SURESOFT

Intelligence for Safe World

PROV

타이밍 및 자원사용량 측정 도구

SURESOFT

TABLE OF CONTENTS

PROV

PROV(프로브)는 슈어소프트테크에서 자체 개발한 성능 측정 도구로서, 제어기 자원과 타이밍을 정밀 측정·가시화하여 실시간 시스템의 성능 향상과 신뢰성 검증을 지원합니다.

핵심경쟁력
(차별점/강점)

자체 개발

테스팅 전문 기술이
집약된 자체 개발 도구

공인 인증

자동차/원자력/철도/
항공분야 국제인증 획득

도구 + 서비스

고객 상황에 맞는 컨설팅,
도구, 엔지니어링 서비스 제공

합리적인 가격

고객의 환경과 예산에 맞춘
다양한 가격 정책 제공

AS-IS

1. 수작업 로그 분석 및 감으로 추정
2. 측정 편차와 누락 가능성 존재
3. 문제 원인 파악에 많은 시간 소요
4. 고가 외산 도구 의존
5. 성능평가 보고서 작성의 어려움

To-BE

1. 실시간 자동 측정과 정밀한 시각화
2. 정량적 지표 기반의 객관적 결과 확보
3. WCET 자동 탐지로 최적화 기간 단축
4. 국산화로 합리적 비용 및 맞춤형 가격 제공
5. 성능평가 보고서 자동 생성 지원

Why PROV?

테스팅 전문 기술이 집약된 자체 개발 도구

도구 확장성이 뛰어나고,
고객 친화적인 기술 지원 가능

기능안전 관련 다양한 국제 인증 보유

자동차/원자력/철도/항공 분야
국제 인증 획득

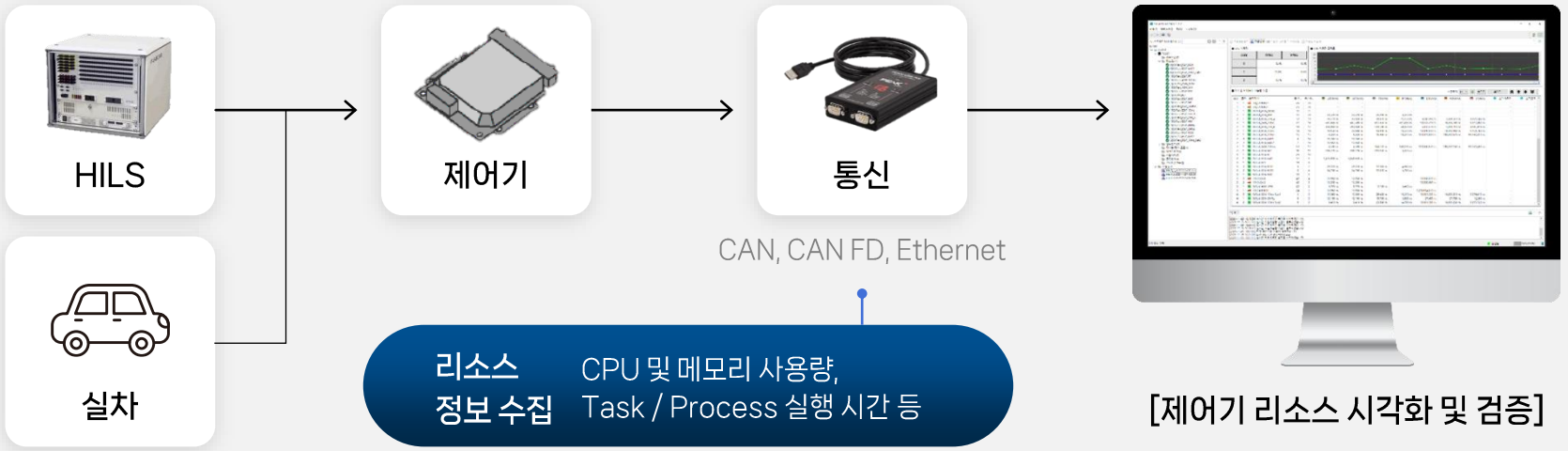
자원 사용량 검증을 위한 토탈 솔루션 제공

고객 상황에 맞는 컨설팅,
도구, 엔지니어링 서비스,
KOLAS 인증 제공

자원 사용량 검사란?

- 시스템 운영 중 제어 시스템 CPU, 메모리 등의 자원 고갈 가능성, 성능 발휘 가능성에 대한 사전 검사

자원 사용량 검사 구성



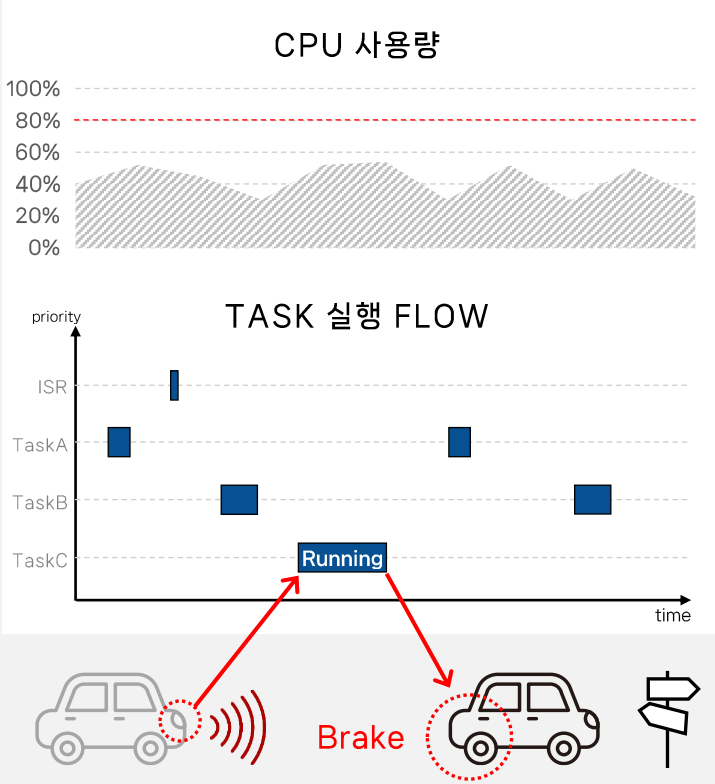
제어 시스템 자원 유형

자원 유형별 장애 요소 (1/2)

