

OpENER STM32 Port

Requirements

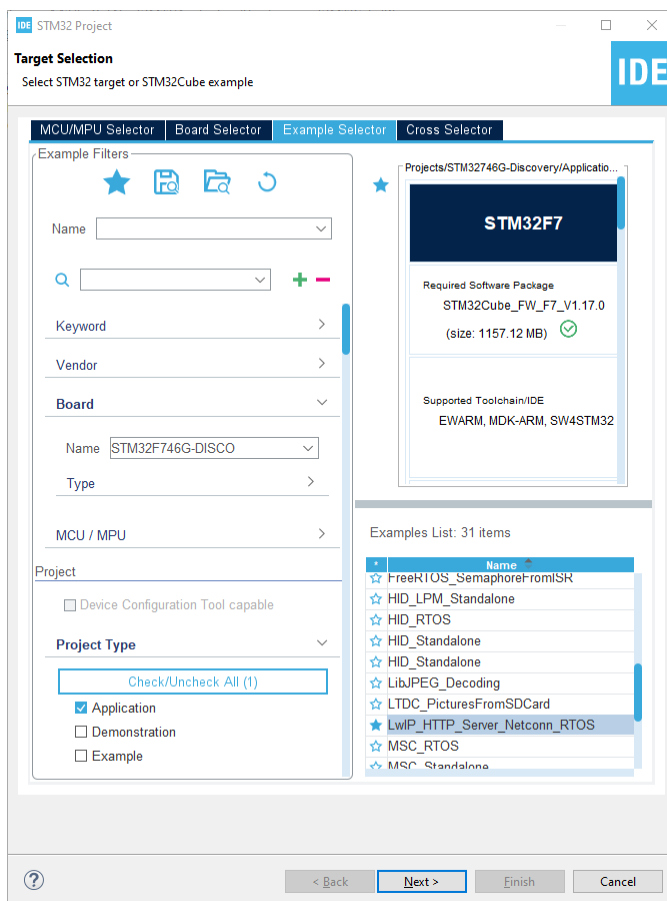
You will need to have the following installed:

- [STM32CubeIDE](#) Integrated Development Environment for STM32 (based on Eclipse)
- STM32 target e.g., the board [32F746GDISCOVERY](#)

Compile for STM32

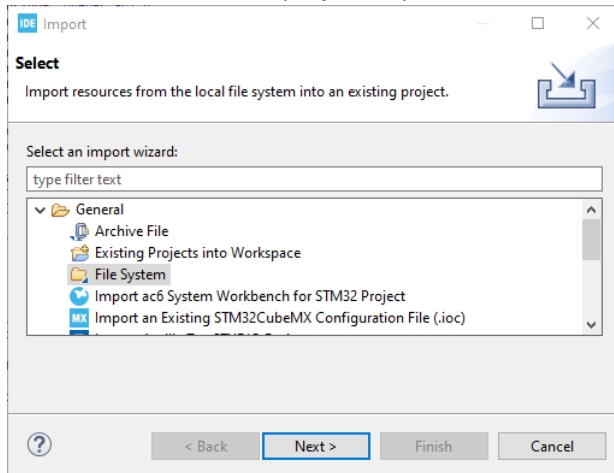
Setup of the project

1. Make sure all the needed tools are available.
2. Open the STM32CubeIDE
3. Create new Project: File - New - STM32 Project
4. Go to Example Selector
5. Select Board e.g.: **STM32F746G-DISCO**
6. Select Project Type: **Application**
7. Select: **LwIP_HTTP_Server_Netconn_RTOS**:

