A76xx Series Open_SDK NTP HTP_应用指导

LTE 模组

芯讯通无线科技(上海)有限公司
上海市长宁区临虹路289号3号楼芯讯通总部大楼
电话：86-21-31575100
技术支持邮箱：support@simcom.com
官网：www.simcom.com

名称：	A76xx Series Open SDK_NTP HTP_应用指导
版本：	V1.00
类别：	应用文档
状态：	已发布

版权声明

芯讯通无线科技(上海)有限公司
上海市长宁区临虹路289号3号楼芯讯通总部大楼
电话：86-21-31575100
邮箱：simcom@simcom.com
官网：www.simcom.com

了解更多资料，请点击以下链接：
http://cn.simcom.com/download/list-230-cn.html

技术支持，请点击以下链接：
http://cn.simcom.com/ask/index-cn.html 或发送邮件至 support@simcom.com

Version History

Version	Date	Owner	What is new
V1.00	2022-11-17		第一版

About this Document

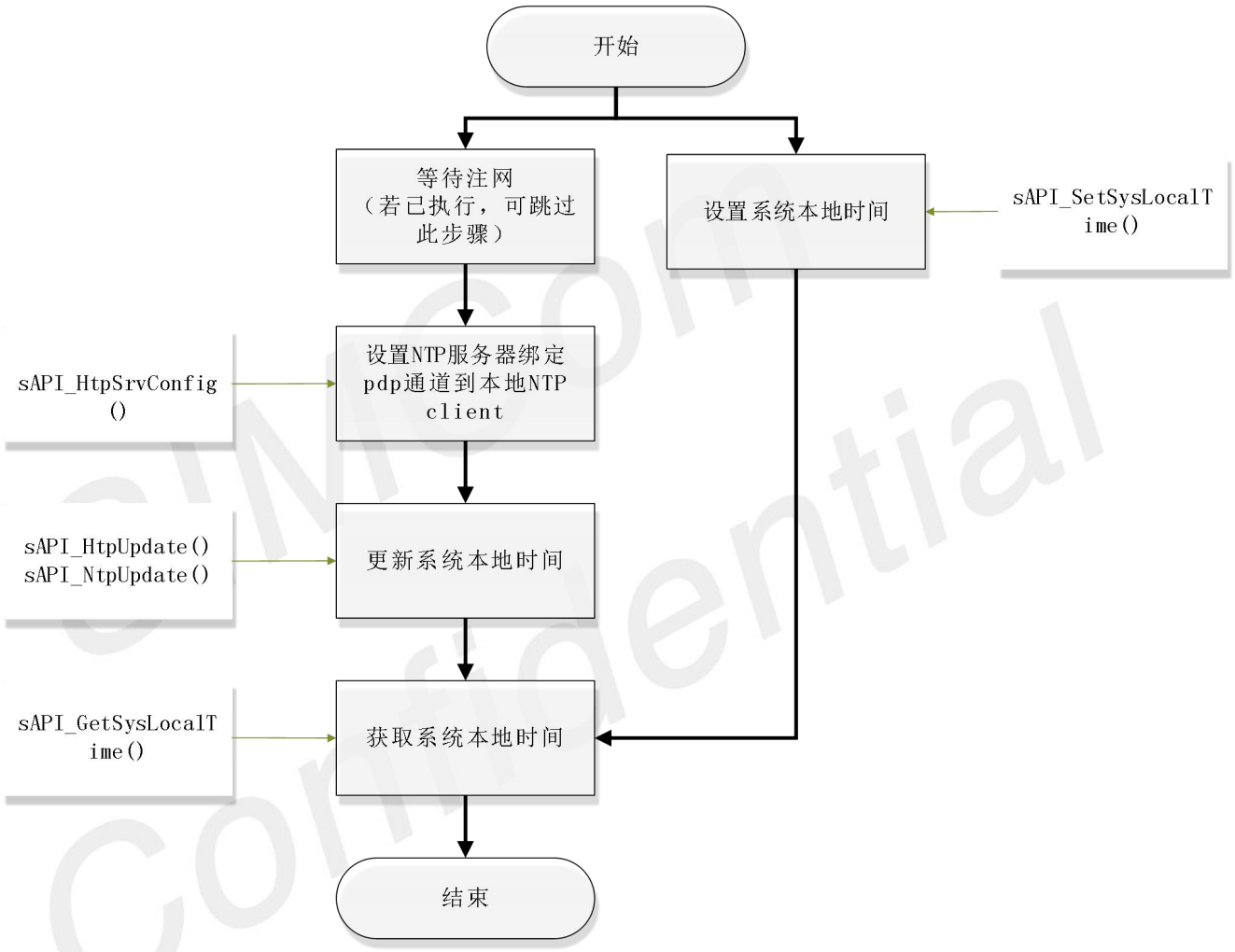
本文档适用于 A1803S open 系列、A1603 open 系列、A1606 open 系列。

