

# “嵌入式与物联网开发技术” 线上分享系列课程

## 第四讲：从零开始学习RTOS 分析工具的使用

主讲人：付元斌

2020年8月20日

# 课程内容



- Tracealyzer可视化工具介绍
  - 产品特性
  - 工作模式
- 在RISC-V处理器上集成Tracealyzer
  - 实现时间戳生成的接口
  - 记录器库的临界区实现
- 跟踪记录器库的配置
  - 快照模式配置
  - 流模式配置
- 快照模式和流模式的使用
  - 操作演示

# Tracealyzer可视化分析工具



- Tracealyzer是Percepio 公司开发的一款用于RTOS或基于linux的嵌入式软件系统的可视化跟踪工具
- 提供了30多种相互关联的运行时行为视图，包括任务调度、中断、任务之间的相互作用，以及从应用程序代码中记录的用户事件，且不需要额外的硬件。
- Tracealyzer作为传统调试的补充，提供更高层次的调试视图，非常适合理解典型的实时问题。

## Tracealyzer支持的OS

FreeRTOS

Keil RTX5

Linux

On Time RTOS-32

ThreadX

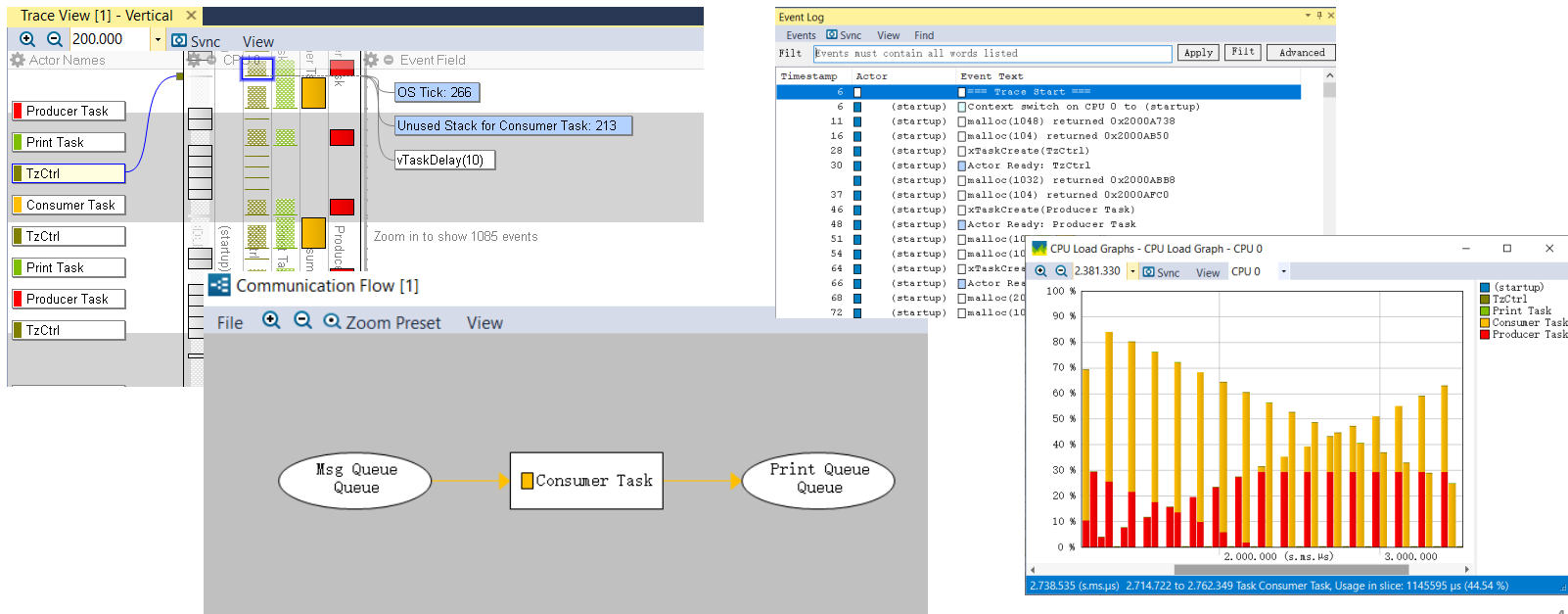
μC/OS-III

OpenVX/Synopsys

VxWorks

# Tracealyzer可以实现的分析功能

- 提供了时间轴, CPU负载, 通信流图、事件窗口等在内的三十多种可视化分析视图, 可非常详细地观测任务调度, 内核对象之间的通信, 以及用户定义的事件



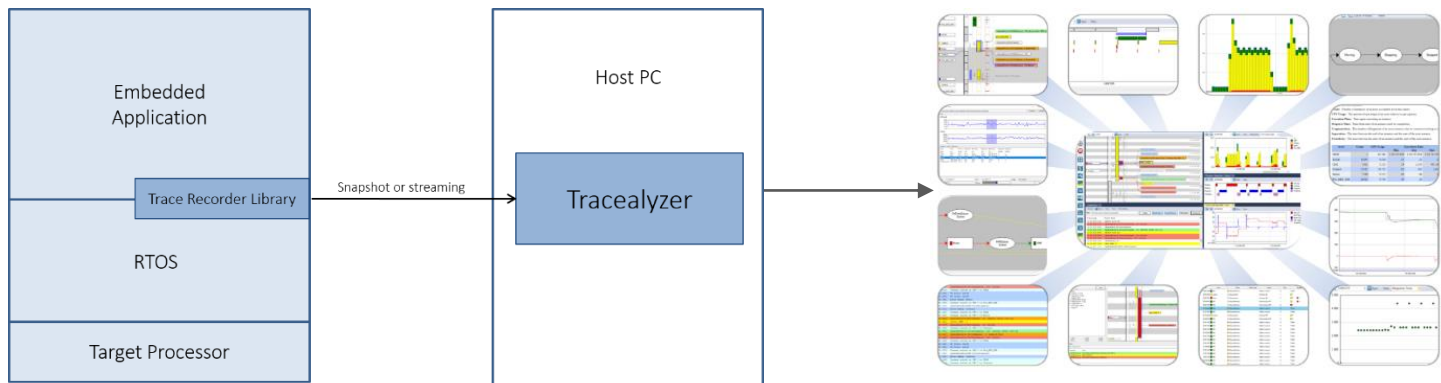
# Tracealyzer的工作模式

- **快照模式(Snapshot Mode)**

- 跟踪数据存储在RAM中的缓存，事后通过调试器将跟踪数据转存为hex/bin文件到主机进行分析
- 可跟踪时间短，时间受限于目标系统RAM可用空间的大小，适合产品部署时使用

