

Arm ETM Programming Dialog

Release 02.2024



TRACE32 Online Help	
TRACE32 Directory	
TRACE32 Index	
TRACE32 Documents	þ
ICD In-Circuit Debugger	
Processor Architecture Manuals	
Arm/CORTEX/XSCALE	
Arm ETM Programming Dialog	1
Initialization	3
Initialization of the ETM	3
Programming	4
How to Start	4
Elements in the Dialog Box	4
Actions	6
Events	7
Levels	8
ETM ALL 0, Clear, Load, View, Save	12
Definitions	13
Address/Range Definition	13
Data Definition	17
Counter Definition	19
Examples	21
Example 1: Selective Trace on an Address Range	21
Example 2: Selective Trace on an Address Range defined through a Symbol	23
Example 3: Selective Trace on Access to a Symbol	25
Example 4: Trace the Entrance and Exit of Function Sieve	27
Example 5: Trace the first 200H Cycles in Function Sieve	28
Example 6: Trace all, when Function Sieve is reached goto Level 1 and stop Sampling a Debugging after 5 Cycles	and 29

Initialization

Initialization of the ETM

Initialize the ETM by using **ETM Settings** in the **Trace** menu or by entering the command **ETM.state**.

a ™B::ETM				_	
_ etm	- TraceInclude	 configuration 	- port		
C OFF		CycleAccurate	🗖 HalfRate	AComp:	8.
🖲 ON	- TraceExclude	🗖 STALL	- PortSize	AUsed:	0.
		CPRT	8 💌	DComp:	8.
RESet		🗖 BBC	- PortMode	Map:	16.
O CLEAR	- DataViewInclude	🗖 DBGRQ	Normal 💌	Counter:	4.
I Register		 SmartTrace 	- ProcID	Seq:	Yes
	– DataViewExclude	1-8 💌	OFF 💌	Extin:	4.
_ TraceDelay		– FifoLevel –	- DataTrace	ExtOut:	4.
0.		16.	Both 💌	FifoFull:	Yes
	- FifoFullInclude	– MapDecode –		Protocol:	0.
		0x0			
	- FifoFullExclude				

The broadcasting of the **program flow** can be controlled by **TraceInclude** (restrict the broadcasting of the program flow to the defined range) or **TraceExclude** (do not broadcast the program flow within the defined range). Either **TraceInclude** or **TraceExclude** can be specified.

The broadcasting of the **data flow** can be controlled by **DataViewInclude** (restrict the broadcasting of the data accesses to the defined range) or **DataViewExclude** (do not broadcast the data flow within the defined range).

For further details of this window please look for the manual **RiscTrace for ARM-ETM**.

How to Start

The **ETM Programming** Dialog can be found on any TRACE32 CD dated September 2000 and later. To open the **ETM Programming** dialog use *Trigger Dialog* ... in the *Trace* menu.

An empty dialog box will appear. Please go on reading this manual to get an overview on the functionality of the dialog.

Elements in the Dialog Box

Remark

- The intention of the **ETM Programming** dialog is to provide an intuitive and easy to use interface to program the **Embedded Trace Macrocell (ETM)** of ARM7 / ARM9.
- The ETM Programming Dialog Window does not represent the full functionality of the ETM.

Overview

ETM Program	ming			ļ	6			
Definition Ac A-Range : V. B-Range : C-Range :	ddress .range(sieve)	Address V or V or V or V or	Addre	ss definit	tion field			
Value DATA0: DATA1: - Level0-	· D	V BYTE		V OUNTO: [OUNT1: [alue Co	ount defii	nition field	
Action Trace	IF A	dress/Range Range 🗾	RD/WR		Data	Co &	ount	Even -
	→ IF → IF → IF	- - -	& &	• & • & • &		• & • & • &		•
- Level1	→ IF → IF	y y	& &	- & - & - & - &		* & * & * &		v V V
	F 	× × ×	& & &	× & × & × &		* & * & * &		7 7 7
ETM ALL 0 C	LEAR LOAD	VIEW SAVE	₽ 	rogram	Program & Sav	e prograr	am & Save & C mming	lose About

Actions

ETM Programming Opfinition Address A-Range : B-Range : C-Range : Value	Address V or V or V or V or V or	V V V Value	
DATA0: DATA1:	V BYTE • V BYTE •	COUNTO: COUNT1:	V
Level0 Action Actions IF Trace View Data Goto Level1 Goto Level2 Decr COUNT0 Decr COUNT1 Reload COUNT1 Trigger TRACE IF Level2	\ddress/Range RD/	WR Data	Count Count Count Count Count
ETM ALL 0 CLEAR LOAD	VIEW SAVE	rogram Program	& V V V V V V V V V V V V V V V V V V V

An ETM Trigger Program performs ACTIONS depending on the state of the user program/target hardware.

