计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.191000048

HATBED:分布式硬件辅助追踪物联网测试平台

马峻岩　李　易　李尚荣　张　特　张　颖
长安大学信息工程学院　西安 710064

摘　要　无线传感器网络等物联网系统通常具有资源高度受限且与物理世界耦合的特性,这使得该类系统部署后的设备通常难以调试。因此,在部署前对整个系统进行充分的测试与评价显得尤为重要。传统基于串口的网络化测试手段具有较大的侵入性,且无法对资源受限设备的运行情况进行详细的跟踪。文中对硬件辅助追踪技术在物联网测试评价中的应用进行了研究,设计并实现了一种硬件辅助追踪测试平台——HATBED(Hardware Assisted Tracing Testbed)。HATBED由控制器、观察者以及被测目标组成,可以提供网络级远程调试、灵活的软件追踪以及非侵入式软件分析等主要功能,并在不依赖操作系统与应用的前提下,实现对系统的非侵入式追踪与分析。实验中,首先使用裸机与FreeRTOS操作系统下的标准例程,从功耗开销、时间精度以及代码覆盖率3个方面对HATBED进行了基准性能的测试;然后,以物联网RIOT-OS代码库程序为例,通过HATBED实现了ping6的高精度时间特性分析,及其底层gnrc协议栈UDP通信过程函数覆盖与基本块覆盖的评估。实例分析表明,借助硬件辅助追踪技术,HATBED可以对资源受限的物联网系统开展更加高效、充分的测试与评价。

关键词:物联网;硬件辅助追踪;测试平台
中图法分类号　TP393.05

HATBED:A Distributed Hardware Assisted Tracing Testbed for IoT

MA Jun-yan,LI Yi,LI Shang-rong,ZHANG Te and ZHANG Ying
School of Information Engineering,Chang'an University,Xi'an 710064,China

Abstract　Internet of Things systems,such as wireless sensor networks,usually have the characteristics of the high restriction of the resources and coupling with the physical world,which makes it difficult to debug the equipment after deployed. Therefore,it is especially important to thoroughly test and profile the systems before deploying to the real world. Due to the intrusiveness,traditional debugging methods based on the serial port are incompetent for detailed tracing on resource-constrained devices. This paper studies the application of hardware assisted tracing technology in the embedded network sensor systems' test and evaluation. Then,it designs and realizes a Hardware Assisted Tracing testBed (HATBED). HATBED consists of a controller,observers,and targets. It can provide three services,network-wide remote debugging,flexible software tracing and non-invasive software profiling. HATBED can support non-intrusive tracing and profiling without relying on operating systems and applications. In the experiment,this paper benchmarks time and power consumption,time accuracy,and code coverage under bare-metal and FreeRTOS. Then,it tests the RIOT-OS examples and completes the ping6 command high time accuracy feature profiling and UDP communication function coverage and basic block coverage. With the help of hardware assisted tracing technology,HATBED can evaluate the resource-constrained Internet of Things systems more efficiently and adequately.

Keywords　IoT,Hardware assisted tracing,Testbed

1　引言

无线传感器网络等物联网系统通常具有资源受限、应用程序实时运行的特点。因此,在调试过程中不应该停止应用程序的运行,调试所带来的入侵性也应该被降至最低。以JTAG调试为代表的传统方式由于频繁在挂起与恢复间切换,因此无法满足调试需求,而且仿真测试也主要针对网络协议与拓扑结构[1]。对于这类系统,调试过程通常包括系统运行轨迹记录与其事后分析,因此,硬件辅助追踪技术是最合适的手段,它拥有低入侵性、高速以及不需要程序插桩等优点。

到稿日期:2019-10-11　返修日期:2020-01-09　本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
基金项目:国家重点研发计划(2018YFB1600800);国家自然科学基金(61402050);创新引智基地(B14043)
This work was supported by the National Key R&D Program of China (2018YFB1600800),National Natural Science Foundation of China (61402050) and 111 Project (B14043).
通信作者:马峻岩(majy@chd.edu.cn)

2 相关工作

以 Indriya[2]，FIT IoT-Lab[3]，MoteLab[4]、X-Sensor[5]，TWIST[6]，WiLAB[7]，D-Cube[8]，SensLAB[9] 及 EasiTest[10] 为代表的大量测试平台关注目标节点的能耗与链路质量。也有一些测试平台提出了针对无线传感器网络调试与分析的追踪手段，使用非打扰式无线传感器网络测试仪来监听 CPU 与 RF 芯片间的通信[11]。AVEKSHA[12] 使用定制调试板支持 8 种触发器并借助程序计数器(Program Counter，PC)轮循进行应用程序分析。Flocklab[13] 与其改进版[14] 使用 GPIO Tracing 记录目标节点的时间敏感事件与行为。Minerva[15] 采用 JTAG 端口执行内存追踪与全局断言。这些方法需要高成本且运行时高开销的定制化硬件[11-13]，而且追踪能力有限[13]。此外，还有许多功能强大的新式测试平台，但其使用场景受限[16-18]，或主要针对某种特定系统[19-20]。硬件辅助追踪技术能够在不影响性能的前提下，追踪微控制器(Microcontroller Unit，MCU)的执行过程与事件，它可以告知开发人员程序运行的一些特定位置，并记录程序运行轨迹。此外，硬件辅助追踪技术不依赖于操作系统和应用程序。现代 MCU 通常集成有强大的专用调试模块，例如 ARM Cortex-M3/M4 内部就集成了指令追踪宏单元(Instrumentation Trace Macrocell，ITM)、数据观察点与追踪(Data Watchpoint and Trace，DWT)以及嵌入式追踪宏单元(Embedded Trace Macrocell，ETM)，如表 1 所列。许多面向物联网应用的 MCU 都配备有以上模块。

