Store web pages externally step by step

Web pages can be stored on any media accessible via MFS, such as USB memory stick, SD card. It is also possible to use FFS and store web pages on NAND flash. The following example is a step by step guide on how MQX can use nandflash on TWR-K70F120M, we start with the httpsrv demo.

1. Add FTP server and nandflash flush support
   This demo starts with default HTTPSRV example, Shell console is enabled in the default httpdsrv.c. Ping, ipconfig and help commands are enabled by default. We add ftpsrv application to support FTP protocol , and nandflash support.

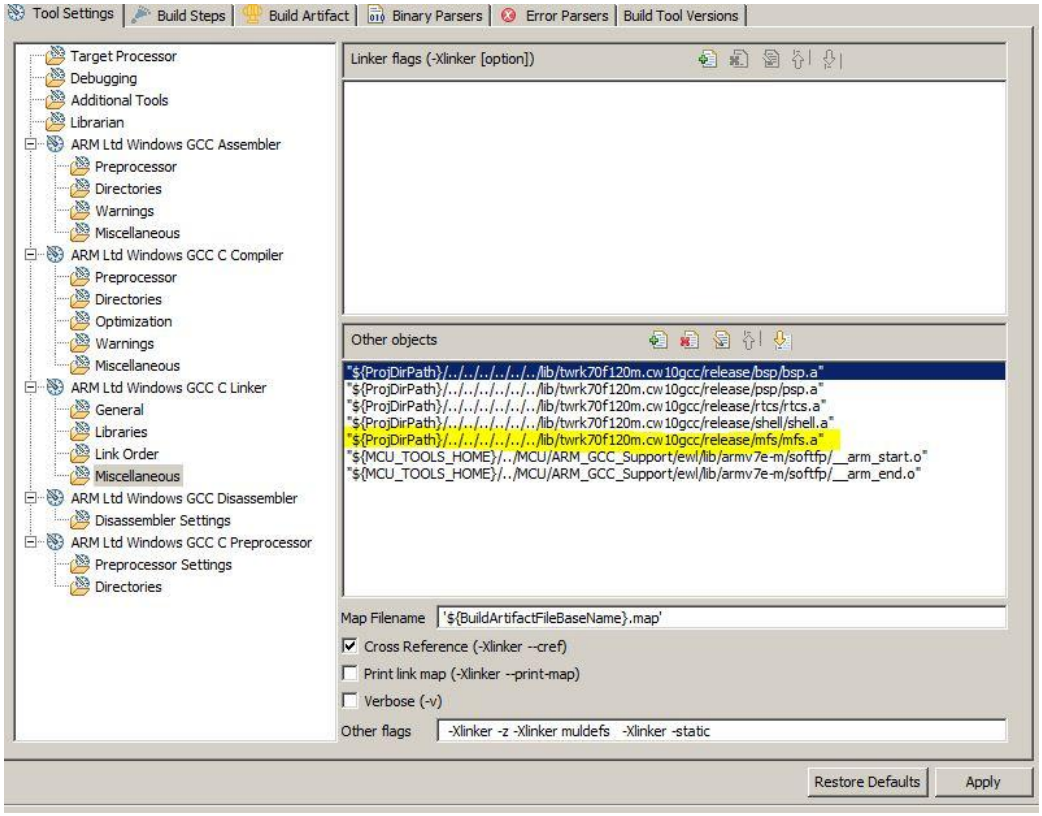
```
const SHELL_COMMAND_STRUCT Shell_commands[] = {
   /* RTCS commands */
   { "ftpsrv",     Shell_ftpsrv },
   { "help",       Shell_help },
   { "ipconfig",   Shell_ipconfig },
#if RTCSCFG_ENABLE_ICMP
   { "ping",       Shell_ping },
#endif
   { "?",          Shell_command_list },
   { "ffsflush",   Shell_ffs_flush },
   { NULL,         NULL }
};
```

