

SOLID IDEのコードカバレッジ機能 を使ってみよう!

2018.06.11 京都マイクロコンピュータ

コードカバレッジの機能



1

コードカバレッジの機能

・ソースコード中の各ベーシックブロック単位ごとに、実行された回数を記録、表示する機能です。

コードカバレッジ 中 × scov.c 音			
(♥) ファイル名	関数カバレッジ	行カバレッジ	範囲カバレッジ
C:¥DriveD¥CURRENT_WORK¥apps¥DB51903PF-demo-AXELL-sample¥ag903_sample¥openvg¥scissor.c	0% 0/4	0% 0/80	0% 0/8
C:¥DriveD¥CURRENT_WORK¥apps¥DB51903PF-demo-AXELL-sample¥ag903_sample¥openvg¥stroke.c	0% 0/4	0% 0/65	0% 0/7
C: ¥DriveD¥CURRENT_WORK¥apps¥DB51903PF-demo-AXELL-sample¥ag903_sample¥osp¥osp_sample.c	0% 0/15	0% 0/625	0% 0/554
C: ¥DriveD¥CURRENT_WORK¥apps¥DB51903PF-demo-AXELL-sample¥ag903_sample¥spiflash¥spi_sample.c	0% 0/1	0% 0/51	0% 0/51
C:¥DriveD¥CURRENT_WORK¥apps¥DB51903PF-demo-AXELL-sample¥ag903_sample¥ssp¥ssp_pcm5100a.c	0% 0/25	0% 0/1058	0% 0/773
C:¥DriveD¥CURRENT_WORK¥apps¥DB51903PF-demo-AXELL-sample¥ag903_sample¥ssp¥ssp_sample.c	0% 0/9	0% 0/92	0% 0/45
C: ¥DriveD¥CURRENT_WORK¥apps¥DB51903PF-demo-AXELL-sample¥ag903_sample¥timr¥timr_sample.c	0% 0/19	0% 0/901	0% 0/630
$\label{eq:c:storiveD} CURRENT_WORK \eqref{eq:stories} eq:stor$	0% 0/23	0% 0/719	0% 0/506
$\label{eq:c:storiveD} C: {\tt VORENT_WORK} apps {\tt VDS} 1903 {\tt PF-demo-AXELL-sample} ag903_sample} usbh {\tt sample_app_cdc.c} approx ap$	0% 0/13	0% 0/705	0% 0/378
$\label{eq:c:storiveD} C. {\tt VORKENT_WORKE} apps {\tt VORSED} D051903 {\tt PF-demo-AXELL-sample} ag903_sample {\tt vorsed} sample {\tt vorsed} app_hid.c$	0% 0/27	0% 0/888	0% 0/450
$\label{eq:c:storiveD} C: {\tt VORENT_WORK} apps {\tt VDS} 1903 {\tt PF-demo-AXELL-sample} ag903_sample} usbh {\tt sample_app_msc.c} approx ap$	42% 3/7	50% 138/272	46% 48/103
$\label{eq:c:storiveD} C: {\tt VORENT_WORK} apps {\tt VDS} 1903 {\tt PF-demo-AXELL-sample} ag 903_sample} usbh {\tt Sample_util.c} approximately a store and the sample {\tt VORENT_WORK} approximately a store and the sample {\tt VORENT_WORK} approximately approximat$	33% 3/9	29% 60/205	28% 26/90
C:¥DriveD¥CURRENT_WORK¥apps¥DB51903PF-demo-AXELL-sample¥ag903_sample¥usbh¥usb_sample.c	21% 4/19	36% 131/362	16% 26/154
C:¥DriveD¥CURRENT_WORK¥apps¥DB51903PF-demo-AXELL-sample¥ag903_sample¥wdt¥wdt_sample.c	0% 0/8	0% 0/139	0% 0/70
C: ¥DriveD¥CURRENT_WORK¥apps¥DB51903PF-demo-AXELL-sample¥demo-AXELL-sample¥com.c	66% 8/12	58% 105/178	33% 78/235
C: ¥DriveD¥CURRENT_WORK¥apps¥DB51903PF-demo-AXELL-sample¥demo-AXELL-sample¥file.c	0% 0/11	0% 0/457	0% 0/276
C: ¥DriveD¥CURRENT_WORK¥apps¥DB51903PF-demo-AXELL-sample¥demo-AXELL-sample¥fsif.c	0% 0/6	0% 0/127	0% 0/61
C:¥DriveD¥CURRENT_WORK¥apps¥DB51903PF-demo-AXELL-sample¥demo-AXELL-sample¥kernel_cfg.c	50% 1/2	50% 3/6	50% 1/2
C: ¥DriveD¥CURRENT_WORK¥apps¥DB51903PF-demo-AXELL-sample¥demo-AXELL-sample¥malloc_check.c	0% 0/4	0% 0/52	0% 0/25
C: *DriveD*CURRENT_WORK*apps*DB51903PF-demo-AXELL-sample*demo-AXELL-sample*sample.h	0% 0/1	0% 0/5	0% 0/1
C: *DriveD*CURRENT_WORK*apps*DB51903PF-demo-AXELL-sample*demo-AXELL-sample*sample_common.c	10% 2/19	10% 27/247	7% 11/140
C: *DriveD*CURRENT_WORK*apps*DB51903PF-demo-AXELL-sample*demo-AXELL-sample*sample_main.c	63% 7/11	71% 169	63% 76/120
C:¥DriveD¥CURRENT_WORK¥core¥inc¥solid_cs_assert.h	0% 0/2	/	0% 0/3
C:¥DriveD¥CURRENT_WORK¥rtos¥toppers_asp3¥asp3¥include¥queue.h	0% 0//		0% 0/6
C:¥DriveD¥CURRENT_WORK¥rtos¥toppers_asp3¥asp3¥include¥t_syslog.h	0	<u></u> 3	0% 0/7
C:#DriveD#CURRENT_WORK#rtos#toppers_asp3¥asp3¥kernel¥arch¥arm_gcc¥common¥arm.h		72	0% 0/12
C:#DriveD#CURRENT_WORK#rtos#toppers_asp3#asp3#kernel#arch#arm_gcc#common#arm_insn.h		0/60	0% 0/14
C:#DriveD#CURRENT_WORK#rtos#toppers_asp3#asp3#kernel#arch#arm_gcc#common#core_kernel_impl.h		6 0/68	0% 0/11
C-¥DriveD¥CLIRRENT_WORK¥rtos¥tonners_asn?¥kernel¥wait_h		<u>/1% 0/60</u>	0% 0/7
RTOS K-7-7-			

