CLK_MCO2_DISABLED
        >;

        st,clkdiv = <
                1 /*MPU*/
                0 /*AXI*/
                0 /*MCU*/
                1 /*APB1*/
                1 /*APB2*/
                1 /*APB3*/
                1 /*APB4*/
                2 /*APB5*/
                23 /*RTC*/
                0 /*MCO1*/
                0 /*MCO2*/
        >;

        st,pkcs = <
                CLK_CKPER_HSE
                CLK_FMC_ACLK
                CLK_QSPI_ACLK
                CLK_ETH_DISABLED
                CLK_SDMMC12_PLL4P
                CLK_DSI_DSIPLL
                CLK_STGEN_HSE
                CLK_USBPHY_HSE
                CLK_SPI2S1_PLL3Q
                CLK_SPI2S23_PLL3Q
                CLK_SPI45_HSI
                CLK_SPI6_HSI
                CLK_I2C46_HSI
                CLK_SDMMC3_PLL4P
                CLK_USBO_USBPHY
                CLK_ADC_CKPER
                CLK_CEC_LSE
                CLK_I2C12_HSI
                CLK_I2C35_HSI
                CLK_UART1_HSI
                CLK_UART24_HSI
                CLK_UART35_HSI
                CLK_UART6_HSI
                CLK_UART78_HSI
                CLK_SPDIF_PLL4P
                CLK_FDCAN_PLL4R
                CLK_SAI1_PLL3Q
                CLK_SAI2_PLL3Q
                CLK_SAI3_PLL3Q
                CLK_SAI4_PLL3Q
                CLK_RNG1_LSI
                CLK_RNG2_LSI
                CLK_LPTIM1_PCLK1
                CLK_LPTIM23_PCLK3
                CLK_LPTIM45_LSE
        >;

        /* VCO = 1300.0 MHz => P = 650 (CPU) */
        pll1: st,pll@0 {
                cfg = < 2 80 0 0 0 PQR(1,0,0) >;
                frac = < 0x800 >;
                u-boot,dm-pre-reloc;
        };

        /* VCO = 1066.0 MHz => P = 266 (AXI), Q = 533 (GPU), R = 533 (DDR) */
```

```
        pll2: st,pll@1 {
                cfg = < 2 65 1 0 0 PQR(1,1,1) >;
                frac = < 0x1400 >;
                u-boot,dm-pre-reloc;
        };

        /* VCO = 417.8 MHz => P = 209, Q = 25, R = 11 */
        pll3: st,pll@2 {
                cfg = < 1 33 1 16 36 PQR(1,1,1) >;
                frac = < 0x1a04 >;
                u-boot,dm-pre-reloc;
        };

        /* VCO = 594.0 MHz => P = 99, Q = 74, R = 74 */
        pll4: st,pll@3 {
                cfg = < 3 98 5 7 7 PQR(1,1,1) >;
                u-boot,dm-pre-reloc;
        };
```

3.2 For ecosystem release v1.0.0

3.2.1 STM32MP157C-EV1 case

This chapter shows the boot time clock tree set by the FSBL on STM32MP157C-EV1 evaluation board. Linux eventual runtime modifications are not covered here.

3.2.1.1 Clock tree

The following table shows what STM32MP157C-EV1 clock tree looks like, as a result of the boot chain execution with the device tree built with STM32CubeMX.

```
Component               | Parent  |   Frequency     | Used?  |
Comment                          |
------------------------|---------|-----------------|--------
|----------------------------------|
LSI                     |N.A.     |    0.032000 MHz |  yes   | Mandatory for IWDG, DAC
  DAC                   |LSI      |    0.032000 MHz |  yes   |
  RNG1                  |LSI      |    0.032000 MHz |  yes   | Input frequency should be
as low as possible
  RNG2                  |LSI      |    0.032000 MHz |  yes   | Input frequency should be
as low as possible
  IWDG1                 |LSI      |    0.032000 MHz |  yes   |
  IWDG2                 |LSI      |    0.032000 MHz |  yes   |
LSE                     |N.A.     |    0.032768 MHz |  yes   | Mandatory for DTS
  RTC                   |LSE      |    0.032768 MHz |  yes   |
  TAMP                  |LSE      |    0.032768 MHz |  yes   |
  CEC                   |LSE      |    0.032768 MHz |  yes   |
  LPTIM4                |LSE      |    0.032768 MHz |  yes   |
  LPTIM5                |LSE      |    0.032768 MHz |  yes   |
HSI                     |N.A.     |   64.000000 MHz |  yes   |
  SPI4                  |HSI      |   64.000000 MHz |  no    |
  SPI5                  |HSI      |   64.000000 MHz |  no    |
  SPI6                  |HSI      |   64.000000 MHz |  yes   |
  I2C4                  |HSI      |   64.000000 MHz |  yes   | PMIC
  I2C6                  |HSI      |   64.000000 MHz |  no    |
  I2C1                  |HSI      |   64.000000 MHz |  no    |
  I2C2                  |HSI      |   64.000000 MHz |  yes   | Rpi and peripherals
  I2C3                  |HSI      |   64.000000 MHz |  no    |
  I2C5                  |HSI      |   64.000000 MHz |  yes   | Rpi
  USART1                |HSI      |   64.000000 MHz |  yes   |
  USART2                |HSI      |   64.000000 MHz |  no    |
  USART3                |HSI      |   64.000000 MHz |  yes   | Rpi
```

