

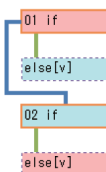
DT+ Trace

✓ Debugging(코드 디버깅)

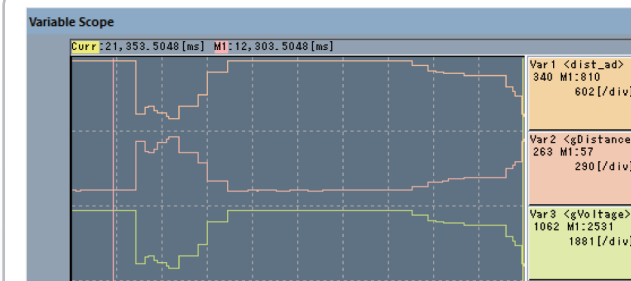
- 코드 흐름, 변수 값, 함수 호출 관계 등을 분석할 수 있는 기능을 제공합니다.
- 각 CPU, Task, Function 단위로 필터링하여 코드 흐름을 파악할 수 있습니다.

코드 구조 트리 분석

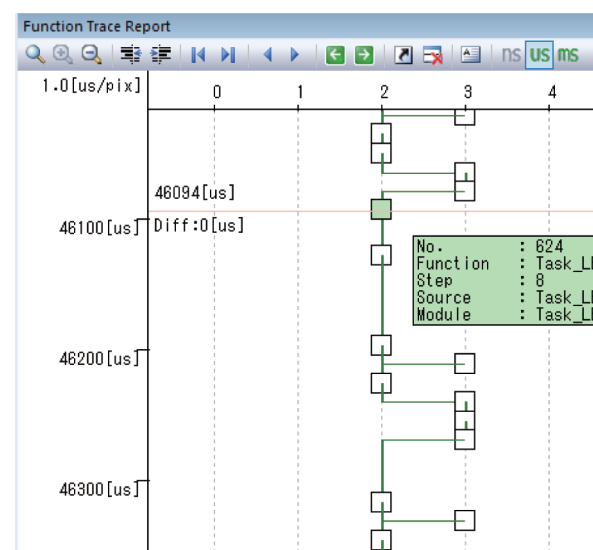
```
void test_func(int a)
{
    __DtTestPoint( __DFunc_test_func, __DtStep_0 );
    if( a != 0 )
    {
        __DtTestPoint( __DFunc_test_func, __DtStep_1 );
        printf( "0 Wn" );
    }
    a = 10;
    if( a == 10 )
    {
        __DtTestPoint( __DFunc_test_func, __DtStep_2 );
        printf( "1 Wn" );
    }
    printf( "Test func close Wn" );
    __DtTestPoint( __DFunc_test_func, __DtStep_3 );
}
```



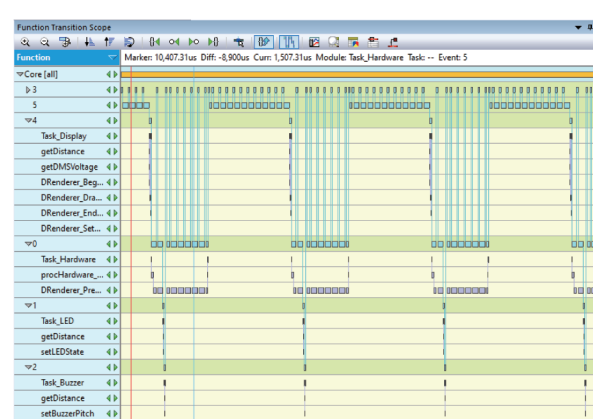
변수 값 모니터링



Function Trace Report



Function Transition Scope



DT+ FS

신뢰성이 보장된 솔루션

DT+Trace의 상위 버전 DT+FS는

기능 안전 도구로 인증 받았습니다.

<ISO 26262>

<IEC 61508>



DT+ Technical Support & Training Course

제품 사용 시 기술 지원이 필요한 경우,
방문/원격/메일/전화를 통해 적극 지원해 드립니다.

DT+Trace 커버리지 교육

- DT+ 소개
- 커버리지 분석
- 커버리지 레포트 생성
- 오버헤드를 최소화한 원비트트레이스

Make Different & Smart



(주)MD스테크 (구.한컴MDS)

13493 경기도 성남시 분당구 대왕판교로 644번길 49, DTCE타워 9층
www.mdstech.co.kr

T. 031-627-3088 E. DT@mdstech.co.kr

Coverage, Profiling & Performance, Debugging

DT+Trace





DT+Trace는 CPU, OS에 상관없이 다양한 개발 환경에서 커버리지 측정, 코드 디버깅, 성능 측정이 가능한 통합 분석 솔루션입니다.