2. Allow RTCS for more sockets
   We will use two TCP servers. There is one socket need for HTTPSRV listening socket, second listening socket for FTP server. One socket for each connected client.

```
¶
····/* Init RTCS */¤¶
····_RTCSPCB_init = 4;¤¶
····_RTCSPCB_grow = 2;¤¶
····_RTCSPCB_max = 20;¤¶
····_RTCS_msgpool_init = 4;¤¶
····_RTCS_msgpool_grow = 2;¤¶
····_RTCS_msgpool_max = 20;¤¶
····_RTCS_socket_part_init = 4;¤¶
····_RTCS_socket_part_grow = 2;¤¶
····_RTCS_socket_part_max = 20;¤¶
····¤¶
····_RTCSTASK_stacksize = 4500;¤¶
····¤¶
····error = RTCS_create();¤¶
····if (error != RTCS_OK)¤¶
····{¤¶
········printf("RTCS failed to initialize, error = 0x
········_task_block();¤¶
····}¤¶
¶
····_IP_forward = FALSE;¤¶
```

3. Add MFS library to the HTTPSRV build project
   By default, MFS library is not enabled in the HTTPSRV project. It uses only TFS for web pages storage. We add path to MFS library into HTTPSRV build project

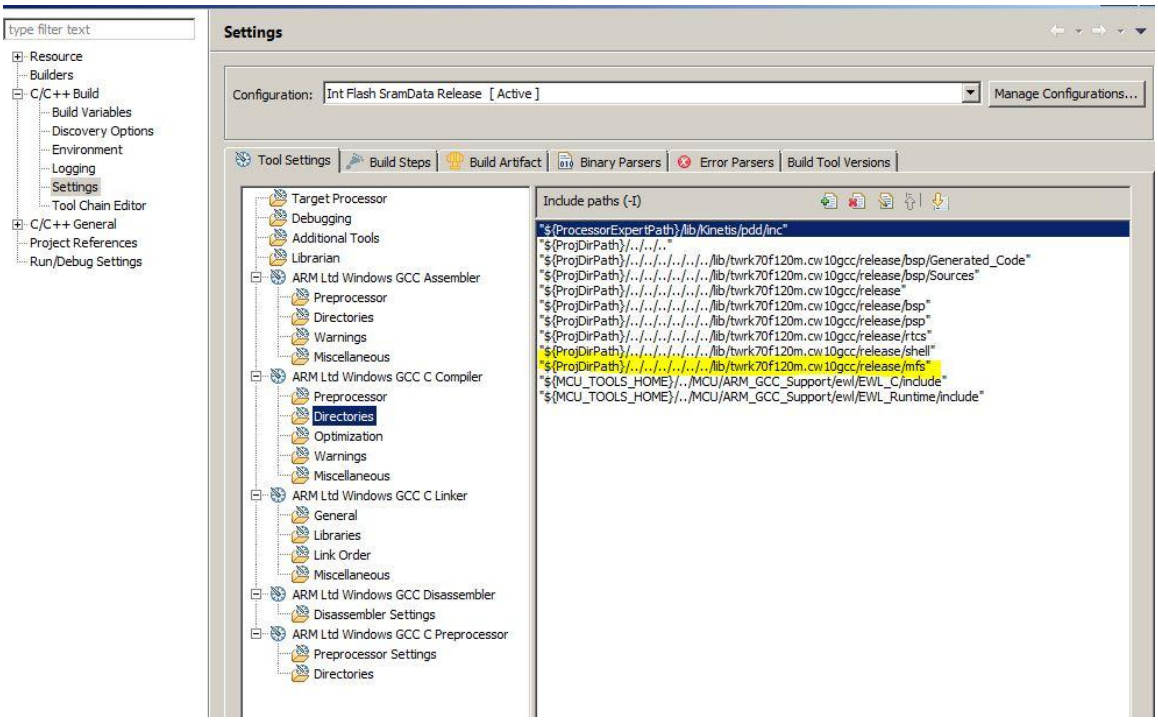
```
/mqx-repo/lib/twrk70f120m.cw10gcc/release/rtcs/rtcs.a(ftpsrv_cmd.o): In function `ftpsrv_list':
/mqx-repo/rtcs/source/apps/ftpsrv_cmd.c:313: undefined reference to `_io_mfs_dir_open'
/mqx-repo/rtcs/source/apps/ftpsrv_cmd.c:329: undefined reference to `_io_mfs_dir_close'
/mqx-repo/rtcs/source/apps/ftpsrv_cmd.c:334: undefined reference to `_io_mfs_dir_read'
/mqx-repo/rtcs/source/apps/ftpsrv_cmd.c:352: undefined reference to `_io_mfs_dir_close'
/mqx-repo/lib/twrk70f120m.cw10gcc/release/rtcs/rtcs.a(ftpsrv_cmd.o): In function `ftpsrv_cwd':
/mqx-repo/rtcs/source/apps/ftpsrv_cmd.c:120: undefined reference to `_io_mfs_dir_open'
/mqx-repo/rtcs/source/apps/ftpsrv_cmd.c:127: undefined reference to `_io_mfs_dir_close'
* [C:/ccwork/mqxdev/mqx-repo/rtcs/examples/httpsrv/build/cw10gcc/httpsrv_twrk70f120m/Int Flash SramData Release/httpsrv.elf] Error 1
```



