

Hardware Abstraction Layer (HAL)

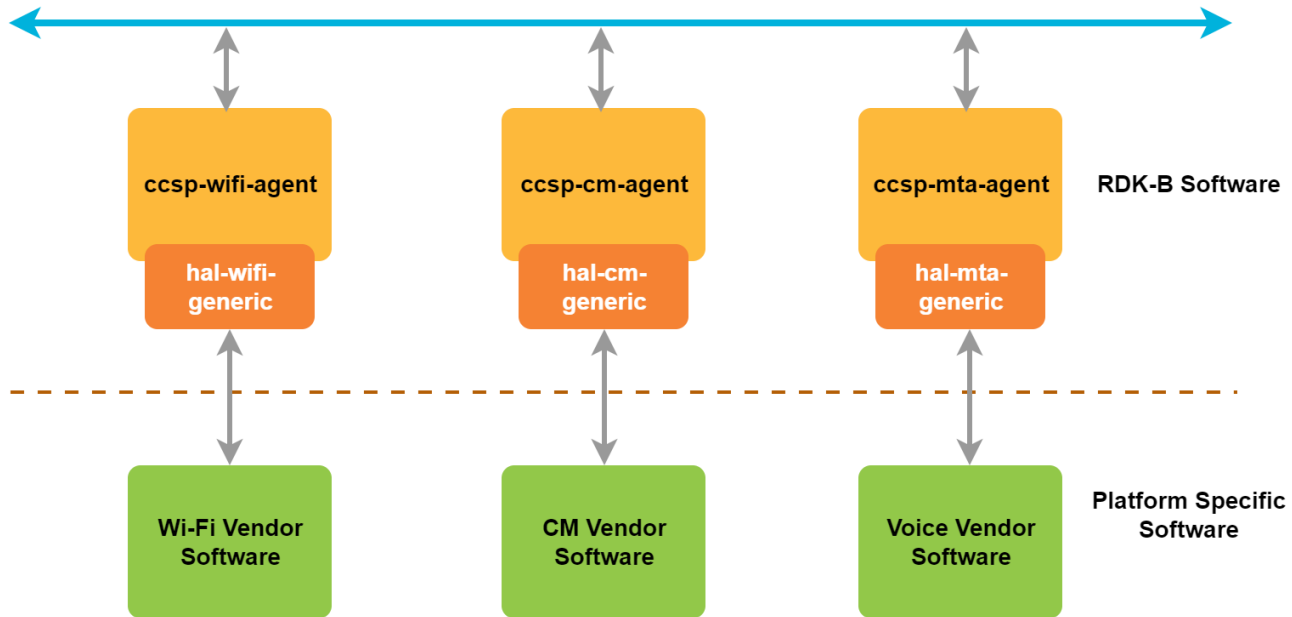
Introduction

RDK-B components are designed to avoid platform or silicon dependencies. Hardware Abstraction Layer (HAL) defines a standard interface for hardware vendors to implement. The HAL layer abstracts the underlying hardware like MOCA, Wi-Fi, etc. through a standard set of APIs defined as part of RDK-B HAL for the respective components. This HAL layer is implemented per platform and the rest of the components can be compiled to run on the new platform without major modifications.

The [HAL](#) in RDK-B Architecture section gives an overview of CCSP framework's Hardware Abstraction Layer.

HAL can be common-HAL or component-specific-HAL. Components may define a component specific HAL to hardware drivers, that are only used by that component

CCSP Message Bus



Component Specific HAL

- HAL APIs will be available in the CMF repo path: `../rdkb/components/opensource/ccsp/hal/source/`
- PandM HAL Integration (back-end) Layer is also known as component specific HAL.
- This layer makes call to underlying Linux system calls/commands, third party modules, open source modules and other CCSP components to execute the requests.
- This layer will be more component specific and will be providing APIs to CCSP so as to manage a particular hardware module of the system.
- Following are some of the component specific HALs available in `../rdkb/components/opensource/ccsp/hal/source/` path.
 - Wifi
 - MoCA
 - MTA Agent
 - CM
 - Ethernet Switch
 - DHCPv4C
 - Virtual LAN
 - Firewall
 - DPoE
 - Bluetooth
 - MSO_Management
 - Voice
 - WAN
 - [TR69_TLV](#)

Wi-Fi HAL

All HAL functions prototypes and structure definitions are available in `wifi_hal.h` file.

