Nordic Semiconductor ASA
Tel.: +47 72 89 89 00
Fax: +47 72 89 89 89
Reg. nr./VAT no.: 966011726

Product Anomaly Notification (PAN)

Device affected (product name):	**Active device version(s)**:
nRF51822-QFAA, nRF51422-QFAA, nRF51822-QFAB, nRF51822-CEAA	nRF51422-QFAA CA nRF51422-QFAA C0 nRF51822-QFAA CA nRF51822-QFAA C0 nRF51822-QFAB AA nRF51822-QFAB A0 nRF51822-CEAA B0
Date (YYYY-MM-DD): 2013-05-15	**PAN no.:** PAN-028
Nordic Semiconductor reference: Thomas Embla Bonnerud	**Document version**: 1.6

Summary

This document lists all known anomalies that exist in the nRF51 series and which device and device version any given anomaly applies to.

Marking / tracing:

nRF51822-QFAA, nRF51822-QFAB, nRF51422-QFAA, and nRF51822-CEAA:

Build: CA

Build: C0

Build: AA

Build: A0

Build: CA

Build: C0

Build: B0

Where the letters on the last line of the chip marking means:

- Y = Year assembly marking, e.g. YY=11
- W = Week assembly marking, e.g. WW=35
- L = Wafer lot, step letters for each lot, e.g. LL={AA, AB,…,AZ, BA,…,ZZ, AA, AB,…}

Authorization for Nordic Semiconductor

Product Manager	Date:	Sign:
Thomas Embla Bonnerud	15/05/2013	

Nordic Semiconductor ASA
Tel.: +47 72 89 89 00
Fax: +47 72 89 89 89
Reg. nr./VAT no.: 966011726

Changelog:

Version	Change
1.6	Added: “CPU: The CPU may fail to wake up if WFI are called from ISR’s”
	Restructured Anomaly overview and moved “Device version affected” from each individual PAN to a common table.
	Added: “TWI: Consumes too much current when it is enabled and the STOP task is triggered”
	Added nRF51822CEAA device.
1.5	Updated “RNG: RNG does not generate a new number after the current number generated.”
	Updated “System: Programmer/Debugger is unable to discover the Cortex-M SW device”
	Added: “System: System OFF and System ON current higher than specified”
	Small nonfunctional grammatical changes to some text.
	Removed one condition from “System: Programmer/Debugger is unable to discover the Cortex-M SW device”
1.4	Added nRF51422 QFAA C0 to “Active device version(s)”
1.3	Updated Device list
1.2	Added detailed description of “HFCLK: XTALFREQ register is not functional”
	Updated “DIF: Missing pull down on SWDCLK” with recommended pull down value
	Updated workaround for “System: Debugger is unable to discover device in some cases”
	Updated “TIMER: BITMODE is not functional for TIMER0”
	Added “POWER: RAMON reset value causes problems under certain conditions”
	Minor clarifications on some PAN’s
	Updated workaround for “LFCLK: Calibration does not request HFCLK”
	Added “ADC: Setting ENABLE register to Disabled does not release the pins captured for GPIO.”
	Added “RNG: RNG does not generate a new number after the current number generated.”
	Removed “The output value of the pin cannot be controlled by GPIOTE when the pin is configured as event generator”
	Removed “RTC and WDT: Reset on 32KiHz domains delayed when peripherals turned off and 32KiHz clock is running”
	Updated Build code for nRF51822
1.1	Changed sorting of PANS to alphabetical sorting
	Added: “GPIO: SENSE mechanism fires under some circumstances when it should not.”
	Added code example to: “GPIOTE: OUTINIT field in CONFIGn register is not functional”
	Added: “HFCLK: XTALFREQ register is not functional”
	Added: “System: Issues with disable system OFF mechanism”
	Added: “TEMP: Negative measured values are not represented correctly.”
	Added: “TEMP: STOP task clears the TEMP register.”
	Added: “TEMP: TEMP module analog front end do not power down when DATARDY event occurs.”
	Replaced “TEMP: The module is not functional” with “TEMP: Temperature offset value has to be manually loaded to the TEMP module”
	Added information on which PAN’s that are planned fixed for the next version of the IC

