Assignment System and OS stack(part 1)
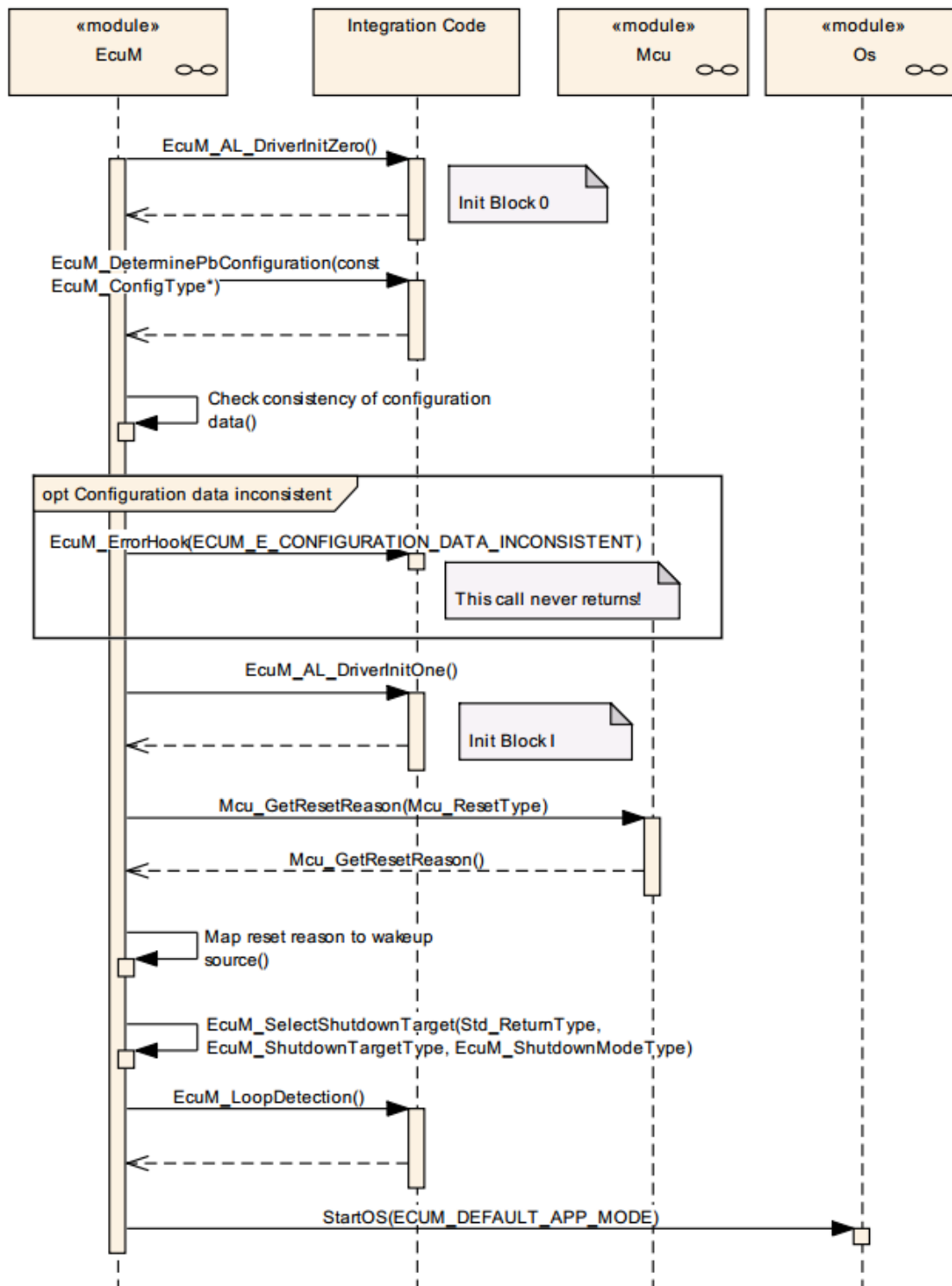
1. PreOs sequence

1.1.   CalloutbEcuM_AL_SetProgrammableInterrupts

    1.1.1.   On ECUs with programmable interrupt priorities, these priorities must be set before the OS is started.

1.2.   CalloutEcuM_AL_DriverInitZero

    1.2.1.   Init block 0 This callout may only initialize BSW modules that do not use post-build configuration parameters. The callout may not only contain driver initialization but also any kind of pre-OS, low level initialization code.(DET,DEM)

1.3.   Callout EcuM_DeterminePbConfiguration

    1.3.1.   This callout is expected to return a pointer to a fully initialized EcuM_ConfigType structure containing the post-build configuration data for the ECU Manager module and all other BSW modules.

1.4.   Check consistency of configuration data

    1.4.1.   If the check fails the EcuM_ErrorHook is called.

1.5.   Callout EcuM_AL_DriverInitOne

    1.5.1.   Init block I The callout may not only contain driver initialization but any kind of pre-OS, low level initialization code.

1.6.   Get reset reason

    1.6.1.   The reset reason is derived from a call to Mcu_GetResetReason and the mapping defined via the EcuMWakeupSource configuration containers.

