SCM1612
Wi-Fi 6 and BLE 5 Low-Power SoC

mDNS Development Guide

Revision 0.1
Date 2024-3-13

Contact Information
Senscomm Semiconductor (www.senscomm.com)
Room 303, International Building, West 2 Suzhou Avenue,
SIP, Suzhou, China
For sales or technical support, please send email to
info@senscomm.com

Disclaimer and Notice

Version History

Version	Date	Description
0.1	2024-3-13	Initial draft

Table of Contents

1 Introduction

This document serves as a guide that helps implementing applications that requires running an mDNS.

1.1 Overview

The SCM1612 SDK uses the lwIP's mdns port:

- API and CLI
  - Located in: lib/net/mdns

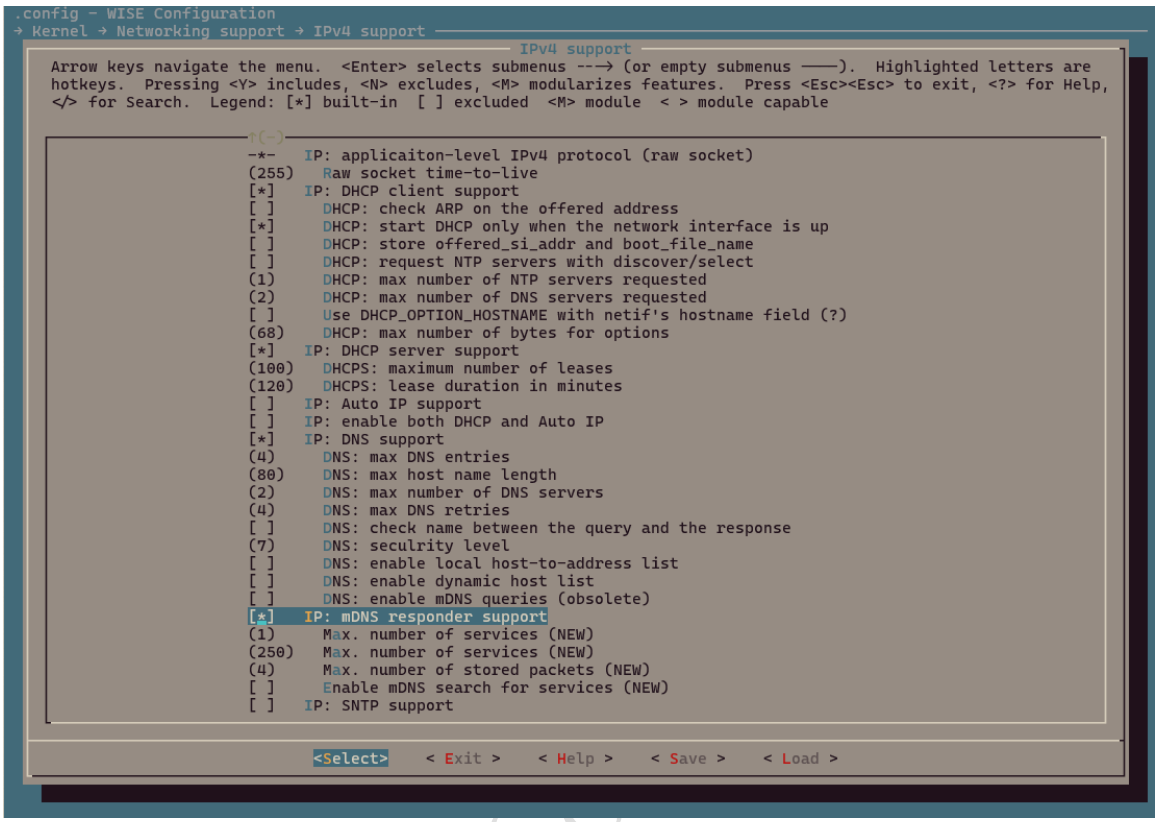
SCM1612's mDNS module can be set up and act as an mDNS responder as well as initiate a service query as an mDNS querier as an optional feature.