表 1 配备 ITM&DWT 和 ETM 的 MCUs

Table 1 MCUs with ITM&DWT and ETM

MCU	Architecture	ITM	ETM	Year
STM32F103	Cortex-M3	√	√	2012
CC2538	Cortex-M3	√	×	2015
EZR32WG	Cortex-M4	√	√	2016
SAM4L	Cortex-M4	√	√	2017
CC2652	Cortex-M4	√	×	2018

3 HATBED 功能设计

借助现代微控制器的调试支持，HATBED 提供以下功能：

1)网络化远程调试。通过远程串行线调试(Serial Wire Debug，SWD)端口，HATBED 的控制器可连接到被测目标调试单元，非侵入地追踪节点状态以及内存单元变化情况，从而对被测节点构成的物联网系统进行全局断点调试，网络化分布式断言。此外，HATBED 还可以通过脚本实现批量测试参数调优自动化。

2)灵活的软件追踪功能。ITM 提供了 32 个端口，能够进行 FIFO 缓冲调试消息输出，在使用方式上它与 printf 十分类似，但与基于 UART 的 printf 输出调试消息相比，其尽管不支持中断模式，但大幅缩短了时间延迟。这是由于缓冲区替代了唯一的 UART 寄存器，而且调试端口的数据输出速率远高于串口波特率。在微控制器运行期间，相关寄存器可以启用或禁用端口，因此该特性可用于时间敏感场景。

3)非侵入式高精度软件分析。ITM 和 DWT 支持硬件数据地址与指令地址观察点，可用来捕获数据/外设变化，并在控制流中标记出来，同时观察点也是沟通追踪数据内外部时钟的桥梁，实现软件高精度时间特性分析的保障。ITM 和 DWT 支持事件追踪、PC 采样和多种性能计数器，这些特性可用于程序的覆盖率分析、轨迹分析与状态统计。在代码的某些关键区域，通过启用 ETM，硬件追踪单元能够以消耗 MCU 时间资源为代价捕获程序的所有指令，尤其是跳转指令，完成细粒度分析。

4 HATBED 结构的设计

HATBED 由被测目标、观察者和控制器组成。被测目标是物联网系统中的节点；分布式观察者监测这些目标，目标与观察者间相互通信；控制器是一台高性能计算机服务器，能够对整个测试平台进行自动控制。图 1 给出了 HATBED 的结构，图 2 给出了 HATBED 观察者实物。

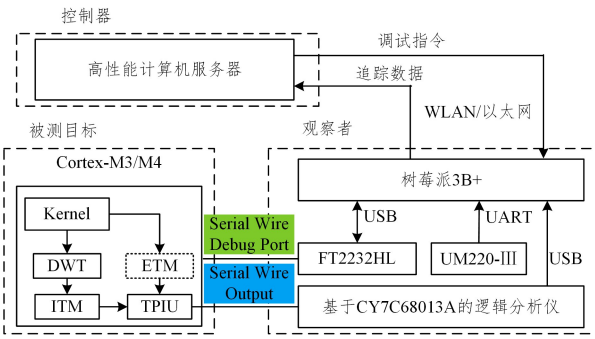

图 1 HATBED 的结构

Fig. 1 Architecture of HATBED

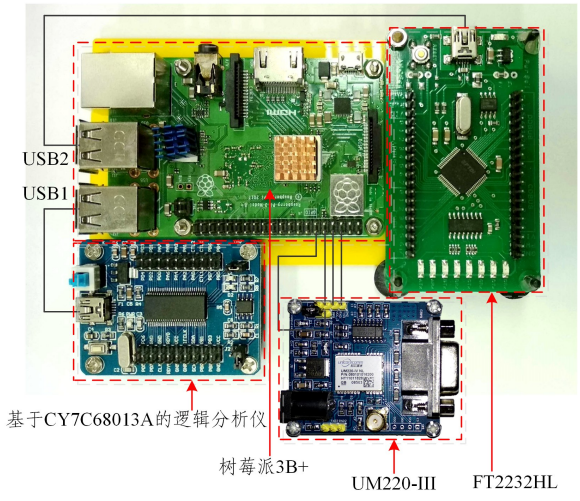

图 2 HATBED 观察者

Fig. 2 HATBED observer

4.1 被测目标

被测目标为具有 ARM Cortex M3/M4 内核的 MCU。并非所有配备该内核的 MCU 都具有 ETM 单元，但它们几乎都具有 ITM 与 DWT 单元，且具备 SWD 与 SWO 接口。