Тгасе	Trace instruction execution cycle						
View Data	w Data Trace data access cycles						
Trigger TRACE Stop tracing							
Decr Count0 Decr Count1Decrement the counter identified by the label COUNT0 or COUNT1.							
Reload Count0 Reload Count1	Set the counter identified by the label COUNT0 or COUNT1 to value specified in definition.						
Goto Level0 Goto Level1 Goto Level2	Go on tracing according to the actions defined under Level0 , Level1 or Level2.						

Events

– Level0 –				
Action	Address/Range	RD/WR	Data	Count
↓ IF	• &		k 🔽	&
↓ IF			k 🔽	& 🗸
- IF	B-Range		k 💽	&
. IF	C-Range	•	k 💽	&
- Level1	AC-Range			
🚽 IF	BC-Range	3 -	k 🗸	& _
🚽 IF	NOT B-Range	3 -	k 🔽	& 🗸
🚽 🖂 🖬	NOT C-Range	3 🖵 🗌	k 🔽	& 🗸
- Level2				

The state of the user program/target system is described by different characteristics. Different characteristics combined through **LOGICAL AND** form an event. An action will only be performed if the user program/target system is in the state defined by the event. The characteristics are:

Address/Range	Address or address range To be defined in the address definition fields A-Range , B-Range or C-Range
RD/WR	CPU cycle type FETCH, EXECUTE, READ , WRITE or ACCESS (READ or WRITE)
Data	CPU data To be defined in the data definition fields DATA0 or DATA1
Count	Counter or Time Counter To be defined in the count definition fields COUNT0 or COUNT1

If no characteristic is chosen the action is carried out at every CPU cycle.

Three different levels can be used to define a set of actions. The levels can be referenced by the logical names **Level0**, **Level1** and **Level2**. Within the ETM the levels are implemented as states. Use the action **Goto Level x** to let the system switch to Level x and to carry out only the actions under Level x.



After programming the ETM with **Goto Level** *x* actions the state sequencer will be set to **Level 0**.

Be aware Goto Level x used in different Levels does not mean the same. It always implies a transaction from the current Level to Level x. e.g using Goto Level 2 in the area at Level 0 means the transaction from Level 0 to Level 2

	- Level0								
I	Action		Address/Range		RD/WR		Data		Count
ł	Goto Level1	IF	A-Range 🔹	&	•	&	-	&	-
I	•	IF	•	&	•	&	•	&	•
I	•	IF	•	&	•	&	•	&	•
I	•	IF	•	&	•	&	•	&	•
►	Level1								
I	Trace	IF	•	&	•	&	•	&	•
I	View Data 🗾	IF	•	&	•	&	•	&	•
I	-	IF	•	&	•	&	•	&	·
	– Level2 –								
		IF	-	2		2		2	-

For this example trace and data access trace is activated if A-Range is reached. The state sequencer changes form Level 0 to Level1 if an address within A-Range is reached. What you don't see is that the actions Trace and View Data only take place if the system is in Level 1. That means Trace and View Data have a implied condition "if Level1".



Special Feature in Level 0:

Actions in Level 0 that are used after a **Goto Level 1** or **Goto Level 2** command get the implied condition "if Level 0". If they are used before the Goto Level x command Actions are carried out in **every** Level.

	- Level0					<u> </u>
	Action	Address/Range	RD/WR	Data	Count	
ſ	Goto Level1 - IF	A-Range 💽	&	- &	▼ &	-
П	Trace IF	•	&	- &	▼ &	-
	Only if Level 0 🖵 IF	•	&	- &	▼ &	-
	. F	•	&	- &	▼ &	-
l	Level1					
	View Data 🗾 IF	•	&	- &	<u>→</u> &	-
L	Goto Level0 🗾 IF	NOT A-Range 📃	&	- &	<u>→</u> &	•
	- IF		2	- 8	- 2	

	► Level0					
	Action Trace in all le	vels Address/R	ange R	D/WR	Data	Count
Ш	Trace	. F	→ &	→ &		k 🔽
١r	Goto Level1	▼IF A-Range	• &	• &		k 💽
ш			- &	↓ &		k 🔽
11			• &	• &		k 💽
4	►Level1					
	View Data		• &	• &		k 💽
4	Goto LevelO	F NOT A-Ra	ange 🔽 🌡	▼ &		k 🔽
	ſ					



Restrictions due to implied Level x condition:

Due to the fact that only one logical combination AND and OR are possible within an event (see also Address Definition) in Level 1 and Level 2 and in the area after a Goto Level x of Level 0 only one further condition is allowed. This means either address range or counter could be used. The first part of the event is used by the Level x condition. If more than one combination has to be used to build an event it will result in an error message.

