DAC internal peripheral

Contents

Stable: 12.02.2020 - 10:42 | Revision: 12.02.2020 - 10:41

A quality version of this page, approved on *12 February 2020*, was based off this revision.

Contents

1 Article purpose

The purpose of this article is to

- briefly introduce the DAC peripheral and its main features
- indicate the level of security supported by this hardware block
- explain how each instance can be allocated to the three runtime contexts and linked to the corresponding software components
- explain how to configure the DAC peripheral.

2 Peripheral overview

The **DAC** peripheral is a voltage output digital-to-analog converter.

2.1 Features

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

- It may be configured in **8- or 12-bit mode** (data could be left- or right-aligned)
- It has **two output channels**, each with its **own converter**
- The dual DAC channel mode could be done **independently or simultaneously**
- It has built-in noise and triangle **waveform generator** and supports **triggers** for conversions, using: TIM[1], LPTIM[2] or EXTI[3]
- The DAC output buffer allows a high drive output current
- It can operate under **Normal mode** or low-power **Sample and Hold mode** (uses LSI clock, from RCC[4]).
- It may be used in conjunction with the **DMA**[5] controller (with underrun error detection)
- The **common reference voltage**, can be provided by either VREFBUF[6] or any other external regulator[7] wired to VREF+ pin.

2.2 Security support

The DAC is a **non-secure** peripheral.

3 Peripheral usage and associated software

3.1 Boot time

The DAC is not used at boot time.

3.2 Runtime

3.2.1 Overview

The DAC instances can be allocated to:

- the Arm®Cortex®-A7 non-secure core to be used under Linux® with the IIO framework

or

- the Arm®Cortex®-M4 core to be used with STM32Cube Package with STM32Cube DAC driver.

The Peripheral assignment chapter describes which peripheral instance can be assigned to which context.

3.2.2 Software frameworks

Domain	Peripheral	Software components		Comment
OP-TEE	Linux	STM32Cube		
Analog	DAC		Linux IIO framework	STM32Cube DAC driver

3.2.3 Peripheral configuration

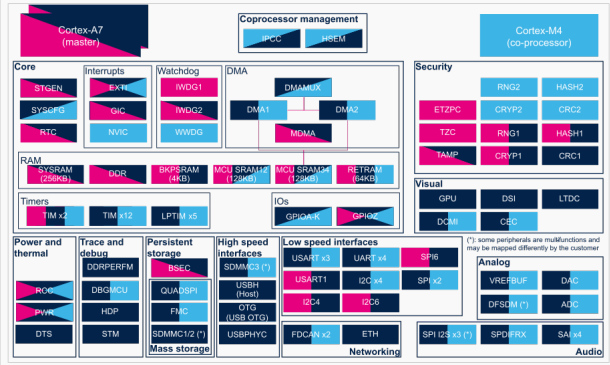
The configuration is applied by the firmware running in the context to which the peripheral is assigned. The configuration by itself can be performed via the STM32CubeMX tool for all internal peripherals. It can then be manually completed (especially for external peripherals) according to the information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

For the Linux kernel configuration, please refer to DAC device tree configuration and DAC Linux driver articles.

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.



Internal peripherals

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.

Domain	Periphera l	Runtime allocation				Comment
Instance	Cortex-A7 secure (OP-TEE)	Cortex-A7 non-secure (Linux)	Cortex-M4 (STM32Cube)			
Analog	DAC	DAC				Assignment (single choice)

4 References

- TIM internal peripheral
- LPTIM internal peripheral
- EXTI internal peripheral
- RCC internal peripheral
- DMA internal peripheral
- VREFBUF internal peripheral
- Regulator overview

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

low-power timer (STM32 specific)

External Interrupt

Low Speed Internal oscillator (STM32 clock source)

Reset and Clock Control

Direct Memory Access

voltage reference buffer (STM32 specific)

Open Portable Trusted Execution Environment