131			······又子列出刀
132		. **	********************************/
133		voi	d·COM_PutStr(int ^{erter} ter)
134	17	{	ソースコート上に美行回数を
135	17	÷	static uint8_t グラフィカルに表示
136	17	÷	ComMsg→ ·*·dtq;
137	17	÷	ER > > ercd
138	17	÷	uinta size;
139	17	7	<pre>uint8_t* buf;</pre>
140	17	÷	
141	17	÷	<pre>size = (uint32_t)strlen((char const*)str);</pre>
142	17	÷	if(COM_PUTBUF_SIZE <size) td="" {<=""></size)>
143	0	+	→ size·=·COM_PUTBUF_SIZE;·/*·Bufサイズで切る·*/
144	0	÷	}
145	17	÷	if(COM_PUTBUF_CNT<=cnt) {
146	2	÷	→ cnt=0;
147	2	÷	}
148	17	÷	dtq·=·&ComPutMsgBuf[cnt];
149	17	÷	<pre>buf -= · &ComPutBuf[cnt++][0];</pre>
150	17	÷	<pre>sys_memcpy(buf, str, size);</pre>
151	17	÷	
152	17	+	<pre>dtq->cmd = COMCMD_TX_STRING;</pre>
153	17	÷	dtq->size = size;
154	17	÷	dtq->buf··=·(void*)buf;
155	17	÷	<pre>ercd = tsnd_dtq(ComDtqID, (intptr_t)dtq, 100*1000);</pre>
156	17	÷	if(E_OK·!=·ercd)·{
157	0	÷	<pre> ComInternalError((int32_t)ercd); </pre>
158	0	÷	}
159	17	÷	return;
160	17	}	