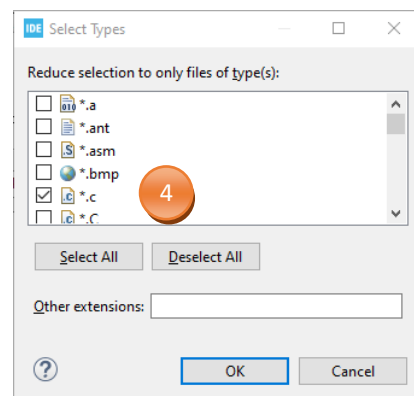
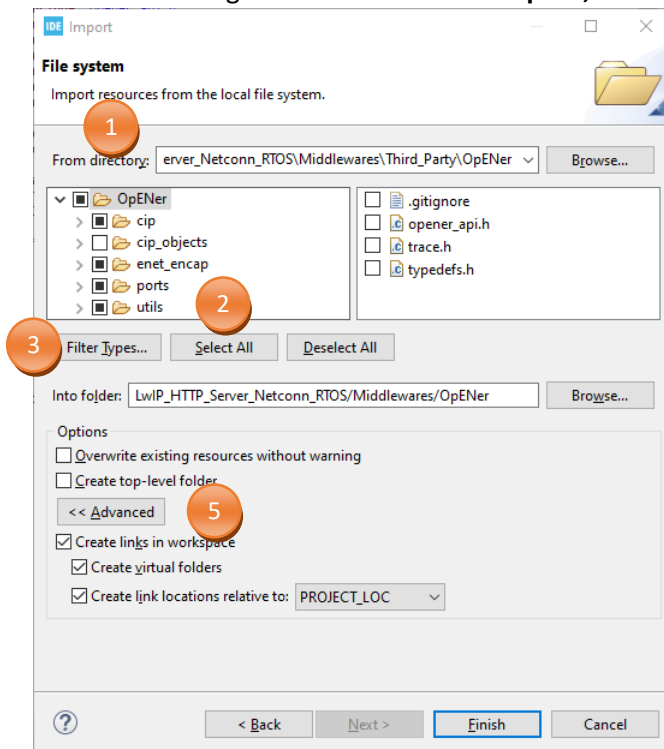