Nordic Semiconductor ASA
Tel.: +47 72 89 89 00
Fax: +47 72 89 89 89
Reg. nr./VAT no.: 966011726

Anomaly overview

#	Description	nRF51422-QFAA CA	nRF51422-QFAA C0	nRF51822-QFAA CA	nRF51822-QFAA C0	nRF51822-QFAB AA	nRF51822-QFAB A0	nRF51822-CEAA B0	PAN will be fixed
1.	ADC: ADC module analog front end does not power down when END event occurs	X	X	X	X	X	X	X	X
2.	ADC: STOP task clears the RESULT register	X	X	X	X	X	X	X	X
3.	ADC: Setting ENABLE register to Disabled does not release the pins captured for GPIO.	X	X	X	X	X	X	X	X
4.	ANT: Erroneous packet transmission from ANT slave channel	X							X
5.	ANT: Bursting on a non-tracking channel	X							X
6.	CPU: The CPU may fail to wake up if WFI are called from ISR's	X	X	X	X	X	X	X	X
7.	DIF: Missing pull down on SWDCLK	X	X	X	X	X	X	X	X
8.	GPIO: SENSE mechanism fires under some circumstances when it should not.	X	X	X	X	X	X	X	X
9.	GPIOTE: OUTINIT field in CONFIGn register is not functional	X	X	X	X	X	X	X	X
10.	GPIOTE: The module cannot receive tasks or detect transitions on pad the first 3 clock periods after being enabled	X	X	X	X	X	X	X	X
11.	HFCLK: Base current with HFCLK running is too high	X	X	X	X	X	X	X	X
12.	HFCLK: Clock is paused when switching clock source for HFCLK clock	X	X	X	X	X	X	X	X
13.	HFCLK: XTALFREQ register is not functional	X	X	X	X	X	X	X	X
14.	LFCLK: Calibration does not request HFCLK	X	X	X	X	X	X	X	X
15.	POWER: It is not possible to distinguish between Power on reset and reset from off by RESETREAS	X	X	X	X	X	X	X	X
16.	POWER: RAMON reset value causes problems under certain conditions	X	X	X	X	X	X	X	X
17.	RADIO: END to START connection using PPI or short is not functional	X	X	X	X	X	X	X	X
18.	RADIO: RSSI module analog front end does not power down when RSSIEND event occurs	X	X	X	X	X	X	X	X
19.	RADIO: RSSISTOP task clears the RSSISAMPLE register	X	X	X	X	X	X	X	X
20.	RADIO: State Register is not functional	X	X	X	X	X	X	X	X

Nordic Semiconductor ASA
Tel.: +47 72 89 89 00
Fax: +47 72 89 89 89
Reg. nr./VAT no.: 966011726

#	Description	nRF51422-QFAA CA	nRF51422-QFAA C0	nRF51822-QFAA CA	nRF51822-QFAA C0	nRF51822-QFAB AA	nRF51822-QFAB A0	nRF51822-CEAA B0	PAN will be fixed
21.	RNG: Generated random value is reset when VALRDY event is cleared	X	X	X	X	X	X	X	X
22.	RNG: RNG does not generate a new number after the current number generated.	X	X	X	X	X	X	X	X
23.	RNG: STOP task clears the VALUE register	X	X	X	X	X	X	X	X
24.	RNG: The STOP task cannot be assigned to a PPI channel	X	X	X	X	X	X	X	X
25.	System: Programmer/Debugger is unable to discover the Cortex-M SW device	X	X	X	X	X	X	X	X
26.	System: Manual setup is required to enable use of peripherals	X	X	X	X	X	X	X	X
27.	System: System OFF and System ON current higher than specified	X	X	X	X	X	X	X	X
28.	TEMP: Negative measured values are not represented correctly.	X	X	X	X	X	X	X	X
29.	TEMP: STOP task clears the TEMP register.	X	X	X	X	X	X	X	X
30.	TEMP: TEMP module analog front end does not power down when DATARDY event occurs.	X	X	X	X	X	X	X	X
31.	TEMP: Temperature offset value has to be manually loaded to the TEMP module	X	X	X	X	X	X	X	X
32.	TIMER: BITMODE is not functional for TIMER0	X	X	X	X	X	X	X	X
33.	TIMER: One CC register is not able to generate an event for the second of two subsequent counter/ timer values.	X	X	X	X	X	X	X	X
34.	TIMER: Timer cannot handle quick START-STOP-START tasks correctly	X	X	X	X	X	X	X	X
35.	TWI: Consumes too much current when it is enabled and the STOP task is triggered	X	X	X	X	X	X	X	X
36.	TWI: Shortcuts described in nRF51 Reference Manual are not functional	X	X	X	X	X	X	X	X
37.	UART: After a STOPRX task the UART will not be able to finish transaction	X	X	X	X	X	X	X	X
38.	WDT: The watchdog config option "RUN while paused by the debugger" does not work	X	X	X	X	X	X	X	X

