SPC58x Power Down / Up test cycling

Introduction

During the testing of an automotive Microcontroller it can be necessary to check the stability and robustness of the whole system and application under critical conditions. An example is given by the case when the power goes down and then up continuously. This could emulate an issue on battery system.

The purpose of this technical note is to show how to set up such kind of test scenario on a development laboratory environment.

In this experiment, the device under testing (DUT) is the SPC584Cx/SPC58ECx discovery board where it is expected that the application stored in the FLASH will start without failing during the power down and up cycles.

1 Hardware setup

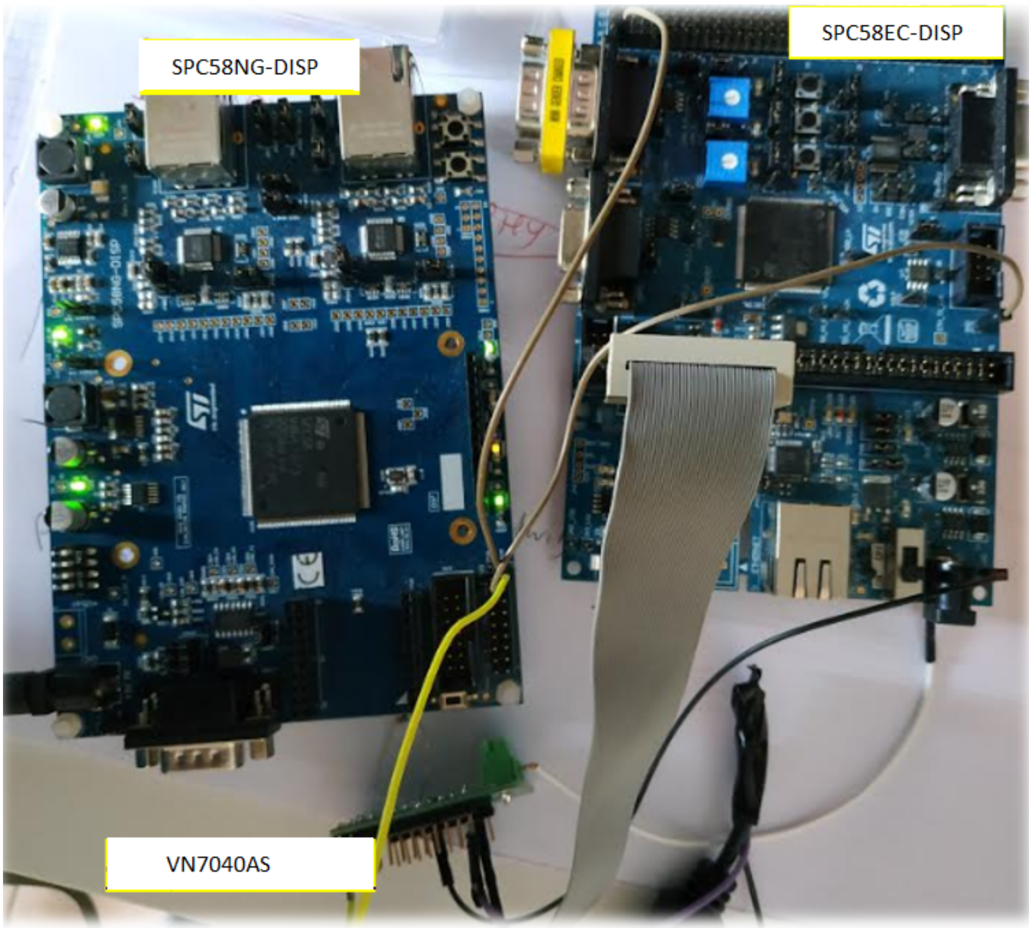
The hardware setup is composed by three boards:

- The SPC58EC-DISP board is designed for the SPC58EC80E Microcontroller and provides access to GPIO and peripherals such as ISO CAN FD, Ethernet, FlexRay, LIN, UART.
- The SPC58NG-DISP for SPC58 Chorus G line Microcontroller.
- VN7040AS evaluation board that is designed to drive 12 V automotive grounded loads through a 3 V and 5 V CMOS-compatible interface and to provide protection and diagnostics.

