

PIETRO LOREFICE

Embedded Developer

Firmware debugging

Develer TPS

30/09/2020

develer

I “DEBUGGARE IL SOFTWARE E’ DIFFICILE”

“Debuggare il firmware
è peggio.”

— *Qualcuno*



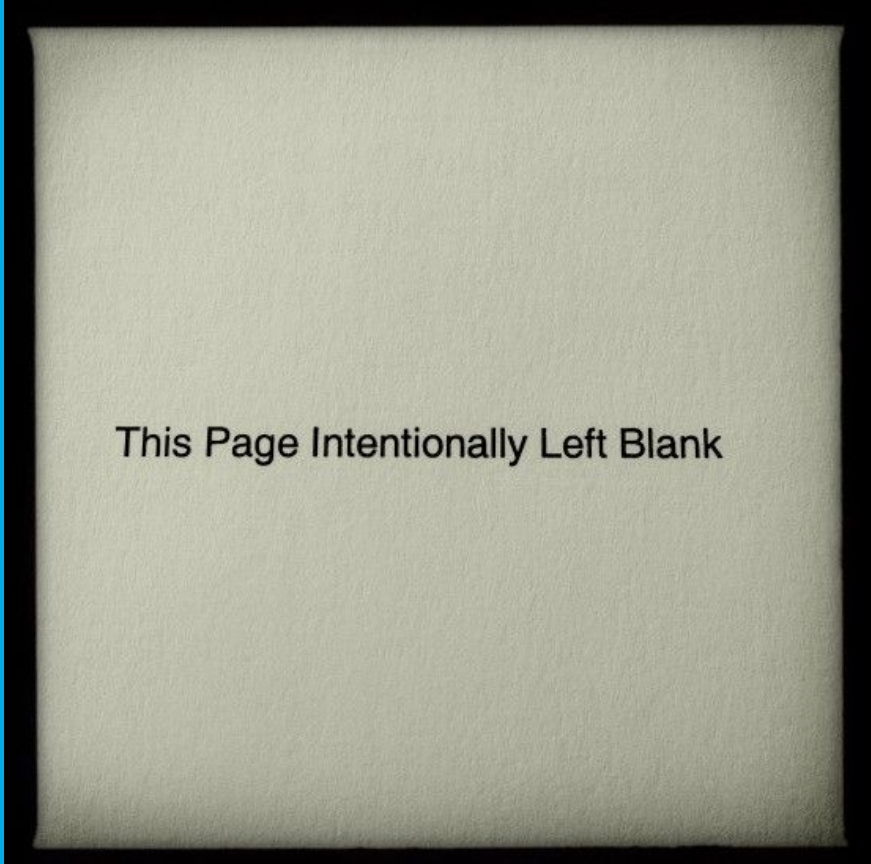
SCENE DI VITA VISSUTA

- | Clock che non clockano
- | Interrupt che non interrompono
- | Interrupt che interrompono troppo
- | Stack overflow silenti
- | Reset casuali

DEBUG DI UN APPLICATIVO DESKTOP

- | *printf()*
- | Rettangoli colorati
- | Debug passo-passo
- | Memory analyzers
- | Core dumps
- | Testing

DEBUG DI UN FIRMWARE

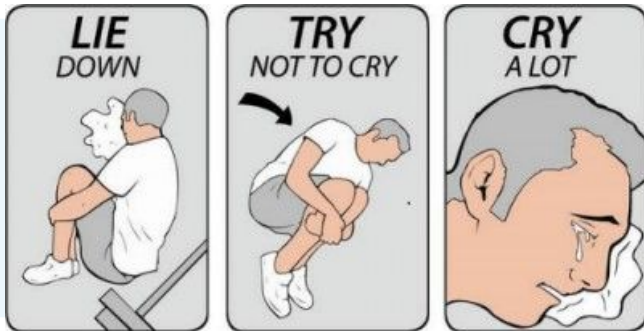


This Page Intentionally Left Blank

PROBLEMI NEL DEBUG DEL FIRMWARE

- I/O testuale limitato o assente
- Tempistiche hardware da rispettare
- No crash log o coredump
- Testing spesso trascurato

E QUINDI CHE SI FA?