- CPU 사용량에 여유가 없으면 Task 실행이 지연되거나 실행 누락이 발생

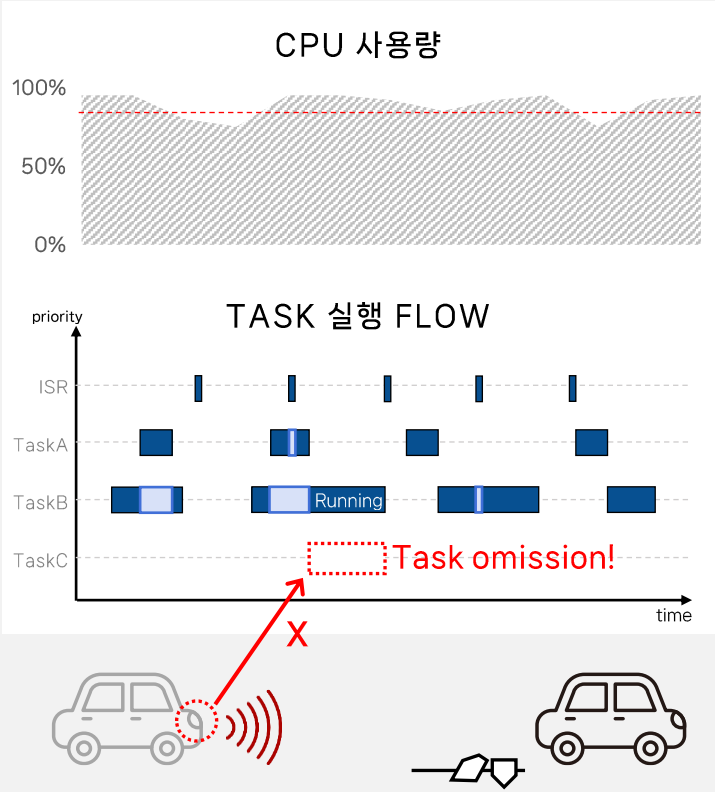
Running Preemption

CPU 사용량이 여유 있는 경우



CPU 사용량이 여유가 있는 경우
AEB(자동긴급제동장치)가 정상적으로 작동함

CPU 사용량이 많은 경우



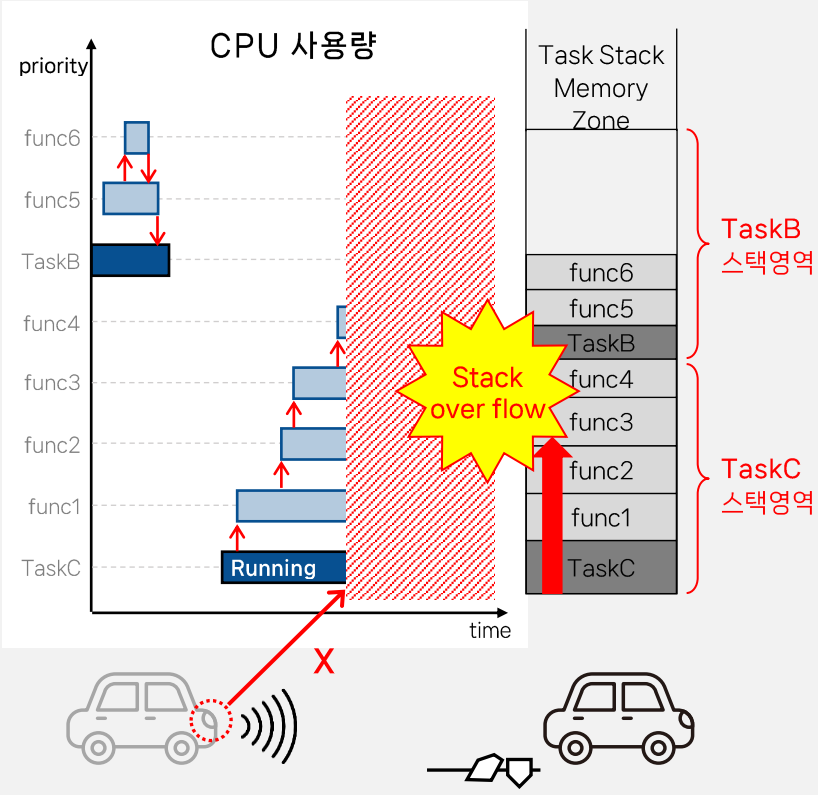
CPU 사용량이 여유가 없는 경우
해당 기능이 동작하지 못하면서 차량 제동에 실패함

자원 유형별 장애 요소 (2/2)

- 메모리가 부족하면 Stack overflow가 발생하여 시스템 전체가 멈춤
- 타이밍이 맞지 않은 경우 기능은 정상 작동하였지만 목표 성능을 충족하지 못함

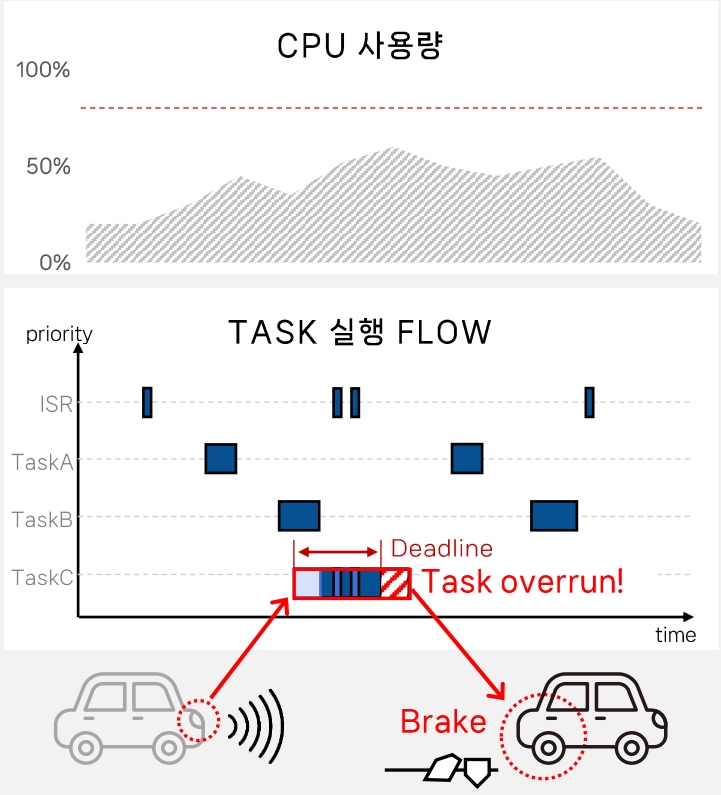
Ready | Running | Preemption | Function

메모리 사용량이 부족한 경우



메모리 오버플로우가 발생하면서
시스템이 다운되어 차량 제동에 실패함

타이밍이 맞지 않는 경우



타이밍이 맞지 않는 경우
차량 제동을 제시간에 완료하지 못하여 충돌 발생

자동차 기능 안전성 표준(ISO26262) 및 전장 SW 표준(AUTOSAR)에서 필수 수행 항목으로 명시

- ISO26262: 안전 등급이 부여된 모든 부품들은 자원 사용량 검사를 수행해야 함
- AUTOSAR: System Services 항목에 Timing Analysis 주제로 문서 정의