- Wi-fi HAL is used for the RDK-Broadband Wifi radio hardware abstraction layer.
- Latest version of RDKB supports 300+ Wi-Fi HAL API's.
- Based on how Wi-Fi vendor exposes their driver capabilities in user space, the HAL API's can be implemented in `wifi_hal.c`
- Some of the APIs are :

1. wifi_getRadioChannelStats
2. wifi_getRadioChannelStats2
3. wifi_getApAssociatedDeviceRxStatsResult
4. wifi_getApAssociatedDeviceTxStatsResult
5. wifi_getApAssociatedDeviceTidStatsResult
6. wifi_getApAssociatedDeviceStats
7. wifi_getHalVersion
8. wifi_factoryReset
9. wifi_factoryResetRadios
10. wifi_factoryResetRadio
11. wifi_setLED
12. wifi_init
13. wifi_reset
14. wifi_down
15. wifi_createInitialConfigFiles
16. wifi_getRadioCountryCode
17. wifi_setRadioCountryCode
18. wifi_pushCountryCode
19. wifi_getATMCApable
20. wifi_setATMCApable

- To see the API specification of Wi-Fi HAL please refer - [Wi-Fi HAL APIs](#)

MOCA HAL

All HAL functions prototypes and structure definitions are available in moca_hal.h file.

- MoCA HAL is used for the RDK-Broadband MoCA hardware abstraction layer.
- An abstraction layer, mainly for interacting with MoCA driver.
- The APIs are :
 1. moca_GetIfConfig
 2. moca_SetIfConfig
 3. moca_GetDynamicInfo
 4. moca_GetStaticInfo
 5. moca_GetStats
 6. moca_GetNumAssociatedDevices
 7. moca_GetExtCounter
 8. moca_GetExtAggrCounter
 9. moca_GetMocaCPEs
 10. moca_GetAssociatedDevices
 11. moca_FreqMaskToValue
 12. moca_HardwareEquipped
 13. moca_GetFullMeshRates
 14. moca_GetFlowStatistics
 15. moca_GetResetCount
 16. moca_setIfAcaConfig
 17. moca_getIfAcaConfig
 18. moca_cancelIfAca
 19. moca_getIfAcaStatus
 20. moca_getIfScmod
- To see the API specification of MoCA HAL please refer - [MoCA HAL APIs](#)

MTA HAL

All HAL functions prototypes and structure definitions are available in mta_hal.h file. An MTA can deliver Home Phone service in addition to High Speed Internet.

- MTA HAL used for the RDK-Broadband hardware abstraction layer for Cable Modem.
- An abstraction layer, implemented to interact with MTA device.
- mta_hal.c file provides the function call prototypes and structure definitions used for the MTA hardware abstraction layer.
- Some of the APIs are :
 1. mta_hal_InitDB
 2. mta_hal_GetDHCPInfo
 3. mta_hal_LineTableGetNumberOfEntries
 4. mta_hal_LineTableGetEntry
 5. mta_hal_TriggerDiagnostics
 6. mta_hal_GetServiceFlow
 7. mta_hal_DectGetEnable
 8. mta_hal_DectSetEnable
 9. mta_hal_DectGetRegistrationMode
 10. mta_hal_DectSetRegistrationMode
 11. mta_hal_DectDeregisterDectHandset
 12. mta_hal_GetCalls
 13. mta_hal_GetDect
 14. mta_hal_GetDectPIN
 15. mta_hal_SetDectPIN
 16. mta_hal_GetHandsets
 17. mta_hal_GetCALLP
 18. mta_hal_GetDSXLogs

- 19. mta_hal_GetDSXLogEnable
- 20. mta_hal_SetDSXLogEnable
- To see the API specification of MTA HAL please refer - [MTA HAL APIs](#)

CM HAL

All HAL functions prototypes and structure definitions are available in cm_hal.h file.

