Technical note

SPC58 family - OPC pads management

Introduction

The aim of this document is to give recommendations to the users that need to play with output pin control (OPC) pads in their applications based on STMicroelectronics SPC58 lines devices.

Particular attention is dedicated on the management when entering/exiting from standby given that a hardware feature (independent from the application software the user uses) can create undesired spikes at the standby entry.

The document does not contain codes for programming the application in run mode and standby. It is an explanation of what the user can expect from OPC pads under specific circumstances.

This document is not dedicated to the devices of SPC582B line and SPC58N line because the OPC feature is not present and smart standby wakeup unit (SSWU) is not present as well.

1 Using OPC pads in standby

OPC pads are low-power pads and so they are maintained operative during standby.

The user can choose to modify their logic level once in standby. They use the smart standby wakeup unit (SSWU) to maintain the logic level from run mode by avoiding playing with the latter. This last way can be enforced by disabling OPC management via SIUL2 accordingly (see next paragraphs).

Note: Refer to the SPC58 reference manuals for details about how to set up SSWU to modify the OPC pads logic level, refer to I/O definition files for the list of OPC pads for each device treated in this document. For further information refer to AN4880.

1.1 The role of SIUL2 SCR0 register

It is important to focus on the role of the SoC configuration register (SCR0) before going into detail on the operability of the OPC in run and standby mode.

This register contains two types of field for each OPC pad available in the given device.

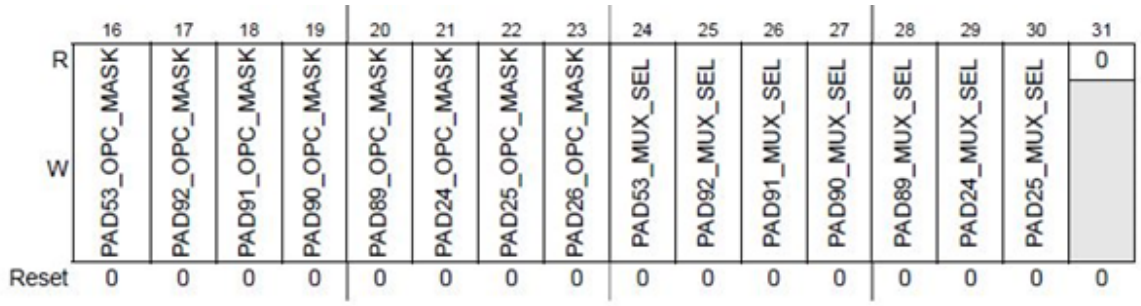
- PADxx_OPC_MASK: set it to avoid the pad is driven as OPC during low power (via SSWU)
- PADxx_MUX_SEL: set the bit to allow the user to change the pad logic level at the standby exit pad.

For all the details, refer to the relative reference manual chapter (see Table 1).

Note: When not explicitly declared, the PADxx_MUX_SEL status in the examples reported in this document has to be considered in the following way:

- PADxx_MUX_SEL = 0 in standby mode
- PADxx_MUX_SEL = 1 in run mode

Figure 1. OPC_MASK and MUX_SEL bits in SIUL2 SCR0 register



Note: Example: SCR0 register on SPC58 C line

1.2 Examples of OPC pad management

This paragraph describes four common situations of usage of OPC pads in run mode and in standby mode.

For all the images in this section, the representation contains two signals:

- Yellow: OPC pad
- Green: low voltage level for indicating the run mode/standby transitions