目录

缩略语

NTP	network time protocol
HTP	High Speed Transmission Protocol
PDP	Packet Data Protocol
IP	Internet Protocol

1NTP HTP 使用流程



NTP HTP 工作流程图

在进行 ntp 同步时间之前，需要先进行对应 pdp 通道的数据拨号，数据拨号成功以后，ntp 服务才可以正常使用。若其他任务中已经执行了对应 pdp 的拨号操作，则不需要再重复拨号。

拨号成功以后，先调用 sAPI_HtpSrvConfig 函数设置用于同步 NTP 时间的 NTP 服务器。此函数支持域名或 IP 地址两种参数类型。

设置完 NTP 服务器信息以后，调用 sAPI_HtpUpdate 或 sAPI_NtpUpdate 来同步系统时间，使用 sAPI_GetSysLocalTime 来获取模块中的本地时间。也可使用 sAPI_SetSysLocalTime 设置模块中的本地时间。

2API 介绍

NTP 头文件名： simcom_ntp_client.h
HTP 头文件名： simcom_htp_client.h

2.1 sAPI_HtpSrvConfig

该接口用于增加、删除 HTP 服务器信息，最大支持 16 个 HTP 服务器。

接口:	SChtpReturnCode **sAPI_HtpSrvConfig**(SChtpOperationType commad_type, char *return_string, char *cmd, char* host_or_idx, int host_port, int http_version, char *proxy, int proxy_port);
参数:	[in] commad_type：命令类型 SC_HTP_OP_SET 设置参数，添加或删除 addr SC_HTP_OP_GET 获取列表中的当前地址 [out] return_string:获取地址列表，仅对 SC_HTP_OP_GET 可用 [in] cmd:该命令用于操作 HTP 服务器列表 “ADD”向列表中添加一个 HTP 服务器项 “DEL”从列表中删除 HTP 服务器项 [in] host_or_idx: 如果<cmd>为”ADD”，该字段与<host>相同，长度为 0-255，需要加引号 如果<cmd>为“DEL”，该字段为待删除的 HTP 服务器项的索引，不需要加引号 [in] port: HTP 服务器端口号，取值范围为(1-65535) [in] http_version: HTTP 服务器的 HTTP 版本 0 HTTP 1.0 1 HTTP 1.1 [in] proxy:代理地址，长度为 1 ~ 255。 [in] proxy_port:代理的端口号，取值范围是(1-65535);
返回值:	SC_HTP_OK:进程成功执行 SC_HTP_ERROR: htp 进程繁忙 SC_HTP_INVALID_PARAM:参数错误
NOTE:	通过设置为 NULL 或 0 忽略部分参数

2.2 sAPI_HtpUpdate

该接口用于使用 HTP 协议更新日期时间。

接口:	SChtpReturnCode **sAPI_HtpUpdate**(sMsgQRef magQ_urc);
参数:	[in] magQ_urc:消息队列，最终结果将作为 urc 发送到该消息队列
返回值:	SC_HTP_OK:进程成功执行 SC_HTP_ERROR: http 进程繁忙 SC_HTP_INVALID_PARAM:参数错误 SC_HTP_NETWORK_ERROR: pdp 激活失败
NOTE:	该消息队列应该是有效的，结果将发送到该队列

2.3 sAPI_NtpUpdate

该接口用于通过 NTP 服务器更新系统时间。

接口:	SCntpReturnCode **sAPI_NtpUpdate**(SCntpOperationType commad_type, char* host_addr, int time_zone, sMsgQRef magQ_urc);
参数:	[in] commad_type: SC_NTP_OP_SET 设置主机地址 SC_NTP_OP_GET 获取当前主机地址 SC_NTP_OP_EXC 通过 ntp 协议更新系统时间 [in/out] host_addr: NTP 服务器地址，对于 SC_NTP_OP_GET，服务器地址字符串将复制到此参数 [in] time_zone:本地时区，(-47 到 48);默认为 32 [in] magQ_urc:消息队列，最终结果将作为 urc 发送到该消息队列
返回值:	SC_NTP_OK:进程成功执行 SC_NTP_ERROR_NETWORK_FAIL: pdp 活动失败 SC_NTP_ERROR_INVALID_PARAM:参数错误
NOTE:	这个 API 中的一些参数是不必要的，它应该设置为 NULL 或 0