4.2 观察者

一个 HATBED 观察者由树莓派 3B+、FT2232HL、UM220-III(仅时钟源观察者)以及基于 CY7C68013A 的逻辑分析仪组成。通过 FT2232HL 调试板将树莓派 3B+与被测目标连接,并且在树莓派 3B+上运行 OpenOCD(Open On-Chip Debugger)与 GDB(GNU Project debugger)对被测目标进行调试。树莓派 3B+使用逻辑分析仪捕获被测目标输出的原始追踪数据,并将追踪数据保存到本地,随后将其上传给控制器。CY7C68013A 使用开源 sigrok fx2lafw 固件。所有观察者通过 WLAN/以太网连接到控制器。HATBED 的一个观察者作为时钟源,装备 GPS 模块 UM220-III,除了完成观察者功能外,还借助 GPS 卫星时钟数据与 PPS 信号获得准确时钟,局域网内所有观察者通过 NTP 协议与此时钟源进行时钟对齐。

除此之外,HATBED 代码中添加了一个周期性 GPIO 翻转,1/2 个周期内置 0,另 1/2 个周期内置 1,并使用 C 语言编写运行于树莓派 3B+上的标记工具来捕获 GPIO 翻转并标记时间戳,如图 3 所示。借助该 GPIO 翻转可为桥梁同步内外时间戳,具体的实施过程请见第 5 节。

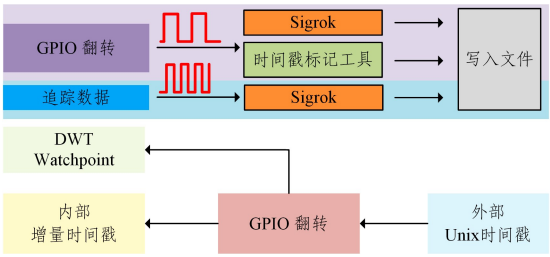

图 3 内外时间戳同步

Fig. 3 Time synchronization between inside and outside

4.3 控制器

高性能计算机服务器作为控制器控制整个测试平台。服务器将追踪配置代码添加到被测工程,编译完成后将其分发给各观察者。控制器执行控制脚本,控制观察者进行停止 CPU、恢复 CPU、烧写程序、轮循指定区域的内存、接收追踪原始数据与 Unix 时间戳数据等操作。控制器还运行 Sigrok 软件,使用 TPIU,DWT,ITM 以及 ETM 解码器对观察者采集的原始数据解码,提取出相关追踪数据,并运行分析脚本,将 Unix 时间戳数据与追踪数据关联。

5 HATBED 基准测试

本节针对 HATBED 的各项性能展开基准测试,以验证 HATBED。

5.1 代码覆盖率、时间开销和功耗评估

实验采用 STM32F103VET6 (Cortex-M3 内核)开发板,选择基于裸机的排序和 FreeRTOS 程序进行测试。为降低代码的不确定性,由 STM32CubeMX 软件生成标准代码。对于 FreeRTOS 用例,两个任务以时间片方式轮转。实验分别从代码覆盖率、时间开销和功耗方面对 HATBED 进行评估。评估过程中,HATBED 观察者捕获被测目标通过 SWO 端口输出的程序运行轨迹和 GPIO 翻转信号。

1)代码覆盖率。代码覆盖系统收集正在运行程序的信息,然后将这些信息与源代码结合,生成关于测试目标的代码覆盖报告[21]。硬件辅助追踪可以完成类似工作,同时不需要插桩手段。若用 ETM 进行全指令捕获,则可准确记录多线程应用与复杂算法的运行过程。

Sigrok 可以将捕获的 PC 与 elf 格式文件中的指令地址信息相结合,以确定 C 程序源代码中有关函数和代码行的执行情况。

如表 2 所列,我们对 10 个函数中的 6 个函数进行了全局变量(GlobalCounter)编码。DWT 观察点将监测这个全局变量,其他函数则由 ITM 的 PC 采样捕获。最后,通过 PC 和 DWT 观察点捕获了 10 个函数中的 9 个函数,得到函数覆盖率为 90%。在 FreeRTOS 中添加了两个排序函数,并开启 ETM 追踪,最终 ETM 捕获了 34 行中的 20 行语句和所有函数,因此语句和函数覆盖率分别为 60%和 100%。

表 2 编码与捕获结果

Table 2 Encode and captured results

MCU	GlobalCounter	Captured
任务 0	0xA	√
任务 1	0xB	√
ITM_Print	0xC	√
Bubble_Sort	0xD	√
FFT	0xE	√
IFFT	0xF	√
HAL_GPIOWrite_Pin	null	×
osDelay	null	√
Binary_InsertSort	null	√
Find_InsertIndex	null	√