```
  UART4                |HSI       |    64.000000 MHz |  yes   | Linux console
  UART5                |HSI       |    64.000000 MHz |  no    |
  USART6               |HSI       |    64.000000 MHz |  no    |
  UART7                |HSI       |    64.000000 MHz |  no    |
  UART8                |HSI       |    64.000000 MHz |  no    |
  MCO1                 |HSI       |    64.000000 MHz |  no    | Available so can be used
HSE                    |N.A.      |    24.000000 MHz |  yes   |
  DSIPLL               |HSE       |   125.000000 MHz |  no    | DSI DPHY PLL
    DSIBL              |DSIPLL    |   125.000000 MHz |  no    | DSI lanebyte clock
  RTCDIV               |HSE       |     1.000000 MHz |  yes   | Only used when RTC source
is HSE
  ck_per               |HSE       |    24.000000 MHz |  yes   |
    ADC                |ck_per    |    24.000000 MHz |  yes   | Can use internal divider to
be < 40MHz
  PLL1                 |HSE       |  1300.000000 MHz |  yes   |
    PLL1P              |PLL1      |   650.000000 MHz |  yes   |
      MPUDIV           |PLL1P     |   325.000000 MHz |  yes   |
      Cortex-A7        |PLL1P     |   650.000000 MHz |  yes   |
        MCO2           |Cortex-A7 |   650.000000 MHz |  no    | Available so can be used
  PLL2                 |HSE       |  1066.000000 MHz |  yes   |
    PLL2P              |PLL2      |   266.500000 MHz |  yes   |
      AXI              |PLL2P     |   266.500000 MHz |  yes   | < 266MHz
        FMC            |AXI       |   266.500000 MHz |  yes   | NAND flash
        QSPI           |AXI       |   266.500000 MHz |  yes   | NOR flash
        SYSRAM         |AXI       |   266.500000 MHz |  yes   |
        ROM            |AXI       |   266.500000 MHz |  yes   |
        AHB5           |AXI       |   266.500000 MHz |  yes   | < 266MHz
          CRYP1        |AHB5      |   266.500000 MHz |  yes   |
          HASH1        |AHB5      |   266.500000 MHz |  yes   |
          GPIOZ        |AHB5      |   266.500000 MHz |  yes   |
          BKPSRAM      |AHB5      |   266.500000 MHz |  yes   |
        AHB6           |AXI       |   266.500000 MHz |  yes   | < 266MHz
          CRC1         |AHB6      |   266.500000 MHz |  yes   |
          MDMA         |AHB6      |   266.500000 MHz |  yes   |
          USBH         |AHB6      |   266.500000 MHz |  yes   | USB Host
          APB4         |AHB6      |   133.250000 MHz |  yes   |
          APB5         |AHB6      |    66.625000 MHz |  yes   |
            BSEC       |APB5      |    66.625000 MHz |  yes   | < 67MHz
            ETZPC      |APB5      |    66.625000 MHz |  yes   |
            TZC        |APB5      |    66.625000 MHz |  yes   |
        DBGAPB         |AXI       |   133.250000 MHz |  yes   | JTAG & Coresight
          DBGMCU       |DBGAPB    |    66.625000 MHz |  yes   |
          STM          |DBGAPB    |    66.625000 MHz |  yes   |
    PLL2Q              |PLL2      |   533.000000 MHz |  yes   |
      GPU              |PLL2Q     |   533.000000 MHz |  yes   | < 533MHz
    PLL2R              |PLL2      |   533.000000 MHz |  yes   |
      DDR              |PLL2R     |   533.000000 MHz |  yes   | < 533MHz
  PLL3                 |HSE       |   417.755859 MHz |  yes   |
    PLL3P              |PLL3      |   208.877930 MHz |  yes   |
      MLAHB            |PLL3P     |   208.877930 MHz |  yes   | < 209MHz
        Cortex-M4      |MLAHB     |   208.877930 MHz |  yes   |
        SRAM1          |MLAHB     |   208.877930 MHz |  yes   |
        SRAM2          |MLAHB     |   208.877930 MHz |  yes   |
        SRAM3          |MLAHB     |   208.877930 MHz |  yes   |
        RETRAM         |MLAHB     |   208.877930 MHz |  yes   |
        AHB1           |MLAHB     |   208.877930 MHz |  yes   | < 209MHz
        AHB2           |MLAHB     |   208.877930 MHz |  yes   | < 209MHz
          DMA1         |AHB2      |   208.877930 MHz |  yes   |
          DMA2         |AHB2      |   208.877930 MHz |  yes   |
          DMAMUX       |AHB2      |   208.877930 MHz |  yes   |
        APB1           |MLAHB     |   104.438965 MHz |  yes   |
          LPTIM1       |APB1      |   104.438965 MHz |  yes   |
          WWDG         |APB1      |   104.438965 MHz |  yes   |
        APB2           |MLAHB     |   104.438965 MHz |  yes   |
        TIM2           |MLAHB     |   208.877930 MHz |  yes   | TIM Group 1
        TIM3           |MLAHB     |   208.877930 MHz |  yes   | TIM Group 1
        TIM4           |MLAHB     |   208.877930 MHz |  yes   | TIM Group 1
```