- Svincolarsi dalla *printf()*
- Utilizzare debugger on-chip
- Sapere quando fare debug passo-passo
- Prediligere debugging "asincrono"

LIMITAZIONI PRINTF SU UART

- | Non utilizzabile in alcuni contesti (eg. ISR)
- | CPU-intensive e *bloated*
- | Legata alla disponibilità hardware
- | Tradizionalmente lenta

UNA POSSIBILE ALTERNATIVA: SWO

- *Serial Wire Output*
- Presente su ARM Cortex-M
-  Veloce (fino a 100 Mbps) e asincrono rispetto al core
-  32 canali su un solo filo
-  Richiede HW e SW dedicati

DEBUG PASSO-PASSO

- | Utile nel debug della business logic
- | Difficile da usare su problemi legati all'I/O
 - Un core stoppato non serve gli interrupt
 - L'hardware spesso non gradisce

UNA POSSIBILE ALTERNATIVA: TRACING

- | Supportato da varie architetture/MCU
- | Registrazione delle ultime N istruzioni eseguite
- | Riproduzione offline del trace
- | Ideale per debuggare eccezioni, fault e reset

TRACING SU ARM: ETM



Non impegna la CPU



Supporta diversi trigger

- Breakpoint
- Watchpoint
- Interrupt
- Reset
- Trigger esterni



Richiede HW e SW dedicati



Tutto bello, ma come
uso questa roba?





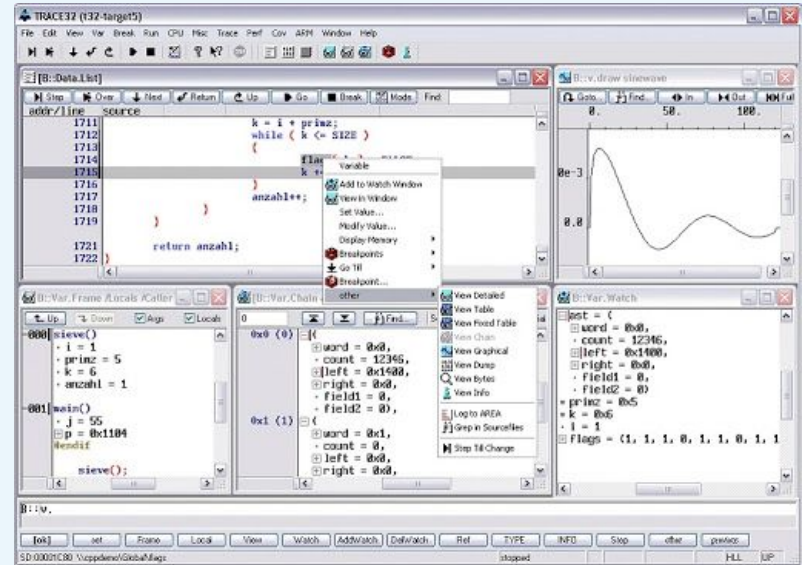
I **Lauterbach PowerDebug USB 3**

- 80+ architetture supportate
- Interfaccia JTAG ed SWD
- Programmazione e debug
- Bare-metal e OS-aware
- Debug userspace e kernel
- Supporto per multicore
- Costo complessivo: svariati euri