ソースコード単位のカバレッジサマリーを一覧表示



コードカバレッジを使うための準備

使うために前もって設定すべき内容

コードカバレッジを使うための準備

• コードカバレッジは Clangコンパイラが必要 「ソリューション構成」で、"Debug_clang" を選択



コードカバレッジを使うための準備

プロジェクト単位で有効化
 1. ソリューションエクスプローラのプロジェクトで右クリック
 2. ポップアップメニューからプロパティを選択

	ソリューション エクスプロー	5 -			• 4 ×				
	· ソリューション エクスプロ-	ラー の	検索 (Ctrl+:)		ب م				
	🗟 solid_tim	er.h			-				
	++ solid_vec	tor.c							
	++ solid vlin	k.c							
	🗄 solid_vlin	k.h							
$\boldsymbol{<}$		*	ビルド(U)						
		>	リビルド(E)						
	digity	\sim	クリーン(N)						
	👂 📁 dmac		プロジェクトのみ(J)	•					
	👂 📁 dsp		ここまで検索(S)						
	eqs	a	新しい ソリューション エクスプローラー のビュー(N)						
	P = gpio								
	i2c		ビルト1公仔関係(B)	•					
	👂 🚚 jpg		追加(D)	•	1.11				
	👂 ≓ openv	0	ローダブルアプリケーションパッケージの作成						
	👂 📁 osp	֯F	スタートアップ プロジェクトに設定(A)						
	▷ 💭 spiflas	5	デバッグ(G)	•					
	v ,≡ ssp ≥ imr	ж	切り取り(T)	Ctrl+X					
	👂 🎜 uart	A	貼り付け(P)	Ctrl+V					
	🔺 🚄 usbh	10	名前の変更(M)	F2					
	*+ sa	0							
	B sa								
	<u>ローカル テハック 1919</u> ト	<u> </u>	7U/(71(K)		2				
	- ₽ ×	呼び	「出し履歴		• 4 ×				
			名前		言語				



コードカバレッジを使うための準備

・プロジェクト単位で有効化
 3. [構成プロパティ]-[C/C++] [コードカバレッジ]を選択
 4. [コードカバレッジの有効化]
 を、「はい」に設定



コードカバレッジを使うための準備

ソースファイル単位で有効化
 1. ソリューションエクスプローラの各ソースで右クリック
 2. ポップアップメニューからプロパティを選択

ソリューション エクスプローラー				• 4 ×					
○ ○ ☆ 'o - 司 ⋟									
ソリューション エクスプローラー の検索 (Ctrl+:)									
 ▶ == spiflash ▶ == ssp ▲ == timr 				^					
 ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓	<i>₹</i>	開く(O) ファイルを開くアプリケーションの選択(N) コンパイル(M)	Ctrl+F7						
++ com.c デバッグ 例外 ソリューション エク		ここまで検索(S) 新しい ソリューション エクスプローラー のビュー(N)							
プロパティ	ж	切り取り(T)	Ctrl+X	- 4 ×					
timr_sample.c	Ð	⊐ピ-(Y)	Ctrl+C	-					
2↓ □	X	削除(V)	Del						
	10	名前の変更(M)	F2						
	۶	לםוּ(דָּז(R) ¥	、 、						
			$\overline{\langle}$						



コードカバレッジを使うための準備

 ソースファイル単位で有効化
 3. [構成プロパティ]-[C/C++]-[コードカバレッジ]を選択
 4. [コードカバレッジの有効化] を、「はい」に設定



コードカバレッジの使い方

実行する際の手順

9

enjoy **D**evelopment

コードカバレッジの使い方

- コードカバレッジを有効化した後、有効化したプロジェクト と有効化したソースファイルを再ビルド
- 2. ビルドが成功したら、デバッグ実行。カバレッジを確認した い時点まで実行したら「すべて中断」
- 3. メニューの[デバッグ(D)] -[ウインドウ(W)] -[コードカバレッジ]を選択