1.2 Build

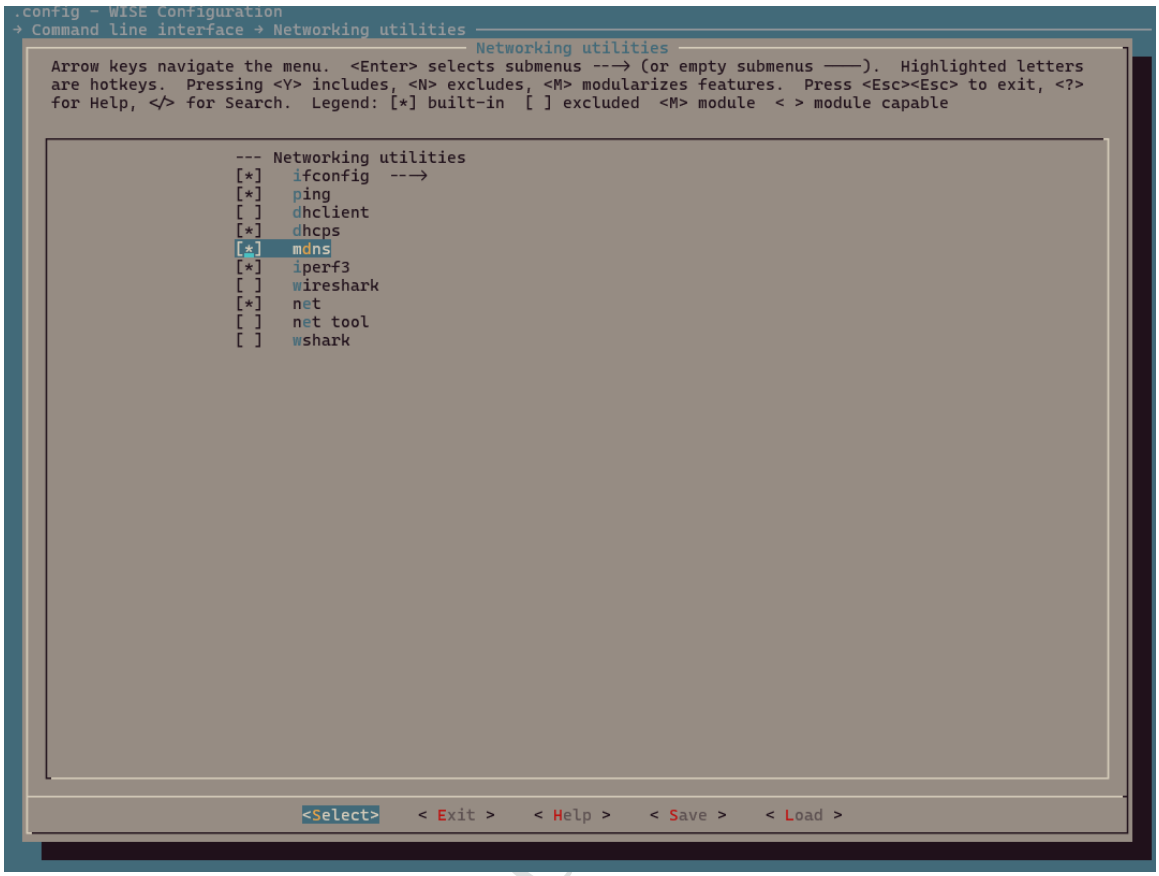
To use mDNS APIs and CLI, user should enable corresponding feature in build configuration.

```
$ make scm1612s_defconfig
$ make menuconfig
```

Select Kernel -> Networking support -> IPv4 support -> IP: mDNS responder support



Select Command Line Interface -> Networking utilities -> mdns



If we want to resolve local hostnames ending with .local, we should also: Select Kernel->Networking support->Enable lwIP TCP/IP stack->IPv4 support->NDS support->enable mDNS queries (obsolete)

```
--- IP: DNS support
(4)   DNS: max DNS entries
(80)  DNS: max host name length
(2)   DNS: max number of DNS servers
(4)   DNS: max DNS retries
[ ]   DNS: check name between the query and the response
(7)   DNS: seculrity level
[ ]   DNS: enable local host-to-address list
[ ]   DNS: enable dynamic host list
[*]   DNS: enable mDNS queries (obsolete)
```

Exit & Save.