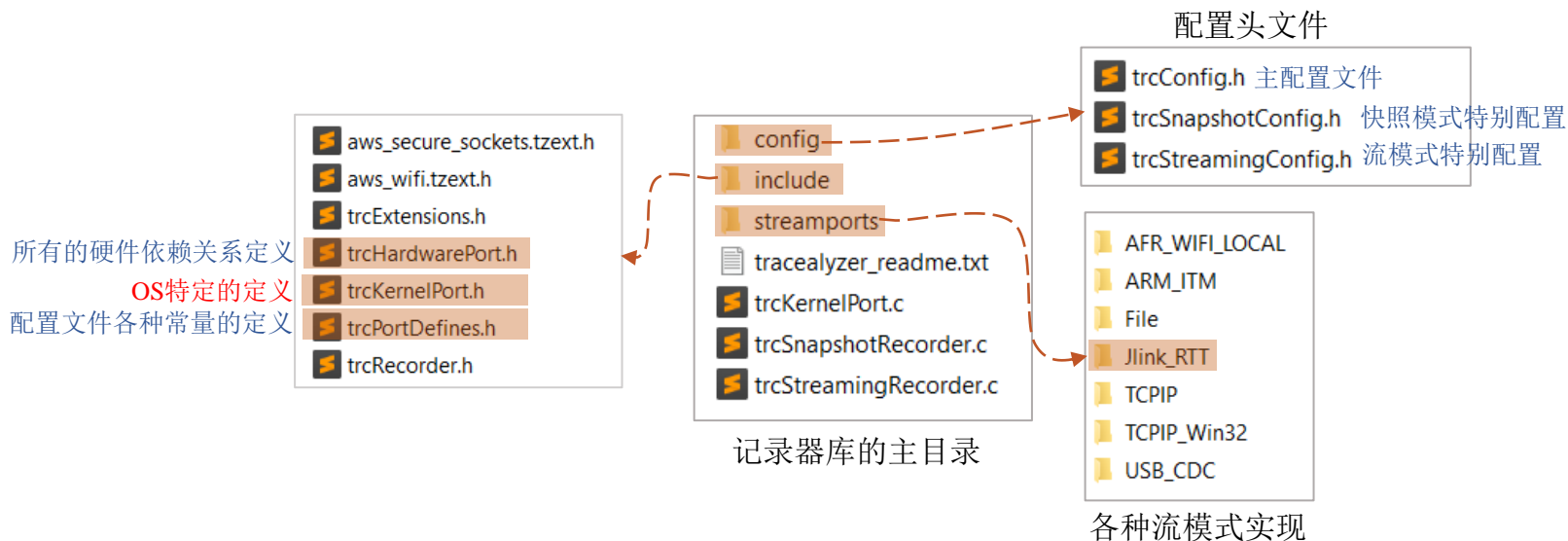
- **流模式(Streaming Mode)**

- 跟踪数据实时地通过调试接口、USB、串口、以太网口等通讯接口发送到主机进行实时分析
- 可长时间跟踪，时间受限于主机硬盘可用空间的大小，适合开发阶段使用



# 跟踪记录器库(Trace Recorder Library)

- 跟踪记录器库与应用一起构建，在系统运行时生成和记录各种事件
- 库的源码文件在Tracealyzer安装目录下以OS命名的文件夹内
- 核心文件：`trcKernelPort.c`、`trcSnapshotRecorder.c`、`trcStramingRecorder.c`



# 在RISC-V处理器上集成Tracealyzer

- Tracealyzer不依赖具体的处理器，但是需要一个高精度的定时/计数器，用于实现时间戳
- 在- TRC\_HWTC\_TYPE: 使用的定时/计数器类型
- TRC\_HWTC\_COUNT: 读取定时/计数器的方法
- TRC\_HWTC\_PERIOD: 定时/计数器结束计数前HWTC\_COUNT 的计数次数
- TRC\_HWTC\_DIVISOR: 定时器分频(仅用于快照模式，减少存储时间戳的带宽)
- TRC\_HWTC\_FREQ\_HZ: 定时/计数器的时钟频率
- TRC\_IRQ\_PRIORITY\_ORDER: 中断优先级数值和级别的对应关系，为0表示越低的 interrupt 优先级数值代表更高的优先级，为1反之



# 在GD32VF103实现

- RISC-V 架构定义了一个 64 位宽的时钟周期计数器，用于反映处理器执行了多少个时钟周期，只要处理器处于执行状态时，此计数器便会不断自增计数
- mcycle 寄存器反映了该计数器低 32 位的值，mcycleh 寄存器反映了该计数器高 32 位的值
- 因此，可以使用mcycle来作为产生时间戳的计数器
  
- `#elif (TRC_CFG_HARDWARE_PORT == TRC_HARDWARE_PORT_RISCV_GD32V)`
- `#define TRC_HWTC_TYPE TRC_FREE_RUNNING_32BIT_INCR`
- `#define TRC_HWTC_COUNT read_csr(cycle)`
- `#define TRC_HWTC_PERIOD 0`
- `#define TRC_HWTC_DIVISOR 4`
- `#define TRC_HWTC_FREQ_HZ TRACE_CPU_CLOCK_HZ`
- `#define TRC_IRQ_PRIORITY_ORDER 1`