1.7.   Select default shutdown target

1.8.   Callout EcuM_LoopDetection

    1.8.1.   If Loop Detection is enabled, this callout is called on every startup.

1.9.   Start OS

    1.9.1.   Start the AUTOSAR OS

The diagram shows a UML sequence diagram with the following lifelines:

- «module» EcuM
- Integration Code
- «module» Mcu
- «module» Os

Message flow:

1. EcuM → Integration Code: EcuM_AL_DriverInitZero()
   - Note: Init Block 0
   - Return to EcuM

2. EcuM → Integration Code: EcuM_DeterminePbConfiguration(const EcuM_ConfigType*)
   - Return to EcuM

3. EcuM (self-call): Check consistency of configuration data()

4. opt [Configuration data inconsistent]
   - EcuM → Integration Code: EcuM_ErrorHook(ECUM_E_CONFIGURATION_DATA_INCONSISTENT)
     - Note: This call never returns!

5. EcuM → Integration Code: EcuM_AL_DriverInitOne()
   - Note: Init Block I
   - Return to EcuM

6. EcuM → Mcu: Mcu_GetResetReason(Mcu_ResetType)
   - Return: Mcu_GetResetReason()

7. EcuM (self-call): Map reset reason to wakeup source()

8. EcuM (self-call): EcuM_SelectShutdownTarget(Std_ReturnType, EcuM_ShutdownTargetType, EcuM_ShutdownModeType)

9. EcuM → Integration Code: EcuM_LoopDetection()
   - Return to EcuM

10. EcuM → Os: StartOS(ECUM_DEFAULT_APP_MODE)

2.   Off PreOS sequence

2.1.   De-init BSW Mode Manager

2.2.   De-init BSW Scheduler

2.3.   Check for pending wakeup events

    2.3.1.   Purpose is to detect wakeup events that occurred during shutdown

2.4.   Set RESET as shutdown target, if wakeup events are pending (default reset mode of*EcuMDefaultResetModeRef* will be used)

    2.4.1.   This action shall only be carried out when pending wakeup events were detected to allow an immediate startup
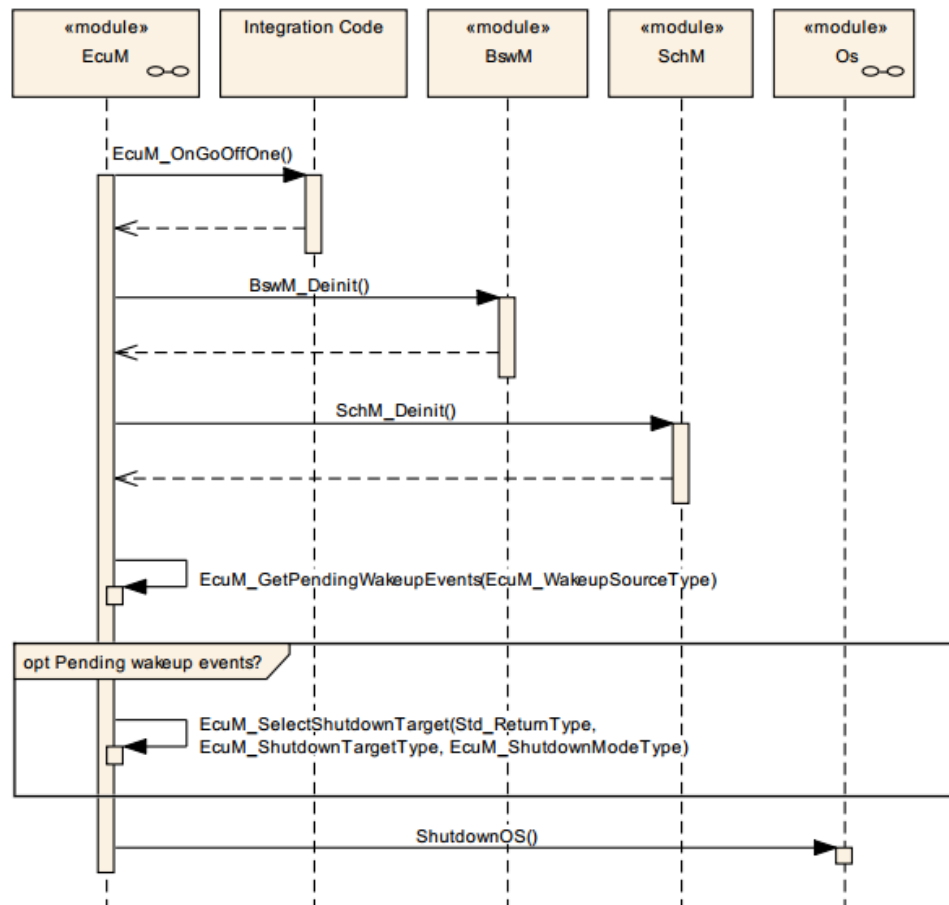
2.5.   ShutdownOS

    2.5.1.   Last operation in this OS task

Figure 8 – OffPreOS Sequence