2)时间开销。在测试中,将追踪信号频率设置为 8 MHz,UART 波特率设置为 1.152 MHz。为捕获所有分支,设置 ETM 可以在缓冲区溢出时停止 MCU。我们在程序中插入两个 GPIO 翻转,GPIO 翻转仅耗时 100 ns,可忽略。使用示波器抓取 GPIO 翻转来获得执行周期,打开 ITM 后,任务周期没有增加。启动 ITM 和 ETM 后,任务 0 的执行周期增加了 998 us,任务 1 则不变。但从波形上看,2 个任务的执行时间明显增加。ITM 没有时间开销,当启用 ITM 和 ETM 时,其执行时间将增加 91 us。增加的原因是 ETM 通过减慢 MCU 速度来防止 FIFO 缓冲区溢出,这表明 ETM 不适合一直处于开启状态。图 4 为裸机与 FreeRTOS 的任务周期波形图。

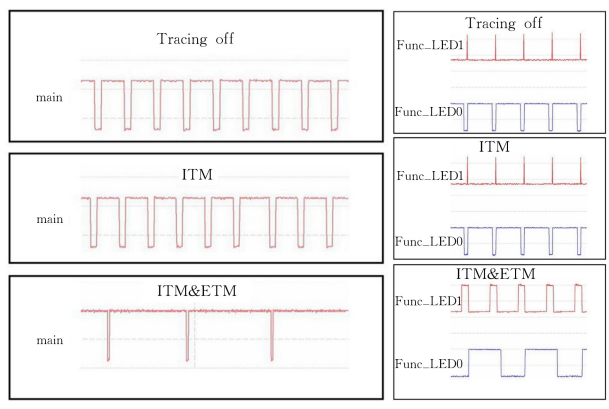

图 4 裸机(左)与 FreeRTOS(右)的任务周期波形图

Fig. 4 Task cycle waveform diagram of bare-metal (left) and FreeRTOS (right)

本文使用相同的方法来测量 ITM_Print 发送 38 个字节需要花费的时间。实验结果为：ITM_Print 仅占用 86 us，占整个周期的 8.6%；HAL_UART_Transmit（直接寄存器操作）占用 1500 us；UART printf（微库）占用 3650 us。进一步地，ITM_Print 发送 4 个字节只占用 2.5 us，占整个周期的 0.25%。

3）功耗。我们使用 Tektronix PA1000 功率分析仪作为测量工具。当目标稳定后，我们进行多次测量并取所有结果的平均值，实验结果如表 3 与表 4 所列。仅开启 ITM 后，FreeRTOS 功耗增加 6.4 mW，较未开启时增加了约 1.54%；裸机功耗增加 6.2 mW，较未开启时增加了约 1.53%。当 ITM 和 ETM 均开启时，FreeRTOS 降低了 24.4 mW，而裸机则降低了 13 mW。这是因为 ETM 的启用导致 MCU 减速。

表 3 FreeRTOS 的功耗测试

Table 3 Power consumption of FreeRTOS

Name	Voltage/V	Current /mA	Power /mA
Tracing OFF	4.841	85.79	415.3
ITM	4.840	87.15	427.1
ITM&ETM	4.845	82.06	397.3

表 4 裸机下的功耗测试

Table 4 Power consumption of Bare-metal

Name	Voltage /V	Current /mA	Power /mA
Tracing OFF	4.841	85.79	415.3
ITM	4.840	87.15	427.1
ITM&ETM	4.845	82.06	397.3

5.2 时间精度评估

在许多时间敏感的应用中，时间精度是最重要的因素。Flocklab 在 Open Embedded Linux 下的观察者中捕获时间戳，我们通过 GPS 和 NTP 协议结合的方式，完成所有观察者的时间同步。

1）多个观察者间的同步误差测试。实验选定 chrony 软件与 UM220-III 北斗/GPS 导航模块搭建 NTP 系统，选定 1 个树莓派 3B+作为 NTP 服务器，2 个树莓派 3B 作为观察者，1 个 STM32F103RCT6 开发板作为被测目标。被测目标执行标准邮箱队列工程代码并在其中添加周期为 1 ms 的 GPIO 翻转代码，设定两个观察者在同一时刻运行 GPIO 采集程序来采集被测目标的 GPIO 翻转，并使用 sys/time.h 中的 gettimeofday 函数标记 Unix 时间戳，最终结果如表 5 所列。

表 5 GPIO 时间戳标记结果

Table 5 Captured GPIO timestamps

Num	Unix timestamp Pi #1	Unix timestamp Pi #2
1	1 555 481 576.001 333	1 555 481 576.001 337
2	1 555 481 576.002 325	1 555 481 576.002 328
3	1 555 481 576.003 324	1 555 481 576.003 327
4	1 555 481 576.004 324	1 555 481 576.004 327

树莓派 1 与树莓派 2 标记的 Unix 时间戳的误差约为 4us，能够有效证明 NTP 时钟同步，以及树莓派进行 GPIO 时间戳标记方案的有效性。

2）内外时间戳关联。实验选定 1 个树莓派 3B 作为观察者，1 个 STM32F103RCT6 开发板作为被测目标，1 个 CY7C68013A 进行追踪数据采集，且 CY7C68013A 通过 USB 接口与观察者连接，被测目标执行标准邮箱队列工程代码，并在其中添加周期为 1 ms 的 GPIO 翻转代码。实验流程如图 5 所示，最终部分结果如图 6 所示。