ID-ID	E					∇
(B)	デバ	ッグ(D) ツール(T) KMC(SOLID_V7/	A_ARM) ウィンドウ(W)	^	ルプ(H)	
ッ・		ウィンドウ(W)	•	5	ブレークポイント(B)	Alt+F9
フサイ		続行(C)	F5	Ľ	デバッグ 例外	
	н	すべて中断(K)	Ctrl+Alt+Break	E.	出力(O)	
	•	デバッグの停止(E)	Shift+F5	Ô	タスク(S)	Ctrl+Shift+D, K
	Х	すべてデタッチ(D)		(==	並列スタック(K)	Ctrl+Shift+D, S
·? ?:		すべて中止(M)			並列ウォッチ(R)	•
15 51	ð	再起動(R)	Ctrl+Shift+F5		ウォッチ(W)	+
222	¢®	プロセスにアタッチ(P)		F	自動変数(A)	Ctrl+Alt+V, A
		その他のデバッグターゲット	•	[*;]	ローカル(L)	Alt+4
		例外(X)	Ctrl+Alt+E	Þ	イミディエイト(I)	Ctrl+Alt+I
	ς.	ステップ イン(I)	F11	>	PARTNER コマンド ウィンドウ	
MODI	G,	ステップ オーバー(O)	F10	Œ	呼び出し履歴(C)	Alt+7
	G	ステップ アウト(T)	Shift+F11	F	コードカバレッジ	×
MOI	⇔	クイック ウォッチ(Q)	Shift+F9		קב	
		ブレークポイントの設定/解除(G)	F9	12.	スレッド(H)	Ctrl+Alt+
		ブレークポイントの作成(B)	•	=	モジュール(O)	Ctrl+Alt+U

コードカバレッジの使い方

1. コードカバレッジ左上のアイコン 「解析」をクリック

com.c 🖻 usb_sample.c 🖻	scov.c 🛱			
ッシュー 🗄 🏗 🔯			関数カバレッジ 行カバレッジ	範囲力
		nple¥ag903_sample¥timr¥timr_sample.c	0% 0/19 0% 0/901	0%
		pple¥ag903_sample¥uart¥uart_sample.c	0% 0/23 0% 0/719	09
		ple¥ag903_sample¥usbh¥sample_app_cdc.c	0% 0/13 0% 0/705	0%
		nple¥ag903_sample¥usbh¥sample_app_hid.c	0% 0/27 0% 0/88	0%
		nple¥ag903_sample¥usbh¥sample_app_msc.c	42% 3/7 50% 138/272	46%
		nple¥ag903_sample¥usbh¥sample_util.c	33% 3/9 29% 60/205	28%
		nple¥ag903_sample¥usbh¥usb_sample.c	21% 4/19 36% 131/362	16%
		nple¥ag903_sample¥wdt¥wdt_sample.c	0% 0/8 0% 0/139	0
		nple¥demo-AXELL-sample¥com.c	66% 8/12 58% 105/178	33%
		nple¥demo- AXELL- sample¥file.c	0% 0/11 0% 0/457	0%
		nple¥demo-AXELL-sample¥fsif.c	0% 0/6 0% 0/127	0
		nple¥demo-AXELL-sample¥kernel_cfg.c	50% 1/2 50% 3/6	5
		nple¥demo- AXELL- sample¥malloc_check.c	0% 0/4 0% 0/52	0
		nple¥demo-AXELL-sample¥sample.h	0% 0/1 0% 0/5	i (
		nple¥demo-AXELL-sample¥sample_common.c	10% 2/19 10% 27/247	7% 1
		nple¥demo-AXELL-sample¥sample_main.c	63% 7/11 71% 166/233	63% 7
			0% 0/2 0% 0/2	0
		Jeue.h	0% 0/6 0% 0/36	0
		syslog.h	0% 0/7 0% 0/63	0
		h¥arm_gcc¥common¥arm.h	0% 0/12 0% 0/72	. 0%