```
        TIM5            |MLAHB    |  208.877930 MHz |  yes   | TIM Group 1
        TIM6            |MLAHB    |  208.877930 MHz |  yes   | TIM Group 1
        TIM7            |MLAHB    |  208.877930 MHz |  yes   | TIM Group 1
        TIM12           |MLAHB    |  208.877930 MHz |  yes   | TIM Group 1
        TIM13           |MLAHB    |  208.877930 MHz |  yes   | TIM Group 1
        TIM14           |MLAHB    |  208.877930 MHz |  yes   | TIM Group 1
        TIM1            |MLAHB    |  208.877930 MHz |  yes   | TIM Group 2
        TIM8            |MLAHB    |  208.877930 MHz |  yes   | TIM Group 2
        TIM15           |MLAHB    |  208.877930 MHz |  yes   | TIM Group 2
        TIM16           |MLAHB    |  208.877930 MHz |  yes   | TIM Group 2
        TIM17           |MLAHB    |  208.877930 MHz |  yes   | TIM Group 2
        APB3            |MLAHB    |  104.438965 MHz |  yes   |
          LPTIM2        |APB3     |  104.438965 MHz |  yes   |
          LPTIM3        |APB3     |  104.438965 MHz |  yes   |
          SYSCFG        |APB3     |  104.438965 MHz |  yes   |
          VREFBUF       |APB3     |  104.438965 MHz |  yes   |
          DTS           |APB3     |  104.438965 MHz |  yes   |
          HDP           |APB3     |  104.438965 MHz |  yes   |
        AHB3            |MLAHB    |  208.877930 MHz |  yes   | < 209MHz
          CRC2          |AHB3     |  208.877930 MHz |  yes   |
          CRYP2         |AHB3     |  208.877930 MHz |  yes   |
          HASH2         |AHB3     |  208.877930 MHz |  yes   |
          DCMI          |AHB3     |  208.877930 MHz |  yes   | Camera
          IPCC          |AHB3     |  208.877930 MHz |  yes   | Mailbox
        AHB4            |MLAHB    |  208.877930 MHz |  yes   | < 209MHz
          PWR           |AHB4     |  208.877930 MHz |  yes   |
          RCC           |AHB4     |  208.877930 MHz |  yes   |
          GPIOA-K       |AHB4     |  208.877930 MHz |  yes   |
          EXTI          |AHB4     |  208.877930 MHz |  yes   |
    PLL3Q               |PLL3     |   24.573874 MHz |  yes   |
      SPI1              |PLL3Q    |   24.573874 MHz |  no    |
      SPI2              |PLL3Q    |   24.573874 MHz |  no    |
      SPI3              |PLL3Q    |   24.573874 MHz |  no    |
      DFSDM             |PLL3Q    |   24.573874 MHz |  yes   | Digital micro
      SAI1              |PLL3Q    |   24.573874 MHz |  no    |
      SAI2              |PLL3Q    |   24.573874 MHz |  yes   | AudCodec 48kHz (use PLL3R
for 44.1kHz)
      SAI3              |PLL3Q    |   24.573874 MHz |  no    |
      SAI4              |PLL3Q    |   24.573874 MHz |  yes   | SPDIF TX 48kHz (use PLL3R
for 44.1kHz)
    PLL3R               |PLL3     |   11.290699 MHz |  yes   |
  PLL4                  |HSE      |  594.000000 MHz |  yes   |
    PLL4P               |PLL4     |   99.000000 MHz |  yes   |
      SDMMC1            |PLL4P    |   99.000000 MHz |  yes   | µSD card
      SDMMC2            |PLL4P    |   99.000000 MHz |  yes   | eMMC
      SDMMC3            |PLL4P    |   99.000000 MHz |  yes   |
      SPDIF             |PLL4P    |   99.000000 MHz |  yes   | SPDIF RX
    PLL4Q               |PLL4     |   74.250000 MHz |  yes   |
      LCD               |PLL4Q    |   74.250000 MHz |  yes   | LTDC & DSI display pixel
clock
      FDCAN             |PLL4Q    |   74.250000 MHz |  yes   | Should be as high as
possible and < 100MHz
    PLL4R               |PLL4     |   74.250000 MHz |  yes   |
  STGEN                 |HSE      |   24.000000 MHz |  yes   |
  USBPHYC               |HSE      |   24.000000 MHz |  yes   | USB PHY Ctrl for USB Host
and OTG
    USBPLL              |USBPHYC  |   48.000000 MHz |  yes   |
      USBO              |USBPLL   |   48.000000 MHz |  yes   | USB OTG
CSI                     |N.A.     |    4.000000 MHz |  yes   | Mandatory for IO
compensation
ETH                     |N.A.     |    0.000000 MHz |  no    | ETH clocked by RGMII PHY
------------------------|---------|-----------------|--------
|-----------------------------------|
```

3.2.1.2 *Device tree*

Here are the corresponding device tree **rcc_clk** sub node properties consumed by the first stage boot loader (FSBL) to configure the clock tree above:

```
        st,clksrc = <
                CLK_MPU_PLL1P
                CLK_AXI_PLL2P
                CLK_MCU_PLL3P
                CLK_PLL12_HSE
                CLK_PLL3_HSE
                CLK_PLL4_HSE
                CLK_RTC_LSE
                CLK_MCO1_DISABLED
                CLK_MCO2_DISABLED
        >;

        st,clkdiv = <
                1 /*MPU*/
                0 /*AXI*/
                0 /*MCU*/
                1 /*APB1*/
                1 /*APB2*/
                1 /*APB3*/
                1 /*APB4*/
                2 /*APB5*/
                23 /*RTC*/
                0 /*MCO1*/
                0 /*MCO2*/
        >;

        st,pkcs = <
                CLK_CKPER_HSE
                CLK_FMC_ACLK
                CLK_QSPI_ACLK
                CLK_ETH_DISABLED
                CLK_SDMMC12_PLL4P
                CLK_DSI_DSIPLL
                CLK_STGEN_HSE
                CLK_USBPHY_HSE
                CLK_SPI2S1_PLL3Q
                CLK_SPI2S23_PLL3Q
                CLK_SPI45_HSI
                CLK_SPI6_HSI
                CLK_I2C46_HSI
                CLK_SDMMC3_PLL4P
                CLK_USBO_USBPHY
                CLK_ADC_CKPER
                CLK_CEC_LSE
                CLK_I2C12_HSI
                CLK_I2C35_HSI
                CLK_UART1_HSI
                CLK_UART24_HSI
                CLK_UART35_HSI
                CLK_UART6_HSI
                CLK_UART78_HSI
                CLK_SPDIF_PLL4P
                CLK_FDCAN_PLL4Q
                CLK_SAI1_PLL3Q
                CLK_SAI2_PLL3Q
                CLK_SAI3_PLL3Q
                CLK_SAI4_PLL3Q
                CLK_RNG1_LSI
                CLK_RNG2_LSI
```

