

依据 RTKLIB 的标准结构和您提供的 rtknavi.c、pntpos.c 等文件的上下文，详细解释这个线程的流程、输入、输出和关键变量。

rtksvrthread 函数解析

1. 总体结构与作用

- **函数原型:** void *rtksvrthread(void *arg)
- **作用:** 作为守护进程在后台持续运行。它实现了 读-解码-同步-解算-写 的完整 RTK/GNSS 处理循环。
- **输入:**
 - arg: 类型为 void *, 实际传递的是一个指向 rtksvr_t 结构体（即 rtknavi.c 中配置的 rtksvr 变量）的指针。
- **输出:**
 - 函数本身返回值通常为 NULL, 因为这是一个不退出的无限循环线程。
 - 主要输出是通过服务器结构体写入到外部数据流（例如，将定位结果写入文件，如 rtknavi.c 中配置的 paths[3]）。

2. rtksvrthread 详细流程（核心循环）

rtksvrthread 函数主要包含一个无限循环，按配置的周期（在 rtknavi.c 中配置为 SvrCycle=10 毫秒）执行以下步骤：

步 骤	流程描 述	关键操作/子函 数调用	输入变量（来自 rtksvr）	赋值有效值的变量
0. 准 备	线程初 始化： 线程启 动后， 首先从 arg 中 取出 rtksvr_t 结构体 指针，	stropen()	svr->stream (包 含路径和类型)	svr->stream[i].fp (文 件指针)

步骤	流程描述	关键操作/子函数调用	输入变量（来自rtksvr）	赋值有效值的变量
	并执行必要的初始化（如打开所有配置的流）。			
1. 数据输入	读取原始数据：循环读取所有活动的输入流（流动站、基站、修正数据等）。	strread()	svr->stream[i]	svr->raw[i] 或 svr->rtcm[i].buff (缓冲区)
2. 数据解码	解码并存储数据：将读取到的原始字节数据解码成GNSS观测值和导航电文。	input_rtcm3() / input_raw() (在rtcm.c中)	svr->rtcm[i].buff 或 svr->raw[i].buff	svr->rtcm[i].obs.data (观测值) svr->rtcm[i].nav (导航电文/星历) svr->rtcm[i].sta.pos (基站坐标)

步骤	流程描述	关键操作/子函数调用	输入变量（来自rtksvr）	赋值有效值的变量
3. 数据同步	观测值同步： 检查流动站和基站是否有最新且时间上接近的观测数据，以及是否有最新的星历可用。	rtksyncedge() (自定义函数)	svr->rtcm[0].obs, svr->rtcm[1].obs	svr->obs[0], svr->obs[1] (解算用的观测值) svr->nav (合并后的星历库)
4. 定位解算	执行定位：如果数据同步成功，则开始定位解算。	A. pntpos() (在 pntpos.c 中) B. relpos() (未在列表中，但依赖 lambda.c)	svr->obs[0] (流动站观测值) svr->obs[1] (基站观测值) svr->nav (星历) svr->prcopt (处理选项)	svr->sol (解算结果)

步 骤	流程描 述	关键操作/子函 数调用	输入变量（来自 rtksvr）	赋值有效值的变量
5. 结 果 输 出	写入解 算结 果： 将 计算出 的最新 解算结 果写入 到输出 流。	out_stat() / out_sol()	svr->sol (最新解) svr->solopt (输 出选项) svr->rtk.sol (RTK 解算器的状态)	svr->stream[3] (解算 日志文件)
6. 循 环 控 制	周期等 待： 根 据配置 的解算 周期等 待， 然 后进入 下一个 循环。	sleep(SvrCycle)	SvrCycle (10ms)	-

3. 关键变量的赋值和传递

在 `rtksvrthread` 内部，核心变量的传递和赋值是通过 `rtksvr_t` 结构体 的成员完成的。

变量成员	赋值/初始化来 源	传递/使用 到的函数	变量的作用
<code>svr->prcopt</code>	赋值： 来自 main 函数传入 的 PrcOpt，在 <code>rtksvrstart</code> 中	<code>pntpos()</code> ： 控制 SPP 的改正模 型和滤波	核心处理选项，如电 离层模型、截止角、 模式 (PMODE_KINEMA)。

变量成员	赋值/初始化来源	传递/使用到的函数	变量的作用
	被复制。	器参数。 relpos(): 控制 RTK 模糊度固 定和解算 模式。	
svr->sol	初始化: 在 rtksvinit 中清 零。 赋值: 由 pntpos() 和 relpos() 函数 修改。	pntpos(): 首次写入 SPP 解 (初始位 置)。 relpos(): 覆盖写入 RTK 高精 度解。	存储当前的定位结果 (位置 rr、速度 rr+3、状态 stat)。
svr->rtcm[i].obs.data	赋值: 由 rtcm.c 中的 input_rtcm3 解 码后写入。	pntpos(): SPP 解算 的观测值 输入。 relpos(): RTK 解算 的观测值	存储原始的伪距、载 波相位和多普勒观测 值。

变量成员	赋值/初始化来源	传递/使用到的函数	变量的作用
		输入。	
svr->rtcm[i].nav	赋值: 由 RTCM 消息 (如 MT1019) 解码后写入。	pntpos(): 传递给 satpos() 计算卫星 状态。	存储流动站/基站接 收到的广播星历等导 航数据。
svr->nav	赋值: 由 rtksvrthread 合 并来自所有输 入流的星历数 据。	pntpos() / relpos(): 作为计算 卫星位置 时查找星 历的最终 数据源。	全局导航数据结构, 包含所有系统的最新 星历。
svr->rtcm[1].sta.pos	赋值: 由基站 RTCM 消息 (如 MT1005/1006) 解码后写入。	relpos(): 作为相对 定位的固 定参考 点。	基准站的精确三维坐 标。

4. 总结

rtksvrthread 函数是 RTKLIB 的**神经中枢**。它持续不断地从各种输入流中提取数据，通过 input_rtcm3 等函数将原始数据转化为可用的观测值和星历，然后交由 pntpos 和 relpos 进行定位解算，最后将高精度的结果写入输出文件。所有数据的流动和状态的维护都围绕着其核心变量——**rtksvr_t** 结构体进行。