Build wise-mcuboot.bin.

$ make

Please refer to the SDK_Getting_Started_Guide to download the image and run it on a SCM1612 EVK.
You will be able to confirm that relevant CLI commands are available as follows.

2 API

mDNS API provides the following set of functions to initialize and set up a mDNS server and send queries to search services available at neighboring devices.

- mdns_resp_init
- mdns_resp_add_netif
- mdns_resp_remove_netif
- mdns_resp_add_service
- mdns_resp_del_service
- mdns_search_service
- mdns_search_stop

It is important to note that these functions must be protected by lwIP core lock because there are shared resources that will be accessed from lwIP TCP/IP core thread.
Refer to mdns() CLI entry function in mdns.c for reference.

2.1 Initialize and set up mDNS responder

- void mdns_resp_init(void)

  Initiate MDNS responder. Will open UDP sockets on port 5353

  Corresponding CLI is:

```
WISE 2018.02+ (Mar 13 2024 - 08:53:14 -0700)
Hello world!
$
$ mdns
Unknown interface ?
Usage: mdns init
  or:  mdns ifname start
  or:  mdns ifname stop
  or:  mdns ifname service add <name> <service> <proto> <port>
  or:  mdns ifname service del <id>
  or:  mdns ifname search browse
  or:  mdns ifname search info <service> <proto>
  or:  mdns ifname search stop <id>
$
```

- err_t mdns_resp_add_netif(struct netif *netif, const char *hostname)

Activate MDNS responder for a network interface.

Parameter	Description
netif	The network interface to activate.
hostname	Name to use. Queries for hostname, .local will be answered with the IP addresses of the netif. The hostname will be copied, the given pointer can be on the stack.

Return
ERR_OK if netif was added, an err_t otherwise

Corresponding CLI is:

- err_t mdns_resp_remove_netif(struct netif *netif)

  Stop responding to MDNS queries on this interface, leave multicast groups, and free the helper structure and any of its services.

Parameter	Description
netif	The network interface to remove.

Return
ERR_OK if netif was removed, an err_t otherwise

Corresponding CLI is:

2.2 Manage services of mDNS responder

- s8_t mdns_resp_add_service(struct netif *netif, const char *name,
  const char *service,
  enum mdns_sd_proto proto, u16_t port,
  service_get_txt_fn_t txt_f, void *txt_data)

Add a service to the selected network interface.

Parameter	Description
netif	The network interface to publish this service on.
name	The name of the service.
service	The service type, like "_http".
proto	The service protocol, DNSSD_PROTO_TCP for TCP (“_tcp”) and DNSSD_PROTO_UDP for others (“_udp”).
port	The port the service listens to.
txt_fn	Callback to get TXT data. Will be called each time a TXT reply is created to allow dynamic replies.
txt_data	Userdata pointer for txt_fn.

Return
Service_id if the service was added to the netif, an err_t otherwise

Corresponding CLI is:

```
WISE 2018.02+ (Mar 13 2024 - 08:53:14 -0700)
Hello world!
$
$ mdns
Unknown interface ?
Usage: mdns init
  or:  mdns ifname start
  or:  mdns ifname stop
  or:  mdns ifname service add <name> <service> <proto> <port>
  or:  mdns ifname service del <id>
  or:  mdns ifname search browse
  or:  mdns ifname search info <service> <proto>
  or:  mdns ifname search stop <id>
$
```

- err_t mdns_resp_del_service(struct netif *netif, u8_t slot)

Delete a service on the selected network interface.

Parameter	
netif	The network interface on which service should be removed.
slot	The service slot number returned by mdns_resp_add_service.