Nordic Semiconductor ASA
Tel.: +47 72 89 89 00
Fax: +47 72 89 89 89
Reg. nr./VAT no.: 966011726

1. ADC: ADC module analog front end does not power down when END event occurs

Symptoms:

Higher current consumption.

Conditions:

Always.

Consequences:

Higher current consumption.

Workaround:

Trigger STOP task to power down analog front end and reduce power consumption.

2. ADC: STOP task clears the RESULT register

Symptoms:

When STOP task is triggered, VALUE register is cleared.

Conditions:

Always.

Consequences:

If STOP task is triggered before reading the converted value in RESULT, the value will be lost and read as 0x00.

Workaround:

Read RESULT before triggering STOP task.

3. ADC: Setting ENABLE register to Disabled does not release the pins captured for GPIO.

Symptoms:

Pins used previously for the ADC cannot be used as GPIO.

Conditions:

After assigning those pins to the ADC and enabling the ADC.

Consequences:

Pins cannot be used as GPIOs after being used as analog pins for the ADC.

Workaround:

Execute the following code after setting ADC.ENABLE register to Disabled.

```
NRF_ADC->CONFIG = (ADC_CONFIG_RES_8bit                          << ADC_CONFIG_RES_Pos)       |
                  (ADC_CONFIG_INPSEL_SupplyTwoThirdsPrescaling << ADC_CONFIG_INPSEL_Pos)    |
                  (ADC_CONFIG_REFSEL_VBG                        << ADC_CONFIG_REFSEL_Pos)    |
                  (ADC_CONFIG_PSEL_Disabled                     << ADC_CONFIG_PSEL_Pos)      |
                  (ADC_CONFIG_EXTREFSEL_None                    << ADC_CONFIG_EXTREFSEL_Pos);
```

4. ANT: Erroneous packet transmission from ANT slave channel

Symptoms:

When running an ANT slave channel in a multi-channel environment, unintended packet transmissions from an ANT slave channel may be received by an ANT master channel.

Conditions:

Condition can occur in a multi-channel scenario (on the same device) where a slave channel and 1 or more channels are assigned with the same channel ID (device number, device type and transmission type).

Consequences:

The peer device running the ANT master channel may receive unintended transmission packets from ANT slave channel on the reverse channel direction even though no packet transmission was initiated by the ANT slave channel.

Workaround:

Avoid setting the same channel ID (device number, device type and transmission type) for multiple channels on the same device.

5. ANT: Bursting on a non-tracking channel

Symptoms:

Attempting to send a burst transfer when the ANT channel is not in a tracking state will result in the burst transfer not completing and will prevent any future tx burst transfers from succeeding.

Conditions:

Sending a burst transfer on a channel when it is not in the tracking state (i.e. when the channel is either searching or not opened).

Consequences:

The ant_burst_handler_request() API functionality will be in a stuck state. Any future burst requests will automatically fail with a SVC return error and/or no burst transmission being performed. Supplied burst handler wait flag will not be cleared. EVENT_TRANSFER_NEXT_DATA_BLOCK event will not be generated.

Workaround:

Before sending a burst transfer, ensure that the channel is in a tracking state by issuing the ant_channel_status_get() API call. If the ANT burst request handler becomes stuck, a stack reset (ant_stack_reset()) will be required to restore the API functionality.

6. CPU: The CPU may fail to wake up if WFI are called from ISR's

Symptoms:
CPU may skip interrupts or not wake up from sleep mode after calling Wait For Interrupt (WFI) from an ISR.