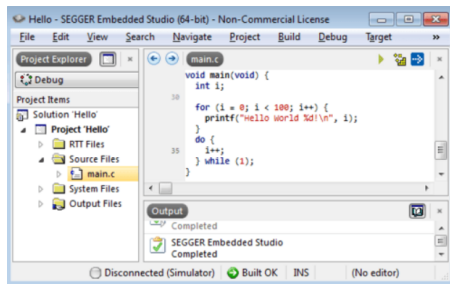
# 在GD32VF103实现



- 在trcKernelPort.h中定义Tracealyzer临界区的进入和退出方式，不建议使用OS实现的进入和退出临界区方法，对Tracealyzer来说不安全；因为有些记录器函数可能会在两个上下文中调用，最好是彻底关闭中断
- ```
#if (TRC_CFG_HARDWARE_PORT == TRC_HARDWARE_PORT_RISCV_GD32V)
```
- ```
    #define TRACE_ALLOC_CRITICAL_SECTION() int __irq_status;
```
- ```
    #define TRACE_ENTER_CRITICAL_SECTION() __irq_status= portSET_INTERRUPT_MASK_FROM_ISR();
```
- ```
    #define TRACE_EXIT_CRITICAL_SECTION() portCLEAR_INTERRUPT_MASK_FROM_ISR(__irq_status);
```
- ```
#endif
```

# 从零开始入门Tracealyzer—软硬件环境

- 硬件采用GD32VF103V-EVAL开发板，GD32VF103系列是基于芯来科技Bumblebee内核的RISC-V架构MCU，提供了108MHz的主频，以及16KB到128KB的片上闪存和6KB到32KB的SRAM缓存
- IDE使用SEGGER的Embedded Studio 的RISC-V版本，调试器使用J-Link
- 运行FreeRTOS，Tracealyzer使用v4.3.11最新版本



## 跟踪记录器库的配置

- 在trcConfig.h中:

- 选择适用的处理器移植实现, 例如:

- #define TRC\_CFG\_HARDWARE\_PORT TRC\_HARDWARE\_PORT\_ARM\_Cortex\_M

- 设置使用的跟踪模式(快照模式)

- #define TRC\_CFG\_RECORDER\_MODE TRC\_RECORDER\_MODE\_SNAPSHOT

- 设置FreeRTOS的版本

- #define TRC\_CFG\_FREERTOS\_VERSION TRC\_FREERTOS\_VERSION\_10\_2\_1

- 在trcSnapshotConfig.h中:

- 设置快照模式的记录模式 (缓存满了后停止跟踪或者覆盖)

- #define TRC\_CFG\_SNAPSHOT\_MODE TRC\_SNAPSHOT\_MODE\_STOP\_WHEN\_FULL

- 设置事件缓存的大小, 单位是字

- #define TRC\_CFG\_EVENT\_BUFFER\_SIZE 1000

- 在trcConfig.h中：
  - 选择适用的处理器移植实现，例如：
    - #define TRC\_CFG\_HARDWARE\_PORT TRC\_HARDWARE\_PORT\_ARM\_Cortex\_M
  - 设置使用的跟踪模式(流模式)
    - #define TRC\_CFG\_RECORDER\_MODE TRC\_RECORDER\_MODE\_STREAMING
  - 设置FreeRTOS的版本
    - #define TRC\_CFG\_FREERTOS\_VERSION TRC\_FREERTOS\_VERSION\_10\_2\_1
- 在trcStreamingConfig.h中：
  - 设置符号槽大小
    - #define TRC\_CFG\_SYMBOL\_TABLE\_SLOTS 40
  - 设置每页的事件缓存
    - #define TRC\_CFG\_PAGED\_EVENT\_BUFFER\_PAGE\_SIZE 500

# 进阶配置(1)

- 在trcConfig.h中有一些配置选项用于控制对某些事件的记录，以此来减少产生的事件，延长快照模式的记录时间，或者避免流模式所用接口速率较低时传输出现溢出
- 仅记录调度事件
  - `#define TRC_CFG_SCHEDULING_ONLY 0`
- 记录内存分配和释放操作(malloc/free)
  - `#define TRC_CFG_INCLUDE_MEMMANG_EVENTS 1`
- 记录用户事件
  - `#define TRC_CFG_INCLUDE_USER_EVENTS 1`
- 记录中断事件
  - `#define TRC_CFG_INCLUDE_ISR_TRACING 1`

# 进阶配置(2)

- 记录任务就绪事件
  - `#define TRC_CFG_INCLUDE_READY_EVENTS 1`
- 记录系统节拍事件(OS Tick)
  - `#define TRC_CFG_INCLUDE_OSTICK_EVENTS 0`
- 记录事件组(event group)
  - `#define TRC_CFG_INCLUDE_EVENT_GROUP_EVENTS 0`
- 记录软件定时器
  - `#define TRC_CFG_INCLUDE_TIMER_EVENTS 0`
- 记录pending函数调用, 如xTimerPendFunctionCall()
  - `#define TRC_CFG_INCLUDE_PEND_FUNC_CALL_EVENTS 0`