🐣 ETM Programming					- D ×
– Definition –––––					
Address	3	Address			
A-Range : v.range	e(flags)	/ or	V		
B-Range : D:0x12	2000x1300	/ or	V		
C-Range :	N	/ or	V		
Value					
				1	vl
			COUNTI:	,	- <u>v</u>
DATAL J			COONTI. J		<u> </u>
– Level0 –					
Action	Address	/Range RD	/WR Data		Count
Goto Level1	▼IF A-Rang	e 🔹 🌾	- &	₹ &	•
		• &	- &	₹ &	_
		• &	↓ &	→ &	•
		• &	- &	₹ &	_
- Level1					
Trace		e 🔽 🕷	- &	√ &	
	F IF	• &	- &	↓ &	
	F F	- &	- &	÷ &	NOT COUNTO
- Level2	_ ,		_ /		COUNT1
	IF 🛛	- &	- &	<u>ه</u> ا ب	
B::					
loo many condi	tions define	ed for Level1	. line 1. !		
emulate Data	Var	trigger devic	ces Analyzer	PERF	
SR:000093BC \\armle\	\arm\sieve				

_		_	-
	_	 Proor	2000000
	_	 I IUUI	

	пI	×.
_		

- Definition							
	Address		Address				
A-Range :	v.range(fla	gs)	V or		V		
B-Range :	D:0x1200-	-0x1300	V or		V		
C-Range :			V or		V		
V	alue				Value	e	
DATA0:				- C			V
DATA1:					DUNT1:		
, 				_	,		_
- Levelu		و د د او او او	. /D	DD 4-70	D-1	_	Count
Action		Addres	s/Hange	RU/WR. ⊐l E	Data	a	
Goto Level1		F A-Ran	ge 💌	<u> </u>	<u> </u>	-	ά <u> </u>
Trace		<u> </u>		· &	▼ &	•	& COUNTO 🔄
		IF			→ &	•	& 🗸
		F IF B.Ban	ge De		- &	*	& 🔹
_ _ Level1		C-Ran	ge	,			
View Data		AB-Ra	nge		- 2		1 ×
		ELLE BC-Ba	nge nge	┣━━━	, [
			N-Range		°	<u> </u>	
			I-Range		▼ &	-	× ·
– Level2 – – –			,-Hange				
R::							
Too many	conditi	ons defin	ied for L	evel0. D	ine Z. !		
emulate	Data	Var	trigger	devices	Analyzer	PERF	
SR:000093BC	\\armle\arm	n\sieve					

At the bottom of the dialog you find the buttons to control the dialog.

ETM ALL 0 CLEAR LOAD VIEW SAVE	Program Program & Save Program & Sav	e & Close
Actual used file:	V	About

If you save the contents of the dialog a **PRACTICE** file is generated. To program the ETM either use the buttons Program or Program&Save or Program & Save & Close. You can also run the generated PRACTICE file from the command line of TRACE32 with the command **DO** *<file>* to program the ETM.

Use the button **ETM ALL 0** to set all ETM registers to 0. Compared to **ETM.RESet** this command does not reset the address and data comparators, the counter actions and the sequencer actions in the ETM. This could cause strange result in the output.

Use the button CLEAR to reset the dialog window if you want to begin from the scratch.

With the button **LOAD** an earlier designed ETM program file can be read into the dialog window. The ETM is not programmed.

Only ETM programs written with the ETM Dialog can be reloaded into the dialog.

Press the button **VIEW** and you will see the contents of the dialog written down in statements of the ETM programming language. Use this button now and then to learn the basics of the ETM programming language.

If you do not want the content of the dialog to be save to a file use the button **Program** to program the ETM.

Program & Save will write the contents of the dialog to the file which is specified under **Actual used file**. After that the ETM will be programmed. If no file name is given you will be asked to define one. The ETM is successfully programmed if you get the message in the state line of TRACE32.

