



N7 SPACE

ARM Board Support Package Criticality B Qualification

Konrad Grochowski

Final Presentation Days – June 2024

N7 Space

- Company devoted to space systems software development
- Joint venture between SPACEBEL and N7 Mobile
- Key activities:
 - Critical software development for real-time embedded systems
 - LEON and ARM based systems
 - Software qualification based on ECSS standards
 - Development and applications of MBSE tools
 - Formal architecture and data modelling for flight software
 - Model checking
 - Rich experience in TASTE and Capella ecosystems
- Engineering team with ~10 years of experience in space engineering
- Team size: 25



Project objectives

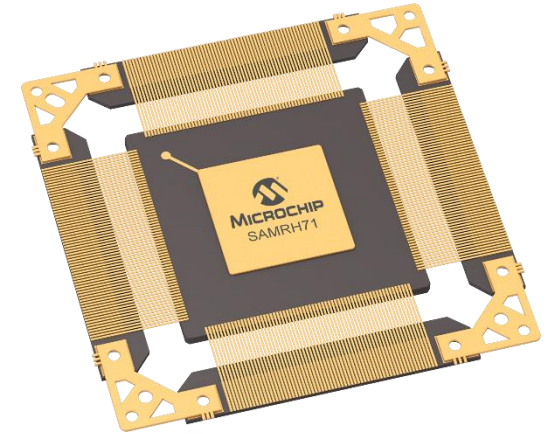
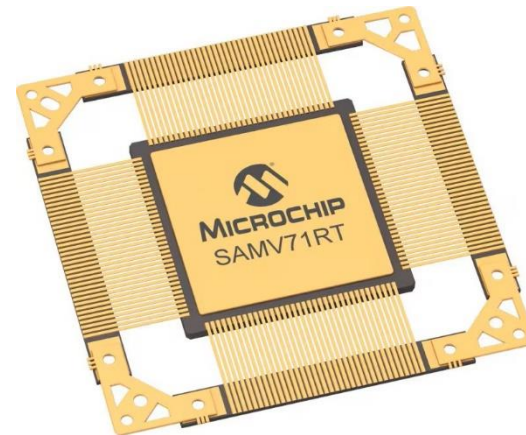
- Qualification of the **Bootloader** and **BSP** to criticality level **B** for **SAMV71Q21** MCU.
- Porting of the **Bootloader** and **BSP** components to **SAMRH71F20** MCU.
- Development of dedicated **BSP** for **RTEMS** operating system for those MCUs.

Background

- Continuation of two previous ESA activities
 - SAMV71 BSP and BSW for category C
 - SAMRH707 BSP and BSW for Microchip (SAMRH71 used as test platform)
- “Cat. C” BSP and BSW used in multiple N7S projects
 - CANopen library validation
 - MBSE projects demonstrations
- Experience from deploying BSW for PROBA3 mission

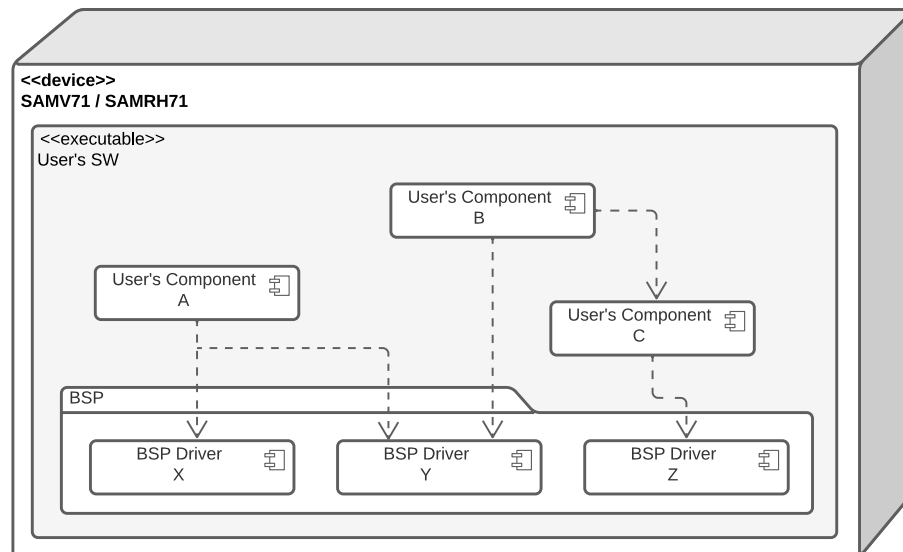
SAMRH71F20 & SAMV71Q21(RT)

- Cortex-M7 ARM microcontrollers
- Radiation Hardened / Radiation Tolerant
- Multiple peripherals
- Various external memories support
- Built-in ECC capabilities