✓ 제품 사양

- Memory, Ethernet, JTAG/GPIO 등 다양한 인터페이스 지원
- C/C++/C#/JAVA/Python 지원

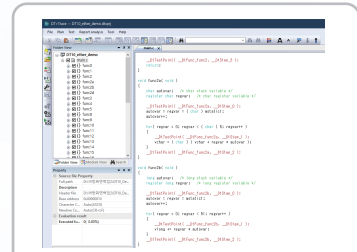
✓ 주요 기능

- > Coverage(코드 실행률)
- Statement, Branch, MC/DC 및 Call coverage 결과 측정
- > Profiling & Performance
- 함수 별 수행 시간 및 주기 측정을 통한 시스템 부하 위치 파악
- > Debugging
- 장시간 소프트웨어 Trace를 통해 간헐적으로 생기는 버그 디버깅

✓ 동작 및 분석 원리

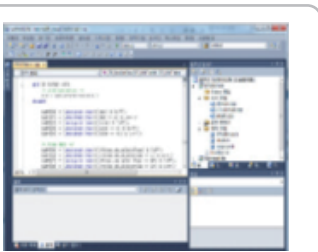
- 소스코드 내 함수의 입/출구, 분기 등 필요한 곳에 자동으로 Test Point 삽입
- 기존 개발 환경에서 컴파일 진행
- 타깃 실행 후 결과 값 수집

Test Point(Macro) 입력



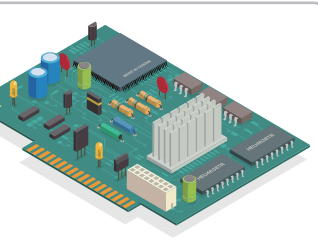
소스코드에 자동으로 삽입

기존 개발환경 사용



드라이버 추가 후 빌드

타깃의 움직임을 Trace



프로그램 수행 결과 수집



ASYN / GPIO(4/2/1 pin)

Data writing / Ethernet / JTAG/ UART / CAN



USB 3.0



✓ Coverage 분석(코드 실행률 측정)

- 소프트웨어의 품질 향상을 위해 Statement, Branch, MC/DC 및 Call coverage 분석을 지원합니다.

DT+ Trace

Statement / Branch Coverage

Coverage Report : Repo_221213_103628										
Function	Source	TP setting number	Normal	Effective TP	Execution(...)	Statement	Statement(Effective)	Number of Bra...	Passed Bra...	Branch(Effective)
func0	main.c	1	1	1	1	100.00%	100.00%	0	0	-
func1	main.c	2	2	2	2	100.00%	100.00%	0	0	-
func2	main.c	4	4	4	4	75.00%	75.00%	3	2	66.67%
func2a	main.c	3	3	3	3	100.00%	100.00%	1	1	100.00%
func2b	main.c	3	3	3	3	100.00%	100.00%	1	1	100.00%
func2d	main.c	3	3	3	3	100.00%	100.00%	1	1	100.00%
func3	main.c	1	1	1	1	100.00%	100.00%	0	0	-
func4	main.c	2	2	2	2	100.00%	100.00%	0	0	-
func5	main.c	2	2	2	2	100.00%	100.00%	0	0	-
func8	main.c	2	2	2	2	100.00%	100.00%	0	0	-
func9	main.c	4	4	4	4	100.00%	100.00%	2	2	100.00%
func10	main.c	35	35	35	35	100.00%	100.00%	33	33	100.00%
func11	main.c	11	11	11	11	100.00%	100.00%	7	7	100.00%
func12	main.c	2	2	2	0	0.00%	0.00%	0	0	-
func13	main.c	3	3	3	3	100.00%	100.00%	2	2	100.00%
func14	main.c	2	2	2	2	100.00%	100.00%	0	0	-
func15	main.c	2	2	2	2	100.00%	100.00%	0	0	-
func16	main.c	2	2	2	2	100.00%	100.00%	0	0	-

DT+ Trace

Function Coverage

Function Coverage Report : Repo_201225_141003			
Source	Function	Execute...	Coverage
DisplayRenderer.c	7	5	71.43%
Task_Buzzer.c	1	1	100.00%
Task_Display.c	1	1	100.00%
Task_Hardware.c	11	8	72.73%
Task_LED.c	1	1	100.00%
Task_RGB-LED.c	1	1	100.00%
tasks.c	2	0	0.00%

DT+ FS

MC/DC Coverage

A && B && (C && D)						
No.	A	B	C	D	Result	
1	T	T	T	T	True	
2	T	T	T	F	False	
3	T	T	F	x	False	
4	T	F	x	x	False	
5	F	x	x	x	False	