Return
ERR_OK if the service was removed from the netif, an err_t otherwise

Corresponding CLI is:

```
WISE 2018.02+ (Mar 13 2024 - 08:53:14 -0700)
Hello world!
$
$ mdns
Unknown interface ?
Usage: mdns init
  or:  mdns ifname start
  or:  mdns ifname stop
  or:  mdns ifname service add <name> <service> <proto> <port>
  or:  mdns ifname service del <id>
  or:  mdns ifname search browse
  or:  mdns ifname search info <service> <proto>
  or:  mdns ifname search stop <id>
$
```

2.3 Start service query

- err_t mdns_search_service(const char *name, const char *search,
  enum mdns_sd_proto proto,
  struct netif *netif,
  search_result_fn_t result_fn,
  void *arg, u8_t *request_id)

Search a specific service on the network.

Parameter	
name	The name of the service.
service	The service type, like "_http".
proto	The service protocol, DNSSD_PROTO_TCP for TCP ("_tcp") and DNSSD_PROTO_UDP for others ("_udp").
netif	The network interface where to send search request.
result_fn	Callback to send answer received. Will be called for each answer of a response frame matching request sent.
arg	Userdata pointer for result_fn.
request_id	Returned request identifier to allow stop it.

Return
ERR_OK if the search request was created and sent, an err_t otherwise

Corresponding CLI is:

```
WISE 2018.02+ (Mar 13 2024 - 08:53:14 -0700)
Hello world!
$
$ mdns
Unknown interface ?
Usage: mdns init
  or:  mdns ifname start
  or:  mdns ifname stop
  or:  mdns ifname service add <name> <service> <proto> <port>
  or:  mdns ifname service del <id>
  or:  mdns ifname search browse
  or:  mdns ifname search info <service> <proto>
  or:  mdns ifname search stop <id>
$
```

Both CLI commands use this API, but with different parameters.

2.4 Stop service query

- void mdns_search_stop(u8_t request_id)

Stop a search request.

Parameter	
request_id	The search request to stop.

Corresponding CLI is:

```
WISE 2018.02+ (Mar 13 2024 - 08:53:14 -0700)
Hello world!
$
$ mdns
Unknown interface ?
Usage: mdns init
  or:  mdns ifname start
  or:  mdns ifname stop
  or:  mdns ifname service add <name> <service> <proto> <port>
  or:  mdns ifname service del <id>
  or:  mdns ifname search browse
  or:  mdns ifname search info <service> <proto>
  or:  mdns ifname search stop <id>
$
```

3 Demo

There is no dedicated demo application for mDNS.
Instead, mDNS CLI commands which were introduced above can be used to test its functionality.

3.1 Connect to an AP

Wi-Fi STA CLI commands can be used to connect the station interface, i.e., wlan0, to an AP.