BSP – Board Support Package

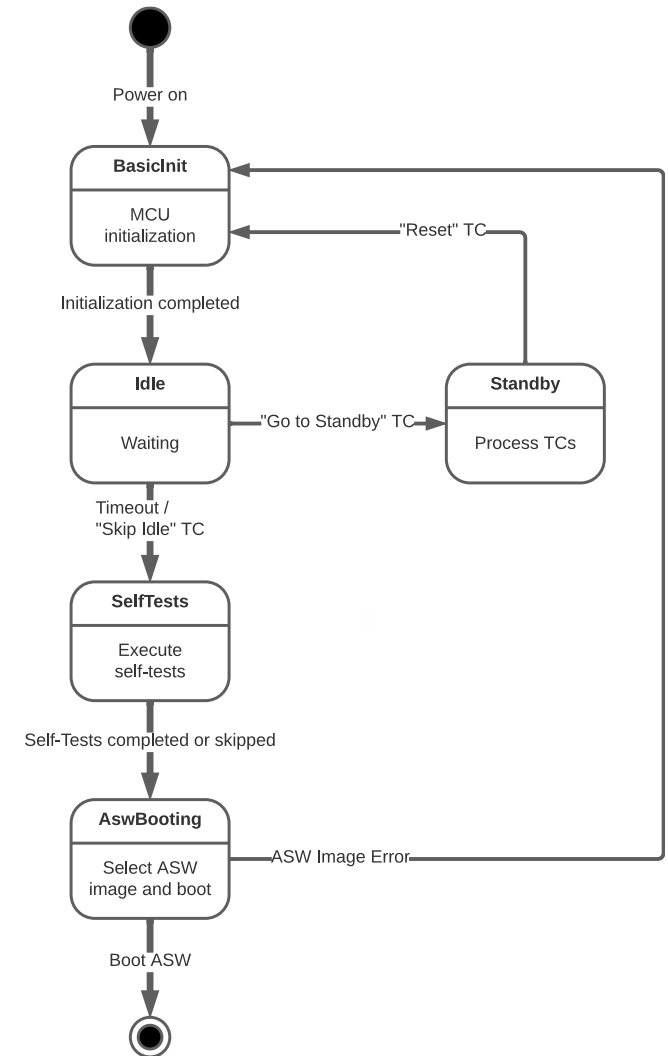
- Set of hardware drivers
- Each driver is a separate stand-alone library
- Provides C99 API to be reused by users
- Object oriented design



Device/Driver ID	Description
FPU	Floating Point Unit
GMAC	Ethernet
MCAN	Controller Area Network
MPU	Memory Protection Unit
NVIC	Nested Vectored Interrupt Controller
PIO	Parallel Input/Output Controller
PMC	Power Management Controller
PWM	Pulse Width Modulation Controller
RSTC	Reset Controller
RTC	Real-time Clock
RTT	Real-time Timer
SCB	System Control Block
SPI	Serial Peripheral Interface
SUPC	Supply Controller
SYSTICK	System timer
TIC	Timer Counter
TWIHS	Two-wire Interface
UART	Universal Asynchronous Receiver Transmitter
WDT	Watchdog Timer
XDMAC	DMA Controller
SAMV71 only	
AFEC	Analog Front-End Controller
DACC	Digital Analog Converter Controller
EEFC	Enhanced Embedded Flash Controller
ISI	Image Sensor Interface
LPOW	Low-power modes
QSPI	Quad Serial Peripheral Interface
RSWDT	Reinforced Safety Watchdog Timer
SDRAMC	SDRAM Controller
SAMRH71 only	
FLEXCOM	Flexible Serial Communication Controller
FLEXRAMECC	FlexRAM Memory and Embedded Hardened ECC Controller
HEFC	Hardened Embedded Flash Controller
HSDRAMC	Hardened SDRAM Controller
HSMC	Hardened Static Memory Controller
MATRIX	Bus Matrix
SPW	SpaceWire
TCM	Tightly Coupled Memory

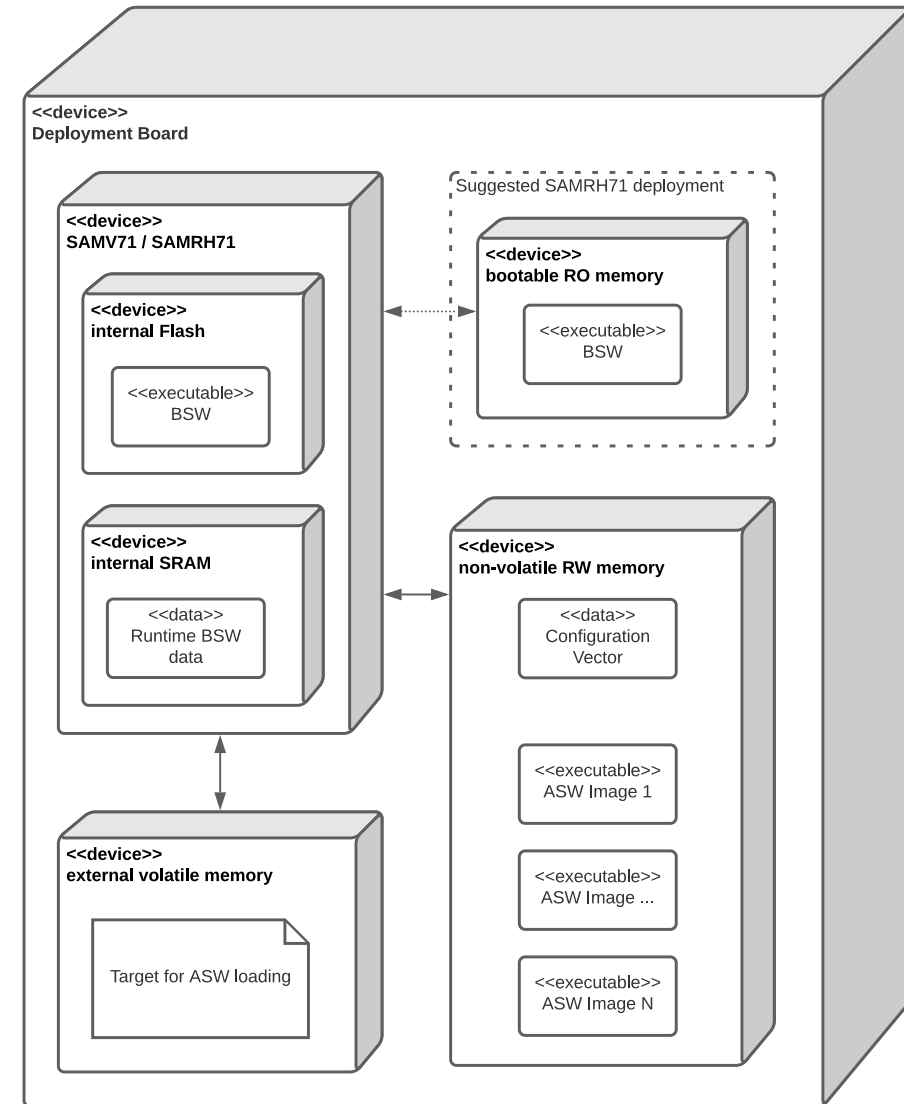
BSW – Boot Software

- Also: Bootloader, Basic Software
- First code executed on MCU reset/power-on
 - Initializes hardware
 - Performs self-tests
 - Loads and verifies ASW (application software)
 - Finishes by executing ASW code
- Allows for ASW Image patching/manipulation
 - Dedicated Standby state
 - PUS-C interface