8. Create project
9. In file system create new folder **OpENER** in folder Middlewares

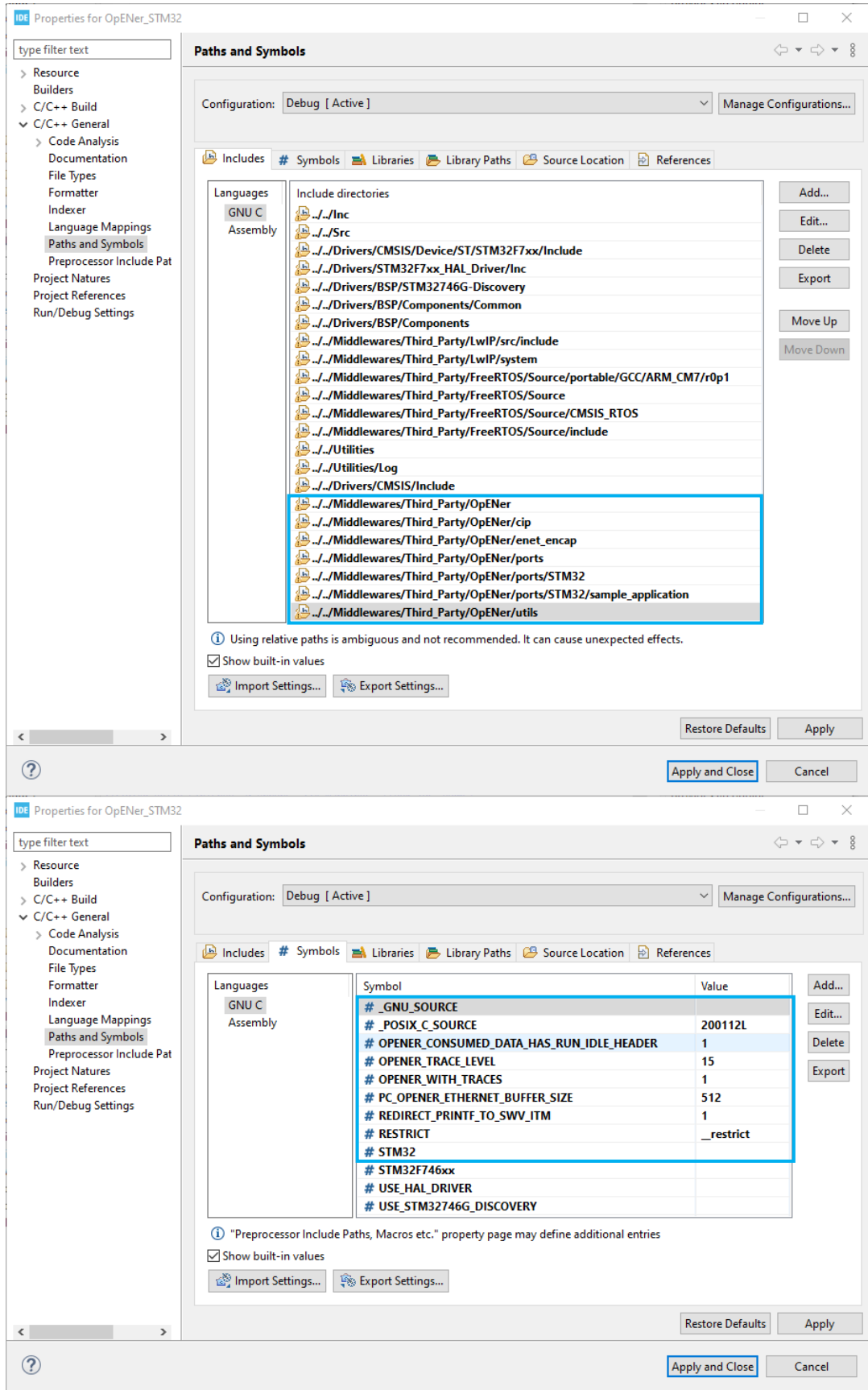
10. Copy the from source folder of OpENER (...\\source\\src) to: ...\\Middlewares\\Third_Party all files except the following folders and files from folder **ports**:
 - a. Folders of other ports than **STM32** (MINGW, POSIX and WIN32)
 - b. Folder **nvddata**
 - c. **devicedata.h.in**, the related file is **devicedata.h** in the port folder STM32
 - d. Cmake files
11. In the IDE select go to the folder **Middlewares** and create subfolder **OpENER**
12. Add source files to the project: OpENER Context menu – Import – General – File System



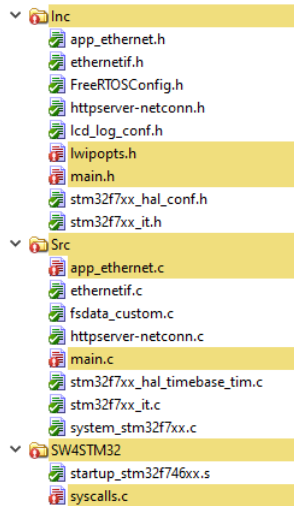
13. From directory: ...\\Middlewares\\Third_Party\\OpENER
14. Select all Files and Filter Types - Select *.c
15. Advanced settings - **Create links in workspace, Create virtual..., Create link locations...:** all checked



16. Add these additional include paths and symbols manually or
Import Settings from: **OpENER STM32 Paths and Symbols.xml**



17. The following files of the project **LwIP_HTTP_Server_Netconn_RTOS** must be modified.
The details are described in the chapter “Integration of OpENER”



18. Build Project
19. Run OpENER on STM32

Integration of OpENER

Apply the patch **LwIP_HTTP_Server_Netconn_RTOS_OpENER.patch** or edit the files as described here.