```
                CLK_LPTIM1_PCLK1
                CLK_LPTIM23_PCLK3
                CLK_LPTIM45_LSE
        >;

        /* VCO = 1300.0 MHz => P = 650 (CPU) */
        pll1: st,pll@0 {
                cfg = < 2 80 0 0 0 PQR(1,0,0) >;
                frac = < 0x800 >;
                u-boot,dm-pre-reloc;
        };

        /* VCO = 1066.0 MHz => P = 266 (AXI), Q = 533 (GPU), R = 533 (DDR) */
        pll2: st,pll@1 {
                cfg = < 2 65 1 0 0 PQR(1,1,1) >;
                frac = < 0x1400 >;
                u-boot,dm-pre-reloc;
        };

        /* VCO = 417.8 MHz => P = 209, Q = 25, R = 11 */
        pll3: st,pll@2 {
                cfg = < 1 33 1 16 36 PQR(1,1,1) >;
                frac = < 0x1a04 >;
                u-boot,dm-pre-reloc;
        };

        /* VCO = 594.0 MHz => P = 99, Q = 74, R = 74 */
        pll4: st,pll@3 {
                cfg = < 3 98 5 7 7 PQR(1,1,1) >;
                u-boot,dm-pre-reloc;
        };
```

3.2.2 STM32MP157C-DK2 case

This chapter shows the boot time clock tree set by the FSBL on STM32MP157C-DK2 DISCO board. Linux eventual runtime modifications are not covered here.

3.2.2.1 Clock tree

The following table shows what STM32MP157C-DK2 clock tree looks like, as a result of the boot chain execution with the device tree built with STM32CubeMX.

```
Component              | Parent  |   Frequency     | Used?  |
Comment                                |
-----------------------|---------|-----------------|--------
|-----------------------------------|
LSI                    |N.A.     |    0.032000 MHz |  yes   | Mandatory for IWDG, DAC
  DAC                  |LSI      |    0.032000 MHz |  no    |
  RNG1                 |LSI      |    0.032000 MHz |  yes   | Input frequency should be
as low as possible
  RNG2                 |LSI      |    0.032000 MHz |  yes   | Input frequency should be
as low as possible
  IWDG1                |LSI      |    0.032000 MHz |  yes   |
  IWDG2                |LSI      |    0.032000 MHz |  yes   |
LSE                    |N.A.     |    0.032768 MHz |  yes   | Mandatory for DTS
  RTC                  |LSE      |    0.032768 MHz |  yes   |
  TAMP                 |LSE      |    0.032768 MHz |  yes   |
  CEC                  |LSE      |    0.032768 MHz |  yes   |
  LPTIM4               |LSE      |    0.032768 MHz |  yes   |
  LPTIM5               |LSE      |    0.032768 MHz |  yes   |
HSI                    |N.A.     |   64.000000 MHz |  yes   |
  SPI4                 |HSI      |   64.000000 MHz |  yes   | Arduino
  SPI5                 |HSI      |   64.000000 MHz |  yes   | Rpi
  SPI6                 |HSI      |   64.000000 MHz |  no    |
```