Refer to SCM1612_Wi-Fi_Software_Development_Guide for use of Wi-Fi station CLI commands for there is almost always one-to-one correspondence between Wi-Fi API functions and Wi-Fi CLI commands.

```
$ wifi help
wifi sta_cfg <ssid> <auth> <key> <bssid> <pairwise> <hidden ap>
  or:  wifi sta_connect
  or:  wifi sta_disconnect
  or:  wifi sta_get_connect
  or:  wifi sta_set_reconnect <enable> <timeout> <period> <count>
  or:  wifi sta_fast_connect <ssid> <auth> <bssid> <pairwise> <psk> <channel>
  or:  wifi sta_start
  or:  wifi sta_get_psk
  or:  wifi sta_scan
  or:  wifi sta_advance_scan <scan_type> <channel>|<ssid>|<bssid>
  or:  wifi sta_scan_results <max_ap_num>
  or:  wifi sap_start
  or:  wifi sap_stop
  or:  wifi sap_cfg <ssid> <key> <ch> <hidden> <auth> <pairwise>
  or:  wifi sap_beacon <interval>
  or:  wifi sap_dtim <period>
  or:  wifi sap_deauth <sta_mac>
  or:  wifi sap_show
  or:  wifi sap_showsta
  or:  wifi ip_set <ifn> <ip> [nm] [gw]
  or:  wifi dhcp_start/dhcp_stop
  or:  wifi dhcps_start/dhcps_stop
  or:  wifi set keepalive <enable> <interval>
  or:  wifi set powersave <enable> <interval>
  or:  wifi reg_evt_cb

I (44626) SCM_CLI: help    OK (0)
$
$ wifi reg_evt_cb
I (47730) SCM_CLI: reg_evt_cb    OK (0)
$
$ wifi sta_start
I (47796) SCM_CLI: STA_STOP
I (47800) SCM_CLI: ifname: wlan0
I (47800) SCM_CLI: sta_start    OK (0)
$ I (47801) SCM_CLI: STA_START

$ wifi sta_cfg Xiaohu_ASUS 0 0 00:00:00:00:00:00 1 0
I (47833) SCM_CLI: sta_cfg    OK (0)
$
$ wifi sta_connect
I (47855) SCM_CLI: sta_connect    OK (0)
$ wifi dhcp_start
I (47857) SCM_CLI: dhcp_start    OK (0)
$ I (49543) SCM_CLI: STA_CONNECTED
I (49544) SCM_API: AP SSID: Xiaohu_ASUS
I (49544) SCM_API: AP BSSID: 50:eb:f8:19:88:a0
I (49545) SCM_API: AP CH: 11
I (49546) SCM_API: AP RSSI: -28
I (49547) SCM_API: AP Country : AA
I (49547) SCM_API: Status: CONNECTED
I (49568) SCM_CLI: WIFI GOT IP
```

3.2 Start mDNS responder on wlan0

```
I (47857) SCM_CLI: dhcp_start    OK (0)
$ I (49543) SCM_CLI: STA_CONNECTED
I (49544) SCM_API: AP SSID: Xiaohu_ASUS
I (49544) SCM_API: AP BSSID: 50:eb:f8:19:88:a0
I (49545) SCM_API: AP CH: 11
I (49546) SCM_API: AP RSSI: -28
I (49547) SCM_API: AP Country : AA
I (49547) SCM_API: Status: CONNECTED
I (49568) SCM_CLI: WIFI GOT IP

$
$ mdns
Unknown interface ·
Usage: mdns init
  or:  mdns ifname start
  or:  mdns ifname stop
  or:  mdns ifname service add <name> <service> <proto> <port>
  or:  mdns ifname service del <id>
  or:  mdns ifname search browse
  or:  mdns ifname search info <service> <proto>
  or:  mdns ifname search stop <id>
$
$ mdns init
$ mdns wlan0 start
$
$
```

3.3 Add a service

```
$ I (49543) SCM_CLI: STA_CONNECTED
I (49544) SCM_API: AP SSID: Xiaohu_ASUS
I (49544) SCM_API: AP BSSID: 50:eb:f8:19:88:a0
I (49545) SCM_API: AP CH: 11
I (49546) SCM_API: AP RSSI: -28
I (49547) SCM_API: AP Country : AA
I (49547) SCM_API: Status: CONNECTED
I (49568) SCM_CLI: WIFI GOT IP

$
$ mdns
Unknown interface ·
Usage: mdns init
  or:  mdns ifname start
  or:  mdns ifname stop
  or:  mdns ifname service add <name> <service> <proto> <port>
  or:  mdns ifname service del <id>
  or:  mdns ifname search browse
  or:  mdns ifname search info <service> <proto>
  or:  mdns ifname search stop <id>
$
$ mdns init
$ mdns wlan0 start
$
$
$ mdns wlan0 service add smart_bulb _http _tcp 1919
service_id:0
$
$
```