Conditions:
CPU is put in sleep mode by Wait For Interrupt (WFI) inside an Interrupt Service Routine (ISR).

When the CPU is in sleep mode, the nRF51 has a HW mechanism to ensure that interrupts from peripherals with equal or lower priority will NOT wake up the CPU.
When a new interrupt arrives, the HW mechanism will use 64 HFCLK cycles to evaluate the priority before it decides whether to wake up the CPU or mask the event, keeping the CPU in sleep mode.

When this HW mechanism masks an interrupt of equal or lower priority, any interrupt arriving in the last of the 64 HFCLK cycles will also be masked, regardless of its priority.

Consequences:
Masking a second interrupt, regardless of priority, may result in the CPU remaining in sleep when it should have woken up on this interrupt. Subsequent events from the same peripheral will also fail to wake up the CPU.

Workaround:

- Only call WFI from Thread mode.

Or

- Ensure that no lower-priority interrupts are enabled in the peripherals before calling WFI.
  Note: The interrupts have to be disabled in the peripherals, not in the NVIC.

7. DIF: Missing pull down on SWDCLK

Symptoms:

- Pin reset function may not work
- High current consumption
- Device is hanging

Conditions:

Always.

Consequences:

- Pin reset function may not work
- High current consumption
- Device is hanging

Workaround:

Add external 12k pull down on SWDCLK pin.
If no programming or debug is ever used the SWDCLK can be connected to GND.

Note: The external resistor shall not be mounted on PCB's when a device that has this PAN fixed is used.

8. GPIO: SENSE mechanism fires under some circumstances when it should not.

Symptoms:

Sometimes a PORT event is generated when it should not have been generated.

Conditions:

Pre-Condition: Input buffer is disabled.
Operation: Connect the input buffer and enable the sense functionality on the pad in the same write operation to PIN_CFG

Consequences:

False interrupt and/or PORT event might be triggered at write to PIN_CNF register.

Workaround:

Always enable the input buffer in a separate write operation, before enabling the sense functionality.

9. GPIOTE: OUTINIT field in CONFIGn register is not functional

Symptoms:

Initial value for GPIOTE output after configuration is undefined.

Conditions:

Configuring a GPIOTE channel as a task.

Consequences:

Application specific.

Workaround:

1. Configure the GPIOTE channel as follows:
   - MODE: TASK
   - PSEL: Set to unused output pin.
   - POLARITY: LOTOHI if initial high desired, or HITOLO if initial low desired.
2. Trigger the OUT task
3. Reconfigure the GPIOTE channel as follows:
   - MODE: TASK
   - PSEL: <GPIO used for function>
   - POLARITY: <Desired polarity (Toggle, LoToHi or HiToLo>
   - OUTINIT: <Desired initial value>

The following inline function can be used to perform this action:

```
static __INLINE void nrf_gpiote_task_config(uint32_t channel_number, uint32_t pin_number,
nrf_gpiote_polarity_t polarity, nrf_gpiote_outinit_t initial_value)
{
    /* Check if the output desired is high or low */
    if (initial_value == GPIOTE_CONFIG_OUTINIT_Low)
    {
        /* Configure channel to Pin31, not connected to the pin,
           and configure as a tasks that will set it to proper level */
        NRF_GPIOTE->CONFIG[channel_number] =
            (GPIOTE_CONFIG_MODE_Task       << GPIOTE_CONFIG_MODE_Pos)     |
            (31UL                          << GPIOTE_CONFIG_PSEL_Pos)     |
            (GPIOTE_CONFIG_POLARITY_HiToLo << GPIOTE_CONFIG_POLARITY_Pos);
    }
    else
    {
        /* Configure channel to Pin31, not connected to the pin,
           and configure as a tasks that will set it to proper level */
        NRF_GPIOTE->CONFIG[channel_number] =
            (GPIOTE_CONFIG_MODE_Task       << GPIOTE_CONFIG_MODE_Pos)     |
            (31UL                          << GPIOTE_CONFIG_PSEL_Pos)     |
            (GPIOTE_CONFIG_POLARITY_LoToHi << GPIOTE_CONFIG_POLARITY_Pos);
    }

    /* Three NOPs are required to make sure configuration
       is written before setting tasks or getting events */
    __NOP();
    __NOP();
    __NOP();

    /* Launch the task to take the GPIOTE channel output to the desired level */
    NRF_GPIOTE->TASKS_OUT[channel_number] = 1;

    /* Finally configure the channel as the caller expects.
       If OUTINIT works, the channel is configured properly.
       If it does not, the channel output inheritance sets the proper level. */
    NRF_GPIOTE->CONFIG[channel_number] = (GPIOTE_CONFIG_MODE_Task << GPIOTE_CONFIG_MODE_Pos)     |
                                         ((uint32_t)pin_number    << GPIOTE_CONFIG_PSEL_Pos)     |
                                         ((uint32_t)polarity      << GPIOTE_CONFIG_POLARITY_Pos) |
                                         ((uint32_t)initial_value << GPIOTE_CONFIG_OUTINIT_Pos);
```

```
    /* Three NOPs are required to make sure configuration is written
       before setting tasks or getting events */
    __NOP();
    __NOP();
    __NOP();
}
```