BSW – Boot Software

- BSW – non-modifiable in-flight
 - Criticality B
 - Stored in read-only memory
 - Small (~32KiB)
- ASW – modifiable in-flight
 - Multiple copies
 - Stored in non-volatile memory
 - Executed from volatile memory (*optionally*)
 - Checksum protected

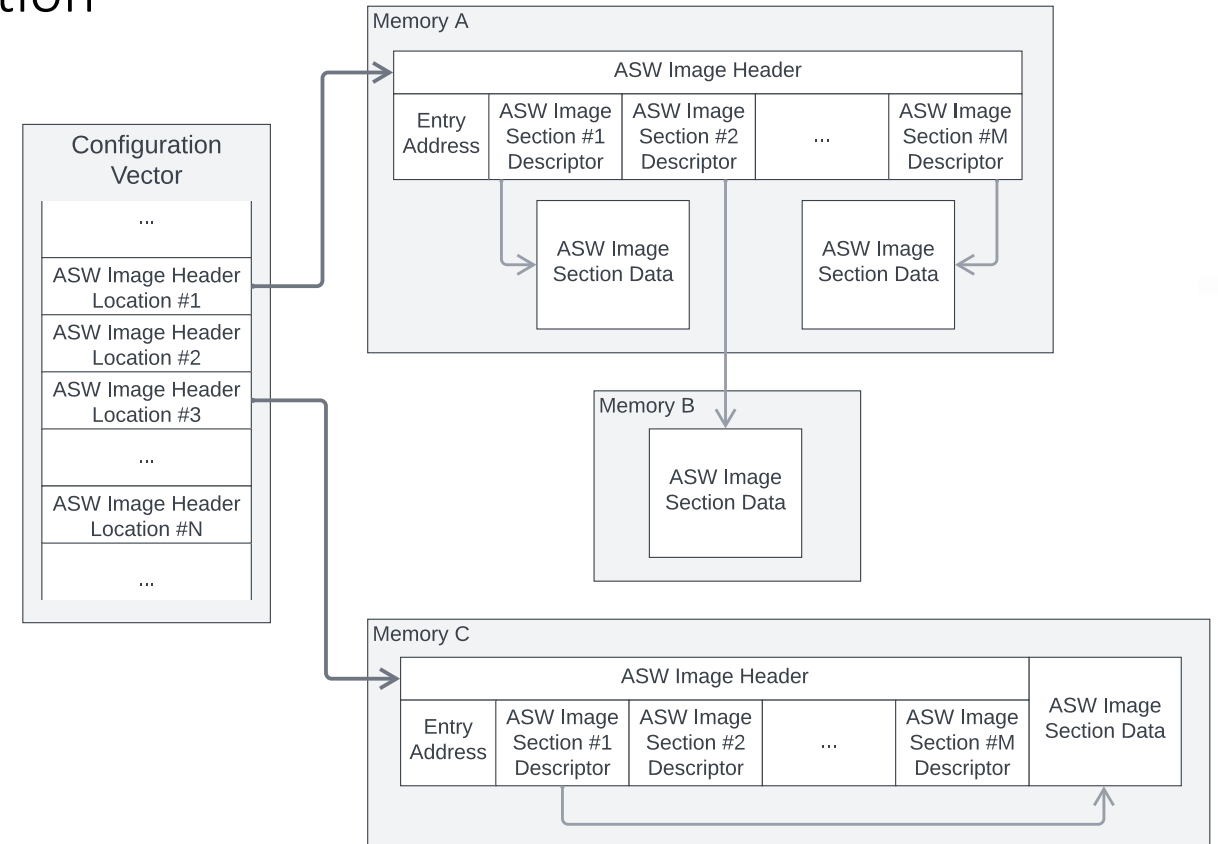


BSW – SAVOIR compliant

- SAVOIR Flight Computer Initialisation Sequence Generic Specification
 - SAVOIR-GS-002 v2.2
 - Space AVionics Open Interface Architecture
 - Set of ESA requirements and guidelines for reusable components
- Requirements used as RB for the BSW
 - Reusable BSW for ESA missions
- Defines “Nominal Sequence” for booting mission software
 - Keep BSW simple and independent (critical software)
 - Always try to boot ASW (mission objectives)