2.4 sAPI_GetSysLocalTime

该接口用于获取系统时间。

接口:	void **sAPI_GetSysLocalTime**(tm_rtc *currUtcTime);

参数:	[in/out] currUtcTime:
	typedef struct sys_time {
	int tm_sec;//秒(0,59)
	int tm_min;//分钟(0,59)
	int tm_hour;//小时(0,23)
	int tm_mday;//日[1,31]
	int tm_mon;//月[1,12]

	int tm_year;//年自 1970 年以来
	int tm_wday;// 星期 Sunday = 0
	} tm_rtc;
	tm_rtc currUtcTime;

返回值:	无
NOTE:	您可以按以下方式调用此函数 tm_rtc currUtcTime; sAPI_GetSysLocalTime (&currUtcTime);

2.5 sAPI_SetSysLocalTime

该接口用于设置系统本地时间。

接口:	void **sAPI_SetSysLocalTime**(char* timeStr);
参数:	[in/out] timeStr yy:年(最后两位数); MM:月 dd:日 hh:小时 mm:分钟 ss:秒
返回值:	无
NOTE:	字符串格式为" yy/MM/dd,hh:mm:ss"

3 变量定义

3.1 SCntpReturnCode

```
typedef enum {
    SC_NTP_OK = 0,   //成功
    SC_NTP_ERROR,    //错误
    SC_NTP_ERROR_INVALID_PARAM,    /非法参数
    SC_NTP_ERROR_TIME_CALCULATED,    //错误的时间计算
    SC_NTP_ERROR_NETWORK_FAIL,    //网络错误
    SC_NTP_ERROR_TIME_ZONE,    //错误的时区
    SC_NTP_ERROR_TIME_OUT,    //超时错误
    SC_NTP_END    //NTP 结束
}SCntpResultType;

typedef SCntpResultType SCntpReturnCode;
```

3.2 SCsysTime_t

```
typedef struct SCsysTime_s {
   int tm_sec;    //秒[0,59]
   int tm_min;    //分[0,59]
   int tm_hour;    //小时[0,23]
   int tm_mday;    //月的某天[1,31]
   int tm_mon;    //年的某月  [1,12]
   int tm_year;    //自 1970 年以来
   int tm_wday;    //将一周定义为 0-6，周日为 0
}SCsysTime_t;

SCsysTime_t currUtcTime;
```

3.3 SCntpOperationType

```
typedef enum {
```

```
    SC_NTP_OP_SET,    //设置
    SC_NTP_OP_GET,    //获取时间
    SC_NTP_OP_EXC,    //执行命令
}SCntpOperationType;
```

3.4 SChtpResultType

```
typedef enum
{
  SC_HTP_OK = 0,    //返回正常
  SC_HTP_ERROR,    //返回错误
  SC_HTP_UNKNOWN_ERROR,    //未知错误
  SC_HTP_INVALID_PARAM,    //无效参数
  SC_HTP_ BAD_DATETIME_GOT,    //获取时间错误
  SC_HTP_NETWORK_ERROR,    //网络错误
  SC_HTP_END    //HTP 结束
}SChtpResultType;
```

3.5 SCntpOperationType

```
typedef enum {
    SC_HTP_OP_SET = 0,    //设置
    SC_HTP_OP_GET,    //获取
}SChtpOperationType;
```

3.6 sMsgQRef magQ_urc

```
phtpSesson->htp_magQ = magQ_urc;

typedef struct htpSessionInfoTag
{
    htpServersInfoTag htpServersInfo[MAX_HTP_SERVER_COUNT];    // htp 服务器信息
    int htpServersCnt;    //htp 服务器计数
    int currentIndex;    //当前值的索引
    OSAMsgQRef htp_magQ;
    int sock_fd;    //监听套接字
    struct sockaddr_in serverAddr;    //服务器地址
```

```
    SChtpResultType resultCode;     //结果码
}htpSessionInfoTag;
```

4Example

4.1 sAPI_HtpSrvConfig

Examples

```
sAPI_HtpSrvConfig(SC_HTP_OP_SET, NULL, "ADD", "www.baidu.com", 80, 1, NULL, 0);
sAPI_HtpSrvConfig(SC_HTP_OP_GET, buff, NULL, NULL, 0, 0, NULL, 0);
```