10. GPIOTE: The module cannot receive tasks or detect transitions on pad the first 3 clock periods after being enabled

Symptoms:

A task is not always detected by the module.

Conditions:

Right after enabling the module.

Consequences:

None other than the effect it will have on the application.

Workaround:

Ensure that no task is sent to the module the first 3 clock cycles after enabling.
Adding 3 NOP statements between enable and setting tasks is recommended.

11. HFCLK: Base current with HFCLK running is too high

Symptoms:

Base current is up to 400 µA higher that stated in the product specification.

Conditions:

When HFCLK clock is running.

Consequences:

1. If TIMER is the only module running, too much power is drawn from the power supply. Operation therefore cannot be guaranteed in all situations.
2. Average current consumption for the system will be higher than specified.

Workaround:

To avoid potential problems while TIMER is running use constant latency mode, see POWER.CONSTLAT task while TIMER is running. To minimize idle current, use the POWER.LOWPWR task when TIMER is not running.

12. HFCLK: Clock is paused when switching clock source for HFCLK clock

Symptoms:

When switching from 16 MHz RC to 16 MHz XO the system clock will be stopped for 8 clock cycles (0.5µs).
When switching from 16 MHz XO to 16 MHz RC the system clock will be stopped for tSTART,RC16M.

Conditions:

When 16 MHz XO is requested by triggering HFCLKSTART the system will run on 16 MHz RC until the 16 MHz XO is started. At that point the system will automatically switch to 16 MHz XO, anomaly will delay this switch.
When 16 MHz XO is no longer requested by triggering HFCLKSTOP the system will automatically switch back to 16 MHz RC, anomaly will delay this switch.

Consequences:

Timing for Serial interfaces and other modules and code that uses GPIO's will be affected during switching of HFCLK clock source.

Workaround:

Care has to be taken to avoid switching clock source when using serial interfaces to avoid timing problems.

13. HFCLK: XTALFREQ register is not functional

Symptoms:

The microcontroller is not functional with a 32 MHz crystal.

Conditions:

Always.

Consequences:

The microcontroller is not functional.

Workaround:

Write the UICR address 0x10001008 with value 0xFFFFFF00.
Note that the UICR is erased whenever you download a SoftDevice.

The UICR can be written by using the debug tools:

```
nrfjprog.exe --snr <your_jlink_debugger_serial_number> --memwr 0x10001008 --val 0xFFFFFF00
```

Or the following code can be included in the SystemInit function in system_nRF51.c file, always before the launch of the TASK_HFCLKSTART task launch:

```
if (*(uint32_t *)0x10001008 == 0xFFFFFFFF)
{
    NRF_NVMC->CONFIG = NVMC_CONFIG_WEN_Wen << NVMC_CONFIG_WEN_Pos;
    while (NRF_NVMC->READY == NVMC_READY_READY_Busy){}
    *(uint32_t *)0x10001008 = 0xFFFFFF00;
    NRF_NVMC->CONFIG = NVMC_CONFIG_WEN_Ren << NVMC_CONFIG_WEN_Pos;
    while (NRF_NVMC->READY == NVMC_READY_READY_Busy){}
    NVIC_SystemReset();
    while (true){}
}
```

14. LFCLK: Calibration does not request HFCLK

Symptoms:

- Calibration of the RC32.768 kHz clock does not finish within expected timeframe
- The frequiency of the RC32.768kHz clock is not within specification after calibration

Conditions:

If RcOsc calibration module is the only module requiring HFCLK clock.