实验结果表明，HATBED 能够为 GPIO 翻转的 DWT 事件标记 Unix 时间戳，这就为原本没有标准时间度量的 SWO 数据带来了标准化的时间度量。

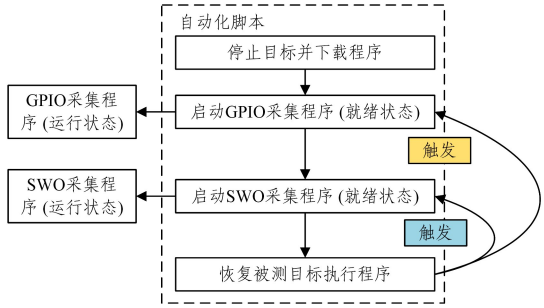

图 5 自动化脚本运行流程

Fig. 5 Workflow of automated script

```
PC:0x08002060
Timestamp:63899 (exact)
Thread
Enter:IRQ 30
Timestamp:64258 (exact)
Watchpoint 0:address 0x???? 1410 --c timestamp:2019/3/17 21:0:38 --
1552827637.460484
Watchpoint 0:Write data 0x00000004
Overflow
IRQ 30
Enter:SysTick
Overflow
```

图 6 Unix 时间戳匹配

Fig. 6 Unix timestamp match

3）最小时间精度。实验选定 2 个树莓派 3B 作为观察者，2 个 STM32F103RCT6 开发板作为被测目标，1 个 CY7C68013A 进行追踪数据采集，且被测目标 1 执行 FreeRTOS 标准定时器中断例程，触发周期为 5 ms，并在回调函数内翻转一个 GPIO 并将其作为模拟唤醒信号，程序又以 10 ms 为周期翻转另一个 GPIO 并将其作为关联内外时间戳；被测目标 2 执行 FreeRTOS 标准外部中断例程，选定一个 GPIO 并将其以外部中断模式接收被测目标 1 的唤醒信号，中断服务函数内翻转一个 GPIO 作为响应信号以表示接收到唤醒信号，程序又以 10 ms 为周期翻转另一 GPIO 并将其作为关联内外时间戳。最终逻辑分析仪捕获的结果如图 7 所示，D1 为唤醒信号，D4 为响应信号。可以看出，包含 GPIO 翻转在内，唤醒-响应过程共持续了 6.928 us；唤醒信号由被测目标 1 的 IRQ28 产生，内部时间戳 T_{IRQ28} 为 924 200，响应信号由被测目标 2 的 IRQ40 产生，内部时间戳 T_{IRQ40} 为 924 829；HATBED 另外捕获到被测目标 1 最相邻的定期 GPIO 内部时间戳 T_{GPIO1}

为 951 028，对应的外部时间戳为 1 563 100 718.750 442，被测目标 2 最近的定期 GPIO 内部时间戳 T_{GPIO2} 为 1001053，对应的外部时间戳为 1 563 100 718.753 536。计算可得，Enter: IRQ28 的外部时间戳为 1 563 100 718.750 442 − (T_{IRQ28} − T_{GPIO1})/16M = 1 563 100 718.748 765 25，Enter: IRQ40 的外部时间戳为 1 563 100 718.753 536 − (T_{IRQ40} − T_{GPIO2})/16M = 1 563 100 718.748 772，最终得到二者的差值为 6.75 us，减去 GPIO 翻转时间（约 0.1 us）后其与逻辑分析仪捕获的唤醒-响应过程的持续时间基本一致，这说明 HATBED 捕获事件的最小时间精度为微秒级。