# 进阶配置(3)

- 堆栈监测控制
  - `#define TRC_CFG_ENABLE_STACK_MONITOR 0`
- 堆栈监测任务数
  - `#define TRC_CFG_STACK_MONITOR_MAX_TASKS 10`
- 每次堆栈检测的任务数
  - `#define TRC_CFG_STACK_MONITOR_MAX_REPORTS 1`
- 控制任务的优先级
  - `#define TRC_CFG_CTRL_TASK_PRIORITY 1`
- 控制任务的延迟时间
  - `#define TRC_CFG_CTRL_TASK_DELAY 10`
- 控制任务的堆栈大小
  - `#define TRC_CFG_CTRL_TASK_STACK_SIZE (configMINIMAL_STACK_SIZE * 2)`

## 快照模式和流模式的使用

# 启用快照模式跟踪

- 在FreeRTOSConfig.h中将跟踪功能使能，并包含跟踪记录器库的主头文件
  - #define configUSE\_TRACE\_FACILITY 1
  - #include "trcRecorder.h"
- 在应用的main函数中调用vTraceEnable(TRC\_START);，调用的位置必须在硬件完成初始化之后，且在创建第一个系统对象之前

```
/* Configure the system clock to 400 MHz */
system_config();

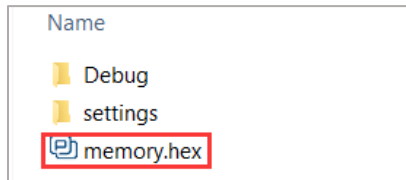
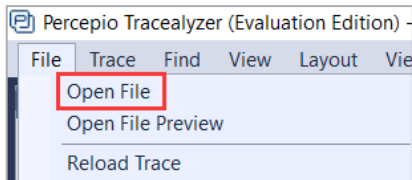
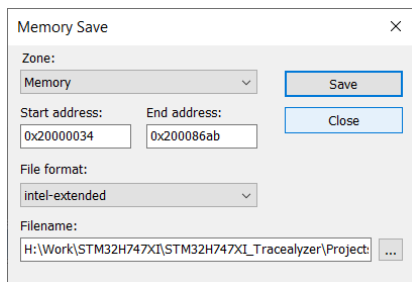
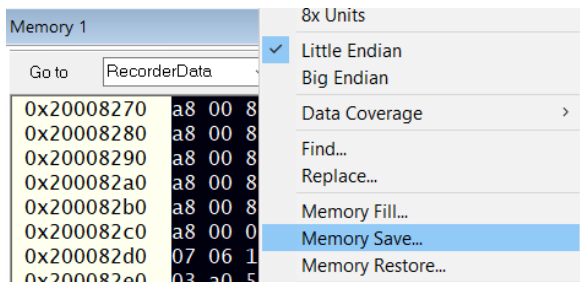
/* Initialize LED2 */
BSP_LED_Init(LED1);
BSP_LED_Init(LED2);

vTraceEnable(TRC_START);

/* Create tasks */
xTaskCreate( vTaskProducer, "Producer Task", PRODUCER_TASK_STK_SIZE,
            NULL, PRODUCER_TASK_PRIO, &ProducerHandle);
```

