

# JTAGjet Emulators

## for Freescale ARM Devices

### Kinetis, i.MX and MAC7100 series



**JTAGjet™** is a small, universal In-Circuit Debugger that connects to targets via the JTAG port. It is equipped with USB 2.0 high-speed port and on-board FPGA to achieve download speeds of up to 2MBytes/sec.

**JTAGjet-Trace™** has the same features as JTAGjet but contains ETM and PTM real-time trace capture logic with up to 18 MBytes of memory.

#### Complete Freescale ARM Support

JTAGjet supports all **ARM7, ARM9, ARM11 and Cortex-M0/M1/M3/M4, Cortex-R4 and Cortex-A8/A9** based devices from all manufacturers. All Freescale ARM & Cortex based devices are supported.

- ❑ **Cortex-A8** i.MX50, i.MX51, i.MX53
- ❑ **Cortex-M4** Kinetis K10, K20, K30, K40, K60
- ❑ **ARM11** i.MX31, i.MX35, i.MX37
- ❑ **ARM9** i.MX1, i.MXL, i.MXS, i.MX2x series
- ❑ **ARM7** MAC7100 series

#### Chameleon Debugger™ Included

JTAGjet emulators come with **Chameleon Debugger** which allows debugging of **embedded Linux** boot codes, kernels, kernel drivers as well as other RTOS and non-RTOS based applications.

#### Multi-Core Debugging

Chameleon is designed to debug Multi-Core devices as well as Multi-CPU boards on the same debug screen with breakpoint, stop and go synchronization.

#### Compatible with All Major IDEs

- ❑ **ARM Ltd ADS & RV**
- ❑ **Eclipse CDT**
- ❑ **eSOL eBinder**
- ❑ **Freescale CodeWarrior**
- ❑ **GNU GDB**
- ❑ **GHS Multi**
- ❑ **IAR EWARM**
- ❑ **Keil RealView MDK-ARM**
- ❑ **Mentor Graphics EDGE**
- ❑ **Signum Chameleon included**

#### Flash Programmer

GUI **Flash Programmer** is included for all CFI compliant **NOR** flash devices and all **on-chip** internal flash in the Kinetis and MC7100 MCUs.

**Batch mode** flashing utility is available for **NAND, NOR, SPI** and on-chip flash devices.

#### JTAG Chain Device Detection

JTAGjet automatically detects all devices on the JTAG chain to properly configure the debugger. It also detects target power and target resets. That is why it is perfect for debugging the power-on/off and reset conditions, informing user about the state of target at all times.

#### Auto-sensing JTAG voltage

JTAGjet supports detachable target headers to accommodate various JTAG pinout standards and voltages between 1.6V and 5V.

#### Chameleon Debugger Features

- ❑ **Multi-Core, Multi-CPU and Multi-Emulator debugging**
- ❑ Compatible with all major **ARM C/C++ compilers**
- ❑ **Embedded Linux debugging** without the need for Ethernet or serial ports (console and TCP/IP over JTAG)
- ❑ Supports Embedded Trace Buffer (ETB)
- ❑ **Code Coverage Analyzer** with source tagging and reports
- ❑ **Code Profiler** to identify where the CPU spends most time
- ❑ Support for all **on-chip breakpoints**, triggers and filtering
- ❑ Automatic **processor initialization** on power-up or reset (memory mapping, peripheral setting, MMU, WD disable etc.)
- ❑ **Macros** for automated board initialization and testing
- ❑ Fly-over variable pop-ups in source window
- ❑ **Registers Window** with bit-fields and descriptions
- ❑ **Virtual-to-physical address mapping** support for ARM cores with MMU
- ❑ **Drag-and-drop** values between windows
- ❑ On-chip and off-chip **Flash Programmer** (GUI and batch mode)
- ❑ **Windows 7, Vista and XP compatible (32 & 64-bit versions)**

#### JTAGjet Features

- ❑ Supports all **ARM7, ARM9, ARM11 and Cortex-M/R/A devices**
- ❑ USB 2.0 (480Mb/s) port with on-board FPGA allows super fast code downloads to RAM at up to **2MBytes/sec**
- ❑ Programmable JTAG clock up to **30MHz**
- ❑ **Active JTAG** probes to support the **ARM-20, Cortex-20** and **Cortex-10** headers in JTAG and Serial Debug Modes (SWD)
- ❑ Support for JTAG voltages from **1.6V to 5V**

#### JTAGjet-Trace Features

- ❑ Up to **400 Ms/s** trace acquisition which supports the latest Cortex-MRA devices at up to **1.6GHz** CPU clocks
- ❑ Supports Embedded Trace Macrocell™ (ETM) and Program Trace Macrocell™ (PTM) ports which allow data variables and program flow tracing in real-time
- ❑ Trace clock timing **calibration** eliminates problems with trace data and clock skew
- ❑ Available with up to **18Mbytes** trace memory
- ❑ 56-bit time stamp with CPU cycle accuracy down to 5 ns
- ❑ Easy access to all ETM/PTM modes, triggers and trace filtering
- ❑ Small form factor - fits in the palm of your hand
- ❑ Quiet operation – no fans, no external heat sinks
- ❑ Only one connection to target – both JTAG and trace are taken from one trace connector