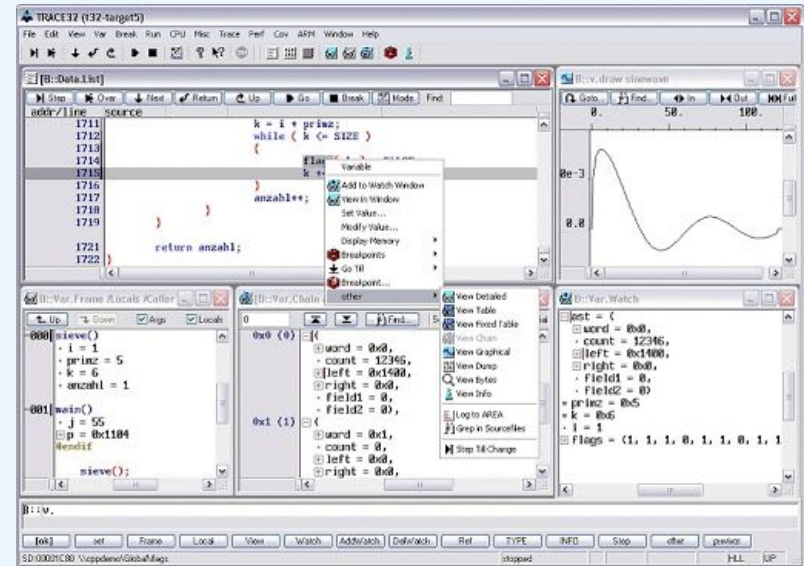
Lauterbach TRACE32

- Interfaccia verso PowerProbe
- Unico per tutte le architetture
- Completamente scriptabile
- Multiplatforma
- UI moderna e user-friendly



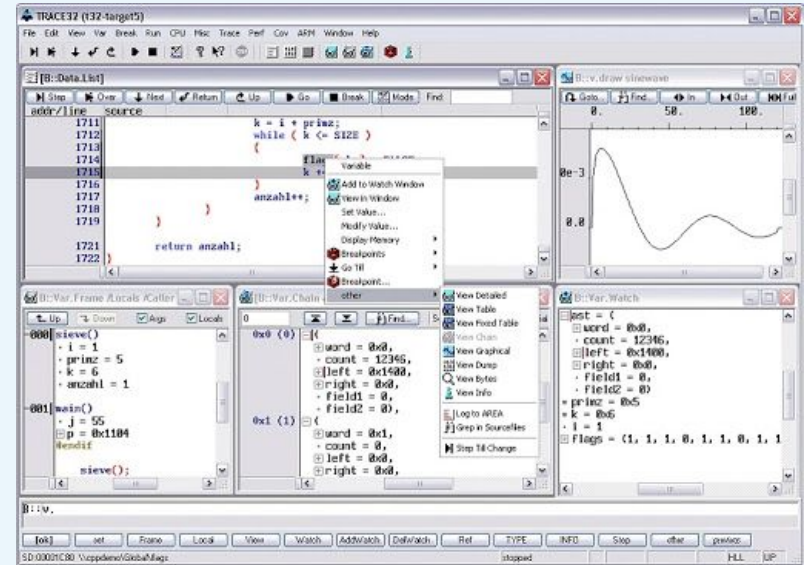
Lauterbach TRACE32 – Funzionalità bare-metal

- Debug passo-passo
- Watchpoint su variabili o registri
- Accesso registri CPU e MMIO
- Breakpoint su reset o fault
- Plot di variabili
- Trace visuale



Lauterbach TRACE32 – Funzionalità OS-aware

- Supporto MMU
- Debug processi e kernelspace
- Analisi risorse del kernel
- Loading e debug di moduli



File Edit View Var Break Run CPU Misc Trace Perf Cov STM32FxX Window Help

B: Register

```

N  R0 00030045 R0 0 SPA 00000001
Z  R1 06030045 R0 0 SPA 00000001
C  R2 0F000000 R10 0 +04 2401F380
V  R3 0F R11 0 +00 24002788
0  R4 58024400 R12 20000000 +0C 24019554
RS 40028000 R13 2401F370 +10 24002738
0  R6 SF R14 08003169 +14 24002780
1  R7 58000400 PC 08002904 +10 240027CC
2  CONTROL XPSR 210F0000 +1C 0804680F
3  CONTROL PSP 2401E500 +20 08005091
4  FAULTMASK 0 PSP 2401F370 +24 E1A29001
BASEPRI 0 +28 24002718
PRDMASK 0 +30 24002780
+32 24002780
+34 24002738
+38 24002780
+3C 240027CC
+40 08005000
+44 24002788