4.2 sAPI_HtpUpdate

Examples

```
sAPI_HtpUpdate(urc_htp_msgq);
```

4.3 sAPI_NtpUpdate

Examples

```
sAPI_NtpUpdate(SC_NTP_OP_SET, "120.25.108.11", 32, NULL);
sAPI_NtpUpdate(SC_NTP_OP_EXC, NULL, 0, urc_ntp_msgq);
```

4.4 sAPI_GetSysLocalTime

Examples

```
SCsysTime_t currUtcTime;
sAPI_GetSysLocalTime(&currUtcTime);
```

4.5 sAPI_SetSysLocalTime

Examples

```
char *timeStr="14/01/01, 02:14:36";
sAPI_SetSysLocalTime(timeStr);
```

5Demo

5.1 Htp

```
/**
  ******************************************************************************
  * @file       demo_htp.c
  * @author    SIMCom OpenSDK Team
  * @brief     Source file of htp operation, HTP can be used to update time over HTTP.
  ******************************************************************************
  * @attention
  *
  * Copyright (c) 2022 SIMCom Wireless.
  * All rights reserved.
  *
  ******************************************************************************
  */
/* Includes ------------------------------------------------------------------*/
#ifdef FEATURE_SIMCOM_HTP
#include "string.h"
#include "stdlib.h"
#include "stdio.h"
#include "simcom_os.h"
#include "simcom_htp_client.h"
#include "simcom_common.h"
#include "simcom_debug.h"
#include "simcom_uart.h"

typedef enum{
    SC_HTP_DEMO_SRVCONFIG           = 1,
    SC_HTP_DEMO_UPDATE          = 2,
    SC_HTP_DEMO_MAX             = 99
}SC_NTP_DEMO_TYPE;

extern sMsgQRef simcomUI_msgq;
extern void PrintfOptionMenu(char* options_list[], int array_size);
extern void PrintfResp(char* format);
sMsgQRef htpUIResp_msgq;
```

```
/**
  * @brief   HTP demo
  * @param   void
  * @note    Need to configure HTTP server before start, here with www.baidu.com as example
  * @retval void
  */
void HtpDemo(void)
{
    UINT32 ret = 0;
    char buff[220]={0};
    UINT32 true = 0;
    SIM_MSG_T optionMsg ={0,0,0,NULL};
    UINT32 opt = 0;
    char *resp = NULL;
    char *note = "\r\nPlease select an option to test from the items listed below.\r\n";
    char *options_list[] = {
        "1. Config server ",
        "2. Update",
        "99. Back",
    };
    while(1)
    {
        SIM_MSG_T htp_result = {SC_SRV_NONE, -1, 0, NULL};
        PrintfResp(note);
        PrintfOptionMenu(options_list,sizeof(options_list)/sizeof(options_list[0]));
        sAPI_MsgQRecv(simcomUI_msgq,&optionMsg,SC_SUSPEND);
        if(SRV_UART != optionMsg.msg_id)
        {
            sAPI_Debug("[HTP] %s,msg_id is error!!",__func__);
            break;
        }
        sAPI_Debug("[HTP] arg3 = [%s]",optionMsg.arg3);
        opt = atoi(optionMsg.arg3);
        sAPI_Free(optionMsg.arg3);
        switch(opt)
        {
            case SC_HTP_DEMO_SRVCONFIG:
            {
                sAPI_Debug("[HTP] Htp server config!");
                if(NULL == htpUIResp_msgq)
                {
                    SC_STATUS status;
                    status = sAPI_MsgQCreate(&htpUIResp_msgq, "htpUIResp_msgq", sizeof(SIM_MSG_T), 4, SC_FIFO);
                    if(SC_SUCCESS != status)
                    {
```

```
                    sAPI_Debug("[HTP] msgQ create fail");
                    resp = "\r\nHTP Fail!\r\n";
                    sAPI_UartWrite(SC_UART,(UINT8*)resp,strlen(resp));
                    break;
                }
            }
            ret = sAPI_HtpSrvConfig(SC_HTP_OP_SET, NULL, "ADD", "www.baidu.com", 80, 1, NULL, 0);