Methods		ASIL			
		A	B	C	D
1a	Requirements-based test [a]	++	++	++	++
1b	Interface test	++	++	++	++
1c	Fault injection test [b]	+	+	++	++
1d	Resource usage test [c, d]	++	++	++	++
1e	Back-to-back comparison test between model and code, if applicable [e]	+	+	++	++
1f	Analyses of the control or data flow	+	+	++	++
1g	Static code analysis [f]	++	++	++	++
1h	Static analyses based on abstract interpretation [g]	+	+	+	+

[a] The software requirements at the architectural level are the basis for this requirements-based test.

[b] In the context of software integration testing, fault injection test means to introduce faults into the software for the purposes described in Clause 10.4.3 and in particular to test the correctness of hardware-software interface related to safety mechanisms. This includes injection of arbitrary faults in order to test safety mechanisms (e.g. by corrupting software interfaces). Fault injection can also be used to verify freedom from interference.

[c] To ensure the fulfilment of requirements influenced by the hardware architectural design with sufficient tolerance, properties such as average and maximum processor performance, minimum or maximum execution times, storage usage (e.g. RAM for stack and heap, ROM for program and data) and the bandwidth of communication links (e.g. data buses) have to be determined.

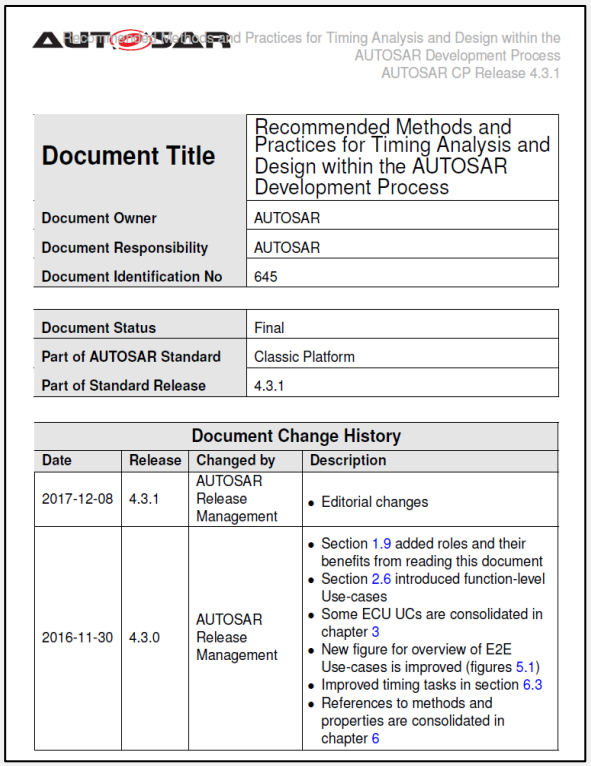
[d] Some aspects of the resource usage test can only be evaluated properly when the software integration tests are executed on the target hardware or if the emulator for the target processor supports resource usage tests.

[e] This method requires a model that can simulate the functionality of the software components. Here, the model and code are stimulated in the same way and results compared with each other.

[f] Static analyses include modelling or coding rule checks (e.g. MISRA) and analyses of resource consumption.

[g] Abstract interpretation of the source code, analyses of pointers, invalid uses of locks.

ISO26262 Part 6, Methods
for verification of software integration



AUTOSAR Recommended Methods and Practices for Timing Analysis and Design within the AUTOSAR Development Process
AUTOSAR CP Release 4.3.1

Document Title	Recommended Methods and Practices for Timing Analysis and Design within the AUTOSAR Development Process
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	645

Document Status	Final
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	4.3.1