```

B: Trace List

record	run	address	cycle	data	symbol
62		lsll r4, r4, r2			
		str r2, [r13, #0x1C]			r2, gpio_mode, r1
		lsl r2, r9, r1			
		lsl r8, r2, #0x2			
		str r8, [r13, #0x20]			
		str r2, [r13, #0x18]			
		movs r9, #0x3			
		ldr r2, [r13, #0x4]			
		and r8, r9, #0x1C			
		lsll r5, r4, r1			
		tst r9, #0x1			
		lsl r2, r2, r1			
		mov r9, #0x0F			
		str r2, [r13, #0x24]			
		lsl r9, r9, r8			
		ldr r2, [r13, #0x0C]			
		str r9, [r13, #0x2C]			
		lsl r2, r2, r1			
		ldr r1, [r13]			
		str r2, [r13, #0x28]			
		mov r2, #0x0F			
		lsl r5, r2, r1			
		ldr r1, [r13, #0x8]			
		str r2, [r13, #0x14]			
		asr r1, r1, r3			
		beq 0x802A08E			
		lsll r1, r1, #0x1E			
		bpl 0x802A154			
		T:0802A154 ptrace			_tm32stm32_gpioInConfig+0x100
		for (int i = 0; i < 16; i++)			
		cpu_flags_t flags;			
		ASSERT(irq < NUM_INTERRUPTS);			
		IRQ_SAVE_DISABLE(flags);			
		irq_table[irq + 16] = unhandled_isr;			
		IRQ_RESTORE(flags);			
		void sysirq_init(void)			
		cpu_flags_t flags;			
		int i;			
		IRQ_SAVE_DISABLE(flags);			
		for (i = 0; i < NUM_INTERRUPTS; i++)			
		irq_table[i] = unhandled_isr;			
		/* configure priority grouping to 4 bits for preemption priority and 0 bit			
		sysirq_setPriorityGrouping(NVIC_PRIORITYGROUP_4);			
		/* custom hard fault ISR to print stack trace */			
		sysirq_setHandler(hardFault_ISR, (sysirq_handler_t) hard_fault_isr);			
		/* Update NVIC to point to the new vector table */			
		SCB->VTOR = (uint32_t)irq_table;			
		IRQ_RESTORE(flags);			
		154			

B: Bper, "Core Registers (Cortex-M7), Nested Vectored Interrupt Controller"

DMAMUX2_CICR	00000000	SYNCR_ID	DMAMUX2_evt0	SPOL	No event	SE	Disabled	EGE	Disabled
DMAMUX2_CICR	00000000	SYNCR_ID	DMAMUX2_evt0	SPOL	No event	SE	Disabled	EGE	Disabled
DMAMUX2_CICR	00000000	SYNCR_ID	DMAMUX2_evt0	SPOL	No event	SE	Disabled	EGE	Disabled
DMAMUX2_CICR	00000000	SYNCR_ID	DMAMUX2_evt0	SPOL	No event	SE	Disabled	EGE	Disabled
DMAMUX2_CICR	00000000	SYNCR_ID	DMAMUX2_evt0	SPOL	No event	SE	Disabled	EGE	Disabled
DMAMUX2_CICR	00000000	SYNCR_ID	DMAMUX2_evt0	SPOL	No event	SE	Disabled	EGE	Disabled
DMAMUX2_CICR	00000000	SYNCR_ID	DMAMUX2_evt0	SPOL	No event	SE	Disabled	EGE	Disabled
DMAMUX2_CICR	00000000	SYNCR_ID	DMAMUX2_evt0	SPOL	No event	SE	Disabled	EGE	Disabled

B: Trace.Chart.Symbol /Track

B: B: Frame /Locals/Caller

```

0000 sysirq_setHandler
  irq = 61
  handler 0x80030045
0001 eth_init()
  GPIO_MODE_AF_PP | GPIO_AF_11, GPIO_SPEED_VERY_HIGH;
244 /* Configure interrupt handler */
  sysirq_setHandler(ETH_IRQn, stm32_eth_irqHandler);
0002 system_init()
  mac = (addr + (96, 39, 0, 36, 204, 39))
...
/* Network interface global variables */
static struct ip4_addr_ipaddr, netmask, gw;
static struct netif netif;

```

components trace Data Var List PERF System Step Go Break Symbol Frame Register FPU MMX MMU TRANSLation CACHE CORE APU other previous

ST08029DC4 \sls\irq_cm7\sysirq_setHandler stopped at breakpoint HLL UP

“BREVI” STORIE TRISTI

- | Inizializzazione dei clock
 - Mancava il quarzo 😊
- | Crash su context switch
 - Errore nel settaggio delle priorità
- | Crash su alcuni IRQ
 - Bug su configurazione D-Cache e MPU

Pietro Lorefice

pietro@develer.com

Vuoi rimanere aggiornato sugli eventi Develer?
Seguici nei nostri canali social:



develer

www.develer.com