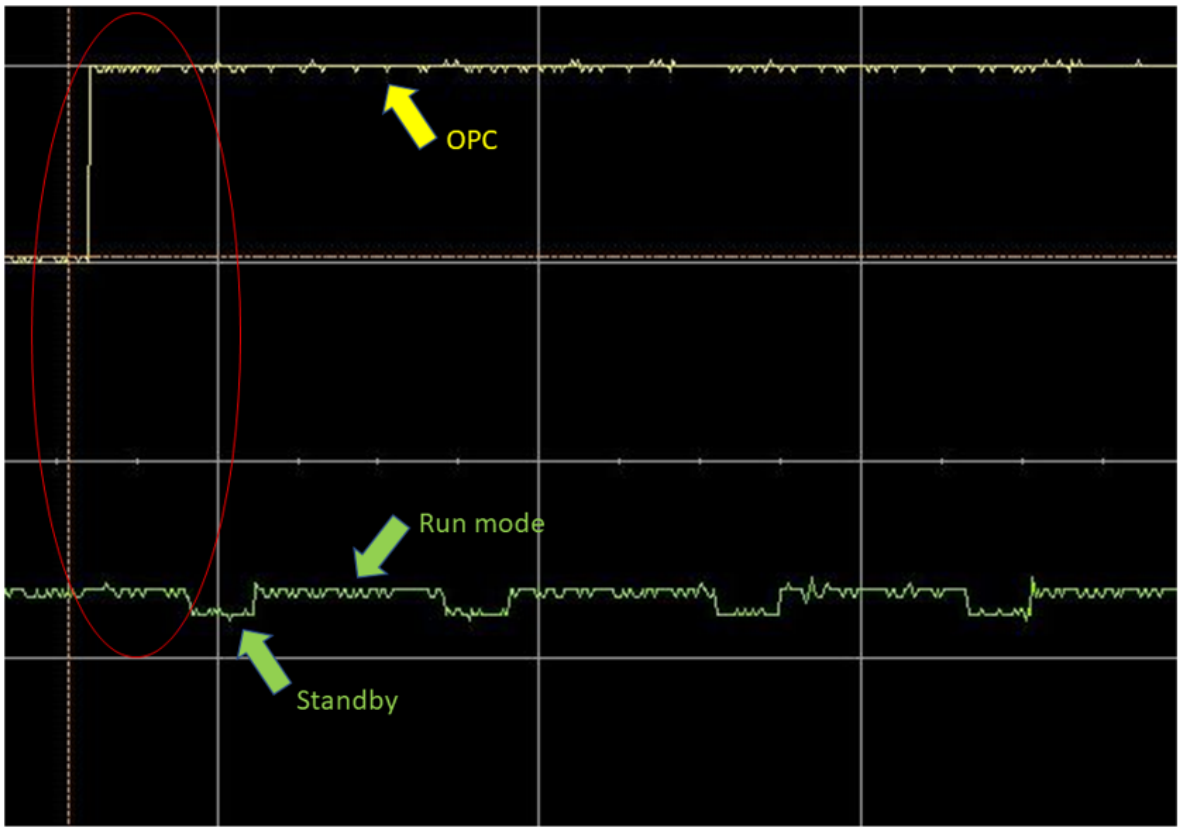
1.2.1 Disable OPC pad in standby

The Figure 2 shows the OPC pad behavior when the OPC feature in standby is disabled (PADxx_OPC_MASK = 1).

This way, once in standby, no action can be performed and the pad level is kept the same as in the play mode.

The Figure 2 shows a representation of the OPC high-level setting in play mode and the input multiple times in standby without changing the logic level of the pad, during multiple play/standby mode transitions.

Figure 2. **Pad logic across standby/run modes with PADxx_OPC_MASK = 1**



1.2.2 Restore OPC pad level across standby entries

By default, each SIUL2.SCR0[PADxx_OPC_MASK] bit is equal to zero. This means that the SSWU can drive the status of the OPC pads during standby mode too. By design, the device restores the previous OPC standby exit level before entering the next standby mode. This means that if the OPC level is low at the standby output, it is moved to the low level at the next standby input by hardware (regardless of the OPC level that has driven in run mode).

Particular attention is required by the user when configuring the OPC pad.

The figure below shows the following example: three run modes, three standby modes in series

- PAD is set high in each run mode
- PAD is maintained high in the first standby entry
- PAD is driven low in the second standby entry (via SSWU OPC feature)
- PAD is driven high in the third standby entry (via SSWU OPC feature)