```
  I2C4                |HSI       |    64.000000 MHz |  yes    | PMIC
  I2C6                |HSI       |    64.000000 MHz |  no     |
  I2C1                |HSI       |    64.000000 MHz |  yes    | Rpi and peripherals
  I2C2                |HSI       |    64.000000 MHz |  no     |
  I2C3                |HSI       |    64.000000 MHz |  no     |
  I2C5                |HSI       |    64.000000 MHz |  yes    | Rpi and Arduino
  USART1              |HSI       |    64.000000 MHz |  no     |
  USART2              |HSI       |    64.000000 MHz |  yes    | Bluetooth
  USART3              |HSI       |    64.000000 MHz |  yes    | Rpi
  UART4               |HSI       |    64.000000 MHz |  yes    | Linux console
  UART5               |HSI       |    64.000000 MHz |  no     |
  USART6              |HSI       |    64.000000 MHz |  no     |
  UART7               |HSI       |    64.000000 MHz |  yes    | Arduino
  UART8               |HSI       |    64.000000 MHz |  no     |
  MCO1                |HSI       |    64.000000 MHz |  no     | Available so can be used
HSE                   |N.A.      |    24.000000 MHz |  yes    |
  DSIPLL              |HSE       |   125.000000 MHz |  no     | DSI DPHY PLL
    DSIBL             |DSIPLL    |   125.000000 MHz |  no     | DSI lanebyte clock
  RTCDIV              |HSE       |     1.000000 MHz |  yes    | Only used when RTC source
is HSE
  ck_per              |HSE       |    24.000000 MHz |  yes    |
    ADC               |ck_per    |    24.000000 MHz |  yes    | Can use internal divider to
be < 40MHz
  PLL1                |HSE       |  1300.000000 MHz |  yes    |
    PLL1P             |PLL1      |   650.000000 MHz |  yes    |
      MPUDIV          |PLL1P     |   325.000000 MHz |  yes    |
      Cortex-A7       |PLL1P     |   650.000000 MHz |  yes    |
        MCO2          |Cortex-A7 |   650.000000 MHz |  no     | Available so can be used
  PLL2                |HSE       |  1066.000000 MHz |  yes    |
    PLL2P             |PLL2      |   266.500000 MHz |  yes    |
      AXI             |PLL2P     |   266.500000 MHz |  yes    | < 266MHz
        FMC           |AXI       |   266.500000 MHz |  no     |
        QSPI          |AXI       |   266.500000 MHz |  no     |
        SYSRAM        |AXI       |   266.500000 MHz |  yes    |
        ROM           |AXI       |   266.500000 MHz |  yes    |
        AHB5          |AXI       |   266.500000 MHz |  yes    | < 266MHz
          CRYP1       |AHB5      |   266.500000 MHz |  yes    |
          HASH1       |AHB5      |   266.500000 MHz |  yes    |
          GPIOZ       |AHB5      |   266.500000 MHz |  yes    |
          BKPSRAM     |AHB5      |   266.500000 MHz |  yes    |
        AHB6          |AXI       |   266.500000 MHz |  yes    | < 266MHz
          CRC1        |AHB6      |   266.500000 MHz |  yes    |
          MDMA        |AHB6      |   266.500000 MHz |  yes    |
          USBH        |AHB6      |   266.500000 MHz |  yes    | USB Host
          APB4        |AHB6      |   133.250000 MHz |  yes    |
          APB5        |AHB6      |    66.625000 MHz |  yes    |
            BSEC      |APB5      |    66.625000 MHz |  yes    | < 67MHz
            ETZPC     |APB5      |    66.625000 MHz |  yes    |
            TZC       |APB5      |    66.625000 MHz |  yes    |
        DBGAPB        |AXI       |   133.250000 MHz |  yes    | JTAG & Coresight
          DBGMCU      |DBGAPB    |    66.625000 MHz |  yes    |
          STM         |DBGAPB    |    66.625000 MHz |  no     |
    PLL2Q             |PLL2      |   533.000000 MHz |  yes    |
      GPU             |PLL2Q     |   533.000000 MHz |  yes    | < 533MHz
    PLL2R             |PLL2      |   533.000000 MHz |  yes    |
      DDR             |PLL2R     |   533.000000 MHz |  yes    | < 533MHz
  PLL3                |HSE       |   417.755859 MHz |  yes    |
    PLL3P             |PLL3      |   208.877930 MHz |  yes    |
      MLAHB           |PLL3P     |   208.877930 MHz |  yes    | < 209MHz
        Cortex-M4     |MLAHB     |   208.877930 MHz |  yes    |
        SRAM1         |MLAHB     |   208.877930 MHz |  yes    |
        SRAM2         |MLAHB     |   208.877930 MHz |  yes    |
        SRAM3         |MLAHB     |   208.877930 MHz |  yes    |
        RETRAM        |MLAHB     |   208.877930 MHz |  yes    |
        AHB1          |MLAHB     |   208.877930 MHz |  yes    | < 209MHz
        AHB2          |MLAHB     |   208.877930 MHz |  yes    | < 209MHz
          DMA1        |AHB2      |   208.877930 MHz |  yes    |
```