Consequences:

DONE event does not occur until a module has requested HFCLK long enough for the calibration to finish. The resulting RC32k clock frequency will not be within specification.

Workaround:

Set the microcontroller in a state were the HFCLK is requested before the launch of the calibration of the 32.768 kHz RC oscillator.
For example:

```
NRF_POWER->TASKS_CONSTLAT = 1;
```

Remember to set the microcontroller to the desired state once the 32.768 kHz RC oscillator has been calibrated.
For example:

```
NRF_POWER->TASKS_LOWPWR = 1;
```

15. POWER: It is not possible to distinguish between Power on reset and reset from off by RESETREAS

Symptoms:

Both PowerOnReset and WakingFromOff status bits are set in RESETREAS register when entering system-off.

Conditions:

When entering system-off.

Consequences:

Impossible to distinguish between the two reset sources in question.

Workaround:

Check state of the General Purpose Retention register GPREGRET, it will keep its contents in System-off but not in Power-on reset.

16. POWER: RAMON reset value causes problems under certain conditions

Symptoms:

Program does not run.

Conditions:

The automatic variable stack is located in RAM block 1.

Consequences:

Program does not run.

Workaround:

Include the following code in the startup_nrf51.s reset routine as the first code run in the microcontroller (before the line "LDR R0, =SystemInit"):

```
LDR     R0, =NRF_POWER_RAMON_ADDRESS
LDR     R2, [R0]
MOVS    R1, #NRF_POWER_RAMON_RAM1ON_ONMODE_Msk
ORRS    R2, R2, R1
STR     R2, [R0]
```

Add as well the following lines before the reset handler procedure:

```
NRF_POWER_RAMON_ADDRESS           EQU   0x40000524  ; NRF_POWER->RAMON address
NRF_POWER_RAMON_RAM1ON_ONMODE_Msk EQU   0xF         ; RAM block 1 on in onmode bit mask
```

17. RADIO: END to START connection using PPI or short is not functional

Symptoms:

1. Radio SHORTS (END->START) is not functional.
2. Connecting the END event to the START event in the Radio using a PPI channel does not work.

Conditions:

Always applies.

Consequences:

Radio will not catch the START task and remain in READY state.

Workaround:

Do not use SHORTS or PPI to connect RADIO END to Radio START.
Connecting Radio END to Radio START has to be done in software, either by polling the END event and/or by setting up an interrupt to be trigged on the END event.

18. RADIO: RSSI module analog front end does not power down when RSSIEND event occurs

Symptoms:

Higher current consumption.

Conditions:

Always after RSSI measurement.

Consequences:

Higher current consumption.

Workaround:

Trigger RSSISTOP task to power down RSSI.

19. RADIO: RSSISTOP task clears the RSSISAMPLE register

Symptoms:

When RSSISTOP task is triggered, RSSISAMPLE register is cleared.

Conditions:

Always.

Consequences:

If RSSISTOP task is triggered before reading the value in RSSISAMPLE, the RSSI value will be lost and read as 0x00.

Workaround:

Read RSSISAMPLE before triggering STOP task.

20. RADIO: State Register is not functional

Symptoms:

Reading the STATE register in the RADIO always returns 0.

Conditions:

Always.

Consequences:

It is not possible to check the current state of the radio.

Workaround:

None.

21. RNG: Generated random value is reset when VALRDY event is cleared

Symptoms:

A random value of 0 is read from the random number generator.

Conditions:

The VALRDY (Value Ready) event is cleared before the generated value is read.

Consequences:

Algorithms based on random numbers will be broken.

Workaround:

Read the generated random number before the VALRDY event is cleared.

22. RNG: RNG does not generate a new number after the current number generated.

Symptoms:

New random values are not automatically generated.