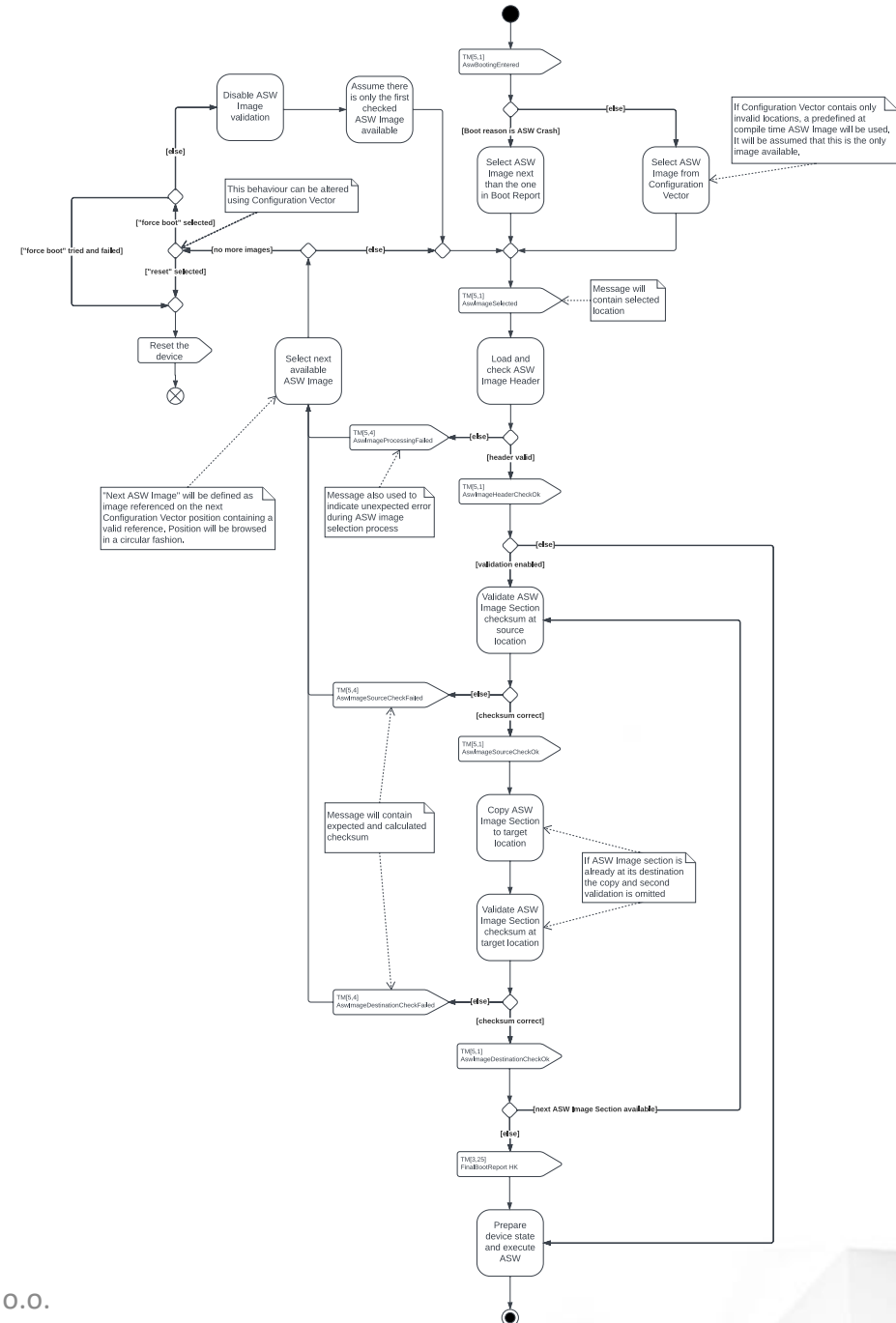
BSW – ASW Images

- Flexible ASW Image locations and selection
 - Multiple ASW Images
 - Multiple sections per image
 - Reconfigurable in-flight locations (ST[20])
 - Modifiable contents (ST[06])



BSW – ASW Images

- Autonomous selection of ASW Image
 - Based on previous execution
 - Uses checksum for validation
 - Behaviour configurable in-flight
- Includes “fast-path” settings



BSW – PUS-C services

- ST[01] request verification
- ST[03] housekeeping
- ST[05] event reporting
- ST[09] time management
- ST[17] test
- ST[20] parameter management