- CM HAL is used for the RDK-Broadband hardware abstraction layer for Cable Modem.
- It provides interface that cable modem software developers can use to interface to RDK-B.
- Some of the APIs are :
 1. cm_hal_InitDB
 2. docsis_InitDS
 3. docsis_InitUS
 4. docsis_getCMStatus
 5. docsis_GetDSChannel
 6. docsis_GetUsStatus
 7. docsis_GetUSChannel
 8. docsis_GetDOCSISInfo
 9. docsis_GetNumOfActiveTxChannels
 10. docsis_GetNumOfActiveRxChannels
 11. docsis_GetErrorCodewords
 12. docsis_SetMddlpModeOverride
 13. docsis_GetMddlpModeOverride
 14. docsis_GetUSChannelId
 15. docsis_SetUSChannelId
 16. docsis_GetDownFreq
 17. docsis_SetStartFreq
 18. docsis_GetDocsisEventLogItems
 19. cm_hal_GetDHCPInfo
 20. cm_hal_GetCPEList
- To see the API specification of CM HAL please refer - [CM HAL APIs](#)

Ethernet Switch HAL

All HAL functions prototypes and structure definitions are available in ccsp_hal_ethsw.h file.

- It provides implementation for Ethernet Switch Control.
- Based on how vendor exposes their driver capabilities in user space, the HAL API's can be implemented in hal-ethsw-generic/git/source/ethsw/ccsp_hal_ethsw.c
- The APIs are :
 1. CcspHalEthSwInit
 2. CcspHalEthSwGetPortStatus
 3. CcspHalEthSwGetPortCfg
 4. CcspHalEthSwSetPortCfg
 5. CcspHalEthSwGetPortAdminStatus
 6. CcspHalEthSwSetPortAdminStatus
 7. CcspHalEthSwSetAgingSpeed
 8. CcspHalEthSwLocatePortByMacAddress
 9. CcspHalExtSw_getAssociatedDevice
 10. CcspHalExtSw_ethAssociatedDevice_callback_register
 11. CcspHalExtSw_getEthWanEnable
 12. CcspHalExtSw_setEthWanEnable
 13. CcspHalExtSw_getEthWanPort
 14. CcspHalExtSw_setEthWanPort
 15. GWP_RegisterEthWan_Callback
 16. GWP_GetEthWanLinkStatus
 17. GWP_GetEthWanInterfaceName
- To see the API specification of Ethernet Switch HAL please refer - [Ethernet Switch HAL APIs](#)

DHCPv4C HAL

All HAL functions prototypes and structure definitions are available in dhcpv4c_api.h file.

- DHCPv4C HAL is used for the RDK-B DHCPv4 Client Status abstraction layer.
- DHCPv4C HAL API's functionality should be implemented by OEMs.
- dhcpv4c_api.c provides the function call prototypes and structure definitions used for the RDK-Broadband DHCPv4 Client Status abstraction layer.
- Some of the APIs are :
 1. dhcpv4c_get_ert_lease_time
 2. dhcpv4c_get_ert_remain_lease_time
 3. dhcpv4c_get_ert_remain_renew_time
 4. dhcpv4c_get_ert_remain_rebind_time
 5. dhcpv4c_get_ert_config_attempts
 6. dhcpv4c_get_ert_ifname
 7. dhcpv4c_get_ert_fsm_state
 8. dhcpv4c_get_ert_ip_addr

9. dhcpv4c_get_ert_mask
10. dhcpv4c_get_ert_gw
11. dhcpv4c_get_ert_dns_svrs
12. dhcpv4c_get_ert_dhcp_svr
13. dhcpv4c_get_ecm_lease_time
14. dhcpv4c_get_ecm_remain_lease_time
15. dhcpv4c_get_ecm_remain_renew_time
16. dhcpv4c_get_ecm_remain_rebind_time
17. dhcpv4c_get_ecm_config_attempts
18. dhcpv4c_get_ecm_ifname
19. dhcpv4c_get_ecm_fsm_state
20. dhcpv4c_get_ecm_ip_addr

- To see the API specification of DHCPv4C HAL please refer - [DHCPv4C HAL APIs](#)

VLAN HAL

All HAL functions prototypes and structure definitions are available in vlan_hal.h file.