```
          DMA2          |AHB2     |   208.877930 MHz |   yes   |
          DMAMUX        |AHB2     |   208.877930 MHz |   yes   |
        APB1            |MLAHB    |   104.438965 MHz |   yes   |
          LPTIM1        |APB1     |   104.438965 MHz |   yes   |
          WWDG          |APB1     |   104.438965 MHz |   yes   |
        APB2            |MLAHB    |   104.438965 MHz |   yes   |
        TIM2            |MLAHB    |   208.877930 MHz |   yes   | TIM Group 1
        TIM3            |MLAHB    |   208.877930 MHz |   yes   | TIM Group 1
        TIM4            |MLAHB    |   208.877930 MHz |   yes   | TIM Group 1
        TIM5            |MLAHB    |   208.877930 MHz |   yes   | TIM Group 1
        TIM6            |MLAHB    |   208.877930 MHz |   yes   | TIM Group 1
        TIM7            |MLAHB    |   208.877930 MHz |   yes   | TIM Group 1
        TIM12           |MLAHB    |   208.877930 MHz |   yes   | TIM Group 1
        TIM13           |MLAHB    |   208.877930 MHz |   yes   | TIM Group 1
        TIM14           |MLAHB    |   208.877930 MHz |   yes   | TIM Group 1
        TIM1            |MLAHB    |   208.877930 MHz |   yes   | TIM Group 2
        TIM8            |MLAHB    |   208.877930 MHz |   yes   | TIM Group 2
        TIM15           |MLAHB    |   208.877930 MHz |   yes   | TIM Group 2
        TIM16           |MLAHB    |   208.877930 MHz |   yes   | TIM Group 2
        TIM17           |MLAHB    |   208.877930 MHz |   yes   | TIM Group 2
        APB3            |MLAHB    |   104.438965 MHz |   yes   |
          LPTIM2        |APB3     |   104.438965 MHz |   yes   |
          LPTIM3        |APB3     |   104.438965 MHz |   yes   |
          SYSCFG        |APB3     |   104.438965 MHz |   yes   |
          VREFBUF       |APB3     |   104.438965 MHz |   yes   |
          DTS           |APB3     |   104.438965 MHz |   yes   |
          HDP           |APB3     |   104.438965 MHz |   no    |
        AHB3            |MLAHB    |   208.877930 MHz |   yes   | < 209MHz
          CRC2          |AHB3     |   208.877930 MHz |   yes   |
          CRYP2         |AHB3     |   208.877930 MHz |   yes   |
          HASH2         |AHB3     |   208.877930 MHz |   yes   |
          DCMI          |AHB3     |   208.877930 MHz |   no    |
          IPCC          |AHB3     |   208.877930 MHz |   yes   | Mailbox
        AHB4            |MLAHB    |   208.877930 MHz |   yes   | < 209MHz
          PWR           |AHB4     |   208.877930 MHz |   yes   |
          RCC           |AHB4     |   208.877930 MHz |   yes   |
          GPIOA-K       |AHB4     |   208.877930 MHz |   yes   |
          EXTI          |AHB4     |   208.877930 MHz |   yes   |
    PLL3Q               |PLL3     |    24.573874 MHz |   yes   |
      SPI1              |PLL3Q    |    24.573874 MHz |   yes   | SPI2S1 for BT PCM
      SPI2              |PLL3Q    |    24.573874 MHz |   yes   | SPI2S2 for HDMI
      SPI3              |PLL3Q    |    24.573874 MHz |   no    |
      DFSDM             |PLL3Q    |    24.573874 MHz |   no    |
      SAI1              |PLL3Q    |    24.573874 MHz |   no    |
      SAI2              |PLL3Q    |    24.573874 MHz |   yes   | AudCodec 48kHz (use PLL3R
for 44.1kHz)
      SAI3              |PLL3Q    |    24.573874 MHz |   no    |
      SAI4              |PLL3Q    |    24.573874 MHz |   no    |
    PLL3R               |PLL3     |    11.290699 MHz |   yes   |
  PLL4                  |HSE      |   594.000000 MHz |   yes   |
    PLL4P               |PLL4     |    99.000000 MHz |   yes   |
      SDMMC1            |PLL4P    |    99.000000 MHz |   yes   | µSD card
      SDMMC2            |PLL4P    |    99.000000 MHz |   yes   | Wifi
      SDMMC3            |PLL4P    |    99.000000 MHz |   yes   | Rpi
      SPDIF             |PLL4P    |    99.000000 MHz |   no    |
    PLL4Q               |PLL4     |    74.250000 MHz |   yes   |
      LCD               |PLL4Q    |    74.250000 MHz |   yes   | LTDC & DSI display pixel
clock
      FDCAN             |PLL4Q    |    74.250000 MHz |   yes   | Should be as high as
possible and < 100MHz
    PLL4R               |PLL4     |    74.250000 MHz |   yes   |
  STGEN                 |HSE      |    24.000000 MHz |   yes   |
  USBPHYC               |HSE      |    24.000000 MHz |   yes   | USB PHY Ctrl for USB Host
and OTG
    USBPLL              |USBPHYC  |    48.000000 MHz |   yes   |
```

```
      USBO              |USBPLL    |   48.000000 MHz |  yes    | USB OTG
CSI                     |N.A.      |    4.000000 MHz |  yes    | Mandatory for IO
compensation
ETH                     |N.A.      |    0.000000 MHz |  no     | ETH clocked by RGMII PHY
------------------------|---------|-----------------|--------
|-------------------------------------|
```

3.2.2.2 *Device tree*

Here are the corresponding device tree **rcc_clk** sub node properties consumed by the first stage boot loader (FSBL) to configure the clock tree above:

```
        st,clksrc = <
                CLK_MPU_PLL1P
                CLK_AXI_PLL2P
                CLK_MCU_PLL3P
                CLK_PLL12_HSE
                CLK_PLL3_HSE
                CLK_PLL4_HSE
                CLK_RTC_LSE
                CLK_MCO1_DISABLED
                CLK_MCO2_DISABLED
        >;

        st,clkdiv = <
                1 /*MPU*/
                0 /*AXI*/
                0 /*MCU*/
                1 /*APB1*/
                1 /*APB2*/
                1 /*APB3*/
                1 /*APB4*/
                2 /*APB5*/
                23 /*RTC*/
                0 /*MCO1*/
                0 /*MCO2*/
        >;

        st,pkcs = <
                CLK_CKPER_HSE
                CLK_FMC_ACLK
                CLK_QSPI_ACLK
                CLK_ETH_DISABLED
                CLK_SDMMC12_PLL4P
                CLK_DSI_DSIPLL
                CLK_STGEN_HSE
                CLK_USBPHY_HSE
                CLK_SPI2S1_PLL3Q
                CLK_SPI2S23_PLL3Q
                CLK_SPI45_HSI
                CLK_SPI6_HSI
                CLK_I2C46_HSI
                CLK_SDMMC3_PLL4P
                CLK_USBO_USBPHY
                CLK_ADC_CKPER
                CLK_CEC_LSE
                CLK_I2C12_HSI
                CLK_I2C35_HSI
                CLK_UART1_HSI
                CLK_UART24_HSI
                CLK_UART35_HSI
                CLK_UART6_HSI
                CLK_UART78_HSI
                CLK_SPDIF_PLL4P
```

```
                CLK_FDCAN_PLL4Q
                CLK_SAI1_PLL3Q
                CLK_SAI2_PLL3Q
                CLK_SAI3_PLL3Q
                CLK_SAI4_PLL3Q
                CLK_RNG1_LSI
                CLK_RNG2_LSI
                CLK_LPTIM1_PCLK1
                CLK_LPTIM23_PCLK3
                CLK_LPTIM45_LSE
        >;

        /* VCO = 1300.0 MHz => P = 650 (CPU) */
        pll1: st,pll@0 {
                cfg = < 2 80 0 0 0 PQR(1,0,0) >;
                frac = < 0x800 >;
                u-boot,dm-pre-reloc;
        };