Conditions:

A random value has already been generated.

Consequences:

- A manual step is required to generate a new random value.
- Data throughput is reduced.

Workaround:

Clear EVENTS_VALRDY event to start generating a new random value.

23. RNG: STOP task clears the VALUE register

Symptoms:

When STOP task is triggered, VALUE register is cleared.

Conditions:

Always.

Consequences:

If STOP task is triggered before reading the random value in VALUE, the random value will be lost and read as 0x00.

Workaround:

Read VALUE before triggering STOP task.

24. RNG: The STOP task cannot be assigned to a PPI channel

Symptoms:

When a PPI channel is configured to stop the Random Number generator the task never reaches the module.

Conditions:

Always.

Consequences:

The module will not be powered down if the PPI is set up to trigger the STOP task.

Workaround:

The random number generator has to be stopped by writing to the STOP task register.

25. System: Programmer/Debugger is unable to discover the Cortex-M SW device

Symptoms:

Cannot to enter debug session or download code to the flash memory.
The programmer/debugger reports JLink – Cortex-M Error: “No Cortex-M SW Device Found”.

Conditions:

One of the following:

- If the device is in system_off when you try to enter debug mode.
- The device is entering system_off within 500 µs after startup.

Consequences:

The debugger is unable to connect to the device when in system_off mode.

Workaround:

The device has to be woken up from system_off mode before it enters debug mode. Also, the program running on the device has to wait up to 500 µs after startup before it enters system_off mode.
In nRFgo Studio there is a Recover function that can erase the flash memory on a device that is continuously failing when trying to download code to the flash memory.

26. System: Manual setup is required to enable use of peripherals

Symptoms:

It is not possible to configure or use peripherals.

Conditions:

Always.

Consequences:

Peripherals are unusable.

Workaround:

The following instructions shall be executed before any peripheral can be used.

```
*(uint32_t *)0x40000504 = 0xC007FFDF;
*(uint32_t *)0x40006C18 = 0x00008000;
```

Execute them as early as possible, for example in the SystemInit function in system_nrf51.c. This operation will allow configuration and use of all peripherals.

27. System: System OFF and System ON current higher than specified

Symptoms:

Higher than specified current is drawn by the device in System OFF and System ON modes.

Conditions:

Always.

Consequences:

Measurements show an increased current consumption in the order of 0.5 µA to 2 µA due to leakage issues. The magnitude of increased current consumption for individual devices may vary more as it is dependent on manufacturing process variation.

Mode	RAM ON	Typical (measured) mode current	Specified mode current	Difference
System-OFF	0 kB	1.4 µA	0.4 µA	1.0 µA
System-OFF	8 kB	1.9 µA	0.6 µA	1.3 µA
System-OFF	16 k	2.4 µA	0.8 µA	1.6 µA
System-ON	16 kB	2.8 µA	2.3 µA	0.5 µA

Workaround:
None.

28. TEMP: Negative measured values are not represented correctly.

Symptoms:

Negative numbers are not represented correctly. Bit extension does not work properly.

Conditions:

Always

Consequences:

When a temperature below 0 degrees is measured, bit extension only happens up to bit 9.

Workaround:

When a value is measured, to correctly read the measured temperature perform the following operations, using an inline function:

```
return ((NRF_TEMP->TEMP & MASK_SIGN) != 0) ? (NRF_TEMP->TEMP |
MASK_SIGN_EXTENSION) : (NRF_TEMP->TEMP);
```

where:

```
MASK_SIGN = (0x00000200)
MASK_SIGN_EXTENSION = (0xFFFFFC00)
```

29. TEMP: STOP task clears the TEMP register.

Symptoms:

When STOP task is triggered, TEMP register is cleared.

Conditions:

Always

Consequences:

If STOP task is triggered before reading the measured temperature in TEMP register, the measurement will be lost. If TEMP is read afterwards, 0x00 will be read.

Workaround:

Read TEMP before triggering STOP task.

30. TEMP: TEMP module analog front end does not power down when DATARDY event occurs.

Symptoms:

Higher power consumption.

Conditions:

Always

Consequences:

Higher current consumption.

Workaround:

Trigger STOP task to power down analog front end and reduce power consumption.