コードカバレッジの使い方

カバレッジのサマリーが 表示されます。

関数単位の カバレッジ	行単位の カバレッジ		ーシックブロ のカバレ	コック単 [,] ッジ
	関数カバレッジ	行カバレッジ	範囲カバレッジ	
ple¥ag903_sample¥timr¥timr_sample.c	0% 0/19	0% 0/901	0% 0/630	
ple¥ag903_sample¥uart¥uart_sample.c	0% 0/23	0% 0/719	0% 0/506	
ple¥ag903_sample¥usbh¥sample_app_cdc.c	0% 0/13	0% 0/705	0% 0/378	
ple¥ag903_sample¥usbh¥sample_app_hid.c	0% 0/27	0% 0/888	0% 0/450	
ple¥ag903_sample¥usbh¥sample_app_msc.c	42% 3/7	50% 138/272	46% 48/103	
ple¥ag903_sample¥usbh¥sample_util.c	33% 3/9	29% 60/205	28% 26/90	
ple¥ag903_sample¥usbh¥usb_sample.c	21% 4/19	36% 131/362	16% 26/154	
ple¥ag903_sample¥wdt¥wdt_sample.c	0% 0/8	0% 0/139	0% 0/70	
ple¥demo- AXELL- sample¥com.c	66% 8/12	58% 105/178	33% 78/235	
ple¥demo- AXELL- sample¥file.c	0% 0/11	0% 0/457	0% 0/276	
ple¥demo- AXELL- sample¥fsif.c	0% 0/6	0% 0/127	0% 0/61	
ple¥demo- AXELL- sample¥kernel_cfg.c	50% 1/2	50% 3/6	50% 1/2	
ple¥demo- AXELL- sample¥malloc_check.c	0% 0/4	0% 0/52	0% 0/25	
ple¥demo- AXELL- sample¥sample.h	0% 0/1	0% 0/5	0% 0/1	
ple¥demo- AXELL- sample¥sample_common.c	10% 2/19	10% 27/247	7% 11/140	
ple¥demo- AXELL- sample¥sample_main.c	63% 7/11	71% 166/233	63% 76/120	
	0% 0/2	0% 0/2	0% 0/3	
eue.h	0% 0/6	0% 0/36	0% 0/6	
yslog.h	0% 0/7	0% 0/63	0% 0/7	
¥arm_gcc¥common¥arm.h	0% 0/12	0% 0/72	0% 0/121	2

コードカバレッジの使い方

サマリのソースコードをダブルクリック するとカバレッジの詳細表示になります

	関数カバレッジ	行カバレッジ	範囲カバレッジ
903_sample¥timr¥timr_sample.c	0% 0/19	0% 0/901	0% 0/630
903_sample¥uart¥uart_sample.c	0% 0/23	0% 0/719	0% 0/506
903_sample¥usbh¥sample_app_cdc.c	0% 0/13	0% 0/705	0% 0/378
903_sample¥usbh¥sample_app_hid.c	0% 0/27	0% 0/888	0% 0/450
903_sample¥usbh¥sample_app_msc.c	42% 3/7	50% 138/272	46% 48/103
903_sample¥usbh¥sample_util.c	33% 3/9	29% 60/205	28% 26/90
903_sample¥usbh¥usb_sample.c	21% 4/19	36% 131/362	16% 26/154
903_sample¥wdt¥wdt_sample.c	0% 0/8	0% 0/139	0% 0/70
emo-AXELL-sample¥com.c	66% 8/12	58% 105/178	33% 78/235
emo- AXELL- sample¥file.c	0% 0/11	0% 0/457	0% 0/276
emo- AXELL- sample¥fsif.c	0% 0/6	0% 0/127	0% 0/61
emo-AXELL-sample¥kernel_cfg.c	50% 1/2	50% 3/6	50% 1/2
emo-AXELL-sample¥malloc_check.c	0% 0/4	0% 0/52	0% 0/25
emo-AXELL- sample¥sample.h	0% 0/1	0% 0/5	0% 0/1
emo-AXELL-sample¥sample_common.c	10% 2/19	10% 27/247	7% 11/140
	6004 7144		C224 70 (420

sample_app	_msc.c i	i -Þ	×	コード カバレッジ	com.c 🕯	i usb_sai
133	1	÷				
134	1	÷	1*	•	*/	
135	1	÷	1*	トバッファ設定	-*/	
136	1	÷	1*	۰ <u>۰</u>	*/	
137	1	÷	/*	*・書き込みデー	夕設定·*/	
138	3	÷	fo	or(i·=·0;·i·<	USBH_MAX_CL	S_MSC_NUM; i
139	2	÷	÷	sample_mem:	set(g_MscTas	kInfoTbl[i].
140	2	÷	÷	writeBuff[6]=·('1'·+·i);
141	2	÷	÷	sample_mem	cpy(g_MscTas	kInfoTbl[i].
142	2	÷	}			
143	1	÷				
144	1	÷	1*	·	*/	
145	1	÷	1*	トメールボック	ス生成・*/	
146	1	÷	1*	•	*/	
147	1	÷	1*	トパラメータ設	定 */	
148	1		- cn	nbx.mbxatr→→	= TA_TFIFO	- TA_MFIFO;
149	1		- cn	nbx.maxmpri⇒	= 1;	
150	1		- cn	nbx.mprihd→→	= (intptr_	t)NULL;
151	1					
152	1	÷	1*	* 接続切断タス	ク用メールボ	ックス生成・*/
153	1	÷	g_	_mbxid_msc_co	nn·=·acre_mb	x(&cmbx);
154	1	÷	if	F(g_mbxid_msc	_conn < < 0) • {	
155	0	÷	÷	/*・エラー・*	1	
156	0	÷	÷	return;		
157	0	÷	}			
158	1					
159	1	÷	/*	*	*/	
160	1	÷	/*	* 接続/切断夕	スク生成・*/	
161	1	÷	/*	*	*/	13
162	1	÷	11	▶ パラメーク設	定-*/	10