- MS[128] state management
- MS[129] code execution

- MBSE data model

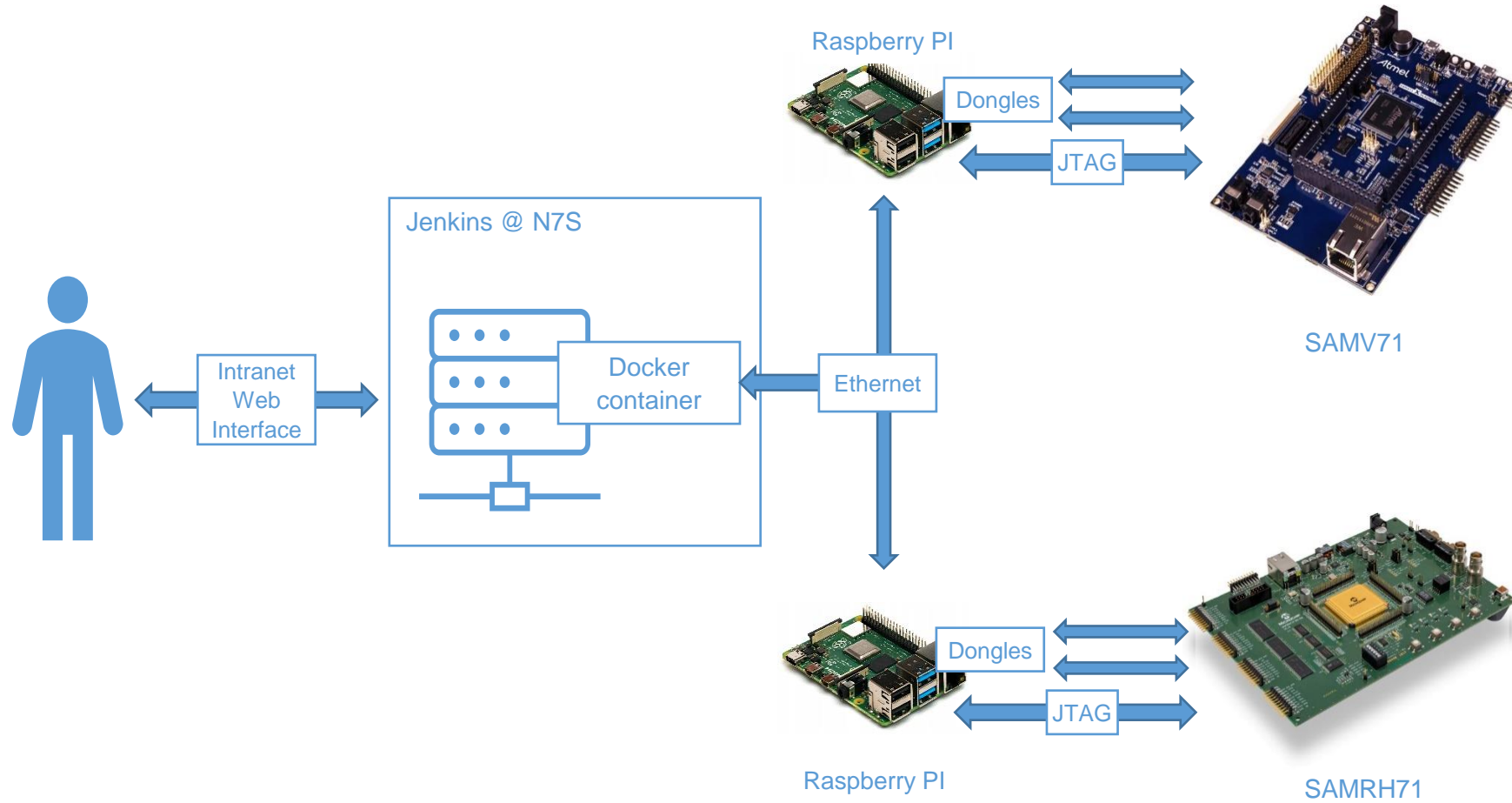
BSW – Tailorable

- Default values for ST[20] parameters
- TC/TM link (UART/SpaceWire)
- Various external memories supports
 - Non-volatile memories used as storage for ASW Images
 - Volatile memories used as target for ASW loading
- Many others
 - Time format
 - Time synchronization
 - Telemetry identifiers (sizes and values)
 - Selected self-tests
 - ...

Pre-qualification approach

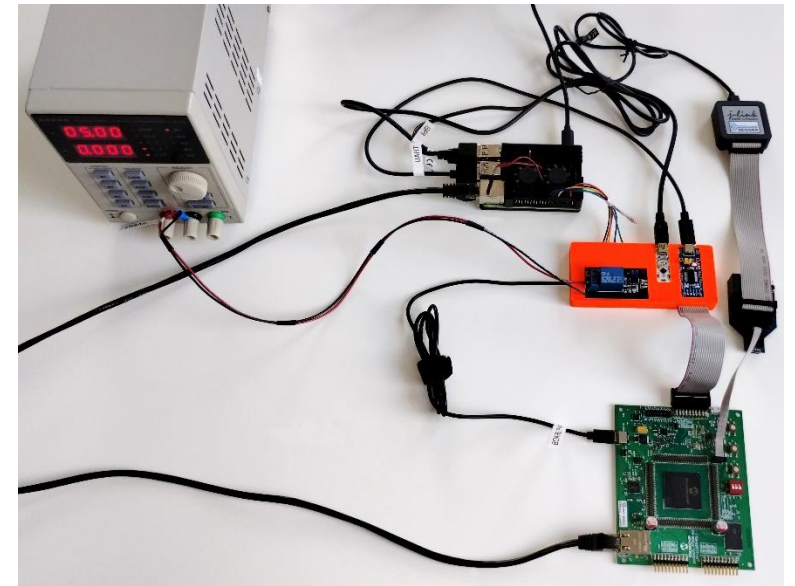
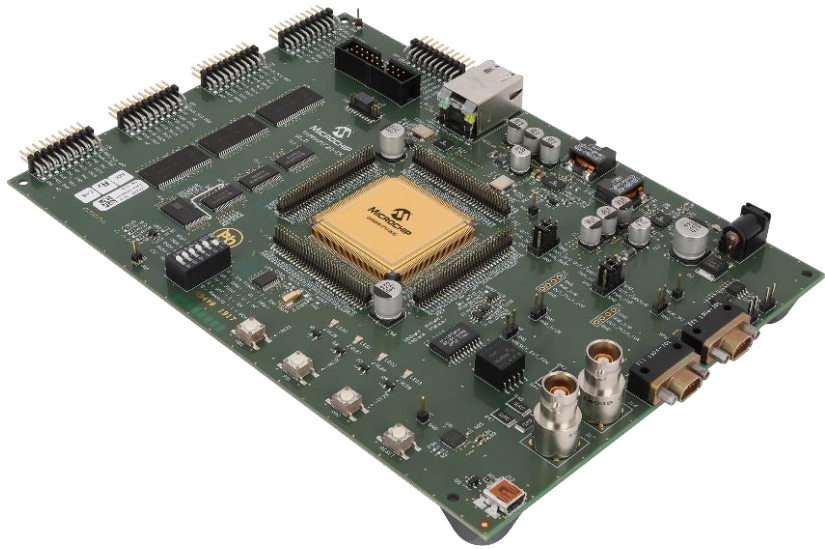
- BSW requirements based on SAVIOR
- BSP requirements extracted from MCUs datasheets
- Unit tested function by function
- Static analysis
- Dedicated integration tests for BSP
- Functional integration tests using TC/TM link for BSW
- All tests executed on the hardware
- Everything should be a “single click” process

Pre-qualification approach



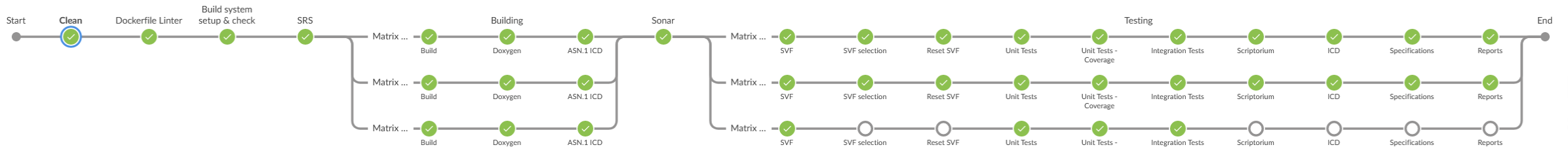
Hardware

- ATSAMV71-XULT – SAM V71 XPLAINED Ultra Evaluation Kit
- SAMRH71F20-EK – SAM RH71 Evaluation Kit
 - SAMRH71F20-TFBGA-EK supported