//Unavailable addr may cause long time suspend,such as google
            sAPI_Debug("[HTP]   func[%s] line[%d] ret[%d]", __FUNCTION__,__LINE__,ret);

            ret = sAPI_HtpSrvConfig(SC_HTP_OP_SET, NULL, "ADD", "www.52im.net", 80, 1, NULL, 0);
            sAPI_Debug("[HTP]   func[%s] line[%d] ret[%d]", __FUNCTION__,__LINE__,ret);

            ret = sAPI_HtpSrvConfig(SC_HTP_OP_GET, buff, NULL, NULL, 0, 0, NULL, 0);
            sAPI_Debug("[HTP]   func[%s] line[%d] return_string[%s], ret[%d]", __FUNCTION__,__LINE__,buff,ret);
            if(SC_HTP_OK == ret)
            {
                true = 1;
                sAPI_Debug("[HTP]   CONFIG SERVER SUCCESSFUL");
                PrintfResp("\r\nHTP Config Server Successful!\r\n");
                break;
            }
            else
            {
                sAPI_Debug("[HTP]   CONFIG SERVER ERROR,ERROR CODE = [%d]",htp_result.arg2);
                PrintfResp("\r\nHTP Config Server Fail!\r\n");
                break;
            }
        }
        case SC_HTP_DEMO_UPDATE:
        {
            if(true) /*Config server successful*/
            {
                sAPI_Debug("[HTP]   update true = %d!",true);
                ret = sAPI_HtpUpdate(htpUIResp_msgq);
                sAPI_Debug("[HTP]   func[%s] line[%d] ret[%d]", __FUNCTION__,__LINE__,ret);
                do
                {
                    sAPI_MsgQRecv(htpUIResp_msgq, &htp_result, SC_SUSPEND);
                    if(SC_SRV_HTP != htp_result.msg_id )                    //wrong msg received
                    {
                        sAPI_Debug("[HTP] htp_result.msg_id = [%d]",htp_result.msg_id);
                        htp_result.msg_id = SC_SRV_NONE;                    //para reset
                        htp_result.arg1 = -1;                               //the result code
                        htp_result.arg3 = NULL;                             //parameter - arg3 should be NULL for msg recv
                        continue;
```

```
                        }
                        if(SC_HTP_OK == htp_result.arg1)                    //it means update succeed
                        {
                            sAPI_Debug("[HTP] successfully update time! ");
                            PrintfResp("\r\nHTP Update Time Successful!\r\n");
                        }
                        else
                        {
                            sAPI_Debug("[HTP] failed to update time! result code = %d",    htp_result.arg1);
                            PrintfResp("\r\nHTP Update Time Failed!\r\n");
                        }
                        break;
                    }while(1);
                }
                else
                {
                    sAPI_Debug("[HTP] failed to update time! result code = %d",    htp_result.arg1);
                    PrintfResp("\r\nHTP Update Time Failed! Please config server first!\r\n");
                }
                break;
            }
            case SC_HTP_DEMO_MAX:
            {
                sAPI_Debug("[HTP] Return to the previous menu!");
                PrintfResp("\r\nReturn to the previous menu!\r\n");
                memset(&htp_result,0,sizeof(htp_result));
                return;
            }
            default :
                break;
        }
    }
}

#endif
```