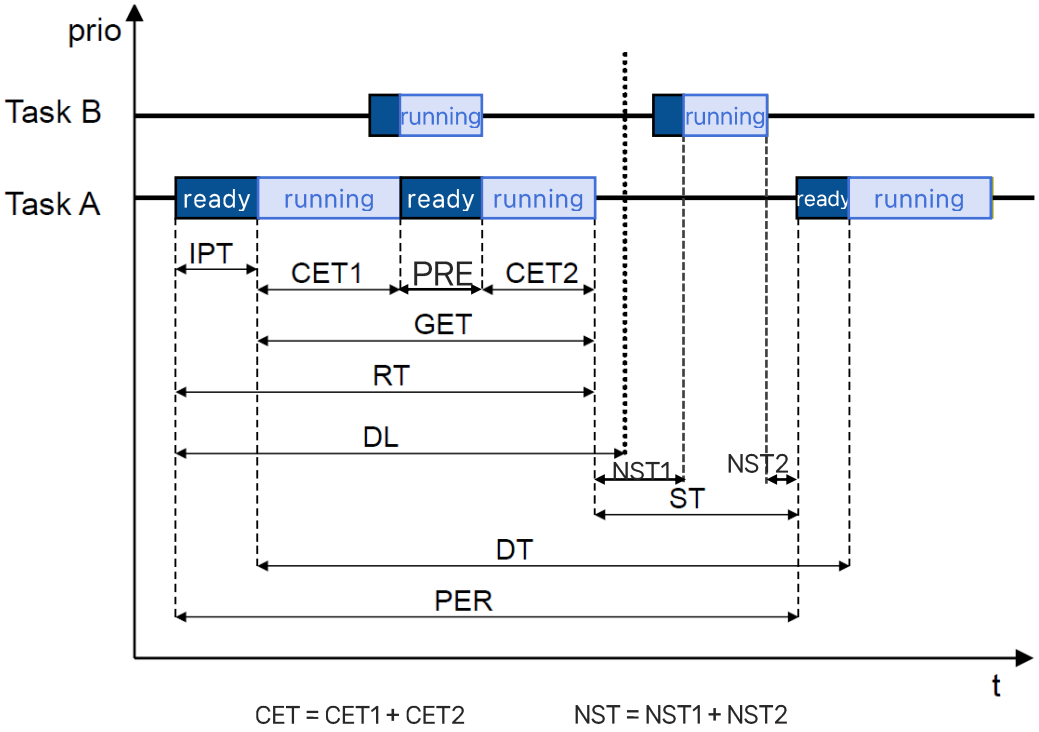
Document Change History

Date	Release	Changed by	Description
2017-12-08	4.3.1	AUTOSAR Release Management	• Editorial changes
2016-11-30	4.3.0	AUTOSAR Release Management	• Section 1.9 added roles and their benefits from reading this document • Section 2.6 introduced function-level Use-cases • Some ECU UCs are consolidated in chapter 3 • New figure for overview of E2E Use-cases is improved (figures 5.1) • Improved timing tasks in section 6.3 • References to methods and properties are consolidated in chapter 6

AUTOSAR_TR_TimingAnalysis

제어 SW Timing 측정 요소

- AUTOSAR에서 타이밍 측정 요소를 정의: IPT, CET, GET, RT, DT, PER, ST, PRE, NST



※ AUTOSAR 4.3.1 Timing Analysis, 6.2.0.1.1, p77

약어	설명
IPT	Initial pending time
CET	Core execution time
GET	Gross execution time
RT	Response time
DL	Deadline
DT	Delta time
PER	Period
ST	Slack time
PRE	Preemption time
JIT	Jitter
CPU	CPU load
NST	Net slack time

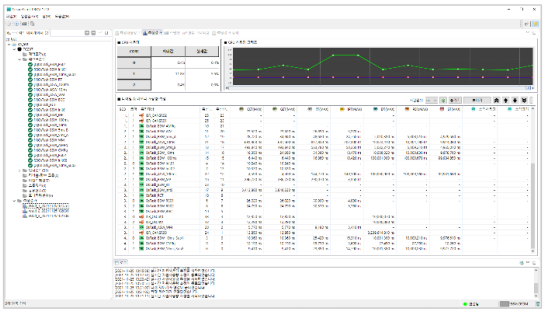
Ready Running

자원 사용량 측정을 위한 탐침 삽입, 실시간 자원 사용량 측정

- 개발 환경에 맞는 탐침 삽입 방법 지원 (Manual, Auto, 사용자 정의)
- 제어기와 실시간 통신으로 측정 데이터 수집 (CAN, CAN FD, Ethernet)

(2) 자원사용량 및 Timing 실시간 측정

자원 사용량
및 Timing
측정 도구



- 실시간 CPU/메모리 사용량
- 실시간 Task/ISR 타이밍 측정
- 스케줄링 Flow 그래프
- Timing 위반 사항 검출
- 자원 사용량 측정 보고서

CAN (FD)
Adapter

CAN/CAN FD

탐침 삽입 (사용자 정의)

탐침 Library 제공

(1) 탐침 코드를 포함한 바이너리 생성

*자동차 산업 예시

모델링 → Code Gen

개발자 → 탐침 삽입 (Manual)

플랫폼 (Mobilgene) → 탐침 삽입 (Auto)

소스코드 (ASW)

소스코드 (BSW)

PROV lib파일
(libPROV.a)

통합 빌드

툴 체인

Binary
(탐침 포함)