[lwipopts.h](#)

Add these options:

Line 74:

```
/* MEMP_NUM_NETCONN: the number of struct netconns.  
   (only needed if you use the sequential API, like api_lib.c) */  
#define MEMP_NUM_NETCONN      12
```

Line 114:

```
/* ----- IGMP options ----- */  
#define LWIP_IGMP              1
```

Line 135:

```
/* ----- Netif options ----- */  
#define LWIP_NETIF_HOSTNAME    1
```

Line 203:

```
#define LWIP_SOCKET            1  
/**  
 * SO_REUSE==1: Enable SO_REUSEADDR option.  
 */  
#define SO_REUSE               1
```

[main.h](#)

Line 34, this option is not used anymore, can be commented out.

```
//#define USE_DHCP             // not used, replaced by LWIP_DHCP
```

Line 37, adapt the static IP address to your requirements.

```
/*Static IP ADDRESS*/  
...  
/*NETMASK*/  
...  
/*Gateway Address*/
```

[app_ethernet.c](#)

Line 31, include OpENER:

```
// for OpENER  
#include "opener.h"
```

Line 61, if DHCP is not used, call **opener_init()** after static IP address is assigned:

```
#if LWIP_DHCP  
    /* Update DHCP state machine */  
    DHCP_state = DHCP_START;  
#elif defined(USE_LCD)  
    uint8_t iptxt[20];  
    sprintf((char *)iptxt, "%s", ip4addr_ntoa(netif_ip4_addr(netif)));  
    LCD_UsrLog ("Static IP address: %s\n", iptxt);  
    /* Start Ethernet/IP Stack (OpENER) */  
    opener_init(netif);
```

Line 132, if DHCP is used, call **opener_init()** after dynamic IP address is assigned:

```
#ifndef USE_LCD  
    sprintf((char *)iptxt, "%s", ip4addr_ntoa(netif_ip4_addr(netif)));  
    LCD_UsrLog ("IP address assigned by a DHCP server: %s\n", iptxt);  
#else  
    BSP_LED_On(LED1);  
    BSP_LED_Off(LED2);
```

```
#endif
/* Start Ethernet/IP Stack (OpENer) */
opener_init(netif);
```

Line 158, if DHCP is used, call **opener_init()** after DHCP timeout when a static IP address is assigned:

```
#ifdef USE_LCD
    sprintf((char *)iptxt, "%s", ip4addr_ntoa(netif_ip4_addr(netif)));
    LCD_UsrLog ("DHCP Timeout !! \n");
    LCD_UsrLog ("Static IP address: %s\n", iptxt);
#else
    BSP_LED_On(LED1);
    BSP_LED_Off(LED2);
#endif
/* Start Ethernet/IP Stack (OpENer) */
opener_init(netif);
```

main.c

Line 74, if the timer should be stopped in single step debug, add these statements:

```
/* For single step debug, e.g. timers with interrupts need to be stopped in Halt */
HAL_DBGMCU_EnableDBGStandbyMode();
HAL_DBGMCU_EnableDBGStopMode();
__HAL_DBGMCU_FREEZE_TIM6();
```

Line 147, add the hostname, used in OpENer, the default from lwIP is "lwIP"

```
/* Define the hostname, is also used by OpENer */
netif_set_hostname(&gnetif, "STM32");
```

Line 190, adapt the header text displayed on the LCD:

```
LCD_LOG_SetHeader((uint8_t *) "Webserver Application Netconn API & OpENer");
```

syscalls.c

If the the output of printf shall redirected to the Serial Wire Viewer (SWV) add these statements:

Line 18:

```
#include <stm32f7xx_hal.h>
```

Line 102:

```
#if REDIRECT_PRINTF_TO_SWV_ITM
attribute__((weak)) int _write(int file, char *ptr, int len) {
    int DataIdx;
    for (DataIdx = 0; DataIdx < len; DataIdx++) {
        ITM_SendChar(*ptr++);
    }
    return len;
}
#else // standard output
int _write(int file, char *ptr, int len)
{
    int DataIdx;

    for (DataIdx = 0; DataIdx < len; DataIdx++)
    {
        __io_putchar( *ptr++ );
    }
    return len;
}
#endif // standard output
```