Integration Manual --

Contents

[1 Dependencies 1](#_Toc320197364)

[2 Configuration 1](#_Toc320197365)

[2.1 Build Time Config 1](#_Toc320197366)

[2.2 Generator Config 2](#_Toc320197367)

[3 Memory Mapping 2](#_Toc320197368)

[3.1 Mapping 2](#_Toc320197369)

[3.2 Usage 2](#_Toc320197370)

# Dependencies

|  |  |
| --- | --- |
| Module | Required Feature |
| Rte | Rte\_Task\_Dispatch() hooks  VFB Trace hooks |
| Dio | Dio\_WriteChannel() |
| Port | Port\_SetPinDirection() |
| NxtrLib | DtrmnElapsedTime\_uS\_u32()  GetSystemTime\_uS\_u32() |
| Os | osdNumberOfAllTasks constant value  osGetStackUsage  oskTcbStackTop  oskTcbStackBottom |

# Configuration

## Build Time Config

|  |  |  |
| --- | --- | --- |
| Constant | Notes | SWC |
| BC\_METRICS\_TASKCPUUSEMASK | STD\_ON enables build of the task use masking functionality to excluded a user defined set of task CPU use times from the CPU use calculation. |  |
| BC\_METRICS\_STACKUSE | STD\_ON enables build of the Stack Use monitoring | Metrics |
| BC\_METRICS\_CPUUSEDIO | STD\_ON enables build of the CPU Use digital output based monitoring | Metrics |
| BC\_METRICS\_CPUUSETMR | STD\_ON enables build of the CPU Use timer based monitoring | Metrics |
| ENABLE\_CPUUSE\_DIO | If defined then CPU usage is output via a DIO where the DIO is high while the CPU is not in the background task. | Metrics |
| RTE\_VFB\_TRACE=1 | Enable Rte’s VFB trace functionality to support Rte\_Task\_Dispatch() hooks | Rte |
| Rte\_Task\_Dispatch | Enable Rte’s Rte\_Task\_Dispatch() hooks | Rte |

## Generator Config

|  |  |  |
| --- | --- | --- |
| Constant | Notes | SWC |
| Dio Channel Name: “Metrics” | Required when ENABLE\_CPUUSE\_DIO is defined | Dio |
| RTE VFB Trace Hooks | 1. Enable the option “Enable VFB Tracing” under Generation Parameters. 2. Define/Import task wait event, and wait event return hooks. Insert task start into the wait event return hook, and task end into wait the wait event hook. 3. Define/Import all desired runnable Start and Return hooks in the “Trace function“ dialog.   Caveats:   * The runnable runtime metrics methodology requires all preemption entry and exit points are instrumented with Metrics\_TaskStart/End to provide the ability to measure and subsequently account for preemption time in the preempted entity.   + Basic Tasks do not provide an Rte Exit hook, therefore the Metrics\_TaskEnd( Id ) must be integrated in the return hook of the last runnable in the Task list.   + Extended Tasks do provide start and end hooks and therefore do not require the integrator to place the Metrics\_TaskEnd( Id ) in any runnable end hook. * **Tracking all runnables is likely to impact task run times significantly. It may make more sense to enable runnable trace hooks only as necessary.** |  |
| MPU Global Read Access | GetSystemTime and Dio\_WriteChannel Metrics Channel accessed registers must be readable across all application and ISR contexts. | Os |
| MPU Global Read/Write Access | Metrics data structures must be R/W across all application and ISR contexts.  ATTENTION! The metrics data structures are very large, so ensure that adequate memory is allocated to the Global Shared region | OS |

## Integration Project Changes

1. Modify the required BSW Irq source functions to place the Metrics\_TaskStart and Metrics\_TaskEnd hooks around the ISR’s. The recommended practice is to make a copy of the delivered source file renamed to \*\_NxtrMetrics.c and conditionaly build the altered file in place of the delivered file for Metrics builds.
2. Modify SchM.c to place the Metrics\_TaskStart hook and Metrics\_TaskEnd calls appropriately at around the tasks. Note that it might be necessary to place the hooks at a point in the task that will be overwritten when the SchM.c is generated. Diligence will need to used after generation to ensure that the hooks are not inadvertently removed.
3. Modify ApplCallbacks.c to place Metrics\_TaskStart and Metrics\_TaskEnd calls appropriately around tasks. Dilligence will be required to ensure the hooks are not removed.

# Memory Mapping

## Mapping

|  |  |
| --- | --- |
| Constant | Notes |
| METRICS\_START\_SEC\_VAR\_CLEARED\_UNSPECIFIED | Writable across all applications |

\* Each …START\_SEC… constant is terminated by a …STOP\_SEC… constant as specified in the AUTOSAR Memory Mapping requirements.

## Usage

|  |  |  |
| --- | --- | --- |
| Feature | RAM | ROM |
| Software task time stamping of task execution |  |  |
| Stack usage monitoring |  |  |

Table : ARM Cortex R4 Memory UsageRevision Control Log

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item #** | **Rev #** | **Change Description** | **Date** | **Author Initials** |
| 1 |  |  |  |  |