4. Add FFS support to the HTTPSRV build project
   Ffs library is not enabled by default. We need to add path into HTTPSRV build library. Install FFS driver and open it.

5. Resolve build errors
   When we try to build now the HTTPSRV example, we have compile errors. This is because by default the HTTPSRV example does not include MFS and FFS library head files. We need to add a path for MFS and FFS

```
"C:/Freescale/CW MCU v10.6/Cross_Tools/arm-none-eabi-gcc-4_7_3/bin/arm-none-eabi-gcc" "C:/ccwork/mqxdev/mqx-repo/rtcs/examples/httpsrv/httpsrv_main.c" @"Source/httpsrv_m
-o"Source/httpsrv_main.o"
C:/ccwork/mqxdev/mqx-repo/rtcs/examples/httpsrv/httpsrv_main.c: In function 'Ram_disk_start':
C:/ccwork/mqxdev/mqx-repo/rtcs/examples/httpsrv/httpsrv_main.c:145:5: warning: implicit declaration of function '_io_mfs_install' [-Wimplicit-function-declaration]
C:/ccwork/mqxdev/mqx-repo/rtcs/examples/httpsrv/httpsrv_main.c:149:23: error: 'MFS_NO_ERROR' undeclared (first use in this function)
C:/ccwork/mqxdev/mqx-repo/rtcs/examples/httpsrv/httpsrv_main.c:168:56: error: 'MFS_NOT_A_DOS_DISK' undeclared (first use in this function)
C:/ccwork/mqxdev/mqx-repo/rtcs/examples/httpsrv/httpsrv_main.c:170:9: warning: implicit declaration of function 'MFS_Error_text' [-Wimplicit-function-declaration]
C:/ccwork/mqxdev/mqx-repo/rtcs/examples/httpsrv/httpsrv_main.c: In function 'Ramdisk_format':
C:/ccwork/mqxdev/mqx-repo/rtcs/examples/httpsrv/httpsrv_main.c:201:36: error: 'IO_IOCTL_DEFAULT_FORMAT' undeclared (first use in this function)
C:/ccwork/mqxdev/mqx-repo/rtcs/examples/httpsrv/httpsrv_main.c:208:27: error: 'IO_IOCTL_FREE_SPACE' undeclared (first use in this function)
mingw32-make: *** [Source/httpsrv_main.o] Error 1
```