        /* VCO = 1066.0 MHz => P = 266 (AXI), Q = 533 (GPU), R = 533 (DDR) */
        pll2: st,pll@1 {
                cfg = < 2 65 1 0 0 PQR(1,1,1) >;
                frac = < 0x1400 >;
                u-boot,dm-pre-reloc;
        };

        /* VCO = 417.8 MHz => P = 209, Q = 25, R = 11 */
        pll3: st,pll@2 {
                cfg = < 1 33 1 16 36 PQR(1,1,1) >;
                frac = < 0x1a04 >;
                u-boot,dm-pre-reloc;
        };

        /* VCO = 594.0 MHz => P = 99, Q = 74, R = 74 */
        pll4: st,pll@3 {
                cfg = < 3 98 5 7 7 PQR(1,1,1) >;
                u-boot,dm-pre-reloc;
        };
```

Reset and Clock Control

Low Speed Internal oscillator (STM32 clock source)

Low Speed External oscillator (STM32 clock source)

High Speed Internal oscillator (STM32 clock source) or High Speed Synchronous Serial Interface (MIPI® Alliance standard)

High Speed External oscillator (STM32 clock source)

Multi Speed Internal oscillator (STM32 clock source)

First Stage Boot Loader

Das U-Boot -- the Universal Boot Loader (see U-Boot_overview)

Secondary Program Loader, *Also known as* ***U-Boot SPL***

Read Only Memory

Independent Watchdog

Digital-to-analog converter (Electronic circuit that converts a binary number into a continuously varying value.)

Device Tree Source (in software context) or Digital Temperature Sensor (in peripheral context)

Real Time Clock

Tamper

Consumer Electronics Control (HDMI standard)

Power Management Integrated Circuit

Display Serial Interface (MIPI® Alliance standard)

Analog-to-digital converter. The process of converting a sampled analog signal to a digital code that represents the amplitude of the original signal sample.

USB Host (STM32 specific)

Boot and Security and OTP control

Extended TrustZone Protection Controller

TrustZone address space Controller for DDR

debug and test protocol, named from the Joint Test Action Group that developed it

System Trace Module

Graphics Processing Units

Doubledata rate (memory domain)

System Configuration

voltage reference buffer (STM32 specific)

Hardware Debug Port

Digital Camera Memory Interface

Inter-Processor Communication Controller

External Interrupt

Digital Filter for Sigma-Delta Modulator

Transmit

former spelling for e•MMC ('e' in italic)

Receive

LCD TFT Display Controller (STM32 specific)

System Time Generator

USB On-The-Go (Capability/type of USB port, acting primarily as USB device, to also act as USB host. Also known as USB OTG.)

input/output

Ethernet

Microprocessor Unit

Microcontroller Unit (MCUs have internal flash memory and are intended to operate with a minimum amount of external support ICs. They commonly are a self-contained, system-on-chip (SoC) designs.)

Central processing unit

Discovery kit

BlueTooth

High-Definition Multimedia Interface (HDMI standard)

Stable: 21.02.2020 - 08:35 / Revision: 12.02.2020 - 11:01

A quality version of this page, approved on *21 February 2020*, was based off this revision.

In a first part, this article shows the STM32MP157 line **part number codification** and **block diagram**. STM32MP157 belongs to STM32MP1 Series (refer to the list of part numbers provided below).

The second part of this article digs into technical aspects, and provides entry points to:

- STM32MP15 **documentation**
- articles dedicated to **Internal peripherals** that make the transition towards the software frameworks required to control these peripherals
- the list of **boards** supporting STM32MP15 devices