B:: ETM prog	jammed sud	cessfully	ļ		
emulate	Data	Var	trigger	devices	Analyzer
SR:00009174	4 \\armle\arm'	\main			

Otherwise the error is shown in the state line of TRACE32 or a message box appears.

Program & Save & Close works the same as **Program & Save** except that the dialog will be closed after the ETM is programmed successfully.

Address/Range Definition

Å ETM Progr	amming			
– Definition —				
	Address		Address	
A-Range :		V or		V
B-Range :		V or		V
C-Range :		V or		V

Up to three different address areas can be defined in this part of the dialog. This areas are later referenced by the logical names **A-Range**, **B-Range** or **C-Range**. The logical name **AB-Range** defines an area declared through the area **A-Range** combined with the area **B-Range** by **LOGICAL AND**.

Each area e.g. **A-Range** allows to define the address space through two different singular address parts. One of this part can hold a single address or an address range. The singular address parts are combined through **LOGICAL OR** to form the characteristic named **A-Range**.

If you don't know how to specify an address or an address range click the V button.

🗢 Define Address 👘				- 🗆 🗵
🔽 all values HEX			43	
Single Address	Mem.Class	Value/Symbol	V	
C Range	Mem.Class	Value/Symbol	Value/Symbol	V
C HLL-Range	Symbol	∀	CLEAR	ОК

With the appearing sub dialog a single address or any kind of range can be defined easily.

As you are familiar with the syntax of defining an address in TRACE32 just fill out the dialog.

Å ETM Prog	ramming			- D X
- Definition				
	Address	Address		
A-Range :	v.range(flags)	V or	V	
B-Range :		V or	V	
C-Range :		V or	V	

The defined address area will be used to define the condition for carrying out an action in the main dialog.

	🐣 ETM P	rogramming						
	- Definition	ı —						
		Address		Address				
	A-Range	v.range(flags)	V or		V			
	B-Range :	:	v or		V			
<	C-Range :	:	V or		V			
		Value				Value		
	BATAO:		V	BYTE 💽	COUNTO:			۷
	DATA1:		V	BYTE 💽	COUNT1:			V
	– Level0 –							
	Action	*	Address/Ran	ge RD/	WR	Data	Co	bur
	Trace	•	IF A-Range		• i	۵ (• &	
			IF	- 8		٤ 🗖	- 2	_

If not address area is defined at the time you want to choose an address condition you will be asked to do so.

🕸 ETM Programming		
- Definition		
Address	Address	
A-Range: \	/ or V	
B-Range: 📃 🗌	/ or V	
C-Range:	/ or V	
Value		
DATA0:	BYTE - Define Address	
DATA1:	V BYTE 💽 🗹 all values HEX	
1		Mem.Class Value/Symbol
	Single Address	
Action Address,	(Hange HL	
Trace IF	<u> </u>	Mem.Class Value/Symbol
	C Range	
■ IF B-Bang		
F C-Rang		Symbol
Level1 AB-Ran	ge 🖆 🔿 HLL-Range	V
IF BC-Ran		
	Range	-181 -11
	Range	



a ETM Prog	gramming							
– Definition –								
	Address		Address	:				
A-Range :	v.range(fla	igs)	V or D:0x12	340x1300	V			
B-Range :	P:0x94bc-	-0x9500	V or		V			
C-Range :			V or		V			
,	/alue				Val	ue		
				- I c	OUNTO:		V	
DATA1:					OUNT1:			
				_				
- LevelU								
Action		Addres:	s/Range	RD/WR	Da	ata	Count	
Trace			1	<u>- &</u>	<u> </u>		<u> </u>	<u>-</u>
		F A-Ban	е		<u> </u>		<u> </u>	<u> </u>
		F B-Rang	je je		<u> </u>		<u> </u>	<u> </u>
		F C-Ran	je nae		→ &		<u>→</u> &	•
- Level1		AC-Ra	nge					
			nge -Range		<u> </u>		<u>~</u> &	7
		IF NOT B	-Range		<u> </u>		<u> </u>	7
			-Hange	┱╼╝	~ &		- &	∇
Louol2								
B::								
Too many	conditi	ons defin	ed for L	evel0. l	ine 1. !			
emulate	Data	Var	trigger	devices	Analyzer	PERF	Port	

SR:00009174 \\armle\arm\main

© ETM Programming	- 🗆 🗙
- Definition	
Address Address	
A-Range : v.range(flags) V or D:0x12340x1300 V	
B-Range : P:0x94bc0x9500 V or V	
C-Range : V or V	
Value Value	
DATA0: V BYTE V COUNTO: V	
DATA1: V BYTE V COUNT1: V	
LevelO	
Action Address/Range RD/WR Data Count	
Trace IF A-Range 🔹 & FETCH 💌 & 💌 &	•
Levelt	
B::	
Too many conditions defined for Level0. line 1. !	
emulate Data Var trigger devices Analyzer PERF Port	
SR:00009174 \\armle\arm\main	

RD/WR.. and **Data** conditions are internally connected to Address/Range therefore you can use them without restrictions for the logical combination.