コードカバレッジの使い方



コードカバレッジの使い方

解析実行後は、ソリューション エクスプローラでダブルクリックでも カバレッジの詳細が表示されます

ソリューション エクスプローラー
○ ○ ☆ ːo - 司 👂 🗕
ソリューション エクスプローラー の検索 (Ctrl+:)
👂 🚛 ssp
🔺 🚛 timr
++ timr_sample.c
🕨 🚚 uart
🔺 🚛 usbh
++ sample_app_cdc.c
🖻 sample_app_cdc.h
++ sample_app_hid.c
sample_app_hid.h
🏘 sample_app_msc.c
🖻 sample_app_msc.h
++ sample_util.c
🖻 sample_util.h
++ usb_sample.c
li usb_sample.h

sar	nple_app	_msc.c 🕯	-12	× J-	-ドカバレッジ	com.c 🕯	usb_sar
	133	1	÷				
	134	1	÷	/*		*/	
	135	1	÷	/*·/	(ッファ設定	*/	
	136	1	÷	/*		*/	
	137	1	÷	/*·書	き込みデー	夕設定·*/	
	138	3	÷	for(:	i-=-0;-i-<	USBH_MAX_CL	S_MSC_NUM; i
	139	2	÷	÷	sample_mems	set(g_MscTas	kInfoTbl[i].p
	140	2	÷	÷ 1	writeBuff[(5]=·('1'·+·i);
	141	2	÷	÷	sample_memo	cpy(g_MscTas	kInfoTbl[i].p
	142	2	÷	}			
	143	1	÷				
	144	1	÷	/*		*/	
	145	1	÷	1*.7	(ールボック)	ス生成・*/	
	146	1	÷	/*		*/	
	147	1	÷	/*·/	(ラメータ設)	定-*/	
	148	1		- cmbx	.mbxatr⇒⇒	= TA_TFIFO	 TA_MFIF0;
	149	1		- cmbx	.maxmpri⇒	= 1;	
	150	1		- cmbx	.mprihd⇒⇒	= (intptr_	t)NULL;
	151	1					
	152	1	÷	/*·接	続切断タス	ク用メールボ	ックス生成・*/
	153	1	÷	g_mb;	xid_msc_com	nn:=:acre_mb	x(&cmbx);
	154	1	÷	if(g	_mbxid_msc	_conn < 0) {	
	155	0	÷	÷	/*・エラー・*	1	
	156	0	÷	÷ i	return;		
	157	0	÷	}			
	158	1					
	159	1	÷	/*		*/	
	160	1	÷	/*·接	続/切断夕	スク生成 */	
	161	1	÷	/*		*/	15
	162	1	÷	1*-1	(ラメータ設	定-*/	10

コードカバレッジの使い方



- ・関数単位のサマリ表示 関数単位のカバレッジを、カ ウント数の順位で一覧表示
- ・ 関数名でソートすることもで きます。

問題が起きたときは...