5.2 Ntp

```
/**
  ******************************************************************************
  * @file       demo_ntp.c
  * @author    SIMCom OpenSDK Team
  * @brief     Source file of ntp demo operation.
  ******************************************************************************
  * @attention
  *
  * Copyright (c) 2022 SIMCom Wireless.
  * All rights reserved.
  *
  ******************************************************************************
  */
/* Includes ------------------------------------------------------------------*/
#ifdef FEATURE_SIMCOM_NTP
#include "string.h"
#include "stdlib.h"
#include "stdio.h"
#include "simcom_os.h"
#include "simcom_ntp_client.h"
#include "simcom_common.h"
#include "simcom_debug.h"
#include "simcom_uart.h"

typedef enum{
    SC_NTP_DEMO_UPDATE              = 1,
    SC_NTP_DEMO_MAX                 = 99
}SC_NTP_DEMO_TYPE;

extern sMsgQRef simcomUI_msgq;
extern void PrintfOptionMenu(char* options_list[], int array_size);
extern void PrintfResp(char* format);
sMsgQRef ntpUIResp_msgq;

/**
  * @brief    NTP Demo operation
  * @param    void
  * @note     Please set NTP server accordingly.
  * @retval void
  */
void NtpDemo(void)
{
```

```
UINT32 ret = 0;
SCsysTime_t currUtcTime;
char buff[220]={0};

SIM_MSG_T optionMsg ={0,0,0,NULL};
UINT32 opt = 0;
char *resp = NULL;
char *note = "\r\nPlease select an option to test from the items listed below.\r\n";
char *options_list[] = {
    "1. Update",
    "99. Back",
};

while(1)
{
    SIM_MSG_T ntp_result = {SC_SRV_NONE, -1, 0, NULL};
    PrintfResp(note);
    PrintfOptionMenu(options_list,sizeof(options_list)/sizeof(options_list[0]));
    sAPI_MsgQRecv(simcomUI_msgq,&optionMsg,SC_SUSPEND);
    if(SRV_UART != optionMsg.msg_id)
    {
        sAPI_Debug("%s,msg_id is error!!",__func__);
        break;
    }

    sAPI_Debug("arg3 = [%s]",optionMsg.arg3);
    opt = atoi(optionMsg.arg3);
    sAPI_Free(optionMsg.arg3);

    switch(opt)
    {
        case SC_NTP_DEMO_UPDATE:
        {
            if(NULL == ntpUIResp_msgq)
            {
                SC_STATUS status;
                status = sAPI_MsgQCreate(&ntpUIResp_msgq, "htpUIResp_msgq", sizeof(SIM_MSG_T), 4, SC_FIFO);
                if(SC_SUCCESS != status)
                {
                    sAPI_Debug("[CNTP]msgQ create fail");
                    resp = "\r\nNTP Update Fail!\r\n";
                    sAPI_UartWrite(SC_UART,(UINT8*)resp,strlen(resp));
                    break;
                }
            }
```

```
    memset(&currUtcTime,0,sizeof(currUtcTime));

    sAPI_GetSysLocalTime(&currUtcTime);
    sAPI_Debug("[CNTP] sAPI_GetSysLocalTime %d - %d - %d
- %d : %d : %d    %d",currUtcTime.tm_year,currUtcTime.tm_mon,currUtcTime.tm_mday,
      currUtcTime.tm_hour,currUtcTime.tm_min,currUtcTime.tm_sec,currUtcTime.tm_wday);
    ret = sAPI_NtpUpdate(SC_NTP_OP_SET, "ntp3.aliyun.com", 32, NULL);                    //Unavailable
addr may cause long time suspend
    sAPI_Debug("[CNTP] func[%s] line[%d] ret[%d]", __FUNCTION__,__LINE__,ret);

    ret = sAPI_NtpUpdate(SC_NTP_OP_GET, buff, 0, NULL);
    sAPI_Debug("[CNTP] func[%s] line[%d] ret[%d] buff[%s]", __FUNCTION__,__LINE__,ret, buff);

    ret = sAPI_NtpUpdate(SC_NTP_OP_EXC, NULL, 0, ntpUIResp_msgq);
    sAPI_Debug("[CNTP] func[%s] line[%d] ret[%d] ", __FUNCTION__,__LINE__,ret );
    do
    {
        sAPI_MsgQRecv(ntpUIResp_msgq, &ntp_result, SC_SUSPEND);

        if(SC_SRV_NTP != ntp_result.msg_id )                    //wrong msg received
        {
            sAPI_Debug("[CNTP] ntp_result.msg_id =[%d], ntp_result.msg_id ");
            ntp_result.msg_id = SC_SRV_NONE;                    //para reset
            ntp_result.arg1 = -1;
            ntp_result.arg3 = NULL;
            continue;
        }
        if(SC_NTP_OK == ntp_result.arg1)                        //it means update succeed
        {
            sAPI_Debug("[CNTP] successfully update time! ");
            PrintfResp("\r\nNTP Update Time Successful!\r\n");
            break;
        }
        else
        {
            sAPI_Debug("[CNTP] failed to update time! result code: %d", ntp_result.arg1);
            PrintfResp("\r\nNTP Update Time Failed!\r\n");
            break;
        }
    }while(1);

    memset(&currUtcTime,0,sizeof(currUtcTime));
    sAPI_GetSysLocalTime(&currUtcTime);
    sAPI_Debug("[CNTP] sAPI_GetSysLocalTime %d - %d - %d
- %d : %d : %d    %d",currUtcTime.tm_year,currUtcTime.tm_mon,currUtcTime.tm_mday,
        currUtcTime.tm_hour,currUtcTime.tm_min,currUtcTime.tm_sec,currUtcTime.tm_wday);
```

```
                    break;
                }

                case SC_NTP_DEMO_MAX:
                {
                    sAPI_Debug("Return to the previous menu!");
                    PrintfResp("\r\nReturn to the previous menu!\r\n");
                    memset(&ntp_result,0,sizeof(ntp_result));

                    return;
                }

                default :
                    break;
            }
        }
    }
#endif
```