- VLAN HAL is or the RDK-B Broadband VLAN abstraction layer.
- VLAN HAL layer is intended to support VLAN drivers through the System Calls.
- The APIs are :
 1. vlan_hal_addGroup
 2. vlan_hal_delGroup
 3. vlan_hal_addInterface
 4. vlan_hal_delInterface
 5. vlan_hal_printGroup
 6. vlan_hal_printAllGroup
 7. vlan_hal_delete_all_Interfaces
 8. _is_this_group_available_in_linux_bridge
 9. _is_this_interface_available_in_linux_bridge
 10. _is_this_interface_available_in_given_linux_bridge
 11. _get_shell_outputbuffer
 12. insert_VLAN_ConfigEntry
 13. delete_VLAN_ConfigEntry
 14. get_vlanId_for_GroupName
 15. print_all_vlanId_Configuration
- To see the API specification of VLAN HAL please refer - [VLAN HAL APIs](#)

Firewall HAL

All HAL functions prototypes and structure definitions are available in hal_firewall.h file.

- This module is responsible for setting firewall rules like port forwarding, port triggering Parental control etc.
- Some of the APIs are :
 1. firewall_service_init
 2. firewall_service_start
 3. firewall_service_restart
 4. firewall_service_stop
 5. firewall_service_close
 6. GetHttpPortValue
 7. Wan2lan_log_deletion_setup
 8. Wan2lan_log_insertion_setup
 9. GettingWanIP_remotemgmt_deletion_logsetup
 10. GettingWanIP_remotemgmt_insertion_logsetup
 11. DeleteRemoteManagementIptablesRules
 12. AddRemoteManagementIptablesRules
 13. DisablingHttps
 14. EnablingHttps
 15. DisablingHttp
 16. EnablingHttp
 17. SetHttpPort
 18. SetHttpsPort
 19. RemoteManagementiptableRulesetoperation
 20. BasicRouting_Wan2Lan_SetupConnection
- To see the API specification of Firewall HAL please refer - [Firewall HAL APIs](#)

DPOE HAL

All HAL functions prototypes and structure definitions are available in dpoe_hal.h file.

- DPOE HAL is used for the RDK-Broadband DPoE hardware abstraction layer as per the DPoE-SP-OAMv1.0-I08-140807 specification.
- Some of the APIs are :
 1. dpoe_getOnuld
 2. dpoe_getFirmwareInfo
 3. dpoe_getEponChipInfo
 4. dpoe_getManufacturerInfo

5. dpoe_getNumberOfNetworkPorts
6. dpoe_getNumberOfS1Interfaces
7. dpoe_getOnuPacketBufferCapabilities
8. dpoe_getOamFrameRate
9. dpoe_getLidForwardingState
10. dpoe_getDeviceSysDescrInfo
11. dpoe_getMaxLogicalLinks
12. dpoe_setResetOnu
13. dpoe_getStaticMacTable
14. dpoe_getEponMode
15. dpoe_getDynamicMacAddressAgeLimit
16. dpoe_getDynamicMacLearningTableSize
17. dpoe_getDynamicMacTable
18. dpoe_getStaticMacTable
19. dpoe_getMacLearningAggregateLimit
20. dpoe_getOnuLinkStatistics

- To see the API specification of DPOE HAL please refer - [DPOE HAL APIs](#)

Bluetooth HAL

All HAL functions prototypes and structure definitions are available in bt_hal.h file.

- The APIs are :
 1. ble_Enable
 2. ble_GetStatus
- To see the API specification of Bluetooth HAL please refer - [Bluetooth HAL APIs](#)

MSO Management HAL

All HAL functions prototypes and structure definitions are available in mso_mgmt_hal.h file.