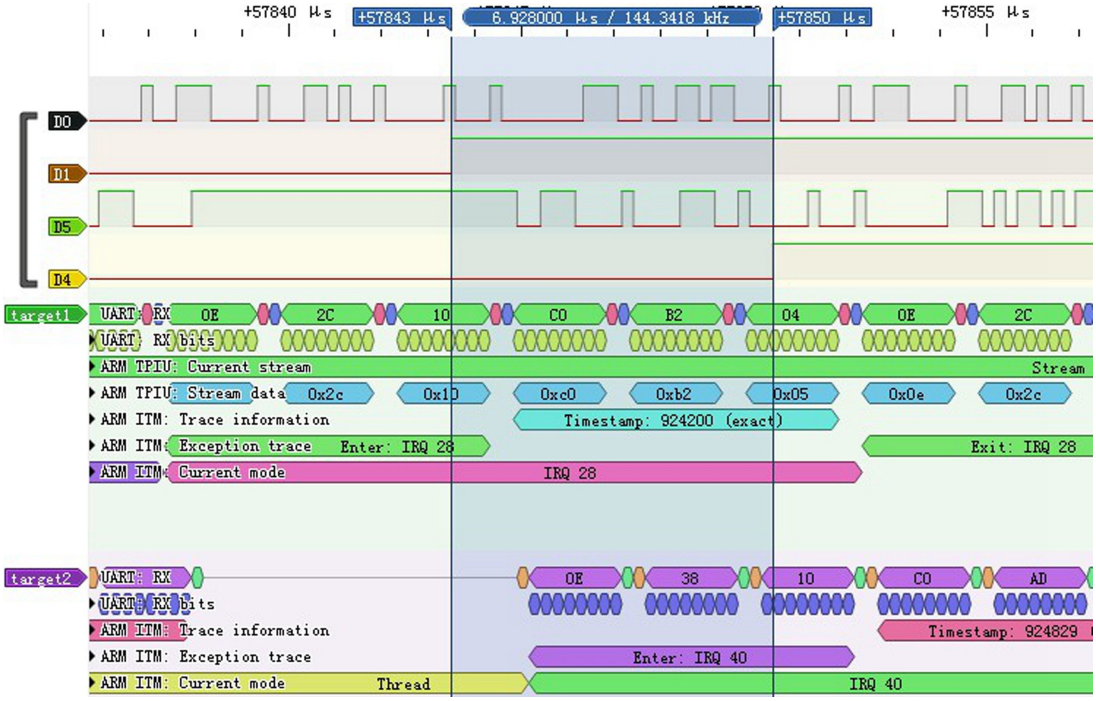

图 7 模拟唤醒-响应过程

Fig. 7 Simulated wake-up and response

6 HATBED 测试实例分析

本节使用 HATBED 对 RIOT-OS 的 gnrc_networking 例程进行了测试，实例 1 是对 ping6 指令进行分析，实例 2 是使用 HATBED 对 gnrc 协议栈中的 udp 通信进行函数覆盖与基本块覆盖测试。被测目标为 2 个配备 CC1101 的 STM32F103RCT6(Cortex-M3 内核)开发板，追踪时钟为 8 MHz。

6.1 实例 1 ping6 指令高时间精度分析

节点 1 使用 ping6 指令测试与节点 2 的连通性，成功时节点 1 的串口 shell 将打印出成功信息，如表 6 所列。

表 6 ping6 执行成功

Table 6 ping6 successful

COM30 # USB-SERIAL CH340
12 bytes from fe80::ff:39:icmp_seq=0 ttl=64 rssi=18 dBm time=12.816 ms
12 bytes from fe80::ff:39:icmp_seq=1 ttl=64 rssi=18 dBm time=12.816 ms
12 bytes from fe80::ff:39:icmp_seq=2 ttl=64 rssi=17 dBm time=12.823 ms
--- fe80::ff:39 PING statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max=12.816/12.818/12.823 ms

HATBED 记录的部分追踪数据使用 Pulseview 可视化后的结果如图 8 所示。

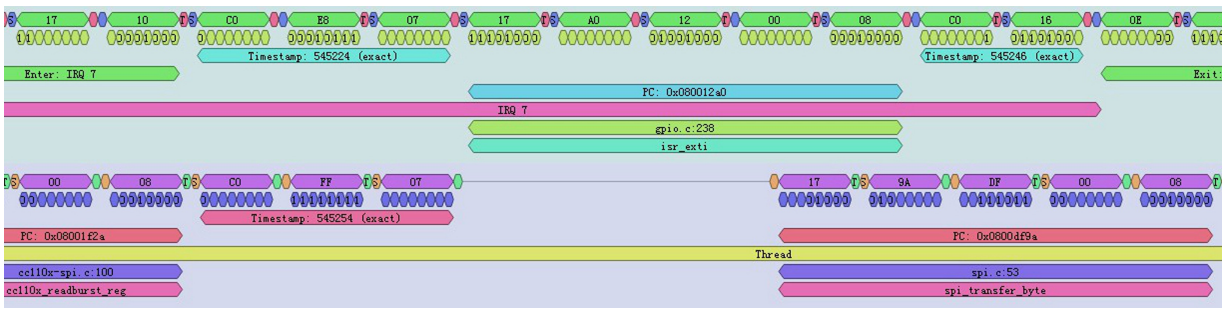

图 8 ping6 追踪记录

Fig. 8 Captured ping6 traces

HATBED 使用 DWT Watchpoint 监测定期 GPIO 翻转以辅助获知 ping6 过程中事件发生的准确时间。例如，图 8 中 CPU 进入中断 IRQ7 的时刻，首先找到最邻近 GPIO 翻转，然后由追踪数据可得 GPIO 翻转内部的增量时间戳为 9 688 768，外部 Unix 的时间戳为 1 561 298 751.047 557，而进入 IRQ7 的内部时间戳为 545 224，内部时间戳的计量单位为指令周期

1/72 M，再计算出进入 IRQ7 的时刻为 1 561 298 751.047 557－(9 688 768－545 224)/72 M＝1 561 298 750.920 563＝2019/6/23 22:5:50.920 563（北京时间）。统计分析的部分事件如表 7 所列，使用 HATBED 测出节点 1 的 ping6 指令大约用时 12.797 4 ms，这与 ping6 的自身返回值非常接近，这说明 ITM 虽然使用均匀采样方式采集 PC，但仍可对记录事件进行高精度的时间分析。