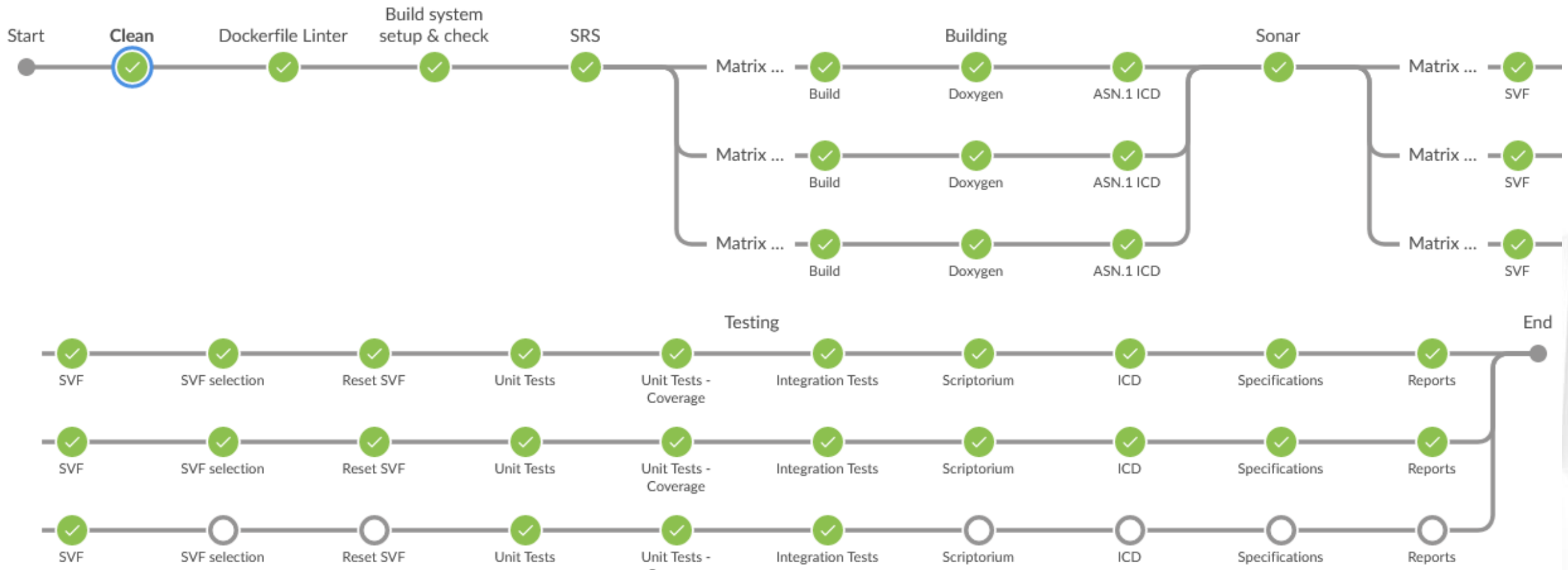


Pre-qualification approach

- Complex Continuous Integration pipeline
- Executed in regression testing mode (both platforms)
- Static analysis included as one of required steps
- All reports generated in a single place



Pre-qualification approach




Pre-qualification approach

- Reports available after each run
- Internal SonarQube

Quality Gate Status ?

✓ **Passed**



New Code Overall Code

Security

0 Open issues

0 H 0 M 0 L

Accepted issues

Current view: top level

Test: BSP C Code coverage for samrh71f20

Date: 2024-03-12 14:59:13

Legend: Rating: low: < 80 % medium: >= 80 % high: >= 100 %

	Hit	Total	Coverage
Lines:	5992	5992	100.0 %
Functions:	771	771	100.0 %
Branches:	1349	1349	100.0 %