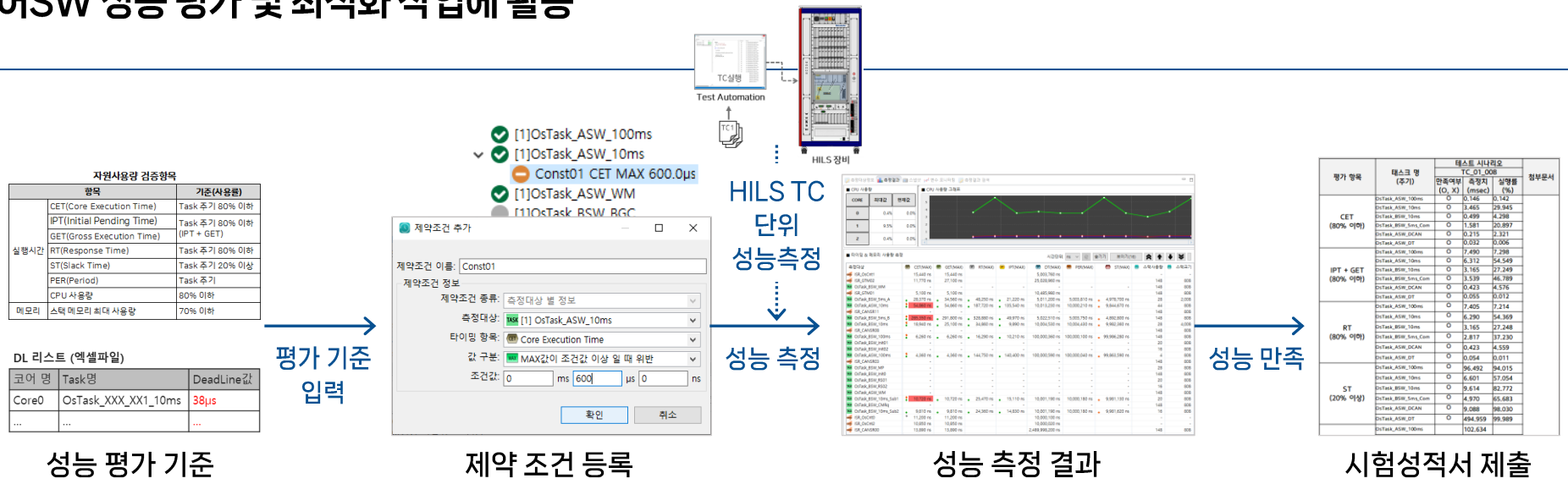
제어기

HIL

실차

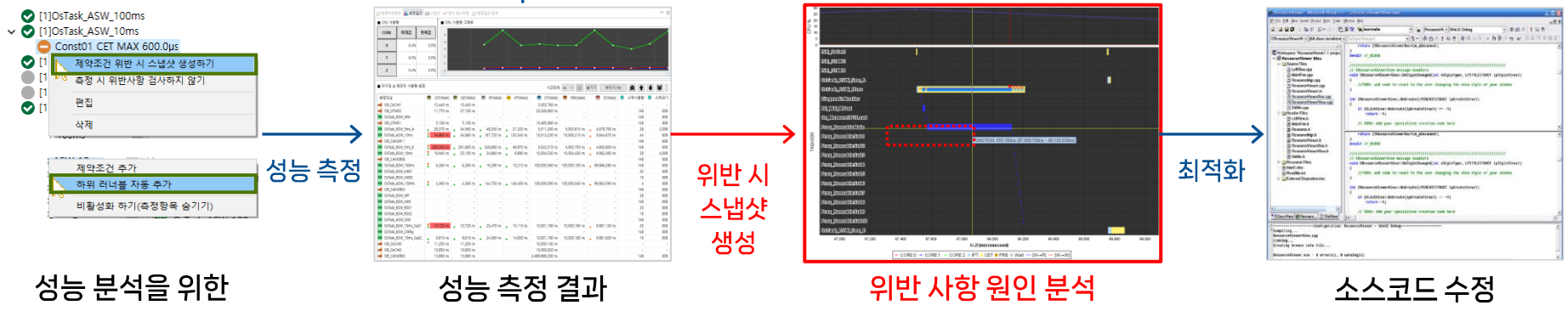
제어SW 성능 평가 및 최적화 작업에 활용

성능 평가



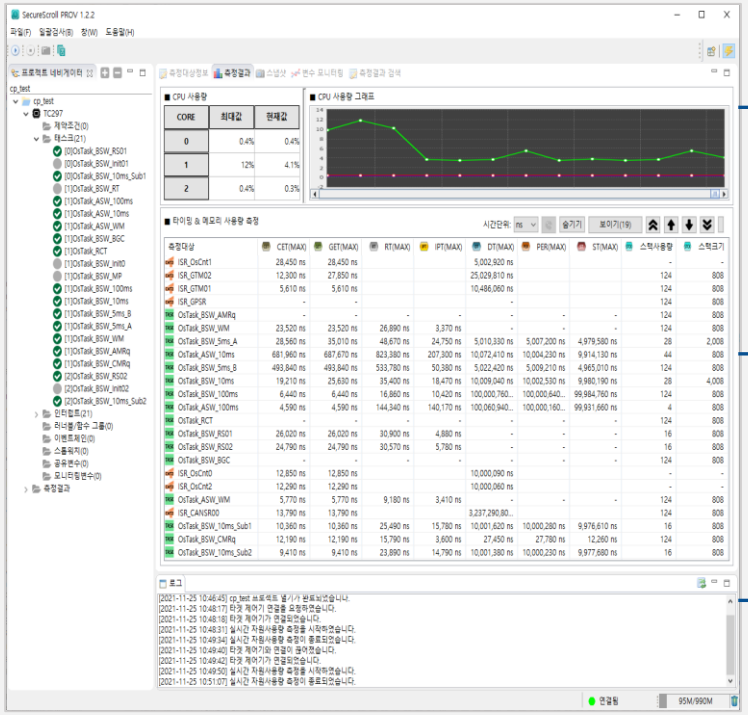
성능 불만족 시 성능 최적화 수행

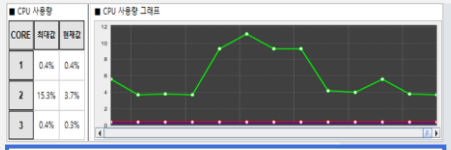
성능 최적화



실시간 측정

- 실시간 CPU/메모리 사용량 및 Task/ISR 등 각 실행 단위별 타이밍 측정 결과 확인

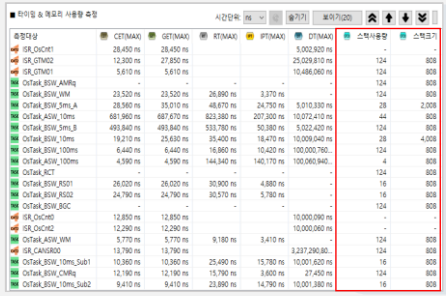




$$CPU사용량\ (U) = \frac{Sum\,of\,CET\ (To)}{Duration\ 시간\ (Te)} \times 100$$

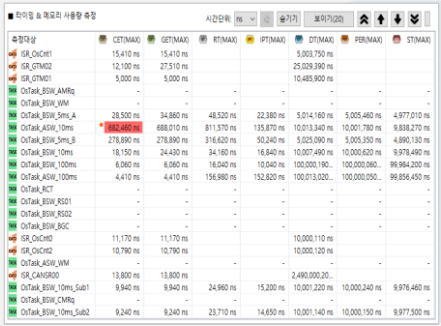
CPU 사용량 01

- Core별 CPU Load 측정 (Current, Max)



메모리 사용량 정보 02

- Task/ISR Stack 사용량
- Process 메모리 사용량

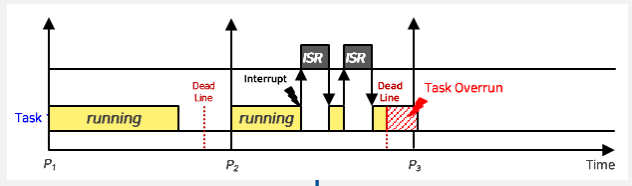


실시간 타이밍 정보 03

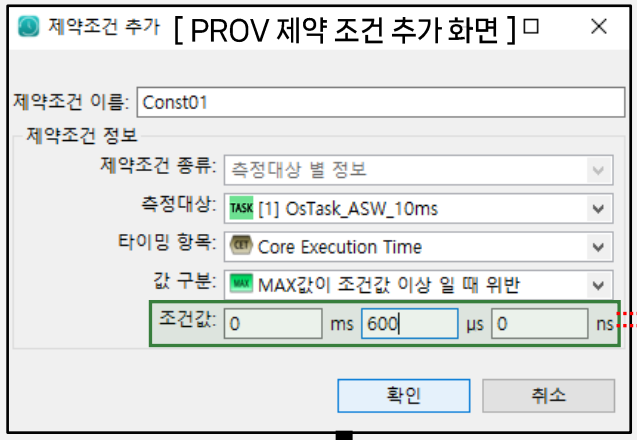
- 8가지 타이밍 요소 측정
- 타이밍 위반 사항 검출 (Task, Process, ISR, Function 단위)