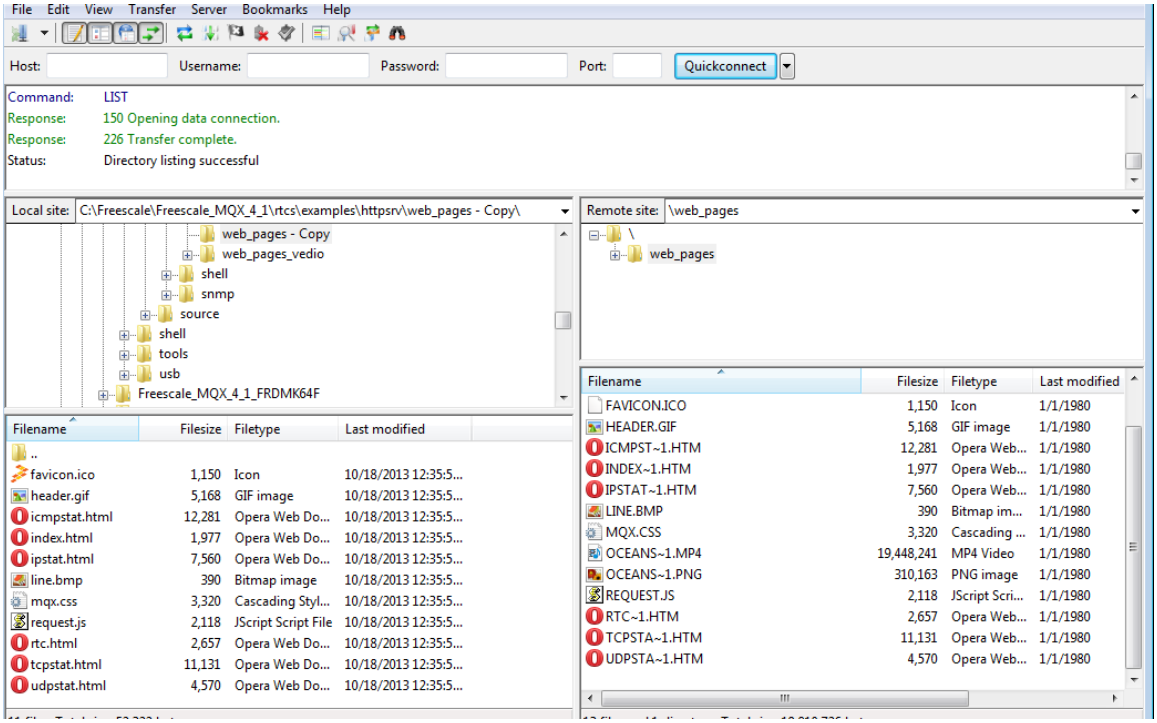


6. Configure HTTPSRV root directory

Web pages of this demo are to be stored in nandflash within web_pages folder. We need to set rootdir and index

```
    }
    #endif

    indexes[i] = (char*) _mem_alloc_zero(sizeof("\\index.html"));
    if (indexes[i] == NULL)
    {
        printf("\n Failed to allocate memory.");
        _task_block();
    }
} ? end for i=0;(i<n_devices)&&(n... ?

/* Install trivial file system. HTTP server pages are stored there. */
error = _io_tfs_install("tfs:", tfs_data);

/* Start HTTP server on each interface */
for (i = 0; i < n_servers; i++)
{
    _mem_zero(&params[i], sizeof(HTTPSRV_PARAM_STRUCT));
    params[i].af = HTTP_INET_AF;   //IPv4, IPv6 or from config.h

    #if RTCSCFG_ENABLE_IP6
    /* Set interface number here. Zero is any. */
    params[i].ipv6_scope_id = HTTP_SCOPE_ID;
    #endif

    sprintf(indexes[i], "\\index.html");
    params[i].root_dir = "a:\\web_pages";
    params[i].index_page = indexes[i];
    params[i].auth_table = auth_realms;
```

Before we can view the web page in browser, we need to copy web pages to nand flash. This is the purpose of FTP server, as one example how to copy files.



If you wish to be sure the web pages survive power cycle, make sure you flush the FFS file system after “web_pages” are copied.

7. Build, Download, Execute
   As hardware we use TWR-K70F120M and TWR-SER. After you download the executable and run, start FTP server by issuing command “ftpsrv start” on serial console; start FTP client; copy over files from PC host to TWR nandflash. Keep the folder structure of the copied files; now open your web browser (support for HTML5 is required); open Url http://192.168.1.202