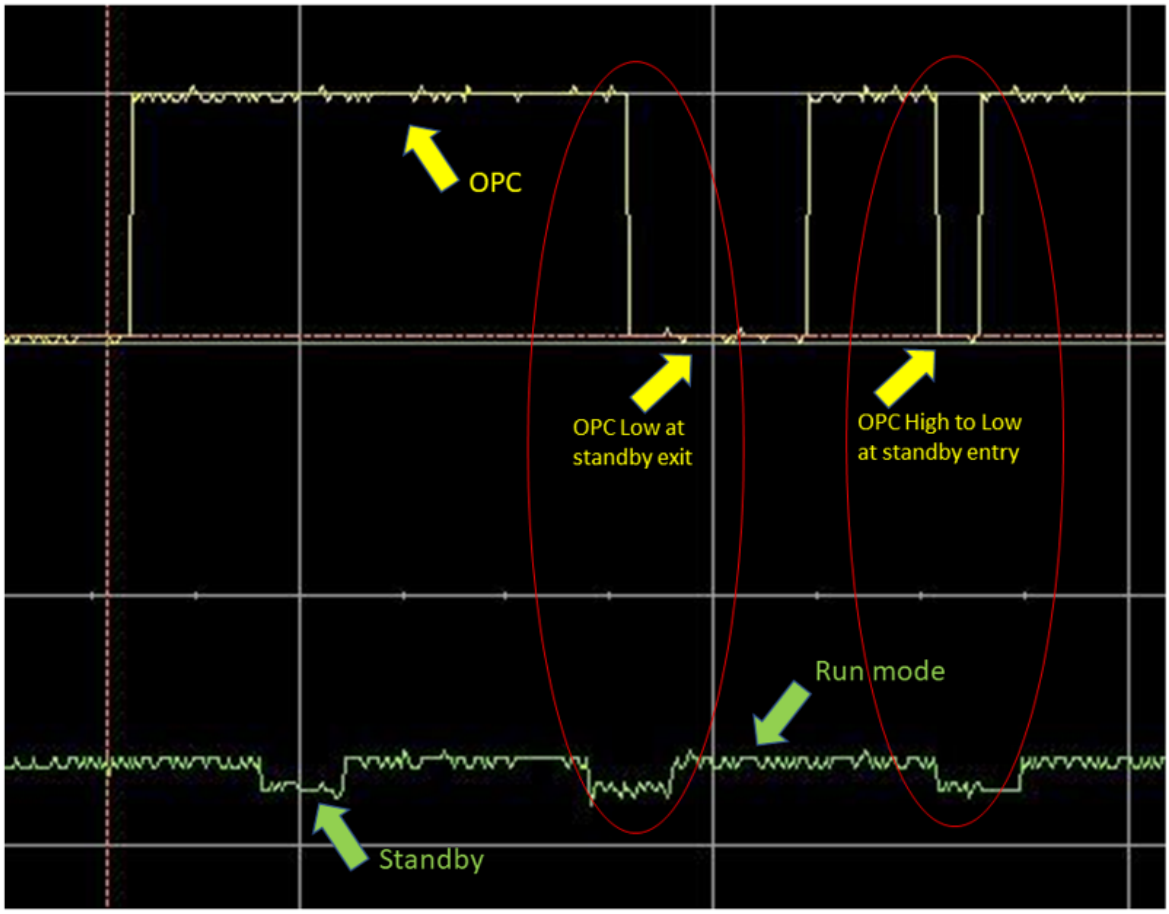
As the image reports, the pad is automatically moved to low when entering the third standby, as the exit status of the previous one was low.

This is made by hardware. It is a feature and not a bug of the internal design.

This behavior is explained in every RM in the SSWU chapter (OPC related paragraph).

So, the recommendation for the user that wants to drive the OPC pads in standby preventing to add unwanted spikes, is to manage the pads in standby mode only: “don' t change OPC pad level via SW in run mode”.

Figure 3. **OPC pad changes status depending on standby exit level**



1.2.3 The role of PADxx_MUX_SEL in standby

Maintaining the PADxx_MUX_SEL set before going into standby creates a spike to pad on standby exit.