Figure 1. Hardware setup

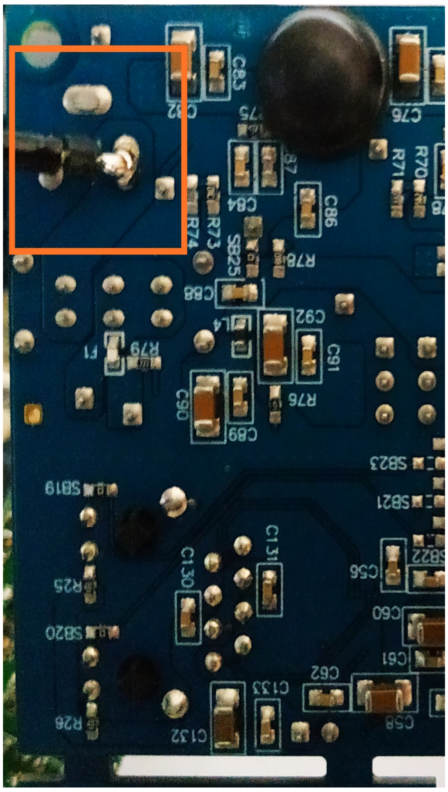
All board grounds need to be wired together.

The following wiring is needed to implement the system:



- On the SPC58EC-DISP board, the A2 pin on X9 plug must be wired to P100[10] on SPC58NG-DISP. This is the Control PIN to ask SPC58NG SW to switch-off the SPC58EC-DISP.
- On the SPC58NG-DISP the P100[9] is wired to the VN7040AS J3 pin 13.
- On the VN7040AS the VBAT from J1 is soldered to the SPC58EC-DISP as shown in the following figure.

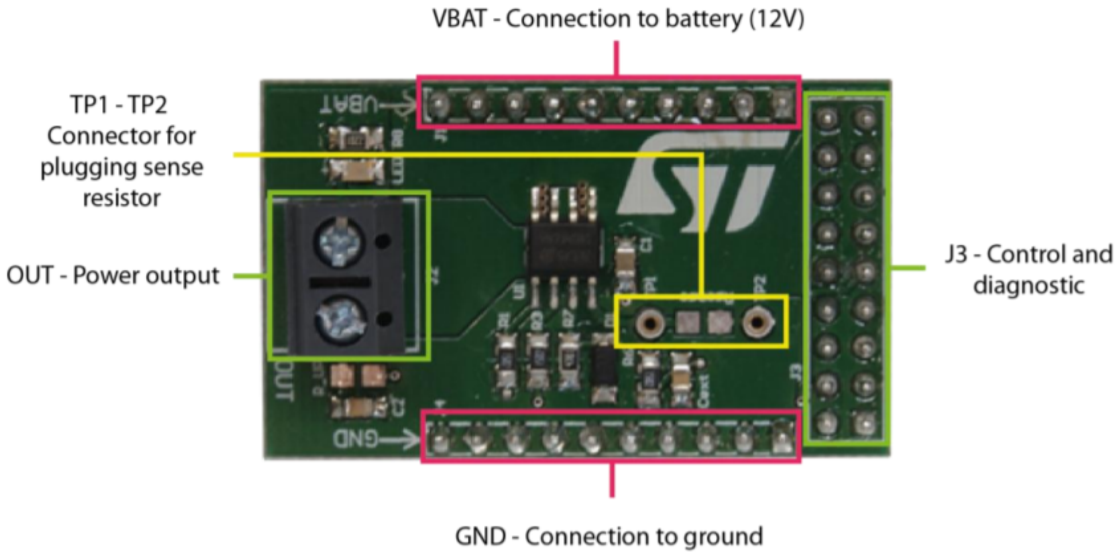
Figure 2. **SPC58EC-DISP bottom view**



1.1 VN7040AS setup

The 12 V power supply has to provide to the eval board as shown in the following figure, one of the available points in the J1, same as ground on J4. On J3 the signal 13 controls OUT switch state. It will be wired to the GPIO at 3.3V (SPC58NG-DISP the P100[9]).

Figure 3. **VN7040AS setup**



2 Applications

The main application runs on the SPC58EC80E microcontroller that is the DUT. This must be loaded in the FLASH device.

The purpose of the project is to power down and up the SPC58EC-DISP board.

According to the user needs, an unexpected failure (maybe to the related power) can be detected.

The whole software has been developed by using SPC5Studio (/www.st.com/en/development-tools/spc5-studio.html)

As introduced in the previous chapter, the SPC58NG-DISP is used to drive the VN7040AS board.

When the application on SPC58NG-DISP starts, the CONTROL GPIO that is the PF[10] wired to the SPC58EC-DISP board. At the beginning this is at 0 logic so the LED1 will toggle and the PF[9] will be set to allow VN7040AS to power-up the SPC58NG-DISP board.