Directory	Line Coverage	Functions	Branches
lib/n7s/bsp/Flexcom	100.0 % 41 / 41	100.0 % 6 / 6	100.0 % 18 / 18
lib/n7s/bsp/FlexramEcc	100.0 % 51 / 51	100.0 % 8 / 8	100.0 % 4 / 4
lib/n7s/bsp/Fpu	100.0 % 145 / 145	100.0 % 13 / 13	- 0 / 0
lib/n7s/bsp/Gmac	100.0 % 645 / 645	100.0 % 80 / 80	100.0 % 113 / 113
lib/n7s/bsp/Hefc	100.0 % 263 / 263	100.0 % 42 / 42	100.0 % 93 / 93
lib/n7s/bsp/Hemc	100.0 % 175 / 175	100.0 % 22 / 22	100.0 % 50 / 50
lib/n7s/bsp/Hsdramc	100.0 % 115 / 115	100.0 % 12 / 12	100.0 % 6 / 6
lib/n7s/bsp/Hsmc	100.0 % 35 / 35	100.0 % 3 / 3	100.0 % 14 / 14
lib/n7s/bsp/Matrix	100.0 % 122 / 122	100.0 % 14 / 14	100.0 % 14 / 14
lib/n7s/bsp/Mcan	100.0 % 674 / 674	100.0 % 58 / 58	100.0 % 157 / 157
lib/n7s/bsp/Mpu	100.0 % 97 / 97	100.0 % 9 / 9	100.0 % 35 / 35
lib/n7s/bsp/Nvic	100.0 % 71 / 71	100.0 % 20 / 20	- 0 / 0
lib/n7s/bsp/Pio/samrh71f20	100.0 % 147 / 147	100.0 % 18 / 18	100.0 % 54 / 54
lib/n7s/bsp/Pmc	100.0 % 450 / 450	100.0 % 43 / 43	100.0 % 142 / 142
lib/n7s/bsp/Pwm	100.0 % 308 / 308	100.0 % 27 / 27	100.0 % 43 / 43
lib/n7s/bsp/Rstc	100.0 % 34 / 34	100.0 % 7 / 7	- 0 / 0
lib/n7s/bsp/Rtc	100.0 % 274 / 274	100.0 % 24 / 24	100.0 % 66 / 66
lib/n7s/bsp/Rtt	100.0 % 32 / 32	100.0 % 6 / 6	- 0 / 0
lib/n7s/bsp/Scb	100.0 % 127 / 127	100.0 % 14 / 14	100.0 % 30 / 30
lib/n7s/bsp/Spi	100.0 % 184 / 184	100.0 % 30 / 30	100.0 % 57 / 57
lib/n7s/bsp/Spw	100.0 % 506 / 506	100.0 % 85 / 85	100.0 % 16 / 16
lib/n7s/bsp/Supc	100.0 % 23 / 23	100.0 % 4 / 4	- 0 / 0
lib/n7s/bsp/Systick	100.0 % 34 / 34	100.0 % 8 / 8	- 0 / 0
lib/n7s/bsp/Tcm	100.0 % 120 / 120	100.0 % 23 / 23	100.0 % 4 / 4
lib/n7s/bsp/Tic	100.0 % 219 / 219	100.0 % 24 / 24	100.0 % 34 / 34
lib/n7s/bsp/Twihs	100.0 % 344 / 344	100.0 % 46 / 46	100.0 % 151 / 151
lib/n7s/bsp/Uart	100.0 % 184 / 184	100.0 % 27 / 27	100.0 % 66 / 66
lib/n7s/bsp/Wdt	100.0 % 31 / 31	100.0 % 6 / 6	- 0 / 0
lib/n7s/bsp/Xdmac	100.0 % 330 / 330	100.0 % 46 / 46	100.0 % 98 / 98
resources/n7-core/lib/n7s/utils	100.0 % 211 / 211	100.0 % 46 / 46	100.0 % 84 / 84

Generated by: LCOV version 1.14

Pre-qualification approach

- Custom reports for verification of validation activities
- Forces traceability
- Provided and archived after each run

ARM Board Support Package Criticality B Qualification - BSW (SAMRH71F20) Requirements Validation

Statistics

Requirements	Count	Total	%
Validated	80	80	100%
Closed	80	80	100%

Matrix

[Toggle all](#)

Requirement	Compliance	Tests	Design reviews	Inspections	Close-out
SAVOIR.BOOTSW.BEF.05	C	[OK] - test_bswBoot.BswBootTest::test_testBswStartsInBasicInitState [OK] - test_aswBoot.AswBootTest::test_aswBoots			Closed
SAVOIR.BOOTSW.BEF.10	C	[OK] - test_bswBoot.BswBootTestWithWdt::test_bswExecutedAfterWatchdogReset [OK] - test_bswCrash.BswCrashTest::test_testBswCrash [OK] - test_bswBoot.BswBootTestWithAswReset::test_bswExecutedAfterSoftwareReset [OK] - test_bswCrashInSelfTests.BswCrashInSelfTestsTest::test_selfTestCrashIsReportedAfterBoot [OK] -	[OK] - ARMB-BSW-Design-Review-DES-04		Closed

Pre-qualification approach

- ISVV tailored out from the ECSS
 - Budget constraints
 - Internal team separation
 - SW achieved pre-qualification status – porting to custom boards require actions on user side
 - Lite ISVV performed by ESA during Comet Interceptor deployment
- Additional activity planned for 2024

Validation approach (BSP)

- Drivers verified on unit-test level
- For peripherals having observable external effects (e.g. UART)
 - dedicated applications using the drivers to perform validation tests

Validation approach (BSW)

- TC/TM link used to interface with BSW
- Various test scenarios, including Nominal Sequence, self-tests, PUS-C services etc.
- Debugger used for fault injection and memory state verification

BSP example code

```
static bool initPio(Pio* const pio, ErrorCode* const errCode)
{
    if (!Pio_init(LED_PIO_PORT, pio, errCode))
        return false;

    Pmc pmc;
    Pmc_init(&pmc, Pmc_getDeviceRegisterStartAddress());
    Pmc_enablePeripheralClk(&pmc, LED_PMC_PERIPH);

    const Pio_Pin_Config pinConfig = {
        .control = Pio_Control_Pio,
        .direction = Pio_Direction_Output,
        .pull = Pio_Pull_None,
        .isOpenDrainEnabled = false,
        .irq = Pio_Irq_EdgeFalling,
        .isIrqEnabled = false,
        .driveStrength = Pio_Current_2m,
        .isSchmittTriggerDisabled = false,
    };

    return Pio_setPinsConfig(pio, LED_PIN_MASK, &pinConfig, errCode);
}
```


BSP example code

```
static void setLedOn(Pio* const pio)
{
    Pio_setPins(pio, LED_PIN_MASK);
}
```

```
static void setLedOff(Pio* const pio)
{
    Pio_resetPins(pio, LED_PIN_MASK);
}
```

```
static void blinkLed(Pio* const pio, const uint32_t delay)
{
    setLedOn(pio);
    delay(delay);
    setLedOff(pio);
    delay(4u * delay);
}
```

BSP example code – polling

```
static bool uartPolling(Uart* const uart, ErrorCode* const errCode)
{
    const uint8_t data = 0x42u;

    if (!Uart_write(uart, data, UART_TIMEOUT, errCode))
        return false;

    while (!Uart_isTxEmpty(uart))
    {
        // polling
    }

    // operation finished
}
```