- the supported **software distributions**, that can be dowloaded into the STM32MP15 device.

Contents

1 Introduction

STM32MP15 microprocessors are based on the Arm®**Cortex®-A7** dual core. They support Trustzone mode for secure operations, a **Vivante GPU** and an Arm®**Cortex®-M4 coprocessor**.

Arm® Cortex®-M4 coprocessor and its peripheral set are directly inherited from the STM32 MCU family [1].

2 Part number codification

The table below shows the STM32MP15 microprocessor different part numbers available, together with their corresponding internal peripherals, security options and packages.

2.1 STM32MP15x lines

	Cortex-A7	Cortex-M4	GPU	Display	CAN
STM32MP15**1**	Single	Yes	No	TFT	No
STM32MP15**3**	Dual	Yes	No	TFT	Yes
STM32MP15**7**	Dual	Yes	Yes	TFT/DSI	Yes

2.2 Security and Cortex-A7 frequency

	Security	Cortex-A7 frequency
STM32MP15x**A**	Basic	650 MHz[2]
STM32MP15x**C**	Secure boot + Cryptography (CRYP)	650 MHz[2]
STM32MP15x**D**	Basic	800 MHz[2][3]
STM32MP15x**F**	Secure boot + Cryptography (CRYP)	800 MHz[2][3]

2.3 Packages

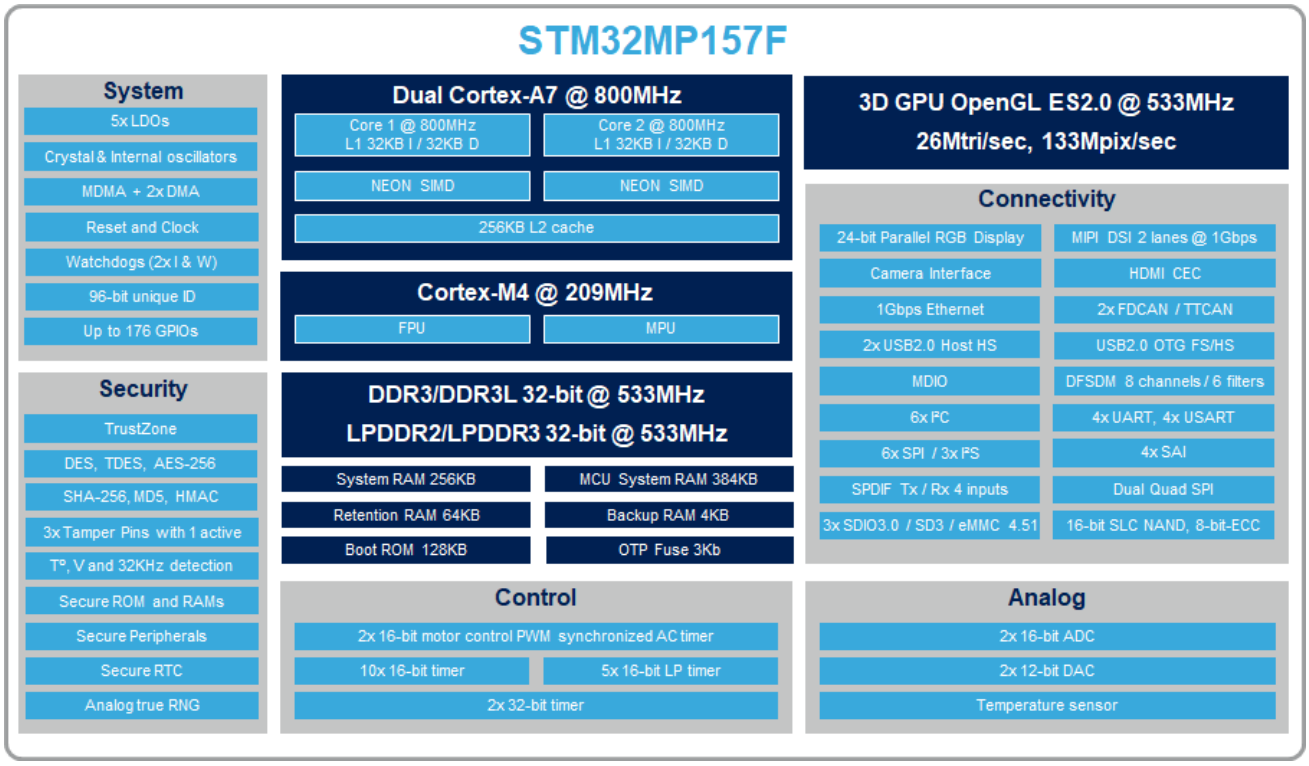
STM32MP15xx**AA**	TFBGA448 18x18
STM32MP15xx**AB**	LFBGA354 16x16
STM32MP15xx**AC**	TFBGA361 12x12
STM32MP15xx**AD**	TFBGA257 10x10

2.4 Junction temperature

STM32MP15xxxx**1**	- 20 to + 105 °C[2][3]
STM32MP15xxxx**3**	- 40 to + 125 °C[2]

3 Block diagram

Here below is the STM32MP157F block diagram offering the richest features set of the STM32MP15 microprocessor.



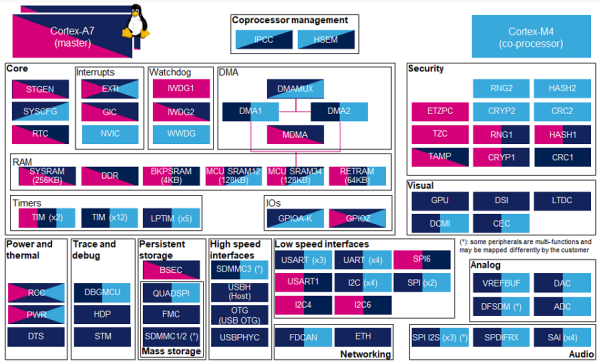
4 Technical documentation

- STM32MP15 Reference Manual: device and internal peripheral user specifications
- STM32MP15 Datasheet: electrical characteristics, package and pinout descriptions

5 Internal peripherals

STM32MP15 peripherals overview article gives a description of all the internal peripherals available on STM32MP15 devices, with direct links to the articles where you can find:

- an overview of each peripheral
- the list of instances available for each peripheral type,
- information on the way each instance can be shared between Arm® Cortex®-A7 and Cortex®-M4 cores,
- direct links to the software frameworks used to control the peripheral from different Arm® cores and security modes such as Cortex®-A7 non secure, Cortex®-A7 secure or Cortex®-M4 (non secure).



Internal peripherals

6 How to get further with STM32MP15 ecosystem

6.1 Boards

The list of boards that integrate STM32MP15 devices can be found in STM32MP15 boards article.

6.2 Supported software distributions

STM32MPU Embedded Software distribution	**STM32MPU Embedded Software distribution for Android**

Click the links above to find information on:

- Distribution composition and associated software architecture
- Associated release notes

7 References and foot notes

- STM32 MCU family
- 2.02.12.22.32.42.5 Exposure to maximum rating conditions for extended periods may affect device reliability. Device mission profile (application conditions) is compliant with JEDEC JESD47 qualification standard. Refer to the STM32MP15 Datasheet and AN5438 for further information.
- 3.03.13.2 800 MHz part numbers are only available with '1' as junction temperatures range suffix (- 20 to + 105 °C).

Graphics Processing Units

Microcontroller Unit (MCUs have internal flash memory and are intended to operate with a minimum amount of external support ICs. They commonly are a self-contained, system-on-chip (SoC) designs.)

Controller Area Network (robust bus mainly used for automotive applications)

Display Serial Interface (MIPI® Alliance standard)