A : gBuzzerMasterOn
B : gBuzzerEnable
C : gLED_ALLOW_Flag
D : gLED_ON_Num > 0

DT+ FS

Function Call Coverage

Task_Buzzer	Task_Buzzer.c	5	4	80.00%
procHardware_LED_AD	Task_Hardware.c	26	25	96.15%
DRenderer_Init	DisplayRenderec.c	4	4	100.00%
DRenderer_SetDisplayString	DisplayRenderec.c	3	3	100.00%
DRenderer_BeginScene	DisplayRenderec.c	2	2	100.00%
DRenderer_EndScene	DisplayRenderec.c	1	1	100.00%
DRenderer_DrawDeviceFor...	DisplayRenderec.c	1	1	100.00%
DRenderer_Present	DisplayRenderec.c	5	5	100.00%
Task_Display	Task_Display.c	18	18	100.00%
initHardware	Task_Hardware.c	3	3	100.00%
Task_Hardware	Task_Hardware.c	4	4	100.00%
initHardware_LED_AD	Task_Hardware.c	8	8	100.00%
initHardware_PWM_DISP	Task_Hardware.c	26	26	100.00%
Task_LED	Task_LED.c	5	5	100.00%

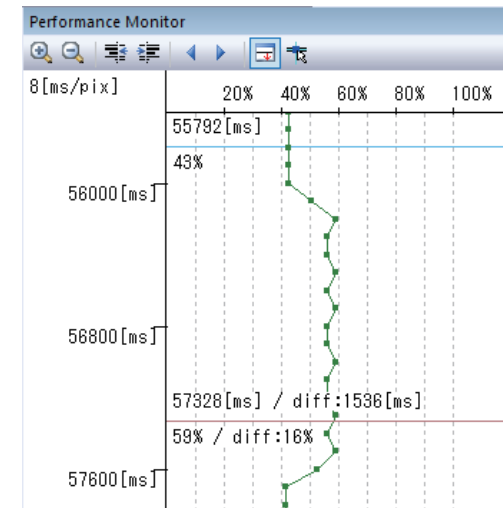
setBuzzerPitch	Task_Hardware.c	2	1	50.00%
DRenderer_Init	DisplayRenderec.c	1	1	100.00%
DRenderer_SetDisplayString	DisplayRenderec.c	1	1	100.00%
DRenderer_BeginScene	DisplayRenderec.c	2	2	100.00%
DRenderer_EndScene	DisplayRenderec.c	2	2	100.00%
DRenderer_DrawDeviceFor...	DisplayRenderec.c	6	6	100.00%
DRenderer_Present	DisplayRenderec.c	1	1	100.00%
getDistance	Task_Hardware.c	3	3	100.00%
getDMSVoltage	Task_Hardware.c	1	1	100.00%
setLEDState	Task_Hardware.c	2	2	100.00%
getColorVolume	Task_Hardware.c	1	1	100.00%
setRGBLEDColor	Task_Hardware.c	2	2	100.00%



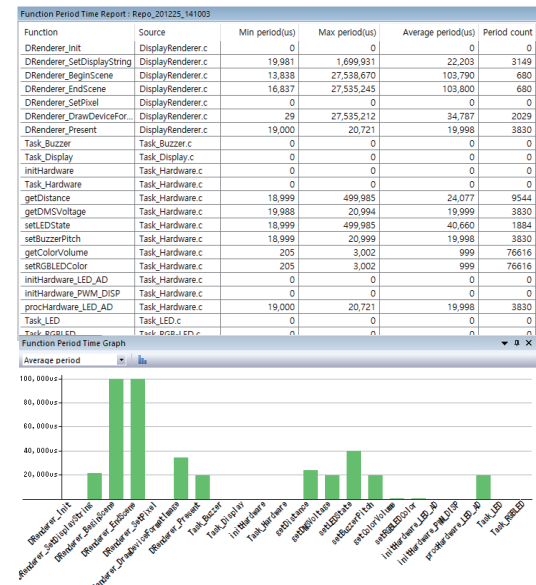
✓ Profiling & Performance(성능측정)

- 시스템의 Core, Process, Thread 동작 흐름을 시각화하여 시스템의 부하요인 파악이 용이합니다.
- OS를 사용하는 경우 CPU 점유율을 실시간으로 분석하고 시각화 할 수 있습니다.

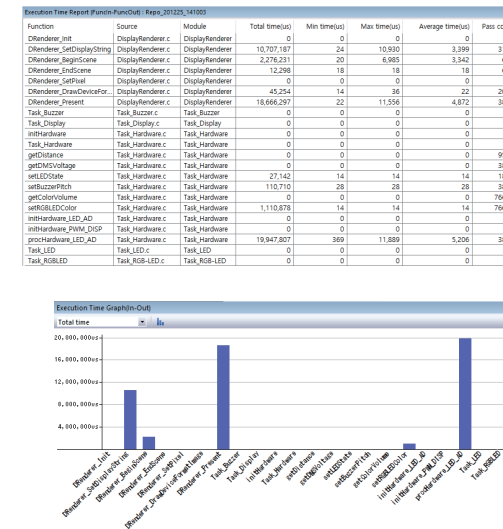
Performance Monitor



Function Period Time Report



Execution Time Report



Process Occupational Ratio Scope