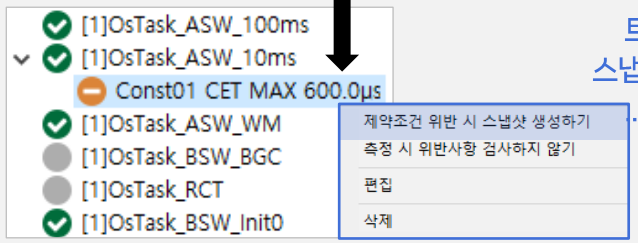
타이밍 위반 사항 검사 및 스냅샷 그래프 생성

- Deadline을 제약 조건으로 입력하면 위반 검출 및 스냅샷 자동 생성 → 스냅샷 그래프로 원인 분석



Deadline 정보 제약 조건 등록



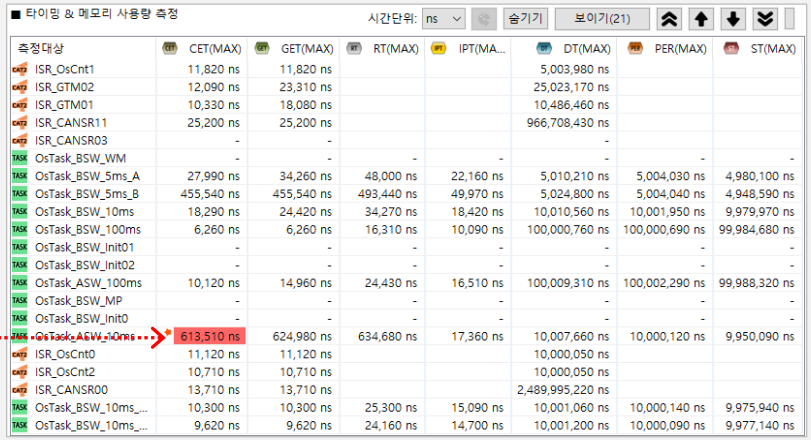


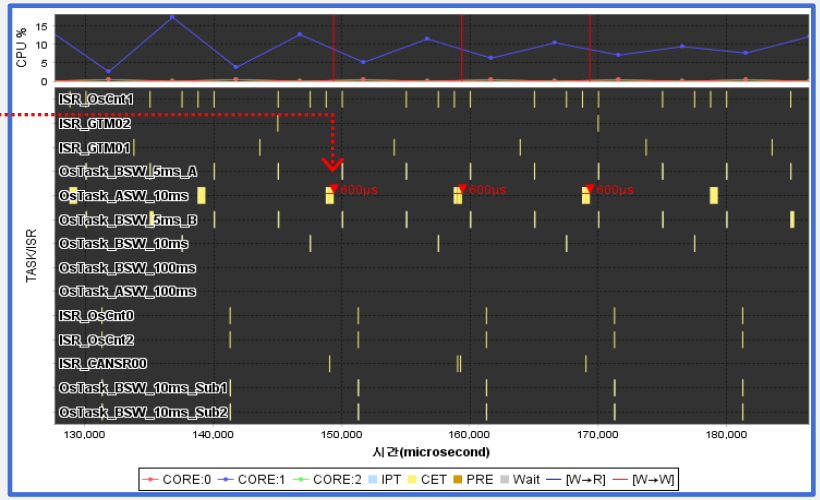
[PROV 프로젝트 네비게이터 팝업 메뉴]

타이밍 위반 검출

트리거에 의해 스냅샷 자동 생성

[PROV 타이밍 & 메모리 사용량 측정 화면]



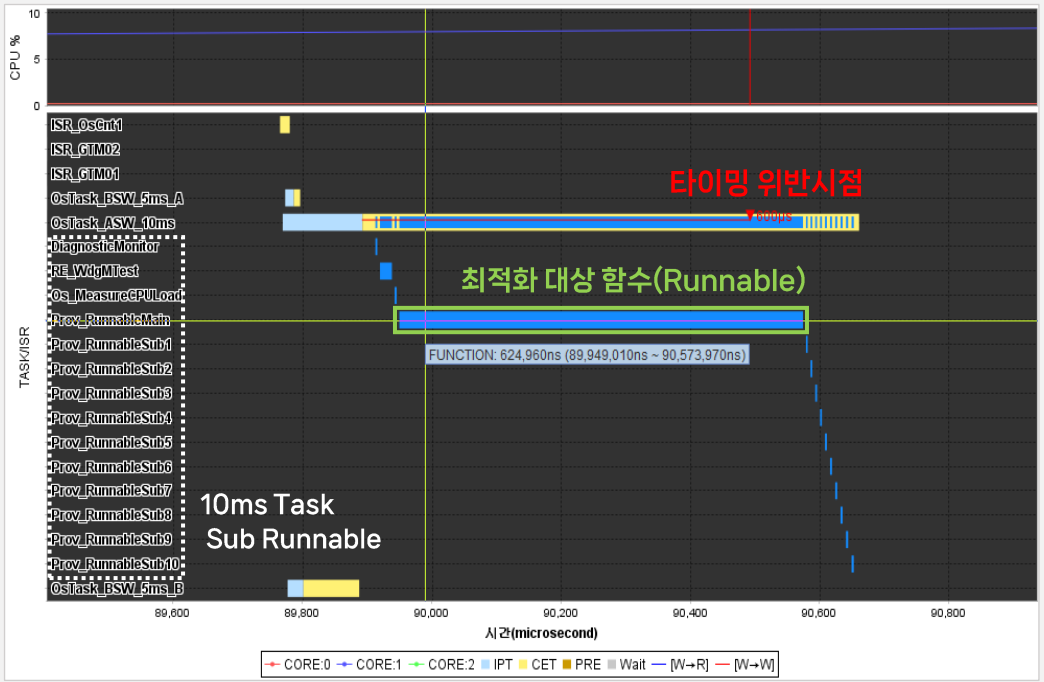


[PROV 스냅샷 그래프 화면]