Be aware if you use RD/WR.. e.g FETCH with a special range e.g. A-Range you are not able to use A-Range with a different RD/WR.. condition e.g. ACCESS. The same applies if you use A-Range with condition DATA0/1.

Data Definition

Value	Value
DATA0:	V BYTE COUNTO: V
DATA1:	V BYTE COUNT1: V
- Level0	LONG

Two different data values can be used to observe the data on the data bus. The data values are later referenced by the labels **DATA0** and **DATA1**.

🌲 ETM Progr	amming			
– Definition —				
	Address	Address		
A-Range :	v.range(flags)	V or	V	
B-Range :		V or	V	
C-Range :		V or	V	
Va	alue		Value	
DATA0: 0x	1234	V WORD -	COUNTO:	V
DATA1:	1	V BYTE -	COUNT1:	V
- Level0				
Action	Addre	ess/Range RE	J/WR Data	Count
Trace		inge 💽 & 🕅	RITE 💽 & DATAO	• & •
	Ţ IF	• &	• & -	
		- 2.	- 2.	- 2

The data is defined through the value and the type. If you don't know how to specify a data value just click the V button.

Å Define Data		- D X
Type: BYTE 💌	R all values HEX	
Single Value	Value	
C Mask	Value Value	
	CLEAR	OK

With the appearing subdialog a single data, a data range or a mask can be defined easily.

With the type you specify the width of the value. **BYTE**, **WORD** and **LONG** are available.

As you are familiar with the syntax of defining data in TRACE32 just fill out the dialog.

	Value			
DATA0:	0x1234	V	WORD	Ŧ
DATA1:		V	BYTE	Ŧ

If there is no data defined at the time you want to use a data condition you will be asked to do so.

🎄 ETM Programming			= 0 ×
- Definition			
Address	A ddeese		
A-Range : v.range(flags)	Define Data		
B-Range :	Type: BYTE 💌	all values HEX	
C-Range :			
Value	Single Value	Value	
DATA0:		· · · · · · · · · · · · · · · · · · ·	
DATA1:	C Mask	Value Value	
- Level0		BIN	
Action Ad			
Trace IF A-			CLEAR OK
IF IF	. ₹	▼ &	
IF IF	- &	• & DATAU	
IF I	- &	• & •	δ

As the Data comparators of the ETM are strongly connected to the address range you cannot use a Data comparator without an address range. If you do so you get an error message.

Valu	e	١	Value	
DATA0: 0x1	V BYTE	COUNTO:		V
DATA1:	V BYTE	COUNT1:		V
– Level0 –				
Action	Address/Range	RD/WR	Data	Count
Trace	▼ IF ▼	& 🗸 &	• &	-
	F F	& 🔹 &	DATAO	
	F F	& 💽 &	DATA1	·
		•		

B::										
Address	condition	not def	ined for	data acc	ess in le	evel0. li	ne 1. !			
emulate	Data	Var	trigger	devices	Analyzer	PERF	Port	Step	Go	
SR:00009174 \\armle\arm\main stopped										

For controlling the flow two different counters can be defined. For each counter you can define the maximum value. The counters are referenced in the condition of an action by the labels **COUNT0** and **COUNT1**.

	Value		Value	
DATA0:	✓ B ¹	YTE 💽 COUNTO:	0x100	V
DATA1:	V B'	YTE 🚽 COUNT1:		
– Level0 —				
Action	Address/Range	RD/WR	Data (Count
Trace		• & • * * * * * * * * * * * * * * * * *		COUNTO 🗾

After programming the ETM the counter is set to 100H. If no action is used to decrement the counter COUNT0 (**Decr Count 0**) defined the counter is continuously decremented at **full system clock speed**.



If you use Trace with **condition COUNT0/1** sampling begins when the counter **COUNT0/1 is zero**. If you use Trace with condition **NOT COUNT0/1** sampling only takes place if the counter **COUNT0/1 is bigger than zero**.