# 主机端读取快照数据(1)

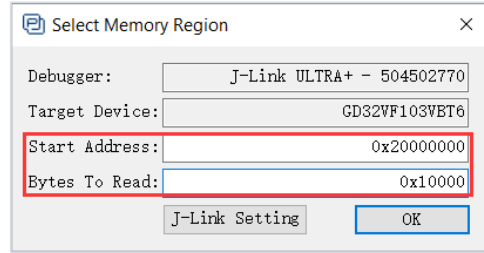
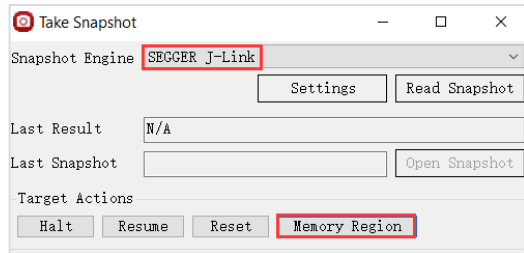
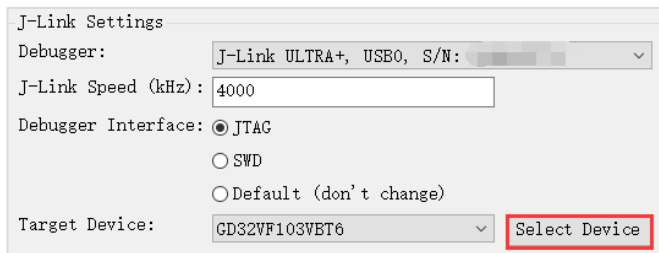
- 通过IDE的Memory转存储功能
  - 通过IDE的watch窗口查看“RecorderData”的起止地址
  - 将RecorderData保持为hex或bin文件
  - 在Tracealyzer加载转存储的hex或bin文件



# 主机端读取快照数据(2)

- 通过J-link

- 从Tracealyzer > File > Settings > J-Link Settings设置所使用的目标芯片，以及调试接口(JTAG/SWD)
- 从Trace > Open Snapshot tool打开快照获取窗口，选择使用SEGGER J-Link，并设置要读取的范围
- 通过快照获取窗口上的“Read Snapshot” 读取快照数据



# 启用流模式跟踪

- 在FreeRTOSConfig.h中将跟踪功能使能，并包含跟踪记录器库的主头文件
  - #define configUSE\_TRACE\_FACILITY 1
  - #include "trcRecorder.h"
- 在应用的main函数中调用vTraceEnable(TRC\_INIT);，调用的位置必须在硬件完成初始化之后，且在创建第一个系统对象之前

```
/* Configure the system clock to 400 MHz */
system_config();

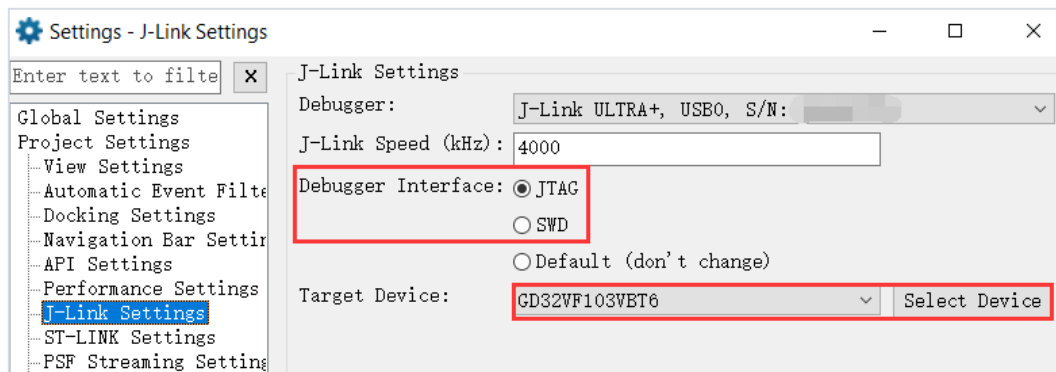
/* Initialize LED2 */
BSP_LED_Init(LED1);
BSP_LED_Init(LED2);

vTraceEnable(TRC_INIT);

/* Create tasks */
xTaskCreate( vTaskProducer, "Producer Task", PRODUCER_TASK_STK_SIZE,
            NULL, PRODUCER_TASK_PRIO, &ProducerHandle);
```

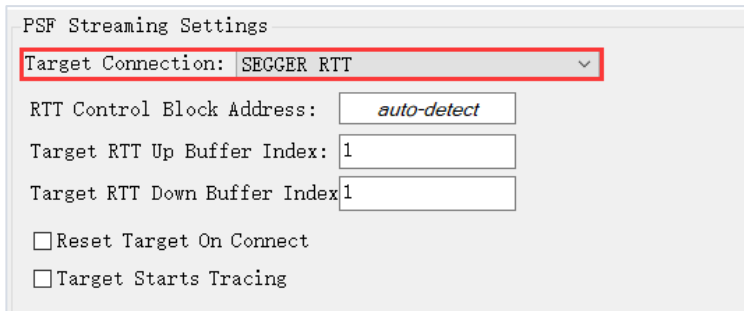
# 流模式连接(1)