스냅샷 그래프를 활용한 타이밍 실행 현황 분석

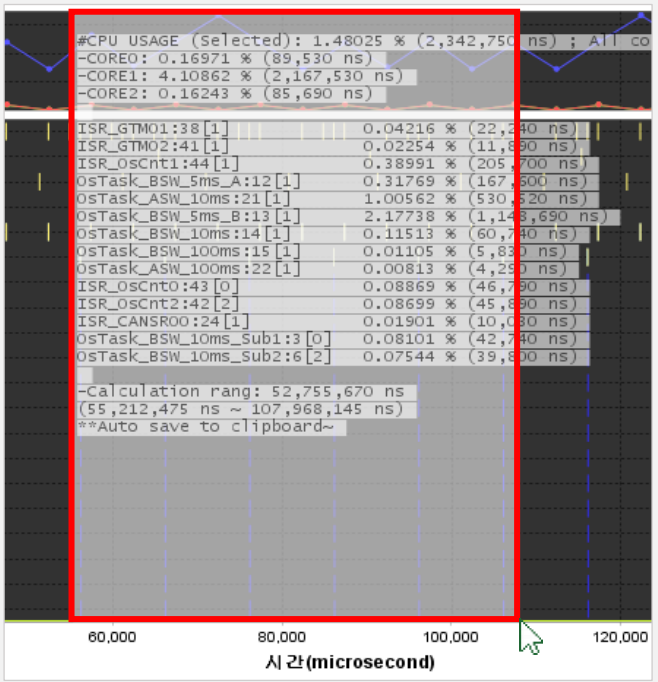
- 스냅샷 그래프: 특정 시점을 기준으로 모든 측정 항목들의 실행 Flow를 그래프로 표현
- 위반 사항이 발생한 Task의 하위 Runnable들의 실행 시간 분석, 구간별 CPU 사용량 분석

스냅샷 그래프를 활용한 타이밍 위반 원인 분석



[스냅샷 그래프 실행 시간 분석 화면]

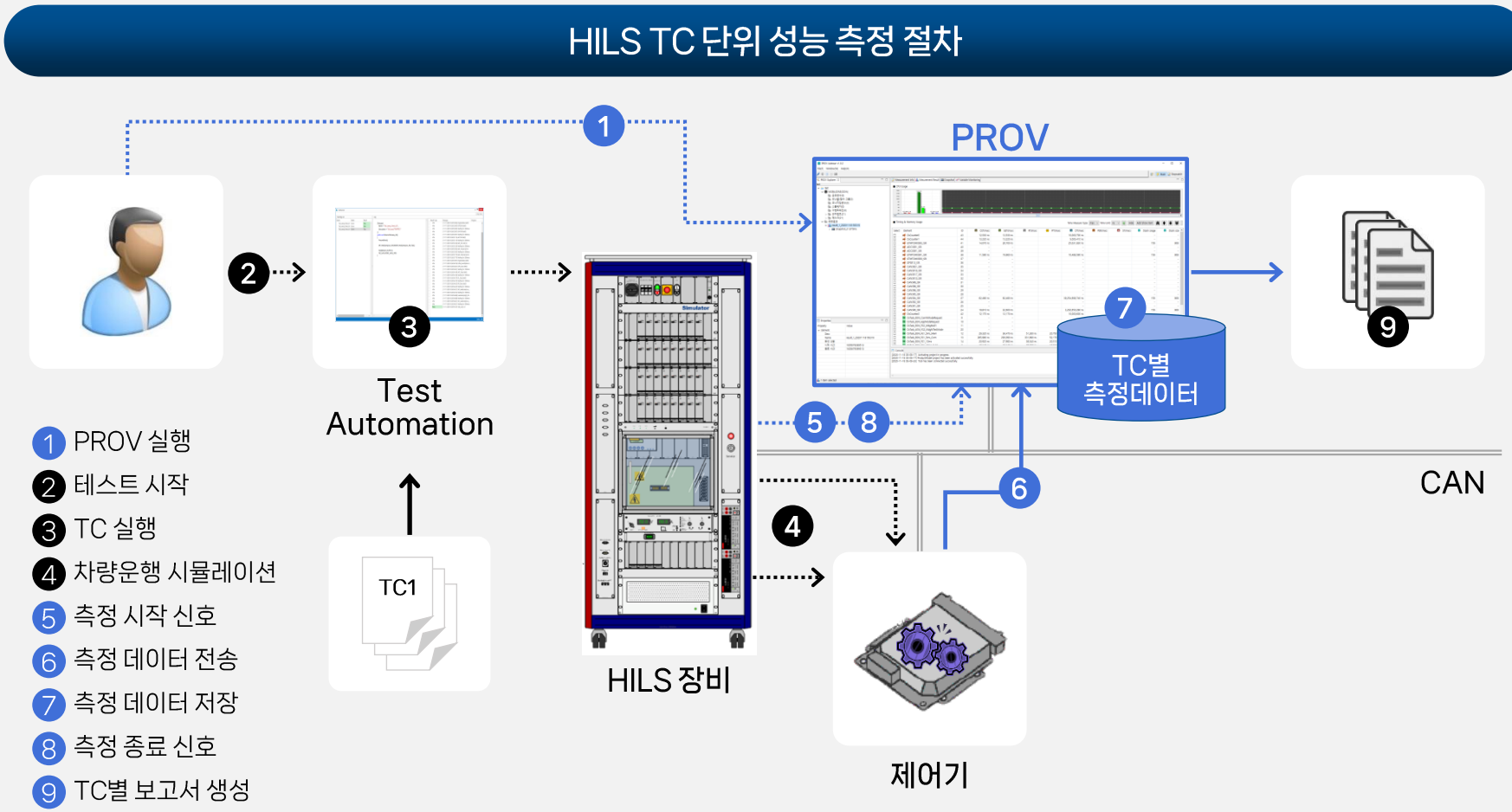
실행 구간별 CPU Load 상세 분석



[스냅샷 그래프 구간별 분석 정보 화면]

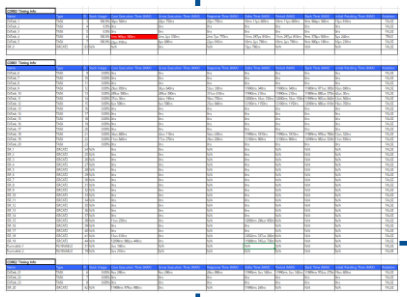
HILS 연동 및 TC기반 성능 측정 지원

- HILS 시뮬레이션 환경에서 TC단위로 성능 측정 및 보고서 생성 지원
- TC 실행 전후에 측정 시작/종료 신호를 보내면 PROV에서 자동으로 측정 수행



자동 보고서 생성 기능

- 측정된 모든 데이터들을 사용자가 필요로 하는 보고서 양식으로 가공하여 제공 가능
  (고객맞춤 보고서 양식을 위한 도구 커스터마이징 지원)