31. TEMP: Temperature offset value has to be manually loaded to the TEMP module

Symptoms:

The temperature sensor gives wrong values.

Conditions:

Always.

Consequences:

Wrong temperature VALUE is read.

Workaround:

Register 0x4000C504 needs to be written to 0x00000000 before triggering the START task. If it is done, the temperature is measured correctly:

```
*(uint32_t *)0x4000C504 = 0x00000000;
```

32. TIMER: BITMODE is not functional for TIMER0

Symptoms:

TIMER 0 does not wrap a value 2^16-1 = 65535. 16 bit mode is not functional for TIMER0.

Conditions:

Always.

Consequences:

TIMER0 can only be used as a 24 bit timer.

Workaround:

Use a shortcut to clear Timer0.

To ensure that code is compatable with future version of the device, write the following instruction and use it before TASKS_START is triggered:

```
NRF_TIMER0->BITMODE = TIMER_BITMODE_BITMODE_24Bit << TIMER_BITMODE_BITMODE_Pos;
```

33. TIMER: One CC register is not able to generate an event for the second of two subsequent counter/ timer values.

Symptoms:

A timer event is not generated when the timer reaches the value stored in the CC register.

Conditions:

When the same CC register was used to generate an event at the desired value minus one.

Consequences:

No event is generated.

Workaround:

Use another CC register to generate the event.

34. TIMER: Timer cannot handle quick START-STOP-START tasks correctly

Symptoms:

STOP task may be ignored by the system in some corner cases.

Conditions:

A STOP task is ignored if it occurs in the same timer period (set by PRESCALER) as a START task.
E.g: Within the same PRESCALER period START has priority, not the latest arriving task.

Consequences:

The timer may continue to run and requests resources, thereby increasing current consumption.

Workaround:

None.

35. TWI: Consumes too much current when it is enabled and the STOP task is triggered.

Symptoms:

TWI consumes between 10 µA and 750 µA more current than expected when enabled.

Conditions:

TWI is enabled but has been stopped using the STOP task.

Consequences:

Increased current consumption.

Workaround:

Disable TWI module when not using it.
Note: Will require re-configuration of module once enabled again.

36. TWI: Shortcuts described in nRF51 Reference Manual are not functional

Symptoms:

The following shortcuts are not implemented:

- Short-cut between BB event and SUSPEND task
- Short-cut between BB event and STOP task

Conditions:

Always.

Consequences:

Connections have to be set up using a PPI channel.

Workaround:

See Consequences.

37. UART: After a STOPRX task the UART will not be able to finish transaction

Symptoms:

When handshake is used the "QOS" cannot be guaranteed with respect to RTS going inactive as result of triggering the STOPRX task.
When the STOPRX task is triggered, the UART will set the RTS line inactive, followed by switching off the clocks to the UART's receiver. The UART specification states that the counterpart UART transmitter can send one byte (and only one) after the RTS line has been deactivated. However, this last byte will be lost if the UART's receiver is stopped (STOPRX) before, or during reception of the last byte.

Conditions:

This may occur when the STOPRX task is used to end a transmission while flow-control is used.

Consequences:

If conditions are met the last byte received will be lost.

Workaround:

Instead of stopping the UART using STOPRX only, firmware can first disconnect the RTS line from the GPIO using PSELRTS=0xFFFFFFFF (while the OUT register for the pin is set to 1, so when the UART releases the PIN the communication partner still sees it un-asserted), then use a timer to time the period to listen for the last byte, before triggering the STOPRX task. The time required to listen for the last byte depends on the link-speed and the protocol used.

Nordic Semiconductor ASA
Tel.: +47 72 89 89 00
Fax: +47 72 89 89 89
Reg. nr./VAT no.: 966011726

38. WDT: The watchdog config option "RUN while paused by the debugger" does not work

Symptoms:

The debugger and micro-controller do not communicate. The micro-controller does not run any code.

Conditions:

The watchdog is configured to "run while halted by the debugger", and the watchdog timer expires with the debugger connected.

Consequences:

The debugger and micro-controller do not communicate. The micro-controller does not run any code.

Workaround:

Do not configure the watchdog timer to run while paused by the debugger.