表 7　ping6 的局部分析结果

Table 7　Part analysis of ping6

Node1 (72 MHz)	Incremental timestamp	Unix timesta mp
GPIO 翻转	9 688 768	1 561 298 751.047 557
进入 IRQ7	545 224	－9 143 544/72 M
PC：0x080012a0	545 246	－9 143 522/72 M
退出 IRQ7	545 775	－9 142 993/72M
PC：0x080024a0(开始 ping6)	123 780	－9 564 988/72 M
PC：0x08009144(结束 ping6)	1 045 193	－8 643 575/72 M

6.2　实例 2　GNRC UDP 通信覆盖率测试

烧写代码恢复运行后，节点 1 与节点 2 使用串口 shell 按顺序运行表 8 中的指令，整个过程由 HATBED 记录下来(使用 DWT Watchpoint 监测相关函数入口点，而不使用 ETM)，记录得到的追踪数据用于函数覆盖与基本块覆盖分析。UDP 通信过程主要在 udp_cmd 函数中进行，udp_cmd 中与 udp 通信过程相关的函数共有 18 个，实验使用 GlobalCounter 为其中的 8 个函数进行编码，并在函数入口处插入一条指令，将 GlobalCounter 改为该函数对应的编码值。DWT Watchpoint 监测 GlobalCounter 并报告赋值时的 PC 与编码值，并将结果记录在追踪数据中。

表 8　GNRC 节点 1 与节点 2 运行 shell 指令

Table 8　GNRC shell commands of node 1 and node 2

Num	Node 1	Node 2
1	ifconfig 7 add fe80::ff:dc	ifconfig 7 add fe80::ff:39
2	ping6 fe80::ff:39	udp server start 8808
3	udp server start 8808	udp send fe80::ff:dc 8888 abcdefg 2
4	udp send fe80::ff:39 8808 abcdefg 2	—

HABTED 最终捕获到 18 个函数中的 16 个，函数覆盖率达到 88.9%，但该结果是在没有专门设计测试方案时得到的。

关于基本块覆盖率，由于考虑外部跳转的全局基本块覆盖测试需要对代码进行完整分析[22]，因此需要插桩大量的 DWT Watchpoint，从而带来巨大的开销，故我们将基本块覆盖率的测试集中在 udp_cmd 函数内部，用内部跳转划分 udp_cmd。首先得到 udp_cmd 函数的汇编代码，然后划分内部基本块，划分过程如图 9 所示。这里将函数调用指令bl 800d2a0(调用 atoi 函数)视为一般指令，而对内部跳转 beq.n 8001838 与 b.n 800183a 进行基本块划分，总共得到 55 个基本块。统计追踪数据中出现的所有 PC 值，并与其所属基本块相匹配，得到的基本块覆盖率达到了 34.5%。

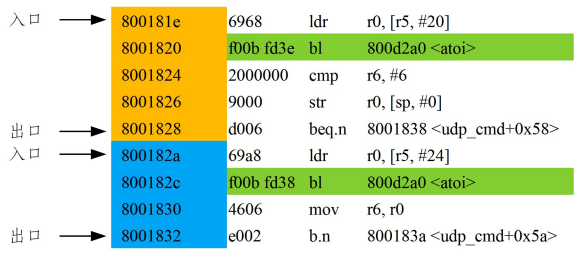

图 9　udp_cmd 基本块示例

Fig. 9　udp_cmd basic block example

结束语　HATBED 是基于现代 32 位 MCU 体系结构的物联网非侵入式测试平台。本文通过基准测试初步评估证实了设计的可行性，通过 2 个 RIOT-OS 测试实例验证了 HATBED 可用于物联网系统高时间精度软件分析测试。本文设计的 HATBED 具有以下创新性和优点：

1)基于标准化调试单元与通用硬件辅助追踪技术。HATBED 具备广泛适用性，不需要复杂的插桩操作，不特定于某个应用程序或某个操作系统。

2)对 Cortex-M3/M4 系列内核硬件辅助追踪单元的深度应用。越来越多的以 IoT 节点为代表的物联网设备具备 Cortex-M3/M4 系列内核，其硬件辅助追踪单元能够极大地帮助开发人员对代码进行充分的测试与分析。本文在真实节点的真实应用上验证了该方法的有效性。

3)易于构建，操作简便。HATBED 由通用设备组成，开发人员可以快速搭建出一套测试平台用于物联网系统测试与评价。

综上所述，与传统的物联网测试平台相比，HATBED 可以在部署前对资源受限的物联网系统开展更加高效、充分的测试与评价，进而提高系统的可靠性。

参 考 文 献

[1] BOANO C A, DUQUENNOY S, FÖRSTER A, et al. IoT-Bench: Towards a benchmark for low-power wireless networking[C]// 2018 IEEE Workshop on Benchmarking Cyber-Physical Networks and Systems (CPSBench). IEEE, 2018: 36-41.