- 采用J-Link RTT通信实现流模式跟踪，按如下设置：
- 启动Tracealyzer应用软件，从File->Settings打开Global Settings，选择J-Link Settings
- 设置调试接口类型(JTAG/SWD)，并选择所使用的芯片型号



## 流模式连接(2)

- 从File->Settings打开Global Settings, 选择PSF Streaming Settings, 将目标连接选择为SEGGER RTT
- RTT Control Block地址可以选择自动, 如果不能自动识别才需要手动设置



PSF Streaming Settings

Target Connection: SEGGER RTT

RTT Control Block Address: *auto-detect*

Target RTT Up Buffer Index: 1

Target RTT Down Buffer Index: 1

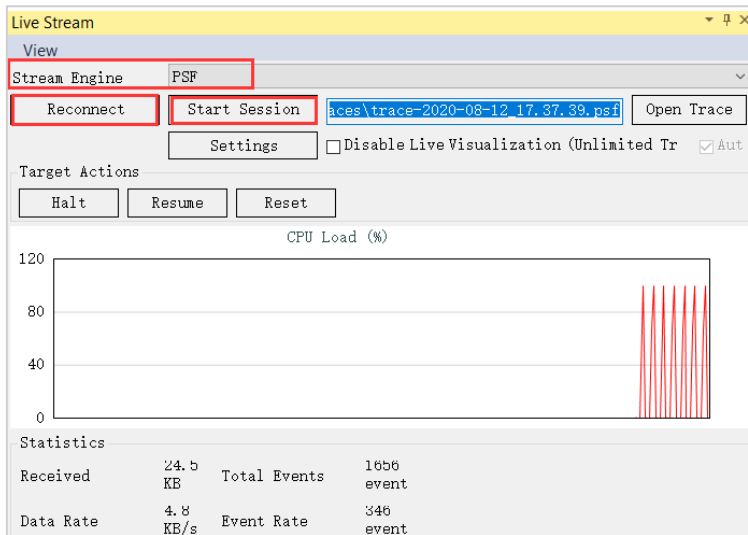
Reset Target On Connect

Target Starts Tracing



# 流模式连接(3)


- 打开Live Stream窗口，然后将Stream Engine选择为PSF
- 点击Connection按钮建立连接，再点击Start Session按钮开启实时流跟踪



# 试用和技术资源

- 获取试用
  - 从Percepio官网获取Tracealyzer评估版: <https://percepio.com/>
- 技术资源
  - Percepio blog: <https://percepio.com/blog/>
  - BMR网站技术资源: <http://www.bmrtech.com/Tech/index/192.html>

Tracealyzer (v4.3.7) for FreeRTOS



Learn more about [Tracealyzer for FreeRTOS](#)

Note that the free evaluation licenses are for evaluation only. If you like our product, purchase a license to support our development. However, we have free or discounted offers for some use cases (see [licensing](#)).

If you want to upgrade from an older version of Tracealyzer (v3.x or earlier), please contact [sales@percepio.com](mailto:sales@percepio.com).

By registering, you allow representatives of Percepio AB to contact you regarding your interest in Tracealyzer. Read more in the [privacy policy](#).

Target Platform\*

Host OS\*

Name (First, Last)\*

Email\*

Company

Phone

Country  (\* required field)

**New Evaluation?**

I would like to evaluate, please send a license.

I'm a customer already, just the download link please.

# 答疑时间

- info@bmrtech.com
- 网站: <http://www.bmrtech.com>
- 本次讲座实验代码
- 链接: <https://eyun.baidu.com/s/3bqvBtPP>
- 密码: 7RtJ



# 感谢您的聆听!

北京麦克泰软件技术有限公司

- 网址: [www.bmrtech.com](http://www.bmrtech.com)
- 邮箱: [info@bmrtech.com](mailto:info@bmrtech.com)
- 北京: 010-57625727
- 上海: 021-62127690
- 深圳: 0755-82977971