1211 Flynn Rd., #104, Camarillo, CA 93012  
**Phone:** (805) 383-3682  
**Fax:** (805) 383-3685  
**Web:** [www.signum.com](http://www.signum.com)

For more details see: [www.signum.com/freescale.htm](http://www.signum.com/freescale.htm)

# Chameleon Debugger Connected to Kinetis

## Screen Example

**Complex Events window setup (Data Value and ITM tracing)**

**Mixed mode disassembly with coverage markings**

**CPU status window with CPU registers, RunPC and cycle count**

**Trace window with data value tracing and function coloring**

**Peripheral registers window. Registers from WDOG group shown**

**Source window with code coverage markings**

**Kinetis:setup**

Comparator 0 Output: ITM Trace, Function: Emit Data on Data Address Match, Access: Read or Write, Value: &Buf1[0]

Comparator 1 Output: Disabled, Function: None, Access: Read or Write, Value: None

Comparator 2 Output: Disabled, Function: None, Access: Read or Write, Value: None

Comparator 3 Output: Disabled, Function: None, Access: Read or Write, Value: None

Vectors: ☐ Enable

ITM Events: ☒ Enable

☐ EXCTRC (exception tracing)

☐ 8-bit overflow counter events

☐ CPI (cycles per instruction)

☐ Scaled CPU cycle ev...

☒ CPU clock

☒ Emit PC Sample

**Kinetis:source.2 - mixed mode**

0x2000009C 7801 LDRB R1, [R0, #0x20000000]

0x2000009E 2900 CMP R1, #0

0x200000A0 D1F3 BNE

24: {

29: {

main:

0x200000A4 B510 PUSH {R4, LR}

32: GlobalStruct.Counter = 0;

0x200000A6 2000 MOV R0, #0

0x200000A8 491F LDR R1, [R1, #0]

**Kinetis:status**

STOP

APSR: 20000000

N ☐ Z ☐ C ☒ V ☐ Q ☐ CONTROL: 00000000

IPSR: 00000000

EPSR: 01000000

T ☒ ICI: 0

PRIMASK: ☐

FAULTMASK: ☐

BASEPRI: 00

R0: 20002000 R7: 00000000

R1: 00000053 R8: 00000000

R2: 00000001 R9: 00000000

R3: 00000001 R10: 00000000

R4: 00000003 R11: 00000000

R5: 00000000 R12: 00000000

R6: 00000000

RunPC: 2000009E Cyc: 074A3ED7

**Kinetis:registers.1**

Register	Value	Address	Attr
WDOG_PRESC	0x0400	0x40052016	RdWr
WDOG_REFRESH	0xB480	0x4005200C	RdWr
WDOG_RSTCNT	0x0000	0x40052014	RdWr
WDOG_STCTRLH	0x0000	0x40052000	RdWr
WDOG_STCTRLH	0x0001	0x40052002	RdWr
WDOG_TMROUTH	0x0000	0x40052010	RdWr
WDOG_TMROUTH	0x0000	0x40052012	RdWr
WDOG_TOVALH	0x004C	0x40052004	RdWr

**Kinetis:trace**

#	PC	Disas	Source
#2838/4	delay+0x1C	CMP R4, R0	
#2838/5	delay+0x1E	[ ]JBLT 0x20000148	
#2842	delay+0x20	ADDS R3, R3, #0x1	for (i = 0; i < cnt;
#2842/1	delay+0x22	CMP R3, R2	
#2842/2	delay+0x24	[ ]JBLT 0x20000144	
#2846	delay+0x26	MOV R0, R5	return k;
#2846/1	delay+0x28	{R4-R5, PC}	
#2852	main+0x34	LDR R0, [PC, #0x50]	processing(Buf1)
#2852/1	main+0x36	BL 0x20000088	
#2856	processing	B 0x2000009C	while (*text != '\0'
#2862	processing+0x14	LDRB R1, [R0, #0]	while (*text != '\0'
#2862/1	processing+0x16	CMP R1, #0	
#2862/2	processing+0x18	BNE 0x2000008A	
#2866	processing+0x2	LDRB R1, [R0, #0]	if (*text != '')
#2866/1	processing+0x4	CMP R1, #0x20	
#2866/2	processing+0x6	[ ]BEQ 0x2000009A	
#2870			
#2874			
#2878			
#2882			
#2888	processing+0x8	LDRSB R1, [R0, #+0]	*text ^= 0x20;
#2888/1	processing+0xC	EOR R1, R1, #0x20	
#2888/2	processing+0x10	STRB R1, [R0, #0]	
#2888/3	processing+0x12	ADDS R0, R0, #0x1	text++;

Status: NotActive

Last Sample #4194299 (10)

Trace Clock: 14.01MHz

**Kinetis:source.1 - source mode**

```

13:
14: struct MYSTRUCT GlobalStruct; /* struc
15:
16: void processing(char *text) /* function 'p
17: {
18:     while (*text != '\0') { /* do so
19:         if (*text != ' ') { /* is it
20:             *text ^= 0x20; /*
21:         }
22:         text++; /* advan
23:     }
24: }
25:
26: int delayloop = 3; /* small value (
27:

```

Buf1[0] Rd 0x73 Byte

Buf1[0] Rd 0x73 Byte

Buf1[0] Rd 0x73 Byte

Buf1[0] Wr 0x53 Byte