To control the value of the counters the actions **Decr Count0/1** and **Restart Count0/1** are available.

Value DATA0: DATA1: Louel0	V BYTE V BYTE	COUNTO:	Value 0x100	v
Action	Address/Range	RD/WR	Data	Count
Trace	× &		×	& COUNTO - & - & - & - & -
Goto Level2 Decr COUNT0 Decr COUNT1 Reload COUNT0 Reload COUNT1 Trigger TRACE		\$		
, F	- &	- &	7	& _

Decr COUNT0/1 substracts 1 from the value of the counter COUNT0/1. **Reload COUNT0/1** sets the value of the counter COUNT0/1 to the value specified in the definition.



If you don't use **Decr Count x** to set the counter or you don't use an event for this action the counter decrements at **full system clock speed**. For this case it is recommended to use the **Reload COUNT** x action to reload the counter at a definite point otherwise the counter will be run to zero till you start the program run.

If you don't know how to define the value of a counter just click the V button.

🕭 Define Counter	= □ ×
Value	2
	🔽 HEX Value
(CLEAR OK

As you are familiar with the syntax of defining counters in TRACE32 just fill out the dialog.

	Value				Value	
DATA0:		V BYTE	-	COUNTO:	0x1234	V
DATA1:		V BYTE	T	COUNT1:		V

If there is no counter defined at the time you want to use a counter name in a condition or an action you will be asked to define it.

Value DATA0: DATA1:	V BYTE V BYTE	COUNTO:	Value	<u>v</u> v	
- Level0					
Action	Address/Range	RD/WR	Data	Count	
· · · · ·	IF 📃	& -	&	- &	•
		1. 7	2	↓ &	-
View Data	👛 Define Cou	Inter		↓ &	-
Goto Level1	Value	~		→ &	-
Decr COUNTO		I HEX	Value		
Decr COUNT1				- &	
Reload COUNTU Reload COUNT1		CLEAR		- &	-
	┉╵	& -	&	- & _	-
Louol2					

Example 1: Selective Trace on an Address Range



Click here to open the Define Address Window



🌯 ETM Pro	ogramming			
- Definition				
	Address	Address		
A-Range :	P:0x92bc0x9458	V or	V	
B-Range :		V or	V	
C-Range :		V or	V	
	Value		Value	
DATA0:	V BYTE	COUNTO:	V	
DATA1:	V BYTE	COUNT1:	V	
- Level0				
Action	Address/Range	RD/WR	Data Count	
Trace	▼IF A-Range ▼	& 🔹 &	▼ &	•
	- IF -	2 - 2	- 2	-

🔛 B::Analyz	er.List ALL		
🖉 Setup	📭 Goto 🏥 Find 🔷 More	X Less	
record	run address cycle	dil Isvmbol	Inaddress Insvi
		🔍 B::A.V -2486. /FT	
668	while (TRUE)	Coto Deve There	
669			
670	sieve();	HIGH HIGH HIGH HIGH HIGH	LOW LOW LOW HIGH LU
	DI UX93BC	A:001C0C0C R:000093BC \\a	rmlo\arm\ciouo
		tnl tnh ti fore	rigger ti ref
	char flags[SIZE+1]:	C8 FA	-0.0001828005
	ondr IIdge[bilb.I],	m.a m.b m.c m.d	0.000.020002
	int sieve()	1	
678	{		
	mov r12,r13		
00000470	stmdb r13!,{r4,r11-	r12,r14,pc}	
-00002476	F D:00000FD4 wr-long	UUBC614E	
-00002475	F D:00000FD8 WF-long		
-00002474	f D:00000FDC wr-long	00000720	
-00002473	f D:00000FE4 wr-long	00003340	
00002.112	sub r11.r12.#0x4		
	,,*		
	char flags[SIZE+1];		
670	int sieve()	/* sieve of erat	hostenes */
678	1 	udman las	
	register int 1, p	rimz, k;	

Example 2: Selective Trace on an Address Range defined through a Symbol

🕭 ETM Programming			
Definition Address A-Range : B-Range : C-Range :	Address or V or V or		
Value DATA0:		Value n- I	vl
Click here to open the Define Address Window	Office Address all values HEX Single Address	Mem.Class Value/S	ymbol
	C Range	Mem.Class Value/	Symbol Value/S
Click here to select the radio button for HLL-Range	• HLL-Range	Symbol	
Press the V button	<mark>≴ B∷syn</mark> *** symbol	hbol.browse * /c "dialo	g.set SE ''''*'''' /d t. Type:
Select the symbol by a double click –	_real_d remove rename _seterr setvbuf sieve signal _signal _signal _signal	efault_signa… (int _init _real_handler	()) R:00
Define Address			1
all values HEX Mem.Class Value/Syr Single Address	mbol	V	
C Range Mem.Class Value/S	ymbol V	alue/Symbol	1
Symbol HLL-Range \\armle\arm\sieve	V	CLEAR OK	