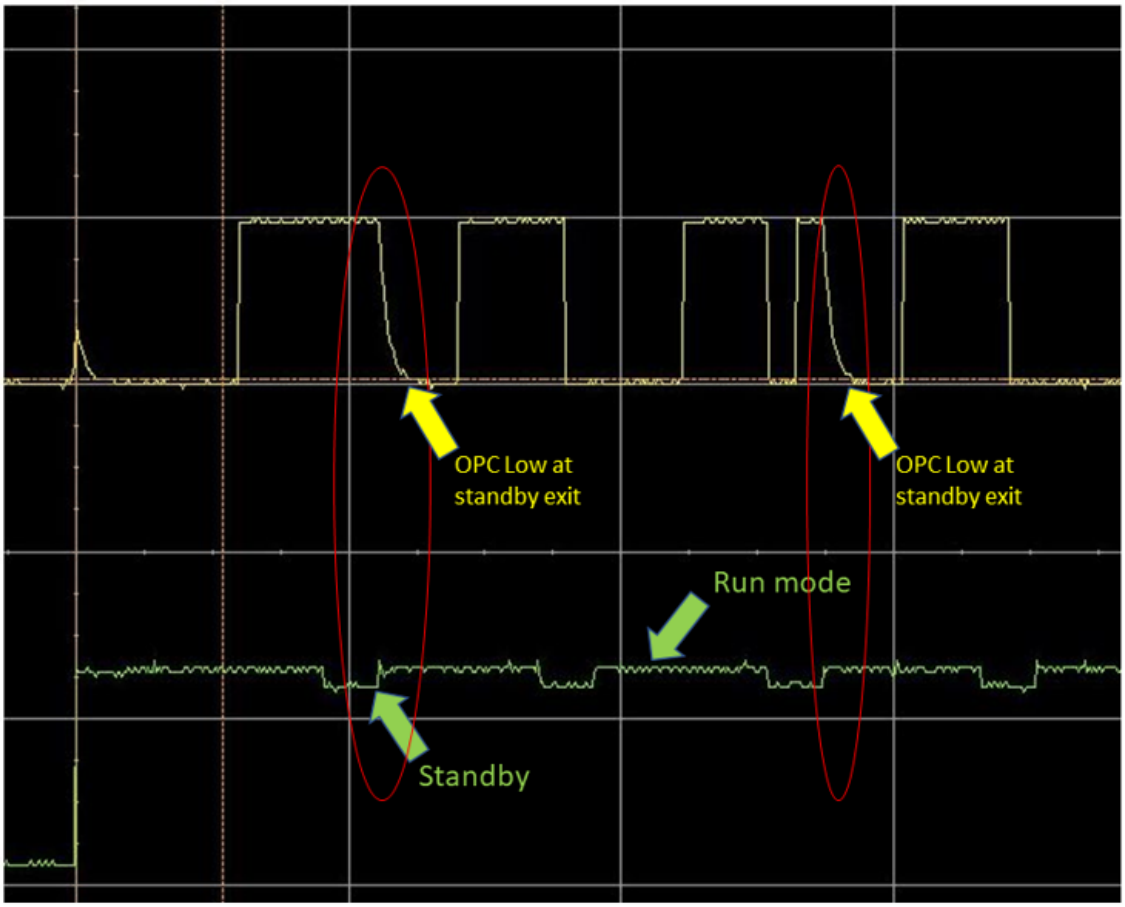
See the next example for a better understanding: on that one, OPC is set high on every run mode and PADxx_MUX_SEL is always set to 1, in run mode and in standby mode.

- First standby: OPC is maintained high
- Second standby: OPC is driven low (via SSWU)
- Third standby: OPC is driven high (via SSWU)
- Fourth standby: OPC is driven low (via SSWU)

Apart from the changes to the logic level of the pad as explained in the previous paragraph, we see that the pad is brought low to the first and the third standby outputs. This creates undesired spikes that can be avoided by setting the PADxx_MUX_SEL to zero before entering in standby.