CORE0 Timing Info

Name	Type	ID	Stack Usage	Core Execution Time (MAX)	Gross Execution Time (MAX)	Response Time (MAX)	Delta Time (MAX)	Period (MAX)	Slack Time (MAX)	Initial Pending Time (MAX)	Violation
OsTask_1	TASK	3	100.0%	36µs 560ns	20µs 750ns	20µs 750ns	10ms 11µs 600ns	10ms 11µs 600ns	9ms 984µs 360ns	15µs 810ns	FALSE
OsTask_2	TASK	4	0.0%	0ns	0ns	0ns	0ns	0ns	0ns	0ns	FALSE
OsTask_3	TASK	5	0.0%	0ns	0ns	0ns	0ns	0ns	0ns	0ns	FALSE
OsTask_4	TASK	6	100.0%	1ms 969µs 580ns	2ms 2µs 530ns	2ms 7µs 770ns	11ms 297µs 810ns	11ms 297µs 810ns	9ms 378µs 920ns	5µs 240ns	TRUE
OsTask_5	TASK	7	100.0%	22µs 910ns	6µs 680ns	22µs 910ns	10ms 2µs 790ns	10ms 2µs 790ns	9ms 980µs 190ns	16µs 230ns	FALSE
ISR_0	ISRCAT2	63	N/A	N/A	0ns	N/A	13µs 790ns	N/A	N/A	N/A	FALSE

Core별 Task/Process 타이밍 정보

- CET, GET, RT, DT, PER, IPT, ST 및 Stack 사용량 표시
- 제약 조건 위반 사항에 대한 표시

Realtime Stack Usage

Core0

System Elements	Type	Stack Usage(%)	Stack Usage	Stack Size
OsTask_1	TASK	100	24584	24584
OsTask_2	TASK	0	0	0
OsTask_3	TASK	0	0	0
OsTask_4	TASK	100	24584	24584
OsTask_5	TASK	100	24584	24584

Stack 메모리 사용량

- Task/Process Stack 상세 사용량 표시

	Duration 10ms
System Core0	
CPU LOAD MAX	19.78%
CPU LOAD AVG	13.78%
CPU LOAD MIN	7.93%
System Core1	
CPU LOAD MAX	19.71%
CPU LOAD AVG	18.02%
CPU LOAD MIN	17.46%
System Core2	
CPU LOAD MAX	23.10%
CPU LOAD AVG	16.89%
CPU LOAD MIN	10.32%

Core별 CPU 사용량 정보

- Duration 기준 CPU 사용량 MAX/MIN/AVG 제공

CORE0 RAW DATA

Core Name	Type	ID	Element Name	Status	Time (ns)
CORE0	ISRCAT2	63	ISR_0	ISR_PRE	2930840
CORE0	TASK	3	OsTask_1	ACTIVATION	2937630
CORE0	ISRCAT2	63	ISR_0	ISR_POST	2942250
CORE0	TASK	3	OsTask_1	RUNNING	2948800
CORE0	TASK	3	OsTask_1	TERMINATE	2954310
CORE0	TASK	4	OsTask_2	ACTIVATION	6676700
CORE0	TASK	4	OsTask_2	RUNNING	6681510
CORE0	ISRCAT2	63	ISR_0	ISR_PRE	6686040
CORE0	TASK	5	OsTask_3	ACTIVATION	6691670
CORE0	ISRCAT2	63	ISR_0	ISR_POST	6695860
CORE0	TASK	4	OsTask_2	PREEMPTION	6700810
CORE0	TASK	5	OsTask_3	RUNNING	6704030
CORE0	TASK	5	OsTask_3	TERMINATE	6713320
CORE0	TASK	4	OsTask_2	RUNNING	6717100
CORE0	TASK	4	OsTask_2	TERMINATE	7477460
CORE0	ISRCAT2	63	ISR_0	ISR_PRE	12930690
CORE0	TASK	3	OsTask_1	ACTIVATION	12937510
CORE0	ISRCAT2	63	ISR_0	ISR_POST	12941890
CORE0	TASK	3	OsTask_1	RUNNING	12948560
CORE0	TASK	3	OsTask_1	TERMINATE	12953940
CORE0	TASK	4	OsTask_2	ACTIVATION	16679440
CORE0	TASK	4	OsTask_2	RUNNING	16684260
CORE0	ISRCAT2	63	ISR_0	ISR_PRE	16688870
CORE0	TASK	5	OsTask_3	ACTIVATION	16694590
CORE0	ISRCAT2	63	ISR_0	ISR_POST	16998650
CORE0	TASK	5	OsTask_3	PREEMPTION	16703640
CORE0	TASK	5	OsTask_3	RUNNING	16706600

System CORE0

System Elements		Stack	CET		GET	
Name	Type		Core Excution Time [MAX]	Core Excution Time [MIN]	Gross Execution Time[MAX]	Gross Execution Time[MIN]
OsTask_1 ID:3	TASK	-	9µs 300ns	9µs 070ns	9µs 300ns	9µs 070ns
OsTask_2 ID:4	TASK	0				
OsTask_1 ID:5	TASK	0				
OsTask_2 ID:6	TASK	-	1ms 954µs 010ns	768µs 800ns	1ms 979µs 590ns	794µs 780ns
OsTask_1 ID:7	TASK	-	5µs 510ns	5µs 220ns	5µs 510ns	5µs 220ns
ISR_0 ID:63	ISRCAT2		11µs 640ns	9µs 660ns	11µs 640ns	9µs 660ns

Snapshot Data 표시

- Task/Process/ISR Snapshot 타이밍 정보 MAX/MIN 표시
- Task/Process/ISR 상태 변화 Raw Data 제공

상세 환경

구분		세부 사항
호스트 PC (도구 운영 환경)	OS	- Windows 10 이상
	CPU	- x86-64 계열
	실행 환경	- JRE v1.8 이상 (설치 패키지에 포함)
타겟 (측정 대상 제어기)	OS	- AUTOSAR 플랫폼 지원 (Mobilgene, MICROSAR, RTA-CAR, ...) - OSEK 계열 (Trampoline, ROSEK, ERIKA OS, ...) - POSIX
	CPU	- Tricore/AURIX 계열 - Freescale MPC 계열 - ARM 계열
	대상 바이너리	- C, C++
	인터페이스	- CAN, CAN FD, Ethernet

권장 사양

구분	CPU	RAM	HDD
PC	Dual Core 2GHz 이상	2GB 이상	최소 500MB 이상

SURESOFT

Intelligence for Safe World

Thank You