🏝 ETM Programming 📃 🗐 🗷
- Definition
Address Address
A-Range : v.range("\\armle\arm\sieve") V or V
B-Bange:
C-Bange: V or V
Value Value
DATA0: V BYTE COUNTO: V
DATA1: V BYTE COUNT1: V
Level0
A device Device DD &//D Device Count
Action Address/Hange HD/WH Data Count
📕 B::Analyzer.List ALL
A Setup 📭 Goto 🏥 Find 🗢 More 🛛 🗶 Less
record run address cycle d.l symbol paddro
668 while (TRUE)
669 { 670 {
bl 0x93BC ; sieve
char flags[SIZE+1];
(here the second s
678 {
mov r12,r13
-00002476 f D:00000FD4 wr-long 00BC614E
-00002475 f D:00000FD8 wr-long 00000FFC
-00002474 F D:00000FDC wr-long 00000FE8
-00002472 f D:00000FE4 wr-long 000093CC
sub r11,r12,#0x4
char flags_SIZE+1_;
int sieve() /* sieve of erathostenes */
678 {



& ETM Programming			
- Definition			
Address		Address	
A-Range:	V or		<u>v</u>
B-Range:			<u> </u>
U-Hange:]	<u> </u>
Value		Value	
DATAO:		COUNTO:	<u> </u>
Click here to open the Define Address Window	A Define Address		
	🔲 all values HEX		
	C	Mem.Class Value/Symbol	
	Single Address	7	
		Mem.Class Value/Symbol	Value/Sym
	C Range	7	V •
		Sumbol	_
	HLL-Range		►v
Click here to browse through			CLE/
the symbol data base			
	B::symbol.bi	owse * /c "dialog.set SE ""*	/d
	J***	tuno	Type:
	fclose	суре	R
Select the symbol by a	fflush _fflush		R
double click	Flags flags	(unsigned char	[19 D:0000FAC8-
	_flushlinebu	ffered	R
	Tmui		R
			-
Define Address			×
	Value/Sumbol		
C Single Address		V	
C Range	Value/Symbol	Value/Symbol	7
Symbol			
Million ange ///armle/arm/f	lags <u>V</u>		-
		CLEAR OK	

🕸 ETM Programming	
- Definition	
Address Address	
A-Range : v.range(flags) V or	V
B-Range : V or	V
C-Range : V or	V
Value Value	
	V
DATA1: V BYTE V COUNT1:	<u>v</u>
Action Address/Range RD/WR Data	
🖩 B::Analyzer.List ALL	
🌽 Setup 📭 Goto 👔 Find 🗢 More 🗶 Les	s
record run address cycle d.l	<u> symbol</u> \\armle\arm\flags+Ox
689	, , a,, a,, , , , , , , , , , ,
$\begin{array}{c} 690 \\ 691 \end{array} \qquad \qquad primz = i \\ k = i + p; \end{array}$	+ i + 3; rimz;
692 while (k	<= ŚIZE)
693 f	lags[k] = FALSE;
-00001213 f D:0000FACE wr-byte 00	\\armle\arm\flags+0>
696 }	+- bitms?

Example 4: Trace the Entrance and Exit of Function Sieve

🐣 ETM Prog	ramming							- - ×
- Definition								
	Address				Address			
A-Range :	sieve(si	eve+0x8)	V	or	(sieve+0)x98)(sieve+0xa0)	V	
B-Range :			V	or			V	
C-Range :			V	or			V	