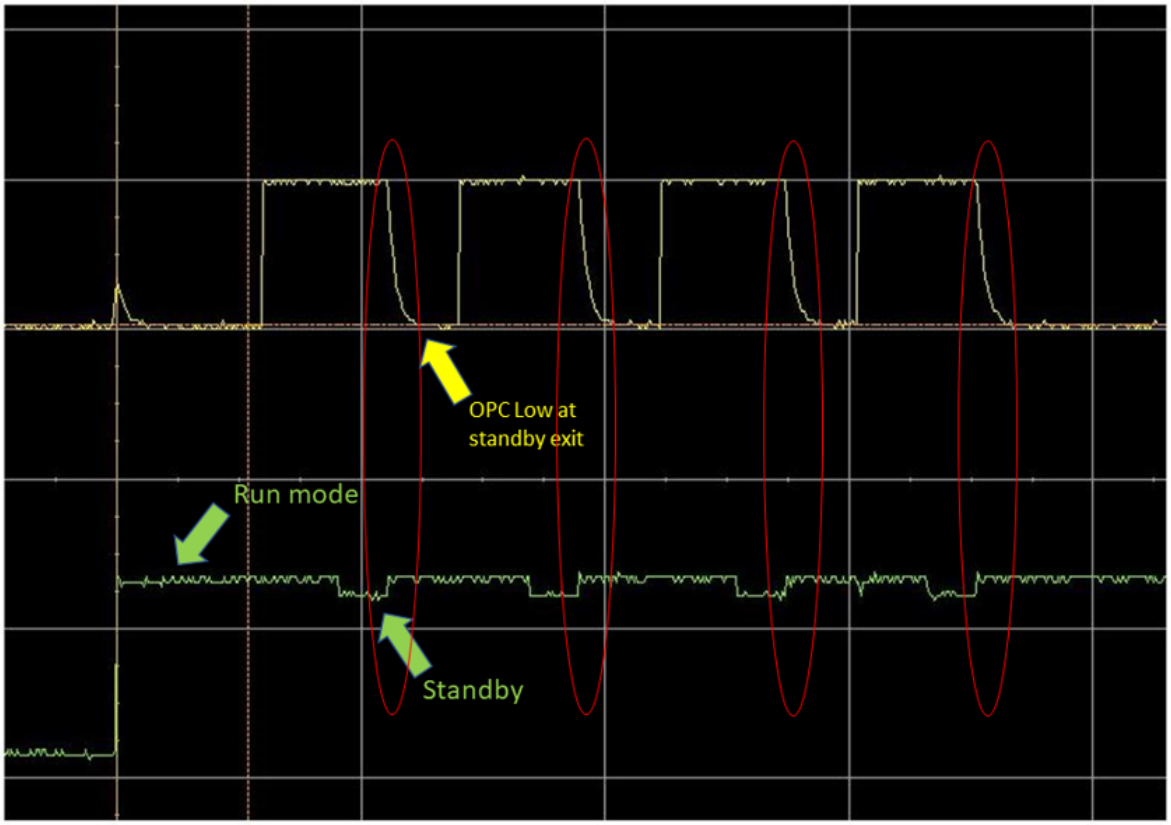
See the figure below for a graphical representation.

Figure 4. **How PADxx_MUX_SEL = 1 drives OPC at standby exit**



As a further (and last) example, the following figure also shows the same behavior when maintaining the PADxx_MUX_SEL = 1 also with PADxx_OPC_MASK = 1. In this case, the OPC pad is not driven in standby and the exact application ran in the previous example generates different waves given that the pad is not driven in standby.

Figure 5. **PADxx_MUX_SEL = 1 and PADxx_OPC_MASK = 1 example**



The recommendation in those cases is to maintain the PADxx_MUX_SEL = 0 when entering standby, this avoids undesired spikes.

1.3 Final considerations

The examples reported in this document have the input to clarify what can happen if the OPC pad is driven both in run mode and standby mode.

The best way to avoid undesired spikes (if they are undesired for the user) is to avoid OPC management outside standby and maintain the PADxx_MUX_SEL to zero before moving to standby mode, as OPC feature is mainly a standby feature for using the pad to open/close external switches.

Appendix A Reference documents

Table 1. **Reference documents**

Document name	Title
RM0449	SPC58 4B line - 32-bit power architecture automotive MCU z4 core 120 MHz, 2 MBytes Flash, HSM, ASIL-B
RM0407	SPC58 C line - 32-bit power architecture automotive MCU dual z4 cores 180 MHz, 4 MBytes Flash, HSM, ASIL-B
RM0391	SPC58 E/G line - 32-bit power architecture automotive MCU triple z4 cores 180 MHz, 6 MBytes Flash, HSM, ASIL-D
RM0452	SPC58 H line - 32-bit power architecture automotive MCU triple z4 cores 200 MHz, 10 MBytes Flash, HSM, ASIL-D
AN4880	SPC58xx hardware design guideline

Revision history

Table 2. **Document revision history**

Date	Revision	Changes
12-Apr-2022	1	Initial release.

Contents

List of tables

List of figures