BSP example code – callbacks

```
struct UserData {
    uint8_t someData;
};

static ByteFifo* endOfTransmissionCallback(void* const data)
{
    struct UserData* user = (struct UserData*)data;
    user->someData = 42u;
    return NULL; // or next portion of data to transmit
}

static void uartHandler(void)
{
    const Uart_TxHandler handler = {
        .callback = endOfTransmissionCallback,
        .arg = &userData,
    };

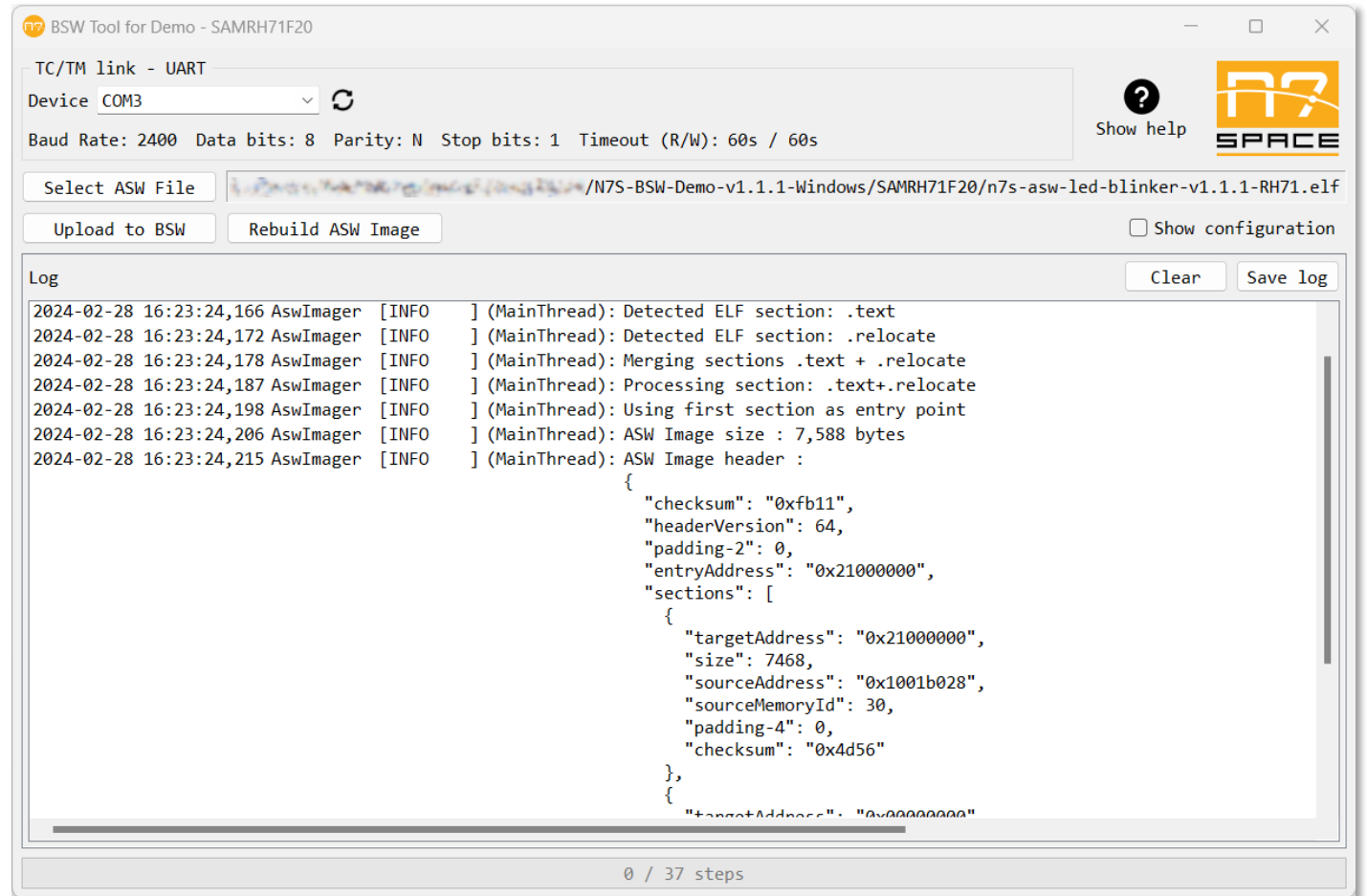
    Uart_writeAsync(&uart, createByteFifo(), handler);
}
```

RTEMS

- RTEMS – Real-Time Operating System, popular in space projects
- Adapters provided for ARM BSP to match “RTEMS BSP” interface
- SAMV71Q21 and SAMRH71F20 support as patch to RTEMS SMP Qualification Data Pack
- Example applications that use BSP directly from RTEMS applications

BSW Tool

- Dedicated application for interfacing with BSW
- Provided with BSW Demo



Deployments

- ESA missions
 - Comet Interceptor (2 payloads)
 - ARIEL (port to LEON3)
 - Envision
 - Truths
- National programme
 - Eagle Eye (scheduled launch for mid-2024)
- Additional commercial deployments
 - Including ISS payloads



Project deliverables

- ECSS compliant data pack for criticality category B
- Qualification guidelines
- Demonstration version available at: <https://bootloader.space>
 - Complete working version of the BSW for MCUs development boards
 - Limited number of ASW Images etc.

Summary

- Objectives of the project were met
- BSP and BSW become reusable components for current and future missions
- Pre-qualification status for category criticality B was achieved

Next steps

- Porting to new platforms
 - SAMRH707F18
 - LEON3
- New TC/TM links
 - Ethernet
 - CAN bus / CANopen
- New standards
 - CiA 710
- Improving ISVV
- More and more deployments...

Thank you for your attention



Konrad Grochowski
kgrochowski@n7space.com

+48 22 299 20 50
<https://n7space.com>

<https://bootloader.space>