17

問題が起きたとき

kernel_cfg.cの_kernel_call_inirtn()で "Data Abort Exception" が発生した場合、solid_cs.ldリンカスクリプトに、コードカバレッジ用の変更が入っていない為起きています。付録で修正すべき点を説明しています。

Х Е Ј 4	- ₽ ×	デバッグ例外 マ 早 ×
アドレス: 0x00000000 - 🔃 自動	-	lu la
0x00000000 ??		Data Abort Exception Access Address: 0.00882560 Access type: Read PC: 0xf0c57024 Status: Translation fault, 2nd level
337 /* 338 -*··CPU·Exception·Management·Functions 339 -*/ 340	ф шеломически по произонали по пр	C:\DriveD\CURRENT_WORK\apps\DB51903PF-demo-AXELL-sample\demo-A Data Abort Exception in 0xf0c57024 at address 0xf0e82560
341 /* 342 -*··Initialization Routine 343 -*/ 344	National and an and an analysis of the second secon	
<pre>346 _kernel_call_inirtn(void) 347 {</pre>		
349 } 350 351 /*	Image: State of the s	
352 **·Termination-Routine 353 ·*/ 354	 More exception, and an encourse of the pro- man of the provided sectors of the pro- man of the provided sectors of the pro- sectors of the pro- sectors of the pro- man of the pro- man of the pro- man of the pro- man of the pro- sectors of the pro- tion of the pro- tion of the pro- tion of the pro- man of the pro- tion of the pro- man of the pro- tion of the pro- tion of the pro- man of the pro- tion of the pro- tion of the pro- tion of the pro- man of the pro- tion of the pro- man of the pro- man of the pro- tion of the pro-	
<pre>355 void 356 _kernel_call_terrtn(void) 357 {</pre>	300 (2000) 1000 (2000) 1000 (2000)	↓

問題が起きたとき

- ・メニューの [デバッグ]-[ウインドウ]の先のメニューに [コードカ バレッジ]が無い。
 - SOLID-IDE のバージョンがVer 1.1.11.0 以降か確認
 - [すべて中断]状態か確認。[デバッグ開始]前であったり、[デバッグの終 了]をした後では、コードカバレッジは機能しません。

問題が起きたとき

 「解析」 をクリックしたら、サマリが表示されずエラー カバレッジのデータが無かった場合に起きます。
 少なくとも一つのプロジェクトかソースのカバレッジを有効になっているか確認してください。

コードカバレッジ	😐 🗙 sample	_app_cdc.c @	sample_app_msc.c i	i com.c	a usb_sa	mple.c 🕯	«
宮 袖 ピュー:	カバレッジ	- 🗄 🎲 🔅					
error: C:¥DriveI coverage: No c error: Could no	D¥CURRENT_WC overage data fo ot load coverage	DRK¥apps¥DB51903 bund information	PF-demo-AXELL-sample	e¥Debug_clang¥	exeClang_SOLID¥d	emo-AXELL-sam	ple.out: Failed to load

メモリ使用量、処理時間 オーバーヘッド

enjoy **Development**

コードカバレッジのオーバーヘッド

※対象コードの複雑さ(ベーシックブロックの量)に依存します

- ・メモリ使用量 ベーシックブロック1カ所につき、以下のサイズが
 - 記録用カウンタ 64bit (8bytes)
 - 記録用実行コード6~8 step (24~32bytes) ※上記カウンタの加算
- 処理時間オーバヘッド
 ベーシックブロック実行時に記録用実行コードの実行

一例としてdhrystone

・dhrystoneアプリでの処理時間測定結果 (Cortex-A5 400MHz)

カバレッジ	Dhrystone 一回の処理時間 (μ秒)			
	最適化無し (-00)	最適化有り (-02)		
無効	585	205		
有効	742	357		

- 開発者が以下の用途に使うのには最適
 - 期待した部分が実行されているかどうかの確認
 - 資料が少ない他人が書いたコードの解析時
 - ・コードの効率化の為、実行回数の多い部分の抽出

リンカスクリプト solid_cs.ld の修正点

・.dataセクションに以下の記述を追加して下さい

```
.data ALIGN(4K) : {
    _solid_data_start = .;
    *(.data .data.*)
    *(.gnu.linkonce.d.*)
```

```
/****追加ここから****/
/* Append the LLVM profiling */
. = ALIGN(8);
PROVIDE(__start___llvm_prf_cnts = .);
KEEP(*(__llvm_prf_cnts))
PROVIDE(__stop___llvm_prf_cnts = .);
```

```
. = ALIGN(8);
PROVIDE(__start___llvm_prf_data = .);
KEEP(*(__llvm_prf_data))
PROVIDE(__stop___llvm_prf_data = .);
```

```
/****追加ここまで*****/
```

_solid_data_end = .;

}

以上です