[2] DODDAVENKATAPPA M, CHAN M C, ANANDA A L. Indriya: A low-cost, 3D wireless sensor network testbed[C]// International Conference on Testbeds and Research Infrastructures. Springer, Berlin, Heidelberg, 2011: 302-316.

[3] ADJIH C, BACCELLI E, FLEURY E, et al. FIT IoT-LAB: A large scale open experimental IoT testbed[C]// 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT). IEEE, 2015: 459-464.

[4] WERNER-ALLEN G, SWIESKOWSKI P, WELSH M. Motelab: A wireless sensor network testbed[C]// Proceedings of the 4th International Symposium on Information Processing in Sensor Networks. IEEE Press, 2005: 68.

[5] KANZAKI A, HARA T, ISHI Y, et al. X-sensor: A sensor network testbed integrating multiple networks[C]// 2009 International Conference on Complex, Intelligent and Software Intensive Systems. IEEE, 2009: 1082-1087.

[6] HANDZISKI V, KÖPKE A, WILLIG A, et al. Twist: a scalable and reconfigurable testbed for wireless indoor experiments with sensor networks[C]// Proceedings of the 2nd International Workshop on Multi-hop Ad Hoc Networks: From Theory to Reality. ACM, 2006: 63-70.

[7] TYTGAT L, JOORIS B, DE MIL P, et al. Demo abstract: WiLab, a real-life wireless sensor testbed with environment emulation[C]// 6th European Conference on Wireless Sensor Networks (EWSN 2009). 2009.

[8] WEBER M, WEBER M. A Competition to Push the Dependability of Low-Power Wireless Protocols to the Edge[C]// International Conference on Embedded Wireless Systems & Networks. Junction Publishing, 2017.

[9] DUCROCQ T, VANDAËLE J, MITTON N, et al. Large scale geolocalization and routing experimentation with the SensLAB testbed[C]// The 7th IEEE International Conference on Mobile Ad-hoc and Sensor Systems (IEEE MASS 2010). IEEE, 2010: 751-753.

[10] ZHAO Z, LIU Q, LI D, et al. EasiTest: A Multi-Radio Testbed for Heterogeneous Wireless Sensor Networks[J]. Jisuanji Yanjiu yu Fazhan (Journal of Computer Research and Development), 2012, 49(3): 506-517.

[11] LI T L, HUANG P W, YANG X, et al. Design and Implementation of Non-intrusive Wireless Sensor Network Tester[J]. Jisuanji Kexue (Computer Science), 2010, 37(4): 45-48.

[12] TANCRETI M, HOSSAIN M S, BAGCHI S, et al. Aveksha: A hardware-software approach for non-intrusive tracing and profiling of wireless embedded systems[C]// Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems. ACM, 2011: 288-301.

[13] LIM R, FERRARI F, ZIMMERLING M, et al. Flocklab: A testbed for distributed, synchronized tracing and profiling of wireless embedded systems[C]// Proceedings of the 12th International Conference on Information Processing in Sensor Networks. ACM, 2013: 153-166.

[14] TRüB R, DA FORNO R, GSELL T, et al. Demo Abstract: A Testbed for Long-Range LoRa Communication[C]// 2019 18th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN). IEEE, 2019: 342-343.

[15] SOMMER P, KUSY B. Minerva: Distributed tracing and debugging in wireless sensor networks[C]// Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems. ACM, 2013: 12.

[16] LI J, WANG C, YANG Y. A General Purpose Testbed for Mobile Data Gathering in Wireless Sensor Networks and a Case Study[C]// 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS). IEEE, 2017.

[17] KAMPIANAKIS E, KIMIONIS J, TOUNTAS K, et al. A Remotely Programmable Modular Testbed for Backscatter Sensor Network Research[J]. Lecture Notes in Electrical Engineering, 2014, 281: 153-161.

[18] LATRE S, LEROUX P, COENEN T, et al. City of things: An integrated and multi-technology testbed for IoT smart city experiments[C]// 2016 IEEE International Smart Cities Conference (ISC2). IEEE, 2016.

[19] PRADEEP P, DIVYA P, DEVI R D A, et al. A remote triggered wireless sensor network testbed[C]// Wireless Telecommunications Symposium (WTS). IEEE, 2015.

[20] BELBACHIR, ASSIA. An embedded testbed architecture to evaluate autonomous car driving[J]. Intelligent Service Robotics, 2017, 10(2): 109-119.

[21] BURR K, YOUNG W. Combinatorial test techniques: Table-based automation, test generation and code coverage[C]// Proc. of the Intl. Conf. on Software Testing Analysis & Review. 1998.

[22] WU Y X, GU G C, WANG K H. Partition method of control flow checking-based low-powered basic block [J]. Jisuanji Gongcheng yu Yingyong (Computer Engineering and Applications), 2007, 42(25): 118-120.

MA Jun-yan, born in 1982, Ph.D, associate professor, postgraduate supervisor, is a member of China Computer Federation. His main research interests include embedded system and software, intelligent vehicle infrastructure system test and evaluation, etc.