The PF[9] is configured as OUTPUT while the PF[10] is configured as INPUT. The reader can find all the necessary information inside the device's reference manual.

```
for ( ; ; ) {
// P100[10] -> PF12 -> CONTROL_IN
uint32_t value = pal_readpad(PORT_F, CONTROL_IN);
if ( value == 0) {
pal_writepad(PORT_F, POWER, 1);
pal_lld_togglepad(PORT_A, PA_LED1);
} else {
/// If Control is ON ==> POWER
// P100[9] -> PF11 -> POWER
pal_lld_togglepad(PORT_A, PA_LED3);
pal_writepad(PORT_F, POWER, 0);
}
}
```

Where

```
#define CONTROL_IN 12U
#define POWER 11U
#define PORT_POWER PORT_F
```

Please refer to the API described in SPC5Studio for MCU programming.

As soon as the SPC58EC-DISP is powered by the VN7040AS, the application will start from FLASH.

This will program a PIT timer instance, its Interrupt service routine, invoked as soon as the timer expires, will write 1 on the output pin PD[10] (A2 on X9).

This will allow the SPC58NG-DISP to power off the SPC58EC-DISP through the VN7040AS.

The following is a portion of code running on SPC58EC80E Microcontroller. The main function calls the Pit_Init.

As soon as the timer (channel 1) expires the related ISR will invoke the power-down function:

```
#define PIT1_CHANNEL 1U

void power_down (void)
{
pal_lld_togglepad(PORT_F, PF_LED3);
// POWER is PD[10] so A2 on X9 wired to the second board
SIUL2.GPDO[58].R = 1;
}

IRQ_HANDLER(SPC5_PIT0_CH1_HANDLER)
{
power_down();
osalExitCriticalFromISR();
```

```
IRQ_EPILOGUE();
}
static void Pit_Init(void)
{
uint32_t val, clk = SPC5_PER_CLK;
INTC_PSR(SPC5_PIT0_CH1_INT_NUMBER) =
INTC_PSR_ENABLE(SPC5_SYSTIMER_CORE, SPC5_SYSTIMER_IRQ_PRIORITY);
SPCSetPeripheralClockMode(SPC5_PIT0_PCTL,
SPC5_ME_PCTL_RUN(2) | SPC5_ME_PCTL_LP(2));
// LDVAL trigger = (period/clock period)-1: e.g. 1s = 1 / (1/ 25MHz)
// 1ms = (1/ 1000) / (1/ 25MHz) where 25Hz is the PER_CLK in this app
val = ((uint32_t)clk) - 1U;
PIT_0.MCR.B.MDIS = 0U; //Enable clocks
PIT_0.CH[PIT1_CHANNEL].TCTRL.B.TEN = 0U; //Disable the timer
PIT_0.CH[PIT1_CHANNEL].LDVAL.B.TSV = val; //Configure the div factor
PIT_0.CH[PIT1_CHANNEL].TCTRL.B.TIE = 1U; //Enable the timer interrupt
PIT_0.CH[PIT1_CHANNEL].TCTRL.B.TEN = 1U; //Enable the timer
}
```

In case of an event is detected, to debug the status, the PIT can be stopped, a led can blink and the SW can loop in a while to allow user to connect the debugger and check the status of the MCU.

```
SIUL2.GPDO[58].R = 0;
PIT_0.CH[1].TCTRL.B.TEN = 0U; //Disable the timer
PIT_0.CH[1].TCTRL.B.TIE = 0U;
while(1) {
pal_lld_togglepad(PORT_F, PF_LED2);
};
```

Appendix A Acronyms and abbreviations

Table 1. **Acronyms and abbreviations**

Terms	Meaning
DTU	Device under testing
PIT	Periodic interrupt timer
MCU	Micro-controller unit

Appendix B Reference documents

- RM0407 reference manual
- RM0391 reference manual
- SPC584Gx, SPC58EGx, SPC58NGx datasheet
- SPC584Cx, SPC58ECx datasheet
- EV-VN7040AS evaluation board data brief

Revision history

Table 2. Document revision history

Date	Version	Changes
03-Nov-2020	1	Initial release.

Contents

1 **Hardware setup** 2
1.1 VN7040AS setup 3
2 **Applications** 4
Appendix A **Acronyms and abbreviations** 6
Appendix B **Reference documents** 7
Revision history 8
Contents 9