Now, mDNS responder will answer queries from neighboring devices on the same network.
To see this, avahi-browse can be used from a Linux PC connected to the same local network with SCM1612 EVB as follows.

```
thomas@thomas-900X3C-900X3D-900X3E-900X4C-900X4D:~$ avahi-browse -d local -a -r
+ wlp1s0 IPv4 smart_bulb                                 Web Site             local
= wlp1s0 IPv4 smart_bulb                                 Web Site             local
   hostname = [wise-wlan0.local]
   address = [192.168.51.66]
   port = [1919]
   txt = ["owner=thomas"]
+ enxf8e43bb23ad3 IPv4 RT-AX86U-55A0                                 _alexa._tcp          local
= enxf8e43bb23ad3 IPv4 RT-AX86U-55A0                                 _alexa._tcp          local
   hostname = [RT-AX86U-55A0.local]
   address = [192.168.50.1]
   port = [80]
   txt = ["version=1" "skillSetupId=8b18386c-1353-4612-9626-714937decf3e"]
+ wlp1s0 IPv4 RT-AX53U-88A0                               _alexa._tcp          local
= wlp1s0 IPv4 RT-AX53U-88A0                               _alexa._tcp          local
   hostname = [RT-AX53U-88A0.local]
   address = [192.168.51.1]
   port = [80]
   txt = ["version=1" "skillSetupId=8b18386c-1353-4612-9626-714937decf3e"]
```

Since the mDNS responder has been started, its local host name can be used by any peer device on the same local network such as this Linux PC as follows.

```
thomas@thomas-900X3C-900X3D-900X3E-900X4C-900X4D:~$ avahi-browse -d local -a -r
+ enxf8e43bb23ad3 IPv4 RT-AX86U-55A0                                 _alexa._tcp          local
+ wlp1s0 IPv4 RT-AX53U-88A0                               _alexa._tcp          local
+ wlp1s0 IPv4 smart_bulb                                 Web Site             local
= wlp1s0 IPv4 smart_bulb                                 Web Site             local
   hostname = [wise-wlan0.local]
   address = [192.168.51.66]
   port = [1919]
   txt = ["owner=thomas"]
= enxf8e43bb23ad3 IPv4 RT-AX86U-55A0                                 _alexa._tcp          local
   hostname = [RT-AX86U-55A0.local]
   address = [192.168.50.1]
   port = [80]
   txt = ["version=1" "skillSetupId=8b18386c-1353-4612-9626-714937decf3e"]
= wlp1s0 IPv4 RT-AX53U-88A0                               _alexa._tcp          local
   hostname = [RT-AX53U-88A0.local]
   address = [192.168.51.1]
   port = [80]
   txt = ["version=1" "skillSetupId=8b18386c-1353-4612-9626-714937decf3e"]
^CGot SIGINT, quitting.
thomas@thomas-900X3C-900X3D-900X3E-900X4C-900X4D:~$
thomas@thomas-900X3C-900X3D-900X3E-900X4C-900X4D:~$
thomas@thomas-900X3C-900X3D-900X3E-900X4C-900X4D:~$ ping wise-wlan0.local
PING wise-wlan0.local (192.168.51.66) 56(84) bytes of data.
64 bytes from 192.168.51.66 (192.168.51.66): icmp_seq=2 ttl=255 time=19.8 ms
64 bytes from 192.168.51.66 (192.168.51.66): icmp_seq=3 ttl=255 time=5.25 ms
64 bytes from 192.168.51.66 (192.168.51.66): icmp_seq=4 ttl=255 time=5.23 ms
64 bytes from 192.168.51.66 (192.168.51.66): icmp_seq=5 ttl=255 time=6.09 ms
64 bytes from 192.168.51.66 (192.168.51.66): icmp_seq=6 ttl=255 time=6.66 ms
64 bytes from 192.168.51.66 (192.168.51.66): icmp_seq=7 ttl=255 time=8.38 ms
^C
--- wise-wlan0.local ping statistics ---
7 packets transmitted, 6 received, 14.2857% packet loss, time 6023ms
rtt min/avg/max/mdev = 5.230/8.574/19.831/5.144 ms
thomas@thomas-900X3C-900X3D-900X3E-900X4C-900X4D:~$
```

‘wise-wlan0’ is the hostname that has been given to mdns_resp_add_netif function by a CLI command. It can be changed to any proper name in an user application.