- MSO Management HAL is used for the RDK-Broadband hardware abstraction layer for MSO Management.
- The APIs are :
 1. mso_pwd_ret_status mso_validatepwd
 2. mso_set_pod_seed
 3. mso_get_pod_seed
- To see the API specification of MSO Management HAL please refer - [MSO Management HAL APIs](#)

Voice HAL

All HAL functions prototypes and structure definitions are available in voice_hal.h file.

- Voice HAL is used for the RDK-Broadband hardware abstraction layer for VoIP.
- Some of the APIs are :
 1. voice_hal_Init
 2. voice_hal_InitDB
 3. voice_hal_Deinit
 4. voice_hal_DeinitDB
 5. voice_hal_setVoiceProcessState
 6. voice_hal_getVoiceProcessState
 7. voice_hal_getVoiceProcessStatus
 8. voice_hal_getConfigSoftwareVersion
 9. voice_hal_getCountProfiles
 10. voice_hal_getServiceVersion
 11. voice_hal_getCountServices
 12. voice_hal_getCountLines
 13. voice_hal_getCountPhyInterfaces
 14. voice_hal_setIpAddressFamily
 15. voice_hal_getBoundIfName
 16. voice_hal_setBoundIfName
 17. voice_hal_setIpAddressFamily
 18. voice_hal_getIpAddressFamily
 19. voice_hal_setLinkState
 20. voice_hal_setIpWanAddress
- To see the API specification of Voice HAL please refer - [Voice HAL APIs](#)

WAN HAL

All HAL functions prototypes and structure definitions are available in wan_hal.h file.

- The APIs are :
 1. wan_hal_Init
 2. wan_hal_SetSelfHealConfig
 3. wan_hal_SetWanConnectionEnable
 4. wan_hal_SetSelfHealConfig
 5. wan_hal_GetWanOEUpstreamCurrRate

- 6. wan_hal_GetWanOEDownstreamCurrRate
- 7. wan_hal_SetQoSConfiguration
- 8. wan_hal_RestartWanService
- To see the API specification of WAN HAL please refer - [WAN HAL APIs](#)

TR69_TLV HAL

All HAL functions prototypes and structure definitions are available in Tr69_Tlv.h file.

- Telemetry Key fields and data fields are stored in the database as TLV (Tag, Length, Value)
 1. Tag - uniquely identifies the field.
 2. Length - gives the size (in number of bytes) of the data associated with the field.
 3. Value - contains the actual data associated with the field stored in network byte ordering.

Common HAL

- A common HAL provides the necessary abstraction to all the CCSP components to interface with other common hardware components.
- Eg : Platform HAL

Platform HAL

- Platform HAL is an abstraction layer, implemented to interact with cable modem device for getting the hardware specific details such as Firmware Name, Boot loader Version, etc.
- This HAL layer is intended to support platform drivers
- platform_hal.c file provides the function call prototypes and structure definitions used for the platform hardware abstraction layer
- Some of the APIs are :

1. platform_hal_GetDeviceConfigStatus
2. platform_hal_GetTelnetEnable
3. platform_hal_GetSSHEnable
4. platform_hal_SetSSHEnable
5. platform_hal_GetSNMPEnable
6. platform_hal_SetSNMPEnable
7. platform_hal_GetSerialNumber
8. platform_hal_GetWebUITimeout
9. platform_hal_SetWebUITimeout
10. platform_hal_GetWebAccessLevel
11. platform_hal_SetWebAccessLevel
12. platform_hal_PandMDBInit
13. platform_hal_DocsisParamsDBInit
14. platform_hal_GetModelName
15. platform_hal_GetFirmwareName
16. platform_hal_GetHardwareVersion
17. platform_hal_GetSoftwareVersion
18. platform_hal_GetBootloaderVersion
19. platform_hal_GetBaseMacAddress
20. platform_hal_GetHardware

To see the API specification of Platform HAL please refer - [Platform HAL APIs](#)