Define the begin of function sieve

Define the end of function sieve

& ETM Prog	gramming	
– Definition –		
	Address	Address
A-Range :	sieve(sieve+0x8)	V or (sieve+0x98)-(sieve+0xa0)
B-Range :		<u>V</u> or <u>V</u>
C-Range :	1	
V	/alue	Value
DATA0:	V BYT	TE COUNTO: V
DATA1:	V BYT	TE 🔽 COUNT1: 🔽
— Level0 —		
Action	Address/Bange	BD/WR Data Count
Trace	IF A-Bange	
	TIF	
		📕 B::Analyzer.List ALL
		🌽 Setup 🔃 Goto 👔 Find 🗢
		char flags[SIZE+1];
		int sieve()
		678 {
🔍 B::A.V -6	6. /FT	mov r12,r13 Idmdb r11,{r4,r
🔒 Goto	🔺 Prev 🛛 🔻 Next 🔡 List	Timing
aa.30 aa.3	31 baddress faddress	fsymbol char flags[SIZE+1];
ps1 ps0	tpl tph ti.fore	trigger int sieve()
•		
🔍 B::A.V -46	6. /FT	
🔒 Goto	🔺 Prev 🔍 🔻 Next 🔛 List	Timing
HIGH LOW	A:001COCOC R:00009458	Tsymbol ts ps2 A \\armle\arm\sieve+0x9C HIGH HIGH
ps1 ps0	tpl tph ti.fore	trigger
كار		

Example 5: Trace the first 200H Cycles in Function Sieve

Value COUNTO: 0x200 COUNT1:	Define Counter Count0
ETM Programming	
- Definition	
Address	Address
A-Range : v.range(sieve)	V or V
B-Range : sieve	V or
C-Range :	V or
Value	Value
DATA0:	V BYTE V COUNTO: 0x200 V
DATA1:	
- Levelu	
Action Adu	ss/Hange RD/WR Data Count
	::Analyzer.List ALL
Lé lé	Setup [↓ Goto]}] Find
-*	*****

	stmdb r13!,{r4,r11-r12,r14,pc}
	000395 f D:00000FD4 wr-long 0000FFC
	000394 f D:00000FDC wr-long 00000FE8
-č	000392 f D:00000FE4 wr-long 000093CC
	sub r11,r12,#0x4
	char Ilags[512E+1];
	int sieve() /* sieve of er
	register int i, primz, k;
	int anzahl;
	682 anzahl = 0;
	mov r3,#UxU
	684 for ($i = 0$; $i \le SIZE$; flags[$i++$] = TRUE
	cmp r1,#0x0
	ble 0x93F8 b 0x93DC
	mov r4,#0x1
	add r1,r1,r4

Example 6: Trace all, when Function Sieve is reached goto Level 1 and stop Sampling and Debugging after 5 Cycles

🕭 ETM Pr	ogramming						E	
- Definition		Define th	e begin of fu	nction				
	Address	sieve in .	A-Range	Address				
A-Range :	sieve(sie	eve+0x4)	V	or		١	Define th	е
B-Range :			V	or		\	max valu	e 30
C-Range :			V	or		\	the cycle	ing s in
	Value				Value			5 111
ΠΔΤΔΩ							V	
DATA1:	<u> </u>						- -	
				Beload COL	UNT0 when s	sieve is re	eached	
– Level0 –			-		-		Jaonou	
Action			:/Range	RD/WR	Data		ount	
Reload C		<u> </u>		<u> </u>	&	``		
Goto Lev	ell	IF A-Ran		<u>⊢ – –</u>	&	`		
		"	~~~~~~~~~~~~~		а •	°		
_ Level1			<u> </u>	Change lev	∝ el if sieve is	reached		<u> </u>
Trace			- 2		2	- 2		
Decr COL			 	├ ── <u></u>	2			
Trigger II	BACE		 		&		COLINTO	
- Level2					" J			
	$\overline{}$		- &		&	- &		-
Stor		when COUN	T0 reaches 0	Subtrac	t one COUN	T0 every (vole	
B::Analyz	zer.List ALL			Cabilac			59010	
🌽 Setup	🔒 Goto 🍵	j Find 🔷 🖨 Mo	re 🛛 🗶 Less					
<u>record</u> -*******	run address	cycle	d.l symbo	סו	ŗ	add _		
_*****	stmdb	r13!,{r4,r11	-r12,r14,pc}	Sampling	starts here			
	char flags	SIZE+1];						
678	int sieve()			/* sieve of e	erathostenes */	· .		
	mov	r12,r13						
	1 300	111,112,#08.						
	char flags	SIZE+1];						
	int sieve()			/* sieve of e	erathostenes */	·		
678	t reg	pister int i,	primz, k;					
	int	anzahl;						
682	anz mov	ahl = 0; r3,#0x0						
684	for	(i=0;i	<= SIZE : flags	s[i++] = TRUB	E);			
	mov gmp	r1,#0x0 r1,#0x12	,9.	Somolin	a etone horo			
+*****	1 amp			Sampin	y stops here	- I		
J								